HERE is the great happening in TVT technology you have long been waiting for. Take a small, single-sided PC card with SIX integrated circuits on it, plug it into your KIM-1 or other microcomputer and display up to several thousand upper and lower case characters of your choice, all on an ordinary TV set with minimum modifications. Despite its "all the bells and whistles" performance, the cost of this new TVT approach is so ludicrously low that there's no comparing it with anything previously available. If you're a real dyed-in-the-wool scavenger (etch your own boards, steal sockets and connectors, burn your own PROMs, etc. . . ), you can put this together for around nine dollars plus the rapidly dropping cost of a character generator IC!

Complete kits and ready-to-go units are also available commercially, at somewhat higher but still unbelievable prices.

This new TVT development uses the microcomputer to do practically all of the system timing and control involved in a video display. This reduces the remaining interface circuitry to three hex inverters, two baby PROMs, and a character generator. Your computer alternates its compute and display modes, just as your KIM-1 now alternates between computer and keyboard modes. With fancy enough software, you can make this alteration nearly or completely transparent. More simply, you let the screen go blank when the computer is busy and doesn't want to talk to you anyway.

An entire book could be written on this whole new TVT ballgame. In fact . . . the book is called Microprocessor Based Video Displays, and Sams will print it. What we'll do here is lift just enough out of the book to show you how to build a video display for your KIM-1. The particular circuit is called a TVT-6L (L for lower case) and we'll show you how to build displays of 16 lines of 32 characters, 13 lines of 64 characters, and 25 lines of 64 characters, along with a fancy full-performance cursor that includes scrolling, erase to end of screen, full motions, and the usual goodies. The larger displays will take more memory than the bare bones KIM-1 has, so we'll show you one way to go with a KIM-1 and KIM-2 (4K add-on RAM) pair of cards.

How It Works

The block diagram of the TVT-6L is shown in Fig. 1a. An area of your microcomputer's regular memory is reserved for your display. On the minimum KIM-1, a 512 character, 16 x 32 display on pages 02 and 03 is a good starting point, although the TVT-6L card can work with any contiguous memory block from 0000 to 0FFF. Since the KIM uses parts of page 00 and 01 for its operating system, these usually aren't available for alphanumeric display use. For the larger displays with added RAM, memory locations from pages four through seven or else four through ten (0400-07FF or 0400-0AFF) are a good choice.

Besides these display memory pages, you'll need a place to put the SCAN program that tricks the KIM-1 and a TV set into talking to each other compatibly. Usually your SCAN program is around ninety words long. On the KIM-1, this is easily stuffed into the leftover scratchpad RAM starting at 1780.

Our DECODE read only memory is the heart of our TVT circuit. This PROM is activated by sending it an address from 2000 to EFFF. When activated, the DECODE memory causes a companion SCAN memory to force the microcomputer into a scan mode that advances the CPU's program counter 32, 64, or some other number of selected steps, advancing once each microsecond, binary counter style.

During this active horizontal scan time of usually 32 or 64 microseconds, all the memory in the microcomputer is sequentially interrogated on a one memory slot per microsecond basis. A new upstream tap is added to the memory to be displayed that always outputs data to the TVT-6L circuit, even and particularly when the display memory does NOT have access to the data bus.

So, during a scan mode, the display memory outputs characters to the TV even though it does not have control of the data bus. The characters have the format shown in Fig. 1b, with an ASCII character using up the lower seven bits of the memory word. An optional cursor bit is placed on bit eight if wanted or needed. A zero in bit eight does nothing: a one optionally displays a winking cursor under both software and hardware control.

The lower seven bits of a character that were sent from the upstream tap go to a character generator IC1. IC1 also receives some "what row of dots do we want?" information from the DECODE
read only memory IC2. This particular character generator has an internal video shift register to directly output serial video in a 7 x 11 dot matrix with descenders format. The internal video shift register in IC1 derives its load and clock timing pulses from the KIM system clock 02 by way of gated oscillator IC6.

The serial output video goes now to a new, simple, and super-important circuit called a bandwidth enhancer in IC5. The bandwidth enhancer predistorts the output video to exactly cancel the way your TV set is going to try and mess it up. With this circuit, it’s a simple matter to output several thousand characters per frame and still stay within the ordinary video bandwidth of a plain old TV set. Our bandwidth enhancer simply makes the dots longer than the undots, with the amount of lengthening set by a CLARITY control that is tuned to your TV for the sharpest and brightest display.

Meanwhile, two other outputs from the DECODE memory IC2 go to a position delay circuit in IC4 to provide horizontal and vertical positioning. The delayed sync signals are combined with the enhanced video in video combiner IC5.

IC5 gives us two outputs. One is the usual monitor output with grounded sync tips and +2 volt white level, used with monitors and completely preconverted TV sets. Our second TV output is translated upwards to put the white level at +4 volts, the usual bias level needed to go directly into the base of the first video amplifier transistor in a portable, transformer operated, solid state, black and white TV. Thus, our TV output greatly simplifies direct video interface. As Fig. 7 will show us later on, all you do is rip off the head phone jack and use it as an automatic video changeover switch, defeat the sound trap, and that’s all you need — at least for the 16 x 32 display.

Three switches on the TVT-6L let you program the module to suit your particular needs. One switch picks 32 versus 64 character lines when used with suitable scan software. A second switch gives you a choice of no cursor or of a blinking underline cursor under software control. The final switch is the neatest one to watch since it gives you a choice of all upper case or mixed upper and lower alphanumerics.

