Instruction Set of 8085

- Consists of
  - 74 operation codes, e.g. MOV, MVI
  - 246 Instructions, e.g. MOV A,B, MVI A,03
- 8085 instructions can be classified as
  1. Data Transfer (Copy)
  2. Arithmetic
  3. Logical and Bit manipulation
  4. Branch
  5. I/O stack and Machine Control

Notation used in instructions

Table 2

Notations	Meaning
M	Memory location pointed by HL register pair
r	8-bit register
R_p	16-bit register pair
R_S	Source register
R_d	Destination register
Addr	16-bit address
Data	8-bit data
data 16-bit	16-bit data

Figure 1[1] Notation used in the instruction set

Data Transfer Instructions

- These instructions move data between registers, or between memory and registers.
- These instructions copy data from source to destination.
- While copying, the contents of source are not modified.

Data Transfer Instructions

1

Opcode	Operand	Description
MOV	Rd, Rs M, Rs Rd, M	Copy from source to destination.

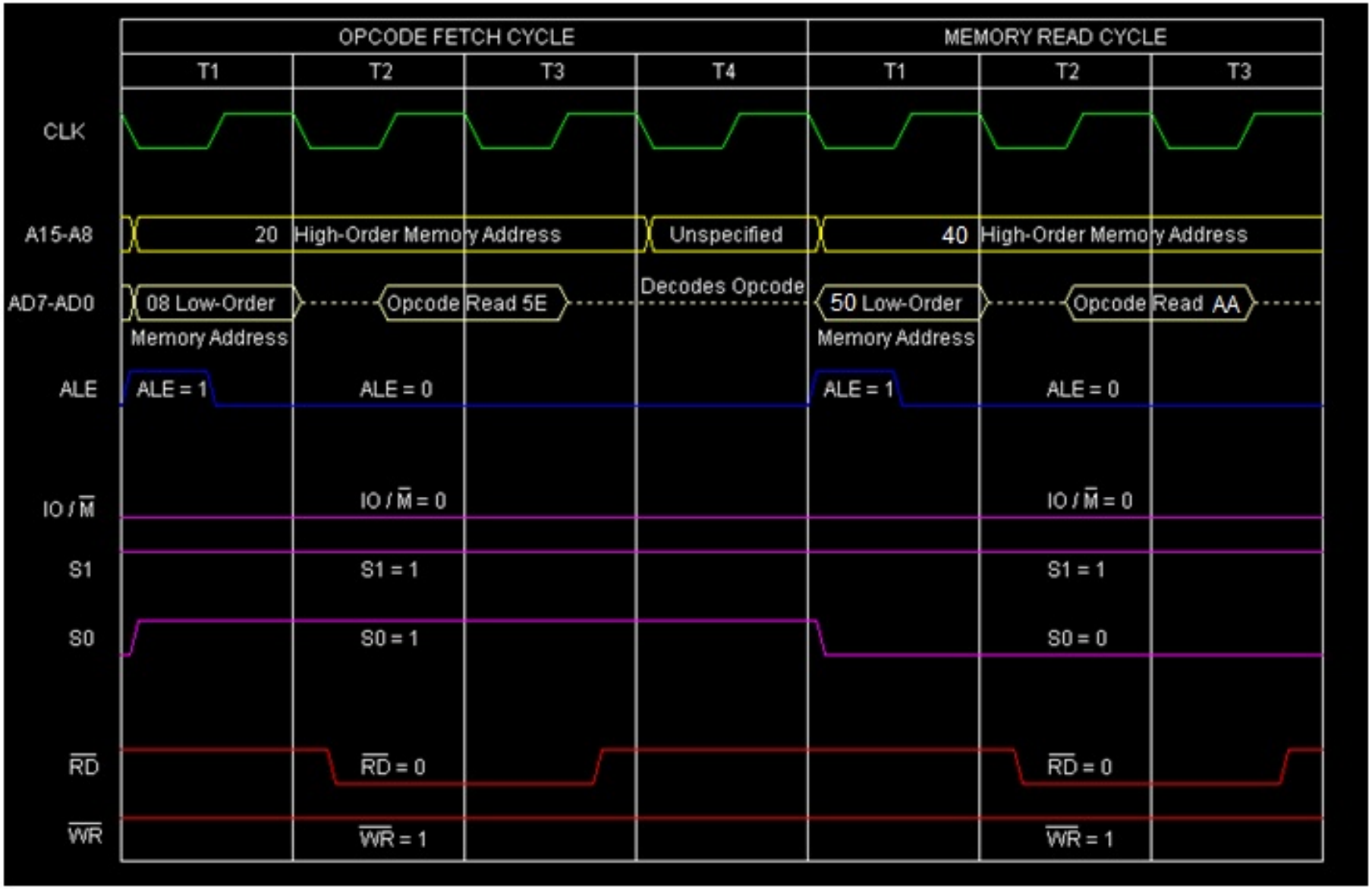
- This instruction copies the contents of the source register into the destination register.
- The contents of the source register are not altered.
- If one of the operands is a memory location, its location is specified by the contents of the HL registers.
- **Example:** MOV B, C or MOV B, M

MOV E, M

	Before	After
(E)	DBH	AAH
(HL)	4050H	4050H
(4050H)	AAH	AAH

Address	Hex Codes	Mnemonic	Comment
2008	2A	MOV E, M	Comment E <- Content of the memory location pointed by HL register pair

1-Byte


OPCODE FETCH CYCLE
MEMORY READ CYCLE
T1
T2
T3
T4
T1
T2
T3
CLK
A15-A8
20
High-Order Memory Address
Unspecified
40
High-Order Memory Address
AD7-AD0
08 Low-Order
Memory Address
Opcode Read 5E
Decodes Opcode
50 Low-Order
Memory Address
Opcode Read AA
ALE
ALE = 1
ALE = 0
ALE = 1
ALE = 0
IO / M
IO / M = 0
IO / M = 0
S1
S1 = 1
S1 = 1
S0
S0 = 1
S0 = 0
RD
RD = 0
RD = 0
WR
WR = 1
WR = 1

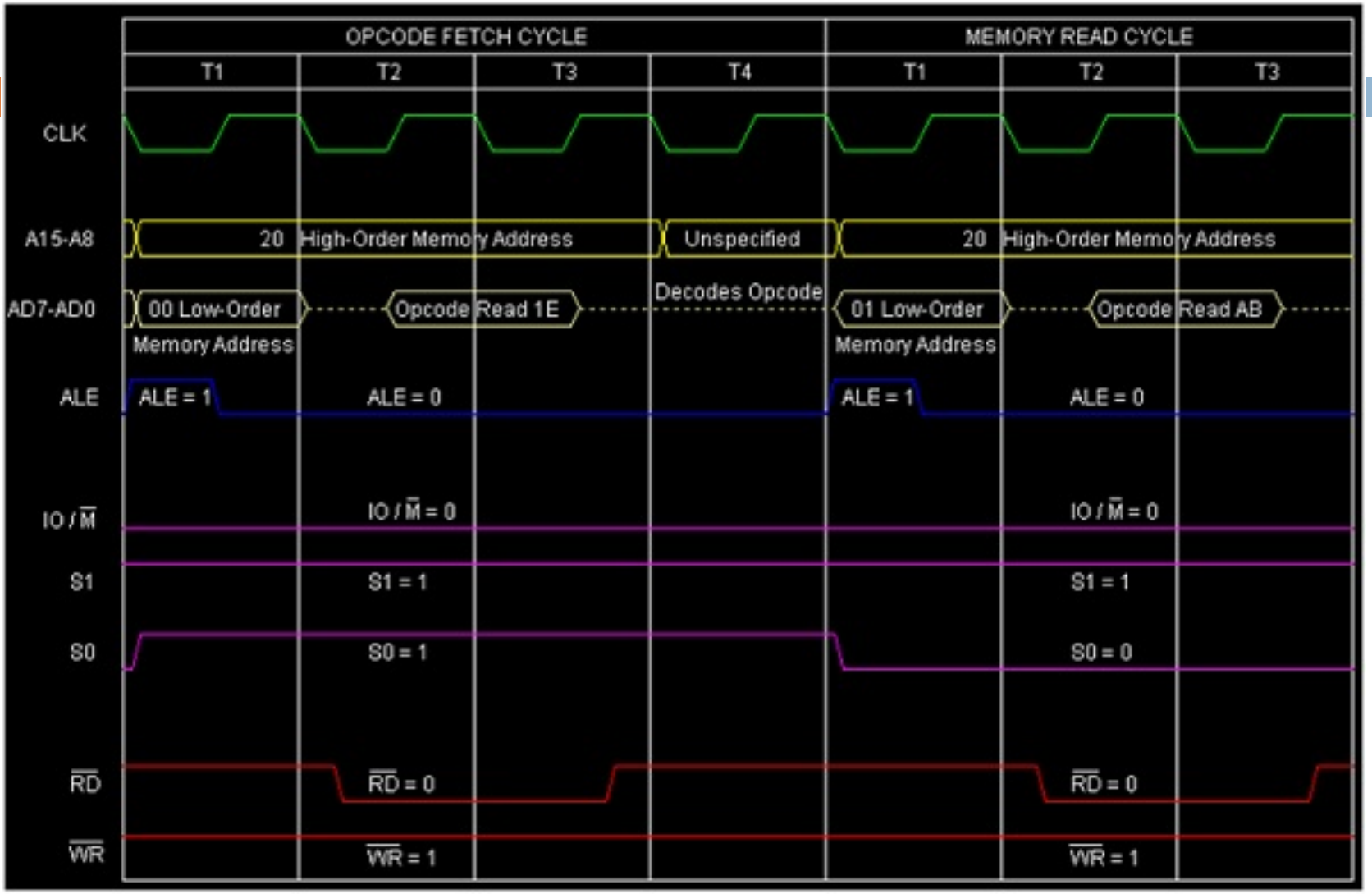
Data Transfer Instructions

Opcode	Operand	Description
MVI	Rd, Data M, Data	Move immediate 8-bit

- The 8-bit data is stored in the destination register or memory.
- If the operand is a memory location, its location is specified by the contents of the H-L registers.
- **Example:** MVI B, 57H or MVI M, 57H

MVI E, ABH

Address	Hex Codes	Mnemonic	Comment
2000	1E	MVI E, ABH	E <- ABH
2001	AB		ABH as operand



So this instruction **MVI E, ABH** requires 2-Bytes, 2-Machine Cycles (Opcode Fetch and Memory Read) and 7 T-States for execution as shown in the timing diagram.

Data Transfer Instructions

1

Opcode	Operand	Description
LDA	16-bit address	Load Accumulator

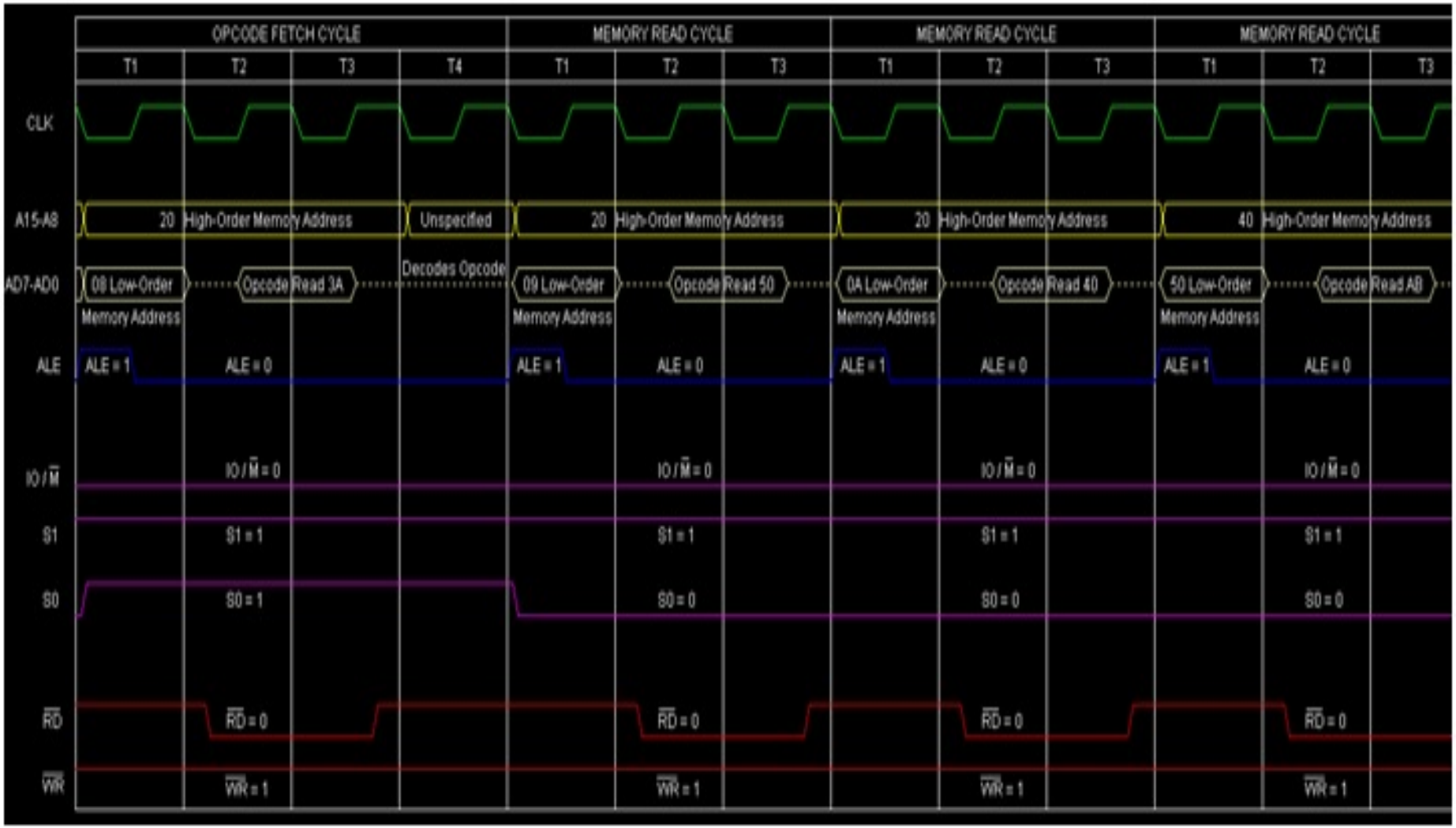
- The contents of a memory location, specified by a 16-bit address in the operand, are copied to the accumulator.
- The contents of the source are not altered.
- **Example:** LDA 2034H

LDA 4050H

3-Byte instruction.

	Before	**After**
(4050)	ABH	ABH
A	CDH	ABH

Address	**Hex Codes**	**Mnemonic**	**Comment**
2008	3A	LDA 4050H	A <- Content of the memory location 4050H
2009	50		Low order Byte of the address
200A	40		High order Byte of the address



So this instruction **LDA 4050H** requires 3-Bytes, 4-Machine Cycles (Opcode Fetch, Memory Read, Memory Read, Memory Read) and 13 T-States for execution as shown in the timing diagram.

Data Transfer Instructions

1

Opcode	Operand	Description
LDAX	B/D Register Pair	Load accumulator indirect

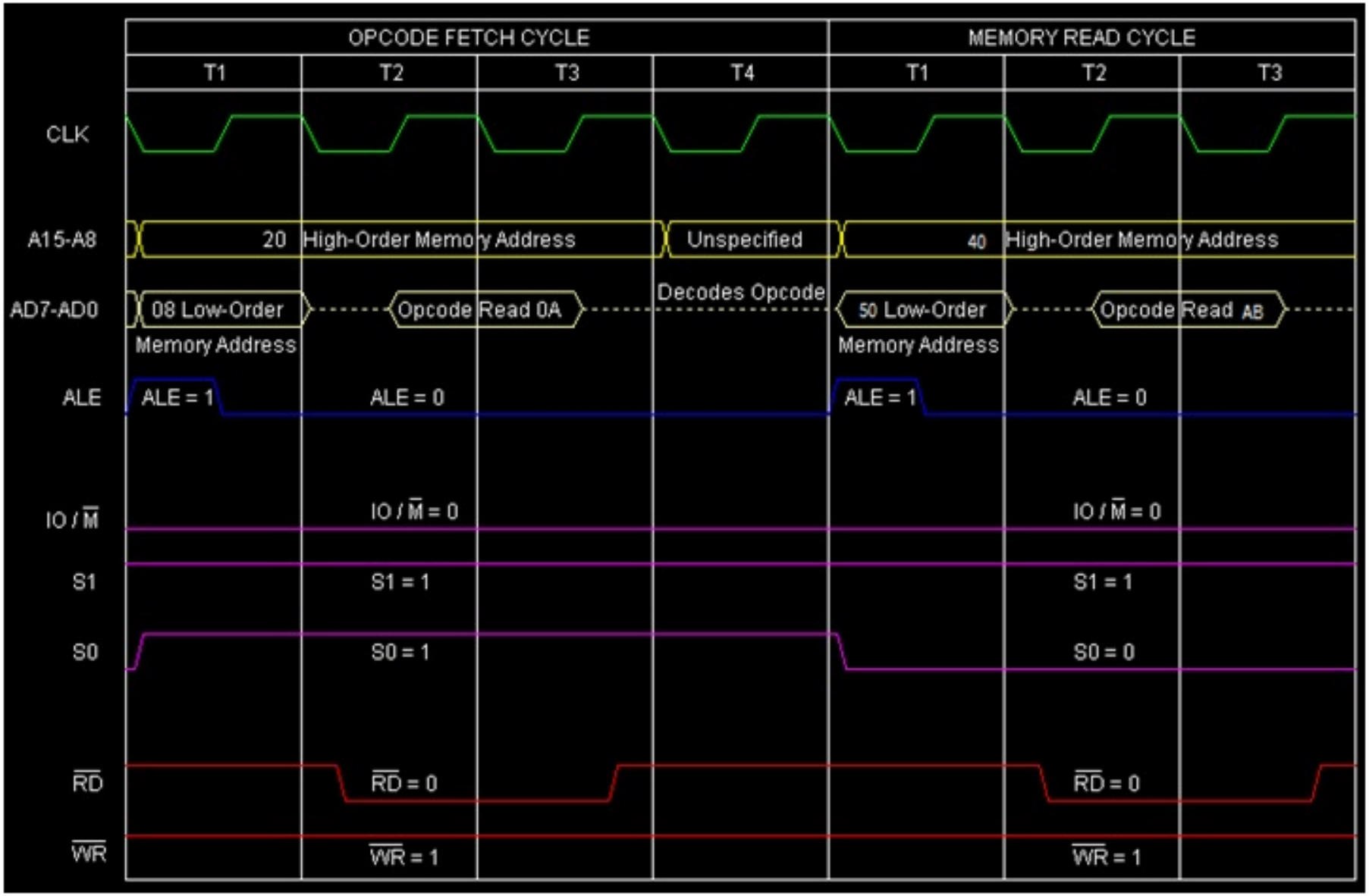
- The contents of the designated register pair point to a memory location.
- This instruction copies the contents of that memory location into the accumulator.
- The contents of either the register pair or the memory location are not altered.
- **Example:** LDAX B

LDAX B

	Before	After
(BC)	4050H	4050H
(4050H)	ABH	ABH
A	CDH	ABH

1-Byte instruction

Address	Hex Codes	Mnemonic	Comment
2008	0A	LDAX B	A <- Content of the memory 4050H



So this instruction **LDAX B** requires 1-Byte, 2-Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

Data Transfer Instructions

1

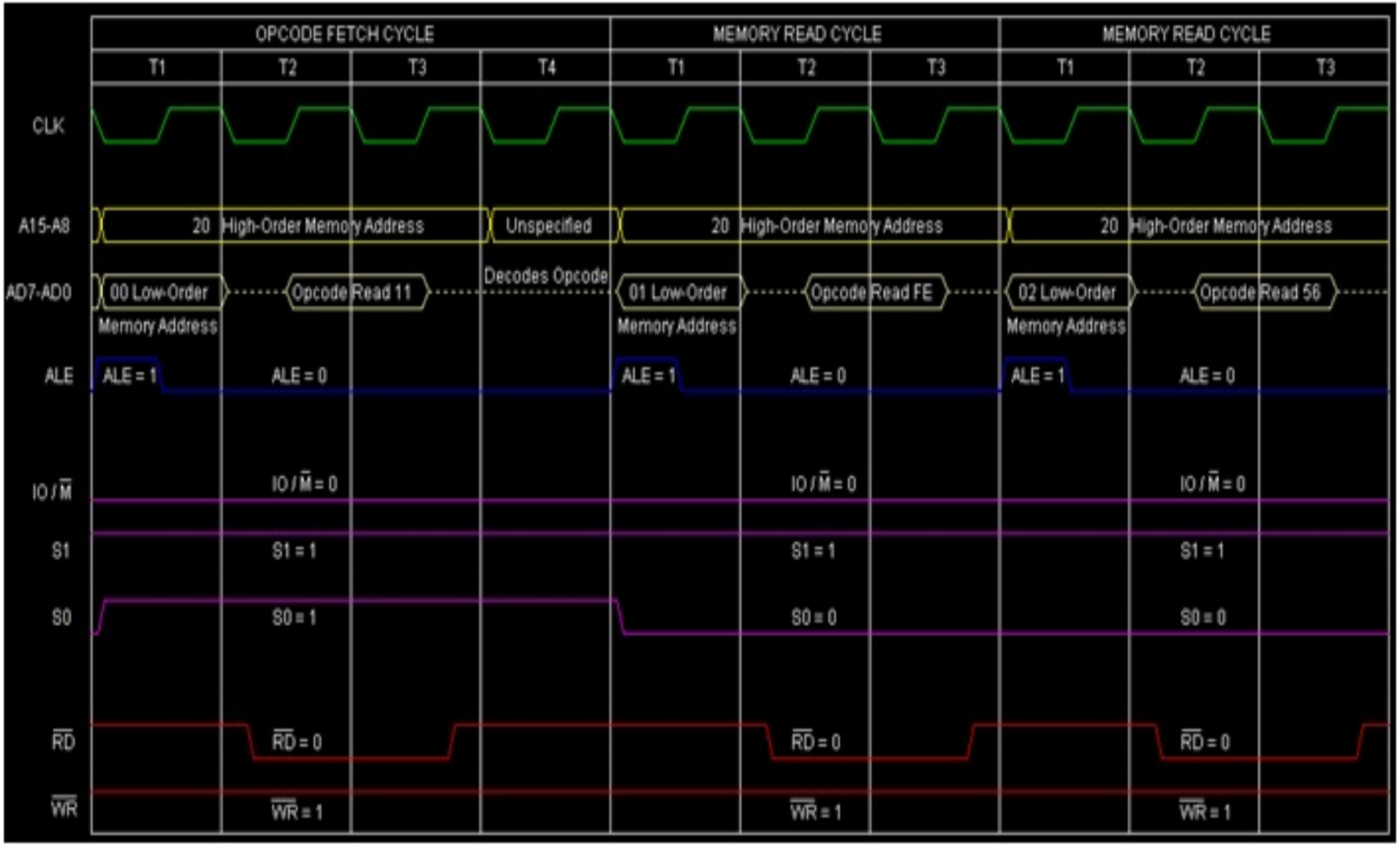
Opcode	Operand	Description
LXI	Reg. pair, 16-bit data	Load register pair immediate

- This instruction loads 16-bit data in the register pair.
- **Example:** LXI H, 2034 H

LXI D, 56FEH 56FEH is loaded into the DE register pair.

