Jim Trageser
5648 Brookbank Dr.
Kettering OH 45440

Budget System with KIM

A lot of us got started with a KIM-1. This article shows you how to expand your basic KIM economically.

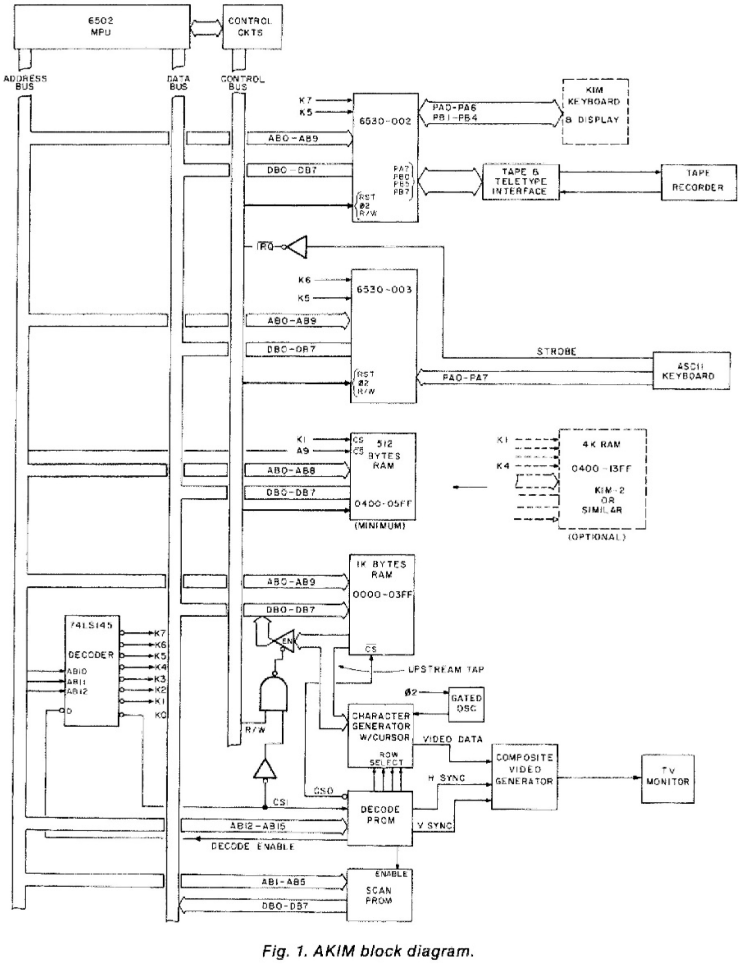


Fig. 1. AKIM block diagram.

This article shows you how to advance beyond the hex keyboard and display without spending more than double your initial investment in KIM. Don Lancaster's article, "A TVT for your KIM," in the June 1977 *Kilobaud* describes an excellent low-cost way to add a quality CRT display to your microcomputer. To obtain full terminal capability, you need only add a low-cost ASCII keyboard to Don's TVT-6L. New ASCII keyboard kits or surplus-assembled keyboards can be purchased for less than $49. Total TVT and keyboard cost can be less than $100.

This article shows how to add the keyboard and use it instead of the hex keyboard on your KIM or home-brew 6502 system. Software AKIM (ASCII keyboard input monitor) will allow you to perform KIM monitor functions using the TVT and ASCII keyboard. If you have been reluctant to add Don's TVT-6L setup because of its use of memory locations 2000 to EFFF, *apparently* excluding them from other use, I will show a simple technique to switch between TVT and full memory usage under software control.

Background

MOS Technology was the first to make it possible for many microcomputer enthusiasts to get hands-on experience without investing a lot of time and money in designing a personal computer or having a high degree of technical expertise.

I started my 6502-based system as a home-brew computer because I was determined to get some hands-on experience with microprocessors (this was before the KIM-1 had been introduced). It was the only way to go at the time.

After five months of designing and building, I was just starting to write some simple software and proudly watched my microcomputer single-step through the instructions. *Then* the KIM-1 was introduced. As a pre-KIM 6502 owner, I was able to purchase the 6530-002 and 6530-003 memory-I/O timer arrays with KIM software in ROM. Although my microcomputer may not look outwardly like a KIM-1, functionally it is identical, including the cassette interface.

Once thoroughly comfortable with my minimum computer system and having developed some expertise in programming skills, I wanted to add a terminal: either a Teletype or CRT with keyboard. Planning this next step, I also wanted to add assembler and BASIC language capability. All this requires another 4K to 8K of memory.

AKIM. Home-brew KIM-1 after TVT-6L and ASCII keyboard are added. Microcomputer, power supplies and keyboard are mounted in surplus KSR 35 enclosure.

For over six months I seriously searched for the lowest-cost method of getting to the next step short of devoting a lot of time to designing and building my own. Even with the significant reductions in the costs of memory and keyboards during the past year, the cost of adding a terminal and 4K to 8K of RAM was still three to four times the original cost of KIM-1.

