

# SAP 1

- SAP-1 Characteristic
- SAP-1 Architecture
- SAP-1 Components
- SAP-1 Instruction Set & Cycle

# SAP Overview

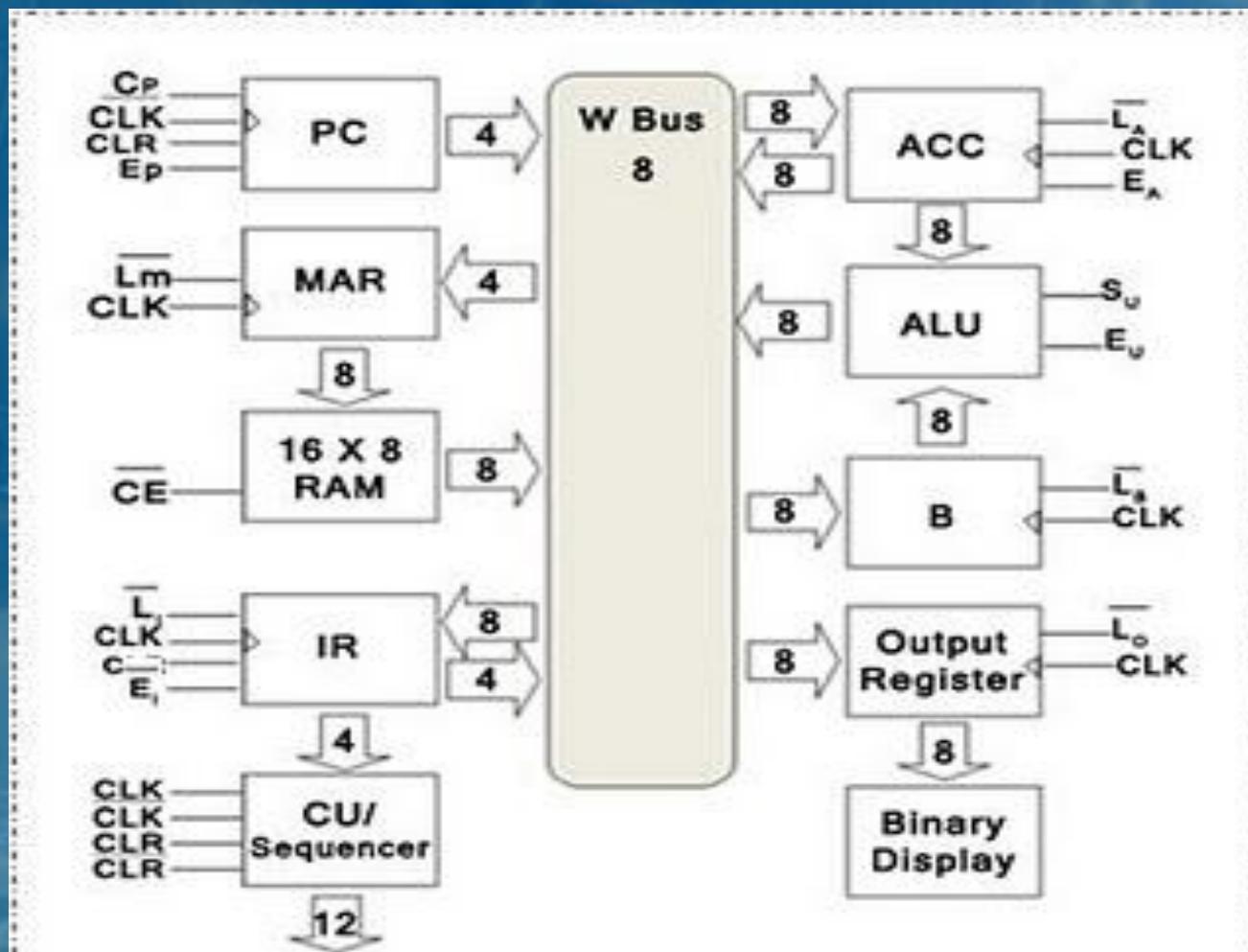
- **Simple As Possible**
- Describe the simplest computer workings
- 1<sup>st</sup> step evolution to advanced computer development
- Bus organized Computer

