CMPE 315: Principles of VLSI Design
Project Cover Page

Project Part 1

Name: Brian Weber
Section: CMPE640 01

Date Submitted: 11/22/2017

TA / Grader Use Only:
Late Submission Deduction (20% per day):

Other Deductions:

Final Lab Grade:

Comments to student:

Contents

1 Current Status of Code (Read Me)

Before getting into the bulk of the report I would like to write a short summary of the status of my code. In it's current state, it works completely and matches both test benches posted on Dr. Patel's website, with one small exception: in the beginning of test bench 2, reset has to be asserted for an extra clock cycle in order to reset the valid cells correctly. In addition, I did not have time to clean up my code very much, so it is currently pretty messy. I expect there are many floating signals and unused inputs/outputs of entities that were abandoned as I rushed to fix bugs. I also have not optimized most of my logic for cmos at this point. My first goal was to get the code working, then polish later, but as it turns out, I didn't have any time for polishing. To test, I have been hard coding timings into test benches, and looking at the wave outputs. I do not have time to write different test benches, so I will not have sample input and output files. Instead, I will include the test benches that I have, along with the figures of the correct timing outputs.

2 Entity Hierarchy

All entities are paired with architecture "structural".

i) chip
   a) Counter
      1) rd_wr_hit_miss_reg
         (i) dff_reset
         (ii) Dlatch_Reset
      2) SR18
         (i) dff_reset_high
      3) dff_reset_high
      4) dff_reset
      5) srff
   b) Cache_Block
      1) Cache_Cell_Row

(i) Cache_Cell_Valid
(a) SRlatch
(b) tx
(ii) Cache_Cell_Tag
(a) Cache_Cell
(iii) Cache_Cell_Data_Block
(a) Cache_Cell

c) Decoder
d) Hit_Miss
1) Compare
e) Output_Enable
1) tx
f) register8
1) dff_reset

3 Chip

For the most part, this design sticks to the overall design shown in the prompt. However, this design only uses one decoder to select which cache cell will be operated on, rather than both a decoder and a multiplexor. Figure shows the overall design of the chip.

4 Counter (State Machine)

The state machine is an entity called Counter, which is centered around a shift register that keeps track of the number of clocks that pass after the busy signal goes high. Depending on hit/miss status of each operation, different things are done depending on the clock count after busy is set high. Another key part of the state machine is a group of registers for storing both the inputs from the cpu, and whether or not there was a hit or a miss. There is an entity called rd_wr_hit_miss_reg that stores the operation (rd or write), as well as whether it was a hit or miss, and outputs 4 separate lines. Figures 4, 4, 4, and 4 show input and output waveforms that were generated from the test bench Counter_Test. These figures prove the correct operation of the state machine. The definitions of the signals are as follows:

- s_reset: (Input) The global reset signal.
- s_clk: (Input) The global clock signal.
- s_busy: (Output) The global busy signal.
- s_start: (Input) The start signal from the CPU.

- s_cpu_dout_en: (Output) The signal which enables the transmission gate from cache output to the CPU.
- s_cache_write: (Output) The signal which triggers all writes to the cache.
- s_hit_miss: (Input) The signal from the Hit_Miss indicating whether there is a hit or a miss.
- s_mem_enable: (Output) The signal that enables external memory.
- s_rd_wr: (Input) The rd_wr signal given from the CPU.
- s_rd_wr_o: (Output) A rd_wr signal that is sent to other modules after the original rd_wr is latched.
- s_rm_wr_en: (Output) A signal that allows writing to the cache during a read miss.
- s_wr_hit: (Output) A signal that is sent to other modules so they know there was a write hit.
- s_write_[0:3]: (Output) Signals that select which block to be written to during read a read miss.

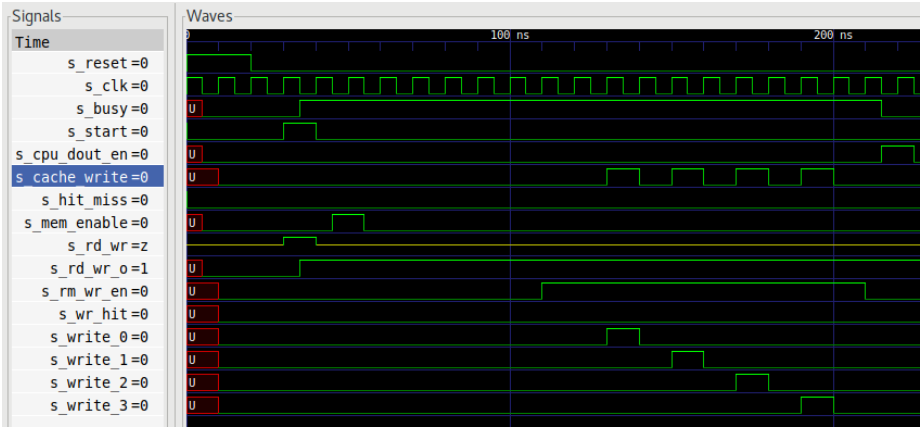

Figure 1: Counter module timing during read miss.

