By Mark Robillard

The Terminal Wrap-Up

And Now, The Complete Terminal System Unveiled

In January, I described the basic operations of a color graphics terminal and outlined the principles of raster scan imaging and character drawing on a CRT. That article was the beginning of a four month miniseries on the design of portable terminal systems. This month's column focuses on the complete terminal systems design.

Reviewing the basics of portable systems design, I arrive at a list of features that are desirable in a portable system.

The first thing that comes to mind is battery operation. This is not always possible, and when you are dealing with a CRT terminal or a similar display device, it becomes almost impossible. So, the first item on my wish list for a portable system, battery operation, is thrown out the window.

This month, the portable terminal that will be described requires a +5V and −12V power supply. This supply does not require that it be large enough to power an entire room of electronics. As it is, the terminal contains only nine integrated circuits.

The second item on my wish list for portable systems is true portability (that is, small relative size). The terminal control electronics provided here measures 4½ × 6 inches. Another item is programmability or the ability to adapt to many situations. This particular terminal is probably more programmable than you will need 90 percent of the time.

The February "Techniques" column introduced a revolutionary integrated circuit called the National Semiconductor NS455 TMP. This product forms the basis of my portable terminal card design. As you recall from that article, it is hard to find a smaller, more portable implementation with all of the features incorporated into this chip. In fact, here you see the bare minimum required to implement a complete intelligent terminal using the NS455.

Last month, the portable system I've been talking about reached out and touched the world through telecommunications. More specifically, it utilizes single chip modem technology to communicate over telephone lines. This month I'll attempt to connect both the TMP and the National Semiconductor MM74HC943 modem chip together into an intelligent telecommunications terminal.

First, let's review terminals in general.

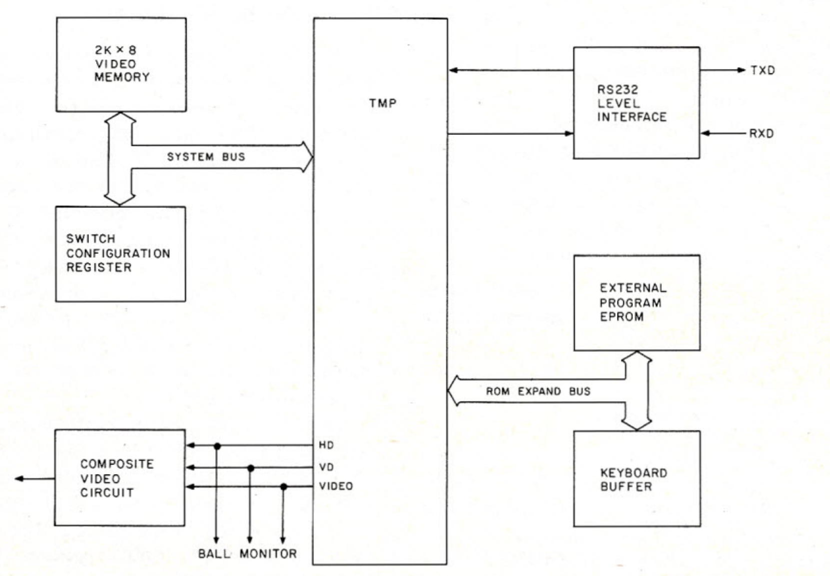


Fig. 1. Block diagram of the TMP Demo board.

Terminal Characteristics

As you may know, there are two basic types of interactive terminals: alphanumeric and graphic. The alphanumeric terminal of today typically displays 24 to 25 horizontal lines of between 80 and 132 characters per line. This data displayed on the screen is allowed to scroll up or move up one line at a time as more information is entered.

Some of these alphanumeric terminals have internal character buffers that allow you to reverse scroll so that none of the information is lost should it go by too fast. The characters that make up the 80 × 24 display are usually composed of a matrix of dots 5 × 7 or 7 × 9 in dimension. The 7 × 9 variety usually gives the best resolution for upper- and lowercase characters.

The alphanumeric terminals of the early to late '70s provided a white character on a black background. Some of the newer systems, such as the Apple Lisa and others, are now displaying a white background with a black character that looks more like a page of text.

From the early alphanumeric terminals came the ability to draw pictures with blocks formed within the character cell. The first graphics characters on personal computers, shown in the old Model I TRS-80, were six blocks divided up as

Mark Robillard is a Principal Engineer/Program Manager for Sanders/Calcomp in Hudson, NH. He can be contacted at MJR Digital, PO Box 630, Townsend, MA 01469.

two vertical lines of three. Through software or command sequences you could selectively turn on or off any combination of the six blocks. This became known as low-resolution graphics.

Soon after, a form of high-resolution graphics, as in the Apple and TRS-80 color microcomputers, became available. Although high-resolution is a misnomer for these relatively crude resolution machines, they were a distinct improvement over the Model I's block graphics.

Pixel graphics next became the buzz word. Pixels (as you will recall from January) are the small elements that make up the dots for the characters. The Apple computer has the ability to turn on or off any combination of pixels, thereby making smooth or fine-line graphics. The name "line graphics" came from this type of technique. Now there are character generator ROMs that provide preprogrammed line graphics characters as well as block graphics characters.

