Computer Architecture Lec 5b

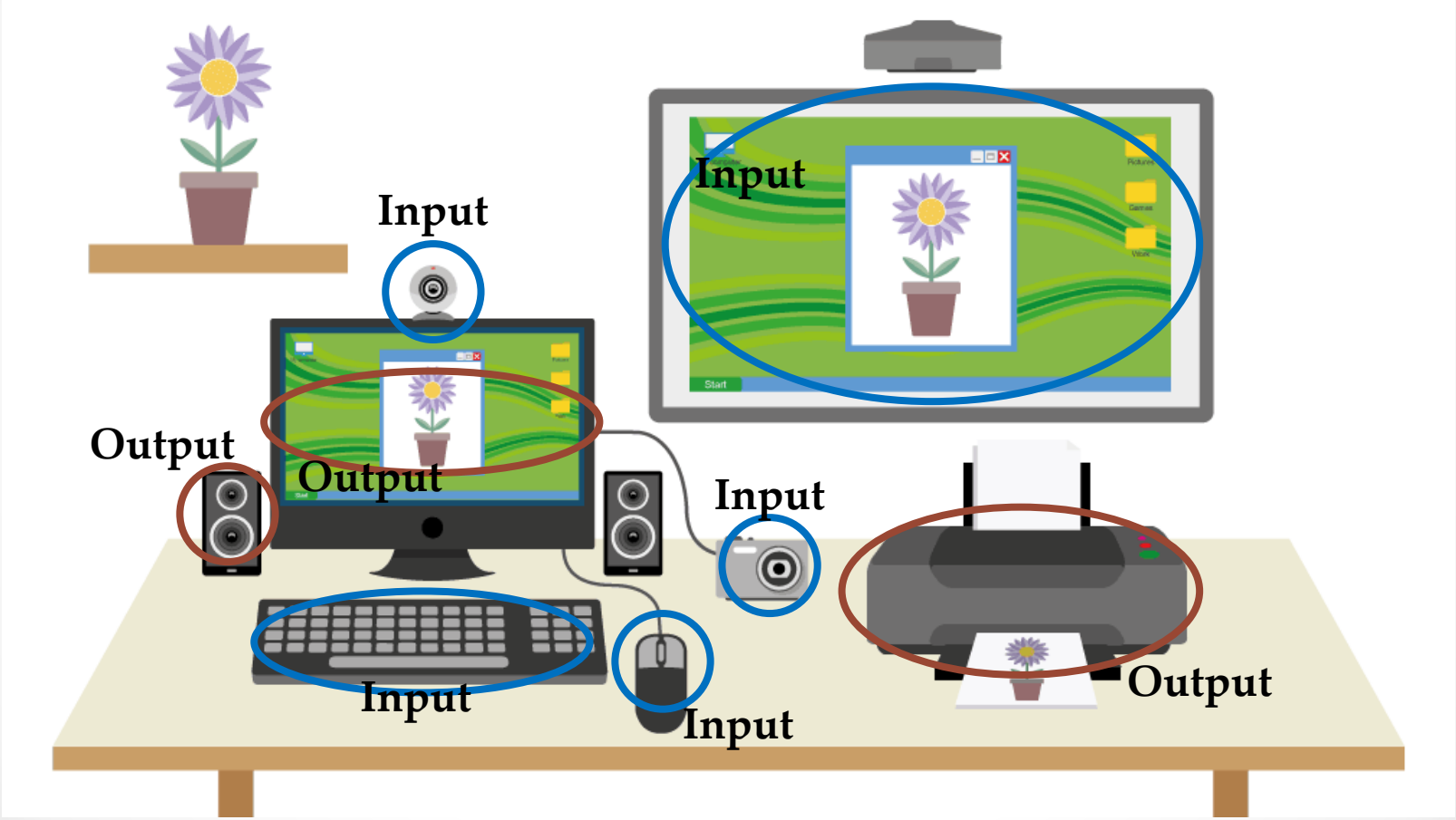
Dr. Esti Stein

(Partly taken from Dr. Alon Schclar slides)

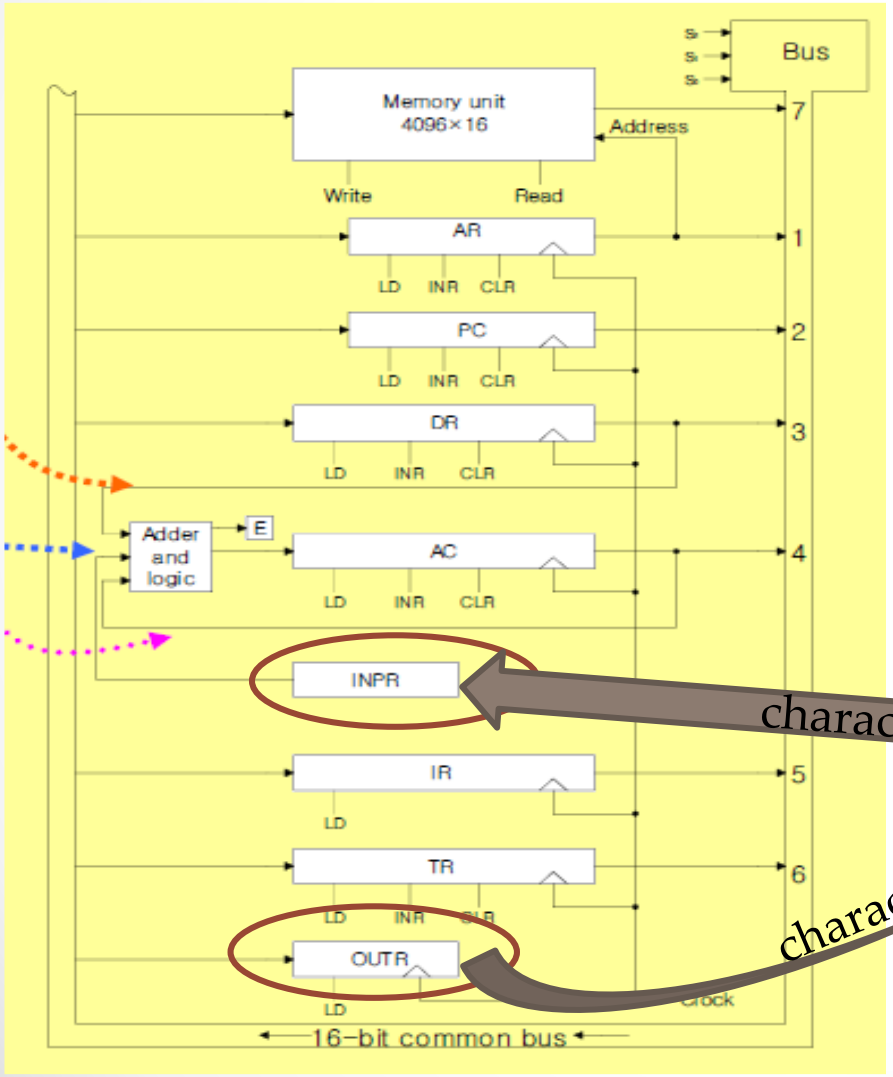
Based on slides by:
Prof. Myung-Eui Lee
Korea University of Technology & Education
Department of Information & Communication

