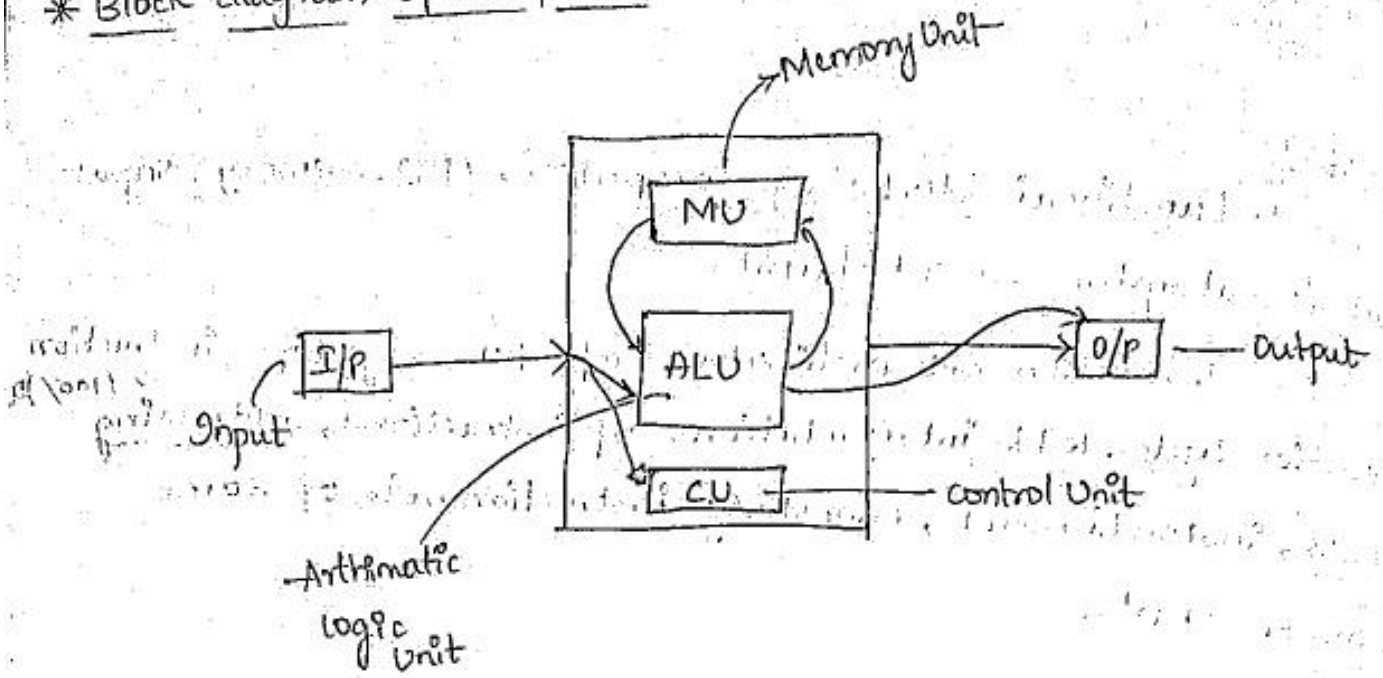
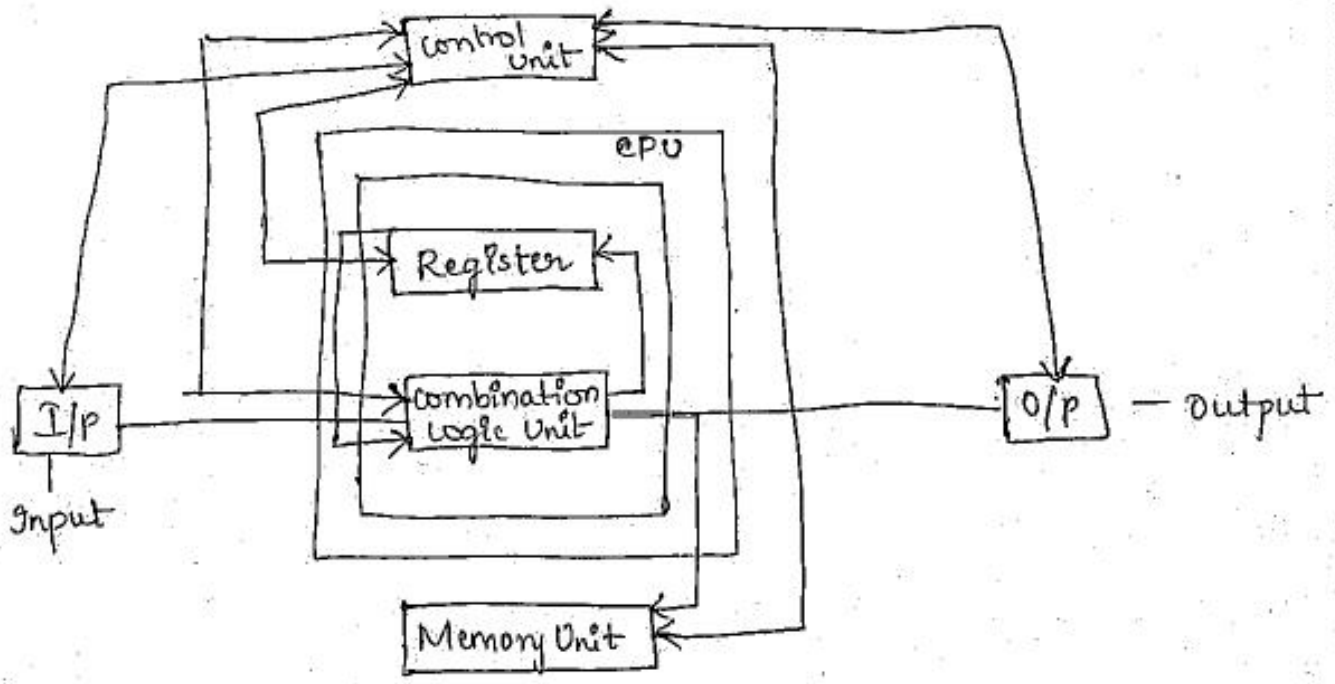
11/09/21

