8085 Addressing Modes and Memory Interfacing

Contents

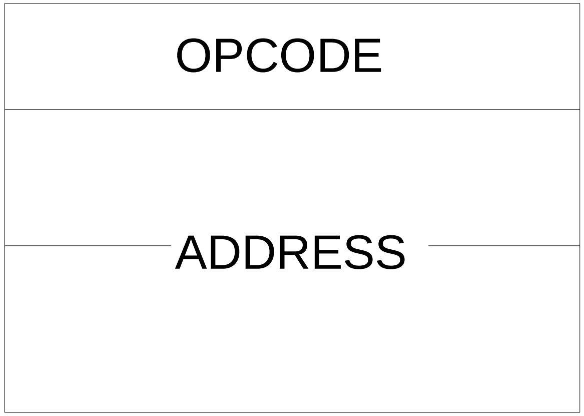
- Addressing Modes
- Timing diagram of Instruction Execution
- Instructions and machine cycle
- Memory Interfacing
- Incompatiblities
- Address space
  - Memory mapped IO
  - IO Mapped IO

Addressing Modes

- Each instruction has,
  - Operation to be performed
  - Address of source data
  - Address of destination data
- The method by which the address of source of data or the address of destination of result is given in the instruction is called Addressing Modes.
- Various addressing modes available in 8085 are:
  - Direct Addressing mode
  - Register Addressing mode
  - Register Indirect Addressing mode
  - Immediate Addressing mode
  - Implicit Addressing mode

Direct Addressing mode

- Here the address is provided as part of the instruction.
- Example: LDA 2500H
- LDA is the operation
- 2500H is the source
- Accumulator is the destination

