Basic Processing Unit: Unit-3 ①

Some fundamental concept :—

→ To execute a program, the processor fetches one instruction at a time and performs the operations specified.

→ Instructions are fetched from successive memory locations until a branch or a jump instruction is encountered.

→ The processor keeps track of the address of the memory location containing the next instruction to be fetched using the program counter (PC).

→ After fetching an instruction, the contents of the PC are updated to point to the next instruction in the sequence. A branch instruction may load a different value into the PC.

→ Another key register in the processor is the instruction register (IR). Each instruction comprises 4 bytes, and that it is stored in one memory word.

→ To execute an instruction, the processor has to perform the following 3 steps:

(1) Fetch the content of the memory location pointed to by the PC. The contents of this location are interpreted as an instruction to be executed. Hence, they are loaded into the IR.

$$IR \leftarrow [[PC]]$$

(2) Assuming that the memory is byte addressable, increment the contents of the PC by 4, that is

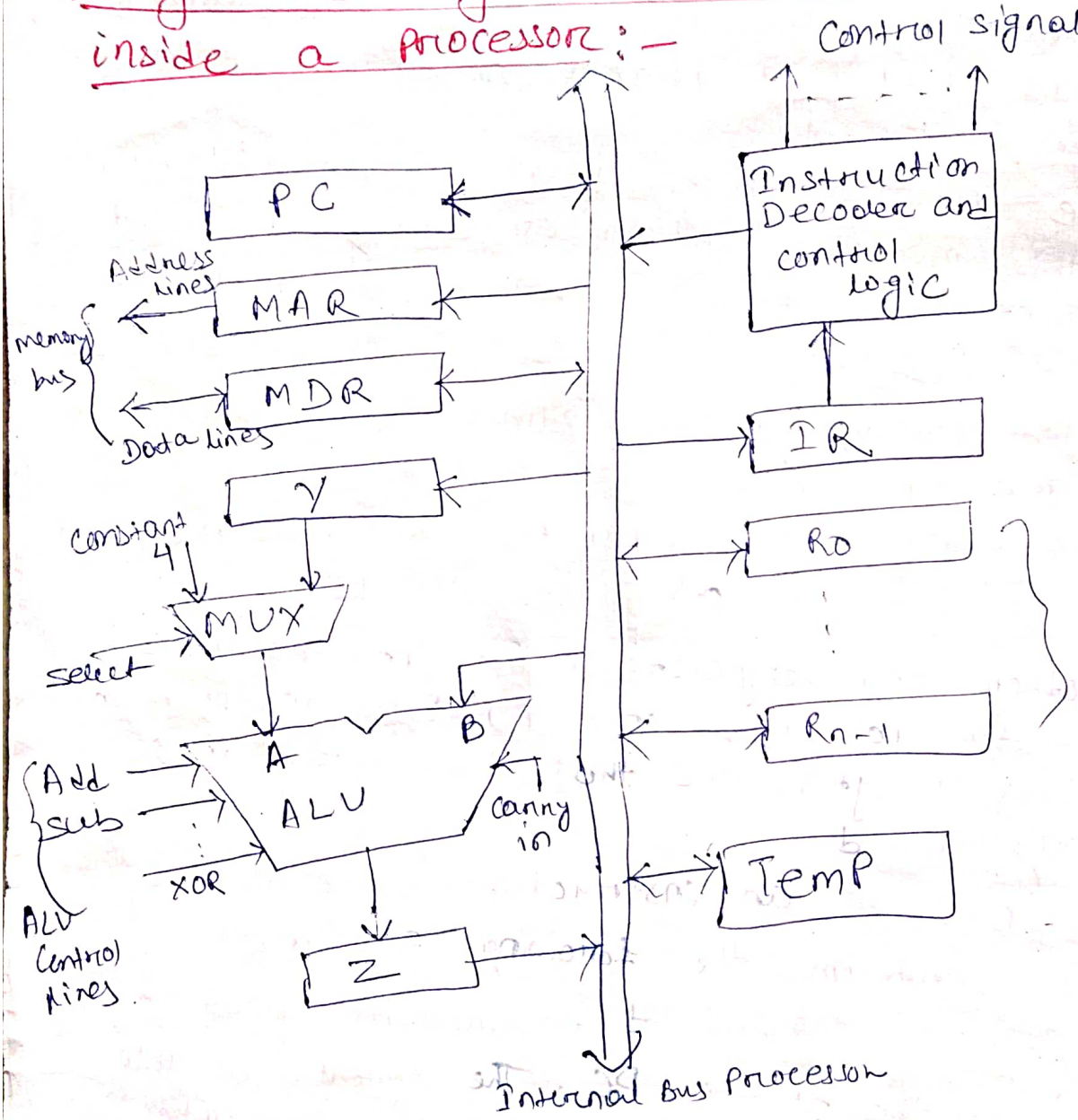
$$PC \leftarrow [PC] + 4$$

(3) (Perform) Carry out the actions specified by the instruction in the IR.