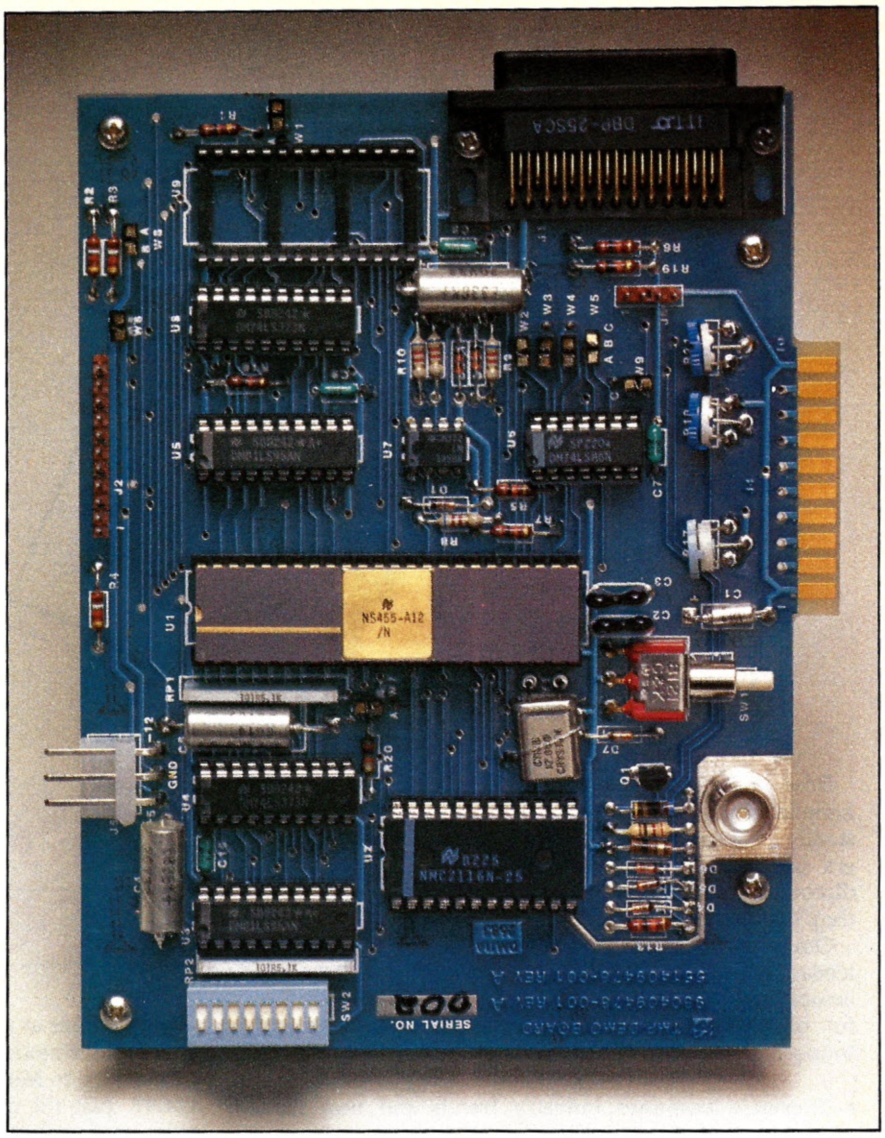
Graphics terminals require a host of control sequences or control codes to allow an external computer to implement the graphics pictures on the screen, just as the earlier alphanumeric terminals required control codes to move the cursor or erase lines. These control codes are now known as escape sequences.

An escape sequence is named for the first character that must be sent to the terminal. In the standard ASCII character codes, escape is known as a control code that has a value of decimal 27. A normal escape sequence showing the decimal 27 coming over alerts the terminal to a control operation coming down, and then a single ASCII character arriving denotes the type of control. From there, any number of ASCII characters may be sent to complete the variables or the argument needed for that particular control. Graphics terminals use many forms of escape sequences to address and turn on or off pixels or blocks. They also use escape sequences to erase the screen or to provide various color sets.

"Oh, Sure . . ."

As you can see from this brief discussion of the characteristics of alphanumeric and graphics terminals, you have to remember and deal with a lot. When you decide to build an alphanumeric or graphics terminal, either one that suits all your needs or one that provides the capabilities of the lower cost, preassembled, pretested systems, you'll quickly find it more cost effective to purchase a unit.

I'm sure many of you out there are saying, "Oh sure, maybe you can afford a $600 Lear Sigler Adam ADM alphanumeric, but on my salary I can't." That's a good point, and I'm not advocating running out and buying a complete terminal. In fact, this is one of the rare times I recommend that you buy a product rather than build it.



The TMP Demo-12 board.

I come from the era when building was the rule; buying was for those less technical. In the case of the NS455 TMP, however, the cost in single quantities of the chip itself and the lack of availability (because it is a brand new product) dictates that it's easier and more cost effective to purchase a board. "What board?" you say. "Is there a board available with the TMP on it?"

You bet! There are two varieties of this board. One is called the TMP Demo-12, the other the TMP Demo-18. The numbers relate to the frequency of the crystal that is driving the chip. They also relate to the resolution on the screen. This month I'll talk about the TMP-12 board. The photo is a picture of the complete terminal electronics with the TMP board.

