Chapter 6
Introduction to 8085 Instructions

8085 Instruction Set

The 8085 instructions can be classified as follows:

- Data transfer operations
  - Between registers
  - Between memory location and a register
  - Direct write to a register / memory
  - Between I/O device and accumulator
- Arithmetic operations (ADD, SUB, INR, DCR)
- Logic operations
- Branching operations (JMP, CALL, RET)
- Machine Control Operations

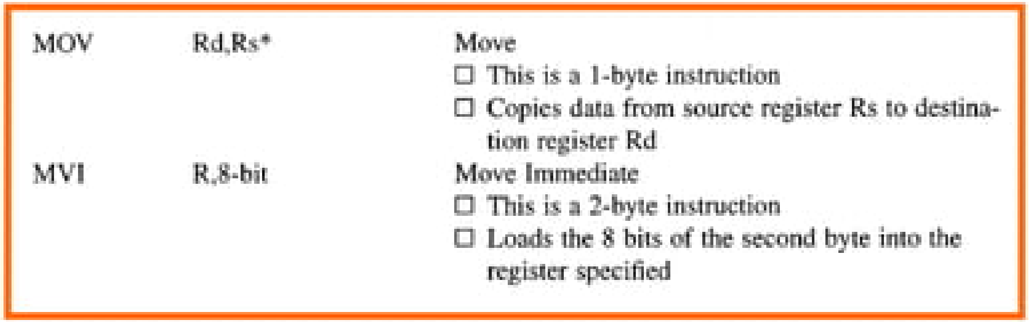
Data Transfer Operations

- These operations simply **COPY** the data from the source to the destination.
- MOV, MVI, IN, OUT

- They transfer:
  - Data between registers.
  - Data Byte to a register or memory location.
  - Data between a memory location and a register.
  - Data between an I\O Device and the accumulator.

- The data in the source is not changed.

Simple Data Transfer Operations

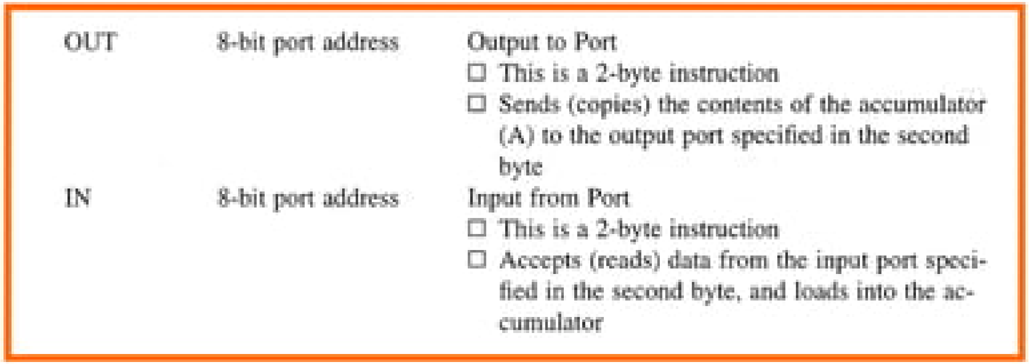


MOV	Rd,Rs*	Move □ This is a 1-byte instruction □ Copies data from source register Rs to destination register Rd
MVI	R,8-bit	Move Immediate □ This is a 2-byte instruction □ Loads the 8 bits of the second byte into the register specified

Examples:

▪	MOV	B,A	47	From ACC to REG
▪	MOV	C,D	4A	Between two REGs
▪	MVI	D,47	16	Direct-write into REG D
			47	

Simple Data Transfer Operations



OUT	8-bit port address	Output to Port □ This is a 2-byte instruction □ Sends (copies) the contents of the accumulator (A) to the output port specified in the second byte
IN	8-bit port address	Input from Port □ This is a 2-byte instruction □ Accepts (reads) data from the input port specified in the second byte, and loads into the accumulator

Example:

- OUT 05 D3
  05

Contents of ACC sent to output port number 05.

Machine Control Operations

- HLT: Halt
  - This is 1 byte instruction
  - The processor stops executing and enters wait state
- NOP: No Operation
  - This is a 1-byte instruction
  - No operation is performed

REVIEW OF IMPORTANT CONCEPTS

- REGISTERS ARE USED TO LOAD DATA DIRECTLY OR TO SAVE DATA BYTES.
- IN DATA TRANSFER (COPYING) THE DESTINATION REGISTER IS MODIFIED BUT THE SOURCE REGISTER RETAINS ITS DATA.
- THE 8085 TRANSFERS DATA FROM AN INPUT PORT TO THE ACCUMULATOR (IN) AND FROM THE ACCUMULATOR TO AN OUTPUT PORT (OUT). THE INSTRUCTION OUT CANNOT SEND DATA FROM ANY OTHER REGISTER.
- IN THE 8085 PROCESSOR, DATA TRANSFER INSTRUCTION DO NOT AFFECT THE FLAGS.

