Computer Architecture Lec 5a

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.**

Register Reference Instructions

- Executed with the clock transition associated with timing variable T_3.
- Each control function needs the Boolean relation $D_7I'T_3$
  - for convenience let $r \equiv D_7I'T_3$.
- The control function is distinguished by one of the bits in **IR(0-11)**.
- Assign the symbol B_i to bit i of **IR**,
  - all control functions can be simply denoted by rB_i.
- After completion
  - The sequence counter **SC** is cleared to **0**
  - The control goes back to fetch the next instruction with timing signal T_0.

Alon Schclar, Tel-Aviv College, 2009

Register Reference Instructions

TABLE 5-3 Execution of Register-Reference Instructions

$D_7I'T_3 = r$ (common to all register-reference instructions)
$IR(i) = B_i$ [bit in $IR(0–11)$ that specifies the operation]

	r:	$SC \leftarrow 0$	Clear SC
CLA	rB_{11}:	$AC \leftarrow 0$	Clear AC
CLE	rB_{10}:	$E \leftarrow 0$	Clear E
CMA	rB_9:	$AC \leftarrow \overline{AC}$	Complement AC
CME	rB_8:	$E \leftarrow \overline{E}$	Complement E
CIR	rB_7:	$AC \leftarrow \text{shr } AC, AC(15) \leftarrow E, E \leftarrow AC(0)$	Circulate right
CIL	rB_6:	$AC \leftarrow \text{shl } AC, AC(0) \leftarrow E, E \leftarrow AC(15)$	Circulate left
INC	rB_5:	$AC \leftarrow AC + 1$	Increment AC
SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$	Skip if positive
SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$	Skip if negative
SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1$	Skip if AC zero
SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$	Skip if E zero
HLT	rB_0:	$S \leftarrow 0$ (S is a start–stop flip-flop)	Halt computer

