SAP-1

Prepared By: Lec Tasmiah Tamzid Anannya
CSE Dept, MIST

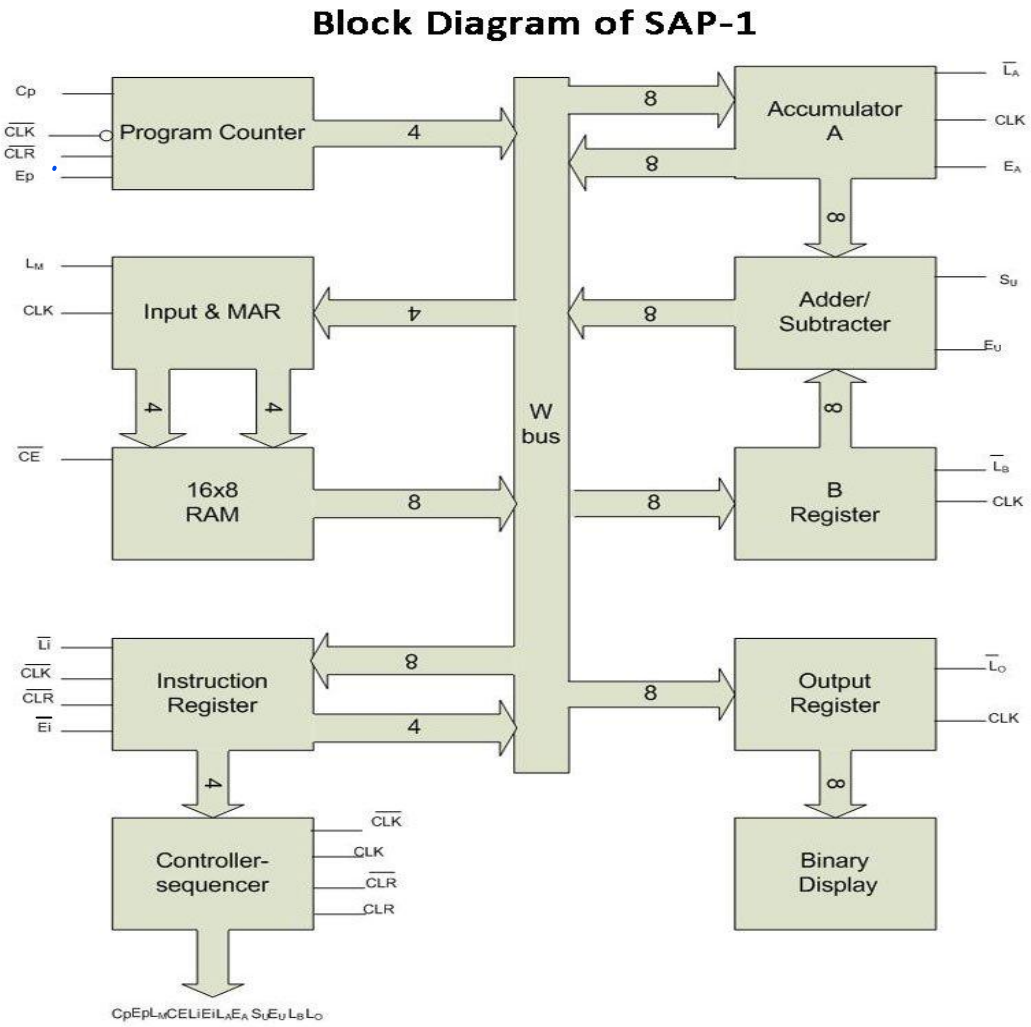
Reference Book

- Digital Computer Electronics
  - By Malvino & Brown

Introduction

- SAP-1 : Simple As Possible
- Simple computer like SAP covers many advanced concepts.
- SAP-1 is the first generation towards advanced computers

SAP-1 Architecture



SAP-1 Architecture

- Control Unit
  - Program counter
  - Instruction Register
  - Controller/Sequencer
- Memory Unit
  - MAR-Memory Address Register
  - 16x8 RAM-Random Access Memory
- ALU
  - Accumulator
  - B register
  - Adder/Subtractor
- Input/Output Unit
  - Input programming switches
  - Output port
  - Binary display

1. Instruction Set for SAP-1

- LDA- Load the Accumulator
  - LDA includes hexadecimal address for data to be loaded.
  - LDA 8H: load the accumulator with contents of memory location 8H
  - Suppose, memory location 8H of RAM contains:
    - R8=0010 1110
  - Execution of LDA 8H will result in:
    - A=0010 1110
  - Similarly, LDA AH means load the accumulator with contents of memory location AH.

2. Instruction Set for SAP-1-2

- ADD- Addition
  - ADD instruction includes the address of the word to be added with current Accumulator contents.
  - ADD 9H means add the contents of memory location 9H to the contents of Accumulator. [A=A+9H]
  - Suppose, decimal 3 is in Accumulator and decimal 2 is in memory location 9H.
    - A = 0000 0011
    - R9= 0000 0010
  - Now while execution ADD 9H following things happen,
    - At first register B is loaded with content of address 9H,
    - B= 0000 0010
    - Almost instantly, Adder/Subtractor forms the SUM of A and B register
    - SUM= 0000 0101
    - Then, this SUM value from Adder/Subtractor is loaded into A register
    - So, A = 0000 0101

3. Instruction Set for SAP-1

- SUB - Subtraction
  - Just like ADD instruction, SUB instruction also includes the address of the word to be subtracted from current Accumulator contents.
  - SUB 9H means subtract the contents of memory location 9H from the contents of Accumulator. [A=A-9H]
  - Suppose, decimal 3 is in Accumulator and decimal 2 is in memory location 9H.
    - A = 0000 0011
    - R9= 0000 0010
  - Now while execution SUB 9H following things happen,
    - At first register B is loaded with content of address 9H,
    - B= 0000 0010
    - Almost instantly, Adder/Subtractor forms the DIFF of A and B register
    - DIFF = 0000 0001
    - Then, this DIFF value from Adder/Subtractor is loaded into A register
    - So, A = 0000 0001

4. Instruction Set for SAP-1

- OUT- Output
  - OUT tells the computer to transfer the contents of Accumulator to Output port.
  - After OUT instruction , we can see the result in the Output port.
  - OUT is complete by itself, no need to include any address after this instruction.
  - That means, it does not involve any memory data.