As you can see, there are only nine integrated circuits. Actually, in the picture there are only eight. The last integrated circuit plugs into the socket shown on the upper half of the board.

Let's go over some of the hardware features and what's basically on this board. Fig. 1 is a block diagram of the TMP Demo-12 board. In the upper right corner, the RS-232C driver and receiver circuits are built into the card. A DB25 delta-type modem connector is also built right into the card. All interfacing to this connector, in fact all modem control signals, are provided by the chip; they exit and enter via this connector. It requires nothing other than a standard modem cable to talk to it. This is your link to whatever host you're communicating with.

On the upper left of the block diagram you see the 2Kb × 8Kb video memory. This provides the screen buffer of 25 lines of 80 characters. Each character occupies one byte of video memory. Also, on the system bus, notice the switch configuration register. This particular demo board has many possible configurations to use. I'll get into the operation of the switch register later.

Moving down to the lower left you see the video output. There are two choices on the TMP board. If you have a Ball Brothers monitor that accepts separate

TTL video signals (that is, horizontal drive, vertical drive and video), the board is designed to use that monitor directly. You'll be able to connect it up to the three outputs provided on the card.

If you decide to use a composite video monitor, much like those now available for personal computers, a circuit is provided on the card to allow it to combine the horizontal, vertical and video signals to provide an industry standard composite video output. In fact, the BNC connector, a standard connection device in video, is provided right on the card.

Moving to the right of the block diagram, you see the keyboard buffer. This is basically an eight-bit buffer chip, an 81LS95, that allows eight-bit encoded keyboard data to enter into the TMP. This will require an external encoded keyboard which is not hard to come by. At the present time the surplus market is loaded with them. Realize, however, that this is not an unencoded keyboard matrix. It requires some circuitry on the keyboard.

Above the buffer you see a box or an optional box entitled External Program ROM. If you recall from the February article, it's possible to program the TMP yourself. The TMP demo board allows you to do this. If you don't want to use the program that's factory-wired into the TMP demo chip, you may plug in your own operating or control program. That's where you use the empty socket on the board that I mentioned earlier.

Big Deal

Now, let's get into the features. So, big deal—it's 4½ × 6 inches and it has all this hardware! What can it do? If you're familiar with the Lear Sigler ADM3A data terminal, it may be well to note that this board not only does everything that terminal does but is much less expensive.

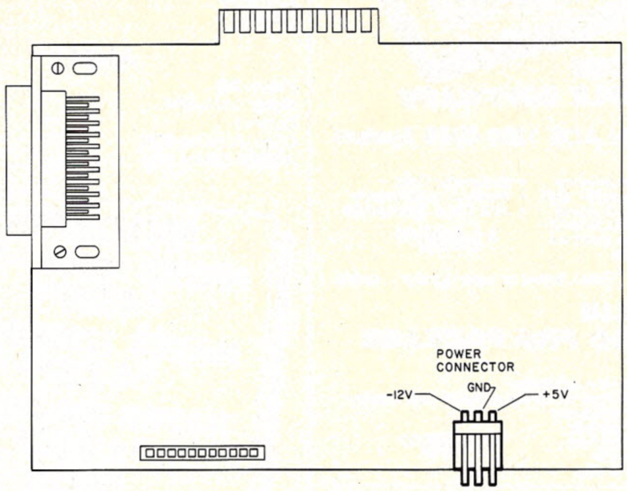


Fig. 2. Connection diagram and physical placement of the dc power source connector.

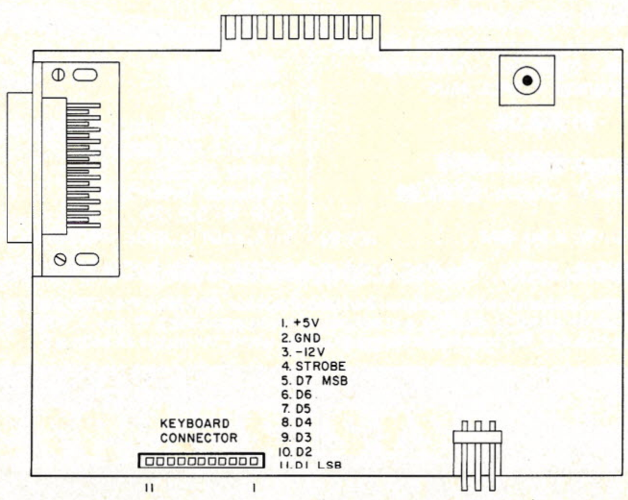


Fig. 3. Keyboard connector and pin out.

An internally masked ROM contains a host of control functions as well as the basic alphanumeric command set.

You can use the external EPROM if you want to design your own terminal program. It displays a set of characters in an 80 column by 25 row display. The 25th row is a status row. This row will show the status on-line, off-line and in various other functions. There are 12 or 18 MHz versions with corresponding 5 × 7 or 7 × 9 character fonts.

There are two different boards—the Demo-12 board or the Demo-18 board. The monitor outputs are Ball monitor or composite video. The board has the ability to work at 50 or 60 Hz.

