Kunthavai Naacchiyaar

Government Arts College for Women (Autonomous)

Thanjavur - 613 007, Tamilnadu, India

Accredited by NAAC with B Grade with a CGPA of 2.22 out of 4.00 in the 4th cycle

Microprocessor Architecture

Subject code:18K5CS07

Faculty: 1. Mr. V. Chezhiyan

2. Mrs. K.Sharmila.

UNIT I

Introduction: Word length of Computer of Microprocessor- Evolution of Microprocessors- Evolution of Digital Computers- Computer Generations- Single chip Micro Computers - Embedded Microprocessors- Hardware, Software and Firmware – CPU- Memory: Semiconductor Memory, Buses: Processing speed of a Processor.

UNIT II

Microprocessor Architecture: Introduction- Intel 8085- Instruction Cycle-Timing Diagram- Instruction Set of Intel 8085: Introduction- Instruction and Data Formats- Addressing Modes- Status Flags- Symbols and Abbreviations - Intel 8085 Instructions.

- A microprocessor is a computer processor that incorporates the functions of a central processing unit on a single integrated circuit of MOSFET construction
- Microprocessor programming languages can typically be divided into three main types: machine language, assembly language, and high-level language.
- A *machine language program* consists of either binary or hexadecimal op-codes. Programming a microcomputer with either one is relatively difficult, because one must deal only with numbers.
- The architecture and micro programs of a microprocessor determine all its instructions. These instructions are called the microprocessor's *instruction set*.
- Programs in *assembly* and *high-level languages* are represented by instructions that use English-language-type statements.

- In computing, a **word** is the natural unit of data used by a particular processor design. A word is a fixed-sized piece of data handled as a unit by the instruction set or the hardware of the processor. The number of bits in a word (the *word size*, *word width*, or *word length*) is an important characteristic of any specific processor design or computer architecture.

First Generation Microprocessors

- The first generation microprocessors were introduced in the year 1971-1972. The instructions of these microprocessors were processed serially, they fetched the instruction, decoded and then executed it.

Second Generation Microprocessors

- In the year 1970, a small number of transistors were available on the integrated circuit in the second-generation microprocessors.

Third Generation Microprocessors

- The third generation microprocessors were introduced in the year 1978, as denoted by Intel’s 8086 and the Zilog Z8000. These were 16-bit processors with a performance like mini computers.

Fourth Generation Microprocessors

- As many industries converted from commercial microprocessors to in house designs, the fourth generation microprocessors are entered with outstanding design with a million transistors.

Fifth Generation Microprocessors

- Fifth-generation microprocessors employed decoupled superscalar processing, and their design soon exceeded 10 million transistors. In the fifth generation, PCs are a low-margin, high volume business conquered by a single microprocessor.

- Single chip microprocessor
- Featured snippet from the web
- Putting a CPU, ROM, RAM, and data input/output circuitry into a single IC will make a single-chip microprocessor. ... Single-chip microprocessors are also known as "microcomputer units (MCUs)," because they are made of a single IC.
- Everything is packaged in the same single physical IC
- The IC contains the CPU core(s)
- The IC contains the memory (ROM and RAM)
- The IC contains all the IO hardware (Video, serial, etc)
- Single-chip computers are mainly of the form known as *Microcontroller* chips (the most commonly known are the PIC range by Microchip inc) and used in embedded devices. They provide much more basic functionality but are far simpler to work with as they don't require any external chips in order to function.

Embedded Microprocessor

- Embedded microprocessors are computer chips used inside devices other than computers to provide added functionality, often in the areas of control and monitoring.

- An embedded microprocessor is a computer chip used inside several devices and equipments to provide added functionality.

- A microprocessor is a digital-electronic component with transistors integrated on a single semiconductor IC that is small and consumes less power. Due to flexibility, cost, programmability and adaptability microcontrollers are popular to implement various types of controllers that we know from the electronics history.

- The functions of the microprocessor include fetching, decoding and processing the data.

Hardware

The physical components of a computer system such as mouse, keyboard, monitor etc. are called as Hardware. Moreover , we can able to touch and feel the different components.

Software

Software, is a collection of programs or applications, which contain the instructions that makes the computer work. For instance, when you type words via the keyboard; the software is responsible for displaying the correct letter in the correct place on the screen. Software is held either on your computer's hard disk. CD-ROM, DVD or on a diskette (floppy disk) and is loaded (i.e. copied) from the disk into the computer RAM (Random Access Memory) when required.

Firmware

The main memory of a computer consists of two parts. The Random Access Memory (RAM) and Read Only Memory (ROM). The RAM is used to load the software when you need to run them, while the ROM contains some programs written by the computer manufacturer, which can only be read but not written. These programs are called firmware and they are used to start up the machine (also called booting) with the necessary checks and then by loading the operating system to the RAM. Once the Operating System is loaded the control of the system is passed onto it.

CPU –MEMORY

- The hardware that defines a computer is the **CPU** and **memory** . ... The **CPU** and **memory** work together to run programs. **CPU** - executes programs using the fetch-decode-execute cycle. **Memory -** stores program operations and data while a program is being executed .

- The **CPU** interacts closely with primary storage, or main **memory**, referring to it for both instructions and data. ... Technically, however, **memory** is not part of the **CPU**. Recall that a computer's **memory** holds data only temporarily, at the time the computer is executing a program.

- **Main memory** is intimately connected to the **processor**, so moving instructions and data into and out of the **processor** is very fast. **Main memory** is sometimes called RAM. RAM stands for Random Access **Memory**.

Buses

- Three types of bus are used.
- Address **bus** - carries memory addresses from the processor to other components such as primary storage and input/output devices. ...
- Data **bus** - carries the data between the processor and other components. ...
- Control **bus** - carries control signals from the processor to other components.

BUS structure : A group of lines that serves as a connecting path for several devices is called **bus**. In addition to the lines that carry the data, the **bus** must have lines for address and control purposes.

Processing speed of processor

Clock speed is measured in units of cycles per second, which is called a Hertz (Hz). Computer boards and CPUs run at rates of millions and billions of Hertz, megahertz (MHz) and gigahertz (GHz). A good **speed** for a PC **microprocessor** in 2004 was 4 GHz

MICROPROCESSOR ARCHITECTURE

The microprocessor is the central processing unit or CPU of a micro computer. it is the heart of the computer.

INTEL 8085:

Intel 8085 is an 8 bit NMOS Microprocessor. It is an 40 pin IC. The Intel 8085 uses a single +5 volt supply. Its clock speed is 3 MHZ. The clock cycle is of 320ns. It has 80 basic instruction and 246 Opcodes. It consists of 3 main sections.

1. Arithmetic Logic Unit(ALU) 2.Timing and control unit.3. set of Registers.

Arithmetic Logic Unit:

It performs various arithmetic and logical operations like addition, subtraction, Logical AND,OR,XOR, complement, increment, Decrement, Left shift, Rotate left, Rotate Right and clear etc.,

