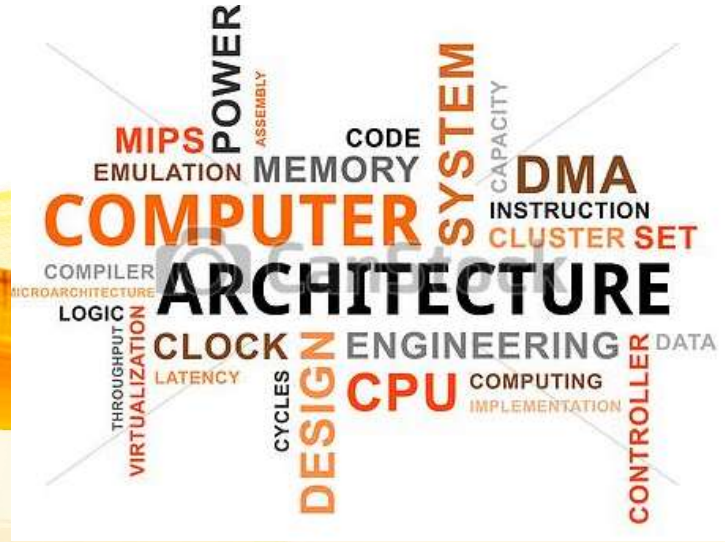
UNIT III
PROCESSOR AND PIPELINING

Fundamental concepts – Execution of a complete instruction – Multiple bus organization – **Hardwired control – Micro programmed control** – Pipelining: Basic concepts – Data hazards – Instruction hazards – Influence on Instruction sets – Data path and control consideration.