→ If instruction occupies more than one word, steps 1 and 2 must be repeated as many times as necessary to fetch the complete instruction.

→ First 2 steps are referred to as fetch phase and step 3 constitutes the execution phase.

Single bus organization of the data path inside a processor :-



→ ALU and all the registers are interconnected via a single common bus.

→ This bus is internal to the processor.

→ Register MDR has two i/ps & two o/ps. Data may be loaded into MDR either from memory bus or from the internal processor bus. The data stored in the MDR may be placed on either bus.

→ The i/p of MAR is connected to the internal bus and it's o/p is connected to external bus.

→ The control lines of the memory bus are connected to the instruction decoder and control logic block. This unit is responsible for issuing the signals that control the operation of all the units inside the processor and for interacting with the memory bus.

→ The general purpose registers R_0 through R_{n-1} are used by the programmer. ~~special purpose registers~~

→ Three registers, Y, Z and TEMP are transparent to programmer because they are never referenced explicitly by any instruction. These registers are never used for storing data generated by one instruction for later use by another instruction.

→ The multiplexer MUX selects either the output of register Y or a constant value 4 to be provided as input A of the ALU. The constant 4 is used to increment the contents of the program counter. Select line of MUX are used for selecting the i/p as o/p.

→ The instruction decoder and control logic unit is responsible for implementing the actions specified by the instruction loaded in the IR register. The decoder generates the control signals needed to select the registers involved and direct the transfer of the data.

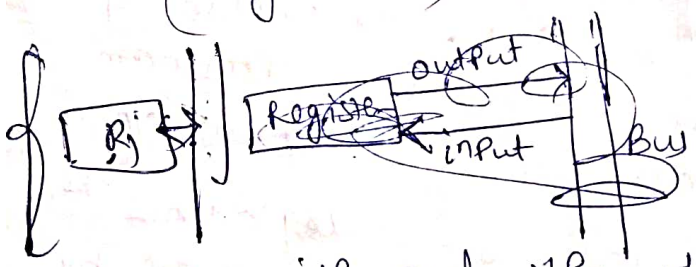
→ The registers, ALU and the interconnecting bus are collectively referred to as the Datapath

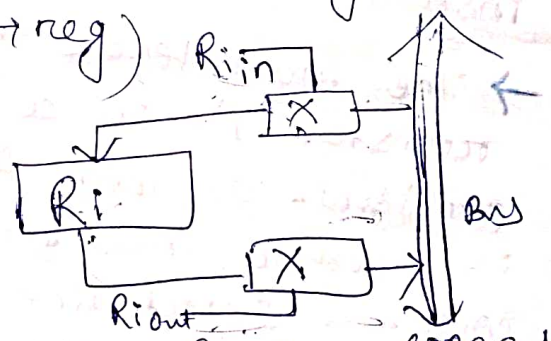
→ An instruction can be executed by performing one or more of following operations in some specified sequence.

* Transfer a word of data from one processor register to another or to the ALU.
* Perform an arithmatic or a logic operation and store the result in a processor register.
* fetch the contents of a given memory location and load them into a processor register.
* Store a word from a processor register into a given memory location.

1) Register transfers :-

→ For each register, two control signals are used to place the contents of that register on the bus (reg → bus), or to load the data on the bus into register (Bus → reg).





→ The i/p and o/p of register Ri are connected to the bus via switches controlled by the signals Ri in and Ri out, respectively.

→ When Ri in is set to 1, the data on the bus are loaded into Ri. Similarly when Ri out is set to 1, the data or content on Register Ri are placed on bus.

→ While Ri out is equal to 0, the bus can be used for transferring data from other registers

Example :-

MOV R_1, R_4 : $R_4 \leftarrow R_1$ $(R_4 = R_1)$

→ Transfer the contents of register R_1 to register R_4. What are the control signals generated to accomplish this task.

Ans : (i) $R_{1out} = 1$ (Enable the o/p of register R_1. This place the contents of R_1 on the processor bus)

(ii) $R_{4in} = 1$ ↳ Enable the i/p of register R_4. This loads data from the processor bus into register R_4.

(iii) $R_{1out} = 0$; $R_{4in} = 0$

→ Processor clock

* All operations and data transfers within the processor takes place within time periods defined by processor clock.

→ Multiphase clocking :-

* two or more clock ~~cycles~~ signals may be needed to guarantee proper transfer of data.

2 Performing an arithmetic or Logic Operation

→ ALU has no internal storage.

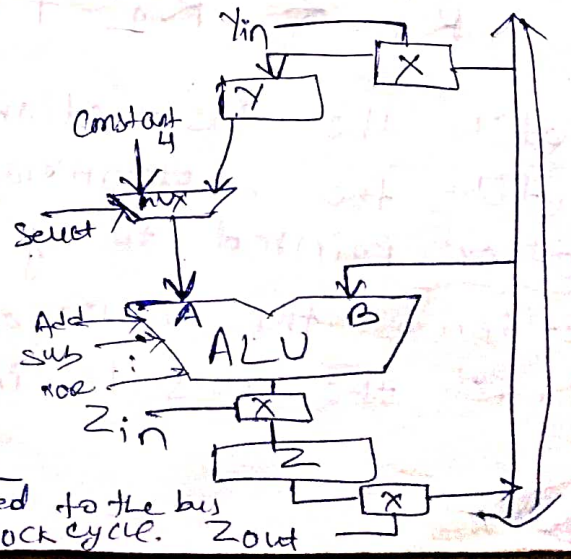
→ performs on the two operands A and B.

