Microprocessor ,Assembly Language & Computer Interfacing Sessional

EEE-3212

Fareha Akter Juha
Adjunct Faculty, Dep. of CSE,
Leading University, Sylhet
Email: farehaakter02@gmail.com

Lab-1: Fundamentals of microprocessor & Introduce to 8085 emulators.

What is a Microprocessor?

The microprocessor is the central unit of a computer system that performs arithmetic and logic operations, which generally include adding, subtracting, transferring numbers from one area to another, and comparing two numbers. It's often known simply as a processor, a central processing unit, or a logic chip.

How Does a Microprocessor Work?

A microprocessor accepts binary data as input, processes that data, and then provides output based on the instructions stored in the memory. The data is processed using the microprocessor's ALU (arithmetical and logical unit), control unit, and a register array. The register array processes the data via a number of registers that act as temporary fast access memory locations. The flow of instructions and data through the system is managed by the control unit.

farehaakter02@gmail.com

8085 Microprocessor

8085 is pronounced as an "eighty-eighty-five" microprocessor. It is an 8-bit microprocessor designed by Intel in 1977 using NMOS technology.

It has the following configuration −

8-bit data bus
16-bit address bus, which can address up to 64KB
A 16-bit program counter
A 16-bit stack pointer
Six 8-bit registers arranged in pairs: BC, DE, HL
Requires +5V supply to operate at 3.2 MHZ single phase clock
It is used in washing machines, microwave ovens, mobile phones, etc.

