Instruction Set of 8085

- An instruction is a command to perform a particular task by the microprocessor
- The entire group of instructions that a microprocessor supports is called **Instruction Set**
- Each instruction is represented by an 8-bit binary value
- These 8-bits of binary value is called Opcode

Classification of Instruction

- Data Transfer Instructions
- Arithmetic Instructions
- Logical Instructions
- Branch control Instructions
- Stack, I/O and Machine Control Instructions

Data Transfer Instructions

- These instructions move data between registers, or between memory and registers.
- These instructions copy data from source to destination(without changing the original data)

MOV-Copy from source to destination

Opcode	Operand	
M O V	**Rd, Rs**	Register addressing , 1 m/c cycle, 4T states
	M, Rs	Register indirect addressing, 2 m/c cycles,7T states
	Rd, M	

- This instruction copies the contents of the source register into the destination register. (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 MOV B,M

BEFORE EXECUTION

A	20	B	xx

MOV B,A

AFTER EXECUTION

A	20	B	20

A		F	
B	30	C	
D		E	
H	20	L	50

MOV M,B

A		F	
B	30	C	
D		E	
H	20	L	50

30

A		F	
B		C	
D		E	
H	20	L	50

MOV C,M

A		F	
B		C	40
D		E	
H	20	L	50

40

MVI-Move immediate data

Opcode	Operand	
MVI	Rd, Data	Immediate addressing , 2 m/c cycles, 7T states
	M, Data	Immediate/ reg indirect addressing, 3 m/c cycles, 10T states

- 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, 60H or MVI M, 40H

BEFORE EXECUTION

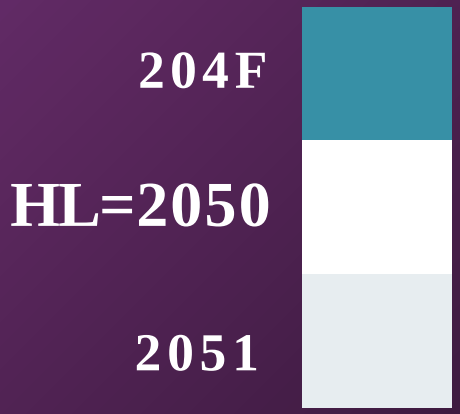
A		F
B		C
D		E
H		L

MVI B, 60H

AFTER EXECUTION

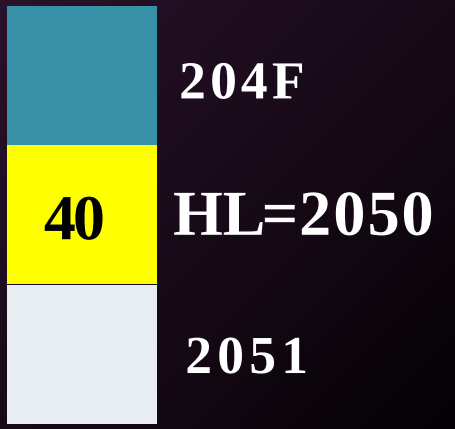
A		F
B	60	C
D		E
H		L

BEFORE EXECUTION



MVI M, 40H

AFTER EXECUTION



LDA-Load accumulator direct

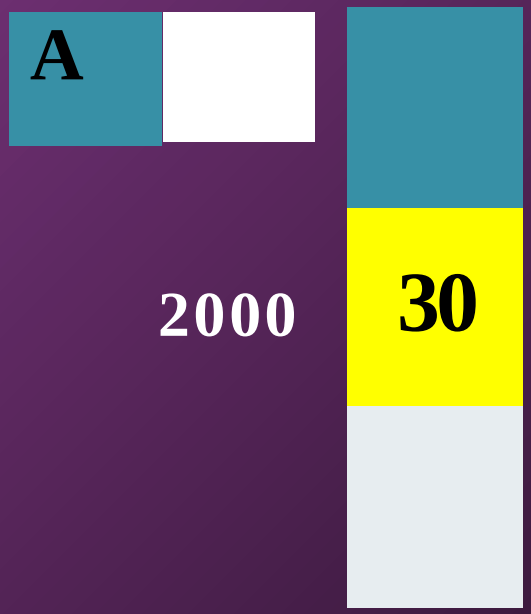
Opcode	Operand	
LDA	16-bit address	Direct addressing , 4 m/c cycles, 13T states

- 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.

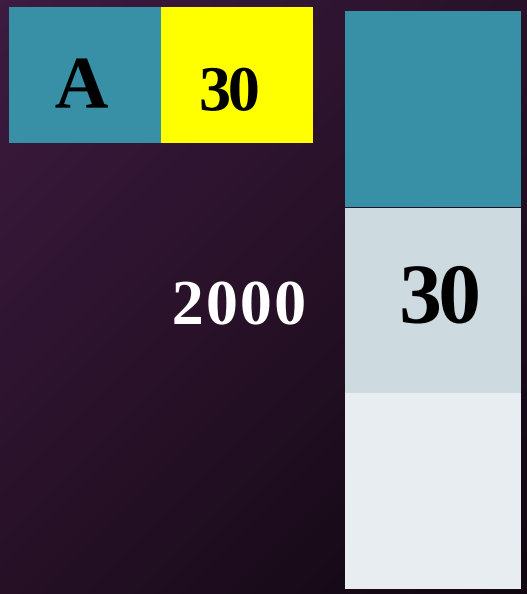
Example: LDA 2000H

LDA 2000H

BEFORE EXECUTION



AFTER EXECUTION



STA-Store accumulator direct

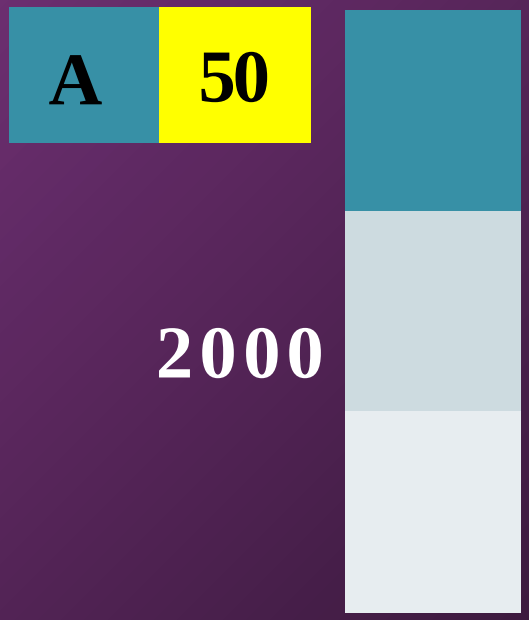
Opcode	Operand	
STA	16-bit address	Direct addressing , 4 m/c cycles, 13T states

- The contents of accumulator are copied into the memory location specified by the operand

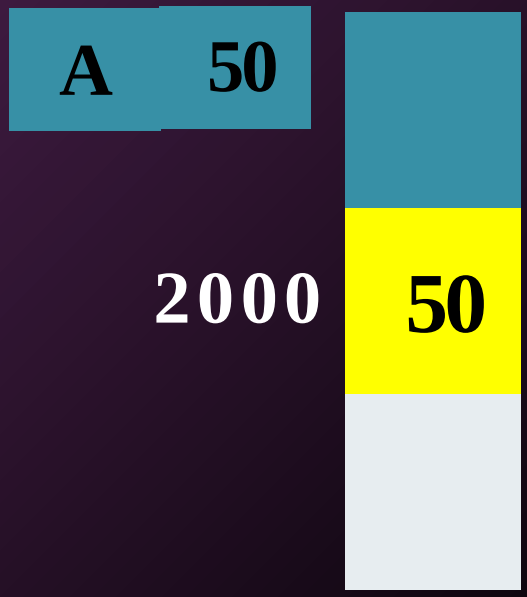
Example: STA 2000H

STA 2000H

BEFORE EXECUTION



AFTEREXECUTION



LHLD-Load H-L pairdirect

Opcode	Operand	
LHLD	16-bit address	Direct addressing , 5 m/c cycles, 16T states

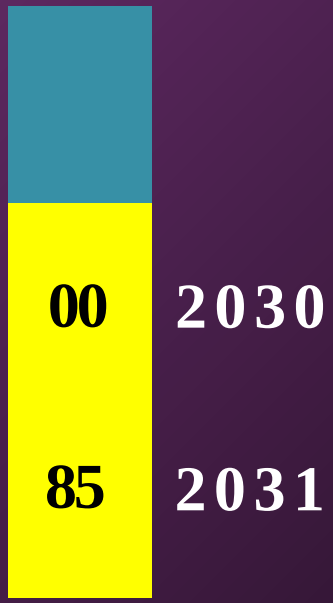
- This instruction copies the contents of memory location pointed out by 16-bit address into register L
- It copies the contents of next memory location into register H

