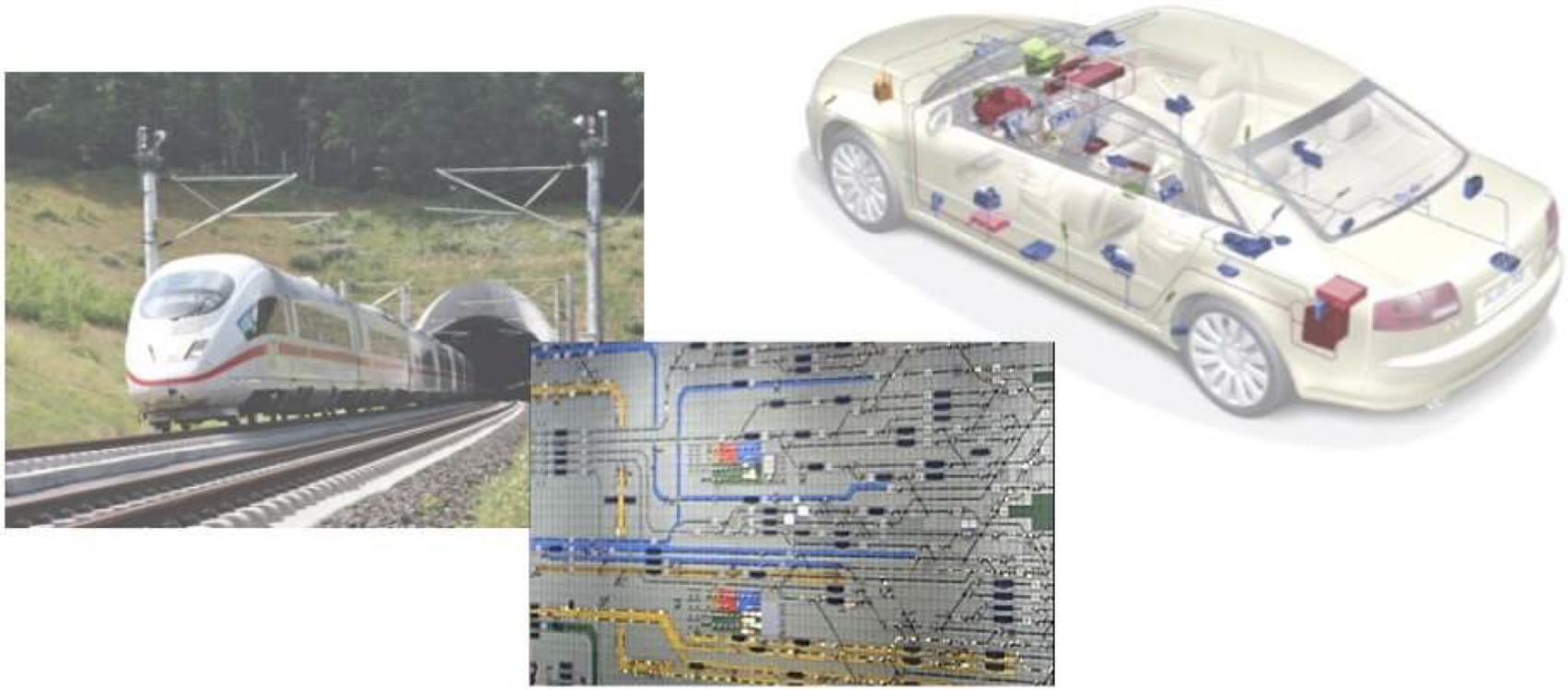
Model Checking and Systems Verification (MCSV)

Instructor; M.K. Srivas
TA: Zubin Duggal

August 11, 2020

Embedded Systems



- Embedded (HW+SW+FW) systems are a complex parts of real life machines

Model Checking: Motivation

- Traditional methods: Simulation and testing

Model Checking: Motivation

- Traditional methods: Simulation and testing
  - Do not guarantee 100% coverage

Model Checking: Motivation

- Traditional methods: Simulation and testing
  - Do not guarantee 100% coverage
  - Expensive; Labour intensive and time consuming

Model Checking: Motivation

- Traditional methods: Simulation and testing
  - Do not guarantee 100% coverage
  - Expensive; Labour intensive and time consuming
- Can we have methods that guarantee 100% coverage

