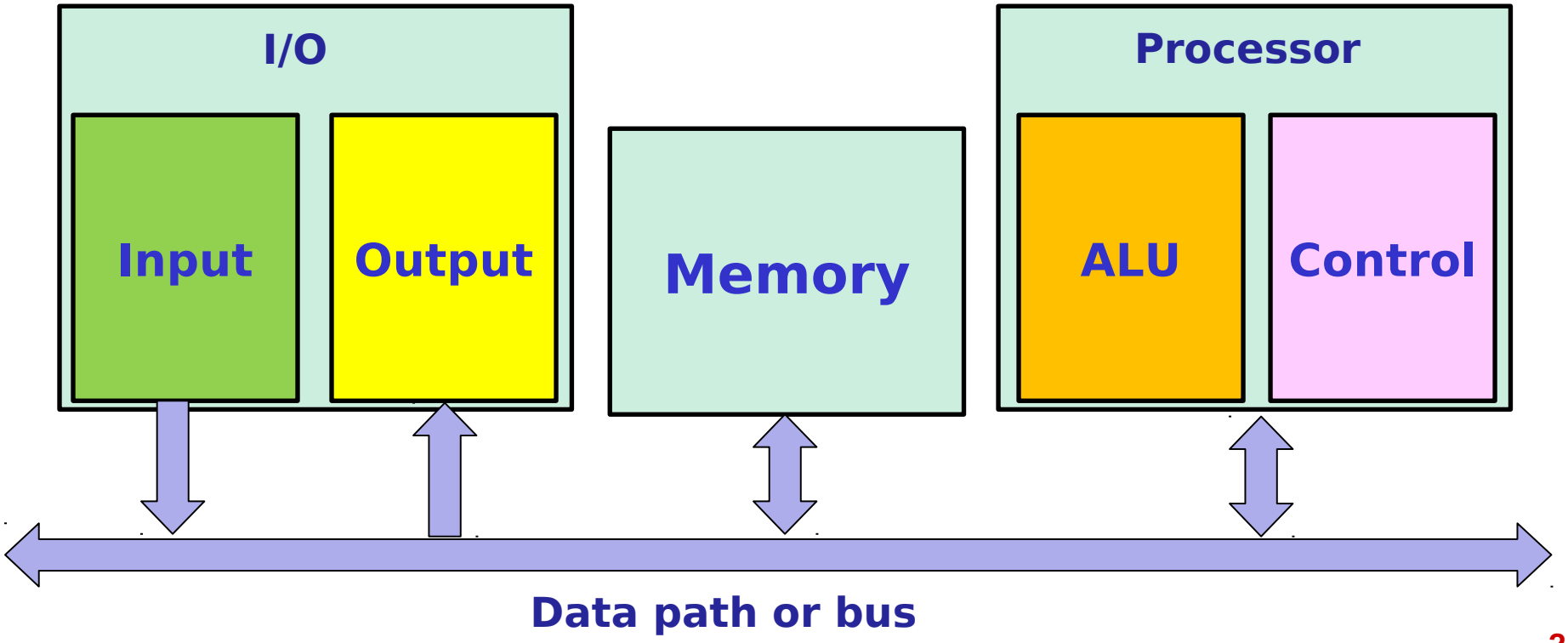
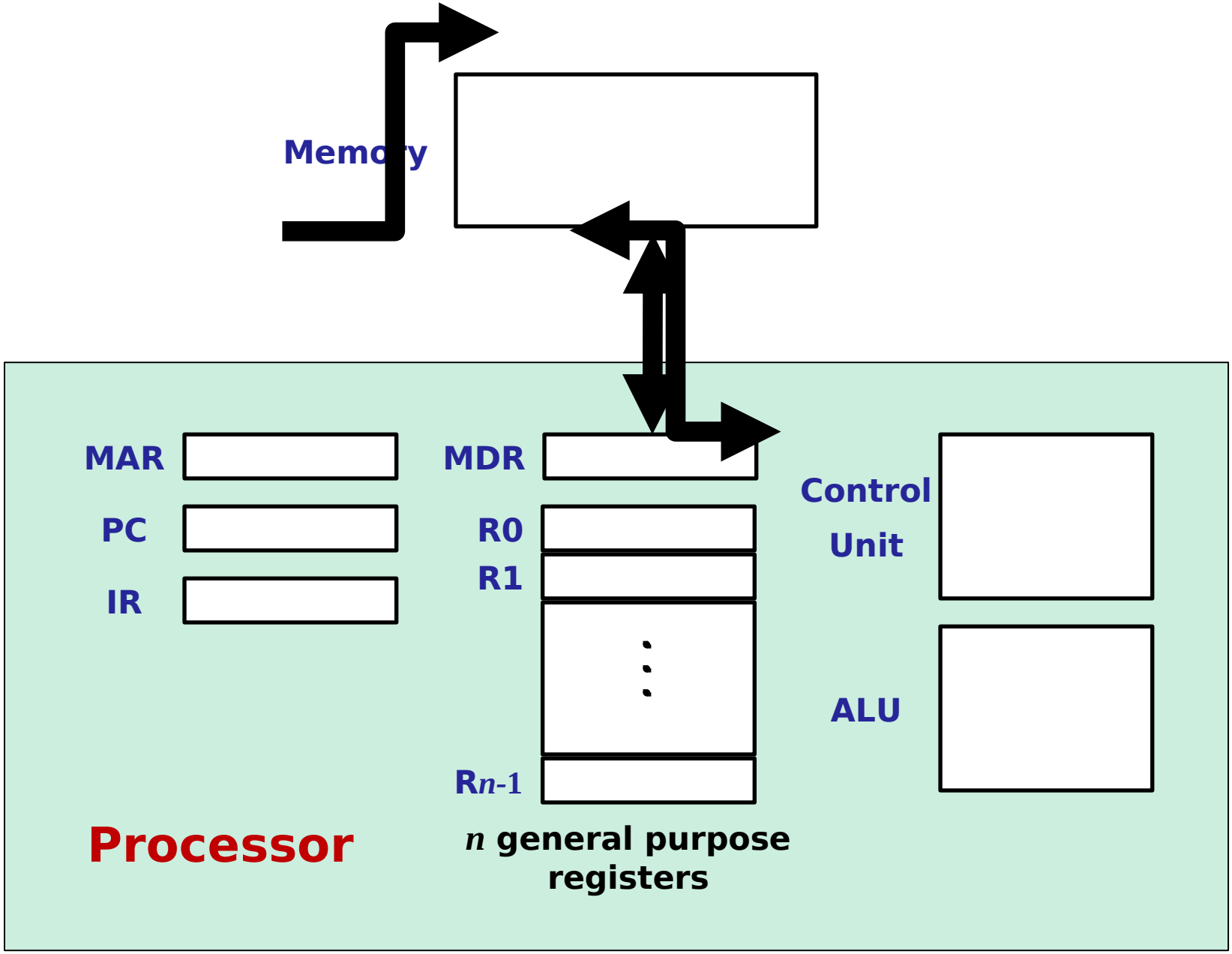
Control Unit

Microarchitecture Level

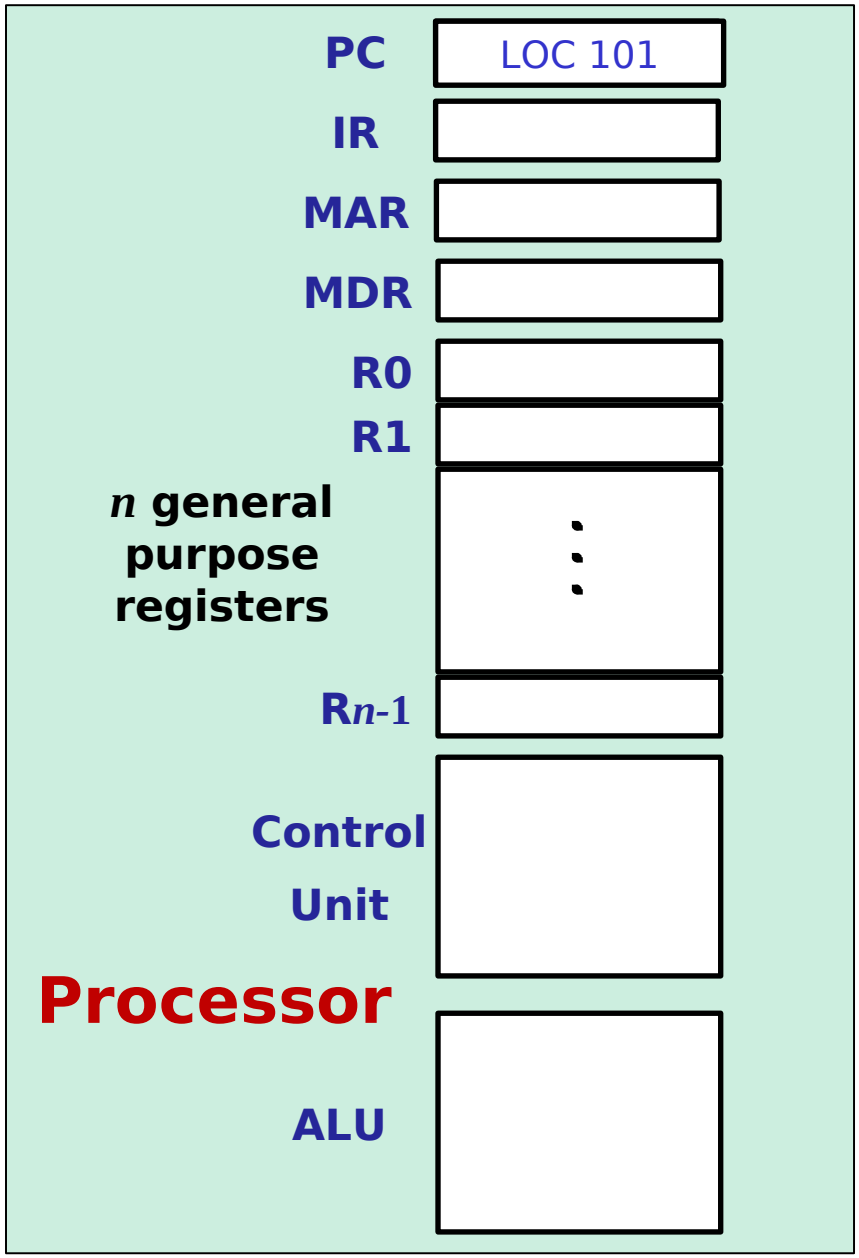
- **Functional Units**:
  - Input Unit
  - Memory Unit
  - Arithmetic and Logic Unit (ALU)
  - Output Unit
  - Control Unit