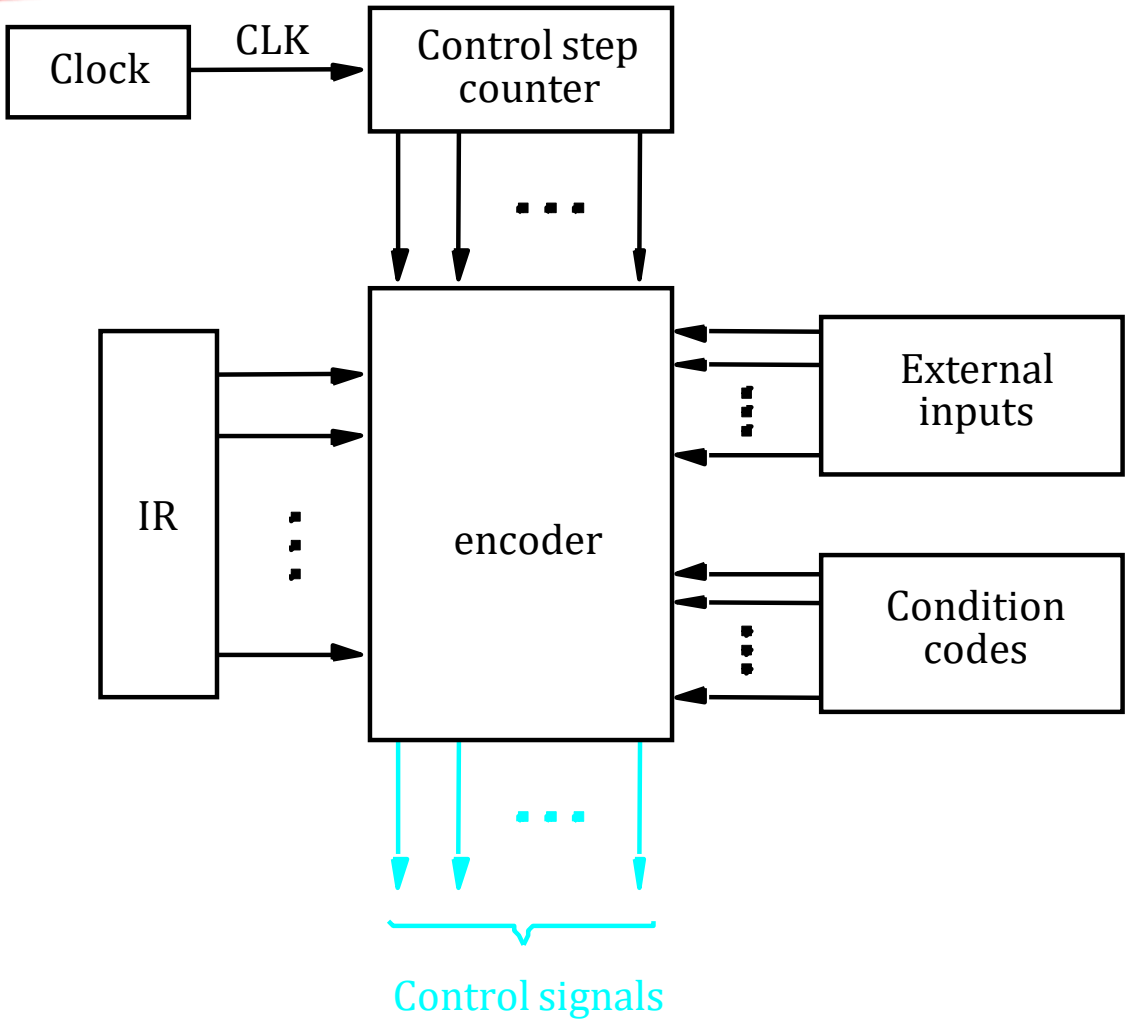


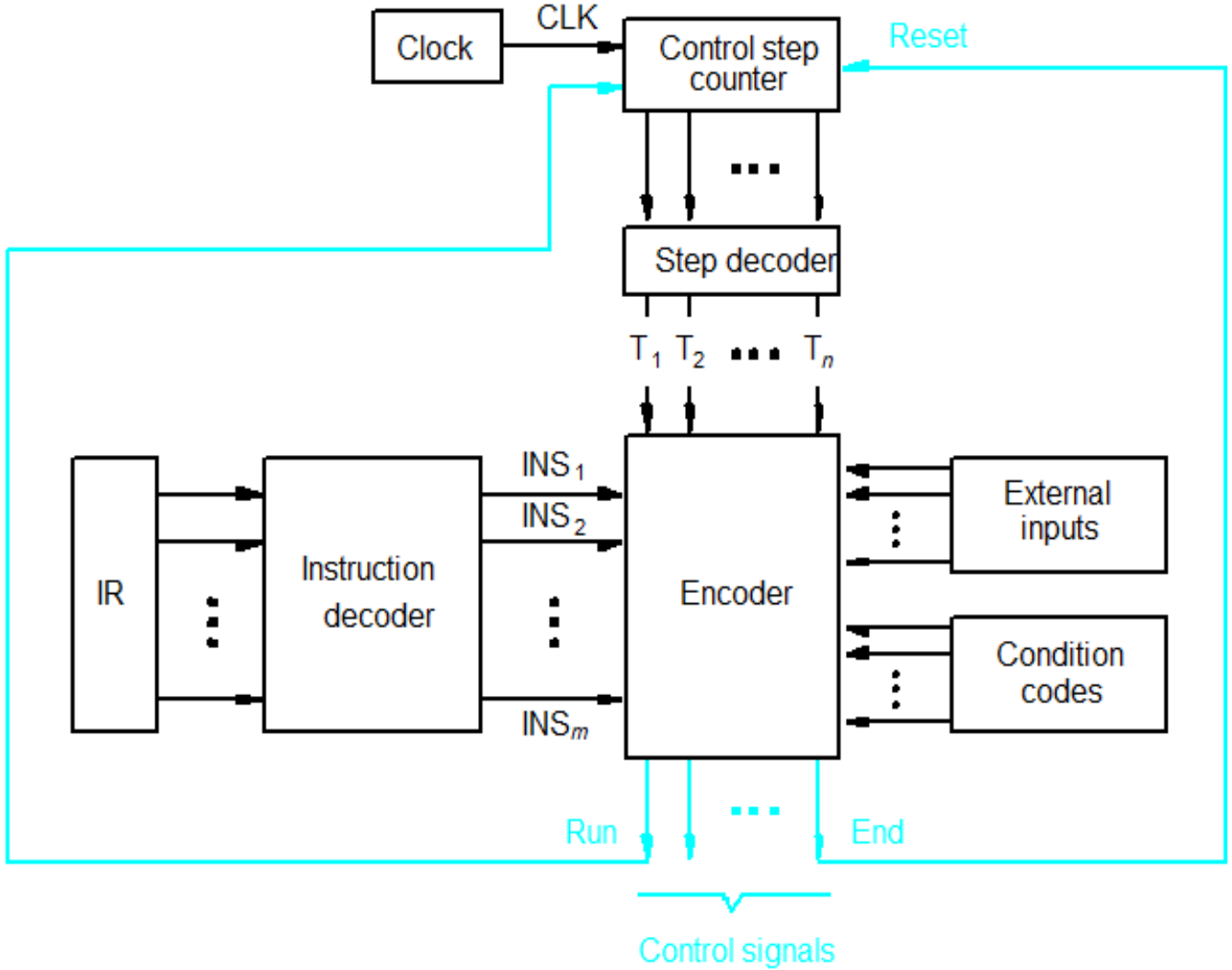
Recap the previous Class

Overview

- To execute instructions, the processor must have some means of generating the control signals needed in the proper sequence.
- Two categories: **hardwired control and microprogrammed control**
- Hardwired system can operate at high speed; but with little flexibility.