5. Instruction Set for SAP-1

- HLT- Halt
  - This instruction tells computer to stop processing data.
  - Must use HLT at the end of each SAP-1 program.
  - This instruction also does not include any memory data.
  - It is also a complete itself.

Instruction Set for SAP-1

- Total 5 instruction for SAP-1
  - LDA
  - ADD
  - SUB
  - OUT
  - HLT

- LDA, ADD, SUB are called memory reference instructions as they use data stored in memory.
- On the other hand, OUT and HLT are not memory reference instructions as the do not use data stored in memory.

Mnemonics

- Abbreviated instructions like LDA, ADD, SUB, OUT, HLT are called mnemonics.

Example of SAP-1 Program

- Suppose, following is a program loaded in the memory address 0H-5H and also data loaded in memory from 6H-FH.

Address	Mnemonics
0H	LDA 9H
1H	ADD AH
2H	ADD BH
3H	SUB CH
4H	OUT
5H	HLT

Address	Data
6H	FFH
7H	FFH
8H	FFH
9H	01H
AH	02H
BH	03H
CH	04H
DH	05H
EH	06H
FH	FEH

Example of SAP-1 Program

- A= 01H
- A= 01H+02H=03H
- A=03H+03H=06H
- A=06H-04H=02H
- OUTPUT PORT- 0000 0010

Address	Mnemonics
0H	LDA 9H
1H	ADD AH
2H	ADD BH
3H	SUB CH
4H	OUT
5H	HLT

Address	Data
6H	FFH
7H	FFH
8H	FFH
9H	01H
AH	02H
BH	03H
CH	04H
DH	05H
EH	06H
FH	FEH

Programming SAP-1

- Computer only understands binary, so we need some code which computer can interpret.
- These are called op code.

Mnemonic	Opcode
LDA	0000
ADD	0001
SUB	0010
OUT	1110
HLT	1111

Mnemonic	Opcode
LDA FH	0000 1111
ADD AH	0001 1010
SUB 3H	0010 0011
OUT	1110 XXXX
HLT	1111 XXXX

Example

This program is in Assembly Language.

Address	Mnemonics
0H	LDA 9H
1H	ADD AH
2H	ADD BH
3H	SUB CH
4H	OUT
5H	HLT

Source Program

Now the program is in Machine Language.

Address	Mnemonics
0000	0000 1001
0001	0001 1010
0010	0001 1011
0011	0010 1100
0100	1110 XXXX
0101	1111 XXXX

Object Program

Instruction= XXXX XXXX

Instruction Field

Address Field

Example

- How would you program SAP-1 to solve the following problem:

16+20+24-32

Address	Contents
0H	LDA 6H
1H	ADD 7H
2H	ADD 8H
3H	SUB 9H
4H	OUT
5H	HLT
6H	10H
7H	14H

