IMSAI 8080 System
General Assembly and
Test Instructions

Edited from the original ***IMSAI Microcomputer System User Manual*** (8/77) and formatted for Adobe Acrobat PDF 9/8/99

SYSTEM FUNCTIONAL TEST

Turn the power on with the front panel rocker switch and depress the RUN/STOP switch momentarily to STOP position and release. The WAIT light should be on and the RUN and HOLD lights should be off, with the other lights in various states at this time. Raise the RESET switch momentarily to the RESET position and release. All the lights on the bottom row in the ADDRESS BUS section should be indicating that the program counter is set to location 0. The WAIT light should still be on with the RUN and HOLD lights off. The DATA BUS lights may show various random bits on, and the STATUS byte should have three lights on: MEMR, Ml, and /WO. With all 16 ADDRESS switches in the down or 0 position, the EXAMINE/EXAMINE NEXT switch should be raised momentarily to the EXAMINE position and released. Check that the lights after this operation are exactly the same as described for after the RESET switch was operated.

The machine is now ready to enter a small test program. For complete description of program operation in computers, read **"An Introduction To Microcomputers Volume 1: Basic concepts" by Adam Osborne (1976, 1977 Osborne and Associates, Berkeley, California)**. For the initial machine test, the following program should be entered:

TEST PROGRAM 1

OCTAL		HEX			
ADDRESS	INSTRUCTION	ADDRESS	INSTRUCTION	BINARY	DESCRIPTION
000 000	333	0000	DB	1101 1011	INPUT
001	377	01	FF	1111 1111	ADDRESS
002	323	02	D3	1101 0011	OUTPUT
003	377	03	FF	1111 1111	ADDRESS
004	303	04	C3	1100 0011	JUMP
005	000	05	00	0000 0000	LOW ADDRESS
006	000	06	00	0000 0000	HIGH ADDRESS

TEST PROGRAM 2					
OCTAL		HEX			
ADDRESS	INSTRUCTION	ADDRESS	INSTRUCTION	BINARY	DESCRIPTION
000 000	333	0000	DB	1101 1011	INPUT
001	377	01	FF	1111 1111	ADDRESS
002	057	02	2F	0010 1111	COMPLEMENT DATA
003	323	03	D3	1101 0011	OUTPUT
004	377	04	FF	1111 1111	ADDRESS
005	303	05	C3	1100 0011	JUMP
006	000	06	00	0000 0000	LOW ADDRESS
007	000	07	00	0000 0000	HIGH ADDRESS

The address is now at 0 as indicated by the lights labeled ADDRESS BUS. Into position 0 we wish to put an input instruction.

The bit pattern for the input instruction must be set in the center group of switches labeled ADDRESS-DATA. Switches 7, 6, 4, 3, 1 and 0 should be placed in the up position. Compare these switch positions with the binary representation of the input instruction listed on the first line of test program 1. We wish now to deposit this bit pattern in memory position 0 - Raise the DEPOSIT/DEPOSIT NEXT switch up momentarily to the DEPOSIT position and release. The address bus, should still show 0 (no lights lit) and the data bus should now show the bit pattern set in the switches (bits 7, 6, 4, 3, 1 and 0 lit and bits 5 and 2 off).

Next, the bit pattern for the address of the input port should be written in position 1. This can be done by setting all eight ADDRESS-DATA switches up, corresponding with the address listed on line 2 of Test Program One and the DEPOSIT/DEPOSIT NEXT switch depressed momentarily to the DEPOSIT NEXT position and released.

Now the address bus light should show position 1 (address bus light 0 on and all other address bus lights off). The data bus should show all eight lights lit corresponding to the bit pattern written here. Similarly, the next five lines of Test Program One should be set into the ADDRESS-DATA switches and deposited by operating the DEPOSIT NEXT switch, each time checking to make sure that the data bus lights correspond with the settings of the ADDRESS-DATA switches, and that the address is correct indicating that no steps have been skipped or done twice.