Example: LHLD 2030 H

LHLD 2030H

BEFORE EXECUTION

A	F
B	C
H	L



AFTER EXECUTION

A			F
B			C
H	85	L	00

60

M=60

SHLD- Store H-L pair direct

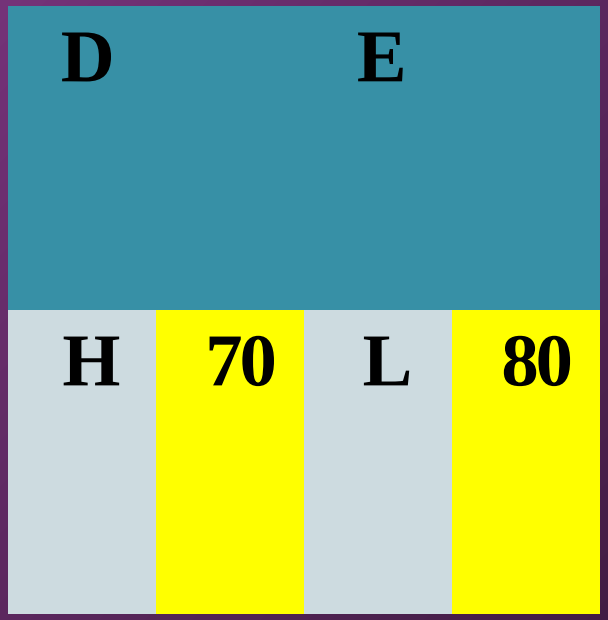
Opcode	Operand	
SHLD	16-bit address	Direct addressing , 5 m/c cycles, 16T states

- 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

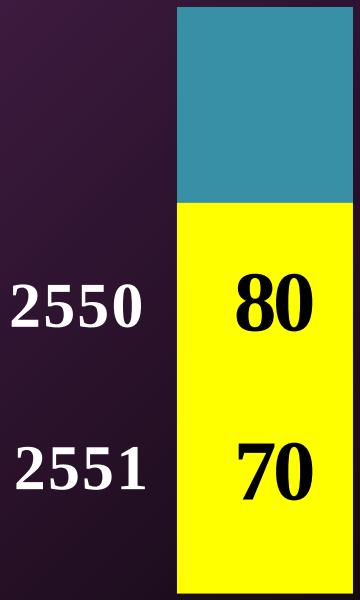
Example: SHLD 2550H

SHLD 2550H

BEFORE EXECUTION



AFTER EXECUTION



LXI-Load register pair immediate

Opcode	Operand	
LXI	Reg. pair, 16 bit data	Immediate addressing , 3 m/c cycles, 10T states

- This instruction loads 16-bit data in the register pair

Example: LXI H, 2030 H

LXIH, 2030H

BEFORE EXECUTION

A	F
B	C
H	L

AFTER EXECUTION

A		F	
B		C	
H	20	L	30

50

M=50

LDAX-Load accumulator indirect

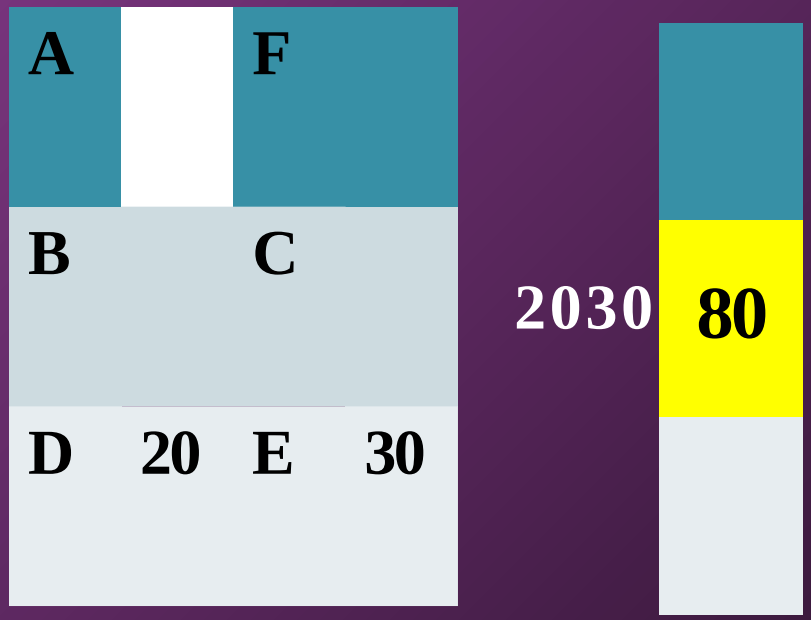
Opcode	Operand	
LDAX	B / D Register pair	Reg. indirect addressing , 2 m/c cycles, 7 T states

- 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.

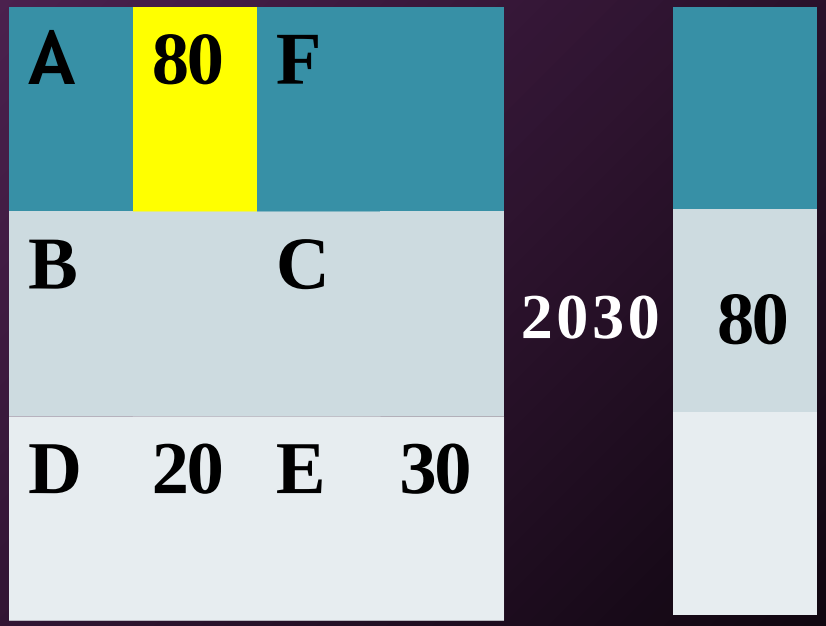
Example: L D A X D

LDAX D

BEFORE EXECUTION



AFTER EXECUTION



STAX-Store accumulator indirect

Opcode	Operand	
STAX	B / D Register pair	Reg. indirect addressing , 2 m/c cycles, 7 T states

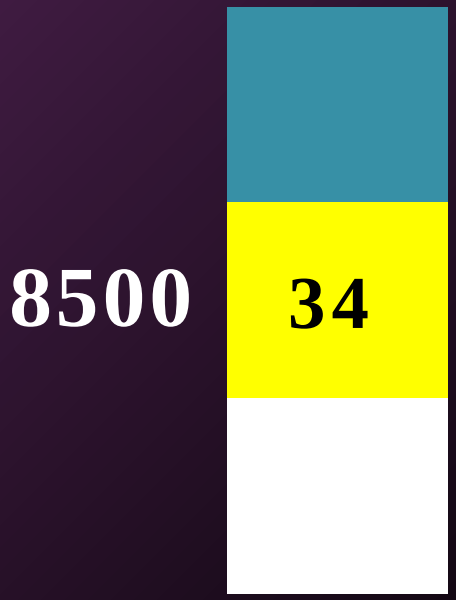
- The contents of accumulator are copied into the memory location specified by the contents of the register pair

