Unit - 1

Computer

Computer is an electronic device that accept input, processes data, stores data and produces outputs, all according to a series of stored instruction.

Hardware → It includes the electronic and mechanical devices that process the data.

Software → A computer program that tells the computer how to perform particular task.

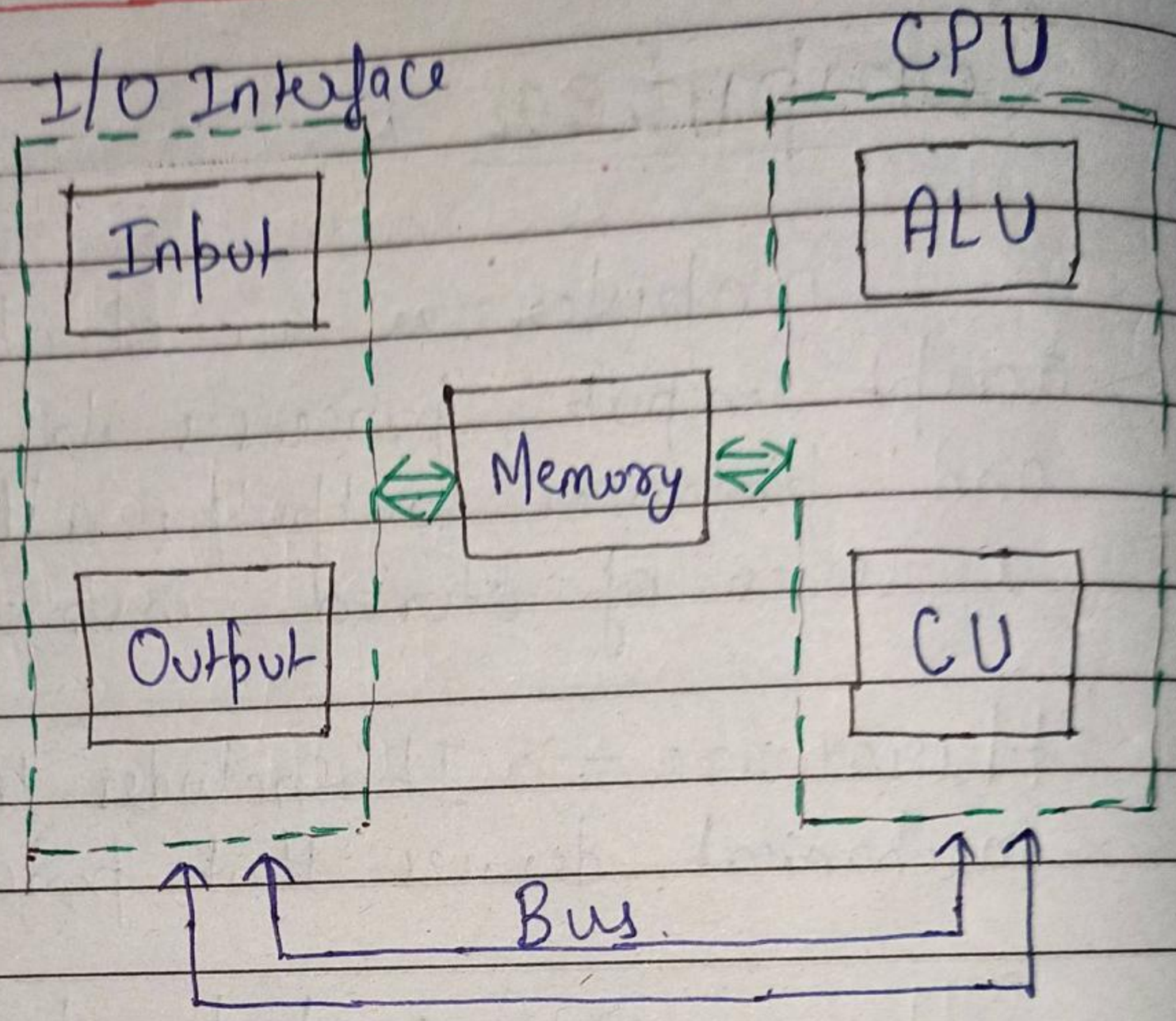
Input :- Whatever we put into a computer system.

Data :- Refers to the symbols that represent facts, objects and ideas.

Processing :- Manipulation of the data in many ways.

Output :- Consist of processing results produced by a computer.

Functional Unit of Computer



Computer consist of following functional units :-

(i) Input Unit :- Converts the external world data to a binary format, which can be understood by CPU.

Ex → Keyboard, Mouse, Joystick, etc.

(ii) Output Unit → Converts the binary format document to a format that a common man can understand.

Ex → printer, Monitor, etc.

(iii) CPU :- It is also known as brain of the computer. Central Processing Unit (CPU) performs the various tasks to process the data.

Its major componets are :-

(a) ALU
(b) CU
(c) Registers

(a) ALU → Arithmetic logic Unit performs various arithmetic operations. It also performs various logical operations.

(b) CU → Control Unit controls the operation of the CPU and hence the computer system.

(c) Register → The registers are used to store the data, addresses and flags that are in the use by the CPU.

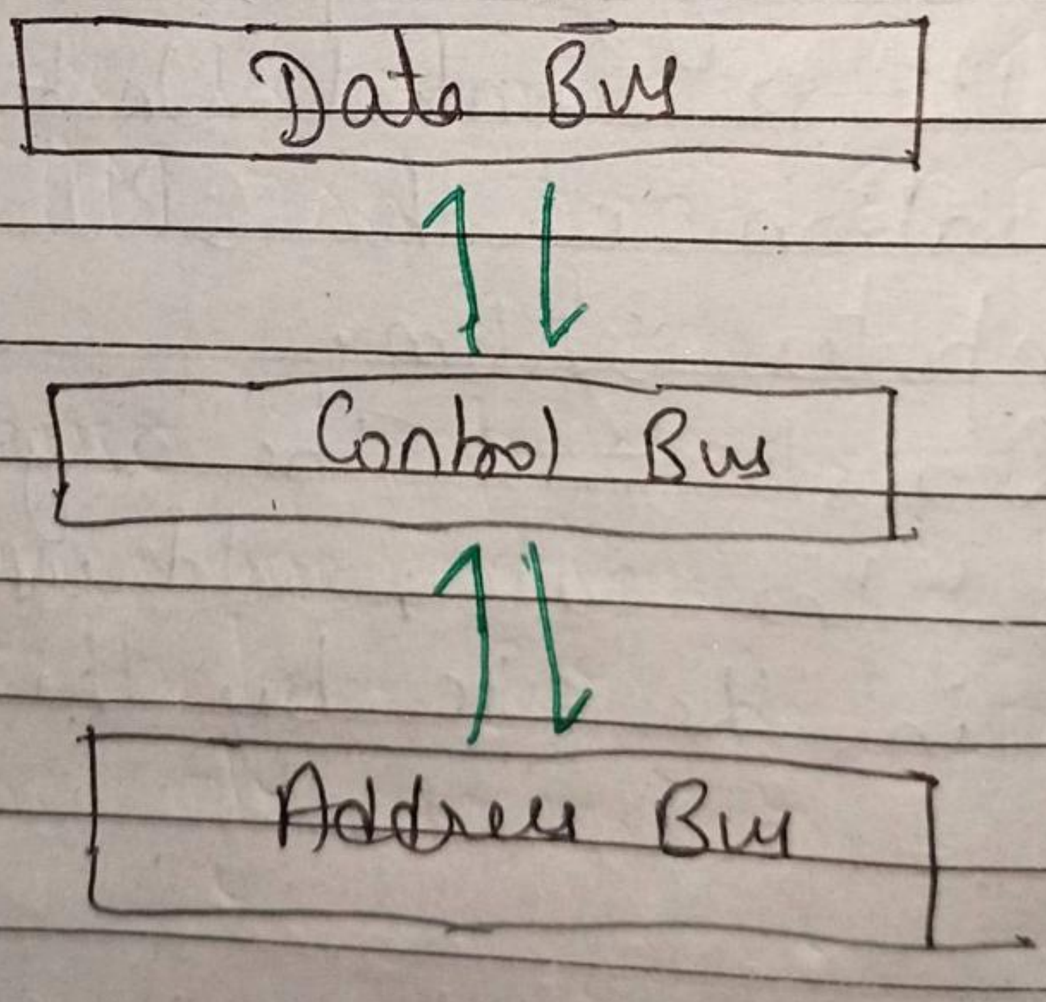
Buses :- "A bus is a communication pathway containing two or more devices". A key characteristic of a bus is that it is a shared transmission medium.

