

TIMING DIAGRAM for various machine cycles

The machine cycles are the basic operations performed by the processor, while instructions are executed. The time taken for performing each machine cycle is expressed in terms of T-states.

One T-state is the time period of one clock cycle of the microprocessor.

The various machine cycles are

- | | |
|--------------------------------|------------|
| 1. Opcode fetch | - 4 / 6 T |
| 2. Memory Read | - 3 T |
| 3. Memory Write | - 3 T |
| 4. I/O Read | - 3 T |
| 5. I/O Write | - 3 T |
| 6. Interrupt Acknowledge | - 6 / 12 T |
| 7. Bus Idle | - 2 / 3 T |

Instruction cycle

- **Instruction:** A command given to the microprocessor to perform an operation
- **Program :** A set of instructions given in a sequential manner to perform a particular task

The CPU fetches one instruction from memory at a time & executes it.

Instruction cycle = Fetch cycle + Execute cycle



Fetch Cycle : The steps taken by CPU to fetch the opcode from the memory

The time taken for fetch cycle is fixed.

Execute Cycle : The steps taken by CPU to fetch data & to perform the operation specified in the instruction

The time taken for execute cycle is variable which depends on the type of instruction ,i.e. 3 -byte , 2-byte & 1-byte instruction.

Machine cycle

The time required by the micro processor to complete the operation of accessing memory or I/O device .

Operations like :

- Opcode fetch
- Memory read
- Memory write
- I/O read
- I/O write

T - states

Microprocessor performs an operation in specific clock cycles.

Each clock cycle is called as T –States.

The number of T – states required to perform an operation is called Machine Cycle .

MPU Communication and Bus Timing

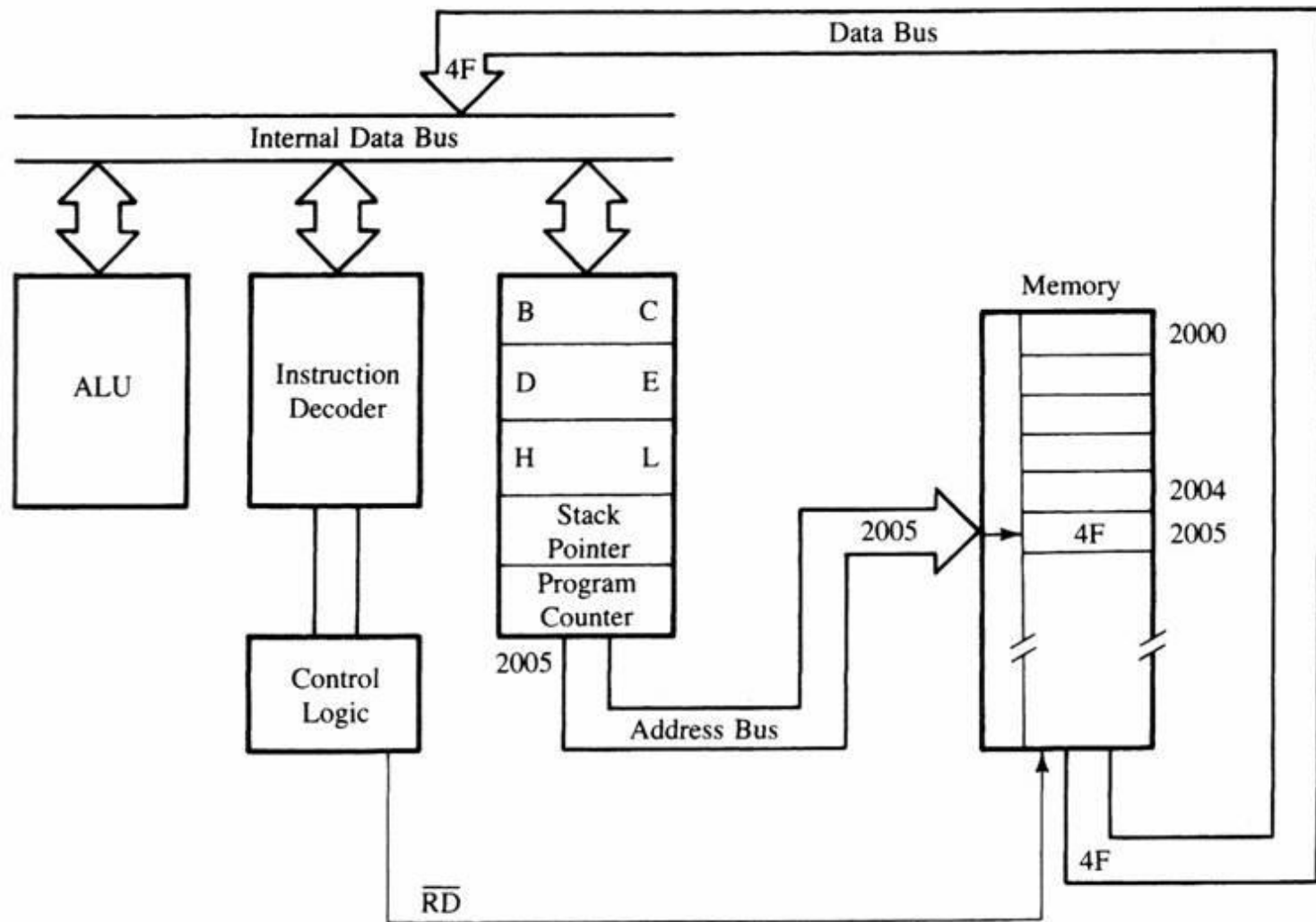


Figure 3: Moving data form memory to MPU using instruction **MOV C,A** (code machine 4FH = 0100 1111)

MPU Communication and Bus Timing

The Fetch Execute Sequence :

1. The μp placed a 16 bit memory address from PC (program counter) to address bus.

Figure 4: at T₁

- The high order address, 20H, is placed at A₁₅ – A₈.
 - the low order address, 05H, is placed at A_{D7} - A_{D0} and ALE is active high.
 - Synchronously the IO/M is in active low condition to show it is a memory operation.
2. At T₂ the active low control signal, RD, is activated so as to activate read operation; it is to indicate that the MPU is in fetch mode operation.