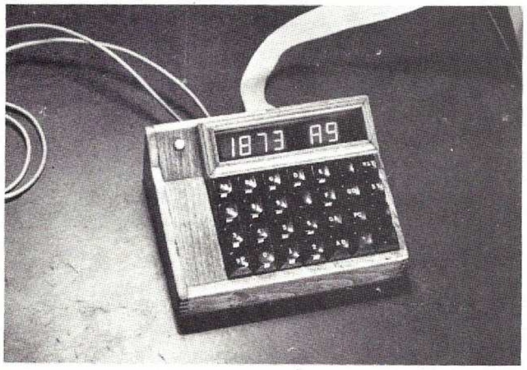
Assuming a potential user has a fixed educational or hobby budget, his goal is to obtain the most capability within the budget. Mine started out as a continuing education budget, but I've had so much fun learning that it has become difficult to differentiate between the educational and the hobby aspects.

I was trying to find a way to reduce the cost of the terminal so I could allocate more of my budget to memory expansion. The TVT-6L was just what I was looking for. The material costs were low, and the simplicity of the hardware required minimum construction time. Furthermore, I could get back to what I really wanted to do—play with the software.

Systems Configuration

AKIM requires a minimum of 512 bytes of RAM added to the BASIC KIM-1 system. This allows you to use the TVT-6L in the 16 line x 32 character-per-line format with AKIM and Don's scrolling cursor program. For 13 x 64 or 25 x 64 screen formats, you need the equivalent of KIMs 1 and 2. I had an extra 512 bytes of memory left over from my original home-brew (pre-KIM) system.

In the KIM-1-configuration conversion, I added a full 1K of RAM, using Fairchild 2102L1s to keep the power-supply requirements low, and located the entire 512 bytes at address locations 0400 to 05FF. A block diagram of the system with TVT-6L, ASCII keyboard and



Home-brew KIM-1 remote keyboard and display. Display drivers, TTY interface and audio cassette interface electronics are located in keyboard enclosure.

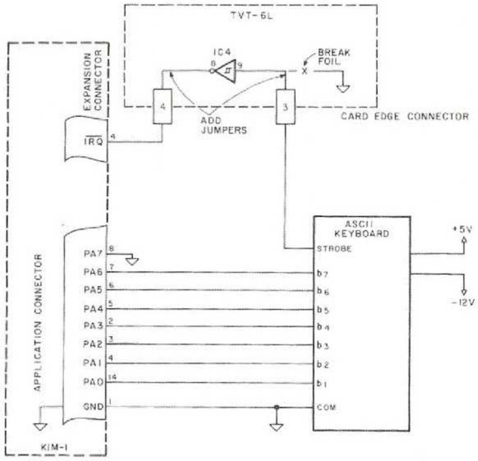
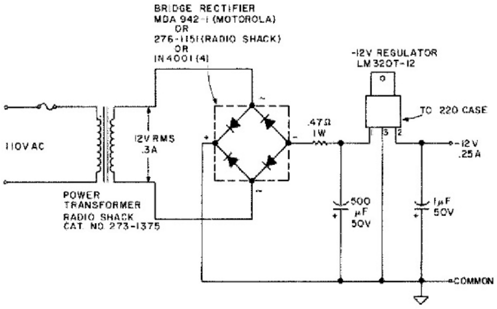


Fig. 2. ASCII keyboard interface wiring.



Notes. For +5 V supply.

- Substitute LM340T-5 for LM320T-12 regulator.
- Reverse connections (+/−) to bridge rectifier and filter capacitors.
- *Separate transformers must be used* for positive and negative supplies.
- To increase output current to 1 A maximum, change transformer to 12.6 V @ 1.2 A rating (Radio Shack catalog no. 273-1505). Add heat sink to voltage regulator, change 500 uF capacitor to 2000 uF.

Fig. 3. Schematic diagram. −12 V power supply for ASCII keyboard.

PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
0	b7	b6	b5	b4	b3	b2	b1

			MOST SIGNIFICANT DIGIT							
	HEX			0	1	2	3	4	5	6	7
		BINARY b4 b3 b2 b1	b7→ b6→ b5→	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
LEAST SIGNIFICANT DIGIT	0	0 0 0 0		NUL	DLE	SPACE	0	@	P	`	p
	1	0 0 0 1		SOH	DC1	!	1	A	Q	a	q
	2	0 0 1 0		STX	DC2	"	2	B	R	b	r
	3	0 0 1 1		ETX	DC3	#	3	C	S	c	s
	4	0 1 0 0		EOT	DC4	$	4	D	T	d	t
	5	0 1 0 1		ENQ	NAK	%	5	E	U	e	u
	6	0 1 1 0		ACK	SYN	&	6	F	V	f	v
	7	0 1 1 1		BEL	ETB	'	7	G	W	g	w
	8	1 0 0 0		BS	CAN	(	8	H	X	h	x
	9	1 0 0 1		HT	EM	)	9	I	Y	i	y
	A	1 0 1 0		LF	SUB	*	:	J	Z	j	z
	B	1 0 1 1		VT	ESC	+	;	K	[	k	{
	C	1 1 0 0		FF	FS	,	<	L	\	l	|
	D	1 1 0 1		CR	GS	-	=	M	]	m	}
	E	1 1 1 0		SO	RS	.	>	N	^	n	~
	F	1 1 1 1		SI	US	/	?	O	_	o	DEL

Table 1. ASCII character set.