When the last byte has been deposited in address position 6, then all 16 address switches should be returned to the 0 position (down) and the EXAMINE switch operated. This should reset the address bus lights to 0, and display the contents of the bottom word in memory on the data bus lights. (This should still be the binary pattern listed in line 1 of the Test Program). The EXAMINE NEXT switch can then be operated and the address bus lights should indicate address 1 (bit 0 on and all other bits off). The Data Bus should show the contents now of memory location one which should correspond to the second line of Test Program One listing (all ones).

The EXAMINE NEXT switch can be repeatedly operated, each time checking that the data located in the consecutive memory location corresponds exactly to the listing for Test Program One.

The EXAMINE switch can again be raised momentarily with the address switches all down, to return -the machine to position 0, once it has been determined that all lines listed in Test Program One are stored correctly in the memory.

Now we can single-step through this program and watch the operation of the machine. With the machine sitting at 0 with the correct instruction on the data bus, and the MEMR, MI and /WO lights lit in the status byte, the processor is reading the first instruction out of memory into the processor for execution. If the SINGLE-STEP switch is either depressed or raised once, it will permit the processor to complete its cycle and begin the next cycle. The address bus lights will show position 1, the data bus will show all ones corresponding to the bit pattern in the Test Program, and the status byte will show MEMORY READ and WO-. The lack of an M1 light in a status byte indicates that the processor is no longer fetching an instruction to execute, but rather this cycle it is fetching the address for the instruction, which it has already stored internally.
If the SINGLE-STEP switch is operated once again, the address bus lights will all be lit. The status byte will show INP and /WO and the data bus will at first show no lights on. If one or more switches in the left hand group of eight switches is now raised or lowered, the corresponding light on the data bus indicators will turn on or off. The processor is now executing the first instruction which was an input data from address FF hex (377 octal) which is the address for the programmed input port on the front panel. By means of this instruction with this address the processor is able to read the position of the eight switches in the left-hand group. (The address being read is indicated by the lights in the address bus and, on input or output instructions, the address appears in both groups of eight lights on the address bus. Thus, for this address, all the lights in the address bus are lit.)

The switches in the left-hand group should be left in the position of some up and some down to provide a recognizable pattern before continuing. With the pattern left in the left-hand group of switches, the single step switch can be operated once more permitting the processor to complete the execution of the input instruction, and begin the next cycle. Having completed the input instruction, the next cycle will be a fetch cycle, during which the processor reads the next instruction to be executed, which it will find in memory address position 2. The address bus lights should now show position 2 (bit 1 on and all others off), and the data bus should indicate the bit pattern listed on line 3 of Test Program 1 for address position 2. This is the output instruction.

The Status Byte will again have MEMR, M1, and /WO lights lit and the others off. When the single-step switch is operated once again, the processor is permitted to complete the cycle during which it reads in the output instruction and begin the next cycle during which it will read the address of the output device. Since it is reading this address from the next memory position, (memory position 3), the address bus will have bits one and 0-ion and the others off. The Data Bus will have all lights on indicating the bit pattern we stored in memory position 3. The status bit will show MEMORY READ and WRITE OUT lights on, and the Ml light is off at this time, indicating that this is not an instruction fetch cycle, but rather it is one of the cycles required to execute the last instruction fetched, in this case, reading the address to which the data will be output. When the SINGLE STEP switch is operated once again, the processor is permitted to complete the cycle of reading the output address in and begin the next cycle, which is the output operation.

The output operation looks similar to the input in that the address of the output device appears in both the upper and lower half of the Address Bus, (again in this case lighting all the lights), and the data being output appears in the Data Bus, which should show the pattern previously set in the left hand group of switches. Since the data is being output from the accumulator in the processor where it was previously stored in the input instruction, it will not be affected by moving the switches in the left-hand group at this time. The Status Byte shows the MEMR light off at this time and shows the out light on indicating that the processor is executing an output instruction. The /WO light is off indicating that the processor's WRITE strobe is active. If the SINGLE STEP switch is operated once more, it will permit the processor to complete the WRITE operation and begin the next cycle. At this time, the PROGRAMMED OUTPUT lights at the top left of the panel should be lit according to the complement of the pattern that was set in the switches. That is, for each switch that was set in the up position, the light will be out, and each switch that was set in the down position, the corresponding light will be on.

