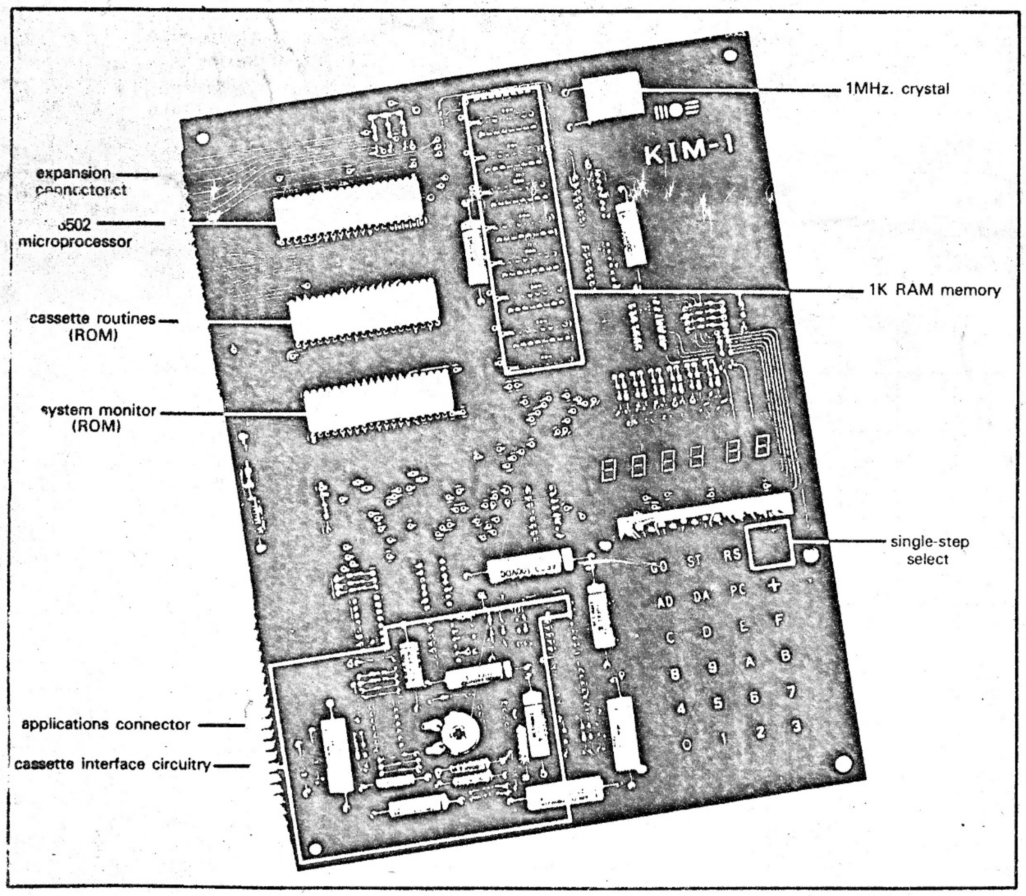
KIM-1 Microcomputer Module

a user's notes

by T. E. Travis

The MOS Technology KIM-1 Microcomputer Module is a *complete* microcomputer system on a single printed circuit card **(fig. 1)**. The system consists of a microprocessor, two ROMs containing a monitor program, 1152 mi s of read/write memory, four I/O ports, two programmable interval timers, a six-digit LED display, a keyboard, an audio cassette interface and a TTY interface **(fig. 2)**, all for only $245. When the author first heard of this system he was impressed with its completeness and its low price compared to other available microprocessor systems. First-hand experience with this module has proven these initial impressions to be correct and has shown the KIM-1 to be even more versatile than previously expected.



KIM-1 features

The microprocessor chip around which the system is built is the MOS Technology 6502 8-bit CPU. The read only memories which contain the KIM-1 monitor routine are MOS Technology 6530 chips, each of which also contains 64 bytes of read/write memory, two I/O ports and one interval timer. The monitor routine **(fig. 3)** allows the user to enter information into the read/write memory from the self-contained calculator-type keyboard, to display information on the six-digit seven-segment LED display, to initiate the execution of programs in either a continuous or single step mode, to stop programs, examine or modify the contents of the microprocessor registers and then resume execution, and finally, to store programs or data on audio cassette tapes, and to reenter such tapes at a later date. In addition to all of these capabilities, the KIM-1 also contains a standard 20 milliampere teletype interface allowing a teletype keyboard, printer, and paper tape reader/punch to be used in place of, or in conjunction with, the self-contained calculator-type keyboard, LED display and audio tape unit. The only component needed by the user which is not provided with the system is a power supply which will output 5 volts at approximately one ampere for the logic and display and, if an audio cassette is to be used, 12 volts at approximately 100 milliamperes for the audio circuitry. The circuit for such a power supply is included along with the ample documentation that accompanies this module.

