Northeastern University
Khoury College of
Computer Sciences

System Specification, Verification, and Synthesis CS 4830 / 7485

Stavros Tripakis

2. Systems and system design methods

Today:

- NuXMV and Spin: quick demo
- Systems
- System design methods

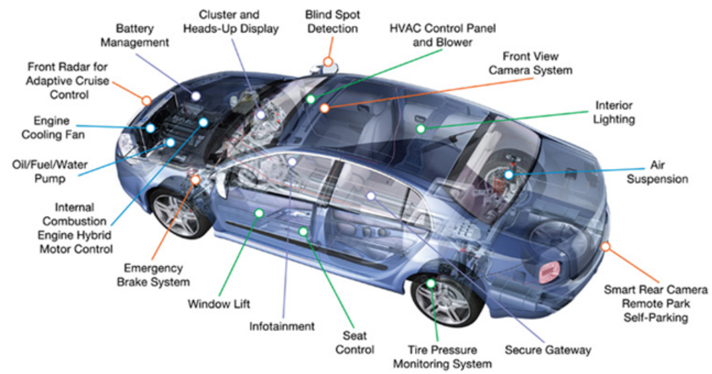
Quick demo

NuXMV and Spin

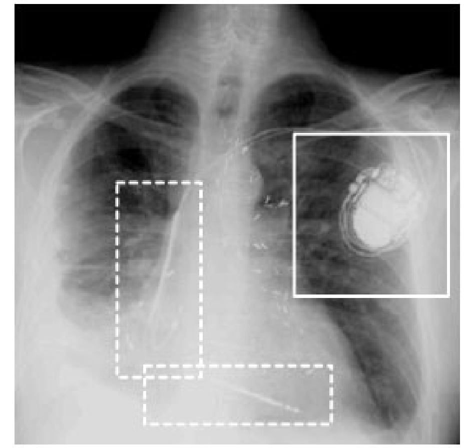
SYSTEMS

Examples of systems

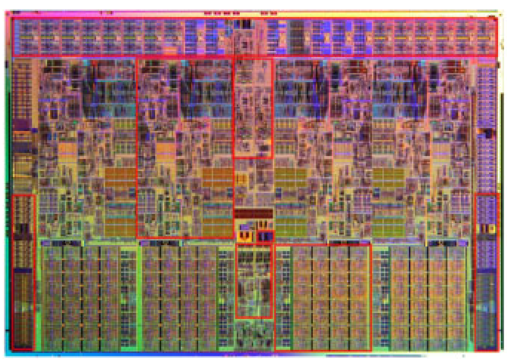
Aerospace/defense



Automotive

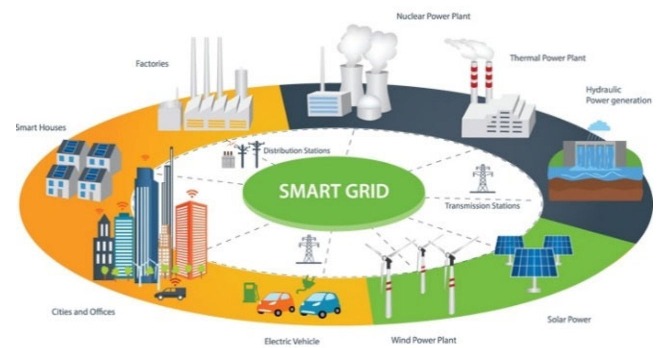


Medical



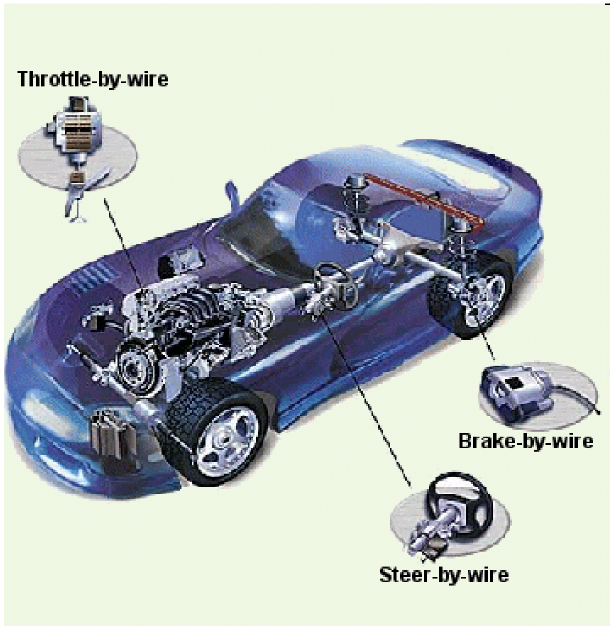
Electronics Design Automation/EDA (chip design)

