Computer System Architecture

Ms. Yuzana Hlaing
M.E(IT), Ph.D. Candidate(thesis)
Lecturer
Computer Engineering and Information Technology Dept.
Yangon Technological University

Course Schedule

Periods	Lectures
Week-1	Introduction to Computer System Architecture
Week-2	Data Presentations
Week-3	Register Transfer and Microoperations
Week-4	Basic Computer Organization and Design
Week-5	Programming the Basic Computer
Week-6	Microprogrammed Control
Week-7	Central Processing Unit
Week-8	Pipeline and Vector Processing
Week-9	Computer Arithmetic
Week-10	Input-Output Organization
Week-11	Memory Organization
Week-12	Multiprocessors

Lecture-6

Microprogrammed Control

Lecture-6
Microprogrammed Control

- What is Control Memory?
- Address Sequencing
- Microprogram Example
- Design of Control Unit

What is Control Memory?

Hardwired Control Technique

The control signals are generated by hardware using conventional logic design techniques, this control unit is said to be hardwired.

Microprogramming Control Technique

The principle of microprogramming is an elegant and systematic method for controlling the microoperation sequences in a digital computer.

Control Word

The group of control variables that are represented by a string of **1**'s or **0**'s is called a control word.

Microprogrammed Control Unit

A control unit whose binary control variables are stored in memory is called a microprogrammed control unit.

Microinstruction

Microinstruction is an instruction stored in control memory. It specifies one or more microoperations for the system.

Microprogram

A sequence of microinstructions constitutes a microprogram that is also called microcode. The microprogram consists of microinstructions that specify various internal control signals for execution of register microoperations.

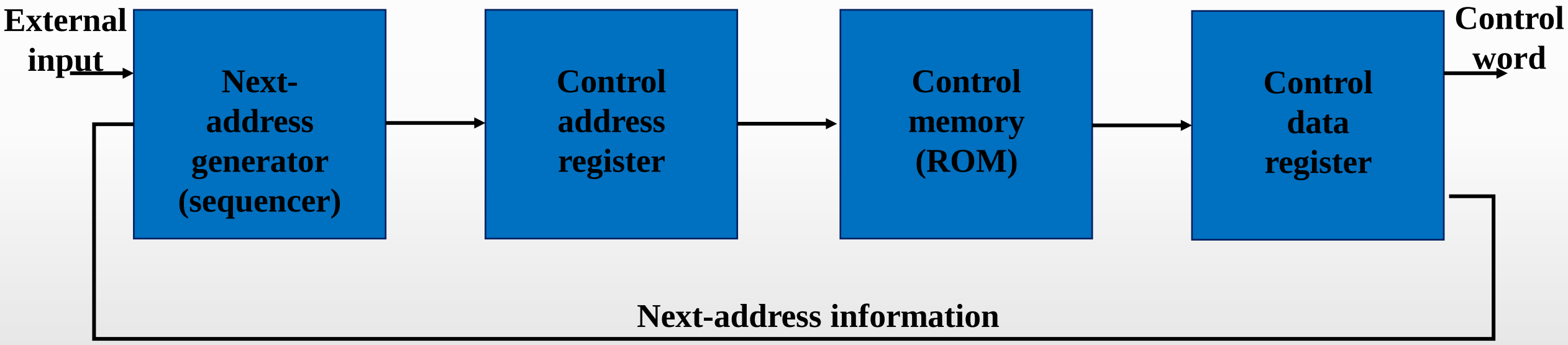
Subroutines

Subroutines are programs that are used by other routines to accomplish a particular task. A subroutine can be called from any point within the main body of the microprogram.

Microprogrammed Control Organization

- There are four main components to design a microprogrammed control organization in a basic computer.
- The next address generator is called a microprogram sequencer , as it determines the address sequence that is read from control memory.
- Control address register specifies the address of the microinstruction which contains a control word that specifies one or more microoperations for the data processor.
- In the control memory (ROM), which all control information is permanently stored.
- The control data register holds the microinstruction read from memory.
- The microprogrammed control organization requires a two-phase clock, with one clock applied to the address register and the other to the data register.
- This system can operate without control data register by applying a single-phase clock to the address register.
- The control word and next-address information are taken directly from the control memory.