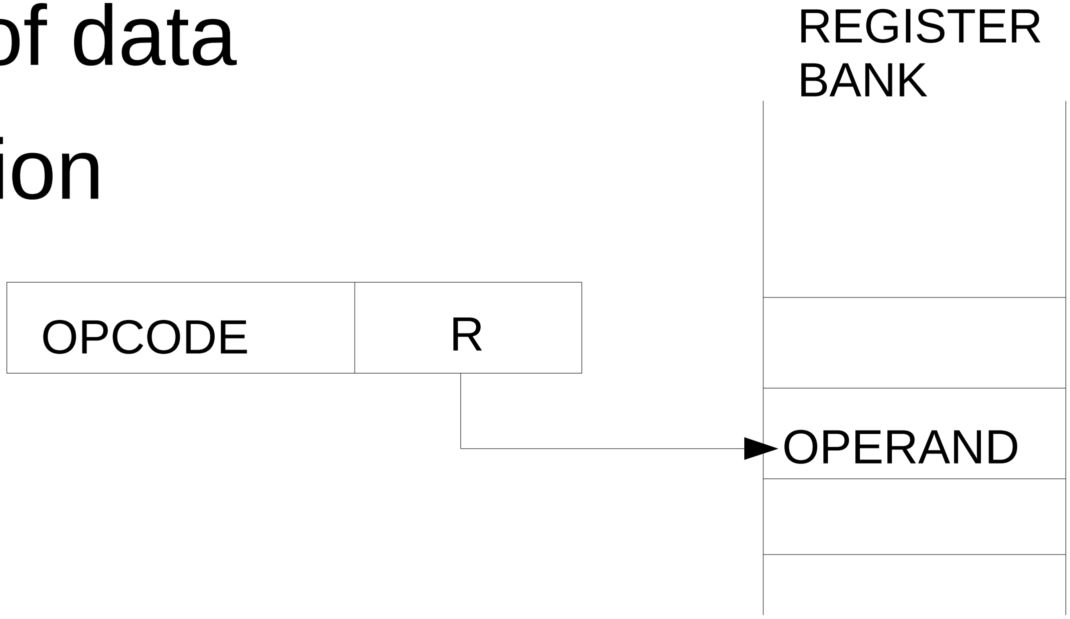


OPCODE

ADDRESS

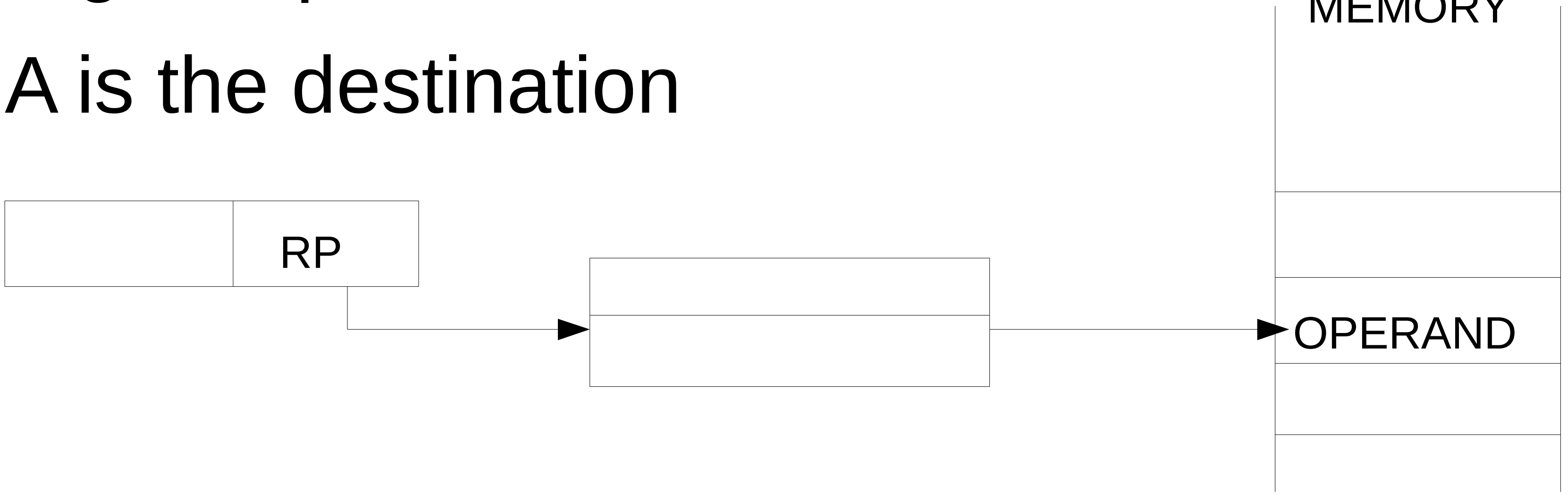
Register Addressing mode

- In this mode of addressing operand is a general purpose register
- Example: MOV A, B
- MOV is the operation
- B is the source of data
- A is the destination



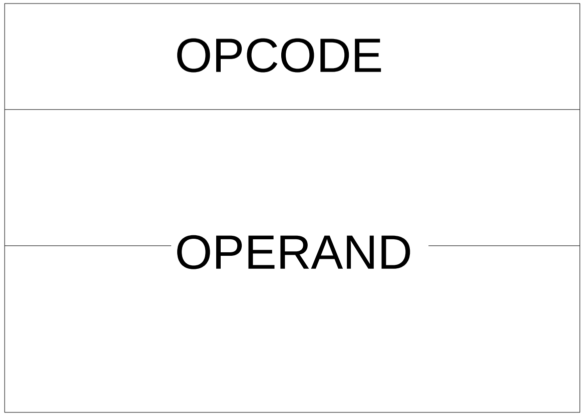
Register Indirect Addressing mode

- In this mode, the address of operand is specified by a register pair.
- Example: MOV A, M
- MOV is the operation
- M is the memory location specified by H-L register pair.
- A is the destination



Immediate Addressing mode

- In this mode, the operand is specified within the instruction itself.
- Example: MVI A, 07H
- MVI is the operation
- 05 H is the immediate data
- A is the destination
- Used to store constant data value or initialize register value

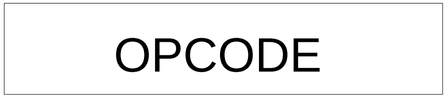


OPCODE

OPERAND

Implicit Addressing mode

- If address of source of data as well as address of destination of result is fixed, then there is no need to give any operand along with the instruction.
- Example: CMA
- CMA is the operation.
- A is the source.
- A is the destination.