The communication between the external environment and the CPU is established through System Bus.

Types of Bus :-

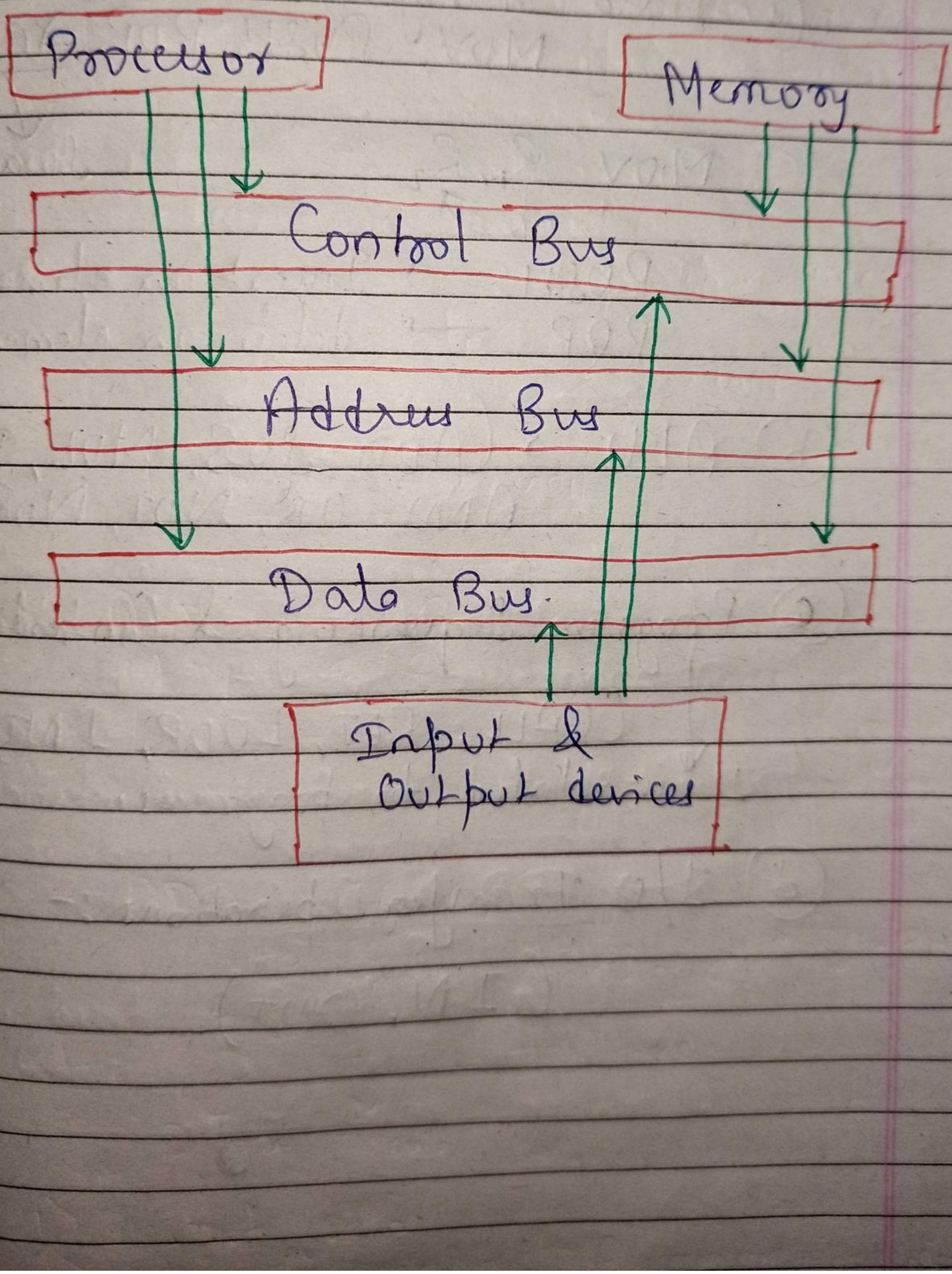
(a) Data Bus → stores data
(b) Control Bus
(c) ~~System~~ Address Bus → tells address of the data of

It is used for controlling of operation. Like it will tell whether any task should be given to ALU or registers.



All of these 3 are inter-related to each other.

Diagram to Represent Bus Structure



Memory Unit :-

Memory is basically a large array of bytes. The main function of memory unit is to store the information needed by the system.

Type of Memory (m/m). :-

(a) Primary Memory
(b) Secondary Memory
(c) Cache

(a) Primary Memory → It is also known as Main memory, which operates at electronic speed. CPU can directly access the program stored in main memory. RAM is an example of main memory.

* RAM → Volatile
  → Stores data only when system is ON
  → Deletes data when we OFF the system

(b) ~~Pri~~ Secondary Memory → It is the external storage which is much slower than the main memory. It stores program large data files which is not regularly used by the CPU.
Ex → CDROM, USB, DISK, Floppy Disk, etc.

(c) Cache → It is the temporary memory created while using the system.

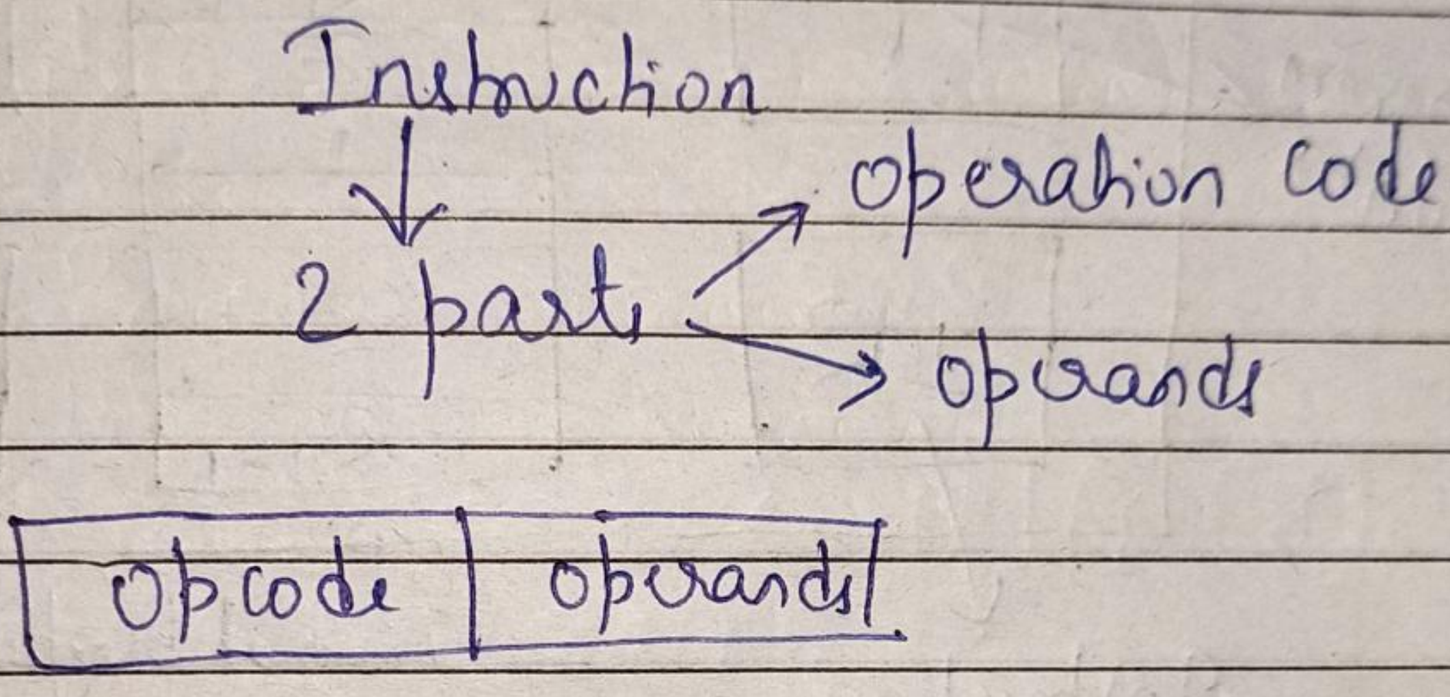
The performance of system get affected if speed difference between processor and main memory is significant. This can be improved by placing a small fast acting buffer memory between processor and main memory. This buffer memory is called cache.