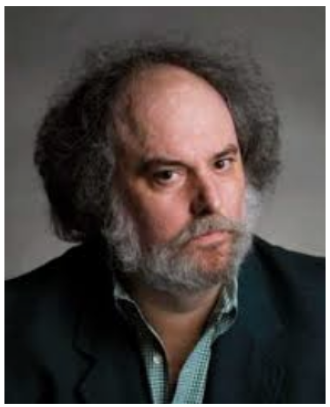
Model Checking: Motivation

- Traditional methods: Simulation and testing
  - Do not guarantee 100% coverage
  - Expensive; Labour intensive and time consuming
- Can we have methods that guarantee 100% coverage
  - Formal Verification: Methods based on Logic and Automata theory

Model Checking: Motivation

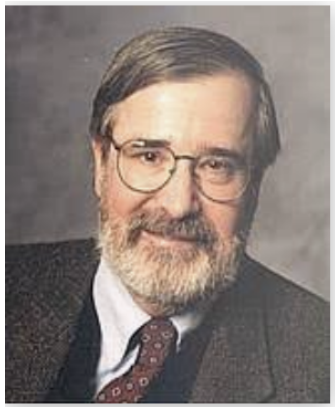
- Traditional methods: Simulation and testing
  - Do not guarantee 100% coverage
  - Expensive; Labour intensive and time consuming
- Can we have methods that guarantee 100% coverage
  - Formal Verification: Methods based on Logic and Automata theory
  - Model Checking is one such popular techinique

Model Checking: Cited for Turing Award in 2007 (Invented: 1986)



ACM Turing Award Hon...
cs.utexas.edu

E. Allen Emerson



Introduction to Model C...
moves.rwth-aachen.de

Edmund Clarke

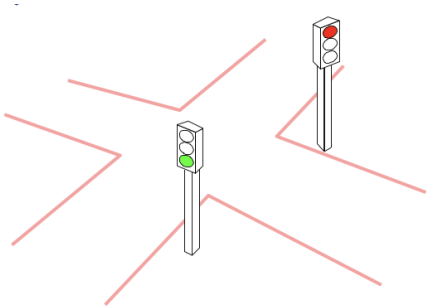
Joseph Sifakis | French com...
britannica.com

Joseph Sifakis

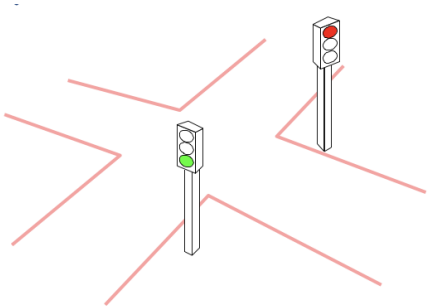
CITATION

Together with E. Allen Emerson and Joseph Sifakis, for their role in developing Model-Checking into a highly effective verification technology that is widely adopted in the hardware and software industries.

What Kind of Systems and Properties?

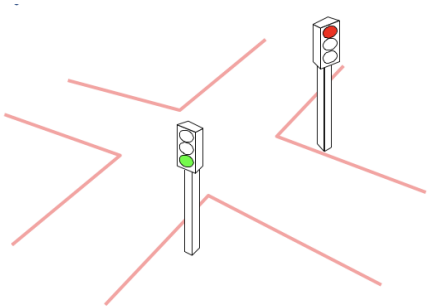


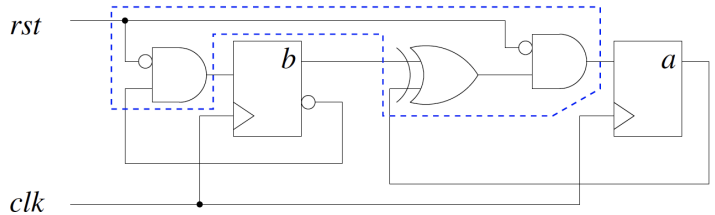
What Kind of Systems and Properties?



"traffic lights should never be green at the same time"

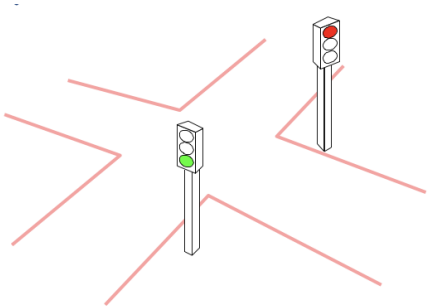
What Kind of Systems and Properties?



