MICROPROCESSOR BASED SYSTEMS

CMP 410

LECTURE 4

https://AssemSite8.wix.com/site8

Instructions for the Program memory (external/internal)

Reading individual permeant data-bytes from the program memories

MOVC A, @A+DPTR

Instructions for the external RAM

Reading Data-Bytes from the external RAM

MOVX A , @DPTR — Using 16-bit pointer

MOVX A , @Ri — Using 8-bit pointer

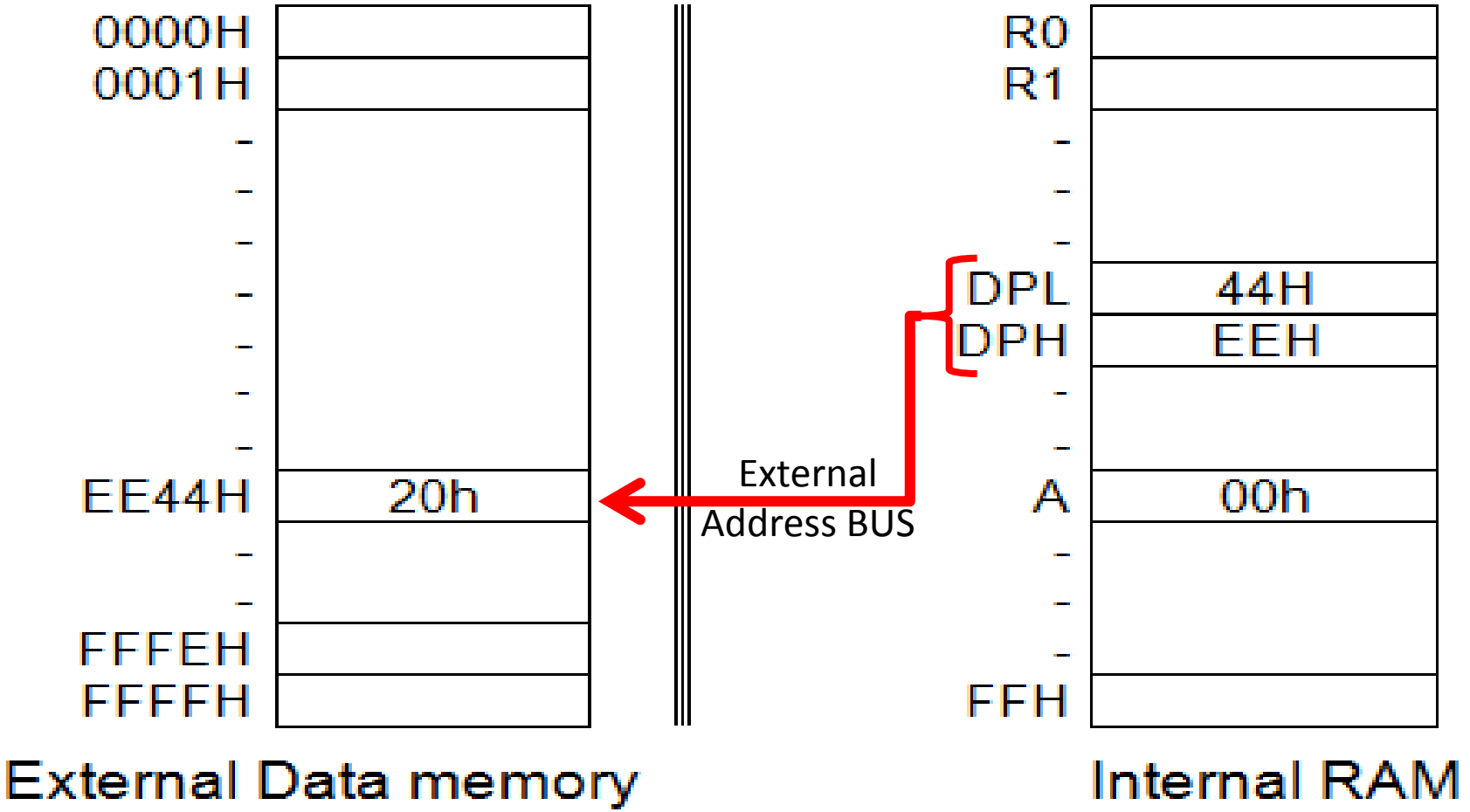
Writing Data-Bytes to the external RAM

MOVX @DPTR , A

MOVX @Ri , A

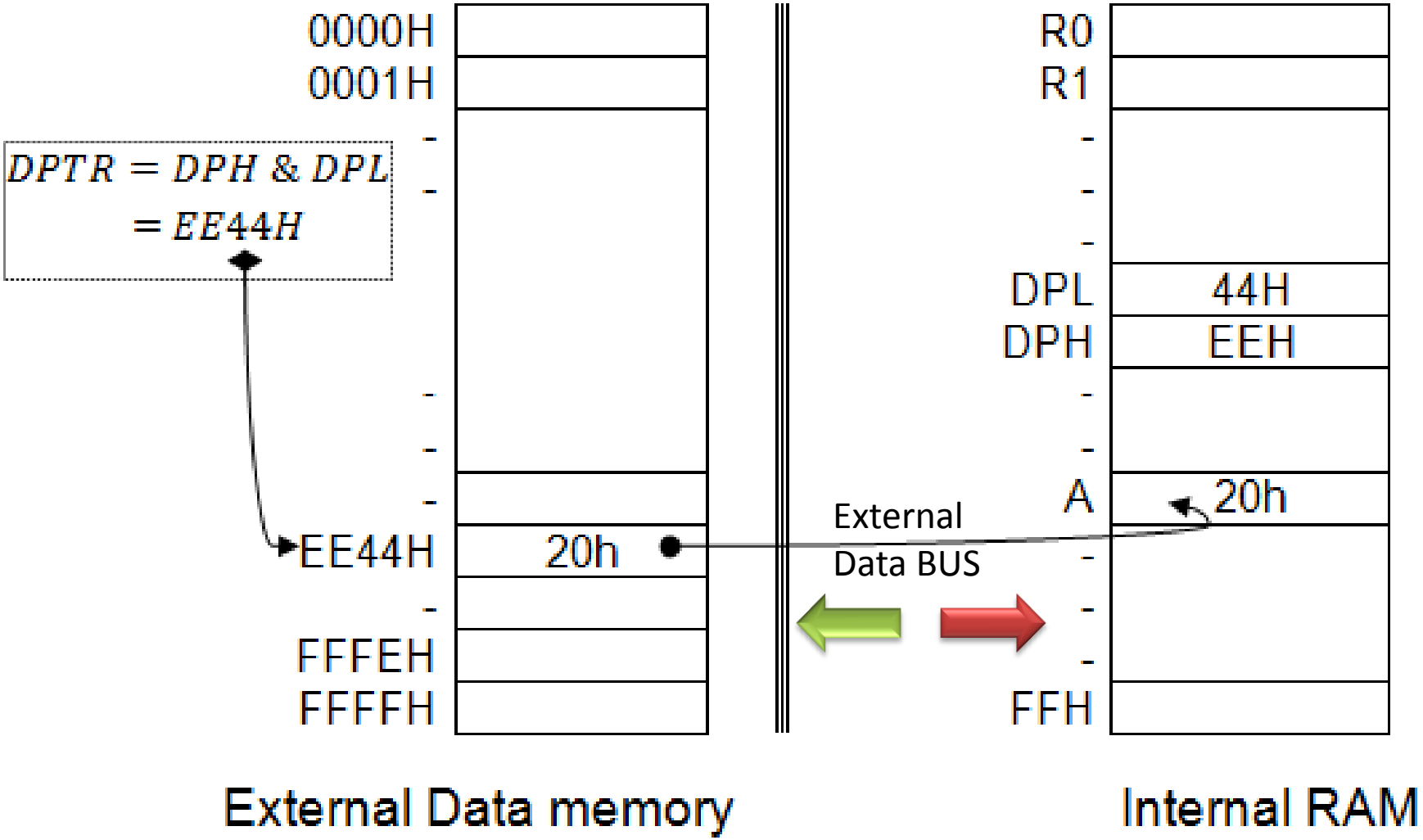
Before executing the instructions

MOVX A, @DPTR or **MOVX @DPTR, A**



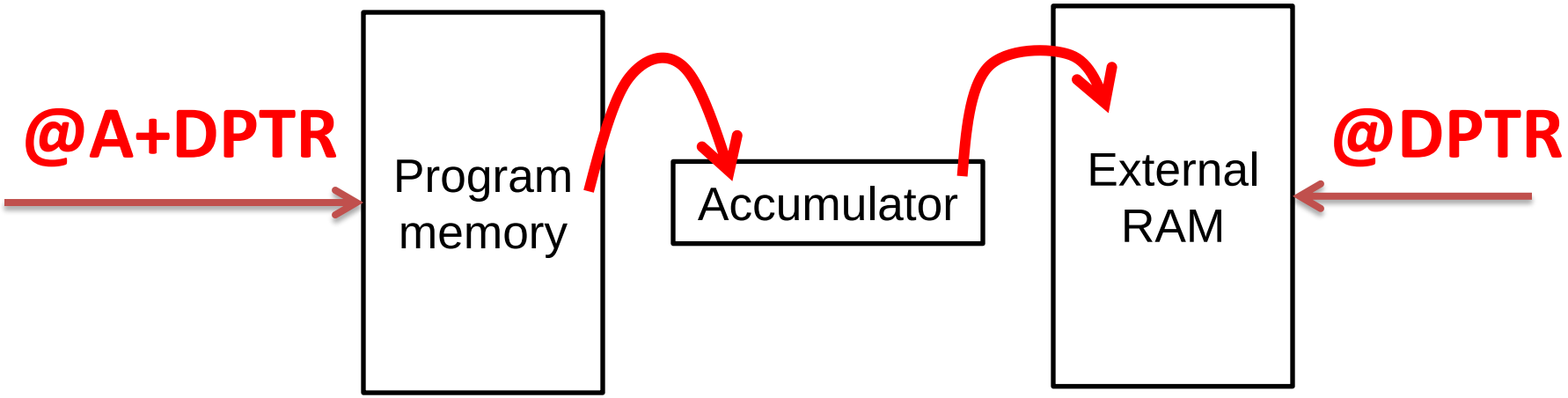
After executing the instructions

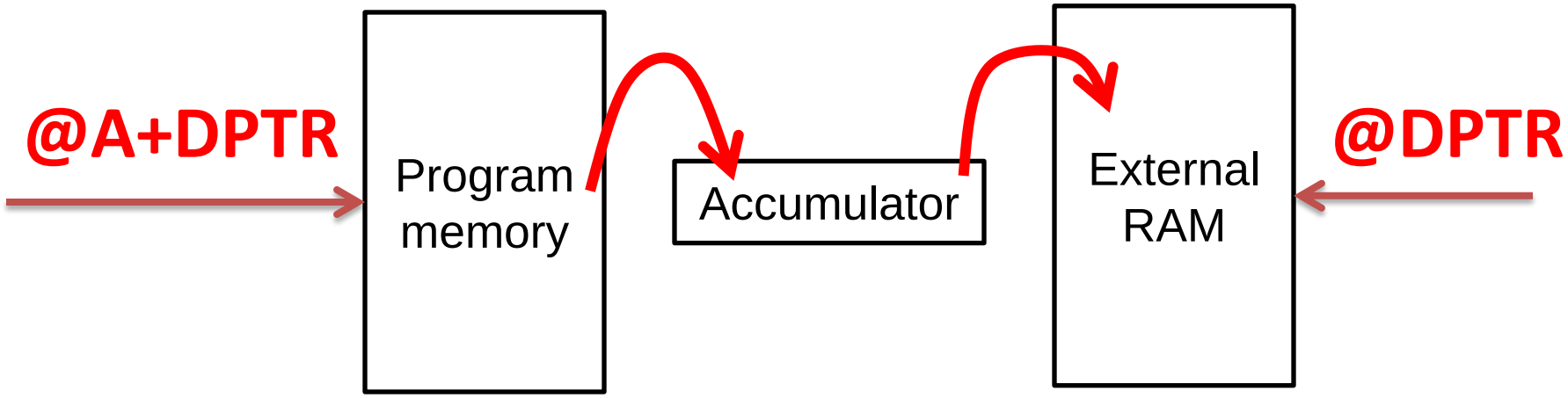
MOVX @DPTR, A **MOVX A, @DPTR**



EXAMPLE