**Scanning**

A SCAN program activates the DECODE memory once each horizontal line, which
results in serial video being output for that particular line. If the program calls for a blank or retrace scan, all zeros are output, resulting in an all-black line. After a horizontal scan, the scan program computes the character and row information needed for the next line, and continues this way, on through one composite frame of fully interlaced video.

The design of a scan program is extra tricky since you have to control the exact number of microseconds everything takes to keep both the computer and the TV set happy. But once the scan program is designed and debugged, it’s nothing but several dozen words of RAM or ROM available when needed to output the contents of the memory pages as video.

All our scan program does is cause the pages of memory reserved for characters to appear on the screen. The SCAN program has absolutely nothing to do with how the characters get onto or off of that memory, and couldn’t care less. Your ordinary
KIM-1 firmware can be used to load and dump memory to cassette tape. Your internal keypad can be used to put messages onto the screen by writing onto the memory pages. This trick gives you a zero cost ASCII keyboard and encoder, but at the hassle of having to write everything in hex rather than ASCII code.

For most uses, you'll want to add an external ASCII keyboard, entering on parallel A inputs and interrupting the Scan program every time you want to change a character. We'll be looking at a full scrolling cursor program later, but the important point now is that you use whatever ordinary KIM-1 compatible programs you like to fill and empty the display memory pages. Your separate Scan program simply puts the memory pages on the screen.

This way, you have total access to the screen memory at any time for any reason. Things like a displayed real time clock are trivial, and you can load and dump characters at a fantastic rate. With a simple Hex-to-ASCII adapter, you can also display op-code directly instead of alphanumeric characters. Note that this new TVT approach isn't DMA (Direct Memory Access) with its related drivers and access hassles. Your character memory is, looks, feels, and tastes just like any other memory in the microcomputer, since we've kept our upstream tap a secret from the CPU.

**Building It**

Fig. 2 shows you the schematic along with its parts list, while Fig. 3 gives you the truth tables for the DECODE and SCAN read only memories. Note that these are Tri-state 32 x 8 PROMs. Their programs obviously change if you use a system different from the KIM-1.

You'll find a full size printed circuit layout in Fig. 4, along with the mechanical and drilling details of Fig. 5. Components are located per the overlay of Fig. 6.

Start construction with the jumpers, using sleeving where shown. Follow this up with the nine test points and
C1  220 pF polystyrene
C2, 6, 8 270 pF polystyrene
C4  0.22 µF mylar
C3, 11, 12 0.1 µF mylar (Disks OK for C11, 12)
C5  130 pF polystyrene
C7  0.015 µF mylar
C9  0.0047 µF mylar
C10 47 pF polystyrene
C13, 14 470 pF disk
C15  0.01 µF disk
D1-9 1N4148 or equivalent silicon computer diode. D1, D9 must
be quality units.
IC1  CG5604L-1 character generator IC (STD Microsystems)
IC2  IM5600 or equivalent 32 x 8 Bipolar Tristate PROM "SCAN"
IC3  IM5600 or equivalent 32 x 8 Bipolar Tristate PROM "DECODE"
IC4  4584 CMOS Hex Schmitt Trigger (Motorola)
IC5  7405 TTL hex open collector inverter
IC6  74LS04 LS TTL hex inverter
J1, 2  PC Mount Phono Jack, Keystone 571
R1, 15, 16 1.5k, 1/4 watt carbon film resistor
R2  680 ohm,
R3, 4 1k, 
R5  220 ohm,
R6  3.3 Megohm,
R7  150 ohm,
R8  22 ohm,
R9  100 ohm,
R10, 11 22k,
R12 10k upright trimmer potentiometer CTS U201 "CLARITY"
R13 100k, "H POS"
R14 500k, "V POS"
S1-3  SPDT miniature switch 3.17 mm pin centers
MISC:  PC Board, etched and drilled per Fig. 4; Test Point
Terminals (9); PC Sockets, 24 pin (1), 16 pin (2), 14 pin (3);
Matching connector (Amphenol 225 or equivalent); Sleeving;
Jumper material; solder.
NOTE:  The following are available from PAIA Electronics, Box
14539, Oklahoma City, OK, 73114:
PC Board, etched and drilled, #TVT-6LB, $4.00
Complete kit of all above parts #TVT-6LK, $89.95
Assembled and Tested TVT-6L, #TVT-6LAT, $75.00
KIM Coded Cassette, #TVT-6LC, $5.00

Complete Parts List, TVT-6L.

---

The low profile IC sockets, the switches, and the output
phono jacks. Finally, add the resistors, pots, capacitors, and
diodes. Be sure to note the polarity of each diode as it is
added. Use fine solder and a small iron, and be sure to
carefully double-check for any splashes or missed
connections.

Software

Table 1 shows some tested and workable KIM software. Program 1-A is the
scan program for a 16 x 32 fully interlaced 512 character
display that can be moved around as needed. Program
1-B is a dual program that you can set up as 25 x 64 or
13 x 64 fully interlaced displays with larger characters.
Finally, Program 1-C gives us a four-in-one full perform-
ance scrolling cursor. 1-C accepts an external ASCII
keyboard on the parallel A inputs and works with any of
the display formats by chang-
ing the key words as shown.
This particular cursor system
includes all the bells and
whistles, such as full, rapid
cursor motions in all direc-
tions, scrolling, erase to end
of screen, and so on. It takes
up most of page 01 in the
KIM-1. You can easily make
the cursor program longer for
super fancy editing or shorter
for a minimum sequential
loading, per your choice.

