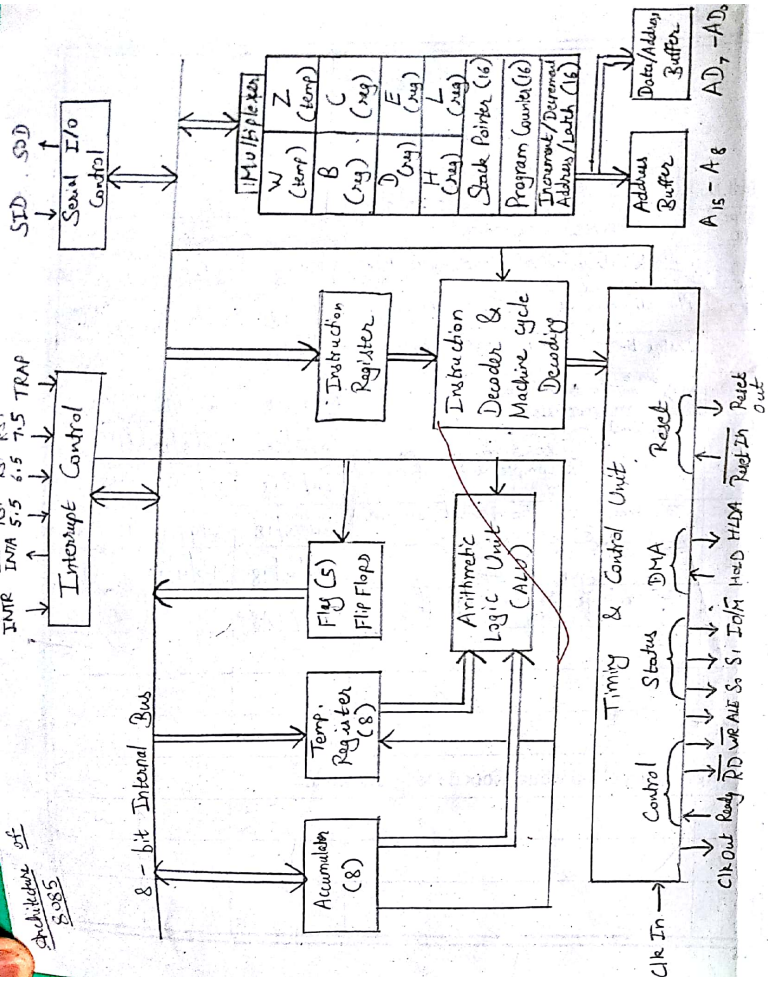


Architecture of 8085

Page No.
Date

Experiment 1

- **Aim**: To draw & explain -
  (i) Block diagram & pin diagram of 8085.
  (ii) Instruction set of 8085.

- **Theory**: Block - diagram of 8085.

Accumulator is an 8-bit register for performing arithmetic & logical operation. Also stores output of any function.

Temporary register is an 8-bit register to hold data on which accumulator is computing operation.

ALU performs arithmetic & logical operations.

Registers: There are 6 general purpose bit register which exist in pairs BC, DE, HL.

Flag Registers: 5 flip flops which change its status according to the results stored in accumulator. E.g. S, Z, AC, P, C.
They are connected to the ALU.

Page No.
Date

Instruction Register is an 8-bit register. When an instruction is fetched, then it's stored in the register.

Program Counter is 16-bit memory pointer which stores the memory address of the next instruction to be executed.

Stack Pointer is 16-bit register used as memory pointer which points to the memory location called stack.

Temporary Registers: It is an 8-bit register, which holds the temporary data of arithmetic & logical operations.

Instruction Decoder decodes the instructions present in the instruction register for further process.

