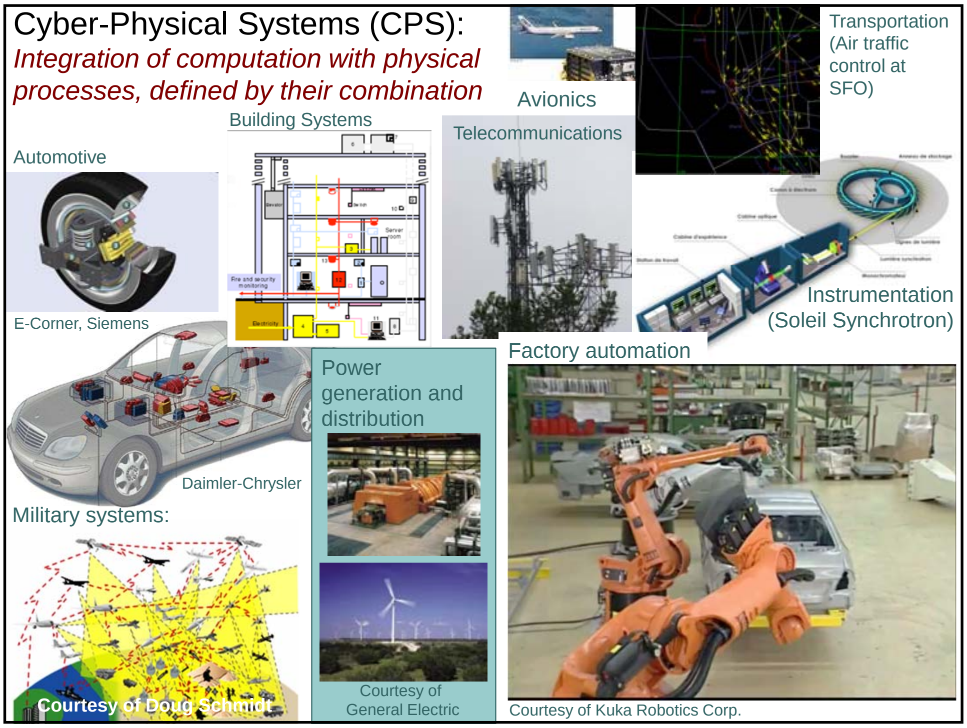
Introduction to Embedded Systems

Sanjit A. Seshia
UC Berkeley
EECS 149/249A
Fall 2015

Lecture 1:
Motivation: Cyber Physical Systems

Embedded Systems
and
Cyber-Physical Systems

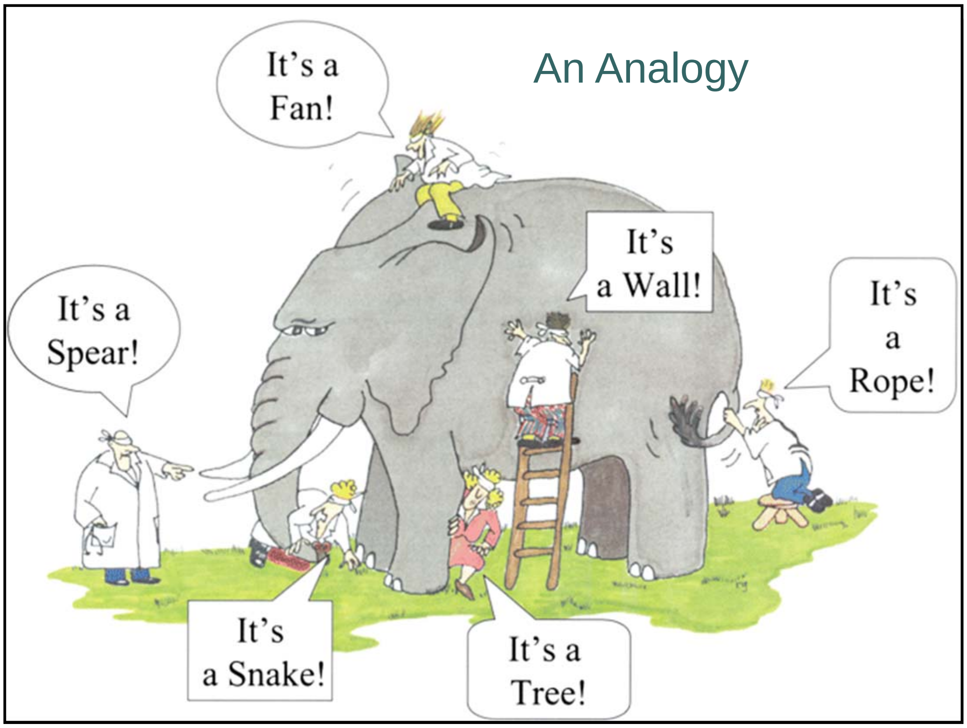
Cyber-Physical Systems (CPS):
Integration of computation with physical processes, defined by their combination