Control Unit Organization

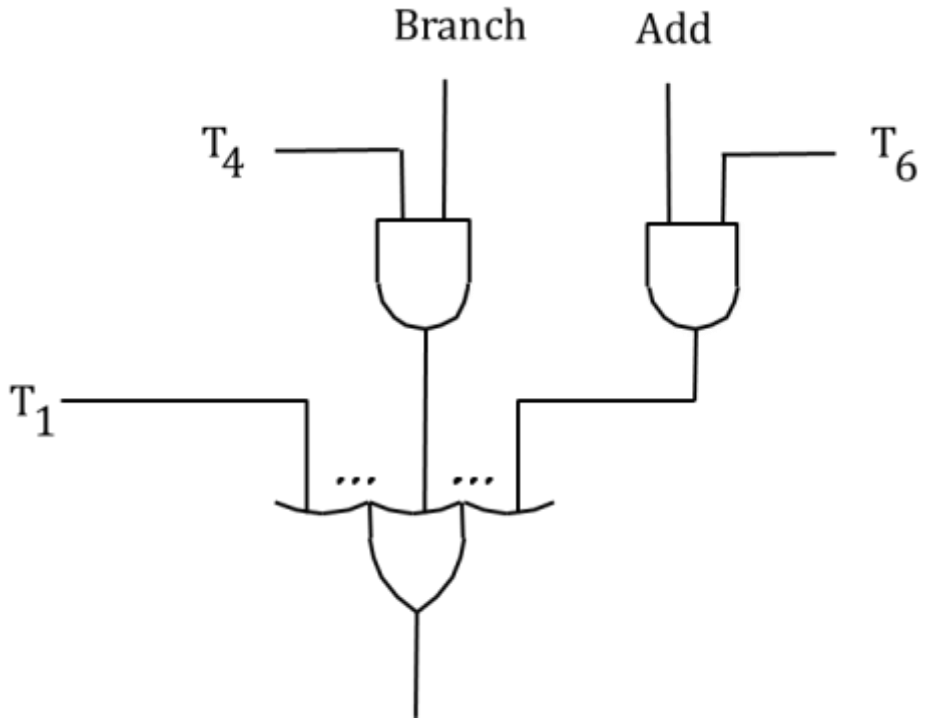


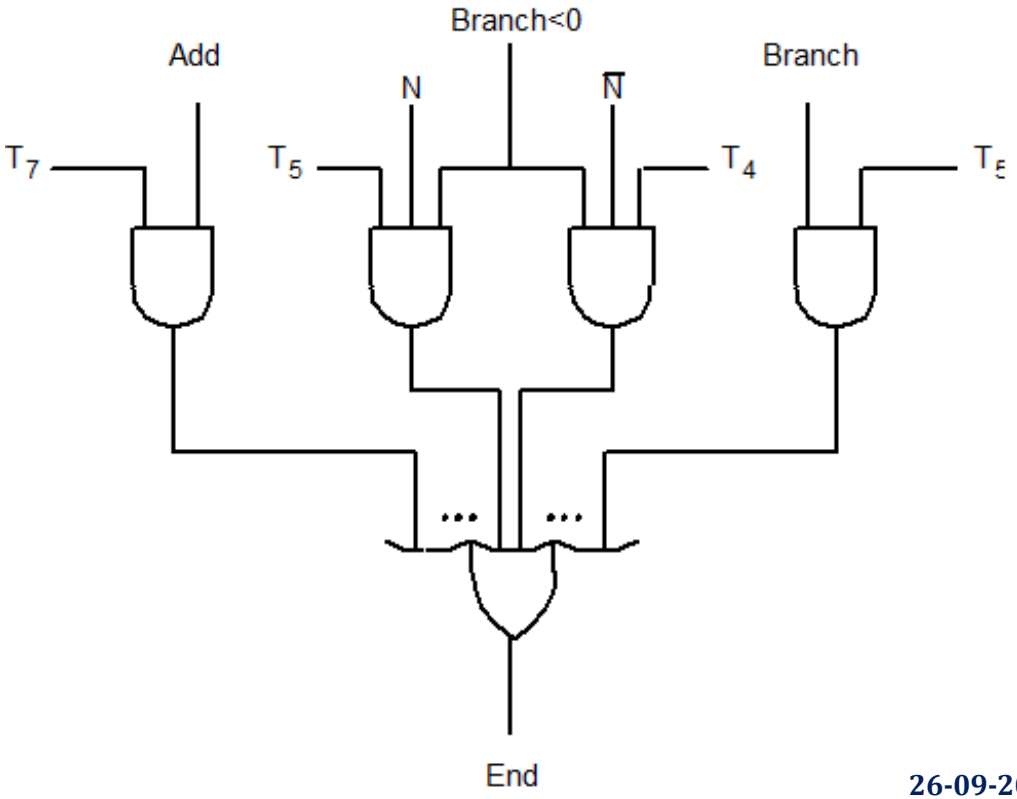


Generation of the Z_{in} and