Write assembly sub-program to read data-byte from the program memory at the address **"0111h"** and store it into external RAM location at the address **"0888h"** using the data transfer instructions.






```
INCLUDE 8051.MC
MOV  A,#00H          ;clear A   ; or CLR A
MOV DPTR,#0111H      ;Loading DPTR by Flash address
MOVC A,@A+DPTR       ;Moving byte from Flash to A
MOV DPTR,#0888H      ;Loading DPTR by Ext. RAM address
MOVX @DPTR, A        ; Moving the Byte to Ext. RAM
```

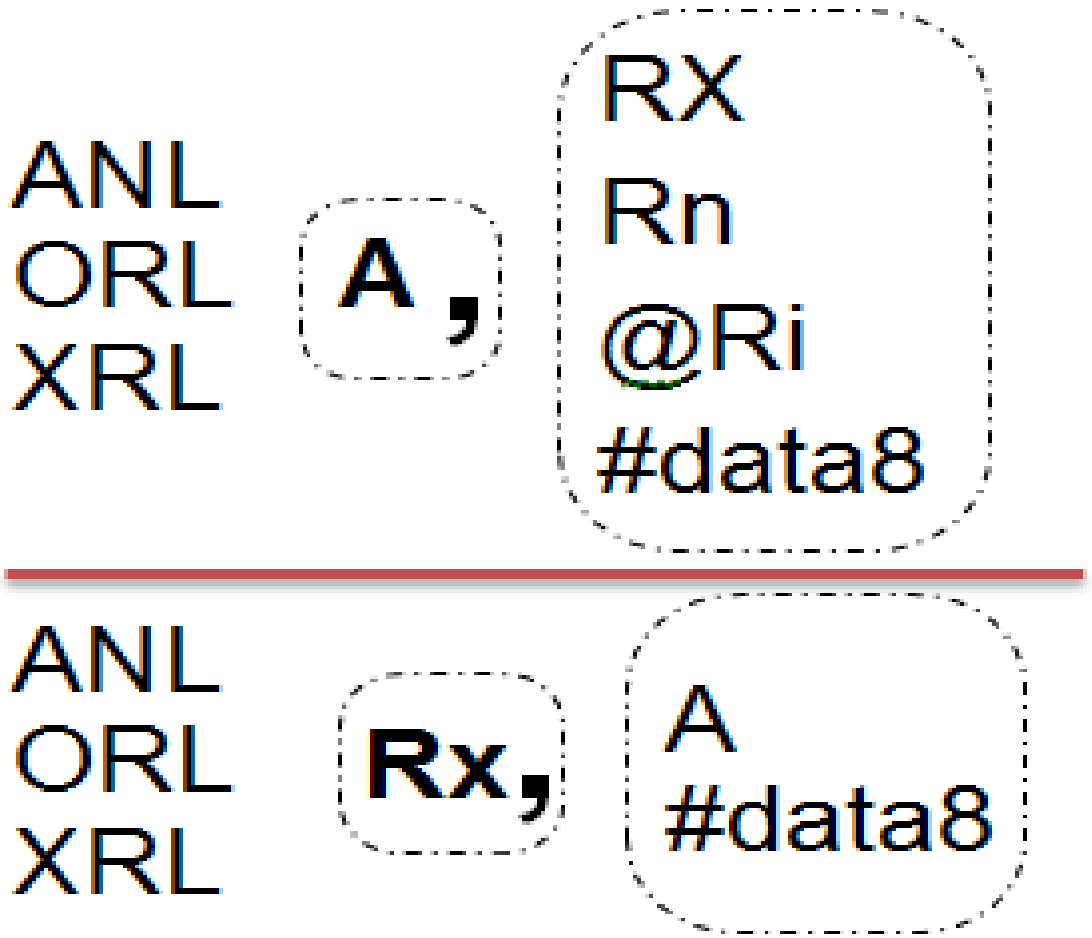
Data Transfer Instructions 2	
Instruction	Description
PUSH Rx	Move data from internal RAM → internal RAM (indirectly[SP])
POP Rx	Move data from internal RAM (indirectly[SP]) → internal RAM
XCH A, Rn	Exchange data A ←→ (R0... R7)
XCH A, Rx	Exchange data A ←→ internal RAM
XCH A, @Ri	Exchange data A ←→ internal RAM (indirectly[Ri])
XCHD A, @Ri	Exchange lower 4-bits data A ←→ internal RAM (indirectly[Ri])

