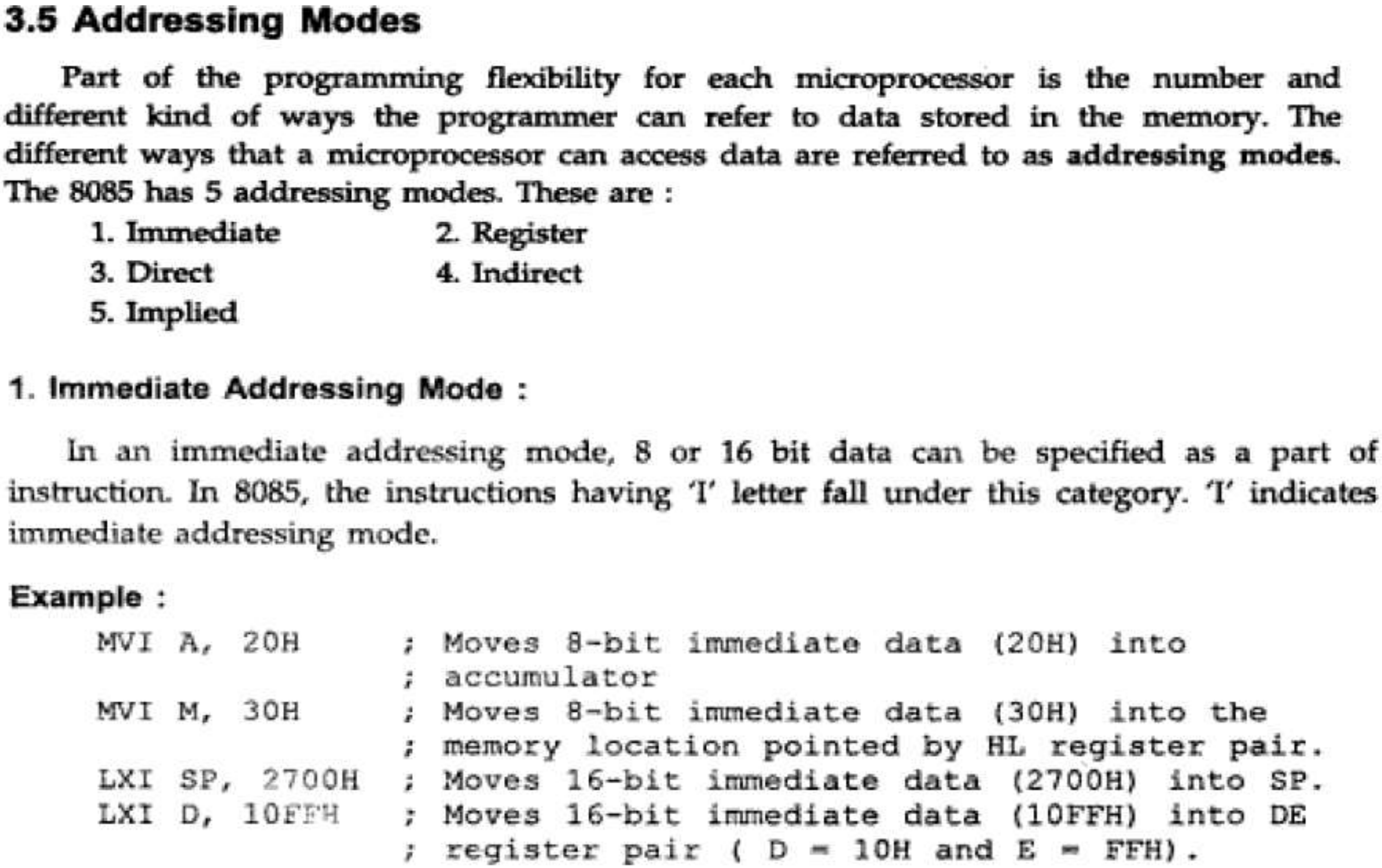


3.5 Addressing Modes

Part of the programming flexibility for each microprocessor is the number and different kind of ways the programmer can refer to data stored in the memory. The different ways that a microprocessor can access data are referred to as **addressing modes.** The 8085 has 5 addressing modes. These are :

1. Immediate	2. Register
3. Direct	4. Indirect
5. Implied	

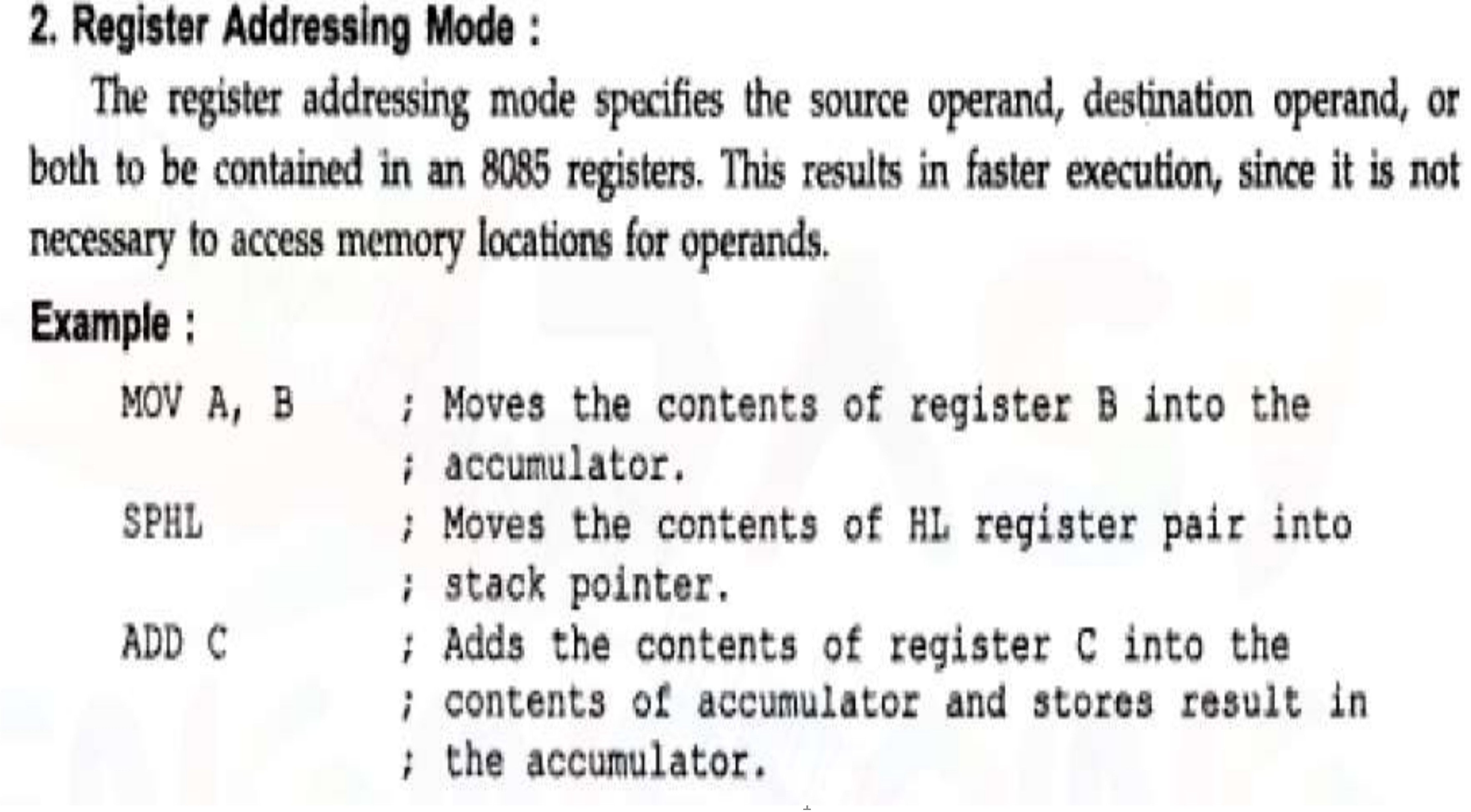
1. Immediate Addressing Mode :

In an immediate addressing mode, 8 or 16 bit data can be specified as a part of instruction. In 8085, the instructions having ‘I’ letter fall under this category. ‘I’ indicates immediate addressing mode.

Example :

```
MVI A, 20H      ; Moves 8-bit immediate data (20H) into
                ; accumulator
MVI M, 30H      ; Moves 8-bit immediate data (30H) into the
                ; memory location pointed by HL register pair.
LXI SP, 2700H   ; Moves 16-bit immediate data (2700H) into SP.
LXI D, 10FFH    ; Moves 16-bit immediate data (10FFH) into DE
                ; register pair ( D = 10H and E = FFH).
```

Microprocessor……..



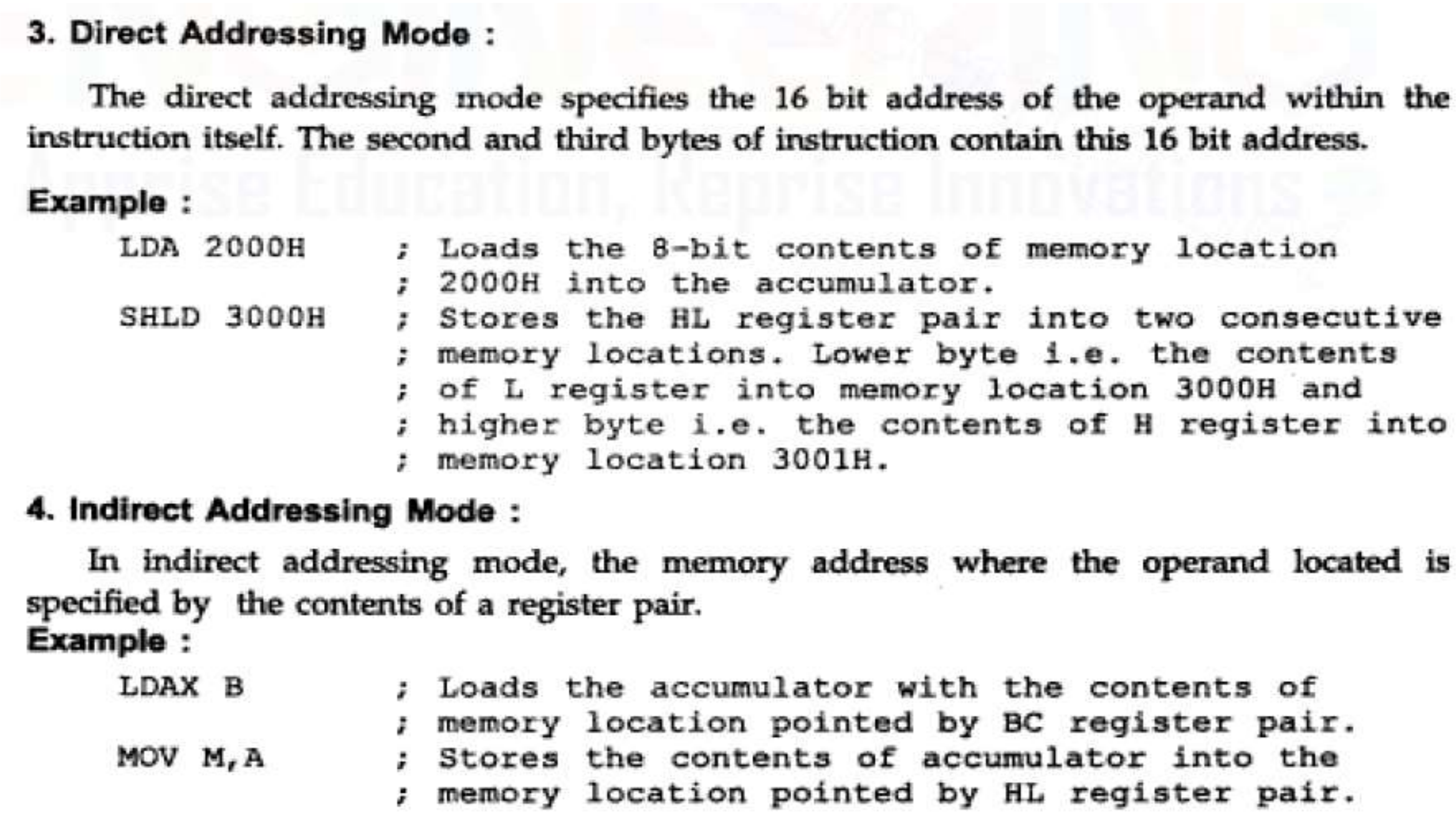
2. Register Addressing Mode :

The register addressing mode specifies the source operand, destination operand, or both to be contained in an 8085 registers. This results in faster execution, since it is not necessary to access memory locations for operands.

Example :

```
MOV A, B        ; Moves the contents of register B into the
                ; accumulator.
SPHL            ; Moves the contents of HL register pair into
                ; stack pointer.
ADD C           ; Adds the contents of register C into the
                ; contents of accumulator and stores result in
                ; the accumulator.
```

Microprocessor……..



3. Direct Addressing Mode :

The direct addressing mode specifies the 16 bit address of the operand within the instruction itself. The second and third bytes of instruction contain this 16 bit address.

Example :

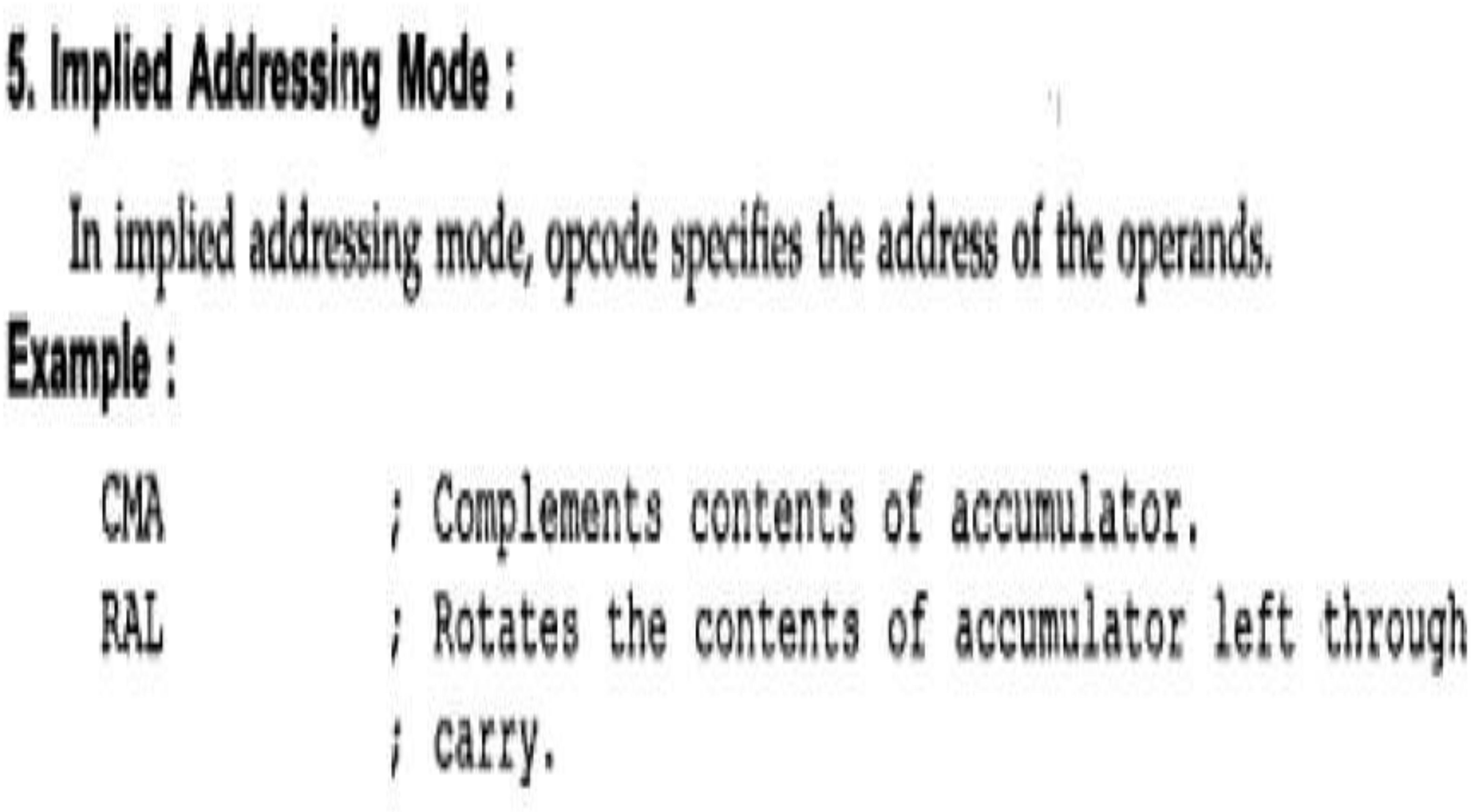
```
LDA 2000H      ; Loads the 8-bit contents of memory location
               ; 2000H into the accumulator.
SHLD 3000H     ; Stores the HL register pair into two consecutive
               ; memory locations. Lower byte i.e. the contents
               ; of L register into memory location 3000H and
               ; higher byte i.e. the contents of H register into
               ; memory location 3001H.
```

4. Indirect Addressing Mode :

In indirect addressing mode, the memory address where the operand located is specified by the contents of a register pair.

Example :

```
LDAX B         ; Loads the accumulator with the contents of
               ; memory location pointed by BC register pair.
MOV M,A        ; Stores the contents of accumulator into the
               ; memory location pointed by HL register pair.
```



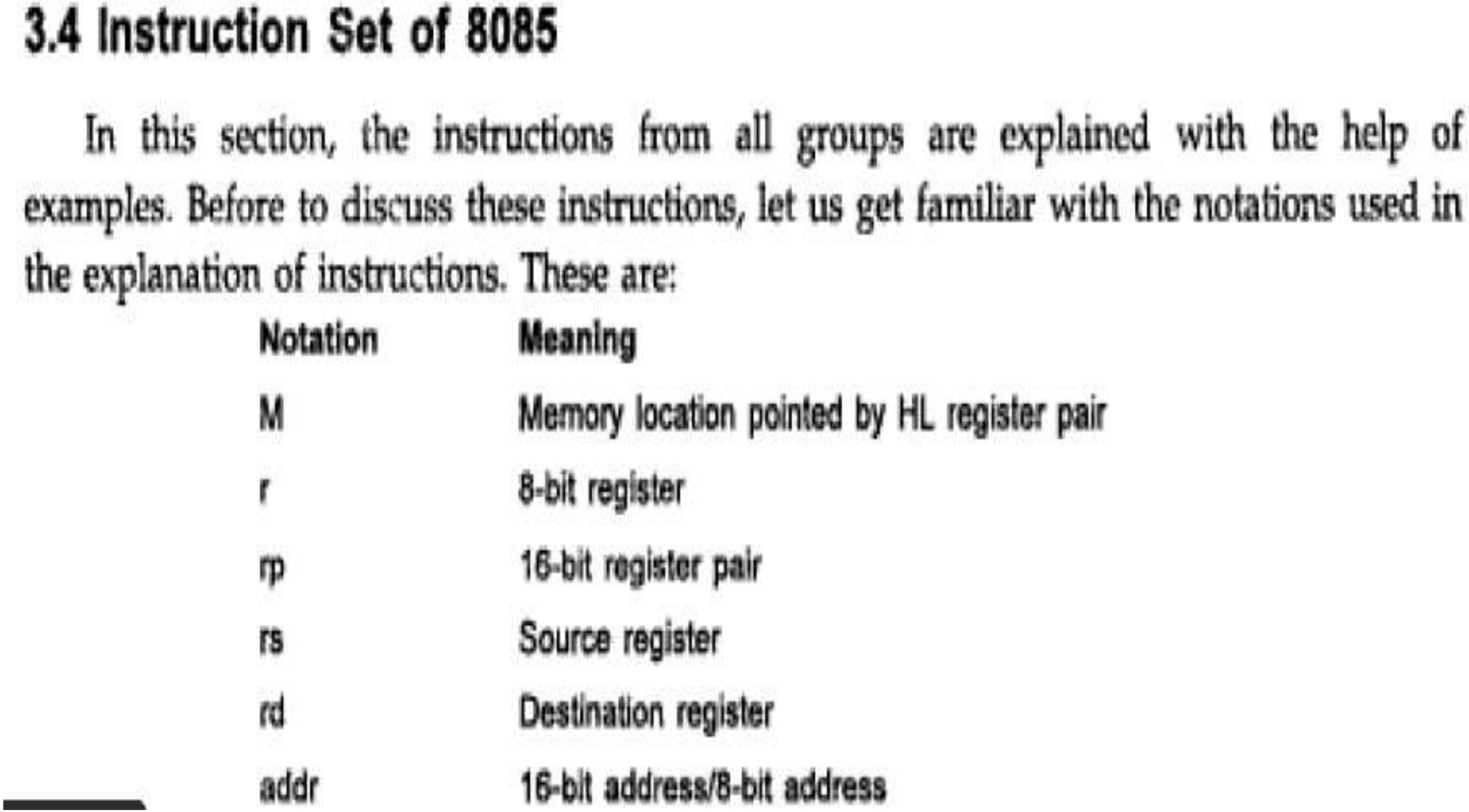
5. Implied Addressing Mode :

In implied addressing mode, opcode specifies the address of the operands.

Example :

```
CMA             ; Complements contents of accumulator.
RAL             ; Rotates the contents of accumulator left through
                ; carry.
```

Microprocessor……..

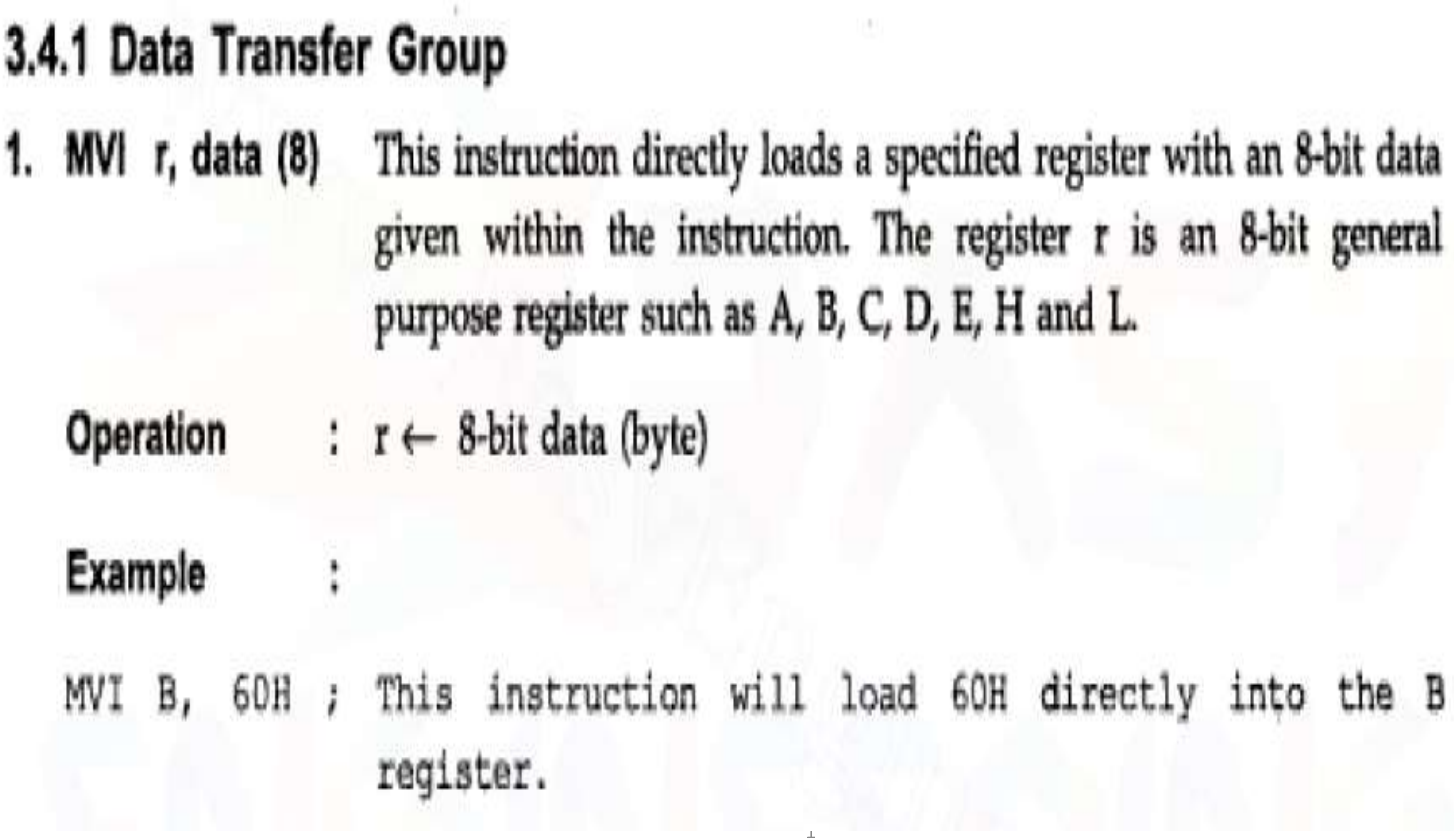


3.4 Instruction Set of 8085

In this section, the instructions from all groups are explained with the help of examples. Before to discuss these instructions, let us get familiar with the notations used in the explanation of instructions. These are:

Notation	Meaning
M	Memory location pointed by HL register pair
r	8-bit register
rp	16-bit register pair
rs	Source register
rd	Destination register
addr	16-bit address/8-bit address

Microprocessor……..