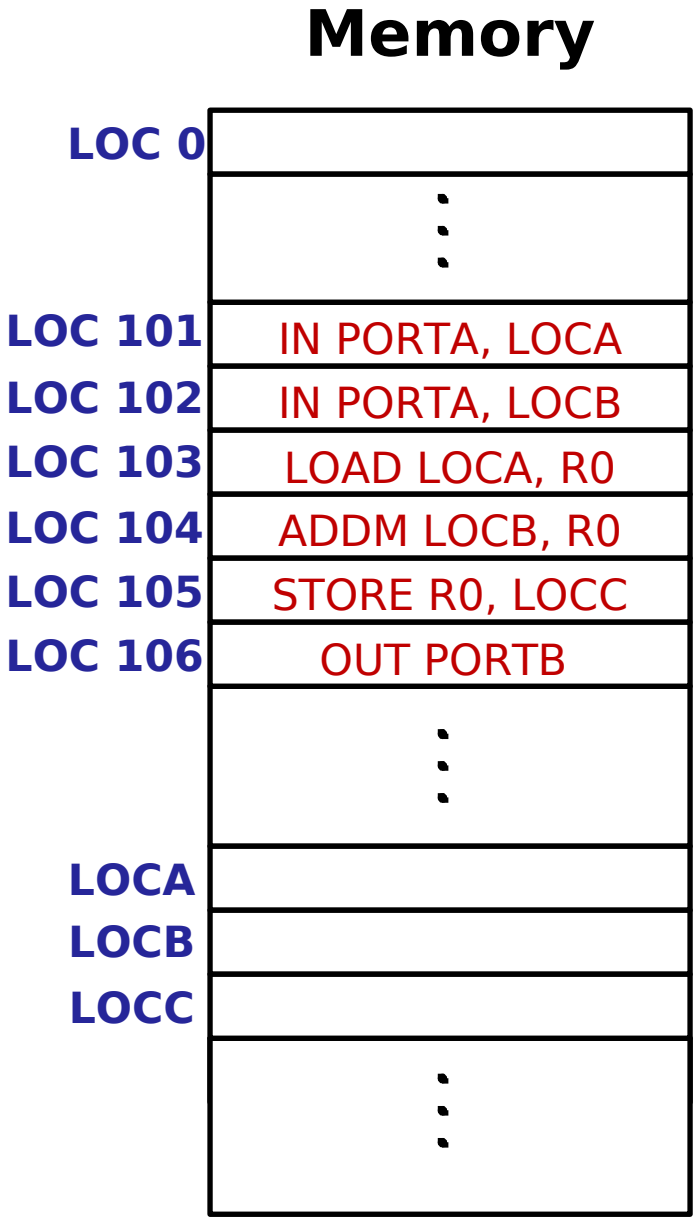


Operational Details

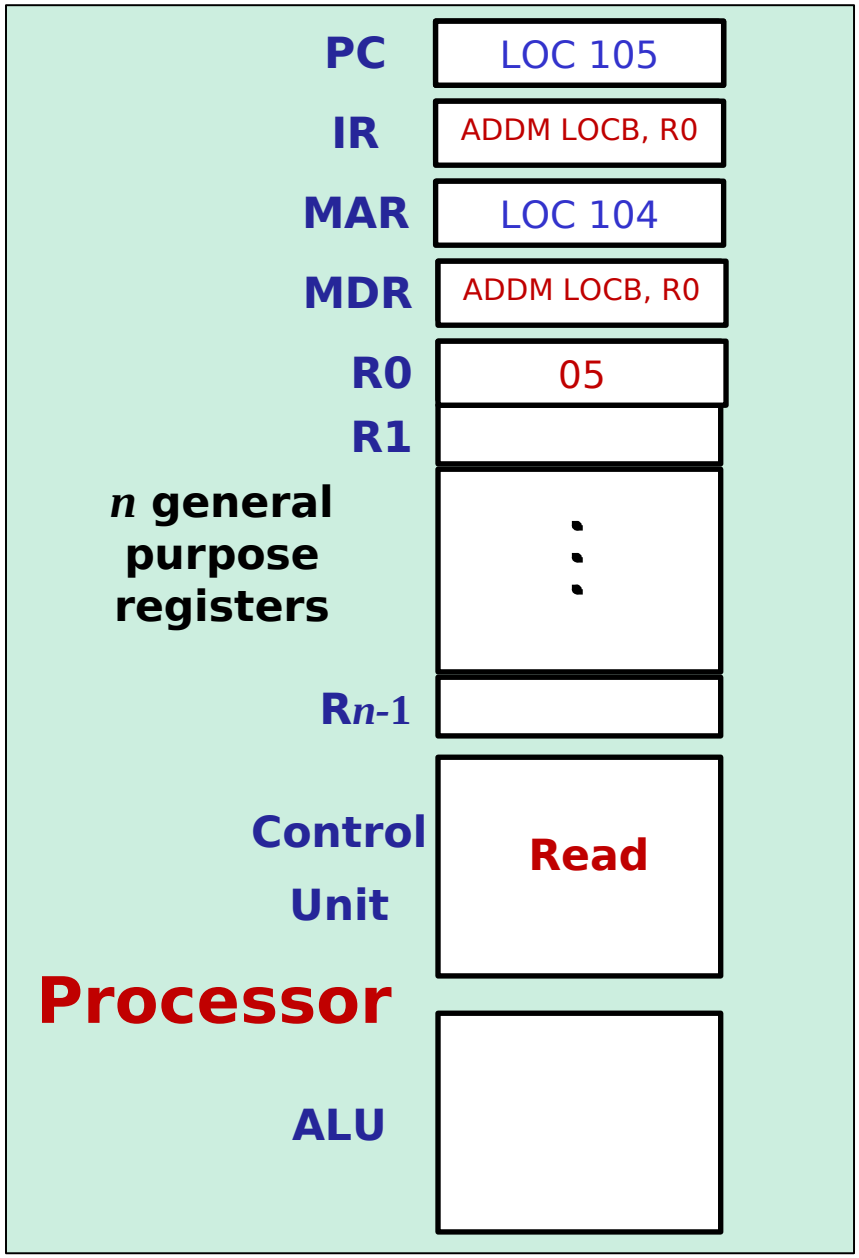


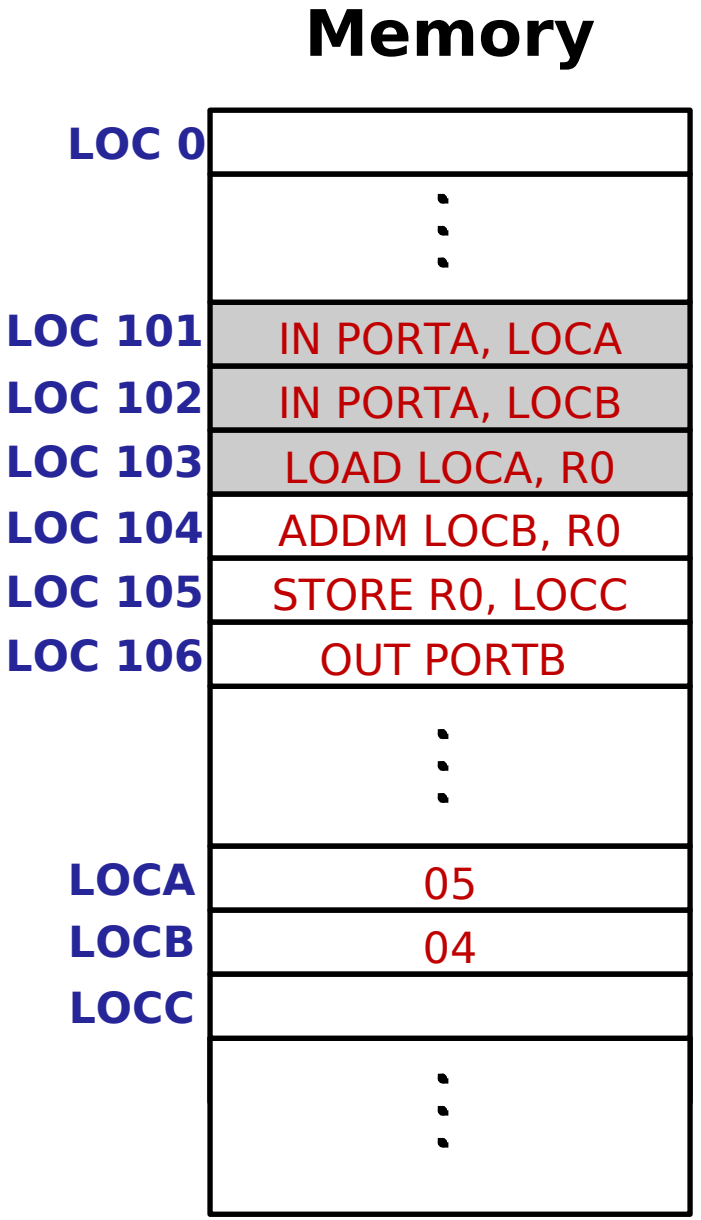
Execution of an Instruction



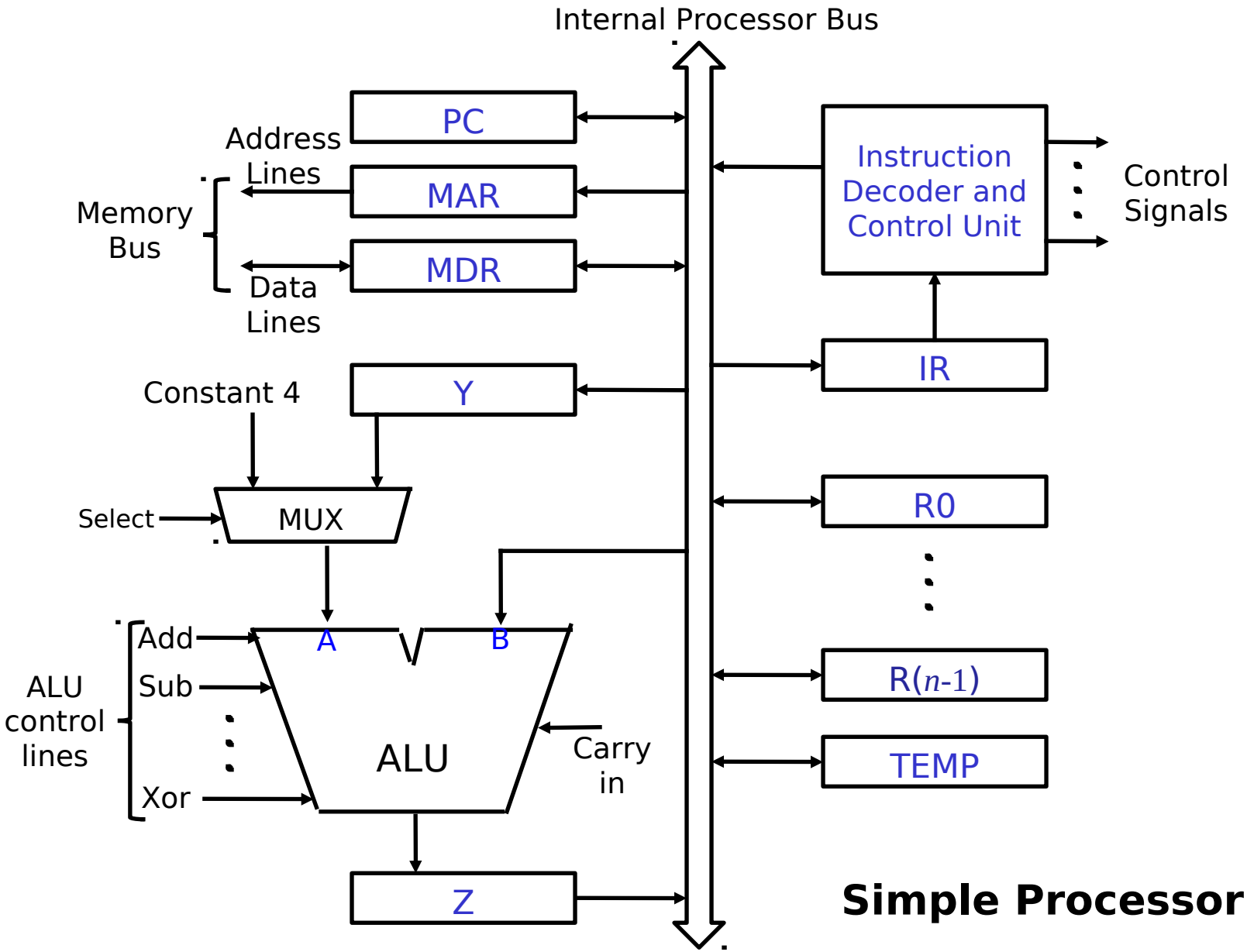


Fetching of an Instruction, ADDM





Single Bus Organization of the Data Path inside Processor



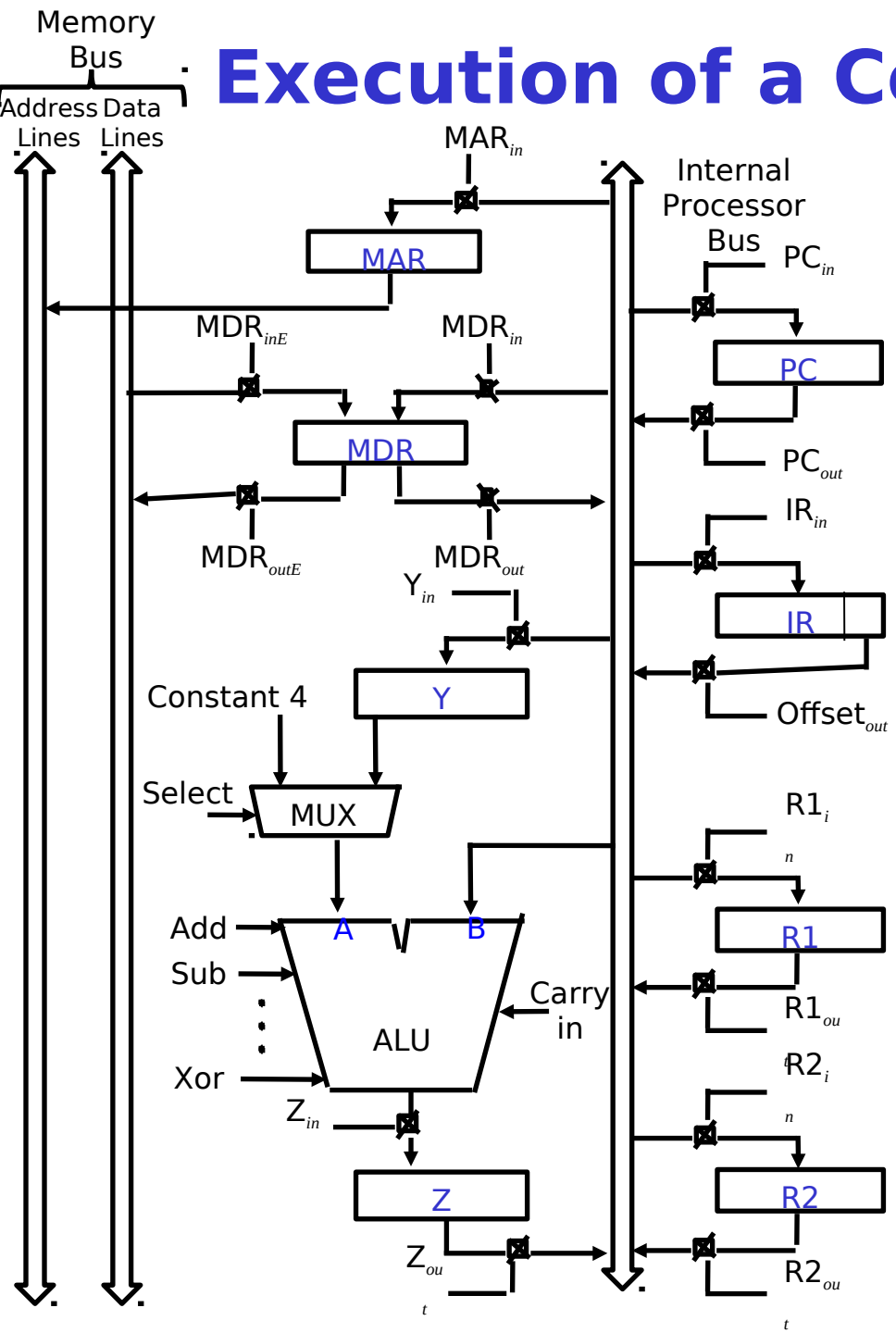
Simple Processor Model

Control Unit

With Fewer exceptions, an instruction can be executed by peforming one or more of the following operations

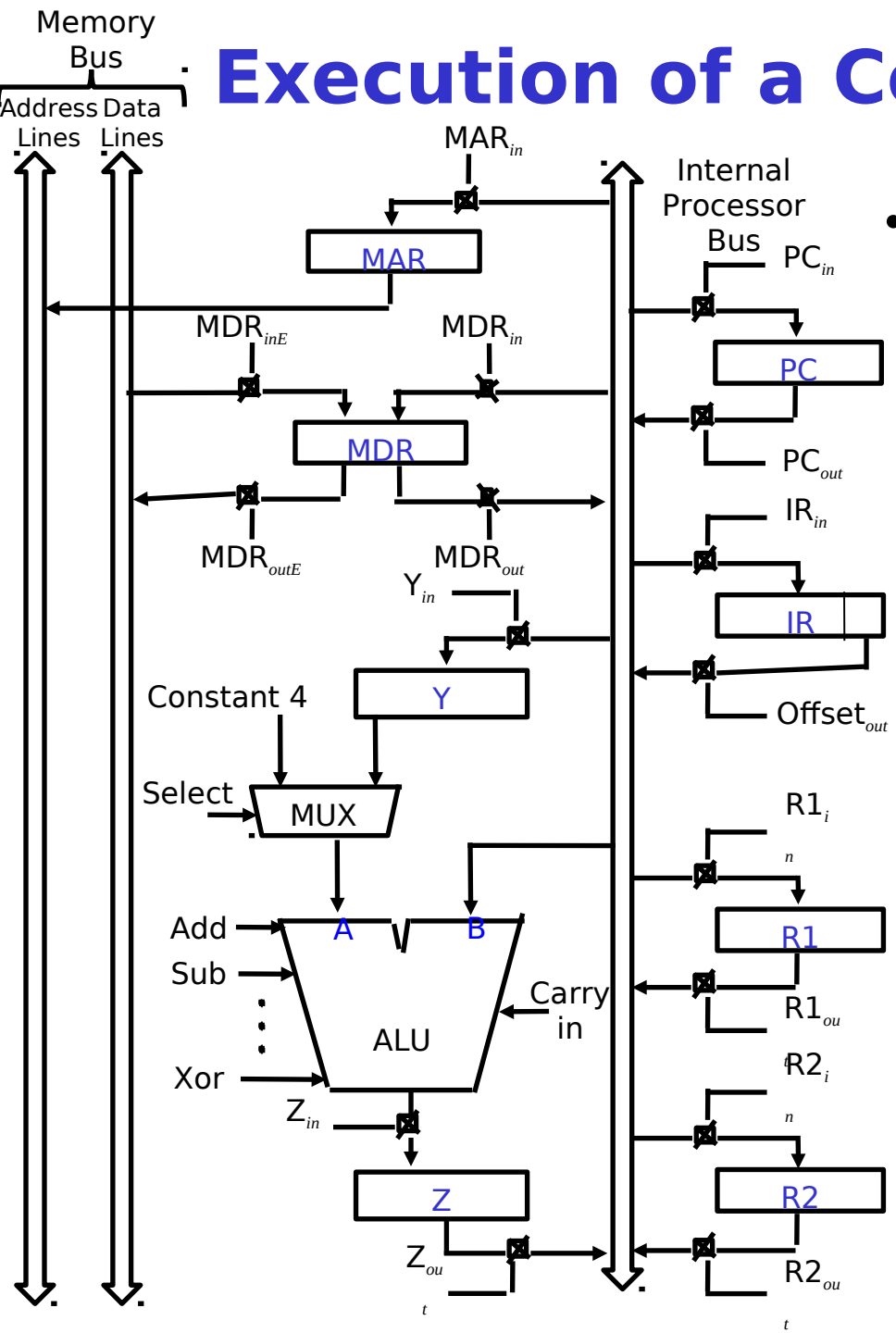
- Transfer a word of data from one processor register to another or to the ALU: Register transfer
- Peform an arithmetic or a logic operation and store the result in a processor register: Arithmetic & logical operation
- Fetch the contents of a given memory location and load them into a processor register: Fetching a word from memory
- Store a word of data from a processor register into a given memory location. : Storing a word into memory

Execution of a Complete Instruction



- Register transfer
- Arithmetic or logic operation
- Fetching a word from memory
- Storing a word in memory

