SYSTEMS I — LAB 4
A 4-bit multiplier

Overview

You are going to build the multipler circuit described in lectures—one that accepts two 4-bit, unsigned integers as inputs and produces an 8-bit, unsigned result. Specifically, this multiplier circuit must be clocked a specific number of times. Therefore, it needs not only a clock, but a clock that will tick a limited number of times before ceasing, requiring a reset before new numbers can be multiplied. Additionally, this multiplier requires one additional clock cycle—an initialization cycle—to load the desired values into the registers.

A counter

As mentioned above, this multiplier is a multi-cycle clocked circuit. It requires one *initialization cycle*, followed by n *calculation cycles*, given n-bit inputs. Therefore, we need a counter to track the $n+1$ cycles required before multiplication is complete. Use a single register chip to build your counter.

Specifically, you can use a different kind of counter for this task, a *ring buffer*:

- Choose an 8-bit 273 register. You will use 5 of the flip-flops for this counter.
- Hard-wire the input to the 1^{st} flip-flop to 1.
- The output of each flip-flop should be connected to the input of the next flip-flop (e.g., $Q1$ should be connected to $D2$).
- Connect the power (V_{cc}), ground (GND), clear ($\overline{CLR}$), and clock (CLK) lines normally.

Clearing this flip-flop will reset the counter, with each memory element emitting 0. After the first clock cycle, the 1^{st} memory element will emit a 1, and the rest will continue to emit 0. The second clock cycle will move the 1 value forward, with the 1^{st} and 2^{nd} memory elements emitting 1 and the rest emitting 0. With each clock cycle, the 1 moves one step further down the memory-element-chain.

Setting up the register

For your implementation of this multiplier, the input will be provided by the eight switches on your board: four bits for the multiplier, and four bits for the multiplicand. Rather than load the multiplicand into a register, you may count in it being persistently provided by those four switches—simply avoid changing the values on those switches in the middle of the multiplication process. An 8-bit product/multiplier register, however, must be initialized to hold four 0's (the initial product) and the four multiplier bits.

Note that your counter will control how this product/multiplier register is managed. Specifically, on the first (initialization) cycle, the initial values should be loaded into this register. On the second through fifth cycles, the input to this register comes from the adder and from the shifting of bits. On **any subsequent cycle**, the clock input to this register should remain 0—that is, the clock signal should not be propagated to this chip.

New and handy chips

For this project, there are two new chips that will be particularly useful:

- `74LS157` **multiplexer**: This chip is a 4-bit multiplexer. Its inputs are two 4-bit values, A and B, that are divided bit-wise into $(A4, A3, A2, A1)$ and $(B4, B3, B2, B1)$. The $\bar{A}/B$ input is the *selector*—an input value of 0 selects the A inputs to appear on the Y outputs $(Y4Y3Y2Y1)$, and a select input of 1 selects the B inputs. Note the $\bar{G}$ *strobe* input: for normal operation, connect it to a permanent 0 input.

- `74LS83A` **adder**: A 4-bit adder. Note that **the pin layout for this chip is substantially different** from others we have used. Power and ground are in the middle, at pins 5 and 12. This chip adds two 4-bit values A and B, represented as $(A4, A3, A2, A1)$ and $(B4, B3, B2, B1)$ respectively, as well as a carry-in bit $C0$, and produces a 4-bit result Σ, which, combined with the carry-out bit, makes a 5-bit output $(C4, \Sigma4, \Sigma3, \Sigma2, \Sigma1)$.

Finishing up

The usual rules for finishing your work apply:

- When done working in the lab, **clean up**. Put away wires, probes, excess chips, wire trimmers, chip pullers, and any other devices that you've used. If you have created garbage by cutting and stripping wire, throw it out.

- To "submit" your work, show it to me. You'll need to demonstrate that your multiplier can correctly multiply any two 4-bit unsigned integer values.

This assignment is due on Friday, October 22, at the start of lab