# SAP-1 Characteristics

## Hardwire Architecture

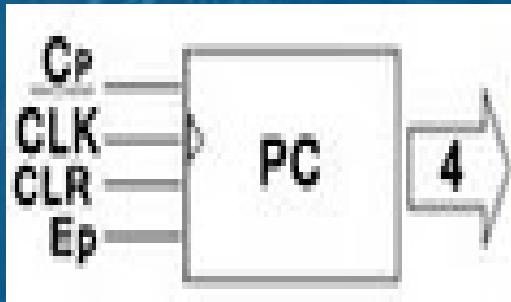
- Two 8-bit general registers (A, B)
- One 8-bit output register
- 8-bit ALU (addition, subtraction)
- 4-bit instructions and 4-bits operands  
(cccc 0ooo)
  - cccc = OP CODE
  - 0ooo = OPERAND
- 16x8 address RAM for mixed program and data
- 12 control signals

# SAP-1 Architecture



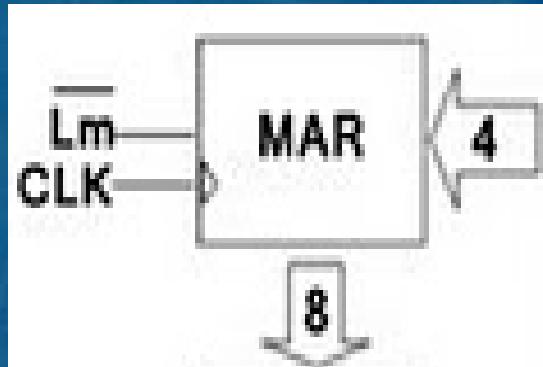
$C_P E_P L' M CE' L'_I E'_I L'_A E_A S_U E_U L'_B L'_O$

# Program Counter



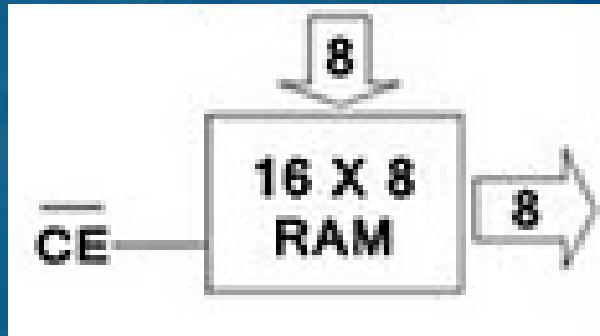
- OUTPUT: 0000 to 1111 (0 to F)
- CLK: Clock cycle
- CLR: reset output to 0000
- Cp:  $(PC) \leftarrow (PC)+1$
- Ep:  $(PC) \rightarrow \text{bus } W$

# Memory Address Register (MAR)



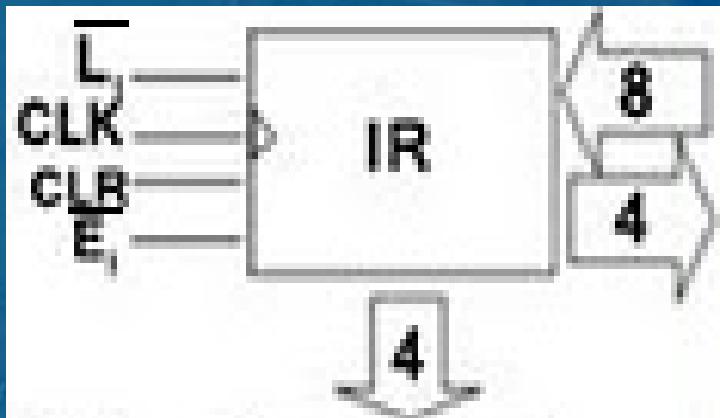
- INPUT: 4 bits
- OUTPUT: 8 bits
- CLK: Clock cycle
- Lm:
  - 0 : (MAR)  $\leftarrow$  bus W

# Random Access Memory (RAM)



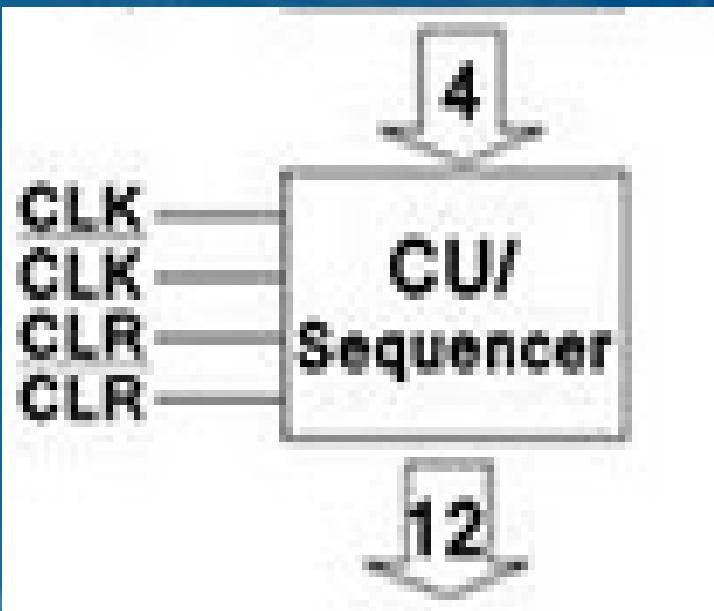
- Store instruction & data
- INPUT: 8 bits from MAR
- OUTPUT: 8 bits to bus W
- CE:
  - 0 : (RAM) → bus W