Ex :- $R_3 = R_1 + R_2$

(i) R_{1out}, Y_{in}

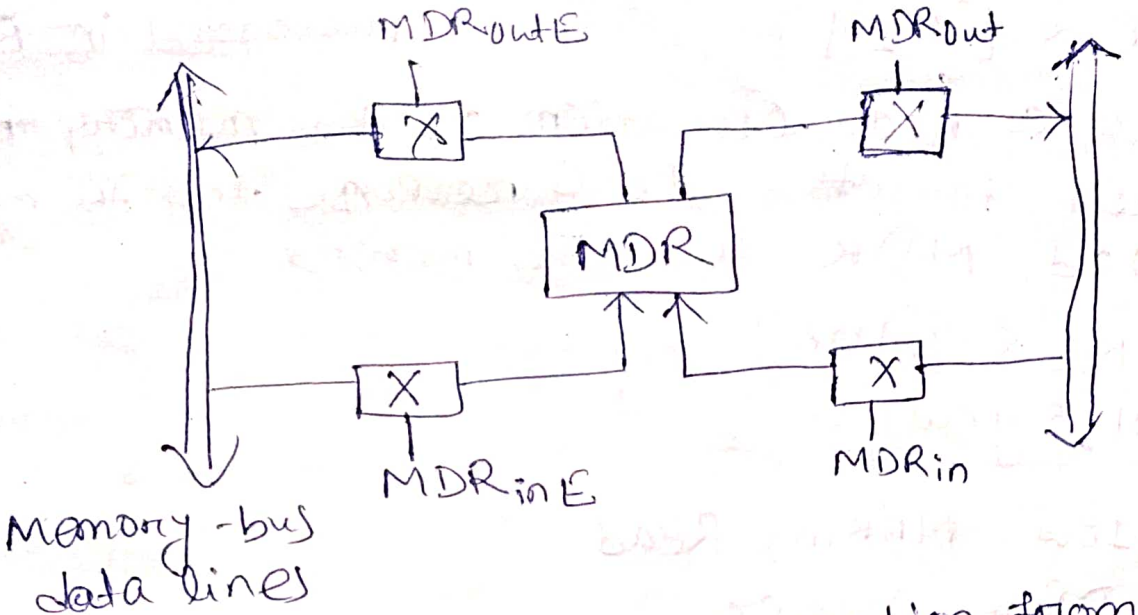
(ii) R_{2out}, Select Y, Add, Z_{in}

(iii) Z_{out}, R_{3in}



In one step, only one register output can be connected to the bus during any clock cycle.

3) Fetching a word from memory :-



→ To fetch a word of information from memory, the processor has to specify the address of the memory location where this information is stored and request a Read operation.

→ The processor transfers the required address to the MAR, whose output is connected to the address lines of the memory bus

→ The processor uses the control lines of the memory bus to indicate that a Read operation is needed.

→ When the requested data are received from the memory they are stored in register MDR, from where they can be transferred to other registers in the processor.

→ It has 4 control signals : MDRin and MDRout control the connection to the internal bus, and MDRinE and MDRoutE control the connection to the external bus.

→ Memory function complete (MFC) is a control signal. The processor waits until it receives an indication i.e. MFC.

Ex:- Move $(R_1), R_2$: $R_2 \leftarrow$ Data present in the memory location presented in R_1.

Actions

(1) $MAR \leftarrow [R_1]$

(2) Start a Read operation on the memory bus.

(3) Wait for the MFC response from the memory.

(4) Load MDR from the memory bus.

(5) $R_2 \leftarrow [MDR]$

Control signals :-

(1) R_{1out}, MARin, Read

(2) MDRinE, WMFC

(3) MDRout, R2in

where WMFC is the control signal that causes the processor's control to wait for the arrival of MFC signal.

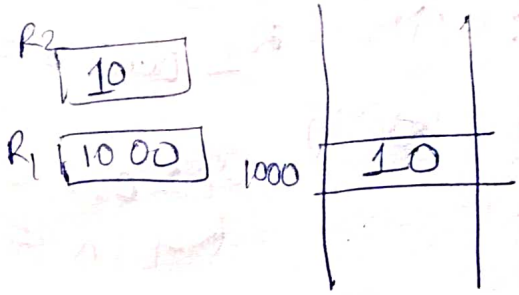
(4) Storing a word in memory :-

→ desired address is loaded into MAR.
→ data to be written are loaded into MDR.
→ a write command is issued.