* Block diagram of computers



* Computer organisation is a set of rules and methods for execution

* Block Diagram of Instruction set of architecture of CPU:



14/09/21

Instruction formats

General Representation:

Opcode	Mode	Address

→ computers may have instruction of several different lengths. CPU organisation
→ there are three types of organisation

1. Single Accumulator.
2. General Register organisation
3. Stack organisation

1. **Single Accumulator:** It is a intermediate storage of arthimatic and logic functions. All data is stored in accumulator. Accumulator is also called as collector.

Eg: Addition of two numbers ADD A, B

X = Address

AC ← ADD X[A], X[B]

2) **General Register Organisation:** Here the data will be stored in registers. Eg: Add of A, B R1 ← ADD X[A], X[B]

3) **Stack Organisation:** Here push and pop instructions are used in stack organisation.

(10M)

Instruction format:

→ It is having three fields Opcode, Mode, Address

→ The bits of instructions are divided into the groups is called field.

1. **Opcode:** Operation code field that specifies the operations to be performed.

   Eg: Addition, substraction, multiplication etc.

2. **Mode:** A mode field specifies the way of operands.

   Eg: a * b Here a, b are operands. These operands are stored in mode field.

3. Address: In Address field locate the memory address or process of registers.

* Instruction formats are divided into four types:

1. Three Address Instruction
2. Two Address Instruction
3. One Address Instruction
4. Zero Address instruction

Note: Three Address and two Address formats are more efficient.

1. Three Address Instruction format:

Example: $X = (A+B) * (C+D)$

ADD R1, A, B $R_1 \leftarrow X[A] + X[B]$

ADD R2, C, D $R_2 \leftarrow X[C] + X[D]$

MUL X, R1, R2 $X[X] \leftarrow R_1 * R_2$

→ Here, we are using two variables and one register to calculate the process

17/09/21

2. Two Address Instruction format:

Example: $X = (A+B) * (C+D)$

MOV R1, A $R1 \leftarrow M[A]$

ADD R1, B $R1 \leftarrow R_1 + M[B]$

MOV R2, C $R2 \leftarrow M[C]$

ADD R2, D $R_2 \leftarrow R_2 + M[D]$

MUL R1, R2 $R1 \leftarrow R1 * R2$

MOV X, R1 $M[X] \leftarrow R_1$

→ MOV Instruction moves or transfers the operands to and from memory and processor registers.