Logical Instructions

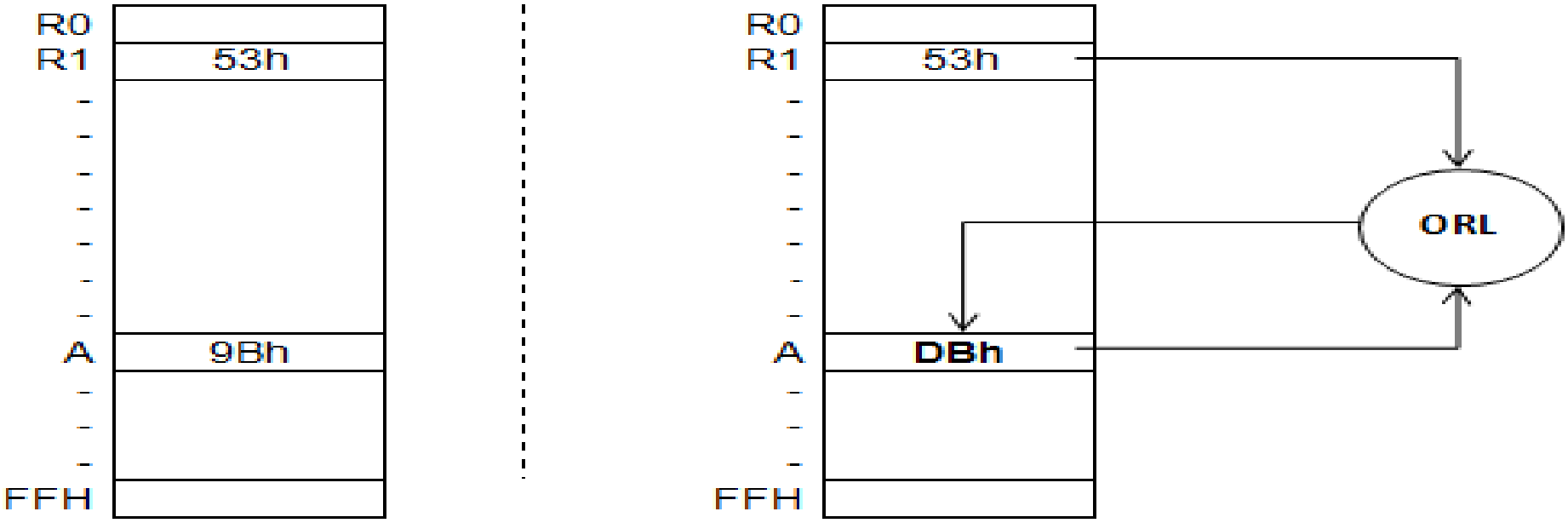
Instruction	Description
ANL A, #data	8-bit data AND A → A
ANL A, @Ri	data from internal RAM (indirectly[Ri]) AND A → A
ANL A, Rx	Data from internal RAM AND A → A
ANL A, Rn	Data from (R0… R7) AND A → A
ANL Rx, #data	8-bit data AND internal RAM → internal RAM
ANL Rx, A	Data from A AND internal RAM → internal RAM
ORL A, #data	8-bit data OR A → A
ORL A, @Ri	data from internal RAM (indirectly[Ri]) OR A → A
ORL A, Rx	Data from internal RAM OR A → A
ORL A, Rn	Data from (R0… R7) OR A → A
ORL Rx, #data	8-bit data OR internal RAM → internal RAM
ORL Rx, A	Data from A OR internal RAM → internal RAM
XRL A, #data	8-bit data AND A → A
XRL A, @Ri	data from internal RAM (indirectly[Ri]) AND A → A
XRL A, Rx	Data from internal RAM AND A → A
XRL A, Rn	Data from (R0… R7) AND A → A
XRL Rx, #data	8-bit data AND internal RAM → internal RAM
XRL Rx, A	Data from A AND internal RAM → internal RAM
CLR A	Clear accumulator
CPL A	Complement accumulator
SWAP A	Exchange the 2 nipples of accumulator
RL A	Rotate left for each bit of A and A (MSB) → A (LSB)
RLC A	Rotate left for each bit of A through Carry flag and A (MSB) →C→ A (LSB)
RR A	Rotate right for each bit of A and A (LSB) → A (MSB)
RRC A	Rotate right for each bit of A through Carry flag and A (LSB) →C→ A (MSB)

The logical-operation instructions

2 operands



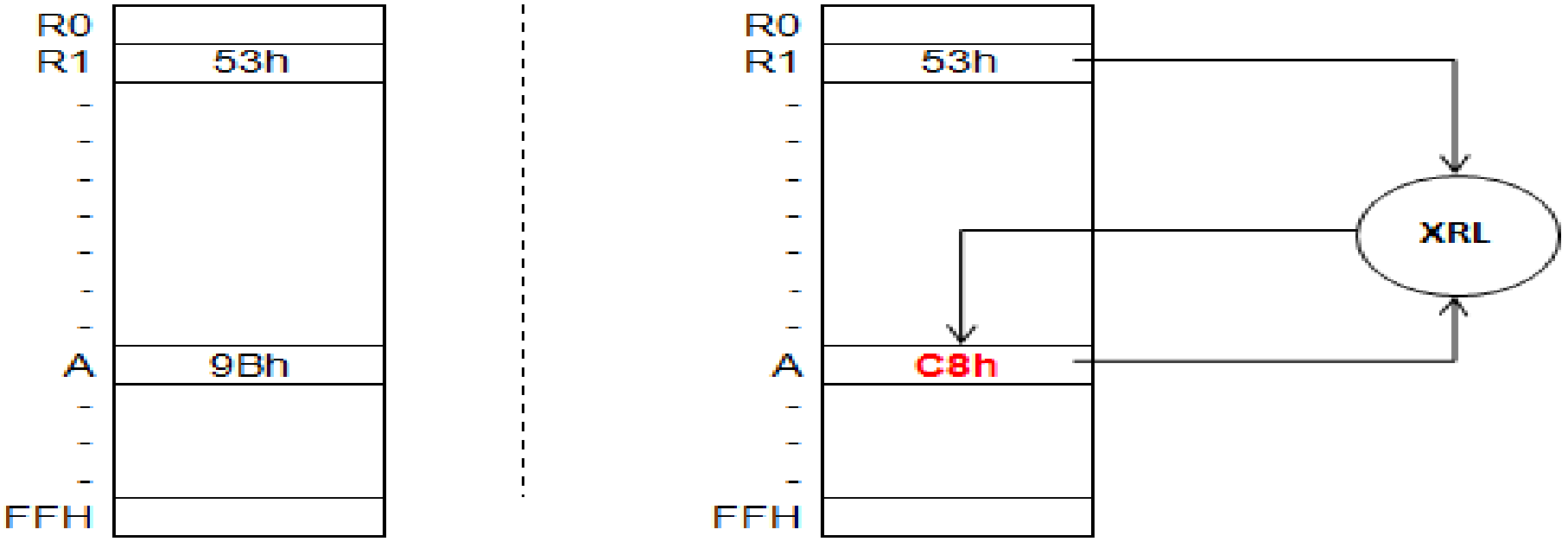
Executing the "ORL" instruction



Before and after the execution of the instruction "**ORL A, R1**"

		Binary								HEX
A		1	0	0	1	1	0	1	1	9Bh
	OR									
R1		0	1	0	1	0	0	1	1	53h
A		1	1	0	1	1	0	1	1	DBh

Executing the "XRL" instruction

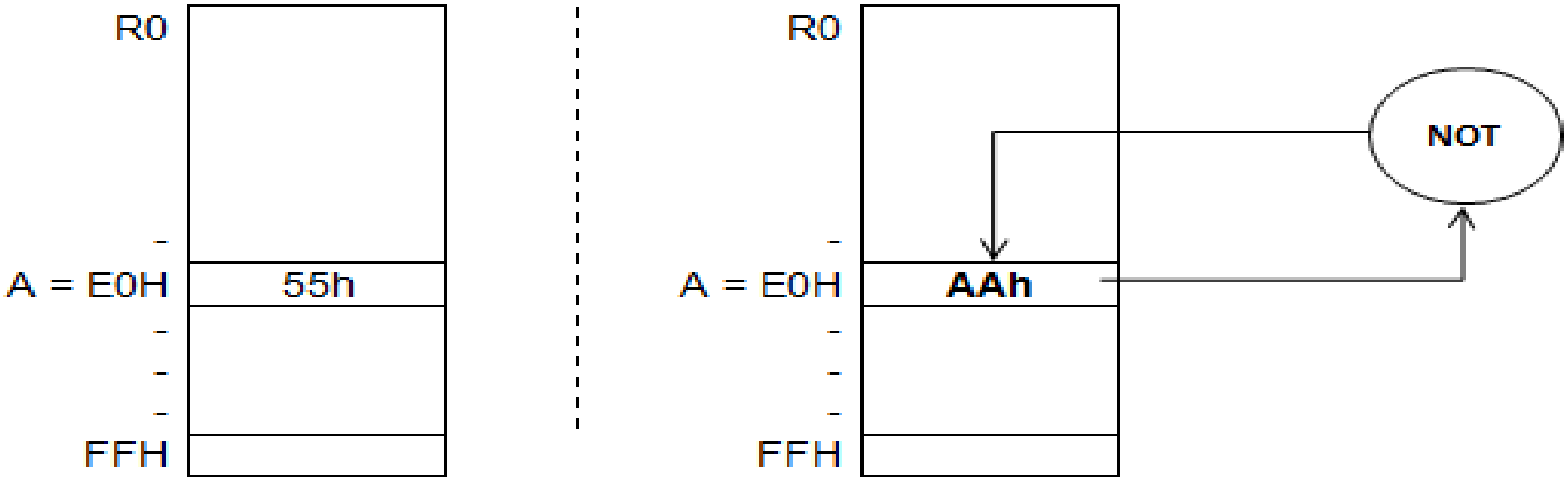


Before and after the execution of the instruction "**XRL A, R1**"

