Computer Architecture Lec 5a

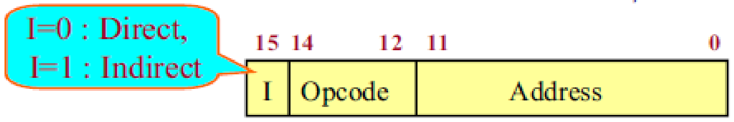
Dr. Esti Stein
(Partly taken from Dr. Alon Schclar slides)

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

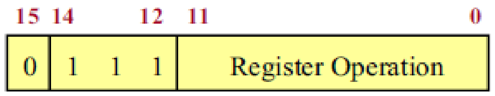
Taken from: **M. Mano/Computer Design and Architecture 3rd Ed.**

Computer Instruction

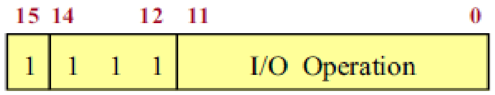
- 5-3 Computer Instruction
  - 3 Instruction Code Formats : ***Fig. 5-5***
    - Memory-reference instruction
      - » Opcode = 000 ~ 110
        - I=0 : 0xxx ~ 6xxx, I=1: 8xxx ~Exxx

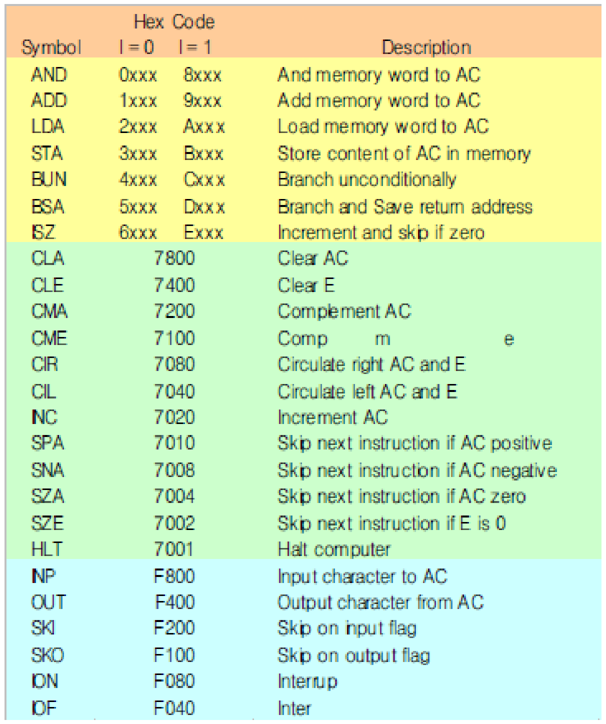


    - Register-reference instruction
      - » 7xxx (7800 ~ 7001) : CLA, CMA, ….



    - Input-Output instruction
      - » Fxxx(F800 ~ F040) : INP, OUT, ION, SKI, ….





Symbol	Hex Code I = 0	I = 1	Description
AND	0xxx	8xxx	And memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load memory word to AC
STA	3xxx	Bxxx	Store content of AC in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and Save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Comp m e
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instruction if AC positive
SNA	7008		Skip next instruction if AC negative
SZA	7004		Skip next instruction if AC zero
SZE	7002		Skip next instruction if E is 0
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrup
IOF	F040		Inter

Instruction Set Completeness

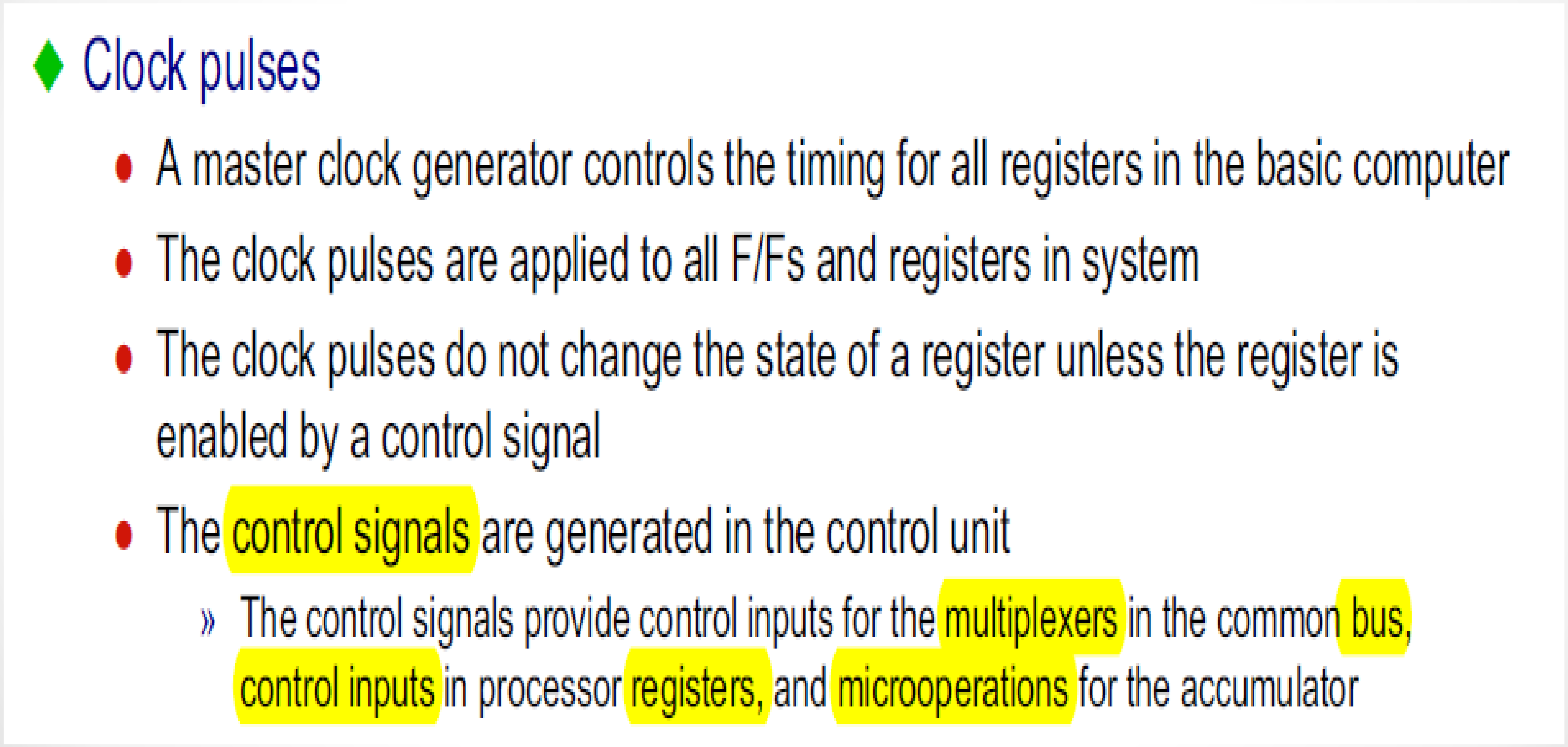
A computer should have a set of instructions so that the user can evaluate any function that is known to be computable.

A complete set must include sufficient number of instructions from the following categories:

1. Arithmetic, logical, and shift - CMA, INC, CIR, AND..
2. Moving information between the registers, and between the registers and memory – LDA, STA
3. Program control, and status check – BUN, ISZ, BSA..
4. Input and output – INP, OUT..

A complete set of instructions, but not efficient.

