Fundamental Algorithms for System Modeling, Analysis, and Optimization

Edward A. Lee, Jaijeet Roychowdhury, Sanjit A. Seshia, Stavros Tripakis

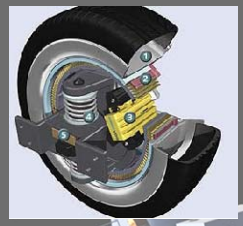
UC Berkeley

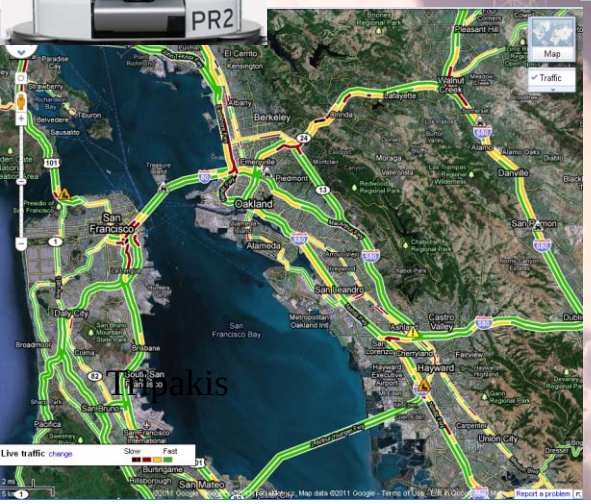
EECS 144/244

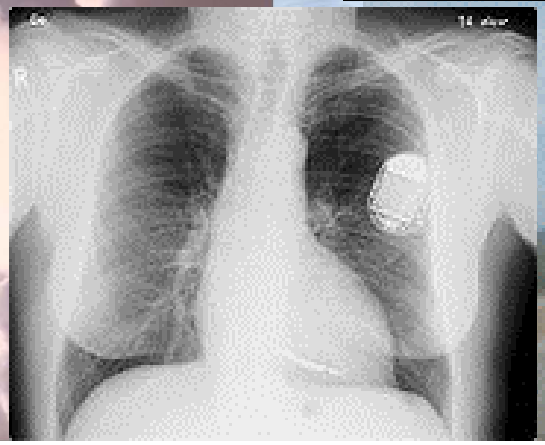
Fall 2013

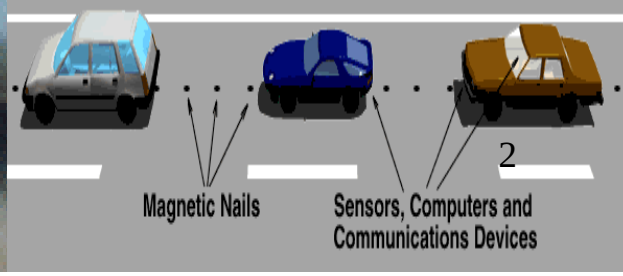
Lecture 1: Introduction, Logistics

Computers as parts of systems



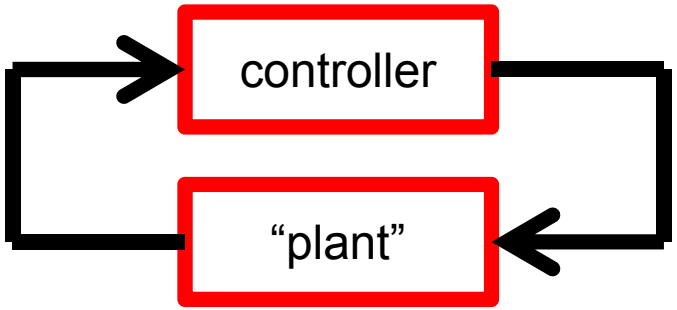






Embedded, Cyber-Physical Systems

- Computers (HW+SW) “embedded” in a physical world
- Cyber = computers (literally “to govern”)
- Physical = the rest
- Typically in a closed-loop (feedback) control configuration (often there are many distributed controllers)

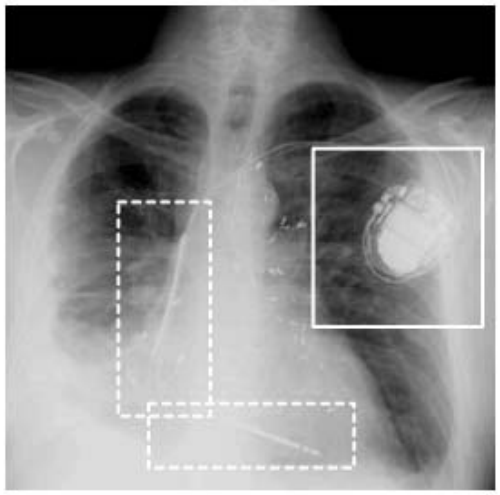


Computers can be dangerous

- This has been known for some time now...

Computers can be dangerous

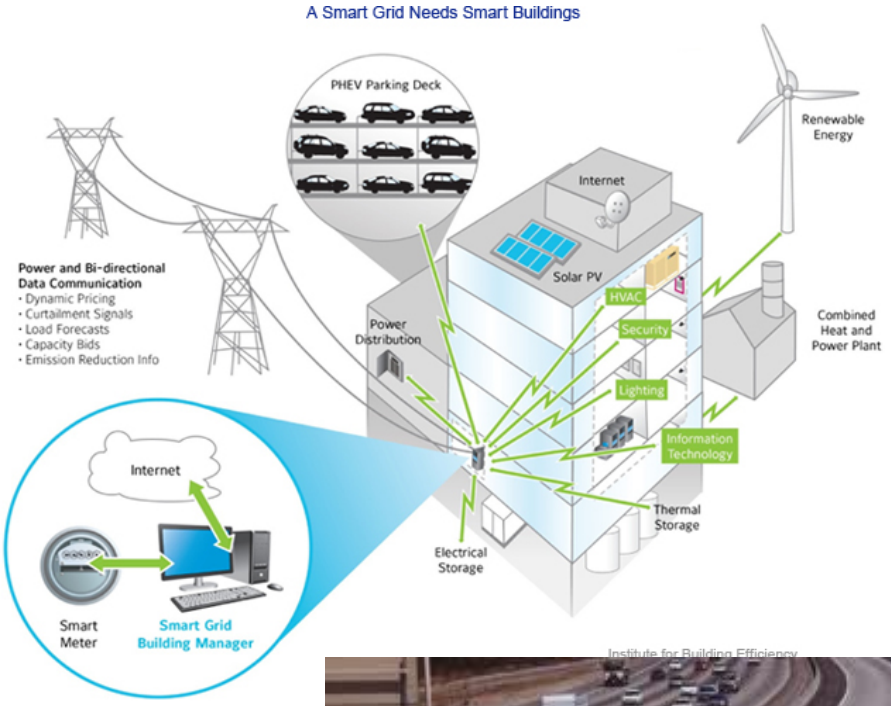
- An important concern today:

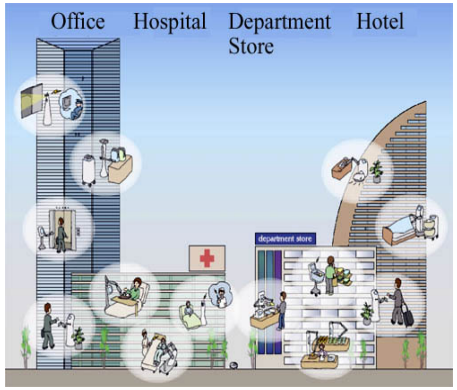


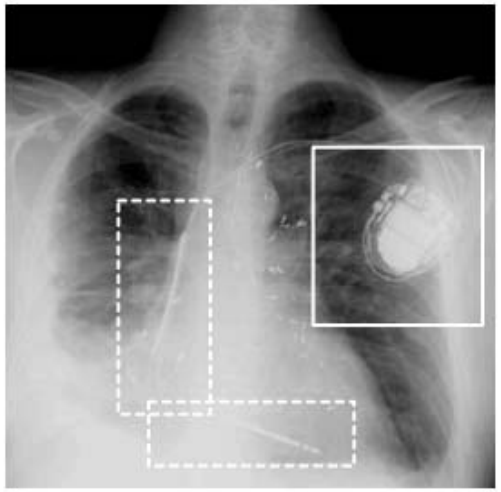
...