Since the processor has completed the output instruction the next cycle is used to fetch the next instruction to be executed, which it will read from memory position 4. In memory position 4 we had stored the jump instruction, which should now appear on the lights on the data 15us indicators. As the SINGLE STEP switch is operated again, permitting the processor to

complete the fetch of the jump instruction, and start the next cycle of executing that jump instruction, we find that the processor is reading the low half of the address from memory position 5. The status byte shows the MEMR and /WO lights lit, and the Ml light is off at this time.

If the SINGLE STEP-switch is operated once again, it will be seen that the processor is reading the high address byte previously stored in memory location 6.

The next operation of the SINGLE STEP switch permits the processor to complete the execution of that jump instruction, which is instructing the processor to take its next instruction to be executed, not from memory position 7, but from memory position 0, as was stored in the two bytes following the jump instruction.

The Address Bus lights should now be all off indicating that the processor is indeed fetching the next instruction from memory location 0. The Data Bus should show the pattern that we wrote in memory position 0 as the input instruction. We have now completed one cycle of the loop in Test Program 1. Further operations of the SINGLE STEP switch will let the processor step through the execution of the loop additional times and each time through the loop it is possible to set a different pattern in the left hand group of switches to be read in and later to be written out to the PROGRAMMED OUTPUT lights. The RUN/STOP switch can be momentarily raised to the RUN position and released. This will permit the processor to run at the full clock speed, which will result in the loop being executed roughly 50,000 times every second. Thus, as any of the switches in the left hand group of eight are moved while the program is running, the machine reads the new position essentially instantly and displays it in the PROGRAMMED OUTPUT port above.

It may have been puzzling that the lights in the PROGRAMMED OUTPUT port seem to indicate the opposite of what might have been expected when a bit was read in as a 1 and output to the PROGRAMMED OUTPUT port. This will serve as an example of the way logic design has been affected by the appearance of large-scale integration and microprocessors. While it would have been entirely possible and easy to provide a circuit modification such that when the data was put out as a 1 the light would be lit rather that turned off, such as addition to the circuit would have cost you more that the cost for byte of memory. The same function as the added circuit can be accomplished by adding one instruction to the loop which complements the data, that is, changes all ones to O's and all O's to l's. Test Program 2 is exactly the same as Test Program 1 with the addition of one instruction between the input instruction and the output instruction, which will complement the data read in from the switches before it is output. If the machine is stopped and reset, Test Program 2 may be entered exactly the same way as Test Program 1 was and checked and then run through one or more cycles with the operation of the machine and to double-check that the program truly has been entered correctly. Then the RUN switch may be actuated to permit the loop to run at high speed.

With this change in the program, the PROGRAMMED OUTPUT port will show a light lit when the switch is positioned up to enter a 1 bit. Not only is this a less expensive way to achieve the function of causing the lights to turn on when the bit is entered as a 1, but it is a much more versatile solution since the operator can change his mind at a later date and either remove the complement instruction or change it to yet another instruction for a different result.

When single stepping through Test Program 2, the compliment data instruction is seen to use up only one cycle of the processor. We are able to see it being fetched to be executed, and when the SINGLE STEP switch is operated again, we are immediately fetching the next instruction. This will be true of any instructions that operate only on data, which is already stored within the processor. Additional cycles are only necessary if additional information must be read in or out of the program processor itself.

After either loop is running, the RUN/STOP switch may be depressed to STOP at any time and the operation processor will stop during the fetch of the neat instruction. Due to the speed at which the processor operates, it is impossible to tell beforehand at what point in the loop the processor will be at the exact instant that the RUN/STOP switch is moved to STOP, so that the processor will stop at different places in the loop for different times when the switch is actuated.

