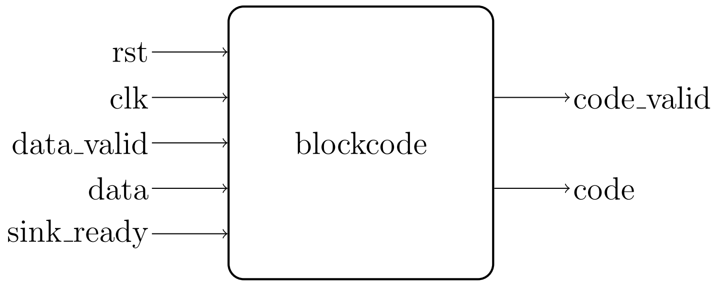
Blockcode

Your task is to program the behavior of an entity called "blockcode". This entity is declared in the attached file "blockcode.vhdl" and has the following ports:

- Input: *rst* with type std_logic; global synchronous reset
- Input: *clk* with type std_logic; global clock
- Input: *data_valid* with type std_logic
- Input: *data* with type std_logic_vector of length 5
- Input: *sink_ready* with type std_logic
- Output: *code_valid* with type std_logic
- Output: *code* with type std_logic_vector of length 8



Do not change the file "blockcode.vhdl".

This entity should work as an intermediate unit, that creates blockcode elements from given data elements and transmits them to the following data sink as seen in Figure 1. In the following sections the transmission and the encoding will be discussed in depth.

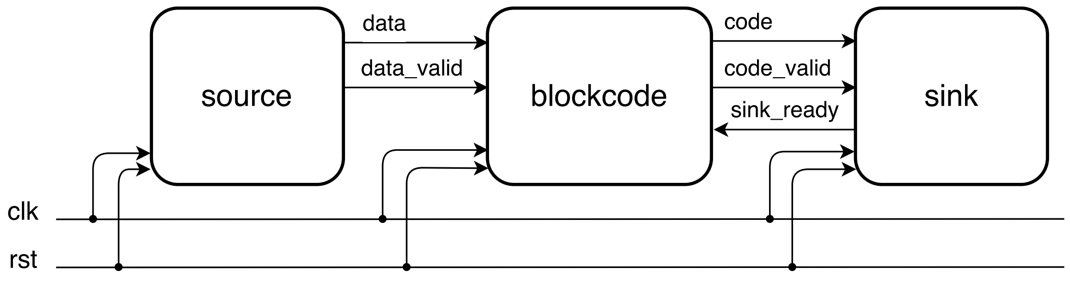


Figure 1: System view. The source creates data to be encoded by your entity, which sends the blockcode encoded data to the following data sink.

Transmission

Transmission data source → "blockcode" entity: This transmission is with the following ports:

- **data:** contains the data which is to be encoded as blockcode.
- **valid_data:** indicates if the data on the port *data* is valid in the clock cycle. The "blockcode" entity should only encode valid data.

Transmission "blockcode" entity → data sink: This is a connection with sink backpressure, meaning the sink can indicate if it is ready to receive data. It takes place with the following ports:

- **code:** contains the code which was generated by encoding given valid data on the port *data*.
- **code_valid:** indicates if the code at the port *code* is valid in the clock cycle.
- **sink_ready:** indicates if the sink is ready to receive code.

Backpressure: If the sink is not ready to receive it unsets the port *sink_ready* (*sink_ready* = 0). The "blockcode" entity sends the same code until the sink it is ready. The sink only captures code if both *code_valid* and *sink_ready* are set.

Buffering: As the sink can stall the transmission you will need a buffer in the "blockcode" entity (e.g. circular FIFO). Your buffer needs to be able to buffer a maximum of 4 elements for this task.

Synchronous outputs: All outputs of entities in the system are registered to the common clock *clk*, meaning that changes on these ports only happen at the rising edge of the common clock *clk*.

