Intel 8085 8-bit Microprocessor

Intel 8085 is an 8-bit, NMOS microprocessor. It is a 40 pin C package fabricated on a single LSI chip. The Intel 8085A uses a single +5V D.C supply for its operation. Its clock speed is about 3 MHz. The clock cycle is of 320 ns. The time for the back cycle of the Intel 8085 A-2 is 200 ns. It has 80 basic instructions and 246 opcodes. Figure 1 shows the block diagram of Intel 8085A. It consists of three main sections, an arithmetic and logic unit a timing and control unit and several registers. These important sections are described as under.

ALU

The arithmetic and logic unit, ALU performs the following arithmetic and logic operations.

1. Addition
2. Subtraction
3. Logical AND
4. Logical OR
5. Logical EXCLUSIVE OR
6. Complement (logical NOT)
7. Increment (add 1)
8. Decrement (subtract 1)
9. Left shift (add input to itself)
10. Clear (result is zero)

Timing and Control Unit

The timing and control unit is a section of the CPU. It generates timing and control signals which are necessary for the execution of instructions. It controls provides status, control and timing signals which are required for the operation of memory and I/O

devices. It controls the entire operation of the microprocessor and peripherals consented to it. Thus it is seen that control unit of the CPU acts as a brain of the computer.

Registers

Figure 1 show the various registers of Intel 8085A. Registers are small memories within the CPU. They are used by the microprocessor for temporary storage and manipulation of data and instructions. Data remain in the registers till they are sent to the memory or I/O devices. In a large computer the number of registers is more and hence the program requires less transfer of data to and from the memory. In small computers the number of registers is small due to the limited size of the chip. Intel 8085 microprocessor has the following registers.

1. One 8-bit accumulator (ACC) i.e. register A.
2. Six 8-bit general purpose registers. These are B, C, D, E, H and L.
3. One 16-bit program counter, PC.
4. Instruction register
5. Status register
6. Temporary register.

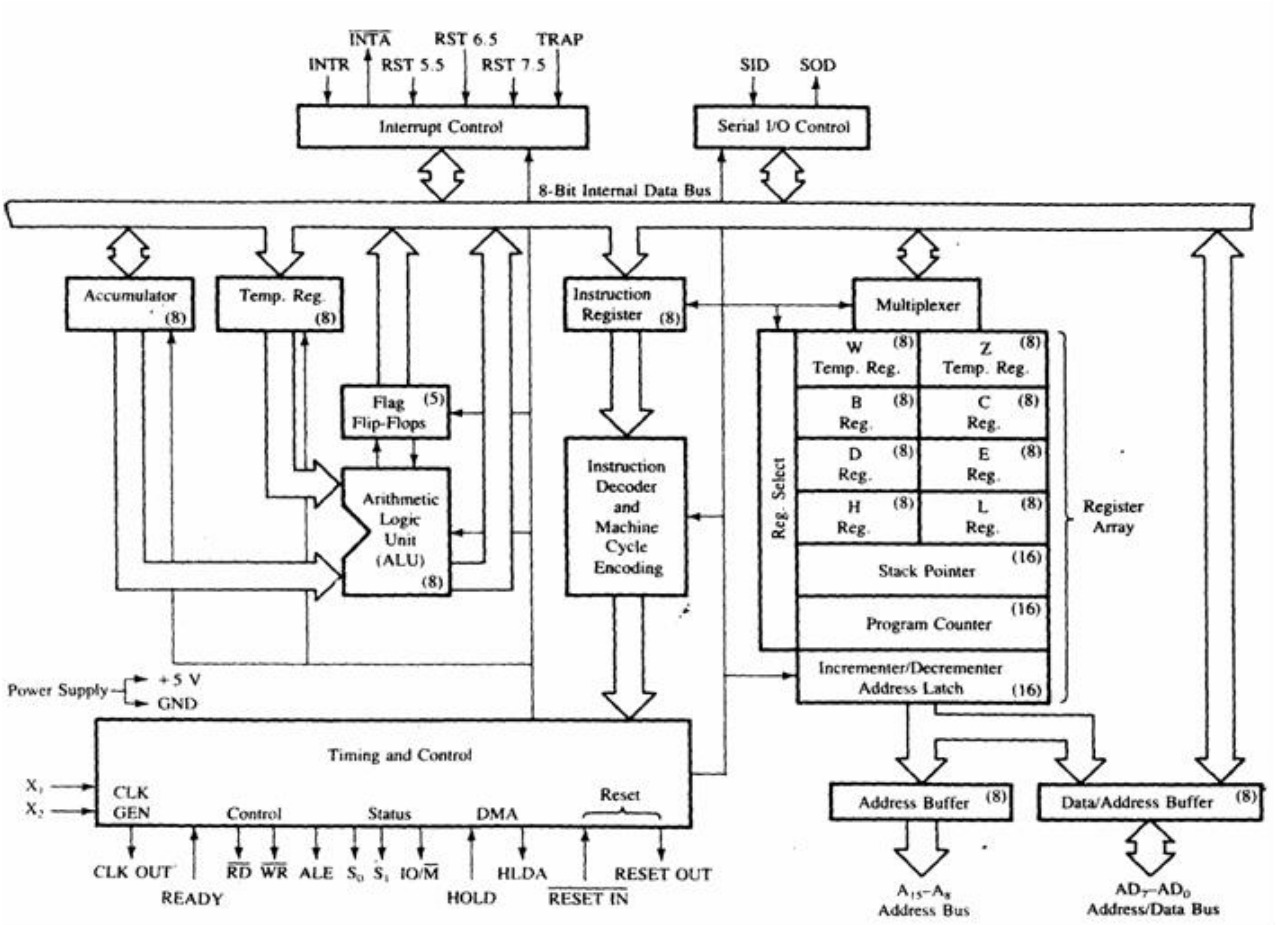


Figure 1: Intel 8085 Microprocessor Internal Block Diagram

In addition to the above mentioned registers Intel 8085 microprocessor also contains address buffer and data/address buffer. Figure 1 the block diagram of Intel 8085. The program counter PC, contains the address of the next instruction. The CPU fetches an instruction from the memory executes it and increments the content of the program counter. Thus in the next instruction cycle it will fetch next instruction. Instructions are executed sequentially unless an instruction changes the content of the program counter. The instruction register holds the instruction until it is decoded. This cannot be accessed by the programmer.

The stack pointer SP, holds the address of the stack top. The stack is a sequence of memory locations defined by the programmer. The stack is used to save the content of a register during the execution of a program. The last memory location of the occupied

portion of the stack is called stack top. For example, suppose that the stack location 2000 is the stack top which is contained by the stack pointer. Now the contents of B-C pair so to be saved. This will be stored in the stack locations 1999 and 1998. The new stack top will be stored in the stack pointer. The new stack top is the location 1998. If more data come they will be stored in the stack location 1997 onwards. Suppose the contents of H-L pair are to be pushed. They will go in 1997 and 1996. The new stack top will be the stack location 1996 and vacant locations are 1995 onward.

There are six 8-bit registers which are used for general purpose as desired by the programmer. These 8-bit registers are A, B, C, D, E, H and L. To handle 16-bit data two 8-bit registers can be combined. The combination of two 8-bit registers is called a register pair. The valid register pairs in Intel 8085 are B-C, D-E and H-L. The H-L pair is used to address memories. The register A is accumulator in Intel 8080/8085. This is for temporary storage used during the execution to a program. It holds one of the operands. The other operand may be either in the memory or in one of the registers.

There is a set of five flip-flops which act as status flags. Each of these flip-flop holds 1-bit flag at indicates certain condition which arises during arithmetic and logic operations. The following status flags have been provided in Intel 8085.

1. CARRY (CS)
2. ZERO (Z)
3. SIGN (S)
4. PARITY (P)
5. AUXILIARY CARRY (AC)

Carry (CS)

The carry status flag 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 comparison, the carry flag CS is set to 1, otherwise 0.

Zero (Z)

The zero status flag Z is set to 1 if the result of an arithmetic or logical operation is zero. For non-zero result it is set to 0.

Sign (S)

The sign status flag is set to 1 if the most significant bit of the result of an arithmetic or logical operation is 1 otherwise 0.

Parity (P)

The parity status flag is set to 1 when result of the operation contains even number of 1's. It is set to zero when there is odd number of 1's.

Auxiliary Carry (AC)

The auxiliary carry status flag holds carry out of bit 3 to 4 resulting from the execution of an arithmetic operation.

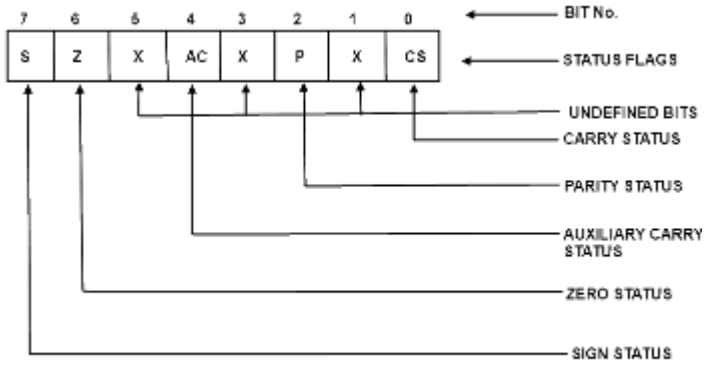


Figure 2: Status Flags of Intel 8085

Figure 3 shows the status flags for ADD operation. Take an example of the instruction ADD B. The execution of the instruction ADD B will add the content of the register B to the contents of the accumulator. Suppose the contents of the accumulator and register B are C.B and ES respectively. Now C.B and ES are added and the result is 01, B4. As the

accumulator is an 8-bit register B4 remains in the accumulator and there is a carry. The various status flags are shown in figure 2.