* Why we see 1GB, 2GB, 4GB, 8GB ... of RAM and don't 3GB, 5GB, 7GB?

⇒ Because computer understands Binary Code only. which has 2 digits only. So the space available can be of order of 2^n only.

Basic Operational Concept :-

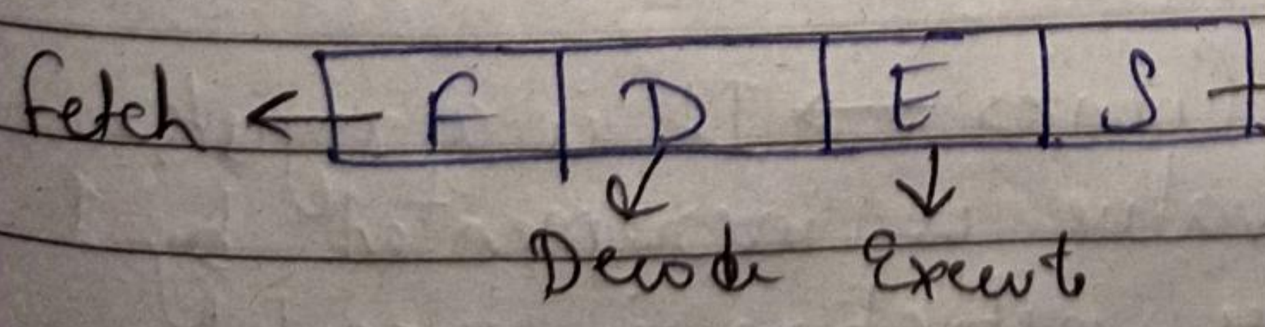
Instructions :- Whatever we give to the computer to perform the particular task is called Instruction.



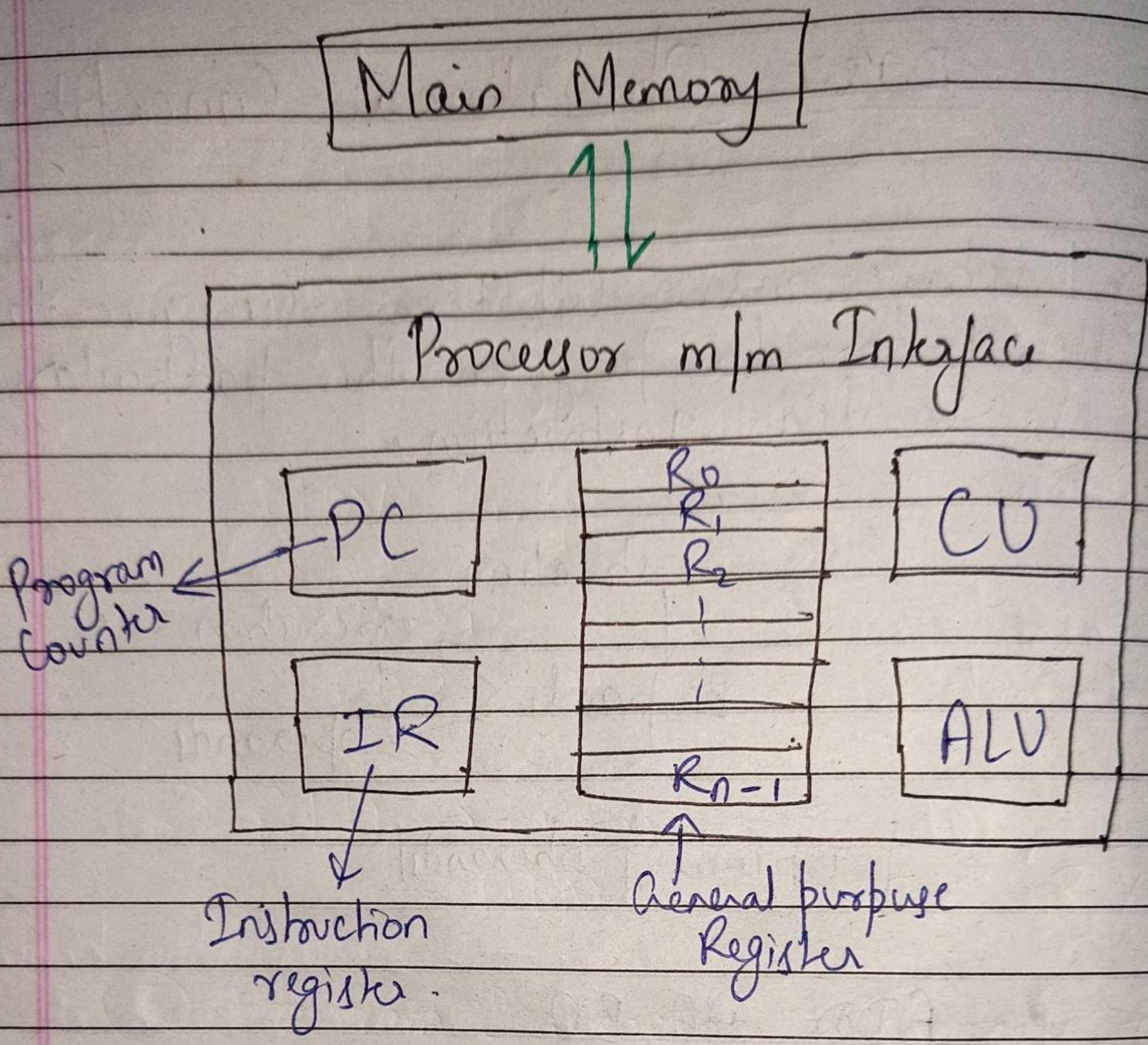
ADD A0, B0
MOV 2, 3
LOAD A

ADD LOCA, R0

Content
Content of R_0 → Registers.



1) Instruct fetch
2) Fetch operands at LOCA from mainmemory to
3) Add memory operands to content of R_0. processor.
4) Store result in R_0.



Relation :- Memory gives data to the processor and after processing is completed it also get data back from the processor.

* Program Counter → It is a register whose value is incremented by 1 when any instruction is executed.

* Instruction Register → IR holds the ~~information~~ instruction which is currently being executed.

4 types of Operations :-

(a) Data transfer :-
MOV, PUSH, POP, XCHG
↓
Swap.

MOV R_1, R_2

PUSH ⟶ Insert an element
POP ⟶ delete an element

(b) ALU ⟶ (ADD, SUB, MUL, DIV, AND, OR, NOT, NOR, XOR)

(c) Program Sequencing & Control :-
↓ (CALL, RET, LOOP, INT)
It is related to programe.

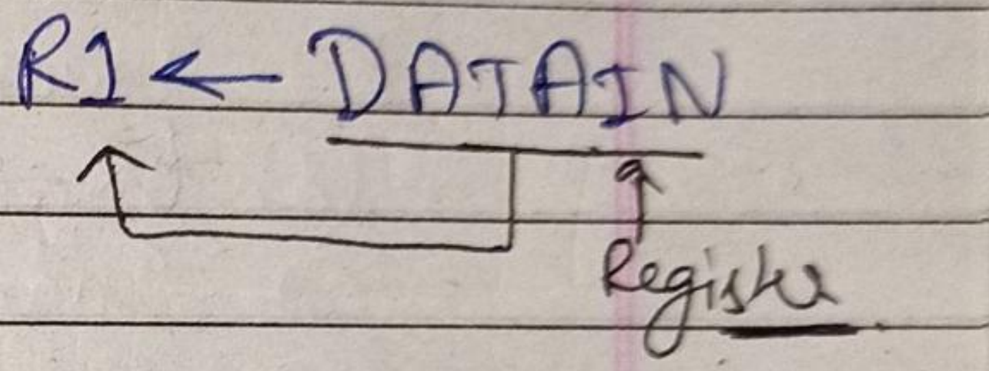
(d) I/O Transfer operations :-
(IN, OUT).

