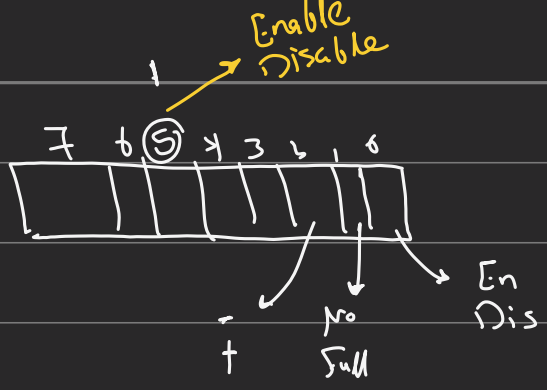
Bit wise operation



& No;

- And operation x & y
- OR operation x | y
- Complement operation ~x ~y
- Right operation >>
- left shift <<

And operator →

A	B	out
0	0	0
0	1	0
1	0	0
1	1	1

char x = 5; char z = 7

x & z → 2

0 0 0 0 0 1 0 1

&

0 0 0 0 0 1 1 1

---

0 0 0 0 0 1 0 1 → 5

clear → for bit

↳ And

OR operation →

A	B	out
0	0	0
0	1	1
1	0	1
1	1	1

x = 10; | y = 20;

x → 0 0 0 0 1 0 1 0

y → 0 0 0 1 0 1 0 0

---

0 0 0 1 1 1 1 0 →

set bit → 1

↳ OR

Created by Notein

Complement

$x = 9$

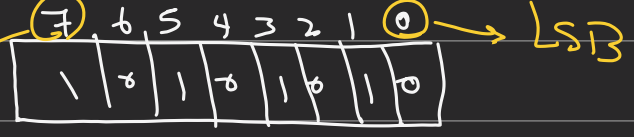
$\sim x \rightarrow$ 0000 1001

1111 0110

A	out
0	1
1	0

toggel → Byte

led



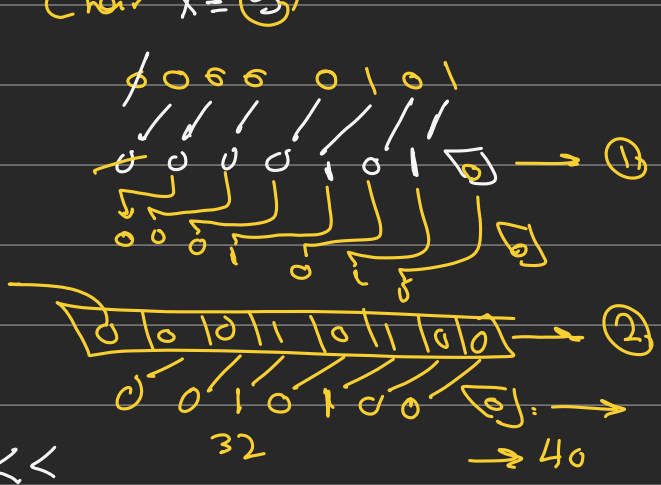
$\sim$ led 0 1 0 1 0 1 0 1

left shift << X<<3

① MSB → out

② shift

③ LSB → zero

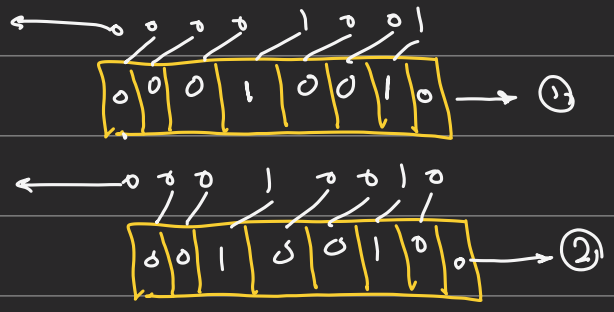


Right shift >>

left shift

X = 9 → 2

X << 2 → left



① MSB out

② shift ← left

③ LSB = 0

Right shift

X = 9 → 2

X >> 2 → Right

① LSB out

② shift ← Right

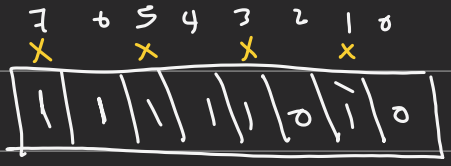
③ MSB = 0

int X = 1005 → MSB → 31

LSB → 0

---

* →



① Clear For 3Bit

ADC = ADC & 0b 0101 0101

& ADC → 1 1 1 1 1 0 1 0

0 1 0 1 0 1 0 1

0 1 0 1 0 1 0

② Clear → Two Bit

ADC = 15 → Bit (2, 3)

ADC → 0000 1111

&

1111 0011

0000 0011

if you need clear some Bits

① → and operation → with var →

Created by Notein

if you need clear some bit

Reg = Reg & ob (value) ; → ✓

set for some bit (1)

4 , 6

X = 5 → 0 0 0 0 0101

0101 0000

↳ use OR operator → X = X | ob 01010000

search

→ set for 1 Bit → Clear Bit
→ toggle for some Bit → toggle for Bit

(Branching & Condition statment) [if | switch]