Taken from: **M. Mano/Computer Design and Architecture 3rd Ed.**

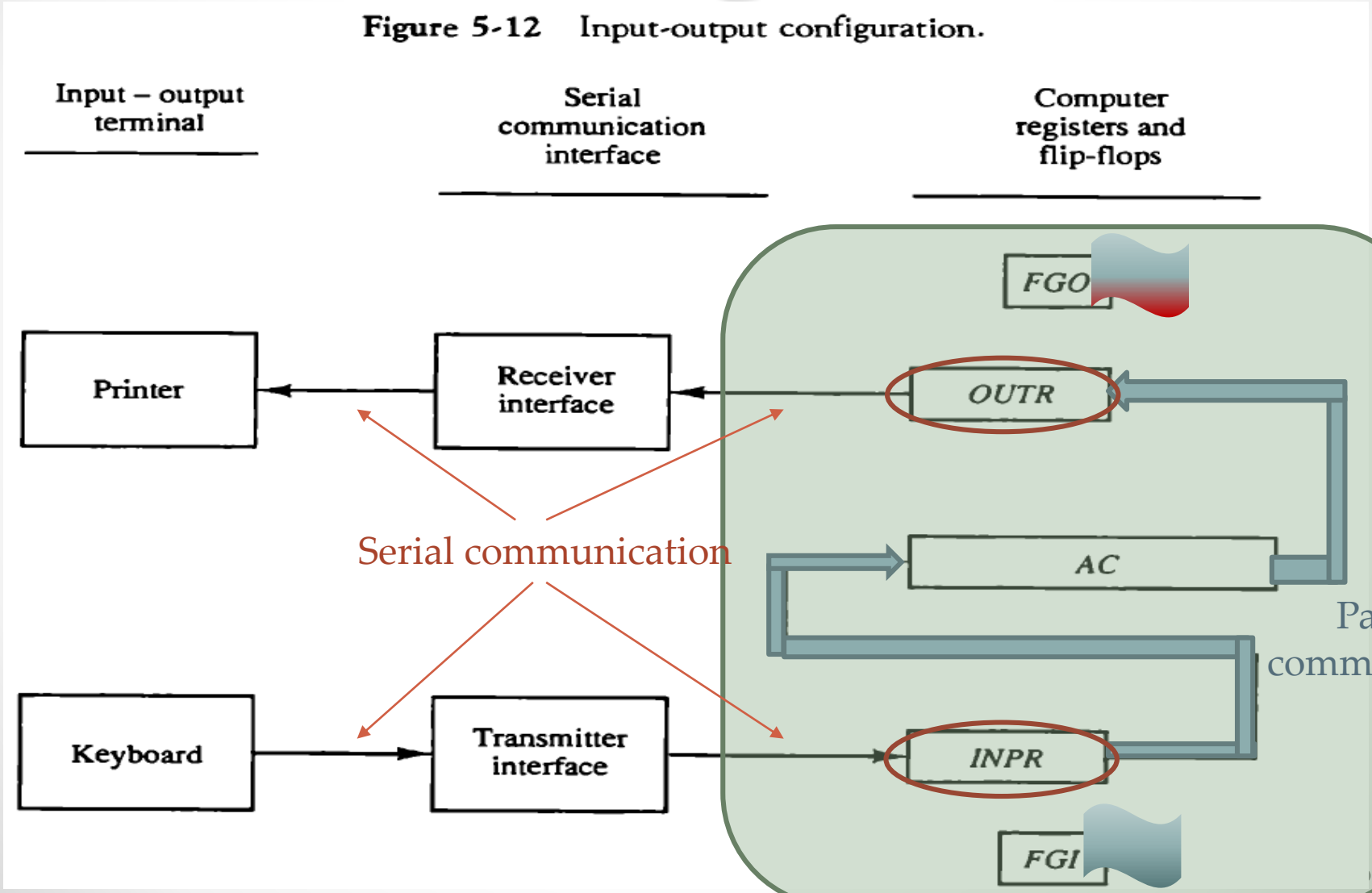
Input / Output



Our I/O Configuration

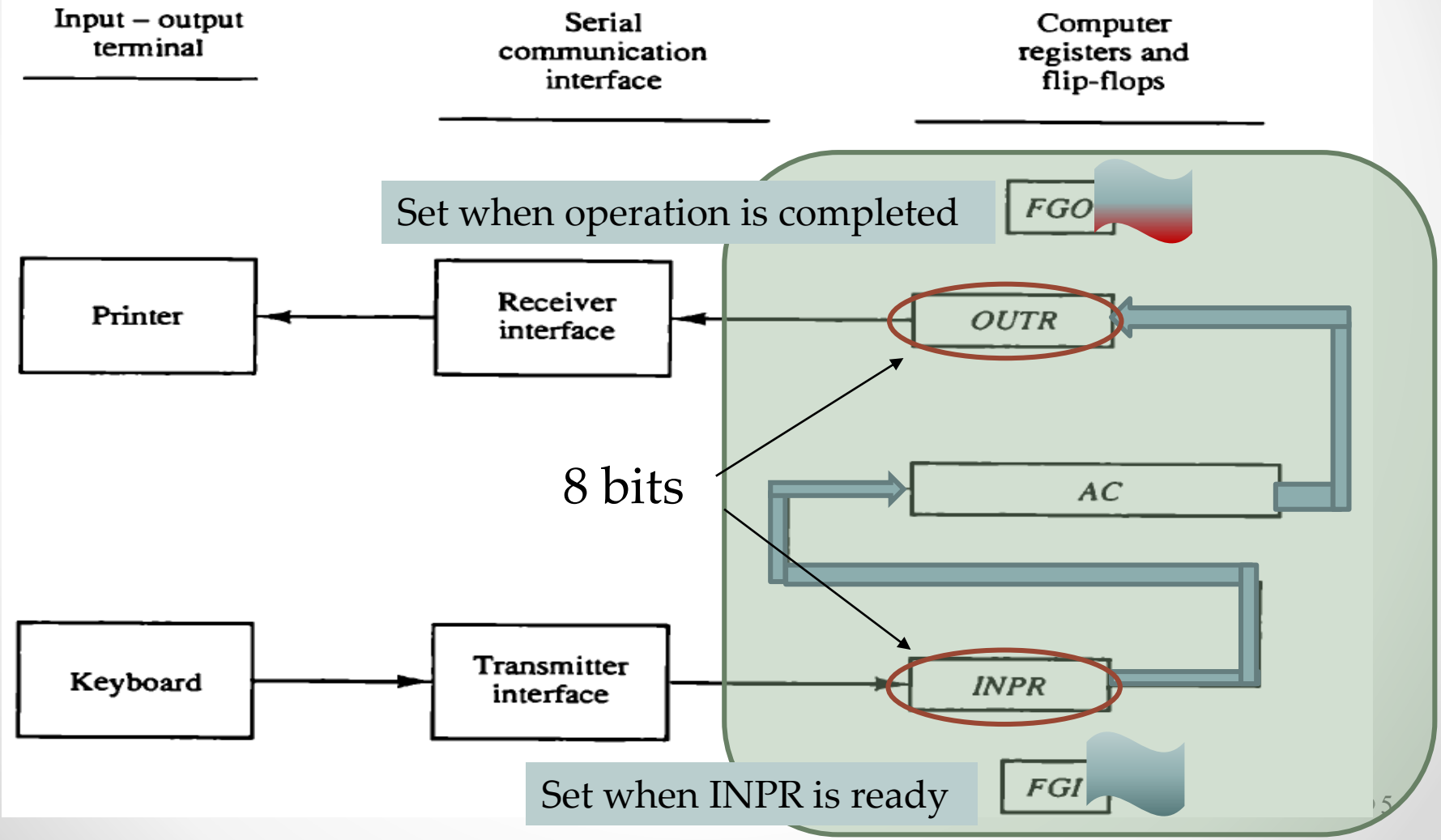


I/O Configuration

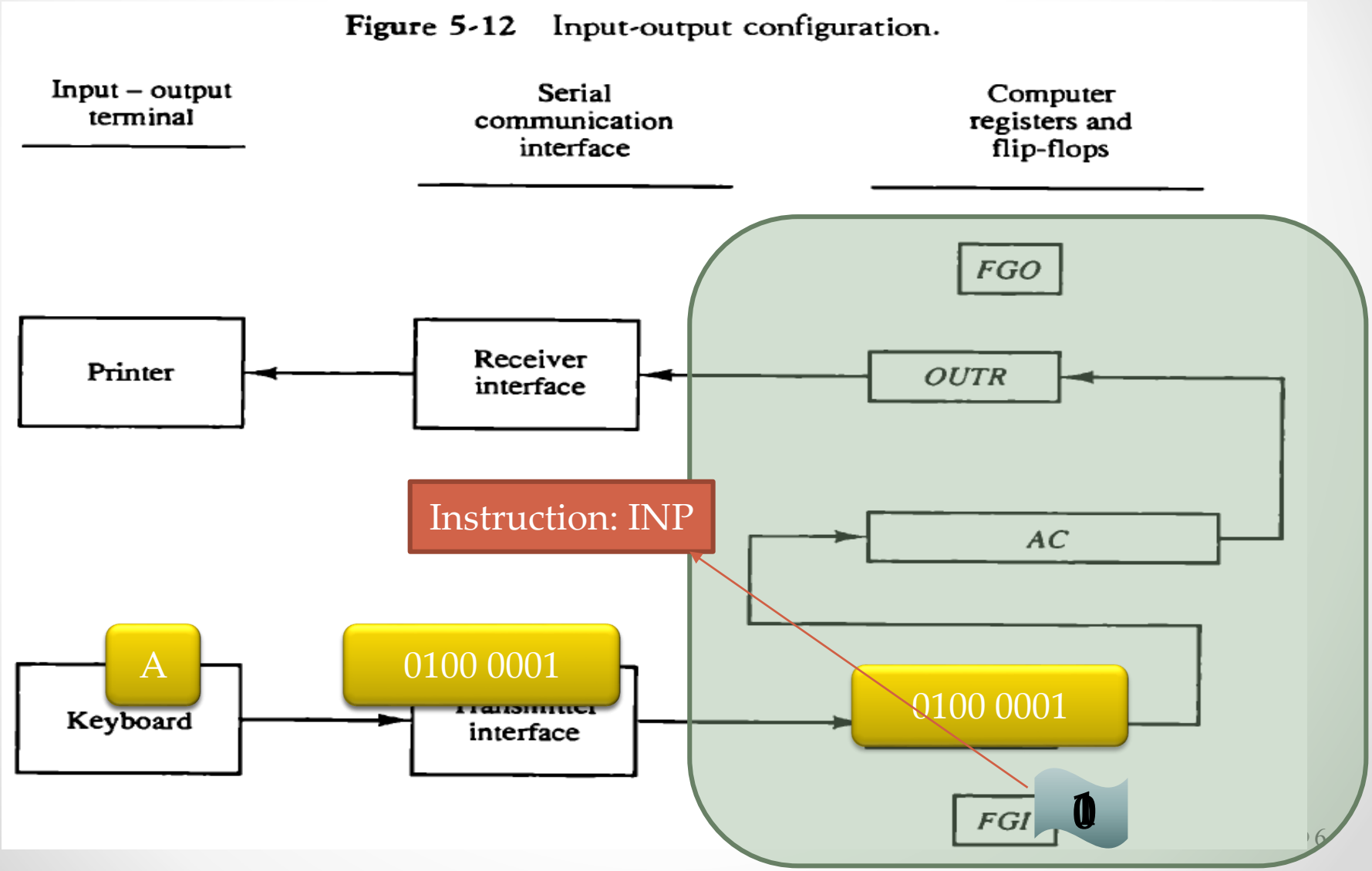


I/O Configuration

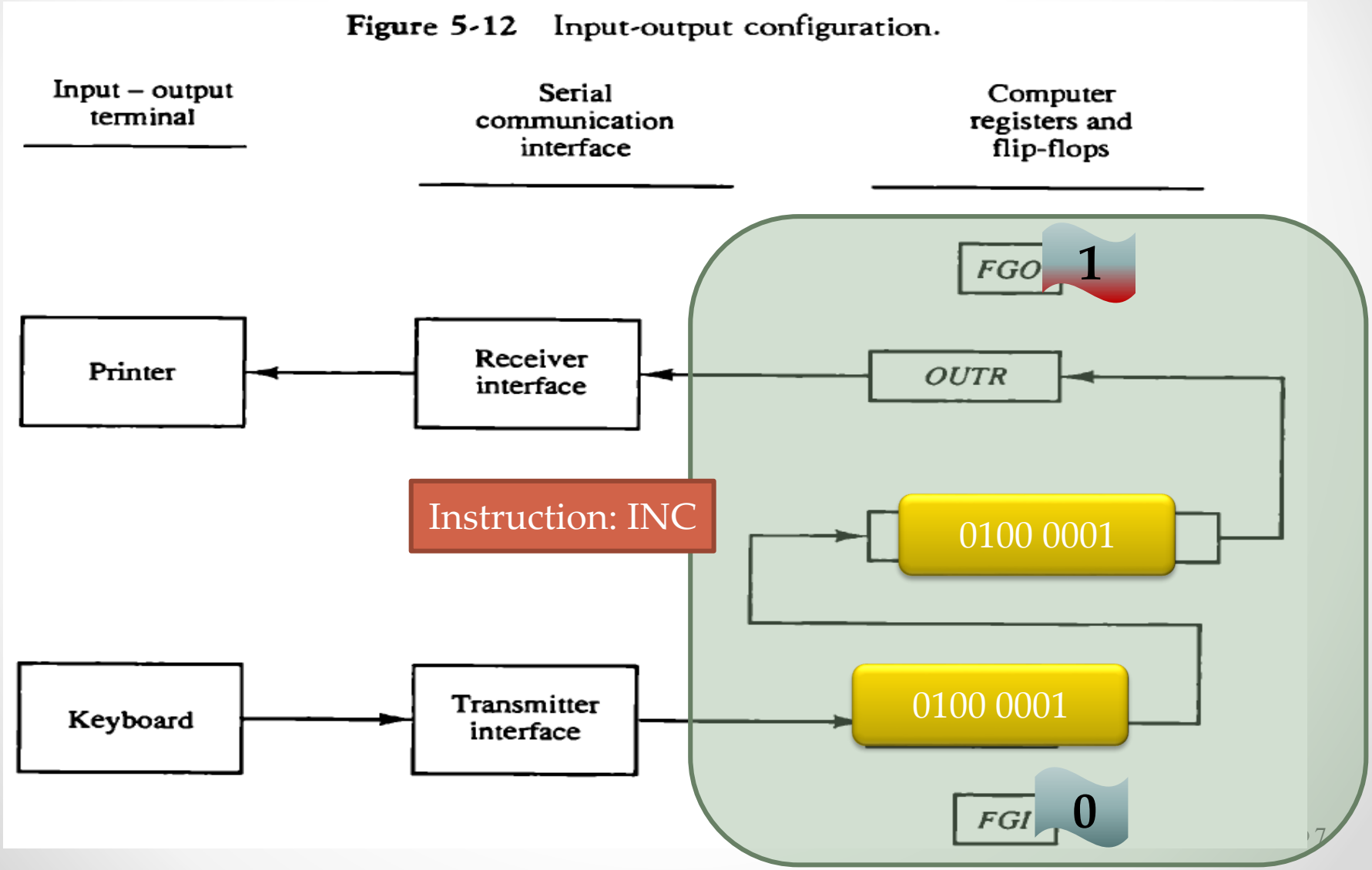
Figure 5-12 Input-output configuration.



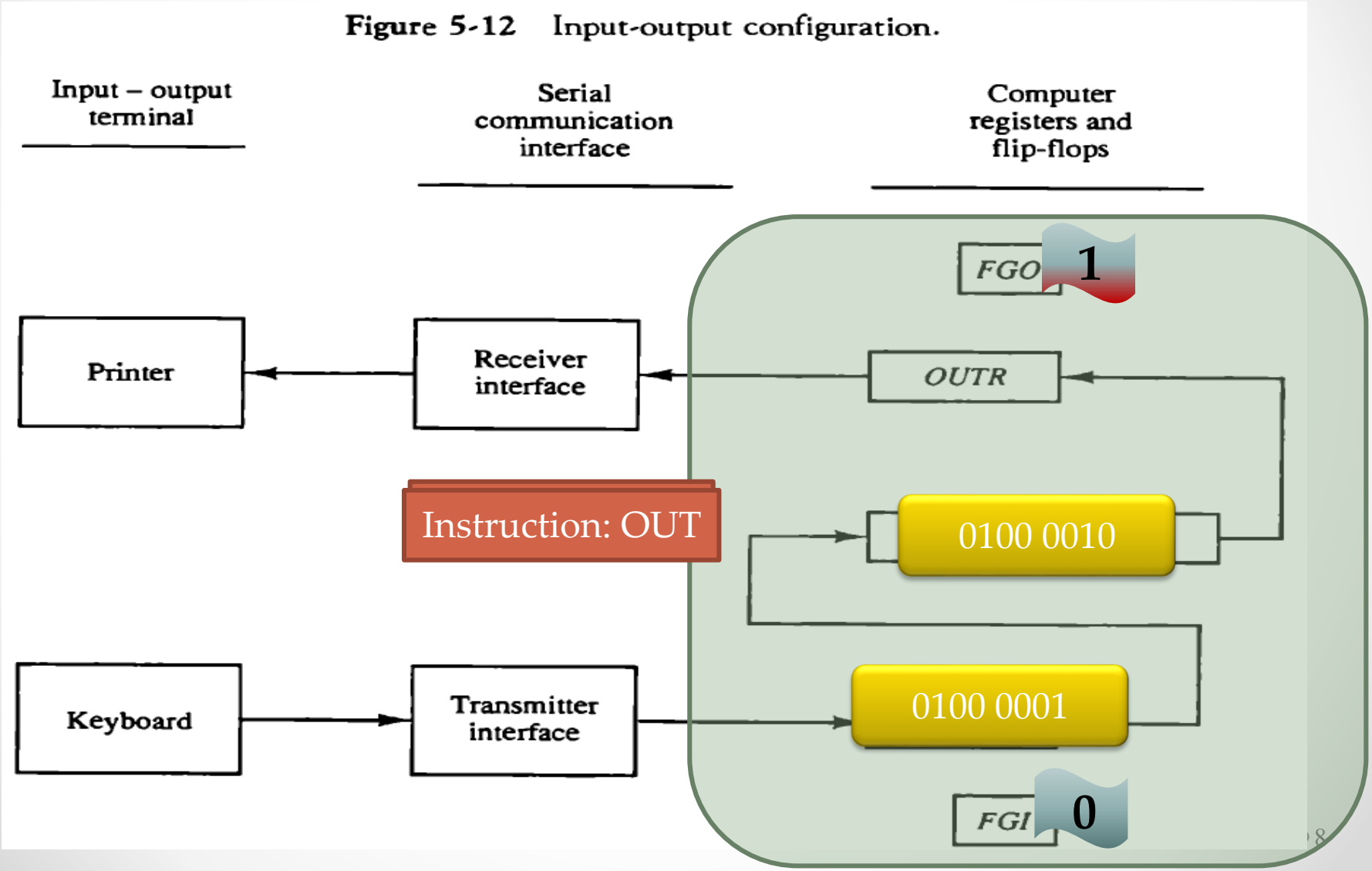
I/O Demonstration



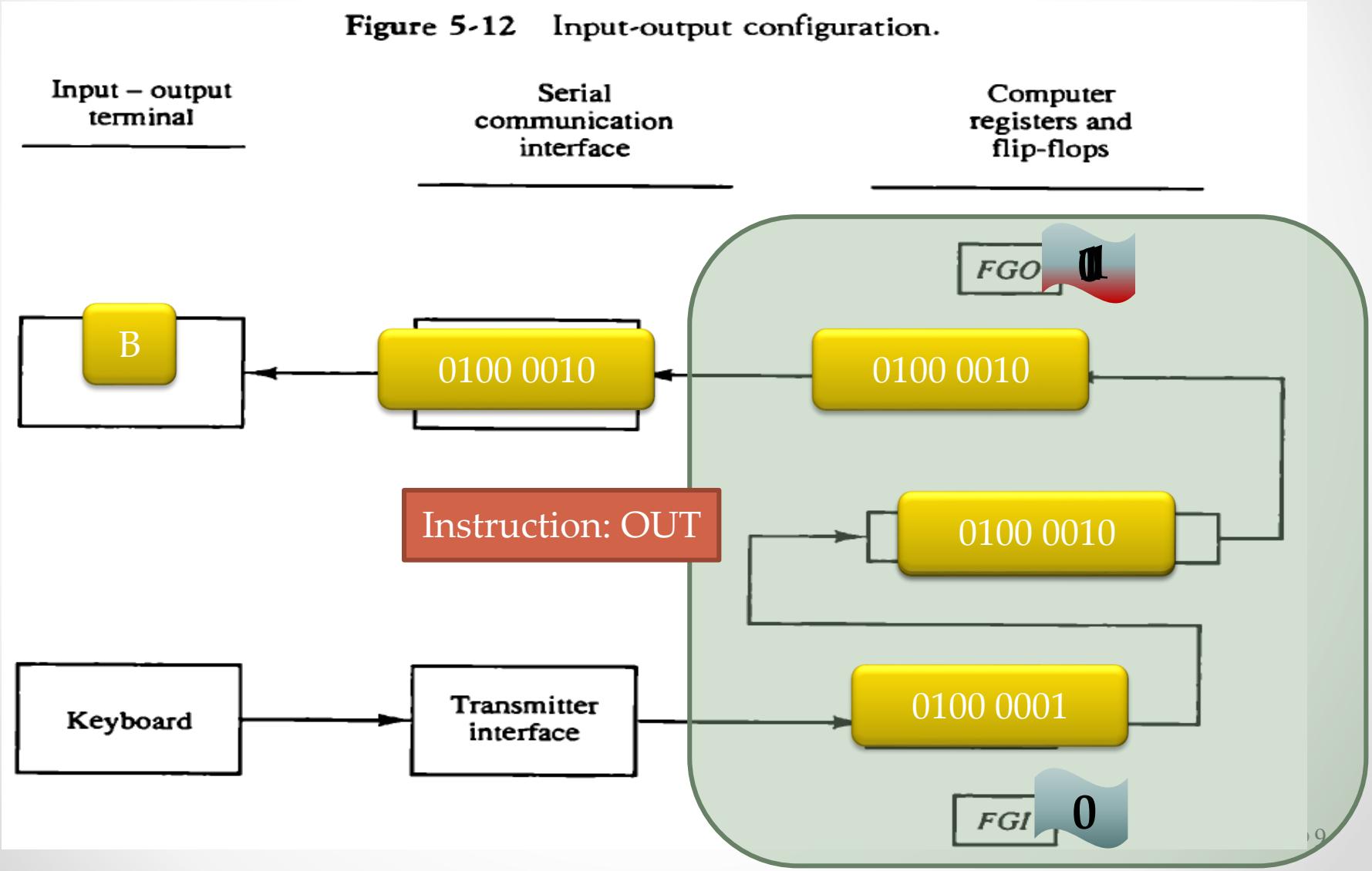
I/O Demonstration



I/O Demonstration



I/O Demonstration



The I/O Instructions

TABLE 5-5 Input-Output Instructions

$D_7IT_3 = p$ (common to all input–output instructions)
$IR(i) = B_i$ [bit in $IR(6\text{–}11)$ that specifies the instruction]

		p:	$SC \leftarrow 0$	Clear SC
F800	INP	pB_{11}:	$AC(0\text{–}7) \leftarrow INPR$, $FGI \leftarrow 0$	Input character
F400	OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{–}7)$, $FGO \leftarrow 0$	Output character
F200	SKI	pB_9:	If $(FGI = 1)$ then $(PC \leftarrow PC + 1)$	Skip on input flag
F100	SKO	pB_8:	If $(FGO = 1)$ then $(PC \leftarrow PC + 1)$	Skip on output flag
F080	ION	pB_7:	$IEN \leftarrow 1$	Interrupt enable on
F040	IOF	pB_6:	$IEN \leftarrow 0$	Interrupt enable off

Determine Instruction Type

- 5-3 Computer Instruction
  - 3 Instruction Code Formats : *F*
    - Memory-reference instruction
      - » Opcode = 000 ~ 110
        - I=0 : 0xxx ~ 6xxx, I=1: 8xx

I=0 : Direct,
I=1 : Indirect

15	14	12	11	0
I	Opcode		Address	

I

- Register-reference instruction
  - » 7xxx (7800 ~ 7001) : CLA, C

15	14	12	11	0
0	1	1	1	Register Operation

- Input-Output instruction
  - » Fxxx(F800 ~ F040) : INP, O

15	14	12	11	0
1	1	1	1	I/O Operation

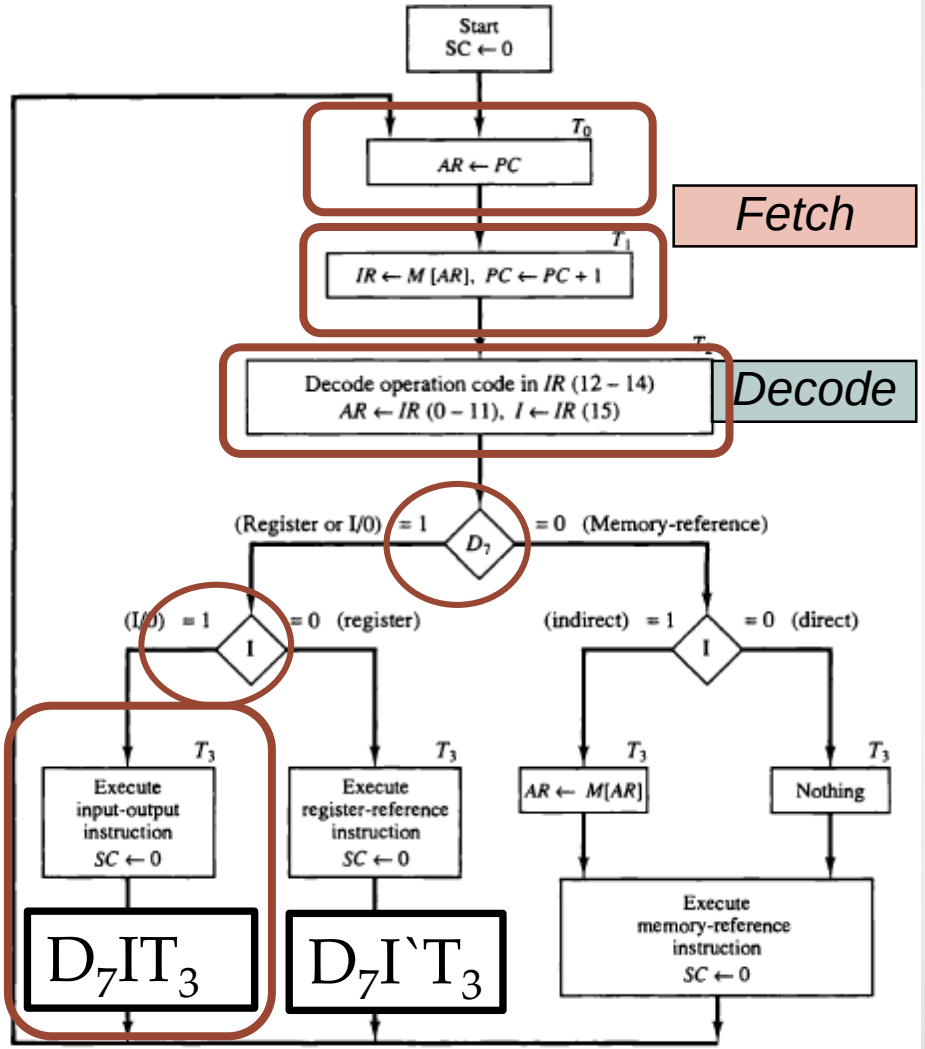


Figure 5-9 Flowchart for instruction cycle (initial configuration).