→ Here, we are using one variable and one register to calculate the process

3. One-Address Instructions

$X = (A+B) * (C+D)$

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	$AC \leftarrow AC + M[D]$
STORE	Y	$M[Y] \leftarrow AC$
MUL	T	$AC \leftarrow AC + M[T]$
STORE	X	$M[X] \leftarrow AC$

Zero-Address Instructions

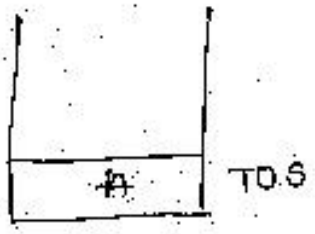
1. This Instruction set completely follows the stack organisation.
2. Here, we are using two types of operations a) push b) pop

TOS – Top of the stack

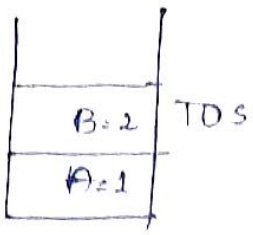
Eg: $X = (A+B) * (C+D)$

LOAD	A	
PUSH	A	$TOS \leftarrow A$
PUSH	B	$TOS \leftarrow B$
ADD		$TOS \leftarrow (A+B)$
PUSH	C	$TOS \leftarrow C$
PUSH	D	$TOS \leftarrow D$
ADD		$TOS \leftarrow (C+D)$
MUL		$TOS \leftarrow (C+D) * (A+B)$
POP	X	$M[X] \leftarrow TOS$

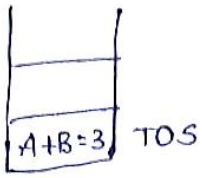
Step: 1 Push A into the stack



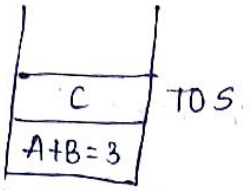
Step:2 push B into the stack



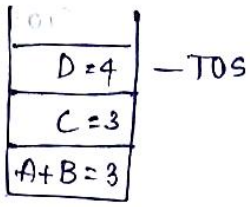
Step:3 Add (A+B)



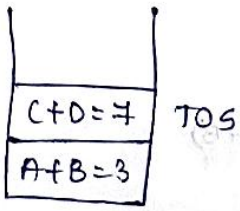
Step:4 Push C into the stack



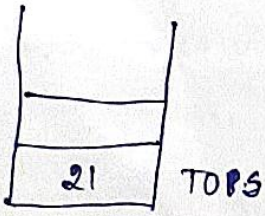
Step:5 Push D into the stack



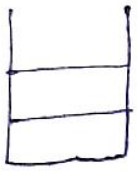
Step:6 Add (C+D)



Step:7 MUL (A+B)*(C+D)



Step: 8 POP (A+B) * (C+D)



finally this top is stored in Address of x m[x]

25/09/21

Addressing Modes:

Each and Every addressing mode definately followed by operands and instruction formats. Addressing mode mostly used for to locate the node (or) device.

Instruction cycle:

Instruction cycles are divided into three phases:

1. fetch instruction from memory → to collect the instructions from the memory to execute the program.
2. Decode the instruction → Here, system can convert the data into binary form.
3. Execute the instruction → Execute the program.

→ Types of addressing modes:

1. Implied addressing mode
2. Immediate Addressing mode
3. Register Addressing mode
4. Register indirect Addressing mode
5. Auto increment (or) Auto-decrement Addressing mode
6. Direct Addressing mode
7. Indirect Addressing mode
8. Relative Addresing mode
9. Index Addressing mode
10. stack Addressing mode.

1. Implied Addressing Mode:

→ These addressing mode totally depends upon opcode.

→ Depends upon operations the data will be locked.

Eg: Zero address instruction format, one address instruction format

$x = (A+B) * (C+D)$

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 ← (A+B) * (C+D)
POP M[X] ← TOS.