Nuclear energy

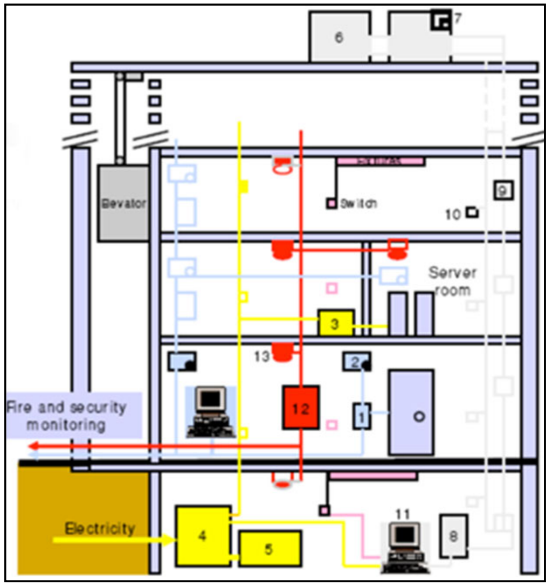


“Smart” infrastructure

More examples of systems

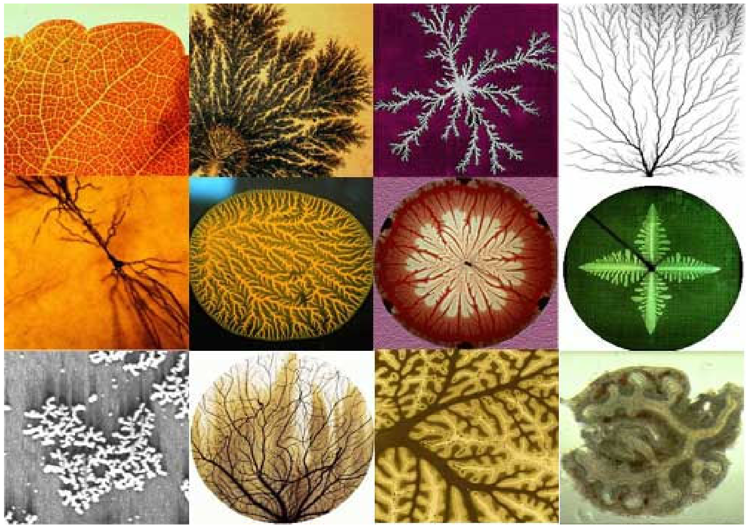


"Smart" car



"Smart" building

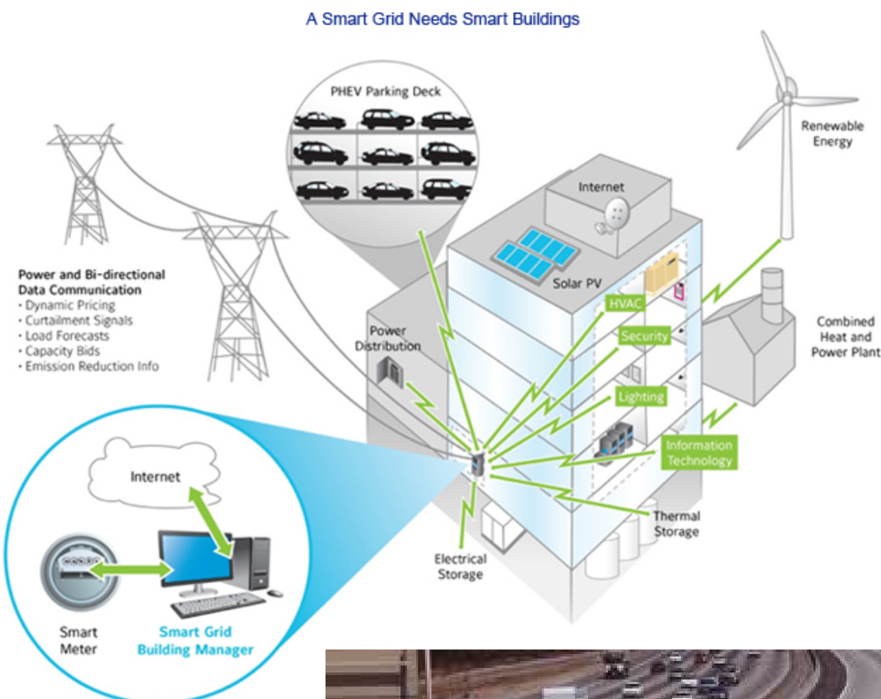

phone

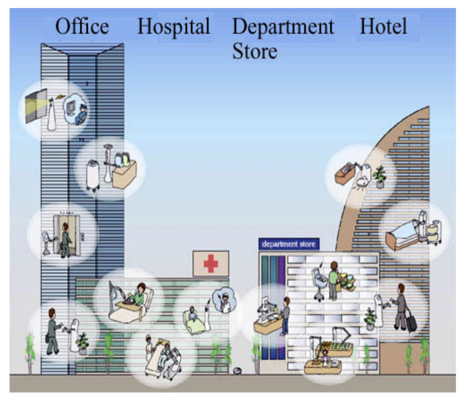

Bio systems, nature

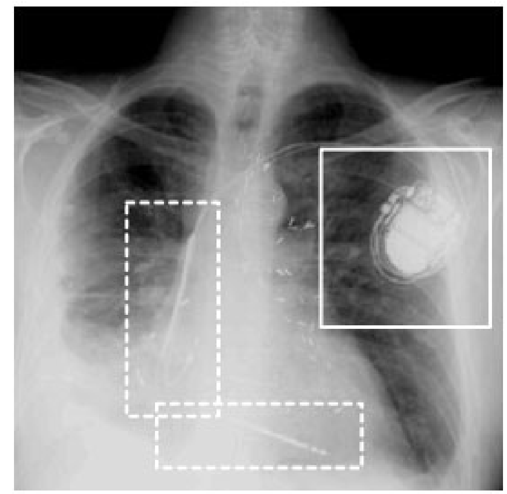

Galaxies, the universe

More examples of systems

- Are the patients systems? The passengers? Cyclists? …







…

Is software a system?

code

OS

What is a “system” ?