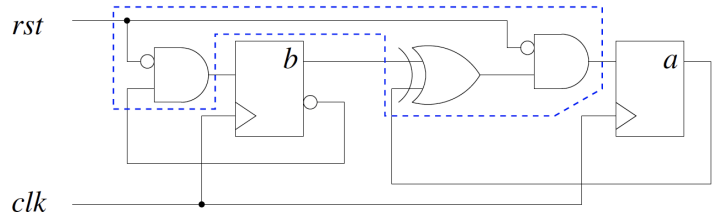


"traffic lights should never be green at the same time"

What Kind of Systems and Properties?

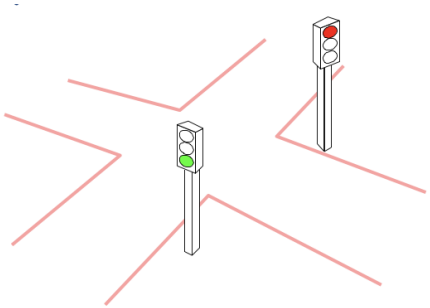


"traffic lights should never be green at the same time"

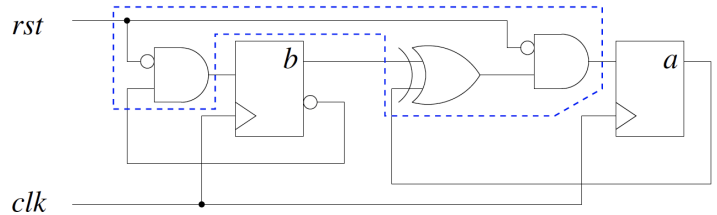


"the counter value ('ab') is always ≤ 2"

What Kind of Systems and Properties?



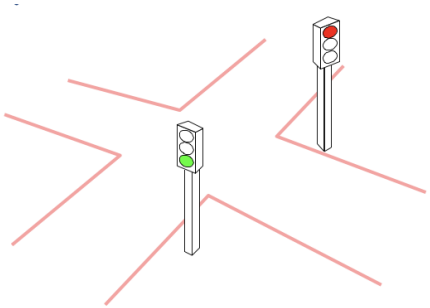
"traffic lights should never be green at the same time"



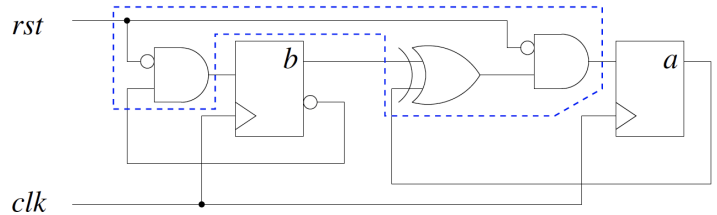
"the counter value ('ab') is always ≤ 2"

```
0: x=0;
1: y=0;
2: while (1) {
3:   x = (x+1) mod 2;
4:   y = x+1; }
```

What Kind of Systems and Properties?



"traffic lights should never be green at the same time"

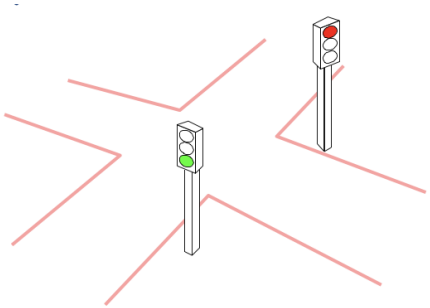


"the counter value ('ab') is always ≤ 2"

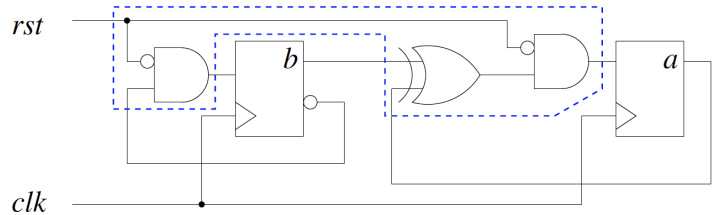
```
0: x=0;
1: y=0;
2: while (1) {
3:    x = (x+1) mod 2;
4:    y = x+1; }
```

"@line 3: $y \geq x$ *always* holds"
"@line 3: x=0 holds infinitely often"

What Kind of Systems and Properties?