Register Transfer Notation : (RTN)

Assembly - m/m

Move LOC, R1 ≡ R1 ← [LOC].

Add R_1, R_2, R_3 ≡ [R3] ← [R1] + [R2]



V.V. Imp

Instruction Type :-

(a) Zero Address :- In this there is no operand present and only opcode is there.

Format :- Opecode

Eg. PUSH, POP.

(b) One - Address :- In this there is one operand present.

Format :- opcode Source / destination

Ex → LOAD A
ADD B
Store C.

ADD B

↓

By this, B is added to the in-built accumulated register (AC register)

(c) Two Address :- In this, the two operands is present.

Format :- operand Source, Destination.

Ex → ADD A, B
SUB A, B
MUL A, B

(d) Three Address :- In this, the three operands is present

Format :- Opcode Source1, Source2, destination

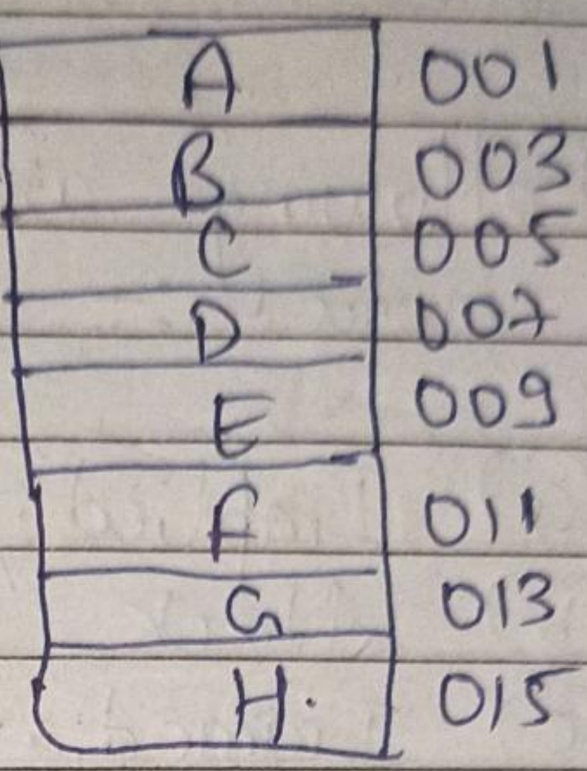
Ex → ADD A, B, C.

↑

In this A+B is stored in C.

Memory Location & Address :-

CPU



001 → m/m location
A → Content of 001

Address → Unique location of the data.

m/m Operation :- (a) Read
Read data of m/m ←
Writ data in m/m ← (b) Writ

* MDR → Memory Data Register is the data register which is used to store data on which operation is being performed

* MAR → Memory Address Register is the address register which is used to store the address of memory location where the operation is being performed.

V.V. Imp

What are various addressing Modes?

Following are the 12 addressing modes :-

(a) Implied / Implicit Addressing mode
(b) Stack Add. mode
(c) Immediat Add. mode
(d) Direct Add. mode
(e) Indirect Add. mode
(f) Register Direct Add. mode
(g) Register Indirect Add. mode
(h) Relative Add. mode.
(i) Indexed Add. Mode
(j) Base Register Add. Mode
(k) Auto Increment Add. Mode
(l) Auto - decrement Add. mode

(a) Implied / Implicit Addressing Mode :-
In this, the operand is specified implicitely.

(b) Stack Addressing mode :-

In this, the operand is contained at the top of the stack.

PUSH, POP, ADD.

ADD in stack means add two element of the top of stack.

(c) Immediate Addressing Mode :-

In this, the operand is specified explicitly.

opcode	operand

ADD R_1

MOV R, #20 → Data

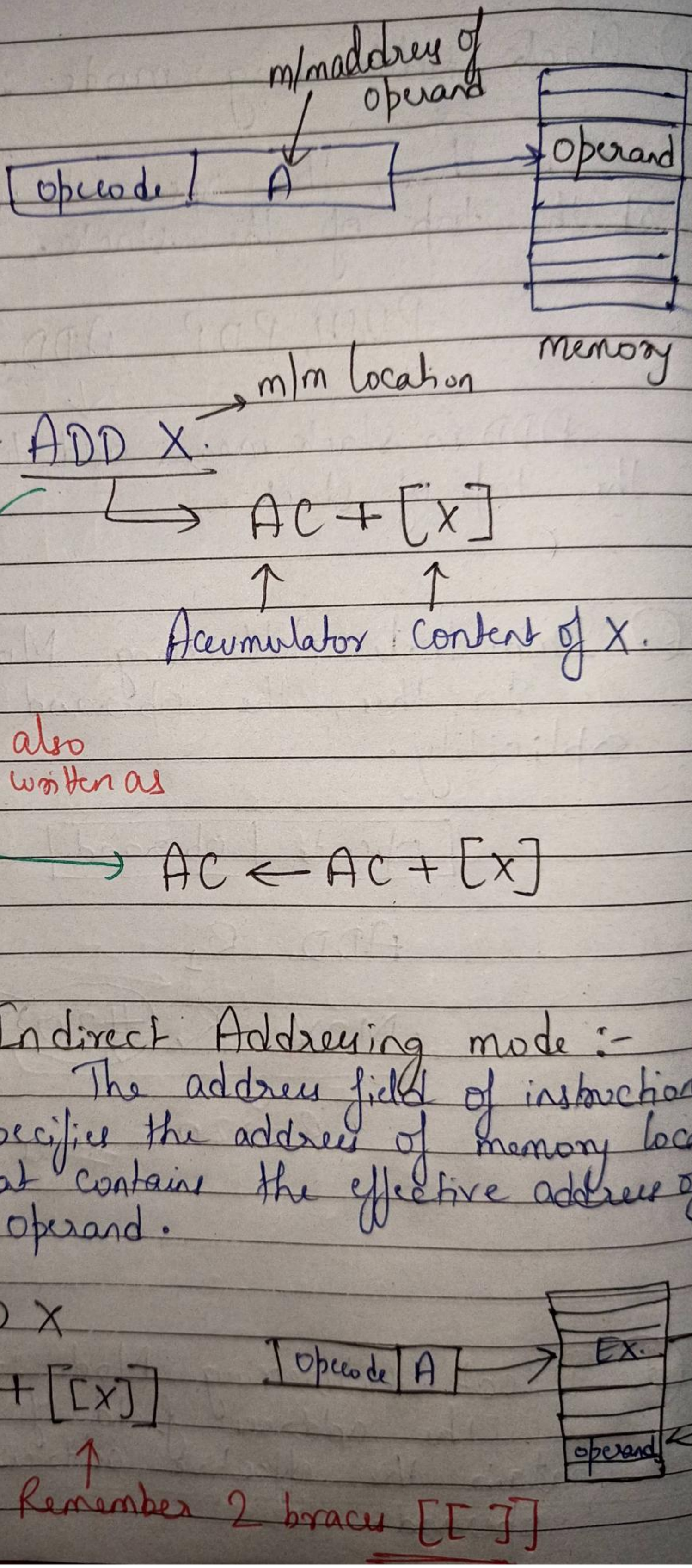
R → 20

is initializing a register.

means it is data not location

(d) Direct Addressing Mode :-

The address field of the instruction contain the effective address of the operand.