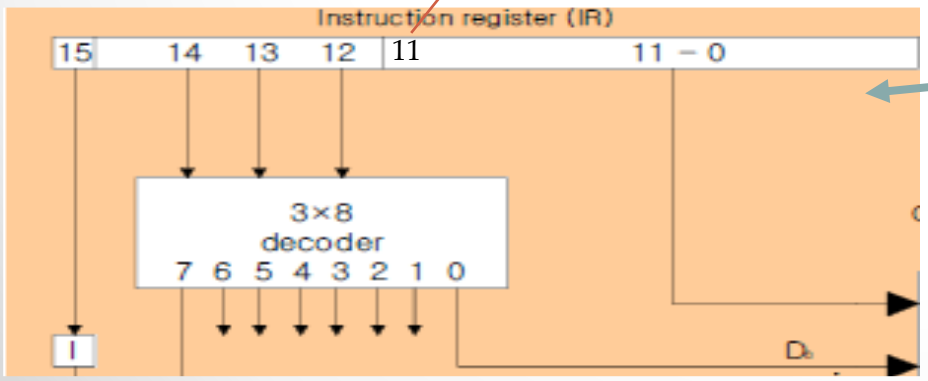
if-else loops

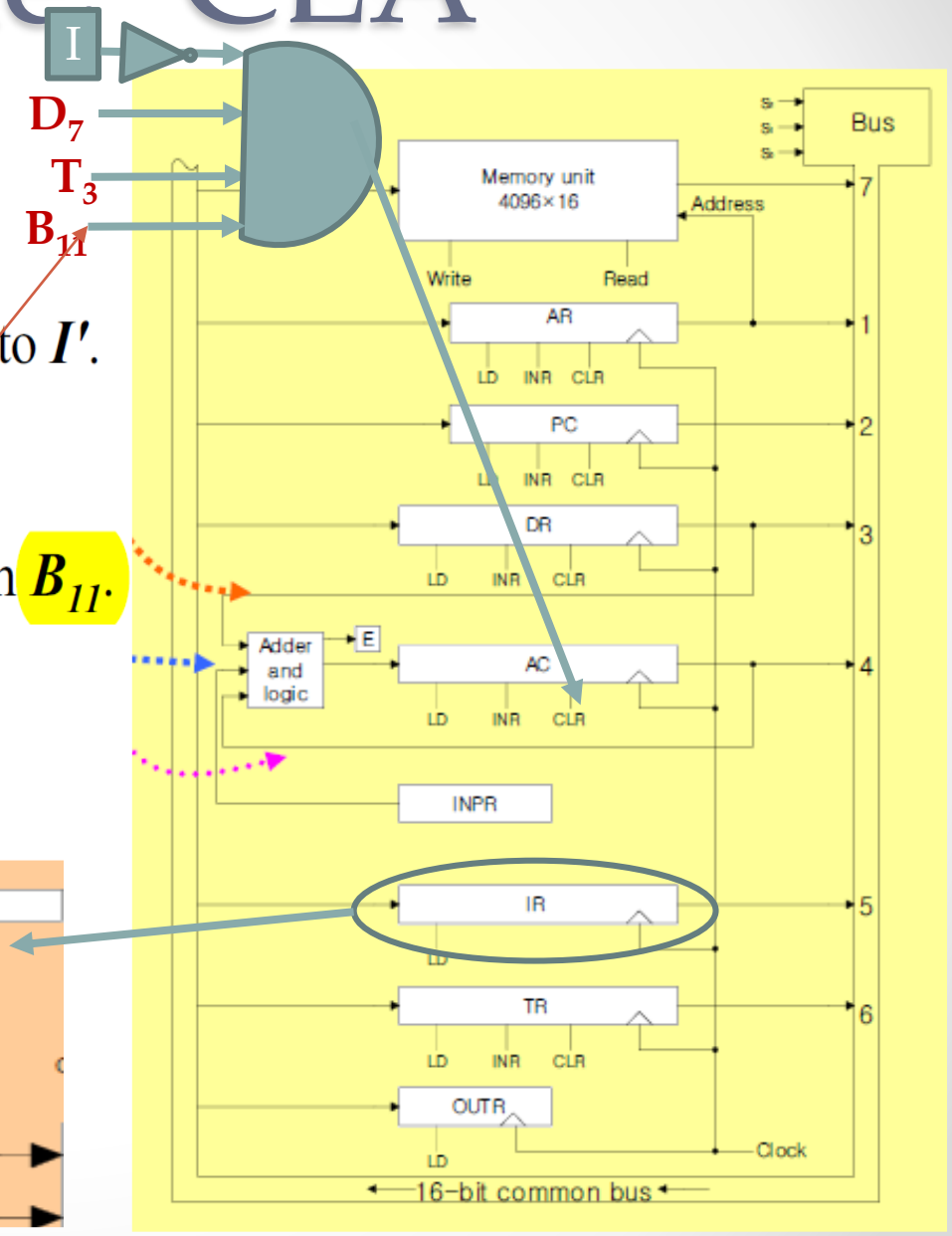
Taken from: M. Mano/Computer Design and Architecture 3rd Ed.

Alon Schclar, Tel-Aviv College, 2009

Example: CLA

- Hexadecimal code
  $(7800)_{16} = (0111\ 1000\ 0000\ 0000)_2$.
- The first bit is a zero and is equivalent to I'.
- Next three bits are the opcode and are recognized from decoder output D_7.
- Bit *11* in **IR** is *1* and is recognized from B_{11}.
- The control function that initiates the microoperation for this instruction is
  $D_7I'T_3B_{11} = rB_{11}$.





