Lab session n°1
Manchester encoding and decoding

Objective:

This lab session deals with a Manchester encoder-decoder for an asynchronous serial link. The system is implemented using the VHDL system description language and simulated using ModelSim software.

Needed software before starting:

Before starting, make sure that you installed the following software:

Quartus web edition:
https://drive.google.com/file/d/15qiVYJshO44PTmFX9qh7CSj0n_6QvCbw/view?usp=sharing

ModelSim:
https://drive.google.com/file/d/1g9sIK2wolsrnOqHnn3Ja843XD9-qiUpj/view?usp=sharing

cyclone III device:
https://drive.google.com/file/d/1qd5EpNJ20hg8O8dx6lb0W_31Z9MVMNzw/view?usp=sharing

cyclone V device:
https://drive.google.com/file/d/1urs77Siz9ZugpXsz2wT8MasC6myIqITC/view?usp=sharing

This tutorial may help you to install Quartus:
https://www.youtube.com/watch?v=RVPuHfK-EBM&feature=youtu.be

Development environment:

- The Notepad++ text editor to write VHDL code (not Altera Quartus).

- Altera ModelSim for the simulations.

- For each part of the lab session, code skeletons (**me.vhd** and **md.vhd**) are provided where certain fields identified by (?) must be filled in. The test benches are also provided and must not be changed (**me_tb.vhd** and **md_tb.vhd**).

Preliminary questions:

1) Present, in two sentences, the OSI (Open System Interconnection) model and its utility.

2) Present, in one sentence, the usefulness of each layer of the OSI model.

3) Present some protocols of the Layer 2 of the OSI model.

4) Present, in two sentences, each of the following IEEE physical layer protocols: 802.3, 802.9, 802.11, 802.15.

5) What is the difference between UART, USART and RS-232? (in one sentence).

6) Explain the format of the UART frame.

7) On the webpage https://learn.sparkfun.com/tutorials/serial-communication/uarts you will find a block diagram of a UART interface. Explain in one sentence the function of each block.

8) We consider the binary data: 010011100101010011.

    a. Trace the unipolar and bipolar NRZ coded sequence
    b. Trace the unipolar and bipolar Manchester coded sequence
    c. Compare the 4 obtained sequences

Part A: Encoder

The objective of this part is to develop and simulate a UART encoder using single-pole Manchester coding. We are interested in the physical layer and the link layer of the OSI model.

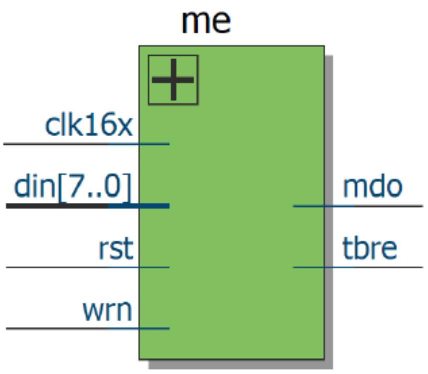


Figure 1: UART encoder

din	8-bit data input
rst	Reset
wrn	Write enable
clk16x	System clock
mdo	Manchester Data output
tbre	Transmission buffer register empty

Table 1: Definition of I/O signals

The code consists of several *process* designed to implement the various blocks you discussed in the preparation. In the following, we will guide you through the implementation of each of the *process*.

1) Process 1 and 2

The objective of the first two processes is the management of the transmission clock. Figure 2 shows the generated RTL (Register Transfer Level) by these two processes.

