SDM Lecture #07.

Date: 20/09/22

2 byte IN address (A) ← 8bit (address).
2 byte OUT address

IN 11011011 → opcode.
OUT 11010011

* Instruction Cycle:

Groups of instruction set:

① Data transfer Instruction: It consists of instructions that allow the data movement b/w the registers.

② Logical Instruction Groups: These instructions perform logical operations such as AND, OR, Ex-OR, Complement, ROTATE.

③ Branching Instruction Group:
These instruction causes a change in the sequence of execution of instructions. This includes conditional & unconditional jump instruction, subroutine call & return instructions.

④ Stack & Machine Control group:
These instruction relate with stack & internal control flags.

⑤ Arithmetic Instruction Group: ADD, SUB, Increment, Decrement of data in the register of the system can be performed by the instructions of this group

Date: 20/09/22

Branching Instruction Group:

- JMP
  - Conditional JMP
- CALL
  - Conditional Call
- RETURN
  - Conditional Return

★ JMP addres$_{16}$

* Memory address is 16-bit.
  - Program counter lower.

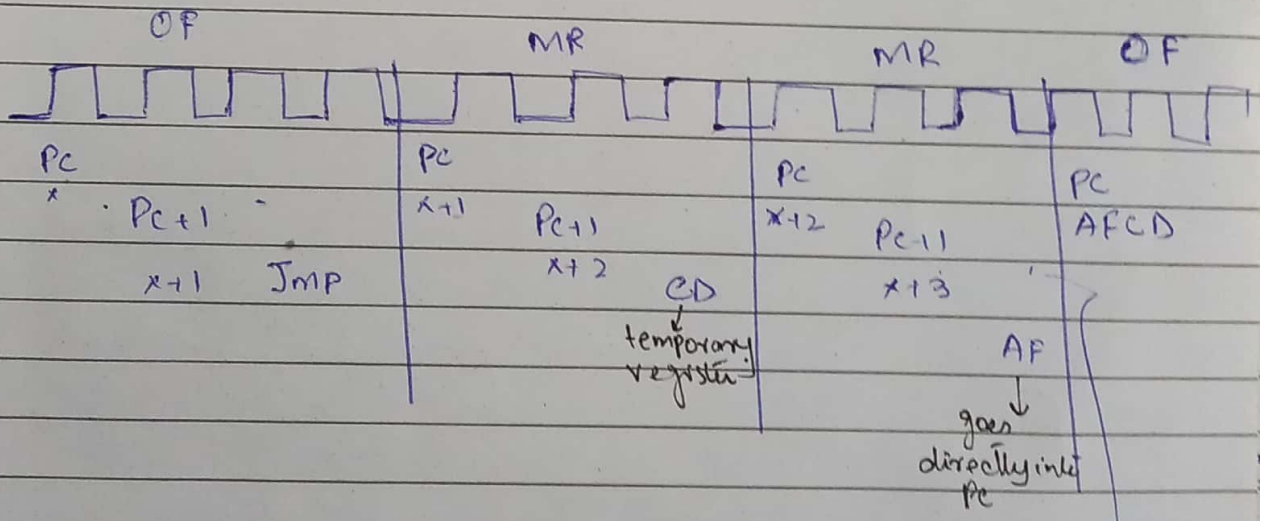
(PCL) ← (Byte 2)

(PCH) ← (Byte 3)

x	11000011	→ (Byte 1)
x+1	Lower byte of address (CD)	
x+2	Higher byte of address (AF)	
x+3		

MC = 3 (OF, MR, MR).
4 3 3 → T states

T states = 10



Date:

★ PCHL → 11101001

(PCL) ← (L)

(PCH) ← (H)

Machine cycle : 1 (OF)

T states : 6.

SDM LEC #08 Date: 27/09

8085A Consist of 74 instructions.

Conditional Jump Instruction

Jcondition address$_{16}$ → 1 byte

(PCL) ⟷ byte 2.

(PCH) ⟵ byte 3.