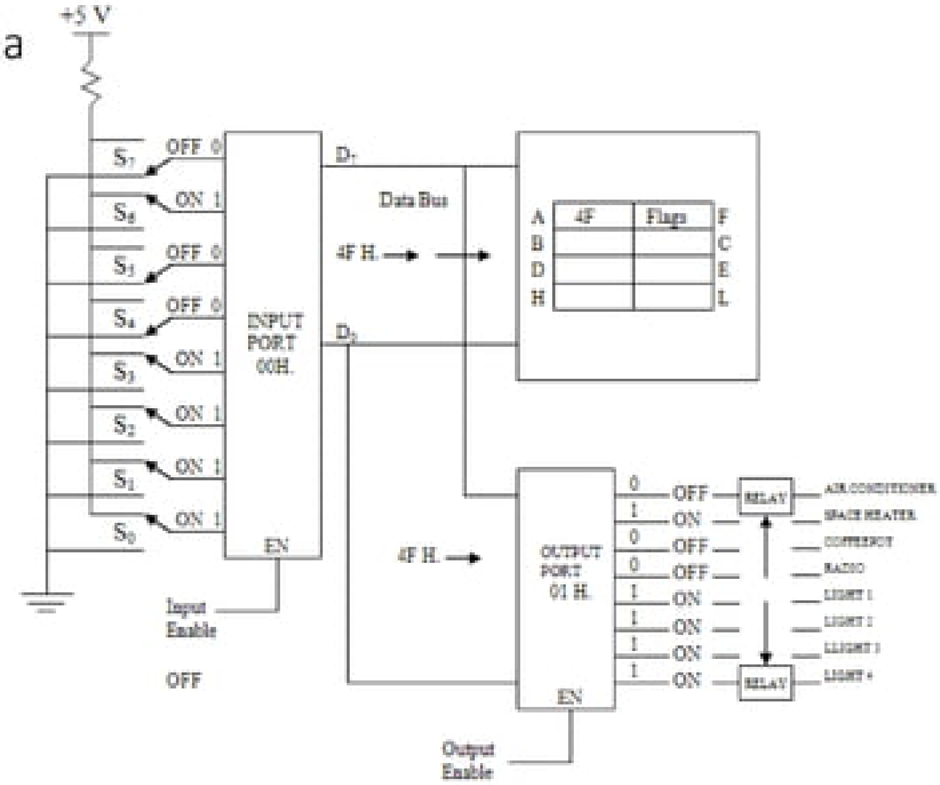
Example 6.1

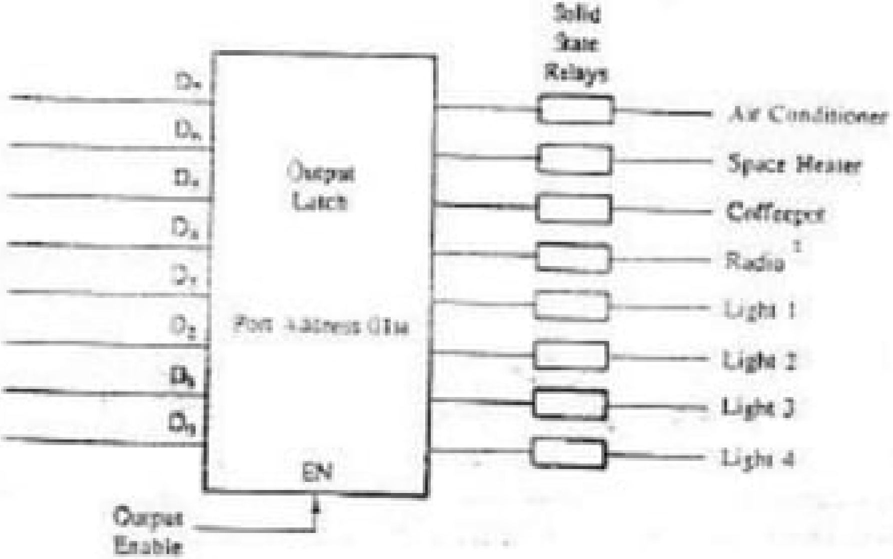
- Load the accumulator A with the data byte 82h. And save the data in register B.

Example 6.1

- Load the accumulator A with the data byte 82h. And save the data in register B.
    - MVI A, 82H.
    - MOV B, A
- The first instruction is a 2-byte instruction that loads the accumulator with the data byte 82h., and the second instruction copies the contents of the accumulator in register B without changing the contents of the accumulator.

Reading Da rt




Solid
State
Relays
D_7
D_6
D_5
D_4
D_3
D_2
D_1
D_0
Output
Latch
Port Address 01H
EN
Output
Enable
Air Conditioner
Space Heater
Coffeepot
Radio
Light 1
Light 2
Light 3
Light 4

