ASK™ VIDEO PLUS Software

"Glass Teletype" Software
for use with the
AIM - SYM - KIM
and the
VIDEO PLUS

Description of the ASK VIDEO PLUS Software

This software package makes the VIDEO PLUS much easier to interface with the AIM, the SYM and the KIM. It has been developed in response to a number of requests and suggestions from VIDEO PLUS owners. The basic features of this new package are:

1. It will work with the AIM, SYM or KIM without modification. The program contains code to determine which microcomputer is being used and makes all necessary adjustments automatically. The program can be placed in EPROM and still be used on all three micros.
2. It interfaces directly to the AIM and SYM monitor via their provisions for OUTPUT and/or INPUT vectors, and it sets up these vectors automatically during initialization of the program.
3. It supports an ASCII keyboard so that a keyboard may be added to the AIM, SYM or KIM with full UPPER and lower case capabilities.
4. It works directly with AIM BASIC and SYM BASIC.
5. It supports the following control functions:

Carriage Return		CR	Position Cursor at start of next line
Home		↑H	Position Cursor at upper left corner
Up/Down/Left/Right		↑U ↑D ↑L ↑R	Move Cursor without altering screen contents
Scroll		↑S	Automatic and Manual Scroll
Upper Case Mode		↑A	Automatically convert alpha characters to upper case
Lower Case Mode		↑A	Permit lower case characters, even on SYM
Echo		↑F	Automatic Echo may be selected or supressed
Parity/PCG Characters		↑P	Bit 80 may be permitted or supressed
Auto Linefeed		↑Q	Linefeed following Carriage Return may be supressed
Escape/Break	[↑B]	ESC	Return to AIM/SYM/KIM Monitor from Keyboard
Delete	[_]	DEL	Delete characters in Editor, BASIC, Monitor
Input from Display		↑Z	Read characters from Display instead of Keyboard
Erase		↑E	Erase Display from Cursor to End of Screen
Clear		↑X	Clear the entire Display

[SYM Control Code]

6. It automatically adjusts to 40 character TV mode or 80 character Monitor mode.
7. The program is totally position independent. It may be placed anywhere in memory [except pages zero and one, of course], may be placed in EPROM, may be moved from one of the ASK family micros to another without modification, and does not require any user programming [except for a short startup program on the KIM]. The user sets the A register to a specified value and starts the program at the video initialization address. The keyboard is separately initialized by executing the keyboard initialization routine.
8. Since the KIM does not have INPUT and OUTPUT vectors, a method of having user defined INPUT and OUTPUT locations initialized on the KIM is supported. It is very easy, for example, to use this software with the MICRO-ADE Assembler/Disassembler/Editor package without modification!

ASK VIDEO PLUS Loading Instructions

The ASK VIDEO PLUS tape contains four files recorded in the standard KIM cassette tape format. This format may be loaded by the AIM, SYM or KIM. The AIM or SYM owner can load the tape in KIM format once, and then save it in the higher speed format supported by his AIM or SYM. A KIM owner who has "Hypertape" can save it in this format.

SYNCS: The first file on the tape is 1024 SYNC characters. This may be used, with the suitable program provided with your AIM, SYM, or KIM to adjust your tape recorder to the optimal value for reading the remainder of the tape.

MEMORY TEST [ID 10]: The Memory Test is the same as that listed in the VIDEO PLUS Manual. It loads into 0000 through 00D8 and can be used to test RAM. Put the address of the first page to be tested into 0000; the address of the last page to be tested into 0001; start the program at 0002. If no errors are encountered, then the program will stop with the LED display containing the address following the highest address tested. Otherwise, it will stop on the address with the error. See the VIDEO PLUS Manual or UPDATE 1 for details.

ASK VIDEO PLUS [ID 20]: This loads into 4000 through 4546. This is the space normally allocated to the VIDEO PLUS Display RAM. Set the switch on your VIDEO PLUS to the 4000 setting before loading this program.

AIM: Set A408 to 5A for KIM format tape. Use the K device for input. Filename is 20.

SYM: Use LD1 to load KIM format tape. Program ID is 20.

KIM: Normal load with Program ID of 20. If you wish to force load it to some other address, set 17F5/17F6 to the desired address, cue the tape to this third tape file, set 17F9 to FF. If ID 20 is used, then the program will load into 4000.

The program may **NOT** be run at 4000! This is the display memory. It must be moved to wherever in memory you plan to normally run it. Since the code is totally position independent, where you put it depends on the configuration of your system. If you are not currently using the Programmable Character Generator RAM on the VIDEO PLUS, then this is a handy place. A natural place for the AIM would be to have the VIDEO PLUS set for 8000 [Display RAM] and the program at 9000 [Programmable Character Generator RAM]. A natural place for the SYM would be to have the VIDEO PLUS set for 6000 and the program at 7000. Since the KIM has so much open space, the VIDEO PLUS and program might be placed almost anywhere. The ASK VIDEO PLUS Software does not have to be located in the VIDEO PLUS memory. If it is located in the VIDEO PLUS memory, then the initialization process is a little simpler. If it is located somewhere else, then the user must provide an initialization table which will be described below.

BMOVE [30]: The KIM user can force load the ASK VIDEO PLUS program anywhere he wants using the KIM loader. The SYM user can move the program after loading it, by using the BMOVE command of the SYM Monitor. The AIM user needs some way to move the program. This BMOVE program loads into 0000. Locations 0000/0001 are set to the FROM address, 0002/0003 are set to the TO address, and 0004/0005 are set to the number of bytes to be moved. For example, to move the ASK program from 4000 to 5000, with the program 546 bytes long [4000 through 4545], the values would be:

0000 00, 0001 40, 0002 00, 0003 50, 0004 46, 0005 05.

The program starts at 0006. After running, the ASK program would now be located at 5000 through 5545. The BMOVE program is a useful utility and will work on the AIM, SYM, or KIM.

Once the program has been moved to its "final resting place", it should be saved on tape in your system's high speed format. The KIM can dump it using "Hypertape" [see **MICRO** #1, "Hypertape and Ultratape", pgs 13-16, or **Best of MICRO Volume 1**, pgs 8-11]. The SYM can use the Save 2 command. The AIM can use its normal dump, but must remember to first restore location A408 to C7 and then use T as the output device.

Initializing the ASK VIDEO PLUS Software

AIM: If the ASK program is located in VIDEO PLUS memory, in the PCG RAM at 5000, then the following steps will initialize the video as output device:

1. **A = EA** Make SETUP call Video Init.
2. *** = 5500** Set Program Counter to start of SETUP.
3. **G/** Execute SETUP and VIDEO Initialization

All output that would normally go to the LED display will now appear on the video monitor. **Caution:** Do not put the AIM into Single Step Mode. This will bomb the video program.

You can now use the AIM Monitor, Editor and BASIC with output going to the display. A note about the Editor: the maximum width of a line for the Editor is 60 characters. If you exceed this limit, the additional characters will be lost. They will appear on the screen as you type them, but will not be placed into the Editor buffer. So, be careful. You can change the screen parameters so that the line length is only 60 characters wide. See **COLROW Subroutine.**

If the ASK VIDEO PLUS program is **NOT** in the VIDEO PLUS memory space, then a slightly different procedure is required. Assume for this example that the program is at 2000 and the VIDEO PLUS is selected at 8000:

1. **A = 00** Make SETUP return to Monitor.
2. *** = 2500** Set Program Counter to start of SETUP.
3. **G/** Execute SETUP.
4. **A = 80** Set A to start of VIDEO PLUS Display RAM at 8000.
5. **X = ED** Low address of Initialization TABLE.
6. **Y = 23** High address of Initialization TABLE.
7. *** = 2067** Address of USER Entry to VIDEO Initialization.
8. **G/** Execute USER VIDEO Initialization.

If the standard TABLE is not used, then X and Y must be set to point at an equivalent initialization table. See page 14 of the listing for the TABLE characteristics and values.

If you desire to use an external ASCII keyboard connected to the VIDEO PLUS, and assuming the ASK program is at 5000, the initialization procedure consists of:

1. *** = 5400** Set Program Counter to Keyboard Initialization.
2. **G/** Execute the Keyboard Initialization.

The ASCII keyboard is not setup as the USER Input device. It can not be used with the Monitor or Editor since they go directly to the AIM Keyboard or a TTY. The ASCII keyboard may be used with BASIC. Run BASIC using the normal **5** command. Set Memory Size as desired and Width will default to 60 characters. When BASIC starts, it is getting input from the AIM Keyboard. To switch to the ASCII Keyboard, location A412 must be changed to a "U" or hex 55. The following statement will accomplish this:

POKE 42002,85 Where 42002 = A412 hex and 85 = 55 hex.

Input will now come from the ASCII Keyboard. It will initially be in UPPER case. Use CTRL A to toggle between UPPER and lower case modes. Remember, BASIC commands must be in UPPER case. To return to the AIM Keyboard for input:

POKE 42002,13 Where 42002 = A412 hex and 13 = 0D or CR.

To restore the LED's as the output device, the following program may be used. This simply changes the output vector back to the LED service routine. A similar routine allows switching back to the video for output.

```
0200 A9 05        LDAIM $05    LED SERVICE AT EF05      Set Output to LED's
0202 8D 06 A4     STA   $A406  OUTPUT VECTOR AT A406, A407
0205 A9 EF        LDAIM $EF
0207 8D 07 A4     STA   $A407
020A 00           BRK          RETURN TO MONITOR

020B A9 9F        LDAIM $9F    VIDEO SERVICE AT XX9F    Set Output to VIDEO
020D 8D 06 A4     STA   $A406  LOW ADDRESS OF OUTTV
0210 A9 51        LDAIM $51    HIGH ADDRESS = PROGRAM START
0212 8D 07 A4     STA   $A407  ADDRESS + 1
0215 00           BRK          RETURN TO MONITOR
```

SYM: If the ASK program is located in VIDEO PLUS memory, in the PCG RAM at 5000, then the following steps will initialize the video as output device [**BOLD** represents the user input, ***ITALIC*** represents the SYM output]:

1. **REG CR *P* → *S* →**
   F → *A* EA CR Make SETUP call Video Initialization.
2. **GO 5500 CR** Execute SETUP and VIDEO Initialization

All output that would normally go to the LED display will now appear on the video monitor. You can now use the SYM Monitor, Editor and BASIC with output going to the display.

If the ASK VIDEO PLUS program is **NOT** in the VIDEO PLUS memory space, then a slightly different procedure is required. Assume for this example that the program is at 2000 and the VIDEO PLUS is selected at 6000:

1. **REG CR *P* → *S* →**
   F → *A* 00 CR SETUP returns to Monitor, if A = 00.
2. **GO 2500 CR** Execute ASK SETUP.
3. **REG CR *P* → *S* →**
   F → *A* 60 → Set A to start of VIDEO PLUS Display RAM at 6000.
4. **X ED →** Low address of Initialization TABLE.
5. **Y 23 CR** High address of Initialization TABLE.
6. **GO 2067 CR** Execute at USER Entry to VIDEO Initialization.

If the standard TABLE at 23ED is not used, then X and Y must be set to point at an equivalent initialization table. See page 14 of the listing for the TABLE characteristics and values.