System: a first definition

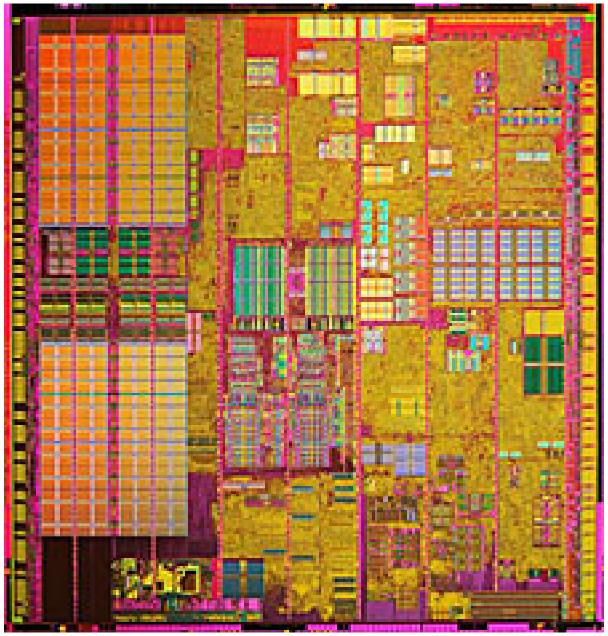
System = something that has

- **State**
- **Dynamics** = how state evolves over **time**

System may also have

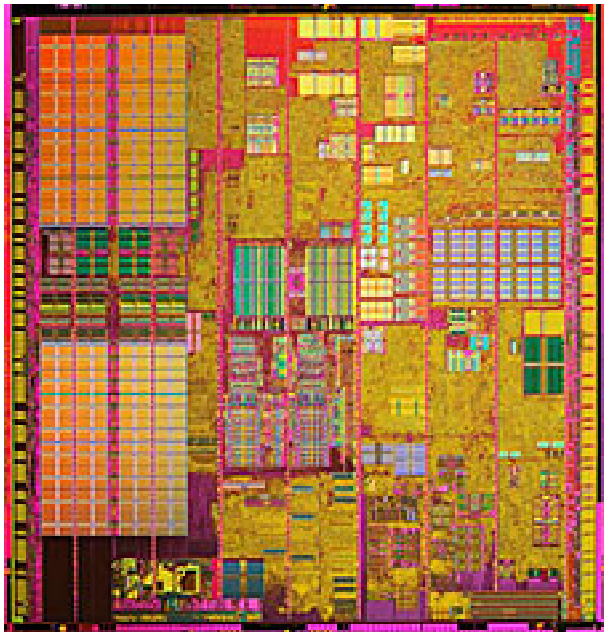
- **Inputs**: influence dynamics
- **Outputs**: observable to the external world (internal state may not be directly observable)

Example: digital circuit



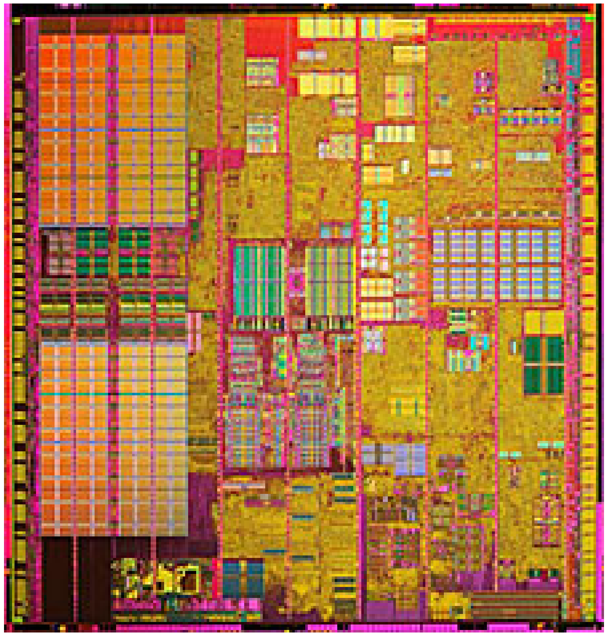
- State = ?
- Dynamics = ?

Example: digital circuit



- State:
  - Value of every register and memory cell
- Dynamics:
  - Defined by the “combinational” part (logical gates, AND, OR, NAND, …)
  - Time: discrete “ticks” of the circuit clock

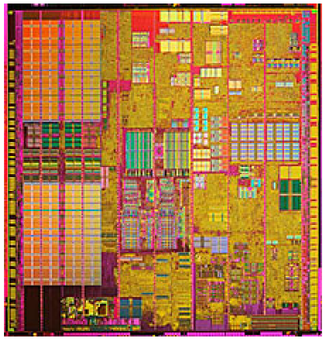
Example: digital circuit



But we could also define these differently:

- State:
  - The currents and voltages of all transistors at a given time t
- Dynamics:
  - Physics of electronic circuits (differential algebraic equations)
  - Time: continuous

Example: digital circuit



Discrete-time system

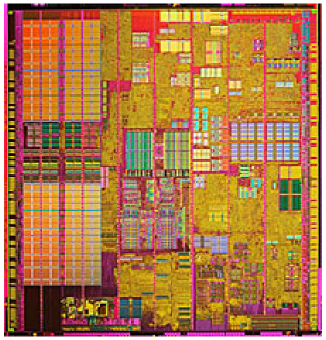
- State:
  - Value of every register and memory cell
- Dynamics:
  - Logical gates
  - Discrete time

Continuous-time system

- State:
  - The currents and voltages of all transistors at a given time t
- Dynamics:
  - Physics of electronic circuits
  - Continuous time