Example 6.2

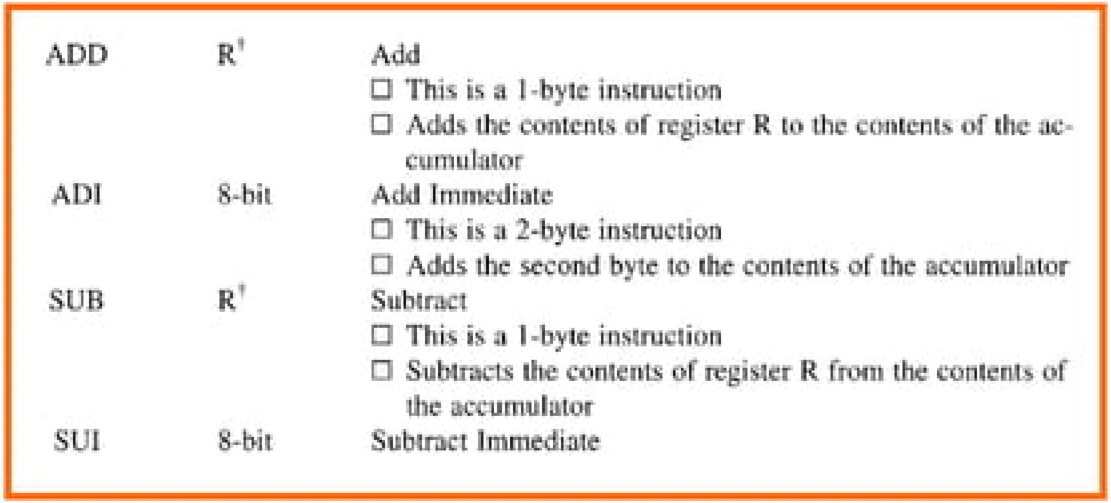
- Write instructions to read eight ON/OFF switches connected to the input with the address 00H. And turn on the devices connected to the output port with the address 01H. As shown in figure.
  - IN 00H.
  - OUT 01H.
  - HLT

Arithmetic Operations

- Addition (ADD, ADI):
    - Any 8-bit number.
    - The contents of a register.
    - The contents of a memory location.
  - Can be added to the contents of the accumulator and the result is stored in the accumulator.

- Subtraction (SUB, SUI):
    - Any 8-bit number
    - The contents of a register
    - The contents of a memory location
  - Can be subtracted **from** the contents of the accumulator. The result is stored in the accumulator.

Arithmetic Operations



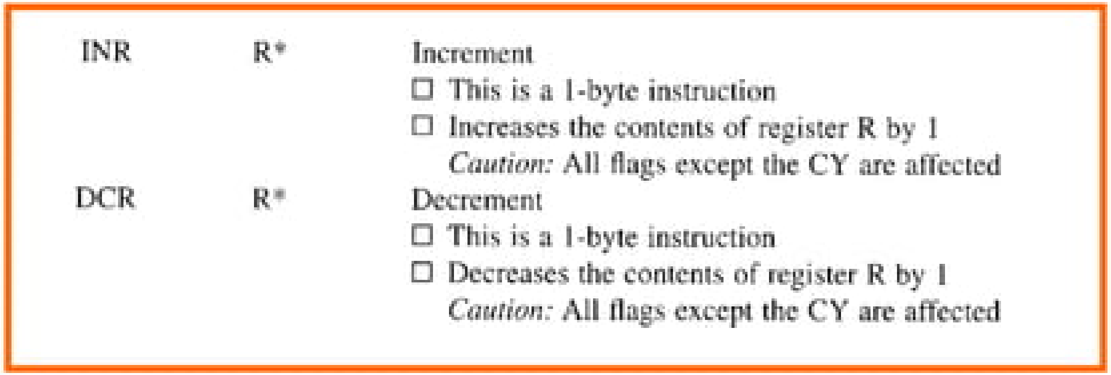
ADD	R[1]	Add □ This is a 1-byte instruction □ Adds the contents of register R to the contents of the accumulator
ADI	8-bit	Add Immediate □ This is a 2-byte instruction □ Adds the second byte to the contents of the accumulator
SUB	R[1]	Subtract □ This is a 1-byte instruction □ Subtracts the contents of register R from the contents of the accumulator
SUI	8-bit	Subtract Immediate

Arithmetic Operations

- Increment (INR) and Decrement (DCR):
  - The 8-bit contents of any memory location or any register can be directly incremented or decremented by 1.
  - No need to disturb the contents of the accumulator.
- Affect all flags except the CY flag.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		CY

Arithmetic Operations



INR	R*	Increment □ This is a 1-byte instruction □ Increases the contents of register R by 1 *Caution:* All flags except the CY are affected
DCR	R*	Decrement □ This is a 1-byte instruction □ Decreases the contents of register R by 1 *Caution:* All flags except the CY are affected