For you European readers, this may be a welcome sight. It is jumper selectable and will automatically lock to whichever line frequency you select, therefore eliminating any swim or interference on your CRT screen. The RS-232C serial interface with full modem control signals allows you, through the switch register, to run full-duplex communications from between 110K to 19.2K bits per second (bps). There are 24 escape sequences to control the positioning of the cursor and various functions within the terminal, and 15 control sequences.

Now that you have an idea of what the board can do, what the board's size is, what's basically on it and what it looks like, let me tell you that the price is $195. That's not from me; that's from National Semiconductor and full details on ordering this board are given at the end of this article. Is it worth that kind of money? If you haven't made up your mind yet, let's go into detail on the use and operation of the board.

TMP Demo Interfacing

Let's start by connecting up the board as you would in building a small terminal system. The first place to start is the power connector. Fig. 2 is a description of the pins necessary and the voltages required for the board. As I promised, the +5V supply is a mere ½ A and the −12V supply is a very minimal 25 mA. The reason for the −12 supply is to provide the RS-232C voltage levels. As you can see from the diagram, a simple three-pin Amp or Molex slide-on connector is used. You can purchase the mating end of this

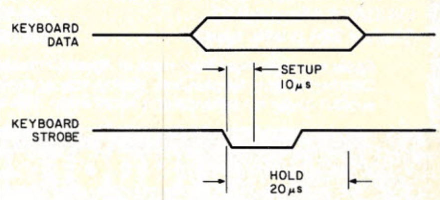


Fig. 4. Timing relationships between the keyboard data inputs and the keyboard strobe line.

connector at any Radio Shack store, and because of the current requirements, small guage wire can be used to bring the power supply signals to the card. In fact, it should be relatively simple to design a power supply for the use of the board.

Now that the power is connected, you have to hook up some way of talking to the board. The next thing should be the keyboard. Fig. 3 is a diagram showing the keyboard connector and its pin designations.

The keyboard input is expecting a seven-bit parallel ASCII code from an encoded keyboard. The row of wire-wrap pins across the edge of the card are centered on 0.1 inch increments. This corresponds to a standard Berg-type connector that can be obtained from many electronics supply houses. Along with the seven bits of parallel data, a strobe must be provided.

Also on this connector, as you can see from the figure, is the +5 and −12V power. This is not an input, but an output to power your keyboard, realizing, of course, that the 500 mA 5V requirement only applies to the TMP demo board. If your keyboard requires an amp or so, you must provide it in the TMP power supply. However, the connector allows for current to flow into the keyboard.

Let's talk a little bit about the strobe. Fig. 4 outlines the timing necessary for the keyboard input. Characters are strobed into the TMP board on each logic low level of the strobe line. The strobe duration may be of any length as long as it has a minimum of 1.25 microseconds. The TMP's internal program will automatically check to see that the strobe line has returned to its logic high state before accepting any new characters.

However, one operational aspect of the board must be noted. You cannot have a long strobe signal without having the autorepeat function. What is autorepeat? You may notice in terminals or even on your personal computer that if you hold down any key more than approximately a half-second, it will automatically start repeating that particular character on the screen. The TMP control program will look for three quarters of a second before it determines that the autorepeat function should be enabled.

When autorepeat is enabled, it will then send out that same character code through the serial port and up to the display at 15 cps. Looking back at the timing diagram, you will notice that the data must be available to be read from the seven-bit parallel lines within ten microseconds after the falling strobe edge, and it must hold at least 20 microseconds after the falling strobe. Most commercially available and even surplus keyboards hold to this spec.

The keyboard data comes in through the buffer IC that I talked earlier and is read into the TMP. On the board, that particular buffer chip, U5, is in a

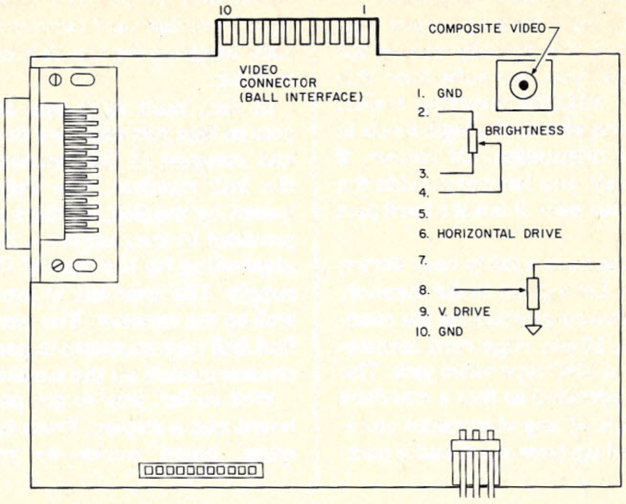


Fig. 5. Video monitor connections.