Microprogrammed control organization

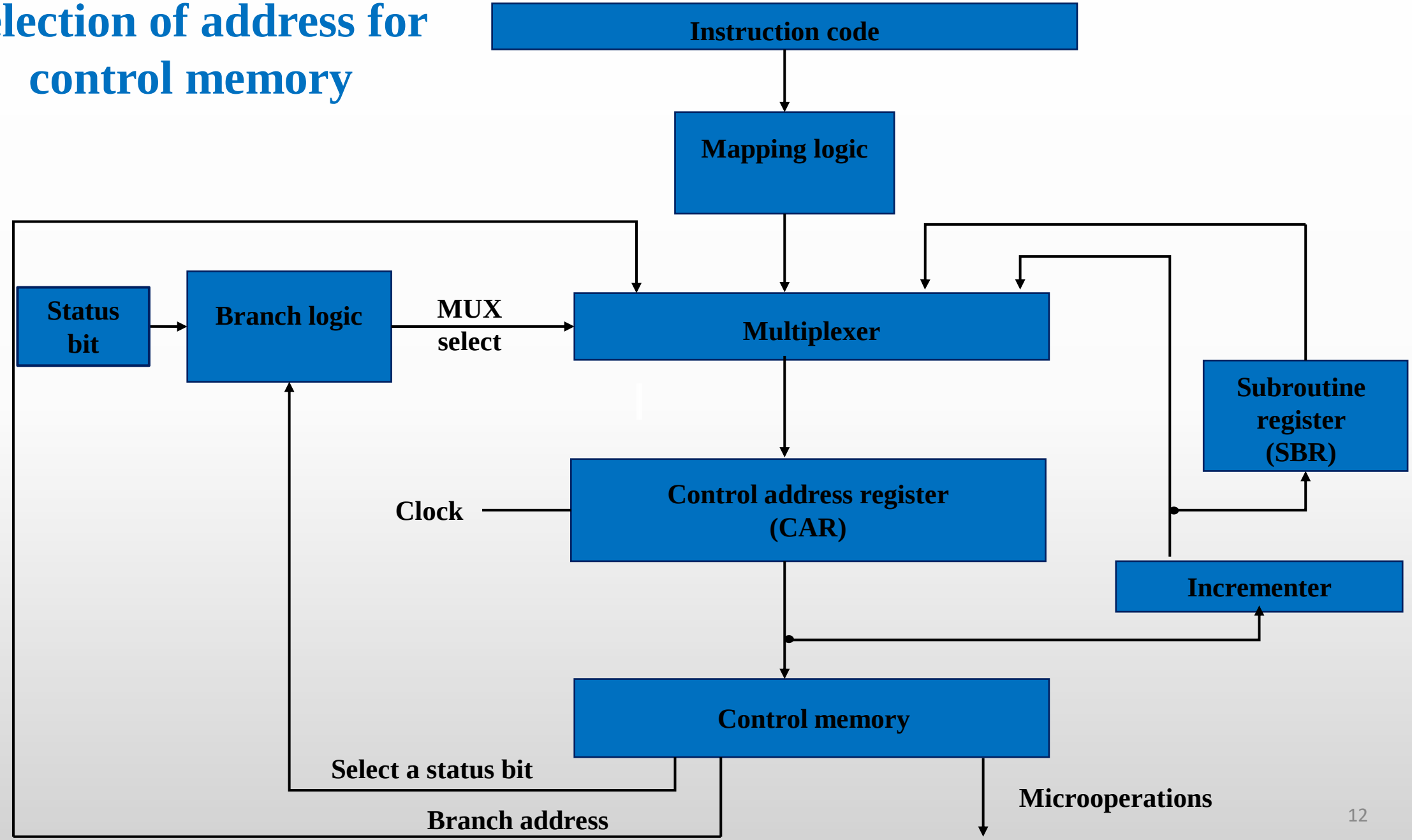


Address Sequencing

Address Sequencing

- The address sequencing capabilities required in a control memory are:
  1. Incrementing of the control address register.
  2. Unconditional branch or conditional branch, depending on status bit conditions.
  3. A mapping process from the bits of the instruction to an address for control memory.
  4. A facility for subroutine call and return.
- For selecting the next microinstruction address, the workflow of a control memory and the associated hardware is designed in the following figure.
- The microinstruction in control memory contains a set of bits to initiate microoperations in computer register and other bits to specify the method by which the next address is obtained.
- The diagram shows four different paths from which the control address register (CAR) receives the address.

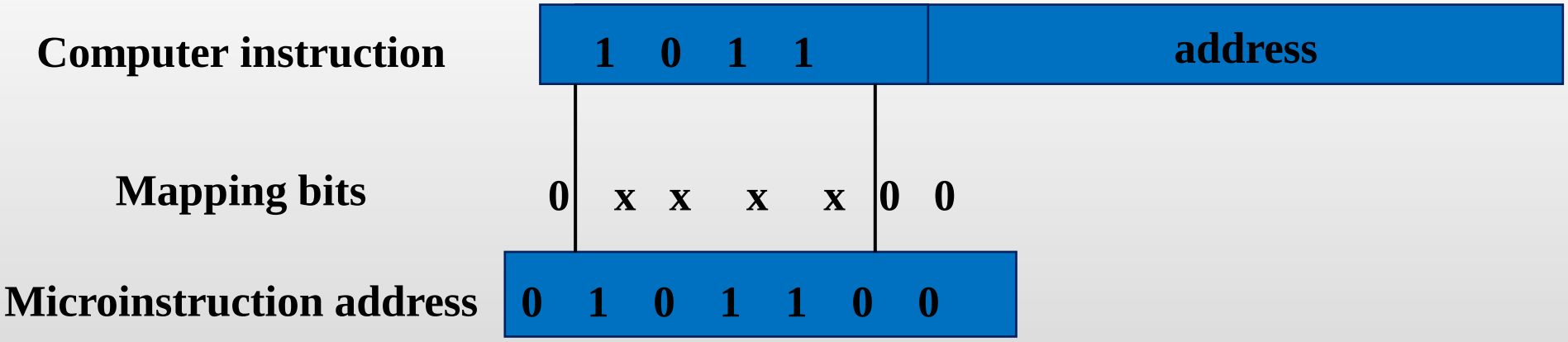
Selection of address for control memory



Mapping of Instruction

- The transformation from the instruction code bits to an address in control memory where the routine is located is referred to as a mapping process.
- A mapping procedure is a rule that transforms the instruction code into a control memory address.

