EECS150: Homework 13, RTL, Scheduling, and Registers

UC Berkeley College of Engineering
Department of Electrical Engineering and Computer Science

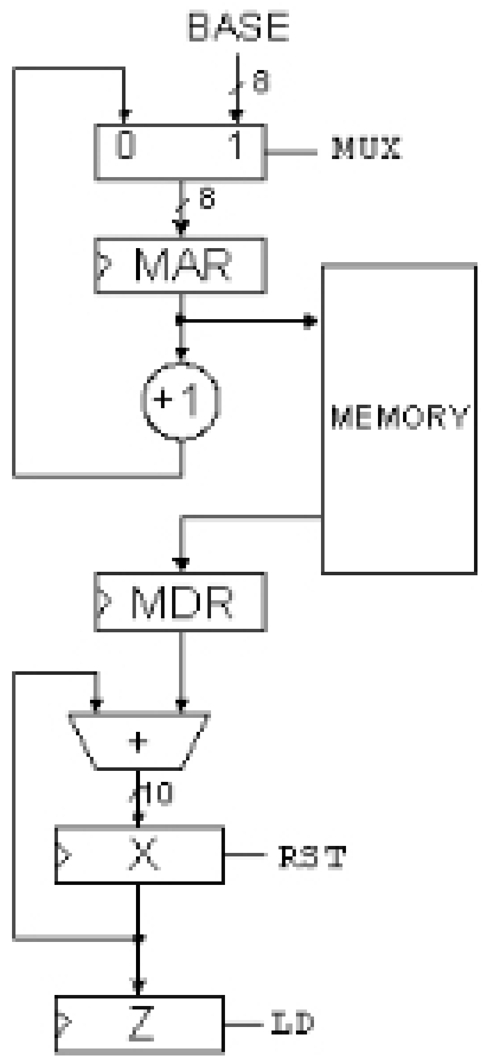
1 Time Table

ASSIGNED	Friday, April 24^{th}
DUE	Friday, May 1^{st} at 2pm

Homework submission will only be through SVN. Email submissions will not be accepted!

1. 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 in Figure 1.

Figure 1 Datapath for the processor used to add the contents of blocks of 4 bytes in consecutive memory locations.

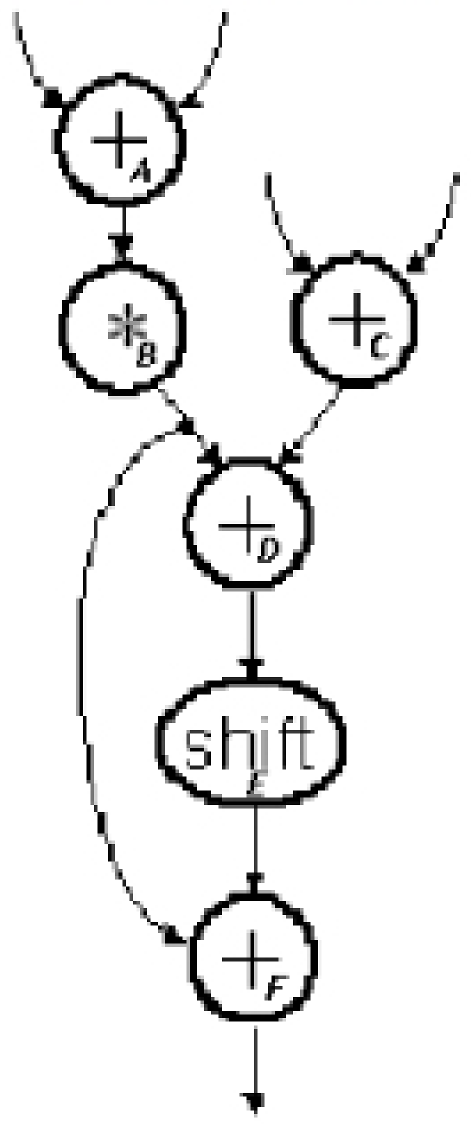


The processor has one data input (8-bit wide) named **BASE**, an input control signal named **ENABLE**, and 3 internal control signals - **MUX**, **LD**, and **RST**. The datapath contains three data registers - **MAR**, **MDR**, and **X**. After the processor performs its operation, the **Z** register 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 RTL level description for the sequence of transwers that must occur after the `ENABLE` signal is asserted. Try to minimize the total number of cycles.

2. 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 Figure 2 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.

Figure 2 Graph of the operations needed for each iteration.



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 in Figure 3 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.

3. Modify the D-type flip-flop given in Figure 4 to add a **synchronous** `Reset` and a **synchronous** `Set`.

4. Modify the D-type flip-flop given in Figure 4 to add an **asynchronous** `Clear` and an **asynchronous** `Preset`.

5. Building flip-flops out of other kinds of flip-flops.

   (a) Sketch a circuit for a D-type flip-flop based on a JK-type flip-flop.

   (b) Sketch a circuit for a JK-type flip-flop based on a D-type flip-flop.

   (c) Sketch a circuit for a T-type flip-flop based on a D-type flip-flop.

6. Draw the circuit for a 5-bit LFSR with the following primitive polynomial: $X^5 + X^2 + 1$.

Figure 3 Fill this in with how you would schedule the operations needed to complete 4 iterations.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
adder1																			
adder2																			
mult																			
shift																			

clock cycle

Figure 4 A D-type flip-flop implemented using cross-coupled NAND gates.