2. Immediate Addressing mode:

Here operand specified in instruction itself.

Instruction

Opcode	Operand

Here, the every field of the size 16bit (total size is 32)

Operand can indirectly acts as address.

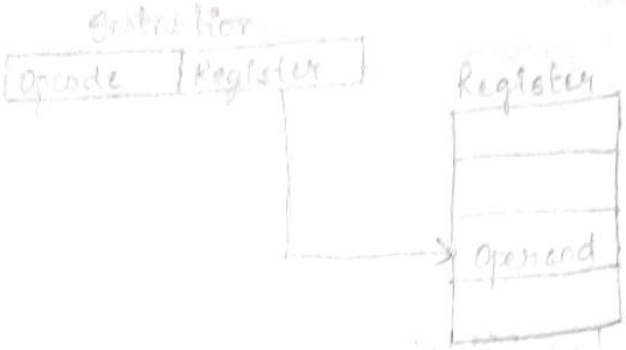
Immediate addressing mode totally depends on operand.

Advantage: no memory reference

Disadvantage: limited operand.

04/10/21

3. Register direct addressing mode:



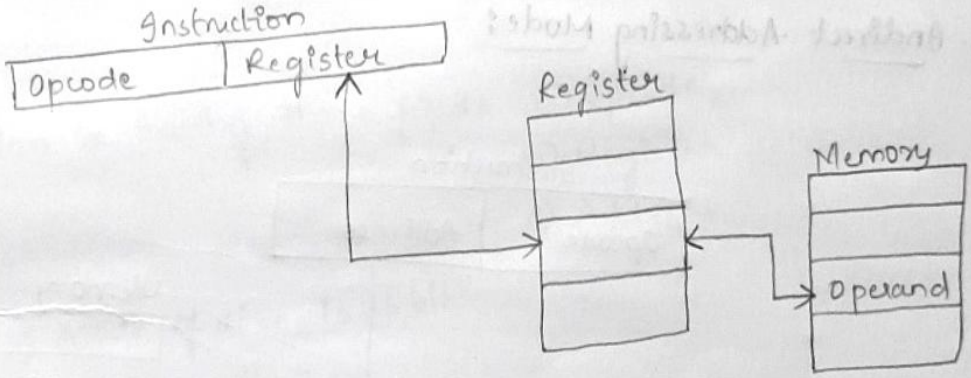
→ Here, instructions are divided into two parts: a) Opcode
b) Register

→ Here, register is linked with another register.

Advantage: no memory reference

Disadvantage: limited address space.

4. Register Indirect addressing mode:



→ Here, we are using three types of memory [Indirectly].

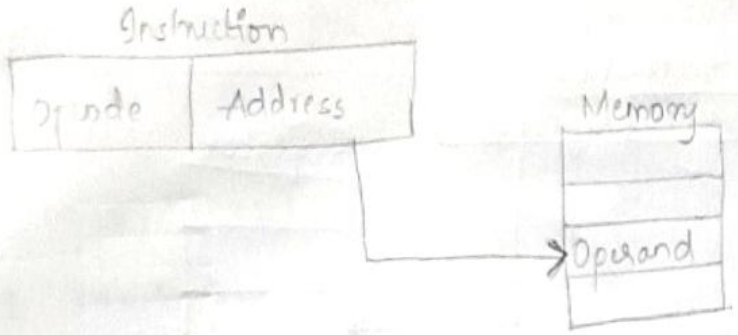
First we use memory of data, if memory of database is full then we use register, if register of database is full then we use instruction of register.

Advantage: Large address space.