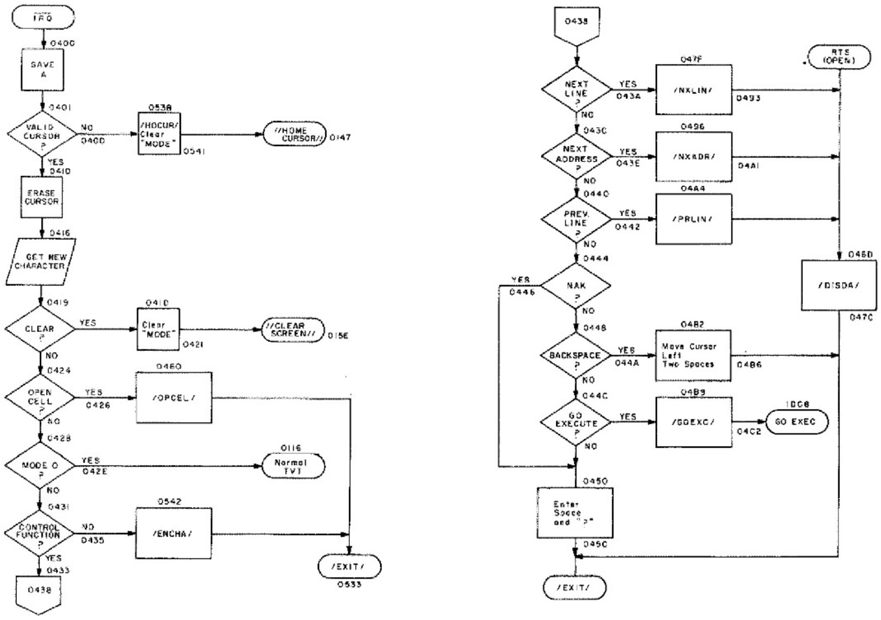
added RAM is shown in Fig. 1.

Note that the KIM-1 decoder IC (U4) is shown as 74LS145. This may not be necessary at this time in your system, but with the added loading of the TVT-6L DECODE PROM on AB12, I recommend changing the existing 74145 with 74LS145 to reduce the address-bus loading. This allows other decoders to be added to the high-address lines for memory expansion without resorting to bus drivers in these lines.

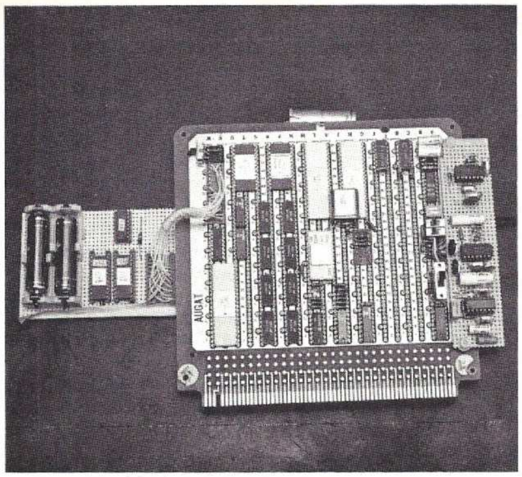
Adding the ASCII Keyboard

AKIM is based on connecting a seven-bit parallel ASCII-coded output keyboard to port A of U3 on KIM-1. This connection is made via PA0-PA6 at the application connector. PA7 must be tied low; otherwise, it may be read and displayed as a cursor bit. The keyboard should have a positive logic output. Some available keyboards use a negative logic (true=0) output. These keyboards require inverters between the keyboard output and the port A input to work with the AKIM and scrolling cursor software as written.

Other than the required power connection to the keyboard, only one more connection is required: the *strobe.* The keyboard strobe output must be connected to the KIM-1 $\overline{IRQ}$ line available at the *expansion* connector. The strobe output



AKIM flowchart. Main routine.

