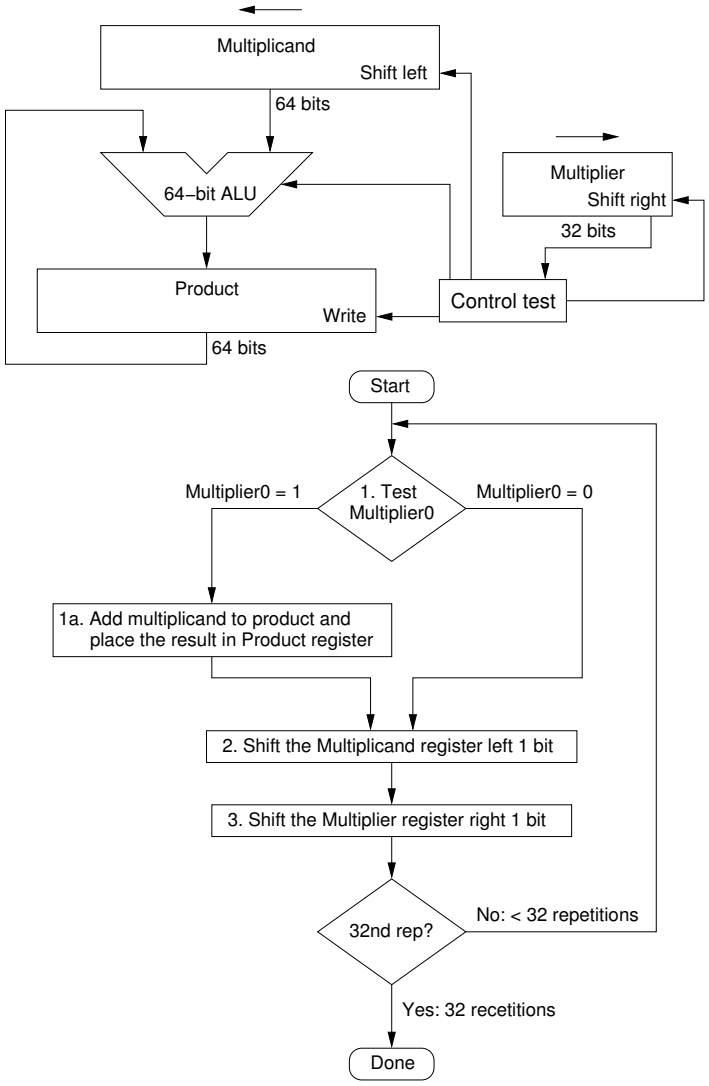
Lab 10: Multiplication Hardware

Submission timestamps will be checked and enforced strictly by the CourseWeb; **late submissions will not be accepted**. Check the due date of this lab on the CourseWeb. Remember that, per the course syllabus, if you are not marked by your recitation instructor as having attended a recitation, your score will be cut in half.

In this lab, we will build an 8-bit multiplication hardware as we discussed in class. The diagram and the flowchart of a 32-bit multiplication hardware is shown below:



Note that the above multiplication hardware is a 32-bit hardware. We are going to build an 8-bit multiplication hardware. So, we need the following components:

1. 16-bit **Multiplicand** register (falling-edge triggered)
2. 16-bit **Product** register (falling-edge triggered)
3. 8-bit **Multiplier** register (falling-edge triggered)
4. 16-bit **Adder**

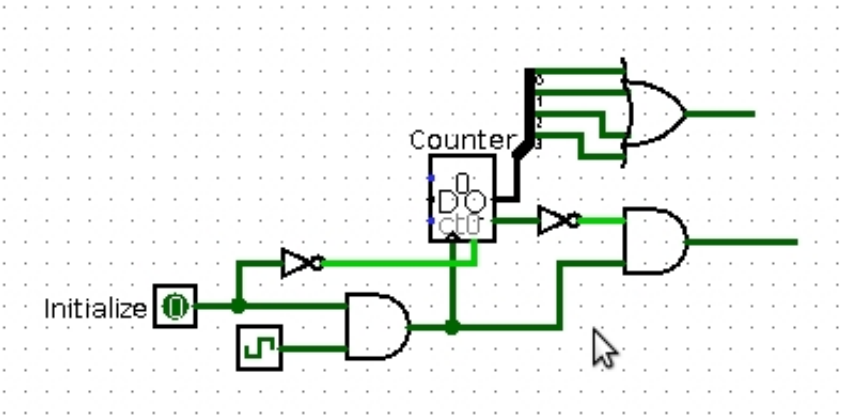
For this lab, simply use pre-built registers and adder from the Logisim.