		Binary								HEX
A		1	0	0	1	1	0	1	1	9Bh
	XOR									
R1		0	1	0	1	0	0	1	1	53h
A		1	1	0	0	1	0	0	0	C8h

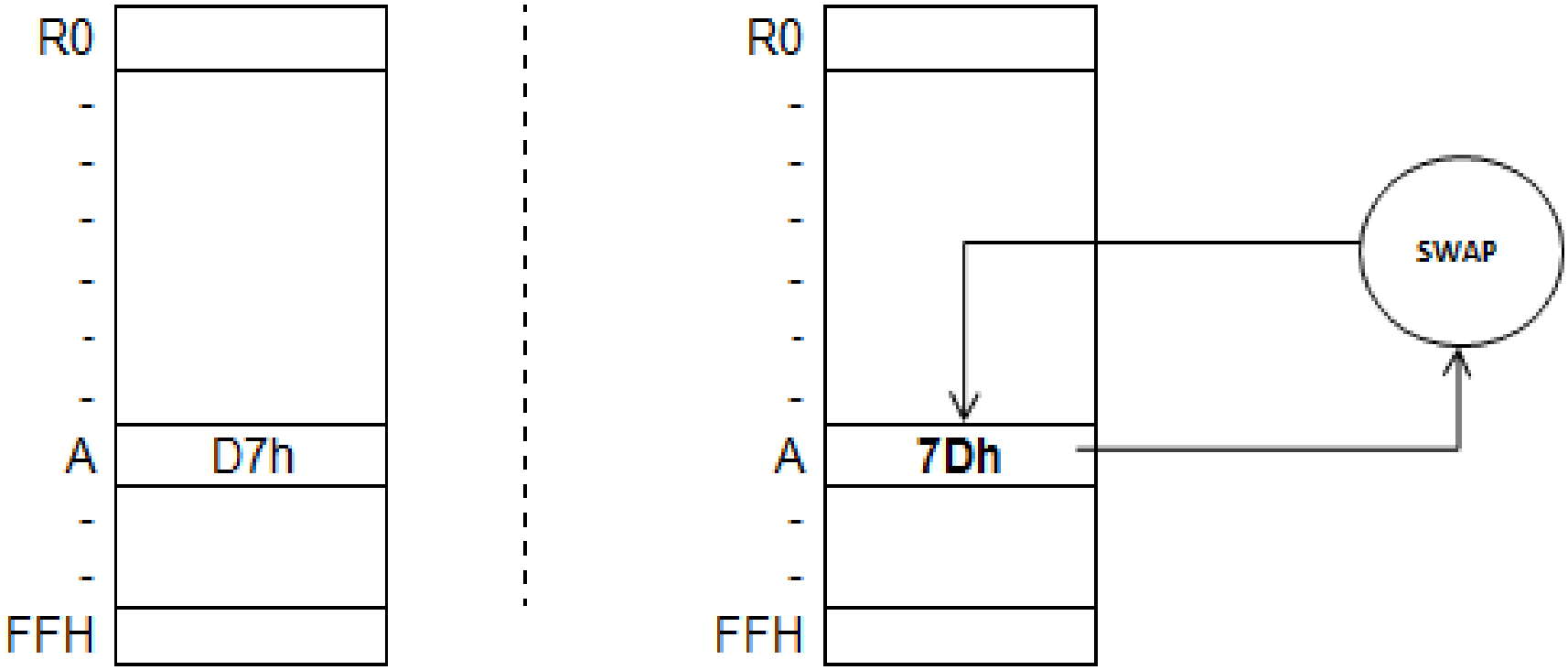
Executing the “CPL” instruction

	Binary								HEX
A	0	1	0	1	0	1	0	1	55h
CPL A	1	0	1	0	1	0	1	0	AAh



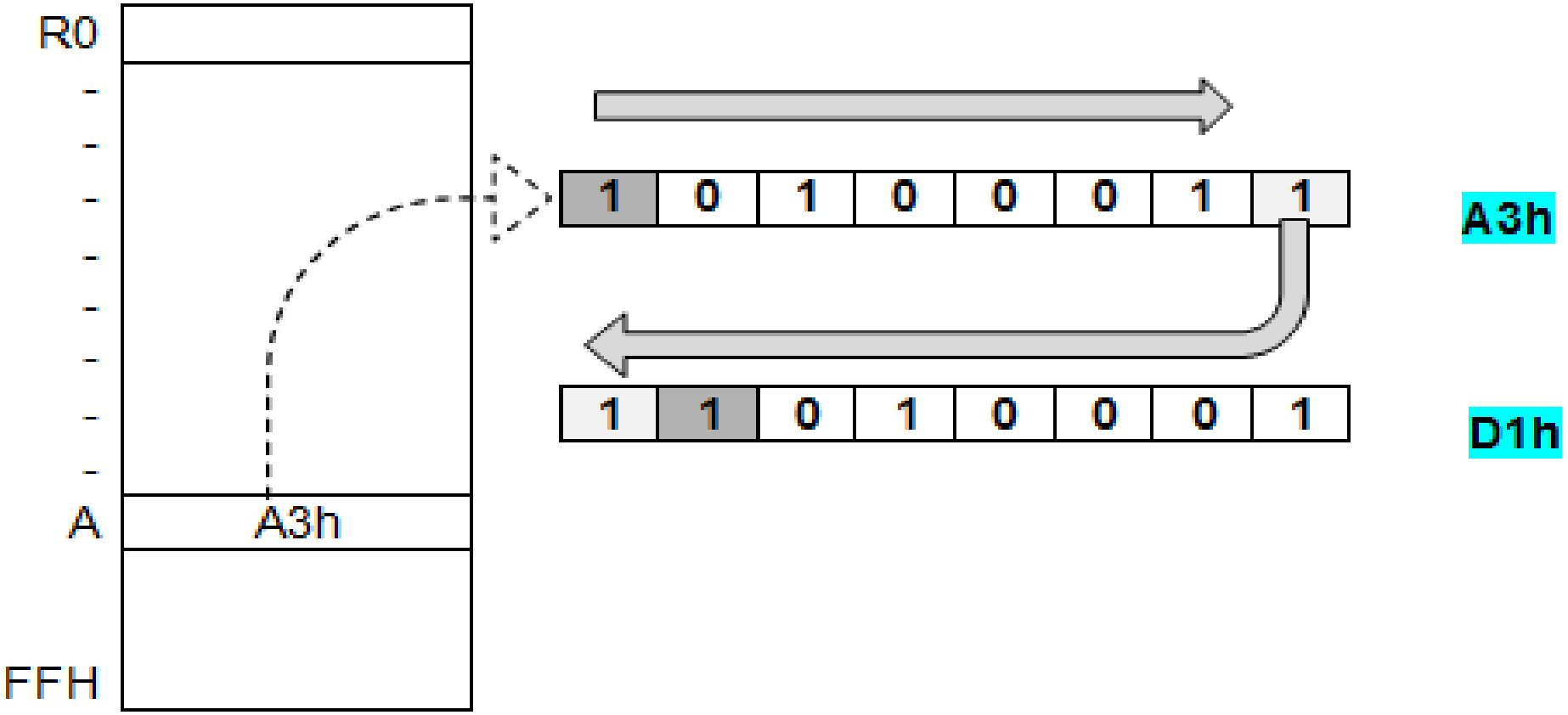
Before and after the execution of the instruction “**CPL A**”