Example 6.3

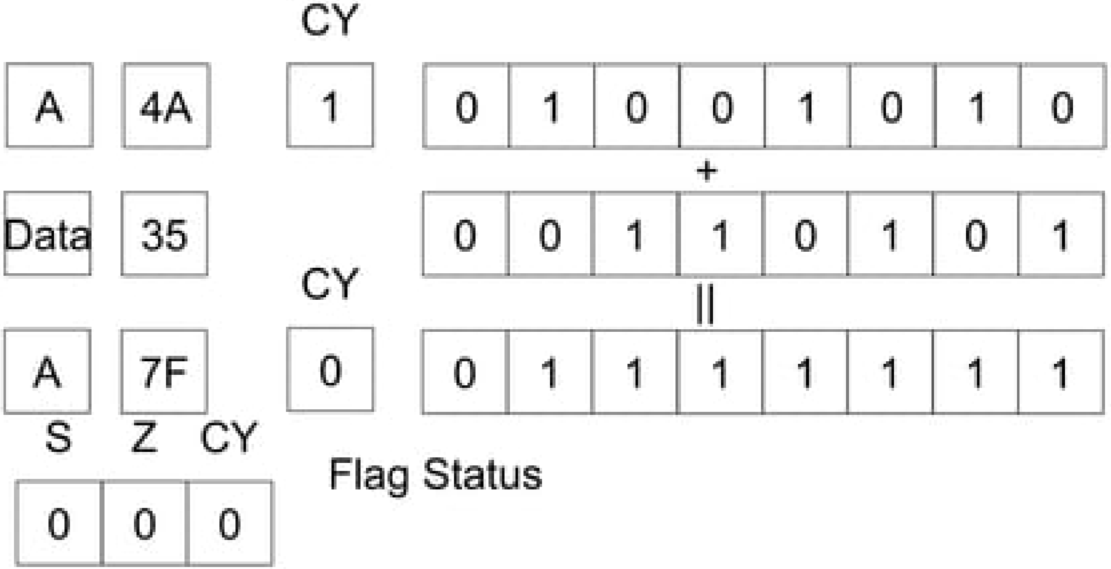
Instruction ADD C

			CY	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
(A)	:	93H =		1	0	0	1	0	0	1	1	
	+											
(C)	:	B7H =		1	0	1	1	0	1	1	1	
			1		1	1		1	1	1		Carry
SUM (A)	:	$\boxed{1}$ 4AH = CY	$\boxed{1}$	0	1	0	0	1	0	1	0	

Flag Status:[†] S = 0, Z = 0, CY = 1

Example 6.4

- Add the number 35H. Directly to the sum in the previous example when the CY flag is set.
    - ADI 35H.

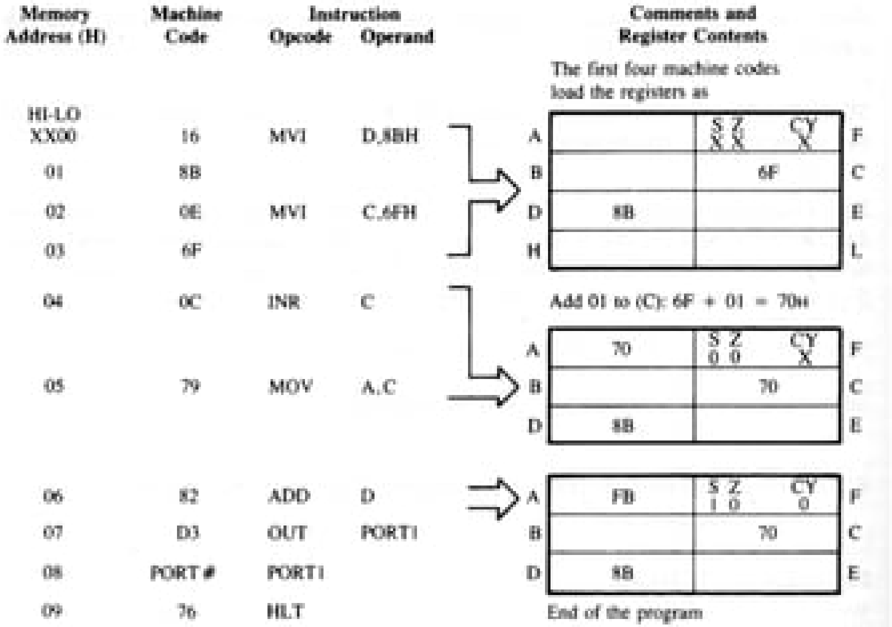


		CY								
A	4A	1	0	1	0	0	1	0	1	0
						+				
Data	35		0	0	1	1	0	1	0	1
		CY				||				
A	7F	0	0	1	1	1	1	1	1	1

S	Z	CY
0	0	0

Flag Status

Arithmetic Operations



Memory Address (H)	Machine Code	Instruction Opcode	Instruction Operand
HI-LO XX00	16	MVI	D,8BH
01	8B		
02	0E	MVI	C,6FH
03	6F		
04	0C	INR	C
05	79	MOV	A,C
06	82	ADD	D
07	D3	OUT	PORT1
08	PORT #	PORT1	
09	76	HLT	