3.4.1 Data Transfer Group

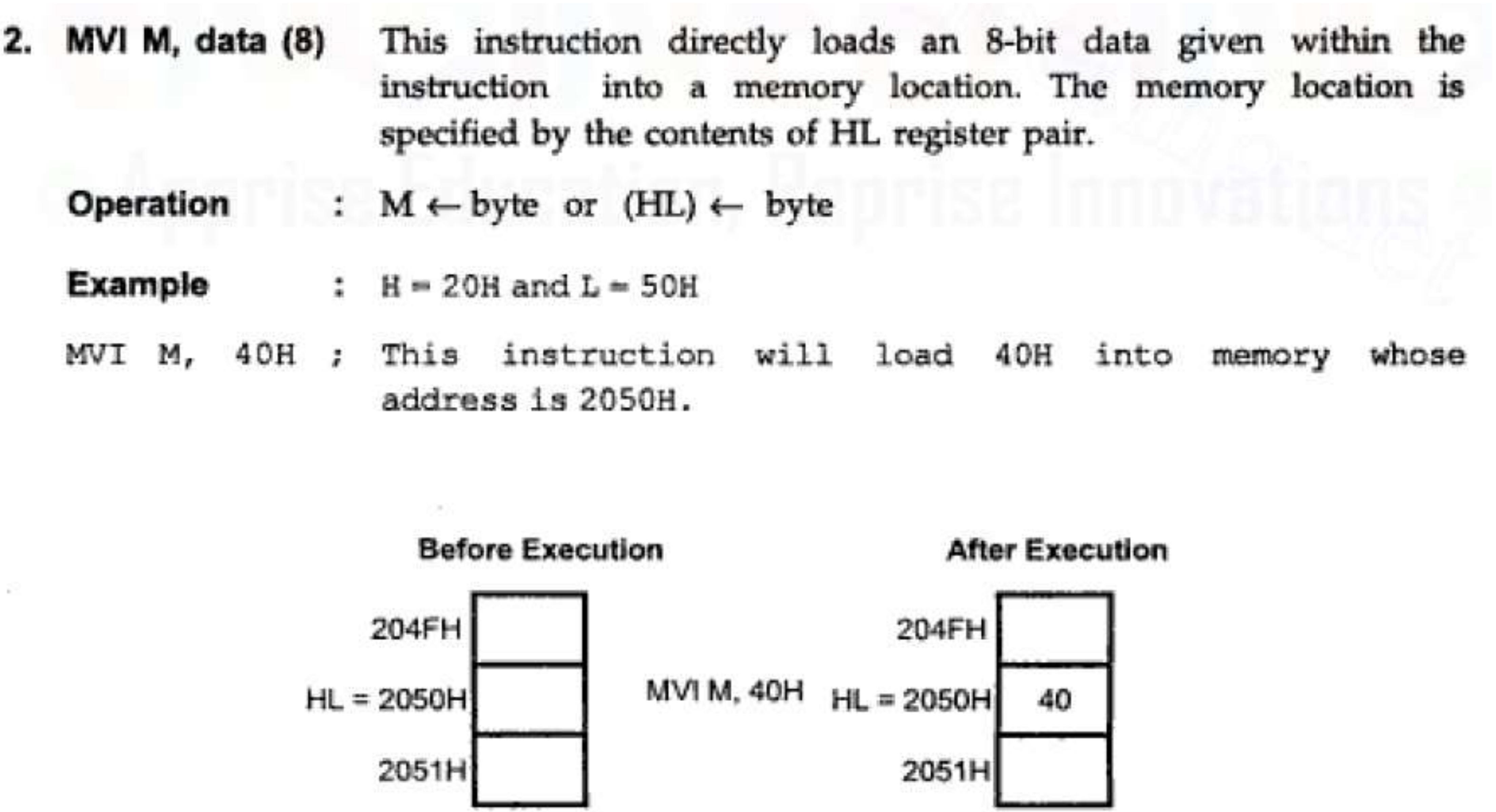
1. **MVI r, data (8)** This instruction directly loads a specified register with an 8-bit data given within the instruction. The register r is an 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : r ← 8-bit data (byte)

Example :

```
MVI B, 60H ; This instruction will load 60H directly into the B
             register.
```

Microprocessor……..



2. MVI M, data (8) This instruction directly loads an 8-bit data given within the instruction into a memory location. The memory location is specified by the contents of HL register pair.

Operation : M ← byte or (HL) ← byte

Example : H = 20H and L = 50H

MVI M, 40H ; This instruction will load 40H into memory whose address is 2050H.

Before Execution

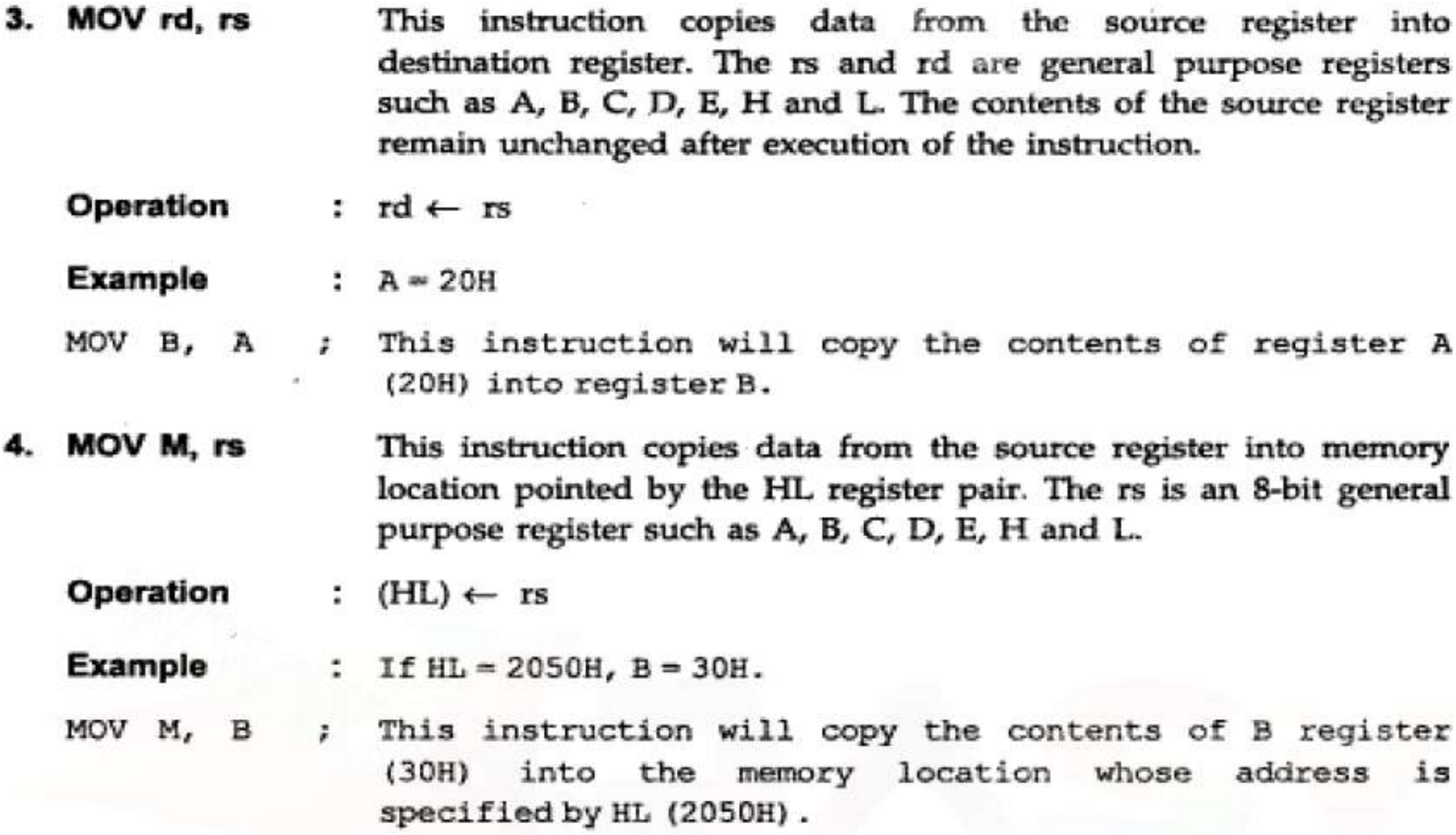
204FH	
HL = 2050H	
2051H	

MVI M, 40H

After Execution

204FH	
HL = 2050H	40
2051H	

Microprocessor……..



3. MOV rd, rs — This instruction copies data from the source register into destination register. The rs and rd are general purpose registers such as A, B, C, D, E, H and L. The contents of the source register remain unchanged after execution of the instruction.

Operation : rd ← rs

Example : `A = 20H`

`MOV B, A ;` This instruction will copy the contents of register A (20H) into register B.

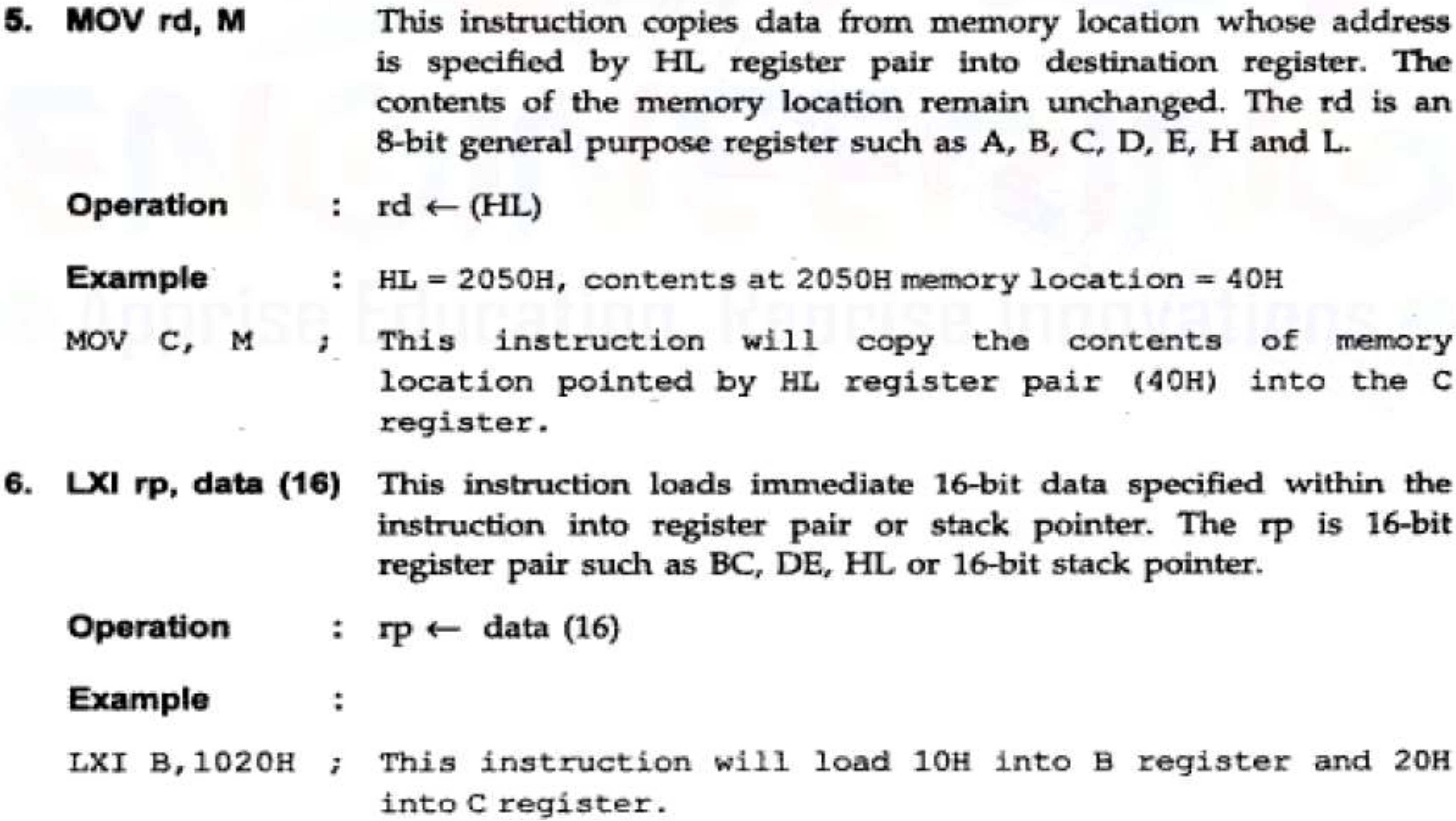
4. MOV M, rs — This instruction copies data from the source register into memory location pointed by the HL register pair. The rs is an 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : (HL) ← rs

Example : `If HL = 2050H, B = 30H.`

`MOV M, B ;` This instruction will copy the contents of B register (30H) into the memory location whose address is specified by HL (2050H).

Microprocessor……..



5. MOV rd, M This instruction copies data from memory location whose address is specified by HL register pair into destination register. The contents of the memory location remain unchanged. The rd is an 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : rd ← (HL)

Example : HL = 2050H, contents at 2050H memory location = 40H

MOV C, M ; This instruction will copy the contents of memory location pointed by HL register pair (40H) into the C register.

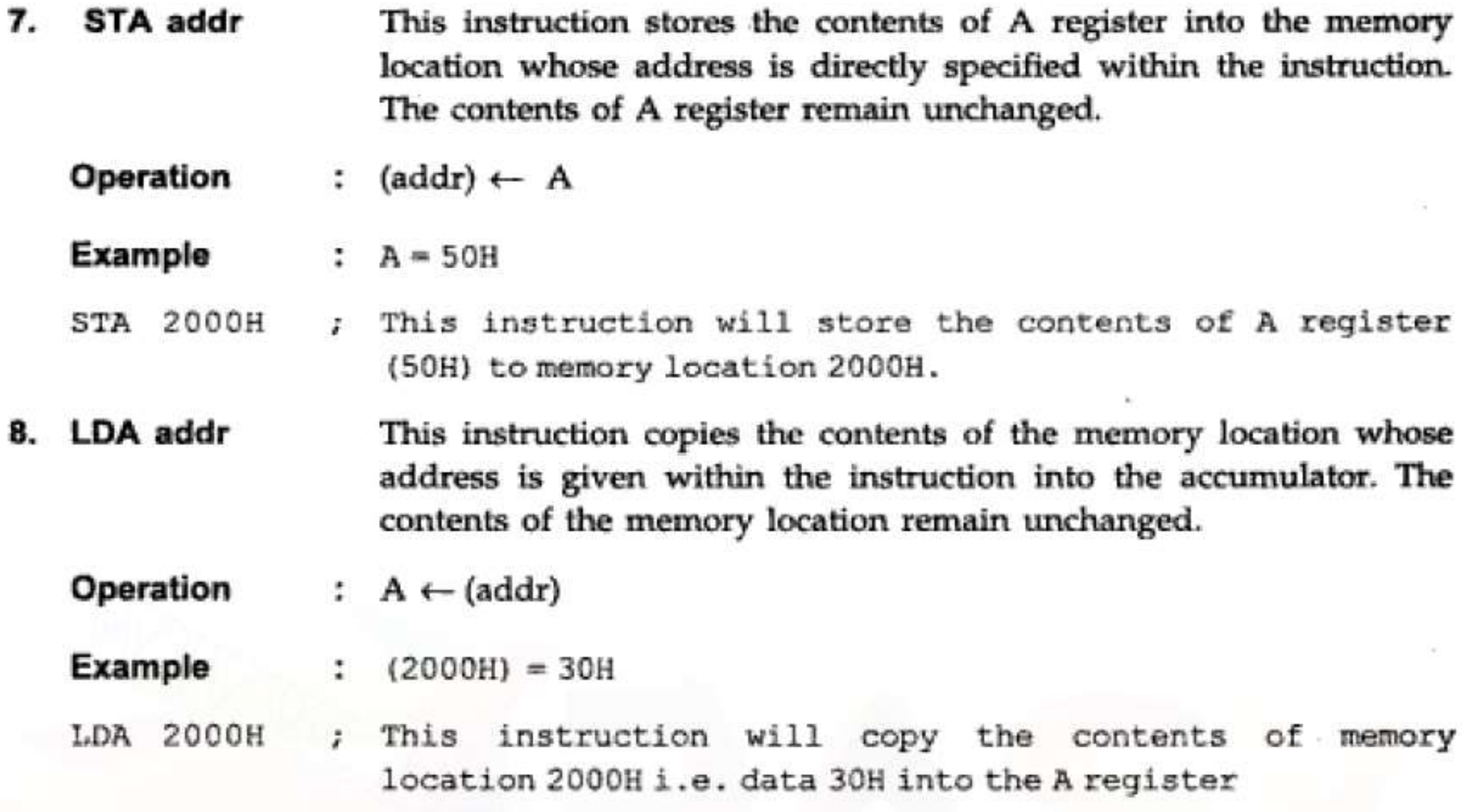
6. LXI rp, data (16) This instruction loads immediate 16-bit data specified within the instruction into register pair or stack pointer. The rp is 16-bit register pair such as BC, DE, HL or 16-bit stack pointer.

Operation : rp ← data (16)

Example :

LXI B,1020H ; This instruction will load 10H into B register and 20H into C register.

Microprocessor……..



7. STA addr This instruction stores the contents of A register into the memory location whose address is directly specified within the instruction. The contents of A register remain unchanged.

Operation : (addr) ← A

Example : A = 50H

```
STA  2000H    ; This instruction will store the contents of A register
                (50H) to memory location 2000H.
```

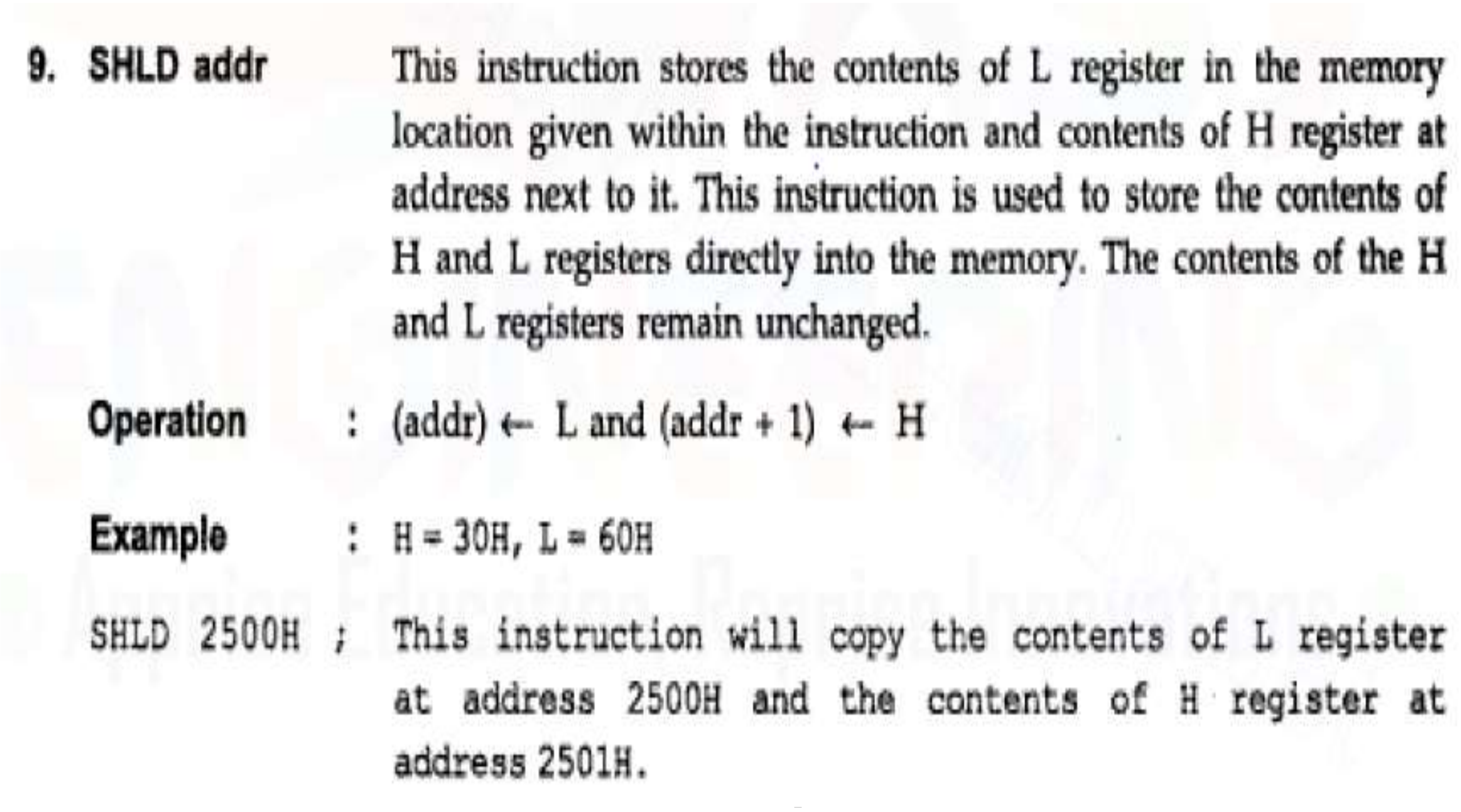
8. LDA addr This instruction copies the contents of the memory location whose address is given within the instruction into the accumulator. The contents of the memory location remain unchanged.

Operation : A ← (addr)

Example : (2000H) = 30H

```
LDA  2000H    ; This instruction will copy the contents of memory
                location 2000H i.e. data 30H into the A register
```

Microprocessor……..



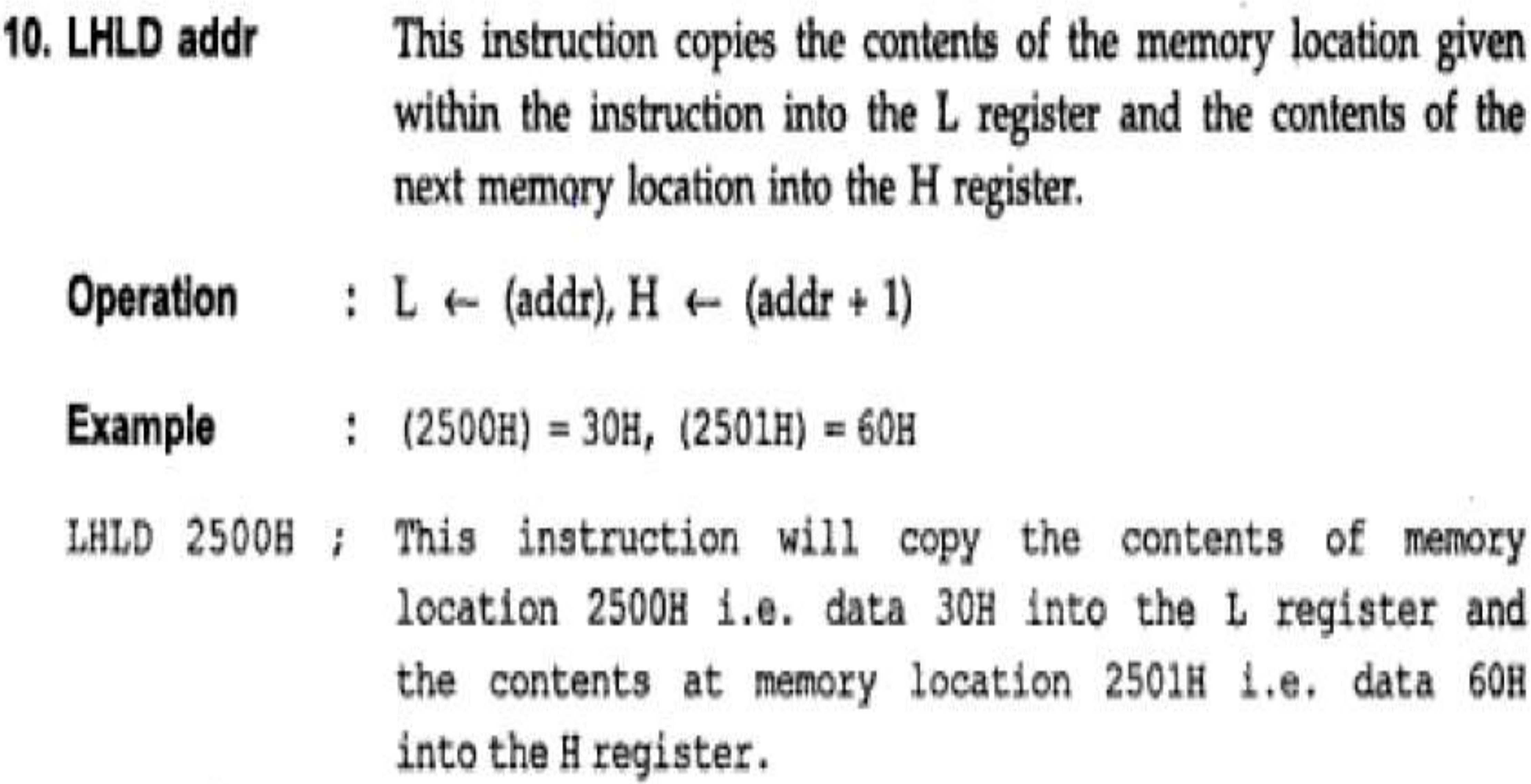
9. SHLD addr This instruction stores the contents of L register in the memory location given within the instruction and contents of H register at address next to it. This instruction is used to store the contents of H and L registers directly into the memory. The contents of the H and L registers remain unchanged.