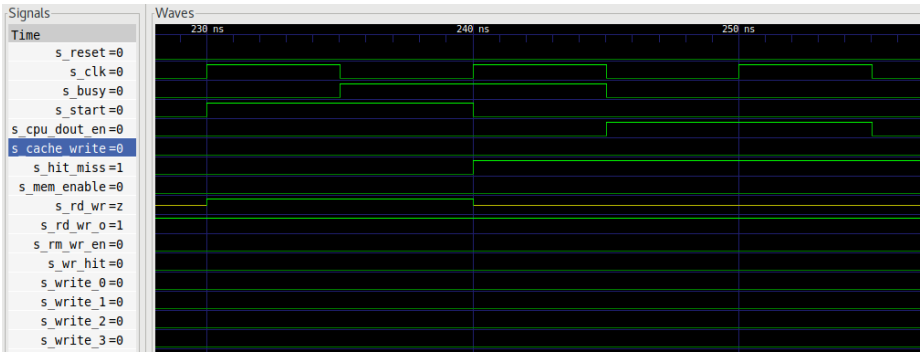

Figure 2: Counter module timing during read hit.

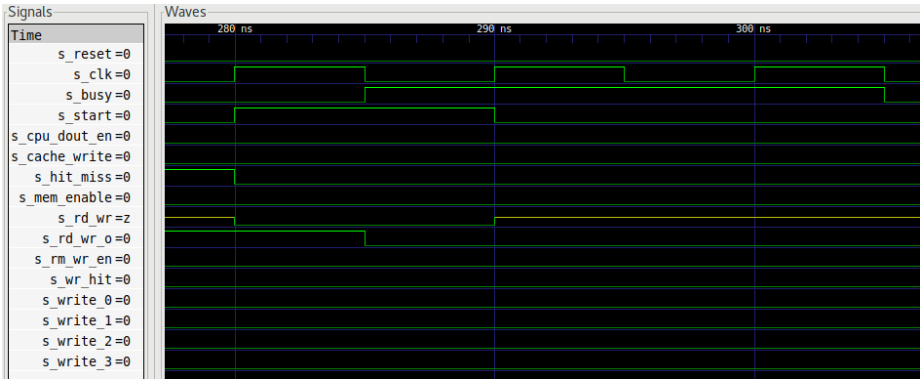

Figure 3: Counter module timing during write miss.

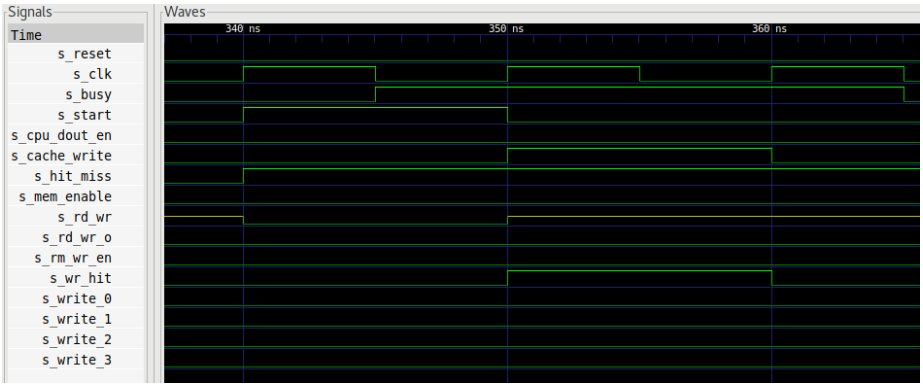

Figure 4: Counter module timing during write hit.

5 Cache_Block

The cache block is a block of positive latch enabled Dlatches, indexed by the cpu address. In front of each Dlatch is a transmission gate. The inputs and outputs of each row all come from and lead to the same 8 bit bus, but only one row at a time can be selected due to the Decoder module. This prevents conflicts on the outputs as well as prevents more than one row from being written to at a time. The valid cells are asynchronous sr latches, and are set while data is being written from memory during a read miss. Tags are also set at this time. The valid and tag bits are output as soon as a row is selected, however, for a byte to be output, the internal rd_wr signal must be high, and a column must be selected. Figure **??** shows a waveform diagram of the cache block. At 10 ns, the data arrives at the input, the rd signal is turned on, and the row and column are selected. At 15 ns, cache_write is turned on, writing the input data. Since rd is on, the data shows on the output as well. At 15 ns, the tag and valid bits are also set. Notice that after the rd signal turns off at 20 ns, the data output becomes high impedence, however the tag and valid bits continue to output. At 30 ns, different data is set at a different position. At 40 ns, the old data is read, and at 45 seconds, the new data is read.

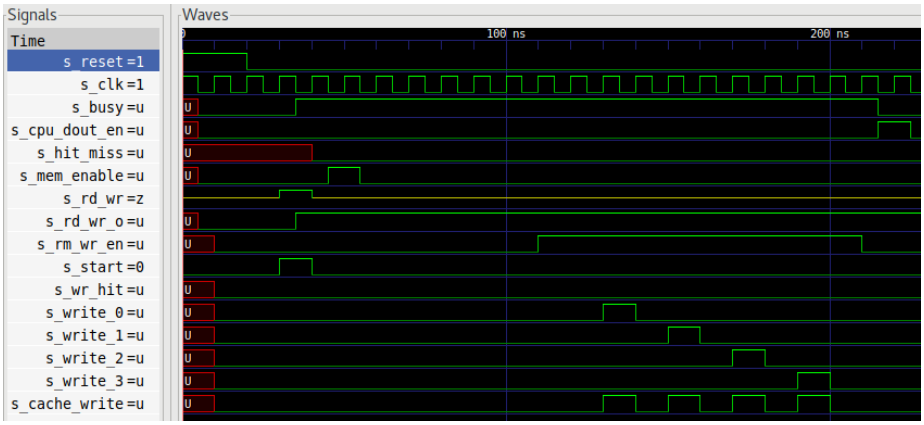

Figure 5: Timing diagram showing the functionality of the cache_block.