Computers can be extremely beneficial!

- Smart cars, roads, buildings, power grid, cities, ...







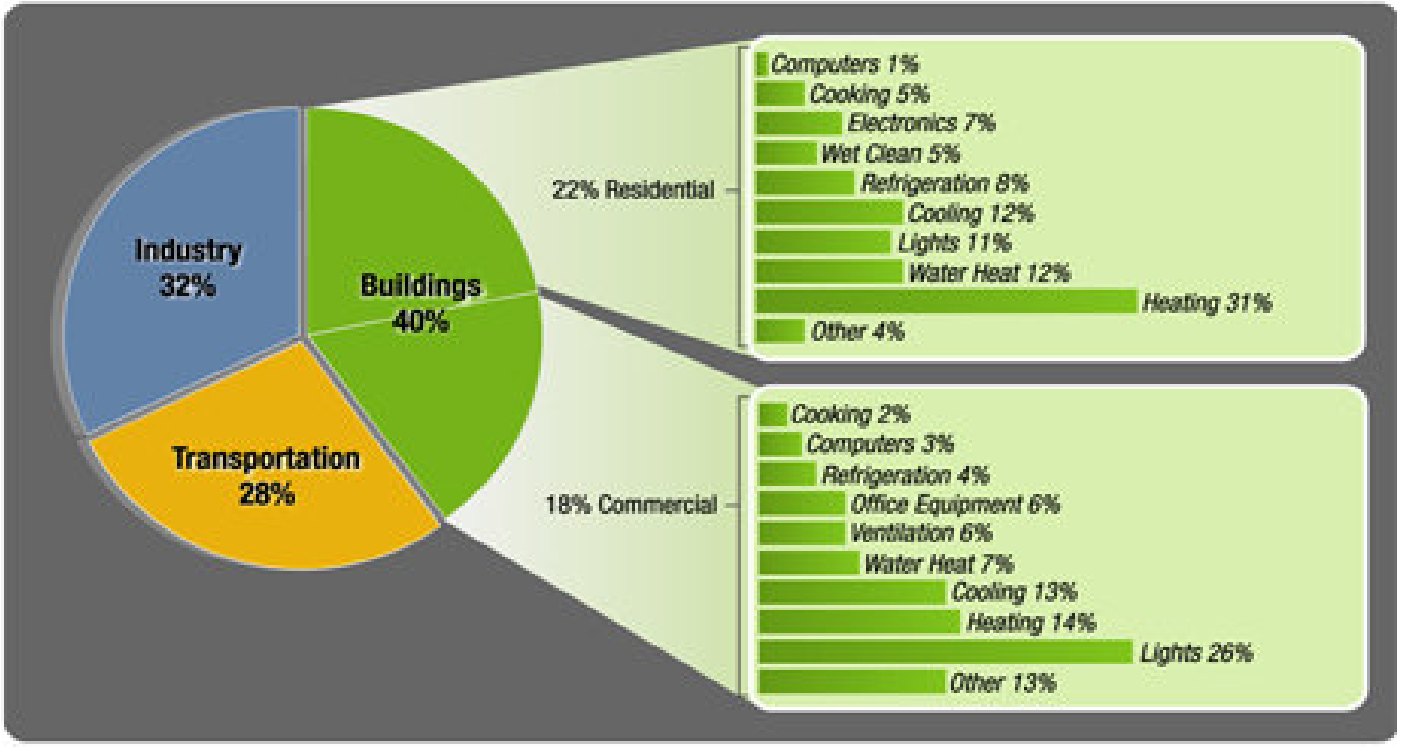
...

Smart cars on smart roads

- PATH project (Partners for Advanced Transportation TecHnology) of UC Berkeley:
  - http://www.path.berkeley.edu/
- Demo'97:
  - http://www.youtube.com/watch?v=C9G6JRUmg_A

Smart / green buildings

- Source: Steven Chu (ex US Secretary of Energy) http://www.facebook.com/notes/steven-chu/this-might-surpise-you/138532906856