Address	Contents
8H	18H
9H	20H
AH	XX
BH	XX
CH	XX
DH	XX
EH	XX
FH	XX

Ring Counter

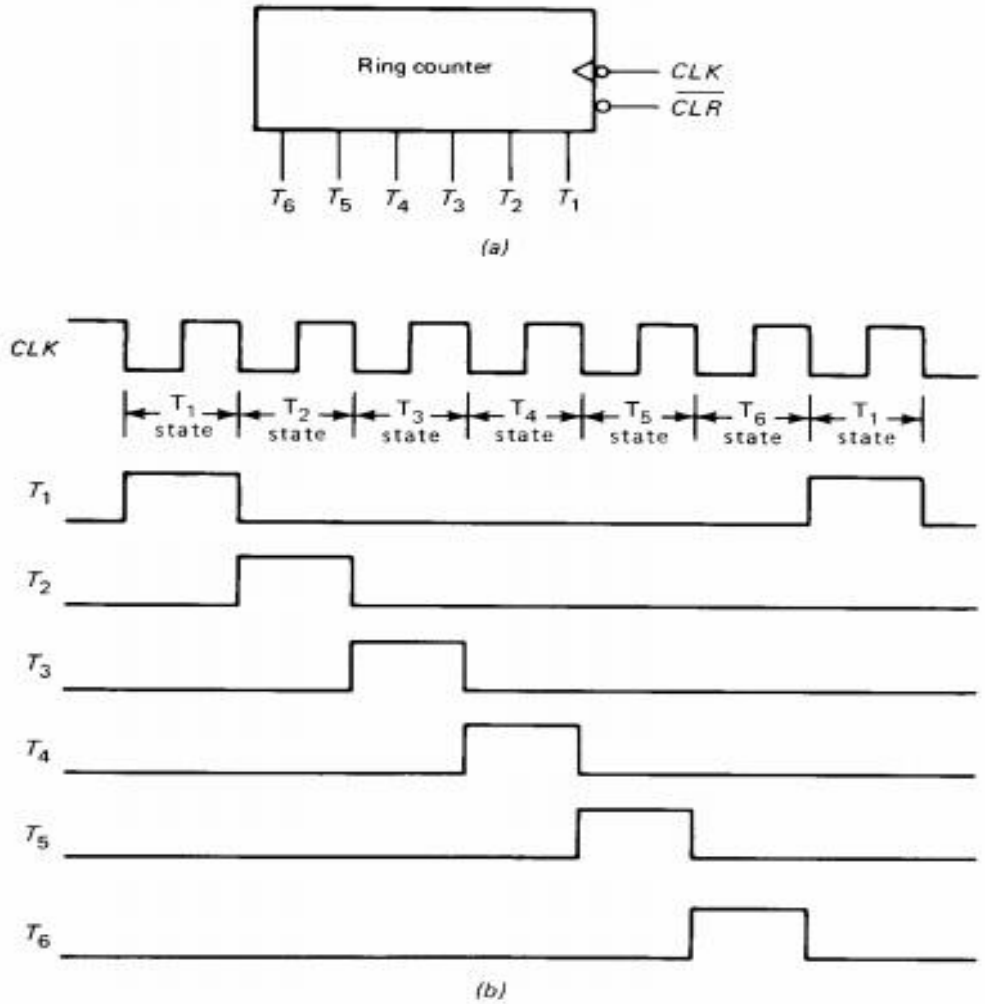


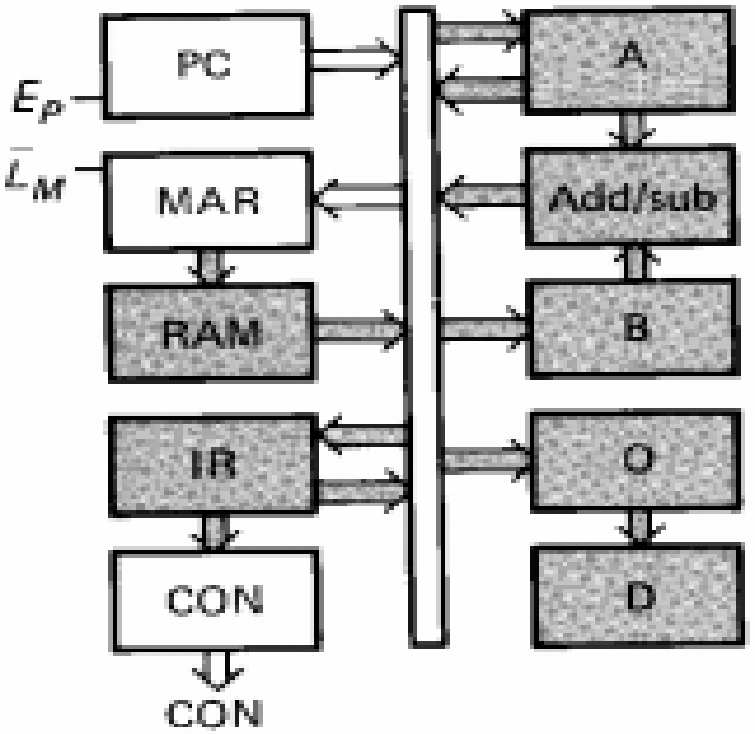
Fig. 10-2 Ring counter: (*a*) symbol; (*b*) clock and timing signals.

Ring Counter

- $T=T_6T_5T_4T_3T_2T_1$
- At the beginning, it will be
  - T=000001
- Then
  - T=000010
  - T=000100
  - T=001000
  - T=010000
  - T=100000
- Then, the ring counter is reset to 000001 and the cycle repeats.

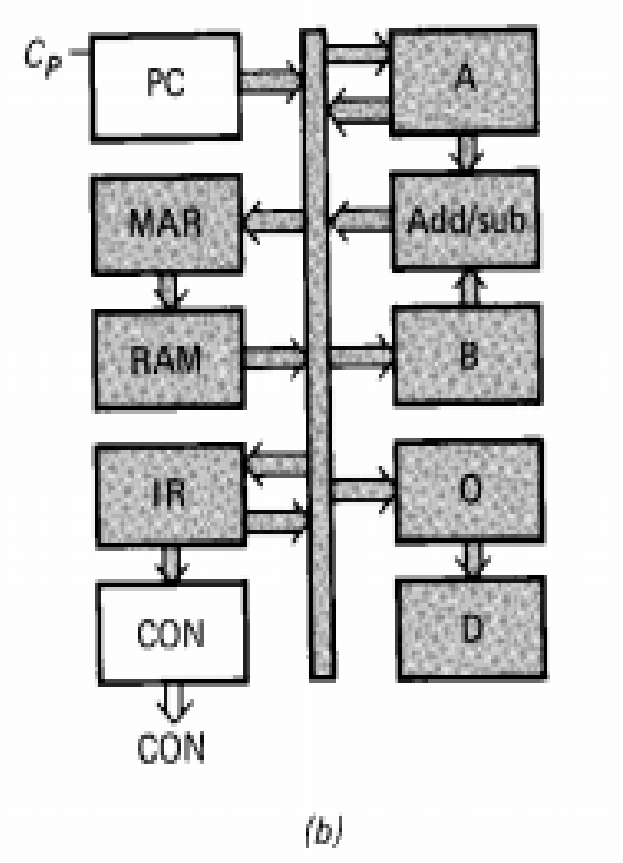
1. Fetch Cycle- Address state (T_1)

- T_1 state is called the address state.
- The address in PC is transferred to MAR in this state.



$$\begin{aligned} \text{CON} &= C_P E_P \bar{L}_M \overline{CE} \quad \bar{L}_I \bar{E}_I \bar{L}_A E_A \quad S_U E_U \bar{L}_B \bar{L}_O \\ &= 0\ 1\ 0\ 1 \quad\ 1\ 1\ 1\ 0 \quad\ 0\ 0\ 1\ 1 \end{aligned}$$

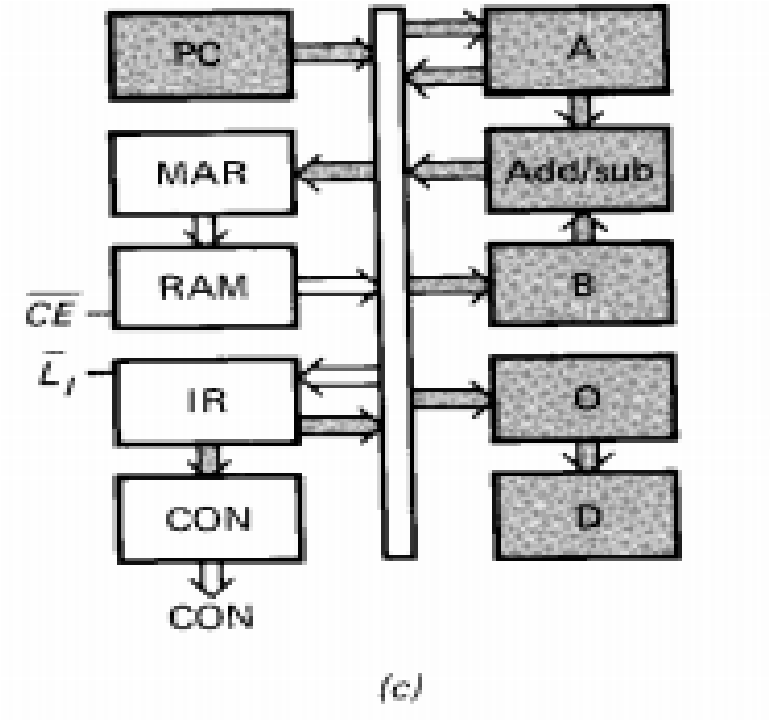
2. Fetch Cycle- Increment State (T_2)



$$CON = C_p E_p \bar{L}_M \overline{CE} \quad \bar{L}_I \bar{E}_I \bar{L}_A E_A \quad S_U E_U \bar{L}_B \bar{L}_O$$
$$= 1\ 0\ 1\ 1 \quad 1\ 1\ 1\ 0 \quad 0\ 0\ 1\ 1$$

