AVoCS'02

Finite and Infinite Model Checking of Dual Transition Petri Net Models

Mauricio Varea,
Bashir Al-Hashimi, and
Michael Leuschel

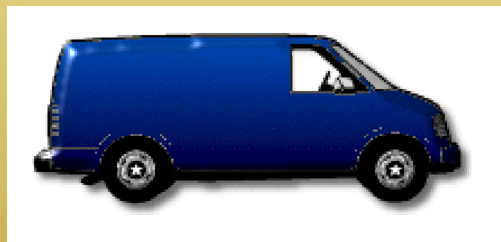
University of Southampton

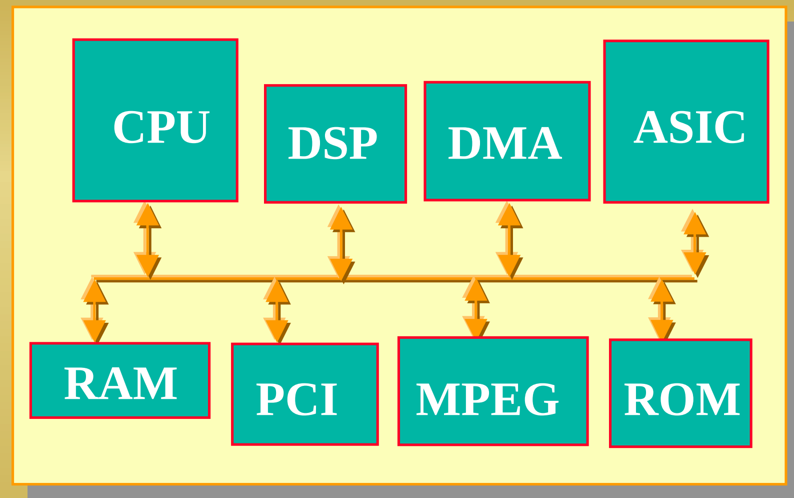
Department of Electronics and Computer Science

Overview

- Target: Embedded Systems
- Model: Dual Transition Petri Nets
- Verification:
  - Finite State Model Checking
  - Infinite State Model Checking
- Concluding Remarks.