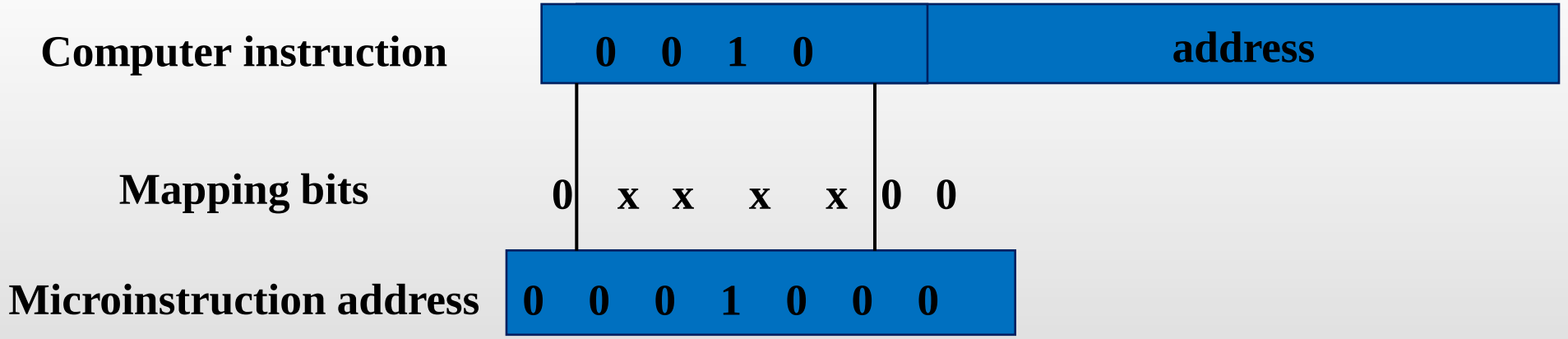
Mapping from instruction code to microinstruction address



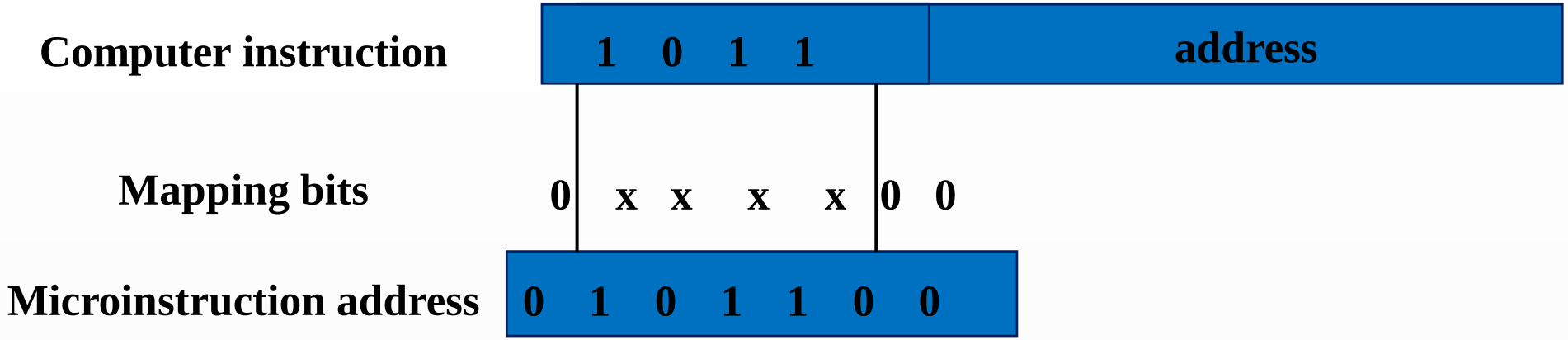
***Example:** Using the mapping procedure, write the decimal value of microoperation address of the following operation codes for 128 words control memory.