Comments and Register Contents

The first four machine codes load the registers as

A		S Z CY X X X	F
B		6F	C
D	8B		E
H			L

Add 01 to (C): 6F + 01 = 70H

A	70	S Z CY 0 0 X	F
B		70	C
D	8B		E

A	FB	S Z CY 1 0 0	F
B		70	C
D	8B		E

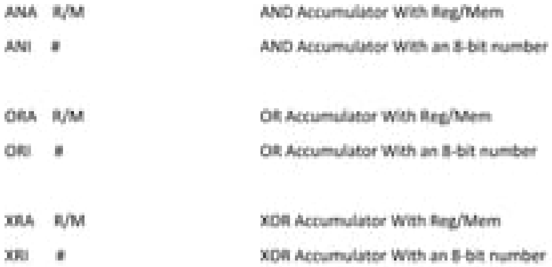
End of the program

REVIEW OF IMPORTANT CONCEPTS

- The arithmetic operations implicitly assume that the contents of the accumulator are one of the operands.
- The result of the arithmetic operations are stored in the accumulator; thus, the previous contents of the content of the accumulator are altered.
- The flags are modified to reflect the data conditions of an operation.
- In the Add operation, if the sum is larger than 8-bit CY is set.
- The Substract operation is performed by using the 2's complement method.
- If a substraction results in a negative number, the answersis in 2's complement and CY (the Borrow flag)
- In unsigned arithmetic operations, the Sign flag (S) should be ignored.
- The instructions INR and DCR are special cases of the arithmetic operations. These instructions can be used for any of the registers, and they do not affect CY, even if the result is larger than 8-bit. All other flags are affected by the result in the register used (not by the contents of the accumulator).

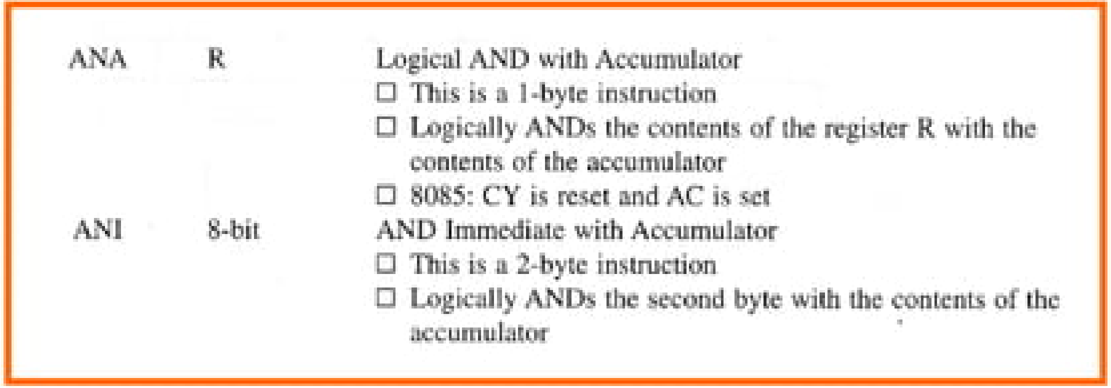
Logic Operations

- These instructions perform logic operations on the contents of the accumulator.
  - ANA, ANI, ORA, ORI, XRA and XRI
    - Source: Accumulator and
      - An 8-bit number
      - The contents of a register
      - The contents of a memory location
    - Destination: Accumulator



ANA	R/M	AND Accumulator With Reg/Mem
ANI	#	AND Accumulator With an 8-bit number
ORA	R/M	OR Accumulator With Reg/Mem
ORI	#	OR Accumulator With an 8-bit number
XRA	R/M	XOR Accumulator With Reg/Mem
XRI	#	XOR Accumulator With an 8-bit number

Logic Operations



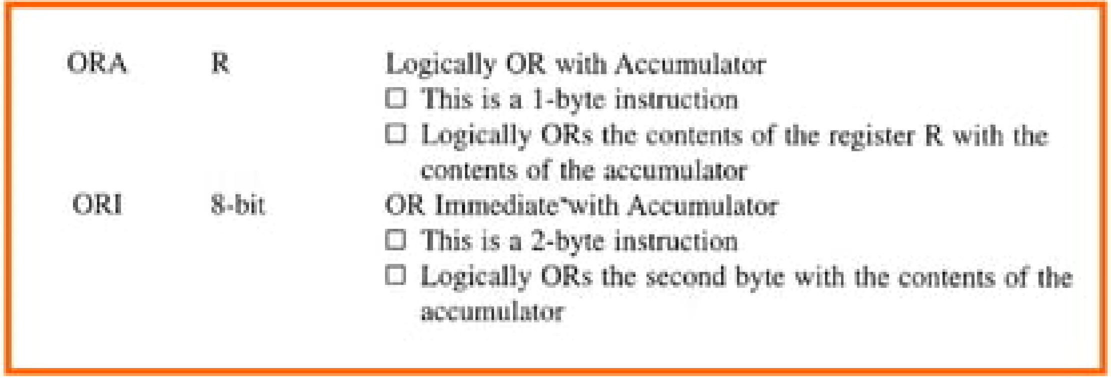
ANA R Logical AND with Accumulator

- □ This is a 1-byte instruction
- □ Logically ANDs the contents of the register R with the contents of the accumulator
- □ 8085: CY is reset and AC is set