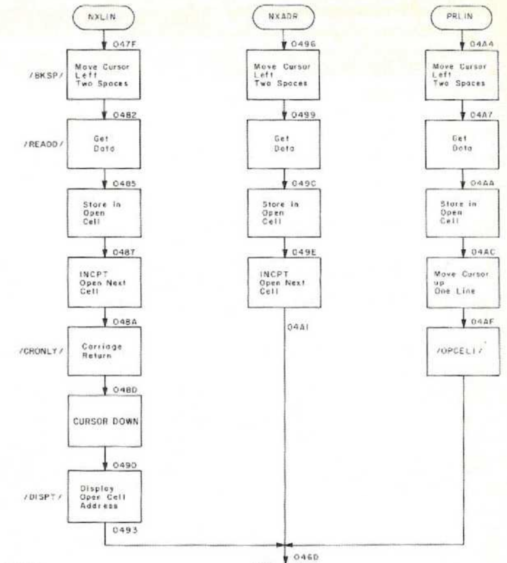


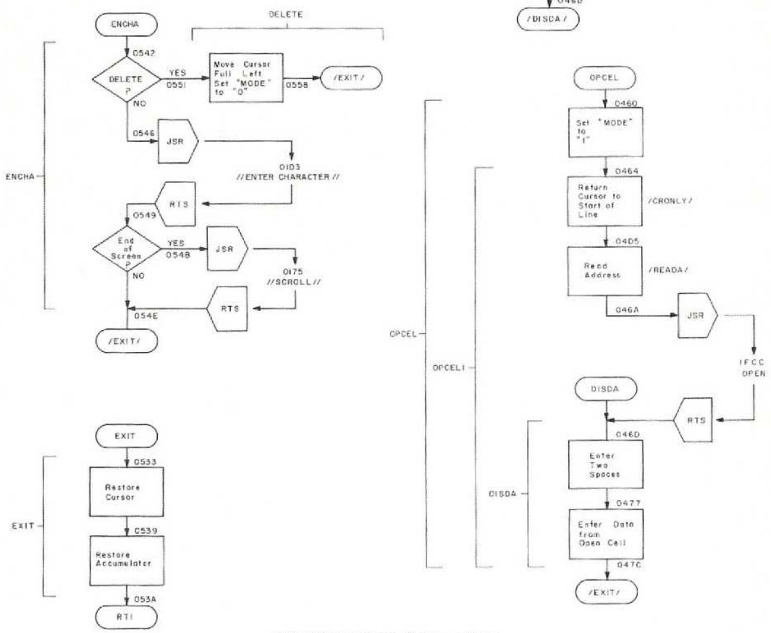
Home-brew KIM board with TVT-6L circuits added on right. CMOS RAM with batteries is on left.

from the keyboard must be normally high and pulse low when active to work with the $\overline{\text{IRQ}}$ input. If your keyboard is like mine, the strobe and data outputs are all positive logic, which requires the strobe signal to be inverted prior to connection to the $\overline{\text{IRQ}}$ input. If you don't want to add an extra IC, there is an unused section (pins 8 and 9) of IC4 on the TVT-6L for this purpose.

If the TVT-6L is on a printed circuit board, you must first cut the trace connecting pin 9 of IC 4 to ground. Then solder jumper wires from pins 8 and 9 to two unused pins (3 and 4) on the TVT-6L connector (see Fig. 2). Some keyboards may also have a *parity* bit output that is not necessary in many applications and is not used in AKIM.

The power requirements for my keyboard were +5 volts at 350 mA and −12 volts at negligible current. If your keyboard is similar and you are using the power supply recommended for the KIM-1, the addition of the TVT-6L and ASCII keyboard will cause the +5 V power requirements to exceed power-supply capacity. The −12 V requirements are attained by using a circuit similar to that shown for the +12 V supply in the KIM-1 user's manual. Fig. 3 shows a variation of the power-supply





AKIM flowchart. Subroutines.

Introduction to AKIM Program

AKIM is a program that allows owners and users of KIM-1/6502 systems with a TVT-6L TV interface and an ASCII keyboard to use the TV terminal in a monitor mode in lieu of the normal KIM-1 keyboard. Functions provided include:

1. Calling any memory location and displaying the contents stored (open cell).
2. Stepping to next address, displaying the address and contents on the next line by pressing LF.
3. Stepping to the next address without displaying address but displaying contents of address two spaces to the right of the previous address data by pressing TAB.
4. Changing contents of open cell by pressing back space and typing in new data prior to execution of LF or TAB.

5. LF and TAB functions automatically load data displayed to left of cursor into open cell and increment address to open next cell.
6. Vertical tab, VT, function moves cursor to start of previous line and reopens cell displayed there.
7. Exit of the monitor routine is accomplished by either the DEL, CAN or GS keys.
8. Execution of the DEL function returns the cursor to the start of the present line and returns the program to the TVT mode for free-form use.
9. Execution of the CAN function clears the screen, returns the cursor to home and returns the program to the free-form TVT mode.
10. Execution of the GS function causes the system to exit the monitor program and execute the program starting at the address displayed at the beginning of the line displaying the cursor.

Monitor routine for KIM-1/6502 systems with TVT-6L interface and ASCII keyboard input.

Hardware Required

(1) KIMs 1 and 2 or KIM-1 with add-on memory of 512 bytes.
(2) TVT-6L TV interface described by Don Lancaster in June 1977 *Kilobaud*.
(3) ASCII keyboard interfaced to Port A application input.

Associated Software

(1) KIM subroutines—INCPT (1F63)
PACK (1FAC)
OPEN (1FCC)
(2) TVT-6L RASTER SCAN program (June 1977 *Kilobaud* p. 58).
(3) Scrolling Cursor program (June 1977 *Kilobaud* pp. 60, 61 and 62) with following correction and change:
—Location 011D listed in error. Change "30" to "40" at this location.
—Replace ///RESTORE CURSOR/// subroutine starting at 014A with the following:

```
014A  JMP  4C    (2C)  (05)   Check "MODE" & finish.
014D  NOP  EA
014E  NOP  EA
014F  NOP  EA
0150  NOP  EA
0151  BRK  00                 Trap
```

Key Function List

CLEAR — CAN (18),	Clear screen & reset MODE to 0. //TVT ONLY//.
OPEN CELL — SI (0F),	Place in monitor mode (MODE 1) and open cell specified at start of line. Display current data in cell.
BACK SPACE BS (08),	Move cursor two spaces to left. Used as a convenience for altering cell data and writing new programs.
TAB — HT (09),	Enter data displayed left of cursor in open cell. Open next cell, tab two spaces to right and display current data of new open cell.
LINE FEED — LF (0A),	Enter data to left of cursor in open cell. Open next cell, command CR & LF, display address of open cell at start of next line, enter two spaces and display current data of open cell.
DELETE — DEL (7F),	Exit monitor mode, (clear "MODE" to 0), return cursor to start (left side) of current line.
PREVIOUS LINE — VT (0B),	Move cursor to start of line above current line. Open cell and display current data in open cell.
GO EXECUTE — GS (1D),	Exit monitor program and execute program starting at location displayed at start of line.