(a) 0010
(b) 1011
(c) 1111

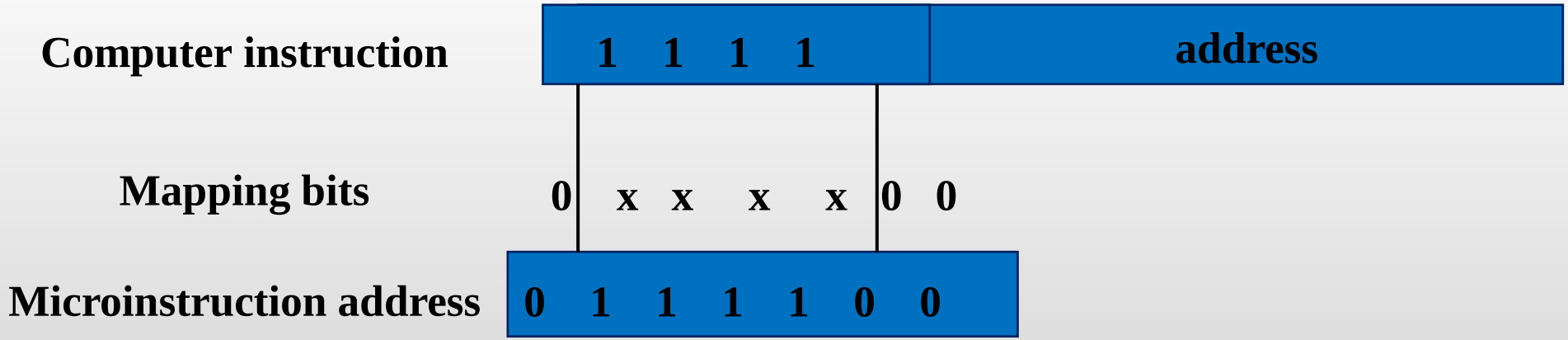
Sol: 128 words = 2^7



(a) 0001000 = 8



(b) 0101100 = 44

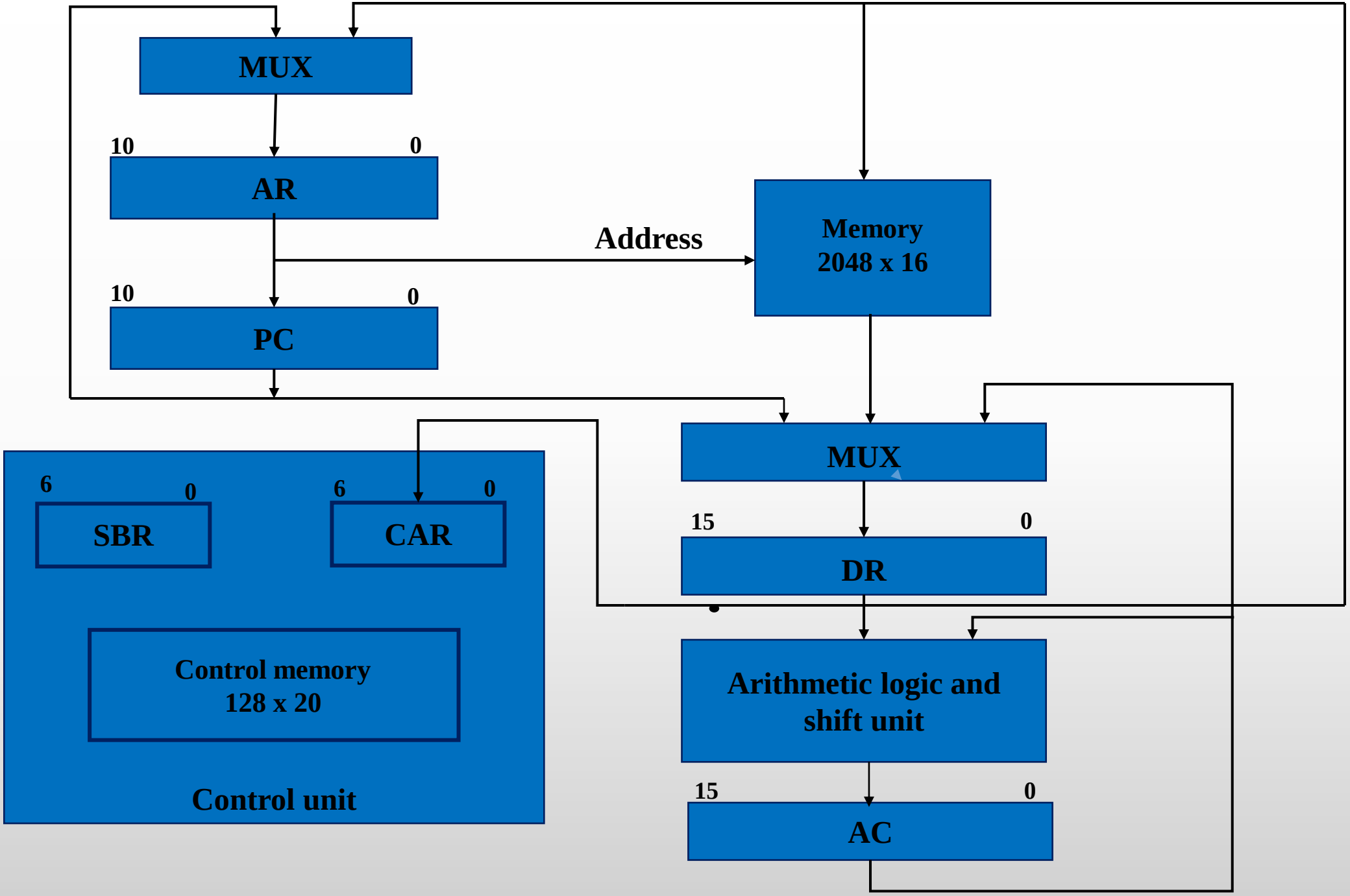


(c) 0111100 = 60

Microprogram Example

Computer Configuration

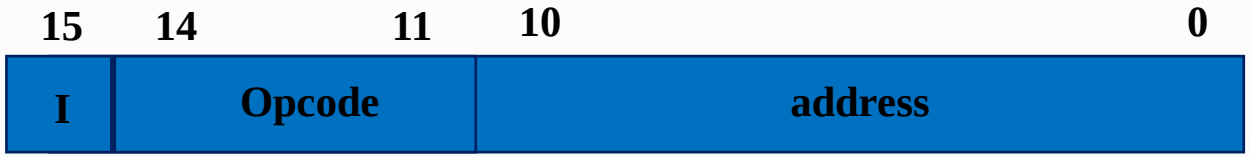
- The block diagram of the computer consists of two memory units: main memory and control memory.
- A main memory for storing instructions and data, and a control memory for storing the microprogram.
- Four registers are associated with the processor unit and two with the control unit.
- The processor registers are program counter **PC**, address register **AR**, data register **DR**, and accumulator register **AC**.
- The control unit has a control address register **CAR** and a subroutine register **SBR**.
- The control memory and its register are organized as a microprogrammed control unit.