Embedded Systems



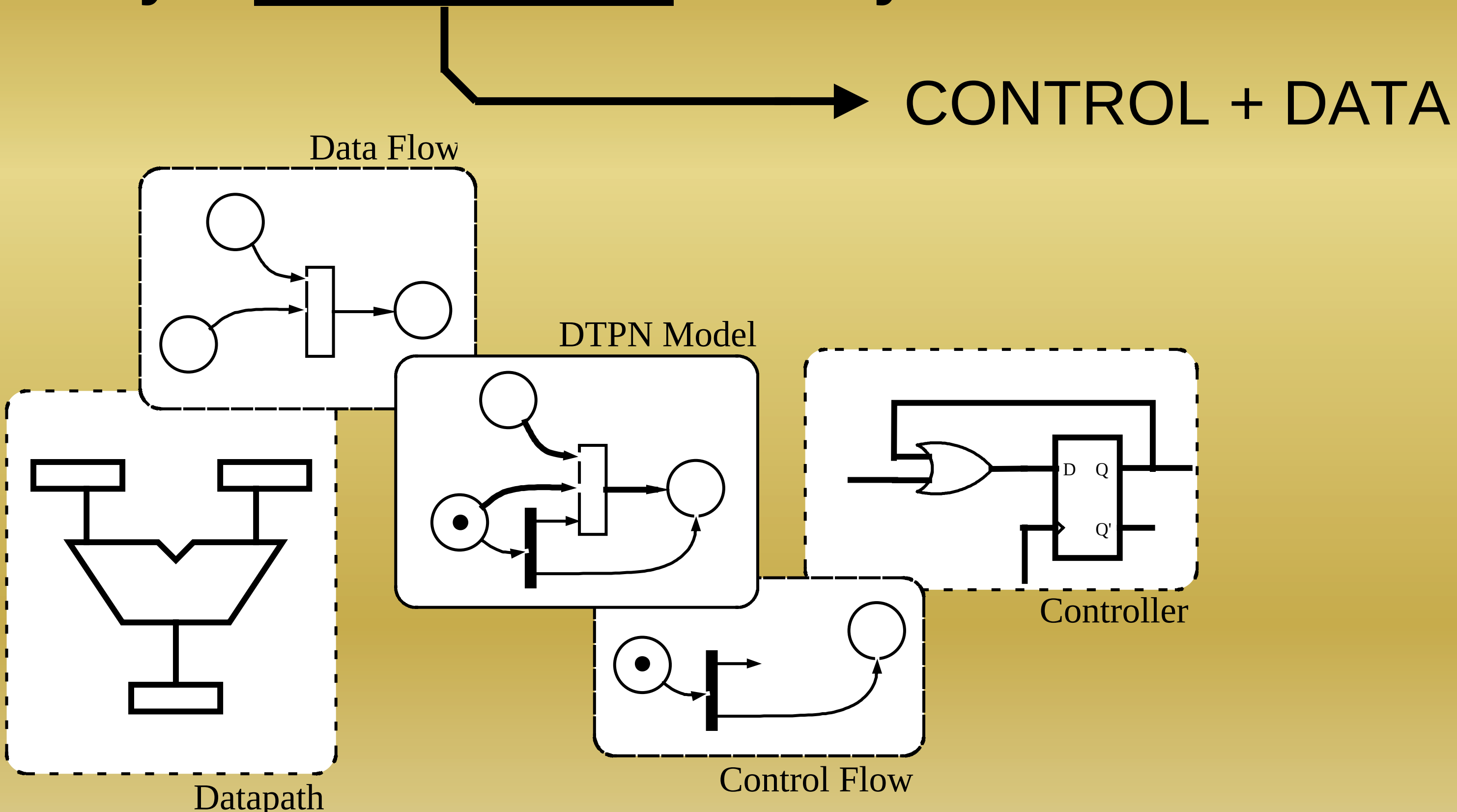


"A modern car may have up to 70 Embedded Systems"

[P. Thoma, proc. of DATE'99]

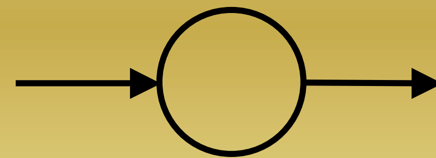
Dual Transition Petri Nets

- Complexity & Heterogeneity are key issues

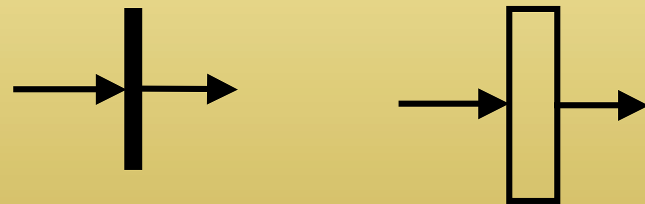


Dual Transition Petri Nets

- **Structure**
  - **Places** ⟶ **State / Storage**



  - **Transitions** ⟶ **Control and Data Flow**



- **Behaviour**
  - **Complex Marking**

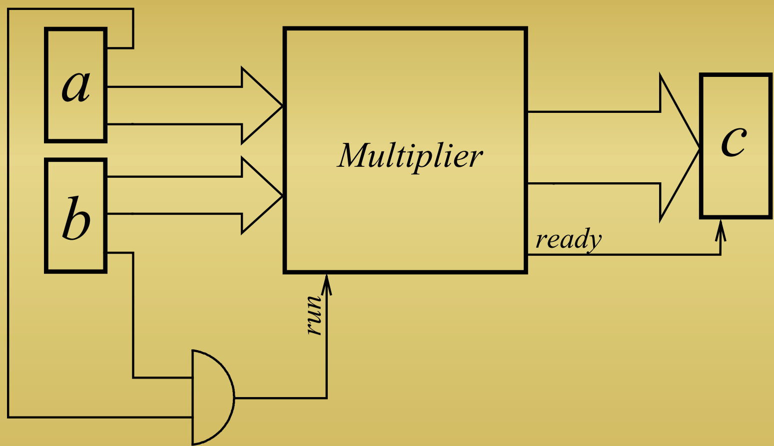
$$\mu : P \cancel{\to} \mathbb{N} \qquad \boxed{\mu : P \to \mathbb{C}}$$

