EECS 151/251A Homework 9

Instructor: Prof. John Wawrzynek, TAs: Christopher Yarp, Arya Reais-Parsi

Due Monday, May 6$^{\text{th}}$, 2019

Problem 1: Multiplying Signed Numbers by Hand [8 pts]

Using the method shown in class, multiply *by hand* the following **signed** 5-bit numbers. Show your work.

(a) 12×5

(b) 3×-12

(c) -15×-1

(d) -8×7

Problem 2: Analyzing Multiplier Structures [14 pts]

Consider multiplying 7×5 using the **unsigned** array multiplier presented in class.

(a) Assume that we use carry-propagate adders, and that the delay of each adder is τ_{FA}. How long does it take to compute the product? What are the intermediate signals in the array (label these signals in a diagram)? In what order are the product bits ready?

(b) Now, assume that we use a carry-save adder instead. How long would it take to compute the product? What are the intermediate signals in the array (label these signals in a diagram)? In what order are the product bits ready?

Next, consider multiplying a -7×-5 using a **signed** carry-propagate array multiplier.

(c) Assume that the multiplier does not use any of the Baugh-Wooley simplifications. How long would it take to compute the product? What are the intermediate signals in the array (label these signals in a diagram)? In what order are the product bits ready?

(d) Now, assume that the multiplier uses the Baugh-Wooley simplifications. How long would it take to compute the product? What are the intermediate signals in the array (label these signals in a diagram)? In what order are the product bits ready?

Problem 3: Booth Recoding [12 pts]

Design a circuit to perform multiplication of two **unsigned** 4-bit numbers with Booth recoding.

Version: 1 - 2019-04-26 16:44:31-07:00

Problem 4: Constant Multiplication [8 pts]

Derive the minimal canonic signed digit (CSD) multiplier for:

(a) $Y = 415 * X$

(b) $Y = 238 * X$

Show your conversion to CSD form and the resulting multiplier circuit using adders and subtractors. Assume that X is a 12-bit unsigned integer.

Problem 5: *251A Only.* Constant Multiplication [5 pts]

The CSD representation multiplier circuits presented in lecture are exclusive to unsigned values for X and C. Explain how the circuits would need to change to accommodate signed two's-complement values for X and for C.

Problem 6: Shifter [8 pts]

Using 2-input multiplexers and simple logic gates, if you need them, draw a combinational circuit that is capable of performing both a left shift or a right arithmetic shift. The circuit takes a 4-bit data input value, X, a 2-bit control for shift amount, SA, and a control bit LR. If LR=1 the circuit shifts left, otherwise it shift right.

Problem 7: FSM Design with Counters [15 pts]

Suppose you wish to design the controller for a packet processor in the interface to a wireless network. A message packet comes into your circuitry bit serially, from left to right as defined by the following packet format:

| preamble (128-bits) | header (32 bits) | body (1024 bit) |

The job of your controller is to split off the three fields of the incoming packets, feeding them each into external (to your controller) shift registers to hold each one. The three external shift registers each have a SE (shift enable) input. These shift enable inputs are named `PRE_se`, `HDR_se`, and `BDY_se`, for preamble, header, and body, respectively.

Your block has these inputs: `Din`, `DV` (data valid from the wireless interface, asserted on a per clock cycle basis), and `RST`.

After reset `DV` will be asserted on each cycle where `Din` holds a valid input. The bits of the packet are passed in serially starting with the preamble. Outputs from your block should be the three enable signals.

Design this controller using a FSM and counter(s). You don't need to show the detailed design of the counters, but show the details of the FSM circuitry. To keep the FSM implementation simple, you might want to consider using a one-hot encoded state machine.

Problem 8: Counter Design [15 pts]

(a) Write a Verilog generator that builds an AND-reduction parallel prefix tree for N inputs, where N is always a power of 2. Note that there are many possible structures for parallel prefix. However, for this problem, please implement the the "alternative parallel prefix circuit" shown in the lecture notes under synchronous counters, and also sometimes referred to as "Kogge-Stone" and shown in the lecture on fast adder circuits.

(b) Using that module write the a Verilog generator to define an N-bit binary counter, inputs are CE, CLK, and outputs are Q[N-1:0], and TC.

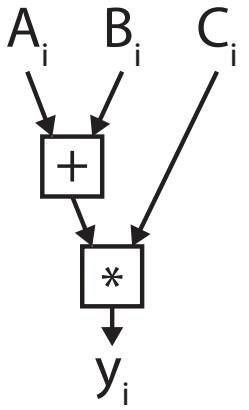
Here are some hints on writing the generator: The compile time function, $clog2 is used to take the log of a parameter. For instance `localparam levels = $clog2(N);` Also, the Verilog compile time operator "**", as in a**b, is equivalent to pow(a, b) in the C language.

Problem 9: LFSR [4 pts]

Draw the circuit for a 16-bit LFSR.

Problem 10: Modulo Scheduling [12 pts]

The following graph represents one iteration of a repeating computation: $y_i = (A_i + B_i) * C_i$. A_i, B_i, and C_i are stored in memory and must be read before they can be used. The result, y_i, must also be stored back into memory. The target hardware platforms for this algorithm include registers that are necessary for pipelining and scheduling. Mem read, write, *, and + each take 1 clock cycle to complete.



Using modulo scheduling, draw the characteristic section schedule with subscripts for the following hardware platforms:

(a) 2 read ports, 1 write port, 1 adder, and 1 multiplier

(b) 3 read ports, 1 write port, 1 adder, and 1 multiplier

(c) 2 read ports, 1 write port, 1 adder, and 1 pipelined multiplier (2 cycles per multiply)

(d) 3 read ports, 1 write port, 1 adder, and 1 pipelined multiplier (2 cycles per multiply)