Operation : (addr) ← L and (addr + 1) ← H

Example : H = 30H, L = 60H

SHLD 2500H ; This instruction will copy the contents of L register at address 2500H and the contents of H register at address 2501H.

Microprocessor……..



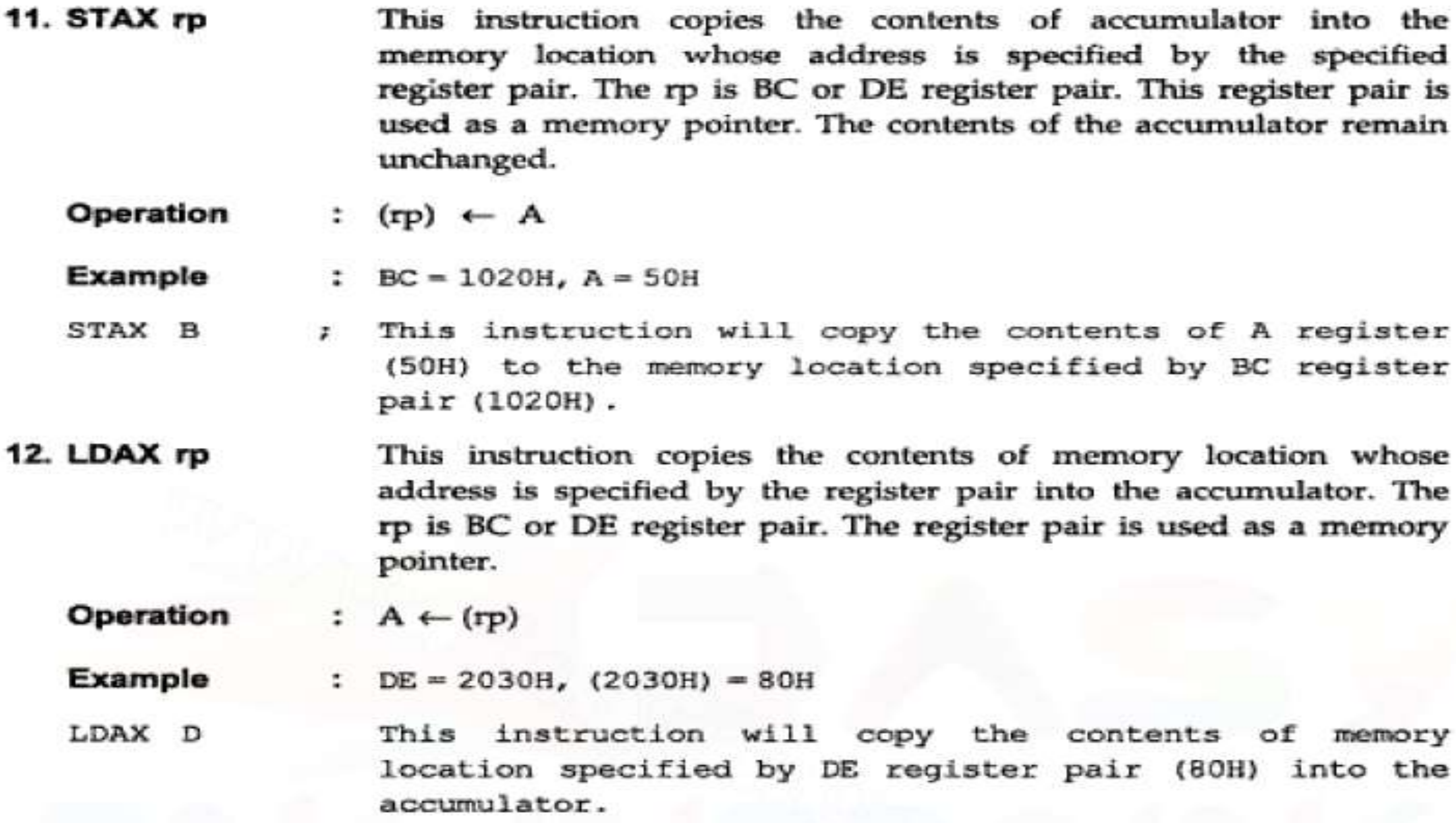
10. LHLD addr This instruction copies the contents of the memory location given within the instruction into the L register and the contents of the next memory location into the H register.

Operation : L ← (addr), H ← (addr + 1)

Example : (2500H) = 30H, (2501H) = 60H

LHLD 2500H ; This instruction will copy the contents of memory location 2500H i.e. data 30H into the L register and the contents at memory location 2501H i.e. data 60H into the H register.

Microprocessor……..



11. STAX rp This instruction copies the contents of accumulator into the memory location whose address is specified by the specified register pair. The rp is BC or DE register pair. This register pair is used as a memory pointer. The contents of the accumulator remain unchanged.

Operation : (rp) ← A

Example : `BC = 1020H, A = 50H`

`STAX B` ; `This instruction will copy the contents of A register (50H) to the memory location specified by BC register pair (1020H).`

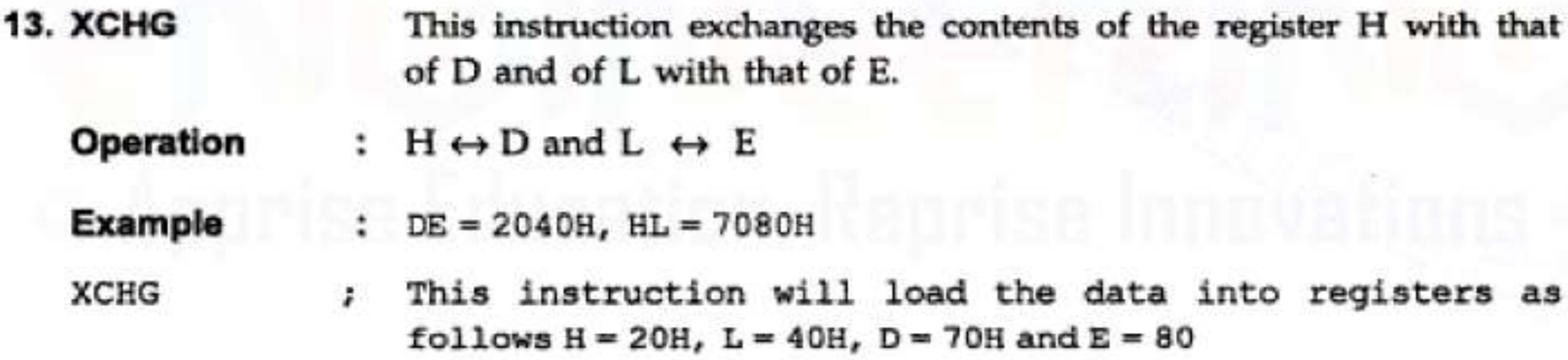
12. LDAX rp This instruction copies the contents of memory location whose address is specified by the register pair into the accumulator. The rp is BC or DE register pair. The register pair is used as a memory pointer.

Operation : A ← (rp)

Example : `DE = 2030H, (2030H) = 80H`

`LDAX D` `This instruction will copy the contents of memory location specified by DE register pair (80H) into the accumulator.`

Microprocessor……..

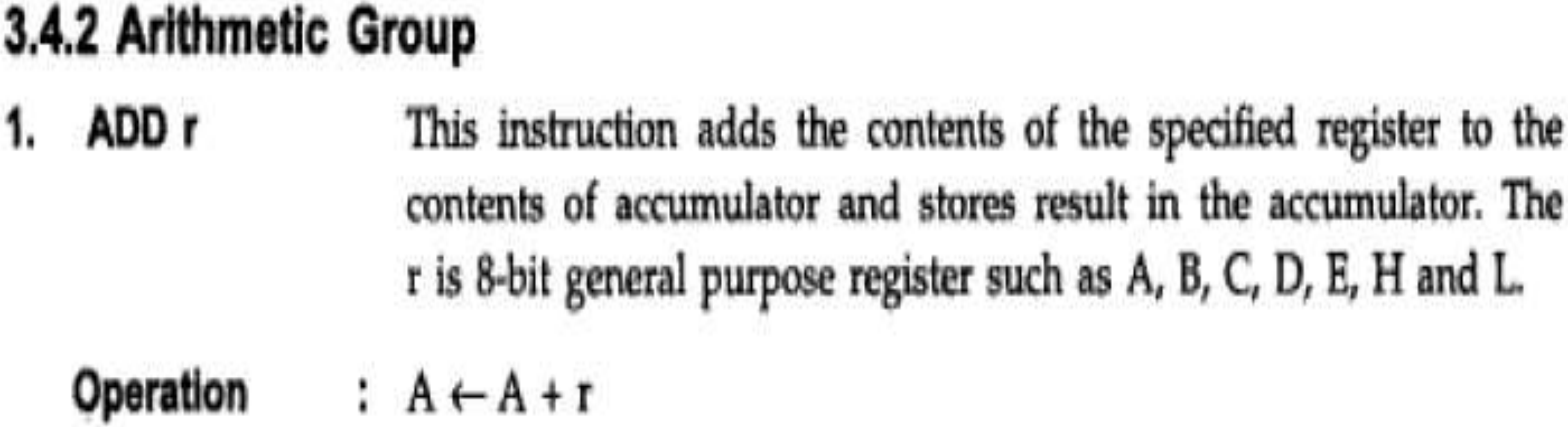


13. XCHG This instruction exchanges the contents of the register H with that of D and of L with that of E.

Operation : H ↔ D and L ↔ E

Example : DE = 2040H, HL = 7080H

```
XCHG          ; This instruction will load the data into registers as
                follows H = 20H, L = 40H, D = 70H and E = 80
```

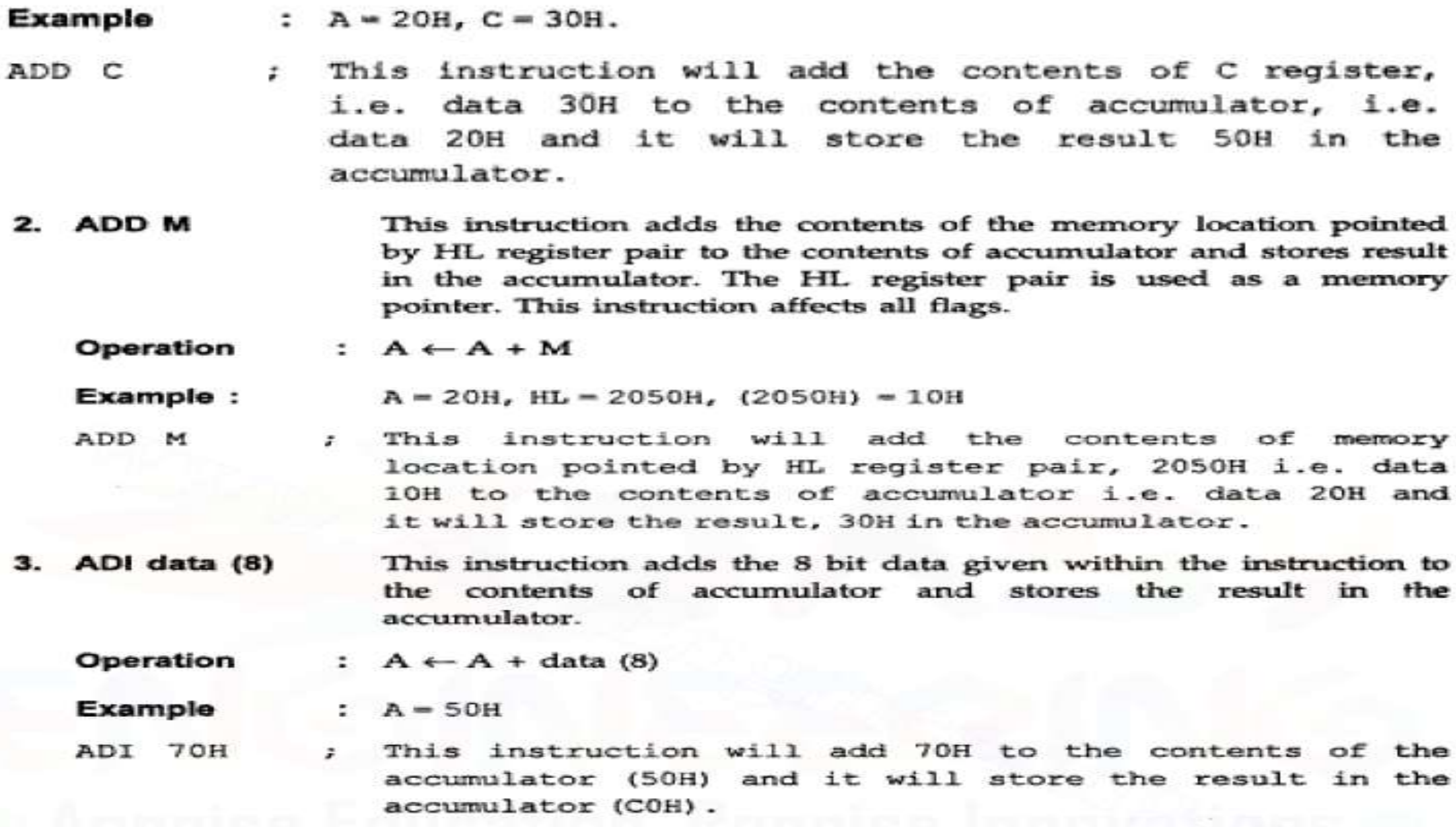


3.4.2 Arithmetic Group

1. ADD r This instruction adds the contents of the specified register to the contents of accumulator and stores result in the accumulator. The r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : A ← A + r

Microprocessor……..



Example : A = 20H, C = 30H.

ADD C ; This instruction will add the contents of C register, i.e. data 30H to the contents of accumulator, i.e. data 20H and it will store the result 50H in the accumulator.

2. ADD M This instruction adds the contents of the memory location pointed by HL register pair to the contents of accumulator and stores result in the accumulator. The HL register pair is used as a memory pointer. This instruction affects all flags.

Operation : A ← A + M

Example : A = 20H, HL = 2050H, (2050H) = 10H

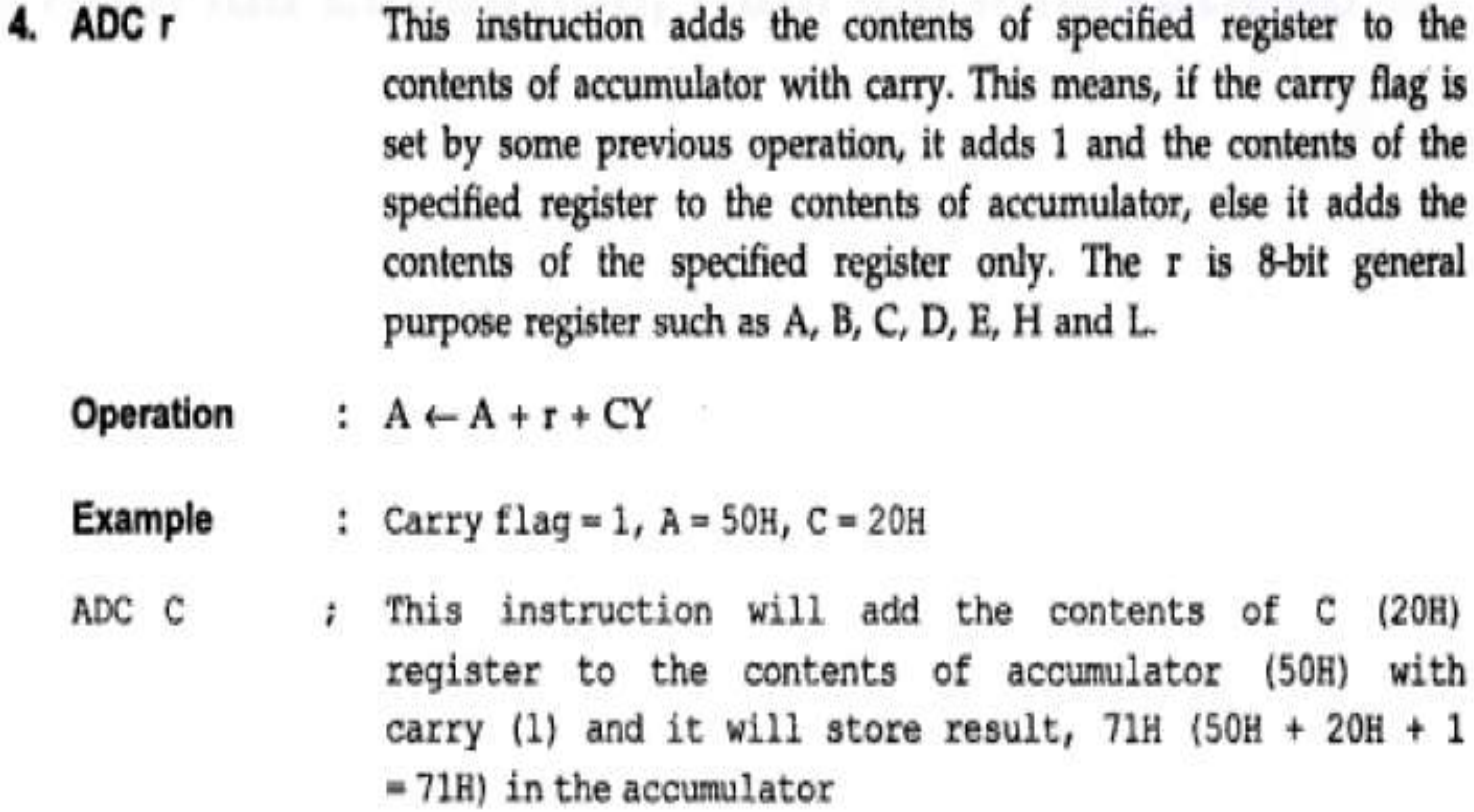
ADD M ; This instruction will add the contents of memory location pointed by HL register pair, 2050H i.e. data 10H to the contents of accumulator i.e. data 20H and it will store the result, 30H in the accumulator.

3. ADI data (8) This instruction adds the 8 bit data given within the instruction to the contents of accumulator and stores the result in the accumulator.

Operation : A ← A + data (8)

Example : A = 50H

ADI 70H ; This instruction will add 70H to the contents of the accumulator (50H) and it will store the result in the accumulator (C0H).



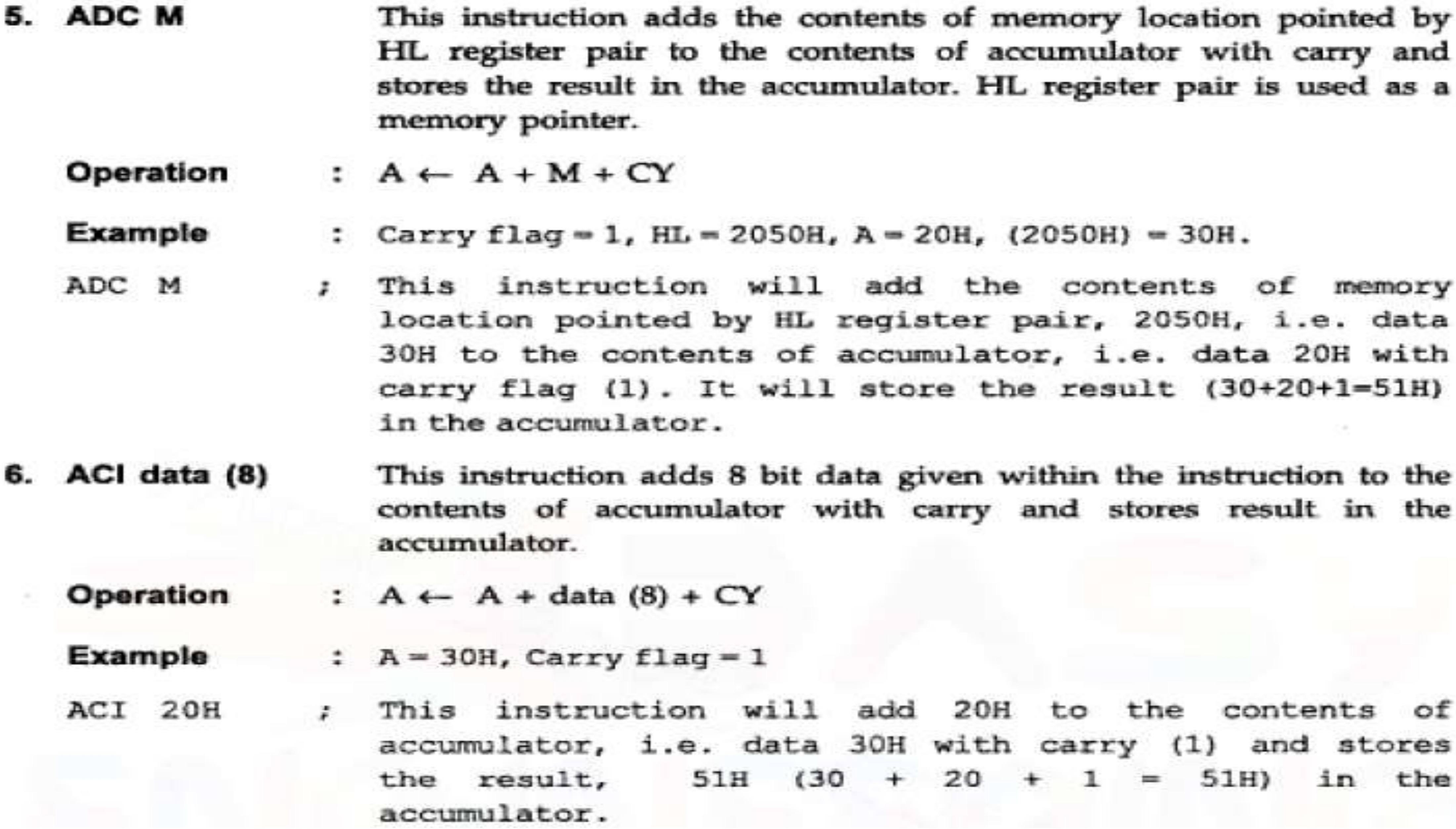
4. ADC r This instruction adds the contents of specified register to the contents of accumulator with carry. This means, if the carry flag is set by some previous operation, it adds 1 and the contents of the specified register to the contents of accumulator, else it adds the contents of the specified register only. The r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : A ← A + r + CY

Example : Carry flag = 1, A = 50H, C = 20H

```
ADC  C        ;  This instruction will add the contents of C (20H)
                 register to the contents of accumulator (50H) with
                 carry (1) and it will store result, 71H (50H + 20H + 1
                 = 71H) in the accumulator
```

Microprocessor……..



5. ADC M This instruction adds the contents of memory location pointed by HL register pair to the contents of accumulator with carry and stores the result in the accumulator. HL register pair is used as a memory pointer.

Operation : $A \leftarrow A + M + CY$

Example : Carry flag = 1, HL = 2050H, A = 20H, (2050H) = 30H.

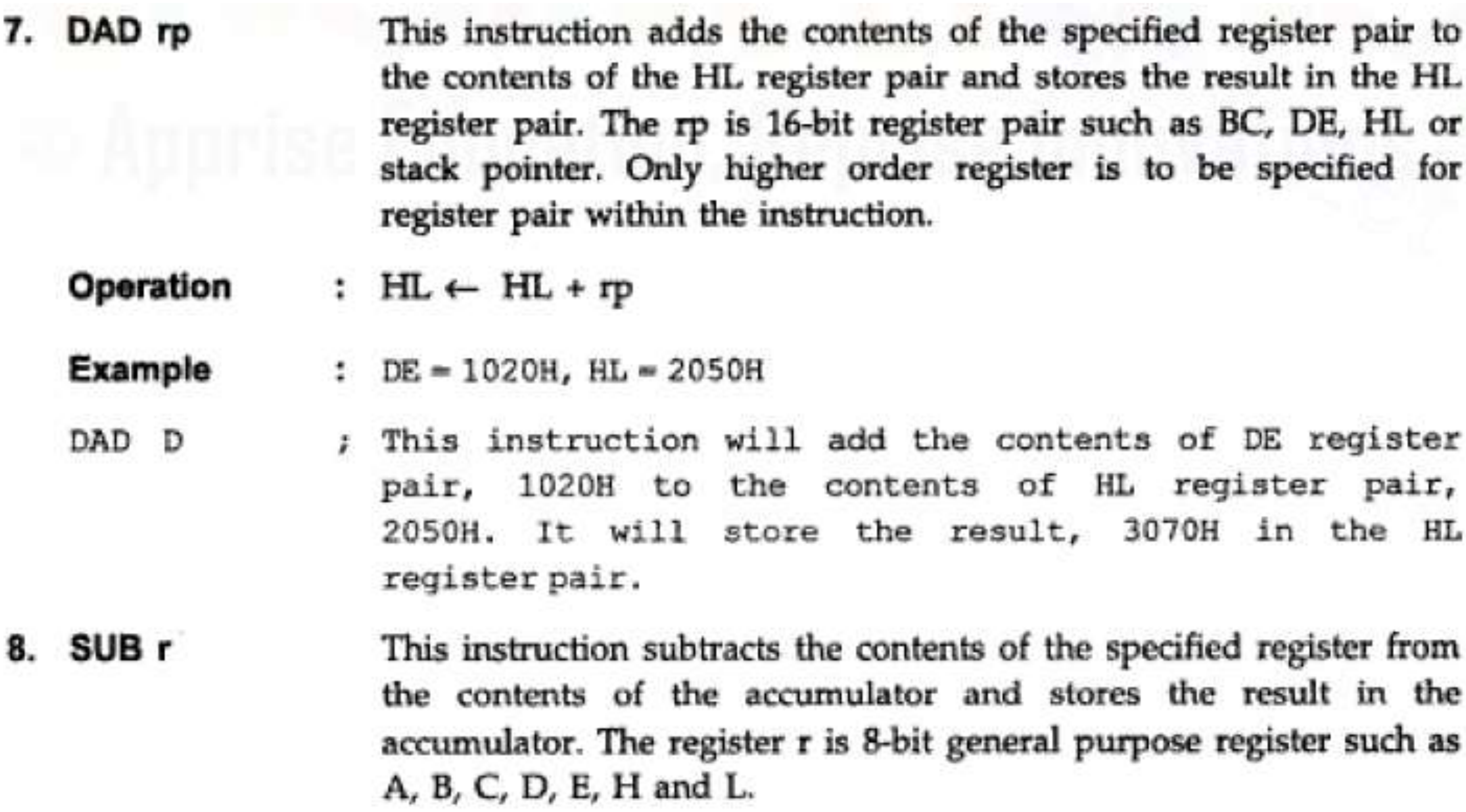
ADC M ; This instruction will add the contents of memory location pointed by HL register pair, 2050H, i.e. data 30H to the contents of accumulator, i.e. data 20H with carry flag (1). It will store the result (30+20+1=51H) in the accumulator.