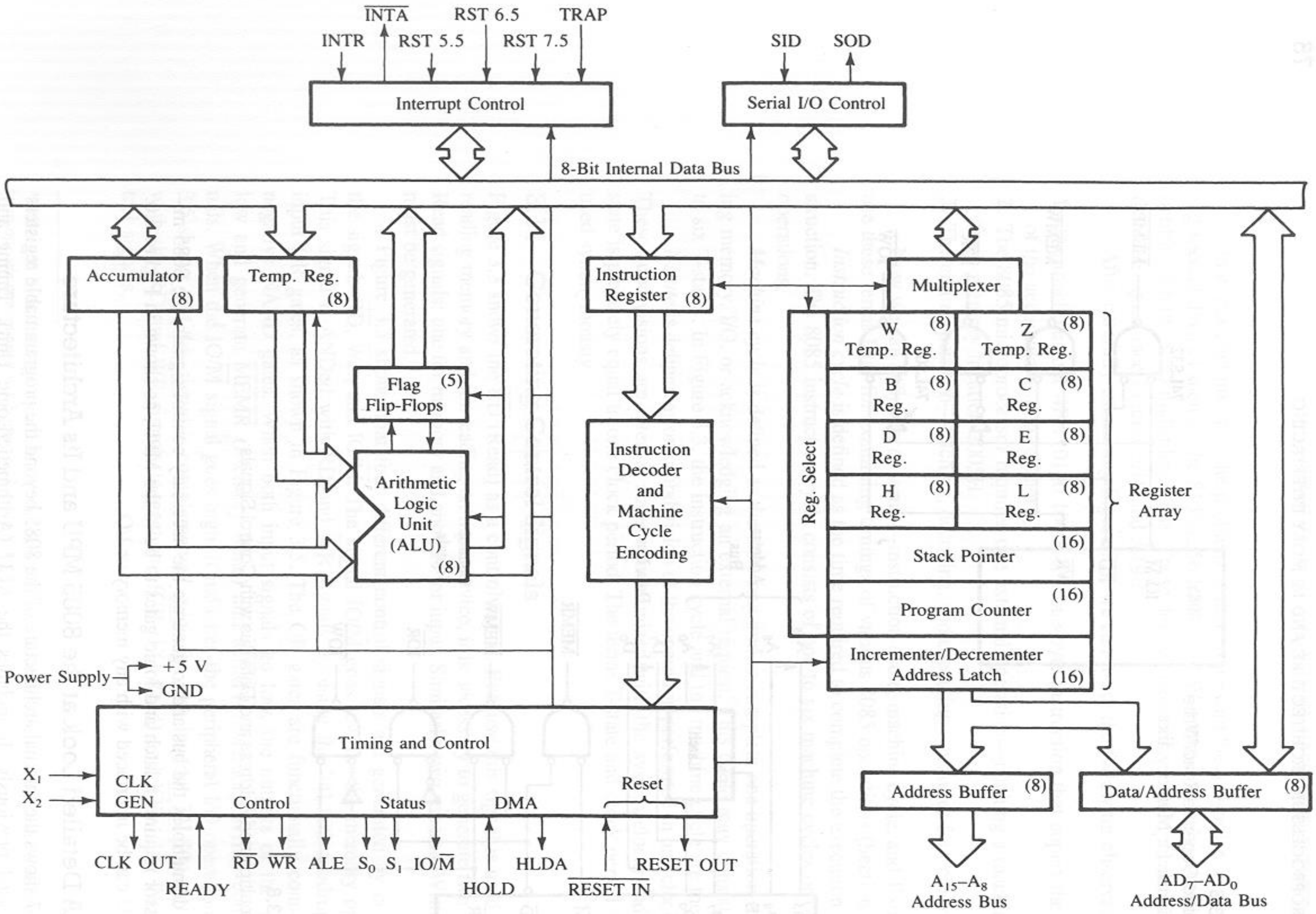
Timing and control unit:

It generates timing and control signals which are necessary for the execution of the instructions. It controls data flow between CPU and peripherals.

Registers:

Registers are used by the microprocessor for the temporary storage and manipulation of data and instructions.8085 has the following registers:

(i) 8 bit accumulator i.e. register A
(ii) Six 8-bit general purpose registers i.e.. B,C,D,E,H,L
(iii) one 16 bit register i.e. stack pointer, SP
(iv) one 16-bit Program counter, PC
(v) Instruction Register
(vi)Temporary Register


INTA
RST 6.5
TRAP
INTR
RST 5.5
RST 7.5
SID
SOD
Interrupt Control
Serial I/O Control
8-Bit Internal Data Bus
Accumulator (8)
Temp. Reg. (8)
Flag Flip-Flops (5)
Arithmetic Logic Unit (ALU) (8)
Instruction Register (8)
Instruction Decoder and Machine Cycle Encoding
Multiplexer
Reg. Select
W Temp. Reg. (8)
Z Temp. Reg. (8)
B Reg. (8)
C Reg. (8)
D Reg. (8)
E Reg. (8)
H Reg. (8)
L Reg. (8)
Stack Pointer (16)
Program Counter (16)
Incrementer/Decrementer Address Latch (16)
Register Array
Power Supply
+5 V
GND
Timing and Control
X1
X2
CLK GEN
Control
Status
DMA
Reset
CLK OUT
READY
RD
WR
ALE
S0
S1
IO/M
HOLD
HLDA
RESET IN
RESET OUT
Address Buffer (8)
Data/Address Buffer (8)
A15–A8 Address Bus
AD7–AD0 Address/Data Bus

Accumulator: Its an 8-bit register associated with ALU. The Register A in 8085 is accumulator. It is used to hold one of the operands of an arithmetic or logic operation. it serves as one input to ALU.

General purpose Registers: 8085 Microprocessor contains six 8-bit general purpose registers. They are: B, C, D, E, H and L Register. The combination of two 8-bit register is known as Register pair. The valid Register pairs are B-C, D-E, and H-L.
Program counter: It is an 16-bit general purpose Register. It is used to hold the memory address of the next instruction to be executed.

Stack Pointer: it is a 16-bit special function register. The stack pointer controls addressing of the stack. The SP holds the address of the top element of data stored in the stack.

Instruction Register: It holds the opcode (operation code) of the instruction which is being decoded and executed.

Temporary register: It is an 8-bit register associated with ALU. It holds the data during arithmetic/logical operation. it is not accessible to programmer.

FLAGS:

It is a set of 5 flip-flops to serve as status flags.

Carry Flag(Cs)

parity Flag(P)

Auxiliary carry Flag(AC)

Zero Flag(Z)

Sign Flag(S)

Carry Flag:

It holds carry out of the Most Significant Bit resulting from the execution of an arithmetic operation.

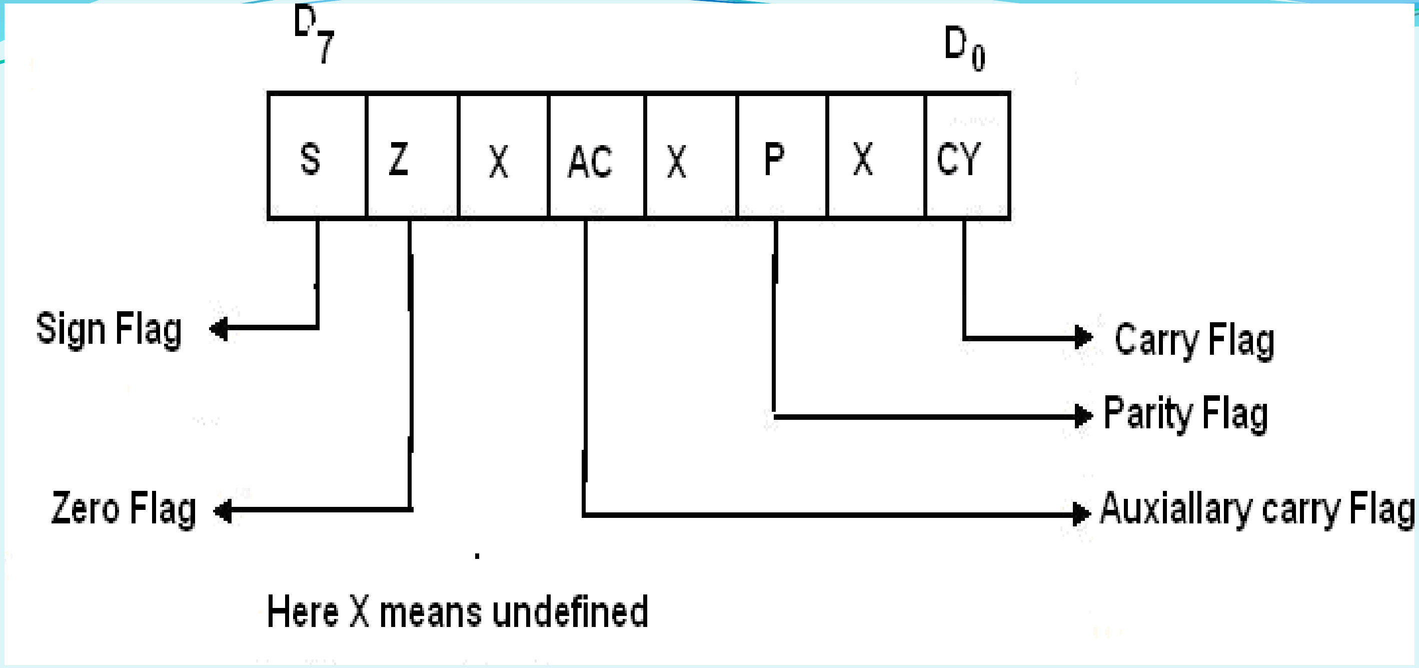
If there is a carry from addition or a borrow from subtraction or comparision, t he carry flag is said to 1 otherwise it is 0.