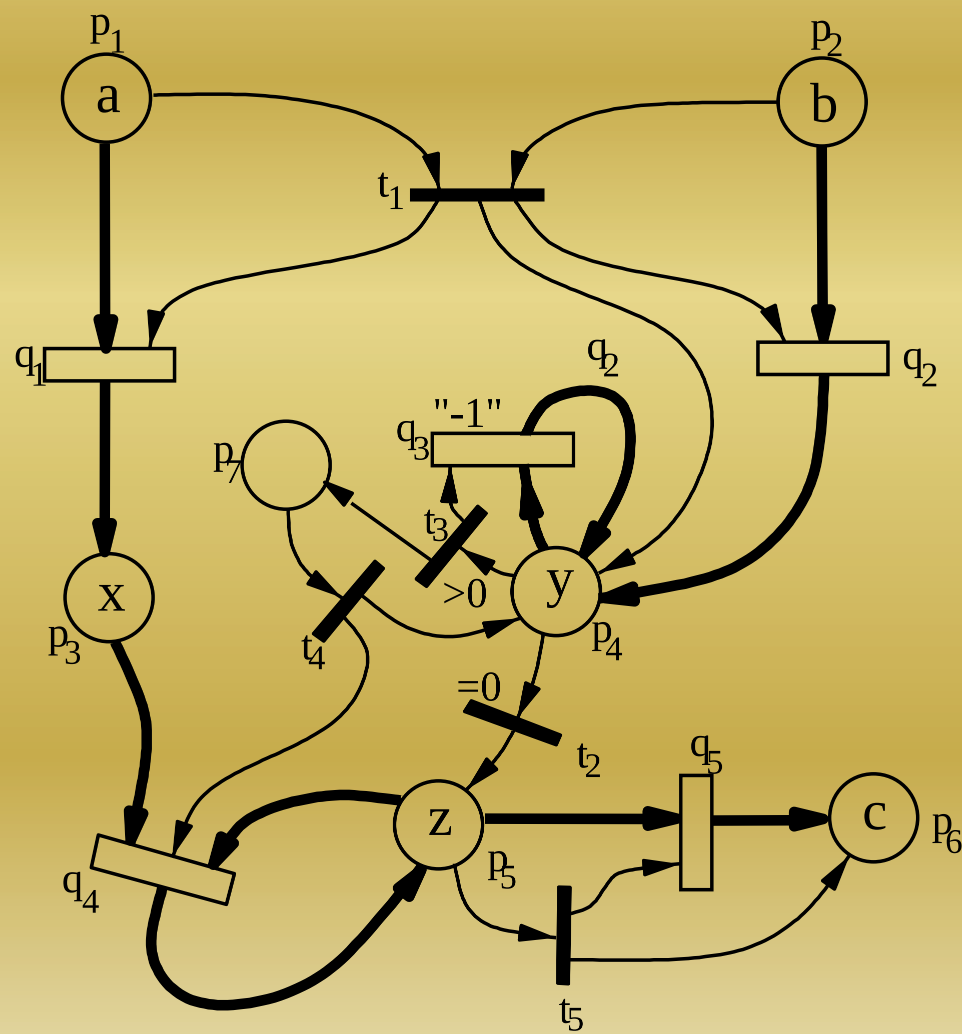
  - **Tokens and “values”**

$$\alpha_i \cdot e^{\mathrm{i} \cdot \pi/2 \cdot \gamma_i}$$

Dual Transition Petri Nets

- Example: Multiplier





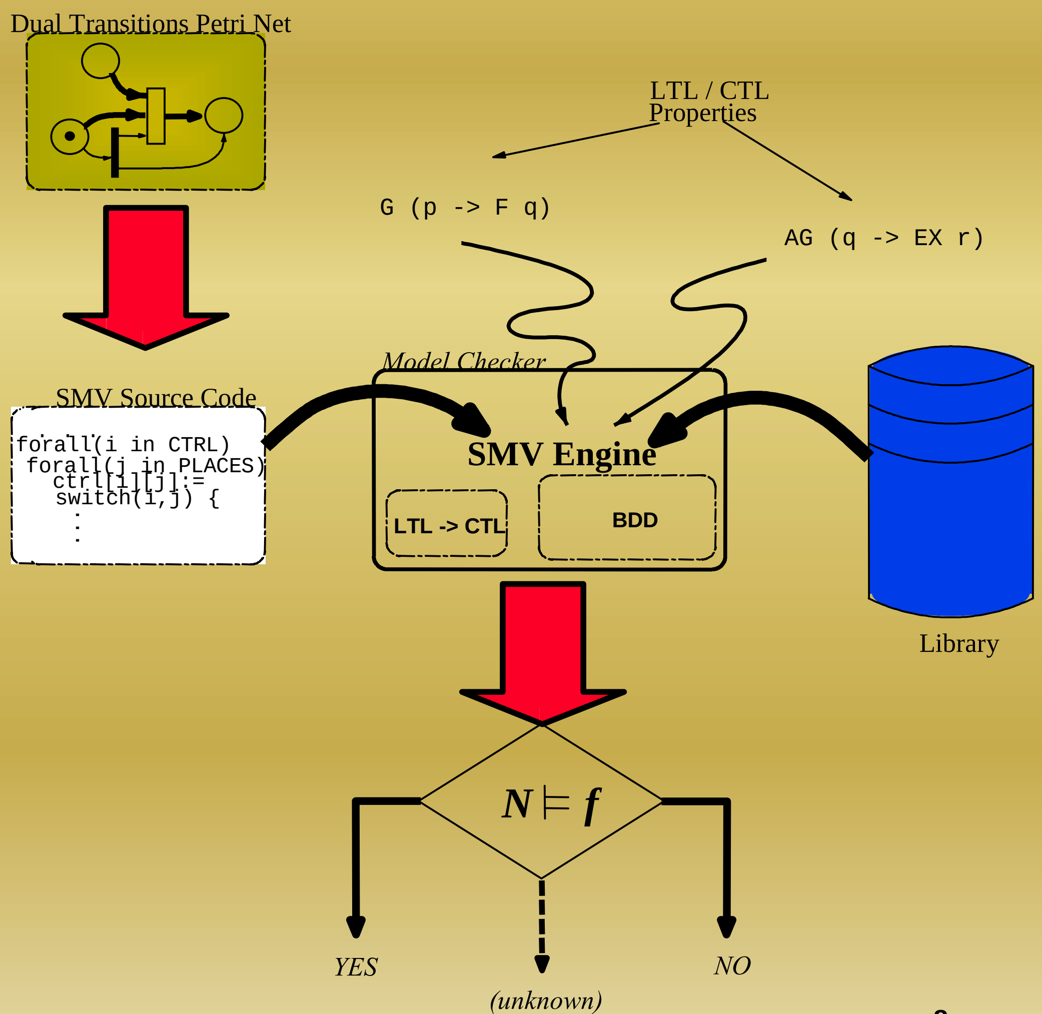
Verification

- Finite State Model Checking → SMV
- Infinite State Model Checking → CLP

University of Southampton