The Interrupt

BUSY WAIT:

The computer is wasting time while checking the flag instead of doing some other useful processing task.

ALTERNATIVE:

The computer lets the external device inform when it is ready for the transfer.

Meanwhile, the computer can be busy with other tasks.

This mechanism is called: INTERRUPT

The Interrupt

- While **running** a program, the computer **does not check** the flags.
- When an I/O flag is set, the computer
  - Is **informed**
  - Is **momentarily interrupted** from proceeding with the current program
  - **Deviates momentarily** from what it is doing to take care of the I/O
  - **Resumes** program execution

Special procedure

Controlling The Interrupt Facility

- ***I*nterrupt *EN*able** flip-flop (IEN)
  - set to 1 via ***ION* instruction**
  - cleared to 0 by the ***IOF* instruction**
- When ***IEN*** is clear (**=0**)
  - the flags **cannot interrupt** the computer.
- When ***IEN*** is set (**=1**)
  - The computer **can be interrupted**.
- **ION, IOF allow** the programmer to **decide whether or not to** use the **interrupt** facility.

Flowchart for interrupt cycle

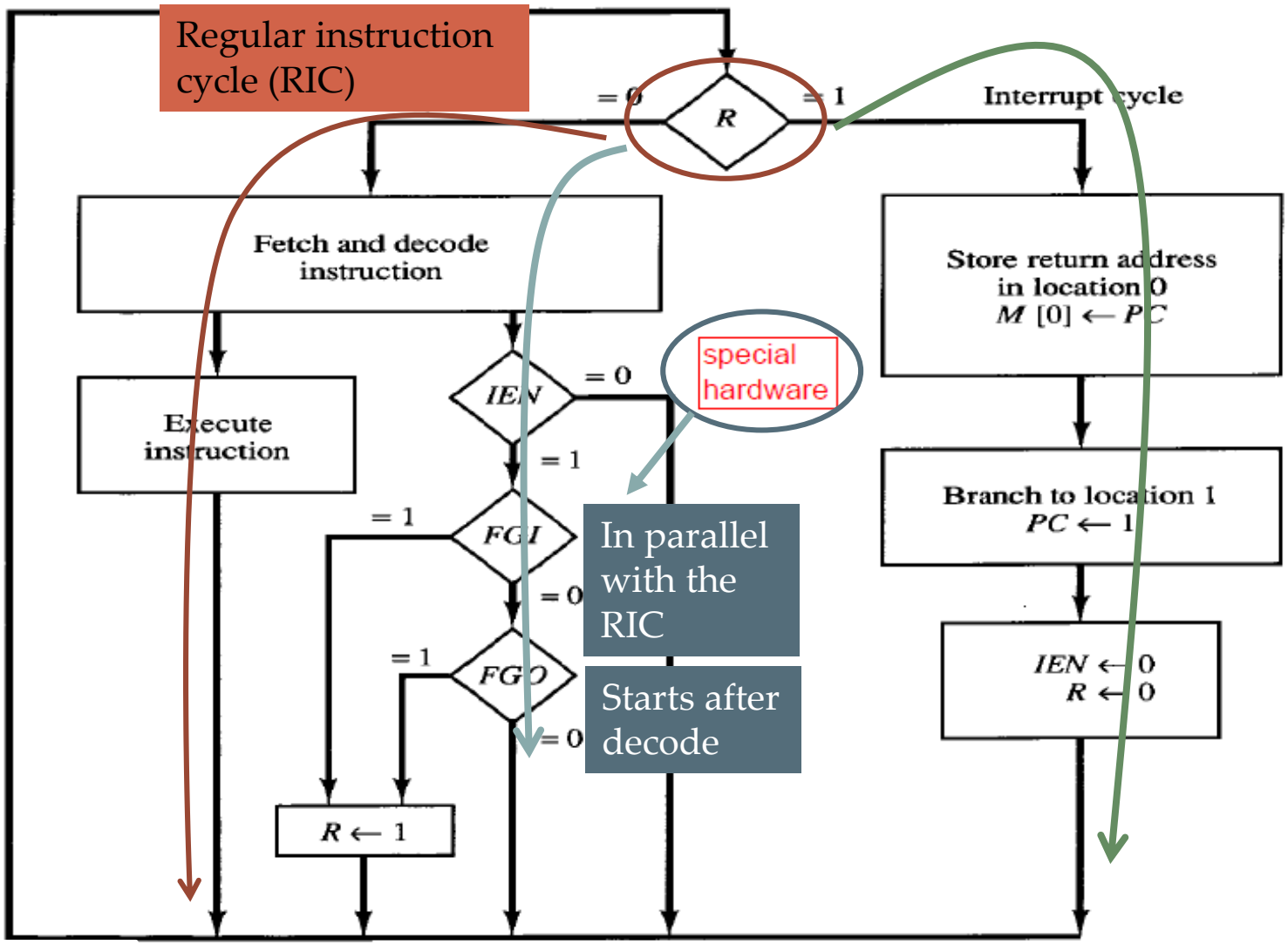
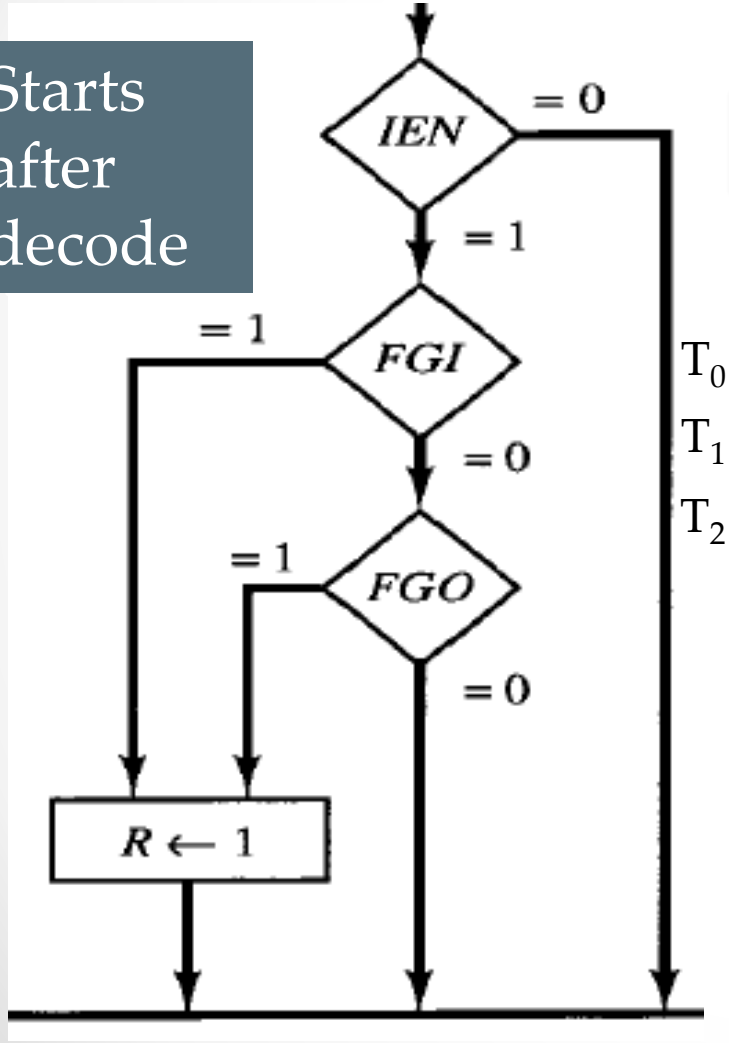


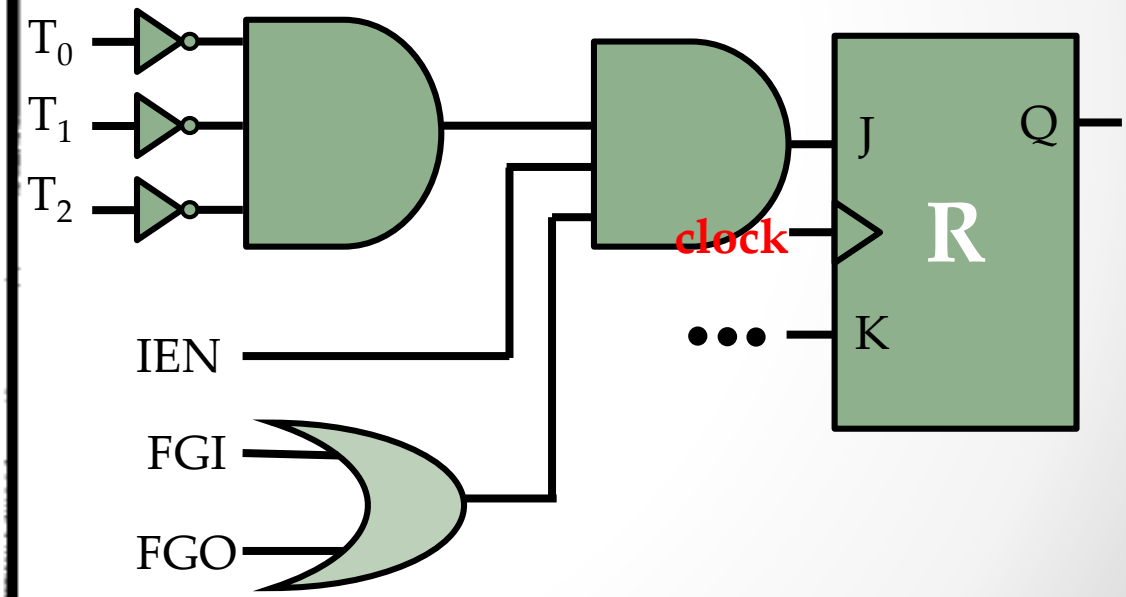
Figure 5-13 Flowchart for interrupt cycle.

Going to Interrupt Cycle

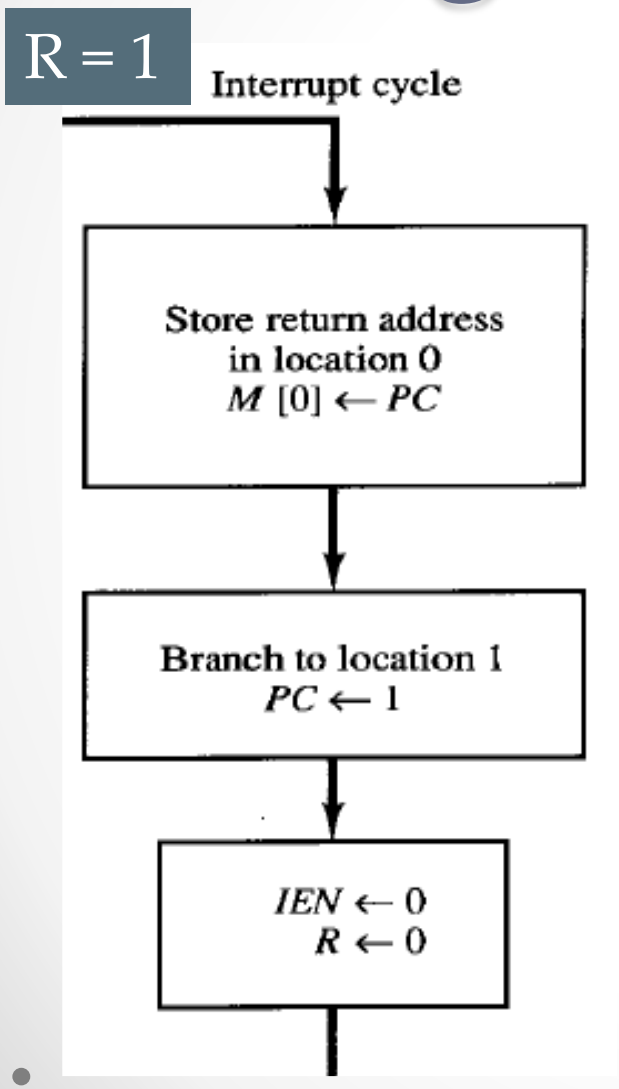
Starts after decode

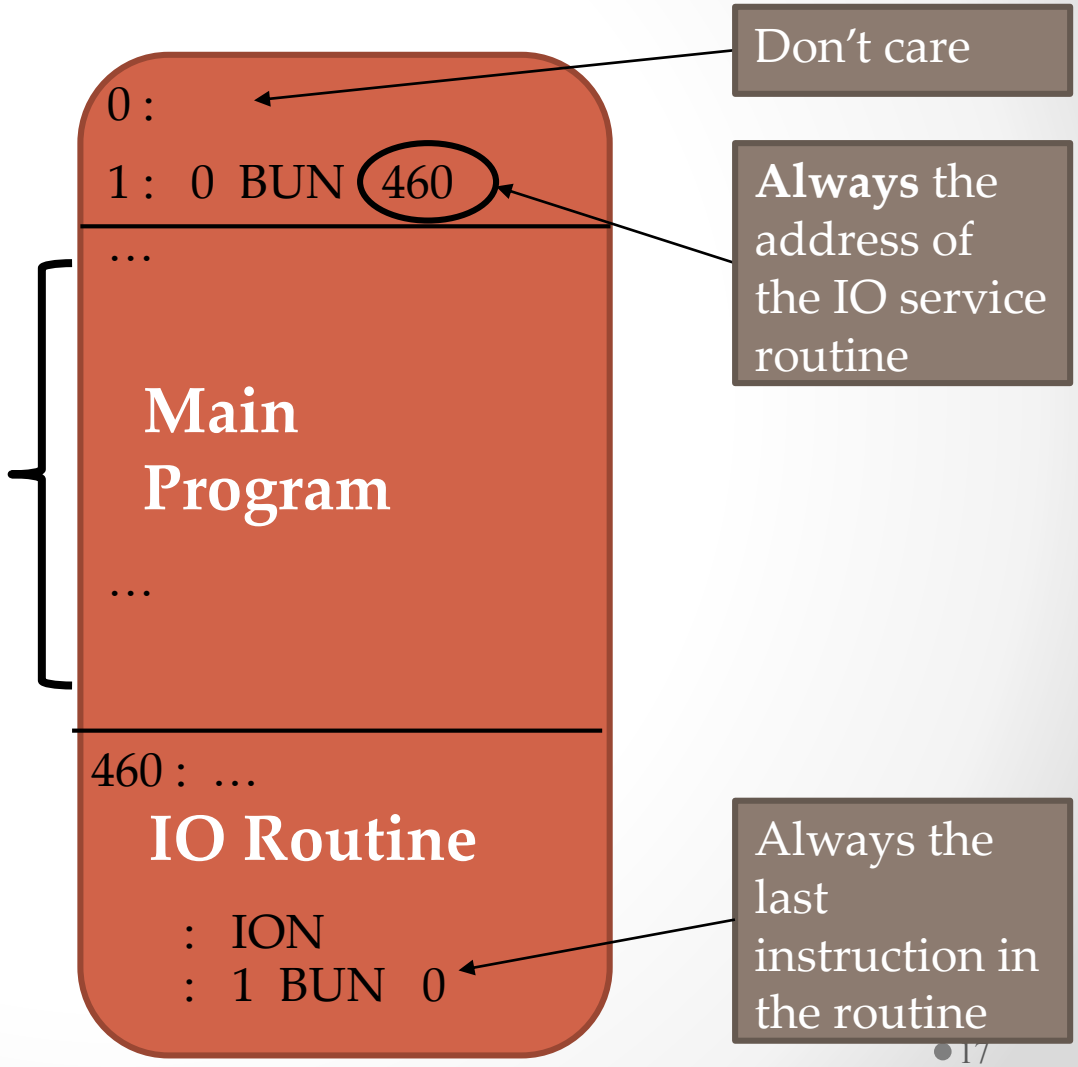


$T_0'T_1'T_2'(IEN)(FGI+FGO) : R \leftarrow 1$



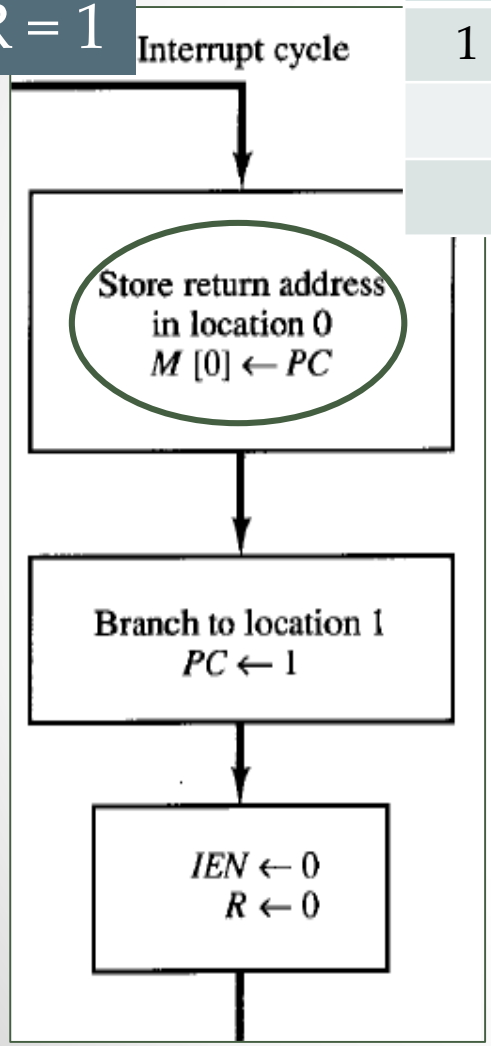
Interrupt Routine Arrangement in Memory





Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1		256		1	T_0
					T_1
					T_2