Example: STAX B

STAX B

BEFORE EXECUTION

AFTER EXECUTION



XCHG- Exchange contents of H-L with D-E pair

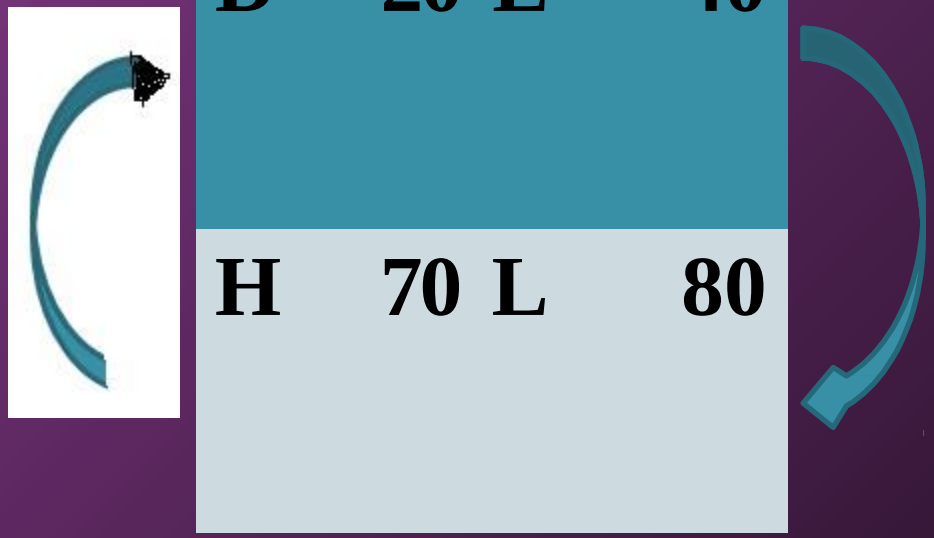
Opcode	Operand	
XCHG	none	Register addressing , 1 m/c cycle, 4 T states

- 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

XCHG

BEFORE EXECUTION



AFTER EXECUTION

D	70	E	80
H	20	L	40

Arithmetic Instructions

Arithmetic instructions perform operations like

- Addition
- Subtraction
- Increment
- Decrement

Addition

- Any 8-bit number, or the contents of register, or the contents of memory location can be added to the contents of accumulator
- The result (sum) is stored in the accumulator
- Two 8-bit registers cannot be added directly. **Ex:** The contents of register B cannot be added directly to the contents of register C
- Flags are affected

ADD-Add register or memory to accumulator

Opcode	Operand	
ADD	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 2 m/c cycles,7T states

- 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

BEFORE EXECUTION

A	04

B	C	05
D	E	
H	L	

ADD C

AFTER EXECUTION

A.	09

B.	C	05
D	E	
H	L	

BEFORE EXECUTION

A	04

B		C	
D		E	
H	20	L	50

10

2050

ADD M

AFTER EXECUTION

A	14

B		C	
D		E	
H	20	L	50

10

2050

ADC-Add register or memory with carry to accumulator

Opcode	Operand
ADC	R—— Register addressing , 1 m/c cycle, 4T states M——Register indirect addressing, 2 m/c cycles,7T states

- 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 EXECUTION

CY	01
A	50

B		C	05
D		E	
H		L	

ADC C

AFTER EXECUTION

A	56

B		C	05
D		E	
H		L	

BEFORE EXECUTION

CY	01
A	06

H	20	L	50

2050	30

ADC M

AFTER EXECUTION

A	37

H	20	L	50

2050	30

ADI-Add immediate data to accumulator

Opcode	Operand	
ADI	8 bit data	Immediate addressing , 2 m/c cycle, 7 T states

- 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 05H

BEFORE EXECUTION

A	03

AFTER EXECUTION

A	08

ACI-Add with carry immediate data to accumulator

Opcode	Operand	
ACI	8 bit data	Immediate addressing , 2 m/c cycle, 7 T states

- ✓ 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 45H

ACI 20H

BEFORE EXECUTION

CY	01
A	05

AFTER EXECUTION

A	26

DAD-Add register pair to H-L pair

Opcode	Operand	
DAD	Reg. pair	Register addressing , 3 m/c cycle, 10 T states

- 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 affected

Example: DAD B or DAD D

DAD D

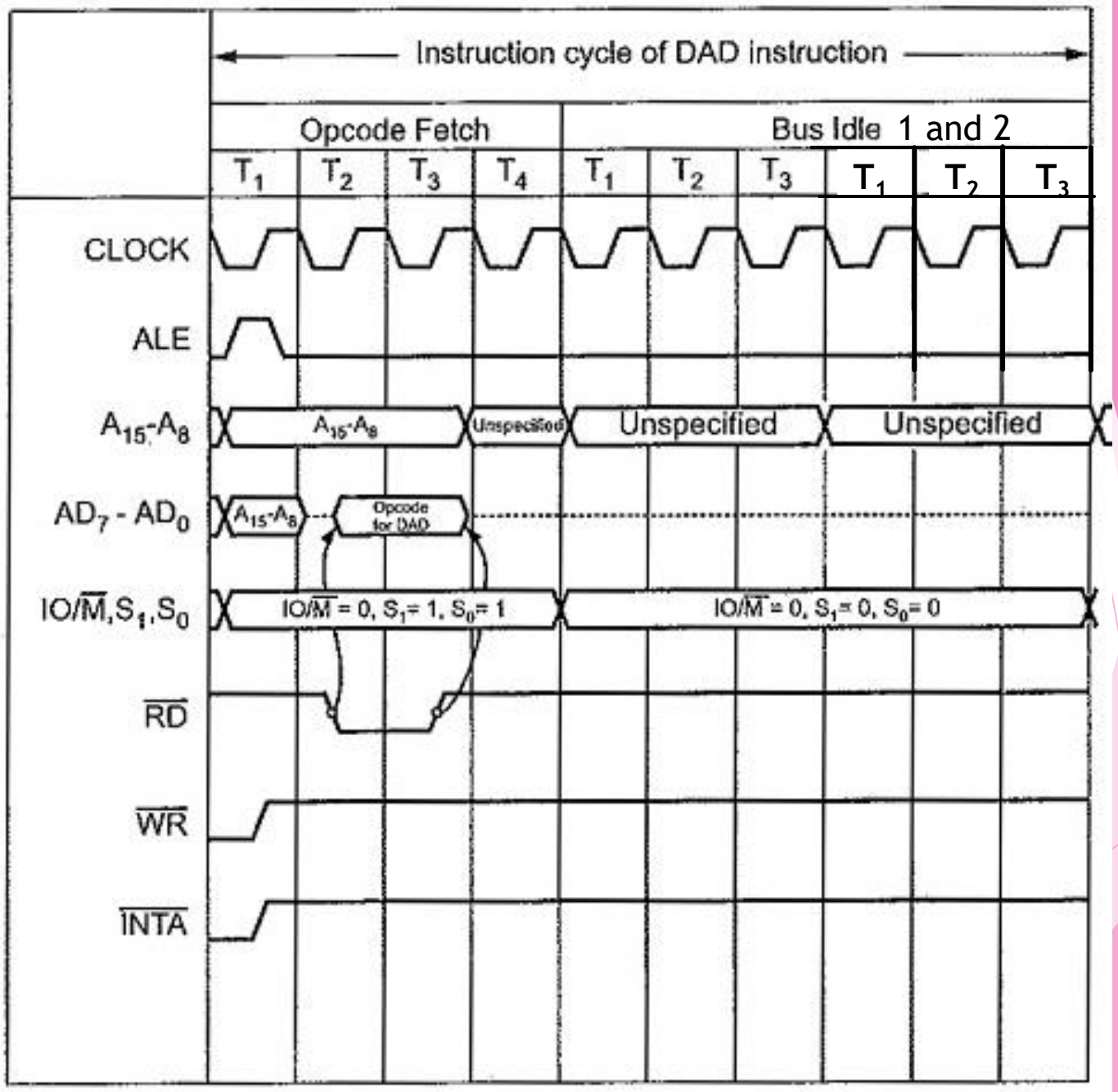
BEFORE EXECUTION

D	12	E	34
H	23	L	45

AFTER EXECUTION

D	12	E	34
H	35	L	79

```
1234
2345  +
-------
3579
```


Instruction cycle of DAD instruction
Opcode Fetch
Bus Idle 1 and 2
T1
T2
T3
T4
T1
T2
T3
T1
T2
T3
CLOCK
ALE
A15-A8
A15-A8
Unspecified
Unspecified
Unspecified
AD7 - AD0
A15-A8
Opcode for DAD
IO/M,S1,S0
IO/M = 0, S1 = 1, S0 = 1
IO/M = 0, S1 = 0, S0 = 0
RD
WR
INTA

Subtraction

- Any 8-bit number, the contents of register, or the contents of memory location can be subtracted from the contents of accumulator
- The result is stored in the accumulator
- Subtraction is performed in 2's complement form
- If the result is negative, it is stored in 2's complement form
- No two other 8-bit registers can be subtracted directly

