Introduction

Levels of Programming Languages

1) Machine Language
   - Consists of individual instructions that will be executed by the CPU one at a time
2) Assembly Language (Low Level Language)
   - Designed for a specific family of processors (different processor groups/family has different Assembly Language)
   - Consists of symbolic instructions directly related to machine language instructions one-for-one and are assembled into machine language.

3) High Level Languages

- e.g. : C, C++ and Vbasic
- Designed to eliminate the technicalities of a particular computer.
- Statements compiled in a high level language typically generate many low-level instructions.

Advantages of Assembly Language

1. Shows how program interfaces with the processor, operating system, and BIOS.
2. Shows how data is represented and stored in memory and on external devices.
3. Clarifies how processor accesses and executes instructions and how instructions access and process data.
4. Clarifies how a program accesses external devices.

Reasons for using Assembly Language

1. A program written in Assembly Language requires considerably less memory and execution time than one written in a high –level language.
2. Assembly Language gives a programmer the ability to perform highly technical tasks that would be difficult, if not impossible in a high-level language.
3. Although most software specialists develop new applications in high-level languages, which are easier to write and maintain, a common practice is to recode in assembly language those sections that are time-critical.

1. **Resident programs (that reside in memory while other program execute) and interrupt service routines (that handle input and output) are almost always develop in Assembly Language.**

The Computer Organization - INTEL PC

(i) 8088

- **Has 16-bit registers and 8-bit data bus**
- **Able to address up to 1 MB of internal memory**
- **Although registers can store up to 16-bits at a time but the data bus is only able to transfer 8 bit data at one time**

(ii) 8086

- **Is similar to 8088 but has a 16-bit data bus and runs faster.**

(iii) 80286

- Runs faster than 8086 and 8088
- Can address up to 16 MB of internal memory
- *multitasking* => more than 1 task can be ran simultaneously

(iv) 80386

- has 32-bit registers and 32-bit data bus
- can address up to 4 billion bytes. of memory
- support "*virtual mode*", whereby it can swap portions of memory onto disk: in this way, programs running concurrently have space to operate.

(v) 80486

– has 32-bit registers and 32-bit data bus
– the presence of CACHE

(vi) Pentium

– has 32-bit registers, 64-bit data bus
– has separate caches for data and instruction
– the processor can decode and execute more than one
– instruction in one clock cycle (pipeline)

(vii) Pentium II & III

In performing its task, the processor (CPU) is partitioned into two logical units:

1) An Execution Unit (EU)

2) A Bus Interface Unit (BIU)

EU

- EU is responsible for program execution
- Contains of an Arithmetic Logic Unit (ALU), a Control Unit (CU) and a number of registers

BIU

- Delivers data and instructions to the EU.
- manage the bus control unit, segment registers and instruction queue.
- The BIU controls the buses that transfer the data to the EU, to memory and to external input/output devices,

EU and BIU work in parallel, with the BIU keeping one step ahead. The EU will notify the BIU when it needs to data in memory or an I/O device or obtain instruction from the BIU instruction queue.

When EU executes an instruction, BIU will fetch the next instruction from the memory and insert it into to instruction queue.