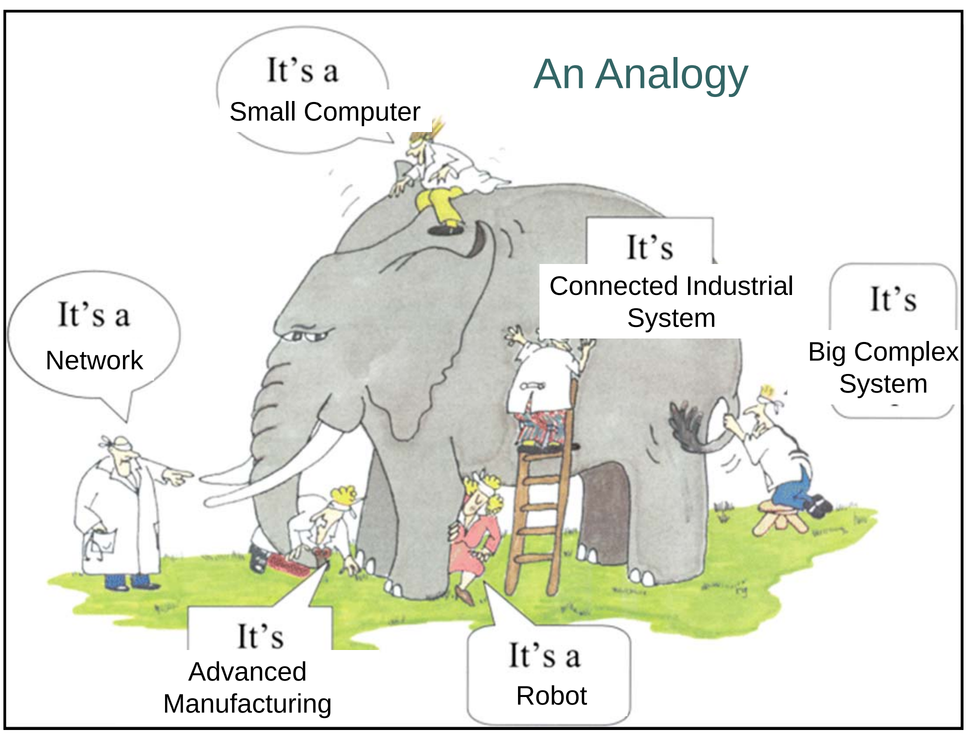


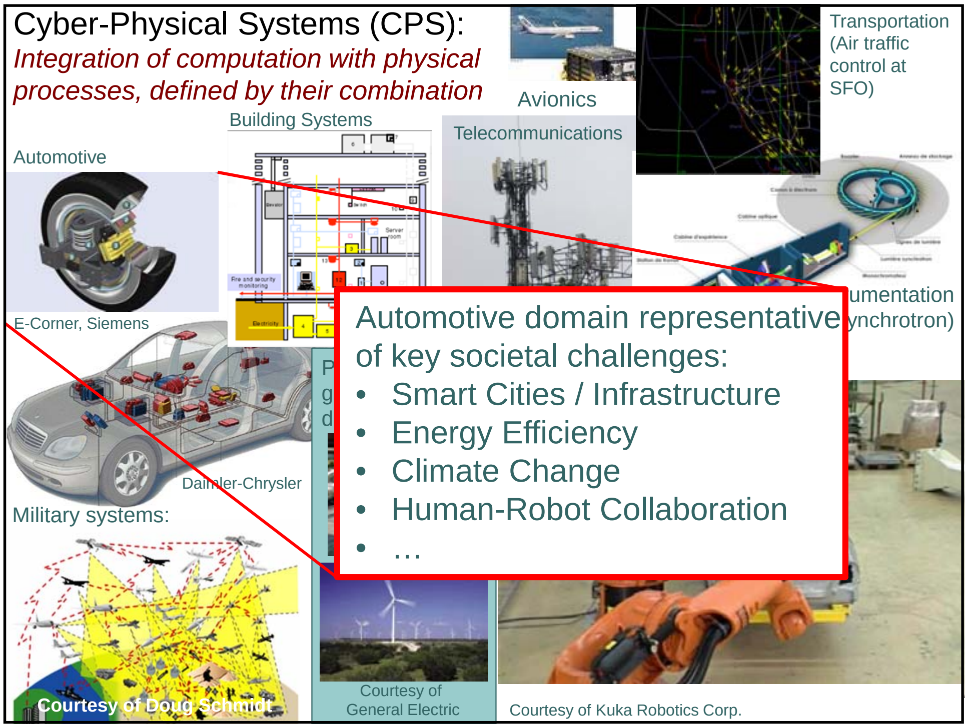
E Pluribus Unum: Out of Many (Terms), One

- Embedded Systems
- Internet of Things (IoT)
- Industrial Internet
- Systems of Systems
- Industrie 4.0
- Internet of Everything (IoE)
- Smart<Everything>

≈ Cyber-Physical Systems


An Analogy
It's a Fan!
It's a Spear!
It's a Wall!
It's a Rope!
It's a Snake!
It's a Tree!


An Analogy
It's a Small Computer
It's a Network
It's Connected Industrial System
It's Big Complex System
It's Advanced Manufacturing
It's a Robot

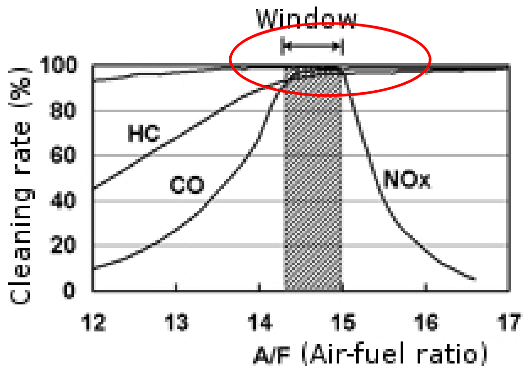


Exercise: Think of a Cyber-Physical System in an Automobile that addresses one of these societal challenges

- Smart Transportation
- Energy Efficiency
- Climate Change
- Human-Robot Collaboration

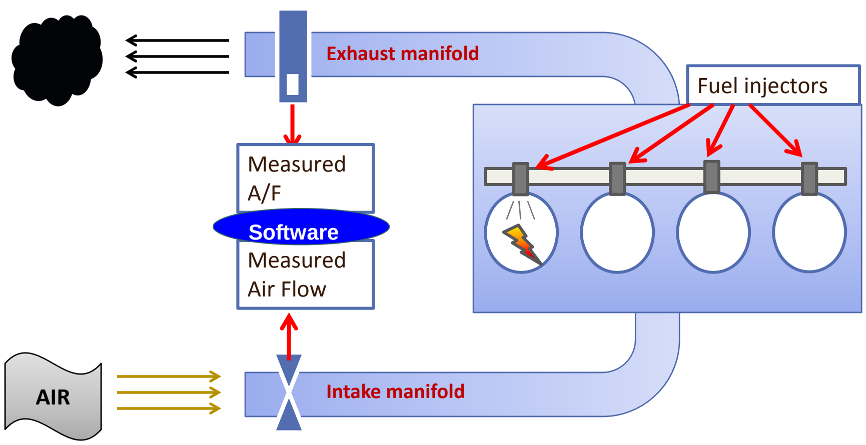
Example: Air-Fuel ratio (A/F) control to reduce emissions

- Catalytic converters reduce CH_4, CO_2, and NO_x emissions
- Conversion efficiency optimal at stoichiometric value



[1] X. Jin. J. Kapinski. J. Deshmukh, K. Ueda, K. Butts, Powertrain Control Verification Benchmark, HSCC 2014

Air-Fuel ratio control: Gasoline Engine setting



A View of the Automotive Industry Trend



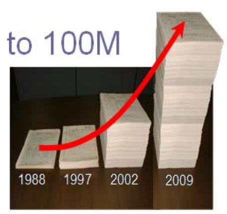



?

