Working Session 2
Programming the CLEARSY Safety Platform

Paulo Bezerra , Thierry Lecomte

ROBOSTAR
robostar.cs.york.ac.uk

11th March 2025

Royal Academy of Engineering

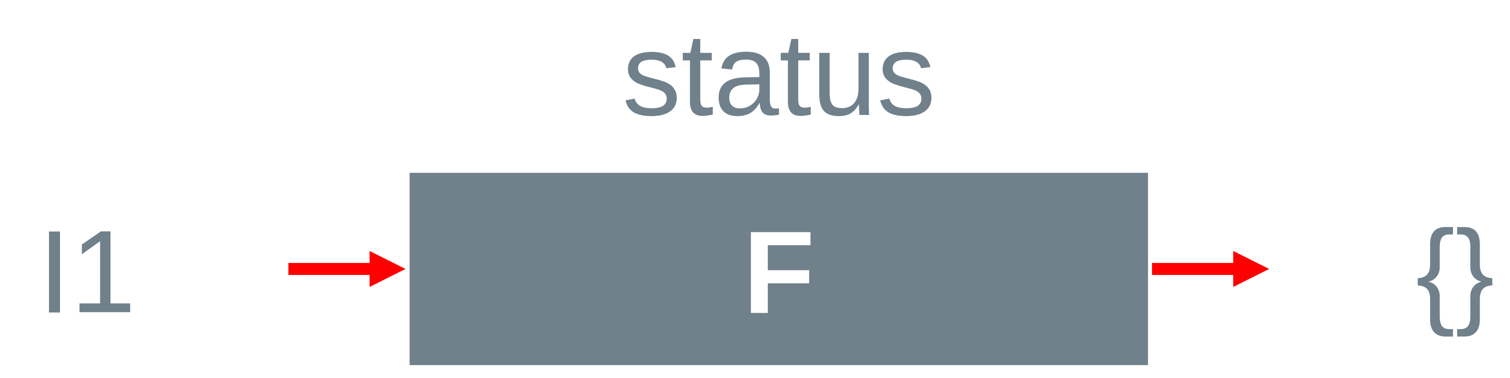
Engineering and Physical Sciences Research Council

Agenda

- Developing a combinatorial function
- Developing a synchronous / timed function

Developping a Combinatorial Function

Combinatorial Function



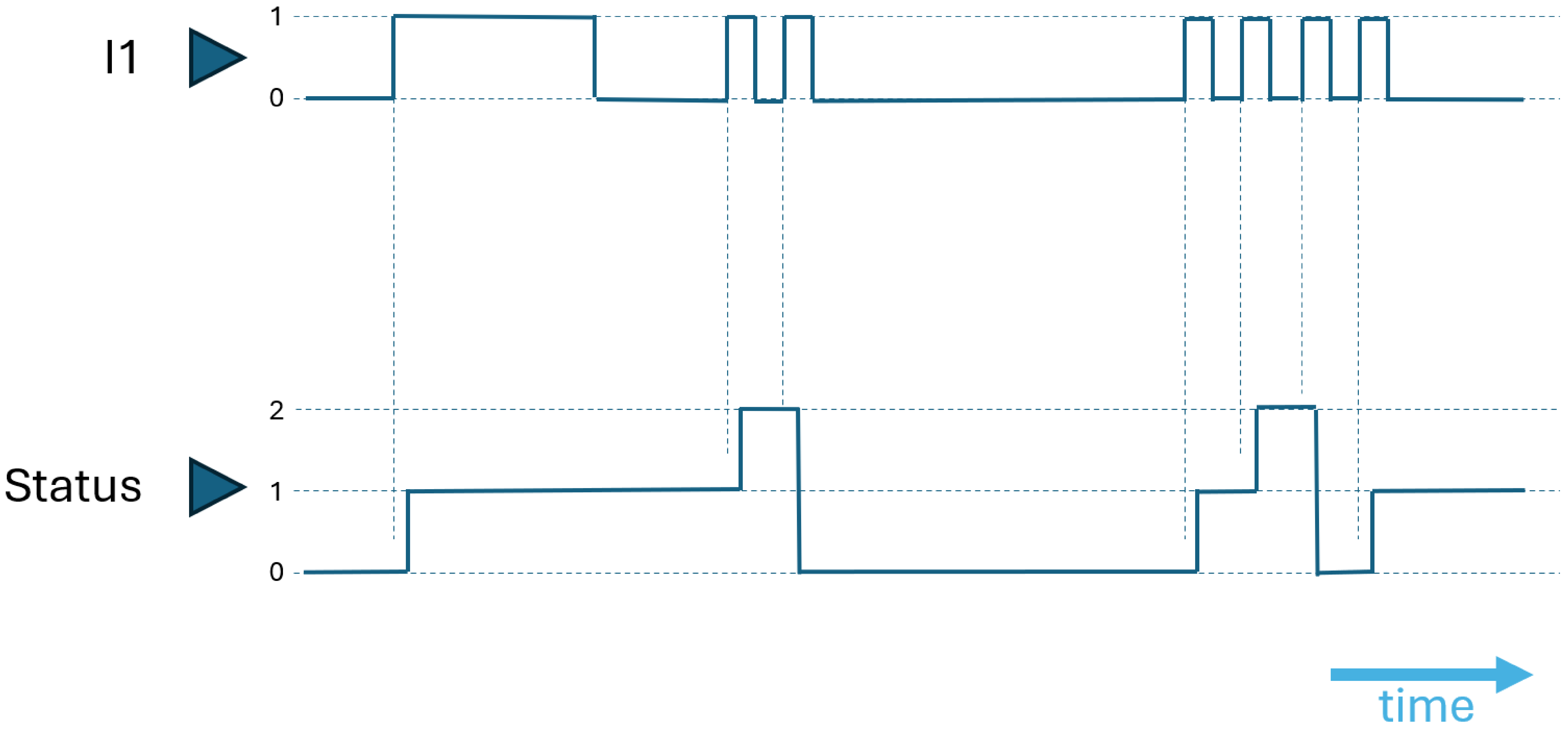
- *I1* belongs to {IO_OFF, IO_ON}
- *status* belongs to 0..2
- Every time *I1* changes from IO_OFF to IO_ON, *status* changes such as 0 → 1 → 2