Modifying your KIM

Table II gives you a com-
plete listing of all pinouts on
the TVT-6L along with the
interface connections needed
for either a KIM-1 or KIM-2
interface. The actual com-
puter mods are outlined in
Fig. 8. For the KIM-1, you
add a connector along the top
and make wiring pencil direct
connections as shown. The
foil is cut in ONE place along
the 1K memory chip select
line and a changeover switch
is added. With the switch in
the NORMAL position, the chip
select is driven from K0 as
usual. In the TVT position,
the chip select line is driven
from the TVT's CSO line,
which is a negative logic OR
of KO and the TVT's scan
access.

For larger displays, you'll
need extra memory. Fig. 8b
shows us the modifications for
a KIM-2 memory. These
mods first convert the KIM-2
decodings so that the KIM-2
works on the second, third,
fourth, and fifth "K" of
memory, or addresses
0400-13FF. Only addresses
0400 through 0FF may be
used for TVT page storage,
although the remaining space
is available for other com-
puter use. The rest of the
KIM-2 mods are similar to
those on the KIM-1.

Even if you are going to
use extra memory (who
isn't?), convert the KIM-1
anyway as it is the simplest
and best way to get started
with your video displays. The
changeover switch lets you
run with the TVT-6L out of
Closeup of TVT-6L module. Three of the six 1 ICs used are hex inverters. Switches give choice of line length, upper and lower case, cursor. Twin jacks give either monitor video or already-translated TV video. Both outputs are enhanced for minimum bandwidth needs. Module is adaptable to many popular microprocessors and microcomputers.

the socket. In one position, the computer works normally; in the other, it will work as a TVT or transparently so long as the TVT is in the socket and so long as memory locations 2000 to EFFF are not called.

TV Mods

The TV output with its +4 white bias level greatly simplifies your direct video interface. Fig. 7 shows how to “borrow” the headphone jack on a Panasonic T126A and convert it into an automatic video changeover switch. The only new parts needed are two short pieces of miniature coax. The sound trap is defeated by lifting the hot end of the 4.5 Megahertz crystal.

This type of conversion works on any small screen, solid state TRANSFORMER OPERATED portable B and W television set, so long as the set needs a bias voltage around +4 volts at the input to the first video stage.

In order to use the 64 character displays on an ordinary TV without extensive video bandwidth changes, the television’s horizontal frequency is run much lower than normal, around 11 kHz. This means that you’ll most likely need a width and hold modification for 64 or other long character lines. On the set shown, you can use a coil of 48 turns of #24 wire on a 1.27 cm diameter nylon form in series with the existing width coil. A new hold mylar capacitor of one third the normal value, or .022 µF, is added in parallel to C408 to extend the hold range downwards.

Note that the reduced horizontal frequency and reduced width are only needed on 64 character lines. The shorter 32 character lines run at normal horizontal speed. This tradeoff buys us a lot in the way of being able to scan characters with the CPU in the first place and eliminates any need for video bandwidth extension, so it is well worth the simple and reversible mods needed. Clip-on RF modulators can also be used as shown in the TV Typewriter Cookbook and Microprocessor Based Video Displays, again thanks to the reduced horizontal rate with long line lengths.

With any TV modification, be sure to have a SAMS photofact on hand and get expert help if you’ve never done a video input conversion before. NEVER ATTEMPT DIRECT VIDEO INTO A HOT CHASSIS OR TRANSFORMERLESS TELEVISION SET.

Initial Checkout

Always have a good oscilloscope on hand for your initial checkout, and always do your first check on a KIM-1 in the 16 x 32 utility scan program 1A mode. Don’t worry about doing anything initially except displaying code that already happens.

1. Add a new 36 pin, single readout connector along the top of the KIM-1 above the crystal. Small “L” brackets can be added to use existing holes.

2. Make short and direct wire connections as shown in Table II. Use a wiring pencil for all connections except +5 and GND, which should be short lengths of #18 wire.

Do not use ribbon cable or attempt extending the TVT-6L.

3. Break ONE foil run as shown, and add a DPDT changeover switch:

With the switch in the TVT position, operation is totally transparent so long as the TVT-6L is in its socket and addresses 2000-FFFF aren’t called.

Fig. 8a. Modifying your KIM-1 for the TVT-6L.

1. Add a new 36 pin, single readout connector along the left edge of the card, the side away from the regulator. Small “L” brackets can use existing holes if one of the handle eyelets are replaced with a #8 screw.

2. Make short and direct wire connections as shown in Table II. Use a wiring pencil for all connections except +5 and GND.

3. Break TWO foil leads as shown, and add a DPDT changeover switch:

Note that we now have a new input pin on Connector 5 that is driven by KIM-1 decodings K1, 2, 3, and 4 in parallel from Application connector C, D, E, and F.

We also have a new output pin on Connector 3 that provides a ground for the KIM-1 Decode Enable. This is connected to Application Connector K on an unmodified KIM-1 and to Expansion Connector 20 or a KIM-1 modified per Fig. 8a.

Note further that BD SELECTED output Connector 16 is not used. These modifications cause your KIM-2 to respond to addresses 0400-13FF. The program address switches are no longer used.