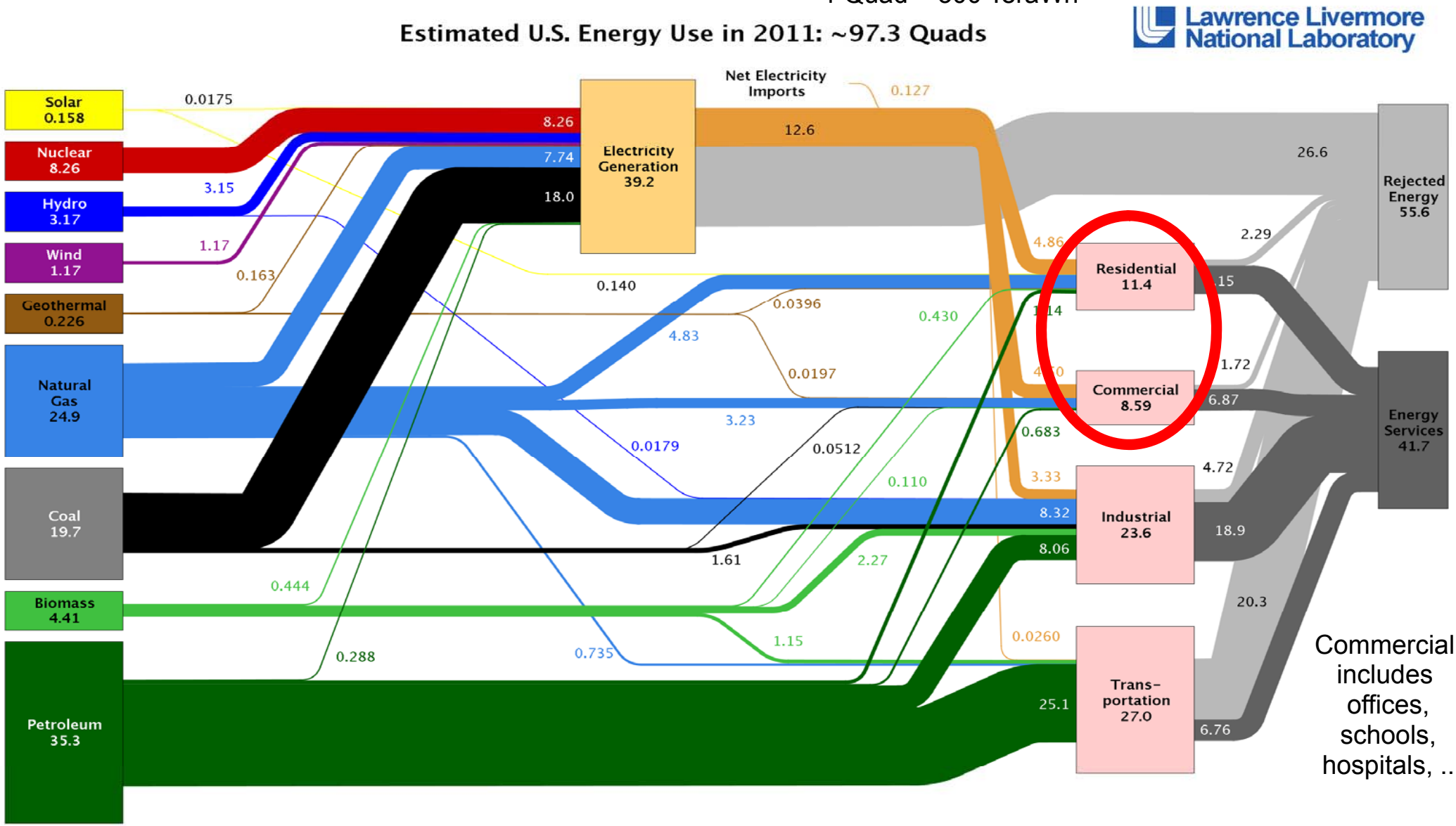


Buildings use about 40 percent of total U.S. energy

Smart / green buildings

- Source: https://flowcharts.llnl.gov/

1 Quad ~ 300 TeraWh

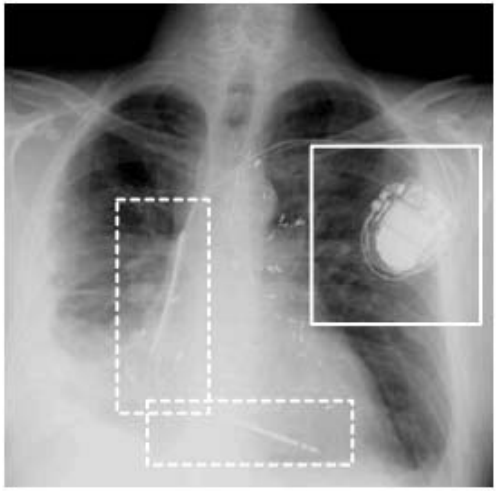


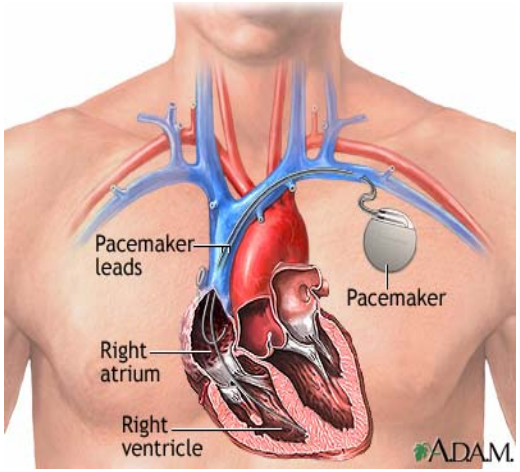
Commercial: includes offices, schools, hospitals, ...

Source: LLNL 2012. Data is based on DOE/EIA-0384(2011), October, 2012. If this information or a reproduction of it is used, credit must be given to the Lawrence Livermore National Laboratory and the Department of Energy, under whose auspices the work was performed. Distributed electricity represents only retail electricity sales and does not include self-generation. EIA reports flows for non-thermal resources (i.e., hydro, wind and solar) in BTU-equivalent values by assuming a typical fossil fuel plant "heat rate." The efficiency of electricity production is calculated as the total retail electricity delivered divided by the primary energy input into electricity generation. End use efficiency is estimated as 80% for the residential, commercial and industrial sectors, and as 25% for the transportation sector. Totals may not equal sum of components due to independent rounding. LLNL-MI-410527

Health care

- Pacemakers:





Pacemakers and Implantable Cardiac Defibrillators: Software Radio Attacks and Zero-Power Defenses

Daniel Halperin† University of Washington
Thomas S. Heydt-Benjamin† University of Massachusetts Amherst
Benjamin Ransford† University of Massachusetts Amherst
Shane S. Clark University of Massachusetts Amherst
Benessa Defend University of Massachusetts Amherst
Will Morgan University of Massachusetts Amherst
Kevin Fu, PhD* University of Massachusetts Amherst
Tadayoshi Kohno, PhD* University of Washington
William H. Maisel, MD, MPH* BIDMC and Harvard Medical School

***Abstract*—Our study analyzes the security and privacy properties of an implantable cardioverter defibrillator (ICD). Introduced to the U.S. market in 2003, this model of ICD includes pacemaker technology and is designed to communicate wirelessly with a nearby external programmer in the 175 kHz frequency range. After partially reverse-engineering the ICD's communications protocol with an oscilloscope and a software radio, we implemented several software radio-based attacks that could compromise patient safety and patient privacy. Motivated by our desire to improve patient safety, and mindful of conventional trade-offs between security and power consumption for resource-constrained devices, we introduce three new zero-power defenses based on RF power harvesting. Two of these defenses are human-centric, bringing patients into the loop with respect to the security and privacy of their implantable medical devices (IMDs). Our contributions provide a scientific baseline for understanding the potential security and privacy risks of current and future IMDs, and introduce human-perceptible and zero-power mitigation techniques that address those risks. To the best of our knowledge, nmunity to use general-purpose ttack previously unknown radio**

DUCTION

implantable medical devices plantable cardioverter defibril-, and implantable *drug pumps* adios to monitor chronic disor-

this event to a health care practitioner who uses a *commercial device programmer*[1] with wireless capabilities to extract data from the ICD or modify its settings without surgery. Between 1990 and 2002, over 2.6 million pacemakers and ICDs were implanted in patients in the United States [19]; clinical trials have shown that these devices significantly improve survival rates in certain populations [18]. Other research has discussed potential security and privacy risks of IMDs [1], [10], but we are unaware of any rigorous public investigation into the observable characteristics of a real commercial device. Without such a study, it is impossible for the research community to assess or address the security and privacy properties of past, current, and future devices. We address that gap in this paper and, based on our findings, propose and implement several prototype attack-mitigation techniques.

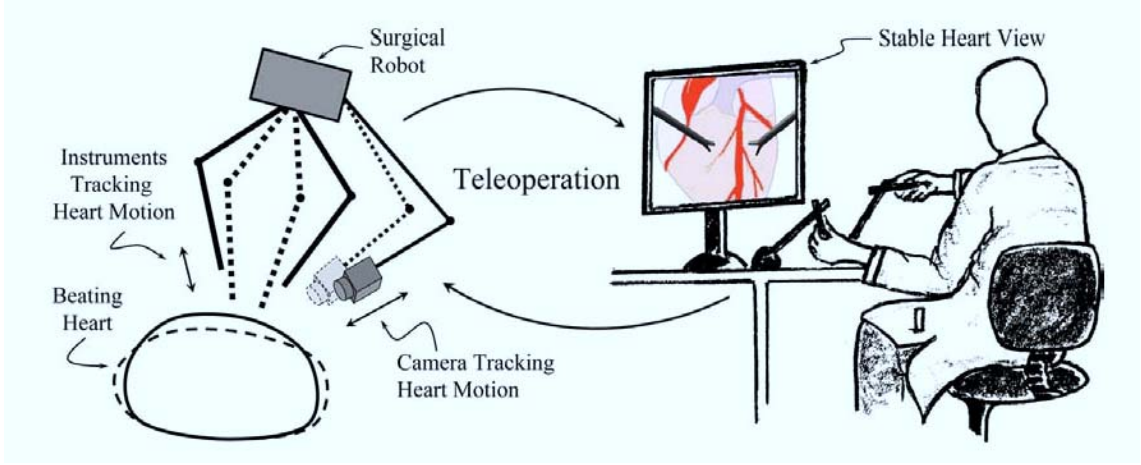
Our investigation was motivated by an interdisciplinary study of medical device safety and security, and relied on a diverse team of area specialists. Team members from the security and privacy community have formal training in computer science, computer engineering, and electrical engineering. One team member from the medical community is a practicing cardiologist with hundreds of pacemaker and implantable defibrillator patients and was past chairperson of the FDA's Circulatory System Medical Device Advisory

