

B 6000 Series

SYSTEM NOTES

RELATIVE TO MARK 3.2 RELEASE

Copyright © 1980, Burroughs Corporation, Detroit Michigan 48232

PRICED ITEM

Burroughs believes that the information described in this manual is accurate and reliable, and much care has been taken in its preparation. However, no responsibility, financial or otherwise, is accepted for any consequences arising out of the use of this material. The information contained herein is subject to change. Revisions may be issued to advise of such changes and/or additions.

Correspondence regarding this document should be addressed directly to Burroughs Corporation, P.O. Box 4040, El Monte, California 91734, Attn: Publications Department, TIO--West.

TABLE OF CONTENTS

GENERAL		PAGE	1
D2782	- KIND=DISK Vs. FAMILYNAME	PAGE	1
D2981	- Mark Level Documentation	PAGE	1
D3090	- Compiler Info Word in Seg Zero	PAGE	1
D3104	- XALGOL Deimplemented	PAGE	1
D3175	- Mark 31 System Notes Corrections	PAGE	1
D3193	- Installation of Mark 32 Software	PAGE	5
D3201	- CONTROLWARE Files	PAGE	6
D3203	- B6900 Overview	PAGE	6
D3205	- Instructions for Printing Documentation	PAGE	10
D3206	- SYSTEMNOTES Tape	PAGE	10
D3207	- Mark 32 Release Tapes	PAGE	11
D3257	- IVR Facility on Mark 33 Release	PAGE	11
D3285	- Deimplementation of ESPOL Compiler	PAGE	12
D3311	- ORGHOSTNAME Attribute Deimplementation	PAGE	12
D3329	- MYUSE=10 Vs. UPDATEFILE Attributes	PAGE	12
D3341	- Support of Old Codefiles	PAGE	12
D3348	- FILETYPE=5 Files	PAGE	12
D3354	- Intrinsic to Library Conversion	PAGE	13
D3356	- New and Old ODT Messages	PAGE	16
D3396	- LISTNOTES Changes	PAGE	77
D3572	- PBIT Time Accounting	PAGE	79
D3650	- Implementation of Port Files	PAGE	79
GENERAL		PAGE	86
P3273	- Copyright, Version Levels Updated	PAGE	86
ALGOL		PAGE	87
D2837	- Compile Time Extension to "<define decl>"	PAGE	87
D3000	- \$BEGINSEGMENT Not For Use Around Blocks	PAGE	87
D3001	- Global WFL Files in \$INCLUDE Option	PAGE	87
D3009	- STRING Expressions in WRITE and REPLACE	PAGE	88
D3062	- Longer Strings with Implicit Concatenation	PAGE	88
D3063	- Events, Event Arrays as Library Parameters	PAGE	88
D3211	- Give Warning for INTMODE=BCL	PAGE	89
D3266	- Allow LONG As Key Word	PAGE	89
D3269	- Binary I/O for Strings	PAGE	89
D3349	- Set \$NOBINDINFO	PAGE	90
D3350	- Flag \$NOBINDINFO	PAGE	90
D3360	- Modifications to Support Portfiles	PAGE	90
D3363	- Remove SIGNAL and RESPONSE	PAGE	91
D3433	- Array Row Equivalence to Own Arrays	PAGE	92
D3434	- SDIGITS in REPLACE Statement	PAGE	92
D3435	- Replace Pointer Valued File Attribute	PAGE	92
D3436	- "<formal parameter specifier>"	PAGE	92
D3437	- MESSAGE IN "<DISPLAY STATEMENT>"	PAGE	92
D3438	- "[NO]" Has No Effect on Remote Files	PAGE	92
D3439	- R is a "<single picture character>"	PAGE	93
D3455	- Clarification of DSCALELEFT Function	PAGE	93
D3471	- Allow Longer Value Arrays	PAGE	93
D3490	- Clarification of \$MERGE	PAGE	93
D3522	- Replace Into Double Precision Array	PAGE	93
D3526	- "<picture declaration>"	PAGE	94
D3530	- FUNCTIONNAME, LIBACCESS Attributes	PAGE	94
D3537	- Maximum Size of Switch Format	PAGE	94
D3563	- No Resize/Deallocate of Segmented Array	PAGE	94
D3578	- Legal Input to Format	PAGE	94
D3610	- Passing Files by Reference to Libraries	PAGE	94
D3623	- OFFSET and DELTA	PAGE	94
D3626	- Resizing EVENT ARRAYS	PAGE	95
D3633	- "REAL (<pointer expression>)"	PAGE	95
D3654	- "REAL (<pointer exp>, <arithmetic exp>)"	PAGE	95
D3657	- NOBCL Compiler Option	PAGE	95
ALGOL		PAGE	96
P2724	- Strings as Attributes, TR Items	PAGE	96
P2903	- Internal Arrays Expanded	PAGE	96
P2916	- Code Optimization Corrected	PAGE	96
P3037	- Incorrect Sign for Complex Expression	PAGE	96
P3038	- Invalid Assignments Not Flagged	PAGE	96
P3039	- Complex Expressions	PAGE	96
P3040	- Erroneous Syntax Error with Complex	PAGE	96
P3041	- Incorrect Software Control Word	PAGE	97
P3091	- INVALID OP with Long Export List	PAGE	97
P3092	- INVALID INDEX with Lex Levels > 15	PAGE	97
P3139	- Correct XREF Output for Library Procedures	PAGE	97
P3154	- INVALID OP, Indexed String Array	PAGE	97
P3155	- INVALID OP for String Expressions	PAGE	97
P3324	- Multiple ELSE Clauses in CASE Statement	PAGE	97
P3406	- Prevent INVALID INDEX	PAGE	97
P3407	- Call BLOCKEXIT to Deallocate Bound Globals	PAGE	97
P3408	- INVALID INDEX	PAGE	97
P3409	- ACCEPT "<string variable>"	PAGE	98
P3410	- Making Use of Available Space	PAGE	98
P3411	- Intrinsic as a Name Parameter	PAGE	98

P3412	- Picture as a Formal Parameter.	PAGE	98
P3413	- CTPROC, CTDEFINE Vs. Parameter List.	PAGE	98
P3414	- Complex Times Real Multiplication.	PAGE	98
P3415	- Clear SCW Information.	PAGE	98
P3416	- String Pool Exceeded with SINTRINSICS.	PAGE	98
P3417	- Flag BCL Pointers with Offset.	PAGE	98
P3418	- Long Character Arrays.	PAGE	98
P3463	- Calling USERIOERROR for "MYSELF.TASKFILE".	PAGE	98
P3498	- INVALID INDEX after Parameter Mismatch.	PAGE	98
P3526	- Prevent Possible Stack Overflow.	PAGE	98
P3532	- Call Resetpoolstringsize.	PAGE	99
P3533	- Corruption of Value Arrays.	PAGE	99
P3560	- BCL Constructs Removed.	PAGE	99
P3611	- Global STRING PROCEDURE and Binding.	PAGE	99
P3625	- Invalid I/O List Elements.	PAGE	99
P3661	- Give Error for Spaces within Numbers.	PAGE	99
P3774	- Missing BLOCKEXIT.	PAGE	99
P3798	- Locking Code File.	PAGE	99
ALGOL INTRINSICS.		PAGE	100
P2940	- Backup File Searching.	PAGE	100
P3298	- CTOD Terminates Abnormally.	PAGE	100
P3325	- Correct DSQRT Errors.	PAGE	100
P3326	- Correct GAMMA, DGAMMA.	PAGE	100
ALGOL/PLI INTRINSICS.		PAGE	101
D3628	- ALGOLPLINTRN Subsumed by GENERALSUPPORT.	PAGE	101
ATTABLEGEN.		PAGE	102
D3076	- APL File Attribute.	PAGE	102
D3425	- Delete PORTS, SIGNALS.	PAGE	102
D3656	- LASTSTATION Synonym "= LASTSUBFILE".	PAGE	102
ATTABLEGEN.		PAGE	103
P3626	- "ROWSIZE=504" for New Patch.	PAGE	103
BACKUP.		PAGE	104
D3587	- HOSTNAME Modifier.	PAGE	104
BACKUP.		PAGE	105
P3327	- BFILE Label Equation.	PAGE	105
P3658	- "FILE.TITLE" Attribute.	PAGE	105
BARS.		PAGE	106
D2978	- SYSTEM/BARS Utility.	PAGE	106
D3277	- Virtual Memory Utilization Measure.	PAGE	107
BARS.		PAGE	108
P2909	- Processor Times Reported on Monolithic Systems.	PAGE	108
P2910	- Correct Swapcore Graphs.	PAGE	108
P3110	- Incomplete Display on System ODT.	PAGE	108
P3251	- Clear Channel Indicators Properly.	PAGE	108
P3464	- SCREEN File Attribute.	PAGE	108
P3481	- Changes to Type 4 SYSTEMSTATUS Call.	PAGE	108
P3642	- "<more>" Displayed Completely.	PAGE	108
P3643	- SPO Mode Displays Long Messages.	PAGE	108
P3644	- Negative IDLETIME.	PAGE	108
P3645	- Single "." as Input.	PAGE	108
BASIC		PAGE	109
D3538	- Matrix Inversion Function.	PAGE	109
D3539	- APPEND Statement.	PAGE	109
D3540	- "Up-Arrows" Required for Exponential Field.	PAGE	109
D3603	- Program of Up to 2048 Statements.	PAGE	109
BASIC		PAGE	110
P3612	- Error for DEF Function.	PAGE	110
P3622	- Flag Question Mark as Invalid Character.	PAGE	110
P3664	- Error on First Program Token.	PAGE	110
BINDER.		PAGE	111
D3002	- Code File Naming Convention for ALGOL.	PAGE	111
D3414	- Delete Old Intrinsics.	PAGE	111
D3487	- MCP Code File Row Size = 504.	PAGE	111
D3495	- Installation Intrinsic Warning.	PAGE	111
D3562	- "Inter-Language" Procedure Parameters.	PAGE	111
D3566	- Array Bound Clarification.	PAGE	112
D3577	- MCP BIND Example.	PAGE	112
BINDER.		PAGE	113
P3010	- Multiple Rebinds of SYSTEM/INTRINSICS.	PAGE	113
P3328	- Binding FORTRAN Routines.	PAGE	113
P3360	- Correct \$WAIT with <mix no.> OF.	PAGE	113
P3419	- Binding with "D[0]" Intrinsics.	PAGE	113
P3587	- Binding Programs with \$DATADICTINFO.	PAGE	113
CANDE		PAGE	114
D3249	- VISIBILITY Task, SCATTER Run-Time Options.	PAGE	114
D3364	- Automatic DESTNAME for CANDE Sessions.	PAGE	115
D3370	- Full Screen Sequence Mode.	PAGE	116
D3456	- FILES <FILENAME> : <DEPTH #>.	PAGE	116
D3544	- COPY/ADD Statement.	PAGE	116
D3580	- Passing Strings Via WFL/CANDE.	PAGE	116
D3642	- CANDE Vs. Foreign Tasks.	PAGE	117
D3646	- Compiler Type FORTRAN77.	PAGE	117
CANDE		PAGE	118
P2798	- SCHEDULE Restart Problems.	PAGE	118
P2833	- Missing SCHEDULE Output File.	PAGE	118
P3200	- Schedule Request on Unnamed Workfile.	PAGE	118
P3329	- Missing SCHOUT File.	PAGE	118

P3420	- Secure Schedule Files Properly	PAGE 118
P3421	- Allow Setting Chargecode to Null	PAGE 118
P3422	- TAPE Command in DO Files	PAGE 118
P3423	- Recognize NDL Sequence Mode Terminals	PAGE 118
P3424	- Security Problem	PAGE 118
P3425	- CANDE Creates Bad Tankfile	PAGE 118
P3426	- Schedule Sessions Vs. Chargecodes	PAGE 118
P3460	- CANDE Errors Now Attributed to User Errors	PAGE 118
P3499	- Allow More than 2 Digits	PAGE 119
P3500	- CANDE DS for Security Violation	PAGE 119
P3501	- Packname with Leading Digit	PAGE 119
P3535	- Backupprocessor Finding End of Block	PAGE 119
P3536	- EOL Character in DO Statement	PAGE 119
P3571	- Folding Lower Case Tokens	PAGE 119
P3574	- SEG ARRAY Error in DCWER	PAGE 119
P3613	- Station Vs. Terminal Settings	PAGE 119
P3691	- Handling Line Errors	PAGE 119
P3692	- END JOB on WFL Statement	PAGE 119
P3785	- Allow "4-Character" Verbs	PAGE 119
COBOL		PAGE 120
D3003	- Binding and Statistics	PAGE 120
D3223	- BCL Warnings	PAGE 120
D3359	- Modifications to Support Port Files	PAGE 120
D3449	- Ignore Nonfunctional Characters	PAGE 121
D3473	- Editing the Value Zero	PAGE 121
D3491	- TIME Function in DIVIDE Statement	PAGE 122
D3503	- Option 4, MOVE Verb	PAGE 122
D3504	- VALUE Clause for "COMP-4,COMP-5" Items	PAGE 122
D3505	- OPEN EXTEND on Variable Length File	PAGE 122
D3510	- Limit on SIZE Clause	PAGE 122
D3517	- Passing Strings from WFL to COBOL	PAGE 122
D3520	- IPC, USE AS EXTERNAL PROCEDURE Statement	PAGE 122
D3521	- Error Limit of 150	PAGE 122
D3523	- Units=Characters	PAGE 122
D3524	- Referencing Two or Three Dimensional Arrays	PAGE 122
D3541	- Close on a Disk File	PAGE 122
D3570	- RECORD AREA	PAGE 123
D3648	- TB Reserved Word	PAGE 123
D3658	- Parameter Passing	PAGE 123
COBOL		PAGE 124
P2701	- INVALID INDEX in COBOL Compiler	PAGE 124
P2702	- USE Routine Not Invoked	PAGE 124
P2703	- VALUE(TERMINATED)	PAGE 124
P2704	- WAIT Statement	PAGE 124
P2705	- IF Statement Generates Bad Code	PAGE 124
P2706	- Results of Exponentiation Improved	PAGE 124
P2707	- Bindinfo for 77 COMP Global Item	PAGE 124
P2708	- Invalid Syntax for File Attributes	PAGE 124
P2727	- Compilation Summary	PAGE 124
P2728	- INVALID INDEX	PAGE 124
P2729	- NEWSEQERR \$ Option	PAGE 124
P2732	- Intrinsic Information in Global Directory	PAGE 124
P2813	- "OPEN O-I FILE1 I-O FILE2"	PAGE 125
P2815	- INVALID OP in IF Statement	PAGE 125
P2834	- Interaction of OPEN Statement	PAGE 125
P2835	- STOP RUN	PAGE 125
P2932	- \$ Options SEQERR, NEWSEQERR, SEQCHECK	PAGE 125
P3054	- Call User Intrinsic	PAGE 125
P3055	- INVALID INDEX in Report Writer	PAGE 125
P3080	- Indexed File with Invalid Key Branch	PAGE 125
P3081	- RERUN Clause	PAGE 125
P3205	- Calls on Untyped User Intrinsics	PAGE 125
P3262	- LIBRARY CALL Within IF	PAGE 125
P3274	- Equal Comparisons	PAGE 125
P3275	- Group Computational Moves	PAGE 125
P3276	- Erroneous Syntax Error	PAGE 125
P3294	- Syntax Checking with NEXT SENTENCE Clause	PAGE 126
P3295	- Syntax Error in IF Statement	PAGE 126
P3302	- OBJECT-COMPUTER Clause Syntax	PAGE 126
P3303	- ANSI 74 Default Line Spacing	PAGE 126
P3304	- LINE NUMBER Clause Syntax	PAGE 126
P3305	- MERGE Statement Syntax	PAGE 126
P3306	- Long Conditional Expressions	PAGE 126
P3307	- KEYSERENTRY Greater Than 63	PAGE 126
P3427	- INVALID INDEX	PAGE 126
P3465	- Maximum Number of Libraries Exceeded	PAGE 126
P3466	- Reserved Words Syntaxed in WRITE Statements	PAGE 126
P3467	- INVALID INDEX with CP CALL	PAGE 126
P3502	- Misalignment of "COMP-2" Sync Values	PAGE 127
P3503	- SEGMENT Clause, 01 Record	PAGE 127
P3525	- Timestamp Differences	PAGE 127
P3538	- ISAM CLOSE Options	PAGE 127
P3559	- MONITOR, Write to Same File	PAGE 127
P3561	- ISAM External File Names	PAGE 127
P3562	- COPY with Bad File Title	PAGE 127
P3563	- Operand Left on Top of Stack	PAGE 127
P3581	- "COMP-2" Numeric Test	PAGE 127

P3679 - User Intrinsic at Levels "> 2".	PAGE 127
COBOL74	PAGE 128
D3010 - File Description Entry	PAGE 128
D3222 - Close WITH LOCK.	PAGE 128
D3224 - BCL Warnings	PAGE 128
D3358 - Modifications to Support Port Files.	PAGE 128
D3505 - OPEN EXTEND on Variable Length File.	PAGE 129
D3531 - Error Limit of 150	PAGE 129
D3574 - Data Comm Interface Library.	PAGE 130
D3597 - File Handling Differences.	PAGE 132
D3659 - D Lines, FREE Option	PAGE 132
COBOL74	PAGE 133
P2701 - INVALID INDEX in COBOL Compiler.	PAGE 133
P2703 - VALUE(TERMINATED).	PAGE 133
P2705 - IF Statement Generates Bad Code.	PAGE 133
P2706 - Results of Exponentiation Improved	PAGE 133
P2727 - Compilation Summary.	PAGE 133
P2728 - INVALID INDEX.	PAGE 133
P2800 - SERIALNO Attribute	PAGE 133
P2801 - Non-numeric Attributes	PAGE 133
P2815 - INVALID OP in IF Statement	PAGE 133
P2819 - Close for Multi-File Tapes	PAGE 133
P2835 - STOP RUN	PAGE 133
P2851 - "SOPTIMIZE" Compiler Error	PAGE 133
P2931 - SIGN Clause for Computational Item	PAGE 133
P2932 - \$ Options SEQERR, NEWSEQERR, SEQCHECK.	PAGE 134
P2937 - File Attributes.	PAGE 134
P3055 - INVALID INDEX in Report Writer	PAGE 134
P3060 - WRITE AFTER ADVANCING PAGE Statement	PAGE 134
P3070 - SEQCHECK \$ Option.	PAGE 134
P3081 - RERUN Clause	PAGE 134
P3084 - Debug Line Values.	PAGE 134
P3086 - WRITE Statement with FOOTING Equal to 1.	PAGE 134
P3274 - Equal Comparisons.	PAGE 134
P3294 - Syntax Checking with NEXT SENTENCE Clause.	PAGE 134
P3295 - Syntax Error in IF Statement	PAGE 134
P3308 - Linage - Footing Value of One.	PAGE 134
P3395 - BDMS Federal level Warning	PAGE 134
P3465 - Maximum Number of Libraries Exceeded	PAGE 135
P3525 - Timestamp Differences.	PAGE 135
P3564 - COMPUTATIONAL Numeric Test	PAGE 135
P3826 - Library Pseudo Text Replacement.	PAGE 135
COMPARE	PAGE 136
P3614 - ARRAY TOO LARGE Error.	PAGE 136
CONFIGURATOR.	PAGE 137
D3406 - Soft Configuration	PAGE 137
CONTROLLER.	PAGE 142
D3565 - J, MX Response	PAGE 142
D3647 - TD Acceleration.	PAGE 142
D3651 - Suppress Frozen Libraries.	PAGE 142
CONTROLLER.	PAGE 143
P3045 - Completed Entries Shows Spurious Subsystem	PAGE 143
P3046 - TERM USER Vs. AT HOSTNAME.	PAGE 143
P3099 - "SUBSYSTEM=" Vs. QF Correction.	PAGE 143
P3236 - Very Long File Names	PAGE 143
P3254 - BACKUPQUEUER Call.	PAGE 143
P3352 - Multipage PER Display.	PAGE 143
P3428 - Missing "#" on ODT for Swaptask.	PAGE 143
P3516 - Hour Field in TIMEAT	PAGE 143
P3517 - NS Correction.	PAGE 143
P3659 - AA Correction.	PAGE 143
P3750 - PA Correction.	PAGE 143
CONTROLWARE	PAGE 144
D3191 - Controlware Vs. Firmware, Pack Controls.	PAGE 144
D3200 - Mark 32 Disk Pack Controlware Files.	PAGE 145
DATA COMMUNICATIONS	PAGE 150
D3202 - B6900 Data Communications.	PAGE 150
DATA COMMUNICATIONS	PAGE 152
P2782 - DCRECON Line Result Not Returned	PAGE 152
P2783 - Extended Line TALLYs Problems.	PAGE 152
P2784 - DCSYSTEMTABLES Returns Wrong Result.	PAGE 152
P2795 - READNIF Test for Valid Record Index.	PAGE 152
P2857 - Unlock Line by DCIOFINISH Callers.	PAGE 152
P2960 - RECALLOBJOOUTPUT on Uninitialized DCP	PAGE 152
P3229 - DCSYSTEMTABLES Option 5 SEG ARRAY Fault	PAGE 152
P3677 - DLS Corrupted in Full Duplex Discards.	PAGE 152
P3771 - DCFILELOCK Deadlock.	PAGE 152
DCALGOL	PAGE 153
D3075 - Queues as Library Parameters	PAGE 153
D3357 - Remove PORT and SIGNAL	PAGE 153
D3444 - Qinsertevent Documentation	PAGE 153
DCALGOL INTRINSICS	PAGE 154
D3629 - DCALGOLINTRN Subsumed by GENERALSUPPORT.	PAGE 154
DCAUDITOR	PAGE 155
D3395 - DCAUDITOR Implementation	PAGE 155
D3637 - Add Additional Information to DCAUDIT File	PAGE 155
DCP PROGRAM GENERATOR	PAGE 156

P2802	- Allow TERMINATE NORMAL	PAGE 156
P3497	- Garbled BAUDOT Translate Tables	PAGE 156
P3666	- Consecutive Line Tally Usage	PAGE 156
P3677	- DLS Corrupted in Full Duplex Discards	PAGE 156
P3775	- INCREMENT TRAN Statement	PAGE 156
DCSTATUS		PAGE 157
P3012	- Print Station Table Base Correctly	PAGE 157
P3539	- "17-Character" Names for GRAPH Output	PAGE 157
P3572	- Prevent NIF/DCPCODE File Changes	PAGE 157
P3600	- Calculate Station Table Base Size	PAGE 157
DIAGNOSTIC	MCS	PAGE 158
D3344	- Input Sources	PAGE 158
D3427	- Nonnumeric Dial Characters	PAGE 158
DIAGNOSTIC	MCS	PAGE 159
P3163	- Error in BTB Attach by LSN/DLS	PAGE 159
P3253	- Error in "BTB ALL REPEAT <string>"	PAGE 159
P3601	- Alter Correct Station Address	PAGE 159
P3602	- Prevent Dump on "<DCL>"	PAGE 159
P3603	- Attached CA Option	PAGE 159
DMS II	GENERAL	PAGE 160
D3108	- New Data Base Stack Structure	PAGE 160
D3170	- Shared ACCESSROUTINES, Data Base Equation	PAGE 160
D3198	- Updating from Mark 31 to Mark 32 DMSII	PAGE 172
D3288	- Mark 31 System Notes Corrections	PAGE 173
D3302	- Efficient Use of Index Sequential	PAGE 174
D3314	- Rebuild Across File Discontinuities	PAGE 175
D3333	- Mark 31 System Notes Corrections	PAGE 175
D3337	- Data Base Stack	PAGE 175
D3382	- Auditing for Halt/Load and Abort Recovery	PAGE 175
D3460	- Preallocation of Direct Data Sets	PAGE 177
D3548	- DBS in Local Memory Vs. Nonexchanged Unit	PAGE 178
DMS II	GENERAL	PAGE 179
P3071	- READLOCKNOPURGE Removed	PAGE 179
P3258	- Normal Vs. Direct Files as Parameters	PAGE 179
DMS II	ACCESSROUTINES	PAGE 180
D3044	- Count Finds Against Index Sets	PAGE 180
D3045	- Statistics Interface	PAGE 180
D3046	- Buffers Moved to Data Base Environment	PAGE 185
D3047	- Allow AUDIT CLOSE Message	PAGE 185
D3306	- B7700CODE Option	PAGE 185
D3315	- Forced, Normal Overlays	PAGE 185
D3331	- ACCESSROUTINES Error Messages	PAGE 185
D3338	- Print Statistics Option	PAGE 190
D3366	- Save and Retrieve Messages	PAGE 190
D3452	- Abort Acceleration	PAGE 190
DMS II	ACCESSROUTINES	PAGE 191
P2499	- Efficient Pack Space Utilization	PAGE 191
P2632	- Audit Restart Information Correctly	PAGE 191
P2633	- DS of "SECTORS REQUIRED" Vs. Reconstruction	PAGE 191
P2752	- System Serial Number Added	PAGE 191
P2753	- Remove RSFILE Declaration	PAGE 191
P2754	- Audit File Error Handling	PAGE 191
P2755	- Flush Buffers for Structure	PAGE 191
P2756	- Do Not Set Inuse Flag	PAGE 191
P2757	- Immediate Overlay of Buffers	PAGE 191
P2788	- Divest Compact Table Block if Read Error	PAGE 191
P2882	- ZERO DISK ADDRESS for Direct Data Set	PAGE 191
P2883	- Quick Fix Creates Empty Audit Files	PAGE 192
P2885	- Reuse Empty audit Correctly	PAGE 192
P2886	- RECONSTRUCT Makes Empty Audit Files	PAGE 192
P2948	- Display Reason for Not Reusing Audit	PAGE 192
P2949	- Unlock Partition if DS in Open	PAGE 192
P2950	- INVALID OP With Readahead	PAGE 192
P2964	- Data Base Messages	PAGE 192
P3003	- Bad Available Tables for Compact Data Set	PAGE 192
P3030	- Nested STARTDB Errors	PAGE 192
P3031	- I/O Timeout	PAGE 192
P3033	- Data Base Subsystem Visible	PAGE 192
P3101	- Nested STARTDB	PAGE 192
P3106	- Interface to Free Stack Records	PAGE 192
P3175	- Control File I/O Lock	PAGE 193
P3176	- Fault on Reblocked Standard Data Set	PAGE 193
P3177	- Totalcore Protected by Memlock	PAGE 193
P3178	- Prevent COPYAUDIT Zip Delay	PAGE 193
P3230	- Deleting Variable Format Records	PAGE 193
P3246	- ERROR IN DCB HANDLING	PAGE 193
P3247	- DBSINFO Replaces MYNUMBER	PAGE 193
P3259	- Missing Divest on Deadlock	PAGE 193
P3284	- DCB Handling Error	PAGE 193
P3312	- Bad Links When Open Inquiry	PAGE 194
P3313	- Links on Select Text Error	PAGE 194
P3314	- Invalid INQUIRY Function Numbers Ignored	PAGE 194
P3333	- Store Restart Area	PAGE 194
P3340	- Standard VF Control Word Corrupted	PAGE 194
P3368	- Do Not Point Links at Overflow Block	PAGE 194
P3369	- Missing Divest Following Version Error	PAGE 194
P3370	- Forget Subblock for Ordered Data Set	PAGE 194

P3371	- Zeroed Out Blocks in Data Base	PAGE	194
P3383	- Partition Open Error	PAGE	194
P3393	- Erroneous BIO/AIO Audit Records	PAGE	194
P3553	- Invalid Standard Variable Format Record Types	PAGE	194
P3554	- Partition Audit Records Out of Order	PAGE	195
P3606	- Linear Search with Signed Numeric Keys	PAGE	195
P3618	- Cancel or Complete I/O Following Timeout	PAGE	195
P3683	- Ordered Data Set Divest Error	PAGE	195
P3696	- DMSECURITYERROR	PAGE	195
P3697	- Fail to Divest	PAGE	195
P3708	- Infinite Loop	PAGE	195
P3755	- NOTLOCKED Exception	PAGE	195
P3773	- FORGETSPACE Timing Window	PAGE	195
P3789	- Cannot Locate Compact Record	PAGE	196
DMS II -	ARCHIVEUPDATER	PAGE	197
D3614	- Eliminate OPEN INITIALIZE	PAGE	197
DMS II -	ARCHIVEUPDATER	PAGE	198
P3341	- Prevent Sort Error #4	PAGE	198
DMS II -	BDMSALGOL	PAGE	199
D3324	- Print Data Base Title	PAGE	199
D3440	- Compiler Identification	PAGE	199
D3467	- Implicit Free for FIND, LOCK	PAGE	199
D3552	- Deimplement OPEN INITIALIZE	PAGE	199
DMS II -	BDMSALGOL	PAGE	200
P3296	- Transaction Record Parameters	PAGE	200
DMS II -	BDMSCOBOL	PAGE	201
D3325	- Data Base Equation Information	PAGE	201
D3467	- Implicit Free for FIND, LOCK	PAGE	201
D3555	- OPEN INITIALIZE Deimplemented	PAGE	201
DMS II -	BDMSCOBOL	PAGE	202
P2730	- BDMSCOBOL Generates Bad Print Line	PAGE	202
P2731	- INVALID INDEX	PAGE	202
P3085	- Transaction Item, 1 or 2 Characters	PAGE	202
P3588	- Invalid Header	PAGE	202
P3589	- INVALID INDEX	PAGE	202
P3590	- INVALID INDEX	PAGE	202
P3591	- "DB-INVOKE" Hardly Readable	PAGE	202
P3635	- INVALID OP Accessing Global Data Items	PAGE	202
P3636	- Linear Search Selection Expression	PAGE	202
P3803	- "DUMP PRINTER (<dataset-name>)" Statement	PAGE	202
DMS II -	BDMS/PL/I	PAGE	203
D3467	- Implicit Free for FIND, LOCK	PAGE	203
D3556	- OPEN INITIALIZE Deimplemented	PAGE	203
DMS II -	BDMS/PL/I	PAGE	204
P2842	- PL/I Compiler Looping	PAGE	204
P2843	- Transaction Items	PAGE	204
P2844	- Moving Data Base Items	PAGE	204
P2845	- XREF Option with BDMS	PAGE	204
P2846	- Incorrect BDMS OPEN Statement	PAGE	204
P2847	- Multidimensional DMS Arrays	PAGE	204
P2848	- Builtin Functions and BDMS	PAGE	204
P3209	- CREATE Statement	PAGE	204
P3210	- DATADICTINFO	PAGE	204
P3594	- PUT EDIT of Data Base Items	PAGE	204
P3595	- Data Base BINDINFO	PAGE	205
DMS II -	BDMSCOBOL74	PAGE	206
D3545	- BDMS Option Implemented	PAGE	206
D3555	- OPEN INITIALIZE Deimplemented	PAGE	206
DMS II -	BDMSCOBOL74	PAGE	207
P3588	- Invalid Header	PAGE	207
P3589	- INVALID INDEX	PAGE	207
P3590	- INVALID INDEX	PAGE	207
P3591	- "DB-INVOKE" Hardly Readable	PAGE	207
P3635	- INVALID OP Accessing Global Data Items	PAGE	207
P3636	- Linear Search Selection Expression	PAGE	207
DMS II -	BUILDINQUIRY	PAGE	208
P3094	- Renamed RESTART Data Set	PAGE	208
P3698	- Multiple Blocks in DMINQDIRECTORY	PAGE	208
P3706	- Global Data in Logical Data Base	PAGE	208
DMS II -	BUILDREORGANIZATION	PAGE	209
D3082	- Implicit GENERATE Statements	PAGE	209
D3083	- New Default for <sort options>	PAGE	209
D3084	- Simplification of REORGANIZATON	PAGE	209
D3088	- REORGANIZATION Limitations	PAGE	212
D3582	- Index Control Option	PAGE	212
DMS II -	COPY AUDIT	PAGE	213
P2956	- Print Tape Labels	PAGE	213
P3384	- Alphanumeric Usercodes	PAGE	213
DMS II -	DASDL	PAGE	214
D3048	- Allow Link to Embedded Ds	PAGE	214
D3086	- Guardfiles Stored Under Usercode Directories	PAGE	214
D3099	- Independence of DASDL UPDATE Compilations	PAGE	214
D3115	- COBOL Reserved Word Table Updated	PAGE	214
D3117	- Allow PROPERTIES Label Equation	PAGE	214
D3244	- AREASIZE for Data Sets	PAGE	214
D3316	- Crunch NEWTAPE File	PAGE	216
D3441	- Better CONTROLPOINT, SYNCPOINT Defaults	PAGE	216

D3453	- 28 to 29 Conversion Options Removed.	PAGE	216
D3458	- Changes to DASDL Reference Manual.	PAGE	216
D3558	- FIND NEXT, FIND PRIOR.	PAGE	216
D3568	- Empty Block List	PAGE	216
D3621	- LOCKTOMODIFYDETAILS.	PAGE	216
DMS II	- DASDL.	PAGE	217
P2951	- Allow Modulus Specification for Access	PAGE	217
P2952	- Sequence Number not Updated.	PAGE	217
P2953	- SERIALBUFFERS Attribute.	PAGE	217
P2967	- AREASZ Greater Than 65536 Truncated	PAGE	217
P3072	- Keychanged Text for Field Item	PAGE	217
P3102	- Set Up FILEKINDF, Packname Correctly	PAGE	217
P3124	- Superfluous Too Many Areas Message	PAGE	217
P3181	- Bad Expand Text for Stored Items	PAGE	217
P3231	- Calculate Reasonable Default REBLOCKFACTOR	PAGE	217
P3241	- Loop After Misspelled Update Card.	PAGE	217
P3248	- Correct Handling of B7700 Dollar Option.	PAGE	217
P3315	- Possible Buffer Overlay.	PAGE	218
P3342	- BAD SELECT/VERIFY TEXT FOR FIELD BOOLEANS.	PAGE	218
P3379	- Disallow Restarts DS Named RECOVERY.	PAGE	218
P3385	- Bad Expandtext	PAGE	218
P3386	- MOVES LIST EXCEEDED Error.	PAGE	218
P3442	- Erroneous Initial Values for Remap	PAGE	218
P3487	- EOF NO LABEL Error	PAGE	218
P3699	- Disallow Reorganization Across Releases.	PAGE	218
P3700	- Identifiers Ending with a Hyphen	PAGE	218
P3756	- BLOCKSIZE TOO SMALL Fault.	PAGE	218
P3758	- Creation of Data Base under "*" Directory.	PAGE	218
P3790	- Limit Error on Restart Data Set.	PAGE	218
P3793	- Average Record Size during Update.	PAGE	218
DMS II	- DBANALYZER	PAGE	219
D3369	- DBANALYZER Implementation.	PAGE	219
DMS II	- DMALGOL	PAGE	228
D3049	- DMALGOL Extensions for DMINQ	PAGE	228
D3634	- "NODE <identifier> *" Deimplemented.	PAGE	228
DMS II	- DMALGOL.	PAGE	229
P3009	- Correct "ERROR Construct	PAGE	229
P3323	- Eliminate Extraneous "?"s for "PRINT"	PAGE	229
DMS II	- DMCONTROL.	PAGE	230
D3100	- Update Level Check for RECOVER UPDATE.	PAGE	230
D3101	- Avoidance of RECOVER INITIALIZE.	PAGE	230
D3119	- 28 to 29 Conversion Options Removed.	PAGE	230
DMS II	- DMCONTROL.	PAGE	231
P2758	- System Identification.	PAGE	231
P3108	- DMCONTROL Resequenced.	PAGE	231
P3175	- Control File I/O Lock.	PAGE	231
P3182	- CFDELETEPART Corrupting Control File	PAGE	231
P3185	- Initial Value of Designated Serial Numbers	PAGE	231
P3249	- Mark 32 DMS on Mark 31 MCP	PAGE	231
P3269	- INVALID INDEX on "OVERRIDE HL"	PAGE	231
P3285	- CF Title for OVERRIDE HL	PAGE	231
P3316	- Reduce Use of REORGINFONODE.	PAGE	231
P3317	- Set Up Prefix Arrays	PAGE	231
P3343	- Structure Details.	PAGE	231
DMS II	- MONITOR.	PAGE	232
D3378	- Data Base Monitoring Facility.	PAGE	232
DMS II	- DUMPDIR.	PAGE	236
D3353	- DUMPDIR Enhancements	PAGE	236
DMS II	- DUMPDIR.	PAGE	238
P3372	- Description File Title	PAGE	238
DMS II	- DUMPDIR/LIBRARY.	PAGE	239
P3488	- RETAIN Corrected	PAGE	239
P3489	- Error Using WRITE=/LIST=	PAGE	239
DMS II	- INQUIRY.	PAGE	240
D3171	- INQ CREATE/DELETE.	PAGE	240
D3245	- Accessing Related Records.	PAGE	243
D3264	- Implicit Qualification Improvement	PAGE	245
D3273	- Setting Items to NULL, Testing for NULL.	PAGE	245
D3450	- SAVE Command	PAGE	245
D3489	- Group Keys in Logical Data Bases	PAGE	246
D3615	- Segmented Value Arrays	PAGE	246
D3621	- LOCKTOMODIFYDETAILS.	PAGE	246
DMS II	- INQUIRY.	PAGE	247
P2759	- Optimize FIND VIA <subset> AT <condition>.	PAGE	247
P3000	- System ID and Patch in Heading	PAGE	247
P3001	- Invalid Page Break on Control Breaks	PAGE	247
P3032	- Change to Maximum Display.	PAGE	247
P3161	- Truncation of Position 132	PAGE	247
P3186	- Reporting Long Subscripted Alpha Items	PAGE	247
P3188	- Use of "0" in Unquoted String.	PAGE	247
P3334	- SET DISPLAY Results in Improper Limit.	PAGE	247
P3387	- Functions Performed Via Embedded Manual Subset	PAGE	247
P3443	- Entering Input Before Previous Output Finishes	PAGE	247
P3444	- "?AX" Not Recognized on Linear Search of Set	PAGE	247
P3445	- Display Item Name.	PAGE	248
P3446	- "#NONE", "#NO MORE" Termination	PAGE	248
P3447	- Rounding Virtual Items	PAGE	248

P3490	- Recall of UPDATE Command	PAGE	248
P3555	- Increase Number of Data Set Functions Allowed	PAGE	248
P3655	- Sorting on Subscripted Items	PAGE	248
P3701	- Search on Index Random Set	PAGE	248
P3745	- Virtual Items Evaluated	PAGE	248
P3747	- Improved Searching Capabilities	PAGE	249
P3791	- DISPLAY ALL Correction	PAGE	249
P3792	- Do Not Close Data Base if TASKVALUE=1	PAGE	249
P3797	- Report Control Items	PAGE	249
P3823	- DISPLAY ALL of Global Data	PAGE	249
DMS II	- INTERFACE	PAGE	250
D3229	- Data Bases at Different Release Levels	PAGE	250
D3305	- Anomalies of Logical Invocations	PAGE	250
DMS II	- INTERFACE	PAGE	251
P3103	- Sets not Invoked in Logical DB Properly	PAGE	251
DMS II	- LOADDUMP	PAGE	252
D3614	- Eliminate OPEN INITIALIZE	PAGE	252
DMS II	- LOADDUMP	PAGE	253
P3712	- TITLE Not Parsed Correctly	PAGE	253
P3713	- Prevent SEG ARRAY Error	PAGE	253
DMS II	- PRINTAUDIT	PAGE	254
P2760	- Use Audit Record Information Table	PAGE	254
DMS II	- PROPERTIES	PAGE	255
D3113	- Delete READAHEADB	PAGE	255
D3116	- Put Subsystem ID in Text	PAGE	255
D3118	- Remove Properties for 27 Links	PAGE	255
D3162	- DASDL Defaults	PAGE	255
D3272	- Restructure Description File Properties	PAGE	255
D3274	- Structure Calculations	PAGE	255
D3317	- Crunch PROPERTIES Symbolic	PAGE	255
DMS II	- PROPERTIES	PAGE	256
P3556	- Initialization of Disjoint Unordered Data Set	PAGE	256
P3814	- Incorrect Output Report	PAGE	256
DMS II	- PARTITION CONTROL	PAGE	257
D3466	- 28 to 29 Conversion Options Removed	PAGE	257
DMS II	- RECOVERY	PAGE	258
D3050	- Halt/Load Recovery Sequencing	PAGE	258
D3051	- Switch Back to Primary Audit	PAGE	258
D3289	- Allow Normal REBUILD/ROLLBACK	PAGE	258
D3464	- Quickfix Fails on Inconsistent Partitions	PAGE	259
D3465	- Audit Discontinuity	PAGE	259
DMS II	- RECOVERY	PAGE	260
P2789	- Ensure REBUILD Restart	PAGE	260
P2879	- RLA for Abort, Halt/Load Only	PAGE	260
P2887	- Do Not Print Audit Block if EOF Encountered	PAGE	260
P3036	- Test Generates Too Much Code	PAGE	260
P3109	- Update Disk Header	PAGE	260
P3144	- Read Wrong Audit Block	PAGE	260
P3491	- Row of Ordered Data Set Locked Out	PAGE	260
P3567	- Rollback Fails on FILEDC/STRDC	PAGE	260
P3667	- Corrupt Audit Stopper	PAGE	260
P3668	- Invalid Unit Number	PAGE	260
P3669	- Option USE DUP	PAGE	260
P3757	- SEG ARRAY Error	PAGE	260
P3820	- Split Index Random Tables	PAGE	260
DMS II	- REORGANIZATION	PAGE	261
D3073	- REORGANIZATION Acceleration	PAGE	261
D3120	- DASDL/REORGANIZATION Enhancements	PAGE	261
DMS II	- REORGANIZATION	PAGE	265
P2880	- Eliminate Attribute Error FAMILYNAME	PAGE	265
P2888	- Invalid Block on Standard Data Set	PAGE	265
P2954	- Bad BCW for Ordered Data Sets with Subblocks	PAGE	265
P3093	- Reorganization of Embedded Ordered Data Set	PAGE	265
P3145	- Corruption of Compact Data Set	PAGE	265
P3190	- Specifying COPY	PAGE	265
P3191	- Reorganization of Standard Data Set	PAGE	265
P3318	- INVALID INDEX by Zero Length Read	PAGE	265
P3397	- IXSEQ with Multi-Coarse Table Levels	PAGE	265
P3401	- CFUPDATEVERSION on Pass 1 of FIXUP	PAGE	265
P3484	- CFAUDINZ Only Valid for Audited Data Bases	PAGE	265
P3486	- Invalid Direct Data Set	PAGE	266
P3492	- Adding Checksum to Compact Data Set	PAGE	266
P3670	- Records with Undefined Record Type	PAGE	266
P3671	- Improper Bit Vector Generation	PAGE	266
P3672	- Disk Resident Structure	PAGE	266
DMS II	- UTILITY	PAGE	267
D3052	- Clear TPS Information	PAGE	267
D3431	- Explanation of <recover source>	PAGE	267
D3451	- Label Equation of Data Base Description File	PAGE	267
D3488	- Row Selection Criteria	PAGE	267
D3571	- Tape SERIALNO Specification	PAGE	268
DMS II	- UTILITY	PAGE	269
P2955	- Version Timestamp Mismatch	PAGE	269
P2956	- Print Tape Labels	PAGE	269
P2999	- Failure to Reload Beyond Two Dumps	PAGE	269
P3146	- Output Header	PAGE	269
P3192	- Printing Control Information	PAGE	269

P3193	- Lower Case in Parameter String	PAGE 269
P3243	- Corrupted Dumptime Timestamp	PAGE 269
P3244	- Error Not Given for Invalid Syntax	PAGE 269
P3261	- No Checksum on Block Zero	PAGE 269
P3373	- INVALID INDEX in INITIALIZE	PAGE 269
P3389	- LIST, WRITE Statements	PAGE 269
P3390	- LIST, WRITE of Block Zero	PAGE 269
P3391	- LIST, WRITE Fail to Print Some Blocks	PAGE 270
P3392	- Block Limits for WRITE, LIST	PAGE 270
P3402	- FLUSHDB Default	PAGE 270
P3403	- BUILDDBDIRECTORY Not Accepted	PAGE 270
P3404	- Syntax Errors	PAGE 270
P3405	- DIRECTION Attribute Error	PAGE 270
P3493	- COPY ONTO May Not Update EOF Pointer	PAGE 270
P3495	- Workers Restartable Only Once	PAGE 270
P3586	- Validate Block Range for LIST, WRITE	PAGE 270
P3616	- Handling of Hyphens	PAGE 270
P3673	- Multiple Row control Files	PAGE 271
P3690	- Checksum Error for Block Zero	PAGE 271
P3702	- Display Nonfatal Errors, Warnings	PAGE 271
P3704	- Deadlock	PAGE 271
P3705	- Multiple Dumpworker Error	PAGE 271
P3714	- Recover Family Index	PAGE 271
DMS II -	WFL/COMPILEACR	PAGE 272
D3303	- Changes to WFL/COMPILEACR	PAGE 272
D3478	- Add Compile MONITOR Function	PAGE 273
DMS II -	WFL/COPYAUDIT	PAGE 274
P3619	- Hyphenated Data Base Names	PAGE 274
DMS II -	WFL/COPYAUDIT	PAGE 275
D3270	- Implement COPYAUDIT WFL Deck	PAGE 275
DMS II -	TRANSACTION PROCESSING	PAGE 278
D3334	- TPS Revision	PAGE 278
D3419	- Updating from 31 to 32 TPS	PAGE 278
DMS II -	TFL	PAGE 279
D3319	- Comments in Transaction Bases	PAGE 279
D3320	- Crunch NEWSOURCE File	PAGE 280
D3336	- Parameters Not Specified	PAGE 280
DMS II -	TFL	PAGE 281
P3073	- Expand Text Generation Array	PAGE 281
P3194	- Error Message	PAGE 281
P3195	- Header Displays Correct System Type	PAGE 281
P3557	- Hyphenated RESTARTDATASET Identifiers	PAGE 281
P3674	- Table Size Exceeded	PAGE 281
P3675	- Infinite Loop	PAGE 281
DMS II -	TRINTERFACE	PAGE 282
P3242	- Correct Offset Generation	PAGE 282
DMS II -	TRUTILITY	PAGE 283
P3104	- Search Using Alpha Items as Keys	PAGE 283
P3388	- Output Entire RANGE Specification	PAGE 283
P3715	- Compilation Fails with Syntax Error	PAGE 283
DMS II -	HOSTLIB	PAGE 284
D3085	- Return Address of Last User Transaction	PAGE 284
D3367	- Simultaneous READ/WRITE Access	PAGE 284
D3454	- New Statistics	PAGE 284
DMS II -	HOSTLIB	PAGE 285
P2864	- Check for Unassigned Transaction Formats	PAGE 285
P3749	- I/O Complete	PAGE 285
P3780	- Write Error	PAGE 285
P3794	- Discontinuity of Block Serial Numbers	PAGE 285
DMS II -	REMOTELIB	PAGE 286
D3230	- Elimination of Response From Ports, Signals	PAGE 286
DMS II -	REMOTELIB	PAGE 287
P2881	- Eliminate Unused Requestcase Values	PAGE 287
P3707	- Use Port Files	PAGE 287
DMS II -	HOSTINTERFACE	PAGE 288
P2775	- Send Entire Restart Record	PAGE 288
P2881	- Eliminate Unused Requestcase Values	PAGE 288
P3350	- Eliminate PORT Option	PAGE 288
P3707	- Use Port Files	PAGE 288
DMS II -	TRPROPERTIES	PAGE 289
D3321	- Crunch TRBASE/PROPERTIES Symbolic	PAGE 289
D3322	- TASKVALUE Attribute	PAGE 289
DMS II -	TRPROPERTIES	PAGE 290
P2889	- Increase Number of Files in Journal	PAGE 290
DMS II -	DATA DICTIONARY	PAGE 291
D3335	- Data Dictionary	PAGE 291
DMS II -	DDDASDL	PAGE 292
D3327	- Add Version Level	PAGE 292
DMS II -	DDUPDATE	PAGE 293
D3149	- Reformat Datadictionary Reports	PAGE 293
DUMPALL		PAGE 294
D3011	- Upper Case Input String	PAGE 294
DUMPALL		PAGE 295
P3468	- "PACK=<packname>" Syntax	PAGE 295
P3504	- Bad COPY Syntax Now Flagged	PAGE 295
P3505	- LIST with "<manual input>" "<packname>"	PAGE 295
P3537	- DUMPALL Overrides ON PACKNAME	PAGE 295

DUMP ANALYZER	PAGE	296
D3012 - Analyze IOCB	PAGE	296
D3013 - Analyze FIB at Address	PAGE	296
D3019 - REPEAT Syntax in INTERACTIVE Mode	PAGE	296
D3136 - Print Interactive Input	PAGE	296
D3137 - Dump UNITMAP	PAGE	296
D3138 - Print File Buffers Text	PAGE	296
D3139 - Analyze UNITCONTROL	PAGE	296
D3292 - OLAYINFO Analysis	PAGE	296
D3417 - PROC Command Implemented	PAGE	297
D3442 - Analyze Library Template	PAGE	297
D3479 - Module Alternative Selection	PAGE	297
D3481 - New Port Analysis	PAGE	297
D3588 - New MODE PIB Option	PAGE	297
D3589 - SAVE Command	PAGE	297
D3612 - Analysis of MLIP I/O Data Structures	PAGE	297
D3643 - Analyzing Area Descriptor	PAGE	297
DUMP ANALYZER	PAGE	298
P2779 - Bad Printer Skip	PAGE	298
P2780 - Run DUMPANALYZER in EBCDIC	PAGE	298
P2797 - Correct ID Initialization	PAGE	298
P3061 - "MD RV <addr> FOR ALL" Correction	PAGE	298
P3353 - Remove Port, SIGNAL Code	PAGE	298
P3361 - SAVE Large Dump	PAGE	298
P3396 - Heading Date for Disk Input	PAGE	298
P3429 - Processor Loop after "?END"	PAGE	298
P3430 - HOSTINFO Deimplemented	PAGE	298
ESPOL INTRINSICS	PAGE	299
D3601 - Delete Old Intrinsics	PAGE	299
ESPOL INTRINSICS	PAGE	300
P2933 - Correctly Handle BASIC Stringpool End	PAGE	300
P3042 - Freefield Input with Complex ALGOL	PAGE	300
P3299 - ALGOL Pointer I/O	PAGE	300
P3300 - Update B7000 Define	PAGE	300
P3309 - Array Row Free Format Read	PAGE	300
FILECOPY	PAGE	301
D3020 - TASKFAULT, CLASS, OLDWFL Features	PAGE	301
FILECOPY	PAGE	302
P3201 - "*" File Requests	PAGE	302
P3220 - EXCLUDE USERCODE/=	PAGE	302
P3431 - WFL Deck Sequence Number Limit	PAGE	302
P3432 - NULLFILE Valid Filekind	PAGE	302
P3433 - INCLUDE Does Not EXCLUDE Automatically	PAGE	302
P3434 - Loop on Invalid Syntax	PAGE	302
FILEDATA	PAGE	303
P3158 - Hang on NO FILE	PAGE	303
P3159 - Valid Requests Rejected After First Error	PAGE	303
P3160 - Incorrect Indication of IAD Disk	PAGE	303
P3506 - Report on 5N Disk	PAGE	303
P3507 - "4-digit" Serial Numbers	PAGE	303
P3508 - Show Last File in CHECKERBOARD	PAGE	303
P3509 - FILEORGANIZATION Attribute	PAGE	303
FORTRAN	PAGE	304
D3004 - NEWTAPE with UNITS=CHARACTERS	PAGE	304
D3005 - MOD Operator When Second Argument <1	PAGE	304
D3006 - ERRORFILE Contains Garbage	PAGE	304
D3021 - Intrinsic as Actual Argument Not Allowed	PAGE	304
D3022 - DEBUG DUMP with OWN, SEPARATE	PAGE	304
D3023 - DEBUG TRACE with No File Specified	PAGE	304
D3093 - Warning Message for \$LEVEL	PAGE	304
D3174 - Deimplement SIGNAL, RESPONSE Clauses	PAGE	305
D3217 - BCL Warnings	PAGE	305
D3231 - Deimplement VECTORMODE Option	PAGE	305
D3362 - Modifications to Support Port Files	PAGE	305
D3474 - Formal Parameter in DEBUG DUMP List	PAGE	306
D3542 - REAL, ALMAG Intrinsics Failed	PAGE	306
D3578 - Legal Input to Format	PAGE	306
D3635 - INVALID OP With "> Parameters"	PAGE	306
FORTRAN	PAGE	307
P2709 - Compiler Loop on DATA Statement	PAGE	307
P2710 - Core to Core I/O in WRITE Statement	PAGE	307
P2711 - Allow Double Precision Expression as Subscript	PAGE	307
P2712 - Detect GOTO <non existing statement number>	PAGE	307
P2734 - Strange Action When CHECK, SEQERR Set	PAGE	307
P2735 - Format Error on Read from Double Array	PAGE	307
P2736 - Invalid File Attribute in FILE Statement	PAGE	307
P2836 - INVALID INDEX With SYLPT Table Overflow	PAGE	308
P2837 - AUTOBIND, SEPARATE Set in Main Program	PAGE	308
P2838 - AUTOBIND and GO Despite Syntax Error	PAGE	308
P2867 - \$ Cards Not in NEWTAPE with INCLNEW Set	PAGE	308
P2868 - Filesize Estimate Incorrect	PAGE	308
P2869 - Parameters are Call By Name	PAGE	308
P2904 - Blank Card at End of Subroutine	PAGE	308
P2905 - Spurious Errors with \$VOIDT, \$MERGE Set	PAGE	308
P2943 - \$SEPARATE Without Other Statements	PAGE	308
P2944 - Parameter Mismatch	PAGE	308
P2945 - Arrays Segmented	PAGE	308

P2993 - INV PCW.	PAGE 309
P3007 - Too Much Storage Allocated	PAGE 309
P3013 - INVALID INDEX Due to Conflicting COMMON.	PAGE 309
P3014 - "W2 COMPILER ERROR" When Using IOLIST.	PAGE 309
P3156 - Allow Family Name in \$INCLUDE.	PAGE 309
P3435 - "\$INCLUDE <intname>" Corrected	PAGE 309
P3469 - Compiler Loop.	PAGE 309
P3540 - Extra Comma in Parameter List.	PAGE 309
P3541 - Blank Caused Syntax in FILE Statement.	PAGE 310
P3542 - Use Core to Core I/O	PAGE 310
P3543 - Invalid Common Allocation.	PAGE 310
P3544 - Exported Untyped Function.	PAGE 310
P3545 - Free Format READ with "REAL *8" Variable	PAGE 310
P3546 - Integer Overflow	PAGE 310
P3573 - Double Array as Parameter.	PAGE 310
P3592 - INVALID INDEX.	PAGE 310
P3615 - Erroneous Program After Warning.	PAGE 311
P3656 - Unordered Parameters in Library.	PAGE 311
P3663 - Using Variable IF in COMMON Statement.	PAGE 311
P3776 - DEBUG MONITOR Statement.	PAGE 311
GENERALSUPPORT.	PAGE 312
D3376 - SYSTEM/GENERALSUPPORT.	PAGE 312
GENERALSUPPORT.	PAGE 313
P3042 - Freefield Input with Complex ALGOL	PAGE 313
P3298 - CTOD Terminates Abnormally	PAGE 313
P3299 - ALGOL Pointer I/O.	PAGE 313
P3309 - Array Row Free Format Read	PAGE 313
P3325 - Correct DSQRT Errors	PAGE 313
P3326 - Correct GAMMA, DGAMMA.	PAGE 313
P3534 - Exponent Underflow in RTOR	PAGE 313
P3657 - Update B7000 Define.	PAGE 313
GUARDFILE	PAGE 314
D3493 - Data Base Guardfiles	PAGE 314
HOSTSERVICES.	PAGE 315
D3015 - HOSTSERVICES Changes	PAGE 315
INPUT-OUTPUT.	PAGE 316
D3055 - CENSUS Attribute	PAGE 316
D3076 - APL File Attribute	PAGE 316
D3222 - Close WITH LOCK.	PAGE 316
D3254 - Direct Datacom I/O for Swapjobs.	PAGE 316
D3339 - LINENUM, PAGESIZE Attributes	PAGE 316
D3347 - FULLTRANSLATION Option	PAGE 317
D3373 - DISPOSITION File Attribute	PAGE 317
D3407 - AREASIZE Vs. NEWFILE	PAGE 317
D3408 - AREAClass Vs. FAMILYINDEX.	PAGE 317
D3410 - MTBF Eliminated.	PAGE 317
D3482 - New Attributes Implemented	PAGE 317
D3499 - OPEN Input Reverse Tape Files.	PAGE 321
D3585 - CYLINDERMODE Description	PAGE 321
D3645 - Tape Density	PAGE 321
INPUT-OUTPUT.	PAGE 322
P2771 - UNITNO Vs. BACKUPTAPE.	PAGE 322
P2811 - Datacom File, Family Addition, Subtraction	PAGE 322
P3067 - KIND File Attribute.	PAGE 322
P3074 - Correct State Attribute on Remote EOF.	PAGE 322
P3149 - UPDATE, Binary I/O Writing to Wrong Unit	PAGE 322
P3169 - Update, Binary I/O Read Write Transition	PAGE 322
P3170 - Minimize Header Update	PAGE 322
P3171 - BCL Backup Files	PAGE 322
P3172 - Use of TD830 ODT for SPO Files	PAGE 322
P3174 - Break on Output.	PAGE 322
P3226 - Protected Files Closed	PAGE 322
P3633 - Error Messages Contain Line Number	PAGE 323
P3634 - Random Badly Blocked I/O	PAGE 323
P3783 - I/O Result from SEEK Statement	PAGE 323
INTERACTIVEXREF	PAGE 324
P3147 - Add DATABASE as XREF Item.	PAGE 324
P3330 - Remove PORT, SIGNAL Variable Types	PAGE 324
JOB FORMATTER	PAGE 325
D3026 - Bad Record Dump.	PAGE 325
D3140 - New EOT/EOJ Format	PAGE 325
D3575 - PBIT Time Accounting	PAGE 325
JOB FORMATTER	PAGE 326
P2863 - Print Boxes in Ascending Order	PAGE 326
P3436 - Change CONRAC to ODT	PAGE 326
P3608 - UNITMNEMONICS Array.	PAGE 326
P3795 - DP ALL Message	PAGE 326
KEYEDIO	PAGE 327
D3393 - KEYEDIO Implementation	PAGE 327
D3649 - KEYEDIO Privileged Program	PAGE 327
LCOBOL.	PAGE 328
D3235 - Workfile Compiled for CANDE Compilers.	PAGE 328
LCOBOL.	PAGE 329
P2713 - INVALID INDEX for WORKING-STORAGE.	PAGE 329
P2873 - SNTX for Compile with Syntax Errors.	PAGE 329
P2906 - Same Address for Two Level 77 Items.	PAGE 329
P3547 - General LOAD Instructions Again.	PAGE 329

LOADER.	PAGE	330
D3014 - LH Command	PAGE	330
D3157 - LOADER Improvements	PAGE	330
D3462 - HALTLOADEU Messages	PAGE	331
D3567 - OLAYROW Size	PAGE	331
LOADER.	PAGE	332
P2778 - IV	PAGE	332
P2962 - Convert LOADER to NEWP	PAGE	332
P2986 - 206,207 Disk Pack Coldstart	PAGE	332
P3053 - INVALID ADDRESS Interrupt	PAGE	332
P3437 - INVALID INDEX on MOD 63	PAGE	332
P3694 - Sequence	PAGE	332
LOG ANALYZER.	PAGE	333
D3384 - Add Starting, Ending Times to Heading	PAGE	333
D3627 - THAW Command	PAGE	333
LOG ANALYZER.	PAGE	334
P3278 - Maintenance Log Entries	PAGE	334
P3362 - DL Message INVALID INDEX?	PAGE	334
P3582 - Log Not Found	PAGE	334
P3660 - Unrecovered Errors Shown	PAGE	335
LOGGER.	PAGE	335
D3416 - LOGGER vs DL LOG	PAGE	335
D3457 - Line Equated to Backup Tape with Cataloging	PAGE	335
LOGGER.	PAGE	336
P3166 - ORGMCS, DESTMCS Integer Type	PAGE	336
P3357 - Commas Between SORT Items	PAGE	336
P3363 - Year to Date Sort Errors	PAGE	336
P3364 - No File Jobsummary	PAGE	336
MCP-GENERAL	PAGE	337
D3054 - SWAPPER Enhancements	PAGE	337
D3251 - GETSTATUS/SETSTATUS Enhancements	PAGE	340
D3252 - Semidependent Tasks, VISIBILITY Attribute	PAGE	345
D3418 - GETSTATUS Warning Messages	PAGE	348
D3500 - Changes to SYSTEMSTATUS Calls	PAGE	348
D3501 - Usage Information for I/O Devices	PAGE	348
D3641 - Compile Time Options	PAGE	349
MCP	PAGE	350
D3056 - Idle Patterns in Printer Dump	PAGE	350
D3089 - Memory Dump Tape Record Format	PAGE	351
D3094 - Autoprint Can Run in Local Memory	PAGE	352
D3095 - Intrinsic in Local Memory	PAGE	352
D3102 - BNA MCS	PAGE	352
D3122 - Implied Concatenations Made Explicit	PAGE	352
D3141 - Decode Error Sectors	PAGE	352
D3228 - Printer Dump Hardware Interrupts	PAGE	352
D3281 - Attribute Handling	PAGE	352
D3282 - Virtual Memory Size Statistics	PAGE	352
D3309 - ITINERARY Task Attribute	PAGE	352
D3379 - Set Library Function	PAGE	353
D3381 - Improvements to Working Set Sheriff	PAGE	353
D3386 - New Userdata Error Code "(42)"	PAGE	354
D3399 - CM Vs. Duplicated MCPs	PAGE	354
D3400 - Analyze Library Parameter Mismatches	PAGE	354
D3409 - IAD Not Supported on B6900	PAGE	354
D3411 - Priority, DI Information	PAGE	354
D3425 - Delete PORTS, SIGNALS	PAGE	354
D3468 - Log New Open, Close Information	PAGE	355
D3480 - Intrinsic Mapping	PAGE	355
D3483 - DONT CARE Libraries	PAGE	355
D3484 - USECATDEFAULT Vs. DIAGNOSTICS	PAGE	355
D3485 - OPEN Function Values	PAGE	356
D3487 - MCP Code File Row Size = 504	PAGE	356
D3502 - Shrink Frozen Library's Stack	PAGE	356
D3516 - Library Function Names	PAGE	356
D3529 - PARTNER, EXCEPTIONTASK Removed	PAGE	356
D3536 - SYSTEMSTATUS, IOTRACE for MLIP	PAGE	356
D3548 - DBS in Local Memory Vs. Nonexchanged Unit	PAGE	356
D3550 - Eliminate DL Network	PAGE	356
D3551 - Password Manipulation	PAGE	356
D3564 - WFL Task Fault Across Network	PAGE	357
D3573 - PBIT Time Accounting	PAGE	357
D3576 - PBIT Time Accounting	PAGE	357
D3579 - Processkill Event Errors	PAGE	357
D3583 - Dump Across Midnight	PAGE	357
D3594 - Replace Logging of ORGHOST by ITINERARY	PAGE	357
D3595 - Intrinsic to Support Library Map	PAGE	357
D3600 - Delete Old Intrinsic	PAGE	358
D3606 - OFFSET and DELTA	PAGE	358
D3609 - Complex WAIT with Zero Time Parameter	PAGE	359
D3617 - Compile Time Option OVERHEADCHARGED	PAGE	359
D3618 - COBOL74 Vs. WORD Mode Files	PAGE	359
D3619 - SYSTEMSTATUS Type 4 General Unit Request	PAGE	359
D3620 - Hostname in Heading	PAGE	360
D3652 - Libraries Vs. "??DS", CM, RECONFIGURE	PAGE	360
MCP	PAGE	361
P2538 - Spurious PACK IN USE Message	PAGE	361
P2625 - Printer Dump Loop	PAGE	361

P2765	- Seek Lost Message.	PAGE 361
P2766	- Print Entire Buffer in PROGRAMDUMP	PAGE 361
P2770	- Non-MCS DCWRITE.	PAGE 361
P2773	- Hung Library Maintenance after I/O Error	PAGE 361
P2790	- RESTORE Vs. DS	PAGE 361
P2792	- Halting DCP 0 in SECONDARYINITIALIZE	PAGE 361
P2856	- DISCSTATUS Vs. BLASTUNIT	PAGE 361
P2859	- UNITSTATISTIC.	PAGE 361
P2862	- JOBDESC Complement	PAGE 361
P2865	- SYSTEMSTATUS (11): SWAPPER Parameters.	PAGE 361
P2877	- Dump Mechanism	PAGE 361
P2878	- Bad Momaddress	PAGE 361
P2892	- FORGETCHECK.	PAGE 362
P2893	- Single Bit Error Logging	PAGE 362
P2896	- COPY Vs. FAMILYINDEX	PAGE 362
P2914	- Record Sequence.	PAGE 362
P2919	- Send Message to Local DBS.	PAGE 362
P2920	- Memory Management.	PAGE 362
P2921	- Saving Memory Mods	PAGE 362
P2924	- PROCID Vs. "[MSGADDR+3]"	PAGE 362
P2925	- Memory Management.	PAGE 362
P2927	- "7-Track" Library Tapes.	PAGE 362
P2929	- RECONFIGURATION.	PAGE 362
P2958	- RESOURCECHECK.	PAGE 362
P2959	- Overlay File Corruption.	PAGE 362
P2961	- Incorrect Family Substitution.	PAGE 362
P2963	- SUPERPLUCK Hangs/Dumps Vs. SCHEDULED	PAGE 362
P2994	- Parameter Mismatch	PAGE 362
P3002	- Remove B6700 FINDMEMORYPARITY.	PAGE 363
P3019	- NOT READY RSVP.	PAGE 363
P3020	- Attribute Handling for Data Base	PAGE 363
P3024	- MOVE Vs. Disk Pack Type 206.	PAGE 363
P3025	- DUMPANALYZER Recognizes Frozen MCP	PAGE 363
P3027	- STARTSYSTEM.	PAGE 363
P3028	- Check for Library Capable.	PAGE 363
P3029	- Correct FORGET CHECK on Library Templates.	PAGE 363
P3047	- Tightly-Coupled Main Memory DCP Code	PAGE 363
P3052	- PATHRES DS	PAGE 363
P3063	- Data Base Equation Implementation.	PAGE 363
P3064	- SWAPPER Hung Vs. Controlcard	PAGE 363
P3065	- Job File Roll Out.	PAGE 363
P3066	- DMSOPEN Contiguous Save Memory Requirement	PAGE 364
P3075	- Printer Dump to Drum Printer	PAGE 364
P3096	- NO GO PAST Protection.	PAGE 364
P3097	- Separate Halt/Load Packs	PAGE 364
P3098	- DMSUPDATEDISKHEADER NOOP	PAGE 364
P3100	- RESIZE Reorganization.	PAGE 364
P3112	- DMSII Exception Categories	PAGE 364
P3115	- FILECARDS Attribute FORGETCHECK.	PAGE 364
P3116	- FORGETCHECK after Memory Exceeded.	PAGE 364
P3117	- SWAPPER, Stack Stretch	PAGE 364
P3127	- TC Overlay File Corruption	PAGE 364
P3128	- COPY&COMPARE Vs. Reel Switch	PAGE 364
P3129	- STATISTICS	PAGE 364
P3130	- Scheduling on B6800 Multiprocessor Systems	PAGE 364
P3131	- DL SUMLOG.	PAGE 364
P3132	- Separate Halt/Load Families.	PAGE 365
P3133	- Library Maintenance IOCB	PAGE 365
P3134	- INTRINSICINFO.	PAGE 365
P3135	- "CHECKPOINTed" Swaptask.	PAGE 365
P3136	- Forgotten PIB.	PAGE 365
P3138	- Ready Queue Time	PAGE 365
P3148	- Memory Dumps on a Shared Resource System	PAGE 365
P3150	- ACTIVETIME	PAGE 365
P3173	- Printing of Inuse Code Segments.	PAGE 365
P3179	- Graph for Data Base Users.	PAGE 365
P3196	- ZOT Library Template Marker in BLOCKEXIT	PAGE 365
P3197	- IPC Swapjobs Vs. Subspace Growth	PAGE 365
P3203	- External By Calling Library Procedure.	PAGE 366
P3206	- Make EBCDICTOWORD Inline	PAGE 366
P3219	- Working Sets	PAGE 366
P3222	- Correct PBIT of Zero Length Dope Vector.	PAGE 366
P3223	- Defunct in DUMPBOOTSTRAPPER.	PAGE 366
P3224	- Attribute Grabber Fault.	PAGE 366
P3227	- PAST Order	PAGE 366
P3228	- Resumed ST Tasks have Excess Overlay Goal.	PAGE 366
P3233	- UNIT 0	PAGE 366
P3238	- Card Reader Error Recovery	PAGE 366
P3239	- Copying Too Many Files	PAGE 366
P3240	- Usercode on 14-level File Names.	PAGE 366
P3255	- Logging Internal File Name	PAGE 366
P3257	- Increase Maximum BDNUMBER.	PAGE 367
P3264	- Logging Row Index.	PAGE 367
P3277	- Tasks Suspended by WSSHERIFF	PAGE 367
P3279	- BACKUPQUEUER	PAGE 367
P3280	- Avoid Bad Search from STACKSTRETCH	PAGE 367
P3281	- Programmed Operator Interrupt.	PAGE 367

P3282	- Pass Control File Pack Name to ACR	PAGE 367
P3283	- AUTOPRINT with FMed Printer	PAGE 367
P3301	- Excess Working Set Sheriff Overhead	PAGE 367
P3319	- INV OP in Attribute Handler	PAGE 367
P3321	- Bounce Dump	PAGE 367
P3335	- Handle Null Subsystem in Unravel	PAGE 367
P3336	- Fault Because of Missing Intrinsic	PAGE 367
P3338	- Analyze Library Template	PAGE 368
P3339	- TAPESEARCH	PAGE 368
P3355	- Program Marked as Swapjob	PAGE 368
P3356	- QT PB MT	PAGE 368
P3358	- LOADALABEL vs MULTIFILE	PAGE 368
P3375	- GETSTATUS FORGETCHECK	PAGE 368
P3376	- LOADCONTROL to Tape	PAGE 368
P3448	- Checkpoint/Restart for Programs Using Strings	PAGE 368
P3449	- Restart of Serial Disk Files Near EOF	PAGE 368
P3450	- Rerun of COBOL Files with Use Routines	PAGE 368
P3451	- Multiple Wait	PAGE 368
P3452	- STACKLIMIT Task Attribute	PAGE 368
P3453	- Correct Time Slice Calculation	PAGE 368
P3454	- Allow Halt/Load After Power Up	PAGE 368
P3455	- BOUNCE Message	PAGE 368
P3456	- SYSTEMSTATUS Vs. UNITMOVER	PAGE 369
P3457	- Calculation of Code Core Estimate	PAGE 369
P3458	- FA Swapjob	PAGE 369
P3461	- Checkpoint of Large Size Stacks	PAGE 369
P3478	- Improve Run Time Parameter Checking	PAGE 369
P3479	- Multiple Flagreaders	PAGE 369
P3480	- Halt/Load Memory Configuration	PAGE 369
P3482	- Dynamic EBIT	PAGE 369
P3496	- LOCKTRACE Option	PAGE 369
P3519	- Report on Exclusive Files	PAGE 369
P3521	- REELSWITCH Vs. Density	PAGE 369
P3522	- Stackswap Vs. Stackstretcher	PAGE 369
P3523	- Destname on ACR Codefiles	PAGE 369
P3524	- Reservedisk and Userdata Header	PAGE 369
P3527	- Data Base TITLE Attribute Verified	PAGE 369
P3528	- Units Equal Character Vs. Backup	PAGE 370
P3529	- Password Handling	PAGE 370
P3530	- "CONTROLCARD(QUEUE,7)"	PAGE 370
P3531	- Stack Overflow Handling	PAGE 370
P3568	- Programs Using Mark 31 Ports and Signals	PAGE 370
P3569	- Close Port Values	PAGE 370
P3570	- AVAILABLE Type of File Open	PAGE 370
P3575	- Forgetcheck After Programdump	PAGE 370
P3576	- INVALID OP in DMSCAUSE	PAGE 370
P3577	- SWAPPER Vs. SIB	PAGE 370
P3578	- Software Interrupt Handling	PAGE 370
P3579	- JOBDESC Vs. Nonexchanged Units	PAGE 370
P3585	- NOT READY Messages	PAGE 370
P3607	- Tape Verify	PAGE 370
P3609	- Checkpoint Restart with Array Parameter	PAGE 371
P3610	- Volume Library	PAGE 371
P3620	- Library Maintenance Tape Errors	PAGE 371
P3621	- Paths, FREE, DISKSTATUS Problems	PAGE 371
P3631	- New Fine Priority Algorithm	PAGE 371
P3632	- Messages	PAGE 371
P3647	- Resource Wait	PAGE 371
P3648	- DESTNAME Attribute	PAGE 371
P3649	- ODT Queue	PAGE 371
P3650	- DS Permanent Library	PAGE 372
P3651	- Expand Max Task Parameters	PAGE 372
P3652	- WFL Subroutines	PAGE 372
P3653	- FS and DS ODT Inputs	PAGE 372
P3654	- Read Header Reorganization	PAGE 372
P3685	- "UR"	PAGE 372
P3686	- INVALID OP in Presence Bit	PAGE 372
P3687	- BDNAME SEG ARRAY Fault	PAGE 372
P3688	- Autoprint AX Command	PAGE 372
P3710	- DMSCLOSE Vs. CONTROLLER	PAGE 372
P3746	- Scratch Tape Without Write Ring	PAGE 372
P3748	- DUP FILE Message	PAGE 373
P3781	- TAPEDUMP, Report Block, Rewrite Count	PAGE 373
P3782	- Resize Overlayable Arrays in Global	PAGE 373
P3784	- Avoid Hung Printers	PAGE 373
P3787	- GUARDFILE Vs. CANDE	PAGE 373
P3801	- Corruption of LIBUSEMAP	PAGE 373
P3802	- FIB Creation Locking	PAGE 373
P3817	- Erroneous DS when DBS Initiation Fails	PAGE 373
P3822	- EXCEPTIONTASK Visibility	PAGE 373
P3824	- Memory Organization, Local MCP Code	PAGE 373
P3825	- Parity on Presencebit Stackoverflow	PAGE 374
MESSAGE LEVEL INTERFACE PORT		PAGE 375
D3142	- Initialization Routines for B6900	PAGE 375
D3355	- Log MLIP I/O Errors	PAGE 375
MESSAGE LEVEL INTERFACE PORT		PAGE 378
P2763	- Implement PRINTERDUMP	PAGE 378

P2764	- PRINTIOCB Interface Analyzes IOCB.	PAGE 378
NETWORK DEFINITION LANGUAGE		PAGE 379
D3028	- Message-Oriented Datacom	PAGE 379
D3204	- Changes for DCP Character Oriented Data Comm	PAGE 379
D3385	- Different Terminal Addresses	PAGE 379
D3527	- Transmit/Receive Delays	PAGE 379
D3543	- "<line definition>" Statements	PAGE 379
NETWORK DEFINITION LANGUAGE		PAGE 380
P2714	- Page Between VOIDT and POP VOIDT	PAGE 380
P2804	- ENTER	PAGE 380
P3256	- Inhibit Sync Edit Wrong	PAGE 380
P3599	- Enlarge UNFO Array	PAGE 380
P3604	- "\$NETWORK" Option	PAGE 380
P3605	- Clear Linetable Array	PAGE 380
P3751	- Call BRANCHLINK	PAGE 380
NDLII		PAGE 381
D3397	- NDLII Implementation	PAGE 381
NEWP		PAGE 382
D2973	- "AT <library id>" Allowed	PAGE 382
D2974	- INITIALIZATION is Reserved Word	PAGE 382
D3031	- Conditional Operators	PAGE 382
D3032	- Implement Control State Blocks	PAGE 382
D3033	- Prevent GOTO Into FOR Statement	PAGE 383
D3058	- Inline Procedures	PAGE 383
D3064	- Arrays with Unspecified Bounds	PAGE 384
D3068	- ALTERNATIVES and INITIALIZATION Procedures	PAGE 384
D3103	- Events and Event Arrays as Parameters	PAGE 385
D3106	- Clarification of MAKEPCW Restrictions	PAGE 386
D3110	- "<procedure name>.VALUE"	PAGE 386
D3151	- Increased Host Blocksize	PAGE 386
D3152	- Allow the MCP to Freeze as a Library	PAGE 386
D3158	- Match NEWP Codefile level to BINDER Level	PAGE 386
D3248	- DEFINE Expansion	PAGE 387
D3260	- FIRSTFREEDOCCELL Now Defaults to 10	PAGE 387
D3261	- New Fault Name, LIBLINKFAULT	PAGE 387
D3262	- New \$ Option, STANDALONE	PAGE 387
D3283	- "XREFing" Alternatives	PAGE 387
D3299	- Close LINE and ERRORFILE	PAGE 388
D3300	- Segment Identifiers	PAGE 388
D3401	- PORTS Option Discontinued	PAGE 388
D3420	- PARITYFAIL Fault	PAGE 388
D3421	- HEYOU Disallowed on B7000 Systems	PAGE 388
D3422	- ZAP Intrinsic for B7000	PAGE 388
D3470	- NEWP String, Numeric Constants	PAGE 388
D3486	- Direct I/O	PAGE 394
D3487	- MCP Code File Row Size = 504	PAGE 394
D3530	- FUNCTIONNAME, LIBACCESS Attributes	PAGE 394
D3535	- PACKDECIMAL Intrinsic	PAGE 394
D3549	- "8-Digit" Patch Marks	PAGE 394
D3590	- DESCRIPTOR Procedures to Libraries	PAGE 394
D3591	- Procedure Entry via References	PAGE 394
D3592	- PROTECTED Option in Library Export List	PAGE 394
D3593	- REGISTERS and DLL	PAGE 395
D3598	- SYSTEMLIB Library Attribute	PAGE 395
D3608	- Arrays, Descriptors as By Reference Parameters	PAGE 395
D3626	- Resizing EVENT ARRAYS	PAGE 395
NEWP		PAGE 396
P2733	- Improve FILE/LIBRARY Declaration Handling	PAGE 396
P2743	- Better Listing for Modular SEPCOMP	PAGE 396
P2805	- Error for Missing Procedure	PAGE 396
P2806	- Attribute for Task Array Element	PAGE 396
P2807	- Error for Empty Parenthesis	PAGE 396
P2934	- IXREF Environments for Cheap Blocks	PAGE 396
P2946	- Reduced Time for NEWP XREF	PAGE 396
P2947	- Prevent INVALID INDEX	PAGE 396
P3057	- Address Equation to Undeclared Identifiers	PAGE 396
P3082	- Error For Duplicate Case Elements	PAGE 396
P3087	- SEPCOMP Loses Source Lines	PAGE 396
P3140	- Prevent Compiler SEG ARRAY Fault	PAGE 396
P3297	- Comparing Pointer	PAGE 397
P3344	- Stack Overflow	PAGE 397
P3470	- SEPCOMP Creates Erroneous Stack Items	PAGE 397
P3627	- Null Environments in XREF	PAGE 397
P3628	- Extraneous XREF Environments	PAGE 397
P3629	- "<arithmetic expression> IN <table pointer>"	PAGE 397
NSP DUMP ANALYZER		PAGE 398
D3430	- NSP Dump Analyzer Implementation	PAGE 398
PATCH		PAGE 399
D3007	- Patch Numbers with \$.VERSION/CYCLE	PAGE 399
D3034	- New List and Compare Options	PAGE 399
D3035	- VERSION May Be RESET	PAGE 400
D3446	- Handling of "\$" Cards	PAGE 400
D3447	- "\$ MARK" Option vs \$CARDS	PAGE 400
D3569	- Clarify Additional SYSTEM/PATCH Rules	PAGE 400
D3599	- MARKBLANK and DEL:MOPT Options	PAGE 400
PATCH		PAGE 402
P2715	- Character Mode Files	PAGE 402

P2716	- Out of Sequence Patches.	PAGE	402
P2717	- S.GUARD Option	PAGE	402
P3630	- Line Width	PAGE	402
PATCHCONTROLWARE.		PAGE	403
D3128	- Initial Release of PATCHCONTROLWARE.	PAGE	403
PLI		PAGE	404
D3077	- TRANSLATE BIF.	PAGE	404
D3172	- REWRITE Statement.	PAGE	404
D3218	- CONVERSION Retry	PAGE	404
D3307	- Exponent on E Format Output.	PAGE	404
D3352	- Modifications to Support Port Files.	PAGE	404
D3511	- Included Compile Time Procedure.	PAGE	405
D3519	- Array Elements as Sort Keys.	PAGE	406
D3546	- Binding Programs with DUMP Option.	PAGE	406
D3559	- ONCHAR, ONSOURCE.	PAGE	406
D3560	- ONLOC Builtin Function	PAGE	406
D3561	- UNSPEC Pseudo Variable	PAGE	406
PLI		PAGE	407
P2718	- Exponentiation Mixing Operands	PAGE	407
P2719	- Correct ATAND.	PAGE	407
P2720	- Double Precision PICTURE 'H'	PAGE	407
P2721	- SUBSTR of Binary Data.	PAGE	407
P2722	- Branching from Start of Segment.	PAGE	407
P2767	- System File Attribute Parameters	PAGE	407
P2769	- Statement Numbers in Error Messages.	PAGE	408
P2808	- Precedence of Operators.	PAGE	408
P2809	- Duplicate Label and Entry Name	PAGE	408
P2810	- PIC'(12)HS'	PAGE	408
P2840	- OR Operation on BDMS Field Bits.	PAGE	408
P2841	- ":= " as Assignment Operator.	PAGE	408
P2850	- Undefined Format	PAGE	408
P2870	- Logical Operations on BIT Strings.	PAGE	409
P2907	- Stack Cell for ELSE.	PAGE	409
P2911	- SORT Compares on Pictured Keys	PAGE	409
P2912	- LABELTYPE='OMITTED'	PAGE	409
P2935	- More than 48 "%DO" Statements.	PAGE	409
P2982	- Independent Task Initiation.	PAGE	409
P3006	- Pointer Initialization	PAGE	409
P3016	- Ignored LENGTH or INITIAL Specifications	PAGE	409
P3017	- String Builtin Functions	PAGE	410
P3018	- DIMENSION Not First Attribute.	PAGE	410
P3043	- EXCEPT Builtin Function.	PAGE	410
P3044	- Illegal Primary.	PAGE	410
P3058	- PIC 'X' Array Elements	PAGE	410
P3062	- Compiletime DO and INCLUDE	PAGE	410
P3088	- BIT Compares	PAGE	410
P3143	- PUT EDIT of PIC I Variables.	PAGE	411
P3211	- Parameter Mismatch with NOBINDINFO	PAGE	411
P3212	- Library Capable Bit.	PAGE	411
P3310	- Inappropriate Warning Messages	PAGE	411
P3337	- Bit String Comparisons	PAGE	411
P3367	- PUT EDIT of Bit Variables.	PAGE	411
P3471	- Error TASK IDENTIFIER REQUIRED	PAGE	411
P3472	- Bit String Defined	PAGE	411
P3473	- Bit Overlay Defining	PAGE	411
P3474	- WRITE Without FROM	PAGE	411
P3475	- Compiler Loop Corrected.	PAGE	412
P3476	- Bad Declaration Caused INVALID INDEX	PAGE	412
P3477	- Lost Text from Compile-Time Procedure.	PAGE	412
P3510	- Question Mark.	PAGE	412
P3511	- Bad Lineinfo	PAGE	412
P3512	- External File Variables.	PAGE	412
P3513	- "LENGTH(STRING(<id>))" in SUBSTR Argument.	PAGE	412
P3548	- PICTURE Y.	PAGE	412
P3549	- ROUND Builtin Function	PAGE	412
P3550	- Level 3 Warning Message.	PAGE	412
P3551	- Problem with Pictures.	PAGE	413
P3593	- Large Structures with INITIAL Attribute.	PAGE	413
P3752	- Bit Expressions.	PAGE	413
PLISUPPORT.		PAGE	414
D3374	- SYSTEM/PLISUPPORT.	PAGE	414
D3394	- Implicit Opening of Files.	PAGE	414
PLISUPPORT.		PAGE	415
P2723	- Incorrect Conversion Condition	PAGE	415
P2872	- CHAR to ARITH Conversion	PAGE	415
P2908	- GET DATA Crossing Record Bounaries	PAGE	415
P3213	- ISAM, DELETE	PAGE	415
P3214	- ISAM, Security Error	PAGE	415
P3215	- ISAM, Duplicate Record Keys.	PAGE	415
P3216	- REWRITE, Zero in First Word.	PAGE	415
P3217	- Empty ISAM File.	PAGE	415
P3365	- ISAM Parity Error to COBOL	PAGE	416
P3366	- ISAM Logical Delete.	PAGE	416
P3367	- PUT EDIT of Bit Variables.	PAGE	416
P3514	- PL/I Programdump	PAGE	416
P3515	- PL/I Programdump	PAGE	416
P3552	- GET DATA Loop.	PAGE	416

PLI INTRINSICS	PAGE 417
D3600 - Delete Old Intrinsics	PAGE 417
D3630 - PLINTRN Subsumed by PLISUPPORT	PAGE 417
PRINT BINDER INFO	PAGE 418
P3438 - BINDINFO for Alternatives	PAGE 418
P3596 - Eliminate EOF NO LABEL Abort	PAGE 418
P3597 - Handling of Procedure Parameters	PAGE 418
REMOTE JOB ENTRY	PAGE 419
D2972 - PUT/FETCH Record Compatibility	PAGE 419
D3036 - Print Queue Rebuild at RJE BOJ	PAGE 419
D3037 - *BACKUP Requests Run Asynchronously	PAGE 419
D3038 - MCS Name Display Change	PAGE 419
D3041 - WORDS Vs. CHARACTERS in File Transfers	PAGE 419
D3042 - File Transfer Code Optimization	PAGE 419
D3043 - BCL Constructs Removed	PAGE 419
D3078 - Blank FTS Record at End of FTS Block	PAGE 419
D3079 - CHARACTERSPERFTBLOCK	PAGE 419
D3080 - Crunching Transferred Files	PAGE 419
D3081 - LOCK PROGRAM	PAGE 420
D3091 - Host to Host LOGON Loop	PAGE 420
D3092 - PUT/FETCH String Field Termination Character	PAGE 420
D3098 - Missing EOF	PAGE 420
D3159 - File Transfer Unexpected Abort	PAGE 420
D3160 - Formmessage Link in REMLP Files	PAGE 420
D3161 - Codefile Record Translation	PAGE 420
D3166 - Not Sending 09 Control Message	PAGE 420
D3168 - ONLINE, OFFLINE By Stationname	PAGE 420
D3219 - Records Larger than File Transfer Block Size	PAGE 421
D3220 - Invalid Character Record Translation	PAGE 421
D3234 - Terminal Transfer	PAGE 421
D3247 - Phone Numbers	PAGE 427
D3255 - File Transfer	PAGE 427
D3286 - WH Display Enhancement	PAGE 435
D3287 - SM Command LEVELS	PAGE 435
D3346 - RJE Vs. DLBACKUP	PAGE 435
D3392 - File Starting with "?" Lost Block	PAGE 435
D3423 - DEBUG vs RAID	PAGE 435
D3424 - SM Commands RSC, SPO	PAGE 436
D3428 - Bad Device Address	PAGE 436
D3429 - Halt/Load Restore	PAGE 436
D3475 - Runtime Options Save Through Linkfile Purge	PAGE 436
D3476 - Programdump Out of FILEX, FILER	PAGE 436
D3639 - RJE Vs. SYCOM	PAGE 437
D3640 - ODT to ODT Communication	PAGE 437
REMOTE JOB ENTRY	PAGE 438
P2626 - COPY Syntax Not Checking for Blanks	PAGE 438
P2746 - Autobackup Optimization	PAGE 438
P2874 - Bad Queued File Transfer Count	PAGE 438
P2875 - TIME(1) Vs. TIME(14)	PAGE 438
P2968 - B1800 with B9247-13 Train Printer	PAGE 438
P2969 - COMMENCEPF Set on Wrong Print Queue Entry	PAGE 438
P3076 - RJE Protocol Version	PAGE 438
P3077 - RJE PUTREPLY Vs. SYCOM	PAGE 438
P3078 - Buffer Size Control Message 02, 04	PAGE 438
P3079 - ABORT COPY Request Vs. SYCOM	PAGE 438
P3126 - Protocol Version Mismatch	PAGE 438
P3151 - SM Command	PAGE 439
P3202 - File Transfer Security	PAGE 439
P3345 - Extra Linkfile Updates Removal	PAGE 439
P3346 - Linkfile Updates	PAGE 439
P3439 - Lost Available Records	PAGE 439
P3440 - Invalid Printer Characters	PAGE 439
P3441 - Disconnect of Switched Lines at Logoff	PAGE 439
P3678 - PB of Files Without Summary	PAGE 439
P3682 - Remove Summary File	PAGE 439
P3753 - "RSO" in Printer Backup Record	PAGE 439
P3754 - Abort Compatibility	PAGE 439
P3777 - File Transfer Input Block Size	PAGE 439
P3778 - "##RJE" Message Removed	PAGE 440
P3779 - Object System in Symbolic Header	PAGE 440
P3804 - Station Logoff at RJE QUIT	PAGE 440
P3805 - "22 Control Message NO-OP"	PAGE 440
P3806 - Copy Requests Rejected	PAGE 440
P3807 - Parity Error on REMLP Files	PAGE 440
P3808 - AUTOPRINT INVALID INDEX	PAGE 440
P3809 - Send Control Length Update	PAGE 440
P3810 - Length of "RS" Reply	PAGE 440
P3811 - Incorrect Backup Family	PAGE 440
SCRMCP	PAGE 441
D2970 - IVR Facility	PAGE 441
D3072 - BUFFMEM Modifier Added	PAGE 443
D3169 - Local MCP Code	PAGE 443
D3256 - IVR for 215,225 Packs	PAGE 444
D3342 - IVR Type 206 Packs	PAGE 445
D3345 - IVR Type 207 Packs	PAGE 445
SCRMCP	PAGE 446
P2794 - Pack Density Not Established Properly	PAGE 446

P2812	- Correct Fault Message.	PAGE	446
P2853	- INVALID OP for COMPARE BUFFER ON ERROR	PAGE	446
P2858	- 7A Magtape Controls.	PAGE	446
P2891	- Stop Repeating First Line of ODT Input	PAGE	446
P2922	- Memory Mods for Maintenance, GMM Intrinsics.	PAGE	446
P2923	- SCR/MCP Version.	PAGE	446
P2957	- Buffer with Address Specified.	PAGE	446
P3005	- Wrong Density for File I/O	PAGE	446
P3049	- IVR Write Disabled Pack.	PAGE	446
P3069	- VERIFY DISKPACK (SELECT ALL REPEAT).	PAGE	446
P3123	- Warning Message.	PAGE	446
P3459	- AX More Than 60 Characters for IVR	PAGE	446
PERIPHERAL	TEST DRIVER (PTD).	PAGE	447
D3123	- B6900 Peripheral Test Driver	PAGE	447
D3199	- PTDTESTS Tape.	PAGE	458
PERIPHERAL	TEST DRIVER (PTD).	PAGE	460
P3680	- Peripheral Test Driver	PAGE	460
SORTMCP		PAGE	461
D3233	- Tape Work File	PAGE	461
SORTMCP		PAGE	462
P3374	- Protected Disk Files Used as Sort Output Files	PAGE	462
P3624	- Large Memory Size Specified.	PAGE	462
SOURCEIDL		PAGE	463
D3557	- Blank Patch Fields	PAGE	463
D3613	- ACIII/BDLC BTB Request	PAGE	463
SOURCEIDLII		PAGE	464
D3398	- SOURCEIDLII	PAGE	464
D3602	- Implement ASCII-APL for NSP Datcom.	PAGE	464
SYSTEST		PAGE	465
D3197	- SYSTESTS Tape Reorganization	PAGE	465
UDSTRUCTURE	TABLE	PAGE	467
D3221	- SYSTEMUSER Bit Moved	PAGE	467
D3388	- Class Locator.	PAGE	467
D3415	- Non Interactive APL.	PAGE	467
D3477	- APL-Detached Workspace	PAGE	467
USERSTRUCTURE	COMPILER.	PAGE	468
D3445	- Example of Binding USERSTRUCTURE	PAGE	468
D3605	- New UDSTRUCTURETABLE Generation.	PAGE	468
USERSTRUCTURE	COMPILER.	PAGE	469
P3583	- Use LONG Array Masksearch.	PAGE	469
UTILITY LOADER.		PAGE	470
D3332	- UTILOADER on MLIP Systems.	PAGE	470
WORK FLOW LANGUAGE.		PAGE	471
D2429	- "New" WFL Syntax Clarification	PAGE	471
D3008	- INSTRUCTION Statement.	PAGE	471
D3293	- Remote Job Transfer.	PAGE	471
D3295	- Data Base Equation Allowed	PAGE	471
D3296	- BCL Warning.	PAGE	471
D3351	- File Equation.	PAGE	471
D3361	- Segment Code Files	PAGE	472
D3512	- OPTIONS Attribute with "# <string primary>"	PAGE	472
D3513	- INSTRUCTION Statement Syntax	PAGE	472
D3514	- Mnemonic File Attributes	PAGE	472
D3518	- String Returns Absolute Value.	PAGE	472
D3525	- HISTORY Subfields are of Type Mnemonic	PAGE	472
D3528	- Improved Handling of "SINCLUDE" in Headings.	PAGE	473
D3544	- COPY/ADD Statement	PAGE	473
D3547	- LOCKED in Task Attribute Assignment.	PAGE	474
D3580	- Passing Strings Via WFL/CANDE.	PAGE	474
D3581	- COPY/ADD Statement with Tape Volumes	PAGE	474
D3584	- FILEKIND Example	PAGE	474
WORK FLOW LANGUAGE.		PAGE	475
P2747	- Correctly Compare File and Task Attribute.	PAGE	475
P2748	- Job on Disk with NEWSOURCE, SYNTAX	PAGE	475
P2749	- Pass Global Files to Processed Subroutine.	PAGE	475
P3263	- Task Passed as By Reference Parameter.	PAGE	475
P3520	- Missing Comma in ON Statement.	PAGE	475
P3584	- FAMILY Specification, "<name constant>"	PAGE	475
P3623	- Prevent WFL Fault.	PAGE	475
P3695	- Syntax "Old" WFL Data Base, Library.	PAGE	475
XREF ANALYZER		PAGE	476
P3147	- Add DATABASE as XREF Item.	PAGE	476
P3332	- Remove PORT, SIGNAL Variable Types	PAGE	476
P3518	- Correctly Identify CHARACTER Arrays.	PAGE	476
DOCUMENTS		PAGE	477
D3607	- Stack Size for ALGOL Compile Example	PAGE	477
D3625	- FIND at "<group item>"	PAGE	477
SYSTEST	- SCR/7ABMTEST.	PAGE	478
P3377	- Count READ and WRITE Errors.	PAGE	478
SYSTEST	- UTIL/RESHELPER.	PAGE	479
P3394	- Allow Runs for Disk Pack Types 206,207	PAGE	479
APPENDIX A	DMALGOL Implementation	A-	1

DOCUMENT CHANGES NOTES (D NOTES)

GENERAL

D2782 GENERAL - "KIND=DISK" VS. "FAMILYNAME"

With the implementation of named head-per-track disk families (especially the ability to name them something other than "DISK") and the ability to name a disk-pack family "DISK", the last major distinction between the two forms of random access mass storage devices has been eliminated. Ultimately the differences in the behavior of the operating system when assigning files declared as KIND=DISK and KIND=PACK will also be eliminated.

Currently (Mark 31) the selection of random access mass storage devices varies depending upon whether or not the KIND attribute has a value of DISK or PACK and whether or not the FAMILYNAME attribute has been assigned a value. If the KIND value is DISK then the system supplies the FAMILYNAME of "DISK" whether or not the FAMILYNAME attribute has been given another value by declaration, label-equation, or direct assignment. If the KIND value is PACK then the system supplies the FAMILYNAME of "PACK" unless it has been explicitly set to another value. In other words, the FAMILYNAME (even if assigned a value) is ignored if KIND is DISK and has two different default values if one has not been explicitly assigned.

On Mark 33 the values of DISK and PACK for the KIND attribute will become synonymous and the default value for FAMILYNAME will be "DISK". This has two implications:

1. The FAMILYNAME attribute will be used when assigning files whose KIND is DISK, eliminating the phenomenon of label-equated FAMILYNAMEs being ignored.
2. The assignment of files on the family "PACK" will require the explicit setting of the FAMILYNAME attribute (or the intervention of family substitution).

To aid in the transition to a single default value for FAMILYNAME, on Mark 32 a run-time warning will be given whenever a file from the family named "PACK" is assigned to a logical file that is declared with KIND=PACK and the FAMILYNAME has not been assigned. The warning message is of the form:

ATTRIBUTE ERROR: <intname>. FAMILYNAME @ (<line number>)
ON MARK 33, "FAMILYNAME=PACK" MUST BE SPECIFIED

The warning and the noted problem can be eliminated by changing the logical file declaration to include FAMILYNAME="PACK".

D2981 GENERAL - MARK LEVEL DOCUMENTATION

Beginning with the 32 system release, for documentation purposes, mark levels will appear in Arabic rather than Roman numerals; e.g., Mark 32 rather than Mark III.2, or Mark 27 rather than Mark II.7.

D3090 GENERAL - COMPILER INFO WORD IN SEG ZERO

The new form of the "compiler info word" in segment zero [word 8] of code files is now generated by all the compilers. This format is described in the Mark 31 GENERAL note D2953.

D3104 GENERAL - "XALGOL" DEIMPLEMENTED

As has been cautioned for the two preceding releases, XALGOL code files are not executable on Mark 32 and subsequent MCPs. Also, CANDE will not allow a file of type XALGOL to be accessed, although existing XALGOL symbol files may be modified by changing their type. Installations wishing to manipulate XALGOL symbol files with CANDE may change the FILEKIND with the TYPE verb, and then GET the file (e.g., as an ALGOL symbol file). Attempts to execute XALGOL programs will result in the following message:

"XALGOL CODE FILES ARE NOT EXECUTABLE"

and cause a DSed task history of CAUSE=PROGRAMCAUSE and REASON=NOTEXECUTABLE. The XALGOL compiler has been deleted from the Mark 32 SYSTEM tape.

D3175 GENERAL - MARK "31" SYSTEM NOTES CORRECTIONS

The system notes for the Mark 31 release contain errors, corrections for which are described below.

"System File Page" indicates the page number in the file SYSTEMNOTES/REL310 contained on tape SYSTEMNOTES310; "Field Eng Page" indicates the page number in the Field Engineering Notes in the file SYSTEMNOTES/REL310/FIELDENG contained on tape SYSTEMNOTES310; "Page" indicates the page number in B6000 Series System Notes (Form No. 5011257).

Corrections to Data Management system notes are described in DMSII-GENERAL note D3288.

GENERAL	D2535	System File Page	8
		Field Eng Page	7
		Page	5

The Mark 31 syntax for the ACQUIRE message should read as follows (this syntax will change on the Mark 32 release):

```
-- ACQUIRE --- SHAREMODS -----<mod range>-----|
| - PRIVATEMODS - |
| - PROC ---<node id>----- |
| -<unit type>--<unit number>-- |
```

System File Page	24
Field Eng Page	24
Page	14

The syntax for the DS message <option list> should also allow "*", the semantics for which are as follows:

"* The options appearing in the list will be ORed with any compiled-in options and the net result used to control the program dump."

System File Page	27
Field Eng Page	27
Page	15

The syntax for the FREE message should read as follows:

```
-- FREE --- SHAREMODS -----<mod range>-----|
| - PRIVATEMODS - |
| - PROC ---<node id>----- |
| -<unit type>--<unit number>-- |
```

System File Page	63
Field Eng Page	63
Page	36

The syntax for the RECONFIGURE message is misspelled; it should read "RECONFIGURE".

System File Page	136
Field Eng Page	134
Page	80

The syntax for the RECONFIGURE statement should read as follows:

```
-- RECONFIGURE --- INSTALLATION --- AS ---<group list>-----|
| - GROUP ----- | | - DEFAULT ----- |
| - INSTALLATION -- AS ---<installation id>- |
| | - DEFAULT ----- |
```

MCP	D2430	System File Page	500
		Page	315

The sample host compile deck should read as follows:

```

JOB NEWP/MCP;
BEGIN

PT(STATUS=0);
PROCESS SYSTEM/PATCH [PT];
  FILE TAPE = SYMBOL/MCP,
    PATCH = PATCH/NEWPMCP;
DATA
$.COMPARE MARK
$#
$CLEAR MERGE LINEINFO MCP MAKEHOST
<patches>
? % END PATCH INPUT

WAIT(PT(VALUE)=1);
IF PT(VALUE) NEQ 1 THEN ABORT "BAD PATCH";

COMPILE NEWP/HOST WITH NEWP [CT] LIBRARY;
  COMPILER FILE TAPE = SYMBOL/MCP ;
  COMPILER FILE CARD = PATCH/NEWPMCP DISK;

IF CT ISNT COMPILEDOK THEN ABORT "BAD COMPILE";
REMOVE PATCH/NEWPMCP;

BIND NEWP/MCP BINDER[BT] LIBRARY;
  BINDER FILE HOST= NEWP/HOST;
DATA
  BIND JOBFORMATTER      FROM SYSTEM/JOBFORMATTER;
  BIND CONTROLLER        FROM SYSTEM/CONTROLLER ;
  BIND WFL, CCSTRINGCONV, CCVARIABLEPPB, CCSTRINGFUNCTION
    FROM SYSTEM/WFL;
?
IF BT ISNT COMPILEDOK THEN ABORT "BAD BIND";

END JOB.

```

Appendix B	Libraries	System File Page	B-9
		Page	B-5

In Section 5.1, Creating Libraries, the library must be declared within the same block in which it is used. The example showing 'Dynamic' and 'Indirect' library linkage should have the following declaration inserted between the 7th and 8th lines of the example:

```
LIBRARY SAMLIB(TITLE="OBJECT/SAMPLE/LIBRARY.");
```

System File Page	B-16
Page	B-9

In Section 6.5, COBOL Entry Points and their Parameters, the corresponding ALGOL parameter for "Comp, 01" should read as follows: "Real array [0]".

The following paragraph should be added at the end of Section 6.5:

"Note: The ALGOL construct using an "*" in place of "0" in an array parameter can be achieved via the WITH LOWER BOUNDS clause."

System File Page	B-18
Page	B-11

In Section 8, Libraries in FORTRAN, add the following paragraph at the end of Section 8:

"Libraries may not be created with or called from FORTRAN programs in which the dollar option OPT=1 is set or FORTRAN programs which use the batch facility."

System File Page	B-19
Page	B-11

In Section 8.1, BLOCK GLOBALS Subprogram, the syntax for the <file statement> should read as follows:

```
<file statement>
-- FILE --<file designator>-- ( ----->
|<----- , -----|
>----- <file attribute> = ---<expression>----- ) -----|
|               |-----|
|               |<mnemonic>---|
|<boolean file attribute>-----|
```

The semantics for <file statement> should be replaced by the following:

"The <file designator> is an unsigned integer from 1 to 99. The <file attribute> is a valid file attribute identifier as listed in the I/O Subsystem Reference Manual. The <expression> or <mnemonic> must be a valid value for the attribute identifier.

The <boolean file attribute> is a file attribute identifier of type Boolean. This form of the file attribute assignment results in the value of TRUE being assigned to the file attribute identifier indicated."

System File Page B-19
Page B-11

In Section 8.2, Creating Libraries, the syntax for the <export statement> should read as follows:

```
<export statement>
|<----- , -----|
-- EXPORT ---<subprogram>-----|
|               |-----|
|               | = <string> -|
```

System File Page B-20
Page B-11

In Section 8.2, Creating Libraries, the semantics for the <export statement> should be replaced by the following:

"The <subprogram> name indicates a subprogram which is to be provided as an entry point to the library. The <string> indicates the name by which the subprogram is to be known to the user; if the "= <string>" is not present, the <subprogram> name will be used."

System File Page B-20
Page B-12

Add the following paragraph before the example of a FORTRAN program:

"Calls on FREEZE can only occur in the main program. All subprograms which are listed in EXPORT statements must be declared prior to the first call on FREEZE."

System File Page B-21
Page B-12

In Section 8.3, Referencing Libraries, the semantics for the IN LIBRARY statement should be revised by replacing the paragraph preceding the syntax diagram with the following:

"The IN LIBRARY statement is used to indicate that the subprogram in which it occurs is to be found in a library."

Also, replace the two paragraphs following the syntax diagram with the following:

"The <library id> name must be declared in the BLOCK GLOBALS subprogram. The <string> indicates the actual name of the subprogram in the library.

The only other statements which are allowed in a subprogram which contains an IN LIBRARY statement are type statements and DIMENSION statements."

D3193 GENERAL - INSTALLATION OF MARK "32" SOFTWARE

This note describes procedures for implementing the Mark 32 B6000 series software system. It mentions certain operational procedures and references other system notes which describe major Mark 32 features. The system notes also contain appendices which document certain software products for which manuals are not yet available. It is recommended that installations study the complete set of system notes before implementing the Mark 32 release on their systems.

The Mark 32 release is required for use on the B6900 system. This system is characterized by a new architecture for implementing I/O facilities on the B6000 series systems, and integration of this new architecture into the B6000 series software has been accomplished on this release. The Large Systems software is thus capable of supporting B6700, B6800 and B6900-style systems with a single set of software which adapts itself to the system on which it is running at initialization time. The operational characteristics of the B6900 are described in system note "B6900 Overview". Changes to the NEWP language to support the Mark 32 software are described in the NEWP section of the system notes. The SYSTEM/LOADER and UTILoader have been converted to NEWP and are capable of loading either from a multiplexor system or a B6900; refer to the LOADER and UTILoader sections of these notes for operating information.

The Mark 32 software release contains a compiler and support facilities for the ANSI74 COBOL language. This compiler supports the full COBOL74 language plus Burroughs extensions, most notably those for the DMSII interface. The four new modules released on Mark 32 are indexed I/O, debug, inter-process communication and data communications. All modules of the language are supported at their highest level.

The ALGOL, DCALGOL, NEWP and WFL compilers can all be compiled on the Mark 31 level software using Mark 31 level ALGOL and DCALGOL compilers. The MCP and CONTROLLER can be compiled on the Mark 31 level system, except the Mark 32 level NEWP and DCALGOL compilers must be used. This allows these items to be compiled as the first phase of a bootstrapping process. Note that the MCP supplied on the SYSTEM tape can be used with no modification if the installation has no local patches.

Two object MCP files are released on the BSYSTEM tape:

1. SYSTEM/MCPXXVVV/DIAGNOSTICS
2. SYSTEM/MCPXXVVV

where XX is the mark level and VV is the version.

The following options are SET in the DIAGNOSTICS code file: DIAGNOSTICS, READLOCK, LOCKTRACE, EXPERIMENTAL, and are RESET in the other code file.

When the installation is ready to move to Mark 32, the new MCP can be copied to system disk and the CM ODT message used to change to that MCP. It is recommended that this step be taken with the mix empty, as loss of information may occur if the primitive (??CM) form of the message is used with a non-empty mix. Following the CM, the Mark 32 intrinsics and other system software can be copied from the system tape.

The number of SUBSYSTEMS definable on B6800 Multiprocessor systems has been increased on the Mark 32 release (as described in the MS ODT message, GENERAL note D3356). However, an attempt to CM back to Mark 31 if more than 10 SUBSYSTEMS have ever been defined will result in loss of the job description file.

In order to improve operator communication with the MCP, many ODT messages will be revised in future software releases. The Mark 32 software supports both forms of those messages in order to allow the installation to make a smooth transition from the old to the new. Installations are advised to begin using the new forms of the messages as soon as possible to minimize any transition problems later. Detailed information about the changes to the ODT messages for Mark 32 is presented in GENERAL note D3356 in these notes.

To compile Mark 32 software using Mark 31 software, adhere to the following steps:

1. Compile all software except HOSTSERVICES on the Mark 31.280 (or later) MCP.
2. CM to the new MCP.
3. To compile HOSTSERVICES, the following WFL deck may be used:

```

?BEGIN JOB HS/COMPILE;
  COMPILE SYSTEM/HOSTSERVICES WITH DCALGOL LIBRARY;
  COMPILER FILE TAPE(TITLE=SYMBOL/HOSTSERVICES);
  COMPILER FILE NEWTAPE(TITLE=SYMBOL/HOSTSERVICES,
    SECURITYTYPE=PUBLIC);
  COMPILER FILE CODE(*,SECURITYTYPE = PUBLIC);
  COMPILER FILE CARD(TITLE=HSDATA,KIND=READER,FILETYPE=8);
  COMPILER STACKLIMIT=8000;
  STACK=512;
?DATA HSDATA
$ MERGE SEQERR NEWSEQERR
$ SET LINEINFO NEW
$ SET VERSION 32.133          % PLEASE CHANGE VERSION
?END JOB

```

Changes required to compile either modified or new software items will be described in a PCN to the sample WFL job presented in the SOG Reference Manual, Volume 2 (Form No. 5001688).

D3201 GENERAL - "CONTROLWARE" FILES

A procedure has been established for distributing controlware tapes. See the following CONTROLWARE system notes:

D3191 a description of the file-naming conventions for controlware files

D3200 a description of the Mark 32 controlware files

D3203 GENERAL - "B6900" OVERVIEW

The B6900 system is object-code compatible with the B6800 and supports monolithic (uncoupled) configurations. Although the I/O processor and the I/O controls are new, changes to the operator and programmatic interfaces to the I/O subsystem have been held to a minimum. Some changes in system operation have resulted from the replacement of the maintenance display panel by the BDS maintenance processor.

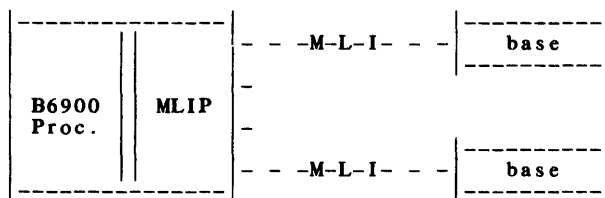
General Configuration Information

The following table lists the B6900 hardware modules and, for comparison, the approximate B6800 counterpart modules:

B6900 Hardware Modules	B6800 Counterparts
Message Level Interface (MLI)	peripheral control interface
MLI Port (MLIP)	multiplexor
Data Link Processor (DLP)	peripheral control
base	peripheral control cabinet
Network Support Processor (NSP)	DCP
Line Support Processor (LSP)	Adapter Cluster

Other hardware components, such as peripherals, disk pack controllers (D-machines), and magnetic tape controllers (MECs), are the same on the B6800 and B6900.

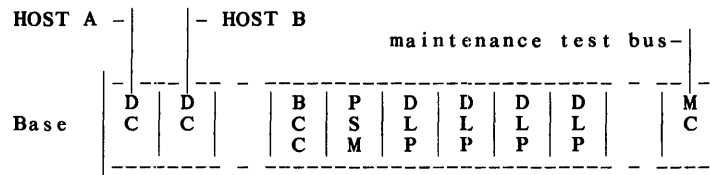
The B6900 processors communicate with the I/O subsystem through the Message Level Interface Port (MLIP). There is one MLIP for each B6900 processor and up to 8 Message Level Interfaces (MLIs) per MLIP. Each MLI connects to a base, as shown in the following diagram:



The MLI is connected to the base through a Distribution Card. There may be up to 6 Distribution Cards in a base, allowing access from up to 6 "hosts". A host is any module that has an MLI Port, such as a B6900 processor, a BDS maintenance processor, or a Network Support Processor.

Each base includes a Base Control Card (BCC), which handles DLP access control between multiple hosts, configuration identity, and maintenance control functions.

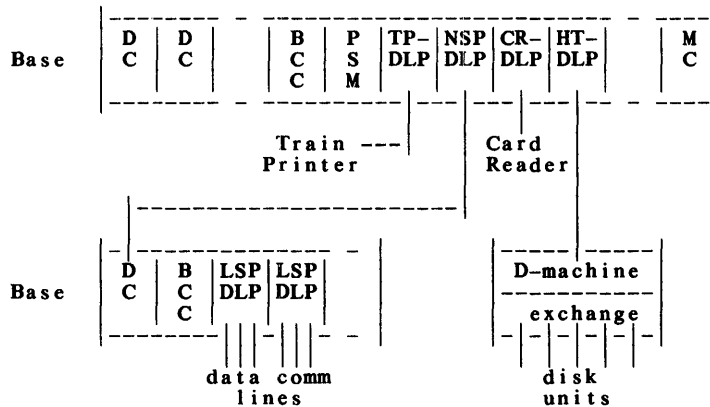
Bases that have more than one Distribution Card must have a Path Selection Module (PSM); the PSM performs priority resolution and return routing from each DLP to the appropriate host. Each base includes a Maintenance Card to connect to the maintenance test bus.



Each base contains up to 8 Data Link Processors (DLPs), which function as peripheral controls. The following DLP types are supported by the Mark 32 software:

DLP	Peripheral
CR-DLP	B9115/6/7 Card Reader
TP-DLP	750/1100/1500 LPM Train Printer
MT-DLP	PE Tape
CP-DLP	Card Punch
HT-DLP	(Host Transfer) 225/235/206/207 Disk Pack
ODT-DLP	Operator Display Terminal
LSP-DLP	Data comm lines
NSP-DLP	Data comm network (LSPs)

A DLP can connect to a peripheral, to a peripheral exchange or controller, or, in the case of an NSP, to another base. An NSP functions as a host to its associated LSPs and connects to the base containing its LSPs through a Distribution Card.



NOTE: The diagrams above are schematic and do not represent actual configuration limits (for example, an LSP can support more than three data comm lines). Additional information about configuration is contained in "D3406 CONFIGURATOR -- Soft Configuration" (of B6900 systems).

Data Communications

The MLIP data communications subsystem consists of Network Support Processors (NSPs) and Line Support Processors (LSPs) in place of the B6800 DCPs and Adapter Clusters. NSPs and LSPs are Data Link Processors and, as such, interface to the main system and main system memory through the MLIP. An NSP acts as a host to any base that contains one or more of its associated LSPs (see diagram above).

Software changes relating to data comm, including the new NDL language and compiler (called NDLII), the DCAUDITOR program, the NSPDUMPANALYZER program, minor DCWRITE restrictions, SOURCENDLII, and the "ID" ODT message, are described or referenced in "D3202 DATACOM -- B6900 Data Communications".

Other Software Changes

The translation between unit numbers and path numbers on the B6900 is different than on the B6800. Thus, configuration information returned when requested by a program or an operator includes path information in the new format. Changes relating to the GETSTATUS intrinsic are described in "D3251 MCP GENERAL -- GETSTATUS/SETSTATUS Enhancements". Changes relating to the "OL" ODT message are described in "D3356 GENERAL -- New and Old ODT Messages".

Maintenance

Each B6900 processor includes a Maintenance Diagnostic Processor (MDP), which consists of a BDS Maintenance Processor (MP) and a display panel. Field support personnel and system operators interface to the MP through an Operator Display Terminal (ODT). The MP requires access to an ODT-DLP and acts as a host to that base (that is, it requires its own Distribution Card); it also has access to the maintenance bus.

In general, the system operator does not need to interface to the MP. However, the MP is involved in system initialization and will request some operator participation during this process (described below).

A new peripheral test package, managed by the Peripheral Test Driver (PTD), has been developed for use with the B6900 and replaces the B6800 SCR package. PTD is described in "D3123 PTDTEST -- B6900 Peripheral Test Driver".

The MCP error logging procedures generate log entries in a new format appropriate for the B6900. LOGANALYZER has been changed to recognize this new format. See "D3355 MLIP -- Log MLIP I/O Errors".

System Operation

The major changes to the operator interface for the B6900 affect ODT commands, configuration file syntax, and system initialization. The changes to the ODT commands are documented in "D3356 GENERAL -- New and Old ODT Messages". The new configuration file syntax is described in "D3406 CONFIGURATOR -- Soft Configuration". System initialization is described in the following paragraphs.

System Initialization

There are three ways to initialize a B6900 MCP, the particular circumstances dictating which form must be used: halt/load (memory load), bootstrap load, and cool/cold start.

Halt/Load

On the B6900, a portion of memory is reserved for the "memory-resident bootstrap", a small program capable of loading the MCP from the Halt/Load unit. If the memory-resident bootstrap is valid and there is a valid MCP on the Halt/Load unit, the system can be initialized by setting the console "MEM LOAD" switch (light ON) and then pressing the console "LOAD" button.

Bootstrap Load

If there is a valid MCP on any disk unit but the memory-resident bootstrap is not valid, the bootstrap should be loaded via UTILoader using the following procedure:

1. Mount the BDSUTILITY tape on a magnetic tape unit selected by switches on the local system control panel.
The flexible disk (Part No. 2379 7830) of the Maintenance Processor must be installed in the MP Flexible Disk Drive.
2. Reset the console "MEM LOAD" switch (light OFF) and press the console "LOAD" button.
3. When the ODT display 'AWAITING "A/T"' appears, enter "T UTILoader".
4. Use the HALTLOAD function of UTILoader, as described in "D3332 UTILoader -- UTILoader on MLIP Systems", to load the memory-resident bootstrap. Specify the Halt/Load unit by entering "PK <nn>".

Cool/Cold Start

If there is no valid MCP on disk, a tape load of the MCP must be performed. UTILoader must be loaded and its TAPELOAD function used to load SYSTEM/LOADER. Then, SYSTEM/LOADER is run to copy the MCP file from tape to disk. (SYSTEM/LOADER is described in the SOG Manual, Volume II, 52 Form No. 5001563, Article 5.) After the MCP is copied, a Bootstrap Load must be performed to initialize the memory-resident bootstrap. The complete sequence is described below:

1. Mount the BDSUTILITY tape on a magnetic tape unit selected by switches on the local system control panel.
The flexible disk (Part No. 2379 7830) of the Maintenance Processor must be installed in the MP Flexible Disk Drive.
2. Reset the console "MEM LOAD" switch (light OFF) and press the console "LOAD" button.
3. When the ODT display 'AWAITING "A/T"' appears, enter "T UTILoader".
4. Use the TAPELOAD function of UTILoader, as described in "D3332 UTILoader -- UTILoader on MLIP Systems", to load SYSTEM/LOADER from tape.
5. Run SYSTEM/LOADER as described in the SOG Manual, Volume II (see also "D3157

LOADER -- LOADER Improvements"). After SYSTEM/LOADER has copied the MCP code file to the appropriate disk unit, the Bootstrap Load process must be performed, as described above.

Changing Halt/Load Units

To change Halt/Load units on a running system, the BOOTUNIT ODT message should be used. BOOTUNIT is described in "D3356 GENERAL -- New and Old ODT Messages". If the system is not running, the Bootstrap Load procedure can be used to establish the Halt/Load unit.

Stand-Alone Programs

In order to perform a Halt/Load, the memory-resident bootstrap code must be present in low-order memory. Loading a stand-alone program into low-order memory (via the TAPELOAD command of UTILoader) will overwrite the bootstrap, so that it is not possible to do a simple Halt/Load to return to the previous MCP. A Bootstrap Load sequence (described above) must be performed.

GLOSSARY OF TERMS

Base	A module that houses multiple DLPs.
BCC	Base Control Card, a base module that identifies a base and controls access to its DLPs and to the base itself.
BDS	The maintenance processor (MP).
DC	Distribution Card, a base module that interfaces a host to a base.
DLP	Data Link Processor, a peripheral control module.
FR-DLP	(obsolete) Frame-Recognition DLP, now called LSP.
HDP	(obsolete) Host Dependent Port, now called MLIP.
Host	An I/O subsystem requestor, such as a processor (MLIP), a BDS, or an NSP.
LCP	(obsolete) Line Control Processor, now called DLP.
LEM	Line Expansion Module, currently not available.
LSP	Line Support Processor DLP, a DLP that directly controls data comm lines.
MDP	Maintenance Diagnostic Processor, the maintenance processor (MP) and display panel associated with each B6900 processor.
MLI	Message Level Interface, the interface between the host (MLIP) and the I/O Subsystem.
MLIP	Message Level Interface Port, a module that controls one or more MLIs on behalf of a processor.
MP	Maintenance Processor.
NDLII	Network Definition Language II, the (new) NDL for the B6900.
NSP	Network Support Processor DLP, a DLP that controls one or more LSPs and provides the logical interface between the host and the data comm network.
PSM	Path Selection Module, a base module that routes messages from a DLP to one of multiple hosts.
PTD	Peripheral Test Driver, the B6900 MCP procedure that runs the peripheral test package.
SC-DLP	(obsolete) Subsystem-Controller DLP, now called NSP.

B6900 Installation

General information about installing the Mark 32 software contained in "D3193 GENERAL -- Installation of Mark 32 Software".

In addition to the system tapes and maintenance tapes, the following tapes are provided for the B6900:

PTDTESTS

The PTDTESTS tapes are described in "D3199 PTDTEST -- PTDTESTS Tape".

CONTROLWARE

The CONTROLWARE tapes are described in "D3200 CONTROLWARE -- Mark 32 Disk Pack

Controlware Files".

D3205 GENERAL - INSTRUCTIONS FOR PRINTING DOCUMENTATION

This release includes documentation files, which may be printed by executing the OBJECT/LISTNOTES program and label-equating the file IN appropriately. Task value varies according to printer capability.

To print a file on an EBCDIC96 printer, use the following commands:

SITE use:
?RUN OBJECT/LISTNOTES; VALUE=0
?FILE IN=<use appropriate file title>

CANDE use:
E LISTNOTES ; FILE IN = <use appropriate file title> ; value 0

To print a file on an EBCDIC72 printer, use the following commands:

SITE use:
?RUN OBJECT/LISTNOTES; VALUE=1
?FILE IN=<use appropriate file title>

CANDE use:
E LISTNOTES ; FILE IN = <use appropriate file title> ; value 0

To create an EBCDIC96 printer backup tape, use the following commands:

SITE use:
?RUN OBJECT/LISTNOTES; VALUE=0
?FILE IN=<use appropriate file title>
?FILE LINE(KIND=PRINTER BACKUP TAPE),
SERIALNO="<use appropriate serialno>")

To create an EBCDIC72 printer backup tape, use the following commands:

SITE use:
?RUN OBJECT/LISTNOTES; VALUE=1
?FILE IN=<use appropriate file title>
?FILE LINE(KIND=PRINTER BACKUP TAPE),
SERIALNO="<use appropriate serialno>")

More complete documentation for LISTNOTES may be found in the MARK 32 GENERAL system note D3396.

D3206 GENERAL - "SYSTEMNOTES" TAPE

Effective with the Mark 32 system software release, the naming convention for tapes containing system software notes included with system software releases is the following:

For major releases: SYSTEMNOTES
For field update releases: NOTESFOR<release>

where <release> is a 5-digit number, MMCCC. MM is the mark level and CCC is the cycle; e.g., 32480.

This note describes the SYSTEMNOTES tape included in the Mark 32 system software release. It contains the following files:

LISTNOTES and OBJECT/LISTNOTES, which is a program which can be used to print the system notes contained on the tape (see GENERAL note D3396 for a description of LISTNOTES).

SYSTEMNOTES, which is the file of D- and P-notes describing the software contained in this release. This file includes the following appendices: DMALGOL Implementation with PCN marks indicating changes since the Mark 31 release.

SYSTEMNOTES/FIELDENG, which is a file of selected P- and D-notes to be used by Field Engineering and system technicians.

NOTE: Some notes cross-referenced in the Field Engineering notes are documented in the full set of system notes only.

DCAUDITOR/EXAMPLE, which is a BD file containing output from a sample run of DCAUDITOR.

NDLII/DOCUMENT, which is a file containing a preliminary user's guide to the NDLII software implementation and its computer controls.

TPS/DOCUMENT, which is a file describing the Transaction Processing System with PCN marks indicating changes since the Mark 31 release.

DATADICT/DOCUMENT, which is a file describing the Data Dictionary system released on Mark 31.

B6000 SERIES MARK 32

MULTIPROCESSOR/DOCUMENT, which is a file describing the B6800 Multiprocessor system released on Mark 31.

SHAREDRESOURCES/DOCUMENT, which is a file describing the Shared Resources system released on Mark 31.

D3207 GENERAL - MARK "32" RELEASE TAPES

The following note describes the contents of the Mark 32 release as shipped with each new system. The actual tape names and configuration of files, as well as product style numbers, are described in the Distribution Availability Letter.

The following "bundled" tapes are included in the Mark 32 system software release and are applicable to the following systems:

Tape	B6700	B6800	B6900
----	----	----	----
BSYSTEM	X	X	X
BSYMBOL1	X	X	X
BSYMBOL2	X	X	X
BSYMBOL3	X	X	X
BSYMBOL4	X	X	X
SYSTESTS *	X	X	X
SYSTESTSMPX *	X	X	
SYSTESTSLANG *	X	X	X
PIDTESTS			X
B9385CW	X	X	X
B9387CW		X	X
B9387CWLIST		X	X
SYSTEMNOTES	X	X	X

* Note: Effective with the Mark 32 release, the SYSTESTS tape has been split into three tapes: SYSTESTS, SYSTESTSMPX, SYSTESTSLANG.

The following "unbundled" DMSIISYSTEM tape is available in the Mark 32 system software release and is applicable to the following systems:

DMSIISYSTEM	X	X	X
-------------	---	---	---

The following files have been added for the Mark 32 system software release:

FIRMWARE/NSP	
SYSTEM/DCAUDITER	SYMBOL/DCAUDITOR
SYSTEM/GENERALSUPPORT	SYMBOL/GENERALSUPPORT
SYSTEM/KEYEDIO	SYMBOL/KEYEDIO
SYSTEM/NDLII	SYMBOL/NDLII
SYSTEM/NSPDUMPANALYZER	SYMBOL/NSPDUMPANALYZER
SYSTEM/PLISUPPORT	SYMBOL/PLISUPPORT
	SYMBOL/SOURCENDLII

The following files have been deleted from the Mark 32 system software release:

SYSTEM/XALGOL	SYMBOL/XALGOL
	SYMBOL/FIRMWARE/215
	SYMBOL/FIRMWARE/225AND235
SYSTEM/DCALGOLINTRINSICS	SYMBOL/DCALGOLINTRINSICS
SYSTEM/PLINTRINSICS	SYMBOL/PLINTRINSICS
SYSTEM/ALGOLPLINTRINSICS	SYMBOL/ALGOLPLINTRINSICS
SYSTEM/BLOCKCHAR	

The following "unbundled" files have been added as separately-priced items to the DMSII system software for the Mark 32 system software release:

SYSTEM/DBANALYZER	SYMBOL/DBANALYZER
	DATABASE/DBMONITOR

For the status of CONTROLWARE/FIRMWARE files, see CONTROLWARE note D3191.

For a description of the FIRMWARE/NSP file, see the syntax and semantics of the ID (Initialize Datacom) ODT message in GENERAL note D3356.

For the status of SYSTESTS files, see SYSTEST note D3197.

D3257 GENERAL - "IVR" FACILITY ON MARK "33" RELEASE

Effective with the Mark 33 release, all packs will be initialized via the IVR maintenance facility.

On the Mark 32 release, when "IV" is entered on the B6700/B6800 systems for Type 215 and 225 packs, the following message is displayed:

"WARNING: IV OPERATIONS MUST BE DONE VIA SCR ON 33."

B6000 SERIES MARK 32

When "IV" is entered on MLIP systems, the following message is displayed:

"ERROR: IV OPERATIONS MUST BE DONE VIA PTD."

For a description of the IVR maintenance syntax for Type 215 and 225 packs, see Mark 32 SCRMCP note D3256.

D3285 GENERAL - DEIMPLEMENTATION OF "ESPOL" COMPILER

The ESPOL language will no longer be supported beginning with the Mark 33 system release. The compiler will not be supplied, CANDE will not MAKE or otherwise handle ESPOL symbolic files and WFL will not recognize the compiler's name.

The Mark 31 PR 1 and Mark 32 ESPOL compiler will unconditionally produce an error message describing its unavailability beginning with the Mark 33 system release.

D3311 GENERAL - "ORGHOSTNAME" ATTRIBUTE DEIMPLEMENTATION

The task attribute ORGHOSTNAME is now contained in the task attribute ITINERARY; consequently, it will be deimplemented on the Mark 33 release. The following warning message is now issued when ORGHOSTNAME is used:

"WARNING - THE ORGHOSTNAME ATTRIBUTE WILL BE DEIMPLEMENTED ON 33. THE INFORMATION FORMERLY CONTAINED IN THE ORGHOSTNAME ATTRIBUTE HAS BEEN SUBSUMED BY THE INFORMATION NOW KEPT IN THE ITINERARY ATTRIBUTE. THE OLD ORGHOSTNAME IS THE SAME AS THE MOST RECENT ENTRY IN THE ITINERARY LIST."

D3329 GENERAL - "MYUSE=IO" VS. "UPDATEFILE" ATTRIBUTES

On the Mark 34 release, the MYUSE file attribute will no longer be used to indicate a request for the Update I/O access method when the logical file is assigned to a disk file. At that time, only the value of the UPDATEFILE attribute will be used to determine the access method. The value of IO for the MYUSE attribute will have the same meaning as for the FILEUSE attribute: the logical file can be both read from and written to.

The UPDATEFILE attribute was implemented on the Mark 25 release to allow the Update I/O access method to be specified in a regular manner. Since Mark 25, the COBOL compiler has set the UPDATEFILE attribute for a file, whenever the file was opened INPUT/OUTPUT, and reset it for all other types of opens; therefore, the only COBOL programs affected by this change will be programs written and compiled before Mark 25.

Beginning with the Mark 32 release, whenever a logical (non-direct I/O) file is assigned to a disk file, MYUSE equals IO, and UPDATEFILE has not been set (currently setting UPDATEFILE to either TRUE or FALSE overrides the setting of MYUSE), a warning message will be displayed along with an UPDATEFILE attribute error message.

A COBOL program will receive the warning

"PROGRAM MUST BE RECOMPILED BEFORE 34 TO CONTINUE
UPDATE I/O ACTION."

Other programs will receive the warning

"ON 34, "UPDATEFILE=TRUE" MUST BE SPECIFIED FOR
UPDATE I/O ACTION."

D3341 GENERAL - SUPPORT OF OLD CODEFILES

On the Mark 32 release, code files more than two releases old (i.e., on Mark 32, the Mark 29 release and older) will be given the following warning:

"WARNING: THIS CODE FILE CANNOT BE RUN ON THE NEXT RELEASE (33)".

As in the past, code files older than Mark 21 will be DSed.

On the Mark 33 release, code files more than three releases old will be DSed.

Examples:

On Mark 32, all code files Mark 29 and older will get the warning
On Mark 33, all code files Mark 30 and older will get the warning
On Mark 33, all code files Mark 29 and older will get DSed
On Mark 34, all code files Mark 30 and older will get DSed

D3348 GENERAL - "FILETYPE=5" FILES

XALGOL linked, variable length files (FILETYPE=5) will not be supported on the Mark 34 release. To help identify programs and files using this record format, a FILETYPE attribute error message will be displayed along with the following warning message:

"ON 34, "FILETYPE=5" FILES WILL NO LONGER BE SUPPORTED"

D3354 GENERAL - INTRINSIC TO LIBRARY CONVERSION

Phasing Plan

Over the next few releases, the system intrinsics will be phased over to the system library scheme. All Burroughs supplied intrinsics will reside in "support" libraries written in NEWP, and user intrinsics must be converted to libraries. The main reason for this conversion is that the handling of intrinsic procedures in libraries is easier and more extensible from both a system and user standpoint than the current mechanism.

On the Mark 32 system software release, the intrinsics and support libraries must be initiated with the following commands:

1. SI+
2. SL GENERALSUPPORT=SYSTEM/GENERALSUPPORT
3. SL PLISUPPORT=SYSTEM/PLISUPPORT

There are more details about operation aspects of intrinsics under Operational Steps.

The intrinsic mechanism will cease to exist on the Mark 34 release, and the following steps will be taken to facilitate the conversion:

1. Mark 32 release

1. MCP support for the library attributes FUNCTIONNAME and LIBACCESS has been added. FUNCTIONNAME allows a library to be referenced by function rather than file title, providing a means for linking to a system library. LIBACCESS determines whether a library is to be accessed BYTITLE (the default) or BYFUNCTION. (See Mark 32 ALGOL note D3530.) This feature is used internally by the MCP to manage support libraries, but is also available in ALGOL and NEWP.
2. The ODT message SL (System Library) has been added to allow definition and management of system libraries, including mapping of function names to file titles. (See Mark 32 GENERAL note D3356.)
3. The ODT message PP (Privileged Program) has been extended to allow the creation of privileged transparent programs. This specification causes procedures which are contained in the privileged transparent programs to inherit the privilege of the calling program. (See Mark 32 GENERAL note D3356.)
4. Many of the procedures which resided in SYSTEM/INTRINSICS on Mark 31 and earlier, now reside in the support library programs. There are two support libraries being released on Mark 32: SYSTEM/GENERALSUPPORT and SYSTEM/PLISUPPORT. The system will automatically link all programs requesting any of these system intrinsic procedures to the proper support library procedure or intrinsic procedure. Therefore, there is no impact on user programs.
5. SYSTEM/INTRINSICS contains all intrinsics which have not been converted. These are the I/O formatting, BASIC, and some miscellaneous intrinsics which will be converted on Mark 32 PR1.
6. SYSTEM/INTRINSICS is now generated by binding only two files, SYSTEM/ESPOLINTRINICS and SYSTEM/ALGOLINTRINSICS. User installation intrinsics may also be bound in until Mark 34. The BINDER issues a warning that user intrinsics are to be deimplemented on Mark 34 in this case.
7. SYSTEM/USERSTRUCTURE no longer generates a code file which is bound into the intrinsics. It now generates a patch which must be compiled into GENERALSUPPORT. (Users who use SYSTEM/USERSTRUCTURE should refer to Mark 32 USERSTRUCTURE note D3605 for details.)
8. The following files have been converted to the support libraries and are no longer released:
 1. SYSTEM/DCALGOLINTRINSICS (and SYMBOL)
 2. SYSTEM/PLINTRINSICS (and SYMBOL)
 3. SYSTEM/ALGOLPLINTRINSICS (and SYMBOL)

The documentation for problem fixes and enhancements (P and D notes) is now under ESPOLINTRINSICS, ALGOLINTRINSICS, GENERALSUPPORT, and PLISUPPORT.

2. Mark 32 PR1 release

1. All intrinsics will be converted to support libraries. An additional support library, SYSTEM/BASICSUPPORT, will be released. This support library will contain the procedures needed to run BASIC.
2. An ESPOL skeletal SYSTEM/INTRINSICS will be released for binding of user intrinsics only.

3. A warning will be issued when a code file links to a user installation intrinsic, that user intrinsics will be deimplemented on Mark 34.
 4. A warning will be issued for the use of array INTRINSICINFO (available in ALGOL), that it will be deimplemented on Mark 34.
 5. A warning will be issued for the use of the \$INTRINSICS option (in ALGOL), that intrinsics will be deimplemented on Mark 34.
 6. A warning will be issued for the use of the \$INSTALLATION option in all compilers, that intrinsics will be deimplemented on Mark 34.
 7. Since all the intrinsics will be converted on this release, the documentation for problem fixes and enhancements (P and D notes) will occur under GENERALSUPPORT, BASICSUPPORT, and PLISUPPORT.
3. Mark 33 release
1. The ESPOL compiler will not be released.
 2. The SYSTEM/INTRINSICS will be released as a skeletal ALGOL host for binding user intrinsics. (Note, this file was ESPOL on Mark 32 PR1.)
 3. The warnings stated under Mark 32 and Mark 32 PR1 will continue to be given.
 4. The compilers will generate code to link directly to the support libraries through the normal library linkage mechanism. Mark 32 and earlier compilers generated calls on the intrinsics which the Mark 32 MCP routed to SYSTEM/INTRINSICS or the proper support library. This routing will continue for Mark 32 and earlier code files, but will not be needed for Mark 33 code files. This change will not impact user programs.
4. Mark 34 release
1. SYSTEM/INTRINSICS will not be released, and the intrinsics mechanism and the ODT messages CI, SI, and WI will be deimplemented.
 2. The array INTRINSICINFO (available in ALGOL) will be deimplemented.
 3. \$INTRINSICS (ALGOL) will no longer be recognized as a system dollar option.
 4. \$INSTALLATION will no longer be recognized as a system dollar option.
 5. User programs compiled with Mark 32 PR1 release and earlier will continue to run because the MCP will continue to link system intrinsic procedures to the correct support library procedures.

Operational Steps

SYSTEM/INTRINSICS have been partially converted to two system support libraries: SYSTEM/GENERALSUPPORT and SYSTEM/PLISUPPORT. Therefore, SYSTEM/INTRINSICS and these two support libraries have been released on 32. When the conversion is complete, there will also be a SYSTEM/BASICSUPPORT (Mark 32 PR1).

User intrinsics may continue to be bound to SYSTEM/INTRINSICS. However, this facility will be deimplemented on Mark 34 and SYSTEM/INTRINSICS will not be released on Mark 34. The user should convert these intrinsics to libraries.

The support libraries have reserved FUNCTIONNAMES. These FUNCTIONNAMES are as follows with a brief description of the library contents:

1. GENERALSUPPORT - This library currently contains all mathematical and various other intrinsic procedures formerly contained in SYSTEM/INTRINSICS. When the conversion is complete on 32 PR1, all the I/O formatting intrinsics will also reside in GENERALSUPPORT.
2. PLISUPPORT - This library contains the support procedures for PL/I.
3. BASICSUPPORT - This library will be released on Mark 32 PR1 and will contain the support procedures for BASIC.

The SL ODT message is used to initiate support libraries, as follows:

SL <functionname> = <library code file title>

For example, GENERALSUPPORT is initiated with the following:

SL GENERALSUPPORT = SYSTEM/GENERALSUPPORT ON SYSPACK

A usercode and family name is permitted in the <library code file title>.

Only installations which use PL/I or the pre-Mark 32 ISAM need to initiate PLISUPPORT.

To initiate SYSTEM/INTRINSICS, the SI (or CI) ODT message is still used.

If the appropriate support library is not running when a program needs to link to it, the program will hang with a "MISSING <functionname>" (e.g. "MISSING GENERALSUPPORT"). When this happens, the library may be Sled, and the program will continue.

NOTE: SLing to a support library is different than SLing to a user library, since SL will initiate a support library but not a user library. In either case, SL links a <functionname> to a <library code file title>.

NOTE: SL differs from SI (CI) in that SL will immediately respond if the support library is not resident. (SI initiates an independent runner which hangs on a "no file" if the intrinsic file is not resident.)

A running support library may be discontinued with SL - <functionname>. The support library will continue to run until all its users terminate. New users which require that <functionname> will hang until another linking SL is performed. Normally, a user should not DS a support library, since this will DS all users connected to it.

NOTE: A support library can only be initiated (or re-initiated) via a linking SL. This may be done at any time. The support libraries may not be explicitly RUN or explicitly linked to (i.e. via LIBRARY declaration).

A support library may be changed at any time with SL. The former support library will continue until all of its current user programs terminate. All new user programs will link to the new support library. If the system halt/loads, all restarting user programs will link to the new support library.

Software Generation

1. The following example WFL deck shows how the support libraries may be recompiled:

```
? BEGIN JOB COMPILESUPLIBS;

SUBROUTINE COMP(STRING LIB);
BEGIN
  TASK T;
  COMPILE SYSTEM/#LIB WITH NEWP[T] LIBRARY;
  COMPILER FILE TAPE(TITLE = SYMBOL/#LIB);
  COMPILER DATA
$ CLEAR MERGE LINEINFO SEQERR NEWSEQERR
?
  IF T IS COMPILEDOK THEN
    WAIT("PP SYSTEM/" & LIB & ":TRANSPARENT",OK)
  ELSE
    ABORT "RECOMPILE " & LIB;
END; % COMP

COMP("GENERALSUPPORT");
COMP("PLISUPPORT");

? END JOB
```

2. Before a system support library is Sled, it must be PPed through the ODT as follows:

```
PP <library title> : TRANSPARENT
```

This is extremely important and establishes the library as privileged transparent. (See Mark 32 GENERAL note D3356.) Essentially, this guarantees that a privileged program which invokes the support library remains privileged during the invocation, and that a non-privileged program which invokes the support library remains non-privileged.

3. The procedure for compiling SYSTEM/ESPOLINTRINSICS and SYSTEM/ALGOLINTRINSICS has not changed.
4. The following example WFL deck shows how the SYSTEM/INTRINSICS are now created:

```

? BEGIN JOB BINDINTRINSICS;
  TASK T;
  BIND SYSTEM/INTRINSICS WITH BINDER[T] LIBRARY;
  COMPILER FILE HOST(TITLE = SYSTEM/INTRINSICS);
  COMPILER DATA
$ INTRINSICS
BIND = FROM SYSTEM/ESPOLINTRINSICS, SYSTEM/ALGOLINTRINSICS;
?

  IF T ISNT COMPILEDOK THEN
    ABORT "RE-BIND INTRINSICS";

? END JOB

```

5. The procedure for binding user intrinsics has not changed.

6. The UDSTRUCTURETABLE is no longer a code file which is bound into SYSTEM/INTRINSICS. SYSTEM/USERSTRUCTURE now generates a patch which must be compiled into the GENERALSUPPORT library. (Users who require this feature should refer to Mark 32 USERSTRUCTURE note D3605 for details of the procedure.)

D3356 GENERAL - NEW AND OLD "ODT" MESSAGES

In order to improve operator communication with the MCP, many ODT messages have been revised or will be revised in future software releases.

Primarily, the changes made allow the mix-related messages to appear only in the following form:

<mix number list> ODT MESSAGE.

Previously, ODT MESSAGE <mix number list> has been also allowed.

The first release of the B6800 Multiprocessor system and the Mark 31 system release allowed either form; the Mark 33 system release will allow only the new syntax forms.

The following list shows the old and new mnemonics released on Mark 31 where applicable, syntax for which is described herein; syntax for all other previous forms is unchanged, except as noted:

Old ---	New ---	
AP	AB	Auto Backup
BRK	CQ	Clear Queued Messages
CI	SI	Display System Intrinsics
DC	ID	Initialize Datacom
DP	DUMP	Dump Memory
EI	HS	Prevent Schedule Jobs Entering Mix
EQ	MQ	Make or Modify Queue
IV	RC	Reconfigure
LR	TL	Transfer Log
M	MX	Display Mix Entries
MIXL	ML	Mixlimit
P	PER	Display Peripheral Status
PC	SC	Display System Configuration
PU	MU	Make User
RESTORE	SUPPRESS-	Restores Display of Active Job
RET	RES	Reserve Specified Area of Disk
RO	OP	Reset Options
RR	SR-	Remove Reader Security Restrictions
SO	OP	Set Options
TD	TDIR	Tape Directory
TO	OP	Options
UA	UR-	Unit Reserved, Available
WD	TD	Display Date
WI	SI	Display System Intrinsics
WT	TD	Display Time
WS	CS	Change Supervisor
XS	FS	Force From Schedule Queue

The following messages were added or modified for the B6800 Multiprocessor system:

A (Active Mix Entries)
 ACQUIRE
 CF (Configuration File)
 CU (Core Usage)
 DL (Disk Location)
 FREE
 GC (Group Configuration)
 HN (Hostname)
 HU (Host Usercode)
 J (Job and Task Structure)
 LP (Lock Program)

ML (Mixlimit)
 MQ (Make or Modify Queue)
 MS (Make Subsystem)
 MX (Mix Entries)
 NET (Network)
 PG (Purge)
 PP (Privileged Program)
 RECONFIGURE
 RY (Ready)
 S (Scheduled Mix Entries)
 SN (Serial Number)
 SV (Save)
 SW (Swapper)
 W (Waiting Mix Entries)
 WM (What MCP)

The following messages have been added or modified on the Mark 32 release:

ACQUIRE
 BOOTUNIT
 CM (Change MCP)
 HN (Hostname)
 HU (Host Usercode)
 ID (Initialize Datacom)
 LH (Load Host)
 MA (May Access)
 NET (Network)
 NW (Network Prefix)
 OL (Label Table)
 OP (Options)
 PP (Privileged Program)
 SP (Show Print Queue)
 SQ (Show Queue)
 SL (System Library)
 SW (Swapper)
 TERM (Terminal)
 THAW
 TI (Times)
 UR (Unit Reserve)
 Y (Status Interrogate)

The Mark 32 system software release supports monolithic and multiprocessor B6700 and B6800 configurations and monolithic B6900 configurations. The examples of some ODT messages may differ slightly depending on the type of system configuration.

The following note describes changes to ODT messages on the Mark 31 and Mark 32 system software releases. PCN marks in the right margin indicate Mark 32 changes.

A (Active Mix Entries) Message

Revised Syntax:

```
-- A ----->
  | -/1\- ALL -|
  |
  | <-----|
  | -/1\- SWAPPER -----|
  |   |
  |   | -/1\- MCSNAME -----<mcsname>-----| |
  |   |   |
  |   |   | - = -|
  |   | -/1\- IN -----<subsystem id>-----|
  |   |   | - ( <processor id list> ) -|
```

<processor id list>

```

|<-----|
|  |<- , -|  |
|-----|
|GLOBAL|
|  |
| - 1 -----|
|  |
| - 2 -----|
|  |
| - 3 -----|
|  |
| - 4 -----|

```

Revised Semantics:

The A (Active Mix Entries) message lists all active jobs and tasks. If ALL is used, any active jobs or tasks which have been suppressed by the SUPPRESS message are displayed in addition to unsuppressed tasks. In order to avoid congestion in ADM, frozen libraries are displayed by the A ALL and the MIX commands; they are no longer displayed in an ACTIVE mix picture.

When SWAPPER is specified, only jobs running in swapspace will be displayed.

When MCSNAME is specified, only jobs that originated from the specified MCS will be displayed.

On a B6800 multiprocessor system, each displayed task is preceded by a subsystem indicator: processor id for a local memory task, "G" for a global memory task, or blank for a task whose subsystem location is currently unassigned.

When IN is specified, only jobs with stacks running in the subsystem identified or the processor identified will be displayed.

A typical response to the A message is as follows:

```

-----7 ACTIVE ENTRIES-----
0232 JOB 80DCP/0
0234 JOB 70SYSTEM/CANDE
0243/0244 55 ESPOL JKL
0254/0254 55#SEPCOMP JHOST2/HOST
*0255x0256 55 ALGOL KAP

```

The lower case "x" between the job and task number in the last entry of the example indicates that the job is under control of an MCS. Swap jobs are flagged with an "#" between the priority number and the file name. The number in the active entry heading (7) is the total number of active entries including suppressed entries.

New Example:

```
A ALL MCSNAME=SYSTEM/CANDE
-----
```

This message will display all active jobs (including suppressed) that originated from SYSTEM/CANDE.

New Example: (B6800 Multi-Processor System):

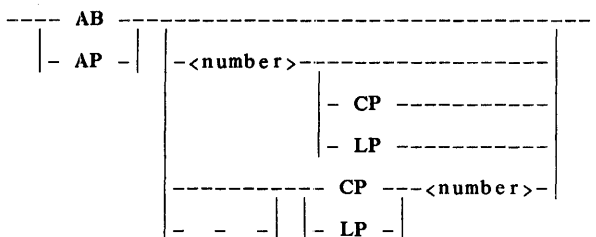
```
A SW IN(3)
-----
```

This message will display all active jobs (excluding suppressed) whose stacks are running in swapspace and are currently swapped into processor 3.

AB (Auto Backup) Message

AB replaces the AP (Auto Backup) Message.

Syntax:



The AB (Auto Backup) message sets the maximum number of ABed line printers and/or card punches. This number is set to zero at Halt/Load time if the MCP option AUTORECOVERY is reset. When AUTOBACKUP is looking for a line printer or card punch to use for automatic output of backup files, it begins looking for on-line ABed units. If AUTOBACKUP cannot find any ABed units, it tries any on-line line printer or card punch.

Example :

AB MAX=2; CP MAX=1; AB-ED LPS=0; AB-ED CPS=0

When a number immediately follows the AB, that number is used as the maximum number of copies of AUTOPRINT allowed.

Example :

AB MAX=3; CP MAX=1; AB-ED LPS=0; AB-ED CPS=0

When a number and device immediately follow the AB, that number is used as the maximum number of copies of AUTOPRINT allowed on that device.

Example:

AB MAX=3; CP MAX=0; AB-ED LPS=1; AB-ED CPS=0

When a device and unit number immediately follow the AB, the indicated output device is ABed.

Example :

The number of devices assigned may exceed the number of allowed copies of AUTOBACKUP. The following sequence of AB messages allows one AUTOBACKUP printing on LP 5.

Example :

AUTOBACKUP may be disabled on the indicated device by preceding the specified device with a hyphen. When such a message references a device upon which output is currently being generated, this activity is allowed to proceed to a normal termination before the device is disabled.

Example :

B6000 SERIES MARK 32 RELEASE 05 NOVEMBER 1980 GENERAL

When use of AB results in all line printers and card punches being freed of the ABed status and AB is set to 0, the queue of disk/diskpack backup yet to be printed is forgotten. This does not affect the actual backup files, except they are not printed automatically by AUTOBACKUP. If subsequently any LP is given AB status or AB is set to a non-zero number, all disk and native mode diskpacks will be searched for backup files which are enqueued for printing. A similar situation holds for card punch files.

Backup files introduced by Library Maintenance (i.e., COPY, COPY&COMPARE and ADD) from tape are picked up by this rebuilding of the queues and will be printed in turn.

Example:

```
COPY BD/0000365/0000366/000TASKFILE
COPY BP/0000367/0000368/000TASKFILE
```

Backup files of this form will be picked up.

```
COPY BD/0000365/0000366/000TASKFILE AS JEV/TASKFILE
```

Backup files of this form will not be picked up.

ACQUIRE Message

Syntax:

```
-- ACQUIRE --- SHAREMODS -----<mod range>-----|
|               |               |               |       |
| - PRIVATEMODS -|               |               |       |
|               |               |               |       |
| - PROC --<node id>-----|       |               |       |
|               |               |               |       |
| -<unit type>--<unit number>-----|       |               |       |
|               |               |               |       |
|               |               | - : SAVE -   |       |
|               |               | - : SV  ---  |       |
|               |               |               |       |
```

Revised Semantics:

Existing groups can be altered by the ACQUIRE and FREE messages. The ACQUIRE message allows an active group to acquire additional resources. When a unit is ACQUIRED, it is automatically RYed unless the SAVE option is specified. Also, if a system has a unit that is not in use and another system wishes to acquire that unit, the desired unit is SAVED (and CLOSED if appropriate) and FREED automatically.

Revised Example:

PER MT

```
----- MT STATUS -----
81*P [HALL ] 1600 #1 1:0 MEMORY/DUMP [1,2]
82  NOT AVAILABLE TO GROUP
83  NOT AVAILABLE TO GROUP
```

ACQUIRE MT 82

MT 82 WILL BE ACQUIRED

PER MT

```
----- MT STATUS -----
81*P [HALL ] 1600 #1 1:0 MEMORY/DUMP [1,2]
82*P [MIKE ] 1600   S C R A T C H [1,2]
83  NOT AVAILABLE TO GROUP
```

ACQUIRE MT 83:SAVE

MT 83 WILL BE ACQUIRED

PER MT

```
----- MT STATUS -----
81*P [HALL ] 1600 #1 1:0 MEMORY/DUMP [1,2]
```

B6000 SERIES MARK 32

82*P [MIKE] 1600 S C R A T C H [1,2]
 83*P [000001] SAVED

AD (Access Duplicate) Message**Revised Syntax:**

```
-- AD -----<familyname>-- ( --<familyindex>-- ) --|
      |-----|
      |-----|
      | - ON - |
```

Semantics:

The AD (Access Duplicate) message causes the access structure (SYSTEM/ACCESS or SYSTEM/CATALOG) to be duplicated. The duplicate structure has the family index number appended; e.g., SYSTEM/ACCESS/002 or SYSTEM/CATALOG/003. The family index represents a member of the Catalog or Access family.

Example:

AD 2

*608 JOB 80 COPYDIR

AD- causes the referenced access structure to no longer be considered a duplicate. The actual file is not removed.

Example:

AD- 2

SYSTEM/ACCESS/002 REMOVED

ADM (Automatic Display Mode) Message

ADM has been revised, as follows:

MIX has been changed to MX.

Revisions made to the following messages have been applied to ADM also:

A (Active Mix Entries)

J (Job and Task Structure Display)

ML (Mixlimit)

MX (Mix Entries)

S (Scheduled Mix Entries)

W (Waiting Mix Entries)

See the description of the individual messages for details.

Syntax:

Unchanged

Semantics:

Unchanged

Example:

Unchanged

AP (Auto Backup) Message

B6000 SERIES MARK 32

AP has been eliminated; see AB (Auto Backup) for new message.

AX (Accept) Message

Revised Syntax:

```
--<mix number list>-- AX --<text>--|
```

Semantics:

Unchanged

Example:

Unchanged

BOOTUNIT Message

The BOOTUNIT message specifies new bootstrap units via software.

This message provides, on MLIP systems, an analogy to the Halt/Load pins present on MPX systems.

The BOOTUNIT message is valid on a B6900 system only.

Syntax:

```
-- BOOTUNIT -----|
      | - PK ---<unit no>-|
      | - DK -|
```

Examples:

- 1) BOOTUNIT PK 50
- 2) BOOTUNIT

Example 1) is the form of the message on monolithic B6900 systems. It specifies PACK 50 to be the next Halt/Load unit.

Example 2) is the interrogation form of the message, which causes the current boot strap unit to be displayed on the ODT.

BRK (Break) Message

BRK has been eliminated; see CQ (Clear Queue) for new message.

CF (Configuration File) Message

Syntax:

```
-- CF -----|
      | - + -----|
      | - - -----|
      | -<file name>-|
```

Semantics:

NOTE: B6900 Multiprocessor systems are not supported on the Mark 32 release.

The CF message is used to designate or display the title of the current configuration file.

CF with no options displays the title of the current configuration file.

CF+ selects SYSTEM/CONFIGURATION, the default, as the configuration file.

CF- removes the designation of the configuration file.

CF <file name> selects the named file as the configuration file.

Example:

CF

CONFIGURATION FILE: (EVANS)SYSTEM/GMMCONFIG/022079

CF +

SYSTEM/CONFIGURATION IS THE CONFIG. FILE

CF XYZ

XYZ IS THE CONFIG. FILE

CI (Change Intrinsic) Message

CI has been eliminated; see SI (System Intrinsic) for new message.