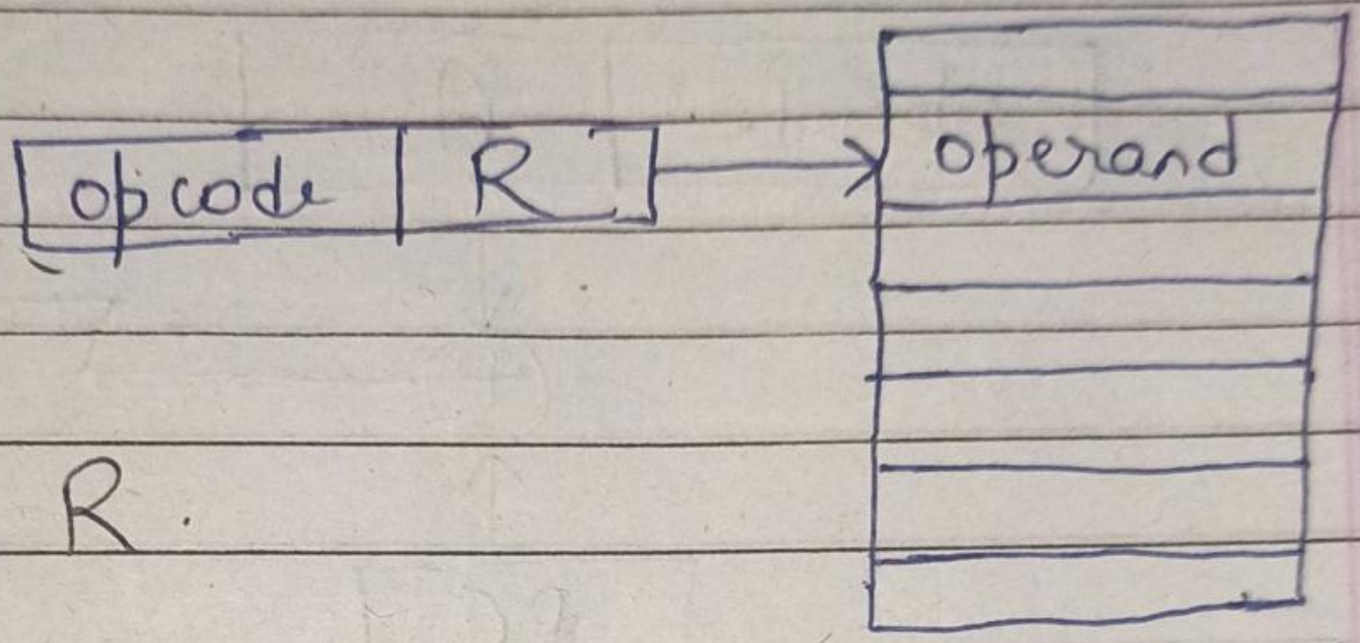


(d) Indirect Addressing mode :-

The address field of instruction specifies the address of memory location that contains the effective address of operand.

(f) Register Direct Addressing Mode :-

The operand is contained in register set. The address field of the instruction refers to CPU register that contains the operand.

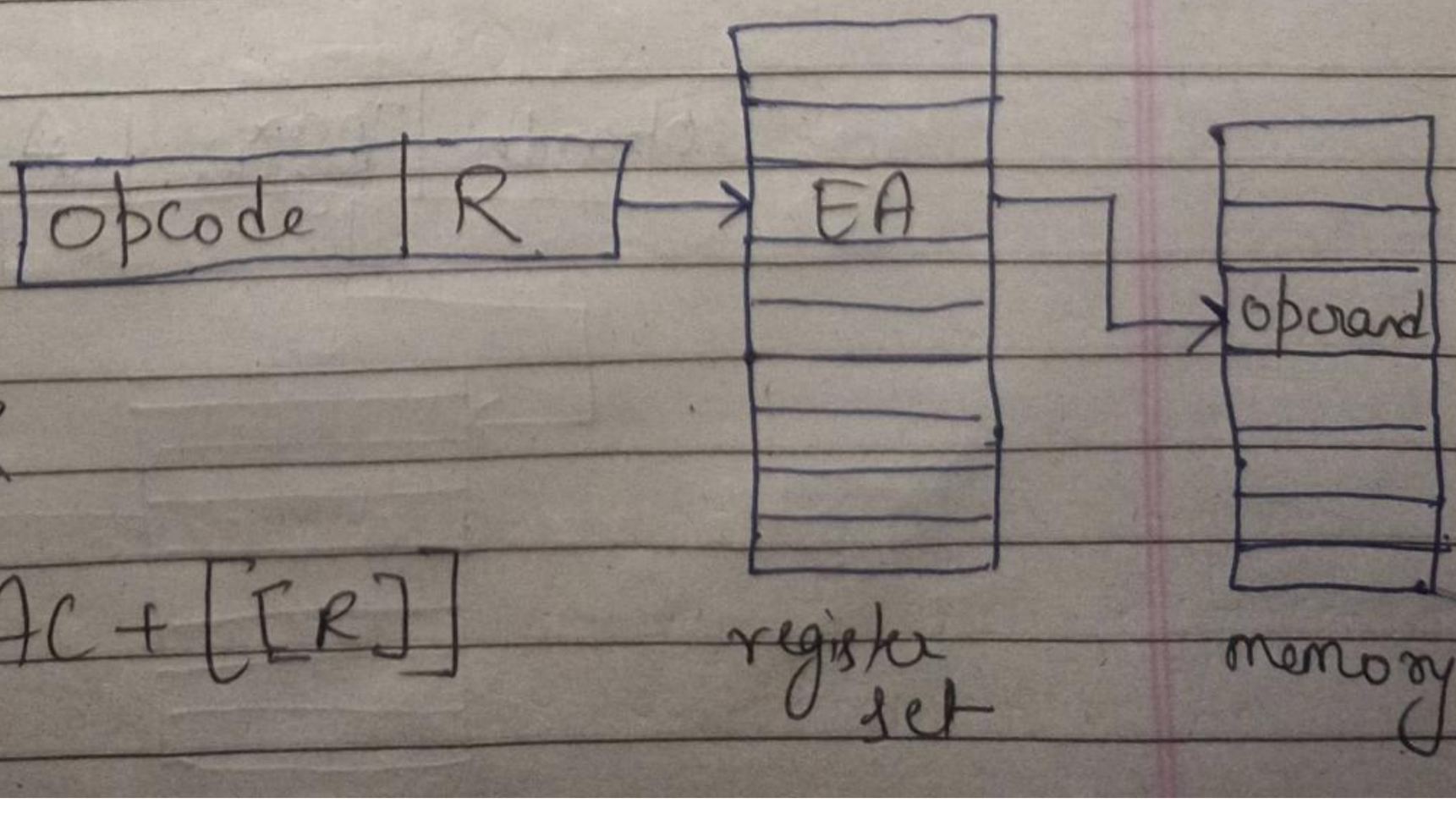


ADD R.

AC ← AC + [R]

(g) Register Indirect Addressing Mode :-

The address field of instruction will refers to the CPU register that contain effective address of operand.

