CLEARSY
Safety Solutions Designer

AIX
LYON
PARIS
STRASBOURG

WWW.CLEARSY.COM

Hackathon
JUL2025

Airlock Access Control

Thierry Lecomte
R&D Director

THIERRY.LECOMTE@CLEARSY.COM

PART I

One Solution

Your turn

- *Develop a B model of validation function [8 pts]*
  - *Transformation for even indexes 0, 2, 4, … 14*

```
idx_odd <: 0..15 &
idx_even <: 0..15 &
idx_odd /\ idx_even = {} &
idx_odd \/ idx_even = 0..15 &

                                        SIGMA(xx).(xx: idx_even | (tab;map)(xx))

map: 0..9 --> 0..9 &
map = {
    0 |-> 0, 1 |-> 2, 2 |-> 4, 3 |-> 6, 4 |-> 8,
    5 |-> 1, 6 |-> 3, 7 |-> 5, 8 |-> 7, 9 |-> 9
} &
```

Your turn

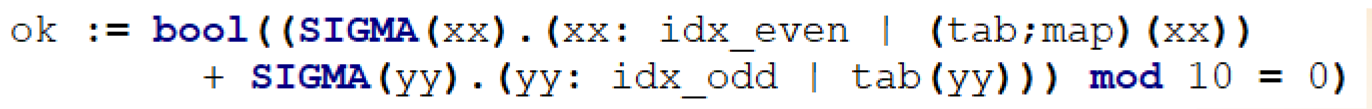
- *Develop a B model of validation function [8 pts]*
  - *Transformation for odd indexes 1, 3, 5 … 15*

```
idx_odd <: 0..15 &
idx_even <: 0..15 &
idx_odd /\ idx_even = {} &
idx_odd \/ idx_even = 0..15 &
```

```
SIGMA(yy).(yy: idx_odd | tab(yy))
```

Your turn

- *Develop a B model of validation function [8 pts]*
  - *The addition of the two sums modulo 10 should return 0 to validate*



```
ok := bool((SIGMA(xx).(xx: idx_even | (tab;map)(xx))
        + SIGMA(yy).(yy: idx_odd | tab(yy))) mod 10 = 0)
```

```
MACHINE
    ACCESS_CARD
ABSTRACT_CONSTANTS
    map,
    idx_odd,
    idx_even
PROPERTIES
    map: 0..9 --> 0..9 &
    map = {
        0 |-> 0, 1 |-> 2, 2 |-> 4, 3 |-> 6, 4 |-> 8,
        5 |-> 1, 6 |-> 3, 7 |-> 5, 8 |-> 7, 9 |-> 9
    } &
    idx_odd <: 0..15 &
    idx_even <: 0..15 &
    idx_odd /\ idx_even = {} &
    idx_odd \/ idx_even = 0..15 &
    idx_odd = {1, 3, 5, 7, 9, 11, 13, 15} & // Required for ProB animation
    idx_even = {0, 2, 4, 6, 8, 10, 12, 14} // Required for ProB animation
OPERATIONS
    ok <-- is_valid(tab) =
    PRE
        tab: 0..15 --> 0..9
    THEN
        ok := bool((SIGMA(xx).(xx: idx_even | (tab;map)(xx))
                + SIGMA(yy).(yy: idx_odd | tab(yy))) mod 10 = 0)
    END
END
```

Your turn

- *Check the numbers with ProB [2 pts]*

```
ok <-- is_valid({ /* 4137 8947 1175 5904 */
          0 |-> 4, 1 |-> 1, 2 |-> 3, 3 |-> 7,
          4 |-> 8, 5 |-> 9, 6 |-> 4, 7 |-> 7,
          8 |-> 1, 9 |-> 1, 10 |-> 7, 11 |-> 5,
          12 |-> 5, 13 |-> 9, 14 |-> 0, 15 |-> 4
     });
```

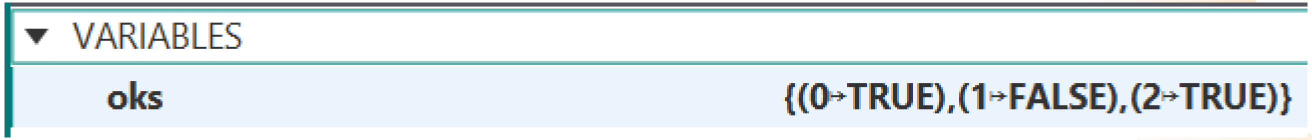
```
test_is_valid =
VAR ok IN
    ok <-- is_valid({ /* 4137 8947 1175 5904 */
            0 |-> 4, 1 |-> 1, 2 |-> 3, 3 |-> 7,
            4 |-> 8, 5 |-> 9, 6 |-> 4, 7 |-> 7,
            8 |-> 1, 9 |-> 1, 10 |-> 7, 11 |-> 5,
            12 |-> 5, 13 |-> 9, 14 |-> 0, 15 |-> 4
        });
    oks(0) := ok;

    ok <-- is_valid({ /* 1234 5678 9012 3456 */
            0 |-> 1, 1 |-> 2, 2 |-> 3, 3 |-> 4,
            4 |-> 5, 5 |-> 6, 6 |-> 7, 7 |-> 8,
            8 |-> 9, 9 |-> 0, 10 |-> 1, 11 |-> 2,
            12 |-> 3, 13 |-> 4, 14 |-> 5, 15 |-> 6
        });
    oks(1) := ok;

    ok <-- is_valid({ /* 0018 2634 4259 6775  */
            0 |-> 0, 1 |-> 0, 2 |-> 1, 3 |-> 8,
            4 |-> 2, 5 |-> 6, 6 |-> 3, 7 |-> 4,
            8 |-> 4, 9 |-> 2, 10 |-> 5, 11 |-> 9,
            12 |-> 6, 13 |-> 7, 14 |-> 7, 15 |-> 5
        });
    oks(2) := ok
END
```

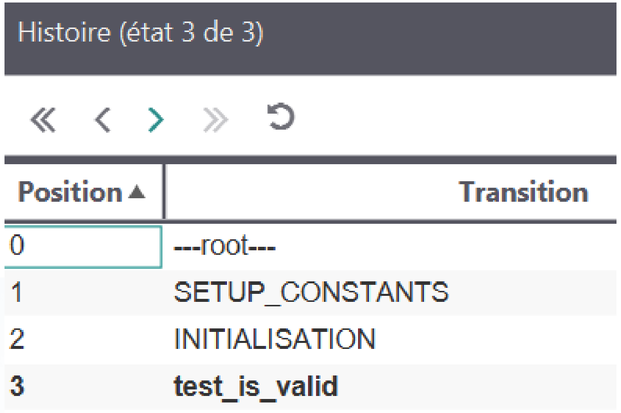
Your turn

- *Check the numbers with ProB [2 pts]*



VARIABLES

oks	**{(0↦TRUE),(1↦FALSE),(2↦TRUE)}**



Histoire (état 3 de 3)

Position ▲	Transition
0	---root---
1	SETUP_CONSTANTS
2	INITIALISATION
3	**test_is_valid**

Your turn

- ***Optional**: Is there any trivial (but suspect) number validated with the algorithm? [1 pt]*
  - *0000 0000 0000 0000*
- ***Optional**: Can you design a simple method to quickly generate some valid numbers without paper and computer? [2 pt]*
  - *For each two successive digit, double the first and add the second to obtain 10, repeat*
  - *Ex: 00 18 26 34 42 59 67 75 83 91*