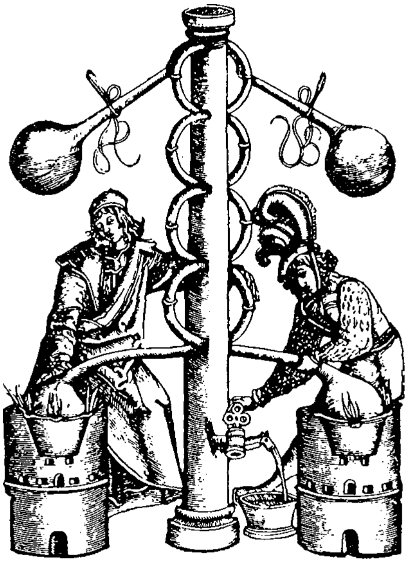
Reprinted with permission from American Laboratory, Vol. 9, No. 3, pp. 108-110.



Microcomputer interfacing

The 8080 logical instructions

By Jonathan A. Titus, David G. Larsen, and Peter R. Rony

THE CONCEPT of the one important use for multibit logical instructions, such as AND, OR, Exclusive-OR, and COMPLEMENT, has been discussed.[1] In this column the twenty-eight logical instructions in the 8080A instruction set are summarized. It is very important to note that in the case of each logical instruction, *the result* is *stored in the accumulator.* The previous contents of the accumulator is one of the logical variables in the two-variable logical operation, or in the case of the complement instruction, the only logical variable.

The eight different logical AND instructions, each with the mnemonic ANA S, have the following general form:

10	100	*sss*
Arithmetic and logical class of **instructions**	AND operation	3-bit binary code for source register

where sss are the three bits that correspond to the register or contents of a memory location that logically operate on the accumulator contents,

Register	Octal code	3-bit register code
B	O	000
C	I	001
D	2	010
E	3	011
H	4	100
L	5	101
M	6	110
A	7	111

The OR and Exclusive-on instructions, which have the mnemonics ORA Sand XRA S, respectively, have the same general form as the ANA S instruction byte. Thus, for the XRA S instruction the instruction byte is

10	101	*sss*
Arithmetic and logical class of instructions	Exclusive-OR operation	3-bit binary code for source register

and for the ORA S instruction,

10	I 10	sss
Arithmetic and logical class of instructions	OR operation	3-bit binary code for source register

Some examples are:

Logical operation	Mnemonic	Octal instruction code
B • A→A	ANAB	240
M • A→A	ANAM	246
A • A→A	ANAA	247
C ⊕ A→A	XRAC	251
L ⊕ A→A	XRAL	255
A ⊕ A→A	XRAA	257
D + A→A	ORAD	262
E + A→A	ORAE	263
M + A→A	ORAM	266
A + A→A	ORAA	267

Another logical instruction is the complement accumulator instruction, which has the mnemonic CMA A and the octal instruction byte 057.

In previous columns,[2,3] we discussed the concept of an *immediate instruction*, a rnultibyte instruction that contains the desired data within the instruction. The three immediate logical operations can be summarized in the following way:

Logical operation	Mnemonic	Octal instruction code
<B2> • A→A	ANI <B2>	346 <B2>
<B2> ⊕ A→A	XRI <B2>	356 <B2>
<B2> + A→A	ORI <B2>	366 <B2>

In the preceding examples, the symbol → means "is replaced by." Thus, the notation B• A→A means that we AND the variable B with the variable A, and then replace the original contents of A by the result of the logical operation. Within the 8080A microprocessor chip, the logical operation is performed in a temporary accumulator, with the logical result in the temporary accumulator being *copied* into the accumulator register, A.

We have demonstrated one use for logical instructions: the testing of flag or comparator bits associated

Mr. Titus is President, Tychon, Inc., Dr. Rony, Department of Chemical Engineering, and Mr. Larsen, Department 01 Chemistry, are with the Virginia Polytechnic Institute and State University.