Execution of a Complete Instruction



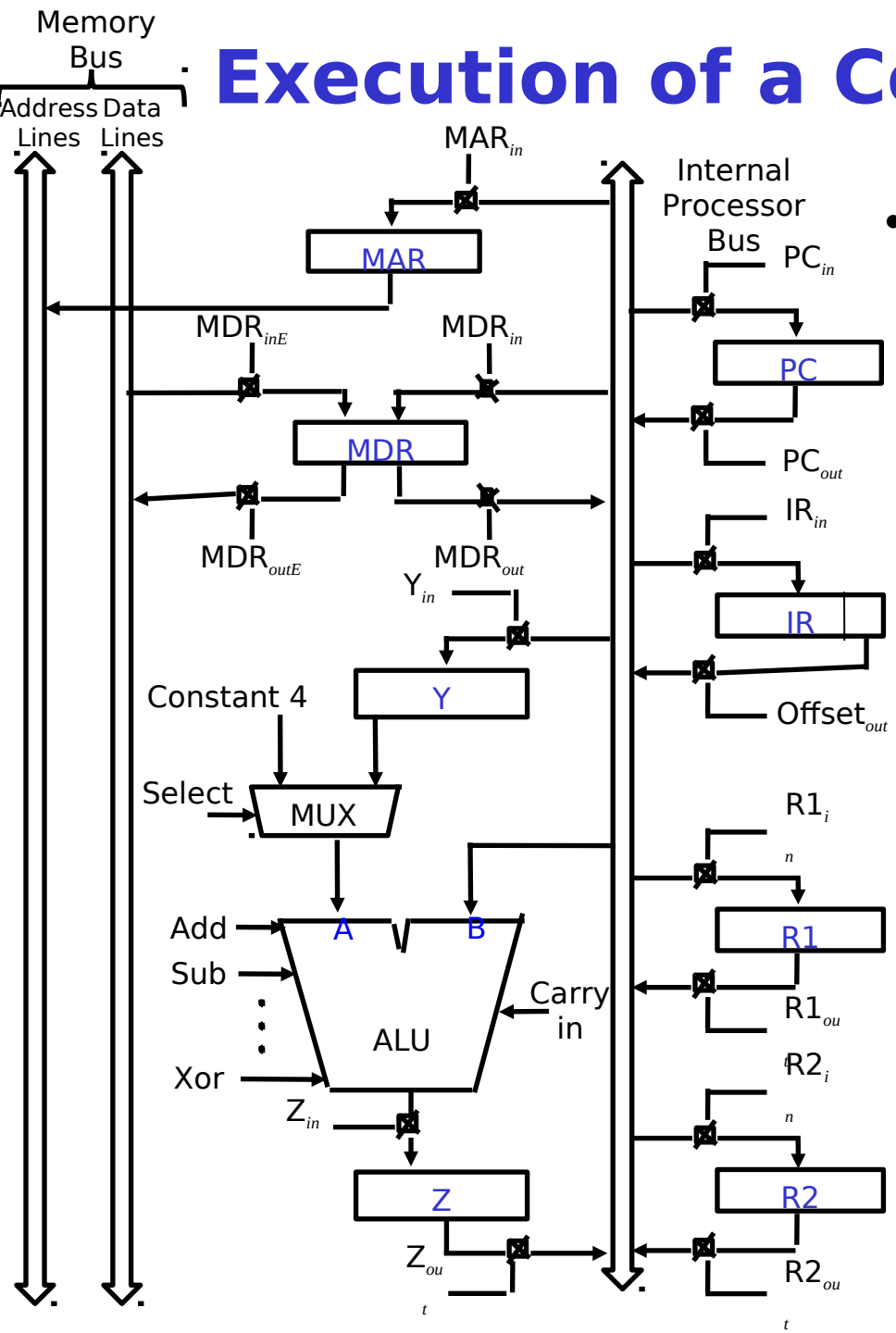
- Control sequence for execution of Add R1, (R3):

1. **PC$_{out}$,MAR$_{in}$,Read,Select 4, Add, Z$_{in}$**
2. **Z$_{out}$, PC$_{in}$, Y$_{in}$, MDR$_{inE}$, WMFC**
3. **MDR$_{out}$, IR$_{in}$**

1. **R3$_{out}$, MAR$_{in}$, Read**
2. **R1$_{out}$, Y$_{in}$, WMFC**
3. **MDR$_{out}$, Select Y, Add, Z$_{in}$**
4. **Z$_{out}$, R1$_{in}$, End.**

MUL R1,R2

Execution of a Complete Instruction

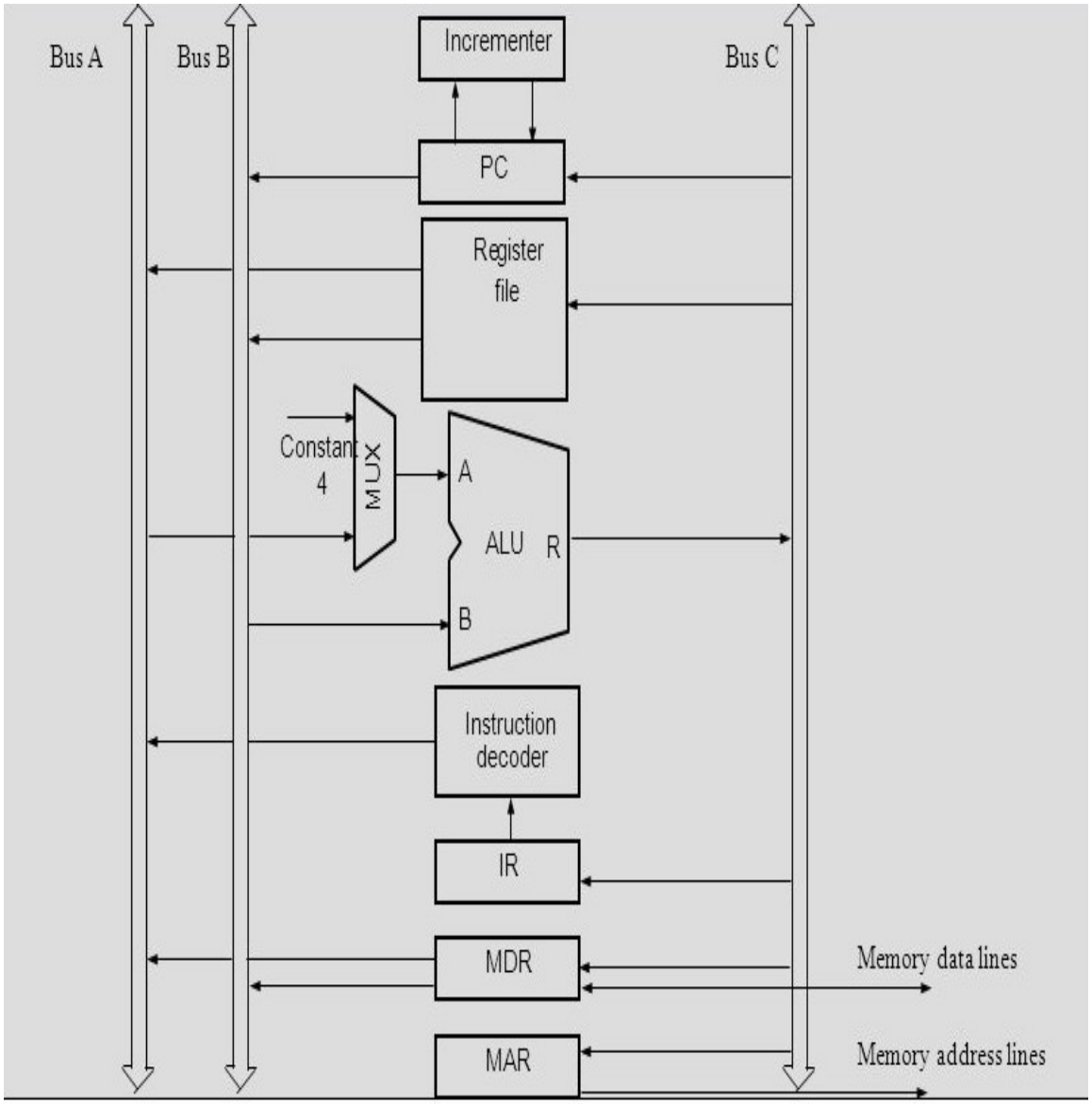


- Control sequence for execution of JMP Loop_Begin:

1. **PC$_{out}$, MAR$_{in}$, Read, Select 4, Add, Z$_{in}$**
2. **Z$_{out}$, PC$_{in}$, Y$_{in}$, MDR$_{inE}$, WMFC**
3. **MDR$_{out}$, IR$_{in}$**

1. **Offset_field_of_IR$_{out}$, Select Y, Add, Z$_{in}$**
2. **Z$_{out}$, PC$_{in}$, End**

Three Bus Organization of the Data Path inside Processor



Control Unit

- Processor must have some means of generating the control signals needed in the proper sequence
- Two categories of approaches used to generate the control signals in proper sequence:
  - Hardwired control
  - Microprogrammed control

Hardwired Control

- Control sequence for execution of ADD R1, [R2]:

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC

T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read

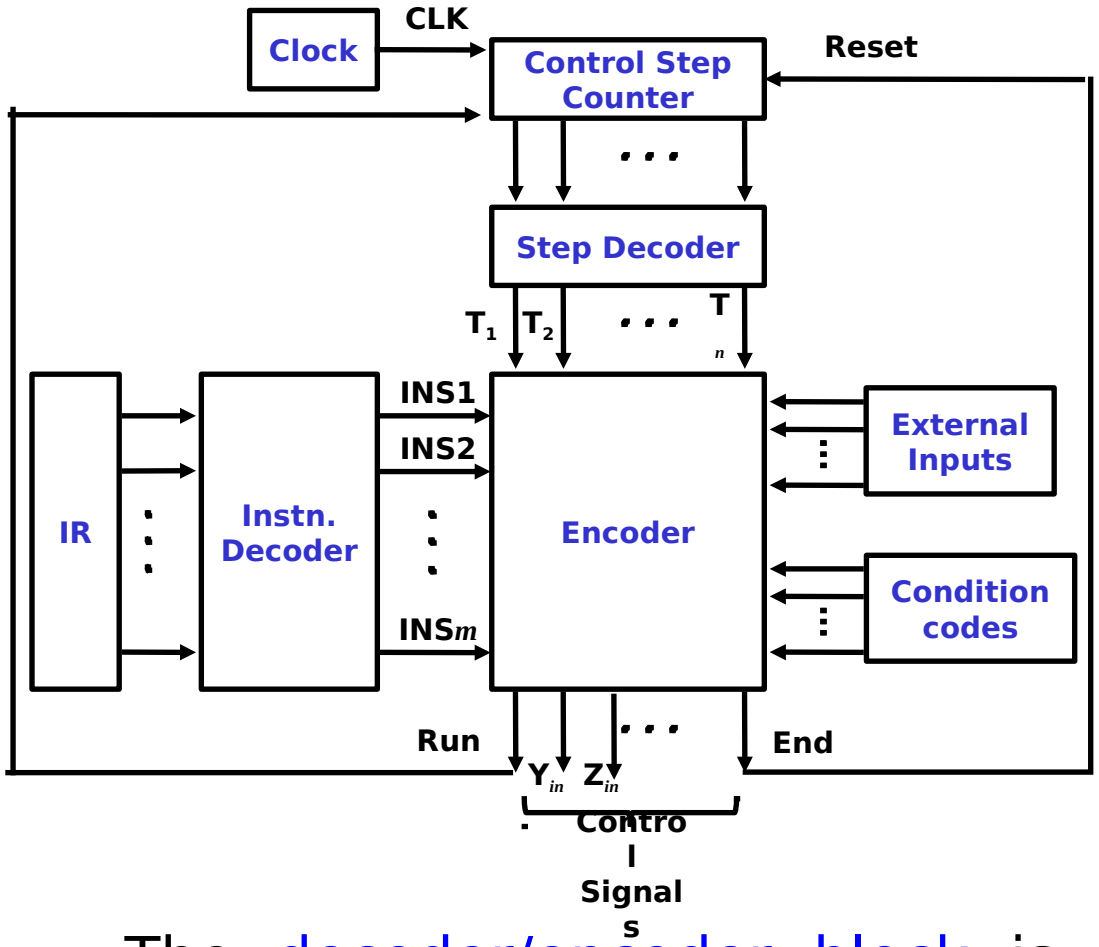
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC

T6 MDR_{out} , Select Y, Add, Z_{in}

T7 Z_{out} , $R1_{in}$, End

- Counter can be used to keep track of the control steps
- Each state or count, of this counter corresponds to one control step

