KIM HINTS

Since you and your KIM-1 are relative strangers, we'd like to help you get better acquainted. The material in this pamphlet will answer questions that are frequently asked by a new KIM-1 user.

---

ANSWERS TO POPULAR KIM SYSTEM QUESTIONS

1. IS IT POSSIBLE TO OUTPUT DIGITS OTHER THAN HEX TO THE 6 OUTPUT LED'S?

Since the 6502 is doing all segment decode and multiplex, it is possible to display data other than hex on a 7-segment readout. A pseudo alphabet has been developed and is displayed in the 7-segment display of the KIM in a scrolling manner.

2. WHEN HANDLING THE BOARD, WOULD THE STATIC HAZARD BE RELIEVED IF ALL EDGE CONNECTORS WERE SHORTED TOGETHER?

The static problems are not as serious once the devices are installed in the P.C. board. Just be sure to use grounded tools and to discharge yourself to ground before touching KIM or the connected circuits.

3. WHAT TYPE OF LED READOUT IS USED ON KIM-1 FOR U18, etc? GENERAL COMMON ANODE OR CATHODE?

USE MAN-72 Type displays, available from many manufacturers. General common anodes should work, although you may find intensity differences between them.

4. WHERE CAN I GET MORE 44-PIN EDGE CONNECTORS FOR KIM?

The connector is a standard part — you can order a Vector No. R644 from most electronic supply houses.
The connector is also carried by most Radio Shack stores as Part No. 276-548.

5. ARE THERE ANY INTERFACES OR PROM PROGRAMMERS AVAILABLE WITH KIM TO PROGRAM EPROMs OR TO DUPLICATE PROMs?

No, not yet.

6. IS THERE AN I/O EXPANSION BOARD AVAILABLE?

Not yet... soon, we hope.

7. IS THERE A BOARD AVAILABLE TO MAKE USE OF MEMORY ADDRESSES 0400-13FF?

Check the "Kilobaud" article (issue #4, April 1, 1977, page 74) entitled "KIM Memory Expansion."

8. HOW DO I SET UP MY KIM FOR AUDIO CASSETTE RECORDING AND PLAYBACK?

A number of KIM-1 customers have reported difficulty in achieving correct results for the sample problem shown in Sec. 2.4 of the KIM-1 User Manual. In addition, some customers have experienced problems in recording or playback of audio cassettes. (Sec. 2.5 of the <IM-1 User Manual). In all cases, the problems have been traced to a single cause: the inadvertent setting of the DECIMAL MODE.
The 6502 Microprocessor Array used in the KIM-1 system is capable of operating in either binary or decimal arithmetic mode. The programmer must be certain that the mode is selected correctly for the program to be executed. Since the system may be in either mode after initial power-on, a specific action is required to insure the selection of the correct mode.
Specifically, the results predicted for the sample problem (Sec. 2.4) are based on the assumption that the system is operating in the binary arithmetic mode. To insure that this is the case, insert the following key sequence prior to the key operations shown at the bottom of Page 11 of the KIM-1 User Manual.

```
AD
00 F1
DA 00
```

This sequence resets the decimal mode flag in the Status Register prior to the execution of the sample program.
The same key sequence may be inserted prior to the key operations shown on pages 14 and 15 for audio cassette recording and playback. These operations will not be performed correctly if the decimal mode is in effect.
In general, whenever a program is to be executed in response to the /GO/ key, the programmer should insure that the correct arithmetic mode has been set in the status register (00F1) prior to program execution.

9. HOW DO I SOLVE AUDIO CASSETTE INTERFACE PROBLEMS?

A. Insure that memory location O0F1 has been set to a value of 00 before recording or playing back the tape. This is the source of 90% of all cassette problems.

B. Mis-adjustment of the variable resistor (VR1) in the cassette circuitry is almost never a problem. Any setting near the center of its rotation will work fine.
C. Make sure that +12V is connected during playback. 
   NOTE: +12V is not required for recording, so a lack of +12V will result in good recording but no playback.

D. If the display frequently relights showing FFFF, 
   the fault is probably in the tape unit itself — not the KIM. 
   Using poor-quality cassettes is usually to blame. 
   Some cassette recorders have such poor power filtering 
   that they will work fine on batteries, but will not work with an AC adapter because of hum induced 
   during record or playback. Tapes should always be rewound before removal from the machine, as a fingerprint on the tape will result in errors on playback.

E. Make sure that only a single ground line is run from 
   the KIM ground to the barrel of the microphone input 
   of the cassette recorder. Leave the barrel of the ear- 
   phone output ungrounded. The shield around the line to 
   the earphone should be attached to ground on KIM.