6. ACI data (8) This instruction adds 8 bit data given within the instruction to the contents of accumulator with carry and stores result in the accumulator.

Operation : $A \leftarrow A + \text{data (8)} + CY$

Example : A = 30H, Carry flag = 1

ACI 20H ; This instruction will add 20H to the contents of accumulator, i.e. data 30H with carry (1) and stores the result, 51H (30 + 20 + 1 = 51H) in the accumulator.



7. DAD rp This instruction adds the contents of the specified register pair to the contents of the HL register pair and stores the result in the HL register pair. The rp is 16-bit register pair such as BC, DE, HL or stack pointer. Only higher order register is to be specified for register pair within the instruction.

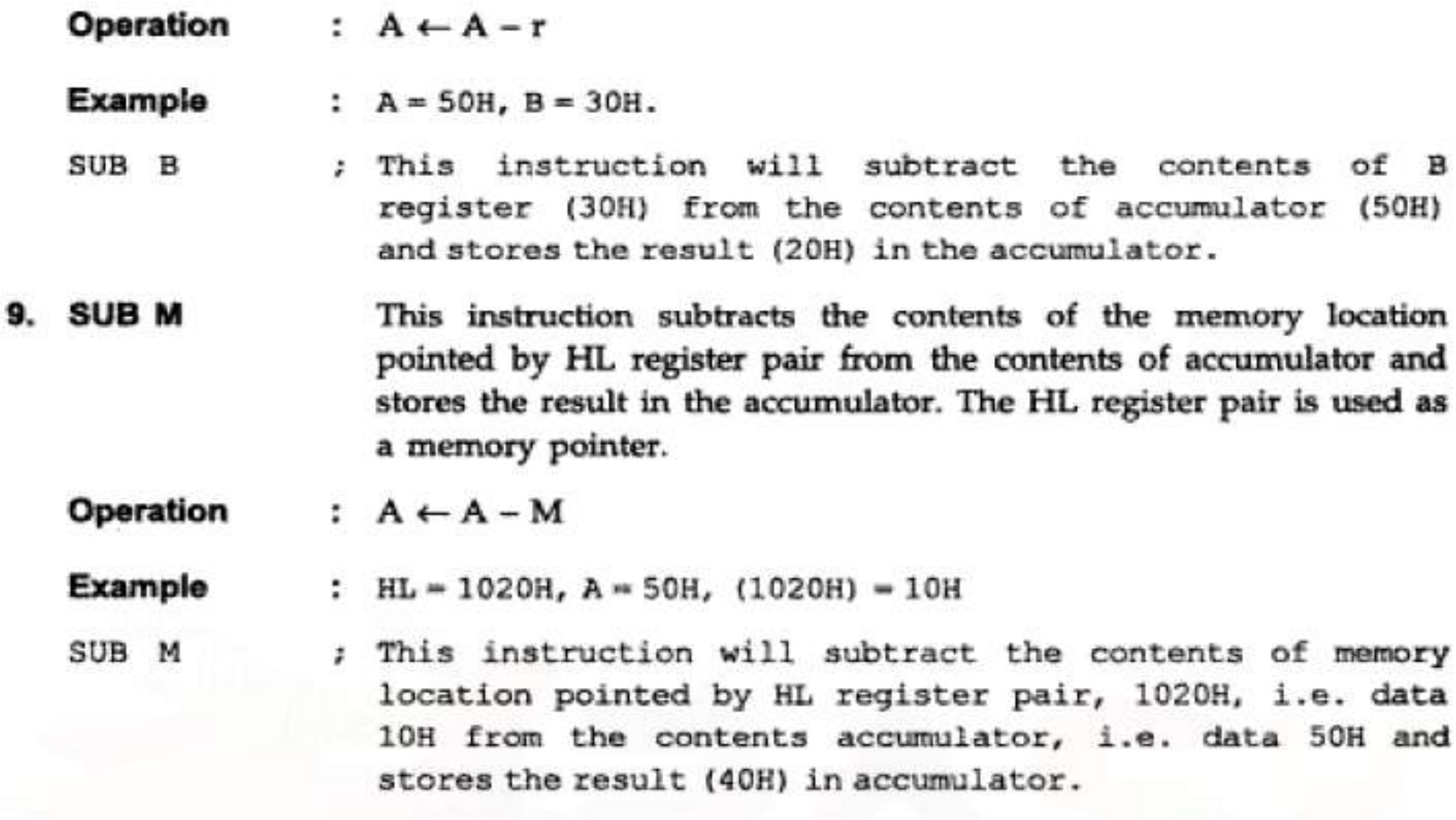
Operation : HL ← HL + rp

Example : DE = 1020H, HL = 2050H

`DAD D` ; This instruction will add the contents of DE register pair, 1020H to the contents of HL register pair, 2050H. It will store the result, 3070H in the HL register pair.

8. SUB r This instruction subtracts the contents of the specified register from the contents of the accumulator and stores the result in the accumulator. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Microprocessor……..



Operation : $A \leftarrow A - r$

Example : A = 50H, B = 30H.

```
SUB  B        ; This instruction will subtract the contents of B
                register (30H) from the contents of accumulator (50H)
                and stores the result (20H) in the accumulator.
```

9. SUB M

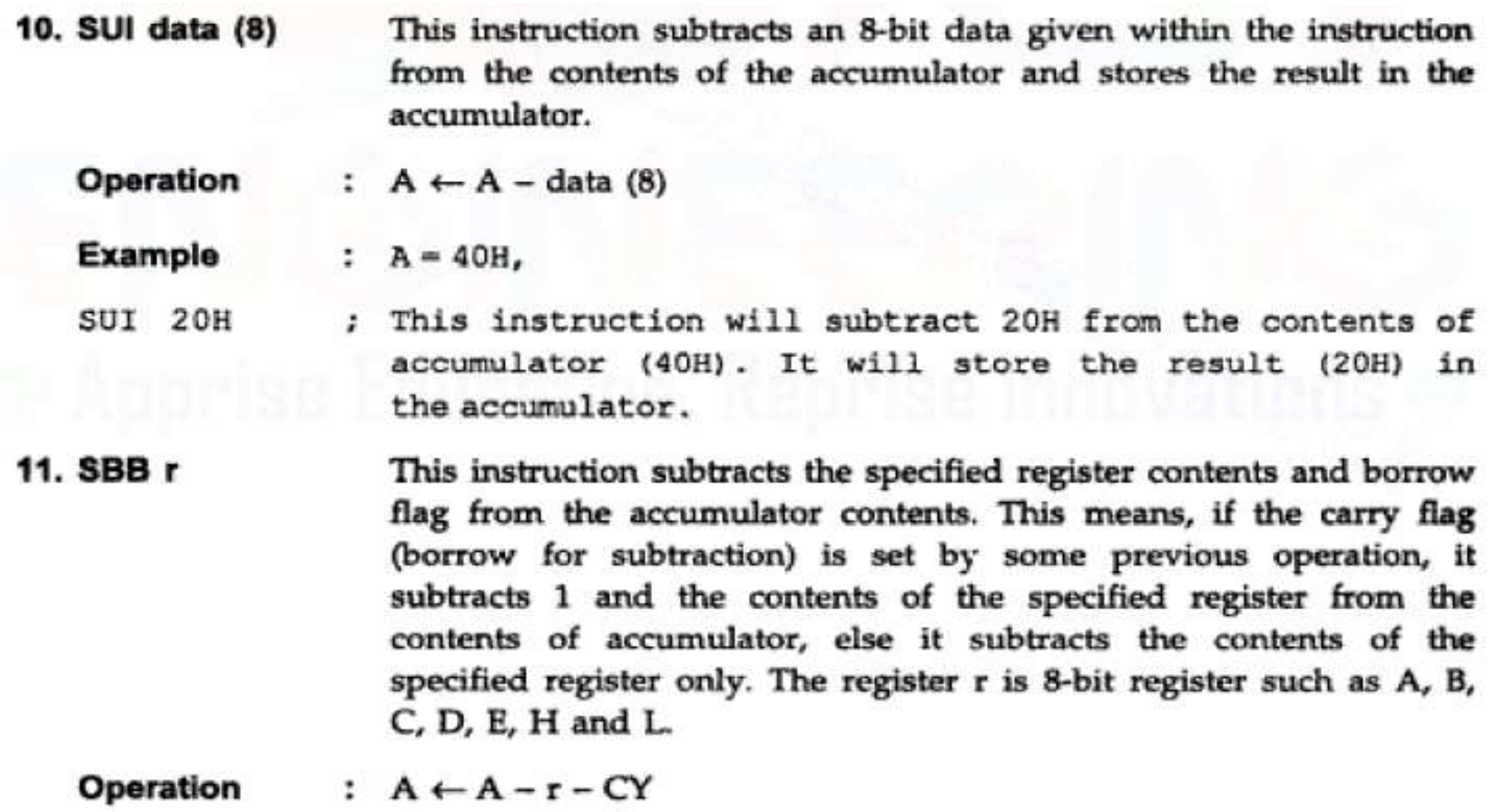
This instruction subtracts the contents of the memory location pointed by HL register pair from the contents of accumulator and stores the result in the accumulator. The HL register pair is used as a memory pointer.

Operation : $A \leftarrow A - M$

Example : HL = 1020H, A = 50H, (1020H) = 10H

```
SUB  M        ; This instruction will subtract the contents of memory
                location pointed by HL register pair, 1020H, i.e. data
                10H from the contents accumulator, i.e. data 50H and
                stores the result (40H) in accumulator.
```

Microprocessor……..



10. SUI data (8) This instruction subtracts an 8-bit data given within the instruction from the contents of the accumulator and stores the result in the accumulator.

Operation : A ← A – data (8)

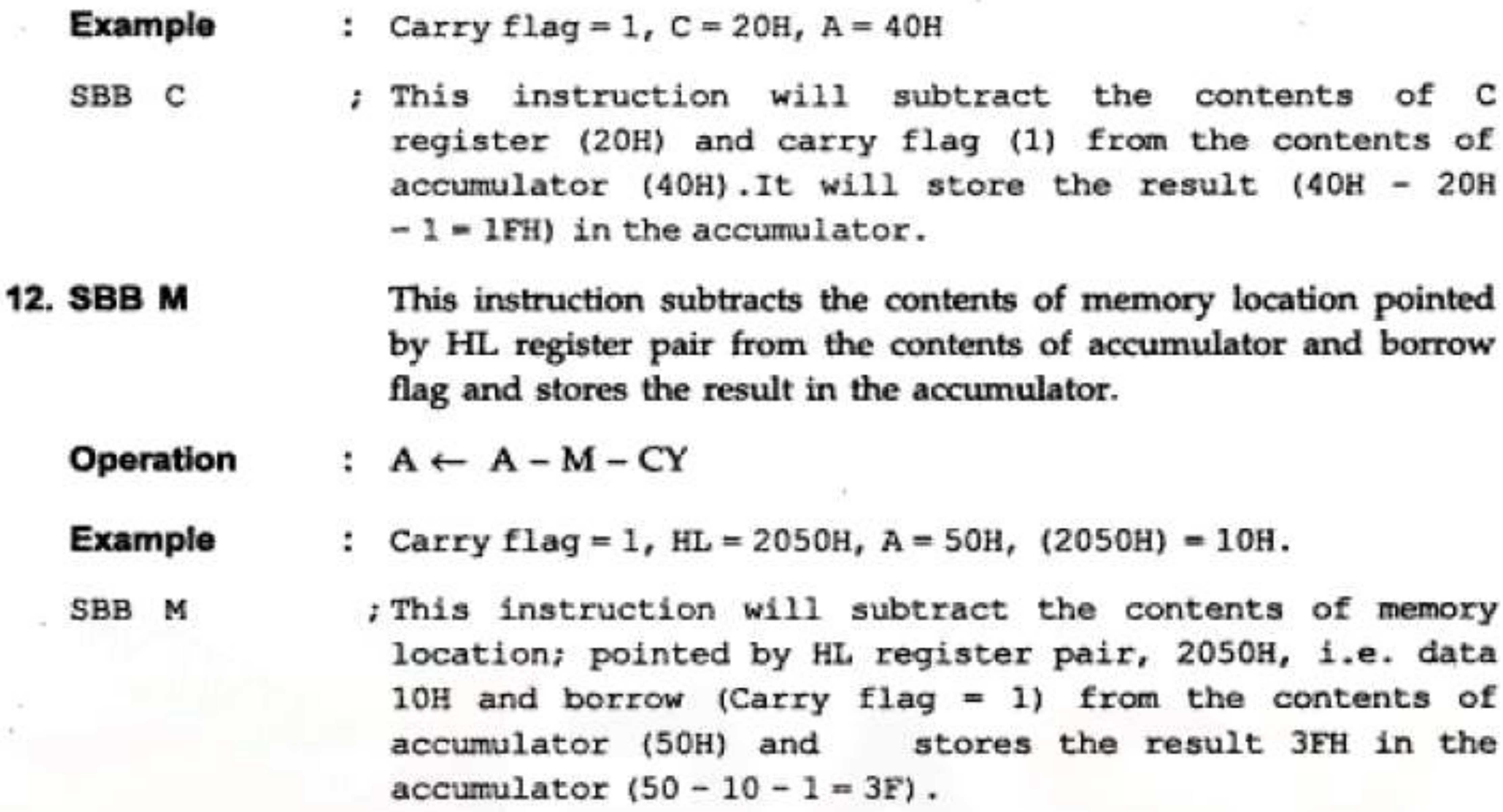
Example : `A = 40H,`

```
SUI  20H      ; This instruction will subtract 20H from the contents of
                accumulator (40H). It will store the result (20H) in
                the accumulator.
```

11. SBB r This instruction subtracts the specified register contents and borrow flag from the accumulator contents. This means, if the carry flag (borrow for subtraction) is set by some previous operation, it subtracts 1 and the contents of the specified register from the contents of accumulator, else it subtracts the contents of the specified register only. The register r is 8-bit register such as A, B, C, D, E, H and L.

Operation : A ← A – r – CY

Microprocessor……..



Example : Carry flag = 1, C = 20H, A = 40H

SBB C ; This instruction will subtract the contents of C register (20H) and carry flag (1) from the contents of accumulator (40H).It will store the result (40H - 20H - 1 = 1FH) in the accumulator.

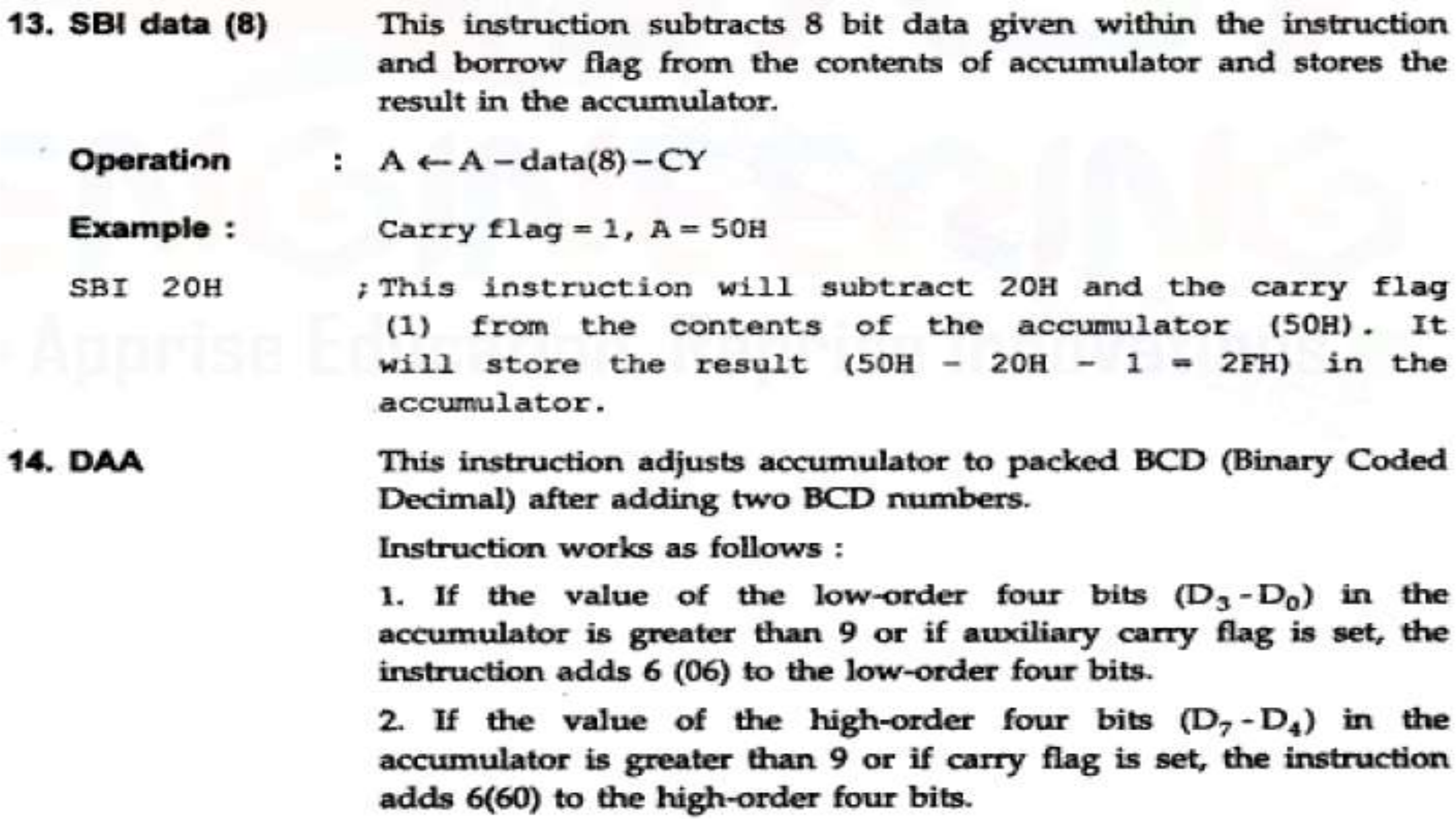
12. SBB M This instruction subtracts the contents of memory location pointed by HL register pair from the contents of accumulator and borrow flag and stores the result in the accumulator.

Operation : A ← A – M – CY

Example : Carry flag = 1, HL = 2050H, A = 50H, (2050H) = 10H.

SBB M ;This instruction will subtract the contents of memory location; pointed by HL register pair, 2050H, i.e. data 10H and borrow (Carry flag = 1) from the contents of accumulator (50H) and stores the result 3FH in the accumulator (50 – 10 – 1 = 3F).

Microprocessor……..



13. SBI data (8) This instruction subtracts 8 bit data given within the instruction and borrow flag from the contents of accumulator and stores the result in the accumulator.

Operation : $A \leftarrow A - data(8) - CY$

Example : `Carry flag = 1, A = 50H`

```
SBI  20H    ;This instruction will subtract 20H and the carry flag
            (1) from the contents of the accumulator (50H). It
            will store the result (50H - 20H - 1 = 2FH) in the
            accumulator.
```

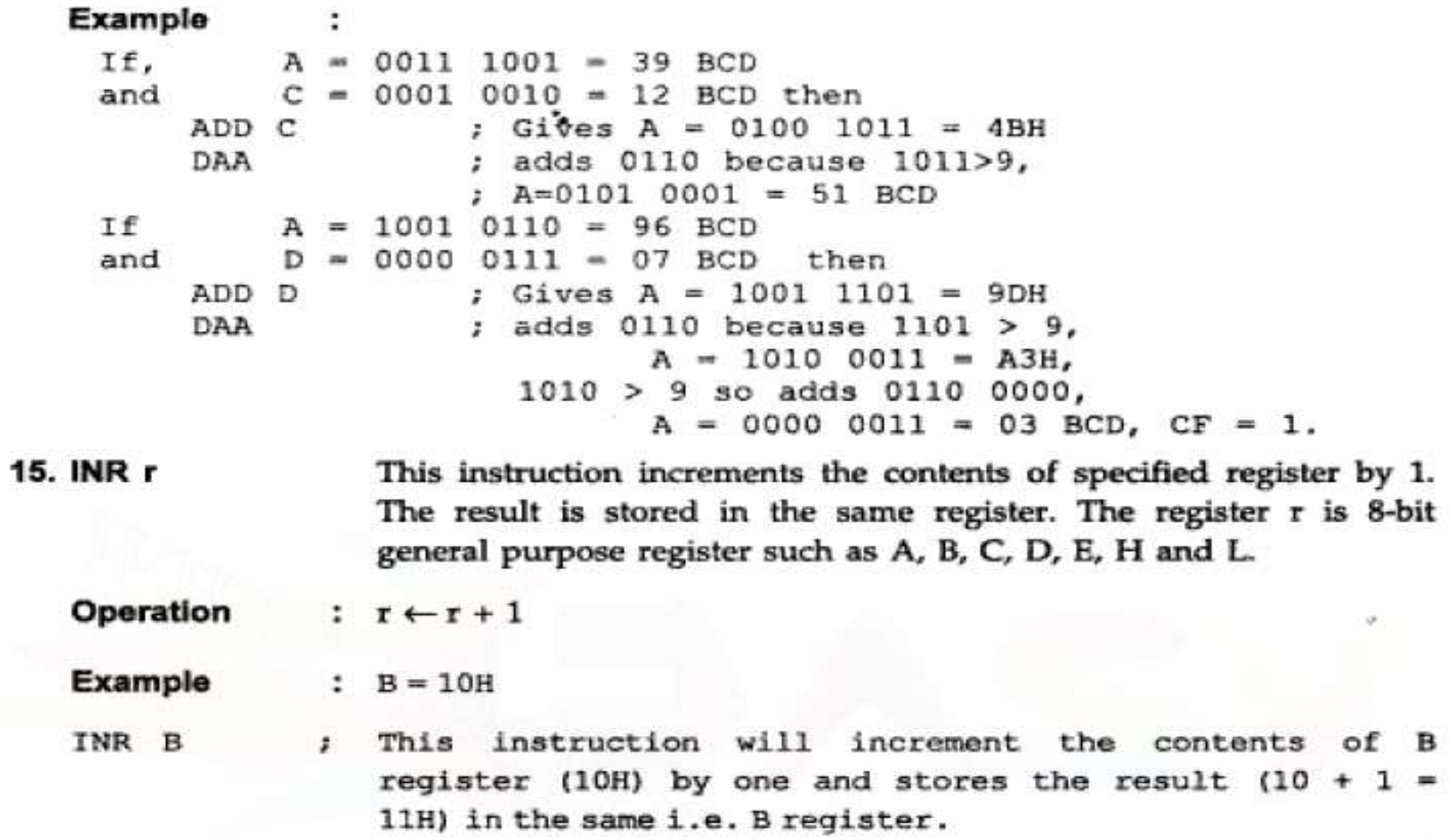
14. DAA This instruction adjusts accumulator to packed BCD (Binary Coded Decimal) after adding two BCD numbers.

Instruction works as follows :

1. If the value of the low-order four bits ($D_3 - D_0$) in the accumulator is greater than 9 or if auxiliary carry flag is set, the instruction adds 6 (06) to the low-order four bits.

2. If the value of the high-order four bits ($D_7 - D_4$) in the accumulator is greater than 9 or if carry flag is set, the instruction adds 6(60) to the high-order four bits.

Microprocessor……..