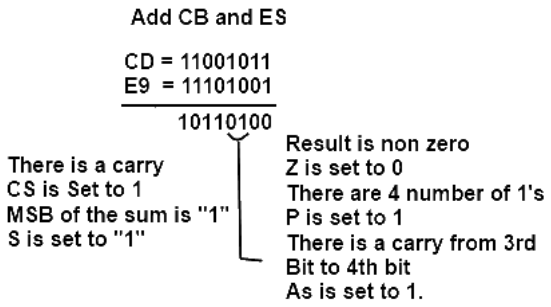


Figure 3: Status Flag for ADD Operation

PSW

In figure 2 Five bits indicate the five status flags and three bits are undefined. The combination of these 8 bits is called Program Status Word (PSW). PSW and the accumulator are treated as a 16-bit unit for stack operations.

Data and Address Bus

The Intel 8085 is an 8-bit microprocessor. Its data bus is 8-bit wide and hence, 8 bits of data can be transmitted in parallel from or to the microprocessor. The Intel 8085 requires a 16-bits. The 8 most significant bits of the address are transmitted by the address bus, (Pins A_8, to A_{15}). The 8 least significant bits of the address are transmitted by address/data bus, (Pins AD_0 to AD_7). The address/data bus transmits data and address at different moments. At a particular moment it transmits either data or address. Thus the AD-bus operates in time shared mode. This technique is known as multiplexing. First of all 16-bit memory address is transmitted by the microprocessor the 8 MSBs of the address on the A-bus and the 8 LSBs of the address on AD-bus. Thus the effective width of the address is latched so that the complete 16-bit address remains available for further operation. The 8-bit AD-bus now becomes free, and it is available for data transmission

2^{16} (i.e. 64K, where 1K = 1024 bytes) memory location can be addressed directly by Intel 8085. Each memory location contains 1 byte of data.

Timing and Control Signals

The timing and control unit generates timing signals for the execution of instruction and control of peripheral devices. The organization of a microprocessor and types of registers differ from processor to processor. The timing used for the execution of instructions and control of peripherals are different for different microprocessors. The selection of a suitable microprocessor for a particular application is a tough task for an engineer. The knowledge of the organization and timing and control system helps an engineer in the selection of a microprocessor. The design and cost of a processor also depends on the timing structure and register organization.

For the execution of an instruction a microprocessor fetches the instruction from the memory and executes it. The time taken for the execution of an instruction is called instruction cycle (IC). An instruction cycle (IC). An instruction cycle consists of a fetch cycle (FC) and an execute cycle (EC). A fetch cycle is the time required for the fetch operation in which the machine code of the instruction (opcode) is fetched from the memory. This time is a fixed slot of time. An execute cycle is of variable width which depends on the instruction to be executed. The total time for the execution is given by

IC = FC + EC

Fetch Operation

In fetch operation the microprocessor gets the 1st byte of the instruction, which is operation code (opcode), from the memory. The program counter keeps the track of address of the next instruction to be executed. In the beginning of the fetch cycle the content of the program counter is sent to the memory. This takes one clock cycle. The memory first reads the opcode. This operation also takes one clock cycle. Then the memory sends the opcode to the microprocessor, which takes one clock period. The total

time for fetch operation is the time required for fetching an opcode from the memory. This time is called fetch cycle. Having received the address from the microprocessor the memory takes two clock cycles to respond as explained above. If the memory is slow, it may take more time. In that case the microprocessor has to wait for some time till it receives the opcode from the memory. The time for which the microprocessor waits is called wait cycle. Most of the microprocessor have provision for wait cycles to cope with slow memory.

Execute 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 is within the microprocessor. After the instruction is decoded, execution begins. If the operand is in the general purpose registers, execution is immediately performed. The time taken in decoding and the address of the data, some read cycles are also necessary to receive the data from the memory. These read cycle are similar to opcode fetch cycle. The fetch quantities in these cycles are address or data. Figure 4 (a) and Figure 4 (b) shows an instruction and fetch cycle respectively.

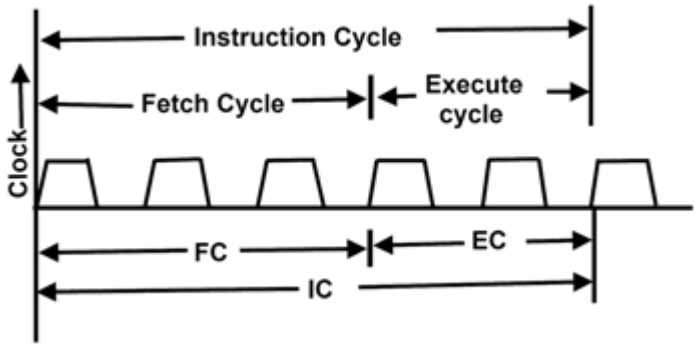


Figure (a) Instruction cycle showing FC, EC and IC

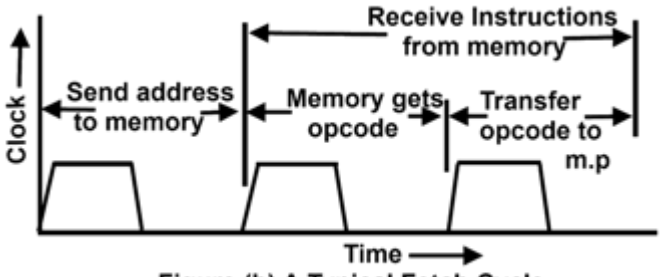


Figure (b) A Typical Fetch Cycle

Figure 4

Machine Cycle

An instruction cycle consists of one or more machine cycles as shown in Figure 5. This figure is for MVI instruction. A machine cycle consists of a number of clock cycles. One clock cycle is known as state.

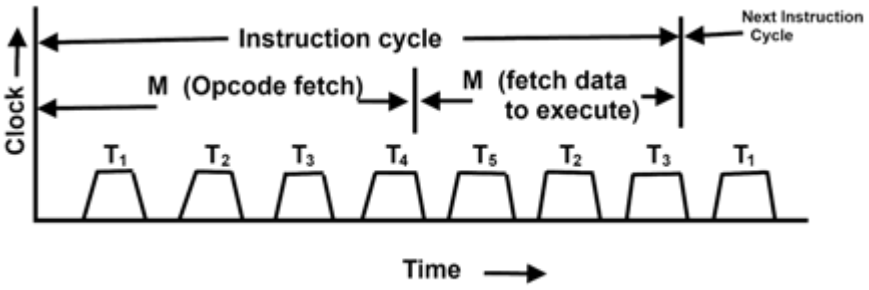


Figure 5: A Typical Instruction Cycle

Applications of Microprocessor

Microprocessor are being used for numerous applications and the list of applications is becoming longer and longer. To give an idea of microprocessor applications few areas are given below.

1. Personal Computer
2. Numerical Control
3. Mobile Phones
4. Automobiles
5. Bending Machines
6. Medical Diagnostic Equipment
7. Automatic voice recognizing systems
8. Prosthetics
9. Traffic light Control
10. Entertainment Games
11. Digital Signal Processing
12. Communication terminals
13. Process Control
14. Calculators
15. Sophisticated Instruments
16. Telecommunication Switching Systems
17. Automatic Test Systems.

INSTRUCTION SET OF INTEL 8085

An Instruction is a command given to the computer to perform a specified operation on given data. The instruction set of a microprocessor is the collection of the instructions that the microprocessor is designed to execute. The instructions described here are of Intel 8085. These instructions are of Intel Corporation. They cannot be used by other microprocessor manufactures. The programmer can write a program in assembly language using these instructions. These instructions have been classified into the following groups:

1. **Data Transfer Group**
2. **Arithmetic Group**
3. **Logical Group**
4. **Branch Control Group**
5. **I/O and Machine Control Group**

Data Transfer Group

Instructions, which are used to transfer data from one register to another register, from memory to register or register to memory, come under this group. Examples are: MOV, MVI, LXI, LDA, STA etc. When an instruction of data transfer group is executed, data is transferred from the source to the destination without altering the contents of the source. For example, when MOV A, B is executed the content of the register B is copied into the register A, and the content of register B remains unaltered. Similarly, when LDA 2500 is executed the content of the memory location 2500 is loaded into the accumulator. But the content of the memory location 2500 remains unaltered.

Arithmetic Group

The instructions of this group perform arithmetic operations such as addition, subtraction; increment or decrement of the content of a register or memory. Examples are: ADD, SUB, INR, DAD etc.

Logical Group

The Instructions under this group perform logical operation such as AND, OR, compare, rotate etc. Examples are: ANA, XRA, ORA, CMP, and RAL etc.

Branch Control Group

This group includes the instructions for conditional and unconditional jump, subroutine call and return, and restart. Examples are: JMP, JC, JZ, CALL, CZ, RST etc.