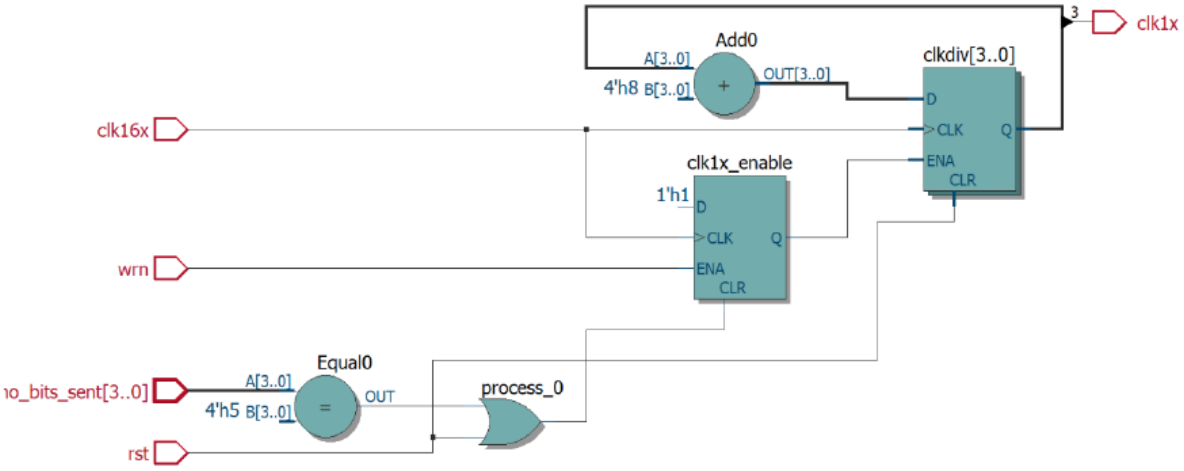


Figure 2: RTL level of transmission clock management

clk1x	Transmission clock
no_bits_sent	Number of transmitted bits
clk1x_enable	Activation of clk1x
Clkdiv	Counter for frequency divider

Table 2: Definition of intermediate signals

Be careful: « 4'h5 » is a literal writing equivalent to a **std_logic_vector (* to *)**

4'	h	5
Number of bits	Base	value

So, « 4'h5 » means "0101" under the convention **std_logic_vector (* to *)**
However, our signals use the convention **std_logic_vector (* downto *).**
Hence, « 4'h5 » means "1010" = 10 !

2) Process 3 and 4

The objective of these two processes is the management of the data to be transmitted. First of all, the data received on the parallel bus are put in a transmission buffer register and then serialized. Figure 3 shows the RTL level generated by these two processes.