Example :-

Move $R_2, (R_1)$

$\Rightarrow M[R_1] \leftarrow R_2$



Write the content of register R_2 into the memory location pointed by register R_1.

(i) R_1 out, MAR in
(ii) R_2 out, MDR in, write
(iii) MDR outE, WMFC

→ The processor remains in step-3, until the memory operation is completed and an MFC response is received.

5 Execution of a Complete Instruction :-

Add $(R_3), R_1$: Adds the content of a memory location pointed to by R_3 to register R_1.

$$R_1 \leftarrow R_1 + M[R_3]$$

→ Fetch the instruction
→ Fetch the first operand (the contents of the memory location pointed to by R3).
→ Perform the additional
→ Load the result into R_1.

The control steps required to perform these operations for the single bus architecture.

(i) PCout, MARin, Read, Select4, Add, Zin
(ii) Zout, PCin, Yin, MDRinE, WMFC
(iii) MDRout, IRin
(iv) R3out, MARin, Read
(v) R1out, Yin, MDRinE, WMFC
(vi) MDRout, SelectY, Add, Zin
(vii) Zout, R1in, End

fetch phase

need in branch instr.

Execution phase

→ Step-1 to step 3 constitute the instruction fetch phase, which is the same for all instruction.

→ The instruction decoding circuit interprets the contents of the IR at the begining of step 4.

→ This enables the control circuitry to activate the control signals for step 4 through 7, which constitute the execution phase.

6 Branch instruction :-

→ A branch instrn replaces the contents of the PC with the branch ~~targ~~ target address.

→ This address is usually obtained by adding an offset X, which is given in the branch instruction, to the updated value of the PC

unconditional branch instrn :-

(i) PCout, MARin, Read, Select4, Add, Zin
(ii) Zout, PCin, Yin, MDRinE, WMFC
(iii) MDRout, IRin
(iv) Offset-field-of-IRout, Add, Zin
(v) Zout, PCin, End

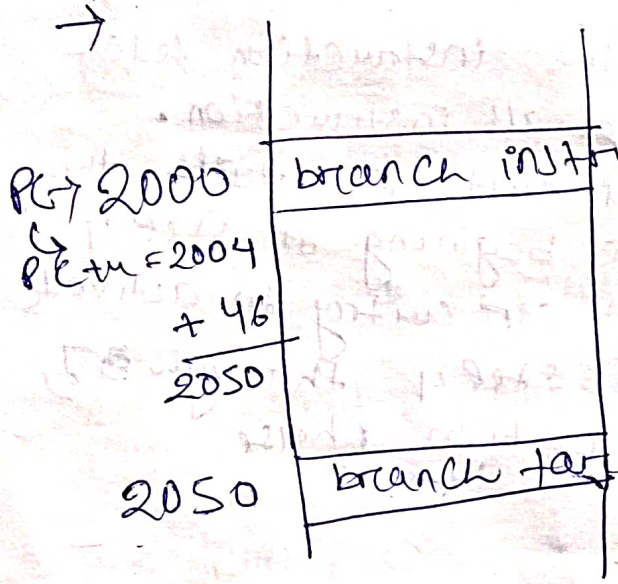
Select Y ??

→ The offset value is extracted from the IR by the instruction decoding circuit.

→ The updated P.C value is already available in register Y, the offset X is gated onto the b and an addition operation is performed.

→ The result which is the branch target addr is loaded into the PC.

→



→ contains offset X.

X = 46 (difference betn the BTA and the addre immediately following the branch instr