Timing and Control



♦ Clock pulses

- A master clock generator controls the timing for all registers in the basic computer
- The clock pulses are applied to all F/Fs and registers in system
- The clock pulses do not change the state of a register unless the register is enabled by a control signal
- The control signals are generated in the control unit
  - » The control signals provide control inputs for the multiplexers in the common bus, control inputs in processor registers, and microoperations for the accumulator

Two (major) Types of Control Organization

Hardwired Control (This chapter)

- » The control logic is implemented with gates, F/Fs, decoders, and other digital circuits
- » + Fast operation, - Wiring change(if the design has to be modified)

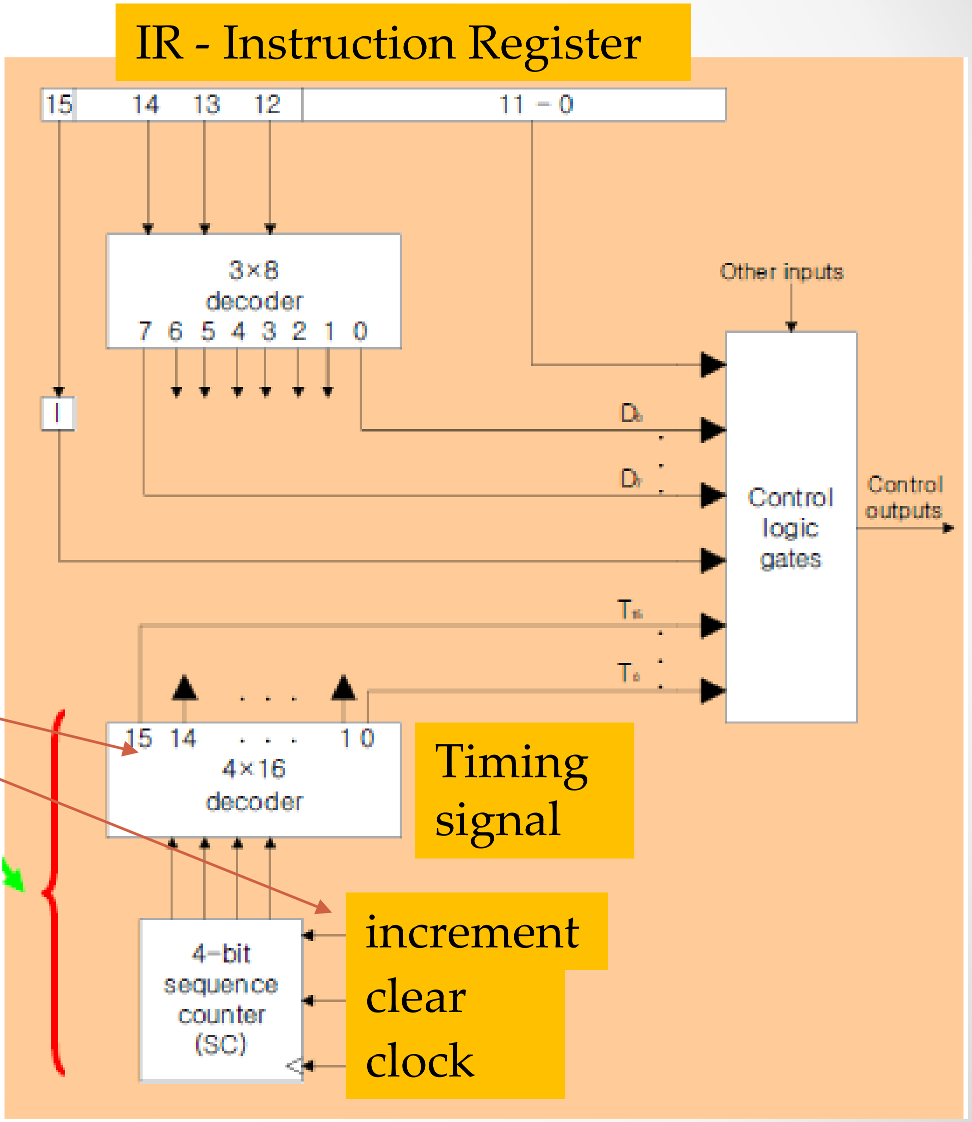
Microprogrammed Control (chapter 7)

- » The control information is stored in a control memory, and the control memory is programmed to initiate the required sequence of microoperations
- » + Any required change can be done by updating the microprogram in control memory,
  - Slow operation

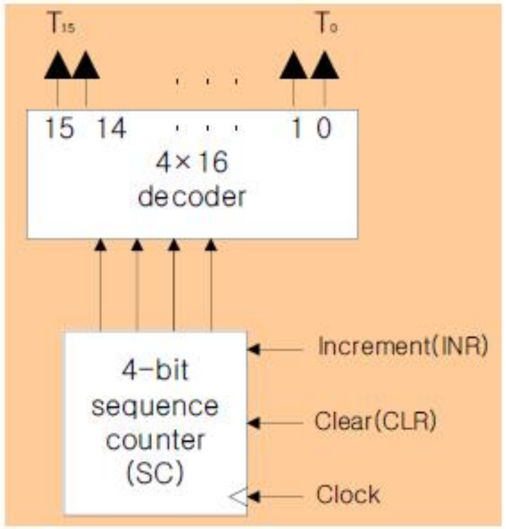
The Control Unit

The Sequence Counter (SC) is incremented synchronously:
$T_0, T_1, T_2, T_3, ..$

The SC is cleared when in RTL we write symbolically:
$SC \leftarrow 0$



The Control Unit



$$D_3T_4 : SC \leftarrow 0$$

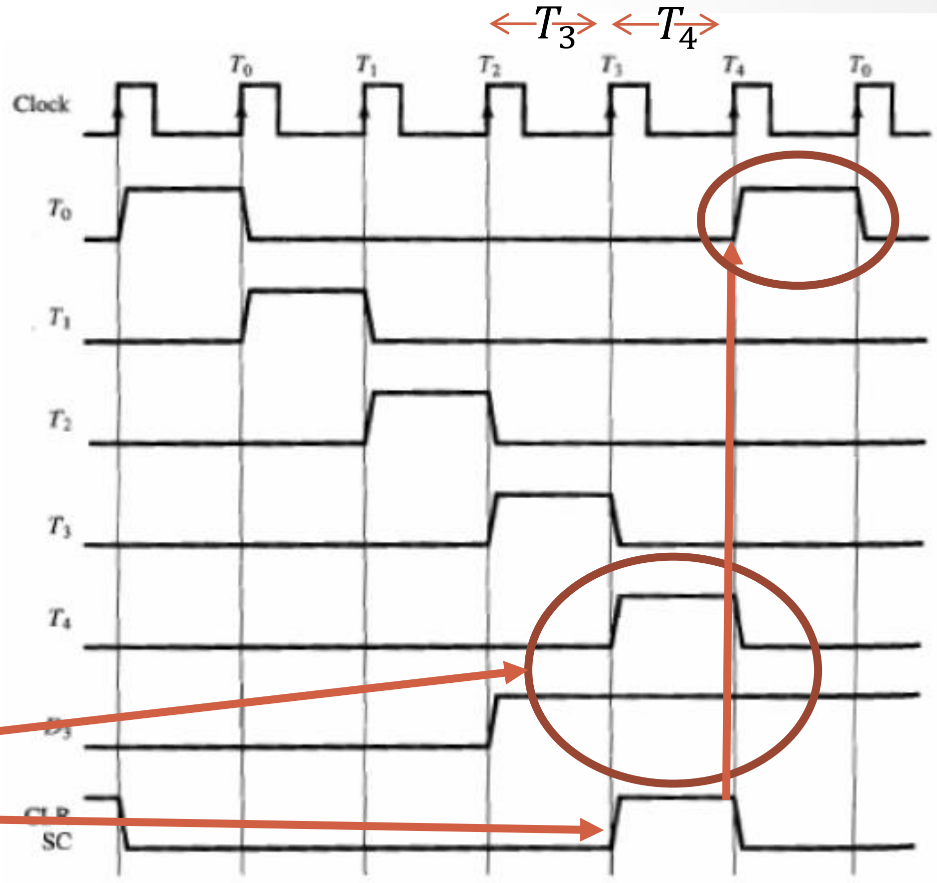


Figure 5-7 Example of control timing signals.

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

Clock and Memory

- The memory read/write cycle is initiated with the rising edge of a timing signal.
- It is assumed that a memory cycle time is less than the clock cycle time.
- The memory R/W cycle is complete by the time the next clock goes through its positive cycle.
- This assumption is not valid in most of computers.