} total 3 bytes

00A0	11CCC010
00A1	CD
00A2	AB
00A3	

Flag Register

does not have a condition.

7	6	5	4	3	2	1	0
S	Z	·	AC	·	P	·	CY

S: Sign, Z: Zero, AC: Auxiliary, P: Parity, CY: Carry

★ Jcondition → binary's 11 CCC 010 (Machine Code)

CCC = 000/111 (binary only). $2^3 = 8$ combinations

8 Instructions (conditions):

1- JNZ address$_{16}$ (Jump if NOT ZERO Z=0).

11000010

2- JZ address$_{16}$. (Jump if zero Z=1).

11001010

3- JNC address$_{16}$, (Jump if Not Carry, CY=0)

11010010

4- JC address$_{16}$, (Jump if Carry, CY=1)

11011010

5- JPO address$_{16}$ (Jump if parity odd, P=0)

11100010

6- JPE address$_{16}$ (Jump if parity Even, P=1)

11101010

7- JP address$_{16}$ (Jump if positive S=0)

11110010

8- JM address$_{16}$ (Jump if minus S=1)

11111010

Date: 27/09/22

J Condition address$_{16}$ JMP

3 cycles.

OF { MR MR.

only if the condition is true. 3 machine cycle runs. 10-T states.

If condition is false

JNZ OF MR

Z=1 T_1: 00A1 T_2: 00A3 (PC+2)

2 machine cycles used if condition is false. and 7 t-states.

* CALL address$_{16}$

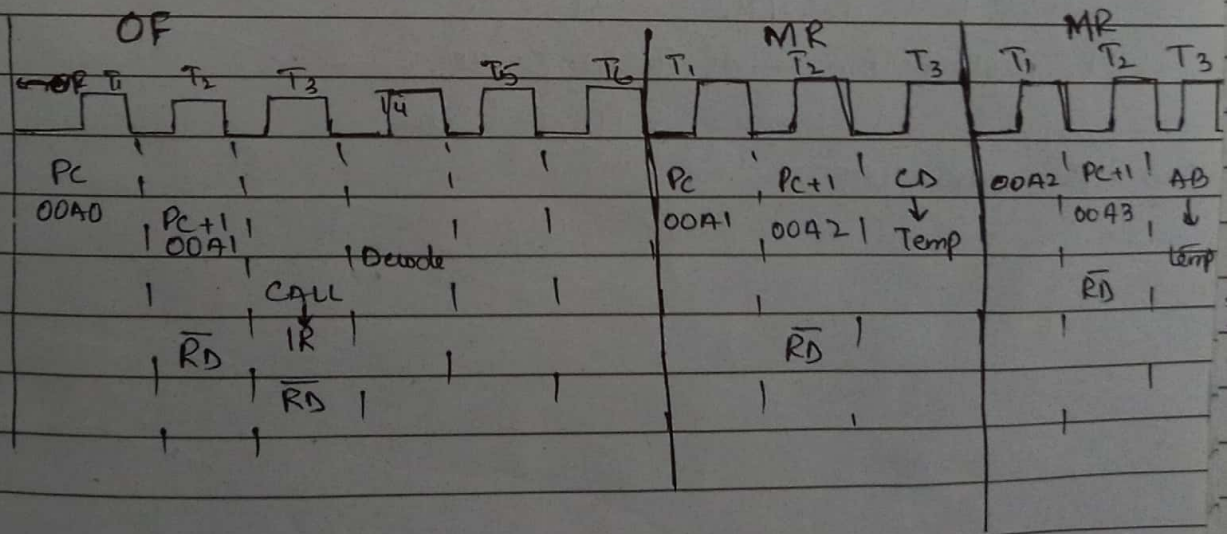
Subroutine, procedure, Method, Module, Function.

Stack in low level language is used to store PC, when call or anyother subroutine is called.