I/O and Machine Control Group

This group includes the instructions for input/output ports, stack and machine control. Examples are: IN, OUT, PUSH, POP, and HLT etc.

Intel 8085 Instructions

1. Data Transfer Group

1. MOV r1, r2 (Move Data; Move the content of the one register to another). [r1] <-- [r2]
2. MOV r, m (Move the content of memory register). r <-- [M]
3. MOV M, r. (Move the content of register to memory). M <-- [r]
4. MVI r, data. (Move immediate data to register). [r] <-- data.
5. MVI M, data. (Move immediate data to memory). M <-- data.
6. LXI rp, data 16. (Load register pair immediate). [rp] <-- data 16 bits, [rh] <-- 8 LSBs of data.
7. LDA addr. (Load Accumulator direct). [A] <-- [addr].
8. STA addr. (Store accumulator direct). [addr] <-- [A].
9. LHLD addr. (Load H-L pair direct). [L] <-- [addr], [H] <-- [addr+1].
10. SHLD addr. (Store H-L pair direct) [addr] <-- [L], [addr+1] <-- [H].
11. LDAX rp. (LOAD accumulator indirect) [A] <-- [[rp]]

12. STAX rp. (Store accumulator indirect) [[rp]] <-- [A].
13. XCHG. (Exchange the contents of H-L with D-E pair) [H-L] <--> [D-E].

2. Arithmetic Group

1. ADD r. (Add register to accumulator) [A] <-- [A] + [r].
2. ADD M. (Add memory to accumulator) [A] <-- [A] + [[H-L]].
3. ADC r. (Add register with carry to accumulator). [A] <-- [A] + [r] + [CS].
4. ADC M. (Add memory with carry to accumulator) [A] <-- [A] + [[H-L]] [CS].
5. ADI data (Add immediate data to accumulator) [A] <-- [A] + data.
6. ACI data (Add with carry immediate data to accumulator). [A] <-- [A] + data + [CS].
7. DAD rp. (Add register paid to H-L pair). [H-L] <-- [H-L] + [rp].
8. SUB r. (Subtract register from accumulator). [A] <-- [A] – [r].
9. SUB M. (Subtract memory from accumulator). [A] <-- [A] – [[H-L]].
10. SBB r. (Subtract register from accumulator with borrow). [A] <-- [A] – [r] – [CS].
11. SBB M. (Subtract memory from accumulator with borrow). [A] <-- [A] – [[H-L]] – [CS].
12. SUI data. (Subtract immediate data from accumulator) [A] <-- [A] – data.
13. SBI data. (Subtract immediate data from accumulator with borrow). [A] <-- [A] – data – [CS].
14. INR r (Increment register content) [r] <-- [r] +1.
15. INR M. (Increment memory content) [[H-L]] <-- [[H-L]] + 1.
16. DCR r. (Decrement register content). [r] <-- [r] – 1.
17. DCR M. (Decrement memory content) [[H-L]] <-- [[H-L]] – 1.
18. INX rp. (Increment register pair) [rp] <-- [rp] – 1.
19. DCX rp (Decrement register pair) [rp] <-- [rp] -1.
20. DAA (Decimal adjust accumulator) .

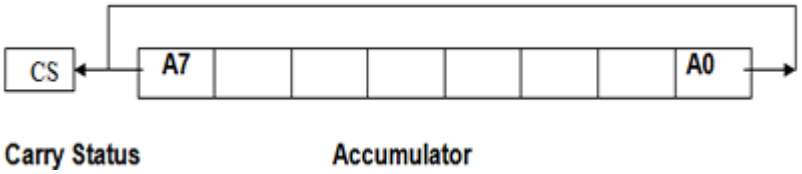
The instruction DAA is used in the program after ADD, ADI, ACI, ADC, etc instructions. After the execution of ADD, ADC, etc instructions the result is in hexadecimal and it is placed in the accumulator. The DAA instruction operates on this result and gives the final result in the decimal system. It uses carry and auxiliary carry for decimal adjustment. 6 is added to 4 LSBs of the content of the accumulator if their value lies in between A and F or the AC flag is set to 1. Similarly, 6 is also added to 4 MSBs of the content of the accumulator if their value lies in between A and F or the CS flag is set to 1. All status flags are affected. When DAA is used data should be in decimal numbers.

3. Logical Group

1. ANA r. (AND register with accumulator) [A] <-- [A] ^ [r].
2. ANA M. (AND memory with accumulator). [A] <-- [A] ^ [[H-L]].
3. ANI data. (AND immediate data with accumulator) [A] <-- [A] ^ data.
4. ORA r. (OR register with accumulator) [A] <-- [A] v [r].
5. ORA M. (OR memory with accumulator) [A] <-- [A] v [[H-L]]
6. ORI data. (OR immediate data with accumulator) [A] <-- [A] v data.
7. XRA r. (EXCLUSIVE – OR register with accumulator) [A] <-- [A] v [r]
8. XRA M. (EXCLUSIVE-OR memory with accumulator) [A] <-- [A] v [[H-L]]
9. XRI data. (EXCLUSIVE-OR immediate data with accumulator) [A] <-- [A]
10. CMA. (Complement the accumulator) [A] <-- [A]
11. CMC. (Complement the carry status) [CS] <-- [CS]
12. STC. (Set carry status) [CS] <-- 1.
13. CMP r. (Compare register with accumulator) [A] – [r]
14. CMP M. (Compare memory with accumulator) [A] – [[H-L]]
15. CPI data. (Compare immediate data with accumulator) [A] – data.

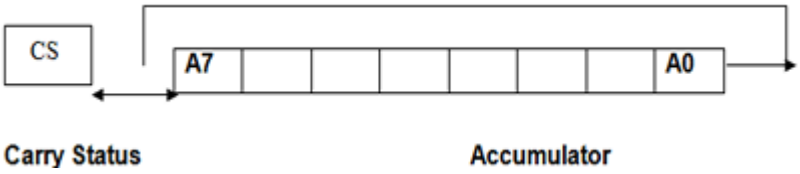
The 2nd byte of the instruction is data, and it is subtracted from the content of the accumulator. The status flags are set according to the result of subtraction. But the result is discarded. The content of the accumulator remains unchanged.

16. RLC (Rotate accumulator left) [An+1] <-- [An], [A0] <-- [A7],[CS] <-- [A7]**.**



The content of the accumulator is rotated left by one bit. The seventh bit of the accumulator is moved to carry bit as well as to the zero bit of the accumulator. Only CS flag is affected.

17. RRC. (Rotate accumulator right) [A7] <-- [A0], [CS] <-- [A0], [An] <-- [An+1].



The content of the accumulator is rotated right by one bit. The zero bit of the accumulator is moved to the seventh bit as well as to carry bit. Only CS flag is affected.

18. RAL. (Rotate accumulator left through carry) [An+1] <-- [An], [CS] <-- [A7], [A0] <-- [CS].
19. RAR. (Rotate accumulator right through carry) [An] <-- [An+1], [CS] <-- [A0], [A7] <-- [CS]

4. Branch Group

1. JMP addr (label). (Unconditional jump: jump to the instruction specified by the address). [PC] <-- Label.
2. Conditional Jump addr (label): After the execution of the conditional jump instruction the program jumps to the instruction specified by the address (label) if the specified condition is fulfilled. The program proceeds further in the normal sequence if the specified condition is not fulfilled. If the condition is true and program jumps to the specified label, the execution of a conditional jump takes 3 machine cycles: 10 states. If condition is not true, only 2 machine cycles; 7 states are required for the execution of the instruction.
    1. **JZ** addr (label). (Jump if the result is zero)
    2. **JNZ** addr (label) (Jump if the result is not zero)
    3. **JC** addr (label). (Jump if there is a carry)
    4. **JNC** addr (label). (Jump if there is no carry)
    5. **JP** addr (label). (Jump if the result is plus)
    6. **JM** addr (label). (Jump if the result is minus)
    7. **JPE** addr (label) (Jump if even parity)
    8. **JPO** addr (label) (Jump if odd parity)
3. CALL addr (label) (Unconditional CALL: call the subroutine identified by the operand)

    CALL instruction is used to call a subroutine. Before the control is transferred to the subroutine, the address of the next instruction of the main program is saved in the stack. The content of the stack pointer is decremented by two to indicate the new stack top. Then the program jumps to subroutine starting at address specified by the label.

4. RET (Return from subroutine)

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

5. Stack, I/O and Machine Control Group

1. IN port-address. (Input to accumulator from I/O port) [A] <-- [Port]
2. OUT port-address (Output from accumulator to I/O port) [Port] <-- [A]
3. PUSH rp (Push the content of register pair to stack)
4. PUSH PSW (PUSH Processor Status Word)
5. POP rp (Pop the content of register pair, which was saved, from the stack)
6. POP PSW (Pop Processor Status Word)
7. HLT (Halt)
8. XTHL (Exchange stack-top with H-L)
9. SPHL (Move the contents of H-L pair to stack pointer)
10. EI (Enable Interrupts)
11. DI (Disable Interrupts)
12. SIM (Set Interrupt Masks)
13. RIM (Read Interrupt Masks)
14. NOP (No Operation)

ADDRESSING MODES OF 8085

- Every instruction of a program has to operate on a data.
- The method of specifying the data to be operated by the instruction is called Addressing.
- The 8085 has the following 5 different types of addressing.