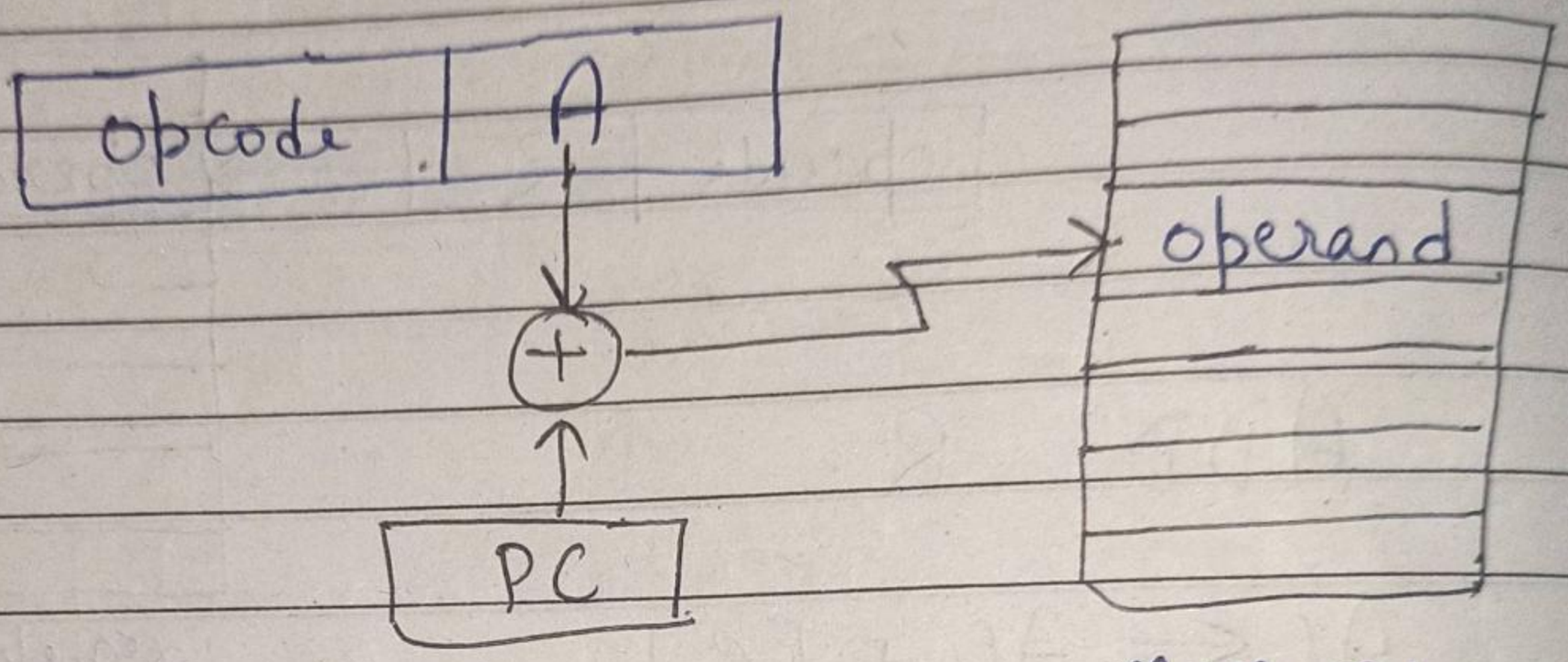


ADD R

AC ← AC + [[R]]

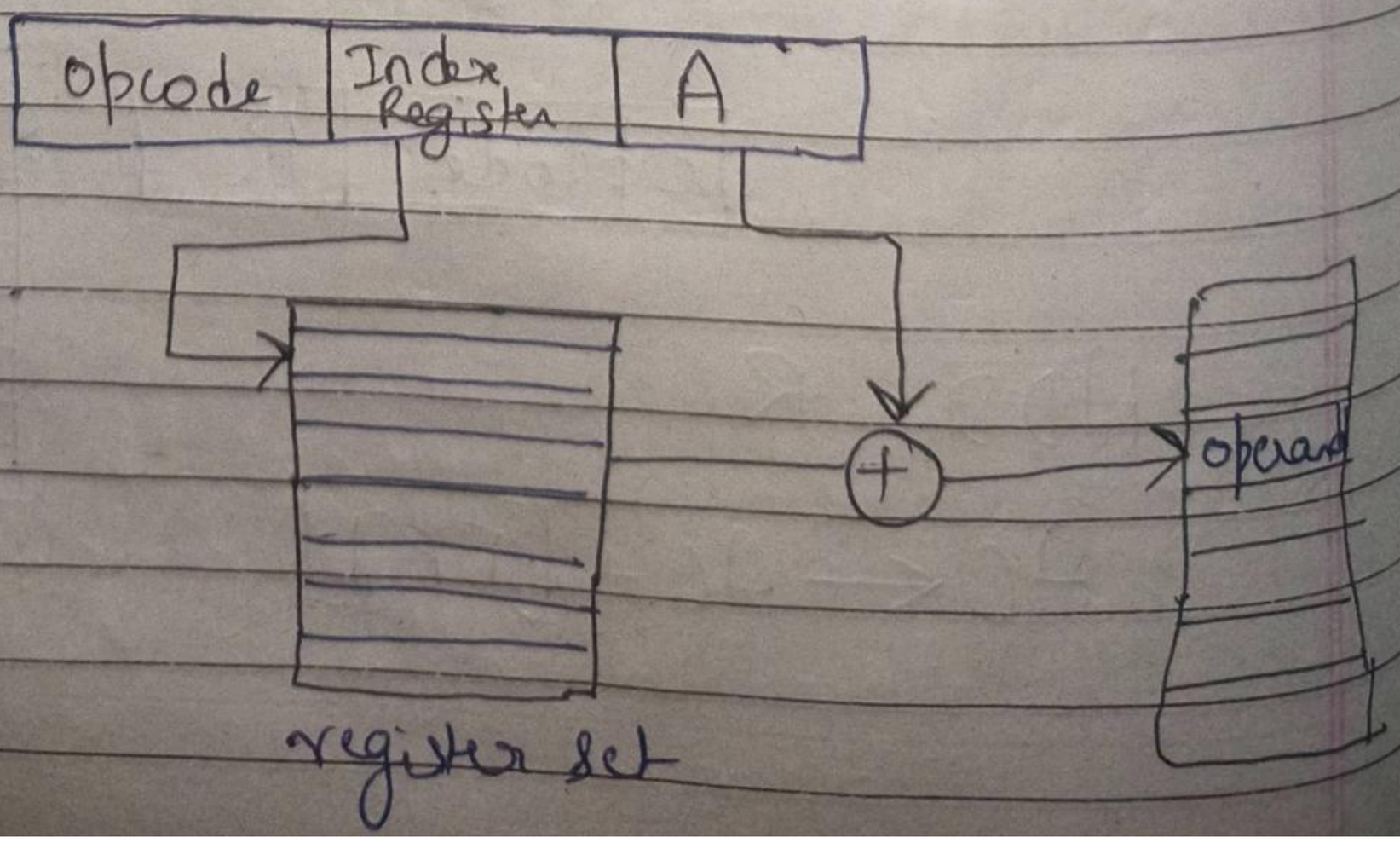
(h) Relative Addressing Mode :-

EA = Content of PC (Program Counter) + Address part of instruction.