OPCODE

Instruction Execution

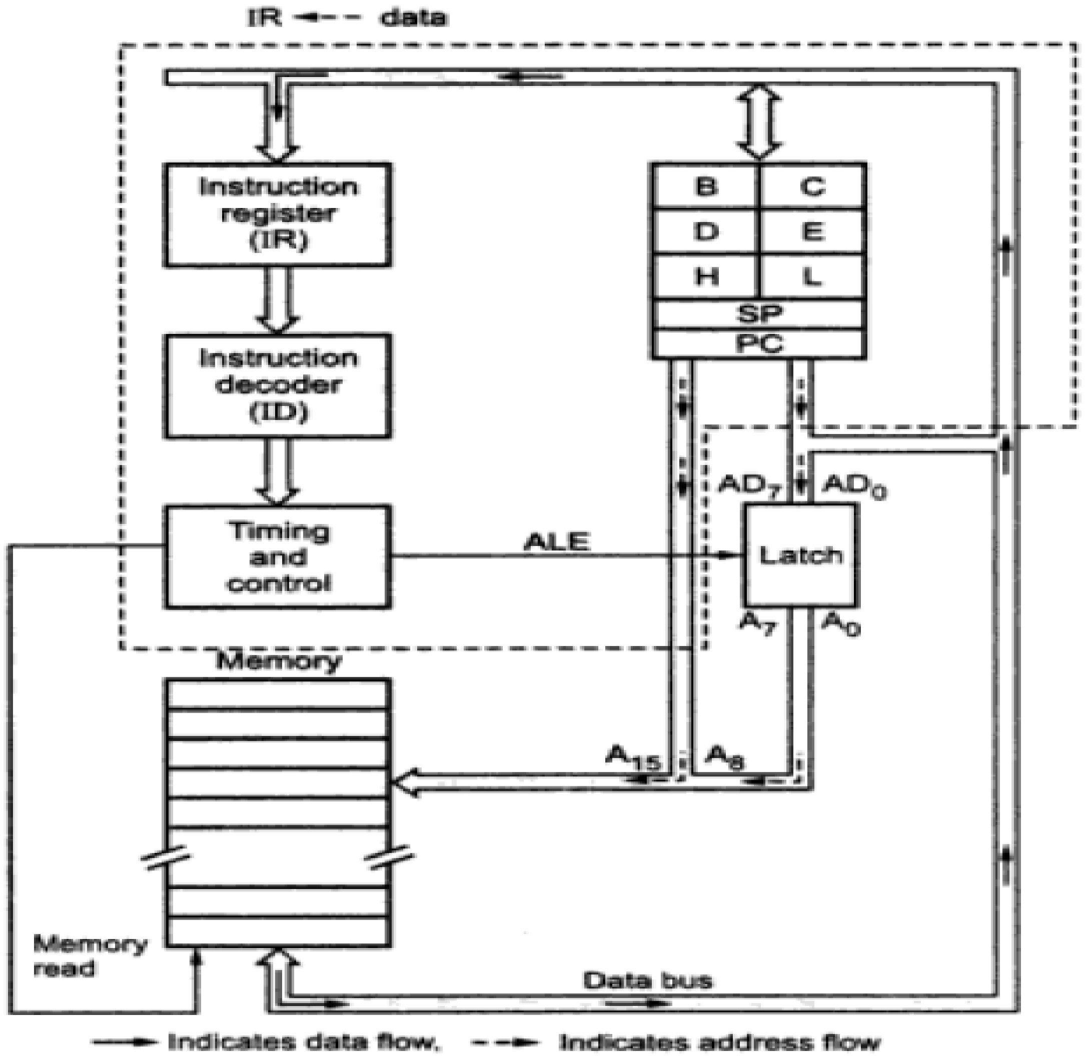
- Instruction execution involves
  - Fetch Cycle: In this cycle instruction to be executed is fetched from memory and decoded.
  - Execution Cycle: This cycle can be variable in time. Because it will involve one or more read cycle or one or more write cycle or combination of them.
- So, the total time taken for instruction execution is (Fetch cycle + Execution cycle) time.
- Address Latch Enable
  - Used to demultiplex address and data bus. When ALE=1, AD0-7 has address and when ALE=0, it has data.
- S1 and S0 (Status lines)
  - Specifies kind of operation to be performed.