# Instruction Register (IR)



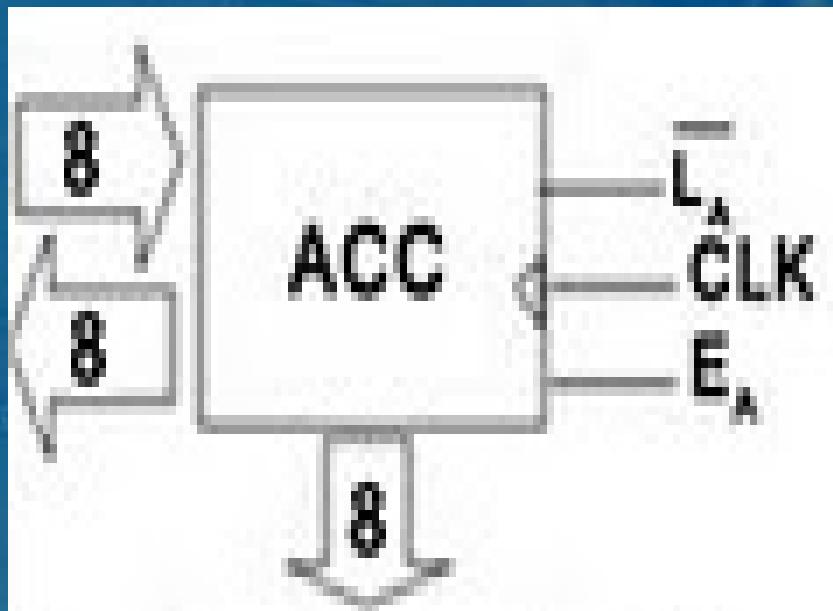
- Read an instruction from RAM
- INPUT: 8 bits from RAM
- OUTPUT:
  - 4 bits to bus W
  - 4 bits to CU
- $L_I$ :
  - 0 : (IR)  $\leftarrow$  8-bit input
- $E_I$ :
  - 0 : (IR) 4-bit  $\rightarrow$  bus W

# CU/Sequencer



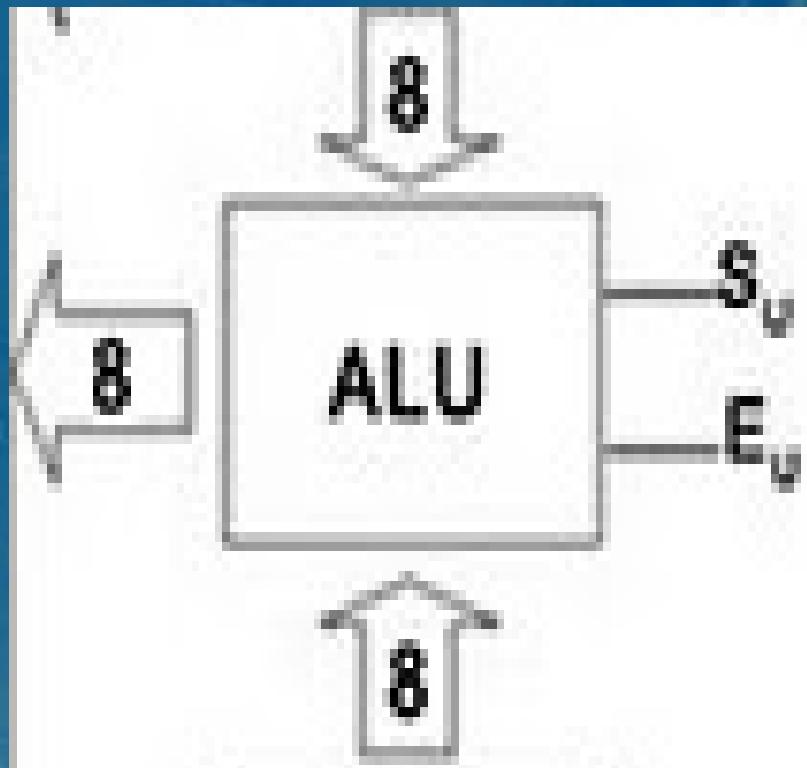
- Control all execution flow
- INPUT: 4 bits from IR
- OUTPUT:
  - 12 bits to be distributed to all components  
(microinstruction)
  - $C_P E_P L'_M C_E' L'_I E'_I L'_A E_A$   
 $S_U E_U L'_B L'_O$