Machine Code = 11001101

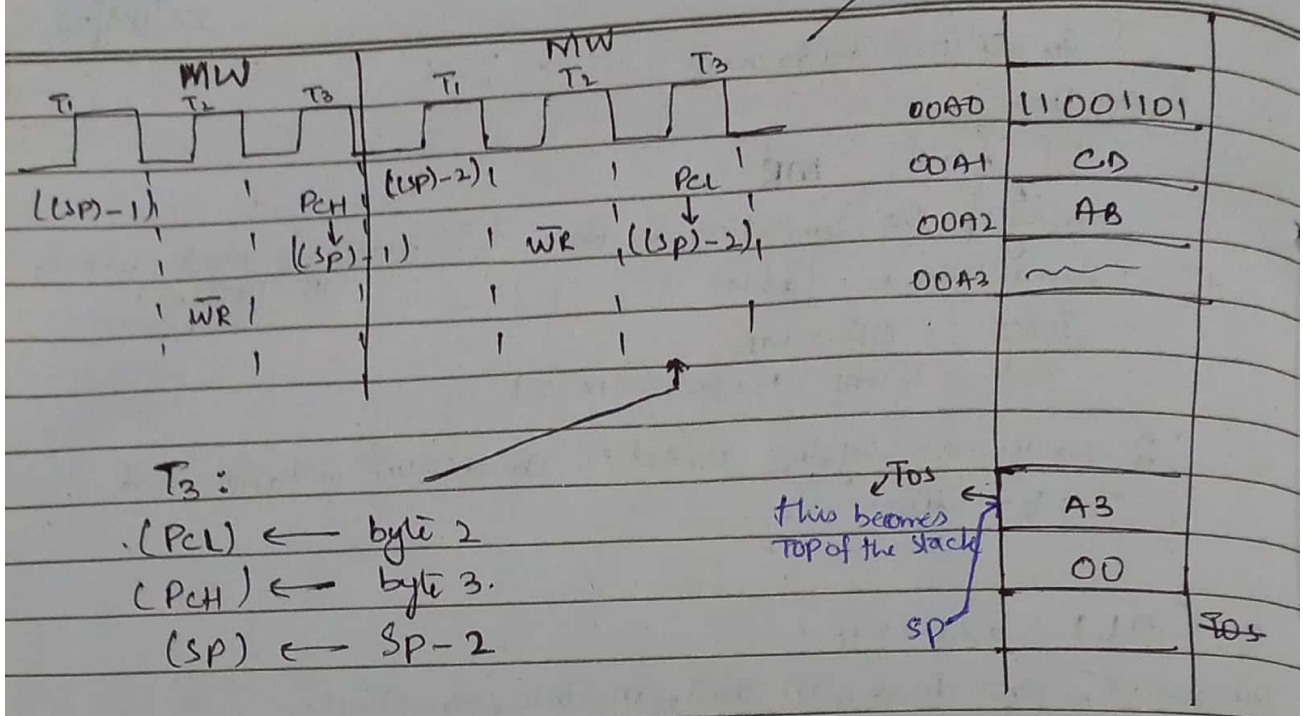
Stack pointer register → 16 bit, points to top of stack

Counter register → 16 bit



Subroutine will start.

Date:



CALL instruction → 5 machine cycles
18 T- states.

★ RET (return)

memory to Processor.

Machine code → 11001001

(PCL) ← ((sp))
(PCH) ← ((sp)+1)

SP ← (SP) + 2.