Conditional branch instrn

Ex:- For a branch-on-negative (Branch<0) instruction

(iv) offset-field-of IRout, select Y, Add, Zin,
If N = 0 then End

(v) Zout, PCin, End

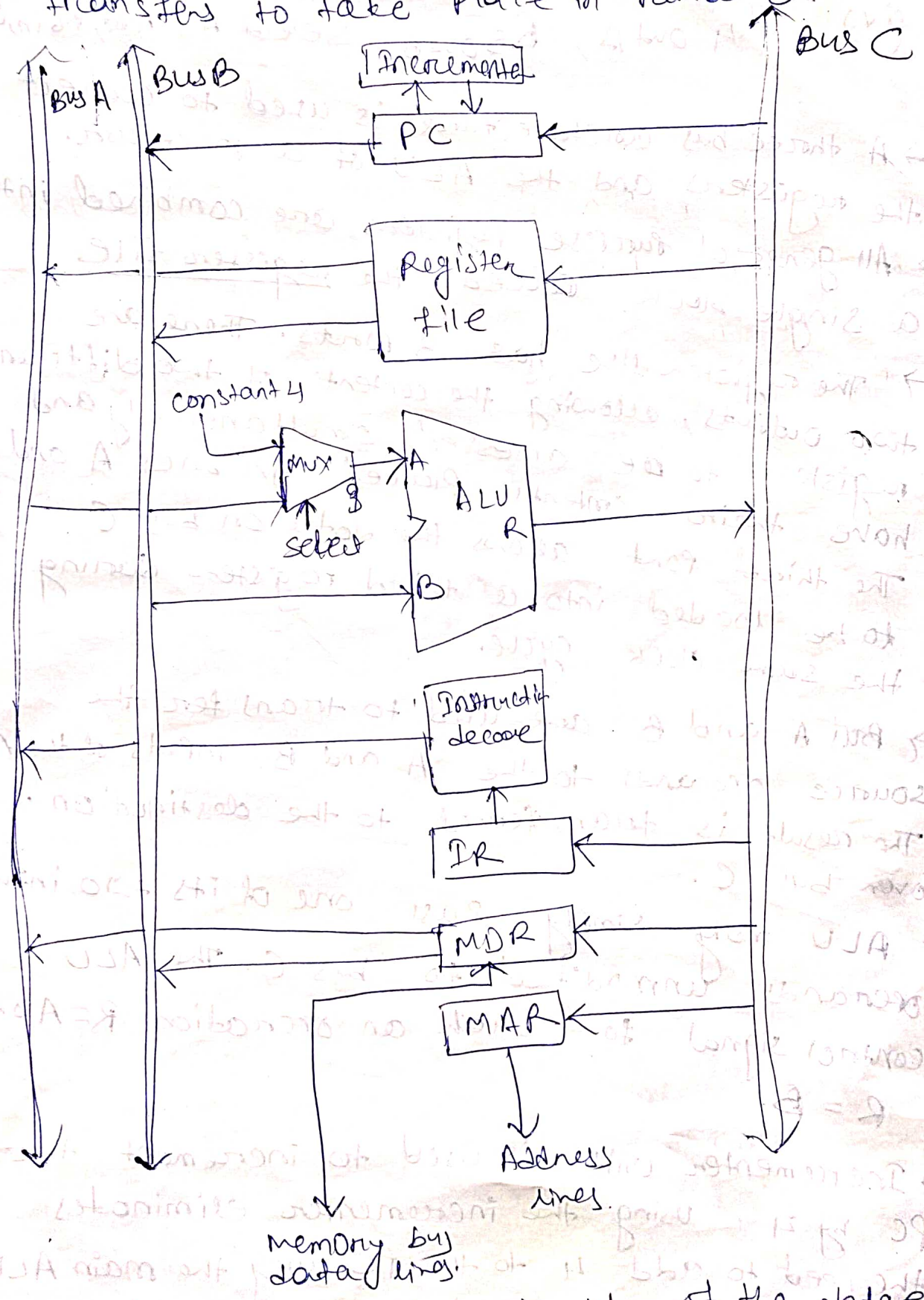
if N=0, the processor returns to step 1 immediatly after 4.

if N=1, step 5 is performed to load a neg value into the PC, thus performing the branch operation.

Multiple Bus organization :-

→ Only one data can be transferred over the bus in a clock cycle.

→ To reduce the no. of steps needed in single bus, processors provides ~~multiple internal paths that enable several transfers to take place in para~~ internal path enables several transfers to take place in parallel.



(Three bus organization of the data path.)

Add R4, R5, R6 : R6 ← R4 + R5

(i) PCout, R=B, MARin, Read, IncPC

(ii) WMFC → the content of PC are passed through the ALU, using R=B control signal

(iii) MDRoutB, R=B, IRin

(iv) R4 outA, R5 outB, Select A, Add, R6in, End

→ A three bus architecture is used to connect the registers and the ALU of a processor.

→ All general purpose registers are combined into a single block called the register file

→ The register file has 3 ports. There are two outputs, allowing the contents of two different registers to be accessed simultaneously and have their contents placed on buses A and B. The third port allows the data on bus C to be loaded into a third register during the same clock cycle

→ Bus A and B are used to transfer the source operands to the A and B inputs of the ALU. The result is transferred to the destination over bus C.

→ ALU may simply pass one of its two inputs operands unmodified to bus C. The ALU control signal for such an operation R=A or R=B.

→ Incrementer unit is used to increment the PC by 4. Using the incrementer eliminates the need to add 4 to the PC using the main ALU.

→ Constant 4 is used to load multiple and store multiple.