Executing the “SWAP” instruction

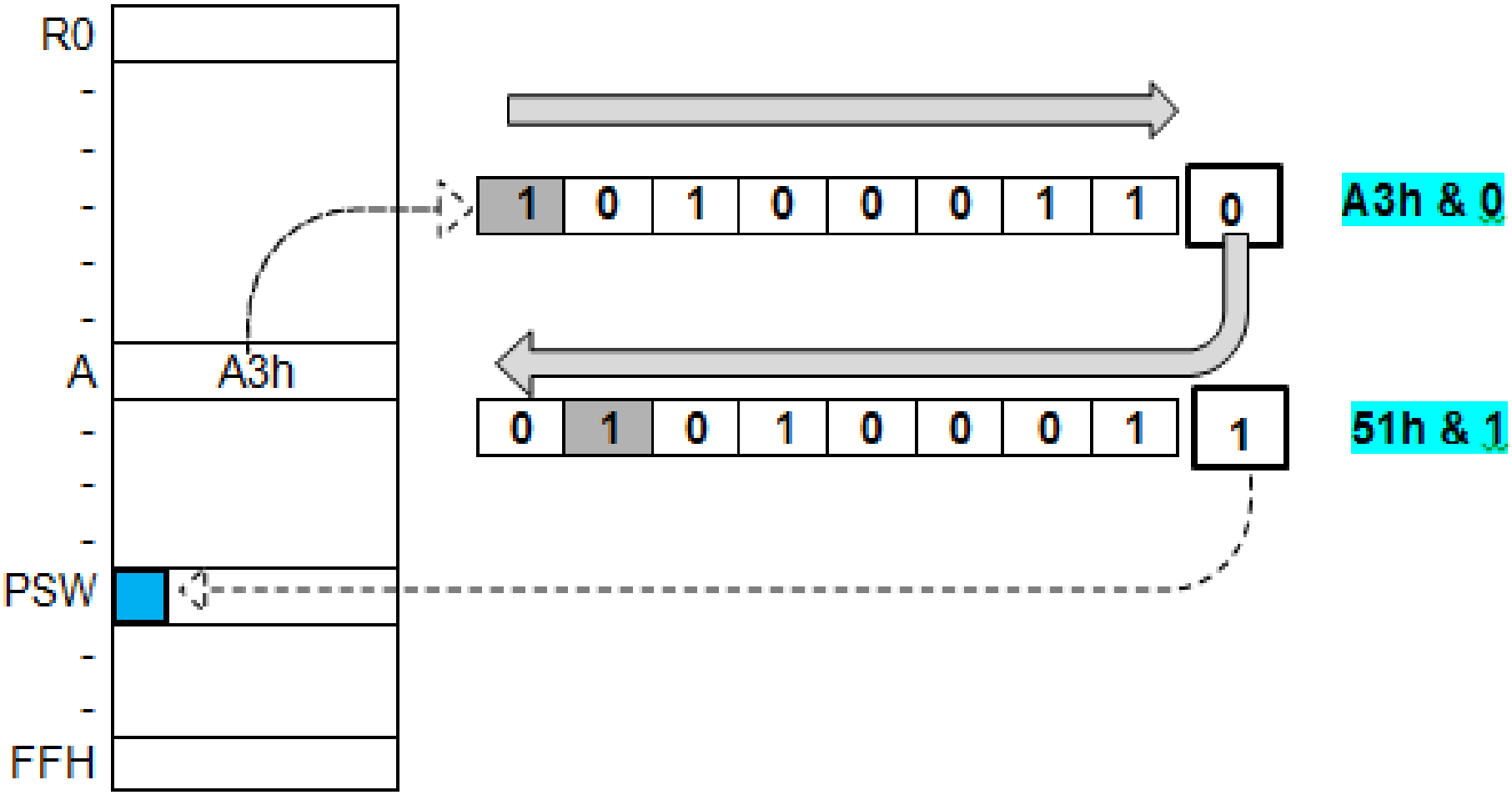


Before and after the execution of the instruction “**SWAP A**”

Executing the “RR” instruction

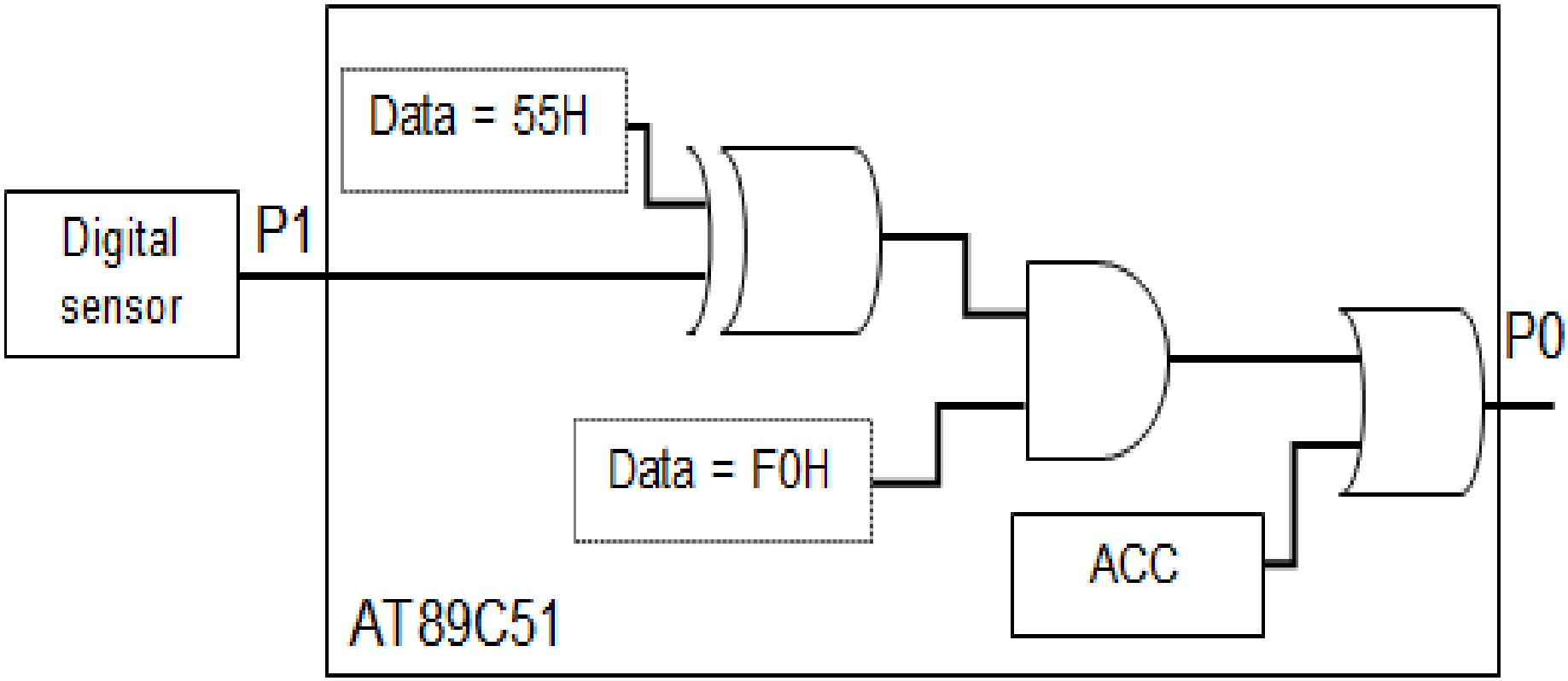


Executing the “RRC” instruction

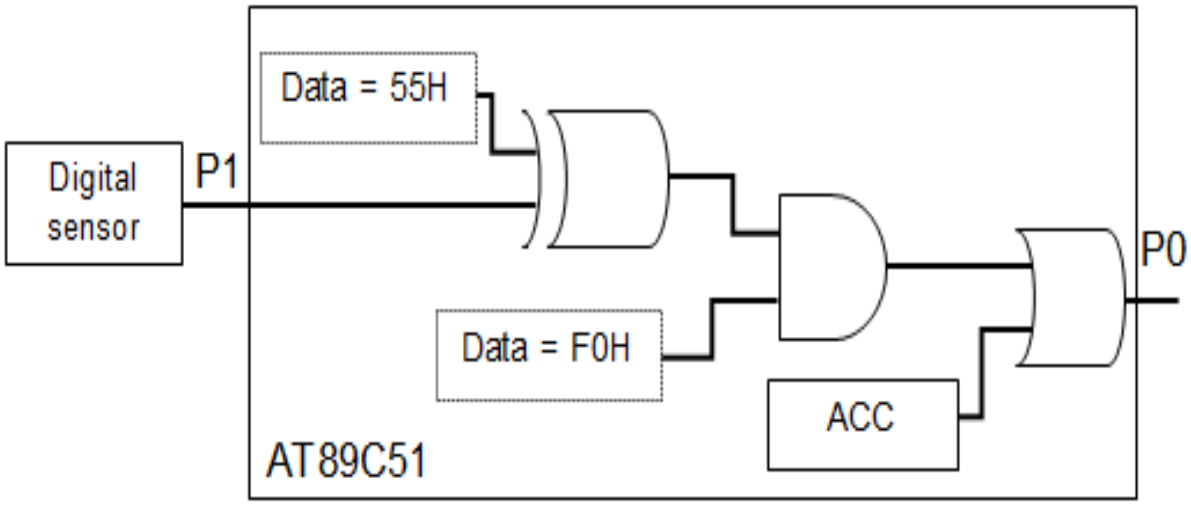


EXAMPLE

Write assembly subprogram to perform the logical operations for the following block diagram using instruction "XCH" of AT89C51.