End control signal for the processor

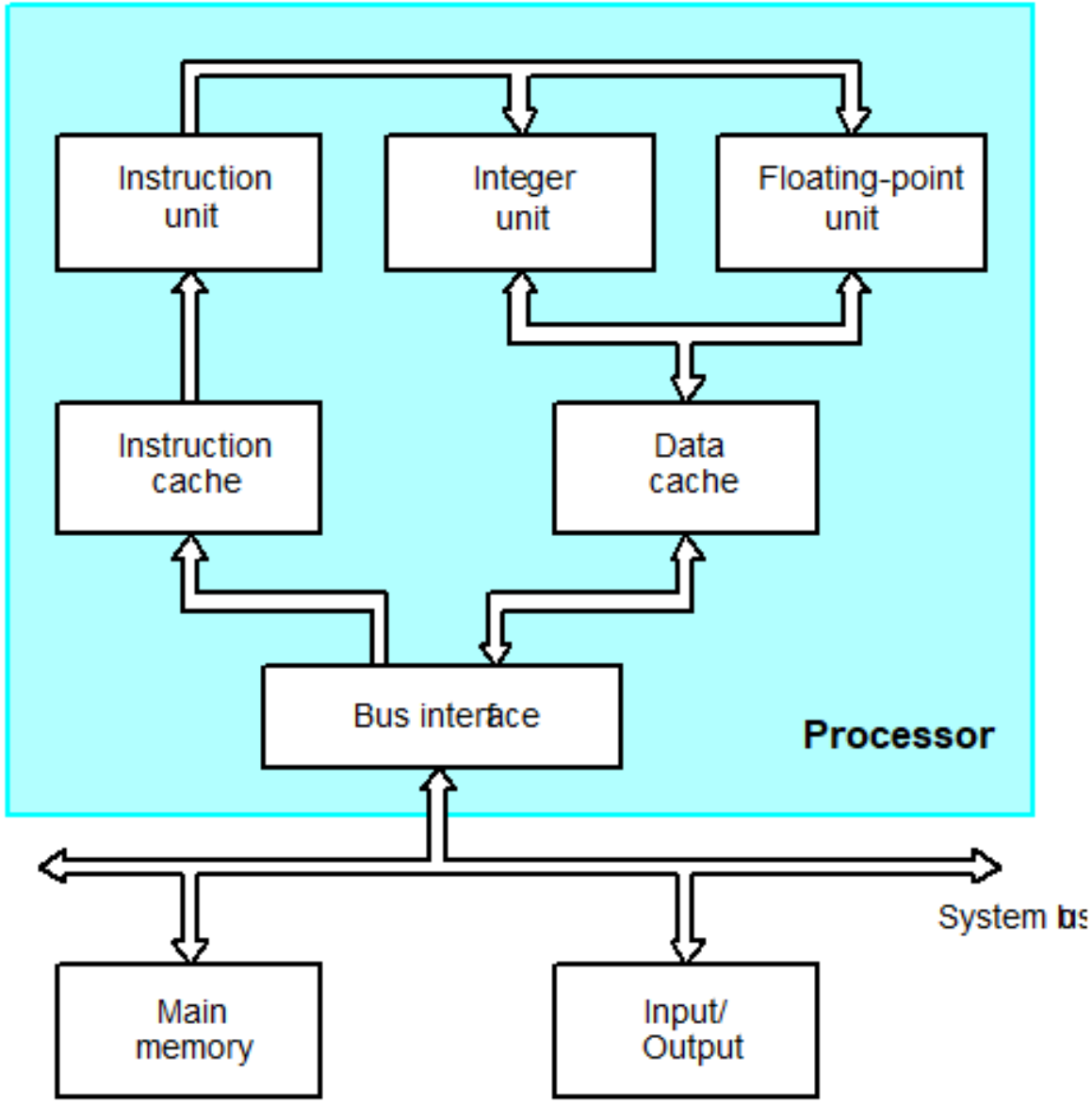
$$Z_{in} = T_1 + T_6 \bullet ADD + T_4 \bullet BR + \ldots$$