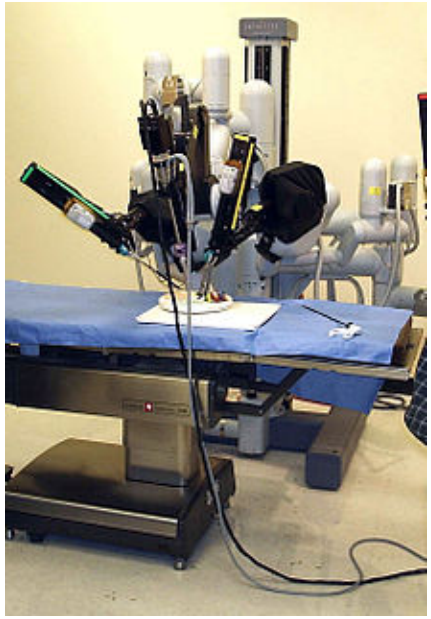
	Commercial programmer	Software radio eavesdropper	Software radio programmer	Primary risk
Determine whether patient has an ICD	✔	✔	✔	Privacy
Determine what kind of ICD patient has	✔	✔	✔	Privacy
Determine ID (serial #) of ICD	✔	✔	✔	Privacy
Obtain private telemetry data from ICD	✔	✔	✔	Privacy
Obtain private information about patient history	✔	✔	✔	Privacy
Determine identity (name, etc.) of patient	✔	✔	✔	Privacy
Change device settings	✔		✔	Integrity
Change or disable therapies	✔		✔	Integrity
Deliver command shock	✔		✔	Integrity

TABLE I
RESULTS OF EXPERIMENTAL ATTACKS. A CHECK MARK INDICATES A SUCCESSFUL IN VITRO ATTACK.

Health care

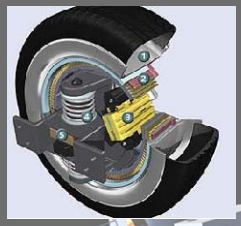
- Robotic-assisted beating heart surgery
  http://www.youtube.com/watch?v=CHnSESXK310





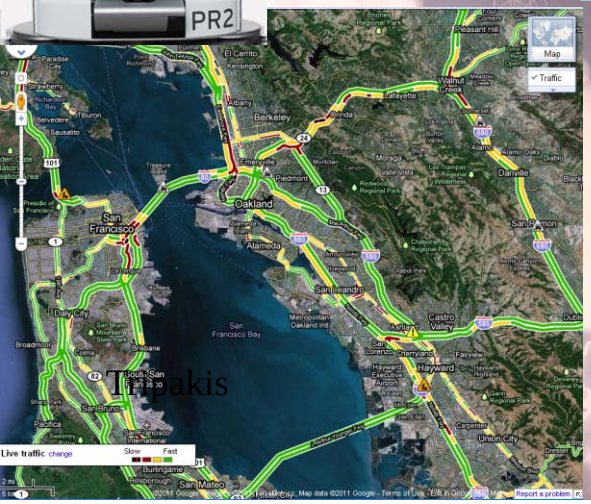
"the surgeon views the surgical scene on a video display and operates on the heart as if it were stationary while the robotic system actively compensates for the motion of the heart" [Bebek'08]

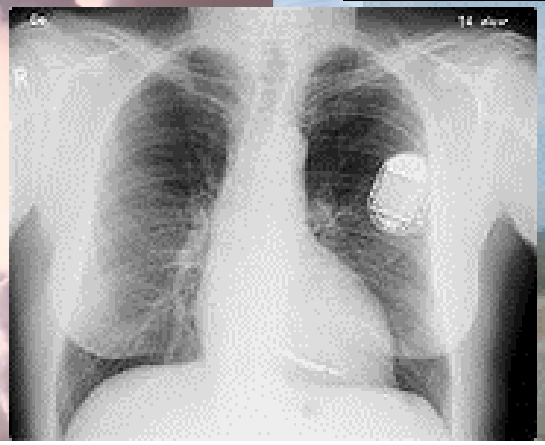
Computers as parts of **complex** systems

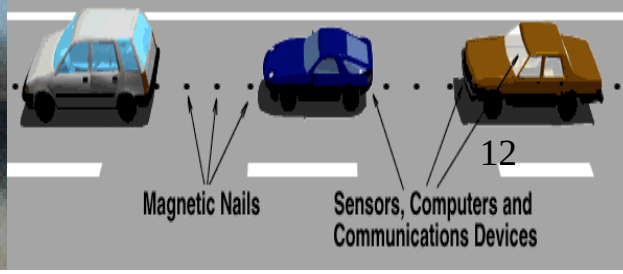


"~98% of the world's processors are not in PCs but are embedded"

"a premium car today has:
- ~80 computers (ECUs – Electronic Control Units)
- ~100 million lines of code"