Fig. 8b. Modifying your KIM-2 for the TVT-6L.
to be on pages 02 and 03.

Center the three controls, switch to “32”, cursor OFF and Lower Case OFF, plug the TVT-6L into your KIM-1, insert IC2 and IC3 only, and apply power. Go through the usual initialization, putting the KIM-1 in its binary mode with the interrupt returning you to keyboard operation. Then, try to operate the KIM-1 with a simple program on page zero to make sure the chip select and decode enable logic on the TVT-6L is passing things through properly and transparently.

Now, check address 2000 with your keyboard monitor. It should contain an A0. If it doesn’t, stop immediately and find out why! Check the next 29 locations for more A0s followed by two 60s followed by another string of 30 A0s and so on. You should now be able to write and single step a simple program that will transfer control from KIM to TVT back to KIM again (see Example 1). Your KIM-1 should start at 0000, jump to 2000, index sixteen times by twos to 201E, return to 0003 and stick there in the trap. Don’t go on till the KIM and TVT can pass control back and forth to each other.

Next, add IC6 and check testpoints LD and CK with a decent scope. The waveforms should look exactly like Fig. 9. In particular, they should be clean and stable. The clock should have eight narrow positive clock pulses between load commands. Do not omit checking these waveforms.

Add the rest of the ICs.

---

**Example 1.**

**A. Program for a 16 line, 32 character per line Interlaced TVT-6L Raster Scan:**

<table>
<thead>
<tr>
<th>uP — 6502</th>
<th>System — KIM-1</th>
<th>Start — JMP 17A6</th>
<th>Displayed 0200-03FF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Program Space 1780-1784</td>
</tr>
</tbody>
</table>

**Upper Address (178A0)**

<table>
<thead>
<tr>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>*</th>
<th>V8</th>
<th>V4</th>
<th>V2</th>
<th>V1</th>
<th>H16</th>
<th>H8</th>
<th>H4</th>
<th>H2</th>
<th>H1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>F</td>
<td>— normal program (no tv)</td>
<td>2</td>
<td>— blank scan</td>
<td>3</td>
<td>— scan row 1</td>
<td>4</td>
<td>— scan row 2</td>
<td>(etc.)</td>
<td>5</td>
<td>— scan row 11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>E</td>
<td>— vertical sync pulse</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 1780 | CLC | 18 |
| 1781 | STA | 8d | (8A) | (17) |
| 1784 | PHA | 48 |
| 1785 | PLA | 68 |
| 1786 | BNE | 40 | 00 |
| 1788 | JSR | 20 | 00 | 20 |
| 178b | ADC | 69 | 10 |
| 178d | CMP | C9 | E0 |
| 178F | BCC | 00 | 00 |
| 1791 | TAX | AA |
| 1792 | LDA | Ad | (89) | (17) |
| 1795 | ADC | 69 | 1F |
| 1797 | STA | 8d | (89) | (17) |
| 179A | TXA | 8A |
| 179b | ADC | 69 | 40 |
| 179d | BNE | d0 | 00 |
| 179F | JSR | 20 | 04 | 20 |
| 17A2 | CMP | C9 | 24 |
| 17A4 | BCC | 00 | 00 |
| 17A6 | LDA | A5 | (EC) |
| 17A8 | ADC | 69 | 7F |
| 17AA | BCC | 00 | 00 |
| 17AC | STA | 8d | (EC) | E0 |
| 17AD | LDX | A2 | 36 |
| 17B1 | LDX | A0 | 05 |
| 17B3 | DEY | 86 |
| 17B4 | BPL | 10 | Fd* |
| 17B6 | BCC | 90 | 05* |
| 17B8 | STA | 8d | (EC) | E0 |
| 17B9 | LDX | A2 | 37 |
| 17bd | JSR | 20 | 1E | 20 |
| 17C0 | PHA | 48 |
| 17C1 | PLA | 68 |
| 17C2 | CLD | d8 |
| 17C3 | LDA | A9 | 00 |
| 17C5 | STA | 8d | (89) | (17) |
| 17C7 | STA | 8d | (89) | (17) |
| 17C8 | LDA | A9 | 22 |
| 17CA | STA | 8d | (8A) | (17) |
| 17Cd | JSR | 20 | 00 | 20 |
| 17D0 | DEX | CA |
| 17D1 | BPL | 10 |
| 17D3 | BMI | 30 |

Tape Ident — 6A
Program Length — 85 words + 1 word page zero (EC)

Clear Carry
Store Upper Address
Equalize 10 microseconds
continued

// Character Scan 0-11///
Increment Character Scan Counter
Character Scan Counter Overflow?
No, Scan next row of character
Save Upper Address
Get Lower Address
Increment Lower Address: Save carry
Restore Lower Address: Save carry
Get Upper Address
Reset Upper Address; add carry
Equalize 3 microseconds

// Blank Character Scan 12///
Is it the “17th” row of characters?
No, start a new row of characters
Get Interface Word
Change Field via Carry bit
Jump if Even Field
Odd Field V Sync; Restore Interface
Load Odd (short) # of blank scans
Equalize 31 microseconds
continued

Jump if odd field
Even Field V Sync; Restore Interface
load Even (long) # of blank scans
//1st V Blanking scan/ /
Equalize 9 Microseconds
continued