RET instruction → 3 m. cycles,
10 T- States

3 => OF MR MR

SDM - LEC #09

Date: 11/10/2022

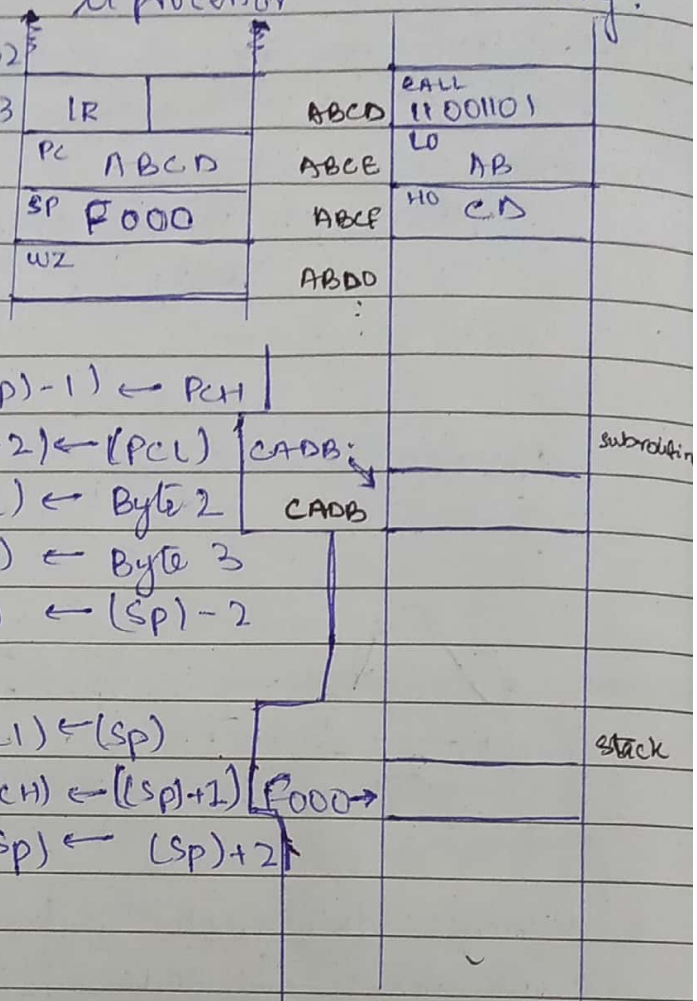
TRANSFER OF CONTROL / BRANCHING INSTRUCTION GROUP

11000011

* JMP address$_{16}$ (PCL) ← Byte 2
(PCH) ← Byte 3
11 CCC 010 Jcondition address

* Call address 5/18
11 CCC 100 Condition ((sp)-1) ← PCH
2/5 ((sp)-2) ← (PCL)
(PCL) ← Byte 2
(PCH) ← Byte 3
(Sp) ← (Sp) - 2

* RET 3/10
11 CCC 000 R Condition (PCL) ← (Sp)
T-3/5 F-> MC (PCH) ← ((sp)+1)
12/9 → T-states (Sp) ← (Sp)+2
I F



- MACRO → Pastes the codes, Program get lengthy
- JMP → need different addresses for Jump.
- CALL → Transfers the control of program.

CCC	
NZ	000
Z	001
NC	010
C	011
PO	100
PE	101
P	110
M	111

Date:

* Characteristics of 8085A Data transfer Instruction Group

- These instructions perform data transfer b/w Registers.
- One of the register can be in microprocessor, I/O or memory.
- Each data transfer instructions must Identify the source & destination register either explicitly or implicitly.
- Identification of one or both of the registers may be implied or explicit by the instruction ~~machine~~ mnemonic
- Data transfer involves copying the content of source register into the destination i.e the content of the source register remains unchanged.
- Internal registers involved are usually implied, whereas external register if involved are usually explicitly refered by their address that is part of instruction.

* Addressing Modes:

(a) Memory Reference Instruction: uses various addressing modes to ~~simplify~~ Identify registers involved in data transfer.

(1) Direct Addressing Mode:

Instruction contain the address of external registers involved with it.

- Direct addressing instruction are of 3 bytes size.

Load Accumulator (LDA Address) 00111010

(A) ⟵ ((byte 3) (byte 2)) 4, 13

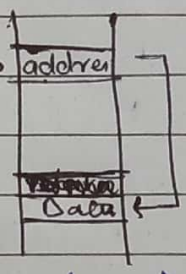
- Store Accumulator (STA address) 00 110 010

(byte 3) (byte 2) ⟵ (A) 4, 13.