Attempts to execute any other control functions while in monitor mode will cause ? to be displayed on screen.

Program Space: 0400—055C

Start: IRQ
End: RTI

Tape Ident: 1D

Start-up: (1) Set IRQ, 17FE = 00
17FF = 04
(2) Initialize 00F1 — 00
(3) After all 3 Programs loaded go to 17A6 & press GO
(4) The first keyboard function initiated should be CLEAR SCREEN (CAN).

circuit shown in the manual.

This should take care of the requirements for most ASCII keyboards. However, you may want to consider adding a power supply at this time to take care of future memory expansion. Power-supply kits and surplus power supplies are available at reasonable prices. Scan the ads in *Kilobaud*.

When you have the keyboard connected, check for proper code inputs from the keyboard before trying the keyboard software. Call up address 1700 on your KIM keyboard. Since the Reset function or initialization of KIM sets all data-direction registers as inputs, PA0 through PA7 at the *application* connector will be set up as inputs. It is not necessary to reinitialize the data-direction registers as inputs. Press any character key on the ASCII keyboard. The data displayed on your KIM display should be the two-digit hex equivalent of the binary ASCII code. See Table 1 for ASCII-to-hex code conversions.

After you have verified that the keyboard is correctly entering data, you are ready to run the cursor or AKIM software. Prior to jumping to the SCAN routine, set location 00F1 to 00 to insure that the IRQ mask is disabled. Load hex 0400 in the IRQ vector (17FE and 17FF) if you are running AKIM.

Before we go on to the AKIM software, a comment regarding the interface to a TV monitor is in order. If you are using a portable TV other than the Panasonic T-126A mentioned in Don's article, some adjustment of the dc offset voltage at the TV output jack (J1) may be necessary. I used a 9-inch RCA model AJ-005B b & w set. The dc bias of the TVT-6L was too high, as evidenced by a white screen and no sync. The dc level can be reduced in .5 V increments by shunting one or more of the series diodes D5-D8 between J1 and J2.

AKIM Software

With the keyboard interface functioning, I loaded in Don's Scan and Scrolling Cursor programs. The TVT came to life.

After minor sync adjustments, I checked out the keyboard functions. Every function operated as expected, except for the Clear function. The problem was traced to a typo in the Scrolling Cursor program listing. The branch instruction at location 011D should be 40 instead of 30. After this correction was made everything operated correctly.

At this point, I turned it over to the kids while I planned the next step. They had a great time drawing pictures with ASCII characters and playing word games and Ticktacktoe.

AKIM was developed to allow the TVT and ASCII keyboard to be used instead of the KIM keyboard and hex display for the basic monitor function using minimum additional program space. Don's original Scrolling Cursor program was retained to allow the keyboard to operate in either the AKIM monitor

