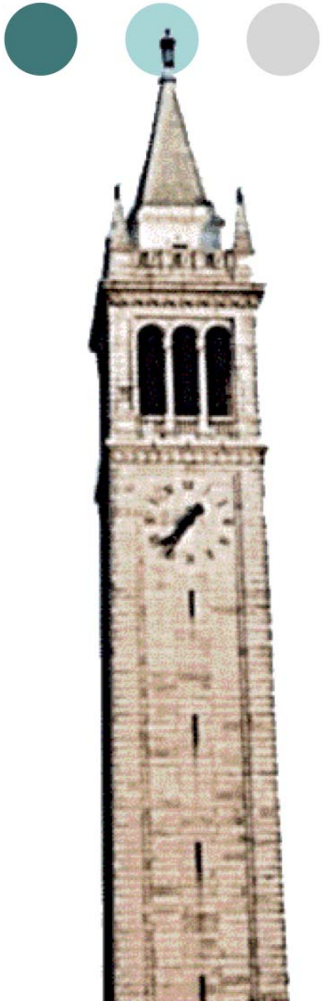


Introducing Embedded Systems: A Cyber- Physical Approach

Edward A. Lee

Robert S. Pepper Distinguished Professor
UC Berkeley

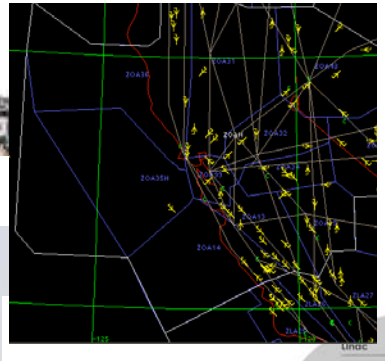
Invited Keynote Talk
Workshop on Embedded Systems Education (WESE)
ES Week

Grenoble, France
October 15, 2009

Cyber-Physical Systems (CPS):

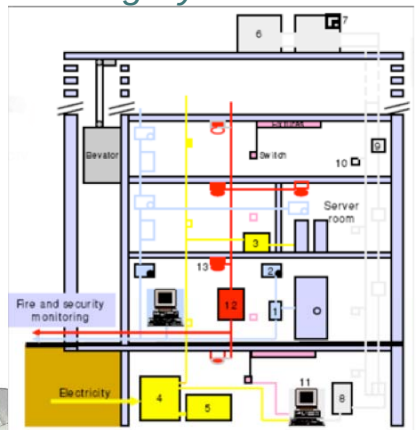
Orchestrating networked computational resources with physical systems

Avionics



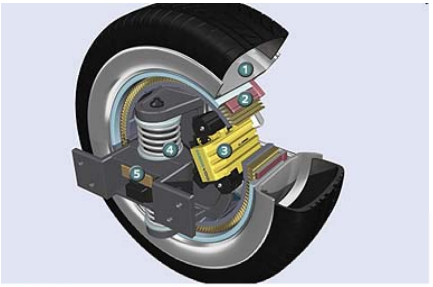
Transportation (Air traffic control at SFO)

Building Systems

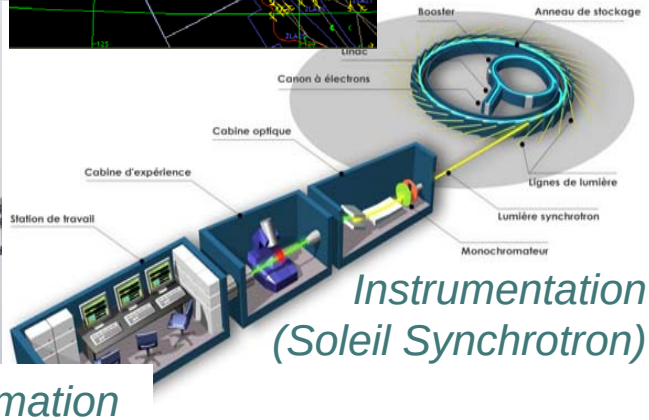


Telecommunications

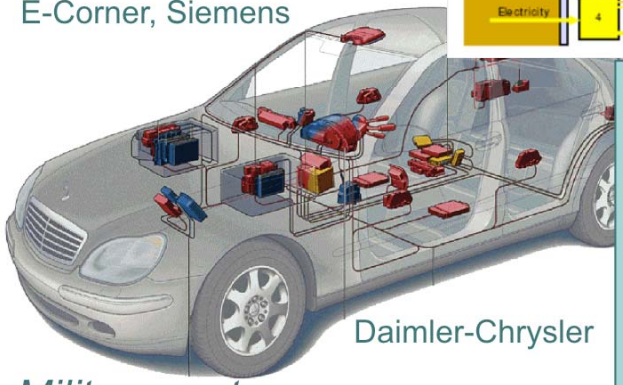
Automotive



E-Corner, Siemens

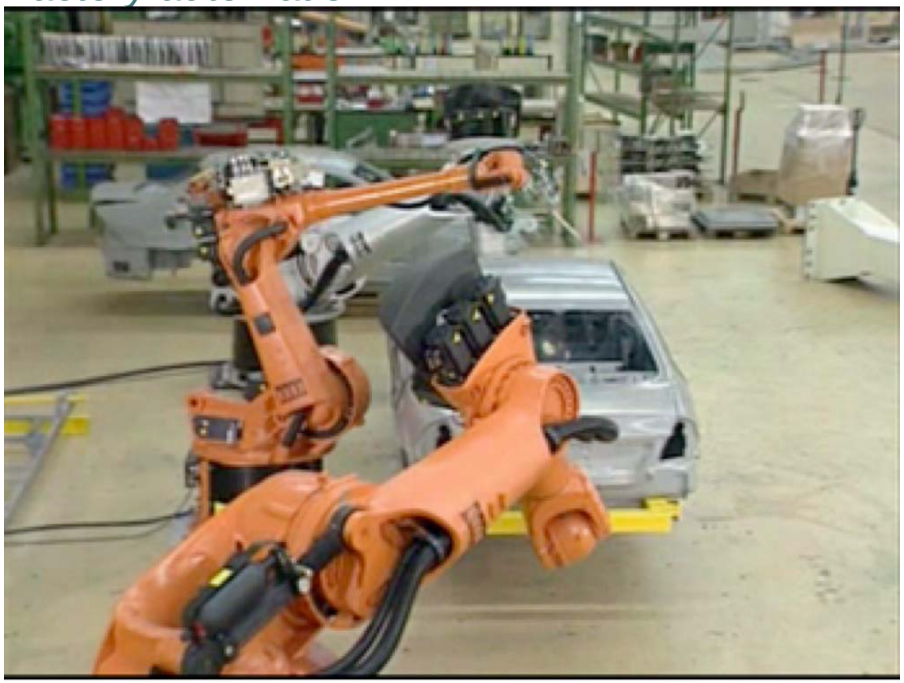


Instrumentation (Soleil Synchrotron)



Daimler-Chrysler

Factory automation

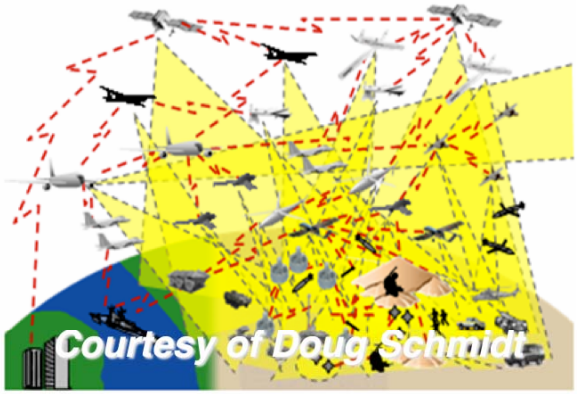


Courtesy of Kuka Robotics Corp.

Power generation and distribution

Courtesy of General Electric

Military systems:



Where CPS Differs from the traditional embedded systems problem:

- *The traditional embedded systems problem:*

  Embedded software is software on small computers. The technical problem is one of optimization (coping with limited resources and extracting performance).

- *The CPS problem:*

  Computation and networking integrated with physical processes. The technical problem is managing dynamics, time, and concurrency in networked computational + physical systems.

Content of an Embedded Systems Course

Traditional focus

- Hardware interfacing
- Interrupts
- Memory systems
- C programming
- Assembly language
- FPGA design
- RTOS design
- …

CPS focus

- Modeling
- Timing
- Dynamics
- Imperative logic
- Concurrency
- Verification
- …

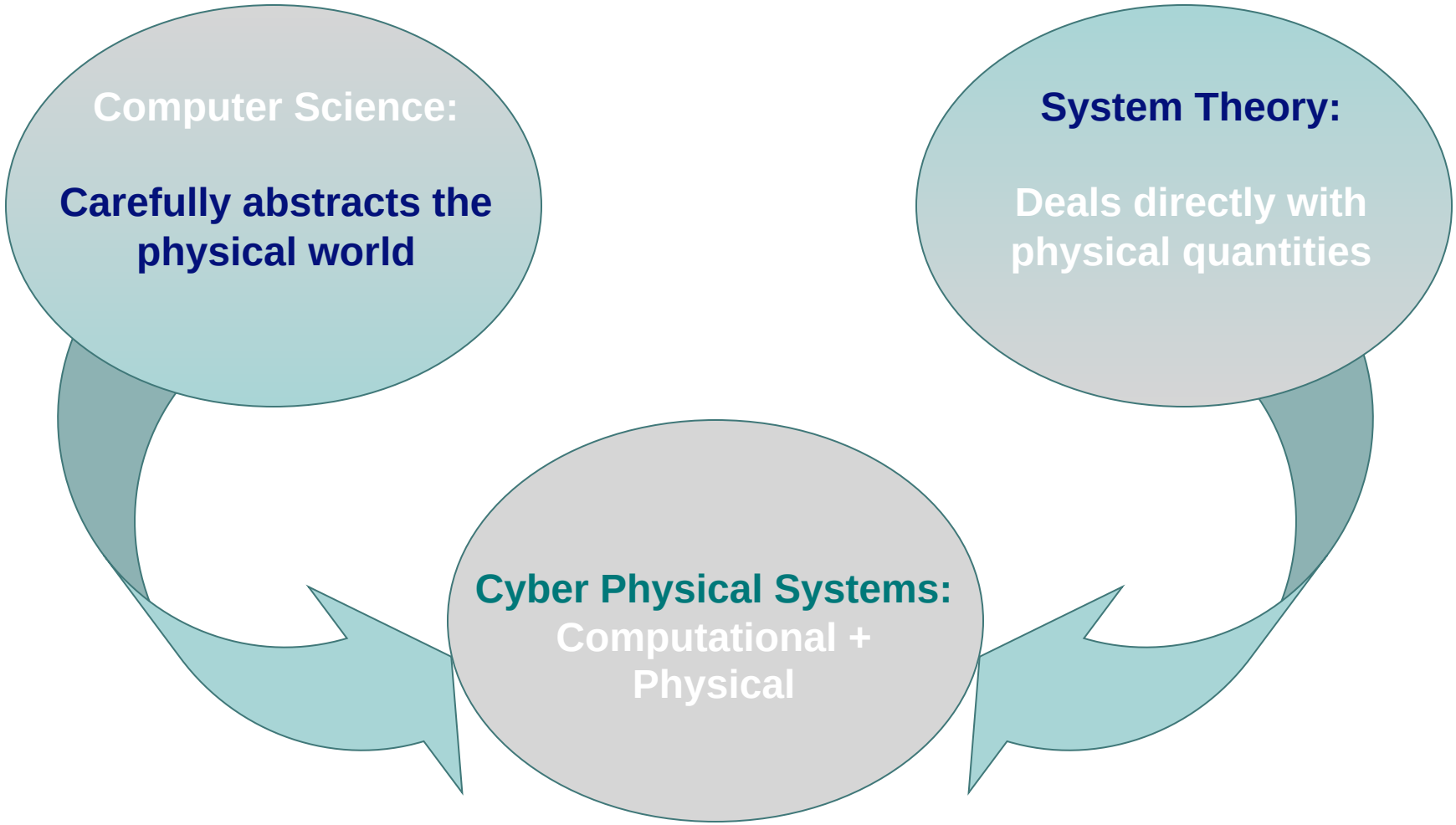
A Theme in our CPS Courses: Model-based design