MUX
10
0
AR
Address
Memory
2048 x 16
10
0
PC
MUX
6
0
SBR
6
0
CAR
15
0
DR
Control memory
128 x 20
Arithmetic logic and
shift unit
Control unit
15
0
AC

Microinstruction Format

- The computer instruction code format is described in the following.

Instruction format



Four Computer Instruction

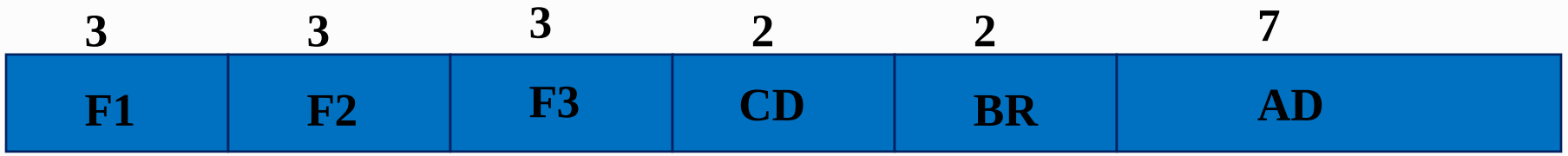
Symbol	Opcode	Description
ADD	0000	AC ← AC + M[EA]
BRANCH	0001	If (AC < 0) then (PC ← EA)
STORE	0010	M[EA] ← AC
EXCHANGE	0011	AC ← M[EA],M[EA] ← AC

EA is the effective address.

Microinstruction Code Format

- The microinstruction code format for the control memory is 20-bits long which is divided into four functional parts.
- The microoperations are subdivided into three fields of three bits each.
- The three fields F1, F2, and F3 specify microoperations for the computer.
- The three bits in each field are encoded to specify seven distinct microoperations.
- The CD field selects status bit conditions.
- The CD field consists o two bits which are encoded to specify four status bit conditions.
- The BR field specifies the type of branch to be used.
- The BR field consists of two bits which is used in conjunction with the address field AD to choose the address o the next microinstruction.
- The AD field contains a branch address.
- The address field is seven bits wide, since the control memory has $128 = 2^7$ words.

20-bits Microinstruction Code Format



F1, **F2**, and **F3**: Microoperations fields
CD: Condition for branching
BR: Branch field
AD: Address field

Symbols and Binary Code for Microinstruction Fields (F1)

F1	Microoperation	Symbol
000	None	NOP
001	AC ← AC + DR	ADD
010	AC ← 0	CLRAC
011	AC ← AC +1	INCAC
100	AC ← DR	DRTAC
101	AR ← DR (0-10)	DRTAR
110	AR ← PC	PCTAR
111	M[AR] ← DR	WRITE

Symbols and Binary Code for Microinstruction Fields (F2)

F2	Microoperation	Symbol
000	None	NOP
001	AC ← AC - DR	SUB
010	AC ← AC ∨ DR	OR
011	AC ← AC ∧ DR	AND
100	DR ← M[AR]	READ
101	DR ← AC	ACTDR
110	DR ← DR + 1	INCDR
111	DR(0-10) ← PC	PCTDR

Symbols and Binary Code for Microinstruction Fields (F3)

F3	Microoperation	Symbol
000	None	NOP
001	AC ← AC ⊕ DR	XOR
010	AC ← $\overline{AC}$	COM
011	AC ← shl AC	SHL
100	AC ← shr AC	SHR
101	PC ← PC + 1	INCPC
110	PC ← AR	ARTPC
111	Reserved	

Example: Write the 9-bit microoperation field for the following microoperations.

(a) AC ← AC + 1, DR ← DR + 1

(b) PC ← PC + 1, DR ← M[AR]

(c) DR ← AC, AC ← DR

(d) AC ← DR, M[AR] ← DR

Sol:

(a) **AC ← AC + 1, DR ← DR + 1**

F1	**F2**	**F3**
011	110	000
INCAC	INCDR	NOP

(b) **PC ← PC + 1, DR ← M[AR]**

F1	**F2**	**F3**
000	100	101
NOP	READ	INCPC

(c) **DR ← AC, AC ← DR**

F1	**F2**	**F3**
100	101	000
DRTAC	ACTDR	NOP

(d) This is impossible because both microoperations use 3-bit microoperation codes of **F1** (100 for **DRTAC** and 111 for **WRITE**).

