Date- 6/11/19

UNIT-3

HARDWIRED CONTROL & MICROPROGRAM CONTROL -

There are two majors types of control organization - hardwired control & microprogram control.

In the hardwired control organization, the control logic is implemented with logic gates, flip flops, decoders, & other digital circuits. It has the advantage that it can be optimized to produce a fast mode of operation. A Hardwired control as the name implies requires changes in the wiring among the various components if the design has to be modified or changed. A hardwire ~~control unit~~ for

In the microprogram organization the control information is store in a control memory. The control memory is programmed to initiate the required sequence of microoperations. In the microprogrammed control, any required changes or modification can be done by updating the micro-program in control memory.

Hardwired Control comes under Reduce instruction set collection & microprogram comes under complet ISC.

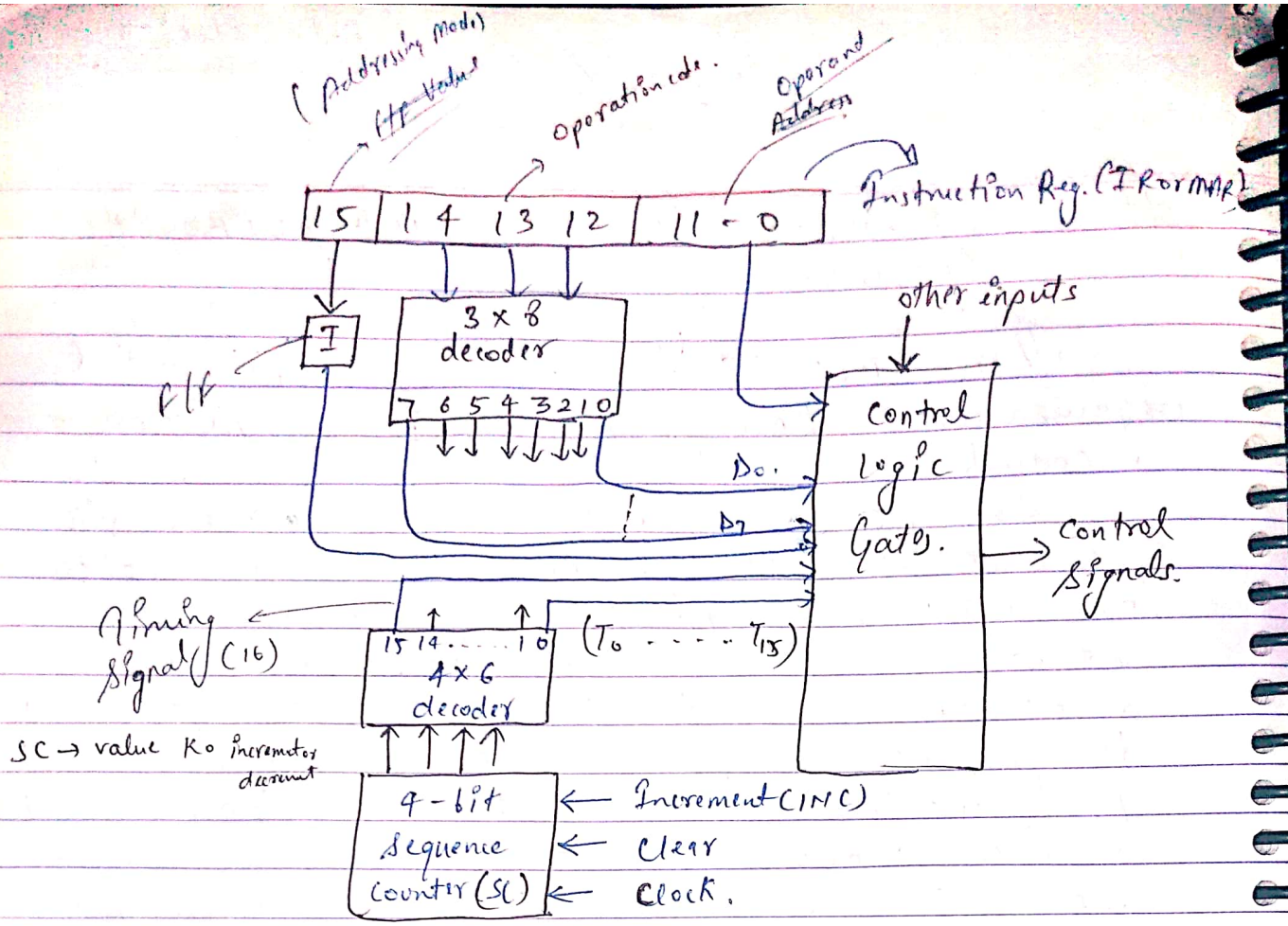
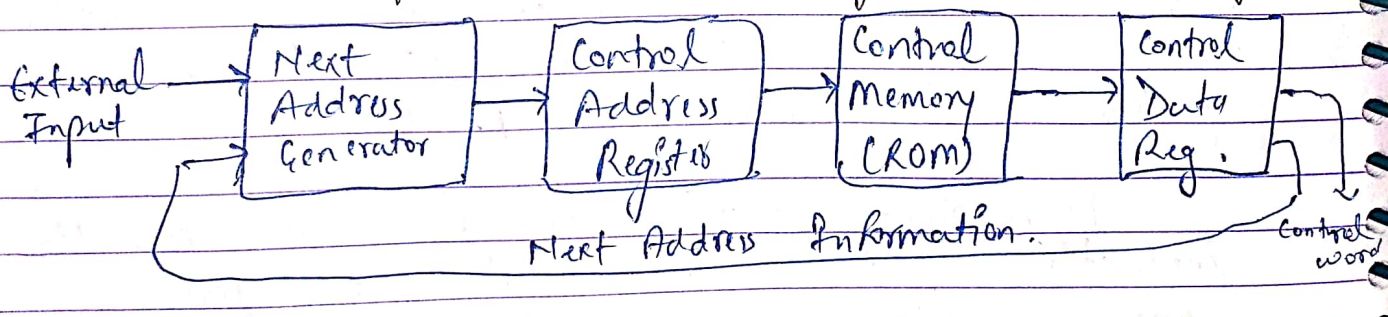