S1	S0	operation
0	0	Halt
0	1	Write
1	0	Read
1	1	Fetch

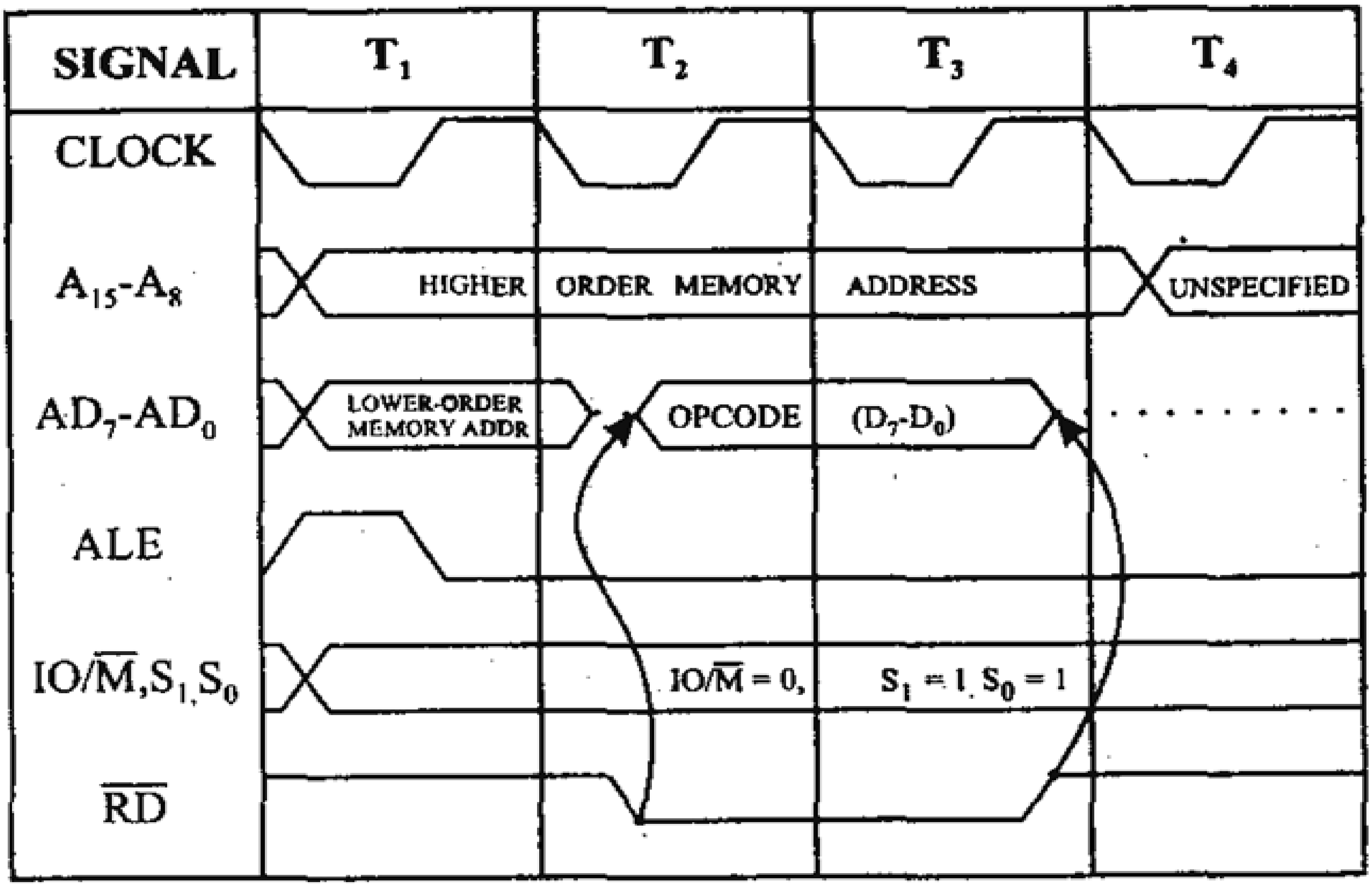
Fetch Cycle

- Program counter holds the address of next instruction to be fetched.
- Contents of PC are passed across to Address Bus.
- Depending on the address in the address bus, memory location of next instruction is located.
- Contents of the memory location are moved to the data bus and then to Instruction Register.
