# Formal methods - Lab1

Francesco Penasa

March 27, 2020

2020 03 27
enrico.magnago@unitn.it enricomagnano.com and patrick trentin website

1. write formal models in smv language for finite, infinite and timed systems
2. inspect possible executions
3. write and verify invariant and temporal properties

**exam**

1. write 2 formal models in SMV language
2. simulate these models
3. verify some properties

## 1 Introduction

1. SMV
2. NuSMV
3. nuXmv

nuXmv allows for the verification of

1. finite-state system
2. infinite-state systems
3. timed systems
4. only *synchronous* systems

```
nuXmv -int = interactive mode
help = show the list of all comands
reset
read_model [-i filename]
go = bdd engine only on finite system
go_bmc
go_msat
```

```
pick state [- ]
simulate []
print_current_state [] = prints out the current state
goto_state
show_traces
show_vars
quit = exit
```

## 2 First SMV model

```
MODULE main                                  -- mandatory
VAR                                                  -- define vars
        b0 : boolean;
ASSIGN
        init(b0) := FALSE;                   --initial constraint
        next(b0) := !b0;                     --constraint of the transition
```

types:

1. boolean: `x : TRUE`
2. enumerative: `s : ready, busy`
3. bounded integers: `n : 1..8;1` (between INT_MIN and INT_MAX)
4. integers
5. somethings
6. words
7. arrays: `x : array 0..10 of boolean; y : array -1..1 of red, green, orange z : array 1..10 of array 1..5 of boolean` (matrix)

```
MODULE main
VAR
        b0 : boolean;
        b1 : boolean;

ASSIGN
        init(b0) := FALSE;
        next(b0) := !b0
        init(y) := {1, 2, 3}; -- y can be either 1, 2 or 3

case
        c1 : e1;
        c2 : e2;
        TRUE : en;
esac

cond_expr ? ep1 : ep2
```

transition relation

```
next(a) := {a, a+1};
```

**not to do**

1. write multiple init for the same value;

## 3 Exercise

TODO before friday 2020 04 03