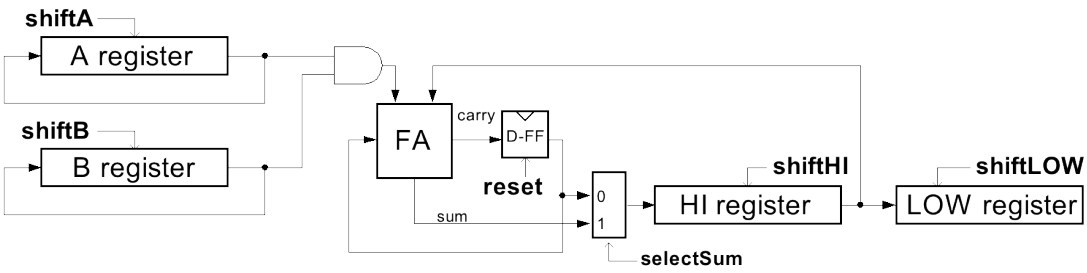
University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Science

EECS150, Spring 2011

Homework Assignment 12: Multipliers and High-Level Design
Due April 28^{th}, 2pm

1. Consider the bit-serial multiplier given in lecture 23, slide 5.

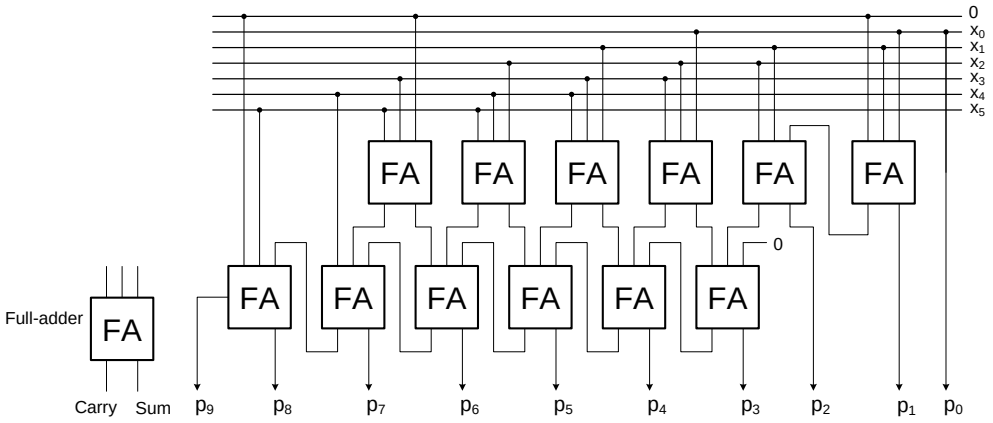


   To better understand this circuit, trace (write down the contents of registers at every cycle) the multiplication of 4'd5 by 4'b9 through a 4-bit bit-serial multiplier.

   Now propose how this multiplier can be extended to handle signed 2's complement multiplication.

2. The circuit shown below is used to multiply the 6-bit number X by a 6-bit constant value, C. It is made up of instances of a full-adder cell. The full-adder takes as input 3 1-bit signals and outputs a 1-bit sum and a 1-bit carry.

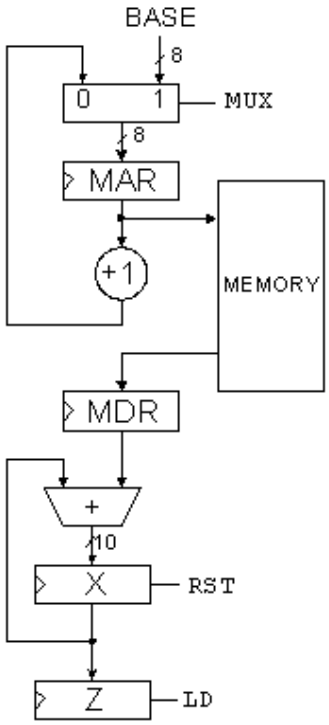
   What is the value of C?



3. Using nothing but instances of full-adder cells, draw a circuit for adding four 3-bit numbers, $w_2w_1w_0$, $x_2x_1x_0$, $y_2y_1y_0$, and $z_2z_1z_0$. First minimize the total delay then the total number of full-adder cells. Label all inputs and outputs.