Date: 11/10/22

- Load H & L Direct (LHLD Address)
  (L) ← ((byte 3)(byte 2)), (H) ← ((byte 3)(byte 2)+1)
  00 101 010
- Store H & L Direct (SHLD address)
  (byte 3)(byte) ← (L)
  ((byte 3)(byte 2)+1) ← (H) 001 000 10.

Effective Address (EA): Address. Address that points to data



② Register Indirect Addressing Mode:
- Address contained in instruction specifies a register (register pair) that contains a address (Effective Address). Instead of data or direct address itself in instruction.

- In Microcomputer Register Indirect (pointer or implied) addressing the register pair that contain the actual address of the operand in an internal register

- internal register used as pointer is implied by the instruction. (Thus a particular internal register is a pointer register for a particular instruction.).

- In many 8085 A register indirect addressing instruction the H&L register pair is used as pointer.

Mov from Memory. (Implicitly)

1 byte ← MOV r, M (r) ← ((H)(L))

01 DDD 110

∴ SSS = Source
∴ DDD = Destination

B	000
C	001
D	010
E	011
H	100
L	101

MOV A, M.

OF, MR

7-T states

MIGHTY PAPER PRODUCT

Date: dont use.

Mov to Memory

MOV M, r.

(H)(L) ← r. 01110 SSS

1	110
A	111

Load Accumulator Indirect

LDAX rp

(A) ← ((rp)) 00 RP 1010

RP	
00	BC
01	DE
10	HL
11	SP

Store Accumulator Indirect

STAX rp 00 RP 0010

(rp) ← (A)

SDM - LEC #10 Date: 18/10/22

Immediate Addressing Mode:

The data to be transfered is part of instruction. Second byte (or 2nd & 3rd Byte, in case of 16 bit data) of instruction is the data to be transfered.

* Move Immediate : MVI r, data (r) ← Byte 2
  00 DDD 110 2(OF, MR) / 7-Tstates

* Load Register Pair Immediate : LXI rp, data 16
  (rl) ← Byte 2
  (rh) ← Byte 3.
  00RP0001 3, (OF, MR, MR) / 10-T states.

* Move to Memory Immediate : MUI M, data 8
  ((H)(L)) ← Byte 2.
  00110110 , 3 (OF, MR, MW)

* Move Register:
  Mov r1, r2 (r1) ← (r2) 01DDDSSS.
  1 (OF) / 4-Tstates

* Exchange H & L with D & E : XCHG
  (L) ↔ (E)
  (H) ↔ (D)
  11101011

LOGICAL OPERATIONS:

- In all logic instructions one operand is in Accumulator (except those operating on carry flag).
- In operations requiring two operands, second operand may be contained in instruction, in general register, external register.

Date: 18/10/22

NOT : ① Complement Accumulator: CMA (A) ← $(\bar{A})$
00101111 , 1 (OF) / 4-Tstates, Flag: none.

② Complement Carry: CMC (CY) ← $(C\bar{Y})$ flag: CY
00111111

Set Carry:
(CY) ← 1 STC 00110111 flag: CY 1(OF)/4

AND:

① AND Immediate: ANI data
(A) ← (A)^ Byte 2 11100110
Cycle: 2 (OF, MR).

② AND Register: ANA r
(A) ← (A)^(r) 10100sss
cycle: 1 (OF)

③ AND memory: ANA M.
(A) ← (A)^((H)(L)). 10100110
Cycle : 2(OF, MR)/7-Tstates.

In the above Instructions: CY is cleared ↑ Z, S, P are set or cleared depending on the result left in Accumulator. AC is set

~~In the above instructions:~~
~~Both AC are cleared.~~

OR

① OR Immediate : ORI data
(A) ← A v Byte 2 11110110
Cycles 2 (OF, MR).

② OR Register : OR Ar
(A) ← (A) v (r) 10110sss

③ OR Memory: ORA M
(A) ← (A) v ((H)(L)) 10110110
Cycle : 2(OF, MR)/7.