Address	Hex Codes	Mnemonic	Comment
8000	11		Store 56FEH
8001	FE	LXI D, 56FEH	into the DE
8002	56		Register Pair.

3-Bytes instruction


OPCODE FETCH CYCLE
MEMORY READ CYCLE
MEMORY READ CYCLE
T1
T2
T3
T4
T1
T2
T3
T1
T2
T3
CLK
A15-A8
20
High-Order Memory Address
Unspecified
20
High-Order Memory Address
20
High-Order Memory Address
AD7-AD0
00 Low-Order
Memory Address
Opcode Read 11
Decodes Opcode
01 Low-Order
Memory Address
Opcode Read FE
02 Low-Order
Memory Address
Opcode Read 56
ALE
ALE = 1
ALE = 0
ALE = 1
ALE = 0
ALE = 1
ALE = 0
IO / M
IO / M = 0
IO / M = 0
IO / M = 0
S1
S1 = 1
S1 = 1
S1 = 1
S0
S0 = 1
S0 = 0
S0 = 0
RD
RD = 0
RD = 0
RD = 0
WR
WR = 1
WR = 1
WR = 1

Data Transfer Instructions

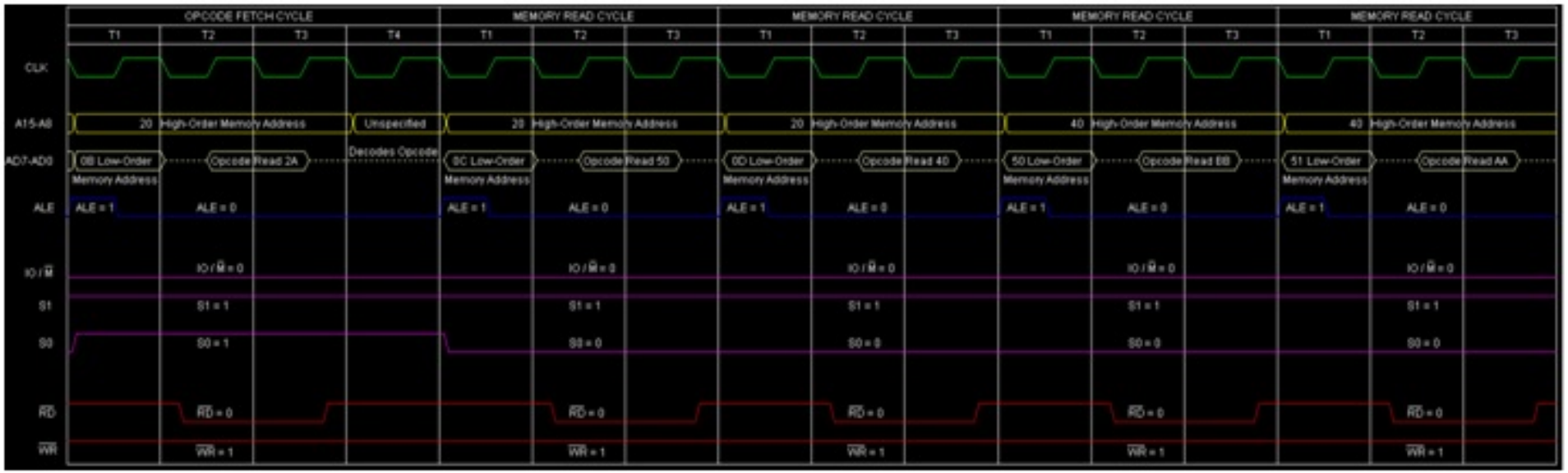
Opcode	Operand	Description
LHLD	16-bit address	Load H-L registers direct

- This instruction copies the contents of memory location pointed out by 16-bit address into register L.
- It copies the contents of next memory location into register H.
- **Example:** LHLD 2040 H

LHLD 4050H

	Before	After
(4050H)	BBH	BBH
(4051H)	AAH	AAH
(H)	CCH	AAH
(L)	DDH	BBH

ddress	Hex Codes	Mnemonic	Comment
200B	2A	LHLD 4050H	Initialize HL register pair from 4050H and 4051H memory locations' contents.
200C	50		Low order Byte of the address
200D	40		High order Byte of the address



So this instruction **LHLD 4050H** requires 3-Bytes, 5-Machine Cycles (Opcode Fetch, Memory Read, Memory Read, Memory Read, Memory Read) and 16 T-States for execution as shown in the timing diagram.

Data Transfer Instructions

Opcode	Operand	Description
STA	16-bit address	Store accumulator direct

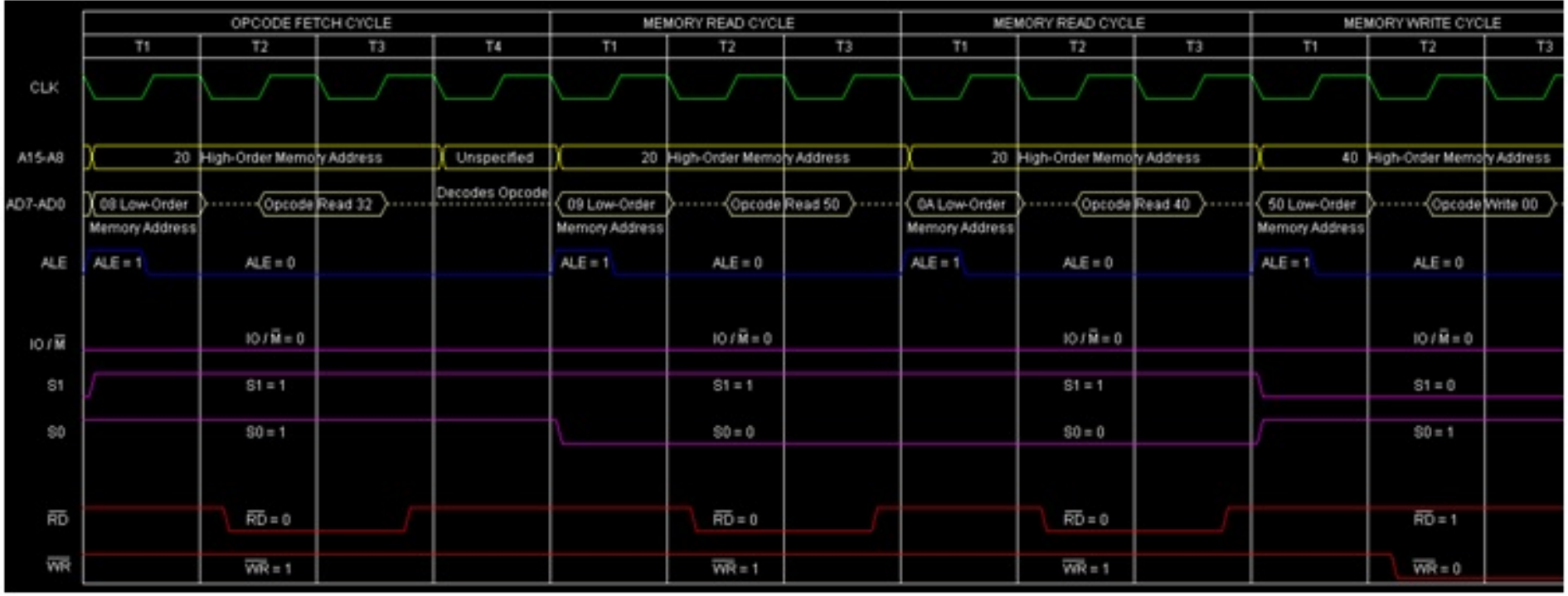
- The contents of accumulator are copied into the memory location specified by the operand.
- **Example:** STA 2500 H

STA 4050H

	Before	After
(A)	ABH	ABH
(4050H)	CDH	ABH

Address	Hex Codes	Mnemonic	Comment
2008	2A	STA 4050H	Content of the memory location 4050H <- A
2009	50		Low order Byte of the address
200A	40		High order Byte of the address

3-Byte instruction.



So this instruction **SDA 4050H** requires 3-Bytes, 4-Machine Cycles (Opcode Fetch, Memory Read, Memory Read, Memory Write) and 13 T-States for execution as shown in the timing diagram.

Data Transfer Instructions

Opcode	Operand	Description
STAX	Reg. pair	Store accumulator indirect

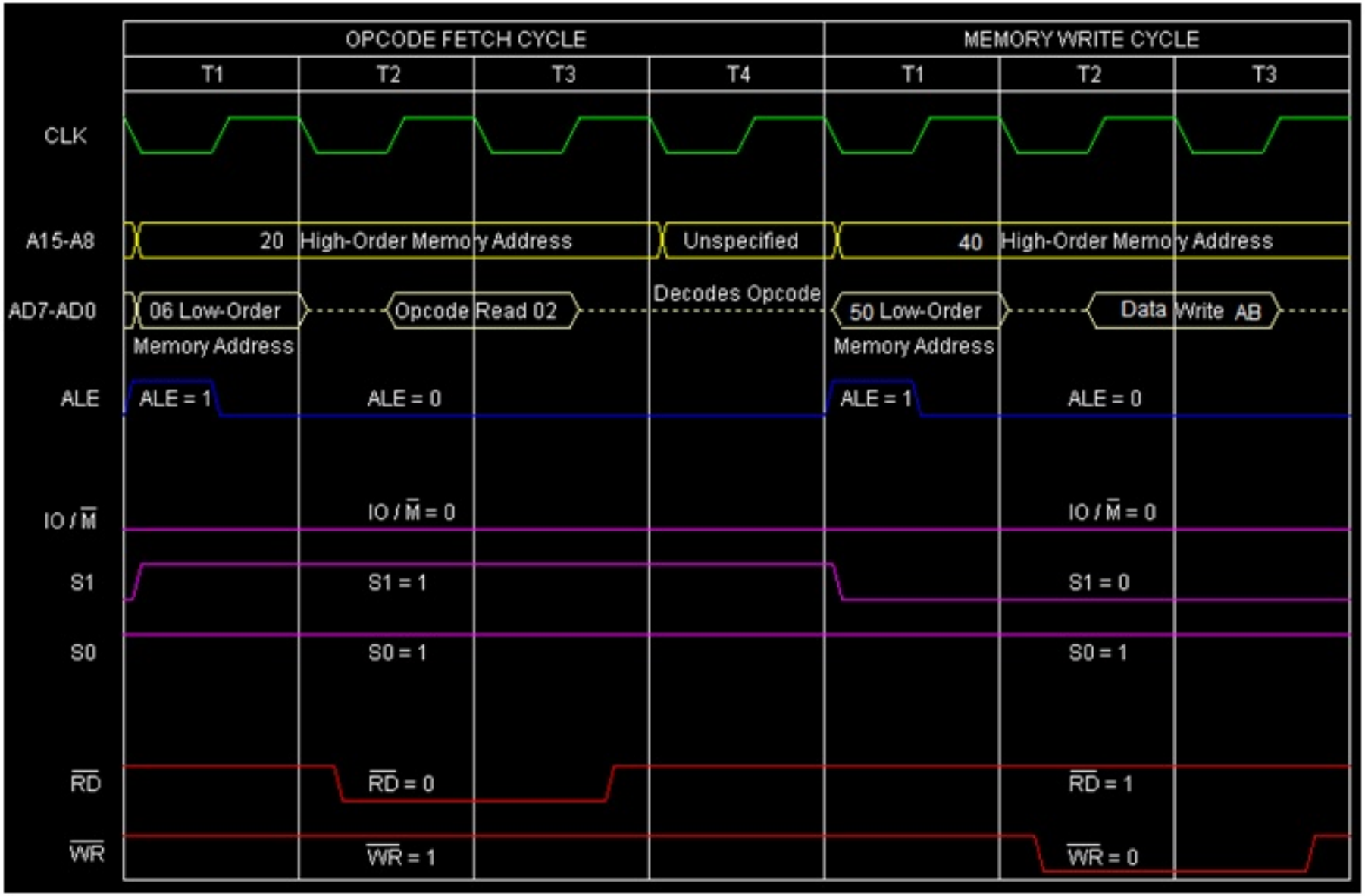
- The contents of accumulator are copied into the memory location specified by the contents of the register pair.
- **Example:** STAX B

STAX B

	Before	After
(BC)	4050H	4050H
(A)	ABH	ABH
(4050h)	CDH	ABH

Address	Hex Codes	Mnemonic	Comment
2006	02	STAX B	Content of the memory location pointer by BC register pair <- A

1-Byte instruction.


OPCODE FETCH CYCLE
MEMORY WRITE CYCLE
T1
T2
T3
T4
T1
T2
T3
CLK
A15-A8
20
High-Order Memory Address
Unspecified
40
High-Order Memory Address
AD7-AD0
06 Low-Order
Memory Address
Opcode Read 02
Decodes Opcode
50 Low-Order
Memory Address
Data Write AB
ALE
ALE = 1
ALE = 0
ALE = 1
ALE = 0
IO / M
IO / M = 0
IO / M = 0
S1
S1 = 1
S1 = 0
S0
S0 = 1
S0 = 1
RD
RD = 0
RD = 1
WR
WR = 1
WR = 0

Data Transfer Instructions

2

Opcode	Operand	Description
SHLD	16-bit address	Store H-L registers direct

- The contents of register L are stored into memory location specified by the 16-bit address.
- The contents of register H are stored into the next memory location.
- **Example:** SHLD 2550 H

SHLD 4050H

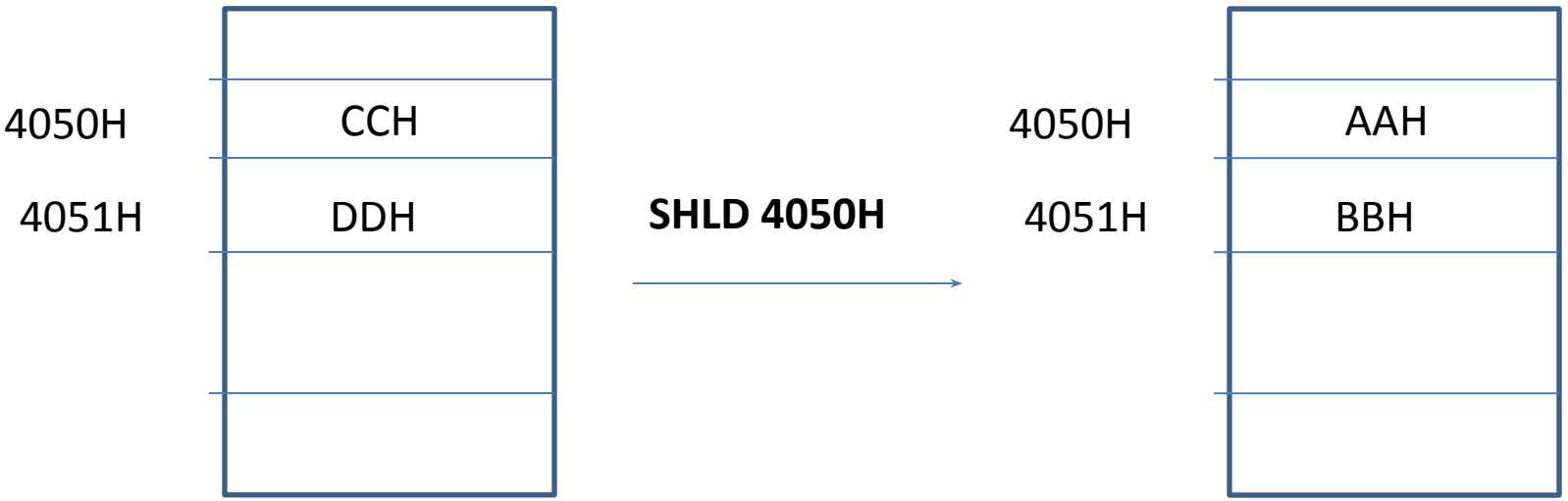
	Before	After
(H)	BBH	BBH
(L)	AAH	AAH
(4050H)	CCH	AAH
(4051H)	DDH	BBH

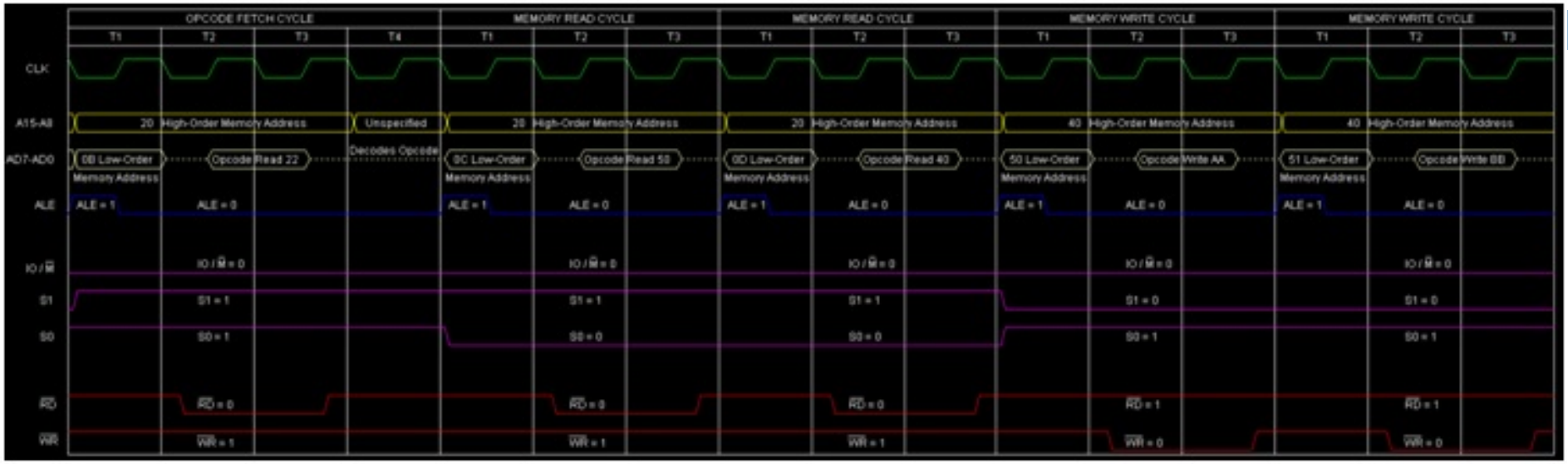
Address	Hex Codes	Mnemonic	Comment
200B	22	SHLD 4050H	Store HL register pair's content to memory locations 4050H and 4051H respectively.
200C	50		Low order Byte of the address
200D	40		High order Byte of the address

3-Byte instruction

SHLD 4050H

	Before	After
(H)	BBH	BBH
(L)	AAH	AAH
(4050H)	CCH	AAH
(4051H)	DDH	BBH





So this instruction **SHLD 4050H** requires 3-Bytes, 5-Machine Cycles (Opcode Fetch, Memory Read, Memory Read, Memory Write, Memory Write) and 16 T-States for execution as shown in the timing diagram.

Data Transfer Instructions

2

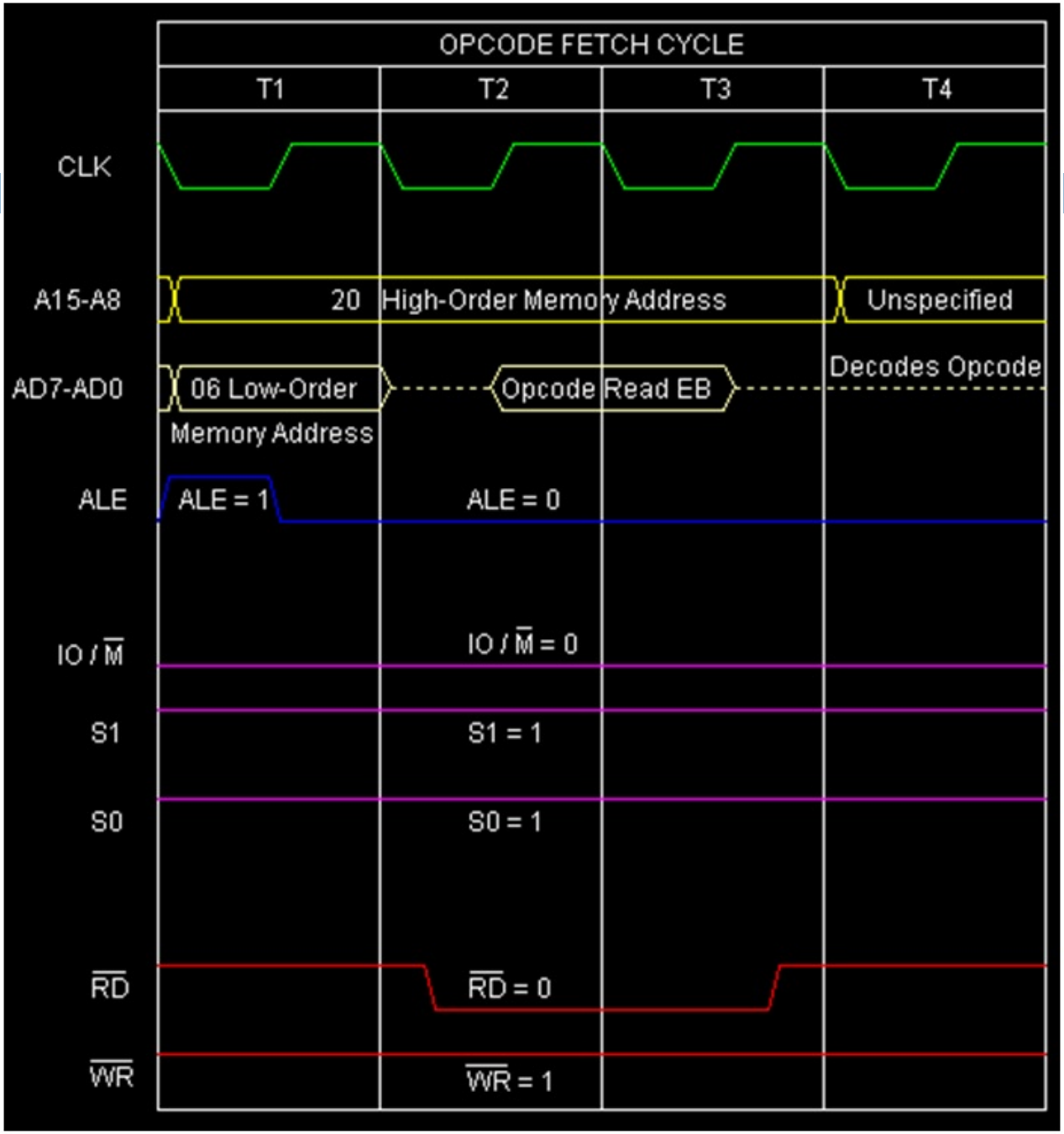
Opcode	Operand	Description
XCHG	None	Exchange H-L with D-E

- The contents of register H are exchanged with the contents of register D.
- The contents of register L are exchanged with the contents of register E.
- **Example:** XCHG

	Before	After
(HL)	ABCDH	6789H
(DE)	6789H	ABCDH

Address	Hex Codes	Mnemonic	Comment
2006	EB	XCHG	Swapping the contents of DE and HL register pairs

1-Byte instruction


OPCODE FETCH CYCLE
T1
T2
T3
T4
CLK
A15-A8
20
High-Order Memory Address
Unspecified
AD7-AD0
06 Low-Order Memory Address
Opcode Read EB
Decodes Opcode
ALE
ALE = 1
ALE = 0
IO / M
IO / M = 0
S1
S1 = 1
S0
S0 = 1
RD
RD = 0
WR
WR = 1

Data Transfer Instruction

16bit Data transfer

LHLD 16bit address

Example:

LHLD C050
Loads data from C050 memory location to Register L
and Loads data from C051 memory location to Register H

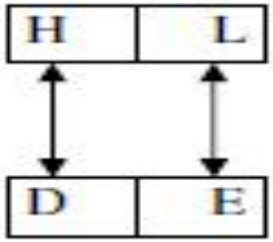
SHLD 16bit address

Example:

SHLD C050
Stores data of *Register L* to memory location C050.
and Stores data of *Register H* to memory location C051.