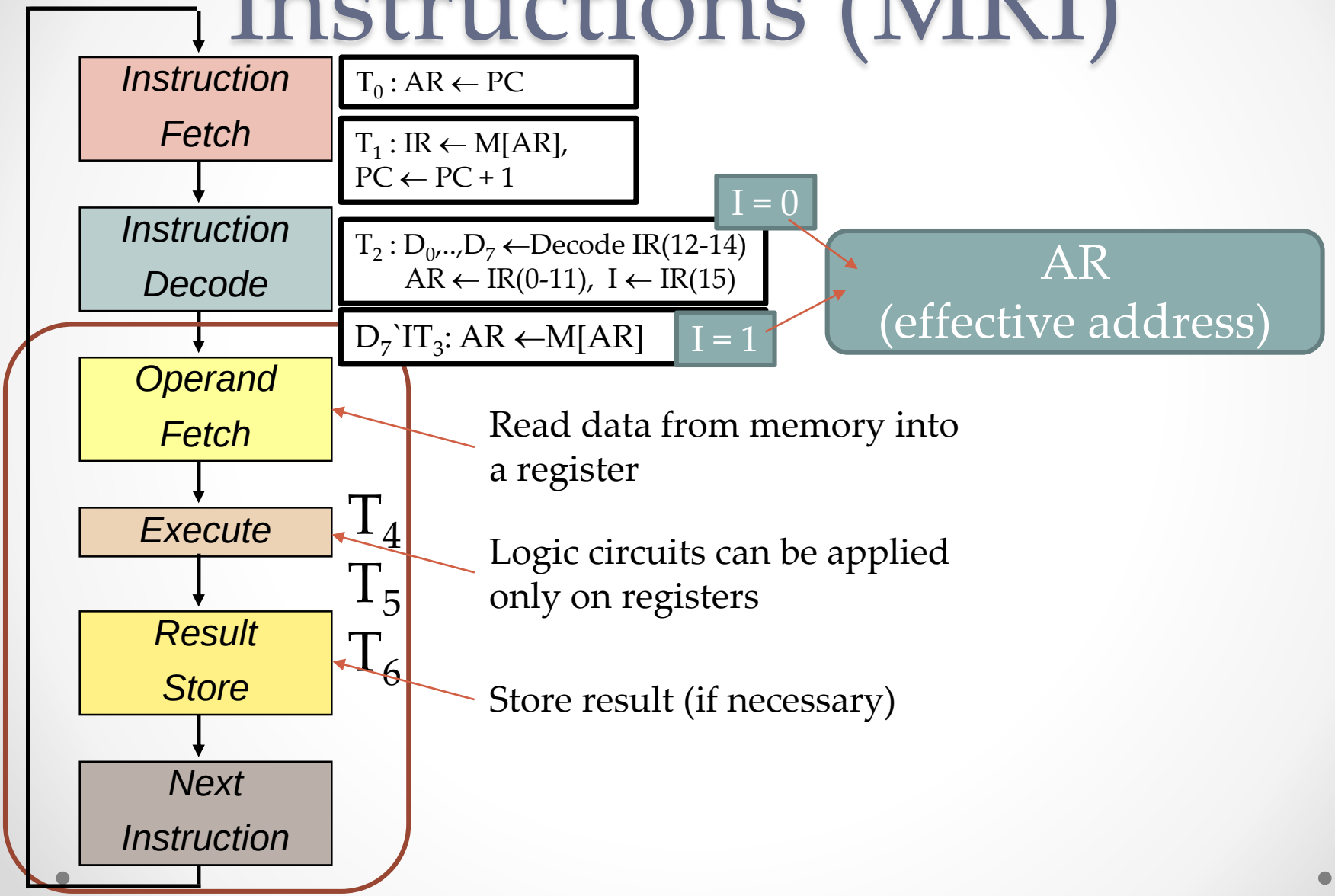
QUIZ6

The content of AC is A937, and E = 1.
Determine the content of the registers below after each instruction Assume the instruction resides in address 21.

		r:	$SC \leftarrow 0$
7800	CLA	rB_{11}:	$AC \leftarrow 0$
7400	CLE	rB_{10}:	$E \leftarrow 0$
7200	CMA	rB_9:	$AC \leftarrow \overline{AC}$
7100	CME	rB_8:	$E \leftarrow \overline{E}$
7080	CIR	rB_7:	$AC \leftarrow$ shr AC, $AC(15) \leftarrow E$, $E \leftarrow AC(0)$
7040	CIL	rB_6:	$AC \leftarrow$ shl AC, $AC(0) \leftarrow E$, $E \leftarrow AC(15)$
7020	INC	rB_5:	$AC \leftarrow AC + 1$
7010	SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$
7008	SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$
7004	SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1$
7002	SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$
7001	HLT	rB_0:	$S \leftarrow 0$ (S is a start–stop flip-flop)