F. Problems of playing a tape recorded on one KIM 
   system back on another system or a different cassette 
   player can usually be solved by adjusting the head 
   adjustment screw on the new cassette recorder. Play 
   back a cassette recorded on the old deck on the new 
   machine and adjust the head screw on the new machine 
   for maximum volume. This adjustment is especially 
   critical when using the SuperTape program.

10. HOW DO I SOLVE TELETYPET PROBLEMS?

A. The most common problem is that the system does 
   not respond to a reset-rubout sequence with a model 
   33 Teletype. This can be fixed by removing the wire 
   connected to pin R on the KIM application connector, 
   connecting a 470 ohm resistor to that wire, and 
   connecting the other end of the resistor to the +12V 
   supply at pin N.

B. No information is available on connecting other 
   Teletype models (14, 28, 32) to KIM.

C. Schematics for interfacing KIM to an RS232C port 
   are in the April, 1976 “Byte” magazine and in the first 
   issue of the KIM user notes. (Reproduced below):

D. Other common sources of Teletype problems are a 
   short circuit in C5 or a burned-out Q7. Signal tracing 
   with a ‘scope should reveal these problems.

11. HOW DO I SOLVE PAPER TAPE PROBLEMS?

A. K-M-1’s having a date code in 1975 on the 6502 
   will not read paper tape correctly. These CPU’s will 
   be replaced by MOS without charge. Tom Pittman’s 
   TINY BASIC will not work on these machines either. 
   The problem occurs because early versions of the pro- 
   cessor did not set the zero flag correctly on TXA, 
   TYA, TAX, or TAY instructions.

B. When using a Texas Instruments Silent 700 data 
   terminal equipped with digital cassettes or other high- 
   speed paper tape devices, a Q’s paper tape dump may 
   be performed at any speed acceptable to the data 
   terminal, but playback (through the L command) must 
   be at 10 cps.

12. WHAT DO I DO ABOUT OTHER PROBLEMS?

A. If the RESET on KIM causes only a single digit or 
   segment to light on the display, the KIM must be 
   returned for repair.

B. When in doubt, check all power supply voltages 
   on the KIM board, not at the power supply terminals.

C. When software works strangely or erratically, 
   decimal/binary mode problems may be involved.

D. There is an error in the KIM Resident Assembler 
   manual regarding the addresses for the symbol table 
   vectors. The vector locations are DF, E0, E1, E2. The 
   text is incorrect, the example is correct.

E. Problems with KIM-2/3’s which fail the memory 
   test program can almost always be traced to excessive 
   cable length between the KIM-1 and the KIM-2/3. 
   Any cable should be 6” in length or less.

13. WHAT ARE THE KIM SYSTEM POWER SUPPLY 
       REQUIREMENTS?

KIM 1 — Microcomputer Board:

   Recommended: 1.2A +5V ±5%
   100mA +12V ±5%

   The actual power measured ranges 700 mA to 1A at 
   +5V and the schematic indicating 3A at transformer 
   is incorrect.

KIM 3A – 8K RAM Memory Board:

   Recommended: +5V, 3A

   Average consumption calculated is about 2.4A. 
   Board has +5V regulator accepting unregulated +8 to 
   +10V DC.

KIM 4 – Mother Board:

   Consumption about 200mA. Board has +5V regula- 
   tor accepting unregulated +8 to +10V DC and 
   +12V regulator accepting unregulated +15V DC 
   to support both KIM1 and KIM 4. KIM 4 has 6 slots 
   for memory expansion with KIM2 and KIM3 and 
   hence a total power supply requirement is a cumulative 
   value dependent on KIM-System configuration.

14. WHAT SOFTWARE IS AVAILABLE?

   The following software is available for use with the 
   KIM-1 and/or other 6502-based systems:
1. Tiny BASIC — runs in 2K. $5 for paper tape from:
   Tom Pittman
   Box 23189
   San Jose, California 95153

2. Many games and other information in the KIM-1
   User Group Newsletter, $5 for 6 issues:
   Eric Rehnke
   109 Centre Avenue
   W. Nerriton, PA 19401

3. An excellent Chess playing program which runs
   in 1K. $10
   MICRO CHESS
   27 Firstbroke Rd.
   Toronto, CANADA M 4E 2L2

4. A good group of games plus an intermediate-level
   language called PLEASE for KIM-1 — $15 from:
   THE COMPUTERIST
   Post Office Box 3
   S. Chelmsford, MA 01824

5. The 6502 Program Exchange
   2920 Moana
   Reno, NV 89509

6. Micro Software Specialists
   2024 Washington Street
   Commerce, TX 75428

7. KIMATH, a complete floating-point math package
   including both source and object code is available from
   MOS Technology for $15.