SUB - Subtract register or memory from accumulator

Opcode	Operand	
SUB	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 2 m/c cycles,7T states

- The contents of register or memory are subtracted from 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 subtraction

Example: SUB B or SUB M

BEFORE EXECUTION

A	09		
B		C	04
D		E	
H		L	

SUB C

AFTER EXECUTION

A.	05		
B.		C	04
D		E	
H		L	

BEFORE EXECUTION

A	14		
B		C	
D		E	
H	20	L	50

10

2050

SUB M

AFTER EXECUTION

A	04		
B		C	
D		E	
H	20	L	50

10

2050

SBB-Subtract register or memory from accumulator with borrow

Opcode	Operand
SBB	R—— Register addressing , 1 m/c cycle, 4T states M——Register indirect addressing, 2 m/c cycles,7T states

- The contents of register or memory and Carry Flag (CY) are subtracted from 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 subtraction

Example: SBB B or SBB M

BEFORE EXECUTION

CY	01
A	08

B		C	05
D		E	
H		L	

SBB C

AFTER EXECUTION

A	02

B		C	05
D		E	
H		L	

BEFORE EXECUTION

CY	01
A	06

H	20	L	50

2050 | 02

SBB M

AFTER EXECUTION

A	03

H	20	L	50

2050 | 02

SUI-Subtract immediate data from accumulator

Opcode	Operand	
SUI	8 bit data	Immediate addressing , 2 m/c cycle, 7 T states

- The 8-bit data is subtracted from the contents of accumulator
- The result is stored in accumulator
- All flags are modified to reflect the result of the subtraction

Example: SUI 45H

SUI 05H

BEFORE EXECUTION

A	08

AFTER EXECUTION

A	03

SBI-Subtract immediate data from accumulator with borrow

Opcode	Operand	
SBI	8 bit data	Immediate addressing , 2 m/c cycle, 7 T states

- ✓ The 8-bit data and the Carry Flag (CY) are subtracted from the contents of accumulator
- ✓ The result is stored in accumulator
- ✓ All flags are modified to reflect the result of the subtraction

Example: SBI 45H

SBI 20H

BEFORE EXECUTION

CY	01
A	25

AFTER EXECUTION

A	04

Increment / Decrement

- The 8-bit contents of a register or a memory location can be incremented or decremented by 1
- The 16-bit contents of a register pair can be incremented or decremented by 1
- Increment or decrement can be performed on any register or a memory location

INR-Increment register or memory content

Opcode	Operand	
INR	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 3m/c cycles, 10T states

- 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

BEFORE EXECUTION

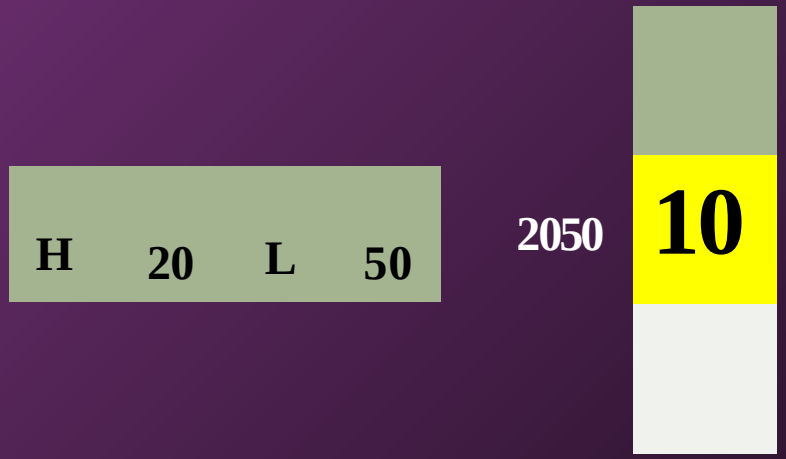
A		
B	10	C
D		E
H		L

INR B

AFTER EXECUTION

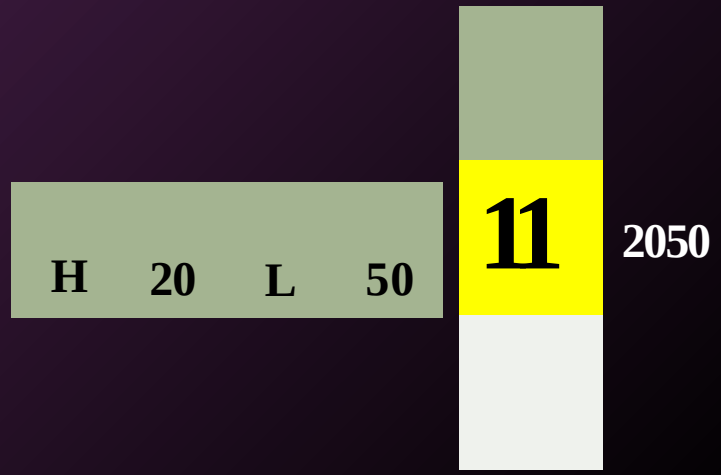
A		
B	11	C
D		E
H		L

BEFORE EXECUTION



INR M

AFTER EXECUTION



INX-Increment register pair

Opcode	Operand	
INX	Reg. pair	Register addressing , 1 m/c cycle, 6T states

- The contents of register pair are incremented by 1
- The result is stored in the same place

Example: INX H, INX B or INX D

INX H

BEFORE EXECUTION

B		C	
D		E	
H	10	L	20

AFTER EXECUTION

B		C	
D		E	
H	10	L	21

DCR-Decrement register or memory content

Opcode	Operand	
DCR	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 3m/c cycles, 10T states

- 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

BEFORE EXECUTION

A		
B	20	C
D		E
H		L

DCR B

AFTER EXECUTION

A		
B	1F	C
D		E
H		L

BEFORE EXECUTION

H	20	L	50

2050

21

DCR M

AFTER EXECUTION

H	20	L	50

20

2050

DCX-Decrement register pair

Opcode	Operand	
DCX	Reg. pair	Register addressing , 1 m/c cycle, 6T states

- The contents of register pair are decremented by 1
- The result is stored in the same place

Example: DCX H, DCX B or DCX D

DCX H

BEFORE EXECUTION

B		C	
D		E	
H	10	L	21

AFTER EXECUTION

B		C	
D		E	
H	10	L	20

Logical Instructions

- 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

ANA-Logical AND register or memory with accumulator

Opcode	Operand	
ANA	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 2m/c cycles, 7T states

- 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
- All flags are modified to reflect the result of the operation
- CY is reset and AC is set

Example: ANA B or ANAM

BEFORE EXECUTION

CY		AC	
A	AA		
B	0F	C	
D		E	
H		L	

1010 1010 = AA H
0000 1111 = 0F H

0000 1010 = 0A H

ANA B

AFTER EXECUTION

CY	0	AC	1
A	0A		
B	0F	C	
D		E	
H		L	

BEFORE EXECUTION

CY		AC		
A	55		2050H	B3
H	20	L	50	

0101 0101 = 55 H
1011 0011 = B3 H

0001 0001 = 11H

ANA M

AFTER EXECUTION

CY	0	AC	1	
A	11		2050H	B3
H	20	L	50	

ANI-Logical AND immediate data with accumulator

Opcode	Operand	
ANI	8 bit data	Immediate addressing , 2 m/c cycles, 7T states

- The contents of the accumulator are logically ANDed with the 8-bit data
- The result is placed in the accumulator
- All flags are modified to reflect the result
- CY is reset, AC is set

Example: ANI 86H

ANI 3FH

BEFORE EXECUTION

CY	AC
A	B3

$1011\ 0011 = B3_H$

$0011\ 1111 = 3F_H$

$0011\ 0011 = 33_H$

AFTER EXECUTION

CY	0	AC	1
	33		

ORA-Logical OR register or memory with accumulator

Opcode	Operand	
ORA	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 2m/c cycles, 7T states

- The contents of the accumulator are logically OR-ed 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
- All flags are modified to reflect the result of the operation
- CY and AC are reset

Example: ORA B or ORA M

BEFORE EXECUTION

CY		AC	
A	AA		
B	12	C	
D		E	
H		L	

1010 1010 = AA H
0001 0010 = 12 H

1011 1010 = BA H

ORA B

AFTER EXECUTION

CY	0	AC	0
A	BA		
B	12	C	
D		E	
H		L	

BEFORE EXECUTION

CY		AC			
A	55		2050H		B3
H	20	L	50		

0101 0101 = 55 H
1011 0011 = B3 H

1111 0111 = F7H

ORA M

AFTER EXECUTION

CY	0	AC	0		
A	F7		2050H		B3
H	20	L	50		

ORI-Logical OR immediate data with accumulator

Opcode	Operand	
ORI	8 bit data	Immediate addressing , 2 m/c cycles, 7T states

- The contents of the accumulator are logically ORed with the 8-bit data
- The result is placed in the accumulator
- All flags are modified to reflect the result
- CY and AC are reset