Clock and Timing Signals

Example:

$$T_0 : AR \leftarrow PC$$

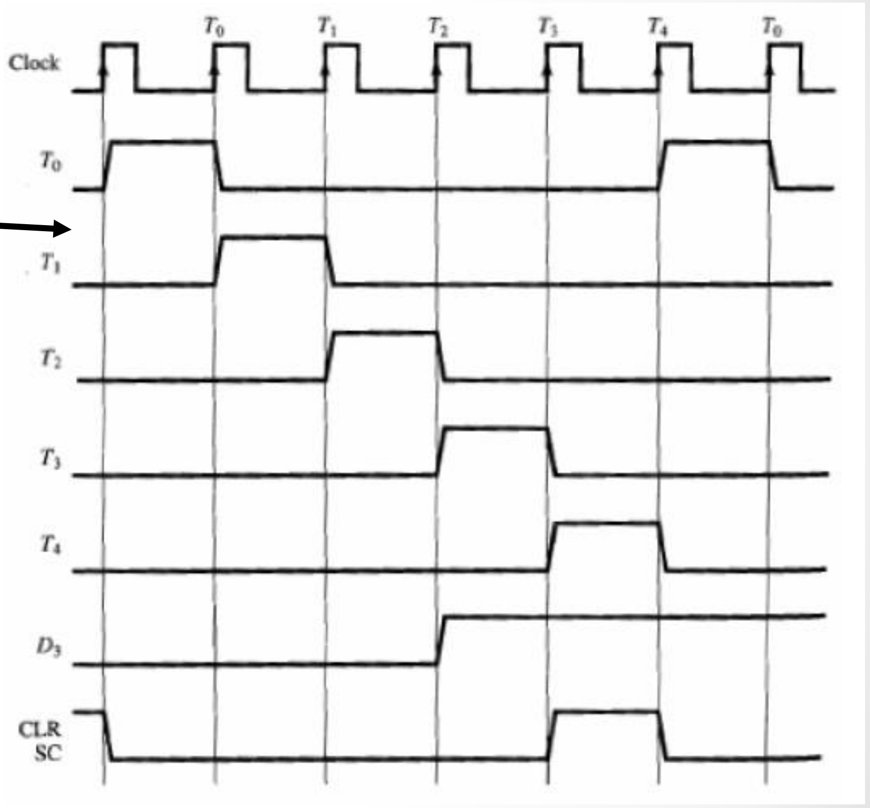
1. T_0 is active during the entire clock cycle interval.
2. During this time the content of the PC is placed onto the bus, and the LD of AR is enabled.
3. The actual transfer occurs when the clock goes through a positive transition (end of cycle).
4. On positive transition SC goes from 0000 to 0001.
5. T_1 is active and T_0 is inactive.

QUIZ4

Draw a timing diagram similar to this one, assuming that SC is cleared to 0 at time T_3 if control signal C_7 is active:

$C_7T_{3:}$ SC $\leftarrow 0$

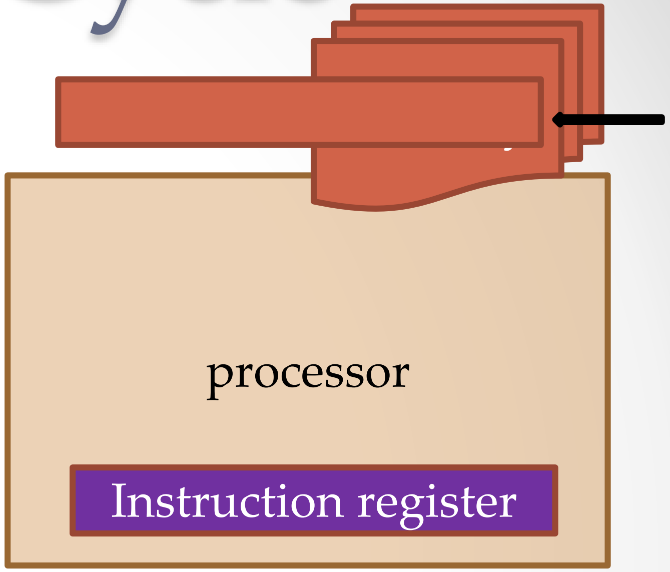
C_7 is activated with the positive clock transition associated with $T_{1.}$



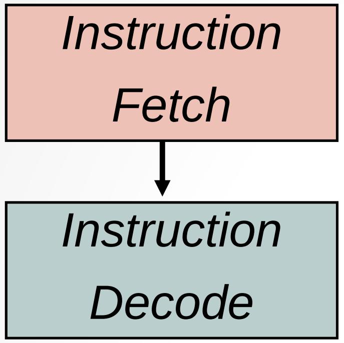
Instruction Cycle

Instruction Fetch

Obtain instruction from program storage in memory

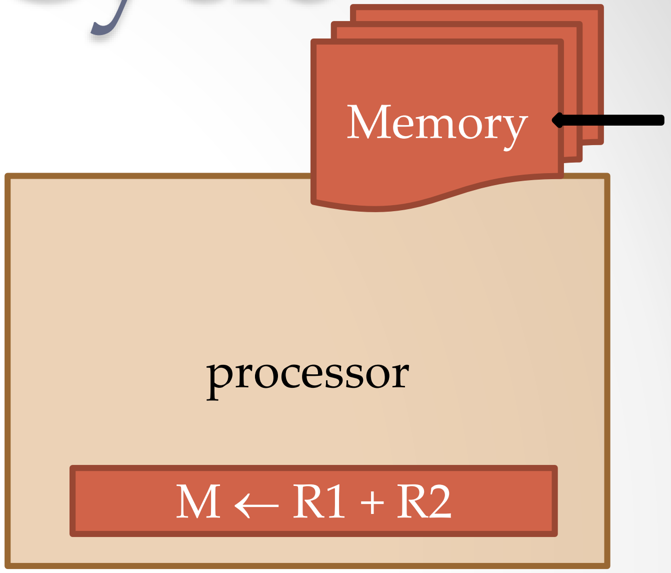


Instruction Cycle



Obtain instruction from program storage in memory

Determine required actions and instruction size



Instruction Cycle

Instruction Fetch

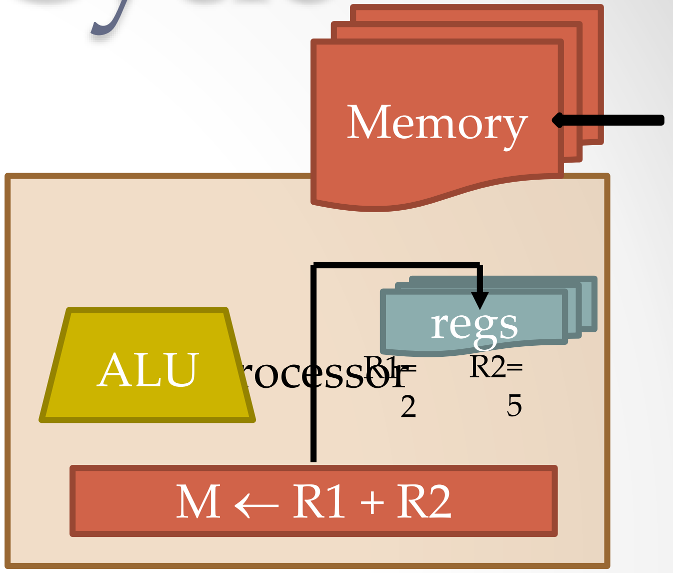
Obtain instruction from program storage in memory

Instruction Decode

Determine required actions and instruction size

Operand Fetch

Locate and obtain operand data



Instruction Cycle

Instruction Fetch — Obtain instruction from program storage in memory

↓

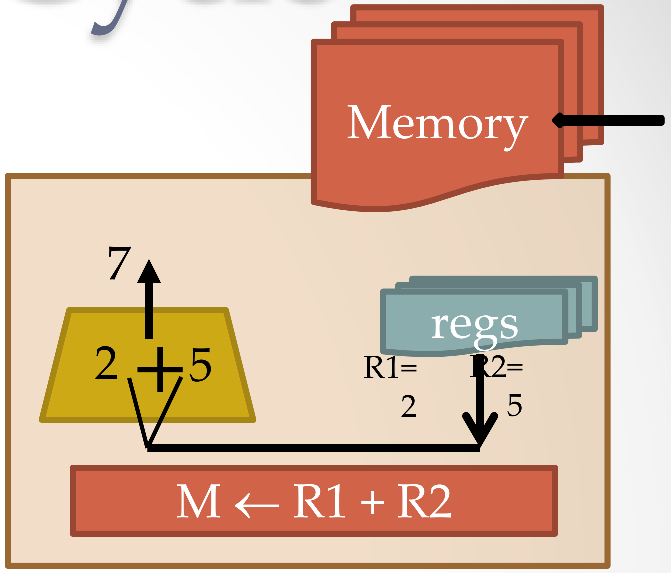
Instruction Decode — Determine required actions and instruction size