Fig: Block Diagram of Hardwired Control Unit for basic Computer.

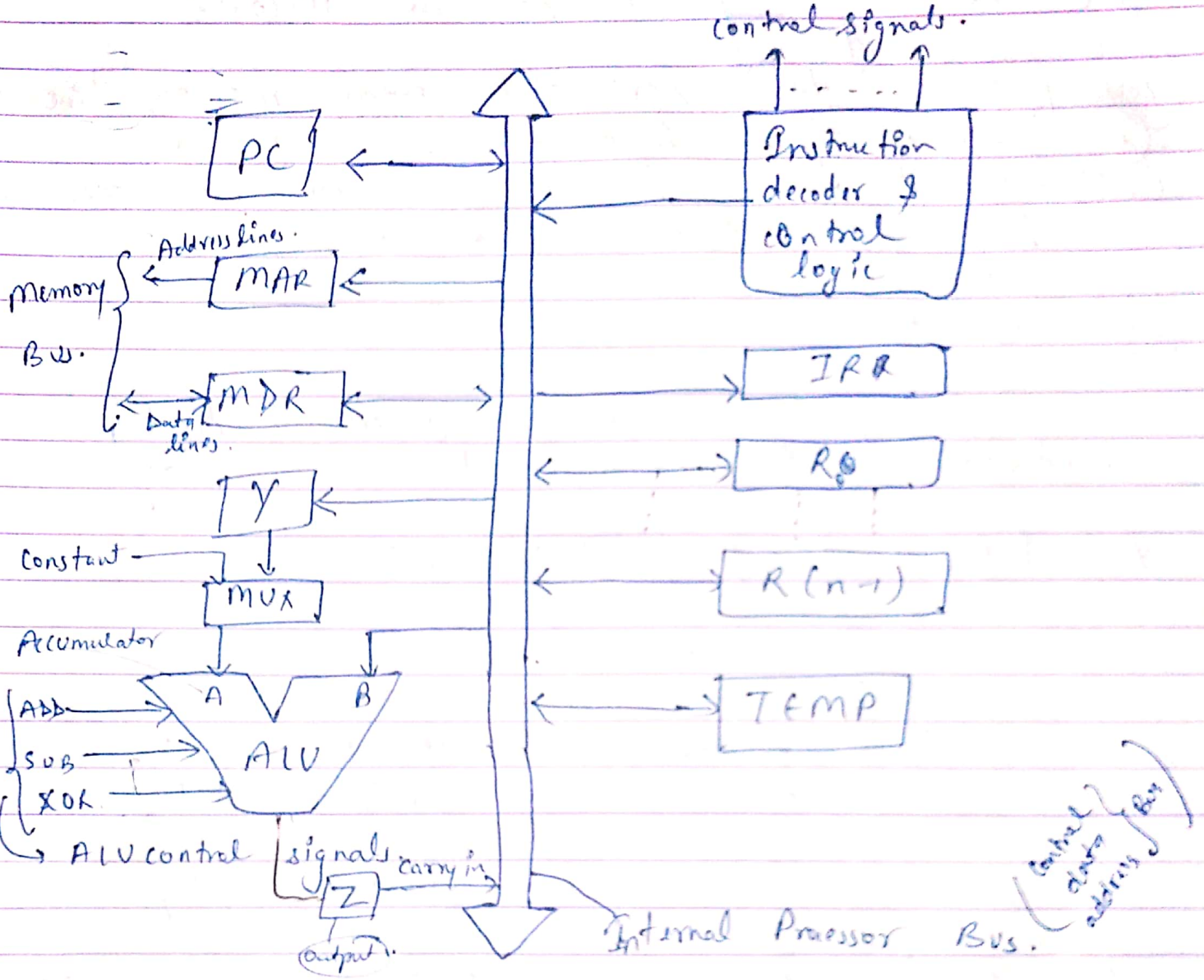
ex-

$D_4 T_4$: SC ← 0.
T_0 : AR ← PC

* Block Diagram of Micro programmed control organization



SINGLE BUS ORGANIZATION



INSTRUCTION CYCLE -

A program residing in the memory unit of the computer consist of a sequence of instructions. The program is executed in the computer by going through a cycle for each instruction. Each instruction cycle is sub-divided

into a sequence of sub-cycles or phases. The following phases are as follow.

(i) Fetch the an instruction from memory (Fetch cycle).
(ii) Decode the instruction. (Decode cycle).
(iii) Read the effective address from memory if the instruction has an indirect address.
(iv) Execute the instruction. } Execution cycle.

This process continues unless a HALT instruction is encountered.

* FETCH & DECODE Cycle -

T_0 : AR ← PC
T_1 : IR ← M[AR], PC ← PC + 1
T_2 : $D_0, D_1, \ldots D_7$ ← Decode IR(12, 13, 14),
AR ← IR(0 ... 11), I ← IR(15)

During fetch cycle following steps are performed -

(i) To transfer the content of PC into the MAR or AR. The control signal does this by activing the control signal that opens the gates b/w the bits of PC & the bits of MAR or AR.