Program listing.

```
     Address         Op code     Symbol  Mnemonic               Comments
      0400    48                         PHA                    Save A
      0401    A0       00                LDY    #$00            Reset Y index
      0403    A5       EE                LDA    CURSH           Get cursor and test
      0405    C9       04                CMP    #$04            Below max.?
      0407    B0       04                BCS    home            No! Home cursor
      0409    C9       02                CMP    #$02            Above min.?
      040B    B0       03                BCS    ecur            Yes! Get cursor
      040D    4C       3B   05   home    JMP    HOCUR           No! Jump to HOME CURSOR
      0410    B1       ED        ecur    LDA    (ED),Y          Get cursor character
      0412    29       7F                AND    #$7F            Erase cursor
      0414    91       ED                STA    (ED),Y          Replace character
      0416    AD       00   17           LDA    (A par. input)  Get new character
      0419    C9       18                CMP    #$18            Clear Screen?
      041B    D0       07                BNE    CONT1           No! Go to CONT1
      041D    A2       00                LDX    #$00            Yes! Clear MODE
      041F    86       FF                STX    MODE
      0421    4C       5E   01           JMP    015E            Go clear screen
      0424    C9       0F        CONT1   CMP    #$0F            Open cell?
0460  0426    F0       38                BEQ    OPCEL           Yes!
      0428    A6       FF                LDX    (MODE)          Check mode
      042A    E0       00                CPX    #$00
      042C    D0       03                BNE    MODE1           Mode 1
      042E    4C       16   01           JMP    0116            Mode 0
      0431    C9       20        MODE1   CMP    #$20            Character to enter?
      0433    90       03                BCC    GETCON          No! Which control?
      0435    4C       42   05           JMP    ENCHA           Enter Character
      0438    C9       0A        GETCON  CMP    LF              Start new line?
047F←043A     F0       43                BEQ    NXLIN           Yes!
      043C    C9       09                CMP    HT              Open next cell?
0496←043E     F0       56                BEQ    NXADR           Yes!
      0440    C9       0B                CMP    VT              Previous Line?
04A4←0442     F0       60                BEQ    PRLIN           Yes!
      0444    C9       15                CMP    NAK             (Future mode)
      0446    F0       08                BEQ    LEAVE           Invalid control
      0448    C9       08                CMP    BS              Back space?
04B2  044A    F0       66                BEQ    BCKSPA          Yes!
      044C    C9       1D                CMP    GS              Go Execute?
04B9  044E    F0       69                BEQ    GOEXC           Exit to called program
      0450    A9       20        LEAVE   LDA    #$20            Print "space"
      0452    91       ED                STA    (ED),Y            cont.
      0454    E6       ED                INC    ED              Increment cursor
      0456    A9       3F                LDA    #$3F            Print "?"
      0458    91       ED                STA    (ED),Y            cont.
      045A    E6       ED                INC    ED              Increment cursor
      045C    4C       33   05           JMP    EXIT            Restore cursor and
                                                                  return to scan
      045F    EA                         NOP
                                 ; Subroutines
                                 ;
                                 ;
                                 ;
                                 ; Open New Cell
      0460    A9       01        OPCEL   LDA    #$01        }   Set "MODE"
      0462    85       FF                STA    MODE        }   to "1"
      0464    20       C5   04   OPCEL1  JSR    CRONLY          Move Cursor to
                                                                start of line
      0467    20       D5   04           JSR    READA           Read address
      046A    20       CC   1F           JSR    OPEN            Open Cell (KIM 1FCC)
      046D    A9       20        DISDA   LDA    #$20        }
      046F    91       ED                STA    (ED),Y      }   Print 2
      0471    E6       ED                INC    CURSL       }   spaces
      0473    91       ED                STA    (ED),Y      }   on TVT
      0475    E6       ED                INC    CURSL       }
      0477    B1       FA                LDA    (Point L),Y }   Print current
      0479    20       E4   04           JSR    DISBYT      }   cell data in
                                                                next 2 TVT spaces
```

mode or free-form TV typewriter mode. The AKIM mode is entered by depressing the CTRL and 0 keys (SI). Return to TVT mode is accomplished by depressing delete (DEL) or CTRL and X keys (CAN).

The Back Space function included in AKIM need not be executed to alter the data in the open memory cell. It was only included to provide *neat* listings. When the BS key is depressed while in AKIM mode, the cursor is automatically moved two spaces to the left so that new data can replace the current data displayed. The data is not written into the open cell until the HT or LF keys are depressed.

If the Back Space key is not depressed, the new data will be displayed to the right of the current data in the open cell. Executing HT or LF will cause the two characters (new data) to the left of the cursor to be written into the open cell. Both the previous data and the current data will continue to be displayed.

The Tab function was includ-

Symbol	Address Defined
CURSL	00ED
CURSH	00EE
MODE	00FF
BCKSP	051E
BCKSPA	04B2
CRONLY	04C5
DELETE	0551
DISDA	046D
DISBYT	04E4
DISPT	0514
ENCHA	0542
EXIT	0533
FINISH	052C
GETCON	0438
GOEXC	04B9
HEXTA	04F5
HEXTA1	04FE
HOCUR	053B
LEAVE	0450
NXADR	0496
NXLIN	047F
OPCEL	0460
OPCEL1	0464
PRLIN	04A4
READA	04D5
READD	0503
VTAB	0523
KIM	
INCPT	1F63
PACK	1FAC
OPEN	1FCC

