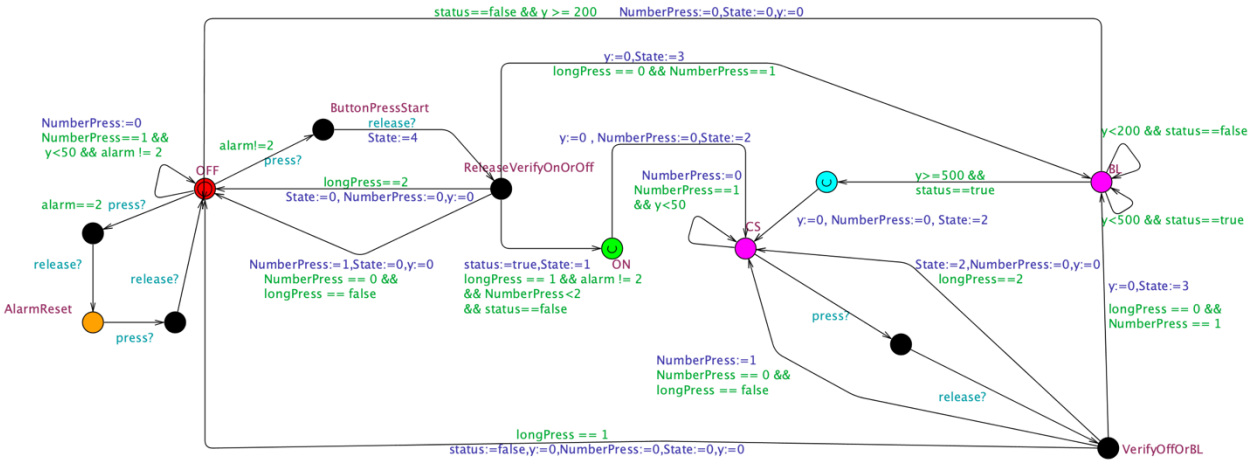
UPPAAL by Ali Dhanani 000470296

The main idea using UPPAAL was to develop the button configuration of the bike. Here we take 100 ticks to represent 1 Sec. The main idea of the button configuration is analyzing the operations such as ON, OFF, Showing of Battery Percentage and showing Speed. Other than these configurations we also must think about the alarm meter that checks if the bike is in a good state.

The Idea is as following:

- Bike Turn ON if it was Turned OFF, by pressing the button continuously for 1 sec
- Bike Turn OFF if it was Turned ON, by pressing the button continuously for 1 sec
- If bike is at State OFF and the user presses button for less, then 1 sec then show the battery light
- If the alarm level is 2 and the bike is at OFF state, then it cannot turn ON unless reset
- Bike can only be turned on If the alarm level is 0 or 1
- Once Bike is ON, it shows the current Speed of the bike
- If at current speed and presses the button for less than 1 sec, then it shows the battery level
- It doesn't stay at the battery level state for more than 5 sec
- If bike is at state OFF and alarm level is 2, then alarm level can be reset by pressing the button 2 time continuously within 15 sec

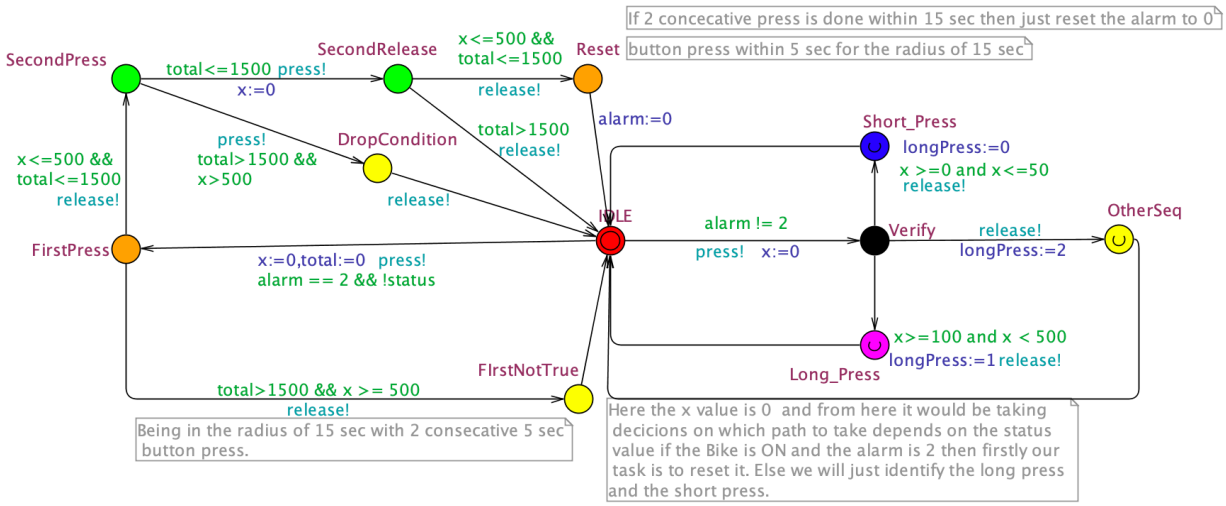
Bike Controller:



This is the Bike Model, the initial state is OFF, you can see that there are 2 presses, this part relates to the user model. The first press in the push button and the other is related to the release of the button. In the user model, we will see in detail why we need 2 presses. Also, the 2 presses will identify the time takes for the push and release. The state "ReleaseVerifyOnOrOff" analysis the condition of the button to be pressed for over 1 sec or less. If the button is pressed over 1 sec and less then 5 sec it will switch the bike on else, it will show the battery level (BL) in the condition where the button will be tapped 2 times for ½ sec. A bike will show the BL just for 5 sec in the condition where the bike is ON after that it will return to the CS state. In the condition where bike is OFF the BL will be shown for just 2 sec after which it will be returned to the OFF state. For switching the bike on, it is also important that the alarm level is not 2. If the alarm level is 2 then it is important to first reset the bike to 0 or 1. Once the bike is in the ON state it will directly take us to the current speed state CS, from the CS state it can either be switched off or we can see the Battery level BL for less than 5 sec. After CS state, you can see that we have added

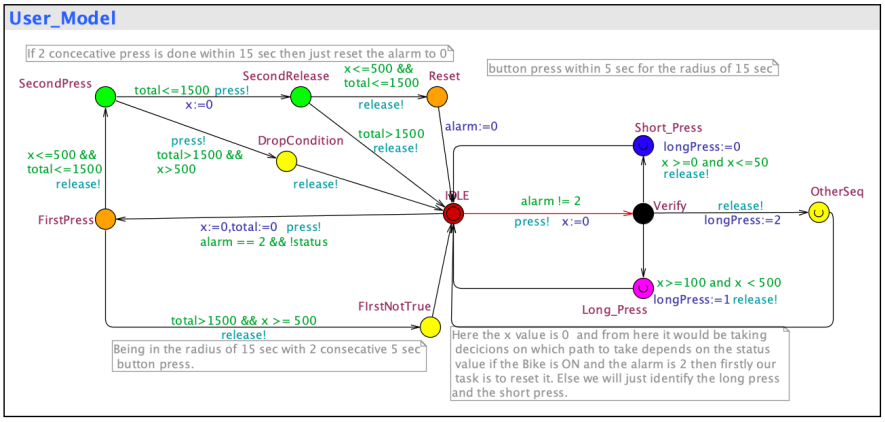
again 2 presses. One for the push and the other for the release. If from the CS the button was pressed for less than 1 sec then it will display the battery level in the condition where we have 2 taps of ½ sec within 1 sec, and in the case where the push and release took over 1 sec, it will switch off the bike. In order to differentiate between the whither the bike is in the ON state or OFF state while being in the BL state, I have added the Boolean variable of state, which is initially false, once it visits the ON state, the value changes to true. This is how it will identify while being in the BL state. In the condition where the press is more than ½ sec and less then 1 sec, we just return back to the previous state as we don't have perform any operation as such. Furthermore, our model doesn't work if the alarm level is 2 so we first start with reset the alarm level in order to do any operation in the bike.