# Accumulator (ACC)



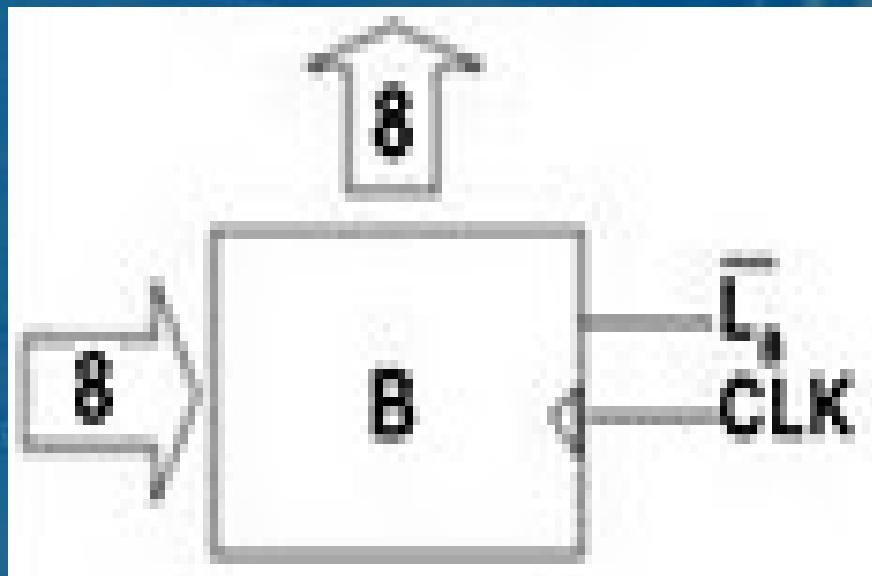
- Register buffer for storing temporary computation result
- INPUT : 8-bit from bus W
- OUTPUT :
  - 8-bit to bus W
  - 8-bit to ALU
- $L'_A$  :
  - 0 :  $ACC \leftarrow bus\ W$
- $E_A$  :
  - 1 :  $ACC \rightarrow bus\ W$

# ALU



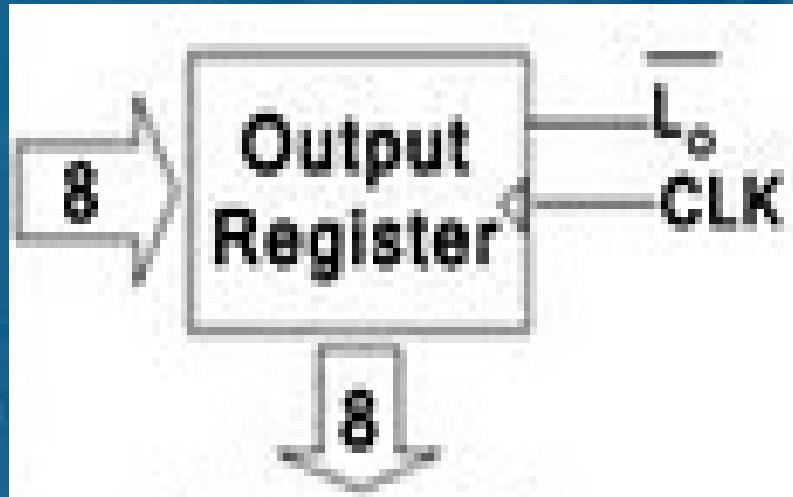
- Adder & Subtractor Only
- INPUT :
  - 8-bit from ACC
  - 8-bit from B register
- Output :
  - 8-bit to bus W
- $S_U$  :
  - 1 : subtract
  - 0 : add
- $E_U$  :
  - 1 : ALU  $\rightarrow$  bus W