Symbol table.

```
047C   4C    33  05           JMP    "EXIT"         Restore cursor
                                                     & RTI
                     ;
                     ;
                      Open cell at start of
                      next line. Print address
                      and data contents
047F   20    1E  05  NXLIN   JSR    BCKSP           Move cursor left
0482   20    03  05          JSR    READD           Get data from TVT
0485   91    FA              STA    (POINTL),Y      Store data
0487   20    63  1F          JSR    INCPT           In current address
048A   20    C5  04          JSR    CRONLY
048D   20    66  01          JSR    //CURSOR DOWN//
0490   20    14  05          JSR    DISPT           Display Point
0493   4C    6D  04          JMP    DISDA           Display Data
                     ; Get data preceding cursor
                     ; and store in open cell.
                     ; Increment address (open next
                     ; cell). Display current data
                     ; stored at open cell.
0496   20    1E  05  NXADR   JSR    BCKSP      }    Store
0499   20    03  05          JSR    READD      }    previous
049C   91    FA              STA    (POINTL),Y }    data
049E   20    63  1F          JSR    INCPT           Incr address
04A1   4C    6D  04          JMP    DISDA
                     ; Get data preceding cursor and
                     ; store in open cell. Move cursor
                     ; to start of line above present
                     ; cursor. Open cell at start of line.
04A4   20    1E  05  PRLIN   JSR    BCKSP
04A7   20    03  05          JSR    READD
04AA   91    FA              STA    (POINTL),Y
04AC   20    23  05          JSR    VTAB
04AF   4C    64  04          JMP    OPCEL1
                     ; Move cursor left two spaces.
                     ; Exit to TVT Scan routine.
04B2   C6    ED      BKSPA   DEC    CURSL
04B4   C6    ED              DEC    CURSL
04B6   4C    33  05          JMP    Exit
                     ; Move cursor to start of line.
                     ; Get address and jump to
                     ; execute program starting at
                     ; address displayed.
04B9   20    C5  04  GOEXC   JSR    CRONLY
04BC   20    D5  04          JSR    READA
04BF   20    CC  1F          JSR    OPEN
04C2   4C    C8  1D          JMP    GO EXEC (IDC8)
                     ; Carriage return only. Move
                     ; cursor full left to start
                     ; of current line
04C5   A5    ED      CRONLY  LDA    CURSL           Get cursor
04C7   09    1F              ORA    #$1F            Move to right
04C9   85    ED              STA    CURSL           Restore
04CB   E6    ED              INC    CURSL           Increment
04CD   A5    ED              LDA    CURSL   }
04CF   38                    SEC            }       Move up 1 line
04D0   E9    20              SBC    #$20    }
04D2   85    ED              STA    CURSL           Restore
04D4   60                    RTS
                     ; Read address specified by cursor.
                     ; Convert from ASCII to hex.
                     ; Store in INH & INL
                     ;
04D5   A2    04      READA   LDX    #$04
04D7   A0    00      PACK 1  LDY    #$00
04D9   B1    ED              LDA    (CURS)
04DB   20    AC  1F          JSR    Pack
04DE   E6    ED              INC    CURSL
04E0   CA                    DEX
04E1   D0    F4              BNE    Pack 1
04E3   60                    RTS
                     ; Display one hex byte in A
                     ; as two ASCII characters
                     ;
                     ;
04E4   85    FC      DISBYT  STA    Temp }
04E6   4A                    LSR    A    }    Get high
04E7   4A                    LSR    A    }    byte
04E8   4A                    LSR    A    }
04E9   4A                    LSR    A    }
```

```
04EA   20      F5   04           JSR     HEXTA
04ED   A5      FC                LDA     Temp              Get low byte
04EF   20      F5   04           JSR     HEXTA
04F2   A5      FC                LDA     Temp
04F4   60                        RTS
                         ; Convert hex to ASCII and load TVT
                         ; at position defined by cursor
04F5   29      0F        HEXTA   AND     #$0F
04F7   C9      0A                CMP     #$0A
04F9   18                        CLC
04FA   30      02                BMI     HEXTA1
04FC   69      07                ADC     #$07
04FE   69      30        HEXTA1  ADC     #$30
0500   4C      D3   01           JMP     01D3
                         ; Get 2 characters from TVT convert
                         ; to hex and load into INL and A
0503   A2      02        READD   LDX     #$02
0505   A0      00                LDY     #$00
0507   B1      ED        PACK1   LDA     (CURS),Y
0509   20      AC   1F           JSR     Pack
050C   E6      ED                INC     CURSL
050E   CA                        DEX
050F   D0      F6                BNE     Pack1
0511   A5      F8                LDA     INL
0513   60                        RTS
                         ; Get current address and
                         ; display on TVT
                         ;
0514   A5      FB        DISPT   LDA     POINTH
0516   20      E4   04           JSR     DISBYT
0519   A5      FA                LDA     POINTL
051B   4C      E4   04           JMP     DISBYT
                         ; Move cursor left 2 spaces
                         ; and return to calling routine
                         ;
051E   C6      ED        BCKSP   DEC     CURSL
0520   C6      ED                DEC     CURSL
0522   60                        RTS
                         ;
                         ; Move cursor up 1 line
                         ;
0523   20      C5   04   VTAB    JSR     CRONLY
0526   38                        SEC
0527   E9      20                SBC     #$20
0529   85      ED                STA     CURSL
052B   60                        RTS
                         ; Check Mode and return to
                         ; calling routine if MODE 1.
                         ; If Mode 0 return to SCAN.
052C   A5      FF        FINISH  LDA     Mode
052E   C9      00                CMP     #$00     Check Mode
0530   F0      01                BEQ     EXIT
0532   60                        RTS                       Return to Mode 1
                                                           Calling Routine
0533   B1      ED        EXIT    LDA     CURSL
0535   09      80                ORA     #$80              Restore cursor
0537   91      ED                STA     CURSL
0539   68                        PLA                       Restore Accum.
053A   40                        RTI                       Return to Scan
                         ; Initialize Mode to 0 and
                         ; home cursor
                         ;
053B   A9      00        HOCUR   LDA     00
053D   85      FF                STA     FF                Clear Mode
053F   4C      47   01           JMP     ///HOME CURSOR///
                         ; Mode 1 character entry. If "DEL"
                         ; reset mode to 0 return cursor to
                         ; start of line and exit. All other
                           characters enter and exit to Scan.
0542   C9      7F        ENCHA   CMP     #$7F              Delete?
0544   F0      0B                BEQ     DELETE            Yes
0546   20      D3   01           JSR     //ENTER CHARACTER//
0549   D0      03                BNE                       Screen overflow?
054B   20      75   01           JSR     //SCROLL//        Yes
054E   4C      33   05           JMP     EXIT              No, Exit
0551   20      C5   04   DELETE  JSR     CRONLY            Move Cursor left
0554   A9      00                LDA     #$00
0556   85      FF                STA     FF                Reset Mode
0558   4C      33   05           JMP     EXIT              Exit to Scan
055B   FF                                END
```

ed for ease of visual editing, so that all the bytes of a single instruction may be displayed on the same line.

