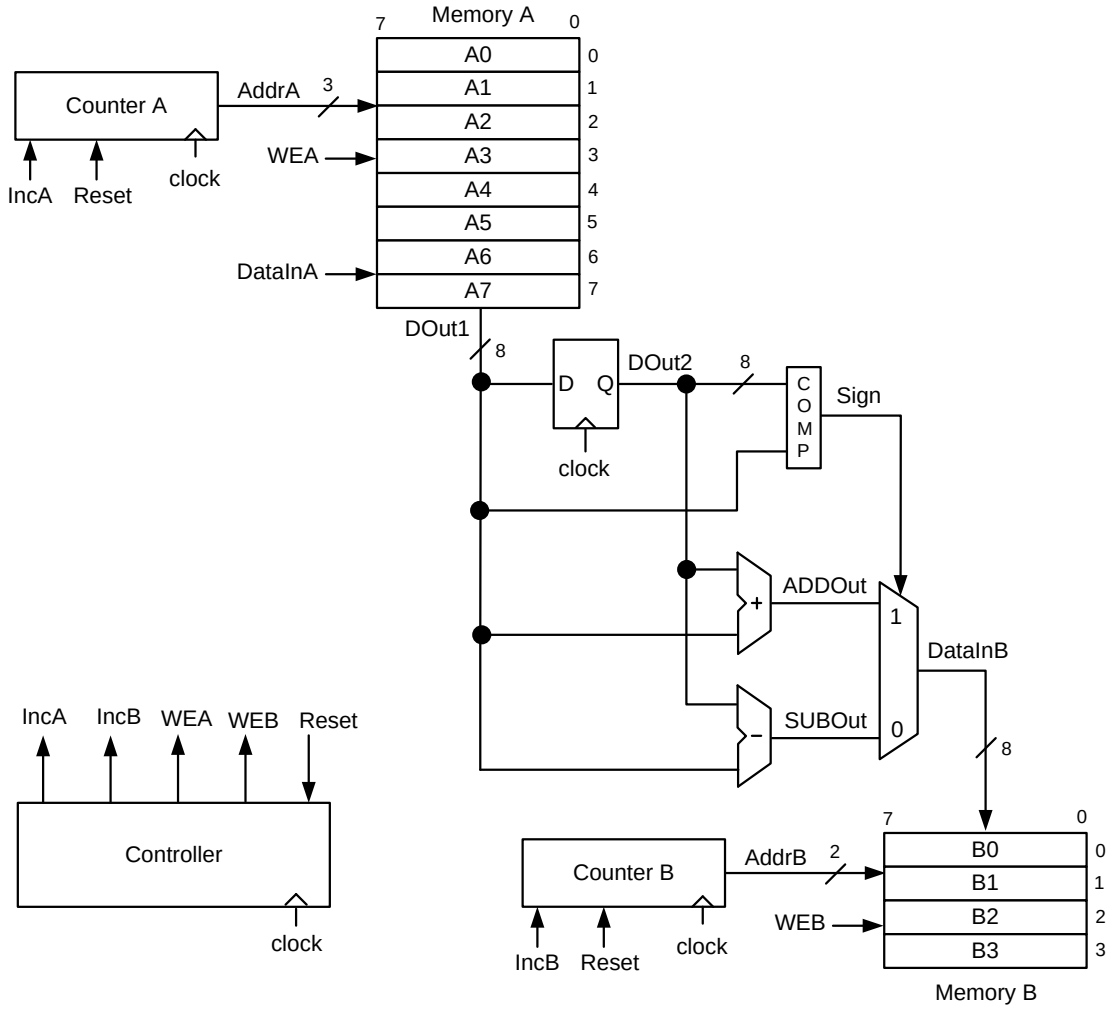
Computer Engineering 240 – 02

TERM Design Assignment 1

Spring 2017

1. The aim of this assignment is to construct the Memory to Memory transfer design described in section 2-13 (page.102) of the assigned course book. All the details of the assignment and the desired output is described below as well.
2. Design and construct the data-path shown below the functionality for which is described ahead.