farehaakter02@gmail.com

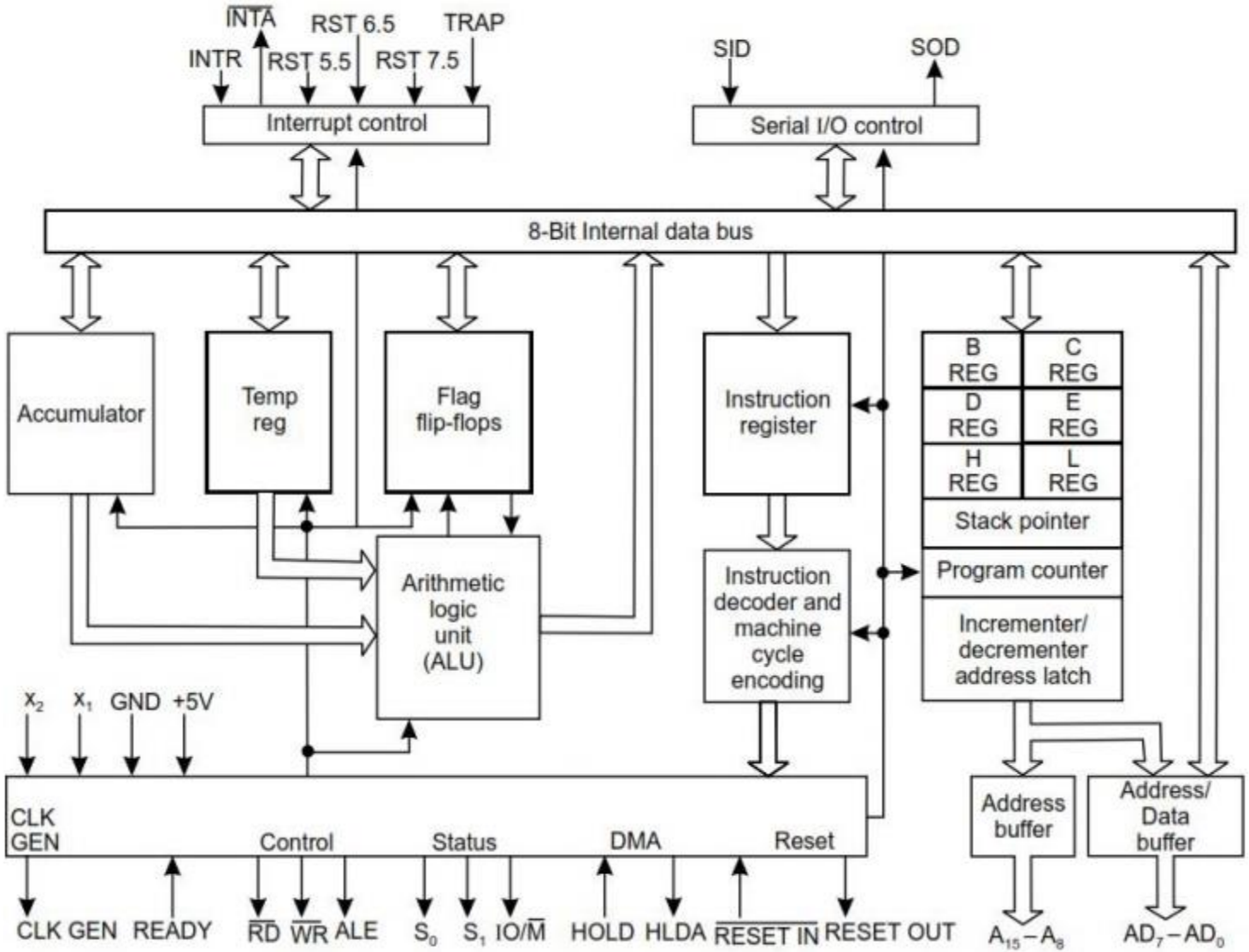


Fig 1.1 Hardware Architecture of 8085

FAREHAAKTER02@GMAIL.COM

Registers in 8085:

(a) General Purpose Registers –

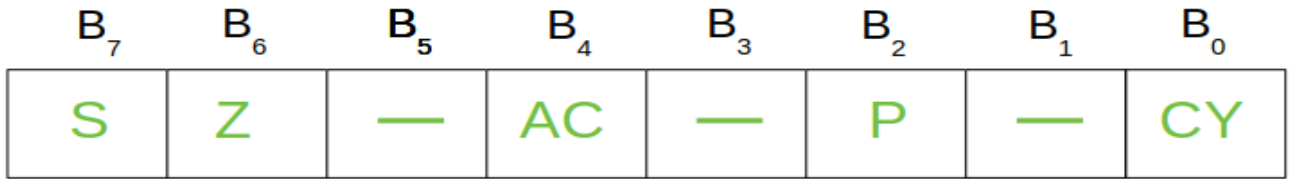
The 8085 has six general-purpose registers to store 8-bit data; these are identified as B, C, D, E, H, and L. These can be combined as register pairs – BC, DE, and HL, to perform some 16-bit operations. These registers are used to store or copy temporary data, by using instructions, during the execution of the program.

(b) Specific Purpose Registers –

Accumulator:

The accumulator is an 8-bit register (which can store 8-bit data) that is part of the arithmetic and logical unit (ALU). After performing arithmetical or logical operations, the result is stored in the accumulator. An accumulator is also defined as register A.

Flag registers:



fig(a)-Bit position of various flags in flag registers of 8085

farehaakter02@gmail.com

The flag register is a special purpose register and it is completely different from other registers in microprocessors. It consists of 8 bits and only 5 of them are useful. The other three are left vacant and are used in future Intel versions. These 5 flags are set or reset (when the value of the flag is 1, then it is said to be set and when the value is 0, then it is said to be reset) after an operation according to the data condition of the result in the accumulator and other registers. The 5 flag registers are:

Sign Flag: It occupies the seventh bit of the flag register, which is also known as the most significant bit. It helps the programmer to know whether the number stored in the accumulator is positive or negative. If the sign flag is set, it means that the number stored in the accumulator is negative, and if reset, then the number is positive.

Zero Flag: It occupies the sixth bit of the flag register. It is set, when the operation performed in the ALU results in zero(all 8 bits are zero), otherwise it is reset. It helps in determining if two numbers are equal or not.

Auxiliary Carry Flag: It occupies the fourth bit of the flag register. In an arithmetic operation, when a carry flag is generated by the third bit and passed on to the fourth bit, then the Auxiliary Carry flag is set. If not flag is reset. This flag is used internally for BCD(Binary-Coded decimal Number) operations.

Note – This is the only flag register in 8085 which is not accessible by the user.

Parity Flag: It occupies the second bit of the flag register. This flag tests for a number of 1's in the accumulator. If the accumulator holds an even number of 1's, then this flag is set and it is said to even parity. On the other hand, if the number of 1's is odd, then it is reset and it is said to be odd parity.
Carry Flag: It occupies the zeroth bit of the flag register. If the arithmetic operation results in a carry(if the result is more than 8 bit), then Carry Flag is set; otherwise, it is reset.