The switch may be raised to the RUN position starting at any point in the loop and the processor will continue to run at high-speed beginning at that point. The flip-flop set by the RUN/STOP switch simply instructs the processor to wait at each cycle for a pulse which is generated by the SINGLE STEP switch to be received before executing the next cycle, and apart from waiting for this pulse, the processor executes exactly the same whether it is in the single run mode or stop made.

The definition of a computer involves both the ability to execute in sequence of instructions which is stored inside the machine, and the ability to make a decision between on the value of data and, as a result of that decision, choose between alternate possible paths of program step sequences to execute. Test programs 1 and 2 involve only the execution of a sequence of stored program steps and do not involve any decisions. Program 3 will illustrate the use of decisions in a computer program and should provide some interesting entertainment as well. It is a game program using the INPUT switches and the PROGRAMMED OUTPUT lights on the IMSAI 8080 front panel.

A pattern of lights in the PROGRAMMED OUTPUT ports is moved to the left one bit at a time, and the left hand bit which is "pushed off" the end of the programmed I/0 register reappears at the right end of the register. The rate at which the bit pattern is shifted to the left can be chosen by the binary number set in the front panel switches when the program is first started or when the machine is reset to start again.

When a higher binary number is entered in these switches and the program restarted, the bit pattern will shift to the left at a higher rate of speed. Initially, switches should be set for 2, that is all switches down except PROGRAMMED INPUT switch bit 1 on, in order that the bit pattern will be shifted slowly enough to easily see what the game program is doing.

Once the program has been started, any further movement of the front panel switches does not affect the rate at which the bit pattern is shifted to the left. From this time on, any time one of the eight switches in the PROGRAMMED I/0 is changed, then the bit in the PROGRAMMED OUTPUT port which is directly above that switch at the moment was moved, will change. If it was off before, it will turn on; and if it was on before, it will turn off. The direction of travel of the switch is not significant- only that its position was changed. After a switch change is detected, and the light above it turned on or off as appropriate, no further switch movements will affect the condition of any of the lights until the next shift to the left has occurred. This was done to give the switches time to stop bouncing and stay closed, as the processor in this machine is quite fast enough to see the slight bouncing of the switch contact when it initially closes.

By waiting for the next data shift before recognizing any more switch changes, we are prevented from falsely interpreting a bouncing contact as a switch, which was repeatedly opened and closed. The object of the game can be either to turn out all the lights in the shifting bit pattern by moving a switch when the bits are passing directly over it, or alternately to turn on all the bits in the shifting bit pattern by moving a switch when a bit which is off is directly over it. Any time the shifting bit pattern is all 0's or all 1's, no movement will be seen in the PROGRAMMED OUTPUT port but, by moving any switch, one of the lights will be changed so that the motion is again apparent.

Players can compete for the shortest time to go from all 0's to all 1's, or the other way - from all l's to all 0's. When the game has been mastered at one rotation speed, the switches can be set for a higher binary number and the system reset to cause the processor to go back to memory location 0 and begin execution of the program again, and a new switch setting will be read to result in a higher rate of rotation, which makes it harder to move a switch at the exact instant the bit desired to be changed is directly above it. If there were only a single light on and circulating across the output port, and the player, (in attempting to turn it off by moving the switch when the bit was directly over that switch) was too slow, then the bit will have shifted away so that it is now over the next switch to the left, not only will that bit not be turned off, but the bit behind will be turned on so that now there are two bits circulating across the register and the player is further away from achieving all bits turned off.

Knowledge of some of the internal structure of the 8080 processor will be necessary to understand the game program. The Intel data book contains complete information and functional specifications on the internal structure of the 8080 processor, but only the basic aspects of the structure need be known to understand the program operation.