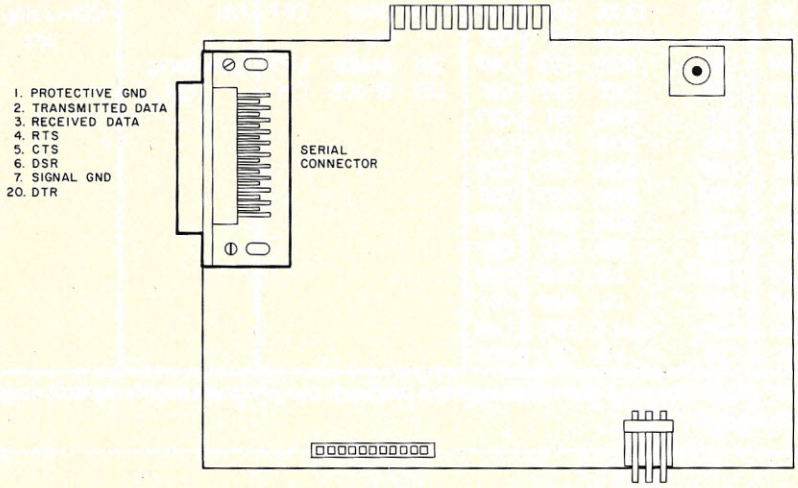


Fig. 6. Serial terminal connections.

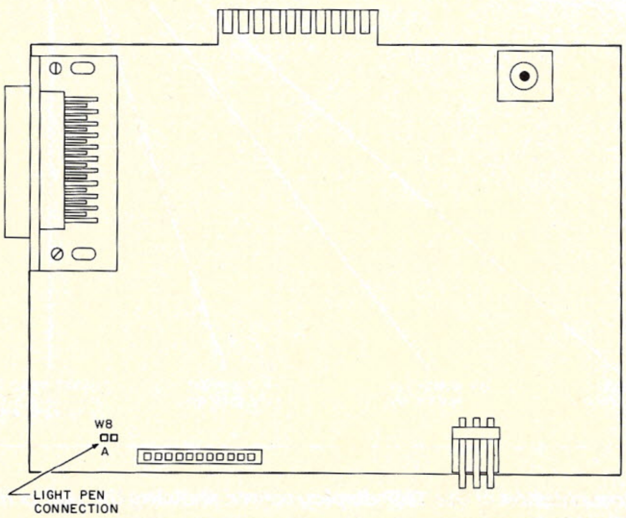


Fig. 7. Light pen input particulars.

socket. The reason for this is that the one provided is a noninverting buffer.

If you have an inverting output keyboard, where a one actually comes out as a logic low, you can substitute this part with an 81LS96 inverting buffer, which will then switch the logic levels to their correct orientation. Of course, if you go this way, you have to provide the 81LS96 on your own. It is not a hard part to come by.

So far, it has been a fairly easy device to hook up to. Let's go on to the monitor.

There are two connectors for the monitor. One is a 10-pin edge card connector, the other a BNC-type video jack. The video jack is provided so that a standard cable, available at any electronics store, can be hooked up between it and a composite video monitor. Fig. 5 shows the details and the pin outs of both connectors. The edge card connector is specifically designed for the Ball Brothers-type monitor.

In fact, built right into the card are pots so that you can vary the brightness and contrast of the monitor signals to the Ball monitor. The only signal required by the Ball Brothers monitor not provided by the card is a + 12V or + 15V (depending on the size of the monitor) supply. The user has to provide this as well as the monitor. You can frequently find Ball type monitors in packages or as chassis models on the surplus market.

Well so far, you've got power, a keyboard and a display. From here, the terminal board works—as you type in things, characters come up on the screen. Of course, in the local mode or the mode where the keyboard only talks to the display, you can entertain yourself with typing messages that you alone can read. The function of a terminal is to talk to some remote host or other control-like system. That's the function of the serial port built into the TMP board.

Fig. 6 is a diagram showing the outputs and inputs available on the serial interface. As you can see from the list, all signals necessary to hook up to a standard modem or any other type terminal device are provided. This is not necessarily the case on many inexpensive terminal cards. I have seen cards where only the transmit and receive lines are available and no way of handshaking between two systems is possible.

In most cases, this would never work at high speeds where data is being sent from one computer to the next. The TMP board, however, has been designed with problems such as this in mind, and as you can see, they've been solved.

8	7		6	5	4		3	2	1
OFF	OFF	4800 bps	OFF	OFF	OFF	Odd	OFF	OFF = Parity	OFF = 1 stop
OFF	OFF	19.2K	OFF	ON		Parity		OFF	bit
OFF	OFF	19.2K	ON	OFF	OFF	Even	OFF	ON	ON = 2 stop
OFF	OFF	19.2K	ON	ON		Parity			bits
OFF	ON	9600	OFF	OFF	ON	MARK	OFF	ON = Parity	
OFF	ON	7200	OFF	ON	ON	SPACE	ON	ON	
OFF	ON	4800	ON	OFF					
OFF	ON	3600	ON	ON					
ON	OFF	2400	OFF	OFF					
ON	OFF	1800	OFF	ON					
ON	OFF	1200	ON	OFF					
ON	OFF	600	ON	ON					
ON	ON	300	OFF	OFF					
ON	ON	150	OFF	ON					
ON	ON	134.5	ON	OFF					
ON	ON	110	ON	ON					

Fig. 8. Chart showing the relationships between the programmable switch configuration pack.