1. **Immediate Addressing**
2. **Direct Addressing**
3. **Register Addressing**
4. **Register Indirect Addressing**
5. **Implied Addressing**

1. Immediate Addressing:

- In immediate addressing mode, the data is specified in the instruction itself. The data will be a part of the program instruction.
- EX. MVI B, 3EH - Move the data 3EH given in the instruction to B register; LXI SP, 2700H.

2. Direct Addressing:

- In direct addressing mode, the address of the data is specified in the instruction. The data will be in memory. In this addressing mode, the program instructions and data can be stored in different memory.
- EX. LDA 1050H - Load the data available in memory location 1050H in to accumulator; SHLD 3000H

3. Register Addressing:

- In register addressing mode, the instruction specifies the name of the register in which the data is available.
- EX. MOV A, B - Move the content of B register to A register; SPHL; ADD C.

4. Register Indirect Addressing:

- In register indirect addressing mode, the instruction specifies the name of the register in which the address of the data is available. Here the data will be in memory and the address will be in the register pair.
- EX. MOV A, M - The memory data addressed by H L pair is moved to A register. LDAX B.

5. Implied Addressing:

- In implied addressing mode, the instruction itself specifies the data to be operated.
- EX. CMA - Complement the content of accumulator; RAL

Store 8-bit data in memory

Program 1:

- MVI A, 52H : "Store 32H in the accumulator"
- STA 4000H : "Copy accumulator contents at address 4000H"
- HLT : "Terminate program execution"

Program 2:

- LXI H : "Load HL with 4000H"
- MVI M : "Store 32H in memory location pointed by HL register pair (4000H)"
- HLT : "Terminate program execution"

Note: The result of both programs will be the same. In program 1 direct addressing instruction is used, whereas in program 2 indirect addressing instruction is used.

Exchange the contents of memory locations

Statement: Exchange the contents of memory locations 2000H and 4000H.

Program 1

- LDA 2000H : "Get the contents of memory location 2000H into accumulator"
- MOV B, A : "Save the contents into B register"
- LDA 4000H : "Get the contents of memory location 4000Hinto accumulator"
- STA 2000H : "Store the contents of accumulator at address 2000H"
- MOV A, B : "Get the saved contents back into A register"
- STA 4000H : "Store the contents of accumulator at address 4000H"

Program 2

- LXI H 2000H : "Initialize HL register pair as a pointer to memory location 2000H."
- LXI D 4000H : "Initialize DE register pair as a pointer to memory location 4000H."
- MOV B, M : "Get the contents of memory location 2000H into B register."
- LDAX D : "Get the contents of memory location 4000H into A register."
- MOV M, A : "Store the contents of A register into memory location 2000H."
- MOV A, B : "Copy the contents of B register into accumulator."
- STAX D : "Store the contents of A register into memory location 4000H."
- HLT : "Terminate program execution."

Note: In Program 1, direct addressing instructions are used, whereas in Program 2, indirect addressing instructions are used.

Add two 8-bit numbers

Statement: Add the contents of memory locations 4000H and 4001H and place the result in memory location 4002H.

- (4000H) = 14H
- (4001H) = 89H
- Result = 14H + 89H = 9DH

- LXI H 4000H : "HL points 4000H"
- MOV A, M : "Get first operand"
- INX H : "HL points 4001H"
- ADD M : "Add second operand"
- INX H : "HL points 4002H"
- MOV M, A : "Store result at 4002H"
- HLT : "Terminate program execution"

Subtract two 8-bit numbers

Statement: Subtract the contents of memory location 4001H from the memory location 2000H and place the result in memory location 4002H.

- **Program –: Subtract two 8-bit numbers**
- (4000H) = 51H
- (4001H) = 19H
- Result = 51H – 19H = 38H

- LXI H, 4000H : "HL points 4000H"
- MOV A, M : "Get first operand"
- INX H : "HL points 4001H"
- SUB M : "Subtract second operand"
- INX H : "HL points 4002H"
- MOV M, A : "Store result at 4002H"
- HLT : "Terminate program execution"

Add two 16-bit numbers

Statement: Add the 16-bit number in memory locations 4000H and 4001H to the 16-bit number in memory locations 4002H and 4003H. The most significant eight bits of the two numbers to be added are in memory locations 4001H and 4003H. Store the result in memory locations 4004H and 4005H with the most significant byte in memory location 4005H.

- (4000H) = 15H
- (4001H) = 1CH
- (4002H) = B7H
- (4003H) = 5AH
- Result = 1C15 + 5AB7H = 76CCH
- (4004H) = CCH
- (4005H) = 76H

1. Source Program 1:
2. LHLD 4000H : "Get first I6-bit number in HL"
3. XCHG : "Save first I6-bit number in DE"
4. LHLD 4002H : "Get second I6-bit number in HL"
5. MOV A, E : "Get lower byte of the first number"
6. ADD L : "Add lower byte of the second number"
7. MOV L, A : "Store result in L register"
8. MOV A, D : "Get higher byte of the first number"
9. ADC H : "Add higher byte of the second number with CARRY"
10. MOV H, A : "Store result in H register"
11. SHLD 4004H : "Store I6-bit result in memory locations 4004H and 4005H"

12. HLT : "Terminate program execution"

1. Source program 2:
2. LHLD 4000H : Get first I6-bit number
3. XCHG : Save first I6-bit number in DE
4. LHLD 4002H : Get second I6-bit number in HL
5. DAD D : Add DE and HL
6. SHLD 4004H : Store I6-bit result in memory locations 4004H and 4005H.
7. HLT : Terminate program execution

NOTE: In program 1, eight bit addition instructions are used (ADD and ADC) and addition is performed in two steps. First lower byte addition using ADD instruction and then higher byte addition using ADC instruction.In program 2, 16-bit addition instruction (DAD) is used.

Add contents of two memory locations

Statement: Add the contents of memory locations 40001H and 4001H and place the result in the memory locations 4002Hand 4003H.

- Sample problem:
- (4000H) = 7FH
- (400lH) = 89H
- Result = 7FH + 89H = lO8H
- (4002H) = 08H
- (4003H) = 0lH

1. Source program:
2. LXI H, 4000H : "HL Points 4000H"
3. MOV A, M : "Get first operand"
4. INX H : "HL Points 4001H"
5. ADD M : "Add second operand"

6. INX H : "HL Points 4002H"
7. MOV M, A : "Store the lower byte of result at 4002H"
8. MVIA, 00 : "Initialize higher byte result with 00H"
9. ADC A : "Add carry in the high byte result"
10. INX H : "HL Points 4003H"
11. MOV M, A : "Store the higher byte of result at 4003H"
12. HLT : "Terminate program execution"

Subtract two 16-bit numbers

Statement: Subtract the 16-bit number in memory locations 4002H and 4003H from the 16-bit number in memory locations 4000H and 4001H. The most significant eight bits of the two numbers are in memory locations 4001H and 4003H. Store the result in memory locations 4004H and 4005H with the most significant byte in memory location 4005H.

- Sample problem:
- (4000H) = 19H
- (400IH) = 6AH
- (4004H) = I5H (4003H) = 5CH
- Result = 6A19H – 5C15H = OE04H
- (4004H) = 04H
- (4005H) = OEH

1. Source program:
2. LHLD 4000H : "Get first 16-bit number in HL"
3. XCHG : "Save first 16-bit number in DE"
4. LHLD 4002H : "Get second 16-bit number in HL"
5. MOV A, E : "Get lower byte of the first number"
6. SUB L : "Subtract lower byte of the second number"
7. MOV L, A : "Store the result in L register"

8. MOV A, D : "Get higher byte of the first number"
9. SBB H : "Subtract higher byte of second number with borrow"
10. MOV H, A : "Store l6-bit result in memory locations 4004H and 4005H"
11. SHLD 4004H : "Store l6-bit result in memory locations 4004H and 4005H"
12. HLT : "Terminate program execution"

Finding one's complement of a number

Statement: Find the l's complement of the number stored at memory location 4400H and store the complemented number at memory location 4300H.

- Sample problem:
- (4400H) = 55H
- Result = (4300B) = AAB

1. Source program:
2. LDA 4400B : "Get the number"
3. CMA : "Complement number"
4. STA 4300H : "Store the result"
5. HLT : "Terminate program execution"

Finding Two's complement of a number

Statement: Find the 2′s complement of the number stored at memory location 4200H and store the complemented number at memory location 4300H

- Sample problem:
- (4200H) = 55H
- Result = (4300H) = AAH + 1 = ABH

1. Source program:
2. LDA 4200H : "Get the number"
3. CMA : "Complement the number"
4. ADI, 01 H : "Add one in the number"
5. STA 4300H : "Store the result"
6. HLT : "Terminate program execution"

Pack the unpacked BCD numbers

Statement: Pack the two unpacked BCD numbers stored in memory locations 4200H and 4201H and store result in memory location 4300H. Assume the least significant digit is stored at 4200H.