Figure 1 shows the structural blocks in the processor, which are important to the programmer. Central to the processor's operation is the register named the ACCUMULATOR. This register and all the others is like one eight bit position in memory or a small "blackboard" with room for only eight bits of either l's or 0's to be written. When the input instruction was executed during programs I and 2, the pattern from the switches on the front panel was read into the ACCUMULATOR register, and when the OUTPUT instruction was given it was again the contents of the ACCUMULATOR which was output to the PROGRAMMED OUTPUT port on the front panel. All arithmetic is done in the ACCUMULATOR and, except for special instructions (to permit other registers to be read to or from memory), all programmed input/output from either memory

or input/out interfaces goes to and from the ACCUMULATOR. The INSTRUCTION register is another "blackboard" (or pointer) with room to store the address where it last read a program byte from memory so that when it finished the execution of that step, it can increment that address by one and use it to determine where to get the next instruction.

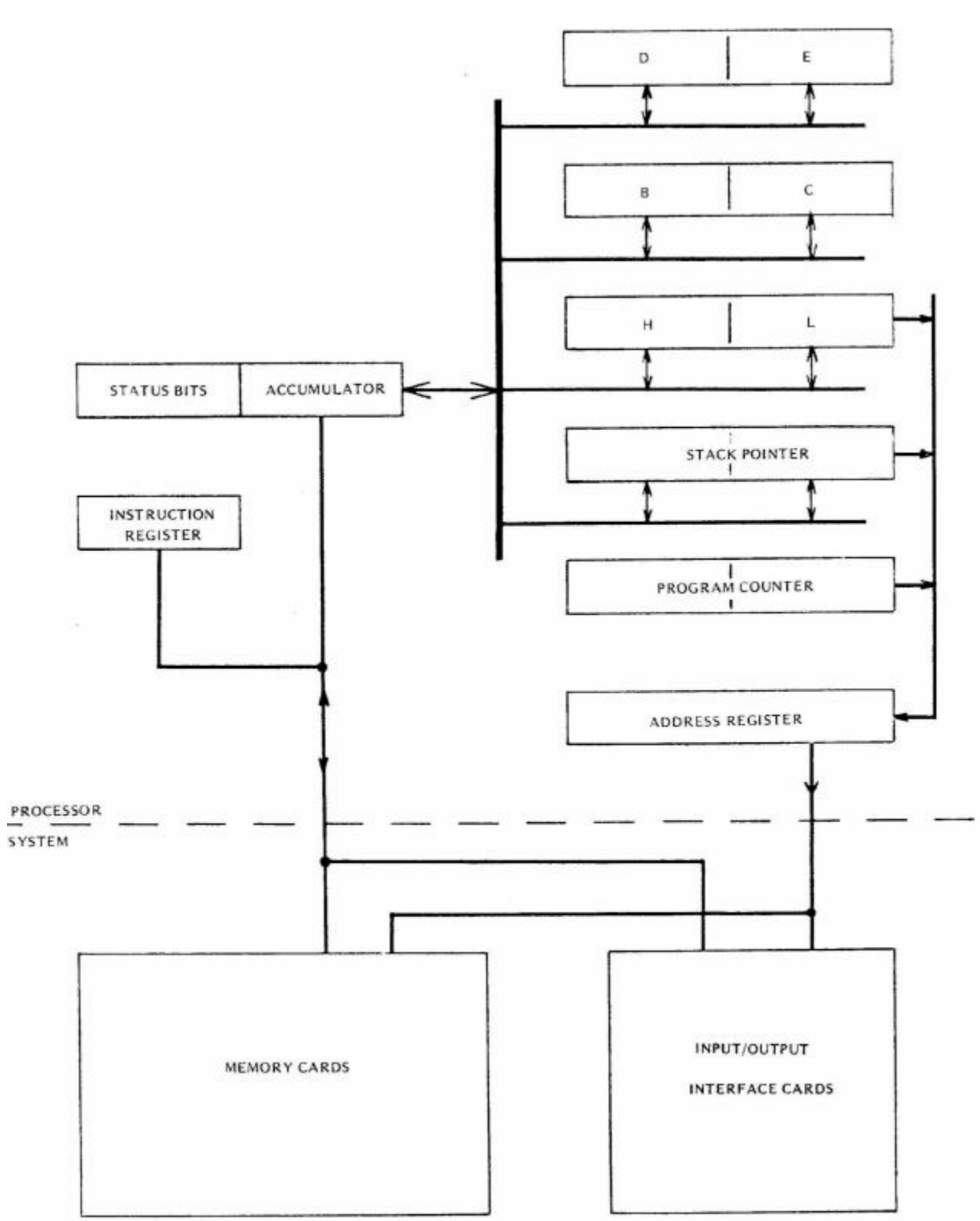
The STATUS BITS are 5 bits that are set to 1 or 0, according to the results of the last data operation performed in the ACCUMULATOR. One of the STATUS BITS or condition flags is the Z bit (zero bit) which is turned on when the last operation in the ACCUMULATOR resulted in the ACCUMULATOR being left all O's. Otherwise, this bit is turned off. The second condition flag is the sign bit. If the most significant bit of the result of the last operation in the ACCUMULATOR has the value 1 this flag is set to 1, otherwise it is reset to 0. Three other condition flags are the sign parity and the auxiliary carry, and their functions are described in the ***Intel Data Book*** on page 4-2. The fifth condition flag is a carry flag which is turned on if the last arithmetic operation produced an overflow. An overflow is produced, for example, when two numbers are added together and their sum is too large to be contained in the register into which it is put. For instance, if the ACCUMULATOR contained eight l's and another number was added which contained the value 6, the correct answer would be the combination of the value 5 and a bit turned on in the 9th position. Since the ACCUMULATOR has only eight positions, the carry bit would be turned on.