P**arity Flag:** It is set to 1 when the result of the operation contains even no. of 1& it is set to 0 if there are odd no. of 1.

Auxiliary Carry Flag: It holds carry from bit 3 to the number 4 resulting from the execution of an arithmetic operation. If there is a carry from bit 3 to 4,the AC flag is set to 1 otherwise it is 0.

Zero Flag:
It is said to 1, if the result of an arithmetic or logical operation is zero. . If the result is not zero, the flag is set to 0.

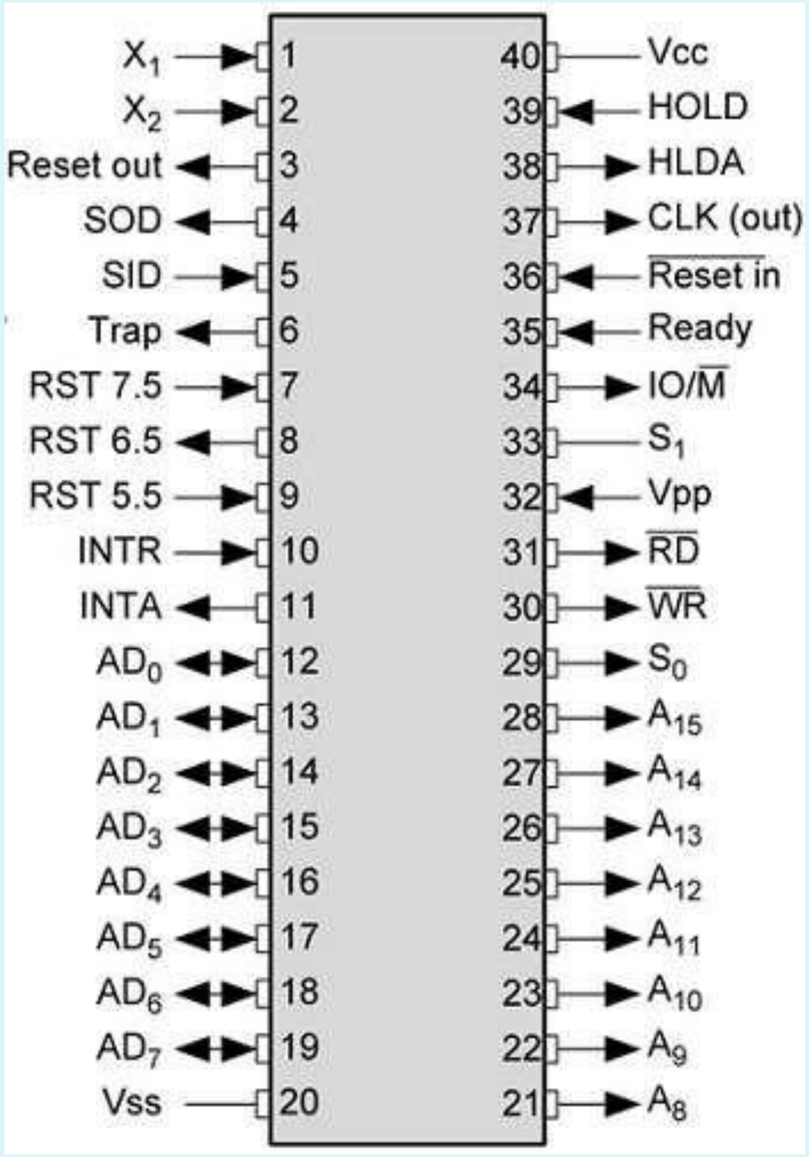
Sign Flag: It is set to 1, if the result of an arithmetic or logical operation is negative` if the result is positive the sign flag is set to 0`.



Program Status Word(PSW): It is a combination of 8-bits is called Program Status Word(PSW). PSW and the accumulator are treated as a 16 bit unit for stack operation.

Data and Address Bus: Data bus is 8 bit wide . 8 bit of data can be transmitted in parallel form or to the microprocessor.
Address bus is 16 bit wide as memory address are of 16 bit. 8 MSB of the address are transmitted by A8-A15. 8 LSB is the address are transmitted by the data bus AD0-AD7.

Pin Configuration



PIN CONFIGURATION OF 8085

A_8-A_{15} (output): These are address bus and are used for the most significant bits of the memory address or 8 bits of I/O address.

AD_0-AD_7 (input/output): these are time multiplexed address /data bus that is they serve dual purpose .they are used for the least significant 8 bits of the memory address or I/O address during the first clock cycle of a machine cycle. Again they are used for data during second and third clock cycles.

ALE (output): It is an address latch enable signal. It goes high during first clock cycle of a machine cycle and enables the lower 8 bits of the address to be latched either into the memory or external latch.

IO/M(output): it is a status signal which distinguishes whether the address is for memory or I/O. when it goes high, the address on the address bus is for an I/O device. When it goes low, the address on the address bus is for a memory location.

S_0, S_1 (output): These are status signals sent by the microprocessor to distinguish the various types of operation.

RD (output): It is a signal to control READ operation .

WR(output): it is a signal to control WRITE operation .when it goes low the data on the data bus, is written into the selected memory or sent to the I/O device.

READY(input): When READY   is high, it indicates that the input or output device is ready to send or receive data. When READY is low, the microprocessor waits till READY becomes high.

HOLD (input): It indicates that another device is requesting for the use of the address and data bus. After receiving the HOLD request, the microprocessor sends out a HLDA signal to the device. Then the microprocessor leaves the control over the buses as soon as the current machine cycle is completed. . Internal Processing may continue.

INTR (input): It is an interrupt signal sent by an external device to the microprocessor. When it goes high, the microprocessor suspends the execution of its normal sequence of instruction.

TRAP: TRAP has the highest priority. It is used in emergency situation. it is an non-mask able interrupt.

TRAP

RST 7.5

RST 6.5

RST 5.5

INTR

When an interrupt is recognize the next instruction is executed from a fixed location in memory. A subroutine is executed which is called ISS(interrupt service subroutine).

INTERRUPTS	ADDRESS
TRAP	0024
RST 5.5	002C
RST 6.5	0034
RST 7.5	003C

HLDA (output): It is an HOLD acknowledgement signal send out by the microprocessor after receiving the HOLD signal. It is send to the device which has issued the HOLD SIGNAL.

RESET IN (input): It resets the program counter to zero .it also resets interrupts enable that is an HLDA flip-flops.

RESETOUT (output): It indicates that the CPU is being reset.

X1, X2 (input): These are terminals to be connected to an external crystal oscillator which drives an internal circuitry of the microprocessor to produce a suitable clock for the operation of microprocessor.

CLK (output): It is a clock output for user, which can be used for other digital integrated circuits

SID (input): It is data line for serial input. The data on this line is loaded into the 7th bit of the accumulator when rim (read interrupt mask) instruction is executed.

SOD (output): It is data line for serial output. The 7th bit of the accumulator is output on sod line when sum instruction is executed.

Vcc: It is +5 volt dc supply.

Vss: It is the ground reference.