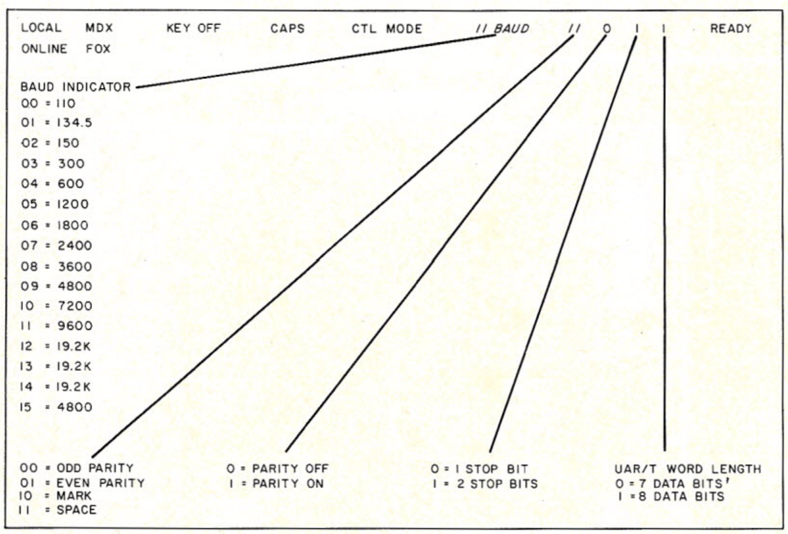


Fig. 9. A representation of the TMP display screen showing the status line and the meaning of its display.

Of Interfacing and Light Pens

On interfacing, last but certainly not least, there is an input to this card that I think will surprise you. I'm not sure if the ADM3A data terminal has this capability, but the TMP Demo board sure does. The TMP chip allows the possibility of being connected to an external light pen. In case you aren't familiar with light pens, let me explain what their functions are.

When using an intelligent terminal, or a terminal that provides graphics or prompting operator-type displays, it is usually necessary for the system to allow the operator to select various options from the screen.

In the case of a graphics terminal, it may be necessary for the operator to manipulate objects on the screen. Using a standard ASCII keyboard to do this type of function becomes very cumbersome, and soon the operator finds it so

CTL G	—	BELL
CTL H	—	Backspace
CTL I	—	Horizontal TAB
CTL J	—	Line Feed
CTL K	—	Cursor UP (ADM3A)
CTL L	—	Cursor RIGHT (ADM3A)
CTL M	—	Carriage Return
CTL N	—	Cursor UP
CTL O	—	Cursor DOWN
CTL P	—	Cursor LEFT
CTL Q	—	Cursor RIGHT
CTL R	—	Cursor HOME
CTL S	—	UAR/T BREAK function
CTL T	—	Increment bit rate pointer
CTL Z	—	Clear Screen (ADM3A)

Fig. 10. List of control codes available in the TMP system.

tedious, he may not perform the operation.

Several years ago the light pen was invented. A light pen looks very much like a ball-point pen, although somewhat larger in diameter, and is connected by a cable to the electronics that control the terminal. In its tip is a photo-sensitive device.

In earlier years, this was a standard photocell or light-dependent resistor. Now it is usually a silicon photo-transistor. If you recall my January discussion of a raster terminal, each dot is refreshed on the screen as it scans by from left to right. Your eye thinks the dot is on the screen constantly because of the refresh rate. However, the light pen will be able to detect, through its photo-sensor, these peaks of light output as it is strobed across the screen.

Realize, of course, that there will always be some light output because of the phosphor dots implanted on the glass faceplate. However, through analog circuitry and the use of comparators, the sensitivity of the photo-transistor can be adjusted to react only to peak pulses of light.

Internally, the TMP chip has the capability, when the light pen input (a TTL input) is supplied, to latch up the current row and column pixel address that it is scanning. This is no easy feat. By the time the light impulse a) gets through the analog circuitry in the light pen, b) becomes a digital TTL level signal and c) is entered into the TMP, several pixels of scan have gone by. However, the TMP has the ability to adjust for this error in placement.

On the TMP board a small wirewrap pin (denoted as W8) allows an external TTL input from a light pen to be applied to the TMP. The light pen is fully supported in the software that is burned into the control program of the TMP chip. The diagram in Fig. 7 shows the necessary connection

Finally, you have the TMP board connected. It's got power, a keyboard and monitor; it's hooked up to your favorite serial host and maybe even has a light pen for detecting areas on the screen. Now, what can it do, what can you make it do and what can it possibly supply in an application program?

<ESC> A — Toggle Auto Line Feed
<ESC> B — Switch Register Status Line
<ESC> C — Control Mode On
<ESC> D — Toggle ONLINE/LOCAL Status
<ESC> E — Toggle Full Duplex/Half Duplex Status
<ESC> F — Control Mode Off
<ESC> G V — Load Attribute Latch 1
<ESC> H — Restore Attribute Latch 1
<ESC> I — Display Status Line
<ESC> J — No Status Line
<ESC> K — Keyboard enable
<ESC> L — Light Pen
<ESC> M X Y
or —Move Cursor
<ESC> = X Y
<ESC> O — Keyboard disable
<ESC> P — Toggle cursor
<ESC> Q — Run Self Test
<ESC> R — Block send current row
<ESC> S — Block send entire screen
<ESC> T — Erase to the end of line
<ESC> V — Toggle the CAPS LOCK
<ESC> W — Erase the Switch Register Status
<ESC> X — Toggle the UAR/T Word Length
<ESC> Y — Erase to the end of page

Fig. 11. Escape sequences available for the TMP Demo board.