$R'T_0$: AR ← PC

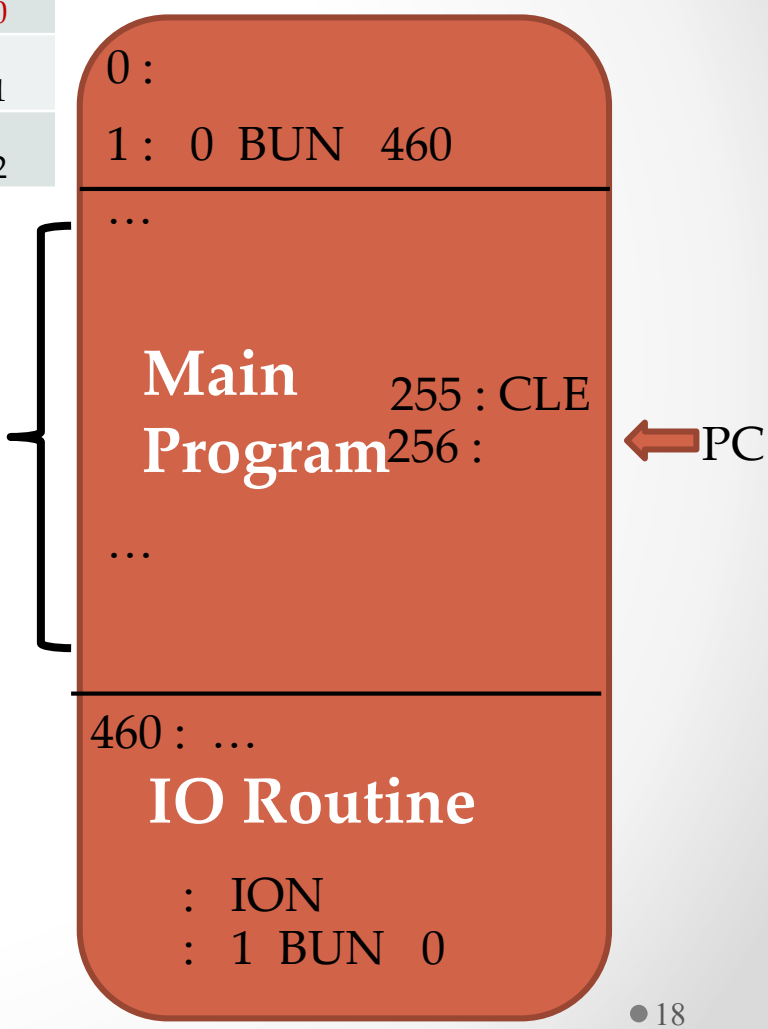
RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

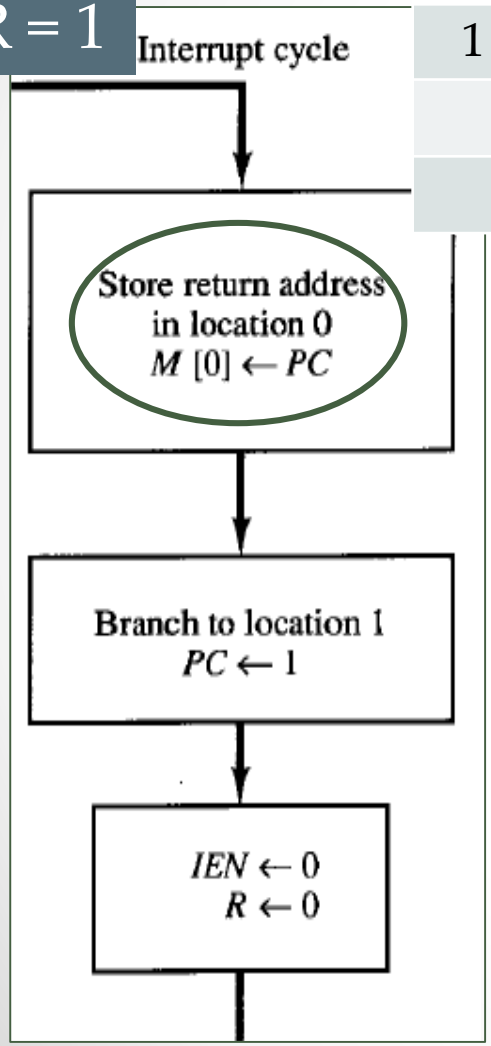
$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0



Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1	256	256	0	1	T_0
					T_1
					T_2

$R'T_0$: AR ← PC

RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0

0 :

1 : 0 BUN 460

…

Main Program

255 : CLE

256 :

…

PC

460 : …

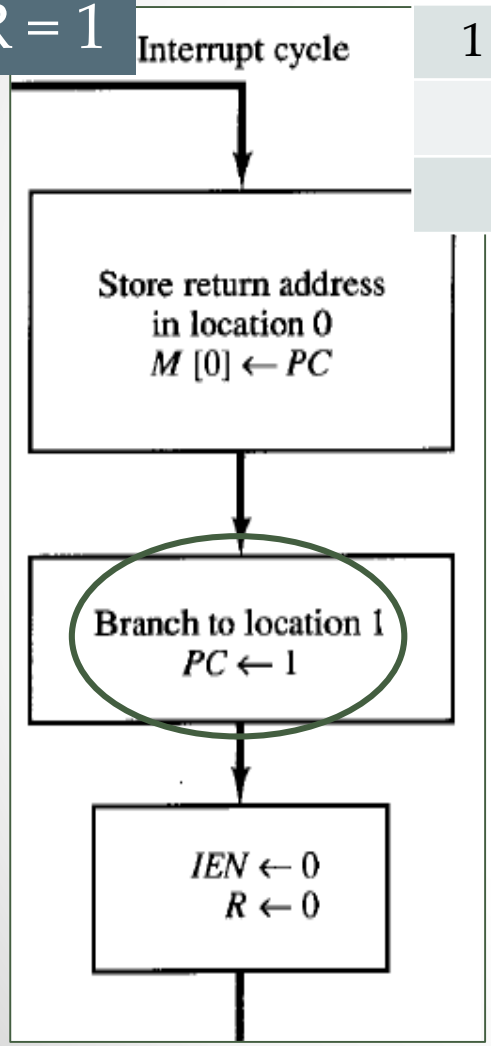
IO Routine

: ION

: 1 BUN 0

Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1	256	256	0	1	T_0
					T_1
					T_2

$R'T_0$: AR ← PC

RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0

0 :

1 : 0 BUN 460

…

Main Program

255 : CLE

256 :

…

PC

460 : …

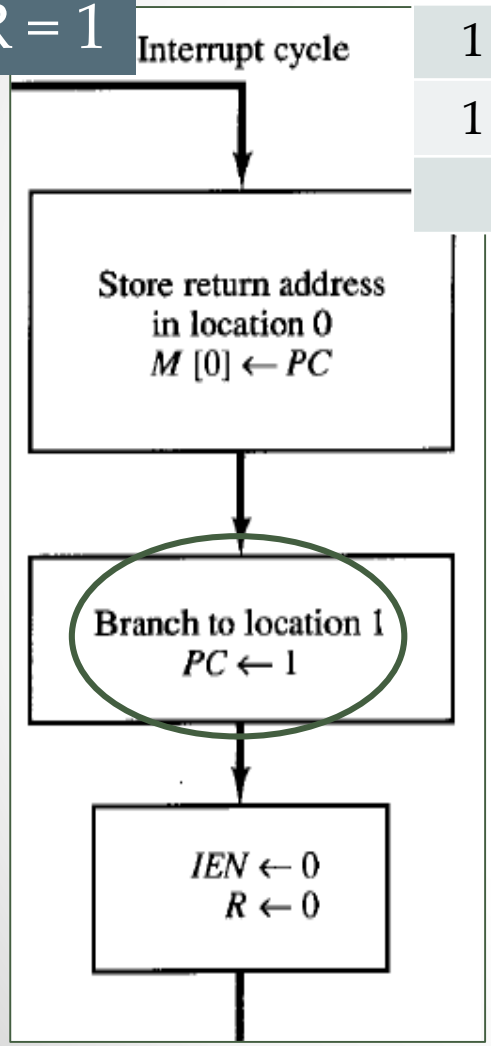
IO Routine

: ION

: 1 BUN 0

Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1	256	256	0	1	T_0
1	256	0	0	1	T_1
					T_2

$R'T_0$: AR ← PC

RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0

0 : 256

1 : 0 BUN 460

…

Main Program

255 : CLE

256 :

…

PC

460 : …

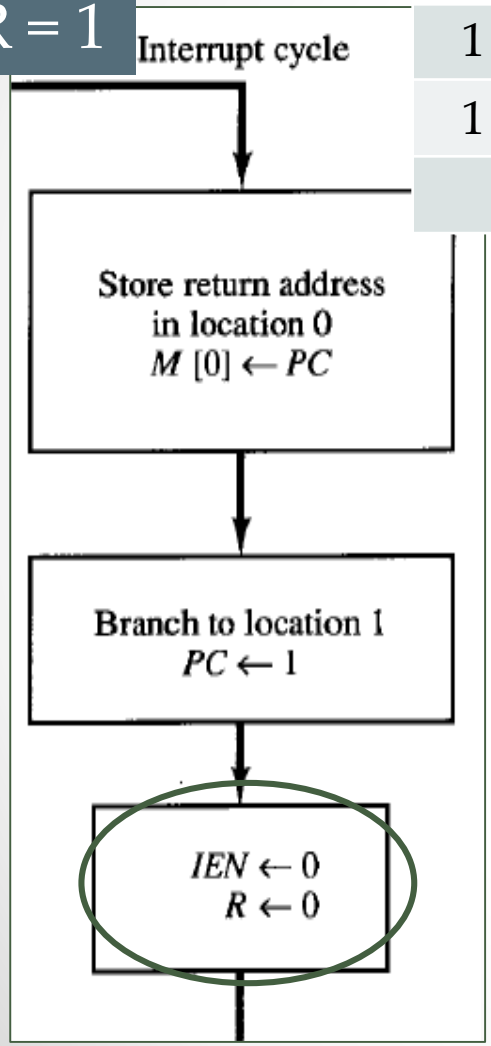
IO Routine

: ION

: 1 BUN 0

Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1	256	256	0	1	T_0
1	256	0	0	1	T_1
					T_2

$R'T_0$: AR ← PC

RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0

0 : **256**

1 : 0 BUN 460

…

Main Program

255 : CLE
256 :

…

460 : …

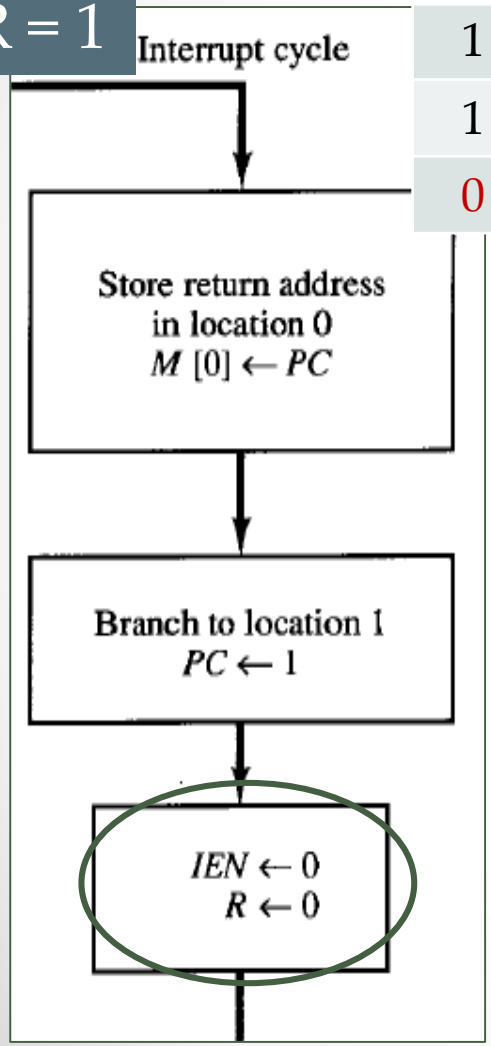
IO Routine

: ION
: 1 BUN 0

PC

Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1	256	256	0	1	T_0
1	256	0	0	1	T_1
0	256	1	0	0	T_2

$R'T_0$: AR ← PC

RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0

0 : **256**

1 : 0 BUN 460 ← PC

…

Main Program

255 : CLE
256 :