The purpose of this article is not only to relate the very favorable experience this author has had in using the KIM-1 but also to present some information the author has derived pertaining to the use of the self-contained keyboard and display from application programs. In addition, a simple real-time monitor is presented which will allow other users to easily implement programs that require the services of the keyboard, the display and some means of keeping track of real time while an application program is executing.

Since the KIM-1 is not a kit it may literally be put on the air in a matter of minutes after it is unpacked from the shipping carto[illegible] All that is required is a 5 vo[illegible] power supply to drive the logi[illegible] and display. However, the man ufacturer does recommend read ing the KIM-1 user's manual be fore turning on the unit to pre vent accidental damage. Onc[illegible] power is applied through th[illegible] supplied edge connector, push ing the RESET key immediatel[illegible] initiates execution of the built-i[illegible] monitor program. At this point the address of any memory loca

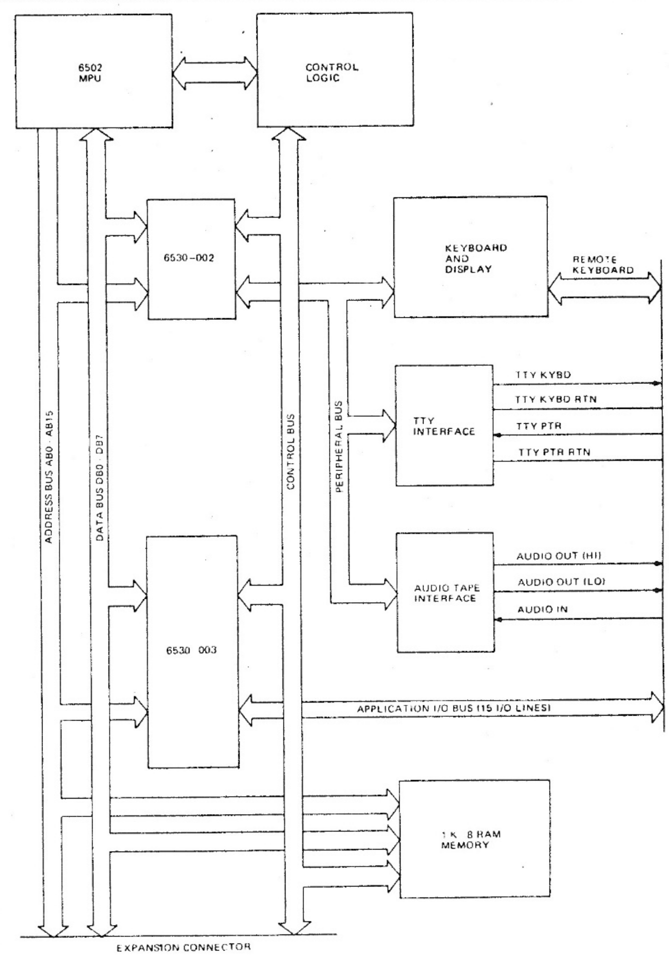


fig. 2. KIM-1 system block diagram.

tion to be examined or modified can be entered from the keyboard and data may be entered into the location selected. Pressing the "+" key steps the address to the next location in memory; it is only necessary to enter the first address of any series of contiguous locations to be examined or modified.

single step mode

Once a program has been entered, execution of the program is initiated by entering its starting location from the keyboard and depressing the GO key. If it is desired to step through a program in order to determine which instructions are being executed and in what order, the microcomputer can be put into the single step mode with a slide switch on the keyboard. This will cause only a single instruction to be executed every time the GO key is pushed. In this mode the address of each instruction and its OP code is displayed as the instructions are executed. Furthermore, by addressing the stack locations in read/write memory and examining their contents, it is also possible at any point to determine what is contained in the microprocessor's internal registers or memory. Pressing the PROGRAM COUNTER key causes the KIM-1 system to prepare to resume execution of the interrupted program where it was halted. In this way, programs may be readily debugged by watching both the sequence of operations being performed by the microprocessor and their results.