Some of the STATUS BITS are affected by the operations in other registers than the ACCUMULATOR. For instance, the carry bit is affected by additions made in the H and L registers by using the double add instructions. Use is made of this in the game program. There are five other registers in the processor, each of which is 16 bits long, and some of which are divided in half so that operations may be done with only 1/2 at a time. The ADDRESS REGISTER is a 16-bit register over which the programmer has no control. It is simply used to output either the memory address or the input/output address necessary to execute the next cycle. The programmer can use all the other four 16-bit registers. There are many instructions in the 8080A processor's instruction set whose function is to move data from any register to any other register, to permit arithmetic operations between a register and the ACCUMULATOR (with the result always being left in the ACCUMULATOR), and some special instructions to permit direct transfer of data from memory to a register, or vice versa.

The B, C, D, and E half-registers are all general-purpose registers. The H and L register pair and the STACK POINTER register pair both have special functions in addition to being usable for general purposes. The game program does not make use of these special functions.

With the basic structure of the processor in mind, we can now look at the operation of the game program. Larger programs cannot be readily understood or written by working directly on the list of machine instructions, such as we did for Test Programs 1 and 2. A flow diagram is essential to quickly follow the sequence of the instructions and understand how they work together to achieve the desired result.

Figure 2 shows a flow diagram for Program 2. Each program function is briefly described in a separate box, and the lines indicate the flow of the executive of the program. Test Program 2 was a simple loop with no decisions so that after executing the short sequence of instructions, the flow of the program is back to the beginning of the loop to begin again. Figure 3 shows the flow diagram for the game program. Although it need not be understood to execute the game program, a thorough understanding of how this flow diagram achieves the operation of the game will be a useful step towards writing your own programs.