$$End = T7 \bullet ADD + T5 \bullet BR + (T5 \bullet N + T4 \bullet N) \bullet BRN + \ldots$$





A Complete Processor



Microprogrammed Control

- Control signals are generated by a program similar to machine language programs.
- Control Word (CW); microroutine; microinstruction

Micro - instruction	..	PC_{in}	PC_{out}	MAR_{in}	Read	MDR_{out}	IR_{in}	Y_{in}	Select	Add	Z_{in}	Z_{out}	$R1_{out}$	$R1_{in}$	$R3_{out}$	WMFC	End	..
1		0	1	1	1	0	0	0	1	1	1	0	0	0	0	0	0	
2		1	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	
3		0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	
4		0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	
5		0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	
6		0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	
7		0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	

Control sequence for the instruction

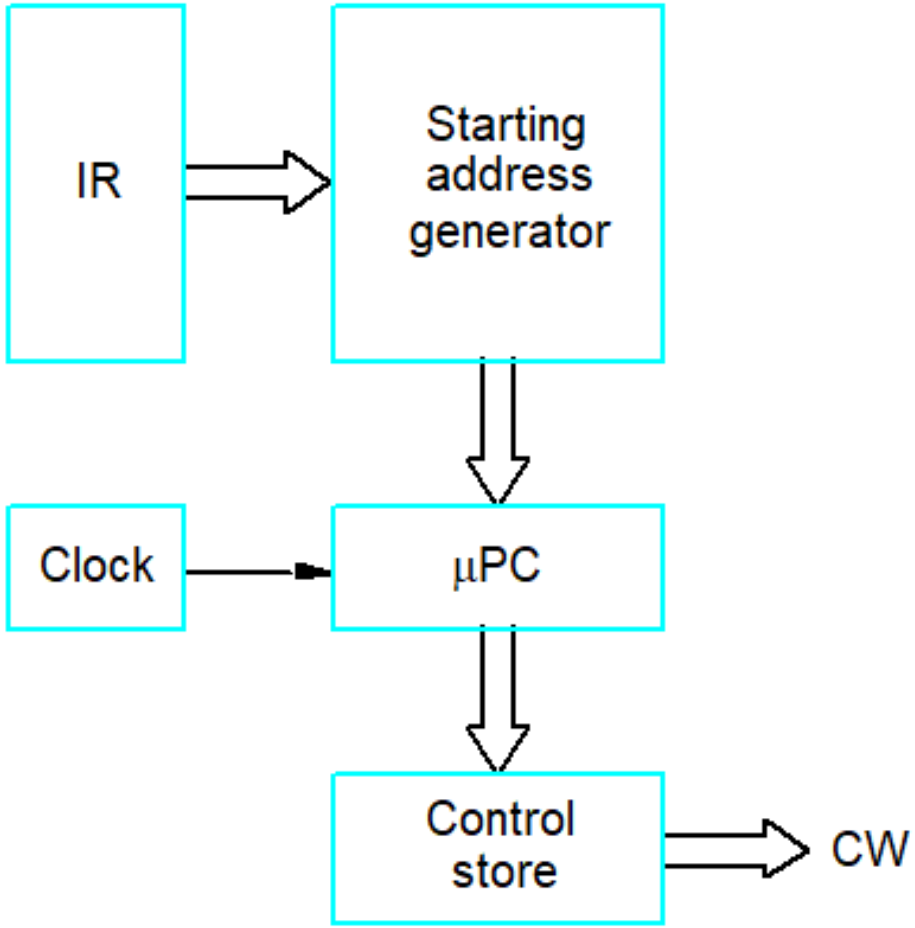
Add (R3),R2

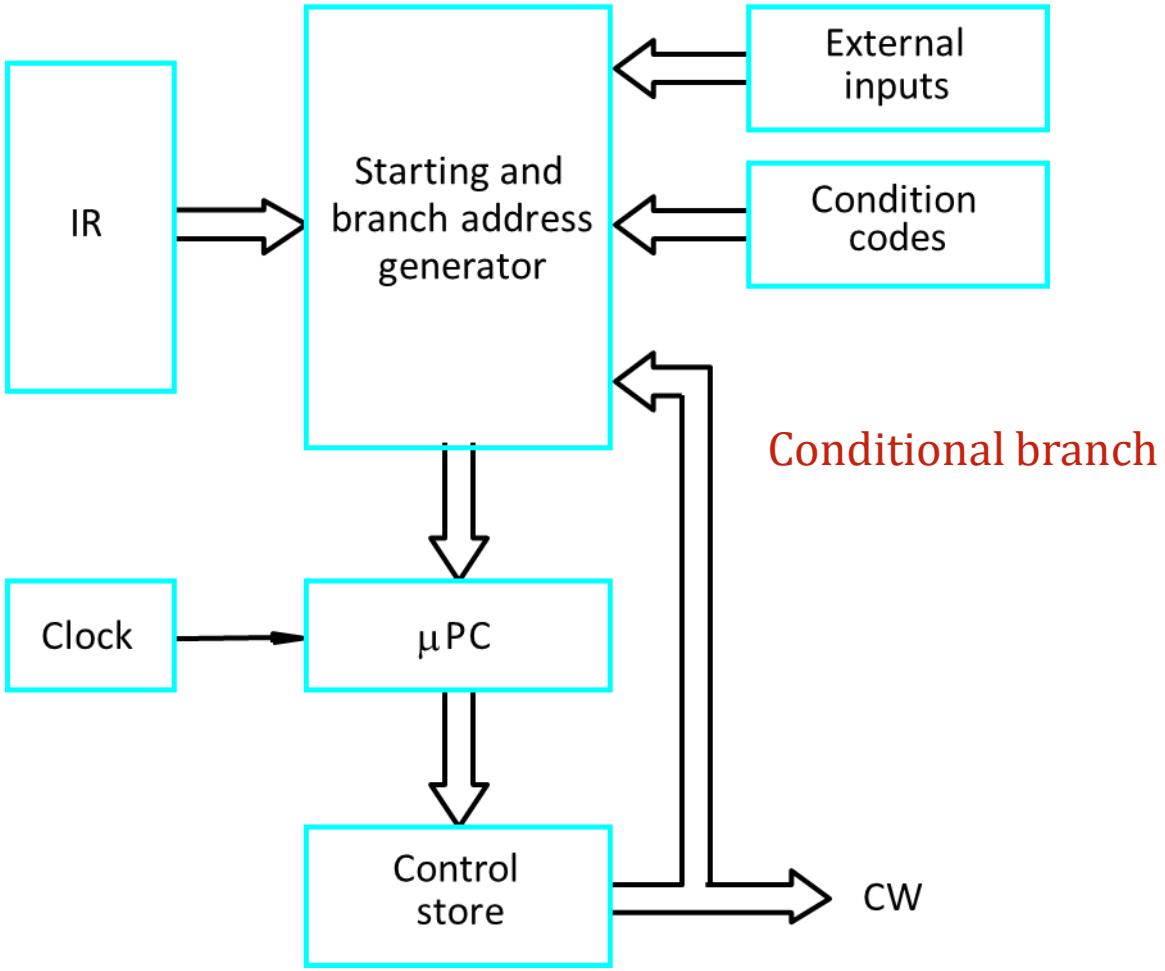
Step	Action
1	PC_{out} , MAR_{in} , Read, Select4,Add, Z_{in}
2	Z_{out} , PC_{in} , Y_{in} , WMF C
3	MDR_{out} , IR_{in}
4	$R3_{out}$, MAR_{in} , Read
5	$R1_{out}$, Y_{in} , WMF C
6	MDR_{out} , SelectY, Add, Z_{in}
7	Z_{out} , $R1_{in}$, End

Conditional branch

Address	Microinstruction
0	PC_{out} , MAR_{in} , Read, Select4, Add, Z_{in}
1	Z_{out} , PC_{in} , Y_{in} , WMF C
2	MDR_{out} , IR_{in}
3	Branch to starting address of appropriate
. . . . .	
25	If N=0, then branch to microinstruction 0
26	Offset-field-of-IR_{out} , SelectY, Add, Z_{in}
27	Z_{out} , PC_{in} , End

Basic organization of a microprogrammed control unit





Microinstructions

- A straightforward way to structure microinstructions is to assign one bit position to each control signal.
- However, this is very inefficient.
- The length can be reduced: most signals are not needed simultaneously, and many signals are mutually exclusive.
- All mutually exclusive signals are placed in the same group in binary coding.

