8. How can we interchange the

| | |
|---|---|
| 5 | 3 |
| 3 | 5 |

→ rotate it 4 times.

```
MOV CL, 4
ROL BL, CL
```

not with carry

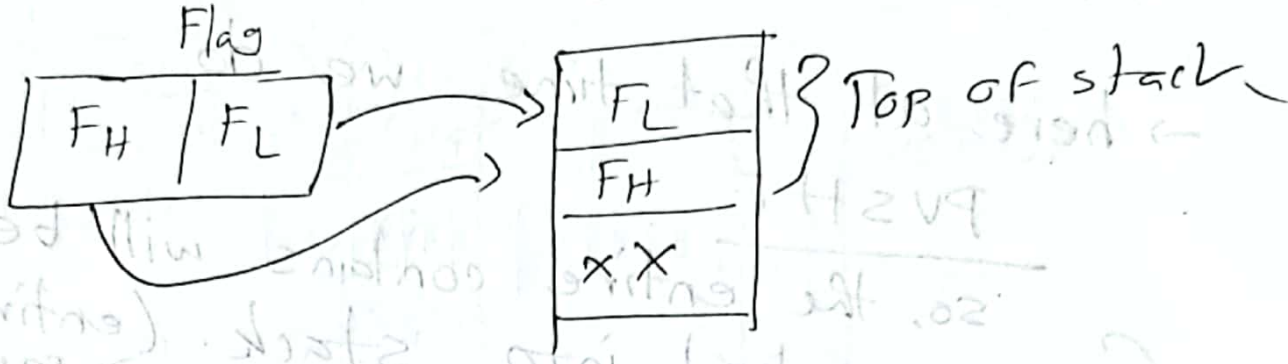
(a) Machine Processor control loops

via these instruction we can directly operate on flags.

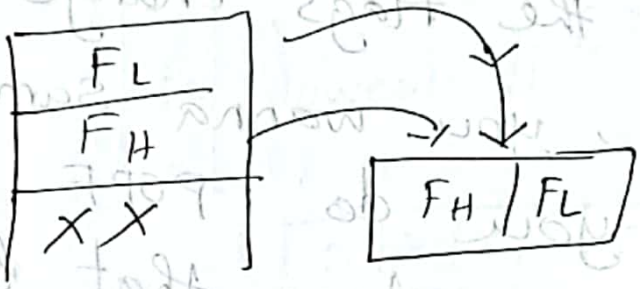
1) Push F → push
2) POP F
3) LAHF
4) SAHF

When we do arithmetic/logical instructions like: ADD BL, DL - by product they are getting effected

## PUSH F




## POP F




**LAHF** = Load AH from Flag

AH ← Flag (FL)

lower byte of flag register loaded on AH.

**SAHF** = store AH back into flag.

AH → Flag

**Use:** for some reason during computation you want the same flags back after 20/30 instructions, (you cannot assume flags will contain same after 20 instructions)

flag will change after ~~at~~ every arithmetic & logical instructions.

→ here, at that time we do

PUSHF

so, the entire contains will be pushed into stack. (entire info is saved in stack)

For whole flag reg.

now, let the flags change 100 times. whenever, you wanna same contains back, you do POPF.

POPF will restore that back from stack into the flag register.

so, we can restore that valve to the previous valve.

For lower byte of flag; we do

LAHF & SAHF

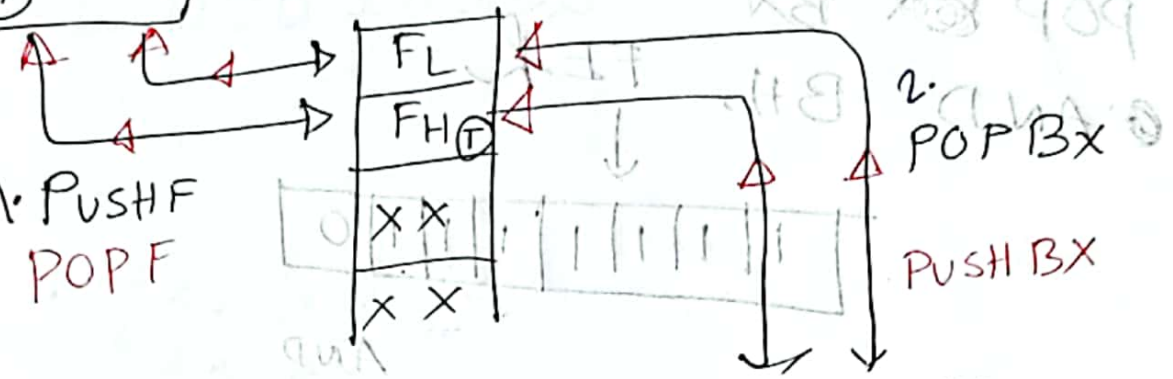
TF

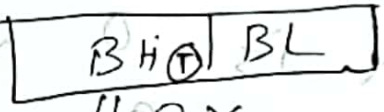
TO SET TF = 1

without effecting other flags.

1(b)

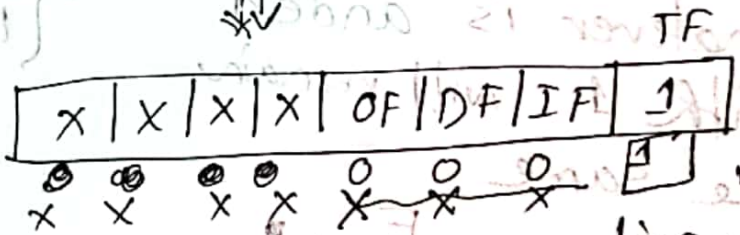
Flag


BX

// Trap flag is the lowest bit in FH byte.



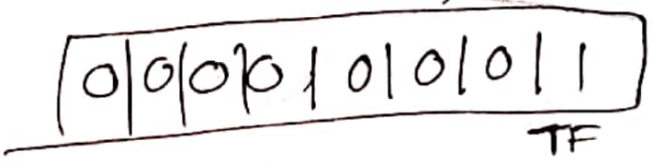

// now, we want to SET/1 to TF, we need a logic operation we need to do OR operation. since

// anything OR with 0, will remain same

// anything OR with 1, becomes 1.

| OR | |
|---|---|
| 0 | 0 → 0 |
| 0 | 1 → 1 |
| 1 | 0 → 1 |
| 1 | 1 → 1 |

OR BH, 01H




Then, PUSH BX
POPF

## To clear TF = 0

PUSH F

POP BX

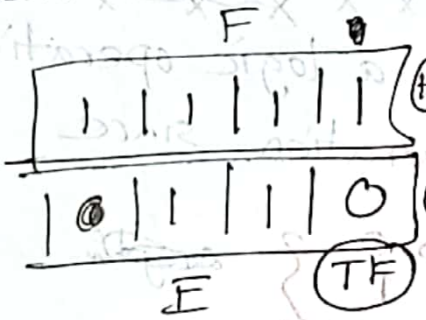
AND BH, FEH


"whatever is 0 has to become 0,

| AND | | |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

whatever is anded with 1, will remain the same.

→ so,




H ANDED will remain same

L ANDED with 0, has to become 0,

PUSH BX

POP F