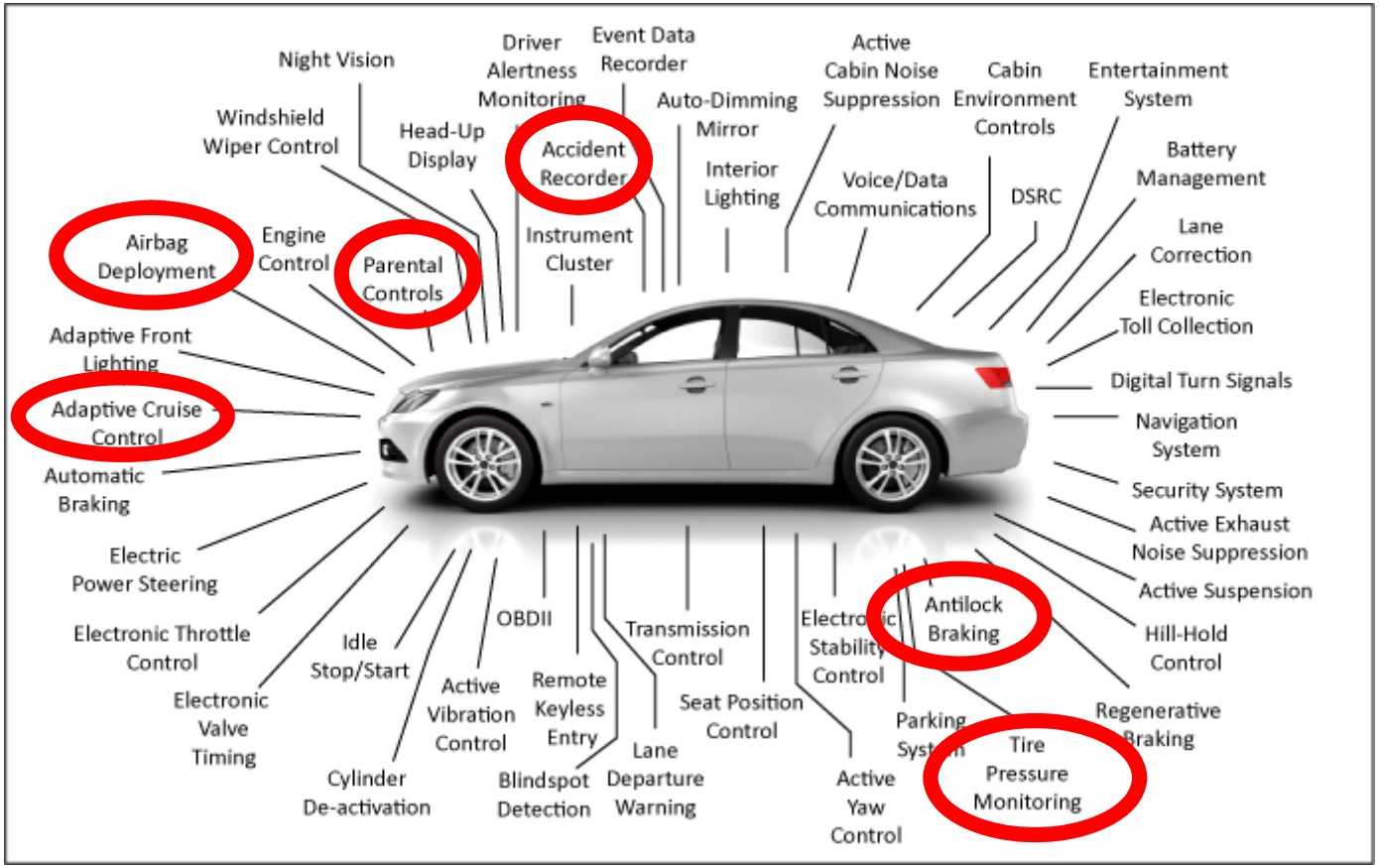






Automotive

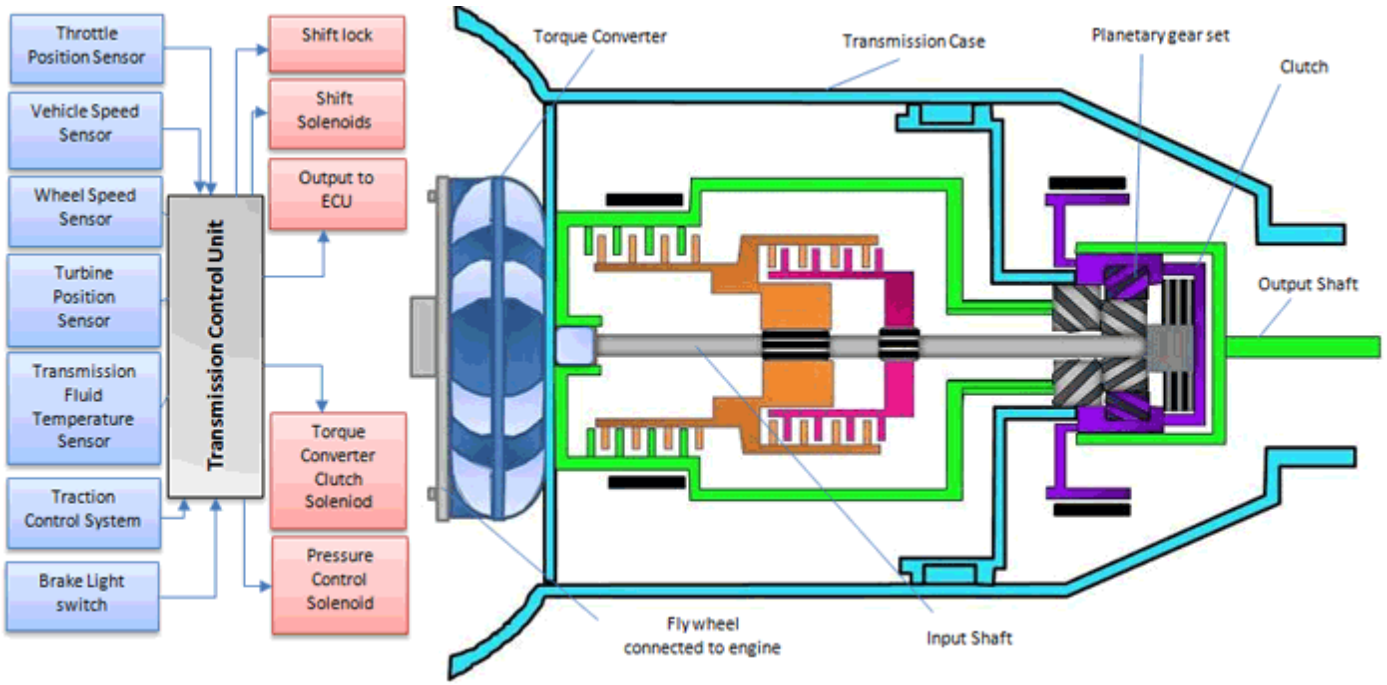
- Computer-controlled automotive systems:



- Source: http://www.cvel.clemson.edu/auto/systems/auto-systems.html (image items are clickable)

Automotive

- E.g., transmission control:

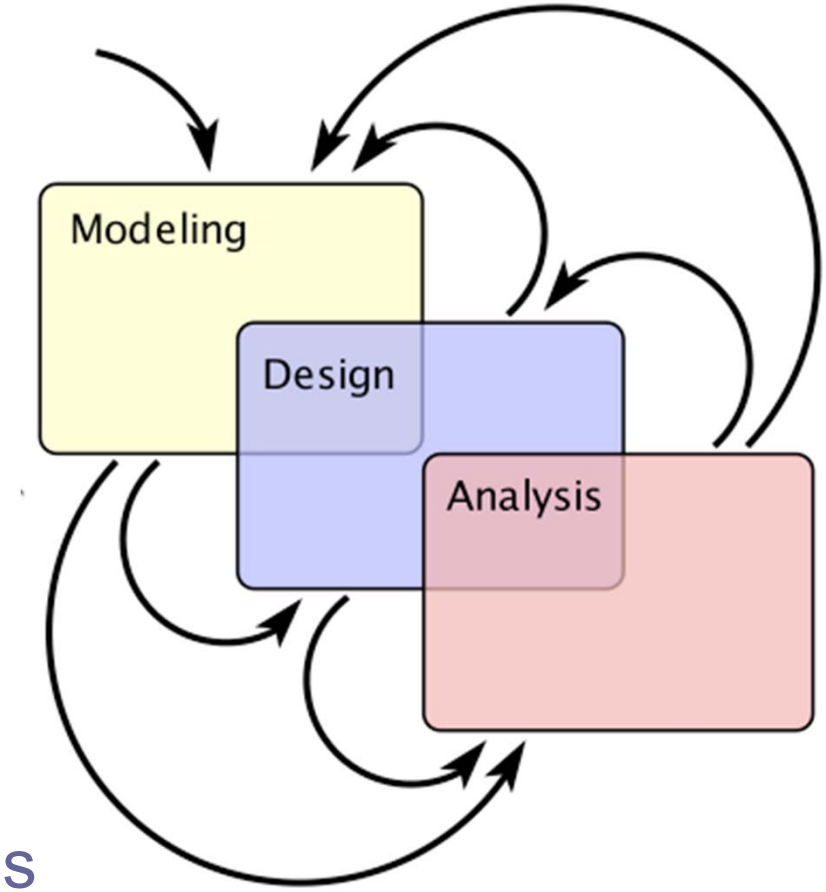


- Source: http://www.cvel.clemson.edu/auto/systems/auto-systems.html (image items are clickable)

This course

- How can we design complex systems?
- *Fundamental Algorithms for System Modeling, Analysis, and Optimization*
- 3 elements:
  - Systems
    - System-oriented view: look at the system as a whole, then focus on its parts
  - Algorithms
    - Focus on computer-aided design techniques
  - Modeling, Analysis, and Optimization
    - Focus on model-based design

Modeling, Design, Analysis



Modeling is the process of gaining a deeper understanding of a system through imitation. Models specify **what** a system does.