Opcode and operands:
Each Instruction contain two parts: Operation code(opcode) and Operands. The first part of the instruction which specifies the task to be performed by the computer is called opcode. The second part of the instruction is the data to be operated on, and it is called operand..

Instruction word Size: the digital computer understands the instruction written in binary codes.(machine codes). The machine codes of all instructions are not all same length. According to the word size instructions are classified into three types.

1. 1- byte instruction
2. 2-byte instruction
3. 3-byte instruction

1-Byte instruction.
Examples of 1-Byte instructions are:
MOV A,B: Move the content of register B to A. 78H is opcode for MOV A,B. The binary form of opcode 78H is 01111000. The first two bit i.e. 01 for MOV operation; the next 3 bits i.e. 111 for register A and last 3 bits 000 are for register B.

Two-Byte instruction.

In case of two byte instruction the 1st byte of the instruction is its opcode and 2nd byte is either data or address.

Ex-MVI B,05; Move 05 to register B

06,05; MVI B,05 in the code form

the 1st byte i.e. 06 is the opcode for MVI B and 2nd byte i.e. 05 is the data which is to be moved to register B.

Three-Byte instruction.

In case of three bytes instruction the 1st byte of instruction is opcode and 2nd and 3rd byte of instruction are either 16-bit data or 16-bit address.

They are stored in three consecutive memory locations.

Ex-LXI H, 2400H ; load H-L pair with 2400H

21,00,24; LXI H, 2400H in code form.

Here 1st byte i.e. 21 is the opcode for instruction LXI H. The 2nd byte i.e. 00 is 8 LSBs of data which is loaded in to register L. The 3rd byte i.e. 24 is 8 MSBs of data which is loaded in to register H.

Instruction Cycle: *An instruction is a command given to the computer to perform a specified operation on a given data. Sequence of instruction is called a program. The CPU fetches one instruction from the memory at a time.*

- CPU carries out to fetch an instruction and data from the memory, and to execute it, constitute an ins*truction cycle.*

- An instruction cycle consists of a Fetch cycle and Execution cycle. In fetch cycle a CPU fetches opcode from the memory.

- The necessary steps which carried out to fetch an opcode from the memory, constitute a fetch cycle.

- The steps are carried out to get data, *if* any, from the memory and to perform the specific operation specified in an instruction called execution cycle.

IC = FC+ EC

Fetch Operation:

The 1st byte of an instruction is its opcode. An instruction may be more than one byte long. The other bytes are data or operand address. The program counter (PC) keeps the memory address of the next instruction to be executed. In the beginning of a fetch cycle the content of the program counter, which is the address of the memory location Where opcode is available, is sent to the memory.

- The entire operation of fetching an opcode takes three clock cycles. A slow memory may take more time.

Execution Operation:

- The opcode fetched from the memory goes to the data register, DR (data/address buffer in Intel 8085) and then to instruction register, IR. From the instruction register it goes to the decoder circuitry which decodes the instruction.

- The decoder circuitry is within the microprocessor. After the instruction is decoded, execution begins.

Execution Operation:

- The opcode fetched from the memory goes to the data register, DR (data/address buffer in Intel 8085) and then to instruction register, IR. From the instruction register it goes to the decoder circuitry which decodes the instruction.

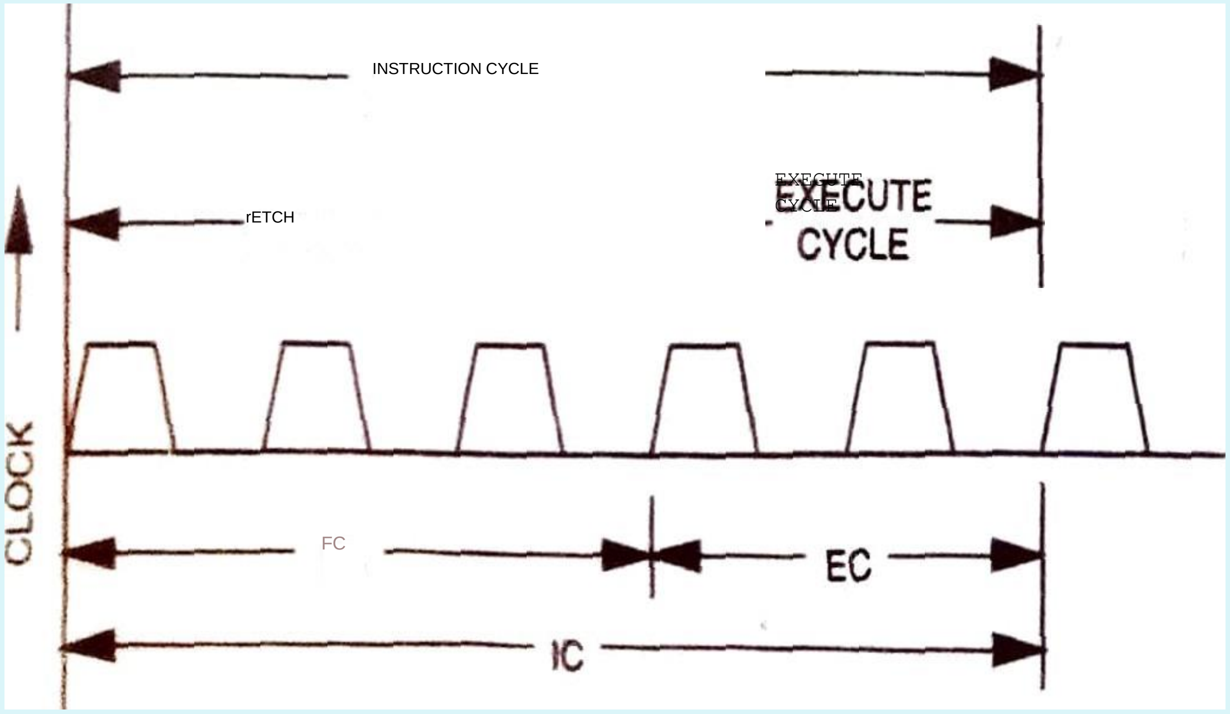
- The decoder circuitry is within the microprocessor. After the instruction is decoded, execution begins.

- If the operand is in general purpose registers, execution is immediately performed.
The time taken in decoding and execution is one clock cycle.

- In read cycle the quantity received from the memory are data or operand address instead of opcode.

- In write cycle data are sent from the CPU to the memory or an output device

Instruction cycle showing FC, EC, IC



Machine Cycle and State:

- To perform the operation of accessing either memory or l/O device, constitute a machine cycle.
- Necessary steps carried out to perform a fetch, a read or a write operation constitute a *Machine cycle and the* steps include to send memory or I/O address; IO/M, ALE, RD (or WR) etc signals.
- In a machine cycle one basic operation such as opcode fetch, memory read, memory write, l/O read or I/O write is performed.
- An instruction cycle consists of several machine cycles.
- The opcode of an instruction is fetched in the first **machine** cycle of an instruction cycle. single-byte instructions require only one machine cycle to fetch the opcode and execute the instruction.
- Two-byte and three-byte instructions require more than one machine cycle.

Timing Diagram:

Timing Diagram is a graphical representation. It represents the execution time taken by each instruction in a graphical format. The execution time is represented in T-states.

Instruction Cycle:

The time required to execute an instruction .