Models are abstractions of systems:

- structural (OO design)
- ontological (type systems)
- imperative logic (“procedural epistemology”)
- functional logic
- actor-oriented (including dataflow models)

All of these have their place…

CPS is Multidisciplinary



The Educational Challenge (1)

Models for the physical world and for computation diverge.

- physical: time continuum, ODEs, dynamics
- computational: a “procedural epistemology,” logic

There is a huge cultural gap.

Physical system models must be viewed as semantic frameworks, and theories of computation must be viewed as alternative ways of talking about dynamics.

An Advantage:
A Unified EECS Department

The Educational Challenge (2)

Students are taught to *use* modeling techniques, not to *evaluate* modeling techniques.

- “this is how computers work”
- “this equation describes that feedback circuit”

rather than

- “this is how VonNeumann proposed that we control automatic machines”
- “ignoring the intrinsic randomness and latency in this circuit, Black proposed that we could idealize its behavior in this way”

Students must be taught meta-modeling, not just modeling. They must learn to think critically about modeling methods, not just about models.

The Graduate Curriculum

We have developed courses in:

- Hybrid Systems
- Embedded Software and Systems Design
- Algorithmic CAD
- Model Integrated Computing

A key challenge is to define the durable intellectual heart of hybrid and embedded systems, not on ad hoc engineering practice.

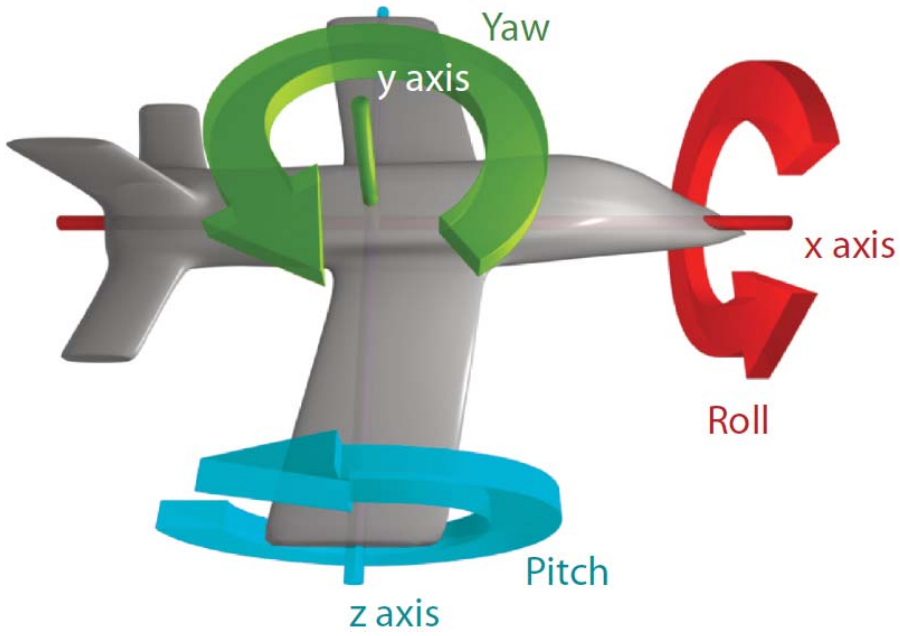
New Undergraduate Course (Spring 2008) "Introduction to Embedded Systems"

This course is intended to introduce students to the design and analysis of computational systems that interact with physical processes. Applications of such systems include medical devices and systems, consumer electronics, toys and games, assisted living, traffic control and safety, automotive systems, process control, energy management and conservation, environmental control, aircraft control systems, communications systems, instrumentation, critical infrastructure control (electric power, water resources, and communications systems for example), robotics and distributed robotics (telepresence, telemedicine), defense systems, manufacturing, and smart structures.

A major *theme of this course will be on the interplay of practical design with formal models of systems*, including both software components and physical dynamics. A major emphasis will be on building high confidence systems with real-time and concurrent behaviors.

- *Cyber-Physical Systems*
- *Model-Based Design*
- *Sensors and Actuators*
- *Interfacing to Sensors and Actuators*
- *Actors, Dataflow*
- *Modeling Modal Behavior*
- *Concurrency: Threads and Interrupts*
- *Hybrid Systems*
- *Simulation*
- *Specification; Temporal Logic*
- *Reachability Analysis*
- *Controller Synthesis*
- *Control Design for FSMs and ODEs*
- *Real-Time Operating Systems (RTOS)*
- *Scheduling: Rate-Monotonic and EDF*
- *Concurrency Models*
- *Execution Time Analysis*
- *Localization and Mapping*
- *Real-Time Networking*
- *Distributed Embedded Systems*

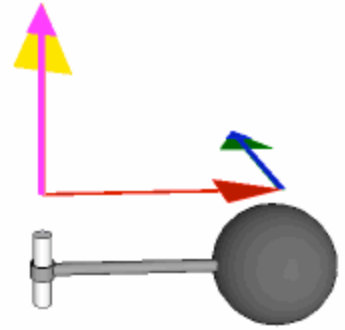
Modeling Physical Dynamics



- Orientation: $\theta : \mathbb{R} \rightarrow \mathbb{R}^3$
- Angular velocity: $\dot{\theta} : \mathbb{R} \rightarrow \mathbb{R}^3$
- Angular acceleration: $\ddot{\theta} : \mathbb{R} \rightarrow \mathbb{R}^3$
- Torque: $\mathbf{T} : \mathbb{R} \rightarrow \mathbb{R}^3$

$$\theta(t) = \begin{bmatrix} \dot{\theta}_x(t) \\ \dot{\theta}_y(t) \\ \dot{\theta}_z(t) \end{bmatrix} = \begin{bmatrix} \text{roll} \\ \text{yaw} \\ \text{pitch} \end{bmatrix}$$

Feedback Control Problem

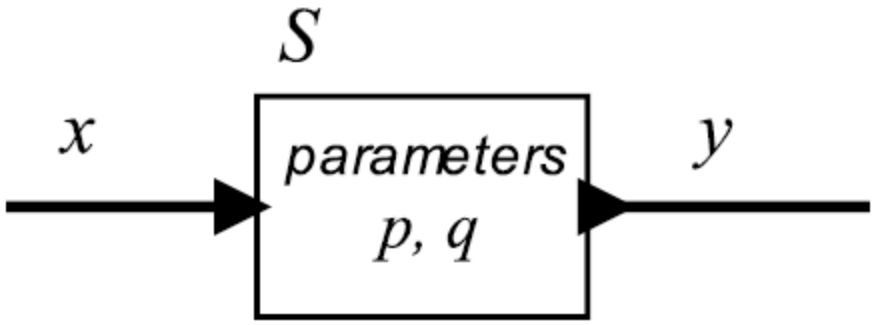


A helicopter without a tail rotor, like the one below, will spin uncontrollably due to the torque induced by friction in the rotor shaft.

Control system problem:
Apply torque using the tail rotor to counterbalance the torque of the top rotor.

Actor Model of Systems

A *system* is a function that accepts an input *signal* and yields an output signal.



The domain and range of the system function are sets of signals, which themselves are functions.

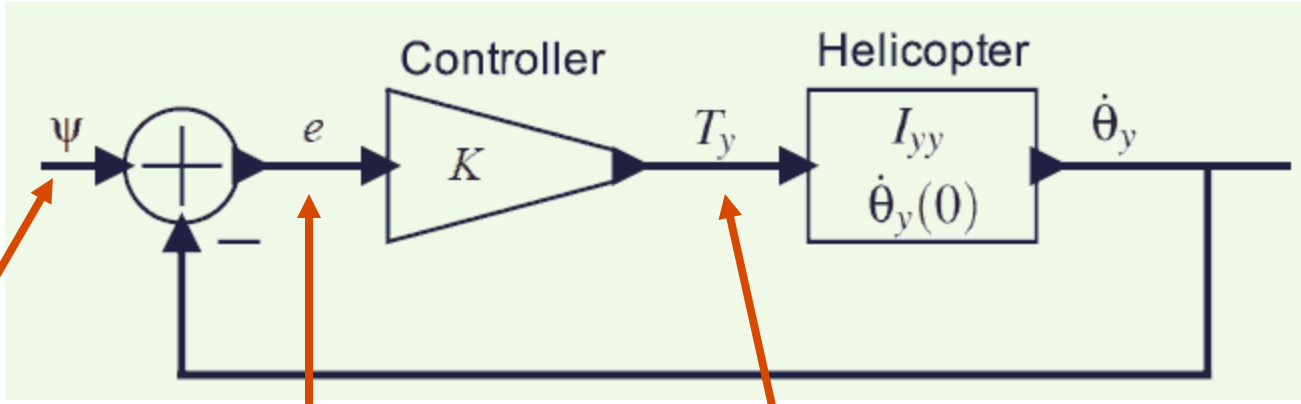
$$x: \mathbb{R} \rightarrow \mathbb{R}, \quad y: \mathbb{R} \rightarrow \mathbb{R}$$

$$S: X \rightarrow Y$$

$$X = Y = (\mathbb{R} \rightarrow \mathbb{R})$$

Parameters may affect the definition of the function *S*.