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

Analysis is the process of gaining a deeper understanding of a system through dissection. It helps understand **why** a system does what it does (or fails to do what it should do). It can be applied both to real systems, but also to models, to gain understanding about system properties before actually building the system.

INTRODUCTION:

SYSTEMS

What is a system?

- EVERYTHING !
- A more useful definition?

System: definition

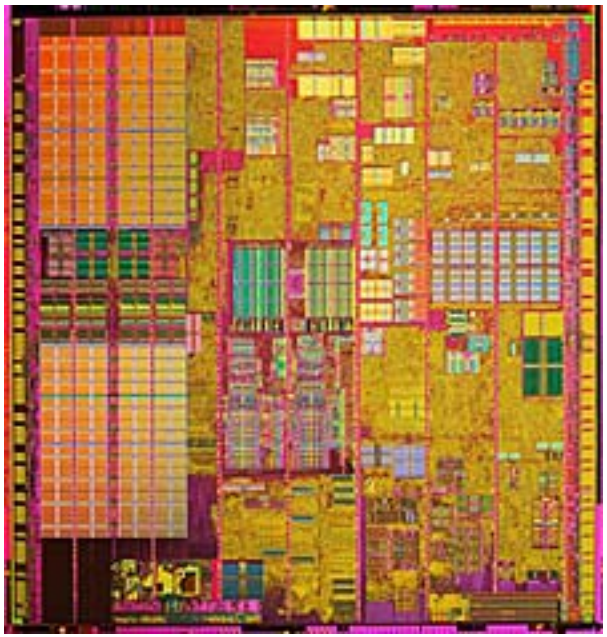
- Something that has:
  - State
  - Dynamics: rules that govern the evolution of the state in time

System: definition

- Something that has:
  - **State**
  - Dynamics: rules that govern the evolution of the state in **time**
- It may also have:
  - Inputs: they influence how system evolves
  - Outputs: this is what we observe

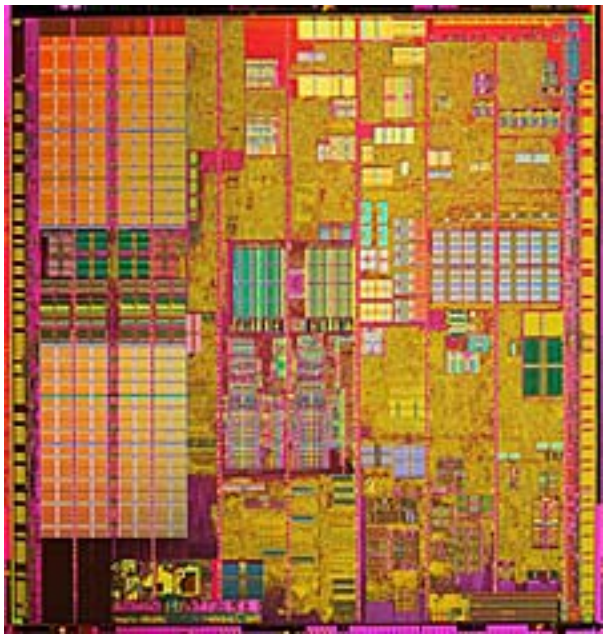
Example: digital circuits

- Digital circuit:
  - State: ???
  - Dynamics: ???



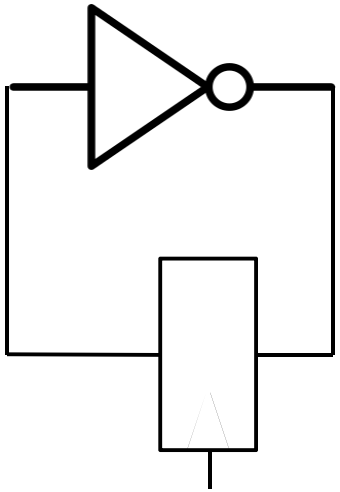
Example: digital circuits

- Digital circuit:
  - State: value of every register, memory element
  - Dynamics:
    - Defined by the combinational part (logical gates)
    - Time: discrete, or “logical” (ticks of the clock)

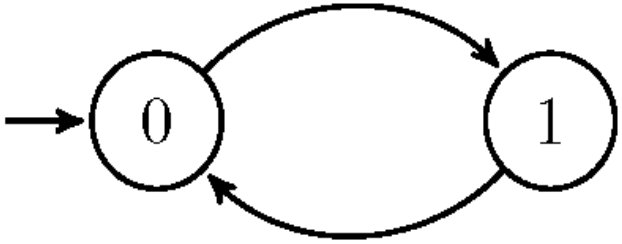


Example: digital circuits

- **Systems** vs. **models**



System
(the “real” circuit)

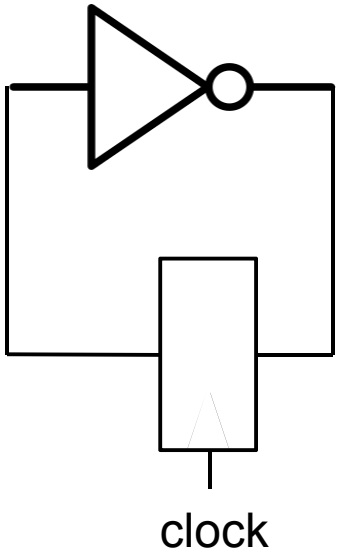


Model
(a finite-state machine)

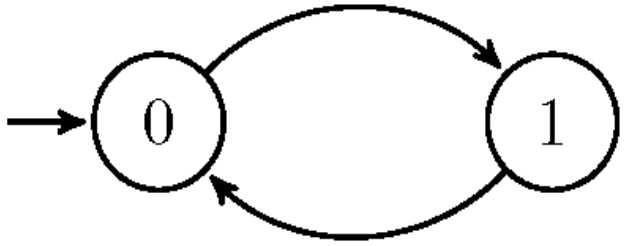
To reason about systems (analyze, make predictions, prove things, ...), we need mathematical models

Example: digital circuits

- **Systems** vs. **models**



System
(the “real” circuit)