Timing & Control Unit synchronises the registers & flow of data through various registers & other units. It has following timing and control system pins -

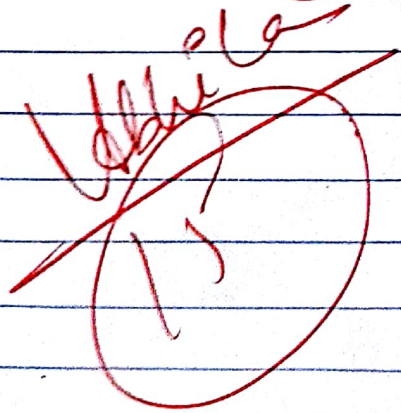
- Control Signals - Ready, $\overline{WR}$, $\overline{RD}$
- Status Signals - S_0, S_1, $IO/\overline{M}$
- DMA Signals : Hold, HLDA

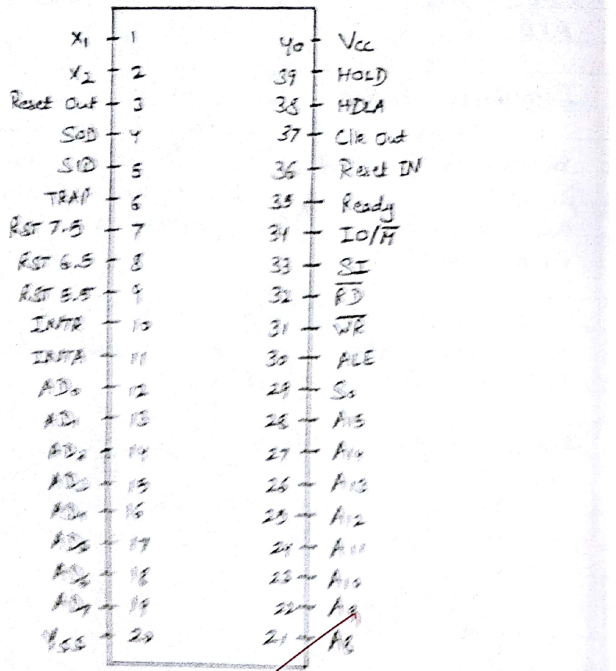
- Reset signals - ~~HOLD~~, Reset IN, Reset OUT
- ALE

Interrupt control : It controls an interrupt during a process. Whenever an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming process & after the completion of the process it goes back to main program.

Serial Input/Output Control : The input and output of serial data can be carried out using SID & SOD instructions.

Address Buffer & Address Data Buffer : The content stored in the stack pointer & program counter is loaded into the address buffer & address data buffer to communicate with the CPU.





PIN DIAGRAM OF 8085 MICROPROCESSOR

Page No.
Date

Experiment 2

- **Aim**: To study the pin diagram of 8085 microprocessor.
- **Theory**: 8085 is a general purpose microprocessor having 40 pins & works on a single power supply. To study the pin diag. we group the signals into following categories -
  (i) Power Supply
  (ii) Clock Signals
  (iii) Interrupt Signals
  (iv) Address & Data Bus
  (v) Control & Status Signal
  (vi) Serial I/O Port
  (vii) DMA request signals

- **Power Supply Signals**
  - V_{cc} : V_{cc} is used to be connected to +5V power supply.
  - V_{ss} : Ground Reference.

- **Clock Signals**
  - X_1 & X_2 : These pins are used for providing the clock frequency to the microprocessor.

Page No.
Date

Crystal Oscillation or LC oscillator is used to generate the frequency. 50 MHz frequency is applied.

(b). Clock Out : It's used as the system clock for devices connected with the microprocessor.

- Interrupt Signals : These pins are used for interrupt signals. These are 5-pins for hardware input -

(i) TRAP
(ii) RST 7.5
(iii) RST 6.5
(iv) RST 5.5
(v) INTR

INTA : It is used for acknowledgement. The microprocessor sends the acknowledgement to external devices through the INTA pin.

- Serial I/O Port

SID & SOD : These pins are used for ~~signal~~ serial data communication.

- Address Buses & Data Buses

(i) $AD_0 - AD_7$: These are multiplexed addresses & data

It can be used to carry 8-bit address as well as data. These lines are demultiplexed using latch.

(ii) $A_8 - A_{15}$: These are address buses used to address the memory location.

- **Control & Status Signal**

S_0 & S_1 : For the status signal in microprocessor.

- **ALE (Address Latch Enable)**

It controls the set of latches in 8085 microprocessor to access the external memory by multiplexing.

RD : Active low I/P used for reading operations.

WR : Active low I/P used for writing operations.

IO/M : Used to select memory or I/O.

