

**Department of Electrical and Computer Engineering
State University of New York at Stony Brook**

ESE 555 Advanced VLSI Systems Design (Fall 2009)

CAD Assignment 3: The Register File

Assignment

To design a Register File (RF) for your microprocessor.

Description

In this CAD assignment, you will design an 16-word, 16-bit RF with 1 write port and two read ports. One of the structures that is central to the datapath is an RF. An RF consists of a set of registers that can be read from or written to. Writing a register requires (i) an address, (ii) the data to be written and (iii) a signal that controls the write timing. Reading an RF can often be controlled by just the address but may include additional logic. The RF for this project is to include an array of flip-flops or latches and properly sized buffers for the associated read/ write circuits. Address decode logic is not required in CAD3, but will have to be add-ed to your design when the RF is integrated with the microprocessor control circuitry.

Register Cell

Conventionally, an RF would consist of an array of static RAM cells with read/ write circuitry and a sense amplifier. Though this type of implementation yields a fast, compact design, it requires much design effort, especially for the sense amplifiers. In small RFs (few registers), the support circuitry for the SRAMs is used by a small number of registers, making the SRAM-based RF larger than some other types. An alternative approach is to use a latch consistent with your clocking scheme, modified to have two read ports and one write port. This method makes use of pass transistors at the input and the output with the decode control signals driving their gates. Alternatively, tri-state buffers could be used to read the RF, and gated clocks could be used to write them. An-other way to implement an RF is with multiplexors, as shown in Figure 1. While this implementation is conceptually simple, routing all of the mux inputs from the array cells to the muxes will create unnecessary routing bottlenecks.

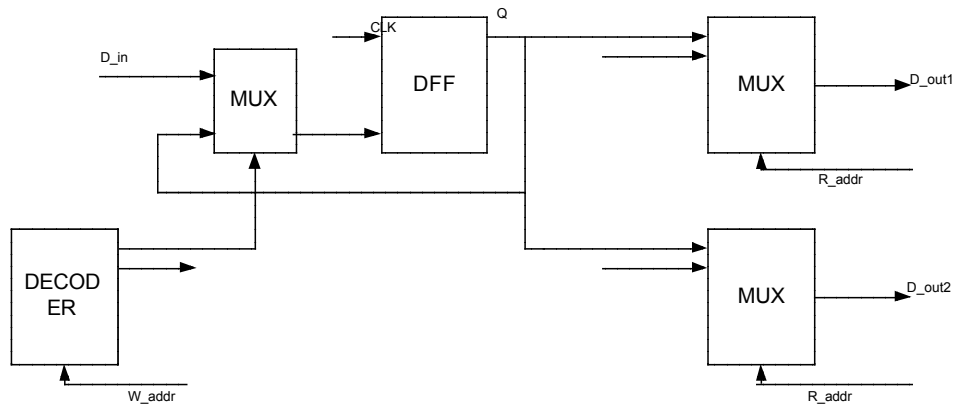


Figure 1: Register File

Drivers and Buffers

Control signals shared by all 16 bits of a register can have large capacitive loading compared to other signals in your design. The data output buses on the RF may also have large loading associated with them. These signals are candidates for transistor sizing on the gates that drive them or possibly even the insertion of buffers. In either case, these design decisions should be based on Spectre simulations.

Bus Strategy

You can use several methods to drive values onto the microcontroller buses. Two common ones that you can use in this class are multiple buses coming out of the RF with a mux to select the proper word. Another possibility is to have tristate drivers on the outputs of each component that can drive the bus. This will require only one bus. Note that tristate buses require keeper latches if they have active gate inputs connected to them and are allowed to float to intermediate voltages. It is important to decide which strategy you will use. After this assignment, it will be very difficult for you to come back and change the RF.

Procedure

Schematic design

Determine your read/ write methodology and select (from among your group's CAD 2 cells) a register, which will serve as the basis of your register file array cell. Also determine your group's bussing strategy. You may also design a new register cell if you wish. You will not be responsible to design the decoder for this CAD assignment. If you decide to use a register from CAD 2, copy the layout and schematic that you select into the directory cad3.

Layout

Since the RF is one of the key modules of your datapath, care is needed to minimize the parasitic capacitances on critical signal nets, as this directly affects the performance of your microprocessor. Keeping the structure dense and compact while avoiding long interconnects is a prime goal. Mirroring a cell and thereby sharing wells and the supply lines is one of the most frequently used strategies. You should design the floor-plan and routing strategy before you start on your layout. Power lines in the cells should be dimensioned so that they can carry the current required by the row of connected cells. A good ballpark figure for this process is to keep average current densities below 1 milliamp/micron of metal width. Use hierarchy effectively to build up your register file layout.

Design Verification

You may end up spending a lot of time on verification if you have errors. The best way to avoid this is to check the design thoroughly as you go along. It is much better to spend 5 minutes in checking the circuit, than to waste 30 minutes debugging it later. Run DRC, LVS and Extraction on the entire register file.

Analog Simulation

For this assignment, you must accurately simulate the register file. Running Spectre on the whole register file may be time consuming and troublesome. Depending on your read-out approach, you should be able to simulate only one bit of one register (along with any control-signal buffering and readout logic), and still get an accurate result. To do this, you must add capacitance to any control or bus lines as appropriate to represent the loads that are not explicitly simulated in your model of the circuit. You'll often develop a "speed-path" schematic like this, prior to starting layout, to analyze the circuit you're designing. This schematic isn't used in LVS since it often contains extra devices and ports. Estimate the capacitance of wires for the simulation. Find the setup and hold times for data being written into the RF.

Requirements

You should have the following in your groups cad3 directory:

- Schematic of the entire register file, including the drivers/buffers.
- Spectre traces showing how you calculated delays (show data being changed from 1 to 0, and from 0 to 1).
- Layout of the Register file.
- DRC report
- LVS report
- Extract report

Deadline

You need to turn in CAD3 by Thursday, October 15, 2009, 12:30 pm.