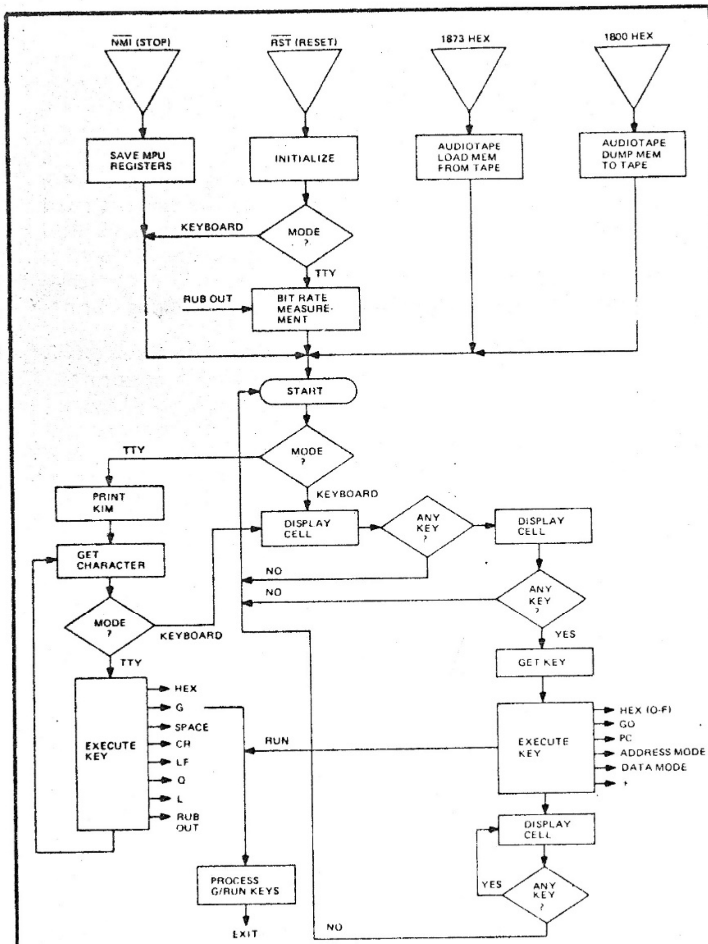


fig. 3. Flowchart of system monitor contained in the two 6530 ROMs.

cassette storage

Once a program has been entered and debugged it may then be saved using the audio cassette interface (or teletype paper tape punch). Under these circumstances the 12 volt power supply is required. An inexpensive audio cassette recorder can be connected through the edge connector to the KIM-1 with no modification and used with this module. The starting and stopping locations of the memory to be transferred to the cassette are entered into reserved locations in the read/write memory. The user selects and enters a two-digit hex record identification number and the *cassette write* program is initiated. Once the program has been transferred to the cassette, control returns to the monitor program.

A previously written cassette record can be read back into the read/write memory by entering its identification number and executing the *cassette read* program. Since each program stored on a cassette has its own identification number, it is possible to store many programs on the same cassette and then let the cassette routine read until the proper program is found. In addition, since the audio cassette interface on the KIM-1 board has very good noise rejection characteristics, it is possible to put voice identifiers between programs on a cassette to facilitate finding a specific program by listening to the voice information on the tape.

As mentioned earlier, each of the 6530 ROMs contains two I/O ports, giving a total of four I/O ports for the KIM-1 module. However, two of the ports are dedicated to processing the KIM-1's own I/O devices; namely the keyboard, display, teletype interface and audio cassette interface. This leaves two uncommitted I/O ports or a total of 15 bits (PB-6 of port B is dedicated during chip masking to function as an additional chip select due to KIM-1 addressing requirements). Each of these I/O bits may be initialized under software control to be either an input or output, or dynamically changed within the same program.

using dedicated ports

Although intended primarily for use by the KIM-1 monitor, the first two I/O ports are not entirely lost to the user, particularly if he desires to use the built-in keyboard or display. Exactly how to use these devices is not detailed in the KIM-1 documentation. However, a little examination of the circuit diagram **(fig. 4)** quickly shows the techniques necessary to make use of these devices. Four bits of the monitor's port B are routed to a BCD-to-decimal decoder. Three of the outputs from this decoder are routed to the keyboard matrix, six of them are used as LED select signals for the multiplexed display and the tenth is not important here. Seven bits of the monitor's port A are used both as inputs from the keyboard and as outputs for display purposes, where they become segment