Attribute Bits ON if bit 0

7	6	5	4	3	2	1	0
Block Graphic	Blank	Underline	Double Width	Double Height	Blink	Half Intensity	Reverse Video

Fig. 12. Representation of the attribute byte showing each bit position and its function.

TMP Software Operation

When you hook up the TMP board and decide to use it, the first thing is to configure the jumpers on the board and the selectable switch register. I promised to go over this later; it's now later.

Eight programmable jumpers are on the card, not including the jumper which basically grounds out the light pen signal or allows for an external light pen. The first one (which I will call W7) will allow two different horizontal sweep rates and frame rates for the board. This basically selects an operation between 50 and 60 Hz.

There is an eight-position switch register located on the card. Fig. 8 shows the various conditions of each switch and the functions that it performs.

As you can see in the figure, the first switch programs the serial port number of stop bits. The second switch provides either parity or no parity and, if there is parity enabled, switches three and four determine whether it is of the even, odd, mark or space variety. The remaining four switches allow the selection of one of the fourteen available bit rates.

After selecting the switch combinations that match the serial host you are communicating with, powering up the card will yield a blinking block cursor in the upper left-hand corner and a system status line on the bottom of the screen. Fig. 9 shows a representation of that status line and what each message means for the terminal.

As you can see, whether the terminal is in the local or on-line condition, its mode currently is displayed. Whether it is in the all caps mode or the control mode, the bit rate is indicated by a two digit number. Then the parity and stop bits, and UAR/T word length, which are all selectable, are shown. Of course, the right-hand corner is the ready indication.

Because this status line occupies one of the rows, the data field available is limited to 24 rows. However, through programming, as you will see in the escape sequences, it is possible to replace this 25th line with a data row to give you the full 25 rows.

From here it works basically like any other ASCII terminal. The control and escape codes that allow for special functions can either be entered from the serial port or locally from the keyboard.

As I mentioned earlier, the escape sequence operation is a two or more character transmission with the decimal 27 escape character being the first character sent. The second character must be a capitalized alphabet character corresponding to the function desired. From here, Fig. 10 shows the control codes possible. Fig. 11 lists possible escape codes.

Looking down the list of control codes, many are similar to all other ASCII terminals. The bell code, or control G, allows for a beeper signal that may be

There is no speaker or beeper on this card—that's the only function that National decided not to implement.

60 Hz

0 0 1 1 1 0 0

0 0 1 1 1 0 0

0 0 1 1 1 0 0

2 2 3 3 3 4 4

2 2 3 3 3 4 4

2 2 3 3 3 4 4

5 5 6 6 6 5 5

5 5 6 6 6 5 5

5 5 6 6 6 5 5

5 5 6 6 6 5 5

50 Hz

0 0 1 1 1 0 0

0 0 1 1 1 0 0

0 0 1 1 1 0 0

2 2 3 3 3 4 4

2 2 3 3 3 4 4

2 2 3 3 3 4 4

5 5 6 6 6 5 5

5 5 6 6 6 5 5

5 5 6 6 6 5 5

5 5 6 6 6 5 5

5 5 6 6 6 5 5

5 5 6 6 6 5 5

Fig. 13. A representation of the block graphics character available on the TMP and its bit assignments.

activated from a host or through the keyboard.

You will note, however, that there is no speaker or beeper on this card. That is the only function that National decided not to implement. However, you can latch the bell signal coming out on one of the connectors which would then be applied to an external beeper circuit. From backspace right down through cursor home, these functions should be fairly familiar to you. The UAR/T Break function will make the serial output line space or give a low condition TTL-wise, which usually means a special code in the host.

Looking at the escape sequences available (Fig. 11), let's go down through each command and find the real power behind this card. The first code, Escape A, allows you to toggle on or off an autolinefeed function after a carriage return. This is normally a dip switch configurable type of operation in a terminal.

As you can see, it is all done through the keyboard or from a host with the TMP board. Escape B will display the switch register status in the status line. Escape C will turn the control mode on. The control codes sent into the board are not acted upon as described in Fig. 10, but are processed like normal alphanumeric characters and are displayed as reverse video, half-intensity characters. This sequence erases the switch register status if it was displayed.

Escape D toggles the on-line or off-line status. This also erases the switch register status if it was displayed. Escape E switches between full-duplex and half-duplex communications. If you are unclear on the difference between these two, look back to last month's discussion of various computer-to-computer communications modes. Escape F turns the control mode off where normal control code processing, as shown in Fig. 10, would be provided.

Escape G will load the attribute latch 1 (AL1) with the eight-bit character following the G. This attribute latch is normally used by the status line to do its reverse video, but for this mode the status line is switched over to attribute latch 0 (AL0) thus freeing AL1 for normal text attribute operation.