Proportional controller



desired angular velocity

error signal

net torque

$$e(t) = \psi(t) - \dot{\theta}_y(t)$$

$$T_y(t) = Ke(t)$$

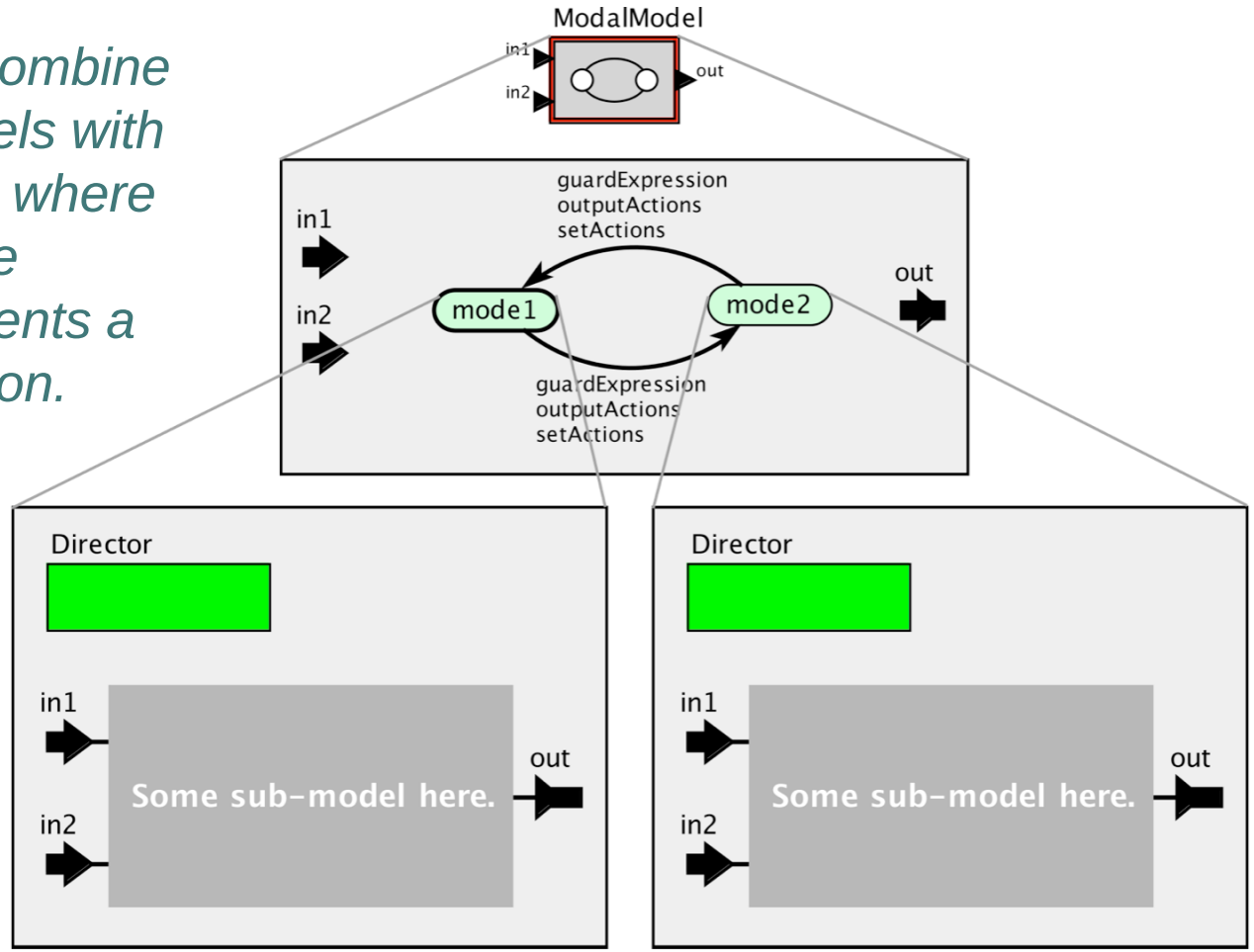
$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau) d\tau$$

$$= \dot{\theta}_y(0) + \frac{K}{I_{yy}} \int_0^t (\psi(\tau) - \dot{\theta}_y(\tau)) d\tau$$

Note that the angular velocity appears on both sides, so this equation is not trivial to solve.

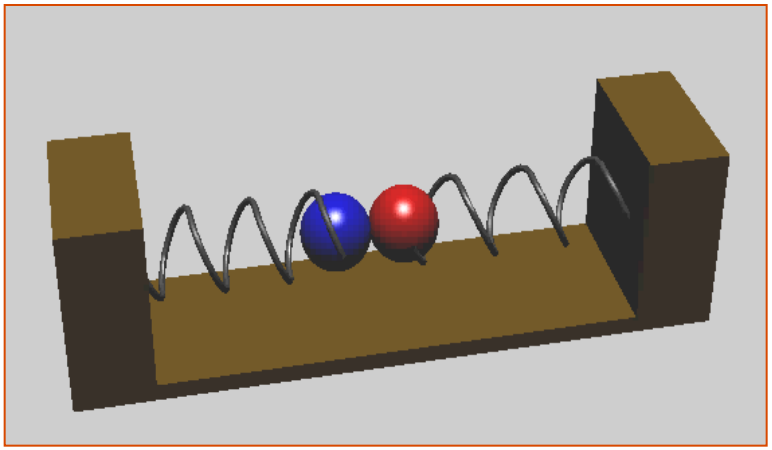
State Machines and Modal Models

Modal models combine such actor models with state machines, where each state of the machine represents a mode of operation.



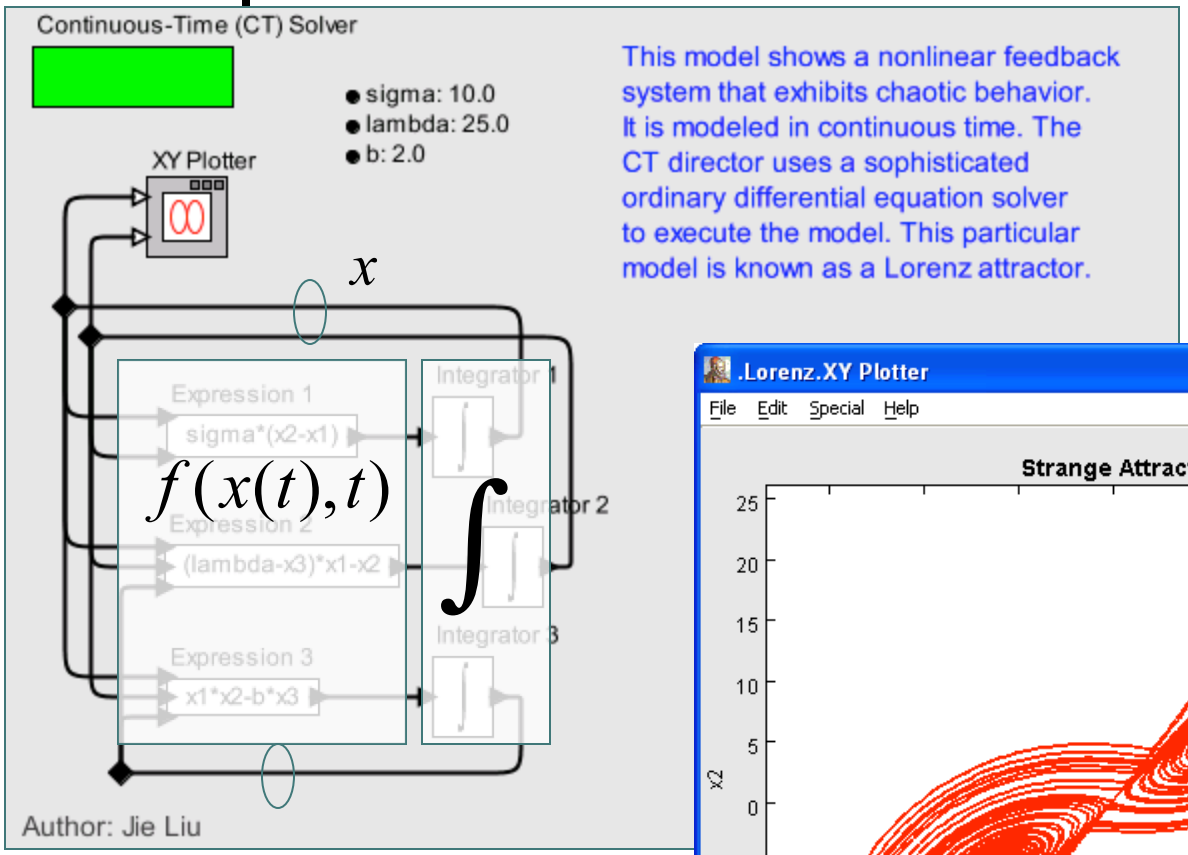
Hybrid Systems are Modal Models
Example: Spring-mass system with collisions

A model of a spring-mass system with collisions:

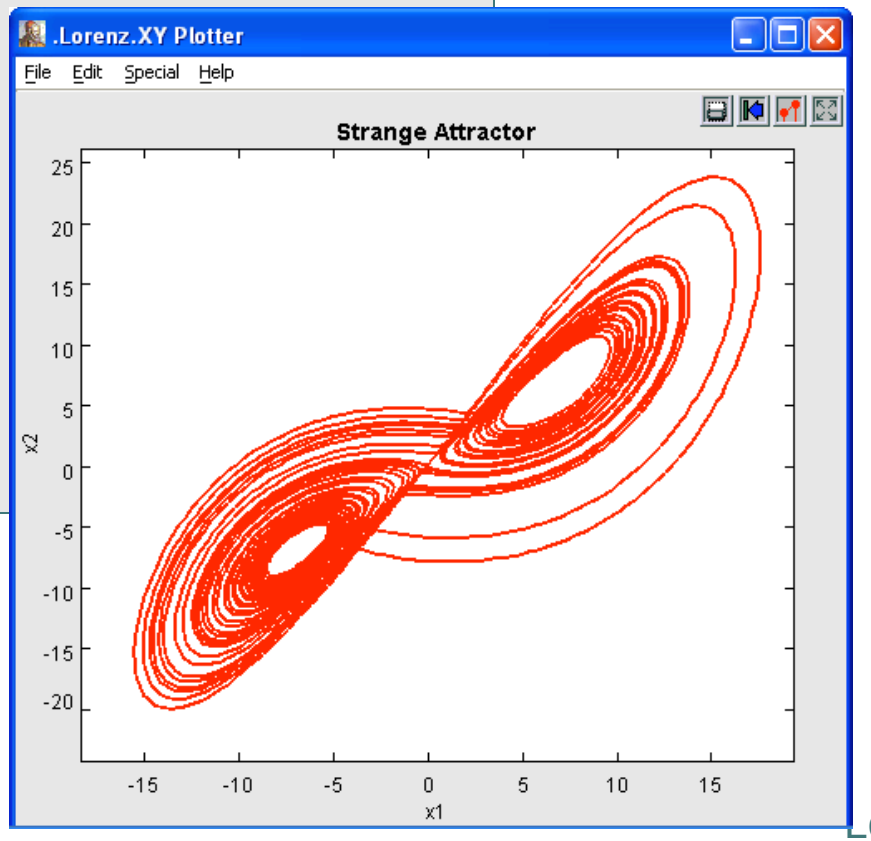


Consider the velocity of each mass. Is it continuous? What about the acceleration?