(ii) To read a word from memory into the MBR & increment the PC. The control unit does this by the

following control signals -

(a) A control signal that open gates & allows the content of the MAR to be placed onto the address bus.

(b) A memory read control signal on the control bus.

(c) A control signal that opens gates & allows the content of the data bus to be stored in the MDR.

(d) A control signal to logic that add one to the content of the PC & Stored the result back to the PC.

(iii) To read the instruction from MDR into the register IR. the control unit sends a control signal that opens gates b/w the MDR & IR.

* $R_{i_{in}} = 1$ or $R_{i_{in}} \longrightarrow$ -

(a) Data on the bus are loaded into R_i.

(b) * $R_{i_{out}} = 1$ or $R_{i_{out}} \longrightarrow$

Contents of register R_i are placed on the bus.

* $R_{i_{out}} = 0$ or ~~$R_{i_{out}}$~~ → .

for can be used for transferring data from other registers.

ex- $R1 \leftarrow R4$.

~~$R4_{in}$~~ $R4_{out} = 1$.

$R1_{in} = 1$.

ex- $R1 \leftarrow R2 + R3$.

Sequence of control signal are -

1. $R2_{out} = 1$, $Y_{in} = 1$.
2. $R3_{out} = 1$, Select Y, Add, $Z_{in} = 1$.
3. $Z_{out} = 1$, $R1_{in} = 1$.

