FORMAL METHODS
THE NUSMVMODEL CHECKER

Alessandro Artale

Faculty of Computer Science – Free University of Bolzano

artale@inf.unibz.it http://www.inf.unibz.it/∼artale/

Some material (text, figures) displayed in these slides is courtesy of:
M. Benerecetti, A. Cimatti, M. Fisher, F. Giunchiglia, M. Pistore, M. Roveri, R.Sebastiani.

The NuSMV language (nusmv.irst.itc.it)

- NuSMV consists of one or more modules and one must be called **main**.
- An SMV program consists of:
  - Type declarations of the system variables;
  - Assignments that define the valid initial states (e.g., `init(b0) := 0`).
  - Assignments that define the transition relation (e.g., `next(b0) := !b0`).
    - They can be **Non-Deterministic**: Several values in braces.
  - CTL or LTL specifications introduced by the keywords SPEC, LTLSPEC, respectively.

The NuSMV language (Cont.)

- NuSMV takes the specification of a model and a set of properties (either in CTL or LTL) as input.
- NuSMVoutput either *True* if the properties hold or *False* with a trace showing the failure.
- The set of states correspond to the set of all possible values for the variables.
- NuSMV uses !,&, |, -> for the boolean $\neg, \wedge, \vee, \Rightarrow$.
- NuSMV uses **G,F,X,U,A,E** for the temporal operators $\Box, \Diamond, \bigcirc, \mathcal{U}$, $\boxed{\text{P}}$, $\Diamond\!\!\!\!\boxed{\text{P}}$.

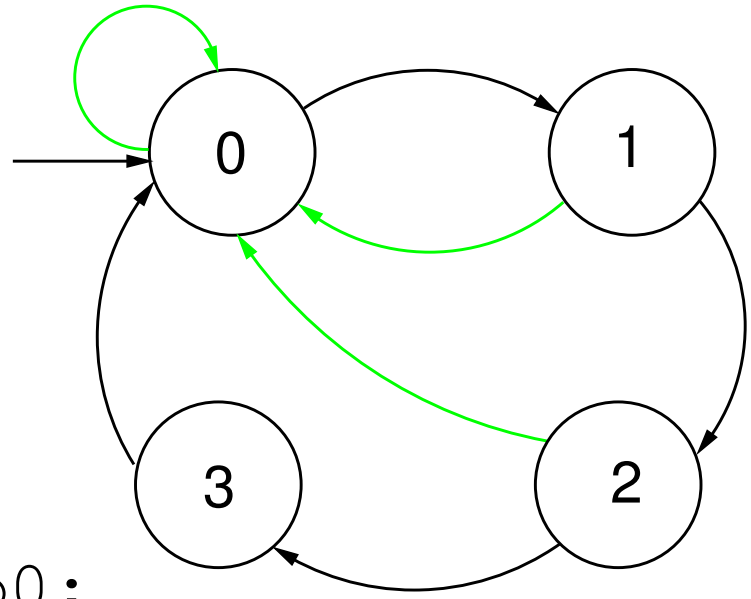
NuSMV: The modulo 4 counter with reset

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

 ASSIGN
   init(b0) := 0;
   next(b0) := case
                 reset = 1: 0;
                 reset = 0: !b0;
               esac;

   init(b1) := 0;
   next(b1) := case
                 reset: 0;
                 1    : ((!b0 & b1)|(b0 & !b1));
               esac;

 DEFINE
   out := b0 + 2*b1;
```



Modules in NuSMV

- NuSMV breaks a system description into *modules*.
- A module is instantiated when a variable having the module as its type is declared.
- Modules can have parameters.
- The notation `module-name.x` is used to access the variable `x` of the `module-name`.
- The keyword DEFINE is used to assign (the current value of) an expression to a symbol without the need to introduce a variable.
  - Defined symbols refer just to an expression then they cannot be assigned non-deterministically.

The "Counter" Example

```
 MODULE main
VAR
  bit0 : counter_cell(1);
  bit1 : counter_cell(bit0.carry_out);
  bit2 : counter_cell(bit1.carry_out);
SPEC
  AG AF bit2.carry_out

SPEC AG(!bit2.carry_out)

 MODULE counter_cell(carry_in)
VAR
  value : boolean;
ASSIGN
  init(value) := 0;
  next(value) := (value + carry_in) mod 2;
DEFINE
  carry_out := value & carry_in;
```

The "Counter" Example: Kripke Model

Do as an exercise!

Modules: Synchronous Vs. Asyncronous

- Modules, by default, are composed Synchronously
  - Each of the modules execute in parallel (e.g., the counter example).
- Using the keyword `process` modules are composed asynchronously
  - Each of the modules execute interleaving arbitrarily: at each tick one of them is non-deterministically chosen and executed.

Running NuSMV

- The main use of NuSMV is true an *interactive* shell.
- The user has the possibility to:
  1. Explore the possible executions called *Traces*;
  2. Construct the Model;
  3. Check specification and/or build counterexamples;
  4. etc.