8. A 4K version of FOCAL, a BASIC-like interpreter, and
   a 6K Resident assemble/text Editor, both with source
   listings and object code on KIM cassette or paper tape
   are available from:
   ARESCO
   314 Second Ave.
   Haddon Heights, NJ 08035
   The FOCAL is $50 and the assembler/Editor is $70.
   A complete information package is $2.

9. An 8K version of BASIC for KIM is available for $99
   from:
   Johnson Computing
   123 W. Washington St.
   Medina, Ohio 44256
   (215) 725-4568

10. "FIRST BOOK OF KIM" is a collection of games,
    utility programs, hints and kinks, etc. (180 pgs).
    $3.00 plus 50¢ postage from:
    ORB
    P.O. Box 311
    Argonne, ILL 60439

**INTERVAL TIMER OPERATION**

**1. OPERATION**

a. Loading the timer
   The divide rate and interrupt option enable/disable are pro-
   grammed by decoding the least significant address bits.

**KIM SUBROUTINES**

<table>
<thead>
<tr>
<th>CALL</th>
<th>ADDRESS</th>
<th>ACTION</th>
<th>ARG.</th>
<th>RESULT</th>
<th>NOTES</th>
</tr>
</thead>
</table>
| JSR AK   | 1EFE    | Check for key depressed               | –    | A      | A = 0 = Key down
|          |         |                                       |      |        | A ≠ 0 = No Key down
|          |         |                                       |      |        | X & Y lost      |
| JSR GETKEY | 1F6A   | Get key from keyboard                 | –    | A      | A > 15 illegal
|          |         |                                       |      |        | or no key       |
| JSR SCANS | 1F1F   | Display F9, FA, FB                    | F9,  | –      | A, X, Y are lost|
|          |         |                                       | FA   |        |                |
| JSR GETCH | 1E5A   | Put character from TTY in A           | –    | A      | X preserved    |
|          |         |                                       |      |        | Y = FF          |
| JSR PRTBYT | 1E3B  | Prints A as 2 Hex Char.               | A    | –      | A preserved    |
|          |         |                                       |      |        | X preserved    |
|          |         |                                       |      |        | Y = FF          |
| JSR PRTPNT | 1E1E  | Prints Contents of FB & FA on TTY     | FB,  | –      | A lost         |
|          |         |                                       | FA   |        | X preserved    |
|          |         |                                       |      |        | Y = FF          |
| JSR OUTCH | 1EA0   | Print ASCII char in A on TTY          | A    | –      | X is preserved |
|          |         |                                       |      |        | Y = FF          |
|          |         |                                       |      |        | A = FF          |
| JSR OUTSP | 1E9E   | Print a space                         | –    | –      | A = FF          |
|          |         |                                       |      |        | X preserved    |
|          |         |                                       |      |        | Y = FF          |
The starting count for the timer is determined by the value written to that address.

<table>
<thead>
<tr>
<th>Writing to Address</th>
<th>Sets Divide Ratio To</th>
<th>Interrupt Capability Is</th>
</tr>
</thead>
<tbody>
<tr>
<td>1704</td>
<td>1</td>
<td>Disabled</td>
</tr>
<tr>
<td>1705</td>
<td>8</td>
<td>Disabled</td>
</tr>
<tr>
<td>1706</td>
<td>64</td>
<td>Disabled</td>
</tr>
<tr>
<td>1707</td>
<td>1024</td>
<td>Disabled</td>
</tr>
<tr>
<td>170C</td>
<td>1</td>
<td>Enabled</td>
</tr>
<tr>
<td>170D</td>
<td>8</td>
<td>Enabled</td>
</tr>
<tr>
<td>170E</td>
<td>64</td>
<td>Enabled</td>
</tr>
<tr>
<td>170F</td>
<td>1024</td>
<td>Enabled</td>
</tr>
</tbody>
</table>

b. Determining the timer status
After timing has begun, reading address location 1707 will provide the timer status. If the counter has passed the count of zero, bit 7 will be set to 1, otherwise, bit 7 (and all other bits in location 1707) will be zero. This allows a program to “watch” location 1707 and determine when the timer has timed out. Note that reading 1707 provides an entirely different function from writing the same location.

c. Reading the count in the timer
If the timer has not counted past zero, reading location 1706 will provide the current timer count and disable the interrupt option; reading location 170E will provide the current timer count and enable the interrupt option. Thus the interrupt option can be changed while the timer is counting down. Note that you read 1706 or 170E regardless of which location (1704-0F) was written to start the timer.