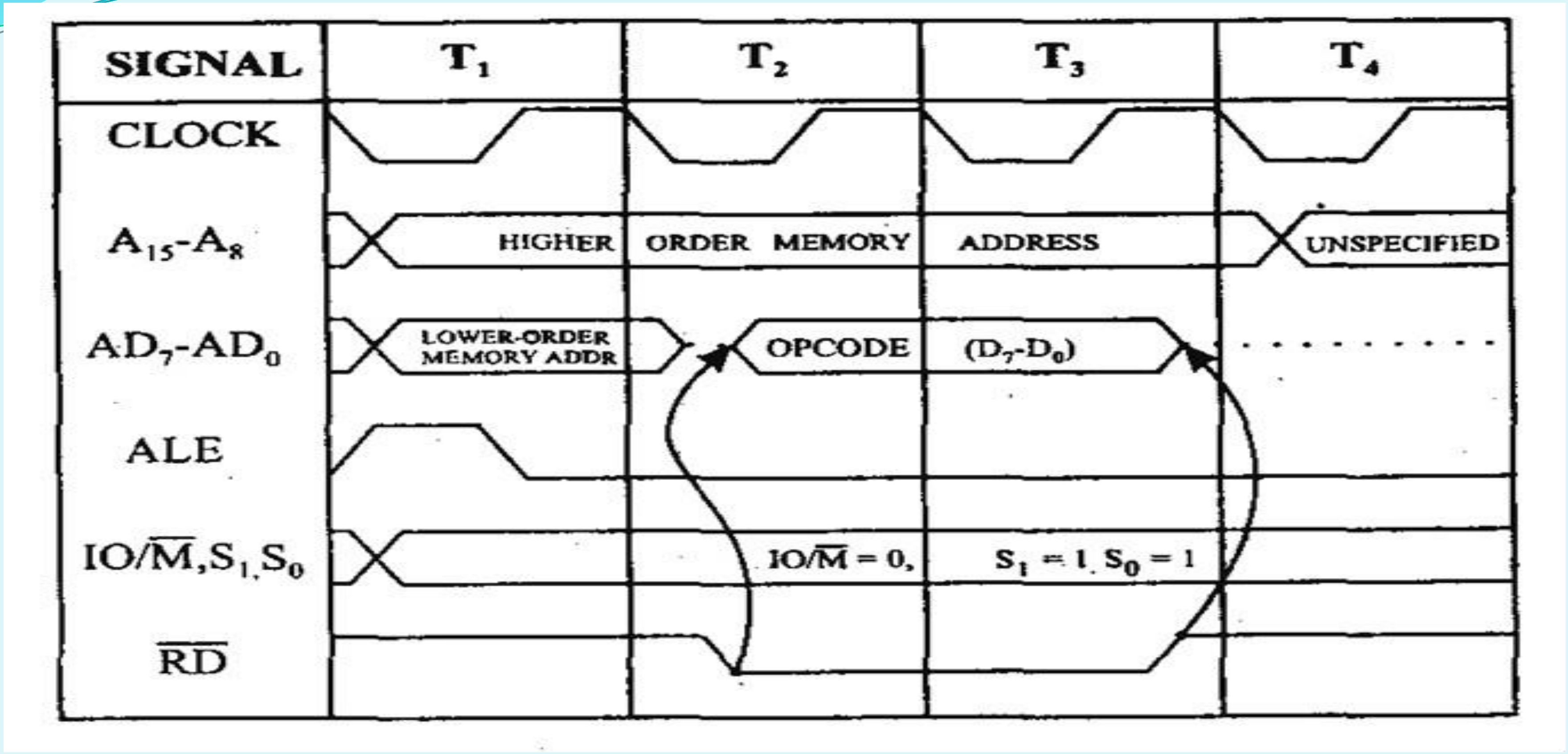
Machine Cycle:

The time required to access the memory or input/output devices .

T-State:

- The machine cycle and instruction cycle takes multiple clock periods.
- A portion of an operation carried out in one system clock period is called as T-state.

Opcode Fetch



OPCODE FETCH

The Opcode fetch cycle, fetches the instructions from memory and delivers it to the instruction register of the Microprocessor. opcode fetch machine cycle consists of 4 T-states.

T1 State:

The contents of the program counter are placed on the 16 bit address bus. The higher order 8 bits are transferred to address bus (A8-A15) and lower order 8 bits are transferred to multiplexed A/D (AD0-AD7) bus.

ALE (address latch enable) signal goes high. As soon as ALE goes high, the memory latches the AD0-AD7 bus. At the middle of the T state the ALE goes low

T2 state:

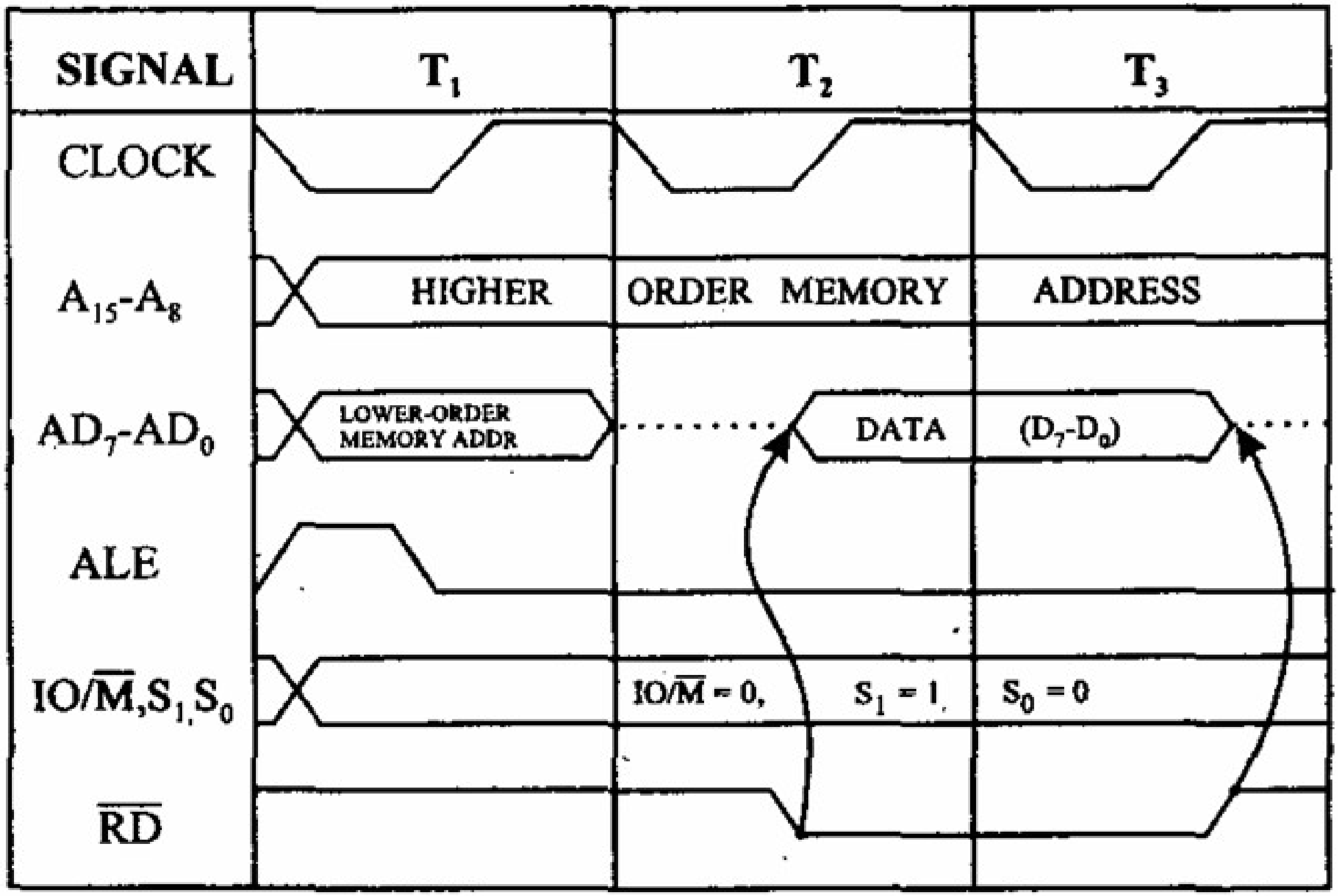
During the beginning of this state, the RD' signal goes low to enable memory. It is during this state, the selected memory location is placed on D0-D7 of the Address/Data multiplexed bus.

T3 State:

The Opcode of the A/D bus is transferred to the instruction register of the microprocessor. Now the **RD' goes high** after this action and thus disables the memory from A/D bus.

T4 Stat

Memory Read Cycle



Memory Read: These machine cycles have 3 T-states.