If you desire to use an external ASCII keyboard connected to the VIDEO PLUS, and assuming the ASK program is at 5000, the initialization procedure consists of:

1. **GO 5400 CR** Execute the Keyboard Initialization.

The ASCII keyboard is now setup as the Input device. It can be used with the Monitor, Editor, or BASIC, since they all go through the Input and Output Vectors. Run BASIC by a **GO C000 CR** command. Set Memory Size as desired and Width will default to 60 characters. It will initially be in UPPER case. Use CTRL A to toggle between UPPER and lower case modes. Remember, BASIC commands must be in UPPER case.

To restore the LED's as the output device, the following command may be used. This simply changes the output vector back to the LED service routine. **SD 8900,A664 CR** where 8900 is the address of the standard display output routine, HDOUT, and A664 is the address of OUTVEC. A similar command allows switching back to the video for output. **SD 519F,A664 CR** where 519F is the entry to the video output service, assuming the program starts at 5000.

KIM: If the ASK program is located in VIDEO PLUS memory, in the PCG RAM at 5000, then the following steps will initialize the video as an output device, the keyboard as an input device, will set up a vector that can be used to test whether or not the keyboard has a character present, and will then permit the user to type to the display:

1. Enter the following program anywhere in free memory:

```
0200 A9 EA      INIT   LDAIM $EA    RUN SETUP AND VIDEO INIT
0202 20 00 55          JSR   SETUP   ASSUME PROGRAM AT 5000
0205 20 00 54          JSR   KBINIT  INIT KEYBOARD
0208 20 00 00  IN      JSR   KBTEST  IS THERE ANY DATA PRESENT
020B 90 FB             BCC   IN      WAIT FOR IT. THIS IS NOT REQUIRED
020D 20 00 00          JSR   KBWAIT  GET DATA FROM KEYBOARD
0210 20 00 00          JSR   OUTTV   OUTPUT IT
0213 4C 08 02          JMP   IN      GET MORE

                ACTUAL VALUES OF KBTEST, KBWAIT AND OUTTV
                WILL BE FILLED IN BY INITIALIZATION.
```

2. In address 0000/0001 put the address of the Output Vector, in this example 0211. In address 0002/0003 put the address of the Keyboard Test Vector, in this example 0209. In address 0004/0005 put the address of the Keyboard Input Vector, in this example 020E.

3. Enter address 0200 and press GO. The display will initialize and clear. Whatever is typed will appear on the screen with all of the control functions working: up, down, erase, and so forth. If the program is stopped and the locations pointed to by 0000 through 0005 at initialization time are examined, it will be found that these location now contain the addresses of the ASK routines: 0209/020A have 7D/54, the address of KBTEST [547D]; 020E/020F have 8E/54 [548E], the address of KBWAIT; and 0211/0212 have 9F/51 [519F], the address of OUTTV. These values will of course change if the ASK Software is moved to other locations in memory.

The KIM Monitor does not have any provision for changing its basic input and output vectors. It always gets data from the hexpad or TTY and always sends data to the LEDs or TTY. There is, unfortunately, no way around this, but most available programs do support input and output through vectors which can be set to interact with the ASK VIDEO PLUS Software.

If you are planning to use ASK VIDEO PLUS with an existing program, then find where the I/O is vectored through, and put the address of the Output Vector in 0000/0001, the Keyboard Test Vector (if any) in 0002/0003, and the Keyboard Input Vector in 0004/0005. Run the following program and you should be in business. For example, **MICRO-ADE** has its Output Vector at 2EA1, its Input Vector at 2E9E, and does not have a Keyboard Test Vector. The initial vector pointers would therefore be set:

0000 A1, 0001 2E, 0002 FE, 0003 FF, 0004 9E, 0005 2E

Since there is no Keyboard Test Vector, its pointer was set to FFFE, which being a KIM ROM location can not be modified and will not be adversely affected by the attempt of the Keyboard Initialization to modify it.

If the ASK VIDEO PLUS program is **NOT** in the VIDEO PLUS memory space, then a slightly different procedure is required. Assume for this example that the program is at 2000 and the VIDEO PLUS is selected at 6000:

1. Enter the following program:

```
0200 A9 60      INIT   LDAIM $60    RUN SETUP AND RETURN
0202 20 00 25          JSR   SETUP   ASSUME PROGRAM AT 2000
0205 A9 60             LDAIM $60    DISPLAY RAM PAGE ADDRESS
0207 A2 ED             LDXIM $ED    LOW TABLE ADDRESS
0209 A0 23             LDYIM $23    HIGH TABLE ADDRESS
020B 20 58 00          JSR   TTABLE INIT VIDEO
020E 20 00 54          JSR   KBINIT INIT KEYBOARD
0211 20 00 00  IN      JSR   KBWAIT GET DATA FROM KEYBOARD
0214 20 00 00          JSR   OUTTV  OUTPUT IT
0217 4C 11 02          JMP   IN     GET MORE

                ACTUAL VALUES OF KBTEST, KBWAIT AND OUTTV
                WILL BE FILLED IN BY INITIALIZATION.
```

2. Enter 0200 and GO.

If the standard TABLE at 23ED is not used, then X and Y must be set to point at an equivalent initialization table. See page 14 of the listing for the TABLE characteristics and values.

ASK VIDEO PLUS Program Notes

The following information is, in general, not required for **using** the ASK software, but is useful in **understanding** how it works. It is included to make the total package more useful. One note: all addresses are given as they are in the listing, that is, relative to zero. To find the actual address in a particular configuration, simply add the base address of where the program is residing to the address given. For example, the Initialization Table is listed at 03ED. If the ASK Software is at 5000, then the Table is at 53ED.

1. **Initialization Table [03ED]:** This TABLE contains the information that is required to initialize the ASK Software, in particular that required by the CRT Controller 6845. See the 6845 Data Sheet included with the VIDEO PLUS Manual for additional detail. The TABLE has the following values and functions:

Address	Hex	Dec	Function
03ED	7A	122.	Horizontal Total in Character Time
03EE	50	80.	Horizontal Characters Displayed
03EF	60	96.	Horizontal Sync Position
03F0	0A	10.	Horizontal Sync Width - a fudge factor!

Note: The Horizontal values are divided in half when operating in the TV mode. This is automatically done by the ASK Software and should be taken into account when setting up or modifying an Initialization Table.

03F1	13	19.	Vertical Total - 1 in Character Lines
03F2	1E	30.	Vertical Total Adjust - a fudge factor!
03F3	14	20.	Vertical Lines Displayed
03F4	14	20.	Vertical Sync Position

Note: The Vertical values are not changed for the TV mode.

03F5	00	0.	Scan Mode: Non-interlace.
03F6	0C	12.	Maximum Scan Line Address: 0 - 12 = 13 Scan Lines
03F7	4C	76.	Cursor Blink Rate [40] and Cursor Raster Start [0C]
03F8	0C	12.	Cursor Raster End
03F9	00	0.	Start Address High or Offset into Display Memory initially at zero
03FA	00	0.	Start Address Low
03FB	00	0.	Cursor Address High is initially zero
03FC	00	0.	Cursor Address Low is initially zero

2. **Important Program Locations:** There are several locations in the program that make interacting with it simple. All addresses are as given in the listing.

SETUP [0500]: This routine is used to establish where the ASK VIDEO PLUS Software is currently residing. It sets up a subroutine return [RTS] on page zero, does a subroutine call to this return [JSR], and then pulls the return address off the stack to determine where it is in memory. It then uses this information to calculate the starting address of the ASK Software which is the beginning of the JUMP processor. It puts a vector to the JUMP processor into 0178. If the ASK Software was located at 5000, then the following would be placed into memory: **0178 4C 00 50**
This is a JMP to 5000, the start of the JUMP processor. Similarly, a JMP to the SUBR processor is placed into memory: **017B 4C 05 50.** These two jump vectors are used whenever the resident ASK Software needs to make an internal JMP or JSR. SETUP now determines what it should do next as a function of the value that was in the A register initially. If A was 00, then a BRK is executed. If A was 60, then an RTS is executed. If A was EA, then SETUP transfers directly to the video initialization at TTABLE [0058].

Video Initialization [TTABLE 0058]: There are two ways to initialize the video. The first assumes that the TABLE of initialization values at 03ED is the correct one to use and that the ASK Software is resident somewhere on the VIDEO PLUS board, in RAM or ROM - it doesn't matter. If this is true, then the entry is at 0058. The pointers to the TABLE are corrected using the data in 0179/017A, and the beginning of the Display Memory RAM is calculated from the program address. If the above assumptions are not true, then entry must be made at **USER [0067]**. The A register must contain the Display RAM Page Number, e.g. 40 if the RAM is at 4000; X is the low address of an initialization table [ED if the standard table is to be used]; and Y is the high address of the initialization table [23 if the ASK Software is at 2000]. The initialization routine calculates the CRT Controller address; tests for up to 8 pages [2K] of Display RAM; determines whether the microcomputer is an AIM, SYM, or other [probably KIM]; checks the TV/Monitor jumper; and then initializes the CRT Controller. If the TV switch/jumper is set for TV, then the horizontal values are divided by 2, so that 80 character per line in the table becomes 40 character per line to the controller. A call is made to the **COLROW** subroutine which sets up the row and column limits and calculates the screen size. The output vectors for the AIM, SYM, or other [KIM] are now set to point to **OUTTV**, the ASK Software entry point for video output. The cursor is HOME'd and the screen cleared and control is returned to the user. An AIM or SYM make a BREAK to the Monitor. The KIM [or other] makes an RTS.

JRTN [0000] and SRTN [0005]: All internal JMP's or JSR's are shown in the listing as: **JMP ADDR, NOP, NOP** or **JSR ADDR, NOP, NOP.** This is not what is actually in memory. The code in memory for a JMP is actually: **JSR 0178, ADDR LO, ADDR HI,** where 0178 is the vector to the JUMP processor, and ADDR LO/ADDR HI are the low and high address of the ADDR (or whatever) location relative to the start of the ASK Software. The code for a JSR is identical except that the JSR goes to 017B. The JUMP and SUBR processors use the return pointer on the stack to retrieve the next two bytes of memory which are the relative offset of the desired address. These are added to the starting address of the ASK Software and a JMP is made to the correctly modified address. The only difference between the JUMP and SUBR processors is that the JUMP processor must correct the stack pointer to remove the unwanted JSR return. Examination of the **JRTN [0000]**and **SRTN [0005]**routine listings will provide additional details.

KBINIT [0400]: The video must be initialized before the keyboard. KBINIT first sets up the KBTEST and KBWAIT vectors, storing them in the appropriate vectors for the AIM, SYM or KIM. The VIA 6522 is then initialized to permit the input of data through the I/O port on the VIDEO PLUS. An AIM or SYM return to the Monitor via a BRK; the KIM returns via an RTS.

KBTEST [047D]: When KBTEST is called by the SYM or KIM (there is no provision for a keyboard test on the AIM) it tests the VIA to determine if any data is present. If there is data present, then the carry bit is set. If there is no data present, then the carry bit is cleared. By testing the carry bit upon return from a KBTEST call, the calling program can determine whether or not there is data present on the keyboard.