3. Fetch Cycle- Memory State (T_3)

- Cause the addressed RAM instruction is transferred from the memory to the IR.



$$CON = C_p E_p \overline{L_M}\,\overline{CE}\;\; \overline{L_I}\,\overline{E_I}\,\overline{L_A} E_A \;\; S_U E_U \overline{L_B}\,\overline{L_O}$$
$$= 0\;0\;1\;0 \;\; 0\;1\;1\;0 \;\; 0\;0\;1\;1$$

Execution Cycle

- The next three states are the execution cycle of SAP-1.
- The register transfer depends on particular instruction being executed.

1. Execution Cycle- LDA

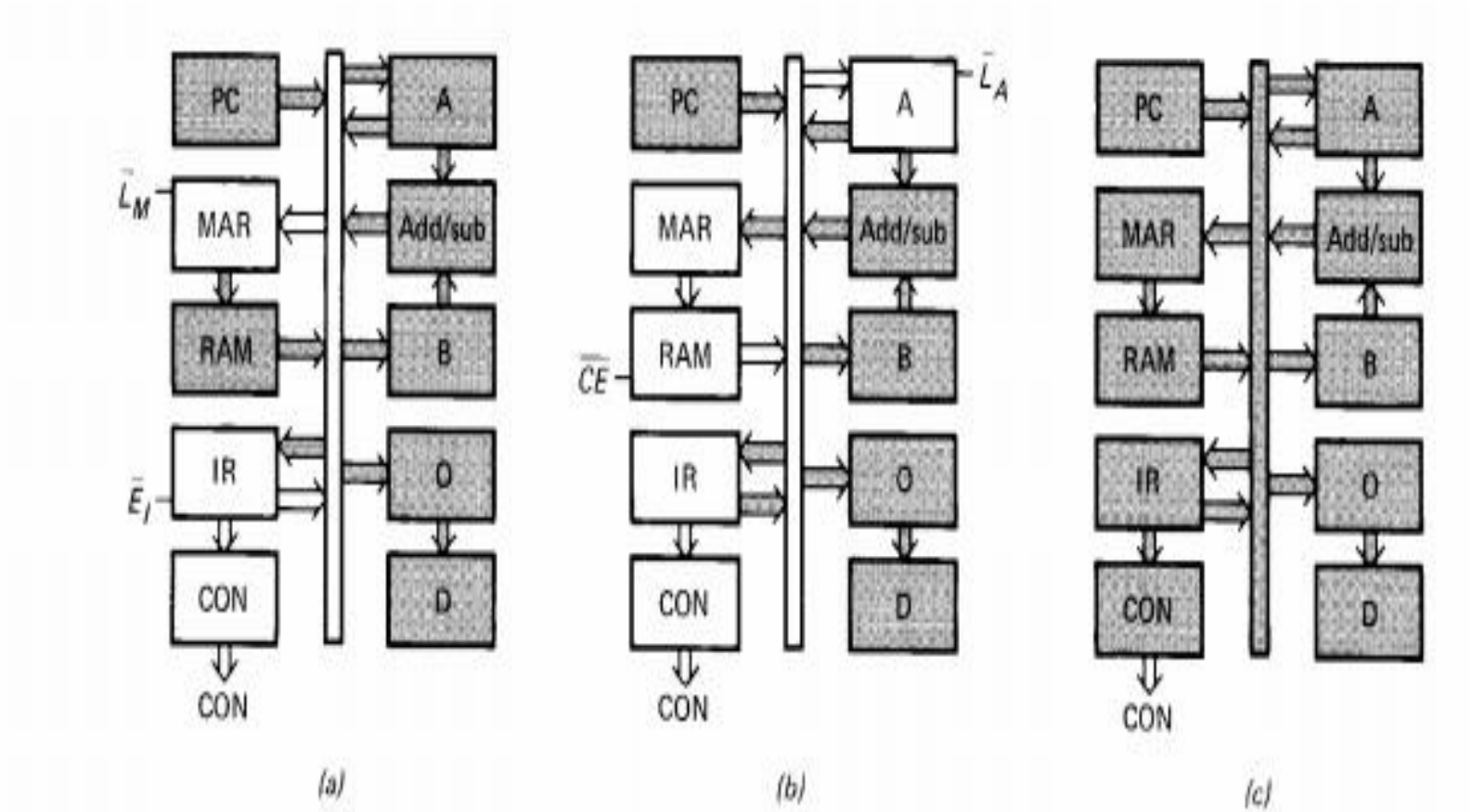


Fig. 10-4 LDA routine: (*a*) T_4 state; (*b*) T_5 state; (*c*) T_6 state.