# B Register



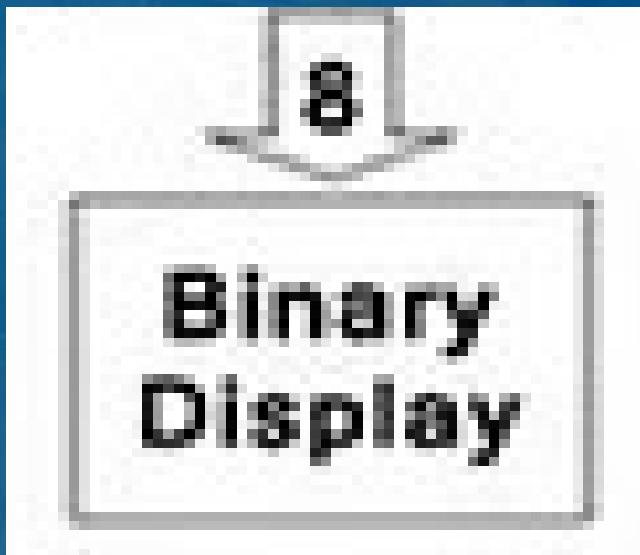
- Buffer for arithmetic operation
- INPUT : 8-bit from bus W
- Output : 8-bit to ALU
- $L'_B$  :
  - 0 :  $B \leftarrow \text{bus } W$

# Output Register



- *Output port*
  - The processed data can be accessed by output device via this register
- INPUT : 8-bit from ACC via bus W
- OUTPUT : 8-bit to output device
- $L'_o$  :
  - 0 : Output Register  $\leftarrow$  bus W

# Binary Display



- An output device that consists of 8 LEDs
- can be changed with any other output devices

# Instruction Set

#	Mnemonic	Operand Num	Operation	Opcode
1	LDA	1	Load memory data to acc.	0000
2	ADD	1	Add acc. with memory data	0001
3	SUB	1	Sub acc. with memory data	0010
4	OUT	0	Move out the acc. Data	1110
5	HLT	0	Stop program	1111

# Instruction Cycle

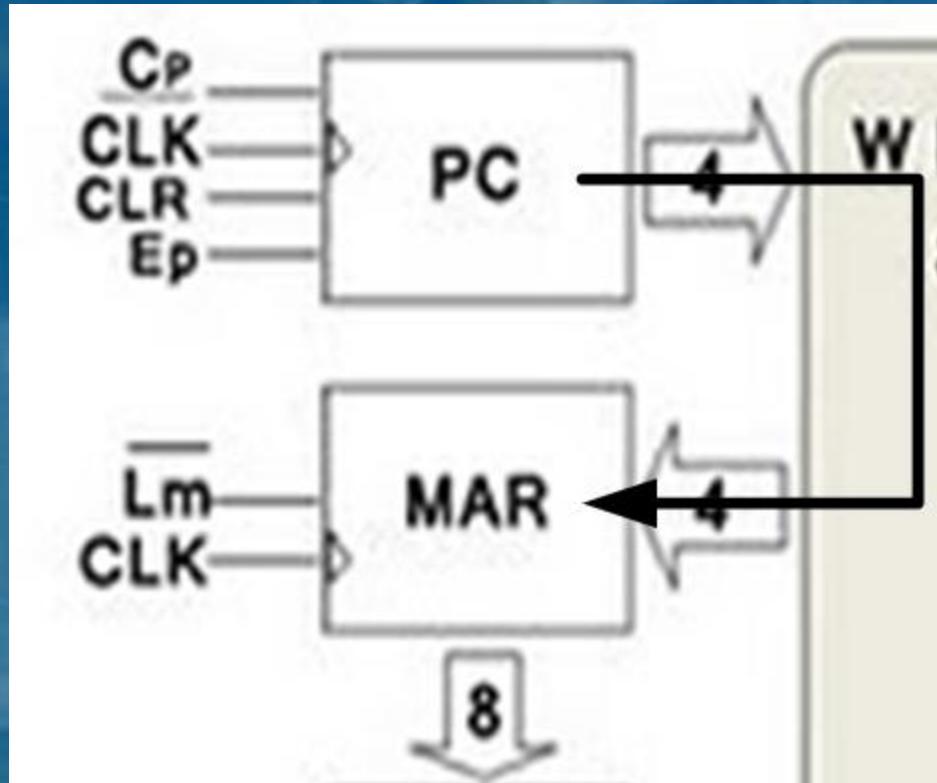
- **Fetch Cycle**
  - T1 (Address State)
  - T2 (Increment State)
  - T3 (Memory State)
- **Execution Cycle**
  - Three step (T4, T5, T6), but the task of each steps depends on the instruction

# Fetch Cycle - T0 (Initial State)

<b>CON=</b>	<b>C<sub>P</sub></b>	<b>E<sub>P</sub></b>	<b>L'<sub>M</sub></b>	<b>C'<sub>E</sub></b>	<b>L'<sub>I</sub></b>	<b>E'<sub>I</sub></b>	<b>L'<sub>A</sub></b>	<b>E<sub>A</sub></b>	<b>S<sub>U</sub></b>	<b>E<sub>U</sub></b>	<b>L'<sub>B</sub></b>	<b>L'<sub>O</sub></b>
	0	0	1	1	1	1	1	0	0	0	1	1

