Team Design Note

In our design, this simulator must do it best to simulate a real computer system. So, in the CPU, we must have a CU (Control Unit) to do the Instruction Cycle. And there are still some classes needed to do the work about the Memory, Instruction and Register. The figure below depicted our system hierarchy. It’s the initial plan of the computer system and we would add up more to it in the future.



With the hierarchy easily seen and understood, now we’d like to address a few design ideas of the LEAF elements (Rom Loader, Memory, ISA, CU, Registers, and ALU) of the computer. The Rom Loader is supposed to actually load the boot program from files, which is used as ROM, to the memory. However, it works now as loading some hard coded instructions from within the Java program temporarily. We will change it soon.

The Memory is implemented as an array of 2048 integers. We did it in order to make efficient bit operations as well as saving space. The size of it is designed to be expandable.

Our system is based on certain Instruction Set Architecture (ISA). It is not explicitly “owned” by CPU, though. A bunch of decoding schemes and instruction definition conventions would be included in this element.

The Control Unit would be in charge of lots of tasks as it is in practice. It needs to be able to direct Data Handling Operations such as load and store, and it is also the one that executes the Instruction Cycle, which is of great significance in our system.

What Processor Registers include INITIALLY has already been put on the figure. Their bit length is not necessarily the same as required in the project description. We packaged them together into the CPU Java class.

ALU would implement some arithmetic and logic operations.

The object design derived from the above figure. In addition, we made use of a few software engineering techniques for the implementation. Examples are Java interface, MVC pattern, etc. Lots of classes are in the source code and the code is arranged as well as possible in order for further development.