ANI 8-bit AND Immediate with Accumulator

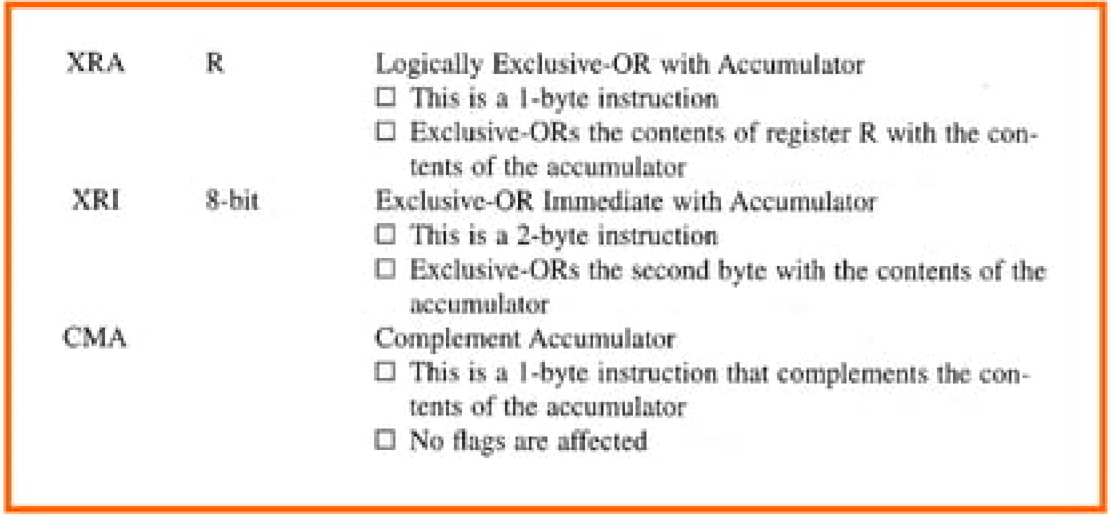
- □ This is a 2-byte instruction
- □ Logically ANDs the second byte with the contents of the accumulator

Logic Operations



ORA	R	Logically OR with Accumulator □ This is a 1-byte instruction □ Logically ORs the contents of the register R with the contents of the accumulator
ORI	8-bit	OR Immediate with Accumulator □ This is a 2-byte instruction □ Logically ORs the second byte with the contents of the accumulator

Logic Operations



XRA R — Logically Exclusive-OR with Accumulator

- □ This is a 1-byte instruction
- □ Exclusive-ORs the contents of register R with the contents of the accumulator

XRI 8-bit — Exclusive-OR Immediate with Accumulator

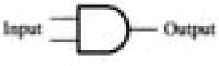
- □ This is a 2-byte instruction
- □ Exclusive-ORs the second byte with the contents of the accumulator

CMA — Complement Accumulator

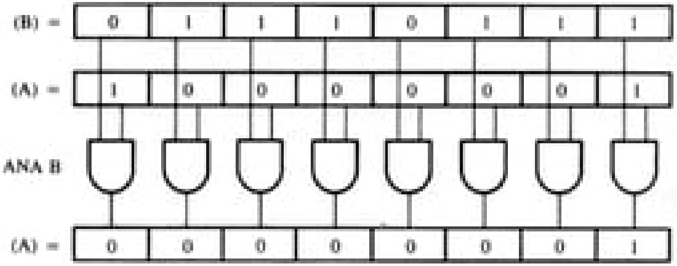
- □ This is a 1-byte instruction that complements the contents of the accumulator
- □ No flags are affected

Overview of Logic Operations

ANA:	AND	Logically AND the contents of a register.
ANI :	AND Immediate	Logically AND 8-bit data.
ORA:	OR	Logically OR the contents of a register.
ORI :	OR Immediate	Logically OR 8-bit data.
XRA:	X-OR	Exclusive-OR the contents of a register.
XRI :	X-OR Immediate	Exclusive-OR 8-bit data.



(a)



(b)