Symbols and Binary Code for Microinstruction Fields (CD)

CD	Condition	Symbol	Comments
00	Always = 1	U	Unconditional branch
01	DR(15)	I	Indirect address bit
10	AC(15)	S	Sign bit of AC
11	AC = 0	Z	Zero value in AC

Symbols and Binary Code for Microinstruction Fields (BR)

BR	Symbol	Function
00	JMP	CAR ← AD if condition = 1
		CAR ← CAR + 1 if condition = 0
01	CALL	CAR ← AD, SBR ← CAR + 1 if condition = 1
		CAR ← CAR +1 if condition =0
10	RET	CAR ← SBR (Return from subroutine)
11	MAP	CAR(2-5) ← DR(11-14),CAR(0,1,6)←0

Symbolic Microinstructions

- Each line of the assembly language microprogram defines a symbolic microinstruction.
- Each symbolic microinstruction is divided into five fields: label, microoperations, CD, BR, and AD.
- The fields specify the following information.
  1. The label field may be empty or it may specify a symbolic address. A label is terminated with a colon (:).
  2. The microoperations field consists of one, two, or three symbols, separated by commas.
  3. The CD field has one of the letters U, I, S, or Z.
  4. The BR field contains one of the four symbols (JMP,CALL, RET,MAP).
  5. The AD field specifies a value for the address field of the microinstruction.

Symbolic Microprogram (Partial)

Label	Microoperations	CD	BR	AD
	ORG 0			
ADD:	NOP	I	CALL	INDRCT
	READ	U	JMP	NEXT
	ADD	U	JMP	FETCH
	ORG4			
BRANCH:	NOP	S	JMP	OVER
	NOP	U	JMP	FETCH
OVER:	NOP	I	CALL	INDRCT
	ARTPC	U	JMP	FETCH

Label	Microoperations	CD	BR	AD
	ORG 8			
STORE:	NOP	I	CALL	INDRCT
	ACTDR	U	JMP	NEXT
	WRITE	U	JMP	FETCH
	ORG12			
EXCHANGE:	NOP	I	CALL	TNDRCT
	READ	U	JMP	NEXT
	ACTDR, DRTAC	U	JMP	NEXT
	WRITE	U	JMP	FETCH

Label	Microoperations	CD	BR	AD
	ORG 64			
FETCH:	PCTAR	U	JMP	NEXT
	READ, INCPC	U	JMP	NEXT
	DRTAR	U	MAP	
INDRCT:	READ	U	JMP	NEXT
	DRTAR	U	RET	

Binary Microprogram (Partial)

Micro Routine	Address Decimal	Address Binary	Binary Microinstruction F1	F2	F3	CD	BR	AD
ADD	0	0000000	000	000	000	01	01	1000011
	1	0000001	000	100	000	00	00	0000010
	2	0000010	001	000	000	00	00	1000000
	3	0000011	000	000	000	00	00	1000000
BRANCH	4	0000100	000	000	000	10	00	0000110
	5	0000101	000	000	000	00	00	1000000
	6	0000110	000	000	000	01	01	1000011
	7	0000111	000	000	110	00	00	1000000
STORE	8	0001000	000	000	000	01	01	1000011
	9	0001001	000	101	000	00	00	0001010
	10	0001010	111	000	000	00	00	1000000
	11	0001011	000	000	000	00	00	1000000

Micro Routine	Address Decimal	Address Binary	Binary Microinstruction F1	F2	F3	CD	BR	AD
EXCHANGE	12	0001100	000	000	000	01	01	1000011
	13	0001101	001	000	000	00	00	0001110
	14	0001110	100	101	000	00	00	0001111
	15	0001111	111	000	000	00	00	1000000
FETCH	64	1000000	110	000	000	00	00	1000001
	65	1000001	000	100	101	00	00	1000010
	66	1000010	101	000	000	00	11	0000000
INDRCT	67	1000011	000	100	000	00	00	1000100
	68	1000100	101	000	000	00	10	0000000

FETCH routine

- The following are microoperations described in register transfer statements for the fetch routine.

  AR ← PC

  DR ← M[AR], PC ← PC+1

  AR ← DR(0-10), CAR(2-5) ← DR(11-14), CAR(0,1,6) ←0
- The first memory address of miroinstructions for these microoperations are 64.

Symbolic Microprogram

Label	Microoperations	CD	BR	AD
	ORG 64			
FETCH:	PCTAR	U	JMP	NEXT
	READ, INCPC	U	JMP	NEXT
	DRTAR	U	MAP	

Binary Microprogram for FETCH routine

Micro Routine	Address Decimal	Address Binary	Binary Microinstruction F1	F2	F3	CD	BR	AD
FETCH	64	1000000	110	000	000	00	00	1000001
	65	1000001	000	100	101	00	00	1000010
	66	1000010	101	000	000	00	11	0000000

ADD Routine

- The following are microoperatios described in register transfer statements for the ADD routine.

  DR ← M[AR]

  AC ← AC + DR
- The first memory address of microinstructions for this routine is 8.
- And, the first address of microinstructions for INDRCT subroutine is 67.

Symbolic Microprogram

Label	Microoperations	CD	BR	AD
	ORG 0			
ADD:	NOP	I	CALL	INDRCT
	READ	U	JMP	NEXT
	ADD	U	JMP	FETCH

Binary Microprogram for ADD Routine

Micro Routine	Address Decimal	Address Binary	Binary Microinstruction F1	F2	F3	CD	BR	AD
ADD	0	0000000	000	000	000	01	01	1000011
	1	0000001	000	100	000	00	00	0000010
	2	0000010	001	000	000	00	00	1000000

STORE Routine

- The following are microoperatios described in register transfer statements for the STORE routine.

