

Block 4: The Programming Layer

Yangtao Ge

June 17, 2019

1 Chapter 6: Low-Level Programming Languages and pseudocode

Abstract

This Chapter focuses on how to use a ‘computer system’. The processing of interpreting is a from lowest to more advanced programming language.

i.e. machine code \rightarrow assembly language \rightarrow pseudocode

* This Chapter is related to ENGF0001(1)

1.1 Computer Operations

Definition: Computer is a programmable electronic device which can *store*, *retrieve* and *process* data

Four Operations:

- programmable: manipulate data are stored within the machine along with the data
- store: store data into the memory of the machine
- retrieve: retrieve data from memory of the machine
- process: process data in some way in the arithmetic/logic unit(ALU)

1.2 Machine Language

Just understanding some basic knowlegde about Machine Language is fine

Each type of computer(i.e. CPU) has limited number of machine language to be executed.

Virtual Computer machine is A hypothetical machine designed to **illustrate** important features of a real machine (i.e. it is not real, just a intimation)

Following subsection uses Pep/8 as a example

1.2.1 Features of Pep/8

65536 bytes in total (i.e. 1 byte = 8 bits)

Word length is 2 bytes

ALU is 16 bits

Three components:

- Program counter(PC): Contains the address of the next instruction to be executed
- Instruction register(IR): Contains a copy of the instruction being executed
- Accumulator: a register

address is a physical representation of the memory (i.e. address is a *name* of memory), **it is not the actual place for storing data**

The address range is: 0000 – FFFF (in Hex Decimal)

1.2.2 Instruction Format

Picture *Ref: pp.155-156*

The Two parts of an Instruction:

- Instruction specifier: 8 bits

- Operand specifier: 16 bits

The instruction specifier part of an instruction:

- Operation code(1-4/1-5): specify which register to use
- Register specifier(5): 0 for register A(Accumulator)
- Addressing mode(6-8): how to interpret the Operand part of the instruction.

Two mode of addressing:

- Immediate: Operand is in the operand specifier of the instruction
- Direct: Operand is the memory address named in the operand specifier

1.2.3 Example Instruction

Instruction specifier:

1 1 0 0 0 0 0

Operand specifier:

0 0 0 0 0 0 0 0 0 0 0 1 1 1

Meaning: under 'Immediate Mode', Load the operand into the A register.

1.2.4 Instruction table

Opcode	Meaning of Instruction
0000	Stop execution
1100	Load operand into the A register
1110	Store the contents of the A register into the operand
0111	Add the operand to the A register
1000	Subtract the operand to the A register
01001	Character input to the operand
01010	Character output from the operand

1.3 A Program Example

Ref: pp.160-165 This section detially metioned about the running process of machine language

1.4 Assembly Language

Definition:

- Assembly language: A low-level programming language in which a mnemonic represents each of the machine-language instruction for a particular computer
- Assembler: A program that *translate* an assembly-language program in machine code
- Assembler directive: *Instructions* to the translating program

N.B. Different Machine has different assembly language.

Assembly process

Program in assembly language $\xrightarrow{\text{input}}$ Assembler $\xrightarrow{\text{output}}$ Program in machine code

***Ref: pp.167-174 A lot of references of real code, also refer to ENGF0001(1) coursework*

1.5 Expressing Algorithms