Table 1

"Stripping" program

La memory address	Octal instruction code	Mnemonic	Comments
000	333	IN	Input ASCII numbers from the following device
001	015	015	Device 015
002	346	ANI	AND the accumulator contents with the following data byte
003	017	017	Mask byte that "masks" the most significant four bits in the ASCII word

Table 2

"Packing" program

LO memory address	Octal instruction code	Mnemonic	Comments
000	333	IN	Input ASCII "5" from the following device
001	015	015	Device 015
002	346	ANI	Mask off the four most significant bits
003	017	017	Mask byte
004	007	RLC	Rotate the BCD digit into the four most significant bits that have just been cleared
005	007	RLC	
006	007	RLC	
007	007	RLC	
010	107	MOVB,A	Store this result in register B
011	333	IN	Input next ACSII character, in this case ASCII "7", from the following device
012	015	015	Device 015
013	346	ANI	Mask off the four most significant bits
014	017	017	Mask byte
015	260	ORAB	OR contents of register B with contents of accumulator
016	167	MOVM,A	Store packed data into memory, the location being specified by the contents of the H,L register pair

with the on/off state of external devices.' The AND multibit operation is particularly useful when we want to clear, filter, or *mask* specific bits in an input data byte. For example, consider the ASCII code for the numeric characters 0 through 9:

Character	Octal ASCII code	Binary ASCII code
0	260	10110000
I	261	10 110001
2	262	10110010
3	263	10110011
4	264	10110100
5	265	10110101
6	266	10110110
7	267	10110 III
8	270	10111000
9	271	10 III 001

Once we input the ASCII code into the microcomputer, the most significant four bits are of little use and can be "stripped" away from the data byte. A simple program that accomplishes such a task is shown in *Table* 1. This program accomplishes the following Boolean operation for ASCII"5":

10110101 • 00001111		00000101
ASCII "5"	Mask byte	BCD data of interest

The logical result of the AND operation is 00000101. We now have a single BCD digit per input data byte, with the BCD digit being the least significant four bits in the byte, D3-DO. The remaining four bits, D7-D4, can be used to store another BCD digit provided that we have some means to position this second digit in these open-bit positions. if we do not take advantage of these four bits of storage space, we will waste 50% of our memory storage space.

To "pack" two BCD digits into a single data byte, we must have the capability to rotate the contents of the accumulator. We use the rotate left instruction, which has the mnemonic RLC and the octal instruction byte 007, and which can be described as follows: "The content of the accumulator is rotated left one position. The low-order bit and the carry flag are both sent to the value shifted out of the high-order bit position.'" The four rotate instructions in the 8080 instruction set have been previously shown.' The accumulator is the only register that can be rotated in an 8080A chip. Other registers are rotated simply by moving them to the accumulator register, performing the necessary rotation operations, and then returning the rotated byte back to the original register. Besides shifting BCD digits back and forth in data bytes, we will also discover important uses for the rotate instructions when we discuss decision-making operations.

A simple program that can be used to "pack" two BCD digits into a single data byte is given in *Table 2.* The result of this sequence of steps is the data byte 01010111 stored in memory. The four most significant bits are BCD "5," and the four least significant bits are BCD "7." Observe the use of the ORA B instruction, which permits the combination of the two data bytes into one without changing either. Special 8080 microcomputer programs, called simulators, are available that permit the user to follow the execution of an 8080 program step by step by observing the changes in the contents of the internal registers. (One such program, called DEBUG, has been developed by Tychon, Inc. It requires the use of a Teletype or CRT.) If applied to the above program, one should observe the following *after the execution of the indicated instruction bytes:*

Executed instruction bytes	Accumulator	Register B
INOl5	10110101	
ANI017	00000101	
RLC, RLC, RLC, RLC	01010000	
MOVB,A	01010000	01010000
INOl5	101101 I I	01010000
ANIOl7	000001 I I	01010000
ORAB	01010111	01010000

This completes our discussion of the more important logical instructions in the 8080A instruction set. Additional examples will be used in subsequent columns, where they will be incorporated into data-manipulation and decision-making tasks.

References

1. RONY, P.R., TITUS, J.A., and LARSEN, D.G., "Microcomputer interfacing: What is a logical instruction?," *Amer. Lab.* 9 (2), 158 (1977).
2. TITUS, LA., LARSEN, D.G., and RONY, P.R., "Microcomputer interfacing: The MOV and MVI 8080 instructions," *Amer. Lab.* 8 (I I), 114 (1976).
3. LARSEN, D.G., RONY, P.R., and TITUS, LA., "Microcomputer interfacing: Register pair instructions," *Amer. Lab.* 8 (12),75 (1976).
4. *Intel 8080 Microcomputer Systems User's Manual* (Intel Corp., Santa Clara, Calif. 1975).
5. FIELD, P.E., LARSEN, D.G., RONY, P.R., and TITUS. J.A., "Microcomputer interfacing: A software UART," *Amer. Lab.* 8 (7), 72 (1916).