Ready : It is used to set the READY pin. Or it enters the waiting stack otherwise.

Reset IN : It is used to reset the microprocessor by setting the program counter to Zero.

Reset OUT : It is used to reset all connected devices when the microprocessor is reset.

- **DMA Request Signal** : HOLD & HLDA : These signals are used for Direct memory Access (DMA). Controller recieves a request from the device & in turn issues the HOLD signal.

Page No.
Date

Experiment 3

- **Aim** : To study about the various instruction set of 8085 microprocessor.

- **Theory** : Instruction sets are classified in 5 sets -

(i) Control Transfer Instruction
(ii) Logical operation Instructions
(iii) Branching Instruction
(iv) Arithmetic Operation Instructions
(v) Data Transfer Instruction

- **Control Transfer Instruction**

NOP : No Operation
HLT : HALT & enter wait state
DI : Disable Interrupt
EI : Enable Interrupt
RIM : Read Interrupt Work
SIM : Set Interrupt Mask.

- **Logical Instruction**

CMP : Compare register with accumulator
CPI : Compare immediate with accumulator.
ANA : Logical AND of register / memory data with accumulator.
ANI : Logical AND of immediate data with accumulator.
XRA : Logical XOR of reg./memory data with accumulator.

XRI : Exclusive OR of immediate data with accumulator.
ORA : Logical OR of reg./memory data with accumulator
ORI : Logical OR of immediate data with accumulator
RLC : Rotate accumulator left.
RRC : Rotate accumulator right
RAL : Rotate accumulator left through carry.
RAR : Rotate accumulator right through carry.
CMA : Complement Carry.
STC : Set carry.

- Branching Instructions

JMP : Jump Unconditionally
RET : Return from subroutine unconditionally.
PCHL : Load program counter with H&L register content.
RST : Restart.

- Arithmetic Instructions

ADD : Add registers to accumulator
ADC : " " " " with carry
ADI : Add immediate data to accumulator.
ACI : " " " " " with carry.
LXI : Load register pair immediate.
DAD : Add register pair to H, L register.
SUB : Subtract register/memory from accumulator
SBB : Subtract source " " with borrow.
SUI : " immediate data from accumulator with borrow.

SBI : Subtract immediate data from acc. with borrow.
INR : Increment register/memory by one.
INX : Increment register pair by one
DCX : Decrement register pair by one.
DAA : Decimal adjust after addition.

- Data Transfer Instructions

MOV : Copy data from one source to destination.
LDA : Load accumulator direct from memory.
LDA X : Load accumulator from address in register pair
LXI : Load register pair with immediate data.
LHLD : Load H x L register directly from memory.
STA : Store accumulator direct in memory.
STAX : Store accumulator in address in register pair.
SHLD : Store H x L registers directly in memory.
XCHG : Exchange H x L with D x E.
SPHL : Move content of H x L to stack pointer.
XTHL : Exchange top of stack with H & L.
PUSH : Push 2 bytes of data from the stack.
POP : Pop 2 bytes of data from the stack.
OUT : Output data from accumulator to port with 8-bit address.
IN : Input data to accumulator from a port with 8-bit address.

Output

Register		Flag	
A	32	S	0
BC	00 00	Z	0
DE	00 00	AC	1
HL	00 0D	P	0
PSW	00 00	C	0
PC	42 0A		
SP	FF FF		
Int-Rq	00		

Memory

Address	Data
000Ch	10
000Dh	20
000Eh	30

Page No.
Date

Experiment 4

- Aim : To add two 8-bit numbers in 8085 micro processor.
- Software Used : GNU.
- Source Code

```
LXI H, 000Ch
MOV A, M
INX H
ADD M
STA 000Eh
HLT
```

14

Output

Register		Flag		Memory	
		S	0	Address	Data
A	04	Z	0	0000h	0
BC	00 01	AC	0	000Bh	4
DE	00 00	P	0	000Eh	100
HL	00 0F	C	1	000Fh	160
PSW	00 00				
PC	42 15				
SP	FF FF				
Int-Reg					

Experiment 4 (b)

- **Aim**: To add two 8-bit numbers in 8085 microprocessor (with carry).
- **Software Used**: GNUSim 8085.
- **Source Code**

```
MVI C, 0000h
LXI H, 00Eh
MOV A, M
INX H
ADD M
JC XYZ
STA 00Ah
MOV A, C
HLT

XYZ : INR C
      STA 00Bh
      HLT
```

- **Result**

Addition of two 8-bit numbers has been performed successfully in 8085 microprocessor (with carry).

