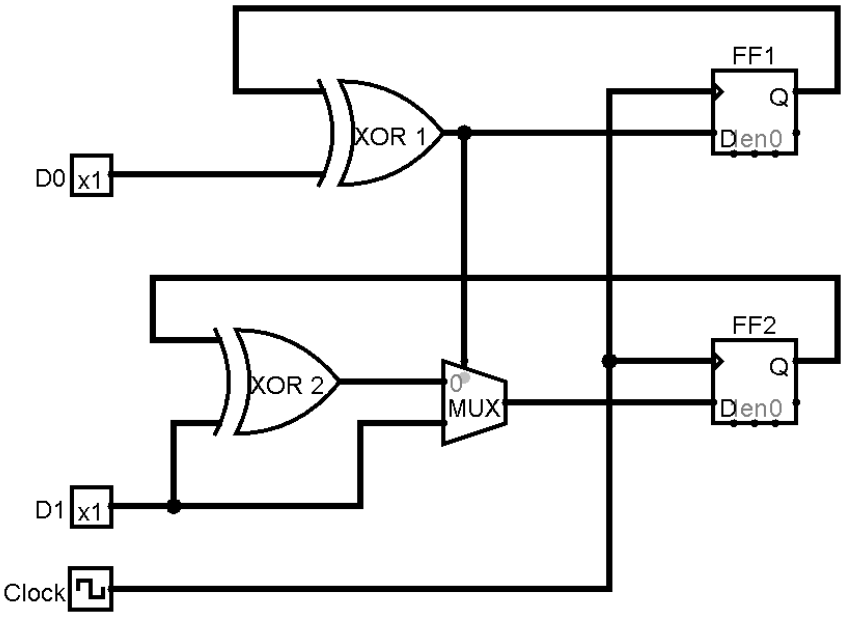
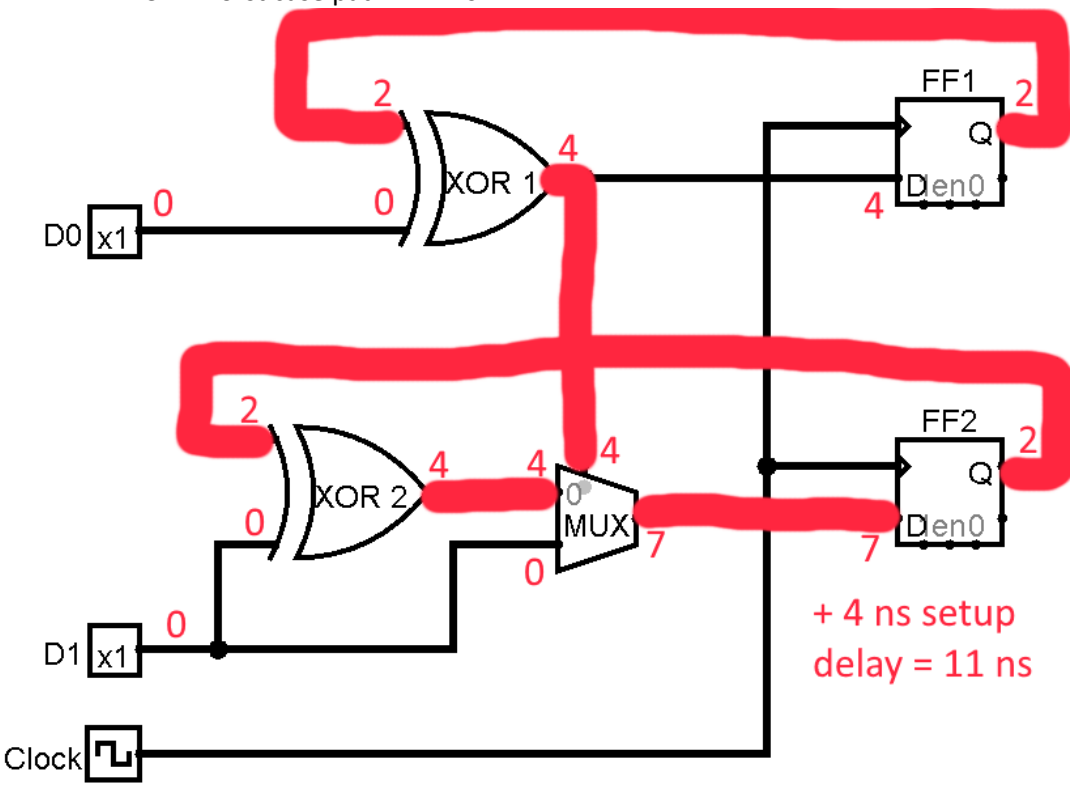
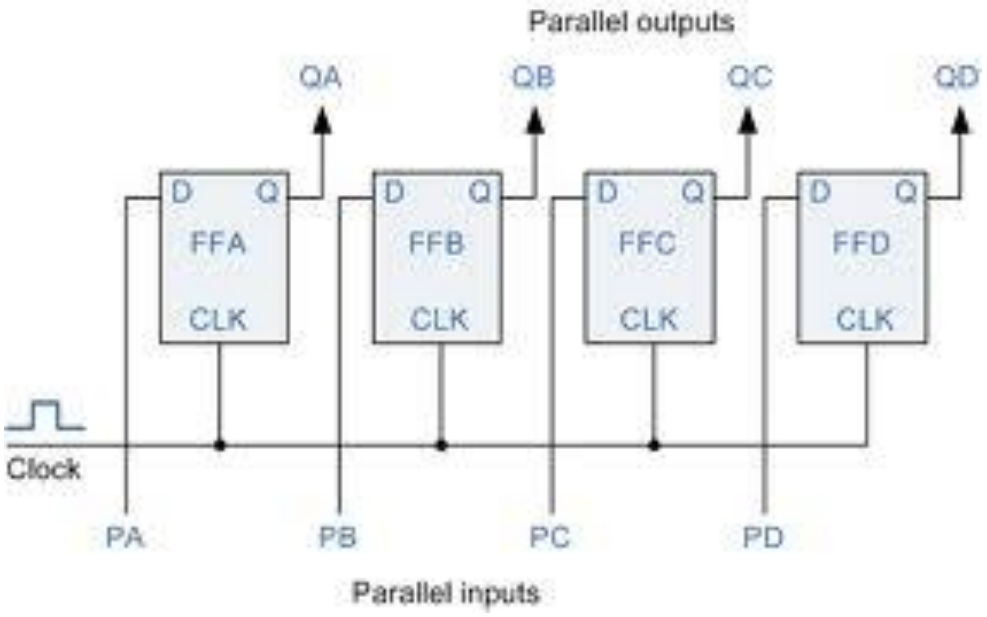
1. Flip flop timing
    a. Remember that in sequential circuits, the current state serves as an input
    b. This means that in worst case path diagrams, flip flops can serve as both a source and final destination
        i. Before, in combinational circuits, only original inputs into the circuit were the start of a path
        ii. Flip flops cannot be in the middle of a worst-case path
            1. Their values only change on a clock edge
            2. Worst case paths assume only one clock cycle, since we use them to determine clock time
    c. Two main times to consider for flip flops
        i. Setup time
            1. If flip flop is a destination, input needs to hold constant
            2. Add setup time as very last term for worst case path
        ii. Flip flop propagation delay
            1. If the flip flop is a source, then need to wait for flip flop to give output
            2. Propagation delay becomes first term of worst case path
        iii. Hold time – not important
            1. Not concerned with input being constant for these problems
    d. Keeping track of times in larger circuit diagrams
        i. Inputs start at time 0
            1. Write a 0 right next to the inputs
            2. For every gate that uses that input, write a 0 on that gate's input
        ii. Flip flops start at their propagation delay
            1. Write the propagation delay next to the flip flop output
            2. For every gate that uses the flip flop value, write the delay on that gate's input
        iii. Calculating delays for each gate
            1. Once all inputs have times for a gate, take the largest of the times
            2. Add the current gate's delay and write the sum on the output of the gate
            3. Write the new sum on all gate inputs that use this output
    e. Example
        i. Without markings
            1. DFF setup delay = 4 ns
            2. DFF propagation delay = 2 ns
            3. XOR gate delay = 2 ns
            4. MUX gate delay = 3 ns