Initialize Lower Address
continued
Initialize Upper Address
continued
///Rest of V Blanking scans///
One less scan
Start Character Scan
Repeat V Blank Scan

NOTES: TVT-6L must be connected and both the SCAN and DECODE PROMs must be in circuit for program to run.
Both 17AC and 17B8 require that page 00 be enabled when page E0 is addressed. This is done automatically in the KIM-1 decode circuitry.
Location 00EC on page zero is reserved as an interlace storage bit.
Step 1788 goes to where the upper address stored in 178A and the lower address stored in 1789 tells it to. Values in these slots continuously change throughout the program.
For a 525 line system, use 17b0 34 and 17bc 35 and a KIM-1 crystal of 992.250 kHz. This is only needed for a video superposition or tilting applications; the stock 1 MHz

---

**Fig. 9.** Key high frequency waveforms. Clock “CK” must be clean, jitter free, and have narrow positive duty cycle shown.
crystal is used for ALL OTHER uses.

Normal program horizontal frequency is 15,873.015 kHz;
Vertical 60.0114. 63 microseconds per line. 264.5 lines
per field; 2 fields per frame 529 lines total.

TVT-6L switch must be in the "32" position.

( ) Denotes an absolute address that is program location
sensitive.

* Denotes a relative branch that is program length sensitive.

TO DISPLAY OTHER PAGES, USE:

<table>
<thead>
<tr>
<th>PAGES DISPLAYED</th>
<th>17A3</th>
<th>17C9</th>
<th>TVT CONNECTED</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000-01FF</td>
<td>22</td>
<td>20</td>
<td>KIM-1</td>
</tr>
<tr>
<td>0200-03FF</td>
<td>24</td>
<td>22</td>
<td>KIM-1</td>
</tr>
<tr>
<td>0400-05FF</td>
<td>26</td>
<td>24</td>
<td>KIM-2</td>
</tr>
<tr>
<td>0600-07FF</td>
<td>28</td>
<td>26</td>
<td>KIM-2</td>
</tr>
<tr>
<td>0800-09FF</td>
<td>2A</td>
<td>28</td>
<td>KIM-2</td>
</tr>
<tr>
<td>0A00-0BFF</td>
<td>2C</td>
<td>2A</td>
<td>KIM-2</td>
</tr>
<tr>
<td>0C00-0DFF</td>
<td>2E</td>
<td>2C</td>
<td>KIM-2</td>
</tr>
<tr>
<td>0E00-0FFF</td>
<td>30</td>
<td>2E</td>
<td>KIM-2</td>
</tr>
</tbody>
</table>

FOR HIGHER PAGES, MOVE CONTENTS TO 0200-03FF
or 0400-05FF

B. Program for a 12 line or a 25 line, 64 character per line
interlaced TVT-6L Raster Scan:

uP — 6502  Start — JMP 17bF
System — KIM-1, 2 End — Interrupt  Program Space 1780-17E2

Upper Address 1788

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>PH</th>
<th>PL</th>
<th>V8</th>
<th>V4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1, 2.4</td>
<td>blank scan</td>
<td>scan row 1</td>
<td>scan row 2</td>
<td>etc</td>
<td>scan row 11</td>
<td>vertical sync pulse</td>
</tr>
</tbody>
</table>

1780 LDA A9 (24) (17)
1782 STA 8d (88) 20
1786 NQP EA 00
1789 ADC 69 20 0
179b CMP C9 E0 0
179d BCC 90 F3*
179F PHA 48 0
1790 LDA AD (87) 17
1793 ADC 69 3F 0
1795 STA 8d (87) 17
1798 PLA 68 0
1799 NOP EA 00 20
179a JSR 20 0C 17
179d ADC 69 4C 0
179f CMP C9 2b 0
17A1 BCC 90 4F*
17A3 LDA A5 (EC) 0
17A5 ADC 69 7F* 0
17A7 BCC 90 9F* 0
17A9 STA 8d (EC) 0
17AC LDX A2 0E 0
17AE LDA A9 24 0
17b0 STA 8d (81) 0
17b3 LDA A9 (AO) 2b 0
17b5 STA 8d 07 0
17b8 LDY A0 07 0
17bA DEY 88 0
17bb BPL 10 2b* 0
17bd BCS b0 0P* 0
17bf STA 8d (EC) E0 0
17c2 LDX A2 0F 0
17C4 LDA A9 34 0
17C6 STA 8d (81) 0

Lower Address 1787

<table>
<thead>
<tr>
<th>V2</th>
<th>V1</th>
<th>H32</th>
<th>H16</th>
<th>H8</th>
<th>H4</th>
<th>H2</th>
<th>H1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tape Id — 6B</td>
<td>Program Length 99 words + 1 word page zero (EC)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Initialize Upper Address
Store Upper Address
Equalize 2
///Character Scans 0-11///
Increment Character Gen by 2
Is it scan 12 or 13?
No, Do next character scan
Save Upper Address
Get Lower Address
Increment L: Set C on V2 Overflow
Restore L: save carry
Get Upper Word
Equalize 2
///Blank scans 12, 13///
Add Carry; Reset Upper Address
Was this the last line of characters?
No, Scan a new line of characters
Get Interface Word
Set Carry If Odd Field Finished
Start Even Field if Carry Set
Even V Sync + Replace Interlace
Load Even #VB Scans -2
Initialize Even Upper Address
continued

Initialize Even Character End Compare continued
Equalize 41 microseconds continued
continued
Skip if Even Field
Odd V Sync + Replace Interlace
Load Odd #VB Scans -2
Initialize Odd Upper Address continued

START
NOTES:

TVT-6L must be connected and both the SCAN and DECODE PROMS must be in circuit for program to run.