[Jyo Deshmukh, Toyota]

Growing Features → Growing Costs

- 70 to 100 ECUs in modern luxury cars, close to 100M LOC
- Engine control: 1.7M LOC
  - F-22 raptor: 1.7M, Boeing 787: 6.5M
- Frost & Sullivan: 200M to 300M LOC
- Electronics & Software: 35-40% of luxury car cost



[from J. Deshmukh]

Charette, R., "This Car Runs on Code", IEEE spectrum,
`http://spectrum.ieee.org/transportation/systems/this-car-runs-on-code`

High Cost of Failures

- Safety-critical: human life at risk
- Recalls, production delays, lawsuits, etc.
- Toyota UA: $1.2B settlement with DoJ in 2014, ongoing lawsuits, …

What this course is about

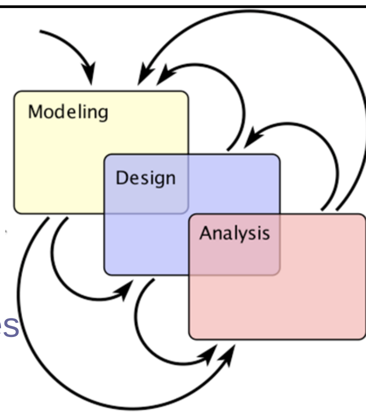
A principled, scientific approach to designing and implementing embedded systems

Not just hacking!!

Hacking can be fun, but it can also be very painful when things go wrong…

Focus on *model-based system design*, and on *embedded software*

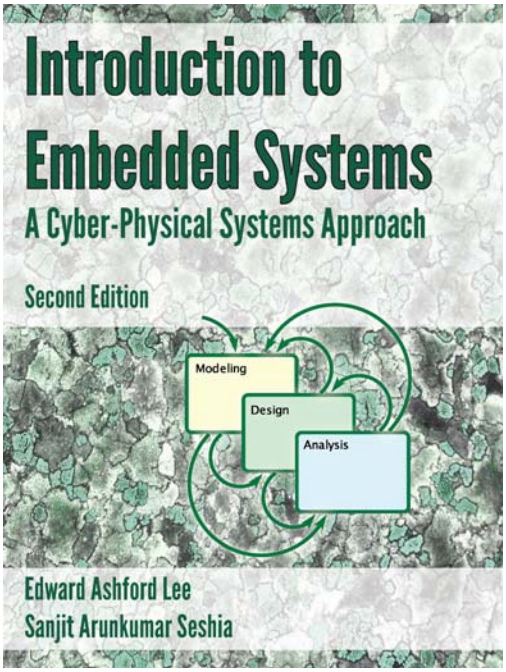
Modeling, Design, Analysis



Modeling is the process of gaining a deeper understanding of a system through imitation. Models express **what** a system does or should do.

Design is the structured creation of artifacts. It specifies **how** a system does what it does.

Analysis is the process of gaining a deeper understanding of a system through dissection. It specifies **why** a system does what it does (or fails to do what a model says it should do).



Your textbook, written for this course, strives to identify and introduce the *durable intellectual ideas* of embedded systems as a technology and as a subject of study. The emphasis is on modeling, design, and analysis of cyber-physical systems, which integrate computing, networking, and physical processes.

http://LeeSeshia.org

Motivating Example of a Cyber-Physical System

(see Chapter 1 in book)

STARMAC quadrotor aircraft (Tomlin, et al.)

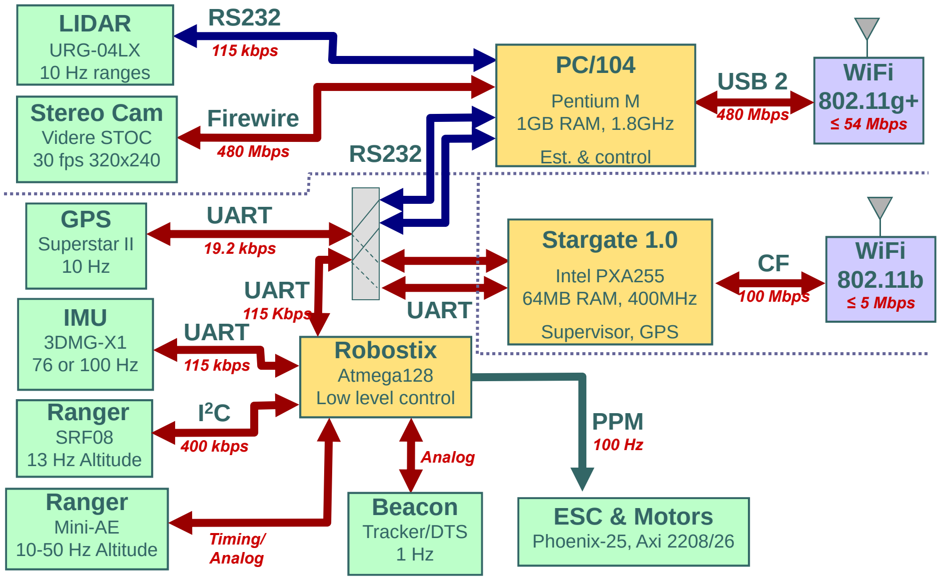
- Introductory Video:
http://www.youtube.com/watch?v=rJ9r2orcaYo
- Back-Flip Manuever:
http://www.youtube.com/watch?v=iD3QgGpzzIM

Modeling:
- Flight dynamics (ch2)
- Modes of operation (ch3)
- Transitions between modes (ch4)
- Composition of behaviors (ch5)
- Multi-vehicle interaction (ch6)

Design:
- Sensors and Actuators (ch7)
- Processors (ch8)
- Memory system (ch9)
- Sensor interfacing (ch10)
- Concurrent software (ch11)
- Real-time scheduling (ch12)

Analysis
- Specifying safe behavior (ch13)
- Achieving safe behavior (ch14)
- Verifying safe behavior (ch15)
- Guaranteeing timeliness (ch16)
- Security and privacy (ch17)

STARMAC Design Block Diagram



EECS 149/249A, UC Berkeley: 17

Outline

The Relevance of CPS

The Future of CPS

EECS 149/249A, UC Berkeley: 18

CPS Relevance:
McKinsey’s Disruptive Technologies

Twelve potentially economically disruptive technologies

Mobile Internet	Increasingly inexpensive and capable mobile computing devices and Internet connectivity
Automation of knowledge work	Intelligent software systems that can perform knowledge work tasks involving unstructured commands and subtle judgments
The Internet of Things	Networks of low-cost sensors and actuators for data collection, monitoring, decision making, and process optimization
Cloud technology	Use of computer hardware and software resources delivered over a network or the Internet, often as a service
Advanced robotics	Increasingly capable robots with enhanced senses, dexterity, and intelligence used to automate tasks or augment humans
Autonomous and near-autonomous vehicles	Vehicles that can navigate and operate with reduced or no human intervention
Next-generation genomics	Fast, low-cost gene sequencing, advanced big data analytics, and synthetic biology (“writing” DNA)
Energy storage	Devices or systems that store energy for later use, including batteries
3D printing	Additive manufacturing techniques to create objects by printing layers of material based on digital models
Advanced materials	Materials designed to have superior characteristics (e.g., strength, weight, conductivity) or functionality
Advanced oil and gas exploration and recovery	Exploration and recovery techniques that make extraction of unconventional oil and gas economical
Renewable energy	Generation of electricity from renewable sources with reduced harmful climate impact