We need to read input I1

We need to program a behaviour

ROBOSTAR

Combinatorial Function - scenario

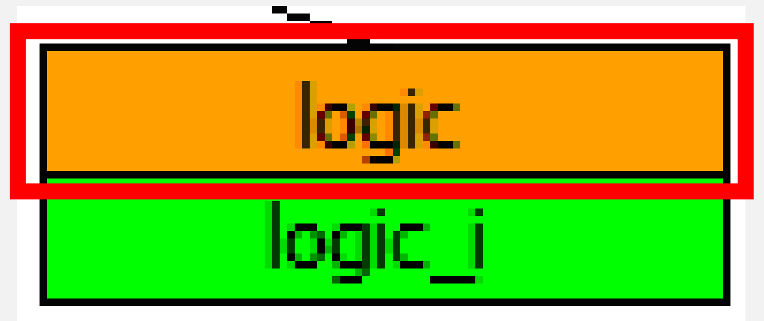


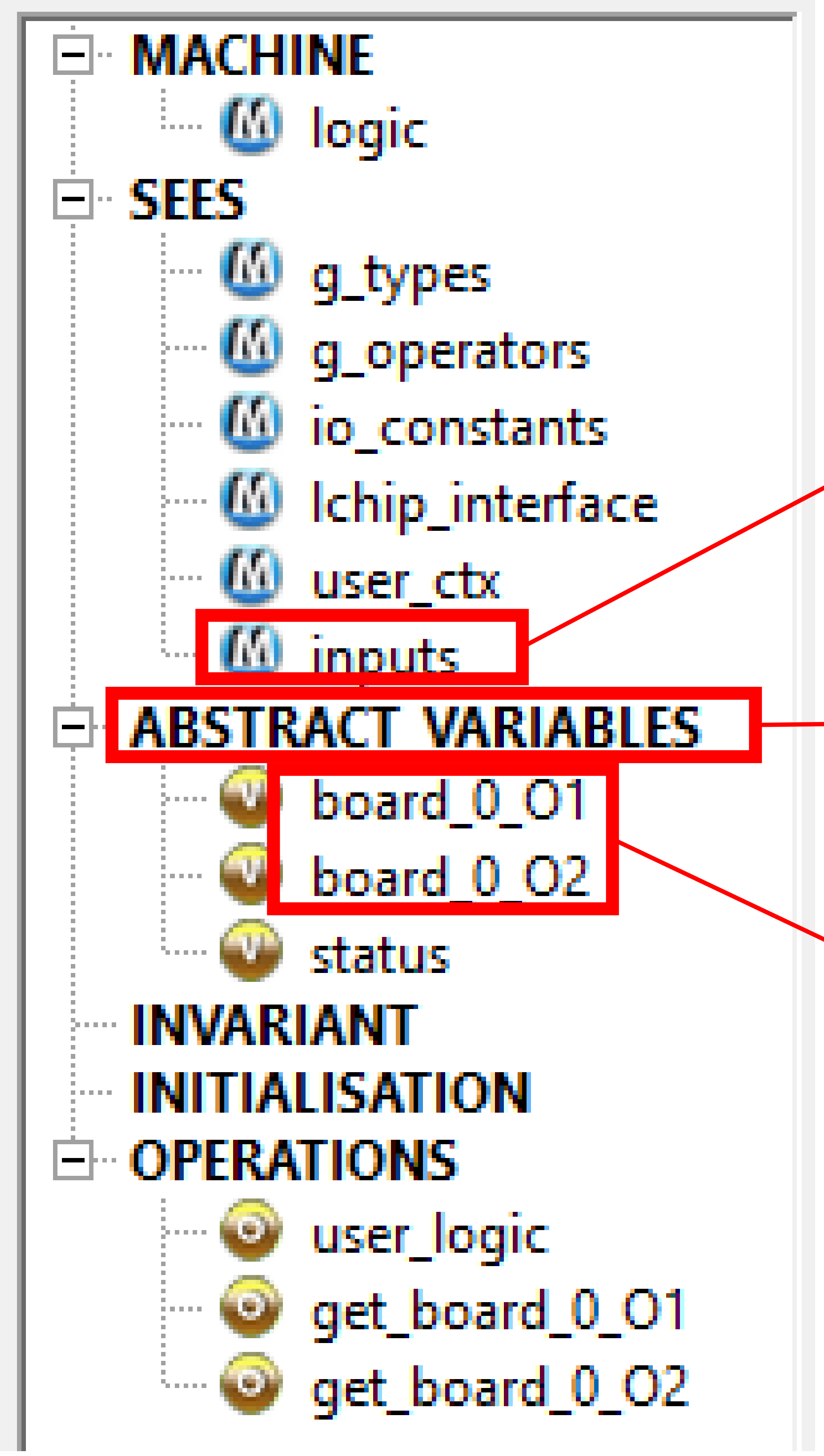
ROBOSTAR

Combinatorial Function

Specification component

double-click to open





Inputs are not variables of this model.
They are read and accessed from the *inputs* component

Abstract variables are not necessarily implemented.
They could appear only in specification for « reasoning »

Outputs are variables of this component.
They are modified in this component.
In B, variables are modified in a single component

Combinatorial Function

Specification component

double-click to open

logic

logic_i

- MACHINE
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- ABSTRACT_VARIABLES
  - board_0_O1
  - board_0_O2
  - status
- INVARIANT
- INITIALISATION
- OPERATIONS
  - user_logic
  - get_board_0_O1
  - get_board_0_O2

addition

modification

modification

```
ABSTRACT_VARIABLES
    board_0_O1,
    board_0_O2,
    status
INVARIANT
    board_0_O1 : uint8_t &
    board_0_O2 : uint8_t &
    status : uint8_t
INITIALISATION
    board_0_O1 :: uint8_t ||
    board_0_O2 :: uint8_t ||
    status := 0
OPERATIONS
    user_logic = BEGIN
        board_0_O1,
        board_0_O2,
        status: (
            board_0_O1 : uint8_t &
            board_0_O2 : uint8_t &
            status : uint8_t
        )
    END;
```

list

conjunct

Parallel initialisation
No order

Non deterministic substitution

Combinatorial Function

Specification component

double-click to open

logic

logic_i

```
ABSTRACT_VARIABLES
    board_0_O1,
    board_0_O2,
    status
INVARIANT
    board_0_O1 : uint8_t &
    board_0_O2 : uint8_t &
    status : uint8_t
INITIALISATION
    board_0_O1 :: uint8_t ||
    board_0_O2 :: uint8_t ||
    status := 0
OPERATIONS
    user_logic = BEGIN
        board_0_O1,
        board_0_O2,
        status: (
            board_0_O1 : uint8_t &
            board_0_O2 : uint8_t &
            status : uint8_t
        )
    END;
```

The 3 variables are going to evolve according to their type and could keep the same value

xx: (xx : T(xx))

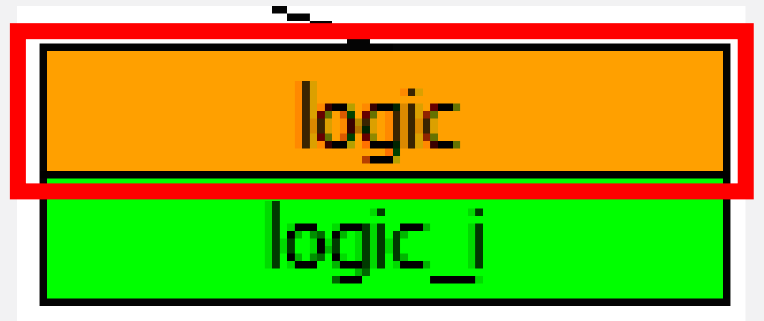
means

« xx becomes such that xx belongs to its type »