# T1 (Address State)

CON=	C <sub>P</sub>	E <sub>P</sub>	L' <sub>M</sub>	C' <sub>E</sub>	L' <sub>I</sub>	E' <sub>I</sub>	L' <sub>A</sub>	E <sub>A</sub>	S <sub>U</sub>	E <sub>U</sub>	L' <sub>B</sub>	L' <sub>O</sub>
	0	1	0	1	1	1	1	0	0	0	1	1



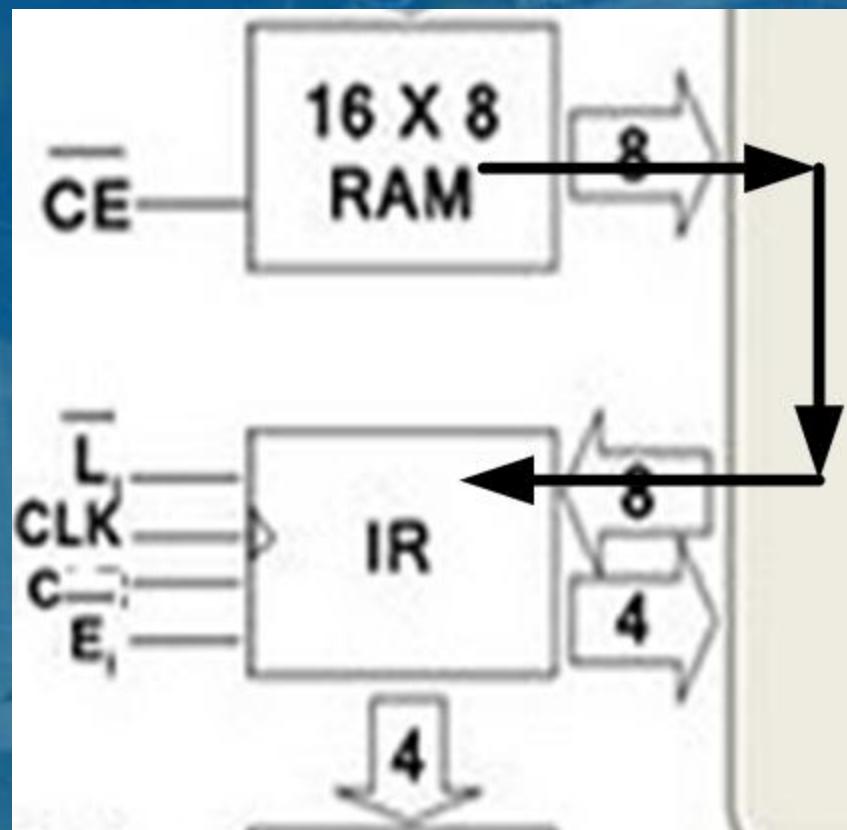
## T2 (Increment State)

<b>CON=</b>	<b>C<sub>P</sub></b>	<b>E<sub>P</sub></b>	<b>L' M</b>	<b>C' E</b>	<b>L' I</b>	<b>E' I</b>	<b>L' A</b>	<b>E<sub>A</sub></b>	<b>S<sub>U</sub></b>	<b>E<sub>U</sub></b>	<b>L' B</b>	<b>L' O</b>
	<b>1</b>	0	1	1	1	1	1	0	0	0	1	1

- Increment the value of PC

# T3 (Memory State)

CON=	C <sub>P</sub>	E <sub>P</sub>	L' <sub>M</sub>	C' <sub>E</sub>	L' <sub>I</sub>	E' <sub>I</sub>	L' <sub>A</sub>	E <sub>A</sub>	S <sub>U</sub>	E <sub>U</sub>	L' <sub>B</sub>	L' <sub>O</sub>
	0	0	1	0	0	1	1	0	0	0	1	1