A bit about Simulation Technologies, both Discrete and Continuous



A basic continuous-time model describes an ordinary differential equation (ODE).

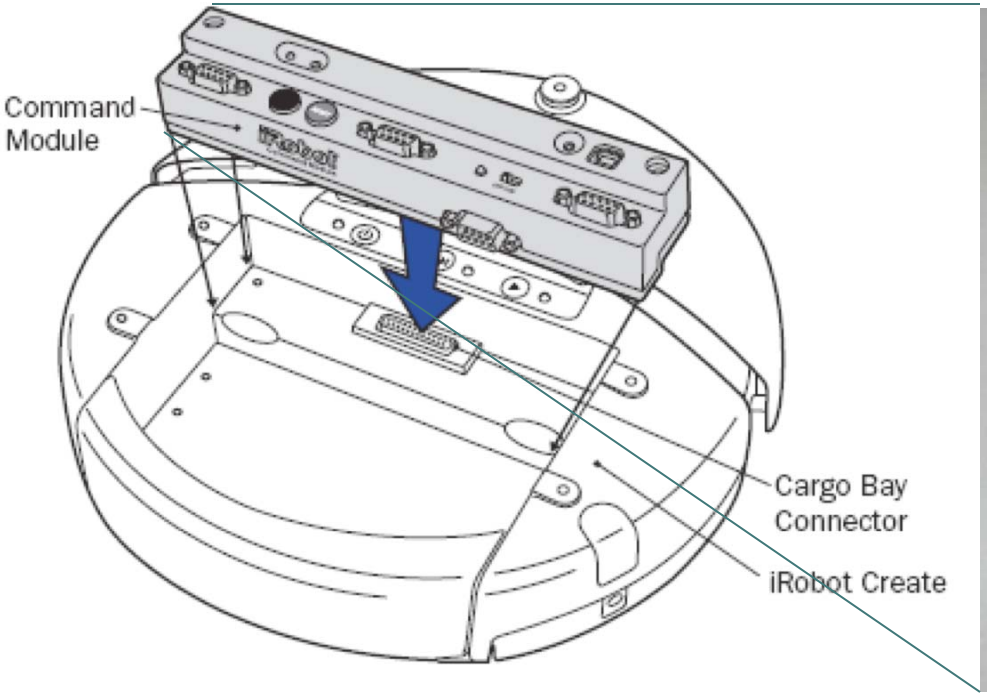


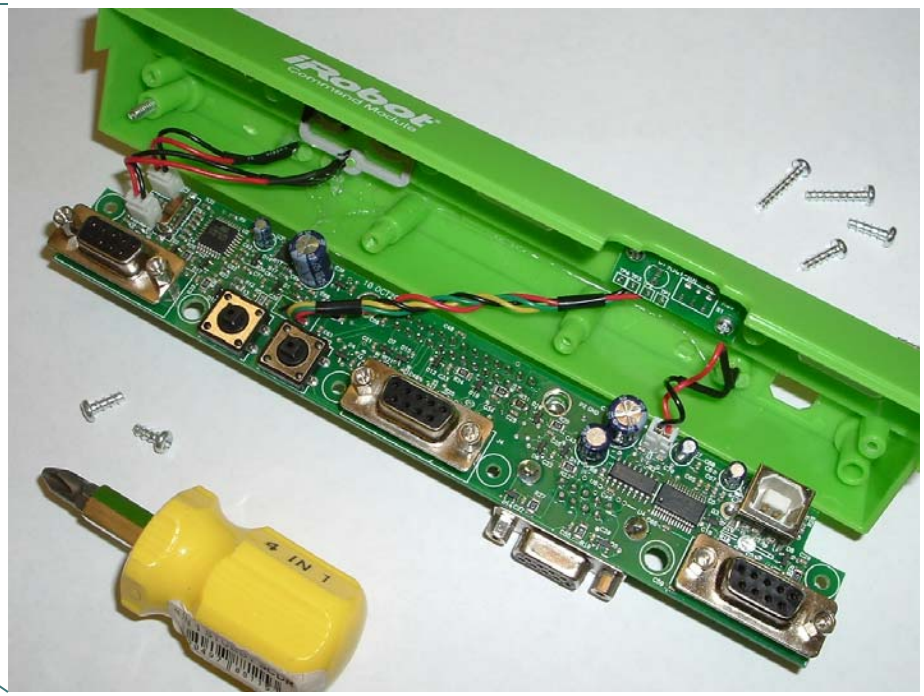
$$\dot{x}(t) = f(x(t), t)$$

$$x(t) = x(t_0) + \int_{t_0}^{t} \dot{x}(\tau) d\tau$$

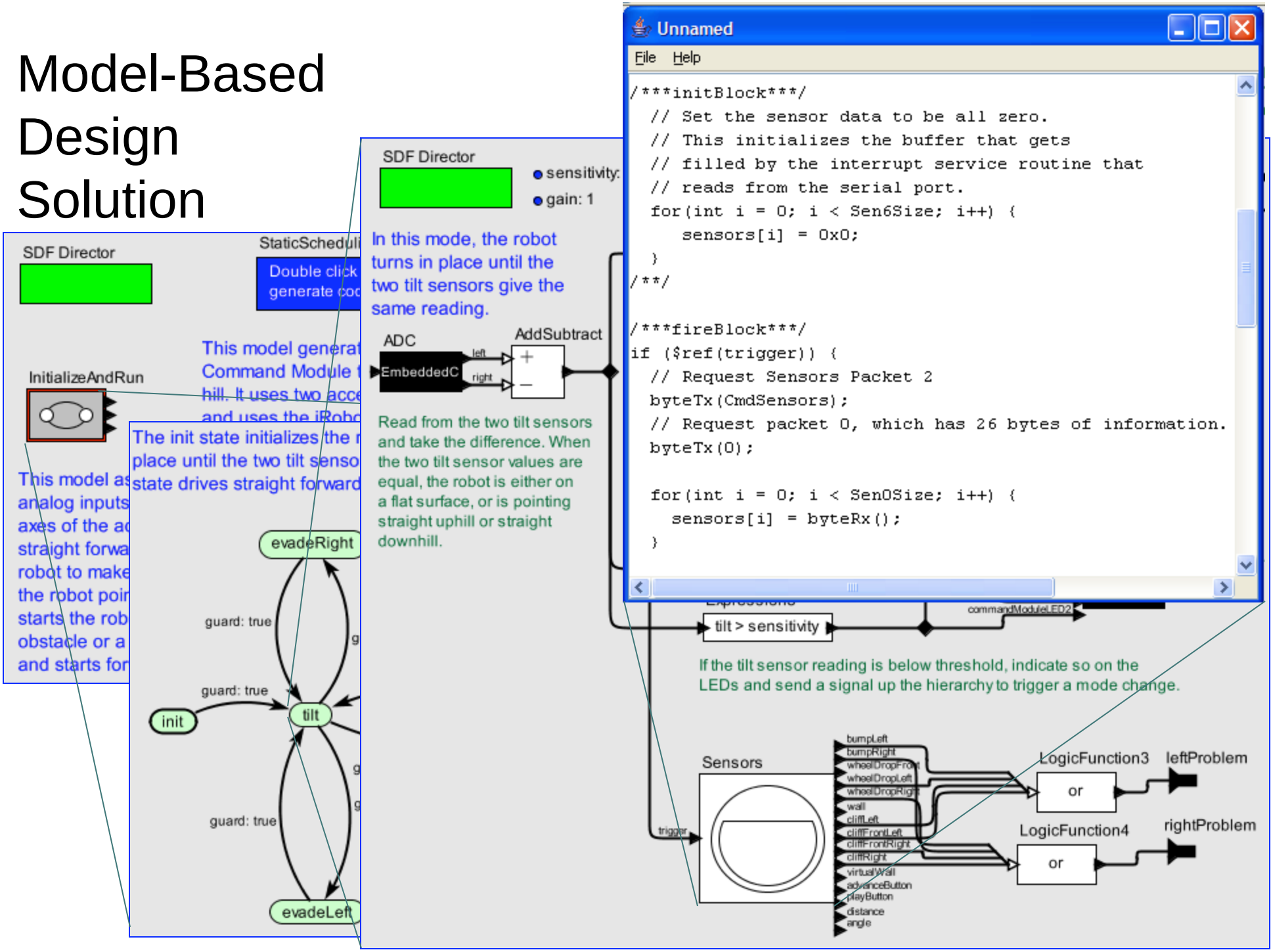
Lab Exercise – Version 1

Train a robot to climb a hill. We use the iRobot Create (the platform for the Roomba vacuum cleaner) with a pluggable Command Module containing an 8-bit Atmel microcontroller. Students have to extend it with sensors, construct models of its behavior, design a control system, and implement the control system in C.