8080 PROCESSOR
FIG. 1

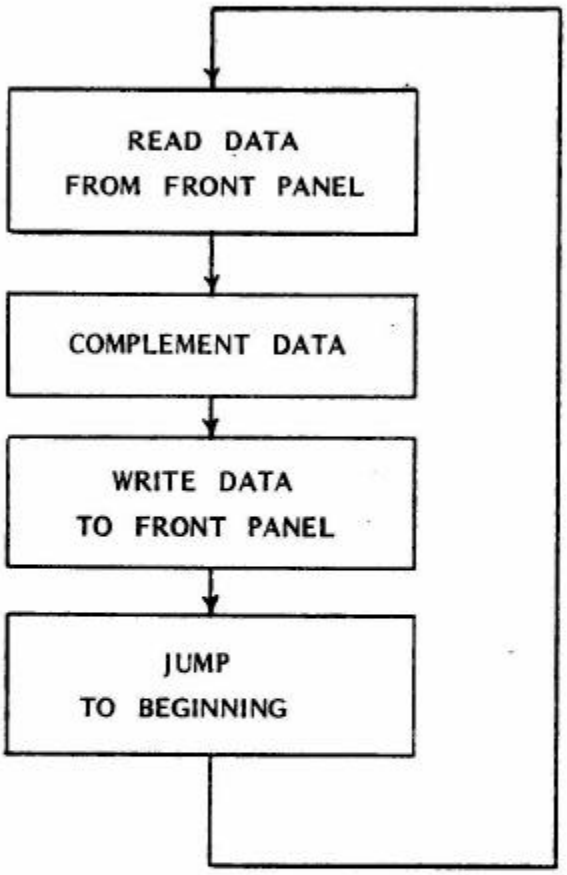

READ DATA
FROM FRONT PANEL
COMPLEMENT DATA
WRITE DATA
TO FRONT PANEL
JUMP
TO BEGINNING

FIG. 2

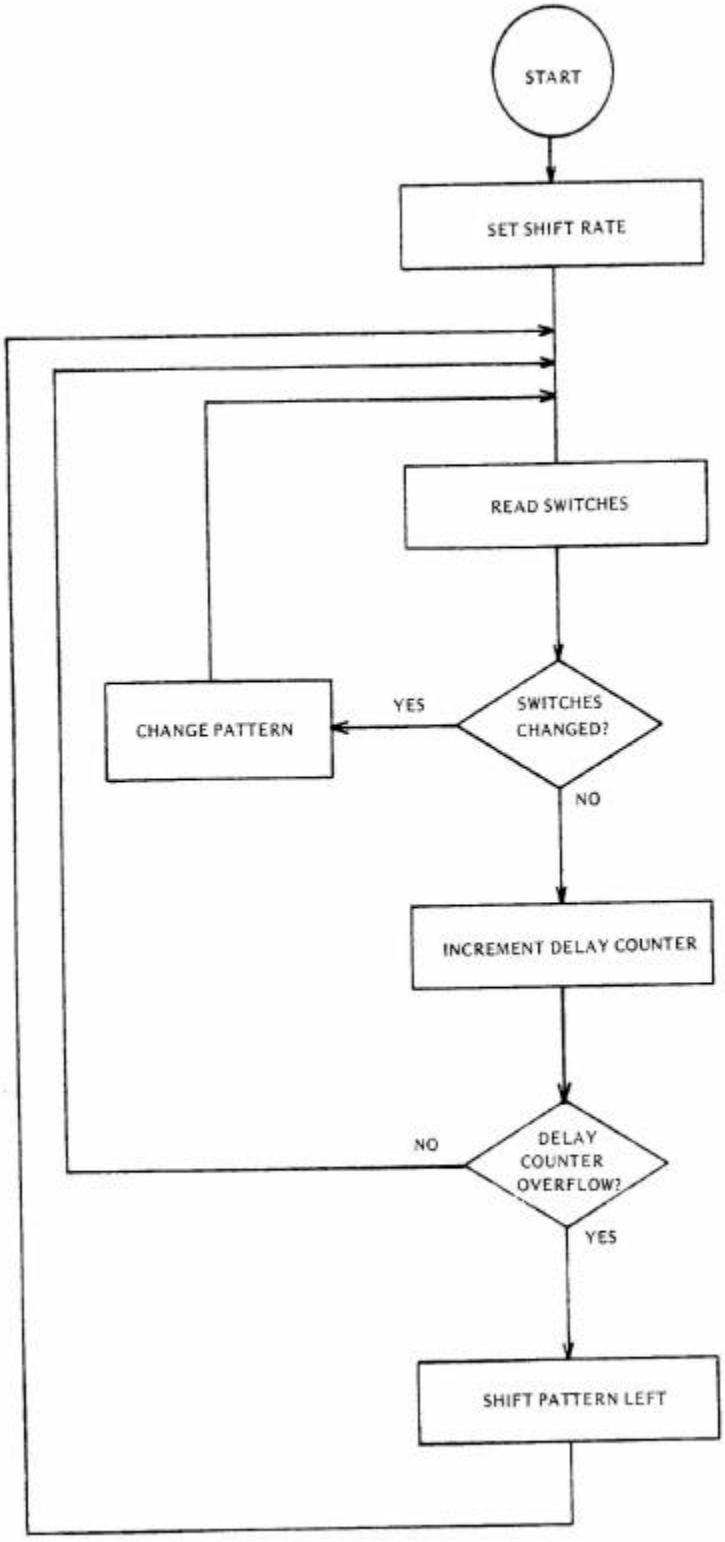


FIG. 3

GAME PROGRAM BASIC FLOWCHART

"Kill the Lights" GAME PROGRAM LISTING					
OCTAL		**HEX**			
ADDRESS	**INSTRUCTION**	**ADDRESS**	**INSTRUCTION**	**MNEMONIC**	**DESCRIPTION**
000 000	257	0000	AF	XRA,A	Exclusive OR A to itself (put zero in A)
001	147	01	67	MOV H, A	Move A to H (put zero in H)
002	333	02	DB	INP	Input data
003	377	03	FF		from front panel switches
004	157	04	6F	MOV L, A	Move A to L
005	371	05	F9	SPHL	Put H&L reg. into SP
006	257	06	AF	XRA, A	Exclusive OR A to itself (put zero in A)
007	201	07	81	ADD C	Put C in A, affecting flag bits
010	302	08	C2	JNZ	Jump if not zero
011	023	09	13		(skip switch test for debounce after a switch change)
012	000	0A	00		
					IF NORMAL, CONTINUE
013	123	0B	53	MOV D, E	Move E to D
014	333	0C	DB	INP	Input data
015	377	0D	FF		from front panel switches
016	137	OE	5F	MOV E,A	Move A to E
017	252	OF	AA	XRA, D	Exclusive OR D to A
020	302	10	C2	JNZ	jump if result not all 0's
021	041	11	21		(change display if switch position changed from last time)
022	000	12	00		

					IF SWITCHES UNCHANGED, CONTINUE
023	071	13	39	DAD SP	Add SP to HL