XCHG

Exchanges the content of HL and DE Register Pair



Arithmetic Instructions

2

Opcode	Operand	Description
ADD	R M	Add register or memory to accumulator

- The contents of register or memory are added to the contents of accumulator.
- The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of the addition.
- **Example:** ADD B or ADD M

ADD R

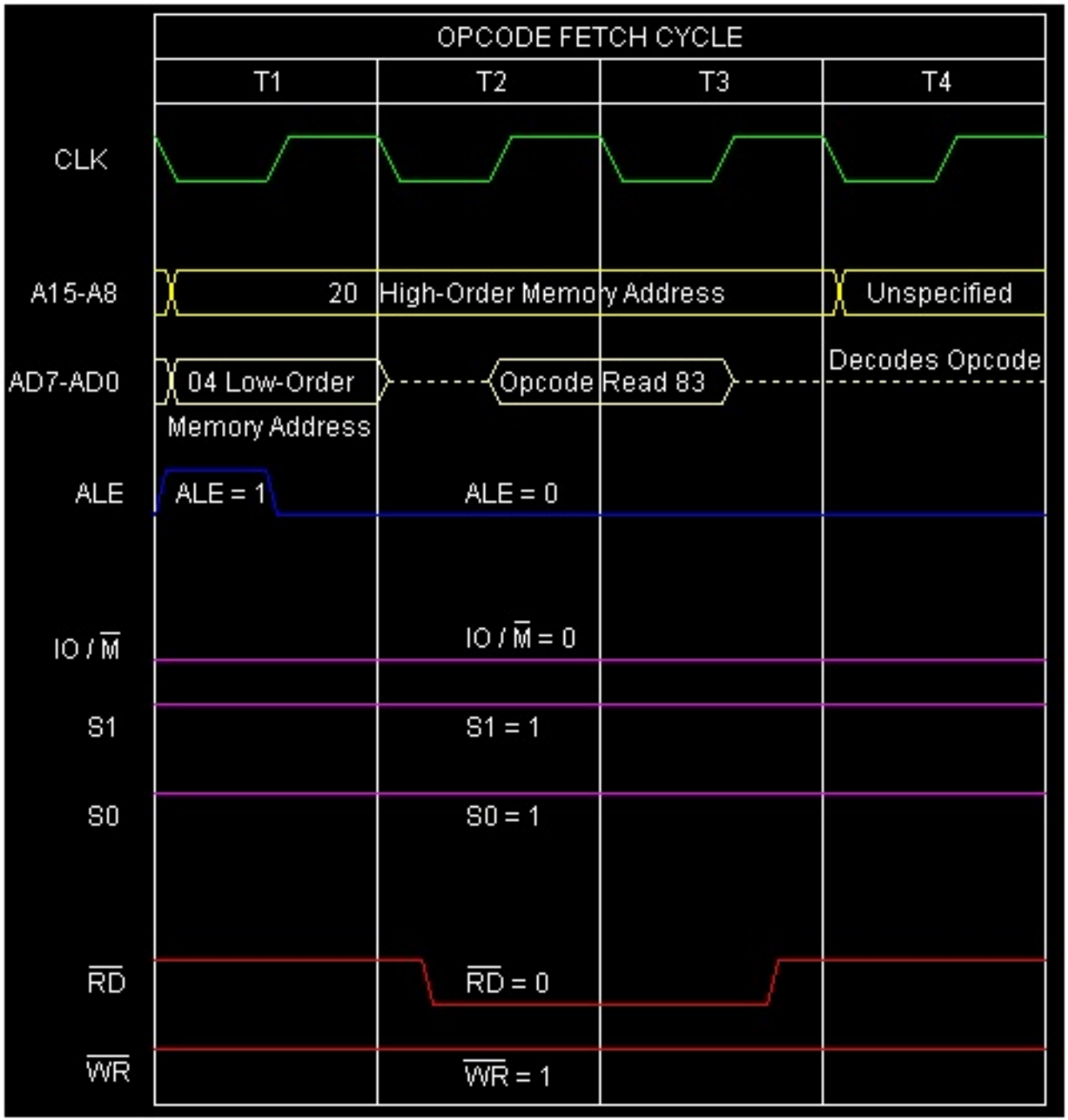
R = A, B, C, D, E, H, L, or M

ADD E.

	Before	After
(E)	03H	03H
(A)	02H	05H

Address	Hex Codes	Mnemonic	Comment
2004	83	ADD E	Accumulator = Accumulator + Register E

1-Byte instruction.


OPCODE FETCH CYCLE
T1
T2
T3
T4
CLK
A15-A8
20
High-Order Memory Address
Unspecified
AD7-AD0
04 Low-Order Memory Address
Opcode
Read 83
Decodes Opcode
ALE
ALE = 1
ALE = 0
IO / M
IO / M = 0
S1
S1 = 1
S0
S0 = 1
RD
RD = 0
WR
WR = 1

Arithmetic Instructions

2

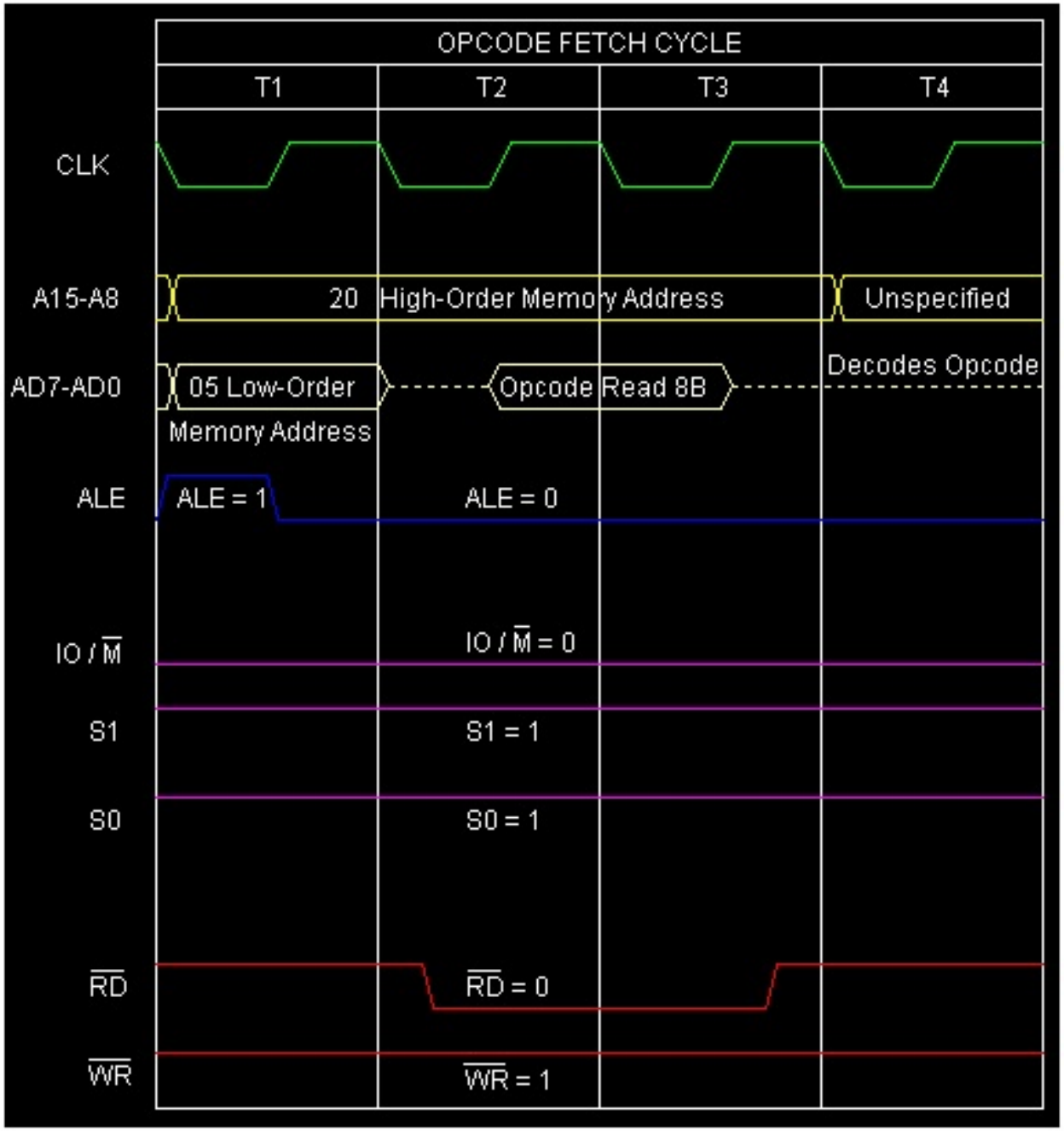
Opcode	Operand	Description
ADC	R M	Add register or memory to accumulator with carry

- The contents of register or memory and Carry Flag (CY) are added to the contents of accumulator.
- The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of the addition.
- **Example:** ADC B or ADC M

	Before	After
(E)	10H	10H
(A)	A0H	B1H
(F)	Cy=1, Other flag bits may have any value	Cy=0,AC=0,Z=0,P=1, S=1

Address	Hex Codes	Mnemonic	Comment
2005	8B	ADC	Accumulator = Accumulator + E Register + Cy Flag Bit

1-Byte instruction.


OPCODE FETCH CYCLE
T1
T2
T3
T4
CLK
A15-A8
20
High-Order Memory Address
Unspecified
AD7-AD0
05 Low-Order
Memory Address
Opcode
Read 8B
Decodes Opcode
ALE
ALE = 1
ALE = 0
IO / M
IO / M = 0
S1
S1 = 1
S0
S0 = 1
RD
RD = 0
WR
WR = 1

Arithmetic Instructions

2

Opcode	Operand	Description
ADI	8-bit data	Add immediate to accumulator

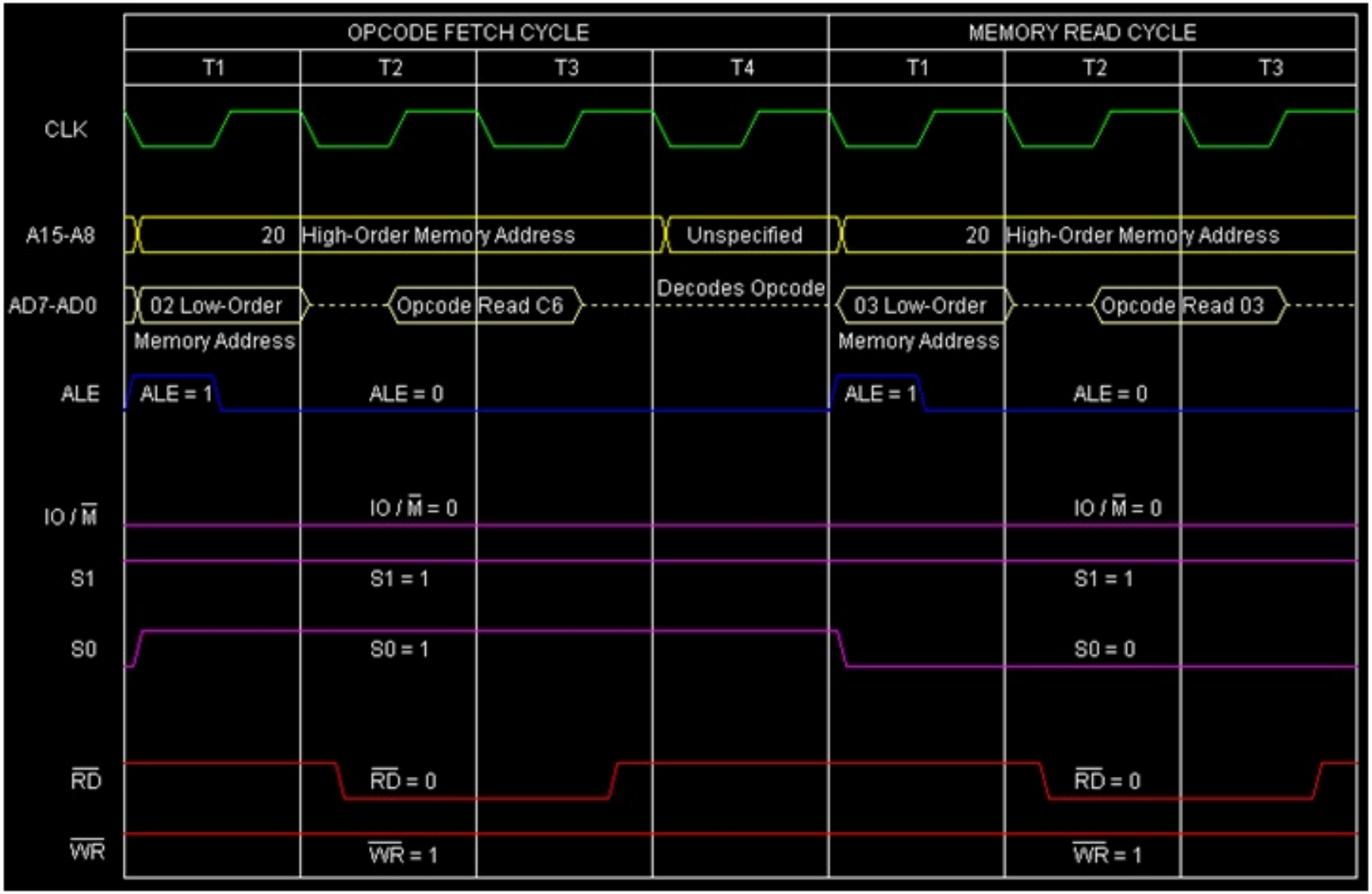
- The 8-bit data is added to the contents of accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of the addition.
- **Example:** ADI 45 H

ADI 03H

	Before	After
(A)	02H	05H
Flag Register (F)	Any values	CY=0,AC=0,S=0,P=1, Z=0

Address	Hex Codes	Mnemonic	Comment
2002	C6	ADI 03H	Accumulator = Accumulator + Operand 03H
2003	03		Data operand 03H

2-Byte instruction


OPCODE FETCH CYCLE
MEMORY READ CYCLE
T1
T2
T3
T4
T1
T2
T3
CLK
A15-A8
20
High-Order Memory Address
Unspecified
20
High-Order Memory Address
AD7-AD0
02 Low-Order Memory Address
Opcode Read C6
Decodes Opcode
03 Low-Order Memory Address
Opcode Read 03
ALE
ALE = 1
ALE = 0
ALE = 1
ALE = 0
IO / M
IO / M = 0
IO / M = 0
S1
S1 = 1
S1 = 1
S0
S0 = 1
S0 = 0
RD
RD = 0
RD = 0
WR
WR = 1
WR = 1

Arithmetic Instructions

2

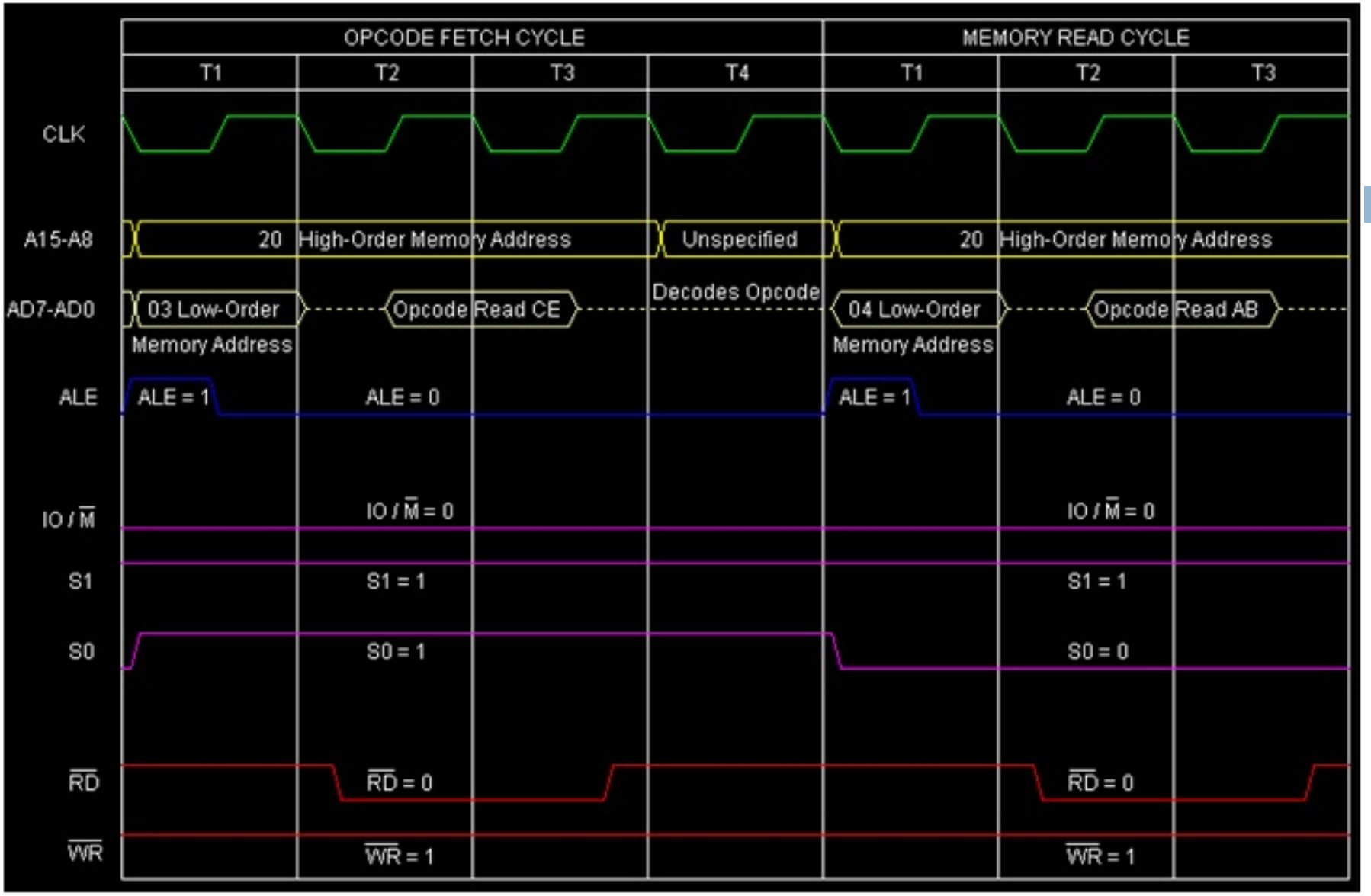
Opcode	Operand	Description
ACI	8-bit data	Add immediate to accumulator with carry

- The 8-bit data and the Carry Flag (CY) are added to the contents of accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of the addition.
- **Example:** ACI 45 H

ACI ABH

	Before	After
(A)	10H	BCH
(F)	Cy=1 Other flag bits=any values	Cy=0,AC=0,S=1,P=0, Z=0

Address	Hex Codes	Mnemonic	Comment
2003	CE	ACI AB	Accumulator = Accumulator + ABH Operand + Cy Flag Bit
2004	AB		Operand ABH



So this instruction **ACI ABH** requires 2-Bytes, 2-Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

2

Opcode	Operand	Description
DAD	Reg. pair	Add register pair to H-L pair

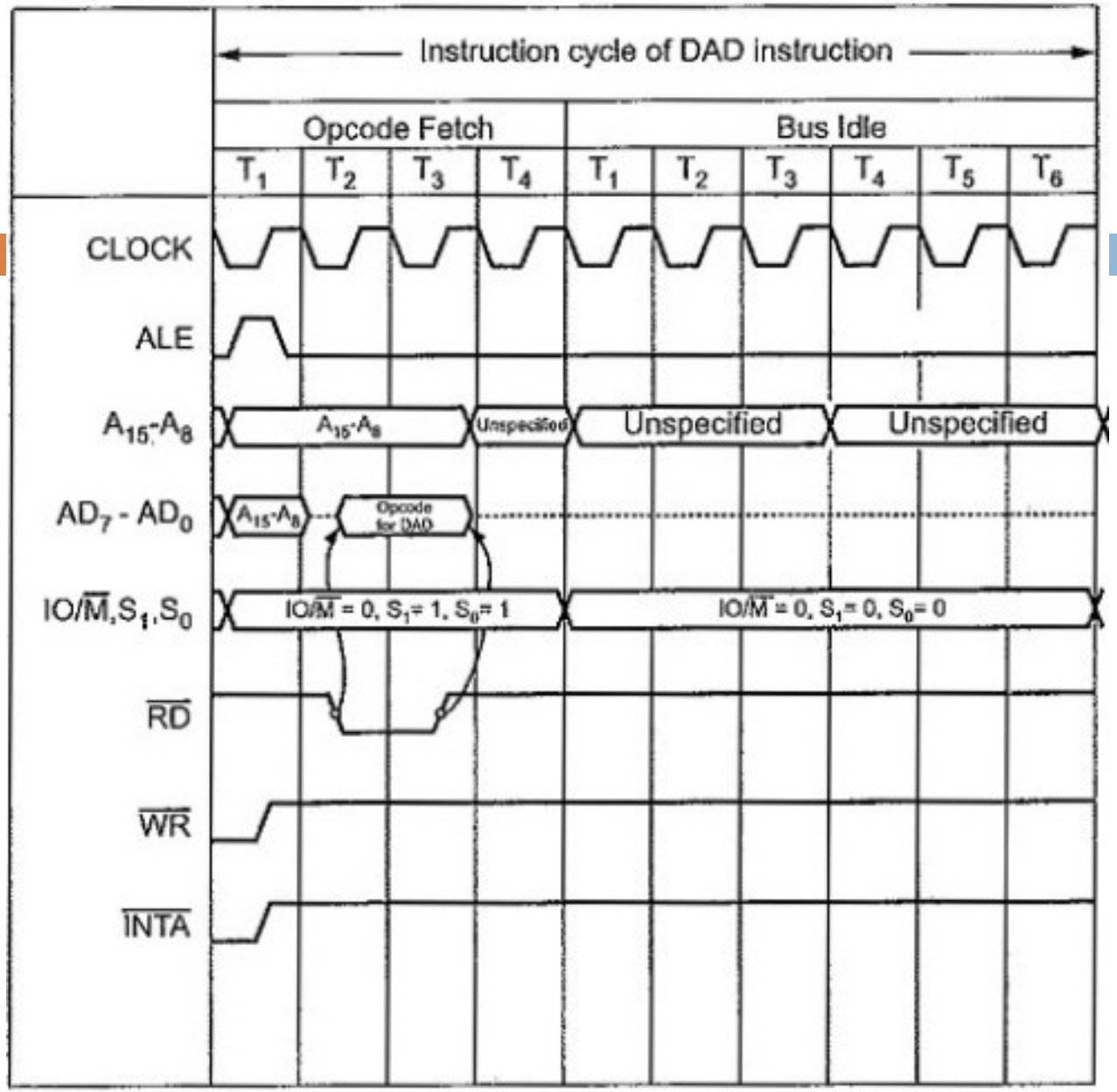
- The 16-bit contents of the register pair are added to the contents of H-L pair.
- The result is stored in H-L pair.
- If the result is larger than 16 bits, then CY is set.
- No other flags are changed.
- **Example:** DAD B