Data Masking With Logic AND

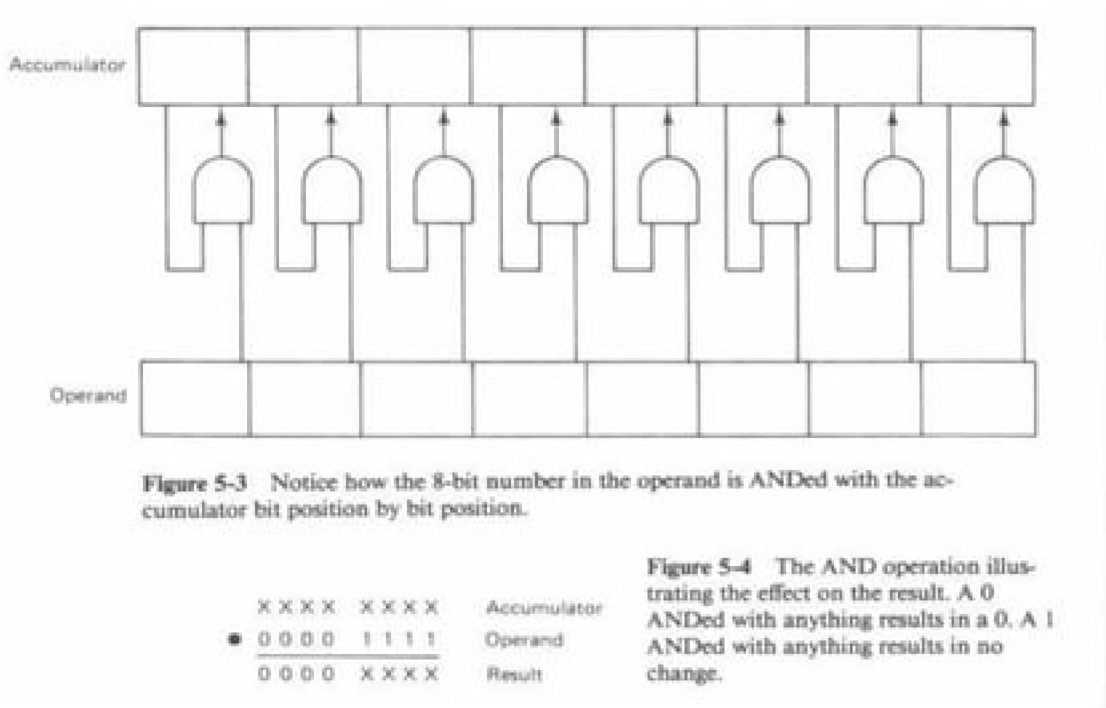

Accumulator
Operand

Figure 5-3 Notice how the 8-bit number in the operand is ANDed with the accumulator bit position by bit position.

	X X X X	X X X X	Accumulator
●	0 0 0 0	1 1 1 1	Operand
	0 0 0 0	X X X X	Result

Figure 5-4 The AND operation illustrating the effect on the result. A 0 ANDed with anything results in a 0. A 1 ANDed with anything results in no change.

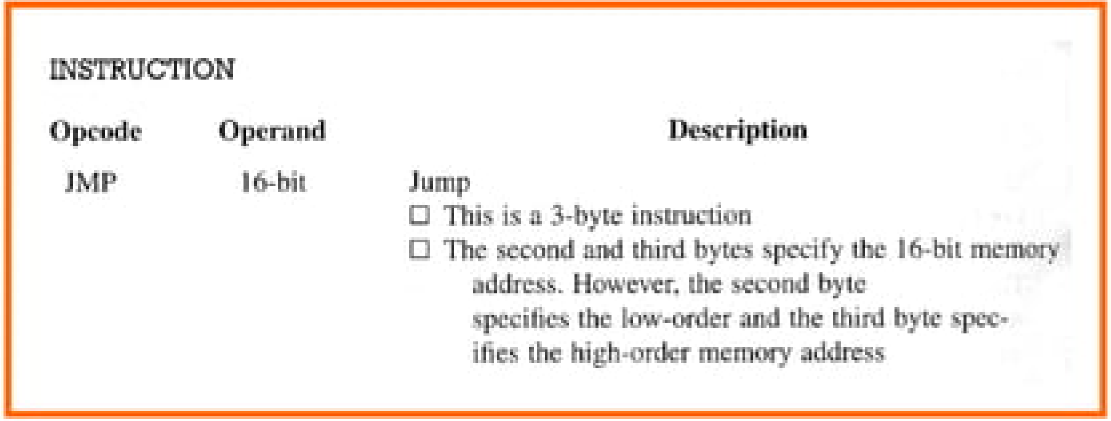
Branch Operations

- Three Types
  - Jump instructions
  - Call and Return Instructions
  - Restart Instructions

JUMP Instructions

- Two types:
  - Unconditional jump.
    - Go to a new location no matter what.
      - JMP Address
        - Jump to the address specified (Go to).
  - Conditional jump.
    - Go to a new location if the condition is true.
  - The addresses supplied to all branch operations must be 16-bits.

Branching Operations

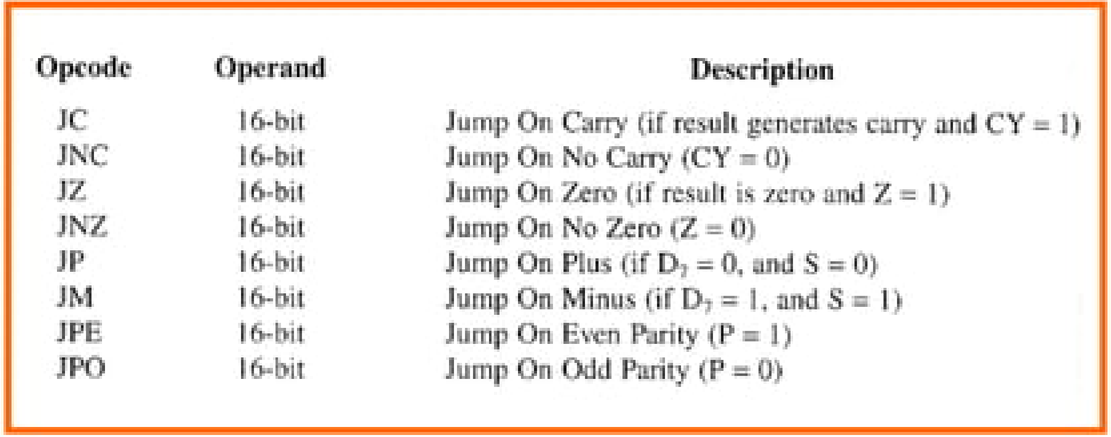


INSTRUCTION

Opcode	Operand	Description
JMP	16-bit	Jump □ This is a 3-byte instruction □ The second and third bytes specify the 16-bit memory address. However, the second byte specifies the low-order and the third byte specifies the high-order memory address

Note: This is an **unconditional** jump operation.
It will **always** force the program counter to a fixed memory address ⟶ continuous loop !