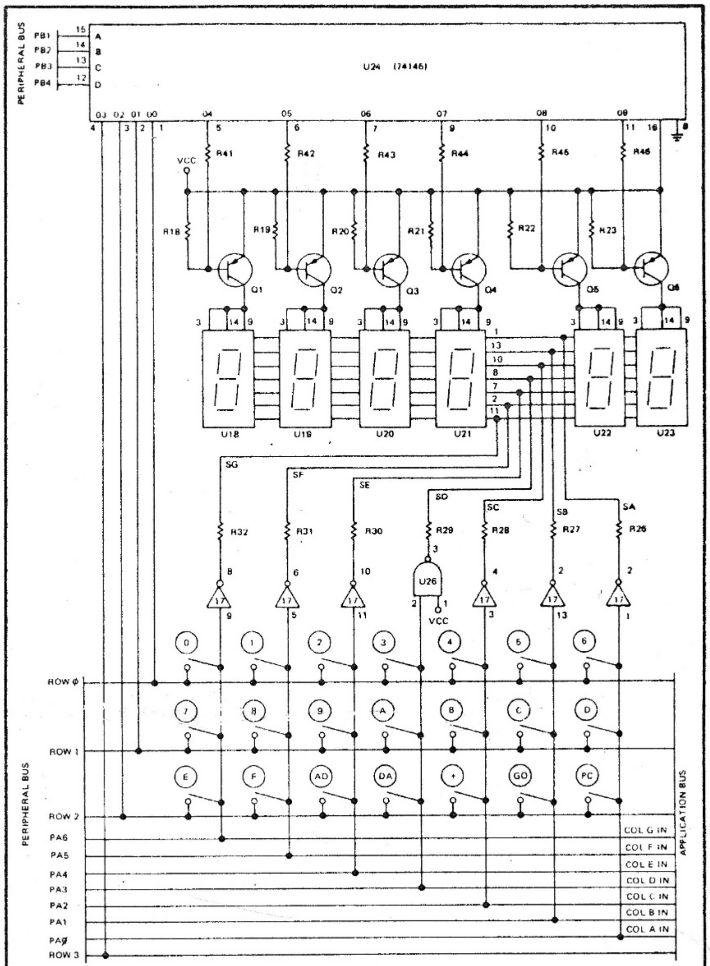


fig. 4. Schematic of keyboard and time multiplexed LED display.

listing 1. Real-Time Executive

LOC	B1	B2	B3	OP	ARG	COMMENTS
200	A9	00		LDA	#0	Zero display counter, display, keyboard status and software clock
202	A2	1F		LDX	#1F	
204	95	CF		STA	CF, X	
206	CA			DEX		
207	D0	FB		BNE	204	
209	A9	7A		LDA	#7A	Initiate 976 µsec delay
20B	8D	45	17	STA	1745	
20E	F8			SED		Add 1 msec to decimal software clock

listing 1 [continued]:

LOC	B1	B2	B3	OP	ARG	COMMENTS
20F	18			CLC		
210	A5	D7		LDA	D7	
212	69	01		ADC	#1	
214	85	D7		STA	D7	
216	A5	D8		LDA	D8	
218	69	00		ADC	#0	
21A	85	D8		STA	D8	
21C	A5	D9		LDA	D9	
21E	69	00		ADC	#0	
220	85	D9		STA	D9	
222	A5	DA		LDA	DA	
224	69	00		ADC	#0	
226	85	DA		STA	DA	
228	D8			CLD		
229	A9	1E		LDA	#1E	Read keyboard & move last scan to last scan area (DE-E0)
22B	8D	43	17	STA	1743	
22E	A9	00		LDA	#0	
230	8D	41	17	STA	1741	
233	A2	02		LDX	#2	
235	B5	DB		LDA	DB, X	Move old byte
237	95	DE		STA	DE, X	
239	8A			TXA		
23A	0A			ASL		Double row index
23B	8D	42	17	STA	1742	Select row
23E	A9	00		LDA	#0	
240	18			CLC		
241	ED	40	17	SBC	1740	Read l's complement of keyboard
244	29	7F		AND	#7F	Mask to seven bits
246	95	DB		STA	DB, X	Store new byte
248	CA			DEX		
249	10	EA		BPL	235	Loop over 3 rows
24B	A6	D0		LDX	D0	Display next digit
24D	CA			DEX		
24E	10	02		BPL	#2	
250	A2	05		LDX	#5	Reset digit counter to 5
252	86	D0		STX	D0	
254	8A			TXA		Multiply digit index by 2 and add 8
255	0A			ASL		
256	69	08		ADC	#8	
258	8D	42	17	STA	1742	Select digit
25B	A9	7F		LDA	#7F	Enable segment outout bits
25D	8D	41	17	STA	1741	
260	B5	D1		LDA	D1, X	Pick up digit
262	8D	40	17	STA	1740	Send to segments
265	20	D2	02	JSR	2D2	Jump to applications program
268	AD	45	17	LDA	1745	Check for timer done
26B	29	80		AND	#80	
26D	C9	80		CMP	#80	
26F	D0	F7		BNE	268	
271	4C	09	02	JMP	209	Start next 1 msec cycle

drivers for the LED displays. Note that these seven bits provide functions which would require fourteen bits in a system without software programmable I/O ports.