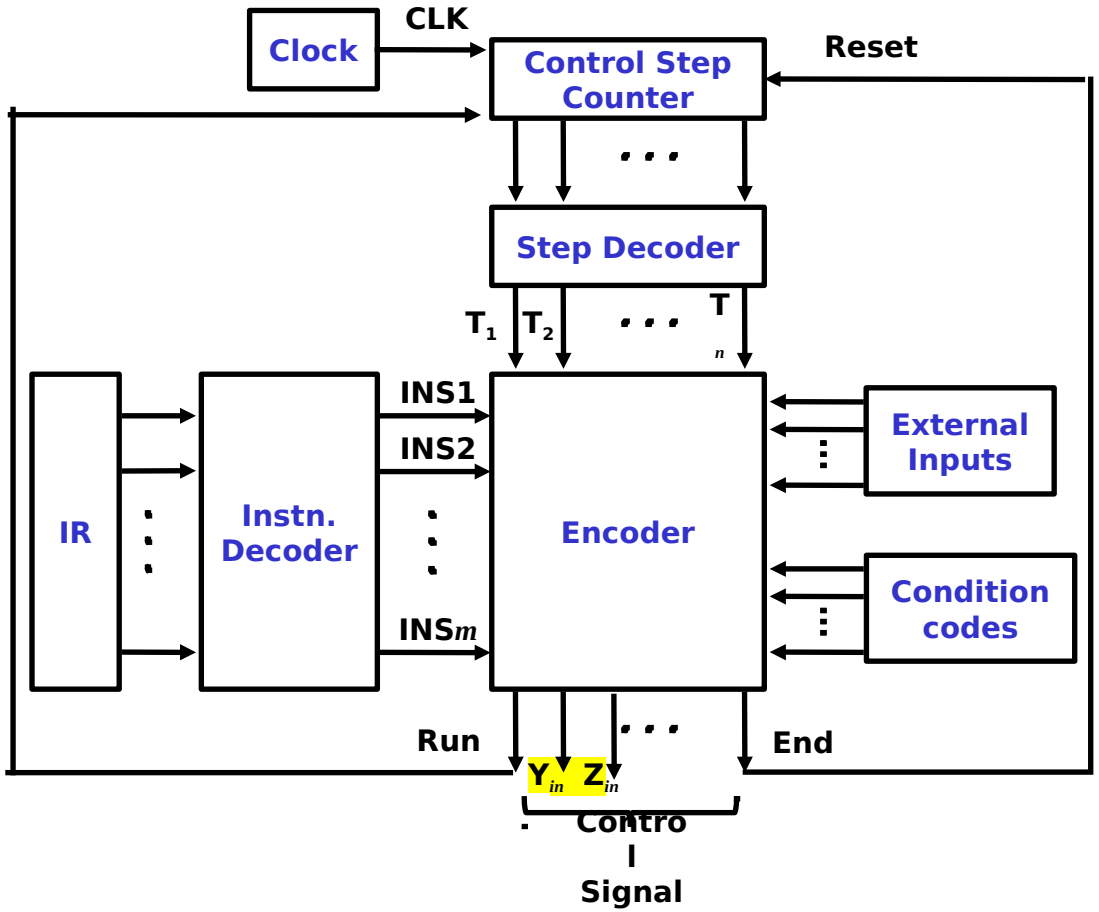
Hardwired Control Unit Organization



- Control signals are obtained using:
  - Contents of control step counter
  - Contents of the instruction register
  - Contents of condition flags
  - External input signals like MFC and interrupt requests

- The decoder/encoder block is a combinational circuit that generates the required control outputs, depending on the state of all its inputs

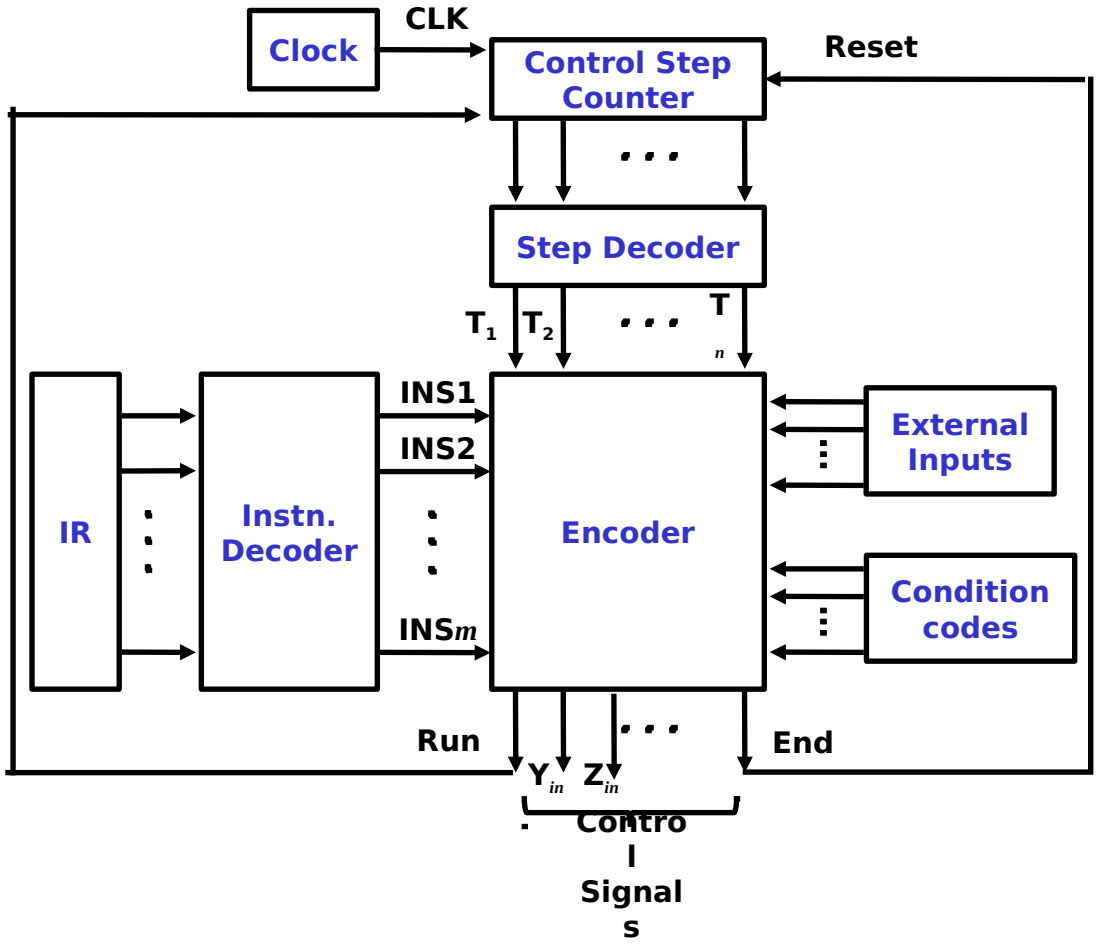
Hardwired Control Unit Organization



- Step decoder:
  - Provides a separate signal line for each step or time slot in the control sequence
- Instruction decoder:
  - Its output consists of separate lines for each machine instruction

- For any instruction in IR, one of the output lines INS1 to INSm is selected (i.e. set to 1) and all other lines are set to 0

Hardwired Control Unit Organization



- Encoder:
  - Input signals to encoder block are combined to generate individual control signals Y_{in}, Z_{in}, PC_{out}, Add, End and so on

Design of Encoder

- Control sequence for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC

T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read

T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC

T6 MDR_{out} , Select Y, Add, Z_{in}

T7 Z_{out} , $R1_{in}$, End

Example: Logic for Generating control signal, Z_{in}

$Z_{in} = T1 + T6 . ADD + T4.JMP + \ldots$

- Control sequence for JMP next

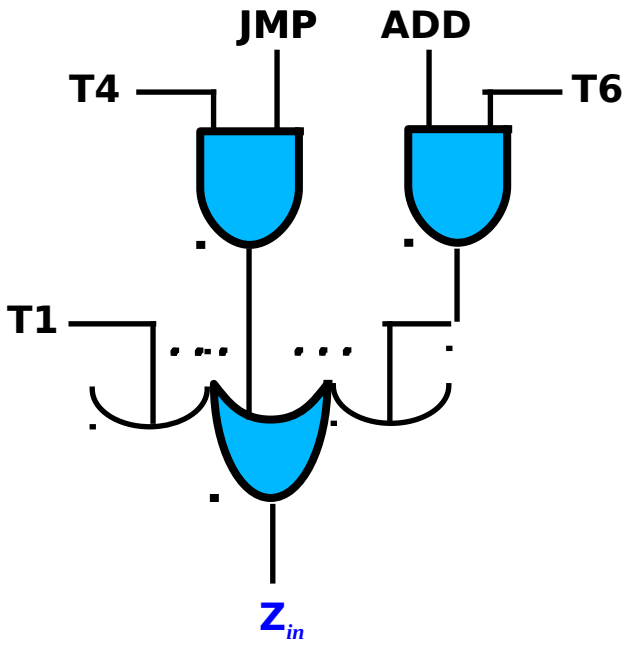
T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC

T3 MDR_{out} , IR_{in}

T4 $Offset\_filed\_of\_IR_{out}$, Add, Select Y, Z_{in}

T5 Z_{out} , PC_{in} , End



Design of Encoder

- Example: Generating control signal, End

$$End = T7.ADD + T5.JMP + (T5.N + T4.\overline{N}).JN + \ldots$$

- Control sequence for JN next

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

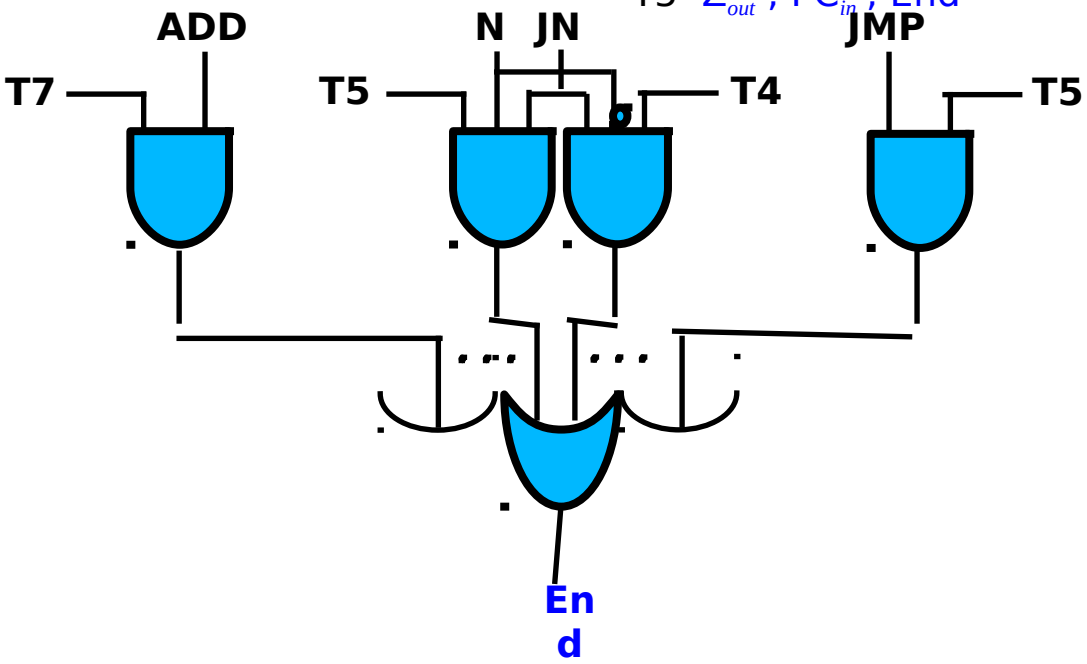
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC