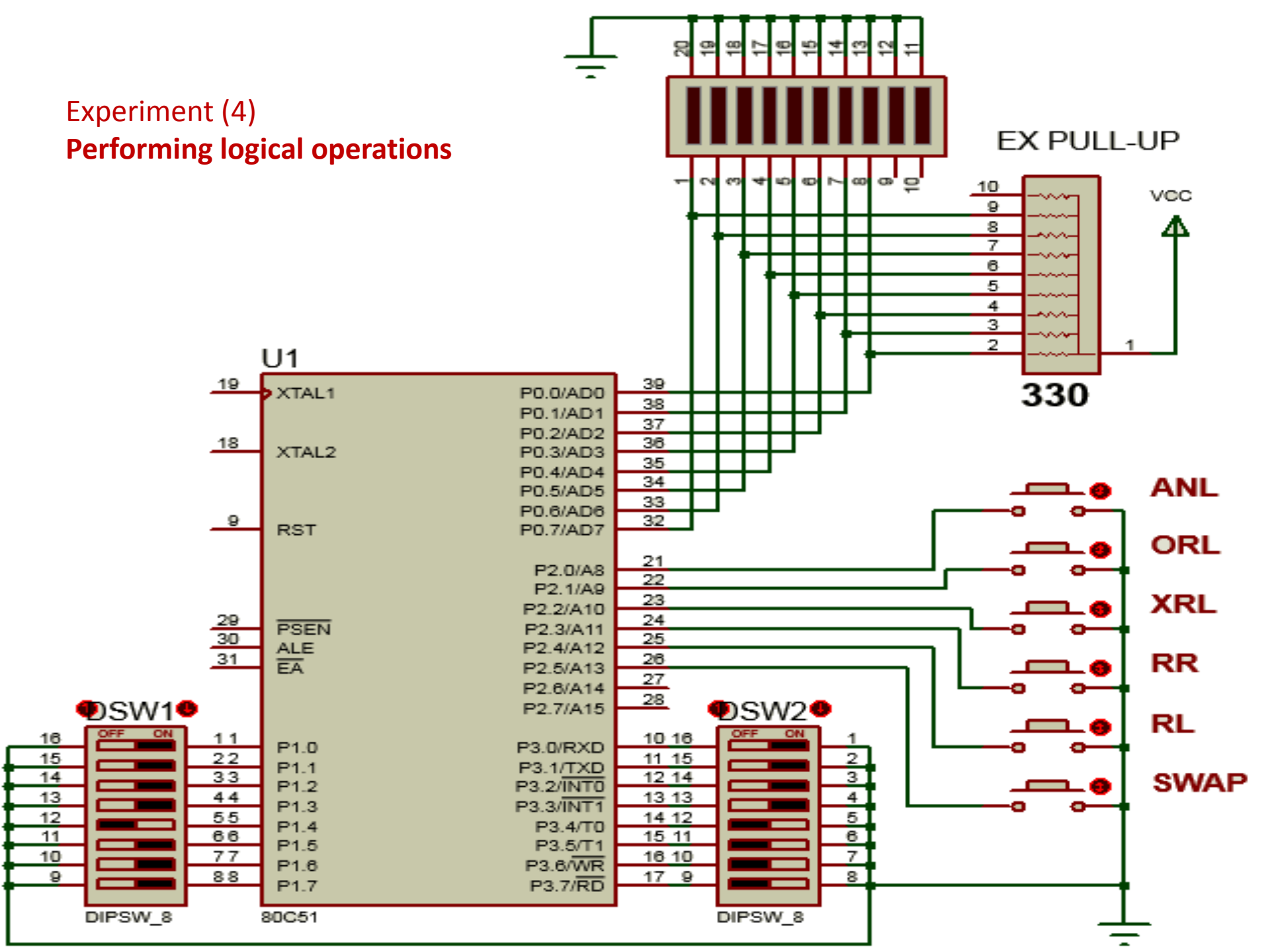
The subprogram

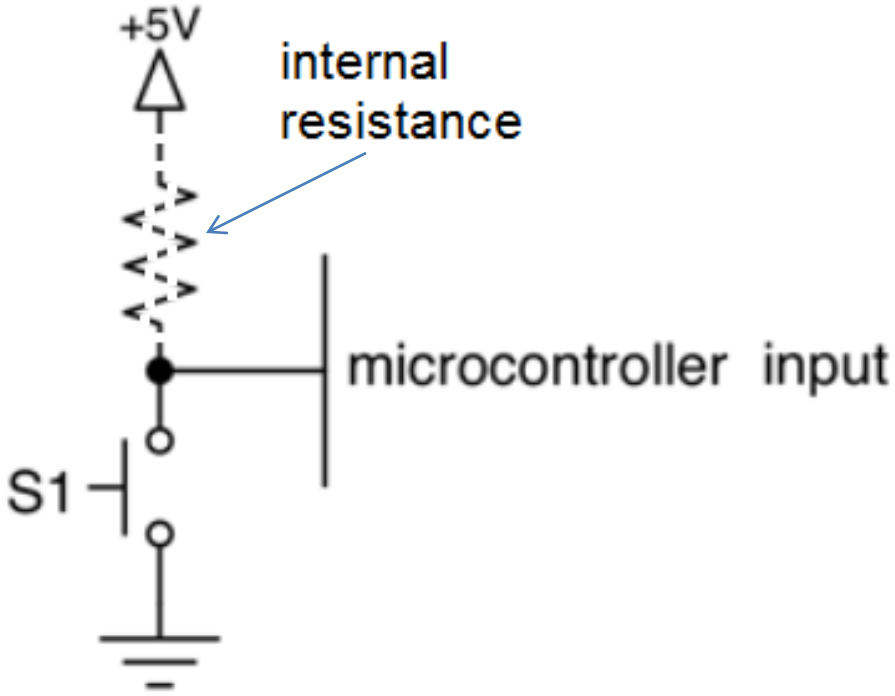



```
INCLUDE 8051.mc
MOV P1, #FFH     ; set P1 input port
XCH A, R0        ; save A into R0
MOV A, P1        ; read P1
XRL A, #55H      ;XOR data with A
ANL A, #F0H      ;AND data with A
ORL A, R0        ;OR data with A
MOV P0, A        ;output data via P0
XCH A, R0        ;retrieve the saved data
```

Experiment (4)

Performing logical operations





JNB pin, label

If input = ‘0’ then go to label

Else go to next

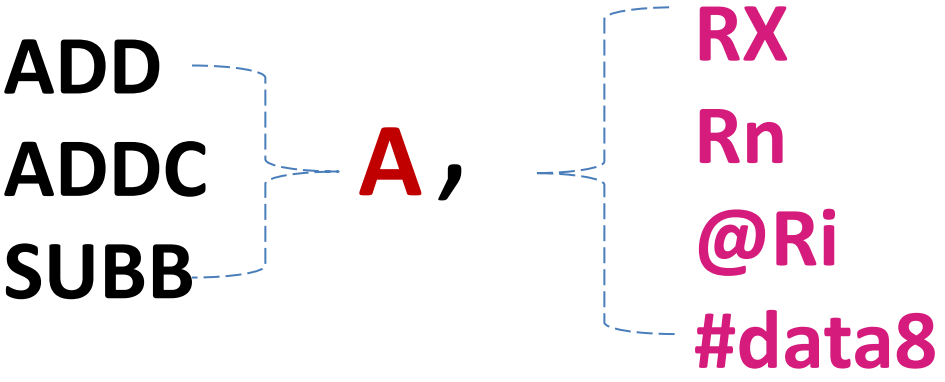
```
12          org    0h
13          sjmp    MAIN
14  ;=============================================================
15          org    30h
16  MAIN:    MOV P1, #0FFH ; SET port TO ACCEPT INPUTs
17           MOV P2, #0FFH ; SET port TO ACCEPT INPUTs
18           MOV P3, #0FFH ; SET port TO ACCEPT INPUTs
19           MOV P0, #00H ;CLEAR THE PORT
20  ;=============================================================
21  START:   MOV A,P1 ; READ INP1 TO ACC
22           MOV R3,P3   ; READ INP2 TO R3
23           MOV 30H,P3    ; READ INP2 TO RAM30
24           JNB P2.0, ANDING
25           JNB P2.1, ORING
26           JNB P2.2, XORING
27           JNB P2.3, ROR_INP1
28           JNB P2.4, ROL_INP1
29           JNB P2.5, SWAP_INP1
30           SJMP START   ; CHECK AGAIN
```