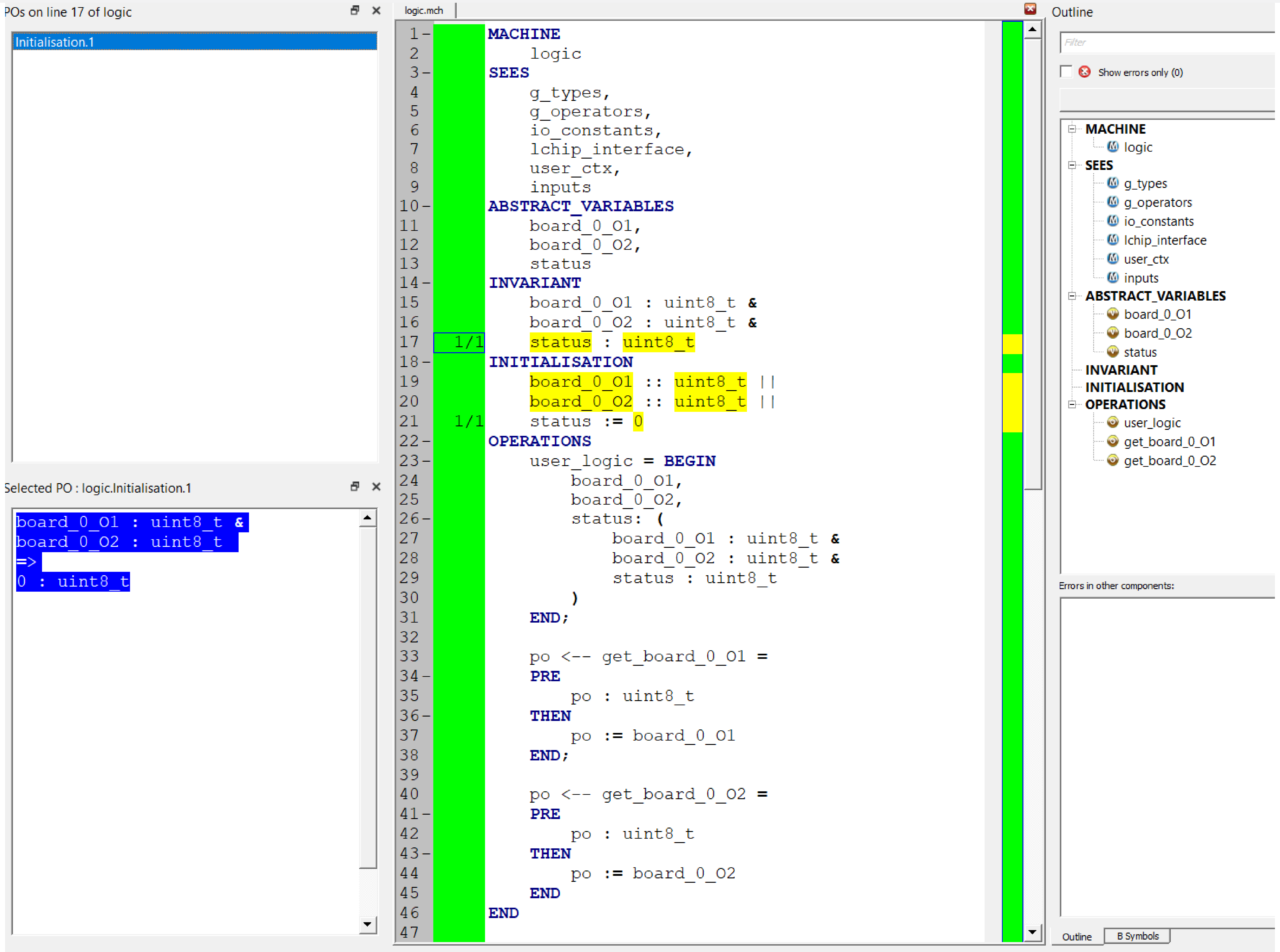
Combinatorial Function

Specification component

double-click to open



- Once the editing is completed
- Type Ctrl-S to save
- Proof is automatically initiated
- You should get a fully green (proved) component



Combinatorial Function

Implementation component

double-click to open

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

addition

modification

```
CONCRETE_VARIABLES
    board_0_O1,
    board_0_O2,
    last_i1,
    status
INVARIANT
    board_0_O1 : uint8_t &
    board_0_O2 : uint8_t &
    last_i1 : uint8_t &
    status : uint8_t
INITIALISATION
    board_0_O1 := IO_OFF;
    board_0_O2 := IO_OFF;
    last_i1 := IO_OFF;
    status := 0
```

means « to appear in code »

We need to store the « previous value of *I1* »

Sequential initialisation
Is separator, not end-of-line

The previous value of *I1* is considered not pressed (or down) at initialisation

ROBOSTAR

Combinatorial Function

Implementation component

double-click to open

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_01
  - board_0_02
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_01
  - get_board_0_02

addition

```
LOCAL_OPERATIONS
    n_ <-- next(state) =
    PRE state: uint8_t & n_ : uint8_t THEN
        n_ :: uint8_t - {state}
    END
OPERATIONS
    user_logic =
    BEGIN
    END;

    n_ <-- next(state) =
    BEGIN
        IF state = 0 THEN n_ := 1
        ELSIF state = 1 THEN n_ := 2
        ELSE n_ := 0
        END
    END;
```

OPERATIONS declared locally

user_logic can call *next*

Operation *next* with one parameter *state* returning *n_*

Precondition used to indicate types for parameter and return value

Non-deterministic substitution: *n_* becomes equal to any uint8_t except the value of *state*

Combinatorial Function

Implementation component

double-click to open

logic

logic_i

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

addition

```
LOCAL_OPERATIONS
    n_ <-- next(state) =
    PRE state: uint8_t & n_ : uint8_t THEN
        n_ :: uint8_t - {state}
    END
OPERATIONS
    user_logic =
    BEGIN
    END;

    n_ <-- next(state) =
    BEGIN
        IF state = 0 THEN n_ := 1
        ELSIF state = 1 THEN n_ := 2
        ELSE n_ := 0
        END
    END;
```

Same signature
Types are kept from
LOCAL_OPERATIONS precondition

Algorithm implemented with
IF THEN ELSE

Assignment

ROBOSTAR

Combinatorial Function

Implementation component

double-click to open

logic

logic_i

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

modification

1/1

```
user_logic =
BEGIN
    VAR i1_ IN
        i1_ : (i1_ : uint8_t);

            TO COMPLETE

    END
END;
```

*i1_* local variable used in this block

Variables list

Typing predicates used by code generator. No impact on behaviour

ROBOSTAR

Combinatorial Function

Implementation component

double-click to open

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

```
user_logic =
BEGIN
    VAR i1_ IN
        i1_ : (i1_ : uint8_t);

    END
END;
```

1/1

If I1 has a raising edge then change status to its next value.

Update the last value of I1

Combinatorial Function

Implementation component

double-click to open

logic

logic_i

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

```
user_logic =
BEGIN
    VAR i1_ IN
        i1_ : (i1_ : uint8_t);



    END
END;
```

1/1

If I1 has a raising edge then change status to its next value.

Update the last value of I1

HINT1
Operation call syntax:
vv <-- ff(pp)