Economic Potential

The Internet of Things	**300%** Increase in connected machine-to-machine devices over past 5 years **80–90%** Price decline in MEMS (microelectromechanical systems) sensors in past 5 years	**1 trillion** Things that could be connected to the Internet across industries such as manufacturing, health care, and mining **100 million** Global machine to machine (M2M) device connections across sectors like transportation, security, health care, and utilities	**$36 trillion** Operating costs of key affected industries (manufacturing, health care, and mining)
Cloud technology	**18 months** Time to double server performance per dollar **3x** Monthly cost of owning a server vs. renting in the cloud	**2 billion** Global users of cloud-based email services like Gmail, Yahoo, and Hotmail **80%** North American institutions hosting or planning to host critical applications on the cloud	**$1.7 trillion** GDP related to the Internet **$3 trillion** Enterprise IT spend
Advanced robotics	**75–85%** Lower price for Baxter[3] than a typical industrial robot **170%** Growth in sales of industrial robots, 2009–11	**320 million** Manufacturing workers, 12% of global workforce **250 million** Annual major surgeries	**$6 trillion** Manufacturing worker employment costs, 19% of global employment costs **$2–3 trillion** Cost of major surgeries
Autonomous and near-autonomous vehicles	**7** Miles driven by top-performing driverless car in 2004 DARPA Grand Challenge along a 150-mile route **1,540** Miles cumulatively driven by cars competing in 2005 Grand Challenge **300,000+** Miles driven by Google’s autonomous cars with only 1 accident (which was human-caused)	**1 billion** Cars and trucks globally **450,000** Civilian, military, and general aviation aircraft in the world	**$4 trillion** Automobile industry revenue **$155 billion** Revenue from sales of civilian, military, and general aviation aircraft

Google Strategy

CNET › Internet › Google closes $3.2 billion purchase of Nest

Google closes $3.2 billion purchase of Nest

The acquisition brings with it the Learning Thermostat and the Protect smoke and CO detector as Google looks to make its mark in the smart home.

by Lance Whitney @lancewhit / February 12, 2014 5:00 AM PST / Updated: February 12, 2014 5:19 AM PST

theguardian | TheObserver

Search

Google's drive into robotics should concern us all

The company's expansion into robotics was developed in tandem with the US military. Where will its power play stop?

John Naughton
The Observer, Sunday 29 December 2013

Google's robotic cars have about $150,000 in equipment including a $70,000 LIDAR (laser radar) system. The range finder mounted on the top is a Velodyne 64-beam laser. This laser allows the vehicle to generate a detailed 3D map of its environment. The car then takes these generated maps and combines them with high-resolution maps of the world, producing different types of data models that allow it to drive itself.

Google and Facebook

Google acquired Titan Aerospace, the drone startup that makes high-flying robots which was previously scoped by Facebook as a potential acquisition target, the WSJ reports.
The deal comes after Facebook disclosed purchase of U.K.-based Ascenta for its globe-spanning Internet plans.
Both Ascenta and Titan Aerospace are in the business of high altitude drones integral to blanketing the globe in cheap, omnipresent Internet connectivity to help bring remote areas online.
That's not all the Titan drones can help Google with, however. ***The company's robots also take high-quality images in real-time that could help with Maps initiatives, as well as contribute to things like "disaster relief" and addressing "deforestation,"….***

Apple iCar?

This week, years after that first sighting, Tesla announced plans for what it calls the "Gigafactory," a 10-million-square-foot plant for making car batteries. …But it's not just the prospect of a gasoline-free future that has sparked such excitement about the Gigafactory. The same basic lithium-ion tech that fuels Tesla's cars also runs most of today's other mobile gadgets, large and small. If Tesla really produces batteries at the scale it's promising, cars could become just one part of what the company does. **One day, Tesla could be a company that powers just about everything, from the phone in your pocket to the electrical grid itself.**

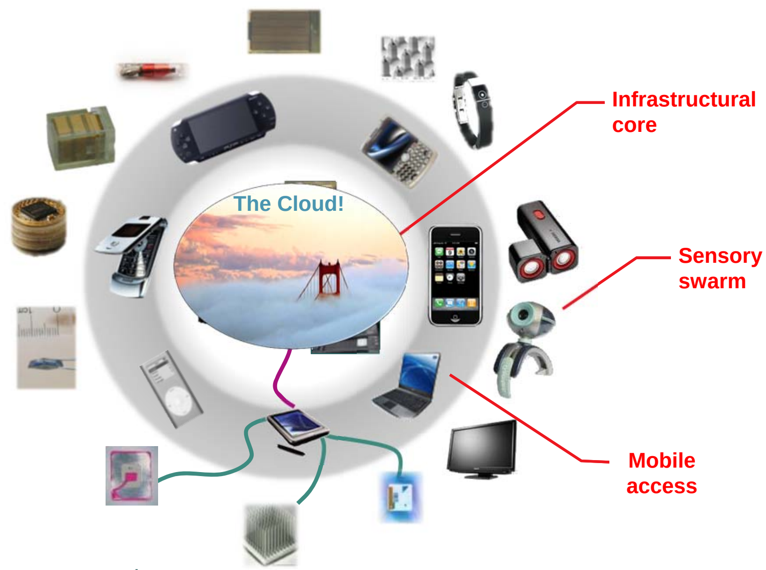
Earlier this month, as rumors swirled that Apple might want to buy Tesla, *San Francisco Chronicle* reported that **Tesla CEO Elon Musk had indeed met with the iPhone maker. Musk later confirmed that Tesla and Apple had talked, but he wouldn't say what about.**

Outline

The Relevance of CPS

The Future of CPS

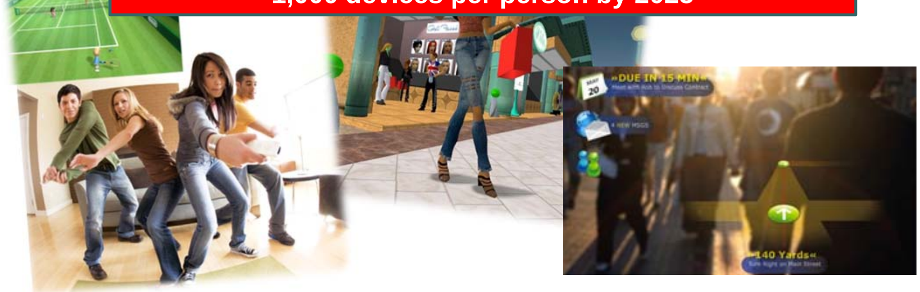
The Emerging IT Scene!