CM (Change MCP) Message

Syntax:

Unchanged

Semantics:

The CM ODT message has been changed to simplify and accelerate the creation of duplicated MCPs. Formerly, when an MCP was to be duplicated (or triplicated), CM always created a brand new file for each family index involved. This is no longer required. Now, CM messages of the following forms:

CM <filename> . . .

or

CM <filename> FMLYINX <nnn> . . .

are handled as follows:

1. CM locates the specified file using the full name given in the command.
2. CM then makes a list of the filenames that are required for each family index. If "<filename> FMLYINX<nnn>" is specified, <nnn> is changed. If "FMLYINX<nnn>" is not specified, CM appends the correct member for each family index.
3. CM then attempts to locate the required file for each member. If a file is found and the code files are "identical" (as determined by the compiler timestamp), that file is used for that member; otherwise, a new file is built for that member.
4. CM then copies or moves rows as required and continues with normal CM.

CQ (Clear Queue) Message

Syntax:

```

----- CQ -----|
| - BRK - |

```

Semantics:

The CQ (Clear Queue) message clears system messages queued for the operator display terminal

A typical response to the CQ message is the following:

MESSAGES FLUSHED

CS (Change Supervisor) Message

CS incorporates the WS (What Supervisor) message.

Revised Syntax:

```
-- CS -----|
      |-----|
      | -<file title>-- ON --<familyname>--|
```

Semantics:

The CS (Change Supervisor) message displays, cancels or designates as a "supervisor" program a code disk file specified by a file title, as follows:

CS	Displays the current supervisor
CS-	Cancels the current supervisor
CS <file title>	Designates the supervisor

The file must be present when CS is executed.

At Halt/Load time, the supervisor program is automatically entered into the mix. If the run-time system option DUPSUPERVISOR is set, the MCP will attempt to execute a code file <file title>/FMLYINX<nnn>, where <nnn> is the family index. If DUPSUPERVISOR is reset, the MCP will attempt to execute the designated supervisor program.

Example:

```
CS
--
```

SUPERVISOR: SYSTEM/SUPERVISOR

```
CS-
---
```

SUPERVISOR: NOT SPECIFIED

```
CS SUPPRESS/MIX/NOS
-----
```

SUPERVISOR: SUPPRESS/MIX/NOS

CU (Core Usage) Message

Revised Syntax:

```
----- CU --|
| -<mix number list> -|
```

Semantics:

Unchanged

New Example:

The resulting display for a B6800 multiprocessor system will depend upon the number of local processors comprising the system. A 2-processor system could produce the following:

CORE USAGE AT 10:21:04

GLOBAL MEMORY

AVAILABLE	255028
NON SAVE	105882
SAVE	32306
TOTAL	393216

LOCAL MEMORY 1

AVAILABLE	111967
NON SAVE	301632
SAVE	98401
TOTAL	512000

LOCAL MEMORY 2

AVAILABLE	31843
NON SAVE	357335
SAVE	122822
TOTAL	512000

SYSTEM TOTAL

AVAILABLE	398838
NON SAVE	764849
SAVE	253529
TOTAL	1417216

As the number of processors increase, the number of local memory statistic blocks will increase.

For a single B6800 with global memory, the display will be of just the local memory with one additional line giving the total of the shared (global) memory available.

DC (Datacom Initiation) Message

DC has been eliminated; see ID (Initialize Datacom) for new messages.

DD (Directory Duplicate) Message

Revised Syntax:

```
-- DD -----<familyname>-- ( --<familyindex>-- ) --|
      | - - - | | - ON - |
```

Semantics:

Unchanged

New Example:

DD TIOMASTER (2)

-----ACTIVE ENTRIES-----

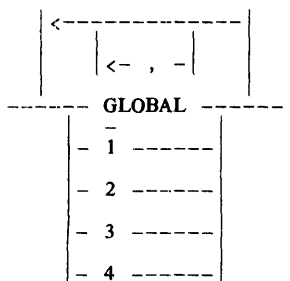
*9303 JOB 80 COPYDIR

DL (Disklocation) Message

Syntax:

```
--- DL ----->
      | - DISKLOCATION - |
>-----|
      | - BACKUP -----|
      | - USERDATA -----| | - ON --<familyname>-|
      | - OVERLAY -- ( <processor id list> ) -|
      | - JOBS -----|
      | - CATALOG -----|
      | - LOG --<familyname>-----|
```

<processor id list>



Semantics:

The DL (Disklocation) message specifies that system files reside on families other than the Halt/Load family.

The simple form of "DL" is used to interrogate the current settings of the various "DL" families.

A typical response to the DL message is as follows:

```

DISK LOCATION:
CATALOG ON DISK
JOBS ON DISK
USERDATA ON DISK
BACKUP ON BKPCK
LOG ON DISK
OVERLAY ON DISK

```

The DL OVERLAY specification allows the user to direct system OVERLAY file allocation (and the consequent overlaying I/O activity) to any desired family. For a B6800 Multiprocessor System, each processor has its own DL OVERLAY specification to ensure direct I/O access to the memory being overlayed. Different local processors may not be able to access directly the same disk-type devices.

The <processor id list> specification for an OVERLAY family is meaningful only for a B6800 Multiprocessor System, and is rejected on ODT inputs to a monolithic system.

The presence of each pertinent DL OVERLAY family is checked at system initialization time, and whenever an ODT input attempts to change a specification. (This "presence" check includes a test for proper I/O visibility on a B6800 Multiprocessor System.) If a family is not present, an RSVP wait allows the user the following options:

1. Mount the family baseunit and "OK" the RSVP wait.
2. "OF" or "DS" the RSVP wait to disassociate DL OVERLAY from the missing family. (At initialization time, the unsatisfied DL OVERLAY specification defaults to the <Halt/Load familyname>; on a B6800 Multiprocessor System, the default for a local processor is the <Halt/Load familyname> for that processor. If the wait arose from an ODT input, the previous family specification for the DL OVERLAY case in question is retained.)

Whenever a DL OVERLAY specification is changed, the MCP overlay file for the system (monolithic) or processor (multiprocessor) is immediately moved to the new family. This file is also moved at system initialization if a DL OVERLAY specification is other than the corresponding <Halt/Load familyname>; other overlay files, however, once allocated on a given family, are never moved. Thus, changing a DL OVERLAY family while the system is running affects only the allocation of new overlay files.

Example:

DL OVERLAY MYPACK

<job#> OVERLAY SET TO MYPACK

The DL BACKUP specification, and a corresponding extension to the "SB" ODT message, allow a site to direct its PRINTER and PUNCH BACKUP files to any desired family. The "SB" extension recognizes "DLBACKUP" as a valid substitute backup specification.

A new task attribute, "BACKUPFAMILY", has been implemented; it may be set only by the MCP or an MCS. Specifying DLBACKUP as a substitute backup medium causes the BACKUPFAMILY in the TASK (if valid, otherwise the current DL BACKUP family specification) to be the family on which printer and punch backup files are opened. The DL BACKUP familyname is stored into the BACKUPFAMILY of the job stack when the job is initiated; task attribute inheritance propagates this BACKUPFAMILY to all stacks initiated under the job. CANDE stores the DL BACKUP familyname at log-on time, and propagates it to all tasks begun under the session. Thus, all backup files for a job/session normally go to a single family, even

if the DL BACKUP family is changed during a job or CANDE session. (This, of course, need not be true if SB specifications were changed during the job/session, and is not true for backup files explicitly directed elsewhere.)

The presence of the specified DL BACKUP family is checked at system initialization time, and whenever an attempt is made to change it via an ODT input. If the family is not present, an RSVP wait allows the user the following options:

1. Mount the family baseunit and "OK" the RSVP wait.
2. "OF" the RSVP wait to keep the missing family as the DL BACKUP specification.
3. "DS" the RSVP wait to disassociate DL BACKUP from the missing family. (At initialization time the DL BACKUP specification defaults to the <Halt/Load familyname>; DSing a wait resulting from an ODT input causes the previous DL BACKUP family specification to be retained.)

The CANDE commands BACK and BDREMOVE automatically find files on DISK, PACK and the session's BACKUPFAMILY.

Example:

DL BACKUP MYPACK

-- -----

<job#> DL BACKUP FAMILY SET TO: MYPACK

The DL JOBS specification allows the operator to specify the family on which JOBDESC is located. No JOBDESC file is actually moved to that family, but all Halt/Loads subsequent to the specification will search for the family. If it is not present, the operator may IL the search to another family (this updates DL to the Iled family) or OF the search (this causes JOBS DL to default to the Halt/Load family).

Example:

DL JOBS ON MYPACK

DISKLOCATION FOR JOBS WILL BE CHANGED

The DL CATALOG specifications allows the operator to specify the catalog family. It works in the same fashion as JOBS.

Example:

DL CATALOG ON MYPACK

DISKLOCATION FOR CATALOG WILL BE CHANGED

The DL LOG specification allows a site to have its SYSTEM/SUMLOG file maintained on any desired family. Efficiency considerations suggest that the DL LOG family chosen for a B6800 Multiprocessor System be directly visible to each local processor, but this need not be the case. SYSTEM/LOGANALYZER has been changed to search the DL LOG family for any log file whose location has not been explicitly specified.

During system initialization, the specified DL LOG family must be present or the DL LOG specification defaults to the <Halt/Load familyname> as a non-IAD file.

When an attempt is made to change the DL LOG specification via an ODT input, an RSVP wait occurs if the newly specified family is not present and the user has the following options:

1. Make the family baseunit available and "OK" the RSVP wait.
2. "OF" or "DS" the RSVP wait to discard the requested change and maintain the previous DL LOG specification.

Assuming any required family is present, MCP action when changing the DL LOG specification is as follows:

1. A Log Release ("LR") is done on the SYSTEM/SUMLOG on the old DL LOG family.
2. If a valid SYSTEM/SUMLOG file already exists on the new DL LOG family, it is continued; otherwise, a new SYSTEM/SUMLOG is opened.

Example:

DL LOG MYPACK

-- -----

<job#> SUMLOG IS ON MYPACK

The DL USERDATA specification, along with a corresponding change to SYSTEM/MAKEUSER, allows a site to maintain its SYSTEM/USERDATAFILE on any desired family.

An attempt to change the DL USERDATA family via an ODT input succeeds provided there was no SYSTEM/USERDATAFILE on the old DL USERDATA family (or the old family was not present). If a SYSTEM/USERDATAFILE does exist, MCP action is as follows:

1. The presence of the new DL USERDATA family is verified; if it is not present, the user may "OF" or "DS" the resulting RSVP wait to abort the change and retain the previous DL USERDATA specification.
2. A check is made to determine whether a SYSTEM/USERDATAFILE already exists on the new family; if it does, an RSVP wait allows the user the following options:
 - a. "DS" the RSVP wait to abort the DL USERDATA change, or
 - b. "RM" the RSVP wait to remove the file, or
 - c. dispose of the file in some other way, and "OK" the wait.
3. The current SYSTEM/USERDATAFILE is copied to the new USERDATA family; the SYSTEM/USERDATAFILE on the old family is renamed "USERDATAFILE/<date-time>". (Should the copy fail, the DL USERDATA change is aborted.)

If there is no SYSTEM/USERDATAFILE on the USERDATA family, or that family is not present, and a non-interactive request is made for userdata information, MCP action is as follows:

1. If there is no family, a "REQUIRES PK" RSVP message is displayed, with OK, DS or OF options. OK causes repeated demand for the family; DS terminates the requestor; OF causes the code to proceed without the family.
2. If the family is present but has no SYSTEM/USERDATAFILE, or if "REQUIRES PK" message was answered "OF", a "NO FILE" RSVP message is displayed. If the "IL" option is used, the USERDATA DL family is set to the familyname of the resulting unit.

Example:

DL USERDATA MYPACK

SYSTEM/USERDATAFILE WAS MOVED TO MYPACK

DP (Dump) Message

DP has been eliminated; see DUMP (Dump Memory) for new message.

DR (Date Reset) Message

Revised Syntax:

-- DR --<mm>-- / --<dd>-- / --<yy>--|

Revised Semantics:

The DR (Date Reset) message may be used to change the date currently in use by the MCP. The desired date is specified by <mm>, <dd>, and <yy>, which are one- to two-digit numeric indications of the month, date, and year, respectively.

The delimiter must be a slash ("/").

New Example:

DR 04/22/75

DATE IS TUESDAY APR 22, 1975 (75112)

(75112) indicates that the year is 1975 and the date is day 112.

DS (Discontinue) Message

Revised Syntax:

```

---<mix number list>-- DS -----|
                        |-----|
                        |-<option list>-|

```

<option list>

```

-----|
|<-----,-----|
|-----|
|  ALL -----|
|-----|
|  - NONE -----|
|-----|
|  - FAULT -----|
|-----|
|  - DSED -----|
|-----|
|  - BASE -----|
|-----|
|  - ARRAYS -----|
|-----|
|  - CODE -----|
|-----|
|  - FILES -----|
|-----|
|  - DBS -----|
|-----|
|  - LIBRARIES -----|
|-----|
|  - PRIVATELIBRARIES -----|
|-----|
|  - SIBS -----|
|-----|

```

Revised Semantics:

With the DS (Discontinue) message, all programs associated with the given mix numbers are terminated or removed from the mix schedule or scheduling queue.

If the program has been initiated and options appear in the DS message, these options will be used to control a program dump taken at the time of termination.

Options for the program dump are as follows:

ALL	Dump all items in the stack and also list code segments.
NONE	Use the default dump options.
FAULT	A program dump will occur only if a fault occurred in the program (divide by zero, segment array, etc.). A program dump is not invoked at the time the message is entered.
DSED	A program dump will occur only if the program is DSed. A program dump is not invoked at the time the message is entered.
BASE	Dump the base of the stack which is used by the operating system.
ARRAYS	Dump all present arrays.
CODE	Dump all code segments.
FILES	Dump all areas used by files.
DBS	Dump SIB and data base stack.
LIBRARIES	Dumps library stack.
PRIVATELIBRARIES	Dumps only private library stacks.
SIBS	Dumps only SIB.

Example:

Unchanged

DUMP (Dump Memory) Message

DUMP replaces the DP (Dump Memory) Message.

Revised Syntax:

```

----- DUMP -----
| - DP --- | | -<nonnumeric text>----- |
|           | | - " <text> " ----- |
| -<mix number list>----- DUMP ----- |
|           | | - DP --- | | -<option list>- |

```

<option list>

```

-----
| |<----- , ----- |
| * ----- |
| - ALL ----- |
| - NONE ----- |
| - FAULT ----- |
| - DSED ----- |
| - BASE ----- |
| - ARRAYS ----- |
| - CODE ----- |
| - FILES ----- |
| - DBS ----- |
| ----- |
| - LIBRARIES ----- |
| - PRIVATELIBRARIES - |
| - SIBS ----- |

```

Semantics:

The DUMP (Dump Memory) message may be used to dump the entire contents of memory to tape or invoke a program dump on a particular program.

When text appears following the DUMP, that string appears in the memory dump tape as the reason for the dump.

Options may be used on a program dump as follows:

- * The options appearing in the list will be ORed with any compiled-in options and the net result used to control the program dump.
- ALL Set all dump options including DSED and FAULT. This will NOT invoke a program dump at the time the message is entered.
- NONE Use the default dump options.
- FAULT A program dump will occur if a fault occurred in the program (divide by zero, segment array, etc.). A program dump is not invoked at the time the message is entered.
- DSED A program dump will occur if the program is DSed. A program dump is not invoked at the time the message is entered.
- BASE Dump the base of the stack which is used by the operating system.
- ARRAYS Dump all present arrays.
- CODE Dump all code segments.

FILES Dump all areas used by files.

DBS Dump SIB and data base stack.

LIBRARIES Dumps library stack.

PRIVATELIBRARIES

Dumps only private library stacks.

SIBS Dumps only SIB.

The DUMP message for a program will set/reset permanently the dump options for this program. If the "*" is not used, all the dump options are reset, except those explicitly specified in the message. All non-dump options are not affected by a DUMP message.

Example:

3132 DUMP ALL

EI (Emergency Interrupt) Message

EI has been eliminated; see HS (Hold Schedule) for new message.

EQ (Eliminate Queue) Message

EQ has been eliminated; see MQ (Make or Modify Queue) for new message.

FA (File Attribute) Message

Revised Syntax:

```

--<mix number list>-- FA ---<file attribute assignment>----|
                        |<----- , ----->|

```

Semantics:

Unchanged

New Example:

7988 FA VERSION=12, CYCLE=13

FM (Form Message) Message

Revised Syntax:

```

--<mix number list>-- FM ---<device>--<unit number>--|

```

Semantics:

Unchanged

Example:

Unchanged

FR (Final Reel) Message

Revised Syntax:

```

--<mix number list>-- FR --|

```

Semantics:

Unchanged

Example:

```
4423 FR
-----
```

FREE Message

Syntax:

```
-- FREE --- SHAREMODS -----<mod range>-----|
| - PRIVATEMODS - |
| - PROC ---<node id>-----|
| -<unit type>--<unit number>--|
```

Semantics:

Existing groups can be altered by the FREE and ACQUIRE messages. The FREE message is used to detach resources from an active group. Devices must be SAVED before they can be "freed". Disk packs must also be closed first.

Example:

```
PER MT
-----
```

```
----- MT STATUS -----
81*P [HALL ] 1600 #1 1:0 MEMORY/DUMP [1,2]
82*P [MIKE ] 1600   S C R A T C H [1,2]
```

```
FREE MT 82
-----
```

UNIT IN USE

```
SV MT 82
-----
```

MT 82 SAVED

```
FREE MT 82
-----
```

MT82 FREED

```
PER MT
-----
```

```
----- MT STATUS -----
81*P [HALL ] 1600 #1 1:0 MEMORY/DUMP [1,2]
82   NOT AVAILABLE TO GROUP
```

FS (Force from Schedule) Message

FS replaces the XS (Exceed Schedule) message.

Syntax:

```
--<mix number list>--- FS ----|
| - XS - |
```

Semantics:

When the FS (Force Schedule) message is entered, the execution of the indicated scheduled job is unconditionally initiated. Also, it may be used to force a job out of a scheduling queue, as well as one initiated but scheduled by the MCP. The system does not display a response to this message.

Example:

7852 FS

GC (Group Configuration) Message

Revised Syntax:

```
-- GC -----|
      | - TO <filename> - |
```

Revised Semantics:

The GC (Group Configuration) message displays the current group configuration.

The GC TO <filename> message causes a disk file to be created containing a "decompiled" configuration group description. This file may then be used, possibly after modification, as input to SYSTEM/CONFIGURATOR to create a new configuration file.

The GC message displays information about the B6900 I/O subsystem showing all DLPs on a base-by-base basis for all BASEs and DLPs on the system. See Mark 32 CONFIGURATOR note D3406 and UTILoader note D3332 for the format of this information.

Example:

GC
--

Response for Multiprocessor System:

***** GROUP CONFIGURATION *****

```
GROUP ID: BLUE
PROCESSOR PORTS: A000  PROC ID. = 1
PERIPHERALS ALLOWED TO GROUP:
  12-13,32-33,42-43,69-71,81-83,193-194
GLOBAL MEMORY STATUS:
PRIVATE MEMORY AVAILABLE: NONE
PRIVATE MEMORY IN USE:  NONE
SHARED MEMORY AVAILABLE:  32-33,38-39,44-45,50-51,56-57,62-63
SHARED MEMORY IN USE:    32-33,38-39,44-45,50-51
```

Response for Monolithic System:

***** GROUP CONFIGURATION *****

```
GROUP ID: DEFAULT
PERIPHERALS ALLOWED TO GROUP:
  1-255
GLOBAL MEMORY STATUS:
PRIVATE MEMORY AVAILABLE: NONE
PRIVATE MEMORY IN USE:    0-63
SHARED MEMORY AVAILABLE:  NONE
SHARED MEMORY IN USE:    NONE
```

Response for B6900 System:

```
GROUP ID: BLUE
PROCESSORS:
  PROC A;
MEMORY:
  PRIVATE MODS 32-59;
  SHAREREAD MODS 60-61;
I/O:
  BASE 2/1/0
    HOST 1
      ADDRESS 0  DLPID 48 HT1
      ADDRESS 1  DLPID 48 HT1
      ADDRESS 3  DLPID 2  ODT1
      ADDRESS 4  DLPID 5  ODT1
      ADDRESS 7  DLPID 70 NSP1;
  BASE 1/1/0
    HOST 1
      ADDRESS 3  DLPID 1  TP1;
    DEPENDENT HOST 4 DLPID 70
      ADDRESS 0  DLPID 71 LSP1
      ADDRESS 1  DLPID 72 LSP1;
PERIPHERALS:
  UNITS 1,2,5,48,49,50;
```

HI (Exceptionevent) Message**Revised Syntax:**

```
--<mix number list>-- HI -----|
                        | -<number>- |
```

Semantics:**Unchanged****Example:**

```
4312 HI
-----
```

HN (Hostname) Message

```
-- HN -----|
      | -<name>- |
```

Each HOST in the network is identified by a HOSTNAME. HOSTNAMEs must be unique; i.e., no two systems in the network may have the same HOSTNAME at the same time. This name is used by programs and users to identify the location of resources to which access is desired. A hostname consists of 1 to 17 alphanumeric characters, beginning with a letter. A system's hostname may be set or modified only at halt/load time. Each host must have a hostname assigned before attempting to establish communication with another host.

HN <name> will establish a hostname at the next halt/load. HN will display the current hostname.

HS (Hold Schedule) Message

HS replaces the EI (Emergency Interrupt) message.

Syntax:

```
----- HS -----|
      | - EI - | | - - - |
      | - ? - - |
```

Semantics:

The HS (Hold Schedule) message stops the selection of jobs from the schedule into execution. HS- resumes job selection. HS? displays the current status of job selection.

Example:

```
HS
---
```

JOB SELECTION STOPPED

and to resume normal operations

```
HS-
---
```

JOB SELECTION RESUMED

```
HS?
---
```

HOLD SCHEDULE IS RESET

HU (Host Usercode) Message

Syntax:

```
-- HU -----|
      |-----|
      |-<usercode>-|
```

An installation has the ability to establish a default usercode for all operations done via the ODT.

HU <usercode> will set the host usercode.

HU will display the current host usercode.

Inquiries from ODTs that have no usercode associated with them, and are directed to another host, will have the host usercode passed along.

ID (Initialize Datacom) Message

ID replaces the DC (Datacom Initiation) message.

Syntax:

```
----- ID ----->
| - DC - | |-<number-1>-| |-----|
|                                     |
|                                     |-<identifier>-|
|                                     | - ON <familyname> -|
|----->
|
| |<-----|
| |<- , -|
|
| : - /1\- QUIT -----|
|
| - /1\- DUMP -----|
|
| - /1\- NSPLEVEL -----<number-2>-|
| | - = - |
|
| |<-----|
| |<- , -|
|
| - /1\- OPTION ----- ALL -----|
| | - = - | | - NONE -----|
| |                                     |<number-3>-|
| | - + - |
| | - - - |
```

Revised Semantics:

The ID ODT message is used for any of the following reasons:

1. To initialize a Data Communications Process (DCP) or Network Support Processor (NSP).
2. To initialize or modify various Data Communications-related items including the prefix for the NIF and DCPCODE files, the suffix for the NSP firmware file and miscellaneous options.
3. To interrogate the current setting of the items mentioned in 2 above.
4. To direct commands to the DCCONTROL process in control of an NSP.

If a number is specified for <number-1>, it is interpreted as a DCP number of multiplexor systems or an NSP unit number for Message Level Interface Port systems. If the specified NSP/DCP is not already running, it will be initialized.

If an identifier appears in the message, that identifier is used as the prefix for the NIF and DCPCODE files used by the NSP/DCP. If a minus (-) appears in the message, the prefix will be set to "SYSTEM".

If QUIT appears in the message, the NSP specified by <number-1> will be shut down. If <number-1> was not specified, all running NSPs will be shut down.

If DUMP appears in the message, the NSP specified by <number-1> will be directed to dump its local memory to the HALTLOAD disk unit in a file titled "DUMP/NSP/<number-1>". If <number-1> is not specified, all running NSPs will be directed to dump their local memories.

If NSPLEVEL appears in the message, the number specified by <number-2> will be used as the suffix for the firmware file used to initialize an NSP. If <number-2> is specified as 0, no suffix will be used. The file title will be "FIRMWARE/NSP/<suffix>". This file is expected to reside on the HALTLOAD disk unit.

If OPTION appears in the message, the option byte will be set as follows:

ALL sets the option byte to all ones and will reset to zero any following options.

NONE sets the option byte to all zeros and will set to one any following options.

<number-3> may be any value from 0 through 7.

<+><number-3> will set that option and any following options.

<-><number-3> will reset that option and any following options.

The following options apply to NSP systems only.

0: AUDIT ALL DCINITIAL INITIATED I/O ACTIVITY TO THE NSP.

1: AUDIT ALL DCCONTROL INITIATED I/O ACTIVITY TO THE NSP.

2: AUDIT ALL DCP FORMATTED REQUESTS AND RESULTS IN DCCONTROL.

3: AUDIT ALL DCWRITE FORMATTED REQUESTS AND RESULTS IN DCCONTROL.

4: WHEN A MEMDUMP IS REQUESTED WITHIN DCCONTROL, AUDIT ALL INTERNAL QUEUE STRUCTURES PRIOR TO TAKING THE DUMP.

5: RESERVED

6: RESERVED

7: AUDIT ALL NSP ERROR CONDITIONS.

The audit activity mentioned above will be written to the HALTLOAD disk unit. This audit trail can then be analyzed using SYSTEM/DCAUDITOR. Since each audited item requires a write to the audit file, the use of auditing (particularly if all datacom messages are audited) can adversely affect system performance.

ID alone will display the values of the current NIF prefix, the next NIF prefix if any, the NSP firmware level, and any options which are set.

IL (Ignore Label) Message

--

Revised Syntax:

--<mix number list>-- IL --<device>--<unit number>--|

Semantics:

Unchanged

Example:

7924 IL MT113

IV (Initialize and Verify) Message

--

IV has been eliminated; see RC (Reconfigure) for new message.

```

----- J -----
| -/1\ - ALL - |
|
|-----|
| |<-----|
|-----/1\ - SWAPPER -----|
|
| -/1\ - MCSNAME -----<mcsname>-----|
|          | - = - |
|
| -/1\ - IN -----<subsystem id>-----|
|          | - ( <processor id list> ) - |

```

←-----	
	←- , -

GLOBAL	
1	-----
2	-----
3	-----
4	-----

The J (Job and Task Structure Display) message lists the tasks by job structure. If ALL is used, any active jobs or tasks which have been suppressed by the SUPPRESS message are displayed in addition to unsuppressed tasks. J does not show jobs in the job queue; the SQ (Show Queue) message serves that purpose.

When MCSNAME is specified, only jobs that originated from the specified MCS will be displayed.

When IN is specified, only jobs with stacks running in the subsystem identified or the processor identified will be displayed.

The mix picture for a job running three tasks in parallel might look like the following:

Note that every task and job has a different mix number. The number 55 is the priority, equal in this case for all the tasks shown.

The different tasks in a job may be in different states. Any of them may be active, scheduled, completed or waiting for operator action. The left margin has a letter flagging any of the job's tasks which are not active, as follows: S for scheduled tasks, W for waiting tasks (RSVP required) and E for compiles which have a syntax error. Completed tasks are not shown within the job structure. An asterisk "*" preceding a job or task indicates the task is displayed for the first time in a particular category. Swap jobs are flagged with an "#" between the priority number and the file name.

A job placed in the reader may never show in the job structure for any of three reasons. First, the job may have finished before a mix display (MX) was requested. In this case, it appears in the completed table if it is one of the list of most recently completed jobs. If Autoprint (AP) is running, the job's wrapup sheet is printed. Second, the job may have been rejected without being run. This condition shows up as a syntax error printout if Autoprint is running. Finally, the job may be in a queue waiting to be initiated, in which case the SQ message will show it.

A typical response to a J message is shown in the example.

Example:

-----JOB STRUCTURE-----

```
* 3044 JOB 50 COPY & COMPARE SY
*W.. 3045 50 LIBRARY/MAINTENANCE
   3072 JOB 60 INTERIMFR26
   E.. 3041 50 SYSTEM/REL/ALGOL ON INTERIM26
*E.. 3060 50#ALGOL (JONES) CANDE/CODE60
   W.. 3080 50 ALGOL (GORD) SYSTEM/PL1
*S.. 3097 50 PL/1 (GORD)CANDE/CODE1080
```

Swap jobs are flagged with an "#" between the priority number and the file name.

New Example:

J ALL MCSNAME=SYSTEM/CANDE

This message will display all active jobs (including suppressed) that originated from SYSTEM/CANDE.

New Example: (B6800 Multi-Processor System):

J SW IN(3)

This message will display all active jobs (excluding suppressed) whose stacks are running in swapspace and are currently swapped into processor 3.

LH (Load Host) Message

Syntax:

```
-- LH -- PK --<unit no>-- MPX <mpx no> PATH <path no> ----->
                               | - PATHID <pathid no> -----|
>-----|
| -<filetitle>-|
```

Revised Semantics:

In the LH ODT message, a particular path can be specified for B6700 and B6800 systems by either MPX PATH or PATH ID (blanks are required between MPX and <mpx no> and PATH and <path no>). On B6900 systems, only PATHID is accepted; any attempt to use MPX PATH is rejected with an "INVALID SYNTAX" message.

Examples:

```
LH PK 60 MPX 1 PATH 2
LH PK 60 PATHID 6
LH PK 60 PATHID 14 SYSTEM/NEWFIRMWARE
```

LJ (Log to Joblog) Message

Revised syntax:

```
--<mix number list>-- LJ --<text>--|
```

Semantics:

Unchanged

Example:

Unchanged

LP (Lock Program) Message

Syntax:

```
--<mix number list>-- LP -----|
                        | - - - |
```

Semantics:

The LP (Lock Program) message is provided to disable certain ODT messages which may interfere with program execution. Once locked, the DS and QT messages will be invalid for the locked task.

LP- removes the LP restriction.

Example:

7094 LP

7094 PROGRAM LOCKED.

7094 LP -

7094 NOT LOCKED.

LR (Log Release) Message

LR has been eliminated; see TL (Transfer Log) for new message.

M (Mix Entries) Message

M has been eliminated; see MX (Mix Entries) for new message.

MA (May Access) Message

Syntax:

```
-- MA ----- APL --<filename>--|
                        | - - - |
```

Semantics:

MA APL makes the specified <filename> accessible to APL only, by setting the file attribute "APL" in the file. MA - APL denies APL access to the specified <filename>

When <filename> is a codefile, the effect of the MA command is to allow any process running from that codefile to access data files which also have been restricted, via MA. Also, the codefile has the ability to create files with the "APL" attribute set.

Examples:

MA APL MYFILE

GIVEN APL ACCESS MYFILE

MA - APL MYFILE

DENIED APL ACCESS MYFILE

PD MYFILE ON PACK

FILE MYFILE on PACK (ALGOLSYMBOL)
 DATE CREATED: Wednesday APR 09, 1979 (79099)
 DATE OF LAST ACCESS: Wednesday APR 09, 1979 (79099)
 SIZE IN SEGMENTS: 10
 SECURITY: PUBLIC-USAGE: READ/WRITE

The file attribute APL has been implemented for disk/pack files to satisfy APL access restrictions. It may be set prior to the creation of a new file, and is effective only if the codefile creating the file itself has the attribute set. The attribute may be read any time the file is open. An attempt to open a file with APL set causes the requestor to be DSed unless it also has APL set.

MIXL (Mixlimit) Message

MIXL has been eliminated; see ML (Mixlimit) for new message.

ML (Mixlimit) Message

ML replaces the MIXL (Mixlimit) message.

Revised Syntax:

```

--- ML -----<number>---|
| - MIXL - | - = - |
|           | - - - - - |

```

Semantics:

The ML (Mixlimit) message is used to set or interrogate the current mixlimit. The system responds to the interrogation ML by displaying queue class, active count, mixlimit and the number of jobs queued for every job queue. If a default queue has been set, the letter D appears in the left margin of the display for that queue. A typical response to a ML message is as follows:

	QUEUE	ACTIVE	LIMIT	QUEUED
D	0	2	10	0
	3	0	2	0
	5	4	4	1
	TOTAL	7	NONE	

ML <number> sets the mixlimit used for introducing new jobs into the system. This message establishes the maximum number of jobs which may be introduced regardless of the sum of all mixlimits set for all queues. Setting the mixlimit equal to zero will allow no jobs to be run. The limits set on each queue need not be changed.

Example:

ML 20

	QUEUE	ACTIVE	LIMIT	QUEUED
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
.	TOTAL	7	20	

ML- removes the mixlimit.

Example:

ML-

	QUEUE	ACTIVE	LIMIT	QUEUED
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

TOTAL 7 NONE

MODE (In or Output Mode) Message

Syntax:

Unchanged

Revised Semantics:

The MODE message is used to control the input/output capability of the indicated device. The IN option (read only) prevents new files from being created, but it does not prevent writes to the given unit. Old files can be updated and/or removed even if the IN option is set. The IO and OUT options allow the device to revert to normal operation. The MODE message allows the operator to inform the MCP that the WRITE ENABLE switch on a given disk pack unit has been changed.

Example:

MODE PK 096 OUT

PK096 MODE IS OUT

This allows files to be opened for output.

GETUSERDISK gives a "SECTORS REQUIRED" message if a request is made for space on a write locked out disk pack.

MQ (Make or Modify Queue) Message

In addition to the pre-3.1 MQ syntax, the EQ (Eliminate Queue) option has been incorporated.

Revised Syntax:

```
-- MQ ---<number>-----|
|                               |
|                               |<----- , -----|
|                               |---<queue attributes>---|
|                               |
| - - --<number>-----|
```

The syntax and semantics for <queue attributes> are unchanged, except as follows:

A new attribute has been added to the list of queue attributes,

Syntax for SUBSYSTEM:

```
--/1\- SUBSYSTEM -- = --<subsystem id>--|
```

Semantics for SUBSYSTEM:

The SUBSYSTEM attribute specifies the memory structures to be used for the jobs and tasks initiated through those queues.

Semantics:

The entry of the MQ (Make or Modify Queue) message results in the modification or creation of a job queue identified by the number following the MQ.

MQ- <number> eliminates the specified queue from the system.

Example:

MQ 37 MIXL = 2, SUBSYSTEM=REDBLUE

```
QUEUE 37:
MIXLIMIT = 2
SUBSYSTEM=REDBLUE
DEFAULTS:
NONE
LIMITS:
```

MS (Make Subsystem) Message

```

----- MS -----
|
|-----<subsystem id>-----|
| - - - |
|-----<subsystem id>----- ( --<processor id list>-- ) -|
|           |
|           |

```

←	
←	, -
GLOBAL	
- 1	-----
- 2	-----
- 3	-----
- 4	-----

MS `<subsystem id>=(<processor id list>)` defines or redefines a subsystem.

Example :

```

MS SYSTEM      = (GLOBAL, 1, 2, 3, 4) ,      CURRENTLY (GLOBAL, 1, 2)
MS GLOBAL      = (GLOBAL) ,                  CURRENTLY (GLOBAL)
MS TWO         = (2) ,                        CURRENTLY (2)
MS BARBARA     = (2) ,                        CURRENTLY (2)
MS RED         = (3) ,                        CURRENTLY EMPTY
MS YELLOW      = (3) ,                        CURRENTLY EMPTY
MS BLUE        = (1) ,                        CURRENTLY (1)

```

MS JUNK = (2) , CURRENTLY (2)

JUNK REMOVED

MU (Make User) Message

MU incorporates the PU (Privileged User) message.

Syntax:

```
-- MU ---<identifier>-----
|                                     |
|                                     | - / <identifier> - | - PRIVILEGED - |
|                                     |                                     |
| - - --<identifier>-- PRIVILEGED -----
```

Semantics:

The MU (Make User) message causes the first specified identifier to be entered into the userdirectory as a valid usercode, and the second specified identifier (if any) to be associated as a password with that usercode. The usercode may be specified as a privileged user. A usercode preceded by a minus sign has privileges removed from it. The MU action is subject to regulation by the "MU MODEL" entry in the SYSTEM/USERDATAFILE; MU may be disallowed entirely, or the PRIVILEGED option may be disabled.

Example:

MU JOHN/DOE

JOHN/DOE CREATED

MU- JOHN PRIVILEGED

JOHN NOT PRIVILEGED

MX (Mix Entries) Message

MX replaces the M (Mix) message.

Revised Syntax:

```
-- MX ----->
|                                     |
| - /1\- ALL - |
|                                     |
|-----|
| |<-----|
| |-----|
| - /1\- SWAPPER -----|
| |-----|
| - /1\- MCSNAME -----<mcsname>-----|
| |-----|
| |-----|
| - /1\- IN ---<subsystem id>-----|
| |-----|
| | - ( <processor id list> ) - |
```

<processor id list>

```
|<-----|
| |<- , -|
|-----|
| GLOBAL -----|
| |-----|
| - 1 -----|
| - 2 -----|
| - 3 -----|
| - 4 -----|
```

Revised Semantics:

The MX (Mix Entries) request yields the same response as the J (Job and Task Structure Display) request, except that display lines (RSVP messages and DISPLAY) are printed with each task. If ALL is used, any active jobs or tasks which have been suppressed by the SUPPRESS message are displayed in addition to unsuppressed tasks.

When SWAPPER is specified, only jobs running in swapspace will be displayed.

When MCSNAME is specified, only jobs that originated from the specified MCS will be displayed.

On a B6800 multiprocessor system, each displayed task is preceded by a subsystem indicator: processor id for a local memory task, "G" for a global memory task, or blank for a task whose subsystem location is currently unassigned.

When IN is specified, only jobs with stacks running in the subsystem identified or the processor identified will be displayed.

Example:

MX

-----JOB STRUCTURE-----

```

S4087 JOB 50 ?RUN DICKEYIN("SUP
4070 JOB 50? RUN DATABASE/DU
S..L4083 50 DATABASE/DUMPANALYZER ON DMS
4053 JOB 70 SYSTEM/CANDE
D:DISPLAY:#
E..4075 60 DMALGOL ON DMS (JKD) CANDE/CODE1100
*S..4120 50 DMALGOL ON DMS
3643 JOB 60 INTERIMFR26
D: (JHHH) JHHHPATCH REMOVED ON INTERIM26 PK065
W..4128 60 LIBRARY/MAINTENANCE
R: RECOPY REQD: SYSTEM/FORTRAN
D: COMPARE ERROR: MT114*
```

Example:

MX ALL MCSNAME=SYSTEM/CANDE

This message will display all active jobs (including suppressed) that originated from SYSTEM/CANDE.

Example: (B6800 Multi-Processor System):

MX SW IN(3)

This message will display all active jobs (excluding suppressed) whose stacks are running in swapspace and are currently swapped into processor 3.

NET (Network) Message

Revised Syntax:

```

-- NETWORK -----|
---|
| - = --- *NULL ---|
|   | - *DEFAULT -|
|   | -<filename>-|
| - + --- *NULL ---|
|   | - *DEFAULT -|
|   | -<filename>-|
| - - - - -|
|   | - NOW -----|
```

Revised Semantics:

The SL (System Library) command is used to specify the file title of the networking library. See the description of the SL command for details of implementation.

NET= specifies whether or not an initialization file is to be used when starting up network services.

*NULL specifies that no initialization file is to be used.

*DEFAULT specifies that the default initialization file (SYSTEM/NETINIT) is to be used.

<filename> specifies that an initialization file other than the default file is to be used.

NET interrogates the setting of the network initialization file.

NET+ initiates network services and optionally specifies the network initialization in the same manner as NET=.

NET- terminates network services. The default (i.e., no parameters) is to initiate a slow shutdown which waits until all port files to remote hosts have been closed, after which network services is terminated.

NET-NOW closes all port files to remote hosts and shuts down network services immediately.

NEXT (Next Screen) Message

NEXT has been eliminated; see NS (Next Screen) for new message.

NS (Next Screen) Message

Syntax:

```

----- NS -----|
| - NEXT - |

```

Semantics:

The function of NS (Next Screen) is to bring up the next screen on a screen terminal if there is a next screen. The message is automatically typed in and the cursor positioned at the lower right corner of the screen.

NW (Network Prefix)

Any text preceded by the NW message is passed to the BNA network operator for parsing if the network is running. The CONTROLLER does not parse anything beyond the NW. The following message is displayed if NW is entered and the network is not running: "LOCAL HOST IS NOT NETWORKING".

OF (Optional File) Message

Revised Syntax:

```
--<mix number list>-- OF --|
```

Semantics:

Unchanged

Example:

Unchanged

OG (Overlay Goal) Message

Revised Syntax:

```

--<mix number list>-- OG -----|
                        | -<number>- |

```

Semantics:

Unchanged

Examples:

```

4869,5269 OG 18
-----

```

```

4869 OVERLAY GOAL=18%
5269 OVERLAY GOAL=18%

```

```

5269 OG
-----

```

```

5269 OVERLAY GOAL=11% PER MINUTE

```

```

4869 OG 20
-----

```

```

4869 OVERLAY GOAL=20%

```

OK (Reactivate) Message

Revised Syntax:

```

--<mix number list>-- OK --|

```

Semantics:

Unchanged

Examples:

```

0963 OK
-----

```

```

0931,0935 OK
-----

```

OL (Label Table) Message

With the new GETSTATUS I/O path information interface, the path display that results from the OL ODT message has been modified. The information displayed depends on the system type.

New Examples:

B6800, B6700 SYSTEM/CONTROLLER Path Display Modification

A new "PATHID" column precedes the "MPX" column.

Example ODT entry:

```

-----
OL PK192

```

```

Display:
-----

```

```

PK 192*B [250265] #1 DATACOM (08) [1,2,3]
  CREATED ON: 04/16/80 AT 00:28:12
  BX385 235
  FIRMWARE=D484
  PATHID  MPX  PATH  STATUS
    00     1     0  ONLINE
    01     1     1  ONLINE
    02     1     2  ONLINE
    03     1     3  ONLINE

```

B6900 SYSTEM/CONTROLLER Path Display Implementation

Given the differences in the new hardware, the path information displayed by SYSTEM/CONTROLLER on a MLIP system is different from that displayed on the B6800 or B6700. Note that firmware level information is now maintained for each path.

Example ODT entry:

OL PK50

Display:

```
PK 50*B [001640] #1 YELLOW (00)
    CREATED ON: 02/21/80 AT 07:39:18
    235
    PATHID  FIRMWARE  PROC  HDPPORT  LEMPORT  DLPNUM  PATHSTATUS
      05      D484      1      2          0          2      ONLINE
```

If the path to the unit is via an outboard host (e.g., NSP DLP), a "HOSTDLP" column will follow the "PATHID" column; the entries in this column will be the physical unit number(s) of the controlling DLP(s).

If there is no firmware level information, the "FIRMWARE" column is omitted from the display.

OP (Options) Message

OP incorporates the SO (Set Options), RO (Reset Options) and TO (Test Options) messages.

```
-- OP -----|
      | - + - | | -<option list>- |
      | - - - |
```

<option list>

<----->

<number>

```

- OPEN -----
- TERMINATE -----
- NOCHECK -----
- LPBDONLY -----
- AUTORM -----
- DIAGNOSTICS -----
- CDONLY -----
- AUTORECOVERY ----
- DUPSUPERVISOR ---
- DUPINTRINSICS ---
- AUTODC -----
- NODUMP -----
- CPBDONLY -----
- CRUNCH -----
- BACKUPBYJOBNR ---
- FULLTRANSLATION -
- NOFETCH -----
- RESOURCECHECK ---
- NOSUMMARY -----
- DIRDEBUG -----
- CATALOGING -----
- NEWPERETRY -----
- OKTIMEANDDATE ---
- LOGPOSITIONING --
- SERIALNUMBER ----
- ARCHIVING -----
- CONTROLOLDWFL ---
- IORANGECHECK ----
- SWAPALLJOBS -----
- NORVRSPAPERTAPE -
- IODIAGNOSTICS ---
- USECATDEFAULT ---
- CATTEST -----
- MCPTEST -----

```

Semantics:

The OP (Options) message displays, sets and resets options as follows:

OP Displays all options and their respective states.

OP <option list> Displays the options in the list and their respective states.

OP+ Displays all options that are set.
 OP+<option list> Sets all options in the option list.
 OP- Displays all options that are reset.
 OP-<option list> Resets the options in the option list.

Revised Example:

An asterisk in the following examples indicates that the option is set.

OP

--

```

----- OPTIONS -----
1 OPEN          2*TERMINATE
3 NOCHECK       4*LPBDONLY
5*AUTORM        6 DIAGNOSTICS
7 CONLY         8*AUTORECOVERY
9 DUPSUPERVISOR 10 DUPINTRINSICS
12*AUTODC       13 NODUMP
14*CPBDONLY     16*CRUNCH
17 BACKUPBYJOBNR 18 FULLTRANSLATION
19*NOFETCH      20*RESOURCECHECK
21*NOSUMMARY    22 DIRDEBUG
23*CATALOGING   24 OKTIMEANDDATE
25*NEWPERETRY   26*LOGPOSITIONING
27*SERIALNUMBER 28*ARCHIVING
29 CONTROLOLDWFL 31 IORANGECHECK
32 SWAPALLJOBS  33 NORVRSPAPERTAPE
43 IODIAGNOSTICS 45*USECATDEFAULT
46*CATEST       47 MCPTEST

```

OP + OPEN

1 OPEN SET

OP -

```

----- RESET OPTIONS -----
1 OPEN          3 NOCHECK
6 DIAGNOSTICS   7 CONLY
9 DUPSUPERVISOR 10 DUPINTRINSICS
13 NODUMP       17 BACKUPBYJOBNR
18 FULLTRANSLATION 22 DIRDEBUG
24 OKTIMEANDDATE 29 CONTROLOLDWFL
31 IORANGECHECK 43 IODIAGNOSTICS
47 MCPTEST

```

OP - OPEN

1 OPEN RESET

The Mark 32 run-time system options are as follows:

<number>

A number appearing in this message must correspond to the number which identifies the option desired. (The run-time system options are not to be confused with the option word assigned to each job.)

OPEN (option 1)

When this option is set, a file-open message is displayed for each job whenever a file is opened.

TERMINATE (option 2)

When this option is set, abnormal job terminations result in an attempted program dump rather than a full memory dump. If this option is reset, such abnormal terminations result in a full memory dump.

NOCHECK (option 3)

When this option is set, memory dumps under both abnormal termination and FORGETCHECK conditions are inhibited. These dumps are automatic when NOCHECK is reset.

LPDONLY (option 4)

When this option is set, all printer output files are assigned to printer backup disk. These files can then be printed by the AUTOBACKUP routine.

AUTORM (option 5)

When this option is set, the MCP automatically removes the old file when a duplicate-file condition occurs. When AUTORM is reset, an RM or OF message is required when such a condition occurs.

DIAGNOSTICS (option 6)

When this option is set, an RSVP message (e.g., RF DEGRADATION) is displayed at the console any time the reliability of a hardware unit is degraded by a set amount.

CDONLY (option 7)

With this option set, any job opening card input which is not internal to the jobfile is DSed. Also, no card reader may be labeled.

AUTORECOVERY (option 8)

When this option is set, a Halt/Load is attempted following all system fatal memory dumps (except a hung processor). DCPs and NSPs which were running prior to the Halt/Load are subsequently reinitialized and the AP number is restored to the value previous to the halt.

If this option is reset, the above conditions do not happen. Furthermore, the mix limit of all queues is set to zero so that no jobs will be automatically restarted.

DUPSUPERVISOR (option 9)

This option is provided for use with directory reconstruction. If a code file <file title> has been designated as the supervisor program by a CS message and this option is set, at Halt/Load time the MCP will attempt to execute a code file <file title>/FMLYINX<nnn>. If this option is reset, the MCP will attempt to execute the designated supervisor program.

DUPINTRINSICS (option 10)

This option is provided for use with directory reconstruction. If a file <file title> has been designated as the intrinsics file by a CI message and this option is set, at Halt/Load time the system will attempt to use as the intrinsics file the code file <file title>/<family index>. If this option is reset, the code file <file title> will be used as the intrinsics file at Halt/Load time, regardless of the EU involved.

AUTODC (option 12)

When this option is set, the automatic generation of a Data Comm control stack is provided for when an executing job requests it.

NODUMP (option 13)

When this option is set, the MCP is prevented from attempting dumps to tape. Potential nonfatal dumps are denoted by a message at the supervisory console and logged. The source of a fatal dump is displayed in a system message at Halt/Load time. When this option is reset, dumps are taken in the normal fashion.

CPBDONLY (option 14)

When this option is set, all card punch output files are assigned to punch backup disk. These files can be punched by the AUTOBACKUP routine.

CRUNCH (option 16)

When this option is set, code files and backup disk files are automatically CRUNCHED when they are closed. Note that if this option is reset, no file can be CRUNCHED although the file may have been explicitly CRUNCHED by other program constructs.

BACKUPBYJOBNR (option 17)

When this option is set, jobs are printed by order of the job number. When reset, jobs are printed by order of lowest output print quantity to highest output print quantity.

FULLTRANSLATION (option 18)

When this option is set on B6700/B6800, B7700/B7800 systems, every logical file is initialized with the file attribute TRANSLATE set to FULLTRANS. This allows software translation whenever translation is required and hardware translation is not provided.

On new systems, beginning with the B6900, this option will not control the default setting of FULLTRANSLATION, which will have a default value of FULLTRANS.

NOFETCH (option 19)

When this option is set, FETCH statements entered by a WFL deck are disabled.

RESOURCECHECK (option 20)

When this option is set, it enables tape resource management.

NOSUMMARY (option 21)

When this option is set, the job summary output is suppressed if no backup files are produced. The job summary is printed if a task terminates abnormally.

DIRDEBUG (option 22)

CAUTION

This option is intended for use only by the Burroughs Large Systems Plant.

CATALOGING (option 23)

At Halt/Load time CATALOGING is tested. If it is true, CATALOGLEVEL is initialized to CATALOGLEVELSET; if it is false, CATALOGLEVEL is set to zero. Note that a CATALOGING MCP is any MCP whose CATALOGLEVEL is greater than zero.

Note: A Halt/Load must occur after this option is set for CATALOGING to take effect.

OKTIMEANDDATE (option 24)

This option is used to verify that system TIME and DATE values are valid.

Verification will be required on B6800 multiprocessor systems regardless of the option setting if the clock of any processor varies from that of the leader by more than 60 seconds during the system initialization sequence. (Whenever B6800 time-of-day registers disagree as a multiprocessor system is initialized, they are all synchronized at the maximum time found in any of them.)

When verification is required, the current TIME and DATE settings will be displayed on the ODT and updated whenever ADM would ordinarily be updated. The operator may enter "TIMEOK" to resume normal processing after resetting any invalid time or date via the "DR" and "TR" messages.

NEWPERETRY (option 25)

This option invokes a special method of repositioning the tape after a write error occurs on any PE tape unit. Do not set NEWPERETRY unless all PE tape controls have attained the following RIN level:

CE-L3486-1
CON-5-L2189-6
MECH-R2254-19

LOGPOSITIONING (option 26)

When this option is set, the MCP will log the positioning actions of tape parity retry as well as the actual retries. This option should be set by a site having trouble in retrying tape errors; e.g., lost blocks on tape.

SERIALNUMBER (option 27)

When this option is set, scratch tapes will not be assigned to output tapes unless the SERIALNO attribute is used or the operator OUs the tape.

ARCHIVING (option 28)

This option enables the MCP archiving function. If this option is set and the catalog level of the system is greater than zero, an archive log will be created in which pertinent information will be stored for later processing by the SYSTEM/ARCHIVE utility. The name of the archive log will be "ARCHIVELOG/<date>/<time>", where <date> and <time> refer to the creation date and time of the file. If the archive log is not successfully set up, the option will automatically be reset by the system.

CONTROLOLDWFL (option 29)

When this option is set, all WFL input from the ODT is treated as new WFL. This option affects only job syntax entered through the ODT.

IORANGECHECK (option 31)

This option verifies that the disk address requested for an I/O is, in fact, within the range of any other rows of the file.

NORVRSPAPERTAPE (option 33)

When this option is set, paper tape-parity-retry action is disallowed at installations using the nonreversible Facit paper tape reader.

SWAPALLJOBS (option 32)

When this option is set, all jobs stacks are run in swapspace without forcing inheritance of the SUBSPACES attribute on the tasks run out of a job stack; thus, an installation may choose to run all job stacks in swapspace, but not other stacks.

IODIAGNOSTICS (option 43)

CAUTION

This option is intended for use only by the Burroughs Large Systems Plant.

USECATDEFAULT (option 45)

At Halt/Load time, this option and cataloging are tested. If both are true, all files have the attribute USECATALOG set to true by default.

CATTEST (option 46)

CAUTION

This option is intended for use only by the Burroughs Large Systems Plant.

MCPTEST (option 47)

CAUTION

This option is intended for use only by the Burroughs Large Systems Plant.

OT (Stack Cell Inspection) Message

Revised Syntax:

--<mix number>-- OT --<number>--|

Semantics:

Unchanged

Examples:

Unchanged

OU (Output Unit) Message

Revised Syntax:

--<mix number list>-- OU --<output device>--|

Semantics:

Unchanged

Examples:

Unchanged

P (Peripheral Status) Message

P has been eliminated; see PER (Peripheral Status) for new message.

PC (Print Configuration) Message

PC has been eliminated; see SC (System Configuration) for new message.

PER (Peripheral Status) Message

PER replaces the P (Peripheral) message.

Revised Syntax:

```
-- PER -----|
|               |
|   <-----, -----|
|               |
|   <device>-----|
|               |
|   = -----|
|   <device>---- = ---|
|               |
|   - - - -|
```

<device>

```
----- CP -----|
| - CR -|
| - DC -|
| - DK -|
| - PK -|
| - LP -|
| - MT -|
| - PP -|
| - PR -|
| - SC -|
```

Revised Semantics:

When the PER (Peripheral Status) message is entered, the status of the specified peripheral unit(s) is displayed on the supervisory console at which the message was entered.

If PER= or PER <device>= is used, unlabeled and not-ready devices are included; otherwise, they are not.

On a B6800 multiprocessor system, the display for each unit is followed by a bracketed list of processor numbers showing which subsystems are connected to the unit.

DC has been added as a device type on the B6900; it designates a datacom type device (NSP or LSP DLP).

The remainder of the semantics remain unchanged.

New Example: (B6800 Multiprocessor System)

PER PK

```
-----
----- PK STATUS -----
64 NOT AVAILABLE TO GROUP
66 NOT AVAILABLE TO GROUP
68*B [152000] #1 DISK (26)[1,2]
70*B [144732] #1 PACK (02)[1,2]
71*B [004092] #1 USERS (00)[1,2]
```

PG (Purge) Message

The PG (Purge) message now allows the specification of tape density.

Revised Syntax:

```

----- PG ----- MT -----<unit number list>-----|
| - PGL - | | - PK - | | - ( --<density>-- ) - |

```

<density>

```

----- 200 -----|
| - 556 -- |
| - 800 -- |
| - 1600 - |
| - 6250 - |

```

Revised Semantics:

The PG (Purge) message may be used to purge specified tape units or disk packs if they are ready, not in use, and are write-enabled. For tapes, use of this message requires that the tape have serial numbers; these serial numbers are not disturbed by the message. If the tapes in question do not have serial numbers, use of the SN message is required. The density of the tape may be specified; however, when the density is specified, it applies to all tapes in the list.

The PGL form of this message, in addition, causes the specified tape units to be locked so that no job can automatically pick up the scratch tapes.

New Example:

PG MT 82 (1600)

MT82 WILL BE PURGED

For disk packs, the PG message causes the pack to be relabeled with the packname SCRATCH and made available to SCR programs. This message, intended primarily for maintenance, should not be used for disk packs in the normal course of work.

PP (Privileged Program) Message

Syntax:

```

--- PP ---<file title>-----|
|                               | - :TRANSPARENT - |
| - --<file title>-----|

```

Semantics:

The PP (Privileged Program) message designates a code file as a privileged program. PP without the TRANSPARENT option gives the program the same capability that it would have running under a privileged usercode. A code file title preceded by a minus sign has privileges removed from it.

Example:

PP (MIKE)OBJECT/HARD

(MIKE)OBJECT/HARD IS A PRIVILEGED PROGRAM

PP- (MIKE)OBJECT/HARD

(MIKE)OBJECT/HARD NON PRIVILEGED PROGRAM

The TRANSPARENT option confers upon the code file a new type of privilege called "privileged transparent". In order to explain this feature, the interaction of code files with respect to privilege needs to be clarified.

Each program has a code file privilege specification and a dynamic run-time privilege. Formerly, the code file was marked either privileged or non-privileged. The run-time privilege was determined as follows:

1. If a program ran under a privileged user code, it was privileged throughout its execution.
2. If a program did not run under a privileged user code, its privileged status was the privileged status of the environment it was running in, which was in turn determined by the PP specifications of the code file containing the environment's code.

Examples:

1. A non-privileged program which calls a privileged library was privileged while executing the library procedure.
2. A privileged program which called a non-privileged library procedure was non-privileged while running in the library procedure.

The above description is still true for Mark 32. However, with the TRANSPARENT option, a code file may now be marked privileged transparent. This means that any code from this code file will inherit the privilege of its caller. ("Calling" in this context refers only to procedure or subroutine calls and not task initiation such as PROCESS, RUN, etc.) This is very useful for centralized libraries which are called by both privileged and non-privileged users.

Examples:

1. A non-privileged program calling a privileged transparent library will be non-privileged in the library code.
2. A privileged program calling a privileged transparent library will remain privileged in the library code.

A code file can now have three states:

1. Non-privileged
2. Privileged
3. Privileged transparent

PP <file title> causes the program to have a normal privileged status.

PP <file title>:TRANSPARENT causes the program to have a privileged transparent status.

PP- <file title> cancels the privileged status of a program (either normal or transparent), setting it to non-privileged. The default status is non-privileged.

When a privileged transparent program is initiated as a separate running task, its privilege is the privilege of the user code it is running under, and does not inherit any privilege from its ancestor.

PR (Priority) Message

Revised Syntax:

--<mix number list>-- PR --<number>--|

Semantics:

Unchanged

Examples:

4972, 4980 PR 90

4975 PR 30

PU (Privileged User) Message

PU has been eliminated; see MU (Make User) for new message.

QT (Quit) Message

Revised Syntax:

```
--<mix number list>-- QT --|
```

Semantics:

Unchanged

Example:

```
6823 QT
-----
```

RC (Reconfigure) Message

In addition to the pre-3.1 syntax, the IV (Initialize and Verify) message has been incorporated.

Revised Syntax:

```
-- RC --- PK ---<unit number>----->
      | - DK - |           | - IV - |
>-----|
      | |<-----|
      | |<parameter options>-----|
```

<parameter options>

```
-----/1\- OWNER -----<name>-----|
      | - = - |
- /1\- NAME -----<packname>-----|
      | - = - |
- /1\- SERIAL -----<integer>-----|
      | - = - |
- /1\- BP -----<integer>-----|
      | - = - |
- /1\- FAMILYINDEX -- = --<number>-----|
                                   | - KEEP -
- /1\- IC -----|
- /1\- SR -----|
- /1\- IAD -----|
```

Revised Semantics:

The RC (Reconfigure) message is used to create a new set of volume labels on a disk pack or head-per-track disk. If the disk pack is to be a native mode base pack or interchange pack, a new master directory is also created. The IV option may be used to format and analyze all of the tracks of the specified pack.

The semantics for <parameter options> is unchanged.

RECONFIGURE Message

Syntax:


```

-- RECONFIGURE -- INSTALLATION -- AS ---<group list>-----|
| - GROUP -----| | - DEFAULT -----|
| - INSTALLATION -- AS ---<installation id>--|
| | - DEFAULT -----|

```

Semantics:

The RECONFIGURE message regroups the hardware resources according to the groups specified in the group list. If INSTALLATION is specified, all installation resources are reconfigured to the new configuration. If GROUP is specified, all resources assigned to the group to which the operator display terminal at which the message was entered will be reconfigured into the new configuration. A reconfiguration causes a Halt/Load of existing groups which are being reconfigured into new groups.

Any subset of the <group id>s defined in the configuration file may be specified in the new configuration. A <group id> cannot be specified more than once.

Example:

```

RECONFIGURE INSTALLATION AS THREEBY
RECONFIGURE INSTALLATION AS SYSA, SYSB, SYSC
RECONFIGURE GROUP AS TWOBY
RECONFIGURE GROUP AS RED

```

RES (Reserve) Message

RES incorporates the RET (Return) Message.

Revised Syntax:

```

-- RES ----->
>-- DK ---<unit number>--<reserve options>-----|
| - PK -| | - COPY ERRORS -|
| - EU -| | - REMOVE -----|
| - - -- DK ---<unit number>--<su or switch>-----|
| | - PK -|
| | - EU -|

```

<reserve options>

```

-----|
| -<su or switch>-| | - AS BADDISK -----|
| - AS MAINT -----|
| - LABEL -----|
| - SEGMENT --<number>-----|
| | - FOR --<number>-----|
| | - THRU -----<number>-|
| | - SEGMENT -|

```

```

|<----- , -----|
|      |<--- , ---|
|----- SU -----<number>-----|
| SU --<number>----- THRU -----<number>-----|
|      |      | - SU - |
|      | - FOR -----|
|<----- , -----|
|      |<--- , ---|
|----- SWITCH -----<number>-----|
|      |<number>----- THRU -----<number>-----| |
|      |      | - SWITCH - |
|      | - FOR -----|

```

Revised Syntax:

```
--<mix number>-- RM --|
```

Semantics:

Unchanged

Example:

Unchanged

RO (Reset Option) Message

RO has been eliminated; see OP (Options) for new message.

RR (Release Reader) Message

RR has been eliminated; see SR (Secure Reader) for new message.

RY (Ready) Message

In addition to the pre-3.1 syntax, the RY message has been modified to accommodate multiprocessor systems.

Revised Syntax:

```

      |<----- , -----|
-- RY -----<device>--<unit number list>-----|
      |
      | - CPU --<processor number>-----|
      | - MOD --<mod number>-----|
      |
      | - IN <processor number> - |

```

Revised Semantics:

The RY (Ready) message causes the indicated devices to be made ready for use by the system if they are in remote status and have been made previously inaccessible via the SV message or rewind and lock. If the <mod number> is greater than or equal to the first mod of * GLOBAL tm Memory, the IN phrase is not used; global is used by default.

An off-line processor (CPU) may also be restored by the RY message as part of the restoring sequence, as follows:

- a. Place the processor in LOCAL.
- b. Enter RY CPU<nnn>. The system will display the following typical message:

IC20(ZERO)30(some hexadecimal starting value); IIHF START

- c. Make sure the processor is in LOCAL.
- d. Set the B-register to hexadecimal 20. Note that everything in the registers must be right-justified.
- e. Set the A-register to hexadecimal 0.
- f. Set AROF and BROF.
- g. Do a WRITE-IC.
- h. Set the B-register to hexadecimal 30.
- i. Set the A-register to the starting value.
- j. Set AROF and BROF.
- k. Do a WRITE-IC.
- l. Set IIHF.
- m. Set the processor to REMOTE.
- n. Make certain that SCII, SCC1 and SCC2 are cycling.
- o. Press START.

The RY CPU message cannot currently be used on B6800 or B6900 multiprocessor systems, due to the architecture and relationship of CPUs to I/Os in the B6800 and B6900; this restriction may be removed on a future software release.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

Revised Example:

RY MT 113

MT 113 READY

RY MOD 11 IN 2

MOD 11 IN LOCAL 2 WILL BE READY

S (Scheduled Mix Entries) Message

Revised Syntax:

```
-- S ----->
  | -/1\- ALL - |
  |
  | <-----|
  | |
  | | <-----| |
  | | -/1\- SWAPPER -----|
  | | |
  | | -/1\- MCSNAME -----<mcsname>-----|
  | | |
  | | | - = - |
  | | |
  | | -/1\- IN -----<subsystem id>-----|
  | | |
  | | | - ( <processor id list> ) - |
```

<processor id list>

```
| <-----|
| |
| | <- , - |
| |
| | GLOBAL -----|
| |
| | - 1 -----|
| |
| | - 2 -----|
| |
| | - 3 -----|
| |
| | - 4 -----|
```

Revised Semantics:

The S (Scheduled Mix Entries) message causes a display of those tasks which are scheduled. This display does not include jobs still in one of the job queues.

When SWAPPER is specified, only jobs running in swapspace will be displayed.

When MCSNAME is specified, only jobs that originated from the specified MCS will be displayed.

On a B6800 multiprocessor system, each displayed task is preceded by a subsystem indicator: processor id for a local memory task, "G" for a global memory task, or blank for a task whose subsystem location is currently unassigned.

When IN is specified, only jobs with stacks running in the subsystem identified or the processor identified will be displayed.

A typical response to an S message is as follows:

-----2 SCHEDULED ENTRIES-----

6950 JOB 70 SYSTEM/CANDE

6962/6963 55 SYSTEM/DUMPANALYZER

Swap jobs are flagged with an "#" between the priority number and the file name. The number in the scheduled entry heading is the total number of scheduled entries including suppressed entries.

New Example:

S ALL MCSNAME=SYSTEM/CANDE

This message will display all scheduled jobs (including suppressed) that originated from SYSTEM/CANDE.

SB (Substitute Backup) Message

Revised Syntax:

<-----,-----	
<-----	
-- SB --	/1\ DISK -----
-/1\ PACK ---	-/1\ PACK -----
-/1\ TAPE ---	-/1\ TAPE -----
-/1\ TAPE7 --	-/1\ TAPE7 -----
-/1\ TAPE9 --	-/1\ TAPE9 -----
-/1\ PETAPE -	-/1\ PETAPE ---
	-/1\ DLBACKUP -

Semantics:

Semantics are unchanged, except for the following:

The default SB specifications are DISK = DLBACKUP and PACK = PACK.

Example:

Unchanged

SC (System Configuration) Message

SC replaces the PC (Print Configuration) message.

Syntax:

----	SC	----
	PC	

Semantics:

When the SC (System Configuration) message is entered, the system responds by displaying the current system configuration at the supervisory console from which the message is entered.

A typical response to an SC message, which yields the total configuration of a B6700 system, is as follows:

3 PROCESSORS 1-3

D0=02100

3 MULTIPLEXORS 1-3

MPX	LIMIT	TRAFFIC
1 (MOD III)	15	2
2 (MOD III)	15	0
3 (MOD III)	13	2

MEMORY STATUS

24 IN USE 0-23

PROGRAMMED HALT/LOADS: 0

STRINGS PRESENT: 1-3

A typical response to an SC message, which yields the total configuration of a B6800 multiprocessor system, is as follows:

```

PROCESSORS
  PROC ID 3 AT PORT AC00
  PROC ID 2 AT PORT AB00
  PROC ID 1 AT PORT AA00
GLOBAL SYSTEM NUMBER = 1
D0=80100
MEMORY STATUS
GLOBAL MEMORY
  32 IN USE 32-63
LOCAL MEMORY 1
  32 IN USE 0-31
LOCAL MEMORY 2
  32 IN USE 0-31

```

A typical response to an SC message, which yields the total configuration of a B6900 system, is as follows:

SI (System Intrinsic) Message

SI incorporates the CI (Change Intrinsic) and WI (What Intrinsic) messages.

Syntax:

```

-- SI -----|
    | - + -----|
    | -<file title>-|

```

Semantics:

The SI (System Intrinsic) message loads or displays the system intrinsic as follows:

SI Displays the current intrinsic.

SI <file title>
Loads the intrinsic named in <file title>.

SI+
Loads the default intrinsic. The default is a define in the Controller such that an installation may change it if so desired.

Jobs that are currently active use the "old" intrinsic stack; new jobs use the "new" intrinsic stack.

Examples:

SI SYSTEM/X

6029 JOB 99 INITIALIZEINTRINSICSTUFF

then

SYSTEM/X (LOADED)

SI+

6039 JOB 99 INITIALIZEINTRINSICSTUFF

then

SYSTEM/INTRINSICS (LOADED)

SI

INTRINSICS: SYSTEM/INTRINSICS

SL (System Library) Message

Syntax:

```
-- SL -----|
|               |
| -----<function>-----|
|               |
| - - - - -|
| - <function> = <file title> -|
```

In order to facilitate the conversion of intrinsics to libraries (described in GENERAL note D3354), a method of mapping functional names to specific libraries has been implemented. The ODT message SL (System Library) has been added to handle this mapping.

Semantics:

The MCP maintains the function mapping tables for the library linkage mechanism. A program may reference a library through the appropriate function name. SL provides the system manager with the ability to change to new libraries without affecting any of the running programs or requiring a new library name to be compiled into the calling program. This feature has been employed to reference the support libraries (e.g., GENERALSUPPORT which is a replacement for the intrinsics).

A program can use the mapping tables with the attributes LIBACCESS and FUNCTIONNAME available currently in ALGOL and NEWP (see Mark 32 ALGOL note D3530).

There are several function names and associated libraries provided in the system software. For example, many of the intrinsics now reside in the SYSTEM/GENERALSUPPORT library which has the function name GENERALSUPPORT. All function names suffixed with "SUPPORT" are reserved for current and future system use. The user may, however, alter the library titles of the support libraries or may create function names through the SL ODT message.

All function names and their libraries are preserved across Halt/Loads by the MCP; therefore, it will not be necessary to reload this information after the Halt/Load.

The simple form of the SL message is used to interrogate the current libraries associated with the various function names.

Example:

SL

```
SL GENERALSUPPORT = SYSTEM/GENERALSUPPORT
SL PLISUPPORT      = SYSTEM/PLISUPPORT
SL USERFUNCTION   = SYSTEM/USERLIBRARY
```

The SL <function> message is used to interrogate the current library associated with a specific function name.

Example:

SL GENERALSUPPORT

```
SL GENERALSUPPORT = SYSTEM/GENERALSUPPORT
```

The SL <function> = <filetitle> is the message used to initialize the various functions or to change the specification of a function which is already in use.

If a function specification is changed to a different library, all tasks presently running will not be affected. Any new tasks will use the new library when they begin. If there should be a Halt/Load after changing a function specification, all tasks will restart using the new library.

Example:

```
SL USERFUNCTION = SYSTEM/USERLIBRARY
```

```
FUNCTION "USERFUNCTION = SYSTEM/USERLIBRARY" ESTABLISHED
```

The SL - <function> message will remove a function specification from the MCP tables.

When a function specification has been removed, all running tasks will continue running with the old library. Any new tasks will receive a FILE NOT FOUND error.

Example:

```
SL - USERFUNCTION
```

```
FUNCTION "USERFUNCTION" IS NO LONGER ESTABLISHED
```

SM (Send Message) Message

Revised Syntax:

```
--<mix number list>-- SM -----<control message>--|
                        | - : - |
```

Semantics:

Unchanged

Example:

Unchanged

SN (Serial Number) Message

The SN (Serial Number) message now allows the specification of tape density.

Revised Syntax:

```
----- SN ----- MT --<sn option list>-----|
| - SNL - | | - ( --<density>-- ) - |
```

<sn option list>

```
| <-----, -----|
| <-----|
|-----<unit number list>-----/6\-----<digit>-----|
| - : - | | -<letter>- |
| | <-----|
| - " -----/6\-----<digit>----- " - |
| | -<letter>- |
```

<density>

```
----- 200 -----|
| - 556 - - |
| - 800 - - |
| - 1600 - |
| - 6250 - |
```


The SN (Serial Number) message is used to purge (with optional locking) and assign a serial number a tape unit. The serial number may consist of up to six alphanumeric characters. If a number is used, it is right justified with leading zeros added. Any serial number containing alphabetic characters or any quoted alphanumeric string is left justified with trailing blanks. The density of the tape may be specified; however, it applies to all tapes in the list.

When the SNL form of the message is used, the tape is locked as well as purged so that no job can automatically pick up the scratch tape.

New Example:

SN MT 82 MIKE (1600)

MT82 WILL BE SN-ED

P MT

82*P [MIKE] 1600 #1 1:00 S C R A T C H

An attempt to SN a tape that has been locked causes the UNIT LOCKED message to be displayed.

SO (Set Option) Message

SO has been eliminated; see OP (Options) for new message.

SP (Show Print Queue) Message

Syntax:

Unchanged

Revised Semantics:

Jobs waiting to be punched will now be displayed in response to the SP ODT message. Formerly, only jobs waiting to be printed were displayed.

SQ (Show Queue) Message

Syntax:

Unchanged

Revised Semantics:

When the SQ (Show Queue) message is entered, the system responds by displaying information regarding the jobs in one or all of the job queues. When no number appears in the message, the system displays a table with a line for each queue, showing queue number, the job number, the time at which the job entered the job queue, and the first card image at the head of the queue.

Example:

SQ

```

QUEUE 5 (FIRST OF 3 ENTRIES):
7597 00 ?JOB X;
  QUEUED: OCT 30, 1979 AT 09:29:11
QUEUE 7 (FIRST OF 4 ENTRIES):
6627 80 ?JOB A;
  QUEUED: OCT 30, 1979 AT 09:21:32
QUEUE 4:
NO ENTRIES
QUEUE 0:
NO ENTRIES

```

When a number appears in the SQ message, the system responds by displaying the numbers and first card image of all jobs in the designated queue.

Example:

SQ7

```

QUEUE 7:
6627 80 ?JOB A;
  QUEUED: OCT 30, 1979 AT 09:21:32
6631 80 ?JOB D;
  QUEUED: OCT 30, 1979 AT 09:21:37
6629 00 ?JOB B;
  QUEUED: OCT 30, 1979 AT 09:21:39
6630 00 ?JOB C;
  QUEUED: OCT 30, 1979 AT 09:22:01

```

A fetch specification associated with a job is displayed as follows:

SQ4

```

QUEUE 4 (SINGLE ENTRY):
F133 00 ?JOB A;
  QUEUED: OCT 30, 1979 AT 13:15:22

```

If two numbers appear, all entries in the queue indicated by the first number with priority equal to the second number are displayed.

Example:

SQ7 80

```

QUEUE 7, PRIORITY 80:
6627 80 ?JOB A;
  QUEUED: OCT 30, 1979 AT 09:21:32
6631 80 ?JOB D;
  QUEUED: OCT 30, 1979 AT 09:21:37

```

SR (Secure Reader) Message

SR incorporates the RR (Release Reader) message.

Revised Syntax:

```

-- SR ----- CR --<unit number list>--|
    | - - - |

```

Revised Semantics:

The SR (Secure Reader) messages causes the system to check the usercode/password combination and reject all card decks entered into the indicated card reader which do not contain a USER system control card. If SR- is entered, the security restrictions imposed on the indicated card reader by a prior SR message will be removed.

SR CR10

CR10 SECURED

If an attempt is made to enter a deck without a USER system control card, the system responds with the typical message:

0941 CONTROL CARD ERROR

SR- CR10

CR10 RELEASED

ST (Stop) Message

Revised Syntax:

```

--<mix number list>-- ST --|

```

Semantics:

Unchanged

Example:

Unchanged

SUPPRESS (Suppress) Message

SUPPRESS incorporates the RESTORE message.

Revised Syntax:

```
--<mix number list>-- SUPPRESS -----|
                        | - - - |
```

Semantics:

The SUPPRESS message prevents indicated jobs from appearing in a display of the job mix when they are active. If SUPPRESS- is entered, the suppression is lifted. Only a mix (e.g., A,C,J) message containing the ALL qualifier may cause such jobs to appear in the mix display.

If an active suppressed task goes into a waiting state, it is displayed. When it returns to an active state, it is again suppressed.

Example:

```
180 SUPPRESS
-----
-----ACTIVE ENTRIES-----

180 JOB 70 SYSTEM/CANDE
.....193 70 STACK2/CANDE
179 JOB 80 DCP/0

results in

-----ACTIVE ENTRIES-----

180x 193 70 STACK2/CANDE
179 JOB 80 DCP/0
```

SV (Save) Message

In addition to the pre-3.1 syntax, the SV message has been modified to accommodate multiprocessor systems.

Revised Syntax:

```
      |<----- , -----|
-- SV -----<device>--<unit number list>-----|
      | - CPU --<processor number>-----|
      | - MOD --<mod number>-----|
      | - IN <processor number> - |
```

Revised Semantics:

The SV (Save) message is provided to allow a unit to be made inaccessible to the system. If the <mod number> is greater than or equal to the first mod of * GLOBAL tm Memory, the IN phrase is not used; global is used by default.

This message may be used in response to an RF DEGRADATION message to take the indicated peripheral unit off-line as soon as it is not in use by the current job which is using it. Such an SV message will inhibit further RSVP messages regarding this unit until an appropriate RY message is input. In addition, the unit is automatically placed in a "saved" status so that upon completion of this job the unit cannot be reassigned. Once a unit has been saved by an SV message, the reliability factor for that unit will remain unchanged until an RY message is entered for that unit.

The SV message may also be used to take a memory module off-line. The restrictions are that this message may not reference a module which is already saved or off-line, may not reference module #0, and may not reference the numerically highest module which was on-line at the time of the last Halt/Load. An arbitrarily long period of time may expire between the input of the SV message and the time the module becomes completely saved; all in-use areas must be returned to the system before the process is complete. A system message is issued when the module is off-line.

The SV CPU message cannot currently be used on B6800 or B6900 multiprocessor systems, due to the architecture and relationship of CPUs to I/Os in the B6800 and B6900; this restriction may be removed on a future software release.

Revised Example:

SV MT 114

MT 114 SAVED

-----MT STATUS-----
114*P [000001] SAVED

SV MOD 11 IN 2

MOD 11 IN LOCAL 2 WILL BE SAVED

A processor (CPU) may be taken off-line by the SV CPU<nnn> message once the CONDITIONAL HALT switch for the desired CPU has been turned on. In order to save one CPU, the other CPUs in a multiprocessor system must be on-line. Use the following procedure to save a CPU:

- a. Put the CONDITIONAL HALT switch in the up position. If the switch is up, OFF is displayed on the MDL panel in the A register. If the switch is not up, the system responds with

SET CONDITIONAL HALT SWITCH FOR CPU<nnn>

- b. Enter SV CPU<nnn>
- c. Put the processor in local mode and CLEAR (CL) the unit.

Note: The processor must remain in local mode; if it is in remote mode, the next Halt/Load will pick up the processor.

SW (Swapper) Message

Revised Syntax:

```
-- SW -----
|
|                                     |<----- , -----|
| - CREATE ON --<packname>--<create options>-----|
| - + -----|
| - - -| |<processor id list>-----|
|                                     |<-----|
|-----<swapper parameters>-----|
|<processor id list>--|
```

<create options>

```
-----/1*\- AREAS = <value> -----|
| -/1*\- AREASIZE = <value> -|
|<file attribute equation>-|
```

<processor id list>

```

      |-----|
      |<-----|
      |<- , -|
-- ( ----- 1 ----- ) --|
      |-----|
      | - 2 - |
      | - 3 - |
      | - 4 - |

```

<swapper parameters>

```

----- CORESIZE ----- = --<new value>-----|
|-----|
| - EXPMAXCORE -----|
| - EXPRESSRESERVE --|
| - EXPTIME -----|
| - IOBIAS -----|
| - MAXCORE -----|
| - MAXIOSIZE -----|
| - MAXSLICENUMBER --|
| - MEMORYBIAS -----|
| - MINCHUNKSIZE ----|
| - MINTIMESLICE ----|
| - PRIORITYBIAS ----|
| - RATIO -----|
| - UTILIZATIONBIAS -|
|-----|
| - SWAPTRANSTATE -----|
| - NOSWAPTRANSTATE -----|
| - FAMILY --- + ---<packname>-----|
|-----|
| - - - |

```

See the ODT Reference Manual (Form No. 5001704) for a description of <swapper parameters>; see MCP-GENERAL note D3054, "SWAPPER Enhancements" for a description of Mark 32 changes to the SWAPPER mechanism.

The CREATE option of the SW ODT message is used to create new SYSTEM/SWAPDISK files. The text of the input message is compiled by WFL (using TASK.FILECARDS format) to produce the attribute equation for the creation of the swapdisk file; thus, any combination of permissible file attributes can be easily specified.

Since the MAXRECSIZE of the swapdisk file is set to 1320 (44*30, one slot), the AREASIZE attribute specifies the number of slots per row.

Example:

To create a swapdisk file of 20 rows of 100 slots each on family KSID, enter the following:

```
SW CREATE ON KSID AREAS=20,AREASIZE=100
```

If this file were only to reside on the base pack of a multi-member family, enter the following:

```
SW CREATE ON KSID AREAS=20,AREASIZE=100,FAMILYINDEX=1
```

The value of the MAXCORE parameter is now permitted to exceed 132. When a single subspace exceeds the value of the MAXIOSIZE parameter, SWAPPER does more than one I/O operation to transfer the subspace to or from disk. When writing subspaces to disk, SWAPPER uses the MAXIOSIZE parameter and the current size of the subspace to determine the number of I/O operations to be performed, after which the subspace is fit into <n> evenly sized pieces.

Example:

If MAXIOSIZE=60, MAXCORE=300:

A 60-slot subspace would be written in a single I/O;

a 61-slot subspace would be written in two I/Os, one of 31 slots and one of 30 slots.

A new SWAPPER parameter, NOSWAPTRANSTATE, has been implemented. When NOSWAPTRANSTATE is set, any task that is in transaction state and exceeds its time slice is given another time slice. If the option SWAPTRANSTATE is set (this is the default), a task is time-sliced regardless of whether it is in transaction state.

On Mark 31 and earlier releases, the swapspace was required to be a single contiguous piece of memory. On Mark 32, the swapspace is now permitted to be any number of pieces of memory. These "chunks" may be located anywhere in memory. Even though the swapspace can be divided into several different pieces, each individual subspace must still reside in a contiguous area of memory. A segmented swapspace may not be able to handle the same number of tasks as a non-segmented swapspace with the same number of slots.

Example:

A swapspace with one chunk of 90 slots could run 3 tasks of 30 slots each simultaneously; a swapspace with a 50-slot and a 40-slot chunk can run at most two 30-slot tasks simultaneously.

To control this situation, a new parameter, MINCHUNKSIZE, has been implemented, which specifies the smallest amount of memory that a single chunk of swapspace may contain (MAXCORE <= MINCHUNKSIZE). By setting the MINCHUNKSIZE attribute to the value of the CORESIZE attribute, behavior similar to that implemented in Mark 31 and earlier releases may be obtained.

The semantics of the CORESIZE attribute remain unchanged from previous releases; however, the semantic effects of dynamically changing this attribute have been changed on the Mark 32 release. At any time the amount of swapspace actually in use can be determined from the value displayed next to ACTUALCORESIZE in response to the SW ODT message.

By setting the value of the CORESIZE parameter greater than the current value of ACTUALCORESIZE, SWAPPER attempts to expand the amount of swapspace available by getting one or more "chunk"s of memory and inserting these into SWAPPER's tables. These new chunks of memory are constrained by the value of the MINCHUNKSIZE parameter. If the difference between the new value of CORESIZE and the current value of ACTUALCORESIZE is less than MINCHUNKSIZE, no increase in swapspace will be performed.

By setting the value of the CORESIZE parameter smaller than the current value of ACTUALCORESIZE, SWAPPER attempts to shrink the amount of swapspace available by shrinking or removing individual chunks of memory. No chunks of memory may be smaller than the MINCHUNKSIZE parameter. In some situations, SWAPPER is unable to shrink the swapspace as small as desired (SWAPPER never shrinks the swapspace smaller than CORESIZE). The current algorithm for deciding which pieces of swapspace are to be shrunk or discarded is to pick the smallest chunks first.

At initialization time, if SWAPPER is unable to obtain the amount of swapspace specified by the CORESIZE parameter, an RSVP is issued notifying the operator of the situation. The operator has four alternatives:

1. Reply DS to the RSVP, aborting SWAPPER initialization.
2. Reply OK, causing SWAPPER to again attempt to find the required memory (this is done periodically).
3. Reply NOTOK, causing SWAPPER to continue initialization using whatever memory it has already obtained.
4. Enter a new value for the CORESIZE parameter using the SW ODT message. SWAPPER attempts to obtain the new amount of memory.

A sample of the new format for displaying SWAPPER parameters is the following:

```
SW ** RUNNING ** CORESIZE=90 SLOTS (89100 WORDS)
SW ACTUALCORESIZE=90 SLOTS (89100 WORDS) MINTIME=3 SECONDS
SW MAXSLICENR=7 RATIO=2 MAXCORE=50 SLOTS (49500 WORDS)
SW MAXIOSIZE=50 SLOTS (49500 WORDS)
SW MINCHUNKSIZE=50 SLOTS (49500 WORDS) EXPRESERVE=0 SLOTS (0 WORDS)
SW EXPMAXCORE=0 SLOTS (0 WORDS) EXPMAXTIME=0.75 SECONDS
SW PRIORITYBIAS=0 UTILIZATIONBIAS=6 IOBIAS=0 MEMORYBIAS=0
SW NOSWAPTRANSTATE ON SWAPPACK ON KSID
```

The initial item "*** RUNNING ***" is an indicator of the current status of SWAPPER; the values range from NOT RUNNING to INITIALIZING or SHUTTING DOWN.

"ACTUALCORESIZE" is an indicator of the amount of swapcore actually being used by SWAPPER (see "Changing the SwapSpace Size via the CORESIZE Parameter" for semantics).

"NOSWAPTRANSTATE" indicates that the NOSWAPTRANSTATE option is set (see "SWAPTRANSTATE, NOSWAPTRANSTATE PARAMETERS" for semantics).

"ON SWAPPACK ON KSID" indicates that SWAPPER is using the SYSTEM/SWAPDISK files on the packs SWAPPACK and KSID on which to swap.

TD (Time and Date) Message

TD incorporates the WD (What Date) and WT (What Time) messages.

Syntax:

-- TD --|

Semantics:

When the TD (Time and Date) message is entered, the system responds by displaying the current time and date in the following manner:

DATE IS MONDAY APR 23, 1979 (79113) hh:mm:ss

where hh, mm and ss are digits denoting hours, minutes and seconds respectively; 79113 indicates the year is 1979 and the date is day 113.

TD (Tape Directory) Message

TD (Tape Directory) has been replaced by TDIR (Tape Directory); see TDIR for new message.

TDIR (Tape Directory) Message

Syntax

```

-- TDIR -----|<----- , -----|
--              |<unit number>-----|
--              | - SPO ---| |<tape name>---|
--              | - PUNCH -|

```

Semantics:

TDIR (Tape Directory) message initiates SYSTEM/FILEDATA, which reads the directory of the specified tape. The default output is to the printer. As many intermixed tape names or unit numbers as desired may be specified, each being separated from the next by commas.

Example:

TDIR SPO X

B6700 SYSTEM 277 REPORT OF 06/09/75 AT 09:48:01.

VERSION 2.7.380

TAPE = X/FILE000. ON UNIT 115

SERIAL#=123456 CREATED 6/09/75

9-TRACK (PE) 1600 BPI

*SYSTEM/DUMPALL

TAPEDIRECTORY INPUT WAS:

"TPDIR SPO X"

.

.

.

TERM Message

A new option, FULLPAGE, has been added to the TERM message.

Revised Syntax:

```

--- TERM -----|
| |<-----| | |
| |      |
| |  LINES  <number>-----|
| |-----|
| |  - WIDTH -|
| |  - FIRST -|
| |  - TRUNCATE ----- TRUE ---|
| |  - MESSAGES -| | - FALSE ---|
| |  - RESPONSE ----- NONE -----|
| |                  - CONCISE --|
| |                  - EXPANDED -|
| |  - FULLPAGE ----- TRUE -----|
| |                  - FALSE -----|

```

Revised Semantics:

The FULLPAGE option, when TRUE, decreases the time for writing a page of information to Burroughs TD850 ODTs. The default setting is FALSE; terminals are not affected unless the option is explicitly set.

THAW Message

Syntax:

```
--<mix number list>-- THAW --|
```

Semantics:

The THAW message allows an operator to change libraries which are frozen permanently to temporary libraries. In the past, the only way to terminate a permanent library was to DS it.

TI (Times) Message

Revised Syntax:

```
--<mix number list>-- TI --|
```

Semantics:

The response to the TI ODT message now includes the task's READYQ time and PBIT time accounting.

Example:

```
5285 TI
```

```

TIMES FOR 5285
PROCESS   = 00:00:37
IO        = 00:00:01
READYQ    = 00:00:56
INITPBIT  = 00:00:06 3271 OPERATIONS
OTHERPBIT = 00:00:02 1521 OPERATIONS
ELAPSED   = 00:11:40

```

TL (Transfer Log) Message

TL replaces the LR (Log Release) Message.

Syntax:

```
-- TL --|
```

Semantics:

A new system log is generated by entering a TL (Transfer Log) message.

The previous log is named:

```
SUMLOG/<system number>/<mmddy>/<number>
```

as appropriate (where number is a log serial number from 1 to 999999) and may be saved or discarded as desired. IAD logs specified at cold start time do not have a new log file created; instead, the rows of the system log are rotated. A typical response to a TL message is as follows:

```
7275 SYSTEM/SUMLOG CHANGED TO SUMLOG/227/053075/000005 ON DISK DK033
```

TO (Test Option) Message

```
-- -----
```

TO has been eliminated; see OP (Options) for new message.

UA (Unit Available) Message

```
-- -----
```

UA has been eliminated; see UR (Unit Reserved) for new message.

UL (Unlabeled) Message

```
-- -----
```

Revised Syntax:

```
--<mix number list>-- UL -----<device>--<unit number>--|
                        | - : - |
```

Semantics:

Unchanged

Example:

Unchanged

UR (Unit Reserved) Message

```
-- -----
```

UR incorporates the UA (Unit Available) message.

Syntax:**Syntax:**

```
-- UR -----<device specifier list>--|
      | - - - |
```

<device specifier list>

```
      | <----- , ----- |
      | <device> <unit number list> ----- |
      | - <device> <unit number> --- MPX <mpx no> PATH <path no> - |
      | - PATHID <pathid no> ----- |
```

Semantics:

The UR (Unit Reserved) message is used to reserve a unit to allow FE maintenance to be performed. The unit is no longer available for assignment. When a path is specified, only the path is reserved, not the unit. For B6700 and B6800 systems, the path is specified by either MPX PATH or PATHID (blanks are required between MPX and <mpx no> and between PATH and <path no>). For B6900 systems, only PATHID is accepted; any attempt to use MPX PATH is rejected with an "INVALID SYNTAX" message. The unit may be restored to the system by entering UR -.

Examples:

```
UR PK 108 MPX 2 PATH 4
UR PK 108 PATHID 14
UR- MT 18 PATHID 12
```

Disk pack types 206 and 207 may be reserved and placed into maintenance mode by entering "UR PK <unit no> MAINT". They may be made available and removed from maintenance mode by entering "UR- PK <unit no> MAINT". If the unit is not already reserved, entering "UR- PK <unit no> MAINT" will remove the unit from maintenance mode. If an I/O error occurs because a 206 or 207 disk pack is in maintenance mode, the error message displayed at the ODT is "UNIT IN MAINTENANCE MODE".

Examples (for B6700, B6800 systems only)

```
UR LP13
-----
```

```
LP13 RESERVED
```

```
UR DK069 MPX 1 PATH 2
-----
```

```
DK069 MPX1 PATH2 RESERVED
```

```
UR- DK 097,098
-----
```

```
DK097 AVAILABLE
DK098 AVAILABLE
```

```
UR PK200 MAINT
-----
```

```
PK200 PLACED INTO MAINTENANCE MODE
PK200 RESERVED
```

```
UR- PK 206-208 MAINT
-----
```

```
PK206 TAKEN OUT OF MAINTENANCE MODE
PK206 AVAILABLE
PK207 TAKEN OUT OF MAINTENANCE MODE
PK207 AVAILABLE
PK208 TAKEN OUT OF MAINTENANCE MODE
PK208 AVAILABLE
```

If the unit is already available, the system responds as follows:

```
UR- PK096
-----
```

```
IS NOT RESERVED
```

If the unit is disk pack type 225 or 235 and is reserved, the system responds as follows:

```
UR- PK096 MAINT
-----
```

```
PK096 AVAILABLE
```

If the unit is disk pack type 225 or 235 and is not reserved, the system responds as follows:

```
UR- PK096 MAINT
-----
```

```
PK096 IS NOT RESERVED
```

If the unit is disk pack type 206 or 207 and is reserved, the system responds as follows:

```
UR- DK206 MAINT
-----
```

```
TAKEN OUT OF MAINTENANCE MODE
```

DK206 AVAILABLE

If the unit is disk pack type 206 or 207 and is not reserved, the system responds as follows:

UR- DK207 MAINT

TAKEN OUT OF MAINTENANCE MODE

W (Waiting Mix Entries) Message

Revised Syntax:

```
-- W ----->
    | -/1\- ALL - |
    |
    | <-----|
    | -/1\- SWAPPER -----|
    | -/1\- MCSNAME -----<mcsname>-----|
    |      | - = - |
    | -/1\- IN ---<subsystem id>-----|
    |      | - ( <processor id list> ) - |
```

<processor id list>

```
| <-----|
| | <- , - |
| -----|
| GLOBAL -----|
| - 1 -----|
| - 2 -----|
| - 3 -----|
| - 4 -----|
```

Revised Semantics:

The W (Waiting Mix Entries) message causes a display of those tasks which require operation action in order to continue; i.e., those suspended on an RSVP condition. The reason for suspension is included in the display.

When SWAPPER is specified, only jobs running in swapspace will be displayed.

When MCSNAME is specified, only jobs that originated from the specified MCS will be displayed.

On a B6800 multiprocessor system, each displayed task is preceded by a subsystem indicator: processor id for a local memory task, "G" for a global memory task, or blank for a task whose subsystem location is currently unassigned.

When IN is specified, only jobs with stacks running in the subsystem identified or the processor identified will be displayed.

A typical response to a W message is as follows:

-----1 WAITING ENTRIES-----

3570/3571 50 LIBRARY MAINTENANCE
1100 SECT REQ ON MCPMAST PK068*

Swap jobs are flagged with an "#" between the priority number and the file name. The number in the waiting entry heading is the total number of waiting entries including suppressed entries.

New Example:

W ALL MCSNAME=SYSTEM/CANDE

This message will display all waiting jobs (including suppressed) that originated from SYSTEM/CANDE.

New Example: (B6800 Multi-Processor System):

W SW IN(3)

This message will display all waiting jobs (excluding suppressed) whose stacks are running in swapspace and are currently swapped into processor 3.

WD (What Date) Message

WD has been eliminated; see TD (Time and Date) for new message.

WI (What Intrinsic) Message

WI has been eliminated; see SI (System Ininsics) for new message.

WM (What MCP) Message

Syntax:

Unchanged

Semantics:

Unchanged, except as follows:

A typical response to the WM message is:

```

MCP: SYSTEM/MCP 31.200.900
H/L UNIT: 64
COMPILED: 3/22/79 @ 16:27:48 (NEWP 31.197)
  COMPILER TIME OPTIONS ARE:
    DIAGNOSTICS      DISKCHECK      MTBF
    LINEINFO         LOCKTRACE
    REVERSEPAPERTAPE SWAPTRACE
H/L REASON: MANUAL
GROUPID: ONEBY
HOSTNAME: BLUE
SYSTEM SERIAL NO: 277
CATALOG LEVEL: 0
NEXT MCP: NOT SPECIFIED

```

WS (What Supervisor) Message

WS has been eliminated; see CS (Change Supervisor) for new message.

WT (What Time) Message

WT has been eliminated; see TD (Time and Date) for new message.

XS (Exceed Schedule) Message

XS has been eliminated; see FS (Force From Schedule) for new message.

Y (Status Interrogate) Message

Syntax:

Unchanged

Semantics:

The stack state line of the Y display has had location information added to it. On monolithic systems, only the display "NOT IN CORE" has been added to indicate a swapped-out swapjob. On B6800 multiprocessor systems, an indication of the processor in which a job is running has been added to the display; however, when a task is scheduled or swapped out, an appropriate display is also shown.

The display now includes the usercode if the task is running with a usercode.

PRIMITIVE Messages

The following "primitive" messages have been added or modified.

```
??CM
-- ??CM -----<filename>--|
      | - # - |
```

This feature allows changing to a new MCP without changing the disk bootstrap. The change is effective immediately (even if the mix is non-empty). If a Halt/Load occurs, the previous MCP is loaded, since it is pointed to by the disk bootstrap.

If the "#" character is omitted, the disk bootstrap is changed and the CM is permanent (until the next CM).

```
??DUMP
-- ?? ----- DUMP -----|
      | -<mix number>-- | - DP ---|
```

The ??DUMP message will invoke a non-fatal memory dump (through the MCP procedure KEYIN).

```
??FS
-- ?? ----- FS -----|
      | -<mix number>-- | - XS - |
```

The ??FS message causes the execution of the indicated scheduled job. If ??FS is used without the mix number, all scheduled jobs are executed.

```
??HS
-- ?? --- HS ---|
      | - EI - |
```

The ??HS message is used to suspend job selection when the CONTROLLER will not accept the HS message. An example of the use of the ??HS message is to inhibit job selection immediately following a Halt/Load.

D3396 GENERAL - "LISTNOTES" CHANGES

LISTNOTES is a program to process and print system notes (D- and P-notes) and other <file>/DOCUMENT files on the SYSTEMNOTES tape accompanying this release.

LISTNOTES was released on previous system release notes tapes. The Mark 32 version has been substantially changed, particularly the interpretation of the task values and label equation.

Some of the changes include the following:

1. Default line printer file is EBCDIC96
2. Default reports are 'FINAL'; i.e., no sequence numbers
3. Option to print PCN pages only (for update releases)
4. Option to convert notes to 'NEWS' (type seqdata) files and removing headings, footings, etc.

Operating Instructions

Remote Use (via CANDE)

E LISTNOTES ; FILE IN = X
 E LISTNOTES; VALUE CN ; FILE IN = X

Batch Use

RUN OBJECT/LISTNOTES ; FILE IN = X
 RUN OBJECT/LISTNOTES ; VALUE CN ; FILE IN = X

The allowable values of 'N' in VALUE = CN are shown in the following table:

N	FINALTOG	UPPER % LOWERCASE	UPPERCASE ONLY	MAKETOG	PRINTER OUTPUT
0	T	T	F	F	T
1	T	F	T	F	T
2	F	T	F	F	T
3	F	F	T	F	T
9	T	F	F	T	F

C = 0 (DEFAULT) PROCESS ALL RECORDS IN THE FILE
 C = 1 PROCESS ONLY PAGES WITH PCN '|' MARKS

N = 0 FINALTOG TRUE, PRINT IN UPPER AND LOWER CASE (DEFAULT VALUE)
 N = 1 FINALTOG TRUE, PRINT ALL IN UPPER CASE
 N = 2 FINALTOG FALSE, PRINT IN UPPER AND LOWER CASE
 N = 3 FINALTOG FALSE, PRINT IN UPPER CASE
 N = 9 FINALTOG TRUE AND CREATE 'DP/NEWS/X' FILES

FINALTOG TRUE (N = 0 OR N = 1)

The notes are printed in the center of the page without line numbers and sequence numbers

FINALTOG FALSE (N = 2 OR N = 3)

The notes are printed in the center of the page with line numbers on the left and two pairs of sequence numbers on the right: the first sequence number on the right is from the sequence field of the input file to the Mission Viejo text editor; the second sequence number is in steps of 100 and may be used as the sequence number to locate records in the input file to LISTNOTES, which is of type 'CDATA'.

UPPER & LOWER CASE (DEFAULT)

The records are printed on the file LINE with TRAINID=EBCDIC96. The records are not translated to uppercase.

UPPER CASE ONLY (N = 1 OR N = 3)

All records are translated to uppercase and printed on the file line with TRAINID = EBCDIC72.

PCNSTOG (C = 1, N = 0,1,2 OR 3)

Only pages which have records with the PCN (publication change notice) flag '|' are printed. This option may be used to print field update notes.

MAKETOG (N = 9)

This option processes the file 'IN = X' and creates the following files:

1. DP/NEWS/X
2. DP/NEWS/INDEX/X
3. DP/NEWS/CONTENTS/X

Each file is type 'SEQDATA'

The records in the file 'X' are converted to a file of type 'SEQ'. All headings, footings, and section headings are removed. Two or more consecutive blank lines are also removed.

The 'INDEX' file contains a record for each software item and contains the sequence numbers of the first and last record in the 'NEWS' file for the corresponding software item. It also contains one record for each D-note and P-note together with the sequence number of the first record in the 'NEWS' file for the corresponding note.

The 'NEWS' file may be used to extract portions of the system notes for publishing (in machine readable form) for system users. Extracting the CANDE D-notes could create a 'CANDE/NEWS' file, etc.

The 'CONTENTS' file contains the table of contents for the system notes.

LABEL EQUATION, FILE IN = X

The input file is specified with the label equation:

FILE IN = <file title>

Versions of LISTNOTES released prior to June 1980 required the label equation 'FILE DISKIN = X'. An attempt to use this label equation results in a warning to re-run with the correct label equation, as follows:

```
"THE LABEL EQUATION: 'FILE DISKIN = X'
WAS USED WITH THE OLD VERSION OF THIS PROGRAM
PLEASE USE: 'FILE IN = X'
PLEASE NOTE THE TASKVALUES HAVE ALSO BEEN CHANGED"
```

The options specified by the task value have also been changed from the previous releases.

D3572 GENERAL - "PBIT" TIME ACCOUNTING

As part of an on-going program to reduce the number of compile-time options in the Large Systems software, the mechanism for calculating presence bit time in a task will be changed on the Mark 34 release. At that time, the PRESENCEBITCHARGED compile-time option in the MCP will be removed and the presence bit time for a task will be obtained from the log. Between Mark 32 and Mark 34, both the "raw" process time and the presence bit time will be available in the log to allow timely modification to installation accounting programs. During this interval, use of the PRESENCEBITCHARGED feature will yield the same results as on previous releases.

For more detail concerning the new implementation, refer to MCP note D3573.

D3650 GENERAL - IMPLEMENTATION OF PORT FILES

Communication between processes can now be performed through the standard input/output facility using files of KIND=PORT. Additional file attributes, parameters to the open and close functions, and language constructs in ALGOL, FORTRAN, PL/I, and COBOL have been defined to provide access to this facility. This note describes the general port file facility, while the specific interface defined for each language is described in a separate note for each language.

Use of Attributes

A port file has one or more associated subports (called "subfiles"), each of which may be connected to a different process. Several file attributes have been added or extended to return information about the port file or its subfiles. The following table describes the file attributes applicable to files of KIND=PORT. Attributes marked "File" apply to the port file as a whole; attributes marked "Subfile" apply to each individual subfile of the file.

<u>File Attribute</u>	<u>Notes</u>
BLOCKSTRUCTURE	File
CENSUS	File, Subfile
CHANGEDSUBFILE	File
CHANGEEVENT	File, Subfile
CURRENTRECORD	Subfile
FILESTATE	Subfile
FRAMESIZE	File
INPUTEVENT	File, Subfile
INTNAME	File
LASTSUBFILE	File
MAXCENSUS	Subfile
MAXRECSIZE	File, Subfile
MAXSUBFILES	File
MYNAME	File
OUTPUTEVENT	Subfile
SECURITYTYPE	File
STATE	File
SUBFILEERROR	Subfile
TITLE	File
YOURNAME	Subfile
YOURUSERCODE	Subfile

Attributes that are only file attributes may be accessed or assigned by not specifying a subfile index. If a subfile index is specified for a file attribute access or assignment, an attribute error is generated.

Those attributes which are only subfile attributes can be assigned for a particular subfile by providing a subfile index. If a subfile index of zero is specified, the attribute assignment applies to all subfiles in the file. If MAXSUBFILES is equal to one, the subfile index may be omitted; the attribute assignment will apply to the only subfile. When accessing a subfile attribute of a particular subfile, the subfile index must be specified if MAXSUBFILES is greater than one. If MAXSUBFILES is equal to one, the subfile index may be omitted; the attribute access will apply to the only subfile. If a subfile index of zero is specified for an attribute access, an attribute error is generated.

For attributes that are both file and subfile attributes, if a subfile index is not specified, the attribute access or assignment applies to the file; otherwise the attribute access or assignment applies to the subfile.

The null value for all string-valued attributes is ".".

The following attribute descriptions apply to port files:

BLOCKSTRUCTURE

The BLOCKSTRUCTURE values of FIXED and EXTERNAL apply to port files, the default being FIXED. BLOCKSTRUCTURE is meaningful only for READ operations. If BLOCKSTRUCTURE is equal to FIXED, the user's buffer is blank filled. If BLOCKSTRUCTURE is equal to EXTERNAL, only the data received is put into the user's buffer; the actual length of the data placed into the user's buffer can be determined by interrogating the CURRENTRECORD attribute.

CENSUS

The CENSUS attribute can be accessed but not assigned. If accessed as a file attribute, CENSUS returns the total number of messages queued for all subfiles. If accessed as a subfile attribute, CENSUS returns the number of messages queued for the specified subfile.

CHANGEDSUBFILE

CHANGEDSUBFILE is an access-only attribute that returns the subfile index of an arbitrary subfile whose CHANGEEVENT is "happened".

CHANGEEVENT

The subfile CHANGEEVENT is caused whenever the value of FILESTATE changes; it is reset as a side effect of interrogating the FILESTATE attribute. The CHANGEEVENT for the file has the value "happened" as long as any of the subfile CHANGEEVENTs have the value "happened". The CHANGEEVENT for the file is reset by the system after all of the subfile CHANGEEVENTs have been reset.

CURRENTRECORD

This attribute returns the length, in FRAMESIZE units, of the last record read or written.

FILESTATE

The FILESTATE attribute can assume the following values:

CLOSED (0)

The initial state of a subfile is CLOSED. The subfile returns to this state when it is closed by the user.

OFFERED (2)

A subfile enters this state when an open has been done but no matching subfile has been found. I/O operations are not valid when the file is in this state.

OPENED (3)

This state indicates that the subfile is open and may be used to send or receive data.

CLOSEPENDING (6)

This state indicates that the user has closed the subfile, but the other subfile has not yet acknowledged the closure. When close acknowledgment is received, FILESTATE changes to CLOSED.

DEACTIVATIONPENDING (7)

This state indicates that the other subfile has been closed and that this subfile has data queued for input.

DEACTIVATED (8)

This state indicates that the other subfile has been closed and that this subfile does not have data queued for input. Close is the only valid operation for a subfile in this state.

FRAMESIZE

This attribute has the same semantics that it has for other types of files. Data is always transmitted in 8-bit units, but the user program may deal with the data using other values for FRAMESIZE.

INPUTEVENT

If accessed as a file attribute, INPUTEVENT returns "happened" if the CENSUS file attribute is greater than zero. If accessed as a subfile attribute, INPUTEVENT returns "happened" if the CENSUS subfile attribute is greater than zero for the specified subfile.

INTNAME

This attribute has the same semantics that it has for other types of files.

LASTSUBFILE

This attribute contains the subfile index of the last subfile that was used for an I/O operation on the file. This value is updated only if the I/O operation was successful. LASTSUBFILE is the preferred synonym for LASTSTATION.

MAXCENSUS

MAXCENSUS specifies the number of input messages that can be queued for this subfile before the other subfile is given a "NO BUFFER AVAILABLE" indication.

MAXRECSIZE

The MAXRECSIZE attribute can be accessed or assigned as a file attribute and is access-only as a subfile attribute. As a file attribute, MAXRECSIZE is used to access or assign the maximum message text size for the port file. When interrogated as a subfile attribute, MAXRECSIZE returns the actual message text size for the subfile, which is negotiated when the subfile is being opened and may be different for each subfile.

MAXSUBFILES

This attribute specifies the maximum number of subfiles that can be opened for the file. The subfiles are assigned indices from 1 to MAXSUBFILES, inclusive.

MYNAME

MYNAME is a string-valued attribute that is used during the subfile matching process; the value of MYNAME must match the value of YOURNAME for the complementary subfile.

OUTPUTEVENT

OUTPUTEVENT is caused whenever output buffers become available and is reset by the system whenever no output buffers are available.

SECURITYTYPE

This attribute has the same semantics that it has for other types of files. The only values allowed for SECURITYTYPE are PUBLIC and PRIVATE.

STATE

The STATE attribute returns result information about the last I/O that was done on the file. The following STATE bits apply to port files:

STATE.[0:1]

This bit indicates that an error has occurred and is set in conjunction with other STATE bits.

STATE.[3:1]

This bit indicates that an invalid subfile index was specified for an I/O operation.

STATE.[8:1]

This bit indicates that an I/O operation failed for one of the following reasons:

- a) A broadcast write failed for at least one subfile.
- b) A write with the DONTWAIT option was not done because no buffer was available.
- c) A read with the DONTWAIT option was not done because no data was available.

STATE.[9:1]

This bit indicates end-of-file.

SUBFILEERROR

The SUBFILEERROR attribute is set to one of the following values after each I/O, OPEN, or CLOSE operation that affects the subfile:

NOERROR(0)

No error occurred during the subfile operation.

NOBUFFER(3)

An attempted write to this subfile failed because no buffer space was available. This error can occur only if DONTWAIT was specified on the write.

NOFILEFOUND(4)

An attempted open on this subfile resulted in a NOFILEFOUND result.

TITLE

TITLE must be a simple name between 1 and 17 characters in length, inclusive. Its value must match the TITLE of the complementary subfile.

YOURNAME

YOURNAME is a string-valued attribute that is used during the subfile matching process; the value of YOURNAME must match the value of MYNAME for the complementary subfile.

YOURUSERCODE

The YOURUSERCODE attribute specifies the usercode under which the process opening the other (complementary) subfile must be running in order to match this subfile if SECURITYTYPE is PRIVATE. The default value for the YOURUSERCODE attribute is the usercode of the process opening this subfile. Setting the value of YOURUSERCODE to null sets the value back to the default.

Open Operations

A subfile provides a two-way, point-to-point, logical communication path between two programs. In order to establish this path, each program must describe the desired connection. The system compares connection descriptions, matches complementary descriptions, and marks the subfiles OPENED. This process is called the matching algorithm. The following subfile attributes are used by the matching algorithm:

MYNAME

The MYNAME attribute is a string of up to 100 characters that is used for matching complementary subfile descriptions. In order to match, the value of MYNAME must match the value of YOURNAME for the complementary subfile. A null value for MYNAME matches only a null value for YOURNAME.

YOURNAME

The YOURNAME attribute is a string of up to 100 characters that is used for matching complementary subfile descriptions. In order to match, the value of YOURNAME must match the value of MYNAME for the complementary subfile. A null value of YOURNAME matches any value for MYNAME.

TITLE

The TITLE attribute is in the form of a simple name (1-17 characters) and must not be null. The TITLE must match the TITLE of the complementary file. The default TITLE is the value of the INTNAME attribute.

SECURITYTYPE

Security checking is performed for each subfile as follows: If the value of its SECURITYTYPE attribute is PUBLIC, security checking is immediately successful. If the value is PRIVATE, the value of its YOURUSERCODE attribute must match the usercode of the process offering the complementary subfile.

The OPEN statement requires two parameters: the subfile that is to be opened and an open option. The processing of the OPEN statement causes the matching algorithm (described above) to be invoked and causes a result to be returned indicating the success or failure of the open attempt. Acceptable values for the two parameters and the possible values for the OPEN statement result are described in the following paragraphs.

The subfile to be opened is specified by a subfile index. If the subfile index is zero, all subfiles with a FILESTATE of CLOSED are opened. When this "open all" facility is used, open results can be obtained by interrogating the SUBFILEERROR attribute. If the subfile index is greater than zero but not greater than MAXSUBFILES, only the specified subfile is opened. If no subfile index is specified and MAXSUBFILES is greater than one, an error of BADSUBFILEINDEX is returned (the open results are described below); if MAXSUBFILES is equal to one, the (only) subfile will be opened.

The following three open options apply to port files:

WAIT

WAIT is the default value. The subfile is offered for matching, and the program is suspended until a matching subfile is found.

OFFER

OFFER causes the subfile to be offered for matching, and the program is resumed without waiting for the subfile to be matched.

AVAILABLE

AVAILABLE causes the subfile to be matched only to a complementary subfile that has been already offered. If a match is found, the subfile is opened. If no match is found, a NOFILEFOUND result is returned (the subfile is NOT left offered for subsequent matching).

The OPEN function may return the following values as its result:

OK(1)

The open was successful. If the open type was OFFER, this result indicates that the open process was successfully started.

NOFILEFOUND(2)

NOFILEFOUND is returned if AVAILABLE was specified for the open option and a matching subfile was not found.

ALREADYOPEN(40)

ALREADYOPEN is returned if the specified subfile does not have FILESTATE equal to CLOSED.

BADSUBFILEINDEX(42)

BADSUBFILEINDEX is returned if the subfile index specified was less than zero or greater than MAXSUBFILES or if a subfile index of zero was specified with the AVAILABLE open type.

As part of the open process, the value of the MAXRECSIZE attribute to be used in the conversation between the two subfiles is negotiated. The negotiated value is always the smaller of the two MAXRECSIZE values.

For languages that allow implicit file open, port files may be implicitly opened by an I/O operation on the file. If a non-zero subfile index is specified, then only that subfile will be opened. If a subfile index of zero is specified, then no implicit open action will take place. If a subfile index is not specified and MAXSUBFILES is equal to one, then the subfile will be implicitly opened.

I/O Operations

Each subfile has a unique subfile index. A program can, by specifying a subfile index, perform a read or write to a particular subfile. If the subfile index is omitted and MAXSUBFILES is equal to one, the I/O is performed on the only subfile of the file. If a subfile index of zero is specified on a read, a non-selective read is performed. The non-selective read provides the ability to read the next message from any subfile. If a subfile index of zero is specified for a write, the message is broadcast to all open subfiles. If an error occurs on any subfile during a broadcast write, the result from the write will indicate that an error has occurred.

I/O statements may include a DONTWAIT specification, which allows the program to continue executing if the I/O operation cannot be completed immediately. If an I/O statement is prematurely terminated because of a DONTWAIT specification, a field in the STATE attribute is set to indicate this occurrence (see Use of Attributes). If a READ statement is executed and CENSUS is equal to zero, the program is suspended until data is available unless DONTWAIT is specified. If a WRITE statement is executed and no buffers are available, the program is suspended until the write can be completed unless DONTWAIT is specified.

If a subfile index less than zero or greater than MAXSUBFILES is specified or if no subfile index is specified and MAXSUBFILES is greater than one, the program attempting the I/O operation is terminated. Some languages have the ability to return a result from an I/O operation. These languages may return a "bad subfile index" indication instead of terminating the user program.

When the value of the FILESTATE attribute becomes DEACTIVATIONPENDING, all subsequent write operations return an end-of-file indication. Read operations will continue to operate normally as long as there are messages queued for input; when there are no more messages queued for input, the FILESTATE changes from DEACTIVATIONPENDING to DEACTIVATED, and all subsequent read operations return an end-of-file indication. If a program is suspended waiting for an I/O operation to complete and the other subfile closes (FILESTATE goes to DEACTIVATED), the program is resumed and end-of-file is returned.

I/O statement options, such as skip-to-channel, skip lines, and stacker selection, are ignored for port files.

When the value of the FILESTATE attribute is OFFERED, all I/O operations will return an end-of-file condition.

The length of data transfers through a port file is dependent on the following criteria:

- a) The length indicated in the I/O statement.
- b) The MAXRECSIZE of the subfile.
- c) For READ operations, the length of the data actually received (this is a factor only if BLOCKSTRUCTURE is equal to EXTERNAL).
- d) The size of the user's buffer.

For WRITE operations, the amount of data sent is the minimum of the subfile MAXRECSIZE, the length indicated in the WRITE statement, and the size of the user's buffer. If an attempt is made to write data that is larger than MAXRECSIZE, the message is truncated to MAXRECSIZE; no indication of this truncation is given to the user. If the length specified in the WRITE is smaller than MAXRECSIZE but larger than the user's buffer, the message is truncated to the size of the user's buffer; no indication of this truncation is given to the user. The value of the BLOCKSTRUCTURE attribute has no effect on WRITE operations.

For READ operations with BLOCKSTRUCTURE equal to FIXED, the length of data delivered to the user's buffer is the minimum of the length indicated in the READ statement, the value of MAXRECSIZE, and the size of the user's buffer. If the message received is smaller than this size, the buffer is blank-filled. If the message received is larger than this size, the message is truncated without any indication given to the user. The value of CURRENTRECORD is always equal to MAXRECSIZE in this case.

For READ operations with BLOCKSTRUCTURE equal to EXTERNAL, the length of the data delivered to the user's buffer is the minimum of the length indicated in the READ statement, the value of MAXRECSIZE, the size of the user's buffer, and the size of the actual message. If the size of the actual message is larger than the length indicated in the READ statement, the value of MAXRECSIZE, or the size of the user's buffer, truncation will take place and the user will not be notified. No blank fill is done for BLOCKSTRUCTURE equal to EXTERNAL. The CURRENTRECORD attribute may be used to determine the length of actual message.

Close Operations

The execution of a CLOSE statement changes the FILESTATE of the specified subfile to CLOSEPENDING or CLOSED. Any messages that have been queued for receipt are discarded.

Closing a port file may take a significant amount of time. Because this delay is unacceptable for time-critical programs, the close option DONTWAIT is provided. If DONTWAIT is not specified, the program will be suspended while the actual close takes place. If DONTWAIT is specified, control returns immediately to the program and the process of actually closing the file takes place in parallel with the execution of the program. The program may detect when the close is complete by monitoring state changes. All other close options (PURGE, LOCK, etc.) are ignored. The CLOSE function may return the following values as its result:

OK(1)
The close was successful.

FILENOTOPEN(30)
The subfile was already closed.

BADSUBFILEINDEX(42)
The subfile index specified was less than zero or greater than MAXSUBFILES.

If the subfile index specified in a CLOSE statement is zero, all open subfiles are closed. When this "close all" facility is used, close results may be obtained by interrogating the SUBFILEERROR attribute. If the subfile index is greater than zero but not greater than MAXSUBFILES, only the specified subfile is closed. If no subfile index is specified and MAXSUBFILES is greater than one, an error of BADSUBFILEINDEX is returned; if MAXSUBFILES is equal to one, the (only) subfile will be closed.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

GENERAL

P3273 GENERAL - COPYRIGHT, VERSION LEVELS UPDATED

The copyright notices have been updated for 1980.

Version levels for all software products have been set to the current level.

DOCUMENT CHANGES NOTES (D NOTES)

ALGOL

D2837 ALGOL - COMPILE TIME EXTENSION TO "<DEFINE DECL>"

The following changes should be made to the ALGOL Language Reference Manual (Form No. 5001639):

On Page H-1, change the third line from "and (4)" to "(4) an extension to the <define declaration>; and (5)".

On Page H-4, before heading "Compiler Options", add the following:

"EXTENSION TO <define declaration>
-----"

The following extension to <define declaration> is available when using the compile-time facility:

```
<definition> ::= <defined identifier> <formal symbol part> = <text> # |
               <defined identifier> <formal symbol part> := <text> #
```

If a define identifier is declared with ":", then any compile-time variables, identifiers, and statements in the <text> are evaluated once, at the time of and in the scope of the define declaration. If a define is declared with "=", then the compile-time items in the <text> are evaluated upon each invocation of the define identifier and in the scope of the invocation."

D3000 ALGOL - "\$BEGINSEGMENT" NOT FOR USE AROUND BLOCKS

The \$BEGINSEGMENT-\$ENDSEGMENT option is intended for use around groups of procedures. The ALGOL Language Reference Manual (Form No. 5001639), however, indicates that it is also possible to use this option around blocks; e.g.,

```
$BEGINSEGMENT
BEGIN
  REAL X;
  .
  .
  .
END;
$ENDSEGMENT
```

This is not true. Consequently, the following changes should be made to the ALGOL Language Reference Manual: The first paragraph on Page D-7 under "BEGINSEGMENT (RESET)" should read as follows:

"The BEGINSEGMENT and ENDSEGMENT options allow user control of procedure segmentation. Procedures encountered between the BEGINSEGMENT and ENDSEGMENT options are placed in the same segment. The BEGINSEGMENT option must appear before the declaration of the first procedure to be included in the user segment. The ENDSEGMENT option must appear after the last source image of the last procedure in the user segment. The first procedure in the user segment must be one that the compiler normally segments; i.e., one with declarations. Only procedures completely contained within a procedure in the segment can be included in a user segment. External procedures cannot be declared in a user segment."

The sixth paragraph on Page D-7 under "BEGINSEGMENT (RESET)" should read as follows:

"A procedure cannot be split across user segments."

The last paragraph on Page D-7 under "BEGINSEGMENT (RESET)" should read as follows:

"Another purpose of BEGINSEGMENT and ENDSEGMENT is to allow the programmer to segregate infrequently called procedures from frequently called procedures; i.e., to group frequently called procedures into one segment to reduce "presence bit" overhead."

D3001 ALGOL - GLOBAL "WFL" FILES IN "\$INCLUDE" OPTION

Global WFL files cannot be used in conjunction with the \$INCLUDE option.

Add the following to the ALGOL Language Reference Manual (Form No. 5001639), as the third sentence on Page D-15, under "INCLUDE (cannot be SET or RESET)":

"Files declared globally in WFL jobs do not work in conjunction with the INCLUDE option and should not be used as INCLUDE files."

D3009 ALGOL - "STRING" EXPRESSIONS IN "WRITE" AND "REPLACE"

The following changes have been made to the ALGOL STRING implementation, which was described in 3.0 ALGOL note D2427:

THE REPLACE STATEMENT

The REPLACE statement has been expanded to allow <string expression>s in the <source part>. Other than <string expression> and <source part>, the metavariables are defined in the ALGOL Language Reference Manual (Form No. 5001639), Page 5-78.

New Syntax:

<replace statement>

```

-- REPLACE --<destination>-- BY ---<source part>---|
                                     |<----- , -----|

```

<source part>

```

-----<source>--<transfer part>-----|
|
| -<arithmetic expression>-----|
|                               | -<unit count>--|
|
| -<digit convert part>-----|
|
| -<numeric convert part>-----|
|
| -<translate part>-----|
|
| -<pointer-valued attribute>-----|
|
| -<string expression>-----|

```

Examples:

```

REPLACE P BY S1 || S2;
REPLACE P BY PTR:PTR FOR 10,
    TAKE(S,2) || SA[4],
    PTR FOR N,
    HEAD(S,ALPHA);

```

INPUT/OUTPUT

A <string expression> is now allowed as a <list element> for formatted and unformatted output.

Examples:

```

WRITE(OUT, 10, S1 || S2);
WRITE(OUT, <2A10>, TAKE(S1,2), S1 || "ABC");

```

D3062 ALGOL - LONGER STRINGS WITH IMPLICIT CONCATENATION

The maximum length of string literals formed by implicit concatenation has been increased from 256 characters to 4095 characters. The maximum number of characters which may be placed within a single set of quotes is still 256, however.

The following example is now legal:

```

ARRAY A[0:683];
REPLACE POINTER (A[0]) BY "<256 chars>" "<256 chars>". . .
. . . <up to a total of 4095 characters> ;

```

The new length limit is valid for assignment to STRING variables, for use within REPLACE statements and FILL statements, and for declaration of VALUE arrays. The new limit is NOT valid within FORMAT declarations.

D3063 ALGOL - EVENTS, EVENT ARRAYS AS LIBRARY PARAMETERS

Events and event arrays are now allowed as parameters to library procedures.

D3211 ALGOL - GIVE WARNING FOR "INTMODE=BCL"

The warning "BCL PROGRAMS ARE NOT PORTABLE TO EBCDIC MACHINES" is now given for any of the following constructs:

```
FILE F(INTMODE=BCL);
F(INTMODE=BCL);
F.INTMODE:=VALUE(BCL);
```

D3266 ALGOL - ALLOW "LONG" AS KEY WORD

Prior to the Mark 32 system release, all VALUE ARRAYS have been treated as LONG. The segmenting of VALUE ARRAYS will be accomplished by the following process:

1. Release 31 PR1

LONG will be allowed as a key word in VALUE ARRAY declarations. All VALUE ARRAYS will still be long.

2. Release 32

A new compiler control option will be implemented: \$SEGMENTEDVALUEARRAYS, which, when set, will cause VALUE ARRAYS more than 1023 words in length to be segmented automatically. VALUE ARRAYS declared LONG will be long. Programs without this option set will still get the long VALUE ARRAYS without requesting them.

A new ALGOL language intrinsic will be available to perform the MASKSEARCH function on all arrays, whether they are segmented or not. This function is called ARRAYSEARCH; its parameters and result are identical to MASKSEARCH.

3. Release 33

VALUE ARRAYS will be segmented by default if they are more than 1023 words in length (unless LONG is specified in the declaration). Use of the SEGMENTEDVALUEARRAYS dollar option will have no effect.

D3269 ALGOL - BINARY "I/O" FOR STRINGS

Binary I/O for Strings in ALGOL

Binary I/O for strings in ALGOL previously did not work because the length of the string was neither read or written. A binary write simply wrote the string text. A binary read attempted to read a string for the former length of the string variable. Binary I/O for strings in ALGOL now works similarly to binary I/O for other data types. Whenever a string is written using a binary write, it can later be read using a binary read.

Example:

```
WRITE(F,*, STR, STRARRAY[5], STR || "ABC");
```

Can later be read by:

```
READ(F,*, STR1, STR2, STRARRAY[0]);
```

When a string is written using a binary write, the length of the string is written as the next word in the file, followed by the string text.

When a string is read using a binary read, the next word in the file is read as the length of the string, then that many characters are read into the string variable.

Strings are stored in the same manner as LEN, PTR FOR LEN, where LEN is an integer and PTR is a pointer.

When a string is truncated by a binary write to a fixed length record, a corresponding binary read of the string will return the correct truncated length of the string.

Free-field Output for Strings

Free-field output for string expressions has been improved for the "*/" designator. Previously, when the name of a string expression was output, the concatenation operator and the string function names would be output as an unknown construct ("").

Example:

```
WRITE(F,*/,TAKE("ABC",2) || DROP("ABC",2));
```

Would output:

```
<UNK>("ABC",2)<UNK><UNK>("ABC",2)=ABC,
```

The concatenation operator and the string function names will now be written correctly.

The above example will now write:

```
TAKE("ABC",2)||DROP("ABC",2))=ABC,
```

D3349 ALGOL - SET "\$NOBINDINFO"

The BINDER cannot properly bind codefiles which contain the timing code necessary for statistics; thus, NOBINDINFO is set automatically if STATISTICS is set. If NOBINDINFO is subsequently reset, an error is given and the following message is emitted:

"SETTING STATISTICS PREVENTS PROGRAM FROM BEING BOUND".

D3350 ALGOL - FLAG "\$NOBINDINFO"

The information saved for the BINDER when NOBINDINFO is reset is also necessary to make the program a host to which separate compilations can be applied. A syntax error is now given if NOBINDINFO and MAKEHOST are both set.

D3360 ALGOL - MODIFICATIONS TO SUPPORT PORTFILES

ALGOL supports files of KIND=PORT and all of the applicable file attributes. (See Mark 32 GENERAL note D3650, "Implementation of Port Files", for a description of port files.)

ALGOL provides access to port subfiles through the new SUBFILE syntax of the READ and WRITE statements. The following diagram describes the <subfile specification>, which appears as an option in the syntax for <record number or carriage control>:

<subfile specification>

```

---- DONTWAIT -----
|
| - SUBFILE -----<subfile index>-----
|               |
|               | - <result> : -
|               |               | - , DONTWAIT -

```

<result>

--<arithmetic variable>--|

<subfile index>

--<arithmetic expression>--|

For READ statements, if the <subfile index> is zero, a non-selective read is performed; if it is non-zero, a read from the specified subfile is performed. In either case, the <result> is set to the index of the subfile from which the data was received.

Example:

```
READ(INFILE[SUBFILE J:I, DONTWAIT],14,A[*]);
```

For WRITE statements, if the <subfile index> is zero, a broadcast write is performed; if it is non-zero, a write to the specified subfile is performed. In the first case, <result> is set to zero; in the second case, it is set to the value of the <subfile index>.

Example:

```
WRITE(OUTFILE,[SUBFILE I],14,A[*]);
```

If DONTWAIT is specified for a READ statement and no message is available, no data is returned (i.e. no variables in the list are altered) and the task is not suspended.

If DONTWAIT is specified for a WRITE statement and no buffer is available, the task is not suspended. The <subfile specification> may be used only on READ and WRITE statements using array-row I/O; the <subfile specification> may not be used with formatted, free field or binary I/O.

In order to provide access to the port open options, the OPEN statement has been added to ALGOL. If OPEN is used as a function, it returns an integer value corresponding to the AVAILABLE attribute.

If the MAXSUBFILES attribute of a port file is greater than 0, then subfiles must be explicitly opened before reading from or writing to them.

<open statement>

```
-- OPEN -- ( --<file designator>----->
                        | - [ SUBFILE <subfile index> ] - |
>----- ) -----|
| - , --<open type>- |
```

<open type>

```
----- WAIT -----|
| - OFFER ----- |
| - AVAILABLE -  |
```

If no <open type> is specified, WAIT is assumed.

Example:

```
OPEN(INFILE[SUBFILE I],WAIT);
```

The CLOSE statement has been modified to allow a subfile to be selected. If CLOSE is used as a function, it returns an integer value which indicates the success or the type of failure of the CLOSE.

<close statement>

```
-- CLOSE -- ( --<file designator>----->
                        | - [SUBFILE <subfile index>] - |
>----- ) -----|
| - , <close option> - |
```

<close option>

```
----- * -----|
| - CRUNCH --- |
| - LOCK  ---  |
| - PURGE ---  |
| - REEL  ---  |
| - REWIND ---  |
| - DONTWAIT - |
```

If the subfile index specified in a CLOSE statement is zero, all open subfiles will be closed. If the subfile index is greater than zero but not greater than MAXSUBFILES, only the specified subfile is closed. If no subfile index is specified and MAXSUBFILES is greater than one, an error of BADSUBFILEINDEX is returned; if MAXSUBFILES is equal to one, the (only) subfile will be closed.

The only meaningful close option for port files is DONTWAIT.

Example:

```
CLOSE(INFILE[SUBFILE I]);
```

D3363 ALGOL - REMOVE "SIGNAL" AND "RESPONSE"

The SIGNAL and RESPONSE clauses for <record number or carriage control> for the READ and WRITE statements are no longer recognized.

D3433 ALGOL - ARRAY ROW EQUIVALENCE TO OWN ARRAYS

The syntax for array declarations does not prohibit an array declaration of the form

```
OWN ARRAY A[0] = B
```

However, the ALGOL compiler will not accept this declaration, because an array row declared by equivalence is OWN if and only if the row it is declared equivalent to is OWN. In the following example, therefore, the array A is OWN, and the array A1 is not OWN.

```
OWN ARRAY B[0:10];
ARRAY B1[0:10];
EBCDIC ARRAY A[0]=B;
HEX ARRAY A1[0]=B1;
```

D3434 ALGOL - "SDIGITS" IN "REPLACE" STATEMENT

The class 2 reserved word "SDIGITS" (read "sign-digits") may be used in a REPLACE statement in the same way as "DIGITS" is used, except that the sign of the <arithmetic expression> is also recorded.

If the <destination> is an 8-bit pointer, the sign is recorded by changing the four high-order bits of the last digit to 1 "1101" or 1 "1100" for negative or positive sign, respectively. Note that the last digit will appear to be a character (e.g. -55 => 48"F5D5" = "5N"), although the arithmetic function

```
INTEGER( <pointer expression>, <arithmetic expression> )
```

will distinguish it as a sign and will thus make the correct reverse mapping.

If the <destination> is a 4-bit pointer, then the sign is encoded by prefixing the hex digit D or C for negative or positive, respectively. INTEGER will not count the sign in its count of digits converted from the pointer, however; e.g., the following program fragment

```
HEX ARRAY A[0:10];
INTEGER I;
REPLACE A BY -12345 FOR *SDIGITS;
I:= INTEGER(A,5);
WRITE (FILEID, */, I);
```

will write

```
I=-12345,
```

D3435 ALGOL - REPLACE POINTER VALUED FILE ATTRIBUTE

The only form of <replace pointer-valued attribute statement> which is accepted by the ALGOL compiler when the destination is a <pointer-valued file attribute> is:

```
REPLACE <pointer-valued file attribute> BY <simple source>
```

D3436 ALGOL - "<FORMAL PARAMETER SPECIFIER>"

The documentation for <formal parameter specifier> on page 4-55 of the B7000/B6000 Algol Reference Manual (Form No. 5001639) is incorrect because it leaves out the <formal parameter list> for the fully-specified form of specification. The correct documentation is:

```
<formal parameter specifier>::=<empty>|
( );FORMAL|
<formal parameter part>;FORMAL
```

D3437 ALGOL - MESSAGE IN "<DISPLAY STATEMENT>"

The B7000/B6000 Algol Reference Manual (Form No. 5001639) page 5-34 under DISPLAY STATEMENT says that 'A message of less than 25 characters must be terminated by the character 4"00".' In fact, all such messages must be terminated by 4"00". This sentence should thus read 'A message must be terminated by 4"00".'

D3438 ALGOL - "[NO]" HAS NO EFFECT ON REMOTE FILES

On page 5-69 of the B7000/B6000 Algol Reference Manual (Form No. 5001639) statement is made that

"If the <record number or carriage control> is [NO], the buffer is not released after it has been read or written; i.e., the record can be read again..." This is not true if the file is a REMOTE file. [NO] has no effect if the file is KIND=REMOTE.

D3439 ALGOL - R IS A "<SINGLE PICTURE CHARACTER>"

The section on PICTURE DECLARATIONS in the B7000/B6000 ALGOL Reference Manual (Form No. 5001639) (pages 4-51 to 4-53) show the edit character "R" as a <picture character> and describes its action incorrectly.

"R" is a <single picture character>. Its action is as follows:

"If an E or F float has not ended (the preceding edit character was E or F but only the B character was transferred), the P or M character is inserted depending on whether the external sign flip-flop- is reset or set, respectively, and the float is ended. Otherwise, no action.

WARNING:

"The state of the external sign flip-flop is independent of the REPLACE statement that is using the picture. The external sign flip-flop is in the state in which it was left after the last operation which could affect it. If the last such operation was REPLACE<destination> BY <arithmetic expression> FOR <arithmetic expression> DIGITS, for example, the external sign flip-flop will reflect the sign of the <arithmetic expression> which was the source."

D3455 ALGOL - CLARIFICATION OF "DSCALELEFT" FUNCTION

The arithmetic function DSCALELEFT is intended for use only with a second parameter greater than or equal to 0 and less than or equal to 12. DSCALELEFT will produce unpredictable results if its second parameter (the power of ten by which the first parameter is to be multiplied) is outside this range.

D3471 ALGOL - ALLOW LONGER VALUE ARRAYS

The maximum size of a value array has been 4095 words. With the implementation of segmented value arrays (see Mark 32 ALGOL note D3266), it is now advantageous to allow value arrays that are much longer.

Value arrays which are "segmented" may now be up to 32767 words in length. Value arrays which are not "segmented" will remain limited to 4095 words.

D3490 ALGOL - CLARIFICATION OF "\$MERGE"

The description of MERGE on Page D-23 of the ALGOL Manual (Form No. 5001639) is correct except for cases where CHECK, SEQERR or NEWSEQERR are reset (these options are reset by default). The following description of the effect of the MERGE option is correct, whether or not sequence errors are being flagged:

"When SET, the MERGE compiler option causes the primary input (the CARD file) to be merged with the secondary input (the TAPE file).

If the MERGE option is not SET, only primary input is used, and the secondary input is totally ignored.

If MERGE is SET, the process of determining the input to the compiler is as follows, starting with the first card image in the TAPE file and the first card image in the CARD file following the image upon which MERGE was set:

If the next card image in the CARD files does not have the same sequence number as the next card image in the TAPE file, the image with the smaller sequence number is taken as input.

If the next card image in the CARD file has the same sequence number as the next card image in the TAPE file, the image from the CARD file is taken as input, and the image in the TAPE file is discarded.

The sequence number of card images from files being INCLUDED are not considered by the MERGE process; merging is suspended until the INCLUDE has been completed."

D3522 ALGOL - REPLACE INTO DOUBLE PRECISION ARRAY

If the <destination> pointer in the construct

```
REPLACE <destination> BY <real value> FOR
  <arithmetic expression> WORDS
```

points to a DOUBLE array, this does not have any effect on the REPLACE; i.e., <arithmetic expression> words will be replaced, not 2*<arithmetic expression> words. Also, if the <real value> is single-precision, then the tags of the words which had their values replaced will be zero (tag indicating single-precision operand).

The statement

```
REPLACE <destination> BY DOUBLE(<real value>) FOR
  2*<arithmetic expression> WORDS
```

will replace <arithmetic expression> double values, retaining the correct tag for double-precision values.

D3526 ALGOL - "<PICTURE DECLARATION>"

The second alternate for the production of the <picture symbol> <introduction> should be changed to show two <hexadecimal character>s are required. The correct production is as follows:

```
<introduction ::= <introduction code><new character> |
4<introduction code><hexadecimal character><hexadecimal character>
```

D3530 ALGOL - "FUNCTIONNAME," "LIBACCESS" ATTRIBUTES

Library declarations may now specify the FUNCTIONNAME and LIBACCESS attributes. FUNCTIONNAME is a string-valued attribute used to specify the system function name that will be used to find the target code file for the library. LIBACCESS is a mnemonic-valued attribute: the value BYTITLE indicates that the TITLE attribute of the library is to be used to find the library's code file. The value BYFUNCTION indicates that the FUNCTIONNAME is to be looked up in the MCP library function table which is maintained by the ODT message SL (see GENERAL note D3356) and the associated code file name will be used.

D3537 ALGOL - MAXIMUM SIZE OF SWITCH FORMAT

The upper bound on the size of a switch format is determined by how large one of the ALGOL compiler's internal arrays may be made by RESIZING. This in turn is determined by the size of the largest contiguous piece of memory which may be allocated on a system; i.e., OLAYROWSIZE. OLAYROWSIZE is set at cold-start time. (There is no uniform limit to the size of a switch format; the limit on any system is usually large, however.)

D3563 ALGOL - NO RESIZE/DEALLOCATE OF SEGMENTED ARRAY

RESIZE and DEALLOCATE of segmented arrays have never been implemented.

The ALGOL Reference Manual (Form No. 5001639) should be revised as follows:

Page 5-91, after the last paragraph, add the following:

"RESTRICTIONS

Segmented arrays may not be resized. See discussion of LONG ARRAYS under ARRAY DECLARATION."

Page 5-31, after the last paragraph, add the following:

"RESTRICTIONS

A row of a segmented array may not be deallocated. See discussion of LONG ARRAYS under ARRAY DECLARATION."

D3578 ALGOL - LEGAL INPUT TO FORMAT

Data of the form 123+4 is legal input to the D, E and F formats. For E and F, it is equivalent to 123E4. For D, it is equivalent to 123D4. There is an example of this in the ALGOL Reference Manual (Form No. 5001639) under output examples for D, E formats.

The following manuals should be revised to include examples of this type of input:

ALGOL Reference Manual, Pages 4-28 to 4-30.
FORTRAN Reference Manual, Page 13-4 to 13-7.

D3610 ALGOL - PASSING FILES BY REFERENCE TO LIBRARIES

The template and directory entries for files will now indicate that the file is being passed by reference instead of by name.

D3623 ALGOL - "OFFSET" AND "DELTA"

The OFFSET intrinsic has been rewritten to run much faster. It has also been changed to be consistent with DELTA and other pointer operations, by reporting single- and double-word pointers in units of eight-bit characters, instead of in 48-bit words.

The DELTA intrinsic is now much faster for the non-optimal cases. Its values remain unchanged for all proper uses.

For details, see Mark 32 MCP note D3606.

D3626 ALGOL - RESIZING "EVENT ARRAYS"

EVENT ARRAYS are now allowed to be resized. The syntax is analogous to that for an array row. The following are the changes to the syntax of the RESIZE statement on Page 5-91 of the ALGOL Reference Manual (Form No. 5001639):

```

<resize statement>::=RESIZE(<resize array identifier>,
                             <arithmetic expression><retain old>)
<resize array identifier>::=<array row>|<event array row>
<event array row>::=<event array identifier>|
                  <event array identifier>[<row designator>]

```

RETAIN must be specified for EVENT ARRAYS. Also, if the new size is not greater than or equal to the old size, a run-time error will occur.

D3633 ALGOL - "REAL (<POINTER EXPRESSION>)"

The function REAL(<pointer expression>) has been deimplemented.

D3654 ALGOL - "REAL (<POINTER EXP>, <ARITHMETIC EXP>)"

The following line on Page 6-26 of the ALGOL Reference Manual (Form No. 5001639) should be deleted:

"All bits of each character are used."

D3657 ALGOL - "NOBCL" COMPILER OPTION

The compiler control option NOBCL has been implemented. When set, this option generates an error for every attempt to use BCL pointers. This will facilitate the user in eliminating BCL constructs, so that programs will run on future non-BCL systems.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

ALGOL

P2724 ALGOL - STRINGS AS ATTRIBUTES, "TR" ITEMS

DISPLAY of a string expression did not work for string expressions which began with a SIGNAL or LIBRARY attribute of type string or a transaction record item of type alpha. This problem has been corrected. [Note that SIGNALs have been de-implemented.]

A boolean primary did not allow comparison of string expressions which began with a LIBRARY attribute of type string or a transaction record item of type alpha. This problem has been corrected.

For a transaction record assignment, if the length of the string expression and the length of the transaction record item are not the same, the lengths will be matched as follows. If the length of the string expression to the right of the assignment operator is less than the length of the transaction record item to which it is being assigned, the string expression will be padded with blanks. If the length of the string expression is longer than the transaction record item, the string expression will be truncated. Previously, this matching of lengths was not done.

The allocation of string temporaries was redone to use a stack of temporaries for each lex level. This will allow string temporaries to be reused more efficiently.

P2903 ALGOL - INTERNAL ARRAYS EXPANDED

The dimensions of some internal arrays have been expanded to avoid getting an INVALID INDEX while creating the stack image for the BINDER. This problem, which only occurred for very large programs, has been corrected.

P2916 ALGOL - CODE OPTIMIZATION CORRECTED

If the first instance of a Boolean expression optimization in a code segment occurred in the same syllable as the last Boolean expression optimization in the previous code segment, the compiler would generate bad code.

This problem has been corrected.

P3037 ALGOL - INCORRECT SIGN FOR COMPLEX EXPRESSION

In a complex expression of the following form, the sign of the imaginary part was not changed:

<real> - <complex>

This problem has been corrected.

P3038 ALGOL - INVALID ASSIGNMENTS NOT FLAGGED

After a relation of the following form occurred, assignments of complex expressions to arithmetic variables were not flagged as syntax errors:

<exp> = <exp>

This problem has been corrected.

P3039 ALGOL - COMPLEX EXPRESSIONS

Incorrect code was being generated for complex expressions with constant arithmetic subexpressions. This problem has been corrected.

P3040 ALGOL - ERRONEOUS SYNTAX ERROR WITH COMPLEX

It was possible to get an erroneous syntax error on complex expressions. The following example would give a syntax error:

Example:

```
BEGIN
    REAL A,B;
    COMPLEX C;
    C:=1 + A DIV B + C;
END.
```

This problem has been corrected.

P3041 ALGOL - INCORRECT SOFTWARE CONTROL WORD

Under certain conditions, involving strings or complex in an I/O list, an incorrect software control word was being built. In case of an error, a Halt/Load could occur. This problem has been corrected.

P3091 ALGOL - "INVALID OP" WITH LONG EXPORT LIST

An INVALID OP was occurring in the ALGOL compiler if an export list being compiled was exceptionally long (more than 60 entry points). This problem has been corrected.

P3092 ALGOL - "INVALID INDEX" WITH LEX LEVELS > "15"

An INVALID INDEX was occurring in the ALGOL compiler when compiling procedures with lex level greater than 15. This problem has been corrected.

P3139 ALGOL - CORRECT "XREF" OUTPUT FOR LIBRARY PROCEDURES

Incorrect environment information was being given to XREFANALYZER by ALGOL for procedures which were imported from libraries.

Example:

```
BEGIN
$SET XREF XREFFILES
  LIBRARY L;
  PROCEDURE P1; LIBRARY L;
  PROCEDURE P2; LIBRARY L;
END.
```

In the above example, P1 and P2 are imported from L. A subsequent request to INTERACTIVEXREF for DEC P2 would indicate that P2 was nested within P1: "PROCEDURE P2 OF P1: DECLARED AT ...", although this is not the case. This problem has been corrected.

P3154 ALGOL - "INVALID OP," INDEXED STRING ARRAY

When an element of a string array was passed as a value parameter and the string passed was long enough to be segmented, an INVALID OP would occur. This problem no longer occurs.

P3155 ALGOL - "INVALID OP" FOR STRING EXPRESSIONS

It was possible for code to be generated to access invalid or incorrect address couples when string expressions were used in WRITE statements, thus causing an INVALID OP or sequence error. This problem no longer occurs.

P3324 ALGOL - MULTIPLE "ELSE" CLAUSES IN "CASE" STATEMENT

Formerly a case statement of the form

```
CASE <arithmetic expression> OF
BEGIN
0: <statement>;
1: <statement>;
ELSE: <statement>;
3: <statement>;
ELSE: <statement>;
END CASE;
```

would not receive a syntax error (only the last ELSE was effective). Multiple ELSE clauses will now cause a syntax error.

P3406 ALGOL - PREVENT "INVALID INDEX"

The ALGOL compiler no longer DSeS with an INVALID INDEX compiling a TAKE function with an invalid parameter list.

P3407 ALGOL - CALL "BLOCKEXIT" TO DEALLOCATE BOUND GLOBALS

An ALGOL program which had no global declarations of the type requiring a call on the MCP procedure BLOCKEXIT could have such globals added by binding. Since no call on BLOCKEXIT was made, however, these added globals would cause dumps by FORGETCHECK, etc. This problem has been corrected.

P3408 ALGOL - "INVALID INDEX"

An INVALID INDEX no longer occurs when a certain type of syntax error (using a string literal as the actual parameter to a procedure expecting a <character type> array) is encountered.

P3409 ALGOL - ACCEPT "<STRING VARIABLE>")"

The ALGOL compiler no longer neglects to emit a conditional branch when ACCEPT (<string variable>) is used as the Boolean condition in an IF statement.

Example:

```
IF ACCEPT(STRING1) THEN . . .
```

Previously, the THEN part of the statement would never execute; occasionally, a stack overflow would occur.

P3410 ALGOL - MAKING USE OF AVAILABLE SPACE

The code in the ALGOL compiler that makes use of UNUSED space at the end of a code file now will correctly achieve its purpose.

P3411 ALGOL - INTRINSIC AS A NAME PARAMETER

The ALGOL compiler failed to test for intrinsics and inline intrinsics when they were passed as name parameters and was emitting a NAMC (0,0) instead of generating the correct code. The problem arose only when the intrinsic being passed had no parameters; e.g., LINENUMBER. This problem has been corrected.

P3412 ALGOL - PICTURE AS A FORMAL PARAMETER

When calling a procedure which had a picture as a formal parameter, no check was made to ensure that the actual parameter had been declared. This problem has been corrected.

P3413 ALGOL - "CTPROC, CTDEFINE" VS. PARAMETER LIST

An erroneous syntax error is no longer emitted when using a compile-time define to generate a parameter list.

P3414 ALGOL - COMPLEX TIMES REAL MULTIPLICATION

Incorrect code was being generated in some cases for multiplication of a complex operand and a real operand. This problem has been corrected.

P3415 ALGOL - CLEAR "SCW" INFORMATION

The Software Control Word information is now cleared before it is used instead of after it is used, thus correcting a problem involving binding.

P3416 ALGOL - STRING POOL EXCEEDED WITH "\$INTRINSICS"

Strings were not being returned to the string pool after they were used in the outer block of a program which had the dollar option INTRINSICS set, thus causing the run-time error STRING POOL EXCEEDED to occur. This problem has been corrected.

P3417 ALGOL - FLAG "BCL" POINTERS WITH OFFSET

The option \$NOBCL would not flag a pointer expression with an offset. An error is now generated.

P3418 ALGOL - LONG CHARACTER ARRAYS

Previously, the ALGOL compiler would flag as an error a long character array declared globally to an externally-compiled procedure. Such global declarations are now allowed.

Example:

```
[
  LONG EBCDICARRAY A[0];
]
PROCEDURE X; BEGIN END.
```

P3463 ALGOL - CALLING USERIOERROR FOR "MYSELF.TASKFILE"

An I/O error when writing to MYSELF.TASKFILE no longer results in a memory dump.

P3498 ALGOL - "INVALID INDEX" AFTER PARAMETER MISMATCH

When a string literal is passed to a procedure expecting an EBCDIC array, an INVALID INDEX no longer occurs.

P3526 ALGOL - PREVENT POSSIBLE STACK OVERFLOW

If a stack overflow condition arose when a program attempted to compare two pointers (not the string pointed at by the pointers), the stack would not be "stretched" and the program would be DSed for "STACK OVERFLOW".

Example:

```

    POINTER P1,P2;
    .
    .
    IF P1=P2 THEN . . .

```

The stack will now properly get stretched in the above example.

P3532 ALGOL - CALL RESETPOOLSTRINGSIZE

A typed procedure which had only a single statement (no BEGIN-END) and used string temporaries no longer fails to deallocate them. The following example now works correctly:

```

    INTEGER PROCEDURE P;
    P:=DECIMAL(STRING("123",3));

```

P3533 ALGOL - CORRUPTION OF VALUE ARRAYS

If part of a value array declaration comes from an INCLUDE file, the contents of the value array are no longer corrupted; previously, the INTNAME of the INCLUDED file would become part of the value array.

P3560 ALGOL - "BCL" CONSTRUCTS REMOVED

BCL constructs within the compiler symbolic have been removed. All references have been replaced with equivalent constructs not using BCL. The compiler will generate BCL code when the program being compiled so specifies.

P3611 ALGOL - GLOBAL "STRING PROCEDURE" AND BINDING

Previously, the use of a global STRING PROCEDURE having a forward which was to be bound would result in the erroneous syntax message "SPECIFICATION DIFFERS FROM FORWARD DECLARATION". This problem has been corrected.

P3625 ALGOL - INVALID "I/O" LIST ELEMENTS

Previously, the compiler would get an INVALID INDEX when a WRITE statement using an invalid LIST statement occurred. Now, the compiler generates a syntax error for the invalid LIST statement and does not get an INVALID INDEX.

P3661 ALGOL - GIVE ERROR FOR SPACES WITHIN NUMBERS

According to the ALGOL Reference Manual (Form No. 5001639), spaces are not allowed within numbers. Unfortunately, some instances of spaces within numbers were not being detected by the ALGOL compiler. Moreover, the code generated for these instances was incorrect; i.e., it was not the same as for the number formed by removing the spaces.

Examples:

```

    X:=1. 2;
    X:=1 .2;
    X:=1.2 3;
    X:=1 . 2;

```

This problem has been corrected; spaces within numbers are now treated as errors.

P3774 ALGOL - MISSING "BLOCKEXIT"

If the option NOBINDINFO were SET, no BLOCKEXIT call or TAG SIX word was emitted for the main block (level 2) of a program. When executed, a FORGETCHECK dump occurred if there were level 2 items to be deallocated. This problem has been corrected; now, a BLOCKEXIT call and a TAG SIX are emitted if necessary.

P3798 ALGOL - LOCKING CODE FILE

When compiling two or more procedures with \$LIBRARY set, the code file would be locked if only the last procedure had no syntax errors. The codefile should be locked only if all procedures are compiled without errors. This problem has now been corrected.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

ALGOL INTRINSICS

P2940 ALGOLINTRN - BACKUP FILE SEARCHING

Backup file searching by the BATCHMONITOR intrinsic has been improved.

P3298 ALGOLINTRN - "CTOD" TERMINATES ABNORMALLY

Under certain conditions, the CTOD intrinsic would terminate with an "INVLD SQRT ARG". This problem has been corrected.

P3325 ALGOLINTRN - CORRECT "DSQRT" ERRORS

Two classes of errors in the Double Precision Square Root Intrinsic have been corrected. Square Roots of values which are double precision integers would not work because of incorrect normalization of the value. Also, very large numbers were handled incorrectly. Both of these errors have been corrected.

P3326 ALGOLINTRN - CORRECT "GAMMA, DGAMMA"

Both GAMMA and DGAMMA produced results for negative integers and zero. These values now produce errors since the function is undefined at these values.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

ALGOL/PLI INTRINSICS

D3628 ALGOLPLINTRN - "ALGOLPLINTRN" SUBSUMED BY "GENERALSUPPORT"

Effective with the Mark 32 release, the ALGOL/PLI intrinsics have been subsumed by the GENERALSUPPORT library, described in GENERAL note D3354.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

ATTABLEGEN

D3076 ATTABLEGEN - "APL" FILE ATTRIBUTE

The file attribute APL has been implemented for disk/pack files to satisfy APL access restrictions. It may be set prior to the creation of a new file, and is effective only if the codefile creating the file itself has the attribute set. The attribute may be read any time the file is open. An attempt to open a file with APL set will cause the accessor to be DSeD unless it also has APL set.

D3425 ATTABLEGEN - DELETE "PORTS, SIGNALS"

Attributes and mnemonics for PORTS and SIGNALS have been removed. In addition, the PORTS mnemonic for the task attribute OPTION has been removed.

D3656 ATTABLEGEN - LASTSTATION SYNONYM "= LASTSUBFILE"

A synonym for the LASTSTATION file attribute has been added, LASTSUBFILE. LASTSUBFILE is the preferred name, which will be displayed on attribute error messages to either name.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

ATTABLEGEN

P3626 ATTABLEGEN - "ROWSIZE=504" FOR NEW PATCH

ATTABLEGEN now uses the standard rowsize of 504 segments for its output patch.

DOCUMENT CHANGES NOTES (D NOTES)

BACKUP
-----**D3587 BACKUP - "HOSTNAME" MODIFIER**

BACKUP syntax has been extended to allow "HOSTNAME=<string>". This causes the output to be printed at the host named by <string>. Appearance of this option causes the ND option to be set, as direct I/O is not allowed across a network.

Example:

PB D 1234 HOSTNAME="HONOLULU"

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

BACKUP

P3327 BACKUP - "BFILE" LABEL EQUATION

When running BACKUP, with ND set and BFILE label equated to certain devices such as disk or tape, bad output would be obtained because the device could not handle the printer backup file's carriage control information. This problem has been fixed. For each printer space or skip, BACKUP will now write a blank line if the device can not handle the carriage control information.

P3658 BACKUP - "FILE.TITLE" ATTRIBUTE

A problem has been corrected where a backup file name with multiple levels (not including the task number and modified file name) would give an attribute error. The error only occurred when IDOPTION was set.

DOCUMENT CHANGES NOTES (D NOTES)

BARS

D2978 BARS - "SYSTEM/BARS" UTILITY

The SYSTEM/BARS utility, described in the Mark 31 system notes, has been revised for the Mark 32 release. The following note describes the SYSTEM/BARS utility; revisions for the Mark 32 release are indicated by PCN bars in the right margin.