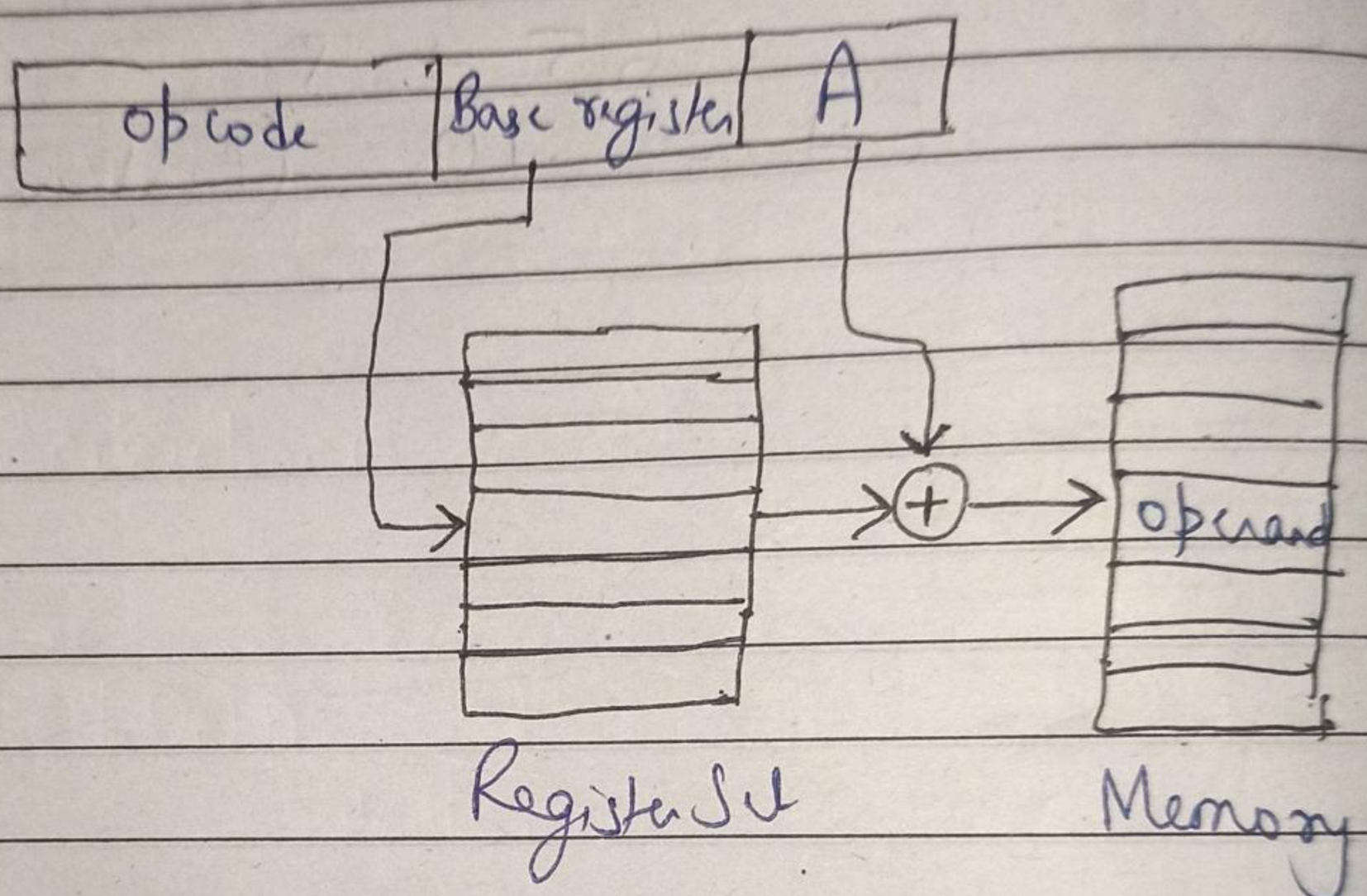
(i) Indexed Addressing Mode :-

EA = Content of Indexed register + Address part of instruction.



(j) Base Register Addressing Mode :-

EA = Content of Base register + Address part of Instruction.

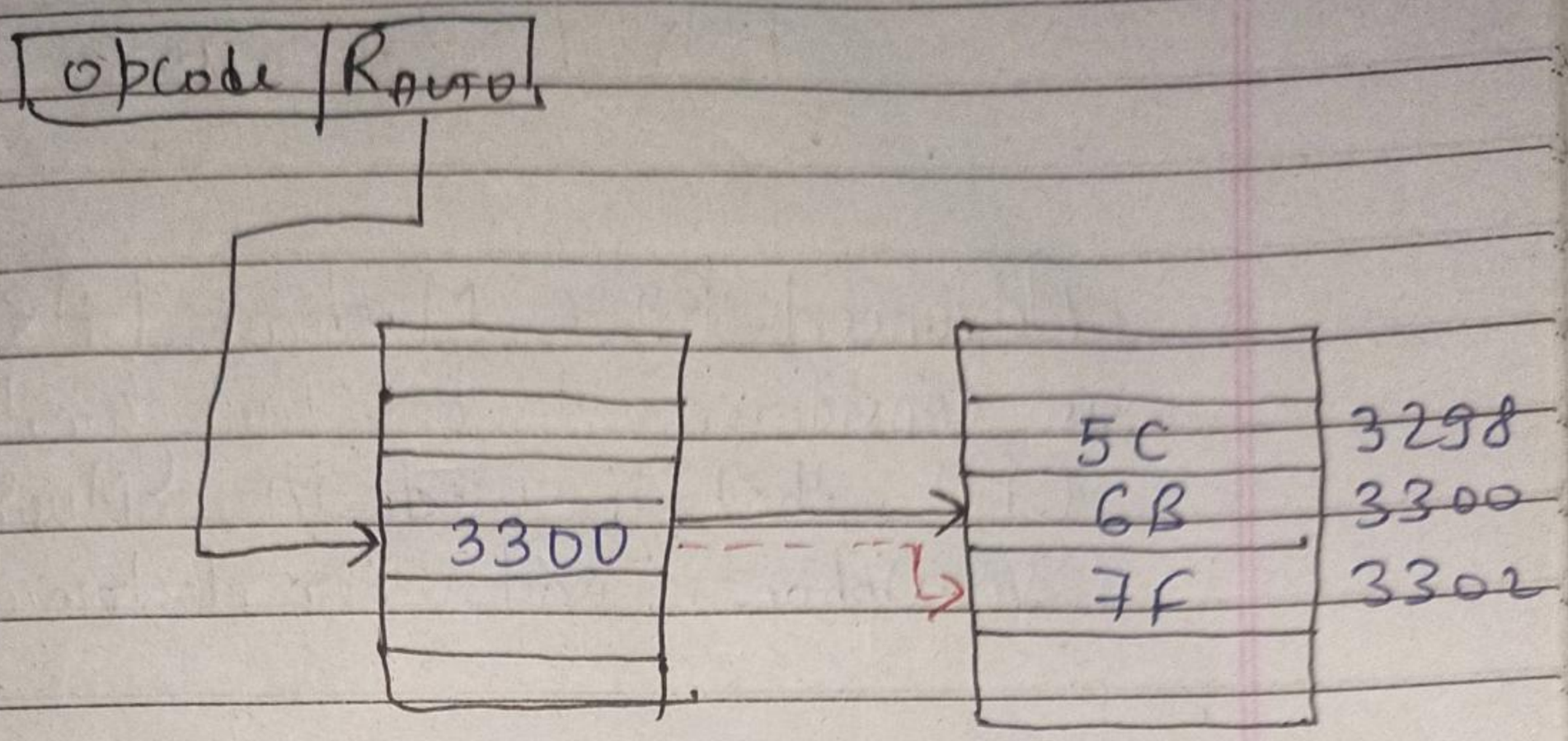


(k) Auto-Increment Addressing Mode :-

Effective Address of operand = Content of register.

In this addressing Mode, after the operand access the content of the register is automatically increment by step size

* Step Size ('d') depends on size of operand.

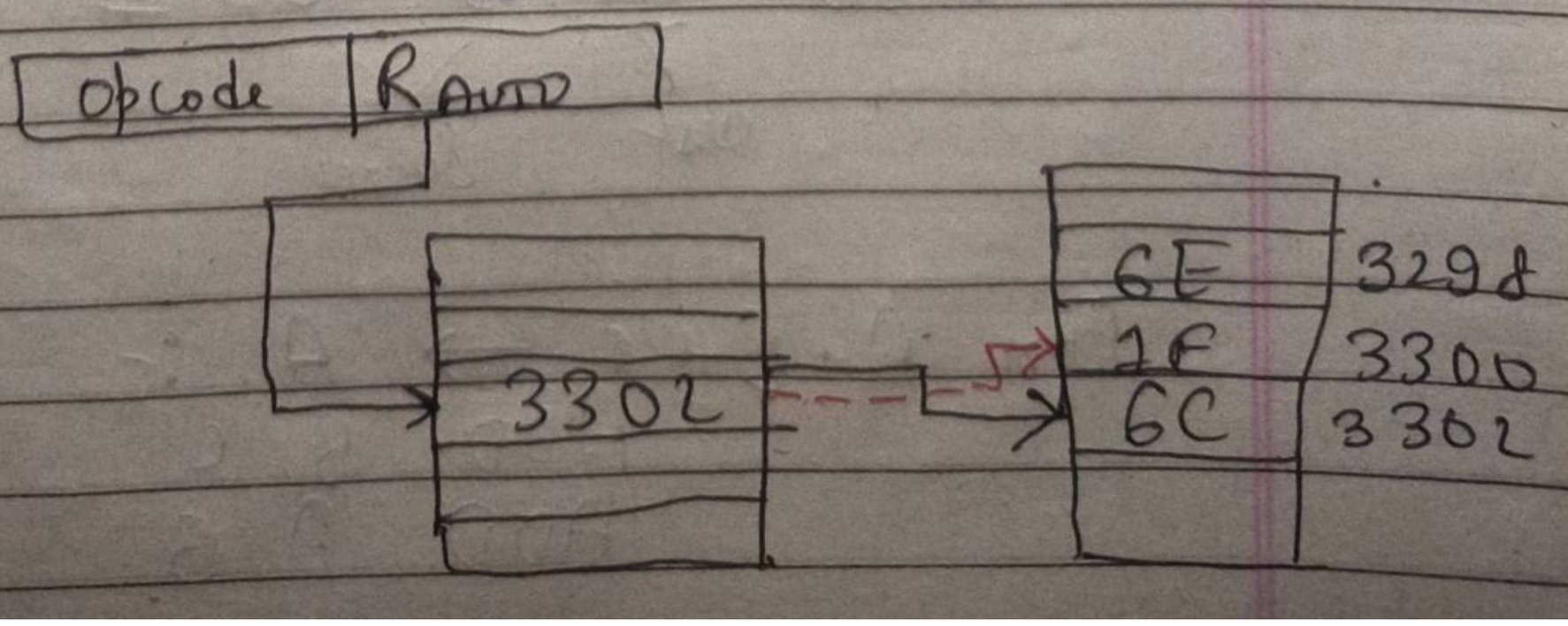


In this, first operand is fetched Then the instruction register R_{AUTO} value is increamented by step size 'd'.