① greater than >
② greather than or eq >=
③ Not equal !=
④ less than <
⑤ less than or eqve <=
⑥ equal ==

true → any number except zero
False → 0

Created by Notein

$x = 5 \quad y = 7$

$x == y \rightarrow 0$

$x != y \rightarrow 1$

$x > y \rightarrow 0$

$x >= y \rightarrow 0$

if_statement

21

x = 20

↳ Syntax → `if (x == 20)` → Condition false

{

`printf("x = 20\n");` → Action

}

≡

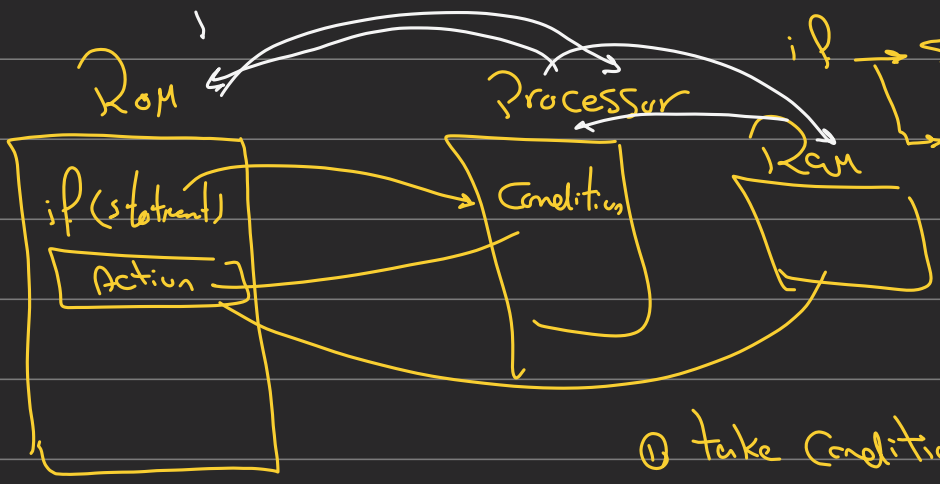
x = 20 → false

```
if (x == 20)
{
    //Action1,
}
else
{
    //Action
}
```

Created by Notein

```
char Degree ;
Scanf ("%d", &Degree);
if( Degree == 50)
  {
    Pass;
  }
else if ( Degree == 60)
  {
    good;
  }
else if ( Degree == 70)
  {
    V. Good;
  }
else
  {
    Faild;
  }
```

Condition true



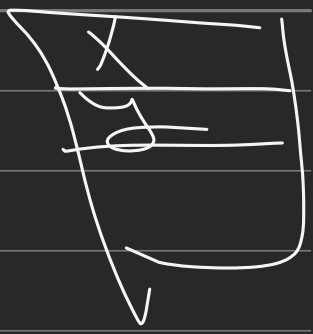
if → save memory
→ take long time

① take Condition from Rom
② if true take Action inside Ram.

Created by Notein

false

```
if ( condition )
{
    int x
    int y
}
else if ( condition )
{
    char x
    char y
}
else
{
    float x
    float y
}
```



---

switch → check equality

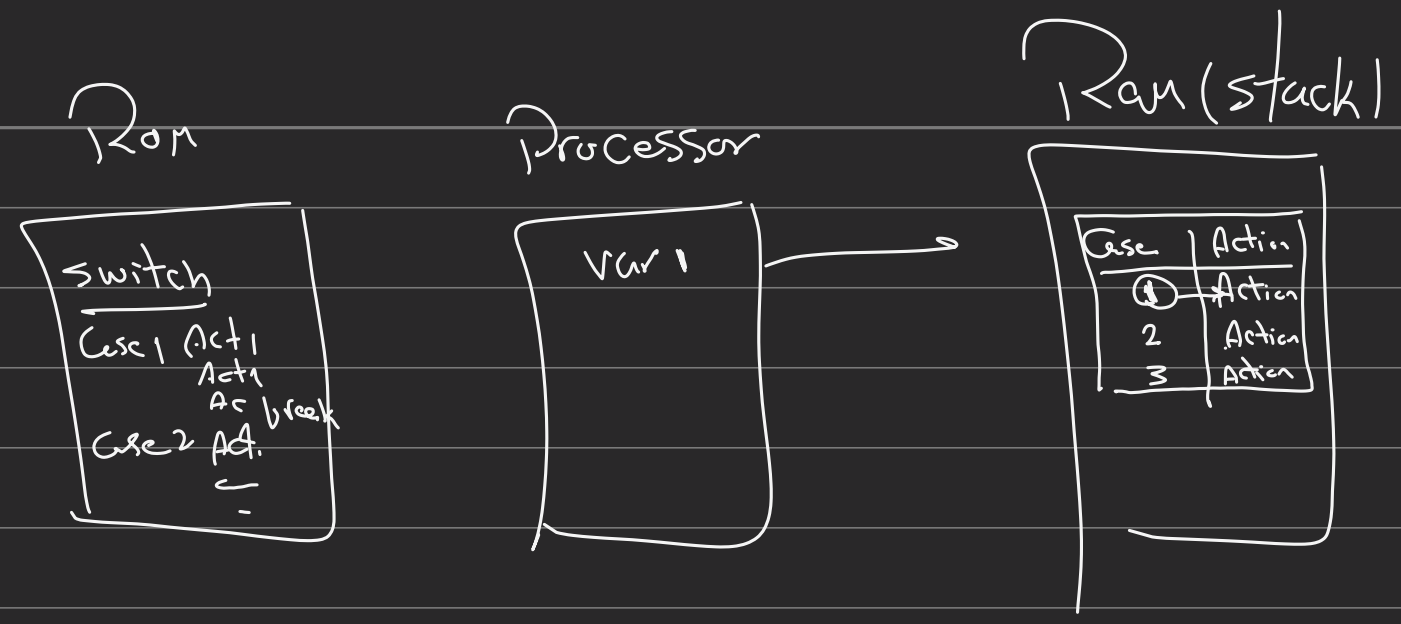
```
switch ( var )
{
    case 1 : Action
    case 2
    case 20
    case 100
    case 00
    default : Action
}
```

var ==

Min Case 2
Max Case unlimited

Constant

Created by Notein



switch → fast execution
↳ take large size of memory

Created by Notein