The SYSTEM/BARS utility monitors the system's performance and displays it in the form of numeric values and bar graphs. Various system performance parameters are sampled and may be displayed in a user-controllable format on screen-type terminals.

Performance elements are represented by a bar on the screen. The format of each bar is ###XXX... where the #'s extend to the minimum value seen, the X's extend to the current value, and the .'s extend to the maximum value seen. The display is updated every 'cycle' seconds, where cycle is a parameter which is dynamically variable from the terminal, and a running average is maintained over a user-settable period.

The following verbs are used to control the display:

HELP (alias TEACH)

Displays this information.

CYCLE

Controls the sampling and terminal update interval.

PERIOD

Several values are computed as running averages using the formula:

NEWAVERAGE:=
(OLDAVERAGE * (PERIOD-CYCLE) + NEWVALUE * CYCLE) / PERIOD

PERIOD changes the value of period. If PERIOD < CYCLE, no averaging is done (i.e. the exact value is displayed).

NEWDISPLAY

Modifies the format and content of the screen. Each item on the screen is specified by giving a key word followed by N's for the value field and B's for the bar graph (both N's and B's are optional); for example:

Idle NN BBBBBBB

will produce a display of the form: Idle 45 ###X. Text may be placed on the screen by placing it within single quotes: 'xyz'. The valid key words and their meanings can be displayed with the WORDS verb. A default display is supplied by the program. The NEWDISPLAY input that creates it can be displayed with the DISPLAY verb.

DISPLAY

Displays the input that would create a specified display screen. Three options exist:

DISPLAY Shows the NEWDISPLAY input for the current screen.

DISPLAY DEFAULT

Shows the NEWDISPLAY input for the default screen.

DISPLAY <file name>

Shows the NEWDISPLAY input for a screen SAVED in the specified file; e.g. DISPLAY X/Y ON P.

SAVE <file name>

Saves the current display in the specified file; e.g. SAVE X/Y ON P.

LOAD

Loads a previously generated screen as the new screen. The options for DISPLAY are also valid for LOAD:

LOAD Loads the current display. The screen does not change.

LOAD DEFAULT

Loads the default screen for monolithic systems.

LOAD TCDEFAULT

Loads the default screen for a B6800 multiprocessor system.

LOAD TCMEMORY

Loads a screen giving overlay information for each memory structure of a B6800 multiprocessor system.

LOAD <file name>

Loads a previously SAVED display.

WORDS

Displays the allowed key words for the NEWDISPLAY input plus a short description of their meaning.

PACK (alias PK, alias PERPK)

Displays the names, channel numbers, and family indices of the packs currently on-line.

STED Displays the number of tasks "SUSPENDED BY THE SYSTEM" as opposed to those stopped by either the operator or programatically.

BYE Ends the program.

The program begins by executing a LOAD DEFAULT command. The program is initialized to a different display if the file DISPLAY is label-equated to a SAVED file; for example, FILE DISPLAY(TITLE=MY/SCREEN).

If the file MONITOR is label-equated, the program will write the raw performance data to that file as it is received from the MCP; for example, FILE MONITOR(TITLE=XX).

D3277 BARS - VIRTUAL MEMORY UTILIZATION MEASURE

BARS now has the ability to display the quantity of virtual memory (overlay space) actually in use by the system. The information may be displayed as a total for the system or box-by-box. Virtual memory data for swap tasks is also available. For batch tasks, the new keywords BVMEM, BVMEM-1, BVMEM-2, BVMEM-3, BVMEM-4 and BVMEM-G are used for monolithic or tightly-coupled summary information, and box number 1, 2, 3, 4 and * GLOBAL tm Memory, respectively. Virtual memory display for SWAPPER is invoked via the keyword SVMEM. The SWAPPER data is available only as a system-wide summary. The new items may be displayed similarly to OLAY, AVAIL, SAVE, etc.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

BARS

P2909 BARS - PROCESSOR TIMES REPORTED ON MONOLITHIC SYSTEMS

BARS was incorrectly computing the processor time allocations on monolithic (B6700, B7700 and 1-processor B6800) systems. This has been corrected.

P2910 BARS - CORRECT SWAPCORE GRAPHS

If a B6800 Multiprocessor system had non-identical local memory configurations, the bar graphs for swapcore could be incorrectly displayed. This has been corrected.

P3110 BARS - INCOMPLETE DISPLAY ON SYSTEM "ODT"

When BARS is run from the system ODT, lines of data or partial lines of data would sometimes be erased. This problem has been corrected. Each screen update will now result in the correct display.

P3251 BARS - CLEAR CHANNEL INDICATORS PROPERLY

The last channel accumulator was not properly being cleared between updates. Now, the accumulator will be cleared with all the other channel accumulators.

P3464 BARS - "SCREEN" FILE ATTRIBUTE

When the file attribute SCREEN is used in BARS, an error "ILLEGAL FILE ATTRIBUTE" would occur because SCREEN is already defined within the program. This problem has been corrected.

P3481 BARS - CHANGES TO TYPE "4 SYSTEMSTATUS" CALL

BARS has been modified to accommodate the changes described in MCP-GENERAL note D3500.

Channel wait conditions can only be reported for MPX systems. The needed information is not available on MLIP systems, and BARS will not attempt to display it on these.

P3642 BARS - "<MORE>" DISPLAYED COMPLETELY

Running BARS in SPO mode, the output message to indicate an additional page of information was displayed as "ORE>" on the top line instead of "<MORE>" on the bottom line. This problem has been corrected.

P3643 BARS - "SPO" MODE DISPLAYS LONG MESSAGES

Running BARS in SPO mode and entering "MSG" caused a SEG ARRAY error if the messages were long (full) lines. This problem has been corrected.

P3644 BARS - NEGATIVE "IDLETIME"

It was possible to obtain a negative value for IDLETIME in certain situations. BARS has been corrected to display only values of 0 or greater IDLETIME.

P3645 BARS - SINGLE "." AS INPUT

Running BARS in SPO mode and inputting a single "." caused an output <MORE> loop. This problem has been corrected; the message "UNRECOGNIZED REQUEST" is displayed in this case.

DOCUMENT CHANGES NOTES (D NOTES)

BASIC

D3538 BASIC - MATRIX INVERSION FUNCTION

The matrix inversion function INV will on occasion complain that the matrix to be inverted is singular, even though that is not the case. The problem is data dependent, as the function works correctly in many cases.

D3539 BASIC - "APPEND" STATEMENT

The APPEND statement was never fully implemented. APPEND is able to add records only within the last block of a file, because BASIC files are closed with CRUNCH. If a program attempts to add records past the last block, an end-of-file condition may occur.

Another known problem is that data written to a file after using the APPEND statement may not appear on the file if the file were previously opened input.

D3540 BASIC - "UP-ARROWS" REQUIRED FOR EXPONENTIAL FIELD

On pages 7-17 and 7-18 of the BASIC Reference Manual (Form No. 5001407), four consecutive percent (%) symbols appear in the explanation and example for exponential fields in PRINT/WRITE USING. The percent symbols are not literally the correct characters in that case; the percent symbol is used to represent the up-arrow character which is actually required.

The sequences of four consecutive percent characters should be replaced by sequences of four up-arrow characters.

D3603 BASIC - PROGRAM OF UP TO "2048" STATEMENTS

The compiler will now accept programs of up to 2048 statements.

Add the following sentence to Page 1-7 of the BASIC Reference Manual (Form No. 5001407):

"A program may contain up to 2048 statements."

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

BASIC

P3612 BASIC - ERROR FOR "DEF" FUNCTION

A syntax error is now generated when the number of parameters supplied during invocation of a DEF function is different from the number of parameters declared for that function.

P3622 BASIC - FLAG QUESTION MARK AS INVALID CHARACTER

The character "?" (question mark) is now flagged as an invalid character. Previously, it caused the compiler to ignore the remainder of the line.

P3664 BASIC - ERROR ON FIRST PROGRAM TOKEN

The compiler will no longer abort with an INVALID OP if the first token in the program is a quoted string. The appropriate syntax error is now generated.

DOCUMENT CHANGES NOTES (D NOTES)

BINDER

D3002 BINDER - CODE FILE NAMING CONVENTION FOR "ALGOL"

The following change to the BINDER Reference Manual (Form No. 5001456), is necessary because there is no SEPARATE option in ALGOL or ESPOL.

Change the second sentence under Pragmatics, Page 5-4, to read as follows:

"When compiling one or more subprograms, the FORTRAN compiler (when the SEPARATE option is SET), the PL/I compiler (when the MULTIPLE option is SET), and the ESPOL and the ALGOL compilers will create one or more code files with a title formed by replacing the last <file identifier> of the code file title on the compile card with the name of the separate subprogram."

Add the following sentence to the last paragraph on Page 5-4:

"LIBRARY is set to TRUE automatically when compilation of an ALGOL or FORTRAN program is initiated from CANDE."

D3414 BINDER - DELETE OLD INTRINSICS

Since XALGOL is no longer supported, the INTRINSICS and the BINDER references have been deleted.

PL/I was changed on Mark 30 PR1 release so that the compiler references the MCP's ATTRIBSEARCHER. Therefore, ATTRIBSEARCHER has been deleted from the PL/I INTRINSICS.

The old BASIC intrinsics have been deleted since the codefiles which reference these are no longer supported.

D3487 BINDER - "MCP" CODE FILE ROW SIZE = "504"

The MCP no longer has any code segments large enough to require that the MCP code file have a row size of 1008 disk segments.

The NEWP compiler and the BINDER will now construct an MCP code file with a row size of 504 segments, as is done for other code and symbol files.

D3495 BINDER - INSTALLATION INTRINSIC WARNING

The BINDER now issues the following warning when requested to bind installaton intrinsics:

"BINDING INSTALLATION INTRINSICS WILL BE DE-IMPLEMENTED ON 34"

GENERAL note D3354 describes the overall plan for replacing intrinsics with support libraries.

D3562 BINDER - "INTER-LANGUAGE" PROCEDURE PARAMETERS

The BINDER does not allow inter-language binding of procedures that have procedure parameters.

For example, a FORTRAN subroutine that has a parameter declared EXTERNAL may not be bound into an ALGOL host that declares an appropriately-named EXTERNAL procedure that has a procedure parameter.

ALGOL HOST:

```
BEGIN
  PROCEDURE P1(PP);
    PROCEDURE PP(X); INTEGER X; FORMAL;
    EXTERNAL;
    PROCEDURE P2(I); INTEGER I;
    BEGIN
      END;
  .
  .
  P1(P2);
END.
```

FORTTRAN SUBROUTINE:

```
SUBROUTINE P1(PP)
  EXTERNAL PP
  INTEGER I
  .
  .
  CALL PP(I)
END
```

B6000 SERIES MARK 32

The BINDER Reference Manual (Form No. 5001456) has never clearly stated this restriction; the following text should be added to the BINDER manual:

Page 7-2A, at the end of the PARAMETERS paragraph:

"Procedures may not be passed as parameters between ALGOL and COBOL."

Page 7-5, at the end of the PARAMETERS paragraph:

"Procedures, functions and subroutines may not be passed as parameters between ALGOL and FORTRAN."

Page 7-7, at the end of the PARAMETERS paragraph:

"Subroutines and functions may not be passed as parameters between COBOL and FORTRAN."

Page 6-1, at the end of the ALGOL SUBPROGRAM paragraph:

"For the most part, ALGOL, DCALGOL and DMALGOL are considered to be the same language. Mixing procedures of these three languages via binding is considered to be INTRA-LANGUAGE; however, the BINDER considers these languages to be distinct when dealing with formal procedure parameters. As with all other inter-language binding, passing formal procedure parameters between these languages is prohibited by the BINDER."

D3566 BINDER - ARRAY BOUND CLARIFICATION

Passing array parameters between ALGOL and FORTRAN should be clarified in the BINDER Reference Manual (Form No. 5001456), by adding the following to sub-paragraph 1 of the PARAMETERS paragraph on Page 7-5:

"For ALGOL arrays passed to FORTRAN routines, the ALGOL subscript value of zero corresponds to FORTRAN subscript value one. For FORTRAN arrays passed to ALGOL, the FORTRAN subscript value used in evaluating the parameter (the value one is used if no subscript is specified) corresponds to the ALGOL subscript value zero."

D3577 BINDER - "MCP BIND" EXAMPLE

The WFL subroutine WFLCOMP on Page 14-2-9 of the SOG Reference Manual, Volume 2, (Form No. 5001688), should have an additional \$ option, LIBRARY, added following NEWSEQERR.

On Page 14-2-10, the following four lines:

```

BIND WFL FROM SYSTEM/=;
BIND CCSTRINGCONV FROM SYSTEM/=;
BIND CCVARIABLEPPB FROM SYSTEM/=;
BIND CCSTRINGFUNCTION FROM SYSTEM/=;

```

should be replaced by the following:

```

BIND WFL,CCSTRINGCONV,CCVARIABLEPPB,CCSTRINGFUNCTION FROM SYSTEM/WFL;

```


SOFTWARE IMPROVEMENTS NOTES (P NOTES)

BINDER
-----**P3010 BINDER - MULTIPLE REBINDS OF "SYSTEM/INTRINSICS"**

When rebinding the SYSTEM/INTRINSICS, the BINDER was not properly updating the binding information stored in the code file. A subsequent rebind would then improperly allocate stack addresses, resulting in unusable intrinsics. This problem has been corrected.

P3328 BINDER - BINDING "FORTRAN" ROUTINES

The FORTRAN compiler on the initial Mark 30 software release (30.140) built incorrect Binding information for some intrinsics. This problem was fixed on the 30 PR1 release and the initial Mark 31 software release. FORTRAN routines compiled with the 30.140 compiler could not Bind to programs created with other compilers nor to programs created with earlier or later FORTRAN compilers. Now, those FORTRAN routines created with 30.140 compiler are properly bound.

P3360 BINDER - CORRECT "\$WAIT" WITH <MIX NO.> "OF"

The \$WAIT option causes the BINDER to wait on a "NO FILE" RSVP when a required input file is missing. If the response to one of these "NO FILE" situations was an OF (Optional File) response, the BINDER would use this response for all future "NO FILE" instances.

This problem has been corrected; the BINDER will only interpret the OF as a response to a single "NO FILE" RSVP.

P3419 BINDER - BINDING WITH "D[0]" INTRINSICS

When a user procedure contained calls on D[0] intrinsics, the BINDER did not correctly fix all references to called procedures, thus causing an INVALID OP when attempting to call the D[0] procedure. This problem has been corrected.

P3587 BINDER - BINDING PROGRAMS WITH "\$DATADICTINFO"

The BINDER was generating "ASSERT=FALSE" error messages when binding COBOL code files with \$DATADICTINFO set. This no longer occurs.

DOCUMENT CHANGES NOTES (D NOTES)

CANDE

D3249 CANDE - "VISIBILITY" TASK, "SCATTER" RUN-TIME OPTIONS

Introduction

This note describes CANDE's interaction with the new semidependent task feature of the MCP. An understanding of that feature (described in MCP-GENERAL note D3252) is assumed in the following discussion.

Semi-dependent tasks permit a parent running in a local box of a B6800 Multiprocessor system to have offspring in other local boxes, subject to certain visibility constraints. The user tasks spawned from CANDE meet these requirements, but certain actions must take place to allow the system to run these offspring to the fullest advantage. These are described below. The term "offspring" refers to user tasks (i.e., COMPILE, EXECUTE, UTILITY).

VISIBILITY Attribute

The VISIBILITY task attribute can be specified as a run-time task modifier; e.g., "RUN X; VISIBILITY=GLOBAL". Values for the attribute are checked via ATTRIBSEARCHER.

Scattering

There are two reasons for running CANDE in a local box. First, it may be desirable to limit CANDE and all of its offspring to that local box to permit the other local box(es) to run free of interference from that source. Second, CANDE may be run in a local box to maximize its performance, with the intent to run user-initiated offspring in the most optimal part of the overall B6800 Multiprocessor system.

In either of these cases, the site must take some overt action to allow or prevent the scattering of the offspring. Because of the current semantics of the SUBSYSTEM attribute as applied to CANDE, the default is to run offspring in the same local box as CANDE, and the new CANDE run-time option SCATTER (#13) is set to allow CANDE's offspring to run in other local boxes.

If SCATTER is set, CANDE sets the task stack's SUBSYSTEM to null; this allows the MCP to pick an appropriate subsystem at task initiation time. If the option is reset, CANDE sets the task stack's subsystem to its own (main CANDE stack) value.

The minimum abbreviation for SCATTER is "SCAT". The setting of this option is maintained in the tankfile in the same manner as the other options. If the value of SCATTER is changed, the effect of the change does not occur until the next initiation of the task stack.

The revised syntax for the CANDE options command is the following:

```
-- ?OP -----|
| - + - | | <----- , -----|
| - - - | | SWAPALL -----|
|         | |
|         | | - ALLLOGIN ---|
|         | | - DIALLOGIN ---|
|         | | - KEEPSTA ----|
|         | | - ALLMSG -----|
|         | | - CATDEFAULT - |
|         | | - CATALOGOK -- |
|         | | - DUMPOK -----|
|         | | - SCATTER ----|
|         | | -----|
```

Operational Considerations

Following are five possible ways to run CANDE to achieve various results. The method selected may be controlled by site management to achieve optimum performance in their environment.

1. Run CANDE in global; scatter both the worker stacks and the user tasks to the local boxes. This remains the default.

2. Run CANDE in global; scatter offspring in some local boxes: Compile CANDE with an appropriate multiprocessor subsystem specification with default VISIBILITY.
3. Run CANDE and all the workers in any one of several local boxes, but scatter the user tasks among all those boxes: Compile CANDE with an appropriate multiprocessor subsystem, and with VISIBILITY=MINIMAL.
4. Run CANDE and all offspring in a single local box: Compile CANDE with an appropriate single-processor subsystem specification with default VISIBILITY.
5. Run CANDE and the workers in a single local box, but scatter user tasks across the system: Use method 3 or 4 and set the SCATTER option to avoid propagating CANDE's SUBSYSTEM attribute.

Note that SYSTEM/CANDE can be compiled or bound once with a special SUBSYSTEM specification, say "CANDESUBSYS". Then method 1, 2 or 4 above can be achieved merely by changing the definition of CANDESUBSYS, via the MS ODT command, to specify all, some, or one of the processors on the system, respectively.

The capability to run CANDE in local with the workers in several local boxes cannot be offered, since the worker processes are fully dependent and use their visibility into the parent stack.

D3364 CANDE - AUTOMATIC "DESTNAME" FOR "CANDE" SESSIONS

The DESTNAME as applied to tasks executed through CANDE provides a means of specifying a non-CANDE station for the task's printed output. This note describes a facility to specify a DESTNAME for a complete session, automatically supply that DESTNAME when a task is executed and to control the setting of the automatic DESTNAME.

This implementation presumes that the physical configuration of an installation is relatively static. For example, an RJE terminal used to provide printed output for program compilations will probably stay "near" the programmers who use it. Thus the DESTNAME for a session is associated with a user, and the user's default DESTNAME is kept in the USERDATAFILE. This default is supplied on all task executions explicitly initiated by the user as well as CANDE-initiated tasks (e.g. CANDE WRITE).

The user may change the DESTNAME for some portions of a session, revert to the default value, or interrogate the current value. One possible value is null; that is, the DESTNAME value may be set so that no DESTNAME is supplied for a task execution. The user may also override the current DESTNAME by explicit specification on a run or compile request.

If the permanent value of the DESTNAME is to be changed in the USERDATAFILE, SYSTEM/MAKEUSER may be invoked by the installation.

Syntax:

```
-- DESTNAME -----|
----|
    | -<station name>-|
    | - * -----|
    | - . -----|
```

If no input other than the DESTNAME verb is entered, CANDE reports the current value of the DESTNAME. If the <station name> is supplied, it is saved by CANDE as the current value until another DESTNAME action takes place. The input "*" causes the current value to revert to the user's default, and the input "." causes the DESTNAME to be set to null. In the latter case, no DESTNAME is supplied by CANDE on any task execution request.

As indicated in the syntax diagram, the minimum abbreviation for DESTNAME is DEST.

There are no changes in the RUN (EXECUTE) statement syntax. If a DESTNAME is explicitly supplied, it overrides the current value for the session for that request only. There is no abbreviation for DESTNAME on the run request.

Backup-related requests (BDREMOVE and BACKUPPROCESS) search for BD, BP, REMLP<nn> and REMCP<nn> files, where <nn> is the NDL-defined MCS number for each destination station used during the current session. Searches are made on DISK, PACK and the DL BACKUP device which was in effect at the beginning of the session. The message "NO BACKUP FILES FOUND" is emitted if no such files are found.

The USERDATAFILE entry is made using the word CANDEDESTNAME, which is of type FILE (that is, a station name has the form of a file title).

D3370 CANDE - FULL SCREEN SEQUENCE MODE

The ability to use a full TD8xx screen for CANDE sequence mode has been available for several releases, but has not been recently documented. If a TD8xx terminal is removed from scroll mode (?-S) and sequence mode is requested from CANDE, the entire screen is set up for sequenced input. Multiple lines must be separated with a carriage return, and sequencing may be terminated (as in single-line sequencing) by inputting an ETX or carriage return character in the first character position of an offered line. This facility is provided by the standard SYMBOL/SOURCENDL.

This note is documentary in nature only; no modifications to the system software are involved.

D3456 CANDE - "FILES" <FILENAME> : <DEPTH #>

In the CANDE Reference Manual (Form No. 5011398), the second paragraph of the discussion of the FILES command (Page 4-30) should read as follows:

"The <depth> option indicates the number of levels of <file identifier> beyond the supplied <file name> or <directory name> which will be listed (if present)."

In the syntax for <directory name> on Page 3-1, <name> should be changed to <file identifier>.

D3544 CANDE - "COPY/ADD" STATEMENT

The syntax specification for the COPY/ADD statement in the WFL Reference Manual (Form No. 5001555) should be changed to read as follows:

```

----- COPY ----->
| - ADD -- | | - & ----- COMPARE - | | - & ----- CATALOG - |
|          | | - AND - | | - AND - | | - BACKUP -- |
>--<libmaint file list>----->
|          | | <----- , ----- | |
|          | | --- TO <volume specification> --- |
>----->
| - [ <task id> ] - |

```

The syntax specification for the COPY/ADD statement in the CANDE Reference Manual (Form No. 5011398) should be changed to read as follows:

```

----- COPY ----->
| - ADD -- | | - & -- COMPARE - | | - & --- CATALOG - |
|          | |          | | - BACKUP -- |
>--<libmaint file list>----->
|          | | <----- , ----- | |
|          | | --- TO <volume specification> --- |
>----->
| - [ <task id> ] - |

```

D3580 CANDE - PASSING STRINGS VIA "WFL/CANDE"

When WFL or CANDE pass a string to a program as a parameter, they use a word array just large enough to hold the string plus one null character. If the declaration of the parameter in the receiving program is longer than the size of the string passed by WFL, a run-time error may occur.

Such run-time errors do not occur on the Mark 30 release, because WFL at that time copied the string into a 256-character array before passing it to a program. Thus, WFL was erroneously allocating too much storage for the string. This was an oversight which was corrected on the Mark 31 release; it is now implemented as intended.

D3642 CANDE - "CANDE" VS. FOREIGN TASKS

THE ?STA command to CANDE now provides the mix number and hostname for foreign tasks run from a CANDE terminal.

D3646 CANDE - COMPILER TYPE "FORTRAN77"

CANDE now recognizes the filekind FORTRAN77. The corresponding CANDE type is F77.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

CANDE

P2798 CANDE - "SCHEDULE" RESTART PROBLEMS

A SCHEDULE session restart was not picking up the right restart point under some circumstances. The effect was either to cause restart at the wrong point in the session or to fail to restart at all. Also, following restart, the beginning of the block containing the end of the output from the aborted session could be overwritten with incorrect information. Both these problems have been corrected.

P2833 CANDE - MISSING "SCHEDULE" OUTPUT FILE

CANDE could fault if errors occurring in the SCHEDULE input file were reported in a non-existent output file. These errors are no longer reported if the file does not exist.

P3200 CANDE - SCHEDULE REQUEST ON UNNAMED WORKFILE

A request to schedule an unnamed workfile without specifying the output file title was incorrectly producing the message "INVALID NAME: SCHOUT/" followed by some incorrect file title (or garbage). The problem has been corrected by having CANDE notice that the workfile has no name and display the message "#WORKFILE NOT NAMED".

P3329 CANDE - MISSING "SCHOUT" FILE

Under some circumstances, an attempt to update the SCHOUT file caused a DIVIDE BY ZERO fault if the SCHOUT file were missing. This problem has been corrected.

P3420 CANDE - SECURE SCHEDULE FILES PROPERLY

The security of the schedule output file is now set to PRIVATE when it is created.

P3421 CANDE - ALLOW SETTING CHARGECODE TO NULL

The code which set a chargecode to null was not consistent with the other code in CANDE which checked for the presence of a chargecode. This problem has been corrected.

P3422 CANDE - "TAPE" COMMAND IN "DO" FILES

Occurrences of a TAPE command in a DO file would cause all following commands in the file to be flushed when the TAPE command was completed. This problem has been corrected.

P3423 CANDE - RECOGNIZE "NDL" SEQUENCE MODE TERMINALS

CANDE was examining the wrong field in its attempt to determine whether a terminal had NDL sequence mode capability. This problem has been corrected.

P3424 CANDE - SECURITY PROBLEM

A security problem in CANDE has been corrected.

P3425 CANDE - "CANDE" CREATES BAD TANKFILE

If the INTERCOMQUEUE could not be set up following creation of a new tankfile, the new (but incompletely initialized) tankfile would not be saved. This could cause problems in a subsequent initiation of CANDE. The new file is now saved only after it has been completely initialized.

P3426 CANDE - SCHEDULE SESSIONS VS. CHARGECODES

Setting of the chargecode in schedule sessions has been added. This area was overlooked during some previous chargecode modifications.

P3460 CANDE - "CANDE" ERRORS NOW ATTRIBUTED TO USER ERRORS

The "FILECD" and "UTLTSK" CANDE errors were originally devised to notify the user of an internal CANDE problem. These errors are in reality caused by inconsistent software levels or by improper user input (possibly due to hardware malfunction; e.g., noisy data comm line). The following error messages are now emitted for these situations:

"FILE LABEL EQUATION ERROR" instead of "FILECD"

or

"FILEKIND OF WORKFILE INCOMPATIBLE WITH THIS MCP" instead of "UTLTSK".

P3499 CANDE - ALLOW MORE THAN "2" DIGITS

CANDE now prints out all digits of a job queue insertion notice for queue numbers above 99.

P3500 CANDE - "CANDE DS" FOR SECURITY VIOLATION

A problem has been corrected that led to occasional termination of a CANDE worker stack for SECURITY VIOLATION.

P3501 CANDE - PACKNAME WITH LEADING DIGIT

Under some circumstances, a packname with a leading digit (e.g., 32PACK) would not be scanned by CANDE as a legitimate packname. This no longer occurs.

P3535 CANDE - BACKUPPROCESSOR FINDING END OF BLOCK

If a sequence range request to Backupprocessor began with the last record of a block, and if that record were the blank (second) line of a double space, CANDE could fault or produce incorrect output. Such conditions are now handled properly.

P3536 CANDE - "EOL" CHARACTER IN "DO" STATEMENT

The EOL character was not being recognized in a statement started with a DO verb unless the character was the last non-blank character on the input line. The EOL character is now treated as a "stopper" character for the scanner; all further information in that record is ignored.

P3571 CANDE - FOLDING LOWER CASE TOKENS

The criterion for splitting (folding) lines did not properly account for strings including lower-case characters. These tokens are now treated as single entities in the same manner as upper-case tokens.

P3574 CANDE - "SEG ARRAY" ERROR IN "DCWER"

A SEG ARRAY error could sometime occur in procedure DCWER if DCSYSTEMTABLES returned a long message. The message is now truncated to prevent the error.

P3613 CANDE - STATION VS. TERMINAL SETTINGS

CANDE was taking the TERMINAL setting for its TERMWIDTH value for a station event even though the STATION value might be different (and therefore override the TERMINAL value). This problem has been corrected.

P3691 CANDE - HANDLING LINE ERRORS

CANDE has a mechanism to detect frequent errors from stations or lines that are disconnected or malfunctioning; the offending station or line is eventually left not-ready.

This mechanism has been inoperative for line errors from non-switched lines, with the result that CANDE could be overloaded handling error messages. This problem has been corrected.

P3692 CANDE - "END JOB" ON "WFL" STATEMENT

CANDE no longer puts an "END JOB" on a WFL statement.

For those WFL statements containing an "AT <hn>", no "END JOB" can be included.

For those WFL statements to be executed on the local host, the WFL compiler appends the "END JOB".

P3785 CANDE - ALLOW "4-CHARACTER" VERBS

Previously, CANDE unconditionally sent only 3-character verbs. Now, CANDE calculates the length of the verb and passes that many characters to the CONTROLLER.

B6000 SERIES MARK 32
DOCUMENT CHANGES NOTES (D NOTES)

COBOL

D3003 COBOL - BINDING AND STATISTICS

It is no longer possible to bind programs with \$STATISTICS set. If STATISTICS is set, no bind info will be generated and the compiler will issue a warning message.

D3223 COBOL - "BCL" WARNINGS

The warning "BCL PROGRAMS ARE NOT PORTABLE TO EBCDIC MACHINES" is now given for a VALUE clause in a file declaration or with a SET statement which specifies VALUE(BCL) for the INTMODE attribute.

D3359 COBOL - MODIFICATIONS TO SUPPORT PORT FILES

Port files are declared and used much like Remote files in COBOL. (See Mark 32 GENERAL note D3650, "Implementation of Port Files", for a description of port files.) COBOL recognizes the value PORT for the KIND attribute and all of the additional attributes and values related to port files. Files are declared by assigning to PORT, and an optional Actual Key clause can be declared which can be used to reference subfiles:

```
SELECT <file-name> ASSIGN TO PORT
```

```
[ ACTUAL KEY IS <data-name-1> ]
```

```
[ FILE STATUS IS <data-name-2> ]
```

The ANSI74 construct FILE STATUS is a Burroughs extension for port files. The ANSI74 dollar option need not be set if the FILE STATUS is being used for a port file.

The general syntax for read and write statements on port files is as follows:

```
READ <file-name> RECORD [ WITH NO WAIT ] [ INTO <identifier> ]
```

```
[ INVALID KEY ... ]
```

```
WRITE <record-name> [ WITH NO WAIT ] [ FROM <identifier> ]
```

```
[ INVALID KEY ... ]
```

If no Actual Key clause is declared for the port file, a read or write statement performs a non-selective read or a broadcast write.

If an Actual Key is declared, the user is responsible for updating the Actual Key with the appropriate value of the subfile he wishes to reference. Read statements cause the Actual Key to be passed to the I/O system, where it is accessed and updated. If the Actual Key is zero, a non-selective read is performed, and the Actual Key is updated to indicate the subfile which is the source of the message. If the Actual Key is non-zero, a read from the specified subfile is performed. Write statements cause the Actual Key to be passed to the I/O system to indicate the desired subfile destination. A zero will cause a broadcast write.

If the Actual Key is less than zero or greater than the value of MAXSUBFILES, the READ or WRITE will result in a "subfile index out of range error. The status key will indicate this error with a value of "34".

A read statement normally causes the program to wait until a message is available, and a write statement normally causes the program to wait until a buffer is available to store the message. The possibility of this suspension can be prevented by using the NO WAIT clause on either a read or a write statement. The status key value "94" indicates that no message was available for the read. The status key value of "95" indicates that no buffer was available for the write.

The OPEN statement has been extended to allow the use of two features relevant to port files: opening the file if it is available and opening the file when it is offered by another process. These options may be invoked by specifying the word OFFER or AVAILABLE immediately before the file name in the Open statement. A port file may be opened with the options I-O, OFFER, or AVAILABLE. For example:

```
OPEN I-O PORTA.  
OPEN OFFER PORTB.  
OPEN AVAILABLE PORTC.
```


These options are mutually exclusive with the NO REWIND, REVERSED, and LOCK ACCESS options. The OFFER and AVAILABLE options are not allowed for any files other than those of KIND equal to PORT. A status key value of "00" indicates that the OPEN statement executed correctly. A status key value of "81" is returned if an error occurred during the execution of the OPEN statement. Note that for OPEN OFFER, a pending open is not an error, and, provided no error has occurred, a status key value of "00" will be returned, although the subfile has not been opened.

The CLOSE statement has also been extended to allow a feature relevant to port files. Currently a program attempting to close a file may be temporarily suspended. This suspension can be avoided by using the WITH NO WAIT option on the CLOSE statement. The syntax for the NO WAIT option is as follows:

```
CLOSE <file-name> [ WITH NO WAIT ].
```

The NO WAIT option is mutually exclusive with all other CLOSE options. It is allowed only for port files.

A status key value of "00" indicates that the CLOSE statement executed correctly. A status key value of "82" is returned if an error occurred during the execution of the CLOSE statement.

For OPEN and CLOSE statements, the user is responsible for updating the Actual Key to an appropriate value of the subfile to be opened or closed; an Actual Key value of zero indicates that the entire file is to be closed or opened as specified.

The event-valued file attributes relevant to port files are recognized in the WAIT statement of COBOL. The syntax for the file attribute identifier is also used for event-valued file attributes. The WAIT statement is available in COBOL in these two formats:

Format 1:

```
WAIT <arithmetic-expression>
```

Format 2:

```
WAIT [<arithmetic-expression>]
```

```
<event-valued-file-attribute> ...
```

```
[ GIVING <identifier-1>]
```

Format 1 is used for waiting the specified number of seconds or fraction thereof. Format 2 may be used for waiting for only one event or for some combination of seconds and events. <identifier-1> must be described as an elementary numeric integer data item without the symbol "P" in its PICTURE character-string. When GIVING is specified, the data item referenced by identifier-1 is set to the position in the list of events which terminated the WAIT statement. For example, if the second event in the list of events is found to be true, the data item referenced by identifier-1 is set to the value 2.

The following example shows a complex WAIT statement in COBOL which waits for 1 second or the CHANGEEVENTs of two port files, giving the ordinal number of the awakening operand in the variable N:

```
WAIT 1, PORTA (CHANGEEVENT),
      PORTB (CHANGEEVENT) GIVING N.
```

COBOL designates subfiles as parenthesized expressions following the file name, as shown in the following example:

```
PORTA (3,CHANGEEVENT)
```

D3449 COBOL - IGNORE NONFUNCTIONAL CHARACTERS

Add the following statement to the discussion of Continuation Indicator (column 7) on page 3-1 of the COBOL Reference Manual: (Form No. 5001464)

"e. Any other character which appears in column 7 is ignored."

D3473 COBOL - EDITING THE VALUE ZERO

It is possible for a data item to have a value of negative zero; in this case, when edited, zero will be treated as a negative number except in the case where all numeric character positions in the edit PICTURE string are represented by insertion characters (described on Page 6-67 of the COBOL Reference Manual, Form No. 5001464).

D3491 COBOL - "TIME" FUNCTION IN "DIVIDE" STATEMENT

The special registers TIME(n) and COMPILETIME(n), described on pages 2-13 and 2-14 of the COBOL Reference Manual, Form No. 5001464, may not be used in the DIVIDE statement.

D3503 COBOL - OPTION "4, MOVE" VERB

Wrap-around is permitted when using option 4 of the MOVE verb; therefore, the restriction that the source-bit or destination-bit minus the number of bits must not be less than -1 has been removed (as described in the COBOL Reference Manual, Form No. 5001464, Page 7-82). The number of bits to be moved, however, must still be less than 48.

D3504 COBOL - VALUE CLAUSE FOR "COMP-4,COMP-5" ITEMS

If the usage of an item is COMP-4 or COMP-5, all literals in the VALUE clause must be floating point literals (see Page 6-95 of the COBOL Reference Manual, Form No. 5001464).

D3505 COBOL - "OPEN EXTEND" ON VARIABLE LENGTH FILE

The EXTEND option of the OPEN statement cannot be used with variable length files.

D3510 COBOL - LIMIT ON "SIZE" CLAUSE

The SIZE clause of a data-item is limited by a maximum value of 65535. Exceeding this value will give a compiler error. This is true even for fixed-length group items where the SIZE clause is used for documentation purposes only.

D3517 COBOL - PASSING STRINGS FROM "WFL" TO "COBOL"

When WFL passes a string as a parameter, it uses an exact size word array to pass the string. If the declaration of the parameter in the COBOL program is greater than the size of the string, a run-time error will occur.

The run-time errors did not occur on the Mark 30 system software release because WFL on that release copied the string into a 256-character array and then passed the character array to the COBOL program, thus erroneously overallocating space for the string. This error had gone unnoticed; now, it is implemented as intended.

D3520 COBOL - "IPC, USE AS EXTERNAL PROCEDURE" STATEMENT

A program with a "USE AS EXTERNAL PROCEDURE" declarative is not marked IPC-capable.

Effective with the Mark 32 system software release, the IPC-capable bit is no longer necessary.

D3521 COBOL - ERROR LIMIT OF "150"

Under the LIMIT paragraph on Page 13-16 of the COBOL Reference Manual (Form No. 5001464), the fourth sentence should read as follows:

"A limit of 150 is assumed if no LIMIT value is specified except when compilation is called for by CANDE."

D3523 COBOL - UNITS=CHARACTERS

The MCP for optimization purposes uses UNITS=WORDS for COBOL files when

(MAXRECSIZE MOD 6)=0 and UNITS=CHARACTERS

Therefore, setting UNITS to CHARACTERS will have no effect.

D3524 COBOL - REFERENCING TWO OR THREE DIMENSIONAL ARRAYS

Page 6-18 of the COBOL Reference Manual (Form No. 5001464) should contain the following statement:

"When a two- or three-dimensional array is declared using the INDEXED BY <index-name-1>, . . . option, the array may not be referenced using data-names."

D3541 COBOL - CLOSE ON A DISK FILE

On Page 7-47 of the COBOL Reference Manual (Form No. 5001464), under item number 3 for close file-name with release, the following sentence:

"For an old file, same as CLOSE file-name."

should be replaced by the following sentence:

"For an old file, the output areas are released, and that particular instance of the file is no longer logically associated with the program."

D3570 COBOL - "RECORD AREA"

The following should be included in the COBOL Reference Manual (Form No. 5001464), Page 5-14, as the second sentence:

"The length of the RECORD AREA specified to be used for direct I/O must be an even number of characters."

D3648 COBOL - "TB" RESERVED WORD

The table of reserved words on Page A-13 of the COBOL Reference Manual (Form No. 5001464) should include the reserved word TB under the following system options: B6700, ANSI74.

D3658 COBOL - PARAMETER PASSING

Add the following paragraph to the COBOL Language Reference Manual (Form No. 5001464) as Item 9 on Page 7-42:

The parameters in the USING clause may not be declared with a REDEFINE clause."

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

COBOL

P2701 COBOL - "INVALID INDEX" IN "COBOL" COMPILER

The handling of error recovery in the RELEASE statement was causing an INVALID INDEX in the COBOL compiler. This has been corrected.

P2702 COBOL - "USE" ROUTINE NOT INVOKED

A USE AFTER STANDARD ERROR PROCEDURE was not invoked when parity errors occurred on a tape file with variable length records. This has been corrected.

P2703 COBOL - "VALUE(TERMINATED)"

The compiler no longer generates a one (1) instead of a negative one (-1) when the attribute mnemonic VALUE(TERMINATED) is referenced.

P2704 COBOL - "WAIT" STATEMENT

The compiler incorrectly generated two items of binding information (PCWs for TIMETUNNEL and SUPERWAIT) for the following statement:

WAIT 60, EVNT GIVING R

where EVNT is an event variable. The BINDER changed the PCW for the WAIT statement twice and generated invalid addressing code. This has been corrected.

P2705 COBOL - "IF" STATEMENT GENERATES BAD CODE

The COBOL compiler generated incorrect code for the following statement:

IF <real literal> <relational op> <arithmetic expression>
STATEMENT1 ELSE STATEMENT2

This has been corrected.

P2706 COBOL - RESULTS OF EXPONENTIATION IMPROVED

The results of exponentiation sometimes gave less than full decimal precision when both operands to the exponentiation were single precision (less than 12 decimal places).

This has been corrected.

P2707 COBOL - BINDINFO FOR "77 COMP" GLOBAL ITEM

If the ANSI74 option was set in a COBOL program, bind information for a 77 level COMP item was not generated. This has been corrected.

P2708 COBOL - INVALID SYNTAX FOR FILE ATTRIBUTES

The COBOL compiler did not generate a syntax error message for invalid syntax for file attributes with parameters. This has been corrected.

P2727 COBOL - COMPILATION SUMMARY

At the end of compilation, the message "COMPILED ON THE B6700 (B7700) FOR THE B6700 (B7700)" was printed. This message would be incorrect if the compilation had been done on a B6800 or B7800. A more accurate message will now be printed: "COMPILED ON THE (B6700/B6800/B7700/B7800) FOR THE B6000 (B7000) SERIES", where the appropriate options will be printed.

P2728 COBOL - "INVALID INDEX"

Bad syntax error recovery in the handling of subscripts was causing an INVALID INDEX in the COBOL compiler. This problem has been corrected.

P2729 COBOL - "NEWSEQERR" \$ OPTION

The \$ option NEWSEQERR was accepting equal record numbers without flagging an error, thereby locking the NEWTAPE file. This problem has been corrected; the COBOL compiler now gives a warning message, and the NEWTAPE file is not locked.

P2732 COBOL - INTRINSIC INFORMATION IN GLOBAL DIRECTORY

BINDER will now correctly bind intrinsics used by subprograms to those intrinsics used in the host program.

P2813 COBOL - "OPEN O-I FILE1 I-O FILE2"

The compiler will now correctly handle an OPEN O-I FILE1 I-O FILE2 statement. Previously, FILE2 would be opened as output.

P2815 COBOL - "INVALID OP" IN "IF" STATEMENT

The handling of exponentiation in an IF statement has been corrected.

P2834 COBOL - INTERACTION OF "OPEN" STATEMENT

For a disk file with a non-zero areaseize which had been opened only for output during a program, the value of the attribute RESIDENT would be TRUE before the OPEN had been executed although there was no file. This has been corrected; the value of RESIDENT will be FALSE until the file is opened.

P2835 COBOL - "STOP RUN"

A STOP RUN statement in a COBOL library was causing a "BAD GOTO". This has been corrected (with support of the MCP).

P2932 COBOL - "\$" OPTIONS "SEQERR, NEWSEQERR, SEQCHECK"

If the \$ options SEQERR or NEWSEQERR in COBOL or the \$ option SEQCHECK in COBOL74 are set and the sequence numbers are not in ascending order, a sequencing warning will be produced.

The warning was also produced if the previous sequence number was comprised of spaces and the current sequence number is also spaces. This has been corrected.

P3054 COBOL - CALL USER INTRINSIC

A call to an untyped user intrinsic generated a spurious delete on the Mark 31 release, which has been corrected.

P3055 COBOL - "INVALID INDEX" IN REPORT WRITER

An INVALID INDEX which occurred in the COBOL compiler when compiling the report writer section of very large COBOL programs has been corrected.

P3080 COBOL - INDEXED FILE WITH INVALID KEY BRANCH

For a WRITE statement with an invalid key clause of an indexed file, an EOF NO LABEL error would occur and discontinue execution if the file limit were exceeded, due to duplicate keys causing the invalid key clause to be taken; however, trying to write past the end of the file would not cause the invalid key clause to be taken. Now, both the end of the file and duplicate keys will cause the invalid key branch to be taken.

P3081 COBOL - "RERUN" CLAUSE

Syntax checking in the RERUN clause has been corrected.

P3205 COBOL - CALLS ON UNTYPED USER INTRINSICS

The compiler no longer generates erroneous code for a CALL of a user intrinsic in which the parameters were passed by value.

P3262 COBOL - "LIBRARY CALL" WITHIN "IF"

The statement after a LIBRARY CALL within an IF statement is no longer ignored by the compiler.

P3274 COBOL - EQUAL COMPARISONS

If a subscripted item were compared for equality against another item and their lengths were different, the compiler emitted the wrong operator, causing the comparison to fail. This problem has been corrected.

P3275 COBOL - GROUP COMPUTATIONAL MOVES

The code generated for a group move of computational items no longer references an invalid descriptor.

P3276 COBOL - ERRONEOUS SYNTAX ERROR

When a program was compiled at a lex level > 2, certain MOVE statements would cause the compiler to terminate with the error message "COMPILER ERROR". MOVE statements that translate data from one form of internal representation to another would cause the compiler to issue this erroneous message. This problem has been corrected.

P3294 COBOL - SYNTAX CHECKING WITH "NEXT SENTENCE" CLAUSE

With the OPTIMIZE dollar option set, the compiler would not catch all syntax errors for nested IF statements that used the NEXT SENTENCE clause. This problem has been corrected.

P3295 COBOL - SYNTAX ERROR IN "IF" STATEMENT

Parentheses around a data name in a conditional statement would cause the compiler to issue an erroneous syntax error. This problem has been corrected.

P3302 COBOL - "OBJECT-COMPUTER" CLAUSE SYNTAX

The compiler will now correctly check the syntax of the OBJECT-COMPUTER clause. Previously, unit types other than disk in the DISK SIZE clause (e.g., DISKPACK) would cause the default disk size to be applied to the sort.

P3303 COBOL - "ANSI 74" DEFAULT LINE SPACING

The default line spacing for the ANSI74 dollar option has been corrected. The incorrect default spacing of "BEFORE ADVANCING 1" was being generated instead of "AFTER ADVANCING 1" when the ANSI74 option had been set when the COBOL compiler itself was compiled and there were no subsequent \$ option cards in the program being compiled.

P3304 COBOL - "LINE NUMBER" CLAUSE SYNTAX

The compiler will now correctly handle a LINE NUMBER clause in a Report Group description. Previously, a duplicate LINE NUMBER clause subordinate to a Report Group description would go undetected.

P3305 COBOL - "MERGE" STATEMENT SYNTAX

The COBOL compiler now flags a MERGE statement containing either a DISK SIZE clause or a MEMORY SIZE clause as invalid syntax. Previously, the compiler did not flag these cases as invalid syntax; instead, it generated incorrect code which could cause the system to hang.

P3306 COBOL - LONG CONDITIONAL EXPRESSIONS

While evaluating a condition containing more than 99 logical operators, the COBOL compiler would get a SEG ARRAY error. This no longer occurs.

P3307 COBOL - "KEYSPERENTRY" GREATER THAN "63"

The compiler will now correctly check the syntax of a VALUE OF ID clause, accessing the KEYSPERENTRY attribute. Previously, a value greater than 63 would cause a descending sequence of records to be expected during creation of a file.

P3427 COBOL - "INVALID INDEX"

An expression whose subject is an alpha-numeric operand and whose object is either a numeric literal or an alpha-numeric literal could cause an INVALID INDEX during the evaluation if the usage of the object was being set incorrectly.

Example:

```
IF AAA (1) = SPACE ALL "*" 1 2 OR 3
```

where

```
01 A
   03 AA OCCURS 10.
   05 AAA PIC X.
```

This problem has been corrected.

P3465 COBOL - MAXIMUM NUMBER OF LIBRARIES EXCEEDED

The compiler no longer terminates with an INVALID INDEX when the maximum number of libraries are exceeded.

P3466 COBOL - RESERVED WORDS SYNTAXED IN "WRITE" STATEMENTS

The COBOL compiler no longer fails with an INVALID INDEX when reserved words are incorrectly used in WRITE statements.

P3467 COBOL - "INVALID INDEX" WITH "CP CALL"

The COBOL compiler would fail with an INVALID INDEX if the control-point-identifier of a CALL statement were a reserved word. This no longer occurs.

P3502 COBOL - MISALIGNMENT OF "COMP-2" SYNC VALUES

COMP-2 sync values are now properly aligned against a word boundary.

P3503 COBOL - "SEGMENT" CLAUSE, "01" RECORD

The following problems have been corrected:

1. Use of the SEGMENT clause caused the compiler to issue a COMPILER ERROR message.
2. References to an 01 record that had an OCCURS clause caused the compiler to generate bad code.

P3525 COBOL - TIMESTAMP DIFFERENCES

Previously, the date and time of a compiler listing and the date and time of the code file could differ for a program which took a large amount of compilation time. Now, the compiler listing and the code file listing will agree. The time and date used are those from the beginning of the compilation.

P3538 COBOL - "ISAM" "CLOSE" OPTIONS

Under the ANSI74 option of COBOL, only three CLOSE actions are available for an ISAM file: CLOSE, CLOSE WITH PURGE and CLOSE WITH LOCK. For any other CLOSE options, an error is given.

P3559 COBOL - "MONITOR," WRITE TO SAME FILE

The compiler now correctly handles a MONITOR statement using a print file selected in a user program. Previously, the program listing was incorrect.

P3561 COBOL - "ISAM" EXTERNAL FILE NAMES

The compiler will now correctly handle ISAM external file names when a data name is used in a VALUE OF ID clause. Previously, the system would ask for the file using the internal file name.

P3562 COBOL - "COPY" WITH BAD FILE TITLE

The compiler will now correctly handle a COPY statement with an incorrect file name. Previously, the file in the previous COPY statement would be copied again.

P3563 COBOL - OPERAND LEFT ON TOP OF STACK

Under certain circumstances, a call on an intrinsic in an arithmetic statement would cause the compiler to generate code that left garbage on the top of the stack. This no longer occurs.

P3581 COBOL - "COMP-2" NUMERIC TEST

A hex "F" will now be considered a valid sign character in a numeric class test of a COMP-2 item. Previously, the compiler treated hex "C"s and "D"s as the only valid signs.

P3679 COBOL - USER INTRINSICS AT LEVELS "> 2"

Calls on a user intrinsic when the code was compiled at a level greater than 2 would sometimes cause a compiler error. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

COBOL74

D3010 COBOL74 - FILE DESCRIPTION ENTRY

Non-numeric literals used at the end of VALUE OF clauses will no longer require a period at the end.

D3222 COBOL74 - CLOSE "WITH LOCK"

The 1974 ANSI standard requires that files which are closed "WITH LOCK" cannot subsequently be reopened by the program. Because the enforcement of this restriction was anticipated, the CLOSE option "WITH SAVE" has been provided as a Burroughs COBOL74 extension intended for making a mass-storage file permanent while also allowing it to be reopened by the program. The "WITH LOCK" option continues to make a mass-storage file permanent and make tape units "NOT READY", and can be used as long as the logical file is not reopened.

D3224 COBOL74 - "BCL" WARNINGS

The warning "BCL PROGRAMS ARE NOT PORTABLE TO EBCDIC MACHINES" is now given for a VALUE clause in a file declaration or with a CHANGE statement which specifies VALUE(BCL) for the INTMODE attribute.

D3358 COBOL74 - MODIFICATIONS TO SUPPORT PORT FILES

Port files are declared and used much like Remote files in COBOL74. (See Mark 32 GENERAL note D3650, "Implementation of Port Files", for a description of port files.) COBOL74 recognizes the value PORT for the KIND attribute and all of the additional attributes and values related to port files. Files are declared by assigning to PORT, and an optional Actual Key clause can be declared which can be used to reference subfiles:

```
SELECT <file-name> ASSIGN TO PORT
-----
[ ACTUAL KEY IS <data-name-1> ]
-----
[ FILE STATUS IS <data-name-2> ]
-----
```

The general syntax for read and write statements on port files is as follows:

```
READ <file-name> RECORD [ WITH NO WAIT ] [ INTO <identifier> ]
-----
[ INVALID KEY ... ]
-----

WRITE <record-name> [ WITH NO WAIT ] [ FROM <identifier> ]
-----
[ INVALID KEY ... ]
-----
```

If no Actual Key clause is declared for the port file, a read or write statement performs a non-selective read or a broadcast write.

If an Actual Key is declared, the user is responsible for updating the Actual Key with the appropriate value of the subfile he wishes to reference. Read statements cause the Actual Key to be passed to the I/O system, where it is accessed and updated. If the Actual Key is zero, a non-selective read is performed, and the Actual Key is updated to indicate the subfile which is the source of the message. If the Actual Key is non-zero, a read from the specified subfile is performed. Write statements cause the Actual Key to be passed to the I/O system to indicate the desired subfile destination. A zero will cause a broadcast write.

If the Actual Key is less than zero or greater than the value of MAXSUBFILES, the READ or WRITE will result in a "subfile index out of range error. The status key will indicate this error with a value of "34".

A read statement normally causes the program to wait until a message is available, and a write statement normally causes the program to wait until a buffer is available to store the message. The possibility of this suspension can be prevented by using the NO WAIT clause on either a read or a write statement. The status key value "94" indicates that no message was available for the read. The status key value of "95" indicates that no buffer was available for the write.

The OPEN statement has been extended to allow the use of two features relevant to port files: opening the file if it is available and opening the file when it is offered by another process. These options may be invoked by specifying the word OFFER or AVAILABLE immediately before the file name in the Open statement. A port file may be opened with the options I-O, OFFER, or AVAILABLE. For example:

OPEN I-O PORTA.
 OPEN OFFER PORTB.
 OPEN AVAILABLE PORTC.

These options are mutually exclusive with the NO REWIND, REVERSED, and LOCK ACCESS options. The OFFER and AVAILABLE options are not allowed for any files other than those of KIND equal to PORT. A status key value of "00" indicates that the OPEN statement executed correctly. A status key value of "81" is returned if an error occurred during the execution of the OPEN statement. Note that for OPEN OFFER, a pending open is not an error, and, provided no error has occurred, a status key value of "00" will be returned, although the subfile has not been opened.

The CLOSE statement has also been extended to allow a feature relevant to port files. Currently a program attempting to close a file may be temporarily suspended. This suspension can be avoided by using the WITH NO WAIT option on the CLOSE statement. The syntax for the NO WAIT option is as follows:

```
CLOSE <file-name> [ WITH NO WAIT ].
```

The NO WAIT option is mutually exclusive with all other CLOSE options. It is allowed only for port files.

A status key value of "00" indicates that the CLOSE statement executed correctly. A status key value of "82" is returned if an error occurred during the execution of the CLOSE statement.

For OPEN and CLOSE statements, the user is responsible for updating the Actual Key to an appropriate value of the subfile to be opened or closed; an Actual Key value of zero indicates that the entire file is to be closed or opened as specified.

The event-valued file attributes relevant to port files are recognized in the WAIT statement of COBOL74. The syntax for the file attribute identifier is also used for event-valued file attributes. The WAIT statement is available in COBOL74 in these two formats:

Format 1:

```
WAIT <arithmetic-expression>
```

Format 2:

```
WAIT [<arithmetic-expression>]
```

```
<event-valued-file-attribute> ...
```

```
[ GIVING <identifier-1> ]
```

Format 1 is used for waiting the specified number of seconds or fraction thereof. Format 2 may be used for waiting for only one event or for some combination of seconds and events. <identifier-1> must be described as an elementary numeric integer data item without the symbol "P" in its PICTURE character-string. When GIVING is specified, the data item referenced by identifier-1 is set to the position in the list of events which terminated the WAIT statement. For example, if the second event in the list of events is found to be true, the data item referenced by identifier-1 is set to the value 2.

The following example shows a complex WAIT statement in COBOL74 which waits for 1 second or the CHANGEEVENTs of two port files, giving the ordinal number of the awakening operand in the variable N:

```
WAIT 1, ATTRIBUTE CHANGEVENT OF PORTA,  

  ATTRIBUTE CHANGEVENT OF PORTB GIVING N.
```

COBOL74 designates subfiles as parenthesized expressions following the file name, as shown in the following example:

```
ATTRIBUTE CHANGEVENT OF PORTA(3)
```

D3505 COBOL74 - "OPEN EXTEND" ON VARIABLE LENGTH FILE

The EXTEND option of the OPEN statement cannot be used with variable length files.

D3531 COBOL74 - ERROR LIMIT OF "150"

A limit of 150 is assumed if no LIMIT value is specified except when compilation is called for by CANDE.

D3574 COBOL74 - DATA COMM INTERFACE LIBRARY

The ANSI74 COBOL Communications Module provides the COBOL74 user an ability to communicate through a 'message control system' with communication devices. (The ANSI74 'message control system' concept is not to be confused with the Burroughs Message Control System concept.) Messages are communicated in a device independent symbolic manner. Specific devices and system structures are known to the COBOL74 user only symbolically. This generality between the compiled program and the particular system is achieved via an interface called the Data Communications Interface (DCI). The DCI adapts the general abilities in the compiler to the specific device needs of an application.

The DCI is a library to which the compiler builds references whenever a COBOL74 program uses the verbs 'Disable', 'Enable', 'Receive', 'Send', or 'Accept <CD-Name>'. This library reference is built with the internal name 'DCILIBRARY' and an entrypoint name 'DCIENTRYPPOINT'. This entrypoint is an untyped procedure with eight parameters:

- 1) Integer, by value : The DCI function having values -

1	Accept
2	Disable
3	Enable
4	Receive
5	Send

- 2) Ebcidic Array (unindexed descriptor) : The CD having the COBOL74 description -

For Accept, Disable Input, Enable Input, or Receive:

01	CD-ARRAY.	
02	QUEUE-NAME	PIC X(12).
02	SUB-QUEUE-1-NAME	PIC X(12).
02	SUB-QUEUE-2-NAME	PIC X(12).
02	SUB-QUEUE-3-NAME	PIC X(12).
02	MESSAGE-DATE-NAME	PIC 9(6).
02	MESSAGE-TIME-NAME	PIC 9(8).
02	SOURCE-NAME	PIC X(12).
02	TEXT-LENGTH-NAME	PIC 9(4).
02	END-KEY-NAME	PIC X.
02	STATUS-KEY-NAME	PIC XX.
02	MESSAGE-COUNT-NAME	PIC 9(6).

For Disable Output, Enable Output, or Send :

01	CD-ARRAY.	
02	DESTINATION-COUNT-NAME	PIC 9(4).
02	TEXT-LENGTH-NAME	PIC 9(4).
02	STATUS-KEY-NAME	PIC XX.
02	DESTINATION-TABLE-NAME	OCCURS <int> TIMES.
03	ERROR-KEY-NAME	PIC X.
03	DESTINATION-NAME	PIC X(12).

- 3) Integer, by value : Occurrences of the Destination Table (The number of occurrences of the 'DESTINATION-TABLE-NAME' record of the CD array in parameter number two. For Accept, Disable input, Enable input, or Receive the Input CD would have a zero value meaning no occurrences. For Disable output, Enable output, and Send the Output CD would have an integer value greater than zero corresponding to the length of the CD array (i.e. (THE-CD-ARRAY-LENGTH - 10) DIV 13 = this parameter).
- 4) Ebcidic Array (unindexed descriptor) : The Message Array or Key password.
- 5) Integer, by value : Length of the Message Array or Key password (The length in characters of the array in parameter number three. The length of the information contained in this array is maintained by the user in the CD TEXT-LENGTH-NAME.)
- 6) Integer, by value : The End Indicator or I/O type -

For Send or Receive, the End Indicator would be:

1	ESI (end of segment/ Receive segment).
2	EMI (end of message/ Receive message).
3	EGI (end of group).

And for Disable or Enable, the I/O type:

11	Input Terminal.
12	Input.

B6000 SERIES MARK 32

13 Output.

- 7) Integer, by reference : The Advancing type or Suspend No Data Indicator -

For Send, the Advancing type:

- 1 After Lines.
- 2 Before Lines.
- 3 After Channel.
- 4 Before Channel.

And for Receive, the Suspend No Data Indicator:

<value>
Suspend program amount if No Data.

If this parameter is zero, then a No Data clause was supplied with the Receive statement. Otherwise the value supplied is the time in seconds to wait. If no data is available, the DCI Library would return a boolean truth value (i.e. an odd value) in parameter seven (passed by reference).

- 8) Integer, by value : Advancing Value for Send, otherwise zero.

The second parameter, the CD array, allows communication between the COBOL74 program and the DCI library. The COBOL74 program is responsible for the maintenance of the CD array passed by the compiler. The DCI is responsible for updating information in the CD array it receives.

The information in the CD should be updated in coordination with the COBOL74 program according to the rules as required by the ANSI74 Standard on pages XIII 1-23, sections 2.2.4, 3.1.4, 3.2.4, 3.3.4, 3.4.4, and 3.5.4. These rules specify details concerning the updating of items in the CD array such as setting values of the Status Key, collection of messages, handling queues, and password validity checking.

The DCI library can be written in any language including COBOL, ALGOL or DCALGOL, allowing access to disk files, remote files, or port files. The symbolic queues, selection algorithms and source and destinations as set out in the ANSI74 COBOL standard can be tailored to the particular application using the DCI library. A typical DCI library written in COBOL would declare parameters as follows:

```

$SET ANSI74
IDENTIFICATION DIVISION.
PROGRAM-ID. DCIENTRYPPOINT.
ENVIRONMENT DIVISION.
DATA DIVISION.
LINKAGE SECTION.
77 DCI-FUNCTION                                COMP PIC 9.
01 THE-CD                                     PIC X(244).
01 INPUT-CD                                  REDEFINES THE-CD.
    02 SYMBOLIC-QUEUE                        PIC X(12).
    02 SYMBOLIC-SUB-QUEUE-1                  PIC X(12).
    02 SYMBOLIC-SUB-QUEUE-2                  PIC X(12).
    02 SYMBOLIC-SUB-QUEUE-3                  PIC X(12).
    02 MESSAGE-DATE                          PIC 9(6).
    02 MESSAGE-TIME                          PIC 9(8).
    02 SYMBOLIC-SOURCE                       PIC X(12).
    02 TEXT-LENGTH                           PIC 9(4).
    02 END-KEY                               PIC X.
    02 STATUS-KEY                            PIC XX.
    02 MESSAGE-COUNT                          PIC 9(6).
01 OUTPUT-CD                                 REDEFINES THE-CD.
    02 DESTINATION-COUNT                      PIC 9(4).
    02 TEXT-LENGTH                           PIC 9(4).
    02 STATUS-KEY                            PIC XX.
    02 DESTINATION-TABLE                     OCCURS 1 TO 18 DEPENDING
                                                ON THE-CD-OCCURRENCES.
        03 ERROR-KEY                         PIC X.
        03 SYMBOLIC-DESTINATION              PIC X(12).
77 THE-CD-OCCURRENCES                        COMP PIC 999.
01 THE-MESSAGE.
    03 THE-MESSAGE-SUB                       PIC X
                                                OCCURS 1 TO 256 DEPENDING
                                                ON THE-MESSAGE-LENGTH.
77 THE-MESSAGE-LENGTH                        COMP PIC 999.
77 IO-OR-END-INDICATOR                       COMP PIC 99.
77 NO-DATA-OR-ADVANCING-TYPE                 COMP PIC 99 BY REFERENCE.
77 ADVANCING-VALUE                           COMP PIC 99.
PROCEDURE DIVISION
    USING DCI-FUNCTION,
           THE-CD,

```

THE-CD-LENGTH,
THE-MESSAGE,
THE-MESSAGE-LENGTH,
IO-OR-END-INDICATOR,
NO-DATA-OR-ADVANCING-TYPE,
ADVANCING-VALUE.

MAIN SECTION.

Pl.

PERFORM WHATEVER-FUNCTION.
EXIT PROGRAM.

D3597 COBOL74 - FILE HANDLING DIFFERENCES

The compiler uniquely identifies file descriptions to distinguish COBOL74 files from COBOL68 files. With this distinction, the I/O subsystem no longer coerces character-mode files to word-mode files for COBOL74, although that action will continue for COBOL68.

All calls on the I/O subsystem will also distinguish COBOL74 from COBOL68.

D3659 COBOL74 - "D" LINES, "FREE" OPTION

The compiler control option FREE (which is set automatically when compiling through CANDE) allows many of the margin restrictions of COBOL to be relaxed. The FREE option must be reset when using a debugging line in the program.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

COBOL74

P2701 COBOL74 - "INVALID INDEX" IN "COBOL" COMPILER

The handling of error recovery in the RELEASE statement was causing an INVALID INDEX in the COBOL compiler. This has been corrected.

P2703 COBOL74 - "VALUE(TERMINATED)"

The compiler no longer generates a one (1) instead of a negative one (-1) when the attribute mnemonic VALUE(TERMINATED) is referenced.

P2705 COBOL74 - "IF" STATEMENT GENERATES BAD CODE

The COBOL compiler generated incorrect code for the following statement:

```
IF <real literal> <relational op> <arithmetic expression>  
STATEMENT1 ELSE STATEMENT2
```

This has been corrected.

P2706 COBOL74 - RESULTS OF EXPONENTIATION IMPROVED

The results of exponentiation sometimes gave less than full decimal precision when both operands to the exponentiation were single precision (less than 12 decimal places).

This has been corrected.

P2727 COBOL74 - COMPILATION SUMMARY

At the end of compilation, the message "COMPILED ON THE B6700 (B7700) FOR THE B6700 (B7700)" was printed. This message would be incorrect if the compilation had been done on a B6800 or B7800. A more accurate message will now be printed: "COMPILED ON THE (B6700/B6800/B7700/B7800) FOR THE B6000 (B7000) SERIES", where the appropriate options will be printed.

P2728 COBOL74 - "INVALID INDEX"

Bad syntax error recovery in the handling of subscripts was causing an INVALID INDEX in the COBOL compiler. This problem has been corrected.

P2800 COBOL74 - "SERIALNO" ATTRIBUTE

In the VALUE OF clause of an FD statement, SERIALNO may now be set to a non-numeric literal of length six.

P2801 COBOL74 - NON-NUMERIC ATTRIBUTES

The period which was necessary at the end of the literal in the CHANGE ATTRIBUTE statement for non-numeric attributes is now optional.

P2815 COBOL74 - "INVALID OP" IN "IF" STATEMENT

The handling of exponentiation in an IF statement has been corrected.

P2819 COBOL74 - CLOSE FOR MULTI-FILE TAPES

Previously, if a CLOSE statement were executed for an output file on a multi-file tape, the file was closed and the tape was rewound; hence, subsequent output files, if any, on the multi-file tape could not be written. Now, the file will be closed, but the tape will not be rewound.

P2835 COBOL74 - "STOP RUN"

A STOP RUN statement in a COBOL library was causing a "BAD GOTO". This has been corrected (with support of the MCP).

P2851 COBOL74 - "\$OPTIMIZE" COMPILER ERROR

\$SET OPTIMIZE was causing a compiler error at the end of the PROCEDURE DIVISION. This has been corrected.

P2931 COBOL74 - "SIGN" CLAUSE FOR COMPUTATIONAL ITEM

The SIGN clause has been implemented for computational items.

P2932 COBOL74 - "\$" OPTIONS "SEQERR, NEWSEQERR, SEQCHECK"

If the \$ options SEQERR or NEWSEQERR in COBOL or the \$ option SEQCHECK in COBOL74 are set and the sequence numbers are not in ascending order, a sequencing warning will be produced.

The warning was also produced if the previous sequence number was comprised of spaces and the current sequence number is also spaces. This has been corrected.

P2937 COBOL74 - FILE ATTRIBUTES

Attribute handling in COBOL74 was incorrect: statements using attributes of the ARITHMETICTYPE, BOOLEANATYPE, TASKTYPE or EVENTTYPE did not compile. This problem has been corrected.

P3055 COBOL74 - "INVALID INDEX" IN REPORT WRITER

An INVALID INDEX which occurred in the COBOL compiler when compiling the report writer section of very large COBOL programs has been corrected.

P3060 COBOL74 - "WRITE AFTER ADVANCING PAGE" STATEMENT

A WRITE AFTER ADVANCING PAGE statement which was associated with a file having a lineage clause resulted in one extra blank line being printed. This problem has been remedied.

P3070 COBOL74 - "SEQCHECK" \$ OPTION

The \$ option SEQCHECK was accepting equal record numbers without flagging an error, thereby locking the NEWSOURCE file. This problem has been corrected; the COBOL74 compiler now gives a warning message, and the NEWSOURCE file is not locked.

P3081 COBOL74 - "RERUN" CLAUSE

Syntax checking in the RERUN clause has been corrected.

P3084 COBOL74 - DEBUG LINE VALUES

The value of the following debug-lines in the special register debug-item has been corrected:

START PROGRAM	- identifies the first statement of the first non-declarative procedure
FALL THROUGH	- identifies the previous statement
SORT	- identifies the beginning of the SORT statement

Also, the scanning of debugging lines when the dollar option FREE is reset has been corrected.

P3086 COBOL74 - "WRITE" STATEMENT WITH "FOOTING" EQUAL TO "1"

A WRITE statement which was associated with a file with a lineage clause where the FOOTING was set to 1 resulted in pagesize attribute errors. This is now handled correctly.

P3274 COBOL74 - EQUAL COMPARISONS

If a subscripted item were compared for equality against another item and their lengths were different, the compiler emitted the wrong operator, causing the comparison to fail. This problem has been corrected.

P3294 COBOL74 - SYNTAX CHECKING WITH "NEXT SENTENCE" CLAUSE

With the OPTIMIZE dollar option set, the compiler would not catch all syntax errors for nested IF statements that used the NEXT SENTENCE clause. This problem has been corrected.

P3295 COBOL74 - SYNTAX ERROR IN "IF" STATEMENT

Parentheses around a data name in a conditional statement would cause the compiler to issue an erroneous syntax error. This problem has been corrected.

P3308 COBOL74 - LINAGE - FOOTING VALUE OF ONE

A WRITE statement to a file with a LINAGE clause and a footing value of one is now handled according to ANSI 74 standards.

P3395 COBOL74 - "BDMS" FEDERAL LEVEL WARNING

The following Federal warning is issued when a BDMS construct is used:

"BURROUGHS EXTENSION EXCEEDS U.S. HIGH LEVEL"

P3465 COBOL74 - MAXIMUM NUMBER OF LIBRARIES EXCEEDED

The compiler no longer terminates with an INVALID INDEX when the maximum number of libraries are exceeded.

P3525 COBOL74 - TIMESTAMP DIFFERENCES

Previously, the date and time of a compiler listing and the date and time of the code file could differ for a program which took a large amount of compilation time. Now, the compiler listing and the code file listing will agree. The time and date used are those from the beginning of the compilation.

P3564 COBOL74 - "COMPUTATIONAL" NUMERIC TEST

A hex "F" will now be considered a valid sign character in a numeric class test of a COMPUTATIONAL item. Previously, the compiler treated hex "C"s and hex "D"s as the only valid signs.

P3826 COBOL74 - LIBRARY PSEUDO TEXT REPLACEMENT

A pseudo-text match failed in the case where the library text contained a debugging D line. This problem has been corrected.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

COMPARE

P3614 COMPARE - "ARRAY TOO LARGE" ERROR

COMPARE is no longer DSed with the error message "ARRAY TOO LARGE" if the MAXRECSIZE of the files to be compared is a large number of characters.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

CONFIGURATOR

D3406 CONFIGURATOR - SOFT CONFIGURATION

The Mark 32 release supports soft reconfiguration of both B6800 and B6900 systems (see Mark 31 notes D2861 and D2867 for an explanation of B6800 soft reconfiguration). It has been necessary to expand the definition of a GROUP to include the B6900 I/O subsystem configuration.

B6800 configuration file group descriptions contain the following major sections: processor, global memory, operations, and peripherals. The B6900 group description also includes an I/O section, which describes I/O subsystem connectivity on a base by base basis. Syntax for the base section is as follows:

GROUP Statement:

```
-- BEGIN -- GROUP -- <group id> -- <processor section> ----->
>----->
| -<global memory section>- | | -<operations section>- |
>-----> END GROUP-- ; --|
| -<I/O section>- | | -<peripheral section>- |
```

<I/O section>

```
-- I/O: ----- <base list> -----|
```

<base list>

```
| <-- /319\ -----|
----- BASE -- <base id> -- <host list> -----|
```

<base id>

```
--- <base bus address> --- / --- <base bus number> --- / ----->
----- <base extension number> -----|
```

SEMANTICS:

The base id is a 16-bit field strap and is composed of three fields: maintenance bus address (8 bits), maintenance bus number (4 bits), and the base extension number (4 bits). The first two fields make up the <unique base id> and, as the name implies, must generate a unique id within any system. The 4-bit base extension number is the upper 4 bits of a unit number. Currently, this field must be zero.

<host list>

```
| <-- /6\ -----|
----- <independent host> --- <dlp list> -- ; -----|
| -- <dependent host> ---|
```

<independent host>

```
----- HOST ----- <host number> -----|
```

SEMANTICS:

An independent host is a processor which has an MLI connection into a base. The host number must match the processor id.

<dependent host>

--- DEPENDENT HOST --- <host number> --- DLPID -- <dlp id> ----- |

SEMANTICS:

A dependent host is a DLP which has an MLI connection into a base. With the cooperation of the dependent host, an independent host can send I/O operations through the dependent host to the base. For example, the Network Support Processor, is capable of having a path from the DLP into a base which contains Line Support Processor DLPs. The host number in this case must match the field strap that is part of the NSP DLP.

<dlp list>

```

      | <---- /7\ ----- |
      |----- <dlp specification> -----|

```

<dlp specification>

```

      ---- ADDRESS --- <dlp address> --- DLPID --- <dlp id> -----> |
      >----- <dlp type mnemonic> -----|

```

SEMANTICS:

The DLP address and DLP id are field installed straps. The address is used to uniquely specify a DLP in the base. It is also used to resolve priority if more than one DLP wishes to be active at one time, with the higher address winning. The DLP id is used by system software to generate unit numbers for the units behind a DLP.

In the case of the TP-DLP, which supports one train printer, the base physical unit number is the unit number of the train printer. The base physical unit number for an HT-DLP is the unit number of the lowest numbered disk pack connected to the DLP. The other 15 possible pack units are numbered sequentially, incrementing by one.

<dlp type mnemonic>

```

      ----- CR1 -----|
      |-----|
      |-- TP1 -----|
      |-----|
      |-- MT1 -----|
      |-----|
      |-- CP1 -----|
      |-----|
      |-- HT1 -----|
      |-----|
      |-- ODT1 -----|
      |-----|
      |-- LSP1 -----|
      |-----|
      |-- NSP1 -----|

```

SEMANTICS:

CR1:

Card reader DLP is used for card readers B9115/16/17.

TP1:

750, 1100, 1500 train printer DLP.

MT1:

Phase encoded magnetic tape DLP.

CP1:

Card punch DLP.

HT1:

225, 235, 206, 207 pack, host transfer DLP.

ODT1:

Operator display terminal DLP.

LSP1:

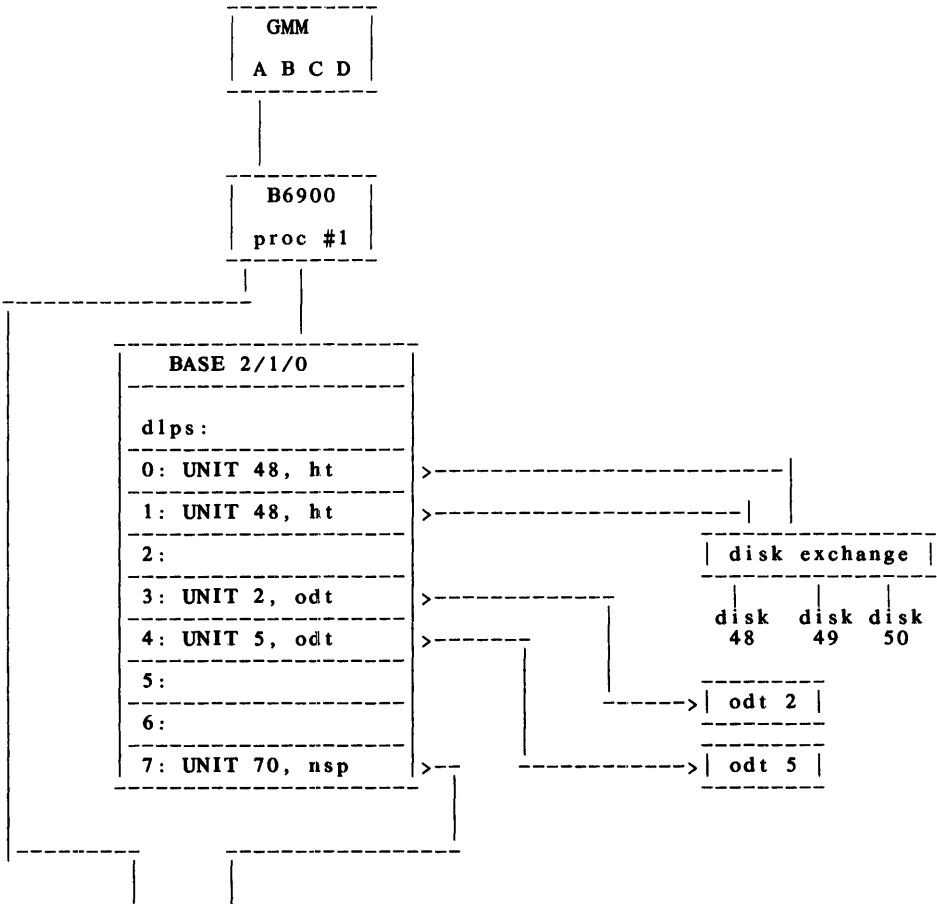
Sub-broadband byte or bit oriented line support processor.

NSP1:

Network support processor DLP.

Example 1:

Assume the following resources:



BASE 1/1/0	
dlps:	
0: UNIT 71, lsp	>---- datacom lines
1: UNIT 72, lsp	>---- datacom lines
2:	
3: UNIT 1, tp	>-----> LP 1
4:	
5:	
6:	
7:	

The following symbolic configuration file describes the above system:

```

BEGIN GROUP BLUE
PROCESSORS:
  PROC A;
MEMORY:
  PRIVATE MODS 32-59;
  SHAREREAD MODS 60-61;
I/O:
  BASE 2/1/0
    HOST 1
      ADDRESS 0 DLPID 48 HT1
      ADDRESS 1 DLPID 48 HT1
      ADDRESS 3 DLPID 2 ODT1
      ADDRESS 4 DLPID 5 ODT1
      ADDRESS 7 DLPID 70 NSP1;
  BASE 1/1/0
    HOST 1
      ADDRESS 3 DLPID 1 TP1;
    DEPENDENT HOST 4 DLPID 70
      ADDRESS 0 DLPID 71 LSP1
      ADDRESS 1 DLPID 72 LSP1;
PERIPHERALS:
  UNITS 1,2,5,48,49,50;
END GROUP;
```

B6000 SERIES MARK 32
DOCUMENT CHANGES NOTES (D NOTES)

CONTROLLER

D3565 CONTROLLER - "J, MX" RESPONSE

In reponse to a J or MX ODT message, "JOB STRUCTURE" will now be displayed, instead of "JOBS STRUCTURE".

D3647 CONTROLLER - "TD" ACCELERATION

Throughput of Burroughs TD850 operator display terminals (ODTs) can be accelerated substantially by decreasing the time for writing a page of information to those ODTs.

The acceleration is accomplished by setting a new TERM attribute.

Syntax:

```
-- TERM -- FULLPAGE --- TRUE -----|  
                | - FALSE - |
```

Setting the FULLPAGE option is recommended for TD850 terminals, which will be faster by a factor of 3 to 4 times (approximately).

The default setting of FULLPAGE is FALSE; terminals are not affected unless the option is explicitly set.

D3651 CONTROLLER - SUPPRESS FROZEN LIBRARIES

To avoid congestion in ADM, frozen libraries will no longer be displayed in an ACTIVE mix picture. They will be displayed by the A ALL ODT message and the MIX ODT message.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

CONTROLLER

P3045 CONTROLLER - COMPLETED ENTRIES SHOWS SPURIOUS SUBSYSTEM

The CONTROLLER now displays the correct heading when an ODT request "ACTIVE IN BLUE" followed by "C" is entered on a B6800 multiprocessor system. Previously, the system would return the correct entries but with the heading "COMPLETED IN BLUE".

P3046 CONTROLLER - "TERM USER" VS. "AT HOSTNAME"

The CONTROLLER no longer sends a bad usercode when an ODT on a loosely-coupled system has had its "TERM USER" disabled followed by an "AT <host>".

P3099 CONTROLLER - "SUBSYSTEM=." VS. QF CORRECTION

When "SUBSYSTEM=." is included in an MQ command for a queue that does not yet exist, the CONTROLLER will no longer take a dump.

P3236 CONTROLLER - VERY LONG FILE NAMES

A file name composed of fourteen 17-character names will no longer cause a fault in the CONTROLLER.

P3254 CONTROLLER - "BACKUPQUEUER" CALL

The PRINTJOB procedure was not passing the proper print priority in its call of BACKUPQUEUER. The usual result was that items would be placed later in the print queue than they should have been, but it was possible to cause a LOOP TIMER interrupt. The correct field is now isolated for the BACKUPQUEUER call.

P3352 CONTROLLER - MULTIPAGE "PER" DISPLAY

If a PER request was made that would display one more unit than the number of lines on the ODT, the last unit would never be displayed. (NO NEXT PAGE). This problem has been corrected.

P3428 CONTROLLER - MISSING "#" ON ODT FOR SWAPTASK

Whenever a swaptask was scheduled, because of an HS or because Olayscout was waiting for sectors, and then the swaptask started execution, the "#" would be missing from the ODT mix display even though the task actually was running as a swaptask. This problem has been corrected.

P3516 CONTROLLER - HOUR FIELD IN "TIMEAT"

The problem of handling more than 2 digits in the TIMEAT define has been corrected.

P3517 CONTROLLER - "NS" CORRECTION

The continuation screen no longer overwrites the first 4 characters on an ODT.

P3659 CONTROLLER - "AA" CORRECTION

A problem has been corrected where entering "AA" at the ODT would sometimes display the wrong information.

P3750 CONTROLLER - "PA" CORRECTION

PA now checks for invalid input/output unit types; previously, no checking was done.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

CONTROLWARE

D3191 CONTROLWARE - CONTROLWARE VS. FIRMWARE, PACK CONTROLS

Prior to the Mark 31 system software release, FIRMWARE files were required for the B9380 (disk pack type 215) and B9385 (disk pack types 225 and 235) controllers. The firmware files were released with the following titles:

SYSTEM/FIRMWARE/215
SYSTEM/FIRMWARE/225AND235

The latter file was also copied as SYSTEM/FIRMWARE so that it could be used as the default in the LH ODT message.

Listings of the code files were also released with the following titles:

SYMBOL/FIRMWARE/215
SYMBOL/FIRMWARE/225AND235

The SYMBOL files were printer backup disk files to be used by Field Engineering.

The B9387 (disk pack types 206 and 207) controller was qualified on the Mark 31 (31.245) system software release. The FIRMWARE files described above were released on the Mark 31 PR1 release, together with two additional tapes: B9385CW and B9387CW. The contents of the B9387CW tape was described in GENERAL note D3126 on the Mark 31 and Mark 31 PR1 releases; the contents of the B9385CW tape was described in GENERAL note D3180 on the Mark 31 PR1 release. The code files on the previously-released CW tapes had titles CONTROLWARE/=: the code listings printer backup disk files had titles LISTING/=.

The above naming conventions for the code files required for the disk pack controllers will be revised by the Mark 33 system software release.

The term CONTROLWARE will be reserved for use in file titles of codefiles required for disk pack controllers. The term FIRMWARE will be used in file titles of code files; e.g., Data Link Processors (DLP) and Network Support Processors (NSP).

The first phase of the plan, described below, has been implemented on the Mark 32 system software release.

Note that B6900 systems require different FIRMWARE files from B6700 and B6800 systems.

The following table represents the current controller numbers and disk pack types:

Controller	Disk Pack
B9380	215
B9385 *	225,235
B9387	206,207

* Note:

Previous levels of B9385 were released as B9383 and B9384.

Mark 32 FIRMWARE and CONTROLWARE Naming Conventions

The naming conventions for the code files for B6700 and B6800 disk pack controllers remain the same as in prior releases. In addition, the file for the B9387 (disk pack types 206 and 207) controller has the following title:

SYSTEM/FIRMWARE/206AND207

Thus the code files for B6700 and B6800 are the following:

SYSTEM/FIRMWARE (copy of SYSTEM/FIRMWARE/225AND235)
SYSTEM/FIRMWARE/215
SYSTEM/FIRMWARE/225AND235
SYSTEM/FIRMWARE/206AND207

The SYMBOL/FIRMWARE files are NOT contained on the Mark 32 system tape; they are contained on the B9385CW, B9387CW and B9387CWLST tapes.

The naming convention for the B6900 controlware files is the following:

CONTROLWARE/BXXXX

The following files are released on the Mark 32 software release SYSTEM tape:

CONTROLWARE/B9385 (disk pack types 225 and 235)
CONTROLWARE/B9387 (disk pack types 206 and 207)

The printer backup disk files on the B9385CW, B9387CW and B9387CWLIST tapes have titles CWLISTING/=.

Mark 33 CONTROLWARE Naming Conventions

The code files for B6900 controlware files will use the same convention as on the Mark 32 system software release.

The titles of the code files for B6700 and B6800 systems will be changed to the following:

CONTROLWARE/MPX/B9380 (disk pack type 215)
 CONTROLWARE/MPX/B9385 (disk pack type 225 and 235)
 CONTROLWARE/MPX/B9387 (disk pack type 206 and 207)

There will be no file equivalent to SYSTEM/FIRMWARE.

The printer backup disk files on the B9385CW, B9387CW and B9387CWLIST tapes will have titles CWLISTING/=.

The syntax of the LH ODT message and the checking code in SYSTEM/PATCHCONTROLWARE will be modified to accommodate these changes.

D3200 CONTROLWARE - MARK "32" DISK PACK CONTROLWARE FILES

This note replaces the following notes previously issued:

D3180 Mark 31 PR1 release
 D3126 Mark 31 release

Controlware files for the Disk Pack Controllers B9380, B9385, and B9387 are maintained and qualified by the Downingtown plant. A release tape is forwarded to the Computer Systems Group (CSG) plants responsible for the host systems. The applicable files are copied to the system release tapes and qualified.

Controlware files used by the B6900 Peripheral Test Driver (PTD) are described in the file PTD/DOC/MAINT/HT on the PTDTESTS tape which accompanies this release.

Note: Previous levels of disk pack controller B9385 were released as B9383 and B9384.

The titles of the files (for the appropriate combination of disk pack controllers and host transfer controls) on the master tape from the Downingtown plant are changed at each release.

The current version of the controlware files required for normal large system operations are copied (using file naming conventions described in note D3191, Mark 32 release) to the Mark 32 SYSTEM tape and titled as follows:

Title on 32 SYSTEM Tape	Title on 'CW' Tape	System, Pack Types
SYSTEM/FIRMWARE	CONTROLWARE/HSTLME #	B6800,B6700 (225,235)
SYSTEM/FIRMWARE/215	CONTROLWARE/HSTLNE #	B6800,B6700 (215)
SYSTEM/FIRMWARE/225AND235	CONTROLWARE/HSTLME #	B6800,B6700 (225,235)
SYSTEM/FIRMWARE/206AND207	CONTROLWARE/HSTLPD ##	B6800 (206,207)
CONTROLWARE/B9385	CONTROLWARE/HSTLKC #	B6900 (225,235)
CONTROLWARE/B9387	CONTROLWARE/HSTLQD ##	B6900 (206,207)

Note: B9385CW

Note: B9387CW

* Note: Includes patch T3702-1 which enables binary addressing which is required for this configuration.

** Note: "Seek timeout" is reported instead of "not ready" when loss of servo phaselock occurs.

The second identifier in the file titles of the controlware files on the 'CW' tapes which are required for normal system operation are composed of the concatenation of the following items: 'HSTL', 'L', 'V'.

HSTL denotes Host Load file (B6700, B6800 system with Model III Multiplexor or B6900 system)

L denotes the combination of host controls (HTC1A or HT-DLP) and disk pack controller, indicated with a letter.

V denotes the current version, indicated with a revision letter: A,B,C, etc.

The current allowable letters for 'L' are denoted in the following table.

Note: 'Col ID' in the following table denotes the number of hex digits of relevant firmware level information.

Note: 'Section' in the following table denotes the section number in the document CWLISTING/A5615A on the B9385CW and B9387CW tapes.

B6000 SERIES MARK 32

L	32 V	Col ID	Disk Pack Controllers	Disk Pack Types	Host Control	Section	System
K	C	D2C3	B9385	225,235	HTDLP	4.1	B6900
M	E	D4C5	B9385	225,235	HTC1A	4.2	B6700/B6800
N	E	D5C5	B9383	215	HTC1A	4.3	B6700/B6800
P	D	D7C4	B9387	206,207	HTC1A	4.4	B6800
Q	D	D8C4	B9387	206,207	HTDLP	4.5	B6900
R *	B	D9C2	B9385	225,235	HTC1A		B6700/B6800

* Note: 'Standalone' IVR, contained on the CW tapes, is not required for normal system operation.

Loading Controlware Files

SYSTEM/LOADER is used to load diskpack controlware files on multiplexor and MLIP systems. The format of the command is the following:

```
-- LH -- PK <nnn> --- MPX <m> PATH <p> ----->
                        | - VIA <port #> <lem #> <dlp #> - |
>- <filename> FROM <tapename> -----|
```

Examples:

```
"LH PK 45 MPX 1 PATH 2 CONTROLWARE/B9387 FROM BSYSTEM ;"
"LH PK 45 VIA 0 0 5 CONTROLWARE/B9385 FROM BSYSTEM; "
```

NOTE: A space is required between PK and <pack #>.

Contents of 'CW' Tapes

All the files for normal system operation, diagnostic and maintenance operations for B6000 series systems are copied to the tapes B9385CW, B9387CW and B9387CWLIST.

The B9385CW tape contains all the controlware and CWLISTING files required for the B9385 disk pack controller for B6700, B6800 and B6900 systems.

The B9387CW tape contains all the controlware files required for the B9387 disk pack controller for B6800 and B6900 systems.

The B9387CWLIST tape contains all the CWLISTING files required for the B9387 disk pack controller for B6800 and B6900 systems.

The file CWLISTING/A5615A is a printer backup disk file containing the system notes describing the changes made to the controlware files since the previous Downingtown release. After copying from tape to disk, the file may be printed (103 pages) using SYSTEM/BACKUP with a command similar to the following:

```
?PB "CWLISTING/A5615A" LP14
```

For each 'CONTROLWARE/<identifier>' file, there is a corresponding 'CWLISTING/<identifier>' file.

The CWLISTING/= files are B6000 series printer backup disk files. A single file may be printed using the following:

```
COPY CWLISTING/<identifier> FROM <tape identifier>; END
?PB "CWLISTING/<identifier>" LP14.
```

All the files may be printed using the following:

```
COPY CWLISTING/<identifier> FROM <tape identifier>; END
?PB "CWLISTING" LP14.
```

Applied Patches

The following patches have been applied:

Patch 09221-2 BX383/4/5 Controlware Release 2.2 (Code ME)

This patch corrects the following problems:

1. If a Unit Busy R/D was generated during a conditional Subsystem Poll (indicating no units are seek ready) and the controller was lock enabled, Controlware erroneously locked the controller.

2. If a seek timeout status was detected from a drive, Controlware erroneously generated a Not Ready R/D instead of a Seek Timeout R/D.
3. If a restore had to be performed on a drive at unit select or initial seek time, the host could accidentally be put into a state which could cause an interface parity error.
4. If a Conditional Cancel was sent from the host at the same time Controlware was trying to place the host into R/D mode, the interface would hang. Controlware now checks the host status after R/D mode and reissues an R/D mode command if Conditional Cancel is present. A new R/D TAG 069 has been added to Controlware.

This patch also sets bit 02 (most significant bit=01) of the buffer memory location "PATCH_CODE" so that a Read Memory C/D may be used to determine if the patch is installed.

Host Load file (HSTLME) patch:

```
CC PATCH HSTLME 0 0 5496 4 E040
CC PATCH HSTLME 0 0 9484 4 D285
CC PATCH HSTLME 0 0 11672 4 F8B2
CC PATCH HSTLME 0 0 6956 4 6BCB
CC PATCH HSTLME 0 0 16316 4 A12C
CC PATCH HSTLME 0 0 17992 12 F871A0054AE6
CC PATCH HSTLME 0 0 16392 4 F844
CC PATCH HSTLME 0 0 16420 4 F819
CC PATCH HSTLME 0 0 16428 4 4BCE
CC PATCH HSTLME 0 0 18004 20 F84CB8FBF8C966F5C069
CC PATCH HSTLME 0 0 18024 20 6B8E66DBC049F8844A46
```

The following files have been updated:

SYSTEM/FIRMWARE	on SYSTEM	tape
SYSTEM/FIRMWARE/225AND235	on SYSTEM	tape
CONTROLWARE/HSTLME	on B9385CW	tape

Patch 09921-3 B9387 Controlware Release 1.3 (Code PD)

This patch forces Controlware to generate a Seek Timeout R/D when a "loss of servo phaselock" status is received from a 207 drive. Controlware previously generated a Not Ready R/D for this condition. This patch also sets bit 09 (least significant bit=00) of buffer memory location "PATCH_CODE" so that a Ready Memory C/D may be used to determine if the patch is installed.

Host Load file (HSTLPD) patch:

```
CC PATCH HSTLPD 0 0 176956 4 0502
CC PATCH HSTLPD 0 0 184072 8 12070BA9
CC PATCH HSTLPD 0 0 189172 20 02AC1C11227C1C113203
```

Patch 09921-4 B9387 Controlware Release 1.3 (Code PD)

This patch corrects the following problems:

1. If a Unit Busy R/D was generated during a conditional Subsystem Poll (indicating no units are seek ready) and the controller was lock enabled, Controlware erroneously locked the controller.
2. If a Conditional Cancel was sent from the host at the same time Controlware was trying to place the host into R/D mode, the interface would hang. Controlware now checks the host status after R/D mode and reissues an R/D mode command if Conditional Cancel is present. A new R/D TAG 069 has been added to the Controlware.

This patch also sets bit 10 (least significant bit=00) of buffer memory location "PATCH_CODE" so that a Read Memory C/D may be used to determine if the patch is installed.

Host Load file (HSTLPD) patch:

```
CC PATCH HSTLPD 0 0 176960 4 0504
CC PATCH HSTLPD 0 0 181800 4 2512
CC PATCH HSTLPD 0 0 188780 12 02D71BAE1000
CC PATCH HSTLPD 0 0 189192 20 04721000268220001000
CC PATCH HSTLPD 0 0 189212 20 03FB1207174A25002569
```

The following files have been updated:

SYSTEM/FIRMWARE/206AND207	on SYSTEM	tape
CONTROLWARE/HSTLPD	on B9387CW	tape

B6000 SERIES MARK 32

Patch 09921-5 B9387 Controlware Release 1.3 (Code QD)

This patch corrects the following problems:

1. The Large Systems HT-DLP cannot handle a state change from State 6 (Command Descriptor) to State 4 (Control Message). Controlware no longer attempts that state change.
2. If a Conditional Cancel was set from the host at the same time Controlware was trying to place the host into R/D mode, the interface would hang. Controlware now checks the host status after R/D mode and reissues an R/D mode command if Conditional Cancel is present. A new R/D TAG 069 has been added to the Controlware.

This patch also sets bit 09 (least significant bit=00) of buffer memory location "PATCH_CODE" so that a Read Memory C/D may be used to determine if the patch is installed.

Host Load file (HSTLQD) patch

```
CC PATCH HSTLQD 0 0 176956 4 0502
CC PATCH HSTLQD 0 0 179848 4 3BA0
CC PATCH HSTLQD 0 0 189136 24 35E3360B28131223020507BE
CC PATCH HSTLQD 0 0 189028 8 12070BA6
CC PATCH HSTLQD 0 0 189160 20 34723CD3037306923248
CC PATCH HSTLQD 0 0 189180 20 240B1207179425001569
```

The following files have been updated:

```
CONTROLWARE/B9387          on SYSTEM   tape
CONTROLWARE/HSTLQD         on B9387CW  tape
```

Directories of the 'CW' Tapes

The following tables show the files contained on the 'CW' tapes. The entries in the tables are sorted by file titles and do not represent the order in which the files occur on the tapes. An '*' in the 'CR' column denotes an uncrunched file.

B9385CW

LIBRARY TAPE: B9385CW HAS 9 FILES, 41110 SEGMENTS

FILE TITLE	CR	SEGS	CREATED	SECURITY	FILEKIND
*CONTROLWARE/D5X12C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/HSTLKC,	1000	05/08/80	PUBLIC	FIRMWARE	
*CONTROLWARE/HSTLME,	1000	06/19/80	PUBLIC	FIRMWARE	
*CONTROLWARE/HSTLRB,	1000	05/07/80	PUBLIC	FIRMWARE	
*CWLISTING/A5615A,	4050	05/07/80	PUBLIC	BACKUPDISK	
*CWLISTING/D5X12L,	9930	05/07/80	PUBLIC	BACKUPDISK	
*CWLISTING/LISTKC,	8060	05/08/80	PUBLIC	BACKUPDISK	
*CWLISTING/LISTME,	8250	05/07/80	PUBLIC	BACKUPDISK	
*CWLISTING/LISTRB,	6820	05/07/80	PUBLIC	BACKUPDISK	

B9387CW

LIBRARY TAPE: B9387CW HAS 26 FILES, 29050 SEGMENTS

FILE TITLE	CR	SEGS	CREATED	SECURITY	FILEKIND
*CONTROLWARE/D7MB1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MB2C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MC1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD2C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD3C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD4C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD5C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD6C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MD7C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7ME1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF2C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF3C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF4C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF5C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF6C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MF7C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MH1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MP1C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MP2C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7MP3C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/D7Y12C,	1000	05/07/80	PUBLIC	FIRMWARE	
*CONTROLWARE/HSTLPD,	1000	05/08/80	PUBLIC	FIRMWARE	
*CONTROLWARE/HSTLQD,	1000	05/08/80	PUBLIC	FIRMWARE	
*CWLISTING/A5615A,	4050	05/07/80	PUBLIC	BACKUPDISK	

B9387CWLIST

LIBRARY TAPE: B9387CWLIST HAS 39 FILES, 190320 SEGMENTS

FILE TITLE	CR	SEGS	CREATED	SECURITY	FILEKIND
*CWLISTING/D7AA2L,	%	1260	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AA3L,	%	710	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AA4L,	%	710	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AA5L,	%	710	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AA6L,	%	730	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AB1L,	%	5770	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AB2L,	%	290	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD1L,	%	4160	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD2L,	%	3040	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD3L,	%	2250	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD4L,	%	2200	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD5L,	%	3030	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD6L,	%	4720	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AD7L,	%	3260	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AE1L,	%	3170	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF1L,	%	3640	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF2L,	%	2920	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF3L,	%	2080	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF4L,	%	3390	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF5L,	%	2980	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF6L,	%	4420	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AF7L,	%	4050	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AP0L,	%	2670	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AP1L,	%	4050	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AP2L,	%	4390	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AP3L,	%	370	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AZ1L,	%	930	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AZ2L,	%	870	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AZ3L,	%	860	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AZ4L,	%	870	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7AZ5L,	%	1420	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7MC1L,	%	2480	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7MH1L,	%	2950	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7PR1L,	%	3800	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7QXRL,	%	34820	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7RXRL,	%	43030	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/D7Y12L,	%	10710	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/LISTPD,	%	8360	05/07/80	PUBLIC	BACKUPDISK
*CWLISTING/LISTQD,	%	8250	05/08/80	PUBLIC	BACKUPDISK

SYSTEM/PATCHCONTROLWARE

For a description of SYSTEM/PATCHCONTROLWARE, see note D3128, Mark 32 release.

DOCUMENT CHANGES NOTES (D NOTES)

DATA COMMUNICATIONS

D3202 DATACOM - "B6900" DATA COMMUNICATIONS

The B6900 Data Communications Subsystem is substantially different at the hardware level from B6700 and B6800 (DCP) data communications. In place of the DCP and Adaptor Cluster modules, the B6900 data comm subsystem uses the Network Support Processor (NSP) and the Line Support Processor (LSP) modules.

The NSP and the LSP are Data Link Processors (DLPs) that communicate with the main system through the Message Level Interface Port. An LSP can support up to 16 data comm lines, and an NSP can control up to 4 LSPs. Additional configuration information is contained in "D3203 GENERAL -- B6900 Overview".

General Software Changes

A new Network Definition Language, called NDII, has been developed for use with this new hardware. NDII is described in the "Network Definition Language II" document provided as NDII/DOCUMENT on the SYSTEMNOTES tape, Mark 32 release. This document also describes the logical partitioning of data comm functions between the NSP and the LSP.

A SOURCENDII symbolic (available as SYMBOL/SOURCENDII on the BSYMBOL tapes, Mark 32 release) has been provided as an example of how NDII programs are written. In order to provide a B6800-compatible interface to an MCS, certain NDII programming conventions must be followed. The SOURCENDII symbolic contains numerous comments explaining these conventions in detail.

In order to initialize an NSP, a NIF file and firmware file must be provided. A standard firmware file called FIRMWARE/NSP is provided. The NIF file is provided by the user by compiling a source NDII program, such as the example SYMBOL/SOURCENDII which is included with this release. The file name of the NIF must include ".../NIF" as the last node. If <prefix>/NIF is the NIF file, the MCP creates an auxiliary file <prefix>/DCPPCODE when the NSP is initialized with the NIF the first time.

The "ID" (Initialize Datacom) ODT message is used to initialize an NSP (or DCP), to interrogate or alter the settings of various data-comm-related options, and to direct commands to the DCCONTROL process. The syntax and semantics of the ID message are described in "D3356 GENERAL -- New and Old ODT Messages".

One of the functions that can be initiated through the ID message is data comm auditing. A new program, DCAUDITOR, has been written to analyze the files produced during the auditing process. DCAUDITOR is described in "D3395 DCAUDITOR -- DCAUDITOR Implementation".

A dump of the local memory of a specified NSP can be requested through the DUMP option of the ID message. The file produced by this message can be analyzed with the new NSPDUMPANALYZER program. This program is described in "D3430 NSPDUMPANALYZER -- NSPDUMPANALYZER Implementation".

DCWRITE Differences

An effort has been made to replicate the function of B6800 DCWRITES on the B6900. However, because of the new hardware configuration, the following hardware-oriented DCWRITES are implemented differently for NSP data comm:

SET APPLICATION NUMBER (TYPE = 38)

In DCP data comm, application numbers are associated with request sets. The NSP data comm equivalent of a request set includes the station's Editor and the station's Algorithm. In NSP data comm, application numbers are associated with Editors; station Algorithms cannot be changed. The SET APPLICATION NUMBER DCWRITE causes a "Change Station Editor" instruction to be issued to the NSP. In addition, if an EXTERNAL variable called APPLICATION is declared for the station in NDII, a "Set External Variable" instruction will be issued to the NSP to set the APPLICATION variable to the application number specified in the DCWRITE.

ANSWER THE PHONE (TYPE = 100)

When the phone rings on a DCP dial-in line, the data comm subsystem will answer the phone if AUTOANSWER is TRUE or will notify the main system if AUTOANSWER is FALSE. In the latter case, the main system would probably send an ANSWER THE PHONE DCWRITE. When the phone rings on an NSP dial-in line, the data comm subsystem will answer the phone if the NDII variable LINE.DISCONNECTION has the value AUTOANSWER or will ignore the ring if LINE.DISCONNECTION is NONE. Thus, there is no opportunity or need for the host to send an ANSWER THE PHONE DCWRITE to an NSP data comm subsystem.

SET/RESET AUTO-ANSWER (TYPE = 102)

For NSP data comm, the SET AUTO-ANSWER DCWRITE has the effect of setting the NDLII variable LINE.DISCONNECTACTION to AUTOANSWER. The RESET AUTO-ANSWER DCWRITE causes LINE.DISCONNECTACTION to be set to NONE. Note that resetting auto-answer on an NSP subsystem causes the ring indication to be ignored; on a DCP subsystem, the main system would be notified that the phone was ringing.

FORCE LINE NOT READY (TYPE = 105)

On NSP subsystems, the action of the FORCE LINE NOT READY DCWRITE is identical to the action of the MAKE LINE NOT READY DCWRITE (TYPE = 97).

UPDATE LINE ATTRIBUTES (TYPE = 131)

The UPDATE LINE ATTRIBUTES DCWRITE is not implemented and will cause a DCWRITE ERROR on NSP subsystems.

ATTACH LINE ADAPTER (TYPE = 160)
DETACH LINE ADAPTER (TYPE = 161)
ATTACH DCP (TYPE = 164)
DETACH DCP (TYPE = 165)
INITIATE DCP TEST (TYPE = 168)
INITIATE CLUSTER TEST (TYPE = 169)
CANCEL CLUSTER TEST (TYPE = 170)

These maintenance DCWRITES are not appropriate for and are not implemented for NSP data comm. Because these are not available, MAINTMCS cannot be used with NSP data comm. The PTD data comm maintenance package should be used in its place.

DCWRITE Results

Data comm result messages that contain hardware-specific information may return incomplete information on an NSP subsystem for items that relate to the DCP and have no counterpart on an NSP subsystem.

LINE INTERROGATE RESULT (CLASS = 24)

On a DCP subsystem, the LINE INTERROGATE RESULT returns information about the specified line and about the station currently referenced by that line. On an NSP subsystem, the LINE INTERROGATE RESULT returns information about the line and, since there may be zero, one, or many "currently referenced" stations, about the station specified in the original LINE INTERROGATE DCWRITE.

ERROR RESULT (CLASS=99), Line/Station Format

Result Field	DCP Value	NSP Value
MSG[0].[47:08] =	99	99
0 [39:08] =	AC Register	0
0 [31:08] =	AI Register	0
0 [23:24] =	LSN	LSN
MSG[1].[47:08] =	Result Byte Index	Result Byte Index
0 [39:06] =	LINE status	LINE status (mapped from NSP info)
0 [33:01] =	NDL LINE(TOG[1])	0
0 [32:01] =	NDL LINE(TOG[0])	0
0 [31:08] =	Last Flag Set	First Flag Set (from FIRSTERROR)
0 [23:24] =	Error Flag field	Error Flag field (from NSP errors)
MSG[2].[47:08] =	Character Register	0
0 [39:16] =	Last Sleep Addr.	0
MSG[4].[23:24] =	Orig. DCWRITE type	35 (ENABLE INPUT)

ERROR RESULT (CLASS=99), Switched Status Format

Result Field	DCP Value	NSP Value
MSG[0].[47:08] =	5, 7, or 99	5, 7, or 99
0 [39:08] =	AC Register	0
0 [31:08] =	AI Register	0
0 [23:24] =	LSN	LSN
MSG[1].[47:08] =	Result Byte Index	Result Byte Index
0 [39:08] =	Termination Reason or Toggles	Termination Reason or 0
0 [30:07] =	Switched Status	Switched Status
MSG[4].[23:08] =	Orig. DCWRITE type	Orig. DCWRITE type

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DATA COMMUNICATIONS

P2782 DATACOM - "DCRECON" LINE RESULT NOT RETURNED

If DCRECON sent a MAKELINENOTREADY or BLASTLINE request to the DCP and station zero on the line was or could be attached to an MCS, the line result message was sent back to DCRECON. If, however, station zero could not be attached because its MCS had been DSed, the line result message was lost and not sent to DCRECON. This problem has been corrected.

P2783 DATACOM - EXTENDED LINE TALLYS PROBLEMS

When using extended line TALLYS, if an "INTERROGATE STATION ENVIRONMENT" (TYPE=4) DCWRITE were issued requesting DCP station information, the DCP station description word in the result was incorrect. This problem has been corrected.

Also, RECALLOBJOOUTPUT also worked improperly when extended line TALLYS were used. This problem has been corrected.

P2784 DATACOM - "DCSYSTEMTABLES" RETURNS WRONG RESULT

When options 1, 5 or 6 are specified on a call to DCSYSTEMTABLES, the procedure did not return the correct size of the information in words as the documentation says it always will. This problem has been corrected.

P2795 DATACOM - "READNIF" TEST FOR VALID RECORD INDEX

When the NIF file was read, the record index specified was not tested to determine if it was within the allowed bounds. This problem has been corrected.

P2857 DATACOM - UNLOCK LINE BY "DCIOFINISH" CALLERS

Procedure DCIOFINISH previously had a parameter which would cause it to UNLOCK a line which had been LOCKed by its caller. However, since sometimes it had to calculate the DLS of the line given only the LSN from the message passed to it, it could attempt to UNLOCK the wrong lock if cluster exchange reconfiguration had just changed the DLS of the station. If the MCP were running under DIAGNOSTICS, this could result in a dump by "UNOWNED LIBERATE". In any event, the wrong lock might be liberated and the correct lock would not be liberated. When a later attempt to procure the correct lock was made, such as by DCRECON or one of the DCC stacks, it would be hung forever. This problem has been corrected by requiring all callers of DCIOFINISH to liberate line locks, if necessary, before the call and to reacquire them afterwards, if required.

P2960 DATACOM - "RECALLOBJOOUTPUT" ON UNINITIALIZED "DCP"

Occasionally, a dump by INVALID DCCLEAR could occur if DCCLEAR forked off DCRECON to have object job output returned for a station on a running DCP, but the DCP was no longer running by the time DCRECON processed the request. The function requested will be performed by DCTERMINATE in that case, so the dump is unnecessary. It will no longer occur under these circumstances.

P3229 DATACOM - "DCSYSTEMTABLES" OPTION "5" "SEG ARRAY" FAULT

The length of the information to be returned by a DCSYSTEMTABLES Option 5 request was being computed incorrectly. If the DCP macro name was an exact multiple of 6 characters, a SEG ARRAY fault could occur. The length is now computed properly and the array resized when necessary.

P3677 DATACOM - "DLS" CORRUPTED IN FULL DUPLEX DISCARDS

In certain situations involving full-duplex lines, the MCP might get an INVALID INDEX in DCCONTROL attempting to use the DLS extracted from a discarded full-duplex message space. The problem arose because DCPPROGEN was corrupting the DLS field when it changed the message class to discarded. The error would probably go unnoticed unless main memory DCP tables were being used.

In a related problem, the station part of the DLS field of an auxiliary line input message was wrong if extended line TALLYS or TOGs were used.

Both these problems have been corrected.

P3771 DATACOM - "DCFILELOCK" DEADLOCK

A problem, which occurred when terminating a DCP which had outstanding object job outputs and main memory datacom tables, has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DCALGOL

D3075 DCALGOL - QUEUES AS LIBRARY PARAMETERS

Queues are now allowed as parameters to library procedures in DCALGOL.

D3357 DCALGOL - REMOVE "PORT" AND "SIGNAL"

PORTs and SIGNALs have been deimplemented. The DCALGOL constructs which relate to PORTs and SIGNALs are no longer recognized by the compiler. These constructs include the following: PORT and SIGNAL declarations, PORT and SIGNAL ARRAY declarations, PORT and SIGNAL attribute assignment and reference, and OPEN and CLOSE of a PORT. Also, the following SIGNAL statements are no longer recognized: SENDMESSAGE, SENDRESPONSE, RECEIVEMESSAGE, RECEIVERRESPONSE and CLEAR. (Note that CLEAR(<diskheader identifier>,<aexp>) has not been deimplemented.) Also, the HOSTINFOFUNCTION function has been deimplemented, along with the port and signal parameters to the SIZE function.

D3444 DCALGOL - QINSERTEVENT DOCUMENTATION

On page 3-26 of the B6700/B7700 DCALGOL Reference Manual (Form No. 5000052), the statement is made that "QINSERTEVENT is an EVENT read-only attribute". This is incorrect. Although it is inappropriate to do anything except wait on it or check whether it has happened, the system will not enforce the READ-ONLY nature of this attribute.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DCALGOL INTRINSICS

D3629 DCALGOLINT - "DCALGOLINTRN" SUBSUMED BY "GENERALSUPPORT"

Effective with the Mark 32 release, the DCALGOL intrinsics have been subsumed by the GENERALSUPPORT library, described in GENERAL note D3354.

DOCUMENT CHANGES NOTES (D NOTES)

DCAUDITOR

D3395 DCAUDITOR - "DCAUDITOR" IMPLEMENTATION

SYSTEM/DCAUDITOR is a program which performs analysis of an audit file produced by the B6900 Datacom subsystem procedures of the MCP. The items which are actually audited are controlled by the ID/DC ODT input message (see GENERAL note D3356 for the enhanced ID/DC message). If any audit options are specified, at least one audit file will be produced. If audit options specify that auditing of datacom initialization is requested, the MCP will produce an audit file with the title "NSPAUDIT/DCINITIAL/<NSP-unitid>". If audit options specify that auditing of the running datacom subsystem is requested, the MCP will produce an audit file with the title "NSPAUDIT/DCCONTROL/<NSP-unitid>".

SYSTEM/DCAUDITOR has several modes of operation.

The mode of operation is controlled by the parameter string which is passed to SYSTEM/DCAUDITOR at initiation time; i.e.

RUN SYSTEM/DCAUDITOR ("<string>"); VALUE=<nnn>
 <nnn> is the unit number of the NSP whose audit file is to be analyzed.

The string parameter <string> can have the following values:

DCCONTROL	The entire audit file produced by the MCP procedure DCCONTROL will be analyzed.
DCCONTROL <nnn>	Only the last <nnn> records of the audit file produced by DCCONTROL will be analyzed.
DCINITIAL	The entire audit file produced by the MCP procedure DCINITIAL will be analyzed.
DCINITIAL <nnn>	The last <nnn> records of the audit file produced by DCINITIAL will be analyzed.

SYSTEM/DCAUDITOR will currently perform detailed analysis for NSP formatted requests and results; however, only the TYPE/CLASS field will be analyzed for DCP and DCWRITE formatted requests and results. For DCP and DCWRITE formatted requests and results, the entire message will be printed in hex.

Samples of the report generated by SYSTEM/DCAUDITOR are included. Remember again that the items which appear in the audit report are controlled by the ID/DC ODT message.

D3637 DCAUDITOR - ADD ADDITIONAL INFORMATION TO "DCAUDIT" FILE

Timestamp information, as well as environmental information, has been added to each audit entry. DCAUDITOR will now print the environment in which the audit file was produced.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DCP PROGRAM GENERATOR

P2802 DCPPROGEN - ALLOW "TERMINATE NORMAL"

A TERMINATE NORMAL statement may now be used in a message request set. The TERMINATE NORMAL statement in an output request set is equivalent to a TERMINATE OUTPUT statement. A TERMINATE NORMAL statement in an input request set is equivalent to a TERMINATE INPUT.

P3497 DCPPROGEN - GARBLED "BAUDOT" TRANSLATE TABLES

Since Mark 27, DCPPROGEN has attempted to merge translate tables and string pools into one memory area. If one of the terminals for a DCP specified CODE=BAUDOT and user-specified translate tables were also declared, the string pool would be incorrectly merged over the DCPPROGEN-supplied BAUDOT translation table. Since Mark 30, the NDL compiler always supplies a "user" translate table for internal use; since that time, CODE=BAUDOT has not generated a proper translate table even without user-declared translate tables. These problems have been corrected.

A related problem since Mark 31 (in which use of CODE=BAUDOT on a DCP with local memory could cause multiple LABEL errors in DCPPROGEN) has also been corrected.

P3666 DCPPROGEN - CONSECUTIVE LINE TALLY USAGE

If consecutive extended line TALLYS were referenced in an NDL program, DCPPROGEN could generate code that was off by one word in accessing them. This problem has been corrected.

P3677 DCPPROGEN - "DLS" CORRUPTED IN FULL DUPLEX DISCARDS

In certain situations involving full-duplex lines, the MCP might get an INVALID INDEX in DCCONTROL attempting to use the DLS extracted from a discarded full-duplex message space. The problem arose because DCPPROGEN was corrupting the DLS field when it changed the message class to discarded. The error would probably go unnoticed unless main memory DCP tables were being used.

In a related problem, the station part of the DLS field of an auxiliary line input message was wrong if extended line TALLYS or TOGs were used.

Both these problems have been corrected.

P3775 DCPPROGEN - "INCREMENT TRAN" STATEMENT

The INCREMENT TRAN statement for Message-Oriented lines was not writing the incremented transmission number back into the station table. This problem has been corrected.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DCSTATUS

P3012 DCSTATUS - PRINT STATION TABLE BASE CORRECTLY

Message-Oriented Datacom station table base words are now interpreted and printed correctly.

Also, the NETWORK option has been altered to correctly report adapter types present on message-oriented lines.

P3539 DCSTATUS - "17-CHARACTER" NAMES FOR GRAPH OUTPUT

DCSTATUS GRAPH option now prints out the first 17 characters of line and station identifiers instead of only the first 9.

P3572 DCSTATUS - PREVENT "NIF/DCPCODE" FILE CHANGES

DCSTATUS no longer, under certain conditions, changes global datacom file titles to titles with a usercode.

P3600 DCSTATUS - CALCULATE STATION TABLE BASE SIZE

In some instances, DCSTATUS was printing out station tables incorrectly for stations after the first station on a line. This no longer occurs.

DOCUMENT CHANGES NOTES (D NOTES)

DIAGNOSTIC MCS

D3344 DIAGNOSTMCS - INPUT SOURCES

Page 4 of the DIAGNOSTICMCS Reference Manual (Form No. 5001514) states that input from a card file will be accepted any time it is present if CDONLY is SET. CDONLY must be RESET for this to work properly.

D3427 DIAGNOSTMCS - NONNUMERIC DIAL CHARACTERS

The DIAL command now accepts, besides digits, various time delays and dialing option characters. However, this additional information has meaning only for DCP Message-Oriented and B6900 NSP Datacom; it cannot be used with Auto Call Units (ACUs) for DCP Datacom.

Revised Syntax:

-- DIAL --<station id>--<dial sequence>--|

<dial sequence>

<----->	
	<- , -
-----<digit>-----	
- *	----
- #	----
- W	----
- D0	----
- D1	----
- D2	----
- D3	----
- D4	----
- D5	----
- D6	----
- D7	----
- D8	----
- D9	----

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DIAGNOSTIC MCS

P3163 DIAGNOSTMCS - ERROR IN "BTB" ATTACH BY "LSN/DLS"

In running the Back-to-Back tests, if an attempt were made to attach a BTB station by LSN or DLS rather than by name, and if this were the first station attached by the MCS, DIAGNOSTICMCS would fault with a memory protect error. This problem, which was caused by scanning an uninitialized station name array, has been corrected. The problem did not occur if the regular ATTACH command were used.

P3253 DIAGNOSTMCS - ERROR IN "BTB ALL REPEAT <STRING>"

In DIAGNOSTICMCS back-to-back testing, one form of the BTB REPEAT command allows the user to specify the string used as the initial text of the test message. When the BTB test number was specified as ALL, rather than as a specific number, the specified string was not being used; instead, an equivalent number of NUL characters were generated. This no longer occurs.

P3601 DIAGNOSTMCS - ALTER CORRECT STATION ADDRESS

DIAGNOSTICMCS was altering the address character of the last station referenced, rather than the designated station. This problem has been corrected.

P3602 DIAGNOSTMCS - PREVENT DUMP ON "<DCL>"

Entering a "<DCL>" with DCP#, cluster # or line # too far out of range could cause program dumps. This problem has been corrected.

P3603 DIAGNOSTMCS - ATTACHED "CA" OPTION

DIAGNOSTICMCS documentaton allowed the CA option in station attaches. SYSTEM/DIAGNOSTICMCS did not have the required code to parse this option. This option is now allowed.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - GENERAL

D3108 DMSII - NEW DATA BASE STACK STRUCTURE

To support Shared ACCESSROUTINES, the data base stack structure has been modified. As a result, the ACCESSROUTINES have been made an executable code file which is run automatically at the first open of a data base. In many respects, the new ACCESSROUTINES act in much the same way as a shared library.

Operational Differences

For the most part, operation of new data bases should be identical with previous data bases. The primary differences arise strictly from the fact that the ACCESSROUTINES are now executable code and that compiled-in values for task attributes like PRIORITY and CORE and others will be meaningful.

When the first user of a data base opens the data base, the ACCESSROUTINES will be run automatically. The ACCESSROUTINES will remain scheduled if the resources required are not available. Proper choice of a compiled-in CORE estimate may be desirable if the compiler estimate of the data base core requirements are inaccurate. If it is desirable that a particular critical data base never be scheduled, it may be advisable to mark the ACCESSROUTINES code file as a Control Program.

Communication with new data bases continues to be available by use of the SM ODT message. In addition, the AX ODT message may be used interchangeably with the SM ODT message.

If the data base stack is manually DSed by the operator, the data base stack will terminate with a Critical Block Exit and all users of the data base will be DSed for a Death in the Family.

Recompilation of User Programs

As a result of MCP changes required to implement new ACCESSROUTINES, user programs compiled on 28 or earlier releases cannot be used against new ACCESSROUTINES. These programs should be recompiled. 28 or earlier user programs may still be used against 31 or earlier data bases on a 32 MCP.

Generating the ACCESSROUTINES

New ACCESSROUTINES may only be compiled using the 32 level DMALGOL compiler. The 32 level DMALGOL compiler may not be used to compile ACCESSROUTINES for 31 or earlier data bases. Attempting to use a 31 or earlier ACCESSROUTINES generated with a 32 DMALGOL compiler will result in a DMOPENERERROR 36. Using a 31 or earlier DMALGOL compiler on a 32 level ACCESSROUTINES will result in syntax errors. Generation of the rest of the tailored software, UTILITY, RECOVERY, etc., does not require a 32 level DMALGOL compiler.

New Stack Structure

New ACCESSROUTINES have true segment dictionaries separate from the data base stack itself. A programdump that has DBS set will dump the data base stack but not the segment dictionary associated with the data base. This will be true even if CODE is requested in the dump.

Outer block declarations of the ACCESSROUTINES will now be at lex level 2 rather than at lex level 1 as before. All other environments have moved to higher lex levels. Entry point procedures to the ACCESSROUTINES now run at lex level 5.

The order of structure environments, now at lex level 3, have been reversed in the data base stack. The last compiled structure environment will be the first lex level 3 environment found near the base of the data base stack.

D3170 DMSII - SHARED "ACCESSROUTINES," DATA BASE EQUATION

Because Mark 32 system software produces re-entrant ACCESSROUTINES code files (described in DMS II note D3108), two or more data bases may simultaneously use one set of ACCESSROUTINES. This note describes new capabilities and procedures for the creation, use, and maintenance of multiple data bases that share one ACCESSROUTINES codefile. To take advantage of these new features, data base user programs must be compiled using Mark 32 level compilers. However, installations choosing not to use the features described here may continue to use DMSII data bases on Mark 32 DMSII software without re-compiling user programs or altering operational procedures.

Conversion of ACCESSROUTINES code files to re-entrant format can provide significant reductions in memory requirements when more than one data base is active and enhances an installation's ability to manage data bases and user program libraries. For example, test and production data bases using a single ACCESSROUTINES code file may both be active at the same time but use separate audit and data files. Alternatively, a large data base could be divided into smaller data bases to permit their independent management, reduce recovery costs following system failures, or more efficiently use secondary storage devices.

Prior to the Mark 32 release, multiple code files had to be produced for user programs that opened different data bases even though the programs were otherwise identical. The data base equation capability described here permits a user program compiled using the description file of one data base to operate on other logically similar data bases without re-compilation. This feature can reduce the size of user program libraries, eliminate the need for multiple but logically identical programs, and permit full use of user program re-entrancy to reduce memory and diskpack storage requirements. In addition, data base equation allows user programs to open data bases stored under other usercodes and on pack families not normally visible to a task via its FAMILY attribute.

This note describes changes in the algorithm used to open Mark 32 level data bases, creation and maintenance of data bases that share ACCESSROUTINES, host language and WFL syntax supporting data base equation, and usercode and security considerations that may restrict a program's ability to open particular data bases.

DATA BASE OPEN ALGORITHM

When a program executes a host language OPEN statement, an MCP procedure is called to service the request. The MCP first determines if the requested data base is currently active. If so, the user program is "attached" to the data base and allowed to proceed. When the requested data base is not active, information about the data base must be located and appropriate memory structures constructed before the MCP can permit the user program to continue. The information required to initialize a data base stack is stored in the ACCESSROUTINES code file. The algorithm used by the MCP to find the ACCESSROUTINES code file has been altered on the Mark 32 release to permit data base equation and sharing of ACCESSROUTINES by multiple data bases.

Prior to Mark 32, a user program requesting an inactive data base called SALES, for example, would cause the MCP to look for a file called ACCESSROUTINES/SALES to obtain information describing the data base stack. This file had to be visible via the user program's FAMILY task attribute and stored in either the system directory or the usercode directory under which the program was executing. This algorithm will continue to be used for Mark 31 and earlier data bases and user programs running on a Mark 32 MCP.

The Mark 32 software release introduces data base attributes. Data base attributes are functionally similar to file attributes and can be manipulated during both compilation and execution. At present, data bases have only one attribute, called TITLE. The MCP procedure servicing data base open requests inspects the TITLE attribute to determine if the requested data base is currently active. If so, the user program is "attached" to the data base and allowed to continue. If an inactive data base is requested, the TITLE attribute is used to construct the title of the data base's control file. The title of the appropriate ACCESSROUTINES code file is obtained from the data base control file. Information in the ACCESSROUTINES code file is then used to construct the data base stack before the user program is allowed to proceed.

Data base equation can be used to alter the TITLE attribute of a data base declared in an application program any time the data base is not open. The TITLE attribute may contain a usercode and family name to be used in locating the data base control file. Complete syntax of a data base TITLE is described in the section on data base equation.

DATABASES SHARING ACCESSROUTINES

This section describes operation procedures and new DASDL syntax for the creation, use, and maintenance of multiple data bases that share a single ACCESSROUTINES code file. Changes to DASDL syntax are summarized at the end of this section.

Dynamic Parameters

Although data bases sharing an ACCESSROUTINES code file are physically independent, they must describe identical data base structures including data sets, remaps, index sets, and accesses. Furthermore, corresponding data sets in each data base must describe identical data items and have identical record formats.

However, such data bases may differ in a number of characteristics. These characteristics, called Dynamic Parameters, include:

- ACCESSROUTINES code file title
- Data base and logical data base guardfile titles
- Audit trail attributes
- Control file attributes
- AREAS, KIND, family name, and buffer specifications of data sets and index sets
- REBLOCKFACTOR of data sets
- ALLOWDCORE
- CONTROLPOINT and SYNCPOINT frequencies

Dynamic parameters (except control file attributes) are stored in the data base control file. These values are retrieved from the control file or altered as appropriate by the data management software.

Data Base Modeling

The process of describing and creating data bases that share ACCESSROUTINES is called modeling and is similar to conventional DASDL update of a single data base. The primary differences between modeling and conventional update are (1) update level does not change, (2) only dynamic parameters may be altered, (3) reorganization clauses are not allowed, and (4) the ACCESSROUTINES need not be re-compiled.

To model a data base after an existing data base, the DASDL compiler is provided with a description file and a DASDL description of a data base containing a <model option>. The compiler ensures that the description file and the DASDL source differ only in settings of dynamic parameters. If no errors are found, a new description file is created with the same update level as the original description file. The new description file can then be used to compile tailored data base software and to initialize the control and data files of the new data base.

The syntax of the <model option> is

```
--- MODEL <database name> -----|
```

When a <model option> is present, <database name> is used to form the title of the original description file. The name of the new description file is obtained from the code file name in the compile statement that invoked DASDL. When used, a <model option> is placed at the beginning of the DASDL source input. If a <model option> is present and the ZIP dollar option is set, the DASDL compiler zips compilation of all data base software except the ACCESSROUTINES.

A <update option> of the form

```
UPDATE;
```

may appear following a <model option>. When present, it indicates that the data files and control file already exist for the data base. UPDATE causes SYSTEM/DMCONTROL to be run to update the control file of the existing data base. If the <update option> is omitted, the SYSTEM/DMCONTROL run will initialize the control file of a new data base. If the DMCONTROL dollar option is reset, SYSTEM/DMCONTROL will not be run even if an <update option> is specified.

The example below shows the creation of a test data base, TESTDB, from a production data base called LIVEDB. The production data files reside on DISK, and the data base writes a duplicated audit to tape. The test data base exists on TESTPACK where it writes a non-duplicated audit. The DASDL source used to create LIVEDB was

```
OPTIONS (AUDIT);
D DATA SET (K NUMBER(6)) AREAS = 100;
S SET OF D KEY K;
RDS RESTART DATA SET (RINFO ALPHA(100));
AUDIT TRAIL (KIND = TAPE, DUPLICATED ON TAPE);
```

The following WFL job creates TESTDB from the description file of the production data base LIVEDB

```
?BEGIN JOB CREATE/TESTDB;
COMPILE TESTDB WITH DASDL;
COMPILER DATA
MODEL LIVEDB;
OPTIONS (AUDIT);
CONTROL FILE (PACK = TESTPACK);
D DATA SET (K NUMBER(6)) PACK = TESTPACK;
S SET OF D KEY K PACK = TESTPACK;
RDS RESTART DATA SET (RINFO ALPHA(100));
AUDIT TRAIL (PACK = TESTPACK);
?END JOB.
```

This DASDL compilation produces the file DESCRIPTION/TESTDB from the description file of the production data base, DESCRIPTION/LIVEDB. The update levels of both description files are equal. Because the ZIP dollar option was not explicitly reset, DASDL zipped the compilation of all tailored software for TESTDB except the ACCESSROUTINES. SYSTEM/DMCONTROL was run to initialize TESTDB's control file.

A user program compiled using DESCRIPTION/LIVEDB could open the production data base LIVEDB or the test data base TESTDB using data base equation.

User dollar options may be used to combine different sets of dynamic parameters into one DASDL source.

* ACCESSROUTINES Title

Because data bases with different <database name>s may use the same set of ACCESSROUTINES, the ACCESSROUTINES code file title may be specified in the DASDL source. This file title is stored in the control file and used to find the appropriate ACCESSROUTINES code file during the data base open process. The syntax of an <accessroutines specification> is

```
-- ACCESSROUTINES = <file title> -----|
```

where <file title> is a file name optionally including a <usercode> and ON <family name> part. If the ZIP dollar option is set and a <model option> was not present, DASDL will zip the compilation of the ACCESSROUTINES code file as <file title>.

The ability to explicitly name ACCESSROUTINES code files permits them to have any valid file name, i.e., their names are no longer restricted to ACCESSROUTINES/<database name>. Thus, a data base may now use an ACCESSROUTINES code file stored under a different usercode or on a pack family not normally visible to the user program. In addition, the update level of the ACCESSROUTINES could be stored in its file title to permit easier identification and management of ACCESSROUTINES code files. Regardless of its title, the data base open algorithm determines whether a particular file is an ACCESSROUTINES code file.

In the absence of an <accessroutines specification> the default ACCESSROUTINES code file title is ACCESSROUTINES/<database name>. If a <model option> is present and an <accessroutines specification> is omitted, the ACCESSROUTINES title is taken from the original description file.

Control File Usercode

The DASDL syntax for <control file> has been extended to permit specification of a usercode for the data base control file. The control file and all data and audit files will be stored under this usercode. SYSTEM/DMCONTROL must be run under this usercode or a privileged usercode to initialize or update a control file.

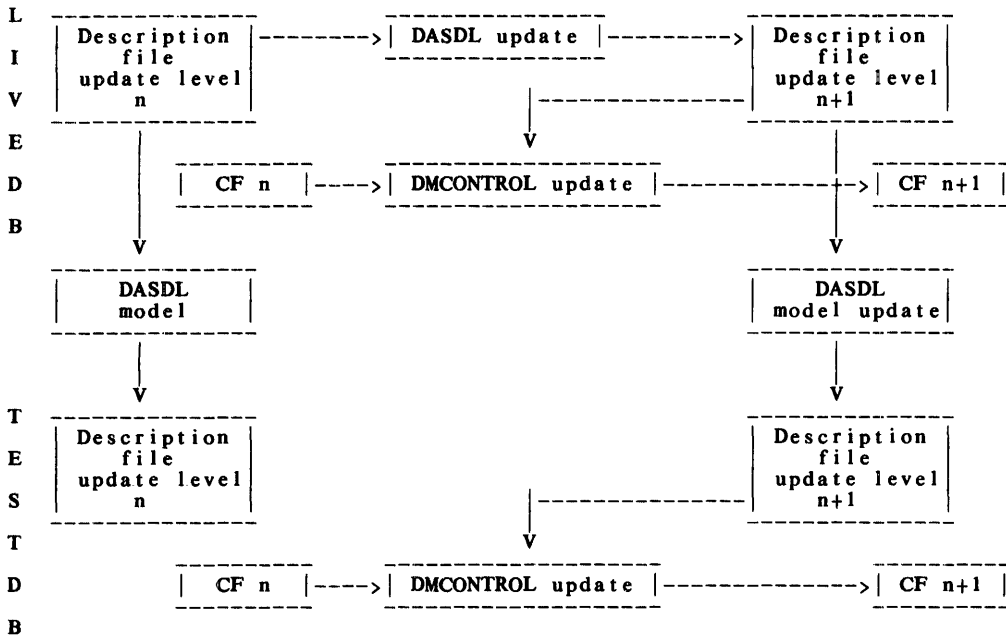
Update

Dynamic data base parameters may be changed during a conventional DASDL update as well as under the <model option>.

Data bases that share the same ACCESSROUTINES must be compatible. The following update procedures guarantee this.

One data base is selected as the original data base. DASDL is run to update the original data base's description file. New tailored software, including the ACCESSROUTINES, is compiled. For the remaining data bases, new description files cannot be created directly from old description files because their update status will not match the new ACCESSROUTINES. Instead, they must be re-created from the updated description file of the original data base. These new description files can then be used to compile the tailored software (excluding the ACCESSROUTINES) and update the control files of their respective data bases.

The following diagram illustrates the update process for two data bases that share one ACCESSROUTINES code file. Description and control files are shown for data bases LIVEDB and TESTDB at update levels n and n+1. TESTDB was created by modeling LIVEDB and did not exist prior to update level n.



LIVEDB was chosen as the original data base and updated first. LIVEDB's updated description file was then used to model an n+1 level description file for TESTDB. A SYSTEM/DMCONTROL update operation brought TESTDB's control file forward. At level n+1, both data bases use the ACCESSROUTINES code file created from LIVEDB's n+1 level description file. In practice, either LIVEDB or TESTDB could have been selected as the original data base in the update process.

There is no requirement that all data bases must be updated at the same time. For example, the updating of TESTDB from level n to n+1 could be performed at any time following the updating of LIVEDB. However, until updated to level n+1, TESTDB must continue to use the ACCESSROUTINES compiled using LIVEDB'S level n description file. The ACCESSROUTINES TITLE option in DASDL can be used to give unique names to each ACCESSROUTINES code file.

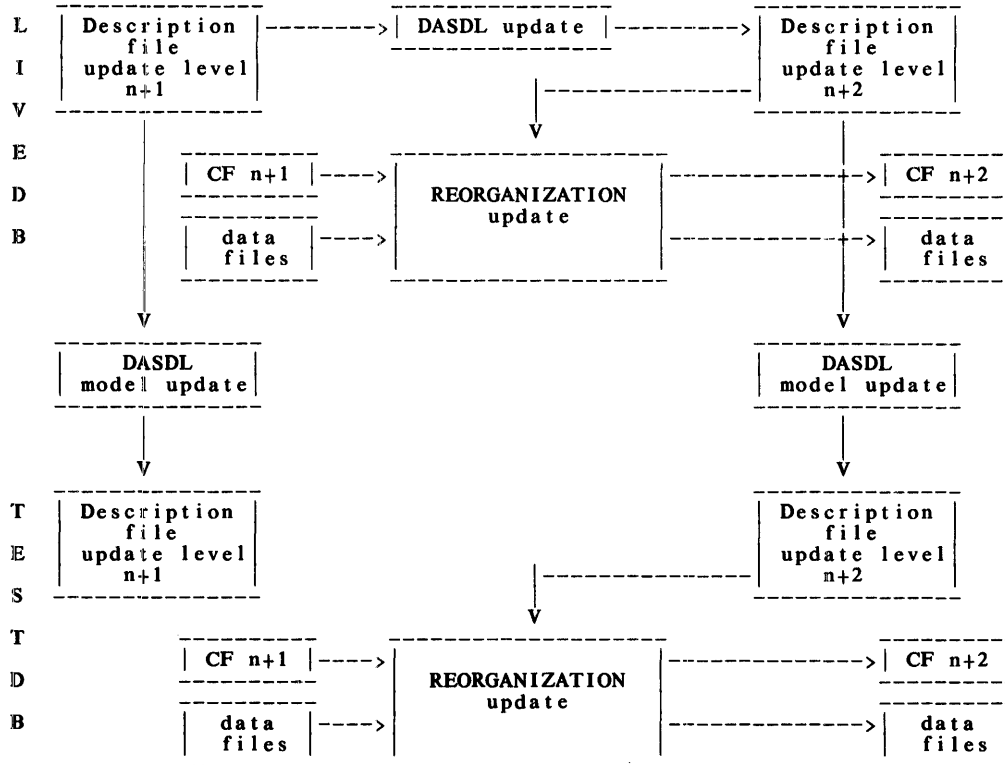
The following WFL job would update both TESTDB and LIVEDB adding the new dataset Q:

```

?BEGIN JOB UPDATE/BOTHDDBS;
%
%   UPDATE LIVEDB
%
%   COMPILER DATA
%   COMPILER DATA
%   UPDATE;
%   OPTIONS (AUDIT);
%   D DATA SET (K NUMBER(6)) AREAS = 100;
%   S SET OF D KEY K;
%   RDS RESTART DATA SET (RINFO ALPHA(100));
%   Q DATA SET (Z ALPHA(12));
%   AUDIT TRAIL (KIND=TAPE, DUPLICATED ON TAPE);
%   ?%----- END LIVEDB DASDL -----
%
%   UPDATE TESTDB
%
%   COMPILER DATA
%   COMPILER DATA
%   MODEL LIVEDB;
%   UPDATE;
%   OPTIONS (AUDIT);
%   CONTROL FILE (PACK = TESTPACK);
%   D DATA SET (K NUMBER(6)) PACK = TESTPACK;
%   S SET OF D KEY K PACK = TESTPACK;
%   RDS RESTART DATA SET (RINFO ALPHA(100));
%   Q DATA SET (Z ALPHA(12)) PACK = TESTPACK;
%   AUDIT TRAIL (PACK = TESTPACK);
%   ?%----- END TESTDB DASDL -----
%
%   ?END JOB.
  
```

Reorganization

Reorganization of modeled data bases proceeds in much the same manner as a DASDL update. The following diagram and WFL job illustrate the reorganization process for LIVEDB and TESTDB caused by increasing the length of Z in dataset Q from 12 to 20 characters:



```

?BEGIN JOB REORGANIZE/BOTHDBS;
%
%   REORGANIZE LIVEDB
%
COMPILE LIVEDB WITH DASDL;
COMPILER DATA
UPDATE;
OPTIONS (AUDIT);
D DATA SET (K NUMBER(6)) AREAS = 100;
S SET OF D KEY K;
RDS RESTART DATA SET (RINFO ALPHA(100));
Q DATA SET REORGANIZE(ITEMS CHANGED) (Z ALPHA(20));
AUDIT TRAIL (KIND = TAPE, DUPLICATED ON TAPE);
?%----- END LIVEDB DASDL -----
%
RUN SYSTEM/BUILDREORG;
FILE DASDL (TITLE = DESCRIPTION/LIVEDB);
DATA CARD
$ NOZIP
UPDATE;
?%----- END BUILDREORG INPUT -----
%
COMPILE REORGANIZATION/LIVEDB WITH DMALGOL LIBRARY;
COMPILER FILE CARD (KIND = DISK, TITLE = DATABASE/REORGSYMBOLIC);
COMPILER FILE DASDL (TITLE = DESCRIPTION/REORGANIZATION/LIVEDB);
%
RUN REORGANIZATION/LIVEDB("GENERATE");
%
RUN REORGANIZATION/LIVEDB("REMOVE");
%
%   REORGANIZE TESTDB
%
COMPILE TESTDB WITH DASDL;
COMPILER DATA
MODEL LIVEDB;
UPDATE;
OPTIONS (AUDIT);
CONTROL FILE (PACK = TESTPACK);
D DATA SET (K NUMBER(6)) PACK = TESTPACK;
S SET OF D KEY K PACK = TESTPACK;
RDS RESTART DATA SET (RINFO ALPHA(100));
Q DATA SET (Z ALPHA(20)) PACK = TESTPACK;
?%----- END TESTDB DASDL -----
%
RUN SYSTEM/BUILDREORG;
FILE DASDL (TITLE = DESCRIPTION/TESTDB);
DATA CARD
$ NOZIP
UPDATE;
?%----- END BUILDREORG INPUT -----
%
COMPILE REORGANIZATION/TESTDB WITH DMALGOL LIBRARY;
COMPILER FILE CARD (KIND = DISK, TITLE = DATABASE/REORGSYMBOLIC);
COMPILER FILE DASDL (TITLE = DESCRIPTION/REORGANIZATION/TESTDB);
%
RUN REORGANIZATION/TESTDB("GENERATE");
%
RUN REORGANIZATION/TESTDB("REMOVE");
%
?END JOB.

```

Note that the DASDL source used to create TESTDB's level n+2 description file has no REORGANIZE clause for dataset Q. The reorganization information from LIVEDB's n+2 description file was copied into TESTDB's n+2 description file by the DASDL compiler during the modeling operation. When a <model option> is present, DASDL will generate a syntax error for any reorganization cards found in the DASDL source.

Recovery

For the most part, all forms of recovery will continue to operate as before. However, the values of dynamic parameters stored in the control file will not be changed by rollbacks or rebuilds. At the completion of a rollback or rebuild, dynamic parameter settings will be the same as those at the beginning of the operation.

Other DMSII Software

The operation of the remaining DMSII software has not been changed to support shared ACCESSROUTINES and data base equation. For example, RECOVERY/LIVEDB can be used to recover LIVEDB but not TESTDB. TESTDB can be recovered by RECOVERY/TESTDB but not by RECOVERY/LIVEDB.

New DASDL Syntax

The DASDL syntax diagram for <database> is extended as follows to support the data bases that share ACCESSROUTINES (page 4-1, DMSII DASDL Reference Manual, form no. 5001480):

<database>

```

> | -<model option>-- ; - | | -<update option>-- ; - |
>-----
> | | <-----
> | | -----/1\->defaults>----- ; -----
> | | |
> | | | -/1\-<initialize>-----
> | | | -/1\-<options>-----
> | | | -/1\-<parameters>-----
> | | | -/1\-<accessroutines specification>-
> | | <-----
> | | -----/1\-<audit trail>-----
> | | |
> | | | -/1\-<control file>-----
> | | | -<data set>-----
> | | | -<global item>-----
> | | | -<physical specifications>-
> | | | -<set-subset-access>-----
> | | | -<logical data base>-----
> | | | -<remap>-----

```

The new DASDL syntax for <model option> is:

<model option>

```
-- MODEL <database name> -----
```

The new DASDL syntax for <accessroutines specification> is:

<accessroutines specification>

```
-- ACCESSROUTINES = --<file title>-----
```

The DASDL syntax for <control file> has been modified to allow specification of a usercode for the data base control file (page 4-32, DMSII DASDL Reference Manual, form no. 5001480):

<control file>

```
-- CONTROL FILE ----- ( <control file attributes> ) -----
```

<control file attributes>

```

<-----,-----
|                                     |
-----/1\-- <disk/pack> -----
|                                     |
---/1\-- <usercode specification> ---

```

<disk/pack>

DISK	
-- KIND = --	
DISKPACK	
-- PACK -----	-- = <family name> -----

<usercode specification>

Data base equation is operationally similar to file equation. WFL equation overrides specification of a data base title in the host language declaration, and run time modification of a data base title overrides both WFL equation and source language specifications. However, data base equation differs from file equation in that a runtime error will result if a user program attempts to set or examine the TITLE attribute while the data base is open.

<database title>

----- <identifier> ----- - * ----- - (<usercode>) -	----- - ON <family name> - -----
---	--

Algol

```
-- DATABASE --<database reference>-----
```

----->

```

|_<internal name>-- = -| |<logical data base name>-- OF -|

```

```
>-<database name>----->
```

```

- (TITLE = <database title>) -

```

> _____

```

| |<----- , -----| |
| - : -----<data set reference>-----|
| |<set reference>-----|

```


The following examples show how the data base TITLE attribute can be manipulated during program execution. Note that <database title>s never end with a period and that a <replace pointer-valued attribute statement> is not valid for data base TITLES.

```
--<internal name>.TITLE -- := --<database title>-----|
--<string variable>-- := -- <internal name>.TITLE -----|
```

Cobol

<database declaration>

```
-- DATA-BASE SECTION. -----|
|                               |
| |<-----|
| |-----<database reference>-----|
```

<database reference>

```
-- DB -----<db spec>-----|
|                               |
| |<internal option>-----| |
| |                               |
| | |<logical db option>-----|
| | |<logical db option>-----|
```

<db spec>

```
--<database name>----->
| - GLOBAL - |
```

```
>--- ALL ---<db attribute>-----|
|                               |
| |<-----| | |
| |                               |
| | |<-----|
| | |<db attribute>-----|
| | |                               |
| | | |<data set reference>-----|
| | | |<set reference>-----|
```

<db attribute>

```
---- VALUE OF TITLE IS --<database title>-----|
```

The following CHANGE and MOVE statements can be used to manipulate the data base TITLE attribute during program execution:

```
-- CHANGE ATTRIBUTE TITLE --- OF --- <internal name> -- TO ----->
|                               |
| | - IN - |
```

```
--- <database title> -----|
| |<alphanumeric data-item> - |
```

```
-- MOVE ATTRIBUTE TITLE --- OF --- <internal name> -- TO ----->
|                               |
| | - IN - |
```

```
-- <alphanumeric data-item> -----|
```

PL/I

```

<database declaration>
-- DATABASE --<database designation>--<DM attribute list>-----|
<DM attribute list>
----->
| - ENVIRONMENT --- ( <db attribute list> ) - |
| - OPTIONS -----|
| <-----|
|-----|
| --/1\-<data set reference>--|
| --/1\-<set reference>-----|
| --/1\- EXTERNAL -----|
<db attribute list>
-- TITLE = <database title> -----|
The following syntax may be used to manipulate the data base TITLE attribute during program
execution:
-- TITLE(<internal name>) -- = --<database title>-----|
--<character variable>-- = -- TITLE(<internal name>) -----|

```

Host Language Examples

In each of the examples below, the first OPEN statement will open LIVEDB whose data and control files are stored under the user's disk directory. The second OPEN statement invokes TESTDB whose files are stored on TESTPACK under the usercode UC.

Algol

```

-----
BEGIN
  DATABASE MYDB (TITLE="LIVEDB");
  OPEN UPDATE MYDB;
  ...
  CLOSE MYDB;
  MYDB.TITLE := "(UC)TESTDB ON TESTPACK";
  OPEN UPDATE MYDB;
  ...
  CLOSE MYDB;
END.

```

Cobol

```

-----
IDENTIFICATION DIVISION.
...
DATA-BASE SECTION.
  DB MYDB ALL
  VALUE OF TITLE IS "LIVEDB".
...
OPEN UPDATE MYDB.
...
CLOSE MYDB.
CHANGE ATTRIBUTE TITLE OF MYDB TO "(UC)TESTDB ON TESTPACK".
OPEN UPDATE MYDB.
...
CLOSE MYDB.
STOP RUN.

```

PL/I

```

EQUATE DB:  PROCEDURE;
            DATABASE MYDB ENVIRONMENT(TITLE='LIVEDB');
            OPEN MYDB OPTIONS(UPDATE);
            .
            .
            .
            CLOSE MYDB;
            TITLE(MYDB) = '(UC)TESTDB ON TESTPACK';
            OPEN MYDB OPTIONS(UPDATE);
            .
            .
            .
            CLOSE MYDB;
END EQUATE_DB;

```

WFL Syntax

A <database title> in a host language <database declaration> may be overridden via WFL during either program compilation or execution. The syntax of a WFL data base equation statement is

```
-- DATABASE <internal name> ( TITLE = <database title> ) -----|
```

The following example illustrates the use of WFL data base equation:

```

?BEGIN JOB WFL/EXAMPLE;
%
%   OVERRIDING EQUATION DURING COMPILATION
%
COMPILE USER/PROGRAM WITH BDMSALGOL LIBRARY;
DATABASE CTIMEDB (TITLE = (UC)TESTDB ON TESTPACK);
COMPILER DATA
BEGIN % ALGOL SOURCE
    DATABASE CTIMEDB;
    DATABASE MYDB = RTIMEDB;
    DATABASE EQUATEDDB (TITLE = "LIVEDB");
    OPEN CTIMEDB;
    .
    .
    .
    CLOSE CTIMEDB;
    OPEN MYDB;
    .
    .
    .
    CLOSE MYDB;
    OPEN EQUATEDDB;
    .
    .
    .
    CLOSE EQUATEDDB;
END.
?%----- END ALGOL INPUT -----
%
%   OVERRIDING EQUATION DURING EXECUTION
%
RUN USER/PROGRAM;
DATABASE MYDB (TITLE = SOMEOTHERDB);
DATABASE EQUATEDDB (TITLE = (UC)HISDB ON THEIRPACK);
?END JOB.

```

Data base equation is recognized only by new WFL. Pre-29 WFL will not accept data base equation statements.

CANDE Syntax

On the Mark 32 software release, data base equation may be used through CANDE only in a WFL statement using new WFL syntax. For example,

```
RUN DB/PROGRAM; DATABASE LIVEDB (TITLE = (UC)TESTDB ON TESTPACK)
```

and

```

WFL RUN OBJECT/DB/PROGRAM; %
##
DATABASE LIVEDB (TITLE = (UC)TESTDB ON TESTPACK)

```

will not work, but

```

WFL BEGIN JOB; RUN OBJECT/DB/PROGRAM; %
##
DATABASE LIVEDB (TITLE = (UC)TESTDB ON TESTPACK); END JOB.

```

will produce the desired results. The "END JOB." is optional.

USERCODES AND FILE SECURITY

Both ACCESSROUTINES code files and data base control files are subject to normal file security rules on Mark 32 software. These files should have their SECURITYTYPE attribute set to PUBLIC when they must be visible to user programs running under various usercodes.

The following chart shows special actions that are necessary to allow a user program running under usercode X to open a data base whose control file and ACCESSROUTINES code file are stored under various combinations of usercodes. This chart does not apply to user programs running under privileged usercodes or to data bases protected by guardfiles.

ACCESSROUTINES Code File Usercode

	(X)	(Z)	*
C o n t r o l F i l e U s e r c o d e	(X)	None	.Specify ACR title in DASDL .ACR security PUBLIC .ACR security PUBLIC
	(Y)	.Data base equation or specify CF usercode in DASDL .CF security PUBLIC	.Specify ACR title in DASDL .Data base equation or specify CF usercode in DASDL .ACR security PUBLIC .CF security PUBLIC
		.CF security PUBLIC	.CF security PUBLIC .ACR security PUBLIC .Specify ACR title in DASDL

D3198 DMSII - UPDATING FROM MARK "31" TO MARK "32 DMSII"

Mark 32 DMSII software may be used with the Mark 31 MCP on the B5000, B6000 or B7000. B5000 or B6000 systems must use the Mark 31 PR1 or later MCP; B7000 systems must use the B7000 Mark 30 PRO or later MCP.

Some new features of the Mark 32 DMSII software require MCP support which is not provided on the Mark 31 MCP. The following features are only available when a Mark 32 or later MCP is used:

- Data base equation requires Mark 32 MCP and WFL. The Mark 31 MCP supports the new Mark 32 data base stack structure; however, it does not recognize WFL data base equation.
- Swapping of data bases is only available on the Mark 32 MCP.

The following procedure may be used to update to the Mark 32 DMSII release.

- Copy and compare the Mark 31 DASDL source, description file and system software to tape using Library Maintenance.
- Dump the data base files to tape using the Mark 31 UTILITY program.
- Load the Mark 32 DMSII software and perform DASDL update. In order to ensure an easy return to Mark 31, do not make any changes to the DASDL description of the data base during the update. Following successful update, Mark 32 system software will automatically be compiled.
- As a precaution, the data base files may be dumped using Mark 32 UTILITY.
- User programs will continue to run on Mark 32 without being recompiled.

The following procedure may be used to return to Mark 31 software provided no changes were made to the DASDL description of the data base during DASDL update.

- Dump the data base files to tape using Mark 32 UTILITY.
- Backup copies of the audit may be made using COPYAUDIT.
- Reload the Mark 31 DASDL source, description file and system software from tape.
- Perform a RECOVER UPDATE of the control file using DMCONTROL. This will transfer the information from the Mark 32 control file to the Mark 31 control file. Do not RECOVER INITIALIZE the control file.

5. User programs compiled with Mark 32 compilers must be recompiled with Mark 31 compilers before they are run with Mark 31 ACCESSROUTINES.

Several D-Notes contain information which is especially useful for conversion to Mark 32 DMSII.

a. DMSII GENERAL D3108

The data base stack structure has been changed to support Shared ACCESSROUTINES.

b. DMSII GENERAL D3170

Shared ACCESSROUTINES and data base equation are now available. With the introduction of these features, the data base open mechanism has been changed.

c. WFL/COPYAUDIT D3270

The ACCESSROUTINES now initiate copyaudit via a job file titled DATABASE/WFL/COPYAUDIT.

d. BUILDREORGANIZATION D3084

Database Reorganization has been simplified.

e. REORGANIZATION D3120

Reorganization has been significantly enhanced.

f. INTERFACE D3229

A single program may now invoke data bases compiled with software at different release levels.

D3288 DMSII - MARK "31" SYSTEM NOTES CORRECTIONS

The system notes for the Mark 31 release contain errors, corrections for which are described below.

"System File Page" indicates the page number in the file SYSTEMNOTES/REL310 contained on tape SYSTEMNOTES310; "Page" indicates the page number in B6000 Series System Notes (Form No. 5011257).

DMSII	D2868	System File Page	259
		Page	159

Some DMSII Openererrors, which were not documented, are described as follows:

Openererror

-
- "40 User program must be recompiled on a more recent release."
 - "41 Update level timestamp mismatch: User program and ACCESSROUTINES compiled with different description files."
 - "42 Insufficient memory for SIB stack image."
 - "43 Open Initialize not allowed by DASDL option."

DMSII	D2965	System File Page	261
		Page	160

Under the procedure for returning to 30 software, paragraph 5 should be replaced by the following:

- "5. User programs compiled with 31 compilers must be recompiled with 30 compilers before they are run with 30 ACCESSROUTINES."

BUILDREORG	P2477	System File Page	308
		Page	189

The fault caused by the error should read "SEG ARRAY", rather than "INVALID INDEX".