- Sample problem:
- (4200H) = 04
- (4201H) = 09
- Result = (4300H) = 94

1. Source program:
2. LDA 4201H : "Get the Most significant BCD digit"
3. RLC
4. RLC
5. RLC
6. RLC : "Adjust the position of the second digit (09 is changed to 90)"
7. ANI FOH : "Make least significant BCD digit zero"
8. MOV C, A : "store the partial result"
9. LDA 4200H : "Get the lower BCD digit"
10. ADD C : "Add lower BCD digit"
11. STA 4300H : "Store the result"
12. HLT : "Terminate program execution"

Unpack a BCD number

Statement: Two digit BCD number is stored in memory location 4200H. Unpack the BCD number and store the two digits in memory locations 4300H and 4301H such that memory location 4300H will have lower BCD digit.

- Sample problem:
- (4200H) = 58
- Result = (4300H) = 08 and
- (4301H) = 05

1. Source program:
2. LDA 4200H : "Get the packed BCD number"
3. ANI FOH : "Mask lower nibble"
4. RRC
5. RRC
6. RRC
7. RRC : "Adjust higher BCD digit as a lower digit"
8. STA 4301H : "Store the partial result"
9. LDA 4200H : "Get the original BCD number"
10. ANI OFH : "Mask higher nibble"
11. STA 4201H : "Store the result"
12. HLT : "Terminate program execution"

Execution format of instructions

Statement: Read the program given below and state the contents of all registers after the execution of each instruction in sequence.

1. Main program:

2. 4000H LXI SP, 27FFH
3. 4003H LXI H, 2000H
4. 4006H LXI B, 1020H
5. 4009H CALL SUB
6. 400CH HLT
7. Subroutine program:
8. 4100H SUB: PUSH B
9. 4101H PUSH H
10. 4102H LXI B, 4080H
11. 4105H LXI H, 4090H
12. 4108H SHLD 2200H
13. 4109H DAD B
14. 410CH POP H
15. 410DH POP B
16. 410EH RET

Right shift, bit of data(8 bit and 16 bit)

Statement: Write a program to shift an eight bit data four bits right. Assume data is in register C.

- Sample problem:
- (4200H) = 58
- Result = (4300H) = 08 and
- (4301H) = 05

1.
2. Source program 1:
3. MOV A, C
4. RAR
5. RAR

6. RAR
7. RAR
8. MOV C, A
9. HLT

Statement: Write a program to shift a 16 bit data, 1 bit right. Assume that data is in BC register pair.

1. Source program 2
2. MOV A, B
3. RAR
4. MOV B, A
5. MOV A, C
6. RAR
7. MOV C, A
8. HLT

INSTRUCTION EXECUTION AND DATA FLOW IN 8085

The program instructions are stored in memory, which is an external device. To execute a program in 8085, the starting address of the program should be loaded in program counter. The 8085 output the content of program counter in address bus and asserts read control signal low. Also, the program counter is incremented. The address and the read control signal enable the memory to output the content of memory location on the data bus. Now the content of data bus is the opcode of an instruction. The read control signal is made high by timing and control unit after a specified time. At the rising edge of read control signals, the opcode is latched into microprocessor internal bus and placed in instruction register. The instruction-decoding unit, decodes the instructions and provides information to timing and control unit to take further actions.

INSTRUCTION FORMAT OF 8085

The 8085 have 74 basic instructions and 246 total instructions. The instruction set of 8085 is defined by the manufacturer Intel Corporation. Each instruction of 8085 has 1 byte opcode. With 8 bit binary code, we can generate 256 different binary codes. In this, 246 codes have been used for opcodes.The size of 8085 instructions can be 1 byte, 2 bytes or 3 bytes.

- The 1-byte instruction has an opcode alone.
- The 2 bytes instruction has an opcode followed by an eight-bit address or data.
- The 3 bytes instruction has an opcode followed by 16 bit address or data.

While storing the 3 bytes instruction in memory, the sequence of storage is, opcode first followed by low byte of address or data and then high byte of address or data.

Interrupts In 8085

What is Interrupt?

Interrupt is a mechanism by which an I/O or an instruction can suspend the normal execution of processor and get itself serviced. Generally, a particular task is assigned to that interrupt signal. In the microprocessor based system the interrupts are used for data transfer between the peripheral devices and the microprocessor**.**

Interrupt Service Routine (ISR)

A small program or a routine that when executed services the corresponding interrupting source is called as an ISR.

Maskable/Non-Maskable Interrupt

An interrupt that can be disabled by writing some instruction is known as Maskable Interrupt otherwise it is called Non-Maskable Interrupt.

There are 6 pins available in 8085 for interrupt:

1. TRAP
2. RST 7.5
3. RST6.5
4. RST5.5
5. INTR
6. INTA

Execution of Interrupts

When there is an interrupt requests to the Microprocessor then after accepting the interrupts Microprocessor send the INTA (active low) signal to the peripheral. The vectored address of particular interrupt is stored in program counter. The processor executes an interrupt service routine (ISR) addressed in program counter.

There are two types of interrupts used in 8085 Microprocessor:

1. Hardware Interrupts
2. Software Interrupts

Software Interrupts

A software interrupts is a particular instructions that can be inserted into the desired location in the rpogram. There are eight Software interrupts in 8085 Microprocessor. From RST0 to RST7.

1. RST0
2. RST1
3. RST2
4. RST3
5. RST4
6. RST5
7. RST6
8. RST7

They allow the microprocessor to transfer program control from the main program to the subroutine program. After completing the subroutine program, the program control

returns back to the main program.

We can calculate the vector address of these interrupts using the formula given below:

```
Vector Address = Interrupt Number * 8
```

```
For Example:
RST2:         vector address=2*8     =  16
RST1:         vector address=1*8     =  08
RST3:         vector address=3*8     =  24
```

Vector address table for the software interrupts:

Interrupt	Vector Address
RST0	0000_H
RST1	0008_H
RST2	0010_H
RST3	0018_H
RST4	0020_H
RST5	0028_H
RST6	0030_H
RST7	0038_H

Hardware Interrupt

As it have already discussed that there are 6 interrupt pins in the microprocessor used as Hardware Interrrupts given below:

1. TRAP
2. RST7.5
3. RST6.5

4. RST5.5
5. INTR

Note:

INTA is not an interrupt. INTA is used by the Microprocessor for sending the acknowledgement. TRAP has highest priority and RST7.5 has second highest priority and so on.

The Vector address of these interrupts is given below:

Interrupt Vector Address

RST7.5	$003C_H$
RST6.5	0034_H
RST5.5	$002C_H$
TRAP	0024_H

TRAP

It is non maskable edge and level triggered interrupt. TRAP has the highest priority and vectores interrupt. Edge and level triggered means that the TRAP must go high and remain high until it is acknowledged. In case of sudden power failure, it executes a ISR and send the data from main memory to backup memory.
As we know that TRAP can not be masked but it can be delayed using HOLD signal. This interrupt transfers the microprocessor's control to location 0024H. TRAP interrupts can only be masked by resetting the microprocessor. There is no other way to mask it.

RST7.5

It has the second highest priority. It is maskable and edge level triggered interrupt. The vector address of this interrupt is 003CH. Edge sensitive means input goes high and no need to maintain high state until it is recognized.
It can also be reset or masked by resetting microprocessor. It can also be resettled by DI instruction.

RST6.5 and RST5.5

These are level triggered and maskable interrupts. When RST6.5 pin is at logic 1, INTE flip-flop is set. RST 6.5 has third highest priority and RST 5.5 has fourth highest priority. It can be masked by giving DI and SIM instructions or by resetting microprocessor .

INTR

It is level triggered and maskable interrupt. The following sequence of events occurs when INTR signal goes high:

1. The 8085 checks the status of INTR signal during execution of each instruction.
2. If INTR signal is high, then 8085 complete its current instruction and sends active low interrupt acknowledge signal, if the interrupt is enabled.
3. On receiving the instruction, the 8085 save the address of next instruction on stack and execute received instruction.

DATA TRANSFER SCHEMES

A wide variety of IO devices having wide range of speed and other different characteristics are available .A slow responding IO device cannot transfer data when microprocessor issues instruction for it as it takes some time to get Transfers rates¬ready. Of peripherals is usually slower ¬than the transfer rates of CPU. Operating modes of peripheral are different from each other and each must be controlled so as not to disturb the operation of each other peripherals connected to CPU.

Different types of data transfer techniques are available ¬which can be broadly divided into two categories:-

MICROPROCESSOR CONTROLLED :- HERE data transfer is controlled by microprocessor. Microprocessor is primarily responsible for data transfer whether from I/o to the CPU or to the memory or vice ¬versa.

DEVICE CONTROLLED:- Here data transfer is controlled by IO device . Data is transferred in between IO device and memory without the intervention of CPU such a transfer increases rate of transfer and makes the system more efficient

Here data transfer is controlled by microprocessor.

Microprocessor is primarily responsible for data transfer whether from I/o to the CPU or to the memory or vice versa.

Program data transfer scheme:

Program data transfer scheme is controlled by the CPU. Data are transferred from an IO device to the CPU or to the memory through CPU or vice versa under the control of programs which are stored in memory. These programs are executed by the CPU when an I/O device is ready to transfer }data. The program data transfer schemes are employed when small amount of data are to be transferred.

Here also synchronous and asynchronous mode of transfer is used.

