# **LAB 1 - TASK 3**

In this task, you will learn how to check the correctness of received frame.

**INSTRUCTIONS**

The MATLAB code in the below window is similar to the code described in Task 1 where we simulated the performance of the slotted ALOHA protocol. The difference here is that we do not use the function **checkReceivedFrame**, but implement it in details. Also, instead of generating the frames on the fly, we load some pre-generated frames to the variable **"frame"** to simplfy simulation and checking the results of your code.

In this task, we detect errors in a given frame. Consider a small network where **n\_users**=4 users share the same channel to access a receiving node, which relays their datagrams the rest of the network. Once the frame has been received by the receiving node, it can check three things: the preamble, the user id, and the checksum. For this simulation, we set the preamble to be **[1 0 1 0]**. The user id of received frame should be a valid value, within 1 and **n\_users**=4 inclusive. To check the id we use the function **id = bin2num(word)** to convert a word of four binary numbers into a decimal number. Finally, we need to check the correctness of the message using the checksum.

To do that, we can divide the frame into four blocks of four bits each. Then, we can compute the checksum of these four blocks in the same way as described in the previous task (attention! Now we consider 4 words instead of 3). If the frame is correct the computed checksum should consist of four zeros. For example, if we receive the frame:

[ 1 0 1 0 ] [ 0 0 0 1 ] [ 1 0 1 0 ] [ 0 0 0 1]

We can divide it into the four words:

[ 1 0 1 0 ]

[ 0 0 0 1 ]

[ 1 0 1 0 ]

[ 0 0 0 1 ]

and the obtained the checksum is **[ 0 0 0 0 ]**.

In the initial MATLAB code, we load the variable **frames**, which contains the list of frames that we want to check. Then it loops over all the frames and your task is to validate the inspected frame (variable **frame**), by checking the preamble, the **id** and the checksum. Set the variable **cs\_ok** to 1 if the checksum is correct and to 0 otherwise. Do the same with the variables **pre\_ok**, **id\_ok** and **frame\_ok** by checking the preamble, the id and frame, respectively. Please, revise the code between the lines

% % % % Revise the following code % % % %

and

% % % % Do not change the code below % % % %

Do not change other parts of the code and do not use the function **checkReceivedFrame**. Note that we use "cs\_ok", "pre\_ok", "id\_ok" and "frame\_ok" to indicate whether the checksum, the preamble, the user id and the whole frame is valid. Do not rename them.

**Output**

Correct preamble: 53/800 Correct id: 39/800 Correct checksum: 800/800 Correct frames: 0/800