Courtesy: J. Rabaey

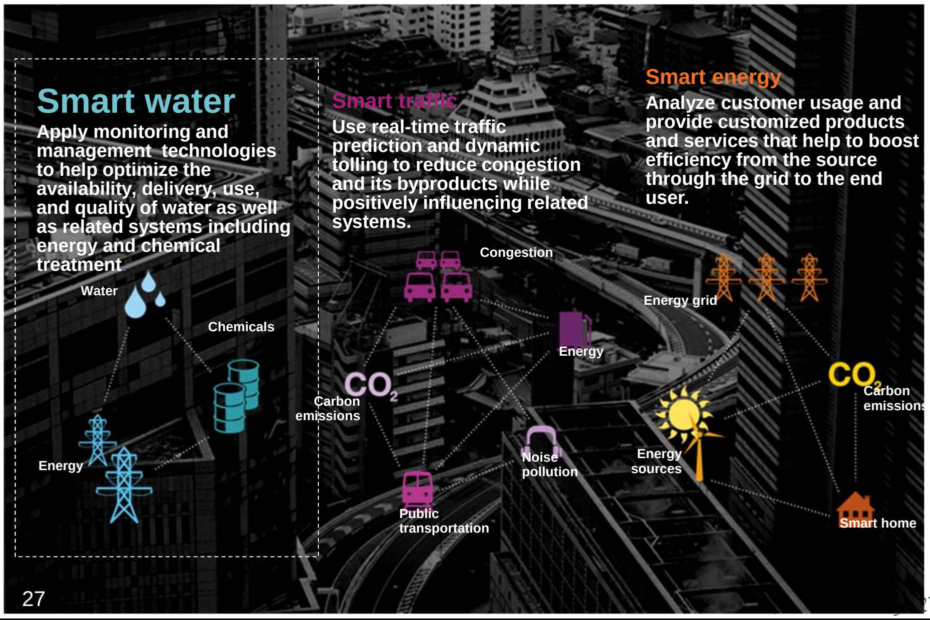
Computers and mobiles to disappear!

Predictions: 7 trillions devices servicing 7 billion people!
1,000 devices per person by 2025



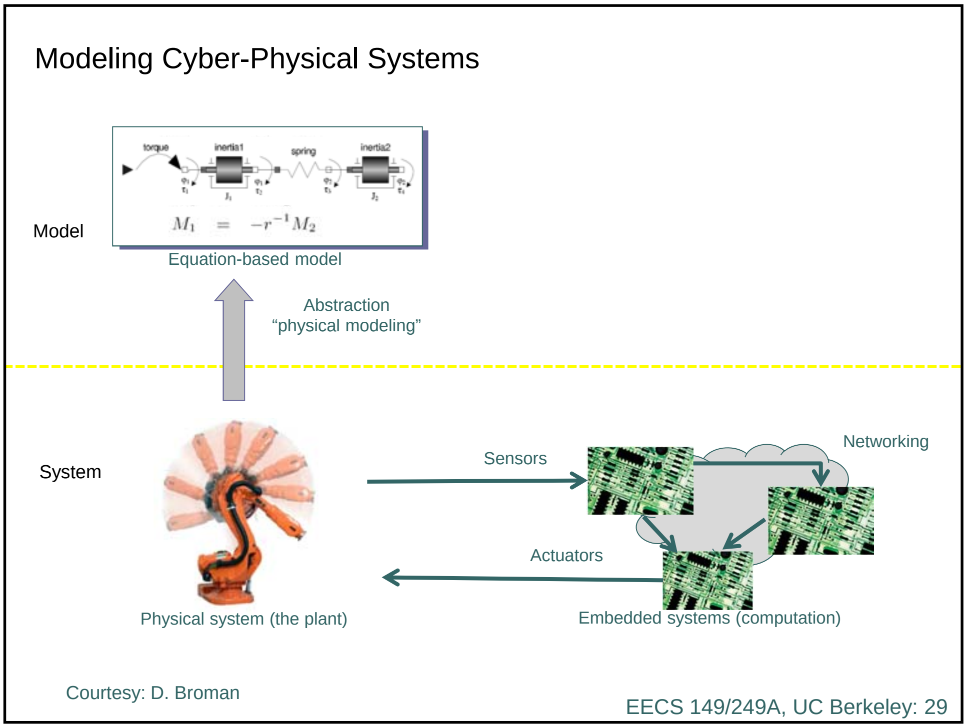
The Immersed Human

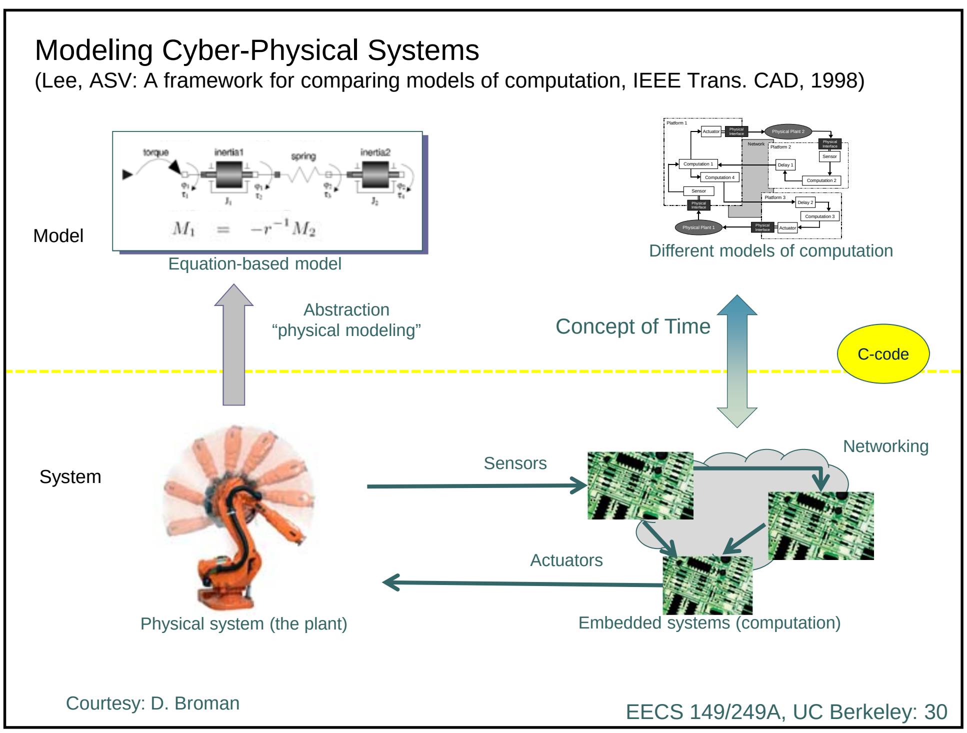
Intelligent systems that gather, synthesize and apply information will change the way entire industries operate.