Both 17A9 and 17bF require that page 00 be enabled when page E0 is addressed. This is done automatically in the KIM-1 decode circuitry.

Location 00EC on page zero is reserved as an interlace storage bit.

Step 1786 goes to where the upper address stored in 1788 and the lower address stored in 1787 tells it to. Values in these slots continuously change throughout the program.

Values in slots 1781 (Upper address start) and 17A0 (Character end compare) alternate with the field being scanned.

Horizontal Scan Frequency = 11.494 kHz. Vertical frequency = 60.0222 Hertz. 87 microseconds per line 191.5 lines per field: 2 fields per frame, 383 lines total.

TVT-6L switch must be in the "64" position.

() Denotes an absolute address that is program location sensitive.

* Denotes a relative branch that is program length sensitive.

Program may be used for 13 x 64 large characters or 25 x 64 small characters by changing the following slots:

<table>
<thead>
<tr>
<th>13 x 64</th>
<th>25 x 64</th>
</tr>
</thead>
<tbody>
<tr>
<td>178A</td>
<td>10</td>
</tr>
<tr>
<td>17Ad</td>
<td>14</td>
</tr>
<tr>
<td>17b4</td>
<td>24</td>
</tr>
<tr>
<td>17C4</td>
<td>28</td>
</tr>
<tr>
<td>17C5</td>
<td>15</td>
</tr>
<tr>
<td>17CA</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>28</td>
</tr>
</tbody>
</table>

FLOWCHART: 13 x 64 or 25 x 64 interlaced scan no 68

C. Program for a Four-in-One full performance Scrolling Cursor:

uP — 6502
System — Kim 1,2
Start — IRQ
End — RTI
Program Space 0100-01dF
+ Two words page zero (ED,EF)

Input to parallel Word A

<table>
<thead>
<tr>
<th>A0</th>
<th>A7</th>
<th>A6</th>
<th>A5</th>
<th>A4</th>
<th>A3</th>
<th>A2</th>
<th>A1</th>
</tr>
</thead>
</table>

IRQ

Clear — CAN (18)
Carriage Return — CR (0d)
Cursor Up — VT (0b)
Cursor Down — LF (0a)
Cursor Left — BS (08)

Cursor Right — HT (09)
Cursor Home — SOH (0a)
Scroll Up — DC1 (11)
Erase To End — ETX (03)
Enter — all characters and all unused
CTRL commands

10 usec.
load program 1A, switch to single step off, jump to 17A6 and hit GO. Your first check should be that the program will run, returning to the keyboard monitor when you hit stop and picking up on go again. Addresses should always be within the program bounds of being somewhere between 2000 and EFFF or somewhere between 1780 and 17D3.

Check test point VR for a one microsecond pulse every 16.7 milliseconds. If your scope has trouble with low duty cycle waveforms, you can try pin 6 of IC4, which should be a one microsecond or so pulse every 16.7 milliseconds. For the acid test, switch to line sync. This pulse is your vertical sync pulse. It should wander around very slowly with respect to the power line sync. This pulse is created both by the hardware and your SCAN program. Stop right here till you have it there and stable.

Now, plug in your fully modified TV or monitor to the VID output, or else a Fig. 8 modified TV to the TV output. You should have a random but stable display of characters, along with some weird control symbols. Position them and sharpen them with the controls. The CLARITY control makes the characters brighter in one direction and sharper in the other — pick what you like. At this point you should have a stable and attractive display. Use minimum contrast for sharpest characters.

The rest should be downhill all the way. Check the LCASE switch and the CURSOR switch. Around half the characters should wink cursors at you, since the cursor recognizes any bit eight set as a cursor and since you have a random page load, rather than a page of characters with a single cursor location. If everything checks out so far, you can now go on to longer character lines, external keyboards, cursor loading (don't forget to load the TRQ
NOTE: For auto-scrolling use 0145 75. For wraparound, use 0145 47.

The vector must be stored in 17FE 00 and 17FF 01.

Total available stack length is 32 words. Approximately 16 are used by operating system, cursor, and scan program. Stack must be initialized to 01FF as is done in KIM operating system. For 30 additional stack locations relocate subroutine starting at 01C2 elsewhere. For total stack availability, relocate entire program elsewhere.

To protect page, load 00F1 04. To enable entry, load 00F1 00

Cursor address is stored at 00Ed low and 00EE high on page zero.

To display cursor, load 01Ed low and 00EE high.