Example :

```
If,        A = 0011 1001 = 39 BCD
and        C = 0001 0010 = 12 BCD then
     ADD C             ; Gives A = 0100 1011 = 4BH
     DAA               ; adds 0110 because 1011>9,
                       ; A=0101 0001 = 51 BCD
If         A = 1001 0110 = 96 BCD
and        D = 0000 0111 = 07 BCD  then
     ADD D             ; Gives A = 1001 1101 = 9DH
     DAA               ; adds 0110 because 1101 > 9,
                                 A = 1010 0011 = A3H,
                         1010 > 9 so adds 0110 0000,
                                 A = 0000 0011 = 03 BCD, CF = 1.
```

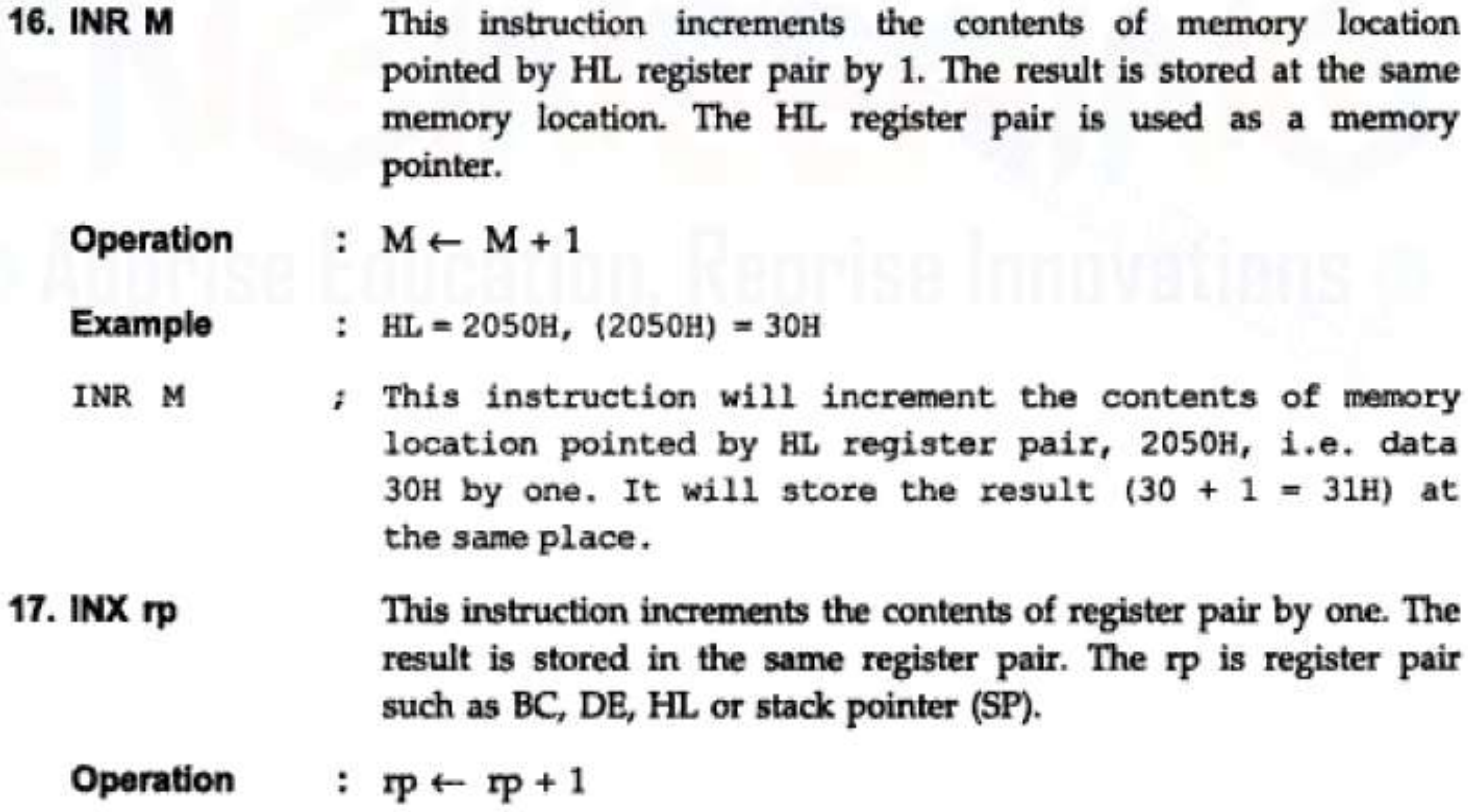
15. INR r This instruction increments the contents of specified register by 1. The result is stored in the same register. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : r ← r + 1

Example : B = 10H

```
INR B      ; This instruction will increment the contents of B
             register (10H) by one and stores the result (10 + 1 =
             11H) in the same i.e. B register.
```

Microprocessor……..



16. INR M This instruction increments the contents of memory location pointed by HL register pair by 1. The result is stored at the same memory location. The HL register pair is used as a memory pointer.

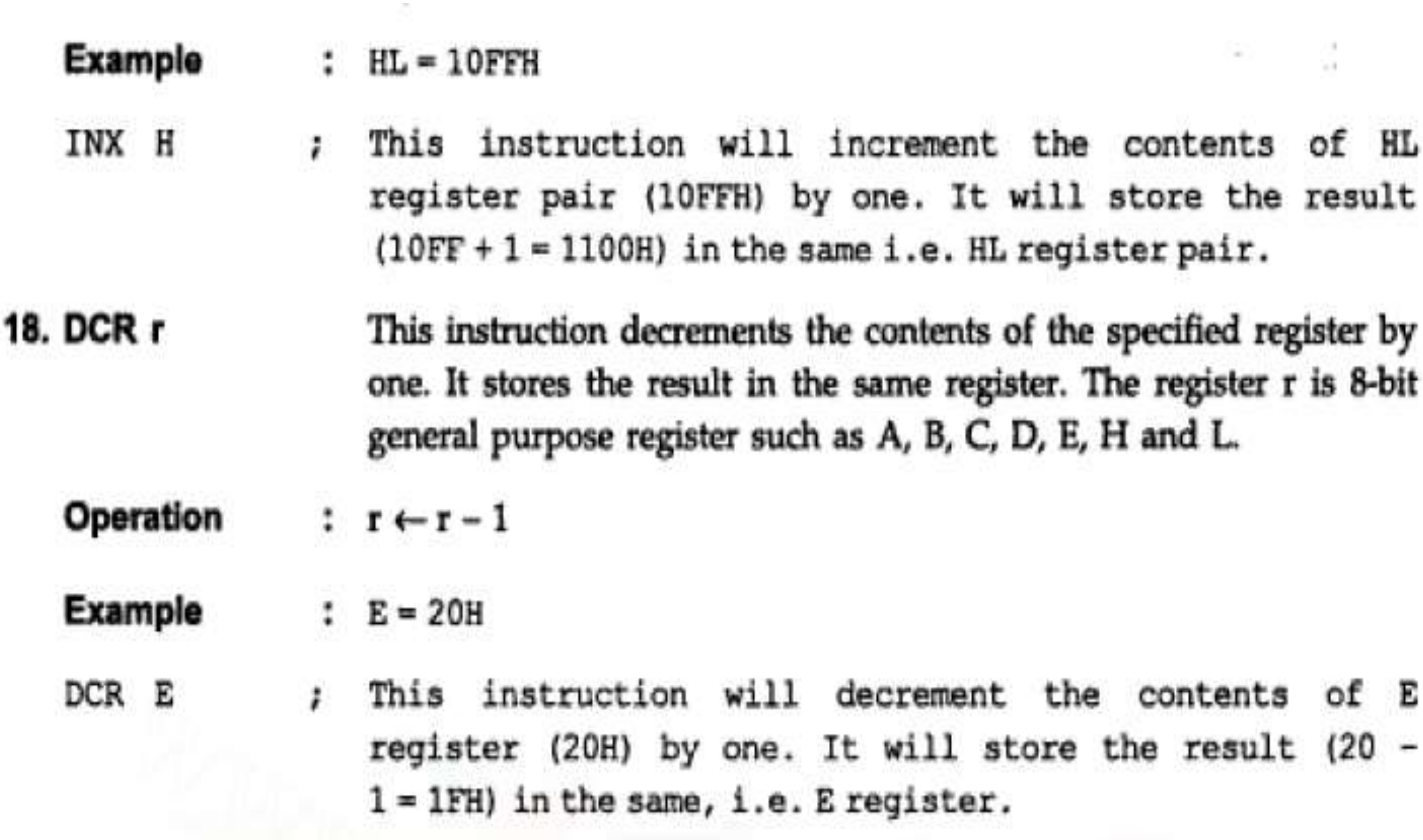
Operation : M ← M + 1

Example : HL = 2050H, (2050H) = 30H

INR M ; This instruction will increment the contents of memory location pointed by HL register pair, 2050H, i.e. data 30H by one. It will store the result (30 + 1 = 31H) at the same place.

17. INX rp This instruction increments the contents of register pair by one. The result is stored in the same register pair. The rp is register pair such as BC, DE, HL or stack pointer (SP).

Operation : rp ← rp + 1



Example : HL = 10FFH

INX H ; This instruction will increment the contents of HL register pair (10FFH) by one. It will store the result (10FF + 1 = 1100H) in the same i.e. HL register pair.

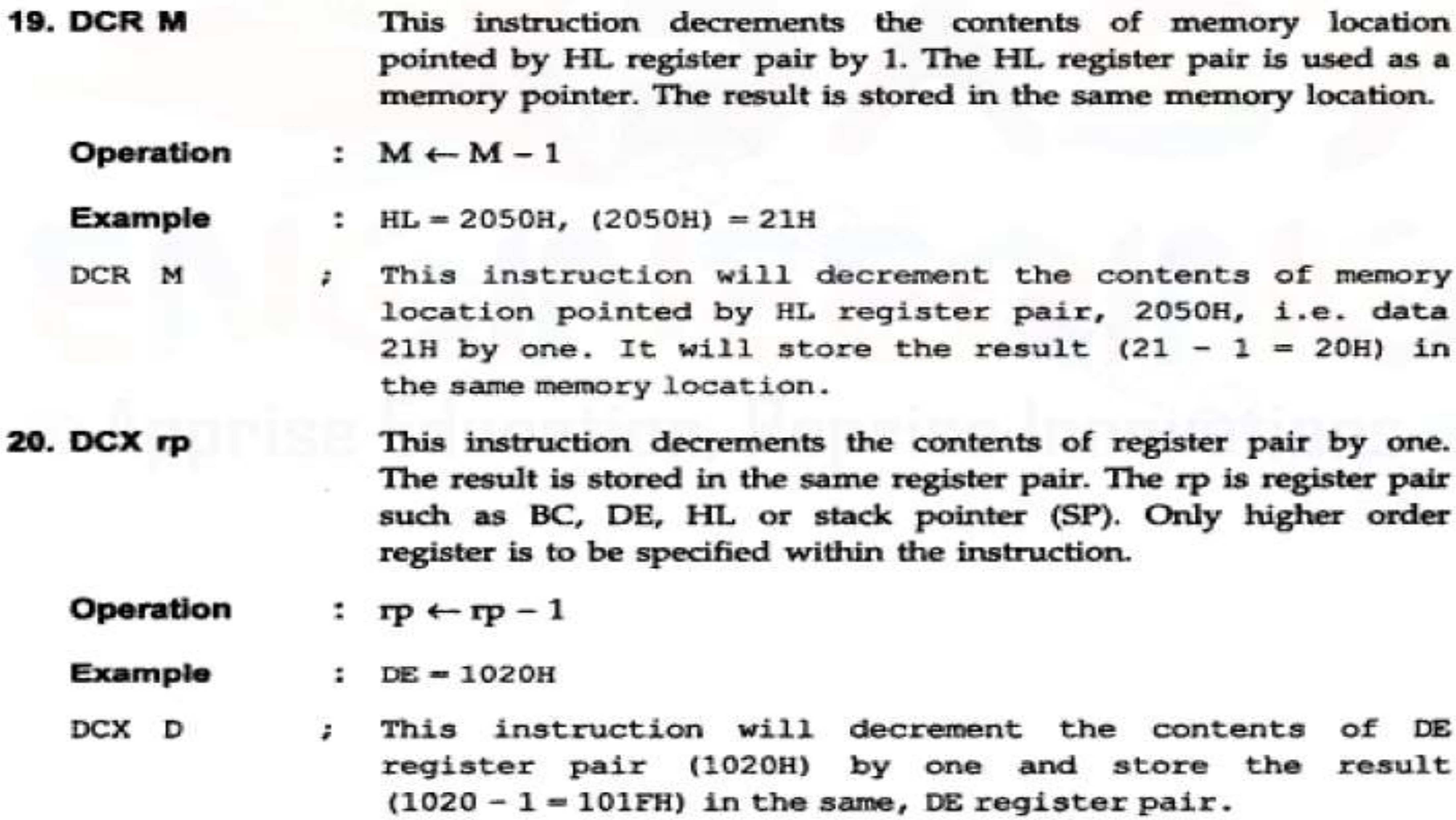
18. DCR r This instruction decrements the contents of the specified register by one. It stores the result in the same register. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : r ← r – 1

Example : E = 20H

DCR E ; This instruction will decrement the contents of E register (20H) by one. It will store the result (20 - 1 = 1FH) in the same, i.e. E register.

Microprocessor……..



19. DCR M This instruction decrements the contents of memory location pointed by HL register pair by 1. The HL register pair is used as a memory pointer. The result is stored in the same memory location.

Operation : M ← M – 1

Example : HL = 2050H, (2050H) = 21H

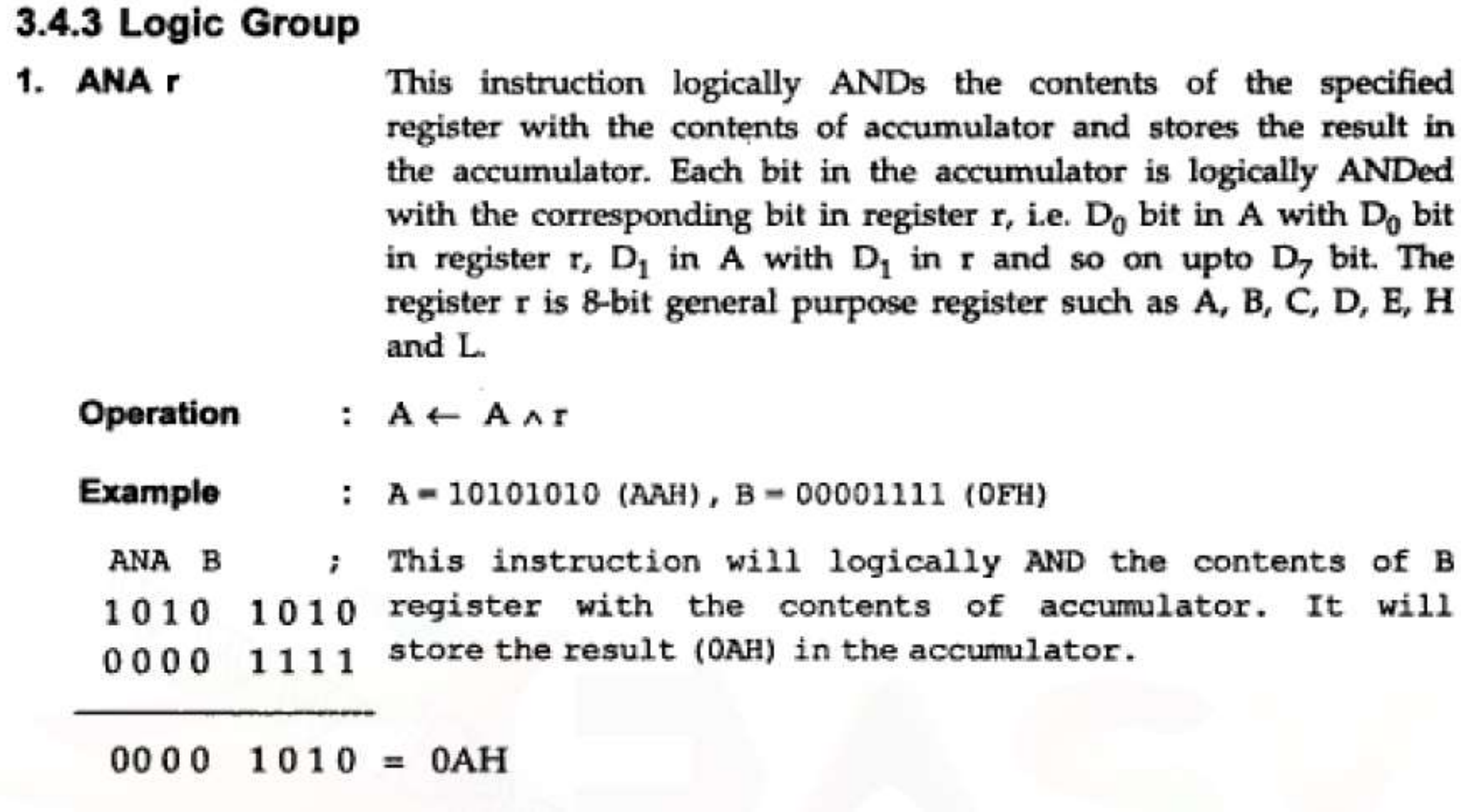
DCR M ; This instruction will decrement the contents of memory location pointed by HL register pair, 2050H, i.e. data 21H by one. It will store the result (21 - 1 = 20H) in the same memory location.

20. DCX rp This instruction decrements the contents of register pair by one. The result is stored in the same register pair. The rp is register pair such as BC, DE, HL or stack pointer (SP). Only higher order register is to be specified within the instruction.

Operation : rp ← rp – 1

Example : DE = 1020H

DCX D ; This instruction will decrement the contents of DE register pair (1020H) by one and store the result (1020 - 1 = 101FH) in the same, DE register pair.



3.4.3 Logic Group

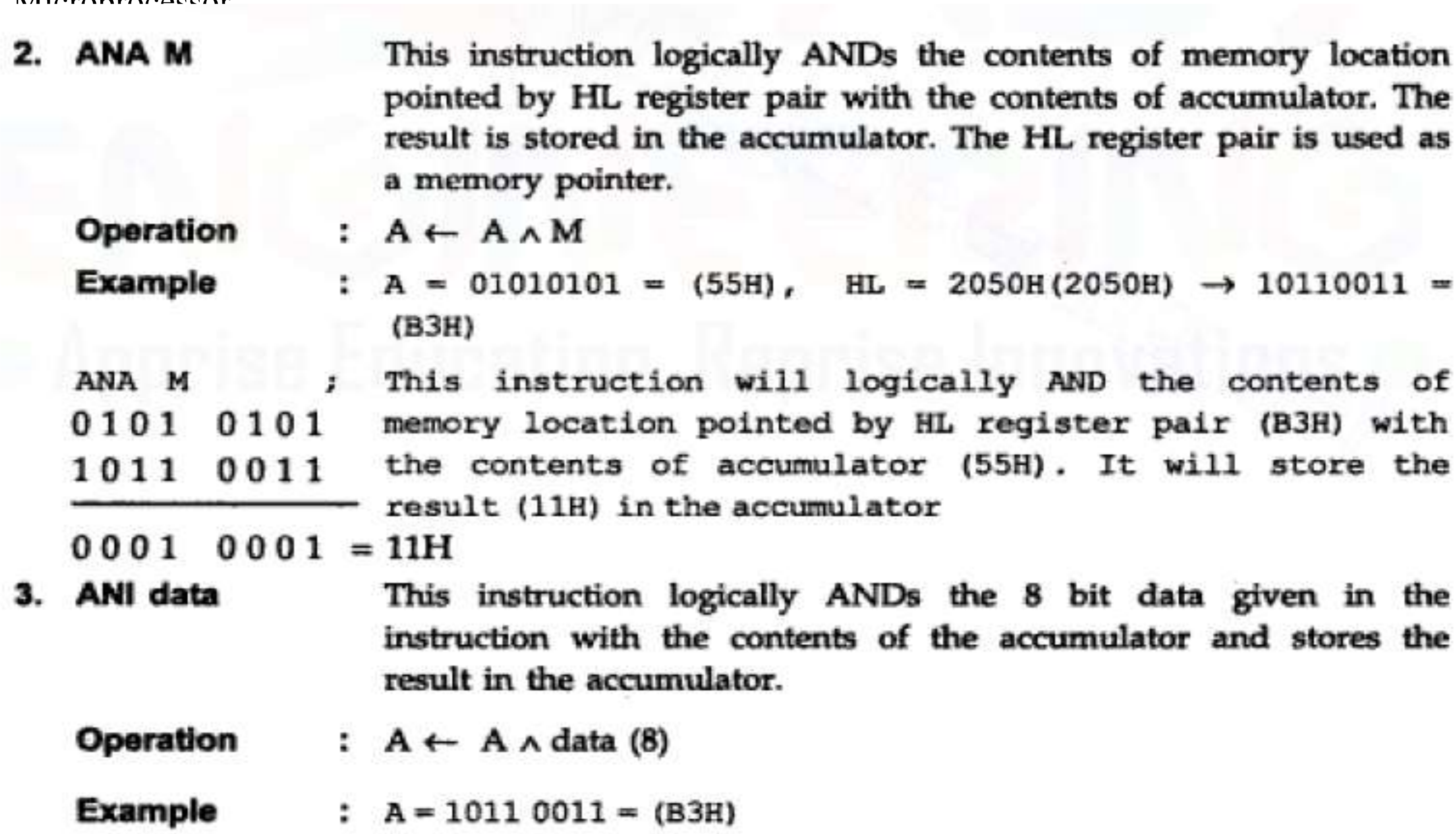
1. ANA r This instruction logically ANDs the contents of the specified register with the contents of accumulator and stores the result in the accumulator. Each bit in the accumulator is logically ANDed with the corresponding bit in register r, i.e. D_0 bit in A with D_0 bit in register r, D_1 in A with D_1 in r and so on upto D_7 bit. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : $A \leftarrow A \wedge r$

Example : A = 10101010 (AAH), B = 00001111 (0FH)

ANA B ; This instruction will logically AND the contents of B register with the contents of accumulator. It will store the result (0AH) in the accumulator.

```
1010 1010
0000 1111
---------
0000 1010 = 0AH
```



2. ANA M This instruction logically ANDs the contents of memory location pointed by HL register pair with the contents of accumulator. The result is stored in the accumulator. The HL register pair is used as a memory pointer.

Operation : A ← A ∧ M

Example : A = 01010101 = (55H), HL = 2050H(2050H) → 10110011 = (B3H)

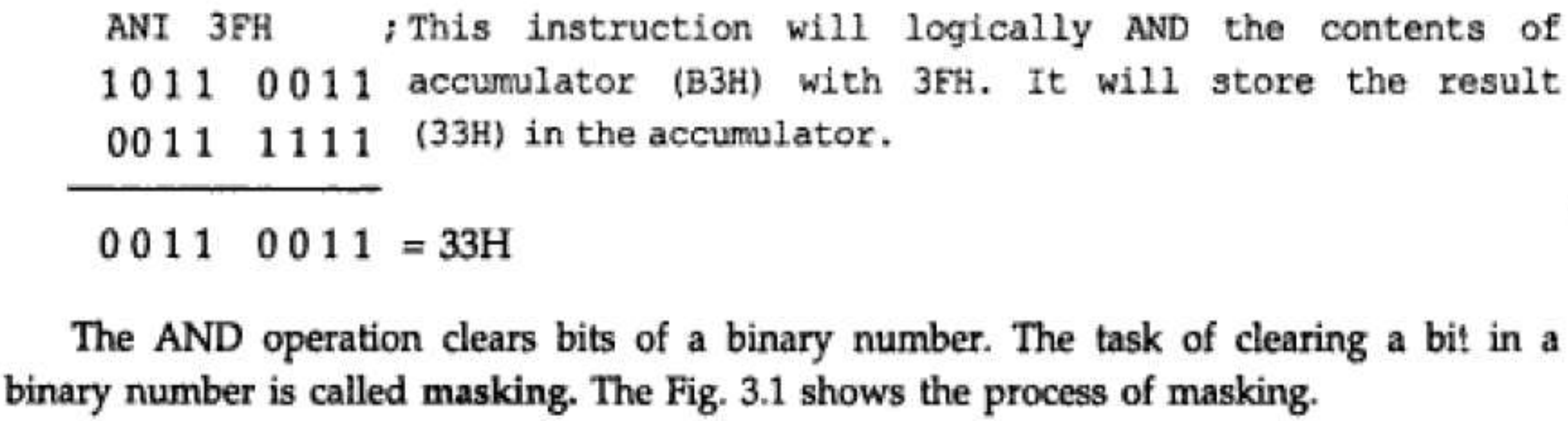
```
ANA  M       ;  This instruction will logically AND the contents of
0101  0101      memory location pointed by HL register pair (B3H) with
1011  0011      the contents of accumulator (55H). It will store the
-----------     result (11H) in the accumulator
0001  0001   = 11H
```

3. ANI data This instruction logically ANDs the 8 bit data given in the instruction with the contents of the accumulator and stores the result in the accumulator.

Operation : A ← A ∧ data (8)

Example : A = 1011 0011 = (B3H)

Microprocessor……..



```
 ANI  3FH     ;This instruction will logically AND the contents of
1011  0011     accumulator (B3H) with 3FH. It will store the result
 0011  1111    (33H) in the accumulator.
----------
0011  0011  = 33H
```

The AND operation clears bits of a binary number. The task of clearing a bit in a binary number is called **masking.** The Fig. 3.1 shows the process of masking.