T1 state: The higher order address bus **(A8-A15)** and lower order address and data multiplexed (AD0-AD7) bus. **ALE goes high** so that the memory latches the (AD0-AD7)

T2 state: Selected memory location is placed on the (D0-D7) of the A/D multiplexed bus. WR' goes **LOW**

T3 State: In the middle of the T3 state WR' goes high and disables the memory write operation. The data which was obtained from the memory is then decoded.

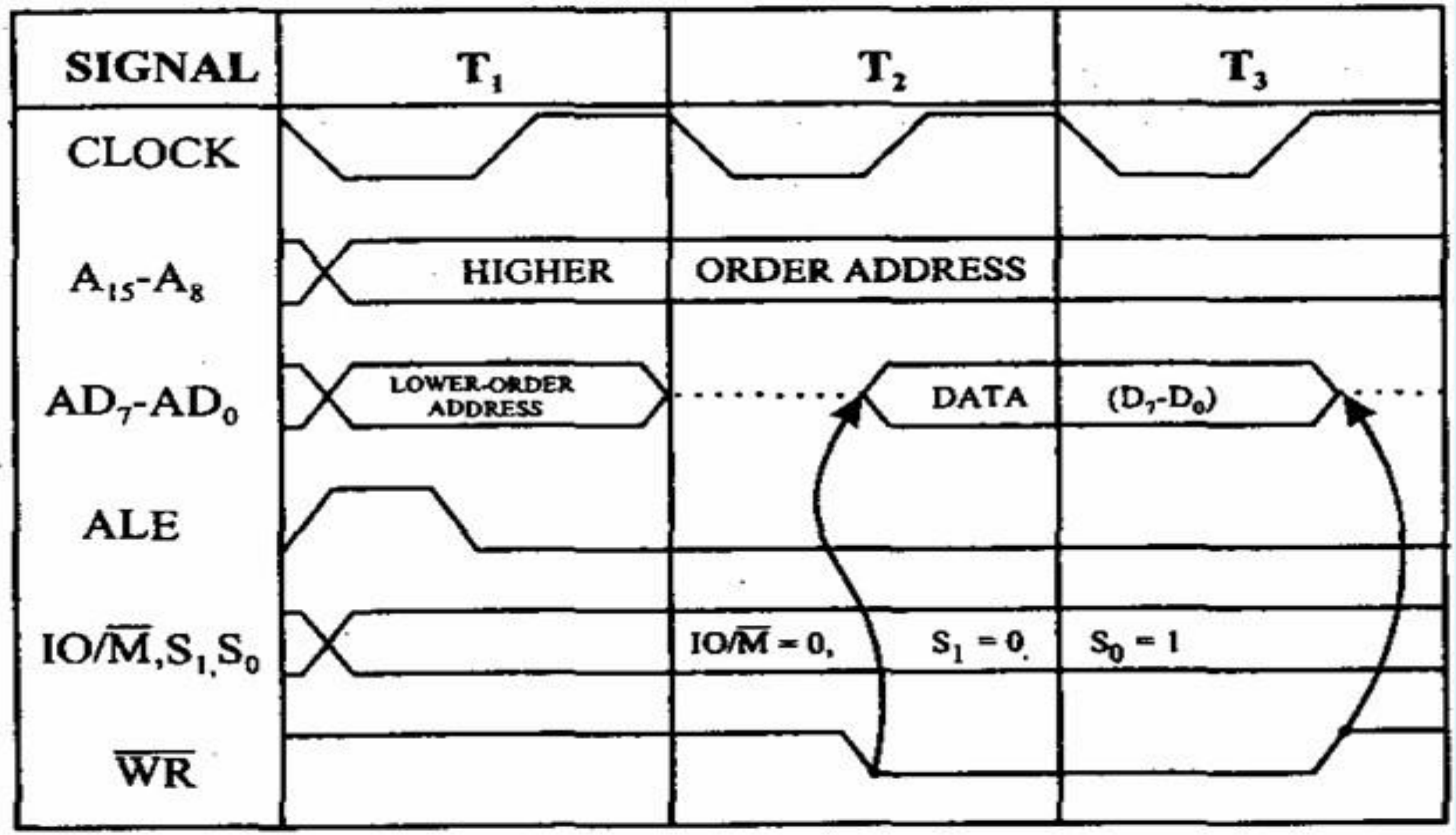
I/O Read:

- **It** reads the data available at an input port or input device. It is similar to memory read. Memory read cycle and I/O read cycle is that Io/M goes high in case of I/O read.

I/O Write:

The CPU sends the data to an IO port or IO device from the accumulator. It is similar to memory write cycle

Memory Write Cycle



Memory Write: These machines cycle have 3 T-states

T1 state:

The higher order address bus **(A8-A15)** and lower order address and data multiplexed (AD0-AD7) bus. **ALE goes high** so that the memory latches the (AD0-AD7) so that complete 16-bit address are available.

The mp identifies the memory read machine cycle from the status signals **IO/M'=0, S1=1, S0=0**. This condition indicates the memory read cycle.

T2 state:

Selected memory location is placed on the (D0-D7) of the A/D multiplexed bus. RD' goes **LOW**

T3 State:

The data which was loaded on the previous state is transferred to the microprocessor. In the middle of the T3 state RD' goes high and disables the memory read operation. The data which was obtained from the memory is then decoded.

INSTRUCTION SET OF 8085

Addressing mode: These are various technique to specify data for instruction

1. Direct addressing mode
2. Register addressing mode
3. Immediate addressing mode
4. Implicit addressing mode.

Direct addressing mode:

In this addressing mode the address of the operand is given in the instruction.

Ex: STA 2000H

```
IN      02H
```

Register addressing mode:

The opcode specify the address of the register and the operation to be Performed.

Ex: MOV A,B

ADD B

Register indirect addressing mode:

In this addressing mode the address of the operand is specify by a register pair.

Immediate addressing mode:

In this addressing mode operand is specify with in the instruction.

Ex: MVI A,05H .

Implicit addressing mode:

This instruction operates on the content of the accumulator.

They don’t required operand address.

EX: CMA,RAL, RAR etc.

Data Transfer Group:

MOV r1,r2 (Move the content of one register to another)

The content of resister r2 is move to register is moved to register .

Ex: MOV A,B moves the contents of resister B to register A.

MOV r, M (Move the content of memory to register)

The content of memory location, whose address is in H-L pair is moved to register r. Example

```
 LXI H,2000H     load H-L pair by 2000H
 MOV B,M         Move the content of the memory location 2000H to register B.
 HLT                     Halt
```

LXI rp, data 16. (load register pair immediate)
This instruction is for register pair.
Ex: LXI H, 2500H loads 2500H into H-L pair.

LDA addr (Load accumulator direct)
The content of memory location , whose address is specified by the 2nd and 3rd bytes of the instruction; is loaded into the accumulator.
Ex: LDA 2400H will load the content of the memory location 2400H into the accumulator .

STA addr (store accumulator direct)
The content of the accumulator is stored in the memory location whose address is specified by the 2nd and 3rd byte of the instruction.STA 2000H will store the content of the accumulator in the memory location 2000H.

LHLD addr (load H-L pair direct).
The content of the memory location ,whose address is specified by 2nd and 3rd bytes of the instruction is loaded into register L.
Example: LHLD 2500H will load the content of the memory location 2500H into register L. The content of the memory location 2501H is loaded into register H.

SHLD addr (store H-L pair direct)
The content of the register L is stored in the memory location whose address is specified by the 2nd and 3rd bytes of the instruction.
Example: SHLD 2500H will stored the content of register L in the memory location 2500H.The content of the register H is stored in the memory location2501H.

LDAX rp. (LOAD accumulator indirect)
The content of the memory location ,whose address is in the register pair rp, is loaded into the accumulator.
Ex: LDAX B will load the content of the memory location, whose address is in B-C pair, into the accumulator.