↓

Operand Fetch — Locate and obtain operand data

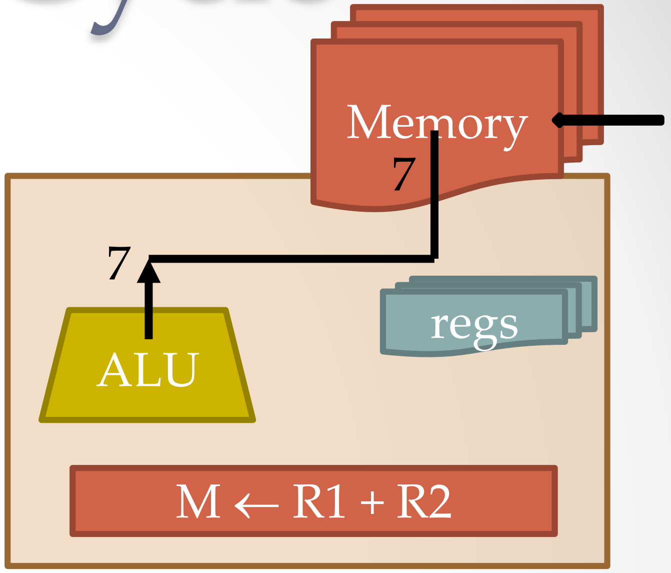
↓

Execute — Compute result value or status

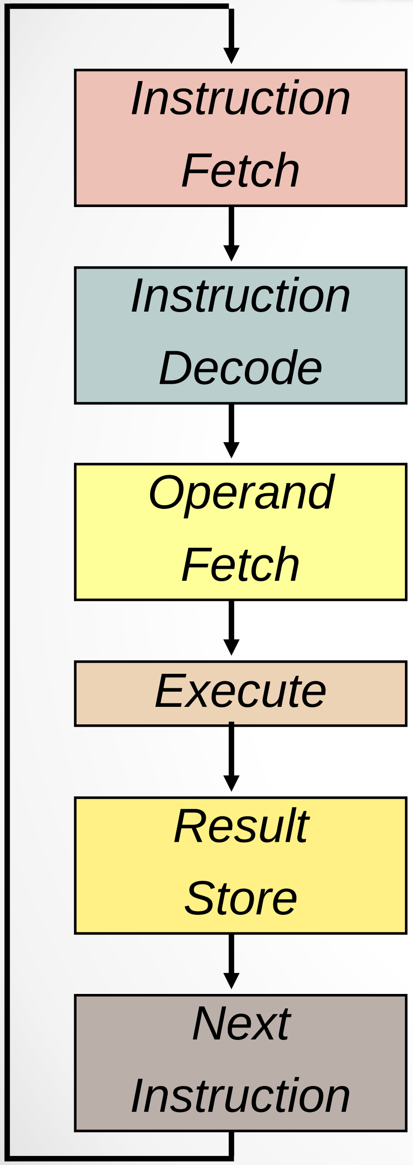


Instruction Cycle

Instruction Fetch	Obtain instruction from program storage in memory
Instruction Decode	Determine required actions and instruction size
Operand Fetch	Locate and obtain operand data
Execute	Compute result value or status
Result Store	Deposit results in storage



Instruction Cycle



Obtain instruction from program storage in memory

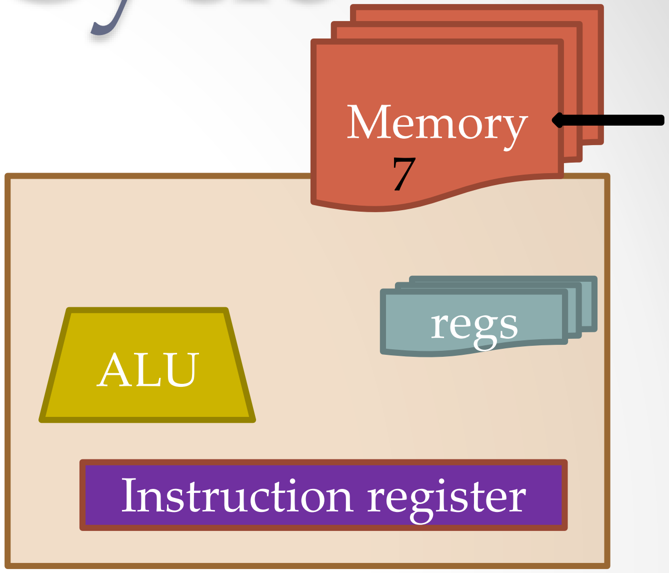
Determine required actions and instruction size

Locate and obtain operand data

Compute result value or status

Deposit results in storage

Determine next instruction (not the next in case of **branch**)

