KOMPUTER KORNER

A look at the arithmetic/logic instructions and processor flags of the 8080 microprocessor

TIM BARRY

WE PREVIOUSLY DISCUSSED THE DATA-TRANSFER instructions that can be performed by the 8080. This month we will continue on with this presentation by examining the arithmetic/logic instructions that the 8080 uses to operate upon the data located in its registers and the system memory. (You may wish to refer to the previous column for an explanation of some of the notations used to represent various register groups, data types, and data transfers.)

Arithmetic/logic instructions

Any computer instruction that modifies the contents of a register, memory location or flag by any arithmetic or logic operation is considered to be a member of this group. As with data-transfer instructions, we can consider arithmetic/logic instructions in the context of data sources and data destinations. The execution of an arithmetic/logic operation causes the contents of one or more data sources to be operated upon by the computer's arithmetic/logic unit. The result of the operation is then placed into the data destination. The contents of the data sources used in the operation are called *operands.* Most computer arithmetic/logic operations use one, two or occasionally three operands. The 8080 uses arithmetic/logic operations that require either one or two operands.

Processor flags

Intimately related to the concept of arithmetic/logic instructions is the concept of *processor flags,* or simply *flags.* A flag is an internal logic element (usually a flip-flop) that indicates the state of the processor. Some of these flags are set and reset based upon the result of the various arithmetic/logic operations. These flags can then be tested and their state used to determine whether or not some operation is to be executed. For example, the operation to take place after an addition operation has been performed may depend on whether or not the result of the operation was zero. If the processor has a flag that is set if the result of an operation is zero, it could be tested and the next step based on whether or not it was set or clear. Without flags to set and test, the ability of the computer to modify its operation based upon the results of operations performed would be lost, and with it the flexibility of programmed control.

The 8080 provides four flags that are affected by arithmetic/logic operations and they are tested under program control. These are the carry flag, the zero flag, the sign flag and the parity flag. The conditions that are indicated by these flags are described below:

Carry flag: The carry flag is set or reset under the following conditions:

1. Set if the result of an addition is greater than 256
2. Set if the result of a subtraction is less than 0
3. Cleared by all logic operations (AND, XRA, ORA)
4. Set or cleared by the rotate instructions

In addition, there are two 8080 instructions that can be used to directly modify the state of the carry flag. Each instruction is listed below (in bold) followed by the data-transfer notation and the meaning.

STC (1)
Operation performed: $1 \longrightarrow C$
The carry flag is set to 1.

CMC (1)
Operation performed: $\bar{C} \longrightarrow C$
The state of the carry flag is complemented.

Zero flag: The zero flag is set to a logic-one anytime an arithmetic or logic operation results in a zero in the accumulator.

Sign flag: The sign flag corresponds to the sign of the two's compliment number in the accumulator after an arithmetic or logic operation has been performed. If bit-7 of the accumulator is a zero (the number is positive), the sign flag will be set to a logic-0. If bit-7 of the accumulator is a logic-1 (the number is negative), the sign flag will be set to a logic-1.

Parity flag: The parity flag is set or reset based upon the number of logic-1's left in the accumulator as the result of an arithmetic or logic operation. If the number of bits set to logic-1 in the accumulator is even (even parity), then the parity flag will be set to logic-1. If the number of bits set to logic-1 in the accumulator is odd (odd parity), then the parity flag will be set to logic-0.

It is important to remember that the flags are only set or cleared as the result of arithmetic and logic operations. With one exception, data-transfer instructions do not alter the state of the flags. This means that to test a value from memory or an input device, you must perform an arithmetic or logic operation with that value to set the flags. The act of transferring data, by itself, does not set flags to correspond to the data transferred.