Appendix A Data Dictionary System File Page A-6
Page A-4

Line 600 of the Example for DDINITIALIZE should read as follows:

600 COMPILER FILE TAPE=*DATADICT/DDINITIALIZE;

D3302 DMSII - EFFICIENT USE OF INDEX SEQUENTIAL

Appendix J of the DMSII Host Reference Manual (Form No. 5001498) should be replaced by the following:

The <key condition>s which result in efficient searches at run-time are those which result in a binary search or at least a partial binary search. For example, given an index sequential set S with key = (K1,K2,K3), and the following <key conditions>s:

S AT K1 = X1 AND K2 = X2 AND K3 <relation> X3

S AT K1 = X1 AND K2 <relation> X2

S AT K1 <relation> X1

The conditions for optimization of <key condition>s are:

- The <key item>s specified must include the highest order <key item> with no <key item>s omitted between the most and least significant <key item>s specified.
- All <key condition>s on individual <key item>s are connected by "AND".
- With the exception of the least significant <key item>, all <key condition>s must be in the form:

<key item> = <expression>

- The <relation> specified in the least significant <key condition> must be in one of the forms listed below:

FIND Type	Final Key	<relation>
-----	-----	-----
FIND NEXT	Ascending	=,>,>=
FIND NEXT	Descending	=,<,<=
FIND PRIOR	Ascending	=,<,<=
FIND PRIOR	Descending	=,>,>=

If all the preceding conditions are satisfied, then at least a partial binary search is performed.

NOTE

Even though "FIND NEXT S AT <key condition>" is efficient, it is not as efficient as specifying "FIND NEXT S".

Since the "FIND KEY OF" form will never result in an I/O on the data set, it can often be used to great advantage. Often, the record may be selected or rejected based only on information in the key entry (including key data). If the record is to be selected, "FIND S" will retrieve the rest of the data items from the data set. Although key data can be used advantageously via "FIND KEY OF", key data items used in a <key condition> cause a linear search on the index, and should be avoided.

It is possible that a more complex <selection condition> is desired than is optimized. In this case, it is best to extract some of the logic from the <selection condition> and code it as tests in the user program. "FIND KEY OF" can often save time if a fair percentage of the records which satisfy the optimized <selection expression> will be rejected.

A range test is typical example. Suppose all records with K1 = X1 and K2 = X2 and K3 >= R1 and K3 <= R2 are desired. Then, the <selection expression>

NEXT S AT K1 = X1 AND K2 = X2 AND K3 >= R1

can be used, and the processing of records stopped when the value of K3 exceeds R2.

D3314 DMSII - REBUILD ACROSS FILE DISCONTINUITIES

Rebuilds can now be done across file discontinuities, even when no dump of the newly-initialized file is present.

File discontinuities are caused when one or more data base files are initialized via UTILITY.

When a UTILITY INITIALIZE is performed, all changes to the data base are audited. As a result, RECOVERY can rebuild through a UTILITY INITIALIZE even when the structure is not dumped following initialization.

D3333 DMSII - MARK "31" SYSTEM NOTES CORRECTIONS

The system notes for the Mark 31 release contain errors, corrections for which are described below.

"System File Page" indicates the page number in the file SYSTEMNOTES/REL310 contained on tape SYSTEMNOTES310; "Page" indicates the page number in B6000 Series System Notes (Form No. 5011257).

Appendix C	TPS	System File Page	C-28
		Page	C-17

The syntax diagram for <transaction journal> should not contain parentheses around <control file attributes> and <data file attributes>

System File Page	C-78
Page	C-45,46

The ALGOL procedure UPDATE uses COBOL syntax in the two CASE statements of the procedure skeleton. These two case statements should read:

```
CASE TRIN.TRFORMAT OF . . .
and
CASE TRIN.TRSUBFORMAT OF . . .
```

System File Page	C-100
Page	C-58

The syntax diagram which describes the SEARCH specification should be replaced by the following diagram which clearly shows that RANGE, USERS and SELECT must each be specified, but the order of specification is irrelevant.

```

-- SEARCH --<journalid>-- , -----|
                        |<-/2*\----- , -----|
                        |-----/1\- RANGE <range options> -----|
                        |-----/1\- USERS <user option> -----|
                        |-----/1\- SELECT <tr options> -----|

```

Note: The "*" above the 2 implies that the path must be traversed a minimum of two times.

D3337 DMSII - DATA BASE STACK

A stack cell containing software version information has been added to all DMSII software. This word can be used to determine the software level of DMS software products from PROGRAMDUMP listings. In addition, the data base stack also contains a cell indicating the dollar option and DASDL options used during ACCESSROUTINES compilation.

D3382 DMSII - AUDITING FOR HALT/LOAD AND ABORT RECOVERY

As soon as possible after a new audit file is opened, the ACCESSROUTINES force two control points. Consequently, abort and Halt/Load recovery rarely have to back up beyond the beginning of the current audit file. Old audit files are usually archived for use in reconstructs, rollbacks, and rebuilds but are not required for either abort or Halt/Load recovery.

In certain DMSII operating environments, rollbacks, rebuilds, and reconstructs may be extremely infrequent. This would be particularly true when the entire data base is dumped to tape frequently. Alternatively, test or other small data bases may be more easily re-created from scratch than through a rebuild. In such situations there is no need to archive old audit files. The current audit file is sufficient to protect these data bases against system Halt/Loads and user program aborts.

The following steps can be taken to automatically remove audit files once they have been closed. Compile the data base with either the COPY or VERIFY option specified for its primary audit and the secondary audit if one exists. The default title of the COPYAUDIT WFL deck is overridden as described in WFLCOPYAUDIT note D3270 and a user-written WFL job is zipped instead of the standard COPYAUDIT deck (DATABASE/WFL/COPYAUDIT).

The following WFL job illustrates how a user-written job substituted for the standard COPYAUDIT WFL job can be used to remove audit files.

```
?BEGIN JOB REMOVEAUDIT (STRING PARMS, STRING SUBSYS);
  TASK REMOVETASK;
  STRING FILENAME, SCRATCH;
EBCDIC DATA/FILEREMOVER
$ SET LEVEL 2 LINEINFO LIST
PROCEDURE FILEREMOVER(FILENAME);
  ARRAY FILENAME[*];
BEGIN
  BOOLEAN RSLT;
  REAL I;
  STRING FILETITLE;
  DEFINE ERRORTYPE = [39:20] #;
  SCAN POINTER(FILENAME) FOR I:256 UNTIL = 48"00";
  I := 256 - I;
  REPLACE POINTER(FILENAME)+I BY "."; % TRAILING PERIOD
  FILETITLE := STRING(POINTER(FILENAME), I-1);
  RSLT := REMOVEFILE(POINTER(FILENAME));
  IF RSLT THEN
    BEGIN % FILE REMOVAL FAILED
      CASE REAL(RSLT.ERRORTYPE) OF
        BEGIN
          10:   DISPLAY("ERROR IN AUDIT FILE TITLE = " CAT FILETITLE);
          30:   DISPLAY("AUDIT FILE " CAT FILETITLE CAT " NOT REMOVED");
          ELSE: DISPLAY("ERROR " CAT STRING(REAL(RSLT.ERRORTYPE),*));
          END; % ERROR CASES
        MYSELF.STATUS := VALUE(TERMINATED);
      END;
      DISPLAY ("AUDIT FILE " CAT FILETITLE CAT " REMOVED");
    END.
  ? %--- END FILEREMOVER SOURCE ---

  ON RESTART,
    GO TO STARTOVER;
```

STARTOVER:

```
% EXTRACT THE AUDIT FILE TITLE FROM THE COPYAUDIT PARAMETER
% BUILT BY THE ACCESSROUTINES
```

```
IF TAKE(PARMS,7) = "VERIFY " THEN
  SCRATCH := DROP(PARMS,7)
ELSE
  IF TAKE(PARMS,5) = "COPY " THEN
    SCRATCH := DROP(PARMS,5)
  ELSE
    ABORT "UNEXPECTED PARAMETER FORMAT: " & PARMS;
    SCRATCH := TAIL(SCRATCH," ");
    FILENAME := HEAD(SCRATCH,NOT " ");
    SCRATCH := DROP(SCRATCH,LENGTH(FILENAME));
    SCRATCH := TAIL(SCRATCH," ");
    IF TAKE(SCRATCH,3) NEQ "ON " THEN
      ABORT " ON <FAMILYNAME> EXPECTED: " & SCRATCH;
    SCRATCH := DROP(SCRATCH,3);
    SCRATCH := TAIL(SCRATCH," ");
    FILENAME := FILENAME & " ON " & HEAD(SCRATCH,ALPHA);
```

```
IF FILE FILEREMOVER ISNT RESIDENT THEN
  BEGIN
    REMOVETASK (STATUS = NEVERUSED);
    COMPILE FILEREMOVER ALGOL [REMOVETASK] LIBRARY;
    ALGOL FILE CARD (TITLE = DATA/FILEREMOVER);
    SUBSYSTEM = #SUBSYS;
    OPTION = (FAULT, DSED, ARRAYS);
    IF REMOVETASK ISNT COMPILEDOK THEN
      ABORT "FILEREMOVER COMPILATION FAILED";
  END;
```

```
REMOVETASK (STATUS = NEVERUSED);
RUN FILEREMOVER(FILENAME) [REMOVETASK];
IF REMOVETASK ISNT COMPLETEDOK THEN
  ABORT "AUDIT FILE REMOVAL FAILED";
```

?END JOB.

D3460 DMSII - PREALLOCATION OF DIRECT DATA SETS

In some processing environments it may be necessary to add new records to the end of a direct data set with a minimum of overhead. In particular, if a new record might be added that has a key value significantly larger than the previous last record, it may be undesirable to wait while all blocks between the previous last block and the new last block are initialized before adding the new record. The overhead can be avoided by preallocating at a convenient time all of the required data blocks by storing a record with a large key value. However, this requires that a dump be taken of all preallocated rows to reconstruct any one of them.

The new facility described below allows a fixed number of records to be preallocated when a direct data set is initialized via UTILITY. When the ACCESSROUTINES add a record anywhere within this preallocated area, no run time initialization is necessary. In addition, when the structure is dumped via UTILITY, the preallocated areas are not dumped, but information is recorded that allows the structure to be reconstructed or rebuilt.

1. Initializing Direct Data Sets

The UTILITY syntax for the INITIALIZE option, described on page 4-24 of the DMSII Utilities and Operations Guide (Form No. 5001803) has been extended as follows to permit specification of the number of records to preallocate:

```
-- INITIALIZE --- = -----
|                                     |
| |<-----, -----|
| |<structure name> -- <initialize options> ---|
|                                     |
<initialize options>
-----
| - ( -- PREALLOCATE -- <integer> -- ) - |
```

<integer> specifies the number of records to be preinitialized.

If a preallocation value is specified, UTILITY will write null records (direct key of all 4"F"s) into the specified number of record locations. For preallocated direct data sets, word 0 of block 0 is used to point beyond the last record placed in the data base via a host language STORE statement. This value is called DATAEOF. LASTRECORD points beyond the last preallocated record. Note that DATAEOF must be less than or equal to LASTRECORD. After UTILITY has preallocated the data set, DATAEOF is set to the address of block 1, and the control file is marked to indicate that this is a different format direct data set from a normal direct data set.

2. ACCESSROUTINES Use of Preallocated Direct Data Sets

As a result of the preallocation feature for direct data sets, the ACCESSROUTINES audit a new record type (STRDC) whenever a structure is initialized. One STRDC record is audited for each initialized structure. STRDC records for direct data sets contain the preallocation value, if specified. The DATAEOF value obtained from a preallocated direct data set is checked for consistency at open time. If the value is inconsistent, DATAEOF is set to F.LASTRECORD+1 and its new value is audited.

A new record stored into the preallocated portion of a direct data set causes DATAEOF to be updated and the new value audited. The new DATAEOF value is the address of the first block beyond the newly added record. Storing a new record beyond the actual end of file causes the ACCESSROUTINES to initialize new file areas. Following such physical extension of the file, preallocated and conventional direct data sets are treated similarly by the ACCESSROUTINES.

3. UTILITY Dump of Preallocated Direct Data Sets

When UTILITY dumps all or a portion of a preallocated direct data set, the DATAEOF value is recorded in the tape directory. Only that portion of the direct data set before the DATAEOF block is actually written to the dump tape. If for some reason the DATAEOF value is unavailable (perhaps row 0 is locked out), all data set rows, including the preallocated ones, are dumped.

4. UTILITY Reload of Preallocated Direct Data Sets

If UTILITY needs to rebuild, reconstruct or copy direct data set rows that were in the preallocated region at the time of the dump, UTILITY simulates loading of these rows by preallocating them. In effect, preallocated rows appear to have actually been written to the dump tape.

5. File Discontinuities Encountered during Rebuild

When rebuilding through a UTILITY initialize of a direct data set with preallocated records, a STRDC audit record causes RECOVERY to mimic the UTILITY initialize by preallocating the appropriate records of the direct data set.

6. Recovering the Control File

Whenever a direct data set is preallocated, its format level in the control file is set to 1, indicating that word 0 of block 0 contains a DATAEOF value. If the control file is lost or destroyed, it should be recovered using a RECOVER UPDATE request to DMCONTROL and supplying a control file that has the correct format level for the direct data set. If a control file is used that indicates a normal (format level 0) direct data set when it is in fact format level 1, all software will work correctly except that UTILITY will dump both conventional and preallocated rows of the file.

D3548 DMSII - "DBS" IN LOCAL MEMORY VS. NONEXCHANGED UNIT

If a Mark 31 or earlier data base has a SUBSYSTEM specified, and the processor selected from that subsystem has no path to the units containing the ACR codefile, any program attempting to open that data base will now get a DMOPEN error #53.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - GENERAL

--- -- - -----

P3071 DMSII - "READLOCKNOPURGE" REMOVED

READLOCKNOPURGE was used improperly in some instances. It is now properly used in software where performance is an issue, and where only global variables are used. The ACCESSROUTINES are the only software that requires READLOCKNOPURGE for locking global variables and exchanging globals.

P3258 DMSII - NORMAL VS. DIRECT FILES AS PARAMETERS

The DMALGOL compiler was allowing direct files to be passed through formal parameters declared to be regular files. Similarly, regular files could be passed through direct file formal parameters. The compiler now generates warnings for each such occurrence. Consequently, DMSII software now passes only direct files through direct file formal parameters and regular files through regular file parameters.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - ACCESSROUTINES

D3044 ACR - COUNT FINDS AGAINST INDEX SETS

FIND/LOCK requests using an index are now counted in the data base usage statistics. FIND/LOCK requests that retrieve data set records (instead of FIND/LOCK KEY OF) will count as a find against the index and a find against the data set.

In addition, statistics are now collected on global data set activity.

D3045 ACR - STATISTICS INTERFACE

A mechanism has been implemented via the DMINQ interface to the ACCESSROUTINES by which statistics information can be retrieved from an active data base. Data can be retrieved concerning the data base as a whole or individual structures. Both dynamic statistics and static data can be independently retrieved.

Statistics information is retrieved by passing a statistics request in an array via the DMINQ interface to the ACCESSROUTINES. The format of the statistics request is as follows:

Word 0 - =25
Word 1 - <structure number>
Word 2 - <type statistics>

Word 0 specifies a statistics request to the DMINQ interface.

Word 1 indicates the structure number for which statistics are desired or equals 0 if global statistics are desired.

Word 2 specifies whether static or dynamic data is desired. A value of zero indicates static statistics are desired; a value of one indicates dynamic statistics are desired.

The ACCESSROUTINES will process the request and will return the result in the array starting at word 3. Words 0 thru 2 will be unaffected by the ACCESSROUTINES. The format of the result is as follows:

Word 3 - statistics result word
Word 4 - total number of words returned
Word 5 - index to header word for sub-group 1
Word[Word 5] - header word for sub-group 1
Word[Word 5 + 1] - first data word for sub-group 1
.
.
Word[Word 5 + n] - header word for sub-group 2
.
.
Word[Word 4 - 1] - end of statistics flag

Word 3 indicates the result of the statistics request. If the request was correctly formatted and honored, this word will be zero. Otherwise, [0:1] will be equal to 1 and [35:8] will contain an error category. The currently defined error categories are as follows:

- 1 - the structure number provided in word 1 did not correspond to an existing data set or set in the data base.
- 2 - the statistics request type provided in word 2 was invalid (not 0 or 1).
- 3 - a fault was encountered while retrieving statistics.

Word 4 contains the total number of words returned in the array including all of the fixed words at the front of the array.

Note: If the array provided is too small to receive all of the statistics, it will be resized. The array passed to INQUIRY must not be a segmented array.

Word 5 contains the index of the first group of statistics information returned. Each group is preceded by a group header word indicating the type of the group and the number of words of information in the group.

The layout of this header word is as follows:

[47:24] - not used
[23:8] - group type
[15:16] - number of words in group (including header)

The last group of statistics is followed by an end-of-statistics header which has a group type of zero (0).

The various group types and their layouts are as follows:

I. Global Static Statistics (word 1 = 0, word 2 = 0)

Group type 6

Word	-	Contents
1		Time data base opened (TIME(7) value)
2		Maximum valid structure number in data base
3		Data base options
		[0:1] = 1 if statistics set in data base
		[1:1] = 1 if data base is audited
		[2:1] = 1 if lockstatistics set in data base

Group type 5

Word	-	Contents
1-n		data base name including usercode prefix, if any. data base name is followed by 4"00".

II. Global Dynamic Statistics (word 1 = 0, word 2 = 1)

Group type 1

Word	-	Contents
1		Current data base open count
2		Current number of users that have data base open for update
3		Current data base open state
		0 = data base not open
		3 = data base is open
		4 = data base is undergoing recovery
4		Maximum total buffer space in words
5		Current total buffer space in words
6		Current allowedcore value
7		Maximum number of buffers allocated

Group Type 2 (present only if STATISTICS is set)

Word	-	Contents
1		Number of forced data base overlays
		Forced overlays result when total buffer space exceeds allowed core.
2		Number of normal data base overlays
		Normal overlays result when structures are closed or user program change from serial to random or random to serial state.
3		Time statistics collections started or was last reset (TIME(7) value)

Group Type 3 (present only if STATISTICS is set and data base is audited)

Word	-	Contents
1		First audit file number
2		Current audit file number
3		Starting audit block serial number
4		Current audit block serial number
5		Average number of words used in audit blocks
6		Actual audit block size
7		Number of audit I/O's initiated
8		Total wait time accumulated on primary audit in ticks
9		Total wait time accumulated on secondary audit in ticks
10		Total transaction count
11		Total number of times processes where held up at BEGIN-TRANSACTION
12		Total time spent waiting at BEGIN-TRANSACTION in

	ticks
13	Total number of sync points taken
14	Total number of control points taken
15	Total time spent taking control points in ticks
16	Sum of the number of buffers present at each control point
17	Sum of the number of buffers flushed at each control point

III. Structure Static Statistics (word 1 = <structure number>,
word 2 = 0)

Group Type 7

Word	Contents
1	Structure number
2-4	Structure name (first byte is length in binary)
5	Structure type 2 = data set 5 = index set
6	Structure subtype (see PROPERTIES 10068000- 10094000)
7	Structure nesting level (1 = disjoint)
8	Structure block factor (in records for data sets, in key entries for index sets)
9	Structure physical block size in words (including integrity checking words)
10	Structure area size in sectors
11	= 1 if structure is checksummed
12	= 1 if structure is address-checked

IV. Structure Dynamic Statistics (word 1 = <structure number>,
word 2 = 1)

Group Type 8

Word	Contents
1	Current number of random access users
2	Current number of serial access users
3	Current number of buffers allocated for structure
4	Current number of big buffers allocated for structure

Group Type 11 (present only if STATISTICS is set)

Word	Contents
1	Number of physical reads against structure
2	Number of physical writes against structure
3	Number of ticks spent waiting for writes to complete
4	Number of ticks spent waiting for reads to complete
5	Total amount of I/O time accumulated on file
6	Number of read-aheads issued against structure
7	Number of write-aheads issued against structure

Group Type 9 (present only if STATISTICS is set and structure is
a data set)

Word	Contents
1	Number of finds against data set
2	Number of Create/Store's against data set
3	Number of Modify/Store's against data set
4	Number of Deletes against data set
5	Number of times control information changed

Group Type 10 (present only if STATISTICS is set and structure is
an index set)

Word	Contents
1	Number of Finds against index set
2	Number of inserts in index set
3	Number of key data changes in index set
4	Number of key deletions from index

Notes:

"ticks" refers to ticks of the processor clock at 2.4 micro-seconds/tick.

Groups may be returned in the array in any order.

EXAMPLE PROGRAM

The following program illustrates the mechanism by which statistics can be retrieved. This program writes all results to a file for later off-line analysis.

```

BEGIN
  DATABASE DB;
  FILE DBSTATISTICS(KIND=DISK, FILETYPE=4, SIZEOFFSET=4,
                    BLOCKSIZE=900);
  ARRAY A[0:4]; % WILL BE RESIZED BY ACCESSROUTINES
  REAL MAXSTRNO;
  REAL I;

  DEFINE
    DMINQCASE = A[0]#,
    STRNO     = A[1]#,
    TYPESTATS = A[2]#,
    RESULT    = BOOLEAN(A[3])#,
    TOTWORDS  = A[4]#,
    GROUP1INX = A[5]#;

  DEFINE
    GROUPTYPE = [23:8]#,
    GROUPSIZE = [15:16]#;

  DEFINE ERR(MSG) =
  BEGIN
    DISPLAY(MSG);
    MYSELF.STATUS := VALUE(TERMINATED);
  END;

  OPEN INQUIRY DB;

  % GET AND WRITE OUT STATIC STATISTICS FIRST

  DMINQCASE := 25;
  STRNO     := 0; % TO GET GLOBAL STATISTICS
  TYPESTATS := 0; % TO GET STATIC STATISTICS
  DMINQ[0](A);
  IF RESULT THEN
    ERR("FAILED TO RETRIEVE GLOBAL STATIC STATISTICS");

  % FIND GROUP TYPE 6

  I := GROUP1INX; % INDEX OF FIRST GROUP HEADER WORD
  WHILE A[I].GROUPTYPE NEQ 0 AND A[I].GROUPTYPE NEQ 6 DO
    I := * + A[I].GROUPSIZE;
  IF A[I].GROUPTYPE = 0 THEN
    ERR("INVALID RESULTS FOR GLOBAL STATIC STATISTICS");
  MAXSTRNO := A[I+2];

  % WRITE OUT GLOBAL STATIC STATISTICS

  A[3] := TIME(7); % TIMESTAMP IT
  WRITE(DBSTATISTICS, TOTWORDS, A);

  % GET AND WRITE OUT ALL STRUCTURE STATIC STATISTICS

  FOR STRNO := 1 STEP 1 UNTIL MAXSTRNO DO
  BEGIN
    DMINQ[0](A);
    IF NOT RESULT THEN
      BEGIN
        A[3] := TIME(7);
        WRITE(DBSTATISTICS, TOTWORDS, A);
      END;
  END;

  % NOW, GET AND WRITE DYNAMIC STATISTICS ONCE A MINUTE UNTIL
  % OPERATOR CHANGES TASKVALUE TO A NON-ZERO VALUE

  TYPESTATS := 1; % DYNAMIC STATISTICS
  WHILE MYSELF.TASKVALUE = 0 DO
  BEGIN
    FOR STRNO := 0 STEP 1 UNTIL MAXSTRNO DO
    BEGIN
      DMINQ[0](A);
      IF NOT RESULT THEN
        BEGIN
          A[3] := TIME(7);
          WRITE(DBSTATISTICS, TOTWORDS, A);

```



```

        END;
    END;
    WAITANDRESET((60),MYSELF.EXCEPTIONEVENT);

    END;

    LOCK(DBSTATISTICS);
END.

```

D3046 ACR - BUFFERS MOVED TO DATA BASE ENVIRONMENT

The data base buffers have been moved from the structure environment to the data base environment. This permits abort recovery to manipulate the information in the buffer.

D3047 ACR - ALLOW "AUDIT CLOSE" MESSAGE

When all updaters of a data base close the data base and inquiry users remain, the audit files are left in use. If it is necessary to release the audits, the SM message "AUDIT CLOSE" will cause the audit files to be released.

D3306 ACR - "B7700CODE" OPTION

Because of some problems in the use of B7700 as a user dollar option, the B7700 option has been renamed the B7700CODE option in the ACCESSROUTINES. Setting B7700CODE in the ACCESSROUTINES causes code optimized for the B7000's to be generated if the DMALGOL compiler has been compiled with the B7700CODE option. Note that this change does not affect the DASDL compiler option B7700. Setting B7700 while compiling DASDL will cause the B7700CODE option to be set when compiling the ACCESSROUTINES.

D3315 ACR - FORCED, NORMAL OVERLAYS

The data base statistics now include the number of forced overlays and the number of normal overlays in the buffer statistics rather than just the total of the two. Forced overlays are caused by the total buffers space exceeding the allowed core value. Normal overlays occur when structures are closed or users change from random to serial or serial to random state.

D3331 ACR - "ACCESSROUTINES" ERROR MESSAGES

The ACCESSROUTINES recognize two general classes of exceptional situations: exceptions and errors. Exceptions are returned to user programs any time the ACCESSROUTINES cannot successfully or correctly complete a requested data management operation. Exceptions affect only the user program detecting the exceptional situation. Other user programs are unaffected unless the program receiving the exception causes an abort. Exception categories and subcategories that may be returned to user programs by DMSII software are documented in Appendix B of the DMSII Host Reference Manual (Form No. 5001498).

Errors detected by the ACCESSROUTINES are usually fatal, causing premature termination of the database stack. In general, the ACCESSROUTINES terminate with a fatal error for one of five reasons: (1) irrecoverable I/O errors on control information used to describe or access user data, (2) detection of inconsistent control information, (3) faults in ACCESSROUTINES code, (4) a system resource limitation has been exceeded, or (5) the operator has DSeD the database. When a fatal error is detected, the ACCESSROUTINES return a SYSTEMERROR exception to all user programs that currently have the database open. After all programs have exited the database, the database stack is terminated.

In the process of terminating the database stack after a fatal error has been detected, the ACCESSROUTINES display a message describing the type of error. This message also contains the sequence number of the line in DATABASE/SYMBOLIC where the error was detected and may contain a structure number if the error was related to a particular structure. The message is displayed under the mix number of the user program in whose stack the ACCESSROUTINES were running when the error was detected. If appropriate, a programdump is also taken to the TASKFILE of this user program.

This note lists the texts of error messages that the ACCESSROUTINES may display and, for each error, describes probable cause(s) and possible action(s) that users may take to remedy the error situation. In addition, certain integrity errors that were fatal on Mark 31 DMSII software but are not fatal on the Mark 32 release are discussed.

Error Messages

The text of each error message is listed below, followed by a probable cause and action (if any) that an installation may take to correct the problem. All these errors are fatal unless otherwise noted.

ABORT OR H/L RECOVERY FAILURE OR DSED

Causes: (1) Operator DSeD an abort or haltload recovery stack.
(2) Fault occurred during abort or haltload recovery.

Action: For Abort, running Recovery may fix the problem.

AUTO SET ENTRY MISSING DURING DELETE

Cause: While deleting a record from a dataset, the corresponding

entry in an automatic index set was not found.

Note: This error is fatal only under a limited set of conditions. Refer to the discussion of integrity errors below.

AUTO SET ENTRY MISSING DURING STORE

Cause: While storing a modified record into a dataset, the corresponding entry in an automatic index set was not found.

Note: This error is fatal only under a limited set of conditions. Refer to discussion of integrity errors below.

AUTO SET POINTS TO DELETED RECORD

Cause: An entry in an automatic index set does not have a corresponding record in the spanned dataset.

Note: This error is fatal only under a limited set of conditions. Refer to discussion of integrity errors below.

BAD DISK ADDRESS

Cause: While attempting to read from a database file, a block address that did not correspond to a block boundary in the file was detected.

Note: This error is only detected if the DASDL option DATACHECK1 is set.

BAD GO FORWARD

Cause: Read error or corrupted table structure detected while deleting an entry from an index sequential set.

BAD RECORD ADDRESS

Cause: An address of a dataset record did not point to a physical record boundary.

Note: This error is only detected if the DASDL option DATACHECK1 is set.

BAD SET TO OPERATION

Cause: An attempt was made to read data from a dataset during a SET <set name> TO BEGINNING on one of its spanning index sets.

CORRUPT DATA IN ORDERED DATA SET

Cause: Corrupted control information detected in an ordered dataset.

DATABASE HAS BEEN DSED

Cause: Another user program has detected a fatal error, or the operator has DSeD the database stack. The database is being terminated.

Note: This message does not result in a program dump.

DB FILE NOT OPEN FOR ABORT DIVEST FOLLOWING DMSREAD

Cause: An attempt was made to release a data buffer assigned to a structure whose file was not currently open.

DBRECONSTRUCTION INTERLOCK FAILURE

Cause: A user was in transaction state or an abort was pending after the database had been stopped for reconstruction.

DIRECT KEY CORRUPTED

Cause: An invalid key field has been detected while accessing a direct dataset.

DIVEST ERROR

Cause: An attempt was made to release a data buffer for reuse that was not in use.

DMSREAD ERROR DURING ABORT

Cause: An I/O error occurred while trying to read from a structure file.

DMSREAD ERROR ON RESTART DATA SET DURING ABORT

Cause: An I/O error occurred while trying to read the restart dataset.

ERROR DURING AUDIT FILE SWITCH

Cause: Failure to open a new audit file following an I/O error on the previous audit file.

ERROR IN BUFFER MANAGEMENT FOR FINAL CLOSE

Cause: Data buffers were not properly written back to disk during the final close of a structure.

ERROR IN CONTROL FILE HANDLING

Cause: An error occurred while manipulating the database control file. A previously displayed message will indicate the actual error.

ERROR IN DCB HANDLING

Causes: Each data base buffer is controlled by its corresponding data control block (DCB).
(1) A data control block has an invalid structure number.
(2) Failure to allocate a new data control block or data buffer.
(3) Attempted to discard a buffer which is still in use.

FAILED TO OPEN DB FILE FROM STRUCTUREFNS FOR ABORT

Cause: The ACCESSROUTINES performed incorrect file handling during an Abort.

FAILED TO SET RLAFILE.LASTRECORD

Cause: The LASTRECORD attribute of the row lockout audit file was not set properly before Abort attempted to close the file. The operator issued more than one DS to the Abort stack.

Action: Do not issue multiple DSs to the Abort stack.

FAILURE TO SET LASTRECORD ON DB FILE

Cause: Following a write error, the LASTRECORD attribute of a data file was incorrectly set. The operator issued more than one DS to the user program on whose stack the ACCESSROUTINES were executing when the write error was detected.

Action: Do not DS a user program more than once.

FAULT IN ACR CODE

Cause: A fault occurred while executing ACCESSROUTINES code.

FAULT IN USER SELECTION EXPRESSION

Cause: A fault occurred while evaluating the <key condition> part of a <selection expression> in a host language DELETE, FIND, LOCK, or MODIFY statement. Probable error in the user program (for example, an INVALID INDEX was detected while accessing a host language data item).

Action: Correct the user program's <selection expression>.

Note: This error is not fatal to the database. The user program will be DSed with an exponent overflow fault unless it directly handles the fault.

I/O ERROR IN STORAGE CONTROL

Cause: An I/O error was encountered while reading control information (such as the DKTABLE of a standard dataset) from a database file.

Action: Reconstruct the affected row of the disk file. If the failure involves the restart data set, data base rebuild may be necessary.

Note: Prior to Mark 32, this error could also occur on writes. Occurrences of this error following write operations have been eliminated on Mark 32 level software. Because control information is maintained in memory following writes, write errors can be ignored until a reread from the file is attempted.

IMPROPER MAXRECSIZE OR BLOCKSIZE IN RLAFILE

Cause: A file with the title <database name>/ROWLOCKOUTAUDIT was opened but is probably not a true row lock out audit file. Probable operational error.

KEY AND DATA MISMATCH

Cause: The key value in an index set does not match the key field of the data set record to which the index set points.

Note: This error is nonfatal and is only detected when the DASDL option DATACHECK2 is set. Refer to the discussion of integrity errors for more information.

LOGIC ERROR IN ABORT DKTABLE HANDLING

Cause: The DKTABLE of a standard dataset was found to be inconsistent with the audit.

LOGIC ERROR IN ROWLOCKOUTAUDIT HANDLING

Cause: Abort tried to reinitialize the row lockout audit.

OPERATOR DS OF DBS

Cause: The operator has DSed the database stack.

Note: This message does not produce a program dump.

OUT OF CORE

Cause: Insufficient memory was available to allocate a new restart area for the restart data set.

PARTITION USER COUNT = 0

Cause: A dataset partition is currently active but has no users.

PARTITIONED STRUCTURE NOT IN PARTITIONDIRECTORY (ABORT)

Cause: Following an Abort, a partition file that should have been in the partitiondirectory dataset was not.

READ ATTEMPT ON UNOPENED PARTITION (ABORT)

Cause: Abort tried to read from an unopened partition file.

READ ERROR ON ROW LOCKOUT AUDIT FILE

Cause: A read of the row lockout audit file failed.

READ PAST END OF FILE

Cause: The ACCESSROUTINES attempted to read beyond the current end of file pointer of a data file.

RLAFILE NOT INITIALIZED IN RECOVERY

Cause: Abort tried to write to the row lockout audit file before the file was initialized.

ROW LOCKOUT AUDIT CHECKSUM ERROR

Cause: A checksum failure was detected on the row lockout audit file.

ROW LOCKOUT AUDIT DATA ERROR

Cause: The row lockout audit contains inconsistent information.

ROW LOCKOUT AUDIT DB DTS/ACR DB DTS MISMATCH

Cause: The database timestamps of the ACCESSROUTINES and the row lockout audit do not match. Probable operational error; either the wrong row lockout audit or the wrong ACCESSROUTINES were used.

B6000 SERIES MARK 32

ROW LOCKOUT AUDIT LEVEL/ACR MISMATCH

Cause: A level mismatch was detected between the row lockout audit and the ACCESSROUTINES. Probable operational problem. The ACCESSROUTINES were either too old or too new for the row lockout audit.

ROW LOCKOUT AUDIT TIME STAMP MISMATCH

Cause: The timestamps of consecutive blocks in the row lockout audit file do not match.

TOO MANY BUFFERS

Cause: The ACCESSROUTINES attempted to allocate more than 4096 data buffers. This can occur in databases with a very large number of structures.

Action: Either of the following may remedy the problem: reduce ALLOWEDCORE or reduce the maximum number of buffers that can be allocated to each structure.

TOO MANY LEVELS OF TABLES FOR I-S

Cause: Too many levels of tables have been allocated for an index sequential set. Either Reorganization or modifying index set tables sizes may help.

UNKNOWN ERROR IN ACR

Cause: The error handling procedure of the ACCESSROUTINES was called to handle an error that it didn't recognize.

UNKNOWN ROW LOCKOUT AUDIT RECORD TYPE

Cause: The row lockout audit file contained a record with an unrecognized type field.

UNRECOGNIZED AUDREC TYPE IN ABORT STRUCTUREFNS

Cause: Abort returned an unknown audit record type to the ACCESSROUTINES.

WAITERS ON UNLOCKED RECORD

Cause: A user program was waiting to lock a record that was not locked.

WRITE ERROR ON RLAFILE

Cause: Abort detected an I/O error while writing the row lockout audit file.

ZERO DISK ADDRESS

Cause: A read was attempted on an invalid disk address.

ZERO FIND ADDRESS

Cause: An index set contained a record address of zero.

Integrity Errors

Mark 32 software handles four of the above errors differently than previous releases. These errors are related to certain types of corruption in control information that is stored with the data for each structure. The error KEY AND DATA MISMATCH is no longer fatal. AUTO SET MISSING DURING DELETE, AUTO SET MISSING DURING STORE, and AUTO SET POINTS TO DELETED RECORD may be fatal only for datasets that are spanned by multiple index sets.

When a non-fatal integrity error is detected, a program dump is taken and a message is displayed indicating the job and task mix numbers of the user program that detected the error. A new category of exceptions is returned to user programs that detect non-fatal integrity errors. The category is called INTEGRITYERROR (exception category 20) and has the following subcategories:

- 1 Key and data mismatch
- 2 Automatic set or subset entry missing during STORE
- 3 Automatic set or subset entry missing during DELETE
- 4 Automatic set or subset points to deleted record

User programs may continue to access database structures in which integrity errors have been found so long as they do not access the corrupted control information. Every program that tries to use corrupted information will cause a program dump and receive an INTEGRITYERROR exception.

D3338 ACR - PRINT STATISTICS OPTION

An option has been added that controls the printing of statistics at the final close of the data base. The option is controlled through the Visible DBS messages "STATISTICS ON" and "STATISTICS OFF". "STATISTICS ON" enables printing; "STATISTICS OFF" disables printing. The state of the option is retained in the control file. Turning the option off does not stop collection of the statistics; they may still be printed via the Visible DBS message "STATISTICS" or "STATISTICS RESTART". The state of the option is not affected by printing them in this manner.

D3366 ACR - SAVE AND RETRIEVE MESSAGES

The most recent 23 displayable messages (except those generated by the Visible DBS) are now saved in a "Message Table" array, for retrieval via the DMINQ function.

D3452 ACR - ABORT ACCELERATION

Previously, Abort would close all the data base files, and, then, reopen them before beginning its image processing phase. After its image processing phase, it would again close down and reopen the files. This opening and closing of the data base files caused Abort to be unnecessarily slow.

This initial and final opening and closing of the files is no longer done. Additionally, Abort now uses the generally more efficient I/O routines of the ACCESSROUTINES (rather than its own I/O routines) to access the data base files. This, coupled with the elimination of the unnecessary file opening and closing, should result in a significant improvement in the speed of Abort, especially for data base with many structures.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - ACCESSROUTINES

P2499 ACR - EFFICIENT PACK SPACE UTILIZATION

When duplicated audit is used and both audits are on disk or pack, the ACCESSROUTINES will now use the disk space as efficiently as it does for single audit to pack. When a partially filled audit block is flushed to the audit files, as, for example, at syncpoint time, the block will continue to be filled and will be reflushed later.

P2632 ACR - AUDIT RESTART INFORMATION CORRECTLY

ACCESSROUTINES no longer audit restart information incorrectly. Formerly, ABORT or Halt/Load recovery would fail to capture restart areas for a program. This was especially prevalent when programs closed the data base when an ABORT was pending.

P2633 ACR - DS OF "SECTORS REQUIRED" VS. RECONSTRUCTION

If a program were DSed waiting for sectors on a data base file:

1. The data base may have been left corrupted.
2. Any reconstruct of the unallocated row would either fail or not correct the corruption.

This problem has been corrected.

Note: It is never a good idea to DS a DB program waiting for sectors, since it may be performing a critical I/O (a DS may cause the data base to go down); however, this change guarantees that, if a DS occurs, reconstruction will work properly.

P2752 ACR - SYSTEM SERIAL NUMBER ADDED

The system serial number has been added to the statistics heading.

P2753 ACR - REMOVE "RSFILE" DECLARATION

The D1 declaration for the restart data set has been removed.

P2754 ACR - AUDIT FILE ERROR HANDLING

A debugging facility has been added which allows the simulation of audit errors. In addition, the error messages for audit errors have been improved. The error messages now include the data base name, the audit name and the pack family name. When an audit is closed as a result of an error, the last good audit block serial number (ABSN) on that audit is displayed. This should prove useful if it is necessary to run COPYAUDIT on the audit.

P2755 ACR - FLUSH BUFFERS FOR STRUCTURE

Previously, for unaudited data bases, modified buffers were not flushed until all users of the structure were gone; thus, a structure could be corrupted by a system failure even though only INQUIRY programs remained active. Now, all modified buffers are flushed when the last updater of the structure closes the data base.

In addition, the number of users that have a structure open update and open inquiry will be displayed in the structure status display.

P2756 ACR - DO NOT SET INUSE FLAG

The ACCESSROUTINES were setting the inuse flag in the control file before opening the audits. If a failure occurred before the audits were successfully opened, RECOVERY would require a non-existent audit. This problem has been corrected.

P2757 ACR - IMMEDIATE OVERLAY OF BUFFERS

When "buffers" are changed through the visible DBS, the ACCESSROUTINES will adjust the buffer pool to conform to the specifications. Formerly, under certain circumstances, this action was delayed and/or done improperly.

P2788 ACR - DIVEST COMPACT TABLE BLOCK IF READ ERROR

If an error occurred while reading a compact table block, the table would remain marked in use. The problem has been corrected.

P2882 ACR - "ZERO DISK ADDRESS" FOR DIRECT DATA SET

For direct data sets with BLOCKSIZE=1, FIND PRIOR failed with a ZERO DISK ADDRESS error. The ACCESSROUTINES were not stopping correctly when all records in the prior direction were exhausted. In addition, Notfound/No Current Record and Notfound/Select Text Error exceptions were not returned correctly for direct data sets. These problems have been corrected.

P2883 ACR - QUICK FIX CREATES EMPTY AUDIT FILES

Under certain conditions, if a quick fix is done while the data base is open inquiry, an empty audit file would be created. This problem has been corrected.

P2885 ACR - REUSE EMPTY AUDIT CORRECTLY

If all audits were on pack or disk and the audits were empty (contained only block 0) at the time the ACCESSROUTINES picked them up for reuse, the first block of the audit would be discarded. This problem has been corrected.

P2886 ACR - "RECONSTRUCT" MAKES EMPTY AUDIT FILES

RECONSTRUCT can produce empty audit files if run while the data base is open inquiry.

P2948 ACR - DISPLAY REASON FOR NOT REUSING AUDIT

If the ACCESSROUTINES do not reuse the current disk or pack audits, a message will now be displayed explaining the reason for not reusing the audit file(s).

P2949 ACR - UNLOCK PARTITION IF "DS" IN OPEN

If a program was DSed while waiting on a NO FILE for a partition, it would be impossible to open the structure again. This problem has been corrected.

P2950 ACR - "INVALID OP" WITH READAHEAD

The ACCESSROUTINES no longer faults with an INVALID OP when READAHEAD is set on an index sequential set or subset. Formerly, this could only happen on a FIND PRIOR (or DELETE PRIOR, LOCK PRIOR).

P2964 ACR - DATA BASE MESSAGES

The following changes have been made to DISPLAY and ACCEPT messages originating from the ACCESSROUTINES:

1. Data base name prefixes all messages except those from the visible DBS.
2. "WAITING FOR ABORT" messages contain the mix numbers of the job/session and task causing the abort.
3. Data base fatal errors display the ACCESSROUTINES sequence number where the error was detected.
4. Some messages have been shortened.

P3003 ACR - BAD AVAILABLE TABLES FOR COMPACT DATA SET

If the ACCESSROUTINES find an available table entry that does not correspond to the actual data block, the entry will now be discarded and ignored. Previously, a fault could occur in the ACCESSROUTINES.

P3030 ACR - NESTED "STARTDB" ERRORS

When running INQUIRY against an ACCESSROUTINES compiled with DEBUG set (an internal debugging option), the data base would die with a BAD LOCK error, due to nested STARTDB calls. This problem has been corrected.

P3031 ACR - "I/O" TIMEOUT

When an I/O operation times out, the ACCESSROUTINES could cause certain user programs to be DSed because the block in which the I/O event was declared had been exited just before the I/O finally completed. This problem has been corrected.

P3033 ACR - DATA BASE SUBSYSTEM VISIBLE

During data base initialization, the ACCESSROUTINES now store the subsystem task attribute of the data base stack; consequently, ACCESSROUTINES code segments running on user stacks can ensure that ZIPed jobs run in the same subsystem as the data base stack.

P3101 ACR - NESTED "STARTDB"

Pathfinder or Set to Beginning calls on an access via the INQ interface would execute nested STARTDBs, thus causing a potential problem when attempting to stop the data base for Abort and Reconstruct. This problem has been corrected.

P3106 ACR - INTERFACE TO FREE STACK RECORDS

As part of the support for swappable data bases, an interface has been implemented in the ACCESSROUTINES to allow the MCP to free a specific stack's locked records.

P3175 ACR - CONTROL FILE "I/O" LOCK

A lock has been added to the control file module to enforce single-threaded I/O operation against the data base control file.

P3176 ACR - FAULT ON REBLOCKED STANDARD DATA SET

In data bases with multiple standard data sets, after image auditing for a reblocked standard data set could cause a fault in the ACCESSROUTINES code under some circumstances. This problem has been corrected.

P3177 ACR - TOTALCORE PROTECTED BY MEMLOCK

Memlock is now acquired before the variable Totalcore is updated. Previously, the ACCESSROUTINES incorrectly acquired the Olaylock.

P3178 ACR - PREVENT "COPYAUDIT" ZIP DELAY

If an audit file switch occurred due to an audit error, the zipping of COPYAUDIT would be delayed until the next audit file switch or the close of the data base. This problem has been corrected.

P3230 ACR - DELETING VARIABLE FORMAT RECORDS

A fault could be encountered when deleting variable format records if the fixed part contained no control information but some variable part did. This problem has been corrected.

P3246 ACR - "ERROR IN DCB HANDLING"

If a LIMITERROR were encountered on a random data set, an ERROR IN DCB HANDLING would result either after ABORT ran or at the final close of the structure. This problem has been corrected.

P3247 ACR - "DBSINFO" REPLACES "MYNUMBER"

Prior to the Mark 32 release, stack cell (2,1) of each structure environment in the data base stack (DBS) contained the structure number (called MYNUMBER in the ACCESSROUTINES). Because of the reorganization of the DBS (DMS II note D3108) to support shared ACCESSROUTINES (DMS II note D3170), structure environments have moved to D[3] in Mark 32 DBSs. The MYNUMBER stack cell located at (3,2) in each environment, has been renamed DBSINFO, and contains additional information on Mark 32 level software. This additional information can be used to facilitate the interpretation of DBS program dumps. The format of DBSINFO is

47:01	DBSINFOLTMDF	Structure has LOCK TO MODIFY DETAILS code compiled into it (Datasets only).
46:01	DBSINFOREMAPF	Structure is remapped (Datasets only).
45:01	DBSINFOSPANNEDF	Structure is spanned by an index set (Datasets only).
44:01	DBSINFOMANSPANNEDF	Structure is spanned by a manual subset (Datasets only).
43:01	DBSINFOSUBSETF	Structure is a subset (Index sets only).
42:11		Not used.
31:08	DBSINFOLEVELF	Embedding level of structure; Disjoint structures are at level 1.
23:04	DBSINFOSUBTYPEF	SUBTYPEF property of the structure.
19:08	DBSINFOTYPEF	TYPEF property of the structure.
11:12	DBSINFOSTRNUMF	Structure number.

P3259 ACR - MISSING DIVEST ON DEADLOCK

If a DEADLOCK condition were detected by the ACCESSROUTINES, a block would remain marked as in-use, eventually leading to an error in DCB HANDLING when the data base was closed. This has been corrected.

P3284 ACR - "DCB" HANDLING ERROR

If duplicates were detected on a create-store for a direct data set, any spanning set tables would be left locked, leading to an "ERROR IN DCB HANDLING" on the final close of the structure. The problem has been corrected.

P3312 ACR - BAD LINKS WHEN OPEN INQUIRY

Previously, the ACCESSROUTINES tried to fix-up bad links when the user had only opened the data base inquiry. This caused an error in DCB handling at close time since there should be no modifications to the data base when it was open inquiry.

P3313 ACR - LINKS ON SELECT TEXT ERROR

When performing a find via a link, if the select text expression failed for the remap referenced by the link, the ACCESSROUTINES would erroneously treat the link as a bad link. Depending on the link type, the ACCESSROUTINES would either try to fix-up the link or make it null. This problem has been corrected. The ACCESSROUTINES will now return a NOTFOUND exception.

P3314 ACR - INVALID "INQUIRY" FUNCTION NUMBERS IGNORED

Invalid DMINQ function numbers will now be ignored by the ACCESSROUTINES.

P3333 ACR - STORE RESTART AREA

In the event that users of a data base are DSed, the DSed programs could receive a deadlock exception when the ACCESSROUTINES stored their Restart Areas. Those user programs receiving the exception would not have their Restart Areas stored. This problem has been corrected.

P3340 ACR - STANDARD "VF" CONTROL WORD CORRUPTED

When a Standard Variable format data set which utilized reblocking was extended, the control words in newly allocated blocks were not initialized correctly. In new blocks, the Datablockflag was not initialized to zero. As a result, FIND NEXT through the data set handled available blocks as if they were data blocks containing valid records. This problem has been corrected.

P3368 ACR - DO NOT POINT LINKS AT OVERFLOW BLOCK

Previously, links to compact data set records which had overflowed their original data block pointed at the overflow block. If the record was modified and relocated to another block, the link no longer pointed to the record. This problem has been corrected. Links to compact data set records now always point to the original data block entry.

P3369 ACR - MISSING DIVEST FOLLOWING VERSION ERROR

Under some circumstances, the ACCESSROUTINES were not releasing data buffers following detection of version error exceptions. Consequently, the ACCESSROUTINES would detect an "in-use" data buffer before all user programs had closed a structure and terminate the data base stack for an "ERROR IN DCB HANDLING". The problem occurred when a user program that had been invalidated by a data base reorganization that altered data set record formats attempted to access a reorganized data set via one of its spanning sets. The ACCESSROUTINES now correctly release data buffers following versionerror exceptions.

P3370 ACR - FORGET SUBBLOCK FOR ORDERED DATA SET

When ordered data sets with subblocks are used, the ACCESSROUTINES keep track of empty subblocks in available tables linked from block zero. Empty subblocks may result from record deletions or creations. Previously, new available tables were added to the end of the available table chain. When long chains were present, many I/Os were required to maintain the chain. Now, new tables are always added to the beginning of the chain.

P3371 ACR - ZEROED OUT BLOCKS IN DATA BASE

Occasionally, zeroed out blocks were written back to the data base. This happened when the ACR procedure DUMPBUFFERS attempted to write back a deallocated buffer. This is now detected and prevented.

P3383 ACR - PARTITION OPEN ERROR

Under some circumstances, a user program that received a LIMITERROR for trying to open too many partitions could cause subsequent attempts to open other partitions to fail with an ERROR IN CONTROL FILE HANDLER. The problem has been corrected.

P3393 ACR - ERRONEOUS "BIO/AIO" AUDIT RECORDS

Previously, an incorrect address was placed in the address word of the BEFORE-IMAGE-ONLY/AFTER-IMAGE-ONLY audit records for bit vectors. This problem has been corrected.

P3553 ACR - INVALID STANDARD VARIABLE FORMAT RECORD TYPES

Previously, when a new standard variable format record was allocated from end of file, GETDATAADDRESS placed high-value in the record type field. Later, STORE placed the correct record type and other data values in the record. If STORE terminated with an exception, the record was added to the available chain but the correct record type was not placed in the record. Subsequent additions to the structure would eventually reuse the record and the correct record type would then be inserted. The ACCESSROUTINES were capable of bypassing records with high-value in the record type, but other software failed when encountering them.

Now, the ACCESSROUTINES always initialize newly-allocated records with the record type and invalid record text.

P3554 ACR - PARTITION AUDIT RECORDS OUT OF ORDER

Previously, a CPNT audit record reflecting a change in the PARTITIONDIRECTORY and PARTITIONNAMES table was written to the audit. Then, a PNT audit record reflecting the state of the PARTITIONDIRECTORY and PARTITIONNAMES table PRIOR to the change indicated by the CPNT was written.

This problem could be manifested in REBUILD, RECONSTRUCT or Halt/Load RECOVERY as a "PARTITION NAME TABLE INCONSISTENT WITH AUDIT" error. This problem has been corrected.

P3606 ACR - LINEAR SEARCH WITH SIGNED NUMERIC KEYS

A fault in the ACCESSROUTINES no longer occurs when a linear search is done on a signed numeric key.

P3618 ACR - CANCEL OR COMPLETE "I/O" FOLLOWING TIMEOUT

Previously, if the ACCESSROUTINES initiated a write to an audit file and the write timed out, it was possible for the user program to be DSed with an I/O ERROR 17 (previously-initiated I/O was neither completed nor cancelled). Now, if a timeout occurs when updating segment zero of a pack or disk type audit, a message will be displayed indicating which audit file received the timeout, along with a message indicating that the ACCESSROUTINES is waiting for this I/O to complete. If a timeout occurs when updating any audit file segment other than segment zero, the ACCESSROUTINES will display a message indicating which audit received the timeout, as well as the location (FAMILYNAME or magnetic TAPE unit number) of the file. Two messages will then be displayed indicating the last good audit block serialnumber and that the ACCESSROUTINES is waiting for the I/O complete on the primary or secondary audit (whichever is appropriate). The ACCESSROUTINES will then wait for the I/O to complete before continuing. This strategy is necessary since I/O operations to disk cannot be cancelled.

P3683 ACR - ORDERED DATA SET DIVEST ERROR

Following an error result returned by a call on GETDATA, the ordered data set DATAFINDER procedure was trying to recall GETDATA passing DCB number zero. Since DCB zero is never used, the data base stack was DSed for a divest error. The problem has been corrected.

P3696 ACR - "DMSECURITYERROR".

Previously, if two or more application programs opened a data base following a Halt/Load and the application program which initiated Halt/Load recovery was not the first program to continue processing after Recovery completed, that program would be terminated with a DMSECURITY ERROR exception. This problem has been corrected.

P3697 ACR - FAIL TO DIVEST

Previously, an attempt to lock global data which resulted in a Deadlock exception would fail to divest the buffer. At final close of the structure, this would result in data base failure with "ERROR IN BUFFER MANAGEMENT FOR FINAL CLOSE". This problem has been corrected.

P3708 ACR - INFINITE LOOP

The ACCESSROUTINES may get into an infinite loop before firing up ABORT in the following situation:

Program P1 which is running at a high priority has locked record R. Program P2 which is running at a very low priority wants to lock record R and so waits for the record to become available. Program P1 is DSed and is attempting to initiate ABORT.

This problem has been corrected.

P3755 ACR - "NOTLOCKED" EXCEPTION

Previously, it was possible for one user to get a NOTLOCKED exception when doing a CREATE-STORE on an ORDERED data set if his current path pointed to a record that was in the process of being deleted by another user. This problem has been corrected.

P3773 ACR - "FORGETSPACE" TIMING WINDOW

A timing window in the ACCESSROUTINES buffer overlaying procedure (FORGETSPACE) could result in the following problems:

1. Data base stack terminates abnormally for a "FAULT IN ACR CODE".
2. A buffer of zeroes (with valid checksum) is written to the data file.

These problems have been corrected.

P3789 ACR - CANNOT LOCATE COMPACT RECORD

A SEG ARRAY error may occur within the ACCESSROUTINES if one program is attempting to FIND a compact data set record, while another program is in the process of STOREing an update version of that same data set record. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - ARCHIVEUPDATER

D3614 ARCHUPDATE - ELIMINATE "OPEN INITIALIZE"

The OPEN INITIALIZE command has been de-implemented. ARCHIVEUPDATER can no longer OPEN INITIALIZE a data base. LOADDUMP now opens the data base UPDATE instead of INITIALIZE.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - ARCHIVEUPDATER

P3341 ARCHUPDATE - PREVENT SORT ERROR #4

ARCHUPDATE now computes the amount of sort disk based upon input volume. This prevents sort errors which result from insufficient work disk.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - BDMSALGOL

D3324 BDMSALGOL - PRINT DATA BASE TITLE

Data base equation information is now printed following the data base declaration when \$LISTDB is set.

D3440 BDMSALGOL - COMPILER IDENTIFICATION

The compiler identification line appearing at the beginning of printer listings now identifies the BDMSALGOL compiler as "BDMSALGOL" instead of "ALGOL".

D3467 BDMSALGOL - IMPLICIT FREE FOR "FIND, LOCK"

The DMSII HOST Manual incorrectly describes when implicit FREES are done for FIND and LOCK statements. The description of FIND and LOCK should be modified as follows:

"FIND and LOCK always FREE the previously-locked record when the operation uses a data set. When the operation uses a set, the previously-locked record is implicitly FREEd only when a new record is located in the set."

D3552 BDMSALGOL - DEIMPLEMENT "OPEN INITIALIZE"

The OPEN INITIALIZE construct has been deimplemented. The compiler no longer recognizes this construct.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - BDMSALGOL

P3296 BDMSALGOL - TRANSACTION RECORD PARAMETERS

The ALGOL compiler generated incorrect code for accessing NUMBER type items of TRANSACTION RECORDS which were formal procedure parameters. When a record was passed as a parameter which was shorter than expected, the procedure was able to read from and write into memory beyond the end of the short record area without getting an INVALID INDEX.

This problem has been corrected. Now, an INVALID INDEX occurs when an attempt is made to reference an item beyond the end of the actual transaction record.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - BDMSCOBOL

--- -- - -----

D3325 BDMSCOBOL - DATA BASE EQUATION INFORMATION

The data base equation information is now printed along with other data base information as a comment in the DATA-BASE section.

D3467 BDMSCOBOL - IMPLICIT FREE FOR "FIND, LOCK"

The DMSII HOST Manual incorrectly describes when implicit FREES are done for FIND and LOCK statements. The description of FIND and LOCK should be modified as follows:

"FIND and LOCK always FREE the previously-locked record when the operation uses a data set. When the operation uses a set, the previously-locked record is implicitly FREEd only when a new record is located in the set."

D3555 BDMSCOBOL - "OPEN INITIALIZE" DEIMPLEMENTED

The OPEN INITIALIZE construct has been deimplemented. The compiler no longer recognizes this construct.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - BDMSCOBOL

P2730 BDMSCOBOL - "BDMSCOBOL" GENERATES BAD PRINT LINE

BDMSCOBOL generates a bad print line for DATAITEM REAL(12,12), causing errors in AUDIT-REPORTER. This problem has been corrected.

P2731 BDMSCOBOL - "INVALID INDEX"

The compiler faulted on the following statement, where B is a Boolean in a data base:

```
COMPUTE I=B+1
```

This problem has been corrected.

P3085 BDMSCOBOL - TRANSACTION ITEM, "1" OR "2" CHARACTERS

The compiler gave a syntax error when referencing a transaction item with a one- or two-character name. This problem has been corrected.

P3588 BDMSCOBOL - INVALID HEADER

Incorrect information was stored in the invocation list in the code file for a transaction subbase invocation. This no longer occurs.

P3589 BDMSCOBOL - "INVALID INDEX"

The compiler aborted with an INVALID INDEX when compiling a DMSII program that had the DATADICTINFO option set. This problem, which occurred only when invoking a large data base, has been corrected.

P3590 BDMSCOBOL - "INVALID INDEX"

A syntax error in an OPEN statement no longer causes an INVALID INDEX.

P3591 BDMSCOBOL - "DB-INVOKE" HARDLY READABLE

Picture and Usage information in a COBOL listing with DB-INVOKE was printed in specific columns, making the data base information unreadable. This has been corrected.

P3635 BDMSCOBOL - "INVALID OP" ACCESSING GLOBAL DATA ITEMS

When a data base was declared global in a bound-in COBOL procedure, an attempt to access global DM-data-items resulted in an INVALID OP. This no longer occurs.

P3636 BDMSCOBOL - LINEAR SEARCH SELECTION EXPRESSION

The COBOL compiler will no longer generate incorrect code (which caused an INVALID OP) when the following conditions are met:

1. The data base is declared to be global.
2. A selection expression causes a linear search.
3. The key item referenced is accessed by other than its default character type.

P3803 BDMSCOBOL - "DUMP PRINTER (<DATASET-NAME>)" STATEMENT

The following BDMSCOBOL statement caused bad code to be generated:

```
DUMP PRINTER (<dataset-name>)
```

The compiler attempted to reference the spanning sets as if they were data-items. This no longer occurs.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - BDMS/PL/I

--- -- - -----

D3467 BDMSPLI - IMPLICIT FREE FOR "FIND, LOCK"

The DMSII HOST Manual incorrectly describes when implicit FREES are done for FIND and LOCK statements. The description of FIND and LOCK should be modified as follows:

"FIND and LOCK always FREE the previously-locked record when the operation uses a data set. When the operation uses a set, the previously-locked record is implicitly FREEd only when a new record is located in the set."

D3556 BDMSPLI - "OPEN INITIALIZE" DEIMPLEMENTED

The OPEN INITIALIZE construct has been deimplemented. The compiler no longer recognizes this construct.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - BDMS/PL/I

P2842 BDMSPLI - "PL/I" COMPILER LOOPING

When a nonexistent data set or set was listed in a data base declaration, the PL/I compiler sometimes looped. This problem has been corrected.

P2843 BDMSPLI - TRANSACTION ITEMS

The following problems have been corrected:

1. Multi-dimensional transaction items could not be referenced.
2. Transaction items could not be used as operands of expressions (other than simple assignments).
3. Numeric transaction items could not be used as operands of builtin functions.

P2844 BDMSPLI - MOVING DATA BASE ITEMS

The following assignment statements all generated incorrect code (where DB1 through DB5 are data base items):

```
DB1(1)=DB1(2);
DB2,DB3=DB4;
DB5,X=DB4;
P:PROC(Y); DCL Y PIC 'H';
Y=DB6;
END P;
```

These problems have been corrected.

P2845 BDMSPLI - "XREF" OPTION WITH "BDMS"

When the XREF option was set when compiling a PL/I program which invoked a data base, the compiler faulted with an INVALID INDEX. This problem has been corrected.

P2846 BDMSPLI - INCORRECT "BDMS OPEN" STATEMENT

When the keyword "OPTIONS" was omitted from a data base OPEN statement, the compiler failed to generate an error message; instead, it generated code to open UPDATE.

Example:

```
OPEN TESTDATABASE (INQUIRY);
```

This statement caused open update and no error message. This problem has been corrected; an appropriate error message is now given.

P2847 BDMSPLI - MULTIDIMENSIONAL "DMS" ARRAYS

The PL/I compiler gave syntax errors for correct references to data base items with more than one subscript. This problem has been corrected.

P2848 BDMSPLI - BUILTIN FUNCTIONS AND "BDMS"

When a DM Variable was the operand of a builtin function such as ABS (), MIN () or MAX (), the value of the variable was sometimes erroneously integerized. This problem has been corrected.

P3209 BDMSPLI - "CREATE" STATEMENT

On the initial Mark 31 release of PL/I, the CREATE statement did not work properly when nested two or more levels deep in procedures. This problem has been corrected.

P3210 BDMSPLI - "DATADICTINFO"

A data base with no global data caused the PL/I compiler to loop when the dollar option DATADICTINFO was set. Now, the compiler will no longer loop.

P3594 BDMSPLI - "PUT EDIT" OF DATA BASE ITEMS

Attempting to use a data base item in the I/O list of a PUT EDIT statement produced a syntax error, "ILLEGAL I/O LIST ITEM". Now, use of data base items in PUT EDIT statements works correctly.

P3595 BDMSPLI - DATA BASE "BINDINFO"

The PL/I compiler no longer generates incorrect BINDINFO for data base structures.

In addition, SYSTEM/PRINTBINDINFO now correctly handles BINDINFO for data base structures; it no longer loops endlessly.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - BDMSCOBOL74

D3545 BDMSCOBOL74 - "BDMS" OPTION IMPLEMENTED

The COBOL74 compiler now supports the BDMS option. The features provided under this option were previously described for COBOL(68).

D3555 BDMSCOBOL74 - "OPEN INITIALIZE" DEIMPLEMENTED

The OPEN INITIALIZE construct has been deimplemented. The compiler no longer recognizes this construct.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - BDMSCOBOL74

P3588 BDMSCOBOL74 - INVALID HEADER

Incorrect information was stored in the invocation list in the code file for a transaction subbase invocation. This no longer occurs.

P3589 BDMSCOBOL74 - "INVALID INDEX"

The compiler aborted with an INVALID INDEX when compiling a DMSII program that had the DATADICTINFO option set. This problem, which occurred only when invoking a large data base, has been corrected.

P3590 BDMSCOBOL74 - "INVALID INDEX"

A syntax error in an OPEN statement no longer causes an INVALID INDEX.

P3591 BDMSCOBOL74 - "DB-INVOKE" HARDLY READABLE

Picture and Usage information in a COBOL listing with DB-INVOKE was printed in specific columns, making the data base information unreadable. This has been corrected.

P3635 BDMSCOBOL74 - "INVALID OP" ACCESSING GLOBAL DATA ITEMS

When a data base was declared global in a bound-in COBOL procedure, an attempt to access global DM-data-items resulted in an INVALID OP. This no longer occurs.

P3636 BDMSCOBOL74 - LINEAR SEARCH SELECTION EXPRESSION

The COBOL compiler will no longer generate incorrect code (which caused an INVALID OP) when the following conditions are met:

1. The data base is declared to be global.
2. A selection expression causes a linear search.
3. The key item referenced is accessed by other than its default character type.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - BUILDINQUIRY

P3094 BUILDINQ - RENAMED "RESTART" DATA SET

A renamed RESTART data set in a logical data base could cause a syntax error in the compilation of INQUIRY. The error would occur when UPDATE was specified to BUILDINQ. This problem has been corrected.

P3698 BUILDINQ - MULTIPLE BLOCKS IN "DMINQDIRECTORY"

The manner in which BUILDINQ creates the DMINQDIRECTORY has been changed. As a result, the limit on the total number of combined data sets, sets and items that an INQUIRY program can access has been increased.

P3706 BUILDINQ - GLOBAL DATA IN LOGICAL DATA BASE

BUILDINQ was dying with a "STRUCTURE INVOKED TWICE" error when it was run to generate an INQUIRY program for a logical data base that invoked global data. Now, it successfully generates such an INQUIRY program.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - BUILDREORGANIZATION

D3082 BUILDREORG - IMPLICIT "GENERATE" STATEMENTS

BUILDREORG now provides additional implicit generates for certain structures. In fact, when UPDATE is specified to BUILDREORG, all generates are automatically provided to implement the changes specified in the previous DASDL UPDATE run. Thus no GENERATE statement is required.

Certain implicit generates may be modified by explicit specification. Modifications must adhere to current GENERATE statement restrictions.

1. If the UPDATE option is specified to BUILDREORG, the following implicit generates are provided for structures specified in DASDL as "REORGANIZE":

- A. The data set having no PRIME set specified in DASDL has the implicit generate:

GENERATE <data set>

- B. The data set having a PRIME set specified in DASDL has the implicit generate:

GENERATE <data set> ORDER BY <index str>

<Istr> references the PRIME set.

- C. All indexing structures specified in DASDL as "REORGANIZE (KEYS CHANGED)" have the implicit generate:

GENERATE <index str>

This implicit generate cannot be modified.

- D. All indexing structures specified in DASDL as "REORGANIZE (KEYS SAME)" have the implicit generate

GENERATE <index str>

provided:

- 1) The data set is generated.
- 2) The index structure is a disjoint automatic set or disjoint automatic subset.
- 3) The index structure is an unordered list or is an ordered list or index sequential structure with DUPLICATES allowed, but FIRST or LAST unspecified.

All other indexing structures specified in DASDL as "REORGANIZE (KEYS SAME)" have the implicit generate:

GENERATE <index str> USING <index str>

2. If the data set is generated, the following implicit generates are provided for spanning index structures which are not generated as described above:

- A. All Bit-vectors and manual subsets have the implicit generate:

GENERATE <index str> USING <index str>

- B. All automatic sets or subsets which are unordered lists or which are ordered lists or index sequential structures with DUPLICATES allowed, but FIRST or LAST unspecified, have the implicit generate

GENERATE <index str>

if disjoint, else

GENERATE <index str> USING <index str>

D3083 BUILDREORG - NEW DEFAULT FOR <SORT OPTIONS>

If <sort options> are not specified to BUILDREORG, "SORT USING 0 TAPES, 20000 SEGMENTS, FAMILYNAME = <id>" is assumed. <Id> is the INTERNAL FILES family name. The INTERNAL FILES family name is DISK by default.

D3084 BUILDREORG - SIMPLIFICATION OF "REORGANIZATON"

Changes have been made to the DMSII Reorganization facility in order to achieve the following:

1. Simplify the steps which must be performed in order to make changes to the data base description.

2. Allow the new ACCESSROUTINES and other data base system software to be compiled before or in conjunction with the reorganization of the data base files.
3. Support the reentrant data base capability.

The changes made to implement the above are briefly the following:

1. The second DASDL UPDATE run is no longer necessary following a DASDL UPDATE with REORGANIZE clauses.
2. An UPDATE option statement must be included in the specifications to BUILDREORG to effect format or record format conversion.
3. The ZIP option is now set by default for a DASDL UPDATE with REORGANIZE clauses.
4. A DASDL UPDATE no longer checks the control file for compatibility with the old description file. (See DASDL D-note "D3099 - INDEPENDENCE OF DASDL UPDATE COMPILATIONS" for a full explanation.)
5. The new description file, DESCRIPTION/<data base name>, is now input to the reorganization program to update the control file rather than the reorganization description file.

The following paragraphs document in more detail the steps which are now required to perform data base reorganization and the associated changes to the reorganization facility.

PERFORMING REORGANIZATION

The data base administrator must decide which data base structures require reorganization. The data base administrator then performs as many "reorganization runs" as necessary to achieve the desired resultant data base. A "reorganization run" is the performance of the following steps for the reorganization of at most one data set and its spanning sets.

1. If file or record format conversion is required, a backup copy of the DASDL source and data base description file should be made for recovery purposes. In addition, all data base system software should be saved.
2. If file or record format conversion is required, a DASDL compilation is performed specifying the UPDATE option, REORGANIZE clauses, and DASDL source changes. This step must be performed when physical attributes such as AREASIZE, BLOCKSIZE, and TABLESIZE are changed or record fields are added, deleted, or changed. A new description file is created which is marked as requiring reorganization. This description file contains both old and new formats and an incremented update level.

Both user programs and data base system software may be compiled against the new description file; however, these programs cannot run until the reorganization of the data base is complete. Old user programs can run with the old ACCESSROUTINES until execution of the reorganization program.

By default, DASDL ZIPs compilation of the ACCESSROUTINES and other data base system software when DASDL compilation is to LIBRARY or is initiated via CANDE. If the data base is to be used until the reorganization program is run, the ZIP option should be RESET.

3. SYSTEM/BUILDREORG is run. This program accepts the current description file and user specifications as input. If file or record format conversion is required, the UPDATE option must be included in the specifications to BUILDREORG. (See description of UPDATE option below.) If format changes are not required, no UPDATE option is given. The specifications to BUILDREORG identify structures requiring reorganization, describe how certain reorganizations are to be performed, and specify available system resources.

If no errors are encountered in the specifications, SYSTEM/BUILDREORG:

- a. Creates the reorganization description file titled "DESCRIPTION/REORGANIZATION/<data base name>" which contains the data base description and the specifications for reorganization. The reorganization description file is created only for the purpose of compiling the reorganization program.
- b. Produces a report which shows both the user and default reorganization specifications. All structures which will be modified by the reorganization program are listed on the report.
- c. ZIPs the compilation of the reorganization program unless explicitly overridden.

The reorganization program, "REORGANIZATION/<data base name>", is compiled via DMAGLOL utilizing "DATABASE/REORGSYMBOLIC" and the reorganization description file created by SYSTEM/BUILDREORG. Since the reorganization program invokes the data base, the description file must be present for the compilation.

4. A backup copy of the control file and all data base files should be made. If dumping the entire data base is not convenient, the control file and all data base structures which will be modified should be dumped.

B6000 SERIES MARK 32

5. The reorganization program is run to perform the actual reorganization of the data base files. The data base must not be in use at this time and all structures requiring reorganization or fixup must be available on the proper media.

The reorganization program is run with one of three parameters: GENERATE, COPY, or REMOVE. The program must first be run with the GENERATE option. In GENERATE mode the reorganization program immediately locks the control file and marks it "in exclusive use by REORGANIZATION." Until this time, user programs could access the data base via the old ACCESSROUTINES. Now they will get OPENERROrs when they attempt to open the data base. If UPDATE was specified to BUILDREORG, then SYSTEM/DMCONTROL is run by the reorganization program to update the control file. (The new description file must be present since it is needed by the control file program.) The DASDL UPDATE level in the control file is incremented and the structure directory is updated. The reorganization program then reorganizes the data base structures.

If a COPY TO intermediate medium with no COPY BACK was specified for one or more structures in the specifications to BUILDREORG, the reorganization program must next be run with the COPY option. In this mode, the program copies all such structures to their final medium.

To complete the data base reorganization, the reorganization program is run with the REMOVE option. The program unlocks the control file. The new data base system software and any recompiled user programs can now run. Old user programs which access structures whose records have been reformatted and old data base system software can no longer run.

6. A backup copy of the new control file and all data base files which were reorganized or fixed up should be made because of the discontinuity which was created in these files by their reorganization.

In addition, if the UPDATE option was specified to BUILDREORG, the reorganization program should be removed since it can no longer be used. If the UPDATE option was not specified to BUILDREORG, the reorganization program may be saved. It can be rerun to perform the same reorganization until the next DASDL UPDATE occurs, at which time it must be respecified and recompiled.

OPTIMIZING REORGANIZATION OF MULTIPLE DATA SETS

Where structure or record format changes are desired on multiple data sets and/or their spanning sets, repetitive performance of the above steps can be optimized. The procedure is outlined as follows:

- A. Step 1 as described above.
- B. Repetitive performance of steps 2 and 3 above with appropriate library maintenance to save all generated description files and reorganization programs. Care must be exercised to ensure that each description file is properly matched with its corresponding reorganization program and that the sequence of generation is remembered. The ZIP option should be reset for all DASDL UPDATES.
- C. Step 4 as described above.
- D. Repetitive performance of step 5 above using the description files and reorganization programs in the sequence of their generation. The data base must remain offline throughout this process. User programs and data base software may be compiled in parallel with the reorganization runs using the last generated description file.
- E. Essentially step 6 above. All reorganization programs and all intermediate description files can be discarded. All data base files which were reorganized or fixed up should be saved.

UPDATE OPTION

The BUILDREORG UPDATE option specifies that data base structures are to be reorganized to affect the file or record format changes which were specified in the previous DASDL UPDATE. The data base structures requiring such reorganization are provided with implicit default GENERATES. The UPDATE option must precede all other reorganization specifications. The BUILDREORG program ensures that the description file is marked as requiring reorganization and contains information on both old and new formats. The reorganization program checks that the update level in the reorganization description file is one greater than the update level in the control file. If this is not the case, the reorganization program aborts with the error message "CONTROL FILE UPDATE LEVEL MUST BE 1 LESS THAN CODE FILE" if the reorganization program was compiled against an incompatible description file, or "CONTROL FILE UPDATE LEVEL SHOULD BE 1 LESS THAN DESCRIPTION" if the reorganization program is run using an incompatible description file.

If the UPDATE option is not specified, the reorganization program checks that the update level in the reorganization description file matches the update level in the control file. If the levels do not match, the reorganization program aborts with an "UPDATE LEVEL MISMATCH" error if the reorganization program was compiled against an incompatible description file, or "CONTROL AND DESCRIPTION FILE UPDATE LEVELS INCOMPATIBLE" if the reorganization program is run using an incompatible description file.

CHANGES TO GENERATE STATEMENT

When the UPDATE option is specified to BUILDREORG, the GENERATE statement is now optional. If the data set is REORGANIZED in the previous DASDL compilation, then the default generate for the data set is "GENERATE <data set name>" and need not be given.

Also, the syntax of the GENERATE statement has been modified so that generation of the data set need not be specified first. The new syntax is as follows:

<generate statement>

```

-- GENERATE -----|-----
|                   |-----| |
|                   |<data set>|
|                   |-----|
|                   |<order option>|
|                   |-----|
|<set>|-----|
|   |<set>|-----|
|   |-----|
|<subset>|-----|
|   |<subset>|-----|
|   |-----|

```

NEW SYNTAX FOR REORGANIZATION SPECIFICATIONS

The following is the updated syntax for specifications to BUILDREORG:

<reorganization specifications>

```

-- UPDATE ; ----->
|                   |<generate statement>|
|<generate statement>|-----|
>
|<-----|
|<index control option> ; --|
>
|<-----| |<procedure sequence> ; -|
|<structure copy option> ; --|
>
|<-----|
|<reorganization global control> ; -|

```

D3088 BUILDREORG - "REORGANIZATION" LIMITATIONS

Generation of an unordered list from a generated RANDOM data set is now permitted. In addition, the significance of other REORGANIZATION limitations, as documented in the Mark 31 D-notes, has been minimized by providing implicit generates for certain structures where appropriate (see Mark 32 BUILDREORG note D3082, "Implicit GENERATE Statements").

D3582 BUILDREORG - INDEX CONTROL OPTION

The syntax diagram for <index control option> in the DMSII Utilities and Operations Guide (Form No. 5001803) incorrectly indicates that multiple <set>s or <subset>s may be specified, separated by commas. Only one <set> or <subset> can actually be specified per <index control option>; however, multiple <index control option>s are permitted.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - COPY AUDIT

P2956 COPYAUD-II - PRINT TAPE LABELS

UTILITY and COPYAUDIT will now print tape labels via the ADM EVENT PRINTLABEL mechanism. Previously, tape labels were not produced because these programs reverse verified their tapes. Now, tape labels are printed as soon as the tapes are opened (unlike normal tapes which have their labels printed when closed).

P3384 COPYAUD-II - ALPHANUMERIC USERCODES

COPYAUDIT produced spurious syntax errors when it was given alphanumeric usercodes beginning with digits. COPYAUDIT now accepts any valid usercode. COPYAUDIT has also been modified to accept data base names containing two or more hyphens. Previously these were unacceptable to COPYAUDIT.

DMS II - DASDL

D3048 DASDL - ALLOW LINK TO EMBEDDED DS

DASDL was erroneously giving a syntax error for a link to an embedded data set whose owner is the same as the owner of the link. This problem has been corrected.

Example:

```
D DATA SET
(
.
.
.
E DATA SET
(
.
.
.
);
L IS IN E WITH NO PROTECTION;
);
```

The following changes should be made to the DMSII DASDL Reference Manual (Form No. 5001480):

Add the following subparagraph to the second paragraph on Page 4-86 before the example:

"c. Owned by <data set> containing the link."

The first paragraph on Page 4-87 should now read as follows:

"<data set> A has no ancestors. A link in A can reference the disjoint <data set>s A or X or the embedded <data set>s B or C."

Subparagraph b on page 4-87 should now read as follows:

"b. If the link references a disjoint <data set>, then the link may point at any record in the <data set>. If the link references an embedded <data set>, then only certain records in the <data set> may be referenced. The record being referenced must be owned by the record containing the link or by an ancestor of the record containing the link. (An ancestor is the owner, the owner's owner, etc.)."

D3086 DASDL - GUARDFILES STORED UNDER USERCODE DIRECTORIES

When guardfiles used to secure data bases and logical data bases are stored under a usercode directory, the usercode must be included in the DASDL specification of the guardfile title. Omission of the guardfile's usercode in DASDL will result in DMOPENERERROR #26 (SECURITY ERROR ON OPEN). However, guardfiles stored under the system directory do not require an asterisk (*) as part of the guardfile title specification.

D3099 DASDL - INDEPENDENCE OF "DASDL UPDATE" COMPILATIONS

DASDL UPDATE may now be performed without immediately affecting the data base control file. DASDL no longer checks the old description file for compatibility with the control file. The control file is updated only after it is input to a successful run of SYSTEM/DMCONTROL("UPDATE") or a REORGANIZATION program generated with the UPDATE option. In either case, the control file is locked, the new description file is checked for compatibility with the control file, and the control file is updated. Once the control file is unlocked, user programs and data base system software compiled against the new description file can run.

D3115 DASDL - "COBOL" RESERVED WORD TABLE UPDATED

The 29 level COBOL reserved words tables has been replaced with the 32 level table.

D3117 DASDL - ALLOW "PROPERTIES" LABEL EQUATION

The DATABASE/PROPERTIES file may now be label-equated when a data base is compiled with DASDL, as follows:

COMPILER FILE PROPERTIES = <alternate file>;

D3244 DASDL - "AREASIZE" FOR DATA SETS

AREASIZE for data sets cannot be specified in BLOCKS as described in the DMSII DASDL Reference Manual (Form No. 5001480). The example on pages 6-14 and 6-15 should read as follows:

Change <data set physical options>.

ORIGINAL DASDL

```

D DATA SET
(
  A ALPHA(10);
  B BOOLEAN;
  N NUMBER(S5,2);
  R REAL;
)
VERIFY R GEQ 0;

S SET OF D
KEY A;

D
(
  AREAS = 10,
  AREASIZE = 500 RECORDS,
  BLOCKSIZE = 5 RECORDS,
  CHECKSUM = FALSE
);

```

Compile with DASDL.

```

* UPDATE;
D DATA SET
* REORGANIZE(ITEMS SAME)
(
  A ALPHA(10);
  B BOOLEAN;
*   N NUMBER(S5,2) REQUIRED;
  R REAL;
)
*   VERIFY B OR R GEQ 0;

S SET OF D
KEY A;

D
(
  AREAS = 10,
*   AREASIZE = 1000 RECORDS,
*   BLOCKSIZE = 20 RECORDS,
  CHECKSUM = TRUE
);

```

STEP #2

Generate and run Reorganization.

STEP #3

```

Compile with DASDL.
* UPDATE;
D DATA SET
(
  A ALPHA(10);
  B BOOLEAN;
*   N NUMBER(S5,2) REQUIRED;
  R REAL;
)
*   VERIFY B OR R GEQ 0;

S SET OF D
KEY A;

D
(
  AREAS = 10,
*   AREASIZE = 1000 RECORDS,
*   BLOCKSIZE = 20 RECORDS,
  CHECKSUM = TRUE
);

```

D3316 DASDL - CRUNCH "NEWTAPE" FILE

New symbolic files produced by DASDL will now be crunched, have a blocksize of 420 words, and an areasize of 1008 records.

D3441 DASDL - BETTER "CONTROLPOINT, SYNCPOINT" DEFAULTS

DASDL will now give better defaults for SYNCPOINT and CONTROLPOINT. SYNCPOINT will default to 100 transactions per syncpoint. CONTROLPOINT will default to 2 syncpoints per controlpoint.

D3453 DASDL - "28" TO "29" CONVERSION OPTIONS REMOVED

The INITPARTITIONS option has been removed from DATABASE/WFL/COMPILEACR. Previously, setting the INITPARTITIONS option caused DMCONTROL to be run with the option "INITIALIZE PARTITIONS". DMCONTROL, in turn, ran PARTITIONCONTROL to initialize the PARTITIONINFO data set.

D3458 DASDL - CHANGES TO "DASDL" REFERENCE MANUAL

Page F-1 of the DASDL Reference Manual should include a footnote beside the COBOL and PL/I representations for Real (n) and Real (n,m).

The footnote should read as follows:

"Real data items occupy one word and cannot be used to store normalized 12 digit numbers greater than 549,755,813,887. PL/I gives a warning for real items with a precision equal to 12. COBOL converts all real items with a precision of 12 to precision 11."

D3558 DASDL - "FIND NEXT, FIND PRIOR"

Page C-9 of the DASDL Reference Manual (Form No. 5001580) is incorrect. "General Restrictions", paragraph b, should be changed to read as follows:

"b. FIND NEXT and FIND PRIOR are not allowed for embedded STANDARD and COMPACT data sets."

D3568 DASDL - EMPTY BLOCK LIST

Page C-7 of the DASDL Reference Manual (Form No. 5001480) is incomplete. Compact data sets should be included in the list of structures using a one-way linked list to keep track of available blocks.

D3621 DASDL - "LOCKTOMODIFYDETAILS"

If a data set has the DASDL option LOCKTOMODIFYDETAILS set, INQUIRY cannot be used to create, delete or modify any of its descendants.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - DASDL

P2951 DASDL - ALLOW MODULUS SPECIFICATION FOR ACCESS

Modulus may now be specified for an access to a random data set via a physical specification statement. This statement would appear as follows:

ACCESSNAME (MODULUS=<unsigned integer>);

A physical specification statement for an access gave syntax errors on the 30 release and caused an INVALID INDEX in DASDL on the 31 release.

P2952 DASDL - SEQUENCE NUMBER NOT UPDATED

When the LIST and NEW dollar options were both RESET, the SEQNUMBER variable was not updated. The progress of a DASDL compile could not then be determined using the ?CS message. The SEQNUMBER variable is now updated properly.

P2953 DASDL - "SERIALBUFFERS" ATTRIBUTE

DASDL was unconditionally assigning the SERIALBUFFERS attribute the value zero when REBLOCKING was RESET. DASDL now only computes a default value for SERIALBUFFERS if no explicit specification is made.

P2967 DASDL - "AREASZ" GREATER THAN "65536" TRUNCATED

DASDL was inadvertently truncating AREASZ if it was greater than 65536. This problem has been corrected on the 32, 31 and 30 releases. If the value specified is greater than 65536, the value printed for AREASZUSER will be incorrect on the 31 release; to avoid compatibility problems, this error will only be corrected completely on the 32 release.

P3072 DASDL - KEYCHANGED TEXT FOR FIELD ITEM

DASDL now uses the "ISNT" relational operator rather than "NEQ" to detect if field key items have changed. Since all bits of fields are significant, the "NEQ" arithmetic comparison would not always produce correct results.

P3102 DASDL - SET UP "FILEKINDF," PACKNAME CORRECTLY

DASDL was incorrectly setting the packname and filekind for data sets, sets, the control file and the audit trail. If DISK or KIND=DISK were specified in the physical attributes, DASDL would erroneously assign the file the packname given in the DEFAULTS specification, if one existed. This problem has been corrected.

Now, when KIND=DISK is specified, DASDL assigns the file the attributes KIND=PACK and PACKNAME=DISK. When KIND=PACK is specified, DASDL assigns the file the attributes KIND=PACK and PACKNAME=PACK.

If a "FILE LOCATION CHANGED" warning is received on the next DASDL update and the old attributes are desired, explicit specifications for the file kind and packname should be given.

P3124 DASDL - SUPERFLUOUS TOO MANY AREAS MESSAGE

DASDL will no longer produce a superfluous "NUMBER OF ROWS IS GREATER THAN 1000" error message when the calculated value for areas is too large.

P3181 DASDL - BAD EXPAND TEXT FOR STORED ITEMS

DASDL generated bad expand text for compact data set records where the stored optionally items bit mask was split across a word boundary. This problem has been corrected.

P3231 DASDL - CALCULATE REASONABLE DEFAULT "REBLOCKFACTOR"

When REBLOCK was specified and no REBLOCKFACTOR was given, DASDL incorrectly set REBLOCKFACTOR to 1. DASDL will now calculate a reasonable default value for REBLOCKFACTOR based on the BLOCKSIZE and AREASIZE.

P3241 DASDL - LOOP AFTER MISSPELLED UPDATE CARD

A misspelled DASDL update card followed by a defaults specification could cause the DASDL compiler to loop indefinitely. This problem has been corrected.

P3248 DASDL - CORRECT HANDLING OF "B7700" DOLLAR OPTION

DASDL was erroneously syntaxing the specification of the B7700 dollar option when it appeared on the same line with other dollar option specifications. This problem has been corrected.

P3315 DASDL - POSSIBLE BUFFER OVERLAY

On a DASDL update, the DASDL compiler could inadvertently overlay a description file buffer when assigning set physical attributes. DASDL would then emit erroneous error messages based on the corrupted buffer. This problem has been corrected.

P3342 DASDL - "BAD SELECT/VERIFY TEXT FOR FIELD BOOLEANS"

If a boolean item within a hidden field was used in a select or verify expression, then DASDL produced bad select text. This problem has been corrected.

P3379 DASDL - DISALLOW RESTARTS "DS" NAMED "RECOVERY"

Previously, DASDL permitted restart data sets named "RECOVERY". This caused syntax errors when compiling RECOVERY. DASDL will now give a syntax error in this situation.

P3385 DASDL - BAD EXPANDTEXT

DASDL was generating bad expand text for any boolean that was the first item in a stored depending on group. The generated replace statement had a count part equal to the boolean's record offset plus one. This problem has been corrected.

P3386 DASDL - "MOVES LIST EXCEEDED" ERROR

While generating ALGOL REORG move text for DASDL reorganization, the DASDL compiler could fill up a compile time move table and emit the error message "MOVES LIST EXCEEDED". This problem has been corrected by allowing DASDL to resize this table when it becomes full.

P3442 DASDL - ERRONEOUS INITIAL VALUES FOR REMAP

Due to an error in DASDL, an initial value specification in a remap could be carried over to subsequent remaps of the same data set. This problem has been corrected.

P3487 DASDL - "EOF NO LABEL" ERROR

Specifying a remap item which was not in the data set caused DASDL to give the error "ITEM NOT IN DATASET". Due to poor error recovery, an "EOF NO LABEL" error could subsequently occur on Propfile. This no longer occurs.

P3699 DASDL - DISALLOW REORGANIZATION ACROSS RELEASES

Since the REORGANIZATION program cannot handle description files from earlier releases, DASDL updates from one release to another cannot include REORGANIZATION. The DASDL compiler now gives an error message when an attempt is made to perform a reorganization across releases.

P3700 DASDL - IDENTIFIERS ENDING WITH A HYPHEN

DASDL now ensures that identifiers do not end with a hyphen.

P3756 DASDL - "BLOCKSIZE TOO SMALL" FAULT

If the blocksize in a variable format data set is too small, the DASDL compiler correctly gives a syntax error, and then fails with a DIVIDE BY ZERO fault. This problem has been corrected.

P3758 DASDL - CREATION OF DATA BASE UNDER "*" DIRECTORY

DASDL currently allows the user to create a description file under the system directory using the following statement:

* <data base name> WITH DASDL LIBRARY.

However, the files created by SYSTEM/DMCONTROL and DATABASE/WFL/COMPILEACR were previously put under the usercode of the DASDL compilation. This has been corrected. The control file and DMSII system software will now be put under the system directory provided the DASDL compilation is done under a privileged usercode.

P3790 DASDL - LIMIT ERROR ON RESTART DATA SET

DASDL failed to allocate enough space for restart data sets with small populations or small record sizes. The first attempt to store into the restart data set would result in a LIMIT ERROR exception. DASDL now allocates one extra block for the restart data set because block zero is never used to store restart records.

P3793 DASDL - AVERAGE RECORD SIZE DURING UPDATE

DASDL now ensures that average record size is not changed for simple update. Average record size may only be changed when REORGANIZE(ITEMS CHANGED) or REORGANIZE(ITEMS SAME) is specified.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - DBANALYZER

D3369 DBANALYZER - "DBANALYZER" IMPLEMENTATION

INTRODUCTION

Information breeds information. Probably there is nowhere that this is more true than with data bases. In particular, the information concerning the data base itself is growing, with important implications in the areas of:

- Data Base Design
- Data Base Modification
- Data Base Tuning
- Data Base Reorganization

DBANALYZER provides a convenient method by which a Data Base Administrator can collect information concerning a DMSII data base. The basic function of DBANALYZER is to analyze the logical and physical structure of a DMSII data base, and to report this by data base structure. DBANALYZER is a product complementary to the run-time dynamic analysis provided by the DMSII STATISTICS option in that DBANALYZER provides static analysis of the data base files.

DBANALYZER may be used to analyze any DMSII data base, providing the data base description and required data base files are available. DBANALYZER examines the DMSII data base description file and uses that information to provide an intelligent analysis of each structure, depending on the structure's characteristics.

DBANALYZER provides the following information:

A. Structure Identification

This includes the structure's name, number, type, relation to other structures, and DASDL comment.

B. Dependent Structures

All sets of each data set are listed.

C. File Attributes

Physical File attributes of the file are provided.

D. DASDL Attributes

DBANALYZER lists structure attributes from the DMSII description file concerning how the ACCESSROUTINES reference the file.

E. Text

All DASDL generated source statements which are included in the ACCESSROUTINES are extracted from the description file and printed.

F. File Analysis

This analysis is very dependent on the structure type, but generally includes an analysis of the available space, control information, and data within the structure. Complete analysis may require that the entire file be read. For many structures, partial analysis is provided. The extent of the analysis performed can be controlled by the user.

G. Analysis Time

A summary of the processor, I/O and elapsed time required to perform the analysis is provided.

DBANALYZER may be used interactively from a terminal or it may be initiated as a batch job. In either case, free format input commands to DBANALYZER control the scope and depth of analysis performed. These commands select which structures are analyzed and which information is produced for each structure.

The user is expected to have a fairly detailed knowledge of the data base, and of the various storage algorithms utilized by the DMSII Data Management System. It is generally true that the more knowledgeable the user is, the more the analysis of the data base is of benefit.

DBANALYZER does not allow a user to examine the actual data base information itself. Since DBANALYZER does not provide access to the user information stored in a data base, no explicit consideration has been given to the security of the data base, other than that security normally provided at the file level by the MCP.

COMPILING DBANALYZER

The following procedure can be used to compile DBANALYZER.

A. Copy and compare SYMBOL/DBANALYZER and DATABASE/PROPERTIES to disk from the DMSII system tape.

B. Compile SYSTEM/DBANALYZER using the following WFL statements.

```
<i> BEGIN JOB COMPILEDBANALYZER;
      COMPILER SYSTEM/DBANALYZER WITH DMALGOL LIBRARY;
      COMPILER FILE TAPE (TITLE=SYMBOL/DBANALYZER);
      COMPILER DATA
      $ SET MERGE LINEINFO
      $ RESET LIST XREF
<i> END JOB
```

RUNNING DBANALYZER

DBANALYZER can be run interactively at a terminal using CANDE or as a batch job through WFL.

INTERACTIVE MODE

To run DBANALYZER in interactive mode, first log on to CANDE and then type:

```
RUN $SYSTEM/DBANALYZER;
FILE DASDL(TITLE=DESCRIPTION/<data basename>ON<packname>);
```

DBANALYZER reads and interprets the DASDL description file; therefore, this file must be equated to the current description file for the data base.

In interactive mode, all input commands are entered at the terminal. Each command is executed when it is entered.

DBANALYZER displays a # each time it is ready for additional input. Normally, commands can be entered on a single line; however, some commands may exceed the width of the terminal. A % character, entered at the end of the line, indicates that more input will follow. When the line is transmitted, DBANALYZER responds with # and additional input may be entered. This process may be repeated as many times as necessary. When the command is complete, the last line can be transmitted without a % and the entire command is executed. Several commands may be contained in a single input message provided each command is followed by a semicolon.

Output is normally displayed at the terminal. However, by using the OPTION command, output may be directed to either the terminal or the printer. When a terminal page length greater than zero is specified, DBANALYZER displays a full page of output and then waits for a message containing one or more blanks to be input, before transmitting the next page. If a non-blank character is transmitted, all remaining output is discarded. DBANALYZER displays a #, and then awaits a new input command.

BATCH MODE

To run DBANALYZER in batch mode, a card deck or JOBSYMBOL file must be made containing WFL statements similar to those below:

```
<i>BEGIN JOB DBANALYZER;
      RUN SYSTEM/DBANALYZER;
      FILE DASDL (TITLE=DESCRIPTION/<data basename>on<packname>);
      DATA CARD
      <one or more DBANALYZER input commands>
<i>END JOB
```

DBANALYZER reads and interprets the DASDL description file; therefore, this file must be equated to the current description file for the data base.

By default, input commands are expected to be in a card file with internal name CARD. This file may be equated to a disk file using WFL file equation. When DBANALYZER commands are entered from cards or disk, only the first 72 characters of each record may contain commands. The remaining columns are reserved for sequence numbers or comments. A command may extend over as many records as desired; however, no word may be split between two records. Every command must be terminated with a semicolon. Several commands may be contained on a single input record provided each is delimited by a semicolon.

A percent sign (%) in the input denotes a comment. Anything between the percent sign and the end of the record is ignored (except when the percent sign is enclosed in quotation marks such as "ABC%DE").

In batch mode, DBANALYZER output is sent to a printer file with the internal name LINE.

COMMON SYNTACTIC ITEMS

< l e t t e r >

-- one of the EBCDIC characters, A through Z --

<digit>

-- one of the decimal digits, 0 through 9 --

<integer>

```

|<-----|
-----<digit>-----|

```

Semantics

<integer>s are used to represent unsigned whole values.

< i d e n t i f i e r >

```

--<letter>-----
|
| |<-----|
| |-----/15\-----<letter>-----<letter>-----
| |-----<digit>-----<digit>-----
| |-----
|

```

Semantics

```
<identifier>s have no intrinsic meaning.  They are used to represent symbolic names of
structures within a DMSII data base.
```

An **<identifier>** is composed of from 1-17 **<letter>**s, **<digit>**s and hyphens. The first character must be a **<letter>**. The last character must not be a hyphen.

Examples

A
EMPLOYEE
ACCOUNTS-PAYABLE
B-1

<string>

```

-- " --- any EBCDIC character except quote --- " ---

```

Semantics

`<string>s` must be contained within quotes. A string can contain at most 255 characters.

DBANALYZER COMMANDS

Syntax

<dbanalyzer commands>

```

|<----- ; -----|
-----<analyze command>-----|
|
| -<help command>-----
| -<option command>----
| -<printer command>---
| -<quit command>-----
| -<terminal command>--

```

Semantics

ANALYZE Causes DBANALYZER to analyze one or more data base structures and display the results of the analysis.

HELP Displays the syntax and semantics for DBANALYZER input commands.

OPTION Allows the current output device to be displayed or altered.

PRINTER Allows the attributes of the line printer file to be displayed or altered.

QUIT Terminates the DBANALYZER program.

TERMINAL Allows the attributes of the terminal file to be displayed or altered.

ANALYZE Command

Syntax

<analyze command>

```

-- ANALYZE --- DATABASE -----|
|
| - ALL -----|
| -<structure list>-- | -<analyze options>--

```

<structure list>

```

|<----- , -----|
-----<structure name>-----|
|
| - .<partition name> ----
| -<structure number>-----
| - .<partition name> --
| - <structure number> - <structure number> -

```

<analyze options>

```

-- ( OPTIONS =  | <----- , ----- |
                |----- ALL -----| ) --|
                |
                | - DASDL -----|
                |   | - ATTRIBUTES - |
                |-----|
                | - DATA -----|
                |-----|
                | - FILE -----|
                |   | - ATTRIBUTES --|
                |-----|
                | - SETS -----|
                |-----|
                | - STRUCTURE -----|
                |-----|
                | - TABLE -----|
                |-----|
                | - TEXT -----|
                |-----|
                | - TIME -----|
                |-----|

```

Semantics

The ANALYZE command causes DBANALYZER to analyze one or more data base structures and display the results of the analysis.

ANALYZE DATABASE

ANALYZE DATABASE displays data base attributes including software levels, timestamps, parameters, options, and audit attributes. A list of all structures is also produced which contains the structure number, name, and type for each valid structure.

ANALYZE <structure list>

A <structure list> is used to select particular structures for analysis. <structure list>s may contain one or more <structure names>, <structure numbers>, or structure number ranges separated by commas. <structure name>s must be <identifier>s and must be the DASDL specified names of valid structures. <structure number>s must be <integer>s and must refer to valid data base structures. When a structure number range is specified, all valid structures within the range are selected. Not all structure numbers in the range need be valid.

For partitioned structures, a <partition name> must follow the <structure name> or <structure number>. This causes the specified partition to be analyzed. <Partition name> is the value of the partition key. The <partition name> may be either an <identifier> or a <string>. A <string> must be specified if the first character of the partition key is not alphabetic or if the partition key contains any special character.

ANALYZE ALL

When ALL is specified, analysis is performed on the data base and on all valid structures.

<Analyze Options>

<Analyze options> control which forms of analysis are performed. ALL is assumed by default when no <analyze options> appear.

ALL

ALL causes complete analysis to occur.

DASDL ATTRIBUTES

When this option is selected, DASDL assigned structure attributes are listed from the description file. These attributes vary slightly according to the structure type, but include all of the following which are appropriate:

```

Accesses
-Modulus

Data Sets
-Allocate          -Open Partitions
-Areas             -Options
-Areasize          -Population
-Blocksize         -Reblockfactor
-Buffers           -Recordsize
-Modulus           -Subblocksize

Sets and Subsets
-Areas             -Modulus
-Areasize          -Open Partitions
-Buffers           -Options
-Key Data Size     -Population
-Key Entry Size    -Tablesize
-Key Size
-Loadfactor

```

DATA

DATA is only meaningful for data sets. When the DATA option is specified, more detailed STRUCTURE analysis is performed. Generally, this option causes the entire structure to be read and analyzed.

FILE ATTRIBUTES

When this option is selected, the physical attributes of the file associated with the structure are listed. These attributes include:

```

-Title
-Format timestamp
-Version timestamp
-Creation date
-Access date
-Areas
-Areasize
-Blocksize
-Recordsize
-Lastrecord

```

SETS

SETS is only meaningful for data sets. When SETS is specified, DBANALYZER lists all sets, subsets, and accesses which reference the data set.

STRUCTURE

When STRUCTURE is specified, the contents of data base structures are analyzed. The depth of analysis performed is controlled by the DATA option for data sets and the TABLE option for index sets. Normally when these options are reset, only control information is examined; otherwise, the entire structure is read and analyzed.

The analysis performed depends upon the structure type. Complete analysis generally produces the following information:

```

DATA SETS
-File size
-Block Utilization
-Data block analysis including available space,
  control information, and data.
-Structure control information

INDEX SETS
-File size
-Block utilization
-Table block analysis including available space,
  control information, and table entries.
-Bias relative to the referred data set
-Structure control information

```

TABLE

TABLE is only meaningful for index sets. When the TABLE option is specified, more detailed STRUCTURE analysis is performed. Generally, this option causes the entire structure to be read and analyzed.

TEXT

When TEXT is specified, all DASDL generated source statements which are included in the ACCESSROUTINES are extracted from the description file and printed.

TIME

When TIME is specified, the processor, I/O, and elapsed time required to analyze each structure are reported.

Examples

ANALYZE ALL

ANALYZE D, S, E.1977, E.1978(OPTIONS=DASDL,FILE,TEXT)

ANALYZE D,10-17,19

HELP Command**Syntax**

<help command>

```
-- HELP -----|
      |           |
      | --<dbanalyzer command>-- |
      | --<metalinguistic id>--  |
```

Semantics

HELP displays information about DBANALYZER input commands

When HELP alone is entered, the DBANALYZER commands are listed.

When HELP <dbanalyzer command> is entered, the syntax of the specified command is listed.

HELP <metalinguistic id> causes the syntax of the selected metalanguage identifier to be displayed.

Examples

```
HELP
HELP ANALYZE
HELP <structure list>
```

OPTION Command**Syntax**

<option command>

```
-- OPTION -----|
      |           |
      | - PRINTER - |
      | - TERMINAL -|
```

Semantics

When OPTION alone is entered, the current options are displayed.

OPTION PRINTER causes output to be listed on the system line printer. The number of lines per page and the number of characters per line may be specified using the PRINTER command.

OPTION TERMINAL causes output to be listed on the terminal.

Example

OPTION PRINTER

PRINTER Command**Syntax**

<printer command>

```

-- PRINTER -----|
|<-----, -----|
|PAGE -----<integer>-----|
|  - - - -|
| - WIDTH -----<integer>-|
|  - - - -|

```

Semantics

PRINTER displays or alters the current printer attributes.

When PRINTER alone is entered, the current printer attributes are displayed.

PAGE specifies the number of lines displayed per page.

WIDTH specifies the number of characters displayed on each line.

The printer attributes are only significant when OPTION PRINTER is set (see the OPTION command).

Example

PRINTER WIDTH=132, PAGE=50

QUIT Command**Syntax**

<quit command>

```

---- QUIT ----|
| - BYE - -|
| - END - -|
| - STOP -|

```

Semantics

QUIT terminates the DBANALYZER program.

Example

QUIT

TERMINAL Command**Syntax**

<terminal command>

```

-- TERMINAL -----|
|<-----, -----|
|PAGE -----<integer>-----|
|  - - - -|
| - WIDTH -----<integer>-|
|  - - - -|

```

Semantics

TERMINAL displays or alters the current terminal attributes.

When TERMINAL alone is entered, the current terminal attributes are displayed.

PAGE specifies the number of lines displayed per page. If PAGE is greater than zero, DBANALYZER pauses after displaying each full page of output. DBANALYZER then waits for a message containing one or more blanks to be input before transmitting the next page. If a non-blank character is transmitted, all remaining output is discarded, DBANALYZER displays a #, and then awaits a new input command. If PAGE is zero, then output is not broken into pages.

WIDTH controls the number of characters displayed on each line.

The terminal attributes are only significant when OPTION TERMINAL is set (see the OPTION command).

Example

TERMINAL WIDTH=80,PAGE=23

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - DMALGOL

D3049 DMALGOL - "DMALGOL" EXTENSIONS FOR "DMINQ"

Extensions have been made to the DM INQUIRY functions described in Mark 31 P and D Notes Appendix E, "DMALGOL Implementation". See Appendix A, "DMALGOL Implementation", for a description of the 32 extensions, which are indicated by PCN bars in the right margin.

D3634 DMALGOL - "NODE <IDENTIFIER> *" DEIMPLEMENTED

The NODE <identifier> * construct was deimplemented on the Mark 30 release; consequently, the Mark 31 appendix for DMALGOL has been revised. See Mark 32 Appendix A, "DMALGOL Implementation", for details.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - DMALGOL

P3009 DMALGOL - CORRECT "'ERROR" CONSTRUCT

A SEG ARRAY error occurred during compilation of the DMALGOL construct 'ERROR. The following is an example of a program which would evoke this error.

Example:

```
BEGIN
  'ERROR "ANY STRING"
END.
```

This problem has been corrected.

P3323 DMALGOL - ELIMINATE EXTRANEIOUS "?"S FOR "'PRINT"

Formerly, the 'PRINT construct in DMALGOL would cause question marks to be printed surrounding the desired text output. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - DMCONTROL

D3100 DMCTL - UPDATE LEVEL CHECK FOR "RECOVER UPDATE"

A check is now performed for "RECOVER UPDATE" to ensure that the description file and the old control file have compatible update levels. If the description file is marked as requiring reorganization, the update levels must be identical. If the description file is not marked as requiring reorganization, then the update level of the old control file must be the same as or one less than the description file update level. If update levels are not compatible, the error message "GIVEN CONTROL FILE TOO OLD FOR PROPER CONTROL FILE RECOVERY" is printed and the "RECOVER UPDATE" is not allowed. When this occurs and a proper control file is not available, "RECOVER INITIALIZE" must be used to recover the control file.

D3101 DMCTL - AVOIDANCE OF "RECOVER INITIALIZE"

Recovery of the control file via the DMCONTROL "RECOVER INITIALIZE" function should be the recovery method of last resort. "RECOVER UPDATE" is the preferred method. A "RECOVER INITIALIZE" invalidates all existing dump tapes and subverts all data base coordination checks.

D3119 DMCTL - "28" TO "29" CONVERSION OPTIONS REMOVED

The following options, which permitted conversion from the Mark 28 to the Mark 29 DMSII software release, have been de-implemented:

INITIALIZE 29
INITIALIZE PARTITIONS
INITIALIZE PARTITIONS 29
CORRECT PARTITIONS

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - DMCONTROL

P2758 DMCTL - SYSTEM IDENTIFICATION

DMCONTROL now places the appropriate system identification in its printer heading.

P3108 DMCTL - "DMCONTROL" RESEQUENCED

The DMCONTROL symbolic has been resequenced.

P3175 DMCTL - CONTROL FILE "I/O" LOCK

A lock has been added to the control file module to enforce single-threaded I/O operation against the data base control file.

P3182 DMCTL - "CFDELETEPART" CORRUPTING CONTROL FILE

CFDELETEPART was directly altering the contents of the control file I/O buffer, potentially leading to corruption of the control file. The situation has been corrected.

P3185 DMCTL - INITIAL VALUE OF DESIGNATED SERIAL NUMBERS

Data bases that audit to tape using designated serial numbers were not using the initial value for serial numbers specified in the DASDL source. For example, the first audit file open would request a tape with SERIALNO="AUD001", even though the starting serial number was specified as "AUD000" in the DASDL source. The problem has been corrected.

P3249 DMCTL - MARK "32 DMS" ON MARK "31 MCP"

Data management software that includes the Mark 32 control file module from DATABASE/DMCONTROL can run under a Mark 31 MCP; however, the DMSII software must be compiled with a Mark 32 DMALGOL compiler, and any attempted data base equation will be ignored.

P3269 DMCTL - INVALID INDEX ON "OVERRIDE HL"

The DMCONTROL function "OVERRIDE HL" could result in an INVALID INDEX if an error occurred while trying to open the control file. This problem has been corrected.

P3285 DMCTL - "CF" TITLE FOR "OVERRIDE HL"

The SYSTEM/DMCONTROL function OVERRIDE HL now constructs a title for the data base control file from the description file if the control file title was not equated at run time.

This change permits all SYSTEM/DMCONTROL functions to work by equating the description file. Previously, the control file was equated for "OVERRIDE HL" and the description file was equated for all other functions. OVERRIDE HL will continue to work as before if the control file is equated.

P3316 DMCTL - REDUCE USE OF "REORGINFONODE"

Only the format time stamp is now taken from reorginfonode during a control file update for reorganization. Previously all structure attributes were taken from reorginfonode.

P3317 DMCTL - SET UP PREFIX ARRAYS

The control file module now sets up the control file name and prefix arrays only when they have not been previously set up. Prior to this, these arrays were setup every time CFOPEN was called.

P3343 DMCTL - STRUCTURE DETAILS

Structure records in the control file now contain the TYPEF (data set or index set) and SUBTYPEF (standard, index sequential, etc.) properties and the file format level number of each structure. The utility WRITE/LIST function will print the format level of each structure.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - MONITOR

D3378 MONITOR - DATA BASE MONITORING FACILITY

DATABASE MONITORING FACILITY

MONITOR is an interactive program which allows access to data base status and statistics and permits changes to data base options and parameters. In addition, data base information may be sampled and captured in a disk file for later off-line analysis.

Capabilities

1. Any Visible DBS message that would normally be input from the console may be input to the MONITOR; the response is returned to the remote terminal.
2. An automatic display mode (ADM) similar to that available for the Operator Display Terminals (ODT's) may be used to constantly display the data base status and statistics.
3. If the data base has the STATISTICS option set, this information may also be included in the MONITOR responses.
4. Output may be displayed on either a screen or hardcopy terminal. Terminal specifications may be modified via commands to the MONITOR.
5. Data base status and statistics may be captured in a disk file for the entire data base or selected structures. This sampling can be turned on and off via MONITOR during the course of a session.
6. A help facility is provided which describes the syntax for all MONITOR commands.

Generation

MONITOR may be compiled using the DATABASE/WFL/COMPILEACR job file. When compiling MONITOR, the dollar option "NOUPDATE" may be set to generate a version of MONITOR which allows inquiry into data base status and statistics information but does not allow any data base parameters or options to be changed. The symbol file for MONITOR is called DATABASE/MONITOR.

Operation

MONITOR may be run via CANDE. MONITOR has no parameters. MONITOR opens the data base inquiry so that it may use the INQUIRY interface to access data base information. Thus, data base statistics and status information reflect MONITOR as an inquiry user of the data base.

Commands

1. Any of the Visible DBS commands may be typed in at any time and the response is displayed on the remote terminal. The current Visible DBS commands begin with the following key words.

STATUS
ALLOWEDCORE
SYNCPOINT
CONTROLPOINT
STATISTICS
STRUCTURE
AUDIT

The ALLOWEDCORE, SYNCPOINT, CONTROLPOINT, STATISTICS, STRUCTURE, and AUDIT commands are prohibited if NOUPDATE is specified at compile time.

2. The DISPLAY command allows information to be displayed on the remote terminal.


```

<display command>
  -- DISPLAY -- <display type> -----|

<display type>
  --- STATUS -----|
  | - MESSAGES -----|
  | - STATISTICS -----|
  |   | -<stats selection>-|

<stats selection>
  --- DATABASE -----|
  | - STRUCTURES --<structure list>-|

<structure list>
  --- ALL -----|
  |   | <----- , -----|
  | - ( ---<structure name>----- ) -|
  |   | -<structure number>-|

```

STATUS returns information similar to the Visible DBS command STATUS including number of inquiry and update users, etc, but in a form that is more suitable for the ADM.

MESSAGES displays up to 24 of the most recent ACCESSROUTINES messages. They do not include normal program display messages.

STATISTICS can only be used when the STATISTICS option is set for the data base.

STATISTICS DATABASE displays data base level statistics. If the data base is audited, audit and transaction statistics are included.

STATISTICS STRUCTURES displays statistics concerning the selected structures. This includes the number of reads, writes, random users, serial users, small buffers, big buffers, etc.

STATISTICS by itself displays summary statistics for all structures. This includes the total reads and total writes against the data base.

To obtain most of this information from the MONITOR, the data base must have been compiled with the STATISTICS option.

3. The ADM command allows constant display of information on the remote terminal.

```

-- ADM -----|
  | <----- , -----|
  | - ( ---<display options>--- ) -|
  |   | -<delay>-|
  | - STOP -----|
  | - GO -----|
  | - -----|

<display options>
  | <-- , -- <number> ---|
  | -<display type> -----|
  |   | - <number> -|

```

```

-----<number>--|
| - DELAY - |

```

ADM by itself lists the current ADM specification.

ADM STOP or **ADM GO** suspends or resumes automatic display.

ADM - cancels the current ADM specifications and restores the default. The ADM is left suspended.

4. **TERM** alters or displays the remote terminal specifications.

```

-- TERM
|
| |<-----|
|
| LINES -----<number>
|
| | WIDTH |
|
| | FIRST |
|
-- SCREEN -----
-- HARDCOPY -----
-- TD820 -----
-- TD830 -----

```

LINES specifies the number of lines on the device. (Default = 24.)

WIDTH specifies the device width in characters. (Default = 80.)

FIRST specifies the line at which display is to begin. (Default = 2.)

SCREEN, HARDCOPY, TD820 and TD830 indicate the type of the remote device. By default, the device type is SCREEN if the station is marked such in the NDL specification and HARDCOPY otherwise. Specifying TD820 or TD830 causes MONITOR to take advantage of the highlighting features of these terminals.

5. The **CAPTURE** command allows statistics information to be captured for the entire data base or for selected structures.

<capture command>

```

-- CAPTURE -----
> -----
| -<stats selection>- | | -<capture options>- | | -<delay>- |
|-----|

```

<capture options>

```
|<-----|
----- FILE --<filename>-----|
```

CAPTURE with no qualification causes all statistics information to be captured and stored in a disk file called <data base name>/MONITORSTATS/LSN<lsn>/<date-time>/<seq.no.>. <lsn> is the logical station number of the terminal originating the MONITOR. <seq.no.> is a sequence number that starts as 001 and increments by 1 for each file created during a particular capture session. If a structure list is specified, only statistics for the given structures are captured. If the FILE clause is used, the statistics file is created under that name followed by a sequence number.

CAPTURE - causes sampling to be terminated.

The disk file created is a FILETYPE 4 disk file with the first word of each record being the size of the record. The second word is a TIME(6) value of the time the statistics record was written. The rest of the record is the result of the statistics request from the ACCESSROUTINES. This format is defined in ACR note D3045 - Statistics Interface. The internal name for the CAPTURE file is CAPTUREFILE.

6. HELP displays the syntax and semantics of MONITOR commands.

```
-- HELP -----|
|               |
| - <Monitor command> ---|
| - <metalinguistic id> -|
```

HELP by itself displays all MONITOR commands.

HELP <Monitor command> displays the syntax and a short explanation of the specified command.

HELP <metalinguistic id> displays the syntax and a short explanation of the specified metalinguistic identifier. For example, "HELP <structure list>".

7. END or its synonyms terminate the MONITOR.

```
--- END -----|
|               |
| - BYE --|
| - STOP -|
| - QUIT -|
```

DMS II - DUMPDIR

D3353 DUMPDIR - "DUMPDIR" ENHANCEMENTS

PURPOSE

Mark 32 SYSTEM/DMDUMPDIR provides more automatic data base recovery. This will be referred to as automatic tape recovery.

Previously, to recover a data base from dump tapes, UTILITY required a list of the dump tapes to be utilized. UTILITY now relieves the user of this task by using the SYSTEM/DMDUMPDIR library to determine which dumps are needed.

AUTOMATIC TAPE RECOVERY

Restrictions on Usage

Reconstruct.

All forms of reconstruct can use the Dump Directory.

Rebuild.

This feature can be used for two of the three types of rebuild:

1. Where a <date-time point> is given for the ending point,
2. Where THRU AUDIT is specified and the most current dumps are used.

Rebuild to <boj/eoj point> cannot use the Dump Directory because this does not provide the recovery ending point time.

Usage Considerations

When a rebuild or rollback is done, dumps may be invalidated.

	-----[dump d1]--	--[dump d2]-----	
(time)	t0	t1	t2

If a data base that is at t2 in time is rebuilt or rolledback to t1, dump d2 is no longer valid.

When RECOVERY has finished, it will delete dumps which are no longer valid.

If any type of manual recovery is done that will make invalid dumps valid or valid dumps invalid, these dumps must be manually added or deleted from the Dump Directory using SYSTEM/DMDUMPDIR.

UTILITY Syntax

Automatic tape recovery is assumed if no <sourcelist> is given, the Dump Directory system has been enabled and all restrictions on usage are met.

If THRU AUDIT is specified, FROM MOST CURRENT must be specified in place of a <sourcelist> for automatic tape recovery to occur.

RECOVERY OF DUMP DIRECTORY FILES

If the Main Directory file is lost, it can be recovered by

1. ENABLEing the Dump Directory System, causing a new Main Directory file with no Dump Directory entries to be built.
2. Dump directory entries for existing Dump Directory files may then be ADDED to the Main Directory.

If Dump Directory files are lost, they can be recovered by using UTILITY to recreate them from UTILITY dump tapes. Dump Directory entries are automatically put into the Main Directory by this function.

UTILITY syntax.

The following syntax rebuilds a Dump Directory file and inserts a Dump Directory entry in the Main Directory from each UTILITY dump tape specified in <tapelist>.

```

BUILDDUMPDIRECTORY ----- <tapelist> -----|
-----|
<tapelist>
|-----,-----|
--- <tape name> -----|
|
|-----,-----|
|-(---/1--- VERSION = <integer> -----)-|
|   -/1--- CYCLE = <integer> -----|
|   -/1--- SERIALNO = --- <integer> -|
|               | - <string6> -|

```

The version, cycle and serialno of the last reel made by the UTILITY dump should be given; otherwise, when automatic tape recovery is done:

1. More reels of the dump than necessary may have to be processed to load the desired rows
2. The list of the tapes needed to load the desired rows may not contain every cycle and version of the tape that will be needed.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - DUMPDIR

P3372 DUMPDIR - DESCRIPTION FILE TITLE

When DUMPDIR is run without a usercode with a non-usercode description file, the message "DESCRIPTION FILE TITLE MUST BE IN THE FORMAT" was erroneously displayed. This problem has been corrected. This failure was most likely to occur when DUMPDIR was run from the ODT console.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - DUMPDIR/LIBRARY

P3488 DUMPDIRLIB - "RETAIN" CORRECTED

RETAIN was keeping the oldest dumps rather than the newest. This problem has been corrected.

P3489 DUMPDIRLIB - ERROR USING "WRITE=/LIST="

Data bases containing remaps or accesses failed with a SEG ARRAY error when WRITE= or LIST= was used. This no longer occurs.

DMS II - INQUIRY

D3171 INQ - "INQ CREATE/DELETE"

Simple record creation and deletion is now provided in INQUIRY. This new capability along with the present query, reporting, and update capabilities in INQUIRY provides a completely "programmerless" user interface to DMSII. For a simple data base application, INQUIRY may be all that is necessary to create, update, and utilize the data base.

Record creation, deletion, and update in INQUIRY does not provide for the maintenance of links and manual subsets. These must be maintained programmatically if they exist in the data base.

For audited data bases, an INQUIRY create, delete, or update operation is done as a single transaction with a sync-point forced on the ENDTRANSACTION. This prevents suspension of application programs due to terminal "think time." In addition, it guarantees that if there are no errors during the transaction, the modification to the data base occurs and is not backed out if some program aborts or a Halt/Load occurs. If any errors are encountered, the user is notified and the modification to the data base is not done.

CREATE STATEMENT

Syntax:

```
-- CREATE --<create specification>--|  
--
```

<create specification>

```
-----<create list>--|  
|<data set name>-----|  
| - ( <record type> ) - | | - WITH - |
```

<create list>

```
|<-----, -----|  
|<alpha item>-- = --<string>-----|  
|<arithmetic item>-- = --<arithmetic expression>--|  
|<boolean item>-- = --- TRUE -----|  
| - FALSE -----|  
|
```

Examples:

```
CREATE B1 = "ABC", B2 = 500, B3 = TRUE
```

```
CREATE X WITH X1 = 34
```

```
CREATE VAR-FMT-DATASET(3) WITH COMMON-VAR = 1, VAR-3 = 10
```

Semantics:

CREATE adds a record to the data set. For embedded data sets, a record of the master data set must be currently selected. A CREATE causes any selection on the data set, itself, and any of its embedded data sets to be discontinued. After a successful CREATE, the currently selected record for the data set is the created record.

<Data set name> explicitly names the data set for which a record is to be created. If not specified, the data set is determined from the items referenced in the <create list>.

<Record type> is a positive <integer> which must specify the record type for a data set with variable format records.

Each item named in the <create list> is assigned the appropriate value. All items assigned a value must belong to the same data set. Variables used in the <arithmetic expression> must reference values in currently selected records of other data sets. Items of the data set record which are not assigned values in the <create list> are initialized to the following values:

1. DASDL declared INITIALVALUE if present, or
2. DASDL declared NULL if present, or
3. Default NULL.

The following items cannot be specified in the <create list>:

- a. Count items
- b. Link items
- c. Population items
- d. Record type items
- e. Items specified as READONLY

Pragmatics:

CREATE invokes the data base access routines. Any violations of the normal DMSII restrictions related to the create/store function will cause an error condition, and an error message will be displayed. If an error condition is detected, the record is not created.

Creations made to an audited data base are audited in the normal DMSII manner.

DELETE STATEMENT

Syntax:

```
-- DELETE --<data set name>--|
---
```

Example:

```
SELECT C
#
DISPLAY ALL
  C1 = 95
  C2 = X
DEL C
#
```

Semantics:

DELETE removes the currently selected record from the data set. After a successful DELETE, the currently selected record for the data set remains the deleted record (e.g., items from this record may still be displayed).

If a record has not been selected for the data set, then an error message is displayed and no deletion is performed.

Pragmatics:

DELETE invokes the data base access routines. Any violations of the normal DMSII restrictions related to the delete function will cause an error condition, and an error message will be displayed. If an error condition is detected, deletion is not performed.

Deletions made to an audited data base are audited in the normal DMSII manner.

UPDATE STATEMENT

The syntax and semantics of the UPDATE statement have not changed. It should be noted, however, that variables used in an <arithmetic expression> in the <update list> always reference current values in the data set record being updated or reference values in currently selected records of other data sets.

Example:

```
UPDATE X1=X1+10, X2=X1
```

If the value of X1 is 10 in the currently selected record of data set X, the value of X1 is 20 and the value of X2 is 10 in the updated record.

SET STATEMENT ADDITION

Syntax:

The following option has been added to the SET statement:

```
-- CREATE ---<create specification>----|
      | - NONE -----|
```

Example:

```
SET O-P-DATA TO CREATE O-P-ORDNUM=O-ORDNUM,O-P-PARTNUM=123
```

Semantics:

SET CREATE modifies or eliminates the create text for the data set.

RECALL STATEMENT ADDITION

Syntax:

The following option has been added to the RECALL statement:

```
--<date set name>-- CREATE --|
--
```

Example:

```
RECALL X CREATE
```

Semantics:

If <data set name> CREATE is specified, INQUIRY loads and displays the <text> of any CREATE statement associated with the data set.

CHANGES TO BUILDINQ FOR RECORD CREATION AND DELETION

After a valid data base name is given, BUILDINQ now responds with the the message

```
ALLOW INQUIRY ONLY (YES OR NO)?
```

Enter YES if only inquiry is to be allowed; record update, creation and deletion is not allowed. BUILDINQ then continues by requesting "WHICH OPTION".

However, if NO is entered, BUILDINQ responds with the message

```
ALLOW UPDATE (YES OR NO)?
```

Enter YES if UPDATE is to be allowed for the data base; otherwise, enter NO.

BUILDINQ then responds with the message

```
ALLOW CREATE (YES OR NO)?
```

Enter YES if CREATE is to be allowed for the data base; otherwise, enter NO.

BUILDINQ then responds with the message

```
ALLOW DELETE (YES OR NO)?
```

Enter YES if DELETE is to be allowed for the data base; otherwise, enter NO.

BUILDINQ then continues as before by requesting "WHICH OPTION".

When INQUIRY is generated using cards, the following cards are now accepted.

1. CREATE Card

```
CREATE
```

If this card is present, records can be added to the data base using INQUIRY.

2. DELETE Card

```
DELETE
```

If this card is present, records can be deleted from the data base using INQUIRY.

EXTENSION OF DMINQ PROCEDURE FOR CREATE AND DELETE

The DMALGOL boolean procedure DMINQ [X] (A[*]) has been extended to provide for dynamic calls to access routine procedures CREATE and DELETE. For calls to these procedures, array A is defined as follows:

A [0] = identifies desired procedure

23 = create
24 = delete current

1. Create (A[0]=23)

A [1] = unused
A [2] = If variable format then record type, else unused
A [i], i>2 is unused

2. Delete current (A[0]=24)

A [i], i>0 is not utilized

D3245 INQ - ACCESSING RELATED RECORDS

The REPEAT statement, as currently implemented, can be used to access records from a disjoint data set which are related to another data set by symbolic key. Examples of this use of the REPEAT statement, which are not in the DMSII Inquiry Reference Manual, are shown below. For an explanation of the syntax and semantics of REPEAT, see DMSII Inquiry Reference Manual (Form No. 5001472), pages 5-24, 5-25, 5-46, and 5-47.

The following DASDL defines the data base used in the examples.

```
EMP DATA SET
(EMPNO      NUMBER(6);
 NAME       ALPHA(20);
 SALARY     NUMBER(8,2);
 EMP-JOBCODE NUMBER(2);
 EMP-DEPTNO NUMBER(4);
);
EMPNOSET SET OF EMP KEY IS EMPNO;
EMP-DEPTNOSET SET OF EMP KEY IS ( EMP-DEPTNOSET, EMPNO);

DEPT DATA SET
(DEPTNO     NUMBER(4);
 DEPTNAME   ALPHA(12);
 LOC        ALPHA(12);
);
DEPTNOSET SET OF DEPT KEY IS DEPTNO;

JOB DATA SET
(JOBCODE    NUMBER(2);
 TITLE      ALPHA(20);
);
```

Example 1

For each record of a disjoint data set, select a related record from another disjoint data set.
List the NAME, SALARY and DEPTNAME of all employees.

```
SET DEPT TO DISPLAY NAME, SALARY, DEPTNAME AT DEPTNO=EMP-DEPTNO
SELECT EMP, THEN REPEAT DEPT
```

These INQUIRY statements result in the selection of an EMP record, then, selection of a DEPT record according to the selection condition given in the DISPLAY statement. For the EMP and DEPT records selected, NAME, SALARY, and DEPTNAME are displayed. Selection of additional EMP and DEPT records requires NEXT EMP statements to be entered.

If a related DEPT record is not found for an EMP record, no items are displayed. However, since an EMP record is selected, a DISPLAY may be given to display any items from the EMP record.

To automatically select all EMP and DEPT records, the following statements can be given:

```
SET DEPT TO DISPLAY NAME, SALARY, DEPTNAME AT DEPTNO=EMP-DEPTNO
SET EMP TO SELECT EMP
SET EMP TO REPEAT DEPT
SET EMP TO LIMIT = NONE
SET DEPT TO LIMIT = NONE
REPEAT EMP
```

The REPEAT EMP statement initiates the selection of EMP records. For each EMP record selected, the DISPLAY command for DEPT is executed. The setting of LIMIT = NONE for EMP and DEPT is required to override any previously specified display limits, implicit or explicit.

If a related DEPT record is not found for an EMP record, no items are displayed.

To automatically select all EMP and DEPT records and display items from the EMP record even when a related DEPT record is not found, the following statements can be given:

```
SET DEPT TO DISPLAY DEPTNAME AT DEPTNO = EMP-DEPTNO
SET DEPT TO LIMIT = NONE
DISPLAY NAME, SALARY VIA EMP, THEN REPEAT DEPT
```

To generate a INQUIRY report listing the NAME, SALARY, and DEPTNAME for each employee, the following statements can be given:

```
SET DEPT TO SELECT AT DEPTNO=EMP-DEPTNO
SET EMP TO SELECT EMP
SET EMP TO REPEAT DEPT
REPORT NAME, SALARY, DEPTNAME
GENERATE REPORT VIA EMP
```

If a related DEPT record is not found for an EMP record, no items are reported.

Example 2

For each record of a disjoint data set, select multiple related records from another disjoint data set.

For each department, print the DEPTNAME and list the NAME and SALARY of all department employees.

```
SET EMP TO DISPLAY NAME, SALARY AT EMP-DEPTNO = DEPTNO
SET EMP TO LIMIT = NONE
DISPLAY DEPTNAME VIA DEPT, THEN REPEAT EMP
```

These INQUIRY statements result in the selection of all DEPT records. For each selected DEPT record, DEPTNAME is displayed, and the REPEAT causes selection of EMP records according to the selection condition given in the DISPLAY for EMP. For each selected EMP record, NAME and SALARY are displayed.

To generate a INQUIRY report containing the required information, the following statements can be given:

```
SET DEPT TO SELECT DEPT
SET EMP TO SELECT AT EMP-DEPTNO = DEPTNO
SET DEPT TO REPEAT EMP
REPORT DEPTNAME: NAME, SALARY
GENERATE REPORT VIA DEPT
```

If no related EMP records are found for a DEPT record, no information is reported for the corresponding department.

Example 3

Select related records which satisfy additional criteria.

For each department, print the DEPTNAME and list the NAME and SALARY of all department employees who earn greater than 20,000 dollars.

This query is identical to the query of Example 2 except that an additional criteria is imposed on the department employees. The following INQUIRY statements can be given:

```
SET EMP TO DISPLAY NAME, SALARY AT EMP-DEPTNO=DEPTNO AND SALARY>20000
SET EMP TO LIMIT = NONE
DISPLAY DEPTNAME VIA DEPT, THEN REPEAT EMP
```

Example 4

Select related records from more than two disjoint data sets.

For each department, print the DEPTNAME and list the NAME, SALARY, and job TITLE of all department employees.

This query is identical to the query of Example 2 except that an additional data set must be accessed in order to provide the employee job TITLE. The following INQUIRY statements can be given:

```
SET JOB TO DISPLAY TITLE AT JOBCODE = EMP-JOBCODE
SET JOB TO LIMIT = NONE
SET EMP TO REPEAT JOB
SET EMP TO DISPLAY NAME, SALARY AT EMP-DEPTNO = DEPTNO
SET EMP TO LIMIT = NONE
DISPLAY DEPTNAME VIA DEPT, THEN REPEAT EMP
```

D3264 INQ - IMPLICIT QUALIFICATION IMPROVEMENT

The following INQUIRY specifications affect the implicit qualification of subsequent duplicate names:

```

SET statement
CREATE statement
UPDATE statement
REPEAT <data set name>
<select specification>
DISPLAY ALL OF <data set name>
NEXT <data set name>
GENERATE of subset

```

Each of the constructs above specifies a data set to INQUIRY. Subsequent statements involving items in this data set need not be qualified. INQUIRY uses the specified data set until one of the constructs above is given for a different data set.

D3273 INQ - SETTING ITEMS TO "NULL," TESTING FOR "NULL"

Data items may now be set to NULL value when records are created or updated in INQUIRY. In addition, data items may be tested for NULL when selecting records.

The revised syntax of <create list> and <update list> is as follows:

```

|<-----, -----|
|<alpha item>-- = ---<string>-----|
|               | - NULL -----|
|<arithmetic item>-- = ---<arithmetic expression>--|
|               | - NULL -----|
|<boolean item>-- = --- TRUE -----|
|               | - FALSE -----|
|               |

```

Data items which are REQUIRED or which do not have NULL values (e.g., Boolean items) cannot be set to NULL.

The following syntax is now valid for a <boolean primary> in a <selection condition>:

```

----<alpha item name>----- EQL --- NULL --|
|<arithmetic item name>--| - NEQ -|
|                         | - = -|
|                         | - ≠ -|

```

Only data items which can have NULL values can be tested for NULL. If a data item is tested for NULL which cannot have NULL values, the following error message is given:

TEST FOR NULL IS SUPERFLUOUS--ITEM CAN NOT BE NULL

If a data item is undefined because the item does not appear in the variable format portion of a particular record or the subscript of the item is undefined, then the condition <item name> = NULL, or <item name> ≠ NULL, will evaluate to FALSE. (See page 4-22 of the DMSII INQUIRY Reference Manual (Form No. 5001472) for a discussion of undefined items.)

D3450 INQ - "SAVE" COMMAND

Page 5-45 of the DMSII INQUIRY Reference Manual, should be changed as follows:

Paragraph 4

FROM

"Non-privileged users can create and save system files, but cannot access them."

TO

"Only privileged users can create and save system files."

Paragraph 5

FROM

"Non-privileged users can create and save files under other usercodes, but cannot access them."

TO

"Only privileged users can create and save files under the usercodes."

D3489 INQ - GROUP KEYS IN LOGICAL DATA BASES

A restriction currently exists in INQUIRY concerning group keys in remapped sets of logical data bases. Elementary items within such keys are not recognized in selection expressions as key items, thus INQUIRY does not always choose the most optimal search strategy when these items are referenced. As a detour, either use the group item name when feasible in the selection expression or declare the set in DASDL using only elementary items.

D3615 INQ - SEGMENTED VALUE ARRAYS

Before segmented value arrays were implemented in ALGOL, data bases with a large total number of structures and items were handled differently from small data bases. For large data bases, the DMINQDIRECTORY created by BUILDINQ was used during INQUIRY initialization. Now, segmented value arrays are used for all data bases.

D3621 INQ - "LOCKTOMODIFYDETAILS"

If a data set has the DASDL option LOCKTOMODIFYDETAILS set, INQUIRY cannot be used to create, delete or modify any of its descendants.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - INQUIRY

P2759 INQ - OPTIMIZE "FIND VIA" <SUBSET> "AT" <CONDITION>

Previously, when a data set was accessed VIA a subset with multiple keys specified in the selection condition, each data set record in the subset would be read and the selection condition applied. Now, when feasible, the selection condition is used to search subset entries. Only data set records for qualifying subset entries are then read.

P3000 INQ - SYSTEM "ID" AND PATCH IN HEADING

The system type and software version number are now displayed by INQUIRY.

P3001 INQ - INVALID PAGE BREAK ON CONTROL BREAKS

If the PAGE option were specified in the REPORT statement for a higher level control item, a PAGE break would incorrectly be taken when a lower level control item changed value. In addition, when a higher level control item changed value but a lower level control item retained the same value, the lower level control item was not properly printed in the control break heading. Both problems have been corrected.

P3032 INQ - CHANGE TO MAXIMUM DISPLAY

Setting LIMITS to NONE or not specifying a limit in a DISPLAY statement would result in a default limit of 4095; thus, a NEXT command was required after 4095 records were displayed. This default upper limit has been removed. Specifying no limit now truly results in no limit.

P3161 INQ - TRUNCATION OF POSITION "132"

INQUIRY now uses character position 132 on the printer file.

P3186 INQ - REPORTING LONG SUBSCRIPTED ALPHA ITEMS

INQUIRY reported incorrect values of subscripted alpha items via the REPORT statement. This would occur whenever the declared length of the item was greater than the column width and the subscript for the item was not 1. This problem has been corrected.

P3188 INQ - USE OF "0" IN UNQUOTED STRING

An unquoted string containing a "0" character (48"F0") which started with one or more numeric characters followed by one or more alpha characters was incorrectly parsed. The problem has been corrected.

P3334 INQ - "SET DISPLAY" RESULTS IN IMPROPER LIMIT

Specifying a DISPLAY via the SET verb could sometimes result in an unexpected and improper display limit. For example, the following sequence would result in a display limit of 5:

```
SET D1 TO DISPLAY 5 VIA SET
SET D1 TO DISPLAY VIA SET
```

Specifying a DISPLAY via the SET statement now results in an implicit LIMIT specification of NONE or the explicit LIMIT given in the DISPLAY.

P3387 INQ - FUNCTIONS PERFORMED VIA EMBEDDED MANUAL SUBSET

INQUIRY failed to recompute a function that was performed VIA an embedded manual subset each time the owner of that subset changed. This problem has been corrected. FOR EXAMPLE: Assume a data base having data sets D1 and D2, and an embedded manual subset EM1 which is owned by D1 and spans D2.

```
DISPLAY <item of D1>,SUM(<item of D2>VIA EM1)VIA D1
```

INQUIRY will now compute the correct sum for each record of D1.

P3443 INQ - ENTERING INPUT BEFORE PREVIOUS OUTPUT FINISHES

On some occasions, if input is entered, other than a space or "NEXT", before the displayed output generated by a macro invocation was completed, an improper syntax error or INVALID INDEX would occur. This problem has been corrected.

P3444 INQ - "?AX" NOT RECOGNIZED ON LINEAR SEARCH OF SET

Previously, a linear search through a set could not be stopped by entering an "?AX". This problem has been corrected.

P3445 INQ - DISPLAY ITEM NAME

INQUIRY failed to print the item name when printing long alpha items under terminal format tab. This problem has been corrected.

P3446 INQ - "#NONE", "#NO MORE" TERMINATION

Previously, #NONE or #NO MORE was treated as an error condition and caused termination of a multiple-statement input or multiple-statement define. This has been corrected.

P3447 INQ - ROUNDING VIRTUAL ITEMS

Previously, commands that compared an INQUIRY virtual item to a literal value which had digits to the right of the decimal point rounded the literal value before doing the comparison. Consequently, records which met the selection condition were not selected. This is no longer the case.

P3490 INQ - RECALL OF "UPDATE" COMMAND

Previously, the RECALL command would change some literal values given in an UPDATE command. For example, assuming an item described as REAL (S5,3) in a data set D, if the following commands were issued:

```
SET D TO UPDATE I=.082
RECALL D UPDATE
```

INQUIRY would respond as follows:

```
UPDATE I:=.08.
```

Similarly, if the following commands were issued:

```
SET D TO UPDATE I=5.428
RECALL D UPDATE
```

INQUIRY would respond as follows:

```
UPDATE I:=5.542
```

This problem has been corrected.

P3555 INQ - INCREASE NUMBER OF DATA SET FUNCTIONS ALLOWED

Previously, the maximum number of data set functions (i.e., AVG, SUM, COUNT, etc.) over a given set data set that a user could perform the following actions simultaneously was 15:

- 1) have as virtual items, and
- 2) use in a command.

This limit has been raised to 48.

P3655 INQ - SORTING ON SUBSCRIPTED ITEMS

Records were not properly sorted when a sort key was a non-occurring item in an occurring group and the item was not the first item within the group. In addition, a syntax error resulted when a multiple-subscripted item was given as a sort key. These problems no longer occur.

P3701 INQ - SEARCH ON INDEX RANDOM SET

INQUIRY sometimes performed a KD search on an index random set when it was possible to do a more efficient KI search on a different set. For example, given the DASDL:

```
D DATASET (D1 NUMBER (2),);
S1 SET OF D KEY IS D1 INDEX RANDOM;
S2 SET OF D KEY IS D1 INDEX SEQUENTIAL;
```

and the command:

```
DISPLAY D1 AT D1 > 10
```

INQUIRY would execute the command by doing an KD search on the set S1. Now, INQUIRY will do a KI search on S2.

P3745 INQ - VIRTUAL ITEMS EVALUATED

Previously, INQUIRY would occasionally lose a little accuracy in computing the value of some virtual items. For instance, given two data items A and B, each having the description NUMBER (S9,2), and a virtual V=A-B:

If A and B each had the value 142.02 in a particular record, V would be computed to be 4.47034835815X10**10, instead of 0.

Not only did this problem cause the wrong value to be reported for V, but it also caused some records to be skipped when INQUIRY was processing a command which used V in a selection expression (like V=0). This problem has been corrected.

P3747 INQ - IMPROVED SEARCHING CAPABILITIES

INQUIRY is now able to better determine the best set to use in searching for the data records that satisfy the selection expression.

P3791 INQ - "DISPLAY ALL" CORRECTION

The DISPLAY ALL command failed to properly display occurring field items. It would only display the first occurrence of a field, and some of the values for the Booleans within the field were occasionally reported wrong. This problem has been corrected.

P3792 INQ - DO NOT CLOSE DATA BASE IF "TASKVALUE=1"

When INQUIRY is run with a TASKVALUE of one, the data base is not opened. Previously, the data base was closed unconditionally; this resulted in a DMOPENERROR. This problem has been corrected.

P3797 INQ - REPORT CONTROL ITEMS

If an item in an occurring group were used as a control item in a report, INQUIRY did not perform control breaks properly. In some cases, INQUIRY would perform a control break even though the value of the control item had not changed. In other cases, INQUIRY would fail to perform a control break when the value of the control item actually did change. This problem has been corrected.

P3823 INQ - "DISPLAY ALL" OF GLOBAL DATA

INQUIRY was dying with a "STACK OVERFLOW" when the following commands were issued:

```
SELECT <global data name>  
DISPLAY ALL
```

This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - INTERFACE

D3229 INTERFACE - DATA BASES AT DIFFERENT RELEASE LEVELS

INTERFACE has been enhanced to accept data bases at different release levels. Previously, all data bases invoked by a single program were required to be at the same release level. This restriction has been eliminated.

On the Mark 31 release, a program may invoke both Mark 31 and Mark 30 data bases. On the Mark 32 release, a program may invoke any combination of Mark 32, Mark 31 and Mark 30 data bases.

Because the description file format was changed extensively on the Mark 30 release, INTERFACE cannot read description files created on Mark 29 or earlier releases. Programs compiled with the new INTERFACE may invoke only Mark 30 or later data bases.

D3305 INTERFACE - ANOMALIES OF LOGICAL INVOCATIONS

Certain logical data base invocations may produce unexpected results. When a link or subset is invoked via a logical data base declaration, DATABASE/INTERFACE attempts to fix-up the link or subset to reference an appropriate structure which was declared in the logical data base. If the data set referenced by the link or subset is found, then no fixing up is needed. If however the data set is not included, then the reference and keys of the link or subset will be fixed-up to point at a remap of the data set which was declared in the logical data base. If two or more remaps of a single data set are included in the logical data base declaration, then the one listed last is chosen. Since this fix-up procedure is based strictly on the logical data base declaration and not on what was actually invoked, it is possible for DATABASE/INTERFACE to fix-up a link or subset to reference an uninvoked structure. Thus, unexpected results may occur when two or more possible objects of a link or subset are declared in a logical data base but not all of them are invoked. Rather than partially invoking a logical data base, a safer method is to create partial views which declare only what you intend to invoke.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - INTERFACE

P3103 INTERFACE - SETS NOT INVOKED IN LOGICAL "DB" PROPERLY

Sets and subsets of remapped data sets which were invoked in a logical data base were not invoked properly. INTERFACE would incorrectly point an invoked set at the invoked remap even if the original data set were invoked. INTERFACE will now point an invoked set at an invoked remap of the original data set only if the data set itself is not invoked.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - LOADDUMP

D3614 LOADDUMP - ELIMINATE "OPEN INITIALIZE"

The OPEN INITIALIZE command has been de-implemented. ARCHIVEUPDATER can no longer OPEN INITIALIZE a data base. LOADDUMP now opens the data base UPDATE instead of INITIALIZE.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - LOADDUMP

P3712 LOADDUMP - "TITLE" NOT PARSED CORRECTLY

Previously, LOADDUMP was not parsing the optional TITLE part of a file specification correctly. This problem has been corrected.

P3713 LOADDUMP - PREVENT "SEG ARRAY" ERROR

Previously, LOADDUMP could cause a SEG ARRAY error while trying to compare the values of two strings. This problem has been corrected.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - PRINTAUDIT

P2760 PRINTAUDIT - USE AUDIT RECORD INFORMATION TABLE

PRINTAUDIT now uses the audit record information table from PROPERTIES to locate various control information in audit records.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - PROPERTIES

D3113 PROPERTIES - DELETE "READAHEADB"

Since the READAHEAD option in DASDL has been deimplemented, this property has been removed.

D3116 PROPERTIES - PUT SUBSYSTEM "ID" IN TEXT

The subsystem identifier given to DASDL via the SUBSYSTEM dollar card option will now be stored in the text of the description file. This will allow ACCESSROUTINES to zip COPYAUDIT under the proper subsystem.

D3118 PROPERTIES - REMOVE PROPERTIES FOR "27" LINKS

The following properties, which were used with 27 links, have been removed:

ORIGOCCURSF
ORIGTOTALSZ
ORIGOFFSET
ORIGOCCURSMIN
ORIGOCCURSMAX
ORIGLINKTYPE

D3162 PROPERTIES - "DASDL" DEFAULTS

The physical attributes computed by DASDL are now used by all tailored software, thus completing the work begun on the Mark 31 DMSII release to consolidate and improve physical attribute calculations.

D3272 PROPERTIES - RESTRUCTURE DESCRIPTION FILE PROPERTIES

The PROPERTIES listing has been alphabetized to facilitate finding individual properties in the listing. In addition, each property has been given a usage list which documents where the property is used and allows the creation of a directory by the property-making program. This directory (located in the symbolic at 16000000-19999999) lists all the properties for the various description file structures in storage order. This directory should be consulted before adding a new property. Instructions for adding new properties are given at 21000000 in the PROPERTIES symbolic.

D3274 PROPERTIES - STRUCTURE CALCULATIONS

PROPERTIES now contains a procedure and associated declarations to perform structure attribute calculations for software that directly interprets data base Description files.

D3317 PROPERTIES - CRUNCH "PROPERTIES" SYMBOLIC

The new symbolic file produced when DATABASE/PROPERTIES is run will now be crunched, have a blocksize of 420, and an areastyle of 1008 records.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - PROPERTIES

P3556 PROPERTIES - INITIALIZATION OF DISJOINT UNORDERED DATA SET

A disjoint unordered data set with blocksize of one record was improperly initialized by UTILITY on the Mark 31 system software release. This would cause the ACCESSROUTINES to infinitely loop when attempting to sequentially access the initialized structure. A reorganization of the initialized structure would result in a SEG ARRAY error. This problem has been corrected.

P3814 PROPERTIES - INCORRECT OUTPUT REPORT

In some cases, the report produced by the UTILITY INITIALIZE procedure contained a double ON <packname> part. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - PARTITION CONTROL

D3466 PTNCTL - "28" TO "29" CONVERSION OPTIONS REMOVED

The PARTITIONCONTROL functions associated with conversion from Mark 28 to Mark 29 data bases have been removed. The program's only remaining function is to extract information from the PARTITIONINFO data set and return it to SYSTEM/DMCONTROL. DMCONTROL uses this information to recover the partition directory of the data base control file in response to a "RECOVER PARTITIONS" request. PARTITIONCONTROL no longer requires user interaction in any form.

DOCUMENT CHANGES NOTES (D NOTES)

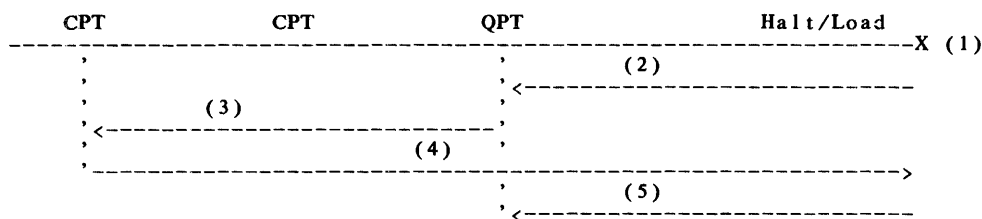
DMS II - RECOVERY

D3050 RECOVERY - HALT/LOAD RECOVERY SEQUENCING

Halt/Load Recovery Sequencing

The sequencing of image application in Halt/Load recovery has been changed. Formerly, before images were applied from the end of the audit to the last quiet point (i.e., an audit record indicating the null transaction state; e.g., a syncpoint). After images were then applied from the second control point (or its equivalent) preceding the end of the audit to the last quiet point. In the present implementation, after images are applied from the second control point through the end of the audit; then before images are applied from the end of the audit to the last quiet point. This new implementation more accurately models the action of the ACCESSROUTINES in creating the audit and ensures that, at the start of before image application, the data base will be logically in the same state as it was when the Halt/Load occurred.

The following is a graphic representation of the new implementation of Halt/Load recovery with respect to the audit (CPT=control point, QPT=quiet point):



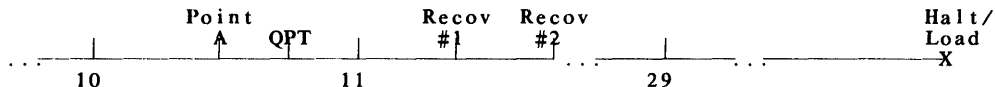
- (1) Find the end of the audit.
- (2) Find the last quiet point (not applying images).
- (3) Go back 2 control points (not applying images).
- (4) Apply after images to the end of the audit.
- (5) Apply before images to the last quiet point.

REBUILD

Formerly, REBUILD would abort while attempting to rebuild to the end of a disk audit of an unrecovered data base. It will now rebuild the data base successfully.

REBUILD and RECONSTRUCT

REBUILD and RECONSTRUCT will now create restart points at all recovery #1 records they encounter (after applying before images to the last preceding quiet point). This will alleviate the problem represented by the following example (numbers represent TSNs):



Assume that a Halt/Load occurred during a REBUILD or RECONSTRUCT, that at the time of the Halt/Load the last restart point was Point A (and not the Recov #1 record) and, further, that 29 was the TSN on disk for the given table. After restarting, the REBUILD or RECONSTRUCT would abort with an invalid TSN while attempting to apply before images to 11. The invalid TSN occurred because before image application requires that the TSN in the audit be greater than or equal to the TSN currently on disk. If the restart point is the Recov #1 record (which the new implementation ensures), the problem is avoided since before images will not be applied until the audit TSN is greater than or equal to the TSN in the data base.

D3051 RECOVERY - SWITCH BACK TO PRIMARY AUDIT

If RECOVERY uses the secondary audit for error recovery, RECOVERY will now revert to the primary when it is finished using the current audit file.

D3289 RECOVERY - ALLOW NORMAL "REBUILD/ROLLBACK"

Formerly, REBUILD and ROLLBACK would terminate abnormally if either:

- A. An audit IO error were encountered, or
- B. An audit file which did not exist or was unusable were required by the REBUILD or ROLLBACK (e.g., "REBUILD THRU AUDIT 2300" where the last audit file available is 2299).

In Case A, RECOVERY would terminate with an error message for the IO error; in Case B, RECOVERY would continue asking for the audit file until DSed.

Now, in both of the above cases, the following options are available:

1. RECOVERY may terminate normally at the point where the IO error occurred (case A), or
2. RECOVERY may terminate normally at the point where the request for the unavailable file was made (Case B).

In Case A, the message "AUDIT IO ERROR: TERMINATE TO FINISH NORMALLY - OTHERWISE DS" is displayed. A reply of "TERMINATE" will cause normal termination. Any other reply will cause abnormal termination. In Case B, along with the usual "RETRY . . ." message, the message "TERMINATE TO FINISH NORMALLY" is displayed. A reply of "TERMINATE" will cause normal termination.

D3464 RECOVERY - QUICKFIX FAILS ON INCONSISTENT PARTITIONS

Previously, for partitioned data bases, QUICKFIX would occasionally fail with "PARTITION NAME TABLE INCONSISTENT WITH AUDIT" because of faulty PARTITION NAME TABLE initialization prior to the AFTER IMAGE application phase. This problem has been corrected.

This correction, however, forces the QUICKFIX PREPASS phase to locate a BCP, DBS1, DBST, RECOV, FILEDC or STRDC audit record; thus, the PREPASS phase could possibly exceed the LIMIT specification indicated to UTILITY.

D3465 RECOVERY - AUDIT DISCONTINUITY

A DASDL update which alters the number of OPENPARTITIONS for a structure, effectively creates a discontinuity in the audit. That is, the RECOVERY compiled following the update cannot use audit created prior to the update and similarly, the RECOVERY compiled prior to the update cannot use audit created after the update. Following the update, therefore, the ENTIRE data base must be dumped using UTILITY.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - RECOVERY

P2789 RECOVERY - ENSURE "REBUILD" RESTART

Previously, it was possible for REBUILD to write information to the REBUILDINFO file and later be DSed or Halt/Loaded without having created a restart point. Under these circumstances, REBUILD would abort on rerun. This problem has been corrected.

P2879 RECOVERY - "RLA" FOR ABORT, HALT/LOAD ONLY

Formerly, it was possible for REBUILD and ROLLBACK to write to the Row Lock-out Audit. In General, this is useless since the locked-out row(s)/structure(s) are not Quickfixable. Now, only Abort and Halt/Load recovery will write to the Row Lock-out Audit.

P2887 RECOVERY - DO NOT PRINT AUDIT BLOCK IF "EOF" ENCOUNTERED

The diagnostic information printed when an error occurred in RECOVERY would fail if end-of-file had just been encountered on the audit. The problem has been corrected.

P3036 RECOVERY - TEST GENERATES TOO MUCH CODE

Previously, a data base with a large number of structures could cause the compilation of RECOVERY to abort with a "PROGRAM SEGMENT TOO LARGE" error in procedure TEST in NEXTPARAMETERREC. This problem has been corrected.

P3109 RECOVERY - UPDATE DISK HEADER

Previously, RECOVERY attempted to update disk headers for a file it had not referenced. This can no longer occur.

P3144 RECOVERY - READ WRONG AUDIT BLOCK

Formerly, the physical and logical positioning for tape audits could get out of sync with one another. This would cause the wrong audit block to be read, resulting in "unexpected ABSN", "unexpected timestamp", etc. This no longer occurs.

P3491 RECOVERY - ROW OF ORDERED DATA SET LOCKED OUT

Previously, RECOVERY could erroneously lock out a row of an ordered data set when attempting to apply a TABLE AFTER IMAGE audit record which referred to a newly-allocated block. This problem has been corrected.

P3567 RECOVERY - ROLLBACK FAILS ON "FILEDC/STRDC"

Previously, under certain circumstances, ROLLBACK would fail with an "UNEXPECTED AUDIT RECORD TYPE" on a FILEDC/STRDC audit record in APPLYBEFOREIMAGES. This problem has been corrected.

P3667 RECOVERY - CORRUPT AUDIT STOPPER

If RECOVERY were DSed when it was fixing up the last audit block (after it had written the audit block and stopper but before it had set auditfile last record), when Halt/Load recovery was run, it would fault with "UNEXPECTED AUDIT BLOCK SERIAL NUMBER". This problem has been corrected.

P3668 RECOVERY - INVALID UNIT NUMBER

When RECOVERY gets an error reading the audit file, it displays the message "ERROR READING AUDIT FILE ON UNIT <n>". The value being displayed for audits to disk was not a valid unit number. Since a unit number is not valid for a disk file, RECOVERY will now display the familyname if the audit is to disk and the unit number if the audit is to tape.