Memory-Memory Transfer Data-Path with Controller

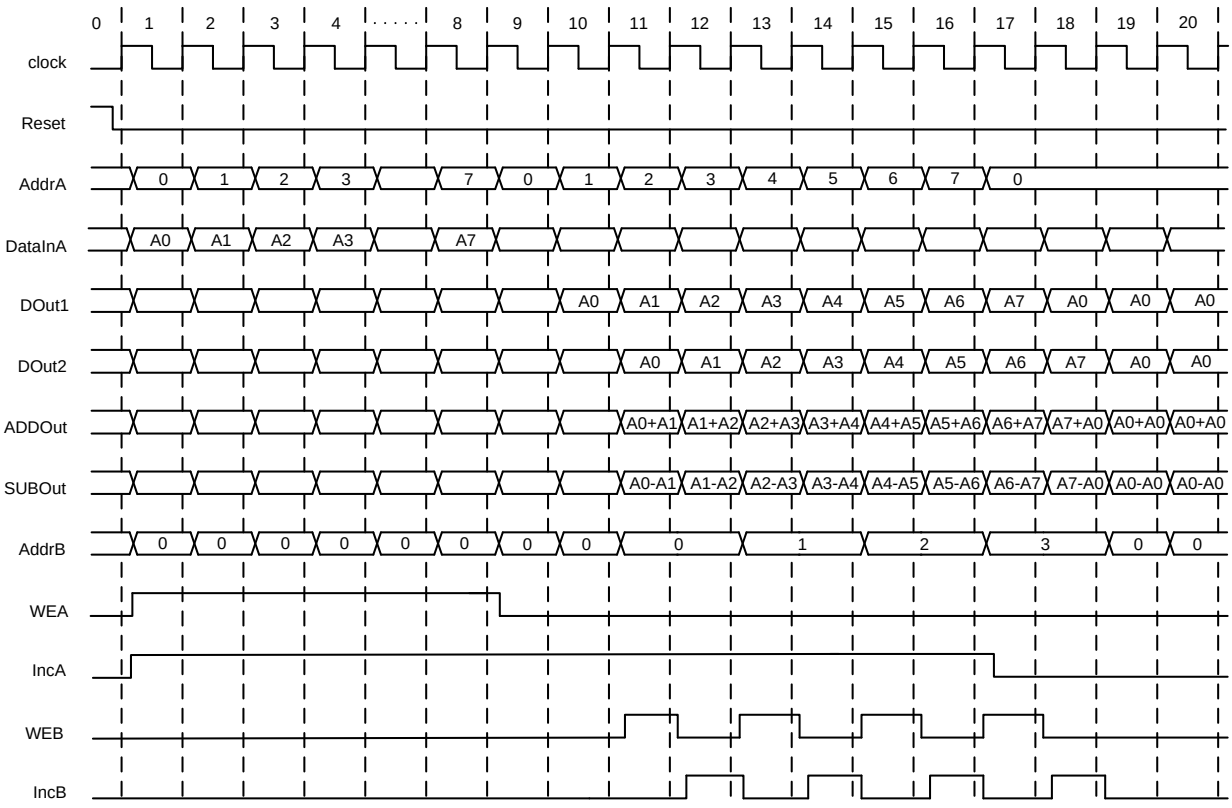


The datapath consists of the following constructs:

1. Datapath controller which is a state machine of sorts.
2. Counters to generate write addresses to memory.
3. 2 Memory macro's – Memory A which is 8 Bits wide and 8 entries deep. Memory B which is 8 Bits wide and 4 entries deep.
4. Random logic to determine how data gets compressed and transferred from Memory A to Memory B.

The design requires that the controller first works to populate Memory A with data provided for through a testbench. Then the data is read out sequentially and compared in pairs. Each pair (A,B) is compared. If the sign of the comparison is positive, **A+B** is stored in Memory B. Else, **A-B** is stored in Memory B. Following this the next pair is compared and stored. To understand this process further, consider the following timing diagram which explains the operation.