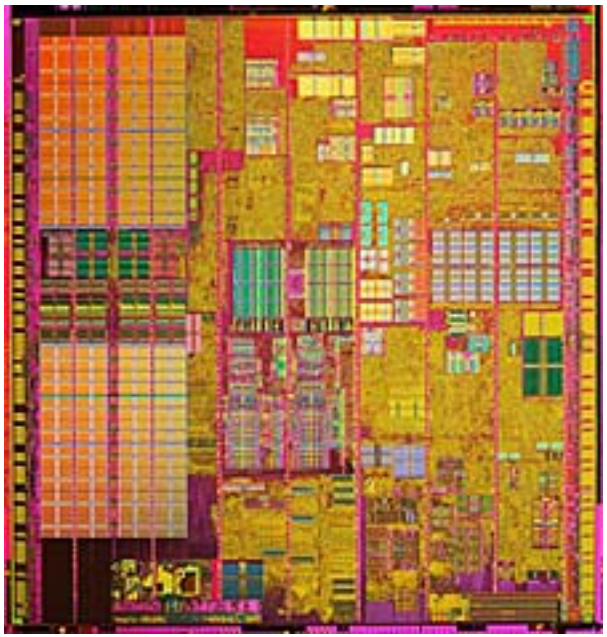

```
node Circuit ()
returns (Output: bool);
let
  Output = false -> not pre Output;
tel
```

Different models
(finite-state machines)

Different representations (languages, syntaxes)
of the same underlying mathematical model

Example: digital circuits

- Digital circuit as a system:
  - State: value of every register, memory element
  - Dynamics:
    - Defined by the combinational part (logical gates)
    - Time: discrete, or “logical” (ticks of the clock)
  - **Or**:
  - State: all currents and voltages at all transistors at a given time t
  - Dynamics: physics of electronic circuits (differential algebraic equations)

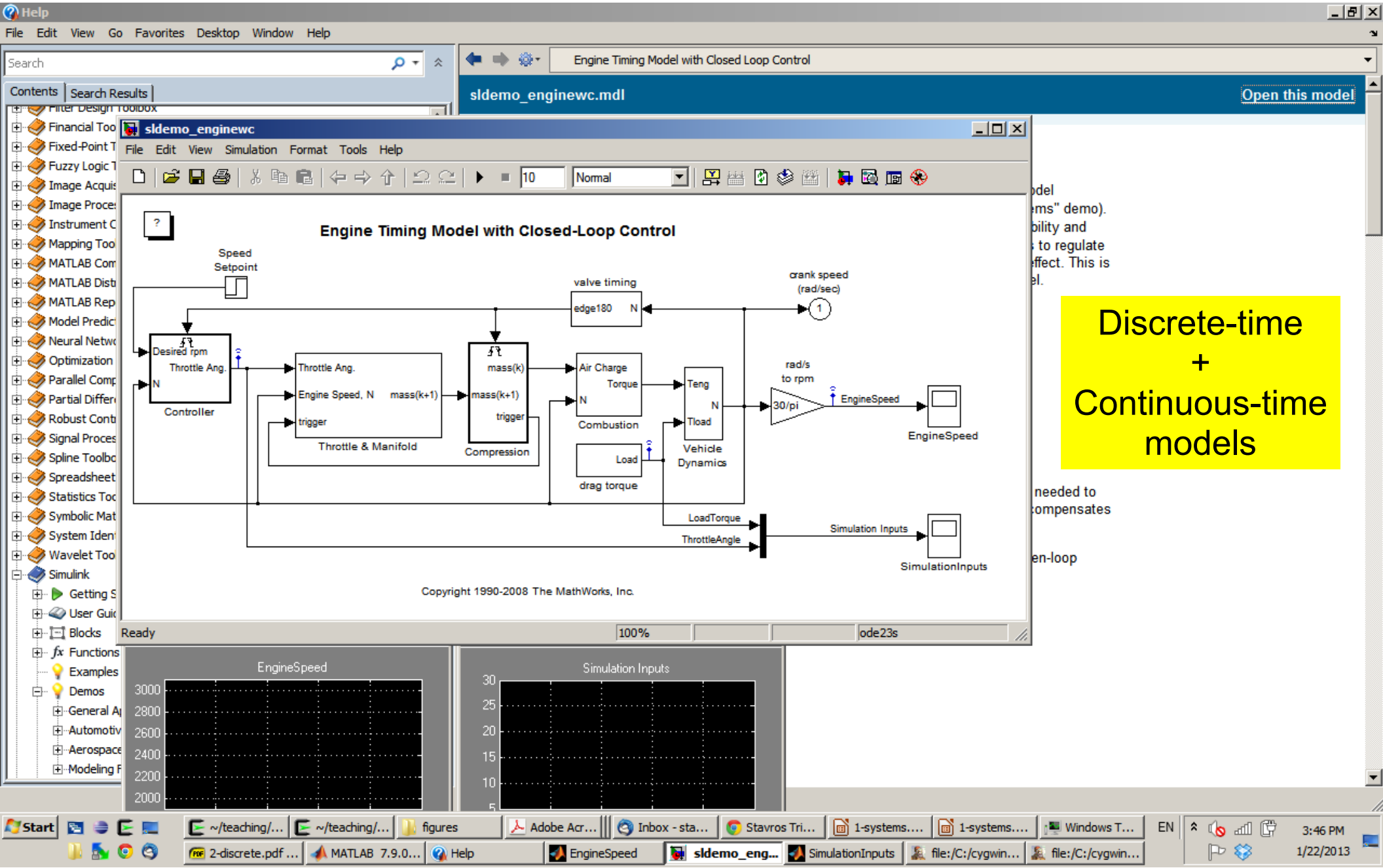


Different levels of **abstraction**

Multi-paradigm modeling

- Different representations (languages, syntaxes) of the same underlying formalism.
- Different modeling formalisms often needed to describe the same system, e.g., at different levels of abstraction.
- Different modeling formalisms often needed to describe different parts of the system (subsystems).

Example: plant + controller



Classes of systems/models considered in this course

- Discrete: state machines, …
- Continuous: differential equations, …
- Timed: discrete-event, timed automata, …
- Dataflow: process networks, SDF, …
- Probabilistic: Markov chains, …

Back to the definition of “system”

- Many kinds of systems:
  - Software = many classes, objects, threads, …
  - Car = chassis + engine + computer + software + ...
  - Human body = heart + lungs + … = many cells = ...
  - Weather
  - Stock market
  - Internet
- How to describe each of these as states + dynamics?

Difficult (impossible) to describe some systems using our current definition

System: monolithic definition

- Something that has:
  - State
  - Dynamics: rules that govern the evolution of the state in time
- It may also have:
  - Inputs: they influence how system evolves
  - Outputs: this is what we observe

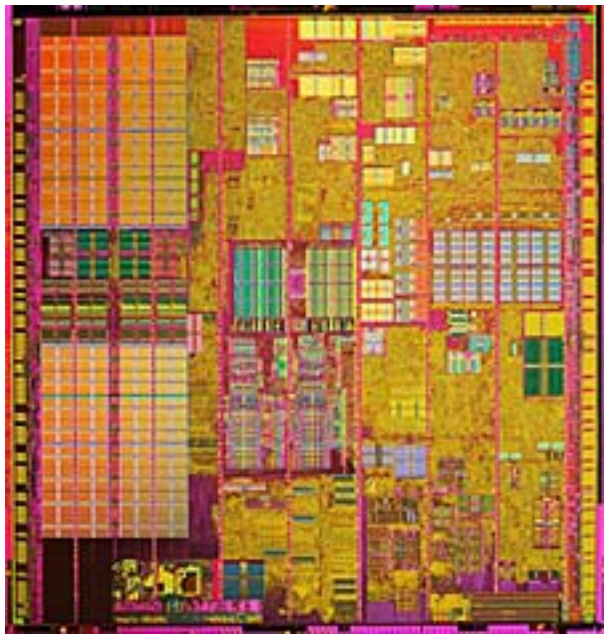
System: compositional definition

- A collection of subsystems that interact

- So, we must describe:
  - Each subsystem (recursive definition)
  - The interaction rules