Disadvantage: Extra memory reference

06/10/21

5. Direct Addressing Mode:



→ Instruction having two fields:
a) opcode
b) Address

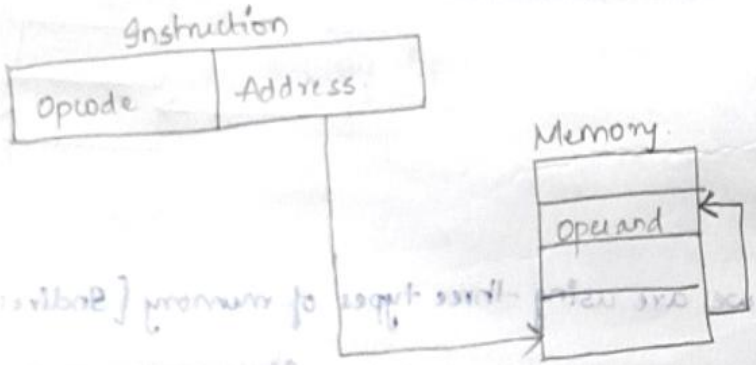
→ Here address directly assigned to the memory of operand.

→ Direct addressing mode of size is 16 bits.

Advantage: Simple (address is directly assigned to memory)

Disadvantage: limited address field

6. Indirect Addressing Mode:



→ Instruction having two fields a) opcode
b) Address

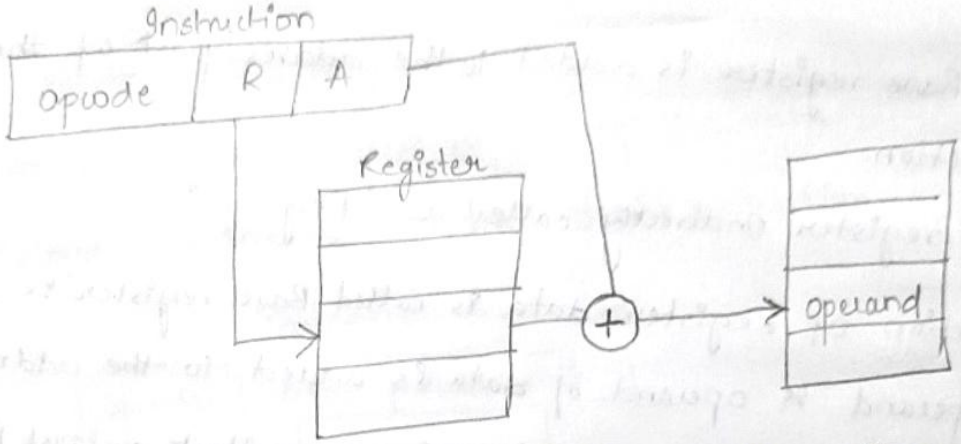
→ Here Address is assigned to empty field of memory and empty field of ~~memory~~ address and address is moved to operand

of memory.

Advantage: Flexibility

Disadvantage: Complexity.

7. Displacement Addressing Mode:



→ A very powerful mode of addressing combines the capabilities of direct addressing and register indirect addressing. The address field of instruction is added to the content of specific register in the CPU.

→ Instruction is having three fields a) opcode
b) Register
c) Address.

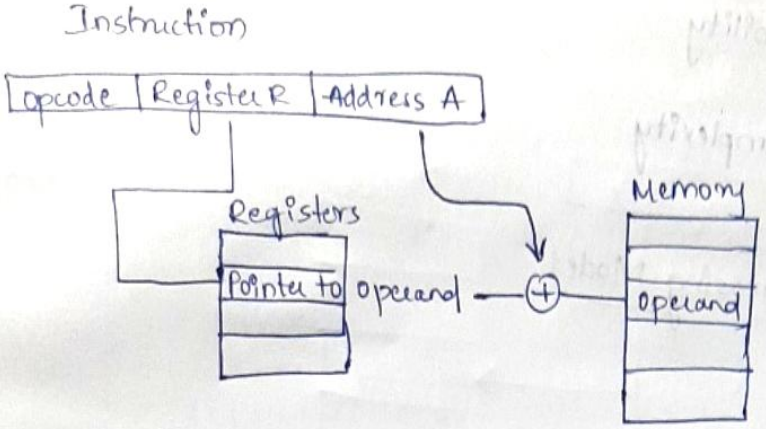
→ Addressing mode of size is 16 bits.

Advantage: Flexibility

Disadvantage: Complexity

→ Size of field is 16 bits.