MPU Communication and Bus Timing

- 3.T3: The active low RD signal enabled the byte instruction, 4FH, to be placed on AD7 – AD0 and transferred to the MPU. While RD high, the data bus will be in high impedance mode.
- 4.T4: The machine code, 4FH, will then be decoded in instruction decoder. The content of accumulator (A) will then be copied into C register at time state, T4.

S_0 and S_1

Pin 29 (Output) and Pin 33 (Output)

- S_0 and S_1 are called Status Pins.
- They tell the current operation which is in progress in 8085.

S_1	S_0	Operation
0	0	Halt
0	1	Write
1	0	Read
1	1	Opcode Fetch

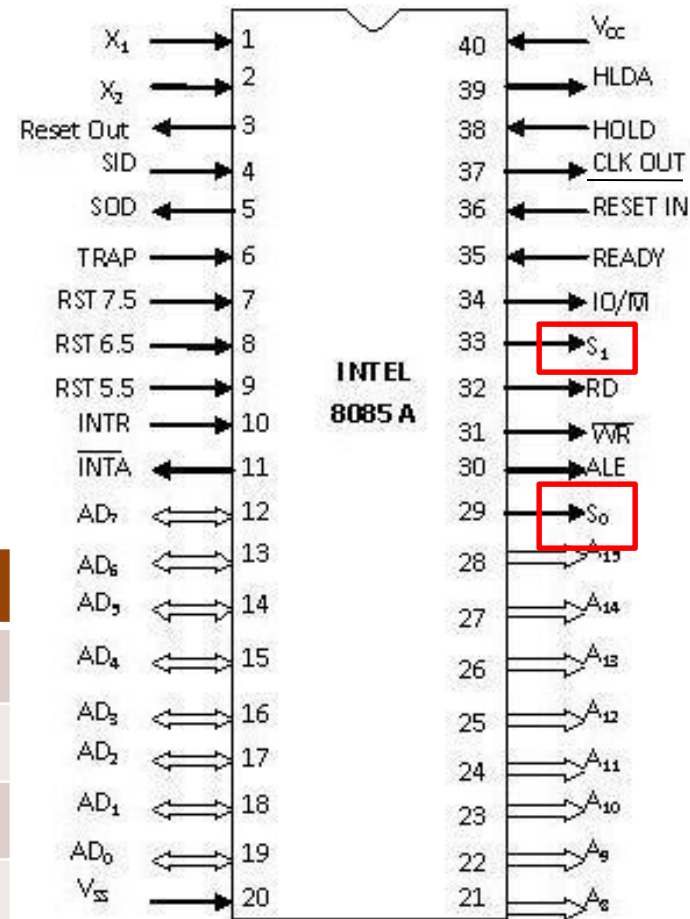


Table Showing $\overline{IO/\overline{M}}$, S_0 , S_1 and Corresponding Operations

Operations	$\overline{IO/\overline{M}}$	S_1	S_0
Opcode Fetch	0	1	1
Memory Read	0	1	0
Memory Write	0	0	1
I/O Read	1	1	0
I/O Write	1	0	1
Interrupt Ack.	1	1	1
Halt	High Impedance	0	0