- This instruction is then decoded using instruction decoder and then control unit controls the operation.
- For instruction fetch IO/M=0, S0=1 and S1=1.
- Insruction fetch is done in 3 clock cycles and decoding is carried out in the 4th clock cycle.
- Some instruction which does not require Read/ Write from memory can be completed in 4 clock cycles (ADD B).
- 1 Machine Cycle = 4 states.

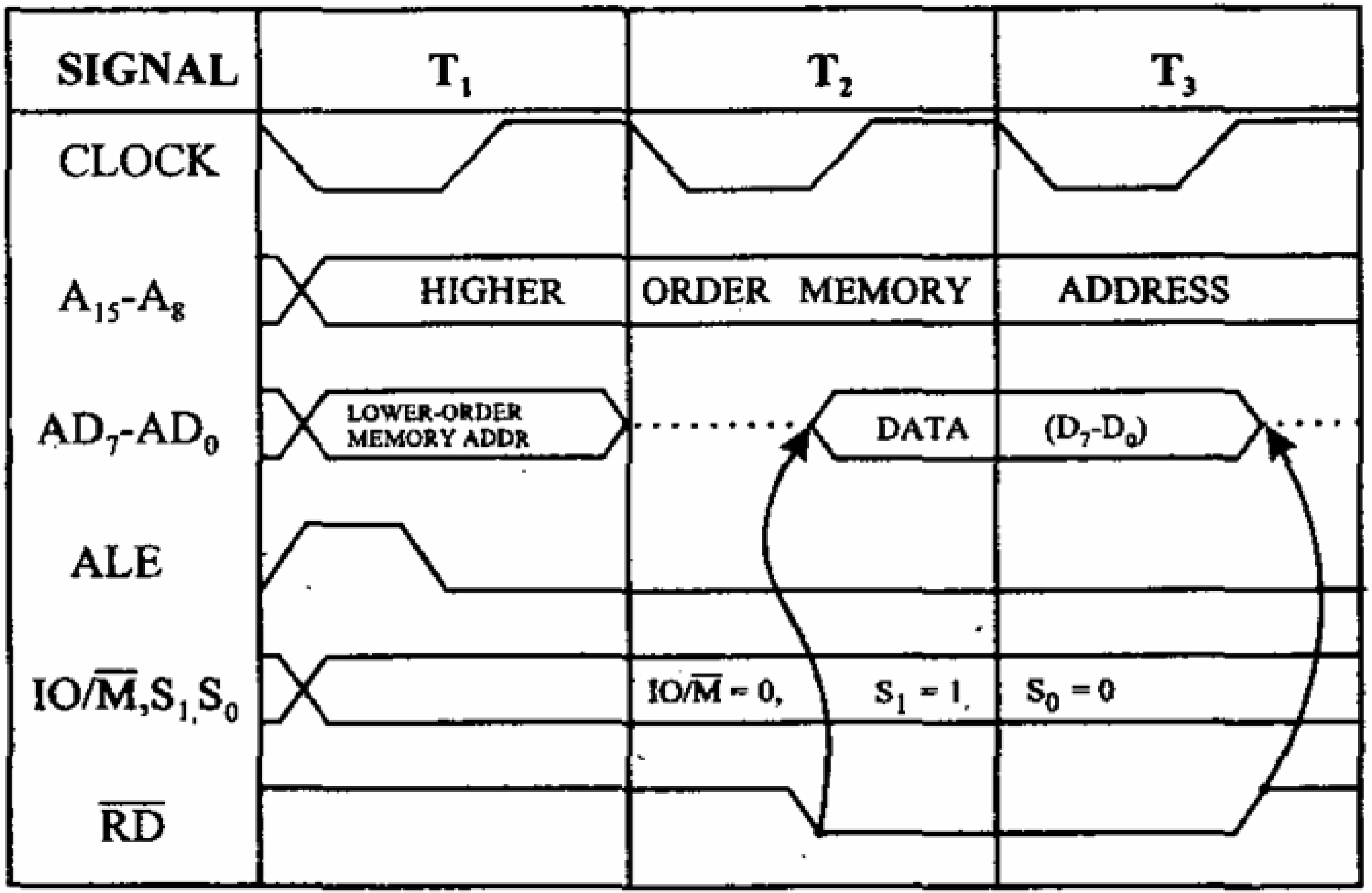
Fetch Cycle (continued)