KBWAIT [408E]: When KBWAIT is called it first determines whether the call was from an AIM. If so, it must then test the carry bit to determine if this is an initialization call or a data call. If the carry pit is clear, then it is an initialization call. Since the initialization has already taken place, no further action is required and a return is made. Otherwise, and always for a SYM or KIM, the keyboard is tested for data. If no data is present, then the test is repeated until data is present. When data is present, it is read in via the VIA. A flag is tested to see if only UPPER case is permitted, or if lower case is permitted as well. If only UPPER case is permitted, the lower case alphabetic characters, "a" to "z" are converted to upper case "A" to "Z". An AIM or KIM then test for the Echo Flag. If it is set, they echo the character via **OUTTV**. If it is not set, they return to the calling routine with the character in the A register. A SYM first tests its own echo flag in TECHO. If its echo flag is set, or if the ASK echo flag is set, then it echos. Otherwise it modifies the return to the SYM by adding 0C to the return address on the stack to skip around the automatic UPPER case conversion routine in the SYM.

OUTTV [019F]: This is the entry point to output a character, or service a control code, to the video. The A, X, and Y registers are saved and a test is made to insure that the cursor is within the screen window. If not, it is restored to the home position. A series of tests are now performed to determine if the character supplied in the A register is a command character. If it is a command character for the microcomputer being run, then it is serviced. If it is not, then it is displayed. All registers are restored before a final return is made. The calling program should **not** try to use the subroutines directly, since the final path always restores the registers and removes one level of stack. It is much easier for the calling program to put the command character in the A register and call OUTTV.

COLROW [0139]: One exception to the above rule is the COLROW subroutine. This routine permits the number of columns and rows on the screen to be easily changed without affecting the various other initialization parameters. The A register contains the column limit and the X register the row limit. COLROW sets these limits and recalculates the screen window.

4. **Control Z Feature:** A command is provided which permits data to be read from the screen by BASIC, Editors, the Monitor, etcetera. This feature only works if input is via the ASCII keyboard through the ASK Keyboard service. The function is simple: whenever a ↑Z is encountered on input from the keyboard, the character at the current cursor position is read, the cursor is incremented to the next position, and the character which was on the screen is passed back to the calling routine. This feature makes is easy to edit lines in BASIC or any other program, by simply moving the cursor to the desired position, "reading" characters from the screen with the ↑Z, typing in new characters wherever desired, and so forth. Try it, you'll like it!

A Final Word

This is the first release of the ASK VIDEO PLUS Software package. While every effort has been made to make it "perfect", I am sure that it contains some mistakes. The program should work without too much difficulty, but may have some bugs. If you find any serious bugs in the program or the documentation, or if you have any suggestions to improve the program or documentation, please be sure that all such input would be appreciated. Send your comments to:

Robert M. Tripp, The COMPUTERIST, Inc., P.O. Box 3, So. Chelmsford, MA 01824.
If a serious problem or misunderstanding arises, you can call me at 617/256-3649.

```
0010:                AIM/SYM/KIM VIDEO PLUS
0020:
0030:                21 DECEMBER 1979
0040:                ROBERT M. TRIPP
0050:
0060:                MODIFIED 11 JANUARY 1980
0070:                BASED ON ROCKWELL INTERNATIONAL APPLICATION NOTE
0080:
0090:                COPYRIGHT (C) 1979 BY:
0100:                  THE COMPUTERIST, INC.
0110:                  P.O. BOX 3
0120:                  SO. CHELMSFORD, MA 01824
0130:                  617/256-3649
0140:
0150:
0160:                PAGE ZERO EQUATES
0170:
0180: 0546           CURSOR *     $00F0
0190: 0546           CRTREG *     $00F2  CRT CONTROLLER/VIA POINTER
0200: 0546           SCRLOW *     $00F4  SCREEN POINTER
0210: 0546           LRT    *     $00F6
0220: 0546           HRT    *     $00F7
0230:
0240:                PAGE ONE EQUATES
0250:
0260: 0546           CURPRM *     $0170  CURSOR POSITION STUFF
0270: 0546           CURPO  *     $0171
0280: 0546           COLMAX *     $0172  COLUMN MAXIMUM
0290: 0546           RAMPAG *     $0173  START OF DISPLAY RAM
0300: 0546           RAMEND *     $0174  END OF DISPLAY RAM
0310: 0546           ASK    *     $0175  AIM/SYM/KIM FLAG
0320: 0546           LSUB   *     $0176
0330: 0546           HSUB   *     $0177
0340: 0546           JUMP   *     $0178
0350: 0546           SUBR   *     $017B
0360: 0546           JFLAG  *     $017E
0370: 0546           XTEMP  *     $017F  TEMPORARY STORAGE
0380: 0546           YTEMP  *     $0180  TEMPORARY STORAGE
0390: 0546           LCHAR  *     $0181  LAST OUTPUT CHARACTER
0400:
0410:                ASK FLAGS
0420:
0430:                0X = AIM
0440:                4X = KIM
0450:                8X = SYM
0460:                X1 = UPPER CASE (0)/LOWER CASE (1)
0470:                X2 = STRIP BIT 80 (0)/PERMIT BIT 80 (1)
0480:                X4 = FULL DUPLEX (0)/HALF DUPLEX (1)
0490:                X8 = NOT AUTO CRLF (0)/AUTO CRLF (1)
0500:
0510:                AIM EQUATES
0520:
0530: 0546           UIN    *     $0108  USER INPUT VECTOR
0540: 0546           CURPOZ *     $A415
0550: 0546           DILINK *     $A406  DISPLAY LINKAGE
0560: 0546           DIBUFF *     $A436  DISPLAY BUFFER
```



```
0570: 0546           COMIN  *     $E1A1  BREAK ENTRY POINT
0580: 0546           DSPVEC *     $EF05  DISPLAY VECTOR VALUE
0590:
0600:                SYM EQUATES
0610:
0620: 0546           ACCESS *     $8B86
0630: 0546           NACCES *     $8B9C  NOT ACCESS
0640: 0546           TECHO  *     $A653  ECHO FLAG
0650: 0546           OUTVEC *     $A663  OUTPUT VECTOR
0660: 0546           INVEC  *     $A660  INPUT VECTOR
0670: 0546           INSVEC *     $A666  TEST INPUT VECTOR
0680:
0690:                KIM EQUATES
0700:
0710: 0546           KOUT   *     $0000  ADDRESS OF KIM OUTPUT VECTOR
0720: 0546           KTST   *     $0002  ADDRESS OF KIM KEYBOARD TEST VECTOR
0730: 0546           KIN    *     $0004  ADDRESS OF KIM INPUT VECTOR
0740:
0750: 0000                 ORG   $0000  RELOCATABLE
0760:
0770:                JUMP SUBROUTINE TO FIX RELOCATABLE JUMPS
0780:
0790:                JSR JUMP
0800:                (OFFSET TO REAL JUMP LOCATION)
0810:
0820: 0000 08        JRTN   PHP          SAVE STATUS
0830: 0001 EE 7E 01        INC   JFLAG   SET JUMP FLAG
0840: 0004 28              PLP          RESTORE STATUS
0850:
0860:                SUBR SUBROUTINE TO FIX RELOCATABLE SUBRS
0870:
0880:                JSR SUBR
0890:                (OFFSET TO REAL SUBR LOCATION)
0900:
0910: 0005 08        SRTN   PHP
0920: 0006 48              PHA          SAVE REGISTERS
0930: 0007 8A              TXA
0940: 0008 48              PHA
0950: 0009 98              TYA
0960: 000A 48              PHA
0970: 000B BA              TSX          GET STACK POINTER
0980: 000C 18              CLC
0990: 000D BD 05 01        LDAX  $0105  GET LOW RETURN ADDRESS - 1
1000: 0010 85 F6           STA   LRT
1010: 0012 BD 06 01        LDAX  $0106  GET HIGH RETURN ADDRESS
1020: 0015 85 F7           STA   HRT
1030:
1040: 0017 A0 01           LDYIM $01    PICKUP LOW OFFSET
1050: 0019 B1 F6           LDAIY LRT
1060: 001B 6D 79 01        ADC   JUMP   +01 LOW OFFSET
1070: 001E 8D 76 01        STA   LSUB
1080: 0021 C8              INY          PICKUP HIGH OFFSET
1090: 0022 B1 F6           LDAIY LRT
1100: 0024 6D 7A 01        ADC   JUMP   +02 HIGH OFFSET
1110: 0027 8D 77 01        STA   HSUB
1120:
```

```
1130: 002A 18             CLC
1140: 002B A5 F6          LDA   LRT     FIX FINAL SUBROUTINE RETURN
1150: 002D 69 02          ADCIM $02     PAST PARAMETERS
1160: 002F 9D 05 01       STAX  $0105   PUT BACK ON STACK
1170: 0032 A5 F7          LDA   HRT     FIX HIGH BYTE
1180: 0034 69 00          ADCIM $00     IN CASE OF CARRY
1190: 0036 9D 06 01       STAX  $0106
1200:
1210: 0039 68             PLA           RESTORE REGISTERS
1220: 003A A8             TAY
1230: 003B 68             PLA
1240: 003C AA             TAX
1250: 003D 68             PLA
1260: 003E CE 7E 01       DEC   JFLAG   TEST JUMP/SUBR
1270: 0041 F0 05          BEQ   JDONE   JUMP
1280: 0043 EE 7E 01       INC   JFLAG   RESTORE FLAG
1290: 0046 F0 0C          BEQ   JSOUT   ALWAYS
1300:
1310: 0048 85 F6    JDONE STA   LRT     MUST CLEANUP STACK
1320: 004A 68             PLA
1330: 004B 85 F7          STA   HRT     STATUS
1340: 004D 68             PLA           LOW
1350: 004E 68             PLA           HIGH
1360: 004F A5 F7          LDA   HRT     STATUS
1370: 0051 48             PHA
1380: 0052 A5 F6          LDA   LRT     A REG
1390: 0054 28       JSOUT PLP           RESTORE STATUS
1400: 0055 6C 76 01       JMI   LSUB
1410:
1420:
ID=11

0010:
0020:               INITIALIZE THE VIDEO OUTPUT ROUTINES
0030:
0040:               THE SETUP CALL WILL COME HERE AUTOMATICALLY IF
0050:               THE A REGISTER IS SET TO $EA BEFORE THE CALL TO
0060:               SETUP.
0070:
0080:               ENTRY POINT IF SOFTWARE IS IN VP ROM
0090:               OR IN PCG RAM ON VIDEO PLUS
0100:
0110: 0058 A9 ED    TTABLE LDAIM TABLE  TABLE RELATIVE TO JRTN
0120: 005A 18              CLC
0130: 005B 6D 79 01        ADC   JUMP   +01
0140: 005E AA              TAX
0150: 005F A9 03           LDAIM TABLE  /
0160: 0061 6D 7A 01        ADC   JUMP   +02
0170: 0064 A8              TAY
0180: 0065 29 30           ANDIM $30    CALC. RAM DISPLAY START
0190:
0200:               ENTRY POINT IF SOFTWARE IS NOT IN VIDEO PLUS
0210:               OR IF USER WANTS TO USE ANOTHER TABLE
0220:
0230:               A = DISPLAY RAM PAGE NUMBER
0240:               X = TABLE ADDRESS LOW
0250:               Y = TABLE ADDRESS HIGH
0260:
```



```
0270: 0067 8D 73 01 USER  STA   RAMPAG SAVE RAM DISPLAY START
0280: 006A 86 F0          STX   CURSOR SET TABLE POINTER LOW
0290: 006C 84 F1          STY   CURSOR +01 AND HIGH IN POINTER
0300: 006E 85 F5          STA   SCRLOW +01 SAVE RAM START
0310: 0070 8D 74 01       STA   RAMEND FOR RAM END TEST
0320: 0073 09 18          ORAIM $18    CALC. CRT ADDRESS
0330: 0075 85 F3          STA   CRTREG +01
0340: 0077 A2 00          LDXIM $00    FIX LOW ADDRESSES
0350: 0079 86 F4          STX   SCRLOW
0360: 007B 86 F2          STX   CRTREG
0370:
0380: 007D A0 08          LDYIM $08    TEST RAM END
0390: 007F A9 00    TLOOP LDAIM $00
0400: 0081 81 F4          STAIX SCRLOW WRITE 00
0410: 0083 A1 F4          LDAIX SCRLOW READ IT BACK
0420: 0085 D0 06          BNE   TDONE  ANYTHING ELSE (FF?)
0430: 0087 EE 74 01       INC   RAMEND BUMP RAM END
0440: 008A 88             DEY
0450: 008B D0 F2          BNE   TLOOP  TRY FOR EIGHT PAGES
0460:
0470: 008D A9 40    TDONE LDAIM $40    SETUP AIM/SYM/KIM FLAG
0480: 008F AE FD FF       LDX   $FFFD  TEST ROM RESET ADDRESS
0490: 0092 E0 8B          CPXIM $8B    SYM ?
0500: 0094 D0 05          BNE   SETAK  NO.
0510: 0096 20 86 8B       JSR   ACCESS SYM
0520: 0099 A9 80          LDAIM $80    SYM FLAG = 80
0530: 009B E0 E0    SETAK CPXIM $E0    AIM ?
0540: 009D D0 02          BNE   SETASK NO
0550: 009F A9 00          LDAIM $00    AIM FLAG = 00
0560:
0570: 00A1 8D 75 01 SETASK STA  ASK    AIM=00/SYM=60/KIM=40
0580: 00A4 B8             CLV
0590: 00A5 A0 04          LDYIM $04    TEST TV OR MONITOR MODE
0600: 00A7 B1 F2          LDAIY CRTREG READ ONBOARD JUMPER
0610: 00A9 4A             LSRA         SHIFT TO TEST
0620: 00AA 4A             LSRA         BIT 2 = 1 FOR TV MODE
0630: 00AB 90 04          BCC   INIT   0 FOR MONITOR
0640: 00AD A9 7F          LDAIM $7F    TV SO SET OVERFLOW FOR
0650: 00AF 69 02          ADCIM $02    TESTING BELOW
0660:
0670: 00B1 A0 00    INIT  LDYIM $00    SET INDEXES
0680: 00B3 A2 00          LDXIM $00
0690: 00B5 98       INITA TYA          NEXT REGISTER IN CRT
0700: 00B6 81 F2          STAIX CRTREG
0710: 00B8 E6 F2          INC   CRTREG POINT TO REAL REGISTER
0720: 00BA B1 F0          LDAIY CURSOR TABLE VALUE
0730: 00BC 50 01          BVC   INITB  TEST TV/MONITOR
0740: 00BE 4A             LSRA         TV SO DIVIDE HORIZONTAL VLAUES
0750: 00BF 81 F2    INITB STAIX CRTREG STORE VALUE
0760: 00C1 C6 F2          DEC   CRTREG POINT TO DUMMY REGISTER
0770: 00C3 C0 01          CPYIM $01    HORZ CHAR PER LINE?
0780: 00C5 D0 01          BNE   INITC
0790: 00C7 48             PHA          SAVE COLMAX
0800: 00C8 C8       INITC INY          BUMP INDEX
0810: 00C9 C0 04          CPYIM $04    TEST DONE WITH HORZ.
0820: 00CB 30 E8          BMI   INITA  NO, MAINTAIN TV TEST
```

```
0830: 00CD B8             CLV          YES, CLEAR TV TEST
0840: 00CE C0 10    INITD CPYIM $10    TEST END OF TABLE
0850: 00D0 D0 E3          BNE  INITA   NO, KEEP GOING
0860: 00D2 A0 06          LDYIM $06
0870: 00D4 B1 F0          LDAIY CURSOR PICK UP ROW MAXIMUM
0880: 00D6 AA             TAX          PUT IN X FOR COLROW PROCESSING
0890: 00D7 68             PLA          GET COLMAX FOR COLROW SUBROUTINE
0900:
0910:               JSR SUBR SUBROUTINE CALL
0920: 00D8 20 39 01       JSR  COLROW SET COL/ROW AND CALCULATE
0930: 00DB EA             NOP          END OF SCREEN
0940: 00DC EA             NOP
0950:
0960:               SET UP TRANSFER VECTORS FROM HOST
0970:               TO VIDEO PLUS.
0980:
0990: 00DD A9 9F          LDAIM OUTTV  CALCULATE OFFSET TO
1000: 00DF 18             CLC          OUTPUT ROUTINE
1010: 00E0 6D 79 01       ADC  JUMP    +01 TO SETUP TRANSFER VECTORS
1020: 00E3 AA             TAX
1030: 00E4 A9 01          LDAIM OUTTV  / PAGE
1040: 00E6 6D 7A 01       ADC  JUMP    +02
1050: 00E9 2C 75 01       BIT  ASK     AIM/SYM/KIM ?
1060: 00EC 70 0A          BVS  INITK   KIM
1070: 00EE 10 13          BPL  INITE   AIM
1080: 00F0 8E 64 A6       STX  OUTVEC +01
1090: 00F3 8D 65 A6       STA  OUTVEC +02
1100: 00F6 30 11          BMI  INITF   ALWAYS
1110:
1120: 00F8 A0 01    INITK LDYIM $01    KIM INIT. STORE
1130: 00FA 91 00          STAIY KOUT   VECTORS VIA POINTERS
1140: 00FC 88             DEY          IN 0000,0001
1150: 00FD 8A             TXA
1160: 00FE 91 00          STAIY KOUT
1170: 0100 B8             CLV          CLEAR OVERFLOW SET BY BIT ASK
1180: 0101 50 06          BVC  INITF   ALWAYS
1190:
1200: 0103 8E 06 A4 INITE STX  DILINK AIM TRANSFER VECTORS
1210: 0106 8D 07 A4       STA  DILINK +01
1220:
1230: 0109 20 0A 03 INITF JSR  HOMECU HOME CURSOR
1240: 010C EA             NOP
1250: 010D EA             NOP
1260: 010E A9 18          LDAIM $18    CLEAR SCREEN COMMAND
1270: 0110 20 9F 01       JSR  OUTTV
1280: 0113 EA             NOP
1290: 0114 EA             NOP
1300:
1310:               CLEAR NEXT UNUSED LINE OF RAM SO
1320:               SCROLL DOESN'T FLASH THIS LINE
1330:
1340: 0115 AD 73 01       LDA  RAMPAG CALC END OF SCREEN
1350: 0118 05 F5          ORA  SCRLOW +01
1360: 011A 85 F5          STA  SCRLOW +01
1370: 011C AC 72 01       LDY  COLMAX
1380: 011F A9 20          LDAIM $20    CLEAR WITH SPACES
```



```
1390: 0121 91 F4    CLR   STAIY SCRLOW
1400: 0123 88             DEY
1410: 0124 10 FB          BPL  CLR
1420: 0126 A5 F5          LDA  SCRLOW +01
1430: 0128 29 0F          ANDIM $0F
1440: 012A 85 F5          STA  SCRLOW +01
1450: 012C 2C 75 01       BIT  ASK     AIM/SYM/KIM ?
1460: 012F 70 06          BVS  FINIS   KIM
1470: 0131 10 03          BPL  FINI    AIM
1480: 0133 20 9C 8B       JSR  NACCES SYM - TURN OFF ACCESS
1490: 0136 00       FINI  BRK          RETURN TO MONITOR
1500:
1510: 0137 B8       FINIS CLV          RETURN TO KIM
1520: 0138 60             RTS
1530:
ID=12


0010:               COLUMN/ROW SET SUBROUTINE
0020:               THIS MAY BE CALLED BY THE USER.
0030:               IT IS CALLED BY THE INITIALIZATION ROUTINES.
0040:
0050:               JSR COLROW
0060:                 A = COLUMN MAXIMUM
0070:                 X = ROW MAXIMUM
0080:
0090:               ON RETURN, COLMAX IS SET AND THE SCREEN END IS
0100:               CALCULATED. THE CRT CONTROLLER IS MODIFIED FOR
0110:               THE NEW COLUMN AND ROW LIMITS
0120:
0130: 0139 8E 7F 01 COLROW STX   XTEMP  X = ROW LIMIT
0140: 013C 8D 72 01       STA   COLMAX A = COLUMN LIMIT
0150: 013F 85 F4          STA   SCRLOW SETUP SCRLOW FOR SNDLOP
0160: 0141 A2 00          LDXIM $00    UPDATE CRT CONTROLLER
0170: 0143 86 F5          STX   SCRLOW +01 SETUP FOR SNDLOP