* Denotes a relative branch that is program length sensitive

( ) Denotes an absolute address that is program location sensitive

To match this program to the scan program, change the following slots:

<table>
<thead>
<tr>
<th>0145 75</th>
<th>0145 47</th>
</tr>
</thead>
</table>
| 01A5 BEQ F0 A0* Yes, Home cursor
| 01A7 DEC C6 (Ed) //Cursor Left/(decrement cursor)
| 01A9 LDA A9 FF Set A to page underflow
| 01Ab CMP C5 (Ed) Test for page underflow

Flowchart:

01Ad BEQ F0 (EE) Change page if off page
01AF BNE d0 99* Finish if on page
01b1 LDA A5 (EE) ///// Erase to EOS/(get cursor)
01b3 PHA 48 Save Upper Cursor location on stack
01b4 LDA A5 (Ed) Get Lower Cursor location
01b6 PHA 48 Save Lower Cursor location on stack
01b7 JSR 20 (Cb) (01) Clear to End of Screen
01b8 PLA 68 Get lower cursor location offset stack
01b9 STA 85 (Ed) Restore lower cursor
01ba STA 85 (EE) Get upper cursor location offset stack
01bb STA 85 (Ed) Restore upper cursor
01bc LDA A9 (Ed) Finish
01bd PLA 68
01be STA 85 (EE)
01bf STA 85 (Ed)
01c0 BNE d0 88* //Subroutine-HOME CURSOR
01c2 LDA A9 00 Set lower cursor to home value
01c4 STA 85 (Ed) Load A with home page value
01c6 LDA A9 02 Set upper cursor to home page
01c8 STA 85 (EE)
01ca RTS 60 Return to main cursor program
01cb LDA A9 20 ///// Subroutine-ENTER SPACES
01cd JSR 20 (d3) (01) Enter space via character entry sub
01d0 BNE d0 08* Repeat if not to end of screen
01d1 RTS 60 Return to main cursor program
01d3 STA 91 (Ed) // Subroutine-ENTER AND INCREMENT
01d5 INC E6 (Ed) Enter character and increment
01d7 BNE d0 06* Overflow of page?
01d9 INC E6 (EE)
01da LDA A9 04 Yes, Increment cursor page
01db CMP C5 (EE) Load A with page above display
01dc RTS 60 Test for Overflow
01df RTS 60 Return to main cursor program

To match this program to the scan program, change the following slots:

<table>
<thead>
<tr>
<th>16 x 32</th>
<th>KIM1 0200-03FF</th>
<th>16 x 32</th>
<th>KIM2 0400-05FF</th>
<th>13 x 64</th>
<th>KIM2 0408-07FF</th>
<th>25 x 64</th>
<th>KIM2 0404-0AFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>0106 04</td>
<td>06 02 0F</td>
<td>02 04</td>
<td>04 0F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>010A 02</td>
<td>0F 02 0F</td>
<td>04 0F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0155 1F</td>
<td>1F 3F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>016A 20</td>
<td>20 40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0179 20</td>
<td>20 40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0185 03</td>
<td>05 07</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0189 E0</td>
<td>E0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0198 20</td>
<td>20 40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01A0 01</td>
<td>03 03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01C3 00</td>
<td>00 0C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01C7 02</td>
<td>04 04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01dC 04</td>
<td>06 08</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

vector!), and so on, but don’t try too much at once. Always get the utility 512 character basic display up on a KIM-1 before trying anything fancy. Thanks to the total software control, once you are up and working and confident, there’s practically no limit to how fancy you get with your display.