An Emphasis on

Engineering Models for CPS


Modeling Cyber-Physical Systems
torque
inertia1
spring
inertia2
$M_1 = -r^{-1} M_2$
Model
Equation-based model
Abstraction
"physical modeling"
System
Sensors
Networking
Actuators
Physical system (the plant)
Embedded systems (computation)
Courtesy: D. Broman
EECS 149/249A, UC Berkeley: 29


Modeling Cyber-Physical Systems
(Lee, ASV: A framework for comparing models of computation, IEEE Trans. CAD, 1998)
torque
inertia1
spring
inertia2
$M_1 = -r^{-1} M_2$
Model
Equation-based model
Different models of computation
Abstraction
"physical modeling"
Concept of Time
C-code
System
Sensors
Networking
Actuators
Physical system (the plant)
Embedded systems (computation)
Courtesy: D. Broman
EECS 149/249A, UC Berkeley: 30

Models vs. Reality

Solomon Golomb: Mathematical models – Uses and limitations. Aeronautical Journal 1968

You will never strike oil by drilling through the map!

Solomon Wolf Golomb (1932) mathematician and engineer and a professor of electrical engineering at the University of Southern California. Best known to the general public and fans of mathematical games as the inventor of polyominoes, the inspiration for the computer game Tetris. He has specialized in problems of combinatorial analysis, number theory, coding theory and communications.

But this does not, in any way, diminish the value of a map!

The Kopetz Principle

Prof. Dr. Hermann Kopetz

Many (predictive) properties that we assert about systems (determinism, timeliness, reliability, safety) are in fact not properties of an *implemented* system, but rather properties of a *model* of the system.

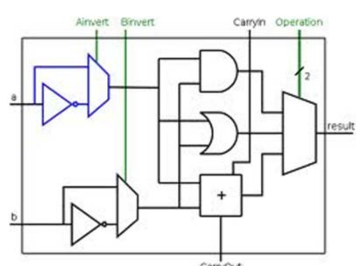
We can make definitive statements about *models*, from which we can *infer* properties of system realizations. The validity of this inference depends on *model fidelity*, which is always approximate.

(paraphrased)

Deterministic Models

Physical System

Model



Synchronous digital logic

Deterministic Models

Physical System

Model

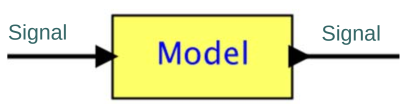
```
/** Reset the output receivers, which are the inside receivers of
 *  the output ports of the container.
 *  @exception IllegalActionException If getting the receivers fails.
 */
private void _resetOutputReceivers() throws IllegalActionException {
    List<IOPort> outputs = ((Actor) getContainer()).outputPortList();
    for (IOPort output : outputs) {
        if (_debugging) {
            _debug("Resetting inside receivers of output port: "
                    + output.getName());
        }
        Receiver[][] receivers = output.getInsideReceivers();
        if (receivers != null) {
            for (int i = 0; i < receivers.length; i++) {
                if (receivers[i] != null) {
                    for (int j = 0; j < receivers[i].length; j++) {
                        if (receivers[i][j] instanceof FSMReceiver) {
                            receivers[i][j].reset();
                        }
                    }
                }
            }
        }
    }
}
```

Single-threaded imperative programs

Deterministic Models

Physical System

Model

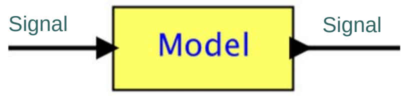


$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Differential Equations

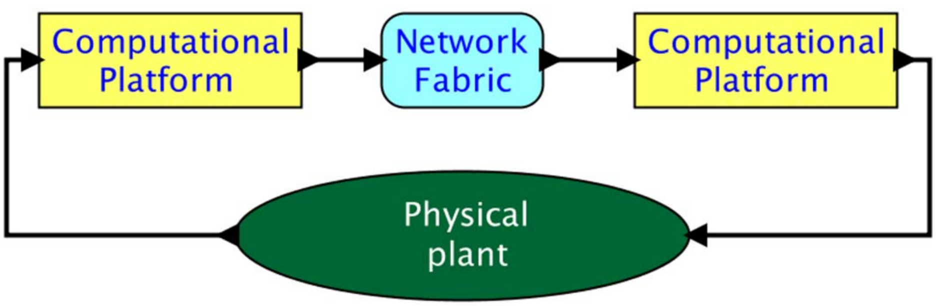
Combinations are Non Deterministic

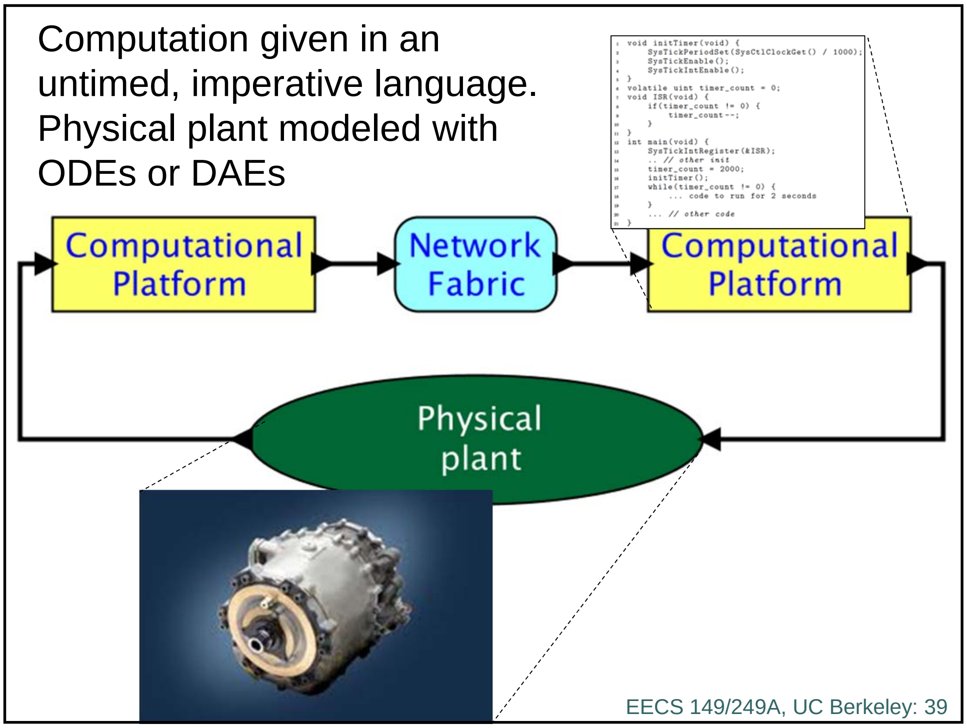

```java
/** Reset the output receivers, which are the inside receivers of
 *  the output ports of the container.
 *  @exception IllegalActionException If getting the receivers fails.
 */
private void _resetOutputReceivers() throws IllegalActionException {
    List<IOPort> outputs = ((Actor) getContainer()).outputPortList();
    for (IOPort output : outputs) {
        if (_debugging) {
            _debug("Resetting inside receivers of output port: "
                    + output.getName());
        }
        Receiver[][] receivers = output.getInsideReceivers();
        if (receivers != null) {
            for (int i = 0; i < receivers.length; i++) {
                if (receivers[i] != null) {
                    for (int j = 0; j < receivers[i].length; j++) {
                        if (receivers[i][j] instanceof FSMReceiver) {
                            receivers[i][j].reset();
                        }
                    }
                }
            }
        }
    }
}
```