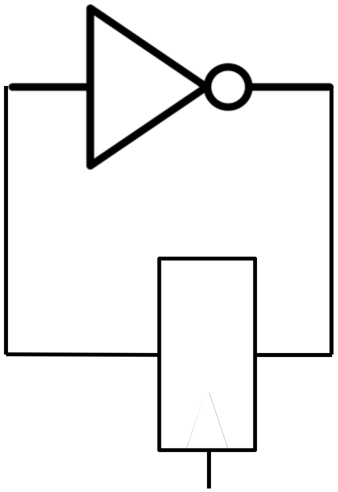
Different levels of **abstraction**

Real systems vs system models

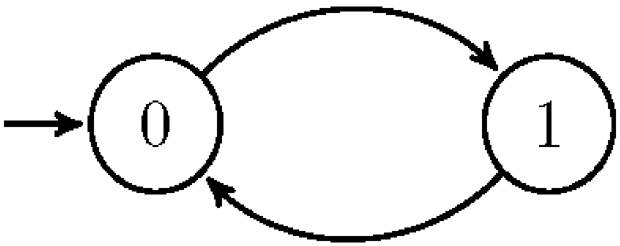


“real” system
(electronic circuit)

Another “real” system:



System model:



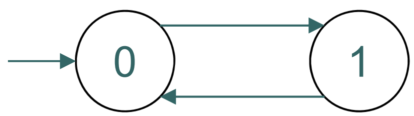
Finite state machine

To reason about systems (analyze, make predictions, prove things, ...), we need mathematical models.

But we often say “system” when we actually mean “system model”.

Many system models

System model 1:



Finite state machine
(FSM) drawn as a graph

System model 2:

```
node Circuit ()
  returns (Output: bool);
let
  Output = false -> not pre Output;
tel
```

The same FSM written as a Lustre program

Different languages/syntaxes for the same underling models/semantics

Multi-paradigm modeling

Different models for the same system (e.g., discrete-time, continuous-time, different levels of abstract)

Different syntaxes for the same model

Different semantics (meaning) for the same syntax

Sometimes need to combine different semantics within the same model (e.g., mix of discrete- and continuous-time)

Analytical models vs computational models

Analytical models: mathematical equations, inequalities, …, e.g., $E = mc^2$

Computational models: programs, or other **executable** models that we will see in this course

e.g., `simulator.c`, `protocol.promela`, `circuit.nuxmv`, …

c.f. Papadimitriou et al's "The computational lens" = computational model transforming all sciences!

SYSTEM DESIGN METHODS

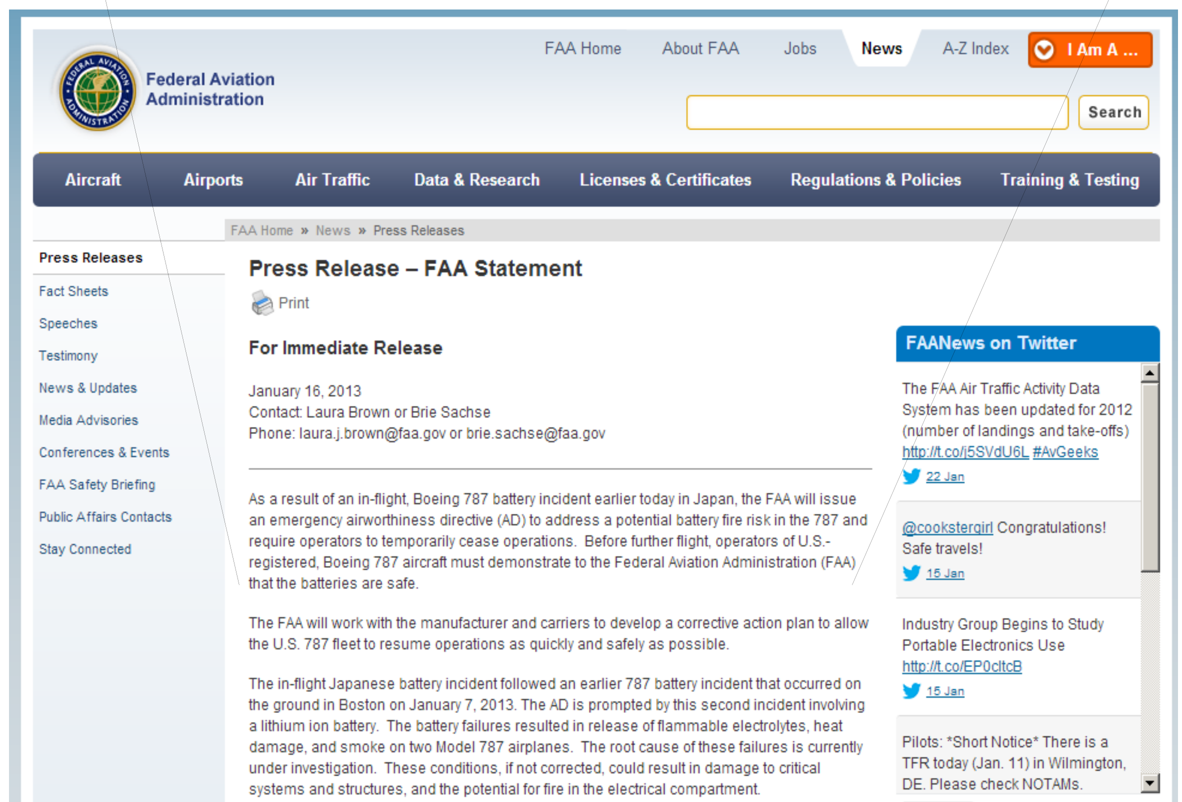
System design by trial-and-error

1. Build prototype
2. Run prototype
3. Find bugs
4. Fix bugs
5. Go to 2 and repeat until …
    1. Bugs become more and more rare to find
    2. Project deadline
    3. …

Design by trial-and-error

- Boeing 787 grounded
- *"All-Nippon today announced it had canceled 320 flights, including 51 international flights, on 787s affecting a total of 46,800 passengers"* [San Jose Mercury News, 1/22/2013]
- FAA restriction finally lifted in April 2013.

