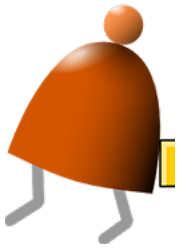
From Nand to Tetris

Building a Modern Computer From First Principles



Project 3: Sequential Chips

Background

The computer's main memory, also called Random Access Memory, registers, each designed to hold an n-bit value. In this project you two main issues: (i) how to use gate logic to store bits persistently locate ("address") the memory register on which we wish to operat

Objective

Build all the chips described in Chapter 3 (see list below), leading The only building blocks that you can use are primitive DFF gates, chips described in previous chapters.

Chips

Chip Name	Description
DFF	Data Flip-Flop (primitive)
Bit	1-bit register
Register	16-bit register
RAM8	16-bit / 8-register memory
RAM64	16-bit / 64-register memory
RAM512	16-bit / 512-register memory
RAM4K	16-bit / 4096-register memory
RAM16K	16-bit / 16384-register memory

Contract

When loaded into the supplied Hardware Simulator, your chip desig
supplied .tst script, should produce the outputs listed in the suppli
simulator will let you know. This contract must be satisfied for eac
which is considered primitive, and thus there is no need to implem

Resources

The relevant reading for this project is Chapter 3 and Appendix A.
Chapter 3 should be implemented in the Hardware Description Lan

For each chip, we supply a skeletal .hdl file with a missing implem
supply a .tst script that instructs the hardware simulator how to te
the correct output that this test should generate. Your job is to co
files.

The resources that you need for this project are the supplied Hard
you've downloaded the Nand2Tstris Software Suite, these files are
directory is further partitioned into two sub-directories, for reasor

Tips

The Data Flip-Flop (DFF) gate is considered primitive and thus the
encounters a DFF chip part in an HDL program, it automatically inv
nand2tetris/tools/builtInChips/DFF.hdl implementation.

Built-in chips: When constructing RAM chips from lower-level RAM
versions of the latter. Otherwise, the simulator will recursively ge
objects, one for each one of the many chip parts that make up a ty
simulator to run slowly, or, worse, out of memory. i.e. out of the r
simulator is running.

To avert this problem, we've partitioned the RAM chips that you ha
directories, named projects/03/a and projects/03/b. This partitio
only: when building the chips stored in b, the simulator is forced t
level chip parts whose .hdl programs are stored in a but not in b.

Tools

All the chips mentioned projects 0-5 can be implemented and test
Here is a screen shot of testing a built-in RAM8.hdl chip implement