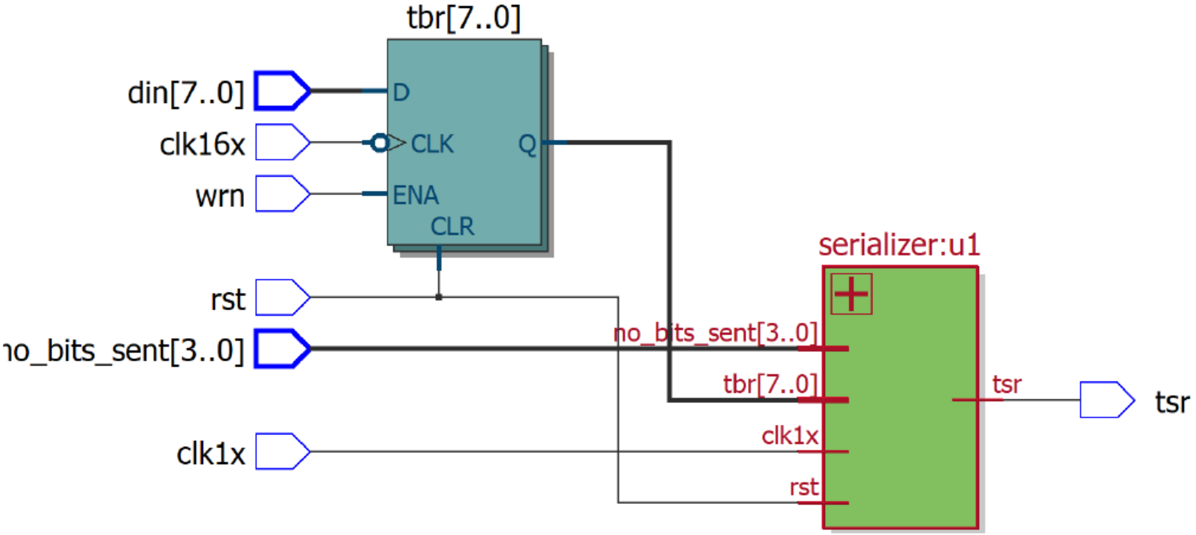


Figure 3: RTL of transmitted data

tbr	Transmission buffer register
tsr	Transmission shift register

Table 3: Definition of intermediate signals

3) Process 5

The purpose of the last process is to calculate the number of bits sent. Figure 4 shows the RTL level generated by this process.

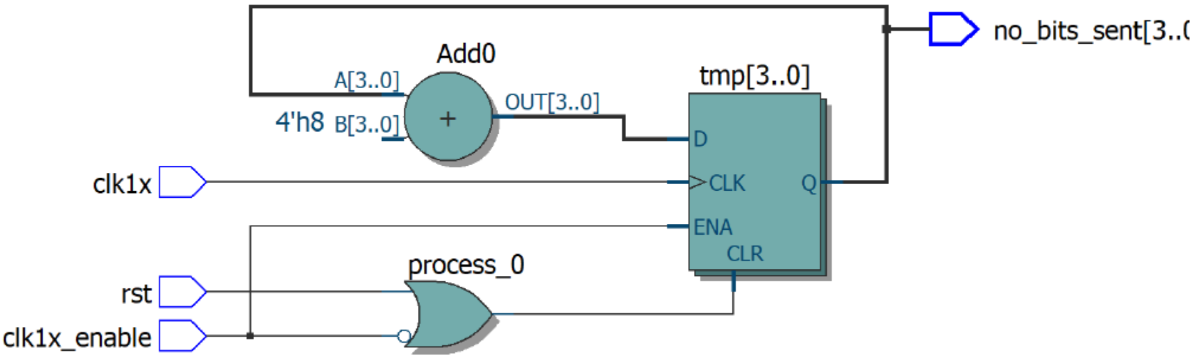


Figure 4: RTL of transmitted bits' counter

tbr	Transmission buffer register
tsr	Transmission shift register

Table 4: Definition of intermediate signals

4) Questions:

1- For figures 2, 3 and 4, write the algorithms to calculate the outputs as functions of the inputs.

2- To transmit a 2-bit frame, how many clk16x clock cycles are required?

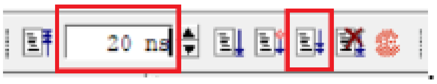
3- What is the purpose of the clk1x and clk1x_enable signals?

4- Propose a simple combinatorial expression to calculate the code Manchester.

5- Complete the skeleton file me.vhd

6- Download the testbench me_tb.vhd file and put it in the same directory. This file contains stimuli to run a simulation.

7- Launch the ModelSim software, compile in the order me.vhd and me_tb.vhd. Set the simulation period to 20 ns and start a full simulation with the Run-all button.



8- Using the zoom tool, explore the different signals and comment on the results.

9- On the Manchester coded signal, we can notice the presence of parasitic impulses ("glitch"). Give an explanation to these interfering signals. Propose a simple solution to filter these parasitic impulses (not to be implemented).

Part B: Decoder

The objective of this part is to develop and simulate a UART decoder using unipolar Manchester coding. We are interested in the physical layer and the link layer of the OSI model.

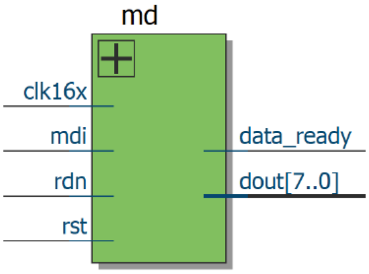


Figure 5: UART decoder

clk16x	System clock
mdi	Unipolar Manchester Data input
rdn	Read enable
rst	Reset
data_ready	Achieved decoding
dout	Useful data

Table 5: Definition of I/O signals

The code consists of several *process* designed to implement the various blocks you discussed in the preparation. In the following, we will guide you through the implementation of each of the *process*.