Timing Diagram of LDA

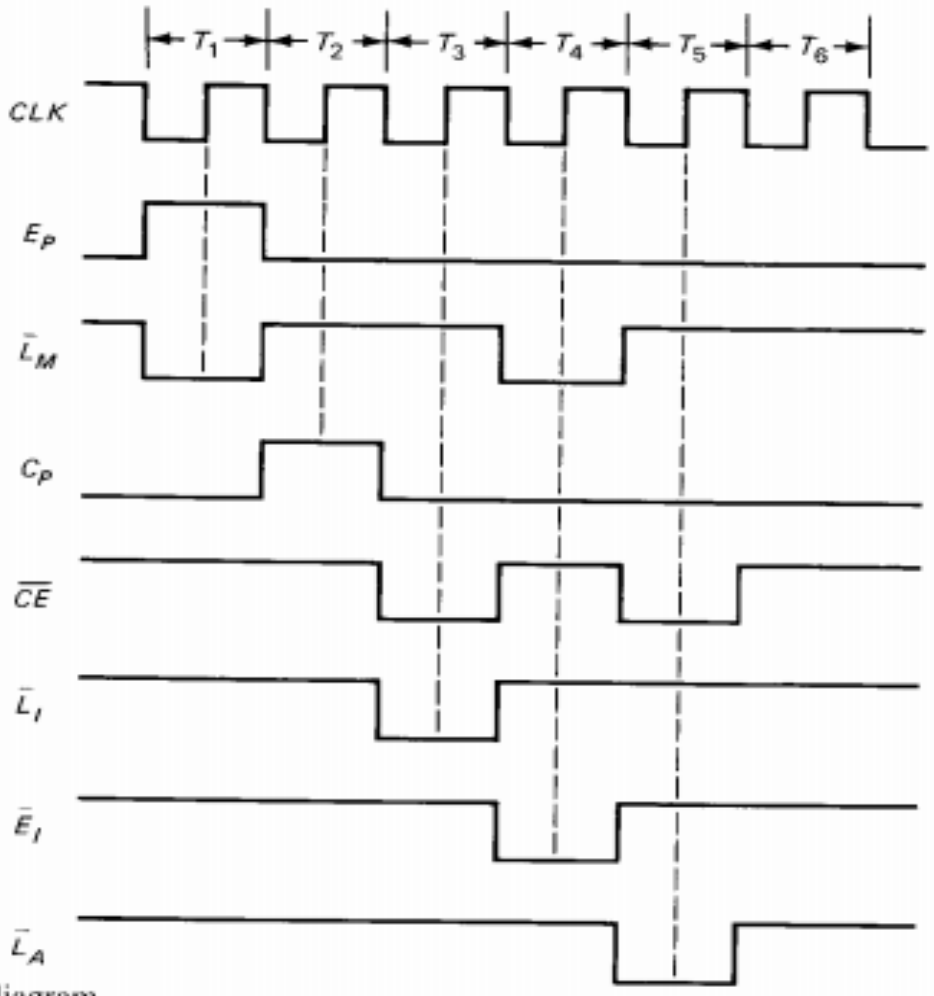


Fig. 10-5 Fetch and LDA timing diagram.

2. Execution Cycle- ADD

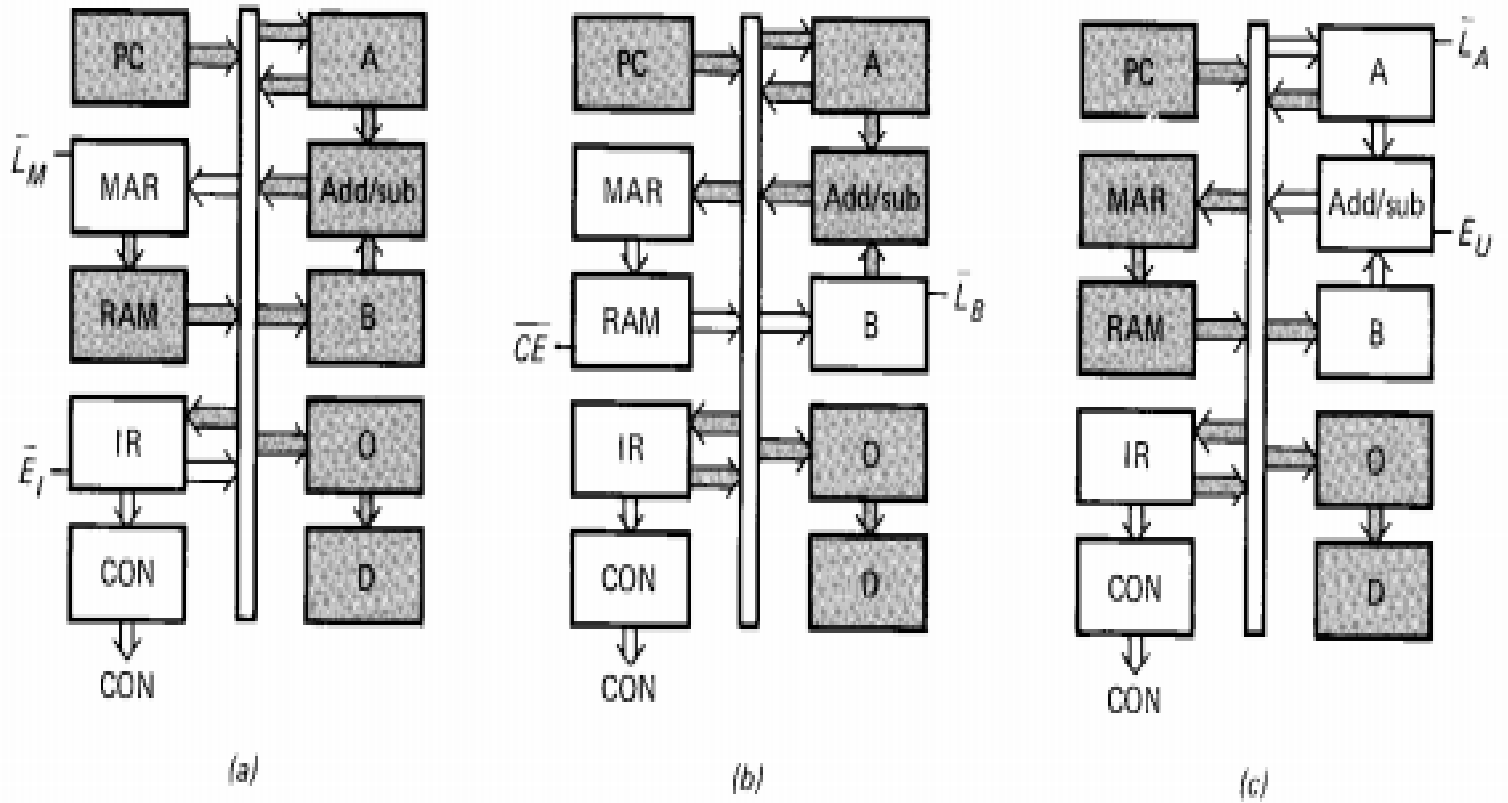


Fig. 10-6 ADD and SUB routines: (*a*) T_4 state; (*b*) T_5 state; (*c*) T_6 state.

3. Execution Cycle- SUB

- Can you tell what will happen during SUB instruction??