024	322	14	D2	JNC	jump if no carry results
025	006	15	06		(return to read switch loop if no carry yet)
026	000	16	00		
					IF CARRY, CONTINUE
027	170	17	78	MOV A, B	Move B to A
030	007	18	07	RLC	Rotate left 1
031	107	19	47	MOV B, A	Store A in B
032	323	1A	D3	OUT	Output A
033	377	1B	FF		in front panel lights
034	257	1C	AF	XRA,A	Exclusive OR A to itself (put zero in A)
035	117	1D	4F	MOV C, A	Move A to C (Reset debounce indicator)
036	303	1E	C3	JMP	Jump
037	006	1F	06		(to read loop)
040	000	20	00		
					CHANGE DISPLAY IF SWITCH DIFFERENT
041	250	21	A8	XRA, B	Exclusive OR B with A
042	107	22	47	MOV B, A	Store A in B
043	323	23	D3	OUT	Output A
044	377	24	FF		in front panel lights
045	257	25	AF	XRA, A	Exclusive OR a with itself A (put zero in A)

046	147	26	67	MOV H, A	Move A to H (set counter to insure enough delay for debounce)
047	057	27	2F	CMA	Complement A (to all 1's)
050	117	28	4F	MOV C, A	Move A to C (set C to debounce)
051	303	29	C3	JMP	jump
052	006	2A	06		(to read loop)
053	000	2B	00		

NOTE:

Exclusive OR of two switch patterns results in 1's in positions, which were changed, with all 0's elsewhere

B= Display byte storage

C= Switch debounce indicator:
1= debounce 0= normal operation

D= Last switch settings

E= Current switch settings

H,L= Delay counter

SP= Increment for delay counter