P3669 RECOVERY - OPTION "USE DUP"

If RECOVERY could not find the desired audit file, it would display the message "RETRY OR USEDUP OR FAMILY=<familyname>". USEDUP was not accepted as a valid response. This problem has been corrected.

P3757 RECOVERY - "SEG ARRAY" ERROR

When RECOVERY cannot find an audit file, it prompts the operator to enter one of several options. If the operator entered "AX FAMILY=<family name>", where <family name> was longer than allowed, a SEG ARRAY error occurred. This problem has been corrected.

P3820 RECOVERY - SPLIT INDEX RANDOM TABLES

In Afterimages, a SPIRT may cause the new table to get the wrong number of entries. If an AIRE for the new table is encountered by RECOVERY after the SPIRT, it will terminate abnormally with a "VALIDITY CHECK FAILED" message.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - REORGANIZATION

D3073 REORG - "REORGANIZATION" ACCELERATION

The speed of REORGANIZATION has been significantly increased by optimizing the serial I/O performed on DMSII structures requiring generation or fixup. Buffers used in REORGANIZATION are resized to facilitate the reading and writing of multiple blocks in one physical I/O; thus, a significant decrease in I/O time is achieved at the expense of some increase in core utilization. Also, better utilization of space on intermediate tape files is achieved by packaging normal blocks into "large" tape blocks.

D3120 REORG - "DASDL/REORGANIZATION" ENHANCEMENTS

Introduction

Several new features have been added to the existing DASDL/REORGANIZATION package which will allow the DMSII user to more conveniently change the data base description. The user will now be able to :

- 1) increase or decrease the size of a data item
- 2) change the type of an item
- 3) change the sign of an item
- 4) change the occurs clause for an item
- 5) add, delete, or reorder booleans in a field item
- 6) change the key, key data, ascending/descending, and duplicates clause for a set
- 7) change the key, key data, ascending/descending, duplicates clause, and where clause for an automatic subset
- 8) change a remap description on a DASDL update

Item Type and Size Changes

The table below shows all the valid item type conversions and the affects of each on the original data. This table also shows the effect of an item size change with no type change. Where possible, COBOL74 move rules were modeled in performing the transformations. For moves not covered by COBOL74, the transformation attempts to preserve as much of the original data as possible.

reorganization move		truncation or			translate
from	to	space fill on right	zero fill on left	zero fill on right	
alpha	alpha	x			
alpha	group	x			
alpha	number		x		x
alpha	real		x		x
alpha	field		x		x
alpha	boolean		x		x
group	alpha	x			
group	group	x			
group	number			x	x
group	real			x	x
group	field			x	x
group	boolean			x	x
number	alpha	x			x
number	group	x			x
number	number		x		
number	real		x		
number	field		x		
number	boolean		x		
real	alpha	x			x
real	group	x			x
real	number		x		
real	real		x		
real	field		x		
real	boolean		x		
field	alpha	x			x
field	group	x			x
field	number		x		
field	real		x		
field	field		x		
field	boolean		x		
boolean	alpha	x			x
boolean	group	x			x
boolean	number		x		
boolean	real		x		
boolean	field		x		
boolean	boolean		x		

Sign Changes

A sign may be added to or deleted from a REAL or NUMBER item. When the sign is added, a positive sign will be generated as the item is moved from the old data set record to the new data set record. If the sign is deleted, the sign will be deleted from the item as it is moved.

Changes to Occurs Clause

Occurrences of an item or group may be increased or decreased. An occurs clause may be added or deleted. Data will be lost when occurrences are decreased. When occurrences are increased, items or groups in the new data set record with no source in the old data set record will be filled with the appropriate initial value or null value using the CLEARDATA text.

Field of Boolean Changes

Booleans may be added to, deleted from, and reordered in an existing field item. This is done by simply specifying the field with the desired modifications in the DASDL reorganization run. Existing booleans in the field which are included in the new specification for the field retain their information across the reorganization.

Changes to Sets and Automatic Subsets

Key, key data, ascending/descending, and duplicates may now be changed for a set. Key, key data, ascending/descending, duplicates, and the where clause for an automatic subset may also be changed. Changes involving items may be explicit (changes to items as used by an index set) or implicit (changes to items as they are described in the data set). All of these changes to indexes require that the index be generated from the data set during reorganization except where no duplicates to duplicates first or last is the only change. The DASDL compiler will notify BUILDREORG when a set or subset must be regenerated from the data set and BUILDREORG

will create a proper default generate. To notify DASDL that you intend to make changes to an index set (whether explicit or implicit), the following new syntax has been added:

```
-- REORGANIZE -----|
      | - ( -- KEY --- CHANGED --- ) - |
      | - SAME ---- |
```

The existing "REORGANIZE" clause without the parenthetic clause is still valid and implies "REORGANIZE (KEY SAME)". To notify DASDL that the key, key data, ascending/descending, duplicates (except NO DUPLICATES to DUPLICATES FIRST or LAST), or the where clause should be changed, "REORGANIZE (KEY CHANGED)" should be specified.

Changes to Remap Descriptions

Remap views of the data set may be redefined on a DASDL update run. DASDL will detect the changes to the remap and invalidate old programs using that remap. See paragraph on UPDATE LEVEL MECHANISM.

Changes to Compact, Verify Clause, and Select Clause Items

Item changes may also be made to compact items, compact controlling items, and items used in select and verify clauses. The user should be cognizant of the fact that decreasing the size of compact controlling items may cause data loss in the controlled field during reorganization.

Translation

```
alpha or group to
      number
      real
      field
      boolean
```

The ALGOL INTEGER function is used to convert EBCDIC characters to integers. These integers are then handled as number to number conversions. (See TRUNCATION for EBCDIC representations of numbers.) The integer representation is returned (e.g., "1234" becomes 1234). When an item is converted to a Boolean item, the new item's value is determined by the contents of the old item. If the old item is an alpha, the Boolean value is obtained from the least significant bit of the rightmost character. If the old item is a group, the Boolean value is obtained from the least significant bit of the leftmost character. Special EBCDIC characters are handled in the following way :

- 1) high order four bits are disregarded
- 2) if binary value of the low order four bits is less than or equal to 9, then that value is returned
- 3) if the binary value is greater than 9, then the value is set to 8 or 9 by turning off bits 1 and 2
(ex. a "\$" which is a hex "5B" becomes a 9 and
a "=" which is a hex "7E" becomes an 8)

```
number
real
field
boolean
      to alpha or group
```

An ALGOL statement of the form "REPLACE...FOR n DIGITS" will be generated to perform the needed translation. Number and Real items with fractional parts will be integerized via truncation and then transformed. Signed items will lose their sign during transformation.

Truncation

If scalefactor decreases, the least significant digits of the fractional part are truncated. If precision decreases, the most significant digits of the number are truncated.

Restrictions

1. Key, key data, ascending/descending, or duplicates clause for a manual subset cannot be changed.

2. Key, ascending/descending, or duplicates clause for an access cannot be changed.
3. Key, key data, ascending/descending, duplicates or the where clause for an embedded set or subset cannot be changed.
4. Items used by self-correcting, symbolic, or verified links between data sets cannot be changed. +PCNALL (10-29-80)

Effect of Reorganization on User Programs

Some changes to data sets, sets, subsets and remaps will invalidate existing programs. These programs must be recompiled following the dasdl UPDATE. Old programs which are not recompiled will receive Version Error #2 exceptions at run time if they attempt to access modified structures.

Programs must be recompiled if they invoke structures changed in the following ways:

1. Data sets updated using the REORGANIZE(ITEMS CHANGED) option.
2. Sets or subsets updated using the REORGANIZE(KEY CHANGED) option.
3. Remaps which include new items, delete existing items or which include data items which have been changed in the data set. Programs which invoke remaps must be recompiled only if the remap changes; changes to the data set that do not affect the remap will not affect the program.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - REORGANIZATION

P2880 REORG - ELIMINATE ATTRIBUTE ERROR "FAMILYNAME"

A FILE ATTRIBUTE ERROR ON FAMILYNAME occurred when COPY was to FINAL medium; this error did not, however, affect REORG operation. This error no longer occurs.

P2888 REORG - INVALID BLOCK ON STANDARD DATA SET

Occasionally, REORGANIZATION would cause an empty block to be created in standard non-variable format data sets. This empty block would cause the following:

1. A CHECKSUM error
2. an entry missing from auto set/subset (delete)

The affected data set can be fixed by using REORGANIZATION with a "ORDER BY" set.

P2954 REORG - BAD "BCW" FOR ORDERED DATA SETS WITH SUBBLOCKS

When generating an ordered data set having subblocks, REORGANIZATION would occasionally generate a block with a bad BCW. Also, a subblock could have been created having a size which was not a multiple of the DASDL-specified SUBBLOCKSIZE. Both problems have been corrected.

P3093 REORG - REORGANIZATION OF EMBEDDED ORDERED DATA SET

The reorganization of an embedded ordered data set could subsequently cause a FATAL ERROR in the ACCESSROUTINES. The error would occur when records were deleted from the data set and blocks were returned to the available space chain. To eliminate this problem, the data set must be reorganized with a REORGANIZATION program which includes this change.

P3145 REORG - CORRUPTION OF COMPACT DATA SET

Previously, a compact data set could become corrupted after REORGANIZATION had been run on the data set. This problem was due in part to REORGANIZATION setting up a table block incorrectly for compact data sets. In addition, the ACCESSROUTINES was not making the proper checks on table blocks and eventually corrupted the table blocks. These problems have been corrected.

P3190 REORG - SPECIFYING "COPY"

If a "COPY TO <medium>" or "COPY TO <medium>, COPY BACK AT END" were given for a structure requiring only fixup, a warning message was issued by BUILDREORG and a "no file" condition subsequently resulted in the REORGANIZATION program. The COPY will now be ignored and not cause the "no file" problem in the REORGANIZATION program.

P3191 REORG - REORGANIZATION OF STANDARD DATA SET

If a standard variable format data set was reorganized having a blocksize within one or two words of a segment boundary and ADDRESSCHECK and/or CHECKSUM were set for the structure, an empty structure could result from the reorganization. This problem has been corrected.

P3318 REORG - "INVALID INDEX" BY ZERO LENGTH READ

Reorganization no longer does zero length reads. This could cause an invalid index in the MCP I/O routines.

P3397 REORG - "IXSEQ" WITH MULTI-COARSE TABLE LEVELS

REORGANIZATION did not correctly layout an index sequential structure which required multiple levels of coarse tables (e.g., the number of entries in the structure was greater than the square of the LOADFACTOR TIMES <the maximum number of entries per table>). The resultant reorganized structure would cause fatal errors in the ACCESSROUTINES. This problem has been corrected.

P3401 REORG - "CFUPDATEVERSION" ON PASS "1" OF "FIXUP"

A CFUPDATEVERSION was done when the structure was opened in Pass 1 of the FIXUP algorithm. This has been changed to a CFCHECKVERSION since Pass 1 does not change the structure but only reads it.

P3484 REORG - "CFAUDINZ" ONLY VALID FOR AUDITED DATA BASES

After reorganization of an unaudited data base, CFFILESTATEF for reorganized structures was set to CFAUDINZ in the control file and was never reset by ACR to CFFILENORMAL. Although this did not cause any subsequent problems, CFFILESTATEF is now set to CFFILENORMAL for reorganized structures in an unaudited data base. Code in UTILITY has been eliminated which tested for CFAUDINZ when the data base is unaudited. Since this code was totally irrelevant to unaudited data bases, it caused no problems.

P3486 REORG - INVALID DIRECT DATA SET

It was possible in a direct data set that an invalid record was improperly recognized as valid by REORGANIZATION or UTILITY. This would cause a fatal error in REORGANIZATION and an invalid record to be printed as valid in UTILITY. These problems have been corrected.

P3492 REORG - ADDING CHECKSUM TO COMPACT DATA SET

When running REORGANIZATION to add a checksum to a compact data set, the REORGANIZATION program could fail with a CHECKSUM I/O error when reading the old structure. This problem has been corrected.

P3670 REORG - RECORDS WITH UNDEFINED RECORD TYPE

Previously, REORG failed with a DATA ERROR #2 when it encountered a STANDARD VARIABLE FORMAT data set record with an undefined (i.e., all Fs) value for the record type. This occurred even though the record was actually a deleted record. These records are now properly ignored by REORG.

P3671 REORG - IMPROPER BIT VECTOR GENERATION

Two situations caused BIT VECTORS to be improperly generated. The first situation occurred when TABLESIZE for the BIT VECTOR was not a multiple of 48. REORGANIZATION would then not use all bits in the last word of a table. This was incompatible with other data base software which assume that all bits in a table are used to represent records in the data set. The second situation occurred when a BIT VECTOR was generated from the data set. If the data set contained any available records within its data blocks and the data set was not also generated, the BIT VECTOR would not contain bits for these records. Both situations resulted in corruption of the BIT VECTOR; both problems have been corrected.

P3672 REORG - DISK RESIDENT STRUCTURE

Under certain circumstances, REORGANIZATION would fail with an ADDRESSCHECK error when attempting to reorganize a structure resident on DISK, even though ADDRESSCHECK was not set for the structure. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - UTILITY

D3052 UTIL - CLEAR "TPS" INFORMATION

When UTILITY is used to initialize all data base structures, Transaction Processing System information in the data base control file is also initialized. If only selected structures are initialized, the TPS information will remain unchanged.

D3431 UTIL - EXPLANATION OF <RECOVER SOURCE>

Add the following to the DMSII Utilities & Operations Guide (Form No. 5001803), Page 4-37, following the last paragraph:

"UTILITY expects all files named in a <recover list> to reside on the corresponding <recover source>. If more than one tape is named in <recover source>, all files must reside on each tape."

D3451 UTIL - LABEL EQUATION OF DATA BASE DESCRIPTION FILE

Label equation of the DMALGOL compiler file DASDL when compiling UTILITY must specify the title of the data base description file as DESCRIPTION/<data base name>. Any other file title for the compiler file DASDL may cause the INTERFACE to wait with a

NO FILE DESCRIPTION/<data base name>

message or may cause an inappropriate description file to be used. This situation arises because UTILITY both invokes the data base and requires the description file be supplied to the compiler. The compiler file DASDL may be equated, but the INTERFACE's DASDL file cannot be equated. The INTERFACE always uses DESCRIPTION/<data base name> [where <data base name> is derived from the data base declaration in the host language program] as the title of the description file even if the compiler's DASDL file has been equated to some other title. The purpose of this note is solely to document this situation. Unfortunately, the problem cannot be circumvented without preventing the compilation of the INQUIRY program.

D3488 UTIL - ROW SELECTION CRITERIA

Row selection criteria for the <copy selector>, <row selector>, and <dump selector> options may be specified only once. A syntax error is now given for multiple specifications of the same criterion. The DMSII Utilities and Operations Guide (Form No. 5001803) has been altered as follows:

On Page 4-13, the syntax for the <copy statement> now reads as follows:

<copy selector>

```

      |<----- AND -----|
      |<- & ---|
-- ( ----- ) --|
      |-/1\- FAMILYINDEX -- = --<range>-----|
      |-/1\- ROW -- = --<range>-----|
      |-/1\- PACKNAME -- = --<family name>-|

```

On Page 4-17, the syntax for the <dbdirectory statement> now reads as follows:

<row selector>

```

      |<----- AND -----|
      |<- & ---|
-- ( ----- ) --|
      |-/1\- FAMILYINDEX -- = --<range>-----|
      |-/1\- ROW -- = --<range>-----|
      |-/1\- PACKNAME -- = --<family name>-----|
      |<----- , -----|
      |-/1\- ROWLOCK -- = -----/1\- LOCKEDROW -----|
      |<----- /1\- READERERROR -|

```

On Page 4-19, the syntax for the <dump statement> now reads as follows:

```

<dump selector>
      |<----- AND -----|
      |<- & ---|
-- ( ----- ) --|
      |-/1\- FAMILYINDEX -- = --<range>-----|
      |-/1\- ROW -- = --<range>-----|
      |-/1\- PACKNAME -- = --<family name>--|

```

on Page 4-30, the syntax for the <recover statement> now reads as follows:

```

<row selector>
      |<----- AND -----|
      |<- & ---|
-- ( ----- ) --|
      |-/1\- FAMILYINDEX -- = --<range>-----|
      |-/1\- ROW -- = --<range>-----|
      |-/1\- PACKNAME -- = --<family name>-----|
      |<----- , -----|
      |-/1\- ROWLOCK -- = -----/1\- LOCKEDROW -----|
      |<-----/1\- READERROR -|

```

D3571 UTIL - TAPE "SERIALNO" SPECIFICATION

UTILITY now accepts unquoted strings of six characters or less in dump tape serial number specifications. In addition, either double (") or single (') quotes may be used to delimit tape serial numbers. As before, tape serial numbers may contain only <letter>s and <digit>s.

The <string6> specification on page 4-11 of the DMSII Utilities and Operations Guide (Form No. 5001803) should be modified as follows to reflect these changes:

<string6>

Syntax

```

      |<-----|
-----/6\-----<letter>-----|
      | - " - |      | -- <digit> --- |      | - " - |
      | - ' - |      |                  |      | - ' - |

```

Semantics

<string6> is a group of from 1 to 6 <letter>s and <digits>s, optionally enclosed within quotes.

<string6> is used in UTILITY to designate dump tape serial numbers.

When UTILITY is initiated via CANDE, double quotes may not be used to delimit tape serial numbers.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - UTILITY

P2955 UTIL - VERSION TIMESTAMP MISMATCH

During COPY, the DMTIMESTAMP in the file header was not properly set if the structure name started with the letters 'CONTROL'. When the data base was accessed after the copy, a version mismatch occurred for these structures. This problem has been corrected.

P2956 UTIL - PRINT TAPE LABELS

UTILITY and COPYAUDIT will now print tape labels via the ADM EVENT PRINTLABEL mechanism. Previously, tape labels were not produced because these programs reverse verified their tapes. Now, tape labels are printed as soon as the tapes are opened (unlike normal tapes which have their labels printed when closed).

P2999 UTIL - FAILURE TO RELOAD BEYOND TWO DUMPS

The following command would fail to load any data from DUMP3: COPY/RECOVER FROM DUMP1, DUMP2, DUMP3. This problem has been corrected.

P3146 UTIL - OUTPUT HEADER

UTILITY now correctly displays the time in the header.

P3192 UTIL - PRINTING CONTROL INFORMATION

When printing the control information for a compact data set, an INTEGER OVERFLOW will no longer occur.

P3193 UTIL - LOWER CASE IN PARAMETER STRING

UTILITY now accepts lower case alphabetic characters in its input parameter string.

P3243 UTIL - CORRUPTED DUMPTIME TIMESTAMP

The timestamp passed from UTILITY to RECOVERY that tells RECOVERY where to start was corrupted. Bit 47:1 was being turned on in the following case:

1. Recover was being done from more than one dump.
2. A dump other than the first one had a dumptime greater than the earliest dumptime.

The correct timestamp is now passed.

P3244 UTIL - ERROR NOT GIVEN FOR INVALID SYNTAX

The following UTILITY command is syntactically invalid:

OFFLINE DUMP <data base>/A/= TO DUMPA, <data base>/B/= TO DUMPB

UTILITY did not generate a error; instead, UTILITY dumped only <data base>/A/=. UTILITY now generates a syntax error.

P3261 UTIL - NO CHECKSUM ON BLOCK ZERO

UTILITY no longer fails with a spurious checksum error on block 0 of an unblocked standard data set.

P3373 UTIL - "INVALID INDEX" IN "INITIALIZE"

When generating a "REQUIRED STRUCTURES NOT NAMED" error for INITIALIZE, an INVALID INDEX occurred if the data base had 48 or more structures. This problem has been corrected.

P3389 UTIL - "LIST, WRITE" STATEMENTS

When large Index Sets or Standard Variable format data sets were printed via a UTILITY LIST or WRITE statement and the RECORD option was specified, UTILITY failed with an INVALID INDEX. The INVALID INDEX occurred at 8015400 for index sets and at 83713000 for Standard Variable format data sets. This problem has been corrected.

P3390 UTIL - "LIST, WRITE" OF BLOCK ZERO

When block zero of an index set was printed via a UTILITY LIST or WRITE statement and both the HEX and RECORD options were specified, UTILITY printed the wrong block. Rather than printing block zero, the last available block was printed.

P3391 UTIL - "LIST, WRITE" FAIL TO PRINT SOME BLOCKS

LIST and WRITE statements failed to print some blocks for Index Sequential, Index Random, Ordered List, Unordered List and Unordered Data Set structures. When the RECORD option was specified, blocks immediately following available blocks were skipped.

P3392 UTIL - BLOCK LIMITS FOR "WRITE, LIST"

When block limits were specified in WRITE and LIST statements and block zero was not specified, UTILITY failed to work properly. Index Sequential, Index Random, Ordered List, Standard Data Sets and Ordered Data Sets were printed correctly only when block zero was selected.

P3402 UTIL - "FLUSHDB" DEFAULT

FLUSHDB was only assigned a default value when either the WORKERS or NOZIP options was specified. The default is now set whenever FLUSHDB is not specified.

P3403 UTIL - "BUILDDUMPDIRECTORY" NOT ACCEPTED

UTILITY was only accepting the abbreviation BUILDDUMPDIR; now it accepts the full command BUILDDUMPDIRECTORY.

P3404 UTIL - SYNTAX ERRORS

UTILITY was not strictly enforcing the syntax described in Section 4 of the DMSII Utility and Operations Guide. Syntax errors are now produced when incorrect options are specified or the same option is specified more than one.

P3405 UTIL - "DIRECTION" ATTRIBUTE ERROR

When an IOERROR occurred during a dump to PACK, an attribute error occurred for DIRECTION. This attribute error has been corrected.

P3493 UTIL - "COPY ONTO" MAY NOT UPDATE "EOF" POINTER

COPY = ONTO *= now properly updates the end-of-file pointer.

P3495 UTIL - WORKERS RESTARTABLE ONLY ONCE

When UTILITY was restarted, any workers started during that run were given a Halt/Load worker recovery file, using the current UTILITY mix number, rather than that of the restarted UTILITY run. If another restart were attempted, these workers were not restarted. This problem has been corrected.

P3586 UTIL - VALIDATE BLOCK RANGE FOR "LIST, WRITE"

When a <block range> was specified for a LIST or WRITE, UTILITY was not:

1. Making sure the address was valid hex numbers.
2. Making sure the second <hex block address> was larger than the first.

A syntax error is now generated if either is incorrect.

P3616 UTIL - HANDLING OF HYPHENS

The use of a hyphen is not documented correctly in the DMSII Utilities and Operations Guide (Form No. 5001803), as follows:

Page 4-6:

"An <identifier> is composed of from 1 to 17 <letter>s, <digit>s, and hyphens. The first character must be a <letter>. The last character must not be a hyphen."

The requirement that an <identifier> not end with a hyphen was not being enforced. Now, a syntax error is generated if the last character in an <identifier> is a hyphen.

Page 4-9:

The syntax for <tape name> should be changed as follows:

<tape name>

```

-----<letter>-----
|  |-----| | |
|  |-----<letter>-----|
|  |-----|  |-----<digit>-----|
|  |-----/15\-----<letter>-----|
|  |-----<digit>-----|
|  |-----hyphen (-)-----|

```

This syntax documents the ability to use a hyphen in a <tape name>, which has been permitted but not documented.

UTILITY INITIALIZE did not work correctly for structures with names beginning with "ALL-". When encountering an initialize of such a structure, all structures would be initialized. Now, just the specified structure will be initialized.

P3673 UTIL - MULTIPLE ROW CONTROL FILES

When writing a multiple row control file for a dump, UTILITY was repeatedly writing out the first row rather than each successive row. Now, each row will be written.

P3690 UTIL - CHECKSUM ERROR FOR BLOCK ZERO

Block zero of unblocked DIRECT data sets was being inappropriately checksummed; consequently, an incorrect checksum error was generated following a reconstruction of row zero. This problem has been corrected.

P3702 UTIL - DISPLAY NONFATAL ERRORS, WARNINGS

Previously, only fatal error messages were written to the UTILITY output and displayed. Nonfatal error messages and warnings were written to the UTILITY output but not displayed. This has been altered so that warnings and nonfatal error messages, as well as fatal error messages, are both displayed and written to the output.

P3704 UTIL - DEADLOCK

Previously, a deadlock would result if a hard I/O error occurred while writing the control file. This problem has been corrected.

P3705 UTIL - MULTIPLE DUMPWORKER ERROR

When a dump was taken to multiple tapes, the second dumpworker was receiving an erroneous software error in the control file handler. This problem has been corrected.

P3714 UTIL - RECOVER FAMILY INDEX

When recovering rows using backup from multiple dump tapes and a destination family index was specified, the family index was only being used for the first dump tape. The family index placement of the rows on the remaining dump tapes was arbitrary. Now, the destination family index specification applies to all the specified dump tapes.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - WFL/COMPILEACR

```

-----

```

D3303 COMPILEACR - CHANGES TO "WFL/COMPILEACR"

With the advent of re-entrant ACCESSROUTINES code files, the ACCESSROUTINES need not be compiled for those data bases which "model" another common data base. This new feature required a change to the DATABASE/WFL/COMPILEACR job deck to allow the ACCESSROUTINES to be conditionally compiled. To solve the problem more generally, the job deck has been modified to accept a list of software items to be compiled. Also, to facilitate the use of re-entrant ACCESSROUTINES, the name of the ACCESSROUTINES code file may be specified. This may be necessary if ACCESSROUTINES must be compiled with a file title different from the default: ACCESSROUTINES/<data base name>. Another change from the 31 release is the addition of abbreviations for some of the longer parameter keywords. This makes running of the job deck easier. The parameter string to the WFL deck still consists of <keyword>=<value> pairs. The syntax and meaning for each keyword is given below. The abbreviation, where there is one, is the underlined portion of the keyword.

DB = <data base name>

This specifies the data base for which DMS software is to be compiled. This name is used to build the description and code file titles.

SOURCE = <pack family>

This specifies the pack family where the DMSII system software resides; i.e., the DATABASE/= and SYSTEM/= files.

OBJECT = <pack family>

This specifies the pack family where the generated code files are placed.

DESCRIPTION = <pack family>

Specifies where the description file resides.

AUDIT = SET or RESET

Set for audited data bases.

SUBSYSTEM = <global memory subsystem identifier>

Enables created code files to run in the proper subsystem.

PARTITIONS = SET or RESET

Set if partitioned structures exist. This forces the compilation of PARTITIONCONTROL, unless an explicit compile list is given which excludes PARTITIONCONTROL.

INITPARTITIONS = SET or RESET

Set to initialize partitions using DMCONTROL.

INITIALIZE = SET or RESET

Set to initialize all data base files using UTILITY.

ACR = <file title>

Specifies an ACCESSROUTINES code file title if different from the default ACCESSROUTINES/<data base name>.

```

      |<----- , -----|
-- COMPILE -- = ---<software name>----|

```

This allows the explicit specification of the software items to be compiled. Valid <software name>s are:

ACCESSROUTINES OR ACR, UTILITY, RECOVERY,

```

-----

```


DATA RECOVERY, RECONSTRUCT, PARTITION CONTROL, MONITOR

Example:

START DATABASE/WFL/COMPILEACR("DB=TESTDB OB=DEVELPK C=RECOV, DATAR,
RECON INIT=SET")

This set of parameters to the job deck causes the compilation of RECOVERY, DATA RECOVERY, AND RECONSTRUCT for a data base "TESTDB" whose description file is DESCRIPTION/TESTDB. The created code files will be placed on "DEVELPK". All the data base files will be initialized using UTILITY.

D3478 COMPILEACR - ADD COMPILE "MONITOR" FUNCTION

The DATABASE/WFL/COMPILEACR WFL deck can now be used to compile the data base monitor program by specifying "MONITOR" or "MON" in the compile list parameter to the deck.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - WFL/COMPILEACR

P3619 COMPILEACR - HYPHENATED DATA BASE NAMES

Hyphenated data base names are now handled correctly.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - WFL/COPYAUDIT

D3270 WFLCOPYAUDIT - IMPLEMENT "COPYAUDIT WFL" DECK

Currently, at audit file switch time the ACCESSROUTINES do one of three things depending on the option specified when the audit trail is declared in DASDL. If no option is specified, no action is taken. If the <diskaudit option> or <tapeaudit option> "VERIFY" is specified, then the COPYAUDIT utility is zipped to verify the audit. If the <diskaudit option> "COPY TO TAPE AND REMOVE" is specified, then the COPYAUDIT utility is zipped to verify the audit, copy it from disk to tape, and remove it from disk.

A WFL file has been implemented to run COPYAUDIT. It can be started by either the ACCESSROUTINES or the user. The ACCESSROUTINES will no longer zip COPYAUDIT directly, but will always zip this WFL job to run COPYAUDIT. When the audit trail is declared in DASDL, the name of this WFL job can be specified; otherwise, the default title of DATABASE/WFL/COPYAUDIT will be used. At the completion of each audit file, the ACCESSROUTINES automatically start the WFL job passing it two string parameters. One is the parameter string to be used to run COPYAUDIT, and the other is the subsystem in which COPYAUDIT should be run. This subsystem is specified in DASDL using the \$ SUBSYSTEM option.

This feature offers the following advantages:

- a. In the WFL job, a CLASS, FAMILY, PRIORITY, USERCODE, etc. can be specified to ensure that COPYAUDIT is inserted in the proper queue, finds the necessary files, and runs at the desired priority.
- b. If an error occurs which prevents COPYAUDIT from running successfully, the WFL job will offer the option of retrying the COPYAUDIT run or terminating the job. If the job must be terminated, then the WFL job may be restarted after the errors have been resolved. The parameters used by the ACCESSROUTINES to zip the WFL job are displayed, so that the WFL job may be restarted using the same parameters.

The following syntax replaces that in the DMSII DASDL Reference Manual (Form No. 5001480), Pages 4-20 through 4-22:

```
-- AUDIT TRAIL -----|
|-----| ( <audit trail attributes> ) -|
|-----|
| - ATTRIBUTES -|

<audit trail attributes>
|<-----, -----|
|-----|
| -/1\- AREAS = <unsigned integer> -----|
|-----|
| -/1\- AREASIZE = <unsigned integer> -----|
|-----|
|-----| - BLOCKS -----|
| -/1\- BLOCKSIZE = <unsigned integer> -----|
|-----|
|-----| - SEGMENTS -----|
|-----| - WORDS -----|
| -/1\- CHECKSUM -----|
|-----|
| - = FALSE -----|
| - = TRUE -----|
| -/1\- DUPLICATED -----|
|-----|
| - ON <secondary audit media> -----|
| -/1\-<primary audit media>-----|
| -/1\--- UPDATE EOF --- = <unsigned integer> -----|
|-----|
| - UPDATE-EOF -| - BLOCKS -|
```

<primary audit media>

```

----- KIND = ----- TAPE -----
|
|   - TAPE7 -- | -<tapeaudit option>-----
|   - TAPE9 -- |
|   - PETAPE - |
|
|----- DISK -----
|
| - KIND = - | | -<diskaudit option>-
|
| - DISKPACK -----
|
| - PACK ----- | | - = <family name> -----
|
| <-----
|
|-----/1\-- KIND = ----- DISKPACK -----
|
|           | - PACK -----
|
| -/1\-- PACKNAME=<family name> -

```

<secondary audit media>

```

----- TAPE -----
|
|   - TAPE7 -- | -<tapeaudit option>-----
|   - TAPE9 -- |
|   - PETAPE - |
|
|----- DISK -----
|
| - DISKPACK ----- | | -<diskaudit option>-
|
| - PACK ----- | | - = <family name> -

```

<diskaudit option>

```

| <----- , -----
|
|-----/1\-- ALTERNATE ----- TAPE -----
|
|           | - IS - |   - TAPE7 --
|           |         |   - TAPE9 --
|           |         |   - PETAPE -
|
| -/1\--<verify option>-----
|         | -<copy option>-----

```

<tapeaudit option>

```

| <----- , -----
|
|-----/1\-- DESIGNATED <serial numbers> -----
|
| -/1\--<verify option>-----

```

<serial numbers>

```
----- <unsigned integer> TO <unsigned integer> -----|
| - <string> TO <string> -----|
```

<verify option>

```
-- VERIFY -----|
| - JOB <file title> -----|
```

<copy option>

```
-- COPY TO --- TAPE ----- AND REMOVE ----->
| - TAPE7 -- | | - 1 TIMES - |
| - TAPE9 -- | | - 2 TIMES - |
| - PETAPE - |
>-----|
| - JOB <file title> -----|
```

<file title>

```
--<file name>-----|
| - ON <family name> - |
```

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - TRANSACTION PROCESSING

D3334 TPS - "TPS" REVISION

The Transaction Processing System document released with the Mark 31 system notes has been revised for the Mark 32 release. See Appendix C, "Transaction Processing System", for details. Revisions are indicated by PCN bars in the right margin.

D3419 TPS - UPDATING FROM "31" TO "32 TPS"

A transaction base and its data base need not be updated to Mark 32 concurrently. A Mark 31 transaction base may access a Mark 31 or a Mark 32 data base. A Mark 32 transaction base may only access a Mark 32 data base. Note that a Mark 32 transaction base may not access a Mark 31 database.

A transaction base and the MCP may be updated to Mark 32 software independently. Mark 32 TPS may be used with the Mark 31 MCP and Mark 31 TPS may be used with the Mark 32 MCP.

Because of the changes made to port files on the Mark 32 MCP, special restrictions apply to the Remote Library and the Host Interface. For this release only, Mark 31 Remote Library and Host Interface software may not be compiled or run on the Mark 31 MCP. Installations which do not use the Remote Library and Host Interface software are not bound by this restriction. This limitation will not exist on future releases.

The following procedure may be used to update to the Mark 32 TPS release.

1. Copy and compare the Mark 31 TFL source, trdescription file and system software to tape using Library Maintenance.
2. As a precaution, you may copy the TPS control files and journals to tape using Library Maintenance.
3. Load the Mark 32 TPS software and perform TFL update. In order to ensure an easy return to Mark 31, do not make any changes to the TFL description of the transaction base during the update. Following successful update, Mark 32 system software will automatically be compiled.
4. User programs will continue to run on Mark 32 without being recompiled.

The following procedure may be used to return to Mark 31 software provided no changes were made to the TFL description of the transaction base during TFL update.

1. Reload the Mark 31 TFL source, trdescription file and system software from tape.
2. User programs compiled with Mark 32 compilers must be recompiled with Mark 31 compilers before they are run with Mark 31 TPS.

The following D-Notes contain information which is especially useful for conversion to Mark 32 TPS.

a. TRANSACTION PROCESSING D3334

TPS documentation has been enhanced and revised for Mark 32.

b. HOSTLIB D3367

The Transaction Library now allows simultaneous read/write access to transaction journals.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - TFL

D3319 TFL - COMMENTS IN TRANSACTION BASES

Comments can be included when a transaction base is described in TFL. This descriptive information is stored in the transaction base description file. When a COBOL program invokes a transaction base, the comments are listed with each transaction format and transaction base. For ALGOL and PL/I programs, transaction formats, transaction items and comments are listed when the LISTDB dollar option is set.

Comments must be enclosed in quotes and may be at most 255 characters in length. Comments may be provided for transaction formats and transaction items. For transaction formats, the comment must immediately follow the keyword FORMAT; for transaction items, the comment must immediately follow the item name.

Syntax:

```
<transaction format>
--<format name>----- FORMAT ----->
      | - TRANSACTION - |      | - " <comment> " - |
>-<record description>-----|
```

```
<group item>
--<group item name>----- GROUP -- ( --<item list>--->
      | - " <comment> " - |
>- ) -----|
      | -<occurs option>-|
```

```
<data item>
---<alpha item>-----|
| -<boolean item>-| | |<----- , -----| |
| -<number item>--| | |---/1\<occurs option>-----| |
| -<real item>----| | |---/1\<initial value option>-| |
| -<field item>---| | |
```

```
<alpha item>
--<alpha item name>----- ALPHA -- ( --<integer>----->
      | - " <comment> " - |
>- ) -----|
      | - SIZE VARYING -----|
      | | - WITH --<numeric item name>-|
      | - SIZE DEPENDING ON --<numeric item name>-----|
```

```
<boolean item>
--<boolean item name>----- BOOLEAN --|
      | - " <comment> " - |
```

```

<number item>
--<number item name>----- NUMBER ----->
      | - " <comment> " - |
>- ( -----<integer>----- ) -----|
      | - S - |      | - , --<integer>- |

<real item>
--<real item name>----- REAL ----->
      | - " <comment> " - |
>-----|
      | - ( -----<integer>----- ) - |
      | - S - |      | - , --<integer>- |

<field item>
--<field item name>----- FIELD ----->
      | - " <comment> " - |
>- ( -----<integer>----->
      | |----- ; -----| |
      | /48\----<bit name>-----| |
      | - " <comment> " - | | - BOOLEAN - |
>- ) -----|

```

D3320 TFL - CRUNCH "NEWSOURCE" FILE

New source files produced by TFL will now be crunched, have a blocksize of 420, and an areaseize of 1008 records.

D3336 TFL - PARAMETERS NOT SPECIFIED

TFL now produces an error when a TFL source specifies either a data base parameter and no restart data set parameter or a restart data set parameter and no data base parameter. This error message has been added to ensure that both parameters are provided if data base processing is to be performed.

Syntax:

```

      | <----- , -----|
-- PARAMETERS -- ( ---<parameter spec>--- ) --|

```

<parameter spec>

```

-----<boolean parameter>-----|
      | - SET -----|
      | - RESET -----|
      |-----|
      |<numeric parameter>-- = ---<numeric literal>-----|
      | /1\-<data base parameter>-- , --<restart data set parameter>-|
      | /1\-<restart data set parameter>-- , --<data base parameter>-|
      | /1\-<host system parameter>-----|

```


SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - TFL

P3073 TFL - EXPAND TEXT GENERATION ARRAY

The size of the array in the TFL compiler which holds ALGOL text has been increased to 4095 words.

P3194 TFL - ERROR MESSAGE

The message "ERROR IN FORMAT" is now displayed; previously, "ERROR IN DATA SET" was displayed.

P3195 TFL - HEADER DISPLAYS CORRECT SYSTEM TYPE

The header line displayed on output listings from the TFL compiler now displays the correct system type.

P3557 TFL - HYPHENATED "RESTARTDATASET" IDENTIFIERS

TFL allowed the data base Restart Data Set to be declared in the PARAMETERS declaration. Previously, if the <restart data set name> was hyphenated, TFL issued a syntax error. This problem has been corrected.

P3674 TFL - TABLE SIZE EXCEEDED

Previously, if the source file was large, TFL terminated with a SEG ARRAY error. The fault occurred when the size of an array containing the properties of a Format/Subformat was exceeded. This problem has been corrected.

P3675 TFL - INFINITE LOOP

Previously, if a syntax error were issued at the beginning of a transaction format, it was possible for the TFL processor to get into an infinite loop. This problem has been corrected.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - TRINTERFACE

P3242 TRINTERFACE - CORRECT OFFSET GENERATION

TRINTERFACE was not generating the correct item offsets within transaction records for items within non-occurring groups. As a result, SEG ARRAY errors were given when transaction record groups items were accessed. This no longer occurs.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - TRUTILITY

P3104 TRUTILITY - SEARCH USING ALPHA ITEMS AS KEYS

TRUTILITY is capable of searching journal files for transaction records which have a specified value in a data item. If an alpha item longer than 18 characters were specified, TRUTILITY would fault with a SEG ARRAY error. This problem has been corrected.

P3388 TRUTILITY - OUTPUT ENTIRE "RANGE" SPECIFICATION

For search requests TRUTILITY lists an expanded version of the "RANGE" specification it received as a parameter. If the length of the generated output was greater than the MAXRECSIZE of the output media, then only a portion of the generated output was listed. This problem has been corrected.

P3715 TRUTILITY - COMPILATION FAILS WITH SYNTAX ERROR

If TRBASE/UTILITY were compiled against a large transaction base, the compilation could terminate with the syntax error "PROGRAM SEGMENT TOO LARGE". This resulted from the inclusion of a large amount of ALGOL text from the transaction base description file. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - HOSTLIB

D3085 HOSTLIB - RETURN ADDRESS OF LAST USER TRANSACTION

A new entry point, RETURNLASTADDRESS, has been provided which returns the address (file, block, offset) of the last transaction to be either tanked or processed by a transaction user. The declaration for the new entry point is the following:

INTEGER PROCEDURE RETURNLASTADDRESS (FILENUM, BLOCKNUM, OFFSET, IDNUM);
REAL FILENUM, BLOCKNUM, OFFSET; INTEGER IDNUM.

IDNUM must represent an active transaction user for this program. The address is returned in the three parameters: FILENUM, BLOCKNUM and OFFSET. This entry point allows transactions to be easily detanked. For example, RETURNLASTADDRESS could be called after tanking one transaction and called again after tanking is complete. SEEKTRANSACTION can be used to reposition the journal at the first tanked transaction.

D3367 HOSTLIB - SIMULTANEOUS "READ/WRITE" ACCESS

The TRANSACTION LIBRARY now allows simultaneous READ/WRITE access to transaction journal data files. As a result, the semantics of the READ USEROPTION parameter of the OPENTRBASE library entry point have been changed as follows:

USEROPTION=READ (4)

Allows any other process to open a journal when this USEROPTION is chosen. The other processes can use any value for USEROPTION they choose; thus, READers can simultaneously process against the same journal as Updater's, Inquirer's, Tanker's and Exclusiveupdater's.

In addition, a new exception has been added to the "Attention" group, as follows:

Exception #36:
Last call to READTRANSACTION resulted in a non-fatal error.

D3454 HOSTLIB - NEW STATISTICS

Three statistics have been added to provide information regarding data base ABORT and HALT/LOAD recovery, which come under two categories: ABORT statistics and Transactions Reprocessed.

ABORT statistics describe the number of ABORTs that have occurred and the average ABORT recovery time; they are only produced if at least one ABORT has occurred.

The statistic describing the number of reprocessed transaction is always produced and is a cumulative total of all transactions reprocessed during a single session.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - HOSTLIB

P2864 HOSTLIB - CHECK FOR UNASSIGNED TRANSACTION FORMATS

Properties of unassigned transaction formats are no longer included at compile time.

P3749 HOSTLIB - "I/O" COMPLETE

When a new journal file is created, a system transaction, with a userinfo subformat, is written to the new file beginning at segment one. Previously, however, the result of the system transaction I/O was not examined; consequently, an I/O failure might go undetected. This problem has been corrected.

P3780 HOSTLIB - WRITE ERROR

If a write error occurred while transaction tanking or data base processing was underway, the statistics for the average I/O wait time would be incorrect. Average wait time was a large negative number. This problem has been corrected.

P3794 HOSTLIB - DISCONTINUITY OF BLOCK SERIAL NUMBERS

Previously, if multiple TANKERS or UPDATERS were processing transactions against their respective journals, and one of them called the RETURNRESTARTINFO entry point, a block serial number discontinuity would be created in either the TANKER's or UPDATER's journal file. This problem has been corrected.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - REMOTELIB

D3230 REMOTELIB - ELIMINATION OF RESPONSE FROM PORTS, SIGNALS

The response required specification of the PORTS and SIGNALS syntax has been eliminated from all READs and WRITEs to PORT files in both the REMOTELIB and HOSTINTERFACE.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - REMOTELIB

P2881 REMOTELIB - ELIMINATE UNUSED REQUESTCASE VALUES

Both the REMOTELIBRARY and HOSTINTERFACE have been modified to eliminate "Requestcase" values which have never been used.

P3707 REMOTELIB - USE PORT FILES

The Remotelibrary and Hostinterface have been changed to utilize Mark 32 port files.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - HOSTINTERFACE

P2775 HOSTINTFACE - SEND ENTIRE RESTART RECORD

HOSTINTERFACE now ensures that the entire restart record, which it received from REMOTELIBRARY, is sent to HOSTLIBRARY along with its user transaction.

P2881 HOSTINTFACE - ELIMINATE UNUSED REQUESTCASE VALUES

Both the REMOTELIBRARY and HOSTINTERFACE have been modified to eliminate "Requestcase" values which have never been used.

P3350 HOSTINTFACE - ELIMINATE "PORT" OPTION

The PORT option has been eliminated from Programdump statements.

P3707 HOSTINTFACE - USE PORT FILES

The Remotelibrary and Hostinterface have been changed to utilize Mark 32 port files.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - TRPROPERTIES

D3321 TRPROPERTY - CRUNCH "TRBASE/PROPERTIES" SYMBOLIC

The new symbolic file produced when TRBASE/PROPERTIES is run will now be crunched, have a blocksize of 420, and an areaseize of 1008 records.

D3322 TRPROPERTY - "TASKVALUE" ATTRIBUTE

The TASKVALUE attribute of TRBASE/PROPERTIES is now 1 if the program terminates normally and 0 if the program terminates abnormally.

E

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DMS II - TRPROPERTIES

P2889 TRPROPERTY - INCREASE NUMBER OF FILES IN JOURNAL

The size of the file number fields has been increased in transaction records, in the journal control file, in a user information array (contained in the HOSTLIB called CFUSERINFO) and in the TRINTERFACE program, so that file numbers up to 9999 will be accepted.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - DATA DICTIONARY

D3335 DATADICT - DATA DICTIONARY

There is a documentation file on the SYSTEMNOTES tape named DATADICT/DOCUMENT, which contains information regarding the implementation of the Data Dictionary.

This file may be printed by using the commands described in GENERAL note D3205.

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - DDDASDL

D3327 DDDASDL - ADD VERSION LEVEL

The DASDL source for the Data Dictionary system now includes a version card.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DMS II - DDUPDATE

D3149 DDUPDATE - REFORMAT DATADictionary REPORTS

The reports generated by the DATADictionary have been reformatted.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DUMPALL
-----**D3011 DUMPALL - UPPER CASE INPUT STRING**

The input string to DUMPALL will now be translated to UPPER CASE before any processing takes place.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DUMPALL

P3468 DUMPALL - "PACK=<PACKNAME>" SYNTAX

DUMPALL did not scan "PACKNAME=<packname>" correctly in the verbs LIST and COPY and in the <device to device> routines. In some cases, DUMPALL did not accept "PACK=<packname>" without delimiting blanks around the equal sign. In other cases, the "PACK=" syntax was totally ignored or completely rejected by DUMPALL.

This problem has been corrected; DUMPALL now scans and process "PACK=<packname>" syntax properly for all verbs.

P3504 DUMPALL - BAD "COPY" SYNTAX NOW FLAGGED

DUMPALL did not flag "COPY" syntax errors, but punched cards as output instead.

Examples:

COPY FILE1 AS JUNK TO DISK
COPY FILE1 TO WHAT

This problem has been corrected; DUMPALL now flags COPY syntax with the error message "UNRECOGNIZED DEVICE KIND".

P3505 DUMPALL - LIST WITH "<MANUAL INPUT>","<PACKNAME>"

Using the LIST verb with <manual input part> and <packname> specification now works properly. Previously, this combination failed with an "ATTRIBUTE ERROR" and "DIVIDE BY ZERO" error.

P3537 DUMPALL - "DUMPALL" OVERRIDES "ON PACKNAME"

DUMPALL now checks if the input string contains any "ON PACKNAME" specification for DPK or DSK routines and sets the familyname to this specification. If there is no "ON PACKNAME" specification, DUMPALL assigns "PACK" as the familyname for DPK routines and "DISK" for DSK routines.

DUMP ANALYZER

D3012 DUMPANALY - ANALYZE "IOCB"

The analysis of the IOCB has been expanded.

D3013 DUMPANALY - ANALYZE "FIB" AT ADDRESS

New syntax has been added to tell DUMPANALYZER to assume that a FIB starts at an arbitrary address and analyze it.

Syntax:

```
-- FIB -----<simple address>--|
      | - AT - | - HEX - |
      |       | - OCT - |
      |       | - DEC - |
```

The FIB specified at address <simple address> is printed.

D3019 DUMPANALY - "REPEAT" SYNTAX IN "INTERACTIVE" MODE

The following syntax now exist:

```
-- REPEAT -----|
      |----- PRINTER -|
      | - TO - |
```

REPEAT causes the last command input to be repeated. If PRINTER is specified, the output goes to the printer.

D3136 DUMPANALY - PRINT INTERACTIVE INPUT

When running the interactive version of DUMPANALYZER and invoking the PRINTER option, DUMPANALYZER would dump the analysis to the rinter but not the command which caused the information to be dumped. It now dumps the command to the printer also.

D3137 DUMPANALY - DUMP "UNITMAP"

A new procedure called PRINTUNITMAP prints and analyzes the UNITMAP entry for each unit. In addition, PRINTUIOERR analyzes UNITIOERR.

D3138 DUMPANALY - PRINT FILE BUFFERS TEXT

The text contained in buffers is now printed unconditionally, if the FIB is being analyzed.

D3139 DUMPANALY - ANALYZE "UNITCONTROL"

DUMPANALYZER can now analyze the UNITCONTROL array of the PHYSICALIO module in response to the "IUNIT" command.

D3292 DUMPANALY - "OLAYINFO" ANALYSIS

The interactive DUMPANALYZER now analyzes the overlay-file allocation. It finds all the descriptors to present or absent overlayable data, checks for overlapping allocation or other errors, and compares them with the OLAYINFODESC bit vectors.

Syntax:

```
-- OLAYINFO -----|
      |-----|
      | -<stack>-----|
      |               | -<olay add>-|
```

If <stack> is present, only the specified stack's OLAYINFO is analyzed; otherwise, all stacks are shown. If <olay add> is present, overlay allocations neighboring <olay add> are shown.

This syntax is also available via the HELP command.

D3417 DUMPANALY - "PROC" COMMAND IMPLEMENTED

The PROC command of the DUMPANALYZER may be used to control the resolution of address not in global memory.

Syntax:

```
-- PROC -----|
      |         |
      | -<number>-|
```

D3442 DUMPANALY - ANALYZE LIBRARY TEMPLATE

Key information in the library template for DUMPANALYZER is now printed out.

D3479 DUMPANALY - MODULE ALTERNATIVE SELECTION

DUMPANALYZER uses the information passed via the MCP procedure TAPEDUMP to select the BINDINFO data for the selected module alternatives. The names of the selected alternatives are now displayed by the OPT command (or in the options section of a non-interactive dump).

D3481 DUMPANALY - NEW PORT ANALYSIS

The DUMPANALYZER now analyzes 32 ports. The analysis happens whenever FILES mode is set as part of the expansion of the file. Additionally, a new command has been added to the DUMPANALYZER to allow selective expansion of ports: PRINTPORT.

Syntax:

```
-- PRINTPORT --<portindex>--|
```

This syntax is also documented within the DUMPANALYZER HELP information.

D3588 DUMPANALY - NEW "MODE" "PIB" OPTION

A new MODE option, PIB, has been implemented for the DUMPANALYZER. When PIB is set, arrays which have an unusual field of PIB mark in their memory links will be analyzed and displayed as PIBs.

D3589 DUMPANALY - "SAVE" COMMAND

The semantics of the non-interactive SAVE command have reverted to their pre-Mark 31 state; i.e., when "SAVE <filename>" is entered, DUMPANALYZER will save the dump information on disk and exit. No further analysis will be performed.

A new command, "SAVEANDGO <filename>" has been implemented. This command causes DUMPANALYZER to save the dump information on disk as above, and then do any additional analysis as specified by other options.

D3612 DUMPANALY - ANALYSIS OF "MLIP I/O" DATA STRUCTURES

The DUMPANALYZER now recognizes MLIP type data structures (B6900) and analyzes them in a manner analogous to MPX data structures given the same commands.

D3643 DUMPANALY - ANALYZING AREA DESCRIPTOR

DUMPANALYZER would sometimes die when analyzing an area descriptor with a large value in the length field.

On the Mark 32 system software release, the semantics of the batch mode FILES command has been changed. FILES now always causes the contents of the buffers to be printed and analyzed. The option BUFF has been removed.

Also on the Mark 32 release, setting the mode FIB in the interactive mode causes the buffers to be dumped and analyzed.

On both the Mark 32 and Mark 31 PR2 releases, whenever the buffers are dumped and analyzed, only the first and last 250 words of a large (greater than 500 words) buffer will be printed.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

DUMP ANALYZER

P2779 DUMPANALY - BAD PRINTER SKIP

Near the end of the output from a non-interactive Mark 31 dump analysis with AREADUMP set, the printer may slew and/or drop ready because of improper carriage-control specification in the memory statistics summary. This problem has been corrected.

P2780 DUMPANALY - RUN "DUMPANALYZER" IN "EBCDIC"

DUMPANALYZER now runs in EBCDIC rather than BCL. During the course of this change a few corrections and modifications were made; these are detailed below except for occasional self-evident changes in the formatting of output.

Operands in a stack dump are no longer displayed in BCL by default; this feature may be invoked by setting the BCL mode (but not ALL). The interactive command PV ... BCL functions as before, and if the buffers for a BCL-mode file are printed, BCL graphics are still shown.

The heading page has been somewhat reorganized for readability.

The convention that hex numbers are shown with leading digits and decimal numbers without leading digits is now followed in all cases, except display of times (e.g., 02:07:45).

In an areadump, areas whose mom descriptors occur in a stack are shown, for example, as MA=07F07 003C IN STK 09A. In the Mark 31 DUMPANALYZER, the stack-relative offset (e.g., 003C) was incorrectly being shown as merely the last four hexits of the mom address; this has been corrected. Also, moms in the MCP stack are shown relative not to the stack base but to D[0]; they are now so marked: for example, MA=0230E D0+20E STK 001.

The display of un-stuffed Indirect Reference Words in stack was occasionally wrong if they appeared above an inactive MSCW between active MSCWs at differing lex levels. This problem has been fixed.

P2797 DUMPANALY - CORRECT "ID" INITIALIZATION

Under some circumstances, DUMPANALYZER could get an INVALID INDEX at line 13430000 while loading "LINEINFO/NAMES"; it would display the fault and proceed without the MCP D0 names. The error has been corrected.

P3061 DUMPANALY - "MD RV <ADDR> FOR ALL" CORRECTION

The DUMPANALYZER was not correctly calculating the number of words to display when the FOR ALL and the RV options of the MD command were used to print a descriptor of other than SZF=0. This problem has been corrected.

P3353 DUMPANALY - REMOVE PORT, "SIGNAL" CODE

The DUMPANALYZER no longer analyzes old style ports and signals.

P3361 DUMPANALY - "SAVE" LARGE DUMP

DUMPANALYZER could not correctly SAVE and later reload a large dump; the LINEINFO dictionary was corrupted if the saved file exceeded 65535 segments of disk. The problem has been corrected.

Dump files SAVED with an earlier DUMPANALYZER are not compatible with the corrected DUMPANALYZER, and vice versa.

P3396 DUMPANALY - HEADING DATE FOR DISK INPUT

The heading produced by DUMPANALYZER from a dump file that had been previously "SAVED" had an incorrect date. The day part of the date was always "1". This has been fixed.

P3429 DUMPANALY - PROCESSOR LOOP AFTER "?END"

The DUMPANALYZER will no longer go into an infinite loop after a ?END is entered while the program is executing a command which has no output.

P3430 DUMPANALY - "HOSTINFO" DEIMPLEMENTED

The interactive DUMPANALYZER command "HOSTINFO" has been deimplemented, as it is not relevant to Mark 32 Ports.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

ESPOL INTRINSICS

D3601 ESPOLINTRN - DELETE OLD INTRINSICS

Since XALGOL is no longer supported, the intrinsics and the BINDER references have been deleted.

The old BASIC intrinsics have been deleted, since the codefiles which reference these are no longer supported.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

ESPOL INTRINSICS

P2933 ESPOLINTRN - CORRECTLY HANDLE "BASIC" STRINGPOOL END

Previously, the intrinsics would get a SEG ARRAY error if a string exceeded the string pool size by less than 6 characters. The intrinsics now handle this situation correctly.

P3042 ESPOLINTRN - FREEFIELD INPUT WITH COMPLEX "ALGOL"

When an ALGOL program read a value into a complex variable with freefield input, the imaginary part was rounded to an integer. This problem has been corrected.

P3299 ESPOLINTRN - "ALGOL" POINTER "I/O"

The construct "WRITE(F,*,PA)" or "READ(F,*,PA)", where PA is a pointer, could cause a fatal error. These constructs are now handled correctly.

P3300 ESPOLINTRN - UPDATE "B7000" DEFINE

The ESPOLINTRINSICS will now correctly determine that the system on which the program is running is a B7000 series system.

P3309 ESPOLINTRN - ARRAY ROW FREE FORMAT READ

Free Format array row READs on a character type array no longer cause a SEG ARRAY error. This problem was caused by erroneous calculation of the record size if the parameter passed were a non-indexed string descriptor.

DOCUMENT CHANGES NOTES (D NOTES)

FILECOPY

D3020 FILECOPY - "TASKFAULT, CLASS, OLDWFL" FEATURES

New options have been added to FILECOPY, as follows:

TASKFAULT:

The use of this option will cause an "ON TASKFAULT, GO TO THETOP;" statement to be added to the task's output WFL deck. With the corresponding label "THETOP:", any abnormal termination (including operator DS) will cause the task to restart. This option is reset by default and will cause no changes in the current format of the output WFL deck. This option must be fully spelled out; i.e., it has no minimum abbreviation.

CLASS:

The use of this option allows the specification of the CLASS statement in the output WFL deck. If the CLASS option is not used, the WFL deck will inherit the CLASS of the SYSTEM/FILECOPY which created it. This option must be fully spelled out; i.e., it has no minimum abbreviation. The numeric value that follows it must be within the range of 0 - 1023.

OLDWFL

The use of this option will cause FILECOPY to create "OLD WFL" output decks which will allow quoted file titles. WFL (as opposed to pre-29 WFL) will not allow any characters in a file title other than letters and numbers, hyphens and underscores; the use of quotes is not allowed. The use of a hyphen and/or an underscore in a file title will cause FILECOPY to quote the title only if "OLDWFL" is set. For FILECOPY task requests that do not use the "OLDWFL" modifier, files that violate the title syntax will be reported on the summary report and will not be copied. The use of the "OLDWFL" modifier precludes the use of the modifier "TASKFAULT" which uses WFL features not supported in pre-29 WFL.

Note: Eventually, OLDWFL will no longer be supported; its use should be curtailed.

As a result of these new options, the syntax diagram on Page 1-1-5 of the SOG Reference Manual, Volume 2 (Form No. 5001688) should now read as follows:

```

|<-----|
|          |<---- , ----|
|----- EXCLUDE FILEKIND ( ---<filekind>--- ) -----|
|          |<----- , -----|
| - PARTITION ( --- <letter><hyphen><letter> --- ) -
| - INDEXLABEL ---<filename>-----
| - DECKLABEL --|          |-<family part>-----
| - LOCKINDEX -----
| - NOSUMMARY -----
| - COMPARE -----
| - BACKUP -----
| - PUNCH -----
| - REMOVE -----
| - LOCKDECK -----
| - NOZIP -----
| - SFACTORZERO -----
| - SYNTAX -----
| - CLASS --<integer>-----
| - /1\--- TASKFAULT -----
|          | - OLDWFL -----

```

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

FILECOPY

P3201 FILECOPY - "*" FILE REQUESTS

FILECOPY will now handle the following requests with the same results:

FILE (= FROM DISK)
EXCLUDE (USERCODE/= FROM DISK)
and
FILE (*= FROM DISK)

Both requests will result in only "SYSTEM" files (non-usercode) being selected. Please note that the "*" request will use much less time and system resources.

P3220 FILECOPY - "EXCLUDE USERCODE/="

FILECOPY will now handle the following in the same way:

EXCLUDE (USERCODE/=)
and
EXCLUDE (=USERCODE/=)

P3431 FILECOPY - "WFL" DECK SEQUENCE NUMBER LIMIT

FILECOPY will now correctly generate its WFL output files with sequence numbers incremented by 100 within a task. A task may now use up to 16,384 lines instead of the old limit of 4095.

P3432 FILECOPY - "NULLFILE" VALID FILEKIND

FILECOPY will now accept "NULLFILE" (VALUE=0) as a valid filekind in an EXCLUDE <filekind> statement.

P3433 FILECOPY - "INCLUDE" DOES NOT "EXCLUDE" AUTOMATICALLY

If a file meets the request criteria to be copied and is also in an INCLUDE list, it would be copied to the output media twice. This problem has been corrected by causing a file in an INCLUDE list to be EXCLUDED automatically.

P3434 FILECOPY - LOOP ON INVALID SYNTAX

FILECOPY will no longer loop giving syntax errors when extra commas or missing parentheses occur in FILE, EXCLUDE and INCLUDE statements.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

FILEDATA

P3158 FILEDATA - HANG ON "NO FILE"

Under certain circumstances, FILEDATA's GETSTATUS call could indicate no errors but return no files to be procssed; subsequently, this would cause a wait for NO FILE JOB <mix #>. This condition is now detected and reported as the error "NO FILES VISIBLE".

P3159 FILEDATA - VALID REQUESTS REJECTED AFTER FIRST ERROR

Multiple requests to FILEDATA may be input by separating them by a seimcolon; however, when an error was encountered in one request, all the following requests were discarded, even though they might be valid. This has been corrected; now, each request will be processed independently.

P3160 FILEDATA - INCORRECT INDICATION OF "IAD" DISK

The AREA LAYOUT OF FILES report incorrectly indicated that files with an odd numbered AREAClass resided on IAD disk. The correct bit in the file header is now tested.

P3506 FILEDATA - REPORT ON "5N" DISK

FILEDATA now properly reports on 5N disk.

P3507 FILEDATA - "4-DIGIT" SERIAL NUMBERS

In the heading of FILEDATA, only 3 digits were being printed for the system serial numbers. They have been expanded to 4 digits.

P3508 FILEDATA - SHOW LAST FILE IN "CHECKERBOARD"

The CHECKERBOARD command now shows the last file at the highest address.

P3509 FILEDATA - "FILEORGANIZATION" ATTRIBUTE

FILEDATA now properly reports the FILEORGANIZATION of a file.

FORTRAN

D3004 FORTRAN - "NEWTAPE" WITH "UNITS=CHARACTERS"

UNITS=CHARACTERS should not be specified for the compiler's NEWTAPE file. The following paragraph should be added to the FORTRAN Reference Manual (Form No. 5001506), at the bottom of Page 18-3:

"Note: If the attributes UNITS or MAXRECSIZE are file equated in the compiler files, the compiler input/output will be undefined."

D3005 FORTRAN - "MOD" OPERATOR WHEN SECOND ARGUMENT <1

Add the following to the FORTRAN Reference Manual (Form No. 5001506), after the second paragraph on Page 15-13:

"In the intrinsics "MOD" and "AMOD", for the second argument <1, the results of the "MOD" operators are undefined by the current implementation."

D3006 FORTRAN - "ERRORFILE" CONTAINS GARBAGE

Add the following to the FORTRAN Reference Manual (Form No. 5001506) to the description of ERRLIST, Page 19-7:

"The compiler ERRORFILE file should be equated to printer or remote only; otherwise, it may contain unexpected characters."

D3021 FORTRAN - INTRINSIC AS ACTUAL ARGUMENT NOT ALLOWED

An intrinsic name cannot be used as an actual argument.

The following changes should be made to the FORTRAN Reference Manual (Form No. 5001506):

Page 8-1 The next-to-last phrase should read:

"f. A FUNCTION or SUBROUTINE name (see Chapter 15 for exceptions)."

Page 15-1 The first paragraph under "Dummy Argument Lists" should read:

"Each element of a dummy argument list may be a simple variable name, an array name, a dummy subroutine name, or an asterisk (*). A simple variable may optionally be enclosed in slashes (/)."

Page 15-10 The next-to-last subparagraph should read:

"* indicates that the intrinsic is an in-line intrinsic; that is, code is placed in the code file at the point of the intrinsic reference to perform the indicated function."

D3022 FORTRAN - "DEBUG DUMP" WITH "OWN, SEPARATE"

Add the following to the FORTRAN Reference Manual (Form No. 5001506), Page 17-3, before the Example:

"Using the DEBUG DUMP statement together with options OWN and SEPARATE set in a subroutine is not allowed."

D3023 FORTRAN - "DEBUG TRACE" WITH NO FILE SPECIFIED

The following changes should be made to the FORTRAN Reference Manual (Form No. 5001506):

Page 17-5 Delete the following from "DEBUG TRACE STATEMENT":

"3. DEBUG TRACE t"

Page 17-6 Delete the following sentences from the first paragraph:

"If f is not specified, then the previous value of f is assumed. If f has never been specified, then tracing action is suppressed."

D3093 FORTRAN - WARNING MESSAGE FOR "\$LEVEL"

The LEVEL option is no longer recognized as a valid option. The documented action for this option has never been implemented; this note merely serves to remove this documentation.

A warning message will be printed when \$LEVEL is used.

As a result, the following changes should be made to the FORTRAN Reference Manual (Form No. 5001506) to delete any references to the LEVEL option:

Page 19-4 Delete the ninth line: "LEVEL (p)".

Page 19-10 and 19-11

Delete the description of the LEVEL option.

Index-8 Delete "LEVEL Option".

D3174 FORTRAN - DEIMPLEMENT "SIGNAL, RESPONSE" CLAUSES

The SIGNAL and RESPONSE clauses of READ and WRITE statements have been deimplemented on the Mark 32 release. A warning is produced when SIGNAL and RESPONSE are used on the Mark 31 PR1 release.

D3217 FORTRAN - "BCL" WARNINGS

A warning will now be issued for INTMODE=BCL in a file declaration or in OPEN and CHANGE statements.

A warning will now be issued that the dollar options BCD, B5500 and B5700 will be deimplemented on the 33 release.

D3231 FORTRAN - DEIMPLEMENT "VECTORMODE" OPTION

The VECTORMODE option for the B7000 series has been deimplemented. An attempt to use the option will result in a warning message.

D3362 FORTRAN - MODIFICATIONS TO SUPPORT PORT FILES

FORTRAN currently does not allow multiple-subfile I/O. The FORTRAN syntax and restrictions for port file I/O are described as follows:

1. Only formatted I/O is allowed when using a port file.

For example:

```
READ (1, 100) A, B, C
```

or

```
WRITE(1, 100, END=99) A, B, C
```

where 1 : is declared as a file with KIND = PORT.

100 : is the format which describes the I/O list.

A,B,C : are the elements of the I/O list.

2. Using unformatted I/O with a port file will result in a run-time error.

For example:

```
READ (1) A, B, C
```

```
WRITE (1, END=99) A, B, C
```

(See Mark 32 GENERAL note D3650, "Implementation of Port Files", for a description of port files.)

In order to provide access to the port open options, the OPEN statement has been modified as follows:

<open statement>

```
-- OPEN -- ( --<file designator>----->
```

```
>----- ) -----|
```

<pre> <----- , ----- -----/1\ OPENTYPE -- = -- "WAIT" ----- -----/1\ RESULT -- = --<identifier>----- -----<file attribute list>----- </pre>

The CLOSE statement has been modified to access a port file:

<close statement>

```
-- CLOSE -- ( --<file designator>----->
```

```
>----- ) -----|
```

<pre> <----- , ----- -----/1\ RESULT -- = --<identifier>----- -----/1\<existing-options>----- </pre>
--

D3474 FORTRAN - FORMAL PARAMETER IN "DEBUG DUMP" LIST

Add the following to the FORTRAN Reference Manual (Form No. 5001506), Page 17-3, after the fourth paragraph:

"Using a formal parameter in the list of simple variables and array names is illegal."

D3542 FORTRAN - "REAL" , "AIMAG" INTRINSICS FAILED

FORTRAN programs fail with an instruction time-out when using the complex intrinsics REAL or AIMAG in a loop with VECTORMODE set.

Add the following to the FORTRAN Reference Manual (Form No. 5001506), Page 20-4:

"e. Intrinsic, REAL, AIMAG are not allowed when the option VECTORMODE is set."

D3578 FORTRAN - LEGAL INPUT TO FORMAT

Data of the form 123+4 is legal input to the D, E and F formats. For E and F, it is equivalent to 123E4. For D, it is equivalent to 123D4. There is an example of this in the ALGOL Reference Manual (Form No. 5001639) under output examples for D, E formats.

The following manuals should be revised to include examples of this type of input:

ALGOL Reference Manual, Pages 4-28 to 4-30.
FORTRAN Reference Manual, Page 13-4 to 13-7.

D3635 FORTRAN - INVALID OP WITH "> PARAMETERS"

The FORTRAN compiler no longer faults with an INVALID OP when compiling a formal subroutine with 24 or more parameters which are arrays, the last one of which is subscripted; e.g.,

[illegible]

Insert the following paragraphs into page 15-8 of the FORTRAN Reference manual (Form No. 5001506) before the description about BLOCK DATA SUBPROGRAMS:

"SUBPROGRAM RESTRICTIONS:

Due to the complexities involved in the matching of actual parameter descriptions to formal parameter descriptions for formal subroutines, there is an implementation restriction which may prevent the use of certain combinations of parameters. This restriction becomes apparent only when the passing of subscripted variables must be resolved.

If the following rules are obeyed, there is no danger of encountering the restriction:

1. Actual parameters should be ordered such that subscripted variables do not occur beyond the twenty-third parameter.
2. Formal parameters should be ordered such that simple variables do not occur beyond the twenty-third parameter."

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

FORTRAN

P2709 FORTRAN - COMPILER LOOP ON "DATA" STATEMENT

An array used as the index of another array in the DATA statement caused an infinite loop in the compiler.

Example:

```
DIMENSION A(10,4), I(10)
DATA ((A(I,K), I=1,10), K=1,40)/40*0/
```

This problem has been corrected.

P2710 FORTRAN - CORE TO CORE "I/O" IN "WRITE" STATEMENT

A core-to-core WRITE statement caused an INVALID OP when trying to write a value, produced by the INTRINSIC function, to an array row.

Example:

```
REAL A(14)
I=1000
WRITE (A,1111) IABS(I)
```

This problem has been corrected.

P2711 FORTRAN - ALLOW DOUBLE PRECISION EXPRESSION AS SUBSCRIPT

A syntax error was given when a double precision expression was used as a subscript of an array. The problem has been corrected to allow double precision expressions as subscripts.

P2712 FORTRAN - DETECT "GOTO" <NON EXISTING STATEMENT NUMBER>

A GOTO <non-existing statement number> was not detected as a syntax error when followed by another syntax error in a subroutine. This problem has been corrected.

P2734 FORTRAN - STRANGE ACTION WHEN "CHECK, SEQERR" SET

When CHECK and SEQERR are both SET and LIST is RESET and the source input is out of sequence, then the message "CODE FILE NOT LOCKED DUE TO SEQUENCE ERROR" appeared at the top of the page. The heading and summary appeared on the next page. The card image in error was not shown on either the listing or the terminal.

This problem has been corrected. The message will not appear on the file LINE, but will be shown in the compiler's error file at the terminal.

P2735 FORTRAN - FORMAT ERROR ON READ FROM DOUBLE ARRAY

A core-to-core READ from a double precision array caused a format error #217 (RECORD OVERFLOW)

Example:

```
REAL D(2)
REAL*8 S(2)
.
.
.
READ (S,1) D
1 FORMAT ( 2(A5,7X))
```

This problem has been corrected.

P2736 FORTRAN - INVALID FILE ATTRIBUTE IN "FILE" STATEMENT

A legal file declaration statement followed by an invalid file attribute option in the file declaration statement caused an INVALID OP.

Example:

```
FILE 1 (TITLE="XX")
FILE 2 (KIND=DISK, UNITS=ZXX)
```

This problem has been corrected by giving a syntax error when the file attribute is invalid.

P2836 FORTRAN - "INVALID INDEX" WITH "SYLPT" TABLE OVERFLOW

When compiling a large subprogram with \$SEPARATE set, the compiler got an INVALID INDEX because the binder table overflowed. This problem has been corrected.

P2837 FORTRAN - "AUTOBIND, SEPARATE" SET IN MAIN PROGRAM

Using \$SET AUTOBIND, SEPARATE in the main program caused the following error message: "CODE. THIS FILE NOT CODE FILE". This problem has been corrected.

P2838 FORTRAN - "AUTOBIND" AND "GO" DESPITE SYNTAX ERROR

Autobinding was done in spite of the fact that the compile just completed had a syntax error. The code file which resulted was executable, but execution would fail with an incompatible level. This problem has been corrected.

P2867 FORTRAN - "\$" CARDS NOT IN "NEWTAPE" WITH "INCLNEW" SET

\$ cards in files being included were not entered in the NEWTAPE file, regardless of the setting of \$ INCLNEW. This problem has been corrected.

P2868 FORTRAN - FILESIZE ESTIMATE INCORRECT

The estimate of the core required by the file takes no account of the fact that the file may be UNITS=CHARACTERS; therefore, jobs required more core than they really needed. This problem has been corrected.

P2869 FORTRAN - PARAMETERS ARE CALL BY NAME

A statement function now will pass parameters "by name" when OPT=0; previously, parameters were passed "by value".

P2904 FORTRAN - BLANK CARD AT END OF SUBROUTINE

An extraneous blank card at the end of a subroutine caused autobinding to a host to fail.

Example:

```
100 $SET AUTOBIND SEPARATE
200 $HOST IS OBJECT/HOST
300 SUBROUTINE SUB
400 RETURN
500 END
600 <" blank card ">
```

This problem has been corrected.

P2905 FORTRAN - SPURIOUS ERRORS WITH "\$VOIDT, \$MERGE" SET

If the card file had \$VOIDT MERGE and a continued line, as follows:

```
$VOIDT MERGE
  I=          300
*2           400
```

and if the tape file had a normal statement which had the sequence number between the range of the continuation lines in the card file, as follows:

```
I=2          350
```

then the compiler would not recognize the continuation line, causing spurious syntax errors. This problem has been corrected.

P2943 FORTRAN - "\$SEPARATE" WITHOUT OTHER STATEMENTS

When compiling a program containing only dollar cards and the option SEPARATE or LIBRARY, the compiler did not give any error. As a result, binding produced an invalid host which could not be used for further binding. The error message "RECOMPILE WITH MARK 2 OR LATER" appeared when trying to bind to the invalid host. This problem has been corrected.

P2944 FORTRAN - PARAMETER MISMATCH

Using arrays as parameters in FORTRAN LIBRARY procedures caused parameter mismatch with programs written in other languages. This problem has been corrected.

P2945 FORTRAN - ARRAYS SEGMENTED

Formerly, if an array were not declared as COMMON, it would be segmented even if the option LONG were set. Now, large arrays are no longer segmented when LONG is set.

P2993 FORTRAN - "INV PCW"

An INV PCW at run-time occurred when an external subroutine was used as an actual argument and subsequently was called directly.

Example:

```
EXTERNAL X
CALL Y(X)
CALL X
END
SUBROUTINE Y(Z)
CALL Z
RETURN
END
SUBROUTINE X
RETURN
END
```

This problem has been corrected.

P3007 FORTRAN - TOO MUCH STORAGE ALLOCATED

For long strings passed as a parameter, the compiler allocated one word for each character in the array even though the string was packed six characters per word. This problem has been corrected.

P3013 FORTRAN - "INVALID INDEX" DUE TO CONFLICTING "COMMON"

An INVALID INDEX, which occurred when a FORTRAN program had a fault due to "CONFLICTING COMMON ALLOCATION", has been corrected.

P3014 FORTRAN - "W2 COMPILER ERROR" WHEN USING IOLIST

A "W2 COMPILER ERROR", which occurred when a freefield I/O with a long IOLIST was used, has been corrected.

P3156 FORTRAN - ALLOW FAMILY NAME IN "\$INCLUDE"

The FORTRAN \$INCLUDE option did not allow an "ON <familyname>" in the file title. This problem has been corrected; a \$INCLUDE statement may now contain a family name.

P3435 FORTRAN - "\$INCLUDE <INTNAME>" CORRECTED

The FORTRAN compiler no longer gets an attribute error nor hangs with a NO FILE condition when the following dollar card is used in a FORTRAN deck:

```
$INCLUDE <intname>
```

P3469 FORTRAN - COMPILER LOOP

Compiling a DATA statement with an extra parenthesis would cause the compiler to go into an infinite loop and not give a syntax error.

Example:

```
DIMENSION A(2,2)
DATA ((A(I,J),I=1,2)/1,0,0,1/
END
```

The above example now produces a syntax error.

P3540 FORTRAN - EXTRA COMMA IN PARAMETER LIST

The compiler now generates an error (instead of an INVALID INDEX) when compiling a function declaration with an extra comma in the parameter list.

Example:

```
100      SUBROUTINE S
200      G = F(A2,B2,C2,D2)
300      RETURN
400      END
500      FUNCTION F(A,B,C,D,)
600      F=A+B+C+D
700      RETURN
800      END
900      STOP
1000     END
```

Line 500 in the above example now generates an error.

P3541 FORTRAN - BLANK CAUSED SYNTAX IN "FILE" STATEMENT

A FORTRAN program compiled with a FILE declaration statement in a BLOCK GLOBALS saying "MYUSE=OUT" no longer generates a syntax error specifying unrecognized file option.

The following example now works properly:

```
BLOCK GLOBALS
  FILE 2(KIND=TAPE, MYUSE=OUT, MAXRECSIZE=80)
END
STOP
END
```

P3542 FORTRAN - USE CORE TO CORE "I/O"

Core-to-core I/O in which the source vector is a formal array parameter with the dimension less than the actual dimension no longer fails with format error #217.

The following example now works properly:

```
FILE 1(KIND=REMOTE)
DIMENSION A(3),B(72)
CALL XX(A,B)
STOP
END
SUBROUTINE XX(SEND,RECV)
  DIMENSION RECV(1),SEND(1)
  N=17
  READ(SEND,10))RECV(I),I=1,N)
  10 FORMAT(72A1)
END
```

P3543 FORTRAN - INVALID COMMON ALLOCATION

The compiler no longer loops after encountering an invalid common/equivalence space allocation.

The following example now works properly:

```
COMMON /A/W
COMMON /B/X,Y,Z
EQUIVALENCE (W,Y),(W,Z),(R,X)
W=0.
X=0.
END
```

P3544 FORTRAN - EXPORTED UNTYPED FUNCTION

The FORTRAN compiler no longer faults with an INVALID OP when an untyped function is exported from a library.

P3545 FORTRAN - FREE FORMAT READ WITH "REAL *8" VARIABLE

When executed from CANDE, a single comma was transmitted to the program with a free-format READ, resulting in the second word of a double-precision variable being set to binary zero.

The variable is now left unchanged.

P3546 FORTRAN - INTEGER OVERFLOW

The FORTRAN compiler no longer aborts with INTEGER OVERFLOW when compiling a DO statement with OPT=1.

P3573 FORTRAN - DOUBLE ARRAY AS PARAMETER

FORTRAN no longer emits an extra "ZERO" after the code for a double-precision indexed array parameter when the subroutine is declared before the call.

P3592 FORTRAN - "INVALID INDEX"

The FORTRAN compiler no longer aborts with an INVALID INDEX when compiling an incorrect DATA statement.

Example:

```
DIMENSION IOP(3,10)
DATA IOP(2,4)/6HDINAM /, IOP(2,5)/6HGRAF /
END
```

This example now causes an error.

P3615 FORTRAN - ERRONEOUS PROGRAM AFTER WARNING

When a "WARNING" error message occurred in the declaration of a DATA statement, no data initializations were effective. This no longer occurs.

P3656 FORTRAN - UNORDERED PARAMETERS IN LIBRARY

Unordered parameters in a LIBRARY procedure did not produce errors; the program compiled successfully, but when executed, incorrect results were produced.

Example of a FORTRAN program using library:

```
BLOCK GLOBALS
FILE 5(KIND=REMOTE)
LIBRARY L(TITLE="OBJECT/LIB")
END
REAL FUNCTION S(A1,N1)
REAL A1(10)
INTEGER N1
IN LIBRARY L(ACTUALNAME="SUM")
END
INTEGER M
REAL B(10)
READ(5,/)M,B
T=S(M,B)
WRITE(5,/)T
STOP
END
```

where: OBJECT/LIB is an ALGOL library.
SUM is the ALGOL procedure name of library OBJECT/LIB.

Function S is called with unordered parameters, but the compiler did not detect the errors.

This problem has been corrected.

P3663 FORTRAN - USING VARIABLE "IF" IN "COMMON" STATEMENT

When IF was used as a variable in a COMMON statement, a syntax error occurred at every following executable statement.

Example:

```
10 COMMON F, IF
20 F=1
30 END
```

A syntax error occurred at line 20, and the following error message was emitted:

"F" STATEMENT ILLEGAL IN THIS PROGRAM UNIT"

This problem has been corrected.

P3776 FORTRAN - "DEBUG MONITOR" STATEMENT

Using the DEBUG MONITOR statement no longer causes the system to super halt.

GENERALSUPPORT

D3376 GENERALSUPP - "SYSTEM/GENERALSUPPORT"

The file SYSTEM/GENERALSUPPORT has been implemented to provide run-time support for codefiles in future releases.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

GENERALSUPPORT

P3042 GENERALSUPP - FREEFIELD INPUT WITH COMPLEX "ALGOL"

When an ALGOL program read a value into a complex variable with freefield input, the imaginary part was rounded to an integer. This problem has been corrected.

P3298 GENERALSUPP - "CTOD" TERMINATES ABNORMALLY

Under certain conditions, the CTOD intrinsic would terminate with an "INVLD SQR ARG". This problem has been corrected.

P3299 GENERALSUPP - "ALGOL" POINTER "I/O"

The construct "WRITE(F,*,PA)" or "READ(F,*,PA)", where PA is a pointer, could cause a fatal error. These constructs are now handled correctly.

P3309 GENERALSUPP - ARRAY ROW FREE FORMAT READ

Free Format array row READs on a character type array no longer cause a SEG ARRAY error. This problem was caused by erroneous calculation of the record size if the parameter passed were a non-indexed string descriptor.

P3325 GENERALSUPP - CORRECT "DSQRT" ERRORS

Two classes of errors in the Double Precision Square Root Intrinsic have been corrected. Square Roots of values which are double precision integers would not work because of incorrect normalization of the value. Also, very large numbers were handled incorrectly. Both of these errors have been corrected.

P3326 GENERALSUPP - CORRECT "GAMMA, DGAMMA"

Both GAMMA and DGAMMA produced results for negative integers and zero. These values now produce errors since the function is undefined at these values.

P3534 GENERALSUPP - EXPONENT UNDERFLOW IN "RTOR"

In the Mark 31 ALGOLINTRINSICS, there was no handling of exponent underflow. Now, RTOR handles the fault so that a fault statement will not accidentally trap the fault.

P3657 GENERALSUPP - UPDATE "B7000" DEFINE

The GENERALSUPPORT library will now correctly determine that the system on which the program is running is a B7000 series system.

DOCUMENT CHANGES NOTES (D NOTES)

GUARDFILE

D3493 GUARDFILE - DATA BASE GUARDFILES

Insert the following in the SOG Reference Manual, Volume 1, (Form No. 5001563), Page 12-1-1, just before the notes at the bottom of the page:

"The guardfile specified for the entire data base is only used when the data base is invoked directly. When a logical data base is invoked, the guardfile specified for that logical data base, if any, is used."

DOCUMENT CHANGES NOTES (D NOTES)

HOSTSERVICES

D3015 HOSTSERVICES - "HOSTSERVICES" CHANGES

SYSTEM/HOSTSERVICES and SYMBOL/HOSTSERVICES are required for Shared Resources (loosely-coupled) systems that are using * GLOBAL tm Memory Module as their only inter-system communications media. If BDLC data communication links are used instead of, or in addition to, the * GLOBAL tm Memory Module for inter-system communications media, SYSTEM/BNA and SYMBOL/BNA are required.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

DOCUMENT CHANGES NOTES (D NOTES)

INPUT-OUTPUT

D3055 IN-OUTPUT - "CENSUS" ATTRIBUTE

The file attribute CENSUS will return the number of queued messages when applied to a file with KIND=PORT. The same conditions and restrictions that apply to remote files also apply to port files.

D3076 IN-OUTPUT - "APL" FILE ATTRIBUTE

The file attribute APL has been implemented for disk/pack files to satisfy APL access restrictions. It may be set prior to the creation of a new file, and is effective only if the codefile creating the file itself has the attribute set. The attribute may be read any time the file is open. An attempt to open a file with APL set will cause the accessor to be DSed unless it also has APL set.

D3222 IN-OUTPUT - CLOSE "WITH LOCK"

The 1974 ANSI standard requires that files which are closed "WITH LOCK" cannot subsequently be reopened by the program. Because the enforcement of this restriction was anticipated, the CLOSE option "WITH SAVE" has been provided as a Burroughs COBOL74 extension intended for making a mass-storage file permanent while also allowing it to be reopened by the program. The "WITH LOCK" option continues to make a mass-storage file permanent and make tape units "NOT READY", and can be used as long as the logical file is not reopened.

D3254 IN-OUTPUT - DIRECT DATACOM "I/O" FOR SWAPJOBS

Direct I/O statements used with files of KIND=REMOTE now function the same for swappable tasks as for tasks not in swap space. Specifically, the TIMELIMIT attribute now works for READ statements of swappable tasks.

D3339 IN-OUTPUT - "LINENUM, PAGESIZE" ATTRIBUTES

The documentation of the LINENUM and PAGESIZE attributes in the I/O Subsystem Reference Manual (Form No. 5001779) should be changed as follows:

The second and third paragraph on page 3-35 should be replaced by the following:

"The [LINE <acxp>] form of an ALGOL WRITE statement is provided for positioning the output page. The action taken depends on the values of <acxp>, PAGESIZE, and LINENUM. The [LINE <acxp>] form also causes the action of that write statement to change from the normal ALGOL mode of space after printing to one of space before printing.

If <acxp> is greater than PAGESIZE, the page is spaced forward, if necessary, to the end of the logical page and the line is printed. LINENUM is set to 1 and PAGE is incremented. No spacing takes place after the line is printed. An end-of-page is returned to the program.

If <acxp> is greater than or equal to LINENUM and less than or equal to PAGESIZE, the page is spaced forward, if necessary, to the logical line number and the line is printed. LINENUM is set to <acxp> and no spacing takes place after the line is printed.

If <acxp> is less than LINENUM, the line is printed without spacing before or after. LINENUM is set to 1, PAGE is incremented, and an end-of-file is returned to the program.

A skip to a channel on the carriage control tape may also affect LINENUM. A skip to channel 1 has special significance and causes LINENUM to be set to 1. The ALGOL form for this is [SKIP 1]. Skipping to any other channel does not change LINENUM, but may cause it to become incorrect as an indicator of the actual position on the page. The value of PAGE is not affected by a skip to any channel."

The last sentence in the fifth paragraph on Page 3-35 should read as follows:

"After an end-of-page result, LINENUM is set to 1 and PAGE is incremented."

Insert the following paragraph after the second complete paragraph on Page 3-40:

"Setting PAGESIZE equal to 255 has special significance. It is similar to other non-zero PAGESIZE settings, except that end-of-page is never returned to the program and PAGE is never incremented by the system. LINENUM is counted continuously and wraps around from 255 to 0. The automatic skip to channel 1 after channel 12 is sensed is still suppressed."

In the ALGOL Language Reference Manual (Form No. 5001639), add the following reference to the discussion of the ALGOL construct "WRITE (<file designator>[LINE<acxp>])":

"See the documentation of the LINENUM and PAGESIZE file attributes in the I/O Subsystem Reference Manual (Form No. 5001779) for further details."

D3347 IN-OUTPUT - "FULLTRANSLATION" OPTION

On new systems, beginning with the B6900, the FULLTRANSLATION run-time option for the OP ODT message will not control the default setting of the file attribute TRANSLATE, which will have a default value of FULLTRANS. FULLTRANSLATION being reset on a B6700 or B6800 will continue to cause the default value for TRANSLATE to be DEFAULTTRANS.

See the I/O Subsystem Reference Manual (Form No. 5001779), Page 4-3 through 4-5, for details on software translation.

D3373 IN-OUTPUT - "DISPOSITION" FILE ATTRIBUTE

The file attribute DISPOSITION now returns the value 11 (eleven) if the file is not open.

D3407 IN-OUTPUT - "AREASIZE" VS. "NEWFILE"

On the Mark 35 release, the value of the AREASIZE attribute will no longer affect the assignment (or creation) of disk files. Currently, on the Mark 32 release, if the value of KIND is equal to DISK or PACK, the value of MYUSE equals OUT, and the NEWFILE attribute is unset, a new file will be created if AREASIZE is non-zero; but if AREASIZE equals zero a new file will be created only after an unsuccessful search has been made for a permanent file. (If NEWFILE has been set to TRUE a new file will be created regardless of the value of MYUSE and AREASIZE; if NEWFILE is equal to FALSE, a permanent file will be assigned unconditionally.)

To aid the transition to the use of NEWFILE (if overriding of the MYUSE attribute is desired) and away from the side effects of the AREASIZE attribute, starting with the Mark 32 release a warning message will be displayed along with a NEWFILE attribute error message, whenever a permanent disk file is assigned to a file whose MYUSE value is OUT and AREASIZE is zero.

The warning for COBOL programs will be as follows:

"ON 35, OPEN INPUT/OUTPUT MUST BE USED TO WRITE ON
PERMANENT FILE."

The warning for other programs will be as follows:

"ON 35, "NEWFILE=FALSE" MUST BE SPECIFIED TO
OVERRIDE "MYUSE=OUT".

D3408 IN-OUTPUT - "AREAClass" VS. "FAMILYINDEX"

The AREAClass and SPEED file attributes will be eliminated on the Mark 34 release. They will not be supported on new systems, starting with the B6900, and attempts to set these attributes will be ignored. Since the Mark 27 release, the file attribute FAMILYINDEX has been supported as an alternative method for controlling disk space allocation. On the 34 release, the FAMILYINDEX attribute will be the only method allowed. Programs compiled on Mark 34 using AREAClass or SPEED will receive syntax errors. The setting of these attributes on Mark 34 in existing programs will be ignored by all systems.

To aid in identifying programs which are using these attributes, warning messages of the form:

"ON 34, THE "AREAClass" ATTRIBUTE WILL BE DEIMPLEMENTED"
"ON 34, "FAMILYINDEX" MUST BE SPECIFIED TO CONTROL
SPACE ALLOCATION."

or:

"ON 34, THE "SPEED" ATTRIBUTE WILL BE DEIMPLEMENTED"
"ON 34, "FAMILYINDEX" MUST BE SPECIFIED TO CONTROL
SPACE ALLOCATION."

will be displayed, preceded by an attribute error message.

D3410 IN-OUTPUT - "MTBF" ELIMINATED

The MCP compile-time option MTBF has been eliminated. The items it previously controlled are now standard features of the MCP. Unit reliability statistics will always be gathered. The RF ODT message is now always valid, and RF DEGRADATION messages are displayed when unit reliability decreases.

The requirement for operator input in response to an RF DEGRADATION message is still under the control of run-time option 6 (DIAGNOSTICS).

D3482 IN-OUTPUT - NEW ATTRIBUTES IMPLEMENTED

The following new file attributes have been implemented, some of which reference others in this set and/or other attributes in the I/O Subsystem Reference Manual (Form No. 5001779):

AREAALLOCATED

Disk only, read only, assigned, Boolean

The attribute **AREAALLOCATED** indicates whether or not (TRUE or FALSE) a specific area of the associated physical file has been allocated. The **AREAALLOCATED** attribute requires an index, the area number, as a parameter. If the disk file is duplicated, the copy number is also required as a parameter. Area numbers begin at zero (0); copy numbers begin at one (1).

AREALENGTH

Disk only, read/write, anytime/closed, Integer

The value of the attribute **AREALENGTH** is the number of **FRAMESIZE** units in an area of disk file.

If the attribute has never been set, either **AREASIZE** or the default for **AREASIZE** will be used. If the attribute has been set, it will be used even if **AREASIZE** has been set. An attribute error will be given if **AREALENGTH** has been set and an attempt is made to set **AREASIZE**.

The **AREALENGTH** attribute may be set only when the file is closed. When a new disk file is created, the value of **AREALENGTH** is adjusted so that it is also divisible by **BLOCKSIZE**. When a file is created this value is associated with the physical file. The default value (used when **AREALENGTH** has been set to zero) is a value which corresponds to the number of records closest to 1000 which is divisible by the number of records per block. The maximum value is dependent upon the size of the disk or pack hardware.

If **AREALENGTH** is read while the file is assigned, the value returned is computed from the physical file. In this case a valid value will be returned even if **AREALENGTH** was never set.

BLOCKSTRUCTURE

General, read/write, anytime/closed, Integer

The attribute **BLOCKSTRUCTURE** specifies the format of the records and the structure of the file.

The mnemonics and meanings of the **BLOCKSTRUCTURE** attribute are as follows:

FIXED	Blocked or unblocked fixed-length records. This corresponds to FILETYPE 0. This value is the default value for BLOCKSTRUCTURE .
EXTERNAL	Variable-length records. Neither the record itself nor the structure of the file contains information about the length of the record; this must be specified externally in the I/O statement. This corresponds to FILETYPE 3.
VARIABLE	Variable-length records. The record length is contained in the first four characters of the record. This corresponds to FILETYPE 1.
LINKED	FORTTRAN-linked variable-length records. The link words are maintained by the logical I/O subsystem and are not part of the records. The mode of the records is assumed to be BINARY and software translation is never attempted. This corresponds to FILETYPE 6.

If the attribute has never been set, either **FILETYPE** will be used or if **FILETYPE** was not set, the default will be used. If the attribute has been set then it will be used even if **FILETYPE** has been set and **FILETYPE** will be set to a value corresponding to **BLOCKSTRUCTURE** when the file is opened. An attribute error will be given if **BLOCKSTRUCTURE** has been set and an attempt is made to set **FILETYPE**.

If **BLOCKSTRUCTURE** is **FIXED**, **MINRECSIZE** is set equal to **MAXRECSIZE**.

The blocking technique used with variable-length blocked files is as follows: If the length of the record just written plus **MINRECSIZE** plus the offset into the block is greater than **BLOCKSIZE**, or the record to be written plus the offset into the block is greater than **BLOCKSIZE**, a physical write is initiated for the block. In the latter case, the record becomes the first record of the next block. If the logical file is assigned to a peripheral unit (**KIND**) that allows variable-length blocks (e.g., tape files), then only the part of the block that is valid will be written. For peripheral units that require fixed-length physical blocks (e.g., disk files), a **MINRECSIZE** field of nulls (4"00") will be added to the block as an end of block marker and the whole block will be written.

The value of **BLOCKSTRUCTURE** has an effect upon the default value of **MINRECSIZE**. Variable-length files which have link words or record length fields contained within the record require **MINRECSIZE** to be at least large enough to hold this information. Files with **BLOCKSTRUCTURE** equal to **FIXED** use a value for **MINRECSIZE** that is equal to **MAXRECSIZE**.

If **BLOCKSTRUCTURE** is read while the file is open, the value returned is computed from the physical file. In this case a valid value can be returned even if **BLOCKSTRUCTURE** was never set. If **FILETYPE** does not correspond to a legal **BLOCKSTRUCTURE** and **FILETYPE** was used to determine the structure of the file, then an attribute error will be given if an attempt is made to read **BLOCKSTRUCTURE**.

COPYNAME

Duplicated disk only, read only, assigned, Pointer

The attribute COPYNAME returns the external file name of the specified copy when the logical file is assigned to a duplicated disk file. The COPYNAME attribute requires an index, the copy number, as a parameter. A null identifier (".") is returned if the copy index is greater than the number of copies of the duplicated file.

When a duplicated disk file whose FILENAME is A is created, the copies of the duplicated file will be given COPYNAMES of the form A/"COPY#01", A/"COPY#02", ..., etc. The order in which a permanent duplicated file's copies are assigned indices upon opening the file is dependent on the order that they occur in the directory. Copies of duplicated disk files can be removed, modifying the assignment of indices for the remaining copies in the duplicated file for all subsequent times the permanent file is opened. In addition, the copies' external file names can be changed, which can modify their order in the directory, thus affecting the assignment of indices when the permanent file is opened.

CURRENTRECORD

General, read only, anytime, Integer

The CURRENTRECORD attribute returns the size, in FRAMESIZE units, of the record currently in use. The CURRENTRECORD is equal to MAXRECSIZE unless BLOCKSTRUCTURE is EXTERNAL.

DEPENDENTSPECS

General, read/write, anytime/closed, Boolean

If the attribute DEPENDENTSPECS is TRUE, the format of the records and the structure of the logical file are to be determined by the structure of the associated labeled permanent file; that is to say, the attributes BLOCKSTRUCTURE, MINRECSIZE, MAXRECSIZE, BLOCKSIZE, and FRAMESIZE (also FILETYPE, UNITS, SIZEOFFSET, SIZE2, and SIZEMODE) will be changed to agree with the values used to create the permanent file. If no permanent file is associated with the logical file (i.e., a new file is being created), or if the permanent file is unlabeled, the attribute DEPENDENTSPECS is ignored.

If DEPENDENTSPECS has never been set and if FILETYPE is 7 or 8 (and BLOCKSTRUCTURE has not been set), FILETYPE will be used to determine the value of the above list of attributes. If DEPENDENTSPECS has never been set and FILETYPE is neither 7 nor 8, the user set or default values for these attributes will be used. If DEPENDENTSPECS has been set, it will be used no matter what value FILETYPE was set to. If DEPENDENTSPECS has been set and FILETYPE is 7 or 8, FILETYPE will be reset to zero before any other use of FILETYPE when attempting to open a file. An attribute error will be given if DEPENDENTSPECS has been set and an attempt is made to set FILETYPE to 7 or 8.

If DEPENDENTSPECS is TRUE and the permanent file was created with a FILETYPE that does not correspond to a BLOCKSTRUCTURE, an attribute error will be given at open time and BLOCKSTRUCTURE will subsequently act as if the user never set it.

FILENAME

General, read/write, anytime/anytime, Pointer

The attribute FILENAME is an external file name and is used to associate a logical file with a physical or permanent file. The default FILENAME for the file is value of the INTNAME attribute.

The FILENAME attribute may be changed when the logical file is open only when it is assigned to a disk file. Under these conditions, changing the FILENAME of the file causes the file name of the physical disk file to be changed also.

FILESTATE

General, read only, anytime, Integer

The FILESTATE attribute indicates the logical state of the file. If KIND=REMOTE or KIND=PORT and the MAXSUBFILES attribute is greater than one, a subfile index is required for the FILESTATE attribute. Only the values CLOSED and OPENED are valid for all KINDs. The other values have special meanings for REMOTE or PORT files.

The mnemonics and meanings of the FILESTATE attribute are as follows:

- CLOSED: The initial state of a subfile is CLOSED. The subfile returns to this state when it is closed by the user.
- AWAITINGHOST: This state indicates that the host specified by the HOSTNAME subfile attribute is unreachable. The subfile will remain in this state until the host becomes reachable. The FILESTATE may then change to OFFERED, OPENED, or CLOSED.
- OFFERED: A subfile enters this state when an open has been done and the host specified by HOSTNAME is reachable, but no matching subfile has been found.
- OPENED: This state indicates that the subfile is open and may be used for I/O operations.
- SHUTTINGDOWN: This state indicates that the system operator has requested that communications with the host involved in the support dialog be terminated.

This notification gives the program the opportunity to terminate in an orderly fashion; the port remains open and all I/O operations are valid.

BLOCKED: This state indicates that the remote host has become temporarily unreachable. The port remains open and all I/O operations are valid.

CLOSEPENDING: This state indicates that the user has closed the subfile, but the remote subfile has not yet acknowledged the closure. When close acknowledgment is received, FILESTATE changes to CLOSED.

DEACTIVATIONPENDING: This state indicates that the remote subfile has been closed and that the local subfile has data queued for input.

DEACTIVATED: This state indicates that the remote subfile has been closed and that the local subfile does not have data queued for input. Close is the only valid operation for a subfile in this state.

DENIED: This value indicates that file assignment has been denied.

POSTPONED: This value indicates that file assignment has been postponed.

DENIEDILLEGALUSE: This value indicates that an illegal open was requested.

The values of the mnemonics for FILESTATE are:

CLOSED	= 0
AWAITINGHOST	= 1
OFFERED	= 2
OPENED	= 3
SHUTTINGDOWN	= 4
BLOCKED	= 5
CLOSEPENDING	= 6
DEACTIVATIONPENDING	= 7
DEACTIVATED	= 8
DENIED	= 9
POSTPONED	= 10
DENIEDILLEGALUSE	= 11

FRAMESIZE

General, read/write, anytime/closed, Integer

The attribute FRAMESIZE indicates the number of bits to be transferred as a unit of data. The values of the attributes MINRECSIZE, MAXRECSIZE, BLOCKSIZE, AREALENGTH, CURRENTRECORD, and CURRENTBLOCK are expressed in FRAMESIZE units.

The possible values of the FRAMESIZE attribute are the framesize of the INTMODE attribute or 48 (one word). Thus, the legitimate values are 4 for HEX, 8 for EBCDIC or ASCII, and 48 for any INTMODE (6 for BCL is not allowed).

If the INTMODE is SINGLE, the value of the FRAMESIZE attribute is 48 (one word).

If the attribute has never been set, either UNITS or the default for UNITS will be used. If the attribute has been set, it will be used even if UNITS has been set and UNITS will be set to a value corresponding to FRAMESIZE when the file is opened. An attribute error will be given if FRAMESIZE has been set and an attempt is made to set UNITS.

If FRAMESIZE is read while the file is open, the value returned is computed from the physical file. In this case a valid value can be returned even if FRAMESIZE was never set.

An open error will occur if FRAMESIZE and INTMODE are incompatible.

SCREENSIZE

Datcom only, read only, open, Integer

SCREENSIZE is a station attribute (that is to say, to access the attribute, a Relative Station Number (RSN) must be specified). The attribute returns the number of lines on a page as specified in the Network Information File (NIF) description of the station. For example:

"NUMBEROFLINES : = REMOTEFID (RSN).SCREENSIZE;"

STATIONNAME

Datcom only, read only, open, Pointer

The attribute STATIONNAME is an external file name. The attribute requires an index, the Relative Station Number (RSN). The STATIONNAME attribute will return the name of that station as it is defined in the Network Information File (NIF). If the RSN is not valid, a null identifier (".") will be returned.

SUBFILEERROR

Port only, read only, assigned, Integer

The SUBFILEERROR attribute is set after each I/O operation and indicates the success or failure of that operation.

The mnemonics and meanings of the SUBFILEERROR attribute are as follows:

NOERROR	OK: No error occurred on the last I/O operation.
DISCONNECTED:	Communication with the remote subfile has been lost due to a network failure.
DATALOST:	As a result of a close operation, some data was not actually sent to the remote subfile before the file was closed.
NOBUFFER:	The last I/O was a Write with DONTWAIT and no buffers were available to do the I/O.

The values of the mnemonics for SUBFILEERROR are:

NOERROR	= 0
DISCONNECTED	= 1
DATALOST	= 2
NOBUFFER	= 3

D3499 IN-OUTPUT - "OPEN" INPUT REVERSE TAPE FILES

The OPEN procedure is now capable of positioning USASI labeled tape files for reading backwards. Prior to this implementation, the only way to read a tape file backwards was to open and then close the file with no rewind (CLOSE* in ALGOL).

D3585 IN-OUTPUT - "CYLINDERMODE" DESCRIPTION

The description of CYLINDERMODE on Page 3-18 of the I/O Subsystem Reference Manual (Form No. 5001779) should read as follows:

"When the CYLINDERMODE attribute is true (and the AREASIZE is less than or equal to the cylinder size), the areas of the diskpack file will be assigned so that no area will span a cylinder boundary."

D3645 IN-OUTPUT - TAPE DENSITY

The tape densities described on Page 3-19 of the I/O Subsystem Reference Manual (Form No. 5001779) should have the following values:

0	BPI800
1	BPI556
2	BPI200
3	BPI1600
4	BPI6250

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

INPUT-OUTPUT

P2771 IN-OUTPUT - "UNITNO" VS. "BACKUPTAPE"

UNITNO can now be set for PRINTER BACKUP and PUNCH BACKUP tape files.

P2811 IN-OUTPUT - DATACOM FILE, FAMILY ADDITION, SUBTRACTION

Adding a datacom family to a remote file which contained a member that had been previously subtracted from the remote file could lead to the file's station list being corrupted. This problem has been corrected.

P3067 IN-OUTPUT - "KIND" FILE ATTRIBUTE

When a PL/I program assigned the KIND file attribute a character string containing a correct mnemonic followed by illegal characters, if the first illegal character was less than "A" in the EBCDIC collating sequence, no error was generated.

Example:

```
DECLARE F FILE  
ENVIRONMENT(KIND='DISK,XXX');
```

Now, an error is properly generated.

P3074 IN-OUTPUT - CORRECT STATE ATTRIBUTE ON REMOTE "EOF"

The file attribute STATE was not being properly set after an EOF on a remote file. This problem has been corrected.

P3149 IN-OUTPUT - "UPDATE," BINARY "I/O" WRITING TO WRONG UNIT

Expanding a permanent disk file, using Binary I/O in update mode; e.g., UPDATEFILE=TRUE and WRITE(F,*,A) would cause the last buffer of the file to be written with an uninitialized I/O Control Word. In a simple example, CLOSE would get a fault; in a more complex example, the I/O would land on top of other records in the file. When the file was closed, the end of the file would not have been changed. These problems have been corrected.

P3169 IN-OUTPUT - UPDATE, BINARY "I/O" READ WRITE TRANSITION

Read write transitions using Binary I/O on a file being accessed with update I/O action no longer cause the end of file to be lost.

P3170 IN-OUTPUT - MINIMIZE HEADER UPDATE

ACTUALCLOSESERIALWRITE has code to check to see if the header should be written to disk; however, this code was effectively ignored if the unit were write-capable. The code now causes header update action only if the EOF is extended.

Note that the header timestamp update is controlled in other procedures; those algorithms are not affected by this change.

P3171 IN-OUTPUT - "BCL" BACKUP FILES

As was promised in the Mark 30 release in note D2516, backup files are now always stored in EBCDIC on disk or tape.

P3172 IN-OUTPUT - USE OF "TD830 ODT" FOR "SPO" FILES

Direct files now have the same output handling for SPO files as logical I/O and the DCALGOL intrinsic WRITESPO. The output data is scanned to verify that it terminates with an ETX; at the same time, the sequence 48"274D" (which causes the data to be retransmitted from the ODT to the system), if found, is changed to blanks. If the data fills the direct array without an ETX character, the last character in the array is changed to an ETX.

P3174 IN-OUTPUT - BREAK ON OUTPUT

The notification of break-on-output on a direct I/O multistation REMOTE file will be given only once if the I/O which receives the notice is a broadcast write. The mistaken use of a global variable previously caused the error reporting to be erratic.

P3226 IN-OUTPUT - PROTECTED FILES CLOSED

Small blocks on protected files are now handled correctly. A protected file closed with the last block smaller than one word was corrupted. The data of the last block was lost. This was only for protected file with UNITS=CHARACTERS.

P3633 IN-OUTPUT - ERROR MESSAGES CONTAIN LINE NUMBER

Error messages will now contain a line number (when line information is available). Where a line number has been substituted for a formatted RCW, the format of the information has been changed from:

"@ NNN:NNN:NN"

to

"@ (NNNNNNNN)".

Where line information is not available, the RCW is displayed as before. If the RCW (and line number) returned by STACKHISTORY differs from the RCW that was previously displayed, the original RCW and the line number from STACKHISTORY are both placed in the message in the form:

"@ NNN:NNN:NN @ (NNNNNNNN)".

P3634 IN-OUTPUT - RANDOM BADLY BLOCKED "I/O"

A sequence of random I/Os on a badly blocked disk file caused the system to do an I/O which extended beyond the end of the allocated area for the file. The problem has been corrected.

P3783 IN-OUTPUT - "I/O" RESULT FROM "SEEK" STATEMENT

Through the use of the STATE attribute, or in ALGOL using the SEEK statement as a Boolean function, it is possible to see the value of the I/O result variable. In most cases, this value is meaningless, because the seek has only been initiated (the operation being designed to cause an asynchronous I/O) and the result is not yet calculated, and is the result of the I/O completed prior to the SEEK statement.

To prevent confusion caused by associating the prior I/O result with the current seek operation, the I/O result value is now changed to zero just prior to the seek initiation.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

INTERACTIVEXREF

P3147 IXREF - ADD "DATABASE" AS "XREF" ITEM

Data bases declared in BDMSALGOL programs will now be properly referenced by the XREFANALYZER and INTERACTIVEXREF.

P3330 IXREF - REMOVE "PORT, SIGNAL" VARIABLE TYPES

The ALGOL variable types PORT, PORT ARRAY, SIGNAL and SIGNAL ARRAY are no longer supported. The EXPAND command was improperly deciding which variable types to expand. The table on which this command relies has been resynchronized with the ALGOL list of variable types.

DOCUMENT CHANGES NOTES (D NOTES)

JOB FORMATTER

D3026 JOBFORMAT - BAD RECORD DUMP

JOBFORMATTER no longer causes a dump when it encounters a record with a bad major field.

D3140 JOBFORMAT - NEW "EOT/EOJ" FORMAT

Previously, when a task/job completed, a line was printed showing "EOT/EOJ", whether or not the task/job completed properly. Now, if the task/job is DSed, the DS reason is printed instead of EOT/EOJ. If the task were a compile that terminated with a syntax error, "SNTX" is printed instead of "EOT". Also, timing information (such as PROCESSOR, I/O and ELAPSED times) are now printed in the following format:

hours:minutes:seconds.1000ths of seconds

D3575 JOBFORMAT - "PBIT" TIME ACCOUNTING

Four new words have been added to the EOJ log record to support the changes in P-bit time accounting. These words are located at the end of the fixed part of the record immediately after the READYQTIME word. For a precise description of the layout of this type of record, refer to the file SYMBOL/JOBFORMATTER on the SYSTEM tape.

The new words are the following:

INITPBITTIME: The processor time spent handling
initial P-bits for this task.
INITPBITCOUNT: The number of initial P-bits for this task.
OTHERPBITTIME: The process time spend handling all other
P-bits for this task.
OTHERPBITCOUNT: The number of other P-bits for this task.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

JOB FORMATTER

P2863 JOBFORMAT - PRINT BOXES IN ASCENDING ORDER

If a job runs in more than one local box on a B6800 multiprocessor system, the box numbers are printed in ascending order (i.e., 1, 2, 3).

P3436 JOBFORMAT - CHANGE "CONRAC" TO "ODT"

A unit type of 2 is now printed by LOGANALYZER as "ODT" instead of "CONRAC".

P3608 JOBFORMAT - "UNITMNEMONICS" ARRAY

The following problems have been corrected:

1. The value array UNITMNEMONICS had an incorrect entry (#24).
2. The define RESETBILLINGVARIABLES was resetting SM twice and not resetting CDP at all.

P3795 JOBFORMAT - "DP ALL" MESSAGE

The job summary message line for the command "?DP ALL" was sometime incorrect. This no longer occurs.

DOCUMENT CHANGES NOTES (D NOTES)

KEYEDIO

D3393 KEYEDIO - "KEYEDIO" IMPLEMENTATION

A new software item has been added for the Mark 32 system software release. SYMBOL/KEYEDIO and SYSTEM/KEYEDIO are used to support certain I/O facilities in COBOL74. The KEYEDIO program runs as a library, and is invoked automatically by code emitted by the COBOL74 compiler.

The KEYEDIO program supports multiple concurrent users, and is arranged so that record modifications are treated internally as a single operation, even if multiple I/O operations are involved. Since the KEYEDIO file consists of control information, record locator tables and data, this design ensures the consistency of all related file information at all times.

D3649 KEYEDIO - "KEYEDIO" PRIVILEGED PROGRAM

In order to ensure proper security checking of keyed files, the KEYEDIO library must be a transparent privileged program.

A "PP SYSTEM/KEYEDIO:TRANSPARENT" ODT message must be entered whenever SYSTEM/KEYEDIO is compiled.

DOCUMENT CHANGES NOTES (D NOTES)

LCOBOL

D3235 LCOBOL - WORKFILE COMPILED FOR "CANDE" COMPILERS

For compilations through CANDE, the workfile will now be compiled without the necessity of file-equating SORSE.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

LCOBOL

P2713 LCOBOL - "INVALID INDEX" FOR "WORKING-STORAGE"

The LCOBOL compiler will no longer get an INVALID INDEX when the total WORKING-STORAGE exceeds 2000 words.

P2873 LCOBOL - "SNTX" FOR COMPILE WITH SYNTAX ERRORS

The LCOBOL compiler now sets the TASKVALUE to -1 if a compile has syntax errors, thus causing SNTX to be reported in the job summary.

P2906 LCOBOL - SAME ADDRESS FOR TWO LEVEL "77" ITEMS

A compilation for SL3 object code would generate the same memory address for the first and second 77 level items. This problem has been corrected.

P3547 LCOBOL - GENERAL "LOAD" INSTRUCTIONS AGAIN

As a result of a previous patch, the LCOBOL compiler failed to generate LOAD instructions in some instances. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

LOADER

D3014 LOADER - "LH" COMMAND

SYSTEM/LOADER can now load diskpack FIRMWARE files on multiplexor and MLIP systems. The format of the command is the following:

```
-- LH -- PK <nnn> --- MPX <m> PATH <p> ----->
      | - VIA <port #> <lem #> <dlp #> - |
>-- <filename> FROM <tapename> -----|
```

D3157 LOADER - "LOADER" IMPROVEMENTS

Improvements have been made which alter the operation of the LOADER.

1. LOADER can now cool and cold start both head-per-track disks and packs.

Syntax:

```
-- LOAD --<filename>--- DISK ----->
      | - FROM <tapename> ----- |
      | - DISKPACK <familyname> - |
>----->
| - TO --- DISK -----|
|         |<unitno>-----| |
|         | - DISKPACK <unitno> -----|
|         |         |<familyname>-----|
|         |         | - SER <serialno> - |
>-- ; -----|
```

- a. If the "source clause" specifies DISK, the source file is taken from the family named DISK.

If the "source clause" specifies <filename> FROM <tapename>, the file is taken from tape.

If the "source clause" specifies DISKPACK <familyname>, the file is taken from the given pack.

- b. The LOADER now determines the Halt/Load unit before reading the LOAD command (from the HALTLOADEU command or as the default disk; i.e., the lowest head-per-track disk or pack on the system).

If the "TO clause" is specified on the LOAD command, it must specify the same disk unit. Previously, it was possible to change the preselected Halt/Load unit with the LOAD command.

- c. In the case of a cool start, the familyname and serial number of the Halt/Load unit will not be changed. If the optional <familyname> and <serialno> clauses are specified, the LOADER will check to ensure that the proper diskpack is mounted. A mismatch will cause the LOADER to abort.

In the case of a cold start, the familyname and serial number can be changed. The default is to leave the name and serial unchanged. If the Halt/Load volume is unlabelled, the default name is DISK and the default serial number is the Halt/Load unit number. The optional <familyname> and <serialno> clauses can be used to override these defaults for diskpacks; the HLFAMILY command can be used to change the familyname for head-per-track disk.

2. The HALTLOADEU command must be used to change the Halt/Load unit number. The default Halt/Load unit is the on-line head-per-track disk with the lowest unit number. If there are no on-line head-per-track disks, the lowest on-line diskpack is used.

Syntax:

```
-- HALTLOADEU --<unitno>-- ; --|
```

When used, the HALTLOADEU command must precede all other cards except LH and READISC.

3. The XD command can be used during a cool start or cold start for disk pack. Only the Halt/Load unit can be XDed.

Syntax:

-- XD --<unitno>-- ADDR --<starting segment>-- FOR --<length>-- ; -|

4. All LOADER cards must be terminated by a semicolon.
5. A card with an asterisk (*) causes the input to switch from the reader to the ODT. An "*" input from the ODT causes the input to switch back to the card reader. If, at BOJ time, the LOADER cannot find an on-line card reader, it will accept input from the ODT. In order to switch to a card reader, the operator should first READY the reader and then enter an "*" from the ODT. Previously, the LOADER required that the first command be read from a card reader; it would not accept input from the ODT until an "*" was read from the card reader.

The following restrictions apply:

1. HPT - only the family name DISK is allowed.

D3462 LOADER - "HALTLOADEU" MESSAGES

"H/LUNIT" has been changed to "HALTLOADEU" in various messages emitted by the LOADER.

D3567 LOADER - "OLAYROW" SIZE

The default overlay row size is 400 (not 800) segments.

The SOG Reference Manual, Volume 2 (Form No. 5001688), Page 5-1-34, the sentence "The default value is 800 segments" should be changed to read as follows:

"If OLAYROW is not set, the LOADER will use the row size already in effect for that disk (by previous COLDSTART). If there was no row size in effect, 400 will be used."

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

LOADER

P2778 LOADER - "IV"

The IV command now works as documented; previously, it was expecting a token (";") at the end of the statement.

P2962 LOADER - CONVERT "LOADER" TO "NEWP"

LOADER has been converted to the NEWP language.

P2986 LOADER - "206,207" DISK PACK COLDSTART

LOADER can now coldstart 206 and 207 disk packs that have "bad label"s. Normally a pack with a "bad label" would have to be Ived, but 206 and 207 disk packs can only be Ived via the SCR IVR command (on B6700/B6800 systems) or via PTD (B6900 systems).

P3053 LOADER - "INVALID ADDRESS" INTERRUPT

An INVALID ADDRESS interrupt, which occurred when running on a B6700 with more than one multiplexor, has been corrected.

P3437 LOADER - "INVALID INDEX" ON "MOD 63"

The LOADER no longer fails to initialize a system which has a MOD 63 on line.

P3694 LOADER - SEQUENCE

The LOADER will now check the record sequence number of blocks it reads from tape. A mismatch will cause an appropriate error message, and the load will terminate.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

LOG ANALYZER

D3384 LOGANALY - ADD STARTING, ENDING TIMES TO HEADING

The date and time of the first log entry and the last log entry have been added to the headings of the LOGANALYZER reports.

Also, the I/O time required to read the log has been reduced approximately 50% by changing the input blocksize from 300 words to 1500 words.

D3627 LOGANALY - "THAW" COMMAND

THAW (described in GENERAL note D3356) is now a valid type for the OPERATOR option.

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

LOG ANALYZER

--- -----

P3278 LOGANALY - MAINTENANCE LOG ENTRIES

Ever since LOGANALYZER was modified to handle IOCONDITION type entries, maintenance log entries have not been sorted properly in the reports. Now the maintenance log entries will again be sorted properly by unit number, date and time.

P3362 LOGANALY - "DL" MESSAGE "INVALID INDEX?"

LOGANALYZER will no longer take program dumps processing valid DL messages.

P3582 LOGANALY - LOG NOT FOUND

LOGANALYZER no longer ignores DL information if no parameters are given.

P3660 LOGANALY - UNRECOVERED ERRORS SHOWN

When the final result descriptor for a tape unit error entry had the BOT/EOT bit on, the retry would be counted as unrecovered, even though the I/O had been successful.

Bit 8 (BOT/EOT) of the tape result descriptor is now masked out along with bit 10 (short block) when testing for unrecovered errors. This also ensures that "101" result descriptors will not be counted as errors during retries.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

LOGGER

D3416 LOGGER - "LOGGER" VS "DL LOG"

Logger will now correctly find the SYSTEM/SUMLOG file if it has been relocated by use of the "DL" ODT command.

D3457 LOGGER - LINE EQUATED TO BACKUP TAPE WITH CATALOGING

For sites with CATALOGING set TRUE and a requirement to equate the LINE file of SYSTEM/LOGGER to BACKUP TAPE, the following syntax should be used:

Pre-29 WFL

FILE LINE (KIND=PRINTER BACKUP TAPE, LABELTYPE=STANDARD)

NEW WFL

FILE LINE (KIND=PRINTER, BACKUPKIND=TAPE, LABELTYPE=STANDARD)

The LABELTYPE=STANDARD will allow a volumed printer backup tape to be created.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

LOGGER

P3166 LOGGER - "ORGMCS, DESTMCS" INTEGER TYPE

The ORGMCS and DESTMCS jobsummary file items in LOGGER have been changed from type string to type integer. This will allow INCLUDEs and EXCLUDEs to work properly. This change will not affect internal storage of the information in the jobsummary file, but any report that uses either ORGMCS or DESTMCS in an INCLUDE or EXCLUDE statement must be changed.

P3357 LOGGER - COMMAS BETWEEN "SORT" ITEMS

LOGGER allows commas between SORT items.

Syntax:

```

-- SORT -----|<----- , -----|
               |<item>-----|
               | - BY - |
               | - ASCENDING -- |
               | - DESCENDING - |

```

P3363 LOGGER - YEAR TO DATE SORT ERRORS

LOGGER will now apply the user request sort parameters to the YEAR TO DATE report generation, thus allowing the user to control the amounts of memory and disk the sorts will use and to correct for sort errors when not enough space is available.

P3364 LOGGER - NO FILE JOBSUMMARY

LOGGER will no longer hang on a no file JOBSUMMARY/XXXX or STATISTICS/XXXX when an empty SYSTEM/SUMLOG file is used as input. LOGGER will take EOF action on the missing files.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

MCP-GENERAL

D3054 MCP-GENERAL - "SWAPPER" ENHANCEMENTS

The SWAPPER mechanism has been improved on the Mark 32 release. Many features have been expanded or enhanced; some features have been redesigned.

CHANGES TO THE SWAPPER PARAMETER MECHANISM

Previously, the SWAPPER parameters were stored in the first record of the SYSTEM/SWAPDISK file currently in use by SWAPPER (the packname was stored in MCPINFO). On the Mark 32 release, the parameters are stored in the directory of the Halt/Load unit. The parameters are read during MCP initialization and are always available (whether or not SWAPPER is running). During a CM operation, the parameters are copied to the new Halt/Load unit. In order to preserve compatibility with previous releases, a copy of the parameters is stored in each swapdisk file in Mark 31 format.

The SW ODT message is used to change or interrogate the SWAPPER parameters. On previous releases, the response to an SW message was given by displays from each SWAPPER independent runner. On the Mark 32 release, the CONTROLLER usually issues the response to the SW message; thus, the values of the parameters are available through the DCKEYIN intrinsic.

CHANGES TO THE SW ODT MESSAGE

Old Forms Deimplemented

The "SW" or "SW ON <packname>" message is now used only to interrogate the SWAPPER parameters. Only the "SW+" message is used to initiate SWAPPER. The "ON" and CORESIZE options of the "SW+" message have been replaced by the "FAMILY +" and "CORESIZE" options of the SW message.

See Mark 32 GENERAL note D3356 for a description of the syntax and semantics for the SW ODT message.

NEW SWAPPABLE ENTITIES

Libraries

General Rules

Libraries are now permitted in swapspace. If a library executes in swapspace, any program using the library as well as any task processed out of the library must also run in swapspace. On a B6800 Multiprocessor system, all tasks associated with the library (users and processes of) must also run in the same subsystem. Swapping a user of a library is functionally the same as swapping an offspring of the library; i.e., whenever the user is in memory, the library must also be in memory. However, the reverse is not true; when the library is in memory, any of the users of the library may or may not be in memory.

Specification

As with all tasks, the swappability of libraries is controlled by the SUBSPACES task attribute. The values of the SUBSPACES attribute for libraries are the same as for regular tasks. The SUBSPACES attribute can be set by two methods: compile-time specification and run-time inheritance. The SUBSPACES attribute can be specified at compile time through WFL. If no value of the SUBSPACES attribute is specified by compile-time equation, a library with a sharing class of PRIVATE inherits the value of the SUBSPACES attribute from the tasks which caused the library task to be initiated.

Data Bases

General Rules

On the Mark 32 release, the DMS ACCESSROUTINES (Data Base Stack) are initiated as an ordinary task with certain privileges. This task may now run in swapspace. As with libraries, if the DBS runs in swapspace, so must all its users.

Specification

The SUBSPACES task attribute is used to control the execution of DBSs in swapspace, just as it does for ordinary tasks. The attribute can only be specified at compile time in the ACCESSROUTINES codefile. The SUBSPACES attribute for a data base is never inherited.

JOBS

General Rules

Job stacks may now be run in swapspace. All of the rules for regular process families run in swapspace also apply to the WFL job and its offspring when run in swapspace. Note that any external process of a WFL job is a semidependent task; i.e., the job itself need not be swapped in in order to execute the offspring tasks. (See MCP-GENERAL note D3252 for a description of semidependent tasks.)

Specification

Again, the SUBSPACES attribute is used; note that only the values 0 or 3 are meaningful for a job stack. The attribute can be specified on the queue through which the job runs or on the job header. If either of these methods are used, the attribute is inherited by all tasks run out of the job. Operational overhead can be reduced by specifying a new option, SWAPALLJOBS (Option 32), which causes all job stacks to run in swapspace without forcing inheritance of the SUBSPACES attribute on the tasks run out of a job stack; thus, an installation could choose to run all job stacks in swapspace but not other stacks.

Stackswap

Programs which use the DCALGOL stackswap construct (such as CANDE) may now run in swapspace.

ELIMINATION OF SYSTEM/SWAPDISKMAKER

Since the SWAPPER parameters are no longer stored on swapdisk, it is no longer necessary to specify the SWAPPER parameters when a SYSTEM/SWAPDISK is created; instead, only the rowsize and the number of rows to be allocated are specified. The CREATE option of the SW ODT message (described in Mark 32 GENERAL note D3356) is used to create new SYSTEM/SWAPDISK files. The text of the input message is compiled by WFL (using TASK.FILECARDS format) to produce the attribute equation for the creation of the swapdisk file; thus, any combination of permissible file attributes can be easily specified.

SWAPPING ON MULTIPLE FAMILIES

SWAPPER now has the ability to use more than one family and more than one SYSTEM/SWAPDISK file on which to swap. The SW ODT message is used to create a swapdisk file on the desired family. The FAMILY option on the SW message is used to dynamically add to or subtract from the list of packs SWAPPER uses, as follows:

SW FAMILY + KSID

This message causes SWAPPER to find the file SYSTEM/SWAPDISK on KSID. This file is opened and made available to SWAPPER.

SW FAMILY - KSID

This message causes SWAPPER to cease using the family KSID. Also, all of the tasks which are currently swapped out on the family KSID are swapped in (from KSID) and then swapped out again (to a family other than KSID). When there are no more tasks resident on KSID, the file SYSTEM/SWAPDISK on KSID is closed and the operator is notified.

Multiple Families on a B6800 Multiprocessor System

On a B6800 Multiprocessor system, each processor running SWAPPER maintains its own list of families available to SWAPPER; thus, it is now possible to swap on a family that is not exchanged to every processor running SWAPPER. To support process family migration, at least one family of swapdisk must be usable by all of the processors in which the migratory task will run.

Multiple Swapdisk Families with Different Row Sizes

There is no requirement that the row sizes for each swapdisk family be the same; however, SWAPPER requires that each I/O operation fit entirely within a single row of a swapdisk. The maximum value of the MAXIOSIZE parameter is the size of the largest row of swapdisk available. This can allow an installation to segregate some classes of SWAPPER I/Os by setting the rowsize of some swapdisk families smaller, thus preventing subspaces with large I/Os (larger than the rowsize) from being swapped on that family. This capability can cause an "out of swapdisk" condition to occur whenever there is insufficient space on swapdisk(s) with large enough rowsizes, even though there may be space available on other swapdisk families. This situation can be relieved by either adding another family on which to swap (with a large enough rowsize) or decreasing the MAXIOSIZE parameter.

B6000 SERIES MARK 32

OUT OF SWAPDISK MECHANISM

When SWAPPER runs out of swapdisk, an independent runner is forked which then waits on an RSVP notifying the operator that more swapdisk is needed. The operator can either wait for the situation to repair itself (i.e., wait for a job in memory to complete) or add a new family available to SWAPPER via the SW ODT message. When running on a B6800 Multiprocessor system, it is possible to be out of certain kinds of swapdisk. Because a migratory task requires swapdisk that is used by each of the processors in which it can run, it is possible for this task to be out of swapdisk even though there may be space on another family (but this family is not being used by all of the processors). When this situation arises, the "out of swapdisk" independent runner displays the list of processor visibility swapdisks which are needed.

Example:

```
--- 1 WAITING ENTRY ---
3496/3496 OUTFSWAPDISK
3496 ** OUT OF SWAPDISK FOR PROC(S) (1),(2,3) **
```

The above example notifies the operator that some swapdisk is required that is usable by processor 1 and some swapdisk is required that is usable by processors 2 and 3.

NEW FORMAT FOR SWAPSPACE

MINCHUNKSIZE Attribute

On Mark 31 and earlier releases, the swapspace was required to be a single contiguous piece of memory. On Mark 32, the swapspace is now permitted to be any number of pieces of memory. These "chunks" may be located anywhere in memory. This feature allows dynamic growth of the swapspace, and it prevents "holes" in memory from severely affecting site operations. Even though the swapspace can be divided into several different pieces, each individual subspace must still reside in a contiguous area of memory. A segmented swapspace may not be able to handle the same number of tasks as a non-segmented swapspace with the same number of slots.

Example:

A swapspace with one chunk of 90 slots could run 3 tasks of 30 slots each simultaneously; a swapspace with a 50-slot and a 40-slot chunk can run at most two 30-slot tasks simultaneously.

To control this situation, a new parameter, MINCHUNKSIZE, has been implemented, which specifies the smallest amount of memory that a single chunk of swapspace may contain (MAXCORE <= MINCHUNKSIZE). By setting the MINCHUNKSIZE attribute to the value of the CORESIZE attribute, behavior similar to that implemented in Mark 31 and earlier releases may be obtained.

Changing the Swapspace Size via the CORESIZE Parameter

The semantics of the CORESIZE attribute remain unchanged from previous releases; however, the semantic effects of dynamically changing this attribute have been changed on the Mark 32 release. The basic philosophy is that SWAPPER will accept the new value of the parameter and attempt to bring the swapspace into conformance. The new value is preserved so that the next time SWAPPER is initialized it can get the exact amount of swapspace desired. At any time the amount of swapspace actually in use can be determined from the value displayed next to ACTUALCORESIZE in response to the SW keyin.

Expanding the Swapspace

By setting the value of the CORESIZE parameter greater than the current value of ACTUALCORESIZE, SWAPPER attempts to expand the amount of swapspace available by getting one or more "chunk"s of memory and inserting these into SWAPPER's tables. These new chunks of memory are constrained by the value of the MINCHUNKSIZE parameter. If the difference between the new value of CORESIZE and the current value of ACTUALCORESIZE is less than MINCHUNKSIZE, no increase in swapspace can be performed.

Shrinking the Swapspace

By setting the value of the CORESIZE parameter smaller than the current value of ACTUALCORESIZE, SWAPPER attempts to shrink the amount of swapspace available by shrinking or removing individual chunks of memory. No chunks of memory may be smaller than the MINCHUNKSIZE parameter. In some situations, SWAPPER is unable to shrink the swapspace as small as desired (SWAPPER never shrinks the swapspace smaller than CORESIZE). The current algorithm for deciding which pieces of swapspace are to be shrunk or discarded is to pick the smallest chunks first.

B6000 SERIES MARK 32

SWAPPER Initialization and Swapspace Size

At initialization time, if SWAPPER is unable to obtain the amount of swap space specified by the CORESIZE parameter, an RSVP is issued notifying the operator of the situation. The operator has four alternatives:

1. Reply DS to the RSVP, aborting SWAPPER initialization.
2. Reply OK, causing SWAPPER to again attempt to find the required memory (this is done periodically).
3. Reply NOTOK, causing SWAPPER to continue initialization using whatever memory it has already obtained.
4. Enter a new value for the CORESIZE parameter using the SW ODT message. SWAPPER attempts to obtain the new amount of memory.

CONTROLLER DISPLAY OF SWAPPER PARAMETERS

A sample of the new format for displaying SWAPPER parameters is the following:

```
SW ** RUNNING ** CORESIZE=90 SLOTS (89100 WORDS)
SW ACTUALCORESIZE=90 SLOTS (89100 WORDS) MINTIME=3 SECONDS
SW MAXSLICENR=7 RATIO=2 MAXCORE=50 SLOTS (49500 WORDS)
SW MAXIOSIZE=50 SLOTS (49500 WORDS)
SW MINCHUNKSIZE=50 SLOTS (49500 WORDS) EXPRESERVE=0 SLOTS (0 WORDS)
SW EXPMAXCORE=0 SLOTS (0 WORDS) EXPMAXTIME=0.75 SECONDS
SW PRIORITYBIAS=0 UTILIZATIONBIAS=6 IOBIAS=0 MEMORYBIAS=0
SW NOSWAPTRANSTATE ON SWAPPACK ON KSID
```

The initial item "** RUNNING **" is an indicator of the current status of SWAPPER; the values range from NOT RUNNING to INITIALIZING or SHUTTING DOWN.

"ACTUALCORESIZE" is an indicator of the amount of swapcore actually being used by SWAPPER (see "Changing the Swapspace Size via the CORESIZE Parameter" for semantics).

"NOSWAPTRANSTATE" indicates that the NOSWAPTRANSTATE option is set (see "SWAPTRANSTATE, NOSWAPTRANSTATE PARAMETERS" for semantics).

"ON SWAPPACK ON KSID" indicates that SWAPPER is using the SYSTEM/SWAPDISK files on the packs SWAPPACK and KSID on which to swap.

I/O ERROR HANDLING MECHANISM

SWAPPER has two distinct algorithms for dealing with I/O errors.

I/O Errors on Write Operations

After receiving an I/O error on a write operation, SWAPPER marks the area of swapdisk in question as bad and attempts to retry the write operation to a different area of swapdisk (possibly on a different family). The table of areas of bad swapdisk is stored in the SYSTEM/SWAPDISK file itself, and is referenced any time SWAPPER opens this file. To clear the table, remove and then re-create the swapdisk file.

I/O Errors on Read Operations

If an I/O error occurs on a read operation, the affected task is stuck on disk; it cannot be DSed. When this situation occurs, SWAPPER writes a message into the job log for the affected task and notifies the operator via an RSVP. If the operator wishes to retry the I/O operation (e.g., if a pack had gone not ready and was now on a different spindle), an appropriate reply to the RSVP would cause a retry. If the situation is not corrected (i.e., there is another I/O error), the above process is repeated.

D3251 MCP-GENERAL - "GETSTATUS/SETSTATUS" ENHANCEMENTS

The DCALGOL Intrinsic GETSTATUS has been enhanced for the Mark 32 system software release.

1. I/O Path Information

GETSTATUS now returns I/O path information for Multiplexor or MLIP I/O subsystems. General information and path-specific information are provided.

Parameters

GETSTATUS is invoked to obtain unit information in the usual manner:

B6000 SERIES MARK 32

GETSTATUS(TYPE,SUBCLASS,MASK,A), where

TYPE	TYPEF	= 1,
TYPE	SUBTYPEF	= 0, 1, or 2,
SUBCLASS		= 1,
MASK		= <info mask>
A		= <array>

In Mark 31 and earlier MCP releases, bit 31 of the mask would cause multiplexor/channel information to be packed into a single element of "A". This bit is still operative, but for B6800 and earlier systems only. Path information will not be provided on a B6900 via setting of bit 31.

For the Mark 32 release, bit 41 is recognized as the path information bit on any system. Instead of packing the information into a single word, the information is returned in the variable length portion of the "A" array; i.e., A[BASE+41+1] is now the index into A of the actual information.

Path Information Format

The first word, starting at A[A[BASE+41+1]], consists of general information.

FIELD	NAME	B6700, B6800	B6900
[47:4]	SYSTEMTYPEF	1	2
[43:8]	PCTYPEF	physical type of the unit.	DLP identity.
[35:8]	UNITTYPEF	same as PCTYPEF.	type of unit outboard the DLP.
[27:12]	BASEUNITF	N/A (always 0)	DLP base number.
[15:4]	RELUNITF	N/A (always 0).	relative number of the unit with respect to the DLP.
[11:8]	NUMPATHSF	total existing paths to the unit.	
[3:1]	OUTBOARDHOSTF	N/A (always 0).	when set, this bit serves as notification that one or more paths to the unit is via an outboard host (e.g., an LSP DLP outboard an NSP DLP). See HOSTDLPF, below.
[2:1]	FIRMWAREFLAGF	1 if firmware level info is available, 0 if not.	
[1:2]	* NOT USED *	always 0.	

The second and subsequent words contain information specific to each existing path, two words per path.

The first word of each path entry contains firmware level information. Note: if A[A[BASE+41+1]].FIRMWAREFLAGF is 0, then this word is 0.

FIELD	NAME	B6700, B6800	B6900
[47:12]	* NOT USED *	always 0.	
[35:4]	FWSIZEF	number of relevant hex digits in FIRMWAREF.	
[31:32]	FIRMWAREF	Controller (e.g., DPDC) level, right justified.	Controller (e.g., DPDC) or DLP firmware level, right justified.

The second word of each path entry contains path status information.

FIELD	NAME	B6700, B6800	B6900
[47:1]	PATHRESERVEDF	1 if path has been "UR"ed, 0 if not.	
[46:1]	PATHOFFLINEF	1 if path has been taken offline by the MCP, 0 if not.	
[45:1]	PATHASSIGNEDF	1 if path has been assigned to a stack, 0 if not.	
[44:6]	PATHIDF	an unique path identification value.	
[38:3]	* NOT USED *	always 0.	
[35:12]	HOSTDLPF	N/A (always 0.)	if OUTBOARDHOSTF is 1, then a non-zero entry in this field is the physical unit number of the outboard host DLP in control of the unit; a zero entry indicates that this particular path is directly connected to the host.
[23:4]	PROCNUMF	Processor number.	Processor number; If OUTBOARDHOSTF is 1 and HOSTDLPF is non-zero, then this field is zero (meaningless).
[19:4]	IOPORTNUMF	Multiplexor number.	MLIP port number. If OUTBOARDHOSTF is 1 and HOSTDLPF is non-zero, then this field is zero (meaningless).
[15:8]	* NOT USED *	always 0.	
[7:4]	IOPORTPORTF	Channel number.	LEM port number.
[3:4]	DLPNUMF	N/A (always 0.)	base-relative DLP number.

2. Unit Subtype Value

The DCALGOL Intrinsic GETSTATUS has been enhanced to return the unit subtype value, if the value exists.

Parameters

GETSTATUS is invoked to obtain unit information in the usual manner:

```
GETSTATUS(TYPE,SUBCLASS,MASK,A), where TYPE.TYPEF      = 1,
                                     TYPE.SUBTYPEF     = 0, 1, or 2,
                                     SUBCLASS           = 1,
                                     MASK               = <info mask>
                                     A                   = <array>
```

To obtain the unit subtype value, bit 42 of the MASK is turned on.

Unit Subtype Values

If the unit does not have a valid subtype, then bit 42 of the validity mask is reset. If a subtype exists, then A[BASE+42+1] will contain the subtype value. The current subtype values are:

MAGNETIC TAPE SUBTYPES

-
- 1: 7 TRACK
 - 2: 9 TRACK NRZ
 - 3: 9 TRACK PE
 - 4: 9 TRACK GCR

DATACOM SUBTYPES (B6900 and later systems only)

-
- 8: NSP DLP
 - 10: LSP (sub broadband)
-

3. ODT Message Cases

BOOTUNIT ODT Message

The BOOTUNIT message case in GETSTATUS/SETSTATUS handles the new BOOTUNIT ODT syntax.

GETSTATUS

Parameters

- 1. TYPE = MISCREQUEST
 - 2. SUBTYPE = 50
 - 3. ARRAYROW
- A[0] = number of entries + 1
- A[1]-A[A[0]-1]
- = each entry as follows:
- A[AI].[19:8] = PROCMASK (see SETSTATUS)
 - A[AI].[11:12] = unit number of disk type device for Halt/Load (see SETSTATUS)

SETSTATUS

BOOTUNIT Modify processor bootstraps

Parameters

- 1. TYPE = MISCREQUEST
 - 2. SUBTYPE = 46
 - 3. VALUE
 - 0 = if monolithic system
 - 1 = multiprocessor B6900 system
 - 4. ARRAYROW
- A[0] = number of entries + 1
- A[1]-A[A[0]-1]
- = each entry as follows:
- A[AI].[19:8] = PROCMASK of processors whose bootstraps should be modified
 - . [19:1] = Processor 7
 - . [18:1] = Processor 6
 - . .
 - . [12:1] = Processor 0
 - A[AI].[11:12] = unit number of disk type device from which the above processor should Halt/Load

ID ODT Message

The ID message case in SETSTATUS has been revised to handle new ID ODT syntax.

SETSTATUS

ID (Initialize datacom, set NSP prefix, set NSP firmware suffix, set NSP audit options, request NSP local memory dump, and terminate datacom)

Parameters

- 1. TYPE = 2
 - 2. SUBTYPE = 45
 - 3. VALUE
- [0:1] = 1 DCP number or NSP unit number is supplied
 - [1:1] = 1 NIF prefix is supplied
 - [2:1] = 1 Request NSP memory dump
 - [3:1] = 1 Terminate NSP
 - [4:1] = 1 NSP firmware suffix is supplied
 - [5:1] = 1 NSP audit options to set is supplied
 - [6:1] = 1 NSP audit options to reset is supplied
 - [23:8] Audit options to set

[31:8] Audit options to reset
 [47:16] NSP firmware suffix value

4. ARRAYROW
 A[0] = length of entry + 1
 A[1]
 [23:8] = DCP/NSP number
 [38:6] = Length of standard form NIF prefix (in words)
 A[2-(A[0]-1)]
 = Standardform NIF prefix

Example:

ID 2 TESTNDL ON PACK

```
-----
A[0]:=5
A[1]:=0 & 2[23:8] & 3[38:6];
REPLACE A[2] BY 48"10060207"8"TESTNDL",
           48"04"8"PACK";
```

RSLT:=SETSTATUS(2,45,3,A)

The following message will be issued warning that the Subtype 18 of the Miscellaneous Request category of SETSTATUS will be deimplemented on the Mark 34 system software release:

"START DATACOM CASE 18 WILL BE DE-IMPLEMENTED ON 34
 (USE CASE 45)"

OL <unit mnemonic> ODT Message

 With the new GETSTATUS I/O path information interface, the path display that results from the "OL" ODT command has been modified in SYSTEM/CONTROLLER. The information displayed depends on the system type.

B6800, B6700 SYSTEM/CONTROLLER Path Display Modification

 A new "PATHID" column precedes the "MPX" column.

B6900 SYSTEM/CONTROLLER Path Display Implementation

 Given the differences in the new hardware, the path information displayed by SYSTEM/CONTROLLER on a MLIP system is different from that displayed on the B6800 or B6700.

If the path to the unit is via an outboard host (e.g., NSP DLP), a "HOSTDLP" column will follow the "PATHID" column; the entries in this column will be the physical unit number(s) of the controlling DLP(s).

If there is no firmware level information, the "FIRMWARE" column is omitted from the display.

A new SETSTATUS case has been added for the LH ODT message.

Parameters

1. TYPE = 1
2. SUBTYPE = 21
3. VALUE - same as Case 16
4. ARRAYROW - same as Case 16 except:
 A[1]
 [47:1] same as Case 16
 [46:1] = 1 PATHID specified in [44:6]
 [45:1] not used
 [44:6] = PATHID
 [38:39] same as Case 16

A new case has been added for the UR ODT message.

Parameters

1. TYPE = 1
2. SUBTYPE = 20
3. VALUE - same as Case 4
4. ARRAYROW - same as Case 4 except:
 A[1]
 [47:1] same as Case 4
 [46:1] = 1 PATHID specified in [44:6]
 [45:1] not used
 [44:6] = PATHID
 [38:39] same as Case 4

D3252 MCP-GENERAL - SEMIDEPENDENT TASKS, "VISIBILITY" ATTRIBUTE

INTRODUCTION

A new category of task has been defined, called "semidependent." The properties of semidependent tasks permit them to be handled by the MCP with more flexibility than is possible with fully dependent tasks.

In order to take full advantage of semidependent tasks on B6800 multiprocessor (tightly-coupled or TC) systems, it has been necessary to modify the rules for allocating tasks to subsystems. To provide full control of this allocation, the semantics of the SUBSYSTEM attribute have been modified, and a new attribute, VISIBILITY, has been created.

DEFINITIONS

Dependency

An independent task is one which has no affiliation with any parent. Examples are WFL jobs, tasks initiated by ??RUN, tasks (such as an MCS or library) initiated by the MCP, or tasks initiated by the RUN statement in ALGOL or COBOL.

A dependent task is one which is generated at the behest of another and runs under the control of its parent. Examples are tasks initiated by the RUN or PROCESS statement in WFL or the CALL or PROCESS statement in ALGOL or COBOL.

A semidependent task is a dependent task which has no access to data of the parent, apart from the TASK variable itself. A task has access to data in its parent only via globals or parameters, so a task may be semidependent if it is external (separate code file, hence no shared globals), and if it is passed no parameters by reference or by name.

A task which has data visibility into its parent is not semidependent; such tasks are called fully dependent.

Subsystem

The word "subsystem" (lower-case) refers to part of a TC system: either a local processor, with its memory, or the * GLOBAL tm Memory.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

SUBSYSTEM is a task attribute which is used to control the assignment of jobs and tasks to subsystems. A SUBSYSTEM value is specified by the user as a name. SUBSYSTEM names are defined by the system operator, who lists the constituent subsystems: G for Global, l for local processor one, etc. (See 31 GENERAL note D2861 for a complete description of the SUBSYSTEM attribute.)

SUBSYSTEM will also be introduced below as one of the mnemonic values for the new VISIBILITY attribute; it makes the effect of VISIBILITY conditional upon the value of the SUBSYSTEM attribute.

MOTIVATION

The visibility rules for TC systems have required that any dependent task be able to see its parent, so a local task could spawn dependents only in its own local box. Similarly, the visibility rules for SWAPPER have required the parent to be in-core whenever a dependent is being executed. Both of these rules must be enforced for fully dependent tasks, but need not apply to semidependent tasks since there is no data visibility across the process boundary.

These rules were particularly onerous for jobs, since jobs run very little but occupy substantial memory. The MCP has put jobs in Global by default, so that the tasks spawned by a job could be freely assigned to local subsystems. This policy produced unnecessary congestion in Global memory, especially on minimal systems. The MCP has not put jobs into subspace, because a job would always have to be swapped in whenever an offspring task is resident in memory. Since external tasks of a job can always be semidependent, we can now run most jobs as local and/or swapped tasks without loss of generality; see below under "JOBS" for the exceptions.

Another example of increased flexibility through recognition of semidependency applies to an MCS like CANDE: A site can now run CANDE entirely in a local box, but still process offspring tasks anywhere in the system. See CANDE note D3249 for further discussion of this subject.

There have been two inadequacies of the SUBSYSTEM specification mechanism in the Mark 31 MCP; both have been addressed in the design changes.

1. There has been an element of inconsistency in interpreting a SUBSYSTEM specification that includes both local and Global components. The Global component was ignored for most tasks, while the local component(s) were ignored for jobs, databases and MCSs.
2. On a TC system with more than two processors, there has been no way to specify that a job, including all its tasks, was to run in two (or more) but not all of the processors on the system.

EXTERNAL CONTROLS

The external control of task location has been modified by adding a new task attribute, **VISIBILITY**, and modifying the **SUBSYSTEM** attribute. There has also been some relaxation in the rules for the **SUBSPACES** attribute. The changes have been made in such a way that the desired flexibility could be achieved without invalidating existing WFL jobs.

In general, the MCP can best manage its workload when it has many tasks to run and few constraints on their location. Subsystem control is typically needed when a few dedicated applications consume a major fraction of the resources. For example, a site manager may assign a large database or dedicated application to a particular subsystem. It is anticipated that relatively few jobs or tasks should need **SUBSYSTEM** specifications, and very few of those should need a **VISIBILITY** specification.

VISIBILITY Attribute

A new task attribute, **VISIBILITY**, has been provided. It is used to ensure that a stack is sufficiently Global to provide visibility from any expected offspring or client stacks. There are four **VISIBILITY** states, with the following mnemonic values and semantics:

- UNSPECIFIED** is the default. **UNSPECIFIED** is treated as **MINIMAL** in most situations, but it is treated as **SUBSYSTEM** for: databases, MCSs initiated by the datacom subsystem, and libraries initiated by the linker with **SHARED BY ALL** or **DONT CARE** specified.
- MINIMAL** means no visibility requirements are imposed; the stack will go as local as possible.
- SUBSYSTEM** causes the stack to be Global if the **SUBSYSTEM** specification contains more than one member, or is unspecified; the stack will be local if a single local processor is specified.
- GLOBAL** causes the stack to go into Global memory.

VISIBILITY may be specified in WFL at the job or task level, and may be specified in CANDE as a run-time but not a compile-time modifier. It may be set or interrogated as an integer/mnemonic task attribute in ALGOL or COBOL, but can be set for only an inactive **TASK**. It may not be specified for a job queue.

The **VISIBILITY** attribute is not inherited by an offspring task.

If a **VISIBILITY** specification is incompatible with the requirement that a fully dependent task must be able to see its parent, the specification is disregarded but a warning message is displayed at the start of the offspring task.

SUBSYSTEM Attribute

The following changes have been made to the semantics of the **SUBSYSTEM** attribute:

1. The **SUBSYSTEM** value is now inherited by offspring tasks. That is, if the parent task has a **SUBSYSTEM** specification and the offspring has none, the offspring is given the same **SUBSYSTEM** value as the parent. A task may be forced into Global memory by a **VISIBILITY** specification without affecting its **SUBSYSTEM** specification, which can be inherited by its offspring. A **SUBSYSTEM** specification on a job queue will propagate by default to all the tasks of any job run through that queue.
2. There is now a distinction between null and unspecified values; any **SUBSYSTEM** specification, including null, may be inherited.
3. An assignment of the reserved **SUBSYSTEM** name "SYSTEM" is treated as an assignment of a null value.
4. When a task or data base is being initiated with a **SUBSYSTEM** specification of (**GLOBAL**), the **SUBSYSTEM** is reset to a null value and the **VISIBILITY** attribute treated as though it were **GLOBAL**. This action is taken whether the **SUBSYSTEM** was named "GLOBAL" or was an operator-defined name with the same specification. No subsystem specifying Global-only will ever be inherited.

Formerly, the **SUBSYSTEM** attribute was not inherited, but any offspring of a local task or job was forced into the same subsystem as its parent. Thus a local **SUBSYSTEM** specification could be applied to a job or task, forcing all its offspring into that same subsystem.

SUBSPACES Attribute

Fully dependent offspring tasks of swaptasks must be swaptasks; this rule no longer applies to semidependent tasks. A semidependent swaptask may be swapped independently of its parent; they need not be resident at the same time, and need not occupy the same subsystem.

If **VISIBILITY** causes a task to run in **GLOBAL**, any **SUBSPACES** specification is disregarded.

Selection

Here is a summary of the revised rules for subsystem selection.

1. Any offspring task from a Global task/job, or a semidependent offspring task from any task/job, can go anywhere. A fully dependent task of a local task must go in the same subsystem.
2. By default, all jobs and tasks go into local memory, except:
 - a. MCSs initiated by the datacom subsystem
 - b. Libraries initiated by the linker with specification SHARED BYALL or DONT CARE
 - c. Database stacks
 - d. Jobs which contain global-file equation (see below)

Rule 2 is applied only if rule 1 has not forced the issue, and can be overridden in most cases by compiled-in SUBSYSTEM and VISIBILITY specifications.

The following table shows the effect of the VISIBILITY attribute in combination with the SUBSYSTEM attribute. For purposes of illustration, a three-processor system is assumed, with local subsystems 1, 2 and 3 and Global subsystem G. Typical SUBSYSTEM definitions are shown as they might be entered in an MS command. For each combination of VISIBILITY and SUBSYSTEM that might apply to a task, the table shows where the task will be initiated. (The "/" denotes "or".)

VISIBILITY Value	SUBSYSTEM Definition						
	---	1	1,2	1,2,3	G	G,2	G,1,3
MINIMAL	1/2/3	1	1/2	1/2/3	G	2	1/3
SUBSYSTEM	G	1	G	G	G	G	G
GLOBAL	G	G	G	G	G	G	G

JOBS

The most noticeable change accompanying the introduction of semidependent tasks is that in a TC system jobs are by default placed in local rather than Global memory. All external tasks which are RUN or PROCESSED by WFL are semidependent. Any parameters from WFL to the task are passed by value, including arrays or strings, so there is usually no visibility from the task back into the job. Thus it is usually immaterial what subsystem the job occupies; the offspring tasks can go anywhere. There are two exceptions which may be of concern to some users:

WFL Subroutines

A WFL subroutine may be PROCESSED to run asynchronously with the parent job. Such a subroutine is an internal process and can see any global variables in the job stack; therefore it is a fully dependent task. If the job resides in a local subsystem, the subroutine will be forced into the same subsystem. Since any external offspring tasks of the subroutine will be semidependent, it will seldom matter to the user where the subroutine itself runs. However, if it is desired that a job be able to spawn subroutines in more than one local subsystem, this can be accomplished by specifying VISIBILITY=GLOBAL for the job.

Global-File Equation

A WFL job can declare a file and then cause an offspring task to use that file by label equation of the form <intname>=<file id>. This mechanism amounts to a hidden by-reference parameter: the task can see a file declared in the job. To ensure visibility from every task, any job which may perform global-file equation is forced to run in Global memory, irrespective of its SUBSYSTEM and VISIBILITY attributes.

In the 31 PR1 and subsequent releases, WFL marks all code files which do not use global-file equation, so that they can safely be run in local. If a job is compiled by a 31 or earlier WFL and then run (after a CM) on a 31 PR1 or later MCP, the job will be unmarked and therefore will run in Global memory.

CRITICAL BLOCK

For any dependent task, there is a "critical block" which the parent task may not exit while the offspring task is still active. For a fully dependent task, the critical block is the one which contains the most local (highest address couple) of: the TASK declaration, the PROCEDURE declaration, and the declaration of any parameter which is passed by reference or by name.

For a semi-dependent task, the critical block is the one in which the TASK is declared. (There are no reference parameters, and the PROCEDURE is external and therefore not significant.)

The actual "parent" of a task is, by definition, the task whose process record (stack) contains the critical block of that task. The parent need not be the task that executes the task-initiation statement. The critical block of a semidependent task may now be more global than when that task was considered simply dependent; thus the parent may even be a different stack.

D3418 MCP-GENERAL - "GETSTATUS" WARNING MESSAGES

On the Mark 34 system software release, the UNITREQUEST routine of GETSTATUS will no longer support CASE 29 (GSFIRMWARE) nor CASE 31 (GSPATHS). Firmware information is now provided along with path information in the new CASE 41; until they are deimplemented, the former case values will be relatively expensive to invoke. See MCP-GENERAL note D3251 for a description of CASE 41.

The warnings for DCALGOL programs are as follows:

CASE 29

"ON 34 GETSTATUS UNITREQUEST CASE 29 WILL NOT BE SUPPORTED
(USE CASE 41)"

CASE 31

"ON 34 GETSTATUS UNITREQUEST CASE 31 WILL NOT BE SUPPORTED
(USE CASE 41)"

D3500 MCP-GENERAL - CHANGES TO "SYSTEMSTATUS" CALLS

Changes to the MCP required by modularization and the implementation of B6900 MLIP I/O have affected the Type 4 SYSTEMSTATUS call.