If the timer has counted past zero, reading either memory location 1706 or 170E will restore the divide ratio to its previously programmed value, disable the interrupt option and leave the timer with its current count.

d. Using the interrupt option
In order to use the interrupt option described above, line PB7 (application connector, pin 15) should be connected to either the IRQ (Expansion Connector, pin 4) or NMI (Expansion Connector, pin 6) pin depending on the desired interrupt function. PB7 should be programmed as an input line (its normal state after a RESET).

3. INTERVAL TIMER AND KEYBOARD OPERATION
The following three programs show the use of the interval timer, keyboard, and seven segment displays in user programs.

The first program loads a value of 50 in the timer and waits for it to time out, repeats the process, and then increments the count in the display register (00FA and 00FB) and calls the display subroutine SCANS. The process then repeats.

The second program performs the same function as the first, but uses the timer to provide interrupts, rather than watching the timer status register (1707). Thus this program is constantly cycling through the display program SCANS except when the timer generates an interrupt. When an interrupt occurs the interrupt service routine (starting at location 010C) resets the timer, increments the display register and returns to the display program. Note that the LED display is brighter when using this program because most of the computer's time is spent displaying rather than watching the timer.

The third example program demonstrates the use of the keyboard and display. Any key depressed will appear in the rightmost digit of the display and will be shifted to the left with each successive keyboard entry.

Notice that the SCANS routine not only displays the contents of 00F9, 00FA and 00FB but also returns with the Z flag set to 0 if a key is currently depressed. The GETKEY routine is then called to determine which key has been depressed. Since the SCANS subroutine takes several milliseconds, a call to this routine can be used to “waste time” and let any keybounce stop.

NOTE

If the programmer desires to use PB7 as a normal I/O line, the programmer is responsible for disabling the timer interrupt option (by writing or reading address 1706) so that it does not interfere with normal operation of PB7. Also, PB7 was designed to be wire-ORed with other possible interrupt sources; if this is not desired, a 5.1K resistor should be used as a pull-up from PB7 to +5v. (The pull-up should NOT be used if PB7 is connected to NMI or IRQ.)
### INTERVAL TIMER

**DEFINITION OF COMMONLY USED LOCATIONS**

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DA</td>
<td>$1700</td>
</tr>
<tr>
<td>DDA</td>
<td>$1701</td>
</tr>
<tr>
<td>DB</td>
<td>$1702</td>
</tr>
<tr>
<td>DDB</td>
<td>$1703</td>
</tr>
</tbody>
</table>

**DATA REG A**

**DATA REG B**

**DATA DIRECT REG A**

**DATA DIRECT REG B**

### TIMERS (WRITE TIME TO)

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1D</td>
<td>$1704</td>
</tr>
<tr>
<td>C8D</td>
<td>$1705</td>
</tr>
<tr>
<td>C64D</td>
<td>$1706</td>
</tr>
<tr>
<td>C1024D</td>
<td>$1707</td>
</tr>
</tbody>
</table>

**DIV BY 1**

**DISABLE INT**

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1E</td>
<td>$170C</td>
</tr>
<tr>
<td>C8E</td>
<td>$170D</td>
</tr>
<tr>
<td>C64E</td>
<td>$170E</td>
</tr>
<tr>
<td>C1024E</td>
<td>$170F</td>
</tr>
</tbody>
</table>

**DIV BY 8**

**DISABLE INT**

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRD</td>
<td>$1706</td>
</tr>
<tr>
<td>SR</td>
<td>$1707</td>
</tr>
<tr>
<td>TRE</td>
<td>$170E</td>
</tr>
</tbody>
</table>

**READ TIME DISABLE INT**

**READ INT STAT**

**READ TIME ENABLE INT**

**WHEN THE INTERRUPT STATUS IS READ**

**THE INTERRUPT IS NEITHER DISABLED OR ENABLED. BIT 7 IS A ONE IF TIME OUT HAS OCCURRED. BIT 7 IS ZERO IF TIME OUT HAS NOT OCCURRED. BITS 0-6 ARE ALL ZERO**

