# Proposed answers to the exercise on Lustre

In some old-fashionned mechanical mice, the deplacement of the mouse (in each direction) was measured using the following device: A wheel, coloured as shown by Fig. 1, can turn in each direction. Two fix sensors are placed according to a radius of the wheel, as shown on Fig 2, and detect the colour in front of them, transmitting them as Boolean flows — say is and os, for "inside sensor" and "outside sensor"—, where "true" and "false" stand for "white" and "black", respectively. In the position of Fig. 2, we have thus is = true and os=false.

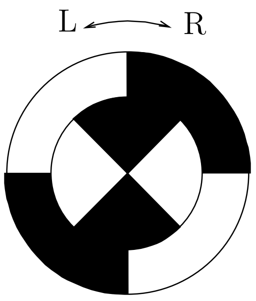


Figure 1: The wheel

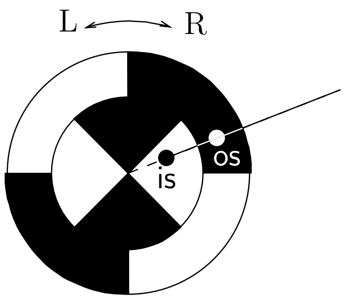


Figure 2: The sensors

**a)** The sampling period of the sensors is assumed to be short enough so that any colour change is perceived. As a consequence, the values of the Boolean variables is and os cannot both change at the same time. Considering is and os as Boolean inputs to a Lustre program, express this assumption by an assertion in Lustre.

Since a change can be detected only after the initial instant, the assumption has effect only after the initial instant. So, we write:

```
assert true ->(is = pre(is)) or (os = pre(os));
```

**b)** Under the preceding assumption, the sequences of values is and os allow to determine the rotation moves of the wheel: one can observe either a left move, or a right move, or nothing. The following figure depicts a possible scenario; complete the rows of corresponding observations.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | time |
|---|---|---|---|---|---|---|---|---|---|---|
| is | T | T | F | F | T | T | T | F | F | |
| os | F | T | T | T | T | T | F | F | T | |
| left | F | T | T | F | F | F | F | F | F | |
| right | F | F | F | F | T | F | T | T | T | |

(notice that both outputs are false when no move is detected)

**c)** Write a Lustre node Position, taking as inputs the Boolean variables is and os, and returning a relative integer x counting the moves of the wheel, i.e., starting from 0 and incremented (resp., decremented) whenever there is a right (resp., left) move of the wheel (one may use the node Edge seen in the course).

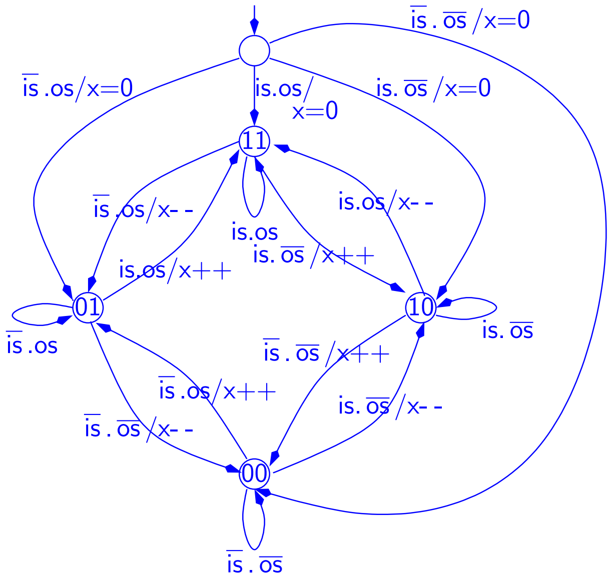
Remember that Edge(c) = false ->c and not pre(c) is true when there is a rising edge of c, so Edge(not c) is true when there is a falling edge of c. According to the 4 possible types of edges, the direction of the move is given by the value of the unchanging sensor. A possible solution is the following:

```
node Position (is, os : bool) returns (x : int);
var left, right: bool;
let
    assert true ->(is = pre(is)) or (os = pre(os));
    x = 0 ->    if right then pre(x) + 1
                else if left then pre(x) - 1
                else pre(x);
    (left, right) = (false,false) ->
            if Edge(is) then (not os, os)
            else if Edge(not is) then (os, not os)
            else if Edge(os) then (is, not is)
            else if Edge(not os) then (not is, is)
            else (false,false);
tel
```

**d)** Draw the control automaton corresponding to the node Position.

Let's label the non initial states by the previous values of is and os. For instance, in state 01, we know that pre(is)=false ans pre(os)=true, and it is the target of all transitions triggered by inputs $\overline{\text{is}}$.os.

From the initial state, where x is set to 0, the next state is chosen according to the current values of inputs. Now, in each state, we know the previous values of inputs, and only one of them can change, according to the assertion.



$\overline{\text{is}}.\overline{\text{os}}$/x=0
$\overline{\text{is}}$.os/x=0
is.os/ x=0
is.$\overline{\text{os}}$/x=0
11
$\overline{\text{is}}$.os/x- -
is.os/x- -
is.os
is.$\overline{\text{os}}$/x++
is.os/x++
01
10
$\overline{\text{is}}$.os
is.$\overline{\text{os}}$
$\overline{\text{is}}.\overline{\text{os}}$/x++
$\overline{\text{is}}$.os/x++
is.$\overline{\text{os}}$/x- -
$\overline{\text{is}}.\overline{\text{os}}$/x- -
00
$\overline{\text{is}}.\overline{\text{os}}$