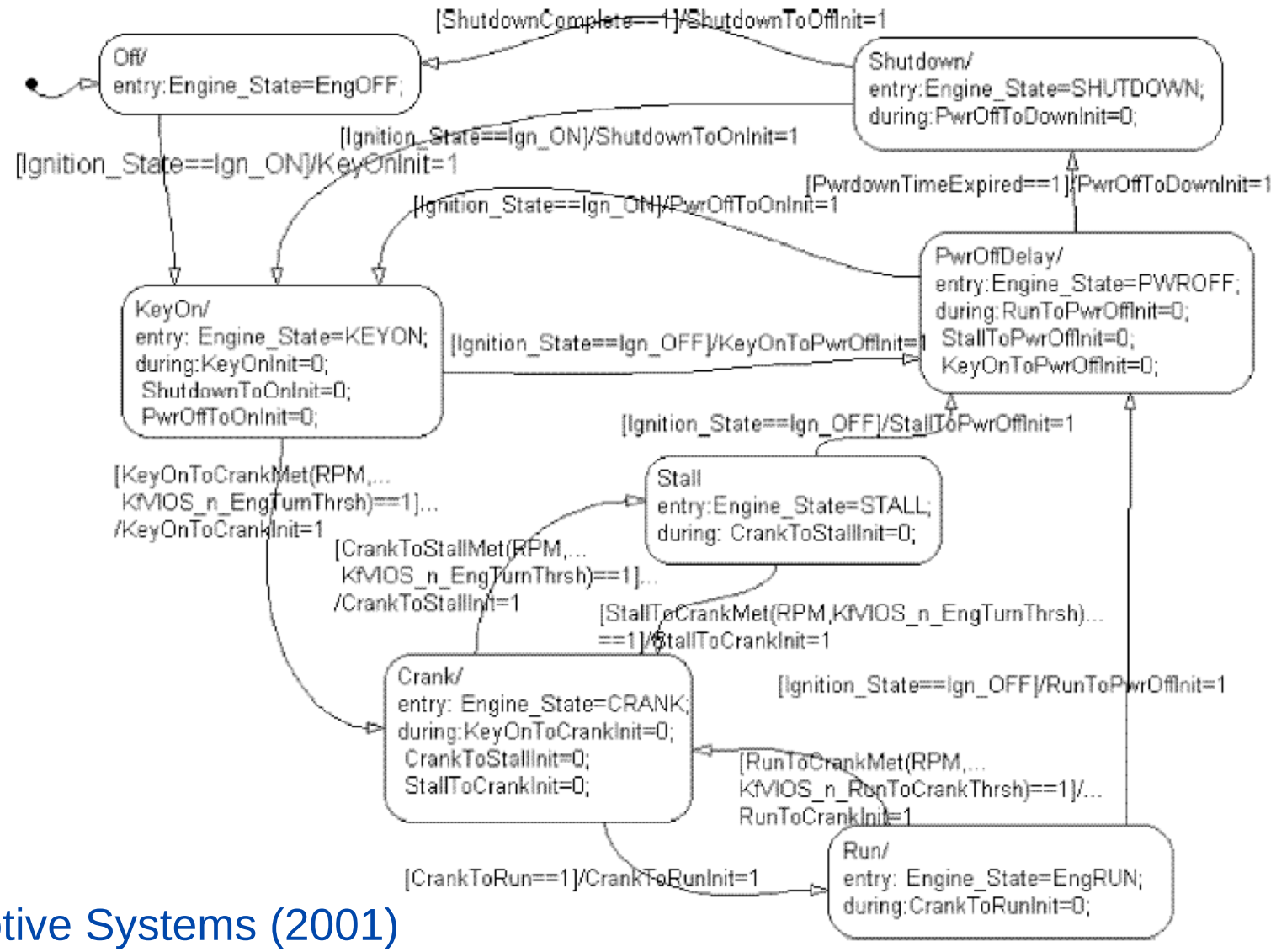




Model-Based Design Solution



Real-World Version: Modal Model (Finite-State Machine) for Engine Control

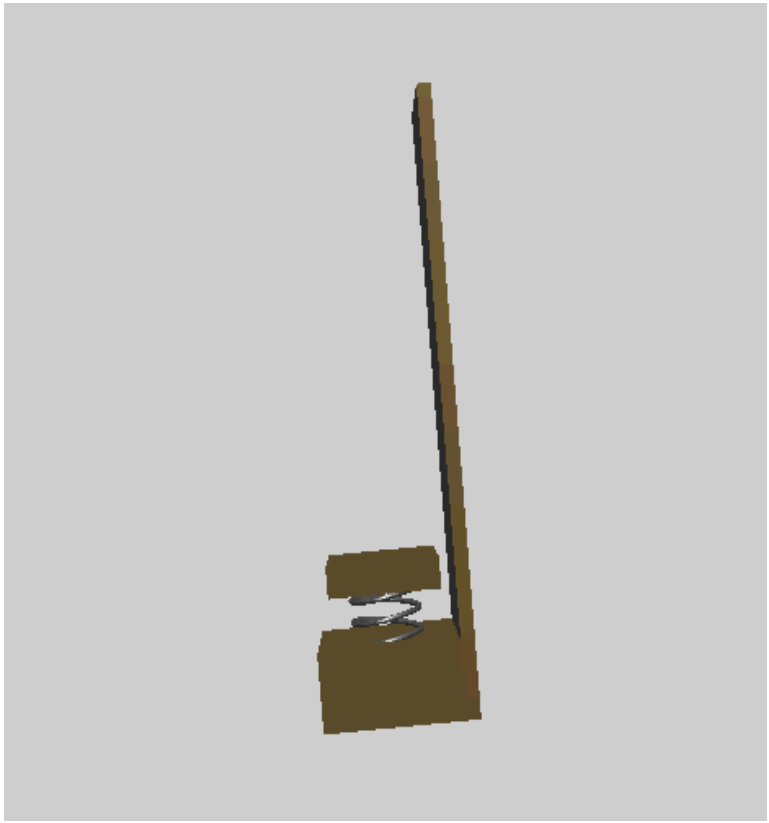


Source:

Delphi Automotive Systems (2001)

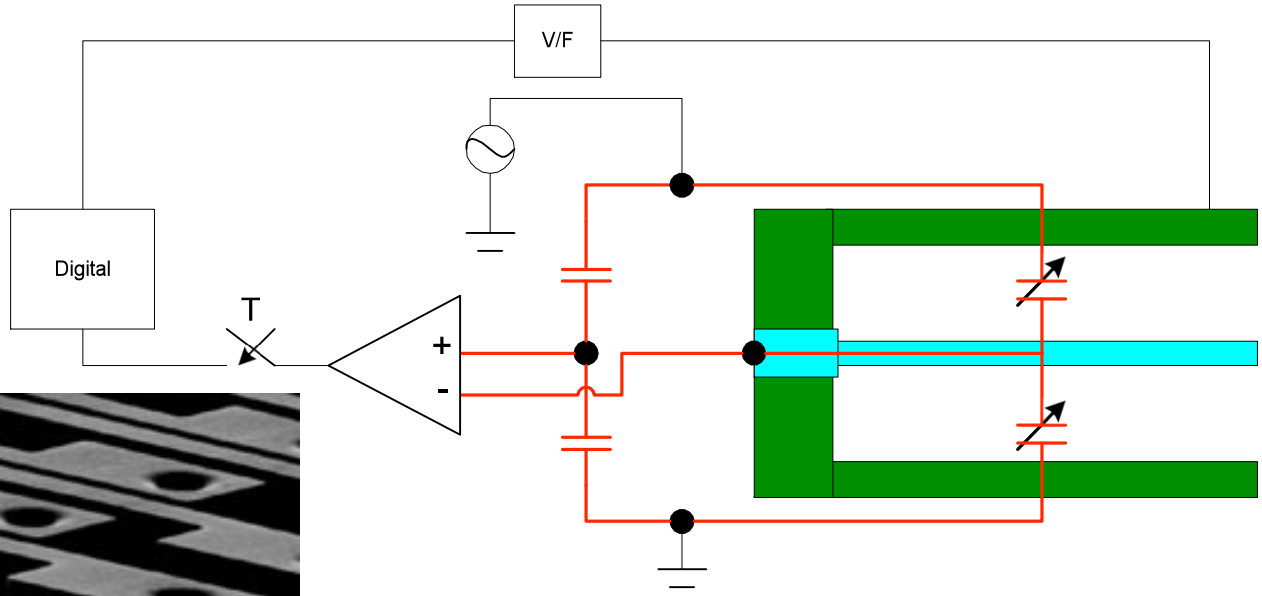
Sensors: Spring-Mass-Damper Model of an Accelerometer: Use as a Tilt Sensor

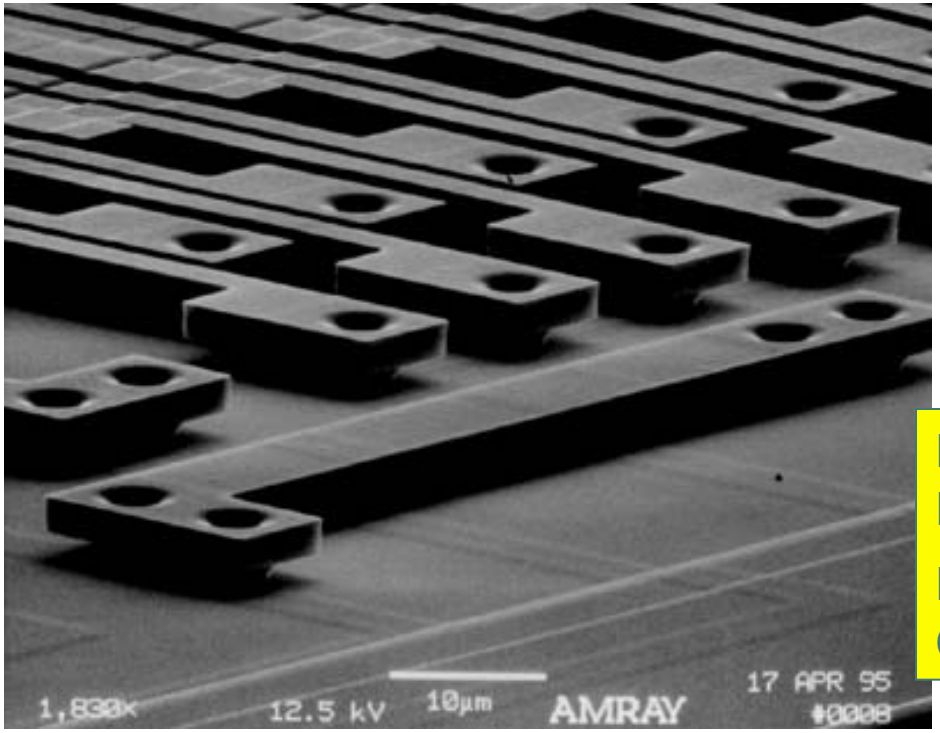
By Newton's second law, F=ma, gravitational force and acceleration have the same effect (up to a scaling constant)



Silicon Accelerometer Design comes from Berkeley

The Berkeley Sensor and Actuator Center (BSAC) created the first silicon microaccelerometers, MEMS devices now used in airbag systems, computer games, disk drives (drop sensors), etc.





M. A. Lemkin, “Micro Accelerometer Design with Digital Feedback Control”,

Ph.D. dissertation, EECS, University of California, Berkeley, Fall 1997

What Sensors to Use: Kingvale Blower Berkeley PATH Project, March, 2005

Sensors used here: Magnetometers

- A very common type is the Hall Effect magnetometer.
- Charge particles (electrons, 1) flow through a conductor (2) serving as a Hall sensor. Magnets (3) induce a magnetic field (4) that causes the charged particles to accumulate on one side of the Hall sensor, inducing a measurable voltage difference from top to bottom.
- The four drawings at the right illustrate electron paths under different current and magnetic field polarities.