Microinstructions

Microinstruction

F1	F2	F3	F4	F5	

F1 (4 bits)	F2 (3 bits)	F3 (3 bits)	F4 (4 bits)	F5 (2 bits)
0000: No transfer 0001: PC_{out} 0010: MDR_{out} 0011: Z_{out} 0100: $R0_{out}$ 0101: $R1_{out}$ 0110: $R2_{out}$ 0111: $R3_{out}$ 1010: $TEMP_{out}$ 1011: $Offset_{out}$	000: No transfer 001: PC_{in} 010: IR_{in} 011: Z_{in} 100: $R0_{in}$ 101: $R1_{in}$ 110: $R2_{in}$ 111: $R3_{in}$	000: No transfer 001: MAR_{in} 010: MDR_{in} 011: $TEMP_{in}$ 100: Y_{in}	0000: Add 0001: Sub ⋮ 1111: XOR 16 ALU functions	00: No action 01: Read 10: Write

F6	F7	F8	...

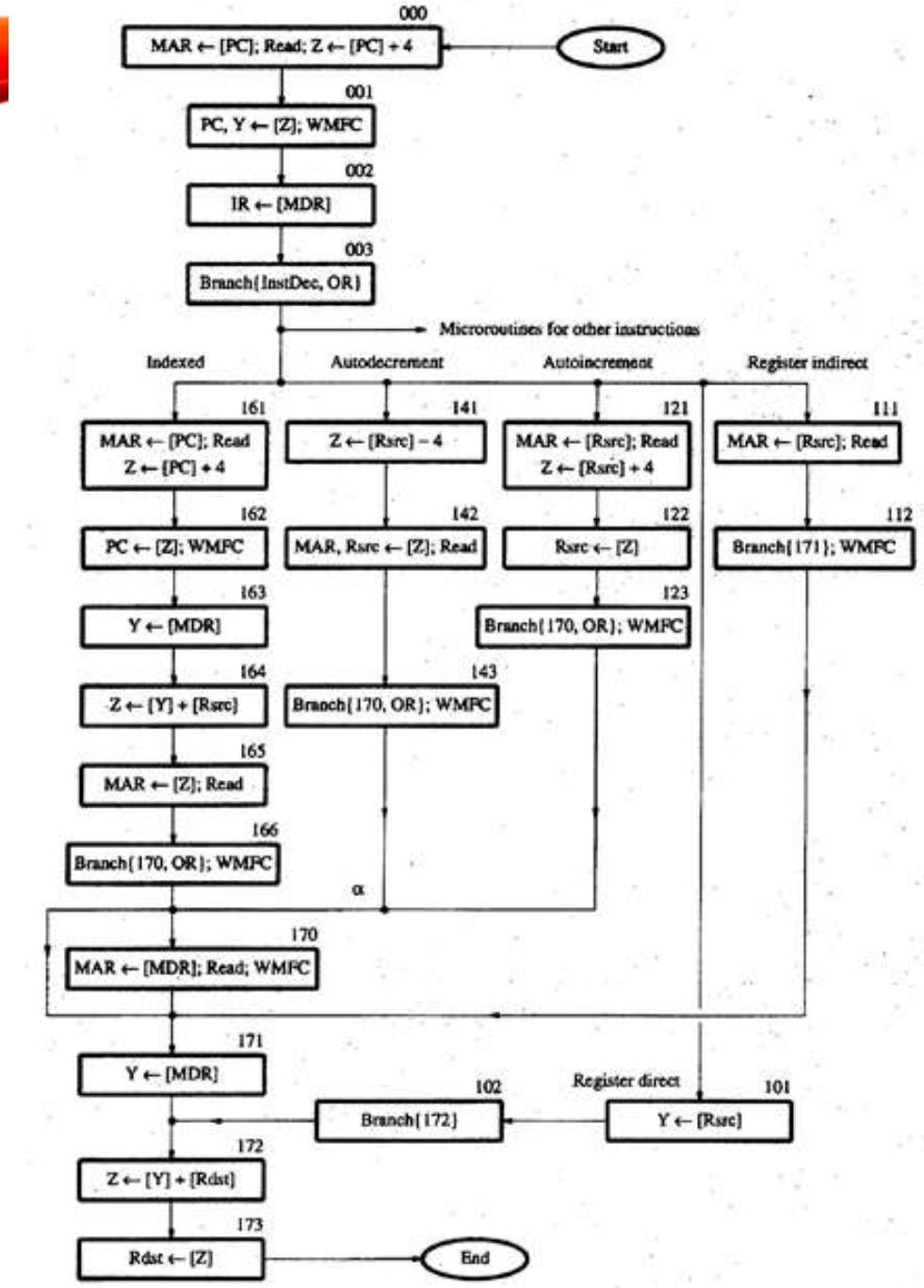
F6 (1 bit)	F7 (1 bit)	F8 (1 bit)
0: SelectY 1: Select4	0: No action 1: WMFC	0: Continue 1: End