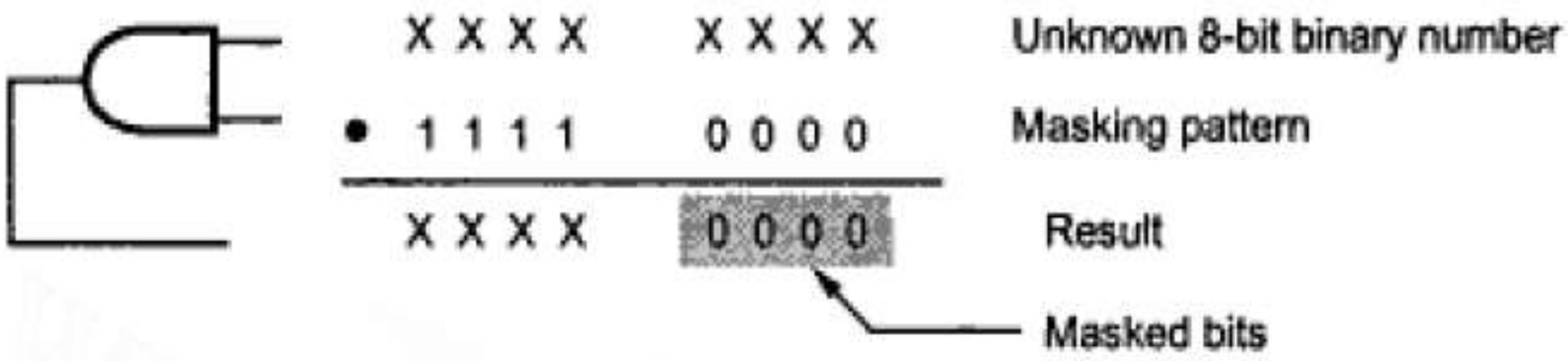
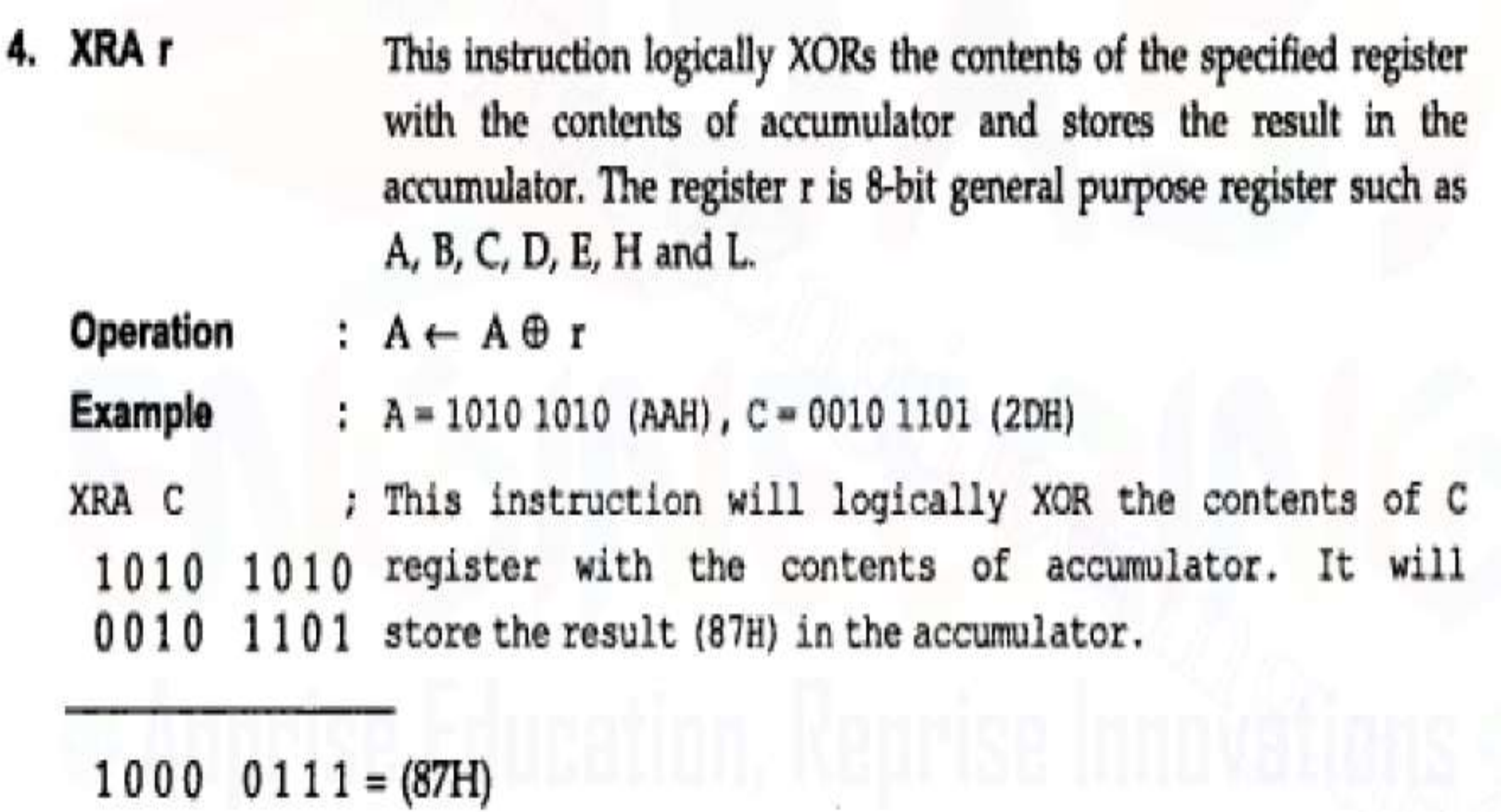


Fig. 3.1 Masking using AND operation

Microprocessor……..



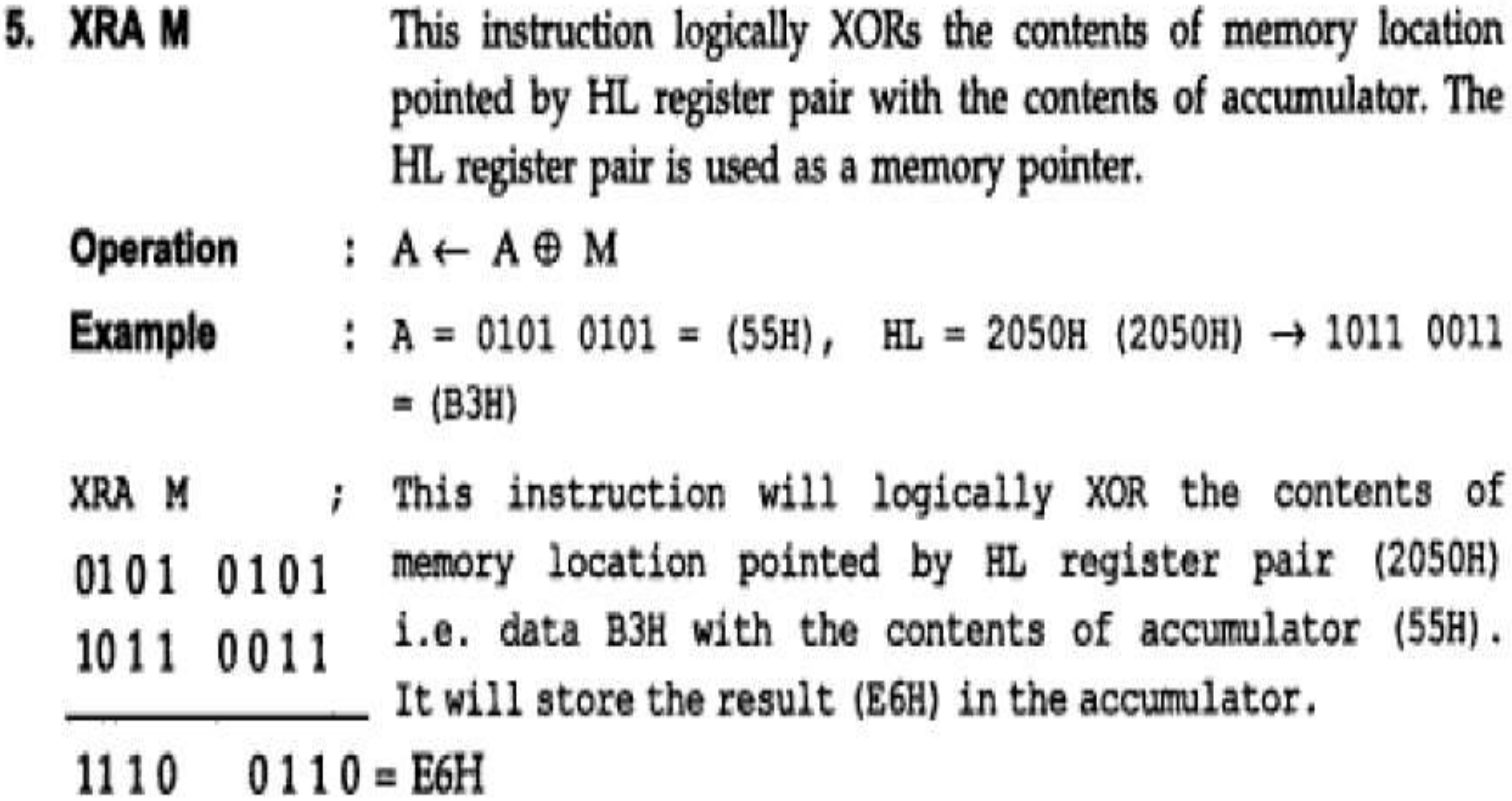
4. XRA r This instruction logically XORs the contents of the specified register with the contents of accumulator and stores the result in the accumulator. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : $A \leftarrow A \oplus r$

Example : A = 1010 1010 (AAH), C = 0010 1101 (2DH)

```
XRA  C         ; This instruction will logically XOR the contents of C
 1010  1010    register with the contents of accumulator. It will
 0010  1101    store the result (87H) in the accumulator.
 ----------
 1000  0111 = (87H)
```

Microprocessor……..

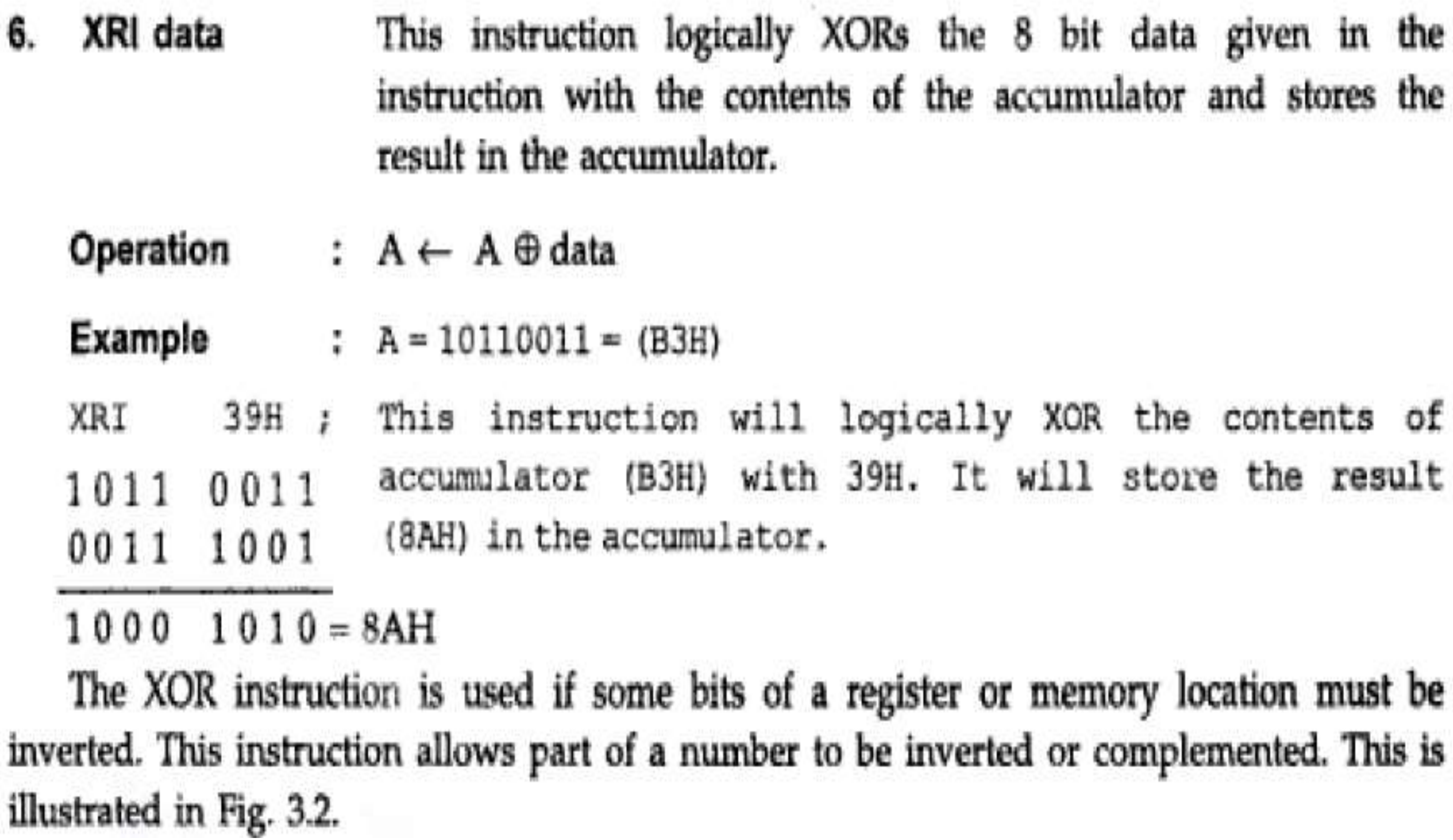


5. XRA M This instruction logically XORs the contents of memory location pointed by HL register pair with the contents of accumulator. The HL register pair is used as a memory pointer.

Operation : A ← A ⊕ M

Example : A = 0101 0101 = (55H), HL = 2050H (2050H) → 1011 0011 = (B3H)

```
XRA M        ; This instruction will logically XOR the contents of
0101  0101     memory location pointed by HL register pair (2050H)
1011  0011     i.e. data B3H with the contents of accumulator (55H).
____________   It will store the result (E6H) in the accumulator.
1110   0110 = E6H
```



6. XRI data This instruction logically XORs the 8 bit data given in the instruction with the contents of the accumulator and stores the result in the accumulator.

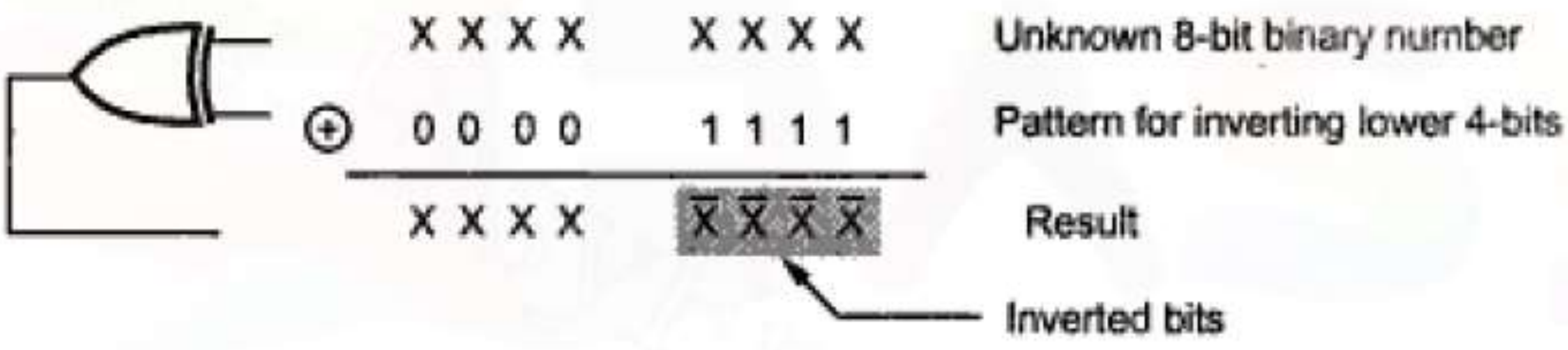
Operation : A ← A ⊕ data

Example : A = 10110011 = (B3H)

XRI 39H ; This instruction will logically XOR the contents of accumulator (B3H) with 39H. It will store the result (8AH) in the accumulator.

```
1011 0011
0011 1001
---------
1000 1010 = 8AH
```

The XOR instruction is used if some bits of a register or memory location must be inverted. This instruction allows part of a number to be inverted or complemented. This is illustrated in Fig. 3.2.



X X X X X X X X
Unknown 8-bit binary number
⊕ 0 0 0 0 1 1 1 1
Pattern for inverting lower 4-bits
X X X X $\overline{X}$ $\overline{X}$ $\overline{X}$ $\overline{X}$
Result
Inverted bits

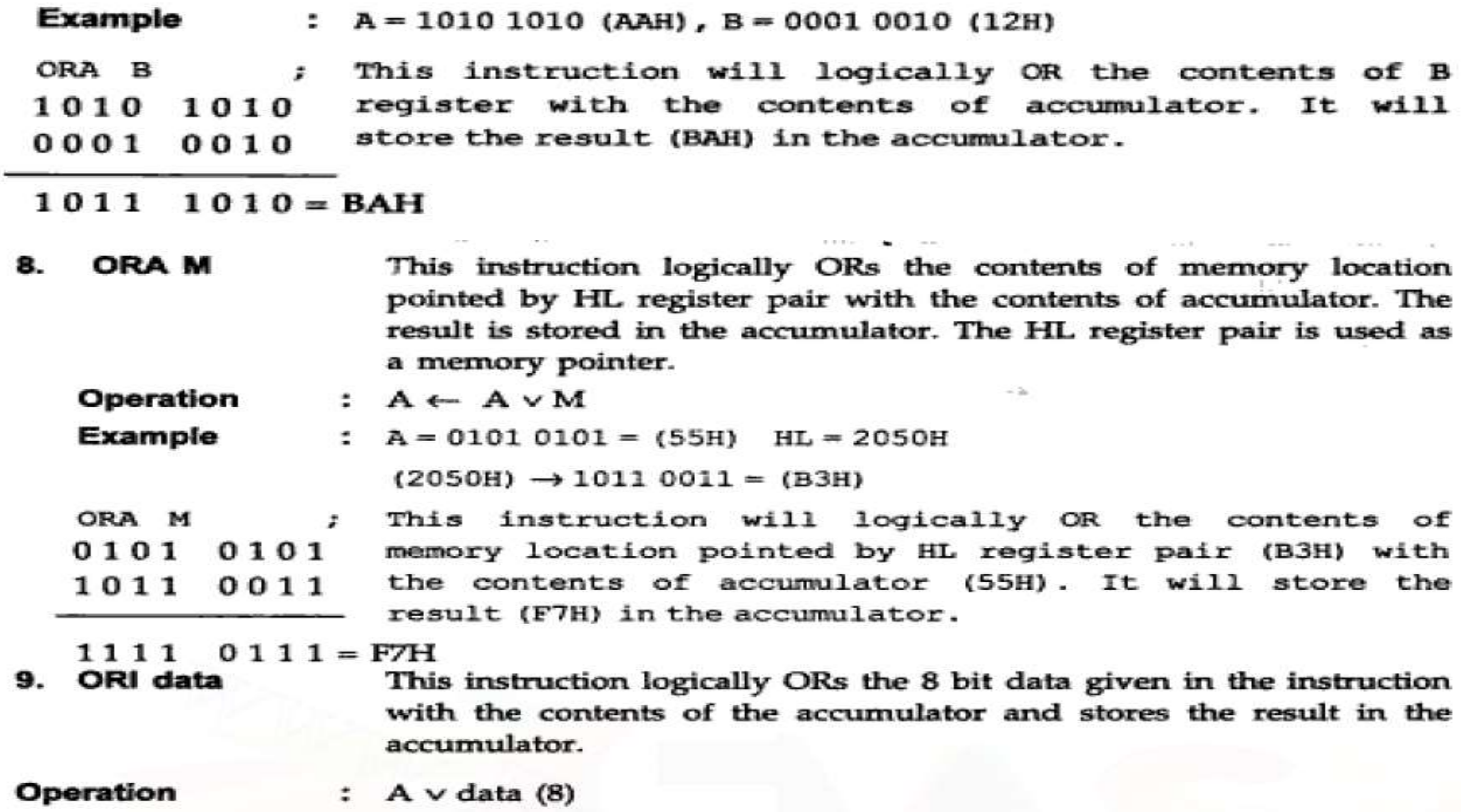
Fig. 3.2 Inversion of part of a number using XOR operation

7. ORA r

This instruction logically ORs the contents of specified register with the contents of accumulator and stores the result in the accumulator. Each bit in the accumulator is ORed with corresponding bit in register r. i.e. D_0 bit in accumulator is ORed with D_0 bit in register r, D_1 in A with D_1 in r and so on upto D_7 bit. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : $A \leftarrow A \vee r$

Microprocessor……..



Example : A = 1010 1010 (AAH), B = 0001 0010 (12H)

```
ORA  B        ;  This instruction will logically OR the contents of B
1010  1010       register with the contents of accumulator. It will
0001  0010       store the result (BAH) in the accumulator.
-----------
1011  1010 = BAH
```

8. ORA M This instruction logically ORs the contents of memory location pointed by HL register pair with the contents of accumulator. The result is stored in the accumulator. The HL register pair is used as a memory pointer.

Operation : A ← A ∨ M

Example : A = 0101 0101 = (55H) HL = 2050H

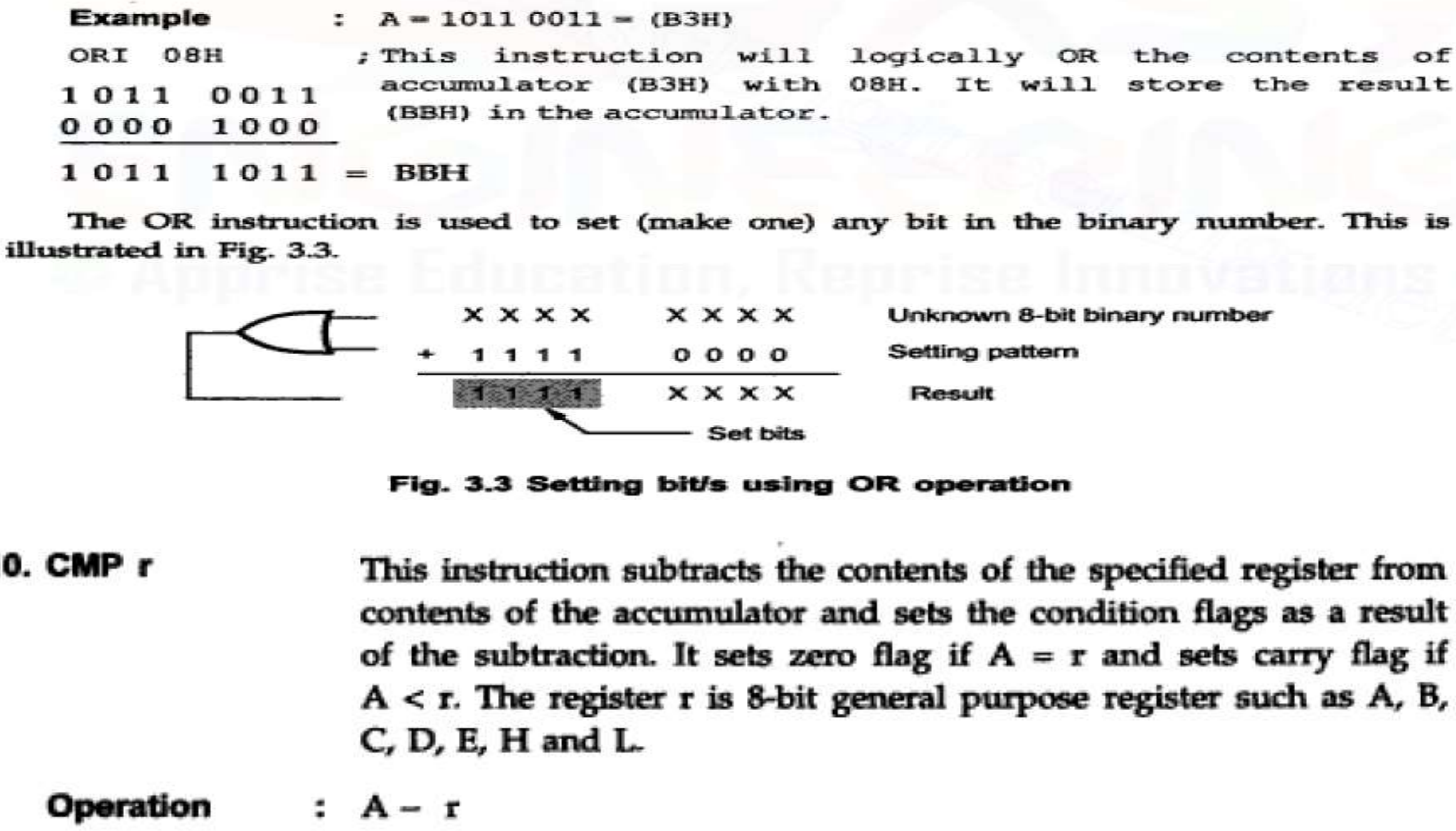
(2050H) → 1011 0011 = (B3H)

```
ORA  M        ;  This instruction will logically OR the contents of
0101  0101       memory location pointed by HL register pair (B3H) with
1011  0011       the contents of accumulator (55H). It will store the
-----------      result (F7H) in the accumulator.
1111  0111 = F7H
```

9. ORI data This instruction logically ORs the 8 bit data given in the instruction with the contents of the accumulator and stores the result in the accumulator.

Operation : A ∨ data (8)

Microprocessor……..



Example : A = 1011 0011 = (B3H)

```
ORI  08H          ;This instruction will logically OR the contents of
1011  0011         accumulator (B3H) with 08H. It will store the result
0000  1000         (BBH) in the accumulator.
-----------
1011  1011  =  BBH
```

The OR instruction is used to set (make one) any bit in the binary number. This is illustrated in Fig. 3.3.