HINT2
The operation to read *I1* is in the *inputs* component

HINT3
To assign a variable *vv*:
vv := value

CONSTRAINT
Only one condition per IF

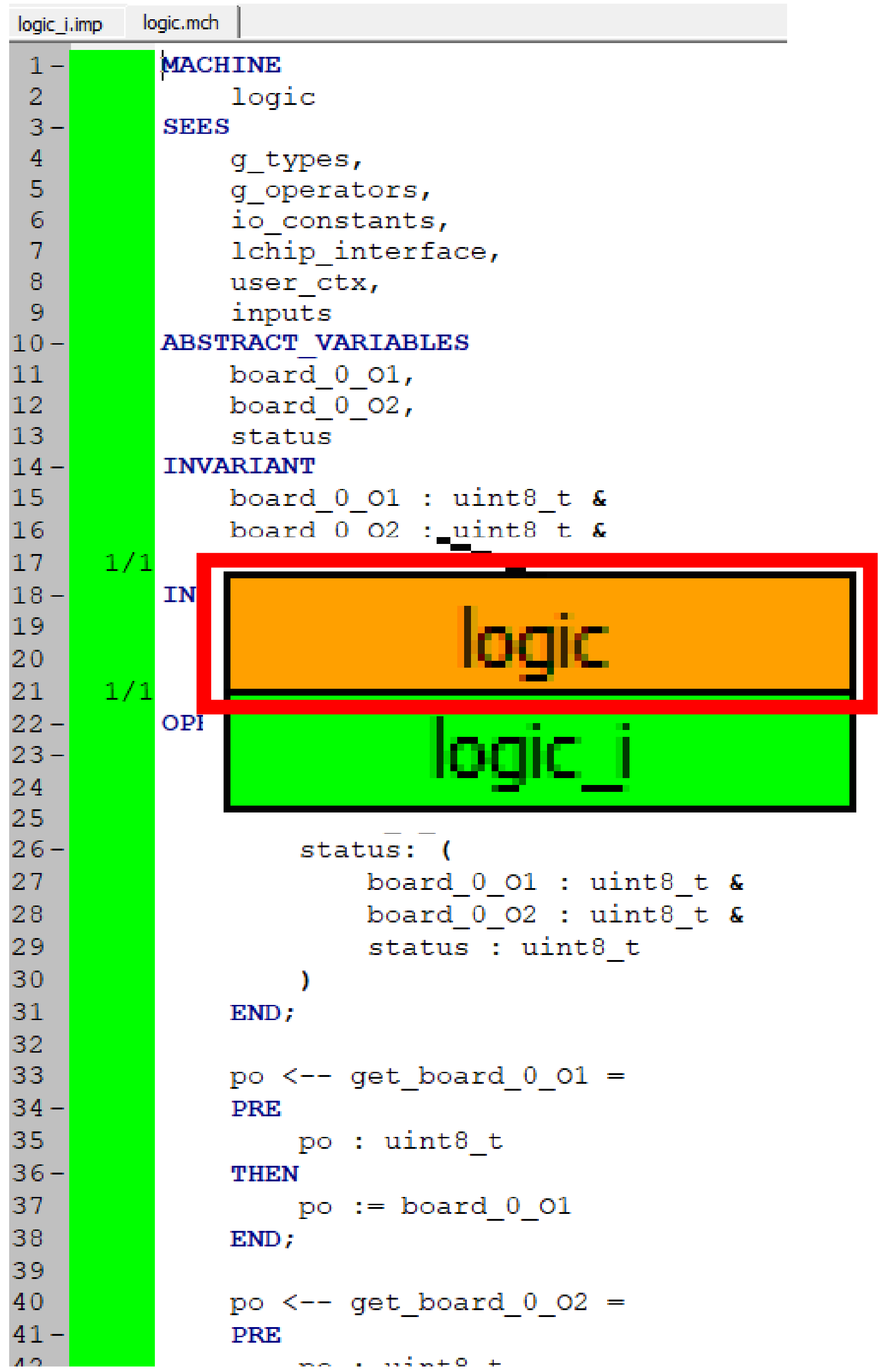
Combinatorial Function: your turn

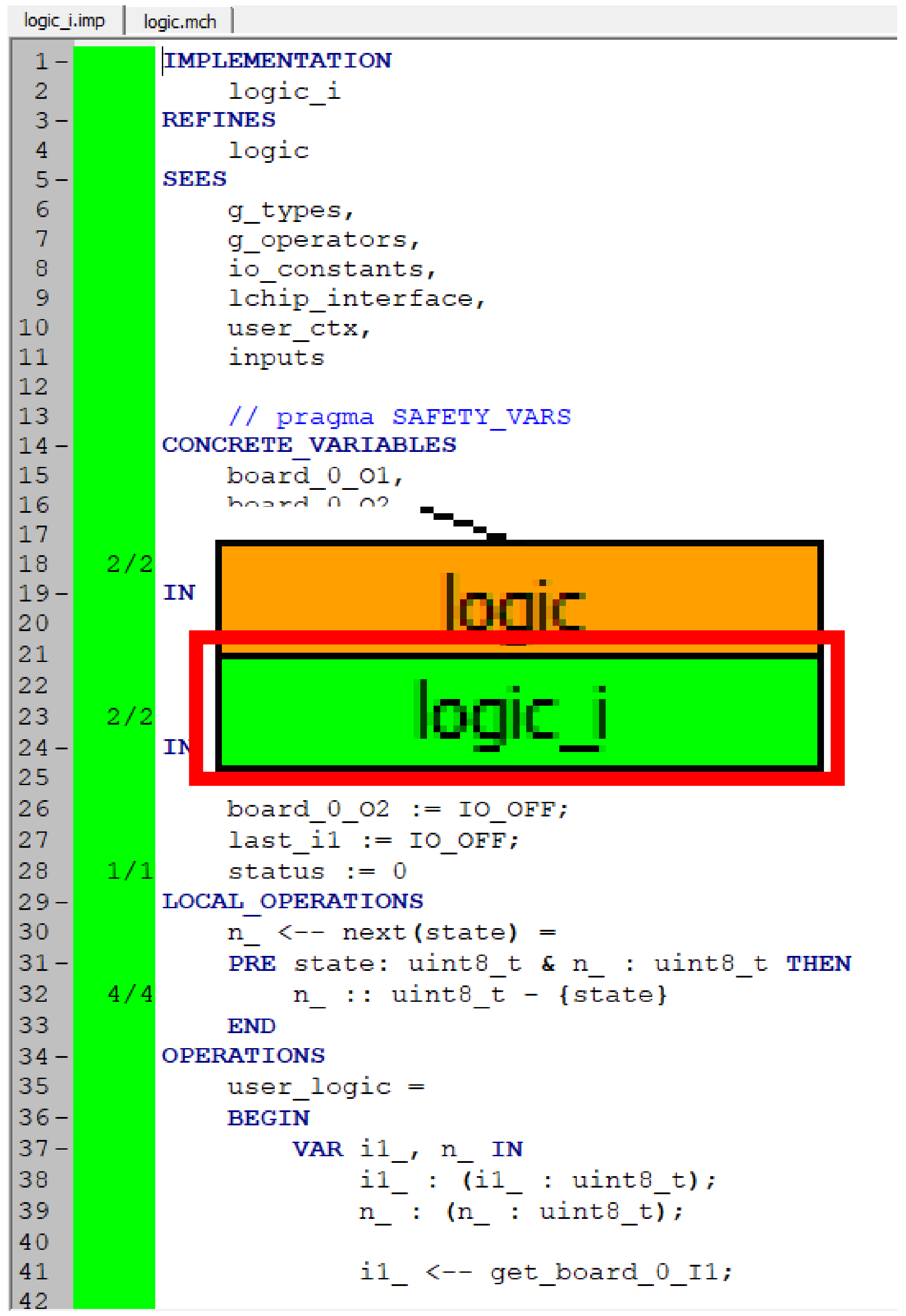
Implementation component

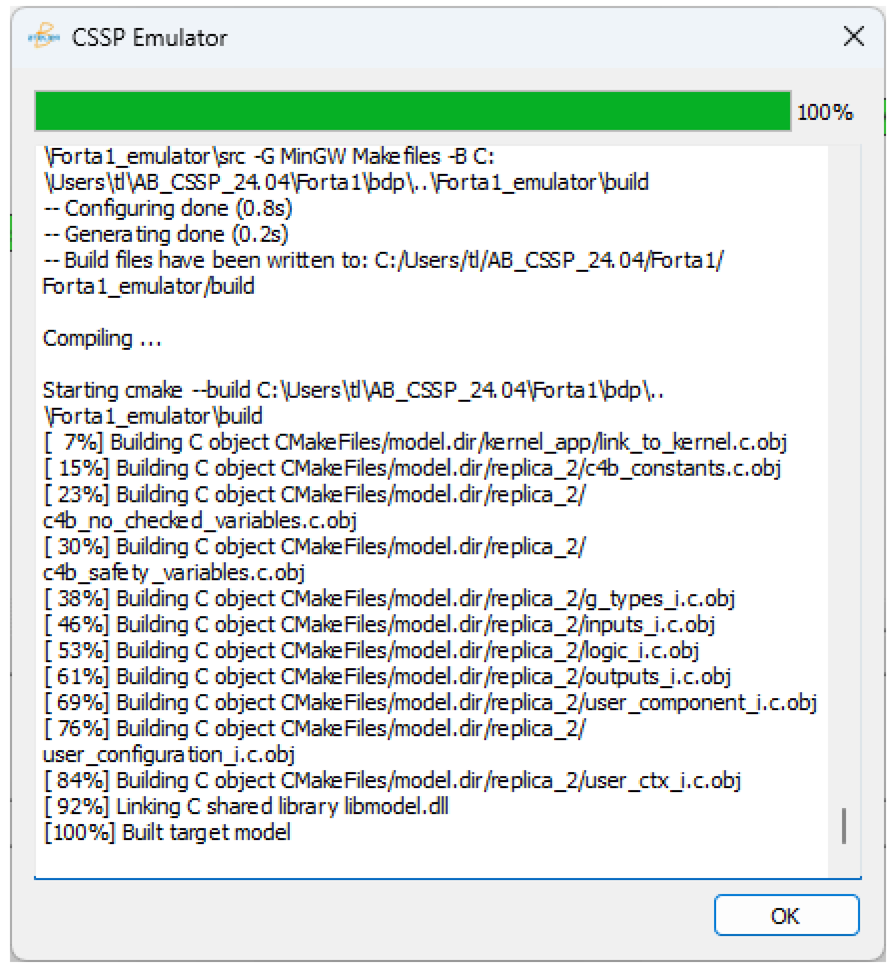
double-click to open

1. **Edit logic and logic_i components**
   Ctrl-S to save and prove
   Models should be « all green »

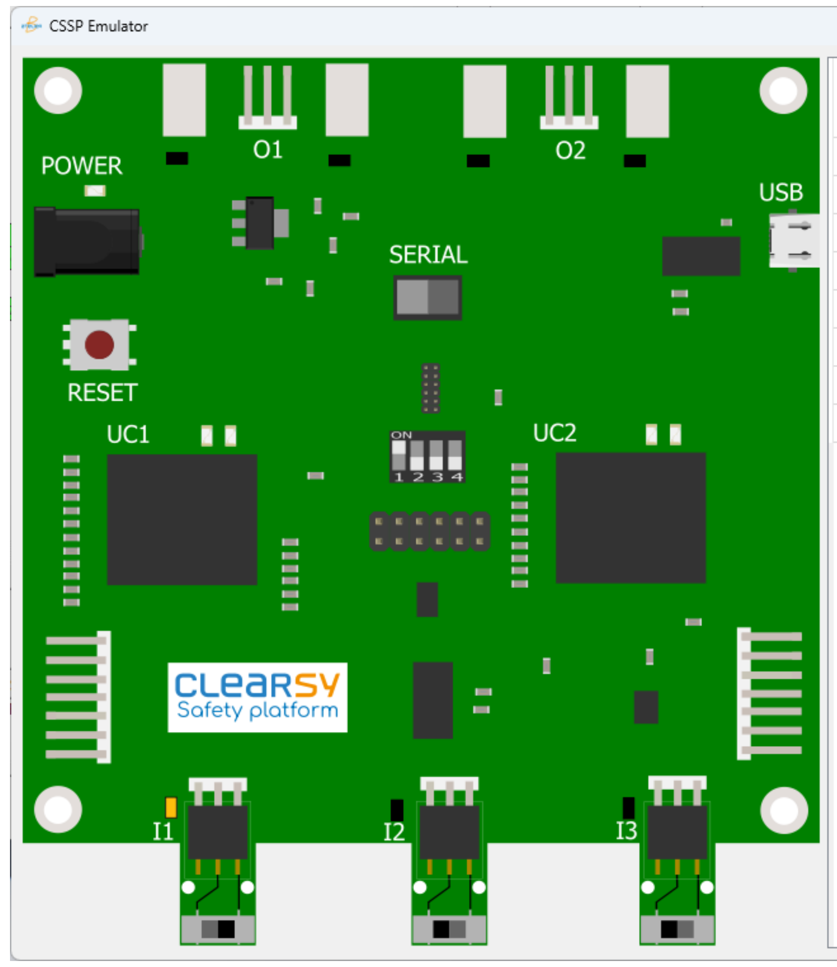
2. **Ctrl-D to start the emulator**
   Reach « 100% built target model »
   Press OK