DAD B

	Before	After
(HL)	4050H	80B0H
(BC)	4060H	4060H
(F)	Any values	Cy=0, no change on other flag bits

Address	Hex Codes	Mnemonic	Comment
2006	09	DAD B	HL = HL + BC

1-Byte instruction



So this instruction **DAD B** requires 1-Byte, 3 Machine Cycle (Opcode Fetch, Bus Idle, Bus Idle) and 10 T-States for execution as shown in the timing diagram

Arithmetic Instructions

3

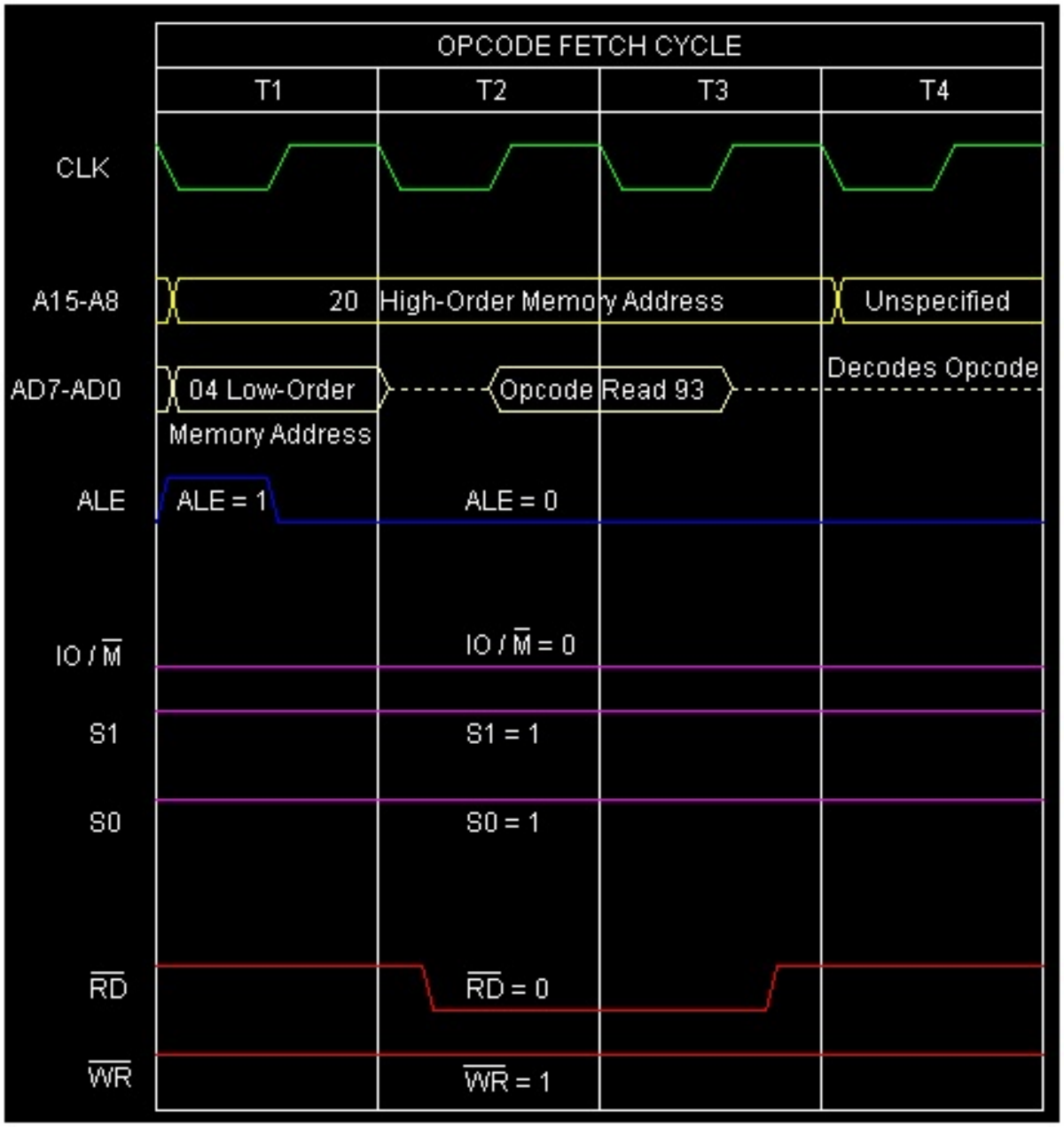
Opcode	Operand	Description
SUB	R M	Subtract register or memory from accumulator

- The contents of the register or memory location are subtracted from the contents of the accumulator.
- The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of subtraction.
- **Example:** SUB B or SUB M

SUB E

	Before	After
(A)	ABH	DEH
(E)	CDH	CDH
(F)	Any values	Cy=1,AC=0,S=1,P=1,Z=0

Address	Hex Codes	Mnemonic	Comment
2004	93	SUB E	Accumulator = Accumulator + 2's Complement of E



So this instruction **SUB E** requires 1-Byte, 1 Machine Cycles (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

Opcode	Operand	Description
SBB	R M	Subtract register or memory from accumulator with borrow

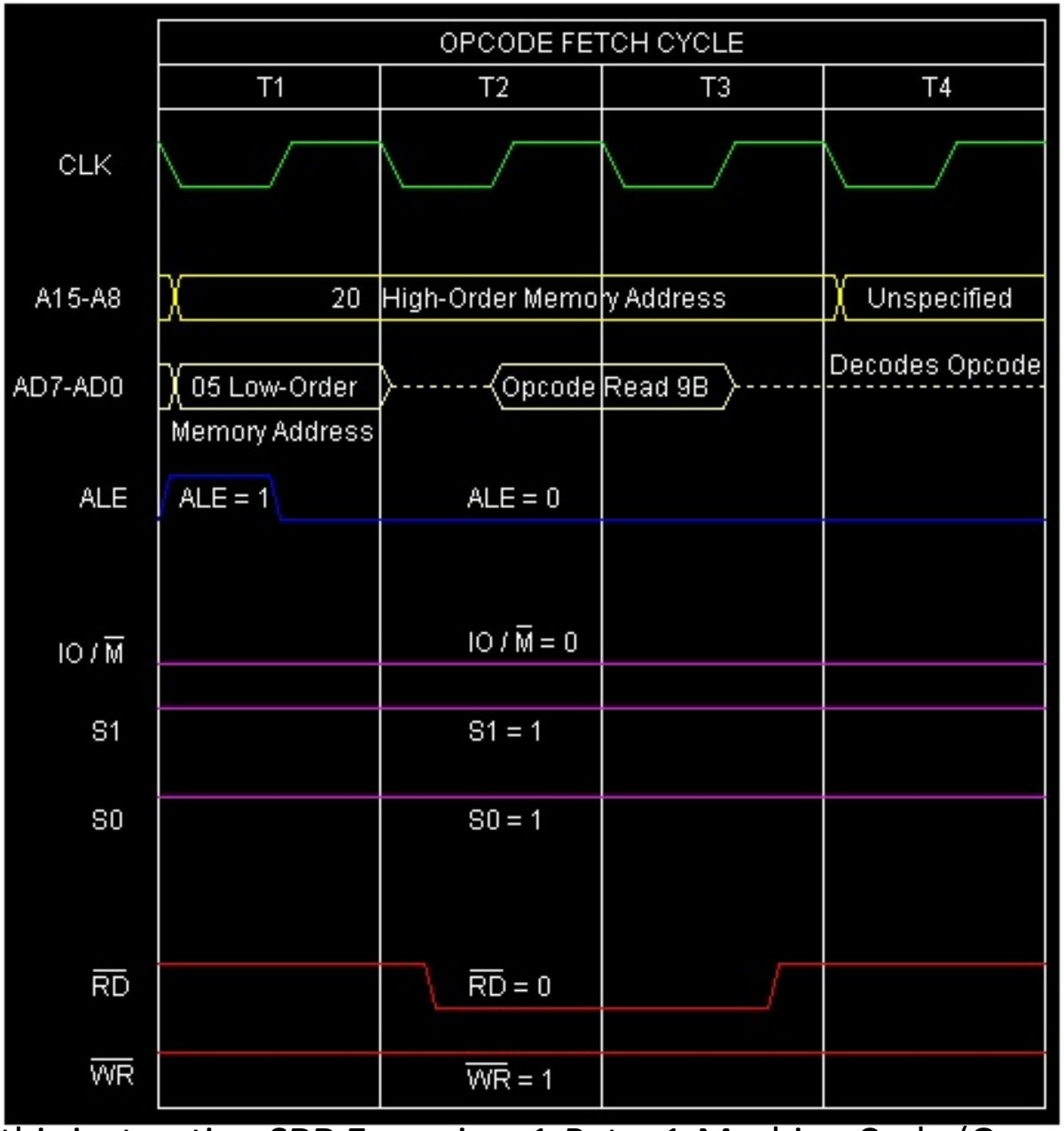
- The contents of the register or memory location and Borrow Flag (i.e. CY) are subtracted from the contents of the accumulator.
- The result is stored in accumulator.
- If the operand is memory location, its address is specified by H-L pair.
- All flags are modified to reflect the result of subtraction.
- **Example:** SBB B or SBB M

SBB E

	Before	After
(E)	13H	13H
(A)	44H	30H
(F)	Cy=1, other flag bits=any value	Cy=0, AC=1,S=0,P=1,Z=0

Address	Hex Codes	Mnemonic	Comment
2005	9B	SBB E	A register = A register – E register – 1

1-Byte instruction



So this instruction **SBB E** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

3

Opcode	Operand	Description
SUI	8-bit data	Subtract immediate from accumulator

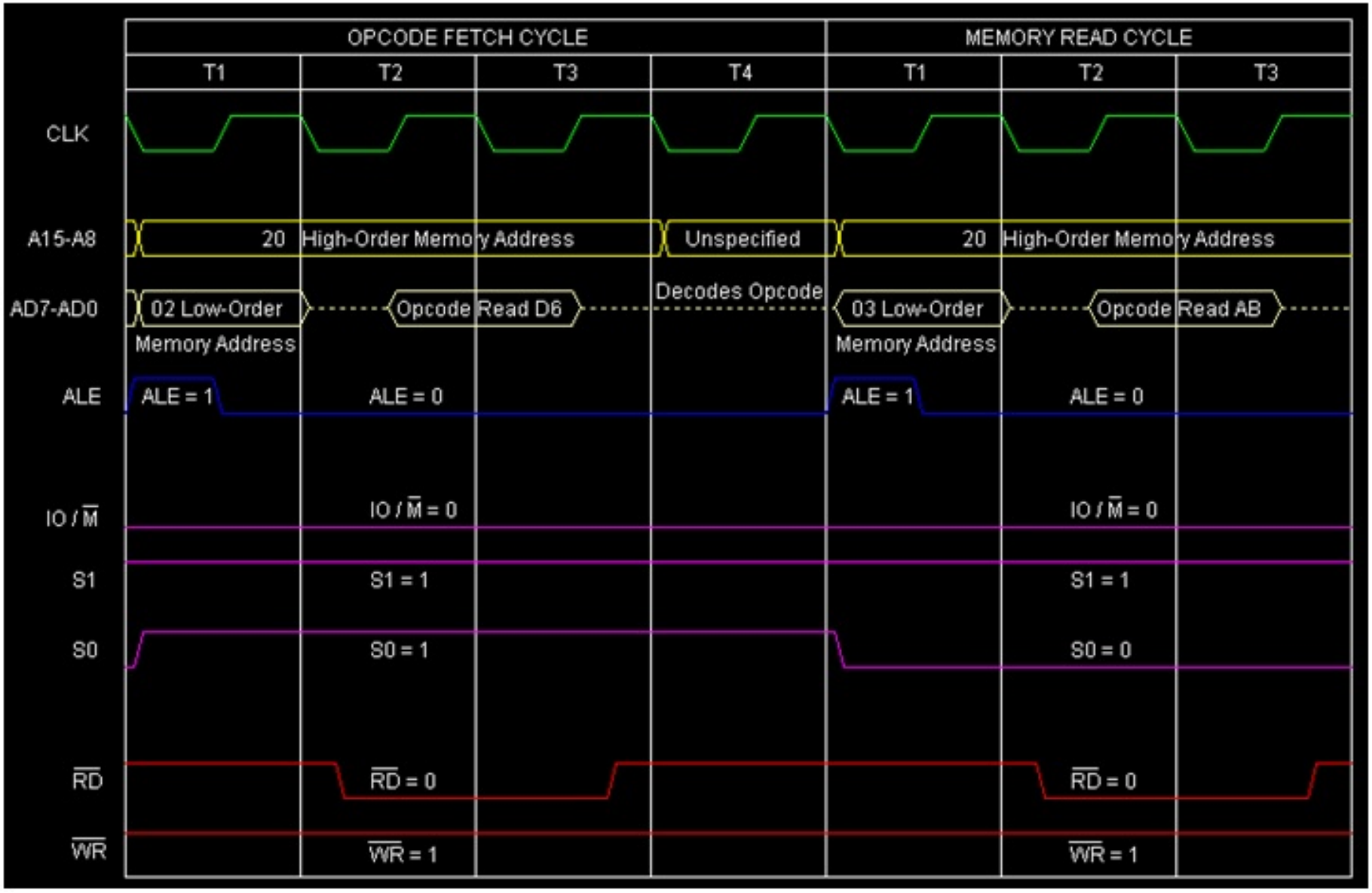
- The 8-bit data is subtracted from the contents of the accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of subtraction.
- **Example:** SUI 45 H

SUI ABH

	Before	After
(A)	CDH	22H
(F)	Any values	Cy=0,AC=1,S=0,P=1, Z=0

Address	Hex Codes	Mnemonic	Comment
2002	D6	SUI ABH	Accumulator = Accumulator + 2's Complement of ABH
2003	AB		Operand ABH

2-Byte instruction



requires 2-Bytes, 2-Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

3

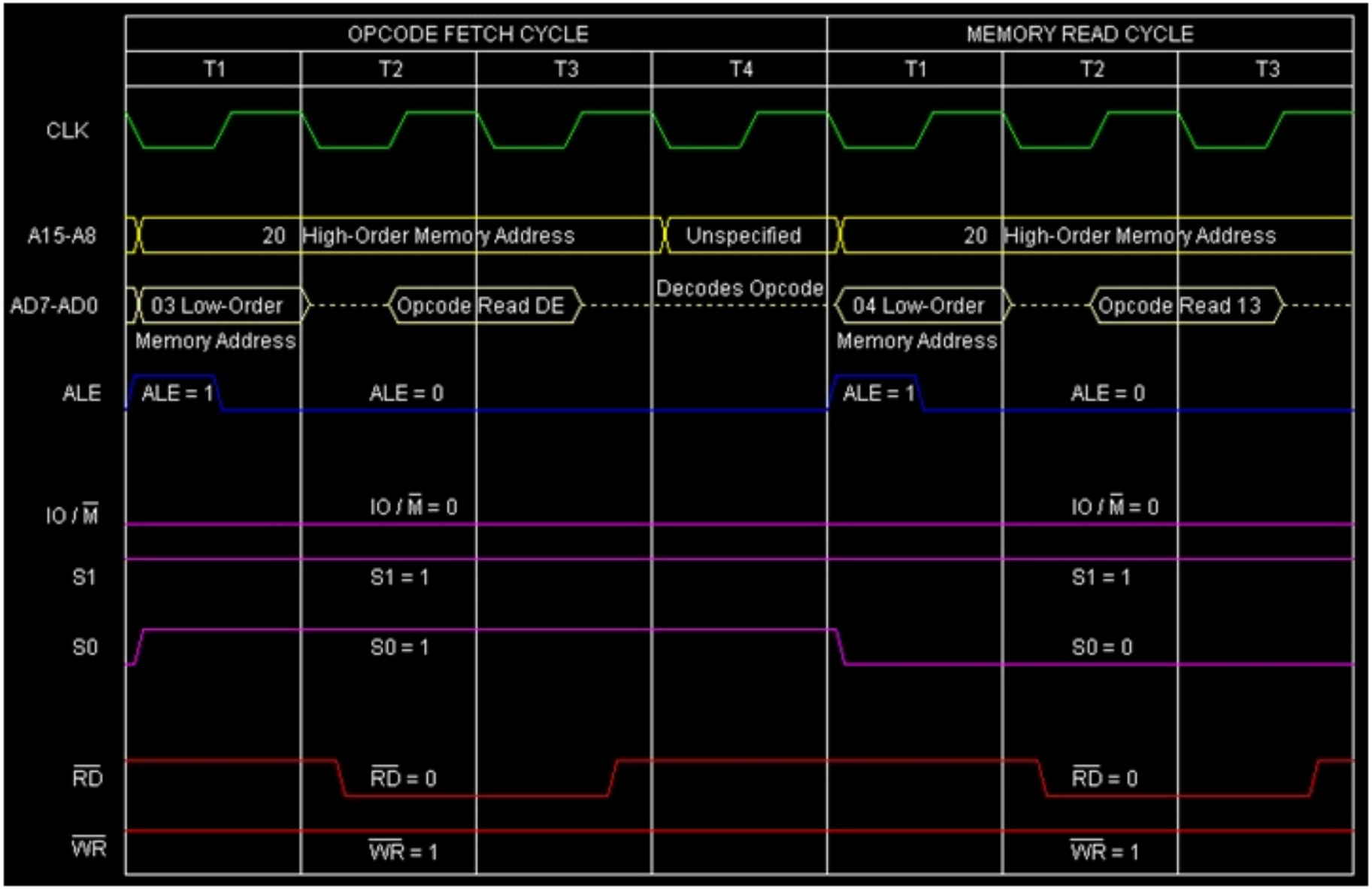
Opcode	Operand	Description
SBI	8-bit data	Subtract immediate from accumulator with borrow

- The 8-bit data and the Borrow Flag (i.e. CY) is subtracted from the contents of the accumulator.
- The result is stored in accumulator.
- All flags are modified to reflect the result of subtraction.
- **Example:** SBI 45 H

SBI 13H

	Before	After
(A)	44H	30H
(F)	Cy=1, other flag bits=any value	Cy=0, AC=1,S=0,P=1,Z=0

Address	Hex Codes	Mnemonic	Comment
2003	DE	SBI 13H	A register = A register – 13H – 1
2004	13H		Operand



So this instruction **SBI 13H** requires 2-Bytes, 2 Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

3

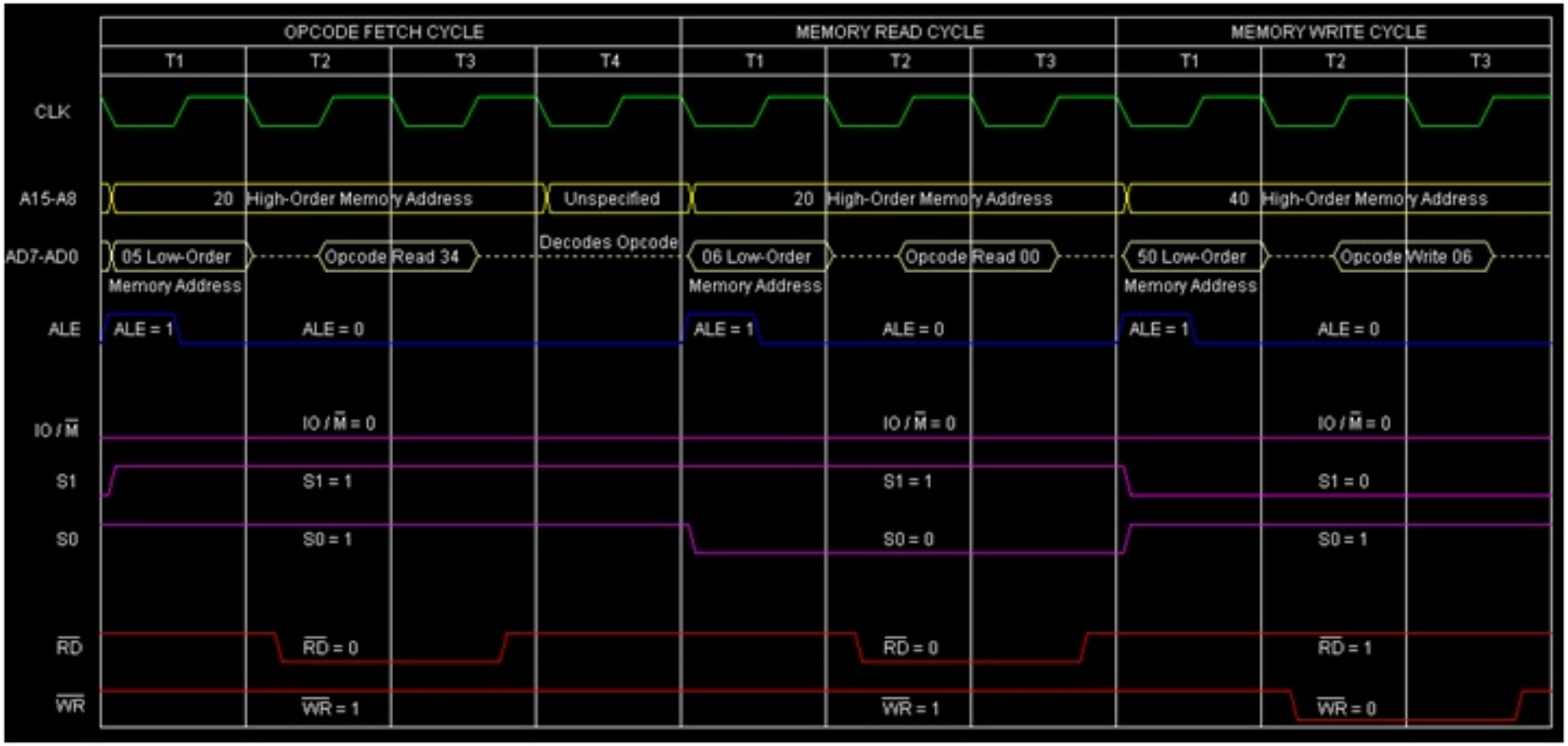
Opcode	Operand	Description
INR	R M	Increment register or memory by 1

