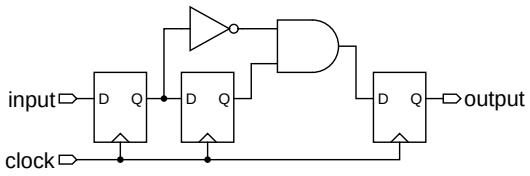
# Assignment 1

*Due Monday, April 25. Show your work. Submit your assignment using the appropriate dropbox on the course web site. Assignments submitted after the solutions are made available will be given a mark of zero.*

**Note:** When answering the following questions you must follow the course restrictions on writing VHDL. In particular, the restrictions on process statements and registering outputs. Use `std_logic[_vector]` and `unsigned` types.

## Question 1

(a) Write the VHDL that would generate the following schematic:




You may place the assignments for the three flip-flops within the same `process()` statement.

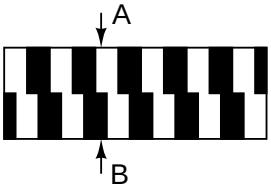
(b) Write a version using a conditional assignment instead of an `xor` operator and draw the corresponding schematic.

(c) What might be the purpose of this circuit?

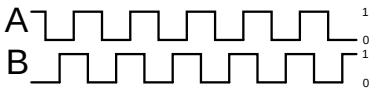
## Question 2

A quadrature encoder consists of a pair of sensors that alternatively output high and low as the sensors move. The sensors are arranged in a way that allows the direction of motion to be determined.

For example, if the arrows below represent the location of two sensors and the outputs A and B are high when the arrows touch white:




then as the sensors move toward the right along the track the following two waveforms would be generated:




so that A goes high before B. Conversely, if the sensor is moving to the left A goes high after B.

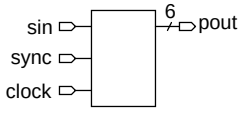
(a) Design a state machine that can determine the direction of motion of the sensors (left or right).

(b) Write the corresponding VHDL. Your entity should have two inputs (`A`, `B` and `clock`) as described above and one output (`right`). The output should be high when the most recent motion detected has been towards the right and low otherwise.

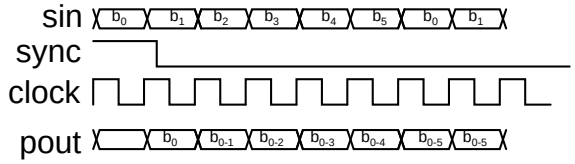
Assume the clock is fast enough that you samples `A` and `B` multiple times when they are high or low. **Do not** use `A` or `B` as a clock.

## Question 3

Design a circuit that converts a serial stream of data to parallel format. The circuit has three one-bit inputs: `sin`, a serial data input; `sync` a signal that is high to indicate the start of a new word; and a `clock`. There is one 6-bit output, `pout`:




On each rising edge of `clock` the value of `sin` is copied to one of the eight bits of `pout`. `sin` is copied to bit 0 (the least-significant bit) if `sync` is high. Otherwise it is copied to successively more-significant bits of `pout` on each clock. As shown the by the following timing diagram:




if all bits have been filled, the values on `sin` are ignored. If `sync` is asserted before all bits have been copied, the values from the previous word are preserved.

(a) Draw the state transition diagram. Note that the bits of `pout` cannot be used as state variables because they are unpredictable.

(b) Write VHDL code that implements the required functionality and draw the corresponding schematic.