To use the display, it is only necessary to send the number corresponding to the digit (LED) position of interest to the BCD-to-decimal decoder via the B port. The seven *segments* forming the selected digit are then sent out via the A port after it has been programmed to function as an output port. Since the outputs are latched (in the 6530 chip), they remain displayed until one of the ports is reprogrammed. Since this allows only one digit at a time to be displayed, it is necessary to display each digit briefly in turn (i.e., the LEDs are time multiplexed for approximately one millisecond duration to give the appearance of a continuous display).

keyboard scanning

The brief period between digits is a convenient time to use the monitor's port A as an input port for scanning the keyboard for depressed keys. This is accomplished by outputting through port B the address of one row of the keyboard at a time, reading in the seven bits corresponding to the seven keys in the addressed row (via port A), and then determining if a zero has been read in at any of the seven key positions. If so, by shifting and counting to determine the location of the zero, it is possible to determine which key has been depressed. An example of how to perform this function and the display multiplexing is presented in the software listing **(listings 1 and 3)**.

main programs

The listing included here consists of two major programs. The first **(fig. 5 - listing 1)** is a real-time executive for the KIM-1 module which can be used to support a wide variety of applications programs. The second (**listings 4-7**) is one such application program; a game designed to measure a participant's reaction time. The executive performs the following functions:

1. It initializes one of the programmable timers contained in the KIM-1 module to time-out a one millisecond interval.
2. Each millisecond it adds one to an internally maintained decimal software clock which keeps track of the time from the initiation of the program.
3. It reads the status of the keyboard and stores it, plus the previous status, in two three-byte arrays.
4. Each millisecond it displays the next digit of the six digits stored in memory locations corresponding to the six digits of the LED display (D1-D6).

subroutines

Also included with this executive are two subroutines: The first

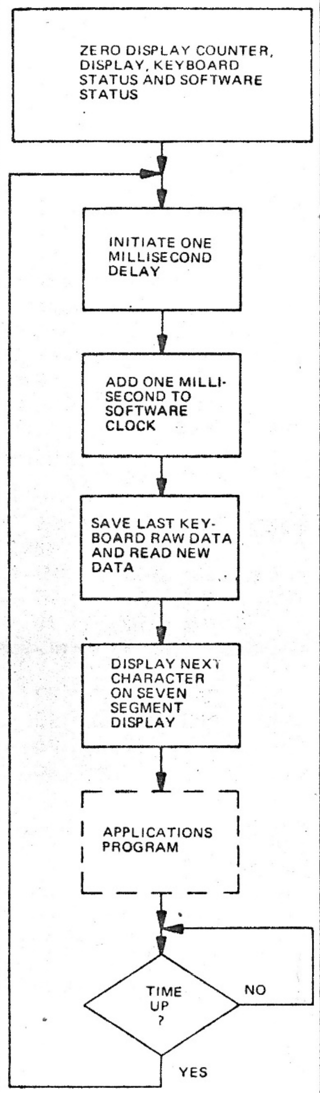


fig. 5. Real-time executive flowchart with display refresh and service to user's application program.

LOCATION	RANGE	USAGE
D0	0-5	Index of next character to be multiplexed to the LED display
D1-D6	—	Six characters to be displayed by executive in seven-segment display format
D7-DA	—	Decimal real-time software clock. Least significant digit (low order half of D7) is in milliseconds
DB-DD	—	Most recent raw keyboard data (complemented and masked to seven bits)
DE-E0	—	Next most recent raw keyboard data

table 1. Variables used by executive.

listing 2. Convert binary to seven - segment code

```
LOC  B1  B2  B3  OP   ARG  COMMENTS
                 HIGH ORDER 4 BITS (SHIFT RIGHT 4)
292  4A          LSR
293  4A          LSR
294  4A          LSR
295  4A          LSR
                      LOW ORDER 4 BITS
296  29  0F      AND  #F
298  A8          TAY
299  B9  76  02  LDA  276, Y Pick up 7 segment code from table
29C  60          RTS
                      7-SEGMENT CODES
276  3F  06  5B            0, 1, 2
279  4F  66  6D            3, 4, 5
27C  7D  07  7F            6, 7, 8
27F  67  77  7C            9, A, B
282  39  5E  79            C, D, E
285  71                    F
```

Note: This same basic conversion table is contained in locations 1FE7-1FF6 in the monitor routine ROM. It is included here for the sake of completeness.