Example: ORI 86H

ORI 08 H

BEFORE EXECUTION

CY	AC
A	B3

$1011\ 0011 = B3_H$

$0000\ 1000 = 08_H$

$1011\ 1011 = BB_H$

AFTER EXECUTION

CY	0	AC	0
	BB		

XRA-Logical XOR register or memory with accumulator

Opcode	Operand	
XRA	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 2m/c cycles, 7T states

- The contents of the accumulator are logically XORed 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
- All flags are modified to reflect the result of the operation
- CY and AC are reset

Example: XRA B or XRAM

BEFORE EXECUTION

CY		AC	
A	AA		
B	2D	C	
D		E	
H		L	

1010 1010 = AA H
0010 1101 = 2D H

1000 0111 = 87 H

XRA B

AFTER EXECUTION

CY	0	AC	0
A	87		
B	2D	C	
D		E	
H		L	

BEFORE EXECUTION

CY		AC	
A	55		
H	20	L	50

2050H

B3

0101 0101 = 55 H
1011 0011 = B3 H

1110 0110 = E6H

XRA M

AFTER EXECUTION

CY	0	AC	0
A	E6		
H	20	L	50

2050H

B3

XRI-Logical XOR immediate data with accumulator

Opcode	Operand	
XRI	8 bit data	Immediate addressing , 2 m/c cycles, 7T states

- The contents of the accumulator are logically XORed with the 8-bit data
- The result is placed in the accumulator
- All flags are modified to reflect the result
- CY and AC are reset

Example: **XRI** 86H

XRI 39 H

BEFORE EXECUTION

CY	AC
A	B3

$1011\ 0011 = B3_H$

$0011\ 1001 = 39_H$

$1000\ 1010 = 8A_H$

AFTER EXECUTION

CY	0	AC	0
	8A		

Compare

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

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

CMP-Compare register or memory with accumulator

Opcode	Operand	
CMP	R———	Register addressing , 1 m/c cycle, 4T states
	M———	Register indirect addressing, 2m/c cycles, 7T states

- ✓ The contents of the operand (register or memory) are subtracted from the contents of the accumulator
- ✓ Both contents are preserved
- ✓ All flags are affected according to the result of subtraction

BEFORE EXECUTION

CY		Z	
A	10		
B		C	
D	20	E	
H		L	

A>R: CY = 0
A=R: ZF = 1
A<R: CY=1

CMP D

AFTER EXECUTION

CY	01	Z	0
A	10		
B		C	
D	20	E	
H		L	

BEFORE EXECUTION

CY		Z	
A	B8	2050H	
H	20	L	50

B8

A>M: CY = 0
A=M: ZF = 1
A<M: CY=1

CMP M

AFTER EXECUTION

CY	0	ZF	1
A	B8		
H	20	L	50

B8

B8=B8 :ZF=01

CPI-Compare immediate data with accumulator

Opcode	Operand	
CPI	8 bit data	Immediate addressing , 2 m/c cycles, 7T states

- The 8-bit data is subtracted from the contents of accumulator
- The values being compared remain unchanged
- All flags are affected by the result of subtraction

CPI 30H

BEFORE EXECUTION

CY	Z
A	BA

AFTER EXECUTION

CY 0	ZF 0
A	BA

A>DATA: CY = 0
A=DATA: ZF = 1
A<DATA: CY=1

BA>30 : CY=00

Rotate

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

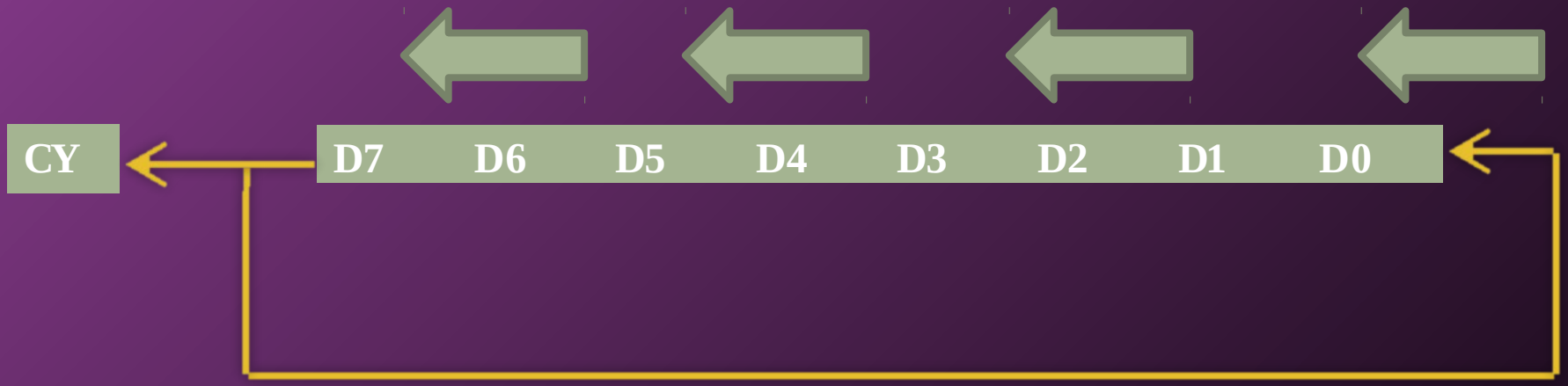
RLC-Rotate accumulator left

Opcode	Operand	
RLC	none	Implicit addressing , 1 m/c cycle, 4T states

- Each binary bit of the accumulator is rotated left by one position
- Bit D7 is placed in the position of D0 as well as in the Carry flag
- CY is modified according to bit D7
- S, Z, P, AC are not affected

Example: RLC

BEFORE EXECUTION



AFTER EXECUTION

D7	D6	D5	D4	D3	D2	D1	D0	D7

RRC-Rotate accumulator right

Opcode	Operand	
RRC	none	Implicit addressing , 1 m/c cycle, 4T states

- Each binary bit of the accumulator is rotated right by one position
- Bit D0 is placed in the position of D7 as well as in the Carry flag
- CY is modified according to bit D0
- S, Z, P, AC are not affected

Example: RRC

BEFORE EXECUTION

AFTER EXECUTION

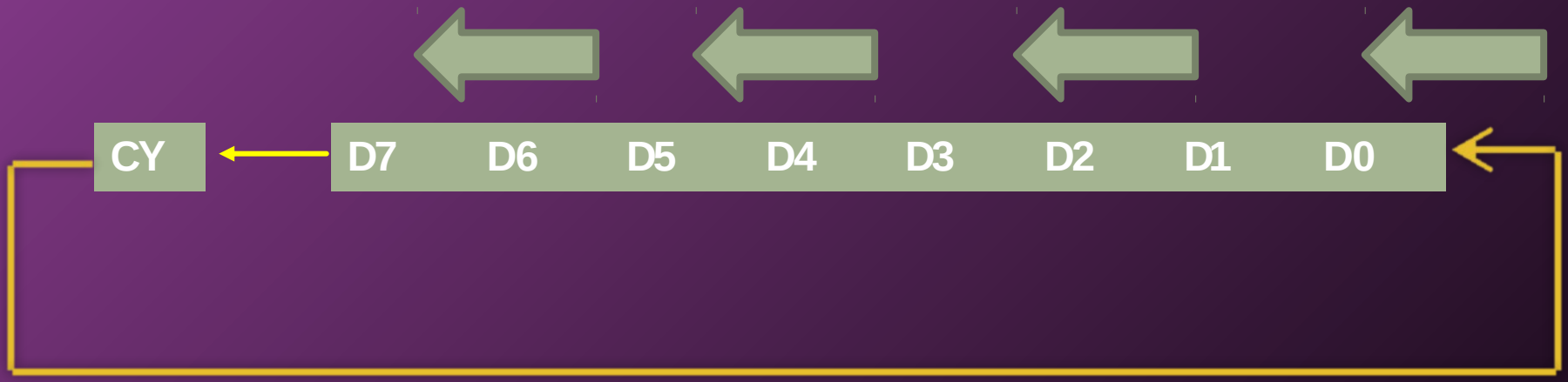
RAL-Rotate accumulator left through carry