STAX rp (store accumulator indirect)
The content of the accumulator is stored in the memory location whose address is in the register pair rp.
Example: STAX D will stored the content of the accumulator in the memory location whose address is in D-E pair. This instruction is true only for register pair B-C and D-E.

XCHG (Exchange the content of the H-L with D-E pair)
The content of H-L pair are exchanged with contents of D-E pair

Arithmetic Group: The Instruction of this group performs arithmetic operation.

Add r(Add register to accumulator) [A] $\leftarrow$ [A] + [r]
The content of register r is added to the content of the accumulator, and the sum is placed in the accumulator.

ADD M(Add memory to accumulator) [A]$\leftarrow$ [A] + [H-L]
The content of the memory location addressed by H-L pair is added to the content to the accumulator.

ADC r(Add register with carry to accumulator) [A] $\leftarrow$ [A] + [r] + [CS]
The content of register r and carry status are added to the content of the accumulator.

ADC M (Add memory with carry to accumulator) [A] $\leftarrow$ [A] + [H-L] + [CS]
The content of the memory location addressed by H-L pair add carry status are added to the content of the accumulator.
ADI data (Add immediate data to accumulator)
[A]$\leftarrow$ [A] + data
The immediate data is added to the content to the accumulator.

ACI data (Add with carry immediate data with Accumulator)
[A] ← [A] +data+ [CS]
The 2nd byte of the instruction and the carry status are added to the content of accumulator. The sum is placed in the accumulator.

DAD rp (Add register pair to H-L pair) [H-L]← [H-L]
The content of register pair rp are added to the content of H-L pair and the result is placed in H-L pair

SUB r (Subtract register from accumulator) [A] ← [A]- [r]
The content of register r is subtracted from the content of accumulator and the result is placed in the accumulator.

SUB M (Subtract memory from accumulator) [A] ← [A] –[[H-L]]
The content of the memory location addressed by H-L pair is subtracted from the content of the accumulator. The result is placed in the accumulator.

SBB r (Subtract register from accumulator with borrow. A]← [A] –[r]-[CS].
The content of register r and carry status are subtracted from the content of accumulator. The result is placed in the accumulator.

SBB M (Subtract memory from accumulator with borrow))
[A] ← [A] –[[H-L]]- [CS]
The content of memory location addressed by H-L and carry status are subtracted from the content of accumulator. The result is placed in the accumulator.

SUI data(Subtract immediate data from accumulator) [A]← [A]- data.
The second byte of the instruction is data. It is subtracted from the content of accumulator. The result is placed in the accumulator.

SBI data (Subtract immediate data from accumulator) [A] ← [A]- [data]
The data and carry status are subtracted from the content of accumulator. The result is placed in the accumulator.

INR r (Increment register content) [r] ← [r]+1
The content of register r is increment by one.

INR M (Increment Memory content) [[H-L]]←[[H-L] +1.
The content of Memory location addressed by H-L pair is incremented by one.

DCR r (Decrement register content)[r] ← [r] –1
The content of register r is decremented by one.

DCR M(Decrement the memory content). [[H-L]]← [[H-L]-1.
The content of Memory location addressed by H-L pair is decremented by one.

INX rp. (increment register pair) [rp] ← [rp]+1
The content of register pair rp is increment by one.

DCX rp (decrement register pair) [rp] ← [rp]-1
The content of register pair rp is decrement by one.

DAA. (Decimal adjust accumulator)
The instruction DAA is used in the program after ADD,ADI,ACI,ADC, etc., instructions. After the execution of ADD ,ADC, etc instructions the result is in hexadecimal and it is placed in the accumulator. The DAA instruction operates on this result and gives the final result in decimal system.

Logical Group: The Instruction of this group performs AND, OR, EXCLUSIVE-OR, operation; compare , rotate or take complement of data in register or memory.

ANA r(AND register with accumulator) [A] ← [A] ^ [r]
The content of register r is ANDed with the content of the accumulator, and the result is placed in the accumulator.

ANA M. (AND memory with accumulator) [A] ← [A] ^ [[H-L]]
The content of memory location is addressed by H-L pair is ANDed with the accumulator. The result is placed in the accumulator.

ANI data. (AND immediate data with accumulator).[A]←[A]^data. The 2nd byte of the instruction is data , and it is ANDed with the content of accumulator. Result is placed in the accumulator.

OR r(OR register with accumulator) [A] ← [A] V [r]
The content of register r is ORed with the content of the accumulator, and the result is placed in the accumulator.

ORA M. (OR memory with accumulator) [A] ← [A] V [[H-L]]

ORI data. (OR immediate data with accumulator).[A]←[A]Vdata. The 2nd byte of the instruction is data , and it is ORed with the content of accumulator.

XOR r(EXCLUSIVE-OR register with accumulator) [A] ← [A] XOR [r]
The content of register r is XORed with the content of the accumulator.

XRA M. (EXCLUSIVE-OR memory with accumulator) [A] ← [A] XOR [[H-L]]
The content of memory location is addressed by H-L pair is XORed with the accumulator. The result is placed in the accumulator.

XRI data. (EXCLUSIVE-OR immediate data with accumulator).[A]←[A] XOR data. The 2nd byte of the instruction is data , and it is XORed with the content of accumulator. Result is placed in the accumulator.

CMA. (Complement the accumulator) [A]←$\overline{[A]}$. The 1's complement of the accumulator is obtained and the result is stored in the accumulator.

CMC. (Complement the carry status) [CS]←$\overline{[CS]}$.

STC. (Set carry status)

CMP r(Compare register with accumulator) [A] 🡐 [r]
The content of register r is subtracted from the content of the accumulator, and the status flag are set according to the result of the subtraction.

CMP M. (Compare memory with accumulator) [A] - [[H-L]]
The content of memory location is addressed by H-L pair is subtracted from the content of accumulator and the status flag are set according to the result of the subtraction.

CPI data. (Compare immediate data with accumulator).[A]- data. The 2nd byte of the instruction is data , and it is subtracted from the content of accumulator and the status flag are set according to the result of the subtraction.

RLC. (Rotate accumulator left). The content of accumulator is rotated left by one bit. The seventh bit of the accumulator is moved to carry bit as well to zero bit of the accumulator.

Rotate accumulator left

RLC none

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

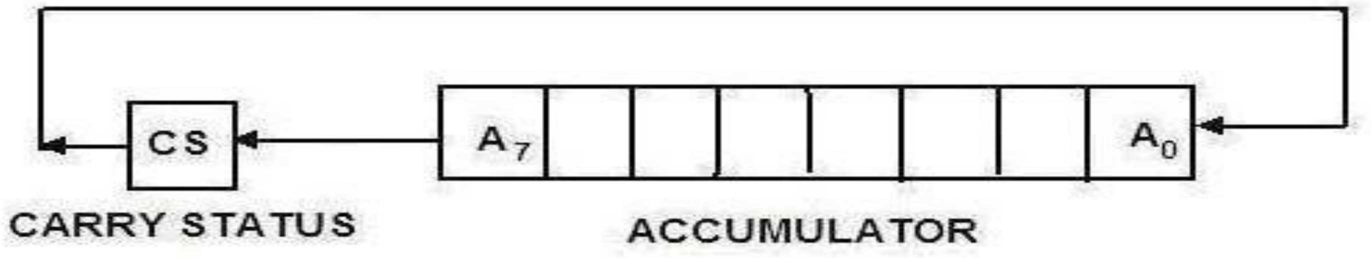


Figure 3.12 Schematic Diagram for RLC

Rotate accumulator right

RRC none

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

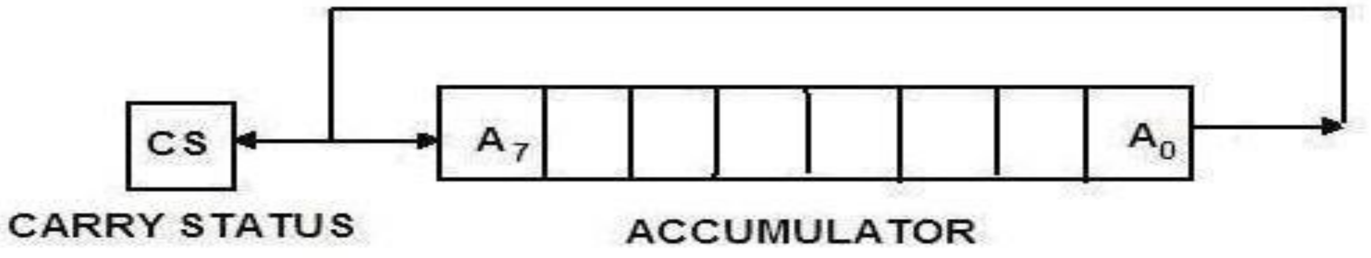


Figure 3.13 Schematic Diagram for RRC

RRC. (Rotate accumulator right). The content of accumulator is rotated right by one bit. The zero bit of the accumulator is moved to seventh bit as well to carry bit.