(**listing 2**) converts from binary to seven-segment display code and the second (**listing 3**) converts raw keyboard data into the equivalent binary number.

executive operation

During each one millisecond interval, after servicing the keyboard, display and programmable timer, the executive branches to location 2D2 (hex) where the applications software is entered (**fig. 5**). The applications program is then free to use the keyboard data, set up information to be displayed on the LED display, or perform any other required function *providing* it does not retain control of the computer for more than one millisecond. When finished with a one millisecond time slice, it returns to the executive which then waits for the initiation of the next one millisecond interval. Should the applications program retain control of the computer for more than one millisecond, the system will still function, however, the internally maintained software clock will begin to run slow and the display may flicker. It is also possible to change the update interval from one millisecond to as much as four milliseconds if longer applications functions are to be accommodated. Beyond four milliseconds display flicker becomes noticeable.

listing 3. Interpret Keyboard

LOC	B1	B2	B3	OP	ARG	COMMENTS
29D	A2	02		LDX	#2	
29F	B5	DB		LDA	DB, X	If not stable for two scans return
2A1	D5	DE		CMP	DE, X	
2A3	F0	01		BEQ	#1	
2A5	60			RTS		
2A6	CA			DEX		
2A7	10	F6		BPL	29F	
2A9	A5	E2		LDA	E2	Move current status to old status
2AB	85	E1		STA	E1	
2AD	A9	00		LDA	#0	Zero index and keyboard status
2AF	AA			TAX		
2B0	48			PHA		Save keyboard status
2B1	B5	DB		LDA	DB, X	Pick up row input
2B3	85	E2		STA	E2	Store temporarily in E2 for shifting
2B5	A0	06		LDY	#6	Load columns index
2B7	06	E2		ASL	E2	Shift
2B9	10	04		BPL	#4	Check for no 1 in bit 7
2BB	68			PLA		Retrieve keyboard status
2BC	85	E2		STA	E2	Store
2BE	60			RTS		Return
2BF	68			PLA		Add 1 to status
2C0	69	01		ADC	#1	
2C2	48			PHA		Push it back
2C3	88			DEY		Decrement column count
2C4	10	F1		BPL	2B7	Check for last column this row
2C6	E8			INX		Increment row count
2C7	E0	04		CMX	#4	Check for last row
2C9	D0	E6		BNE	2B1	
2CB	68			PLA		No key depressed, set status to FF
2CC	A9	FF		LDA	#FF	
2CE	85	E2		STA	E2	
2D0	60			RTS		Return

slice-up long applications

This is a relatively simple executive. However, it does not require much memory and is quite versatile for a large number of modest applications. When an applications task requires much more than one millisecond for execution it is only necessary that before returning to the executive, which it *must* do at the end of each time slice, the task post flags to itself so it can resume operation where it left off at the end of the previous time slice.

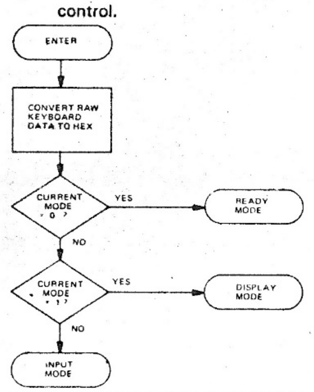
sobering application

The sample application program which is included in this article has been called a *drunk test*. This is due to the fact that it resembles a proposed device that would be connected to the ignition of an automobile and which would not allow the automobile to be started unless the driver was sufficiently sober to be able to enter a displayed sequence of random digits within a specified amount of time. In this program, the player presses the "+" key on the keyboard and a four digit random number is displayed for two seconds. As soon as the display goes blank, the player tries to re-enter the four digits as rapidly as possible. As the number is entered it shows up on the display. When the last digit is entered, the time required to enter the number is displayed in the last two digits of the display in tenths of a second. Should the player make a mistake or require more than *ten seconds* to enter his number, "FF" is displayed — indicating failure. When the "+" key is again depressed a new random number is displayed and the cycle is repeated. Although this is a fairly straight forward program it does use all the features of the executive and has turned out to be very popular with a variety of players. It therefore makes a good demonstration program.

operating modes

There are four major portions of this program which demonstrate how the executive is used in a real application. The first part of the program **(fig. 6 — listing 4)** must decide what *mode* the game is in at the current time. These modes are: waiting for the "+" key to be depressed, waiting for the two second display interval to elapse, and servicing the keyboard as the player re-enters

fig. 6. Flowchart for Drunk Test control.



the random number. Also included in this initial segment of code is a call to the keyboard interpretation subroutine so that subsequent modes of the program have access to the current status of the keyboard without calling this routine themselves.

ready mode

In the ready mode **(listing 5)** the program simply checks whether or not the "+" key is currently depressed. If not, it returns to the executive to wait out the rest of the one millisecond interval. If the "+" key is depressed, the mode indicator (E3)

listing 4. Drunk Test Control

LOC	B1	B2	B3	OP	ARG	COMMENTS
2D2	20	9D	02	JSR	29D	Call keyboard conversion
2D5	A6	E3		LDX	E3	Pick up current mode from E3
2D7	F0	06		BEQ	2DF	If zero, go to ready mode
2D9	CA			DEX		
2DA	F0	4C		BEQ	328	If 1, go to display mode
2DC	4C	50	03	JMP	350	Otherwise, go to input mode

listing 5. Ready Mode (E3 = 0)

