Massachvsetts Institvte of Technology

Department of Electrical Engineering and Computer Science

6.313 Contemporary Computer Design

Problem Set #4 Fall, 1998, Due Wed 4 November

1.) A number of processors are connected to a single bus. The bus is shared using the following protocol:

- On a given clock cycle, exactly one processor is bus master. Only that processor is allowed to write to the bus.
- Each processor generates a 1 bit signal REQ which is high if that process or wants the bus on the next clock cycle.
- The processors are arranged in a ring. On a rising clock edge, list the processors starting with the bus master and moving clockwise. The first processor in the list which is asserting REQ becomes bus master for the next clock cycle (in particular a bus master which is asserting REQ will not lose its status). If no processor is asserting REQ, then the current bus master remains bus master for the next cycle.

Your job is to design a per-processor bus controller to implement this protocol. Each controller contains one bit M of state which is high on a given cycle if the processor is bus master during that cycle. The inputs to the contoller consist of the processor-generated REQ bit and a signal P_{in}, containing one or more bits, which comes from the previous contoller in the ring. The controller must generate M_{next}, which determines whether or not the processor will be bus master on the next cycle, and P_{out} which is sent to the next controller in the ring (P_{out} of one controller connects to P_{in} of the next).

a) How can you define the P_{in}/P_{out} signal so that it can be used to implement the given protocol? How many bits do you need to use?

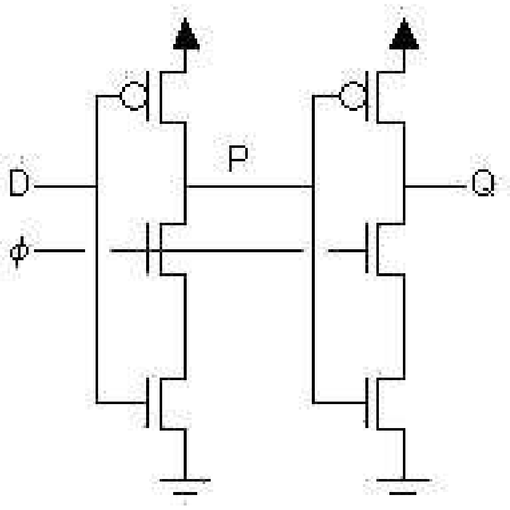
b) Give logic equations for generating M_{next} and P_{out}.

c) Draw the complete circuit for a single controller using NAND/NOR/NOT gates and a single positive edge-triggered flip-flop. Try to minimize the number of logic gates used. You may assume that the controllers are magically powered up to a state where exactly one controller has its M bit set.

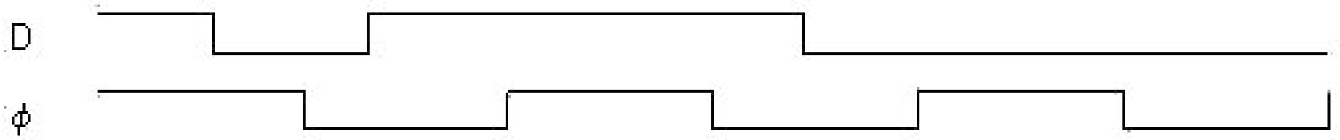
d) When the controllers are connected in a ring, what is the critical path in the circuit? Explain why this is a bad design if there are a large number of processors and speed is an issue.

e) Show how to make the critical path logarithmic in the number of processors by modifying the controllers and adding a modest amount of circuitry outside of the controllers. You may assume that the number of processors is a power of 2 (this is a hint).

2. The following diagram shows a Svenson latch:



a) Draw waveforms for P, Q given the following input waveforms for D and the clock:



b) Show how to use two Svenson latches to construct a positive edge-triggered flip-flop (draw the complete circuit).

3. You need to implement a fast 2048 bit shift register, and have a choice of available storage elements and clocking strategies. The basic gate delay is 100 ps. Fall time and rise times of typical logic signals are also 100ps. You have determined that clock skew between different locations can be held to 150ps with careful wiring and clock distribution, but that clock skew between multiple clock phases, if used, cannot be controlled to better than 250 ps.

a) Design a single phase clock waveform shape and a single dynamic latch memory structure which will correctly implement the shift operation at maximum speed. What is the maximum speed? What constraints must be obeyed in the single phase clock methodology, and how does your design address these constraints?

b) Design a two phase non overlapping clock waveform pair and a dynamic latch pair memory structure which will correctly implement the shift operation at maximum speed. Again, what is the maximum speed? What constraints exist now, and how does your design address these constraints?

c) Design single phase clock waveform and a positive edge-triggered flip flip memory structure which will correctly implement the shift operation at maximum speed. Once again (and for the last time) what is the maximum speed? What constraints exist, and how does your design address these constraints?

4. In two's complement integer arithmetic, we don't have to worry about negative numbers when designing adders; the adders "just work" when we feed them negative numbers. Is the same true for multipliers? Justify your answer.