Synchronous Data Transfer:-

Synchronous means 'at the same time'. The device Which sends data and the device which received data are synchronized with the same clock. When the CPU and IO devices match in speed, Synchronous Data Transfer technique is employed. The data transfer with IO devices is performed by executing IN and OUT instruction. The IN instruction is used to read data from an input device or input port. The OUT instruction is used to sends data from CPU to the output device or output port. As the CPU and the IO devices match in speed, the I/O device is ready to transfer data when IN or OUT instruction is executes. The status of the I/O device, whether it is ready or not, is not examined before the data is transferred.

Asynchronous mode of transfer:-

Asynchronous means 'at irregular intervals'. In this method data transfer is not based on predetermined timing pattern. This technique of data transfer is used when the speed of an I/O device does not match the speed of the microprocessor. In this technique the status of the I/O device i.e. whether the device is ready or not, is checked by the microprocessor before the data are transferred. The microprocessor initiates the I/O device to get ready and then continuously checks the status of I/O device till the I/O device becomes ready to transfer data. When I/O device becomes ready, the microprocessor executes instruction to transfer data

This mode of data transfer is also called handshaking mode of data transfer because some signals are exchanged between microprocessor and I/O devices before the actual data transfer takes place. Such signals are called handshake signals.

The microprocessor is too busy. CPU is wasting time while checking the flag instead of doing some useful work.

The problem with programmed I/O is that CPU has to wait along time for the I/O device to be ready for reception or transmission of data .The CPU while waiting, must repeatedly interrogate the status of the I/O device. As a result the level of the performance of the entire }system is severely degraded. An alternative is

Interrupt driven IO data transfer:

In this scheme when the I/O device becomes ready to transfer data, it sends a high signal to the microprocessor through a special input line called an interrupt line. In other words it interrupts the normal processing sequence of the microprocessor. On receiving interrupt the microprocessor completes the current instruction, saves the contents of the program counter on stack first and then attends the I/O devices. It takes up a subroutine called ISS (Interrupt Service Subroutine). It executes ISS to transfer data from or to the I/O device. Different ISS are to be provided for different IO devices. After completing the data transfer the microprocessor returns back to the main program which it was executing before the interrupt occurred.

The¬ normal operation of the microprocessor is It need to continue monitoring for interrupt¬interrupted Signals.

DMA Data Transfer:

The transfer of data between the mass storage device and a system memory is often limited by the speed of microprocessor. Removing the microprocessor during such a transfer and letting the peripheral manage the transfer to or from memory would improve the speed of transfer and hence will make the system more efficient. This transfer technique is called DMA Data Transfer.

During DMA transfer microprocessor is idle, so it has no longer control on the system buses. A DMA Controller takes over the buses and manage the transfer directly between the peripheral and the memory It is fastest scheme then Programmed Data Transfer Scheme and the microprocessor regains the control of buses after data transfer

PROGRAMMABLE PERIPHERAL INTERFACE

(8255)

The 8255 is a widely used, programmable, parallel I/O device.

• It can be programmed to transfer data under various conditions, from simple I/O to interrupt I/O.
• It is flexible, versatile and economical and complex.

The 8255 has 24 I/O pins that can be grouped primarily into two 8 bit parallel ports: A and B, with the remaining 8 bits a port C. The 8 bits of port C can be used as individual bits or be grouped in two 4-bit ports: CUPPER (CU) and CLOWER (CL), as shown in the figure 1.1. The functions of these ports are defined by writing a control word in the control register.

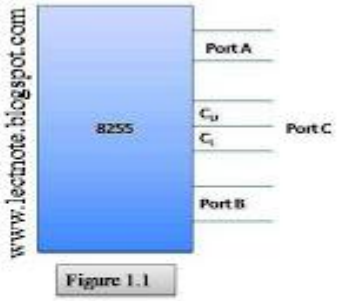


Figure 1.1

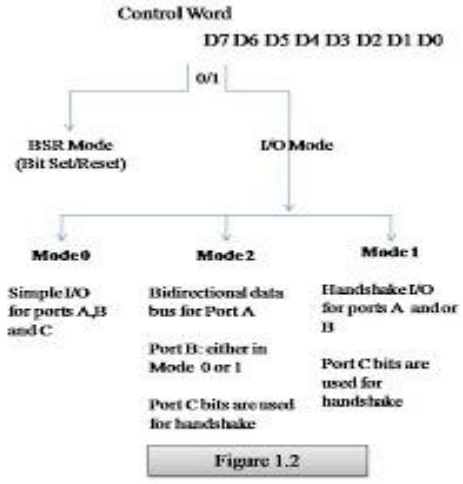
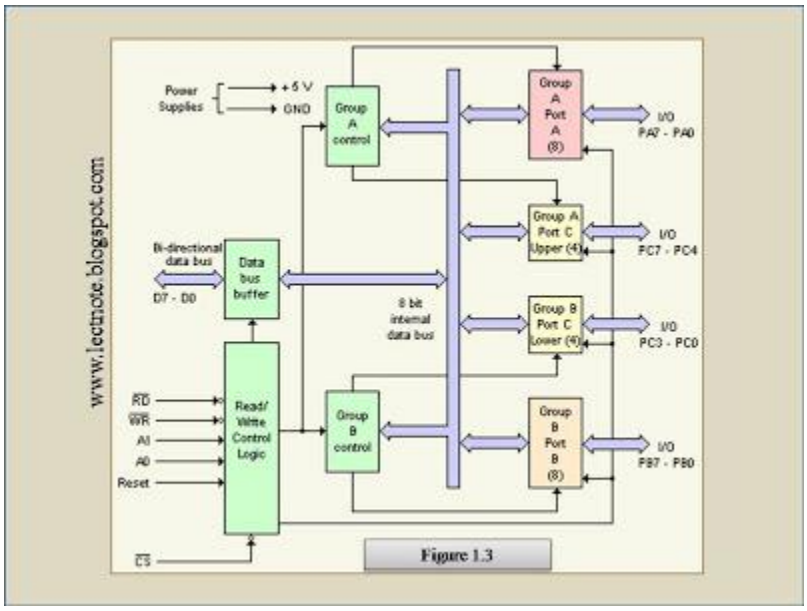


Figure 1.2

Figure 1.2 shows all the functions of 8255; classified according to two modes: the Bit Set/Reset (BSR) mode and I/O mode. The BSR mode is used to set or reset the bits in port C. The I/O mode is further divided into three modes: Mode 0, Mode 1 and Mode 2. In Mode 0, all ports function as simple I/O ports. Mode 1 is a hand shake mode whereby

Ports A and/or B use bits from port C as handshake signals. In Mode 2 Port A can be set up for bidirectional data transfer using handshaking signals from Port C, and Port B can be set up either in Mode 0 or Mode 1.

Block Diagram of the 8255



CONTROL LOGIC

($\overline{RD}$)

(Read): This control signal enables the Read operation. When the signal is low, the MPU reads data fro a selected I/O Port of the 8255.

($\overline{WR}$)

(Write): This control signal enables the write operation. When the signal goes low, MPU writes into a selected I/O Port or control register.

RESET (Reset): This is an active high signal; it clears the control register and sets all ports in the input mode.

($\overline{CS}$)

, A0 and A1: Theses are device select signals. Chip Select is connected to a decoded address, and A0 and A1 are generally connected to MPU address lines A0 and A1 respectively

($\overline{CS}$)	A_1	A_0	Selected
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	Control Register
1	X	X	Not Selected

CONTROL WORD

Figure 1.5 shows a register called the control register. The contents of this register called control word. This register can be accessed to write a control word when A0 and A1 are at logic 1. This control register is not accessible for a read operation.

Bit D7 of the control register specifies either **I/O** function or the **Bit Set/Reset** function. If bit D7=1, bits D6-D0 determines I/O functions in various modes. If bit D7=0, Port C operates in the Bit Set/Reset (BSR) mode. The BSR control word does not affect the functions of Port A and Port B.

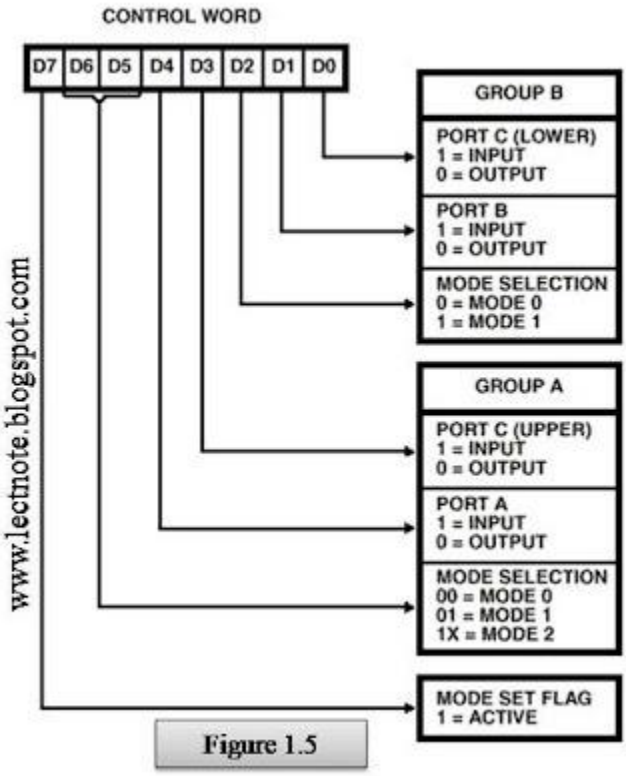


Figure 1.5

To communicate with peripherals through the 8255, three steps are necessary:

1. Determine the address of ports A, B and C and of the control register according to the

chip select logic and address lines A0 andA1.