LOC	B1	B2	B3	OP	ARG	COMMENT
2DF	A5	E2		LDA	E2	"+" pressed? Pick up current keyboard status from E2 and test for 12
2E1	C9	12		CMP	12	
2E3	F0	01		BEQ	#1	
2E5	60			RTS		If not return
2E6	A9	01		LDA	#1	Set display mode (E3 = 1)
2E8	85	E3		STA	E3	
2EA	A2	00		LDX	#0	Transfer random number to E7-EA
2EC	86	E6		STX	E6	after unpacking. Also transfer
2EE	A2	02		LDX	#2	it to display after conversion.
2F0	B5	D6		LDA	D6, X	E5 is the index for picking up the
2F2	86	E5		STX	E5	two least significant bytes of the
2F4	48			PHA		software clock and E6 is the
2F5	29	0F		AND	#F	index for storing the unpacked
2F7	A6	E6		LDX	E6	number and display digits
2F9	95	E7		STA	E7, X	
2FB	20	96	02	JSR	296	Convert to display format
2FE	95	D1		STA	D1, X	Store in display locations (D1-D4)
300	E8			INX		
301	68			PLA		Unpack high order digit
302	4A			LSR		
303	4A			LSR		
304	4A			LSR		
305	4A			LSR		
306	29	0F		AND	#F	
308	95	E7		STA	E7, X	Save in E7-EA
30A	20	96	02	JSR	296	Convert to display format
30D	95	D1		STA	D1, X	Store
30F	E8			INX		
310	86	E6		STX	E6	
312	A6	E5		LDX	E5	Loop until E5 = 0
314	CA			DEX		
315	D0	D9		BNE	2F0	
317	A2	05		LDX	#5	Clear D5-DA (Last two digits
319	A9	00		LDA	#00	of the display and first
31B	95	D4		STA	D4, X	three bytes of software clock)
31D	CA			DEX		
31E	D0	FB		BNE	31B	
320	60			RTS		Return to Exec

set to the next mode; that of isplaying the random number (the least significant four digits of the real-time clock). This number is set up in the *first* four digits of the seven-segment display and is unpacked into four four-bit characters for easier use during the keyboard input portion of the program. The clock is then reset to zero in preparation for waiting out the two second display-duration delay period. Finally, the last two characters of the display are cleared.

display mode

In the display mode **(fig. 7 — listing 6)** the first check is to determine whether or not the two second display interval has elapsed. If not, the program simply returns to the real-time executive. If two seconds have elapsed, the display is cleared and a test is made to determine whether or not any key on the keyboard is currently depressed.

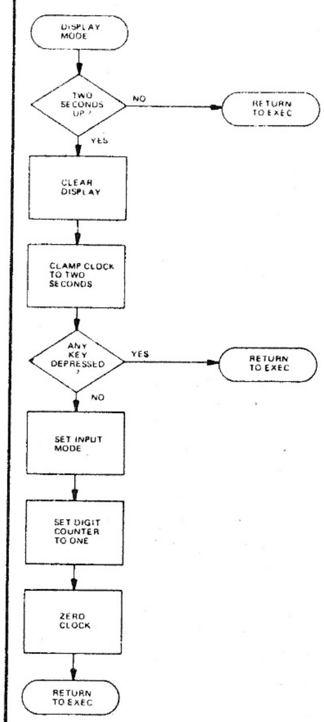
listing 6. Display Mode (E3 = 1)

```
LOC  B1  B2  B3  OP   ARG    COMMENTS
328  A5  D8      LDA  D8     Check for 2 seconds up
 \   C9  20      CMP  #20
32C  F0  01      BEQ  #1
32E  60          RTS         If not, return to exec
32F  A2  07      LDX  #7     Clear display and clamp clock
331  A9  00      LDA  #00      to 2 seconds
333  95  D0      STA  D0, X
335  CA          DEX
336  D0  FB      BNE  333
38   A5  E2      LDA  E2     Check for no key depressed
                               (E2 = FF)
33A  30  01      BMI  #1
33C  60          RTS         If one is depressed, return to Exec
33D  A9  02      LDA  #02    Set mode to input (E3 = 2)
33F  85  E3      STA  E3
341  A9  01      LDA  #01    Set input digit counter to 1
343  85  E4      STA  E4
345  A9  00      LDA  #00    Clear second and third bytes of the
                               software clock
342  85  D8      STA  D8
     85  D9      STA  D9
34B  60          RTS
```

LOCATION	RANGE	USAGE
E1	0-15 or FF	Numeric value of key depressed during last time slice (FF if none)
E2	0-15 or FF	Numeric value of key depressed during current time slice
E3	0-2	Current software mode 0 = Ready 1 = Display 2 = Input
E4	1-4	Input digit counter
E5-E6	—	Temporary index storages
E7-EA	0-9	Unpacked digits of random number

table 2. Variables used by drunk test.

fig. 7. Display Mode flowchart.

