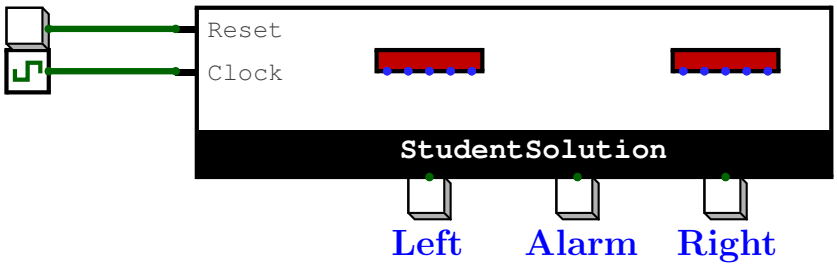
1 Introduction

In this practical evaluation we are going to design a new direction light system for a car (see below).



This system has a button connected to the `Reset` input that represents the Power On Reset that is activated when the car is started. Unless otherwise stated this `Reset` input is a *synchronous reset*. Furthermore, there are three buttons:

- `Left`: This button is activated when the car driver wants to go left, only activating the left direction lights.
- `Right`: This button is activated when the car driver wants to go right, only activating the right direction lights.
- `Alarm`: This button is activated in an alarm case and hence both the direction lights are activated.

Finally each of the direction lights consists of 5 lamps that are connected MSB-left....LSB-right.
For the buttons you can assume that `Left` and `Right` can never be activated together due to their mechanical construction. Finally the `Alarm` button has precedence over the other two.

2 General restrictions

The system has to be realized in the template that you can download from digsys.epfl.ch (please use a VPN-connection). You have to use Logisim-evolution V3.4.1 or V3.4.2. Your solution has to be submitted to digsys.epfl.ch and you have a maximum of 5 tries. Note that you can submit to digsys.epfl.ch as often as you want (the system allows it), but only the submissions 2...6 will be taken into account, and from these 5 submissions the last one submitted will be graded. Furthermore: Only the solutions submitted to digsys.epfl.ch will count for grading, there is no other means to submit your solution.

For your solution you are allowed to use following components (unless otherwise noted in the exercise):

Memory	
D-FlipFlop	Register

Wiring					
Splitter	Pin	Probe	Tunnel	Clock	Constant

Gates						
Not	And	Or	Nand	Nor	Xor	Xnor

3 The design

We will build-up the complete system step by step by using an incremental modular based design methodology.

Exercise 1:

We will start with designing one direction light. The direction light circuit must be implemented trough a *Medvedev* Finite State Machine (FSM). This circuit will be designed in the sheet `E1Solution` in the logisim template. The direction light circuit has a synchronous input `Activate` that is at least 1 clock cycle high. When this input is activated the light will start it's *sequence*. The *sequence* the lamp will make can be seen on the video that you can download from digsys.epfl.ch. This *sequence* will always continue to it's end even if the input `Activate` is deactivated before the sequence ended. Furthermore the sequence will start over when the input `Activate` is still active when the sequence ended.

Once we have the above circuit, we can build our first system that implements a *simple* direction system.

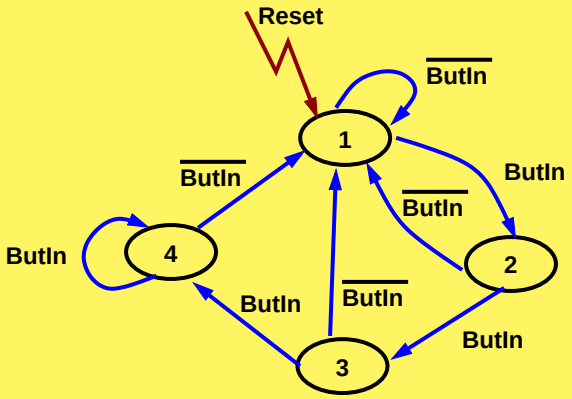
Exercise 2:

Build in the sheet `StudentSolution` your first system that meets the requirements of the introduction, using your solution of exercise 1.

Now that we have our first system, we are going to add some extra *features* to it. We want to detect that a button is at least activated for 3 clock cycles. For this purpose we are going to design an FSM.

Exercise 3:

For the press detection we are going to implement an FSM with following state-diagram:



For coding the states we use a one-hot encoding where state `1` is encoded with 0001_b. The output `State` shows the current state the FSM is in. The output `ButOut` is only active when the current state is `4`. Build this FSM in the sheet `E3Solution`.

Now that we have this button detector, we are going to use it.

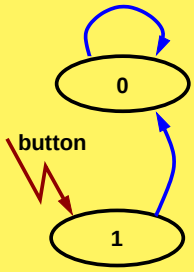
Exercise 4:

Extend the sheet `StudentSolution` with the above button detector for the buttons `Left` and `Right`.

Now that we are able to detect long button activation, we also want to detect short button activation.

Exercise 5:

To detect a short button activation we start with building a *Medvedev* FSM with following state-diagram:



The output of this *Medvedev* FSM is connected to the LSB of a 4-bit shift register. The shift register is asynchronously reset to 0.
To have the correct detection the output `short` will only be activated in case the `button` is not activated. The output `short` will be activated in case the MSB of the shift register is 0_b and the (MSB-1) of the shift register is 1_b. The output `state` represent the 4-bits of the shift-register.
Create the circuit for this system in sheet `E5Solution`.

We now have two circuits that can detect a long press and a short press, which are mutual exclusive. We are now going to finalize our system.

Exercise 6:

To finalize our system we are going to implement following functionality:

1. Alarm behavior: The alarm behavior will overrule all other behaviors and will let both direction indicators perform their sequence.

2. Long press behavior: As long as we have a long-press on the button `left` or `right` the corresponding direction indicator perform their sequence.

3. Short press behavior: If a short press is detected on the button `left` or `right` the corresponding direction indicator will perform exactly 4 sequences.

Extend the sheet `StudentSolution` with the above functionality.