Unit-3.

Page No:
Date: / /

Instruction :- An instruction is a command given to the computer to perform a specific function.

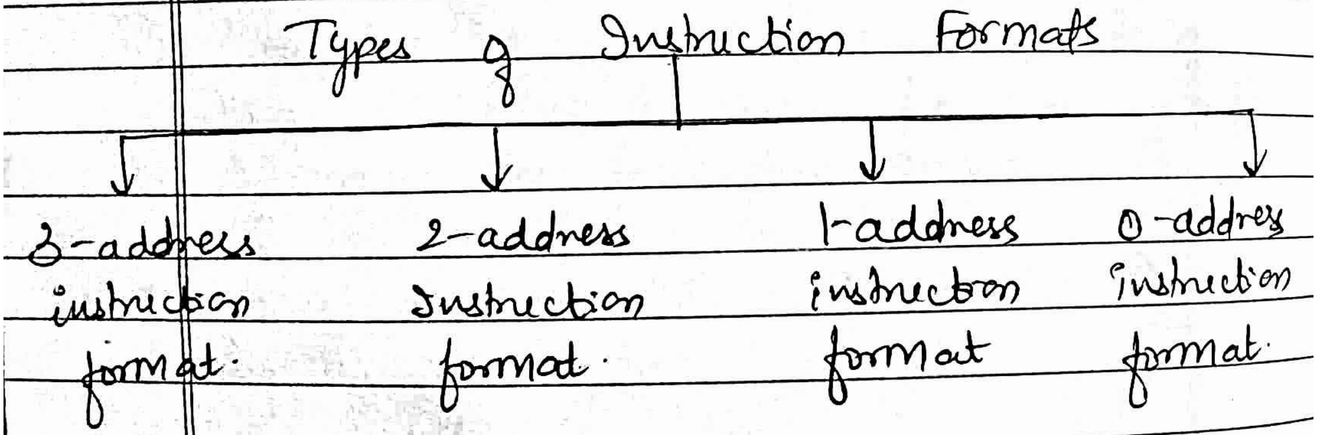
Instruction format

opcode	Mode	Address field

Operation code (opcode) :- Specifies operation to be performed.

Mode field :- specifies which addressing mode is used.

Address field :- Specify a memory address or register address.



Types of Instruction Formats
3-address instruction format.
2-address Instruction format.
1-address instruction format
0-address instruction format.

Example :- Evaluate the arithmetic statement
$X = (A+B) * (C+D)$

using
(i) 3-address instructions
(ii) 2-address instructions
(iii) 1-address instructions
(iv) 0-address instructions

} Assume A, B, C, D and X are memory addresses.

(i) 3-address instructions (General Register organisation)

ADD R_1, A, B	$R_1 \leftarrow M[A] + M[B]$
ADD R_2, C, D	$R_2 \leftarrow M[C] + M[B]$
MUL X, R_1, R_2	$M[X] \leftarrow R_1 * R_2$

(ii) 2-address instruction (General Register organisation)

MOV R_1, A	$R_1 \leftarrow M[A]$
ADD R_1, B	$R_1 \leftarrow R_1 + M[B]$
MOV R_2, C	$R_2 \leftarrow M[C]$
ADD R_2, D	$R_2 \leftarrow R_2 + M[D]$
MUL R_1, R_2	$R_1 \leftarrow R_1 * R_2$
MOV X, R_1	$M[X] \leftarrow R_1$

(iii) 1-address instruction (Accumulator based CPU)

LOAD A	$Ac \leftarrow M[A]$
ADD B	$Ac \leftarrow AC + M[B]$
STORE T	$M[T] \leftarrow Ac$
LOAD C	$Ac \leftarrow M[C]$
ADD D	$Ac \leftarrow Ac + M[D]$
MUL T	$Ac \leftarrow Ac * M[T]$
STORE X	$M[X] \leftarrow Ac$

* Such type of instructions are used in Accumulator based CPU organisation.

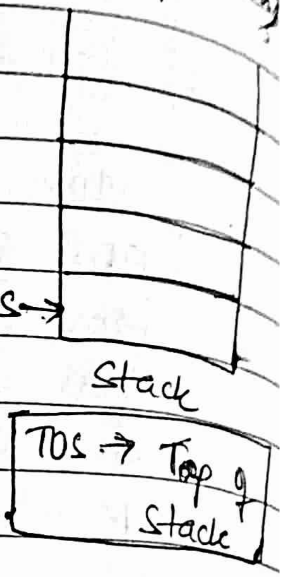
Date: / /

(iv) O address instruction. (Stack Based CPU)

First write in Postfix Notation.

