1DT301: Computer Technology

Homework 1

Due in class on October 10, 2018

1. **What are the five functional units of a computer?**

The five-consistence function unit in a computer are as following:

* Input Unit: Reads the instruction and data from outside world.
* Output Unit: It reverses the input unit.
* CU (Control Unit): Can maintain order and directs the operation of the entire system.
* Memory Unit: It provides space for storing data and instructions, space for intermediate results and also space for the final results.
* Arithmetic Logic Unit: All calculation is performed and all comparison(decisions)
* Are made in ALU.

1. **In a byte addressable memory with a 64-bit address, what is the maximum size of memory?**

* Memory limit in 16, 32 and 64-bit machine are as follows:
  + 16-bits =65,536 bytes (64 kilobytes)
  + 32-bits=4,294,967,295 byte (4 Gigabytes)
  + 64-bits=18,446, 744,073, 709,551,616 (16 Exabytes)

1. **Describe the addressing modes you have learned.**

* Absolute mode: address of the memory location is given explicitly int the instruction. It is also called as (“Direct Mode”). E.g. (Clear A)
* Register mode: Operand is the contents of a processor register (memory location)

: address of an operand is given in the instruction e.g. (Clear R1)

* Register indirect mode: Effective address (EA) of an operand is in a processor register which is specified in an instruction
* Autoincrement mode: Effective address is in one of the address registers which is specified in the instruction
* Autoincrement and autodecrement modes are useful for implementing Last-in-first-out data structure.
* After the operand is addressed, the contents of the address register are incremented by 1 byte and 2 word and 4 long-word
* Autodecrement mode: Same as above, except the contents are decremented before the operand is accessed
* Basic index mode: 16-bits signed offset and an address register are specified
* Full index mode: 8-bits offset, an address register and an index register (either a data or address register) are specified
* Basic relative and full relative mode: Same as basic index and full index mode respectively, except PC is used instead of address register

1. **Describe what the code “Add 20(R1), R0” computes?**

Effective address of the operand is generated by adding a constant value to the contents of the register.

|  |
| --- |
| **Add 20(R21), R0** |
|  |
|  |
|  |
| **Operand** |

* + Operand is address 1020
  + Register R1 contains 1000
  + Offset 20 is added to the contents of R1 to generate the address 20
  + Contents of R1 don not change in the process of the generating the address
  + R1 is called as an “index register”

|  |
| --- |
| 1000 |

R1

**5. Write an assembly language code which adds 100 numbers. Assume that it is a 32-bit machine and all the numbers are put in an array with the starting address of 1000.**

**6. Describe what the code “Move (SP)+, A” computes.**

**7. Describe the difference between a RISC machine and a CISC machine.**

* CISC instructions utilize more cycles than RISC
* CISC has way more complex instructions than RISC
* CISC typically has fewer instructions than RISC
* CISC implementations tend to be slower than RISC implementations
* Computers typically use CISC while tablets, smartphones and other devices use RISC

**8. Describe what an edge triggered D flipflop is.**

The operations of a D flip-flop is much more simpler.  It has only one input addition to the clock.  It is very useful when a single data bit (0 or 1) is to be stored.  If there is a HIGH on the D input when a clock pulse is applied, the flip-flop SETs and stores a 1.  If there is a LOW on the D input when a clock pulse is applied, the flip-flop RESETs and stores a 0.  The truth table below summarize the operations of the positive edge-triggered D flip-flop.  As before, the negative edge-triggered flip-flop works the same except that the falling edge of the clock pulse is the triggering edge.

|  |  |  |
| --- | --- | --- |
| Inputs | Outputs |  |
| D C | Q Q | Comments |
| 0 ^ | 0 1 | Reset |
| 1 ^ | 1 0 | Set |

**9. Describe the 3-clock cycle computation process for the code “Add R1, R2, R3”.**

**10. Draw** the diagram to compute A-B assuming that they are 8-bit numbers.