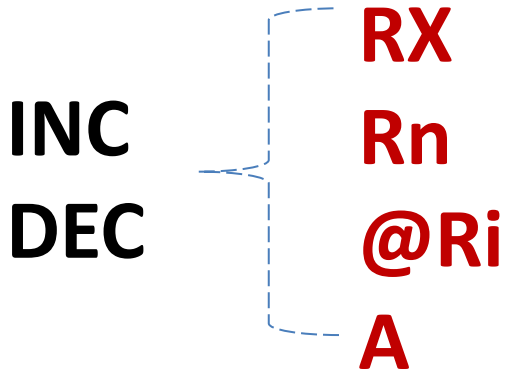
```
32 ANDING:      ANL  A,R3   ; or ANL  A,30h
33              MOV P0, A
34              SJMP START
35 ORING:       ORL  A,R3   ; or ORL  A,30h
36              MOV P0, A
37              SJMP START
38 XORING:      XRL  A,R3   ; or XRL  A,30h
39              MOV P0, A
40              SJMP START
41 ROR_INP1:    RR A
42              MOV P0, A
43              SJMP START
44 ROL_INP1:    RL A
45              MOV P0, A
46              SJMP START
47 SWAP_INP1:   SWAP A
48              MOV P0, A
49              SJMP START
```

The Arithmetic Operation Instructions of MCS-51

Arithmetic Instructions	
Instruction	Description
ADD A, #data	8-bit data + A → A
ADD A, @Ri	data from internal RAM (indirectly[Ri]) + A → A
ADD A, Rx	Data from internal RAM + A → A
ADD A, Rn	Data from (R0… R7) + A → A
ADDC A, #data	8-bit data + A +CY → A
ADDC A, @Ri	data from internal RAM (indirectly[Ri]) + A +CY → A
ADDC A, Rx	Data from internal RAM + A + CY → A
ADDC A, Rn	Data from (R0… R7) + A +CY → A
SUBB A, #data	A – (8-bit data + CY) → A
SUBB A, @Ri	A – (data from internal RAM (indirectly[Ri]) + CY) → A
SUBB A, Rx	A – (Data from internal RAM + CY) → A
SUBB A, Rn	A – (Data from (R0… R7) + CY) → A
INC A	Incrementing the content of accumulator
INC Rn	Incrementing the content of RAM location (R0:R7)
INC Rx	Incrementing the content of any RAM location
INC @Ri	Incrementing the content of any RAM location indirectly using (R0 and R1)
INC DPTR	Incrementing the content of DPTR
DEC A	Decrement the content of accumulator
DEC Rn	Decrementing the content of RAM location (R0:R7)
DEC Rx	Decrementing the content of any RAM location
DEC @Ri	Decrementing the content of any RAM location indirectly using (R0 and R1)
MUL AB	Multiplying the accumulator with the B-register
DIV AB	Dividing the accumulator by the B-register
DA A	Digital adjustment (BCD) for the accumulator