Output

Registers		Flag	
A	ΣE	S	0
BC	00 F6	Z	0
DE	00 00	Ac	0
HL	00 00	P	1
PSW	00 00	C	1
PC	42 11		
SP	FF FF		
INT-Rq			

Memory	
Address	Data
000Ah	100
000Bh	246
000Ch	40
000Dh	30

Experiment 5

- **Aim**: To subtract two 8-bit numbers in 8085.
- **Software Used**

GNU.Sim8085

- **Source Code**

```
LDA 000Ah
CMA
INR A
STA 000Bh
MOV C, A
LDA 000Ch
ADD C
STA 000Dh
HLT
```

- **Result**

Subtraction of two 8-bit numbers in 8085 microprocessor has been performed successfully.

Output

Registers		Flag	
A	F0	S	0
BC	00 00	Z	0
DE	00 00	AC	0
HL	00 00	P	0
PSW	00 00	C	0
PC	42 08		
SP	FF FF		
Int-Rg	00		

Memory	
Address	Data
000Ah	15
000Bh	240

Experiment 6

- Aim : To find 1's complement of an 8-bit numb
- Software Used : GNUSim8085
- Source Code

```
LDA 00Ah
CMA
STA 00Bh
HLT
```

- Result

1's complement of an 8-bit number has been found successfully.

Output

Registers		Flag	
A	EC	S	1
BC	00 00	Z	0
DE	00 00	AC	0
HL	00 00	P	0
PSW	42 09	C	0
SP	FF FF		
Int Reg	00		

Memory	
Address	Data
000Ah	20
000Bh	256

Experiment 7

- **Aim** : To find 2's complement of an 8-bit no.
- **Software Used**
  GNUsim8085
- **Source Code**

```
LDA 000Ah
CMA
INR A
STA 000Bh
HLT
```

- **Result**

2's complement of an 8-bit no. has been found successfully.

Output

(i) Left Shift

Registers		Flag	
A	84 3C	S	0
BC	00 00	Z	0
DE	00 00	AC	1
HL	00 00	P	1
PSW	00 00	C	0
PC	42 08		
SP	FF FF		
Int-Ry	00		

Memory	
Address	Data
000B	30
000D	60

(ii) Right Shift

Registers		Flag	
A	8F		
BC	00 00	S	0
DE	00 00	Z	0
HPL	00 00	AC	1
PSW	00 00	P	1
PC	42 08	C	1
SP	FF FF		
Int-Ry	00		

Memory	
Address	Data
000B	31
000D	143

Page No.
Date

Experiment - 8

- Aim : To shift an 8-bit no. right & left
- S/w Used : GNU Sim 8085

Source Code

(a) Shift left

```
LDA 000BH
ADD A
STA 000DH
HLT
```

(b). Shift Right

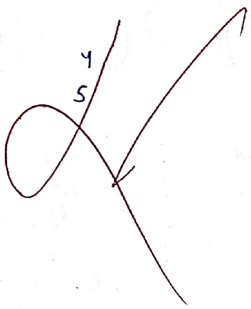
```
LDA 000BH
RRC
STA 000DH
HLT
```

Output

0030 3
0031 4

After sum

0030 4
0031 5



Page No.
Date

Experiment 9

- Aim: Write a program to exchange the contents of memory location.
- Software Used: 8085 microprocessor
- Procedure

1. Get the contents of memory location 001H into accumulator.
2. Save the contents into B register.
3. Get the contents of memory location 0002H into acc.
4. Store the contents of accumulator at 0001H.
5. Get the saved contents from B back to acc.
6. Store the contents of acc. to address 0002H.
7. HLT.

- Program

```
LDA 0001H ;
MOV B, A ;
LDA 0002H ;
STA 0001H ;
MOV A, B ;
STA 0002H ;
HLT ;
```

14