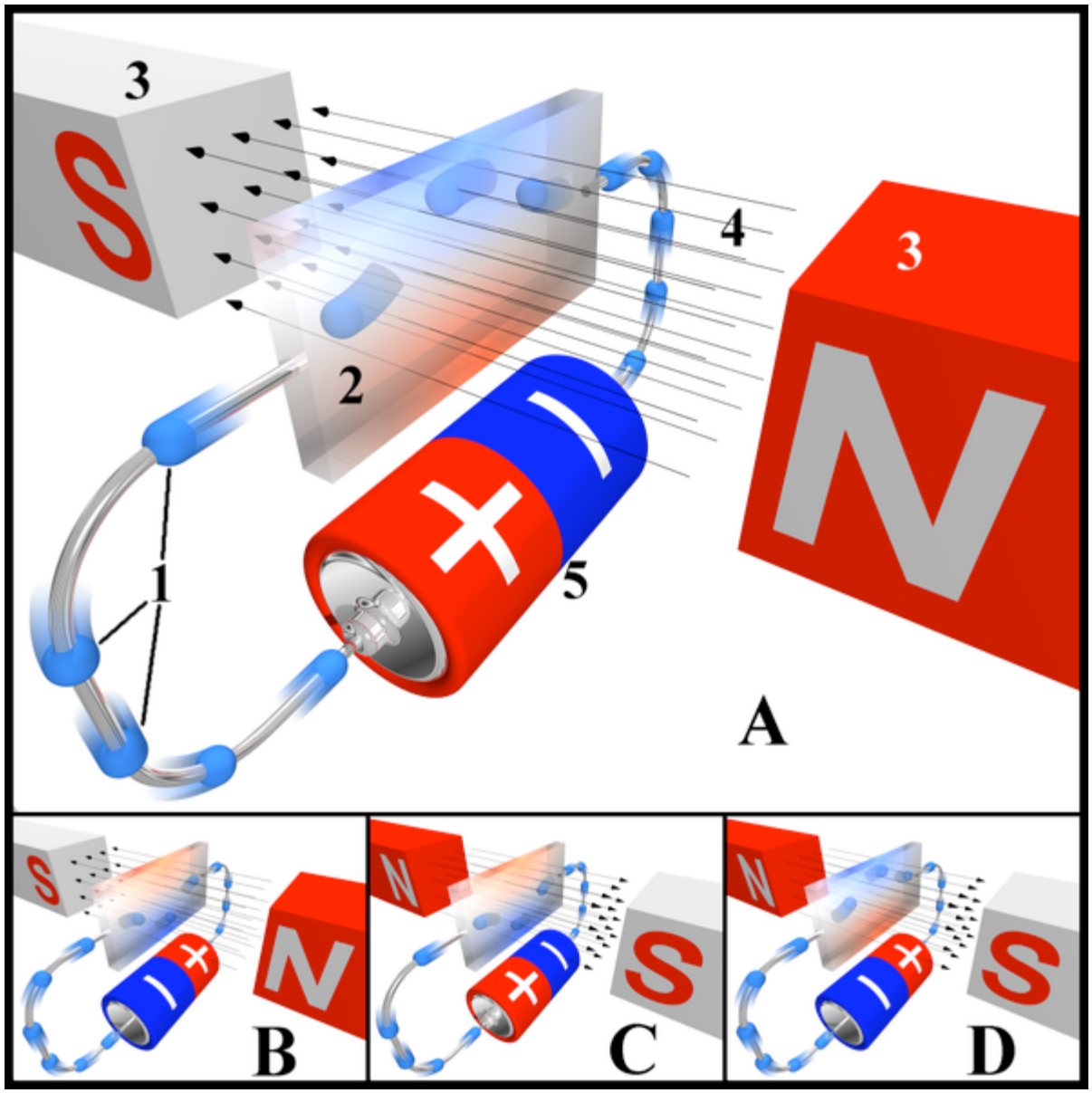


Image source: Wikipedia Commons

Edwin Hall discovered this effect in 1879.

Design Issues with Sensors

- Calibration
  - Relating measurements to the physical phenomenon
  - Can dramatically increase manufacturing costs
- Nonlinearity
  - Measurements may not be proportional to physical phenomenon
  - Correction may be required
  - Feedback can be used to keep operating point in the linear region
- Sampling
  - Aliasing
  - Missed events
- Noise
  - Analog signal conditioning
  - Digital filtering
  - Introduces latency

Students study these problems using LabVIEW.

A thought experiment:
Browser of the future? (courtesy of Ras Bodik, UCB)

Other Topics

- Temporal logic
- Reachability
- Verification
- Controller synthesis
- Concurrency: threads and interrupts
- Real-time operating systems
- Concurrent models of computation

Lab Structure

- 5 weeks of structured labs introducing students to some of the tools of the trade.
- 10 weeks of group project efforts.

Tools Used in the Lab

Nintendo Wii

iRobot Create

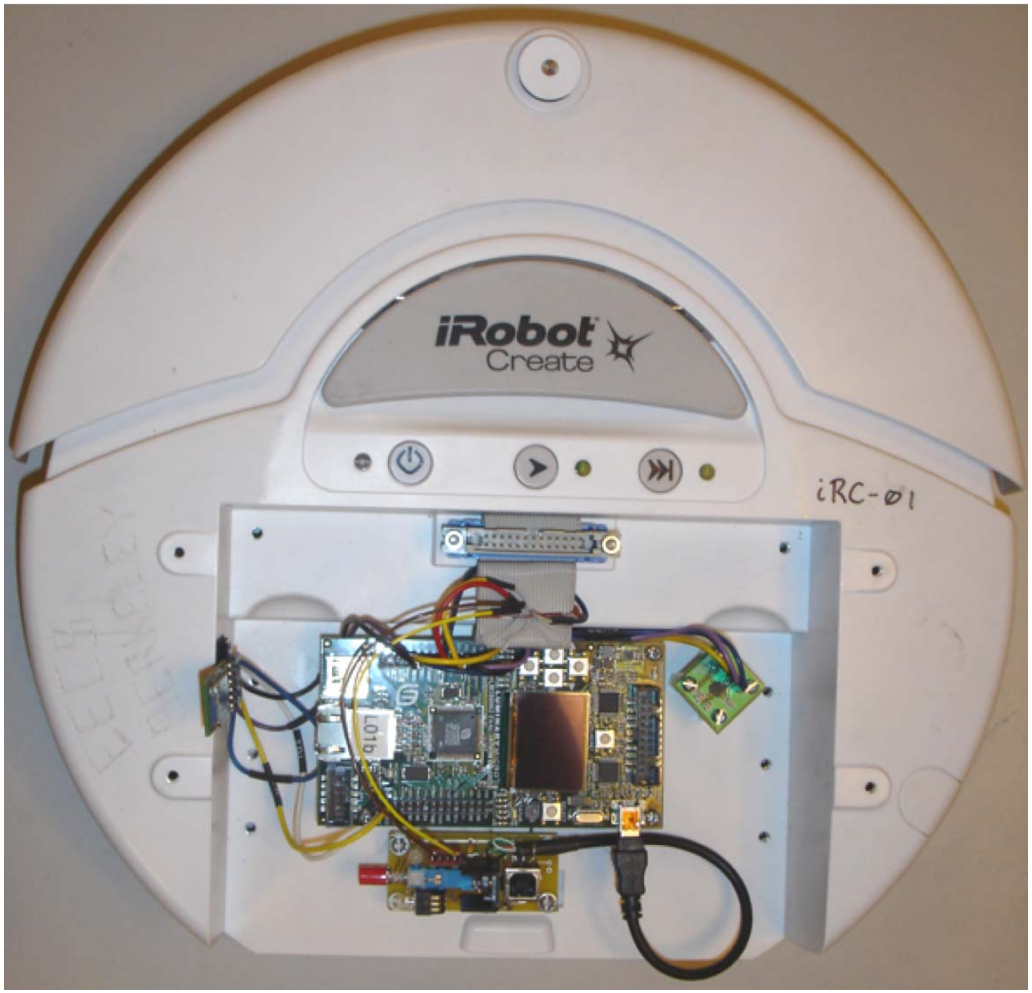
```
/* Park-Miller "minimal standard" 31 bit
 * pseudo-random number generator, implemented
 * with David G. Carta's optimisation: with
 * 32 bit math and without division.
 */

long unsigned int rand31_next()
{
    long unsigned int hi, lo;

    lo = 16807 * (seed & 0xFFFF);
    hi = 16807 * (seed >> 16);

    lo += (hi & 0x7FFF) << 16;
    lo += hi >> 15;

    if (lo > 0x7FFFFFFF) lo -= 0x7FFFFFFF;

    return ( seed = (long)lo );
}
```

Embedded C

LabVIEW

Microcontroller

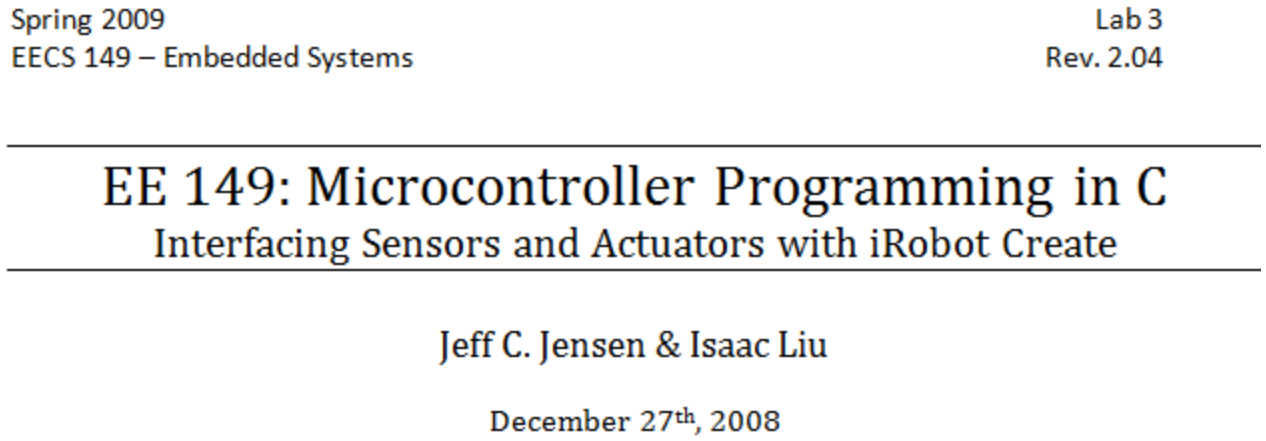
Version 2: Modified Platform



- *iRobot Create*
- *Luminary Micro LM3s8962 (ARM)*
- *ADXL-322 accelerometer*
- *BlueSMiRF Bluetooth radio*
- *Custom PCB for power and USB*

Structured Labs followed by Team Projects

- *Hill Climb*
  - *Sensing*
  - *Controller synthesis*
  - *Actuation*
- *Embedded C*
- *NI LabVIEW*
  - *Simulation*
  - *Embedded*