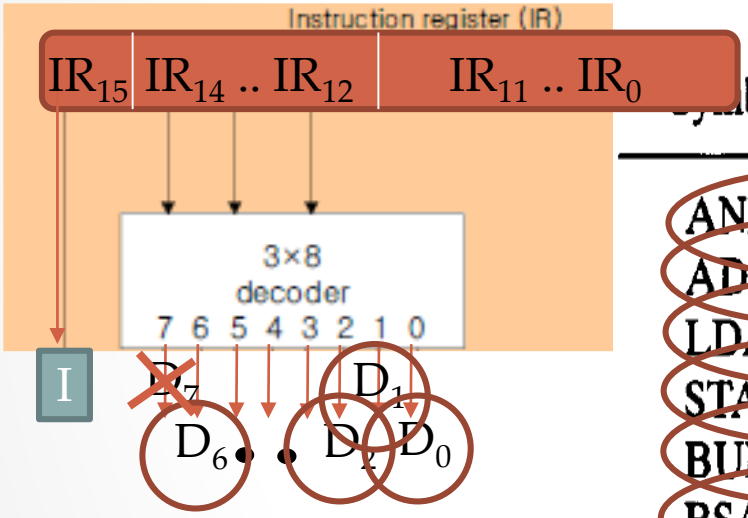
AC	E	PC	AR	IR	instruction
A937	**1**	**21**	**-**	**-**	**initial**
					CLA
					CMA
					CIL
					SNA
					SZE

Memory Reference Instructions (MRI)



Memory Reference Instructions (MRI)

At the end of the T_3 cycle, AR holds the effective Address



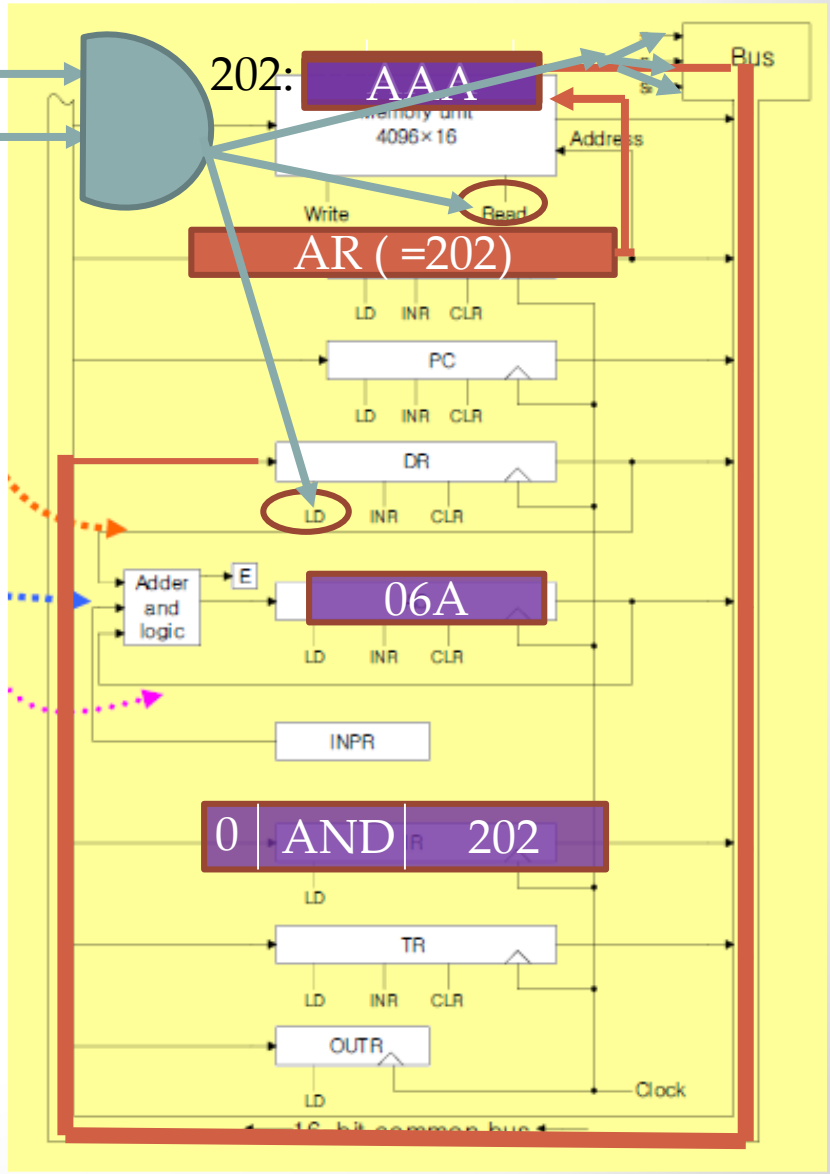
Symbol	Operation decoder	Symbolic description
AND	D_0	$AC \leftarrow AC \wedge M[AR]$
ADD	D_1	$AC \leftarrow AC + M[AR], \quad E \leftarrow C_{out}$
LDA	D_2	$AC \leftarrow M[AR]$
STA	D_3	$M[AR] \leftarrow AC$
BUN	D_4	$PC \leftarrow AR$
BSA	D_5	$M[AR] \leftarrow PC, \quad PC \leftarrow AR + 1$
ISZ	D_6	$M[AR] \leftarrow M[AR] + 1,$ If $M[AR] + 1 = 0$ then $PC \leftarrow PC + 1$