**WHEN THE TIMER TIMES OUT THE DIVIDER IS SET TO A DIV BY ONE AND THE TIMER CONTINUES TO COUNT AT CLOCK RATE**

**WHEN THE TIMER IS READ THE DIVIDER IS RESTORED TO ITS ORIGINAL VALUE AND THE INTERRUPT IS RESET**

### EXTERNAL SUBROUTINES

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCANS</td>
<td>$1F1F</td>
</tr>
<tr>
<td>INCP</td>
<td>$1F63</td>
</tr>
<tr>
<td>GETKEY</td>
<td>$1F6A</td>
</tr>
</tbody>
</table>

TO USE INTERRUPT PB7 MUST BE EXTERNALLY WIRED TO IRQ

### Program 1

**THIS EXAMPLE DOES NOT USE INTERRUPTS – THE DISPLAY WILL DIM AS A RESULT OF SLOW SCANNING**

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>A2 02</td>
</tr>
<tr>
<td>0002</td>
<td>A9 32</td>
</tr>
<tr>
<td>0004</td>
<td>80 06 17</td>
</tr>
<tr>
<td>0007</td>
<td>2C 07 17</td>
</tr>
<tr>
<td>000A</td>
<td>10 F8</td>
</tr>
<tr>
<td>000C</td>
<td>CA</td>
</tr>
<tr>
<td>000D</td>
<td>D0 F5</td>
</tr>
<tr>
<td>000F</td>
<td>20 63 1F</td>
</tr>
<tr>
<td>0012</td>
<td>20 1F 1F</td>
</tr>
<tr>
<td>0015</td>
<td>4C 00 00</td>
</tr>
</tbody>
</table>

**COUNT**

**=2**

**DELAY**

**=50**

**=*=$0000**

**LDX =COUNT**

**LDA =DELAY**

<table>
<thead>
<tr>
<th>LOC</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>START1</td>
</tr>
<tr>
<td>0004</td>
<td>AGAIN</td>
</tr>
<tr>
<td>0007</td>
<td>WAIT</td>
</tr>
<tr>
<td>000A</td>
<td></td>
</tr>
<tr>
<td>000C</td>
<td></td>
</tr>
<tr>
<td>000D</td>
<td></td>
</tr>
<tr>
<td>000F</td>
<td></td>
</tr>
<tr>
<td>0012</td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td></td>
</tr>
</tbody>
</table>

**DIV BY 64 DISABLE INT**

**READ STATUS DISABLE INT**

**BIT 7 = 1 TIME OUT COMPLETE**

**LOOP ON COUNT**

**MONITOR UTIL INC FA,FB**

**MONITOR UTIL DISP F9,FA,FB**

**COUNT DOWN 2 TIMES EACH DELAY 50 CYCLES ORG AT 0**

**DIV BY 64 DISABLE INT**

**READ STATUS DISABLE INT**

**BIT 7 = 1 TIME OUT COMPLETE**

**LOOP ON COUNT**

**MONITOR UTIL INC FA,FB**

**MONITOR UTIL DISP F9,FA,FB**
INTERVAL TIMER (Continued)

Program 2

THIS EXAMPLE USES INT.
WIRE PB7 TO IRQ EXTERNALLY

```
0018  *$0100
0100  58  START 2
0101  A9 FF  LDA =$FF
0103  8D 0F 17  STA C1024E
0106  20 1F 1F  JSR SCANS
0109  4C 06 01  JMP DISP
```

DISPLAYS CONTENTS OF F9,FA,FB

INTERUPT SERVICE ROUTINE

```
010C  A9 FF  INTSVC  LDA =$FF
010E  8D 0F 17  STA C1024F
0111  20 63 1F  JSR INCPT
0114  40  RTI
0115  *$17FE
17FE  0C 01  IRQT .WORD INTSVC
```

ORG AT IRQ VECTOR

SET = TO INT SERVICE RTN

Program 3

THIS EXAMPLE DESCRIBES USE OF
KEYBOARD AND DISPLAY

```
1800  *$0200
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
```

LSD'S

THESE 3 BYTES ARE DISPLAY BVF

MSD'S

IF KEY IS DEPRESSED WAIT FOR

ITS RELEASE

WAIT FOR KEY DEPRESSED

WHEN DEPRESSED GO TO VALIDATION

THIS USED AS DEBOUNCE

MONITOR UTIL WHICH GETS KEY VAL

IF MPU IN DEC MODE THEN GET KEY

GETS DECIMAL VALUE A=10

LEFT JUSTIFY KEY VALUE

SET UP LOOP COUNT=4

SHIFT ALL DIGITS 1 PLACE LEFT

DO THIS ONE BIT AT A TIME

END OF MOS/TECHNOLOGY 65CX ASSEMBLY VERSION 4
NUMBER OF ERRORS = 0, NUMBER OF WARNINGS = 0