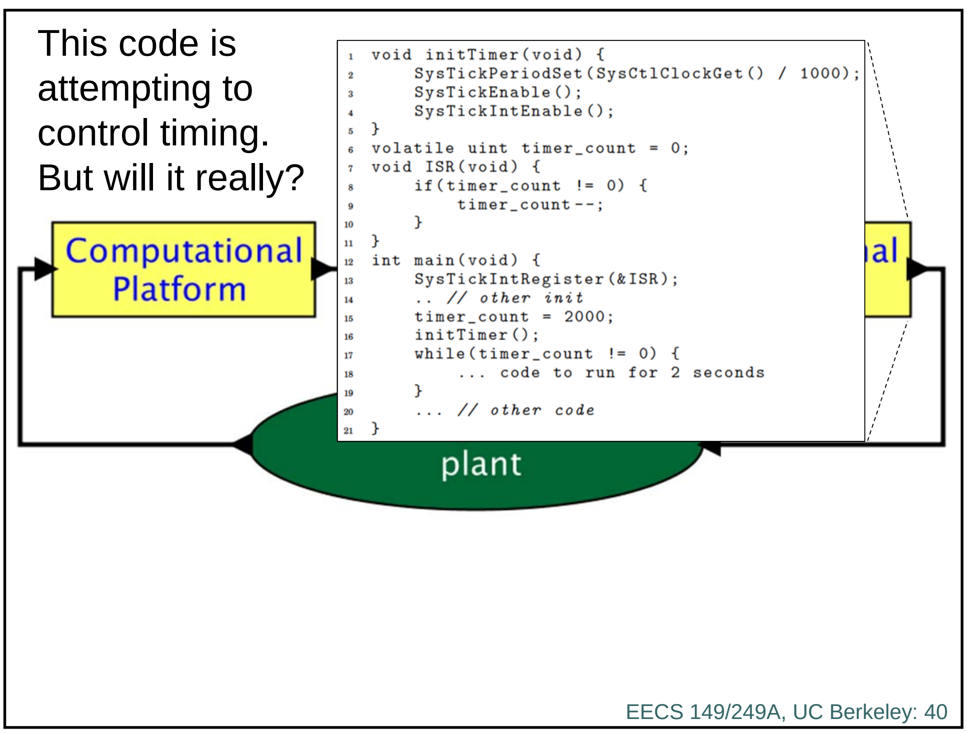
$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Schematic of a simple CPS:



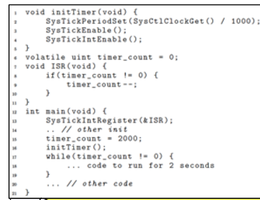


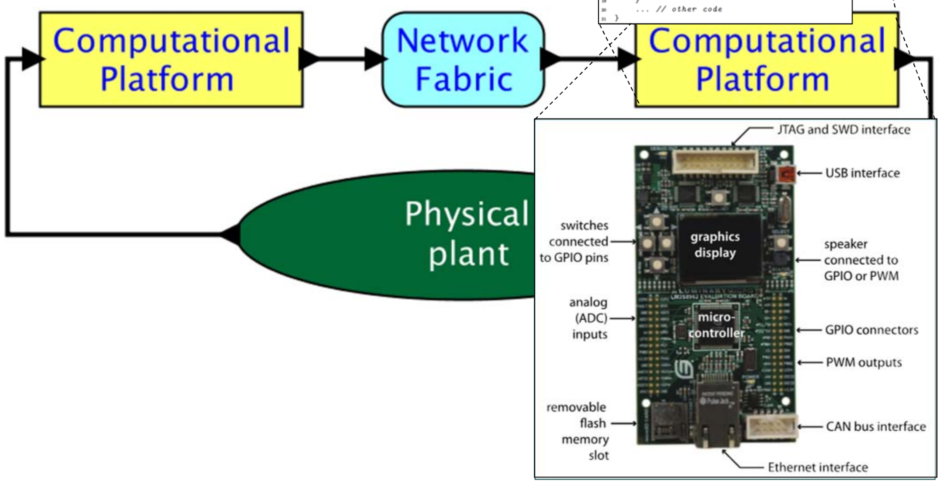
Computation given in an untimed, imperative language. Physical plant modeled with ODEs or DAEs
Computational Platform
Network Fabric
Computational Platform
Physical plant
EECS 149/249A, UC Berkeley: 39



This code is attempting to control timing. But will it really?
Computational Platform
void initTimer(void) {
SysTickPeriodSet(SysCtlClockGet() / 1000);
SysTickEnable();
SysTickIntEnable();
}
volatile uint timer_count = 0;
void ISR(void) {
if(timer_count != 0) {
timer_count--;
}
}
int main(void) {
SysTickIntRegister(&ISR);
.. // other init
timer_count = 2000;
initTimer();
while(timer_count != 0) {
... code to run for 2 seconds
}
... // other code
}
plant
EECS 149/249A, UC Berkeley: 40

Timing behavior emerges from the combination of the program and the hardware platform.