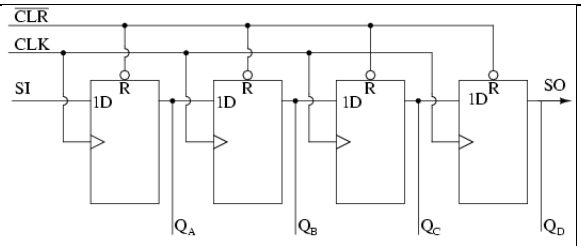
ii. Marked up solution
1. Worst case paths in red
2. Two different worst-case paths
   a. FF2 propagation (2 ns) -> XOR 2 (2 ns) -> MUX (3 ns) -> FF2 setup (4 ns)
   b. FF1 propagation (2 ns) -> XOR 1 (2 ns) -> MUX (3 ns) -> FF2 setup (4 ns)
3. Worst case path = 11 ns



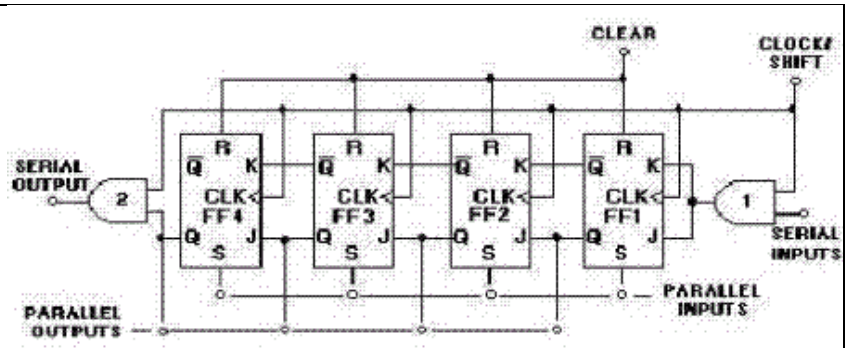
2. Registers
   a. Set of flip-flops used to store *n* bits of information
      i. Common clock used for each flip flop in the register
   b. Parallel register – set of 1-bit memories that can be read or written simultaneously
      i. Tend to use D flip flops, but can use others with some extra logic



c. Shift register – register that accepts and transfers data serially
   i. Already seen these in lab 1 – takes previous values and shifts them over in register
      1. Equal to a queue – first in, first out or FIFO
      2. 4-bit example: push values into left side, values removed from right side
         a. xxxx (to start)
         b. 1xxx (input 1)
         c. 01xx (input 0)
         d. 001x (input 0)
         e. 1001 (input 1)
         f. 0100 (input 0)
   ii. Interface with serial I/O devices, delay signals, do logical bit shifts in ALU
   iii. Main idea – take previous FF value and pipe into current FF



Serial-in/ Parallel out shift register details



Shift register with variety of function modes