3. **Click on I1 several times and observe the behaviour of *status***



Variables	Values
v_board_0_I1	1
v_board_0_I2	0
v_board_0_I3	0
v_board_0_O1	0
v_board_0_O2	0
v_last_i1	1
v_status	2
v_divergence_test_var	0
v_ms_tick	10186

ROBOSTAR

Combinatorial Function: your turn

Hard Work On Going !

Spoiler Alert: Solution on the next slide !

Combinatorial Function : the solution

Implementation component

double-click to open

logic

logic_i

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

modification

```
user_logic =
BEGIN
    VAR i1_ IN
        i1_ : (i1_ : uint8_t);

        i1_ <-- get_board_0_I1;

        IF last_i1 = IO_OFF THEN
            IF i1_ = IO_ON THEN
                status <-- next(status)
            END
        END;
        last_i1 := i1_
    END
END;

n_ <-- next(state) =
BEGIN
    IF state = 0 THEN n_ := 1
    ELSIF state = 1 THEN n_ := 2
    ELSE n_ := 0
    END
END;
```

We call the predefined operation *get_board_0_I1* to get the value of I1 and store it in *i1_*

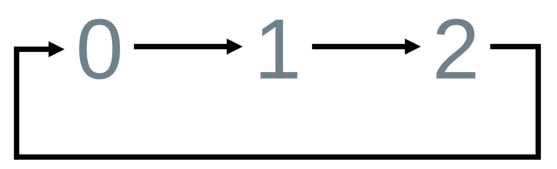
We call *next* to get the new value of *status*

ROBOSTAR

Developping a Synchronous Function

Synchronous / Timed Function

- I1 belongs to {IO_OFF, IO_ON}
- status belongs to 0..2
- Every time I1 changes from IO_OFF to IO_ON, status changes such as



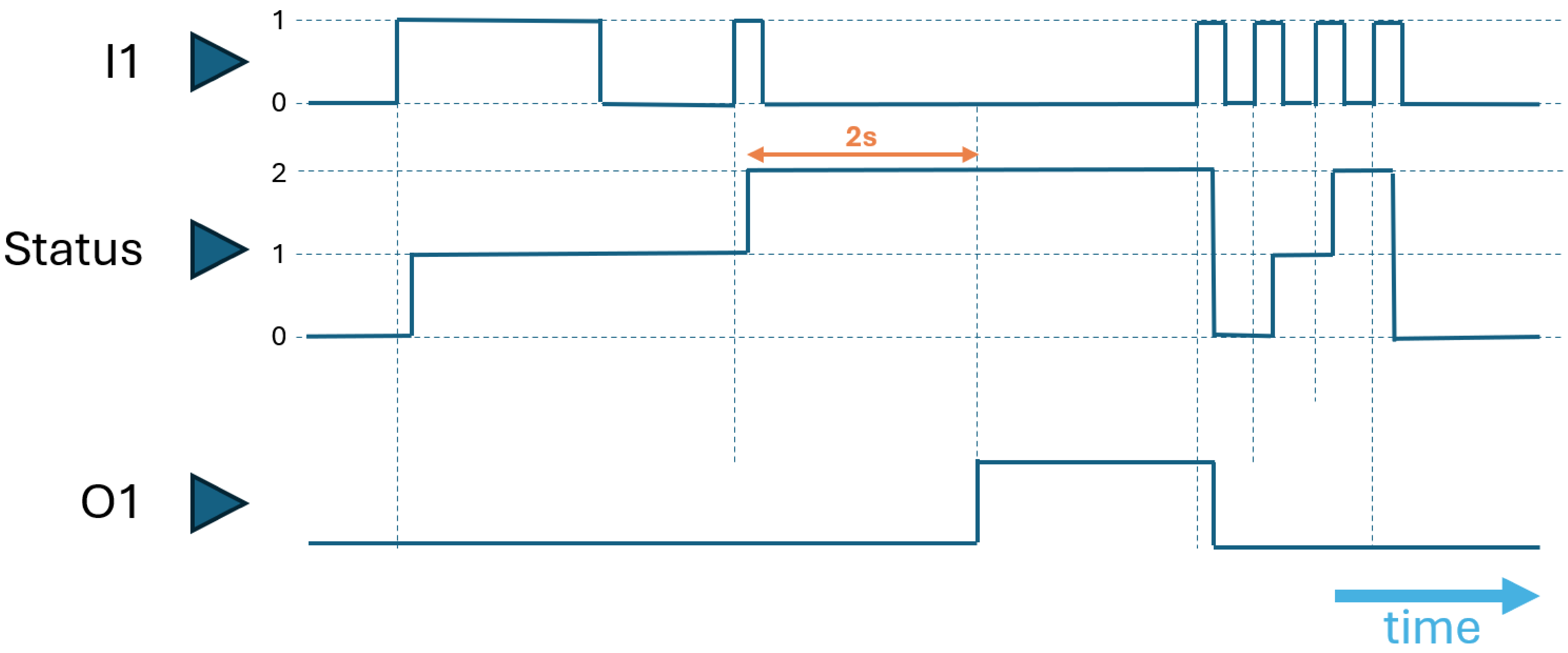
- If status = 2 during 2 s or more then sets O1 to IO_ON
- Otherwise sets O1 to IO_OFF

We need to specify a timer

We need to command output O1

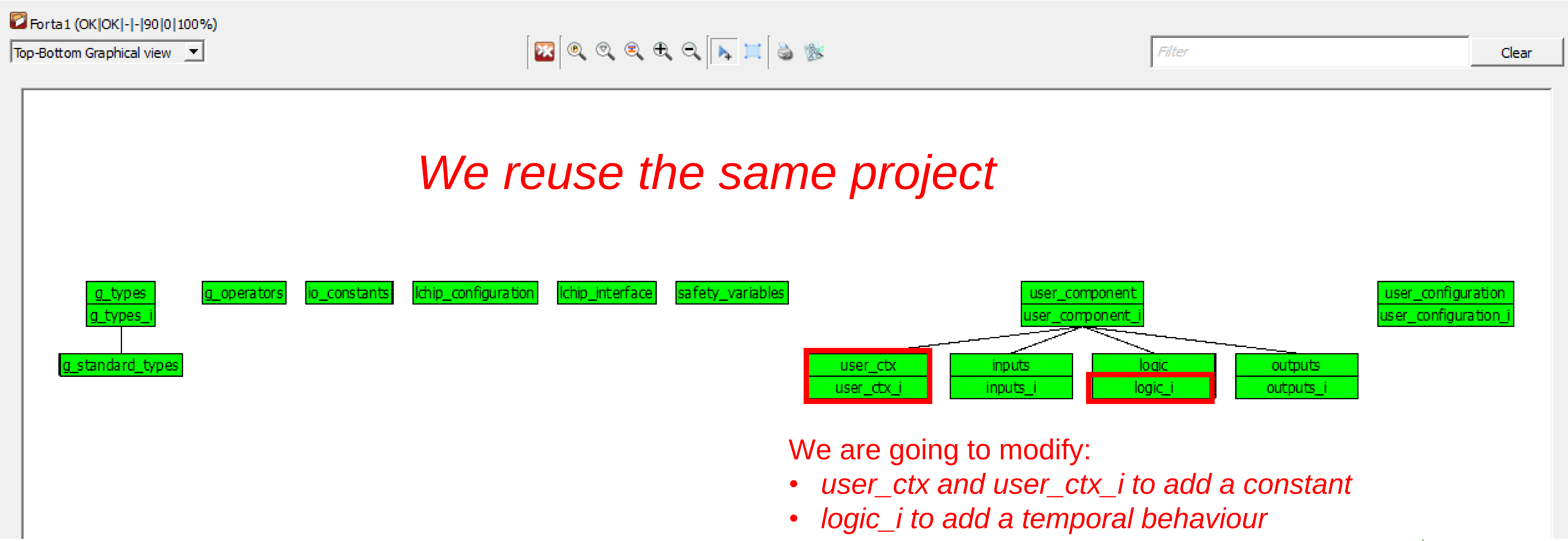
ROBOSTAR

Synchronous Function - scenario



ROBOSTAR

Synchronous / Timed Function



We reuse the same project

We are going to modify:

- *user_ctx and user_ctx_i to add a constant*
- *logic_i to add a temporal behaviour*

Synchronous / Timed Function

Specification component

double-click to open

user_ctx

user_ctx_i

- MACHINE
  - user_ctx
- SEES
  - g_types
- CONCRETE_CONSTANTS
  - DELAY
- PROPERTIES

addition

user_ctx.mch

```
MACHINE
    user_ctx
SEES
    g_types
CONSTANTS
    DELAY
PROPERTIES
    DELAY : uint32_t
END
```

CONSTANTS are defined and used with their properties whatever their values

DELAY is Integer

ROBOSTAR

University of York, UK

Synchronous / Timed Function

Implementation component

double-click to open

user_ctx

user_ctx_i

- **MACHINE**
  - user_ctx_i
- **REFINES**
  - user_ctx
- **SEES**
  - g_types
- **VALUES**

addition

user_ctx.mch | user_ctx_i.imp

```
IMPLEMENTATION
    user_ctx_i
REFINES
    user_ctx

    // pragma CONSTANTS
SEES
    g_types
VALUES
    DELAY = 2000
END
```

In

DELAY : uint32_t &

DELAY > 10 &

DELAY < 2

It is not possible to value DELAY: **it is a miracle**

Each CONSTANT should be valued to demonstrate it is not a miracle

i.e. *2000* belongs to uint32_t

Time returned by *get_ms_tick* OPERATION is in ms

Hence 2s == 2000 ms

ROBOSTAR

Synchronous / Timed Function

Specification component

double-click to open

logic

logic_i

logic_i.imp | logic.mch

- **MACHINE**
  - logic
- **SEES**
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- **ABSTRACT_VARIABLES**
  - board_0_O1
  - board_0_O2
  - status
- **INVARIANT**
- **INITIALISATION**
- **OPERATIONS**
  - user_logic
  - get_board_0_O1
  - get_board_0_O2

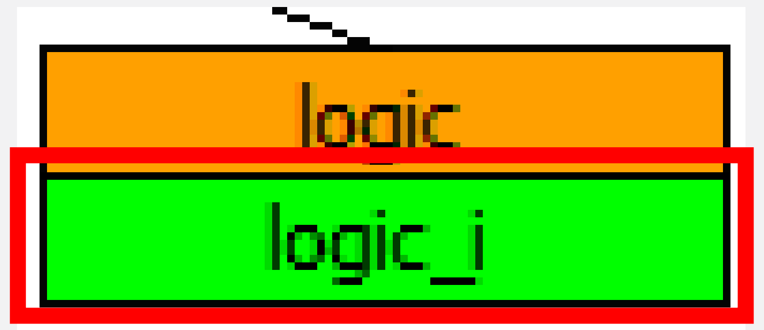
No change !

```
MACHINE
    logic
SEES
    g_types,
    g_operators,
    io_constants,
    lchip_interface,
    user_ctx,
    inputs
ABSTRACT_VARIABLES
    board_0_O1,
    board_0_O2,
    status
INVARIANT
    board_0_O1 : uint8_t &
    board_0_O2 : uint8_t &
    status : uint8_t
INITIALISATION
    board_0_O1 :: uint8_t ||
    board_0_O2 :: uint8_t ||
    status := 0
OPERATIONS
    user_logic = BEGIN
        board_0_O1,
        board_0_O2,
        status: (
            board_0_O1 : uint8_t &
            board_0_O2 : uint8_t &
            status : uint8_t
        )
    END;
```

Synchronous / Timed Function

Implementation component

double-click to open




```
MACHINE
    logic_i
REFINES
    logic
SEES
    g_types
    g_operators
    io_constants
    lchip_interface
    user_ctx
    inputs
CONCRETE_VARIABLES
    board_0_O1
    board_0_O2
    last_i1
    status
    t0_s_2
INVARIANT
INITIALISATION
LOCAL_OPERATIONS
    next
OPERATIONS
    user_logic
    next
    get_board_0_O1
    get_board_0_O2
```

addition

modification

logic_i.imp

```
CONCRETE_VARIABLES
    board_0_O1,
    board_0_O2,
    last_i1,
    status,
    t0_s_2
INVARIANT
    board_0_O1 : uint8_t &
    board_0_O2 : uint8_t &
    last_i1 : uint8_t &
    status : uint8_t &
    t0_s_2 : uint32_t
INITIALISATION
    board_0_O1 := IO_OFF;
    board_0_O2 := IO_OFF;
    last_i1 := IO_OFF;
    status := 0;
    t0_s_2 := 0
```

We need to store the last time *status* changed to the value 2

Timing information returned by the board is uint32_t

The initial value is 0, even if *status* is different from 2

ROBOSTAR

Synchronous / Timed Function

Implementation component

double-click to open

logic

logic_i

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
  - t0_s_2
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

get_ms_tick return the number of ms since last reboot

modification

```
user_logic =
BEGIN
    VAR i1_, t_, d_ IN
        i1_ : (i1_ : uint8_t);
        t_ : (            );
        d_ : (            );

        i1_ <-- get_board_0_I1;
        t_ <-- get_ms_tick;
        IF last_i1 = IO_OFF THEN
            IF i1_ = IO_ON THEN
                status <-- next(status);
                IF            THEN
                    t0_s_2 :=
                END
            END
        END;
        last_i1 := i1_;
```

Turn on *board_0_O1* if *status* has been 2 during at least 2s.
If not, turn it off

```
        END
    END
END;
```

Type *t_* used to store the current time

Type *d_* used to store the difference between current time and last time *status* changed to 2

What is the condition if we are going to modify the last time *status* changed to 2 ?

What value is assigned to *t0_s_2* ?

ROBOSTAR

Synchronous / Timed Function

Implementation component

double-click to open

logic

logic_i

- **MACHINE**
  - logic_i
- **REFINES**
  - logic
- **SEES**
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- **CONCRETE_VARIABLES**
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
  - t0_s_2
- **INVARIANT**
- **INITIALISATION**
- **LOCAL_OPERATIONS**
  - next
- **OPERATIONS**
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

modification

```
user_logic =
BEGIN
    VAR i1_, t_, d_ IN
        i1_ : (i1_ : uint8_t);
        t_ : (              );
        d_ : (              );

        i1_ <-- get_board_0_I1;
        t_ <-- get_ms_tick;
        IF last_i1 = IO_OFF THEN
            IF i1_ = IO_ON THEN
                status <-- next(status);
                IF          THEN
                    t0_s_2 :=
                END
            END
        END;
        last_i1 := i1_;
```

Turn on *board_0_O1* if *status* has been 2 during at least 2s.
If not turn off 01

```
        END
    END
END;
```

HINT1

To turn on an ouput OO

OO := IO_ON

To turn it off

OO := IO_OFF

HINT2

Substraction of two uint32_t with *sub_uint32*

Ex: vv := *sub_uint32*(aa,bb)