The SYSTEMSTATUS Type 1 call will now report the proper MCP name after a CM# and when different MCP names have been specified for different processors on a B6800 Multiprocessor system.

explained in the SYSTEMSTATUS Reference Manual, Form No. 5011786. The changes do affect existing programs, and the documentation should be examined carefully.

D3501 MCP-GENERAL - USAGE INFORMATION FOR "I/O" DEVICES

Several enhancements have been made to the collection of data on the usage of I/O devices. The changes are visible to the user primarily in the SYSTEMSTATUS Intrinsic, though the STATISTICALDATA word and the maintenance log entry have also been affected.

For each unit, information is now available on the number of physical reads and writes, the total bytes transferred, the total I/O time accumulated, and the number of read and write errors. For each MPX/MLIP, information is available on the number of reads and writes, the total bytes transferred, and the total I/O time for user tasks and non-user (invisible) stacks. These statistics are accumulated continuously from the time of the last Halt/Load.

The MPX/MLIP usage information has been added to the type 7 SYSTEMSTATUS call. The I/O device statistics are available individually in the type 4 specific unit call and as a group in a new type 13 call. The SYSTEMSTATUS interface for accessing this data and its format are explained in the SYSTEMSTATUS Reference Manual, Form No. 5011786.

The current number of physical reads and writes and number of read and write errors have been added to the maintenance log I/O error entry. For MLIP entries, they appear in words 34 and 35, and their format is described in MLIP note D3355. For MPX I/O error entries, a new type of unit dependent information entry has been created. The type field in the key word is 12 and it is followed by two words of I/O statistics for that device. Their format is:

Word 1	[47:24]	total physical reads since last H/L
	[23:24]	total physical writes since last H/L
Word 2	[47:24]	total I/O errors in reads since last H/L
	[23:24]	total I/O errors in writes since last H/L.

Improvements were made to the computation of the unit reliability factor, which is based on the device's error rate over the last 500 I/O operations. The reliability factor may be interrogated with the RF ODT input message. It is computed by subtracting the error rate over the last 500 I/Os from 100%.

When the reliability of a device is degrading, the MCP will issue messages to inform the operator when the reliability factor declines to 95%, 90%, and for every 10% decline thereafter.

If MCP option 6, DIAGNOSTICS, is set, then these messages will be displayed as an operator RSVP, and activity will not be allowed to continue until the operator acknowledges the message with a <mix>OK input. This provides an opportunity to save or clear the degrading unit.

The format of the STATISTICALDATA word which holds the information needed to compute a unit's reliability factor has been changed. The new format is:

[47:10]	error rate for the last 500 I/Os (multiplied by 1000).
[37:17]	number of errors since H/L or the last time the unit was made available (UA-ed).
[20:21]	number of I/Os since H/L. Note that if the number of I/Os exceeds $2^{21} - 1$ (2,097,151) the high order bits will be truncated.

The I/O usage and error statistics are based on counting "normal" I/O activity. Some special MCP I/O, maintenance I/O, and all error retry I/O is excluded. The pair of I/Os (seek initiated and data transfer) resulting from a conditional seek to pack on a MPX machine are treated as a single I/O operation. I/O test operations are counted as writes, which may cause a small amount of write activity to appear for read-only devices and some non-existent units. It is possible, though very unlikely, for some unit types to show more errors than I/Os. This is because the activity information is updated when the I/O request is finished, while the error counts are updated before that in the error handling routines. Only I/O errors (not I/O conditions) which will be logged are counted. An I/O condition is defined as the situation in which the I/O device or its controller encountered a problem, but was able to execute the I/O operation successfully. This event is reported to the system and logged, but no error handling is required.

D3641 MCP-GENERAL - COMPILE TIME OPTIONS

The compile-time options are set as follows in the MCP symbolic and the non-diagnostics MCP object file released to the field.

INTERNAL	RESET
LINEINFO	SET
MCP	SET
STATISTICSET	RESET
DIAGNOSTICS	RESET
READLOCK	RESET
SWAPTRACE	RESET
OVERHEADCHARGED	RESET
LOCKTRACE	RESET
B7700	RESET
ACTIVETIME	RESET
TAPEOP	RESET
PRESENCEBITCHARGED	RESET
USERDATATROUBLE	RESET
NODUMP	RESET
EXPERIMENTAL	RESET
MAKEHOST	RESET

The compile-time options are set as follows in the diagnostics MCP object file released to the field.

INTERNAL	SET
LINEINFO	SET
MCP	SET
STATISTICSET	RESET
DIAGNOSTICS	SET
READLOCK	SET
SWAPTRACE	SET
OVERHEADCHARGED	RESET
LOCKTRACE	SET
B7700	RESET
ACTIVETIME	RESET
TAPEOP	RESET
PRESENCEBITCHARGED	RESET
USERDATATROUBLE	RESET
NODUMP	RESET
EXPERIMENTAL	SET
MAKEHOST	SET

The DISKCHECK compile-time option has been deleted. DISKCHECK controlled two functions. The first function, which was to ensure that pack labels are never overwritten, has been deleted. The second function, which was to trap long and short I/Os, has been left intact under DIAGNOSTICS as was previously the case.

REVERSEPAPERTAPE has been changed to a run-time option and will function the same as the REVERSEPAPERTAPE compile-time option. The new run-time option is called NORVRSPAPERTAPE and is option # 33.

Any user who requires the NORVRSPAPERTAPE option to be set should notify Burroughs as soon as possible. If no users require that the option be set, the run-time option will be deleted on the Mark 34 system software release and all paper tape readers will be assumed to be capable of reverse operations.

A new column-2 compile-time option, INTERNAL, has been added, which is SET to make a diagnostics version of the MCP and RESET to make a non-diagnostics version.

B6000 SERIES MARK 32
DOCUMENT CHANGES NOTES (D NOTES)

MCP

D3056 MCP - IDLE PATTERNS IN PRINTER DUMP

The MCP often displays a recognizable pattern in the stack registers while waiting in an IDLE operator. A few patterns have been changed since the 31 system release.

This note provides an annotated list of all the idle patterns displayed by the MCP. (There are other instances of the IDLE operator, mostly very early in system initialization, that show no distinct pattern.)

For the most common cases, the B6700 and B6800 have different patterns. B6700 patterns are designed to show one, two or four letters in the A, B, X and Y registers taken as a 12-by-16 dot matrix; those letters are shown here.

B6800 patterns are shown as the hex value actually displayed in each register, except:

- (a) "Linenumber" means that the sequence number of the line in the MCP where the IDLE occurs is displayed in decimal.
- (b) "RCW" means that the return-control-word of the procedure that contains the IDLE is displayed in raw form.
- (c) Occasionally some variable data are displayed, as noted.

The Y register contents are unspecified unless the B register tag is 2.

B6700	B6800	Significance (and location)

I	A: 2 AAAA1111AAAA	Normal idle during system initialization. (PAWS)
	X: 2 A00A00A00A00	
	B: 2 BBBB1111BBBB	
	Y: 2 B00B00B00B00	
B	A: 2 A1111111111A	Normal idle. (PAWS)
	X: 2 A00A00A00A00	
	B: 2 B1111111111B	
	Y: 2 B00B00B00B00	
C M	A: 2 CECECECECECE	Changing MCP. (CHANGEMCP) "CM" in EBCDIC
	X: 2 linenumber	
	B: 0 C3D440404040 =	
D U M P	A: 2 DBDBDBDBDBDB	Normal idle during initiation and termination of dump. (LOITER of DUMPBOOTSTRAPPER)
	X: 2 linenumber	
	B: 3 RCW	
D U M P	A: 2 DBDBDBDBDBDB	Dump waiting for system I/O to finish. (GETRDS of DUMPBOOTSTRAPPER) "GETRDS" in EBCDIC
	X: 2 linenumber	
	B: 0 C7C5E3D9C4E2 =	
D U M P	A: 2 DDDDDDDDDDDD	Normal idle during dump. (DPPAUSE of TAPEDUMP)
	X: 2 linenumber	
	B: 3 RCW	
D E A D	A: 2 DEADDEADDEAD	System is hung, displaying "... PLEASE HALT LOAD". (DEFUNCT)
	X: 2 linenumber	
	B: 3 RCW	
D E A D	A: 2 DEADDEADEAD2	System is hung; some other processor is already in DEFUNCT. (DEFUNCT)
	X: 2 linenumber	
	B: 3 RCW	
D E A D	A: 2 DDEADEADEADD	System hung taking a dump. (SPOUTP OF TAPEDUMP)
	X: 2 linenumber	
	B: 3 RCW	

The following patterns occur only during the initialization of B6800 multiprocessor systems and never appear on the B6700.

B6000 SERIES MARK 32

B6800	Significance (and location)
A: 2 CCCCCCCCCC0 X: 2 linenumbr B: 2 addressee proc Y: 2 message	Lead processor waiting to send a message to a follower. (MAILACARD of SECONDARYINITIALIZE)
A: 2 CCCCCCCCCC1 X: 2 linenumbr B: 0 000000000000	Follower processor waiting for a message from the leader. (PICKUPMAIL of SECONDARYINITIALIZE)
A: 2 CCCCCCCCCC2 X: 2 linenumbr B: 2 addressee proc Y: 2 message	Lead processor waiting for follower to receive a message. (MAILACARD of SECONDARYINITIALIZE)
A: 2 CCCCCCCCCCF X: 2 linenumbr B: 0 D3D4C5D9C7C5 =	Lead processor waiting for followers to complete initialization. (MERGER) "LMERGE" in EBCDIC

The printer-dump program, which is invoked for system errors early in halt/load initialization, uses a single set of patterns for both the B6700 and B6800. The A and B registers are designed for hex display, while the X and Y registers are used as a bit matrix.

X&Y	Registers	Significance (and location)
M D	A: 2 DEDEDEDEDED1 X: 2 F424F0F88430 B: 2 C961D6404040 = Y: 2 F000F0F112C0	Initial idle in printer dump. (DOIOP of MEMDUMP) "I/O" in EBCDIC
M D	A: 2 DEDEDEDEDED1 X: 2 F424F0F88430 B: 2 dump address Y: 2 F000F0F112C0	Normal idle in printer dump. (DOIOP of MEMDUMP)
N O	A: 2 DEDEDEDEDED0 X: 2 F610F0788870 B: 2 D7D9C9D5E3D9 = Y: 2 F086F0E111E0	No printer is on line. (DUMPITP of MEMDUMP) "PRINTR" in EBCDIC
N R	A: 2 DEDEDEDEDED2 X: 2 F610F0788870 B: 2 D9C5C1C4E840 = Y: 2 F086F0E111E0	Printer went not ready. (DOIOP of MEMDUMP) "READY " IN EBCDIC
N P	A: 2 DEDEDEDEDED3 X: 2 F610F0F99960 B: 2 D7C1E3C84040 = Y: 2 F086F0F00000	No path to printer. (DOIOP of MEMDUMP) "PATH " in EBCDIC
FIN	A: 2 DEDEDEDEDEDF X: 2 F9980F0F610F B: 2 C6C9D5C9E2C8 = Y: 2 F0000F0F086F	Printer dump is finished. (DUMPITP of MEMDUMP) "FINISH" IN EBCDIC

D3089 MCP - MEMORY DUMP TAPE RECORD FORMAT

Memory dump tapes have been given a new format to simplify TAPEDUMP's write parity error recovery algorithm. The blocksize has been increased by one word and a block count is appended to the block. If TAPEDUMP encounters a parity error while writing a block, it continues rewriting the block until the write is successful. The occurrence of parity errors while reading dump tapes is a normal phenomenon.

DUMPANALYZER, when reading the tape, uses the record count value stored in each record to sort out copies and accepts only the last good record as valid data.

Since the MCP makes no attempt to erase badly written records, but simply repeats them, a few unrecoverable parity errors on a dump tape are a normal occurrence. The DUMPANALYZER suppresses all automatic retry and logging of parity errors on the input; instead, a simple retry algorithm designed for the particular writing mode used to create dump tapes is performed. It backs up and rereads only when some record(s) have been read with errors, and the sequence of record numbers indicates that a correct copy of the given record has not been found.

D3094 MCP - AUTOPRINT CAN RUN IN LOCAL MEMORY

For B6800 Multiprocessor systems, the MCP now forks Autoprint into the local memory of the processor which can see the printer. If a tape or disk file is PBed, the MCP forks Autoprint into global memory because a printer is not selected until the print file is examined for forms information or train requirements.

If, during the printing of a job's BD files, Autoprint must switch printers due to forms requirements or trainid requirements, Autoprint moves its buffers to * GLOBAL tm Memory and completes printing for that job. Autoprint then goes to EOT to allow BACKUPQUEUER to fork an Autoprint in the correct local memory.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

D3095 MCP - INTRINSICS IN LOCAL MEMORY

For B6800 Multiprocessor systems, the MCP establishes an intrinsics stack in each local memory that has visibility of the family which contains the intrinsic code file. Tasks are attached to the intrinsic stack of the memory in which they are running. Migrating SWAPPER tasks are attached to the intrinsic stack in * GLOBAL tm Memory.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

D3102 MCP - "BNA" "MCS"

The initial implementation of an MCS to manage BDLC stations is released.

D3122 MCP - IMPLIED CONCATENATIONS MADE EXPLICIT

All implied concatenations of string constants have been made explicit by adding the concatenation symbol "|". Also, all strings that were concatenated have been made the same character type and all quote characters have been represented by the DEFINE QUOTE (=48"7F") In addition, the appropriate compiler options have been set such that the above syntax errors have been made violations, except for the use of the quote character which has been made a warning.

D3141 MCP - DECODE ERROR SECTORS

1. The sector where the error occurred is now decoded into cylinder, head and sector for packs for Log Maintenance.
2. The length of the extended result descriptor read and logged has been increased from 20 words to 26 words.

D3228 MCP - PRINTER DUMP HARDWARE INTERRUPTS

Printer dump now has an interrupt routine for each system, which determines the system type and configuration. It does not reference MCP D0 variables.

D3281 MCP - ATTRIBUTE HANDLING

The following task attributes may now be set to "." without getting task attribute INCORRECT SYNTAX errors:

BDNAME
CHARGECODE
DESTNAME

D3282 MCP - VIRTUAL MEMORY SIZE STATISTICS

Additional data has been added to that already returned for SYSTEMSTATUS Case 2. Words 41 and 42 of each group reported give the virtual memory size for batch and swap jobs, respectively. The virtual memory size for batch tasks is the total number of words of overlay disk space actually in use. The virtual memory size for swap tasks is the sum of the total number of words of overlay disk space actually in use and the number of words of disk occupied by swapped-out swap tasks. Batch virtual memory size is calculated and reported for each box on the system. Swap virtual memory is reported only for the "global" box, as it is impossible to determine into which box a swap task will be returned on multiprocessor systems.

D3309 MCP - "ITINERARY" TASK ATTRIBUTE

A new task attribute, ITINERARY, has been implemented, which is read only to the user and contains a record of the task's foreign ancestry.

Syntax:

```
|<---- , ----|
----<hostname>----
```


Semantics:

The left most entry in the string is the hostname of the most recent foreign ancestor of the task, the next entry in the string is the hostname of the host where the next eldest foreign ancestor is, etc. The ITINERARY attribute is inherited from parent to sibling verbatim when the parent and offspring are running on the same host. When the parent and offspring are on different hosts, the sibling task will have an ITINERARY attribute which will contain the Hostname of the host where the parent is running as the left most entry in the string, followed by the contents of the parent's ITINERARY attribute. The ITINERARY attribute is NULL until the first time the task family ancestry crosses a HOST boundary (i.e., a parent starts a sibling at some host other than the host where the parent is running).

Example:

The following table shows the contents of the ITINERARY attribute for four related tasks. The relationship of the tasks is as follows:

Task A is started on host BLUE,
Task A starts task B on host YELLOW,
Task B starts task C also on host YELLOW,
Task C starts Task D on host RED.

Task	ITINERARY string of TASK
-----	-----
A	NULL (i.e. ".")
B	BLUE
C	BLUE
D	YELLOW,BLUE

D3379 MCP - SET LIBRARY FUNCTION

The routine to handle the SL (Set Library) ODT message has been implemented. It manages the table of "function names" and the library "function" stacks. (See GENERAL note D3356 for a description of the SL message.)

D3381 MCP - IMPROVEMENTS TO WORKING SET SHERIFF

Several improvements have been made to the operation of the Working Set Sheriff.

1. When initiated, all tasks receive a default overlay goal which is a function of their priority and the system overlay goal. It is computed as $((100 - \text{MIN}(90, \text{PRIORITY}) / 50) * \text{OLAYGOAL})$. This default overlay goal may be overridden with the OG ODT message. Previously a change to the system's overlay goal was not reflected in the default overlay goals of running tasks, and their overlay goal remained set to the value computed at BOT. Now the default overlay goal for all tasks will be recomputed whenever the system overlay goal is changed.

This makes the Sheriff more responsive to changing the setting of the system overlay goal. The effect is especially noticeable when the Sheriff is first activated, since the former overlay goal was zero, and all running tasks had a default overlay goal of zero. Thus the previous code would allow them to continue executing unaffected by the Sheriff.

2. Overlay goals are now assigned to several types of stacks that previously did not have them. Examples are the CONTROLLER, INTRINSICS, and DMSII data base stacks.
3. The operation of the Sheriff when AVAILMIN is greater than zero has been changed. For the purposes of the discussion that follows, AVAIL is the amount of available memory and MEMMIN is the minimum amount of available memory desired by the Sheriff. MEMMIN is computed as AVAILMIN percent of the amount of usable memory.

If $\text{AVAIL} > \text{MEMMIN}$, the Sheriff will start resuming any tasks it has suspended. All tasks that were suspended by the Sheriff must be resumed before new tasks will be started. This rule is unchanged from previous releases.

If $\text{AVAIL} < \text{MEMMIN}$, new tasks will not be started (unless FS-ed). This rule is unchanged from previous releases.

If $\text{AVAIL} < 0.5 * \text{MEMMIN}$, the Sheriff will start suspending tasks in order to increase the amount of available memory. On previous releases, this process started as soon as $\text{AVAIL} < \text{MEMMIN}$.

The effect of changing the rule for suspending tasks is to smooth the action of the Working Set Sheriff when AVAIL is fluctuating above and below MEMMIN. Since suspending tasks is a fairly drastic remedy, it should not begin until the refusal to accept new tasks has had a chance to work and has not been sufficient.

For this last change to be properly effective, AVAILMIN must be set to a value sufficiently high (5% to 10%) that MEMMIN has a reasonable value (10K to 20K words minimum).

4. When resuming tasks it has suspended, the Sheriff looks first for those with the highest priority. Among suspended tasks with equal priority, it will resume the oldest (longest running) first. Previously, if there was not sufficient available memory ($\text{AVAIL} - \text{MEMMIN} > \text{memory needed to restart task}$) to resume the oldest, highest priority task, then no task would be resumed. Now if any of the highest priority tasks will fit, the oldest of these will be resumed.

5. The MCP procedure which implements the Working Set Sheriff, WSSHERIFF, has been rewritten for greater clarity and efficiency.

D3386 MCP - NEW USERDATA ERROR CODE "(42)"

The error code 42 has been added to USERDATA error codes.

Add the following paragraph to the SOG Reference Manual, Volume 2 (Form No. 5001688) on Page 9-5-14, after Code 41:

"42 EUDLLGT A locator has been passed with a null (0) value in UDLENGTHF when the USERDATAAREBUILD function needs a value GEQ 1."

This new error detection will prevent a processor loop.

D3399 MCP - "CM" VS. DUPLICATED MCPs

The CM ODT message has been changed to simplify and accelerate the creation of duplicated MCPs. Formerly, when an MCP was to be duplicated (or triplicated), CM always created a brand new file for each family index involved. This is no longer required. Now, CM messages of the following forms:

CM <filename> . . .

or

CM <filename>/FMLYINX <nnn> . . .

are handled as follows:

1. CM locates the specified file using the full name given in the command.
2. CM then makes a list of the filenames that are required for each family index. If "<filename> FMLYINX<nnn>" is specified, it is used as one of the copies and copies are made on other family members as required. If "FMLYINX<nnn>" is not specified, CM appends the correct member for each family index.
3. CM then attempts to locate the required file for each member. If a file is found and the code files are "identical" (as determined by the compiler timestamp), that file is used for that member; otherwise, a new file is built for that member.
4. CM then copies or moves rows as required and continues with normal CM.

D3400 MCP - ANALYZE LIBRARY PARAMETER MISMATCHES

When a parameter mismatch is discovered during library invocation, additional information is now supplied to assist in finding the source of the mismatch.

D3409 MCP - "IAD" NOT SUPPORTED ON "B6900"

Installation Allocated Disk (IAD) will not be supported on new systems, starting with the B6900. To prevent its use the following changes have been made. When running on a new system, the MCP procedures WRITEHEADER, COPYINT, EXCHANGEINT, RESERVEANDRESTORE, and DISKPACKCONFIGURE no longer allow the creation or updating of IAD files, areas, or packs. The RC Operator Input Command displays "IAD ILLEGAL ON NEW MACHINE" if it is specified and rejects the request. The REServe Operator Input Command no longer defaults to IAD and, if "AS BADDISK" or "AS MAINT" is not specified, displays "AS EXPECTED" and rejects the request. The DCALGOL WRITE Diskheader intrinsic will not update an IAD header and returns an error value of 17. The DCALGOL COPY intrinsic returns an error value of -99 and will not move data to/from IAD areas. The ALGOL EXCHANGE intrinsic will not exchange the rows of two files if either one of them is IAD. The program SYSTEM/IADMAPPER has been modified to handle the new error value returned by the WRITE Diskheader intrinsic.

D3411 MCP - PRIORITY, "D1" INFORMATION

A word containing the full priority (bias, declared and fine) field for the task has been added to the end of the SYSTEMSTATUS general mix information.

A word containing the STACKINFO data and the RUNNINGCOUNT for the task's D1 stack has been added to the end of the SYSTEMSTATUS specific mix information.

Full documentation of the location and layout of these words is contained in the file SYSTEMSTATUS/DOCUMENT on the SYSTEMNOTES tape. Instructions for printing the file are contained in GENERAL note D3205.

D3425 MCP - DELETE "PORTS, SIGNALS"

Attributes and mnemonics for PORTS and SIGNALS have been removed. In addition, the PORTS mnemonic for the task attribute OPTION has been removed.

D3468 MCP - LOG NEW OPEN, CLOSE INFORMATION

A new word has been added to the open and close log records. This new word contains the parameter information for the open and close routines. Until the Mark 33 release, this new information will be mapped onto old values and stored in the "old" parameter information field of the log record. Mark 32 log records will contain both the old and new values.

The 9th word of the open log record has the following format:

```

VALID SUB FILE = [46:1] - TRUE IF THE SUBFILE FIELD IS VALID.
SUBFILE        = [39:8] - THE RELATIVE INTEGER SUBFILE INDEX.
TYPE OF OPEN   = [ 7:8]
                 1 -> OPEN WAIT
                 2 -> AVAILABLE
                 3 -> OFFER
                 4 -> RESIDENT
                 5 -> PRESENT
                 6 -> PBT REEL SWITCH OPEN
                 7 -> STACK ASSIGNED OPEN REVERSE

POSITION       = [15:8]
                 1 -> AT FRONT
                 2 -> AT END

MOTION         = [23:8]
                 1 -> MOVE IT
                 2 -> DON'T BOTHER

```

The 15th word of the close log record has the following format:

```

VALID SUB FILE = [46:1] TRUE IF THE SUBFILE FIELD IS VALID.
SUBFILE        = [39:8] THE RELATIVE INTEGER SUBFILE INDEX.
DISPOSITION    = [23:8]
                 1 -> REWIND
                 2 -> NOREWIND
                 3 -> SAVE
                 4 -> LOCK
                 5 -> PURGE
                 6 -> CRUNCH
                 7 -> HERE
                 8 -> BLOCKEXIT

ASSOCIATION     = [15:8]
                 1 -> RELEASE
                 2 -> RETAIN
                 3 -> RESERVE
                 4 -> DISABLE

CLOSE TYPE      = [ 7:8]
                 1 -> REGULAR CLOSE
                 2 -> REEL CLOSE
                 3 -> DONT WAIT

```

After the Mark 32 release, the fields which contained the open and close types (word 7, [15:8] of the open and close records) will be returned to system; the values contained in them cannot be guaranteed as being correct.

D3480 MCP - INTRINSIC MAPPING

MISSING TEXT

D3483 MCP - "DONT CARE" LIBRARIES

Libraries with a sharing type of DONT CARE have been implemented. Libraries of this class are treated the same as libraries of sharing type SHAREDYBLL, except when running on a B6800 Multiprocessor system. In that case, a copy of the library will be initiated in each memory subsystem (one in the local memory of each processor and one in global memory). Within each memory subsystem, the library will be shared by all tasks within that subsystem. Migratory swaptasks will use the library in global memory.

D3484 MCP - "USECATDEFAULT" VS. "DIAGNOSTICS"

The SPO option USECATDEFAULT is now available on both DIAGNOSTIC and non-DIAGNOSTIC MCPs.

D3485 MCP - "OPEN" FUNCTION VALUES

The values returned by the OPEN function (and the AVAILABLE attribute) now include the following:

38 = UNREACHABLE HOST
40 = ALREADY OPEN
42 = BADSUBFILE INDEX

D3487 MCP - "MCP" CODE FILE ROW SIZE = "504"

The MCP no longer has any code segments large enough to require that the MCP code file have a row size of 1008 disk segments.

The NEWP compiler and the BINDER will now construct an MCP code file with a row size of 504 segments, as is done for other code and symbol files.

D3502 MCP - SHRINK FROZEN LIBRARY'S STACK

When a library (except MCP) freezes, the stack is shrunk, returning the unused portions to the system. Before the library unfreezes, the stack is expanded.

D3516 MCP - LIBRARY FUNCTION NAMES

When linking to a library, the linker will now use the "functiontable" and the LIBACCESS and LIBFUNCTION attributes when deciding to which code file it should link.

D3529 MCP - "PARTNER" , "EXCEPTIONTASK" REMOVED

Both SYSTEMSTATUS and GETSTATUS attempt to report the mix number of a task's PARTNER and EXCEPTIONTASK as part of the information they return about a task.

There is no safe and efficient way to determine this information. Attempts to do so could cause faults in SYSTEMSTATUS, GETSTATUS, and other parts of the MCP.

Since this information is of minimal use, it has been eliminated from the data returned. A -1 is returned in the places where these items previously occurred.

D3536 MCP - "SYSTEMSTATUS, IOTRACE" FOR "MLIP"

The implementation of IOTRACE for MLIP machines is similar, but not identical, to that for multiplexor machines. A primary difference is that a logical rather than physical result descriptor word is returned. The format of the MLIP IOTRACE data is fully described in the SYSTEMSTATUS Reference Manual, Form No. 5011786.

D3548 MCP - "DBS" IN LOCAL MEMORY VS. NONEXCHANGED UNIT

If a Mark 31 or earlier data base has a SUBSYSTEM specified, and the processor selected from that subsystem has no path to the units containing the ACR codefile, any program attempting to open that data base will now get a DMOEN error #53.

D3550 MCP - ELIMINATE "DL" NETWORK

On the Mark 31 release, the NETWORK modifier of the DL ODT input message was used to specify the packname for the Host Services library codefile, the default file title for which was *SYSTEM/HOSTSERVICES. On the Mark 32 release, this mechanism has been eliminated. The SL ODT input message is now used to specify the file title and packname for the Host Services (and Network Services) library. The function name is BNASUPPORT. For details, see the description of the SL ODT message in GENERAL note D3356.

D3551 MCP - PASSWORD MANIPULATION

Two changes have been made in the mechanisms for changing passwords:

1. The ability to change password via the USERCODE task attribute has been eliminated in the Mark 32 MCP. (If T.USERCODE = "U", a statement to replace T.USERCODE by "U/P" was interpreted as a request to install "P" as the password in the USERDATAFILE entry for "U".) This obscure feature is not in any current documentation, and was usually invoked inadvertently. (For example, successive assignments of the same usercode/password resulted in the second assignment being interpreted as above, sometimes with unexpected results.)
2. Any password-change action must specify the old password as well as the new one. This affects a change in the specification for the PASSWORD statement in WFL; the PASSWORD command in CANDE already requires the old password. For example, if user U has password P, the following WFL job will change the password to Q:

JOB CP; USER=U/P; PASSWORD=P/Q; END JOB

On the Mark 31 MCP, the statement PASSWORD=Q is enough. On the Mark 32 MCP, this older form will be allowed but a warning message will be produced: "PASSWORD CHANGE ON THE 33 RELEASE WILL REQUIRE THE <OLDPW>/<NEWPW> FORM." On the Mark 33 MCP, the old form will be disallowed; any non-privileged program which uses the old form will be terminated with a "security violation".

Specifically, the USERDATA intrinsic is being changed: forms (a) and (f) are being eliminated from function (6), as described in SOG Reference Manual, Volume 2 (Form No. 5001688), Page 9-5-7. The other four forms, (b) through (e), remain available through USERDATA, CANDE and pre-29 WFL; forms (c) and (d) are unusable in WFL because of the "=" and "+" codes.

Password assignments through the MAKEUSER utility continue to function as specified; this mechanism uses USERDATA function (7) which remains unchanged.

D3564 MCP - "WFL" TASK FAULT ACROSS NETWORK

The WFL construct "ON TASKFAULT" is now supported across networks.

D3573 MCP - "PBIT" TIME ACCOUNTING

The processor time and count of the presence-bit operations requested by each task are now accounted for by the MCP. These values are available via the CONTROLLER TI ODT message and are also recorded in the job log. The times are separated into two categories: INITPBIT, P-bits on virgin data or code segments, and OTHERPBIT, P-bits on overlayed data or code segments. The cost of a presence bit operation is charged to the process that requested the operation, not to the process that owned that data. The cost of P-bits for tasks which are not logged (independent runners) are charged to the MCP stack when that task goes to EOJ. The accumulated cost of P-bits on currently-running tasks is available through the SYSTEMSTATUS interface (see MCP note D3576). The effect of the PRESENCEBITCHARGED MCP option is that the INITPBIT and OTHERPBIT times are added to the processor time. On the Mark 34 MCP, this option will be eliminated. Installations that use this feature need only modify their accounting programs to add in these two extra categories.

D3576 MCP - "PBIT" TIME ACCOUNTING

Four new words have been added to the specific mix information part of SYSTEMSTATUS cases 3 and 9. These words fall at the end of the entry and are located after the D1 stack STACKINFO word.

The new words are the following:

INITPBITTIME: The processor time used to process initial P-bits.
 INITPBITCOUNT: The number of initial P-bits processed.
 OTHERPBITTIME: The processor time used to process all other P-bits.
 OTHERPBITCOUNT: The number of other P-bits processed.

D3579 MCP - PROCESSKILL EVENT ERRORS

Event valued file attribute errors are now fatal. If an error occurs in Attgrabber while attempting to get an event attribute, the user is processkilled to prevent him from calling WAITP with a bad event reference.

D3583 MCP - DUMP ACROSS MIDNIGHT

The date may not be updated across midnight if the system is dumping during that time. Users concerned with the problem should set the OKTIMEANDDATE option (system option #24).

D3594 MCP - REPLACE LOGGING OF "ORGHOST" BY "ITINERARY"

Because of the replacement of the task attribute ORGHOSTNAME by the task attribute ITINERARY, the logging of the ORGHOSTNAME attribute in BOT and EOT log records has been replaced by the logging of ITINERARY, as follows:

BOT WORD	EOT WORD	CONTENTS
17	28	Link to the value of the ITINERARY chain. The length field in these words is in characters rather than words.

D3595 MCP - INTRINSIC TO SUPPORT LIBRARY MAP

Most intrinsic references from existing code files are now mapped to the appropriate entry point in the appropriate support library (GENERAL, BASIC, PLI). The references that are mapped are:

1. All BASIC support entry points
2. All PL/I support entry points
3. All mathematical procedures for ALGOL, FORTRAN and PL/I.

All other intrinsic references continue to cause linkage to SYSTEM/INTRINSICS. The support libraries (GENERALSUPPORT, BASICSUPPORT and PLISUPPORT) must be properly SLED via the new (on Mark 32) SL ODT message (described in GENERAL note D3356).

D3600 MCP - DELETE OLD INTRINSICS

PL/I was changed on the Mark 30 PR1 release so that the compiler references the MCP's ATTRIBSEARCHER; therefore, ATTRIBSEARCHER has been deleted from the PL/I intrinsics.

D3606 MCP - "OFFSET" AND "DELTA"

The OFFSET intrinsic has been rewritten to run much faster. It has also been changed to be consistent with DELTA and other pointer operations, by reporting single- and double-word pointers in units of eight-bit characters, instead of in 48-bit words.

The DELTA intrinsic is now much faster for the non-optimal cases. Its values remain unchanged for all proper uses.

The following section of this note provides functional definitions of OFFSET and DELTA. The next section calls attention to the changes in specifications. The final section provides some comparative timings.

DEFINITIONS

OFFSET(<pointer expression>) is an INTEGER function; its value is the character index of the pointer. Thus if P is a pointer at the Nth character in some array, OFFSET(P)=N (relative to zero). For EBCDIC, ASCII, BCL and HEX pointers, OFFSET is reported in units of 8-, 8-, 6- and 4-bit characters, respectively. For single- and double-word pointers, OFFSET is reported in 8-bit units. (Such pointers are produced by the POINTER(<array>,0) constructs and by coercions of arrays into pointers; they are transformed into character pointers by a skip (e.g. P:=P+1) or by update from any operation except REPLACE ... WORDS.)

DELTA(<pointer expression 1>,<pointer expression 2>) is equivalent to (OFFSET(<pointer expression 2>) - OFFSET(<pointer expression 1>)).

CHANGES IN SPECIFICATIONS

The new OFFSET differs from the former implementation by reporting eight-bit characters, rather than 48-bit words, for pointers which are single- or double-word descriptors. Consider the following example:

```

ARRAY A[0:14];
DOUBLE ARRAY DA[0:9];
POINTER P,Q;
INTEGER I;
P:=A[5];
Q:=P+1;
I:=OFFSET(P);           % old: I:=5      new: I:=30
I:=OFFSET(Q);           % both:          I:=31
I:=DELTA(A[0],P);       % both:          I:=30
I:=DELTA(P,Q);          % both:          I:=1
P:=DA[5];
REPLACE Q:P BY "XX";
I:=OFFSET(P);           % old: I:=10     new: I:=60
I:=OFFSET(Q);           % both:          I:=62
I:=DELTA(DA[0],P);      % both:          I:=60
I:=DELTA(P,Q);          % both:          I:=2

```

Because of the change in specification for OFFSET, the MCP provides a warning message if OFFSET is passed a word-mode pointer with non-zero index:

'OFFSET' NOW REPORTS 8-BIT CHARACTERS FOR WORD POINTERS:
SEE MCP NOTE D3606. @ <address>

The <address> is a line number or RCW of the first use of OFFSET with a word pointer; only one message is given per task.

The new specification of DELTA is based on the number of characters 'before' the pointer (OFFSET), while the old was based on the number of characters in the array 'after' the pointer. There is no difference in the result for proper uses of the function, as in the prior example. The following extension of that example shows two abuses of DELTA:

```

P:=A[10];
Q:=POINTER(A[10],4)+3;
I:=OFFSET(P);           % old: I:=10     new: I:=60
I:=OFFSET(Q);           % both:          I:=123
I:=DELTA(P,Q);          % old: I:=-27    new: I:=63
Q:=POINTER(DA[5]);
I:=OFFSET(Q);           % both:          I:=60
I:=DELTA(P,Q);          % old: I:=30     I:=0

```

In the first case, DELTA is nonsense because one pointer is 4-bit and the other is (by default) 8-bit. In the second case, DELTA is nonsense because the pointers are into different arrays. The new definition is consistent, in that DELTA(P,Q) always equals OFFSET(Q)-OFFSET(P), but the function is still intended only for use with two pointers with the same granularity, into the same array.

TIMING

Following are timing comparisons of several cases. The first column shows the ratio of processor time for the construct in the Mark 32 implementation to that in Mark 31. The second column shows the processor time for each construct in the Mark 32 implementation relative to the time for simple OFFSET. (Measurements on the B6800 and B6700 were consistent.)

OFFSET	unsegmented array	.10	1	
OFFSET	segmented array	.14	1.5	*
DELTA	like pointers, same segment	.99	1.1	
DELTA	one 8-bit, one word; same segment	.06	1.3	
DELTA	like pointers, different segment	.13	3.6	*

* The time for a segmented array case increases by about two microseconds for each segment between the beginning of the array and the pointer.

D3609 MCP - COMPLEX "WAIT" WITH ZERO TIME PARAMETER

The complex WAIT statement with a time parameter of zero does not evaluate the events as it did on the Mark 30 release. As a result, the wait always returns by the timeout of the time parameter. For example,

```

I:=WAIT((0,EV1)    ALGOL
or
I:=WAIT((0,EV1) GIVING I    COBOL

```

always returns a value of 1 in I.

D3617 MCP - COMPILE TIME OPTION "OVERHEADCHARGED"

As part of the effort to reduce the number of MCP compile-time options, the OVERHEADCHARGED option and its effects will be eliminated on the Mark 34 release.

When OVERHEADCHARGED was set, whatever task was currently active would be arbitrarily charged with the processor time required by the MCP to perform overhead activities requested by another processor. This made the task processor times highly variable and mix dependent, and resulted in tasks being charged for processor time that was completely unrelated to their activity.

On the Mark 33 release, compiling an MCP with the OVERHEADCHARGED compile-time option set will cause a warning message to appear in the compile listing.

D3618 MCP - "COBOL74" VS. "WORD" MODE FILES

When a COBOL program opens a file whose MAXRECSIZE and BLOCKSIZE are multiples of 6, the MODE of the file is changed to WORD and the MAXRECSIZE and BLOCKSIZE of the file are adjusted accordingly. On output files, this modification is externally visible in disk headers and tape labels.

Code files produced by the COBOL74 compiler now identify themselves to the MCP with a different language type. For these types of code files, no such modification of the attributes of the file are made.

Code files produced by the COBOL compiler continue to work as described above.

D3619 MCP - "SYSTEMSTATUS" TYPE "4" GENERAL UNIT REQUEST

When a SYSTEMSTATUS Type 4 Unit Information request is made with the V2 parameter > 255, it returns one word of information for each unit on the system. If V2 <= 255, detailed unit information for physical unit number V2 is returned.

On the Mark 33 release, the SYSTEMSTATUS Type 4 General Unit request with V2 > 255 will be de-implemented. The Type 4 call will then only return detailed unit information for the unit number specified in V2.

General Unit information is now provided by the Type 13 call, implemented in the Mark 32 release, which is described in the SYSTEMSTATUS Reference Manual, Form No. 5011786.

On the Mark 32 release, a warning message is given the first time a program calls SYSTEMSTATUS Type 4 with a V2 parameter > 255, as follows:

"WARNING - SYSTEMSTATUS TYPE 4, V2>255 DE-IMPLEMENTED ON 33"

De-implementation of the Type 4 General Unit Request will also cause elimination of the Type 0 General Information request, since Type 0 assumes availability of the Type 4 information. No special warning of the Type 0 de-implementation will be given, since any call on it will produce the warning for Type 4.

D3620 MCP - HOSTNAME IN HEADING

The hostname of the dumping system is now printed with the dump heading.

D3652 MCP - LIBRARIES VS. "??DS", CM , RECONFIGURE

Frozen permanent libraries will no longer be DSed by the ??DS primitive ODT message. They may still be DSed by supplying the mix number in the form "<mix no> DS". To avoid the need to DS permanent libraries prior to CM or RECONFIGURE, those libraries will not be accounted for in MIXCOUNT; thus, CM and RECONFIGURE are permitted with frozen permanent libraries (with no users) still in the mix.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

MCP

P2538 MCP - SPURIOUS "PACK IN USE" MESSAGE

On a B6800 Multiprocessor system, a spurious PK<nn> IN USE message would occasionally be generated, as well as a spurious SECTORS REQUIRED message. These problems have been corrected.

P2625 MCP - PRINTER DUMP LOOP

Occasionally, the raw printer dump program would go into a loop doing futile outputs to the printer, which never moves. The problem occurred only when the paper had been left aligned at the end-of-page line; this has been corrected.

P2765 MCP - SEEK LOST MESSAGE

When a seek has been timed out by CHECKFORHUNGIOS, a message to this effect is now displayed.

P2766 MCP - PRINT ENTIRE BUFFER IN "PROGRAMDUMP"

In the past, the PRINTBUFFER routine in PROGRAMDUMP would print only the amount of data specified by AREADESC in the IOCB. Now the entire buffer is dumped.

P2770 MCP - NON-MCS "DCWRITE"

DCWRITE now allows a stack not marked as an MCS to do the DCWRITE, providing the last user procedure is from a stack that is marked as an MCS.

P2773 MCP - HUNG LIBRARY MAINTENANCE AFTER "I/O" ERROR

If an I/O error occurred during the compare portion of a COPY&COMPARE involving tape, the Library Maintenance task might hang waiting on a event which would never be caused. This would hang the tape drives involved in the copy portion as well. This problem has been corrected.

P2790 MCP - "RESTORE" VS. "DS"

DSing a RESTORE of an IAD pack occasionally caused an INVALID OP in PACKMOUNT. This problem has been corrected.

P2792 MCP - HALTING "DCP 0" IN "SECONDARYINITIALIZE"

During the development of the Mark 31 MCP, a line of code was accidentally dropped from SECONDARYINITIALIZE. As a result, DCP 0 was not being halted during the Halt/Load sequence prior to running memory confidence. Were DCP 0 to alter memory during this period, it could have caused the Halt/Load to fail and have to be reinitiated. This potential problem is now eliminated.

P2856 MCP - "DISCSTATUS" VS. "BLASTUNIT"

Occasionally, an I/O error in DISCSTATUS would cause the I/O queue for a disk unit to be corrupted. This problem has been corrected.

P2859 MCP - "UNITSTATISTIC"

The STATISTICALDATA array (available through SYSTEMSTATUS) would often show the total of I/O's since the last Halt/Load to be decreasing. This problem has been corrected.

P2862 MCP - "JOBDESC" COMPLEMENT

COMPLEMENTOR now tests for valid lengths in JOB DESC records and faults with DIV BY 0 if a zero length is found, thus preventing possible loops.

P2865 MCP - "SYSTEMSTATUS" (11): "SWAPPER" PARAMETERS

The SYSTEMSTATUS type 11 call has been corrected to return the documented number of words.

P2877 MCP - DUMP MECHANISM

The dump reason "MEMORY TESTOR" has been changed to "FORCED MANUALLY". This notation is used for dumps forced either by the planting of an IRW to 0,14 at M[D0+3] or keying the following into the P register: MKST;NAMC 0,14;ZERO;ZERO;ENTR.

P2878 MCP - BAD MOMADDRESS

CONTROLCARD created a bad momaddress problem when being called with a DCALGOL parameter type that was invalid or incompatible with the request. This problem has been corrected.

P2892 MCP - "FORGETCHECK"

The efficiency of the procedure FORGETCHECK has been improved.

P2893 MCP - SINGLE BIT ERROR LOGGING

The MCP now handles the logging of single-bit memory errors more efficiently.

P2896 MCP - "COPY" VS. "FAMILYINDEX"

When copying a file from tape that had FAMILYINDEX set for one or more rows, Library Maintenance would often blow up. This problem has been corrected.

P2914 MCP - RECORD SEQUENCE

Library Maintenance puts a record number in the first word of every tape record, which is checked when reading from tapes. If a mismatch occurs, Library Maintenance now issues the following message:

MT<nnn> EXPECTED RECORD <nnnn> BUT WORD 1&2 OF RECORD=<xxxxxxxxxxxx>

P2919 MCP - SEND MESSAGE TO LOCAL "DBS"

Doing a <DBS stack number> SM for a DBS that is in a local memory on a B6800 Multiprocessor system no longer results in an invalid program word interrupt.

P2920 MCP - MEMORY MANAGEMENT

On a B6800 Multiprocessor system, GETSPACE would occasionally fault when the area at the bottom of a local memory was not a save area. This problem has been corrected.

P2921 MCP - SAVING MEMORY MODS

Aborting an attempt to save a mod of memory (by an RY of that mod) will no longer result in some chunks of that mod being marked "UNAVAILABLE".

P2924 MCP - PROCID VS. "M[MSGADDR+3]"

The MCP now displays the correct PROCID for General Control, Alarm, Hardware and SDI interrupts. The display has also been shortened, causing the P3 word to remain on the system ODT.

P2925 MCP - MEMORY MANAGEMENT

Small chunks of save memory, created when controlcards were entered at the ODT, are no longer left around in memory.

P2927 MCP - "7-TRACK" LIBRARY TAPES

Copying from a 7-track library tape no longer causes Library Maintenance to fail.

P2929 MCP - "RECONFIGURATION"

RECONFIGURATION now works properly when the Halt/Load unit is 5N disk.

P2958 MCP - "RESOURCECHECK"

A problem that caused 7-track tapes to be unusable when the option RESOURCECHECK was set has been corrected.

P2959 MCP - OVERLAY FILE CORRUPTION

A problem that could cause data corruption in the MCP overlay file on B6800 multiprocessor systems has been corrected.

P2961 MCP - INCORRECT FAMILY SUBSTITUTION

When finding a file, the MCP was indicating in the "NO FILE" message the family substitution of the process finding the file rather than the process owning the file. This problem has been corrected.

P2963 MCP - "SUPERPLUCK" HANGS/DUMPS VS. "SCHEDULED"

An error has been corrected that could cause a control-state loop or memory dump in the MCP procedure SUPERPLUCK when a SCHEDULED task was XSed.

P2994 MCP - PARAMETER MISMATCH

A parameter mismatch occurred when, in a FORTRAN program, a function or subroutine had an array name as a dummy parameter and it was called with an array element to an actual parameter. This problem has been corrected.

P3002 MCP - REMOVE "B6700 FINDMEMORYPARITY"

When a B6700 processor detects a memory parity error (and the stop-on-parity switches are off), the MCP no longer attempts to locate the word in error by accessing every word in memory. The logged error address will always be zero.

There has been no change for the B6800.

P3019 MCP - "NOT READY" "RSVP"

When an inuse disk goes not ready, DISKPACKER will now put out a "NOT READY" RSVP to alert the operator.

P3020 MCP - ATTRIBUTE HANDLING FOR DATA BASE

The appropriate routines for run-time data base attribute handling have been added to the MCP.

P3024 MCP - "MOVE" VS. DISK PACK TYPE "206"

Disk pack type 206 does not report correct unit type if it is not ready; therefore, UNITMOVER has to wait until the destination unit becomes ready before comparing unit densities. This problem has been corrected.

P3025 MCP - "DUMPANALYZER" RECOGNIZES FROZEN "MCP"

When the MCP froze as a library, DUMPANALYZER did not recognize the stack kind literal. The value array STACKKINDS now has the stack kind literal of "FROZEN MCP".

P3027 MCP - "STARTSYSTEM"

PRESERVER and STARTSYSTEM have been corrected so that a configuration file can be changed (via ACQUIRE/FREE) before the catalog family is found. Now the operator may ACQUIRE the unit that has the catalog, Halt/Load and have the system locate the catalog family.

P3028 MCP - CHECK FOR LIBRARY CAPABLE

The MCP never checked to ensure that a program it was initiating as a library was library-capable. This problem has been corrected.

P3029 MCP - CORRECT "FORGET CHECK" ON LIBRARY TEMPLATES

Programdump no longer prints out library templates that have never been used.

P3047 MCP - TIGHTLY-COUPLED MAIN MEMORY "DCP" CODE

On a B6800 multiprocessor (tightly-coupled) system, the DCP code area is allocated in the memory of the local processor to which the DCP is connected, rather than in global memory. This change reduces * GLOBAL tm Memory congestion, and reduces global memory contention when main memory DCP code is used instead of DCP local memory.

Note that DCP tables are not affected: they will be in either DCP local memory or in global main memory on a tightly-coupled system, which is to be avoided if possible. For optimum system performance, it is recommended that sufficient DCP local memory be available to contain both DCP code and tables.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

P3052 MCP - "PATHRES DS"

PATHRES can now be DSed out of the schedule queue.

DISCSTATUS now sets the UDECIMAL bit for BX380 disk packs.

P3063 MCP - DATA BASE EQUATION IMPLEMENTATION

MCP support has been added for data base equation via WFL.

P3064 MCP - "SWAPPER" HUNG VS. CONTROLCARD

On systems with a small swap space, SWAPPER would sometimes hang after a ZIP statement with the controlcard independent runner stuck in the schedule. This problem has been corrected.

P3065 MCP - JOB FILE ROLL OUT

When the CONTROLLER was starting a job, it was possible for it to hang on an RSVP waiting for disk segments. DOCTOR has been changed to issue the following error message when this occurs:

```
<mix #>PK<nnn> CANNOT ALLOCATE <sssss> SEGS FOR JOB FILE ROLL OUT
```

P3066 MCP - "DMSOPEN" CONTIGUOUS SAVE MEMORY REQUIREMENT

For large data bases, DMSOPEN could require large amounts of contiguous memory space; now, all large arrays are segmented and overlayable.

P3075 MCP - PRINTER DUMP TO DRUM PRINTER

The MCP can now take a printer dump to a drum printer.

P3096 MCP - "NO GO PAST" PROTECTION

All ACCESSROUTINES calls out of DMSOPEN and DMSCLOSE are now protected by NO GO PAST block to ensure that these DMS routines can clean up and finish their work.

P3097 MCP - SEPARATE HALT/LOAD PACKS

Use of separate pack families as the Halt/Load units for B6800 multiprocessor systems could result in a system hang. This problem has been corrected.

P3098 MCP - "DMSUPDATEDISKHEADER NOOP"

DMSUPDATEDISKHEADER is now a NOOP when calling for an uninitialized file.

P3100 MCP - "RESIZE" REORGANIZATION

RESIZEANDDEALLOCATE has been reorganized to eliminate a timing problem that could arise with SWAPPER with resultant BAD MOM ADDRESSF and DIALED GEORGE dumps.

P3112 MCP - "DMSII" EXCEPTION CATEGORIES

The DMSII exception categories FATALERROR and INTEGRITYERROR are now recognized.

P3115 MCP - "FILECARDS" ATTRIBUTE "FORGETCHECK"

A problem that caused a dump by FORGETCHECK when a program got a syntax error using the FILECARDS task attribute has been corrected.

P3116 MCP - "FORGETCHECK" AFTER MEMORY EXCEEDED

Sometimes a task running in swapspace could get a FORGETCHECK dump if it were DSed for "MEMORY EXCEEDED" and requested a programdump with arrays. This problem has been corrected.

P3117 MCP - "SWAPPER," STACK STRETCH

If a process running in swapspace were to be DSed for "MEMORY EXCEEDED" while in the process of having its stack stretched, the system would loop. This problem has been corrected; the system no longer loops.

P3127 MCP - "TC" OVERLAY FILE CORRUPTION

A situation in which overlayable data could be corrupted has been corrected by revising the mechanism for setting up overlay files. (The problem occurred only in B6800 Multiprocessor (TC) systems with multiple SWAPPERS, and was triggered by running CONTROLCARD as a migratory swaptask, as in ZIP from a swaptask.)

One change is that a fully-dependent offspring task in the same memory subsystem as its parent will share the overlay file of its parent. An independent overlay file is created for a semi-dependent task (see MCP-GENERAL note D3252), a task in a different box from the parent, or a swaptask with a non-swap parent.

P3128 MCP - COPY&COMPARE VS. REEL SWITCH

COPY&COMPARE would get a directory compare error if reel switch occurred in two destination tapes at the same time. This problem has been corrected.

P3129 MCP - "STATISTICS"

SCR may now be run on an MCP with the compile-time option STATISTICS set and with statistics enabled.

P3130 MCP - SCHEDULING ON "B6800" MULTIPROCESSOR SYSTEMS

The scheduling algorithm for B6800 Multiprocessor systems has been improved to avoid a bias towards processor #1.

P3131 MCP - "DL SUMLOG"

Non-privileged users running SYSTEM/LOGANALYZER will now find the SYSTEM/SUMLOG when the log has been DLed to a family other than disk.

P3132 MCP - SEPARATE HALT/LOAD FAMILIES

On a B6800 Multiprocessor system using different pack families as Halt/Load units, a dump by UNOWNED LIBERATE could occur if a program opened and closed the MCP code file of the follower processor. This no longer occurs.

P3133 MCP - LIBRARY MAINTENANCE "IOCB"

On B6800 Multiprocessor systems, the IOCB areas for the Library Maintenance file buffers had been located in local memory, causing unnecessary copying of the IOCBs to * GLOBAL tm Memory. This has been corrected.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

P3134 MCP - "INTRINSICINFO"

Two long-standing problems involving INTRINSICINFO have been corrected.

INTRINSICINFO is an intrinsic ARRAY in the ALGOL language; it is used by compilers to access a description of installation intrinsics. Because of this change, any code file compiled with pre-30 ALGOL will fault if it attempts to access INTRINSICINFO; therefore, the 29 ALGOL compiler can no longer be used with \$SET INSTALLATION.

One of the corrected problems involved possible dangling references after a Change Intrinsics operation. The other involved unnecessary extra presencbit interrupts through an absent copy (in the MCP) of a present descriptor (in the intrinsics).

P3135 MCP - "CHECKPOINTED" SWAPTASK

A swaptask may call CHECKPOINT and subsequently be RESTARTed, but will no longer be a swaptask. A bookkeeping error has been corrected, so such restarted tasks have their event-wait linkages handled properly.

P3136 MCP - FORGOTTEN "PIB"

Since INITIATE can now forget a PIB and return an error, callers of DOCTOR must compensate for the possibility of an already-forgotten PIB.

P3138 MCP - READY QUEUE TIME

An error has been corrected in the computation of time spent by a task in the ready queue.

The effect of the error was to charge some tasks with ready time when they were actually waiting. The error was significant only in a lightly-loaded system.

An important effect of the error was felt on B6800 Multiprocessor systems as a biasing of the processor selection algorithm against a lightly-loaded processor.

P3148 MCP - MEMORY DUMPS ON A SHARED RESOURCE SYSTEM

The TAPEDUMP procedure will no longer attempt to read the labels of tapes belonging to other members of a Shared Resource system, thus avoiding the erroneous rewind of in-use tapes. Also, the only time a tape of serial number "DUMMMP" will now be automatically used is when it is mounted on a drive that had been owned or acquired by the dumping system.

P3150 MCP - "ACTIVETIME"

A syntax error that prevented compilation of the MCP with ACTIVETIME set has been corrected.

P3173 MCP - PRINTING OF INUSE CODE SEGMENTS

A change to PROGRAMDUMP, which inadvertently prevented the printing of inuse code segments, has been corrected.

P3179 MCP - GRAPH FOR DATA BASE USERS

A separate graph has been added for users of data bases.

P3196 MCP - "ZOT" LIBRARY TEMPLATE MARKER IN "BLOCKEXIT"

An INVALID OP occurred because DISCONNECTLIB was being called with an empty descriptor. This case is now handled correctly.

P3197 MCP - "IPC" SWAPJOBS VS. SUBSPACE GROWTH

If an IPC swaptask were to increase its subspace size to a point where it would no longer fit in swapspace, the MCP sometimes would not DS the program for memory exceeded. This problem has been corrected.

P3203 MCP - EXTERNAL BY CALLING LIBRARY PROCEDURE

When external procedures were processed off and then became libraries, problems would arise in the LIBUSEMCP. If the library were DSed, the header index node of the map was never removed. This has been corrected.

P3206 MCP - MAKE "EBCDICTOWORD" INLINE

The procedure EBCDICTOWORD is now an inline procedure.

P3219 MCP - WORKING SETS

Use of working sets on Mark 31 could occasionally result in corruption of the low order 20 bits of a word in a stack. This problem has been corrected.

P3222 MCP - CORRECT "PBIT" OF ZERO LENGTH DOPE VECTOR

An error has been corrected that could sometimes cause memory links to be corrupted when a program declared and touched a zero-length dope vector. The symptoms include a possible dump by BAD FORGETSPACE, if DIAGNOSTICS is set, and a control-state loop in GETSPACE. The problem arose when a program declared a two-dimension array with dynamic bounds, so that the upper bound of the first dimension was one less than the lower. (In general, the problem could occur with any but the last dimension of any multi-dimension array.) Because of hardware differences, the problem appears on B6800 but not B6700 systems.

P3223 MCP - DEFUNCT IN "DUMPBOOTSTRAPPER"

DUMPBOOTSTRAPPER no longer gets an INVALID INDEX when recovering from a dump on a follower processor on a B6800 multiprocessor system if the dump occurs after CMing from an MCP with a different memdumpdisk size (i.e., 31.0 to 31.1 or 30.0 to 31.1).

P3224 MCP - ATTRIBUTE GRABBER FAULT

The ATTRIBUTEGRABBER interface for pointer-valued FILE attributes from pre-Mark 30 compilers has been corrected to avoid confusion in cases that the top-of-stack item is not a pointer.

P3227 MCP - "PAST" ORDER

The order of entries in PAST blocks, which was inadvertently changed, has been restored to its original ordering.

P3228 MCP - RESUMED "ST" TASKS HAVE EXCESS OVERLAY GOAL

When a task is STed, its overlay goal is arbitrarily increased while the task is suspended.

If the task had ever had its priority or overlay goal set by the operator, this temporary high overlay goal would be retained when the task was resumed. Its previous overlay goal will now be restored properly. If the operator explicitly sets the overlay goal while the task is suspended, the new value will be retained.

Also, only the first priority change by the operator would automatically adjust the task's default overlay goal. Now it will be adjusted for subsequent changes as well.

P3233 MCP - "UNIT 0"

Systems with mag tape units configured to units 1-7 would get an INVALID INDEX when a VOLUME ADD was done for tape volumes. This problem has been corrected.

P3238 MCP - CARD READER ERROR RECOVERY

Under some conditions, it was possible to cause the MCP to fault after a READ-CHECK on a card-reader file with a very short read. This problem no longer occurs.

P3239 MCP - COPYING TOO MANY FILES

Copying too many files on tape will now cause Library Maintenance to display an error message. Previously, Library Maintenance failed with an INVALID INDEX.

P3240 MCP - USERCODE ON 14-LEVEL FILE NAMES

If there are already fourteen levels in a file title, it is impossible to append a usercode to the title. The MCP now detects two instances of this error that were previously undetected.

P3255 MCP - LOGGING INTERNAL FILE NAME

IOERROR no longer fails to log the internal file name properly.

P3257 MCP - INCREASE MAXIMUM "BDNUMBER"

The 3-digit number of the <modified file name> part of a backup disk file name now goes from 000-999 as documented. Previously, the number only went from 000-255.

P3264 MCP - LOGGING ROW INDEX

The row index was not logged properly for disk or pack file I/O errors. Now it is logged correctly for not only disk or pack files, but also backup files on disk or pack.

P3277 MCP - TASKS SUSPENDED BY "WSSHERIFF"

When the Working Set Sheriff was turned off (OLAYGOAL set to zero) with tasks it had suspended still STed, it could take a long amount of time for them to be resumed, forcing the operator to OK them. This was caused by the Sheriff continuing to check for available memory greater than AVAILMIN percent of total memory before resuming tasks. Now, when the Sheriff is turned off, it will resume tasks it has suspended regardless of the amount of available memory.

P3279 MCP - "BACKUPQUEUER"

The BACKUPQUEUER function of the AUTOBACKUP module was incorrectly updating the parameter indicating a CONTROLLER call. This can result in the CONTROLLER hanging on train printer I/O or an unwanted fork of BACKUPQUEUER to fire up AUTOPRINT. This no longer occurs.

P3280 MCP - AVOID BAD SEARCH FROM "STACKSTRETCH"

On B6800 multiprocessor (tightly-coupled) systems, an erroneous STACKSEARCH operation was being performed in connection with STACKSTRETCH. The problem has been corrected. The symptom was a copy descriptor with a bad address.

P3281 MCP - PROGRAMMED OPERATOR INTERRUPT

During the revision of hardware interrupt routines, the interrupt Programmed Operator Interrupt was omitted, causing an INVALID INTERRUPT instead of the PROG OP. This problem has been corrected.

P3282 MCP - PASS CONTROL FILE PACK NAME TO "ACR"

The pack name of the control file was being incorrectly passed to the ACCESSROUTINES. This no longer occurs.

P3283 MCP - "AUTOPRINT" WITH FMed PRINTER

When AUTOPRINT was directed to a particular printer by the operator input "FM LP <nn>", after the file requiring the FM has been printed, any immediately following files which did not require the special forms would be printed on that printer also.

This problem has been corrected; now, the FMed printer will be saved as soon as the first file has been printed.

A side effect of this change is that if any printer in use by AUTOPRINT is saved, AUTOPRINT will release it as soon as the current file has been printed.

P3301 MCP - EXCESS WORKING SET SHERIFF OVERHEAD

On a B6800 multiprocessor (tightly-coupled) system, the Working Set Sheriff was attempting to overlay the MCP stack's overlay amount of memory in every box, rather than only in the global box where the MCP stack is located. This caused the Sheriffs running in the other boxes to look at many more memory links than necessary, trying to overlay this amount of memory. The problem has been corrected. Now, only the Sheriff running in the global box will look for memory belonging to the MCP stack.

P3319 MCP - "INV OP" IN ATTRIBUTE HANDLER

With a data base stack larger than 8K words, the procedure SIRWOFADDRESS would return bad results. This would cause an INV-OP in attribute handler.

P3321 MCP - BOUNCE DUMP

Except at Halt/Load time, BAD DIR MEMDUMP's will now be PROGRAMDUMP's.

P3335 MCP - HANDLE NULL SUBSYSTEM IN UNRAVEL

Unravel will not cause a "FAULT ON BAD TASK ATTRIBUTE" if the SUBSYSTEM attribute is null.

P3336 MCP - FAULT BECAUSE OF MISSING INTRINSIC

Attempting to use a missing installation intrinsic will no longer cause a fault in the MCP procedure LOSEOLAYSPACE.

P3338 MCP - ANALYZE LIBRARY TEMPLATE

Prints out key information stored in the library template when taking a program dump with the option ARRAYS set.

P3339 MCP - "TAPESEARCH"

When copying from tape with files in order:

A/B, X/Y, A/C, X/Z

using "COPY A/=, X/=, ..."

TAPESEARCH would get fault. This problem has been corrected.

P3355 MCP - PROGRAM MARKED AS SWAPJOB

If a codefile had SUBSPACES=0 specified as a compile-time attribute specification and it was initiated with a task variable which previously had TSK.SUBSPACES:=2 (this will happen for a CANDE task if the CANDE option SWAPALL is not set), it would cause the system to go into a lock loop only if SWAPPER were not running. This problem has been corrected.

P3356 MCP - "QT PB MT"

When a tapeparity error occurs while AUTOBACKUP is printing a PB tape, AUTOBACKUP will now print a "QTED" ending banner.

P3358 MCP - "LOADALABEL" VS "MULTIFILE"

PARITY on POSITIONING open error for multifile GCR tapes has been corrected.

P3375 MCP - "GETSTATUS FORGETCHECK"

Programs calling GETSTATUS to get an MCS number will no longer cause dumps by FORGETCHECK.

P3376 MCP - "LOADCONTROL" TO TAPE

LOADCONTROL to tape will no longer cause dumps by "ILLEGAL I/O" on a B6800 multiprocessor system.

P3448 MCP - CHECKPOINT/RESTART FOR PROGRAMS USING STRINGS

The string pool of a program is now saved in the CP file and correctly restored.

P3449 MCP - RESTART OF SERIAL DISK FILES NEAR "EOF"

Restart will now compute correctly the last row for serial disk files restarted near the EOF. Previously, the restart would abort.

P3450 MCP - RERUN OF "COBOL" FILES WITH USE ROUTINES

Rerun of a COBOL program with use routines for a not-opened file now works properly. An SIRW unchanged by RESTORER, with unpredictable results, has been corrected.

P3451 MCP - MULTIPLE WAIT

A program that waits on time and an element of a large, untouched event array will no longer hang the system when run in a heavy mix.

P3452 MCP - "STACKLIMIT" TASK ATTRIBUTE

The STACKLIMIT attribute may no longer exceed the value of 65000.

P3453 MCP - CORRECT TIME SLICE CALCULATION

Time slices for interactive tasks are now properly calculated.

P3454 MCP - ALLOW HALT/LOAD AFTER POWER UP

The MCP will now Halt/Load after a power up.

P3455 MCP - "BOUNCE" MESSAGE

BOUNCE will now issue FLATREADER error messages more often. Previously, at most four messages were issued; now, up to five messages per row are allowed. Previously, a message was issued only if all backup copies failed; now, a message will be issued for every failure, even if the failure is subsequently corrected by reading from a backup directory.

P3456 MCP - "SYSTEMSTATUS" VS. "UNITMOVER"

After a pack has been moved, SYSTEMSTATUS now properly locates the unit.

P3457 MCP - CALCULATION OF CODE CORE ESTIMATE

The MCP now correctly calculates the estimated core requirements of a new task.

P3458 MCP - "FA" SWAPJOB

Sometimes a swapjob would issue a "NO FILE" RSVP after a "REQUIRES PACK" RSVP and not respond to the reply "FA". This problem has been corrected.

P3461 MCP - CHECKPOINT OF LARGE SIZE STACKS

Checkpoint will now verify that the stack can be written in a row of the CP file. If the stack is too large, the checkpoint will be aborted with message #30: "ROW SIZE OF CP FILE TOO SMALL".

The row size for CP files is now 1008 segments, thus allowing the restart of stacks of less than 22600 words.

P3478 MCP - IMPROVE RUN TIME PARAMETER CHECKING

Run-time parameter checking failed to catch mismatches in the parameters of procedures which are themselves formal parameters. This allowed "external" procedures to be invoked with totally unacceptable parameter types.

Now, procedure parameters will have their own parameters fully checked.

P3479 MCP - MULTIPLE FLAGREADERS

The MCP no longer gets a dump by "BAD DIR RECORDS" when closing several packs at once.

P3480 MCP - HALT/LOAD MEMORY CONFIGURATION

The minimum configuration to halt/load the MCP is 5 "mods" of memory (where a "mod" is 16384 words). Mods 0 and 1 must be present, plus any three others.

The Mark 31.242 MCP would not halt/load without mods 0-3 available; the Mark 31.280 MCP would not halt/load without mods 0-4 available. These restrictions were unintentional; the errors have been corrected.

P3482 MCP - DYNAMIC "EBIT"

The EBIT of the MSCW on future machines may be moved to bit 18. A variable will tell where the EBIT is. GETITGOING has its own EBIT variable also.

P3496 MCP - "LOCKTRACE" OPTION

The MCP diagnostic option LOCKTRACE (Option 30) now works correctly.

P3519 MCP - REPORT ON EXCLUSIVE FILES

FILEDATA and "PD" entered from the ODT did not report the existence of files which were in use Exclusive; thus, large files could be effectively hidden, and if the file were IAD, it might be assumed that the space were available and could be relocated. This no longer occurs.

P3521 MCP - "REELSWITCH" VS. DENSITY

OUTPUT REELSWITCH will now request the same density as the previous reel.

P3522 MCP - STACKSWAP VS. STACKSTRETCHER

CANDE was taking dumps by "BAD PBIT" because CANDE worker stacks were being stackstretched and copy descriptors were not being updated. This no longer occurs.

P3523 MCP - DESTNAME ON "ACR" CODEFILES

Setting Destname on an ACR codefile will no longer hang the system.

P3524 MCP - RESERVEDISK AND USERDATA HEADER

Reservedisk will no longer occasionally cause the header for the userdatafile to be removed from memory.

P3527 MCP - DATA BASE "TITLE" ATTRIBUTE VERIFIED

DMSATTRIBUTEHANDLER now verifies that data base names assigned to the data base TITLE attribute are syntactically valid. In addition, if the data base name does not include an ON <familyname> specification, ON DISK is added to the name.

P3528 MCP - UNITS EQUAL CHARACTER VS. BACKUP

Backup files were being created with the units field of the tape header set to characters instead of words, causing only 300 characters of each block to be read instead of 300 words. This no longer occurs.

P3529 MCP - PASSWORD HANDLING

An error has been corrected in password handling. The most common symptom was a SECURITY VIOLATION resulting from an assignment like the following, where P specified the same usercode that the task was already using:

REPLACE MYSELF.USERCODE BY P

It would also fail in a direct call of USERDATA function 6, under some circumstances.

P3530 MCP - "CONTROLCARD(QUEUE,7)"

An error has been corrected so that the DCALGOL construct CONTROLCARD(Q,7) will no longer produce an INACTIVE QUEUE fault when the queue is proper.

P3531 MCP - STACK OVERFLOW HANDLING

Programs that get a stackoverflow in the format intrinsics while handling a fault in the intrinsic will no longer cause the stack to be corrupted with a misplaced RCW. In addition, these programs will get R-Dsed if their stacklimit is exceeded.

P3568 MCP - PROGRAMS USING MARK "31" PORTS AND SIGNALS

Programs using the Mark 31 version of Ports and Signals are no longer supported. The HOSTINFO function has also been deimplemented. Programs using these features are now terminated.

P3569 MCP - CLOSE PORT VALUES

Errors reported by the port procedures from close of the fibstack are now reported to the user.

P3570 MCP - "AVAILABLE" TYPE OF FILE OPEN

AVAILABLE (and RESIDENT) types of file opens now work properly if NEWFILE=FALSE or both NEWFILE was not set and AREASIZE=0, and in either case, if the explicit or computed MYUSE=OUT. For example, an AVAILABLE open with NEWFILE=FALSE, MYUSE=OUT will not wait on an exclusive file (MYUSE could be set to OUT if the file were previously implicitly opened with a WRITE statement).

P3575 MCP - FORGETCHECK AFTER PROGRAMDUMP

A forgetcheck no longer occurs when taking a programdump while closing a data base.

P3576 MCP - "INVALID OP" IN "DMSCAUSE"

A stack doing a DMS WAIT with the MAXWAIT option specified no longer causes an INVALID OP in the stack doing the corresponding DMSCAUSE.

P3577 MCP - "SWAPPER" VS. "SIB"

SWAPPER now properly relocates a SIB if one of the MSCWs in the SIB falls on a pseudo-stack boundary.

P3578 MCP - SOFTWARE INTERRUPT HANDLING

Stacks that have been erroneously marked as STed will no longer hang the system if they use software interrupts.

P3579 MCP - "JOBDESC" VS. NONEXCHANGED UNITS

A JOBDESC file on a non-exchanged unit no longer causes a dump by ILLEGAL I/O.

P3585 MCP - "NOT READY" MESSAGES

UNIT NOT READY messages will now be printed in the job log.

P3607 MCP - TAPE VERIFY

When doing a tape verify, the heading line with the MCP level will now be printed correctly, and the system serial number will be printed with 4 digits.

Also, a problem using "VIA MPX <n>" on a B6800 Multiprocessor system has been corrected.

P3609 MCP - CHECKPOINT RESTART WITH ARRAY PARAMETER

The rerun of a checkpointed task that was passed an array parameter will no longer abort with an instruction timeout when going to EOT. In addition, entering "<mix #> OT" will now work correctly on the restarted task.

P3610 MCP - VOLUME LIBRARY

A recently-introduced problem in the volume library has been corrected. Continuation entries in the volume library no longer leads to VAST and VOLLIB errors and rebuilds.

P3620 MCP - LIBRARY MAINTENACE TAPE ERRORS

The following problems have been corrected:

1. If the reel "switch directory" needed for copying to tape cannot be allocated (or an error occurs using it), Library Maintenance will terminate with an error message.
2. Certain "CLOSE errors" will now cause Library Maintenance to FR the affected destination tape; i.e., FR (Final Reel) stops output to the given destination.

P3621 MCP - PATHS, "FREE, DISKSTATUS" PROBLEMS

The following problems have been corrected:

1. A problem in DISKSTATUS caused by a "first action" result descriptor.
2. On a B6800 Multiprocessor system, FREE and ACQUIRE conflict with MOVED packs.
3. A "MISSING MPX" problem on a B6800 Multiprocessor system.

P3631 MCP - NEW FINE PRIORITY ALGORITHM

The fine priority is calculated as the sum of two terms (the higher the number, the lower the priority):

1. The age of the task: a number ranging from 0-4 based on the amount of processor time a task has used since it was started. The current calculation is:

PRIORITY	ACCUMULATED PROCESSOR TIME (SECS)
0	0- 3.99999
1	4- 15.99999
2	16- 63.99999
3	64- 255.99999
4	256- 999999999

2. The recent activity of the task: a number ranging from 0-3 based on the amount of processor time used since the last resurrect for this task.

PRIORITY	ACCUMULATED PROCESSOR TIME (MILLS)
0	0- 7.9999
1	8- 127.9999
2	128- 2047.9999
3	2048- 999999999

P3632 MCP - MESSAGES

The following two messages were accidentally concatenated into one message used for two purposes:

"WFL JOB FILE IS INCOMPATIBLE WITH THIS MCP, UNABLE TO RESTART"
and
"DEALLOCATING ACTIVE TASK VARIABLE"

The messages have now been separated.

P3647 MCP - RESOURCE WAIT

The calculation to determine the number of tapes available for resource allocation was returning too small a number. This problem has been corrected.

P3648 MCP - "DESTNAME" ATTRIBUTE

Jobs that use the DESTNAME attribute will no longer occasionally hang at EOJ time waiting forever on DESCUNLOCK.

P3649 MCP - "ODT" QUEUE

The following problems have been corrected:

1. The console ODT occasionally got hung in XMIT. This could occur if a WRITE request with an error (timeout) were ahead of a READ request. The READ is now moved ahead of the write.

2. The ODT hanging in XMIT also could be caused by a hardware problem. If an operator XMITs a request between the time that a WRITE to the ODT is started and the "write timeout" error condition, the ODT would not issue a status change for the operator's request to be read.

Example:

- a. An ADM page is being transmitted to the ODT.
- b. Before the page is complete, the ODT is put in local.
- c. A request is transmitted; the ODT is hung.

If, at step "c", the operator waits a couple of seconds (to allow the WRITE to timeout), the ODT will not hang.

3. The Mark 32 queue-driven keyin implementation has been removed and replaced by the Mark 31 (and earlier) forking of keyin.

P3650 MCP - "DS" PERMANENT LIBRARY

Libraries which were processed out of a user routine and froze permanent were not being DSed if the parent stack did the DS (using LIBTASK.STATUS:=TERMINATED). This caused a "SHAREDYBALL LIBRARY ALREADY INITIATED" message to be generated when the user attempted to process off the library again. This problem has been corrected.

P3651 MCP - EXPAND MAX TASK PARAMETERS

On the Mark 31 PR1 release, the maximum number of words which could be passed as parameters was 128. That number has been increased to 256; the error message has been changed accordingly.

P3652 MCP - "WFL" SUBROUTINES

WFL subroutines that have been processed by the jobstack will now work as they did on the Mark 31 release when using task variables declared in the jobstack.

P3653 MCP - "FS" AND "DS ODT" INPUTS

The FS (XS) and DS ODT commands will now work more reliably when used after an HS (EI) ODT command has been entered.

P3654 MCP - READ HEADER REORGANIZATION

The DCALGOL READ DISKHEADER command would not work if the file title passed as a parameter were of the form "<file> ON <packname>." This problem has been corrected.

P3685 MCP - "UR-"

UR- (or UA) of a disk or tape path that is not reserved will no longer cause a "PATH MARKED OFFLINE" error message.

P3686 MCP - "INVALID OP" IN PRESENCE BIT

If an installation intrinsic initiated an internal process, when that process went to EOJ the MCP would improperly destroy the code file header for the intrinsics stack. This would cause the MCP procedure PRESENCEBIT to get a fault on the next P-bit of an intrinsics segment. This no longer occurs.

P3687 MCP - "BDNAME" "SEG ARRAY" FAULT

A problem has been corrected where setting BDNAME to a non-usercoded value of 2 or less characters caused a dump.

P3688 MCP - AUTOPRINT "AX" COMMAND

The AX command to Autoprint now works correctly for the printing of backup tapes.

P3710 MCP - "DMSCLOSE" VS. "CONTROLLER"

When a data base stack went away, the CONTROLLER was not notified of the current in-use count of the data base.

This would cause the CONTROLLER to fail to release the data base stack number after the data base had gone to EOJ. The next task using that stack number would appear in the active entry list, but with the name of the old data base.

This problem has been corrected.

P3746 MCP - SCRATCH TAPE WITHOUT WRITE RING

A scratch tape without a write ring will now be treated as an unlabeled tape.

P3748 MCP - "DUP FILE" MESSAGE

The following RSVP message will now display the CR mnemonic when the job contains a WFL data deck that is causing a duplicate file condition:

<filename> DUP FILE ON: . . .

P3781 MCP - "TAPEDUMP," REPORT BLOCK, REWRITE COUNT

The "progress report" which TAPEDUMP writes on the ODT now includes the count of tape blocks written and the number of blocks rewritten because of error, if any.

Examples of Output:

ADDR=78000, PROC 2 BLOCKS WRITTEN: 137, REWRITTEN: 2
ADDR=C0000 BLOCKS WRITTEN: 29

Because the write-error recovery mechanism in TAPEDUMP is to serially number each block and simply rewrite any erroneous block without backspacing or erasing, it is difficult to spot a bad tape or drive by merely watching the tape. A few rewritten blocks are a normal occurrence, but a large or fast-growing REWRITTEN count is an indication that the dump should be CLEARED and restarted elsewhere. The REWRITTEN count is incremented once for each retry, whether of the same or different dump blocks.

The progress report is now written every 8000 (hex) words, rather than every 4000.

P3782 MCP - RESIZE OVERLAYABLE ARRAYS IN GLOBAL

A local memory task resizing an overlayable array belonging to a stack in global memory no longer causes an occasional dump by BAD MOM ADDRESS.

P3784 MCP - AVOID HUNG PRINTERS

Some internal MCP logic dealing with no-path situations has been revised to alleviate an occasional problem in which all a system's printers, card readers, etc., are hung. The problem has been observed on B6800 Multiprocessor systems, with more than one printer on a subsystem.

P3787 MCP - "GUARDFILE" VS. "CANDE"

CANDE no longer hangs waiting for missing guardfiles.

P3801 MCP - CORRUPTION OF "LIBUSEMAP"

Entries are no longer left in LIBUSEMAP which could cause users to hang waiting on libraries.

P3802 MCP - "FIB" CREATION LOCKING

A small window has been closed, so that ATTRIBUTEHANDLER will not fault when setting up a FIB in some other stack which is being stretched.

P3817 MCP - ERRONEOUS "DS" WHEN "DBS" INITIATION FAILS

When DMSOPEN was unable to initiate the ACCESSROUTINES code file because the user program did not have visibility to the data base stack, the user program could be erroneously DSed and not allowed to handle the openererror exception returned by DMSOPEN. This problem has been corrected.

P3822 MCP - "EXCEPTIONTASK" VISIBILITY

Under some circumstances in a Tightly-Coupled or Swapper environment, it is not always possible to cause the EXCEPTIONTASK.EXCEPTIONEVENT of a task which has had EXCEPTIONTASK assigned explicitly. Under these circumstances, the MCP will now skip causing the event, rather than take a dump or other failure action. There is no difficulty when, as by default, EXCEPTIONTASK is the parent (critical-block) task.

P3824 MCP - MEMORY ORGANIZATION, LOCAL "MCP" CODE

The organization of memory and its initialization in the MCP have been extensively revised. All permanently-save MCP code is now resident in one of two areas. The main effect of the change is that in B6800 multiprocessor (tightly-coupled) systems, the code for often-used MCP routines is now kept in each local memory subsystem. The result is a substantial reduction of memory accesses to * GLOBAL tm Memory, and a corresponding increase in throughput.

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

In a B6800 multiprocessor system, D0 is now placed at address 100 (hex) in the first mod of Global tm memory. Thus, a system which had D0 at 80046 will now put it at 80100. D0 in monolithic systems remains at 2100.

These changes constitute a major step in the elimination of "segment 1" and "segment 5" from the internal structure of the MCP code file: these segments now include only BOOTSTRAPMAKER and the outer-block call on that procedure. Because MAINTENANCE used segment 5 as a pattern for some disk and pack tests, there have been some changes in that area; see SCRMCPC note D3169 for details.

B6000 SERIES MARK 32

Some of the SYSTEST DCP tests depend upon a data descriptor to "seg 5" in absolute location zero. This mechanism was broken in the Mark 31 release. It has now been restored, but the area described is local MCP save code rather than segment 5. On a B6800 multiprocessor system, the DCP accesses the MCP save code in the local memory of the processor to which the DCP is connected.

P3825 MCP - PARITY ON PRESENCEBIT STACKOVERFLOW

A problem that could cause a fatal stackoverflow in some parity-on-presencebit situations has been corrected.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

MESSAGE LEVEL INTERFACE PORT

D3142 MLIP - INITIALIZATION ROUTINES FOR "B6900"

The initialization routines of the MCP have been extensively modified to allow for the initialization of B6900 systems using the new IO subsystem hardware.

The B6900 is initialized by means of a permanently resident bootstrap situated at location zero in memory. In order to accommodate this, the layout of that portion of memory below the D0 stack has been changed.

The procedure GETITGOING has been extended such that it is now responsible for the initial formatting of the low end of memory, and is also responsible for selecting and reading in the appropriate segments of code, depending on the type of system. The READIT calls previously in SECONDARYINITIALIZE are thus subsumed into the extended GETITGOING procedure.

New hardware operations exist on MLIP systems to read the time-of-day clock, and to keep the running light on. The defines invoking these functions have been modified to execute the appropriate operators, depending on the type of system.

See GENERAL note D2303, "B6900 Overview", for a description of the B6900 system.

D3355 MLIP - LOG "MLIP I/O" ERRORS

The LOGANALYZER now analyzes MLIP I/O error records (MAJOR TYPE 2, MINOR TYPE 16) in the log.

The MCP will log up to 500 retries per I/O error.

The SOG Reference Manual, Volume 2 Page 6-1-25, should be changed as follows:

In Table 1 under MAJOR TYPE 2, add the new MINOR TYPE 16 "MLIP I/O Error Record".

The PHYSICALIO MLIP module now logs its I/O errors with a major type of 2 (i.e. Maintenance record) and a minor type of 16. This minor type is a MLIP I/O error record and is the MLIP counterpart to the MPX I/O error record which has a minor type of 10. The first four words of the MLIP I/O error record contain the standard information found in any log record. The rest of the MLIP log record is different from the MPX log record.

WORD 4:

[47:8]: MLIP LOG TYPE specifies the type of MLIP log record.

= 1 indicates this is an error-recovery log record.

= 2 indicates this is an error-IOCB log record.

The rest of this note specifies the format for an MLIP I/O error-recovery log record.

The next three words of the log record contain control information about the structure of the record and contain unit information.

WORD 4:

[39:8]: MLIP LOG VARIANT specifies the subtype of the MLIP log record.

= 1 indicates the log record is for normal error recovery and all information for a retry is present.

= 2 indicates the log record is for limited error recovery and some of the information for each retry is not present. The RETRY DLP COMMAND and the RETRY MLIP CONTROL are not present.

[27:12]: NUMBER OF RETRIES which were logged.

[15:1]: IRRECOVERED.

= 0 indicates the I/O recovered.

= 1 indicates the I/O was irrecoverable.

WORD 5:

[47:12]: LOGICAL UNIT NUMBER.

[35:12]: PHYSICAL UNIT NUMBER.

[23:6]: LOGICAL UNIT TYPE.

[17:6]: UNIT SUBTYPE.

[11:12]: PHYSICAL UNIT TYPE.

WORD 6:
[47:4]: NUMBER OF FILE AND JOB INFO ITEMS is the number of items of file information and job information logged. File information includes the file title, the internal name, and the serial number. Job information is the job name.

[11:12]: FILE AND JOB INFO LOCATION is the number of the word in the log where the file and job information starts.

The next eighteen words of the log record contain information about the original I/O.

WORD 7:
[19:20]: MLIP CONTROL.

WORD 8: DLP ADDRESS.

WORD 9:

WORD 10:
WORD 11: MLIP STATE AND RESULT.

WORD 12: I/O START TIME.

WORD 13: I/O FINISH TIME.

WORD 14:
[45:10]: STACK NUMBER OF IOCB OWNER.
[33:10]: STACK NUMBER OF IO INITIATOR.

WORD 15: I/O CONTROL WORD.

WORD 16:
[47:20]: MEMORY ADDRESS OF START OF AREA.
[27:20]: LENGTH OF AREA.

WORD 17: LOGICAL RESULT DESCRIPTOR.

WORD 18: DLP COMMAND.

WORD 19: DLP RESULT (WORD 1).

WORD 20: DLP RESULT (WORD 2).

The next thirteen words are the extended result descriptor and the results of reading the extended result descriptor.

WORD 21 - WORD 33: EXTENDED RESULT DESCRIPTOR AREA.

The next two words contain I/O statistics.

WORD 34: [47:24] Total since H/L of read I/Os to this unit
[23:24] Total since H/L of write I/Os to this unit

WORD 35: [47:24] Total since H/L of read errors logged for this unit
[23:24] Total since H/L of write errors logged for this unit

The next section contains retry entries of five words each. The NUMBER OF RETRIES (WORD 4) which were logged is variable, so this section is of variable length. The format of each entry is as follows:

ENTRY WORD 0: RETRY DLP COMMAND. (Not present if LOG VARIANT is 2.)

ENTRY WORD 1: RETRY DLP RESULT (WORD 1).

ENTRY WORD 2: RETRY DLP RESULT (WORD 2).

ENTRY WORD 3:
[47:1]: ACTUAL RETRY.

= 0 if entry is not an actual attempt to retry the I/O (e.g. repositioning a tape).

= 1 if entry is an actual attempt to retry the I/O.

[19:20]: RETRY MLIPCONTROL. (Not present if LOG VARIANT is 2.)

ENTRY WORD 4: RETRY MLIP STATE AND RESULT.

The last section contains file and job information. Since the length of the previous section varies, the starting word of this section is also variable and is the value of FILE AND JOB INFO LOCATION in the control section (WORD 6). The NUMBER OF FILE AND JOB INFO ITEMS (WORD 6) is variable as well as the length of each item, so this section is of variable length. The format of each item is identical to the file and job info items of the MPX I/O error log records. The first word of each item specifies the type and length of the item:

ITEM WORD 0:
[39:20]: LENGTH OF ITEM (including this word).

[19:20]: TYPE OF ITEM.

= 1 indicates item is SERIAL NUMBER info.

= 2 indicates item is FILE TITLE info.

= 3 indicates item is JOB NAME info.

= 7 indicates item is INTERNAL NAME info.

If the type of item is SERIAL NUMBER information, the rest of the item's layout is as follows:

ITEM WORD 1: LEBCONTROL.

ITEM WORD 2: GENEALOGY1.

ITEM WORD 3: GENEALOGY2.

If the unit is a pack, magtape, or diskette, the SERIAL NUMBER item contains a fifth word:

ITEM WORD 4: SERIAL NUMBER in EBCDIC characters.

If the type of item is FILE TITLE information, the rest of the item's layout is as follows:

ITEM WORD 1: ROW INDEX of the file on pack or disk. This word will be all ones if there is no valid row index or unit is not a pack or disk.

If the length of the FILE TITLE item is greater than two, the rest of the item is the file title in standardform.

If the type of item is JOB NAME information, the rest of the item is the job name in standardform.

If the type of item is INTERNAL NAME information, the rest of the item is the internal name in standardform.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

MESSAGE LEVEL INTERFACE PORT

P2763 MLIP - IMPLEMENT "PRINTERDUMP"

The MCP routine MEMDUMP, used for direct memory dumps to the printer, has been modified to work on the B6900.

Since this portion of the MCP symbolic may be extracted and compiled as a stand-alone program, it contains all necessary declarations and code to determine the type of system on which it is running, and to execute accordingly. An indication of the type of system is printed as part of the heading.

Other than the above, there are no functional changes to the routine.

P2764 MLIP - "PRINTIOCB" INTERFACE ANALYZES "IOCB"

Instead of invoking its own PRINTIOCB procedure to print the IOCB, PROGRAMDUMP now invokes a new PHYSICALIO interface called PRINTIOCB. PRINTIOCB has two parameters: the IOCB and a "print" procedure (which accepts an array as a parameter). PROGRAMDUMP passes as the procedure parameter a new local procedure called PRINTLINE. The PRINTIOCB interface calls PRINTLINE after building a new line it wants printed.

DOCUMENT CHANGES NOTES (D NOTES)

NETWORK DEFINITION LANGUAGE

D3028 NDL -- MESSAGE-ORIENTED DATACOM

The capabilities of Message-Oriented Data Comm have been extended by the development of the Adapter Cluster Model III (AC III). This software release supports the use of the BDLC protocol on the AC III and is to be used in conjunction with Burroughs Network Architecture (BNA).

A complete description of the changes to NDL as a result of this new implementation may be found in the Message-Oriented Data Communications Reference Manual, Form No. 5011836.

D3204 NDL -- CHANGES FOR "DCP" CHARACTER ORIENTED DATA COMM

The following NDL compiler and NDL Reference Manual changes are relevant to DCP character-oriented data comm (i.e. non-message-oriented, non-B6900 data comm):

- a. In the NDL Reference Manual, the Control Definition INITIALIZE statement (page 5-28) has been revised to delete the INITIALIZE SEQUENCE option.

The syntax diagram should read as follows:

```
-- INITIALIZE --- BCC -----|
                        |
                        | - CRC ---|
                        | - RETRY -|
```

- b. In the NDL Reference Manual, the description for the Request Definition TERMINATE statement has been revised to delete the last paragraph on page 5-141.
- c. The following station characters have been added as read-only byte variables: END, BACKSPACE, CONTROL, LINEDELETE, WRU. These variables can be interrogated in Control definitions, Transmit requests, and Receive requests.
- d. The Request Definition INITIALIZE statement allows two new options: TOGS and ERRORFLAGS.
- e. The Request Definition STORE statement has been expanded to allow STORE TOGS.

* NOTE: In order to provide one consistent syntax diagram for each NDL statement changed this release, these features are described, along with the features added for message-oriented data comm, in the Message-Oriented Data Communications Reference Manual, Form No. 5011836.

D3385 NDL -- DIFFERENT TERMINAL ADDRESSES

In the past, if two terminal definitions referenced the same request set, and one specified "ADDRESS=2." while the other specified "ADDRESS=2(DIFFERENT).", the NDL compiler would produce two copies of the code for the request set. One copy always used the transmit address, even if the receive address were specified (since they were identical), while the other copy used the proper address always. Furthermore, if the line control code accessed the receive address and had stations of both terminals on its line, its code might be generated either way.

Since whenever the addresses are identical, NDL makes the receive and transmit addresses be identical, it is possible now to always generate the code as if the addresses were different. Such code will work in either case, so that change has now been made.

D3527 NDL -- TRANSMIT/RECEIVE DELAYS

The NDL manual is misleading in its description of transmit/receive delays and timeout delays. While these quantities are calculated for each station as described in the NDL Reference Manual (Form No. 5001522), the DCP actually stores and uses only one value for each line. The line value is determined by maximizing the station delays for all stations assigned to the line. The discussion of the INITIATE TRANSMIT and INITIATE RECEIVE statements on Pages 5-20, 5-30, 5-120 and 5-121 and the discussion of the RECEIVE statement on Page 5-34 and 5-124 should be read with this fact in mind.

D3543 NDL -- "<LINE DEFINITION>" STATEMENTS

The NDL Reference Manual (Form No. 5001522) is not very specific about which <line definition> statements can be in a default <line definition> and which cannot. The following statements cannot be part of a default <line definition>:

```
<line address statement>
<line answer statement>
<line commandblock statement>
<line endofnumber statement>
<line station statement>
<line type statement>
```

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

NETWORK DEFINITION LANGUAGE

P2714 NDL - PAGE BETWEEN "VOIDT" AND "POP VOIDT"

The NDL compiler would ignore any \$PAGE cards in the card file which appeared within the range of \$VOIDT and \$POP VOIDT pair. This problem has been corrected.

P2804 NDL - "ENTER"

The NDL compiler would get an INVALID INDEX if a label were used but not found, if the label usage occurred on an internal table boundary. This problem has been corrected.

P3256 NDL - INHIBIT SYNC EDIT WRONG

In a reorganization of NDL during 30 PR1, an error was accidentally introduced into the cluster type field generation for synchronous lines. As a result, the terminal INHIBITSYNC (SYNCS)=TRUE/FALSE attribute was being incorrectly applied, and the type field generated might or might not have the proper setting for editing or not editing sync characters. This problem has been corrected.

P3599 NDL - ENLARGE "UNFO" ARRAY

The compiler could get an INVALID INDEX fault when compiling a network containing more than 1300 stations. This problem has been corrected by increasing the size of an internal table.

P3604 NDL - "\$NETWORK" OPTION

In NDL, the compile-time option \$NETWORK, which prints a network summary showing the datacom network, can now be used safely when more than 26 stations are defined on a line.

P3605 NDL - CLEAR LINETABLE ARRAY

Currently, if MAXSTATIONS is defined on a line which has less than maxstations stations, an INVALID INDEX may result if the \$SET NETWORK option is used. This problem has been corrected.

P3751 NDL - CALL "BRANCHLINK"

Several Message-Oriented Datacom constructs were unable to correctly optimize GOTO chains. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

NDLII

D3397 NDLII - "NDLII" IMPLEMENTATION

A new Network Definition Language, NDLII, is available for use with the B6900 Data Comm Subsystem. NDLII is a procedural language which allows flexible use of the B6900 data comm system. NDLII is not compatible with B6700/B6800 datacom systems; the current NDL will still be used on these systems.

NDLII is described in the file NDLII/DOCUMENT contained on the SYSTEMNOTES tape included with the Mark 32 release.

DOCUMENT CHANGES NOTES (D NOTES)

NEWP

D2973 NEWP - "AT <LIBRARY ID>" ALLOWED

A library identifier is now allowed as the <location designator> part in the "<type> AT <location designator>" syntax.

D2974 NEWP - "INITIALIZATION" IS RESERVED WORD

"INITIALIZATION" is now a reserved word; it cannot be used as an identifier.

D3031 NEWP - CONDITIONAL OPERATORS

The NEWP language now has binary infix <conditional-operator>s which are used to combine <boolean-primary>s in boolean expressions in the same way that <logical-operator>s are used.

The <conditional-operator>s are similar to the <logical-operator>s, except that the right-hand operand is not evaluated if the value of the left-hand operand is sufficient to determine the value of the operation.

Syntax:

<conditional operator>

```

----- CAND -----|
| - COR  -- |
| - CIMP - |

```

Operands		Operations			
L	R	L CAND R	L COR R	L CIMP R	
TRUE	bool	bool	TRUE	bool	
FALSE	bool	FALSE	bool	TRUE	

The <conditional-operator> 'CAND' has the same precedence as the <logical-operator> 'AND', 'COR' the same precedence as 'OR', and 'CIMP' the same precedence as 'IMP'.

EXAMPLES:

```

B := R1 NEQ 0 CAND R2/R1 EQL R3;
B := R1 GEQ 0 AND R1 LSS SIZE(A) COR R2 NEQ A[R1];
B := R1 GTR 0 CIMP A[R1-1] = R1;

```

D3032 NEWP - IMPLEMENT CONTROL STATE BLOCKS

Control state blocks are now available in the NEWP language. The "unsafe" block directions CONTROLSTATE and NORMALSTATE are now allowed following any BEGIN, specifying that the compound statement is to be a block which is run in control state or normal state, respectively. This concept allows the static specification of normal state versus control state in a NEWP program like the MCP. Previously, NEWP programs used the DISALLOW and ALLOW statements (inherited from ESPOL) to dynamically alter the "control" state of the processor. ALLOW and DISALLOW have been de-implemented as NEWP language constructs.

In the absence of any explicit compiler direction, the "state" of a block (and the body of an ON declaration) is inherited from the containing block; the outer block and all procedures are by default normal state. (The Mark 31 NEWP compiler caused procedures without these explicit state directions to inherit the state from their containing environment and caused all ON declarations to be normal state.)

The explicit use of CONTROLSTATE or NORMALSTATE block directions creates a true block; in particular, a GOTO from outside the range of such a block cannot transfer into the body of the block. Also, a GOTO which transfers from within such a block to a more global block is allowed, but is treated as a "bad-go-to". All "bad-go-to"s cause an effective loss of control, even if both the GOTO statement and the destination label are in (different) "control" state blocks.

The BUZZ and UNLOCK constructs remain in the NEWP language, but the compiler now restricts their use to only control state environments; a syntax error is given for their use in normal state.

In addition, compound statements which specify the STATSUMMARY compiler direction are now also treated as distinct blocks.

D3033 NEWP - PREVENT "GOTO" INTO "FOR" STATEMENT

The body of a FOR statement is now considered to be a new environment, thus preventing GOTO statements from branching from the outside to the inside of the body of a FOR statement.

D3058 NEWP - INLINE PROCEDURES

Inline procedures combine the efficiency of defines with the semantics of procedures. Each invocation of an inline procedure will result in an inline expansion of its code at the point of the invocation. Each allowable invocation of an inline procedure maintains the semantics of a procedure call.

The lineinfo for an expanded inline procedure will include both the sequence number(s) of the invoking code and the sequence number of the invoked code. These sequence numbers will be in order from local to global and separated by a slash ('/').

Syntax:

a). Declarations

The new keyword 'INLINE' will be a block directive allowed after the first 'BEGIN' of the procedure body of a procedure declaration.

Example:

```
REAL PROCEDURE PROC;
  BEGIN [INLINE]
    :
    :
  END PROC;
```

b). Modules

An inline procedure that is exported from a module must have its body occur in the module heading. A non-inline procedure may not have its body occur in the module heading.

Restrictions:

The following restrictions exist for inline procedures. Syntax errors will be issued if they are violated.

a). Recursion

Recursion of inline procedures is not allowed.

b). Parameters

An inline procedure may not be passed as a formal parameter to a non-inline procedure. However, both inline and non-inline procedures may be passed as formal parameters to an inline procedure.

c). IPC

IPC commands, (e.g. PROCESS, RUN, etc.), may not specify an inline procedure.

d). Libraries

An inline procedure may not be exported as a library.

e). Non-invocation references

Non-invocation references, (e.g. MAKEPCW, LEXOFFSET, etc.), to inline procedures are not allowed.

f). RETURN and EXIT

The RETURN and EXIT statements will not be allowed within the body of an inline procedure

g). Forwards

An inline procedure may not be declared forward.

h). Initialization procedures

An initialization procedure may not also be an inline procedure.

i). SORT statement

An inline procedure may not be used in a SORT statement.

j). Module visibility

As a temporary restriction, the only imported items an exported inline procedure may use are those exported from modules declared prior to the module containing the exported inline procedure.

D3064 NEWP - ARRAYS WITH UNSPECIFIED BOUNDS

NEWP allows the indication that an array bound is unspecified. A bound pair of 0:-1 indicates an unspecified bound. Except for this special case, the lower bound may not exceed the upper bound. If a dimension of an array is unspecified, it must either be the last dimension or all the dimensions must be unspecified.

When unspecified bounds are used, the MCP (ARRAYDEC) is called if only the last dimension of a multi-dimensional array is unspecified. For all other cases of unspecified bounds, an empty descriptor is built for the array.

D3068 NEWP - "ALTERNATIVES" AND "INITIALIZATION" PROCEDURES

Alternatives and Initialization Procedures

Modules may now contain <initialization procedure>s and <alternative>s.

Following is the new syntax for <module body>.

<module body>

```

----->
| |<-----| | |<----- ; -----| |
| |<import-list>-- ; --| |<declaration>--|
-----|
> | |<-----| | |<initialization procedure>--|
| |<alternative>--|

```

<alternative>

```

-- ALTERNATIVE --<alternative id>-- ; ----->
>- BEGIN --<alternative id>-- ; ----->
| |<-----| | |<declaration>--|
| |<declaration>--|
>- END --<alternative id>-- ; -----|

```

If a module contains any alternatives, then an initialization procedure must appear last in the <module body>.

If an initialization procedure occurs in a module, all the items declared in the module, except the initialization procedure, are unavailable until the initialization procedure is entered. At that time the items declared in the module but outside any alternatives are initialized. Items declared in an alternative are not available until and unless a select statement for that alternative is executed.

Items declared in a module but outside any alternatives are available inside all the alternatives in that module. Therefore, if a procedure is declared forward outside the alternatives, its actual declaration must occur either outside the alternatives or inside each alternative.

Alternatives may not contain external procedures.

Initialization Procedure

The differences between initialization procedures and other procedures are:

1. An initialization procedure may contain select statements.
2. An initialization procedure may only occur as the last declaration in a <module body>.
3. All items declared in the module, but outside any alternative modules that might exist, will be initialized when the initialization procedure is entered.

4. An initialization procedure may only be executed once. A run-time fault will occur if an attempt is made to execute it a second time.
5. The SEGMENT block direction may not be specified in the block directions for an initialization procedure.

Select Statement

<select statement>

-- SELECT -- (--<alternative id>--) -----|

The select statement initializes the module to include the the items declared inside the specified <alternative id>.

A select statement may only occur in an initialization procedure.

Only one select statement may be executed in an initialization procedure. A run-time fault will occur if a second select statement is executed.

Example:

```
BEGIN
MODULE PHYSICALIO;
  EXPORT PHYSICALIOINITIALIZATION,
    DOCHARIO,
    T;
  BOOLEAN INITIALIZATION PROCEDURE PHYSICALIOINITIALIZATION(WHICHONE);
    VALUE WHICHONE; BOOLEAN WHICHONE; FORWARD;
  PROCEDURE DOCHARIO; FORWARD;
  INTEGER T;
BEGIN PHYSICALIO;
  REAL R;
  ALTERNATIVE HDPHYSICALIO;
  BEGIN HDPHYSICALIO;
    INTEGER I;
    PROCEDURE DOCHARIO;
      BEGIN
        END DOCHARIO;
    END HDPHYSICALIO;
  ALTERNATIVE MPXPHYSICALIO;
  BEGIN MPXPHYSICALIO;
    PROCEDURE DOCHARIO;
      BEGIN
        END DOCHARIO;
    REAL PROCEDURE IOFINISH68;
      BEGIN
        END IOFINISH68;
    END MPXPHYSICALIO;
  BOOLEAN INITIALIZATION PROCEDURE PHYSICALIOINITIALIZATION(WHICHONE);
    VALUE WHICHONE; BOOLEAN WHICHONE;
  BEGIN
    IF WHICHONE THEN
      SELECT(HDPHYSICALIO)
    ELSE
      SELECT(MPXPHYSICALIO);
    PHYSICALIOINITIALIZATION:=WHICHONE;
  END PHYSICALIOINITIALIZATION;
END PHYSICALIO;
MODULE INITIALIZER;
BEGIN INITIALIZER;
  IMPORT FROM PHYSICALIO(PHYSICALIOINITIALIZATION);
  BOOLEAN WHICHONE;
  PROCEDURE GETITGOING;
  BEGIN
    PHYSICALIOINITIALIZATION(WHICHONE);
  END GETITGOING;
END INITIALIZER;
END.
```

D3103 NEWP - EVENTS AND EVENT ARRAYS AS PARAMETERS

Events may now be passed as parameters to libraries; event arrays may now be passed as parameters to procedures and libraries.

D3106 NEWP - CLARIFICATION OF "MAKEPCW" RESTRICTIONS

This note describes the use of and restrictions on the unsafe NEWP function MAKEPCW.

MAKEPCW accepts as a parameter either a procedure id or a label id. The result returned is a word value containing the hardware PCW pointing to the code for the procedure or label. This PCW will properly have the NCSF field set to one for control state procedures and for labels declared in control state environments.

Restrictions on the use of procedure ids:

1. The procedure must not be declared EXTERNAL, NULL, LIBRARY, BY CALLING, or INLINE.
2. The procedure must not still be FORWARD at the time of the MAKEPCW invocation.
3. The MAKEPCW invocation cannot occur within the body of the procedure being passed as the parameter to MAKEPCW.

Restrictions on the use of label ids:

1. The declaration of the label must not be more global than the beginning of the code segment in which the label is used as a MAKEPCW parameter.
2. The PCW generated points to the actual label occurrence, rather than to any hidden label generated for "bad GO TO" optimizations.

D3110 NEWP - "<PROCEDURE NAME>.VALUE"

Within the scope of a typed procedure, the value of the typed procedure may now be accessed via the following construct:

```
<procedure name>.VALUE
```

This construct may be used in expressions, assigned to, and address equated to.

Example:

```
REAL PROCEDURE PROC;
BEGIN [UNSAFE(MISC)]
  BOOLEAN B= PROC. VALUE;
  PROC.VALUE:=10;
  IF PROC.VALUE=20 THEN
    :
    :
  END PROC;
```

D3151 NEWP - INCREASED HOST BLOCKSIZE

The BLOCKSIZE of the compiler's HOST file has been increased from 270 to 2160 to reduce the I/O time and elapsed time for SEPCOMP.

D3152 NEWP - ALLOW THE "MCP" TO FREEZE AS A LIBRARY

The following changes have been made to NEWP to allow the MCP to freeze as a library:

1. MCP is allowed as a value for the parameter to the FREEZE statement. This will only be allowed if the program being compiled has set the MCP dollar option.
2. FREEZE(MCP) will be allowed in a block with no library export lists. It will export all entry points that occurred in library export lists at D0 prior to that point in the symbolic. Any subsequent D0 library export lists will get syntax errors. The FREEZE(TEMPORARY) and FREEZE(PERMANENT) cases of the FREEZE statement will continue to be allowed only in the same block as a library export list.

If a D0 library export list is changed during sepcomp then any procedures with FREEZE(MCP) statements must be touched so that the FREEZE statement will be recompiled.

Note that an export list in a module head (i.e. between the 'MODULE <module-id>' and the 'BEGIN <module-id>') is a module export list. An export list in any other location is considered to be a library export list.

D3158 NEWP - MATCH "NEWP" CODEFILE LEVEL TO "BINDER" LEVEL

The NEWP compiler now stores a bindinfo level in the codefile for the BINDER to check to ensure that the BINDER can bind that level of NEWP codefile. If the BINDER and NEWP levels mismatch, an error will be given at bind time.

D3248 NEWP - "DEFINE" EXPANSION

DEFINE expansion will not occur when processing the individual identifiers in the <value part> or <specification part> of a procedure heading. This differs from ALGOL, which expands defines in these places.

D3260 NEWP - "FIRSTFREEDOCCELL" NOW DEFAULTS TO "10"

The default value of the FIRSTFREEDOCCELL block direction is now 10 (decimal) rather than 224 (decimal). Furthermore, FIRSTFREEDOCCELL is prevented from having a value less than 10 (decimal). This causes no problem with the MCP, as its symbolic has an explicit setting for this block direction.

In addition, a warning will now be issued when FIRSTFREEDOCCELL has a value less than 222 (decimal) and the dollar option STATISTICS is set. (The compiler sets the value of MCPHIGHSTATNUM=(0,222) to the number of statistic entries when \$STATISTICS is set and FIRSTFREEDOCCELL is greater than 222).

D3261 NEWP - NEW FAULT NAME, "LIBLINKFAULT"

A new fault name, LIBLINKFAULT, is now recognized by the compiler. This fault occurs during an unsuccessful attempt at linking libraries. The fault number is 21. This fault is trapped by the ON ANYFAULT declaration also.

D3262 NEWP - NEW \$ OPTION, "STANDALONE"

A new \$ option has been added to the NEWP compiler for use in compiling stand-alone system programs such as the SYSTEM/LOADER. This option is \$STANDALONE. To be effective, this option must be set prior to the beginning of the program and \$MCP must also be set. As with the MCP itself, the compiler block directive on the outer block must specify into which code segment the outer block is to be compiled; e.g., SEGMENT=5. In addition, the procedures that must run in control state must be specified as CONTROLSTATE in their block direction.

This option causes the compiler to prepare a complete memory image of the program so that the program is "ready to run" when loaded into the system (starting at memory location zero). The maximum size of this memory image is 22,000 words.

This memory image consists of the following items:

1. The D0 stack image (location (0,9) has been set to zero, this was the LINEINFO dictionary descriptor).
2. All code segments.
3. All value arrays.
4. All "pool data" items.
5. The storage space for all SAVE arrays declared at a D0 location.
6. The proper entry PCW for the outer block at (0,3).
7. A "memory" descriptor at (0,4).

The memory image does NOT include:

1. Allocated data storage for any array declared within a procedure or any D0 array not declared to be SAVE. (No errors or warnings are given for these, as it is presumed that a proper presence-bit handling routine will be provided by the user.)
2. No "memory links" are provided to separate any of the allocated storage areas.

The code file generated by the NEWP compiler when the \$STANDALONE option is set contains only three items:

1. A bootstrap in code segment 0.
2. A valid "SEGO" in code segment 1.
3. The memory image as described above, starting in code segment 2 and continuing to the end of the file. The SEGO[18] word (the segment dictionary pointer) properly describes the D0 image in the first part of the complete memory image.

The code file does NOT contain any SEPCOMP information, no BINDINFO, no LINEINFO, and no PPB.

If \$STACK is set, the \$STANDALONE option will print a table of the code segment descriptors and data descriptors in D0, indicating which ones remain absent and showing the memory locations of the ones made present.

D3283 NEWP - "XREFING" ALTERNATIVES

Alternatives XREF in the same manner that modules XREF.

If a procedure has its forward in a module but outside any alternatives and its actual body in each alternative, each "alternative" procedure body is XREFed as an alias of each of the other "alternative" procedure bodies.

D3299 NEWP - CLOSE "LINE" AND "ERRORFILE"

The compiler output files LINE (program listing) and ERRORFILE (error message) will now be closed at the end of compilation before production of the cross-reference (XREF) is begun, thus allowing examination of those output files without waiting until the XREF is finished.

D3300 NEWP - SEGMENT IDENTIFIERS

A new type of identifier, the <segment identifier>, has been added. It is used to refer to a code segment. <Segment identifier>s are declared in <segment declaration>s.

<segment declaration>

--- SEGMENT ---<identifier list>-----|

An implicitly declared segment identifier, OUTERBLOCK, exists. It refers to the code segment that contains the outerblock code.

SEGMENT is not a general type. It can only be used as described below.

1. <segment identifier>s are allowed in block directions of the form SEGMENT = <segment identifier>.
2. Any variable that can normally be address equated, can be address equated to a <segment id>. Also, <segment id>s can be address equated but only to absolute addresses with lex level = 0 and displacement less than the fixed D0 fence.

Examples:

```

SEGMENT SEG1 = (0,1),      % legal
        SEGA = (0,1000),  % syntax error if 1000 is
                        % above the fixed D0 fence
        SEGB = (1,2),    % syntax error because it's
                        % not at lex level 0
        SEGC = REALID;   % syntax error because it's
                        % relative, not absolute
                        % address equation
WORD    WSEG1 = SEG1;    % legal

```

3. <segment identifier>s may occur in export lists and import lists.
4. <segment identifier>s are allowed as the parameter to LEXOFFSET and as the <location designator> in <type> AT <location designator>.

Examples:

```

REALID:=LEXOFFSET( SEGID );
WORDID:=WORD AT SEGID;

```

A syntax error will be given if a <segment id> other than OUTERBLOCK is referenced with LEXOFFSET or AT but is never used in a SEGMENT block direction. This will prevent accessing a <segment id> when there is no code segment associated with it.

D3401 NEWP - "PORTS" OPTION DISCONTINUED

The PORTS option of Programdump is no longer recognized by NEWP.

D3420 NEWP - "PARITYFAIL1" FAULT

PARITYFAIL1 may now be specified in an on statement when \$B7000 is set.

D3421 NEWP - "HEYOU" DISALLOWED ON "B7000" SYSTEMS

Use of HEYOU under \$B7000 is now a syntax error.

D3422 NEWP - "ZAP" INTRINSIC FOR "B7000"

The compiler no longer recognizes special syntax in the ZAP intrinsic under the \$B7000 option.

D3470 NEWP - "NEWP" STRING, NUMERIC CONSTANTS

STRING AND NUMERIC CONSTANTS

String constants and numeric constants have been added to NEWP. These two language components are syntactically distinct, thus the ambiguity between string and numeric constants present in ALGOL has been eliminated.

STRING CONSTANTS

<ebcdic string>

<ascii string>

<hexadecimal string>

<bcl string>

```

----- 6 ----- " <bcl character> " -----
                    |-----|
                    |<-----|

```

<hex character>

```

----- 0 -----|
|
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| A |
| B |
| C |
| D |
| E |
| F |

```

<ebcdic string character>

```

----- any ebcdic character except a quote (") -----|

```

<ascii string character>

```

----- any ascii character except a quote (") -----|

```

<bcl character>

```

-- any bcl character except a quote (") -----|

```

A string constant can be composed of EBCDIC(8-bit), ASCII(7-bit in 8-bit format), or hexadecimal(4-bit) characters. A string constant is always left justified.

NOTE: The following are differences between NEWP and ALGOL string constants.

1. There will not be any implicit concatenation in NEWP as in ALGOL. The concatenation symbol is required.
2. NEWP does not allow string constants to be considered as numeric constants without using the explicit type transfer function REAL.
3. NEWP uses the WFL rule for quoting a quote character not the ALGOL rule. To quote a quote character, two quote characters are required within a string constant.
4. An empty string is denoted by "" rather than the word EMPTY.
5. The "left justifying" prefixes (i.e. 80, 480, ...) are not implemented.

Examples

```

8"ABCD123";          result = 'ABCD123'
""WHY" & 48"6F" & ""; result = '"WHY?'"
"";                  result = an empty ebcdic string
7"";                 result = an empty ascii string
4"";                 result = an empty hex string

```

THE CONCATENATION OPERATION WITH STRING CONSTANTS

Two or more string constants may be concatenated together by use of the <concatenation operator>. The concatenation of two strings yields a new string whose length is the sum of the lengths of the two original strings. The value of the new string is formed by joining the second string immediately onto the end of the first string.

Only string constants of the same character type may be concatenated. If they are not of the same type, a syntax error occurs.

Examples

```
"STRING" & "CONSTANT"
"NUMBER" & 48"F1"
47"3138" & 7"ASCII CHARACTERS"
" " & "DOESNT DO MUCH"
```

THE REPLACE STATEMENT WITH <string constant> SOURCE

To make it possible to transfer data from a string to an array, a <string constant> may be used as <source part> in a <source list> in a REPLACE statement.

Examples

```
REPLACE P BY "STRING CONSTANT";
REPLACE P BY "STRING NUMBER " & 48"F1";
```

TYPE TRANSFER FUNCTION

<REAL function>

```
-- REAL -- ( -- <string constant> -- ) -----|
```

The REAL function returns as a real value the right justified bit image of the <string constant>. All bits in each character are used. The <string constant> may not exceed 48 bits in length.

Examples

```
R := REAL("STRING");
R := REAL("TOO LONG");           compile-time error
```

NUMERIC CONSTANTS

<number>

```
----- <sign> --- <unsigned number> -----|
|----- <numeric constant> -----|
```

```

----- <EBCDIC constant> -----
|
| --- <ASCII constant> -----
|
| --- <BCL constant> -----
|
| --- <hexadecimal constant> -
|
| --- <octal constant> -----
|
| --- <quaternary constant> --
|
| --- <binary constant> -----
|

```

```

-----, |<-----|
| 8 |, |-----|
| 80 | |'-----|

```

```

----- 7 ----- , -----/6\----- <ASCII numeric character> ----- , -----
| -- 70 -- | | -- ' -- |

```

```

----- 6 ----- , -----/8\----- <BCL numeric character> ----- , -----
| -- 60 -- | | -- ' ' ----- |

```

```

----- 4 ----- , -----/12\----- <hexadecimal character> ----- , -----
| -- 40 -- |

```

```

----- 3 ----- , -----/16\----- <octal character> ----- , -----
| -- 30 -- |

```

```

----- 2 ----- , -----/24\----- <quaternary character> ----- , -----
| -- 20 -- |

```

```

----- 1 ----- , -----/48\----- <binary character> ----- , -----
| -- 10 -- |

```


<binary character>

```

----- 0 -----|
|  -- 1  --  |

```

<quaternary character>

```

----- <binary character> -----|
|  -- 2 -----  |
|  -- 3 -----  |

```

<octal character>

```

----- <quaternary character> -----|
|  -- 4 -----  |
|  -- 5 -----  |
|  -- 6 -----  |
|  -- 7 -----  |

```

<hexadecimal character>

```

----- <octal character> -----|
|  -- 8 -----  |
|  -- 9 -----  |
|  -- A -----  |
|  -- B -----  |
|  -- C -----  |
|  -- D -----  |
|  -- E -----  |
|  -- F -----  |

```

<BCL numeric character>

```

----- any BCL character except a single quote (') -----|

```

<ASCII numeric character>

```

----- any ASCII character except a single quote (') -----|

```

<EBCDIC numeric character>

```

----- any EBCDIC character except a single quote (') -----|

```

Numeric constants provide a way to specify a number as a bit mask (in 7 different bases or character sizes) in a general way which is not ambiguous with string constants and does not require any type transfer. This is accomplished by using the single quote (') character to specify a numeric constant.

To specify a BCL, EBCDIC, or ASCII numeric constant which contains a single quote, two adjacent single quotes are used.

As with ALGOL string constant semantics, left justification may be denoted by using a character code which is a multiple of 10.

Numeric constants may not be less than 1 bit, nor may they exceed 48 bits in size.

D3486 NEWP - DIRECT "I/O"

Direct I/O has been implemented in NEWP. The syntax and semantics for Direct I/O is the same as for the ALGOL Language.

D3487 NEWP - "MCP" CODE FILE ROW SIZE = "504"

The MCP no longer has any code segments large enough to require that the MCP code file have a row size of 1008 disk segments.

The NEWP compiler and the BINDER will now construct an MCP code file with a row size of 504 segments, as is done for other code and symbol files.

D3530 NEWP - "FUNCTIONNAME," "LIBACCESS" ATTRIBUTES

Library declarations may now specify the FUNCTIONNAME and LIBACCESS attributes. FUNCTIONNAME is a string-valued attribute used to specify the system function name that will be used to find the target code file for the library. LIBACCESS is a mnemonic-valued attribute: the value BYTITLE indicates that the TITLE attribute of the library is to be used to find the library's code file. The value BYFUNCTION indicates that the FUNCTIONNAME is to be looked up in the MCP library function table which is maintained by the ODT message SL (see GENERAL note D3356) and the associated code file name will be used.

D3535 NEWP - "PACKDECIMAL" INTRINSIC

The PACKDECIMAL intrinsic has been implemented in NEWP.

Syntax:

```
-- PACKDECIMAL -- ( --<arithmetic expression 1>-- , ----->
>-<arithmetic expression 2>-- ) -----|
```

The semantics of PACKDECIMAL are identical to the SCALERIGHTF function semantics in ALGOL.

D3549 NEWP - "8-DIGIT" PATCH MARKS

The NEWP compiler will now detect patch marks (in the TAPE file) which consist of exactly 8 digits followed by either two blanks or two NULs. These patches marks have dots inserted so that they are listed as "dd.ddd.ddd".

D3590 NEWP - "DESCRIPTOR" PROCEDURES TO LIBRARIES

Parameters of type DESCRIPTOR are now properly distinguished from the type word for procedures being exported or used by a library.

D3591 NEWP - PROCEDURE ENTRY VIA REFERENCES

The "VIA" reference notation has been extended. In addition to the

<type> VIA <word primary>

form for simple objects, a similar form applies to procedure invocation:

<procedure name> VIA <word primary>.

The <word primary> is used as a reference to effect procedure entry; it should result in an IRW to a PCW for the desired code in the appropriate environment. All type checking and parameter matching is performed according to the declared procedure heading, but the address couple of the procedure is irrelevant.

Examples:

```
T:=PROC VIA REFERENCE TO W (PARAM)
MYGEORGE VIA WORDSPIB[SNR,SIRWTOPALACE] (WHY)
```

D3592 NEWP - "PROTECTED" OPTION IN LIBRARY EXPORT LIST

Library entry points declared in NEWP programs may now be exported with "protection". Such entry points may only be linked to by system libraries.

New Syntax:

```
-- EXPORT ----->
      | - [ PROTECTED ] - |
      | ----- |
      | <----- |
>---<procedure id>-----|
      | - AS <ebcdic string> - |
```

This feature is intended to be replaced with more general protection constructs in a future release.

D3593 NEWP - "REGISTERS" AND "DLL"

The unsafe intrinsic REGISTERS may now be used with an integer constant parameter only. In addition, a new intrinsic, DLL, has been added which references the register D[LL]. DLL is available under UNSAFE(REGISTERS).

D3598 NEWP - "SYSTEMLIB" LIBRARY ATTRIBUTE

A new Boolean library attribute, SYSTEMLIB, has been added to NEWP. When SYSTEMLIB is set, the associated library is to be initiated as a "system library"; therefore, it has access to protected MCP procedures. Use of this attribute requires the dollar option \$MCP.

D3608 NEWP - ARRAYS, DESCRIPTORS AS BY REFERENCE PARAMETERS

Arrays and descriptors as by-reference parameters are distinguished as follows:

1. Formal parameter as a descriptor: If an array row is the actual parameter, a reference to the one-word descriptor for the array row is passed (i.e., an SIRW to the descriptor). In this case, it is the descriptor in itself that is the object and may be directly modified.
2. Formal parameter as an array: The reference passed is to the data segment of the array (i.e., a COPY descriptor is passed directly). The fact that a descriptor is used should be of no interest to the programmer.

D3626 NEWP - RESIZING "EVENT ARRAYS"

EVENT ARRAYs are now allowed to be resized. The syntax is analogous to that for an array row. The following are the changes to the syntax of the RESIZE statement on Page 5-91 of the ALGOL Reference Manual (Form No. 5001639):

```
<resize statement> ::= RESIZE(<resize array identifier>,
                                <arithmetic expression> <retain old>)
<resize array identifier> ::= <array row> | <event array row>
<event array row> ::= <event array identifier> |
                     <event array identifier> [<row designator>]
```

RETAIN must be specified for EVENT ARRAYs. Also, if the new size is not greater than or equal to the old size, a run-time error will occur.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

NEWP

P2733 NEWP - IMPROVE "FILE/LIBRARY" DECLARATION HANDLING

The NEWP compiler will no longer fault with an INVALID OP when a file identifier with an attribute list is address-equated, nor when a direct file identifier is address-equated. Also, redundant commas following a file or library identifier in a declaration list will now cause a syntax error. Finally, the DIRECT attribute will be placed in the FPB for a direct file before the attributes supplied by the user. In this way, other attributes related to a direct file may be properly set by the user.

P2743 NEWP - BETTER LISTING FOR MODULAR "SEPCOMP"

Several improvements have been made in the listing produced for SEPCOMP of a modular program. Previously, error messages and additional information (such as \$STACK) were improperly merged with the final listing when the messages related to a module head. Also, listing of source lines from a module head was not possible. These problems with modular SEPCOMP listings have been corrected.

P2805 NEWP - ERROR FOR MISSING PROCEDURE

An error is now given, rather than a warning, for a procedure with a forward but no body.

P2806 NEWP - ATTRIBUTE FOR TASK ARRAY ELEMENT

The compiler now recognizes task attributes following a task array element and correctly compiles the expression.

P2807 NEWP - ERROR FOR EMPTY PARENTHESIS

The compiler will give a syntax error if a parameter list is not found when a left parenthesis follows the declaration of a procedure identifier.

P2934 NEWP - "IXREF" ENVIRONMENTS FOR CHEAP BLOCKS

The NEWP compiler now generates the correct block environment information in its XREFFILES output files. This allows SYSTEM/INTERACTIVEXREF to more correctly describe NEWP programs.

P2946 NEWP - REDUCED TIME FOR "NEWP" "XREF"

The amount of time (processor and I/O) needed to produce an XREF listing has been significantly reduced.

P2947 NEWP - PREVENT "INVALID INDEX"

The NEWP compiler will no longer get an INVALID INDEX when very long expressions are compiled or when an MCP that is very large is being compiled without resetting \$NOBINDINFO.

P3057 NEWP - ADDRESS EQUATION TO UNDECLARED IDENTIFIERS

An error will now be given when a variable is address-equated to another variable that has occurred in an export list but has not been declared.

P3082 NEWP - ERROR FOR DUPLICATE CASE ELEMENTS

An error, rather than a warning, will be given for duplicate numbered case statement elements.

Example:

```
CASE I OF
  BEGIN
    1: <STATEMENT>;
    1: <STATEMENT>;      % SYNTAX ERROR
  END;
```

P3087 NEWP - "SEPCOMP" LOSES SOURCE LINES

There were several situations in which SEPCOMP would lose the first source line of a region. The compiler has been improved to correctly handle these situations. All of these situations involved patches which fell between or touched the boundaries of regions.

P3140 NEWP - PREVENT COMPILER "SEG ARRAY" FAULT

The compiler will no longer get a SEG ARRAY fault compiling programs with long library export lists.

P3297 NEWP - COMPARING POINTER

When a short (less than 7 characters) ASCII string constant was compared to a pointer (e.g., PTR=7"ABCD"), the compiler generated the literal for the string constant incorrectly. This problem no longer occurs.

P3344 NEWP - STACK OVERFLOW

NEWP no longer faults with a stack overflow when handling many \$OMITs, \$VOIDs, or \$VOIDTs within a module.

P3470 NEWP - "SEPCOMP" CREATES ERRONEOUS STACK ITEMS

NEWP no longer creates erroneous stack items when additions are interspersed in a declaration list.

Example:

```
REAL
  A,
  B,
  X,    % ADDED BY SEPCOMP
  C,
  Y,    % ADDED BY SEPCOMP
  D;
```

Previously, the above condition caused the compiler to fault.

P3627 NEWP - NULL ENVIRONMENTS IN "XREF"

Under some circumstances, the interactive XREF program functioned incorrectly in using a procedure name to specify a range. The problem, due to null entries made by the compiler, has been corrected.

P3628 NEWP - EXTRANEIOUS "XREF" ENVIRONMENTS

The compiler was improperly treating FOR and THRU loops as separate environments. This no longer occurs.

P3629 NEWP - "<ARITHMETIC EXPRESSION> IN <TABLE POINTER>"

NEWP now generates proper code for the following Boolean expression:

```
<arithmetic expression> IN <table pointer>
```

DOCUMENT CHANGES NOTES (D NOTES)

NSP DUMP ANALYZER

D3430 NSPDUMPANALY - "NSP" DUMP ANALYZER IMPLEMENTATION

SYSTEM/NSPDUMPANALYZER is a program which accepts a Network Support Processor (NSP) dump file as input (as produced by the B6900 Data Comm subsystem) and produces a printer file containing an analysis of the dump as output.

This program is intended to be used by Burroughs' personnel for the support of the B6900 Data Comm Subsystem and by customer personnel for the generation of supporting documentation for Field Trouble Reports (FTRs) regarding the B6900 Data Comm subsystem.

It may be run by using the following WFL command:

RUN *SYSTEM/NSPDUMPANALYZER; VALUE=<NSP unit number>

The file *DUMP/NSP/<NSP unit number> will be analyzed.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES).

PATCH

D3007 PATCH - PATCH NUMBERS WITH "\$.VERSION/CYCLE"

PATCH now allows 4-digit patch numbers when using the \$.VERSION and \$.CYCLE options. The mark field in the record will contain the version number (2 digits), the cycle number (3 digits) and the patch number (4 digits) with no separating periods. Column 90 will contain a blank. When the mark field is listed, PATCH will insert periods between the fields for readability.

D3034 PATCH - NEW LIST AND COMPARE OPTIONS

Several facilities have been added to SYSTEM/PATCH to make it more convenient to study the effect of merging several patches applied to one source symbolic. Briefly summarized, the changes are:

The COMPARE option will now flag instances where lines from two different patches are adjacent or interleaved at the same point in the source.

It is now possible, by setting COMPARE or a new option LISTD, to list the patch delimiter (\$#) cards but not the body of the patches.

A new control option, FLAG, may be used to cause flagging of any line in the source whose patchmark version.cycle is greater than a specified value.

Also, a minor error has been corrected which caused the patch mark to be incorrectly shown as blank on a deleted line in the BRIEF summary of a long deletion.

The following changes should be made to the System Software Operational Guide, Volume 1 (Form No. 5001563):

Page 10-1-3 The description of the \$. COMPARE option should now read as follows:

\$. COMPARE

This option causes SYSTEM/PATCH to print a report comparing the PATCH input with the TAPE file. All the patch cards are listed. Source lines from TAPE are shown when they are deleted by the patch or when they appear just before or just after a newly inserted line. Each line in the patch is identified in the rightmost column of the report: simple patch lines bear the ordinal patch number, while moved, resequenced or inserted lines are identified with original patch number (or as source) and the modifying patch number. If lines from more than one patch fall adjacent to each other, with no intervening unmodified TAPE source lines, the final column is prefixed with a ">" symbol. (Such "second-order conflicts" often require investigation.) Because all \$ cards from the patch decks are shown in the compare listing, the same flags will appear if one patch inserts into an area voided by an earlier one. (If \$.BRIEF is set, listing of deleted sections of the TAPE file are abridged.)

COMPARE may be SET, RESET or POPped; the default is RESET. A compare listing will be produced only if COMPARE was last left SET. However, any time COMPARE is SET while LISTP is RESET during the input phase, patch delimiter cards are listed as though LISTD were set."

The description of two new control cards, \$.FLAG and \$.LISTD should be added as follows:

\$. FLAG

Syntax

FLAG option ::= \$. FLAG <version number> <cycle part>
(<Version number> and <cycle part> are the same as for \$.VERSION.)

Semantics

When the source file contains patchmarks in the form of version and cycle numbers, the FLAG option may be used to call attention (in the compare listing) to neighboring or deleted lines with marks equal to or greater than the specified value. Specifically, those source lines whose marks begin vv.ccc or vvccc (where v and c represent digits) will be flagged if vv exceeds <version number> or vv equals <version number> and ccc is as great as <cycle number>. If no <cycle part> is provided, <cycle number> defaults to zero. The flag consists of an asterisk ("*") preceding the mark field.

SET or RESET context is ignored for FLAG. If more than one FLAG value is provided, the last one specified is used throughout the compare phase; FLAG is ignored if COMPARE is not set and is not fully effective if BRIEF is set."

\$.LISTD

If LISTD (or COMPARE) is set and LISTP is reset, each patch delimiter (\$#) card and certain control (\$) cards will be listed. Each delimiter card is shown with the ordinal "patch number" assigned to that patch; this is the number used in the conflict and compare listings to refer to individual input patches. The following control cards are listed if LISTD or COMPARE is set when the control card is processed: \$.BRIEF, \$.CYCLE, \$.FLAG, \$.VERSION. LISTD may be SET, RESET or POPped. The default value is RESET."

D3035 PATCH - "VERSION" MAY BE "RESET"

The construct \$.RESET VERSION now negates the effect of any \$.VERSION or \$.CYCLE specifications. No numbers may follow the keyword VERSION in a RESET context.

Page 10-1-12 of the System Software Operational Guide, Volume 1 (Form No. 5001563) should be changed as follows:

The syntax for the VERSION option should now read as follows:

```
"VERSION option ::= $.VERSION <version number> <cycle part>
                    | $.RESET VERSION"
```

Add the following sentence at the end of the Semantics:

"\$.RESET VERSION restores the default mechanism, as though no \$.VERSION or \$.CYCLE had appeared."

D3446 PATCH - HANDLING OF "\$" CARDS

The B7000/B6000 System Software Operating Guide, Volume 1, (Form No. 5001563-001) should be modified as follows: On page 10-1-2, add the following new paragraph following first paragraph.

"SYSTEM/PATCH requires that the compiler options SEQ, VOID and VOIDT be in a "RESET" state at the end of each patch. This requirement is unlike the compilers that allow these options various to remain "SET" at the time the end of the program text (or EOP) is encountered."

D3447 PATCH - "\$. MARK" OPTION VS \$CARDS

The B7000/B6000 System Software Operational Guide, Volume 1 (Form No. 5001563-001), should be modified as follows: on page 10-1-10, add the following as the last sentence to the first paragraph

"The information retained by \$.MARK is not stored into card images which contain a dollar(\$) in column one."

D3569 PATCH - CLARIFY ADDITIONAL "SYSTEM/PATCH" RULES

The SYSTEM/PATCH description in SOG Reference Manual, Volume 1 (Form No. 5011661), should be revised by adding the following after the first paragraph on Page 10-3-2:

"SYSTEM/PATCH enforced the following rules for sequence number and some \$ cards.

- a. Within a patch (delimited by \$# cards), all records not being resequenced must occur with increasing sequence numbers. Records which occur while \$SEQ is set or when using \$.INSERT (syntax 2) or \$.MOVE (syntax 2) are not checked for the order of their sequence numbers.
- b. SYSTEM/PATCH requires that the \$ options SEQ, VOID, VOIDT be in a RESET state at the end of each patch."

D3599 PATCH - "MARKBLANK" AND "DELIMOPT" OPTIONS

Two new options have been provided for SYSTEM/PATCH. The following documentation may be added to the "\$. Cards" section of SOG Reference Manual, Volume 1, (Form No. 5011661), Chapter 10.

\$.MARKBLANK

Syntax

\$. MARKBLANK

Semantics

The MARKBLANK option provides a conditional form of MARK: If columns 81-90 of the input card image contain blanks, the Mark Level information is inserted as for MARK, but if columns 81-90 already contain non-blank data, the field contents are retained. (Any card read from a file with MAXRECSIZE < 15 words or 90 characters is automatically blanked in columns 81-90, so in this case MARKBLANK is equivalent to MARK.)

The MARKBLANK option may be SET, RESET or POPped at any point in the input deck, and is effective for all normal input cards read until the option is changed. The default value is RESET. If both MARKBLANK and MARK are SET, MARKBLANK is effective.

\$.DELIMOPT**Syntax****\$. DELIMOPT****Semantics**

The DELIMOPT option provides a mechanism to insulate succeeding patches from option changes in a particular patch. It causes certain options to be restored whenever a \$# patch delimiter card is read. Specifically:

Whenever DELIMOPT appears in on a \$. card in a SET (or default) context, the option is set and the current value is recorded for each of the following options:

BCL CONFLICT LISTP MARK MARKBLANK OUT

Then whenever a \$# patch delimiter card is read while DELIMOPT remains SET, all these options are restored to the values they had when DELIMOPT was last SET. (The stacked historical values of these options are RESET; thus a POP is equivalent to a RESET for an option like LISTP that has been restored by a \$# card read while DELIMOPT was set.)

DELIMOPT may be SET, RESET or POPped at any time; the default is RESET. If DELIMOPT is POPped from a RESET to a SET state, the option values to be restored remain those from the most recent occasion when DELIMOPT was SET.

Pragmatics

The simplest way to use DELIMOPT is to SET it as the last option before the first \$# delimiter in the input set. Then the specified or default values of the several options will apply to each patch in turn, even if a prior patch had changed one or more option.

Example

```
$.SET COMPARE MARK DELIMOPT
$#PATCH ONE
$.MARKBLANK
...
$#PATCH TWO
$.RESET MARK
...
$#PATCH THREE
...
```

MARKBLANK will apply to the input for patch one but not patch two or three. Input in patch two will be unmarked, but the MARK specification will apply again to patch three.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

PATCH

P2715 PATCH - CHARACTER MODE FILES

PATCH will now handle character mode files (i.e., files with UNITS=CHARACTERS). The entire record area (up to 90 characters) will be read rather than the first 15 characters.

P2716 PATCH - OUT OF SEQUENCE PATCHES

PATCH now detects out of sequence patches that have \$VOID cards in them.

P2717 PATCH - "\$.GUARD" OPTION

PATCH will no longer print garbage lines in the GUARDED output listing when \$.GUARD is used and there are \$ cards with no sequence numbers on them.

P3630 PATCH - LINE WIDTH

PATCH now correctly calculates the input line width for BCL and/or character-mode files.

DOCUMENT CHANGES NOTES (D NOTES)

PATCHCONTROLWARE

D3128 PATCHCW - INITIAL RELEASE OF "PATCHCONTROLWARE"

SYSTEM/PATCHCONTROLWARE is a utility to patch Controlware files. It may be initiated from an ODT or remote terminal. The internal file name CONTROLWARE may be label-equated to the Controlware file to be patched. If the Controlware file is not found, the utility will request the file name to be entered.

Patches are entered, one at a time, using the following format:

```
CC PATCH [/DISP] [/PREV] <FILENAME> <SEGMENT> <ADDRESS-1>
      <ADDRESS-2> <LENGTH> [UA] [UN] <DATA>
```

where:

```
[...] denotes optional input
/DISP causes <DATA> to be displayed/printed
/PREV causes patched data to be displayed/printed
<FILENAME> ::= name of file to be patched
<SEGMENT>  ::= zero relative segment address
<ADDRESS-1> ::= zero relative memory address
<ADDRESS-2> ::= zero relative digit address in segment
<LENGTH>   ::= length of patch (maximum value of 30)
UA          ::= alpha data follows
UN          ::= numeric/hex data follows
<DATA>     ::= the patch itself
<SEGMENT>, <ADDRESS>, and <LENGTH> are in decimal
```

Enter STOP to terminate the program.

PLI

D3077 PLI - "TRANSLATE BIF"

Page A1-12 of the PL/I Language Reference Manual (Form No. 5001530), in the section about the TRANSLATE BIF, should be corrected as follows:

REFERENCE Paragraph

The REFERENCE paragraph should read as follows:

"Reference: TRANSLATE(S,R[,M])"

ARGUMENTS Paragraph

Add the following sentence to the end of the ARGUMENTS paragraph:

"If M is omitted, COLLATE() is assumed as its value."

D3172 PLI - "REWRITE" STATEMENT

The following corrections should be made to the PL/I Language Reference Manual (Form No. 5001530):

1. Page 4-37

The last sentence under "Semantics" should be changed to read as follows:

"To access such a file, the sequence of statements should normally be READ followed by REWRITE."

2. Page 7-45

Add the following new paragraph under "Semantics":

"For a SEQUENTIAL UPDATE file, consecutive REWRITES, without READs, will increment the record pointer."

D3218 PLI - "CONVERSION" RETRY

The PL/I Language Reference Manual (Form No. 5001530), Page A2-2, should be changed as follows:

1. In the last sentence of the first paragraph under "CONVERSION:", the literal ", and the" should be changed to ". The".
2. The following should be added to the end of the second paragraph under "CONVERSION:":

"Note that expressions which are not variables, or variables which are procedure entry parameters, based, or overlay defined, may not be reevaluated after an on-unit. Therefore, changing the value of such an expression in an on-unit may not affect the conversion retry."

D3307 PLI - EXPONENT ON "E" FORMAT OUTPUT

The following changes should be made to the PL/I Language Reference Manual (Form No. 5001530), Page 8-23, under "Output Semantics":

1. The sentence which reads "The <exponent> is an integer constant of N digits" should be changed to read as follows: "The <exponent> is an integer of 5 digits."
2. The last paragraph should be changed as follows:

"S+N+3" should be changed to "S+8"

"S+N+4" should be changed to "S+9"

"S+N+2" should be changed to "S+7"

D3352 PLI - MODIFICATIONS TO SUPPORT PORT FILES

The following changes to PL/I have been made to support multi-subfile port files. (See Mark 32 GENERAL note D3650, "Implementation of Port Files", for a description of port files.)

1. The SIGNAL and RESPONSE elements of READ and WRITE statements have been deimplemented.
2. The syntax of READ, WRITE, OPEN and CLOSE statements is appended with an optional subfile identification clause.

New SUBFILE Clause Syntax:

-- SUBFILE -- (--<scalar-expression>--) --|

A SUBFILE clause has no effect on OPEN and CLOSE statements when a file's KIND is not PORT. However, execution of a READ or WRITE statement with a SUBFILE clause for a file with KIND not equal to PORT raises the UNDEFINEDFILE condition.

The value of the expression in the SUBFILE clause determines which subfile in a port file is to be read, written, opened or closed. A value of zero refers to an entire port file: a READ will read from any subfile, a WRITE will write to all subfiles, and an OPEN or CLOSE will open or close all subfiles. A READ, WRITE or OPEN with an invalid subfile index raises the UNDEFINEDFILE condition. A CLOSE with an invalid subfile index is ignored.

If the subfile index specified in a CLOSE statement is zero, all open subfiles will be closed. If the subfile index is greater than zero but not greater than MAXSUBFILES, only the specified subfile is closed. If no subfile index is specified and MAXSUBFILES is greater than one, an UNDEFINEDFILE condition is raised; if MAXSUBFILES is equal to one, the (only) subfile will be closed.

For READ statements, if the expression in a SUBFILE clause is a variable or pseudo-variable, after the read the subfile number of the origin of the record read is stored in the variable or pseudo-variable.

3. An optional DONTWAIT clause has been added to the syntax for READ and WRITE statements.

New DONTWAIT Clause Syntax:

```
-- DONTWAIT -- ( --<scalar-expression>-- ) --|
```

The expression is converted to the type BIT(1). A value of '0'B, or absence of the DONTWAIT clause, causes program execution to wait until the read or write is finished. If the value is '1'B, a read is done only if data is ready to be read, and a write is done only if a buffer is available.

If the expression in the DONTWAIT clause is a variable or pseudo-variable, after the read or write the following is stored in the variable or pseudo-variable:

```
'1'B  if the read or write was not done because no data
      was ready to be read or no buffer was available,
'0'B  if the read or write was done.
```

Execution of a READ or WRITE statement with a DONTWAIT clause for a file with KIND not equal to PORT raises the UNDEFINEDFILE condition.

4. Three additional option keywords are allowed in OPEN statements:

```
WAIT
AVAILABLE
OFFER
```

WAIT is the default.

5. A new close option, DONTWAIT, is allowed for port files. This option causes processing to continue without waiting for the file to be closed.
6. A WAIT statement has been added to PL/I.

WAIT Statement Syntax:

```

      |<----- , -----|
-- WAIT -- ( ---<event-valued-attribute>--- ) ----->
>-----|
      | - EVENTNO -- ( --<scalar-variable>-- ) -|
```

The statement causes suspension of the task until one of the listed event attributes is caused. If the EVENTNO clause is used, the ordinal position of the event attribute which caused activation of the task is stored in the variable. The variable may be a pseudo-variable.

D3511 PLI - INCLUDED COMPILE TIME PROCEDURE

The following sentence should be added to the end of Section 11.7.7 of the PL/I Reference Manual (Form No. 5001530):

"If a compile-time procedure in an include file is to be called from the containing text, the containing text must have an entry declaration for the procedure."

D3519 PLI - ARRAY ELEMENTS AS SORT KEYS

Section 7.3.2.8 of the PL/I Reference Manual (Form No. 5001530), should be revised as follows:

Add the following to the "Restrictions" listed:

"Indexed array elements cannot be used as keys."

Delete the following from the example:

"DCL AR(10);"

The second statement from the list of legal sort statements.

D3546 PLI - BINDING PROGRAMS WITH "DUMP" OPTION

Binding of dump information in PL/I programs has never been implemented. When a binder-produced codefile calls the PLI PROGRAMDUMP intrinsic, the intrinsic gets an error because no dump information is in the codefile. To ensure that this error does not occur, setting the PL/I compiler option DUMP will automatically set NOBINDINFO. Resetting NOBINDINFO when DUMP is set will cause a syntax error.

D3559 PLI - "ONCHAR" , "ONSOURCE"

The PL/I Reference Manual (Form No. 5001530), should be corrected as follows:

On Page A1-31, add the following sentence to the paragraph about the results of the ONCHAR builtin function:

"If the conversion condition occurred during a stream I/O statement, using ONCHAR in or after a stream I/O statement in the on-unit will produce undefined results."

On Page A1-33, add the following sentence to the paragraph about the results of the ONSOURCE builtin function:

"If the conversion condition occurred during a stream I/O statement, using ONSOURCE in or after a stream I/O statement in the on-unit will produce undefined results."

D3560 PLI - "ONLOC" BUILTIN FUNCTION

On Page A1-33 of the PL/I Reference Manual, (Form No. 5001530), Add the following sentence to the end of the paragraph about the results of the ONLOC builtin function:

"ONLOC is not presently implemented."

D3561 PLI - "UNSPEC" PSEUDO VARIABLE

On Page A1-38 of the PL/I Reference Manual (Form No. 5001530), add the following sentence to the end of the paragraph about the UNSPEC pseudo-variable:

"V may not be a double-precision arithmetic item."

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

PLI

P2718 PLI - EXPONENTIATION MIXING OPERANDS

The PL/I compiler neglected to generate code to extend single precision operands for performing exponentiation between single and double precision operands.

This problem has been corrected.

Example:

```
DCL SP BIN FLOAT (30),
      DP BIN FLOAT (60);
DP=DP**SP; DP=SP**DP;
```

Both expressions previously gave incorrect results; now, they both give correct results.

P2719 PLI - CORRECT "ATAND"

The ATAND function returns its value in radians rather than degrees when called with a single operand. This problem has been corrected.

P2720 PLI - DOUBLE PRECISION "PICTURE 'H'"

When PICTURE 'H' data items were used as arithmetic operands, they were not properly extended to double precision when their values were less than 8×13 .

This has been corrected.

Example:

```
DCL (A,B,C) PIC '(13)H';
A=65258542947; B=514511850145;
C=A+B;
```

Previously, the value of C was incorrectly calculated to 579770393096; now it is properly calculated to 579770393092.

P2721 PLI - "SUBSTR" OF BINARY DATA

SUBSTR converted binary data to a character string instead of a bit string.

This has been corrected.

Example:

```
DCL (X,Y) BIN FIXED;
Y=SUBSTR(X,6,6);
```

Previously, X was converted to character, resulting in a "STR-TO-ARITH CONVERSION" error; now, X is properly converted to a bit string.

P2722 PLI - BRANCHING FROM START OF SEGMENT

Branches from the first syllable of code segments were not always generated correctly.

Example:

```
P:PROC;
  DCL A(5);
  GO TO Q;
  PUT EDIT (A(1)) (F(3));
Q:END;
```

This program previously produced an INVALID OP because the GOTO statement was not generated in the proper place in the code; the problem has been corrected.

P2767 PLI - SYSTEM FILE ATTRIBUTE PARAMETERS

The syntax for run-time references of system file attributes, for both reading and setting, has been changed to include a place for two optional parameters.

Old Syntax:

```
<attribute name> (<internal-file-name>)
```

New Syntax:

```
<attribute-name> (<internal-file-name>
[,<scalar-expression> [,<scalar-expression>]])
```

The two new parameters will be interpreted as follows: If KIND='REMOTE', parameter one will identify STATION number and parameter two will not be allowed; if KIND='PORT', parameter one will identify subfile index and parameter two will not be allowed; if KIND='DISK' or 'PACK', one parameter will refer to row number, and two parameters will refer to row number followed by copy number.

P2769 PLI - STATEMENT NUMBERS IN ERROR MESSAGES

The statement numbers and record numbers in error messages displayed in CANDE from pass one of the PL/I compiler pointed to the statement before the one to which the message was addressed. Now, the correct statement numbers are given.

P2808 PLI - PRECEDENCE OF OPERATORS

Previously, operations of the same priority were always performed from left to right. Now operations of the same priority are performed from left to right except when they are of the highest priority (not, **, prefix -, prefix +), in which case they are performed from right to left.

Examples:

Expression	Was Interpreted	Now Interpreted
A**B**C	(A**B)**C	A**(B**C)
-A**B	(-A)**B	-(A**B)
A**-B**C	(A**(-B))**C	A**(-(B**C))

This is in accordance with the PL/I Language Reference Manual and ANSI Standards.

P2809 PLI - DUPLICATE LABEL AND ENTRY NAME

The PL/I compiler looped when a statement label was followed by an identical procedure on the same level; e.g.,

```
... L: ... ; L:PROC; ... ; END ; ...
```

This problem has been corrected.

P2810 PLI - "PIC'(12)HS'"

The PL/I compiler generated incorrect code for storing a value in a field declared PIC'(12)HS'. This problem has been corrected.

P2840 PLI - "OR" OPERATION ON "BDMS" FIELD BITS

When a bit from a BDMS item defined as FIELD (BIT1,BIT2...) was ORed with a data item defined BIT(1), incorrect code was sometimes generated by the PL/I compiler.

Example:

```
DCL B BIT(1); B='1'B; DATASET.FIELD.BIT1='0'B;
IF DATASET.FIELD.BIT1 |~B THEN
  DISPLAY ('THIS SHOULDN'T HAPPEN');
```

Previously, the IF statement gave the wrong results; this problem has been corrected.

P2841 PLI - "[:=" AS ASSIGNMENT OPERATOR

Previously, the use of "[:=" as an assignment operator caused the compiler to loop. The problem has been corrected; now, an error message is given.

P2850 PLI - UNDEFINED FORMAT

Previously, the PL/I compiler aborted with an INVALID INDEX when an undeclared format was referenced by a GET EDIT statement.

Example:

```
GET EDIT(A) (R(<format label>));
```

The example caused the INVALID INDEX when <format label> was not defined.

The compilation now completes, generating the appropriate error message, without getting an INVALID INDEX.

P2870 PLI - LOGICAL OPERATIONS ON "BIT" STRINGS

The PL/I compiler sometimes generated incorrect code for expressions which used both NOT and OR/AND operations on operands of the same length. The problem occurred only when at least one operand was part of a structure.

Example:

```
DCL 1 S,
      2 (B1,B2) BIT(1);
B1='0'B; B2='1'B;
IF B1 | ~ B2 THEN
.
.
.
returned true.
```

Now, correct code is generated for such expressions.

P2907 PLI - STACK CELL FOR "ELSE"

The PL/I compiler erroneously allocated a stack cell for an ELSE if the ELSE were followed by a condition prefix.

Example:

```
IF FALSE THEN; ELSE (SIZE): I=I+1;
```

The stack cell is no longer allocated.

P2911 PLI - "SORT" COMPARES ON PICTURED KEYS

Previously, the PL/I compiler generated a character compare for all PICTURED SORT KEYS. Now, all SORT KEYS with PICTUREs using S, +, -, CR, DB, T, I, R, E, K, l or an odd length of H cause an evaluation and arithmetic compare.

Note that use of such keys, particularly type 9, will cause slower sorts because of the PICTURE-TO-ARITHMETIC conversion. PICTUREs using A or X, or PICTUREs which are type 9 and do not use the above listed symbols, will continue to cause generation of characters compares in SORTs.

P2912 PLI - "LABELTYPE='OMITTED'"

When the file attribute LABELTYPE was assigned 'OMITTED', the PL/I compiler caused the attribute to be set incorrectly to 'OMITTEDEOF'. Now the correct attribute assignment is made.

P2935 PLI - MORE THAN 48 "%DO" STATEMENTS

The PL/I compiler would get an INVALID INDEX when more than 48 %DO statements occurred, even if they were not nested. Now, the compiler handles any number of non-nested %DO statements.

P2982 PLI - INDEPENDENT TASK INITIATION

PL/I's facility for initiating an independent task did not work properly, as follows:

1. A CALL of a TASK with OPTIONS(INDEPENDENT) and no parameters caused the compiler to emit code which resulted in an INVALID OP.
2. The data descriptor to bindinfo for an EXTERNAL ENTRY was generated with an incorrect length field.

These problems have been corrected.

P3006 PLI - POINTER INITIALIZATION

Initialization of a pointer with NULL(), as shown in the following example, caused a PL/I compiler error, even though the pointer initialization worked correctly for other values:

```
DCL P POINTER INITIAL(NULL());
```

Now, a pointer can be initialized with NULL().

P3016 PLI - IGNORED "LENGTH" OR "INITIAL" SPECIFICATIONS

Previously, no warning message was generated when the PL/I compiler ignored LENGTH or INITIAL specifications on group items.

Example:

```
DECLARE 1 A CHAR(10),
        2 B CHAR(20),
        2 C CHAR(30);
```

The "CHAR(10)" following "A" was ignored by the compiler.

Now, a warning message is generated when LENGTH or INITIAL attribute specifications are ignored.

P3017 PLI - STRING BUILTIN FUNCTIONS

The following string builtin functions sometimes produced incorrect results when an argument expression contained a string concatenation:

TRANSLATE	BEFORE	AFTER	UPTO
FROM	DECAT	COPY	REPEAT

Example:

```
TRANSLATE(S1,S2||S3,S4)
```

These builtin functions now produce correct results.

P3018 PLI - "DIMENSION" NOT FIRST ATTRIBUTE

When a DIMENSION attribute specification was not the first attribute of a declaration and appeared without the keyword "DIMENSION", the PL/I compiler failed to give a warning message.

Example:

```
DCL A CHAR(1) (10);
```

The example should give a warning message because "(10)" should either immediately follow "A" or be preceded by the keyword "DIMENSION".

An appropriate warning message is now generated.

P3043 PLI - "EXCEPT" BUILTIN FUNCTION

The builtin function EXCEPT did not work correctly if either of its arguments was not a character string, because the arguments were not automatically converted to character strings. Now, arguments are automatically converted to character strings.

P3044 PLI - ILLEGAL PRIMARY

An assignment of a structure to a numeric variable was flagged by the PL/I compiler as both a user error and a compiler error. Now, the compiler error is no longer generated.

P3058 PLI - "PIC 'X'" ARRAY ELEMENTS

Array elements of type PIC 'X' could not be converted to or moved into any data type other than CHAR without causing an INVALID OP. Now, PIX 'X' array items are converted and moved correctly.

Example:

```
DCL A BIN FIXED,
     B PIC'(25)X',
     C(0:3) PIC'(25)X';
A=C(1);
B=C(2);
```

Previously, both of the above assignment statements caused INVALID OPs; now, both work correctly.

P3062 PLI - COMPILETIME "DO" AND "INCLUDE"

When the record number of a %DO in a %INCLUDE file was equal to the record number of a %DO in the source file or another %INCLUDE file, the PL/I compiler sometimes looped. This problem has been corrected.

P3088 PLI - "BIT" COMPARES

The PL/I compiler sometimes generated incorrect code for comparisons of BIT constants with some BUILTIN functions which produce variable length results.

Example:

```
IF '0'B=REPEAT('0',L) THEN;
```

The compiler generated code which caused an INVALID INDEX.

Now, correct code is generated for such comparisons.

P3143 PLI - "PUT EDIT" OF "PIC 1" VARIABLES

Previously, a PUT EDIT of a variable declared PIC'(n)1' caused either incorrect results or an INVALID OP. Now PIC 1 variables are properly handled by PUT EDIT statements.

Example:

```
DCL P PIC'(31)1';
PUT EDIT(P) (E(20,10));
```

The example PUT EDIT statement failed previously; now, it works correctly.

P3211 PLI - PARAMETER MISMATCH WITH "NOBINDINFO"

PL/I compilations with NOBINDINFO set failed to produce information in the codefile which is necessary for initiating a user task. Now, the necessary information is generated.