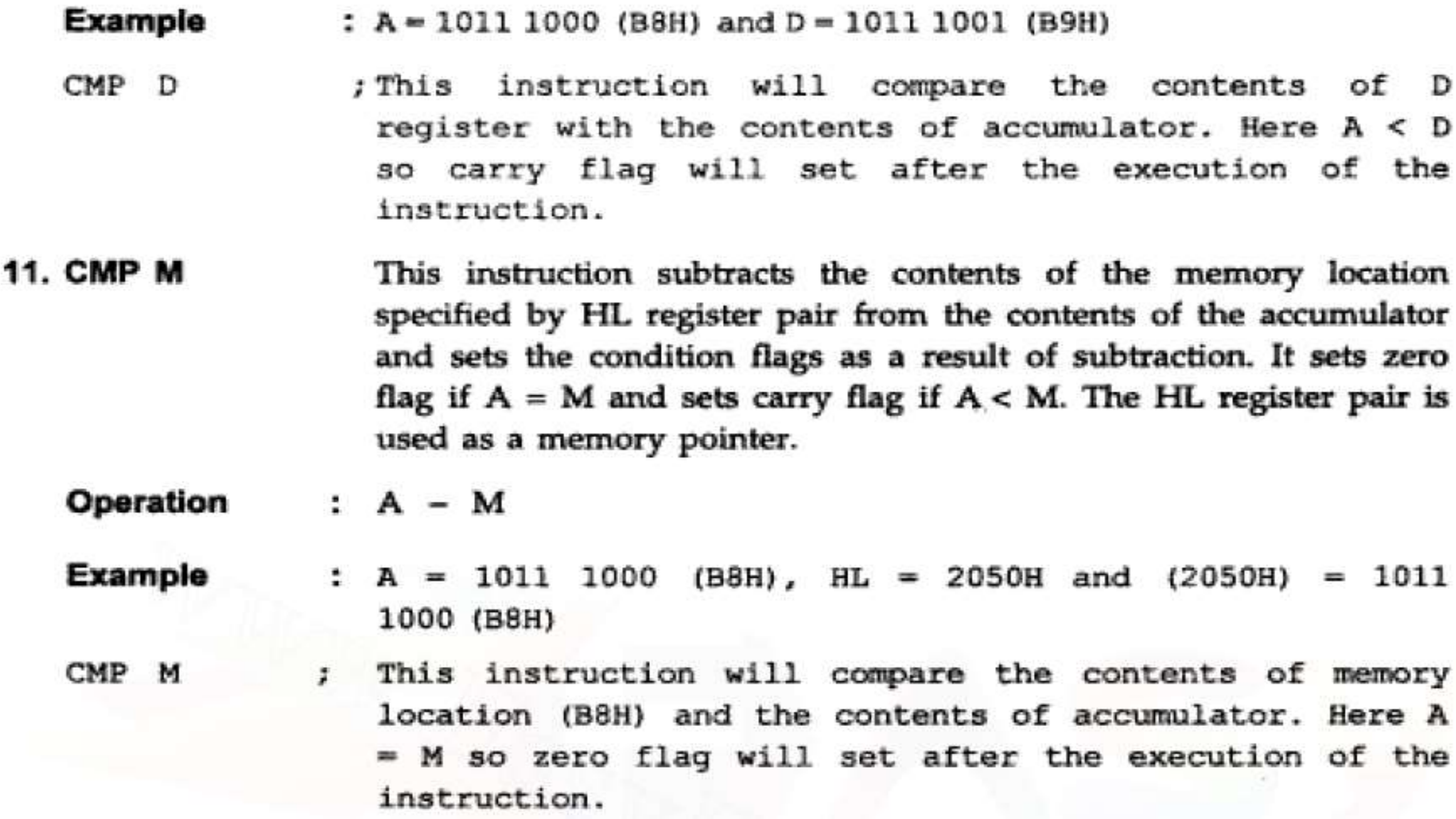
X X X X X X X X
Unknown 8-bit binary number
+ 1 1 1 1 0 0 0 0
Setting pattern
1 1 1 1 X X X X
Result
Set bits

Fig. 3.3 Setting bit/s using OR operation

10. CMP r This instruction subtracts the contents of the specified register from contents of the accumulator and sets the condition flags as a result of the subtraction. It sets zero flag if A = r and sets carry flag if A < r. The register r is 8-bit general purpose register such as A, B, C, D, E, H and L.

Operation : A – r

Microprocessor……..



Example : A = 1011 1000 (B8H) and D = 1011 1001 (B9H)

CMP D ; This instruction will compare the contents of D register with the contents of accumulator. Here A < D so carry flag will set after the execution of the instruction.

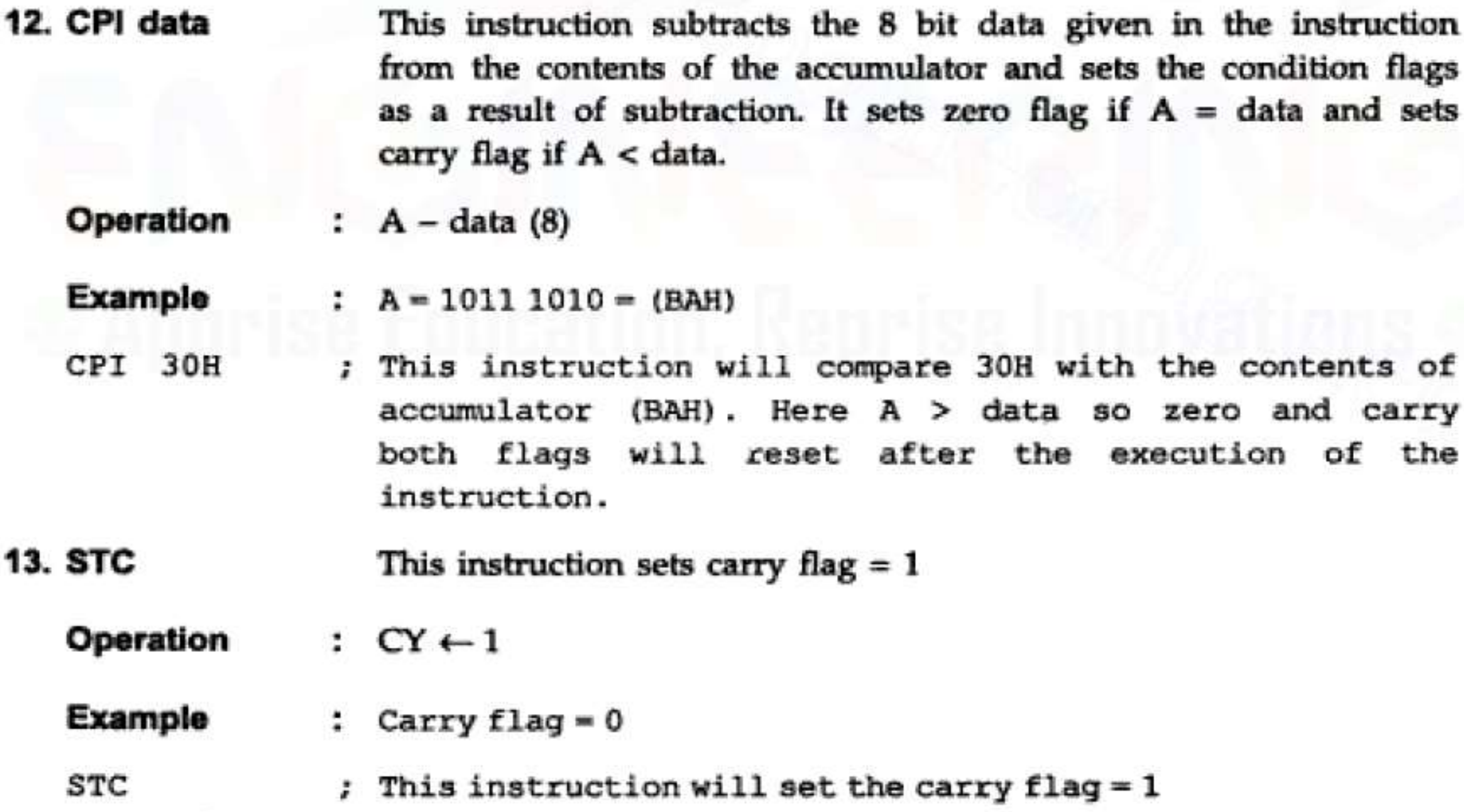
11. CMP M This instruction subtracts the contents of the memory location specified by HL register pair from the contents of the accumulator and sets the condition flags as a result of subtraction. It sets zero flag if A = M and sets carry flag if A < M. The HL register pair is used as a memory pointer.

Operation : A – M

Example : A = 1011 1000 (B8H), HL = 2050H and (2050H) = 1011 1000 (B8H)

CMP M ; This instruction will compare the contents of memory location (B8H) and the contents of accumulator. Here A = M so zero flag will set after the execution of the instruction.

Microprocessor……..



12. CPI data This instruction subtracts the 8 bit data given in the instruction from the contents of the accumulator and sets the condition flags as a result of subtraction. It sets zero flag if A = data and sets carry flag if A < data.

Operation : A – data (8)

Example : A = 1011 1010 = (BAH)

CPI 30H ; This instruction will compare 30H with the contents of accumulator (BAH). Here A > data so zero and carry both flags will reset after the execution of the instruction.

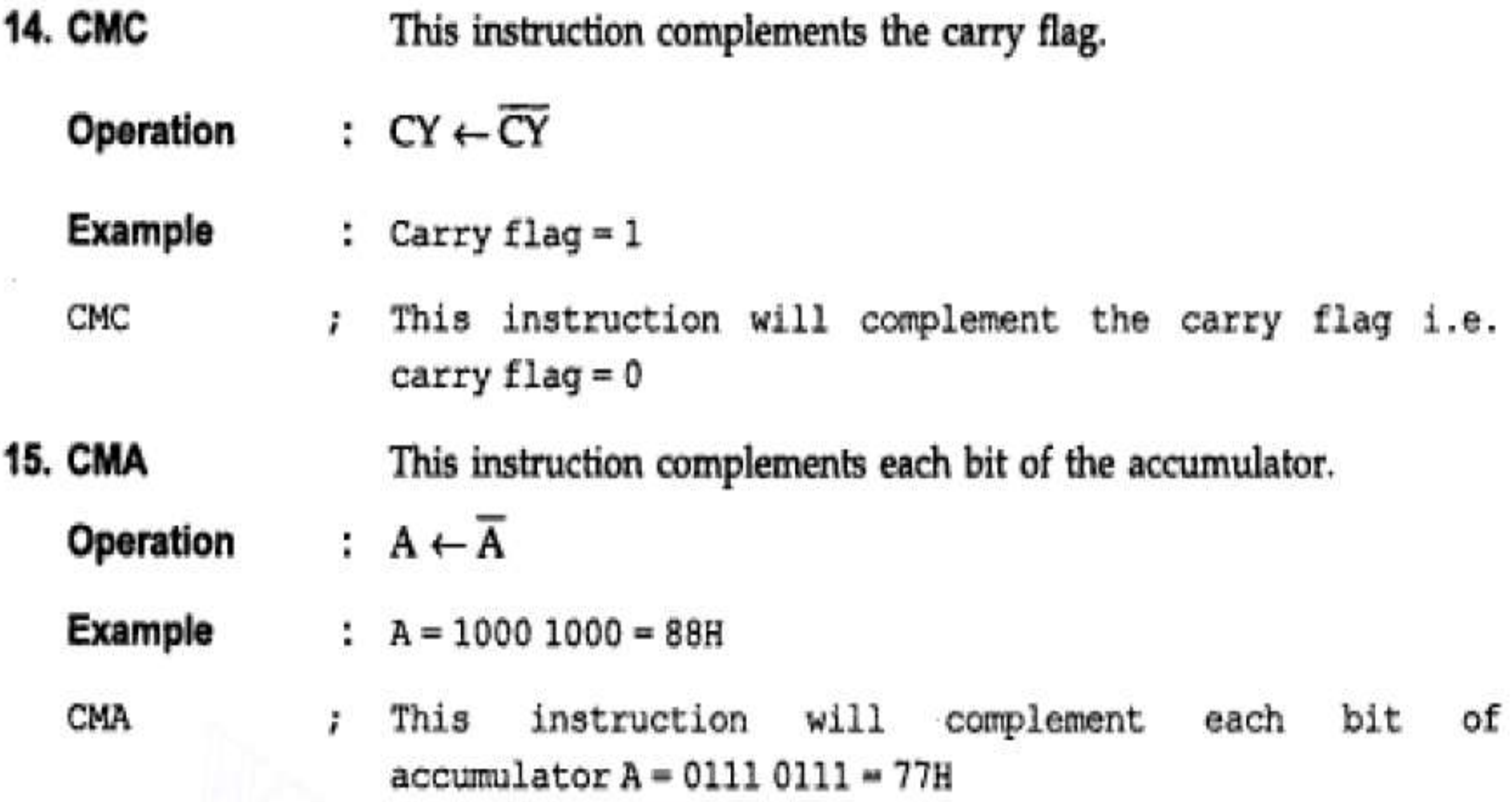
13. STC This instruction sets carry flag = 1

Operation : CY ← 1

Example : Carry flag = 0

STC ; This instruction will set the carry flag = 1

Microprocessor……..



14. CMC This instruction complements the carry flag.

Operation : $CY \leftarrow \overline{CY}$

Example : Carry flag = 1

CMC ; This instruction will complement the carry flag i.e. carry flag = 0

15. CMA This instruction complements each bit of the accumulator.

Operation : $A \leftarrow \overline{A}$

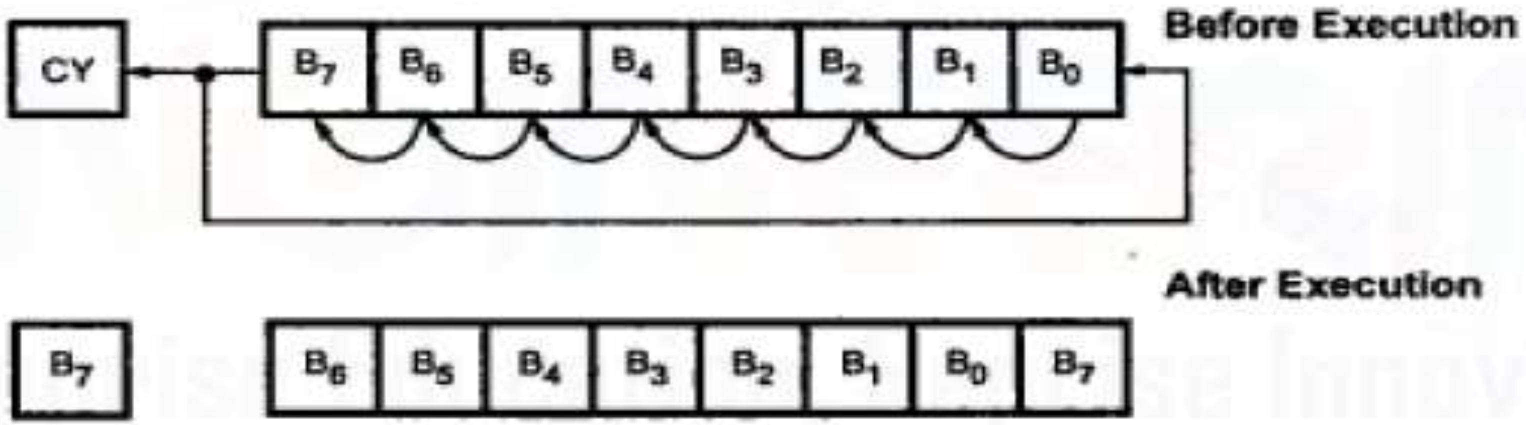
Example : A = 1000 1000 = 88H

CMA ; This instruction will complement each bit of accumulator A = 0111 0111 = 77H

3.4.4 Rotate Group

1. RLC This instruction rotates the contents of the accumulator left by one position. Bit B_7 is placed in B_0 as well as in CY.

Operation :


Before Execution
CY
B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0
After Execution
B_7
B_6 B_5 B_4 B_3 B_2 B_1 B_0 B_7

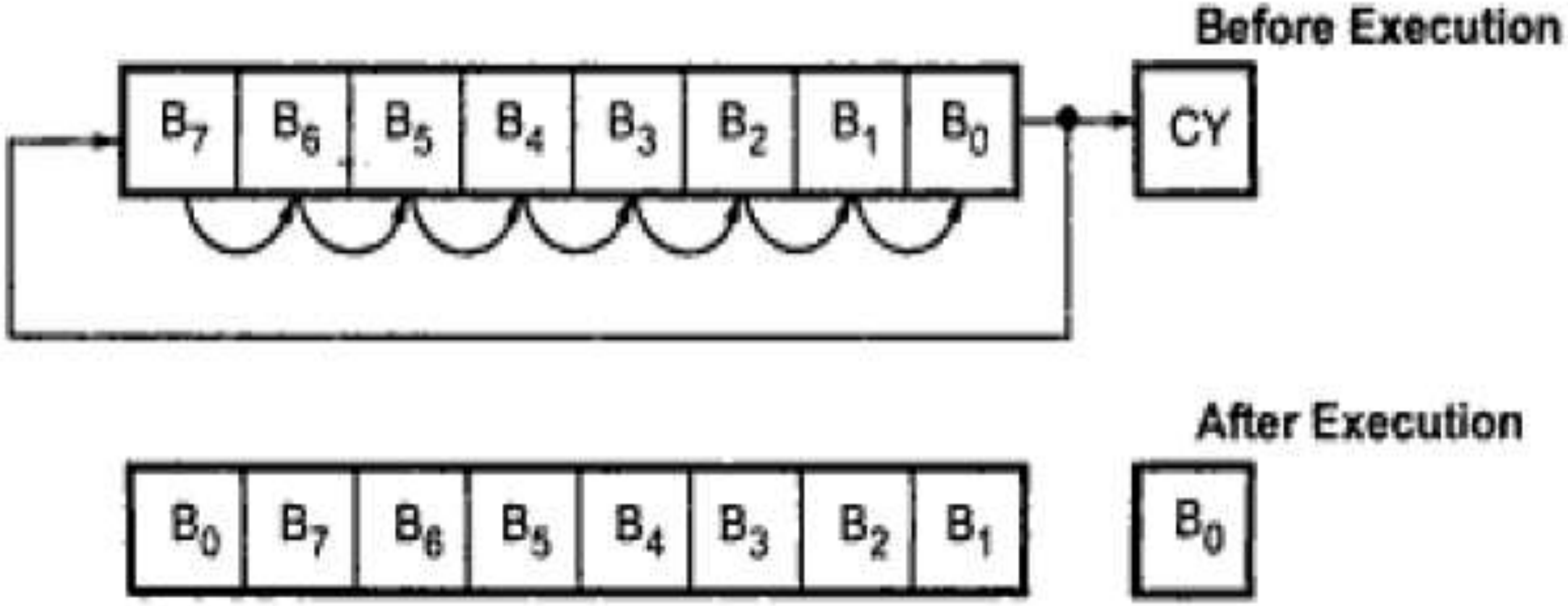
Example : A = 01010111 (57H) and CY = 1

RLC ; After execution of the instruction the accumulator contents will be (1010 1110) AEH and carry flag will reset.

2. RRC This instruction rotates the contents of the accumulator right by one position. Bit B_0 is placed in B_7 as well as in CY.

Microprocessor……..

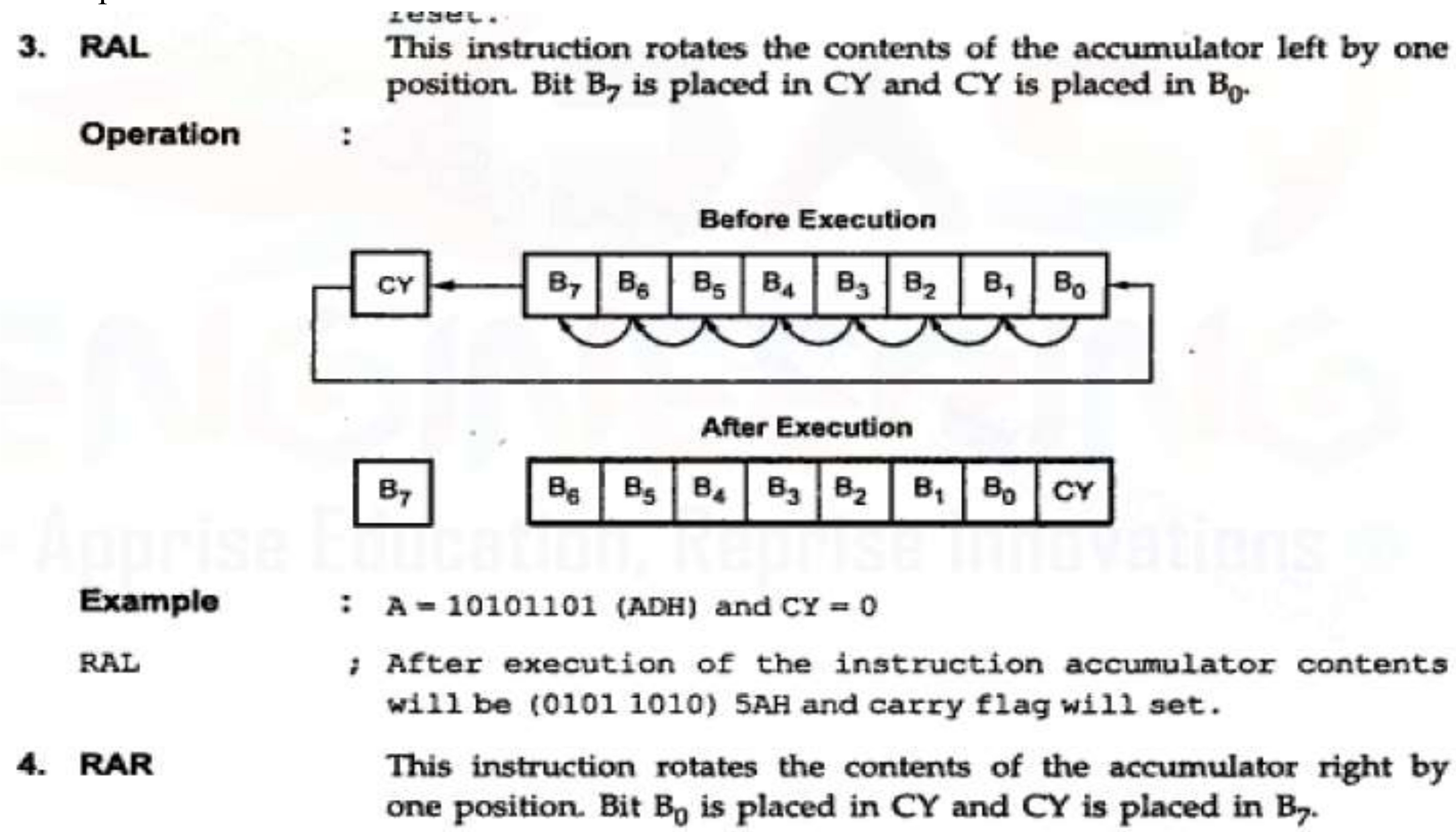
Operation :


Before Execution
B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0
CY
After Execution
B_0 B_7 B_6 B_5 B_4 B_3 B_2 B_1
B_0

Example : A = 1001 1010 (9AH) and CY = 1

RRC ; After execution of the instruction the accumulator contents will be (0100 1101) 4DH and carry flag will reset.

Microprocessor……..



reset.

3. **RAL** This instruction rotates the contents of the accumulator left by one position. Bit B_7 is placed in CY and CY is placed in B_0.

Operation :

Before Execution
CY
B7 B6 B5 B4 B3 B2 B1 B0
After Execution
B7
B6 B5 B4 B3 B2 B1 B0 CY

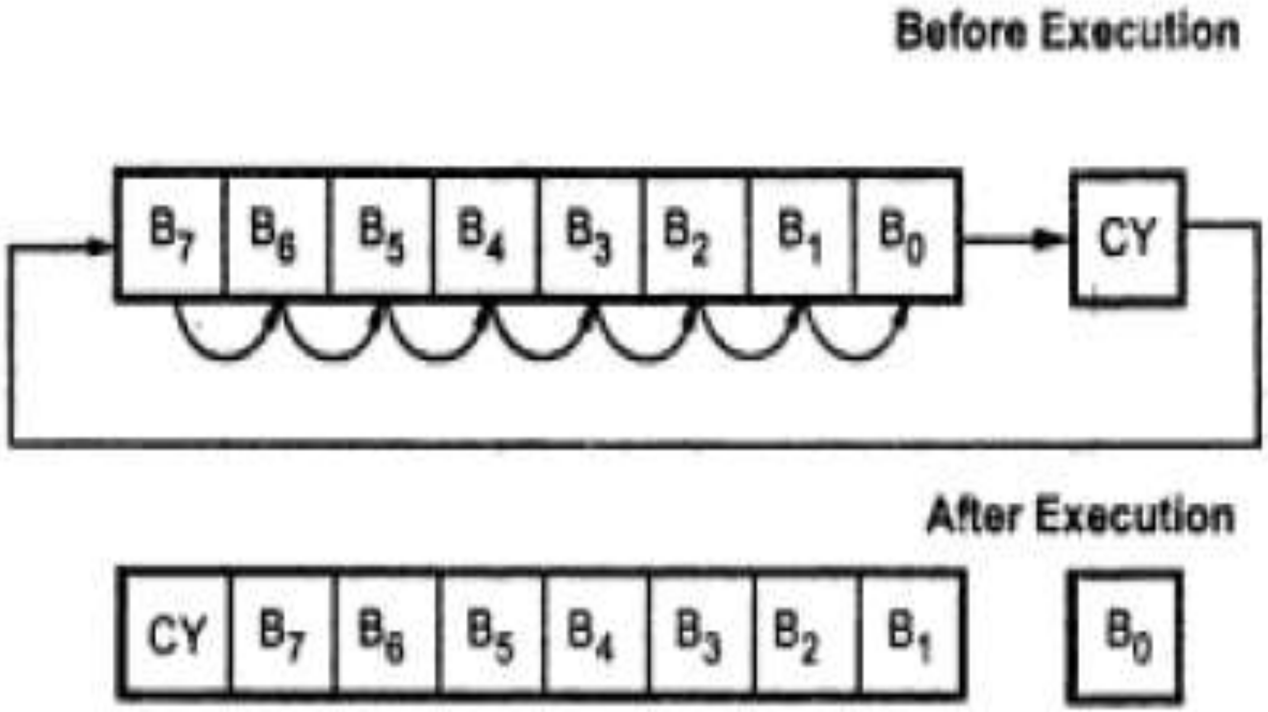
Example : A = 10101101 (ADH) and CY = 0

RAL ; After execution of the instruction accumulator contents will be (0101 1010) 5AH and carry flag will set.