  DR ← AC

  M[AR] ← DR
- The first memory address of microinstructions for this routine is 8.
- And, the first address of microinstructions for INDRCT subroutine is 67.

Symbolic Microprogram

Label	Microoperations	CD	BR	AD
	ORG 8			
STORE:	NOP	I	CALL	INDRCT
	ACTDR	U	JMP	NEXT
	WRITE	U	JMP	FETCH

Binary Microprogram for STORE Routine

Micro	Address		Binary Microinstruction					
Routine	Decimal	Binary	F1	F2	F3	CD	BR	AD
STORE	8	0001000	000	000	000	01	01	1000011
	9	0001001	000	101	000	00	00	0001010
	10	0001010	111	000	000	00	00	1000000

Label INDRCT:

- The following are microoperations described in register transfer statements for the label, INDRCT:

  DR ← M[AR]

  AR ← DR
- The first address of microinstructions for INDRCT subroutine is 67.

Symbolic Microprogram for label INDRCT:

Label	Microoperations	CD	BR	AD
INDRCT:	READ	U	JMP	NEXT
	DRTAR	U	RET	

Binary Microprogram for Label INDRCT:

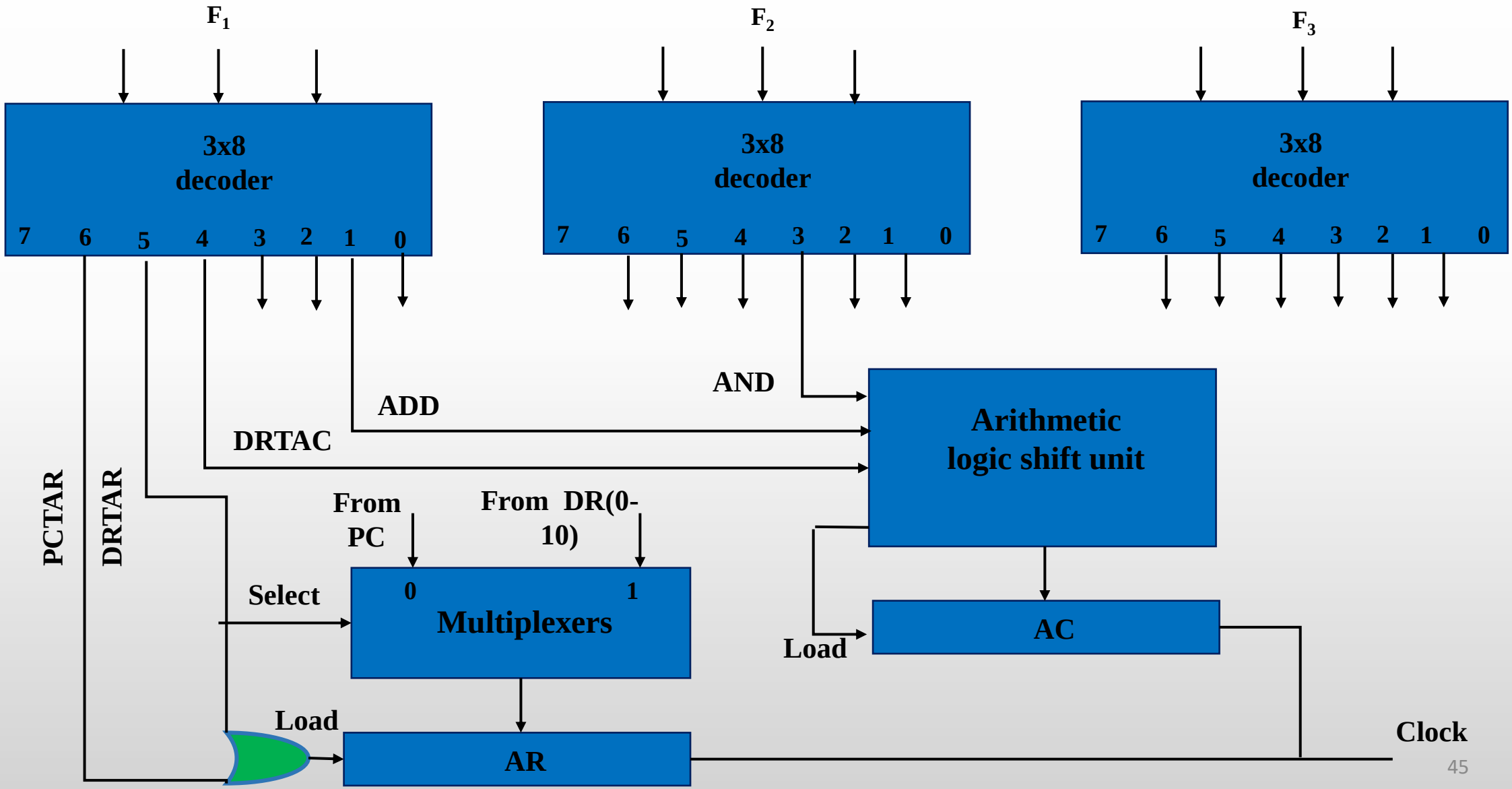
Micro Routine	Address		Binary Microinstruction					
	Decimal	Binary	F1	F2	F3	CD	BR	AD
INDRCT	67	1000011	000	100	000	00	00	1000100
	68	1000100	101	000	000	00	10	0000000