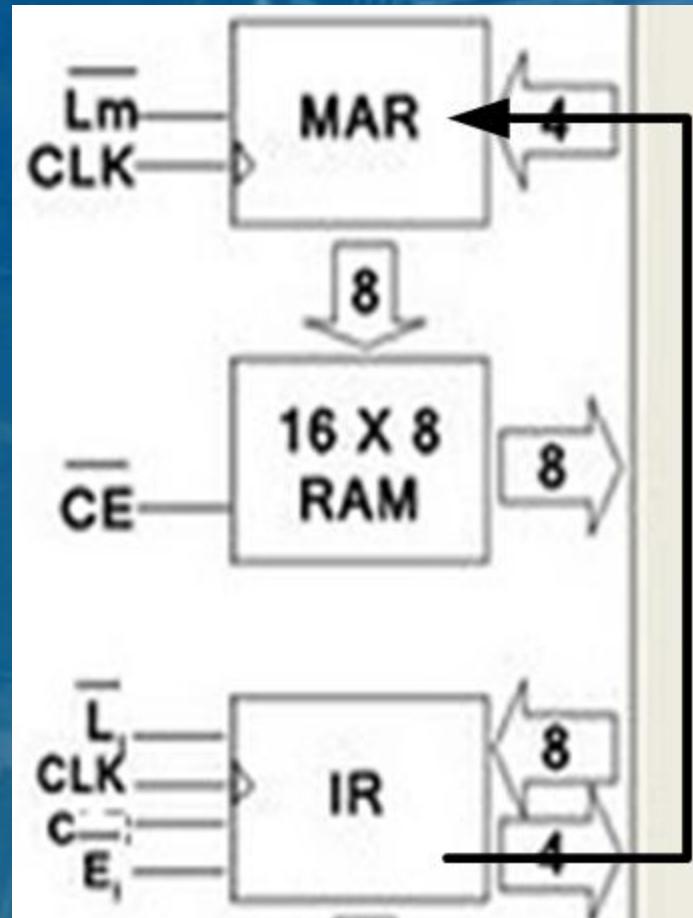


# Execution Cycle (LDA Instruction)

- For LDA instruction, only T4 and T5 states that will be active
  - T4 : memory address is sent from IR to MAR
  - T5 : data from memory is fetched and send to ACC
  - T6 : do nothing ! = T0

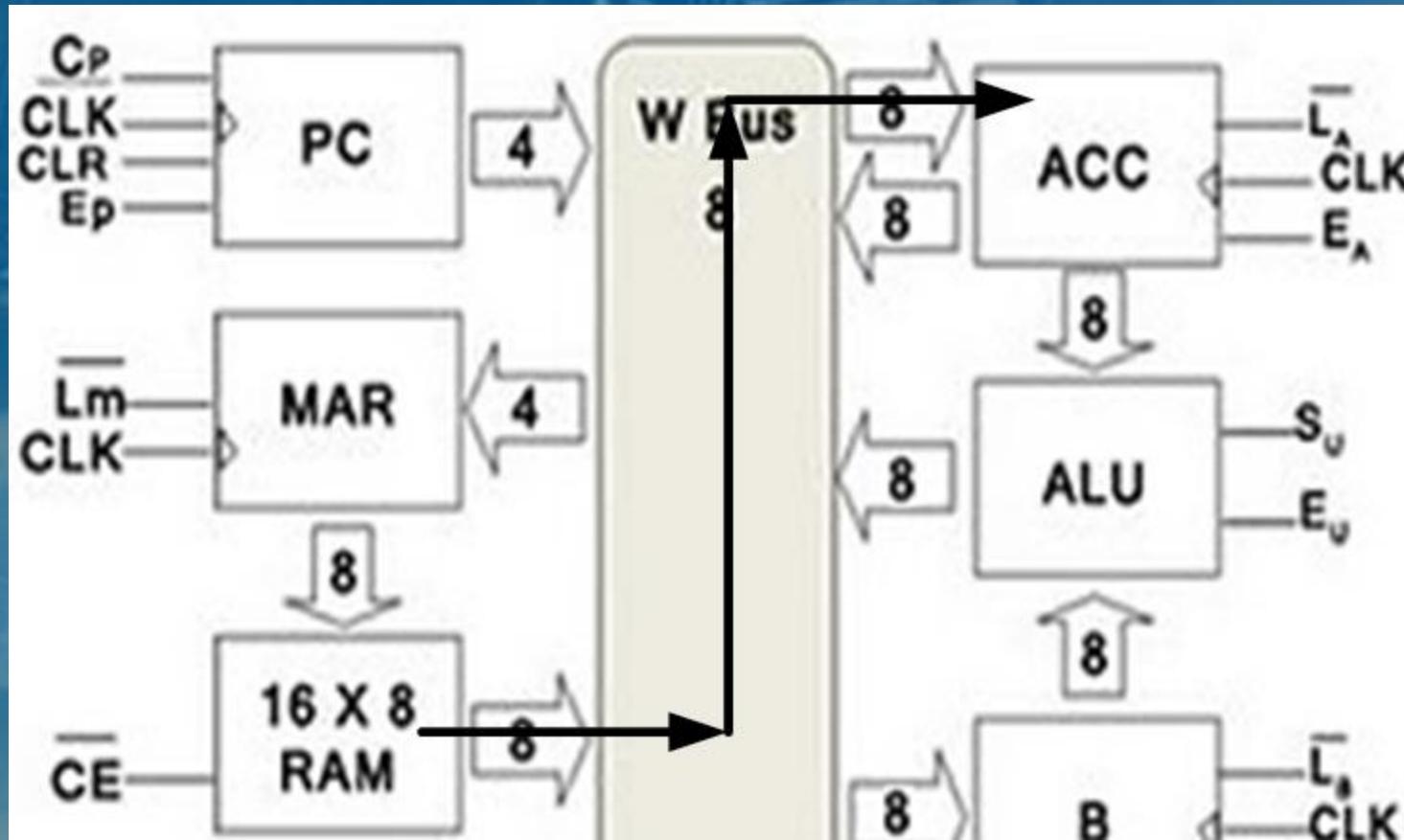
# T4 (send IR value to MAR)

