**Computer Structure Lab - Fall 2002**

Dr. Guy Even

# Handout #5: A Load/Store Machine

## 1 Assignment #1 <br> *submission deadline: beginning of 8th lab meeting*

Design of the Load/Store Machine.

1. (15 pts.) Address translation. The main memory address space to which the Load/Store Machine can both read and write is 0x00800000-0x008fffff.

   Suppose we are interested in giving the Load/Store Machine the illusion of a 16-bit main memory address space with the addresses 0x0000-0xffff. Show how the Address Translation Module can support this illusion. Refer to the PC register and the address used for memory accesses.

2. (15 pts.) Assume that due to the library, we always want to have 32 registers in the GPR, although we only really need 31 registers. There are two ways to implement the register `R0` in the GPR. In one way, write access to `R0` are disabled to keep it with a zero. In the second way, read accesses to `R0` are pulled down to output a zero. Compare these two methods and explain which method is cheaper/faster.

3. (10 pts.) Design the Memory Access Control module. Outline the differences between this module and the Read and Write Machines.

4. (40 pts.) Complete the the Load/Store Machine design. Your design should be as simple as possible. Do not try to make a design that will be easy to use as a basis for the DLX! That will complicate your design.

   (a) (25 pts.) Prepare designs of the blocks in the datapath (PC environment, GPR environment, IR environment, etc.). Each design should be organized as follows: list of inputs and outputs, definition of functionality (equations describing relations between outputs and inputs), and drawing (you may use counters, decoders, etc. as building blocks of your designs).

   (b) (10 pts.) Prepare a table listing *all* the control signals, their meanings, their equations , and the ports they are connected to.

   (c) (5 pts.) Submit printouts of your VHDL design and schematics.

5. (20 pts.) Prepare test vectors for the control of the Load/Store Machine. Submit a list of paths, input values for each path, and expected output values for each path.

Guidelines: (a) Name all signals and modules using only uppercase letters. Do not use non-letters in the names except for underscores. (b) When the Load/Store Machine fetches an instruction which is not a load or a store instruction it halts by entering a “halt” state. (c) We suggest to add an output signal signifying the state of the control to simplify testing of your your design.

## 2 Assignment #2
## *submission deadline: beginning of 9th lab meeting*

Simulation of the Load/Store Machine.

1. Test the control of the Load/Store Machine using the test vectors you designed. Submit printouts of your simulation showing that the outputs are as expected. Annotate the printouts by hand written explanations (which path is tested, what is seen, etc.).

2. Using the IO_SIMUL module, prepare tests to verify all the RTL instructions. Submit a printout of the initialization of the main memory as well as printouts of simulations of these tests annotated with explanations.

3. Submit printouts of executions of whole instructions demonstrating the correctness of your design. Annotate the simulations with explanations.

   Remark: The last two items can be shown in the same simulation. If you choose to do so, make sure not to omit any of the required data and explanations.

## 3 Assignment #3
## *submission deadline: beginning of 10th lab meeting*

Implementation of Load/Store Machine with a monitor slave and a logic analyzer.

1. Implement a design of your Load/Store Machine with a monitor slave and a logic analyzer.

2. Use DLX assembler to generate a short program. An example of a program appears in the appendix.

3. Submit printouts of the RESACTRL program. Show that your machine executes instructions correctly. Annotate the printout with explanations of what is happening in each cycle.

## The GPR Environment

The GPR environment is depicted in the following figure. There are two ways to protect register $R0$. The first way is to zero the value that is read from register $R0$; this method is depicted as "Option#1" in the figure. The second way is to cancel writes to $R0$; this method is depicted as "Option#2" in the figure.

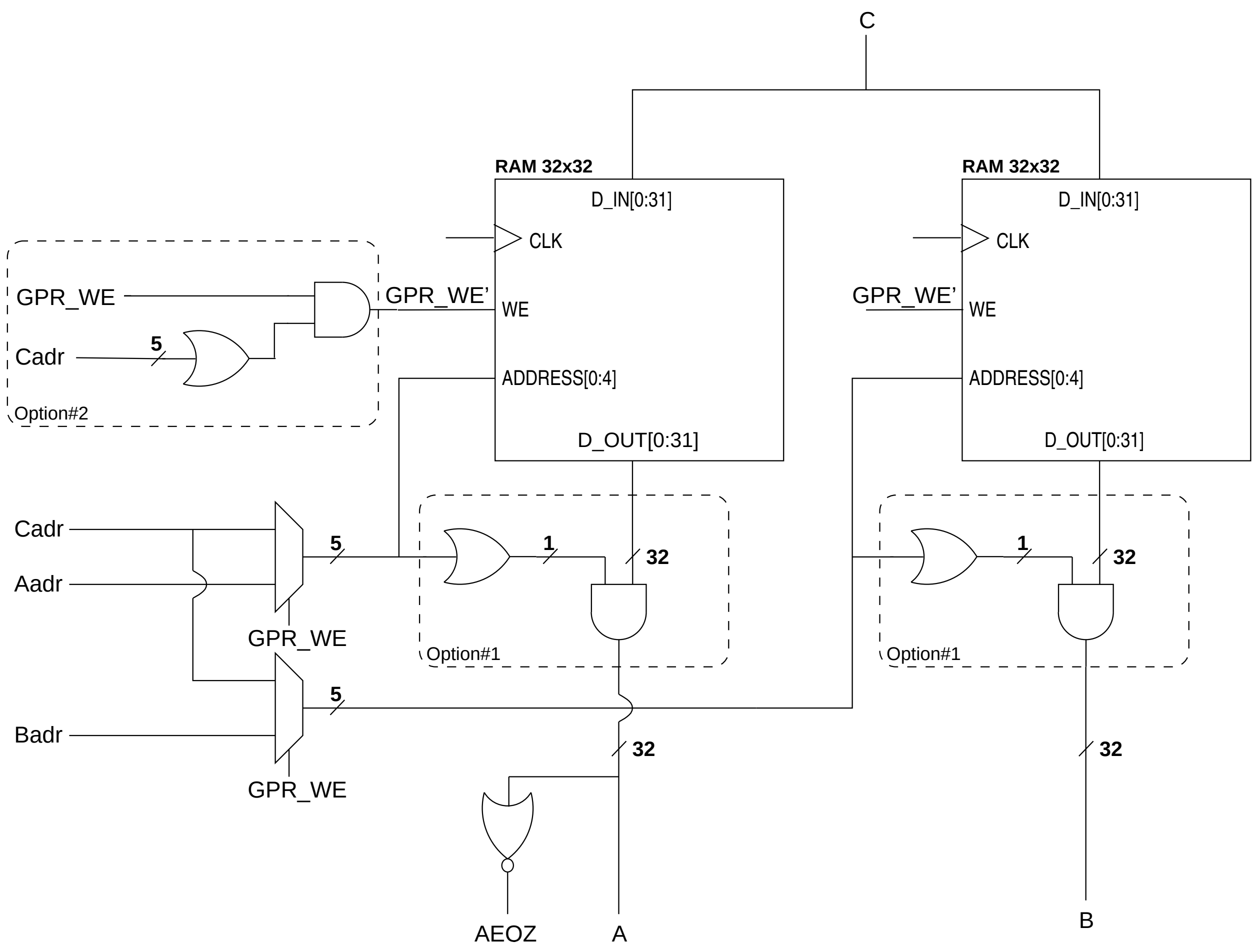


Figure 1: GPR Environment