- The contents of register or memory location are incremented by 1.
- The result is stored in the same place.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- **Example:** INR B or INR M

INR M

	Before	After
(HL)	4050H	4050H
(4050H)	05H	06H
Flag Register (F)	Any Value	Cy=no change, Ac=0,S=0,P=1,Z=0

Address	Hex Codes	Mnemonic	Comment
2005	34	INR M	(HL) = (HL) + 1, i.e. 4050H memory locations content will be increased by 1 as HL register pair is having 16-bit address 4050H



So this instruction **INR M** requires 1-Byte, 3-Machine Cycles (Opcode Fetch, Memory Read, Memory Write) and 10 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

3

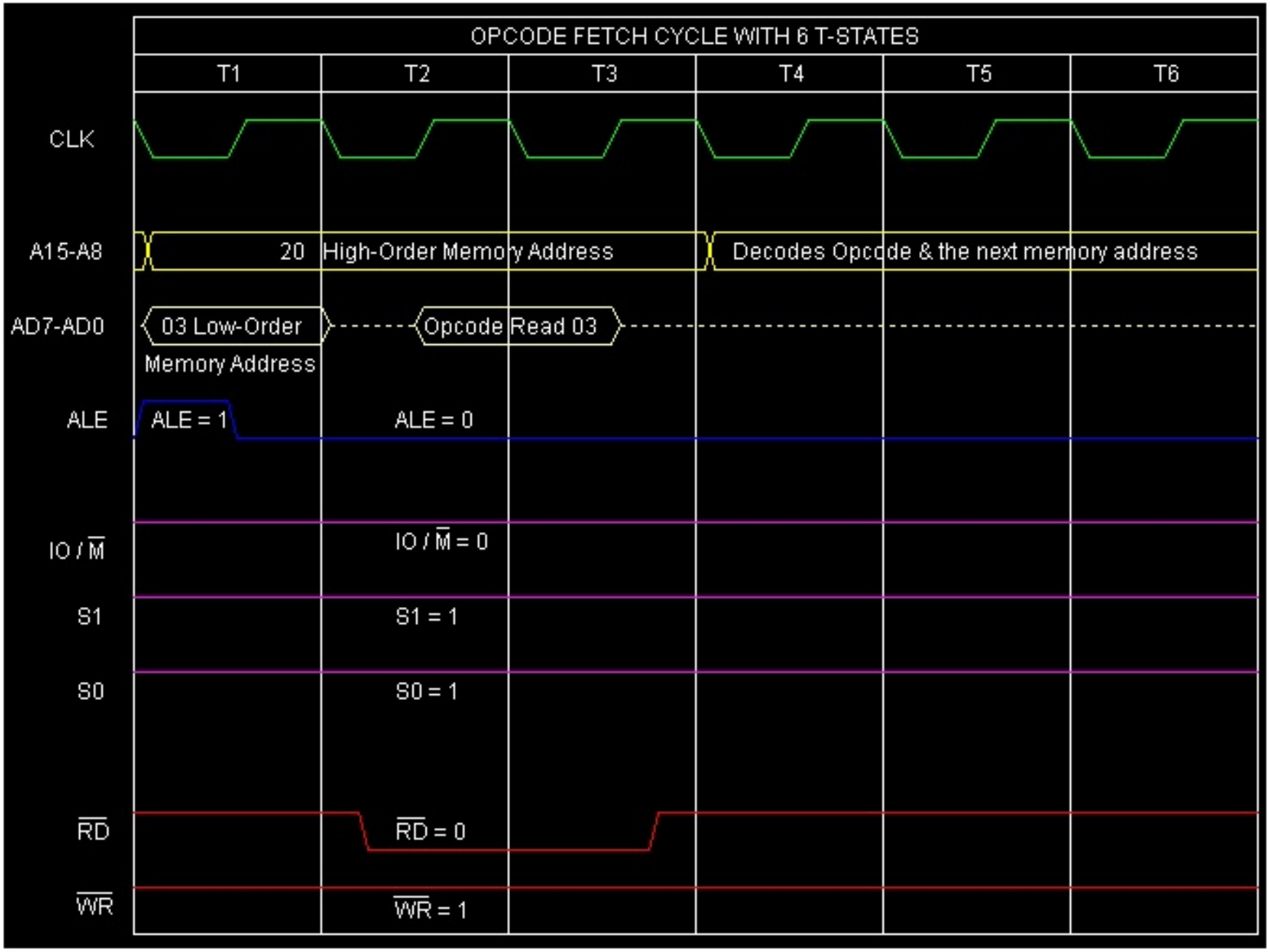
Opcode	Operand	Description
INX	R	Increment register pair by 1

- The contents of register pair are incremented by 1.
- The result is stored in the same place.
- **Example:** INX H

INX B

	Before	After
(BC)	4050H	4051H

Address	Hex Codes	Mnemonic	Comment
2003	03	INX B	BC = BC + 1



So this instruction **INX B** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 6 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

3

Opcode	Operand	Description
DCR	R M	Decrement register or memory by 1

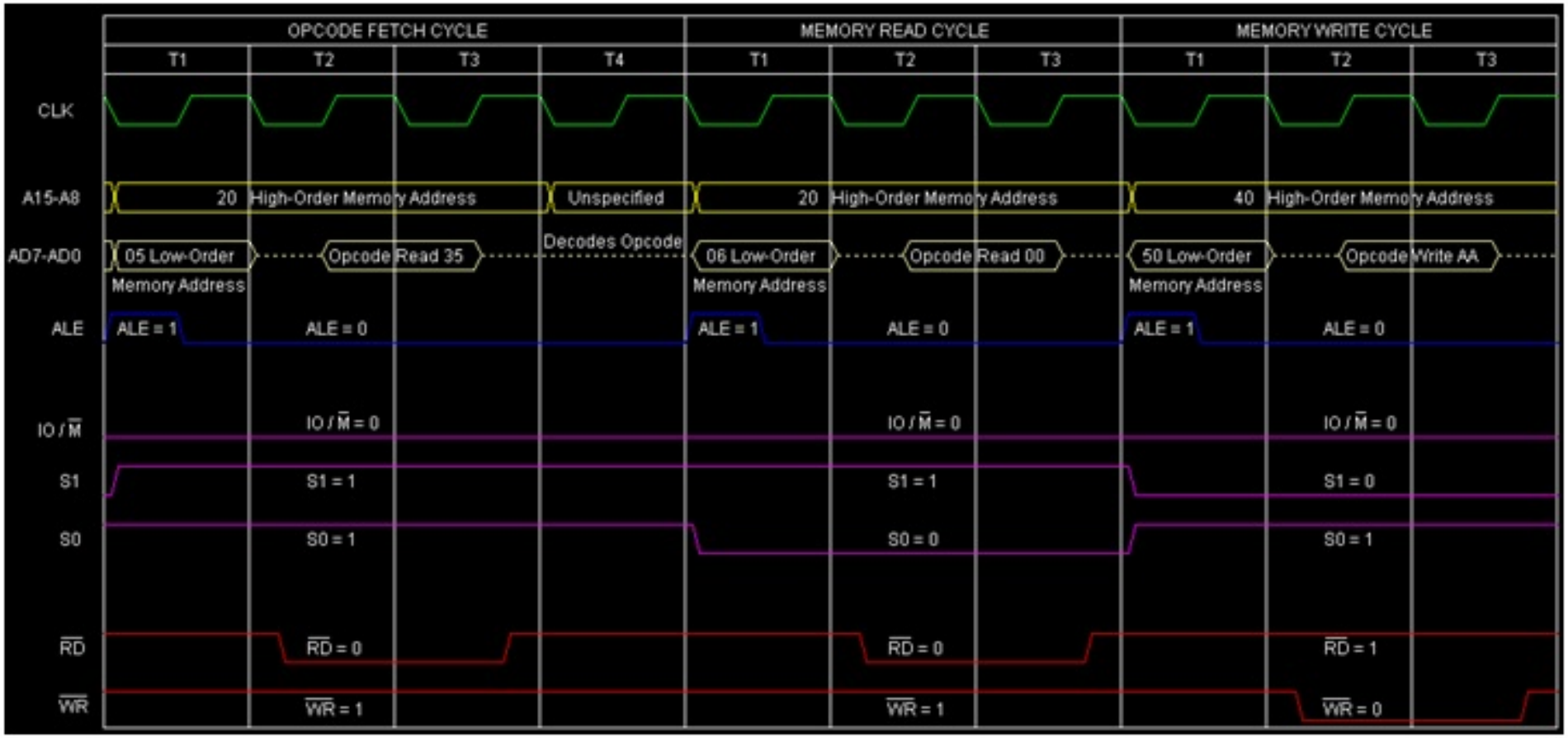
- The contents of register or memory location are decremented by 1.
- The result is stored in the same place.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- **Example:** DCR B or DCR M

DCR M

	Before	After
(HL)	4050H	4050H
(4050H)	ABH	AAH
(F)	Any values	Cy=no change, AC=1,S=1,P=1,Z=0

Address	Hex Codes	Mnemonic	Comment
2005	35	DCR M	(HL) = (HL) - 1

1 byte instruction



So this instruction **DCR M** requires 1-Bytes, 3-Machine Cycles (Opcode Fetch, Memory Read, Memory Write) and 10 T-States for execution as shown in the timing diagram.

Arithmetic Instructions

3

Opcode	Operand	Description
DCX	R	Decrement register pair by 1

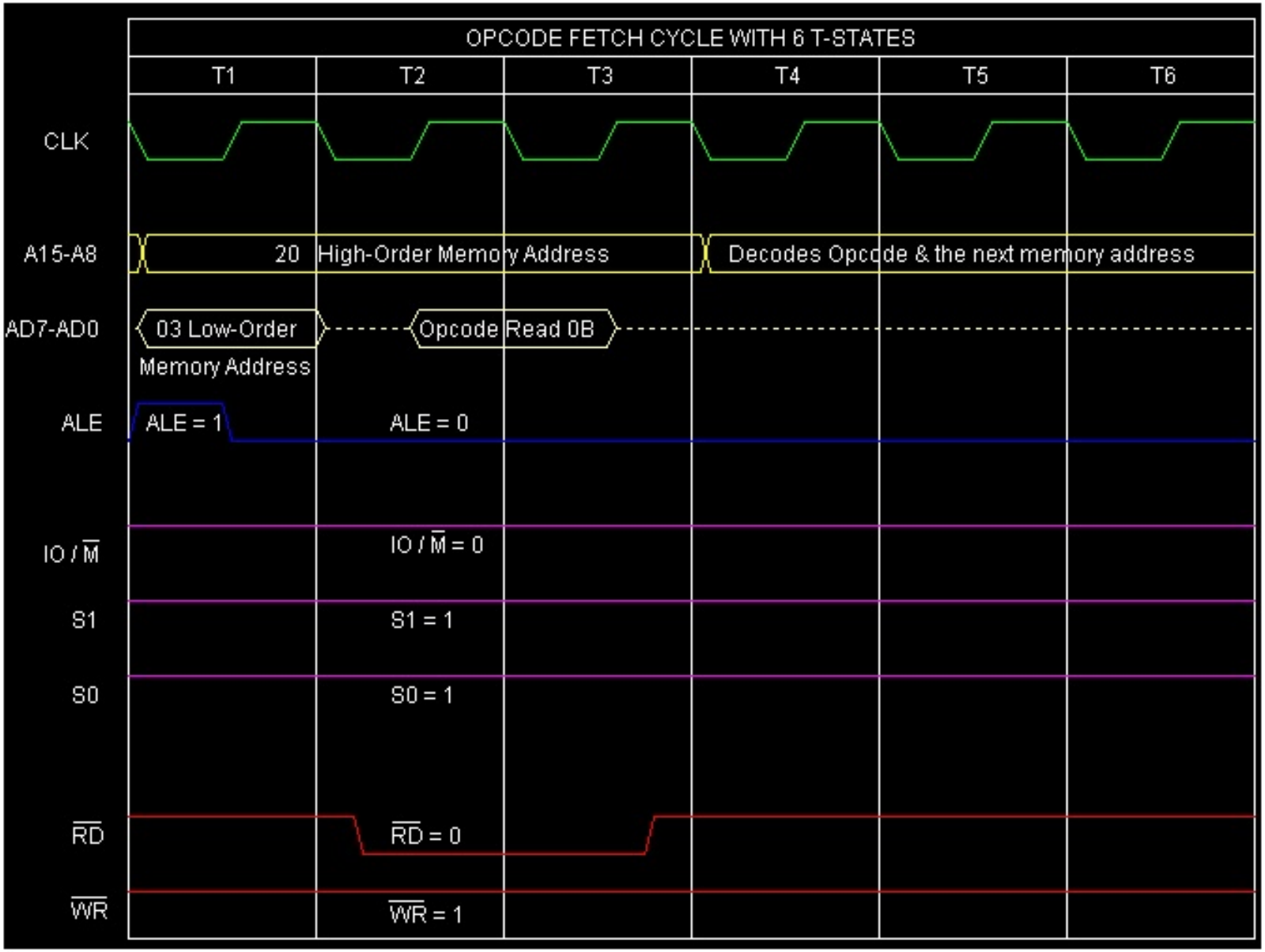
- The contents of register pair are decremented by 1.
- The result is stored in the same place.
- **Example:** DCX H

DCX B

	Before	After
(BC)	4055H	4054H

Address	Hex Codes	Mnemonic	Comment
2003	0B	DCX B	BC = BC – 1

1-Byte instruction,



instruction **DCX B** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 6 T-States for execution as shown in the timing diagram.

Arithmetic

16 bit addition instructions

DAD Rp : Adds the content of the HL register with the content of the Register pair specified by Rp and stores the result in the HL register pair

Example:

DAD B

Adds the content of the HL register with the content of the BC register pair stores the result in the HL register pair

HL ← HL+BC

41	01
H	L

06	02
D	E

DAD B will result in :

47	03
H	L

What DAD B does :

```
  4101      [ HL ]        0100 0001 0000 0001
+ 0602      [ DE ]        0000 0110 0000 0010
----------                ---------------------
  4703                    0100 0111  0000 0011
```

Logical Instructions

3

- These instructions perform logical operations on data stored in registers, memory and status flags.

- The logical operations are:
  - AND
  - OR
  - XOR
  - Rotate
  - Compare
  - Complement

AND, OR, XOR

- Any 8-bit data, or the contents of register, or memory location can logically have
  - AND operation
  - OR operation
  - XOR operation

  with the contents of accumulator.
- The result is stored in accumulator.

Rotate

4

- Each bit in the accumulator can be shifted either left or right to the next position.

Compare

4

- Any 8-bit data, or the contents of register, or memory location can be compares for:
  - Equality
  - Greater Than
  - Less Than

  with the contents of accumulator.
- The result is reflected in status flags.

Complement

4

- The contents of accumulator can be complemented.
- Each 0 is replaced by 1 and each 1 is replaced by 0.

Logical Instructions

4

Opcode	Operand	Description
CMP	R M	Compare register or memory with accumulator

- The contents of the operand (register or memory) are compared with the contents of the accumulator.
- Both contents are preserved .
- The result of the comparison is shown by setting the flags of the PSW as follows:

Logical Instructions

4

Opcode	Operand	Description
CMP	R M	Compare register or memory with accumulator

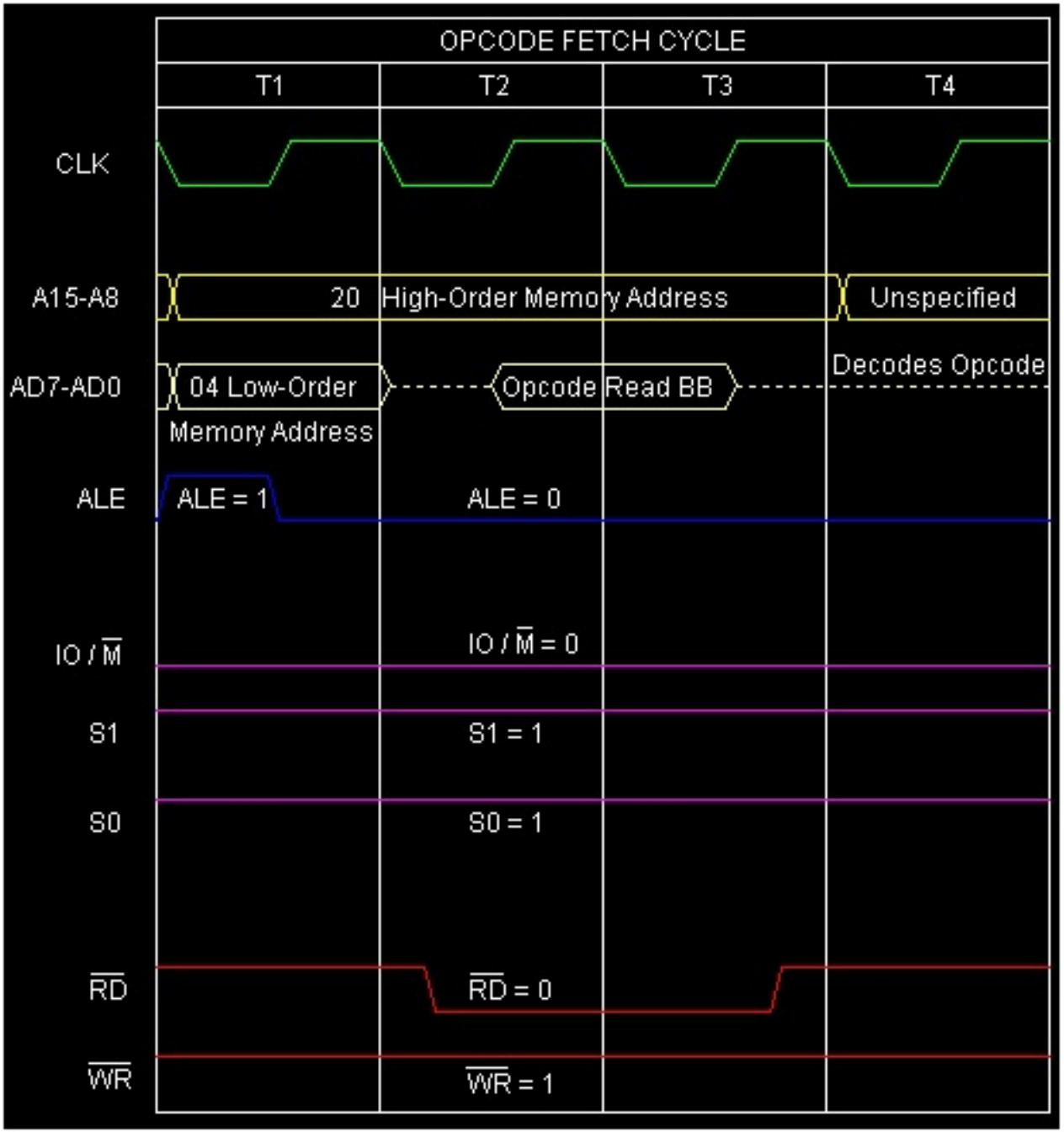
- if (A) < (reg/mem): carry flag is set
- if (A) = (reg/mem): zero flag is set
- if (A) > (reg/mem): carry and zero flags are reset.
- **Example:** CMP B or CMP M

CMP E

	Before	After
(A)	50H	50H
(E)	70H	70H
(Temp)	Any value	E0H
(F)	Any value	Cy=1,AC=0,S=1,P=0, Z=0

1-Byte instruction

Address	Hex Codes	Mnemonic	Comment
2004	BB	CMP E	Temp = Register A - Register E


OPCODE FETCH CYCLE
T1
T2
T3
T4
CLK
A15-A8
20
High-Order Memory Address
Unspecified
AD7-AD0
04 Low-Order
Memory Address
Opcode
Read BB
Decodes Opcode
ALE
ALE = 1
ALE = 0
IO / M
IO / M = 0
S1
S1 = 1
S0
S0 = 1
RD
RD = 0
WR
WR = 1

Logical Instructions

4

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

- The 8-bit data is compared with the contents of accumulator.
- The values being compared remain unchanged.
- The result of the comparison is shown by setting the flags of the PSW as follows:

Logical Instructions

4

Opcode	Operand	Description
CPI	8-bit data	Compare immediate with accumulator

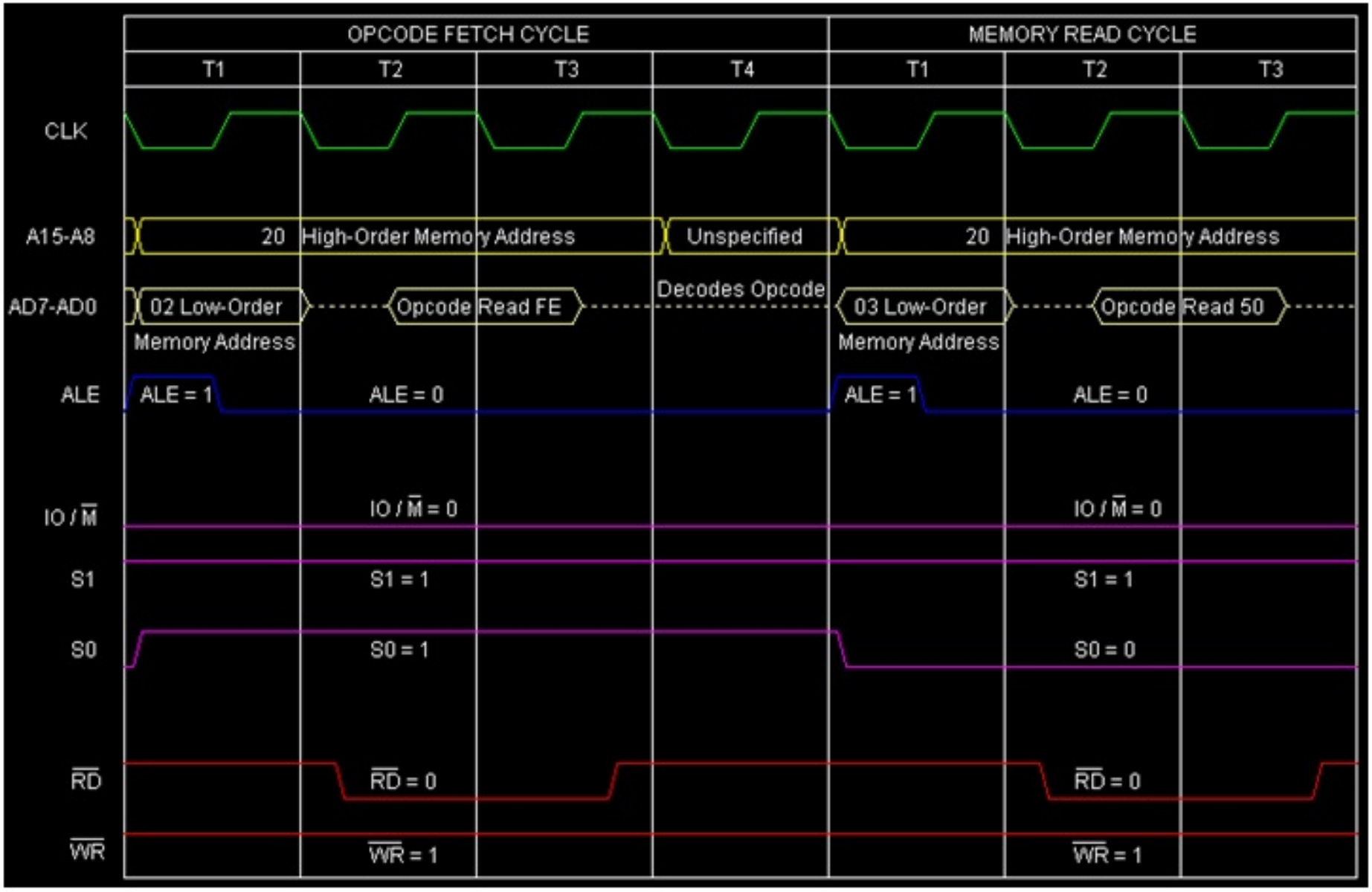
- if (A) < data: carry flag is set
- if (A) = data: zero flag is set
- if (A) > data: carry and zero flags are reset
- **Example:** CPI 89H