Full Memory Usage with TVT

Now that you have an ASCII terminal, it's time to *think* BASIC or Assembler. But since the TVT-6L uses address location 2000 to EFFF, there appears to be no place to put the required memory. Look again! A very minor modification to the TVT-6L and a few simple software instructions allow you to switch between TVT and full-memory access under program control.

Since the input to the Decode PROM (IC3) on the TVT-6L is tied low, the TVT function is enabled anytime addresses 2000 to EFFF are called up. The secret to full-memory access is to turn off the Decode PROM under software control. To do this we need only remove the ground connection on the $\overline{EN}$ input to the Decode PROM (TVT-6, IC3 pin 15) and connect it to an unused peripheral output port on KIM-1, such as PB0 on U3. Fig. 4 shows the necessary modification to the TVT-6L and the addition of 74LS145s for address decoding to high-order RAM.

If you use the TVT-6L PC board, remove the jumper wire connecting IC3 pin 15 to ground and add a jumper between IC3 pin 15 and the unused PC connector pin 5. Add a new wire between the TVT-6L mating connector pin 5 and KIM-1 application connector pin 9 *or* U3 pin 25. Note that a Tri-state buffer (74LS125) is connected between pins 15 and 16 of the TVT-6L board. This allows normal functioning of KIM-1 decode line K0 when the TVT-6L function is disabled. The three remaining sections of the 74LS125 buffer address lines AB10, AB11 and AB12.

When PB0 is low, the TVT is enabled; when PB0 is high, the TVT is disabled and full-memory access is enabled. Since PB0 is initialized as an input, the pull-up resistor on PB0 will disable the Decode PROM and cause the system to start out in the normal KIM mode. When starting up, the first step after Reset is to write a 1 into the data-direction register corresponding to U3-PB0 to set it as an output. After this step, insert instructions in your programs to write a 1 in PB0 to leave the TVT mode and to write a 0 in PB0 to return to the TVT mode.

A word of caution: Instructions to enable the TVT Decode PROM must reside below address 2000. When running programs residing at 2000 and up, use a linking routine located in lower pages to enter the TVT mode.

KIM was great, but Don's TVT-6L with AKIM has opened up exciting new possibilities at a cost every KIM owner should be able to afford.■

Notes.

- ●Light lines are existing KIM-1 & TVT-6L circuits.
- ●Heavy lines are changes and additions.
- ●Only portions of KIM-1 & TVT-61 affected by change are shown.
- ●Pull-up resistors required on outputs of open-collector decoders (74LS145) not shown.
- ●PB0 = 0, TVT mode.
- ●PB0 = 1, KIM mode with full memory use.

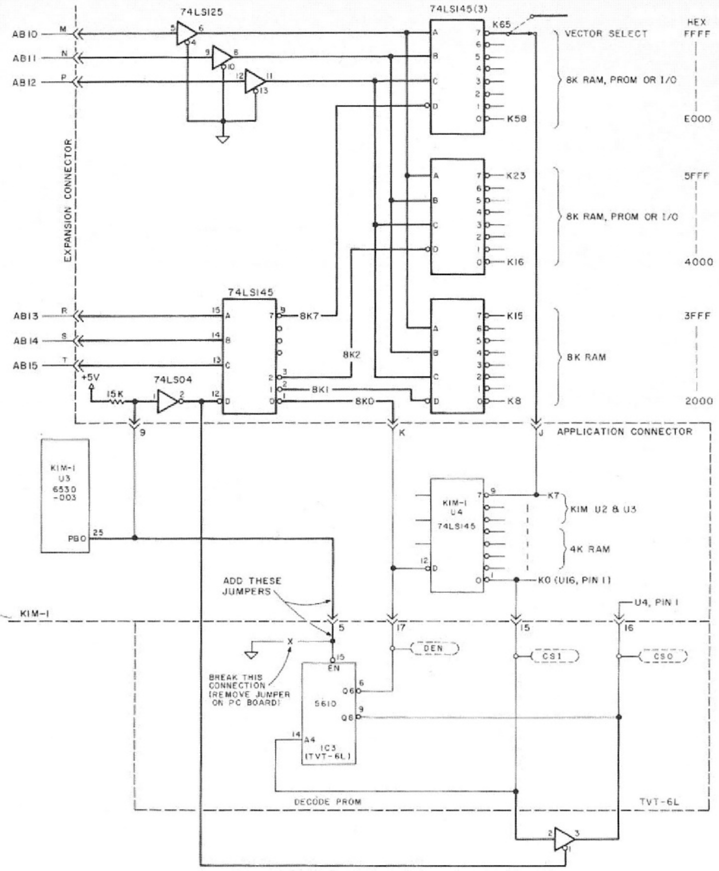


Fig. 4. KIM/TVT-6L memory expansion schematic.