1) Process 1, 2 and 3

The objective of the first three processes is the management of the reception clock. Figure 6 shows the generated RTL by these three processes.

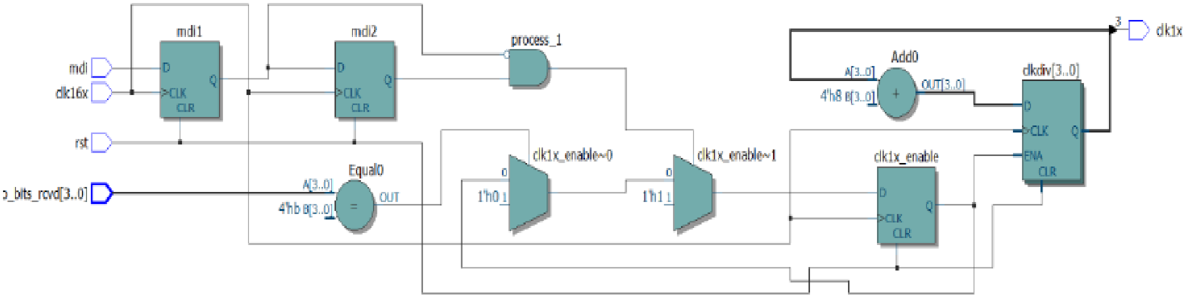


Figure 6: RTL level of reception clock management

clk1x	Reception clock
clk1x_enable	Activation of clk1x
mdi1, mdi2	Last state of mdi
no_bits_rcvd	Number of received bits
Clkdiv	Counter for frequency divider

Table 6: Definition of intermediate signals

Note: in your analyses, do not look at mdi1 and mdi2. This is just one way to implement the expression VHDL « wait ... until » which can't be synthetized on FPGA.

2) Process 4 and 5

The objective of these two processes is the management of the data received. Figure 7 shows the RTL level generated by these two processes.

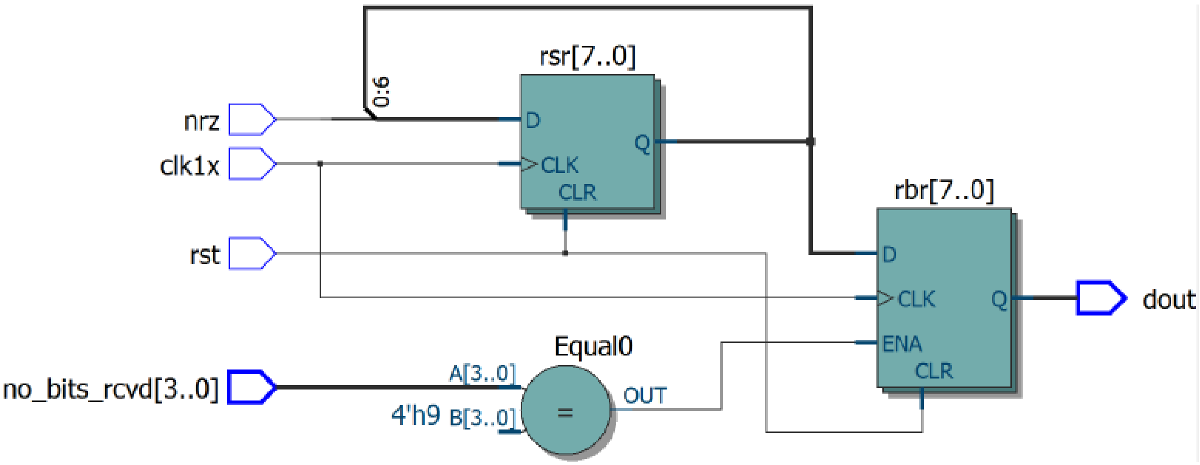


Figure 7: RTL of decoded data management

nrz	NRZ data after Manchester decoding
rsr	Reception shift register *mdi*
rbr	Reception buffer register

Table 7: Definition of intermediate signals

3) Process 6

The purpose of the last process is to calculate the number of bits received. The RTL level is similar to figure 4.