CONSTRAINT1

Condition only with values, no calculation

CONSTRAINT2

Only < or <= , no > or >= for condition

ROBOSTAR

Synchronous / Timed Function: your turn

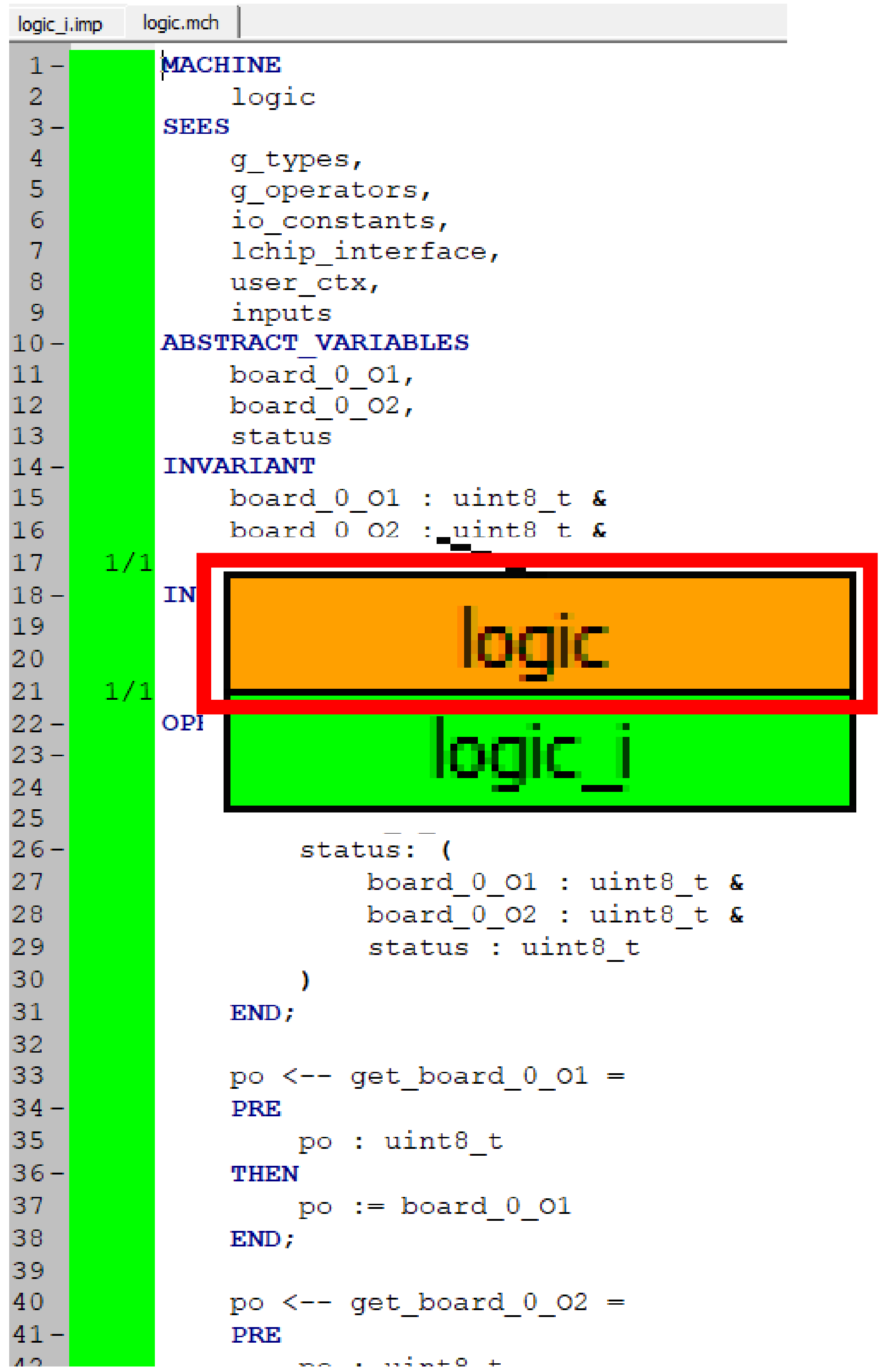
Implementation component

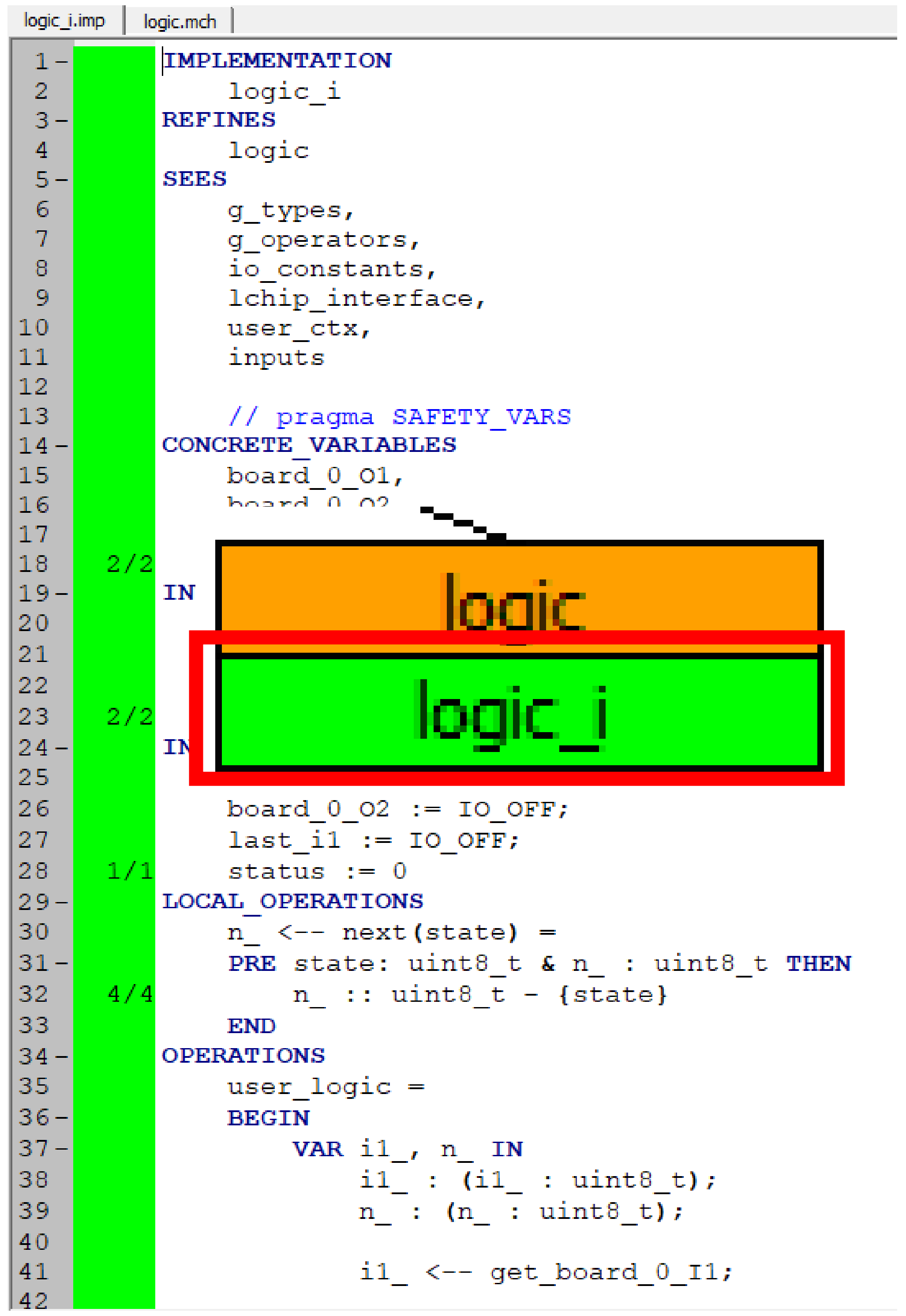
double-click to open

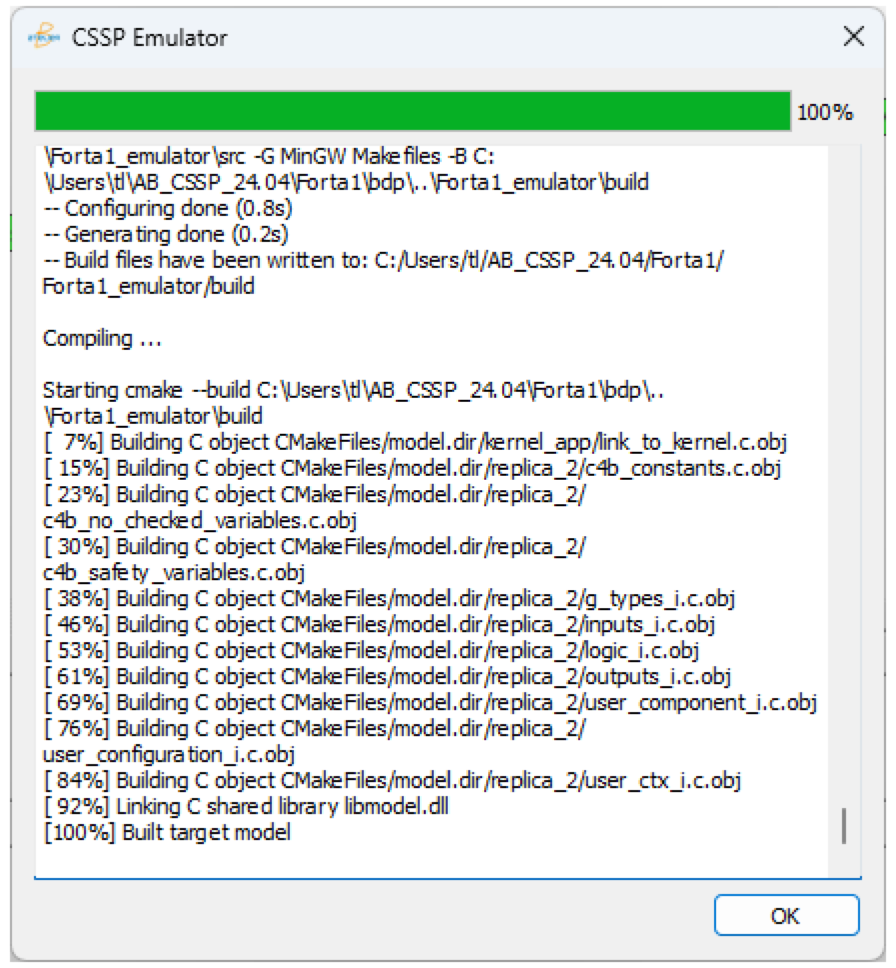
1 **Edit logic and logic_i components**
Ctrl-S to save and prove
Models should be « all green »

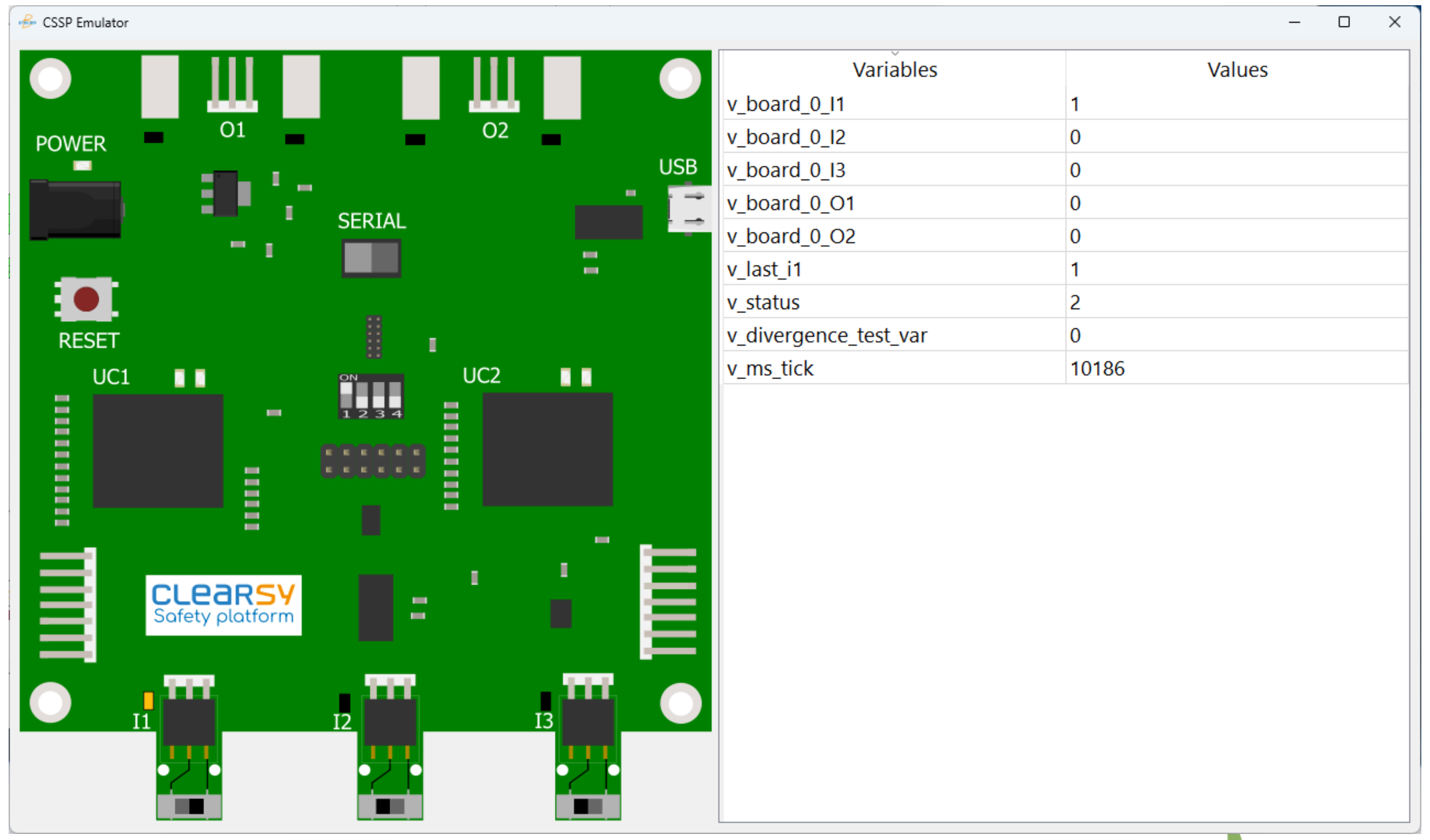
2 **Ctrl-D to start the emulator**
Reach « 100% built target model »
Press OK

3 **Click on I1 several times and observe the behaviour of *status* and *board_0_O1***









ROBOSTAR

Synchronous / Timed Function: your turn

Hard Work On Going !

Spoiler Alert: Solution on the next slide !

Synchronous / Timed Function : the solution

Implementation component

double-click to open

- MACHINE
  - logic_i
- REFINES
  - logic
- SEES
  - g_types
  - g_operators
  - io_constants
  - lchip_interface
  - user_ctx
  - inputs
- CONCRETE_VARIABLES
  - board_0_O1
  - board_0_O2
  - last_i1
  - status
  - t0_s_2
- INVARIANT
- INITIALISATION
- LOCAL_OPERATIONS
  - next
- OPERATIONS
  - user_logic
  - next
  - get_board_0_O1
  - get_board_0_O2

modification

```
user_logic =
BEGIN
    VAR i1_, t_, d_ IN
        i1_ : (i1_ : uint8_t);
        t_ : (t_ : uint32_t);