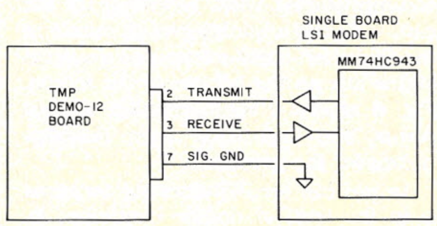


Fig. 14. Connection diagram between March "Techniques" single chip modem circuit and the TMP Demo board.

The operations of each of the bits in this attribute latch are shown in Fig. 12. As you can see, depending upon which bit is on, the following characters then utilize the attributes chosen. If bit 0 is on, any characters following the Escape G and the bit 0 will be in reverse video. Likewise, you can select between half-intensity, blinking, double height, double width, underlined, blanked or block graphic.

The block graphic brings up a good point. How are the blocks mapped in the TMP operation?

Block That Graphic

Earlier I covered the two rows of six blocks in the old Model I TRS-80. It's not quite that easy in the TMP. It is a rather strange arrangement (shown in Fig. 13), which provides the bits or blocks for the graphics characters. A data byte bit location, being a one, corresponding in this block will turn on the areas denoted by its data bit position as shown in the figure. After some study you should be able to see the complex graphics possibilities.

Going back to the chart in Fig. 11, Escape H control sequence then restores the attribute latch to its initial power-up condition. Escape I displays the status line on the 25th row; therefore, only 24 data rows are allowed. The screen is cleared and the cursor is homed to the upper left-hand corner.

Performing an Escape J won't display the status line and will give you 25 data rows. This command also clears the screen and homes the cursor. Escape K enables the TMP board to accept data from the keyboard.

As I stated earlier, the light pen is fully supported in the software on the TMP. The Escape L code returns to the host the light pen registers: horizontal pen and vertical pen, denoted as HPEN and VPEN. These are adjusted binary values and are sent out the RS-232C port with the horizontal value first.

Escape M, with an x and y variable, moves the cursor to that specified x,y position. It's a four-bit character sequence, x is the column position 20–6F, y is the row position 20–37 relative to the home position, which is the upper left-hand corner.

You will note that 20 in hexadecimal is an offset in the number ranges. As an example, the fourth column on the third row from the upper left-hand corner would be 23 hex, 22 hex. This same sequence or command is used in the ADM3A terminals.

Escape 0 is a keyboard disable which doesn't accept data from the keyboard; however, escape sequences are accepted. Escape P toggles between various cursor presentations. There are four available. First is the blinking block cursor that comes up on power-on; from there you can move to a solid underline cursor; a blinking underline cursor; and finally to solid block nonblinking cursor.

I suppose if you're a really picky terminal operator, this might come in very handy. Escape Q runs a self-test diagnostic on the board and then performs a software reset. Escape R sends out to the RS-232C port the current row of data pointed to on the screen by the cursor. An Escape S sends out the entire screen's worth of data. This can be very handy when saving the screen to a printer device or possibly to some sort of an off-line storage system.

Escape T will erase the screen or the characters from the screen beginning at the cursor position to the end of the current line. Escape V will toggle a software caps lock function. If you do not have the caps lock function on your keyboard, this will provide you with one.

Escape W erases the switch register status from the status line. Escape X is the selector (you were probably wondering where it was) for the UAR/T word length, between seven and eight bits. Finally, Escape Y erases the data on the screen from the current cursor position to the end of the page. As you can see from the Escape and Control codes available on the TMP board, the possibilities are many.

Where do we go from here? Well, following the sequence of articles from January through March it seems only right, now that the terminal is operational, to touch the world of telecommunications. It is relatively simple to hook up the circuit shown in last month's "Techniques" column to the TMP board.

The connections necessary are outlined in Fig. 14. I have decided not to repeat the schematic of the card; however, I think you get the idea of the possibilities.

Close-Out

I must confess that when I originally planned to write an article on a design using the TMP, I had in mind a proprietary design that I would supply to you. After working with the folks at National and actually getting my hands on a TMP card, I was convinced that it is more effective to purchase the final item—assembled, tested and guaranteed to work.

Let me also explain that when you buy the board, it comes with not only applications information and a more detailed description of what I've given you in this article, but also a multipage, commented source listing of the control program burned into the TMP.

This is something that you don't normally get with a standard ASCII terminal. This commented source listing will help you in understanding how the TMP works and how to provide your own version of the TMP control program in the EPROM socket available.

You must realize that the EPROM socket can take any size EPROM up to 8Kb x 8Kb so you can design your own control system no matter how complex. As I said earlier, it is not the custom of this column to advocate the purchasing of simple hardware. However, in this case I feel it is justified.

Once again, my thanks go out to Bill Kofoed of National Semiconductor for bringing this product to my attention and for the generous use of the card for a two week period. If you decide to order a board from National, specify the TMP Demo-12 card. I believe the $195 price to be accurate, but when this article is published the price may have changed.

Next month, "Techniques" will take on a slightly different flavor. I won't be talking about any one item or theme. A couple of new products have been announced in the personal robotics field, which many of you know is a heartthrob of mine. I will be reviewing these and I will set the atmosphere for an introduction to the next few month's articles. Enjoy your TMP boards, those of you who buy them.□