Opcode	Operand	
RAL	none	Implicit addressing , 1 m/c cycle, 4T states

- 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 D0
- CY is modified according to bit D7
- S, Z, P, AC are not affected

Example: RAL

BEFORE EXECUTION



AFTER EXECUTION

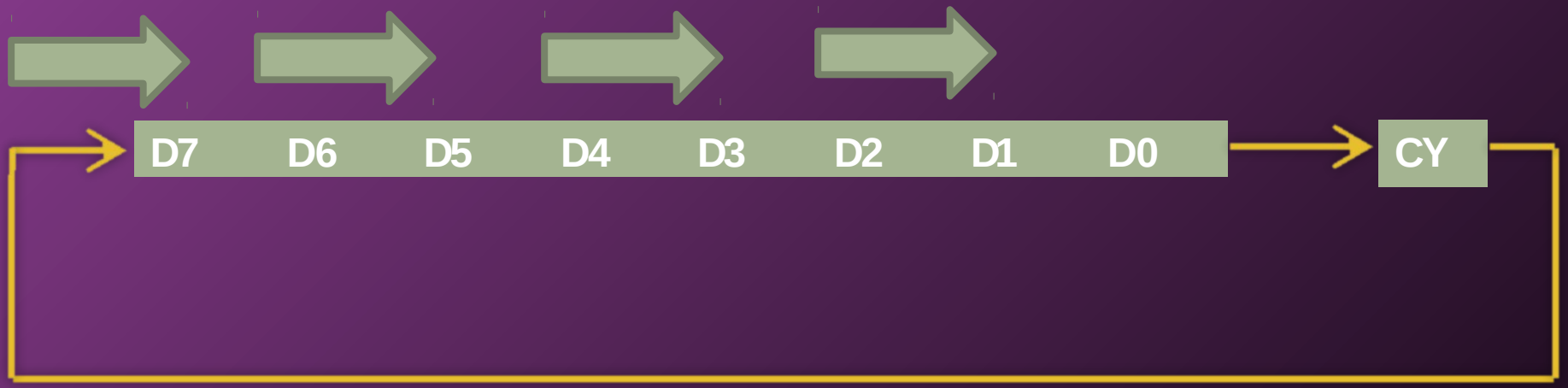
RAR-Rotate accumulator right through carry

Opcode	Operand	
RAR	none	Implicit addressing , 1 m/c cycle, 4T states

- Each binary bit of the accumulator is rotated right by one position through the carry flag
- Bit D0 is placed in the Carry flag, and the Carry flag is placed in the least significant position D7
- CY is modified according to bit D0
- S, Z, P, AC are not affected

Example: RAR

BEFORE EXECUTION



AFTER EXECUTION

Complement

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

CMA-Complement accumulator

Opcode	Operand	
CMA	none	Implicit addressing , 1 m/c cycle, 4T states

- The contents of the accumulator are complemented
- No flags are affected

Example: CMA

BEFORE EXECUTION

A	00

AFTER EXECUTION

A	FF

CMC-Complement carry

Opcode	Operand	
CMC	none	1 m/c cycle, 4T states

- The Carry flag is complemented
- No other flags are affected

Example: CMC

BEFORE EXECUTION

CY	0

AFTER EXECUTION

CY	1

STC-Set carry

Opcode	Operand	
STC	none	1 m/c cycle, 4T states

- The Carry flag is set to 1
- No other flags are affected

Example: STC

Branch control instructions

The branching instructions allows the microprocessor to change the sequence of program either unconditionally or under certain test conditions. The group includes

(1) Jump instructions

(2) Call and Return instructions

(3) Restart instructions

JMP-Jump unconditionally

Opcode	Operand	
JMP	16 bit address	Immediate addressing , 3 m/c cycles, 10T states

The program sequence is transferred to the memory location specified by the 16-bit address given in the operand

Example: JMP 2034 H

JX-Jump conditionally

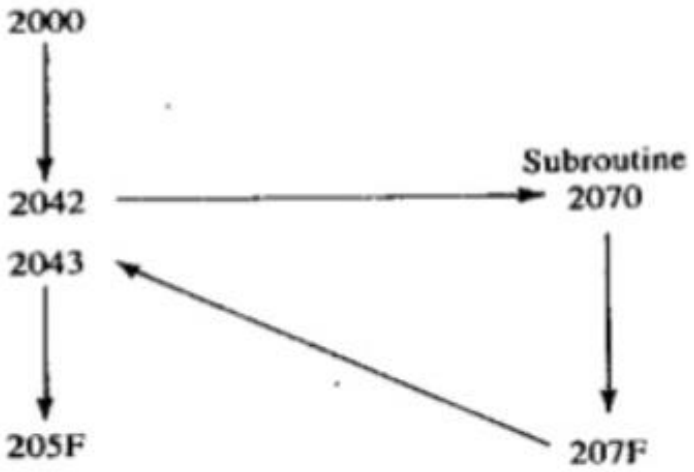
Opcode	Operand	
JX	16 bit address	Immediate addressing , 2/3 m/c cycles, 7/10T states

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

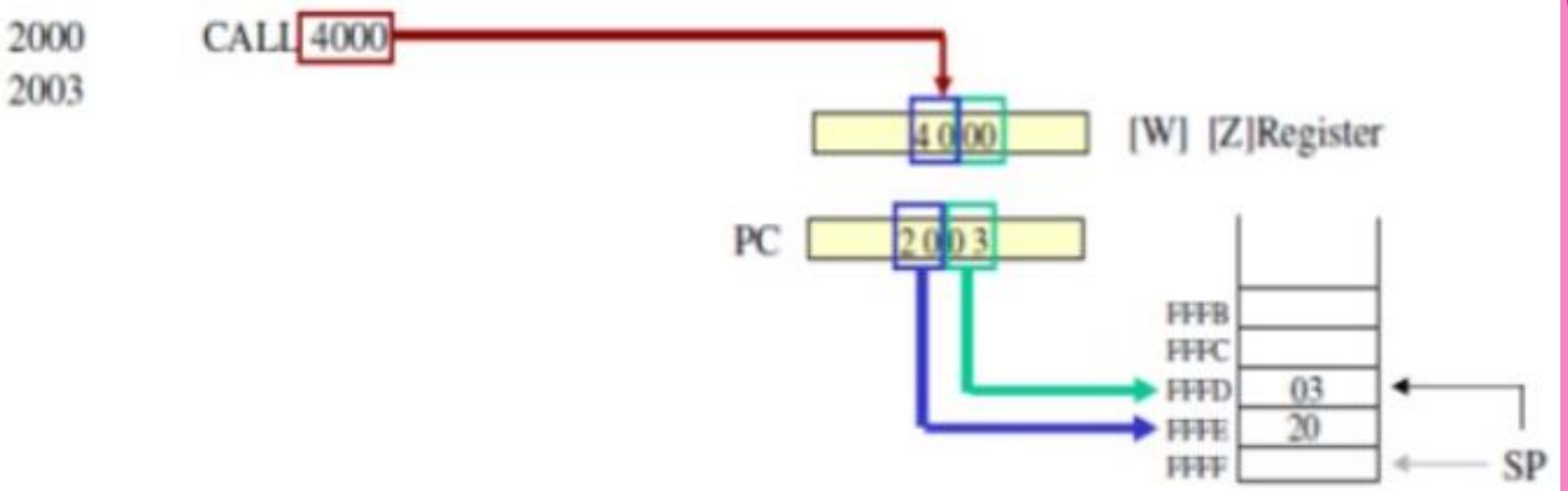
Opcode	Description	Status Flags
JC	Jump if Carry	CY = 1
JNC	Jump if No Carry	CY = 0
JZ	Jump if Zero	Z = 1
JNZ	Jump if Not Zero	Z = 0
JPE	Jump if Parity Even	P = 1
JPO	Jump if Parity Odd	P = 0
JP	Jump if Result positive	S=0
JM	Jump if Result negative	S=1

SUBROUTINE



Memory Address	Machine Code	Mnemonics	Comments
2040	CD	CALL 2070H	;Call subroutine located at the memory
2041	70		; location 2070H
2042	20		
2043	NEXT	INSTRUCTION	

CALL Instruction



RETURN Instruction

CALL-Call unconditionally

Opcode	Operand	
CALL	16 bit address	Immediate/reg. indirect addressing , 5 m/c cycles, 18T states

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

CX-Call conditionally

Opcode	Operand	
CX	16 bit address	Immediate/Reg. indirect addressing , 2/5 m/c cycles, 9/18T states