As a result of an in-flight, Boeing 787 battery incident earlier today in Japan, the FAA will issue an emergency airworthiness directive (AD) to address a potential battery fire risk in the 787 and require operators to temporarily cease operations.
Before further flight, operators of U.S.-registered, Boeing 787 aircraft must demonstrate to the Federal Aviation Administration (FAA) that the batteries are safe.


FAA Home | About FAA | Jobs | News | A-Z Index | I Am A ...

Federal Aviation Administration

Search

Aircraft | Airports | Air Traffic | Data & Research | Licenses & Certificates | Regulations & Policies | Training & Testing

FAA Home » News » Press Releases

Press Releases
Fact Sheets
Speeches
Testimony
News & Updates
Media Advisories
Conferences & Events
FAA Safety Briefing
Public Affairs Contacts
Stay Connected

Press Release – FAA Statement

Print

For Immediate Release

January 16, 2013
Contact: Laura Brown or Brie Sachse
Phone: laura.j.brown@faa.gov or brie.sachse@faa.gov

As a result of an in-flight, Boeing 787 battery incident earlier today in Japan, the FAA will issue an emergency airworthiness directive (AD) to address a potential battery fire risk in the 787 and require operators to temporarily cease operations. Before further flight, operators of U.S.-registered, Boeing 787 aircraft must demonstrate to the Federal Aviation Administration (FAA) that the batteries are safe.

The FAA will work with the manufacturer and carriers to develop a corrective action plan to allow the U.S. 787 fleet to resume operations as quickly and safely as possible.

The in-flight Japanese battery incident followed an earlier 787 battery incident that occurred on the ground in Boston on January 7, 2013. The AD is prompted by this second incident involving a lithium ion battery. The battery failures resulted in release of flammable electrolytes, heat damage, and smoke on two Model 787 airplanes. The root cause of these failures is currently under investigation. These conditions, if not corrected, could result in damage to critical systems and structures, and the potential for fire in the electrical compartment.

FAANews on Twitter

The FAA Air Traffic Activity Data System has been updated for 2012 (number of landings and take-offs) http://t.co/j5SVdU6L #AvGeeks
22 Jan

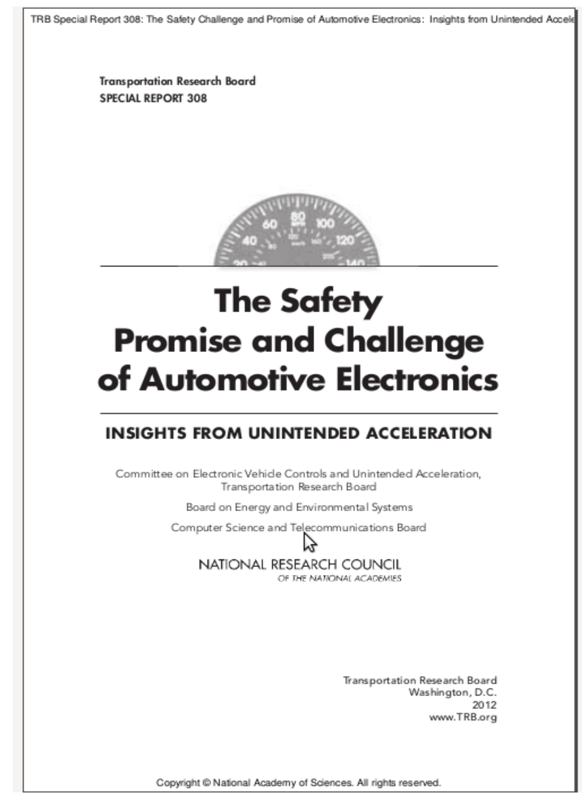
@cookstergirl Congratulations! Safe travels!
15 Jan

Industry Group Begins to Study Portable Electronics Use http://t.co/EP0cltcB
15 Jan

Pilots: *Short Notice* There is a TFR today (Jan. 11) in Wilmington, DE. Please check NOTAMs.

Design by trial-and-error

- Toyota unintended acceleration incidents
- Millions of cars recalled
- Cost: $ billions
- U.S. National Highway Transportation Safety Administration's (NHTSA) report concluded that electronic throttle control systems were not the cause.



TRB Special Report 308: The Safety Challenge and Promise of Automotive Electronics: Insights from Unintended Accele

Transportation Research Board
SPECIAL REPORT 308

The Safety
Promise and Challenge
of Automotive Electronics

INSIGHTS FROM UNINTENDED ACCELERATION

Committee on Electronic Vehicle Controls and Unintended Acceleration,
Transportation Research Board

Board on Energy and Environmental Systems

Computer Science and Telecommunications Board

NATIONAL RESEARCH COUNCIL
OF THE NATIONAL ACADEMIES

Transportation Research Board
Washington, D.C.
2012
www.TRB.org

Copyright © National Academy of Sciences. All rights reserved.

Should we design safety-critical systems by trial and error?

Are the drivers supposed to debug the autopilot?