Memory-Memory Transfer Timing Diagram – Address, Data and Controls



When the controller comes out of reset, for the first 8 cycles it populates Memory A by asserting **WEA** and at the same time generating incrementing addresses from the counter by asserting **IncA** to the counter. **DataInA** is an input to the system which will be driven by the test bench with the timing of the controller kept in mind. Once the Memory A is populated, the controller de-asserts **WEA** (which signals to the Memory to start a read operation. At this point, Memory begins to write data to **Dout1** in successive cycles based on addresses again generated by the controller. The controller keeps asserting **IncA** through this process, once the write operation is complete, since this is 3 bit counter (counter A) it wraps around and starts counting from 0 again. Therefore, Memory A starts reading from address 0 after write is done.

The aim is to compress a pair of entries from Memory A to Memory B. This requires:

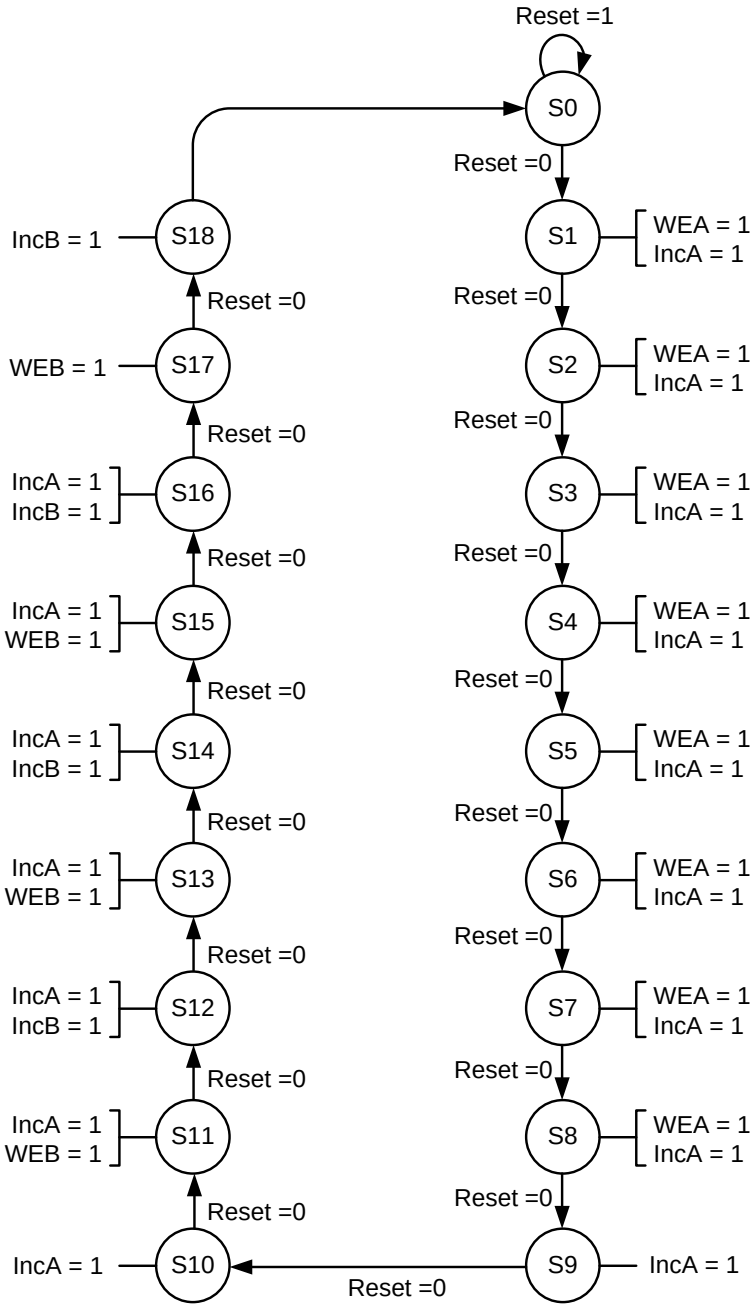
1. MemoryA[0] , MemoryA[1] => MemoryB[0]
2. MemoryA[2] , MemoryA[3] => MemoryB[1]
3. MemoryA[4] , MemoryA[5] => MemoryB[2]
4. MemoryA[6] , MemoryA[7] => MemoryB[3]

The controller is designed with this consideration and as is evident in the timing diagram, it asserts **WEB** and **IncB** only every alternate cycle to give enough time to have 2 addresses read out from Memory A.