Postfix Notation :- First operands the operation

(A+B) * (C+D)
(AB+) * (CD+)
AB+ CD+ *



PUSH A	[TOS ← A]
PUSH B	[TOS ← B]
ADD	[TOS ← (A+B)]
PUSH C	[TOS ← C]
PUSH D	[TOS ← D]
ADD	[TOS ← (C+D)]
MUL	[TOS ← (C+D) * (A+B)]
POP X	M[X] ← TOS.

Q A computer uses a memory unit with 256 k words of 32 bits each. A binary instruction code is stored in one word of memory. The instruction has 4 parts: an indirect, bit, an operation code, a register code part to specify one of 64 registers, and an address field part.

(a) How many bits are there in the operation code, the register code part, and the address part.

(b) Draw the instruction word format and indicate the no. of bits in each part.
(c) How many bits are there in the data and address inputs of the memory.

Soln (c) Memory unit = 256 k words of 32 bits each.

Since memory unit have = 256 k words
$= 2^8 * 1024$ words
$= 2^8 * 2^{10}$ words.
$= 2^{\textcircled{18}}$ words.

$\boxed{\text{So, No. of Addr bits = 18 bits.}}$

Since each word is of 32 bits
So, $\boxed{\text{No. of data bits = 32 bits}}$

(b) Instruction code format

$\xleftarrow{\hspace{3em}} 32\text{-bit} \xrightarrow{\hspace{3em}}$

Indirect bit	opcode	register code	address ~~part~~
1 bit	7 bits	6 bits	18 bit

(i) indirect bit = 1 bit
(ii) address part bits = No. of addr. bits in memory
= 18 bits
(iii) register code bits = No. of bits required to specify one of 64 registers
$= 2^{\textcircled{6}}$ registers
= 6 bits.

(iv) operation code bits = Total bits – (indirect b
+ address bit +
register code b

$= 32 - (1 + 18 + 6)$
$= 32 - 25$
$= 7$ bit.

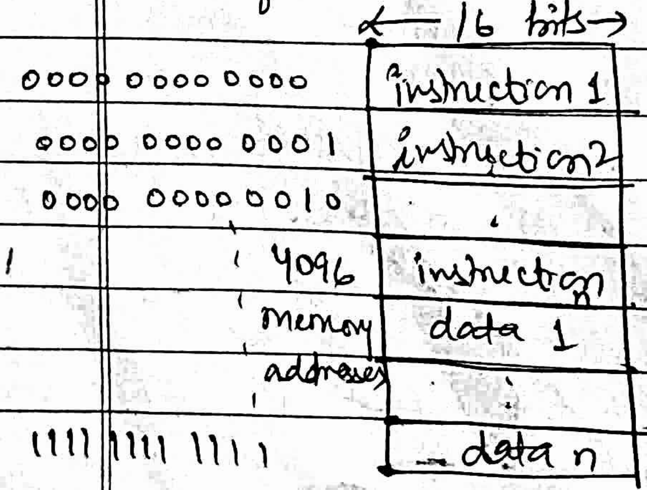
So, Indirect bit = 1 bit
address bits = 18 bits
register code = 6 bits
operation code = 7 bits.

Instruction Code & Types of Instruction Code formats.

Instruction Code :- An instruction code is a group of bits that instruct the computer to perform a specific operation.

Assume a memory system of size $= 4096 \times 16$
$= 2^{12} \times 16$

No. of address bits = 12 bits
No. of ~~add~~ data bits = 16 bits



Types of Instruction code format

1) Memory Reference instruction

(1-bit) 15	14 (3 bits) 12	11 (12 bits) 0
I = 0/1 (Mode bit)	opcode = 000 to 110	Memory address