T3 MDR_{out} , IR_{in}

T4 $Offset\_filed\_of\_IR_{out}$, Add, Select Y, Z_{in} ,

If N=0 then End

T5 Z_{out} , PC_{in} , End



Generation of Control Signals

- Example: T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

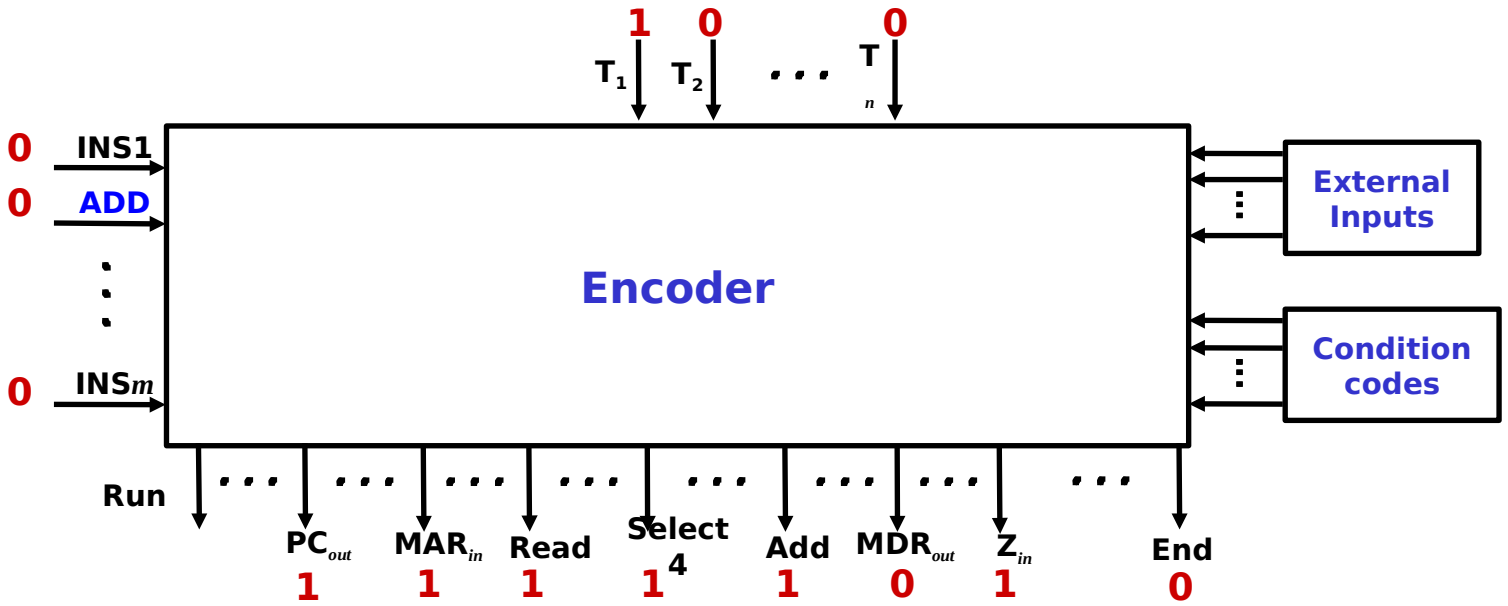
PC_{out} = T1 + ...

MAR_{in} = T1 + T4.ADD + ...

Select Y = T1 + ...

Add = T1 + T6.ADD + T4.(JMP + JN + ...) + ...

Z_{in} = T1 + T6.ADD + T4.JMP + ...



Generation of Control Signals

- Example: ADD R1, [R2]

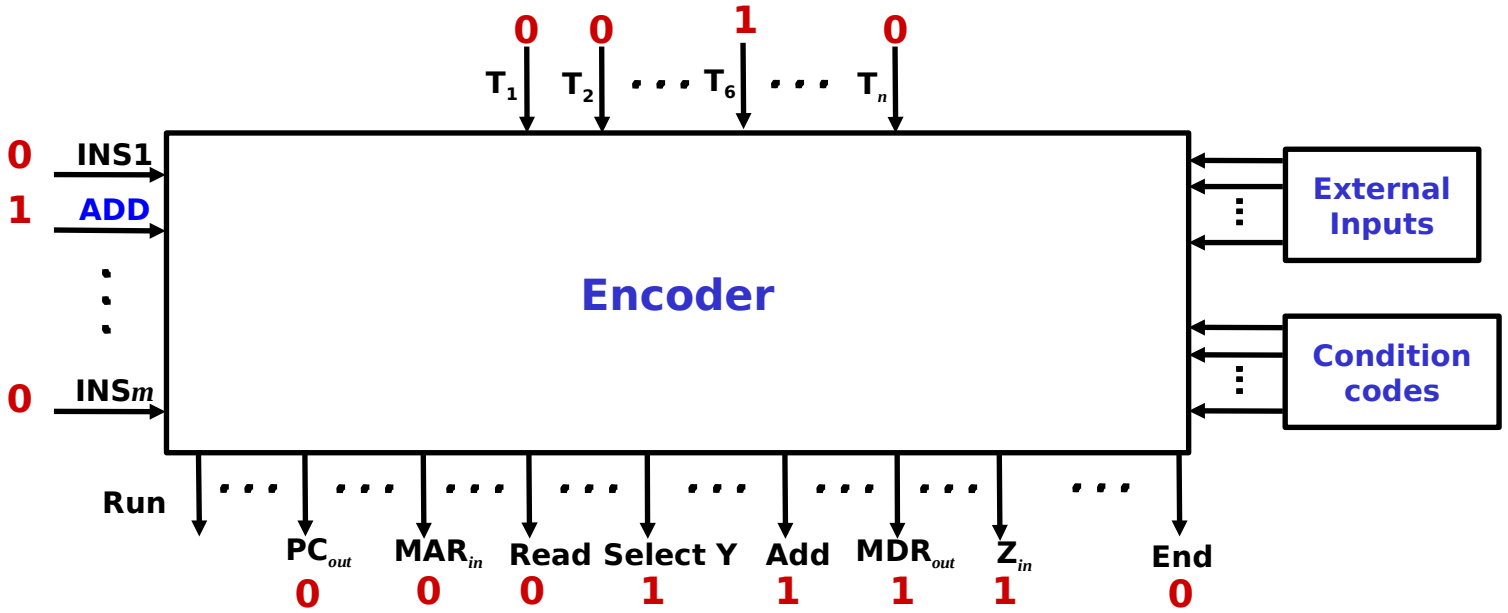
  T6 MDR_{out} , Select Y, Add, Z_{in}

MDR_{out} = T3 + T6.ADD + …

Select Y = T6.ADD + T4. (JMP + JN + …) + …

Add = T1 + T6.ADD + T4.(JMP + JN + …) + …

Z_{in} = T1 + T6.ADD + T4.JMP + …



Advantages and Disadvantages

- The hardwired control unit operate in high speed
- Useful when the instructions in the set are limited and simple

- It has less flexibility
- It is used in RISC processors

Microprogrammed Control Unit

Microprogrammed Control Unit

- Control signals are generated by a program
- The control signals are stored as control word (CW) in a control memory (control store)
- Control sequence for execution of ADD R1, [R2]:

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC

T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read

T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC

T6 MDR_{out} , Select Y, Add, Z_{in}

T7 Z_{out} , $R1_{in}$, End

Control Word and Control Memory

- Control word is a word whose individual bits represents the various control signals

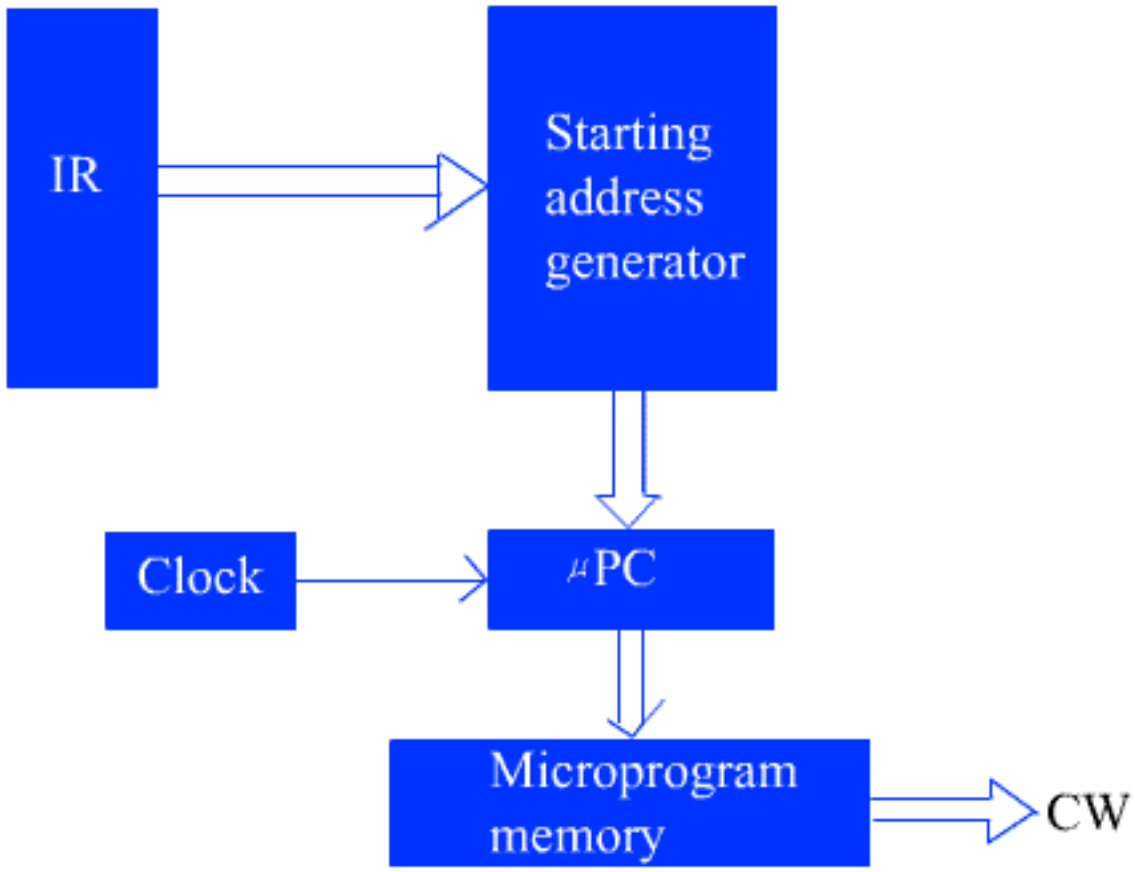
T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}