MRI - AND

At the end of the T_3 cycle, AR holds the **effective Address**

$$AND : AC \leftarrow AC \wedge M[AR]$$

$$D_0T_4 : DR \leftarrow M[AR]$$
$$D_0T_5 : AC \leftarrow AC \wedge DR, SC \leftarrow 0$$



MRI - AND

At the end of the T_3 cycle, AR holds the **effective Address**

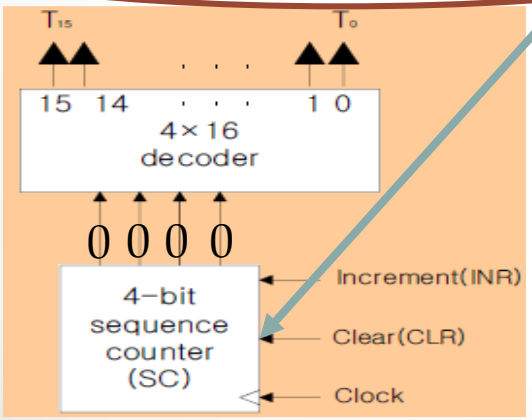
D_0

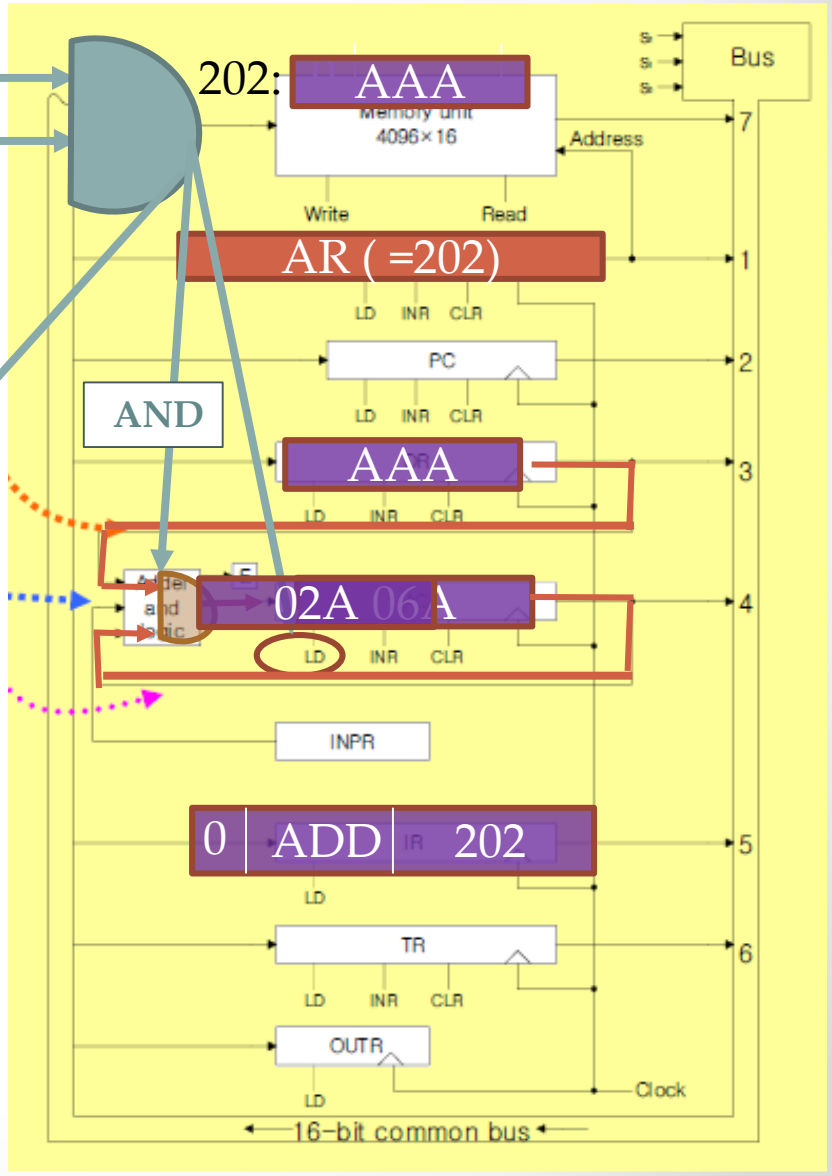