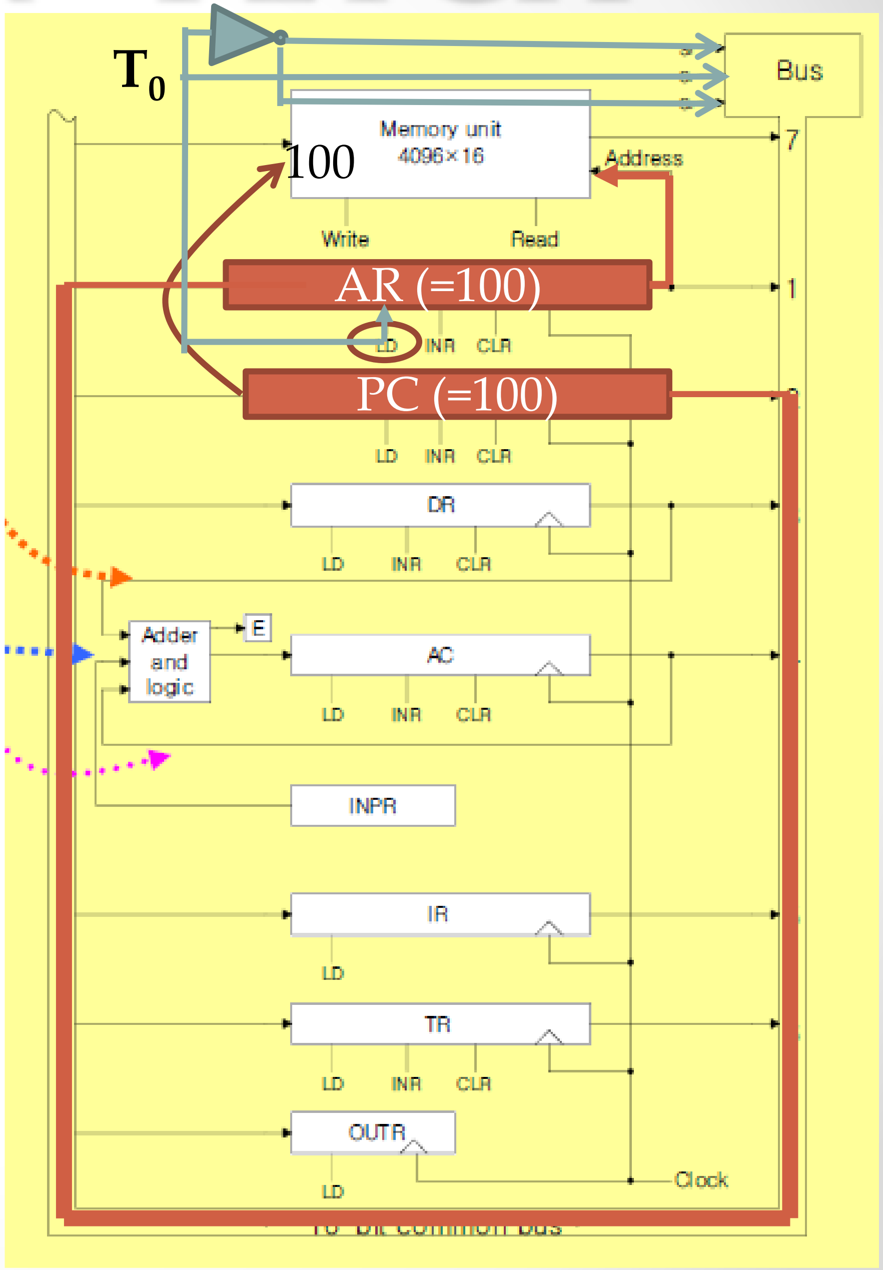


Instruction - FETCH

Instruction Fetch

Obtain instruction from program storage in memory

$$\boxed{T_0 : AR \leftarrow PC}$$



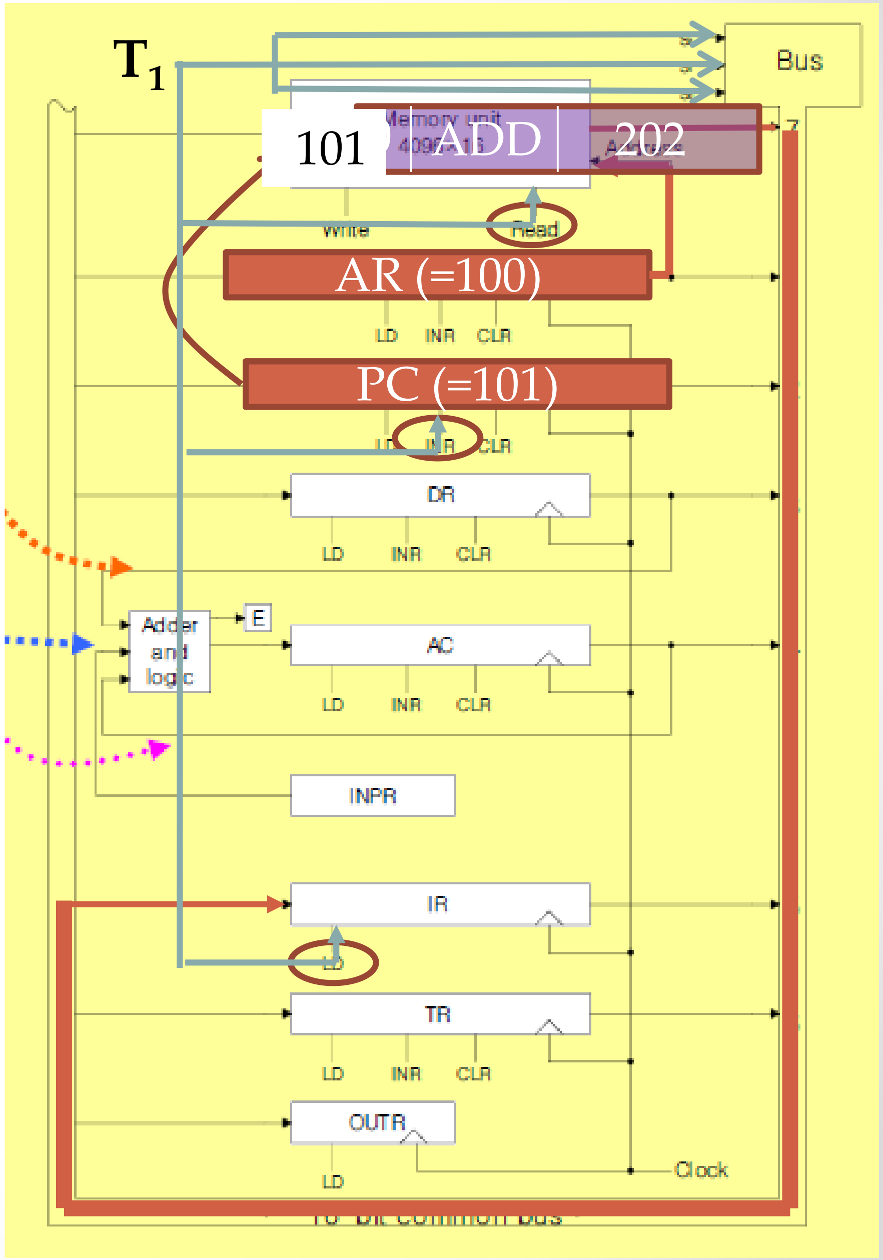
Instruction - FETCH

Instruction Fetch

Obtain instruction from program storage in memory

$T_0 : AR \leftarrow PC$

$T_1 : IR \leftarrow M[AR], \quad PC \leftarrow PC + 1$



Instruction - DECODE

Instruction Fetch

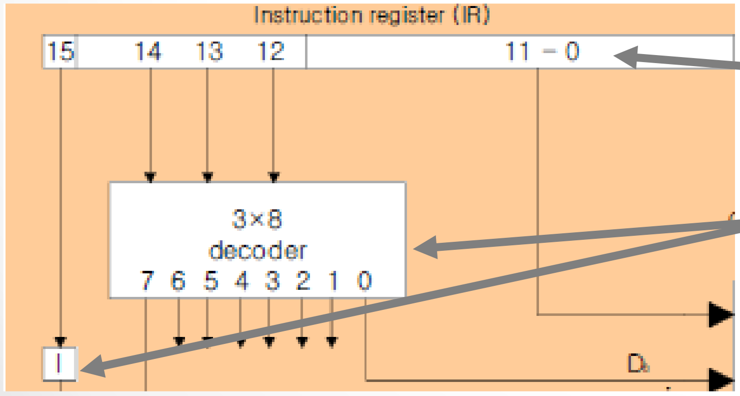
Instruction Decode

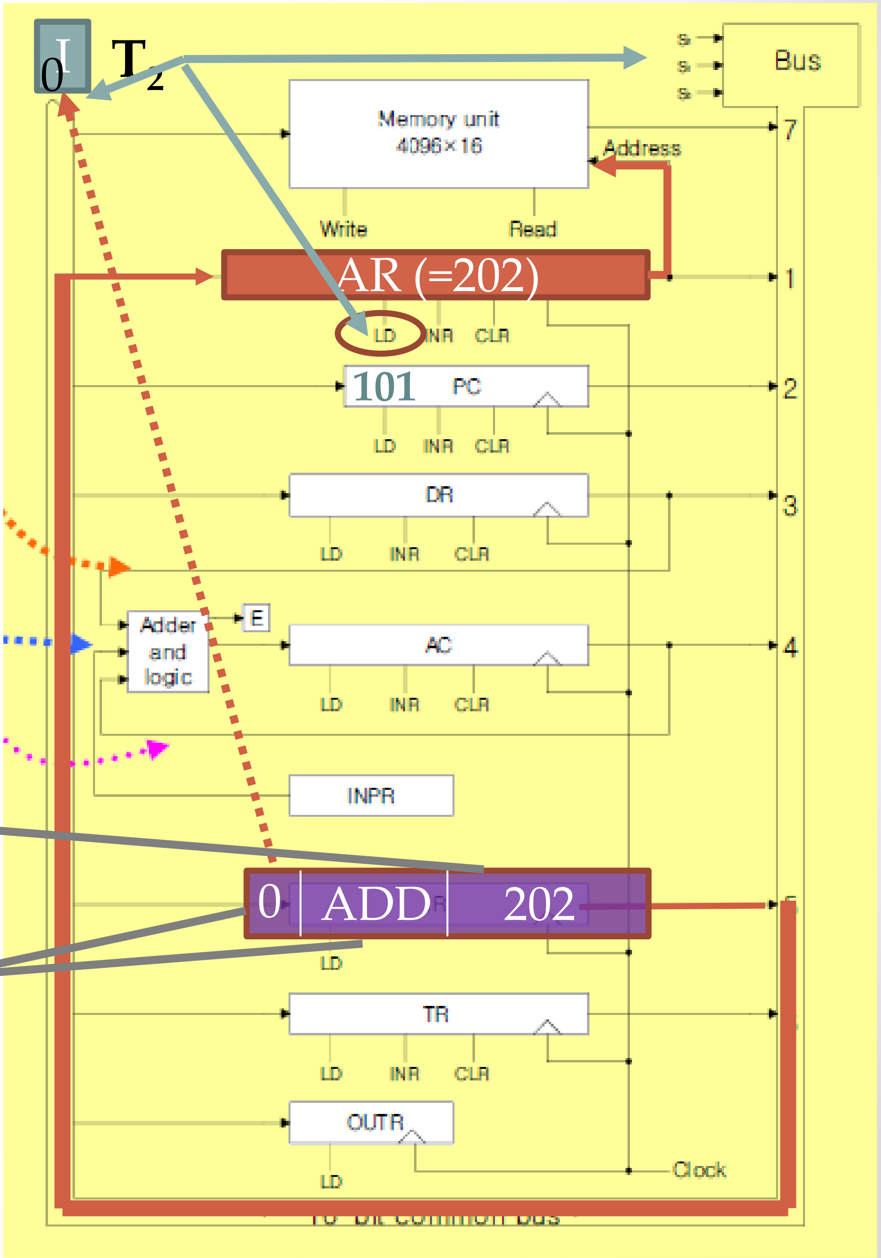
$T_0 : AR \leftarrow PC$

$T_1 : IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$

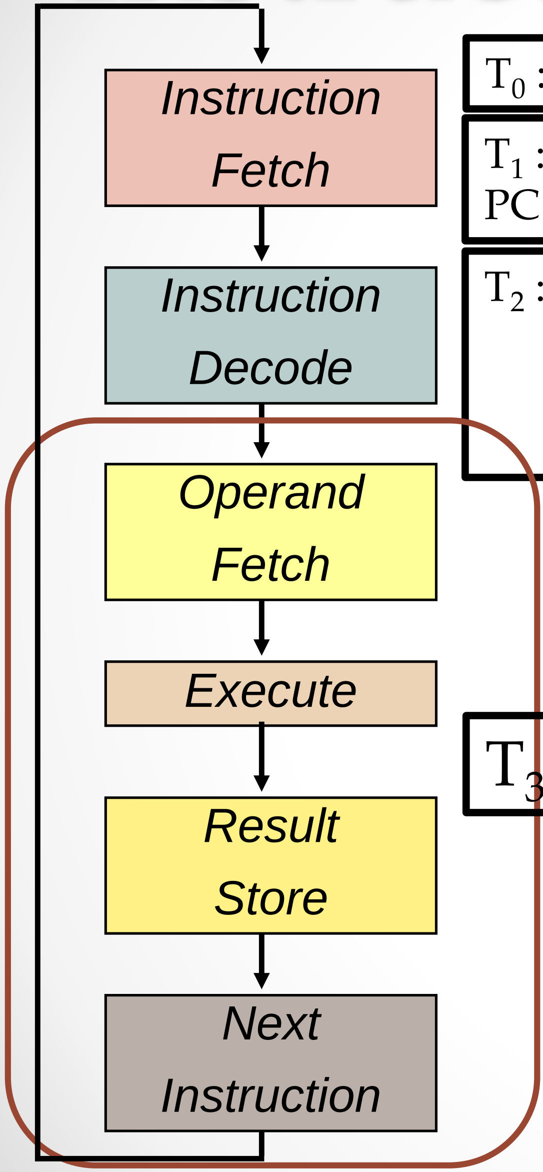
Determine required actions and instruction size