CY & AC is ~~not~~ Cleared. Z, S, P are set or cleared

Date:

XOR : ① XOR Immediate
XRI data (A) ← A ⊻ Byte 2
Cycle: 2 (OF, MR)

② XOR Register
XRA r (A) ← (A) ⊻ (r) 10101SSS

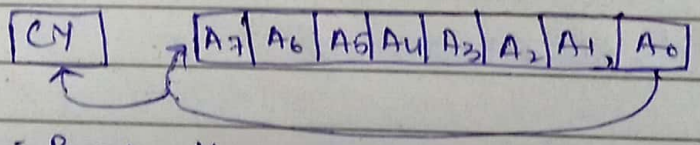
③ XOR Memory
XRA M (A) ← (A) ⊻ ((H)(L)) 10101110
Cycle 2 (OF, MR) T.

In the above instruction:
CY, AC are cleared
Z, S, P are set or cleared depending on the result left in Accumulator.

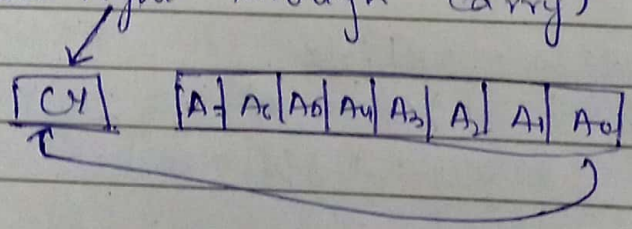
COMPARE MEMORY: Pg 9 pdf
CMP M (A) — ((H)(L)) 10111110
(A) ← A − ((H)(L))
Z = 1 if (A) = ((H)(L))
CY = 1 if A < ((H)(L))

S, Z on result in Accumulator is unchanged

RRC (Rotate Right) Pg 10



RAR (Rotate Right through Carry)



PROGRAM ASSEMBLY & TESTING

Date: 25/10/2022

Assembler:
1. Microprocessor Instruction
2. Assembler Directives / Pseudo Instruction
3. Comments

"." all instructions starting with dot are Assembler Directives.

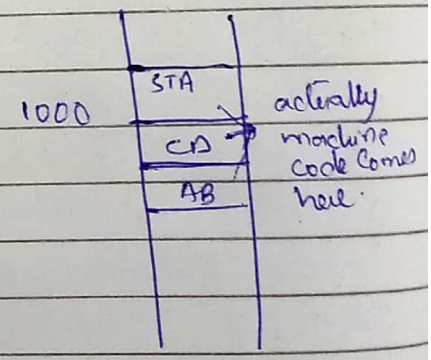
Fields in Assembly Language:
1) Label (optional) label_name:
2) Operation Code
3) Operand field
4) Comments (optional)

① label_name: opcode <spaces>, operand 1, operand 2, ...
operand n; Comments

Assembler Directives:

1) The Origin: ORG <expression>

```
ORG 1000
STA ABCD
```



2) END

3) SET & EQU

A SET 10 (value)

Symbol_name (label name has a colon, symbol name doesn't have a colon)

Date:

Symbol-name EQU ID

We cannot change value once it is assigned using Equale, value using SET can be change.

4) Define Storage : DS

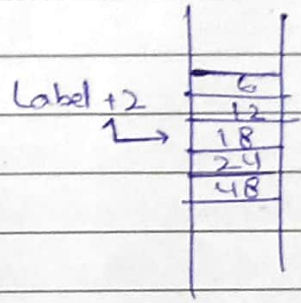
optional-label : DS

when a program is anembeled a symbol Table is made

Temp: DS 4

5) Define Byte: DB

Label: DB 6, 12, 18, 24, 48, ...