2. Write the control word in the control register.

3. Write I/O instructions to communicate with peripherals through Ports A, B and C.

Operating Modes

Mode 0: Simple Input or Output

In this mode, ports A, B are used as two simple 8-bit I/O ports and port C as two 4-bit ports. Each port can be programmed to function as simply an input port or an output port. The input/output features in Mode 0 are as follows.

1. Outputs are latched.

2. Inputs are not latched.

3. Ports don't have handshake or interrupt capability.

Mode 1: Input or Output with Handshake

In this mode, handshake signals are exchanged between the MPU and peripherals prior to data transfer. The features of the mode include the following:

1. Two ports (A and B) function as 8-bit I/O ports. They can be configured as either as input or output ports.

2. Each port uses three lines from ort C as handshake signals. The remaining two lines of Port C can be used for simple I/O operations.

3. Input and Output data are latched.

4. Interrupt logic is supported.

Mode 2: Bidirectional Data Transfer

This mode is used primarily in applications such as data transfer between two computers. In this mode, Port A can be configured as the bidirectional port and Port B either in Mode 0 or Mode 1. Port A uses five signals from Port C as handshake signals for data transfer. The remaining three signals from port C can be used either as simple I/O or as handshake for port B.

BSR (Bit Set/Reset) Mode

The BSR mode is concerned only with the eight bits of port C, which can be set or reset by writing an appropriate control word in the control register. A control word with bit D7 =0 is recognized as a BSR control word, and it does not alter any previously transmitted control word with bit D7=1; thus the I/O operations of ports A and B are not affected by a BSR control word. In BSR mode, individual bits of port C can be used for applications such as an on/off switch. Ports A and B are not affected by the BSR Mode.

What s Microprocessor? Give the power supply & clock frequency of 8085?

Ans: A microprocessor is amultipurpose, programmable logic device that reads binary instructions from a storage device called memory accepts binary data as input and processes data according to those instructions and provides result as output. The power supply of 8085 is +5V and clock frequency in 3MHz.

2. List few applications of microprocessor-based system.

Ans:

It is used:

i. For measurements, display

and control of current, voltage, temperature, pressure, etc.

ii. For traffic control and industrial tool control.

iii. For speed control of machines.

3. What are the functions of an accumulator?

Ans: The accumulator is the register associated with the ALU operations and sometimes I/O operations. It is an integral part of ALU. It holds one of data to be processed by ALU. It also temporarily stores the result of the operation performed by the ALU.

4. List the 16 – bit registers of 8085 microprocessor.

Ans: Stack pointer (SP) and Program counter (PC).

5. List the allowed register pairs of8085.

Ans:

B-C register pair

D-E register pair

H-L register pair

6. Mention the purpose of SID and SOD lines

Ans: SID (Serial input data line): It is an input line through which the microprocessor accepts serial data. SOD (Serial output data line): It is an output line through which the microprocessor sends output serial data.

7. What is an Opcode?

Ans: The part of the instruction that specifies the operation to be performed is called the operation code or opcode.

8. What is the function of IO/M signal in the 8085?

Ans: It is a status signal. It is used to differentiate between memory locations and I/O operations. When this signal is low (IO/M = 0) it denotes the memory related operations. When this signal is high (IO/M = 1) it denotes an I/O operation.

9. What is an Operand?

Ans: The data on which the operation is to be performed is called as an Operand.

10. How many operations are there in the instruction set of 8085 microprocessor?

Ans: There are 74 operations in the 8085 microprocessor.

11. List out the five categories of the 8085 instructions. Give examples of the instructions for each group.

Ans:

- Data transfer group – MOV, MVI, LXI.
- Arithmetic group – ADD, SUB, INR.
- Logical group –ANA, XRA, CMP.
- Branch group – JMP, JNZ, CALL.
- Stack I/O and Machine control group – PUSH, POP, IN, HLT.

12. Explain the difference between a JMP instruction and CALL instruction.

Ans: A JMP instruction permanently changes the program counter. A CALL instruction leaves information on the stack so that the original program execution sequence can be resumed.

13. Explain the purpose of the I/O instructions IN and OUT.

Ans: The IN instruction is used to move data from an I/O port into the accumulator. The OUT instruction is used to move data from the accumulator to an I/O port. The IN & OUT instructions are used only on microprocessor, which use a separate address space for interfacing.

14. What is the difference between the shift and rotate instructions?

Ans: A rotate instruction is a closed loop instruction. That is, the data moved out at one end is put back in at the other end. The shift instruction loses the data that is moved out of the last bit locations.

15. How many address lines in a 4096 x 8 EPROM CHIP?

Ans: 12 address lines.

16. What are the Control signals used for DMA operation?

Ans:-HOLD & HLDA.

17. What is meant by Wait State?

Ans:-This state is used by slow peripheral devices. The peripheral devices can transfer the data to or from the microprocessor by using READY input line. The microprocessor remains in wait state as long as READY line is low. During the wait state, the contents of the address, address/data and control buses are held constant.

18. List the four instructions which control the interrupt structure of the 8085 microprocessor.

Ans:-

- DI (Disable Interrupts)
- EI (Enable Interrupts)
- RIM (Read Interrupt Masks)
- SIM (Set Interrupt Masks)

19. What is meant by polling?

Ans:-Polling or device polling is a process which identifies the device that has interrupted the microprocessor.

20. What is meant by interrupt?

Ans:-Interrupt is an external signal that causes a microprocessor to jump to a specific subroutine.

21. Explain priority interrupts of 8085.

Ans:-The 8085 microprocessor has five interrupt inputs. They are TRAP, RST 7.5, RST 6.5, RST 5.5, and INTR. These interrupts have a fixed priority of interrupt service. If two or more interrupts go high at the same time, the 8085 will service them on priority basis. The TRAP has the highest priority followed by RST 7.5, RST 6.5, RST 5.5. The priority of interrupts in 8085 is shown in the table.

TRAP	**1**
RST 7.5	**2**
RST 6.5	**3**
RST 5.5	**4**
INTR	**5**

22. What is a microcomputer?

Ans:-A computer that is designed using a microprocessor as its CPU is called microcomputer.

23. What is the signal classification of 8085

Ans:-All the signals of 8085 can be classified into 6 groups

- Address bus
- Data bus
- Control and status signals
- Power supply and frequency signals
- Externally initiated signals
- Serial I/O ports

24. What are operations performed on data in 8085

Ans:- The various operations performed are

- Store 8-bit data
- Perform arithmetic and logical operations
- Test for conditions
- Sequence the execution of instructions
- Store data temporarily during execution in the defined R/W memory locations called the stack

25. Steps involved to fetch a byte in 8085

Ans:-

i. The PC places the 16-bit memory address on the address bus ii. The control unit sends the control signal RD to enable the memory chip iii. The byte from the memory location is placed on the data bus iv. The byte is placed in the instruction decoder of the microprocessor and the task is carried out according to the instruction

26. How many interrupts does 8085 have, mention them

Ans:-The 8085 has 5 interrupt signals; they are INTR, RST7.5, RST6.5, RST5.5 and TRAP

27. Basic concepts in memory interfacing

Ans:-The primary function of memory interfacing is that the microprocessor should be able to read from and write into a given register of a memory chip. To perform these operations the microprocessor should

- Be able to select the chip
- Identify the register
- Enable the appropriate buffer

28. Define instruction cycle, machine cycle and T-state

Ans:-Instruction cycle is defined, as the time required completing the execution of an instruction. Machine cycle is defined as the time required completing one operation of accessing memory, I/O or acknowledging an external request. Tcycle is defined as one subdivision of the operation performed in one clock period

29. What is an instruction?

Ans:-An instruction is a binary pattern entered through an input device to command the microprocessor to perform that specific function

30. What is the use of ALE?

Ans:-The ALE is used to latch the lower order address so that it can be available in T2 and T3 and used for identifying the memory address. During T1 the ALE goes high, the latch is transparent ie, the output changes according to the input data, so the output of the latch is the lower order address. When ALE goes low the lower order address is latched until the next ALE.

31. How many machine cycles does 8085 have, mention them

Ans: The 8085 have seven machine cycles. They are

- Opcode fetch
- Memory read
- Memory write
- I/O read
- I/O write
- Interrupt acknowledge
- Bus idle

32. Explain the signals HOLD, READY and SID

Ans:HOLD indicates that a peripheral such as DMA controller is requesting the use of address bus, data bus and control bus. READY is used to delay the microprocessor read or write cycles until a slow responding peripheral is ready to send or accept data.SID is used to accept serial data bit by bit

33. Mention the categories of instruction and give two examples for each category.

Ans: The instructions of 8085 can be categorized into the following five categories

- Data transfer Instructions -MOV Rd,Rs STA 16-bit
- Arithmetic
  Instructions -ADD R DCR M
- Logical
  Instructions -XRI 8-bit RAR
- Branching
  Instructions -JNZ CALL 16-bit
- Machine control Instructions -HLT NOP

34. Explain LDA, STA and DAA instructions

Ans:LDA copies the data byte into accumulator from the memory location specified by the 16-bit address. STA copies the data byte from the accumulator in the memory location specified by 16-bit address. DAA changes the contents of the accumulator from binary to 4-bit BCD digits.