<b>CON=</b>	<b>C<sub>P</sub></b>	<b>E<sub>P</sub></b>	<b>L'<sub>M</sub></b>	<b>C'<sub>E</sub></b>	<b>L'<sub>I</sub></b>	<b>E'<sub>I</sub></b>	<b>L'<sub>A</sub></b>	<b>E<sub>A</sub></b>	<b>S<sub>U</sub></b>	<b>E<sub>U</sub></b>	<b>L'<sub>B</sub></b>	<b>L'<sub>O</sub></b>
	0	0	<b>0</b>	1	1	<b>0</b>	1	0	0	0	1	1



# T5 (send RAM data to ACC)

<b>CON=</b>	<b>C<sub>P</sub></b>	<b>E<sub>P</sub></b>	<b>L'<sub>M</sub></b>	<b>C'<sub>E</sub></b>	<b>L'<sub>I</sub></b>	<b>E'<sub>I</sub></b>	<b>L'<sub>A</sub></b>	<b>E<sub>A</sub></b>	<b>S<sub>U</sub></b>	<b>E<sub>U</sub></b>	<b>L'<sub>B</sub></b>	<b>L'<sub>O</sub></b>
	0	0	1	0	1	1	0	0	0	0	1	1



# Execution Cycle

- What about the execution cycle for ADD / SUB / OUT instruction ?

# SAP-1 Programming

- Problem :
  - Write a program using SAP-1 computer that computes the following operation :  
 $16+20+24+28-32$

# SAP-1 Programming (2)

- Solution :
  - Assign the data memory with this data:
    - R9=00010000(16), RA=00010100(20),  
RB=00011000(24), RC=00011100(28),  
RD=00100000 (32)
  - The codes:
    1. LDA R9
    2. ADD RA
    3. ADD RB
    4. ADD RC
    5. SUB RD
    6. OUT
    7. HLT

# SAP-1 Programming (3)

Press start button

(PC)  $\leftarrow$  0000      Start

## LDA R9

(MAR) $\leftarrow$ (PC)	fetch T1	(MAR)= 0000
R(MAR) $\leftarrow$ 0000		(R) = (R0)
(PC) $\leftarrow$ (PC)+1	fetch T2	(PC) = 0001
IR $\leftarrow$ 0000 1001	fetch T3	
(MAR) $\leftarrow$ IR(low)	exec T4	(MAR)= 1001
(ACC) $\leftarrow$ RAM	exec T5	(ACC)= 16
{NOOP}	exec T6	

## ADD RA

(MAR) $\leftarrow$ (PC)	fetch T1	(MAR)= 0001
R(MAR) $\leftarrow$ 0001		(R)= (R1)
(PC) $\leftarrow$ (PC)+1	fetch T2	(PC) = 0010
IR $\leftarrow$ 0001 1010	fetch T3	
(MAR) $\leftarrow$ IR(low)	exec T4	(MAR)= 1010
(B) $\leftarrow$ RAM	exec T5	(B)= (RA)
(ACC) $\leftarrow$ Adder/Subt.	exec T6	(ACC)= 16+20

## ADD RB

.

## PROGRAM/DATA MEMORY

R0 - 0000 1001	(LDA R9)
R1 - 0001 1010	(ADD RA)
R2 - 0001 1011	(ADD RB)
R3 - 0001 1100	(ADD RC)
R4 - 0010 1101	(SUB RD)
R5 - 1110 xxxx	(OUT)
R6 - 1111 xxxx	(HLT)
R7 - xxxx xxxx	
R8 - xxxx xxxx	
R9 - 0001 0000	(16)
RA - 0001 0100	(20)
RB - 0001 1000	(24)
RC - 0001 1100	(28)
RD - 0010 0000	(32)
RE - xxxx xxxx	
RF - xxxx xxxx	