0180: 0145 A9 01          LDAIM $01    HOR. DISPLAY
0190: 0147 81 F2          STAIX CRTREG REGISTER
0200: 0149 E6 F2          INC   CRTREG
0210: 014B AD 72 01       LDA   COLMAX
0220: 014E 81 F2          STAIX CRTREG
0230: 0150 C6 F2          DEC   CRTREG
0240: 0152 A9 06          LDAIM $06    VERT. DISPLAY
0250: 0154 81 F2          STAIX CRTREG REGISTER
0260: 0156 E6 F2          INC   CRTREG
0270: 0158 AD 7F 01       LDA   XTEMP  ROW MAX
0280: 015B 81 F2          STAIX CRTREG
0290: 015D C6 F2          DEC   CRTREG
0300:
0310:               CALCULATE END OF SCREEN = COLUMNS * ROWS
0320:
0330: 015F CE 7F 01 SNDLOP DEC   XTEMP  ROW COUNTER
0340: 0162 F0 D3          BEQ   FINIS  RETURN
0350: 0164 18             CLC          GET READY TO ADD
0360: 0165 AD 72 01       LDA   COLMAX
0370: 0168 65 F4          ADC   SCRLOW
0380: 016A 85 F4          STA   SCRLOW
0390: 016C 90 F1          BCC   SNDLOP
0400: 016E E6 F5          INC   SCRLOW +01 ADD IN CARRY
0410: 0170 10 ED          BPL   SNDLOP ALWAYS BRANCH
```

```
0420:
ID=13

0010:
0020:                   OUTPUT <CR> OR <LF> TO TV
0030:
0040: 0172 AD 72 01  CRLFTV LDA   COLMAX
0050: 0175 18               CLC           MOVE CURSOR ALL THE WAY RIGHT
0060: 0176 ED 71 01         SBC   CURPO
0070: 0179 18        CRTVO  CLC
0080: 017A 65 F0            ADC   CURSOR CURSOR + (LINMAX - CURPO)
0090: 017C 85 F0            STA   CURSOR
0100: 017E 90 02            BCC   CRTVI
0110: 0180 E6 F1            INC   CURSOR +01
0120: 0182 A9 00     CRTVI  LDAIM $00
0130: 0184 2C 75 01         BIT   ASK     AIM/SYM/KIM ?
0140: 0187 70 05            BVS   CRTVS   KIM
0150: 0189 30 03            BMI   CRTVS   SYM
0160: 018B 8D 15 A4         STA   CURPOZ CLEAR DISPLAY POINTER (AIM)
0170: 018E B8        CRTVS  CLV           CLEAN UP BIT ASK
0180: 018F 8D 70 01         STA   CURPRM
0190: 0192 8D 71 01         STA   CURPO   CLR DISP PNTR
0200: 0195 20 65 03  CURRIG JSR   STCHB
0210: 0198 EA               NOP           STCHB OFFSET
0220: 0199 EA               NOP
0230: 019A 4C 53 02         JMP   ENTCHA
0240: 019D EA               NOP           ENTCHA OFFSET
0250: 019E EA               NOP
0260:
0270:
0280:                   AIM/SYM/KIM OUTPUT VECTORED HERE
0290:
0300:                   ENTRY POINT
0310:                   OUTPUT A CHARACTER TO TV
0320:
0330: 019F 48        OUTTV  PHA           SAVE A
0340: 01A0 B8               CLV           CLEAR OVERFLOW
0350: 01A1 8E 7F 01         STX   XTEMP
0360: 01A4 8C 80 01         STY   YTEMP
0370: 01A7 A4 F1            LDY   CURSOR +01 TEST FOR RANGE
0380: 01A9 CC 74 01         CPY   RAMEND CURSOR BELOW MAXIMUM ?
0390: 01AC B0 5A            BCS   HOMEX   NO. HOME AND RETURN
0400: 01AE CC 73 01         CPY   RAMPAG IS CURSOR ABOVE MINIMUM ?
0410: 01B1 90 55            BCC   HOMEX   NO. HOME CURSOR
0420: 01B3 A8               TAY           SAVE CHARACTER
0430: 01B4 AD 75 01         LDA   ASK     TEST PCG FLAG, BIT 2
0440: 01B7 4A               LSRA
0450: 01B8 4A               LSRA
0460: 01B9 98               TYA           RESTORE CHARACTER
0470: 01BA B0 02            BCS   OUTNXT IF BIT SET, DO NOT STRIP
0480: 01BC 29 7F            ANDIM $7F     IF NOT SET, CONVERT CHARACTER
0490:
0500:                   IT IS A COMMAND, NOW WHICH ONE
0510:
0520: 01BE A8        OUTNXT TAY           AND SAVE
0530: 01BF C9 0D            CMPIM $0D     CARRIAGE RETURN ?
0540: 01C1 F0 AF            BEQ   CRLFTV
0550: 01C3 C9 10            CMPIM $10     CTRL P TOGGLE PCG CHARACTERS
```



```
0560: 01C5 D0 04            BNE   NYY
0570: 01C7 A9 02            LDAIM $02
0580: 01C9 D0 06            BNE   TGL     SETUP TO TOGGLE BIT 2
0590: 01CB C9 01     NYY    CMPIM $01     CTRL A FOR ASCII SWITCH
0600: 01CD D0 0A            BNE   NXX
0610: 01CF A9 01            LDAIM $01     BIT 01 IS ASCII FLAG
0620: 01D1 4D 75 01  TGL    EOR   ASK     TOGGLE UPPER/PCG/ECHO FLAGS
0630: 01D4 8D 75 01         STA   ASK
0640: 01D7 50 25            BVC   HOMAX   ALWAYS
0650: 01D9 C9 06     NXX    CMPIM $06     CTRL F FULL/HALF DUPLEX ECHO
0660: 01DB D0 04            BNE   NAA
0670: 01DD A9 04            LDAIM $04     BIT HEX 4
0680: 01DF D0 F0            BNE   TGL     ALWAYS
0690: 01E1 C9 11     NAA    CMPIM $11     CRTL Q QUASH CRLF
0700: 01E3 D0 04            BNE   NWW
0710: 01E5 A9 08            LDAIM $08     TOGGLE BIT 8
0720: 01E7 D0 E8            BNE   TGL     ALWAYS
0730: 01E9 C9 05     NWW    CMPIM $05     CTRL E ERASE TO END OF SCREEN
0740: 01EB D0 13            BNE   NTT
0750:
0760:                   **** ERASE TO THE END OF SCREEN ****
0770:
0780: 01ED A5 F1     EOS    LDA   CURSOR +01 SAVE CURSOR
0790: 01EF 48               PHA
0800: 01F0 A5 F0            LDA   CURSOR
0810: 01F2 48               PHA
0820: 01F3 20 17 03         JSR   SPACES
0830: 01F6 EA               NOP           SPACES OFFSET
0840: 01F7 EA               NOP
0850: 01F8 68               PLA
0860: 01F9 85 F0            STA   CURSOR RESTORE CURSOR
0870: 01FB 68               PLA
0880: 01FC 85 F1            STA   CURSOR +01
0890: 01FE 50 5C     HOMAX  BVC   HOMA    EXIT ALWAYS
0900:
0910: 0200 C9 18     NTT    CMPIM $18     CTRL X CLEAR SCREEN
0920: 0202 F0 68            BEQ   CLRSCR
0930: 0204 C9 08            CMPIM $08     CTRL H HOME CURSOR
0940: 0206 D0 02            BNE   NZZ
0950: 0208 50 4D     HOMEX  BVC   HOME    TRANSFER TO HOME, ALWAYS
0960: 020A C9 13     NZZ    CMPIM $13     CTRL S SCROLL UP
0970: 020C F0 72            BEQ   SCROLL
0980: 020E C9 12            CMPIM $12     CTRL R CURSOR RIGHT
0990: 0210 F0 66            BEQ   RIGHT
1000: 0212 C9 04            CMPIM $04     CTRL D CURSOR DOWN
1010: 0214 F0 64            BEQ   DOWN
1020: 0216 C9 15            CMPIM $15     CTRL U CURSOR UP
1030: 0218 F0 62            BEQ   UP
1040: 021A C9 0C            CMPIM $0C     CTRL L CURSOR LEFT
1050: 021C F0 60            BEQ   LEFT    TRANSFER TO CURLEF
1060: 021E 2C 75 01         BIT   ASK     AIM/SYM/KIM ?
1070: 0221 50 0E            BVC   TSTAS   AIM OR SYM
1080: 0223 B8        KIMTST CLV           TEST KIM SPECIAL CODES
1090: 0224 C9 7F            CMPIM $7F     DELETE = DELETE
1100: 0226 F0 18            BEQ   KIMDEL
1110: 0228 C9 0A            CMPIM $0A     LINEFEED = DOWN
```

```
1120: 022A F0 09           BEQ   SYMLF
1130: 022C C9 1B           CMPIM $1B    ESCAPE = BREAK
1140: 022E D0 13           BNE   ENTCHR
1150: 0230 00              BRK
1160:
1170: 0231 10 14     TSTAS BPL   AIMTST TEST SPECIAL AIM STUFF
1180: 0233 C9 0A     SYMTST CMPIM $0A    SYM LINEFEED?
1190: 0235 F0 7D     SYMLF BEQ   LFEED  TEST Q FLAG
1200: 0237 C9 02           CMPIM $02    CTRL B BREAK
1210: 0239 D0 01           BNE   SYMX   NO
1220: 023B 00              BRK
1230: 023C C9 5F     SYMX  CMPIM $5F    UNDERLINE = DELETE
1240: 023E D0 0E           BNE   ENTCHR NOT SPECIAL STUFF
1250: 0240 20 95 03  KIMDEL JSR  SYMDEL SYM/KIM DELETE
1260: 0243 EA              NOP
1270: 0244 EA              NOP
1280: 0245 D0 15           BNE   HOMA   ALWAYS
1290:
1300: 0247 C9 1B     AIMTST CMPIM $1B    ESCAPE = BREAK
1310: 0249 D0 03           BNE   ENTCHR NOT SPECIAL
1320: 024B 4C A1 E1        JMP   COMIN  EXIT TO AIM MONITOR
1330:
1340:                **** ENTER A CHR & SCROLL IF NEEDED ****
1350:
1360: 024E 20 21 03  ENTCHR JSR  STORE
1370: 0251 EA              NOP          STORE OFFSET
1380: 0252 EA              NOP
1390: 0253 F0 2B     ENTCHA BEQ   SCROLL SCREEN OVERFLOW SO SCROLL THEM
1400: 0255 D0 05     ENTCHB BNE   HOMA   SCREEN DIDN'T OVERFLOW SO RTN
1410:
1420:                **** HOME CURSOR ****
1430:
1440: 0257 20 0A 03  HOME  JSR   HOMECU
1450: 025A EA              NOP          HOMECU OFFSET
1460: 025B EA              NOP
1470: 025C 20 78 03  HOMA  JSR   TRANSF
1480: 025F EA              NOP          TRANSF OFFSET
1490: 0260 EA              NOP
1500:
1510: 0261 AE 7F 01        LDX   XTEMP  RESTORE X AND Y
1520: 0264 AC 80 01        LDY   YTEMP
1530: 0267 68              PLA
1540: 0268 8D 81 01        STA   LCHAR  SAVE LAST CHAR FOR TESTING
1550: 026B 60              RTS
1560:
1570:                **** CLEAR SCREEN ****
1580:
1590: 026C 20 0A 03  CLRSCR JSR  HOMECU
1600: 026F EA              NOP          HOMECU OFFSET
1610: 0270 EA              NOP
1620: 0271 20 17 03        JSR   SPACES
1630: 0274 EA              NOP          SPACES OFFSET
1640: 0275 EA              NOP
1650: 0276 F0 DF           BEQ   HOME   PUT CURSOR AT TOP AND EXIT
1660:
1670:
TD=14
```



```
0010:
0020: 0278 F0 70     RIGHT BEQ   CURITE TRANSFER TO CURITE
0030: 027A F0 46     DOWN  BEQ   CURDOW TRANSFER TO CURDOW
0040: 027C F0 58     UP    BEQ   CURUP  TRANSFER TO CURUP
0050: 027E F0 7D     LEFT  BEQ   CURLEF TRANSFER TO CURLEF
0060:
0070:                **** SCROLL ONE LINE UP ****
0080:
0090: 0280 20 0A 03  SCROLL JSR  HOMECU
0100: 0283 EA              NOP          HOMECU OFFSET
0110: 0284 EA              NOP
0120: 0285 AC 72 01  SCRA  LDY   COLMAX ADD OFFSET TO INDEX
0130: 0288 B1 F0           LDAIY CURSOR TRANSFER CHARACTERS
0140: 028A 20 61 03        JSR   STCHA
0150: 028D EA              NOP          STCHA OFFSET
0160: 028E EA              NOP
0170: 028F D0 F4           BNE   SCRA
0180: 0291 20 A2 02  SCRB  JSR   SETEND
0190: 0294 EA              NOP          SETEND OFFSET
0200: 0295 EA              NOP
0210: 0296 20 17 03        JSR   SPACES
0220: 0299 EA              NOP          SPACES OFFSET
0230: 029A EA              NOP
0240: 029B 20 A2 02        JSR   SETEND
0250: 029E EA              NOP          SETEND OFFSET
0260: 029F EA              NOP
0270: 02A0 50 BA           BVC   HOMA   FINISH ALWAYS
0280:
0290: 02A2 A5 F4     SETEND LDA   SCRLOW CALCULATE START OF LAST LINE
0300: 02A4 38              SEC
0310: 02A5 ED 72 01        SBC   COLMAX
0320: 02A8 85 F0           STA   CURSOR
0330: 02AA A5 F5           LDA   SCRLOW +01 SCREEN END
0340: 02AC E9 00           SBCIM $00    IN CASE OF CARRY
0350: 02AE 0D 73 01        ORA   RAMPAG
0360: 02B1 85 F1           STA   CURSOR +01
0370: 02B3 60              RTS
0380:
0390:                **** MOVE CURSOR DOWN ****
0400:
0410: 02B4 A9 08     LFEED LDAIM $08    TEST Q FLAG
0420: 02B6 2D 75 01        AND   ASK
0430: 02B9 F0 07           BEQ   CURDOW NOT SET
0440: 02BB AD 81 01        LDA   LCHAR  TEST PRIOR CR
0450: 02BE C9 0D           CMPIM $0D    TO IGNORE EXTRA LF
0460: 02C0 F0 9A           BEQ   HOMA   IF YES, EXIT
0470:
0480: 02C2 A5 F0     CURDOW LDA   CURSOR ADD LINMAX TO CURSOR
0490: 02C4 18              CLC
0500: 02C5 6D 72 01        ADC   COLMAX MAXIMUM CHARACTERS PER LINE
0510: 02C8 85 F0           STA   CURSOR
0520: 02CA 90 05           BCC   CURDA
0530: 02CC 20 69 03        JSR   STCHC
0540: 02CF EA              NOP          STCHC OFFSET
0550: 02D0 EA              NOP
0560: 02D1 4C 53 02  CURDA JMP   ENTCHA
```

```
0570: 02D4 EA              NOP          ENTCHA OFFSET
0580: 02D5 EA              NOP
0590:
0600:                 **** MOVE CURSOR UP ****
0610:
0620: 02D6 A5 F0      CURUP  LDA   CURSOR SUB LINMAX TO CURSOR
0630: 02D8 38                SEC
0640: 02D9 ED 72 01          SBC   COLMAX MOVE UP ONE LINE
0650: 02DC 85 F0             STA   CURSOR
0660: 02DE B0 25             BCS   HOMAXX UNDER FLOW OF PAGE
0670: 02E0 20 B2 03          JSR   DECRA
0680: 02E3 EA                NOP          DECRA
0690: 02E4 EA                NOP
0700: 02E5 4C 55 02          JMP   ENTCHB
0710: 02E8 EA                NOP          ENTCHB OFFSET
0720: 02E9 EA                NOP
0730:
0740:                 **** MOVE CURSOR TO THE RIGHT ****
0750:
0760: 02EA AC 71 01   CURITE LDY   CURPO  FIX UP CURSOR POSITION
0770: 02ED C8                INY
0780: 02EE CC 72 01          CPY   COLMAX TEST LIMIT
0790: 02F1 90 02             BCC   CRX    NOT EXCEEDED
0800: 02F3 A0 00             LDYIM $00    RESET IF EXCEEDED
0810: 02F5 8C 71 01   CRX    STY   CURPO  STORE NEW VALUE
0820: 02F8 4C 95 01          JMP   CURRIG AND FINISH UP
0830: 02FB EA                NOP
0840: 02FC EA                NOP
0850:
0860:                 **** MOVE CURSOR TO THE LEFT ****
0870:
0880: 02FD 20 DA 03   CURLEF JSR   DECREM
0890: 0300 EA                NOP          DECREM
0900: 0301 EA                NOP
0910: 0302 CE 71 01          DEC   CURPO  FIX CURSOR POSITION
0920: 0305 4C 5C 02   HOMAXX JMP   HOMA   TRANSFER TO HOMA
0930: 0308 EA                NOP          ENTCHB OFFSET
0940: 0309 EA                NOP
0950:
0960:
ID=15

0010:                 SUBROUTINES
0020:
0030: 030A A9 00      HOMECU LDAIM $00     SET CURSOR TO BEGINNING
0040: 030C 85 F0             STA   CURSOR
0050: 030E 8D 71 01          STA   CURPO  CLR DISP PNTR
0060: 0311 AD 73 01          LDA   RAMPAG
0070: 0314 85 F1             STA   CURSOR +01
0080: 0316 60                RTS
0090:
0100: 0317 A9 20      SPACES LDAIM $20     PUT BLANKS
0110: 0319 20 61 03          JSR   STCHA
0120: 031C EA                NOP          STCHA OFFSET
0130: 031D EA                NOP
0140: 031E D0 F7             BNE   SPACES
0150: 0320 60                RTS
0160:
```



```
0170:                 STORE ROUTINE FIRST TESTS FOR AIM OR SYM
0180:
0190: 0321 2C 75 01   STORE  BIT   ASK    AIM/SYM/KIM
0200: 0324 70 2C             BVS   STCHAA KIM
0210: 0326 30 2A             BMI   STCHAA SYM IGNORES AIM DELETE STUFF
0220:
0230:                 FIRST CHECK IF DELETE WAS KEYED.
0240:                 IF SC, A JSR PSLS (DELETE SUBROUTINE) LOCATIONS
0250:                 FROM STACK POINTER PLUS 4,5 SHOULD BE $E7F2
0260:
0270: 0328 BA                TSX          GET STACK POINTER
0280: 0329 BD 05 01          LDAX  $0105  LOOK 5 PLACES UP FROM STACK POINTER
0290: 032C C9 E7             CMPIM $E7
0300: 032E D0 07             BNE   STCHR  IT IS NOT A DELETE
0310: 0330 BD 04 01          LDAX  $0104  CHECK LOWER HALF OF ADDRESS
0320: 0333 C9 F2             CMPIM $F2
0330: 0335 F0 62             BEQ   DELETE YES, DELETE ONE CHAR
0340:
0350:                 STORE A CHAR IN RAM (4XXX) AND CHECK FOR CURSOR
0360:
0370: 0337 98         STCHR  TYA           CHAR WAS SAVED IN Y
0380:
0390:                 WRAP CURPOZ AROUND 20 TO BE ABLE TO
0400:                 RECEIVE DELETES
0410:
0420: 0338 AC 15 A4           LDY   CURPOZ DON'T LET CURPOZ >= 20 CHR
0430: 033B C8                 INY
0440: 033C C0 14              CPYIM $14    CURPOZ >= 20 ?
0450: 033E 90 02              BCC   XXA    YES, INCR CURPOZ
0460: 0340 A0 13              LDYIM $13    NO, RESET TO 19
0470: 0342 8C 15 A4   XXA     STY   CURPOZ
0480:
0490:                 MAINTAIN DISPLAY BUFFER FOR EDITOR
0500:
0510: 0345 AC 70 01           LDY   CURPRM IF > 60 DON'T PUT IN ON DISBUFFER
0520: 0348 C0 3C              CPYIM $3C
0530: 034A B0 06              BCS   STCHAA
0540: 034C 99 38 A4           STAY  DIBUFF EDITOR & M-COMMAND USE THIS
0550: 034F EE 70 01           INC   CURPRM BUFFER
0560:
0570:                 WRAP AROUND LINMAX FOR CRT (START NEW LINE)
0580:
0590: 0352 B8         STCHAA CLV           CLEAR FROM BIT ASK TEST
0600: 0353 AC 71 01           LDY   CURPO  NO. OF CHAR = LINMAX?
0610: 0356 C8                 INY
0620: 0357 CC 72 01           CPY   COLMAX CURPO >= MAX CHARACTERS
0630: 035A 90 02              BCC   XXB    NO, INCR CURPO
0640: 035C A0 00              LDYIM $00    YES, RESET TO ZERO
0650: 035E 8C 71 01   XXB     STY   CURPO
0660: 0361 A0 00      STCHA   LDYIM $00
0670: 0363 91 F0              STAIY CURSOR STORE INTO DISPLAY RAM
0680: 0365 E6 F0      STCHB   INC   CURSOR INCR CURSOR
0690: 0367 D0 02              BNE   STCHD
0700: 0369 E6 F1      STCHC   INC   CURSOR +01
0710: 036B A5 F0      STCHD   LDA   CURSOR SEE IF > SCRMAX
0720: 036D C5 F4              CMP   SCRLOW LOW HALF
```

```
0730: 036F D0 06          BNE   STCHE
0740: 0371 A5 F1          LDA   CURSOR +01
0750: 0373 29 0F          ANDIM $0F
0760: 0375 C5 F5          CMP   SCRLOW +01 HIGH HALF
0770: 0377 60       STCHE RTS
0780:
0790:                     TRANSFER CURSOR TO ACTUAL CURSOR IN
0800:                     6845 AND TO AIM CURSOR
0810:
0820: 0378 A0 00    TRANSF LDYIM $00    SET INDEX
0830: 037A A9 0E          LDAIM $0E    CURSOR HIGH
0840: 037C 91 F2          STAIY CRTREG SETUP 6845
0850: 037E E6 F2          INC   CRTREG
0860: 0380 A5 F1          LDA   CURSOR +01
0870: 0382 29 0F          ANDIM $0F
0880: 0384 91 F2          STAIY CRTREG
0890: 0386 C6 F2          DEC   CRTREG
0900: 0388 A9 0F          LDAIM $0F    CURSOR LOW
0910: 038A 91 F2          STAIY CRTREG
0920: 038C E6 F2          INC   CRTREG
0930: 038E A5 F0          LDA   CURSOR
0940: 0390 91 F2          STAIY CRTREG
0950: 0392 C6 F2          DEC   CRTREG
0960: 0394 60             RTS
0970:
0980: 0395 A0 20    SYMDEL LDYIM $20    SPACE CHARACTER
0990: 0397 D0 0F          BNE   SDEL   SKIP SOME AIM STUFF
1000:
1010: 0399 BD 02 01 DELETE LDAX  $0102  DON'T DECR BEYOND ZERO
1020: 039C C9 14          CMPIM $14    Y REG >= 20 ?
1030: 039E 90 02          BCC   XXC    NO, RESET CURPO TO THAT VALUE
1040: 03A0 A9 13          LDAIM $13    YES, RESET TO 19 TO SEE DELETES
1050: 03A2 8D 15 A4 XXC   STA   CURPOZ
1060: 03A5 CE 70 01       DEC   CURPRM DECR OTHER POINTER
1070: 03A8 CE 71 01 SDEL  DEC   CURPO  WRAP AROUND ZERO
1080: 03AB 10 09          BPL   DELA
1090: 03AD AD 72 01       LDA   COLMAX RESET
1100: 03B0 8D 71 01       STA   CURPO
1110: 03B3 CE 71 01       DEC   CURPO  SET TO COLMAX - 1
1120: 03B6 2C 75 01 DELA  BIT   ASK    AIM/SYM/KIM
1130: 03B9 70 12          BVS   SDELA  KIM
1140: 03BB 30 10          BMI   SDELA  SYM
1150: 03BD A9 E8          LDAIM $E8    STORE NEW RTN ADDRESS FOR OUTDP1 - AIM
1160: 03BF 9D 05 01       STAX  $0105
1170: 03C2 A9 04          LDAIM $04    RTN TO PSL00+3 WITH NEW POINTER
1180: 03C4 9D 04 01       STAX  $0104
1190: 03C7 AD 70 01       LDA   CURPRM NEW POINTER TO SAVED ACC
1200: 03CA 9D 03 01       STAX  $0103
1210: 03CD B8       SDELA CLV          CLEAR FROM BIT ASK
1220: 03CE 20 DA 03       JSR   DECREM
1230: 03D1 EA             NOP          DECREM OFFSET
1240: 03D2 EA             NOP
1250: 03D3 98             TYA          CLEAR LAST CHAR
1260: 03D4 A0 00          LDYIM $00
1270: 03D6 91 F0          STAIY CURSOR
1280: 03D8 C8             INY          SET Z FLAG TO 1
```



```
1290: 03D9 60             RTS
1300:
1310: 03DA C6 F0    DECREM DEC   CURSOR
1320: 03DC A5 F0           LDA   CURSOR
1330: 03DE C9 FF           CMPIM $FF    WAS IT 00 LAST TIME ?
1340: 03E0 D0 02           BNE   DECRB
1350: 03E2 C6 F1    DECRA  DEC   CURSOR +01
1360: 03E4 AD 73 01 DECRB  LDA   RAMPAG CURSOR < RAM ?
1370: 03E7 38              SEC
1380: 03E8 E9 01           SBCIM $01
1390: 03EA C5 F1           CMP   CURSOR +01
1400: 03EC 60              RTS
1410:
1420: 03ED 7A       TABLE  =     $7A    H TOTAL
1430: 03EE 50              =     $50    H DISPLAYED ( 80 CHARACTERS)
1440: 03EF 60              =     $60    H SYNC POSITION
1450: 03F0 0A              =     $0A    H SYNC WIDTH
1460: 03F1 13              =     $13    V TOTAL
1470: 03F2 1E              =     $1E    V TOTAL ADJUST
1480: 03F3 14              =     $14    V DISPLAYED
1490: 03F4 14              =     $14    V SYNC POSITION
1500: 03F5 00              =     $00    NON INTERLACE
1510: 03F6 0C              =     $0C    MAX SCAN LINE ADDRESS
1520: 03F7 4C              =     $4C    CURSOR RASTER START
1530: 03F8 0C              =     $0C    CURSOR RASTER END
1540: 03F9 00              =     $00    START ADDRESS HI
1550: 03FA 00              =     $00    START ADDRESS LO
1560: 03FB 00              =     $00    CURSOR HI
1570: 03FC 00              =     $00    CURSOR LO
1580:
1590:
ID=
```

```
0010:
0020: 0400                    ORG   $0400  NOT REQUIRED FOR OUTPUT
0030:
0040: 0400 A9 7D     KBINIT LDAIM KBTEST SETUP INSVEC
0050: 0402 18                 CLC
0060: 0403 6D 79 01           ADC   JUMP   +01 OFFSET
0070: 0406 AA                 TAX
0080: 0407 A9 04              LDAIM KBTEST / PAGE
0090: 0409 6D 7A 01           ADC   JUMP   +02
0100: 040C 2C 75 01           BIT   ASK    AIM/SYM/KIM ?
0110: 040F 70 0D              BVS   KIMA   KIM
0120: 0411 10 14              BPL   KBDA   AIM
0130: 0413 20 86 8B           JSR   ACCESS SYM
0140: 0416 8E 67 A6           STX   INSVEC +01
0150: 0419 8D 68 A6           STA   INSVEC +02
0160: 041C 50 09              BVC   KBDA   ALWAYS
0170:
0180: 041E B8        KIMA     CLV
0190: 041F A0 01              LDYIM $01    STORE KEYBOARD TEST VECTOR
0200: 0421 91 02              STAIY KTST   VIA TEMPORARY PAGE ZERO POINTER
0210: 0423 88                 DEY
0220: 0424 8A                 TXA
0230: 0425 91 02              STAIY KTST   STORE LOW HALF OF ADDRESS
0240:
0250: 0427 A9 8E     KBDA     LDAIM KBWAIT SETUP INPUT VECTOR
0260: 0429 18                 CLC
0270: 042A 6D 79 01           ADC   JUMP   +01
0280: 042D AA                 TAX          SAVE LOW ADDRESS
0290: 042E A9 04              LDAIM KBWAIT /
0300: 0430 6D 7A 01           ADC   JUMP   +02
0310: 0433 2C 75 01           BIT   ASK    AIM/SYM/KIM ?
0320: 0436 70 0A              BVS   KIMB   KIM
0330: 0438 10 12              BPL   KBDB   AIM
0340: 043A 8E 61 A6           STX   INVEC  +01 LOW VECTOR
0350: 043D 8D 62 A6           STA   INVEC  +02 HIGH VECTOR FOR SYM
0360: 0440 D0 10              BNE   KBDC   ALWAYS
0370:
0380: 0442 B8        KIMB     CLV
0390: 0443 C8                 INY          STORE HIGH HALF OF KEYBOARD
0400: 0444 91 04              STAIY KIN    ADDRESS VIA TEMP. KIN POINTER
0410: 0446 88                 DEY
0420: 0447 8A                 TXA
0430: 0448 91 04              STAIY KIN    STORE HIGH HALF
0440: 044A 50 06              BVC   KBDC   ALWAYS
0450: 044C 8E 06 01  KBDB     STX   UIN    LOW VECTOR FOR AIM
0460: 044F 8D 09 01           STA   UIN    +01 HIGH VECTOR
0470: 0452 A9 7F     KBDC     LDAIM $7F    DISABLE ALL INTERRUPTS
0480: 0454 A0 1E              LDYIM $1E    IER OFFSET
0490: 0456 91 F2              STAIY CRTREG RELATIVE TO CRT CONTROLLER
0500: 0458 A9 FF              LDAIM $FF    CLEAR ANY INTERRUPTS PENDING
0510: 045A A0 1D              LDYIM $1D    IFR OFFSET
0520: 045C 91 F2              STAIY CRTREG
0530: 045E A9 00              LDAIM $00    SET DATA DIRECTION FOR INPUT
0540: 0460 A0 13              LDYIM $13    DATA DIRECTION
0550: 0462 91 F2              STAIY CRTREG
0560: 0464 A9 01              LDAIM $01    SET TO LATCH KEYBOARD DATA
```



```
0570: 0466 A0 1B              LDYIM $1B    ACR
0580: 0468 91 F2              STAIY CRTREG LATCH KEYBOARD DATA
0590: 046A A9 C6              LDAIM $C6    KBRD CA1 SET ON POSITIVE TRANS.
0600: 046C A0 1C              LDYIM $1C    PCR
0610: 046E 91 F2              STAIY CRTREG KBRD POS, SUPRESS CONTROL CODES
0620: 0470 AD 75 01           LDA   ASK    AIM/SYM/KIM
0630: 0473 10 03              BPL   INDONE AIM/KIM
0640: 0475 20 9C 8B           JSR   NACCES SYM
0650: 0478 29 40     INDONE   ANDIM $40    AIM/SYM/KIM
0660: 047A D0 11              BNE   KMDONE KIM
0670: 047C 00                 BRK          AIM OR SYM
0680:
0690: 047D 8C 80 01  KBTEST   STY   YTEMP  SAVE Y
0700: 0480 18                 CLC          C = 0 FOR NO DATA
0710: 0481 A0 1D              LDYIM $1D    READ IFR
0720: 0483 B1 F2              LDAIY CRTREG
0730: 0485 29 02              ANDIM $02    MASK TO CA1 FLAG
0740: 0487 F0 01              BEQ   NODATA IF ZERO, THEN NO DATA
0750: 0489 38                 SEC          ELSE, SET C = 1 FOR DATA
0760: 048A AC 80 01  NODATA   LDY   YTEMP
0770: 048D 60        KMDONE   RTS
0780:
0790: 048E 8C 80 01  KBWAIT   STY   YTEMP  SAVE Y
0800: 0491 2C 75 01           BIT   ASK    AIM/SYM/KIM ?
0810: 0494 70 04              BVS   KBGET  KIM, SO KBGET
0820: 0496 30 02              BMI   KBGET  CHECK AIM INIT CALL
0830: 0498 90 F0              BCC   NODATA RETURN ON INIT CALL
0840: 049A B8        KBGET    CLV
0850: 049B A0 1D              LDYIM $1D    TEST DATA PRESENT
0860: 049D B1 F2              LDAIY CRTREG
0870: 049F 29 02              ANDIM $02
0880: 04A1 F0 F7              BEQ   KBGET  WAIT FOR IT
0890: 04A3 A0 11              LDYIM $11    READ DATA
0900: 04A5 B1 F2              LDAIY CRTREG
0910: 04A7 A8                 TAY          SAVE CHARACTER
0920:
0930: 04A8 C9 1A              CMPIM $1A    CTRL Z ?
0940: 04AA D0 0B              BNE   KNORM  NO
0950: 04AC A0 00              LDYIM $00    READ FROM CURRENT CURSOR
0960: 04AE B1 F0              LDAIY CURSOR POSITION
0970: 04B0 AC 75 01           LDY   ASK    AIM/SYM/KIM RETURN ?
0980: 04B3 10 3D              BPL   KIMOUT AIM OR KIM WITHOUT ECHO
0990: 04B5 30 33              BMI   UOUT   SYM WITHOUT ECHO
1000:
1010: 04B7 AD 75 01  KNORM    LDA   ASK    TEST UPPER/LOWER ASCII FLAG
1020: 04BA 4A                 LSRA
1030: 04BB 98                 TYA          RESTORE CHARACTER
1040: 04BC B0 0A              BCS   NOTUP  IF SET, NOT UPPER ONLY
1050: 04BE C9 61              CMPIM $61    UPPER CASE ONLY
1060: 04C0 90 06              BCC   NOTUP  NOT LOWER CASE ALPHA
1070: 04C2 C9 7B              CMPIM $7B
1080: 04C4 B0 02              BCS   NOTUP
1090: 04C6 29 DF              ANDIM $DF    CONVERT LOWER TO UPPER CASE
1100:
1110: 04C8 AC 75 01  NOTUP    LDY   ASK    AIM/SYM/KIM ?
1120: 04CB 10 06              BPL   AUOUT  AIM OR KIM
```

```
1130: 04CD 2C 53 A6        BIT   TECHO  ECHO FLAG IN SYM
1140: 04D0 B8              CLV          JUST IN CASE
1150: 04D1 30 09           BMI   ECHO   YES, ECHO BIT SET
1160: 04D3 A8        AUCUT TAY          SAVE CHARACTER
1170: 04D4 A9 04           LDAIM $04    TEST ECHO FLAG
1180: 04D6 2D 75 01        AND   ASK    TOGGLED BY CTRL F
1190: 04D9 F0 0A           BEQ   ALLTST NO ECHO IF ZERO
1200: 04DB 98              TYA          RESTORE CHARACTER
1210: 04DC 20 9F 01  ECHO  JSR   OUTTV  ECHO TO VIDEO
1220: 04DF EA              NOP
1230: 04E0 EA              NOP
1240: 04E1 C0 06           CPYIM $06    WAS IT A CTRL F TO TURN OFF ECHO?
1250: 04E3 F0 B5           BEQ   KBGET  IF SO, DO NOT ECHO TO USER
1260: 04E5 AC 75 01  ALLTST LDY  ASK    AIM/KIM
1270: 04E8 10 07           BPL   ALLOUT AIM/KIM
1280:
1290: 04EA A8        UCUT  TAY          SAVE CHARACTER
1300: 04EB 68              PLA          MODIFY RETURN TO AVOID THE
1310: 04EC 18              CLC          AUTOMATIC LOWER CASE TO UPPER CASE
1320: 04ED 69 0C           ADCIM $0C    CONVERSION
1330: 04EF 48              PHA          NEW RETURN ADDRESS
1340: 04F0 38              SEC          SET CARRY FOR DATA
1350:
1360: 04F1 98        ALLOUT TYA         RESTORE CHARACTER
1370: 04F2 B8        KIMOUT CLV
1380: 04F3 AC 80 01        LDY   YTEMP
1390: 04F6 60              RTS
1400:
1410:
ID=