…

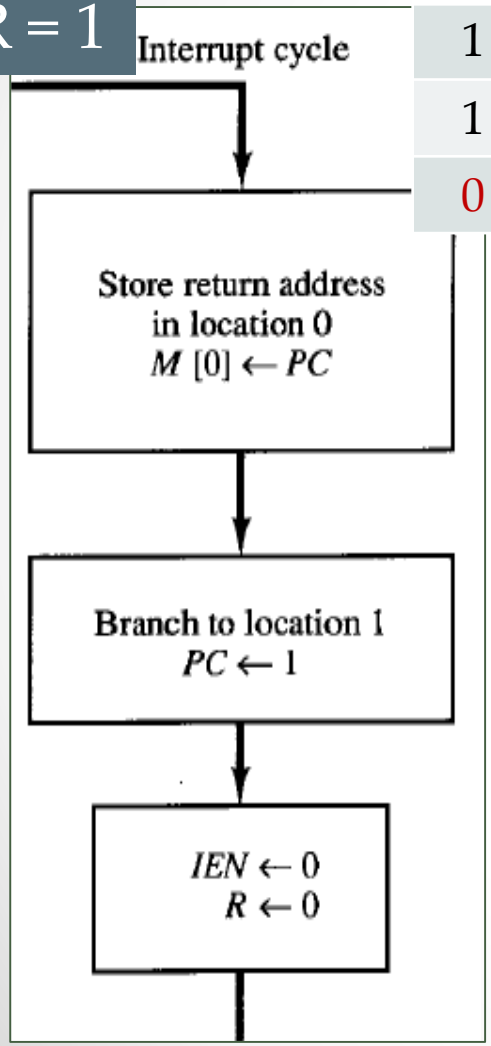
460 : …

IO Routine

: ION
: 1 BUN 0

Interrupt Cycle Execution

R = 1



IEN	TR	PC	AR	R	tick
1	256	256	0	1	T_0
1	256	0	0	1	T_1
0	256	1	0	0	T_2

$R'T_0$: AR ← PC

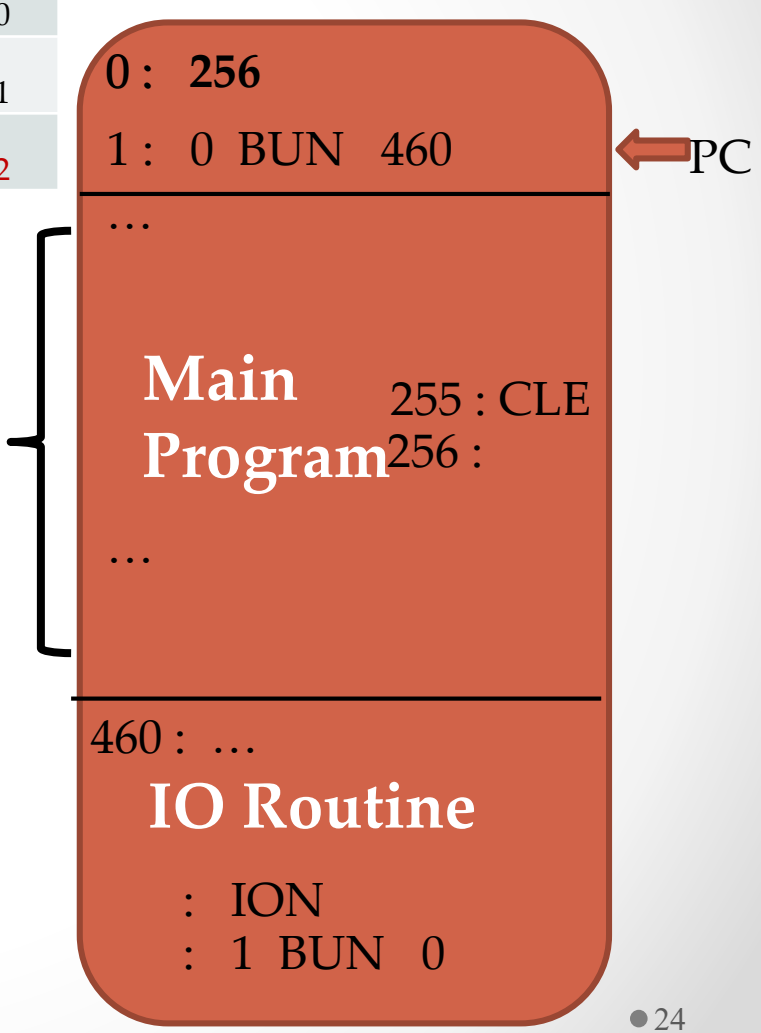
RT_0 : AR ← 0, TR ← PC

$R'T_1$: IR ← M[AR],
PC ← PC + 1

RT_1 : M[AR] ← TR ,
PC ← 0

$R'T_2$: D_0 ,.., D_7 ←
Decode IR(12-14)
AR ← IR(0-11),
I ← IR(15)

RT_2 : PC ← PC + 1,
IEN ← 0,
R ← 0, SC ← 0



Flowchart for computer operation

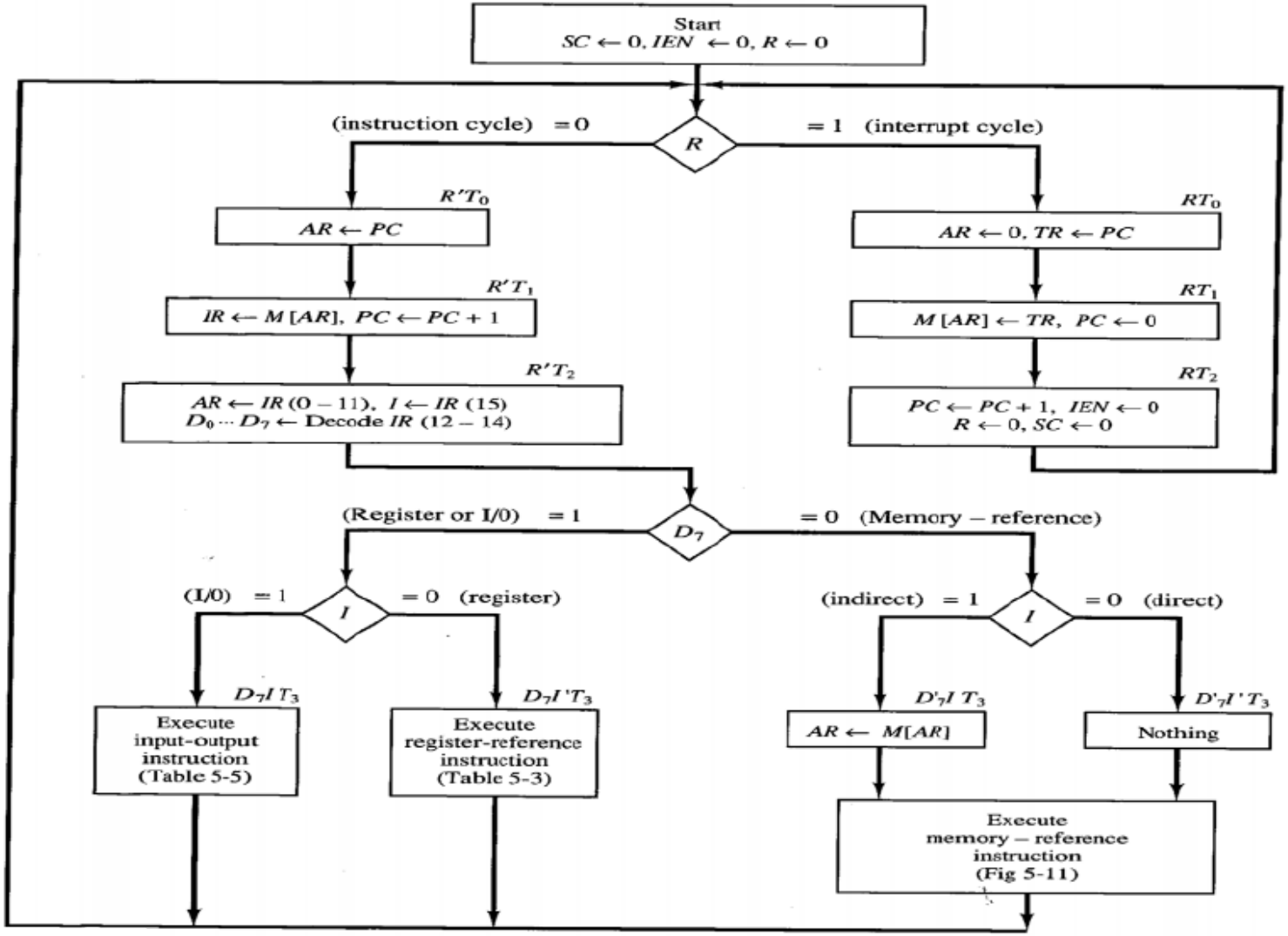


Figure 5-15 Flowchart for computer operation.

Summery of Control Functions & Microoperations

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$,
		$AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
$T_0'T_1'T_2'(IEN)(FGI + FGO)$:		$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

Summery of Control Functions & Microoperations

Register-reference:

$D_7I'T_3 = r$ (common to all register-reference instructions)

$IR(i) = B_i$ $(i = 0, 1, 2, \ldots, 11)$

	r:	$SC \leftarrow 0$
CLA	rB_{11}:	$AC \leftarrow 0$
CLE	rB_{10}:	$E \leftarrow 0$
CMA	rB_9:	$AC \leftarrow \overline{AC}$
CME	rB_8:	$E \leftarrow \overline{E}$
CIR	rB_7:	$AC \leftarrow \text{shr } AC,\quad AC(15) \leftarrow E,\quad E \leftarrow AC(0)$
CIL	rB_6:	$AC \leftarrow \text{shl } AC,\quad AC(0) \leftarrow E,\quad E \leftarrow AC(15)$
INC	rB_5:	$AC \leftarrow AC + 1$
SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$
SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$
SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1)$
SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$
HLT	rB_0:	$S \leftarrow 0$

Input-output:

$D_7IT_3 = p$ (common to all input–output instructions)

$IR(i) = B_i$ $(i = 6, 7, 8, 9, 10, 11)$

	p:	$SC \leftarrow 0$
INP	pB_{11}:	$AC(0\text{–}7) \leftarrow INPR,\quad FGI \leftarrow 0$
OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{–}7),\quad FGO \leftarrow 0$
SKI	pB_9:	If $(FGI = 1)$ then $(PC \leftarrow PC + 1)$
SKO	pB_8:	If $(FGO = 1)$ then $(PC \leftarrow PC + 1)$
ION	pB_7:	$IEN \leftarrow 1$
IOF	pB_6:	$IEN \leftarrow 0$

The Basic Computer Components

1. A memory unit with 4096 words of 16 bits each
2. Nine registers: *AR*, *PC*, *DR*, *AC*, *IR*, *TR*, *OUTR*, *INPR*, and *SC*
3. Seven flip-flops: *I*, *S*, *E*, *R*, *IEN*, *FGI*, and *FGO* **(JK or D).**
4. Two decoders: a 3 × 8 operation decoder and a 4 × 16 timing decoder
5. A 16-bit common bus **with 16 8×1 multiplexers**
6. Control logic gates
7. Adder and logic circuit connected to the input of *AC*

Control Logic Gates

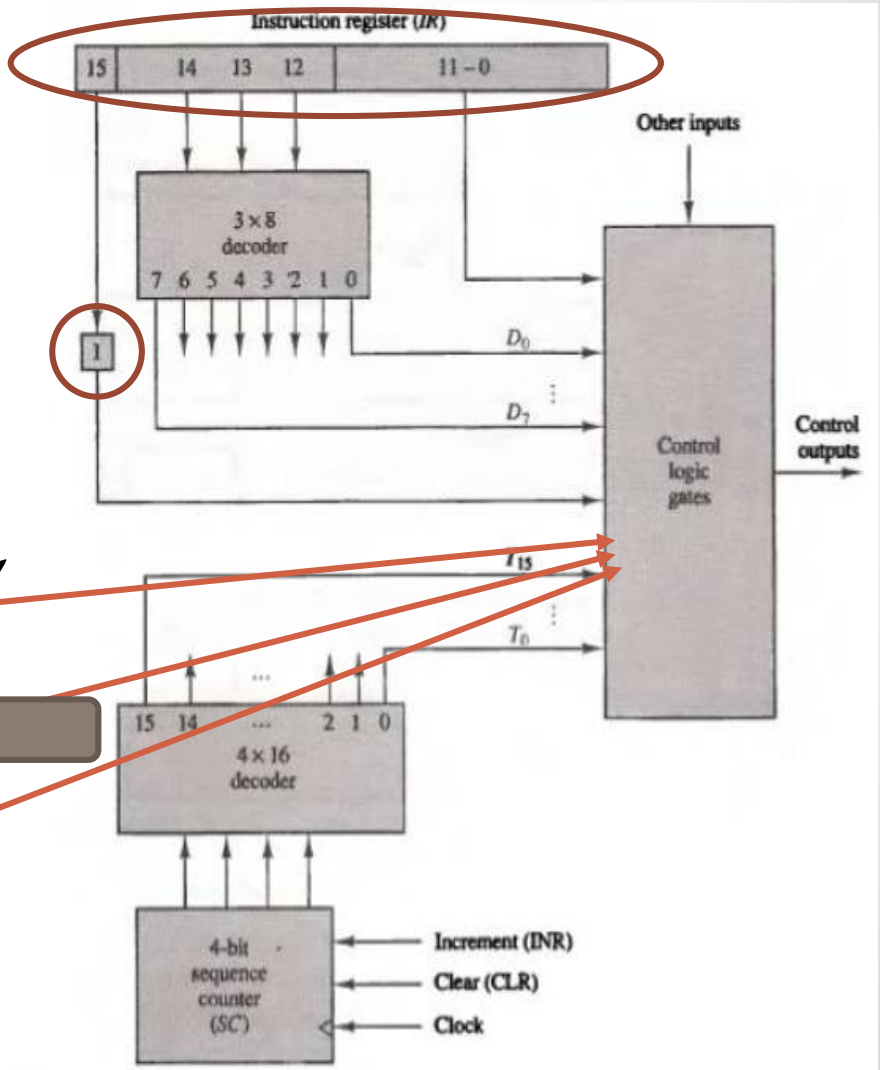
The inputs to this circuit comes from:

1. I
2. IR (11:0)
3. AC (15:0) [is AC = 0, is AC < 0, is AC ≥ 0]
4. DR (15:0) [is DR = 0]
5. S, E, R, IEN, FGI, FGO

AC

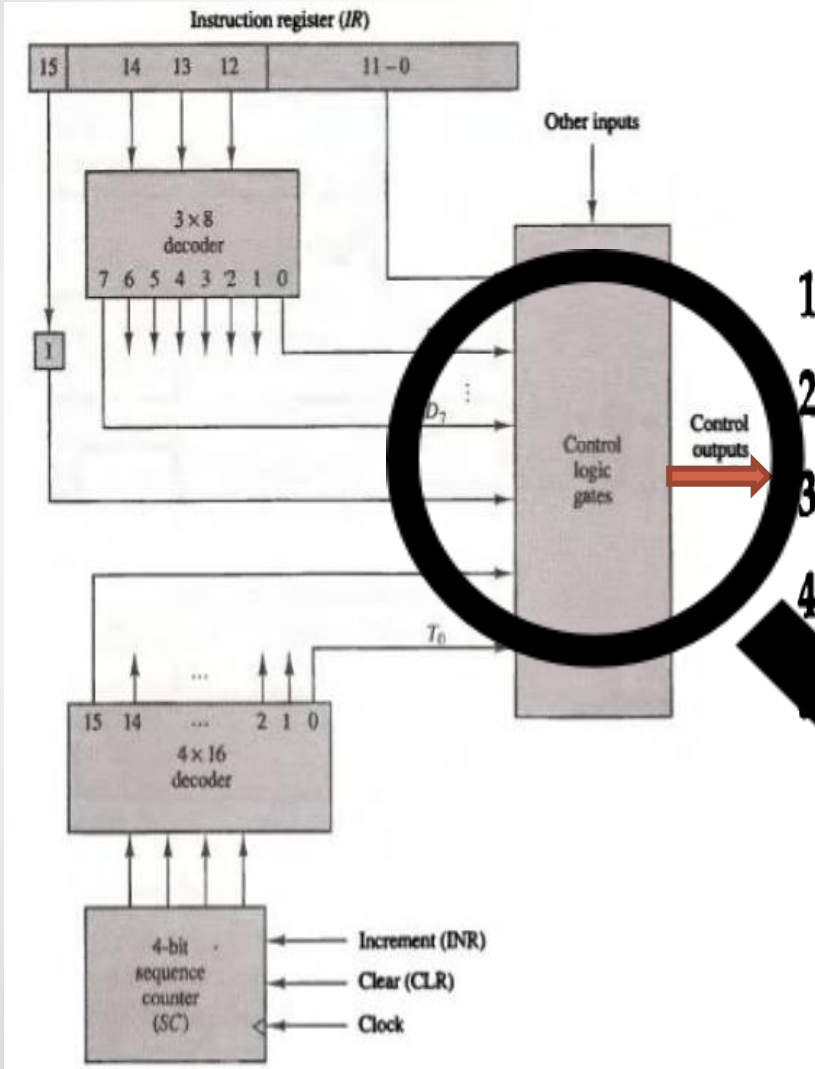
DR

S E R IEN FGI FGO



Control Logic Block Diagram

Control Logic Gates

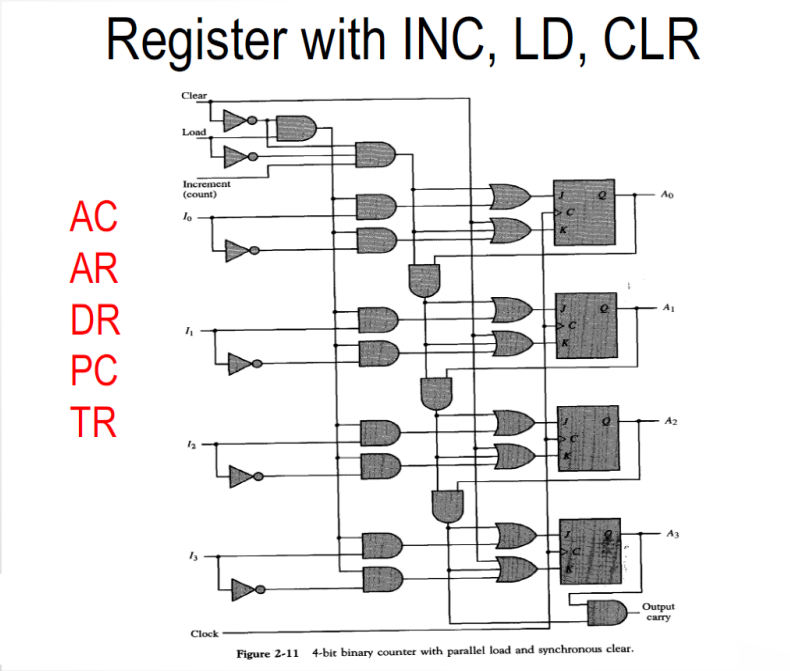


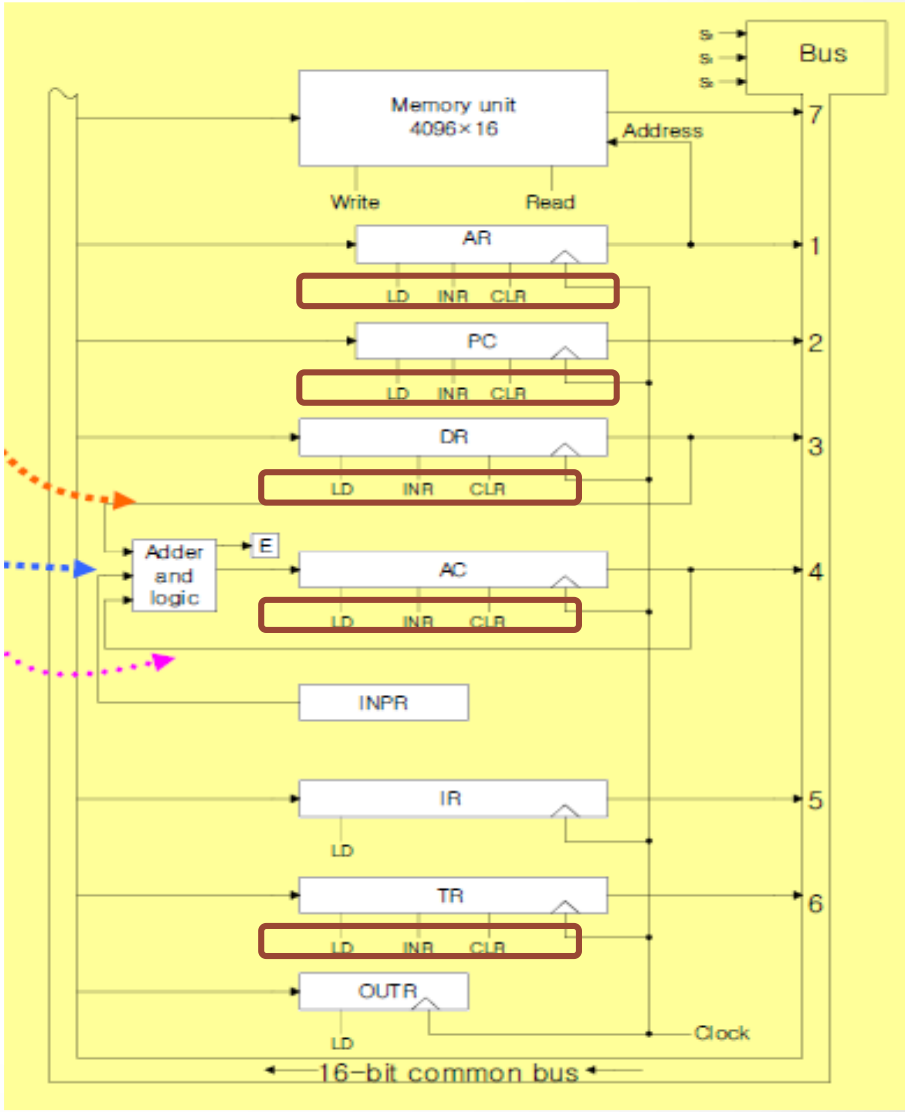
1. Signals to control the inputs of the nine registers
2. Signals to control the read and write inputs of memory
3. Signals to set, clear, or complement the flip-flops
4. Signals for S_2, S_1, and S_0 to select a register for the bus
5. Signals to control the *AC* adder and logic circuit

Control of Registers

Here, the control inputs of a register Ri are:

LD (Ri), **INC** (Ri), **CLR** (Ri)





Control of Registers – AR(1)

Look for statements that change the content of the register

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$,
		$AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

Control of Registers – AR(2)

Look for statements that change the content of the register

NONE

Register-reference:

$D_7 I' T_3 = r$ (common to all register-reference instructions)
$IR(i) = B_i$ $(i = 0, 1, 2, \ldots, 11)$

	r:	$SC \leftarrow 0$
CLA	rB_{11}:	$AC \leftarrow 0$
CLE	rB_{10}:	$E \leftarrow 0$
CMA	rB_9:	$AC \leftarrow \overline{AC}$
CME	rB_8:	$E \leftarrow \overline{E}$
CIR	rB_7:	$AC \leftarrow \text{shr } AC,\quad AC(15) \leftarrow E,\quad E \leftarrow AC(0)$
CIL	rB_6:	$AC \leftarrow \text{shl } AC,\quad AC(0) \leftarrow E,\quad E \leftarrow AC(15)$
INC	rB_5:	$AC \leftarrow AC + 1$
SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$
SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$
SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1)$
SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$
HLT	rB_0:	$S \leftarrow 0$

Input-output:

$D_7 I T_3 = p$ (common to all input–output instructions)
$IR(i) = B_i$ $(i = 6, 7, 8, 9, 10, 11)$

	p:	$SC \leftarrow 0$
INP	pB_{11}:	$AC(0\text{–}7) \leftarrow INPR,\quad FGI \leftarrow 0$
OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{–}7),\quad FGO \leftarrow 0$
SKI	pB_9:	If $(FGI = 1)$ then $(PC \leftarrow PC + 1)$
SKO	pB_8:	If $(FGO = 1)$ then $(PC \leftarrow PC + 1)$
ION	pB_7:	$IEN \leftarrow 1$
IOF	pB_6:	$IEN \leftarrow 0$

Control of Registers – AR(3)

Extract the RTL statements that was chosen

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$,
		$AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
$T_0'T_1'T_2'(IEN)(FGI + FGO)$:		$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

$R'T_0 : AR \leftarrow PC$

$R'T_2 : AR \leftarrow IR(0\text{-}11)$

$D_7'IT_3 : AR \leftarrow M[AR]$

$RT_0 : AR \leftarrow 0$

$D_5T_4 : AR \leftarrow AR + 1$

Control of Registers – AR(4)

Divide the RTL statements according to the register control inputs

The control inputs of register AR are:

LD (R), **INC** (R), **CLR** (R)

LD (R) = $R'T_0 + R'T_2 + D_7'IT_3$

INC (R) = D_5T_4

CLR (R) = RT_0

$R'T_0 : AR \leftarrow PC$

$R'T_2 : AR \leftarrow IR(0\text{-}11)$

$D_7'IT_3 : AR \leftarrow M[AR]$

$RT_0 : AR \leftarrow 0$

$D_5T_4 : AR \leftarrow AR + 1$

Control of Registers – AR(5)

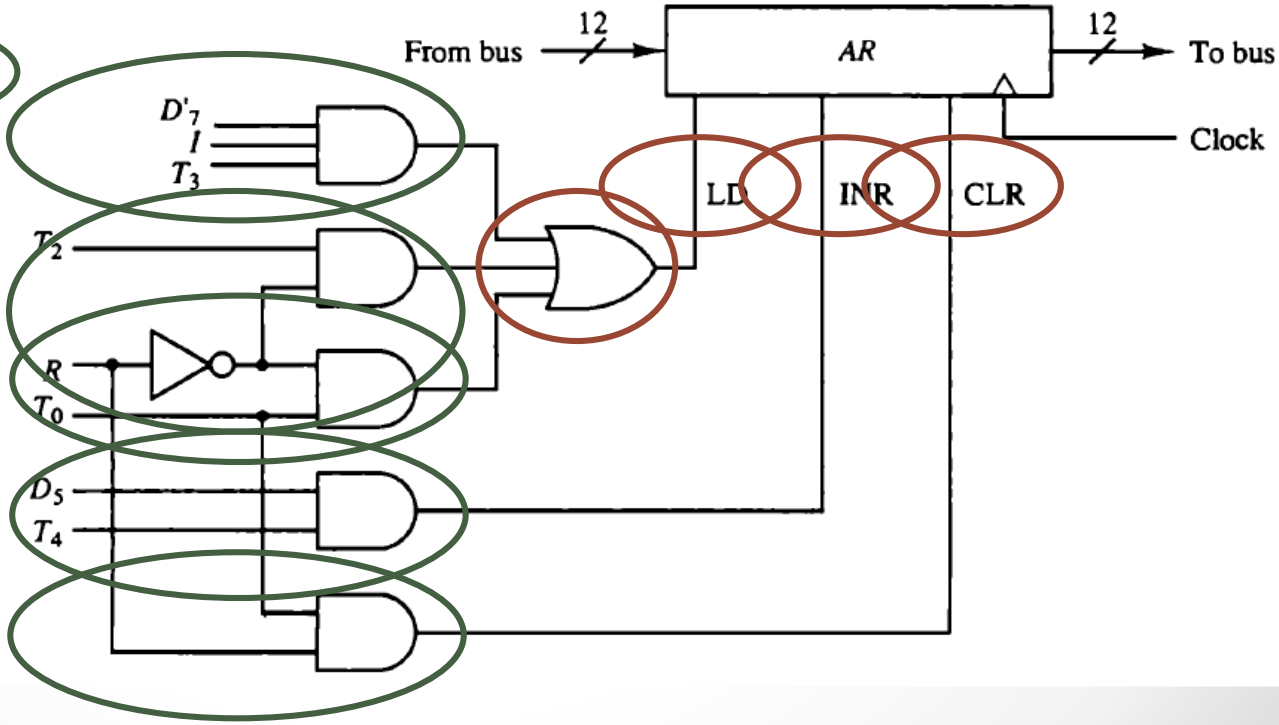
Construct the logic circle for these inputs

LD (R) = $R'T_0 + R'T_2 + D_7'IT_3$

INC (R) = D_5T_4

CLR (R) = RT_0

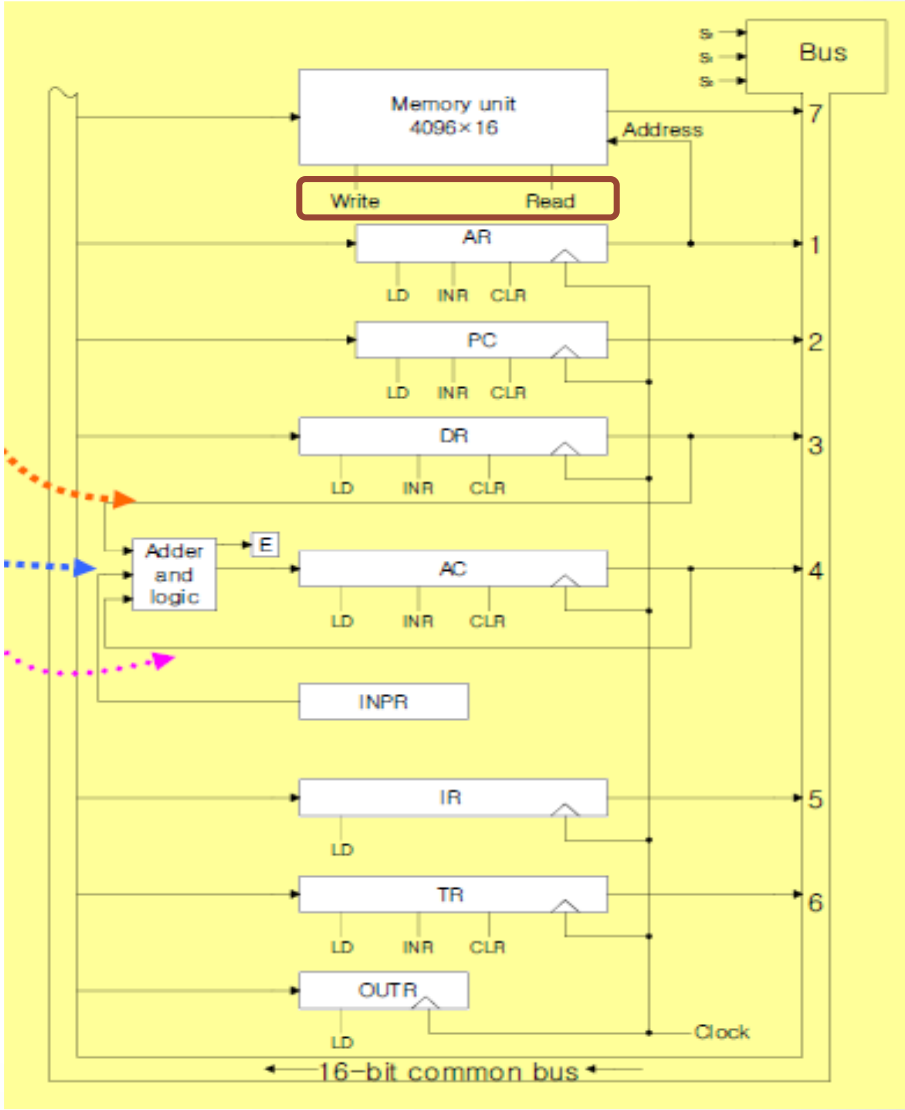
Figure 5-16 Control gates associated with *AR*.



Control of Memory

Here, the control inputs of the memory are:

READ, WRITE



Control of Memory

Look for statements that READs from the memory: ← M[AR]

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$, $AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

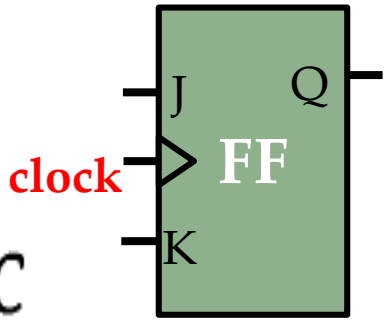
Control of Memory

$READ = R'T_1 + D_7'IT_3 + (D_0 + D_1 + D_2 + D_6)T_4$

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$, $AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
$T_0'T_1'T_2'(IEN)(FGI + FGO)$:		$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

Flip-flop Control

1. A memory unit with 4096 words of 16 bits each
2. Nine registers: *AR*, *PC*, *DR*, *AC*, *IR*, *TR*, *OUTR*, *INPR*, and *SC*
3. Seven flip-flops: *I*, *S*, *E*, *R*, *IEN*, *FGI*, and *FGO*
4. Two decoders: a 3 × 8 operation decoder and a 4 × 16 timing decoder
5. A 16-bit common bus
6. Control logic gates
7. Adder and logic circuit connected to the input of *AC*



Flip-flop Control - IEN

Look for statements that change the content of IEN

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$, $AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

$RT_2 : IEN \leftarrow 0$

Flip-flop Control - IEN

Look for statements that change the content of IEN

Register-reference:

$D_7I'T_3 = r$ (common to all register-reference instructions)
$IR(i) = B_i$ $(i = 0, 1, 2, \ldots, 11)$

	r:	$SC \leftarrow 0$
CLA	rB_{11}:	$AC \leftarrow 0$
CLE	rB_{10}:	$E \leftarrow 0$
CMA	rB_9:	$AC \leftarrow \overline{AC}$
CME	rB_8:	$E \leftarrow \overline{E}$
CIR	rB_7:	$AC \leftarrow \text{shr } AC,\quad AC(15) \leftarrow E,\quad E \leftarrow AC(0)$
CIL	rB_6:	$AC \leftarrow \text{shl } AC,\quad AC(0) \leftarrow E,\quad E \leftarrow AC(15)$
INC	rB_5:	$AC \leftarrow AC + 1$
SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$
SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$
SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1)$
SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$
HLT	rB_0:	$S \leftarrow 0$

Input-output:

$D_7IT_3 = p$ (common to all input–output instructions)
$IR(i) = B_i$ $(i = 6, 7, 8, 9, 10, 11)$

	p:	$SC \leftarrow 0$
INP	pB_{11}:	$AC(0\text{–}7) \leftarrow INPR,\quad FGI \leftarrow 0$
OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{–}7),\quad FGO \leftarrow 0$
SKI	pB_9:	If $(FGI = 1)$ then $(PC \leftarrow PC + 1)$
SKO	pB_8:	If $(FGO = 1)$ then $(PC \leftarrow PC + 1)$
ION	pB_7:	$IEN \leftarrow 1$
IOF	pB_6:	$IEN \leftarrow 0$