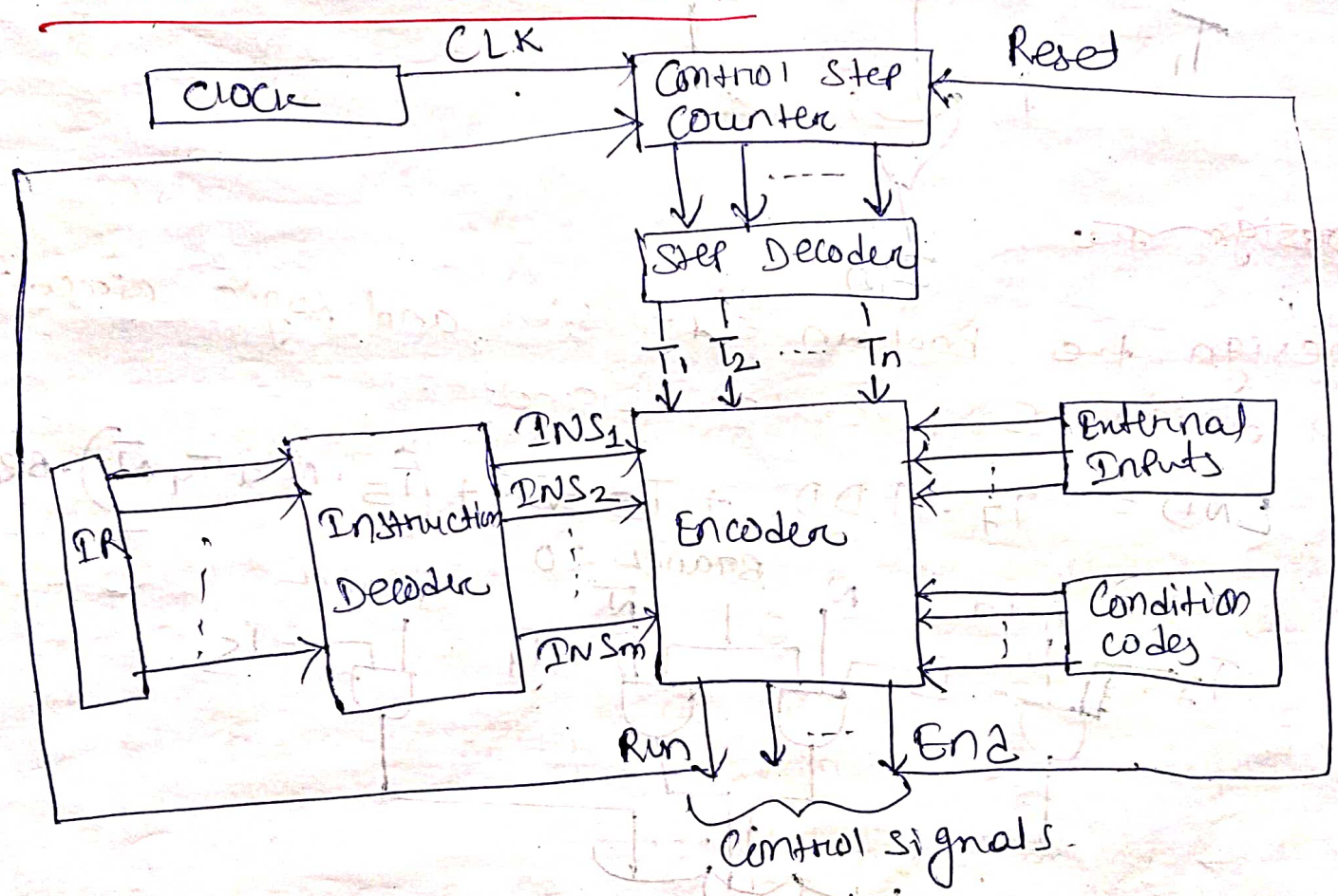
Control unit organization :-

→ To execute instruction, the processor generates control signals in a proper sequence and these control signal are generated from control unit.

→ There are two types of control organization techniques :

(i) Hardwired control unit
(ii) Micro programmed control unit

Hardwired Control Unit :-



→ Control step counter is used to keep track of the control steps.

→ Step decoder provides a separate signal line for each step or time slot, in the control sequence.

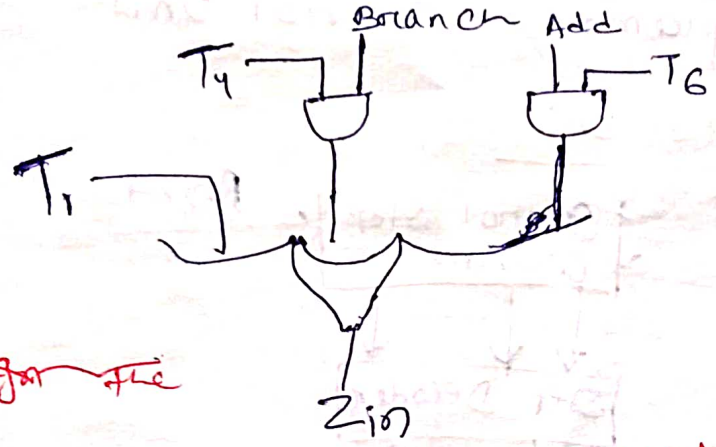
→ When instruction loaded in IR, any one of the output lines (INS_1 through INS_m) is set to 1 and all other lines are set to 0.

→ The control signals are

* Contents of the ~~control~~ control step counter
* Contents of the instruction register.
* Contents of the condition code flags
* External input signals, such as MFC and interrupt request.

→ Write the boolean expression and logic circuit to design Z_{in} control signal.