EECS 149/249A, UC Berkeley: 41

A Theme in This Course:
Think Critically

Can we change programming models so that a *correct* execution of a system always delivers the same temporal behavior (up to some precision) at its input/output interfaces?

i.e. we need deterministic CPS models

EECS 149/249A, UC Berkeley: 42

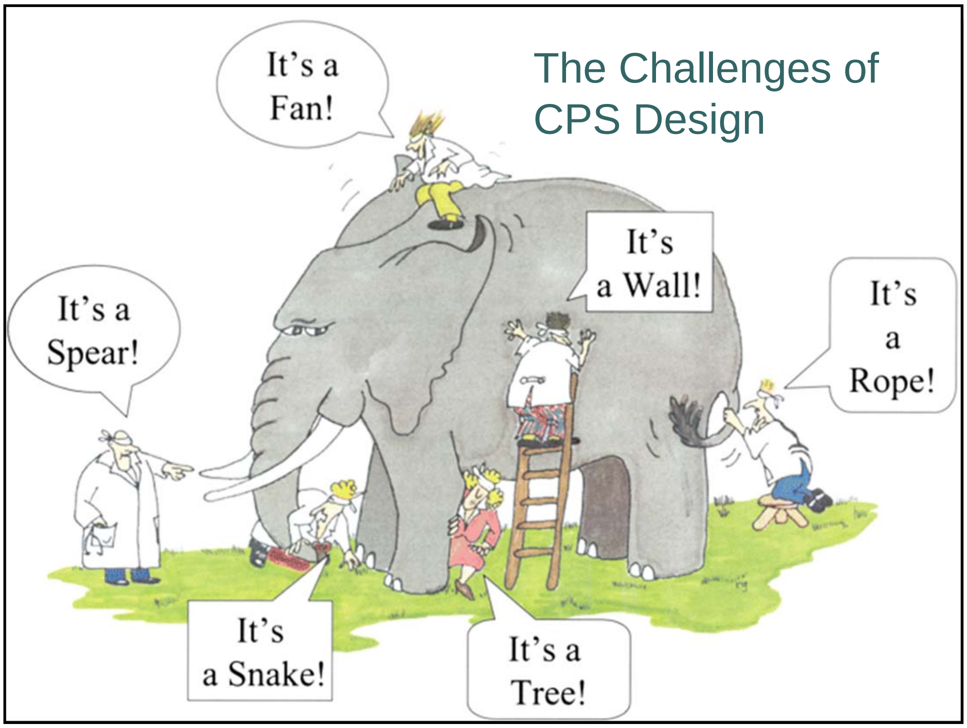
A Theme in This Course: Think Critically

How can we overcome the powerful inertia created by existing languages, tools, and methodologies to allow innovation that may change key abstractions?

- → Heterogeneity
- → Lack of interoperability: “a major obstacle to IoT”

i.e. we need open minds

The Challenges of CPS Design



E.g. Electric Power Systems (EPS) for Aircraft

Physically:
- Generators
- Contactors
- Busses
- Loads

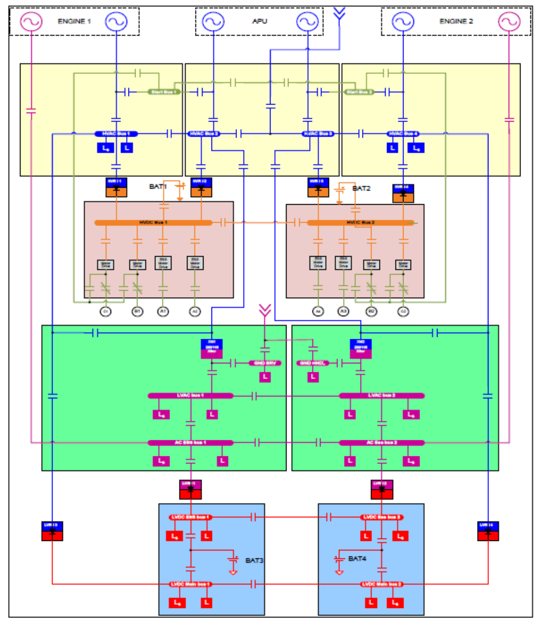
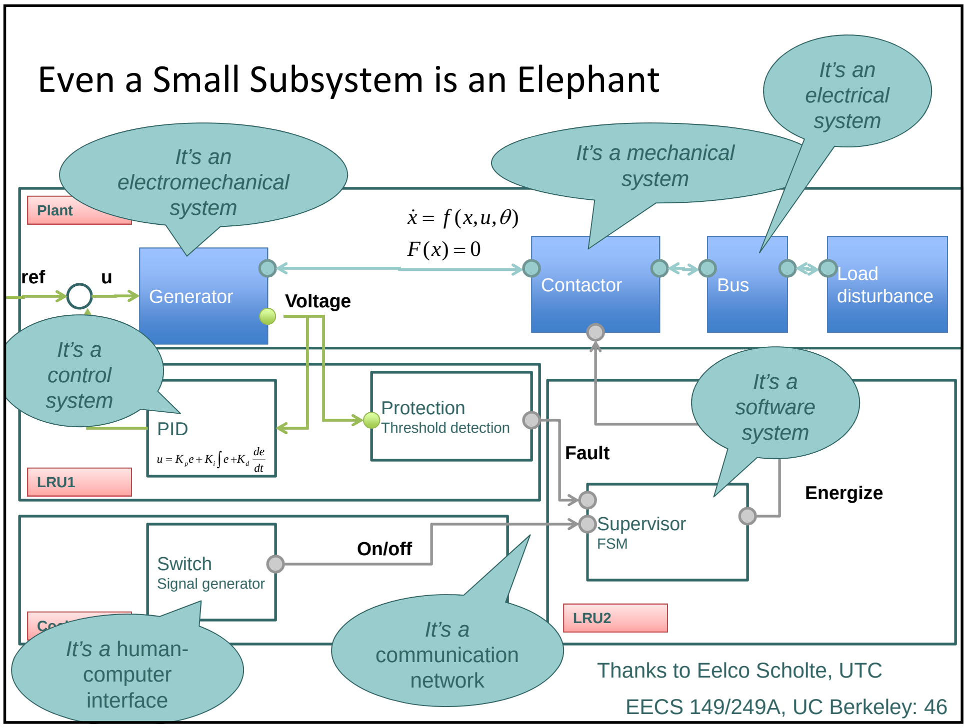


Figure 1: Single line diagram of an electric power system adapted from Honeywell Patent US 7,439,634 B2. Figure courtesy of Rich Poisson, Hamilton-Sundstrand.

Even a Small Subsystem is an Elephant



Thanks to Eelco Scholte, UTC

A Theme in This Course:
Think Critically *and Holistically*!

Any course that purports to teach you how to design embedded systems is misleading you.

The technology will change!

Our goal is to teach you how things are done today, what is done well, and what is not good enough. So you will not be surprised by the changes that *are* coming.