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: CZ 2034 H

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 Not Zero	Z = 0
CPE	Call if Parity Even	P = 1
CPO	Call if Parity Odd	P = 0

RET-Return unconditionally

Opcode	Operand	
RET	none	Reg. indirect addressing , 3 m/c cycles, 10 T states

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

RX-Return conditionally

Opcode	Operand	
RX	none	Reg. indirect addressing , 1/3 m/c cycles, 6/12T states

The program sequence is transferred from the subroutine to the calling program based on the flag condition

The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address

Example: RZ

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 Not Zero	Z = 0
RPE	Return if Parity Even	P = 1
RPO	Return if Parity Odd	P = 0

RST-Restart instruction

Opcode	Operand	
RST	0-7	Reg. indirect addressing , 3 m/c cycles, 12 T states

- The RST instruction is a 1 byte specialized CALL instruction that jumps to one of the eight memory locations depending upon the operand
- These are used as software instructions in a program to transfer program execution to one of the eight locations

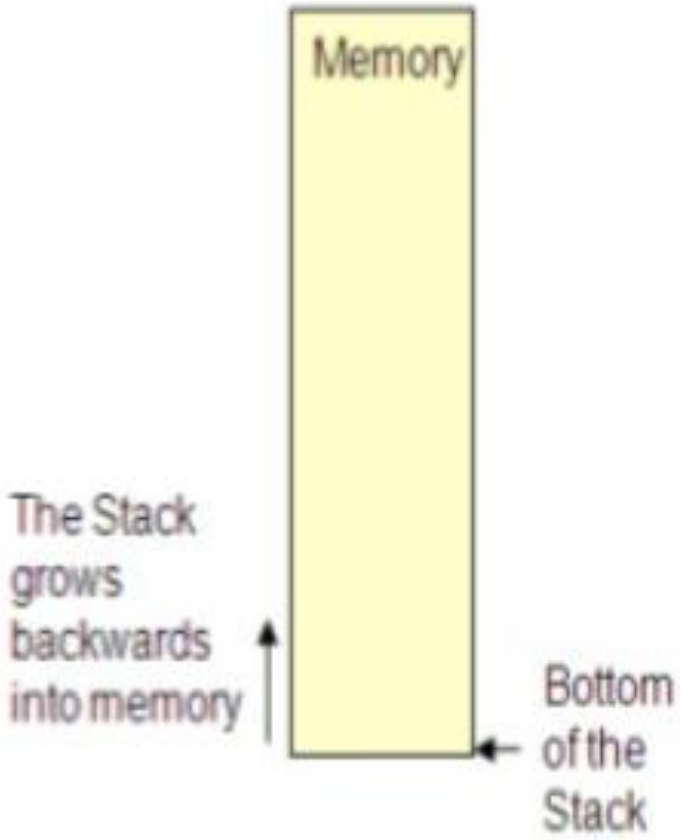
Example: RST 1 or RST 2

Instruction Code	Vector Address
RST 0	0*8=0000H
RST 1	1*8=0008H
RST 2	2*8=0010H
RST 3	3*8=0018H
RST 4	4*8=0020H
RST 5	5*8=0028H
RST 6	6*8=0030H
RST 7	7*8=0038H

Stack, I/O and machine control instructions

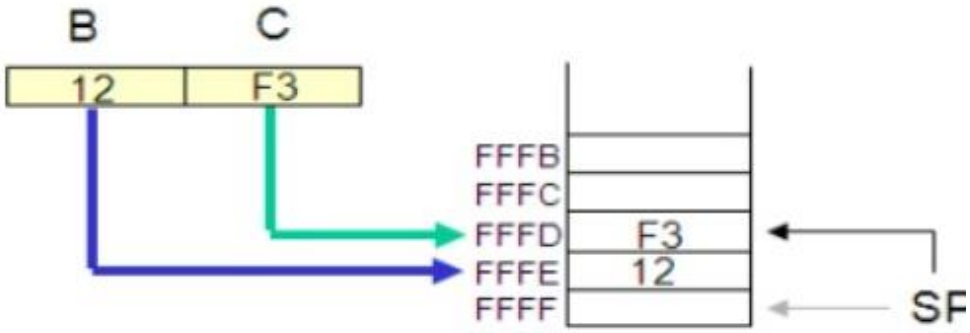
These instructions control the operation of microprocessor and include instructions related to stack operations

Stack

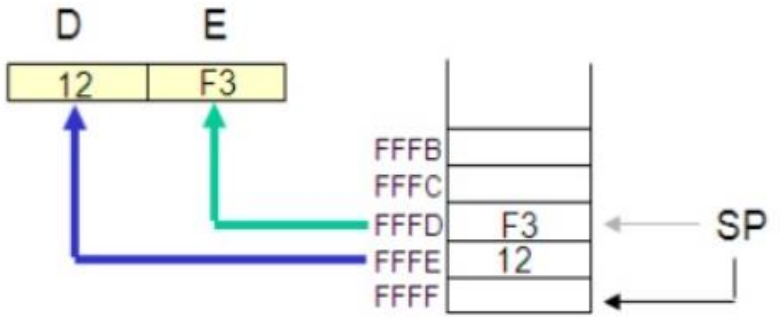

Memory
The Stack grows backwards into memory
Bottom of the Stack

PUSH and POP OPERATIONS

PUSH



POP



PUSH-Push the content of register pair/PSW to stack

Opcode	Operand	
PUSH	Reg. pair PSW	Register/Reg. indirect addressing , 3 m/c cycles, 12 T states

- PUSH Reg. pair-The content of register pair is pushed into stack
- PUSH PSW- Content of accumulator is pushed to stack. Contents of flag register is also pushed to the stack
- The content of stack pointer register is decremented by two to indicate new stack top

Example: PUSH D, PUSH PSW

POP-Copy two bytes from top of stack into register pair/PSW

Opcode	Operand	
POP	Reg. pair PSW	Register/Reg. indirect addressing , 3 m/c cycles, 10 T states

- POP Reg. pair-The two bytes from the top of the stack is moved to the register pair
- POP PSW- Byte from the top of the stack is moved to the flag register. Byte from the incremented stack location is moved to the accumulator.
- The content of stack pointer register is incremented by two to indicate new stack top

Example: POP D, POP PSW

IN-Input to accumulator from input port

Opcode	Operand	
IN	8 bit port address	Direct addressing , 3 m/c cycles, 10 T states

- The data available on the input port is moved to the accumulator
- The second byte of the instruction contains the address of the port which is an 8 bit address
- No flags are affected

Example: IN 01H

BEFORE EXECUTION

IN 80H

AFTER EXECUTION

OUT-Out from accumulator to output port

Opcode	Operand	
OUT	8 bit port address	Direct addressing , 3 m/c cycles, 10 T states

- The content of the accumulator is moved to the output port
- The second byte of the instruction contains the address of the port which is an 8 bit address
- No flags are affected

Example: OUT 00H

BEFORE EXECUTION

PORT 50H 0

A 40

OUT 50H

AFTER EXECUTION

PORT 50H 40

A 40

NOP-No operation

Opcode	Operand	
NOP	none	1 m/c cycle, 4 T states

- No operation is performed when this instruction is executed
- Registers and flags remain unaffected
- Program counter content is incremented by 1

Example: NOP

HLT-Halt

Opcode	Operand	
HLT	none	1 m/c cycle, 5 T states

- The CPU finishes executing the current instruction and halts any further program execution
- An interrupt or reset is necessary to exit from the halt state
- Registers and flags remain unaffected

Example: HLT

DI-Disable Interrupts

Opcode	Operand	
DI	none	1 m/c cycle, 4 T states

- When this instruction is executed, all the interrupts except TRAP are disabled
- The interrupt enable flip-flop is reset
- Flags remain unaffected

Example: DI

EI-Enable Interrupts

Opcode	Operand	
EI	none	1 m/c cycle, 4 T states

- When this instruction is executed, all the interrupts are enabled. The interrupt enable flip-flop is set
- This instruction is necessary to re-enable the interrupts (except TRAP)
- Flags remain unaffected

