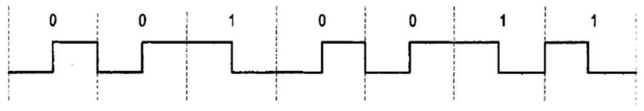
Lab session #8

Activity #1: Manchester encoding circuit
In telecommunication and data storage, Manchester coding (also known as phase encoding, or PE) is a code in which the encoding of each data bit is either low then high, or high then low, of equal time. It is self-clocking, which means that a clock signal can be recovered from the encoded data.



As an encoded bit includes a sequence of "01" or "10", two clock cycles are needed. Thus, the maximal data rate is only half of the clock rate. There are two opposing conventions for the representation of data:

1. The first of these specifies that for a 0 bit the signal levels will be low-high with a low level in the first half of the bit period and a high level in the second half. For a 1 bit the signal levels will be high-low (G.E. Thomas convention)
2. The second convention states that a logic 0 is represented by a high-low signal sequence and a logic 1 is represented by a low-high signal sequence (IEEE 802.3 convention).

Design a **Manchester encoder** (Moore machine) with 2 inputs:
- *data* is the serial line
- *valid* indicates a valid data on *data*

and 1 output *output*: it is set to '0' whenever *valid* is '0', otherwise it encodes *data* according to the first convention.

Resort to a VHDL behavioural description style with 2 processes and a synchronous reset. Create a suitable testbench to test the Manchester encoder on relevant values reading inputs from file.

Activity #2: Programmable one-shot pulse generator
A one-shot pulse generator is a circuit that generates a single fixed-width pulse upon activation of a trigger signal. We assume that the width of the pulse is programmable by the user in the range between 1 and 7 clock cycles. The detailed specifications are listed below:
- There are two input signals, *go* and *stop*, and one output signal, *pulse*.
- The width of the pulse can be programmed as follows:
  - The *go* and *stop* signals are asserted at the same time to indicate the beginning of the programming mode
  - The desired value is shifted in via the *go* signal in the next three clock cycles.
- The *go* signal is the trigger signal that is asserted for only one clock cycle. During normal operation, assertion of the *go* signal activates the *pulse* signal for the number of clock cycles programmed at the beginning of operation. If the *go* signal is asserted again during this interval, it will be ignored.
- If the *stop* signal is asserted during this interval, the *pulse* signal will be cut short.

Design a FSM-D that implements the above behaviour.

Steps:
1. Draw the HLSM state diagram and write the VHDL code to capture the HLSM behavior (cfr. VHDL: Embedded Systems RTL Design, capture HLSM behavior)

2. Design a FSM-D:
    a. creating 2 processes for behavioural datapath description from the HLSM state diagram (cfr. VHDL: Embedded Systems RTL Design, describing a datapath behaviourally)
    b. create 2 processes for behavioural controller description from the HLSM state diagram (cfr. VHDL: Embedded Systems RTL Design, describing the controller behaviourally)
    c. connecting datapath processes to controller processes to create a single architecture (cfr. VHDL: Embedded Systems RTL Design, connecting controller and datapath).

Create a suitable testbench to test the circuit on relevant values.