When using the 8080, the contents of the accumulator and all flags can be treated as a program status word (PSW). PSW can be saved in the stack by using the PUSH PSW instruction. This saves the status of the flags when the push instruction is executed. This status can then be restored later by using a POP PSW instruction. (POP PSW is the data transfer-instruction that violates the general statement that data transfers don't alter flags.)

continued on page 26

KOMPUTER KORNER
continued from page 24

Two-operand instructions

The 8080 provides nine basic two-operand arithmetic/logic operations. Eight of these are 8-bit operations and one is a 16-bit operation. For the 8-bit operations, the two data-sources are the A register and one of the following: A single register (A, B, C, D, E, H, or L), the contents of memory as addressed by the HL register pair (M), or an 8-bit immediate data value (D8). The data destination is always the A register. For the 16-bit operation, the two data-sources are the HL register pair and one of the 16-bit register pairs (BD, DE, HL) or the stack pointer (SP). The data destination is always the HL register pair.

ADD S $(A) + (S) \rightarrow A$ (1)
ADD M $(A) + ([HL]) \rightarrow A$ (1)
ADI D8 $(A) + D8 \rightarrow A$ (2)
The above program adds the contents of the specified data source to the contents of A and the result is placed in A. All flags are affected.

ADC S $(A) + (S) + C \rightarrow A$ (1)
ADC M $(A) + ([HL]) + C \rightarrow A$ (1)
ACI D8 $(A) + D8 + C \rightarrow A$ (2)
The above program adds together the contents of the specified data source, the contents of A and the carry flag, and the result is placed in A. All flags are affected.

SUB S $(A) - (S) \rightarrow A$ (1)
SUB M $(A) - ([HL]) \rightarrow A$ (1)
SUI D8 $(A) - D8 \rightarrow A$ (2)
Here, the contents of the selected data source are subtracted from the contents of A and the result is placed in A. All flags are affected by this operation.

SBB S $(A) - (S) - C \rightarrow A$ (1)
SBB M $(A) - ([HL]) - C \rightarrow A$ (1)
SBI D8 $(A) - D8 - C \rightarrow A$ (2)
The contents of the selected data source and the carry flag are subtracted from the contents of A and the result is placed in A. All flags are affected.

ANA S $(A) \wedge (S) \rightarrow A$ (1)
ANA M $(A) \wedge ([HL]) \rightarrow A$ (1)
ANI D8 $(A) \wedge D8 \rightarrow A$ (2)
The logic and operation is performed using the contents of A and the selected data source. The result is placed in A. All flags are affected.

XRA S $(A) \veebar (S) \rightarrow A$ (1)
XRA M $(A) \veebar ([HL]) \rightarrow A$ (1)
XRI D8 $(A) \veebar D8 \rightarrow A$ (2)
The exclusive-or operation is performed using the contents of A and the selected data source. The result is placed in A. All flags are affected.

ORA S $(A) \vee (S) \rightarrow A$ (1)
ORA M $(A) \vee ([HL]) \rightarrow A$ (1)
ORI D8 $(A) \vee D8 \rightarrow A$ (2)
The logic-or operation is performed using the contents of A and the selected data source. The result is placed in A. All flags are affected.

CMP S $(A)-(S)$ (1)
CMP M $(A)-([HL])$ (1)
CPI D8 $(A)-D8$ (2)
The contents of the selected data-source are compared to the contents of A. The comparison is made by subtracting the contents of the data source from A, with the flags set based on what the result would have been. *Neither A nor the data source is modified by the operation.* All flags are affected.

DAD RP $(HL) + (RP) \rightarrow HL$ (1)
The contents of the selected register pair are added to the contents of the HL register pair. The result is placed in HL. Only the carry flag is affected.

Single-operand instructions

Single operand arithmetic/logic instructions use the same data resource as both data source and data destination. These operations are used to modify the individual registers, register pairs, and memory locations in the 8080 system.