4. Consider the design of an unsigned combinational multiplier (no flip-flops or controller) for multiplying the unsigned constant value 11012 by the variable X (X3X2X1X0). Using only full-adder cells, draw a circuit that implements the multiplier, minimizing the total amount of hardware and the delay from input to output. **Hint: think about carry-save addition.**

5. Consider the design of a simple processor used to add the contents of blocks of 4 bytes in consecutive memory locations. The datapath circuit for the processor is shown below.

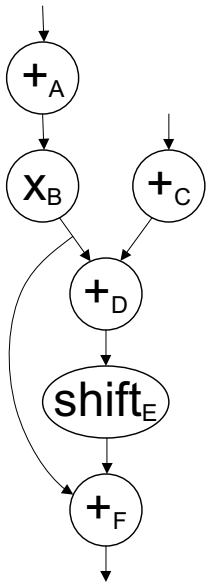


The processor has one data input (8-bit wide) named **BASE**, an input control signal named **ENABLE**, and 3 internal control signals - **MUX**, **LD**, and **RST**. The datapath contains three data registers - **MAR**, **MDR**, and **X**. After the processor performs its operation, the **Z** register is left with the sum of memory locations **BASE**, **BASE + 1**, **BASE + 2**, and **BASE + 3**. We assume that a controller (not shown) will take as input the **ENABLE** signal and generate **MUX**, **RST**, and **LD**. To begin the addition operation, an external circuit asserts **ENABLE** for 1 clock cycle then lowers it for a minimum of 12 cycles.

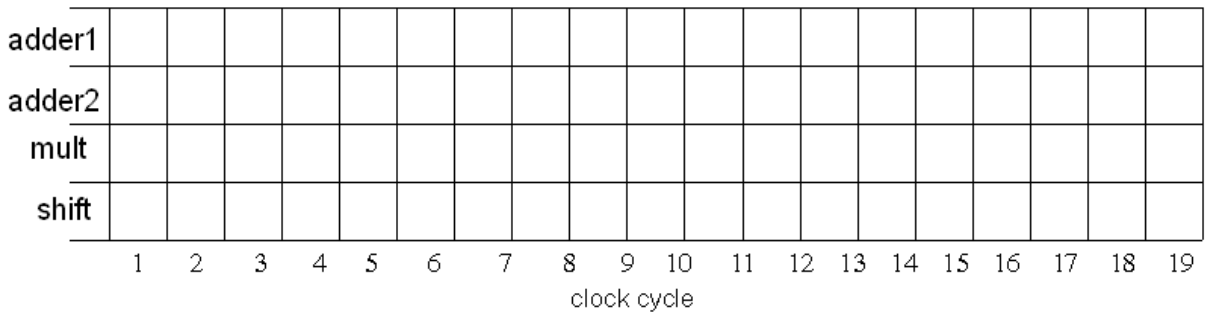
Write the register transfer language level description for the sequence of transfers that must occur after the **ENABLE** signal is asserted. Try to minimize the total number of cycles.

6. Imagine a datapath that has four computation units; two adders, a multiplier, and a shifter. Each unit requires an entire clock cycle (minus flip-flop overheads) to complete its operation and is followed by a register to hold its output. The graph in below represents an iterative operation to be completed on the datapath. Each node is labeled with the name of the computation unit that

it requires plus a unique letter identifying the node. Note that there is no feedback (or loop carry dependence) in this computation.



Use modulo scheduling to show how to complete four iterations of the loop in the minimum number of cycles. Show your work. Then, fill in the chart shown below the unique integer node numbers from the graph. Use subscripts (1, 2, 3, and 4) to indicate the iteration number. For instance, "C_2" indicates node C of iteration 2.



7. Consider the design of a special processor connected to a dual-ported memory, shown below (next page). In the memory is stored an **array** of 8-bit integers, starting at address 0. When started, the processor begins at location 0 and moves through memory forming the sum of all the integers up to that point, storing the sum in each memory location as it goes. The process continues for the entire array.

   An input signal call **`START`** is used to start the process, and an input called **`ENDADDR`** is used to specify the address of the final element in the array.

Write the register transfer language description of the processor operation and draw the design of the processor datapath.

- Use only the following circuit elements:
  - binary adder(s) of any width,
  - register(s) with reset and load-enable,
  - equal-comparator(s),
  - and the memory show below. The memory has *asynchronous read* and *synchronous write* operations.
- In your design, minimize the processor cycle time and the number of cycles in the inner-loop.
- Remember to use a comma, ",", to separate RTL operations that occur on the same clock cycle, and the ";" to seperate operations on different cycles.