T_5

AND : AC ← AC ∧ M[AR]

D_0T_4 : DR ← M[AR]

D_0T_5 : AC ← AC ∧ DR, SC ← 0



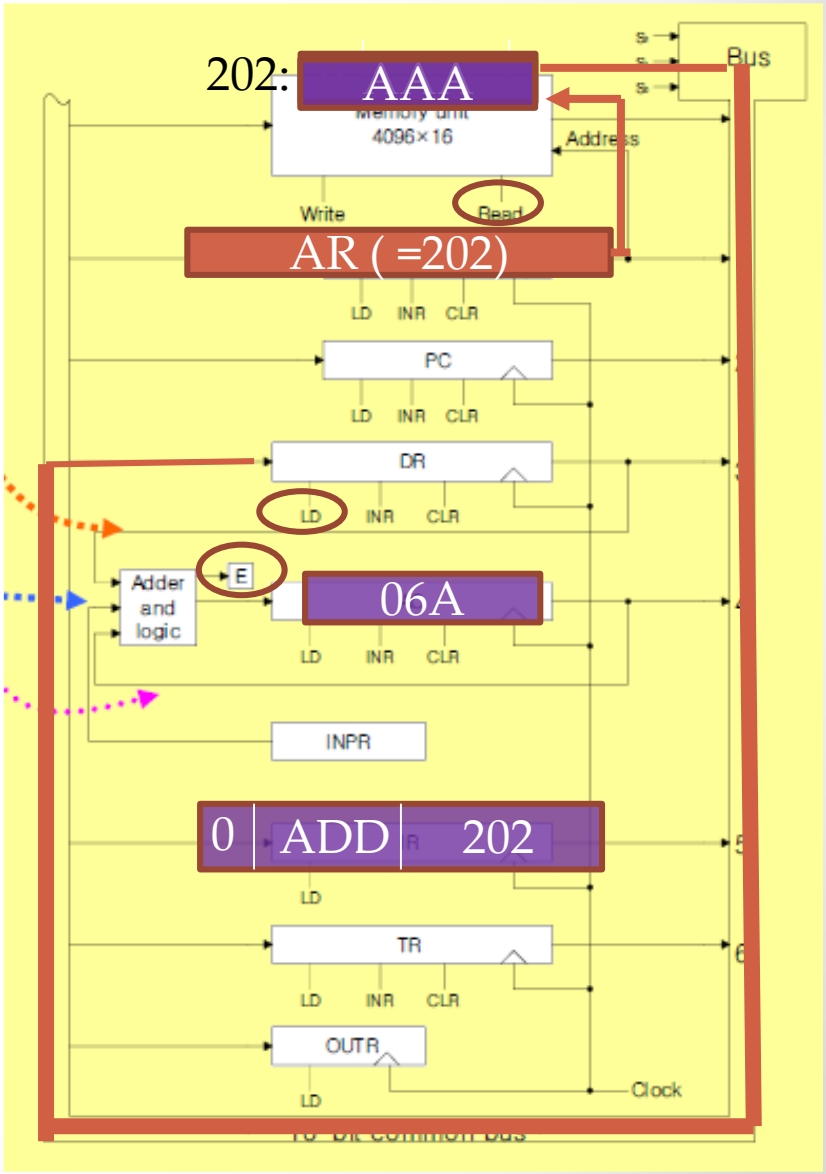


MRI - ADD

At the end of the T_3 cycle, AR holds the **effective Address**

ADD : $AC \leftarrow AC + M[AR]$,
$E \leftarrow C_{out}$

D_1T_4 : $DR \leftarrow M[AR]$
D_1T_5 : $AC \leftarrow AC + DR$,
$E \leftarrow C_{out}$, $SC \leftarrow 0$