P3212 PLI - LIBRARY CAPABLE BIT

The library capable bit in the compiler info word in Seg zero was not being set by the compiler. It is now set for all PL/I programs which contain a FREEZE statement.

P3310 PLI - INAPPROPRIATE WARNING MESSAGES

Two warning messages were sometimes emitted by the PL/I compiler when no such warnings were appropriate. The messages were the following:

1. "SCALE FACTOR IGNORED FOR FLOAT ATTRIBUTE (LEVEL 3)" which was sometimes wrongly generated for ENTRY VARIABLE declarations.
2. "LENGTH, PRECISION, OR INITIALIZATION OF GROUP ITEM IGNORED (LEVEL 1)" which was sometimes incorrectly emitted for group items with the DEFINED attribute.

Now, the warnings are given only when appropriate.

P3337 PLI - BIT STRING COMPARISONS

The PL/I compiler will now generate correct code for bit string comparisons of the form.

```
(A=B) = C.
```

P3367 PLI - "PUT EDIT" OF BIT VARIABLES

Statements of the following form, which previously could cause fatal system dumps, now work correctly:

```
PUT EDIT (B) (F(n));
```

where B is a bit variable, and n is an integer.

P3471 PLI - ERROR "TASK IDENTIFIER REQUIRED"

The error "TASK IDENTIFIER REQUIRED" is now always a level 9 error; thus, the compiler no longer generates a code file when the error is detected.

P3472 PLI - BIT STRING DEFINED

When overlay defining is used, the POSITION attribute is in bit units if the defined item is a bit string; otherwise, it is in character units. Previously, when a bit string was defined over a character string, the compiler incorrectly syntax-checked the POSITION attribute as if it were in character units. Now, the POSITION attribute is syntax-checked in bit units as it should be.

P3473 PLI - BIT OVERLAY DEFINING

On the Mark 31 release of the PL/I compiler, defining a bit string over a bit string sometimes caused incorrect syntax errors or incorrect results.

Example:

```
DCL A BIT (16);
DCL B BIT (8) DEF A POS (5);
```

The above example now compiles correctly. The above example also compiled correctly on the Mark 30 release, but compiled incorrectly on the initial Mark 31 software.

P3474 PLI - "WRITE" WITHOUT "FROM"

On the Mark 31 release, the PL/I compiler began emitting an error message for WRITE statements which did not have a FROM option. WRITE SKIP and WRITE PAGE without the FROM option had previously worked properly; however, on the Mark 31 release, they were erroneously deimplemented, and caused the previously-described error message. Now, the error message is emitted only if a WRITE statement has no FROM, SKIP nor PAGE option. WRITE SKIP and WRITE PAGE now work properly as they did on the Mark 30 release.

P3475 PLI - COMPILER LOOP CORRECTED

If the last item on a line ended in column 72, and was part of a statement, and the next line were blank, and LIST2 were set, the PL/I compiler would loop indefinitely. This no longer occurs.

P3476 PLI - BAD DECLARATION CAUSED "INVALID INDEX"

If the first procedure declaration of a PL/I program was not preceded by an entry name and a colon, the PL/I compiler would sometimes get an INVALID INDEX. Now, the proper error message is emitted and an INVALID INDEX does not abort the compile.

P3477 PLI - LOST TEXT FROM COMPILE-TIME PROCEDURE

When a compile-time procedure was invoked without parameters and without an empty set of parenthesis, and the invocation was immediately followed by a comment, part of the source text following the comment was sometimes lost by the PL/I compiler.

Example:

```
COMPILETIMEPROC /* COMMENT */
PUT DATA;
```

The above example now compiles correctly; previously, the "PUT" was lost by the compiler's scanner.

P3510 PLI - QUESTION MARK

A question mark in column 1 or 2 of a PL/I program should terminate a compile (as described in Mark 29 note D2120); however, the PL/I compiler was erroneously terminating at the occurrence of a question mark anywhere when not included in a comment or literal. Now, a question mark not in columns 1 or 2, not contained in a comment or literal, is treated as an invalid character, and does not terminate the compile. A blank is assumed in its place, and a warning message is emitted.

P3511 PLI - BAD LINEINFO

Previously, when a PL/I program's segment dictionary length exceeded 99 words, bad lineinfo was generated. This no longer occurs.

P3512 PLI - EXTERNAL FILE VARIABLES

Previously, a file variable declared external was syntaxed with "REFERENCE IS EXPECTED" when an attempt was made to assign a file to it. Now, external file variables are implemented correctly.

P3513 PLI - "LENGTH(String(<ID>))" IN SUBSTR ARGUMENT

If the second argument of the SUBSTR built-in function were an expression containing the following:

```
LENGTH(STRING(<exp>))
```

and <exp> had a constant length, incorrect code was sometimes generated by the PL/I compiler. This problem has been corrected.

P3548 PLI - "PICTURE Y"

When a picture contains leading-zero-suppression ('Z' or '**') followed by unconditional-zero-suppression ('Y'), a zero in the digit referenced by the first Y of the picture was sometimes not changed to a blank. Now, mixing leading and unconditional zero-suppression works correctly.

P3549 PLI - "ROUND" BUILTIN FUNCTION

Use of a complicated expression, rather than an integer constant, as the second argument to the ROUND builtin function sometimes caused the PL/I compiler to loop indefinitely. This no longer occurs.

P3550 PLI - LEVEL "3" WARNING MESSAGE

When the PL/I compiler's estimate of the memory requirements of a program exceeds the amount of space allotted to PL/I programs, the following level-3 warning message is now emitted:

```
"ESTIMATED MEMORY REQUIREMENTS EXCEED ALLOTTED SPACE"
```

P3551 PLI - PROBLEM WITH PICTURES

An inconsistency in the interface between the PL/I compiler and the PL/I INTRINSICS, which was introduced by a patch in the Mark 31 PR1 compiler dealing with PICTURED sort keys, has been corrected. The problem was most evident in PUT EDIT of PICTURED variables.

P3593 PLI - LARGE STRUCTURES WITH "INITIAL" ATTRIBUTE

If more than 155 data items of the same level were declared in one structure, each with an INITIAL attribute, the PL/I compiler faulted with an INVALID INDEX. The compiler no longer faults in this situation.

P3752 PLI - BIT EXPRESSIONS

Expressions of the following form were sometimes evaluated to incorrect results:

<bit expression> &~ B

such that B is a BIT(1) item in a structure or data set.

Now, correct code is generated by the PL/I compiler, so that results of such expressions are always correct.

DOCUMENT CHANGES NOTES (D NOTES)

PLISUPPORT

D3374 PLISUPP - "SYSTEM/PLISUPPORT"

The file SYSTEM/PLISUPPORT has been implemented to provide run-time support for PLI in future releases.

D3394 PLISUPP - IMPLICIT OPENING OF FILES

On releases Mark 31 and earlier, the MYUSE attribute of a file was unchanged by implicit file opens. Now, in accordance with ANSI standards, implicit file opens will have the following effects on the MYUSE attribute:

GET: MYUSE will be INPUT.
PUT: MYUSE will be OUTPUT.
READ: If MYUSE is not already UPDATE, it will be changed to INPUT.
WRITE: If MYUSE is not already UPDATE, it will be changed to OUTPUT.
REWRITE: MYUSE will be UPDATE.
DELETE: MYUSE will be UPDATE.
LOCATE: MYUSE will be OUTPUT.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

PLISUPPORT

P2723 PLISUPP - INCORRECT CONVERSION CONDITION

A conversion condition occurred when picture fields such as PIC 'ZZV.ZZ' were evaluated if they held a value between 1 and 0. The problem has been corrected.

Example:

```
DCL P PIC 'ZZV.ZZ';
P=0.5;
A=P;
```

The last statement previously raised the conversion condition; now, the statement is properly executed.

P2872 PLISUPP - "CHAR" TO "ARITH" CONVERSION

When converting character strings into numeric values, the CONVERSION CONDITION was not raised when a plus or minus sign appeared out of place. The following are examples of strings which should have raised the CONVERSION CONDITION, but did not:

```
'1234-'
'12+-23'
'-1+'

```

Now, the CONVERSION CONDITION is raised whenever plus or minus signs appear in the wrong places.

P2908 PLISUPP - "GET DATA" CROSSING RECORD BOUNARIES

A GET DATA statement would produce incorrect results if a number to be read crossed a record boundary.

Example:

```
X = 123456
```

The record boundary occurred between 3 and 4; consequently, an incorrect value was assigned to X.

The problem has been corrected; now, a GET data statement will get correct results when data items cross record boundaries.

P3213 PLISUPP - "ISAM, DELETE"

When using COBOL with \$ANSI74 set, a sequential DELETE sometimes failed to remove the deleted record's key from the Fine and Coarse tables. Now, the keys are deleted from the Fine and Coarse tables.

P3214 PLISUPP - "ISAM," SECURITY ERROR

The following sequence of events sometimes caused a security error (IO ERROR #19):

1. Update the ISAM file.
2. Close the file.
3. Change the security of the ISAM file to CLASSA IN.
4. Attempt to open the file for INPUT and read from the file.

The security error will no longer occur.

P3215 PLISUPP - "ISAM," DUPLICATE RECORD KEYS

Since the 29 PR1 release, writing of records with duplicate keys to ISAM files opened UPDATE often produced incorrect links in the ISAM file. Now, correct links are maintained.

P3216 PLISUPP - "REWRITE," ZERO IN FIRST WORD

In a PL/I program, REWRITing from a structure with zero in the first word to a keyed file caused the entire record to be zeroed. Now, the REWRITE is done correctly.

P3217 PLISUPP - EMPTY "ISAM" FILE

Closing an empty ISAM file which was opened OUTPUT will now cause an exception condition, as follows:

```
Primitive Method:
    In the returned Boolean, 25:1 will be set to 1.
```

```
COBOL with ANSI74 set:
    FILE STATUS will be set to 23.
```

PL/I: Condition code 2517 - CANNOT CREATE EMPTY KEYED FILE.

P3365 PLISUPP - "ISAM" PARITY ERROR TO "COBOL"

When a parity error occurs on an ISAM file, an incorrect value is no longer returned to COBOL programs.

P3366 PLISUPP - "ISAM" LOGICAL DELETE

If the last record of an ISAM file is logically deleted, a garbage record is no longer returned when the file is read sequentially.

P3367 PLISUPP - "PUT EDIT" OF BIT VARIABLES

Statements of the following form, which previously could cause fatal system dumps, now work correctly:

PUT EDIT (B) (F(n));

where B is a bit variable, and n is an integer.

P3514 PLISUPP - "PL/I" PROGRAMDUMP

The PL/I Programdump intrinsic did not work correctly on the initial Mark 31 release. It now works as it did on the Mark 30 release.

P3515 PLISUPP - "PL/I" PROGRAMDUMP

The PL/I Programdump intrinsic failed when processing a block containing more than 130 identifiers. Now, any number of identifiers in a block are handled.

P3552 PLISUPP - "GET DATA" LOOP

A parenthesis or equal sign out of place in an input stream to a GET DATA statement sometimes caused the GET DATA intrinsic to loop indefinitely. This no longer occurs.

DOCUMENT CHANGES NOTES (D NOTES)

PLI INTRINSICS

D3600 PLINTRN - DELETE OLD INTRINSICS

PL/I was changed on the Mark 30 PR1 release so that the compiler references the MCP's ATTRIBSEARCHER; therefore, ATTRIBSEARCHER has been deleted from the PL/I intrinsics.

D3630 PLINTRN - "PLINTRN" SUBSUMED BY "PLISUPPORT"

Effective with the Mark 32 release, the PL/I intrinsics have been subsumed by the PLISUPPORT library, described in GENERAL note D3354.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

PRINT BINDER INFO

P3438 PRINTBIND - "BINDINFO" FOR ALTERNATIVES

BINDINFO is now generated for MCP ALTERNATIVES. This was added mainly for DUMPANALYZER.

P3596 PRINTBIND - ELIMINATE "EOF NO LABEL" ABORT

PRINTBINDINFO could terminate for an "EOF NO LABEL" error when analyzing what appeared to be an external procedure, but what was really a non-existent entry after the end of the segment dictionary of the program. This no longer occurs.

P3597 PRINTBIND - HANDLING OF PROCEDURE PARAMETERS

PRINTBINDINFO was incorrectly handling "BY REFERENCE" parameters which followed a procedure parameter. This problem has been corrected.

DOCUMENT CHANGES NOTES (D NOTES)

REMOTE JOB ENTRY

D2972 RJE - "PUT/FETCH" RECORD COMPATIBILITY

RJE's PUT/FETCH (COPY) record format is now compatible with non-6000/7000 systems (B1800, B800).

D3036 RJE - PRINT QUEUE REBUILD AT "RJE" "BOJ"

The rebuild of RJE's print queue at BOJ now takes place as an asynchronous process, thus allowing RJE to handle LOGON, job initiation, etc., while the print queue is being rebuilt. Requests for information that use the print queue and REMLP files have been modified in their response during the rebuild.

An *SP request displays an extra line, as follows:

"(PRINT Q REBUILD IN PROGRESS LIST MAY NOT BE COMPLETE.)"

An *BACKUP request is not allowed during the rebuild and displays the following:

"#NOT ALLOWED WHILE PRINT QUEUE REBUILD IN PROGRESS."

Any station that requests information during the rebuild is notified of the rebuild finish by the following:

"#PRINT QUEUE REBUILD FINISHED."

D3037 RJE - *BACKUP REQUESTS RUN ASYNCHRONOUSLY

The *BACKUP RJE request now causes the procedure GETBACKUPFILENAMES to run ASYNCHRONOUSLY with RJE. This allows the other stations in the RJE network to continue working when large numbers of REMLP and REMCP files are involved.

A *BACKUP RJE request while one is in progress results in the following message:

"#BACKUP REQUEST ALREADY ACTIVE."

D3038 RJE - "MCS" NAME DISPLAY CHANGE

The name that is displayed in response to a *WM, RJE BOJ, station LOGON, is now the actual name of the MCS, thus allowing sites to run with multiple copies of RJE with different names.

D3041 RJE - "WORDS" VS. "CHARACTERS" IN FILE TRANSFERS

All MAXRECSIZE and BLOCKSIZE values in RJE's PUT/FETCH records are as if the requested file were of type CHARACTERS. When a file with UNITS EQUAL 0 (WORDS) is requested for transfer, the MAXRECSIZE and BLOCKSIZE are adjusted to the UNITS EQUAL 1 (CHARACTERS) values for transmission. If the receiving host is a B6000/B7000 system and UNITS EQUAL 0, the destination file has its MAXRECSIZE and BLOCKSIZE adjusted back to WORDS. For non-B6000/B7000 destination hosts, there is no record type of WORDS; all files remain in CHARACTERS format.

D3042 RJE - FILE TRANSFER CODE OPTIMIZATION

The readability of the file transfer code in RJE has been improved.

D3043 RJE - "BCL" CONSTRUCTS REMOVED

The use of BCL constructs in the binary punch label translation routine have been replaced by HEX constructs.

D3078 RJE - BLANK "FTS" RECORD AT END OF "FTS" BLOCK

RJE now handles a blank record as the last record of a file transfer block. Also, the occurrence of lower case letters in a FTS record does not cause it to be expanded before transmission.

D3079 RJE - "CHARACTERSPERFTBLOCK"

RJE now uses CHARACTERSPERFTBLOCK in sizing its FILEX and FILER input and output arrays. This allows file transfer blocks up to 2000 characters. It should be noted that CHARACTERSPERFTBLOCK should be the same on both host systems involved in a file transfer.

D3080 RJE - CRUNCHING TRANSFERRED FILES

RJE now only LOCKs with CRUNCH files that are crunched on the source host after being transferred.

D3081 RJE - "LOCK PROGRAM"

RJE now protects itself from accidental DSes by performing a LOCK PROGRAM request during initialization. This causes any ODT request to DS RJE to respond with "PROGRAM IS LOCKED". To cause RJE to go to EOJ, the following command should always be used: "<mix #> SM QUIT". If a DS of RJE is necessary, the program can be unlocked by the following command: "<mix #> LP -".

D3091 RJE - HOST TO HOST "LOGON" LOOP

RJE could lock in a LOGON loop between host systems due to incomplete status information zeroing at LOGOFF. This problem has been corrected.

D3092 RJE - "PUT/FETCH" STRING FIELD TERMINATION CHARACTER

RJE now blank fills the following fields in the file transfer PUT/FETCH records: FILENAME, PACKNAME, USERCODE, PASSWORD (in both source and destination sections).

This causes all "string" type items in the record to terminate with a blank unless the item is field size in length.

D3098 RJE - MISSING "EOF"

Under some conditions, RJE would fail to record the reception of a "23" control message in the SEND word of FTQARRAY. This problem has been corrected.

D3159 RJE - FILE TRANSFER UNEXPECTED ABORT

RJE aborts a file transfer when any of the following occur:

1. Program fault in either FILEX or FILER tasks (SEG ARRAY, INVALID INDEX, etc.).
2. Operator DS of either FILEX or FILER tasks.
3. Parity error reading input file in FILEX.
4. The result of expanding a record because of an invalid character (non-graphic) produces a record greater than the FILE TRANSFER BLOCKSIZE.

All of the above error conditions produce a message to the originator stating the reason for the abort. The format of this error message is as follows:

```
#FT ABORT @ <hostname> - <abort reason>
```

where <hostname> is the hostname at the site where the fault, DS, etc., occurred.

<abort reason> is one of the following:

1. OPERATOR DS
2. PROG FAULT(TYPE=X, CAUSE=Y, REASON=Z)

Note: X, Y, Z are fields of the HISTORY task attribute.

3. EXPANDED RECORD > FT BLOCKSIZE
4. PARITY READING REQUESTED FILE

D3160 RJE - FORMMESSAGE LINK IN "REMLP" FILES

RJE now uses the FORMMESSAGE link in BLOCK CHARACTER CONTROL word (word 1) of the REMLP files control record to find the formmessage (if there is one).

D3161 RJE - CODEFILE RECORD TRANSLATION

RJE now unconditionally translates all records for all code files (19 <filekind <64), thus accelerating the transmission in a file transfer.

D3166 RJE - NOT SENDING "09" CONTROL MESSAGE

RJE would not send the 09 control message when a new host was logging on. This problem has been corrected.

D3168 RJE - "ONLINE, OFFLINE" BY STATIONNAME

RJE now allows the use of a stationname in the ONLINE and OFFLINE commands.

Syntax:

```
--<rje mix #>-- SM --- ONLINE -----<lsn>-----|
                  | - OFFLINE - | -<stationname>- |
```

The <stationname> must be that of the base (device address 00) station in the RJE family. If an invalid lsn or stationname is used, the following error message is displayed:

UNKNOWN STATION.

If no lsn or stationname is supplied, the following error message is displayed:

LSN OR STATIONNAME REQUIRED.

D3219 RJE - RECORDS LARGER THAN FILE TRANSFER BLOCK SIZE

RJE now allows the transfer of files whose MAXRECSIZES are greater than FILE TRANSFER BLOCKSIZE. This is accomplished by splitting the record across several blocks.

Note: This feature is only allowed between B6000/B7000 systems.

D3220 RJE - INVALID CHARACTER RECORD TRANSLATION

The number of INVALID CHARACTERS that cause RJE to translate a file transfer record has been reduced to those used by the DCP and its support software, thus reducing the number of records that have to be translated and increasing file transfer throughput.

Note: This feature is only allowed between B6000/B7000 systems.

D3234 RJE - TERMINAL TRANSFER

The RJE protocol supports a user at a terminal on one host connecting to an interactive applications program on another host, and using the terminal as if it were locally attached to the program. This is done by using the File Transfer Link and a set of stations associated with it between the two hosts.

For use in this document, the terms "local" host and "remote" host have the following meanings: the "local" host is the system where the user's terminal is physically attached, while the "remote" host is the system where the desired application is located.

Because this feature of RJE requires the use of the File Transfer Link the user should be familiar with the RJE commands ONLINE, OFFLINE, FTS and WH and the NDL stations associated with it before proceeding with this document.

The method for conveying terminal data between hosts makes use of the device address field, DA1 DA2, to specify the terminal with which the data is associated. "DA1, DA2" in this context is termed a "virtual station address", and is the name used by local and remote hosts to refer to a particular terminal. The addressing scheme is as follows:

1. DA1 and DA2 are ASCII letters, one upper case and the other lower case.
2. If "DA1, DA2" is the transmit address for a particular terminal, then "DA2, DA1" is the receive address for that terminal.
3. When the local host refers to a terminal (by transmit address), "DA1" is a lower-case letter and "DA2" is an upper-case letter. When the remote host refers to the same terminal, since the roles of "transmit" and "receive" are reversed, "DA1" is an upper-case letter and "DA2" is a lower-case letter.
4. If a local host supports <n> different terminals connecting to a particular remote host, the local host refers to them by the following virtual station addresses (as transmit addresses):

aA, bA, cA, ..., zA, aB, bB, cB, ...,

and the remote host refers to the same terminals by these virtual station addresses (also as transmit addresses):

Aa, Ab, Ac, ..., Az, Ba, Bb, Bc, ...

The local and remote hosts decide independently how many terminals they will support, so these numbers will likely be different. There is an example of the above described stations in the SYMBOL/SOURCENDL under the RJE1 family of stations. The VSIN1RJE1 and VSIN2RJE1 are receiving stations (they are used when RJE1 is acting as the remote host), while VSOUT1RJE1 and VSOUT2RJE1 are transmitting stations (they are used when RJE1 is acting as the local host). The number of stations and their type (input or output) determines the number of terminals that can be transferred at any one time. In the example RJE1 family, the host can support two input terminals as a remote host and two output terminals as a local host.

Note that the above described "virtual stations" have a terminal type of VSRJEA for asynchronous and VSRJES for synchronous. Due to the inability to assign terminal classes in the NDL, the terminal type is used to distinguish stations within RJE. Any terminal type that starts with the letters "RJE" is assumed to be an RJE Terminal with device addresses in the range "00" to "04" (RSCs, LINEPRINTERS, etc), while terminal types that do not start "RJE" are assumed to be "virtual stations". This requires any user NDL that does not conform to the above to be changed and recompiled.

In addition to the "virtual station", there is another class of station needed. The "pseudo station" is used by the remote host as a means of transferring the data from RJE's File Transfer Link (input virtual station) to a station controlled by some other MCS (CANDE, APL). The pseudo station has the same characteristics as the virtual station except the device addresses are "09", "09". The RJE1 family station has two examples of this type of station. The pseudo stations are handled in a pool by RJE, meaning that all the pseudo stations are available to any virtual station attached to RJE on the host. Therefore the assignment of the pseudo station to a particular line is not required. There should be one pseudo station declared and assigned to a line for every input virtual station attached to RJE.

New or modified commands

Connect Terminal

```
-- *CT --<hostname>--|
```

The Connect Terminal causes an attempt to attach the user's terminal to RJE on the remote host request by <hostname>. RJE on the local host will respond with one of the following messages:

#HOST <hostname> DOES NOT SUPPORT THIS FUNCTION.

The requested host is not Terminal Transfer capable.

#NOT ALLOWED FOR THIS STATION.

The station at which the *CT was entered cannot be transferred (RJE Terminals with device addresses of "00" to "04").

#HOST <hostname> IS NOT CONNECTED TO THIS HOST.

There is no File Transfer Link (see ONLINE command) to requested host.

#HOST <hostname> CANNOT SUPPORT ANOTHER STATION AT THIS TIME.

All virtual stations (output stations on local host) are in use.

#STATION ALREADY CONNECTED TO <hostname>.

An attempt to transfer a station already transferred.

If the local host accepts the request it is forwarded to the remote host for its checks. The following are the remote hosts responses:

#UNABLE TO TRANSFER TERMINAL BECAUSE OF INVALID ADDRESS OR SYSTEM FULL

The remote host cannot accept another station or the virtual station address is not valid.

#UNABLE TO TRANSFER TERMINAL NO MCS READING ON STATION.

This message can only be received from a B1800 or B800 host.

#UNABLE TO TRANSFER TERMINAL STATION INUSE.

The virtual station address is already in use on the remote host.

#UNABLE TO TRANSFER TERMINAL FOR REPLY VALUE = xx.

A reply to the *CT request of unknown value, where xx is the value.

If the *CT request is accepted by the remote host the following message is displayed:

#SUCCESSFUL TERMINAL TRANSFER TO HOST <hostname>.

Disconnect Terminal

-- *DT --|

The Disconnect Terminal command will unconditionally disconnect the station from a remote host. This message is the only message to which the local RJE will respond without forwarding it as text to the remote RJE. The following are the responses to the *DT request:

#INVALID REQUEST - TERMINAL IS NOT TRANSFERRED.

The *DT command was entered at a station that was not transferred to a remote host.

#TERMINAL TRANSFER TO HOST <hostname> HAS BEEN TERMINATED.

Successful termination of the terminal transfer.

Release

```
-- *RE -----<mcs name>--|
      |-<lsn>-| | - TO -|
```

Examples:

```
*RE 39 SYSTEM/APL
*RE SYSTEM/CANDE
```

This command causes the station indicated by the optional <lsn>, or if no <lsn> is specified, the station at which the request is entered to be released (detached and attached) to the indicated MCS. If the <lsn> does not belong to one of the stations of the terminal at which the keyin was entered, RJE will respond with the following:

#INVALID LSN

If the requested <mcs name> is not declared in the host NDL, RJE will respond with the following:

#INVALID MCS

If, through the use of the *CT <hostname> command, the terminal has been transferred to a remote RJE, only the non-lsn version of the request is allowed. Use of an <lsn> will cause the following message:

#LSN NOT ALLOWED

If the remote RJE cannot complete the release because of insufficient pseudo stations, RJE will respond with the following:

#NO PSEUDO STATION AVAILABLE.

New SM commands

Virtual Stations

```
--<rje mix #>-- SM -- VS -----|
                        |-<lsn>-|
```

The Virtual Station command will display the status of all the virtual station associated with RJE. The optional <lsn> will restrict the display to only the virtual station associated with that station. The following is an example of the display:

```
[084] RJE1 @ SYS1073 (3 TOTAL, 1 INUSE STATIONS)
[089] INPUT STATION NOT CONNECTED
[090] INPUT STATION NOT CONNECTED
[091] OUTPUT STATION CONNECTION ATTEMPT IN PROGRESS
[124] RJE1 @ B68001244 (2 TOTAL, 1 INUSE STATIONS)
[130] INPUT STATION NOT CONNECTED
[131] OUTPUT STATION CONNECTED TO TD426361 [199]
[153] RJE2 NO HOSTNAME (1 TOTAL, 0 INUSE STATIONS)
[160] INPUT STATION NOT CONNECTED
```

In the above example, all the numbers in brackets ([]) are LSNs. The identifiers RJE1 and RJE2 are station names of the device address "00" stations in control of a File Transfer Link. The identifiers SYS1073, B68001244 are the host names of the systems to which the File Transfer Links are connected. The remainder of the lines are the virtual stations with their status (CONNECTED or NOT CONNECTED) and type (INPUT or OUTPUT). LSN 91 is processing a *CT request and has not yet received a reply from remote host SYS1073. LSN 131, which is an output station, is currently being used by a station called "TD426361" at LSN 199. LSN 153 station RJE2 is currently not connected to any remote host indicated by "NO HOSTNAME".

Non-RJE Terminal Communication With RJE

A non-RJE terminal is a terminal whose NDL Terminal type identifier does not start with the letters "RJE"; e.g., TD800s and TELETYPES. These terminals have usually been released to RJE from some other MCS and do not belong to an RJE family of stations (PROCESSOR, RSC, CARDREADER, LINEPRINTER, FTS). To control these new terminals several changes have been made to RJE. Two new defines have been added to the "MAXTERMINALS" define to allow specification of the mix of RJE to non-RJE terminals. "MAXNONRJETERMINALS" specifies the maximum number of this type of terminals allowed at any one time, while "MAXRJETERMINALS" specifies the maximum for the standard RJE type terminals. The define "MAXTERMINALS" is the total of "MAXRJETERMINALS" and "MAXNONRJETERMINALS".

Once a non-RJE terminal has been released to RJE, any Log-on and or Stationid checks that are required by RJE will be performed. After a successful Log-on, the terminal will respond just like a RSC at any RJE terminal with all the same restrictions placed on it by SYSTEM/SCABLEGEN. All of RJE's "*" commands are valid and access to WFL through CONTROLCARD is allowed. To return the terminal to its original MCS, the *RE or BYE command is required.

A new "*" command has been created for use by these non-RJE terminals which allows RJE ODT capabilities. The command syntax is as follows:

```
-- *SM --<any RJE ODT SM command>--|
```

Examples:

```
*SM WH
*SM COPY A (USER=X/Y) FROM DISK (HOST=SYS1) TO FRED (HOST=SYS2)
*SM TF
```

The *SM command allows the terminal users the ability to enter any of RJE's ODT SM commands. The use of this command is controlled by the setting of the NDL's SPO station bit. If the users station is not SPO = TRUE then the following error message is displayed:

```
#NOT ALLOWED.
```

Sample RJE Terminal Transfer Session

The following is a sample of what a Terminal Transfer might look like between two B6800 systems.

SYS1 Stations	SYS2 Stations
RJE1 (device address "00")	RJE2 (device address "00")
RJE1RSC ("01")	RJE2RSC ("01")
RJE1CR ("02")	RJE2CR ("02")
RJE1LP ("03")	RJE2LP ("03")

RJE1FTS ("04")	RJE2FTS ("04")	
VSOUT1 ("Aa","aA")	VSIN2 ("aA","Aa")	
TD83XX ("XX") LSN 296	PSEUDO2 ("09","09") LSN 19	

The above stations form the Datacom Network on two systems with the HOSTNAMEs of SYS1 and SYS2. All the stations on SYS2 are assigned to a single line and form an RJE family. The TD83XX station on SYS1 is a TD830 terminal attached to SYSTEM/CANDE, while remainder of the stations form an RJE family.

It is assumed that the two hosts are connected via the File Transfer Link between RJE1FTS on SYS1 and RJE2FTS on SYS2. The user at station TD83XX wishes to use SYSTEM/CANDE on host SYS2 and enters the following:

?MCS SYSTEM/RJE

This causes CANDE to release control of the station to SYSTEM/RJE, which causes the following output:

```
#END SESSION 4567 ET=2:13:49 PT=0:0 IO=0:0
#USER = MIKE 11:40:31 04/28/80
296: RELEASE COMPLETED
```

The #END SESSION and #USER messages are CANDE's while the third response is from RJE showing that the release was successful. RJE now controls TD83XX and any input at the terminal is now responded to by RJE. It is recommended that the RJE input "HELLO" be used at this point to activate the station. RJE will then respond with the following:

B6800 SYSTEM/RJE SYSTEM #456 HOSTNAME = SYS1 32.82.48 MIX = 6019.

At this point RJE will either ask for a USERCODE and PASSWORD or assign a SESSION, depending upon the setting of LOGON. For this example, assume LOGON is not set causing the following:

```
TD83XX LOGGED ON AT 11:41:02 04/28/80.
SESSION 6282.
```

The terminal user may now use any RJE command ("*") or CONTROLLER request he chooses. The following are example of the types of commands and their responses:

```
WT      TIME IS 11:50:23

*TF
#BLKSZ = 400  BUFSZ = 820  FTBLK = 1500

*BACKUP
#NO BACKUP FILES.
#END BACKUP REQUEST.

A      ----- NO ACTIVE ENTRIES -----
```

If TD83XX has the NDL SPO bit set the *SM command is also valid for use.

```
*SM WH
084:ACTIVE RJE1 @ SYS2 (6800)(TTP,FTS,HOST,VER=2)

*SM FTS
[084] RJE1 @ SYS2 NO ACTIVE FILE TRANSFER
```

Now that the terminal is released to RJE on SYS1 the connection to SYS2 is made by use of the *CT <hostname> command.

*CT SYS2

If the transfer is successful RJE on host SYS1 will respond with the following:

#SUCCESSFUL TERMINAL TRANSFER TO HOST SYS2.

What has now taken place is that RJE on SYS1 has associated TD83XX with the virtual station VSOUT1 and is transferring all input and output between them, while RJE on host SYS2 is now reading and writing to station VSIN2. At this point, the terminal user can use any of RJE's or the CONTROLLER's commands that are allowed on the remote host SYS2.

For the terminal user to now use SYSTEM/CANDE (or any other MCS) on the remote host SYS2, the *RE command is used as in the following:

*RE SYSTEM/CANDE

If the release to CANDE is successful, the following will be displayed:

```
19: RELEASE COMPLETED
#B6800 :678 CANDE 32.58; YOU ARE PSEUDO2(19)
#ENTER USERCODE PLEASE
```

At this point, the terminal user is attached to SYSTEM/CANDE on the remote host SYS2. On SYS2 the virtual station VSIN2 is now associated with the pseudo station PSEUDO2 which has been released to CANDE. For the life of the connection, all input from the terminal will go to CANDE on SYS2 while all output from CANDE will go to the users terminal on SYS1.

When the user is finished with CANDE on SYS2, he needs to cause the pseudo station to be released back to RJE. This can be done by entering to CANDE "?MCS SYSTEM/RJE". At this point, the user is again connected to RJE on SYS2; to break this connection, he enters the *DT command which will result in the following:

#TERMINAL TRANSFER TO HOST SYS2 HAS BEEN TERMINATED

The user is now talking to RJE on the local host SYS1 and a "BYE" will return him to SYSTEM/CANDE.

Halt Load Effects

If the remote host should Halt/Load during the life of the connection of a terminal, the association between the remote virtual and pseudo stations is lost and any tasks using the stations are DSed. The user should use the *DT command to free his terminal from the local virtual station to be able to communicate with the local RJE. Once the remote host is back up and the File Transfer Link is active, the terminal can be re-transferred.

If the local host should Halt/Load, the user's terminal is released back to its original MCS by the reinitializing DCP. At this point, the remote pseudo station is still released to whatever MCS last owned it and will remain so until manually released back to RJE by that MCS or when the remote host next Halt/Loads. Until the pseudo station is released back to RJE, it is not available to any virtual station.

MCP

Several changes have been made to the MCP to support RJE's Terminal Transfer.

Room was needed in the STATION TABLE for an additional MCS number (PSEUDOMCSNRF) of 6 bits. The PRIMARY, CURRENT, and STATION Queue number fields were reduced by 1 bit each and combined with 3 available bits to form the field.

The "TRANSFER STATION CONTROL" (TYPE=45) DCWRITE has been modified to allow the transfer of a station from one MCS (e.g., RJE) to another (e.g., CANDE) yet still maintain partial control of the transferred station. This variant of DCWRITE 45 is invoked by setting bit 26 in word 0 of the message. The MCS number of the MCS making the request will be stored in word 2 of the STATION TABLE at PSEUDOMCSNRF ([41:6]) while the MCS number of the MCS being transferred to will be stored in word 0 at STAMCSNRF ([37:8]). If, after a station has been transferred with the above variant set, it is transferred again by the new controlling MCS, the PSEUDOMCSNRF field will still contain the original controlling MCS number. This allows the user to transfer the terminal from remote MCS to remote MCS without going back to the original controlling remote MCS (RJE) each time. When the transfer is back to the original controlling MCS, the PSEUDOMCSNRF is returned to its initial value of zero.

A new "WRITE TO TRANSFERRED STATION" (TYPE=53) DCWRITE has been created to allow the controlling MCS the ability to direct messages from its stations to their corresponding transferred stations now controlled by another MCS. There are three variants of DCWRITE 53:

- 0: A GOOD RESULT message is to be forwarded to the MCS (STAMCSNRF) in control of the requested LSN.
- 1: A STATION EVENT message is to be forwarded to the MCS (STAMCSNRF) in control of the requested LSN.
- 2: The result of some write from the transferred station is to be returned to either the controlling MCS or to Logical I/O by a call on DCIOFINISH.

In variants 0 and 1, the message is forwarded with only the MSG[0] word modified to reflect a Message Class of either GOOD RESULT or STATION EVENT, a VARIANT of 0, and the requested LSN in MSGUNITF.

In variant 2, the message to be forwarded is in the text portion of the type 53 message and is 6 words in length. It is the result of RJE successfully placing a message onto the File Transfer Link to the local host. This result will be returned to either Logical I/O if the original message type was of type OBJECTJOBOUTPUT (output of Logical I/O) or to the MCS now in control of the transferred station. If the MCS does not want results through the setting of the variant of the "STATION ATTACH" DCWRITE, the message will be discarded.

For stations that have been transferred and are doing Logical I/O, special FILE OPEN and FILE CLOSE messages are generated and sent to the MCS pointed to by PSEUDOMCSNRF. The layout of these messages are as follows:

Word [0].	[23:24]	LSN of transferred station
	[26:01]	1 Flag for audit and debugging use
	[47:08]	FILEOPENMSG or FILECLOSEMSG type
[2].	[39:16]	3 text size
[6].	[23:24]	FRSN of lsn

To allow RJE Terminal Transfer to work without modifying any other MCS, the MCP will intercept all messages of type WRITEREQ from stations that have been transferred (indicated by the fact that PSEUDOMCSNSF is non-zero) by use of the above described DCWRITE 45. These messages are then directed at the MCS pointed to by PSEUDOMCSNRF with a "TRANSFERRED INPUT RESULT" (CLASS=29) MESSAGE and the MSGUNITF field changed to LSN from DLS.

DCSTATUS and DUMPANALYZER

The defines for Primary, Current and Station Queues have been changed to match the new defines in the MCP. Also the new field PSEUDOMCSNRF of the STATION TABLES (word 2) has been added.

The display of PSEUDOMCSNR has been added to the STATION TABLE display of the DUMPANALYZER option DCPANALYSIS and the DCSTATUS display of a station.

SOURCENDL

The SOURCENDL has been modified with the addition of several new terminal types and some example stations. The new terminals are used by RJE for File Transfer and Terminal Transfer. They consist of RJEAFTS and RJEFTS which are asynchronous and synchronous terminals which allow RJE to transfer blocks of up to 2000 characters without the need of defining excessive message space for the other stations needed on the line. There are also four new stations attached to RJE1 which are to be used as examples of Virtual and Pseudo stations.

D3247 RJE - PHONE NUMBERS

RJE will now allow alphanumeric phone numbers of up to 71 characters in length. This change has caused a modification of the LINKFILE and a change to LINKFILEVERSION, making it incompatible with previous versions.

For phone number strings greater than 71 characters, the following error message is displayed:

"# PHONE NUMBER GTR 71 CHARACTERS"

D3255 RJE - FILE TRANSFER

The File Transfer facility has been enhanced for the Mark 32 system release. PCN marks in the right margin in the following system note indicate changes since the Mark 31 implementation.

The implementation requirements of the following new features have caused an increase in the number of "system control messages" (device address 00 messages) plus format changes to some of the old messages. The use of these new "system control messages" may cause compatibility problems with old or non-Burroughs RJE terminals and Mark 32 SYSTEM/RJE.

SYSTEM/RJE has been modified to allow files to be transferred between host systems and between host systems and terminals. The file transfer is done through a new station added to a site's NDL for each RJE station family. This new station has a device address of 04 but no other special attributes. An example of this new station can be found in the SYMBOL/SOURCENDL file for this software release.

Since this new feature is symmetric, it may be invoked between any two systems without regard to "HOST" vs. "TERMINAL" status; thus, the term "HOST" will be used in the remainder of this document. There are two classes of connection between hosts, "peer" and "non-peer". A peer host is a system of the same software type; e.g., B6700, B6800, B7700 are peers while a B1800 connected to a B6800 would be a non-peer. For the remainder of this document a peer host will mean two "Large System" hosts.

File transfer is supported for disk and pack files with the following restrictions:

1. Only code and 8-bit data files (including text files) may be transferred.
2. Only one file at a time may be transferred in each direction.
3. File titles cannot be greater than 100 characters in length.
4. The following restriction applies only to non-peer connections. To support code file transfers (and in some case data files) RJE must scan every record looking for characters that are outside of its "valid character" set. Upon finding a record with an offending character the whole record is translated into a new form which is comprised of only valid characters but twice the original length. If this new record length or the original record length (maxrecsize) are larger than the allowed File Transfer Blocksize the transfer will be terminated.

The user initiates a file transfer by entering a variation of a WFL "COPY" statement on the host ODT. This command specifies source and destination hosts and file titles. The hosts are specified by the use of their "HOSTNAME"; in the case of a "LARGE SYSTEM" host (B6700, B6800, B7700, etc.), this hostname is created in the following fashion:

If the "HN" (HOSTNAME) ODT command has not been used, then SYSTEM/RJE creates a hostname at run time. This hostname is the letter "B" followed by the 4 digit system model number (6700, 6800, etc.) and then the 3 or 4 digit system serial number. For sites that have specified a hostname by the "HN" ODT command, that hostname is used. Note that if two sites are connected to a host with the same hostname, requests to transfer files may not produce the desired results.

The actual transfer of a file is initiated by the exchange of control information between the hosts. If the specified source host is the user's local host, the transfer is a "PUT"; if the source host is the remote host, then the transfer is a "FETCH". This means that if the file to be transferred from my host to your host is on my host, the "COPY" request will generate a "PUT" control message; if the file is on your host, it will generate a "FETCH" control message. All file transfers occur as "PUTS"; if one host wishes to do a "FETCH", it sends a "FETCH" message to the other host which causes it to initiate a "PUT" sequence. The "PUT" message elicits a "PUT REPLY" which signals that data transfer may begin or supplies a reason for the rejection of the "PUT" request. If the transfer is OKed, the sending host then sends one or more data messages to device address 04 followed by an end-of-file message. The sending host processes a task called "FILEX[<lsn>]", where <lsn> is the LSN of the station connected to the remote host. This task runs independently of SYSTEM/RJE. It opens the requested source file and performs any data compaction or translation required in the course of the transfer. The receiving host processes a task called "FILER[<lsn>]" which also runs independently of SYSTEM/RJE. This task reverses whatever FILEX did to the data and creates the requested destination file. Upon completion of the file transfer, the receiving host informs the requestor of completion of the request.

Any file transfer requests that occur while another transfer is in progress are queued and then executed when possible. When a request is queued, the following message is displayed:

#FILE TRANSFER STATION IN USE - COPY REQUEST WILL BE QUEUED.

All requests are queued as "PUTS", so if a user requests a "COPY" that results in a "FETCH" being generated, the request is sent to the other host and may be queued for the reason given above. If the "COPY" request is syntactically correct and passes resident and security checks, the following message is displayed:

#COPY REQUEST VALID - WILL FORWARD.

The other host responds either with an error display or a request accepted display, in which case the transfer will start.

All files from non-peer hosts will be locked as type data, while files from peers will be locked with as many of the original file's attributes as possible (including FILEKIND).

To implement this new feature, a set of new commands has been added to SYSTEM/RJE and several old commands have been modified. The file transfer requests and their progress displays are made on the host ODT by using SYSTEM/RJE "SM" message syntax.

New SYSTEM/RJE SM Commands

The syntax checking of the COPY command has been modified to allow requests to non-B6000/B7000 systems (via SYCOM). RJE syntax checks only the part of the request involving its host; the remainder of the request is forwarded as supplied by the user.

Syntax:

COPY

```

-- COPY --<filename>-----
--                                     | -<usercode spec>- | | - BINARY - |
>----- FROM --<volume spec>-----
| - AS -----<filename>----- |
| - ONTO - | | -<usercode spec>- |
> TO --<volume spec>-----

<usercode spec>
-- ( -- USER = --<usercode>----- ) --
| - , -- PASSWORD = --<password>- |
| - / --<password>----- |

<volume spec>
----- ( ----- HOST = ----->
| - DISK ----- | | ----- TAPE ----- , - | | |
| - PACK ----- | | - KIND = - | | - TAPE7 -- |
| -<volume name>- | | - TAPE9 -- |
| | - PETAPE -- |
| | - DISK --- |
| | - PACK --- |
>--<hostname>-- ) -----

```

Semantics:

The "COPY" command transfers the requested file from the source HOST to the destination HOST. The <usercode spec> is used to verify access rights to the specified file and to the use of the file transfer station. It may also affect defaults for parts of the file title. For file names that start with the characters "*" or "(", the <usercode spec> usercode is not used as a prefix. The <usercode spec> is required for any source or destination that is a B6000/B7000 system HOST. The default volume kind is "PACK". The <volume name> may be omitted from either the source or destination parts of the request for non-B6000/B7000 systems; the affected HOST does whatever defaulting is required for that system. The "COPY" request does not overwrite an existing file on the destination HOST unless the "ONTO" modifier is used.

The "BINARY" option causes RJE to transmit the requested data (noncode) file in an encoded fashion. This is required in some cases when the requested transfer is between ASCII (B800) and EBCDIC (B1800, B6000/B7000) hosts due to differences in allowable characters. In transfers of data files between ASCII and EBCDIC hosts where the "BINARY" option is not used, all non-graphic characters are replaced by a graphic question mark. The "BINARY" option has no effect on the transfer of code files as they are always encoded between ASCII and EBCDIC hosts.

Examples:

```

<RJE MIX#> SM COPY TESTFILE (USER=MIKE) FROM MCPMAST (HOST=B6700282)
TO DISK (HOST=B68001004)

```

Transfers "(MIKE)TESTFILE" from a pack called "MCPMAST" on HOST "B6700282" to a pack named "DISK" on HOST "B68001004". Usercode "MIKE" (with no password) must be a valid usercode on both hosts.

```

<RJE MIX#> SM COPY TESTFILE (USER=MIKE) AS X (USER=RJE/RJE) FROM
MCPMAST (HOST=B6700282) TO DISK (HOST=B68001004)

```

Same as the first example except the destination file title is "(RJE)X ON DISK". Usercode "MIKE" is used for all security and resident checks on HOST "B6700282" while "RJE/RJE" is used on HOST "B68001004".

```

<RJE MIX#> SM COPY *WFL (USER=MIKE) ONTO TEMP (USER=WHY,PASSWORD=ME)
FROM DISK (HOST=B6700282) TO PACK (HOST=B7700003)

```

B6000 SERIES MARK 32

Transfers the system file "*WFL" from DISK on HOST "B6700282" as a file called "(WHY)TEMP ON PACK" (RJE overwrites the file if it already exists) on HOST "B7700003".

<RJE MIX#> SM COPY *WFL (USER=MIKE) FROM MCPMAST (HOST=B6700282)
TO (HOST=B1800XX)

Transfers the system file "*WFL" from MCPMAST on HOST "B6700282" to a B1800 system with the hostname of "B1800XX". The B1800 uses whatever defaults it has defined for destination family.

<RJE MIX#> SM COPY *WFL (USER=MIKE) FROM MCPMAST (HOST=B6700282)
TO (HOST=B6800135)

<RJE MIX#> SM COPY *WFL (USER=MIKE) FROM (HOST=B6700282)
TO PACK (HOST=B6800135)

The two examples above both cause syntax errors because of missing volume names (both hosts are B6000/B7000 systems).

The following are error messages that are produced as a result of an invalid "COPY" command:

FILENAME GTR 100 CHAR.

INVALID FILENAME: <filename>.

NO FROM OR TO PARTS.

NO FROM PART.

INVALID VOLUME NAME: <volume name>.

MISSING VOLUME NAME.

VOLUME NAME AND KIND MISMATCH.

Example: Setting "KIND=DISK" when <volume name> is "DISK".

MISSING EQUAL.

Equal signs are required after key words:
"USER", "PASSWORD", "HOST", "KIND"

INVALID MEDIA TYPE: <media type>.

NO HOSTNAME.

EXPECTING PARENTHESIS.

NO TO PART.

NO USERCODE/PASSWORD.

INVALID USER/PASS SYNTAX.

SOURCE AND DEST HOSTNAMES THE SAME.

THIS HOST NOT SOURCE OR DEST.

USERDATAFILE FROZEN - TRY LATER.

The USERDATAFILE was not available at security
check time.

SECURITY ERROR ON USER = <usercode>.

FILE <filename> NOT AVAILABLE.

FILE <filename> NOT RESIDENT.

FILE <filename> ALREADY EXISTS.

FILE <filename> IS OPEN EXCLUSIVE AND CANNOT BE REPLACED.

FAMILY <familyname> IS NOT PRESENT.

ABORT

```

-- ABORT ---<lsn>--- SEND -----|
-----| - RECV -| - = -----|
          |      |      | |
          |      |      | -<integer>--|
          |      |      | -<filename>--|

```

The ABORT command is used to stop an active file transfer or delete one or all entries that are queued for a host. The key words "SEND" and "RECV" are used to specify the type of transfer to abort. The <lsn> is the LSN of the station to which the remote host is connected. See the "FTS" command for more information concerning the <integer> and <filename> formats and meanings.

Examples:

```
<rje mix#> SM ABORT <lsn> SEND
```

Aborts the current active "FILEX" task (the sender).

```
<rje mix#> SM ABORT <lsn> RECV
```

Aborts the current active "FILER" task (the receiver).

```
<rje mix#> SM ABORT <lsn> SEND =
```

Deletes all requests queued at this host for the requested <lsn>.

```
<rje mix#> SM ABORT <lsn> RECV =
```

Deletes all requests queued at the remote host for transmission to the requested <lsn>.

```
<rje mix#> SM ABORT <lsn> SEND <integer>
```

Deletes the <integer> entry from the queued list but does not delete the active entry (queued number 0).

```
<rje mix#> SM ABORT <lsn> RECV <integer>
```

Deletes the <integer> entry from the queued list of the remote host connected to the requested <lsn>, but does not delete active entry (queued number 0).

```
<rje mix#> SM ABORT <lsn> SEND <filename>
```

Deletes the first queued entry from the queued list for the <lsn> requested where the queued source filename matches <filename>.

```
<rje mix#> SM ABORT <lsn> RECV <filename>
```

Deletes the first queued entry from the queued list of the host that is using the <lsn> where the queued source filename matches <filename>.

ONLINE

```

-- ONLINE ---<lsn>-----|
-----| -<stationname>-|

```

The ONLINE command initiates the connection between two hosts. Depending on the type of datacom connection, this command has different effects.

SWITCHED LINE WITHOUT AUTOMATIC CALLING UNIT

The request has no effect and elicits a response of :

#NON DIALOUT CAPABLE STATION.

SWITCHED LINE WITH AUTOMATIC CALLING UNIT

Causes dialout if phone number is set up (see PH command).
RJE will respond with either :

#NO DIALOUT PHONE NUMBER
or
#DIALING OUT STARTED

DIRECT OR LEASED LINE

Causes the 09 station-id messages to be sent.
RJE will respond with :

#OK

If the <lsn> is already connected and active RJE displays:

#STATION ALREADY ACTIVE.

OFFLINE

```
-- OFFLINE ---<lsn>-----|
-----| -<stationname>-| - EOJ -|
```

The OFFLINE command causes termination of the datacom connection between two host systems. Depending on the type of datacom connection, this command has different effects.

SWITCHED LINE - causes log-off and line disconnect.

DIRECT OR LEASED LINE - causes log-off, but leaves line active.

The EOJ modifier to the OFFLINE command causes RJE to complete its current file transfer (if one is active) and then terminate the session.

RJE responds with the following message to the OFFLINE command:

#OK.

PH (Phone Number)

```
-- PH ---<lsn>-----|
      | - CLEAR -----|
      | -<phone number>-|
      | - INT -----|
```

<phone number>

-- Maximum of 71 characters ---|

The PH (phone number) command is used to display, clear, change or retry the automatic calling unit phone number associated with a station.

Examples:

<rje mix#> SM PH <lsn>

Displays the current phone number (if any).

<rje mix#> SM PH <lsn> CLEAR

Clears the phone number.

<rje mix#> SM PH <lsn> <phone number>

Enters new phone number.

<rje mix#> SM PH <lsn> INT

Causes dialout retry if the callback option is set for <lsn> station.

SF (Set Factors)

```

-- SF --<lsn> |-----|
               |<integer>-----|
               |
               | - BLKSZ - |
               | - BUFSZ - | - = - |
               | - FTBLK - |
  
```

This command specifies the characters per transmission blocking factor (BLKSZ), buffer size in characters of remote printer and or punch output (BUFSZ) and the maximum number of characters allowed in a file transfer block (FTBLK). The following defaults and limits are imposed :

	Defaults	Limits
BLKSZ	400	132 through 1500 (inclusive)
BUFSZ	820	132 through 2000 (inclusive)
FTBLK	400	400 through 2000 (inclusive)

Changing any of these values should be done with the utmost care to avoid exceeding the limits of the remote terminal or of the datacom line.

TF (Type Factors)

-- TF --|

This command has been modified to display the characters per transmission blocking factor (BLKSZ), buffer size in characters of remote printer and or punch output (BUFSZ) and the maximum number of characters allowed in a file transfer block (FTBLK). The defaults and limits imposed are the same as for the SF command.

```

-- FTS -----|
               |<lsn>-----|
               |
               | - LOCKQ ---|
               | - UNLOCKQ -|
  
```

The FTS command monitors the file transfer activity. This command without the optional <lsn> displays one or three lines for each active station attached to SYSTEM/RJE. If the station has no "HOSTNAME", or if there are no file transfers active on the station, one line will be displayed showing LSN, station name, hostname (if any) and a comment of "NO ACTIVE FILE TRANSFER". For stations with an active file transfer, three lines are displayed showing the status of the station and of the active file transfers.

If the optional <lsn> is used with the FTS command, only the information for that station is displayed. In this case, the display is either one or four or more lines depending on the number of queued requests.

The use of the modifiers "LOCKQ" and "UNLOCKQ" cause RJE to stop and start the file transfer send queue for the selected station. Once the queue is locked, RJE continues to queue new entries. The queue is also locked by use of the EOJ modifier of the OFFLINE command.

Example outputs:

The examples that follow are the results of using the FTS command without the optional <lsn>.

```

[034] RJE1 @ B6800282 NO ACTIVE FILE TRANSFER
[039] RJETIO NO HOSTNAME NO ACTIVE FILE TRANSFER
[044] RJEGMM @ B6700282 2 REQUEST QUEUED
      RECV: FILER ACTIVE (4321) 21 OF 54
      SEND: FILEX ACTIVE (4401) 1053 OF 10049 (PACING)
  
```

The first line of the above example has the following meanings:

```
[034]      = The base LSN of the RJE station
RJE1       = Station name
B6800282   = Remote host name that is connected to this station
```

In the second line of the above example the host or terminal using station RJETIO has not sent its host name (if it has one) and no file transfers are allowed.

In lines three through five of the above example, the host B6700282 is active on station RJEGMM and there is a file transfer running in each direction. The "RECV" line shows the status of the receiving task FILER. The number in parentheses is the mix number of FILER while the XXX OF YYY numbers are the current count of records transferred of the total for the file. The "SEND" line shows the status of the sending task FILEX. The number in parentheses is the mix number of FILEX while the XXX OF YYY numbers are the current count of records transferred of the total for the file. The word "PACING" indicates that FILEX is waiting for a pacing flag before continuing with the next file transfer block of records.

The example that follows is of LSN 44 in the last example using the optional <lsn>.

```
[044] RJEGMM @ B6700282 2 REQUEST QUEUED
RECV: FILER ACTIVE (4321) 21 OF 54
      <00> (MIKE)TESTFILE ON TESTPACK
          AS (MIKE)NEW/TESTFILE ON MCPMAST.
SEND: FILEX ACTIVE (4401) 1053 OF 10046 (PACING)
      <00> (MIKE)RJEX ON MCPMAST.
      <01> *SYMBOL/WFL
          AS (MIKE)WFL ON TESTPACK.
      <02> (RJE)DUMMY ON PACK
          ONTO (MIKE)SAVE/THIS/FILE/FOREVER.
```

The lines that start with the items [044], RECV: and SEND: have the same meanings as above. The lines that start with <00> are the file names of the active file transfers and the counts displayed in the rcv and send lines refer to them. The lines under the send line that start with <01> through <02> are requests that have been queued. The numbers displayed within the <> and the file names on those lines are the numbers and file names used in the "ABORT" command described above.

436

*SF and *TF

The RSC keyin commands *SF and *TF have been changed to reflect the addition of the new factor "FTBLK". The syntax of these commands is the same as their SM command counterparts.

File Transfer Recovery

If during a file transfer, one or both of the hosts stops (Halt/Load, RJE fault, DCP death, etc.), SYSTEM/RJE stores enough information about the transfer to restart it where it left off. This is done by storing the current record counts and the "COPY" requests in RJE's "LINKFILE".

On the receiving host, the destination file is not created directly but a temporary file is created to allow restart and recovery. This temporary file is called "FILERFILE[<lsn>]" and is created under the destination usercode, unless it is a system file (*), on the destination <volumename>. Upon successful completion of the transfer, the temporary file is re-titled and locked.

At initialization RJE checks to see if it was involved in a file transfer that was not completed. The requirements to continue a file transfer are the following:

1. Connection is made to the same host on the same station (datacom line) that was last using it.
2. The linkfile on both hosts are still intact (do not remove the linkfile).
3. The temporary output file on the receiving host is available.

If the above conditions exist, then at initialization the receiving host requests the file transfer to continue. The "SEND" and "RECV" lines of the "FTS" command display have the word "RESTARTED" in parentheses for any file transfer that has been continued.

If the re-connection of the datacom line is to a different host, all queued request and the active requests are deleted and all restart information is discarded.

Linkfile Restructure

Several changes have been made to SYSTEM/RJE to support the file transfer feature. One of these is the restructuring of the linkfile. To ensure the correct linkfile format, SYSTEM/RJE checks at initialization the "LINKFILE VERSION" which is stored in word 56 of record 0. If the version of SYSTEM/RJE and its linkfile do not agree, the linkfile will be discarded. There are currently four reasons for discarding an old linkfile :

1. NDL file (DCPCODE and NIF) changes.
2. Maxterminals in SYSTEM/RJE has changed.
3. Linkfile version not correct.
4. Parity error reading old file.

Each of these conditions causes SYSTEM/RJE to display an appropriate message.

D3286 RJE - "WH" DISPLAY ENHANCEMENT

The RJE command WH will now display the protocol level and version for stations which supply the correct 09 control message. The abbreviations used and their meanings are the following:

TTP The station is capable of supporting terminal-to-program connections.

FTS The station is file transfer capable.

TERM The station is acting as a terminal.

HOST The station is acting as a host.

VER=<n>
The protocol version number.

D3287 RJE - "SM" COMMAND "LEVELS"

A new RJE SM command has been implemented: LEVELS.

Syntax:

--<rje mix #>-- SM -- LEVELS --|

The use of LEVELS allows display of the following:

RJEPROTOCOLVERSION The version of file transfer and terminal-to-program protocols.

SYMBOLICHEADERLEVEL The level of the symbolic header used in a file transfer.

LINKFILEVERSION The version of RJE Halt/Load recovery LINKFILE.

D3346 RJE - "RJE" VS. "DLBACKUP"

RJE will now search "DISK", "PACK" and the "DLBACKUP" for REMLP and REMCP files. The location of the "DLBACKUP" is tested for only at RJE BOJ and used only if it points to a family other than "DISK" or "PACK". If "DLBACKUP" is changed to a family other than "DISK" or "PACK" with RJE active, any REMLP file created on the new family will not be found by RJE. The REMLP files can only be found by QUITting RJE and reinitializing with the correct "DLBACKUP". This will cause RJE to rebuild its print queues from "DISK", "PACK" and the new "DLBACKUP" family. Any files on the old "DLBACKUP" family (if the family is not "DISK" or "PACK") that have not been printed will not be found by RJE.

D3392 RJE - FILE STARTING WITH "?" LOST BLOCK

RJE will now successfully transfer files whose first record starts with the character "?". The "?", because it is RJE's NDL control character, previously caused a STATION EVENT message instead of a GOOD INPUT message.

D3423 RJE - "DEBUG" VS "RAID"

The RJE dollar option "DEBUG" has been changed to "RAID" to eliminate a conflict with the DCALGOL compiler.

D3424 RJE - "SM" COMMANDS "RSC, SPO"

Two new SM commands have been implemented to control the use of the *SM command by terminal users. The commands RSC and SPO have the following syntax:

```
--<rje mix#>-- SM --- RSC -----|
                   | - SPO - | | - - - |
                   |         | | - + - |
```

The RSC command is used to control the device address "01" station (the Remote Supervisory Console) in an RJE family of stations, while the SPO command does the same thing but for non-RJE terminals (e.g., terminals transferred to RJE from some other MCS). The setting (+) of RSC and SPO allows the affected terminals to use the *SM syntax regardless of the setting of the SPO NDL bit. The resetting (-) of the options causes the setting of the SPO NDL bit to control access to *SM. A request of just RSC or SPO causes the display of the current setting of the requested run-time option.

The following are the possible responses to an RSC or SPO request:

```
SPO IS SET
SPO IS RESET
RSC IS SET
RSC IS RESET
```

Corresponding compile-time dollar options, RSC and SPO, have been added. They are RESET by default, and have the same meanings as the run-time options.

D3428 RJE - BAD DEVICE ADDRESS

RJE will now display the correct device addresses (both transmit and receive) for bad stations (addresses that do not conform to the RJE standard).

D3429 RJE - HALT/LOAD RESTORE

RJE will now automatically restart a session to a remote terminal following a Halt/Load at the host site if the following are true:

1. The remote terminal was active before Halt/Load.
2. The remote terminal has LOGON reset (no usercode required).
3. The LINKFILE is intact after the Halt/Load.
4. The remote terminal is on a leased line (non-switched).

RJE will then display its header message followed by the word "RESTARTING". Any "AUTOBACKUP" that was running at Halt/Load time will be continued or restarted depending upon the setting of the user terminal option CONTBACKUP. Any card input or "WFLCOMPILER[xx]" task that was active will be lost.

It should be noted that RJE will not restart automatically after the Halt/Load if it was initiated in the following manners:

1. Run by use of the ODT primitive ??RUN.
2. Run by the DCP in response to new station activity.

In the above cases, RJE will only restart when there is station activity, forced to run by use of the WFL RUN command (or ??RUN ODT primitive), or EOT, EOJ notification from the CONTROLLER of a AUTORECOVERY restarted RJE task or job.

For RJE to restart automatically, AUTORECOVERY and AUTODC run-time ODT options must be set and RJE must have been run (not ??RUN ODT primitive).

Regardless of how RJE restarts, if the above described conditions are met, the remote terminal will restart.

D3475 RJE - RUNTIME OPTIONS SAVE THROUGH LINKFILE PURGE

The RJE run-time options (WAIT, FTSBUG, SPO, RSC) will be maintained at BOJ, even if the old linkfile is being purged due to incompatibility.

D3476 RJE - PROGRAMDUMP OUT OF "FILEX, FILER"

RJE will now take PROGRAMDUMP out of FILEX and FILER for any fault if RAID is set TRUE.

D3639 RJE - "RJE" VS. "SYCOM"

RJE now sends an FS1 <filename> message before the first record of a printer session and an FS2 after the last record of the session to systems that have the "I-AM-A-HOST" bit set in their 09 control message. All other stations will receive printer sessions as in the past.

D3640 RJE - "ODT" TO "ODT" COMMUNICATION

RJE now handles *SM SS <lsn> <text> requests where <lsn> is the LSN of a "HOST SYSTEM". The <text> of the message will be prefixed by "#FROM <hostname> :" before it is written to the host system's RSC station.

*SM SS to non-host systems will still be prefixed by "#FROM SPO :".

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

REMOTE JOB ENTRY

P2626 RJE - "COPY" SYNTAX NOT CHECKING FOR BLANKS

RJE now allows blanks in the COPY syntax when the volume kind is being specified.

Example:

(KIND = PACK)

This example is now acceptable with blanks around the equal sign.

P2746 RJE - AUTOBACKUP OPTIMIZATION

The number of I/Os required to read backup files by RJE has been reduced by 50%. This should cause faster printing to RJE stations with high speed datacom lines and printers.

P2874 RJE - BAD QUEUED FILE TRANSFER COUNT

The queued file transfer count is now correct at restart. The QUEUECOPY procedure does not allow the queued count to be less than zero.

P2875 RJE - "TIME(1)" VS. "TIME(14)"

RJE now uses TIME(14) for all its internal timers instead of TIME(1) and TIME(11). This corrects several problems when the time of day on the host system was reset to an earlier time or the midnight time change occurred.

P2968 RJE - "B1800" WITH "B9247-13" TRAIN PRINTER

If the last two characters generated by RJE in the final print record are "RS" "0" (4"1EF0"), they are stripped from the message. This allows Model B9247-13 train printers to be made not ready. It has no effect on any other model printers.

P2969 RJE - "COMMENCEPF" SET ON WRONG PRINT QUEUE ENTRY

RJE would update an incorrect word in PRINTQARRAY if the same jobnumber were PBed more than once. This would cause the AUTOPRINT routine not to be able to skip forward or backward upon request. In addition, the duplicate request in the PRINT QUEUE would terminate with no BD file on DK or PK.

RJE now disallows entry into the PRINT QUEUE of a job already queued through the use of the command: "*PB <job number>". This command is still valid to move a PRINT QUEUE entry to the head of the queue, but results in the following error message if the entry is already at the head of the queue: "JOB ALREADY AT HEAD OF PRINT QUEUE".

P3076 RJE - "RJE" PROTOCOL VERSION

The protocol version byte in the STATION-ID (09) control message is now compatible with B1800 and B800 SYCOM.

P3077 RJE - "RJE PUTREPLY" VS. "SYCOM"

RJE's PUTREPLY (17) control message is now compatible with B1800 and B800 SYCOM.

P3078 RJE - BUFFER SIZE CONTROL MESSAGE "02, 04"

RJE now responds correctly with an 04 (BLOCKSIZE REPLY) control message upon receiving an 02 (BLOCKSIZE) control message.

The maximum value RJE allows for BLOCKSIZE is 1500. The first 02 control message received by RJE sets maximum BLOCKSIZE value for the duration of the session. The value can then be changed to any value between 132 (the minimum value) and the value specified in the first 02 message.

The BLOCKSIZE will revert to the default value of 400 at log-off.

P3079 RJE - "ABORT COPY" REQUEST VS. "SYCOM"

RJE's handling of ABORT COPY requests (18 and 19) control messages is now compatible with B1800 and B800 SYCOM.

P3126 RJE - PROTOCOL VERSION MISMATCH

RJE will not allow any file transfers or terminal-to-program connections between hosts if the protocol versions do not match.

P3151 RJE - "SM" COMMAND

An *SM command not followed by a verb or any text from a terminal transferred to RJE no longer hangs the procedure PERFORMSM.

P3202 RJE - FILE TRANSFER SECURITY

RJE's file transfer feature will correctly handle the source and destination files with regard to security. The following restrictions are now enforced:

1. A non-privileged usercode cannot use the "*" syntax to copy non-usercoded files. The following error message is displayed: "# FILE MUST BE IN OWN DIRECTORY".
2. A non-privileged usercode cannot use the "*" syntax to create a non-usercoded file. The following error message is displayed: "# FILE MUST BE CREATED IN OWN DIRECTORY".
3. A non-privileged usercode may only copy some other usercode files if the security of the file allows it. If the security requirements are not met, the following error message is displayed: "# FILE <filename> NOT RESIDENT".
4. A non-privileged usercode may not create a file under some other usercode. The following error message is displayed: "# FILE MUST BE CREATED IN OWN DIRECTORY".

None of the above restrictions apply to privileged usercodes.

P3345 RJE - EXTRA LINKFILE UPDATES REMOVAL

An extra update of the Linkfile has been removed from the code handling the *SM ODT command "BUG" and an extra update in "LOGENTRYON".

P3346 RJE - LINKFILE UPDATES

The linkfile updates caused by the File Transfer procedures FILEX and FILER now take place from the outer-block of RJE. This will eliminate corruption of the linkfile when both FILEX and FILER try to update simultaneously.

P3439 RJE - LOST AVAILABLE RECORDS

In some cases, RESTARTRECORDMANAGER would forget about a group of available records after a restart. This would cause the possible corruption of the link file if all RJE stations had current printer output. This problem has been corrected.

P3440 RJE - INVALID PRINTER CHARACTERS

The following hex characters have been added to RJE's invalid character list of translation to "?" (4"6F") for printer output:

"48", "49", "59", "69", "6A", "E0"

P3441 RJE - DISCONNECT OF SWITCHED LINES AT LOGOFF

RJE will now correctly disconnect a dialin line at logoff.

P3678 RJE - "PB" OF FILES WITHOUT SUMMARY

Entering "PB" of a printer backup file that had no summary file on a system without a pack called "PACK" would cause the Autobackup stack to hang on an RSVP "REQUIRES PK PACK" attempting to remove the nonexistent summary file from "PACK". This problem has been corrected by causing RJE to start its directory searches with the family "DISK" instead of the last family in the summary file search.

P3682 RJE - REMOVE SUMMARY FILE

RJE will not correctly remove the summary file when it is now on the same family as the backup files.

P3753 RJE - "RS0" IN PRINTER BACKUP RECORD

RJE was incorrectly positioning its output pointers after skipping an "RS0" in a printer backup record. This problem has been corrected.

P3754 RJE - ABORT COMPATIBILITY

RJE is now compatible with B1800 and B800 SYCOM in its handling of file transfer ABORT messages 18 and 19.

P3777 RJE - FILE TRANSFER INPUT BLOCK SIZE

RJE now allows input file transfer block sizes up to 3000 characters automatically to accommodate B1800 and B800 SYCOM. The FYBLKSZ value of the SF and TF RJE commands now only affect the output file transfer blocksize.

P3778 RJE - "##RJE" MESSAGE REMOVED

All "##RJE" messages that were issued at logon have been removed for compatibility with B1800 and B800 SYCOM.

P3779 RJE - OBJECT SYSTEM IN SYMBOLIC HEADER

RJE now enters the correct object system model in the file transfer symbolic header. B6800 was previously entered.

P3804 RJE - STATION LOGOFF AT "RJE" "QUIT"

RJE now logs-off all stations when the SM QUIT command is used, thus allowing a cleaner restart when RJE is subsequently executed.

P3805 RJE - "22 CONTROL MESSAGE NO-OP"

If RJE receives a 22 control message (terminal disconnect) for a station not connected, it will be ignored.

P3806 RJE - COPY REQUESTS REJECTED

RJE now rejects all copy requests until a 23 (EOF confirmation) or an 11 (FETCH recovery) control message has been received. This corrects a timing problem with file transfer during restart of RJE. The copy requests are rejected with the following message:

"HOST <hostname> IS NOT FILE TRANSFER CAPABLE-INITIALIZATION NOT COMPLETE"

P3807 RJE - PARITY ERROR ON "REMLP" FILES

REBUILDPRINTQUEUES no longer DS when they encounter a parity error reading a REMLP file. The file causing the error is removed.

P3808 RJE - "AUTOPRINT" "INVALID INDEX"

It was possible for AUTOPRINT to fault DS with an INVALID INDEX when an "END" banner was being printed. This no longer occurs.

P3809 RJE - SEND CONTROL LENGTH UPDATE

An erroneous reference to the remote ODT's terminal width when updating a message length has been corrected.

P3810 RJE - LENGTH OF "*RS" REPLY

Procedure RSVP now correctly calculates the length of an ODT reply to a *RS command. Previously, an extra null character was appended to the end of the reply. In some cases, this would cause a B771 terminal to hang.

P3811 RJE - INCORRECT BACKUP FAMILY

A problem, where RJE AUTOBACKUP failed to find CANDE backup files due to an incorrectly-set FAMILYNAME file attribute, has been corrected.

SCRMCP

An Initialize, Verify, and Relocate (IVR) facility has been implemented in MAINTENANCE for the Model 206 Disk Pack Drive and the Model 207 Fixed Drive Disk. The IVR facility has also been implemented for Model 235 Disk Packs. However, 235 packs may only be IVRed a cylinder at a time; i.e., "ALL" may not be used in the IVR statement. This facility establishes sector addresses and boundaries, verifies the media, and relocates erroneous sectors.

There is no standalone facility for IVR.

A new construct in the Maintenance And Test (MAT) language has been added to allow easy initiation of the IVR Facility. The syntax is:

```
-- IVR --
```

```
| | <-----|  
-- /1\ PK -----<pack number>---  
      |- = -|  
- /1\ SERIAL -----<serial number>---  
        |- = -|  
- /1\ CYLINDER -----<cylinder number>-  
          |- = -|  
    - ALL -----  
- /1\ NAME -----<name>---  
      |- = -|   | " ---<name>--- "
```

This construct is an <I/O statement>, and may appear anywhere in a MAT program where the <I/O statement> is allowed. It may not appear in a program loop and none of the numbers used as parameters may be variable.

The IVR Function may be initiated simply by entering at the ODT "SCR IVR". The IVR Facility will respond with requests for any additional required data.

Note: Prior to an IVR, the pack must be closed and reserved (via the CLOSE and UR ODT commands; see Mark 31 GENERAL note D2535 for details). This implies that the pack is not in use.

Note: If cylinder 0 or the entire pack is IVRed, the label area on the pack is overwritten. When the pack is made ready, this label will cause a "NO VOL1" message to be displayed, and the pack status will be "LABEL ERROR". It is necessary to reconfigure the pack (via the RC ODT command; see Mark 31 GENERAL note D2535 for details) to label it properly and make it operational.

The operator can enter the following requests during IVR execution (where <mix number> is the mix number of MAINTENANCE):

- a. `<mix number>AX STATUS`

This request will cause IVR to display the "status" message described in "IVR Messages".

- b. <mix number>AX QUIT

This message will cause immediate normal termination of IVR execution.

- c. `<mix number>DS`

This message will cause immediate abnormal termination of IVR execution. All IVR processing

stops upon detection of the DS message.

IVR MESSAGES

During IVR the following messages will be displayed on the ODT:

a. IVR PKuuu: INITIATED(PACK IS OF TYPE <type>)

This message marks the start of IVR execution and specifies the type of pack being IVRed.

b. IVR PKuuu: I/O ERROR(CYLcc,HDhh,SECTORss,<opcode> OP, RD=<result descriptor>)

This message is displayed for every I/O error encountered during IVR execution.

c. IVR PKuuu: CYLcc,HDhh,SECss RELOCATED

This message is displayed each time the IVR Facility relocates a sector that has failed verification. The failing sector is specified in cylinder/head/sector format.

d. IVR PKuuu: COMPLETED @ <time>

This message indicates normal termination of IVR.

e. IVR PKuuu: ABORTED @ <time>(<abort cause>)

This message indicates an abnormal termination of IVR and is caused by an operator DS/QUIT or by an irrecoverable error encountered during execution.

f. IVR PKuuu: STATUS @ <time>(CYLcc, RELOCrr, APPROX END TIME <time>)

This message displays the current status of IVR execution. It is displayed in response to an operator "STATUS" request, and specifies the cylinder currently being IVRed, the number of sectors relocated during the total IVR, and an approximate end time.

g. IVR PKuuu: THIS PACK IS OWNED BY "<owner>"
ENTER OK (TO OVERRIDE) OR QUIT

This message is displayed whenever a pack to be IVRed contains an owner in the label. IVR will not proceed until the operator enters "<mix number>AX OK".

IVR LOGFILE

During IVR execution all operational events are logged into a printer file. This file will be printed at the completion of the MAT program. The format of a log file entry is:

<time> <entry type> <text>

There are six different log entries:

- a. BOT marks the beginning of the task.
- b. INP records all input parameters to IVR.
- c. OPR records all input requests from the operator.
- d. MSG records all output from the IVR Facility.
- e. ERR records all errors detected during IVR execution.
- f. EOT marks the end of IVR execution.

FATAL ERRORS

The following is an explanation of errors that will cause abnormal termination of IVR execution:

a. IVR PKuuu: ILLEGAL UNIT NUMBER

The specified unit number is outside the permissible range of 1-255.

b. IVR PKuuu: UNIT NOT READY, NOT CLOSED OR NOT RESERVED

The pack must be closed, in a reserved state and physically ready before an IVR may occur.

c. IVR PKuuu: INVALID PACK TYPE

The IVR Facility was unable to identify a valid type of pack.

d. IVR PKuuu: UNIT ASSIGNED TO ANOTHER PROGRAM

The specified pack is already in-use by another MAT program.

e. IVR PKuuu: ILLEGAL CYLINDER NUMBER

The specified cylinder is either less than 1 or greater than the maximum allowed for the specified pack.

f. IVR PKuuu: FULL-PACK IVR NOT VALID FOR THIS DISK PACK TYPE

"ALL" was entered instead of "CYLINDER=cc". Model 235 packs may only be IVRed a cylinder at a time.

g. IVR PKuuu: INVALID SYNTAX

The operator entered data in an illegal format.

h. IVR PKuuu: I/O ERROR(CYLcc,HDhh,SECTORss,<opcode> OP, RD=<result descriptor>)

Any fatal I/O error will cause IVR to terminate. The operation causing the failure is displayed along with the failing result descriptor.

D3072 SCRMCPC - "BUFFMEM" MODIFIER ADDED

A new I/O modifier, BUFFMEM, has been added. When a READ or WRITE to a 7A magtape control is modified by BUFFMEM, the system will read from or write into its control buffer memory.

The following changes should be made to the On-Line Maintenance and Test (MAT) Language Manual (Form No. 5000169):

Page 7-14:

<read mt modifier> syntax should read as follows:

```
"<READ MT MODIFIER>::=<STANDARD I/O MODIFIER>/
<DENSITY I/O MODIFIER>/<PARITY I/O MODIFIER>/
<DIRECTION I/O MODIFIER>/
<BUFFER MEMORY I/O MODIFIER>"
```

Page 7-35:

<write mt modifier> syntax should read as follows:

```
"<WRITE MT MODIFIER>::=<STANDARD I/O MODIFIER>/
<DENSITY I/O MODIFIER>/<PARITY I/O MODIFIER>/
<BUFFER MEMORY I/O MODIFIER>"
```

Section 8:

Add the following to the description of "I/O MODIFIERS":

"BUFFER MEMORY I/O MODIFIER

SYNTAX:

<BUFFER MEMORY I/O MODIFIER>::=BUFFMEM/<EMPTY>

EXAMPLES:

1. BUFFMEM . . .

2. . . . BUFFMEM

SEMANTICS:

THE <BUFFER MEMORY I/O MODIFIER> CAUSES A READ OR A WRITE TO BE PERFORMED ON A 7A MAGTAPE CONTROL BUFFER MEMORY.

BUFFMEM MAY ONLY BE USED WITH A 7A MAGTAPE CONTROL."

D3169 SCRMCPC - LOCAL "MCP" CODE

A large area of MCP save code is used as a test pattern source for disk tests 14, 15 and 16, as well as diskpack tests 4, 5 and 7. Because of a reorganization of MCP save code, there have been some changes to these tests. The changes have no major effect on the operation of the tests, but do affect the documentation.

These tests all use MCP save code as the test pattern. This code is no longer in "segment 5" and all references to "seg 5" in the documentation should be eliminated. On a B6800 multiprocessor (tightly-coupled) system, there is some MCP save code in each local memory subsystem and some in the Global subsystem; SCR will use the save code in the subsystem in which the SCR stack is running. That is, if SCR runs (by default) in a local processor, that processor's local save code is used, but if SCR is forced to run in * GLOBAL tm Memory, the Global save code is used. (The base pattern from a given MCP code file is the same in a monolithic system as in any local subsystem; the base pattern in Global memory is different MCP code, so it will have similar bit distributions but not identical bit sequence. The actual

pattern written is varied from one test to another by varying the starting point in the base pattern.)

* "GLOBAL Memory" is a trademark of Burroughs Corporation.

The documentation for these tests refers to a constraint that the test area must be at least 20% as long as seg 5. This constraint was not properly documented; it actually concerned the length of some overhead space associated with SCR's use of seg 5. The constraint has been rendered inapplicable and has been removed; all documentary references to it should be eliminated.

D3256 SCRMCPC - "IVR" FOR "215,225" PACKS

Initialization and verification (IV) of Type 215 and 225 packs has been subsumed by the IVR maintenance facility. However, the IVR syntax for Type 215 and 225 packs differs from the syntax for Type 206, 207 and 235 packs. The syntax is a subset of the options which are available for IV. The excluded options may be used when RCing the pack.

Note: Until the Mark 33 release, IV can be used; effective with the Mark 33 release, IVR must be used. The following message is issued if IV is attempted:

"WARNING: IV OPERATIONS MUST BE DONE VIA SCR ON THE 33 RELEASE".

Note: Only one IVR statement for a Type 215 or 225 pack is allowed in a SCR program.

IVR Syntax for Type 215 and 225 Packs

```

-----
-- IVR -- PK -----<pack number>----->
          | - = - |
          |
>-----|
| <-----|
| /1\- SERIAL -----<serial number>-----|
|          | - = - |
| -/1\- NAME -----<name>-----|
|          | - = - | | - " --<name>-- " -|
|          | <-----|
|          | <----- , -----|
| -/1\- XD -----/52\--<cylinderhead>-----|

```

The IVR statement is used to format and analyze all of the tracks of the given pack. The master available table, volume labels and pack directory are also created and written on the pack.

If the NAME = or SERIAL = options are omitted, the corresponding value is set to blanks or zeroes, respectively. XD cylinderhead is of the form CCC HH, where CCC is the cylinder address (0-405) and HH is the head number address (0-19) of the track to be XDed because it is defective. Counting begins at the bottom head.

Example:

```
IVR PK 96 NAME=JUDP SERIAL=123456 XD 20 5, 300 19
```

The IVR statement causes two passes to be made over the designated packs. During the first pass, an initialize command is used to preformat each track with 30-word records. During the second pass, a verify command is used to test each record written during the first pass. The IVR routine attempts to relocate bad sectors to the spares provided on track zero of each cylinder. If this cannot be done, the track is automatically XDed. A report is produced that indicates I/O errors encountered, sectors relocated, and what tracks and cylinders were XDed.

If an IV is attempted on a Type 235, 206 or 207 pack, the following message is displayed:

"PK<nnn> NOT PACK TYPE 215 OR 225 - SHOULD IVR WITH SCR"

D3342 SCRMCP - "IVR" TYPE "206" PACKS

Previously, the default pattern used in initializing type 206 disk packs was 4"6DB6" (previous release "MD") and the patterns "6363" and "9C9C" were used for write verification. Now, the default pattern used in initializing type 206 disk packs on CONTROLLER release "MD" is "6363"; therefore, only pattern "9C9C" is used for write verification.

D3345 SCRMCP - "IVR" TYPE "207" PACKS

IVR will now retry errors which are not address errors or data errors three times if the pack is a Type 207 pack; otherwise, errors will be retried only once. Data-corrected errors are not really errors; they are not retried.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

SCRMCP

P2794 SCRMCP - PACK DENSITY NOT ESTABLISHED PROPERLY

Under certain circumstances, the pack density would not be determined correctly in SCR (e.g., the pack was reserved and write-disabled at Halt/Load time). This problem has been corrected.

P2812 SCRMCP - CORRECT FAULT MESSAGE

Previously, a fault in MAINTENANCE would print the wrong RCW where the fault occurred. Now, the correct RCW is printed, and the sequence number where the fault occurred is printed if LINEINFO was set.

P2853 SCRMCP - "INVALID OP" FOR "COMPARE BUFFER ON ERROR"

The system would get an INVALID OP when a compare error occurred doing "COMPARE BUFFER ...ON ERROR (<compare error action list>)" and <compare error action list> contained "PRINT DATA" but did not contain "PRINT RESULTS". This problem has been corrected.

P2858 SCRMCP - "7A" MAGTAPE CONTROLS

Tape erases have been broken into 1600-word segments.

P2891 SCRMCP - STOP REPEATING FIRST LINE OF "ODT" INPUT

Previously, MAINTENANCE would print the first line twice in a program listing if it was initiated from the ODT. This problem has been corrected.

P2922 SCRMCP - MEMORY MODS FOR MAINTENANCE, "GMM" INTRINSICS

The bookkeeping for maintenance testing of memory mods has been corrected for B6800 Multiprocessor systems. The SYSTEST/GMM routines are affected, as well as the "BUFFER...ADDRESS=..." constant in SCR; no change has occurred for correctly-used tests.

The intrinsics GLOBALMEMORYTEST and GMMSCAN are now incorporated in the MCP.

P2923 SCRMCP - "SCR/MCP" VERSION

The code to learn and display the PATCHNO in the MCP level for SCR has been corrected. The separate MAT level is no longer displayed (it has to be the same as the MCP).

P2957 SCRMCP - BUFFER WITH ADDRESS SPECIFIED

Accessing a buffer declared at a specific address in memory would cause a SEG ARRAY error. This problem has been corrected.

P3005 SCRMCP - WRONG DENSITY FOR FILE "I/O"

The wrong density was used for an I/O with a disk or pack file specified. This problem has been corrected.

P3049 SCRMCP - "IVR" WRITE DISABLED PACK

IVRing a write-disabled pack no longer gives a WRITE LOCKOUT error when reading the pack.

P3069 SCRMCP - "VERIFY DISKPACK (SELECT ALL REPEAT)"

Previously, when verifying a diskpack, selecting all tests and repeating them a number of times, all of the tests would be done only once. This problem has been corrected; now, all the tests are repeated the specified number of times.

P3123 SCRMCP - WARNING MESSAGE

Previously, if two successive errors occurred while reading the label on a dedicated unit or on a pack being IVRed, MAINTENANCE would give a confusing warning message. This problem has been corrected.

P3459 SCRMCP - "AX" MORE THAN "60" CHARACTERS FOR "IVR"

Previously, during a maintenance IVR of a pack, entering "<job #> AX <msg>" at the ODT, where <msg> exceeded 60 characters, caused a SEG ARRAY fault. This problem has been corrected; now, up to 80 characters may be entered.

DOCUMENT CHANGES NOTES (D NOTES)

PERIPHERAL TEST DRIVER (PTD)

D3123 PTDTEST - "B6900" PERIPHERAL TEST DRIVER

1. PTD EXECUTION

On the B6900 the Peripheral Test Driver is a selectable module whose equivalent on the B6700, B6800 is SCR. "PTD" is recognized by the WFL compiler, running on a B6900, as the command to start SYSTEM/MAINTENANCE as an independent runner. "PTD" is not recognized by old WFL which means that either option 29 (CONTROLOLDWFL) must be set or the request to initiate PTD must be preceded by "BEGIN JOB;".

PTD is an MCP procedure that interprets op-codes ("S-ops") found in the test case S-code file selected by the user. Thus, to initiate a test on a particular type of peripheral/DLP, the user must label-equate the appropriate IOSO-provided test case code file to PTD's internal SCORE file.

In the Large Systems implementation, PTD can be operated from either an ODT or a datacom terminal. In either case, all PTD input and displays are through a logical file titled PTDSPO; the use of the "<mix #>AX" construct is eliminated. With the ODT, however, every message input to PTD must be preceded in column one by the triangular group-separator character (GS) so that the entry can be directed to PTD instead of to the CONTROLLER. To direct the operator dialogue to a datacom station, the user need only change the KIND of PTDSPO to REMOTE and the TITLE of PTDSPO to the station name; the group-separator is not used with this medium.

PTD can be executed with a VALUE clause that sets certain of its options immediately at run time. This feature is useful primarily in debugging test cases or PTD itself. The relevant bits are:

[2:1]	TRACE	(test case S-ops in execution)
[1:1]	DUMP	(test case virtual memory dump)
[0:1]	PGMDUMP	(program dump of PTD)

Hence, executing PTD with VALUE = 2 causes the DUMP option to be set, VALUE = 3 sets both DUMP and PGMDUMP, etc.

GENERAL EXECUTION SYNTAX:

In general, the Large Systems PTD is executed by entering a command of the form:

PTD; FILE SCORE(TITLE = <S-code file name>)

Other items are entered depending on the setting of option 29, the user's choice of dialogue device, and the user's (rare) decision to add the VALUE clause. Actual SCORE file names are used in the following examples. See the field support kit provided by IOSO for detailed descriptions of the release tape, file naming conventions, and individual test case documentation.

ODT EXECUTION EXAMPLES:

PTD is executed at an ODT and all subsequent operator dialogue occurs at that ODT. All three forms are acceptable if CONTROLOLDWFL (option 29) is set; only the third form ("BEGIN JOB; . . .") is acceptable if CONTROLOLDWFL is reset.

PTD; FILE SCORE(TITLE = PTD/MAINT/CP)

PTD; FILE SCORE(TITLE = PTD/CONF/CR ON DMS); VALUE = 1

BEGIN JOB; PTD; FILE SCORE(TITLE = PTD/MAINT/PE); VALUE = 2

ODT INITIATION, REMOTE (DATACOM) DIALOGUE DEVICE EXAMPLES:

PTD is executed at an ODT but all further dialogue occurs at the specified datacom terminal. All three forms are acceptable if CONTROLOLDWFL (option 29) is set; only the third form ("BEGIN JOB; . . .") is acceptable if CONTROLOLDWFL is reset.

```
PTD; FILE SCODE(TITLE = PTD/MAINT/HTS ON DISK);  
      FILE PTDSPO(KIND = REMOTE, TITLE = TD450365)  
  
PTD; FILE SCODE(TITLE = PTD/CONF/PSS); STATION = 55; VALUE = 3  
  
BEGIN JOB; PTD; FILE SCODE(TITLE = PTD/CONF/PE); STATION = 55
```

REMOTE EXECUTION EXAMPLES:

PTD is executed at a logged-on CANDE terminal and all subsequent operator dialogue occurs at that terminal. Notice that "BEGIN JOB;" is required, irrespective of the setting of CONTROLOLDWFL.

```
WFL BEGIN JOB; PTD; FILE SCODE(TITLE = PTD/MAINT/PE);  
      VALUE = 1  
  
WFL BEGIN JOB; PTD; FILE SCODE (TITLE=PTD/MAINT/PE)  
  
WFL BEGIN JOB; PTD; FILE SCODE(TITLE = PTD/MAINT/HTS  
      ON TESTPACK);
```


2. PTD DIRECTIVES

When executed, PTD first performs some internal initialization steps that may take a few seconds to accomplish depending on the size of the test case code file and the number of other jobs in the mix. As noted elsewhere, PTD displays the testcase file name and release-level data and then enters its idle state after displaying:

AWAITING DIRECTIVE

Any valid PTD directive can now be entered, one directive per transmission. PTD recognizes any number of correctly spelled characters from the minimum (underlined>) set to the fully formed word; i.e., PRI, PRIN, and PRINT are acceptable, while PR and PRINNT are not. The directives fall into four general categories and are summarized below.

Remember that if the ODT is used, the group-separator character (GS) must appear in column one.

A. TOGGLES

These are options that are either on or off. To set a toggle the user enters the option name only; to reset a toggle the user enters the option name followed by "O", "OF", or "OFF". Hence, "PRI" sets the PRINT toggle; "PRI O" or "PRINT OFF" resets it.

DISPLAY - Messages are displayed to the ODT or remote terminal.
 --- When the medium is an ODT, resetting DISPLAY causes PTD to close its PTDSPO file so that normal system display traffic may resume.

Default: DISPLAY IS SET.

DUMP - Prior to EOJ, the test case's data memory area is dumped
 --- to a printer file. Set this option for documentation of test case or PTD problems.

Default: DUMP is RESET.

IOTRACE - I/O information is displayed before and after every test
 --- case I/O operation. Prior to initiating the I/O, the MLI REQUEST and IOLENGTH are shown. After the I/O has completed, PTD displays the MLI RESULT, the number of data-bytes transferred, and an "MLIP" result.

The user should be aware that whenever the words "MLIP RESULT" are used in the context of PTD or an IOSO test case, they refer to a value constructed strictly for the purposes of a programmatic, host independent result descriptor that summarizes errors that occur within the host to MLI interface. On Large Systems, the true MLIP result descriptor can be obtained only by inspecting the test case IOCB(s) as they complete. The address of the IOCB is made available through the use of the special I/O options, described in Section 5 below.

Default: IOTRACE is RESET.

LOG - to be specified.

Default: LOG is RESET.

MONITOR - All display traffic is written to a printer file and is
 --- formatted such that all operator entries are flagged with an asterisk ("*") in column one.
 (The MONITOR and PRINT functions are soon to be merged into the PRINT toggle; for now, the user should set both MONITOR and PRINT.)

Default: MONITOR is RESET.

PGMDUMP - Prior to EOJ, PTD dumps itself (via PROGRAMDUMP) to a printer file. Set this option for documentation of test case or PTD problems.

Default: PGMDUMP is RESET.

PRINT - PTD prints test case specified text to a printer file when the PRNT S-op is encountered. (See the note in the description of MONITOR, above.)

Default: PRINT is RESET.

TRACE - PTD writes each test case S-op to a printer file after the S-op is executed. It is used for test case and PTD debugging. Since vast numbers of S-ops are executed in the usual test case, tracing them may waste storage space and paper.

DEFAULT: TRACE is RESET.

B. I/O ERROR CONTROL

Generally, the test case issues the I/O (via the EXIO S-op), waits for it to complete (IOCW S-op), and then checks the results. If it finds a result descriptor, data, or length error, it notifies PTD of the error via the EROP S-op.

When PTD encounters the EROP S-op, it either stops and idles or ignores it, depending on what the user has directed. By default, PTD halts on the EROP S-op.

ERROR HALT

--- - PTD halts on any non-fatal I/O error and displays:

STOPPED ON ERROR

At this point, any valid PTD directive may be entered. (The GO directive, explained below, causes PTD to continue execution at the very next S-op.)

ERROR IGNORE

--- - PTD treats all non-fatal test case I/O errors as no-ops; execution continues uninterrupted.

Note: The test case may retry an I/O as part of its testing algorithm, but no I/O is ever automatically retried by PTD or the MCP.

C. PTD STATUS

The current status of PTD is obtained via the STATUS directive.

STATUS - PTD displays the number of the section currently in execution, the toggles that are set, and the setting of the ERROR option.

D. SECTION CONTROL DIRECTIVES

This class of directives controls the actual execution of the test case. As described elsewhere, test cases are organized into numbered "sections" that perform certain operations. There are groups of sections that fall into logical categories called "blocks".

For example, block 100 may consist of sections 101, 102, and 103, where those particular sections test the TESTID, the READBUFFER, and the WRITEBUFFER operations respectively. Other blocks may consist of sections designed to perform complex sequences of data transfer that drive a peripheral or DLP at a certain level of tolerance. Sections from 3000 above are "special" sections which must be explicitly invoked by the user; section 4000 lists what the other sections do and is particularly useful. Refer to the IOSO-provided documentation for precise descriptions of the organization of each test case.

The RUN, REPEAT, and GO directives place PTD into immediate execution. When execution of the specified section(s) has been completed, PTD displays:

AWAITING DIRECTIVE

and idles until a new directive is entered.

RUN <section list>

PTD executes those sections or ranges of sections contained in the <section list>. The section list may be empty, or the list may contain a single section number, or a range of sections, or it may contain up to ten sections and ranges separated by commas.

If only the word "RUN" is entered, PTD ALWAYS executes sections 100 to 2999. (Recall that sections from 3000 above must be deliberately specified.)

Examples: RUN
 RUN 3999
 RUN 200
 RUN 101, 201
 RUN 203 TO 207, 301-302, 4000-END

- Rules:
- The maximum number of pairs of section numbers that can occur in a <section list> is 10. (A single section number, 305 say, is treated as the pair: 305-305.)
 - In any section pair, the second number must be greater than or equal to the first.
 For example: "RUN 300-101" is an error.
 - The word END can appear only as the last item in a <section list> and must be preceded by "<number> -" or "<number> TO".
 For example: "RUN END" and "RUN 101, END" are both erroneous entries.
 - The word END is interpreted to mean "the very last section of the test case". When END appears in a RUN directive, PTD goes to end of job when execution of the final section has completed.

REPEAT <section list>

or

REPEAT <section list> FOR <count>

- The <section list> is repeated (looped on) either an infinite number of times or <count> times. As in the RUN directive, if no list follows the word REPEAT, sections 100-2999 are automatically executed.

Examples: REPEAT
 REP F 3
 REPEAT 102 FOR 10
 REPEAT 101, 301, 102, 302
 REPEAT 207-209, 101, 305 - 400

- Rules:
- The maximum allowable value of <count> is 999999999.
 - The RUN <section list> rules also apply to the REPEAT directive, with the single exception that the occurrence of END does not cause PTD to go to EOJ.

GO

--

- When PTD either stops on an error (see ERROR directive) or is interrupted by the user (see item 5, PTD OPERATOR INTERRUPTION), the GO directive causes PTD to resume execution from the point where it was stopped. If the user had entered, say, "RUN 304", PTD would continue running section 304. Similarly, if PTD had been REPEATING a list, GO would cause it to continue exactly where it left off. However, if PTD has finished executing a section list, it responds to "GO" by displaying:

SECTION LIST COMPLETED. USE 'RUN' OR 'REPEAT'.

Hence, "GO" always means "continue", and there has to be a function which PTD can continue.

QUIT - Terminates PTD.

474

3. SPECIAL LARGE SYSTEMS PTD DIRECTIVES

There are Large Systems PTD directives that are intended primarily as aids in developing and debugging PTD and test cases. They may be useful in observing the flow of test case I/Os and have certain value (see IOSTOP) in stepping an I/O through the processor.

With the exception of IOSTOP and LINES, the setting of a special option results in the analysis by PTD of the test case I/O currently in process at a selectable point in the execution of the test. Relevant DEVICESTRUCTURE information is displayed on the ODT or remote terminal. (A DEVICESTRUCTURE is the programmatic I/O interface between the test case and PTD and is analogous to a FIB, or File Information Block. Up to ten separate DEVICESTRUCTURES may be declared by the test case writer in the PTL (Peripheral Test Language) test case program; each carries all information relating to the I/O device, the I/O request, and the result of the operation.)

In every case, an asterisk must appear before the non-standard option name; they are set and reset like all other PTD flags (see item 2A, TOGGLES above).

*HELP IODEBUG -

- PTD displays explanation of special I/O options.

*BOJ

- Display each DEVICESTRUCTURE the first time it is touched after this option has been set.

*BOPEN

- Display before each DEVICESTRUCTURE is opened.

*AOPEN

- Display after each DEVICESTRUCTURE is opened.

*BIIO

- Display before the I/O is initiated out of each DEVICESTRUCTURE.

*AIIO

- Display after the I/O has completed.

*BCLOSE

- Display before each DEVICESTRUCTURE is closed.

*ACLOSE

- Display after each DEVICESTRUCTURE is closed.

*EOJ

- Display each DEVICESTRUCTURE immediately prior to EOJ.

*IOSTOP

- At each user-selected analysis point, PTD programmatically stops after the DEVICESTRUCTURE display; entering any input or null reactivates PTD. If *BIIO has been set in addition to *IOSTOP, then after the I/O information display and attendant stop/start, PTD executes a conditional halt prior to firing the I/O (CHLT must be set on the machine). 4"IOCBIOCBIOCB" is placed in the B-register and the address of the IOCB to be fired is placed in the A-reg.

*ALL

- Sets all of the above flags.

*IODEBUG-

Setting any of the above sets this toggle. Resetting this toggle disables all analysis, but the settings of all other toggles are preserved. If IODEBUG is subsequently set again, the flags are restored and the analysis resumes.

In addition to the above toggles, the operator may specify the number lines of data to be displayed in the analysis output, allowing the user to view an entire data buffer on request.

B6000 SERIES MARK 32

*LINES <n> - sets the number of lines of data to be displayed,
----- 40 characters per line. Default is one line.

All non-standard options are reset by default.

4. TEST DEVICES

A test case is written with either a physical device or a baddisk file as its target. The connection between the test case and test object is established by way of:

- a. A programmatic test case data structure called a DEVICESTRUCTURE,
- b. The OPEN S-op,
- c. A device mnemonic or disk file name entered by the user.

Once running, PTD eventually encounters an OPEN S-op, whereupon it displays:

ENTER DEVICE FOR <peripheral designation>

or

ENTER FILE NAME FOR <baddisk designation>

where the <peripheral designation> or the <baddisk designation> is merely a string of characters that PTD finds in the DEVICESTRUCTURE and displays on the screen. For example, in the cardreader test, the display looks like:

ENTER DEVICE FOR CARDREADER

In the case of a peripheral/DLP test, PTD expects the user to enter a Large Systems unit mnemonic and optional path specification. The general form of the response is the following:

<unit> VIA PATHID <pathid>

--- ----

Examples:

CR12
PK57 VIA PATHID 17

For baddisk testing, PTD expects the title of a valid Large Systems baddisk file, including family name, of the form

<file title> ON <family name>

--

Example:

BADDISK/FMLYINX1/UNIT192/AD405000 ON PACK

Note: the "VIA" construct is not allowed for baddisk testing as only the spindle is assumed to be involved.

PATH SELECTION

The importance of a path selection capability is made obvious by the maintenance test cases, where a failure in a particular DLP (path) is the reason for running the test.

On the Large Systems PTD, path selection is achieved by use of the word "VIA"; if only the word "VIA" follows the <peripheral designation>, PTD displays all paths to the specified unit and then asks the user to choose a path. If "VIA PATHID <pathid>" is entered, PTD issues all test case I/Os through that path. If "VIA" does not appear at all, PTD chooses an appropriate available path and uses it for as long as that DEVICESTRUCTURE is open.

Example responses for the device-request in a PE tape test are:

B6000 SERIES MARK 32

MT17 % PTD selects an available
 % path to the tape drive.

MT17 VIA % PTD shows all paths and then
 % polls the user for a selection.
 % (The form of the path
 % information display is similar
 % to that given in response to
 % the "OL" ODT message.)

MT 17 VIA PATHID 5 % All I/Os are through the
 % path (DLP) whose path ID is 5.
 % "5" was a PATHID displayed
 % when the user typed "OL MT17"
 % to the system.

DEVICE STATUS RULES

For the Mark 32 MCP release, PTD requires that devices be in a secure status before online testing can commence. Therefore, for maintenance tests (which are defined to be DLP-oriented):

- a. The target unit and all units outboard the DLP must have been reserved through the "UR <unit>" ODT message.
- b. The path (DLP) must itself be reserved by way of the "UR <unit> PATHID <path id>" ODT message.

PTD ensures that the device is in the proper status before it completes the OPEN and notifies the user of any inconsistency.

5. PTD OPERATOR INTERRUPTION

PTD operates in a "fetch S-op/execute S-op" cycle and is sensitive to an operator interrupt request before each fetch operation. An interrupt request is serviced only when all outstanding test case I/Os have been completed. When the operator-dialogue device is an ODT, PTD is interrupted by entering:

<SYSTEM/MAINTENANCE mix number> HI

Example:

1127 HI

When the operator dialogue is through a REMOTE station, the "HI" statement must be preceded by a question mark in column one.

Example:

?9142 HI

Once interrupted, PTD responds by displaying:

AWAITING DIRECTIVE

At this point, any valid PTD directive is accepted.

6. TESTCASE EXAMPLE

The following is an example of the printed output of the PE tape DLP maintenance test showing the use of the RUN, REPEAT, and GO directives, and illustrating one of the path selection techniques. PTD was operated at an ODT and, with CONTROLOLDWFL reset, was executed by entering:

BEGIN JOB; PTD; FILE SCODE(TITLE = PTD/MAINT/PE)

After the AWAITING DIRECTIVE message appeared, MONITOR was set -- all subsequent operator entries are flagged with the asterisk in column one. When "MT17" was first entered, the unit was found not to be reserved; the operator reserved it with the "UR MT17" ODT message (directed to the system by omitting the GS character). Next, "MT 17 VIA" was entered to PTD, resulting in the path information display and path selection query. When the device was successfully open, the test case later reported errors that were due to the operator not having mounted a tape on the drive. With the exception of the operator dialogue, all messages are produced are produced by the test case.

LISTING FOR TESTCASE=PTD/MAINT/PE
AWAITING DIRECTIVE

```
*PRINT
AWAITING DIRECTIVE
*RUN 101
STARTING SECTION 0101
--> ECHO OP - ALL BITS OFF
ENTER DEVICE FOR MAGTAPE
*MT 17
UNIT MT 17 : NOT RESERVED.
ENTER DEVICE FOR MAGTAPE
[COMMENT: operator reserved MT 17]
*MT 17 VIA
```

```
UNIT 'MT17':
DLP ID =04
BASE NUMBER = 000
RELATIVE UNIT (W.R.T. DLP) = 1
PATH INFORMATION
-----
```

PATHID	PROC	MLIPPORT	LEMPORT	DLPNUM	PATHSTATUS
17	3	1	0	6	RESERVED, ONLINE

ENTER DESIRED PATH NUMBER

```
*17
AWAITING DIRECTIVE
```

```

*REPEAT 102 FOR 3
  STARTING SECTION 0102
  --> ECHO OP - ALL BITS ON
  STARTING SECTION 0102
  --> ECHO OP - ALL BITS ON
  STARTING SECTION 0102
  --> ECHO OP - ALL BITS ON
  AWAITING DIRECTIVE
*RUN 200-END
  STARTING SECTION 0200
  --> OP CODES TEST
  ### TEST SEL = 0200 TEST RUN = 0201 #####
  OPCODE+VAR = 2F1000
  OPCODE = 2F0000 = TEST
  + UNIT NUMBER = 1(01)
  & IDLENGTH = 0006 CHARACTERS
  CYCLE=001 I/O=00013 ERR:MLIP=000 L=000 RD=001 DATA=000
  RESULT DESCRIPTOR IS 41100000
  RD WD1 4000 = DESCRIPTOR ERROR
  RD WD1 0100 = TAPE UNIT NOT READY
  RD WD1 0010 = WRITE LOCKOUT
  RESULT DESCRIPTOR EX 00800000
  RD WD1 0080 = BOT (BEGINNING OF TAPE)
  ### RD ERROR ##### RD ERROR #####
  STOPPED ON ERROR
*GO
  ### TEST SEL = 0200 TEST RUN = 0201 #####
  OPCODE+VAR = 2F1000
  OPCODE = 2F0000 = TEST
  + UNIT NUMBER = 1(01)
  & IDLENGTH = 0006 CHARACTERS
  CYCLE=002 I/O=00014 ERR:MLIP=000 L=000 RD=002 DATA=000
  RESULT DESCRIPTOR IS 41100000
  RD WD1 4000 = DESCRIPTOR ERROR
  RD WD1 0100 = TAPE UNIT NOT READY
  RD WD1 0010 = WRITE LOCKOUT
  RESULT DESCRIPTOR EX 00800000
  RD WD1 0080 = BOT (BEGINNING OF TAPE)
  ### RD ERROR ##### RD ERROR #####
  STOPPED ON ERROR
*QUIT
  *** EOJ PTD ***

```

D3199 PTDTTEST - "PTDTTESTS" TAPE

A tape, "PTDTTESTS", is included in the B6000 series support release. It contains files required by hardware support personnel for maintenance and confidence testing of the new I/O facilities on the B6000 series systems.

The files on the PTDTTESTS tape may be categorized as follows:

1. Confidence tests, PTD/CONF/=
2. Documentation for the confidence tests, PTD/DOC/CONF/=
3. Maintenance tests, PTD/MAINT/=
4. Documentation for the maintenance tests, PTD/DOC/MAINT/=
5. A summary of the confidence and maintenance files, PTD/DOC/INSTRUCTIONS
(Any reference to files containing 'NS' in their titles should be ignored by B6000 series systems customers.)
6. A users guide for the PTD (Peripheral Test Driver), PTD/DOC/USERS/GUIDE, which is augmented for B6000 series systems by note D3123 (B6900 Peripheral Test Driver), Mark 32 release.
7. The symbolic and object files (LISTNOTES, OBJECT/LISTNOTES) of a program for processing the 'DOC' files.

All the files, except LISTNOTES, are generated by the I/O Systems Organization and copied to the PTDTTESTS tape.

Processing the 'DOC' Files

A 'DOC' file (e.g., PTD/DOC/INSTRUCTIONS) may be printed by one of the following methods:

1. Batch Use

```

RUN *OBJECT/LISTNOTES; FILE IN(TITLE=PTD/DOC/INSTRUCTIONS);
VALUE N

```

2. Remote Use

R *LISTNOTES; FILE IN(TITLE=PTD/DOC/INSTRUCTIONS); VALUE N

The default value N=0 will cause the document to be printed, without translation, on an EBCDIC96 printer.

Value N=1 will allow the document to be printed, with translation to upper case, on an EBCDIC72 printer.

Detailed operating instructions for LISTNOTES appear in note D3396, Mark 32 release. The operating instructions differ from those on releases prior to Mark 32. The instructions are also contained at the start of the symbolic file LISTNOTES.

LIBRARY TAPE: PTDTESTS HAS 36 FILES, 25728 SEGMENTS

FILE TITLE	CR	SEGS	CREATED	SECURITY	FILEKIND
*LISTNOTES,	%	360	07/05/80	PUBLIC	ALGOLSYMBOL
*OBJECT/LISTNOTES,	%	126	07/05/80	PUBLIC	ALGOLCODE
PTD/CONF/BD2X5,	/	300	09/04/80	PUBLIC	MDLCODE
PTD/CONF/BD20X,	/	400	09/04/80	PUBLIC	MDLCODE
PTD/CONF/CP,	/	200	09/04/80	PUBLIC	MDLCODE
PTD/CONF/CR,	/	200	09/05/80	PUBLIC	MDLCODE
PTD/CONF/IVR,	/	400	09/04/80	PUBLIC	MDLCODE
PTD/CONF/PE,	/	200	09/04/80	PUBLIC	MDLCODE
PTD/CONF/PS,	/	300	09/11/80	PUBLIC	MDLCODE
PTD/CONF/TP,	/	300	09/04/80	PUBLIC	MDLCODE
*PTD/DOC/CONF/BD,	%	966	06/18/80	PUBLIC	CDATA
*PTD/DOC/CONF/CP,	%	266	05/27/80	PUBLIC	CDATA
*PTD/DOC/CONF/CR,	%	266	05/30/80	PUBLIC	CDATA
*PTD/DOC/CONF/IVR,	%	840	07/23/80	PUBLIC	CDATA
*PTD/DOC/CONF/PE,	%	266	06/18/80	PUBLIC	CDATA
*PTD/DOC/CONF/PS,	%	644	06/18/80	PUBLIC	CDATA
*PTD/DOC/CONF/TP,	%	182	06/18/80	PUBLIC	CDATA
*PTD/DOC/INSTRUCTIONS,	%	182	07/11/80	PUBLIC	CDATA
*PTD/DOC/MAINT/CP,	%	2646	08/20/80	PUBLIC	CDATA
*PTD/DOC/MAINT/CR,	%	2072	08/20/80	PUBLIC	CDATA
*PTD/DOC/MAINT/HT,	%	1456	06/18/80	PUBLIC	CDATA
*PTD/DOC/MAINT/LSP,	%	658	08/06/80	PUBLIC	CDATA
*PTD/DOC/MAINT/NSP,	%	910	09/11/80	PUBLIC	CDATA
*PTD/DOC/MAINT/ODT,	%	700	06/19/80	PUBLIC	CDATA
*PTD/DOC/MAINT/PE,	%	2366	08/29/80	PUBLIC	CDATA
*PTD/DOC/MAINT/TP,	%	1974	08/20/80	PUBLIC	CDATA
*PTD/DOC/USERS/GUIDE,	%	448	08/08/80	PUBLIC	CDATA
PTD/MAINT/CP,	/	600	09/04/80	PUBLIC	MDLCODE
PTD/MAINT/CR,	/	600	09/04/80	PUBLIC	MDLCODE
PTD/MAINT/HT2X5,	/	500	09/04/80	PUBLIC	MDLCODE
PTD/MAINT/HT20X,	/	500	09/04/80	PUBLIC	MDLCODE
PTD/MAINT/LSP,	/	400	09/12/80	PUBLIC	MDLCODE
PTD/MAINT/NSP,	/	2000	09/11/80	PUBLIC	MDLCODE
PTD/MAINT/ODT,	/	300	09/12/80	PUBLIC	MDLCODE
PTD/MAINT/PE,	/	600	09/11/80	PUBLIC	MDLCODE
PTD/MAINT/TP,	/	600	09/04/80	PUBLIC	MDLCODE

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

PERIPHERAL TEST DRIVER (PTD)

P3680 PTDTEST - PERIPHERAL TEST DRIVER

The initial implementation of PTD (Peripheral Test Driver) has been released.

DOCUMENT CHANGES NOTES (D NOTES)

SORTMCP

D3233 SORTMCP - TAPE WORK FILE

SORT now checks to ensure that tape work files are assigned to tape units; if not, Error #76 is generated.

Consequently, add the following to "SORT Error Messages" in the System Software Operational Guide Reference Manual, Vol. 1 (Form No. 5001563), Page 11-A-2:

"76

No tape work file was assigned to tape units."

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

SORTMCP

P3374 SORTMCP - PROTECTED DISK FILES USED AS SORT OUTPUT FILES

SORT will now lock any output disk file with PROTECTION=PROTECTED and SAVEFACTOR=0. Previously, this type of file was closed with RETAIN and would result in extra records when the sort caller did not open the file before exiting the block where the file was declared.

P3624 SORTMCP - LARGE MEMORY SIZE SPECIFIED

When an extremely large (>128K) amount of memory was specified for a SORT, and the size was sufficient to hold more than 32,768 records, the SORT would cause an INVALID OP fault in the COMPARE procedure. This no longer occurs.

484

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

SOURCENDL

D3557 SOURCENDL - BLANK PATCH FIELDS

Due to an operational problem that occurred during release Mark 30 PR1, Mark 31 patches 1 through 13 made it into SOURCENDL with blank patch id fields. The problem was a transient one caused by standardization of copyright notices and version cards, and will not recur.

D3613 SOURCENDL - "ACIII/BDLC BTB" REQUEST

Examples have been added for ACIII/BDLC back-to-back request sets with station, terminal and line definitions for both Two Way Alternate Switched and Two Way Simultaneous direct connect.

DOCUMENT CHANGES NOTES (D NOTES)

SOURCENDLII

D3398 SOURCENDLII - "SOURCENDLII"

The Source NDII program on the Mark 32 release is provided only as an example of how NDII programs may be written. This example may be modified periodically to provide new or improved examples.

D3602 SOURCENDLII - IMPLEMENT "ASCII-APL" FOR "NSP" DATACOM

The supplied source NDII program includes an ALGORITHM and an EDITOR that APL uses for ASCII and ASCII/APL contention terminals. This request set (ALGORITHM and EDITOR) supports most of the features currently supported in APL's NDL request set. It also supports the features required to use CANDE and Logical I/O (object program I/O).

The main features are:

- A station can be in ASCII, APL bit-pairing, or APL type-pairing. The initial setting is in the station's declaration and can be modified during input by the control characters SI(ctrl-O)/SO(ctrl-N) to select ASCII/APL and DC3(ctrl-S)/DC4(ctrl-T) to select bit/type pairing.
- A station can be in half-duplex or echoplex (each character is transmitted back to the station after it is received). The initial setting is in the station's declaration and can be modified during input by the control characters DC1(ctrl-Q)/DC2(ctrl-R) to select half/echo.
- A station can optionally have software translation of lowercase letters into uppercase letters during input. The initial setting is in the station's declaration and can be modified during input by the control characters STX(ctrl-B)/ETX(ctrl-C) to select translation/no translation.
- A station can optionally have paging done by the request set. If selected, the page size is `TERMINAL.PAGESIZE` and the message ".page." appears when a page is full. A carriage return continues output, a `BREAK` interrupts output, any other character is ignored. The option is selected in the station's declaration.
- A station can optionally have tabs be simulated by spaces (pseudotabs). If selected, the tabsize is settable when talking to APL and is fixed at 5 otherwise. The option is selected in the station's declaration.
- A station can optionally have a delay occur after a linefeed, formfeed, and carriage return. If selected, the delay is 33 milliseconds after a linefeed, 500 milliseconds after a formfeed, and $90+2*\text{cursorposition}$ milliseconds after a carriage return. The option is selected in the station's declaration.
- Fully supported functions used by APL are:
 - My turn/your turn enforcement.
 - Break on output and input detection.
 - Optional use of uninterpreted output (selected in the station's declaration).
- The common functions required by CANDE and Logical I/O are supported. The pre-declared ASCII translate table is used when the station is in ASCII. In output, the SUB control character is not printed, and in input the STX, ETX, SO, SI, DC1, DC2, DC3, DC4, LF, FF, BS, and CR control characters are not passed on to the host. Other control characters are not affected.

Some features not supported are:

- CANDE control character substitution. The backspace, line delete, and line end characters cannot be changed from BS, DEL, and CR.
- Control messages to APL and Who-Are-You messages (containing ENQ, ctrl-E).
- White space optimization.
- Software setting of tabs on AJ630 and AJ832 terminals.
- Use of `TERMINAL.WRAPAROUND` to avoid transmitting extra linefeed and carriage return.

The ALGORITHM is called "APL_ASCII_ALG" and the EDITOR is called "APL_ASCII_EDITOR". This ALGORITHM can only use, and is only useable with, this EDITOR.

The default station "APL_ASCII_HARD" is for hardcopy ASCII/APL terminals and "APL_ASCII_SCREEN" is for screen ASCII/APL terminals. The station "APL_AS_A_MCS" exists only to ensure that APL is declared as a mcs.

DOCUMENT CHANGES NOTES (D NOTES)

SYSTEST

D3197 SYSTEST - "SYSTESTS" TAPE REORGANIZATION

On Mark 32, XALGOL programs are disallowed; therefore, the XALGOL programs SYSTEST/LANG/XALGOL and SYSTEST/UTIL/DCPUTILITY have been removed from the SYSTESTS tape. Since SYMTEST/OFF/LMDCP2, SYMTEST/OFF/PROCTRANS, and SYMTEST/OFF/DCPCLUSTER are source programs assembled by SYSTEST/UTIL/DCPUTILITY to create SYSTEST/OFF/LMDCP2, SYMTEST/OFF/PROCTRANS, and SYMTEST/OFF/DCPCLUSTER, these symbolics have been removed. Instructions for doing the compile have been removed from the corresponding documentation files. Only the objects SYSTEST/OFF/LMDCP2, SYMTEST/OFF/PROCTRANS, and SYMTEST/OFF/DCPCLUSTER have been released.

If the files that have been removed are required for any reason, the Mark 31 SYSTESTS tape must be used.

For the Mark 32 release, the SYSTESTS tape has been reorganized into three tapes, as follows:

SYSTESTS: Tests designed for machine-independent systems

SYSTEST/DOCUMENTOR, SYMTEST/DOCUMENTOR.
 SYSTEST/UTIL/DKADDR, SYMTEST/UTIL/DKADDR.
 SYSTEST/UTIL/PKTEST, SYMTEST/UTIL/PKTEST.
 SYSTEST/UTIL/DECODER, SYMTEST/UTIL/DECODER.
 SYSTEST/IO/=. SYMTEST/IO/=.
 SYSTEST/GMM/=. SYMTEST/MATGMM,
 SYMTEST/GMM/PROGRAMDOC.
 SYSTEST/PROC/=. SYMTEST/PROC/=.
 SYSTEST/TAPEHANDLER, SYMTEST/TAPEHANDLER.

SYSTESTSMPX: Tests designed for multiplexor systems

The SCR-related files:

SYSTEST/SCR/=.
 SYSTEST/ESCR/=.
 SYSTEST/UTIL/SCRPATCH, SYMTEST/UTIL/SCRPATCH.
 SYSTEST/UTIL/SCRTAPEALIGN,
 SYMTEST/UTIL/SCRTAPEALIGN.
 SYSTEST/UTIL/SCANCONT, SYMTEST/UTIL/SCANCONT.
 SYSTEST/UTIL/SCANCONTLOG, SYMTEST/UTIL/SCANCONTLOG.

Files which use the MPX I/O error log as data:

SYSTEST/UTIL/DKERRANAL, SYMTEST/UTIL/DKERRANAL.
 SYSTEST/UTIL/LOGMAPPER, SYMTEST/UTIL/LOGMAPPER,
 SYMTEST/UTIL/LOGMAPPERDOC.
 SYSTEST/UTIL/LOGSTRIPPER, SYMTEST/UTIL/LOGSTRIPPER.
 SYSTEST/UTIL/SCANHISTORY, SYMTEST/UTIL/SCANHISTORY.

The DCP-related files:

SYSTEST/DCP/=. SYMTEST/DCP/=.
 DOCFILE/DCP/=.
 SYSTEST/OFF/LMMM, SYMTEST/OFF/DCPLMMMDOC.
 SYSTEST/OFF/MMMM, SYMTEST/OFF/DCPMMMDOC.
 SYSTEST/OFF/SMLM.
 SYSTEST/OFF/SMMM.
 SYSTEST/OFF/LMDCP2, SYMTEST/OFF/LMDCP2DOC.
 SYSTEST/OFF/ALLORD1&2, SYMTEST/OFF/ALLORDLM1-8,
 SYMTEST/OFF/DCPALLORDSDOC.
 SYSTEST/OFF/LMLOADER, SYMTEST/OFF/DCPLMLOADERDOC.
 SYSTEST/OFF/PROCTRANS, SYMTEST/OFF/PROCTRANSDOC.
 SYSTEST/OFF/DCPCLUSTER, SYMTEST/OFF/DCPCLUSTERDOC.
 SYSTEST/OFF/DCPHANDLER.
 SYSTEST/UTIL/DCPHELPER.
 SYSTEST/UTIL/DCPLOADER, SYMTEST/UTIL/DCPLOADER.

Miscellaneous:

SYSTEST/DOCUMENTOR, SYMTEST/DOCUMENTOR.
 SYMTEST/MDL3/BASICSYS1, SYMTEST/MDL3/TESTGEN.
 SYSTEST/TAPEHANDLER, SYMTEST/TAPEHANDLER.
 SYSTEST/UTIL/RESHELPER, SYMTEST/UTIL/RESHELPER.
 SYSTEST/UTIL/ALLPEPCOPY, SYMTEST/UTIL/ALLPEPCOPY.
 SYSTEST/UTIL/MAKEONECARDLDR,
 SYMTEST/UTIL/MAKEONECARDLDR,
 SYMTEST/UTIL/ONECARDLOADERS.
 SYMTEST/DECK/ALLORDPK/CD23200082,
 SYMTEST/DECK/ALLORDERS/CD19146562,
 SYMTEST/OFF/ALLORDLOADERDOC.

SYSTESTSLANG: Tests designed for languages

SYSTESTSLANG/=

SYSTEST/LANG/=. SYMTEST/LANG/=. .

SYSTEST/DOCUMENTOR, SYMTEST/DOCUMENTOR.

01
02
03

DOCUMENT CHANGES NOTES (D NOTES)

UDSTRUCTURE TABLE

D3221 UDSTRCTTAB - "SYSTEMUSER" BIT MOVED

On the Mark 31 release, a new bit was added to the userdatafile for Host Services. This bit, SYSTEMUSER, is used to allow or deny privileges when entering inter-system ODT commands. The location of this bit conflicts with a bit currently in use on B7000 series systems, CHARGERREQ. In order to resolve this conflict, the SYSTEMUSER bit is being moved. Sites that have not set the SYSTEMUSER bit for any of their users should disregard the remainder of this note.

For sites that have set the SYSTEMUSER bit, the following procedure should be followed:

- a) Make sure that the 31 PR1 intrinsics are being used.
- b) Recompile any software that may reference the SYSTEMUSER bit. SYSTEM/HOSTSERVICES is the only standard software that uses it. The 31 PR1 version has already been compiled.
- c) For users that currently have the SYSTEMUSER bit set, reset the CHARGERREQ bit. CHARGERREQ has been added to the B6000 series UDSTRUCTURETABLE to become compatible with the B7000 series.
- d) Set the SYSTEMUSER bit for those users that had previously been given that distinction.

D3388 UDSTRCTTAB - CLASS LOCATOR

Page 9-2-9 in the SOG Reference Manual, Volume 2, (Form No. 5001688), paragraph g, should be replaced by the following:

"g. CLASS - CLASS is used to specify into which queue a job initiated by this user will go by default. A value of 0 indicates the system default queue. Any other value indicates a queue number. Queue 0 cannot be specified. This field allows references to queues numbered 1 through 255."

D3415 UDSTRCTTAB - NON INTERACTIVE "APL"

The APL node is modified to support the non-interactive APL implementation.

ADDED: MPCMAXSESSIONS, APLSESSIONCOUNT, APLMAXINTERACTIVE, APLMAXDETACHED.

DELETED: APLMAXBKRNDWS, APLBKRNDWSCOUNT, APLMAXBKRNDACTIVE, APLBKRNDACTIVECNT, APLMINKRNDCLASS.

D3477 UDSTRCTTAB - APL-DETACHED WORKSPACE

The APL node has been modified to support further APL detached work space implementation, as follows:

Added: APLTOTALUSAGERATE
Deleted: APLMAXSESSIONS
APLSESSIONCOUNT

DOCUMENT CHANGES NOTES (D NOTES)

USERSTRUCTURE COMPILER

D3445 USERSTRUCT - EXAMPLE OF BINDING "USERSTRUCTURE"

The B7000/B6000 System Software Operational Guide, Volume 2 (Form No. 5001688), should be corrected as follows: On page 9-2-12, the example in section (b) should be replaced with the following:

```
"<i>BEGIN JOB COMPILE/UDSTRUCTURTABLE;
<i>COMPILE SEP/UDSTRUCTURETABLE with USERSTRUCTURE;
    COMPILER FILE CARD(TITLE=SYMBOL/UDSTRUCTURETABLE
                        ON DISK);
<i>BIND NEW/INTRINSICS with BINDER=
    BINDER DATA
    HOST IS SYSTEM/INTRINSICS;
    BIND UDSTRUCTURETABLE FROM SEP/=;
<i>END JOB"
```

D3605 USERSTRUCT - NEW "UDSTRUCTURETABLE" GENERATION

The handling of SYMBOL/UDSTRUCTURETABLE by the SYSTEM/USERSTRUCTURE compiler has been changed because of migration from intrinsics to support libraries. Stated simply, USERSTRUCTURE is no longer a compiler that generates a code file which must be bound into SYSTEM/INTRINSICS. USERSTRUCTURE now generates a symbolic patch which must be compiled into SYMBOL/GENERALSUPPORT. Details of the implementation are as follows:

SYMBOL/UDSTRUCTURETABLE

No changes have been made to this symbolic because of support libraries. No procedural changes have been made regarding the patching of this file.

SYSTEM/USERSTRUCTURE

- This program is no longer a compiler and may no longer be invoked via a WFL or CANDE COMPILE statement. USERSTRUCTURE should not be MCed.
- The input file remains CARD; however, the default KIND is now PACK (rather than READER) and the default title is "SYMBOL/UDSTRUCTURETABLE".
- The output file is now PATCH (rather than CODE). The default title for this output file is "PATCH/GENERALSUPPORT/UDSTRUCTURETABLE ON DISK". This title can be changed via file attribute equation to any desired title. USERSTRUCTURE will set its TASKVALUE attribute to one if the patch was correctly generated; the TASKVALUE will be set to zero otherwise.
- The patch file generated by USERSTRUCTURE is properly sequenced and contains the correct \$SET (and POP) VOIDT cards. No further editing is necessary.
- The following WFL job sequence can be used to parse the SYMBOL/UDSTRUCTURETABLE and compile the resulting patch into SYMBOL/GENERALSUPPORT.

```
?RUN SYSTEM/USERSTRUCTURE;    % will read SYMBOL/UDSTRUCTURETABLE
                                % on DISK
                                % will write PATCH/GENERALSUPPORT/
                                % UDSTRUCTURETABLE on DISK

?RUN SYSTEM/PATCH;
    FILE PATCH=PATCHDECK/GENERALSUPPORT;
    DATA
$.LIST P
$#GENERAL
$CLEAR LISTP MERGE LINEINFO
$#GENERAL
$.FILE PATCH/GENERALSUPPORT/UDSTRUCTURETABLE
... other GENERALSUPPORT patches ...
?COMPILE SYSTEM/GENERALSUPPORT WITH NEWP;
    NEWP FILE CARD=PATCHDECK/GENERALSUPPORT;
    NEW FILE TAPE=SYMBOL/GENERALSUPPORT;
```

Existing documentation is affected by this change as follows:

SOG Reference Manual, Volume 2:

Chapter 9 - MAKEUSER:
Page 9-2-12; Modifying UDSTRUCTURETABLE

Chapter 14 - Software Compilation
Page 14-2-5; USERSTRUCTURE
Page 14-2-6 through 14-2-7; Compile and Bind Intrinsics

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

USERSTRUCTURE COMPILER

P3583 USERSTRUCT - USE "LONG" ARRAY MASKSEARCH

USERSTRUCTURE will now use a LONG array to hold the information it is building while syntaxing the UDSTRUCTURETABLE, thus allowing the program to work correctly when this internal array becomes larger than 255 words.

DOCUMENT CHANGES NOTES (D NOTES)

UTILITY LOADER

D3332 UTILOADER - "UTILOADER" ON "MLIP" SYSTEMS

UTILOADER for MLIP systems resides on the BDSUTILITY tape and is loaded by the MDP. The program UTILOADER functions much the same as it did on MPX systems, except that operator input is via the ODT instead of cards. UTILOADER can perform the following functions:

1. Halt/Load the processor from a pack to which CM has previously been done (there is an MCP and a bootstrap pointing to it).
2. Tape load a standalone program from a Library Maintenance tape.
3. Display configuration information.

When the initial message appears on the ODT, UTILOADER expects the operator to enter either "HALTLOAD", "TAPELOAD" or "CONFIGURATION".

If HALTLOAD is entered, UTILOADER prompts the operator with the message:

"ENTER INPUT REQUEST IN THE FOLLOWING FORM:
PK VVV OR PK VVV VIA PORT # LEM PORT # DLP #".

UTILOADER expects the operator to specify the physical unit number of a pack that will be the "HALTLOAD UNIT DESIGNATE" (there are no pins). There are two means by which this unit can be specified: (a) use the first available path, (b) use a specified path. By specifying the "VIA" option, UTILOADER uses the path specified by the operator (port, lem port and DLP). In either case, UTILOADER checks to ensure the pack is ready, then passes control to the resident bootstrap. Since UTILOADER does not verify that there is an MCP and a bootstrap on the pack, the operator must ensure that the pack is one to which CM has previously been done.

If TAPELOAD is entered, UTILOADER prompts the operator with the message:

"ENTER INPUT REQUEST IN THE FOLLOWING FORM:
FILENAME FROM VOLUMENAME ON MT VVV VIA PORT # LEM PORT # DLP #".

The VIA part is optional (as with the HALTLOAD function). UTILOADER reads the volume label on the tape drive specified to verify that it matches, then checks the directory for the file to be loaded. When the file is read into memory, UTILOADER passes control to it.

If CONFIGURATION is entered, UTILOADER displays all the DLPs on a base-by-base basis for all BASES and DLPs on the system in the following form:

BASE 4/1/0 PATH: MLIP PORT 1, LEMPORT 0
 ADDRESS 1 DLPID 048 STANDARD HT (PACK)
 ADDRESS 2 DLPID 032 ODT

BASE 3/1/0 PATH: MLIP PORT 2, LEMPORT 0
 ADDRESS 2 DLPID 001 TRAIN PRINTER

The first line gives pertinent base information:

BASE <baseid> PATH: MLIP PORT <port number> LEMPORT <lemport number>.

Indented lines give DLP information of the form:

 ADDRESS <dlp address> DLPID <dlpid> <dlp type>.

The <baseid> is the field strappable value that is returned from the Base Control Card (see CONFIGURATOR note D3406, "Soft Configuration", for details on base id). The <port number> and <lemport number> identify the host path into the base. The <dlp address> gives the relative address within a base of that DLP. <portnumber>, <lemport number>, and <dlp address> are the input required for the VIA option to the HALTLOAD and TAPELOAD commands. The <dlpid> is the field strappable value that forms the base physical unit number of the peripherals connected to the DLP. The other 15 possible pack units are numbered sequentially, incrementing by one. The <dlp type> specifies the DLP type; e.g. ODT, PRINTER, MAGTAPE, etc. If there are multiple paths to a set of units, UTILOADER will arbitrarily pick the first path (i.e. the path listed first) listed; if a different path is desired, the VIA option must be used.

The CONFIGURATION command expects no input.

DOCUMENT CHANGES NOTES (D NOTES)

WORK FLOW LANGUAGE

D2429 WFL - "NEW" WFL SYNTAX CLARIFICATION

WFL system notes that discuss syntax refer to "new" Mark 29 syntax as described in Mark 29 WFL note D2077. Any other level of syntax will be explicitly described.

D3008 WFL - "INSTRUCTION" STATEMENT

The following examples should be added to the INSTRUCTION statement examples on Page 8-23 of the WFL Reference Manual (Form No. 5001555). The Example section should follow the Semantics section.

"INSTRUCTION 2 MOUNT TAPE TEST3.

```
?BEGIN JOB COMPILE/TESTS;
FAMILY DISK = USERS OTHERWISE DISK;
INSTRUCTION 1 TESTTAPE IS IN TAPE RACK 3.;
COPY&COMPARE = FROM TESTTAPE TO USERS(PACK);
INSTRUCTION 2 IF T17 OR T17A WERE NOT COPIED FROM TESTTAPE TO
USERS, PLEASE DS THIS JOB AND LEAVE JK A NOTE.;
COMPILE TEST/17 ALGOL;
ALGOL FILE CARD(TITLE=T17, KIND=DISK);
FILE F(TITLE=T17A);
IF FILE TEST/17 ISNT RESIDENT THEN ABORT "***BAD COMPILE***";
?END JOB.
```

During execution of the COPY statement in the above job, the system will need tape TESTTAPE. If the operator asks for the most recent instruction, instruction 1 will be displayed, indicating where TESTTAPE can be found. Later, the job will need files T17 and T17A. An instruction request at this point will display instruction 2, with instructions on what to do if T17 and T17A are not present."

The description of the INSTRUCTION statement under "Semantics" on Page 8-23 of the WFL Reference Manual should be replaced with the following:

"The INSTRUCTION statement is used to supply job instructions to operators. Instructions 1 through 63 may be specified.

As a WFL job executes, any INSTRUCTION statements encountered are stored in a table. As each INSTRUCTION statement is found, it is marked as the "most current" instruction until another one is found. At any point during the job's execution, the operator may display any individual instruction by number, via the IB (Instruction Block) ODT message. If the operator does not specify an instruction number, the system will display the "most current" instruction."

D3293 WFL - REMOTE JOB TRANSFER

If a job is intended to be sent to a remote host ("?AT <hostname>" before BEGINJOB), it must not contain BCL or binary data decks. Any occurrence of a BCL or binary deck will cause the following syntax error:

"BCL OR BINARY DECK NOT ALLOWED IN REMOTE JOB TRANSFER"

D3295 WFL - DATA BASE EQUATION ALLOWED

Data base equation statements may now be included in WFL job decks. The form is the same as the format for file equation statements.

Example:

```
DATABASE TESTDB(TITLE=MYDB);
```

D3296 WFL - "BCL" WARNING

WFL will now display the following warning whenever a BCL deck is used:

"BCL CARD DECKS ARE NOT PORTABLE TO EBCDIC MACHINES"

D3351 WFL - FILE EQUATION

Page 6-7 of the WFL Reference Manual (Form No. 5001555), should be changed as follows:

Insert the following paragraph between the second and third paragraphs of the "Semantics for File Equation":

"If the object program to which a <file equation> is applied opens a file whose INTNAME is <intname>, the <file attribute assignment>s which are specified in the <file equation> are merged with the attributes specified in the program. If the same attribute is specified in both the <file equation> and in the program, the <file attribute assignment> which is specified in the <file equation> takes precedence. If a file which is file equated is not opened by the program, the <file equation> has no

effect.

D3361 WFL - SEGMENT CODE FILES

WFL now starts using a new code segment at the beginning of a subroutine declaration or at the beginning of the main body of executable statements if its current code segment has a large amount of code already in it. Consequently, any WFL job which would have extended past the end of a segment can now be broken into subroutines, to execute in multiple segments.

D3512 WFL - OPTIONS ATTRIBUTE WITH "# <STRING PRIMARY>"

The following may be used to specify the <task option list> in the <task attribute assignment>:

```
-- # --<string primary>--|
```

Syntax:

```
---- OPTION ---- = --<task option list>--|
| - OPTIONS - |
```

When this version of the <task option list> is used, the value of <string primary> should not contain parentheses.

Example of Proper Use:

```
STRING ;
S:="LONG,FAULT";
RUN X;
OPTIONS=#S;
.
.
.
```

D3513 WFL - "INSTRUCTION" STATEMENT SYNTAX

The following is the correct syntax specification for the INSTRUCTION statement:

```

                                     |<-----|
-- INSTRUCTION --<integer constant>---/1500\---<string char>----|

```

A default value of <integer constant> is not provided; thus, a syntax error will result if not specified.

D3514 WFL - MNEMONIC FILE ATTRIBUTES

WFL only allows <file mnemonic primary>s to be used with <mnemonic file attribute>s. <arithmetic expression>s are not allowed.

D3518 WFL - STRING RETURNS ABSOLUTE VALUE

On Page 4-10 of the WFL Reference Manual (Form No. 5001555), the following sentence is incorrect:

"The STRING function generates a new string whose value is the decimal representation of the value of the first <integer expression>."

The sentence should read as follows:

"The STRING function generates a new string whose value is the decimal representation of the absolute value of the first integer expression."

D3525 WFL - "HISTORY" SUBFIELDS ARE OF TYPE MNEMONIC

The HISTORY <task attribute> is of type real; its subfields are of type mnemonic (not real, as stated in the WFL Reference Manual, Form No. 5001555). Numeric values may not be used. Furthermore, the proper mnemonic values for HISTORYTYPE and HISTORYCAUSE are the following:

HISTORYTYPE:	HISTORYCAUSE:
NORMALV	OPERATORCAUSEV
DUMPINGV	PROGRAMCAUSEV
QTEDV	RESOURCECAUSEV
STEDV	FAULTCAUSEV
DSEV	SYSTEMCAUSEV
NORMALEOTV	DCERRV
SYNTAXERRORV	IOERRV
UNKNOWNEOTV	SOFTIOERRV

NEWIOERRV
 UNIMPLEMENTEDV
 UNSPECIFIEDCAUSEV
 EBDMSERRV
 NETWORKCAUSEV

D3528 WFL - IMPROVED HANDLING OF "\$INCLUDE" IN HEADINGS

The Mark 31 initial system release of WFL closed a potential security problem by disallowing a \$INCLUDE record from appearing within the heading of a job.

This solution has been overly restrictive; \$INCLUDE may again be used within the heading of a WFL job. The following restrictions apply:

Either

(1) the file being INCLUDED must be "PUBLIC IO" or "PUBLIC IN".

or

(2) the file being INCLUDED must have the same usercode as the disk job symbolic file that contains the \$INCLUDE record.

These two restrictions will still allow jobs entered through a card reader or an ODT to INCLUDE any public file, and will continue to allow STARTed and ZIPed jobs to \$INCLUDE job heading information from other files with the same usercode as that with which the job symbolic itself is stored.

The file security rules for \$INCLUDE within the main body of a job (i.e., following the job heading) remain unchanged. Normal system file security measures are applied as if the WFL compilation were running with the usercode specification that occurred last in the job heading.

D3544 WFL - "COPY/ADD" STATEMENT

The syntax specification for the COPY/ADD statement in the WFL Reference Manual (Form No. 5001555) should be changed to read as follows:

```

----- COPY ----->
| - ADD - | | - & ----- COMPARE - | | - & ----- CATALOG - |
|         | | - AND - | | - AND - | | - BACKUP - |
>--<libmaint file list>----->
|         | | <-----, -----> |
|         | | TO <volume specification> --- |
>----->
| - [ <task id> ] - |

```

The syntax specification for the COPY/ADD statement in the CANDE Reference Manual (Form No. 5011398) should be changed to read as follows:

```

----- COPY ----->
| - ADD - | | - & -- COMPARE - | | - & --- CATALOG - |
|         | | - BACKUP - |
>--<libmaint file list>----->
|         | | <-----, -----> |
|         | | TO <volume specification> --- |
>----->
| - [ <task id> ] - |

```

D3547 WFL - "LOCKED" IN TASK ATTRIBUTE ASSIGNMENT

WFL does not allow the use of the LOCKED <task attribute> in a <task attribute assignment>.

D3580 WFL - PASSING STRINGS VIA "WFL/CANDE"

When WFL or CANDE pass a string to a program as a parameter, they use a word array just large enough to hold the string plus one null character. If the declaration of the parameter in the receiving program is longer than the size of the string passed by WFL, a run-time error may occur.

Such run-time errors do not occur on the Mark 30 release, because WFL at that time copied the string into a 256-character array before passing it to a program. Thus, WFL was erroneously allocating too much stroage for the string. This was an oversight which was corrected on the Mark 31 release; it is now implemented as intended.

D3581 WFL - "COPY/ADD" STATEMENT WITH TAPE VOLUMES

The same tape volume may not be used in more than one FROM clause in the same COPY/ADD statement.

D3584 WFL - "FILEKIND" EXAMPLE

The example on Page 4-11 of the WFL Reference Manual (Form No. 5001555) which reads as follows:

```
IF F(FILEKIND) = #(S&"SYMBOL#) THEN . . .
```

should read as follows:

```
IF F(FILEKIND) IS #(S&"SYMBOL") THEN . . .
```

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

WORK FLOW LANGUAGE

P2747 WFL - CORRECTLY COMPARE FILE AND TASK ATTRIBUTE

Previously, a WFL job would get an INVALID OP when a #<string primary> was used as a file or task mnemonic primary in a file or task mnemonic comparison. WFL now works as documented; it will correctly execute jobs such as the following:

```
?BEGIN JOB FILECOMP;
  STRING S;
  FILE F(KIND=DISK);
  S:="DISK";
  IF F(KIND) IS #S THEN DISPLAY "SAME";
?END JOB.
```

P2748 WFL - JOB ON DISK WITH "NEWSOURCE, SYNTAX"

Previously, a job entered at a card reader with NEWSOURCE and SYNTAX specified would be compiled for syntax and placed in a disk file. Subsequent attempts to execute the job would result in a compile for syntax; the job would not execute.

The WFL compiler now assumes that if a job residing on disk contains a NEWSOURCE specification, it should be executable, so WFL will ignore any syntax specification which may be present.

P2749 WFL - PASS GLOBAL FILES TO PROCESSED SUBROUTINE

Previously, a global disk file passed as a parameter to a processed subroutine would get an INVALID INDEX in the MCP. Global files passed as parameters to a processed subroutine are now handled correctly.

P3263 WFL - TASK PASSED AS BY REFERENCE PARAMETER

WFL was not passing tasks as by-reference parameters correctly. This would usually result in an INVALID OP upon execution. This problem has been corrected.

P3520 WFL - MISSING COMMA IN "ON" STATEMENT

WFL now handles properly a missing comma in the ON statement. Previously, the following statement would not cause a syntax error:

```
ON TASKFAULT
  BEGIN
    <statement list>
  END;
```

As a result of the above statement, the compiler no longer generates code equivalent to the following:

```
ON TASKFAULT;
BEGIN
  <statement list>
END;
```

P3584 WFL - FAMILY SPECIFICATION, "<NAME CONSTANT>"

WFL will now accept any legal <name constant> as a <family name> in a FAMILY specification and as a <usercode> in file titles. Previously, the following statements would cause syntax errors.

```
FAMILY 123A=456B OTHERWISE 567C;
or
COPY (321Z) FILEA FROM A TO B;
```

P3623 WFL - PREVENT "WFL" FAULT

The WFL compiler no longer faults if, while parsing a file equation list, a standalone mnemonic is encountered immediately after an assignment using #<string primary>; e.g., FILE X(TITLE=#S,DISK);.

P3695 WFL - SYNTAX "OLD" WFL DATA BASE, LIBRARY

Pre-2.9 WFL now issues a syntax error when an attempt is made to use DATABASE equation or library equation.

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

XREF ANALYZER

P3147 XREFANALY - ADD "DATABASE" AS "XREF" ITEM

Data bases declared in BDMSALGOL programs will now be properly referenced by the XREFANALYZER and INTERACTIVEXREF.

P3332 XREFANALY - REMOVE "PORT, SIGNAL" VARIABLE TYPES

The ALGOL variable types PORT, PORT ARRAY, SIGNAL and SIGNAL ARRAY are no longer supported.

P3518 XREFANALY - CORRECTLY IDENTIFY "CHARACTER" ARRAYS

The XREFANALYZER will now correctly identify CHARACTER arrays rather than reading them as INTERFACE arrays.

B6000 SERIES MARK 32

DOCUMENT CHANGES NOTES (D NOTES)

DOCUMENTS

D3607 DOCUMENTS - STACK SIZE FOR "ALGOL" COMPILE EXAMPLE

The Software Compilation of the SOG Reference Manual, Volume 2 (Form No. 5001688), Page 14-2-13, should be corrected so that the STACK size of the DCALGOL, ALGOL, and NDLI1 compilers and the BINDER is 2000.

D3625 DOCUMENTS - FIND AT "<GROUP ITEM>"

The following information should be inserted in the DMSII Host Reference Manual (Form No. 5001498), on Page 13-2, following "case d of "IF <data-item-1> is NUMBER, then":

"If <data-item-1> is a GROUP, then:

- a. If the items in a data base GROUP item utilize an odd number of 4-bit digits, the group is padded with a one-digit filler containing the value all bits on. If <data-item-2> is a GROUP item in the program, it must contain a corresponding one-digit filler with the value all bits on."

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

SYSTEST - SCR/7ABMTEST

P3377 7ABMTEST - COUNT "READ" AND "WRITE" ERRORS

Previously, READ and WRITE errors were not included in the total error count; now, they are.

005

B6000 SERIES MARK 32

SOFTWARE IMPROVEMENTS NOTES (P NOTES)

SYSTEST - UTIL/RESHELPER

P3394 RESHELPER - ALLOW RUNS FOR DISK PACK TYPES "206,207"

SYSTEST/UTIL/RESHELPER now runs for disk pack types 206 and 207.