$T_2 : D_0 ,.., D_7 \leftarrow$ Decode IR(12-14)
$AR \leftarrow$ IR(0-11),
$I \leftarrow$ IR(15)





Instruction - EXECUTION

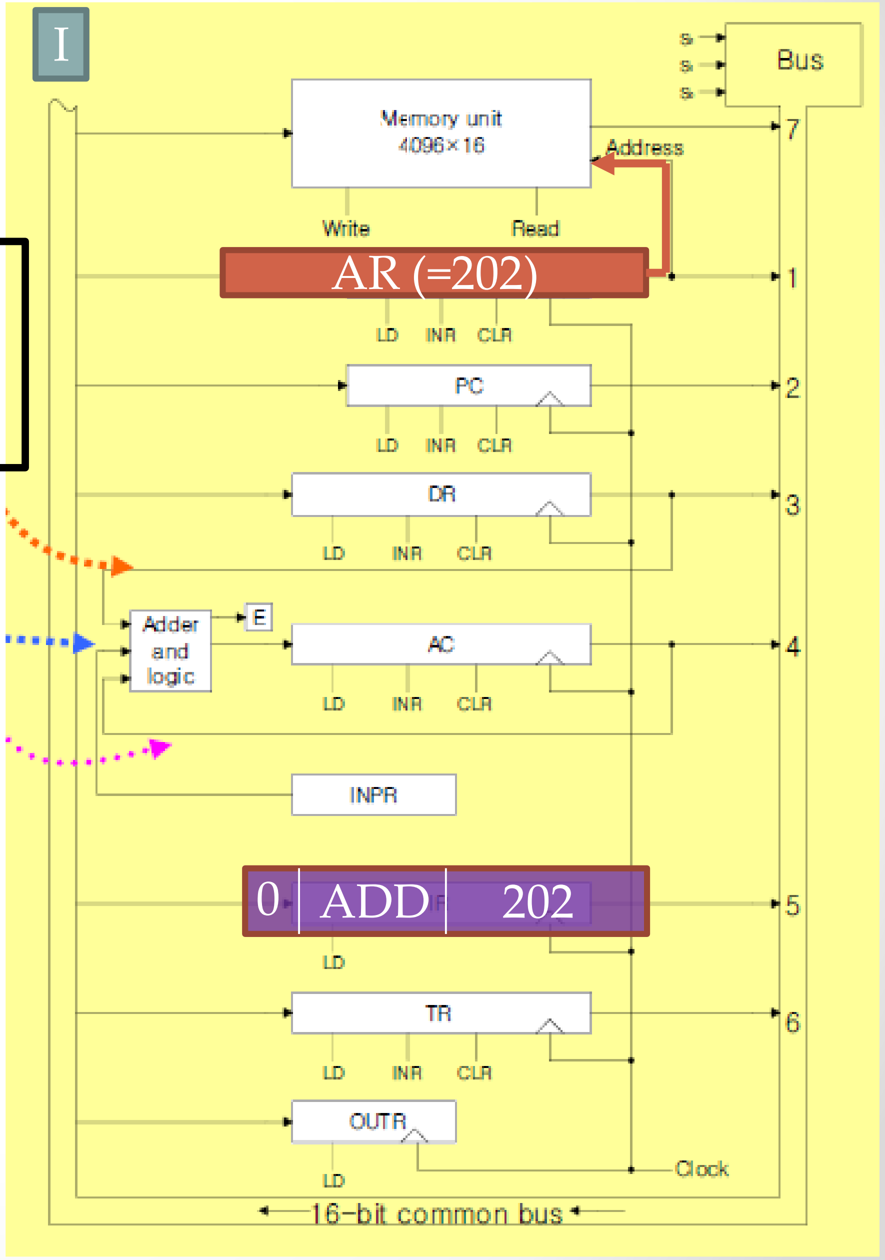


$T_0 : AR \leftarrow PC$

$T_1 : IR \leftarrow M[AR],$
$PC \leftarrow PC + 1$

$T_2 : D_0 ,.., D_7 \leftarrow$ Decode IR(12-14)
$AR \leftarrow IR(0\text{-}11),$
$I \leftarrow IR(15)$