8. Base Register Addressing mode:



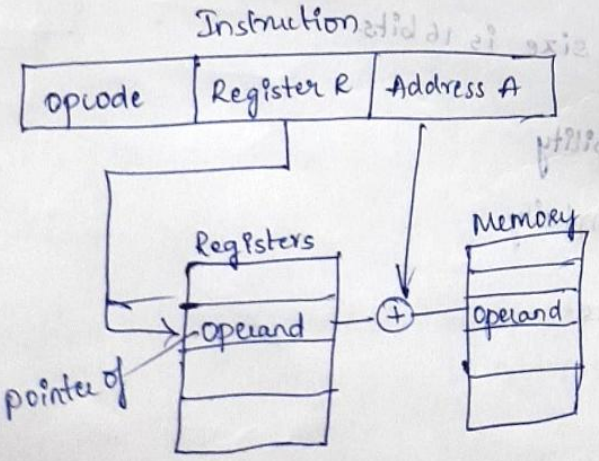
→ Here, Base register is added to the address part of the Instruction.

→ Base register Indirectly called as "Register".

→ Instruction of Register data is called Base register is stored in operand & operand of data is added to the address of instruction after addition of process that output B is stored in memory of operand.

9. Indexed Addressing Mode:-

Index Register is added to the Address part of the Instruction.



→ Here, the Index Register Indirectly is called Register.

→ Here, Index Register is added to the address part of Instruction.

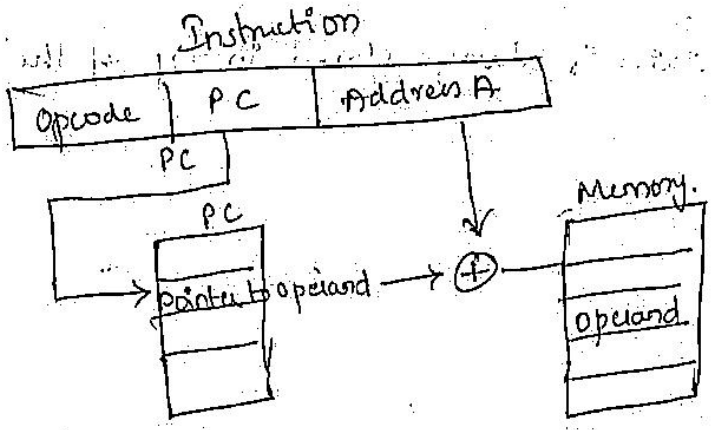
→ Instruction of Index Register is stored in operand & operand of data is added to the address of Instruction after Addition of process the output is stored in memory.

10. Relative Addressing mode:

→ Here, we are using program counter (pc)

→ program counters are used for storing the data.

→ PC is added to the address part of the Instruction.



→ Here PC is added to the address part of Instruction.

→ Here, the PC indirectly is called Register.

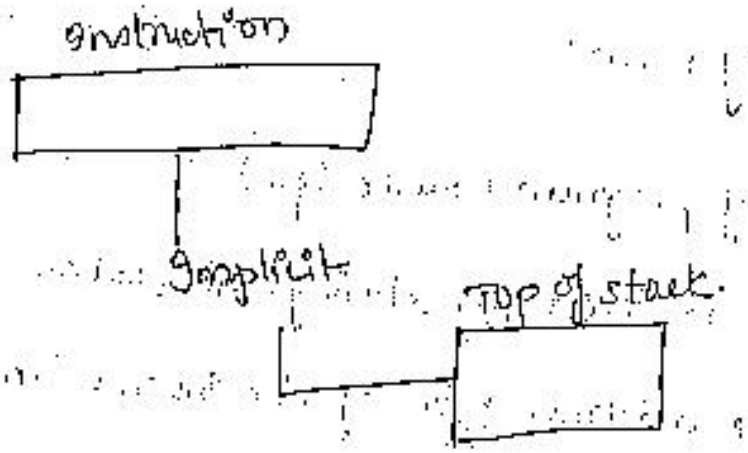
→ Instruction of PC is stored in operand & operand of data is added to the address of Instruction after Addition of process the output is stored in memory.

11. Stack Addressing modes

→ stack is a linear data structure of an address

→ Here, we are using push & pop operations.

→ Every stack followed by LIFO principle



Instruction of data is always stored in top of the stack.