"It was described as a beta release. The system will learn over time and get better and that's exactly what it's doing. It will start to feel quite refined within a couple of months." – Elon Musk, Tesla CEO, April 2015

Tesla driver dies in first fatal crash while using autopilot mode

June 2016

The autopilot sensors on the Model S failed to distinguish a white tractor-trailer crossing the highway against a bright sky

Tesla autopilot video (source: youtube)

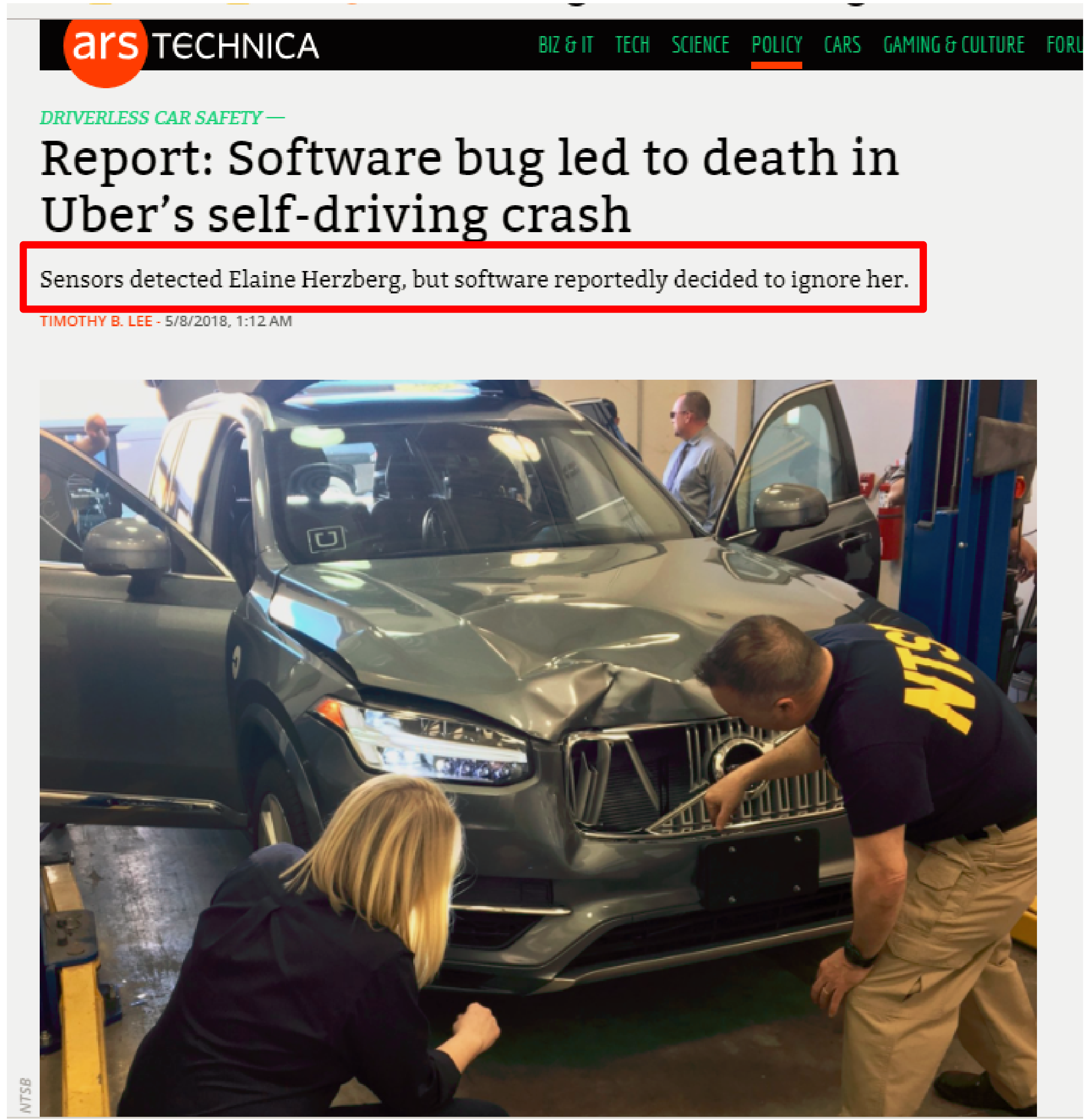
Brave new world

Tempe, Arizona, March 18, 2018

“Software designers face a basic tradeoff […]. If the software is programmed to be too cautious, the ride will be slow and jerky […]. Tuning the software in the opposite direction will produce a smooth ride most of the time—but at the risk that the software will occasionally ignore a real object. […] that's what happened in Tempe in March—and unfortunately the "real object" was a human being.”

"There's a reason Uber would tune its system to be less cautious about objects around the car, […] It is trying to develop a self-driving car that is comfortable to ride in."

specification:
safety, comfort, or both?


ars TECHNICA BIZ & IT TECH SCIENCE POLICY CARS GAMING & CULTURE FORU

DRIVERLESS CAR SAFETY —

Report: Software bug led to death in Uber’s self-driving crash

Sensors detected Elaine Herzberg, but software reportedly decided to ignore her.

TIMOTHY B. LEE - 5/8/2018, 1:12 AM

NTSB

AI: untestable?

RAND CORPORATION

Driving to Safety

How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?

Nidhi Kalra, Susan M. Paddock

Key findings

- Autonomous vehicles would have to be driven hundreds of millions of miles and sometimes hundreds of billions of miles to demonstrate their reliability in terms of fatalities and injuries.
- Under even aggressive testing assumptions, existing fleets would take tens and sometimes hundreds of years

In the United States, roughly 32,000 people are killed and more than two million injured in crashes every year (Bureau of Transportation Statistics, 2015). U.S. motor vehicle crashes as a whole can pose economic and social costs of more than \$800 billion in a single year (Blincoe et al., 2015). And, more than 90 percent of crashes are caused by human errors (National Highway Traffic Safety Administration, 2015)—such as driving too fast and misjudging other drivers' behaviors, as well as alcohol impairment, distraction, and fatigue.