$T_3\ T_4\ T_5\ T_6$

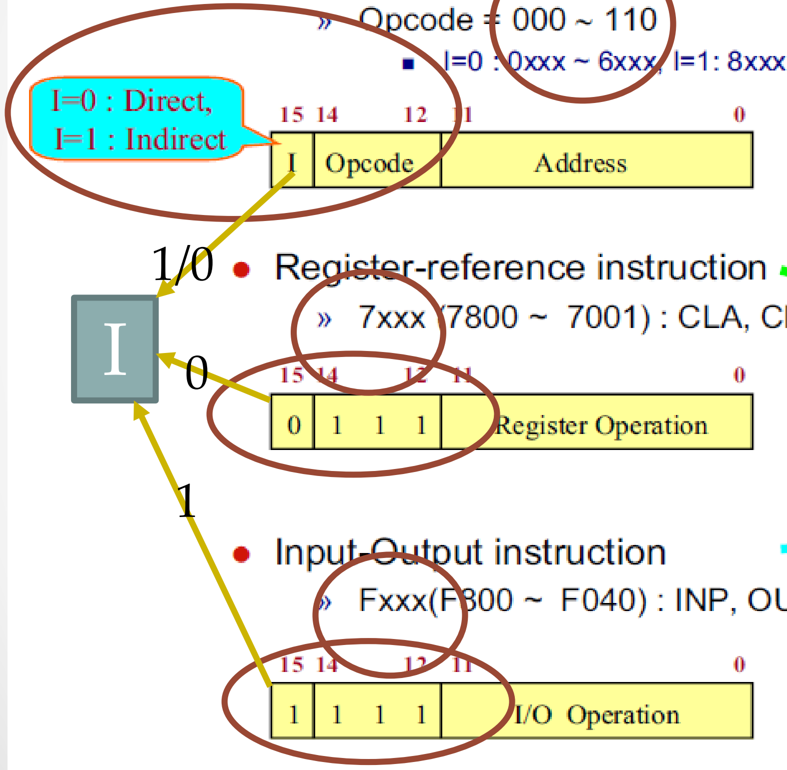


The Instruction Format

■ 5-3 Computer Instruction

◆ 3 Instruction Code Formats : *Fig. 5-5*

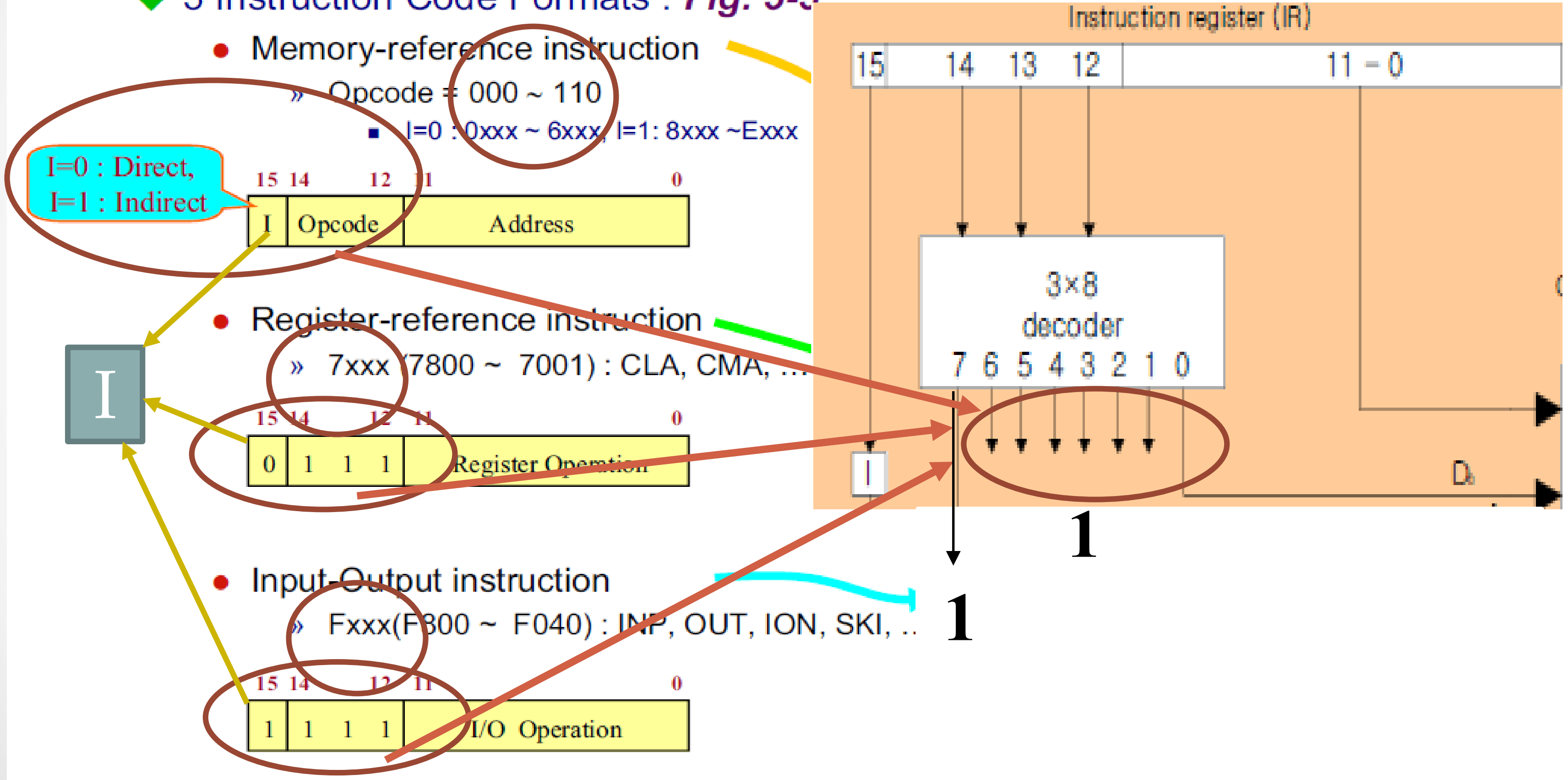
- Memory-reference instruction
  - » Opcode = 000 ~ 110
    - ■ I=0 : 0xxx ~ 6xxx, I=1: 8xxx ~Exxx
- Register-reference instruction
  - » 7xxx (7800 ~ 7001) : CLA, CMA, ….
- Input-Output instruction
  - » Fxxx(F800 ~ F040) : INP, OUT, ION, SKI, ….



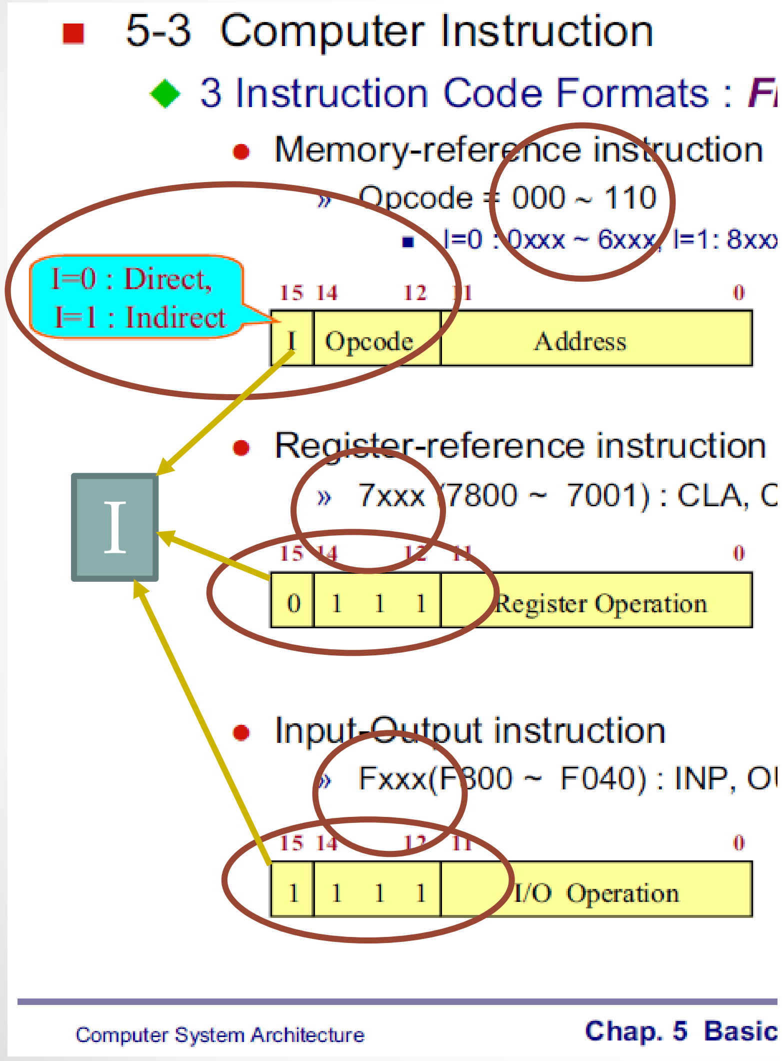
Symbol	Hex Code I = 0	Hex Code I = 1	Description
AND	0xxx	8xxx	And memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load memory word to AC
STA	3xxx	Bxxx	Store content of AC in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and Save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Comp m e
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instruction if AC positive
SNA	7008		Skip next instruction if AC negative
SZA	7004		Skip next instruction if AC zero
SZE	7002		Skip next instruction if E is 0
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrup
IOF	F040		Inter

The Instruction Format

- 5-3 Computer Instruction
  - 3 Instruction Code Formats : ***Fig. 5-5***
    - Memory-reference instruction
      - » Opcode = 000 ~ 110
        - I=0 : 0xxx ~ 6xxx, I=1: 8xxx ~Exxx
    - Register-reference instruction
      - » 7xxx (7800 ~ 7001) : CLA, CMA, ...
    - Input-Output instruction
      - » Fxxx(F800 ~ F040) : INP, OUT, ION, SKI, ..