4) Manchester to NRZ decoding

The purpose of this part is to decode the data received, encoded in Manchester, into a single-pole NRZ signal. The figure 8 shows the corresponding RTL level.

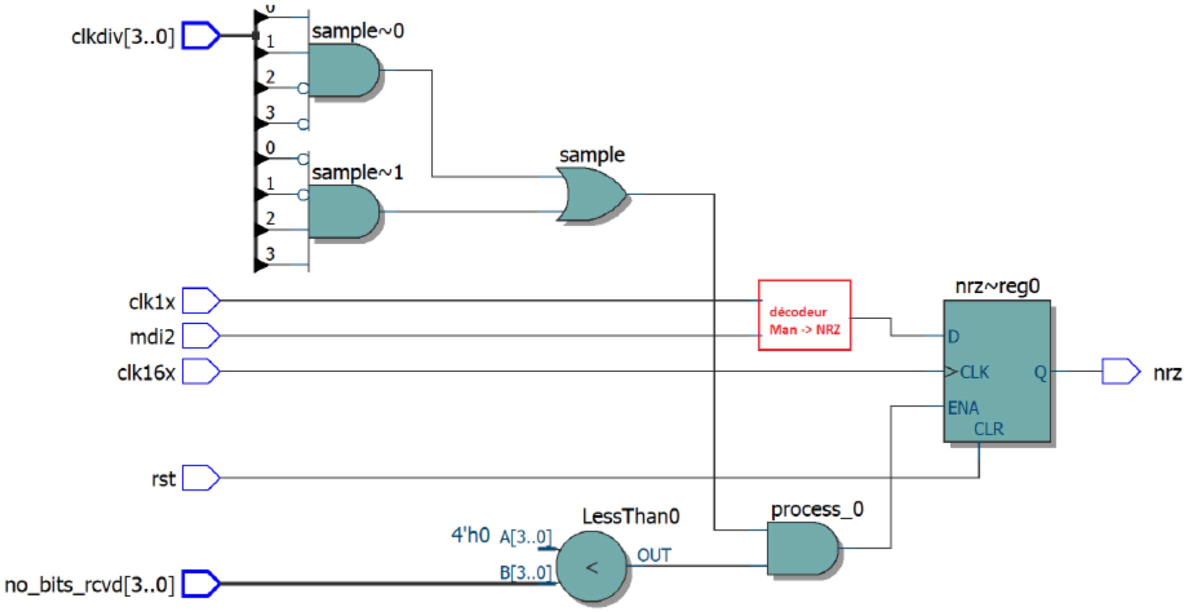


Figure 8: RTL of Manchester to NRZ decoder

5) Questions:

1- For figures 6, 7 and 8, write the algorithms to calculate the outputs as functions of the inputs.

2- To transmit a 2-bit frame, how many clk16x clock cycles are required? And for 8-bit frame?

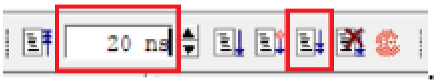
3- What is the purpose of the "sample" signal? (this signal is the key for Manchester decoding).

4- Propose a simple combinatorial expression to replace the red block in figure 8.

5- Complete the skeleton file md.vhd

6- Download the testbench md_tb.vhd file and put it in the same directory. This file contains stimuli to run a simulation.

7- Launch the ModelSim software, compile in the order md.vhd and md_tb.vhd. Set the simulation period to 20 ns and start a full simulation with the Run-all button.



8- Using the zoom tool, explore the different signals and comment on the results.