COMP 4735 – Operating Systems

Lab1

Ignacio Marquez

1.

AC - accumulator

ALU – arithmetic logic unit

PC – program counter

MAR – memory address register  
MDR – memory data register

CIR – current instruction register

CU – control unit

Memory

The PC will increment, and tell the MAR to see what the relevant instruction in memory is.

The instruction will then be stored in the MDR and CIR. This instruction will move into the control unit, PC will increment, and MAR will store what the CU was instructed to do. This instruction will be fetched from memory, stored in the MDR and AC. This cycle repeats and the next time gets stored in the ALU and calculated.

2.

300 – 1(0001) 940 – Load 940 (3)

301 – 5(0101) 941 – Add 941 (3+2)

302 – 2(0010) 941 – Store 941 (5)

3.

Sign Magnitude:

0001 0101 0111 0010 - Load address 0x572 to accumulator (Accumulator value: 0000 0000 0000 0110, 0x572 value: 0000 0000 0000 0110)

0111 0101 0111 0011 - Load address 0x573 to R1 (Accumulator value: 0000 0000 0000 0110, 0x573 value: 1000 0000 0000 0111)

0100 0000 0000 0000 - Adds contents of accumulator and R1 (Accumulator value: 1000 0000 0000 0001, R1 value: 1000 0000 0000 0111)

0010 0101 0111 0100 - Store contents of accumulator to 0x574 (Accumulator/0x574 value: 1000 0000 0000 0001)

Two's Complement:

0001 0101 0111 0010 - Load address 0x572 to accumulator (Accumulator value: 0000 0000 0000 0110, 0x572 value: 0000 0000 0000 0110)

0111 0101 0111 0011 - Load address 0x573 to R1 (Accumulator value: 0000 0000 0000 0110, 0x573 value: 1111 1111 1111 1001)

0100 0000 0000 0000 - Adds contents of R1 to accumulator (Accumulator value after: 1111 1111 1111 1111, R1 value: 1111 1111 1111 1001)

0010 0101 0111 0100 - Store contents of accumulator to 0x574 (Accumulator/0x574 value: 1111 1111 1111 1111)

4.

Heap is a place in memory. It stores what a user tells it to store and must be allocated manually. It is managed by the programmer.

5.

Stack is a place in memory. It stores temporary variables from a function. The stack is managed by the system.

6.

Variables, etc are passed to the stack via a push every time a function declares a new variable.

7.

value – the actual value assigned to a variable

pointer – points to a value in memory

reference – takes the address of the variable