-
```



```
0010:
0020: 0500                 ORG   $0500
0030:
0040:               THIS CODE DOES NOT HAVE TO BE RESIDENT
0050:               AFTER IT IS INITIALLY RUN
0060:
0070:               THIS SETUP PROGRAM MUST BE RUN FIRST TO GET
0080:               THE RELOCATION FACILITY SETUP.
0090:
0100:               A REGISTER MUST CONTAIN RETURN INFO:
0110:
0120:                 BREAK AT END OF SETUP  = 00
0130:                 RTS AT END OF SETUP    = 60
0140:                 INIT VIDEO AFTER SETUP = EA
0150:
0160:
0170: 0500 48        SETUP PHA          SAVE RETURN INFO IN A REG.
0180: 0501 A9 00           LDAIM $00    CLEAR ALL STATUS FLAGS
0190: 0503 48              PHA
0200: 0504 28              PLP
0210: 0505 A9 60           LDAIM $60    MAKE A SUBROUTINE RETURN
0220: 0507 85 F6           STA   LRT    CREATE SUBROUTINE RETURN
0230: 0509 20 F6 00        JSR   LRT    GO AND RETURN
0240: 050C BA              TSX          STACK POINTER
0250: 050D CA              DEX          PUSH STACK POINTER BACK DOWN TO
0260: 050E CA              DEX          THE RETURN ADDRESSES
0270: 050F 9A              TXS
0280: 0510 68              PLA          LOW RETURN ADDRESS
0290: 0511 38              SEC          FIX UP POINTERS
0300: 0512 E9 0B           SBCIM SETUP  +0B CORRECT POINTER
0310: 0514 8D 79 01        STA   JUMP   +01 STORE LOW
0320: 0517 68              PLA          HIGH ADDRESS OF JUMP
0330: 0518 E9 05           SBCIM SETUP  / PAGE OFFSET
0340: 051A 48              PHA          SAVE
0350: 051B 8D 7A 01        STA   JUMP   +02
0360:
0370: 051E AD 79 01        LDA   JUMP   +01 NOW SETUP SUBROUTINE SERVICE
0380: 0521 18              CLC
0390: 0522 69 05           ADCIM $05    FIVE BYTES BEYOND JUMP
0400: 0524 8D 7C 01        STA   SUBR   +01
0410: 0527 68              PLA          HIGH
0420: 0528 69 00           ADCIM $00    CARRY IF ANY
0430: 052A 8D 7D 01        STA   SUBR   +02
0440: 052D A9 4C           LDAIM $4C    JMP COMMAND
0450: 052F 8D 78 01        STA   JUMP
0460: 0532 8D 7B 01        STA   SUBR
0470: 0535 A9 00           LDAIM $00    INIT JFLAG
0480: 0537 8D 7E 01        STA   JFLAG  SHOULD ALWAYS BE ZERO
0490:
0500:               DETERMINE THE METHOD OF RETURNING AFTER SETUP
0510:
0520: 053A 68              PLA          A REG HAD INFO ON ENTRY
0530: 053B F0 03           BEQ   BREAK  00 = BRK
0540: 053D 30 02           BMI   VIDEO  EA = INIT VIDEO
0550: 053F 60              RTS          60 = RETURN
0560: 0540 00        BREAK BRK
```

```
0570:
0580:                IF SETUP WAS ENTERED WITH A = EA, THEN
0590:                DO THE SETUP AND THEN INIT THE VIDEO
0600:
0610: 0541 4C 58 00  VIDEO  JMP   TTABLE START WITH PROGRAM IN VIDEO MEMORY
0620: 0544 EA               NOP
0630: 0545 EA               NOP
0640:
0650:
ID=
```