CPI 50H

	Before	After
(A)	70H	70H
(Temp)	Any value	20H
(F)	Any value	Cy=0,AC=0,S=0,P=0, Z=0

2-Byte instruction

Address	Hex Codes	Mnemonic	Comment
2002	FE	CPI 50H	Temp = Register A – 50H
2003	50		Operand 50H



So this instruction **CPI 50H** requires 2-Bytes, 2-Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

Logical Instructions

4

Opcode	Operand	Description
ANA	R M	Logical AND register or memory with accumulator

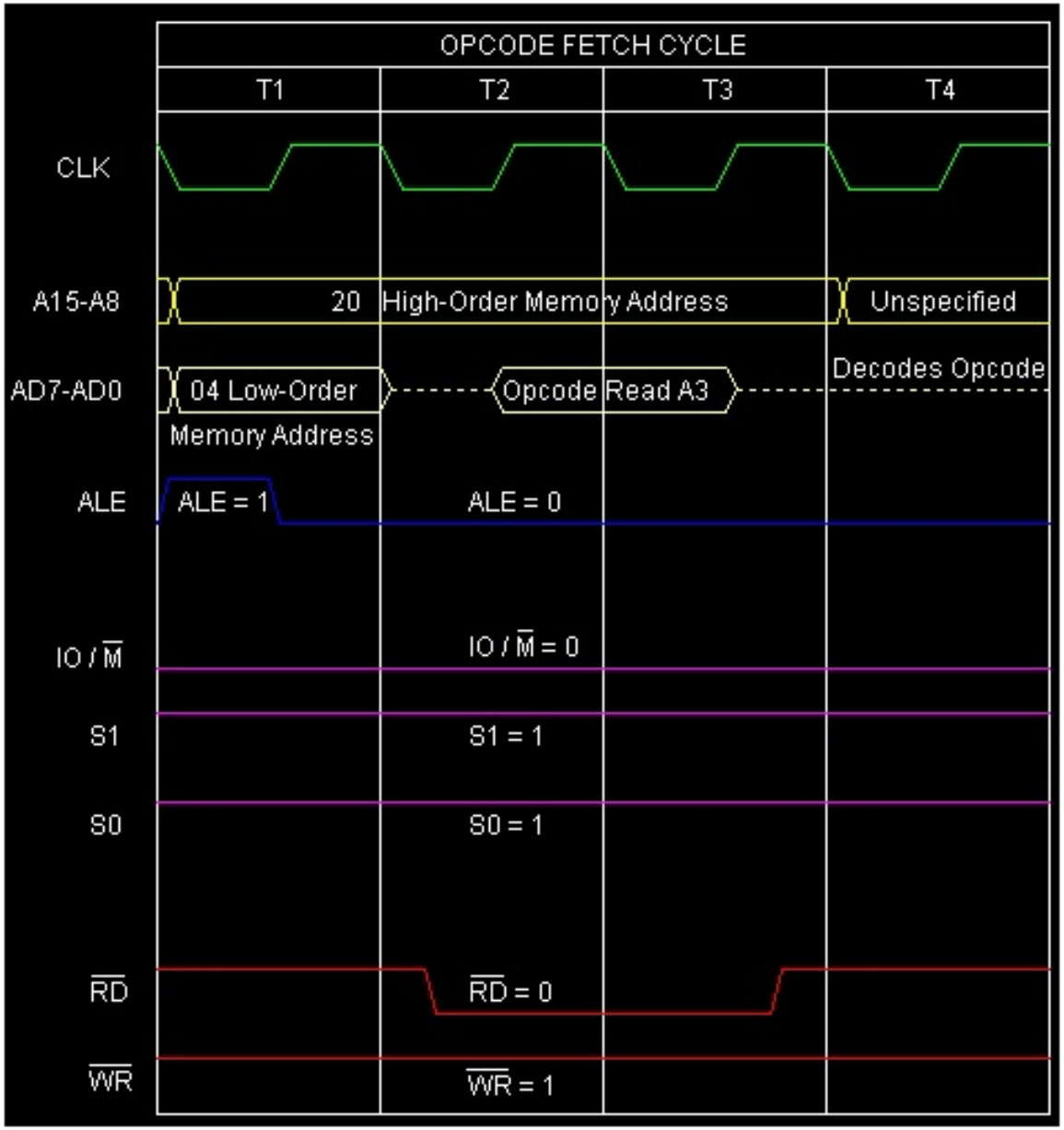
- The contents of the accumulator are logically ANDed with the contents of register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result of the operation.
- CY is reset and AC is set.
- **Example:** ANA B or ANA M.

ANA E

	Before	After
(E)	CDH	CDH
(A)	ABH	89H
(F)	Any values	Cy=0,AC=1,Z=0,P=0, S=1

Address	Hex Codes	Mnemonic	Comment
2004	A3	ANA E	A = A and E

1-Byte instruction



So this instruction **ANA E** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Logical Instructions

4

Opcode	Operand	Description
ANI	8-bit data	Logical AND immediate with accumulator

- The contents of the accumulator are logically ANDed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY is reset, AC is set.
- **Example:** ANI 86H.

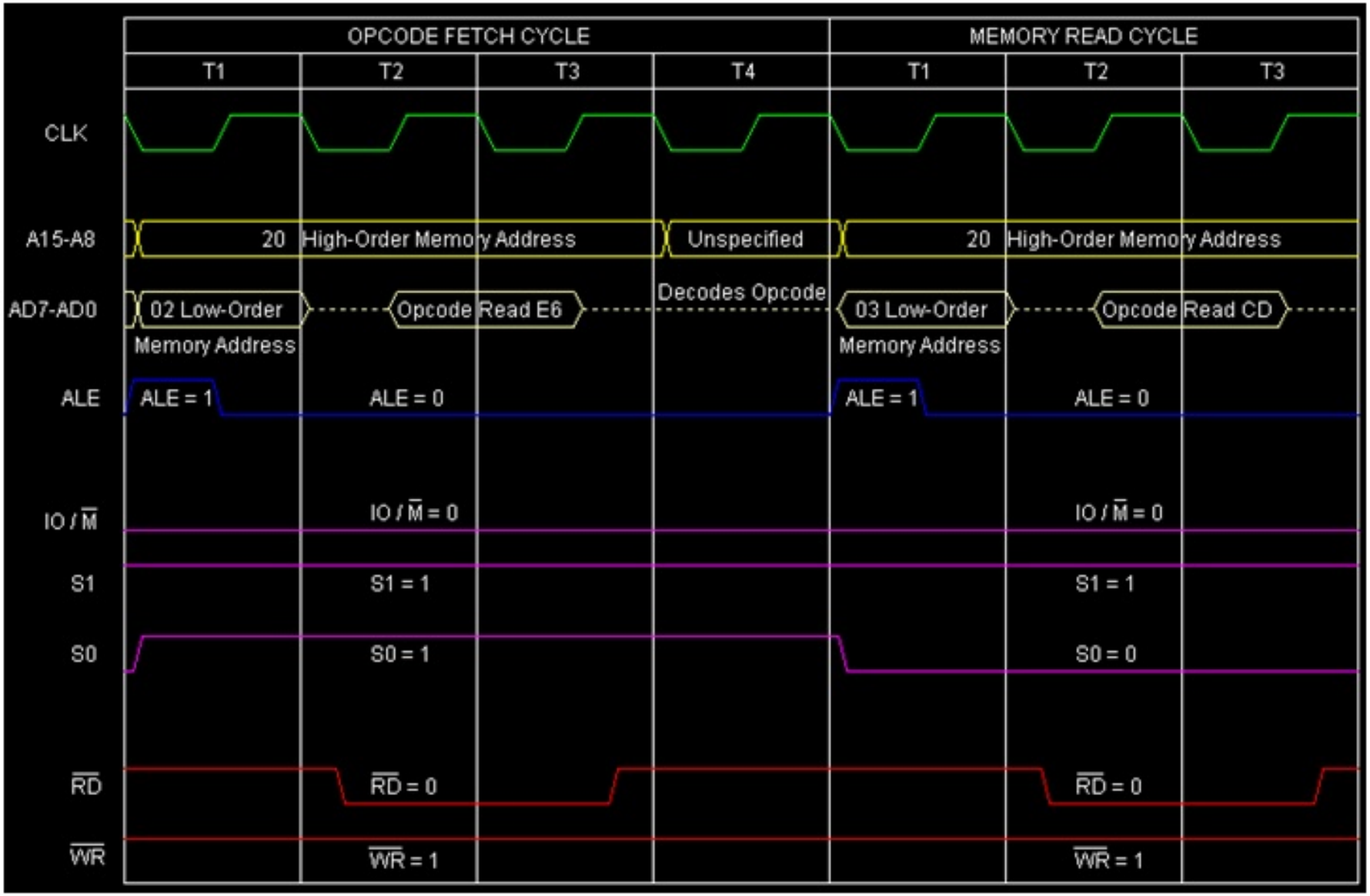
ANI CDH

	Before	After
(A)	ABH	89H
(F)	Any values	Cy=0,AC=1,S=1,P=0, Z=0

The internal calculation has been done as follows −

(A) ABH ---> 1010 1011
(B) (d8) CDH ---> 1100 1101 ---------
(C) ANI CDH ---> 1000 1001 (89H)

Address	Hex Codes	Mnemonic	Comment
2002	E6	ANI CD	A = A and CDH
2003	CD		Operand CDH



So this instruction **ANA E** requires 2-Bytes, 2 Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

Logical Instructions

5

Opcode	Operand	Description
ORA	R M	Logical OR register or memory with accumulator

- The contents of the accumulator are logically ORed with the contents of the register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** ORA B or ORA M.

ORA E

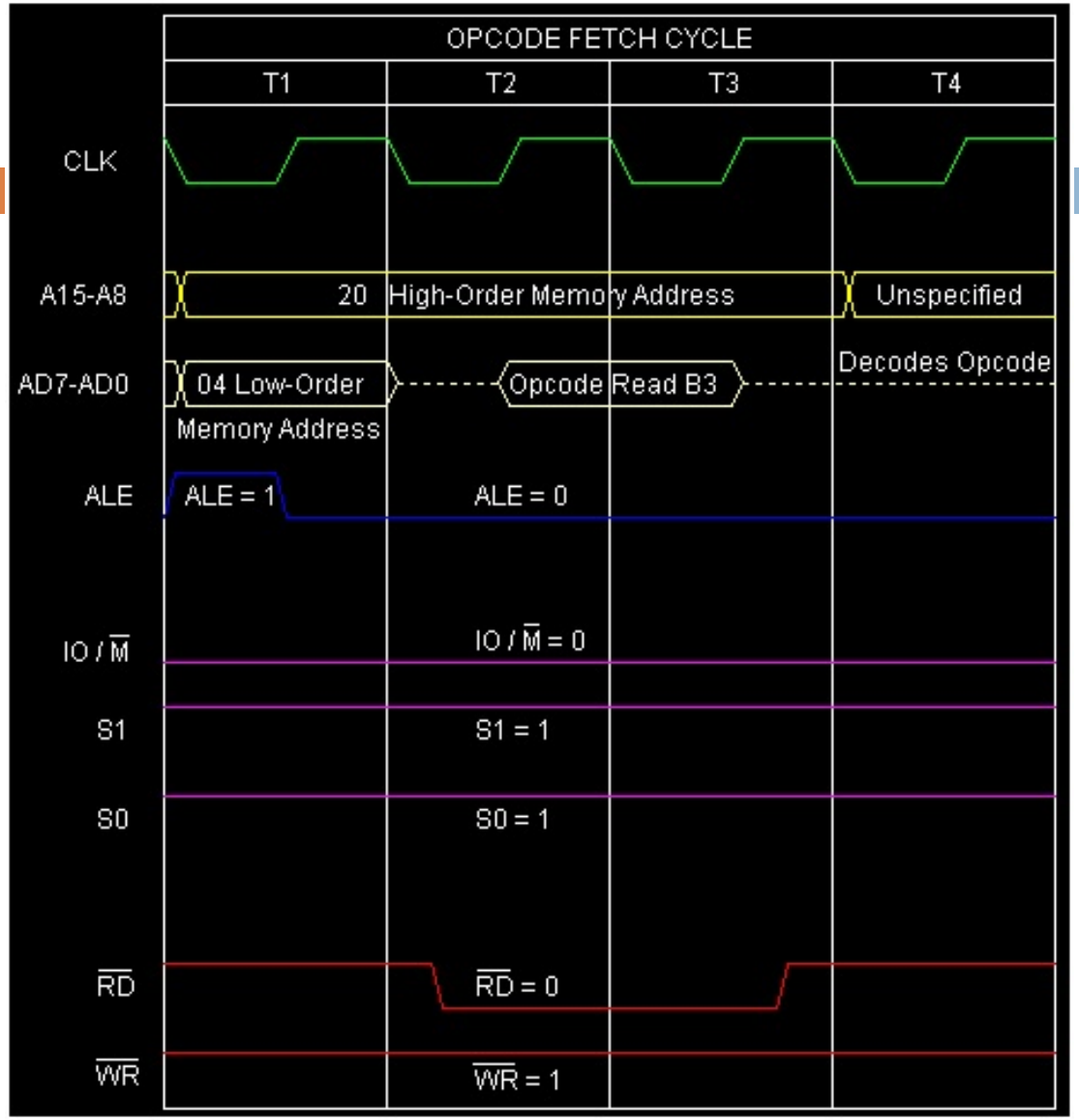
	Before	After
(E)	CDH	CDH
(A)	ABH	EFH
(F)	Any values	Cy=0,AC=0,S=1,P=0, Z=0

The internal calculation has been done as shown below −

(A) ABH ---> 1010 1011
(B) (E) CDH ---> 1100 1101 ---------
(C) ORA E ---> 1110 1111 (EFH)

Address	Hex Codes	Mnemonic	Comment
2004	B3	ORA E	A = A OR E

1-Byte instruction



So this instruction **ORA E** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Logical Instructions

5

Opcode	Operand	Description
ORI	8-bit data	Logical OR immediate with accumulator

- The contents of the accumulator are logically ORed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** ORI 86H.

ORI CDH

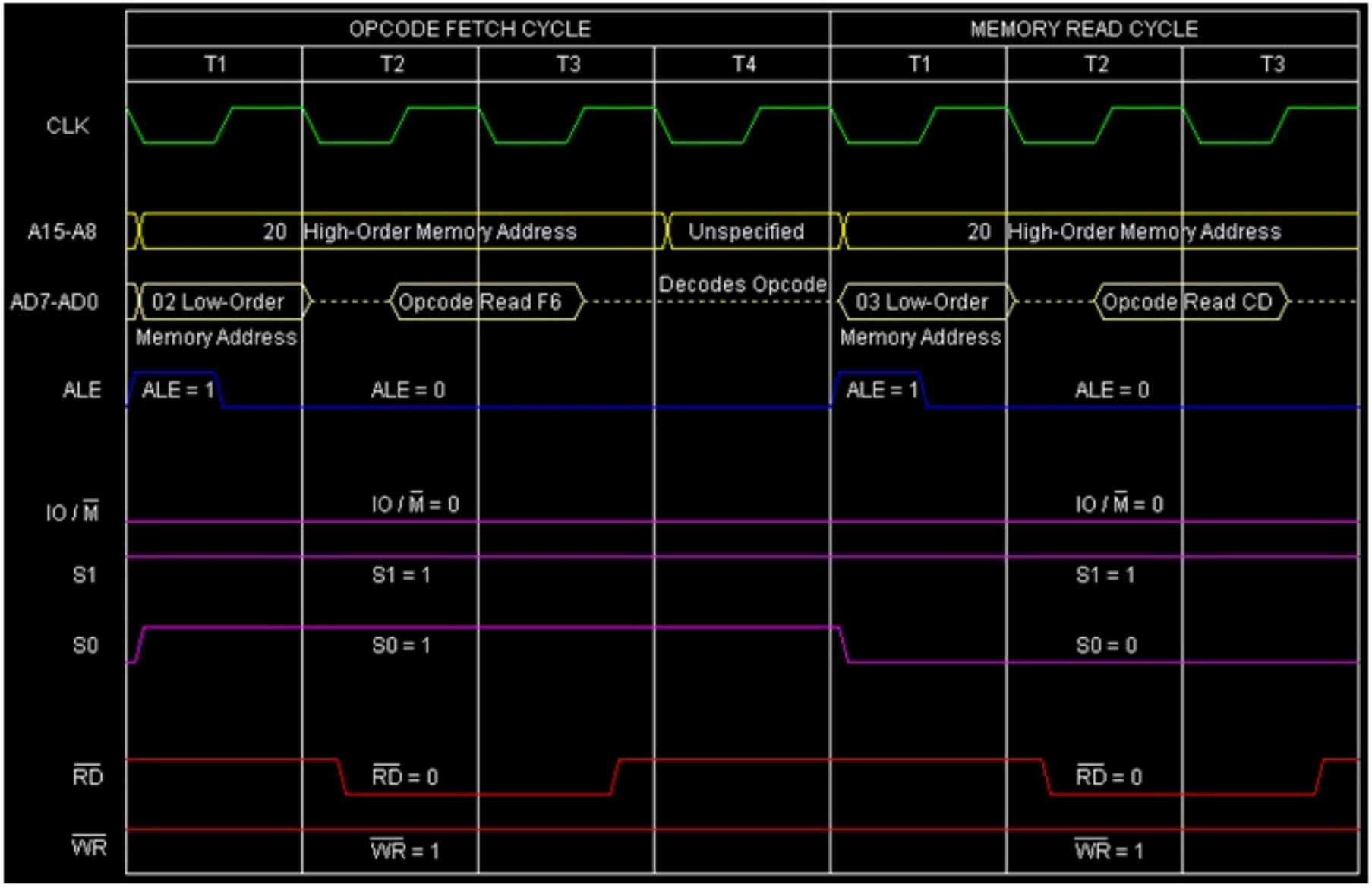
	Before	After
(A)	ABH	EFH
(F)	Any values	Cy=0,AC=0,S=1,P=0, Z=0

The internal calculation has been done as shown below −

(A) ABH ---> 1010 1011 (
(B) d8) CDH ---> 1100 1101 ---------
(C) ORI CDH ---> 1110 1111 (EFH)

Address	Hex Codes	Mnemonic	Comment
2002	F6	ORI CDH	A = A or CDH
2003	CD		Operand CDH

2-Byte instruction



So this instruction **ORI CDH** requires 2-Bytes, 2-Machine Cycles (Opcode Fetch, Memory Red) and 7 T-States for execution as shown

Logical Instructions

5

Opcode	Operand	Description
XRA	R M	Logical XOR register or memory with accumulator

- The contents of the accumulator are XORed with the contents of the register or memory.
- The result is placed in the accumulator.
- If the operand is a memory location, its address is specified by the contents of H-L pair.
- S, Z, P are modified to reflect the result of the operation.
- CY and AC are reset.
- **Example:** XRA B or XRA M.

XRA E

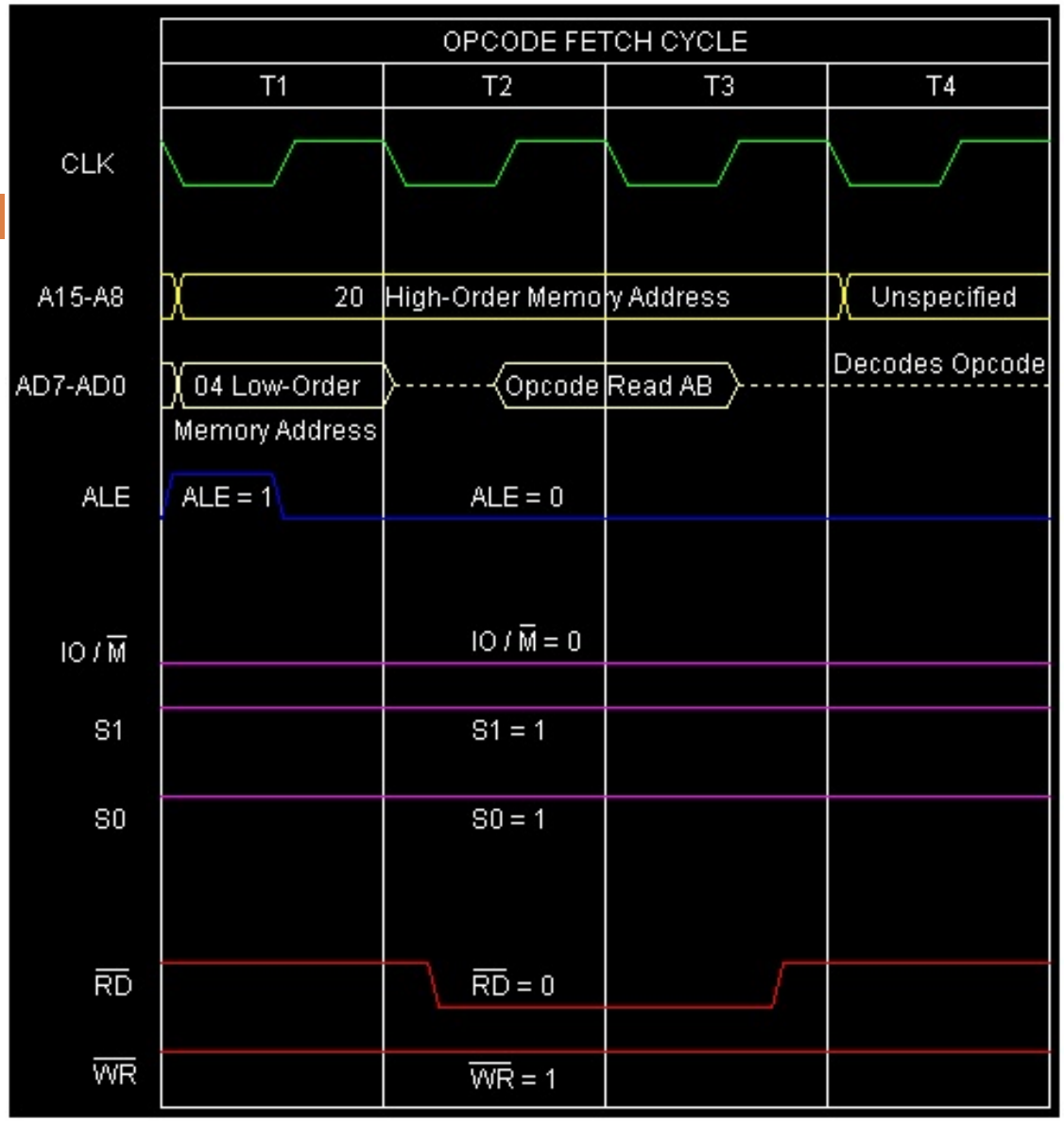
	Before	After
(E)	CDH	CDH
(A)	ABH	66H
(F)	Any values	Cy=0,AC=0,S=0,P=1, Z=0

The internal calculation has been done as shown below −

(A) ABH ---> 1010 1011 (E) CDH ---> 1100 1101 --------- XRAE --->0110 0110 (66H)

Address	Hex Codes	Mnemonic	Comment
2004	AB	XRA E	A = A XOR E

1-Byte instruction



So this instruction **XRA E** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Logical Instructions

5

Opcode	Operand	Description
XRI	8-bit data	XOR immediate with accumulator

- The contents of the accumulator are XORed with the 8-bit data.
- The result is placed in the accumulator.
- S, Z, P are modified to reflect the result.
- CY and AC are reset.
- **Example:** XRI 86H.