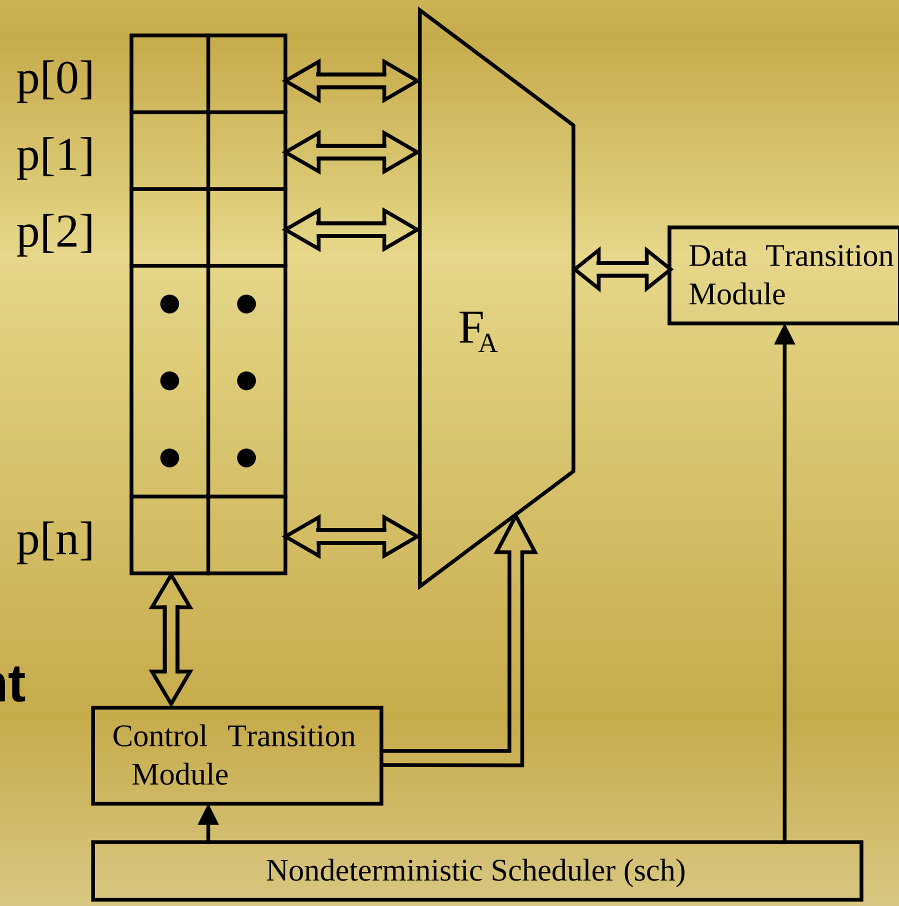
Finite State Model Checking

```
module ctrl_transition(p,preset,postset,g){
  en : boolean;
  en := &[ (preset[i] > 0) ->
     (p[i].phase >= preset[i]) : i = 1..(NN)]
     & g;

  for(i=1;i<=(NN);i=i+1)
    if (en) pp[i] := (p[i].phase - preset[i] + postset[i])
                     mod Kc;
       else pp[i] := p[i].phase;
}

module data_transition(p,preset,postset,ctrl,h,k,dl){
  en : boolean;
  en := (k in ctrl) & (~dl);

  for(i=1;i<=(NN);i=i+1)
    sum[i] := (p[i].modulus * preset[i]) mod Kd;
  for(i=1;i<=(NN);i=i+1)
    if (en) pp[i] := ( postset[i] * ((+sum) + h) ) mod Kd;
       else pp[i] := p[i].modulus;
}

typedef place;

module guard(op,sgn,val);
```