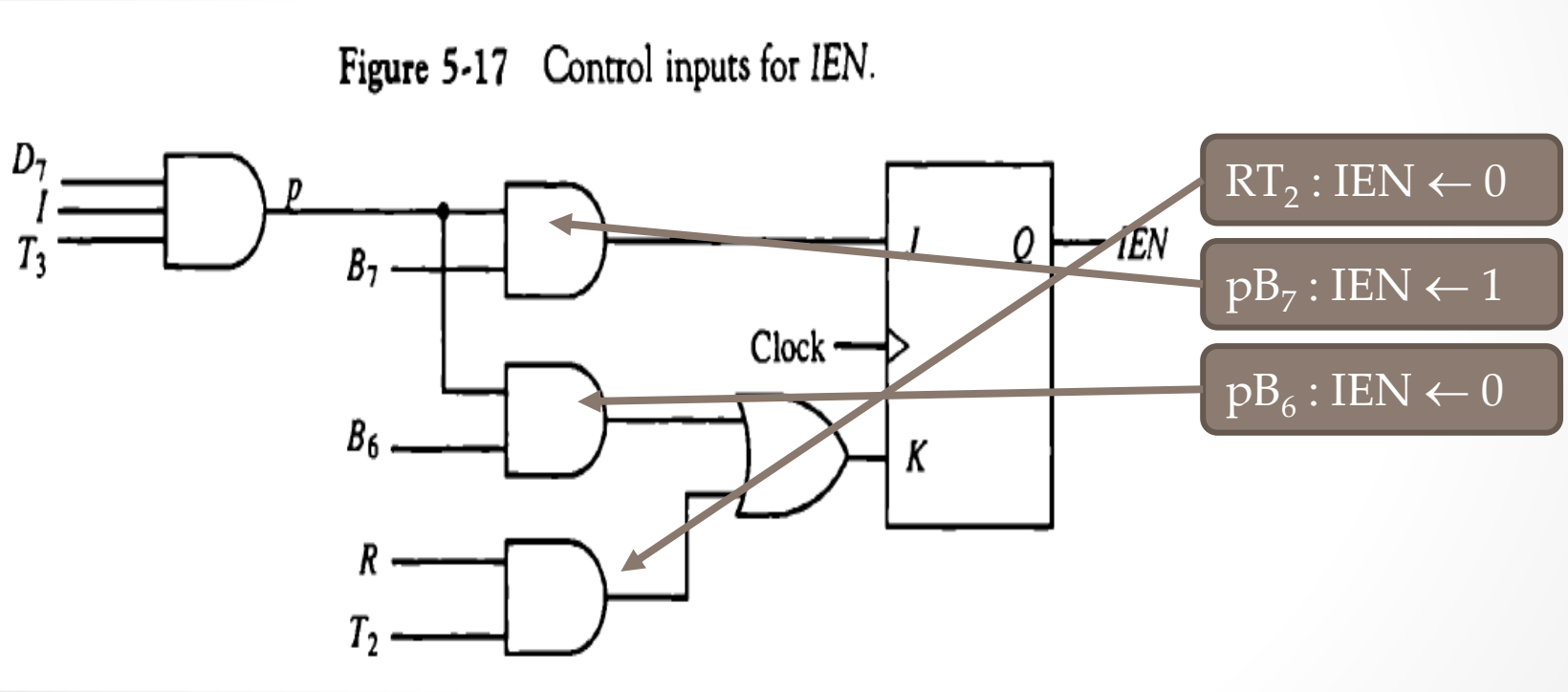
$pB_7 : IEN \leftarrow 1$

$pB_6 : IEN \leftarrow 0$

$p = D_7IT_3$

Flip-flop Control - IEN

Construct the logic circle for these inputs



The Bus Control

The bus is controlled by the selection inputs : $S_2\ S_1\ S_0$

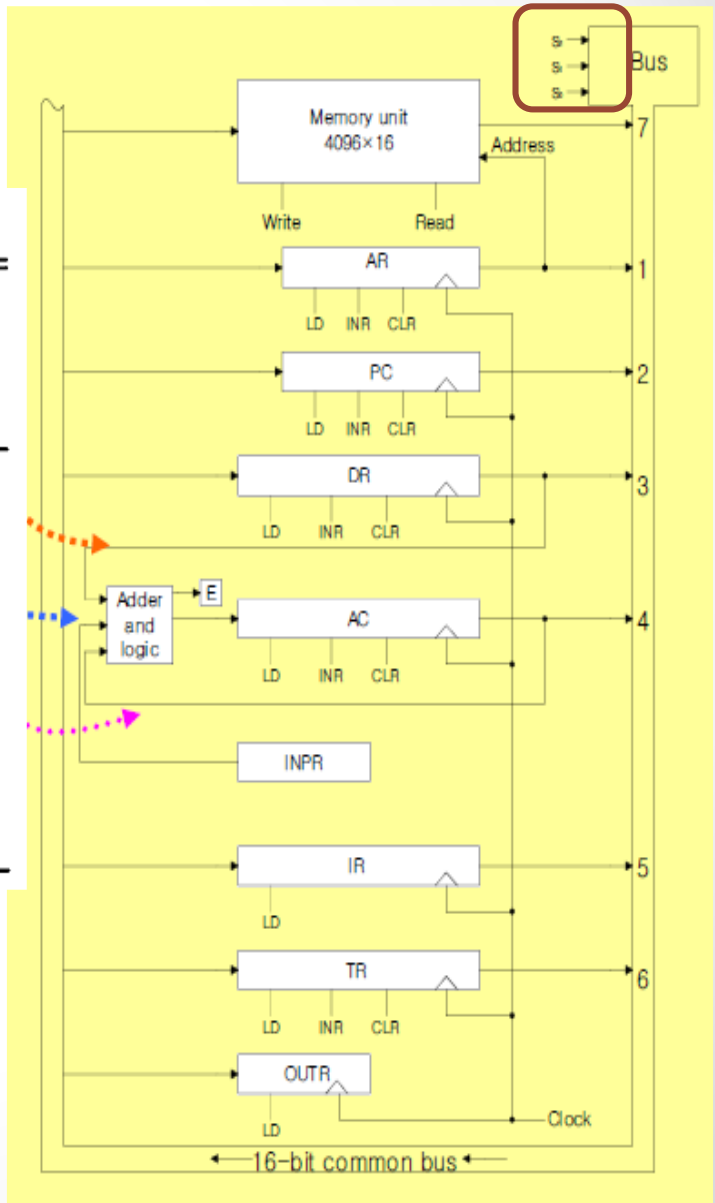
TABLE 5-7 Encoder for Bus Selection Circuit

Inputs							Outputs			Register selected for bus
x_1	x_2	x_3	x_4	x_5	x_6	x_7	S_2	S_1	S_0	
0	0	0	0	0	0	0	0	0	0	None
1	0	0	0	0	0	0	0	0	1	*AR*
0	1	0	0	0	0	0	0	1	0	*PC*
0	0	1	0	0	0	0	0	1	1	*DR*
0	0	0	1	0	0	0	1	0	0	*AC*
0	0	0	0	1	0	0	1	0	1	*IR*
0	0	0	0	0	1	0	1	1	0	*TR*
0	0	0	0	0	0	1	1	1	1	Memory

$S_0 = x_1 + x_3 + x_5 + x_7$

$S_1 = x_2 + x_3 + x_6 + x_7$

$S_2 = x_4 + x_5 + x_6 + x_7$



The Bus Control

The bus is controlled by the selection inputs : $S_2\ S_1\ S_0$

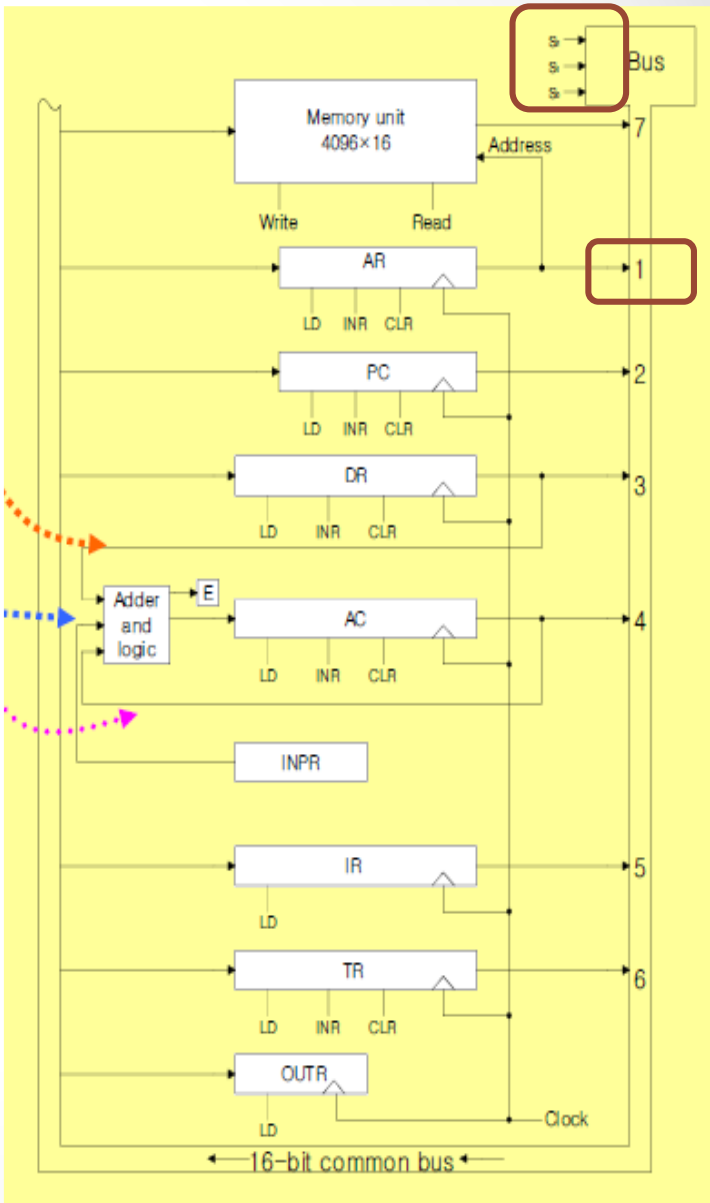
$$S_0 = x_1 + x_3 + x_5 + x_7$$

$$S_1 = x_2 + x_3 + x_6 + x_7$$

$$S_2 = x_4 + x_5 + x_6 + x_7$$

When $x_1 = 1$, the value of $S_2S_1S_0$ must be 001, and the output of AR will be selected for the bus.

Look for : .. ← AR



The Bus Control

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$,
		$AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	$D_7'IT_3$:	$AR \leftarrow M[AR]$
Interrupt:		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$

$$x_1 = D_4T_4 + D_5T_5$$

The Bus Control

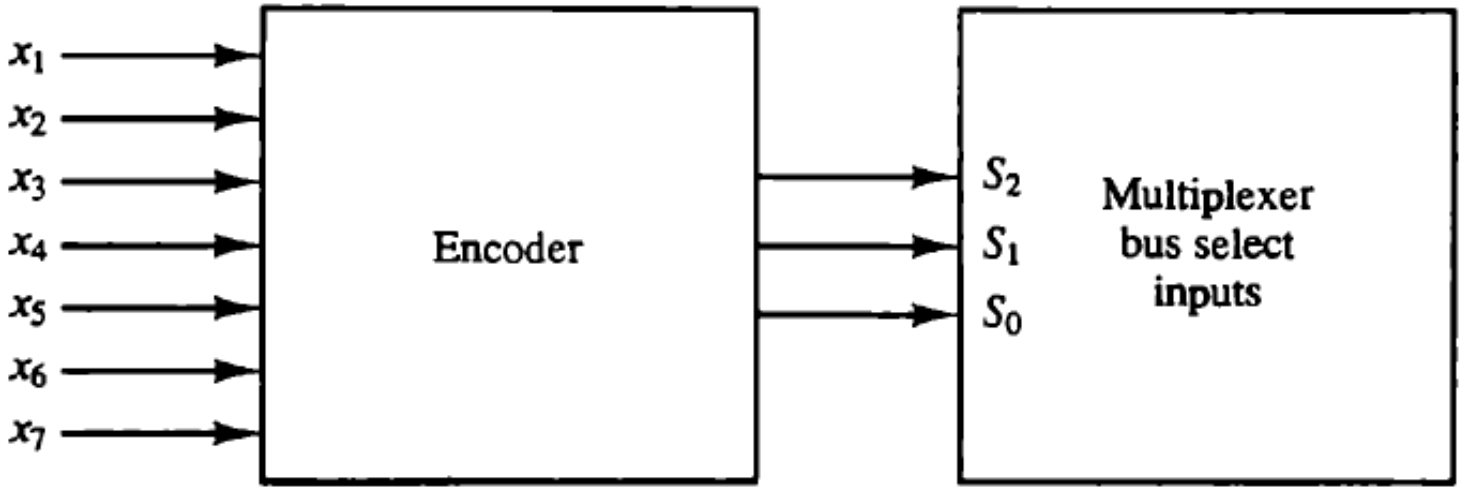
The Boolean functions for the Encoder

$$S_0 = x_1 + x_3 + x_5 + x_7$$

$$S_1 = x_2 + x_3 + x_6 + x_7$$

$$S_2 = x_4 + x_5 + x_6 + x_7$$

Figure 5-18 Encoder for bus selection inputs.