The arithmetic-operation instructions





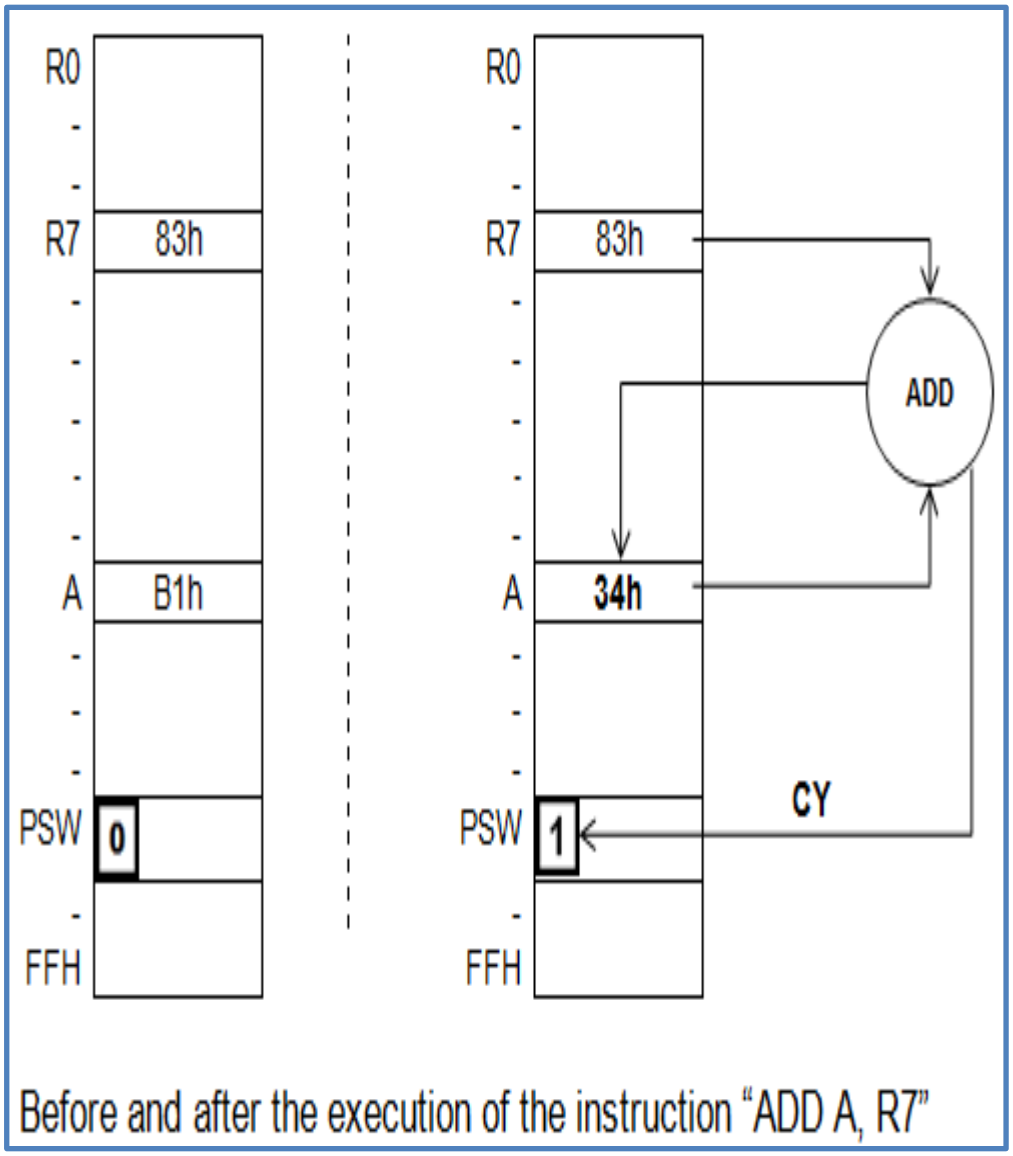
INC DPTR

DIV AB

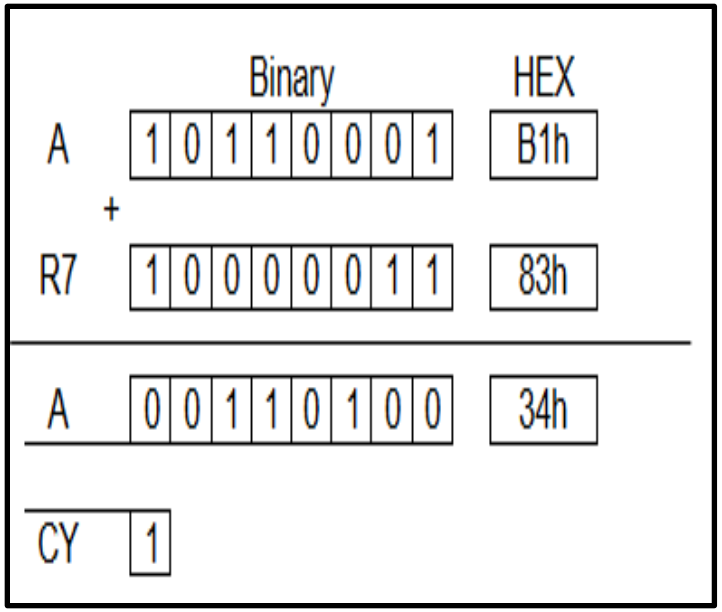
MUL AB

DA A

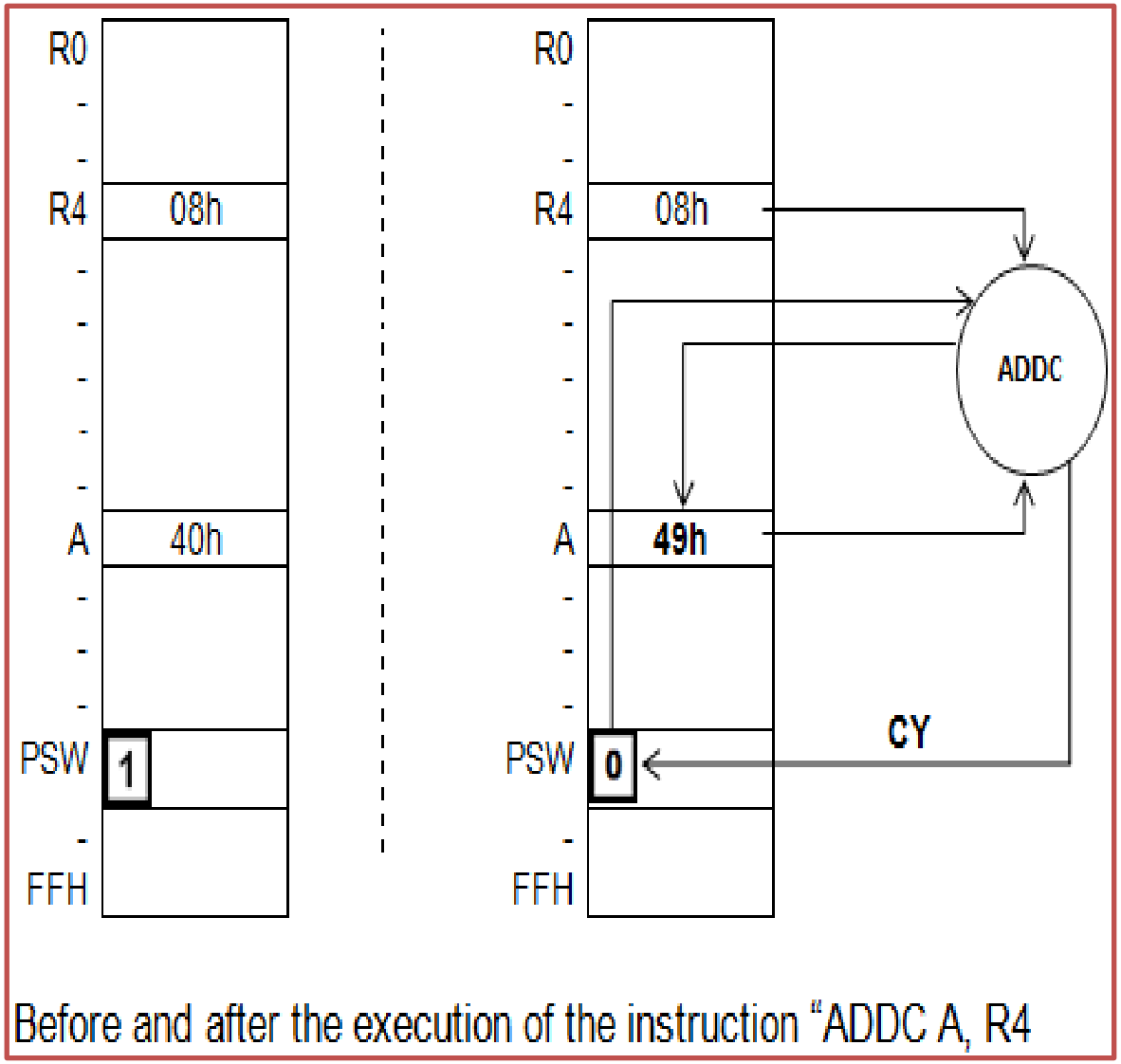
Executing the "ADD" instruction



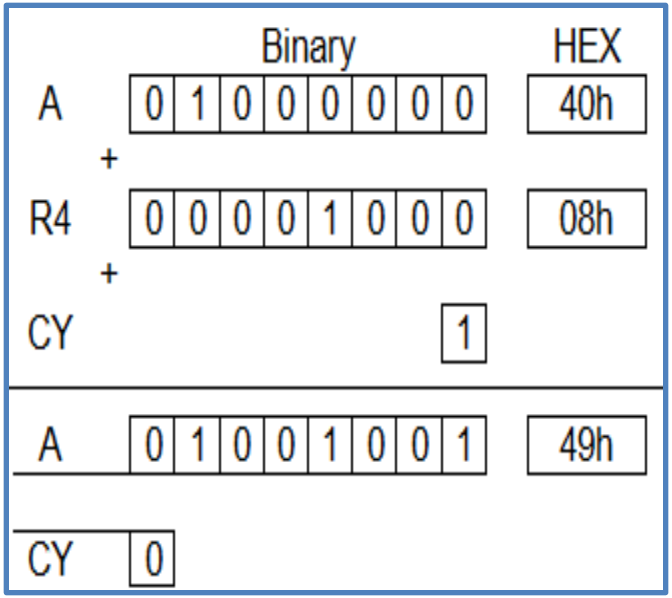
Before and after the execution of the instruction "ADD A, R7"



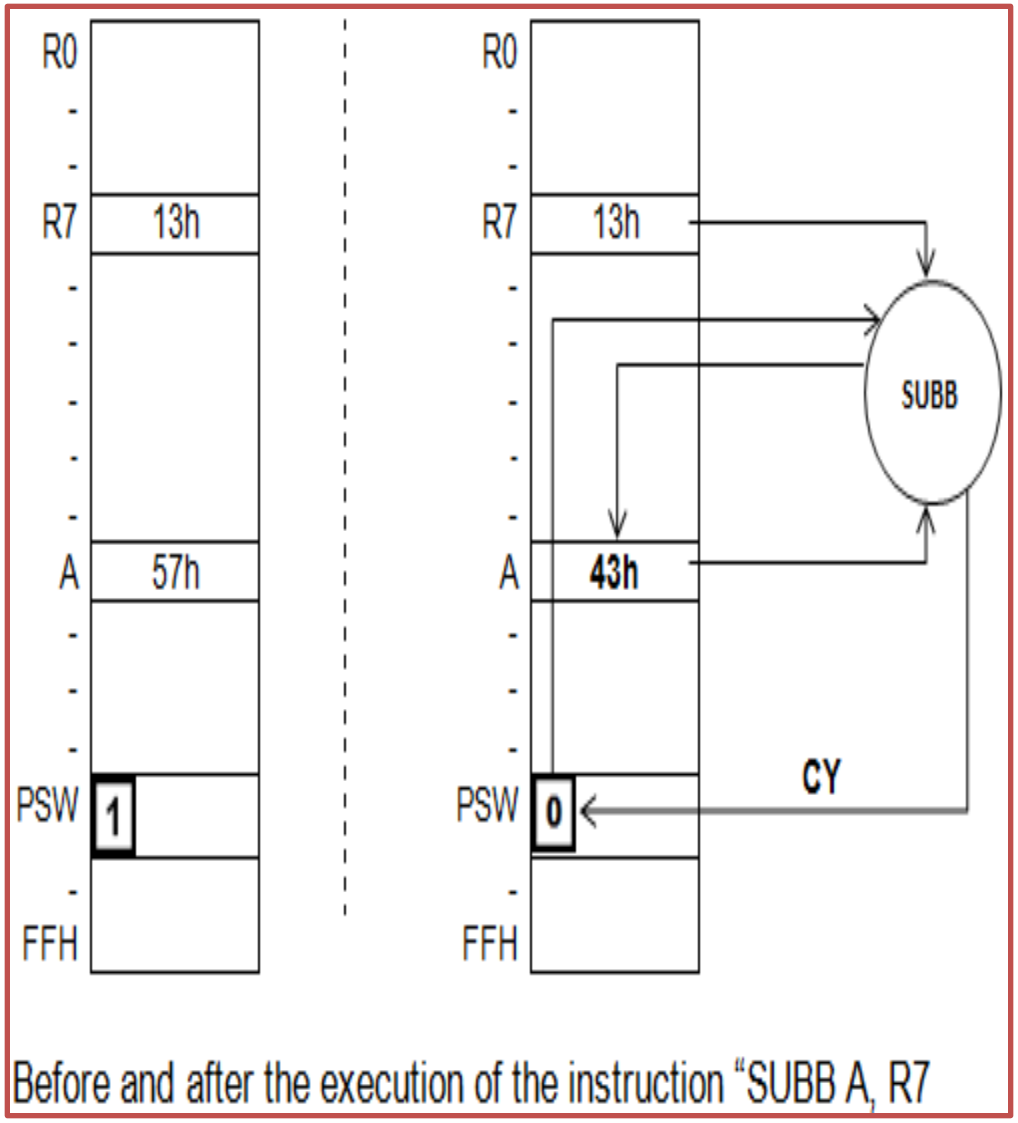
Executing the "ADDC" instruction



Before and after the execution of the instruction "ADDC A, R4



Executing the "SUBB" instruction



Before and after the execution of the instruction "SUBB A, R7

	Binary	HEX
A	0 1 0 1 0 1 1 1	57h
R7	0 0 0 1 0 0 1 1	13h
CY	1	
A	0 1 0 0 0 0 1 1	43h
CY	0	