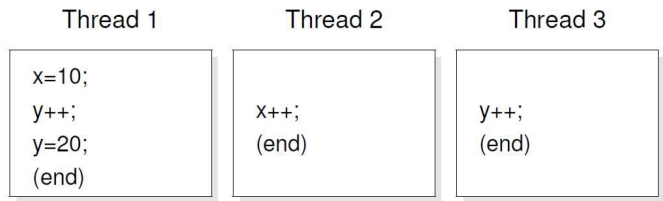
"traffic lights should never be green at the same time"



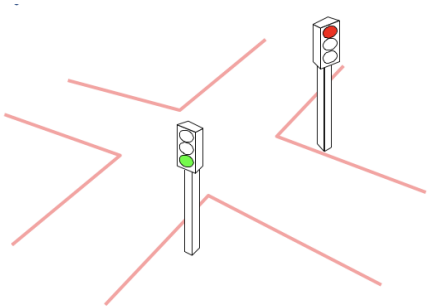
"the counter value ('ab') is always ≤ 2"

```
0: x=0;
1: y=0;
2: while (1) {
3:    x = (x+1) mod 2;
4:    y = x+1; }
```

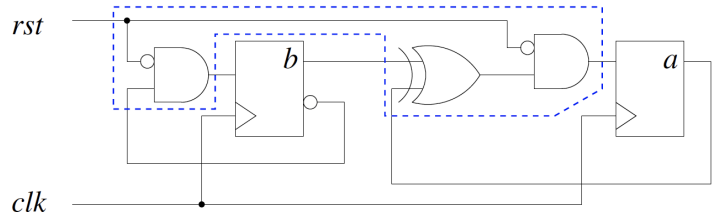
"@line 3: $y \geq x$ *always* holds"
"@line 3: x=0 holds infinitely often"



What Kind of Systems and Properties?



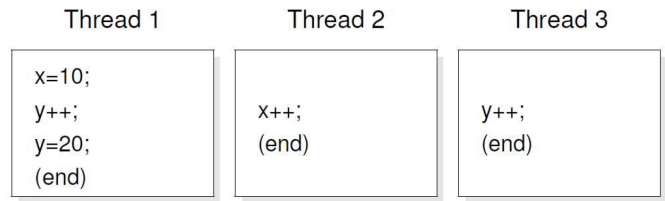
"traffic lights should never be green at the same time"



"the counter value ('ab') is always ≤ 2"

```
0: x=0;
1: y=0;
2: while (1) {
3:    x = (x+1) mod 2;
4:    y = x+1; }
```

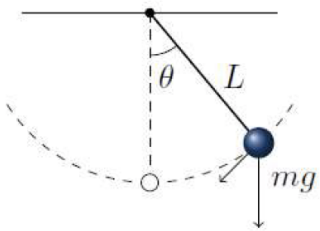
"@line 3: $y \geq x$ *always* holds"
"@line 3: x=0 holds infinitely often"

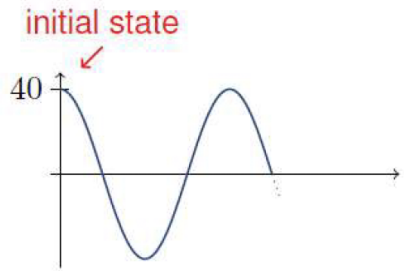


"the program has no data races"

What Kind of Systems Not

An Example





Our model:

$$\frac{\delta^2\theta}{\delta t^2} = -\frac{g}{L}\sin\theta$$

Plot of θ over time

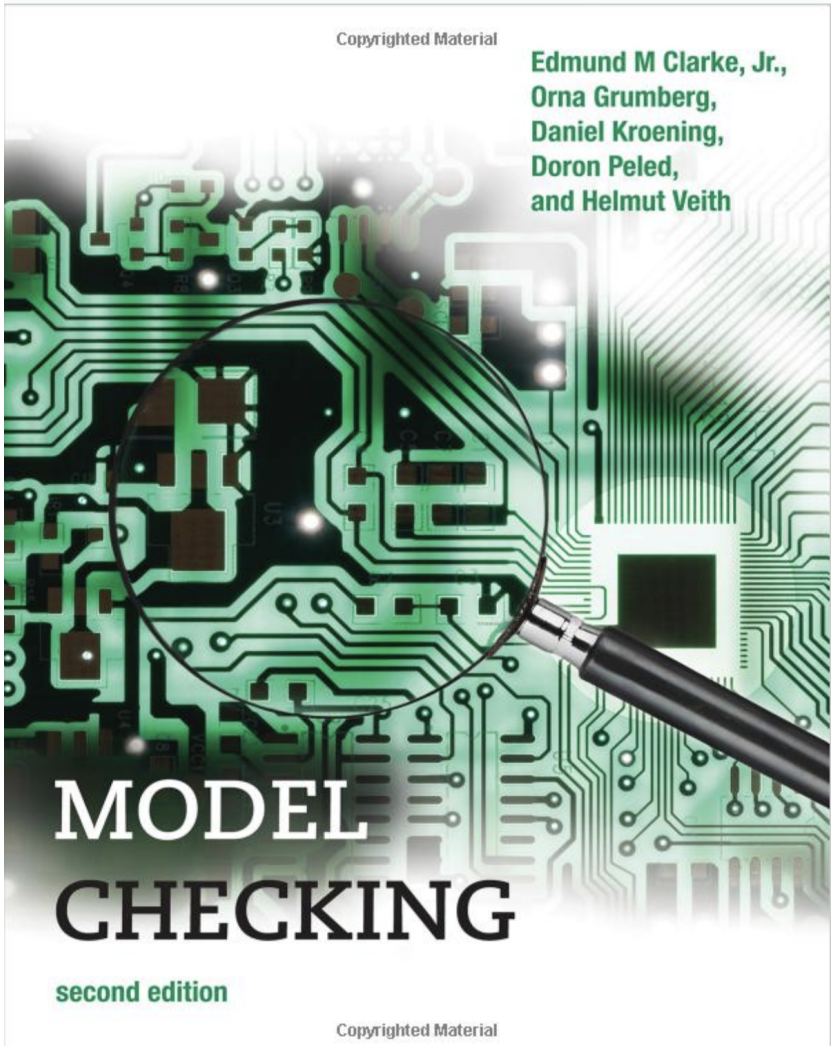
Brief Course Outline

1. Modeling Systems (Kripke Structures)
2. Specifying Properties (Temporal logics: CTL, LTL, CTL*)
3. Verifying Properties (Model Checking for Temporal logics)
4. Abstraction Techniques
5. Automata-Based Model Checking

Course Work and Grade Distribution

1. Written Assignments: 30%
2. Tool Exercises: 30%
3. 2 Quizzes: 40%
   - One can be a small programming project

Reference Text Book



eTextBook will be made available for class usage on shared (sign-up sheet) basis

Access details will be shared on Moodle

Some Definitions

- Bounded Satisfaction of a path: $\pi \models_k \phi$
  - SAT of π requires inspecting only a k-long (or shorter) π-prefix
  - π is k-bounded lasso $\implies$ $\pi \models_k \phi$

Some Definitions

- Bounded Satisfaction of a path: $\pi \models_k \phi$
  - SAT of π requires inspecting only a k-long (or shorter) π-prefix
  - π is k-bounded lasso $\implies \pi \models_k \phi$
- Bounded satisfaction by a TS M: $M \models_k \phi$
  - every lasso-shaped k-bounded path π in M satisfies ϕ

Some Definitions

- Bounded Satisfaction of a path: $\pi \models_k \phi$
  - SAT of π requires inspecting only a k-long (or shorter) π-prefix
  - π is k-bounded lasso $\implies$ $\pi \models_k \phi$
- Bounded satisfaction by a TS M: $M \models_k \phi$
  - every lasso-shaped k-bounded path π in M satisfies ϕ
- A *CompletenessThreshold* (CT) K for an (M, ϕ) is a K s.t.:
  - $M \models_K \phi \implies M \models \phi$

Some Definitions

- Bounded Satisfaction of a path: $\pi \models_k \phi$
  - SAT of π requires inspecting only a k-long (or shorter) π-prefix
  - π is k-bounded lasso $\implies$ $\pi \models_k \phi$
- Bounded satisfaction by a TS M: $M \models_k \phi$
  - every lasso-shaped k-bounded path π in M satisfies ϕ
- A *CompletenessThreshold* (CT) K for an (M, ϕ) is a K s.t.:
  - $M \models_K \phi \implies M \models \phi$
  - why should such a K exist at all?