RLC $(A_7) \rightarrow C$, $(A_7) \rightarrow A_0$, $(A_n) \rightarrow A_{n+1}$ (1)
RRC $(A_0) \rightarrow C$, $(A_0) \rightarrow A_7$, $(A_n) \rightarrow A_{n-1}$ (1)
RAL $(A_7) \rightarrow C$, $C \rightarrow A_0$, $(A_n) \rightarrow A_{n+1}$ (1)
RAR $(A_0) \rightarrow C$, $C \rightarrow A_7$, $(A_n) \rightarrow A_{n-1}$ (1)

These four instructions treat the A register and carry flag as a serial rotation register. The graphical representation of these rotations is shown in Fig. 1. Note how RAL and
continued on page 28

KOMPUTER KORNER

continued from page 26

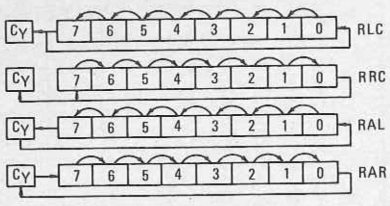

FIG. 1

RAR actually include the contents of the carry flag in the rotation. RLC and RRC modify the carry flag but do not include it in the rotation. No flags other than carry are affected by these instructions.

INR S (S) + 1 → S (1)
INR M ([HL]) + 1 → [HL] (1)
The contents of the selected data source are incremented by one. All flags *except* carry are affected.

DCR S (S) − 1 → S (1)
DCR M ([HL]) − 1 → [HL] (1)
The contents of the selected data source are decremented by one. All flags *except* carry are affected.

INX RP (RP) + 1 → RP (1)
The contents of the selected register pair are incremented by 1. No flags are affected.

DCX RP (RP) − 1 → RP (1)
The contents of the selected register pair are decremented by 1. No flags are affected.

DAA IF $(A_0 - A_3) > 9$, (1)
then (A) + 6 → A
If $(A_4 - A_7) > 9$,
then (A) + 60H → A
The DAA instruction is used to adjust the hexidecimal contents of A into two BCD digits. If the four least-significant bits of A (commonly called the least significant nibble, since a nibble is about half a byte) is a value greater than nine, six is added to A. This forces the least significant nibble into the range zero to nine. The most significant nibble is then tested. If it is now greater than nine, 60H is added to A to force the value into the range zero to nine. If the result of adjusting the value in A results in a value greater than 99 decimal, the carry flag is set. All flags are affected by the operation.

CMA $\bar{A}$ → A (1)
The contents of the accumulator are complemented. No flags are affected.

This column has presented a look at the arithmetic/logic instructions offered by the 8080. In this short space it is practically impossible to provide all the details of a complex computer instruction set. We have attempted instead to give a good overview into the operation of these instructions. In future articles we will use these instructions as the building blocks of more complex operations. If we need a more detailed analysis of the operation of a particular instruction, it will be presented at that time.

This article was the second in our series on the 8080 instruction set. In a future column we will conclude the presentation by examining the transfer of control and processor control instructions. **R-E**

Optoelectronics now used in auto ignition systems

As a new type of "breaker points" for the transistor auto ignition system, Siemens is experimenting with what they call a "light barrier", or light switch, as being simpler and longer lasting than systems now in use. The first electronic ignition systems simply used the breaker points already on the car, on the basis that the points might last indefinitely with the negligible current drawn by a transistor circuit. Later, magnetic pick-up systems that require no physical contact of the "breaker points" have been used, and the Siemens report also speaks of magnetoresistor sensors and Hall generators being used to trigger the transistor ignition.

In the Siemens device, an LED transmits the trigger signal for the ignition and a phototransistor picks it up. Between the light-emitting diode and the phototransistor is a slotted disc that rotates on the distributor shaft, allowing a narrow beam of light to pass at the firing point. The duration, or dwell, depends on the slot width.

Before full-scale production can be started, there are temperature and economic problems to be ironed out. Metal cases would be satisfactory at the temperatures to be withstood, but cost considerations indicate the use of plastic. **R-E**