Spring 2009
EECS 149 – Embedded Systems

Lab 3
Rev. 2.04

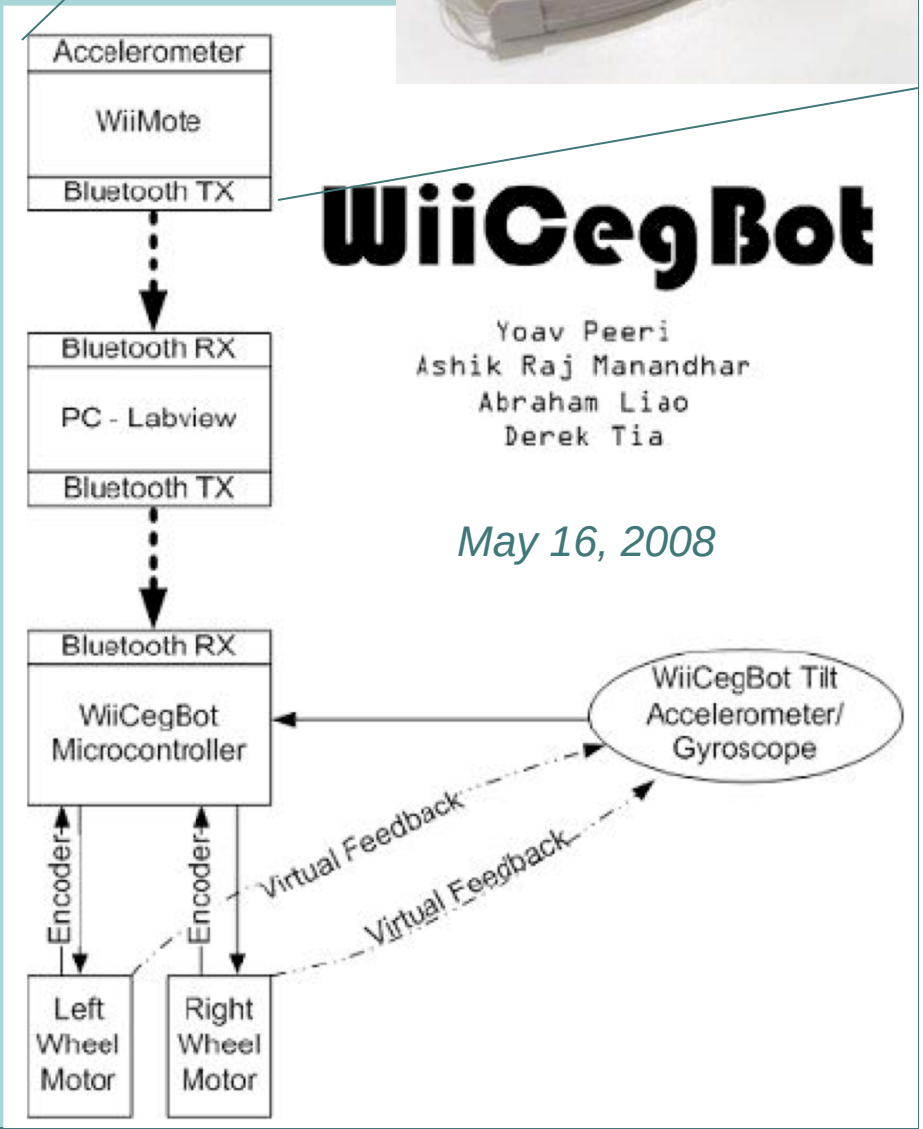
EE 149: Microcontroller Programming in C
Interfacing Sensors and Actuators with iRobot Create

Jeff C. Jensen & Isaac Liu

December 27th, 2008

Example Class Project

One of the five project teams developed a balancing robot inspired by the Segway. They used a Nintendo Wiimote as a controller communicating with a PC running LabVIEW, communicating with a Lego Mindstorm NXT, which they programmed in C.



WiiCegBot

Yoav Peeri
Ashik Raj Manandhar
Abraham Liao
Derek Tia

May 16, 2008

Example Class Project

Robot Mapping & Localization

Richard Schmidt
Christopher Bakker
Jeff C. Jensen

This team mounted lasers and photo detectors on instances of the iRobot Create platform and programmed the robots to identify their positions relative to one another and communicate wirelessly to a mapping application running on a host PC.

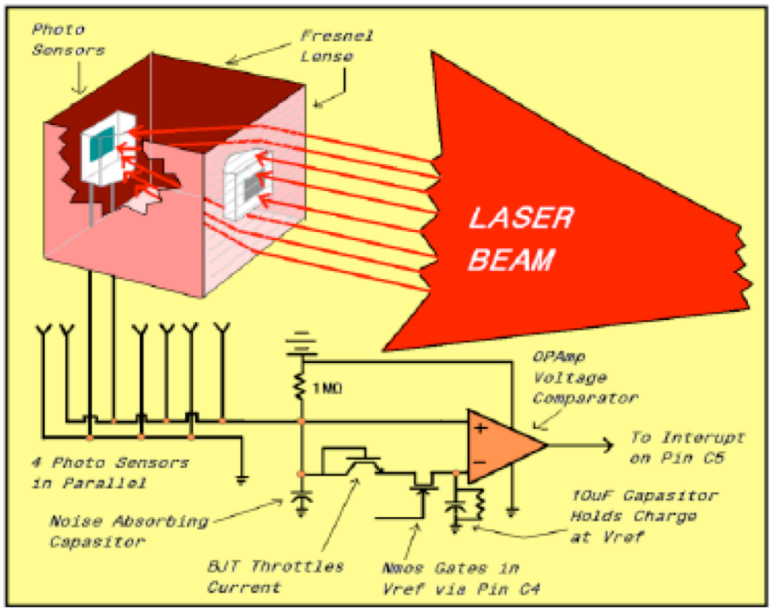


Figure 5: Laser detector (sensor)

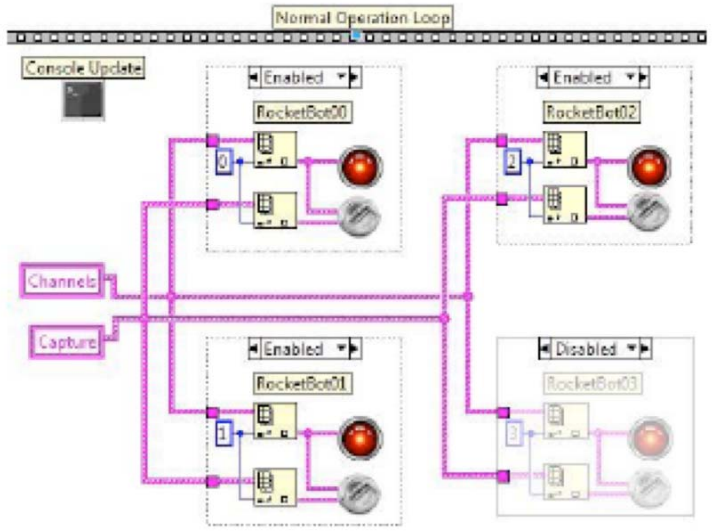
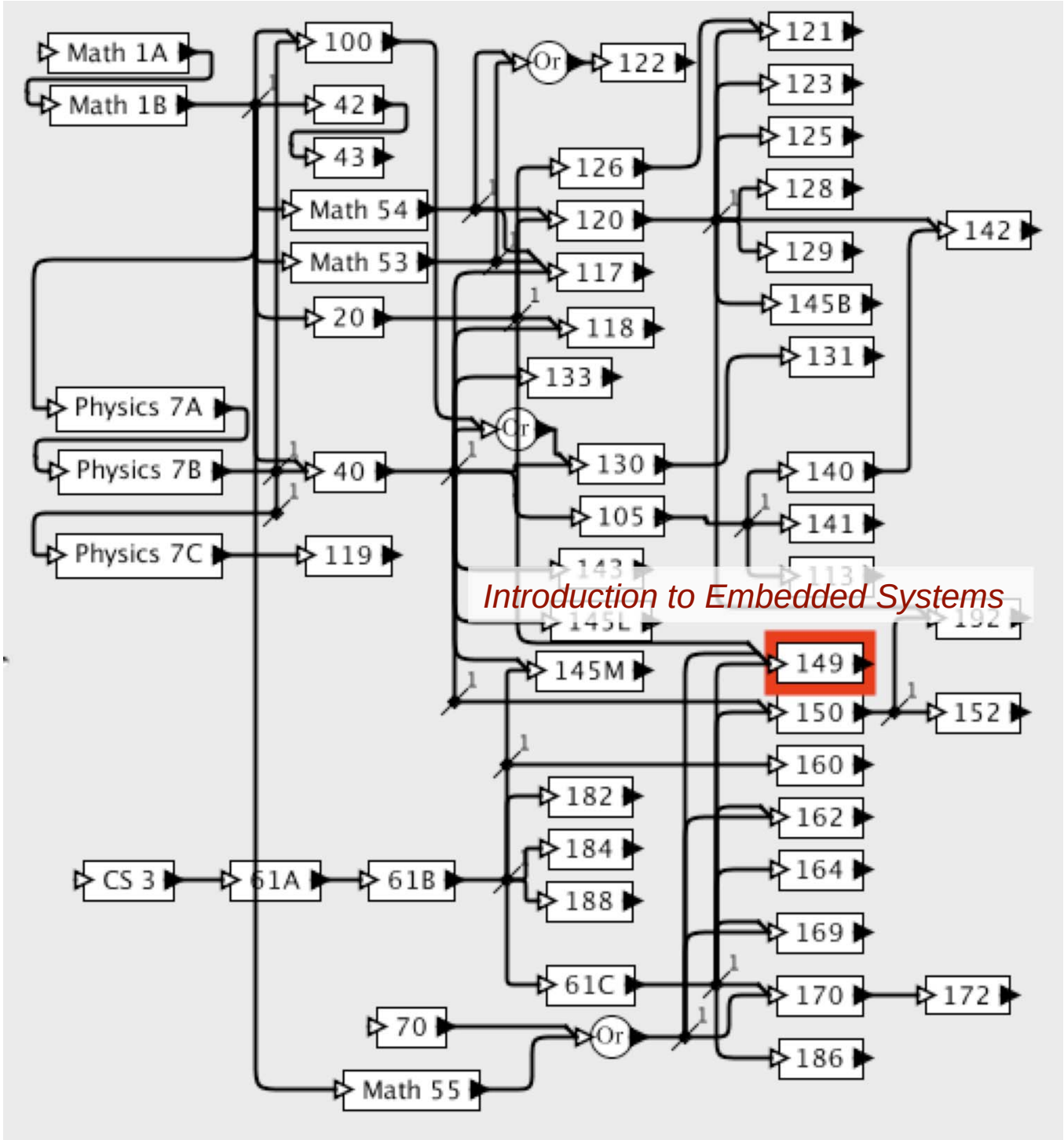


Figure 3: Concurrent robot communication / DLL pairs.