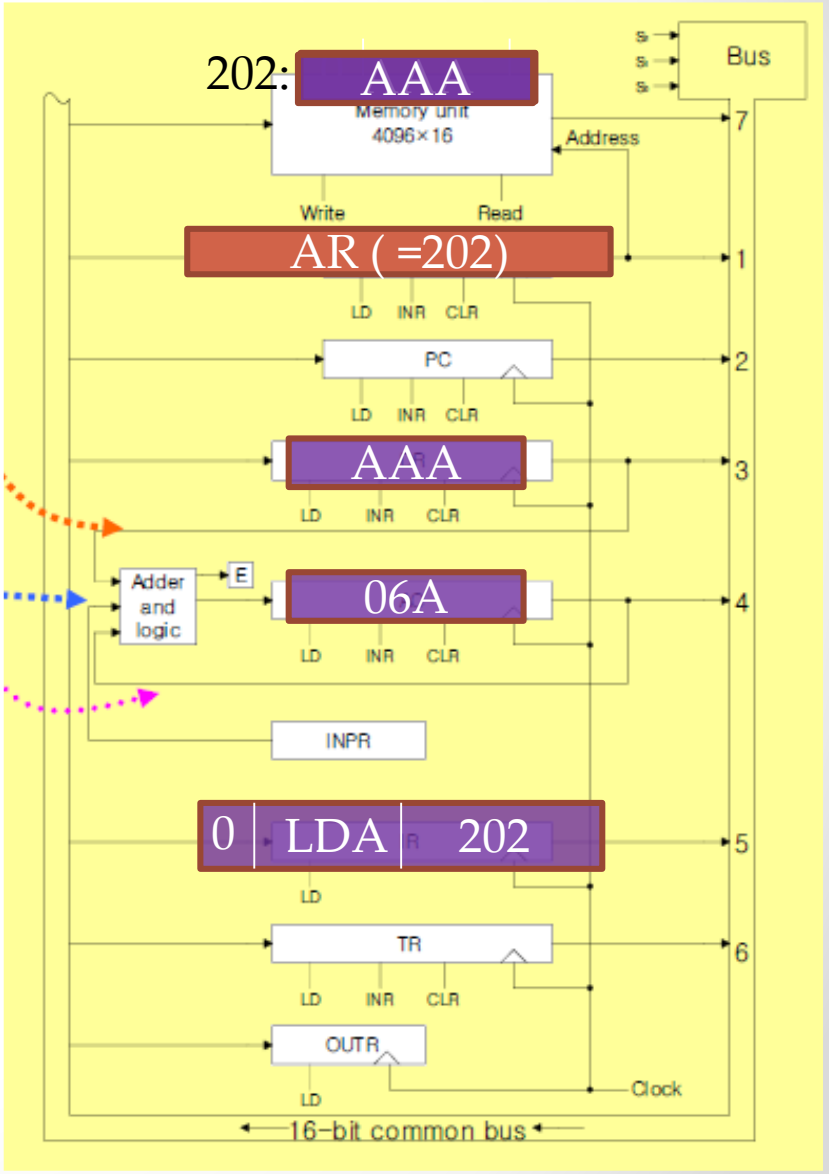


MRI - LDA

At the end of the T_3 cycle, AR holds the **effective Address**

LDA : AC ← M[AR]

D_2T_4 : DR ← M[AR]
D_2T_5 : AC ← DR, SC ← 0



Why not connecting the bus to the inputs of ***AC*** ?

- a delay is encountered in the adder and logic circuit.

  Time(Mem read) + Time(Bus transfer) + Time(A&L) > 1 cycle

- Not connecting the bus to the inputs of AC maintains **one clock cycle per microoperation**.

MRI - STA

At the end of the T_3 cycle, AR holds the **effective Address**

STA : M[AR] ← AC

D_3T_4 : M[AR] ← AC, SC ← 0