The Nand to Tetris Software Package (“Desktop Version”)

All the recommended go-to tools for completing and testing all the Nand to Tetris projects are available in the new online IDE, which can also be used offline. If you plan to use this IDE, there is no reason to keep reading this page.

If, instead, you wish to use the old desktop version of the Nand to Tetris software package, this page will guide you how to download and install it. This version of the Nand to Tetris tools is no longer supported, and will be decommissioned soon.

The desktop version of the Nand to Tetris software package runs on Windows, Unix, and Mac OS.

Download

Download the Nand2tetris Software Suite Version 2.7 (about 1MB).

Mac users: after downloading, read this Setup Guide for Apple MacOS, written by Yong Bakos.

Windows users: after downloading, put the downloaded zip file in an empty folder on your computer, and extract its contents as is, without changing the folders structure and folder/file names.

In order to use the nand2tetris software tools, your computer must be equipped with a Java Run-time Environment. The JRE can be downloaded freely from many sites including this one. For best performance, download the latest available version.

About the Software

The software suite consists of two folders: projects, and tools.

The projects folder is divided into 14 project folders named 0, 1, ..., 13. These folders contain files that you have to modify and complete as you work on various nand2tetris projects.

The tools folder contains the nand2tetris software tools. It's a collection of Java programs and files.

The .bat and .sh files are batch and script files, used to invoke the nand2tetris software tools. These files are explained below.

The bin folder contains the code of the nand2tetris software tools. It consists of several sub-folders containing Java class files and supporting files.

The builtInChips and the builtInVMCode folders contain files that are used by the supplied hardware simulator and VM Emulator, respectively.

The OS folder contains a compiled version of the Jack operating system.

Setup

The supplied software tools are designed to be run from your computer's command-line environment (also known as "terminal", or "shell"). Command-line environments vary from one operating system to another, and using them requires basic knowledge of various OS shell commands.

In order to eliminate this overhead, we supply batch files (for Windows) and scripts (for Unix and Mac OS), developed by Mark Armbrust. These batch and script files enable invoking the supplied nand2tetris tools from the command line. They can be used from any folder on your computer, without requiring full paths to the files on which they operate. The software tools accept spaces in folder and file names.

Mac and Linux users: Before running the scripts, you must first change their file attributes to include "executable". You can then run the scripts by typing their name, as well as the .sh extension, in the terminal environment.

If you want to avoid typing the 'sh' extensions, you can create (once and for all) symbolic links in your ~/bin folder. Here is an example how to do it for, say, the HardwareSimulator tool:

```
ln -s ~/nand2tetris/tools/HardwareSimulator.sh HardwareSimulator
chmod +x HardwareSimulator
```

Windows users: For the batch files to work from the command line, you must add (once and for all) the nand2tetris/tools folder to your PATH variable.

To run a batch file from command-line, type its name, without the .bat extension.

If you use Windows 7 64-bit you need to install the 64-bit version of Java so that 64-bit cmd exe can run Java commands in batch files. If you get the output "'java' is not recognized..." you likely only have the 32-bit Java installed on your computer.

You can create desktop icons and use them to invoke the interactive versions of the following supplied tools: HardwareSimulator, Assembler, CPUEmulator and VMEmulator. This can be done by finding the disk locations of the respective batch files, right-clicking on them and picking "Send to > Desktop." Edit the shortcuts' properties and set "Run" to "minimized."

Usage

Hardware Simulator: To invoke the hardware simulator in interactive mode, type "HardwareSimulator" in the command line. For example:

```
C:\...\projects\2> HardwareSimulator
```

(a window will open up, running the interactive version of the Hardware Simulator)

To invoke the hardware simulator in batch (shell/cmd) mode, type "HardwareSimulator" in the command line. For example:

```
C:\...\projects\2> HardwareSimulator ALU.tst
```

(invokes the simulator, loads the given test script, executes it, and reports the result). Note that the simulator's interactive mode also enables loading and executing test scripts.

Successful test example:

```
C:\...\projects\2> HardwareSimulator ALU.tst
End of script - Comparison ended successfully
```

Failed test example:

```
C:\...\projects\2> HardwareSimulator ALU.tst
Comparison failure at line 24
```

Error in the associated HDL file:

```
C:\...\projects\2> HardwareSimulator ALU.tst
In HDL file C:\...\projects\2\ALU.hdl, Line 60, out[16]: the specified sub bus is not in the bus
range: load ALU.hdl
```

CPU Emulator and VM Emulator: The operations of these tools follow the same convention described above. If you invoke either tool without a parameter, the tool will work in interactive mode; if you supply a parameter (test script), the tool will run batch-style.

Assembler: Typing "Assembler" starts the supplied assembler in interactive mode. Typing "Assembler xxx.asm" assembles the specified xxx.asm file and generates a file named xxx.hack, containing the translated binary code. Note that the assembler's interactive mode also enables loading and translating .asm files.

Successful assembly example:

```
C:\...\projects\4\fill> Assembler Fill.asm Assembling "c:\...\projects\4\fill\Fill.asm"
```

Failed assembly example:

```
C:\...\projects\4\fill> Assembler Fill.asm
Assembling "C:\...\projects\4\fill\Fill.asm" In line 15, Expression expected
```

To compare the resulting .hack code file to some expected .hack file, use the supplied TextComparer tool, described below.

Compiler: Typing "JackCompiler fileName.jack" compiles the supplied Jack file. Typing "JackCompiler folderName" compiles all the Jack files in the specified folder. Wildcards are not supported. Examples:

Compile a single file example:

```
C:\...\projects\9\Reflect> JackCompiler Mirrors.jack
Compiling "C:\...\projects\9\Reflect\Mirrors.jack"
```

Compile a folder named "reflect" (example):

```
C:\...\projects\9> JackCompiler reflect
Compiling "C:\...\projects\9\reflect"
```

TextComparer: Compares two given files ignoring white space, and reports success or failure. For example, suppose you run the hardware simulator with some test script and get a comparison failure. If you want, you can then use the TextComparer to investigate the problem:

```
C:\...\projects\2> HardwareSimulator ALU.tst
Comparison failure at line 24
```

```
C:\...\projects\2> TextComparer ALU.cmp ALU.out
Comparison failure in line 23:
|0101101110100000|0001111011010010|1|1|0|0|0|0|0001111011010010|0|0|
|0101101110100000|0001111011010010|1|1|0|0|0|0|0001111011010010|0|1|
```

(Note the line number discrepancy between the reports of the two tools).

Help: In Windows, each batch file accepts a "/?" option that shows its intended usage. On Mac and Unix, use "-h". For example:

```
C:\...\projects\9>JackCompiler /?
Usage:
JackCompiler Compiles all the .jack files in the current working folder.
JackCompiler folderName Compiles all the .jack files in the specified folder.
JackCompiler fikeName.jack Compiles the specified Jack file
```

Source Code

All the software tools in the old desktop version of the Nand to Tetris software package are written in Java. If you wish to inspect, modify, or extend some tool, you can download the source code. Before compiling the source code on your computer, read Readme.txt and the ChangeLog.txt file.

License: GNU GPL (General Public License).