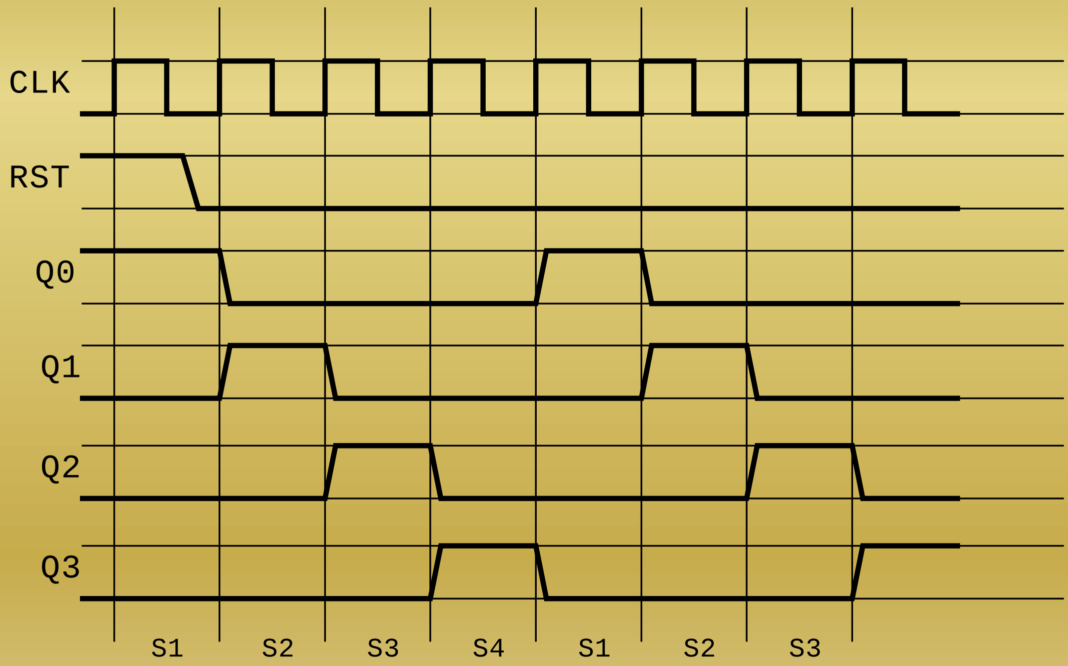
Finite State Model Checking

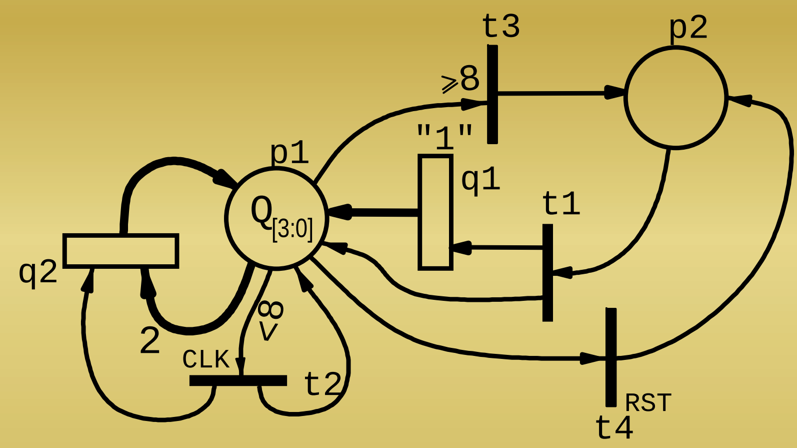
- LTL / CTL
  - Logic Operators
  - Temporal Operators
  - Path Quantifier
- Tool used: Cadence SMV
  - Non-deterministic scheduler
  - Deadlock free fairness constraint



Finite State Model Checking

- **Example: 4-bit Ring Counter**





G(p[1]=1 -> X p[1]=2)

G(p[1]=2 -> X p[1]=4)