MPU Communication and Bus Timing

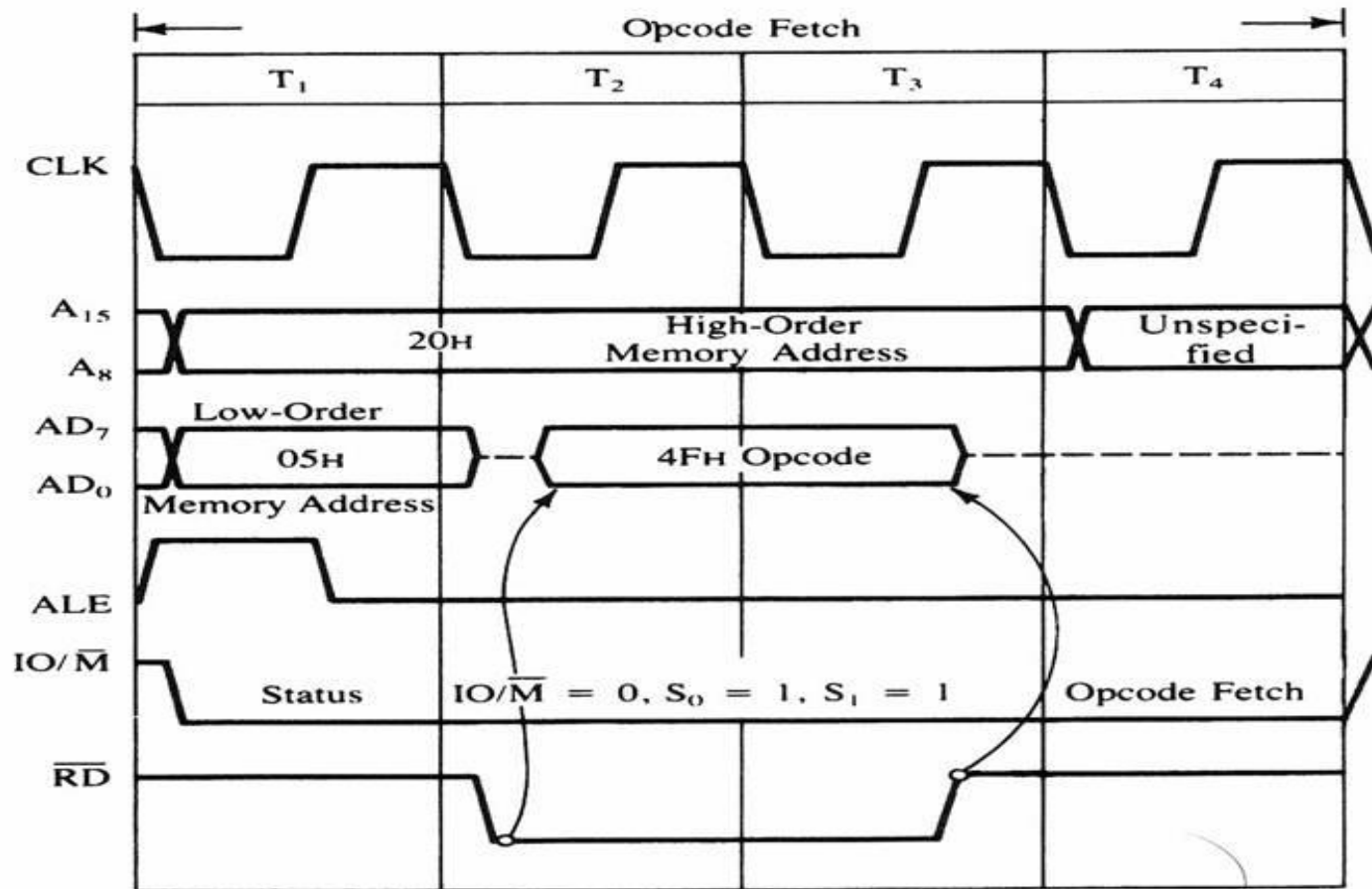


Figure 4: 8085 timing diagram for Opcode fetch cycle for MOV C,A .

Memory Read: The flow of data from the memory to the microprocessor.

Step 1 (T1) : microprocessor places the address on the address lines from program counter & activates ALE signal to multiplex the low order address.

It also sends status signals $\overline{IO/\overline{M}} = 0$, $S_1 = 1$, $S_0 = 0$ for memory read operation.

Step 2 (T2) : SE

Timing Diagram

Instruction:

A000h MOV A,B

Corresponding Coding:

A000h 78

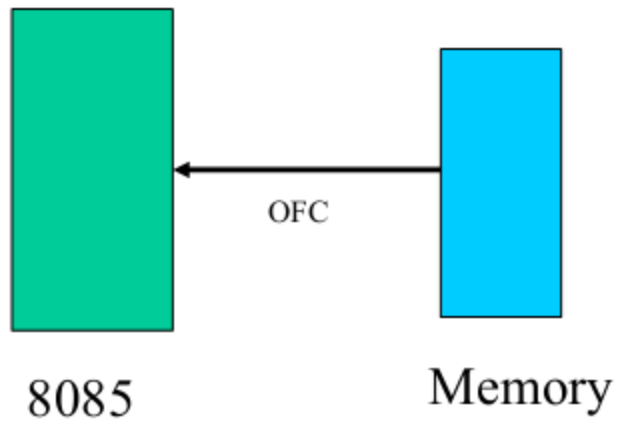
Timing Diagram

Instruction:

A000h MOV A,B

Corresponding Coding:

A000h 78



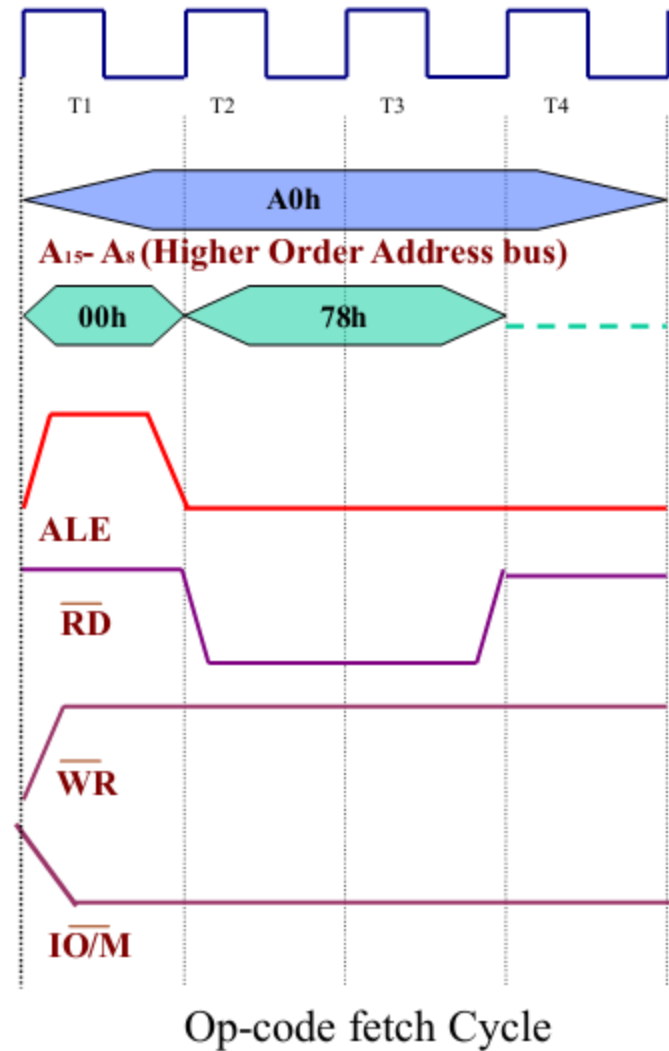
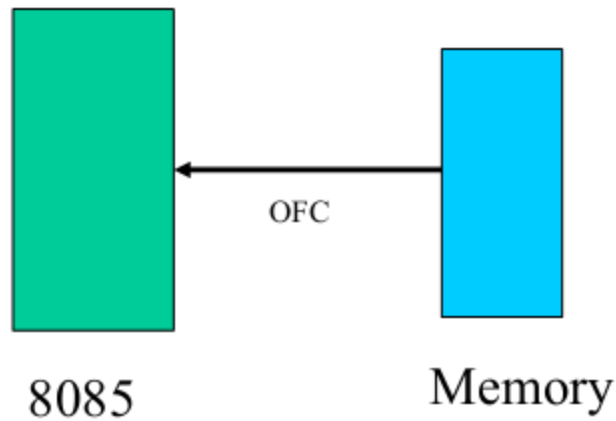
Timing Diagram