(c) Memory Registers –

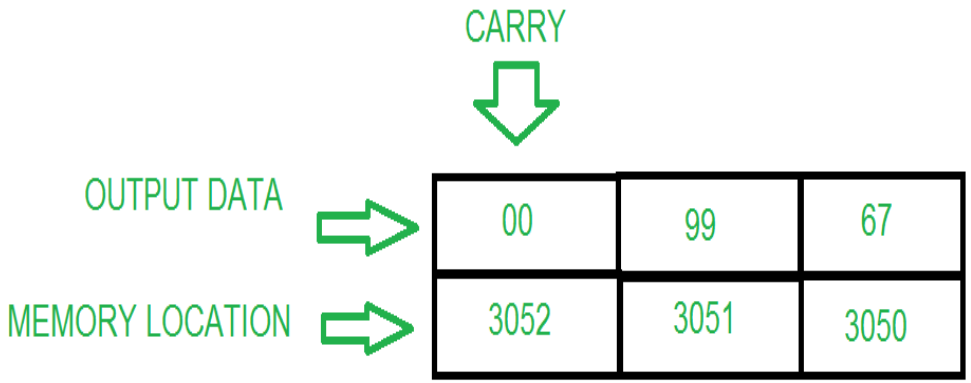
There are two 16-bit registers used to hold memory addresses. The size of these registers is 16 bits because the memory addresses are 16 bits. They are:-

Program Counter: This register is used to sequence the execution of the instructions. The function of the program counter is to point to the memory address from which the next byte is to be fetched. When a byte (machine code) is being fetched, the program counter is incremented by one to point to the next memory location.
Stack Pointer: It is used as a memory pointer. It points to a memory location in reading/write memory, called the stack. It is always incremented/decremented by 2 during push and pop operation.

8085 program to add two 16 bit numbers

Problem: Write an assembly language program to add two 16 bit numbers by using:

8-bit operation
16-bit operation
Example:



Program:

MEMORY ADDRESS	MNEMONICS	COMMENTS
2000	LDA 2050	A ← 2050
2003	MOV B, A	B ← A
2004	LDA 2052	A ← 2052
2007	ADD B	A ← A+B
2008	STA 3050	A → 3050
200B	LDA 2051	A ← 2051
200E	MOV B, A	B ← A
200F	LDA 2053	A ← 2053
2012	ADC B	A ← A+B+CY
2013	STA 3051	A → 3051
2016	HLT	Stops execution

FAREHAAKTER02@GMAIL.COM

1. Addition of 16-bit numbers using 8-bit operation:

It is a lengthy method and requires more memory as compared to the 16-bit operation.

Algorithm:

1. Load the lower part of the first number in the B register.
2. Load the lower part of the second number in A (accumulator).
3. Add both the numbers and store.
4. Load the higher part of the first number in the B register.
5. Load the higher part of the second number in A (accumulator).
6. Add both the numbers with carrying from the lower bytes (if any) and store them at the next location.

FAREHAAKTER02@GMAIL.COM

Explanation:

1. LDA 2050 stores the value at 2050 in A (accumulator).
2. MOV B, A stores the value of A into the B register.
3. LDA 2052 stores the value at 2052 in A.
4. ADD B add the contents of B and A and store them in A.
5. STA 3050 stores the result in memory location 3050.
6. LDA 2051 stores the value at 2051 in A.
7. MOV B, A stores the value of A into the B register.
8. LDA 2053 stores the value at 2053 in A.
9. ADC B adds the contents of B, A, and carry from the lower bit addition and store in A.
10. STA 3051 stores the result in memory location 3051.
11. HLT stops execution.

FAREHAAKTER02@GMAIL.COM

2. Addition of 16 bit numbers using 16-bit operation:

It is a very short method and less memory is also required as compared to 8-bit operations.

Algorithm:

1. Load both the lower and the higher bits of the first number at once.
2. Copy the first number to another registered pair.
3. Load both the lower and the higher bits of second number at once.
4. Add both the register pairs and store the result in a memory location.

MEMORY ADDRESS	MNEMONICS	COMMENTS
2000	LHLD 2050	H-L ← 2050
2003	XCHG	D H & E L
2004	LHLD 2052	H-L ← 2052
2007	DAD D	H ← H+D & L ← L+E
2008	SHLD 3050	L → 3050 & H → 3051
200B	HLT	Stops execution

FAREHAAKTER02@GMAIL.COM

Explanation:

1.LHLD 2050 loads the value at 2050 in L register and that in 2051 in the H register (first number)

2.XCHG copies the content of the H to D register and L to E register

3.LHLD 2052 loads the value at 2052 in L register and that in 2053 in the H register (second number)

4.DAD D adds the value of H with D and L with E and stores the result in H and L

5.SHLD 3050 stores the result at memory location 3050

6.HLT stops execution

FAREHAAKTER02@GMAIL.COM

Emulators

Download 8085 emulators from this link:

GNUSim8085 (https://www.gnusim8085.org/)

GNUSim8085 is a graphical cross-platform simulator plus assembler with a debugger for the Intel 8085 microprocessor. You have to enter the assembly code in order to use the simulator.

FAREHAAKTER02@GMAIL.COM

Thank You

Farehaakter02@gmail.Com