Determine Instruction Type



■ 5-3 Computer Instruction

◆ 3 Instruction Code Formats : Fi

- 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

- 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

I

Computer System Architecture — Chap. 5 Basic

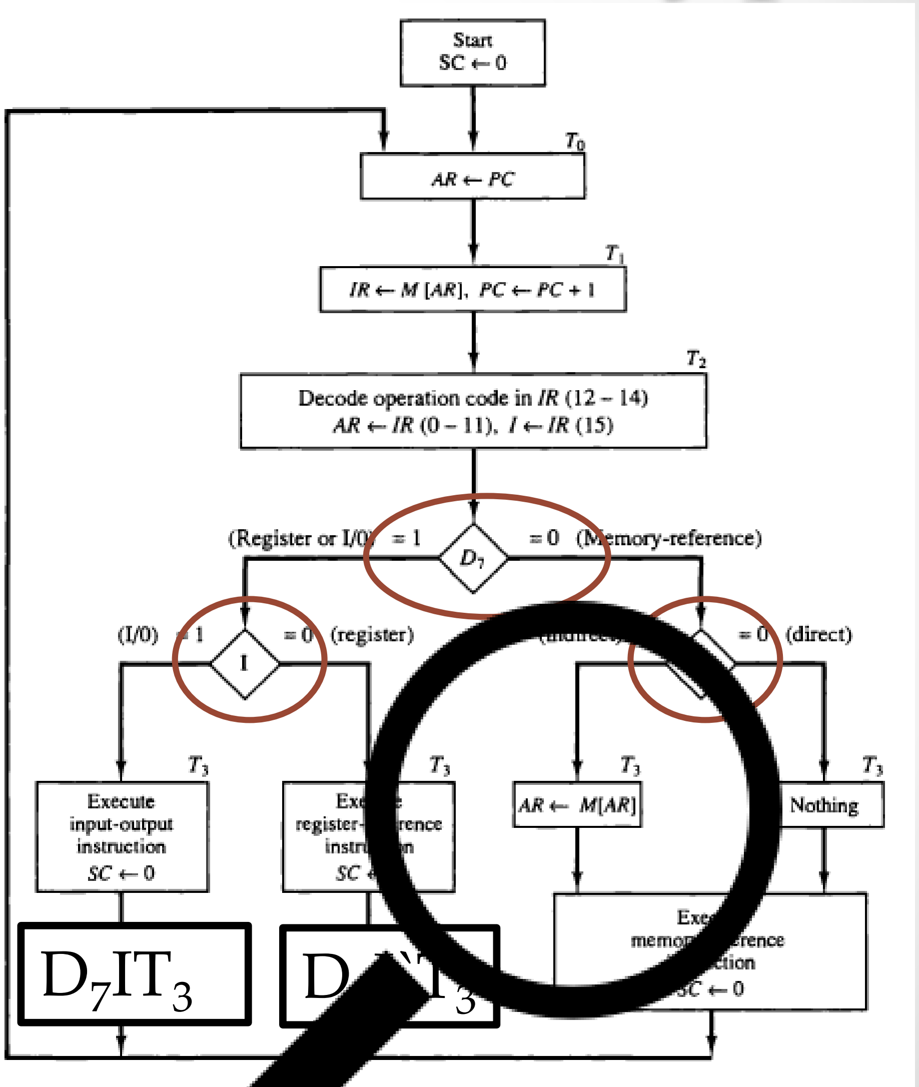
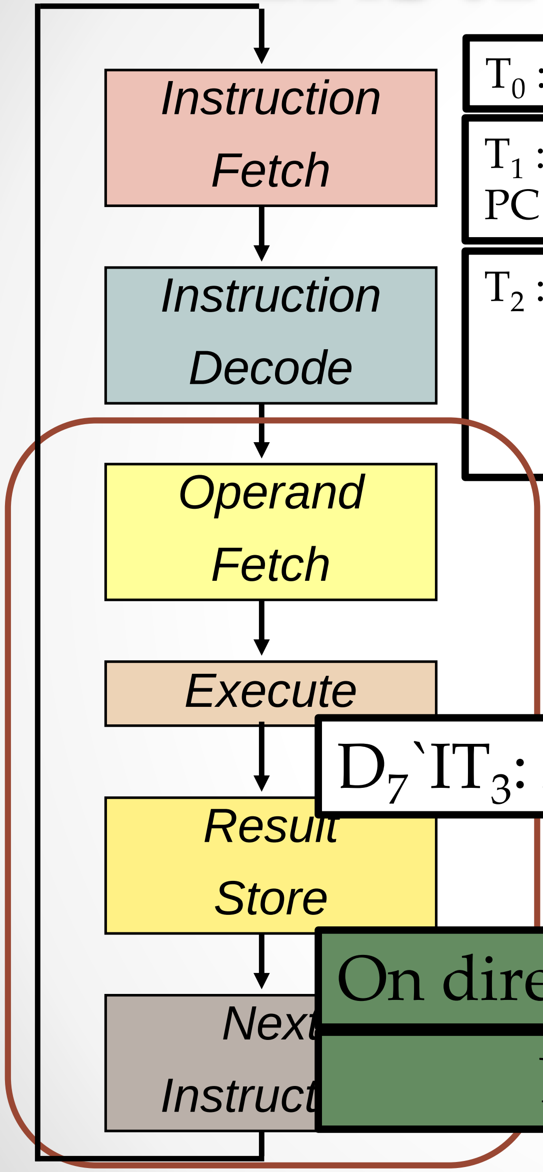


Figure 5-9 ...rt for instruction cycle (initial configuration).

Instruction - Indirect



$T_0 : AR \leftarrow PC$

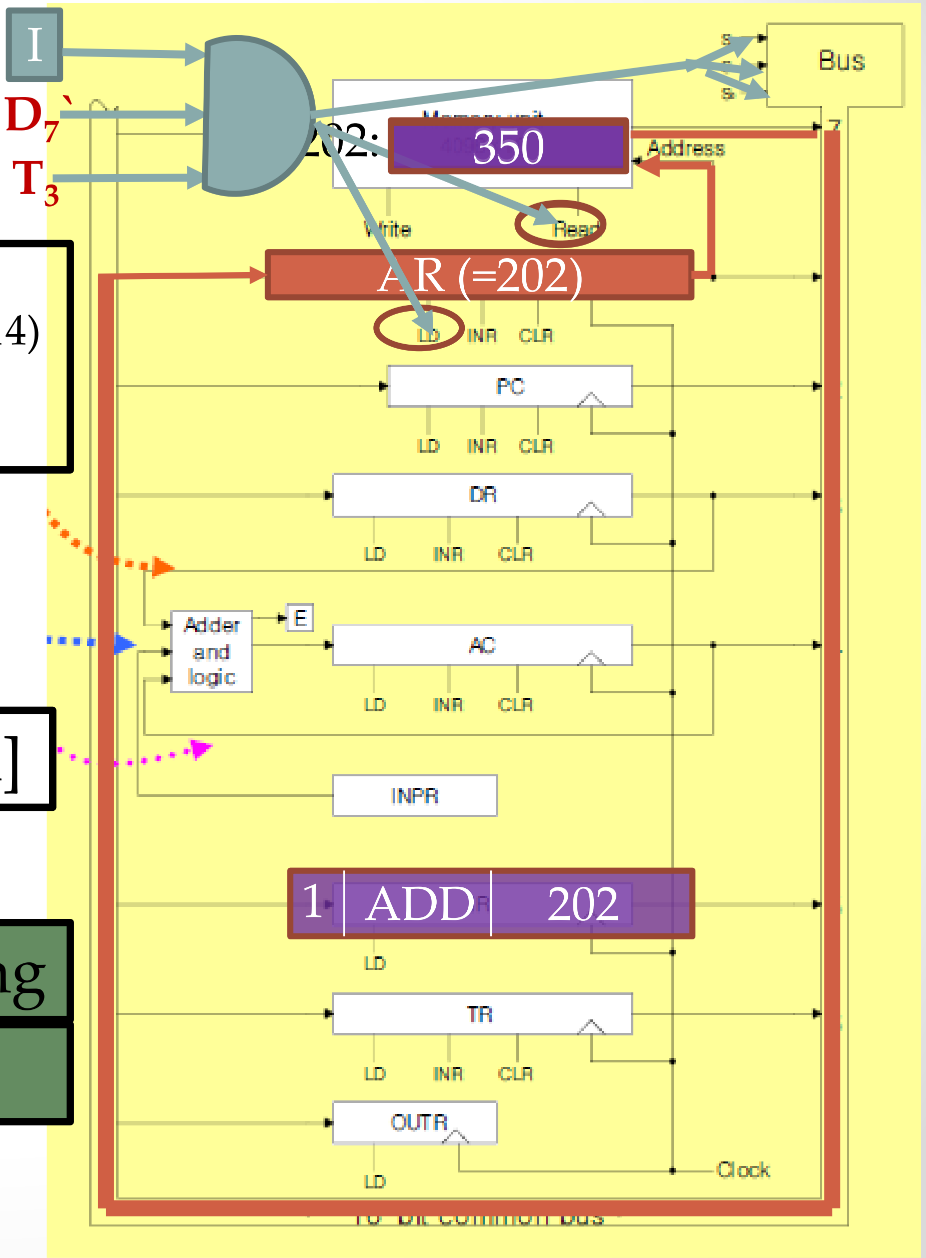
$T_1 : IR \leftarrow M[AR],$
$PC \leftarrow PC + 1$

$T_2 : D_0 ,.., D_7 \leftarrow$ Decode IR(12-14)
$AR \leftarrow IR(0\text{-}11),$
$I \leftarrow IR(15)$

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

On direct do nothing

$D_7`I`T_3$



QUIZ5

$T_0 : AR \leftarrow PC$

$T_1 : IR \leftarrow M[AR],$
$PC \leftarrow PC + 1$

$T_2 : D_0 ,.., D_7 \leftarrow$ Decode IR(12-14)
$AR \leftarrow$ IR(0-11),
$I \leftarrow$ IR(15)

The following words reside in some memory addresses

ADDR (hexa)	DATA (binary)
100	1001 0001 0100 0000
101	0111 1000 0000 0000
…	…
120	0000 0001 0000 0001
…	…
13F	1111 1111 0000 0000
140	0000 0001 0010 0000

Let PC = 100H

And the next clock cycle is T_0

What will be the values of the following registers at the end of cycle T_3 ?

PC, AR, IR in hexadecimal?