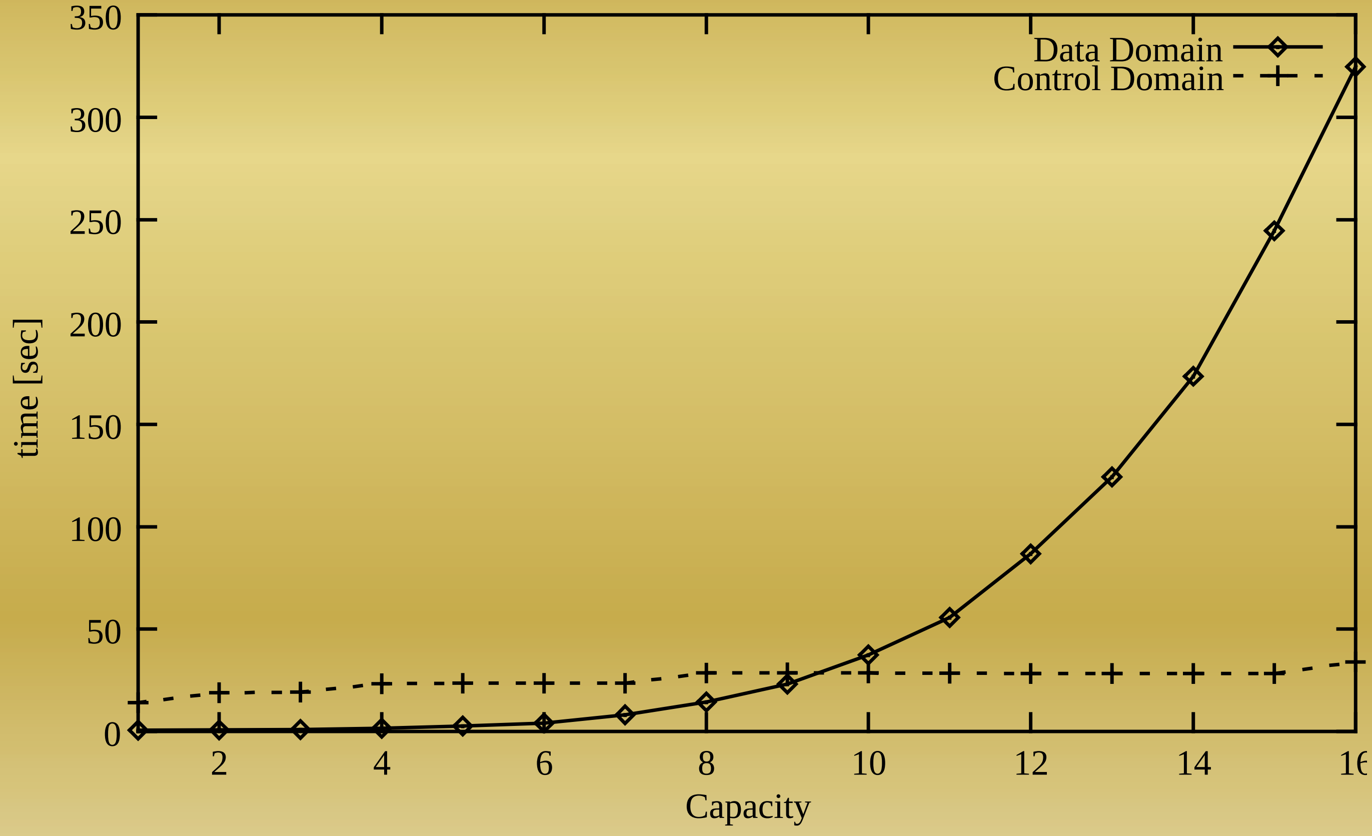
G(p[1]=4 -> X p[1]=8)

G(p[1]=8 -> F p[1]=1)

University of Southampton

Finite State Model Checking

- State Space Exploration



University of Southampton

Infinite State Model Checking

- **Nondeterministic Finite Automaton = Infinite State System**
- **Verification of Infinite State Systems is UNDECIDABLE, therefore, restrictions are needed in order to make it DECIDABLE**
- **CLP is ideal for NP-hard problems based on constraints**

Concluding Remarks

- DTPN models have a very tight control and data flow representation, which is suitable for Embedded Systems design.
- safe DTPN models → SMV
- k-bounded DTPN models → CLP
- unbounded DTPN models → CLP
- Future Work:
  - Applicability to Hw/Sw Co-Synthesis.