Model-based system design: a more systematic approach

1. Build ~~prototype~~ system model
2. Run ~~prototype~~ Simulate/verify system model
3. Find bugs in the model
4. Fix bugs in the model
5. Go to 2 and repeat until …
    1. Bugs become more and more rare to find
    2. Project deadline
    3. …
6. Synthesize code/prototype automatically from system model

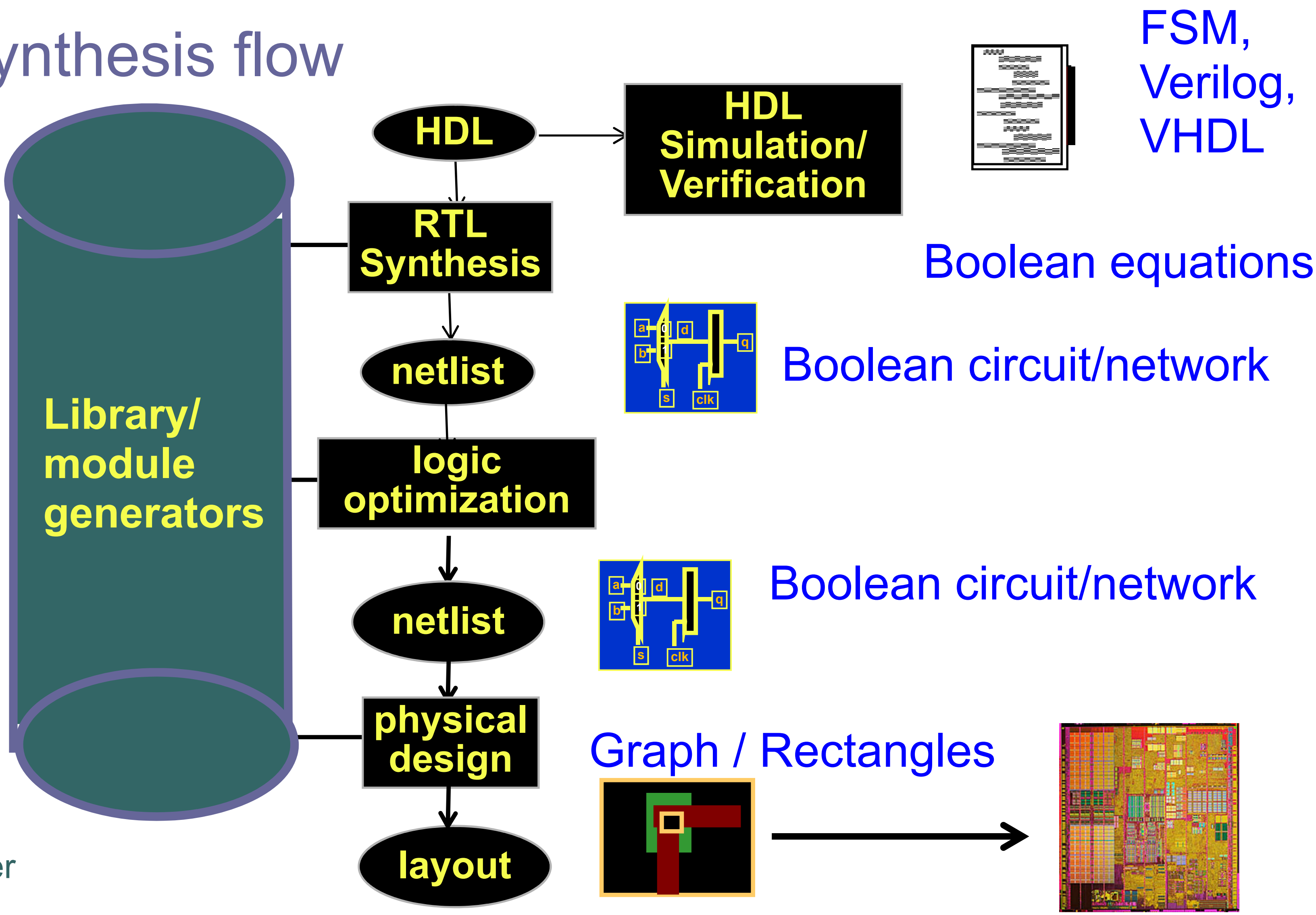
Caveat

In real life, we need both MBD and trial-and-error methods. Why?

1. We cannot trust our models 100%
2. All models are abstractions of reality. They make assumptions that need not hold.
   - E.g., road condition, weather condition, …
3. Verification methods also have their limitations (e.g., scalability problems).
   - As we will see in this course.