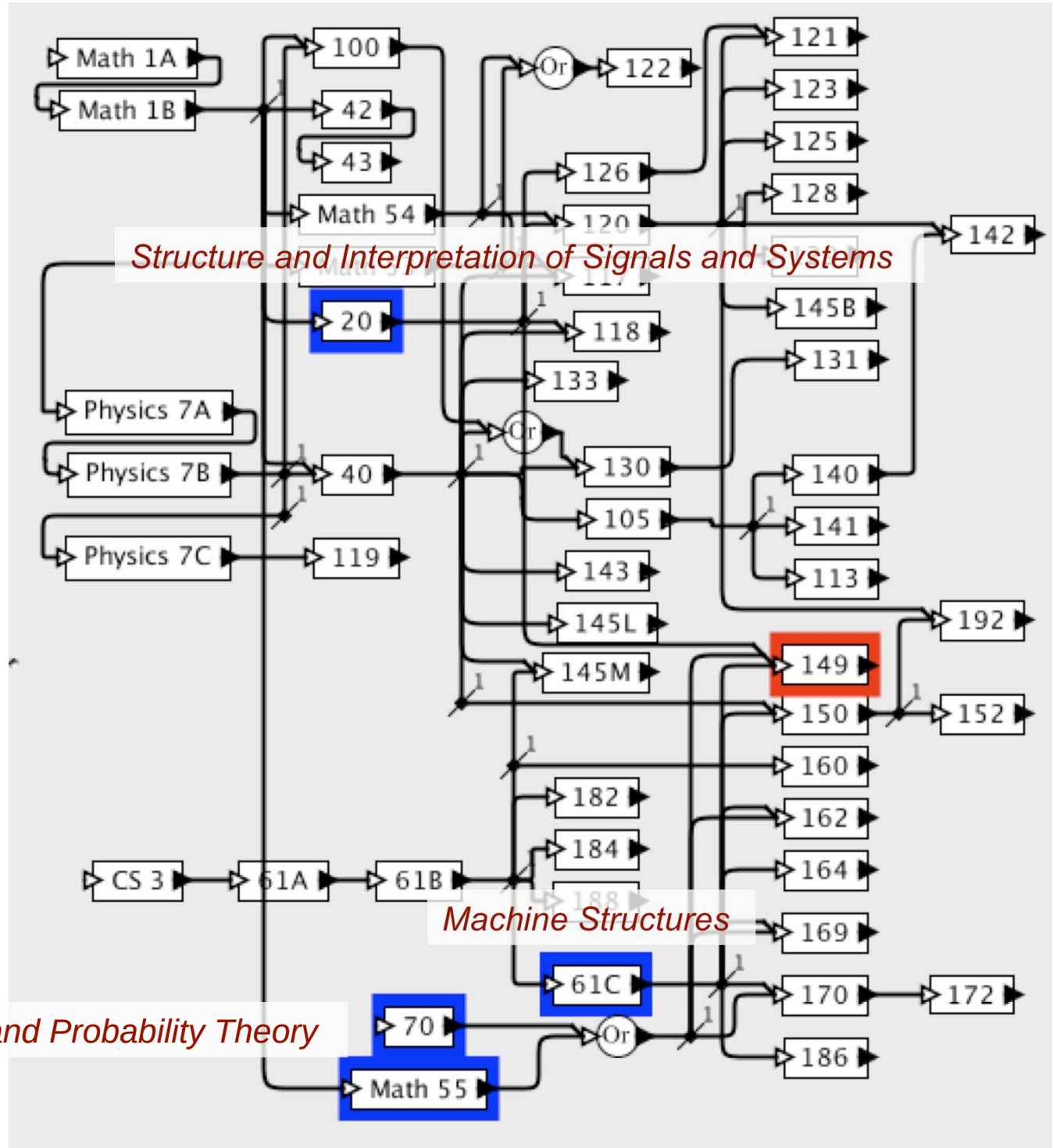
The Prerequisite Challenge

- Core course



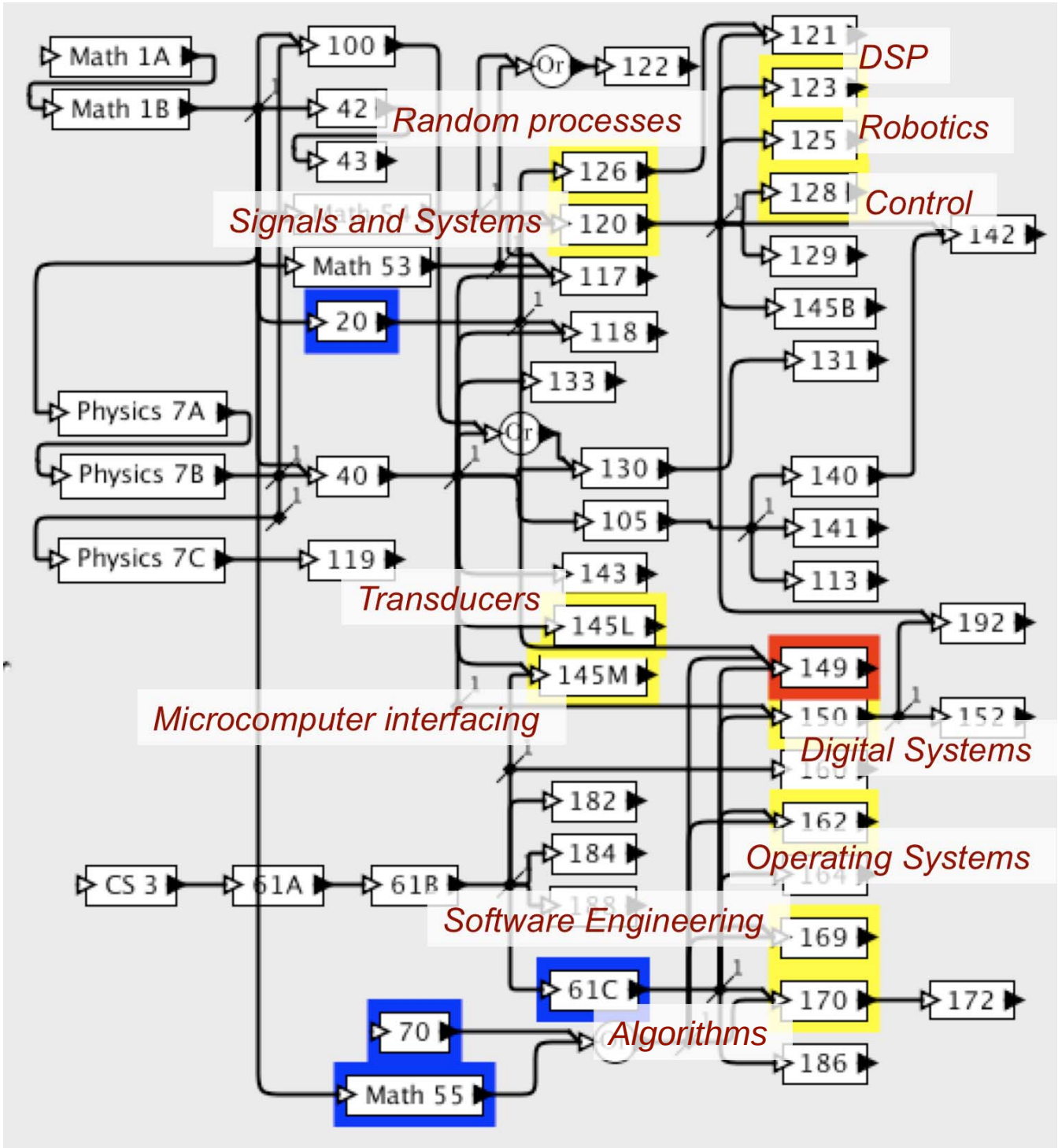
The Prerequisite Challenge

- Core course
- Prerequisites



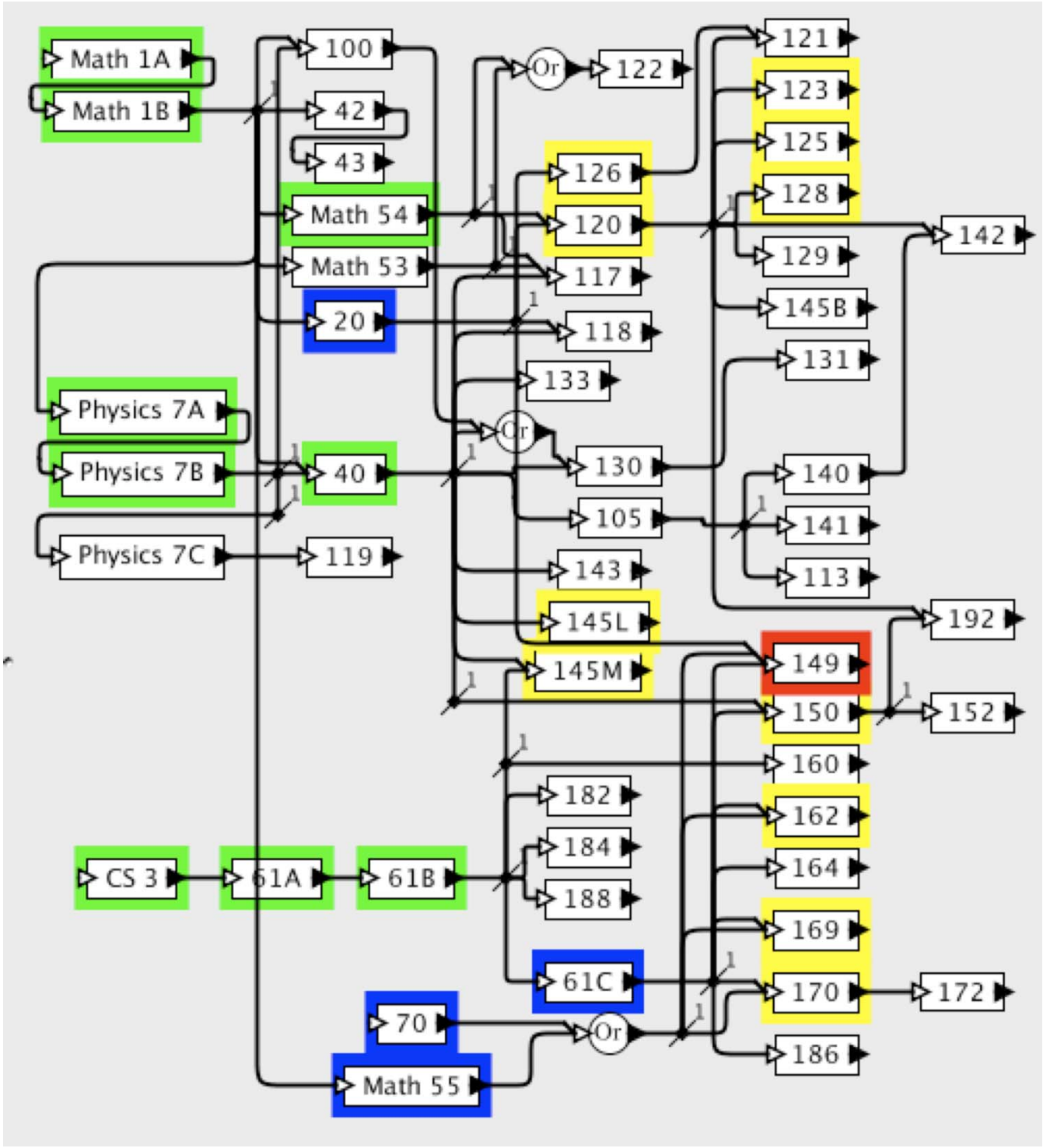
The Prerequisite Challenge

- Core course
- Prerequisites
- Defensible prerequisites



The Prerequisite Challenge

- Core course
- Prerequisites
- Defensible prerequisites
- Transitive closure



Conclusion

A *reductionist approach* (where we teach all the foundations first) cannot work!

We have to be prepared to review topics as needed, and deal with a mix of backgrounds in the students.

The intellectual core is MODELS (as approximations, as abstractions, as ways of understanding interaction, etc.).