Design of Control Unit

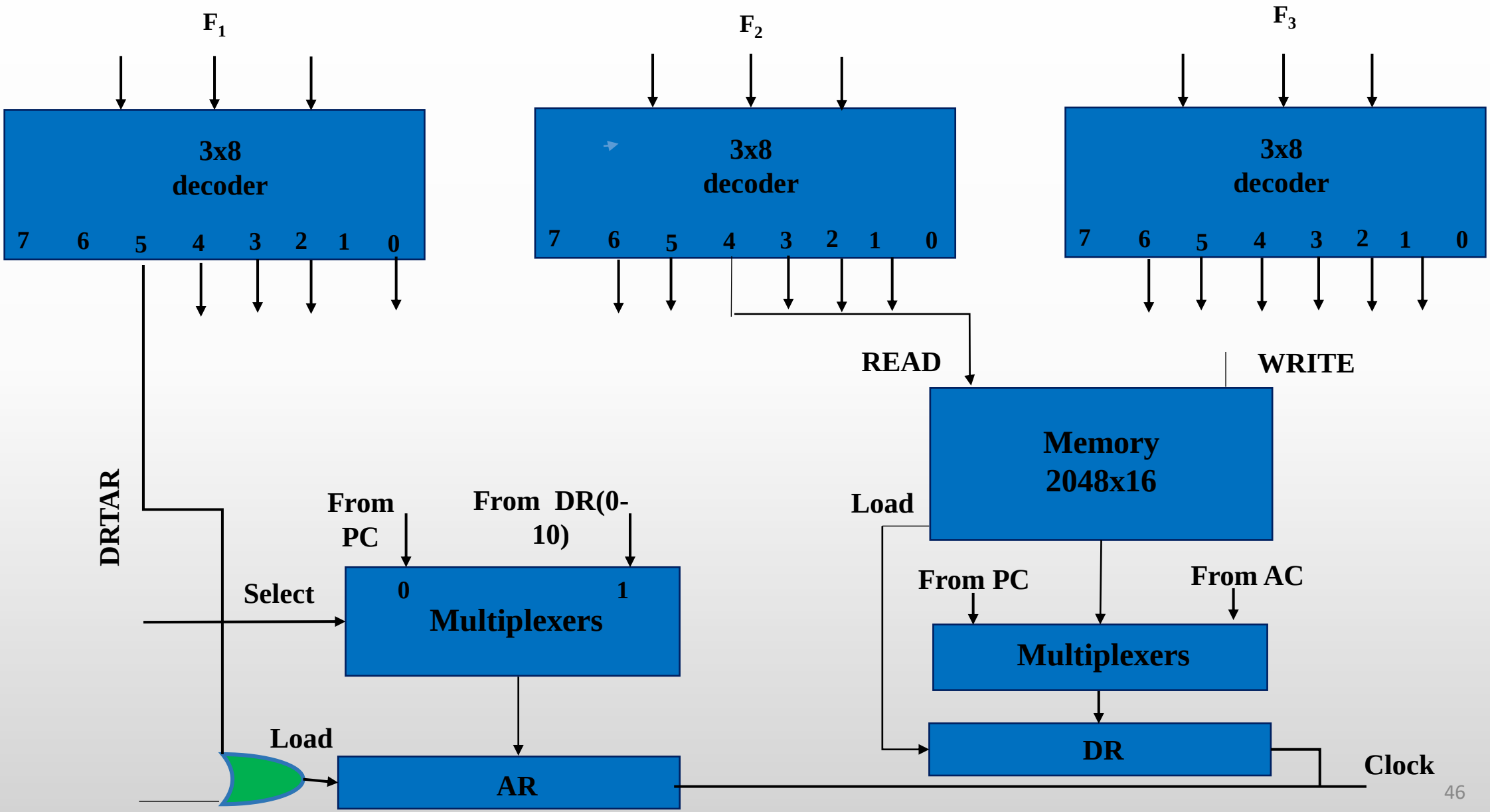
Design of Control Unit

- To design the control unit for the microinstruction code, each of three fields of the microinstruction in the output of control memory are decoded with a **3** x **8** decoder.
- Each of these outputs must be connected to the proper circuit to initiate the corresponding microoperation.
- In the following figure, three decoders, arithmetic logic unit (ALU), and required registers are used to design the control unit to provide some microoperations.
- For example, When **F1 = 101**, the next clock pulse transition transfers the content of **DR** to **AR**.
- When, **F1 = 110**, there is a transfer from **PC** to **AR**.

Design of Control Unit



Design of Control Unit for Label INDRCT:



Thank You

Lecture for Next Week

- **Central Processing Unit**
  - **General Register Organization**
  - **Stack Organization**
  - **Instruction Formats**
  - **Addressing Modes**
  - **Data Transfer and Manipulation**
  - **Program Control**
  - **Reduced Instruction Set Computer**