T1

SYMBOL TABLE 2000 237E

KOUT	0000	JRTN	0000	KTST	0002	KIN	0004
SRTN	0005	JDONE	0048	JSOUT	0054	TTABLE	0058
USER	0067	TLOOP	007F	TDONE	008D	SETAK	009B
SETASK	00A1	INIT	00B1	INITA	00B5	INITB	00BF
INITC	00C8	INITD	00CE	CURSOR	00F0	CRTREG	00F2
SCRLOW	00F4	LRT	00F6	HRT	00F7	INITK	00F8
INITE	0103	UIN	0108	INITF	0109	CLR	0121
FINI	0136	FINIS	0137	COLROW	0139	SNDLOP	015F
CURPRM	0170	CURPO	0171	COLMAX	0172	CRLFTV	0172
RAMPAG	0173	RAMEND	0174	ASK	0175	LSUB	0176
HSUB	0177	JUMP	0178	CRTVO	0179	SUBR	017B
JFLAG	017E	XTEMP	017F	YTEMP	0180	LCHAR	0181
CRTVI	0182	CRTVS	018E	CURRIG	0195	OUTTV	019F
OUTNXT	01BE	NYY	01CB	TGL	01D1	NXX	01D9
NAA	01E1	NWW	01E9	EOS	01ED	HOMAX	01FE
NTT	0200	HOMEX	0208	NZZ	020A	KIMTST	0223
TSTAS	0231	SYMTST	0233	SYMLF	0235	SYMX	023C
KIMDEL	0240	AIMTST	0247	ENTCHR	024E	ENTCHA	0253
ENTCHB	0255	HOME	0257	HOMA	025C	CLRSCR	026C
RIGHT	0278	DOWN	027A	UP	027C	LEFT	027E
SCROLL	0280	SCRA	0285	SCRB	0291	SETEND	02A2
LFEED	02B4	CURDOW	02C2	CURDA	02D1	CURUP	02D6
CURITE	02EA	CRX	02F5	CURLEF	02FD	HOMAXX	0305
HOMECU	030A	SPACES	0317	STORE	0321	STCHR	0337
XXA	0342	STCHAA	0352	XXB	035E	STCHA	0361
STCHB	0365	STCHC	0369	STCHD	036B	STCHE	0377
TRANSF	0378	SYMDEL	0395	DELETE	0399	XXC	03A2
SDEL	03A8	DELA	03B6	SDELA	03CD	DECREM	03DA
DECRA	03E2	DECRB	03E4	TABLE	03ED	KBINIT	0400
KIMA	041E	KBDA	0427	KIMB	0442	KBDB	044C
KBDC	0452	INDONE	0478	KBTEST	047D	NODATA	048A
KMDONE	048D	KBWAIT	048E	KBGET	049A	KNORM	04B7
NOTUP	04C8	AUOUT	04D3	ECHO	04DC	ALLTST	04E5
UOUT	04EA	ALLOUT	04F1	KIMOUT	04F2	SETUP	0500
BREAK	0540	VIDEO	0541	ACCESS	8B86	NACCES	8B9C
DILINK	A406	CURPOZ	A415	DIBUFF	A438	TECHO	A653
INVEC	A660	OUTVEC	A663	INSVEC	A666	COMIN	E1A1
DSPVEC	EF05						

T

SYMBOL TABLE 2000 237E

ACCESS	6B66	AIMTST	0247	ALLOUT	04F1	ALLTST	04E5
ASK	0175	AUOUT	04D3	BREAK	0540	CLRSCR	026C
CLR	0121	COLMAX	0172	COLROW	0139	COMIN	E1A1
CRLFTV	0172	CRTREG	00F2	CRTVI	0182	CRTVO	0179
CRTVS	018E	CRX	02F5	CURDA	02D1	CURDOW	02C2
CURITE	02EA	CURLEF	02FD	CURPO	0171	CURPOZ	A415
CURPRM	0170	CURRIG	0195	CURSOR	00F0	CURUP	02D6
DECRA	03E2	DECRB	03E4	DECREM	03DA	DELA	03B6
DELETE	0399	DIBUFF	A438	DILINK	A406	DOWN	027A
DSPVEC	BF05	ECHO	04DC	ENTCHA	0253	ENTCHB	0255
ENTCHR	024E	EOS	01ED	FINI	0136	FINIS	0137
HOMA	025C	HOMAX	01FE	HOMAXX	0305	HOME	0257
HOMECU	030A	HOMEX	0208	HRT	00F7	HSUB	0177
INDONE	0478	INIT	00B1	INITA	00B5	INITB	00BF
INITC	00C8	INITD	00CE	INITE	0103	INITF	0109
INITK	00F8	INSVEC	A666	INVEC	A660	JDONE	0048
JFLAG	017E	JRTN	0000	JSOUT	0054	JUMP	0178
KBDA	0427	KBDB	044C	KBDC	0452	KBGET	049A
KBINIT	0400	KBTEST	047D	KBWAIT	048E	KIMA	041E
KIMB	0442	KIMDEL	0240	KIMOUT	04F2	KIMTST	0223
KIN	0004	KMDONE	048D	KNORM	04B7	KOUT	0000
KTST	0002	LCHAR	0181	LEFT	027E	LFEED	02B4
LRT	00F6	LSUB	0176	NAA	01E1	NACCES	8B9C
NODATA	048A	NOTUP	04C8	NTT	0200	NWW	01E9
NXX	01D9	NYY	01CB	NZZ	020A	OUTNXT	01BE
OUTTV	019F	OUTVEC	A663	RAMEND	0174	RAMPAG	0173
RIGHT	0278	SCRA	0285	SCRB	0291	SCRLOW	00F4
SCROLL	0280	SDEL	03A8	SDELA	03CD	SETAK	009B
SETASK	00A1	SETEND	02A2	SETUP	0500	SNDLOP	015F
SPACES	0317	SRTN	0005	STCHA	0361	STCHAA	0352
STCHB	0365	STCHC	0369	STCHD	036B	STCHE	0377
STCHR	0337	STORE	0321	SUBR	017B	SYMDEL	0395
SYMLF	0235	SYMTST	0233	SYMX	023C	TABLE	03ED
TDONE	008D	TECHO	A653	TGL	01D1	TLOOP	007F
TRANSF	0378	TSTAS	0231	TTABLE	0058	UIN	0108
UOUT	04EA	UP	027C	USER	0067	VIDEO	0541
XTEMP	017F	XXA	0342	XXB	035E	XXC	03A2
YTEMP	0180						