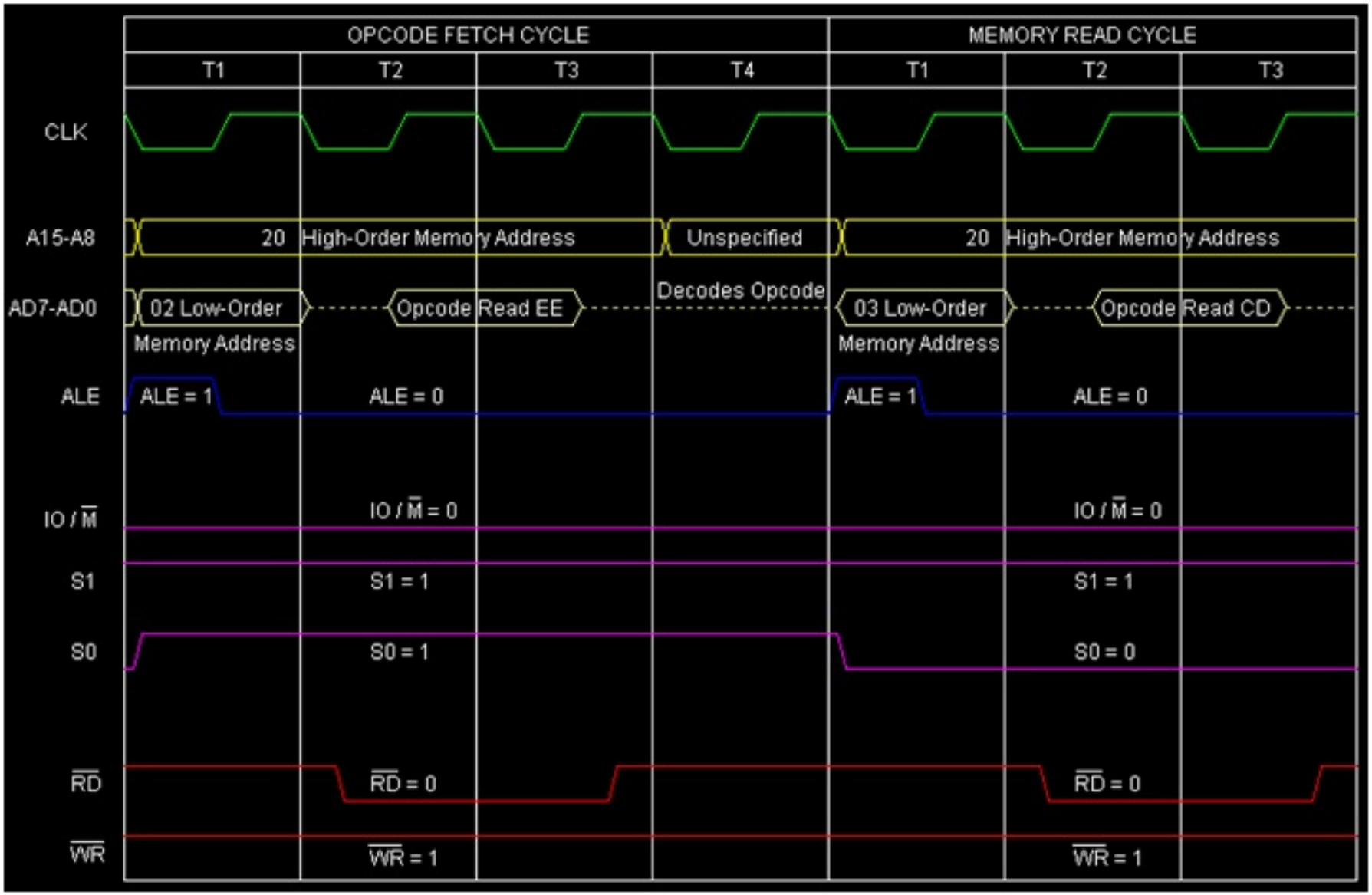
XRI CDH

	Before	After
(A)	ABH	66H
(F)	Any values	Cy=0,AC=0,S=0,P=1, Z=0

The internal calculation has been done as shown below −

(A) ABH ---> 1010 1011 (d8) CDH ---> 1100 1101 --------- XRICDH --->0110 0110 (66H)

Address	Hex Codes	Mnemonic	Comment
2002	EE	XRI CDH	A = A XOR CDH
2003	CD		Operand CDH



So this instruction **ORI CDH** requires 2-Bytes, 2-Machine Cycles (Opcode Fetch, Memory Read) and 7 T-States for execution as shown in the timing diagram.

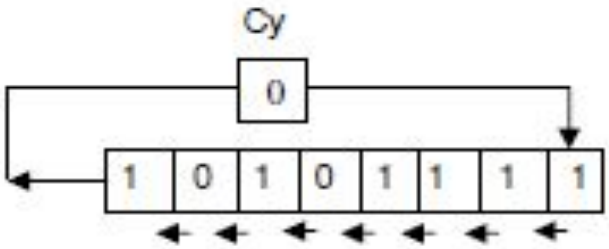
Opcode	Operand	Description
RAL	None	Rotate accumulator left through carry

- Each binary bit of the accumulator is rotated left by one position through the Carry flag.
- Bit D7 is placed in the Carry flag, and the Carry flag is placed in the least significant position Do.
- CY is modified according to bit D7.
- S, Z, P, AC are not affected.
- **Example:** RAL.

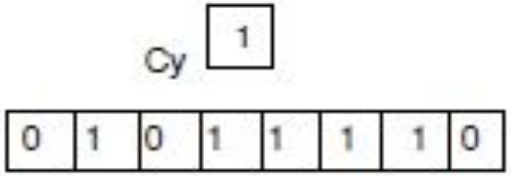
Logical Instructions

RAL (Rotate Accumulator Left through Carry)

Every bit of the Accumulator is shifted one bit left and the MSB bit of the Accumulator is copied into the Carry flag and the Carry flag value is copied into the A0th bit



After the RAL instruction is executed the content of the Accumulator and the carry flag will be

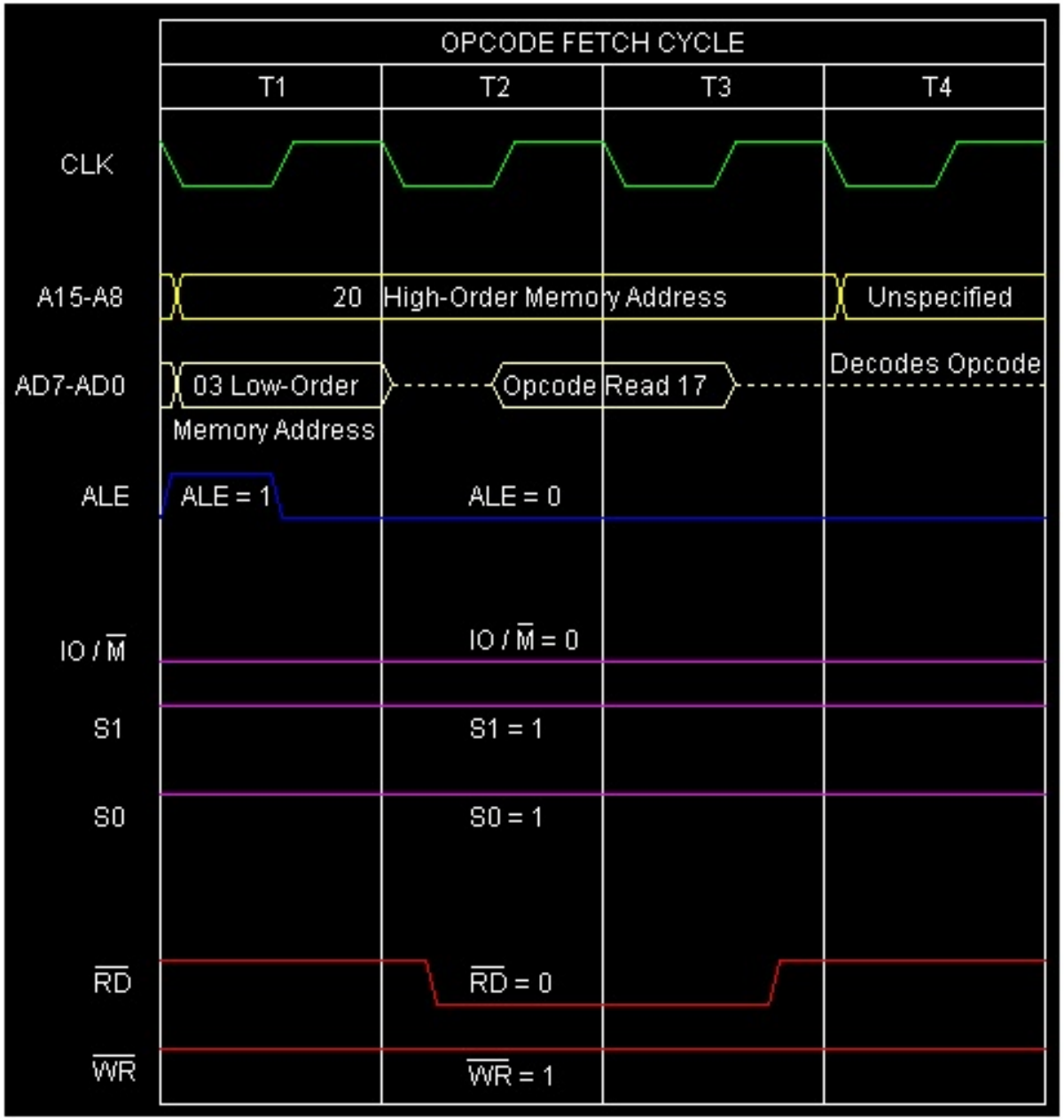


$A_0 \leftarrow Cy$
$A_n \leftarrow A_{n-1}$
$Cy \leftarrow A_7$

	Before	After
(A)	35H	6BH
(Cy)	1	0

35H ---> 0011 0101
0 0110 1011 ---> 6BH(Prev. Cy bit as the Last bit)

Address	Hex Codes	Mnemonic	Comment
2003	17	RAL	RotateAccumulator Left



So this instruction **RAL** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Opcode	Operand	Description
RAR	None	Rotate accumulator right through carry

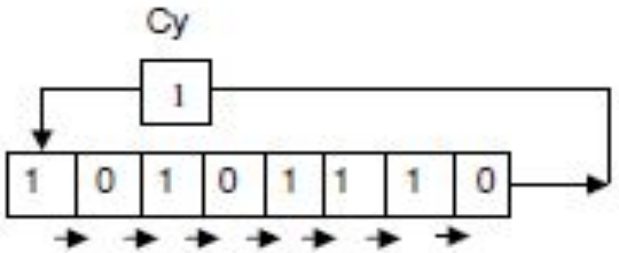
- Each binary bit of the accumulator is rotated right by one position through the Carry flag.
- Bit Do is placed in the Carry flag, and the Carry flag is placed in the most significant position D7.
- CY is modified according to bit Do.
- S, Z, P, AC are not affected.
- **Example:** RAR.

Logical Instructions

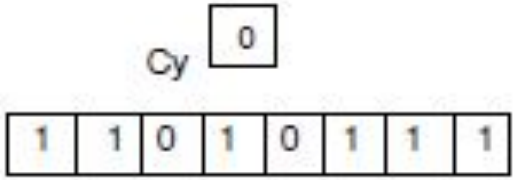
5

RAR (Rotate Accumulator Right through Carry)

Every bit of the Accumulator is shifted one bit Right and the LSB bit of the Accumulator is copied into the Carry flag and the Carry flag value is copied into the A7th bit



After the RAR instruction is executed the content of the Accumulator and the carry flag will be

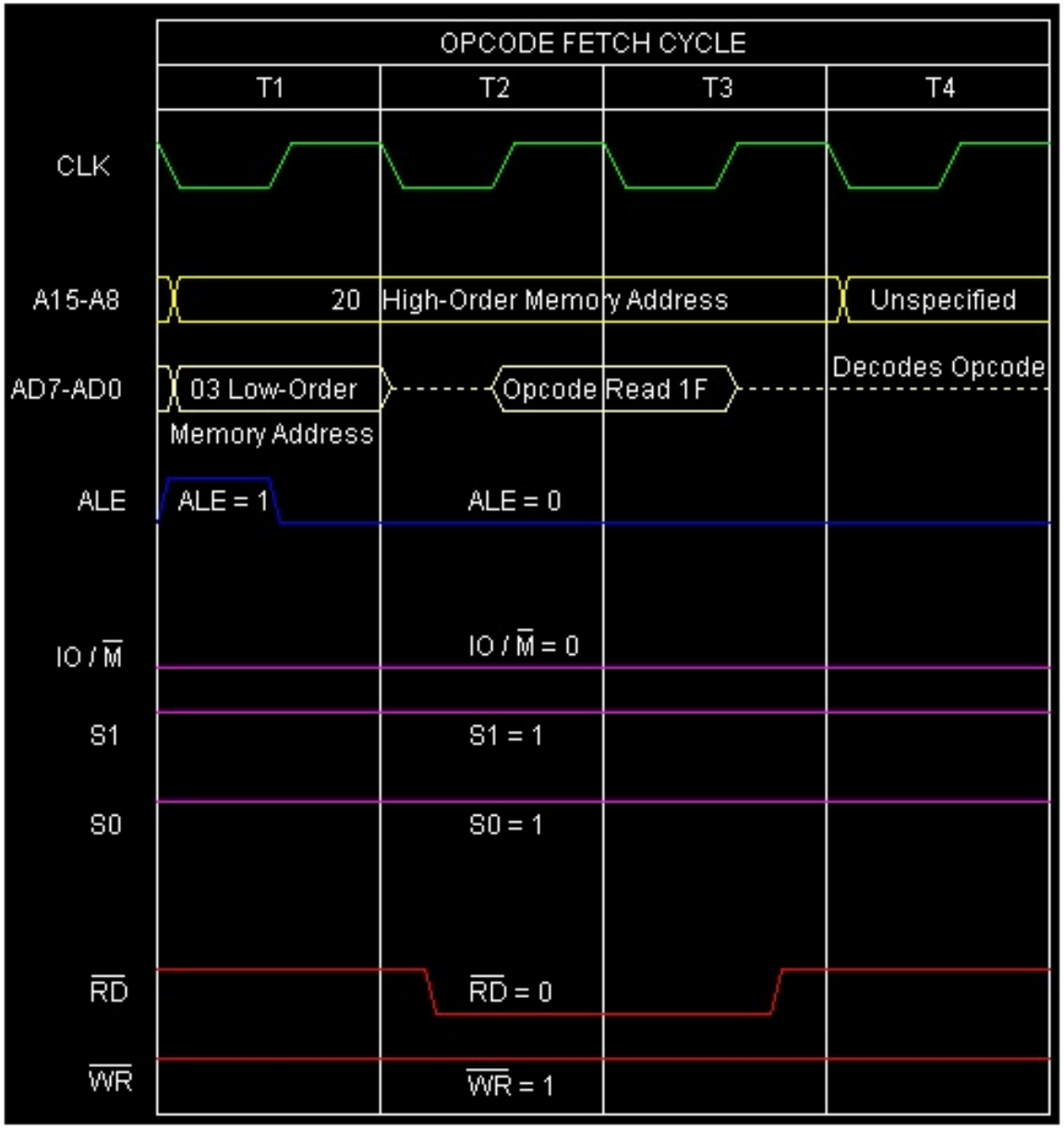


$A_7 \leftarrow Cy$
$An \leftarrow An+1$
$Cy \leftarrow A_0$

	Before	After
(A)	8AH	C5H
(Cy)	1	1

8AH ---> 1000 1011
1100 0101 --->C5H (Last bit 1 is copied to Cy bit) (And initial Cy=1 bit at the MS place)

Address	Hex Codes	Mnemonic	Comment
2003	1F	RAR	Rotate Accumulator Right



So this instruction **RAR** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

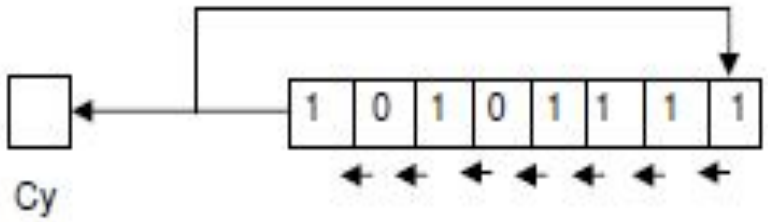
Opcode	Operand	Description
RLC	None	Rotate accumulator left

- Each binary bit of the accumulator is rotated left by one position.
- Bit D7 is placed in the position of Do as well as in the Carry flag.
- CY is modified according to bit D7.
- S, Z, P, AC are not affected.
- **Example:** RLC.

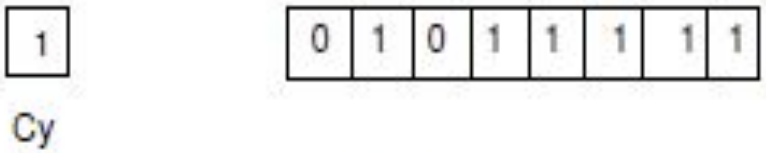
Logical Instructions

RLC (Rotate Accumulator Left)

Every bit of the Accumulator is shifted one bit left and the MSB bit of the Accumulator is copied into the Carry flag and into the A0th bit



After the RLC instruction is executed the content of the Accumulator and the carry flag will be

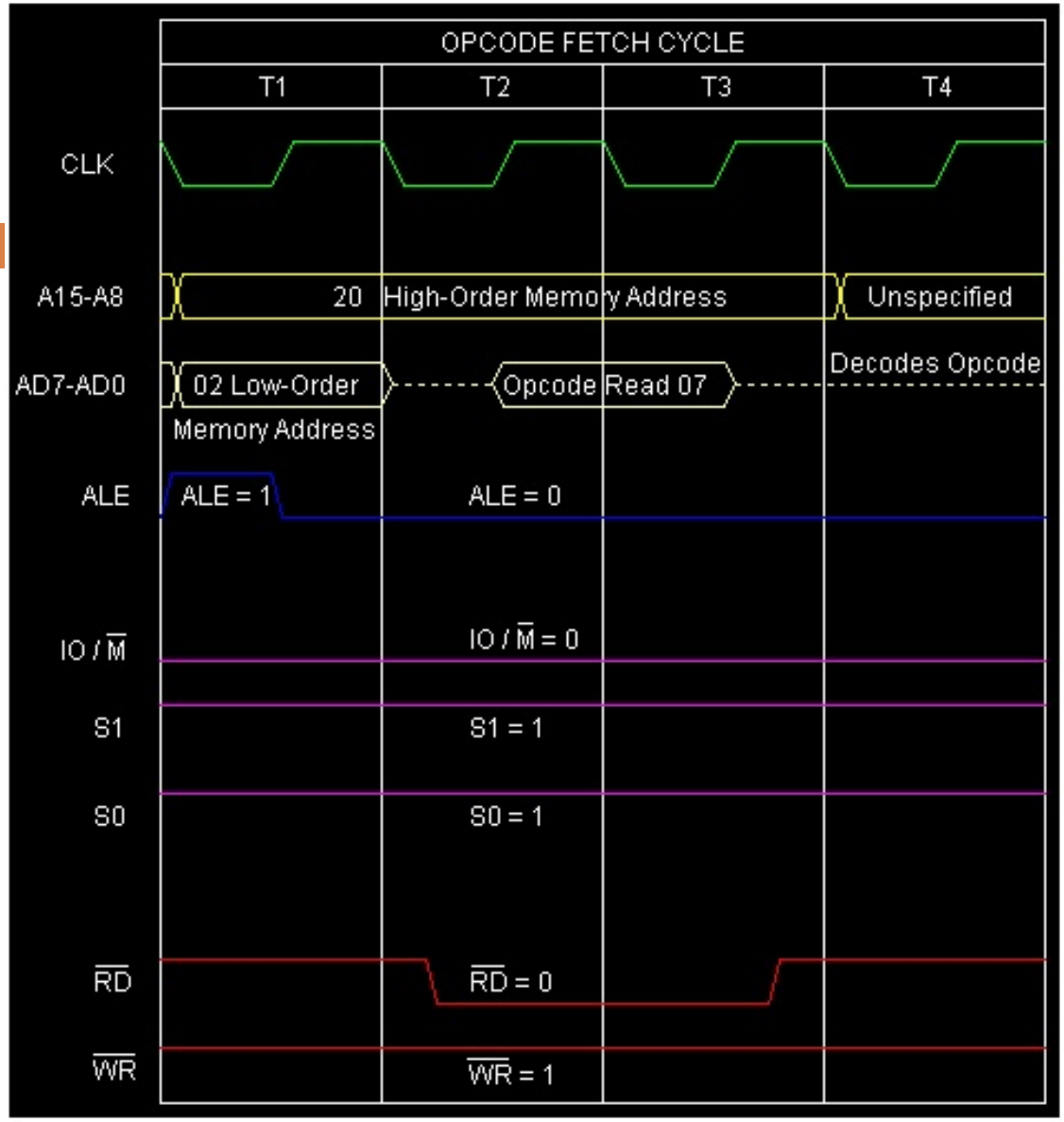


$A_0 \leftarrow A_7$
$A_n \leftarrow A_{n-1}$
$Cy \leftarrow A_7$

	Before	After
(A)	35H	6AH
(Cy)	Any value	0

35H ---> 0011 0101
0 0110 1010 ---> 6AH

Address	Hex Codes	Mnemonic	Comment
2002	07	RLC	Rotate Left Accumulator



So this instruction **RLC** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Opcode	Operand	Description
RRC	None	Rotate accumulator right

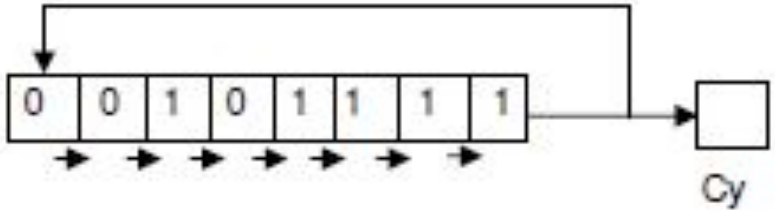
- Each binary bit of the accumulator is rotated right by one position.
- Bit Do is placed in the position of D7 as well as in the Carry flag.
- CY is modified according to bit Do.
- S, Z, P, AC are not affected.
- **Example:** RRC.