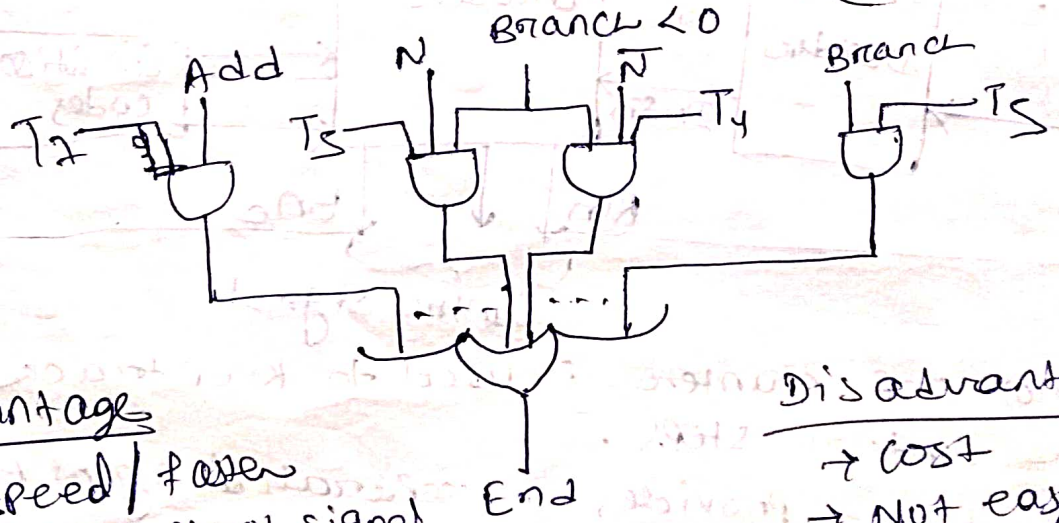
$$Z_{in} = T_1 + T_6 \cdot ADD + T_4 \cdot BR + \cdots$$



~~Design the~~

Design the Boolean expression and logic circuit to design END control signal

$$END = T_7 \cdot ADD + T_5 \cdot BR + (T_5 \cdot N + T_4 \bar{N}) \cdot BRN + \cdots$$



Advantage

→ speed / faster because control signal by using combinational ckt

→ Performance is high in compare to μCU.

Disadvantage

→ cost

→ Not easy to modify for error correction

→ control signal required by CPU more complex

→ modifications very difficult

Microprogrammed Control unit :-

→ Microprogrammed control is an alternative scheme, in which control signals are generated by a program similar to machine language program.

→ * The control signals required inside the processor can be generated using a control step counter and a decoder/encoder circuit.

→ EX :- ADD (R3), R1

The control sequence for the above instruction is:

1. PCout, MARin, Read, select4, Add, Zin
2. Zout, PCin, Yin, MDRout, WMFC
3. MDRout, IRin
4. R3out, MARin, Read
5. R1out, Yin, MDRinE, WMFC
6. MDRout, Select Y, Add, Zin
7. Zout, R1 in, End.

The corresponding micro-instruction are:

Micro instn : Machine programming language

PCin	PCout	MARin	Read	MDRout	MDRinE	IRin	Yin	Select	Add	Zin
1	0	1	1	1	0	0	0	0	1	1

→ A control word (CW) is a word whose individual bits represent the various control signals.

→ A sequence of CWs corresponding to the control sequence of a machine instruction constitutes the microroutine for that instruction.

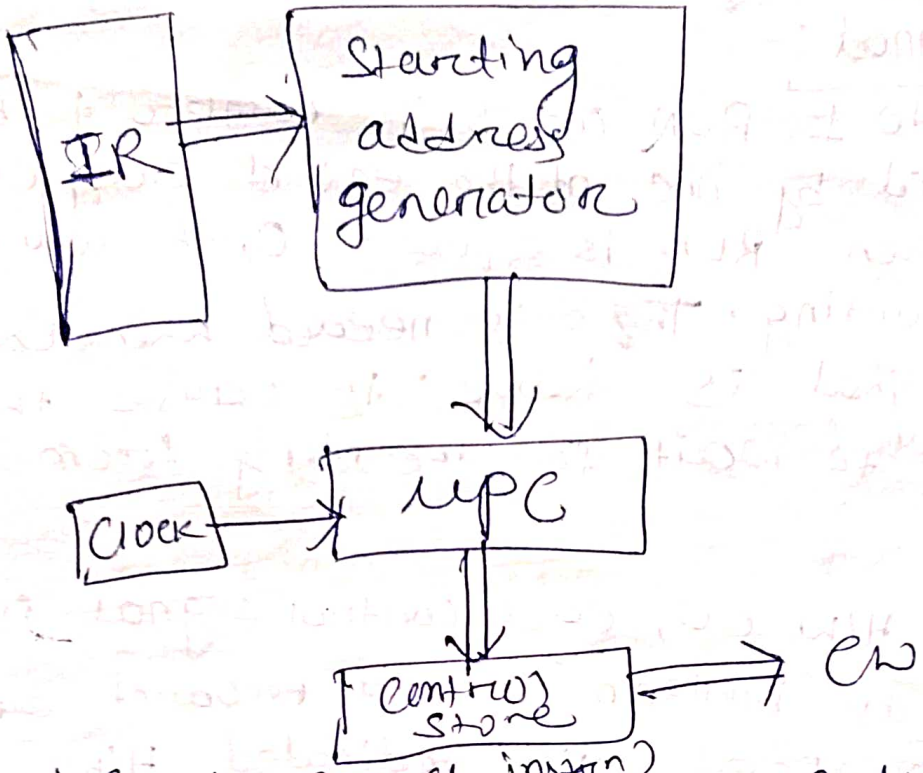
→ Each individual control words in this microroutine are called micro instruction.