Instruction:

A000h MOV A,B

Corresponding Coding:

A000h 78



Timing Diagram

Instruction:

A000h MVI A,45h

Corresponding Coding:

A000h 3E

A001h 45

Timing Diagram

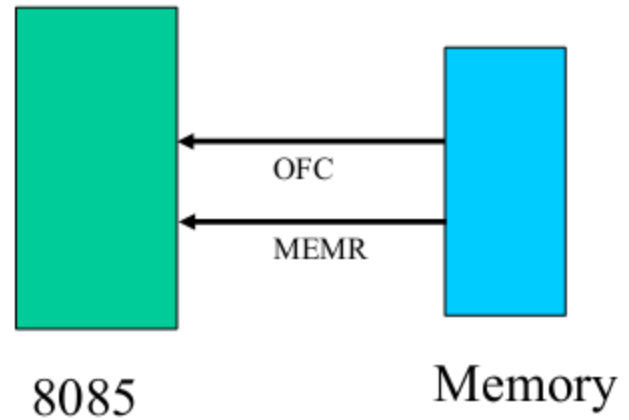
Instruction:

A000h MVI A,45h

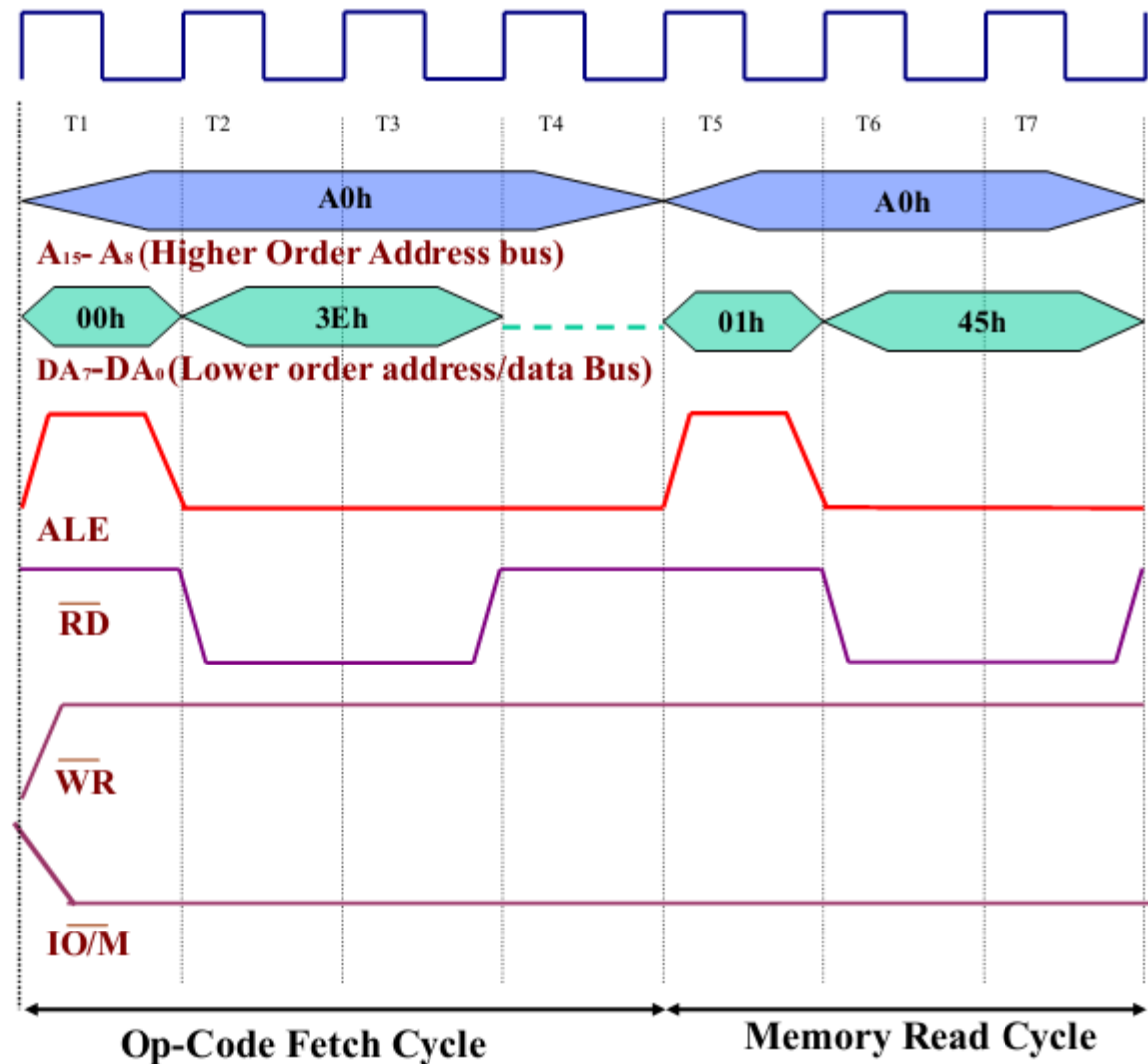
Corresponding Coding:

A000h 3E

A001h 45



Timing Diagram



Timing Diagram

Instruction:

A000h LXI A,FO45h

Corresponding Coding:

A000h 21

A001h 45

A002h F0

Timing Diagram

Instruction: **LXI H, Data**

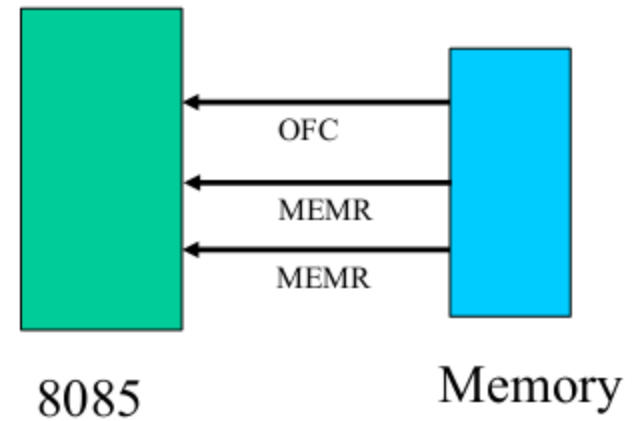
A000h LXI A,FO45h

Corresponding Coding:

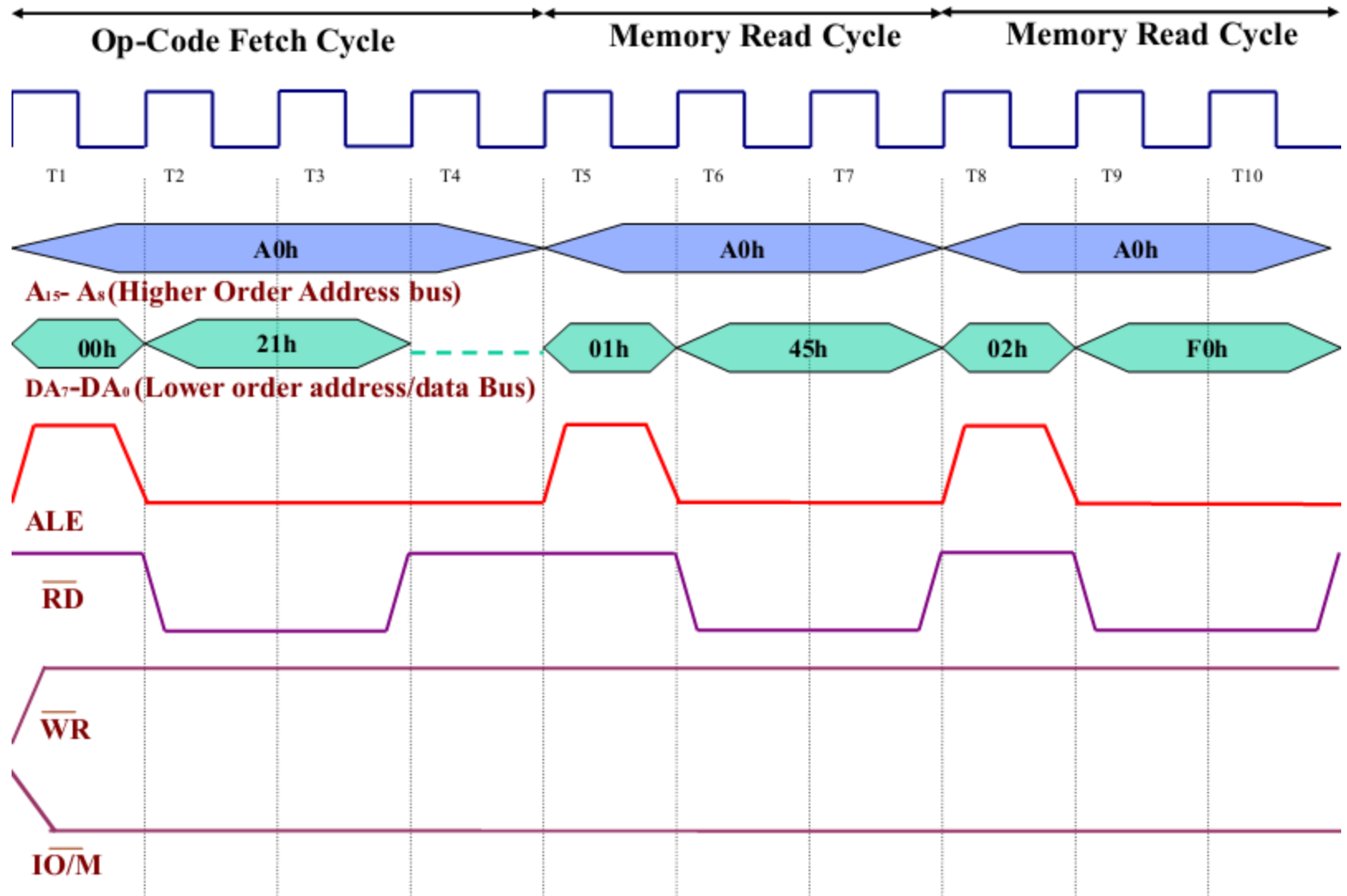
A000h 21

A001h 45

A002h F0



Timing Diagram



Timing Diagram

Instruction:

A000h MOV A,M

Corresponding Coding:

A000h 7E

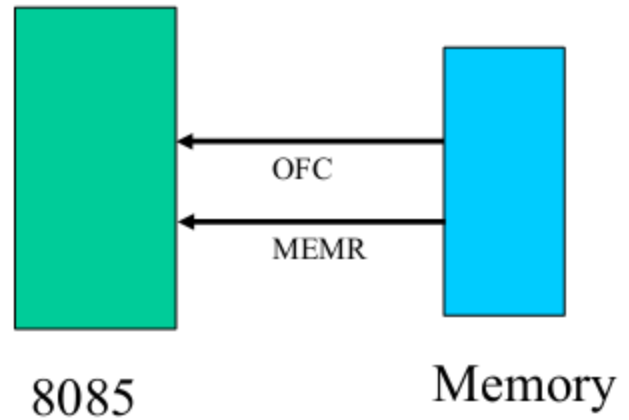
Timing Diagram

Instruction:

A000h MOV A,M

Corresponding Coding:

A000h 7E



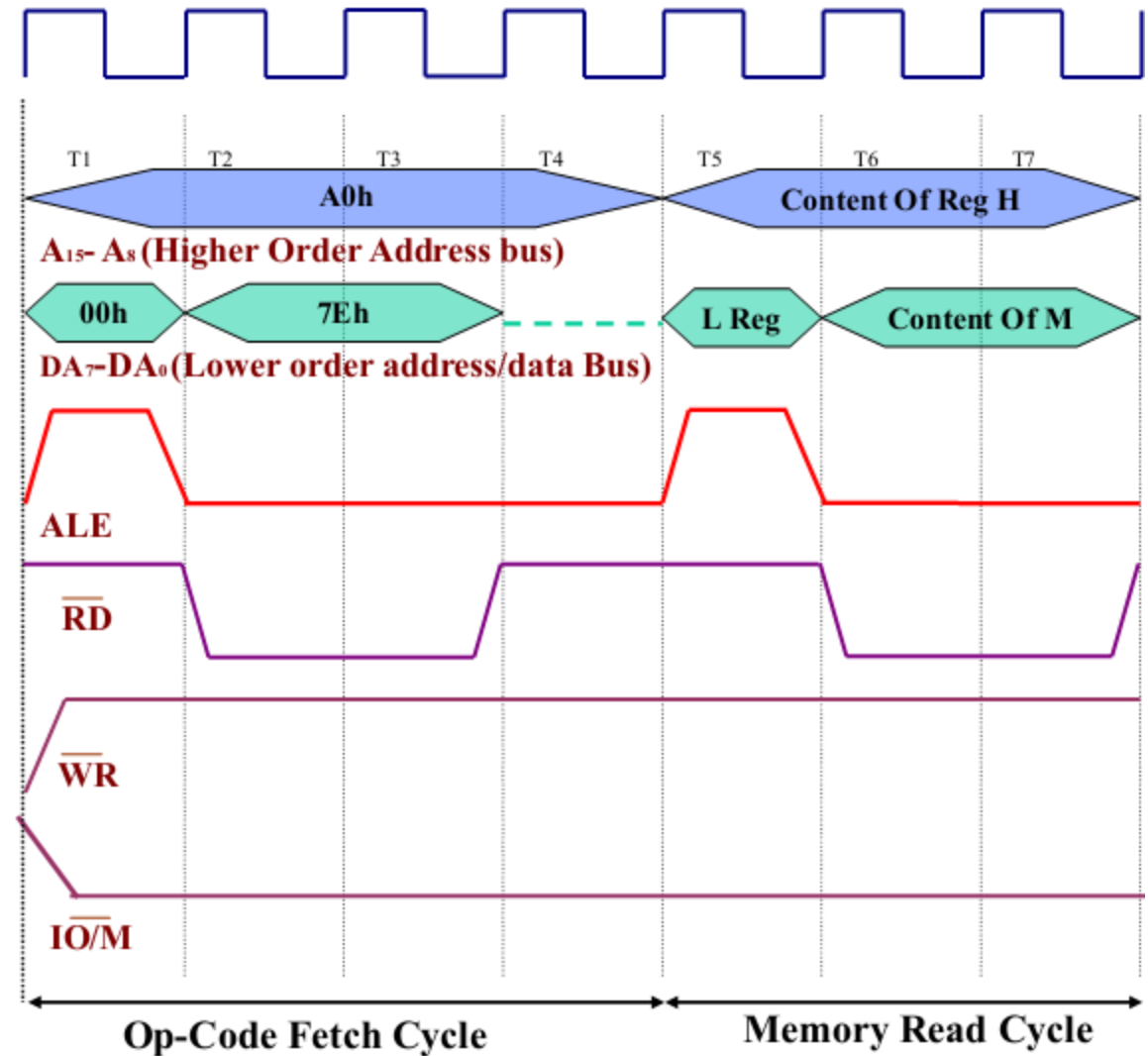
Timing Diagram

Instruction:

A000h MOV A,M

Corresponding Coding:

A000h 7E



Timing Diagram

Instruction:

A000h MOV M,A

Corresponding Coding:

A000h 77

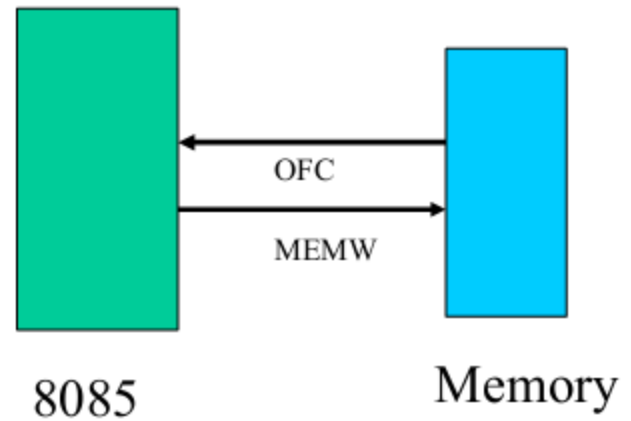
Timing Diagram

Instruction:

A000h MOV M,A

Corresponding Coding:

A000h 77



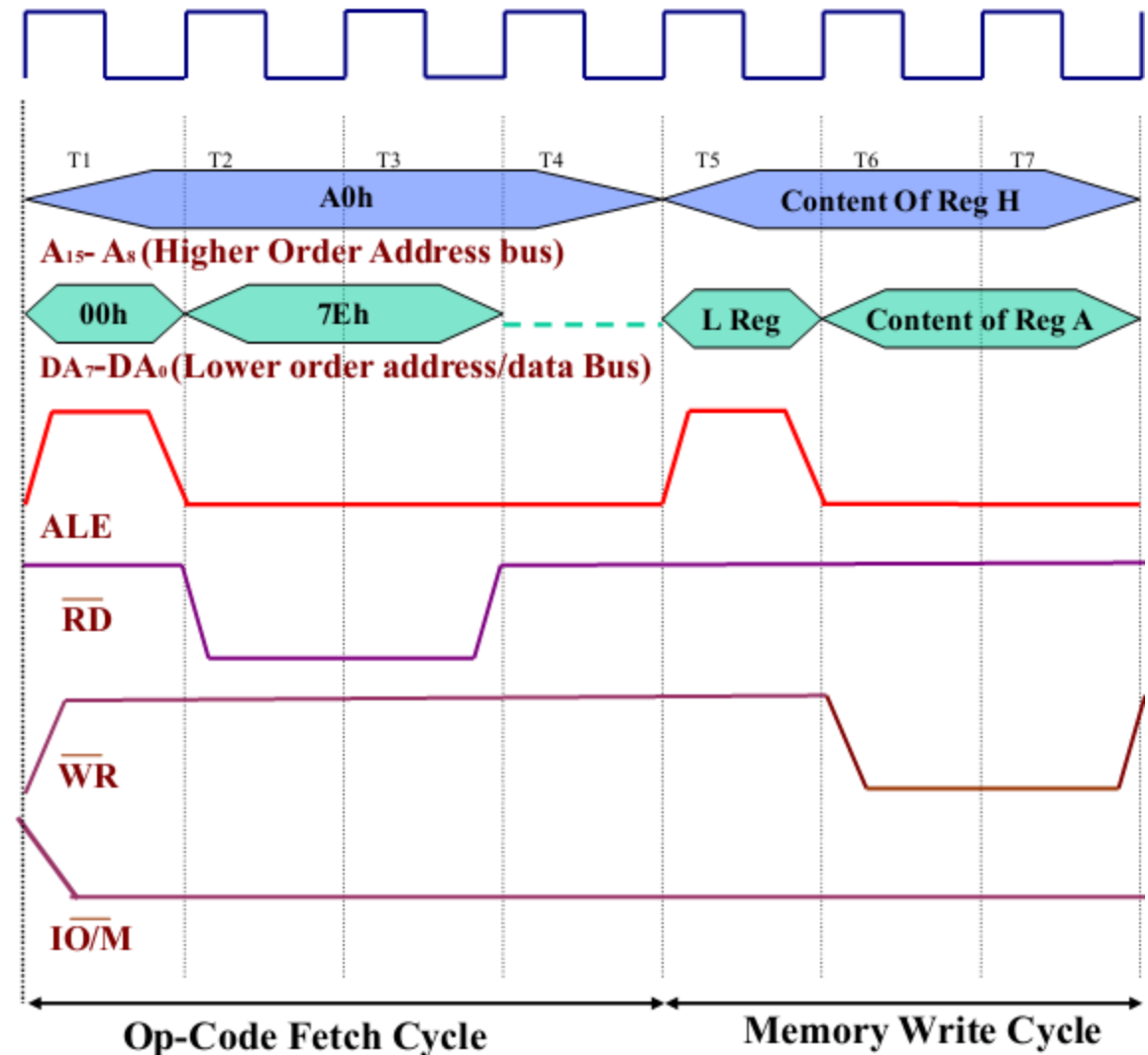
Timing Diagram

Instruction:

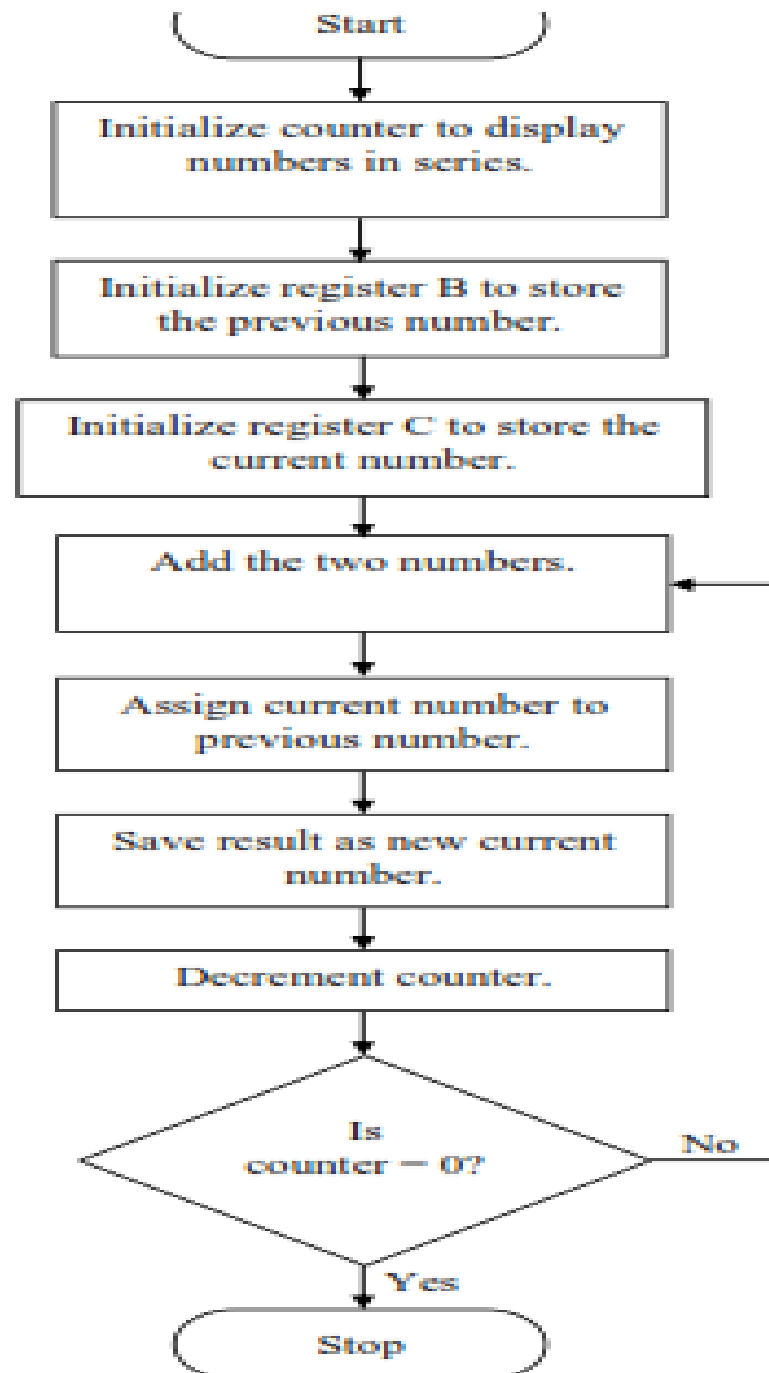
A000h MOV M,A

Corresponding Coding:

A000h 77



: Generation of Fibonacci series.



Address	Mnemonics	Operand	Opcode	Remarks
2000	MVI	D, 08H	16	Initialize counter to display numbers in series.
2001			08	Immediate value 07H.
2002	MVI	B, 00H	06	Initialize reg. B to store previous number.
2003			00	Immediate value 00H.
2004	MVI	C, 01H	0E	Initialize reg. C to store current number.
2005			01	Immediate value 01H.
2006	LXI	H, 3000H	21	Initialize H-L pair to point to memory.
2007			00	Lower-order of 3000H.
2008			30	Higher-order of 3000H.
2009	MOV	M, B	70	Move 00H from reg. B to memory.
200A	INX	H	23	Increment H-L pair.
200B	MOV	M, C	71	Move 01H from reg. C to memory.
200C	MOV	A, B	78	Move previous number from reg. B to reg. A.
200D	ADD	C	81	Add the two numbers.
200E	MOV	B, C	41	Assign current number to previous number.
200F	MOV	C, A	4F	Save result as new current number.
2010	INX	H	23	Increment H-L pair.
2011	MOV	M, A	77	Move number from reg. A to memory.
2012	DCR	D	15	Decrement counter.
2013	JNZ	200DH	C2	Jump to address 200DH if counter is not zero.
2014			0D	Lower-order of 200DH.
2015			20	Higher-order of 200DH.
2016	HLT		76	Halt.

Explanation:

- This program generates the Fibonacci series. The Fibonacci series is:

0 1 1 2 3 5 8 13 21 34

- In hexadecimal, it will be:

00 01 01 02 03 05 08 0D 15 22

- The first two numbers of the series are 0 and 1. The third number is computed as $0 + 1 = 1$, fourth number is $1 + 1 = 2$, fifth number is $1 + 2 = 3$ and so on.
- The count is initialized in register D to display the numbers in series.
- Initialize register B to first number 00H and register C to second number 01H.
- Initialize H-L pair to point to memory location 3000H.
- Move the first two numbers from registers B and C to memory locations 3000H and 3001H.
- Add the two numbers and store the result as first number.
- Increment H-L pair and move the result from accumulator to memory location.
- The next term is then computed by making the result equal to previous number.
- The process is repeated until all the numbers are calculated.

Output:**After Execution:**

3000H:	00H
3001H:	01H
3002H:	01H
3003H:	02H
3004H:	03H
3005H:	05H
3006H:	08H
3007H:	0DH
3008H:	15H
3009H:	22H