(i) Auto-Decrement Addressing Mode :-

Effective address of operand = Content of Register − Step size.

* In this addressing mode, first the content of register is decremented by step size. After decrementing the operand is read.



Application of Various Addressing Mode:-

1) Immediate Addressing Mode :-
To initialize register to a constant value.

2) Direct Addressing Mode &
Register Direct Addressing Mode :-
To access static data & to implement variables.

3) Indirect Addressing Mode &
Register Indirect addressing Mode :-
To implement pointers because pointer are memory location that store the address of another variable.
To ~~int~~ pass array as a parameter. because array name is base address and pointer is needed to point address.

4) Relative Addressing Mode :-
For program relocation at run time.
To change the normal sequencing of execution of instruction.
For branch type instruction since it directly update the program counter.

5) Index Addressing Mode :-
For array implementation or array addressing
For record implementation.

6) Base Register Addressing Mode
For writing relocatable code i.e
for relocation of program in memory
even at run time.
For handling recursive procedure

7) Auto-increment Addressing Mode &
Auto-decrement Addressing Mode
For implementing loops
For stepping through array in loop
For implementing a stack as push
or pop.

ARM Processor

Advanced RISC Machine (ARM) Processor is considered to be the family of CPU that is used in Music Player, smartphones, and other electronic devices.

Instructions In ARM :-

LOAD A, B.

A ← destination
B → Source

STORE A, B

A ← source
B → dest

ADD A, B ≡ A + B → A

A ← dest
B → source

ADD A, B, C ≡ B + C → A.

A ← dest
B, C ← source

For A + B + C → A. :-

ADD B, C
ADD A, B

Feature of ARM Procuor :-

1) Multiprocessing System :-
ARM processor are used in cases of multiprocessing systems where more than one processor are used to process information.

2) Tightly Coupled Memory :-
Due to tightly coupled memory, ARM processor has very fast response time.

3) Memory Managment :-
ARM has management section which consist of Memory Management Unit and Memory Protection Unit. This help in managing memory efficiently.

4) large no. of registers
large no. of registers are used to prevent large amount of memory interaction.

5) Pipelining :-
In ARM, processing of instruction is done in parallel using pipelines. Instruction are booken down and decoded in one pipeline stage. The pipeline advances one step at a time to increase rate of processing.

Assembly Language

A processor understands only machine language instruction which are string of 0 & 1. But machine language is too ~~obscr~~ complex for using in software development. So, the low level Assembly Language is designed for a specific family of processor that represents various instruction in symbolic code and more understandable form.

Basically Assembly language is mostly a thin layer above machine structure. Assembly language is converted into machine code by a ~~und~~ utility program refered to as an assembler. The conversion process is refered to as assembly or assembling the code.

Benifits of Assembly Language :-

1) Having the understanding of assembly language makes one aware of how program interface with OS, processor and BIOS. and how data is represented and how instruction accus & ~~pro~~ procus data.

2) It requires less memory & execution time.

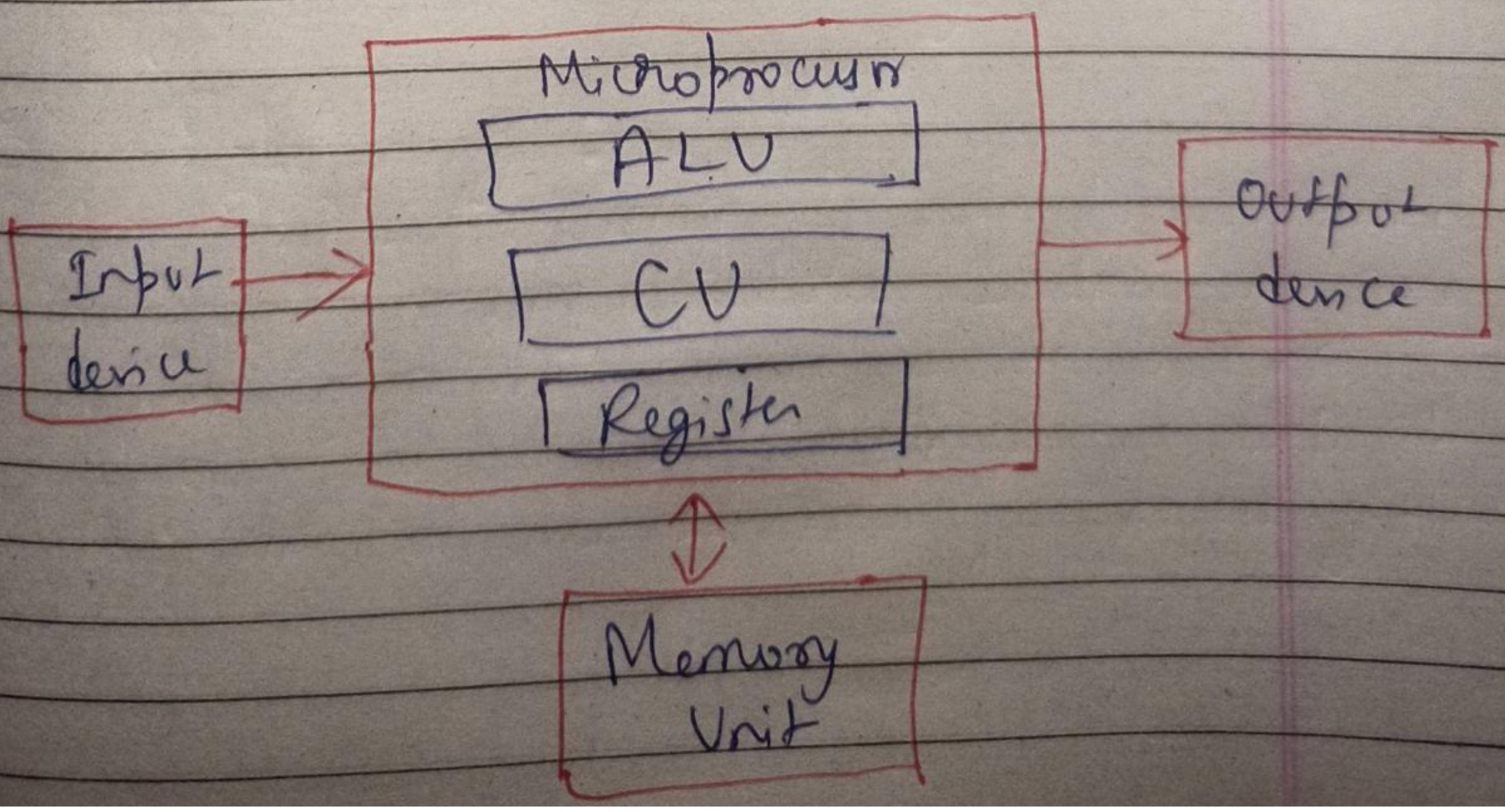
3) It allows hardware specific complex jobs is easy way.

4) It is suitable for time-critical jobs.

Microprocessor

Microprocessor is a programmable device that takes in input, performs arithmetic or logical operation over it and produces the desired output.

Microprocessor is a digital device on a chip that can fetch instructions from memory, decode and execute them and give results.



* Difference between a CPU & a microprocessor is that the microprocessor is very small compared to an ordinary CPU.

Features of Microprocessor :-

1) Low Cost
2) High speed
3) Small size
4) Versatile
5) Low power consumption
6) Less heat generation
7) Portable