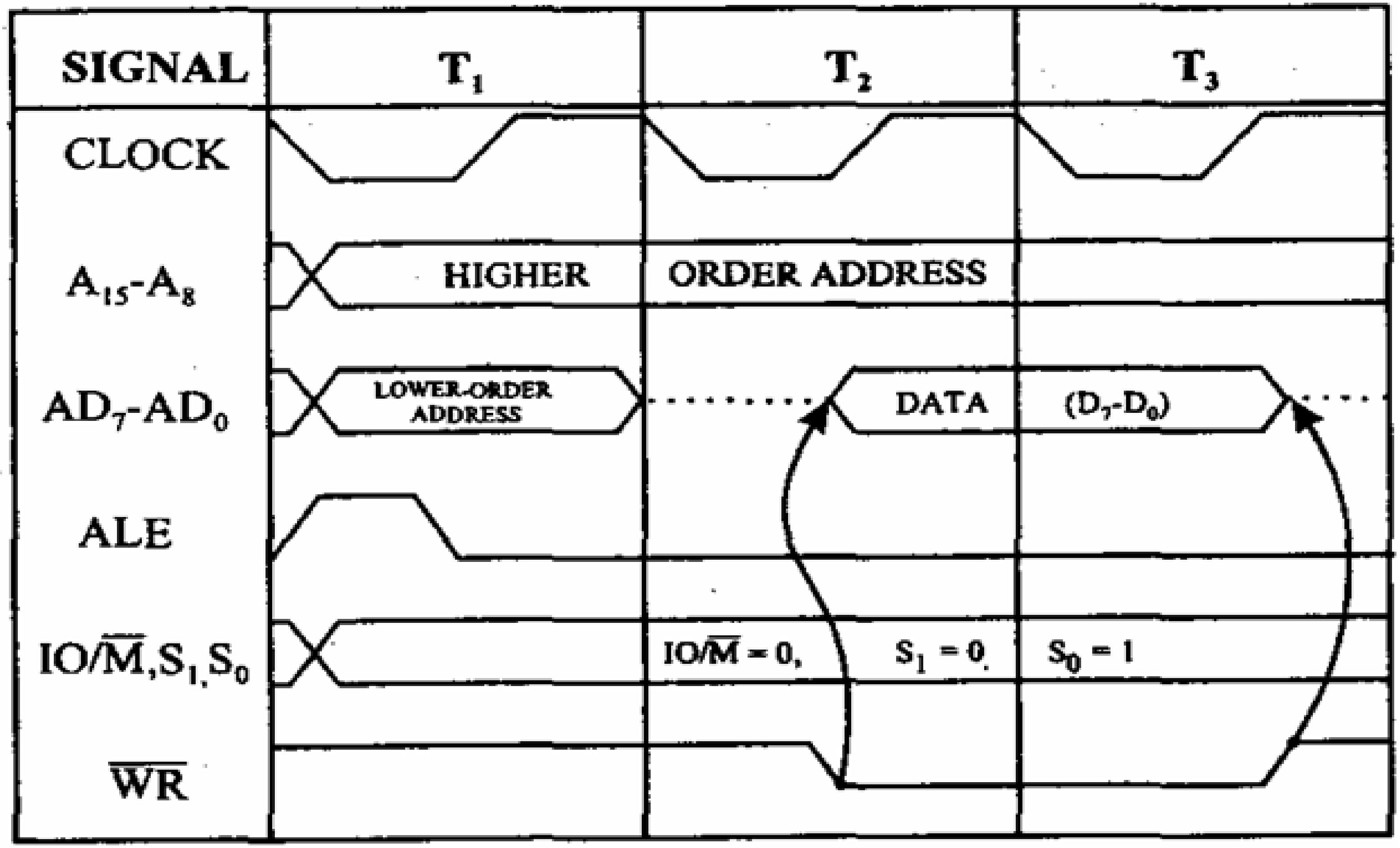
Fetch Cycle (continued)