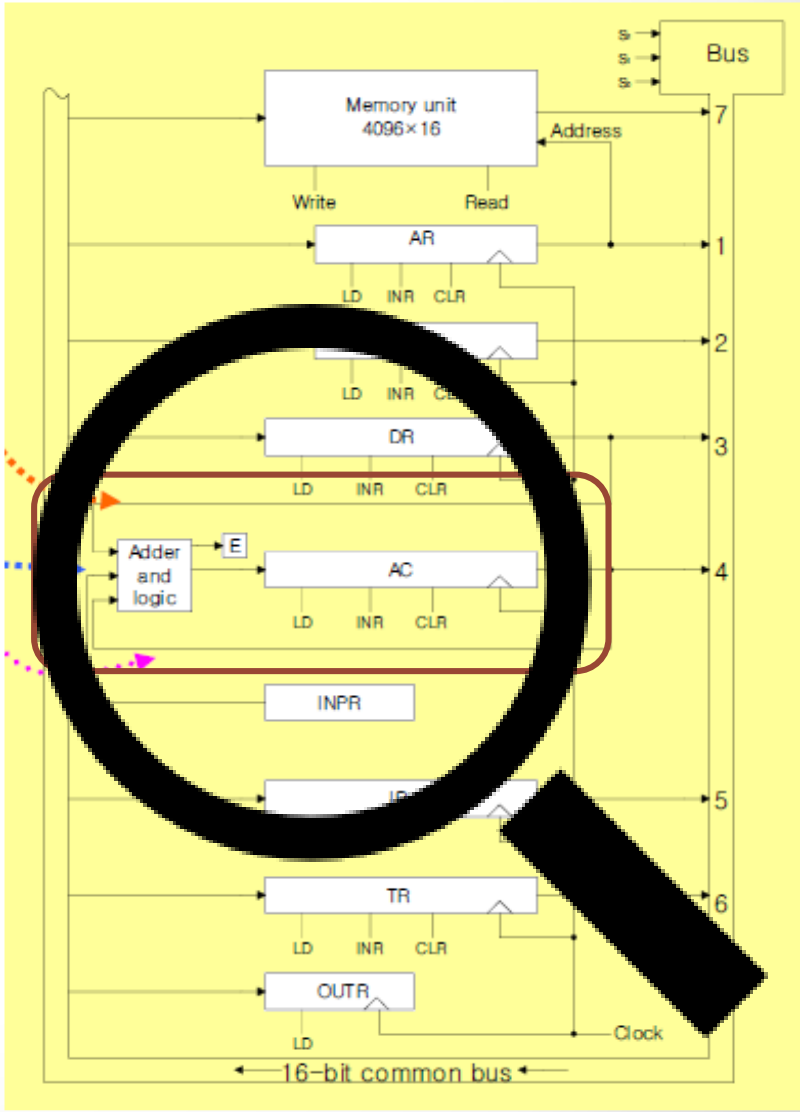


The Basic Computer Components

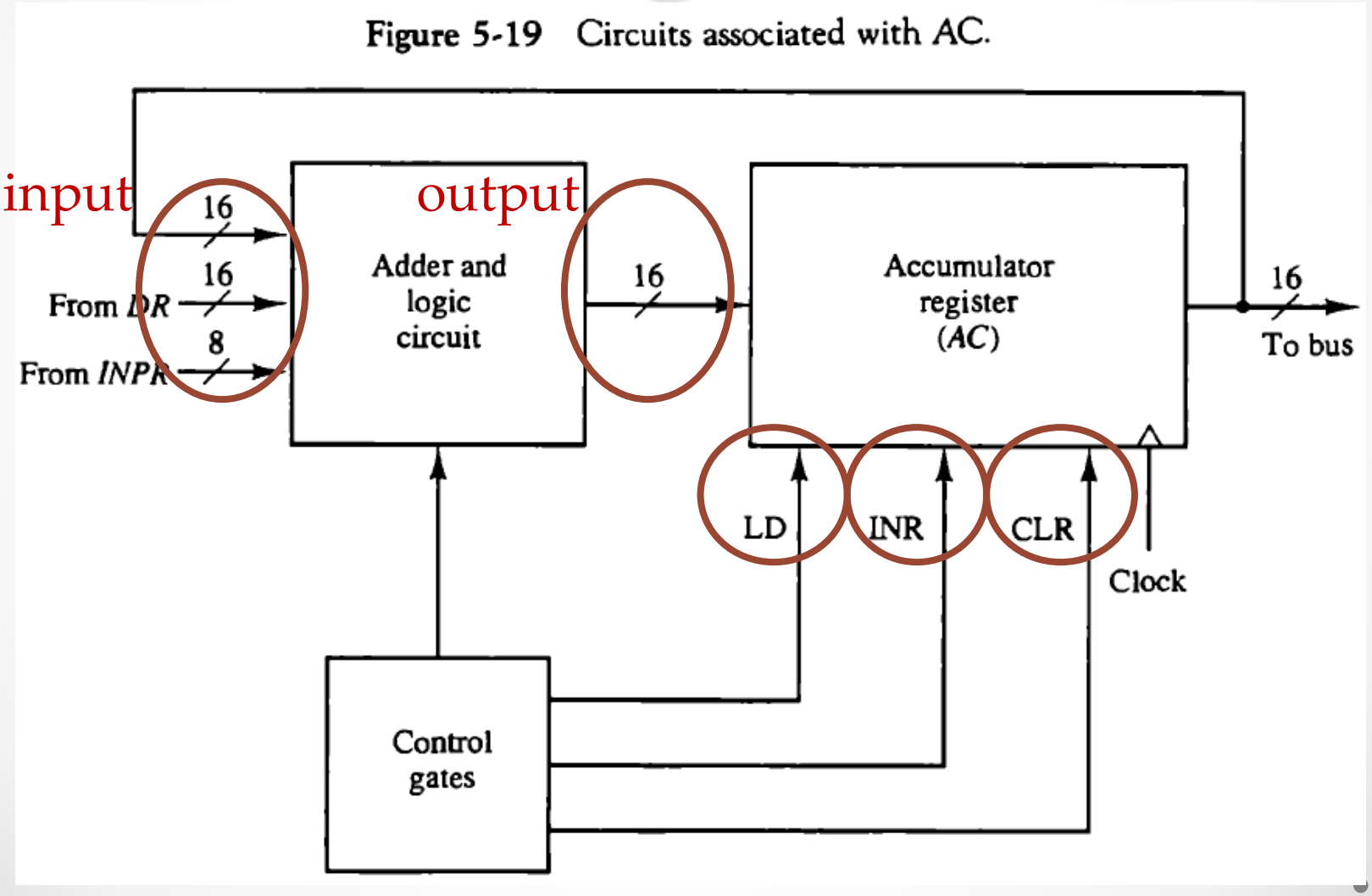
1. A memory unit with 4096 words of 16 bits each
2. Nine registers: *AR*, *PC*, *DR*, *AC*, *IR*, *TR*, *OUTR*, *INPR*, and *SC*
3. Seven flip-flops: *I*, *S*, *E*, *R*, *IEN*, *FGI*, and *FGO* **(JK or D).**
4. Two decoders: a 3 × 8 operation decoder and a 4 × 16 timing decoder
5. A 16-bit common bus **with 16 8×1 multiplexers**
6. Control logic gates
7. Adder and logic circuit connected to the input of *AC*

Design the Accumulator Logic

The circuits associated with the AC register are shown here:



Design the Accumulator Logic



Design the AC Control

Look for statements that change the AC register

LOAD

D_0T_5:	$AC \leftarrow AC \wedge DR$	AND with *DR*
D_1T_5:	$AC \leftarrow AC + DR$	Add with *DR*
D_2T_5:	$AC \leftarrow DR$	Transfer from *DR*
pB_{11}:	$AC(0\text{–}7) \leftarrow INPR$	Transfer from *INPR*
rB_9:	$AC \leftarrow \overline{AC}$	Complement
rB_7:	$AC \leftarrow \text{shr } AC, \quad AC(15) \leftarrow E$	Shift right
rB_6:	$AC \leftarrow \text{shl } AC, \quad AC(0) \leftarrow E$	Shift left
rB_{11}:	$AC \leftarrow 0$	Clear
rB_5:	$AC \leftarrow AC + 1$	Increment

Design the AC Logic

D_0T_5: $AC \leftarrow AC \wedge DR$

D_1T_5: $AC \leftarrow AC + DR$

D_2T_5: $AC \leftarrow DR$

pB_{11}: $AC(0\text{–}7) \leftarrow INPR$

rB_9: $AC \leftarrow \overline{AC}$

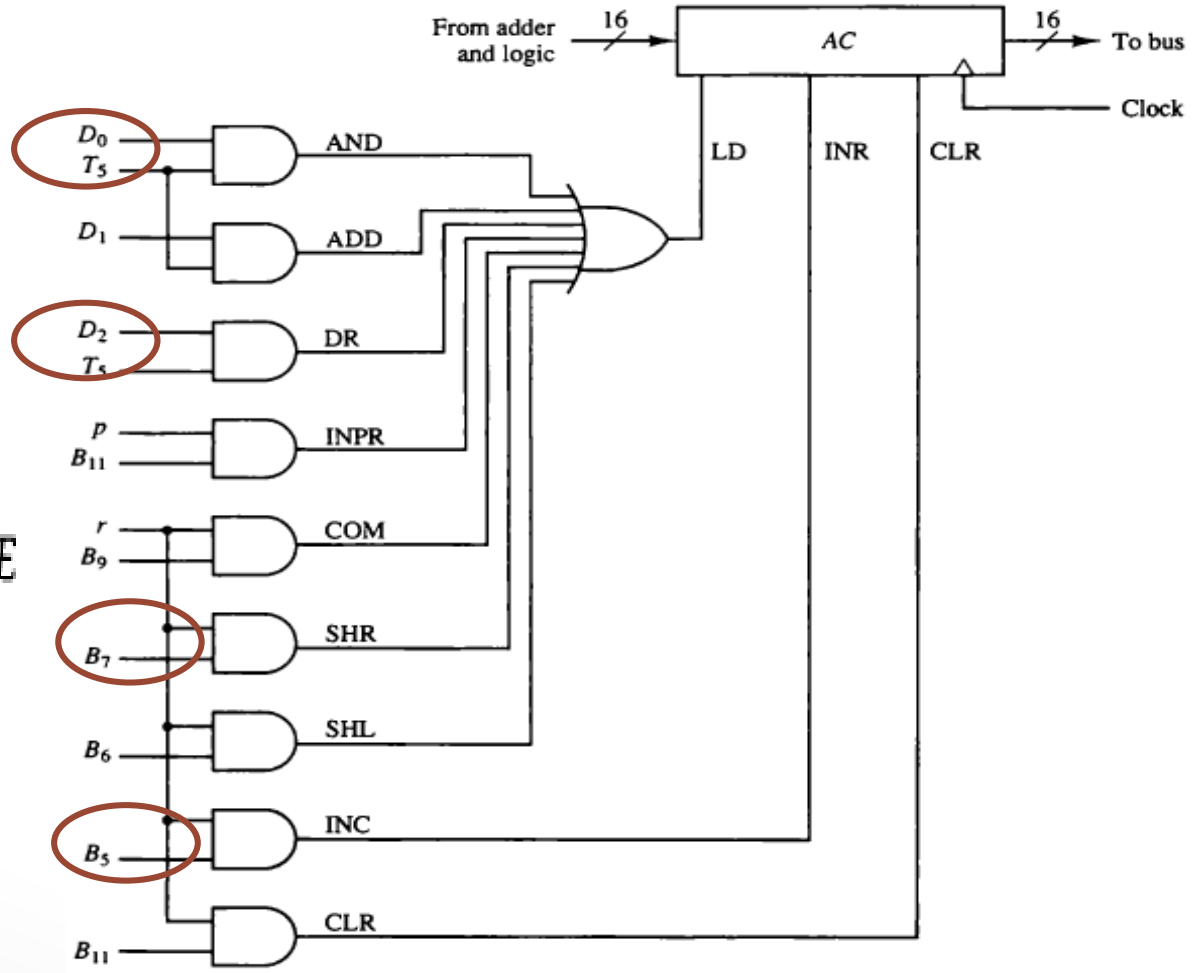
rB_7: $AC \leftarrow \text{shr } AC,\quad AC(15) \leftarrow E$

rB_6: $AC \leftarrow \text{shl } AC,\quad AC(0) \leftarrow E$

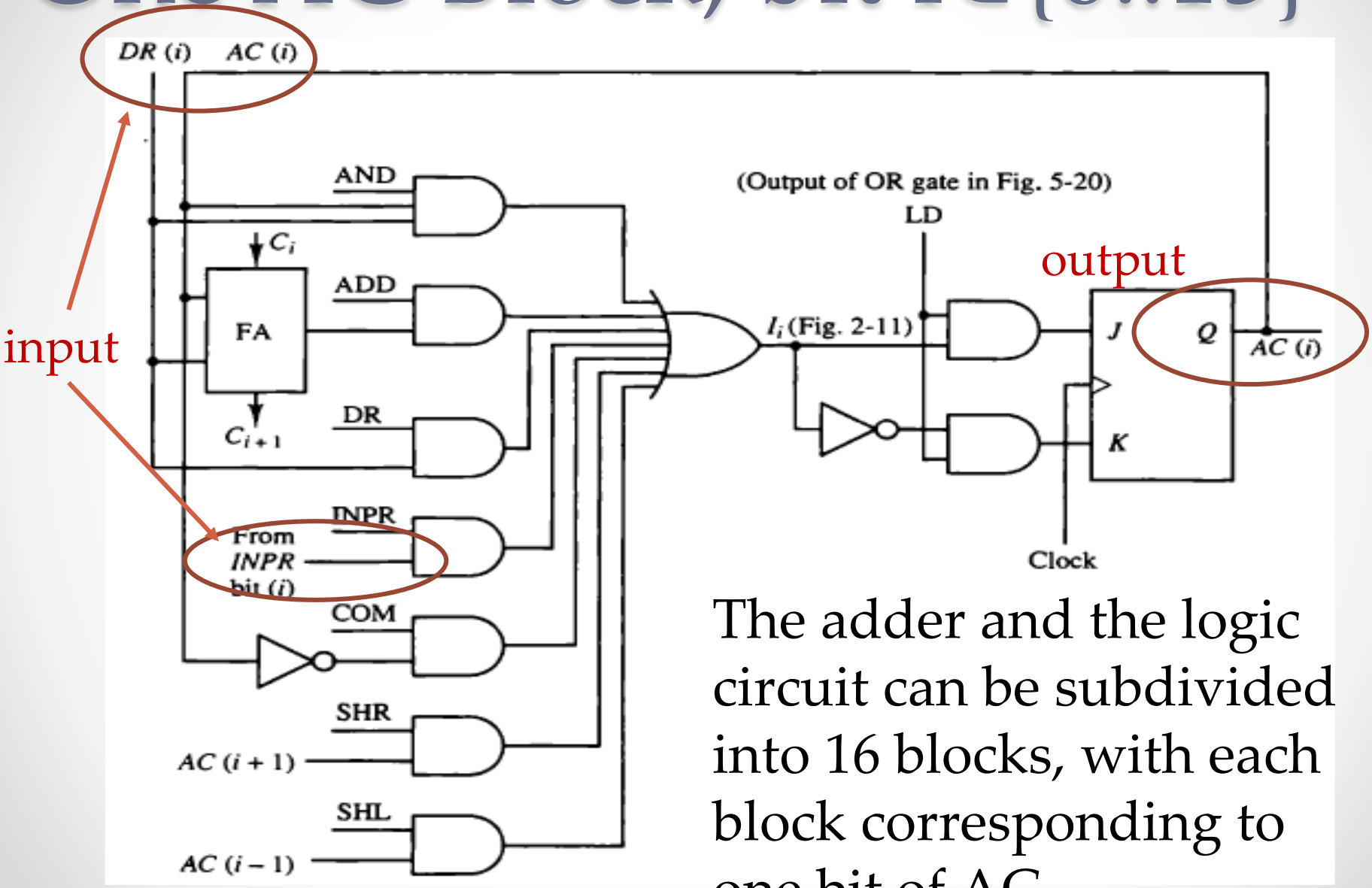
rB_{11}: $AC \leftarrow 0$

rB_5: $AC \leftarrow AC + 1$

Figure 5-20 Gate structure for controlling the LD, INR, and CLR of AC.



One AC Block, bit i∈{0..15}



The adder and the logic circuit can be subdivided into 16 blocks, with each block corresponding to one bit of AC.

One AC Block, bit i∈{0..15}

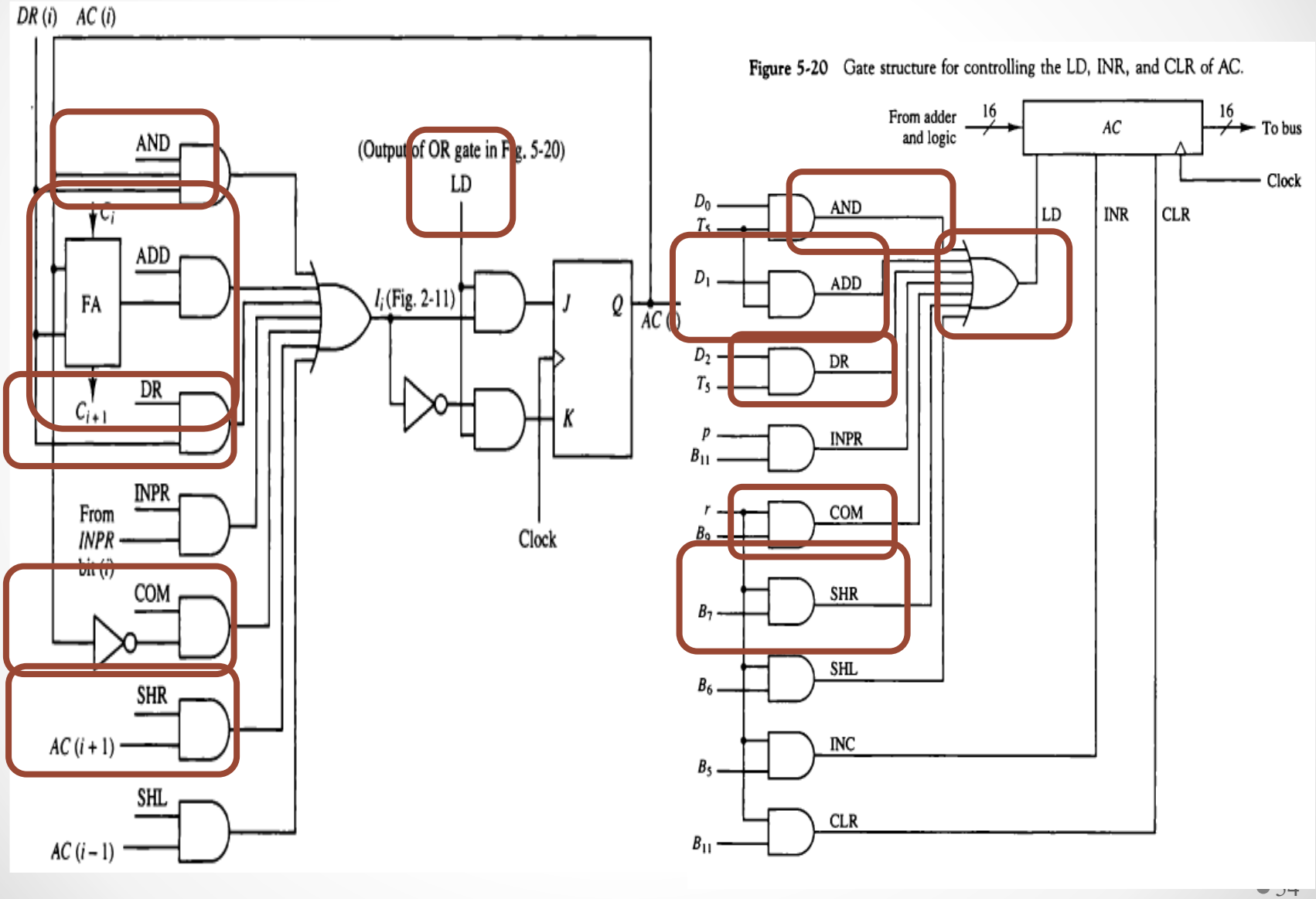


Figure 5-20 Gate structure for controlling the LD, INR, and CLR of AC.