4. Execution Cycle- OUT

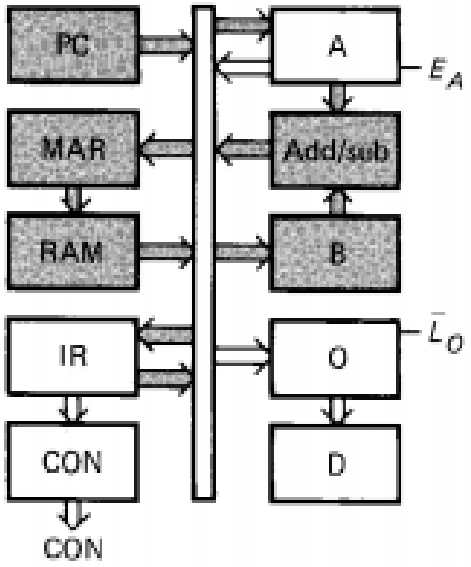


Fig. 10-8 T_4 state of OUT instruction.

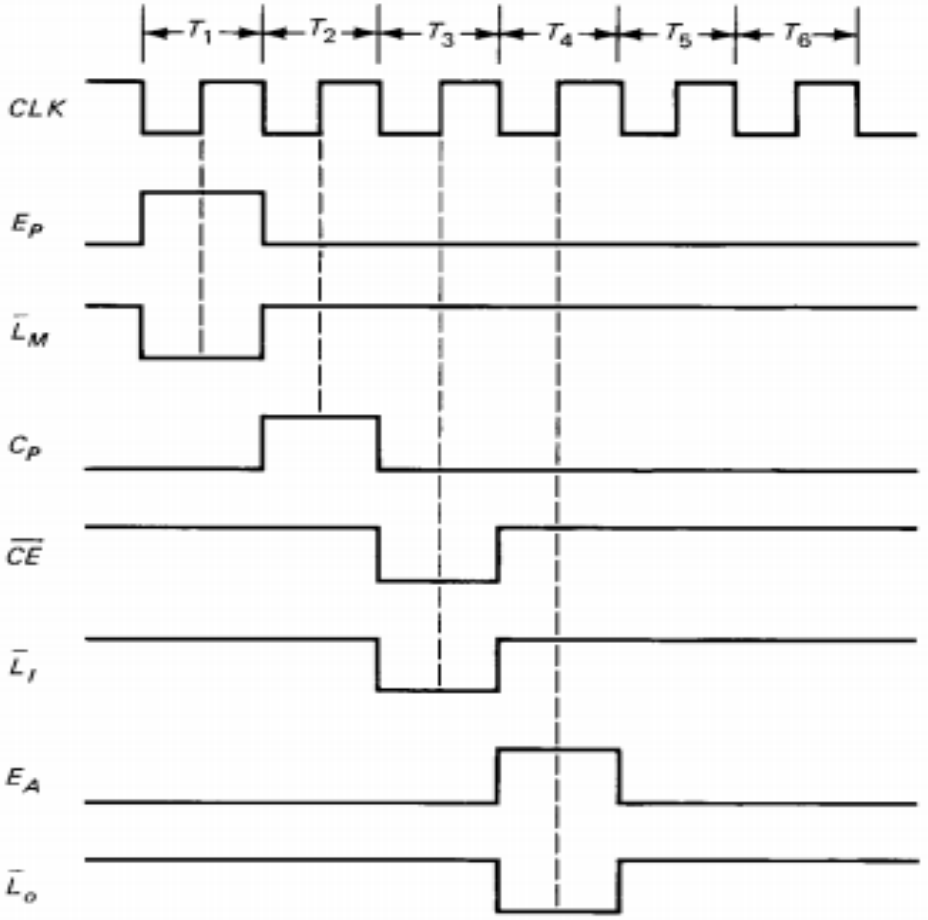


Fig. 10-9 Fetch and OUT timing diagram.

5. Execution Cycle- HLT

- HLT does not require control routine.
- Because no registers are involved during this instruction.
- The controller sequencer stops the computer by turning off the clock.

Machine Cycle and Instruction Cycle

- SAP-1 has six T states (3 fetch and 3 execution)
- These six states are called machine cycle.
- It takes one machine cycle to fetch and execute one instruction.

- SAP-1 clock has frequency of 1 khz.
- Which is equivalent to period of 1 ms.
- So, it takes 6 ms for a SAP-1 machine cycle.

Machine Cycle and Instruction Cycle

- For SAP-2, some of the instructions take more than one machine cycle to fetch and execute.
- The number of T states required to fetch and execute an instruction is called instruction cycle.
- For SAP-1, machine cycle is equal to instruction cycle. For SAP-2 and other microcomputers an instruction cycle may equal two or more machine cycles.