6) Define Word: DW

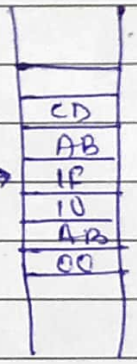
Label: DW ; list

Addren_1 : DW ABCD, 101F, 00AB

symbol table

Address_1 = memory Address (ABCD)

Address_1 + 1 (will point to 101F)



Conditional Anembler Directives:

Compile time:

IF expression

ENDIF

Date:

Local Pseudo Directives → used in MACRO

Local Label_name

used in MACRO as, everytime MACRO is called the directive is called.

```
Local MAT DS 10
      MAT 0001
      MAT 0002
      MAT 0003
      MAT 0004
```

using local gives a unique address in symbol table without local only the last address will be stored.

SDM. #12 Date: 1/11/2022

Two Pass Assembler:

LC (Location Counter)

1000

16 bits

Symbol/Label	Memory Address
START	1000H
L1	1003
Temp1	1004
Temp2	1008

Symbol Table

Opcode Table

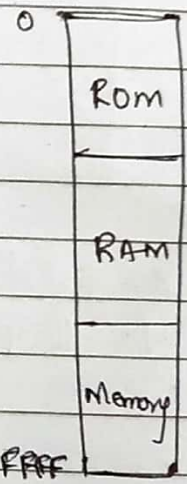
Mnemonics	Machine Code	Description Word
LDA		3.
Mov R_1, R_2		

Label Opcode Operand Comments

JMP addrew

CALL

JMP Symbol/Label



Location Counter is also reset to zero, but the problem is that zero is read only memory, so we need to give it a different address.

ORG → updates location counter.

ORG 1000H

START: LDA ABCD

as instruction is 3 bytes the LC will be updated by 3 bytes

1003

Location Counter (LC)

MIGHTY PAPER PRODUCT

Date: 1 | 11 | 22

```
ORG 1000H
START: LDA ABCD
L1:  MOV B,A
```

As MOV instruction is 1 Byte LC will increment by one

1004

LC

```
ORG 1000H
START: LDA ABCD
L1:  MOV B,A
Temp 1: DS 4
```

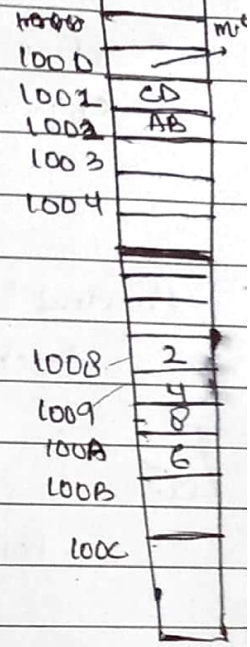
1008

LC

```
ORG 1000H
START : LDA ABCD
 L1 : MOV B,A
Temp 1 : DS 4
Temp 2 : DB 2,4,8,6
         STA 158B
         JMP START
         END
```

100C

LC

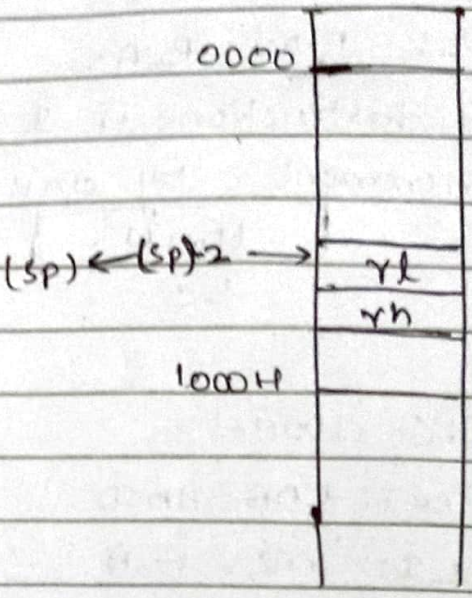


Date: 1/11

load register pair immediate

Stack & Subroutine:

```
LXI  SP, 1000H
SPHL
push PSW
pop  PSW
```



rl ← Sp
rh ← Sp + 1
Sp ← (Sp) + 2

Handout (The Stack & Subroutines)

Exchange H & L with D & E : XCHG

Q.

POP H → means H mai pop karwana.