        d_ : (d_ : uint32_t);

        i1_ <-- get_board_0_I1;
        t_ <-- get_ms_tick;
        IF last_i1 = IO_OFF THEN
            IF i1_ = IO_ON THEN
                status <-- next(status);
                IF status = 2 THEN
                    t0_s_2 := t_
                END
            END
        END;
        last_i1 := i1_;
        board_0_O1 := IO_OFF;
        IF status = 2 THEN
            d_ := sub_uint32(t_, t0_s_2);
            IF DELAY <= d_ THEN
                board_0_O1 := IO_ON
            END
        END
    END
END;
```

Both *t_* and *d_* are unit32_t

status has just changed to 2.
It is the right time to store the current time

*d_* is meaningful only if *status* is 2

ROBOSTAR

Exercise to Go Further

- I1 {IO_OFF, IO_ON}
- I1 is noisy, changing value up to thousands times per second
- Only I1 constant during at least 100 ms have to be considered and are repeated on O1 – and O2 has to be IO_ON
- If no constant behavior is observed, then O2 has to be IO_OFF – in this case, the status of O1 is not considered (could be any value)
- Summary:
  - O2 is IO_ON when I1 has been constant during at least 100 ms and O1 has the status of the observed I1
  - O2 is IO_OFF when I1 has not been constant during the last 100 ms

Next

From RoboSim To The CLEARSY Safety Platform

Paulo Bezerra