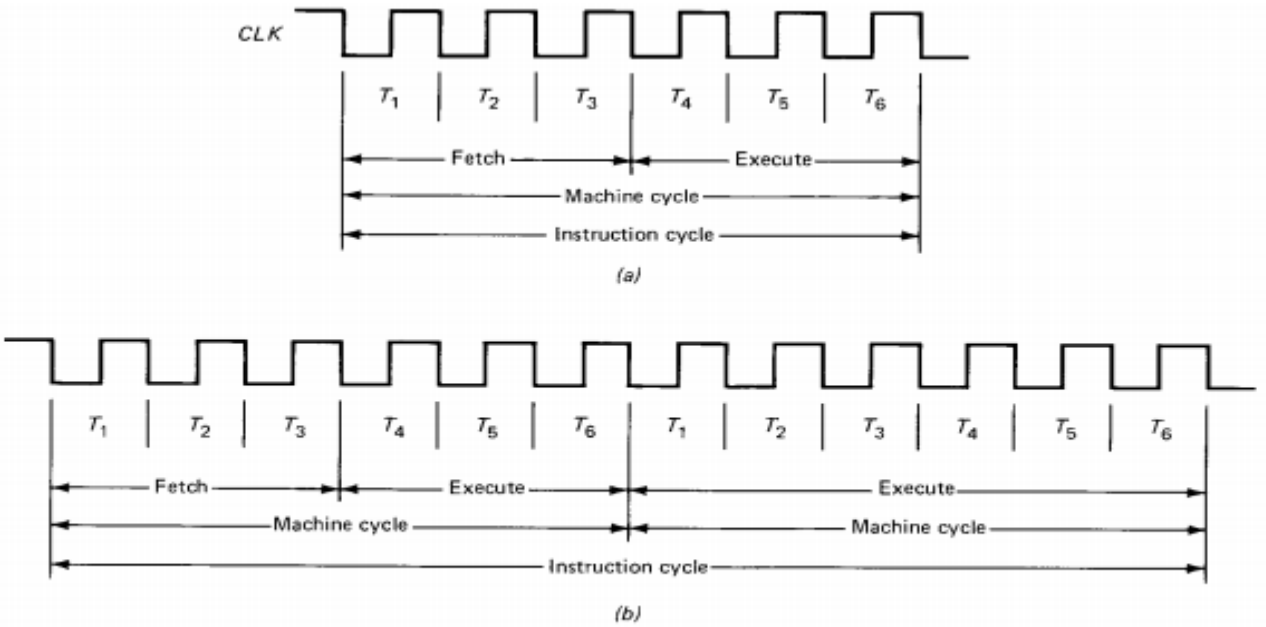


Fig. 10-10 (*a*) SAP-1 instruction cycle; (*b*) instruction cycle with two machine cycles.

Example

- The 8080/8085 programming manual says that it takes 13 T states to fetch and execute LDA instruction. If the system clock has a frequency of 2.5 MHz, how long is an instruction cycle?

Microinstructions

- The controller sequencer sends out control words, one word during each T states.
- Because it produces small steps towards data processing they are known as microinstructions.

Macroinstructions

- The instructions that are used in programming are called macroinstructions. – LDA, ADD, SUB…
- Each SAP-1 macroinstructions are made up of three microinstructions.

TABLE 10-3

Macro	State	$C_P E_P \overline{L}_M \overline{CE}$	$\overline{L}_I \overline{E}_I \overline{L}_A E_A$	$S_U E_U \overline{L}_B \overline{L}_O$	Active	
LDA	T_4	0 0 0 1	1 0 1 0	0 0 1 1	$\overline{L}_M$, $\overline{E}_I$	1A3H
	T_5	0 0 1 0	1 1 0 0	0 0 1 1	$\overline{CE}$, $\overline{L}_A$	2E3H
	T_6	0 0 1 1	1 1 1 0	0 0 1 1	None	3E3H

Microinstructions and Macroinstructions

TABLE 10-5. SAP-1 MICROPROGRAM†

Macro	State	CON	Active
LDA	T_4	1A3H	$\overline{L}_M$, $\overline{E}_I$
	T_5	2C3H	$\overline{CE}$, $\overline{L}_A$
	T_6	3E3H	None
ADD	T_4	1A3H	$\overline{L}_M$, $\overline{E}_I$
	T_5	2E1H	$\overline{CE}$, $\overline{L}_B$
	T_6	3C7H	$\overline{L}_A$, E_U
SUB	T_4	1A3H	$\overline{L}_M$, $\overline{E}_I$
	T_5	2E1H	$\overline{CE}$, $\overline{L}_B$
	T_6	3CFH	$\overline{L}_A$, S_U, E_U
OUT	T_4	3F2H	E_A, $\overline{L}_O$
	T_5	3E3H	None
	T_6	3E3H	None

† **CON** = $C_P E_P \overline{L}_M \overline{CE}$ $\overline{L}_I \overline{E}_I \overline{L}_A E_A$ $S_U E_U \overline{L}_B \overline{L}_O$.

Control Matrix

Microprogramming

- With larger instruction sets, control matrix becomes very complicated and requires hundreds or even thousands of gates.
- That's why require an alternative way.
- Microprogramming is the alternative way.

Storing the microinstructions in Control ROM

TABLE 10-6. SAP-1 CONTROL ROM

Address	Contents†	Routine	Active
0H	5E3H	Fetch	E_P, $\overline{L}_M$
1H	BE3H		C_P
2H	263H		$\overline{CE}$, $\overline{L}_I$
3H	1A3H	LDA	$\overline{L}_M$, $\overline{E}_I$
4H	2C3H		$\overline{CE}$, $\overline{L}_A$
5H	3E3H		None
6H	1A3H	ADD	$\overline{L}_M$, $\overline{E}_I$
7H	2E1H		$\overline{CE}$, $\overline{L}_B$
8H	3C7H		$\overline{L}_A$, E_U
9H	1A3H	SUB	$\overline{L}_M$, $\overline{E}_I$
AH	2E1H		$\overline{CE}$, $\overline{L}_B$
BH	3CFH		$\overline{L}_A$, S_U, E_U
CH	3F2H	OUT	E_A, $\overline{L}_O$
DH	3E3H		None
EH	3E3H		None
FH	X	X	Not used

† **CON** = $C_P E_P \overline{L}_M \overline{CE}$ $\overline{L}_I \overline{E}_I \overline{L}_A E_A$ $S_U E_U \overline{L}_B \overline{L}_O$.

Address ROM

TABLE 10-7. ADDRESS ROM

Address	Contents	Routine
0000	0011	LDA
0001	0110	ADD
0010	1001	SUB
0011	XXXX	None
0100	XXXX	None
0101	XXXX	None
0110	XXXX	None
0111	XXXX	None
1000	XXXX	None
1001	XXXX	None
1010	XXXX	None
1011	XXXX	None
1100	XXXX	None
1101	XXXX	None
1110	1100	OUT
1111	XXXX	None