Logical Instructions

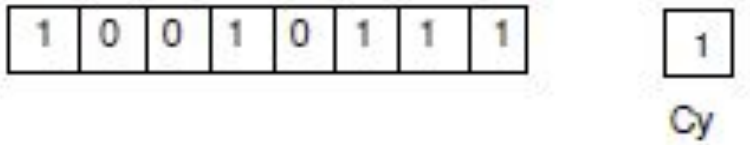
6

RRC (Rotate Accumulator Right)

Every bit of the Accumulator is shifted one bit Right and the LSB bit of the Accumulator is copied into the Carry flag and into the A7th bit



After the RRC instruction is executed the content of the Accumulator and the carry flag will be

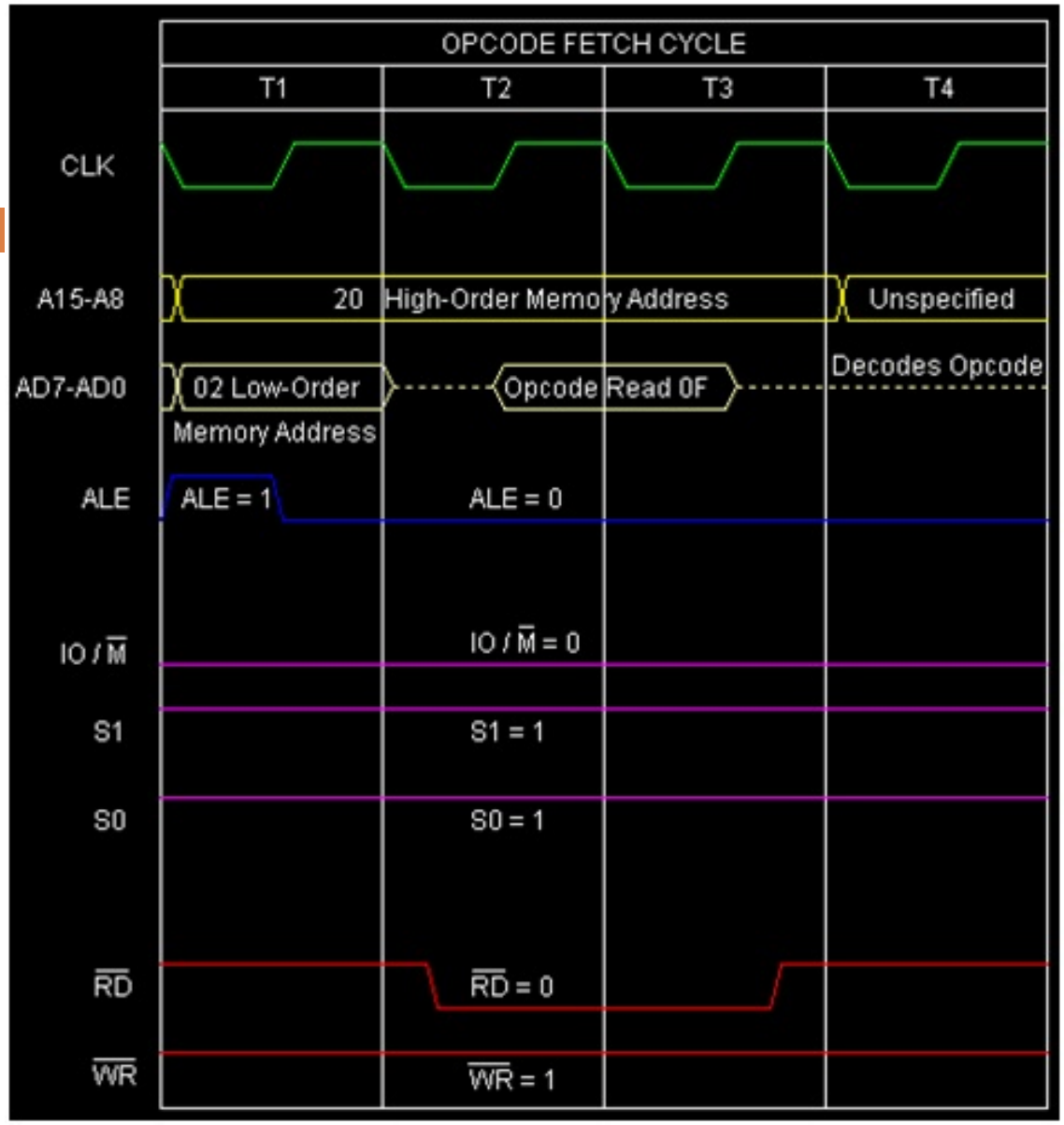


$A_7 \leftarrow A_0$
$A_n \leftarrow A_{n+1}$
$Cy \leftarrow A_0$

	Before	After
(A)	8AH	45H
(Cy)	Any Value	0

8AH ---> 1000 1010
0100 0101 ---> 45H (Last bit 0 is copied to Cy bit)

Address	Hex Codes	Mnemonic	Comment
2002	0F	RRC	Rotate Right Accumulator



So this instruction **RRC** requires 1-Byte, 1-Machine Cycle (Opcode Fetch) and 4 T-States for execution as shown in the timing diagram.

Logical Instructions

Opcode	Operand	Description
CMA	None	Complement accumulator

- The contents of the accumulator are complemented.
- No flags are affected.
- **Example:** CMA.

Logical Instructions

6

Opcode	Operand	Description
CMC	None	Complement carry

- The Carry flag is complemented.
- No other flags are affected.
- **Example:** CMC.

Logical Instructions

6

Opcode	Operand	Description
STC	None	Set carry

- The Carry flag is set to 1.
- No other flags are affected.
- **Example:** STC.

Branching Instructions

- The branching instruction alter the normal sequential flow.
- These instructions alter either unconditionally or conditionally.

Branching Instructions

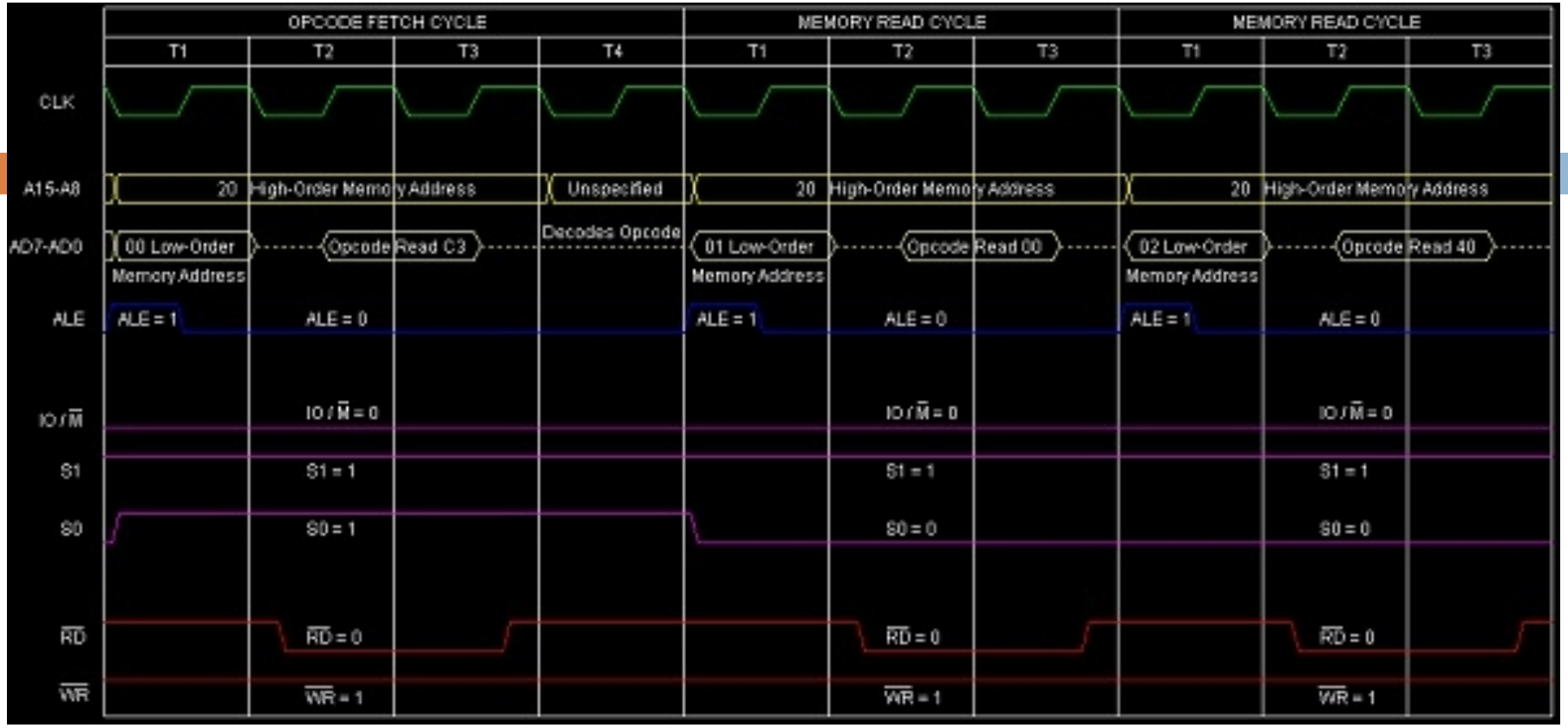
6

Opcode	Operand	Description
JMP	16-bit address	Jump unconditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand.
- **Example:** JMP 2034 H.

Opcode	Operand	Meaning	Explanation
JMP	16-bit address	Jump unconditionally	The program sequence is transferred to the memory address given in the operand.

Address	Hex Codes	Mnemonic	Comment
2000	C3	JMP 4000H	Unconditionally Jump to the memory address 4000H
2001	00		Low order Byte of the address
2002	40		High order Byte of the address



So this instruction **JMP** requires 3-Bytes, 3-Machine Cycles (Opcode Fetch, Memory Read, MemoryRead) and 10 T-States for execution as shown in the timing diagram.

Branching Instructions

Opcode	Operand	Description
Jx	16-bit address	Jump conditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW.
- **Example:** JZ 2034 H.

Timing diagram will be same for all as JMP

Jump Instructions

Opcode	Description	Status Flags
JC	Jump if Carry	CY = 1
JNC	Jump if No Carry	CY = 0
JP	Jump if Positive	S = 0
JM	Jump if Minus	S = 1
JZ	Jump if Zero	Z = 1
JNZ	Jump if No Zero	Z = 0
JPE	Jump if Parity Even	P = 1
JPO	Jump if Parity Odd	P = 0

Timing diagram will be same for all as JMP

Branching Instructions

Opcode	Operand	Description
CALL	16-bit address	Call unconditionally

- The program sequence is transferred to the memory location specified by the 16-bit address given in the operand.
- Before the transfer, the address of the next instruction after CALL (the contents of the program counter) is pushed onto the stack.
- **Example:** CALL 2034 H.

Call Instructions--Conditional

7

Opcode	Description	Status Flags
CC	Call if Carry	CY = 1
CNC	Call if No Carry	CY = 0
CP	Call if Positive	S = 0
CM	Call if Minus	S = 1
CZ	Call if Zero	Z = 1
CNZ	Call if No Zero	Z = 0
CPE	Call if Parity Even	P = 1
CPO	Call if Parity Odd	P = 0

Branching Instructions

7

Opcode	Operand	Description
RET	None	Return unconditionally

- The program sequence is transferred from the subroutine to the calling program.
- The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address.
- **Example:** RET.

Return Instructions--Conditional

Opcode	Description	Status Flags
RC	Return if Carry	CY = 1
RNC	Return if No Carry	CY = 0
RP	Return if Positive	S = 0
RM	Return if Minus	S = 1
RZ	Return if Zero	Z = 1
RNZ	Return if No Zero	Z = 0
RPE	Return if Parity Even	P = 1
RPO	Return if Parity Odd	P = 0

Control Instructions

7

Opcode	Operand	Description
NOP	None	No operation

- No operation is performed.
- The instruction is fetched and decoded but no operation is executed.
- **Example:** NOP

Control Instructions

Opcode	Operand	Description
HLT	None	Halt

- The CPU finishes executing the current instruction and halts any further execution.
- An interrupt or reset is necessary to exit from the halt state.
- **Example:** HLT

Control Instructions

Opcode	Operand	Description
DI	None	Disable interrupt

- The interrupt enable flip-flop is reset and all the interrupts except the TRAP are disabled.
- No flags are affected.
- **Example:** DI

Control Instructions

7

Opcode	Operand	Description
EI	None	Enable interrupt

- The interrupt enable flip-flop is set and all interrupts are enabled.
- No flags are affected.
- This instruction is necessary to re-enable the interrupts (except TRAP).
- **Example:** EI

Stack, I/O, and Machine Control Instructions.

Stack related instruction

Opcode	Operand	Description
SPHL	None	Copy H-L pair to the Stack Pointer (SP)

- This instruction loads the contents of H-L pair into SP.
- **Example:** SPHL

Stack, I/O, and Machine Control Instructions cont..

Opcode	Operand	Description
XTHL	None	Exchange H–L with top of stack

- The contents of L register are exchanged with the location pointed out by the contents of the SP.
- The contents of H register are exchanged with the next location (SP + 1).
- **Example:** XTHL

Stack, I/O, and Machine Control Instructions cont..

Opcode	Operand	Description
PCHL	None	Load program counter with H-L contents

- The contents of registers H and L are copied into the program counter (PC).
- The contents of H are placed as the high-order byte and the contents of L as the low-order byte.
- **Example:** PCHL

Stack, I/O, and Machine Control Instructions cont..

Opcode	Operand	Description
PUSH	Reg. pair	Push register pair onto stack

- The contents of register pair are copied onto stack.
- SP is decremented and the contents of high-order registers (B, D, H, A) are copied into stack.
- SP is again decremented and the contents of low-order registers (C, E, L, Flags) are copied into stack.
- **Example:** PUSH B

Stack, I/O, and Machine Control Instructions cont..

Opcode	Operand	Description
POP	Reg. pair	Pop stack to register pair

- The contents of top of stack are copied into register pair.
- The contents of location pointed out by SP are copied to the low-order register (C, E, L, Flags).
- SP is incremented and the contents of location are copied to the high-order register (B, D, H, A).
- **Example:** POP H

Stack, I/O, and Machine Control Instructions cont..

I/O related instruction

Opcode	Operand	Description
OUT	8-bit port address	Copy data from accumulator to a port with 8-bit address

- The contents of accumulator are copied into the I/O port.
- **Example:** OUT 78 H

Stack, I/O, and Machine Control Instructions cont..

Data Transfer Instructions

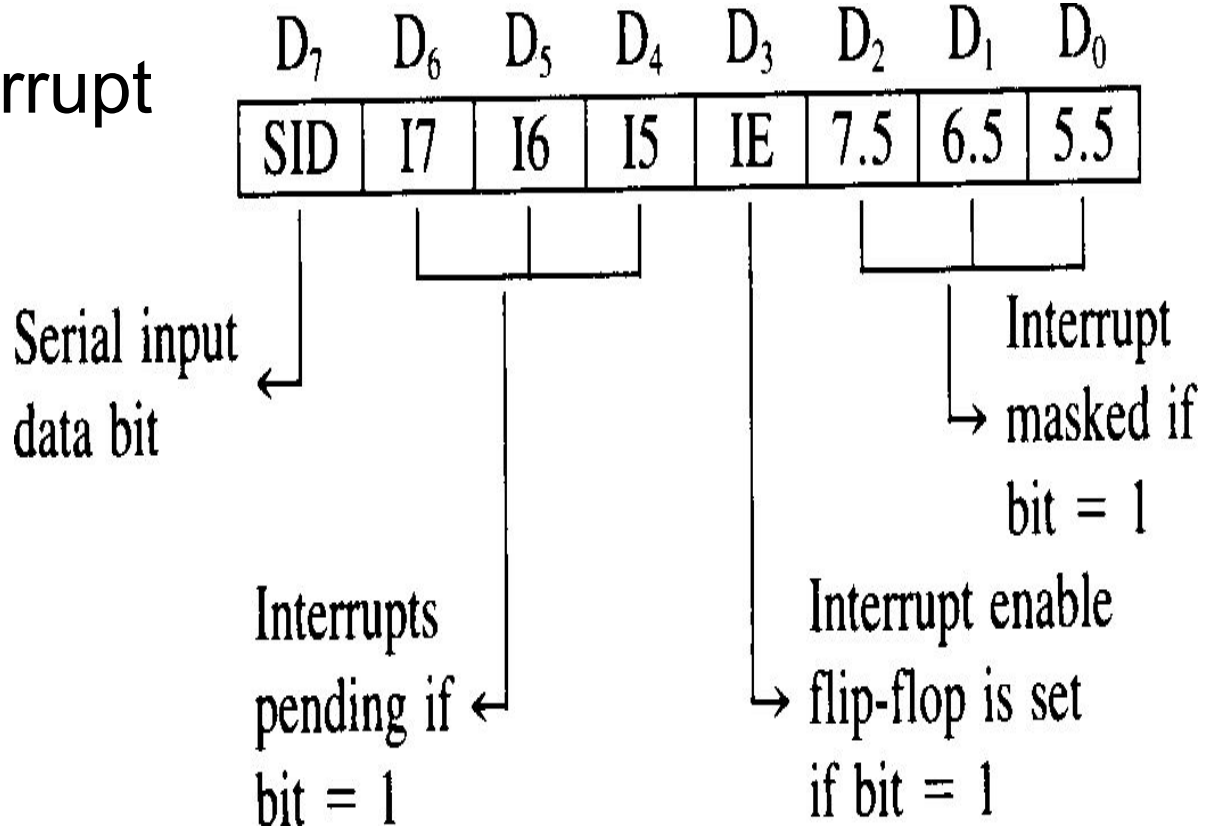
Opcode	Operand	Description
IN	8-bit port address	Copy data to accumulator from a port with 8-bit address

- The contents of I/O port are copied into accumulator.
- **Example:** IN 8C H

Stack, I/O, and Machine Control Instructions cont..

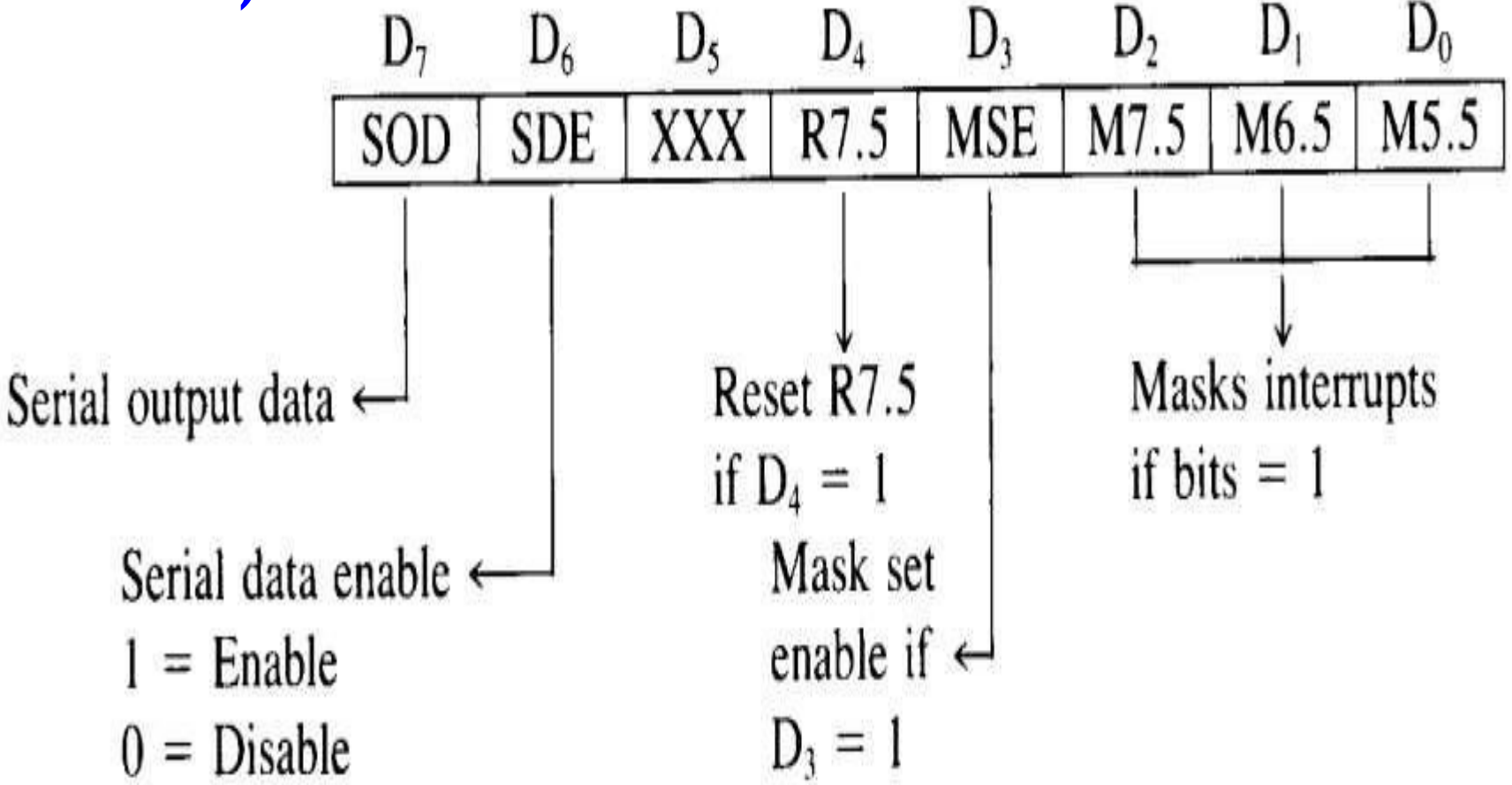
Machine control instruction

RIM (Read interrupt mask)



Stack, I/O, and Machine Control Instructions

- SIM (Set interrupt mask)



References

1. Gaonkar, R. S. (1990). *Microprocessor Architecture, Programming and Applications with the 8085*. Fifth Edition Prentice Hall PTR.