CSE331-Lab: Introduction to assembly language.

Introduction:

In this session, you will be introduced to assembly language programming and to the emu 8086 emu-lator software. Emu 8086 will be used as both an editor and as an assembler for all your assembly language. progamming.

Steps required to run an assembly program:

1. Write the necessary assembly source code.
2. Save the assembly source code.
3. Compile / Assemble source code to create machine code.
4. Emulate / Run the machine code.

First, familiarize yourself with the software before in-class instructions regarding the layout of emu 8086.

Micro controllers vs micro processors

* A microprocessors is a CPU on a singl chip.

* If a microprocessors, Its associated support circuitry, memory and peripheral I/O components are implemented on a a single chip. It is microcontroller.

Features of 8086

* 8086 is a 16 bit processors. Its ALU registers work with 16 bit binary word

* 8086 has a 16 bit data bus. It can read or write data to a memory / Port eithers 16 bits or 8 bits at a time.

* 8086 has a 20 bits address bus which means, it can address upto $2^{20} = 1$MB memory location.

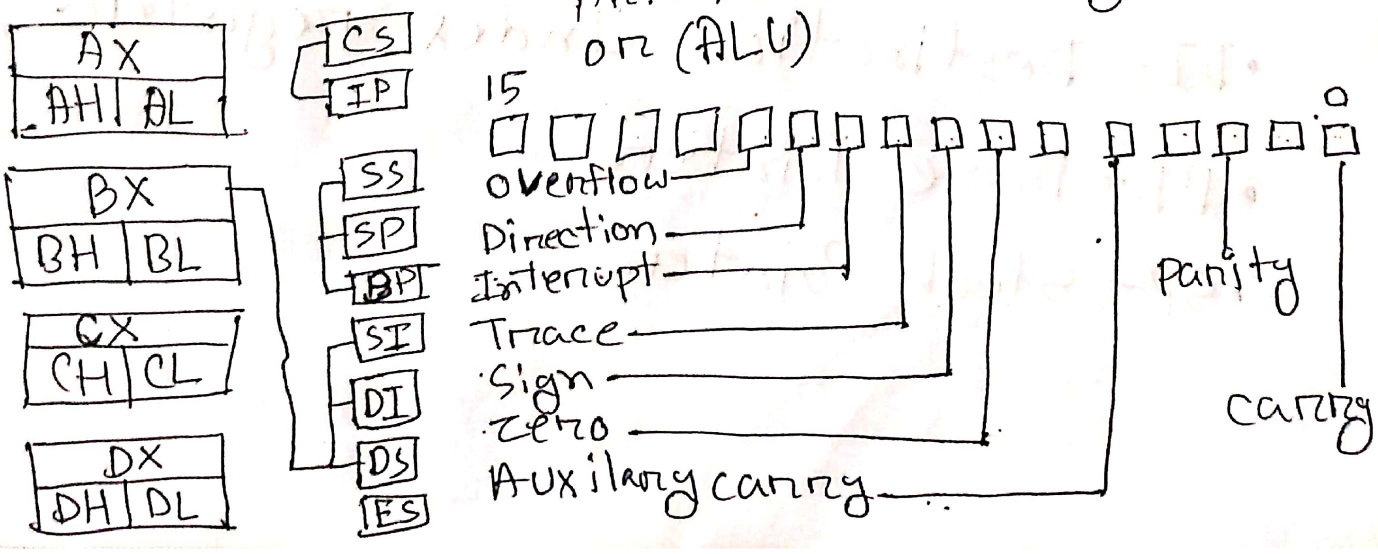
⊡Register – Registor – Registrar.

⊛ Both ALU and FPU have a very small amount of super-fast private memory placed right next to them for their exclusive use. These are called registers.

⊛ ALU and FPU stores intermediate and final results from their calculations in these registers.

⊛ Processed data goes back to the data cache then to the main memory from these registers.

⊡ Inside the CPU: Get to know the various registers central processing unit (CPU) Arithmetic and logical unit or (ALU)



Registers are basically cpu's own internal memory. They are used among other purpose to store temporary data while performing calculations. Let's look at each one in details.

⊕ General purpose Register (GPR):

The 8086 CPU has 8 general-purpose registers; each register has its own name:

- AX - The Accumulator register (divided int AH/AL)
- BX - The Base Address register (divided into BH/BL).
- CX - The Count register (divided into CH/CL).
- DX - The Data register (divided int DH/DL).
- SI - Source Index register
- DI - Destination Index register
- BP - Base Pointer
- SP - Stack Pointer.

Segment Registers

CS- Points at the segment containing the current program.

DS- generally, points at the segment where variables are defined.

ES- extra segment register, it's up to a coder to define its usage.

SS- Points at the segment containing the stack.

Although its possible to store any data in the segment registers, this is never a good idea. The segment registers have a very special purpose - pointing at accessible blocks of memory. This will be discussed further in upcoming classes.

⊞ Special ~~Pu~~ Purposes Registers

- IP - The Instruction pointer. Points to the next location of instruction in the memory.
- Flags Register - Determines the current state of the microprocessor. Modified automatically by the CPU after some mathematical operations, determines certain types of results and determine how to transfer control of a program.

⊞ Writing Your First Assembly Code.

In order to write programs in assembly language, you will need to familiarize yourself with most, if not all, of the instructions in the 8086 - instruction set.

- REG: Any valid register
- Memory: Referring to a memory location in RAM
- Immediate: Using direct values.

Instruction	Operands	Description
MOV	REG, memory memory, REG REG, REG memory, immediate REG, immediate	Copy Operand2 to Operand1 • The MOV instruction cannot set the value of the CS and IP reg. • Copy value of one segment register to another segment register. • Copy an immediate value to segment register. Algorith: Operand1 = Operand2
ADD	REG, memory memory, REG REG, REG memory, immediate REG, immediate	Adds two numbers. Algorithm: Operand1 = Operand 1 + operand2