* Memory Read Operation (Control Signal for)

1. $R_{i_{out}}$, MAR_{in} , Read,
2. MDR_{in} , WMFC
3. MDR_{out} , $R_{i_{in}}$.

where WMFC (wait) is the control signal that causes the processor to wait for the arrival of the MFC (memory function completed) signal.

* Storing a word in memory

Let the content of R1 is to be stored in location X.

1. Xout , MARin.
2. R2out , MDRin = 1 , Write.
3. MDRout, WMFC.

EXECUTION OF A COMPLETE INSTRUCTION

ADD [M], R1 : R1 ← R1 + [M].

1. PCout , MARin , Read, Select (constant), Add, Zin.
2. Zout , PCin , Yin , WMFC
3. MDRout , IRin.
4. [X]out , MARin , Read
5. R1out , Yin , WMFC.
6. MDRout , Select Y, Add, Zin
7. Zout , R1in , End.

From Step 1 to 3 Fetch Cycle.
From Step 4 to 7 execution Cycle.

ex-Q Write the sequence of control steps required for the single bus organization structure for the following instructions.

(a) Add the immediate number to the content register R1.
(b) Add the content of memory location X to the content of register R1.
(c) Add the content of memory location whose address is at memory location X to the content of register R1.

Solutions:- (a) ADD R1, 25 : R1 ← R1 + 25.

(b) ADD R1, x : R1 ← R1 + M[x].

(c) ADD R1, x : R1 ← R1 + M[M[x]].

(a) Single Bus Organization Structure –

Fetch cycle
STEP-1 PCout, MARin, Read, Select constant, Add., Zin.
-2 Zout, PCin, WMFC.
-3 MDRout, IRin.

Execution cycle
-4 PCout, MARin, Read, Select constant, Add, Zin.
-5 Zout, PCin
-6 R1out, Yin, WMFC
-7 MDRout, Select Y, Add, Zin.
-8 Zout, R1in, End.

(b) STEP 1 - PCout, MARin, Read, Select constant, Add, Zin.
2 - Zout, PCin, WMFC.
3 - MDRout, IRin.
4 - PCout, MARin, Read, Select constant, Add, Zin.
5 - Zout, PCin, WMFC
6 - MDRout, MARin, Read.
7 - R1out, Yin, WMFC
8 - MDRout, Select Y, Add, Zin.
9 - Zout, R1in, End.

(c) STEP 1 - to Step 6 are same as b.
7 - MDRout, MDRin, Read.
8 - R1out, Yin, WMFC.
9 - MDRout, Add, Zin.
10 - Zout, Rin, End.

Call (address) → offset.

Jmp.

BRANCH INSTRUCTION

A branch instruction replaces the content of the programe counter (PC) with the branch target address. This address is obtained by adding an offset (location) X to the updated value of the PC. Branch instruction are of two types unconditional branch or conditional branch.

* UNCONDITIONAL BRANCH-

STEP - 1 PC_{out}, MAR_{in}, Read, Select const., Add, Z_{in}.
2 Z_{out}, PC_{in}, Y_{in}, ~~WMFC~~ WMFC.
3 MDR_{out}, IR_{in}.
4 Offset - field - of IR_{out}, Add, Z_{in}
5 Z_{out}, PC_{in}, End.

* CONDITIONAL BRANCH-

In this case the status of condition codes is required to check before loading a new value into the PC for example- For a branch on negative condition Step 4 is replaced with
~~if N=0~~
4 - Offset - field - of IR_{out}, Add, Z_{in}, if N = 0 then End. i.e., if N = 0 the processor returns to step 1 immediately after step 4.

If n=1 Step 5 is performed to a new value into the PC, thus performing the branch operation.

(Jmp)

MAIN MEMORY & CENTRAL MEMORY-

* A computer that employes a microprogram unit which have two seperate memory, a main memory & central memory.
* A main memory is available to the users for storing the programs. The contents of main memory utter manimulated & every time that the program change.
* The user program the main memory consist of machine instruction & data.
* In contrast the central memory holds a fixed microprogram that can not be altered.
* The microprogram consist of microinstruction that specify various internal central signals. for execution of register micro operation.
* Each main instruction initiate a series of micro instruction in central memory
* These micro instruction generates the micro

operation to fetch the instruction from the main memory to evaluate the effective ~~addr~~ memory to evaluate the effective address to execute the operation specified by the instruction & return to the central the fetch phase in order to repeat the cycle for the next instruction.