Indirect bit ← 16 bit →

2) Register Reference instruction

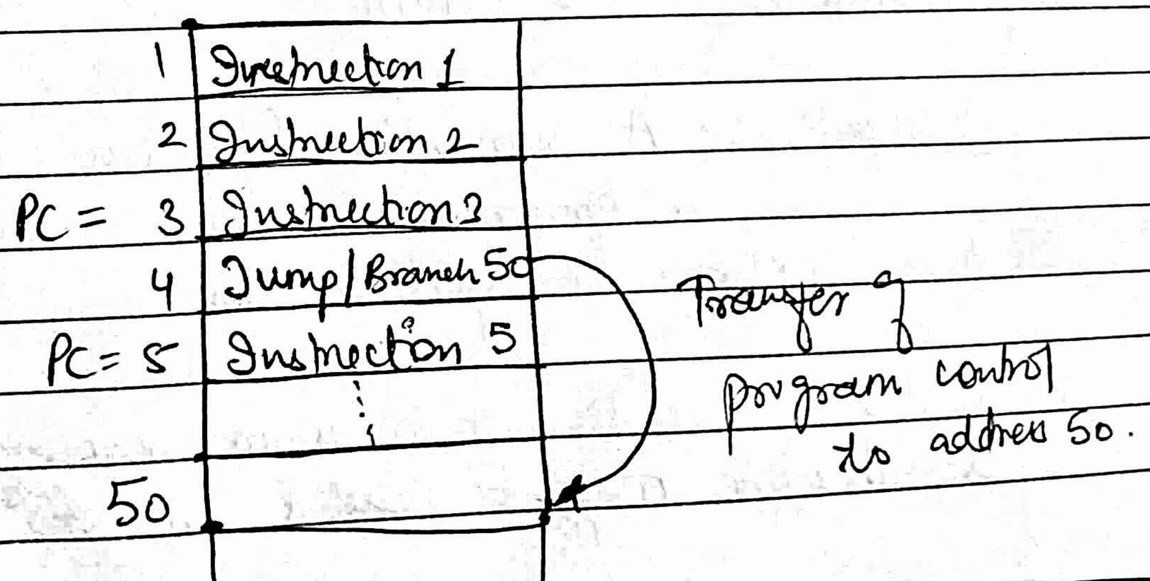
15	14 12	11 0
I = 0	opcode = 111	Register operations

3) Input- output Instruction

15	14 12	11 0
I = 1	opcode = 111	I/O operations

Program Control.

* A program control instruction is a instruction, which when executed, may change the address value in the program counter and causes the control of the program to be shifted to some other location rather than next location.



Page No:
Date: / /

~~Prog~~

* Program control Instructions may be conditional or unconditional.
* Unconditional program control instructions causes the program control to be ~~satisfied~~ shifted to the new address without any condition.
* Conditional program instruction causes the program control to be shifted to the new address only when certain states condition is met, else the program goes to next memory location in sequence.