Microprogramming of SAP-1

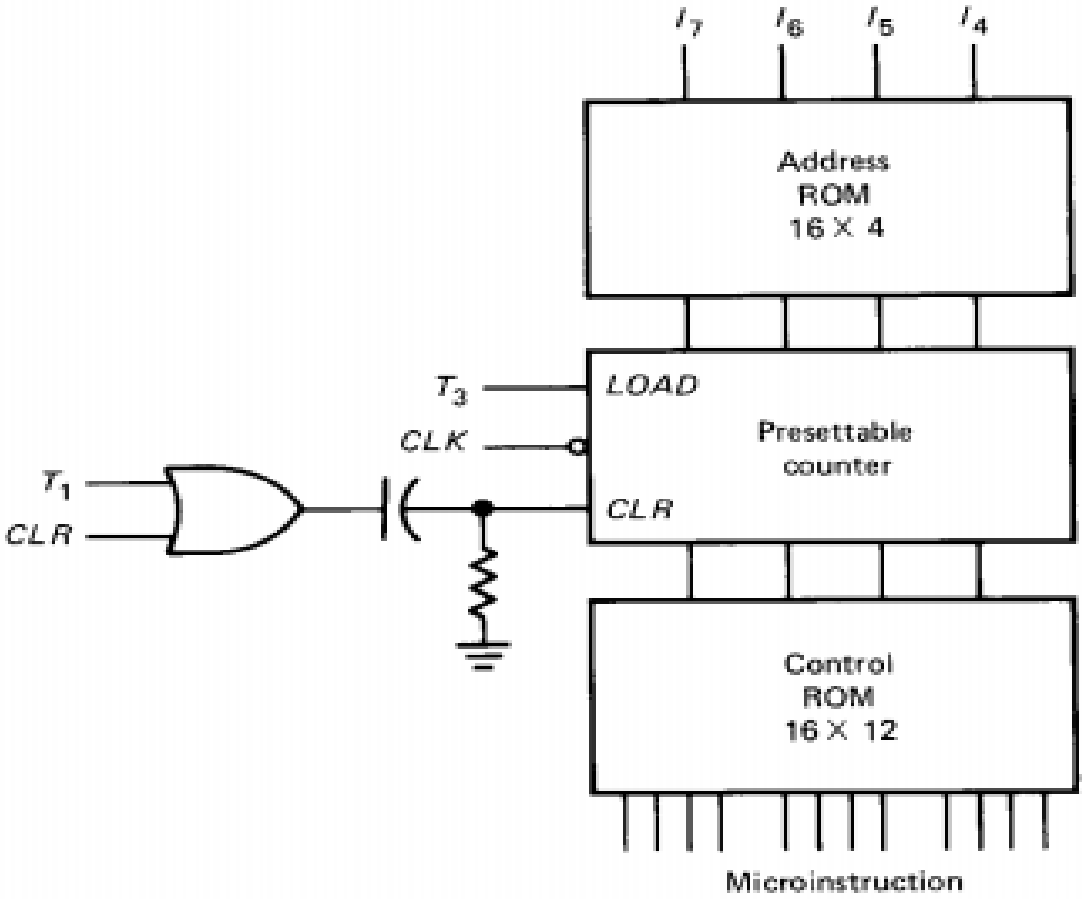


Fig. 10-16 Microprogrammed control of SAP-1.

Variable machine cycle

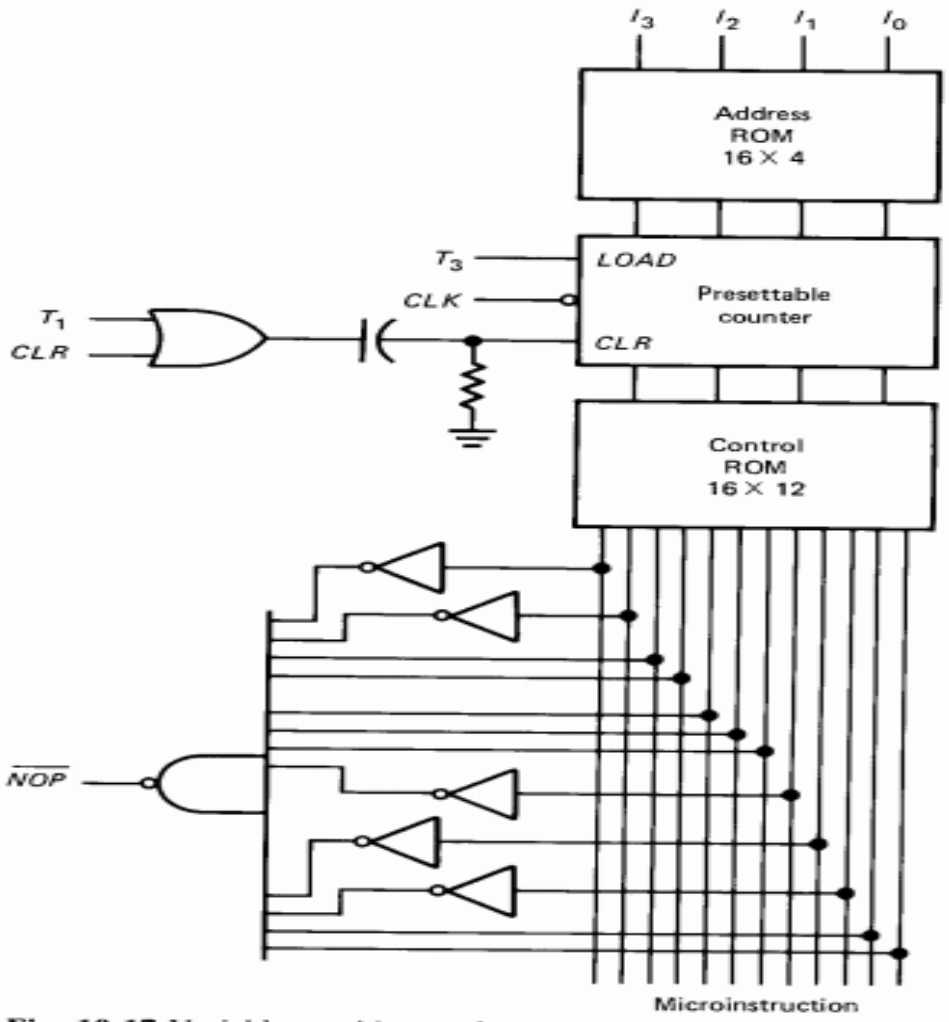


Fig. 10-17 Variable machine cycle.

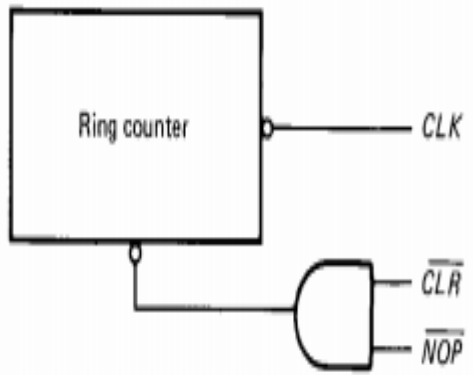


Fig. 10-18