Encoding

The dataword has to be encoded as a $[8, 5]$ blockcode, meaning 5 data bits are padded with 3 parity bits:

$$d = (d_0, d_1, ..d_4) \longrightarrow c = (d_0, d_1, ..., d_4, p_0, ..., p_2) \tag{1}$$

The encoding is done with the help of a generator matrix G:

$$c = d \cdot G \tag{2}$$

Use the given generator matrix consisting of an identity matrix and the parity matrix (separated by a vertical line):

$$\left(\begin{array}{ccccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right)$$

Tip: As + in binary is the same as XOR, use XOR to calculate the parity.

Encoding example: (01111) $\longrightarrow$ (01111010)

System behavior

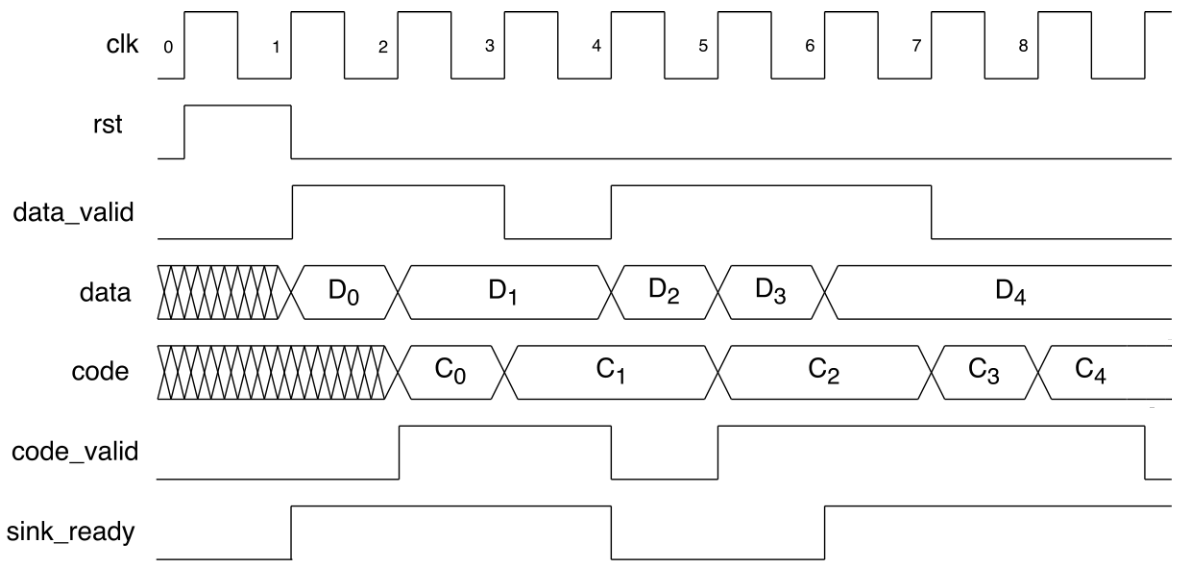


Figure 2: Example behavior of the whole system. The numbers denote the start of clock cycles.

Figure 2 illustrates the desired behavior of the entity "blockcode" by example for each clock cycle:

0. Global reset.
1. A valid data element D_0 is given. This is indicated by a *data_valid* being set.
2. A new valid data element D_1 is given. The encoded element C_0 is set to be transmitted. The sink is ready as seen by the set *sink_ready*. Therefore the sink captures the code element in this clock cycle.
3. No new valid data element is given, as seen by the unset *data_valid*. The encoded element C_1 is set to be transmitted and captured by the sink.
4. A new valid data element D_2 is given. As C_1 was already captured, the set code element is not valid, no transmission will occur. Therefore *code_valid* is unset.
5. A new valid data element D_3 is given. The encoded element C_2 is set to be transmitted and captured by the sink. The sink is not ready to capture as seen by the unset *sink_ready*.
6. A new valid data element D_4 is given. As C_2 was not captured in the last cycle it remains set. The sink is ready again and captures C_2.
7. No new valid data element is given. C_3 is set and captured.
8. No new valid data element is given. C_4 is set and captured.

The described behavior has to be programmed in the attached file “blockcode_beh.vhdl”.

To turn in your solution write an email to vhdl-dis+e384@tuwien.ac.at with Subject “Result Task 6” and attach your behavior file(s):

- “blockcode_beh.vhdl”

Good Luck and May the Force be with you.