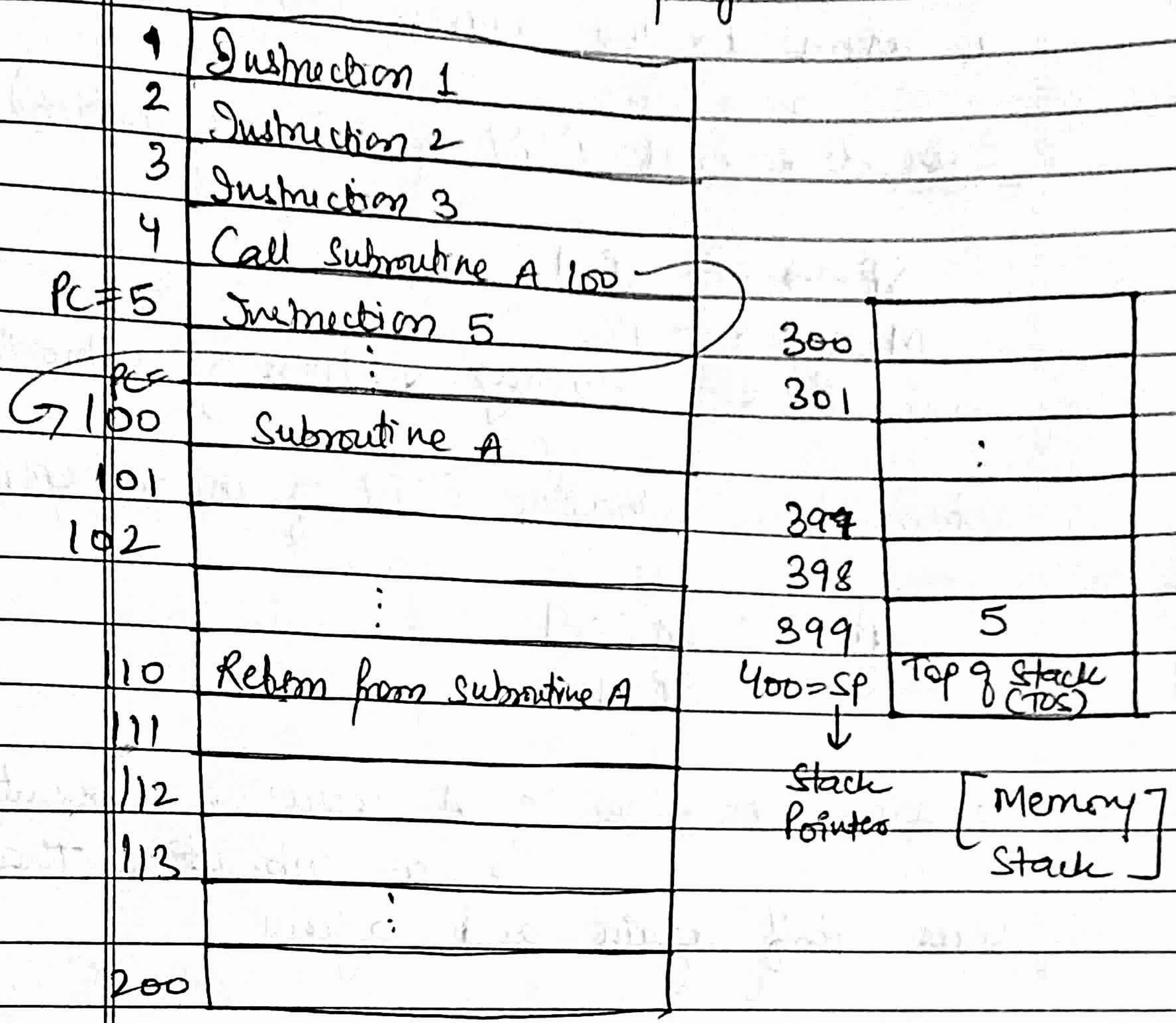
Types of program control instructions

Name	Symbol	
① Jump	JMP	Can be conditional or unconditional.
② Branch	BR	
③ Skip	SKP	
④ Call	CALL	
⑤ Return	RET	
⑥ Compare	CMP	
⑦ Test (by ANDing)	TST.	

Subroutine Call & Return.

Subroutine:- A subroutine (function) is a subprogram in the main program that performs a specific task

① During the execution of a main ~~memory~~ program, a subroutine may be called at various points in the main program.



Main Memory

② A call subroutine instruction contains an operation code along with an address that specify the beginning address of a subroutine.

③ This instruction is executed by performing two operations:-

(i) The address of next instruction (PC) is stored in memory stack.

(ii) The PC (Program Counter) is loaded with the starting address of the subroutine.

Page No:
Date: / /

(4) The last instruction from subroutine causes a return to the address stored in stack.

<u>Subroutine call (List of Micro-operations)</u>

$SP \leftarrow SP - 1$
$M[SP] \leftarrow PC$
$PC \leftarrow$ Starting address of subroutine.

<u>Return from subroutine (List of micro-operations)</u>

$PC \leftarrow M[SP]$
$SP \leftarrow SP + 1$

<u>Recursive Subroutine</u> :- A recursive subroutine is a subroutine that calls itself again and again.

Status bit Conditions / Condition Codes

* Status bit conditions specify the state of the CPU. after the last ALU operation
* These collection of all status bit conditions is called "Program Status word (PSW)".
* The PSW is stored in a special hardware register called "Status Register".
* The value of status bits are set (1) or reset (0) as a result of ALU operation

Common Status flags.

① Carry flag (C) :- Set to 1, if "carry" is generated after ALU operation, else reset to 0.

$C = 1$, If carry generated
$C = 0$, If no carry generated.

② Sign flag (S) :-

$S = 1$, If MSB is 1
$S = 0$, If MSB is 0.

MSB = Most significant bit (Leftmost bit

③ Zero flag (Z) :-

$Z = 1$, when all the bits after ALU Operation are zero. (Output = 0)
$Z = 0$, when output $\neq 0$

Date: / /

④ Overflow flag (V):-

V=1 , XOR of two leftmost bits are 1

V=0, XOR of two leftmost bits are 0.

Example :- let A= 10110110
B= 11001011

ALU operation :- A+B (Add)

```
A=  ~~10110110~~
B=  ~~11001011~~
   ~~10000001~~

     10110110
  +  11111011
   1 10110001
```

Carry generated
C=1

MSB=1
S=1

Z=0
Output ≠0

$1 \oplus 0 = 1$
V=1

Status bit conditions