Example of a successful model-based design flow

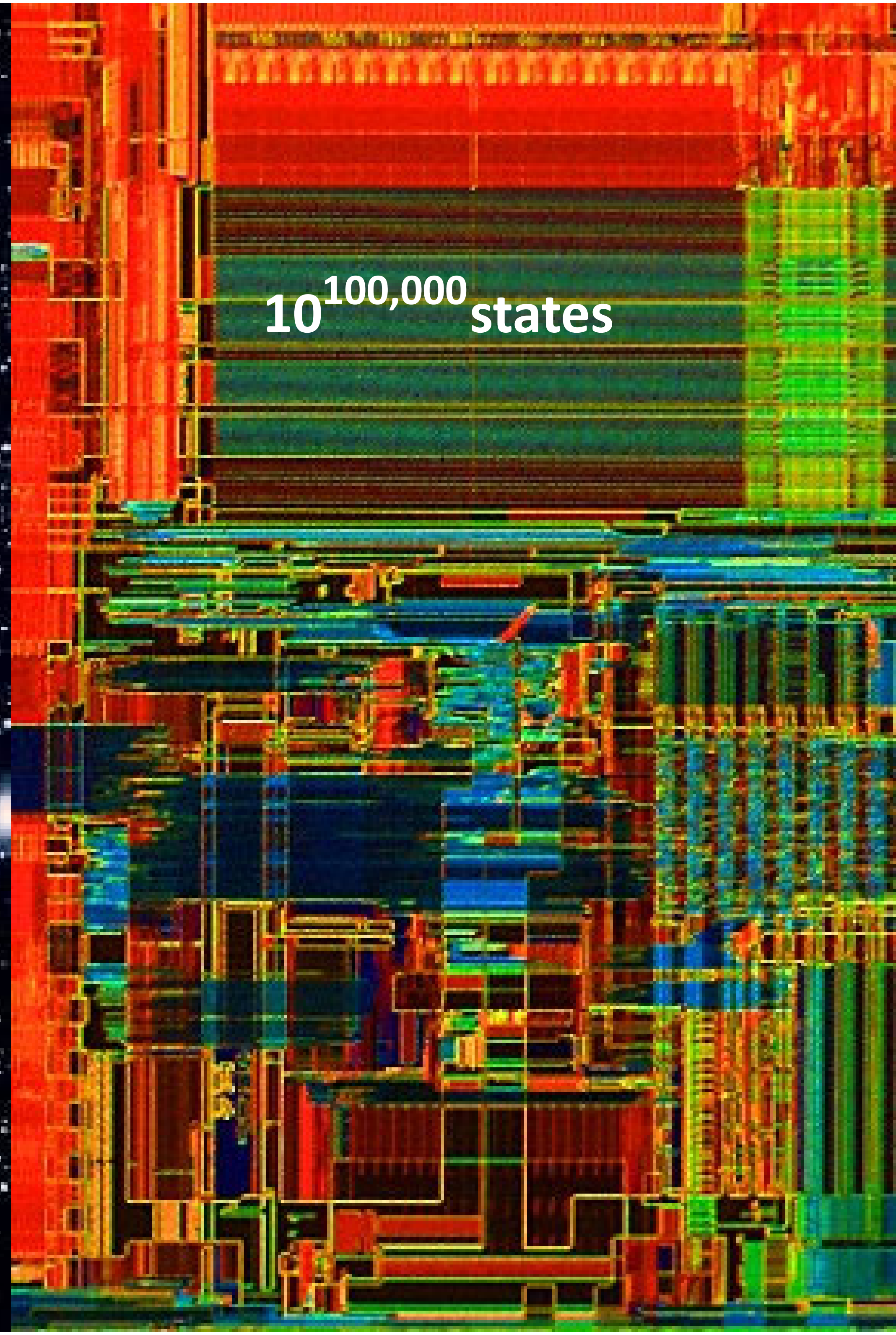
RTL synthesis flow



K. Keutzer

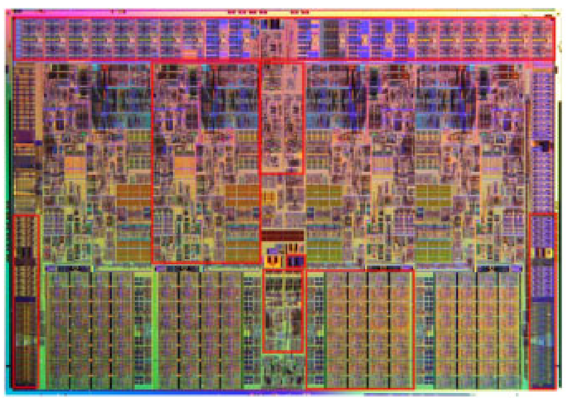
SYSTEM DESIGN COMPLEXITY

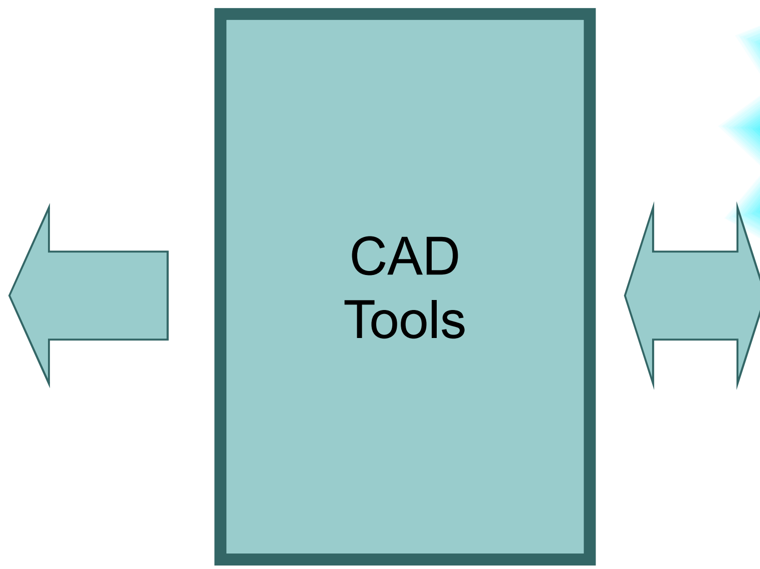

10^{11} stars


$10^{100,000}$ states

Computer-Aided Design (CAD) for ICs / Electronic Design Automation (EDA)

731M transistors





Computer-aided system design

System design is complex => cannot be done “by hand”.

Designers need **tools**!

Not just paper and pencil: computer **automation**.

=> **computer-aided design**

Goal of this course:
Teach you the fundamentals so that you become a good tool user, and also perhaps a tool maker.

Recap

- Everything is a system
  - A medical drug is a system (a program running on an execution platform = the human body)
- System design is mainly software design
  - We can do it by testing / trial-and-error
  - Or we can do it by proving / formal methods / model-based design
  - Usually both
- System correctness is crucial => more formal methods
- Testing is expensive => more formal methods
- Modern system theory => formal methods
  - Profound implications into everything: life, politics, philosophy, how we look at the world (causality, nature vs nurture, …)