Branching Operations



Opcode	Operand	Description
JC	16-bit	Jump On Carry (if result generates carry and CY = 1)
JNC	16-bit	Jump On No Carry (CY = 0)
JZ	16-bit	Jump On Zero (if result is zero and Z = 1)
JNZ	16-bit	Jump On No Zero (Z = 0)
JP	16-bit	Jump On Plus (if $D_7 = 0$, and S = 0)
JM	16-bit	Jump On Minus (if $D_7 = 1$, and S = 1)
JPE	16-bit	Jump On Even Parity (P = 1)
JPO	16-bit	Jump On Odd Parity (P = 0)

Conditional jump operations are very useful for decision making during the execution of the program.

Conditional Jump

- Go to new location if a specified condition is met.
  - JZ Address (Jump on Zero)
    - Go to address specified if the **Zero flag is set**.
  - JNZ Address (Jump on NOT Zero)
    - Go to address specified if the **Zero flag is not set**.
  - JC Address (Jump on Carry)
    - Go to the address specified if the **Carry flag is set**.
  - JNC Address (Jump on No Carry)
    - Go to the address specified if the **Carry flag is not set**.
  - JP Address (Jump on Plus)
    - Go to the address specified if the **Sign flag is not set**
  - JM Address (Jump on Minus)
    - Go to the address specified if the **Sign flag is set**.

Example

Write a 8085 machine code program:

- Read two different memory locations (2050H. & 2051H.)
- Add the contents
- Send the result to output port 02 (display) if there is no overflow
- Display “FF” if there is an overflow
- Stop
- **Assume that the 8085 program begin at 2000H.**

Example

```
2000    LDA   2050     3A   Load contents of memory
2001                   50   location 2050 into accumulator
2002                   20
2003    MOV   B,A      47   Save the first number in B
2004    LDA   2051     3A   Load contents of memory
2005                   51   location 2051 into accumulator
2006                   20
2007    ADD   B        80   Add accumulator with B
2008    JNC   XXYY     D2   Jump to YYXX if no carry !
2009                   YY
2010                   XX
2011    MVI   A,FF     3E   Direct write FF into
2012                   FF   accumulator
2013    OUT   02       D3   Display accumulator contents
2014                   02   at output port 02
2015    HLT            76   Stop
```

Updated Code

```
2000      LDA   2050        3A   Load contents of memory
2001                        50   location 2050 into accumulator
2002                        20
2003      MOV   B,A         47   Save the first number in B
2004      LDA   2051        3A   Load contents of memory
2005                        51   location 2051 into accumulator
2006                        20
2007      ADD   B           80   Add accumulator with B
2008      JNC   2013        D2   Jump to 2013 if no carry !
2009                        13
2010                        20
2011      MVI   A,FF        3E   Direct write FF into
2012                        FF   accumulator
2013      OUT   02          D3   Display accumulator contents
2014                        02   at output port 02
2015      HLT               76   Stop
```

Unconditional Branch

- CALL Address
  - Jump to the address specified but treat it as a subroutine.

- RET
  - Return from a subroutine.

- The addresses supplied to all branch operations must be 16-bits.

Conditional Call

- Go to new location if a specified condition is met.
  - CZ Address (Jump on Zero)
    - Call subroutine if the **Zero flag is set**.
  - CNZ Address (Jump on NOT Zero)
    - Call subroutine if the **Zero flag is not set**.
  - CC Address (Jump on Carry)
    - Call subroutine if the **Carry flag is set.**
  - CNC Address (Jump on No Carry)
    - Call subroutine if the **Carry flag is not set**.
  - CP Address (Jump on Plus)
    - Call subroutine if the **Sign flag is not set**
  - CM Address (Jump on Minus)
    - Call subroutine if the **Sign flag is set**.

Conditional Return

- Go to new location if a specified condition is met.
  - RZ Address (Jump on Zero)
    - Return if the **Zero flag is set.**
  - RNZ Address (Jump on NOT Zero)
    - Return if the **Zero flag is not set.**
  - RC Address (Jump on Carry)
    - Return if the **Carry flag is set.**
  - RNC Address (Jump on No Carry)
    - Return if the **Carry flag is not set.**
  - RP Address (Jump on Plus)
    - Return if the **Sign flag is not set**
  - RM Address (Jump on Minus)
    - Return if the **Sign flag is set.**

Machine Control

- HLT
  - Stop executing the program.
- NOP
  - No operation
  - Exactly as it says, do nothing.
  - Usually used for delay or to replace instructions during debugging.

- Example program using 8085 microprocessor coding

The Algorithm of the program

```
1:    total = 0, i = 0
2:    i = i + 1
3:    total = total + i
4:    IF i ≠ n THEN GOTO 2
```

$$\left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\} n + (n - 1) + \ldots + 1$$

The 8085 coding of the program

```
        LDA  n          i = n
        MOV  B, A
        XRA  A          sum = A ⊕ A = 0
Loop:   ADD  B          sum = sum + i
        DCR  B          i = i - 1
        JNZ  Loop       IF i ≠ 0 THEN GOTO Loop
        STA  total      total = sum
```