More details on all this will appear in Microprocessor Based Video Displays, along with such options as a Hex-ASCII converter that displays Super Front Panel Op-Code (your whole program at once — how’s that for a debug aide?); color graphics options, use of different character generators, different micro-processors, and so on. Watch for it.
<table>
<thead>
<tr>
<th>Pin</th>
<th>Ident</th>
<th>Function</th>
<th>Load</th>
<th>8-A KIM-1 Connections</th>
<th>8-B KIM-2 Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2*</td>
<td>GND</td>
<td>Ground Return — Heavy foil or wire</td>
<td>—</td>
<td>Expansion 22</td>
<td>Connector 1</td>
</tr>
<tr>
<td>3, 4, 5</td>
<td>NC</td>
<td>No Connection — reserved</td>
<td>—</td>
<td></td>
<td>Pin 4 of U10</td>
</tr>
<tr>
<td>6*</td>
<td>VCL</td>
<td>Video Clock 02</td>
<td>1 LSTTL</td>
<td>Expansion U</td>
<td>Pin 2 of U3</td>
</tr>
<tr>
<td>7</td>
<td>VD7</td>
<td>Cursor from Display memory</td>
<td>1 LSTTL</td>
<td>Pin 12 of U5</td>
<td>Pin 6 of U3</td>
</tr>
<tr>
<td>8</td>
<td>VD6</td>
<td>ASCII Bit 7 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 12 of U6</td>
<td>Pin 10 of U2</td>
</tr>
<tr>
<td>9</td>
<td>VD5</td>
<td>ASCII Bit 6 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 12 of U7</td>
<td>Pin 2 of U2</td>
</tr>
<tr>
<td>10</td>
<td>VD4</td>
<td>ASCII Bit 5 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 12 of U8</td>
<td>Pin 6 of U2</td>
</tr>
<tr>
<td>11</td>
<td>VD3</td>
<td>ASCII Bit 4 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 12 of U9</td>
<td>Pin 10 of U1</td>
</tr>
<tr>
<td>12</td>
<td>VD2</td>
<td>ASCII Bit 3 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 12 of U10</td>
<td>Pin 2 of U1</td>
</tr>
<tr>
<td>13</td>
<td>VD1</td>
<td>ASCII Bit 2 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 12 of U11</td>
<td>Pin 6 of U1</td>
</tr>
<tr>
<td>14</td>
<td>VD0</td>
<td>ASCII Bit 1 from Display memory</td>
<td>1 NMOS</td>
<td>Pin 1 of U4</td>
<td>Pin 4 of U11</td>
</tr>
<tr>
<td>15*</td>
<td>CSI</td>
<td>Chip Select from Enable Decoding</td>
<td>1 LSTTL</td>
<td>Pin 13 of U5-U12</td>
<td>Pin 2 of U6</td>
</tr>
<tr>
<td>16*</td>
<td>CSO</td>
<td>Chip Select to Display Memory</td>
<td>TTL Out</td>
<td>Pin 12 of U4</td>
<td>Connector 3</td>
</tr>
<tr>
<td>17*</td>
<td>DEN</td>
<td>Decode Enable to KIM</td>
<td>TTL Out</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>18</td>
<td>A11</td>
<td>No Connection — reserved</td>
<td>—</td>
<td>Expansion P</td>
<td>Connector R</td>
</tr>
<tr>
<td>19</td>
<td>A12</td>
<td>Address Line 12</td>
<td>1 LSTTL</td>
<td>Expansion R</td>
<td>Connector S</td>
</tr>
<tr>
<td>20</td>
<td>A13</td>
<td>Address Line 13</td>
<td>1 LSTTL</td>
<td>Expansion S</td>
<td>Connector T</td>
</tr>
<tr>
<td>21</td>
<td>A14</td>
<td>Address Line 14</td>
<td>1 LSTTL</td>
<td>Expansion T</td>
<td>Connector U</td>
</tr>
<tr>
<td>22</td>
<td>AB15</td>
<td>Address Line 15</td>
<td>1 LSTTL</td>
<td>Expansion F</td>
<td>Connector H</td>
</tr>
<tr>
<td>23</td>
<td>A5</td>
<td>Address Line 5</td>
<td>1 LSTTL</td>
<td>Expansion E</td>
<td>Connector F</td>
</tr>
<tr>
<td>24</td>
<td>A4</td>
<td>Address Line 4</td>
<td>1 LSTTL</td>
<td>Expansion D</td>
<td>Connector E</td>
</tr>
<tr>
<td>25</td>
<td>A3</td>
<td>Address Line 3</td>
<td>1 LSTTL</td>
<td>Expansion C</td>
<td>Connector D</td>
</tr>
<tr>
<td>26</td>
<td>A2</td>
<td>Address Line 2</td>
<td>1 LSTTL</td>
<td>Expansion B</td>
<td>Connector C</td>
</tr>
<tr>
<td>27*</td>
<td>A1</td>
<td>Address Line 1</td>
<td>1 LSTTL</td>
<td>Expansion 8</td>
<td>Connector 8</td>
</tr>
<tr>
<td>28</td>
<td>DB7</td>
<td>Data Bus 7</td>
<td>TTL TS Out</td>
<td>Expansion 9</td>
<td>Connector 9</td>
</tr>
<tr>
<td>29</td>
<td>DB6</td>
<td>Data Bus 6</td>
<td>TTL TS Out</td>
<td>Expansion 10</td>
<td>Connector 10</td>
</tr>
<tr>
<td>30</td>
<td>DB5</td>
<td>Data Bus 5</td>
<td>TTL TS Out</td>
<td>Expansion 11</td>
<td>Connector 11</td>
</tr>
<tr>
<td>31</td>
<td>DB4</td>
<td>Data Bus 4</td>
<td>TTL TS Out</td>
<td>Expansion 12</td>
<td>Connector 12</td>
</tr>
<tr>
<td>32</td>
<td>DB3</td>
<td>Data Bus 3</td>
<td>TTL TS Out</td>
<td>Expansion 13</td>
<td>Connector 13</td>
</tr>
<tr>
<td>33</td>
<td>DB2</td>
<td>Data Bus 2</td>
<td>TTL TS Out</td>
<td>Expansion 14</td>
<td>Connector 14</td>
</tr>
<tr>
<td>34</td>
<td>DB1</td>
<td>Data Bus 1</td>
<td>TTL TS Out</td>
<td>Expansion 15</td>
<td>Connector 15</td>
</tr>
<tr>
<td>35</td>
<td>DB0</td>
<td>Data Bus 0</td>
<td>TTL TS Out</td>
<td>—</td>
<td>Connector Y</td>
</tr>
<tr>
<td>36*</td>
<td>+5V</td>
<td>+5 Volt Supply</td>
<td>200 ma</td>
<td>Expansion 21</td>
<td>—</td>
</tr>
</tbody>
</table>

NOTES: (See * Above)

Pins 1, 2 — Ground should be heavy foil or #18 wire — all other connections are wire pencil short leads. Do not use ribbon cables or attempt extension.

Pin 6 — Video clock must load character generator only when data output is stable and valid. Clock 02 on the KIM.

Pins 15, 16 — Chip select line from decoding to display memory is broken by cutting the foil and then replaced with a negative logic OR (positive AND) of the original chip select and the TVT chip select. See Figure 8b.

Pin 17 — Decode Enable output goes low when TVT is not scanning; goes high otherwise. Decoding must be disabled during active scans to allow SCAN memory access to data buss. See Figure 8b.

Pin 27 — Address line A0 is not used in the TVT module as the SCAN memory indexes every second microsecond. A0 is used in the display memory addressing.

Pin 36 — +5 power borrowed from computer. Extra noise on the +5 line will cause skewed or awkward characters; may be fixed usually with extra bypassing. Use heavy foil or #18 wire.

Table II. TVT-6L Interface.