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	PRI	NOTE	DESCRIPTION
ACR	32.0.0024	37073	D3171	INQ CREATE/DELETE
ACR	32.0.0026	37219	P2752	System Serial Number Added
ACR	32.0.0027	37351	P2753	Remove RSFILE Declaration
ACR	32.0.0028*	37210	P2499	Efficient Pack Space Utilizati
ACR	32.0.0030	37343	P2754	Audit File Error Handling
ACR	32.0.0032	37322	D3044	Count Finds Against Index Sets
ACR	32.0.0033	37540	D3084	Simplification of REORGANIZATO
ACR	32.0.0044*	38013	P2755	Flush Buffers for Structure
ACR	32.0.0046	38012	P2756	Do Not Set Inuse Flag
ACR	32.0.0047	38020	P2633	DS of "SECTORS REQUIRED" Vs. R
ACR	32.0.0048	37530	P2632	Audit Restart Information Corr
ACR	32.0.0052	38110	D3045	Statistics Interface
ACR	32.0.0053	38109	D3046	Buffers Moved to Data Base Env
ACR	32.0.0055	38118	P2757	Immediate Overlay of Buffers
ACR	32.0.0057	38128	D3047	Allow AUDIT CLOSE Message
ACR	32.0.0059	38037	D3052	Clear TPS Information
ACR	32.0.0061	38190	P2788	Divest Compact Table Block if
ACR	32.0.0062	38199	P2882	ZERO DISK ADDRESS for Direct D
ACR	32.0.0063	38207	P2883	Quick Fix Creates Empty Audit
ACR	32.0.0064*	38213	P2885	Reuse Empty audit Correctly
ACR	32.0.0065	38214	P2886	RECONSTRUCT Makes Empty Audit
ACR	32.0.0066	38125	P2948	Display Reason for Not Reusing
ACR	32.0.0068*	38120	P2949	Unlock Partition if DS in Open
ACR	32.0.0069	38197	D3108	New Data Base Stack Structure
ACR	32.0.0070	38137	P2964	Data Base Messages
ACR	32.0.0071	38134	D3118	Remove Properties for 27 Links
ACR	32.0.0072*	38761	P2950	INVALID OP With Readahead
ACR	32.0.0073	38011	D3170	Shared ACCESSROUTINES, Data Ba
ACR	32.0.0074*	38752	P3003	Bad Available Tables for Compa
ACR	32.0.0075	39188	P3033	Data Base Subsystem Visible
ACR	32.0.0076	39189	P3030	Nested STARTDB Errors
ACR	32.0.0077	39195	P3031	I/O Timeout
ACR	32.0.0078*	37723	P3071	READLOCKNOPURGE Removed
ACR	32.0.0079	38762	D3270	Implement COPYAUDIT WFL Deck
ACR	32.0.0080	39367	P3106	Interface to Free Stack Record
ACR	32.0.0081	38011	D3170	Shared ACCESSROUTINES, Data Ba
ACR	32.0.0082	39388	P3101	Nested STARTDB
ACR	32.0.0083	39356	D3331	ACCESSROUTINES Error Messages
ACR	32.0.0084*	39620	P3145	Corruption of Compact Data Set
ACR	32.0.0085	39356	D3331	ACCESSROUTINES Error Messages
ACR	32.0.0086	39642	P3175	Control File I/O Lock
ACR	32.0.0087*	39658	P3176	Fault on Reblocked Standard Da
ACR	32.0.0088	39639	D3314	Rebuild Across File Discontinuu
ACR	32.0.0089*	39664	D3306	B7700CODE Option
ACR	32.0.0090*	39668	P3230	Deleting Variable Format Recor
ACR	32.0.0091	39669	P3177	Totalcore Protected by Memlock
ACR	32.0.0093*	39900	P3178	Prevent COPYAUDIT Zip Delay
ACR	32.0.0094	39905	D3315	Forced, Normal Overlays
ACR	32.0.0095	39906	P3179	Graph for Data Base Users
ACR	32.0.0098	39910	D3452	Abort Acceleration
ACR	32.0.0099	39981	D3337	Data Base Stack
ACR	32.0.0100	39356	D3331	ACCESSROUTINES Error Messages
ACR	32.0.0102	39989	D3338	Print Statistics Option
ACR	32.0.0104	40198	P3246	ERROR IN DCB HANDLING
ACR	32.0.0105	40200	P3247	DBSINFO Replaces MYNUMBER
ACR	32.0.0106	40203	P3258	Normal Vs. Direct Files as Par
ACR	32.0.0112*	40221	P3259	Missing Divest on Deadlock
ACR	32.0.0113	40220	P3393	Erroneous BIO/AIO Audit Record
ACR	32.0.0114	40231	D3366	Save and Retrieve Messages
ACR	32.0.0115*	40492	P3284	DCB Handling Error
ACR	32.0.0116	40234	P3312	Bad Links When Open Inquiry
ACR	32.0.0117*	40239	P3313	Links on Select Text Error
ACR	32.0.0118	40506	P3314	Invalid INQUIRY Function Numbe
ACR	32.0.0120*	40516	P3333	Store Restart Area
ACR	32.0.0121	40887	D3460	Preallocation of Direct Data S
ACR	32.0.0122*	40522	P3340	Standard VF Control Word Corru
ACR	32.0.0124	40887	D3460	Preallocation of Direct Data S
ACR	32.0.0125*	40889	P3368	Do Not Point Links at Overflow
ACR	32.0.0126*	40893	P3369	Missing Divest Following Versi
ACR	32.0.0127*	40886	P3370	Forget Subblock for Ordered Da
ACR	32.0.0128*	40902	P3371	Zeroed Out Blocks in Data Base
ACR	32.0.0129	40887	D3460	Preallocation of Direct Data S
ACR	32.0.0130	40887	D3460	Preallocation of Direct Data S
ACR	32.0.0131*	40928	P3383	Partition Open Error
ACR	32.0.0134*	41358	P3486	Invalid Direct Data Set
ACR	32.0.0137*	41387	P3553	Invalid Standard Variable Form
ACR	32.0.0138*	41386	P3554	Partition Audit Records Out of
ACR	32.0.0141*	41782	P3606	Linear Search with Signed Nume
ACR	32.0.0143*	41778	P3618	Cancel or Complete I/O Followi
ACR	32.0.0144	41805	P3696	DMSECURITYERROR
ACR	32.0.0146*	41813	P3773	FORGETSPACE Timing Window
ACR	32.0.0147*	41810	P3683	Ordered Data Set Divest Error

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '**')	PRI	NOTE	DESCRIPTION
ACR	32.0.0148	41394	P3697	Fail to Divest	
ACR	32.0.0149*	41395	P3708	Infinite Loop	
ACR	32.0.0153*	42316	P3755	NOTLOCKED Exception	
ACR	32.0.0155*	42320	P3789	Cannot Locate Compact Record	
ALGOL	32.0.0008	34517	P2724	Strings as Attributes, TR Item	
ALGOL	32.0.0009	34516	D3009	STRING Expressions in WRITE an	
ALGOL	32.0.0014	38188	D3090	Compiler Info Word in Seg Zero	
ALGOL	32.0.0015	37419	D3062	Longer Strings with Implicit C	
ALGOL	32.0.0016	37420	D3063	Events, Event Arrays as Librar	
ALGOL	32.0.0018*	37417	P2903	Internal Arrays Expanded	
ALGOL	32.0.0019	38197	D3108	New Data Base Stack Structure	
ALGOL	32.0.0020	38197	D3108	New Data Base Stack Structure	
ALGOL	32.0.0022	37416	P2916	Code Optimization Corrected	
ALGOL	32.0.0027	37419	D3062	Longer Strings with Implicit C	
ALGOL	32.0.0029	39226	D3211	Give Warning for INTMODE=BCL	
ALGOL	32.0.0030	37312	D3363	Remove SIGNAL and RESPONSE	
ALGOL	32.0.0031	39228	D3266	Allow LONG As Key Word	
ALGOL	32.0.0032	37312	D3363	Remove SIGNAL and RESPONSE	
ALGOL	32.0.0033	38541	P3037	Incorrect Sign for Complex Exp	
ALGOL	32.0.0034	38540	P3038	Invalid Assignments Not Flagge	
ALGOL	32.0.0035	38539	P3039	Complex Expressions	
ALGOL	32.0.0036	38538	P3040	Erroneous Syntax Error with Co	
ALGOL	32.0.0037	39284	P3041	Incorrect Software Control Wor	
ALGOL	32.0.0038	38543	D3269	Binary I/O for Strings	
ALGOL	32.0.0040	39232	P3091	INVALID OP with Long Export Li	
ALGOL	32.0.0041	39233	P3092	INVALID INDEX with Lex Levels	
ALGOL	32.0.0044	39287	P3154	INVALID OP, Indexed String Arr	
ALGOL	32.0.0045	39288	P3155	INVALID OP for String Expressi	
ALGOL	32.0.0046	39234	P3139	Correct XREF Output for Librar	
ALGOL	32.0.0058	39243	D3349	Set \$NOBINDINFO	
ALGOL	32.0.0059	39255	D3350	Flag \$NOBINDINFO	
ALGOL	32.0.0066	39881	D3360	Modifications to Support Portf	
ALGOL	32.0.0067	39881	D3360	Modifications to Support Portf	
ALGOL	32.0.0068	39881	D3360	Modifications to Support Portf	
ALGOL	32.0.0070	39881	D3360	Modifications to Support Portf	
ALGOL	32.0.0080	39247	P3324	Multiple ELSE Clauses in CASE	
ALGOL	32.0.0083	39243	D3349	Set \$NOBINDINFO	
ALGOL	32.0.0084*	40945	P3798	Locking Code File	
ALGOL	32.0.0085*	40947	P3661	Give Error for Spaces within N	
ALGOL	32.0.0086	39228	D3266	Allow LONG As Key Word	
ALGOL	32.0.0087	39228	D3266	Allow LONG As Key Word	
ALGOL	32.0.0088*	40953	P3406	Prevent INVALID INDEX	
ALGOL	32.0.0089*	40961	P3407	Call BLOCKEXIT to Deallocate B	
ALGOL	32.0.0091*	40954	P3408	INVALID INDEX	
ALGOL	32.0.0092*	40955	P3409	ACCEPT "<string variable>")"	
ALGOL	32.0.0093	39228	D3266	Allow LONG As Key Word	
ALGOL	32.0.0094	40551	D3471	Allow Longer Value Arrays	
ALGOL	32.0.0097	41120	P3410	Making Use of Available Space	
ALGOL	32.0.0099*	41123	P3411	Intrinsic as a Name Parameter	
ALGOL	32.0.0100	41124	P3412	Picture as a Formal Parameter	
ALGOL	32.0.0104	41128	P3413	CTPROC, CTDEFINE Vs. Parameter	
ALGOL	32.0.0107	41307	P3414	Complex Times Real Multiplicat	
ALGOL	32.0.0108*	41308	P3415	Clear SCW Information	
ALGOL	32.0.0109*	41309	P3416	String Pool Exceeded with \$INT	
ALGOL	32.0.0110	41238	P3417	Flag BCL Pointers with Offset	
ALGOL	32.0.0111*	40957	P3418	Long Character Arrays	
ALGOL	32.0.0113*	41573	P3463	Calling USERIOERROR for "MYSEL	
ALGOL	32.0.0114	41498	P3526	Prevent Possible Stack Overflo	
ALGOL	32.0.0115	40549	D3530	FUNCTIONNAME, LIBACCESS Attri	
ALGOL	32.0.0117*	41574	P3498	INVALID INDEX after Parameter	
ALGOL	32.0.0118	41240	P3560	BCL Constructs Removed	
ALGOL	32.0.0120*	41577	P3532	Call Resetpoolstringsize	
ALGOL	32.0.0122*	41579	P3533	Corruption of Value Arrays	
ALGOL	32.0.0125*	42053	P3611	Global STRING PROCEDURE and Bi	
ALGOL	32.0.0126	42051	P3625	Invalid I/O List Elements	
ALGOL	32.0.0127	42052	D3626	Resizing EVENT ARRAYS	
ALGOL	32.0.0129	42052	D3626	Resizing EVENT ARRAYS	
ALGOL	32.0.0130	42240	D3633	"REAL (<pointer expression>)"	
ALGOL	32.0.0131	42235	D3610	Passing Files by Reference to	
ALGOL	32.0.0132	42240	D3633	"REAL (<pointer expression>)"	
ALGOL	32.0.0134	42237	P3774	Missing BLOCKEXIT	
ALGOLINTRN	32.0.0002*	38243	P2940	Backup File Searching	
ALGOLINTRN	32.0.0003	39826	P3298	CTOD Terminates Abnormally	
ALGOLINTRN	32.0.0004*	40657	P3325	Correct DSQRT Errors	
ALGOLINTRN	32.0.0005*	40658	P3326	Correct GAMMA, DGAMMA	
ALGOLTABLE	32.0.0003	38197	D3108	New Data Base Stack Structure	
ALGOLTABLE	32.0.0006	37312	D3363	Remove SIGNAL and RESPONSE	
ALGOLTABLE	32.0.0007	38011	D3170	Shared ACCESSROUTINES, Data Ba	
ALGOLTABLE	32.0.0008	39235	P3147	Add DATABASE as XREF Item	
ALGOLTABLE	32.0.0012	39255	D3350	Flag \$NOBINDINFO	
ALGOLTABLE	32.0.0014	39881	D3360	Modifications to Support Portf	
ALGOLTABLE	32.0.0016	39880	D3357	Remove PORT and SIGNAL	

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '**')	PRI	NOTE	DESCRIPTION
ALGOLTABLE	32.0.0018	39247	P3324		Multiple ELSE Clauses in CASE
ALGOLTABLE	32.0.0019	39243	D3349		Set \$NOBINDINFO
ALGOLTABLE	32.0.0020	39228	D3266		Allow LONG As Key Word
ALGOLTABLE	32.0.0024	42052	D3626		Resizing EVENT ARRAYS
ALGOLTABLE	32.0.0025	42240	D3633		"REAL (<pointer expression>)"
ARCHUPDATE	32.0.0004	40528	P3341		Prevent Sort Error #4
ARCHUPDATE	32.0.0005	41293	D3614		Eliminate OPEN INITIALIZE
ATABLEGEN	32.0.0002	38663	D3076		APL File Attribute
ATABLEGEN	32.0.0006	39063	D3252		Semidependent Tasks, VISIBILITIES
ATABLEGEN	32.0.0012	40986	D3425		Delete PORTS, SIGNALS
ATABLEGEN	32.0.0015	40873	D3482		New Attributes Implemented
ATABLEGEN	32.0.0017	42060	P3626		"ROWSIZE=504" for New Patch
BACKUP	32.0.0011	40403	P3327		BFILE Label Equation
BACKUP	32.0.0012	41062	D3587		HOSTNAME Modifier
BACKUP	32.0.0014*	42113	P3658		"FILE.TITLE" Attribute
BARS	32.0.0001	38584	P2909		Processor Times Reported on Mo
BARS	32.0.0002	38583	P2910		Correct Swapcore Graphs
BARS	32.0.0004	38181	D2978		SYSTEM/BARS Utility
BARS	32.0.0005	38498	D3277		Virtual Memory Utilization Mea
BARS	32.0.0006	38499	P3110		Incomplete Display on System O
BARS	32.0.0007	40106	P3251		Clear Channel Indicators Prope
BARS	32.0.0008	41467	P3464		SCREEN File Attribute
BARS	32.0.0009	41823	P3481		Changes to Type 4 SYSTEMSTATUS
BARS	32.0.0010*	42094	P3642		"<more>" Displayed Completely
BARS	32.0.0011*	42146	P3643		SPO Mode Displays Long Message
BARS	32.0.0012*	42194	P3644		Negative IDLETIME
BARS	32.0.0013*	42195	P3645		Single "." as Input
BASIC	32.0.0002	38188	D3090		Compiler Info Word in Seg Zero
BASIC	32.0.0005*	42041	P3612		Error for DEF Function
BASIC	32.0.0006*	42040	P3622		Flag Question Mark as Invalid
BASIC	32.0.0007*	42046	D3603		Program of Up to 2048 Statemen
BASIC	32.0.0008*	42042	P3664		Error on First Program Token
BDMSALGOL	32.0.0042	38011	D3170		Shared ACCESSROUTINES, Data Ba
BDMSALGOL	32.0.0043	38011	D3170		Shared ACCESSROUTINES, Data Ba
BDMSALGOL	32.0.0047	39235	P3147		Add DATABASE as XREF Item
BDMSALGOL	32.0.0052	39240	D3324		Print Data Base Title
BDMSALGOL	32.0.0072*	40257	P3296		Transaction Record Parameters
BDMSALGOL	32.0.0082	40882	D3440		Compiler Identification
BDMSALGOL	32.0.0123	41580	D3552		Deimplement OPEN INITIALIZE
BDMSCOBOL	32.0.0024*	35937	P2730		BDMSCOBOL Generates Bad Print
BDMSCOBOL	32.0.0032*	38706	P2731		INVALID INDEX
BDMSCOBOL	32.0.0036	38011	D3170		Shared ACCESSROUTINES, Data Ba
BDMSCOBOL	32.0.0037	38011	D3170		Shared ACCESSROUTINES, Data Ba
BDMSCOBOL	32.0.0044	39444	P3085		Transaction Item, 1 or 2 Chara
BDMSCOBOL	32.0.0048	39479	D3325		Data Base Equation Information
BDMSCOBOL	32.0.0075	41756	D3555		OPEN INITIALIZE Deimplemented
BDMSCOBOL	32.0.0087*	42010	P3588		Invalid Header
BDMSCOBOL	32.0.0088*	42012	P3589		INVALID INDEX
BDMSCOBOL	32.0.0089*	42013	P3590		INVALID INDEX
BDMSCOBOL	32.0.0090*	42015	P3591		"DB-INVOKE" Hardly Readable
BDMSCOBOL	32.0.0092*	42016	P3635		INVALID OP Accessing Global Da
BDMSCOBOL	32.0.0093	42033	P3636		Linear Search Selection Expres
BDMSCOBOL	32.0.0100	42026	P3803		"DUMP PRINTER (<dataset-name>)"
BDMSCOBOL74	32.0.0106	41756	D3555		OPEN INITIALIZE Deimplemented
BDMSCOBOL74	32.0.0111*	42010	P3588		Invalid Header
BDMSCOBOL74	32.0.0112*	42012	P3589		INVALID INDEX
BDMSCOBOL74	32.0.0113*	42013	P3590		INVALID INDEX
BDMSCOBOL74	32.0.0114*	42015	P3591		"DB-INVOKE" Hardly Readable
BDMSCOBOL74	32.0.0115*	42016	P3635		INVALID OP Accessing Global Da
BDMSCOBOL74	32.0.0116	42033	P3636		Linear Search Selection Expres
BDMSPLI	32.0.0011	38449	P2843		Transaction Items
BDMSPLI	32.0.0012*	37882	P2844		Moving Data Base Items
BDMSPLI	32.0.0013*	37883	P2845		XREF Option with BDMS
BDMSPLI	32.0.0014*	37886	P2846		Incorrect BDMS OPEN Statement
BDMSPLI	32.0.0015*	37893	P2847		Multidimensional DMS Arrays
BDMSPLI	32.0.0018*	37894	P2848		Builtin Functions and BDMS
BDMSPLI	32.0.0019*	37888	P2842		PL/I Compiler Looping
BDMSPLI	32.0.0028	38011	D3170		Shared ACCESSROUTINES, Data Ba
BDMSPLI	32.0.0042*	39766	P3209		CREATE Statement
BDMSPLI	32.0.0043*	39949	P3210		DATADICTINFO
BDMSPLI	32.0.0062	41641	D3556		OPEN INITIALIZE Deimplemented
BDMSPLI	32.0.0070*	41332	P3595		Data Base BINDINFO
BDMSPLI	32.0.0071*	41622	P3594		PUT EDIT of Data Base Items
BINDER	32.0.0003	38241	D3414		Delete Old Intrinsics
BINDER	32.0.0005	39145	D3158		Match NEWP Codefile level to B
BINDER	32.0.0006*	39280	P3010		Multiple Rebinds of SYSTEM/INT
BINDER	32.0.0013	40557	P3328		Binding FORTRAN Routines
BINDER	32.0.0014	40558	P3360		Correct \$WAIT with <mix no.> O
BINDER	32.0.0015*	40995	P3419		Binding with "D[0]" Intrinsics
BINDER	32.0.0016	40135	P3438		BINDINFO for Alternatives
BINDER	32.0.0017	41229	D3487		MCP Code File Row Size = 504
BINDER	32.0.0019	41230	D3495		Installation Intrinsic Warning

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	PRI	NOTE	DESCRIPTION
BINDER	32.0.0023*	41510	P3587	Binding Programs with \$DATADIC
BNA	32.0.0007	35594	D3012	Analyze IOCB
BNA	32.0.0010	35594	D3012	Analyze IOCB
BNA	32.0.0011	40663	D3356	New and Old ODT Messages
BNA	32.0.0021	38069	D3102	BNA MCS
BNA	32.0.0027	40663	D3356	New and Old ODT Messages
BNA	32.0.0048	40663	D3356	New and Old ODT Messages
BNA	32.0.0051	40663	D3356	New and Old ODT Messages
BNA	32.0.0069	38790	D3281	Attribute Handling
BNA	32.0.0308	41710	D3550	Eliminate DL Network
BUILDINQ	32.0.0003	37073	D3171	INQ CREATE/DELETE
BUILDINQ	32.0.0007*	37723	P3071	READLOCKNOPURGE Removed
BUILDINQ	32.0.0008*	39215	P3094	Renamed RESTART Data Set
BUILDINQ	32.0.0009	39981	D3337	Data Base Stack
BUILDINQ	32.0.0012	41296	P3706	Global Data in Logical Data Ba
BUILDINQ	32.0.0013	41297	D3615	Segmented Value Arrays
BUILDINQ	32.0.0014	41298	P3698	Multiple Blocks in DMINQDIRECT
BUILDREORG	32.0.0002	37320	D3120	DASDL/REORGANIZATION Enhanceme
BUILDREORG	32.0.0003	37540	D3084	Simplification of REORGANIZATO
BUILDREORG	32.0.0005	38009	D3082	Implicit GENERATE Statements
BUILDREORG	32.0.0006	38007	D3083	New Default for <sort options>
BUILDREORG	32.0.0007	38011	D3170	Shared ACCESSROUTINES, Data Ba
BUILDREORG	32.0.0008	39981	D3337	Data Base Stack
CANDE	32.0.0007	37835	P2798	SCHEDULE Restart Problems
CANDE	32.0.0009*	37831	P2833	Missing SCHEDULE Output File
CANDE	32.0.0010	38597	D3104	XALGOL Deimplemented
CANDE	32.0.0011	39111	D3249	VISIBILITY Task, SCATTER Run-T
CANDE	32.0.0012*	40013	P3200	Schedule Request on Unnamed Wo
CANDE	32.0.0013	40366	D3364	Automatic DESTNAME for CANDE S
CANDE	32.0.0015	40415	P3329	Missing SCHOUT File
CANDE	32.0.0016	41179	P3420	Secure Schedule Files Properly
CANDE	32.0.0017	41180	P3421	Allow Setting Chargecode to Nu
CANDE	32.0.0019*	41182	P3422	TAPE Command in DO Files
CANDE	32.0.0020	41183	P3423	Recognize NDL Sequence Mode Te
CANDE	32.0.0021	41184	P3424	Security Problem
CANDE	32.0.0022	41185	P3425	CANDE Creates Bad Tankfile
CANDE	32.0.0023	41186	P3426	Schedule Sessions Vs. Chargeco
CANDE	32.0.0024*	40832	P3460	CANDE Errors Now Attributed to
CANDE	32.0.0025*	40833	P3499	Allow More than 2 Digits
CANDE	32.0.0026*	41487	P3500	CANDE DS for Security Violatio
CANDE	32.0.0027*	41685	P3501	Packname with Leading Digit
CANDE	32.0.0028*	41731	P3535	Backupprocessor Finding End of
CANDE	32.0.0029*	41733	P3536	EOL Character in DO Statement
CANDE	32.0.0031	41734	P3574	SEG ARRAY Error in DCWER
CANDE	32.0.0032	41966	P3571	Folding Lower Case Tokens
CANDE	32.0.0033*	41732	P3613	Station Vs. Terminal Settings
CANDE	32.0.0034	40663	D3356	New and Old ODT Messages
CANDE	32.0.0035	42117	P3691	Handling Line Errors
CANDE	32.0.0036	42418	P3692	END JOB on WFL Statement
CANDE	32.0.0038	42457	P3785	Allow "4-Character" Verbs
CANDE	32.0.0039	42566	D3642	CANDE Vs. Foreign Tasks
CANDE	32.0.0040	41913	D3646	Compiler Type FORTRAN77
CCTABLEGEN	32.0.0001	38055	D3123	B6900 Peripheral Test Driver
COBOL	32.0.0010*	37811	P2701	INVALID INDEX in COBOL Compile
COBOL	32.0.0011*	37810	P2702	USE Routine Not Invoked
COBOL	32.0.0012*	35960	P2703	VALUE(TERMINATED)
COBOL	32.0.0014*	37808	P2704	WAIT Statement
COBOL	32.0.0015*	37807	P2705	IF Statement Generates Bad Cod
COBOL	32.0.0016*	35961	P2706	Results of Exponentiation Impr
COBOL	32.0.0017*	35952	P2707	Bindinfo for 77 COMP Global It
COBOL	32.0.0018*	38151	P2708	Invalid Syntax for File Attrib
COBOL	32.0.0019*	38152	D3003	Binding and Statistics
COBOL	32.0.0020	38167	P2727	Compilation Summary
COBOL	32.0.0021*	38153	P2728	INVALID INDEX
COBOL	32.0.0022*	38154	P2729	NEWSEQERR \$ Option
COBOL	32.0.0023	38188	D3090	Compiler Info Word in Seg Zero
COBOL	32.0.0025	38156	P2813	"OPEN O-I FILE1 I-O FILE2"
COBOL	32.0.0027	38158	P2815	INVALID OP in IF Statement
COBOL	32.0.0028*	38176	P2834	Interaction of OPEN Statement
COBOL	32.0.0029	38160	P2835	STOP RUN
COBOL	32.0.0031	38166	P2932	\$ Options SEQERR, NEWSEQERR, S
COBOL	32.0.0033	38178	P2732	Intrinsic Information in Globa
COBOL	32.0.0038	39258	D3223	BCL Warnings
COBOL	32.0.0040*	39264	P3054	Call User Intrinsic
COBOL	32.0.0041	39265	P3055	INVALID INDEX in Report Writer
COBOL	32.0.0042*	39429	P3080	Indexed File with Invalid Key
COBOL	32.0.0043	39439	P3081	RERUN Clause
COBOL	32.0.0052*	40266	P3205	Calls on Untyped User Intrinsi
COBOL	32.0.0053*	40267	P3275	Group Computational Moves
COBOL	32.0.0055*	40269	P3274	Equal Comparisons
COBOL	32.0.0056*	40270	P3276	Erroneous Syntax Error
COBOL	32.0.0057*	40271	P3262	LIBRARY CALL Within IF

B6000 SERIES MARK 32

SOFTWARE	PATCH	PRI	NOTE	DESCRIPTION
COBOL	32.0.0058*	40272	P3295	Syntax Error in IF Statement
COBOL	32.0.0059	40540	D3359	Modifications to Support Port
COBOL	32.0.0060*	40273	P3294	Syntax Checking with NEXT SENT
COBOL	32.0.0061	40274	P3302	OBJECT-COMPUTER Clause Syntax
COBOL	32.0.0062	40275	P3303	ANSI 74 Default Line Spacing
COBOL	32.0.0063	40276	P3304	LINE NUMBER Clause Syntax
COBOL	32.0.0064	40277	P3305	MERGE Statement Syntax
COBOL	32.0.0065*	40279	P3306	Long Conditional Expressions
COBOL	32.0.0066*	40280	P3307	KEYSPERENTRY Greater Than 63
COBOL	32.0.0067*	40587	P3427	INVALID INDEX
COBOL	32.0.0072	41531	P3465	Maximum Number of Libraries Ex
COBOL	32.0.0073*	41532	P3466	Reserved Words Syntaxed in WRI
COBOL	32.0.0074*	41533	P3467	INVALID INDEX with CP CALL
COBOL	32.0.0077*	41538	P3502	Misalignment of "COMP-2" Sync
COBOL	32.0.0078*	41539	P3503	SEGMENT Clause, 01 Record
COBOL	32.0.0079	41758	P3525	Timestamp Differences
COBOL	32.0.0080	41542	P3581	"COMP-2" Numeric Test
COBOL	32.0.0081*	41544	P3561	ISAM External File Names
COBOL	32.0.0082	41545	P3562	COPY with Bad File Title
COBOL	32.0.0083*	41759	P3538	ISAM CLOSE Options
COBOL	32.0.0084*	41546	P3559	MONITOR, Write to Same File
COBOL	32.0.0085	41547	P3563	Operand Left on Top of Stack
COBOL	32.0.0094	40538	P3679	User Intrinsics at Levels "> 2
COBOL74	32.0.0032*	37811	P2701	INVALID INDEX in COBOL Compile
COBOL74	32.0.0033*	35960	P2703	VALUE(TERMINATED)
COBOL74	32.0.0034*	37807	P2705	IF Statement Generates Bad Cod
COBOL74	32.0.0036*	35961	P2706	Results of Exponentiation Impr
COBOL74	32.0.0037	35964	D3010	File Description Entry
COBOL74	32.0.0038	38167	P2727	Compilation Summary
COBOL74	32.0.0039*	38153	P2728	INVALID INDEX
COBOL74	32.0.0040	35996	P3070	SEQCHECK \$ Option
COBOL74	32.0.0041	38188	D3090	Compiler Info Word in Seg Zero
COBOL74	32.0.0046	38158	P2815	INVALID OP in IF Statement
COBOL74	32.0.0049	38171	P2800	SERIALNO Attribute
COBOL74	32.0.0052	38172	P2819	Close for Multi-File Tapes
COBOL74	32.0.0053	38175	P2801	Non-numeric Attributes
COBOL74	32.0.0056	38160	P2835	STOP RUN
COBOL74	32.0.0057	38162	P2851	"\$OPTIMIZE" Compiler Error
COBOL74	32.0.0061	38177	P2931	SIGN Clause for Computational
COBOL74	32.0.0062	38166	P2932	\$ Options SEQERR, NEWSEQERR, S
COBOL74	32.0.0063	38702	P3395	BDMS Federal level Warning
COBOL74	32.0.0064	38703	P2937	File Attributes
COBOL74	32.0.0076	39152	D3224	BCL Warnings
COBOL74	32.0.0080	39265	P3055	INVALID INDEX in Report Writer
COBOL74	32.0.0081	39428	P3060	WRITE AFTER ADVANCING PAGE Sta
COBOL74	32.0.0082	39439	P3081	RERUN Clause
COBOL74	32.0.0083	39441	P3084	Debug Line Values
COBOL74	32.0.0084	39441	P3084	Debug Line Values
COBOL74	32.0.0086	39441	P3084	Debug Line Values
COBOL74	32.0.0087	39441	P3084	Debug Line Values
COBOL74	32.0.0089	39430	P3086	WRITE Statement with FOOTING E
COBOL74	32.0.0099*	40269	P3274	Equal Comparisons
COBOL74	32.0.0100	40337	P3308	Linage - Footing Value of One
COBOL74	32.0.0101*	40272	P3295	Syntax Error in IF Statement
COBOL74	32.0.0102	40541	D3358	Modifications to Support Port
COBOL74	32.0.0103*	40273	P3294	Syntax Checking with NEXT SENT
COBOL74	32.0.0104	40276	P3304	LINE NUMBER Clause Syntax
COBOL74	32.0.0105	41531	P3465	Maximum Number of Libraries Ex
COBOL74	32.0.0108	41758	P3525	Timestamp Differences
COBOL74	32.0.0109	41543	P3564	COMPUTATIONAL Numeric Test
COBOL74	32.0.0117	42045	D3597	File Handling Differences
COBOL74	32.0.0124	42025	P3826	Library Pseudo Text Replacemen
COMPARE	32.0.0002*	41464	P3614	ARRAY TOO LARGE Error
CONFIGURATOR	32.0.0001	39337	D3406	Soft Configuration
CONTROLLER	32.0.0001	37702	D3054	SWAPPER Enhancements
CONTROLLER	32.0.0003	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0005	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0006	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0007	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0008	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0010	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0011	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0012	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0014	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0015	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0016*	39289	P3045	Completed Entries Shows Spurio
CONTROLLER	32.0.0017*	39038	P3046	TERM USER Vs. AT HOSTNAME
CONTROLLER	32.0.0018	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0020	38795	D3251	GETSTATUS/SETSTATUS Enhancemen
CONTROLLER	32.0.0021	39727	P3099	"SUBSYSTEM=." Vs. QF Correctio
CONTROLLER	32.0.0025	40663	D3356	New and Old ODT Messages
CONTROLLER	32.0.0027	40663	D3356	New and Old ODT Messages

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FIRS MARKED WITH '*')	PRI	NOTE	DESCRIPTION
CONTROLLER	32.0.0029	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0031	39526	P3236		Very Long File Names
CONTROLLER	32.0.0032	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0034	38795	D3251		GETSTATUS/SETSTATUS Enhancemen
CONTROLLER	32.0.0035	40102	P3254		BACKUPQUEUER Call
CONTROLLER	32.0.0036	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0039	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0042	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0043	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0044	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0049	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0054	40057	P3352		Multipage PER Display
CONTROLLER	32.0.0057	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0061	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0064*	41210	P3428		Missing "#" on ODT for Swaptas
CONTROLLER	32.0.0065	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0068*	41462	P3516		Hour Field in TIMEAT
CONTROLLER	32.0.0069*	41463	P3517		NS Correction
CONTROLLER	32.0.0070	41710	D3550		Eliminate DL Network
CONTROLLER	32.0.0072	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0074*	41941	D3565		J, MX Response
CONTROLLER	32.0.0080*	42142	P3659		AA Correction
CONTROLLER	32.0.0081	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0085*	41689	D3529		PARTNER , EXCEPTIONTASK Remov
CONTROLLER	32.0.0087*	42182	P3750		PA Correction
CONTROLLER	32.0.0090	40663	D3356		New and Old ODT Messages
CONTROLLER	32.0.0096	42848	D3647		TD Acceleration
CONTROLLER	32.0.0097	42849	D3651		Suppress Frozen Libraries
COPYAUD-II	32.0.0002*	38760	P2956		Print Tape Labels
COPYAUD-II	32.0.0003	39981	D3337		Data Base Stack
COPYAUD-II	32.0.0005*	40900	P3384		Alphanumeric Usercodes
DASDL	32.0.0013	38699	D3099		Independence of DASDL UPDATE C
DASDL	32.0.0018	37320	D3120		DASDL/REORGANIZATION Enhanceme
DASDL	32.0.0020	38011	D3170		Shared ACCESSROUTINES, Data Ba
DASDL	32.0.0021	38194	D3048		Allow Link to Embedded Ds
DASDL	32.0.0022	38211	P2951		Allow Modulus Specification fo
DASDL	32.0.0023	38210	D3113		Delete READAHEADB
DASDL	32.0.0027	38215	D3115		COBOL Reserved Word Table Upda
DASDL	32.0.0028	38122	D3116		Put Subsystem ID in Text
DASDL	32.0.0029	38011	D3170		Shared ACCESSROUTINES, Data Ba
DASDL	32.0.0030	38133	P2952		Sequence Number not Updated
DASDL	32.0.0031*	38135	P2953		SERIALBUFFERS Attribute
DASDL	32.0.0032	38134	D3118		Remove Properties for 27 Links
DASDL	32.0.0033	38764	D3117		Allow PROPERTIES Label Equatio
DASDL	32.0.0034*	38763	P2967		AREASZ Greater Than 65536 Tru
DASDL	32.0.0035	38759	D3162		DASDL Defaults
DASDL	32.0.0036*	37723	P3071		READLOCKNOPURGE Removed
DASDL	32.0.0037*	39209	P3072		Keychanged Text for Field Item
DASDL	32.0.0038	38762	D3270		Implement COPYAUDIT WFL Deck
DASDL	32.0.0039	38011	D3170		Shared ACCESSROUTINES, Data Ba
DASDL	32.0.0042	39363	D3272		Restructure Description File P
DASDL	32.0.0043*	39379	P3102		Set Up FILEKINDF, Packname Cor
DASDL	32.0.0044	39380	P3124		Superfluous Too Many Areas Mes
DASDL	32.0.0045	39651	D3316		Crunch NEWTAPE File
DASDL	32.0.0046*	39657	P3231		Calculate Reasonable Default R
DASDL	32.0.0048	39675	D3441		Better CONTROLPOINT, SYNCPOINT
DASDL	32.0.0050	39676	P3181		Bad Expand Text for Stored Ite
DASDL	32.0.0051	39981	D3337		Data Base Stack
DASDL	32.0.0052	39983	P3248		Correct Handling of B7700 Doll
DASDL	32.0.0053*	39621	P3241		Loop After Misspelled Update C
DASDL	32.0.0055	40499	P3315		Possible Buffer Overlay
DASDL	32.0.0056*	40511	P3342		BAD SELECT/VERIFY TEXT FOR FIE
DASDL	32.0.0057*	40911	P3385		Bad Expandtext
DASDL	32.0.0058*	40916	P3386		MOVES LIST EXCEEDED Error
DASDL	32.0.0061*	41354	P3442		Erroneous Initial Values for R
DASDL	32.0.0062*	41107	P3487		EOF NO LABEL Error
DASDL	32.0.0063	41773	P3700		Identifiers Ending with a Hyph
DASDL	32.0.0064*	42314	P3758		Creation of Data Base under "
DASDL	32.0.0065*	42322	P3756		BLOCKSIZE TOO SMALL Fault
DASDL	32.0.0067*	42698	P3790		Limit Error on Restart Data Se
DASDL	32.0.0068	42697	P3793		Average Record Size during Upd
DATACOM	32.0.0167*	37852	P2782		DCRECON Line Result Not Return
DATACOM	32.0.0169*	37853	P2783		Extended Line TALLYs Problems
DATACOM	32.0.0170*	37854	P2784		DCSYSTEMTABLES Returns Wrong R
DATACOM	32.0.0180	37856	P2795		READNIF Test for Valid Record
DATACOM	32.0.0226	38512	P2857		Unlock Line by DCIOFINISH Call
DATACOM	32.0.0322	38774	P2960		RECALLOBJECTOUTPUT on Uninitial
DATACOM	32.0.0676	38795	D3251		GETSTATUS/SETSTATUS Enhancemen
DATACOM	32.0.0690*	40082	P3229		DCSYSTEMTABLES Option 5 SEG A
DATACOM	32.0.0695	38795	D3251		GETSTATUS/SETSTATUS Enhancemen
DATACOM	32.0.0848	38795	D3251		GETSTATUS/SETSTATUS Enhancemen
DATACOM	32.0.1161*	42305	P3771		DCFILELOCK Deadlock

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '*')	PRI	NOTE	DESCRIPTION
DCALGOL	32.0.0073		39880	D3357	Remove PORT and SIGNAL
DCAUDITOR	32.0.0006		43207	D3637	Add Additional Information to
DCPPROGEN	32.0.0003		37761	D3028	Message-Oriented Datacom
DCPPROGEN	32.0.0004		37761	D3028	Message-Oriented Datacom
DCPPROGEN	32.0.0005		41149	D3204	Changes for DCP Character Orie
DCPPROGEN	32.0.0006		37865	P2802	Allow TERMINATE NORMAL
DCPPROGEN	32.0.0007		37761	D3028	Message-Oriented Datacom
DCPPROGEN	32.0.0026*		41738	P3497	Garbled BAUDOT Translate Table
DCPPROGEN	32.0.0029*		42038	P3666	Consecutive Line Tally Usage
DCPPROGEN	32.0.0030*		42006	P3677	DLS Corrupted in Full Duplex D
DCPPROGEN	32.0.0031		42374	P3775	INCREMENT TRAN Statement
DCSTATUS	32.0.0002		38656	P3012	Print Station Table Base Corre
DCSTATUS	32.0.0003		38100	D3234	Terminal Transfer
DCSTATUS	32.0.0004*		40451	P3539	"17-Character" Names for GRAPH
DCSTATUS	32.0.0005*		41881	P3572	Prevent NIF/DCPCODE File Chang
DCSTATUS	32.0.0006*		42037	P3600	Calculate Station Table Base S
DDDASDL	32.0.0005		39653	D3327	Add Version Level
DDINITIAL	32.0.0001		39981	D3337	Data Base Stack
DDUPDATE	32.0.0013		38136	D3149	Reformat Datadictionary Report
DDUPDATE	32.0.0015		39981	D3337	Data Base Stack
DIAGNOSTMCS	32.0.0002		39159	P3163	Error in BTB Attach by LSN/DLS
DIAGNOSTMCS	32.0.0003		39937	P3253	Error in "BTB ALL REPEAT <stri
DIAGNOSTMCS	32.0.0004		40454	D3427	Nonnumeric Dial Characters
DIAGNOSTMCS	32.0.0005*		41882	P3601	Alter Correct Station Address
DIAGNOSTMCS	32.0.0006*		41884	P3602	Prevent Dump on "<DCL>"
DIAGNOSTMCS	32.0.0007*		41886	P3603	Attached CA Option
DMALGOL	32.0.0028		39225	P3009	Correct 'ERROR Construct
DMALGOL	32.0.0039*		37723	P3071	READLOCKNOPURGE Removed
DMALGOL	32.0.0079*		39248	P3323	Eliminate Extraneous "?"s for
DMCTL	32.0.0005		38698	D3100	Update Level Check for RECOVER
DMCTL	32.0.0007		38014	P2758	System Identification
DMCTL	32.0.0008		38137	P2964	Data Base Messages
DMCTL	32.0.0009		38011	D3170	Shared ACCESSROUTINES, Data Ba
DMCTL	32.0.0010		38755	D3119	28 to 29 Conversion Options Re
DMCTL	32.0.0012*		37723	P3071	READLOCKNOPURGE Removed
DMCTL	32.0.0013		38755	D3119	28 to 29 Conversion Options Re
DMCTL	32.0.0015		38011	D3170	Shared ACCESSROUTINES, Data Ba
DMCTL	32.0.0016		38011	D3170	Shared ACCESSROUTINES, Data Ba
DMCTL	32.0.0018		39376	P3108	DMCONTROL Resequenced
DMCTL	32.0.0019		39634	P3182	CFDELETEPART Corrupting Contro
DMCTL	32.0.0021		39642	P3175	Control File I/O Lock
DMCTL	32.0.0022		39639	D3314	Rebuild Across File Discontin
DMCTL	32.0.0024		39901	P3185	Initial Value of Designated Se
DMCTL	32.0.0025		39981	D3337	Data Base Stack
DMCTL	32.0.0026		39989	D3338	Print Statistics Option
DMCTL	32.0.0027		40201	P3249	Mark 32 DMS on Mark 31 MCP
DMCTL	32.0.0028		40203	P3258	Normal Vs. Direct Files as Par
DMCTL	32.0.0031		40231	D3366	Save and Retrieve Messages
DMCTL	32.0.0032		40232	P3269	INVALID INDEX on "OVERRIDE HL"
DMCTL	32.0.0033		40236	P3285	CF Title for OVERRIDE HL
DMCTL	32.0.0036		40509	P3316	Reduce Use of REORGINFONODE
DMCTL	32.0.0037		40513	P3317	Set Up Prefix Arrays
DMCTL	32.0.0038		40523	P3343	Structure Details
DMCTL	32.0.0039		40887	D3460	Preallocation of Direct Data S
DMCTL	32.0.0040		38755	D3119	28 to 29 Conversion Options Re
DMCTL	32.0.0041		40887	D3460	Preallocation of Direct Data S
DUMPALL	32.0.0002		37698	D3011	Upper Case Input String
DUMPALL	32.0.0003*		41468	P3468	"PACK=<packname>" Syntax
DUMPALL	32.0.0004*		41452	P3504	Bad COPY Syntax Now Flagged
DUMPALL	32.0.0005*		41469	P3505	LIST with "<manual input>","<p
DUMPALL	32.0.0006*		41466	P3537	DUMPALL Overrides ON PACKNAME
DUMPANALY	32.0.0004		35594	D3012	Analyze IOCB
DUMPANALY	32.0.0005		35596	D3013	Analyze FIB at Address
DUMPANALY	32.0.0006		35594	D3012	Analyze IOCB
DUMPANALY	32.0.0007		35594	D3012	Analyze IOCB
DUMPANALY	32.0.0008		35594	D3012	Analyze IOCB
DUMPANALY	32.0.0009		35594	D3012	Analyze IOCB
DUMPANALY	32.0.0010		35594	D3012	Analyze IOCB
DUMPANALY	32.0.0012		37196	D3136	Print Interactive Input
DUMPANALY	32.0.0015		37702	D3054	SWAPPER Enhancements
DUMPANALY	32.0.0017		37183	D3137	Dump UNITMAP
DUMPANALY	32.0.0018		37170	D3139	Analyze UNITCONTROL
DUMPANALY	32.0.0019		37168	D3019	REPEAT Syntax in INTERACTIVE M
DUMPANALY	32.0.0020		37167	D3138	Print File Buffers Text
DUMPANALY	32.0.0023		38047	P2779	Bad Printer Skip
DUMPANALY	32.0.0024		38048	P2780	Run DUMPANALYZER in FBCDIC
DUMPANALY	32.0.0026		37821	P2797	Correct ID Initialization
DUMPANALY	32.0.0028		37167	D3138	Print File Buffers Text
DUMPANALY	32.0.0029		38469	D3089	Memory Dump Tape Record Format
DUMPANALY	32.0.0030		38197	D3108	New Data Base Stack Structure
DUMPANALY	32.0.0034		39068	P3025	DUMPANALYZER Recognizes Frozen
DUMPANALY	32.0.0037		39109	P3061	"MD RV <addr> FOR ALL" Correct

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '*')	PRI	NOTE	DESCRIPTION
DUMPANALY	32.0.0038		38790	D3281	Attribute Handling
DUMPANALY	32.0.0041		42792	P3824	Memory Organization, Local MCP
DUMPANALY	32.0.0042		39522	D3292	OLAYINFO Analysis
DUMPANALY	32.0.0044		39906	P3179	Graph for Data Base Users
DUMPANALY	32.0.0051		38100	D3234	Terminal Transfer
DUMPANALY	32.0.0054		40768	D3417	PROC Command Implemented
DUMPANALY	32.0.0055		40409	P3353	Remove Port, SIGNAL Code
DUMPANALY	32.0.0056		40818	D3442	Analyze Library Template
DUMPANALY	32.0.0060		40826	P3361	SAVE Large Dump
DUMPANALY	32.0.0061*		41172	P3396	Heading Date for Disk Input
DUMPANALY	32.0.0062		41021	D3481	New Port Analysis
DUMPANALY	32.0.0063*		41199	P3429	Processor Loop after "?END"
DUMPANALY	32.0.0066		41030	P3430	HOSTINFO Deimplemented
DUMPANALY	32.0.0068		40822	D3479	Module Alternative Selection
DUMPANALY	32.0.0073		41057	D3588	New MODE PIB Option
DUMPANALY	32.0.0074		41717	D3589	SAVE Command
DUMPANALY	32.0.0080		42215	D3620	Hostname in Heading
DUMPANALY	32.0.0085		42183	D3643	Analyzing Area Descriptor
DUMPANALY	32.0.0090		38469	D3089	Memory Dump Tape Record Format
DUMPANALY	32.0.0091		38469	D3089	Memory Dump Tape Record Format
DUMPDIR	32.0.0002		39217	D3353	DUMPDIR Enhancements
DUMPDIR	32.0.0003		39981	D3337	Data Base Stack
DUMPDIR	32.0.0005*		40903	P3372	Description File Title
DUMPDIRLIB	32.0.0007		38011	D3170	Shared ACCESSROUTINES, Data Ba
DUMPDIRLIB	32.0.0008		39217	D3353	DUMPDIR Enhancements
DUMPDIRLIB	32.0.0010		39981	D3337	Data Base Stack
DUMPDIRLIB	32.0.0012		40887	D3460	Preallocation of Direct Data S
DUMPDIRLIB	32.0.0013*		41368	P3488	RETAIN Corrected
DUMPDIRLIB	32.0.0014*		41360	P3489	Error Using WRITE=/LIST=
ESPOL	32.0.0002		39485	D3285	Deimplementation of ESPOL Comp
ESPOLINTRN	32.0.0005*		37407	P2933	Correctly Handle BASIC Stringp
ESPOLINTRN	32.0.0007		41829	D3601	Delete Old Intrinsics
ESPOLINTRN	32.0.0010		38543	D3269	Binary I/O for Strings
ESPOLINTRN	32.0.0011		38542	P3042	Freefield Input with Complex A
ESPOLINTRN	32.0.0012		39825	P3299	ALGOL Pointer I/O
ESPOLINTRN	32.0.0013		39437	P3300	Update B7000 Define
ESPOLINTRN	32.0.0015		40654	P3309	Array Row Free Format Read
FILECOPY	32.0.0001		37554	D3020	TASKFAULT, CLASS, OLDWFL Featu
FILECOPY	32.0.0004		37554	D3020	TASKFAULT, CLASS, OLDWFL Featu
FILECOPY	32.0.0005		37554	D3020	TASKFAULT, CLASS, OLDWFL Featu
FILECOPY	32.0.0006*		39543	P3220	EXCLUDE USERCODE/=
FILECOPY	32.0.0007*		40036	P3201	"*=" File Requests
FILECOPY	32.0.0008		40066	P3431	WFL Deck Sequence Number Limit
FILECOPY	32.0.0009*		41165	P3432	NULLFILE Valid Filekind
FILECOPY	32.0.0010*		41163	P3433	INCLUDE Does Not EXCLUDE Autom
FILECOPY	32.0.0011*		41162	P3434	Loop on Invalid Syntax
FILEDATA	32.0.0002*		39597	P3158	Hang on NO FILE
FILEDATA	32.0.0003*		39596	P3159	Valid Requests Rejected After
FILEDATA	32.0.0004*		39595	P3160	Incorrect Indication of IAD Di
FILEDATA	32.0.0005*		41484	P3506	Report on 5N Disk
FILEDATA	32.0.0006*		41188	P3507	"4-digit" Serial Numbers
FILEDATA	32.0.0007*		41189	P3508	Show Last File in CHECKERBOARD
FILEDATA	32.0.0008*		41694	P3509	FILEORGANIZATION Attribute
FORTTRAN	32.0.0005*		38139	P2709	Compiler Loop on DATA Statemen
FORTTRAN	32.0.0006*		38140	P2710	Core to Core I/O in WRITE Stat
FORTTRAN	32.0.0007*		38141	P2711	Allow Double Precision Express
FORTTRAN	32.0.0008*		38142	P2712	Detect GOTO <non existing stat
FORTTRAN	32.0.0009*		38144	P2734	Strange Action When CHECK, SEQ
FORTTRAN	32.0.0010*		38145	P2735	Format Error on Read from Doub
FORTTRAN	32.0.0011*		38146	P2736	Invalid File Attribute in FILE
FORTTRAN	32.0.0013*		38221	P2836	INVALID INDEX With SYLPT Table
FORTTRAN	32.0.0014*		38224	P2837	AUTOBIND, SEPARATE Set in Main
FORTTRAN	32.0.0015*		38223	P2838	AUTOBIND and GO Despite Syntax
FORTTRAN	32.0.0016*		38501	P2867	\$ Cards Not in NEWTAPE with IN
FORTTRAN	32.0.0017*		38502	P2868	Filesize Estimate Incorrect
FORTTRAN	32.0.0018*		38222	P2869	Parameters are Call By Name
FORTTRAN	32.0.0020		38188	D3090	Compiler Info Word in Seg Zero
FORTTRAN	32.0.0021*		38505	P2904	Blank Card at End of Subroutin
FORTTRAN	32.0.0022		38504	D3093	Warning Message for \$LEVEL
FORTTRAN	32.0.0023*		38554	P2905	Spurious Errors with \$VOIDT, \$
FORTTRAN	32.0.0024*		38552	P2943	\$SEPARATE Without Other Statem
FORTTRAN	32.0.0025*		38551	P2944	Parameter Mismatch
FORTTRAN	32.0.0026		38553	P3656	Unordered Parameters in Librar
FORTTRAN	32.0.0027*		38550	P2945	Arrays Segmented
FORTTRAN	32.0.0028*		39133	P2993	INV PCW
FORTTRAN	32.0.0029*		39133	P2993	INV PCW
FORTTRAN	32.0.0030*		39137	P3007	Too Much Storage Allocated
FORTTRAN	32.0.0031		39136	D3174	Deimplement SIGNAL, RESPONSE C
FORTTRAN	32.0.0032		39138	D3217	BCL Warnings
FORTTRAN	32.0.0033*		39139	P3013	INVALID INDEX Due to Conflicti
FORTTRAN	32.0.0034*		39140	P3014	"W2 COMPILER ERROR" When Using
FORTTRAN	32.0.0035		39141	D3231	Deimplement VECTORMODE Option

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	PRI	NOTE	DESCRIPTION
FORTTRAN	32.0.0036*	39137	P3007	Too Much Storage Allocated
FORTTRAN	32.0.0037	39454	P3156	Allow Family Name in \$INCLUDE
FORTTRAN	32.0.0039	39456	D3362	Modifications to Support Port
FORTTRAN	32.0.0040*	39458	P3663	Using Variable IF in COMMON St
FORTTRAN	32.0.0043*	40981	P3435	"\$INCLUDE <intname>" Corrected
FORTTRAN	32.0.0045*	40985	P3469	Compiler Loop
FORTTRAN	32.0.0046*	41521	P3540	Extra Comma in Parameter List
FORTTRAN	32.0.0047*	41523	P3541	Blank Caused Syntax in FILE St
FORTTRAN	32.0.0048*	41524	P3542	Use Core to Core I/O
FORTTRAN	32.0.0049*	41525	P3543	Invalid Common Allocation
FORTTRAN	32.0.0050*	41519	P3544	Exported Untyped Function
FORTTRAN	32.0.0051*	41901	P3545	Free Format READ with "REAL *8
FORTTRAN	32.0.0052*	41903	P3546	Integer Overflow
FORTTRAN	32.0.0053*	41902	P3573	Double Array as Parameter
FORTTRAN	32.0.0055*	41905	P3592	INVALID INDEX
FORTTRAN	32.0.0056*	41906	P3615	Erroneous Program After Warnin
FORTTRAN	32.0.0057	42550	P3776	DEBUG MONITOR Statement
FORTTRAN	32.0.0058*	42341	D3635	INVALID OP With "> Parameters"
GENERALSUPP	32.0.0006*	40657	P3325	Correct DSQRT Errors
GENERALSUPP	32.0.0007*	40658	P3326	Correct GAMMA, DGAMMA
GENERALSUPP	32.0.0010*	41888	P3534	Exponent Underflow in RTOR
GENERALSUPP	32.0.0012	39826	P3298	CTOD Terminates Abnormally
GENERALSUPP	32.0.0016	38543	D3269	Binary I/O for Strings
GENERALSUPP	32.0.0017	38542	P3042	Freefield Input with Complex A
GENERALSUPP	32.0.0018	39825	P3299	ALGOL Pointer I/O
GENERALSUPP	32.0.0019	41828	P3657	Update B7000 Define
GENERALSUPP	32.0.0020	40654	P3309	Array Row Free Format Read
HELPINQ	32.0.0001	37073	D3171	INQ CREATE/DELETE
HOSTINTFACE	32.0.0003	38113	P2775	Send Entire Restart Record
HOSTINTFACE	32.0.0004	38114	D3085	Return Address of Last User Tr
HOSTINTFACE	32.0.0005	38198	P2881	Eliminate Unused Requestcase V
HOSTINTFACE	32.0.0006	38742	D3230	Elimination of Response From P
HOSTINTFACE	32.0.0007	39981	D3337	Data Base Stack
HOSTINTFACE	32.0.0010	40885	P3350	Eliminate PORT Option
HOSTINTFACE	32.0.0011	41801	P3707	Use Port Files
HOSTLIB	32.0.0005	38114	D3085	Return Address of Last User Tr
HOSTLIB	32.0.0006	38202	P2889	Increase Number of Files in Jo
HOSTLIB	32.0.0007	38747	P2864	Check for Unassigned Transacti
HOSTLIB	32.0.0008	39981	D3337	Data Base Stack
HOSTLIB	32.0.0009	40199	D3367	Simultaneous READ/WRITE Access
HOSTLIB	32.0.0012	40524	D3454	New Statistics
HOSTLIB	32.0.0013	41396	P3749	I/O Complete
HOSTLIB	32.0.0014	42315	P3780	Write Error
HOSTLIB	32.0.0015	42325	P3794	Discontinuity of Block Seriak
IADMAPPER	32.0.0001	40709	D3409	IAD Not Supported on B6900
IN-OUTPUT	32.0.0115	37993	D3055	CENSUS Attribute
IN-OUTPUT	32.0.0143*	38077	P2771	UNITNO Vs. BACKUPTAPE
IN-OUTPUT	32.0.0188	37948	P2811	Datacom File, Family Addition,
IN-OUTPUT	32.0.0381	39257	D3222	Close WITH LOCK
IN-OUTPUT	32.0.0437*	39172	P3067	KIND File Attribute
IN-OUTPUT	32.0.0456	38786	D3254	Direct Datacom I/O for Swapjob
IN-OUTPUT	32.0.0457*	37829	P3074	Correct State Attribute on Rem
IN-OUTPUT	32.0.0535	30505	D2782	KIND=DISK Vs. FAMILYNAME
IN-OUTPUT	32.0.0558*	39566	P3149	UPDATE, Binary I/O Writing to
IN-OUTPUT	32.0.0579	39569	P3169	Update, Binary I/O Read Write
IN-OUTPUT	32.0.0605	39593	P3170	Minimize Header Update
IN-OUTPUT	32.0.0606	39573	P3171	BCL Backup Files
IN-OUTPUT	32.0.0607	39572	P3172	Use of TD830 ODT for SPO Files
IN-OUTPUT	32.0.0632	39576	P3174	Break on Output
IN-OUTPUT	32.0.0660	39570	D3329	MYUSE=IO Vs. UPDATEFILE Attrib
IN-OUTPUT	32.0.0685*	37494	P3226	Protected Files Closed
IN-OUTPUT	32.0.0734	40088	D3347	FULLTRANSLATION Option
IN-OUTPUT	32.0.0737	40091	D3348	FILETYPE=5 Files
IN-OUTPUT	32.0.0790	40291	D3373	DISPOSITION File Attribute
IN-OUTPUT	32.0.0794	40091	D3348	FILETYPE=5 Files
IN-OUTPUT	32.0.0813	40714	D3408	AREAClass Vs. FAMILYINDEX
IN-OUTPUT	32.0.0839	40713	D3407	AREASIZE Vs. NEWFILE
IN-OUTPUT	32.0.0841	40714	D3408	AREAClass Vs. FAMILYINDEX
IN-OUTPUT	32.0.0849	40747	D3410	MTBF Eliminated
IN-OUTPUT	32.0.0862	40747	D3410	MTBF Eliminated
IN-OUTPUT	32.0.0970	40873	D3482	New Attributes Implemented
IN-OUTPUT	32.0.0979	40873	D3482	New Attributes Implemented
IN-OUTPUT	32.0.0980	40873	D3482	New Attributes Implemented
IN-OUTPUT	32.0.0981	40873	D3482	New Attributes Implemented
IN-OUTPUT	32.0.0982	40873	D3482	New Attributes Implemented
IN-OUTPUT	32.0.0983	40873	D3482	New Attributes Implemented
IN-OUTPUT	32.0.1010	41442	D3499	OPEN Input Reverse Tape Files
IN-OUTPUT	32.0.1119	42110	P3633	Error Messages Contain Line Nu
IN-OUTPUT	32.0.1120	40714	D3408	AREAClass Vs. FAMILYINDEX
IN-OUTPUT	32.0.1121	42111	P3634	Random Badly Blocked I/O
IN-OUTPUT	32.0.1241	42601	P3783	I/O Result from SEEK Statement
INQ	32.0.0003	37073	D3171	INQ CREATE/DELETE

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '*')	PRI	NOTE	DESCRIPTION
INQ	32.0.0005	37217	D3264		Implicit Qualification Improve
INQ	32.0.0008	38033	P2759		Optimize FIND VIA <subset> AT
INQ	32.0.0009	38750	P3001		Invalid Page Break on Control
INQ	32.0.0010	38745	P3000		System ID and Patch in Heading
INQ	32.0.0011	39193	P3032		Change to Maximum Display
INQ	32.0.0012	38011	D3170		Shared ACCESSROUTINES, Data Ba
INQ	32.0.0013*	37723	P3071		READLOCKNOPURGE Removed
INQ	32.0.0014	39196	D3273		Setting Items to NULL, Testing
INQ	32.0.0018*	39647	P3161		Truncation of Position 132
INQ	32.0.0019	39644	P3186		Reporting Long Subscripted Alp
INQ	32.0.0021	39654	P3188		Use of "0" in Unquoted String
INQ	32.0.0023	39981	D3337		Data Base Stack
INQ	32.0.0024*	40208	P3334		SET DISPLAY Results in Imprope
INQ	32.0.0026	40230	P3655		Sorting on Subscripted Items
INQ	32.0.0027*	40905	P3387		Functions Performed Via Embedd
INQ	32.0.0031*	41349	P3443		Entering Input Before Previous
INQ	32.0.0032*	41353	P3444		"?AX" Not Recognized on Linear
INQ	32.0.0033*	41301	P3445		Display Item Name
INQ	32.0.0034*	41355	P3446		"#NONE", "#NO MORE" Termination
INQ	32.0.0035*	41099	P3447		Rounding Virtual Items
INQ	32.0.0036*	41097	P3555		Increase Number of Data Set Fu
INQ	32.0.0037*	41098	P3490		Recall of UPDATE Command
INQ	32.0.0038	41297	D3615		Segmented Value Arrays
INQ	32.0.0039	42415	P3701		Search on Index Random Set
INQ	32.0.0040*	42476	P3745		Virtual Items Evaluated
INQ	32.0.0041	42417	P3747		Improved Searching Capabilitie
INQ	32.0.0042*	42742	P3791		DISPLAY ALL Correction
INQ	32.0.0043*	42782	P3797		Report Control Items
INQ	32.0.0044*	42699	P3792		Do Not Close Data Base if TASK
INQ	32.0.0045	42783	P3823		DISPLAY ALL of Global Data
INTERFACE	32.0.0006	38011	D3170		Shared ACCESSROUTINES, Data Ba
INTERFACE	32.0.0007	38741	D3229		Data Bases at Different Releas
INTERFACE	32.0.0008	38011	D3170		Shared ACCESSROUTINES, Data Ba
INTERFACE	32.0.0009	38011	D3170		Shared ACCESSROUTINES, Data Ba
INTERFACE	32.0.0010*	39387	P3103		Sets not Invoked in Logical DB
INTERFACE	32.0.0011	39981	D3337		Data Base Stack
IXREF	32.0.0002	39235	P3147		Add DATABASE as XREF Item
IXREF	32.0.0003	40641	P3330		Remove PORT, SIGNAL Variable T
JOBFORMAT	32.0.0001	37177	D3140		New EOT/EOJ Format
JOBFORMAT	32.0.0002	37850	D3026		Bad Record Dump
JOBFORMAT	32.0.0003	38256	D3141		Decode Error Sectors
JOBFORMAT	32.0.0004	38436	P2863		Print Boxes in Ascending Order
JOBFORMAT	32.0.0005	37911	D3355		Log MLIP I/O Errors
JOBFORMAT	32.0.0006	41825	D3575		PBIT Time Accounting
JOBFORMAT	32.0.0008	38328	P3436		Change CONRAC to ODT
JOBFORMAT	32.0.0009	40835	D3468		Log New Open, Close Informatio
JOBFORMAT	32.0.0010	41457	D3501		Usage Information for I/O Devi
JOBFORMAT	32.0.0011*	41477	P3585		NOT READY Messages
JOBFORMAT	32.0.0012*	42084	P3608		UNITMNEMONICS Array
JOBFORMAT	32.0.0014*	42085	P3795		DP ALL Message
JOBFORMAT	32.0.0015	41060	D3594		Replace Logging of ORGHOST by
JOBFORMAT	32.0.0017	40663	D3356		New and Old ODT Messages
LCOBOL	32.0.0002*	34514	P2713		INVALID INDEX for WORKING-STOR
LCOBOL	32.0.0003	38547	P2873		SNTX for Compile with Syntax E
LCOBOL	32.0.0005	38546	D3235		Workfile Compiled for CANDE Co
LCOBOL	32.0.0006*	38544	P2906		Same Address for Two Level 77
LCOBOL	32.0.0007*	40644	P3547		General LOAD Instructions Agai
LOADDUMP	32.0.0001	39981	D3337		Data Base Stack
LOADDUMP	32.0.0004	41293	D3614		Eliminate OPEN INITIALIZE
LOADDUMP	32.0.0005*	41814	P3712		TITLE Not Parsed Correctly
LOADDUMP	32.0.0006	41816	P3713		Prevent SEG ARRAY Error
LOADER	32.0.0001	37991	P2778		IV
LOADER	32.0.0002	37991	P2778		IV
LOADER	32.0.0005	39120	D3157		LOADER Improvements
LOADER	32.0.0006	38733	P2962		Convert LOADER to NEWP
LOADER	32.0.0007	39120	D3157		LOADER Improvements
LOADER	32.0.0008	39052	P2986		206,207 Disk Pack Coldstart
LOADER	32.0.0009*	39056	P3053		INVALID ADDRESS Interrupt
LOADER	32.0.0011	38733	P2962		Convert LOADER to NEWP
LOADER	32.0.0019	40084	D3014		LH Command
LOADER	32.0.0029	41178	D3462		HALTLOADEU Messages
LOADER	32.0.0031*	41207	P3437		INVALID INDEX on MOD 63
LOADER	32.0.0035	41688	P3694		Sequence
LOGANALY	32.0.0004	38256	D3141		Decode Error Sectors
LOGANALY	32.0.0005	37911	D3355		Log MLIP I/O Errors
LOGANALY	32.0.0006*	40396	P3278		Maintenance Log Entries
LOGANALY	32.0.0007	40397	D3384		Add Starting, Ending Times to
LOGANALY	32.0.0009*	40065	P3362		DL Message INVALID INDEX?
LOGANALY	32.0.0010*	41943	P3582		Log Not Found
LOGANALY	32.0.0011	41060	D3594		Replace Logging of ORGHOST by
LOGANALY	32.0.0012*	42148	P3660		Unrecovered Errors Shown
LOGANALY	32.0.0013	42801	D3627		THAW Command

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '**')	PRI	NOTE	DESCRIPTION
LOGGER	32.0.0003*	39538	P3166	ORGMCS, DESTMCS Integer Type	
LOGGER	32.0.0004	40051	D3416	LOGGER vs DL LOG	
LOGGER	32.0.0005*	40062	P3363	Year to Date Sort Errors	
LOGGER	32.0.0006*	40064	P3364	No File Jobsummary	
LOGGER	32.0.0007	40835	D3468	Log New Open, Close Informatio	
MCP	32.0.0040	37117	P2538	Spurious PACK IN USE Message	
MCP	32.0.0070	37187	P2765	Seek Lost Message	
MCP	32.0.0080	37702	D3054	SWAPPER Enhancements	
MCP	32.0.0087	40663	D3356	New and Old ODT Messages	
MCP	32.0.0095	37186	P2766	Print Entire Buffer in PROGRAM	
MCP	32.0.0125	37990	D3056	Idle Patterns in Printer Dump	
MCP	32.0.0126	37988	P2625	Printer Dump Loop	
MCP	32.0.0151	38063	P2770	Non-MCS DCWRITE	
MCP	32.0.0153	38069	D3102	BNA MCS	
MCP	32.0.0163	37155	P2773	Hung Library Maintenance after	
MCP	32.0.0173	38256	D3141	Decode Error Sectors	
MCP	32.0.0179	38197	D3108	New Data Base Stack Structure	
MCP	32.0.0182*	38050	P2790	RESTORE Vs. DS	
MCP	32.0.0184	37765	P2792	Halting DCP 0 in SECONDARYINIT	
MCP	32.0.0202	40663	D3356	New and Old ODT Messages	
MCP	32.0.0221	40663	D3356	New and Old ODT Messages	
MCP	32.0.0225	38472	P2856	DISCSTATUS Vs. BLASTUNIT	
MCP	32.0.0237	38481	P2859	UNITSTATISTIC	
MCP	32.0.0247	37824	P2877	Dump Mechanism	
MCP	32.0.0253	38561	P2892	FORGETCHECK	
MCP	32.0.0254*	38560	P2893	Single Bit Error Logging	
MCP	32.0.0264	40663	D3356	New and Old ODT Messages	
MCP	32.0.0265	38469	D3089	Memory Dump Tape Record Format	
MCP	32.0.0272*	38571	P2896	COPY Vs. FAMILYINDEX	
MCP	32.0.0277	31307	D3094	Autoprint Can Run in Local Mem	
MCP	32.0.0278	37798	D3095	Intrinsics in Local Memory	
MCP	32.0.0288	40663	D3356	New and Old ODT Messages	
MCP	32.0.0290	38597	D3104	XALGOL Deimplemented	
MCP	32.0.0292	38600	P2919	Send Message to Local DBS	
MCP	32.0.0293	38486	P2920	Memory Management	
MCP	32.0.0294*	38484	P2921	Saving Memory Mods	
MCP	32.0.0298	38566	P2924	PROCID Vs. "M[MSGADDR+3]"	
MCP	32.0.0299	38496	P2925	Memory Management	
MCP	32.0.0301*	38740	P2927	"7-Track" Library Tapes	
MCP	32.0.0303	38736	P2929	RECONFIGURATION	
MCP	32.0.0307	38735	P2958	RESOURCECHECK	
MCP	32.0.0311*	38724	P2959	Overlay File Corruption	
MCP	32.0.0323	38564	D3122	Implied Concatenations Made Ex	
MCP	32.0.0324	38757	P2961	Incorrect Family Substitution	
MCP	32.0.0329	36196	P2914	Record Sequence	
MCP	32.0.0330	41830	D3600	Delete Old Intrinsics	
MCP	32.0.0336	40663	D3356	New and Old ODT Messages	
MCP	32.0.0337	38564	D3122	Implied Concatenations Made Ex	
MCP	32.0.0338	38492	P2865	SYSTEMSTATUS (11): SWAPPER Par	
MCP	32.0.0349	39121	P2862	JOBDESC Complement	
MCP	32.0.0352	38494	P2963	SUPERPLUCK Hangs/Dumps Vs. SCH	
MCP	32.0.0363	40663	D3356	New and Old ODT Messages	
MCP	32.0.0367*	39135	P2994	Parameter Mismatch	
MCP	32.0.0369	39058	P3002	Rem ve B6700 FINDMEMORYPARITY	
MCP	32.0.0373	42792	P3824	Memory Organization, Local MCP	
MCP	32.0.0378	39044	P2878	Bad Momaddress	
MCP	32.0.0380	39068	P3025	DUMPANALYZER Recognizes Frozen	
MCP	32.0.0382	39279	P3027	STARTSYSTEM	
MCP	32.0.0383	39085	P3028	Check for Library Capable	
MCP	32.0.0384	39086	P3029	Correct FORGET CHECK on Librar	
MCP	32.0.0392	39090	D3228	Printer Dump Hardware Interrup	
MCP	32.0.0395	39088	P3019	NOT READY RSVP	
MCP	32.0.0396	39085	P3028	Check for Library Capable	
MCP	32.0.0397	39089	P3020	Attribute Handling for Data Ba	
MCP	32.0.0399	39122	P3024	MOVE Vs. Disk Pack Type 206	
MCP	32.0.0411	39060	P3047	Tightly-Coupled Main Memory DC	
MCP	32.0.0423*	39123	P3052	PATHRES DS	
MCP	32.0.0424*	39087	P3196	ZOT Library Template Marker in	
MCP	32.0.0427	38378	D3257	IVR Facility on Mark 33 Releas	
MCP	32.0.0429	39212	P3063	Data Base Equation Implementat	
MCP	32.0.0430*	39104	P3064	SWAPPER Hung Vs. Controlcard	
MCP	32.0.0431*	39099	P3065	Job File Roll Out	
MCP	32.0.0433	39214	P3066	DMSOPEN Contiguous Save Memory	
MCP	32.0.0446	39110	P3096	NO GO PAST Protection	
MCP	32.0.0448	40663	D3356	New and Old ODT Messages	
MCP	32.0.0458	39063	D3252	Semidependent Tasks, VISIBILIT	
MCP	32.0.0459	38785	P3075	Printer Dump to Drum Printer	
MCP	32.0.0460*	38784	P3097	Separate Halt/Load Packs	
MCP	32.0.0463	39110	P3096	NO GO PAST Protection	
MCP	32.0.0468	38564	D3122	Implied Concatenations Made Ex	
MCP	32.0.0474	38790	D3281	Attribute Handling	
MCP	32.0.0481	39065	P3100	RESIZE Reorganization	

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '**')	PRI	NOTE	DESCRIPTION
MCP	32.0.0492	39678	P3098		DMSUPDATEDISKHEADER NOOP
MCP	32.0.0493	39066	P3127		TC Overlay File Corruption
MCP	32.0.0494	39389	P3112		DMSII Exception Categories
MCP	32.0.0500	39680	P3115		FILECARDS Attribute FORGETCHEC
MCP	32.0.0516	39700	P3116		FORGETCHECK after Memory Excee
MCP	32.0.0517	39701	P3117		SWAPPER, Stack Stretch
MCP	32.0.0518	39618	P3136		Forgotten PIB
MCP	32.0.0519	39691	D3282		Virtual Memory Size Statistics
MCP	32.0.0529	39535	P3128		COPY&COMPARE Vs. Reel Switch
MCP	32.0.0537	39549	P3129		STATISTICS
MCP	32.0.0538	39548	P3130		Scheduling on B6800 Multiproce
MCP	32.0.0539	39547	P3131		DL SUMLOG
MCP	32.0.0540	39550	P3132		Separate Halt/Load Families
MCP	32.0.0552	39693	P3133		Library Maintenance IOCB
MCP	32.0.0553	39520	P3134		INTRINSICINFO
MCP	32.0.0554	39521	P3135		"CHECKPOINTED" Swaptask
MCP	32.0.0555	39522	D3292		OLAYINFO Analysis
MCP	32.0.0556	39523	P3138		Ready Queue Time
MCP	32.0.0557	39549	P3129		STATISTICS
MCP	32.0.0561	39582	P3148		Memory Dumps on a Shared Resou
MCP	32.0.0563	39524	P3150		ACTIVETIME
MCP	32.0.0577	39705	D3309		ITINERARY Task Attribute
MCP	32.0.0582	39587	P3203		External By Calling Library Pr
MCP	32.0.0603	39707	D3311		ORGHOSTNAME Attribute Deimplem
MCP	32.0.0609	39671	P3173		Printing of Inuse Code Segment
MCP	32.0.0610	40663	D3356		New and Old ODT Messages
MCP	32.0.0628*	39609	P3197		IPC Swapjobs Vs. Subspace Grow
MCP	32.0.0629	40663	D3356		New and Old ODT Messages
MCP	32.0.0634	39906	P3179		Graph for Data Base Users
MCP	32.0.0640*	39615	P3219		Working Sets
MCP	32.0.0651*	39696	P3233		UNIT 0
MCP	32.0.0652	40016	P3206		Make EBCDICTOWORD Inline
MCP	32.0.0654	39525	P3222		Correct PBIT of Zero Length Do
MCP	32.0.0664	40067	P3223		Defunct in DUMPBOOTSTRAPPER
MCP	32.0.0684	40080	P3224		Attribute Grabber Fault
MCP	32.0.0686	39126	P3227		PAST Order
MCP	32.0.0689*	40081	P3228		Resumed ST Tasks have Excess O
MCP	32.0.0693	39527	P3240		Usercode on 14-level File Name
MCP	32.0.0696*	40332	P3238		Card Reader Error Recovery
MCP	32.0.0702*	40094	P3239		Copying Too Many Files
MCP	32.0.0707	40264	D3352		Modifications to Support Port
MCP	32.0.0710	38305	P3255		Logging Internal File Name
MCP	32.0.0711	40104	D3399		CM Vs. Duplicated MCPs
MCP	32.0.0718	40287	D3341		Support of Old Codefiles
MCP	32.0.0724	38311	P3264		Logging Row Index
MCP	32.0.0725	40424	P3280		Avoid Bad Search from STACKSTR
MCP	32.0.0729	40038	D3346		RJE Vs. DLBACKUP
MCP	32.0.0732*	40105	P3257		Increase Maximum BDNUMBER
MCP	32.0.0743	40373	P3279		BACKUPQUEUER
MCP	32.0.0744	40374	D3400		Analyze Library Parameter Mism
MCP	32.0.0746	40103	D3381		Improvements to Working Set Sh
MCP	32.0.0747	40360	P3281		Programmed Operator Interrupt
MCP	32.0.0754	40229	P3282		Pass Control File Pack Name to
MCP	32.0.0756	40378	P3283		AUTOPRINT with FMed Printer
MCP	32.0.0764	40379	D3386		New Userdata Error Code "(42)"
MCP	32.0.0774	40040	P3277		Tasks Suspended by WSSHERIFF
MCP	32.0.0775	40103	D3381		Improvements to Working Set Sh
MCP	32.0.0784	40389	D3379		Set Library Function
MCP	32.0.0787	40663	D3356		New and Old ODT Messages
MCP	32.0.0792	38100	D3234		Terminal Transfer
MCP	32.0.0798	40488	P3335		Handle Null Subsystem in Unrav
MCP	32.0.0805	40395	P3301		Excess Working Set Sheriff Ove
MCP	32.0.0835	40744	P3321		Bounce Dump
MCP	32.0.0836*	40750	P3319		INV OP in Attribute Handler
MCP	32.0.0843	40709	D3409		IAD Not Supported on B6900
MCP	32.0.0850	40756	D3411		Priority, DI Information
MCP	32.0.0887	40776	D3573		PBIT Time Accounting
MCP	32.0.0888*	40778	P3339		TAPESEARCH
MCP	32.0.0894*	40785	P3336		Fault Because of Missing Intri
MCP	32.0.0899	40773	P3338		Analyze Library Template
MCP	32.0.0900	40986	D3425		Delete PORTS, SIGNALS
MCP	32.0.0905	40103	D3381		Improvements to Working Set Sh
MCP	32.0.0910*	40805	P3355		Program Marked as Swapjob
MCP	32.0.0911	40794	P3356		QT PB MT
MCP	32.0.0933*	40820	P3358		LOADALABEL vs MULTIFILE
MCP	32.0.0940*	40786	P3496		LOCKTRACE Option
MCP	32.0.0941*	40783	P3375		GETSTATUS FORGETCHECK
MCP	32.0.0942	40787	P3376		LOADCONTROL to Tape
MCP	32.0.0947	40834	D3480		Intrinsic Mapping
MCP	32.0.0950	40835	D3468		Log New Open, Close Informatio
MCP	32.0.0957	41159	P3448		Checkpoint/Restart for Program
MCP	32.0.0958*	41173	P3449		Restart of Serial Disk Files N

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '**')	PRI	NOTE	DESCRIPTION
MCP	32.0.0959	41174	P3450		Rerun of COBOL Files with Use
MCP	32.0.0962	41021	D3481		New Port Analysis
MCP	32.0.0963	40749	P3451		Multiple Wait
MCP	32.0.0967	41190	P3452		STACKLIMIT Task Attribute
MCP	32.0.0971	40298	D3485		OPEN Function Values
MCP	32.0.0972*	41191	P3453		Correct Time Slice Calculation
MCP	32.0.0973	41197	D3483		DONT CARE Libraries
MCP	32.0.0974*	41198	P3454		Allow Halt/Load After Power Up
MCP	32.0.0976	41196	D3484		USECATDEFAULT Vs. DIAGNOSTICS
MCP	32.0.0986	41200	P3455		BOUNCE Message
MCP	32.0.0988*	41206	P3456		SYSTEMSTATUS Vs. UNITMOVER
MCP	32.0.0991*	41209	P3522		Stackswap Vs. Stackstretcher
MCP	32.0.0993	41202	P3461		Checkpoint of Large Size Stack
MCP	32.0.0997*	41213	P3457		Calculation of Code Core Estim
MCP	32.0.0998*	41437	P3458		FA Swapjob
MCP	32.0.1000	37702	D3054		SWAPPER Enhancements
MCP	32.0.1007	41212	P3620		Library Maintenance Tape Errors
MCP	32.0.1009*	40808	P3521		REELSWITCH Vs. Density
MCP	32.0.1013	41447	P3482		Dynamic EBIT
MCP	32.0.1017	41496	P3478		Improve Run Time Parameter Che
MCP	32.0.1020	41474	D3502		Shrink Frozen Library's Stack
MCP	32.0.1021*	41476	P3479		Multiple Flagreaders
MCP	32.0.1022	41485	D3516		Library Function Names
MCP	32.0.1023*	41486	P3480		Halt/Load Memory Configuration
MCP	32.0.1024	41460	D3500		Changes to SYSTEMSTATUS Calls
MCP	32.0.1025	41457	D3501		Usage Information for I/O Devi
MCP	32.0.1026	41040	P3568		Programs Using Mark 31 Ports a
MCP	32.0.1028*	41477	P3585		NOT READY Messages
MCP	32.0.1029	41384	P3527		Data Base TITLE Attribute Veri
MCP	32.0.1031*	41692	P3523		Destname on ACR Codefiles
MCP	32.0.1032*	41689	D3529		PARTNER , EXCEPTIONTASK Remov
MCP	32.0.1035*	36098	P3524		Reservedisk and Userdata Heade
MCP	32.0.1036	41693	D3536		SYSTEMSTATUS, IOTRACE for MLIP
MCP	32.0.1038*	41702	P3607		Tape Verify
MCP	32.0.1040*	41699	P3519		Report on Exclusive Files
MCP	32.0.1044	41449	P3569		Close Port Values
MCP	32.0.1045*	41698	D3548		DBS in Local Memory Vs. Nonexc
MCP	32.0.1047*	41478	P3621		Paths, FREE, DISKSTATUS Proble
MCP	32.0.1048	41710	D3550		Eliminate DL Network
MCP	32.0.1050*	41696	P3528		Units Equal Character Vs. Back
MCP	32.0.1051	40877	P3570		AVAILABLE Type of File Open
MCP	32.0.1057*	41489	P3529		Password Handling
MCP	32.0.1058	41714	D3551		Password Manipulation
MCP	32.0.1059*	41715	P3530		"CONTROLCARD(QUEUE,7)"
MCP	32.0.1063*	41719	P3531		Stack Overflow Handling
MCP	32.0.1077*	41736	P3575		Forgetcheck After Programdump
MCP	32.0.1078	41049	D3564		WFL Task Fault Across Network
MCP	32.0.1079*	41938	P3576		INVALID OP in DMSCAUSE
MCP	32.0.1080*	41940	P3577		SWAPPER Vs. SIB
MCP	32.0.1081*	41720	P3578		Software Interrupt Handling
MCP	32.0.1083*	41944	P3579		JOBDESC Vs. Nonexchanged Units
MCP	32.0.1089	41952	D3579		Processkill Event Errors
MCP	32.0.1094*	41955	P3609		Checkpoint Restart with Array
MCP	32.0.1095	41951	P3631		New Fine Priority Algorithm
MCP	32.0.1100*	41687	P3610		Volume Library
MCP	32.0.1110	42106	P3632		Messages
MCP	32.0.1112*	42109	P3647		Resource Wait
MCP	32.0.1113	41060	D3594		Replace Logging of ORGHOST by
MCP	32.0.1115	42068	D3595		Intrinsic to Support Library M
MCP	32.0.1128*	41956	P3648		DESTNAME Attribute
MCP	32.0.1136*	42139	P3649		ODT Queue
MCP	32.0.1137*	42139	P3649		ODT Queue
MCP	32.0.1138*	42192	P3650		DS Permanent Library
MCP	32.0.1140*	42197	P3651		Expand Max Task Parameters
MCP	32.0.1142*	42149	P3652		WFL Subroutines
MCP	32.0.1143*	41479	P3746		Scratch Tape Without Write Rin
MCP	32.0.1145	40287	D3341		Support of Old Codefiles
MCP	32.0.1147*	42162	P3653		FS and DS ODT Inputs
MCP	32.0.1148	40663	D3356		New and Old ODT Messages
MCP	32.0.1149	40663	D3356		New and Old ODT Messages
MCP	32.0.1154*	42086	P3654		Read Header Reorganization
MCP	32.0.1160	42252	D3618		COBOL74 Vs. WORD Mode Files
MCP	32.0.1171*	42089	P3685		"UR-"
MCP	32.0.1173	42090	P3748		DUP FILE Message
MCP	32.0.1176	42169	D3619		SYSTEMSTATUS Type 4 General Un
MCP	32.0.1185	42215	D3620		Hostname in Heading
MCP	32.0.1186*	40757	P3686		INVALID OP in Presence Bit
MCP	32.0.1189*	42180	P3687		BDNAME SEG ARRAY Fault
MCP	32.0.1195*	42184	P3688		Autoprint AX Command
MCP	32.0.1199	42119	D3606		OFFSET and DELTA
MCP	32.0.1212*	42178	P3710		DMSCLOSE Vs. CONTROLLER
MCP	32.0.1215	42122	P3781		TAPEDUMP, Report Block, Rewrit

B6000 SERIES MARK 32

B6000 SERIES PATCH TABLE (PATCH CLOSING FTRS MARKED WITH '*')
 SOFTWARE PATCH PRI NOTE DESCRIPTION

MCP	32.0.1231	42468	P3782	Resize Overlayable Arrays in G
MCP	32.0.1238	42503	P3801	Corruption of LIBUSEMAP
MCP	32.0.1246	42381	P3787	GUARDFILE Vs. CANDE
MCP	32.0.1248	42566	D3642	CANDE Vs. Foreign Tasks
MCP	32.0.1254	42532	P3784	Avoid Hung Printers
MCP	32.0.1266	42501	D3641	Compile Time Options
MCP	32.0.1275	42328	P3817	Erroneous DS when DBS Initiati
MCP	32.0.1287	43207	D3637	Add Additional Information to
MCP	32.0.1295	42122	P3781	TAPE_DUMP, Report Block, Rewrit
MCP	32.0.1297	42501	D3641	Compile Time Options
MCP	32.0.1301	40080	P3224	Attribute Grabber Fault
MCP	32.0.1305	42850	D3652	Libraries Vs. "??DS", CM , REC
MCP	32.0.1309	42790	P3802	FIB Creation Locking
MCP	32.0.1310	42792	P3824	Memory Organization, Local MCP
MCP	32.0.1311	42793	P3822	EXCEPTIONTASK Visibility
MCP	32.0.1334	39787	D3354	Intrinsic to Library Conversio
MCP	32.0.1355	42864	P3825	Parity on Presencebit Stackove
MCP-GENERAL	32.0.1229	42501	D3641	Compile Time Options
MLIP	32.0.0033	37495	D3142	Initialization Routines for B6
MLIP	32.0.0034	37275	P2763	Implement PRINTER_DUMP
MLIP	32.0.0048	37198	P2764	PRINTIOCB Interface Analyzes I
MLIP	32.0.0049	37495	D3142	Initialization Routines for B6
MLIP	32.0.0058	37495	D3142	Initialization Routines for B6
MLIP	32.0.0069	37461	P3680	Peripheral Test Driver
MLIP	32.0.0083	37461	P3680	Peripheral Test Driver
MLIP	32.0.0154	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0156	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0157	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0158	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0174	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0175	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0196	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0208	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0209	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0267	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0270	37911	D3355	Log MLIP I/O Errors
MLIP	32.0.0312	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0313	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0348	37495	D3142	Initialization Routines for B6
MLIP	32.0.0376	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0454	38795	D3251	GETSTATUS/SETSTATUS Enhancemen
MLIP	32.0.0466	38055	D3123	B6900 Peripheral Test Driver
MLIP	32.0.0659	38795	D3251	GETSTATUS/SETSTATUS Enhancemen
MLIP	32.0.0723	38795	D3251	GETSTATUS/SETSTATUS Enhancemen
MLIP	32.0.0943	40663	D3356	New and Old ODT Messages
MLIP	32.0.1064	40663	D3356	New and Old ODT Messages
NDL	32.0.0003*	37862	P2714	Page Between VOIDT and POP VOI
NDL	32.0.0004	37761	D3028	Message-Oriented Datacom
NDL	32.0.0005	37761	D3028	Message-Oriented Datacom
NDL	32.0.0006	41149	D3204	Changes for DCP Character Orie
NDL	32.0.0007	37863	P2804	ENTER
NDL	32.0.0015*	39940	P3256	Inhibit Sync Edit Wrong
NDL	32.0.0016	40434	D3385	Different Terminal Addresses
NDL	32.0.0021	41068	P3599	Enlarge UNFO Array
NDL	32.0.0022*	41067	P3604	"\$NETWORK" Option
NDL	32.0.0023*	39471	P3605	Clear Linetable Array
NDL	32.0.0024*	42678	P3751	Call BRANCHLINK
NEWP	32.0.0014	37757	D2973	"AT <library id>" Allowed
NEWP	32.0.0015	37758	D2974	INITIALIZATION is Reserved Wor
NEWP	32.0.0016	34311	D3031	Conditional Operators
NEWP	32.0.0019	37286	D3032	Implement Control State Blocks
NEWP	32.0.0020	37283	D3033	Prevent GOTO Into FOR Statemen
NEWP	32.0.0023	38234	P2743	Better Listing for Modular SEP
NEWP	32.0.0025	37753	D3068	ALTERNATIVES and INITIALIZATIO
NEWP	32.0.0026	37750	P2805	Error for Missing Procedure
NEWP	32.0.0028	38231	P2806	Attribute for Task Array Eleme
NEWP	32.0.0029	38230	P2807	Error for Empty Parenthesis
NEWP	32.0.0031	38232	D3064	Arrays with Unspecified Bounds
NEWP	32.0.0037	38536	D3106	Clarification of MAKEPCW Restr
NEWP	32.0.0038	38548	P2934	IXREF Environments for Cheap B
NEWP	32.0.0039	38445	D3103	Events and Event Arrays as Par
NEWP	32.0.0040	38773	D3110	"<procedure name>.VALUE"
NEWP	32.0.0041	38549	P2946	Reduced Time for NEWP XREF
NEWP	32.0.0042	38772	D3058	Inline Procedures
NEWP	32.0.0043	38446	P2947	Prevent INVALID INDEX
NEWP	32.0.0044	38228	D3151	Increased Host Blocksize
NEWP	32.0.0045	38533	D3152	Allow the MCP to Freeze as a L
NEWP	32.0.0046	38227	P2733	Improve FILE/LIBRARY Declarati
NEWP	32.0.0049	39145	D3158	Match NEWP Codefile level to B
NEWP	32.0.0052	39146	P3057	Address Equation to Undeclared
NEWP	32.0.0053	39147	D3283	"XREFing" Alternatives
NEWP	32.0.0055	39426	D3260	FIRSTFREEDOCCELL Now Defaults t

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '*')	PRI	NOTE	DESCRIPTION
NEWP	32.0.0056	39434	P3082	Error For Duplicate Case Eleme	
NEWP	32.0.0058	38226	P3087	SEPCOMP Loses Source Lines	
NEWP	32.0.0059	38225	D3261	New Fault Name, LIBLINKFAULT	
NEWP	32.0.0060	39482	D3262	New \$ Option, STANDALONE	
NEWP	32.0.0064*	39514	D3299	Close LINE and ERRORFILE	
NEWP	32.0.0065	39464	D3300	Segment Identifiers	
NEWP	32.0.0066	39750	P3140	Prevent Compiler SEG ARRAY Fau	
NEWP	32.0.0080	39783	D3401	PORTS Option Discontinued	
NEWP	32.0.0081*	40639	P3297	Comparing Pointer	
NEWP	32.0.0089	40133	P3344	Stack Overflow	
NEWP	32.0.0090	40128	D3420	PARITYFAIL1 Fault	
NEWP	32.0.0091	40130	D3421	HEYOU Disallowed on B7000 Syst	
NEWP	32.0.0092	40131	D3422	ZAP Intrinsic for B7000	
NEWP	32.0.0098	40139	D3486	Direct I/O	
NEWP	32.0.0101	40135	P3438	BINDINFO for Alternatives	
NEWP	32.0.0102	41229	D3487	MCP Code File Row Size = 504	
NEWP	32.0.0103	41405	P3470	SEPCOMP Creates Erroneous Stac	
NEWP	32.0.0108	41424	D3535	PACKDECIMAL Intrinsic	
NEWP	32.0.0109	40549	D3530	FUNCTIONNAME, LIBACCESS Attri	
NEWP	32.0.0110	41502	D3549	"8-Digit" Patch Marks	
NEWP	32.0.0115	41508	D3590	DESCRIPTOR Procedures to Libra	
NEWP	32.0.0116	41718	D3591	Procedure Entry via References	
NEWP	32.0.0119	42099	P3627	Null Environments in XREF	
NEWP	32.0.0120	42100	P3628	Extraneous XREF Environments	
NEWP	32.0.0121	40131	D3422	ZAP Intrinsic for B7000	
NEWP	32.0.0122	41853	D3592	PROTECTED Option in Library Ex	
NEWP	32.0.0123	41857	D3593	REGISTERS and DLL	
NEWP	32.0.0124	41862	P3629	"<arithmetic expression> IN <t	
NEWP	32.0.0127	42052	D3626	Resizing EVENT ARRAYS	
NEWP	32.0.0128	41865	D3598	SYSTEMLIB Library Attribute	
PATCH	32.0.0002	34320	D3007	Patch Numbers with \$.VERSION/C	
PATCH	32.0.0003*	34319	P2715	Character Mode Files	
PATCH	32.0.0004*	34317	P2716	Out of Sequence Patches	
PATCH	32.0.0005*	34316	P2717	\$.GUARD Option	
PATCH	32.0.0006	38089	D3034	New List and Compare Options	
PATCH	32.0.0007	38088	D3035	VERSION May Be RESET	
PATCH	32.0.0011	42105	P3630	Line Width	
PATCH	32.0.0012	42107	D3599	MARKBLANK and DELIMOPT Options	
PLI	32.0.0002*	33467	P2718	Exponentiation Mixing Operands	
PLI	32.0.0003*	33465	P2719	Correct ATAND	
PLI	32.0.0004*	37896	P2720	Double Precision PICTURE 'H'	
PLI	32.0.0005*	37877	P2721	SUBSTR of Binary Data	
PLI	32.0.0006*	37878	P2722	Branching from Start of Segmen	
PLI	32.0.0007*	37881	P2808	Precedence of Operators	
PLI	32.0.0008*	37880	P2809	Duplicate Label and Entry Name	
PLI	32.0.0009*	37895	P2810	PIC'(12)HS'	
PLI	32.0.0010	38188	D3090	Compiler Info Word in Seg Zero	
PLI	32.0.0016*	37892	P2840	OR Operation on BDMS Field Bit	
PLI	32.0.0017*	37891	P2841	":=" as Assignment Operator	
PLI	32.0.0020*	38461	P2850	Undefined Format	
PLI	32.0.0021*	38448	P2870	Logical Operations on BIT Stri	
PLI	32.0.0022*	38459	P2907	Stack Cell for ELSE	
PLI	32.0.0023*	38455	P2911	SORT Compares on Pictured Keys	
PLI	32.0.0024*	38456	P2912	LABELTYPE='OMITTED'	
PLI	32.0.0025	38458	P2935	More than 48 "%DO" Statements	
PLI	32.0.0026*	38450	P2769	Statement Numbers in Error Mes	
PLI	32.0.0027	38457	P2767	System File Attribute Paramete	
PLI	32.0.0029*	39173	P2982	Independent Task Initiation	
PLI	32.0.0030*	39183	P3006	Pointer Initialization	
PLI	32.0.0031*	39180	P3016	Ignored LENGTH or INITIAL Spec	
PLI	32.0.0032*	39178	P3017	String Builtin Functions	
PLI	32.0.0033*	39184	P3018	DIMENSION Not First Attribute	
PLI	32.0.0034*	39163	P3043	EXCEPT Builtin Function	
PLI	32.0.0035	39177	P3044	Illegal Primary	
PLI	32.0.0037*	39175	P3058	PIC 'X' Array Elements	
PLI	32.0.0038*	39174	P3062	Compiletime DO and INCLUDE	
PLI	32.0.0039	39171	P3088	BIT Compares	
PLI	32.0.0041*	39167	P3143	PUT EDIT of PIC 1 Variables	
PLI	32.0.0044*	39951	P3211	Parameter Mismatch with NOBIND	
PLI	32.0.0045	39952	P3212	Library Capable Bit	
PLI	32.0.0047	40264	D3352	Modifications to Support Port	
PLI	32.0.0048	40258	P3310	Inappropriate Warning Messages	
PLI	32.0.0049	40251	P3367	PUT EDIT of Bit Variables	
PLI	32.0.0050	40252	P3337	Bit String Comparisons	
PLI	32.0.0051*	41656	P3471	Error TASK IDENTIFIER REQUIRED	
PLI	32.0.0052*	41657	P3472	Bit String Defined	
PLI	32.0.0053*	41655	P3473	Bit Overlay Defining	
PLI	32.0.0054*	41652	P3474	WRITE Without FROM	
PLI	32.0.0055*	41650	P3475	Compiler Loop Corrected	
PLI	32.0.0056*	41653	P3476	Bad Declaration Caused INVALID	
PLI	32.0.0057*	41649	P3477	Lost Text from Compile-Time Pr	
PLI	32.0.0058*	41647	P3510	Question Mark	

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '*')	PRI	NOTE	DESCRIPTION
PLI	32.0.0059*	41645	P3511		Bad Lineinfo
PLI	32.0.0060*	41646	P3512		External File Variables
PLI	32.0.0063*	41636	P3513		"LENGTH(String(<id>))" in SUBS
PLI	32.0.0064*	41634	P3548		PICTURE Y
PLI	32.0.0065*	41632	D3546		Binding Programs with DUMP Opt
PLI	32.0.0066*	41631	P3549		ROUND Builtin Function
PLI	32.0.0067*	41630	P3550		Level 3 Warning Message
PLI	32.0.0068*	41628	P3551		Problem with Pictures
PLI	32.0.0069*	41624	P3593		Large Structures with INITIAL
PLI	32.0.0072*	41617	P3752		Bit Expressions
PLINTRN	32.0.0006	41830	D3600		Delete Old Intrinsics
PLINTRN	32.0.0014	40264	D3352		Modifications to Support Port
PLISUPP	32.0.0007*	39947	P3213		ISAM, DELETE
PLISUPP	32.0.0008*	39768	P3214		ISAM, Security Error
PLISUPP	32.0.0009*	39771	P3215		ISAM, Duplicate Record Keys
PLISUPP	32.0.0010*	39946	P3216		REWRITE, Zero in First Word
PLISUPP	32.0.0011*	39767	P3217		Empty ISAM File
PLISUPP	32.0.0012	39948	D3394		Implicit Opening of Files
PLISUPP	32.0.0013	40264	D3352		Modifications to Support Port
PLISUPP	32.0.0014	40250	P3365		ISAM Parity Error to COBOL
PLISUPP	32.0.0015	40249	P3366		ISAM Logical Delete
PLISUPP	32.0.0016	40251	P3367		PUT EDIT of Bit Variables
PLISUPP	32.0.0020*	41638	P3514		PL/I Programdump
PLISUPP	32.0.0021*	41637	P3515		PL/I Programdump
PLISUPP	32.0.0022*	41633	P3552		GET DATA Loop
PRINTAUDIT	32.0.0001	37218	P2760		Use Audit Record Information T
PRINTAUDIT	32.0.0005*	37723	P3071		READLOCKNOPURGE Removed
PRINTAUDIT	32.0.0006	39981	D3337		Data Base Stack
PRINTAUDIT	32.0.0008	40887	D3460		Preallocation of Direct Data S
PRINTAUDIT	32.0.0009	40887	D3460		Preallocation of Direct Data S
PRINTBIND	32.0.0002	40135	P3438		BINDINFO for Alternatives
PRINTBIND	32.0.0003*	41330	P3596		Eliminate EOF NO LABEL Abort
PRINTBIND	32.0.0004*	41329	P3597		Handling of Procedure Paramete
PRINTBIND	32.0.0005*	41332	P3595		Data Base BINDINFO
PROPERTIES	32.0.0004	37320	D3120		DASDL/REORGANIZATION Enhanceme
PROPERTIES	32.0.0006	37540	D3084		Simplification of REORGANIZATO
PROPERTIES	32.0.0009	38011	D3170		Shared ACCESSROUTINES, Data Ba
PROPERTIES	32.0.0010	38021	D3050		Halt/Load Recovery Sequencing
PROPERTIES	32.0.0011	38011	D3170		Shared ACCESSROUTINES, Data Ba
PROPERTIES	32.0.0013	38210	D3113		Delete READAHEADB
PROPERTIES	32.0.0014	38122	D3116		Put Subsystem ID in Text
PROPERTIES	32.0.0015	38197	D3108		New Data Base Stack Structure
PROPERTIES	32.0.0016	38134	D3118		Remove Properties for 27 Links
PROPERTIES	32.0.0017	38011	D3170		Shared ACCESSROUTINES, Data Ba
PROPERTIES	32.0.0019*	38763	P2967		AREASZ Greater Than 65536 Tru
PROPERTIES	32.0.0020	38011	D3170		Shared ACCESSROUTINES, Data Ba
PROPERTIES	32.0.0021	38762	D3270		Implement COPYAUDIT WFL Deck
PROPERTIES	32.0.0022	38759	D3162		DASDL Defaults
PROPERTIES	32.0.0023	38011	D3170		Shared ACCESSROUTINES, Data Ba
PROPERTIES	32.0.0024	39216	D3274		Structure Calculations
PROPERTIES	32.0.0026	39363	D3272		Restructure Description File P
PROPERTIES	32.0.0027	38011	D3170		Shared ACCESSROUTINES, Data Ba
PROPERTIES	32.0.0028	39217	D3353		DUMPDIR Enhancements
PROPERTIES	32.0.0029	39356	D3331		ACCESSROUTINES Error Messages
PROPERTIES	32.0.0030	39649	D3317		Crunch PROPERTIES Symbolic
PROPERTIES	32.0.0031	39639	D3314		Rebuild Across File Discontinu
PROPERTIES	32.0.0032	39981	D3337		Data Base Stack
PROPERTIES	32.0.0034	40887	D3460		Preallocation of Direct Data S
PROPERTIES	32.0.0035	40887	D3460		Preallocation of Direct Data S
PROPERTIES	32.0.0037	40887	D3460		Preallocation of Direct Data S
PROPERTIES	32.0.0038	40887	D3460		Preallocation of Direct Data S
PROPERTIES	32.0.0039*	41770	P3556		Initialization of Disjoint Uno
PROPERTIES	32.0.0041	42985	P3814		Incorrect Output Report
PTNCTL	32.0.0001	37540	D3084		Simplification of REORGANIZATO
PTNCTL	32.0.0002	38748	D3466		28 to 29 Conversion Options Re
PTNCTL	32.0.0004	38748	D3466		28 to 29 Conversion Options Re
PTNCTL	32.0.0005	39981	D3337		Data Base Stack
RECOVERY	32.0.0010	37351	P2753		Remove RSFILE Declaration
RECOVERY	32.0.0013	37540	D3084		Simplification of REORGANIZATO
RECOVERY	32.0.0015	38021	D3050		Halt/Load Recovery Sequencing
RECOVERY	32.0.0017	38193	D3051		Switch Back to Primary Audit
RECOVERY	32.0.0018	38196	P2789		Ensure REBUILD Restart
RECOVERY	32.0.0019	38205	P2879		RLA for Abort, Halt/Load Only
RECOVERY	32.0.0020	38201	P2887		Do Not Print Audit Block if EO
RECOVERY	32.0.0022	38011	D3170		Shared ACCESSROUTINES, Data Ba
RECOVERY	32.0.0023*	39200	P3036		Test Generates Too Much Code
RECOVERY	32.0.0024*	37723	P3071		READLOCKNOPURGE Removed
RECOVERY	32.0.0025	38011	D3170		Shared ACCESSROUTINES, Data Ba
RECOVERY	32.0.0026	39385	P3109		Update Disk Header
RECOVERY	32.0.0027	39390	P3144		Read Wrong Audit Block
RECOVERY	32.0.0028	39394	D3289		Allow Normal REBUILD/ROLLBACK
RECOVERY	32.0.0029	39639	D3314		Rebuild Across File Discontinu

B6000 SERIES MARK 32

SOFTWARE	PATCH	PRI	NOTE	DESCRIPTION
RECOVERY	32.0.0030	39217	D3353	DUMPDIR Enhancements
RECOVERY	32.0.0031	39910	D3452	Abort Acceleration
RECOVERY	32.0.0032	39981	D3337	Data Base Stack
RECOVERY	32.0.0039	38011	D3170	Shared ACCESSROUTINES, Data Ba
RECOVERY	32.0.0040	40887	D3460	Preallocation of Direct Data S
RECOVERY	32.0.0041	40887	D3460	Preallocation of Direct Data S
RECOVERY	32.0.0042	40887	D3460	Preallocation of Direct Data S
RECOVERY	32.0.0043*	41090	D3464	Quickfix Fails on Inconsistent
RECOVERY	32.0.0044	41359	P3567	Rollback Fails on FILEDC/STRDC
RECOVERY	32.0.0045*	41383	P3491	Row of Ordered Data Set Locked
RECOVERY	32.0.0048*	41796	P3667	Corrupt Audit Stopper
RECOVERY	32.0.0049*	41795	P3668	Invalid Unit Number
RECOVERY	32.0.0050*	41811	P3669	Option USE DUP
RECOVERY	32.0.0053*	42319	P3757	SEG ARRAY Error
RECOVERY	32.0.0055*	42714	P3820	Split Index Random Tables
REMODELIB	32.0.0005	38114	D3085	Return Address of Last User Tr
REMODELIB	32.0.0006	38198	P2881	Eliminate Unused Requestcase V
REMODELIB	32.0.0007	38747	P2864	Check for Unassigned Transacti
REMODELIB	32.0.0008	38742	D3230	Elimination of Response From P
REMODELIB	32.0.0009	39981	D3337	Data Base Stack
REMODELIB	32.0.0012	41801	P3707	Use Port Files
REORG	32.0.0001	35471	D3073	REORGANIZATION Acceleration
REORG	32.0.0005	37320	D3120	DASDL/REORGANIZATION Enhanceme
REORG	32.0.0006	37540	D3084	Simplification of REORGANIZATO
REORG	32.0.0008	38009	D3082	Implicit GENERATE Statements
REORG	32.0.0009*	38004	P2880	Eliminate Attribute Error FAMI
REORG	32.0.0010*	38203	P2888	Invalid Block on Standard Data
REORG	32.0.0011	38011	D3170	Shared ACCESSROUTINES, Data Ba
REORG	32.0.0012*	38754	P2954	Bad BCW for Ordered Data Sets
REORG	32.0.0013*	37723	P3071	READLOCKNOPURGE Removed
REORG	32.0.0014	39386	P3093	Reorganization of Embedded Ord
REORG	32.0.0015*	39620	P3145	Corruption of Compact Data Set
REORG	32.0.0016*	39903	P3190	Specifying COPY
REORG	32.0.0017*	39907	P3191	Reorganization of Standard Dat
REORG	32.0.0018	39981	D3337	Data Base Stack
REORG	32.0.0020	40505	P3318	INVALID INDEX by Zero Length R
REORG	32.0.0021*	41091	P3397	IXSEQ with Multi-Coarse Table
REORG	32.0.0022	41093	P3401	CFUPDATEVERSION on Pass 1 of F
REORG	32.0.0023	40887	D3460	Preallocation of Direct Data S
REORG	32.0.0024*	41369	P3492	Adding Checksum to Compact Dat
REORG	32.0.0025*	41358	P3486	Invalid Direct Data Set
REORG	32.0.0026*	41389	P3484	CFAUDINZ Only Valid for Audite
REORG	32.0.0027*	41803	P3670	Records with Undefined Record
REORG	32.0.0028*	41804	P3671	Improper Bit Vector Generation
REORG	32.0.0029*	41807	P3672	Disk Resident Structure
RESHELPER	32.0.0001	38324	P3394	Allow Runs for Disk Pack Types
RJE	32.0.0002	37015	D3255	File Transfer
RJE	32.0.0003	37261	D3036	Print Queue Rebuild at RJE BO
RJE	32.0.0004	37260	D3037	*BACKUP Requests Run Asynchron
RJE	32.0.0005	37252	D3038	MCS Name Display Change
RJE	32.0.0011	37985	P2626	COPY Syntax Not Checking for B
RJE	32.0.0012	37984	D2972	PUT/FETCH Record Compatibility
RJE	32.0.0013	37015	D3255	File Transfer
RJE	32.0.0014	38103	D3041	WORDS Vs. CHARACTERS in File T
RJE	32.0.0015	37849	D3042	File Transfer Code Optimizatio
RJE	32.0.0016	38101	P2746	Autobackup Optimization
RJE	32.0.0017	38098	D3043	BCL Constructs Removed
RJE	32.0.0018	38074	D3080	Crunching Transferred Files
RJE	32.0.0019	38480	D3078	Blank FTS Record at End of FTS
RJE	32.0.0020	38477	P2874	Bad Queued File Transfer Count
RJE	32.0.0021	38479	P2875	TIME(1) Vs. TIME(14)
RJE	32.0.0022	38475	D3081	LOCK PROGRAM
RJE	32.0.0023	38478	D3079	CHARACTERSPERFTBLOCK
RJE	32.0.0024	38476	D3091	Host to Host LOGON Loop
RJE	32.0.0025	38100	D3234	Terminal Transfer
RJE	32.0.0026	38569	D3092	PUT/FETCH String Field Termina
RJE	32.0.0027	38567	D3098	Missing EOF
RJE	32.0.0028*	39028	P2969	COMMENCEPF Set on Wrong Print
RJE	32.0.0029	39029	D3159	File Transfer Unexpected Abort
RJE	32.0.0030	39030	D3160	Formmessage Link in REMLP File
RJE	32.0.0031	39033	D3161	Codefile Record Translation
RJE	32.0.0032*	39034	P2968	B1800 with B9247-13 Train Prin
RJE	32.0.0033	39041	D3168	ONLINE, OFFLINE By Stationname
RJE	32.0.0034	39039	D3166	Not Sending 09 Control Message
RJE	32.0.0035	39036	D3219	Records Larger than File Trans
RJE	32.0.0036	39037	D3220	Invalid Character Record Trans
RJE	32.0.0037	39290	D3247	Phone Numbers
RJE	32.0.0038	39294	P3076	RJE Protocol Version
RJE	32.0.0039	39293	P3077	RJE PUTREPLY Vs. SYCOM
RJE	32.0.0040	39291	P3078	Buffer Size Control Message 02
RJE	32.0.0041	39489	P3079	ABORT COPY Request Vs. SYCOM
RJE	32.0.0042	39491	D3286	WH Display Enhancement

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	PRI	NOTE	DESCRIPTION
RJE	32.0.0043	39494	D3287	SM Command LEVELS
RJE	32.0.0044	39493	P3126	Protocol Version Mismatch
RJE	32.0.0045	37015	D3255	File Transfer
RJE	32.0.0046	39536	P3151	SM Command
RJE	32.0.0047	39541	P3202	File Transfer Security
RJE	32.0.0048	40038	D3346	RJE Vs. DLBACKUP
RJE	32.0.0049	40044	D3392	File Starting with "?" Lost Bl
RJE	32.0.0050	40047	P3345	Extra Linkfile Updates Removal
RJE	32.0.0051*	40049	P3346	Linkfile Updates
RJE	32.0.0052	40050	D3423	DEBUG vs RAID
RJE	32.0.0053	40053	D3424	SM Commands RSC, SPO
RJE	32.0.0054	40054	D3428	Bad Device Address
RJE	32.0.0055	40055	D3429	Halt/Load Restore
RJE	32.0.0056*	40058	P3439	Lost Available Records
RJE	32.0.0057*	40059	P3440	Invalid Printer Characters
RJE	32.0.0058*	40063	P3441	Disconnect of Switched Lines a
RJE	32.0.0059	40061	D3475	Runtime Options Save Through L
RJE	32.0.0060	41160	D3476	Programdump Out of FILEX, FILE
RJE	32.0.0061	40053	D3424	SM Commands RSC, SPO
RJE	32.0.0062*	41168	P3678	PB of Files Without Summary
RJE	32.0.0063*	41169	P3682	Remove Summary File
RJE	32.0.0065*	42461	P3753	"RS0" in Printer Backup Record
RJE	32.0.0066*	42463	P3754	Abort Compatibility
RJE	32.0.0068	42465	D3639	RJE Vs. SYCOM
RJE	32.0.0069	42466	P3777	File Transfer Input Block Size
RJE	32.0.0070	42469	P3778	"##RJE" Message Removed
RJE	32.0.0071	42470	P3779	Object System in Symbolic Head
RJE	32.0.0072	42471	D3640	ODT to ODT Communication
RJE	32.0.0073	42465	D3639	RJE Vs. SYCOM
RJE	32.0.0074*	42463	P3754	Abort Compatibility
RJE	32.0.0075	42471	D3640	ODT to ODT Communication
RJE	32.0.0076*	42463	P3754	Abort Compatibility
RJE	32.0.0077	42509	P3804	Station Logoff at RJE QUIT
RJE	32.0.0078	42510	P3805	"22 Control Message NO-OP"
RJE	32.0.0079	42511	P3806	Copy Requests Rejected
RJE	32.0.0082	42469	P3778	"##RJE" Message Removed
RJE	32.0.0083	42504	P3807	Parity Error on REMLP Files
RJE	32.0.0084	42505	P3808	AUTOPRINT INVALID INDEX
RJE	32.0.0085	42506	P3809	Send Control Length Update
RJE	32.0.0086	42507	P3810	Length of "RS" Reply
RJE	32.0.0087	42508	P3811	Incorrect Backup Family
SCRMCP	32.0.0195	38261	P2812	Correct Fault Message
SCRMCP	32.0.0222	38265	P2853	INVALID OP for COMPARE BUFFER
SCRMCP	32.0.0227	38264	P2858	7A Magtape Controls
SCRMCP	32.0.0236	38262	D3072	BUFFMEM Modifier Added
SCRMCP	32.0.0250	38264	P2858	7A Magtape Controls
SCRMCP	32.0.0252	38267	P2891	Stop Repeating First Line of O
SCRMCP	32.0.0271	37674	D2970	IVR Facility
SCRMCP	32.0.0287	37674	D2970	IVR Facility
SCRMCP	32.0.0295	38490	P2922	Memory Mods for Maintenance, G
SCRMCP	32.0.0296	38491	P2923	SCR/MCP Version
SCRMCP	32.0.0305	37674	D2970	IVR Facility
SCRMCP	32.0.0310*	38272	P2957	Buffer with Address Specified
SCRMCP	32.0.0350	38274	P2794	Pack Density Not Established P
SCRMCP	32.0.0366	37674	D2970	IVR Facility
SCRMCP	32.0.0387	38280	P3005	Wrong Density for File I/O
SCRMCP	32.0.0414	38283	P3049	IVR Write Disabled Pack
SCRMCP	32.0.0445	38285	P3069	VERIFY DISKPACK (SELECT ALL RE
SCRMCP	32.0.0461	38286	D3256	IVR for 215,225 Packs
SCRMCP	32.0.0520	38291	P3123	Warning Message
SCRMCP	32.0.0637	38286	D3256	IVR for 215,225 Packs
SCRMCP	32.0.0716	38310	D3342	IVR Type 206 Packs
SCRMCP	32.0.0726	38312	D3345	IVR Type 207 Packs
SCRMCP	32.0.1006	38332	P3459	AX More Than 60 Characters for
SCTABLEGEN	32.0.0001	37702	D3054	SWAPPER Enhancements
SCTABLEGEN	32.0.0002	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0003	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0004	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0005	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0006	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0007	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0009	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0010	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0012	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0013	41710	D3550	Eliminate DL Network
SCTABLEGEN	32.0.0014	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0015	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0016	40663	D3356	New and Old ODT Messages
SCTABLEGEN	32.0.0018	42501	D3641	Compile Time Options
SCTABLEGEN	32.0.0021	42501	D3641	Compile Time Options
SCTABLEGEN	32.0.0022	42848	D3647	TD Acceleration
SORTMCP	32.0.0398*	39094	D3233	Tape Work File

B6000 SERIES MARK 32

B6000 SERIES SOFTWARE	PATCH TABLE PATCH	(PATCH CLOSING FTRS MARKED WITH '**')	PRI	NOTE	DESCRIPTION
SORTMCP	32.0.0938*	40830	P3374		Protected Disk Files Used as S
SORTMCP	32.0.1122*	42092	P3624		Large Memory Size Specified
SOURCENDL	32.0.0002	38478	D3079		CHARACTERSPERFTBLOCK
SOURCENDL	32.0.0003	38100	D3234		Terminal Transfer
SOURCENDL	32.0.0004	39299	D3613		ACIII/BDLC BTB Request
SOURCENDLII	32.0.0020	42005	D3602		Implement ASCII-APL for NSP Da
TFL	32.0.0004*	39206	P3073		Expand Text Generation Array
TFL	32.0.0005	39652	D3320		Crunch NEWSOURCE File
TFL	32.0.0006	39667	P3194		Error Message
TFL	32.0.0007*	39902	P3195		Header Displays Correct System
TFL	32.0.0008	39981	D3337		Data Base Stack
TFL	32.0.0009*	39979	D3336		Parameters Not Specified
TFL	32.0.0011*	41769	P3557		Hyphenated RESTARTDATASET Iden
TFL	32.0.0012*	41806	P3675		Infinite Loop
TFL	32.0.0013*	41818	P3674		Table Size Exceeded
TRINTERFACE	32.0.0001	38202	P2889		Increase Number of Files in Jo
TRINTERFACE	32.0.0002	39981	D3337		Data Base Stack
TRINTERFACE	32.0.0003*	40194	P3242		Correct Offset Generation
TRPROPERTY	32.0.0003	38202	P2889		Increase Number of Files in Jo
TRPROPERTY	32.0.0004	39650	D3321		Crunch TRBASE/PROPERTIES Symbo
TRPROPERTY	32.0.0005	39659	D3322		TASKVALUE Attribute
TRPROPERTY	32.0.0006	39981	D3337		Data Base Stack
TRPROPERTY	32.0.0007	40199	D3367		Simultaneous READ/WRITE Access
TRPROPERTY	32.0.0009*	41818	P3674		Table Size Exceeded
TRUTILITY	32.0.0003	38202	P2889		Increase Number of Files in Jo
TRUTILITY	32.0.0004	38747	P2864		Check for Unassigned Transacti
TRUTILITY	32.0.0005*	39378	P3104		Search Using Alpha Items as Ke
TRUTILITY	32.0.0006	39981	D3337		Data Base Stack
TRUTILITY	32.0.0008	40530	P3388		Output Entire RANGE Specificat
TRUTILITY	32.0.0009*	41392	P3715		Compilation Fails with Syntax
UDSTRCTTAB	32.0.0002	39082	D3221		SYSTEMUSER Bit Moved
UDSTRCTTAB	32.0.0003	40366	D3364		Automatic DESTNAME for CANDE S
UDSTRCTTAB	32.0.0004	40547	D3415		Non Interactive APL
UDSTRCTTAB	32.0.0005	40546	D3477		APL-Detached Workspace
USERSTRUCT	32.0.0002*	41509	P3583		Use LONG Array Masksearch
USERSTRUCT	32.0.0003	42062	D3605		New UDSTRUCTURETABLE Generatio
UTIL	32.0.0015	38037	D3052		Clear TPS Information
UTIL	32.0.0016	38138	P2955		Version Timestamp Mismatch
UTIL	32.0.0017*	38760	P2956		Print Tape Labels
UTIL	32.0.0018	38011	D3170		Shared ACCESSROUTINES, Data Ba
UTIL	32.0.0019*	38753	P2999		Failure to Reload Beyond Two D
UTIL	32.0.0020	39217	D3353		DUMPDIR Enhancements
UTIL	32.0.0021	39376	P3108		DMCONTROL Resequenced
UTIL	32.0.0023	39625	P3146		Output Header
UTIL	32.0.0024*	39628	P3192		Printing Control Information
UTIL	32.0.0026	39639	D3314		Rebuild Across File Discontinuu
UTIL	32.0.0027	39661	P3193		Lower Case in Parameter String
UTIL	32.0.0028	39981	D3337		Data Base Stack
UTIL	32.0.0029	39989	D3338		Print Statistics Option
UTIL	32.0.0030*	39991	P3243		Corrupted Dumptime Timestamp
UTIL	32.0.0031*	39985	P3244		Error Not Given for Invalid Sy
UTIL	32.0.0032	39627	P3261		No Checksum on Block Zero
UTIL	32.0.0035	40203	P3258		Normal Vs. Direct Files as Par
UTIL	32.0.0038	39217	D3353		DUMPDIR Enhancements
UTIL	32.0.0039	40523	P3343		Structure Details
UTIL	32.0.0040*	40963	P3373		INVALID INDEX in INITIALIZE
UTIL	32.0.0041	40887	D3460		Preallocation of Direct Data S
UTIL	32.0.0042	40887	D3460		Preallocation of Direct Data S
UTIL	32.0.0043	40887	D3460		Preallocation of Direct Data S
UTIL	32.0.0044	40887	D3460		Preallocation of Direct Data S
UTIL	32.0.0045*	40922	P3389		LIST, WRITE Statements
UTIL	32.0.0046	40923	P3390		LIST, WRITE of Block Zero
UTIL	32.0.0047*	40924	P3391		LIST, WRITE Fail to Print Some
UTIL	32.0.0048*	40926	P3392		Block Limits for WRITE, LIST
UTIL	32.0.0049	41244	P3402		FLUSHDB Default
UTIL	32.0.0050	41104	P3403		BUILDDUMPDIRECTORY Not Accepte
UTIL	32.0.0051*	41111	P3404		Syntax Errors
UTIL	32.0.0052	41114	P3405		DIRECTION Attribute Error
UTIL	32.0.0053*	41358	P3486		Invalid Direct Data Set
UTIL	32.0.0054*	40915	P3493		COPY ONTO May Not Update EOF P
UTIL	32.0.0055*	41389	P3484		CFAUDINZ Only Valid for Audite
UTIL	32.0.0056*	41382	P3495		Workers Restartable Only Once
UTIL	32.0.0057	41772	D3571		Tape SERIALNO Specification
UTIL	32.0.0058	41791	P3586		Validate Block Range for LIST,
UTIL	32.0.0059	41789	P3702		Display Nonfatal Errors, Warni
UTIL	32.0.0060*	41790	P3616		Handling of Hyphens
UTIL	32.0.0062	41797	P3704		Deadlock
UTIL	32.0.0063	41802	P3705		Multiple Dumpworker Error
UTIL	32.0.0065*	41112	P3673		Multiple Row control Files
UTIL	32.0.0066	41831	P3690		Checksum Error for Block Zero
UTIL	32.0.0067*	41390	P3714		Recover Family Index
UTILOADER	32.0.0020	40344	D3332		UTILOADER on MLIP Systems

B6000 SERIES PATCH TABLE (PATCH CLOSING FTNS MARKED WITH '**')				
SOFTWARE	PATCH	PRI	NOTE	DESCRIPTION
WFL	32.0.0003	38055	D3123	B6900 Peripheral Test Driver
WFL	32.0.0004*	37383	P2747	Correctly Compare File and Tas
WFL	32.0.0005*	37376	P2748	Job on Disk with NEWSOURCE, SY
WFL	32.0.0006*	37387	P2749	Pass Global Files to Processed
WFL	32.0.0009	38188	D3090	Compiler Info Word in Seg Zero
WFL	32.0.0011	39063	D3252	Semidependent Tasks, VISIBILIT
WFL	32.0.0016	37414	D3293	Remote Job Transfer
WFL	32.0.0018	38466	D3295	Data Base Equation Allowed
WFL	32.0.0019	38465	D3296	BCL Warning
WFL	32.0.0024*	39878	P3263	Task Passed as By Reference Pa
WFL	32.0.0025	39739	D3361	Segment Code Files
WFL	32.0.0031*	41320	D3528	Improved Handling of "\$INCLUDE
WFL	32.0.0034*	41413	P3520	Missing Comma in ON Statement
WFL	32.0.0035*	41428	D3547	LOCKED in Task Attribute Assig
WFL	32.0.0036	41710	D3550	Eliminate DL Network
WFL	32.0.0037*	41843	P3584	FAMILY Specification, "<name c
WFL	32.0.0039*	41845	P3623	Prevent WFL Fault
WFL	32.0.0040	41866	P3695	Syntax "Old" WFL Data Base, Li
XREFANALY	32.0.0002	39235	P3147	Add DATABASE as XREF Item
XREFANALY	32.0.0003	40640	P3332	Remove PORT, SIGNAL Variable T
XREFANALY	32.0.0004	41317	P3518	Correctly Identify CHARACTER A
7ABMTEST	32.0.0001	38308	P3377	Count READ and WRITE Errors

DMALGOL Implementation

The following document describes the DMALGOL language, which was implemented in previous releases and documented in the Mark 31 system release. Changes since the Mark 31 release are indicated by PCN bars in the right margin.

1. INTRODUCTION

DMSII requires a special compiler for two reasons: The compiler must be able to retrieve information from the description file and it must build stack structures which are different from those in ALGOL. DMALGOL consists of ALGOL with extensions to meet these requirements.

The DMALGOL language is an "implementation" language for the DMSII system and is not intended for general use. New features may be added at any time and existing features may be changed or deleted without notice. For this reason, users should not rely upon DMALGOL as an application programming language.

2. OVERVIEW

2.1 Compile-Time Facilities

DMALGOL compile-time facilities consist of elements which are of a general nature and have been documented in ALGOL, and elements specific to DMSII.

The basic technique used to write the ACCESSROUTINES is conditional compilation. Various pieces of code are omitted, included and parameterized based on information from the DASDL description file. This file is read by the DMALGOL compiler; the DMALGOL language contains elements used to conveniently reference its information.

NODE variables and PROPERTY definitions are used to reference "nodes" within the description file. A node is a data structure which consists of two parts: a list and a set of properties, either of which may be absent. The elements of lists are frequently other nodes, but they may be integers or other data items. The syntax in DMALGOL to reference list element I of node N is N[I]. The properties of a node are in a non-homogeneous substructure whose format is defined by PROPERTY declarations.

Example:

```

NODE N;
INTEGER T;
PROPERTY RECORDSZ=[11].[11:12];
T:=N.RECORDSZ;
```

This will extract the property RECORDSZ of the node N. The properties actually used in the ACCESSROUTINES are included from the file called DATABASE/PROPERTIES.

A special form of the 'FOR statement is used to access the members of a node's list. It uses the fact that DASDL stores the number of list items at OFFSET=0 in the list, so that the compiler knows how many elements are in the list.

Example:

```

NODE STRUCTURE, SPANSET;
'FOR EACH SPANSET OF STRUCTURE DO . . .
```

This statement goes through each member of the list belonging to the node STRUCTURE, assigning its successive list elements to the node SPANSET.

A special form of the 'INCLUDE statement is used to access the text section of the DASDL description file. This section consists of a set of source language DMALGOL constructs; these "texts" are always referenced via an appropriate property. For example, data set nodes contain a property called VERIFYSTORETEXT which checks whether a record meets all verification rules before it is stored. To access this text for node DS, the following would be written:

```
'INCLUDE DS.VERIFYSTORETEXT.
```

DMALGOL would extract the appropriate text and compile it.

2.2 Environments

A normal ALGOL program contains block initialization code to reserve space in the stack for its local variables. DMSII SIB environments, however, are not contained in a running stack and cannot, therefore, execute code to build the stack. Rather, the MCP procedure DMSOPEN sets up the stack variables based on an "image" of the SIB stack kept in the ACCESSROUTINES code file. Thus, one task of DMALGOL is to build the stack image. This is done by using the ENVIRONMENT declaration, which delimits the boundaries of each environment, in much the same way as a procedure declaration does.

Environments may not be nested more than three deep.

The ACCESSROUTINES code file is an executable program that builds the DBS and then freezes in a special way using the DMSFREEZE construct of DMALGOL. Code is generated so that the outermost environment begins execution, sets up its stack, executes its outer block and then calls the last compiled environment at the next level. Each of these inner environments in turn builds its portion of the stack, executes its outer block, and then calls the next environment. The last environment called (the first inner environment compiled) is a special environment that executes the DMSFREEZE statement which signals the MCP that the DBS has been built.

DMSII uses environments at all three levels. The outermost environment contains global variables common to the entire data base. Intermediate level environments, DBS environments, contain variables unique to each data base structure. The innermost environments, SIB environments, contain the variables which are unique to a single invocation of a structure.

It is possible to call a procedure in a different environment using the following construct:

```
<procedure name> ' <structure number>
```

<structure number> is the number declared in the ENVIRONMENT declaration containing the desired procedure.

Example:

```
ENVIRONMENT X OF 2;
BEGIN
    PROCEDURE P;
END;

ENVIRONMENT X OF 3;
BEGIN
    PROCEDURE P;
    P;
    P'2
END;
```

The call P will call the routine in its local environment; i.e., the second one. The call P'2 will call the routine in environment 2; i.e., the first one.

2.3 Reference Variables

DMALGOL provides the ability to declare procedure reference variables, file reference variables and direct file reference variables. These reference variables allow for dynamic selection of procedures, files and direct files within the DMSII system.

2.4 "ONCE ONLY" Compilation

The outermost and intermediate level environments act like procedures and may contain any declarations that a normal procedure may contain. Since the innermost environments do not have stack building code generated for them, restrictions are placed on the type of declarations that can be used. In particular, three or more dimensional arrays, task arrays, queues, libraries, etc., may not be declared in the innermost environments.

Each structure of the data base has one D3 and one D4 environment compiled for it. The same lines in DATABASE/SYMBOLIC are typically compiled many times over. For example, the procedure STORE is compiled for every data set. Conditional compile-time statements alter it to fit each successive data set; each has its own VERIFYSTORETEXT included, for example, and calls to insert it into each of its automatic sets.

Some procedures, however, do not differ in any major respect between two structures of the same type. An example is GENERATE; the main difference in its operation is the block size and end-of-file, which are easily parameterized. To save compile time and code space, such procedures are compiled only once. It is not possible to use calls on the shared procedure using the <procedure name>'<structure number> syntax, since that would change the environment as well. Rather, a PCW is constructed for each structure; for a shared procedure, the PCW in each structure will point to the same segment descriptor. This allows the procedure to be both shared by different environments, but behave as if it were local to each environment.

3. DMALGOL EXTENSIONS

3.1 Compile-Time Facilities

3.1.1 NODE Declaration

Syntax:

```

-----
|<-----, -----|
-- NODE --<identifier>-----|
| - :=<arithmetic expression 1> - |

```

Semantics:

1. An identifier declared to be a NODE is a compile-time variable similar to a NUMBER; however, it must be used in conjunction with a compile-time array created by DASDL.
2. A node variable represents an index into the DASDL array.
3. The value of a node variable (index into the DASDL array) may be changed at any time during compilation by means of a compile-time 'LET statement.
4. <arithmetic expression 1>, if used, must be a constant arithmetic expression. It represents the initial index of the node variable. By default, the initial value is zero, which is otherwise an illegal value; thus, a node must be initialized or assigned a value before it is used.
5. Normally, the node variable represents the index of a "node" in the DASDL array. A list and a set of properties is associated with a node; the list is usually a list of nodes, and the properties contain values. The node variable may be used to reference members of the list and values of the properties, as follows: If N is a node variable and P is a property identifier, then N[<i>] is the <i>th member of the list of N, and N.P is the value of the property P of N (<i> is a constant or constant expression).

3.1.2 PROPERTY Declaration

Syntax:

```

----- PROPERTY ----->
| -<type> - |
|<-----, -----|
>--<identifier>-- = --<property spec>-----|

<property spec>
-- [ --<arithmetic expression 1>-- ] ----->
|-----|
| - . [<arithmetic expression 2>:<arithmetic expression 3>] - |

```

Semantics:

1. A property identifier defines the location and format of a property value associated with a node in the DASDL array.
2. Each arithmetic expression must be a constant arithmetic expression. <arithmetic expression 1> specifies the word in the property set. <arithmetic expression 2> and <arithmetic expression 3> specify a field within the word; if they are not given, the entire word is used.
3. If a type is declared, it must be a single-precision arithmetic or Boolean; when no type is specified, REAL is assumed.
4. A property identifier is used only with a node variable, as follows:
 --<node variable>-- . --<property identifier>--|

This construct represents the value of the property for the given node. It may be used wherever constants of the specified type may appear.

5. The "node property" construct is only used to retrieve a value from the DASDL array; it may not be used to change a property value (in fact, no construct can change the DASDL array: it is read-only).

3.1.3 'FOR Statement Extensions

Syntax:

```
-- 'FOR  --- EACH  -----<nodeid 1>--- OF ---<nodeid 2>----->
      |         |         | |
      | - ALL - | | -<number id>- |
      |         |         |
>-----> DO -----<ct statement>-----|
      |         |         |
      | - WHERE ---<boolean expression>- | | -<begin clause>- |
```

<begin clause>

```
-- 'BEGIN ---<text 1>-- 'NEXT ---<text 2>-- 'PRIOR ---<text 3>--->
>- 'END ----->
```

Semantics:

1. This statement provides for iterative compilation of ALGOL source code. One iteration is made over the source code for each entry in the list belonging to <nodeid 2>.
2. <nodeid 2> must be the index of a node in the DASDL array.
3. For each iteration, the control variable is assigned the index of the list element (if <nodeid 1> is specified) or the contents of the list element (if <number id> is specified).
4. If the WHERE clause is used, a constant <boolean expression> must be specified. Before the statement following DO is processed but after the control variable has assumed its next value, the <boolean expression> is evaluated. If the expression is TRUE, the statement following DO is processed; otherwise, the statement is ignored.
5. If the body of the 'FOR statement is the 'BEGIN-'NEXT-'PRIOR-'END form, a "telescoping" form of iteration is performed. In that case, the 'NEXT behaves like an 'END, so that the <text 1> is processed for each iteration. After the last iteration, <text 2> is processed just once. Finally, the iteration is repeated, backwards, for <text 3>. If a WHERE clause is used, the backwards iteration processes exactly the same cases as the first iteration; i.e., the <boolean expression> is not re-evaluated while backing out of the telescope.
6. <ct statement> is a compile-time statement including any DMALGOL extensions.

3.1.4 'INCLUDE Statement Extensions

Syntax:

```
-- 'INCLUDE ---<nodeid>-- . ---<property id>--|
```

Semantics:

1. This statement causes the compiler to process text directly out of the DASDL array.
2. The value of the specified property is assumed to be an index of text in the DASDL array. That text must be terminated by a pound sign (#) and at least one null character (4"00").
3. The compiler does not expand defines in data base declarations where the define identifier was the result of a compile-time INCLUDE statement.

3.1.5 'LET Statement Extensions

Syntax:

```
-- 'LET ---<number variable>-- := ---<procedure name>--|
```


Semantics:

This statement stores a copy of the procedure's PCW in the number variable. This construct is intended for use with compile once only procedures. Refer to the section on "ONCE ONLY" Compilation for additional details.

3.1.6 'PRINT, 'ERROR, 'DISPLAY Statements**Syntax:**

```

----- 'PRINT -----|
| - 'ERROR  ---| | | <-----, -----| |
| - 'DISPLAY -| | | -----<nodeid>-----| |
|               | | | -----<string>-----| |
|               | | | -----<arithmetic expression>-----|

```

Semantics

1. This statement causes one or more lines to be printed on the compilation listing.
 2. Each line may contain up to 87 characters, starting in position 18.
 3. If the 'ERROR form is used, the compiler's error count is incremented by one, and the printed line is bracketed by the ">" and "<" characters, as for a normal syntax error message.
 4. The lines are created from the information given in the statement, as follows:
 - a. If a <nodeid> is specified, the alpha name property for the node is inserted into the line.
 - b. If a string is specified, it is inserted into the line.
 - c. If an <arithmetic expression> is used, it must be a constant arithmetic expression. It is assumed to be an integer, and the value of the integer, zero suppressed, is inserted into the string.
- No blanks are inserted into the line between specified items.
5. If a 'PRINT or 'ERROR statement is processed (not skipped) by the compile-time processor, the lines are printed whether or not any listing \$ card options are set (LIST, LISTOMIT, etc.).

528

3.1.7 PROCEDURE Declaration Extensions**Syntax:****Form 1:**

```
-- PROCEDURE --<procedure id 1>-- := --<procedure id 2>--|
```

Form 2:

```

--<procedure type>-- PROCEDURE --<procedure heading>-- ; ----->
>- EXTERNAL --<number variable>-----|

```

Semantics:

Two forms of procedure declaration are used for compile once only procedures. Refer to the section on "ONCE ONLY" Compilation for additional details.

3.2 ENVIRONMENT Declaration**Syntax:**

629

1. The ENVIRONMENT declaration is used to declare the contents of the Data Base Stack (DBS) and Set Information Block (SIB) stacks for a data base.
- 2.=The entire "program" for a data base is an environment containing all data base globals.
- 3.=Following the data base globals, one or more DBS environments may be declared. One DBS environment is present for each data set or set in the data base.
4. Within each DBS environment, and after other DBS items, one or more SIB environments must be declared.
5. The identifier specified in the ENVIRONMENT declaration is primarily for documentation; it is not used again. <arithmetic expression 1> is a constant used to aid structure allocation within the ACCESSROUTINES. <arithmetic expression 2> (a constant, normally a node variable) is the environment identification. Every DBS environment must have a distinct environment identification. Each SIB environment must have an environment id equal to that of the DBS which contains it. The outer level environment is not declared with an environment id.
6. In a DBS environment, only global items and DBS items declared in that DBS are accessible. In a SIB environment, all global items, all items in the corresponding DBS environment, and all SIB items are accessible. In addition, all procedures declared in prior SIB environments are accessible. Those procedures are referenced via dynamic identifiers, using the environment id after the apostrophe.
7. For the innermost (SIB) environments, the <block head> must be followed by "END"; i.e., the SIB environments may not have any executable statements in their outer block.

[illegible]

Abstract

```
--<procedure reference id 1>-- := ----->
>---<procedure id>-----|
|   -<procedure reference id 2>-|
```

Semantics:

Form 1 of the declaration declares the procedure reference variable <procedure reference id>. The initial value of this variable refers to <procedure id>. The parameter description for <procedure reference id> is derived from <procedure id>.

Form 2 of the declaration declares the procedure reference variable <procedure reference id 1>, specifies the parameter description for the reference variable and specifies a body of code to be executed if <procedure reference id 1> is used before an assignment is done to it.

The procedure reference assignment statement causes the <procedure reference id 1> to refer to the procedure specified on the right side of the "=". Both procedures must be of the same type and have the same parameter descriptions.

A <procedure reference id> may be used to invoke a procedure as if the procedure were invoked directly.

3.3.2 File References**Declaration Syntax:**

```

      |<----- , -----|
----- FILE REFERENCE ---<file reference id>--- ; --|
      | - DIRECT - |

```

Assignment Syntax:

```

--<file reference id 1>-- := ---<file id>----- ; --|
      | -<file reference id 2>- |

```

Semantics:

1. Prior to being assigned a value, a procedure reference variable does not point to a file.
2. In the file reference assignment statement, the left and right file identifiers must both be DIRECT or must both be non-DIRECT.

3.4 "ONCE ONLY" Compilation

Compile once only procedures may be declared in two ways.

3.4.1 Form 1**Syntax:**

```

<procedure declaration>
-- PROCEDURE --<procedure id 1>-- := --<procedure id 2>--|

```

Example:

```

      BOOLEAN PROCEDURE GETADDRESS (A);
            VALUE A;
            REAL A;
            BEGIN
            .
            .
            .
            END;

      PROCEDURE GETADDR := GETADDRESS;

```

Semantics:

This is the most common type of compile once only procedure. <procedure id 1> and <procedure id 2> are identical. The parameters and procedure type of the two procedures must be exactly the same. The body of the procedure is compiled only once. Subsequent references to the procedure merely copy the PCW for <procedure id 2> into the environment that contains <procedure id 1>.

This technique can only be used for procedures declared in the outer block of a DBS or SIB environment. Embedded procedures must use the alternate form of compile once only.

3.4.1 Form 2

Syntax:

```
<let statement>
-- 'LET --<number variable>-- := --<procedure name>--|

<procedure declaration>
--<procedure type>-- PROCEDURE --<procedure heading>----->
>- ; -- EXTERNAL --<number variable>-----|
```

Example:

```
NUMBER GETADDRESSPCW;
BOOLEAN PROCEDURE GETADDRESS (A);
  VALUE A;
  REAL A;
  BEGIN
    .
    .
    .
  END;

'LET GETADDRESSPCW := GETADDRESS;

BOOLEAN PROCEDURE GETADDR (A);
  VALUE A;
  REAL A;
  EXTERNAL GETADDRESSPCW;
```

Semantics:

This type of compile once only procedure must be used very carefully. The user must ensure that the procedure declarations are identical. DMALGOL does not check the procedure declarations for consistency; it simply copies the PCW for the original procedure into the environment containing the EXTERNAL procedure declaration.

3.5 DMIO File Attribute

The DMIO direct file attribute is used by the ACCESSROUTINES to indicate that the DMSII system is using the file.

3.6 DMALGOL Functions and Statements

3.6.1 ALLOW and DISALLOW Statements

Syntax:

```
---- ALLOW -----|
| - DISALLOW - |
```

Semantics:

The DISALLOW statement disables external interrupts. The ALLOW statement enables external interrupts. These statements are intended for use in the ACCESSROUTINES and are used in critical sections where a series of operations must be performed without interruption.

3.6.2 ATTACHDBS Statement

Syntax:

```
-- ATTACHDBS -- ( --<arithmetic expression>-- ) --|
```

Semantics:

<arithmetic expression> must be a constant expression which identifies a D3 environment to which the currently active D4 environment will be linked up. The Mark Stack Control Word (MSCW) at the base of the current D4 environment is linked to the MSCW at the base of the specified D3 environment. The effect of this operation is not immediate since the processor's D3 register is not affected by the relinking done by ATTACHDBS. Calling a procedure declared at lex level 3 is sufficient to force the necessary display register update.

3.6.3 DMINQ Functions

The DM INQUIRY interface permits direct communication with the DMSII ACCESSROUTINES. It is intended only as a tool for use in implementing special purpose packages, such as on-line inquiry, and is specifically not intended for general user interface to DMSII.

It is assumed that exactly one data base is invoked in the normal manner in the program which uses these constructs; otherwise, a syntax error will result.

I. DMINQ [<arithmetic expression>] (<array row>)

The <arithmetic expression> is the SIB index for the desired structure. The <array row> is a one-dimensional array used to communicate with the ACCESSROUTINES. The contents of the array control the function performed by the system.

```
A [0] =      identifies desired procedure
            1 = pathfinder (find key only)
            10 = set to beginning
            11 = data finder (find/lock next/current)
            12 = getdata
            13 = DMS read (access data portion only)
            14 = get link
            16 = store current
            17 = free current
            18 = set or check date-timestamp for TPS
            19 = clear TPS flag and date-timestamp
            20 = return last transaction address
            21 = get status of abort
            22 = rerun finished-reset TPS abort pending
            23 = create
            24 = delete current
            25 = get statistics information
            26 = visible DBS message
            27 = return displayed messages
            30 = return standard data set DATAEOFs
```

A [i], i>0, depend upon A[0] as follows:

1. Pathfinder (A[0] = 1)

```
A[1] = FIND op :
```

```
-1 = current record
0 = next in set
1 = next in set = UKA
2 = next in set > UKA
3 = next in set >= UKA
4 = link in set
```

```
A [2] = SZ parameter to pathfinder
```

This parameter specifies the size of the key passed to pathfinder in the user key area. If SZ is less than zero (0), then ABS(SZ) is the number of hex characters in the user key; otherwise, SZ is the number of bytes in the user key.

A [3] = SZ2 parameter to pathfinder

This parameter specifies the size of the major portion of the user key which must exactly match the retrieved key. If SZ2 is less than zero (0), then ABS(SZ2) is the number of hex characters in the user key; otherwise, SZ2 is the number of bytes in the user key.

As an example, if an index set had a concatenated key (A,B,C) with each portion two bytes long, to retrieve the next key where A=UKA and B=UKA requires SZ=4 and SZ2=4. To retrieve the next key where A=UKA and B>UKA requires SZ=4 and SZ2=2.

If a record is found as specified, the AA word is returned in A [1]. A getdata call must be used to move the record to the user's work area.

2. Set to beginning (A[0] = 10)

A [i], i>0, not used.

3. Datafinder (A[0] = 11)

A[1] = FIND op:

0 = find current data set
1 = lock current data set
2 = find next data set
3 = lock next data set

If a record is found, its AA word is returned in A[1], and the record is moved to the user's work area.

4. Getdata (A[0] = 12)

A[1] = AA word of desired record. Moves the desired record to the user's work area.

5. DMSREAD (A[0] = 13)

A[1] = AA word of desired record.

This function is similar to the Getdata call, except that only the data portion of the desired record is moved to the user's work area; structures embedded in the accessed record are unaffected.

6. Get link (A[0] = 14)

"Fetchkey" call on control manager:

A[1] = Unused
A[2] = LLOC parameter
A[3] = LLEN parameter

The link entry is returned in the array A[*].

7. Store current (A[0] = 16)

A[i], i>0, not used.

8. Free current (A[0] = 17)

A[i], i>0, not used.

9. Set or Check Timestamp (A[0] = 18)

Used to set or check the value of the Timestamp (TIME(6)) given to the ACCESSROUTINES when the TPS Journal library is initiated. A[1] contains the Timestamp to be given to the ACCESSROUTINES. If the existing Timestamp in the ACCESSROUTINES is not zero and a call on this DMINQ function is made (passing a new Timestamp), this DMINQ function returns a value of true and the new Timestamp is not captured by the ACCESSROUTINES.

10. Clear TPS Flag (A[0] = 19)

Signals the ACCESSROUTINES to set the TPSCLOSEDFLAG true and reset the Timestamp, which was given to the ACCESSROUTINES upon initiation of a TPS

Journal library, to zero.

11. Return Transaction Address (A[0] = 20)

The Transaction Address in the Database Control File is returned in words A[0], A[1] and A[2]. This is the transaction address of the

A[0] : File number of the address
A[1] : Block number of the address
A[2] : Offset number of the address

12. Status of Abort (A[0] = 21)

The DMINQ function returns a value to TPS (in A[0]) which indicates the status of an Abort. The possible values returned are: 1 : ACCESSROUTINES is waiting for updaters to leave transaction state. 2 : All updaters are gone - waiting for an Abort 3 : Abort finished - waiting for TPS to reprocess transactions 4 : No Abort or reprocessing of transactions is necessary

13. Rerun Finished (A[0] = 22)

This function tells the ACCESSROUTINES that TPS has finished reprocessing transactions. The ACCESSROUTINES resets the TPSABORTPENDING flag to false.

15. Create (A[0] = 23)

A[1] = unused.
A[2] = if variable format then
record type, else unused.

15. Delete current (A[0] = 24)

A[i], i>0, not used.

16. Get statistics information (A[0] = 25).

Word 0 - =25
Word 1 - <structure number>
Word 2 - <type statistics>

Word 0 specifies a statistics request to the DMINQ interface.

Word 1 indicates the structure number for which statistics are desired or equals 0 if global statistics are desired.

Word 2 specifies whether static or dynamic data is desired. A value of zero indicates static statistics are desired; a value of one indicates dynamic statistics are desired.

The ACCESSROUTINES will process the request and will return the result in the array starting at word 3. Words 0 thru 2 will be unaffected by the ACCESSROUTINES. The format of the result is as follows:

Word 3 - statistics result word
Word 4 - total number of words returned
Word 5 - index to header word for sub-group 1
Word[Word 5] - header word for sub-group 1
Word[Word 5 + 1] - first data word for sub-group 1

...

Word[Word 5 + n] - header word for for sub-group 2

...

Word[Word 4 - 1] - end of statistics flag

Word 3 indicates the result of the statistics request. If the request was correctly formatted and honored, this word will be zero. Otherwise, [0:1] will be equal to 1 and [35:8] will contain an error category. The currently defined error categories are as follows:

- 1 - the structure number provided in word 1 did not correspond to an existing data set or set in the data base.
- 2 - the statistics request type provided in word 2 was invalid (not 0 or 1).
- 3 - a fault was encountered while retrieving statistics.

Word 4 contains the total number of words returned in the array including all of the fixed words at the front of the array.

Note:

If the array provided is too small to receive all of the statistics, it will be resized.

Word 5 contains the index of the first group of statistics information returned. Each group is preceded by a group header word indicating the type of the group and the number of words of information in the group.

The layout of this header word is as follows:

[47:24] - not used
 [23:8] - group type
 [15:16] - number of words in group (including header)

The last group of statistics is followed by an end-of-statistics header which has a group type of zero (0).

The various group types and their layouts are as follows:

I. Global Static Statistics (word 1 = 0, word 2 = 0)

Group type 6

Word	-	Contents
1		Time data base opened (TIME(7) value)
2		Maximum valid structure number in data base
3		Data base options
		[0:1] = 1 if statistics set in data base
		[1:1] = 1 if data base is audited
		[2:1] = 1 if lockstatistics set in data base

Group type 5

Word	-	Contents
1-n		data base name including usercode prefix, if any. data base name is followed by 4"00".

II. Global Dynamic Statistics (word 1 = 0, word 2 = 1)

Group type 1

Word	-	Contents
1		Current data base open count
2		Current number of users that have data base open for update
3		Current data base open state
		0 = data base not open
		1 = data base is opened temporary
		2 = data base is opened initialize
		3 = data base is opened normally
		4 = data base is undergoing recovery
4		Maximum total buffer space in words
5		Current total buffer space in words
6		Current allowedcore value
7		Maximum number of buffers allocated

Group Type 2 (present only if STATISTICS is set)

Word	-	Contents
1		Number of forced data base overlays
2		Number of normal data base overlays
3		Time statistics collections started or was last reset (TIME(7) value)

Group Type 3 (present only if STATISTICS is set and data base is audited)

Word	Contents
1	First audit file number
2	Current audit file number
3	Starting audit block serial number
4	Current audit block serial number
5	Average number of words used in audit blocks
6	Actual audit block size
7	Number of audit I/O's initiated
8	Total wait time accumulated on primary audit in ticks
9	Total wait time accumulated on secondary audit in ticks
10	Total transaction count
11	Total number of times processes where held up at BEGIN-TRANSACTION
12	Total time spent waiting at BEGIN-TRANSACTION in ticks
13	Total number of sync points taken
14	Total number of control points taken
15	Total time spent taking control points in ticks
16	Sum of the number of buffers present at each control point
17	Sum of the number of buffers flushed at each control point

III. Structure Static Statistics (word 1 = <structure number>,
word 2 = 0)

Group Type 7

Word	Contents
1	Structure number
2-4	Structure name (first byte is length in binary)
5	Structure type 2 = data set 5 = index set
6	Structure subtype (see PROPERTIES 10068000-10094000)
7	Structure nesting level (1 = disjoint)
8	Structure block factor (in records for data sets, in key entries for index sets)
9	Structure physical block size in words (including integrity checking words)
10	Structure area size in sectors
11	= 1 if structure is checksummed
12	= 1 if structure is address-checked

IV. Structure Dynamic Statistics (word 1 = <structure number>,
word 2 = 1)

Group Type 8

Word	Contents
1	Current number of random access users
2	Current number of serial access users
3	Current number of buffers allocated for structure
4	Current number of big buffers allocated for structure

Group Type 11 (present only if STATISTICS is set)

Word	Contents
1	Number of physical reads against structure
2	Number of physical writes against structure
3	Number of ticks spent waiting for writes to complete
4	Number of ticks spent waiting for reads to complete
5	Total amount of I/O time accumulated on file
6	Number of read-aheads issued against structure
7	Number of write-aheads issued against structure

Group Type 9 (present only if STATISTICS is set and structure is
----- a data set)

Word	-	Contents
1		Number of finds against data set
2		Number of Create/Store's against data set
3		Number of Modify/Store's against data set
4		Number of Deletes against data set
5		Number of times control information changed

Group Type 10 (present only if STATISTICS is set and structure is
----- an index set)

Word	-	Contents
1		Number of Finds against index set
2		Number of inserts in index set
3		Number of key data changes in index set
4		Number of key deletions from index

Notes:

"ticks" refers to ticks of the processor clock at 2.4
microseconds/tick.

Groups may be returned in the array in any order.

17. Visible DBS message (A[0] = 26)

Starting at A[1] is the Visible DBS message to be processed. The message must be terminated by 4"00".

The message is passed to the Visible DBS message processing routine and any response message are returned starting at A[1]. If the response contains multiple lines 4"0D" is used to separate the multiple lines. The last line is terminated by 4"00".

18. Return All Displayable Messages (A[0] = 27)

Returns the most recent displayable messages (does not include messages generated by the Visible DBS) in the array A, starting at the first character of word 0. If no messages exist, then seven nulls (48"00") are returned starting at the first character of word 0. The most recent 23 messages are returned in the order most recent to least recent. Each message is terminated by 48"0D" and the final message of the group is terminated by a null (48"00").

19. Collect standard data set DATAEOF's (A[0] = 30)

A[1] = Number of words in array

Starting at A[2] are pairs of words. One pair per structure.

A[N]
 .[47:12] Structure number
 .[35:16] Partition number
 (= 0 if not partitioned)

A[N+1] The value of DATAEOF is returned in this word.

UTILITY uses this DMINQ function to determine the DATAEOF values for standard data sets when performing an on-line dump. An entry is made in the array for each structure being dumped. The ACCESSROUTINES will return the DATAEOF value for the structure as it was two control points ago. UTILITY will then only check the checksums for those blocks of data that have an address less than or equal to DATAEOF.

II. Three "intrinsic" arrays

Note that these arrays may be used only in array reference assignment statements. They may not be used as ordinary arrays.

1. DMKEYAREA

Hex array; the user's key area for the one SIB invoked.

2. DMWORKAREA [X]

Hex array; the user's work area for a particular structure.

X is an arithmetic expression, value = SIB index for that structure (see below).

3. DMSIBDESC

Real array; the SIB description for the SIB invoked. Used primarily to determine the SIB index for each invoked structure. The first N words of this array (0 to N-1) contain the structure number (plus other stuff) for each invoked structure. The index in the SIB description of this word is the value of the SIB index for that structure. N is equal to SIZE(DMSIBDESC)-8.

3.6.4 DMTRANSLOCK Statement

Syntax:

```
-- DMTRANSLOCK -- ( --<formal array 1>-- , ----->
>--<formal array 2>-- ) -----|
```

Semantics:

DMTRANSLOCK is intended only for use in compilation of ACCESSROUTINES. It performs transaction locking for DMS jobs where <formal array 1> refers to a transaction lock and <formal array 2> is its new value.

3.6.5 DMSCAUSE Statement

Syntax:

```
-- DMSCAUSE -- ( --<arithmetic expression>-- ) --|
```

Semantics:

The DMSCAUSE statement calls the MCP procedure DMSCAUSE and passes a single real-valued parameter. The effect of the call is dependent on the parameter, as follows:

parameter < 0	Indicates that the calling program has left transaction state. The program is delinked from the transaction state linkage chain.
parameter = 0	Indicates that a syncpoint has been completed. All programs waiting for a syncpoint for this data base are awakened.
parameter > 0	Indicates that a record for which other users are waiting has been freed. The parameter is the stack number of the previous owner. All programs waiting on that stack number will be awakened.

3.6.6 DMSFREE Statement

Syntax:

```
-- DMSFREE --|
```

Semantics:

This statement calls the MCP procedure DMSFREE, which causes all records locked by this process to be freed in every data base visible to the process.

3.6.7 DMSUPDATEDISKHEADER Statement

Syntax:

```
-- DMSUPDATEDISKHEADER -- ( --<file designator>-- ) --|
```

Semantics:

This statement causes the disk header for the designated file to be flushed to the directory.

3.6.8 DMSWAIT Function

Syntax:

```
-- DMSWAIT -- ( --<arithmetic exp. 1>-- , ----->
>-<arithmetic exp. 2>-- , --<arithmetic exp. 3>-- , ----->
>-<array identifier>-- ) -----|
```

Semantics:

The DMSWAIT function is a Boolean-valued function with four parameters. The DMSWAIT function calls the DMSWAIT procedure in the MCP. The first three parameters are real-valued and the last is an array. The effect of the DMSWAIT function is dependent on the values of parameters 2 and 3.

- a. Parameter 2 = 0, parameter 3 = 0

Indicates that the calling program needs to wait for a syncpoint to complete on this data base. The program is linked into the sleeper chain for the data base and is suspended. When a syncpoint is complete on this data base (indicated by some other program doing a DMSCAUSE(0)), this program will be awakened and a result of FALSE returned. If a deadlock is detected, this program is awakened and returned a result of BOOLEAN(1). If the program has specified a wait limit and the limit expires before the syncpoint occurs, the program is awakened and returned a value of BOOLEAN(3).

Parameter 1 is a control word obtained from the location specified by parameter 3 in the array given as parameter 4. If the value of the control word from the array changes before suspending the program, a value of FALSE is immediately returned.

- b. Parameter 2 = 0, parameter 3 = -1

Indicates that the calling program has entered transaction state. The program is linked into the transaction state linkage for the appropriate data base. If at the time of this call the program is already in transaction state, a result of TRUE is returned; otherwise, a result of FALSE is returned.

- c. Parameter 2 = -1, parameter 3 = -1

If the number of processes waiting for locked records in this data base is less than the first parameter, a result of TRUE is returned; otherwise, a result of FALSE is returned.

- d. Parameter 2 > 0, parameter 3 > 0

Indicates that the calling program needs to wait for a locked record. Parameter 2 is the stack number of the current owner. The calling program is linked into the sleeper chain for the current owner of the record, the sleep count for this data base is incremented by one, and the program is suspended. When the current owner of the record frees it (indicated by doing a DMSCAUSE (<owner's stack number>)), this program is awakened and a result of FALSE returned. If a deadlock is detected, a result of BOOLEAN(1) is returned. If the program has specified a wait limit and the limit expires before the record is freed, the program is awakened and returned a value of BOOLEAN(3).

Parameter 1 is a control word obtained from the location specified by parameter 3 in the array given as parameter 4. If the value of the control word from the array changes before suspending the program, a value of FALSE is immediately returned.

3.6.9 DSED Function

Syntax:

```
-- DSED --|
```

Semantics:

The DSED function is a parameterless function which returns a Boolean result of TRUE if the program is DSed, and FALSE otherwise.

3.6.10 DSWAIT and DSWAITANDRESET Functions

DSWAIT and DSWAITANDRESET constructs are identical to WAIT and WAITANDRESET, except that a result of -1 indicates the job was DSed while waiting. Note that these constructs are intended for DMSII ACCESSROUTINES only.

3.6.11 NEWDOPEVECTOR Function

Syntax:

```
-- NEWDOPEVECTOR -- ( --<array identifier>-- , ----->
>-<arithmetic exp>-- ) -----|
```

Semantics:

The NEWDOPEVECTOR function is a Boolean function with two parameters. The first parameter is a 2-dimensional array designator and the second is the new size for the first dimension. The new size may be larger or smaller than the current size. If the new size is smaller, rows will be deallocated. If the new size is larger, unallocated rows will be added. Care must be taken when using this function, since no copy descriptors will be fixed up.

3.6.12 SIBOFFSET Function

Syntax:

```
-- SIBOFFSET -- ( --<procedure name>-- ) --|
```

Semantics:

The SIBOFFSET function is a compile-time function that accepts a procedure name as its only parameter. The result of this function is the offset of this procedure in its environment.

3.6.13 SNR Function

Syntax:

```
-- SNR --|
```

Semantics:

The SNR function is a parameterless function which returns as a real value the current value of the processor's stack number register.

3.6.14 DMSFREEZE function

Syntax:

```
-- DMSFREEZE --|
```

Semantics:

The DMSFREEZE function calls the MCP procedure DMSFREEZE. The DMSFREEZE function is invoked to indicate to the MCP when a data base stack has been built and users may be attached. The DMSFREEZE function is a boolean value function which returns FALSE if the freeze is successful or returns a TRUE in the low order bit and an exception type in [19:16] if the freeze fails.

The exception types possible are:

- 1 The environment calling DMSFREEZE does not have an SCW within it.
- 2 There are no stacks waiting to attach to the data base.
- 3 The caller is already an active frozen data base.
- 4 The caller is not a data base stack initiated by DMSOPEN.

4. DMALGOL Reserved Words

The reserved word type is indicated following each word. Refer to the ALGOL Language Reference Manual (Form No. 5001639) for details.

ALLOW (2)
ATTACHDBS (2)
DMINQ (2)
DMKEYAREA (2)
DMWORKAREA (2)
DMSIBDESC (2)
DISALLOW (2)
DSWAIT (2)
DSWAITANDRESET (2)
DMIO (2)
DMSCAUSE (2)
DMSFREE (2)
DMTRANSLOCK (2)
DMSWAIT (2)
DMSUPDATEDISKHEADER (2)
DSED (2)
ENVIRONMENT (2)
NEWDOPEVECTOR (2)
NODE (2)
PROPERTY (1)
SIBOFFSET (2)
SNR (2)

Documentation Evaluation Form

Title: B 6000 Series System Notes

Form No: 5011570
Date: November 1980

Burroughs Corporation is interested in receiving your comments and suggestions regarding this manual. Comments will be utilized in ensuing revisions to improve this manual.

Please check type of Suggestion:

□ Addition

☐ **Deletion**

□ Revision

□ Error

Comments:[illegible]

From:

Name _____

Title	Author	Year	Journal	Volume	Issue	Page
1. The Effect of Temperature on the Rate of Reaction of Hydrogen Peroxide with Potassium Iodide	John Doe	2023	Journal of Chemical Education	90	5	1234
2. Synthesis and Characterization of a New Polymeric Material	Jane Smith	2022	Polymer Letters	10	3	5678
3. Kinetic Study of the Reaction Between Sulfur Dioxide and Nitric Oxide	Michael Brown	2021	Chemical Kinetics and Catalysis	15	2	9101
4. Investigation of the Mechanism of the Reaction of Ethanol with Hydrogen Bromide	Sarah White	2020	Journal of Organic Chemistry	85	1	2345
5. Study of the Effect of pH on the Solubility of Certain Salts	David Green	2019	Journal of Analytical Chemistry	70	4	6789
6. Determination of the Rate Constant for the Reaction of Carbon Monoxide with Hydrogen	Emily Black	2018	Physical Chemistry	42	1	1011
7. Analysis of the Reaction Pathway for the Oxidation of Ethanol	Robert Grey	2017	Journal of Physical Chemistry	121	3	1212
8. Investigation of the Effect of Temperature on the Rate of Reaction of Hydrogen Peroxide with Potassium Iodide	John Doe	2023	Journal of Chemical Education	90	5	1234
9. Synthesis and Characterization of a New Polymeric Material	Jane Smith	2022	Polymer Letters	10	3	5678
10. Kinetic Study of the Reaction Between Sulfur Dioxide and Nitric Oxide	Michael Brown	2021	Chemical Kinetics and Catalysis	15	2	9101
11. Investigation of the Mechanism of the Reaction of Ethanol with Hydrogen Bromide	Sarah White	2020	Journal of Organic Chemistry	85	1	2345
12. Study of the Effect of pH on the Solubility of Certain Salts	David Green	2019	Journal of Analytical Chemistry	70	4	6789
13. Determination of the Rate Constant for the Reaction of Carbon Monoxide with Hydrogen	Emily Black	2018	Physical Chemistry	42	1	1011
14. Analysis of the Reaction Pathway for the Oxidation of Ethanol	Robert Grey	2017	Journal of Physical Chemistry	121	3	1212

Company

Address _____

Phone Number _____ Date _____

Remove form and mail to:

**Burroughs Corporation
Documentation Dept., TIO - West
P.O. Box 4040
El Monte, CA 91734**