35. Explain the different instruction formats with examples

Ans: The instruction set is grouped into the following formats

- One byte instruction -MOV C,A
- Two byte instruction -MVI A,39H
- Three byte instruction -JMP 2345H

36. What is the use of addressing modes, mention the different types

Ans: The various formats of specifying the operands are called addressing modes, it is used to access the operands or data. The different types are as follows

- Immediate addressing
- Register addressing
- Direct addressing
- Indirect addressing
- Implicit addressing

37. What is the use of bi-directional buffers?

Ans: It is used to increase the driving capacity of the data bus. The data bus of a microcomputer system is bi-directional, so it requires a buffer that allows the data to flow in both directions.

38. Give the register organization of 8085

Ans:

W(8)	**Temp. Reg**
Z(8)	**Temp. Reg**
B(8)	**Register**
C(8)	**Register**
D(8)	**Register**
E(8)	**Register**
H(8)	**Register**
L(8)	**Register**
Stack Pointer	**(16)**
Program Counter	**(16)**

39. Define stack and explain stack related instructions

Ans: The stack is a group of memory locations in the R/W memory that is used for the temporary storage of binary information during the execution of the program. The stack related instructions are PUSH & POP

40. Why do we use XRA A instruction

Ans: The XRA A instruction is used to clear the contents of the Accumulator and store the value 00H.

41. Compare CALL and PUSH instructions

Ans:

CALL	**PUSH**
1.When CALL is executed the microprocessor automatically stores the 16-bit address of the instruction next to CALL on the stack.	**1.PUSH The programmer uses the instruction to save the contents of the register pair on the stack**
2.When CALL is executed the stack pointer is decremented by two	**2. When PUSH is executed the stack pointer is decremented by two**

42. What is Microcontroller and Microcomputer

Ans: Microcontroller is a device that includes microprocessor; memory and I/O signal lines on a single chip, fabricated using VLSI technology. Microcomputer is a computer

that is designed using microprocessor as its CPU. It includes microprocessor, memory and I/O.

43. Define Flags

Ans: The flags are used to reflect the data conditions in the accumulator. The 8085 flags are S-Sign flag, Z-Zero flag, AC-Auxiliary carry flag, P-Parity flag, CYCarry flag, D7 D6 D5 D4 D3 D2 D1 D0 **44. How does the microprocessor differentiate between data and instruction?**

Ans: When the first m/c code of an instruction is fetched and decoded in the instruction register, the microprocessor recognizes the number of bytes required to fetch the entire instruction. For example MVI A, Data, the second byte is always considered as data. If the data byte is omitted by mistake whatever is in that memory location will be considered as data & the byte after the "data" will be treated as the next instruction.

45. Compare RET and POP

Ans:

RET	**POP**
1.RET transfers the contents of the top two locations of the stack to the PC	1.POP transfers the contents of the top two locations of the stack to the specified register pair
2.When RET is executed the SP is incremented by two	2. When POP is executed the SP is incremented by two
3.Has 8 conditional RETURN instructions	3.No conditional POP instructions

46. What is assembler?

Ans: The assembler translates the assembly language program text which is given as input to the assembler to their binary equivalents known as object code. The time required to

translate the assembly code to object code is called access time. The assembler checks for syntax errors & displays them before giving the object code.

47. What is loader?

Ans: The loader copies the program into the computer's main memory at load time and begins the program execution at execution time.

48. What is linker?

Ans:A linker is a program used to join together several object files into one large object file. For large programs it is more efficient to divide the large program modules into smaller modules. Each module is individually written, tested & debugged. When all the modules work they are linked together to form a large functioning program.

49. What is interrupt service routine?

Ans: Interrupt means to break the sequence of operation. While the CPU is executing a program an interrupt breaks the normal sequence of execution of instructions & diverts its execution to some other program. This program to which the control is transferred is called the interrupt service routine. **50.What are the various programmed data transfer methods?**

Ans: i) Synchronous data transfer ii) Asynchronous data transfer iii) Interrupt driven data transfer

51. What is synchronous data transfer?

Ans: It is a data method which is used when the I/O device and the microprocessor match in speed. To transfer a data to or from the device, the user program issues a suitable instruction addressing the device. The data transfer is completed at the end of the execution of this instruction.

52. What is asynchronous data transfer?

Ans: It is a data transfer method which is used when the speed of an I/O device does not match with the speed of the microprocessor. Asynchronous data transfer is also called as Handshaking.

53. What are the basic modes of operation of 8255?

Ans: There are two basic modes of operation of 8255, They are:

1. I/O mode. 2. BSR mode.

In I/O mode, the 8255 ports work as programmable I/O ports, while In BSR mode only port C (PC0-PC7) can be used to set or reset its individual port bits. Under the IO mode of operation, further there are three modes of operation of 8255, So as to support different types of applications, viz. mode 0, mode 1 and mode 2.

- Mode 0 - Basic I/O mode
- Mode 1 - Strobed I/O mode
- Mode 2 - Strobed bi-directional I/O.

54. Write the features of mode 0 in 8255?

Ans:1. Two 8-bit ports (port A and port B) and two 4-bit ports (port C upper and lower) are available. The two 4-bit ports can be combined used as a third 8-bit port. 2. Any port can be used as an input or output port. 3.Output ports are latched. Input ports are not latched. 4. A maximum of four ports are available so that overall 16 I/O configurations are possible.

55. What are the features used mode 1 in 8255?

Ans: Two groups – group A and group B are available for strobed data transfer.

1. Each group contains one 8-bit data I/O port and one 4-bit control/data port. 2. The 8-bit data port can be either used as input or output port. The inputs and outputs both are latched. 3. Out of 8-bit port C, PC0-PC2 is used to generate control signals for port B and PC3=PC5 are used to generate control signals for port A. The lines PC6, PC7 may be used as independent data lines.

56. What are the signals used in input control signal & output control signal?

Ans: Input control signal

- STB (Strobe input)
- IBF (Input buffer full)
- INTR(Interrupt request)

Output control signal

- OBF (Output buffer full)
- ACK (Acknowledge input)

- INTR(Interrupt request)

57. What are the features used mode 2 in 8255?

Ans: The single 8-bit port in-group A is available.

1. The 8-bit port is bi-directional and additionally a 5-bit control port is available. 2. Three I/O lines are available at port C, viz PC2-PC0. 3. Inputs and outputs are both latched. 4. The 5-bit control port C (PC3=PC7) is used for generating/accepting handshake signals for the 8-bit data transfer on port A.

58. What are the modes of operations used in 8253?

Ans: Each of the three counters of 8253 can be operated in one of the following six modes of operation.

1. Mode 0 (Interrupt on terminal count) 2. Mode 1 (Programmable monoshot) 3. Mode 2 (Rate generator) 4. Mode 3 (Square wave generator) 5. Mode 4 (Software triggered strobe) 6. Mode 5 (Hardware triggered strobe)

59. What are the different types of write operations used in 8253?

Ans: There are two types of write operations in 8253

(1) Writing a control word register (2) Writing a count value into a count register

The control word register accepts data from the data buffer and initializes the counters, as required. The control word register contents are used for

(a) Initializing the operating modes (mode 0-mode4) (b) Selection of counters (counter 0-counter 2) (c) Choosing binary /BCD counters (d) Loading of the counter registers.

The mode control register is a write only register and the CPU cannot read its contents.

60. What is the function of mode set register in 8257?

Ans: The mode set register is used for programming the 8257 as per the requirements of the system. The function of the mode set register is to enable the DMA channels individually and also to set the various modes of operation.

61. What is interfacing?

Ans: An interface is a shared boundary between the devices which involves sharing information. Interfacing is the process of making two different systems communicate with each other.

62. List the operation modes of 8255

Ans: a) I.O Mode

i. Mode 0-Simple Input/Output. ii. Mode 1-Strobed Input/Output (Handshake mode) iii. Mode 2-Strobed bidirectional mode

b) Bit Set/Reset Mode.

63. What is a control word?

Ans: It is a word stored in a register (control register) used to control the operation of a program digital device.

64. What is the purpose of control word written to control register in 8255?

Ans: The control words written to control register specify an I/O function for each I.O port. The bit D7 of the control word determines either the I/O function of the BSR function.

65. What is the size of ports in 8255?

Ans:

Port-A : 8-bits

Port-B : 8-bits

Port-CU : 4-bits

Port-CL : 4-bits

66. Distinguish between the memories mapped I/O peripheral I/O?

Ans:

Memory Mapped I/O	Peripheral MappedI/O
16-bit device address	8-bit device address
Data transfer between any general-purpose register and I/O port.	Data is transfer only between accumulator and I.O port
The memory map (64K) is shared between I/O device and system memory.	The I/O map is independent of the memory map; 256 input device and 256 output device can be connected
More hardware is required to	Less hardware is required to

decode 16-bit address	decode 8-bit address
Arithmetic or logic operation can be directly performed with I/O data	Arithmetic or logical operation cannot be directly performed with I/O data

67. What is memory mapping?

Ans: The assignment of memory addresses to various registers in a memory chip is called as memory mapping.

68. What is I/O mapping?

Ans: The assignment of addresses to various I/O devices in the memory chip is called as I/O mapping.