* The central memory address register (CMAR or CAR) specifies the address of micro instruction & the central data register (CDR). holds the micro instruction read from memory the micro instruction contains a control word that specifies one or more micro operation for data processor.

The next address generator is some time called a micro-program sequencer. as it determines the address sequence that is read from central memory. The address of the next micro instruction can be specified in several ways depending upon the sequencer inputs. The ~~depending~~ addtess sequencing capabilities required in a control memory are -

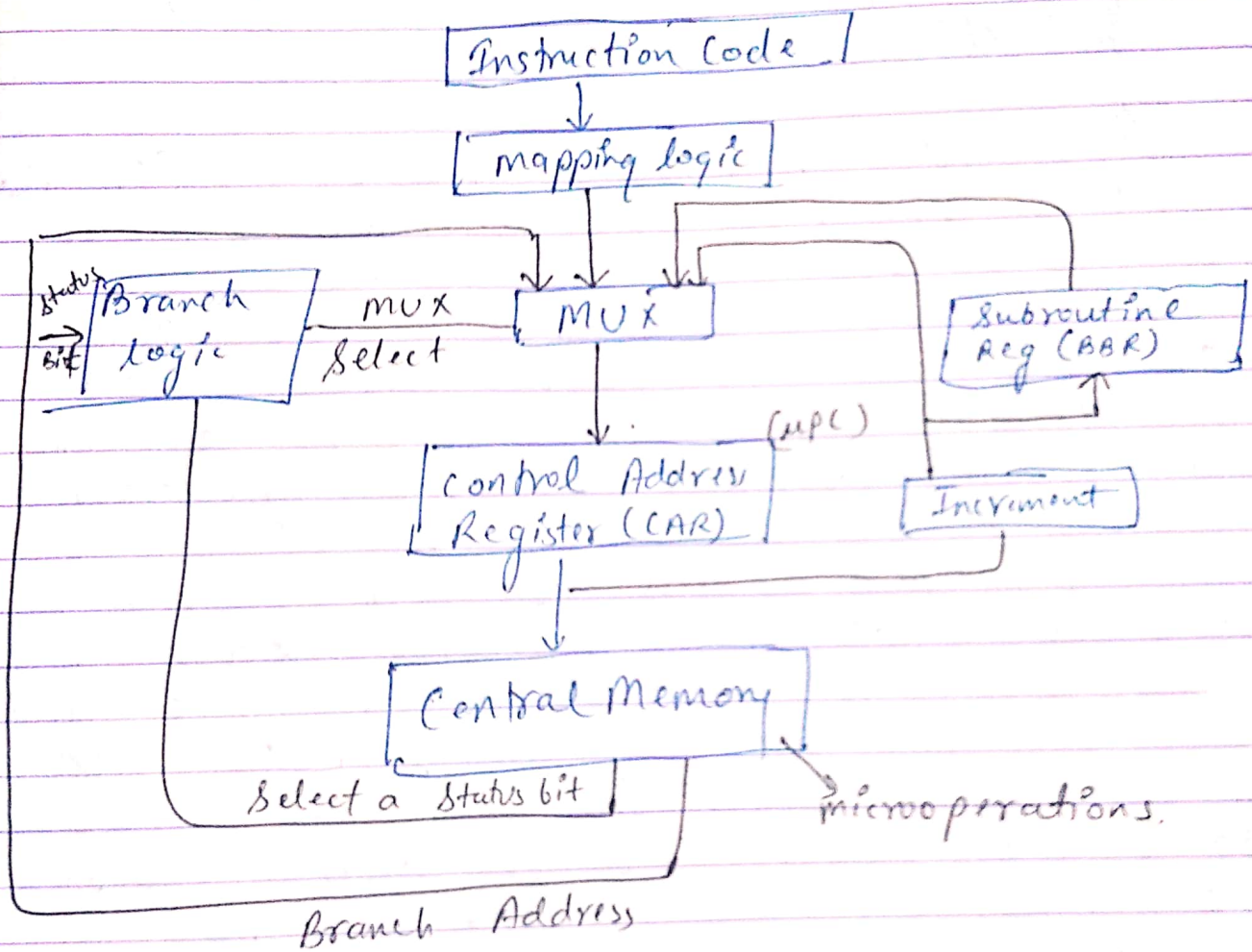
(i) Increment the control address registor.
(ii) Unconolition or condition branch depending on status bit condition.
(iii) A mapping progress from the bits of the instruction to the address for

central memory.