Multiplication Procedure

Suppose we want to perform $x \times y$ where x and y are 8-bit **unsigned** number, the first step is to perform the following:

- Initialize the top 8-bit of the multiplicand register to 0s and the bottom 8-bit of the multiplicand to x
- Initialize the multiplier register to y
- Initialize the product register to 0

As we discussed in class, this multiplication hardware requires multiple clock cycles. In case of this 8-bit multiplication, we need to perform the total of 8 iterations. At each iteration, we need to perform; (1) update the product register, (2) shift multiplicand register left by 1, and (3) shift the multiplier right by 1. Ideally, we need $8 \times 3 = 24$ clock cycles. However, we are going to perform all three tasks in parallel in each iteration. **Note** that we need one extra clock cycle for initialization the hardware.
First we need to be able to distinguish between the initialization clock cycle and the normal clock cycle of multiplication. To be able to do this, we need a hardware that will be 0 for one clock cycle and the remaining 8 clock will be 1. This can be achieve by using a counter that count up to 9. A counter is a hardware that can be reset, and send a one bit signal when it reached the maximum value. So, first build the circuit as shown below:



The main component in the middle is a counter. Use the counter from pre-built component of the Logisim. Set the attributes of the counter as follows:

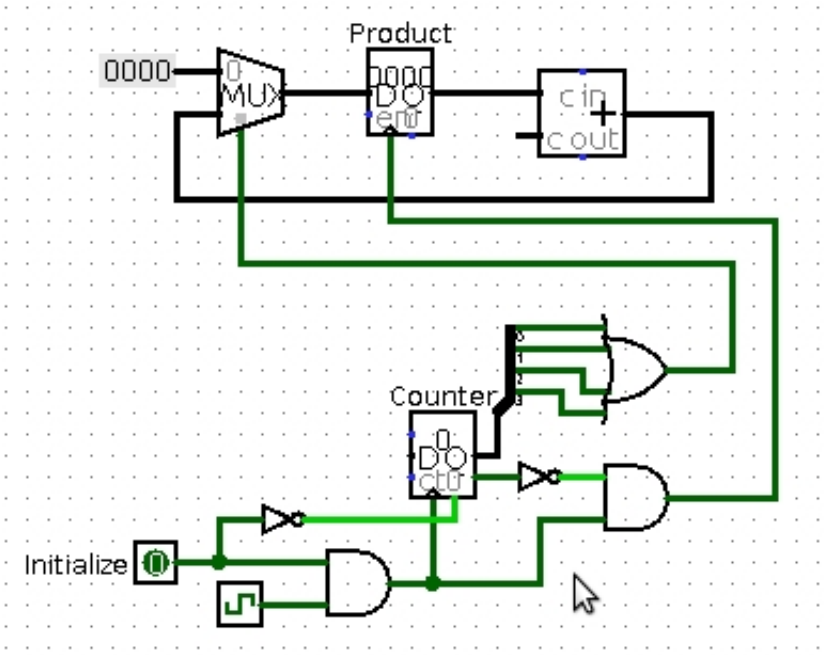
- Data Bits: 4
- Maximum Value: 0x9
- Action on Overflow: Stay at value
- Trigger: Falling Edge

Then put the rest of the components and connect them. Now, let's test it to make sure you built the circuit correctly. If you set the Initialize input 1 and start click the clock using the hand tool, you will see that the output of the OR gate at the top will be 0 during the first clock cycle and then it will change to 1 after the first clock cycle. We can use this signal as an indicator

whether this cycle is an initialize cycle or normal multiplication cycle. The output of the AND gate on the right will behave exactly like a clock but it will stop behaving like a clock as soon as the counter reach 9. The output of this AND gate will be the clock for the whole multiplication cycle (for all three registers) which will maintain values in all registers when the calculation is done.

Note that for this multiplication circuit, before performing a multiplication, a user has to first, set the multiplicand (8-bit input), set the multiplier (8-bit input), and set the Initialize input to 1. Then simply click the clock symbol for 9 cycles (18 times) until the counter reaches 9. Then the result of multiplication will be in the product register.

Now, let's focus on the product register. Recall that we need to initialize this register to 0 at the beginning and the rest of the remaining clock cycle, the product register will be updated by the result from the adder. To be able to initialize it with 0 at the beginning and using the output of the adder for the rest, we use the following circuit:



As you may notice, if the Initialize input is set to 1 (reset it to 0 and set it back to 1) and start clicking the clock, the select line of the multiplexer will be 0 for one clock cycle which allows value 0 to go into the product register. After the first clock cycle, the select line will be 1 which allows the output of the adder to be used to update the product register. Since the clock of the product register comes from the output of the AND gate (on the right), after we reach 9 cycles, the clock input will remain 0 which allows the product register to maintain the result of multiplication.

What to do?

Your job is to complete the multiplication circuit by your self. You need two 8-bit input, multiplicand and multiplier which will allow a user to set to his/her desired value. One 16-bit output for the product that comes directly from the output of the product register. Do not forget that the multiplicand and the multiplier registers must be initialized to their proper values during the

initialization cycle which can be done in the same ways as the product register.

For this circuit, we will not use the control circuit as shown in the above multiplication diagram. The control simply comes directly from the least-significant bit of the multiplier. This will be use whether to perform Product = Product + 0, or Product = Product + Multiplicand. Again, use multiplexer to choose.

Note that you are allowed to use the pre-built shifter for shifting left and right. Be careful with the distance input (the amount to shift). If the shifter is a 16-bit, the distance input should be a 4-bit data. If the shifter is an 8-bit, the distance input should be a 3-bit data. You can use a splitter to extract the least-significant-bit of the multiplier.

Submission

Submit the file `lab10.circ` via CourseWeb before the due date stated on the CourseWeb.