STEPS	PCin	PCout	MARin	Read	MDRout	MDRinE	IRin	Yin	Select	Add	Zin	Zout	R1out	R1in	R3out	WMFC	Write	Cs
1.	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0
2.	1	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0
3.	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
4.	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0
5.	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0
6.	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0
7.	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1

Assume select = 0 for selectY
select = 1 for selectY

→ The micro routine for all instructions are stored in a special memory called control store.

→ To read the control words sequentially from control store, a microprogram counter (μPC) is used.



(Homework: CW for Branch instrn)

Advantage of microprogram control unit

→ It is simpler to design, so it is cheaper and less error prone.

→ It is implemented in s/w rather than h/w.

→ It is more flexible, because it can accomodate new instruction also.

Disadvantage:

→ It is slower than the h/w control unit because time requires to access the micro-instruction from the control memory

→ The flexibility can be achieved with some extra h/w cost.

Hard Wired CU (Continue)

END Signal :-

→ The End signal starts a new instruction fetch cycle by resetting the control step counter to its starting value.

→ **Run Signal :-**

When set to 1, RUN causes the counter to be incremented by one at the end of every clock cycle. When RUN is equal to 0, the counter stops counting. This is needed whenever the WMFC signal is issue, to cause the processor to wait for the reply from the memory.

→ In the H/W CU, each control signal is expressed as boolean sum of product expression and is realized using dedicated H/W.

Micro Programmed C.U. :- (Continue)

→ The C.U. generates control signals for any instruction by sequentially reading the CWs of the corresponding micro-routine from the control store.

→ To read the control words sequentially from the control store, a micro program counter (μPC) is used.

→ Every time a new instruction is loaded into the IR, the output of the block labeled "Starting address generator" is loaded into the μPC.

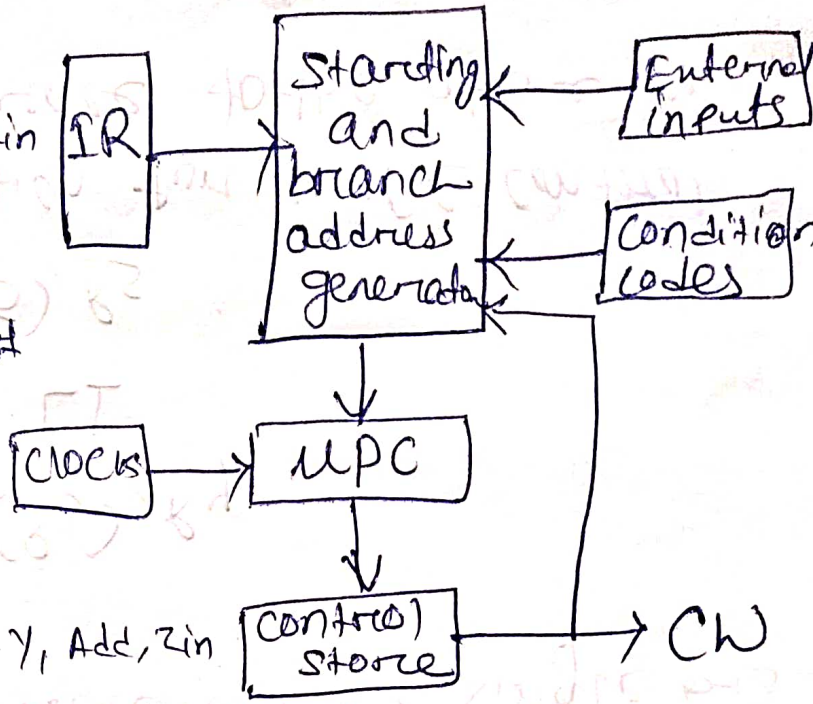
→ The μPC is then automatically incremented by the clock, causing successive microinstructions to be read from the control store.

Branch instruction :-

0. PCout, MARin, Read, Select4, Add, Zin
1. Zout, PCin, Yin, WMFC, MDRinE
2. MDRout, IRin
3. Branch to starting address of appropriate microroutine

- - - - - - - - - - -

25. If N=0, then branch to microinstruction 0
26. offset-field-of-IRout, select Y, Add, Zin
27. Zout, PCin, End.



(conditional branching)

1. When a new instr^n is loaded into the IR, the μPC is loaded with the starting address of the microroutine for that instruction.
2. When an End microinstruction is encountered, the μPC is loaded with the address of the first CW in the microroutine for the instruction fetch cycle.
2. When a Branch microinstruction is encountered and the branch condition is satisfied, the μPC is loaded wi the branch address.