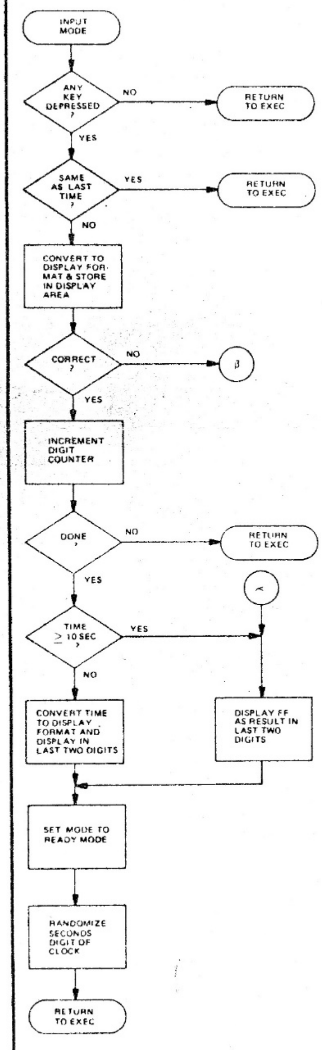


If so, the game is *not* allowed to proceed until that key is released. To hold off the start of the timing function, the clock is clamped to two seconds while the key is depressed and is not released until after the program has detected that no key is currently depressed. When the keyboard is clear the mode is set to the input mode, the counter for the four digits is set to one, and the clock is again set to zero.

When the program is waiting for input **(fig. 8 — listing 7)** it first checks to see if any key is depressed. If not, it returns and waits out the one millisecond interval. If a key has been depressed, a test is made to see if the same key was depressed during the last cycle. If so, the program returns to the executive because the key has already been entered into the display. If it is a new digit, the digit is added to the display and then checked against the correct random number to make sure that no error has been made. If an error has occured, FF is immediately displayed in the time digits and the program returns to the ready mode and waits for the "+" key to again be depressed. If no error is made, the digit counter is incremented and if four digits have already been entered, the results can be displayed. Should those results exceed ten seconds, FF is displayed instead. Otherwise the seconds and tenths of seconds characters of the software clock are converted to the display format and entered into the last two digits of the display.

In conclusion, through developing the above program and others for the KIM-1, the author has found this microcomputer to be very convenient and natural to use. Without a doubt, it is the most complete system available in its price range. **END**

fig. 8. Input Mode flowchart.



listing 7. Waiting For Input (E3 = 2)

LOC	B1	B2	B3	OP	ARG	COMMENTS
350	A5	E2		LDA	E2	Pick up current keyboard status
352	10	01		BPL	#1	Test for negative
354	60			RTS		If negative (FF) return
355	C5	E1		CMP	E1	Compare with last scan
357	D0	01		BNE	#1	
359	60			RTS		If the same, return
35A	20	96	02	JSR	296	New digit, display it
35D	A6	E4		LDX	E4	Store using digit counter as the index
35F	95	D0		STA	D0, X	
361	B5	E6		LDA	E6, X	Compare with random number
363	C5	E2		CMP	E2	
365	D0	1D		BNE	384	If not the same, display FF result
367	E8			INX		Increment digit counter
368	86	E4		STX	E4	
36A	E0	05		CPX	#5	Done?
36C	F0	01		BEQ	#1	
36E	60			RTS		No, return to exec
						DONE
36F	A5	D9		LDA	D9	Time ≥10 sec
371	D0	11		BNE	384	If so, display FF result
373	A5	D8		LDA	D8	Time <10 sec, display
375	20	92	02	JSR	292	Convert to display format and store in last two digits
378	85	D5		STA	D5	
37A	A5	D8		LDA	D8	
37C	20	96	02	JSR	296	
37F	85	D6		STA	D6	
381	4C	8A	03	JMP	38A	
384	A9	71		LDA	71	Error or ≥10 sec, display FF
386	85	D5		STA	D5	
388	85	D6		STA	D6	
38A	A9	00		LDA	00	Set mode to ready (E3 = 0)
38C	85	E3		STA	E3	
38E	A5	D7		LDA	D7	Randomize seconds digit of clock
390	85	D8		STA	D8	
392	60			RTS		Return to exec