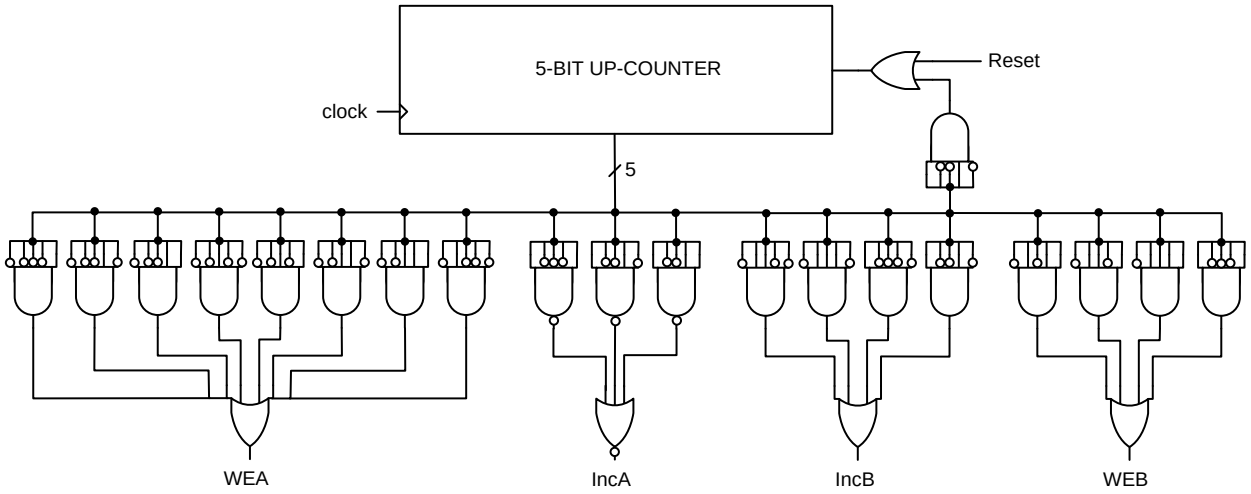
If the controller is designed as a Moore State Machine, the state transition diagram is expected to look like the diagram in the following page.

The following is what is expected as a result of this assignment:

1. **Construct the entire design in Verilog and test it by driving it through a testbench.**
2. **State all assumptions you've made for the design.**
3. **The controller can be designed as a Moore Machine, Mealy Machine or a simple counter implementation. Describe what the design is in your report and why you chose to design it in that way.**
4. **Make sure the design works with both Negative and Positive numbers.**
5. **The system should be able to properly reset at the end of operation and restart if required.**
6. **The testbench should be easily modifiable for the instructor to try other sets of data.**
7. **Submit your entire code base in a zipped folder with the name of the folder being: [FirstName]-[Last 4 Digits of Student ID]. This will be submitted on Canvas.**
8. **Submit a report in class which describes:**
    a. **Your design assumptions.**
    b. **Your test cases**
        i. **Why you think your test cases are exhaustive**
    c. **The design of the control logic**
    d. **Waveform of a complete simulation with explanation of events.**
    e. **If the timing of your waveform looks different from what is expected explain why and why you were not able to correct the issue.**
9. **As a further restriction, the logic element 'COMP' which is the comparator that compares value A and B should NOT be designed using '>' '<' operators.**
10. **The assignment is due 2/23/2017 and is worth 10% of your total grade. Every day after the submission day will cause a 20% deduction from the maximum available grade for the work.**



This is a Moore representation of the controller. A counter based design is shown on the next page.


5-BIT UP-COUNTER
clock
Reset
5
WEA
IncA
IncB
WEB