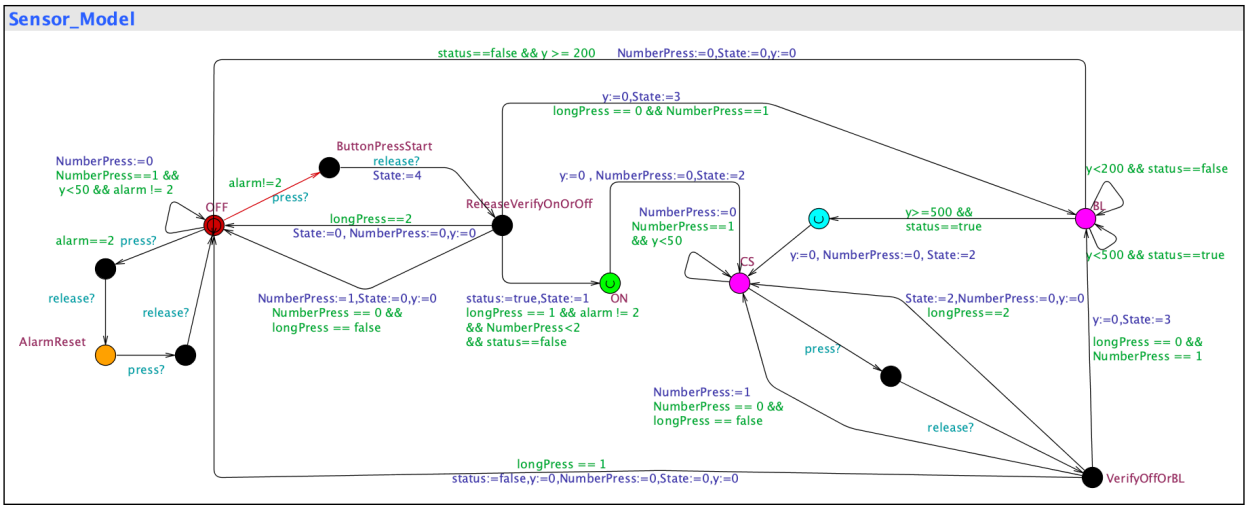
User Model:



This is the User Model template that keeps track of the user's behavior on the button press duration. Here initially the user is in the IDLE state which means the user hasn't pressed any button yet. Firstly, I will be explaining in the condition where alarm is not 2. x is the clock variable that keep track of the press, hold and release of the button. Initially the clock x is 0, while being in the state Verify where it will move to the other state depending on the button release. The button if released within ½ sec, it will move to the state Short_Press else if it will move to Long_Press if it is more than 1 sec, also in the case where the it is more than ½ sec and less than 1 sec it moves to other seq meaning moving back to the previous state in the controller . We have a Boolean variable longPress which is initially false and will only be true of the button was released after 1 sec. Now coming to the condition where alarm is 2. The idea is that it can be reset if the button is pressed 2 times consecutively within 15 sec. If clock goes above 15 sec then we will move back to the idle state and start all over again.

Simulation Design:





This is the simulation view of UPPAAL for the bike and the user template. We can see that the 2 templates relate to each other on the bases of the press operation, model state and the longPress value.

Verification:

Here we have written some LTL specification of the bike that we want to check on UPPAAL. As we cannot write nested property, so I have created the User model and Bike Model separately to have a link between the 2 on certain properties to be satisfied.

Firstly, it is to check, from all the states there is eventually a way to turn the bike ON when there is a long press

```
A<> Sensor_Model.ON imply User_Model.Long_Press
```

We are checking, from the reachable states, there is a way to go back to the ON state of the Bike by E<> Sensor_Model.ON

```
User_Model.Short_Press --> not Sensor_Model.ON
```

We are here identifying whether there exists a deadlock in the model

```
A[] not deadlock
```

Here is a check to see if it remains in the BL state button for less than 5 sec.

To Check for all the possible state there is a way to turn the bike OFF

If the bike has the alarm level 2 and it is at state OFF, it cannot be turned ON

Here we are using the idea of the it does not move to the ON state in the condition where have a short press or any press less then 1 sec

Reference:

https://www.it.uu.se/research/group/darts/uppaal/small_tutorial.pdf