Example: EI

RIM-Read Interrupt Mask

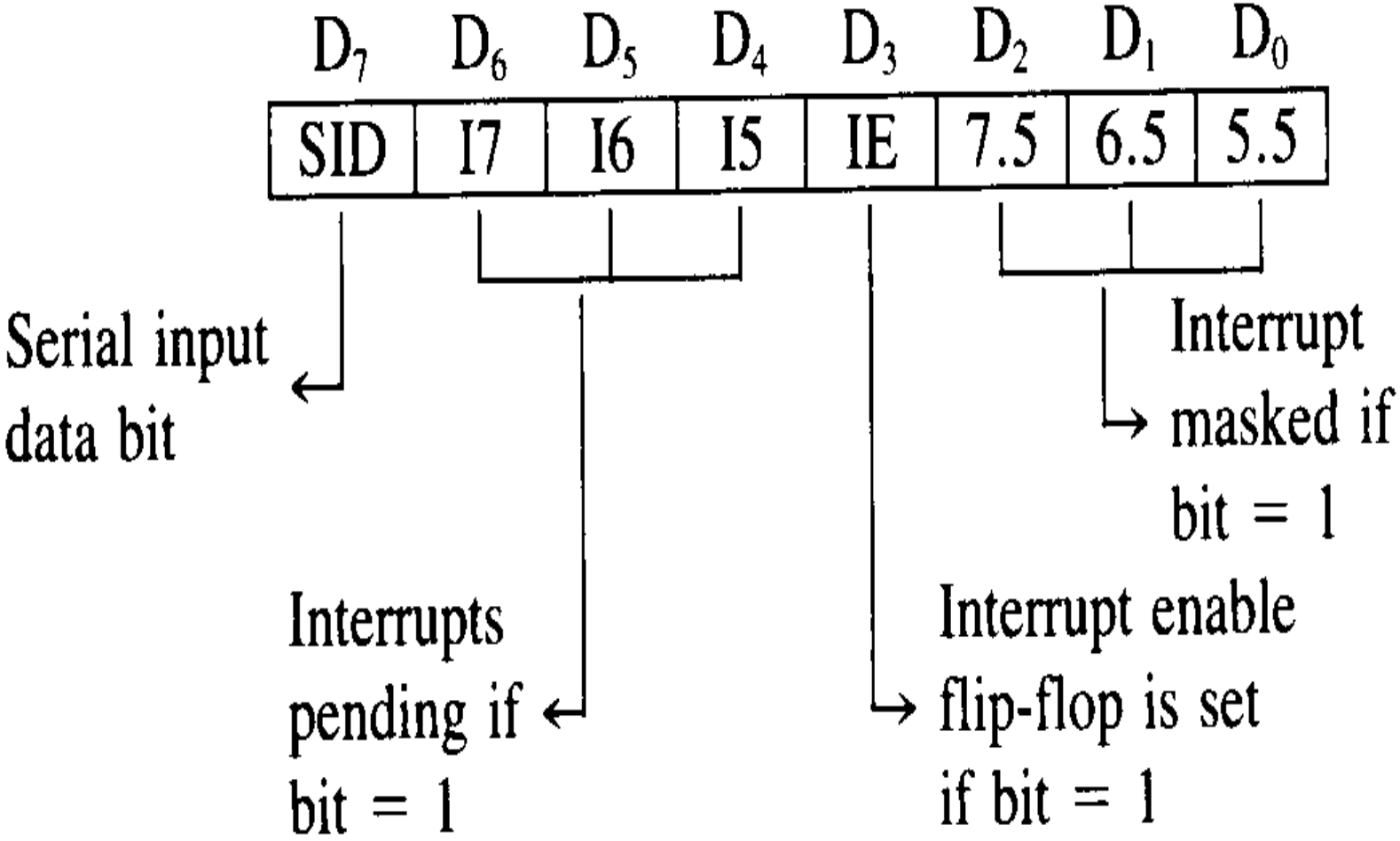
Opcode	Operand	
RIM	none	1 m/c cycle, 4 T states

- When this instruction is executed, the accumulator is loaded with pending interrupts, restart interrupt masks and the contents of SID

- Flags remain unaffected

Example: RIM

RIM-Read Interrupt Mask



SIM-Set Interrupt Masks

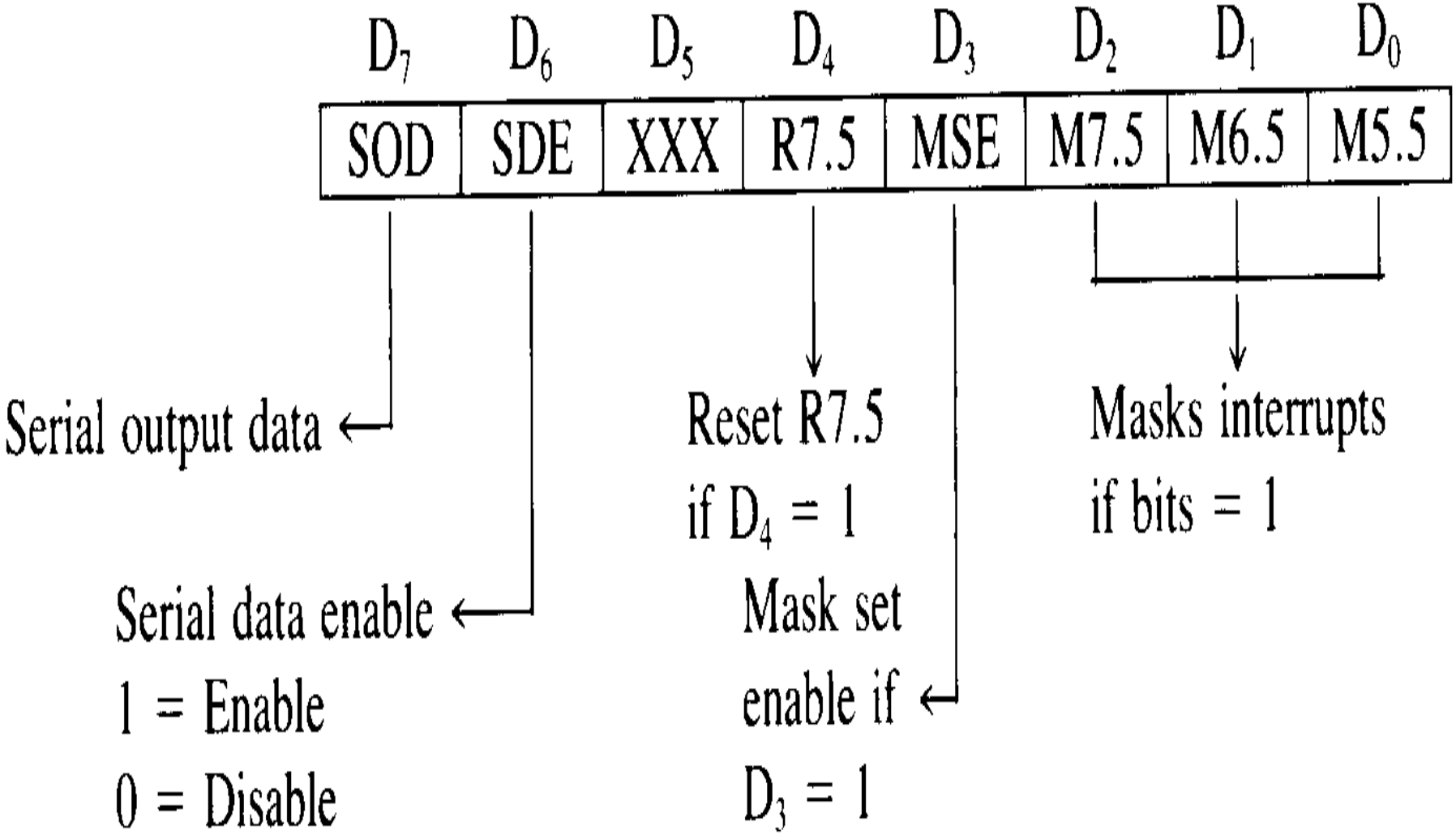
Opcode	Operand	
SIM	none	1 m/c cycle, 4 T states

- When this instruction is executed, bits 0-5 of the accumulator are used in programming the restart interrupt masks. Bits 6-7 of the accumulator are used in making serial output on SOD line

- Flags remain unaffected

Example: SIM

SIM-Set Interrupt Masks



DAA-Decimal Adjust Accumulator

Opcode	Operand	
DAA	none	Implicit addrsiing,1 m/c cycle, 4 T states

- After ADD instruction, the result is in binary. DAA instruction just placed after ADD instruction in the program operates on the result and gives the result in BCD. It uses AC and CY flags for decimal adjustment.6 is added to 4LSBs of the accumulator content if their value lies between A and F or AC flag is set to 1. 6 is added to 4MSBs of the accumulator content if their value lies between A and F or CY flag is set to 1.
- All Flags are affected.