(Rv) A facility for subroutine call & return.

(Imp)

BLOCK DIAGRAM OF MICRO PROGRAM OR SEQUENCES -

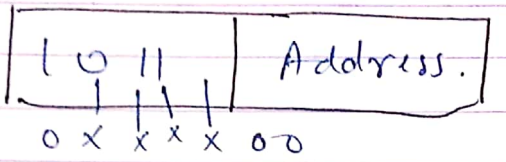


for Address Mapping. Answer →

The transformation from the instruction code bit to an address in control memory where the routine microprogram is located is referred to as a mapping processor.

A mapping procedure is a rule that transform the instruction code into a control memory address for each operation code (Add, Sub, decode) there is a microprogram routine in control memory that execute the instruction.

Instruction code mapping bits



microprogram instruction 0 10 1100 (7 bits) (CAR addrs)

ex- 1011
let 0 1011000

ex- 011100
0 1110 0000

microprogram that use subroutine must have procedure for storing the return address during a subroutine call & restoring the address during a sub-routine return.

This may be fullfilled by placing the increment output from the control address registor into a (SBR) & branching to the beginning the subroutine.

Speed (S) = $\frac{nt_n}{(K+n-1)\times t_p}$

n → no. of task

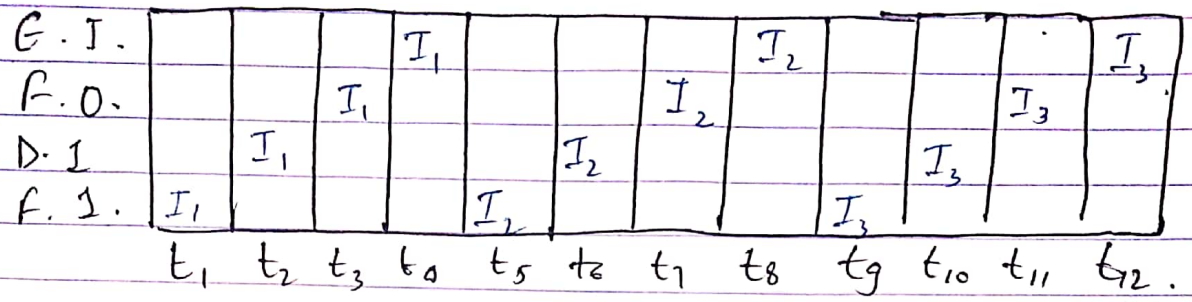
t_n → time taken by non-pipelining system for task

K → Segments.

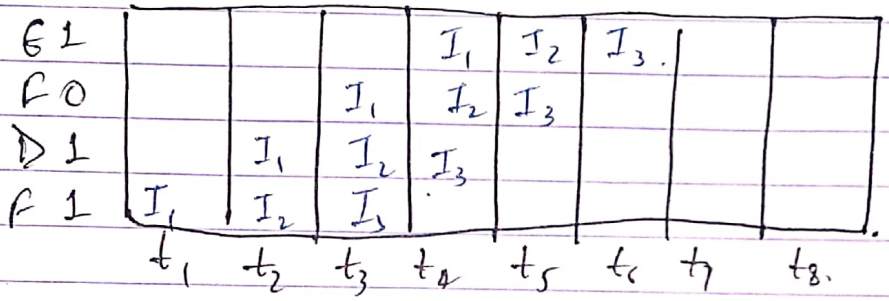
t_p → time required, for.

PIPELINING -

Instructions - I_1, I_2 & I_3.



12 unit Time (Non-Pipelining).



6 unit Time (Pipelining).

Q Arithmetic Pipelining of $A_i * B_i + C_i$.