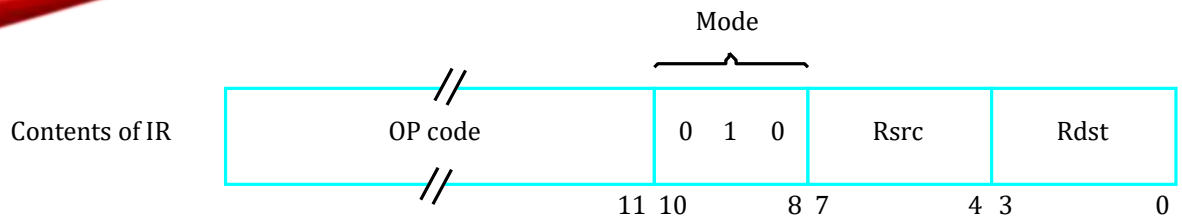
Microprogram Sequencing

Add src, Rdst

Four addressing modes:

- Register,
- Autoincrement,
- Autodecrement
- Indexed (with indirect forms).





Address (octal)	Microinstruction
000	PC_{out}, MAR_{in}, Read, Select 4, Add, Z_{in}
001	Z_{out}, PC_{in}, Y_{in}, WMFC
002	MDR_{out}, IR_{in}
003	mBranch { mPC ¬ 101 (from Instruction decoder); $mPC_{5,4} \neg [IR_{10,9}]$; $mPC_3 \neg [\overline{IR_{10}}] \times [\overline{IR_9}] \times [IR_8]$}
121	$Rsrc_{out}$, MAR_{in}, Read, Select4, Add, Z_{in}
122	Z_{out}, $Rsrc_{in}$
123	mBranch { $mPC \neg 170$; $mPC_0 \neg [\overline{IR_8}]$}, WMFC
170	MDR_{out}, MAR_{in}, Read, WMFC
171	MDR_{out}, Y_{in}
172	$Rdst_{out}$, SelectY , Add, Z_{in}
173	Z_{out}, $Rdst_{in}$, End

Microinstructions with Next-Address Field

- A powerful alternative approach is to include an address field as a part of every microinstruction to indicate the location of the next microinstruction to be fetched.

Microinstructions with Next-Address Field

Microinstruction

F0	F1	F2	F3

F0 (8 bits)	F1 (3 bits)	F2 (3 bits)	F3 (3 bits)
Address of next microinstruction	000: No transfer 001: PC_{out} 010: MDR_{out} 011: Z_{out} 100: $Rsrc_{out}$ 101: $Rdst_{out}$ 110: $TEMP_{out}$	000: No transfer 001: PC_{in} 010: IR_{in} 011: Z_{in} 100: $Rsrc_{in}$ 101: $Rdst_{in}$	000: No transfer 001: MAR_{in} 010: MDR_{in} 011: $TEMP_{in}$ 100: Y_{in}

F4	F5	F6	F7

F4 (4 bits)	F5 (2 bits)	F6 (1 bit)	F7 (1 bit)
0000: Add 0001: Sub ⋮ 1111: XOR	00: No action 01: Read 10: Write	0: SelectY 1: Select4	0: No action 1: WMFC

F8	F9	F10

F8 (1 bit)	F9 (1 bit)	F10 (1 bit)
0: NextAdrs 1: InstDec	0: No action 1: OR_{mode}	0: No action 1: OR_{indsrc}

Implementation of the Microroutine


Thank You