4. **RAR** This instruction rotates the contents of the accumulator right by one position. Bit B_0 is placed in CY and CY is placed in B_7.

Microprocessor……..

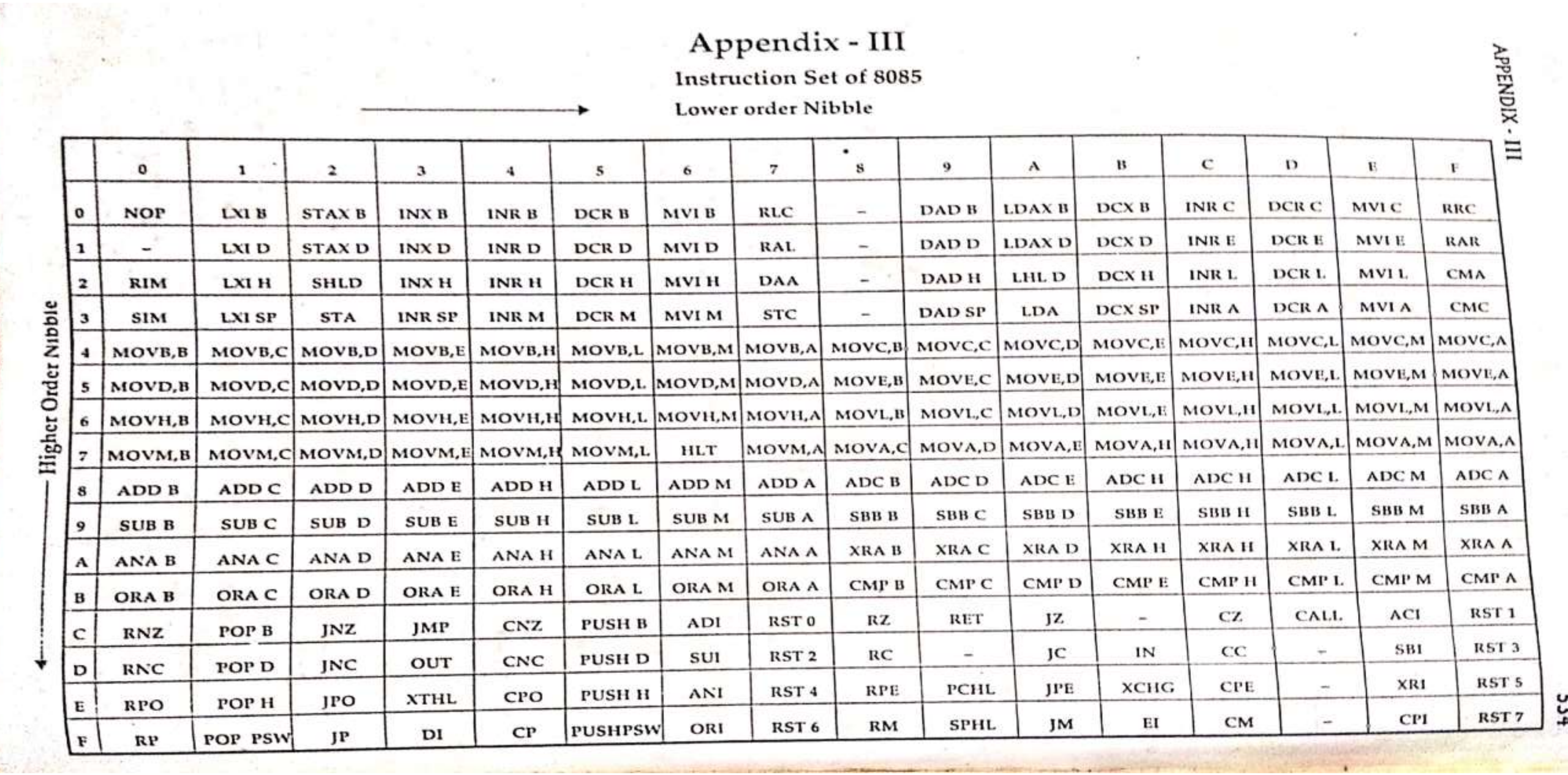
Operation :



Example : A = 1010 0011 (A3H) and CY = 0

RAR ; After execution of the instruction accumulator contents will be (0101 0001) 51H and carry flag will set.

Microprocessor……..



Appendix - III

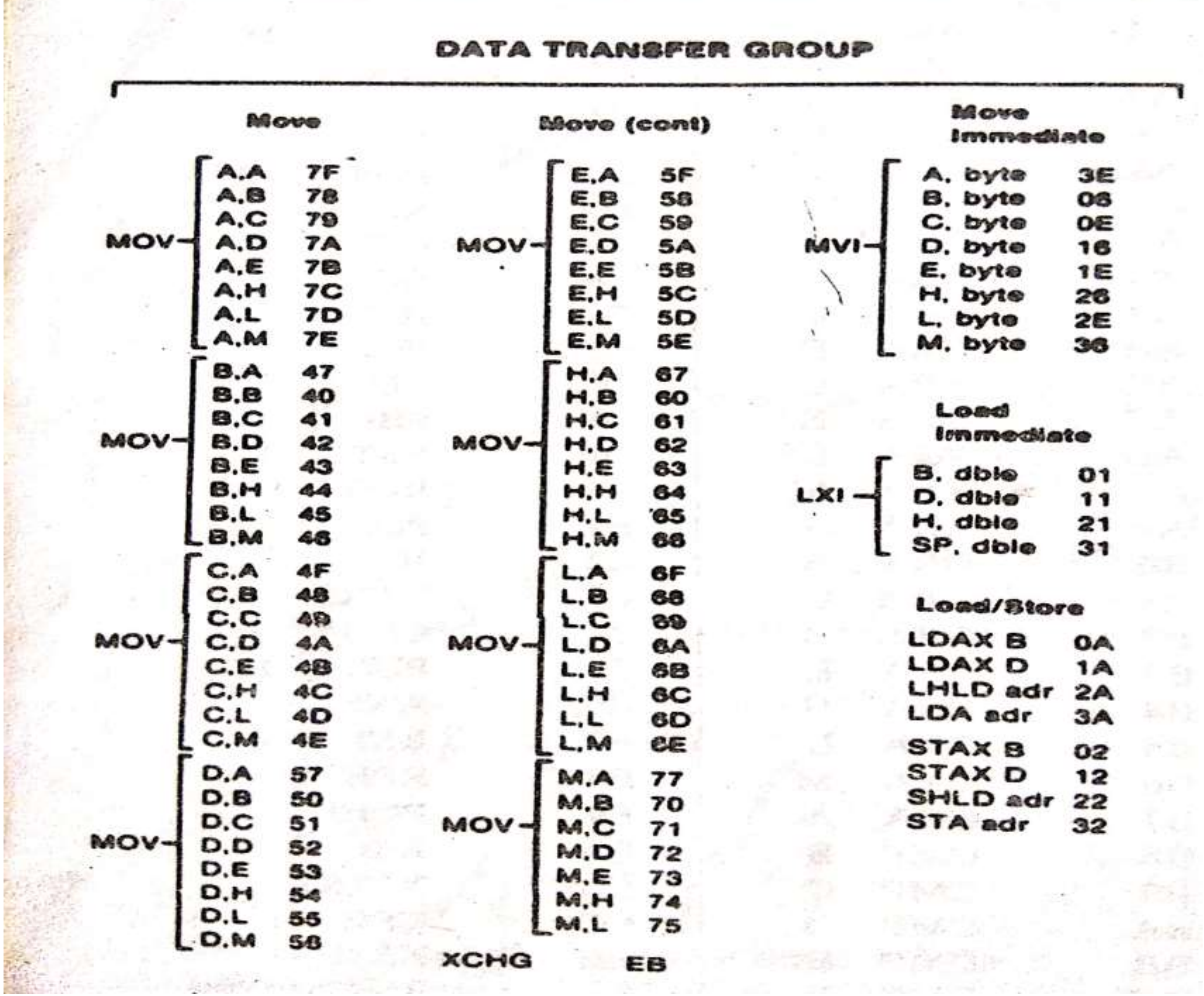
Instruction Set of 8085

→ Lower order Nibble

Higher Order Nibble ↓

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	LXI B	STAX B	INX B	INR B	DCR B	MVI B	RLC	–	DAD B	LDAX B	DCX B	INR C	DCR C	MVI C	RRC
1	–	LXI D	STAX D	INX D	INR D	DCR D	MVI D	RAL	–	DAD D	LDAX D	DCX D	INR E	DCR E	MVI E	RAR
2	RIM	LXI H	SHLD	INX H	INR H	DCR H	MVI H	DAA	–	DAD H	LHL D	DCX H	INR L	DCR L	MVI L	CMA
3	SIM	LXI SP	STA	INR SP	INR M	DCR M	MVI M	STC	–	DAD SP	LDA	DCX SP	INR A	DCR A	MVI A	CMC
4	MOVB,B	MOVB,C	MOVB,D	MOVB,E	MOVB,H	MOVB,L	MOVB,M	MOVB,A	MOVC,B	MOVC,C	MOVC,D	MOVC,E	MOVC,H	MOVC,L	MOVC,M	MOVC,A
5	MOVD,B	MOVD,C	MOVD,D	MOVD,E	MOVD,H	MOVD,L	MOVD,M	MOVD,A	MOVE,B	MOVE,C	MOVE,D	MOVE,E	MOVE,H	MOVE,L	MOVE,M	MOVE,A
6	MOVH,B	MOVH,C	MOVH,D	MOVH,E	MOVH,H	MOVH,L	MOVH,M	MOVH,A	MOVL,B	MOVL,C	MOVL,D	MOVL,E	MOVL,H	MOVL,L	MOVL,M	MOVL,A
7	MOVM,B	MOVM,C	MOVM,D	MOVM,E	MOVM,H	MOVM,L	HLT	MOVM,A	MOVA,C	MOVA,D	MOVA,E	MOVA,H	MOVA,H	MOVA,L	MOVA,M	MOVA,A
8	ADD B	ADD C	ADD D	ADD E	ADD H	ADD L	ADD M	ADD A	ADC B	ADC D	ADC E	ADC H	ADC H	ADC L	ADC M	ADC A
9	SUB B	SUB C	SUB D	SUB E	SUB H	SUB L	SUB M	SUB A	SBB B	SBB C	SBB D	SBB E	SBB H	SBB L	SBB M	SBB A
A	ANA B	ANA C	ANA D	ANA E	ANA H	ANA L	ANA M	ANA A	XRA B	XRA C	XRA D	XRA H	XRA H	XRA L	XRA M	XRA A
B	ORA B	ORA C	ORA D	ORA E	ORA H	ORA L	ORA M	ORA A	CMP B	CMP C	CMP D	CMP E	CMP H	CMP L	CMP M	CMP A
C	RNZ	POP B	JNZ	JMP	CNZ	PUSH B	ADI	RST 0	RZ	RET	JZ	–	CZ	CALL	ACI	RST 1
D	RNC	POP D	JNC	OUT	CNC	PUSH D	SUI	RST 2	RC	–	JC	IN	CC	–	SBI	RST 3
E	RPO	POP H	JPO	XTHL	CPO	PUSH H	ANI	RST 4	RPE	PCHL	JPE	XCHG	CPE	–	XRI	RST 5
F	RP	POP PSW	JP	DI	CP	PUSHPSW	ORI	RST 6	RM	SPHL	JM	EI	CM	–	CPI	RST 7

Microprocessor…



DATA TRANSFER GROUP

Move

MOV	A,A	7F
	A,B	78
	A,C	79
	A,D	7A
	A,E	7B
	A,H	7C
	A,L	7D
	A,M	7E
MOV	B,A	47
	B,B	40
	B,C	41
	B,D	42
	B,E	43
	B,H	44
	B,L	45
	B,M	46
MOV	C,A	4F
	C,B	48
	C,C	49
	C,D	4A
	C,E	4B
	C,H	4C
	C,L	4D
	C,M	4E
MOV	D,A	57
	D,B	50
	D,C	51
	D,D	52
	D,E	53
	D,H	54
	D,L	55
	D,M	56

Move (cont)

MOV	E,A	5F
	E,B	58
	E,C	59
	E,D	5A
	E,E	5B
	E,H	5C
	E,L	5D
	E,M	5E
MOV	H,A	67
	H,B	60
	H,C	61
	H,D	62
	H,E	63
	H,H	64
	H,L	65
	H,M	66
MOV	L,A	6F
	L,B	68
	L,C	69
	L,D	6A
	L,E	6B
	L,H	6C
	L,L	6D
	L,M	6E
MOV	M,A	77
	M,B	70
	M,C	71
	M,D	72
	M,E	73
	M,H	74
	M,L	75
XCHG		EB

Move Immediate

MVI	A, byte	3E
	B, byte	06
	C, byte	0E
	D, byte	16
	E, byte	1E
	H, byte	26
	L, byte	2E
	M, byte	36

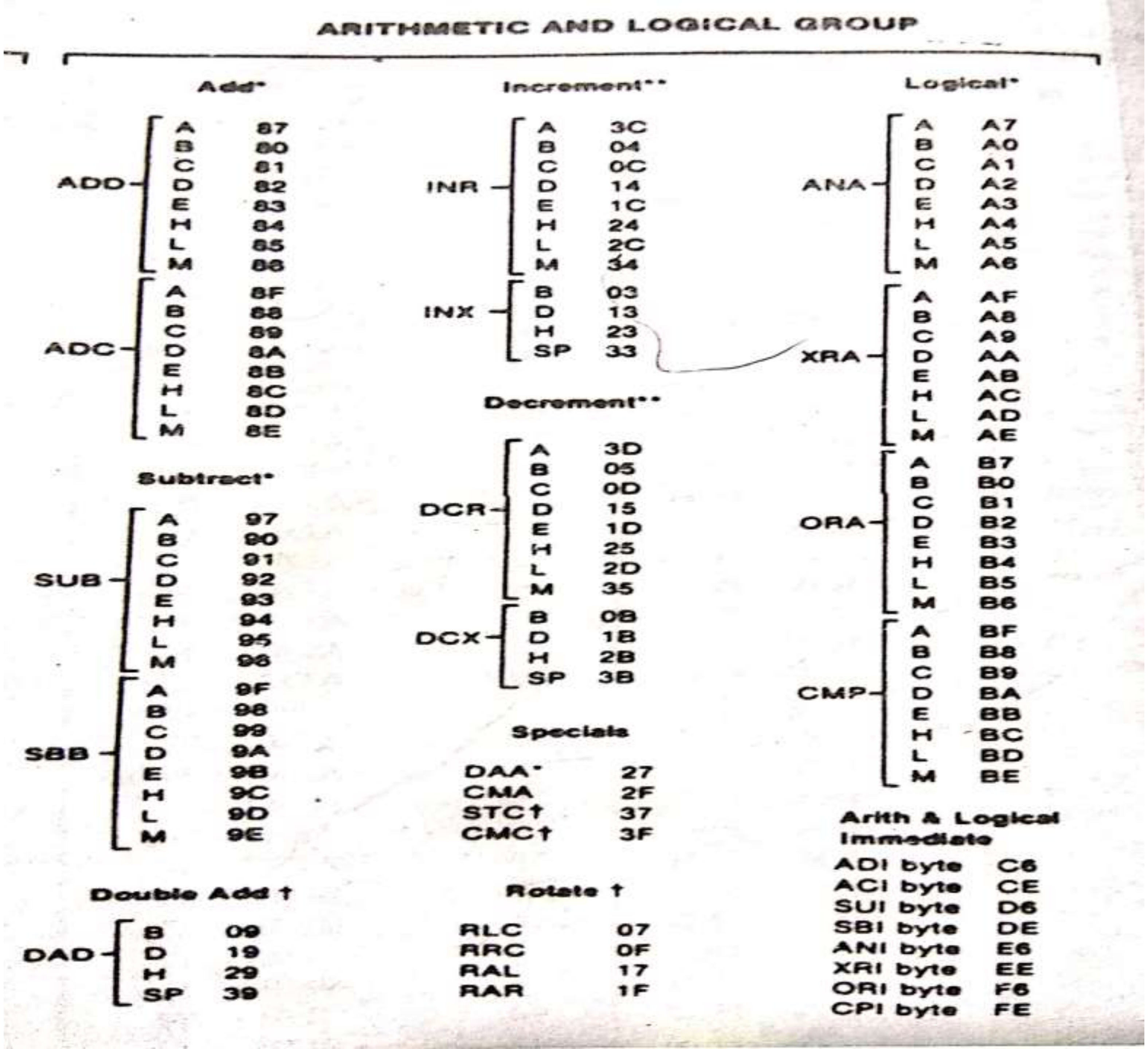
Load Immediate

LXI	B, dble	01
	D, dble	11
	H, dble	21
	SP, dble	31

Load/Store

LDAX B	0A
LDAX D	1A
LHLD adr	2A
LDA adr	3A
STAX B	02
STAX D	12
SHLD adr	22
STA adr	32

Microprocessor……..



ARITHMETIC AND LOGICAL GROUP

Add*

ADD	A	87
	B	80
	C	81
	D	82
	E	83
	H	84
	L	85
	M	86
ADC	A	8F
	B	88
	C	89
	D	8A
	E	8B
	H	8C
	L	8D
	M	8E

Subtrect*

SUB	A	97
	B	90
	C	91
	D	92
	E	93
	H	94
	L	95
	M	96
SBB	A	9F
	B	98
	C	99
	D	9A
	E	9B
	H	9C
	L	9D
	M	9E

Double Add †

DAD	B	09
	D	19
	H	29
	SP	39

Increment**

INR	A	3C
	B	04
	C	0C
	D	14
	E	1C
	H	24
	L	2C
	M	34
INX	B	03
	D	13
	H	23
	SP	33

Decrement**

DCR	A	3D
	B	05
	C	0D
	D	15
	E	1D
	H	25
	L	2D
	M	35
DCX	B	0B
	D	1B
	H	2B
	SP	3B

Specials

DAA*	27
CMA	2F
STC†	37
CMC†	3F

Rotate †

RLC	07
RRC	0F
RAL	17
RAR	1F

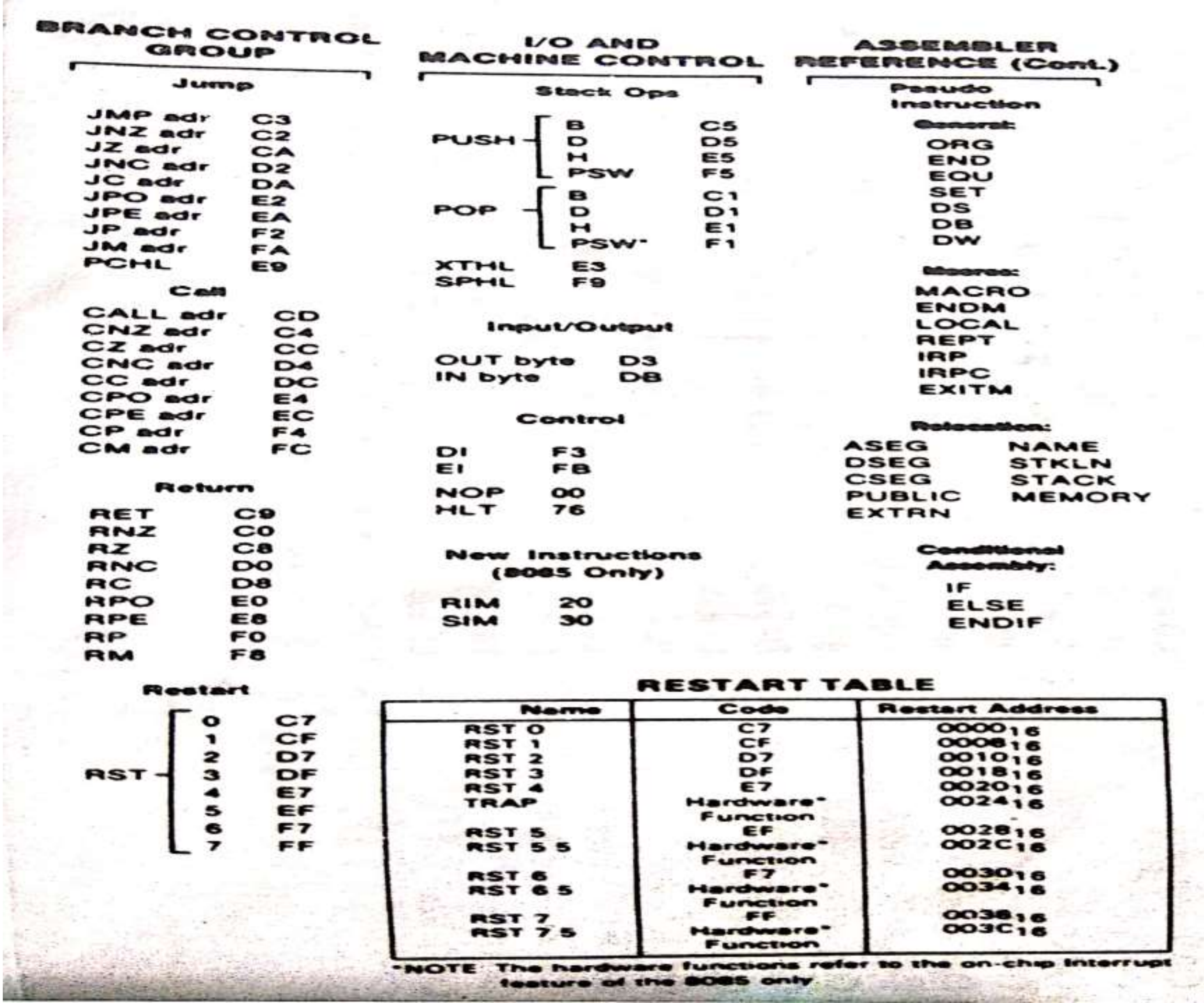
Logical*

ANA	A	A7
	B	A0
	C	A1
	D	A2
	E	A3
	H	A4
	L	A5
	M	A6
XRA	A	AF
	B	A8
	C	A9
	D	AA
	E	AB
	H	AC
	L	AD
	M	AE
ORA	A	B7
	B	B0
	C	B1
	D	B2
	E	B3
	H	B4
	L	B5
	M	B6
CMP	A	BF
	B	B8
	C	B9
	D	BA
	E	BB
	H	BC
	L	BD
	M	BE

Arith & Logical Immediate

ADI byte	C6
ACI byte	CE
SUI byte	D6
SBI byte	DE
ANI byte	E6
XRI byte	EE
ORI byte	F6
CPI byte	FE

Microprocessor…..



BRANCH CONTROL GROUP

Jump

JMP adr	C3
JNZ adr	C2
JZ adr	CA
JNC adr	D2
JC adr	DA
JPO adr	E2
JPE adr	EA
JP adr	F2
JM adr	FA
PCHL	E9

Call

CALL adr	CD
CNZ adr	C4
CZ adr	CC
CNC adr	D4
CC adr	DC
CPO adr	E4
CPE adr	EC
CP adr	F4
CM adr	FC

Return

RET	C9
RNZ	C0
RZ	C8
RNC	D0
RC	D8
RPO	E0
RPE	E8
RP	F0
RM	F8

Restart

RST	0	C7
	1	CF
	2	D7
	3	DF
	4	E7
	5	EF
	6	F7
	7	FF

I/O AND MACHINE CONTROL

Stack Ops

PUSH	B	C5
	D	D5
	H	E5
	PSW	F5
POP	B	C1
	D	D1
	H	E1
	PSW*	F1
XTHL		E3
SPHL		F9

Input/Output

OUT byte	D3
IN byte	DB

Control

DI	F3
EI	FB
NOP	00
HLT	76

New Instructions (8085 Only)

RIM	20
SIM	30

ASSEMBLER REFERENCE (Cont.)

Pseudo Instruction

General:

ORG
END
EQU
SET
DS
DB
DW

Macros:

MACRO
ENDM
LOCAL
REPT
IRP
IRPC
EXITM

Relocation:

ASEG	NAME
DSEG	STKLN
CSEG	STACK
PUBLIC	MEMORY
EXTRN	

Conditional Assembly:

IF
ELSE
ENDIF

RESTART TABLE

Name	Code	Restart Address
RST 0	C7	0000_{16}
RST 1	CF	0008_{16}
RST 2	D7	0010_{16}
RST 3	DF	0018_{16}
RST 4	E7	0020_{16}
TRAP	Hardware* Function	0024_{16}
RST 5	EF	0028_{16}
RST 5 5	Hardware* Function	$002C_{16}$
RST 6	F7	0030_{16}
RST 6 5	Hardware* Function	0034_{16}
RST 7	FF	0038_{16}
RST 7 5	Hardware* Function	$003C_{16}$

*NOTE The hardware functions refer to the on-chip interrupt feature of the 8085 only