Read cycle


SIGNAL
T_1
T_2
T_3
CLOCK
A_{15}-A_8
HIGHER
ORDER MEMORY
ADDRESS
AD_7-AD_0
LOWER-ORDER
MEMORY ADDR
DATA
(D_7-D_0)
ALE
IO/$\overline{M}$,S_1,S_0
IO/$\overline{M}$ = 0,
S_1 = 1,
S_0 = 0
$\overline{RD}$

Write Cycle


SIGNAL
T$_1$
T$_2$
T$_3$
CLOCK
A$_{15}$-A$_8$
HIGHER
ORDER ADDRESS
AD$_7$-AD$_0$
LOWER-ORDER ADDRESS
DATA
(D$_7$-D$_0$)
ALE
IO/$\overline{M}$,S$_1$,S$_0$
IO/$\overline{M}$ = 0,
S$_1$ = 0,
S$_0$ = 1
$\overline{WR}$

Instructions and Machine cycle

- MOV r1, r2
  - Number of cycles needed = 1
  - Number of states needed = 4
- MVI r, data
  - Number of cycles needed = 2
  - Number of states needed = 7
- LXI rp, data
  - Number of cycles needed = ?
  - Number of states needed = ?
- JMP addr
- LDA aadr
- STA addr
- Condtional Jump
- Subroutine Call
- Conditional subroutines
- Software Interrupts

Memory Interfacing

- Memory has following pins
  - Address lines
  - Data line(s)
  - Power supply
  - Ground line
  - Chip enable
  - Read/ write control

IO/M	RD	WR	Control Signal	Oper ation
0	0	1	MEMR	M/M Read
0	1	0	MEMW	M/M write
1	0	1	IOR	I/O Read
1	1	0	IOW	I/O Write

- Depending on whether the data line in memory is single line or 8 lines, memory is called bit organized or byte organized. Since, the number of pins required for bit organized memory less cost of this kind of memory is less.

Incompatibilities with memory

- The signals generated by CPU and signals required by memroy may not be the same. This leads to incompatibilities.
- The different types of incompatibilites are
  - Bus incompatibility: The address and data are multiplexed in 8085. Whereas, memory needs separate address and data. To accomplish this latch is used which separates address and data with the help of ALE signal.
  - Electrical incomptibilty: This can be either volatge or current (beacuse microprocessor can drive a lot of chip). To solve this issue, unidirectional and bidirectional drivers are used.
  - Timing incompatibility: Electronic circuits come with delay. So, get valid data, the signals have to held in its state for certain amount of time. If the memory is not accessed within specified time then the data becomes invalid.
- To address the issue of interfacing with slow memory, wait cycles are introduced. Wait cycle is introduced if the ready signal is low. If ready signal is low then it means that the data from memory is still not ready.

Address space

- Number of address lines specifies the main memory that can be interfaced.
- IO also has to be interfaced with memory. This interfacing can be done in two ways.
- Memory Mapped IO
  - Part of the address space is alocated for IO device which enables IO to be accessed as memory location. Hence, no separate instruction is required to access IO device. All ALU operation can be done directly with IO.
  - Entire address space is not available for memory because part of it is used by IO
- IO Mapped IO
  - IO/M is used to specify whether the address is for memory or IO. 8-bit address is used to map IO device. Hence, lower order address is copied to higher order address.
  - An extra pin is necessary to distiguish between IO and memory. Dedicated instructions are required for this mapping (IN, OUT). ALU operation cannot be direclty perfomed with IO. Hence, data has to be first trasfered to accumulator.