T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC

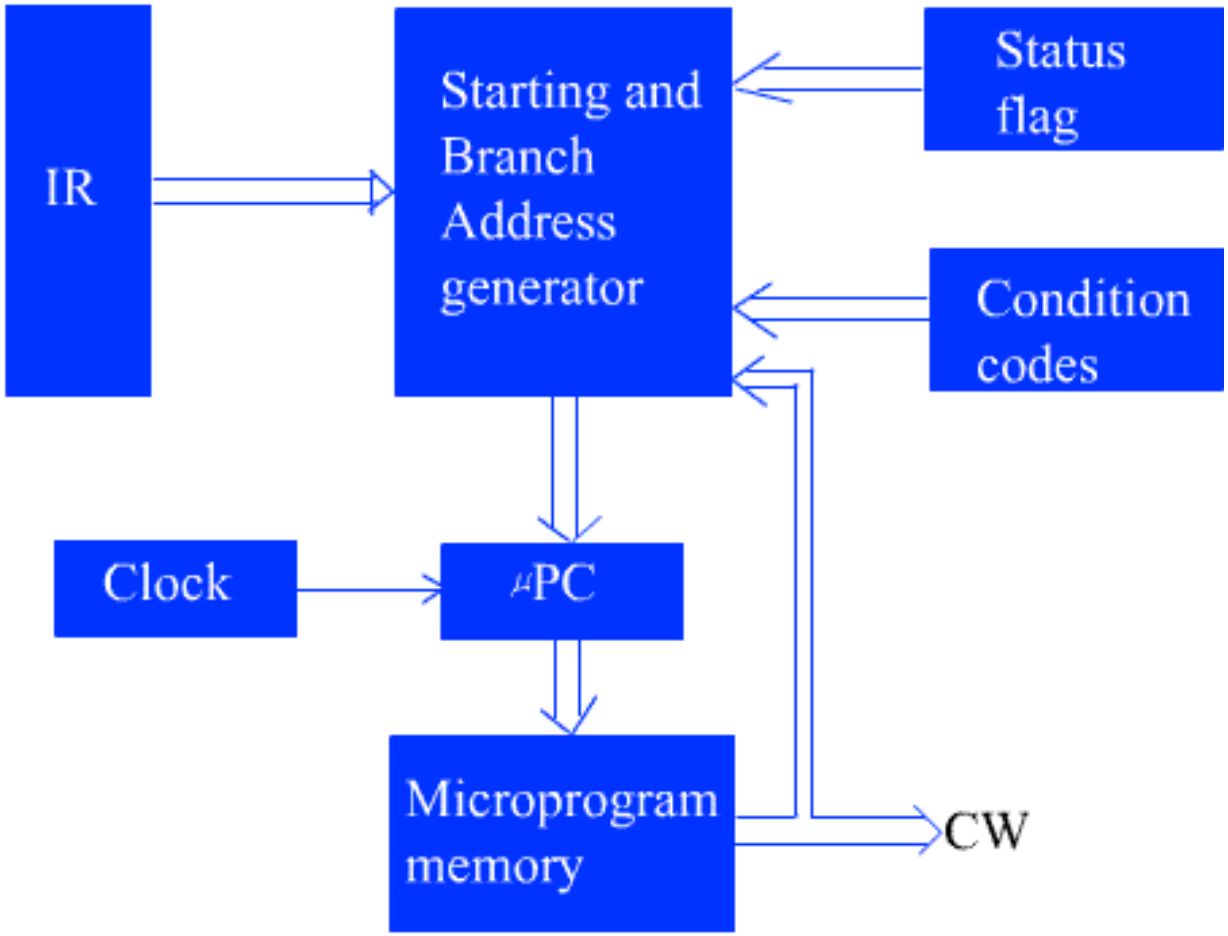
T3 MDR_{out} , IR_{in}

PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	...
0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	...
1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	...
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	...

- Each control word is called as microinstruction
- All the sequence of control words corresponding to a machine instruction is called microroutine
- All the microinstructions are stored in control memory in a specific location



Basic Organization of a microprogrammed control unit



Organization control unit to allow conditional branching in the microprogram

a) a common microprogram is used to fetch the instruction and is stored in a specific location and execution of each instruction start from that memory location.

b) After fetching of common microprogram, the starting address generator unit calculate the appropriate starting address of the microprogram for the instruction which is currently present in IR.

μPC is always incremented everytime a new microinstruction is fetched from the microprogram memory, except in the following situations :

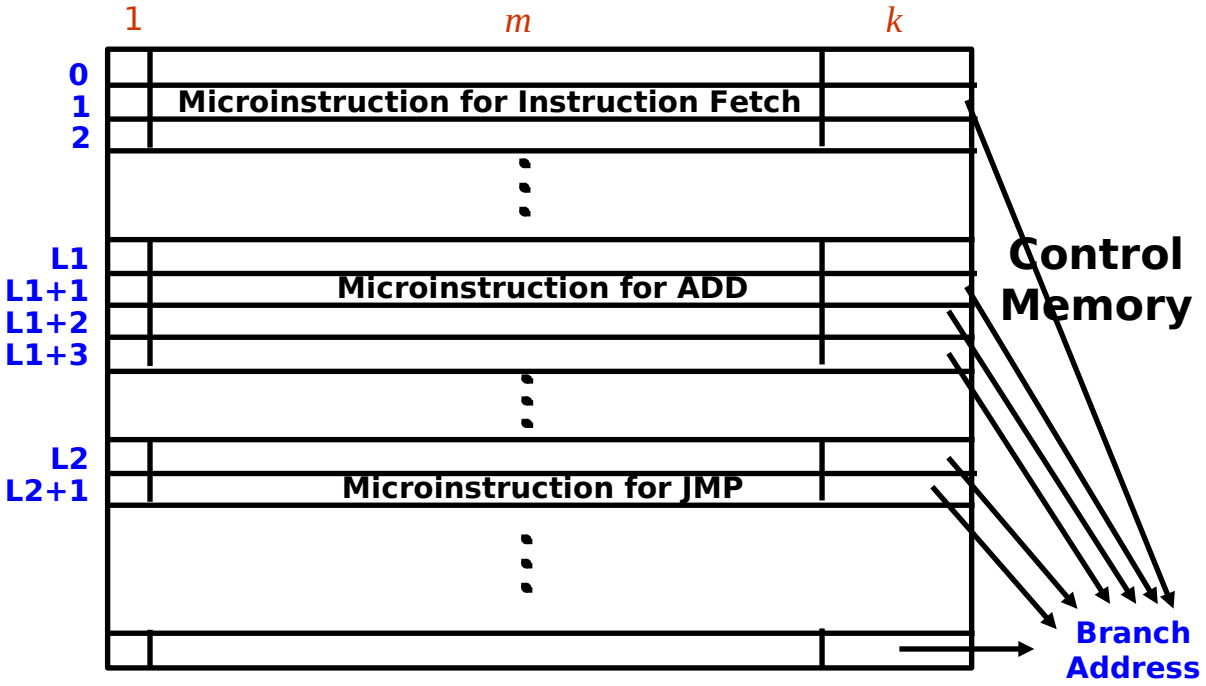
(a)When an End instruction is encountered

(b)When a new instruction is loaded into the IR

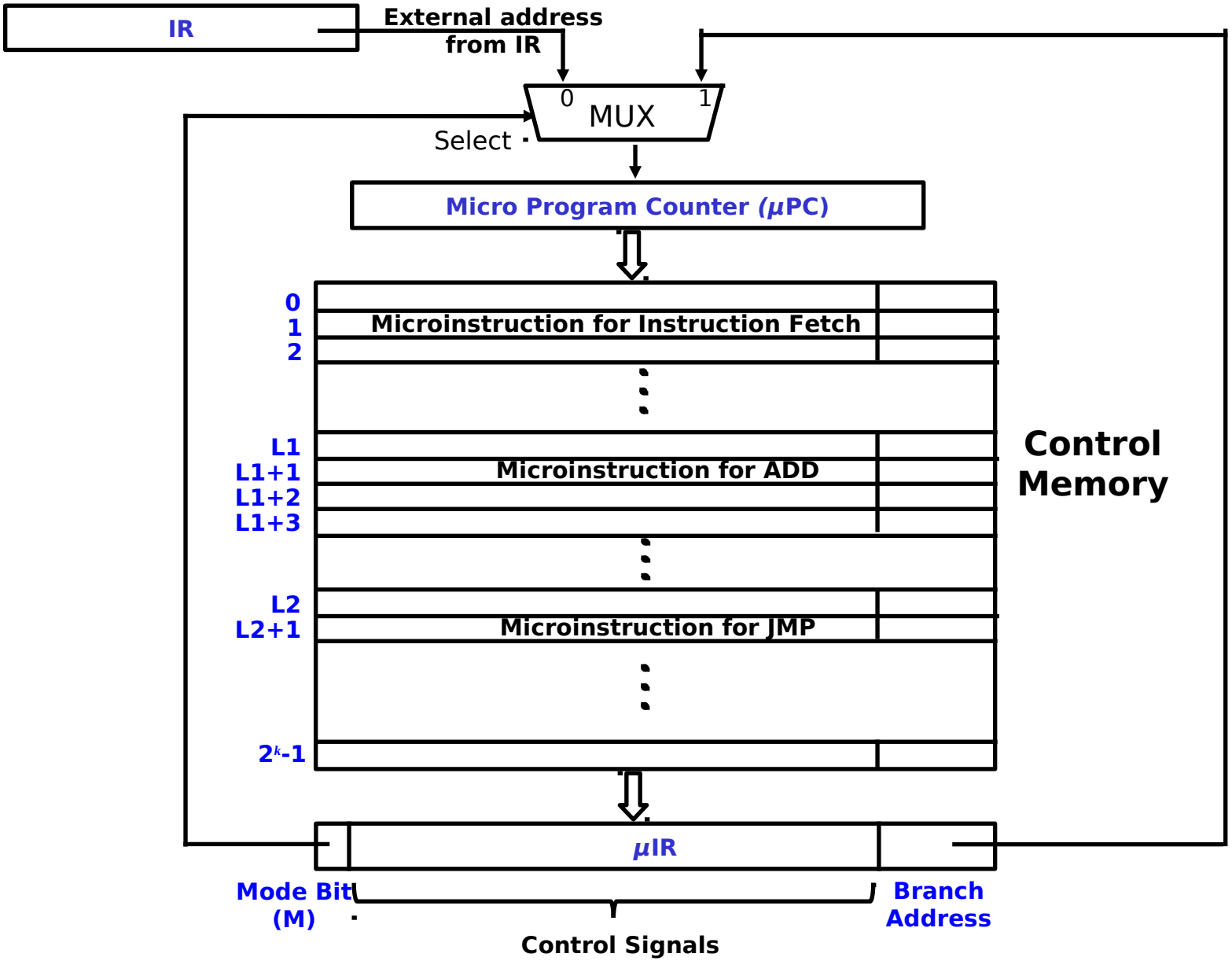
(c)When a branch microinstruction is encountered, and the branch condition is satisfied , the PC is loaded with the branch address.

