Project 5

In this project you will integrate the ALU and RAM devices built in projects 2 and 3 into a computer system capable of executing programs written in the machine language introduced in project 4.

Objective

Build the following chips:

Memory

CPU

Computer

The building blocks that you can use are the chips listed in previous projects and the chips listed in this project.

Contract: Build a hardware platform capable of executing programs written in the Hack machine language. Demonstrate the platform's operations by having your Computer chip execute correctly the three test programs described below.

Test Programs

A natural way to test your topmost Computer-on-a-chip implementation is to have it execute sample programs written in the Hack machine language. In order to run such a test, one can write a test script that loads the Computer chip into the hardware simulator, loads a program from a text file into its ROM32K chip-part (the instruction memory), and then runs the clock enough cycles to execute the program. We provide three such test programs, along with corresponding test scripts and compare files:

Add: Adds the integers 2 and 3, and writes the result in RAM[0].

Max: Computes the maximum of RAM[0] and RAM[1], and writes the result in RAM[2].

Rect: Draws on the screen a rectangle of RAM[0] rows of 16 pixels each. The rectangle's top left corner is located at the top left corner of the screen.

Before testing your Computer chip on any one of these programs, review the test script associated with each program, and be sure to understand the instructions given to the hardware simulator. If needed, consult the Test Description Language Guide (link below).

Chips

Implement the computer platform in the following order:

Memory: The three main chip-parts of this chip are RAM16K, Screen, and Keyboard. The Screen and the Keyboard are available as builtin chips. Although the RAM16K chip was built in project 3, we recommend using its builtin version.

CPU: The Central Processing Unit can be built from the ALU built in project 2, the Register and PC chips built in project 3, and logic gates built in project 1. As usual though, we'll use their builtin versions. In particular, use the builtin ARegister, DRegister, and PC chips. These builtin chips have exactly the same functionality as the Register and PC chips built in project 3, but they are required in order to be consistent with the project test scripts.

Instruction Memory: Use the builtin ROM32K chip. This chip is essentially the same as the RAM chips built in project 3, but the builtin version has an interface that allows loading a program from a text file into the memory.

Computer: Can be built from three chip-parts: CPU, Memory, and ROM32K. The first two chips are built in this project. The latter chip is builtin.

References

Hardware Description Language

Test Description Language (to be used as needed, for understanding the supplied test scripts)

Hack Chip Set API

Screen chip demo

Keyboard chip demo

ROM32K chip demo

Implementation Tips

1. The three chips in this project (Memory, CPU, and Computer), are to be implemented in HDL and tested in the supplied hardware simulator.

2. The three test programs with which you have to test your Computer chip are Add.hack, Max.hack, and Rect.hack. These programs are written in the binary Hack machine language. The symbolic versions of these programs – in case you want to review their logic – are available in the projects/6 folder.

3. Normally, when running a program on some computer, and not getting the desired results, we conclude that the program is buggy. In this project, the supplied test programs are bug-free. Therefore, if running a test program yields unexpected results, it means that the computer platform on which the program runs (Computer.hdl and/or some of its chip-parts) is buggy. If that is the case, debug your chips and keep testing.

4. In the course of implementing the CPU and other chips in this project, you may be tempted to specify and build some internal ("helper") chips of your own. Be advised that there is no need to do so; The Hack CPU can be implemented elegantly and efficiently using only the chip-parts mentioned above, plus some elementary logic gates built in project 1 (of which we recommend using their builtin versions).

5. How to make sure that you use only builtin chip-parts? Simple: Use only the .hdl files stored in the projects/5 folder. Don't add any other .hdl files to this folder.

6. The projects/5 folder may contain a set of files whose names include the label "external". These test files are no longer in use, and will be removed from future versions of the Nand to Tetris software suite. Therefore, if these files exist in your project folder, ignore them.