INTRODUCTION TO COMPUTER ORGANIZATION AND ARCHITECTURE

**TUTORIAL 1**

Q1. Describe the function or purpose of the following components of a Computer:

1. Control Unit
2. Arithmetic Logic Unit
3. Bus

Q2. Explain the steps performed by the CPU in an instruction cycle

Load  
Program(Instruction-data) is loaded from storage unit into memory

Fetch

(fetch instruction from memory line-by-line sequentially)

The control unit decodes the instruction to identify the instruction with it’s field

Execute each instruction is executed by CPU

Q3. Specify the hierarchy of memory in a computer system in terms of volatility, speed and density.

Fast, expensive (Small numbers), volatile ^

Register

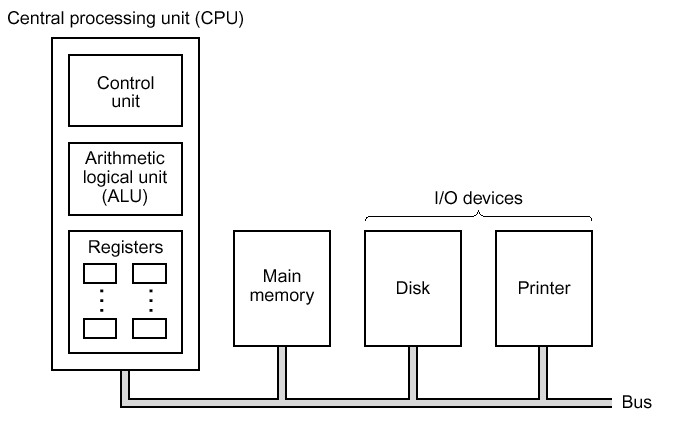
Main memory

Disk storage

Magnetic tapes

Slow cheap(large numbers) non-volatile

Q4. Refer to a simple computer below and the complete instructions are given in the table below. There are sixteen 32-bit registers in the CPU which can be accessed as R0 to R15.



|  |  |  |  |
| --- | --- | --- | --- |
| Instructions | Descriptions | Example | |
| **LOAD** LOC, Ra | Transfers the contents of main memory location LOC into processor register Ra | LOAD A, R0 | Transfer the data from memory location A into R0 |
| **LOAD** LOC, #*index,* Ra | Transfers the contents of main memory location LOC+*index* into processor register Ra | LOADA, #4, R0 | Transfer the data from memory location A+4 into R0 |
| **STORE** Ra, LOC | Transfers the contents of Ra to memory location LOC | STORE R0, B | Transfer the data from R0 into memory location B |
| **STORE** Ra, LOC, #index | Transfers the contents of Ra to memory location LOC+ *index* | STORE, R0, B, #4 | Transfers the data from R0 into memory location B+*4* |
| **ADD** Ra, Rb, Rc | Adds Rb and Rc and store the result into Ra | ADD R0, R1, R2 | R0 🡨 R1 + R2 |
| **ADD** Ra, Rb, #*Num* | Adds Rb and the immediate value *Num*. Store the result into Ra | ADD R0, R1, #12 | R0 🡨 R1 + 12 |
| **MULT** Ra, Rb, Rc | Multiplies Rb and Rc and store the result into Ra  (Assumption: no overflow for multiplication) | MULT R0, R1, R2 | R0 🡨 R1 × R2 |
| **MULT** Ra, Rb, #*Num* | Multiplies Rb with the immediate value *Num*. Store the result into Ra (Assumption: no overflow for multiplication) | MULT R0, R1, #12 | R0 🡨 R1 × 12 |

Part 1:

We are interested to get the solutions for the following four functions. For each function, implement them as a series of instructions issued to the processor.

Assume that all input and output variables have been properly defined in the memory, and the values of all input variables have been initialized. For example, for function (a), the variables H, B and A are located at memory location H, B and A in the processor, respectively.

|  |  |  |  |
| --- | --- | --- | --- |
| Problems Description | | | |
| 1. Compute the area of a parallelogram | 1. Compute the sum of element-wise square | 1. Compute the value of a linear function | 1. Compute the volume of a cube |
| H  B  A = B\*H | F = X2 + Y2 | Y = 3X + 7 | LB  H  W  V = W\*H\*L |

a)

LOAD H, R0

LOAD B, R1

MULT R2, R0, R1

STORE R2, A

b)

LOAD X, R0

LOAD Y,R1

MULT R2, R0, R0

MULT R3, R1, R1

ADD R4, R2, R3

STORE R4, F

c)

LOAD X, R0

MULT R1, R0, #3

ADD R2, R1, #7

STORE R2, Y

d)

LOAD W, R0

LOAD H, R1

LOAD L,R2

MULT R3, R0,R1

MULT R4, R3, R2

STORE R4, V

Part 2:

Write a series of instructions to ***copy*** the array of integers in memory location A = {4, 3, 12, 24, 9} to memory location B. Each element is 4 bytes (32 bits) in length. For example, the first element is located at memory location A, the second element is at memory location A+4, and so on, as shown in the Figure below.

4

A

3

12

24

9

A+4

A+8

A+12

A+16

Memory

B

B+4

B+8

B+12

B+16

LOAD A, R0

STORE R0, B

LOAD A, #4, R0

STORE R0, B #4

LOAD A, #8, R0

STORE R0, B, #8

LOAD A, #12, R0

STORE R0, B, #12

LOAD A, #16, R0

STORE R0, B, #16