Microprogrammed Control Unit Organization

- m = number of control signals
- k = branch address bits
- Length of a microinstruction $p = 1+m+k$



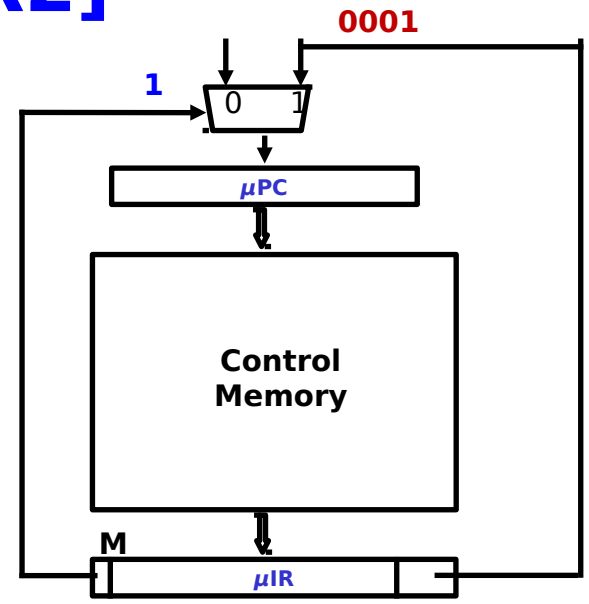
Microprogrammed Control Unit Organization



Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

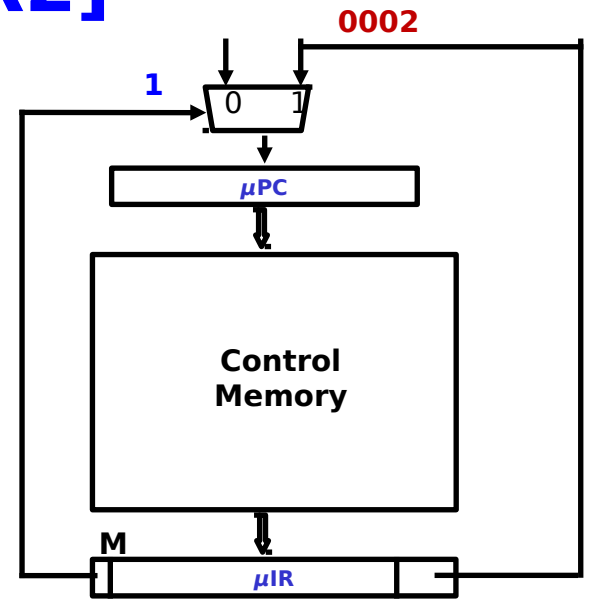


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001 •••
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0002 •••
0002	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx •••
										⋮									
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1 •••
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	L1+2 •••
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3 •••
L1+3	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0000 •••

Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

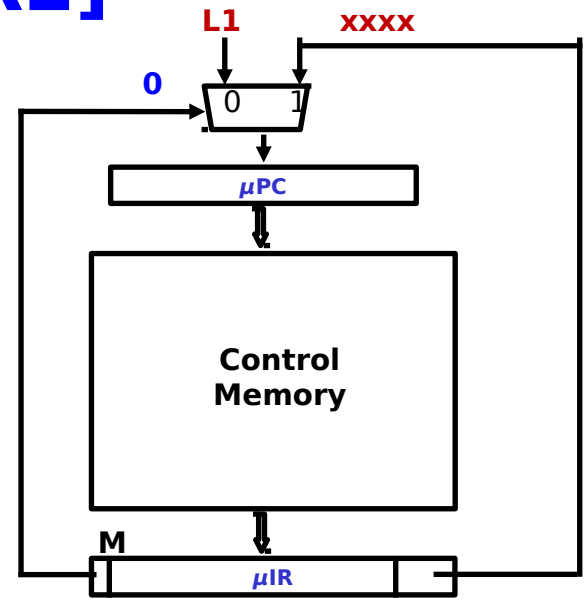


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0002
0002	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx
⋮																			
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	L1+2
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3
L1+3	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0000

Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

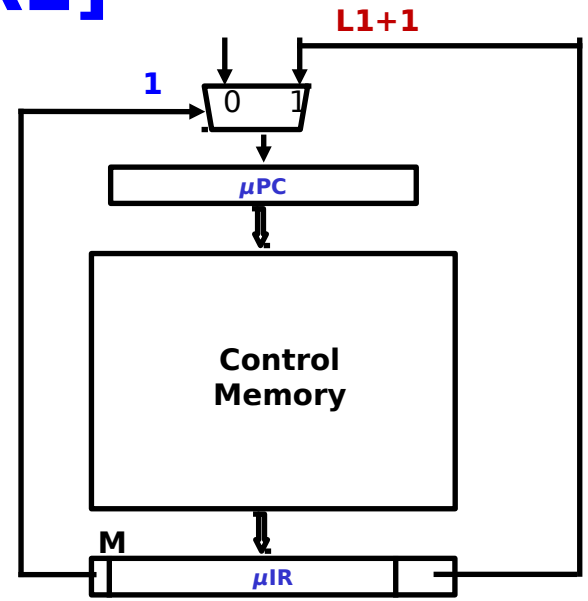


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0002
0002	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx
⋮																			
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	L1+2
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3
L1+3	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0000

Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

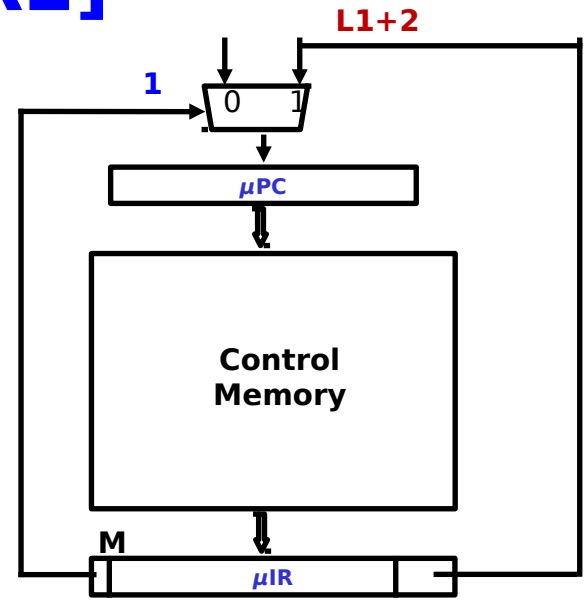


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0002
0002	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx
⋮																			
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	L1+2
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3
L1+3	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0000

Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

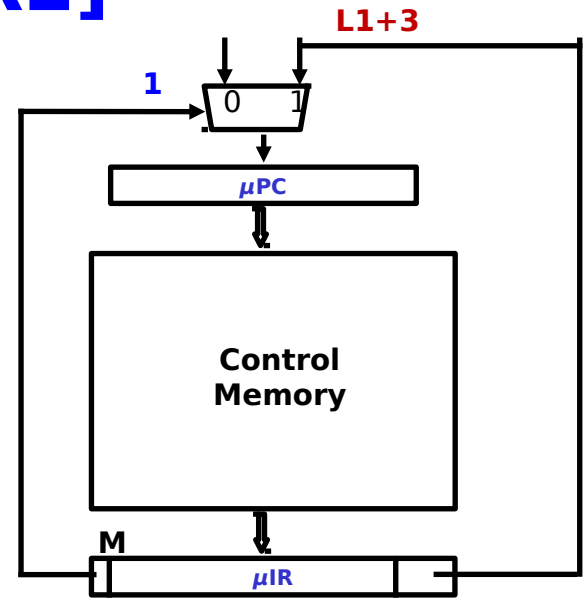


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0002
0002	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx
⋮																			
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	L1+2
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3
L1+3	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0000

Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

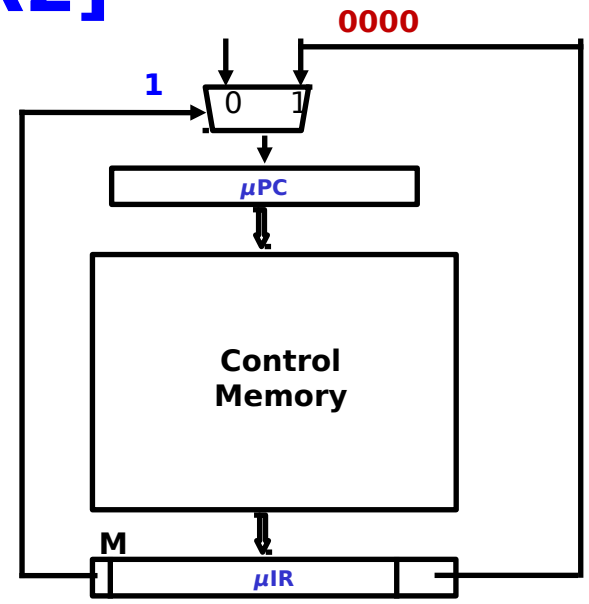


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0002
0002	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx
⋮																			
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	L1+2
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3
L1+3	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0000

Generation of Control Signals for ADD R1, [R2]

T1 PC_{out} , MAR_{in} , Read, Select 4, Add, Z_{in}
T2 Z_{out} , PC_{in} , Y_{in} , MDR_{inE} , WMFC
T3 MDR_{out} , IR_{in}

T4 $R2_{out}$, MAR_{in} , Read
T5 $R1_{out}$, Y_{in} , MDR_{inE} , WMFC
T6 MDR_{out} , Select Y, Add, Z_{in}
T7 Z_{out} , $R1_{in}$, End

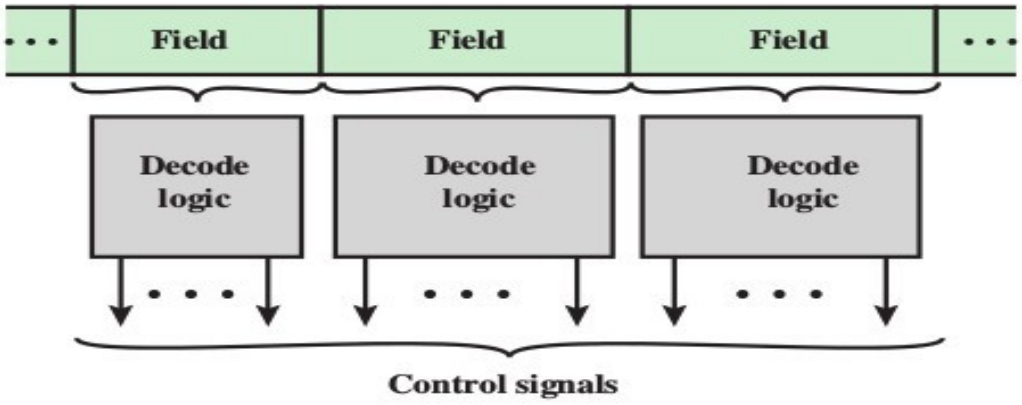


	M	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select 4	Select Y	Add	Z_{in}	Z_{out}	$R1_{in}$	$R1_{out}$	$R2_{out}$	WMFC	End	Branch Address
0000	1	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0001
0001	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0010
0010	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	xxxx
⋮																			
L1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	L1+1
L1+1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	L1+2
L1+2	1	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	L1+3
L1+3	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0000

Advantages and Disadvantages

- The Microprogrammed control unit is more compact and flexible
- Useful when the instructions in the set are complex and varying in length

- It is slow
- It is used in CISC processors



(a) Direct encoding