RAL. (Rotate accumulator left through carry). The content of accumulator is rotated left one bit through carry. The seventh bit of the accumulator is moved to carry, and carry bit is moved to zero bit of the accumulator.

Rotate accumulator right through carry

RAR	none	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 most significant position D7. CY is modified according to bit D0. S, Z, P, AC are not affected. Example: RAR

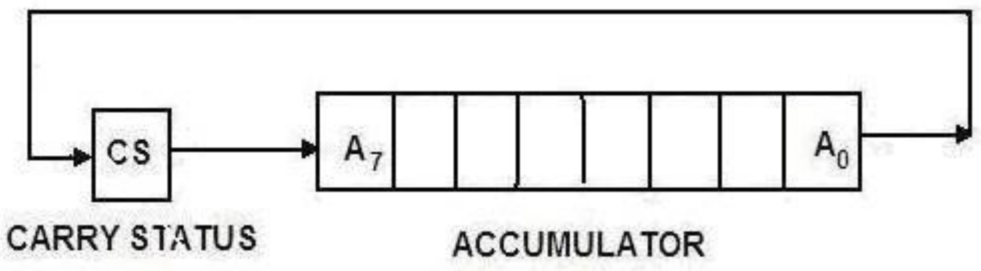


Figure 3.15 Schematic Diagram for RAR

Branch Control Group

The instruction of this group change the normal sequence of the program. There are of two types of branch instruction.

1. Conditional branch instruction
2. Unconditional branch instruction

1. Conditional branch instruction: It transfer the program to the specified level when certain condition is satisfied.

2. Unconditional branch instruction: It transfer the program to the specified level unconditionally. Example:-JMP addr (label).

Conditional Jump addr (label)

JZ addr(label): Jump if the result is zero, Z=1. The program jumps to the instrustion specified by the address(label) if the result is zero.

JNZ addr(label): jump if the result is not zero, Z=0. The program jumps the instruction specified by the address(label) if the result is non-zero.

JC addr (label): jump if there is a carry, CS=1. The program jumps to the instruction specified by the address(label) if there is a carry

JNC addr (label): jump if there is no carry, CS=0. The program jumps to the instruction specified by the address(label) if there is no carry.

JP addr (label): jump if the result is plus, S=0. The program jumps to the instruction specified by the address(label) if the result is plus.

JM addr (label): S=1. if the result the program jumps to the instruction specified by the address(label).

JPE addr (label): jump if even parity, P=1. if the result contains even number of 1s, the program jumps to the instruction specified by the address.

JPO addr (label): jump if odd parity, P=0. if the result contains odd number of 1s, the program jumps to the instruction specified by the address.

CALL addr(label): Used in unconditional branch instruction. It is used to call a sub-routine, before control its transfer to the subroutine .The content of program counter is saved in the stack.

Conditional CALL addr(label):

CC addr (label) Call subroutine if carry status CS=1.
CNC addr (label) Call subroutine if carry status CS=0.
CZ   addr (label) Call subroutine if the result is zero ;the zero status Z=1.
CNZ addr (label) Call subroutine if the result is not zero; the zero status Z=0.
CP addr(label) Call subroutine if the result is plus; the sign status S=0.
CM addr (label) Call subroutine if the result is minus; the sign status S=1.
CPE addr(label) Call subroutine if even parity; the parity status P=1.
CPO addr(label) Call subroutine if odd parity; the parity status P=0.

RET(Return sub routine):

- It is used at the end of a subroutine.
- Before the execution of a subroutine the address of the next instruction of the main program is saved in the stack.
- The content of the stack pointer is incremented by 2 to indicate the new stack top. Then the program jumps to the instruction of the main program next to CALL instruction which is called subroutine.

Conditional Return:

RC	Return from subroutine if carry status CS=1.
RNC	Return from subroutine if carry status CS=0.
RZ	Return from subroutine if the result is zero; the zero status Z=1.
RNZ	Return from subroutine if the result is not zero; the zero status Z=0.
RP	Return from subroutine if the result is plus; the sign status S=0.
RM	Return from subroutine if the result is minus ,the sign status S=1.
RPE	Return from subroutine if even parity, the parity status P=1.
RPO	Return from subroutine if odd parity, the parity status P=0.

RST n (restart) Instruction:

Restart is a one-word CALL instruction. The content of a program counter is saved in the stack, The program jumps to the instruction starting at restart location.

The address of the restart location is 8 times n. There are 8 RST restart instruction carrying from RST0-RST7. These are software interrupts used by the programmer to interrupt the microprocessor.

The restart instruction and location are as follows

Instruction	**Opcode**	**Restart Location**
RST0	C7	0000
RST1	CF	0008
RST2	D7	0010
RST3	DF	0018
RST4	E7	0020
RST5	EF	0028
RST6	F7	0030
RST7	FF	0038

PCHL (Jump to address specified by H-Lpair)
The content of H-L pair are transferred to the program counter. The content of register L will be loaded to 8 LSBs of PC and content of register H will loaded to 8 MSBs.

Stack, I/O and Machine control Group:

IN port-address: (Input to accumulator from IO Port). [A]🡐[Port]
The data available on the port is moved to the accumulator. The Instruction IN, the address of port is specified. Address of the port is an 8-bit port.

OUT Port address:(Output from accumulator to I/O port) [Port]🡐[A]
The content of the accumulator is moved to the port specified by its address. After the OUT instruction the port is specified .The second byte of the instruction contains the address of the port.

PUSH rp. (Push the content of register pair to stack)
[[SP]-1]🡐[rh],
[[SP]-2]🡐[rl],
[SP]🡐([SP]-2).
The content of register pair rp is pushed into the stack.

PUSH PSW. (PUSH program status word to the stack)
[[SP]-1]🡐[A]
[[SP]-2]🡐PSW(Program status word)
[[sp]🡐([SP]-2). The content of accumulator is pushed into the stack .the content

POP rp. (Copy two bytes from the top of the stack into the specified register)
[rl]🡐[[SP]
[rh]🡐[[SP]+1]
[SP]🡐([SP]+2).
The content of register pair which was saved earlier is moved from the stack to the register pair.

POP PSW. (Copy two bytes from the top of the stack into PSW and accumulator)
PSW🡐[[SP]
[A]🡐[[SP]+1]
[SP]🡐([SP]+2).
The content of accumulator which was also saved is moved from the stack to the accumulator.

HLT(Halt). When this instruction is executed any further program execution is stopped.

XTHL(Exchange stack-top with H-L).
[L]🡐>[[SP]]

SPHL. (Move the content of H-L pair to stack pointer).
[H-L]->[SP]

EI.(Enable Interrupts). When this instruction are executed the interrupts are enabled.

DI.(Disable Interrupts). When this instruction are executed the interrupts are disabled.

SIM(Set Interrupts Masks). When this instruction are 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.

RIM. (Read Interrupt Masks). The accumulator is loaded with pending interrupts, the restart interrupt masks and the content of SID.

NOP. (No operation). No operation is performed when this instruction is executed.