Project 1b – Overview Document

Project Overview:

Contained in this document is an overview of the simulator.c file for Project 1. This project is a simulator for UST-3400 (Rip Van Sawinkle) machine code that is passed in from the assembler provided to us by Prof. Myre, as our assembler had flaws that would have carried over to the simulator, since it would have passed incorrect machine code for certain situations.

main():

The main begins by reading in the machine code from a file specified through a command line argument. For each line in the machine code file we will read in the machine code on that line, and pass it to the main routine function for interpretation and execution of that machine code.

int mainRoutine(int machineCode, statetype *stateptr):

This function calls getOppCode and passes the current argument to the correct function based off of the opcode. (if the opcode is for 'add', go to the function that executes add etc.)

Int determineOppcode(int machineCode):

The passed in machineCode has a bitmask applied to it and a bitshift in order to isolate the opcode. The opcode is then returned.

Void add(int machineCode, statetype *stateptr):

Bitmasks are applied to the machineCode in order to isolate register A and register B, and destReg. The values in register A and register B are added together and put in destReg.

Void lw(int machineCode, statetype *stateptr):

Bitmasks are applied to the machineCode in order to isolate register A, register B, and offset. The value at the memory location of register B is then taken plus the offset and stored in register A.

Void sw(int machineCode, statetype *stateptr):

Bitmasks are applied to the machineCode in order to isolate register A, register B, and offset. The value at the memory of register A is then stored at register B plus the offset.

Void beq(int machineCode, statetype *stateptr)

Bitmasks are applied to the machineCode in order to isolate register A, register B, and offset. If the values in register A and register B are equal, then we jump to the address of pc +1 + the offset value.

Void printstate(statetype *stateptr):

Simply prints out the state details of the machine.

Difficulties in implementation:

The only real problem we had was when Mike's Virtual Box completely crashed. This resulted in Tyler taking the lead on a large portion of the simulator while Mike tried to fix it to no avail, ultimately giving up and using Hank anyway. Otherwise, the simulator was very straightforward and simple in comparison to the assembler.