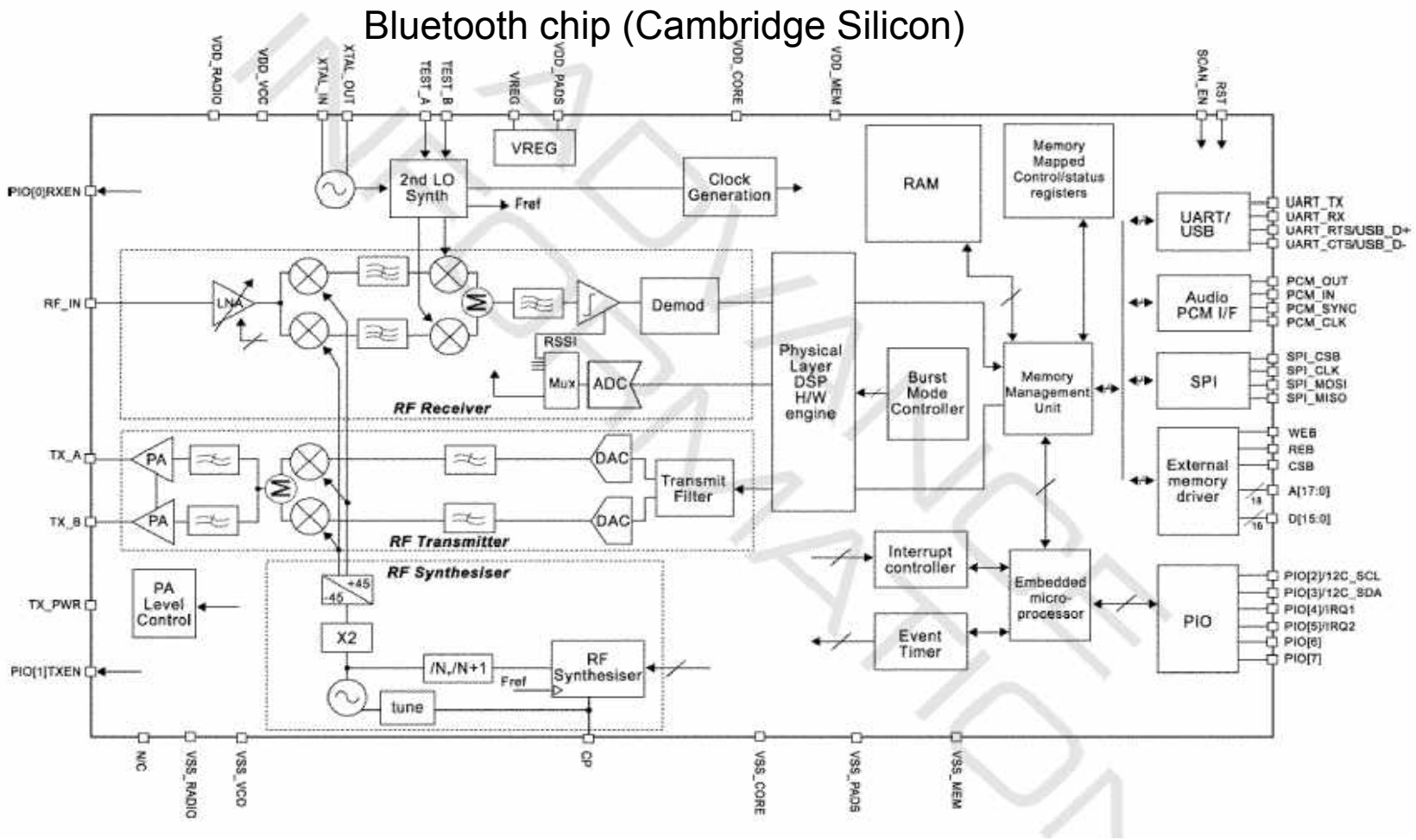
Example: digital circuits

- Subsystems: latches, gates, …
- All governed by the same clock
- Synchronous interaction



Example

Bluetooth chip (Cambridge Silicon)

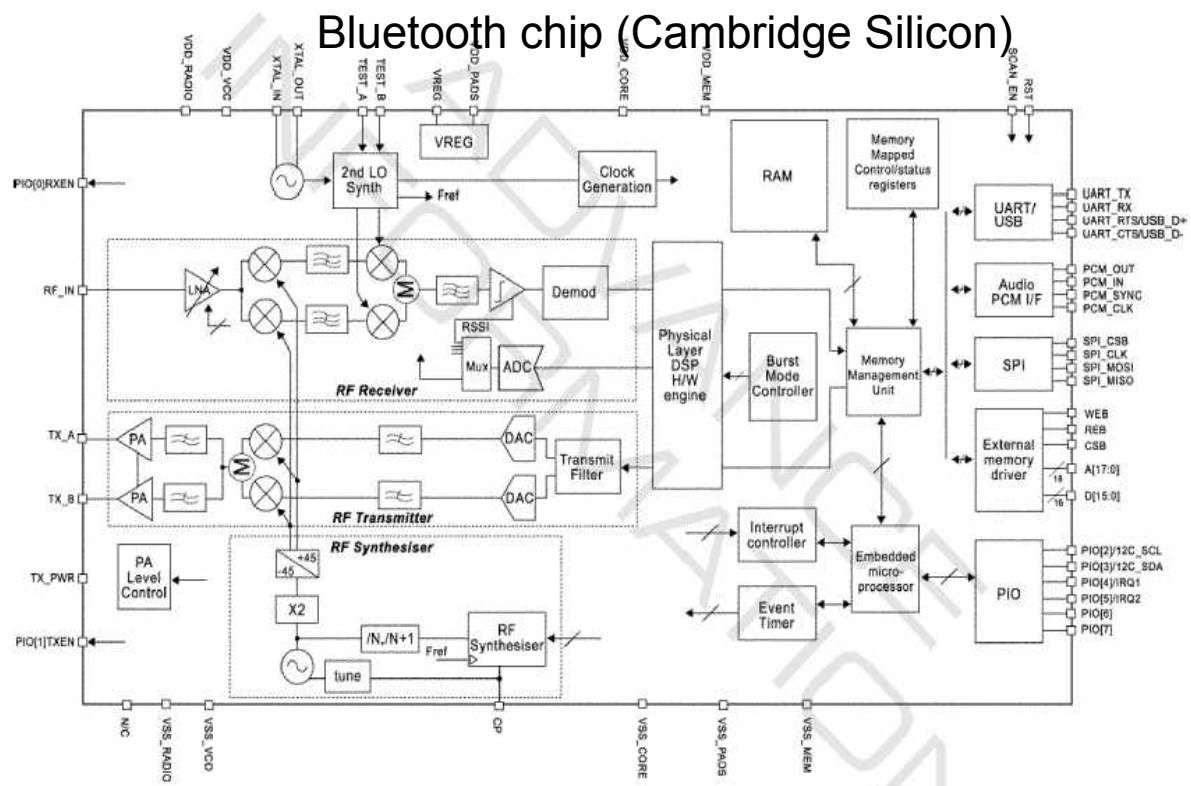


Courtesy of J. Roychowdhury, UC Berkeley

Example

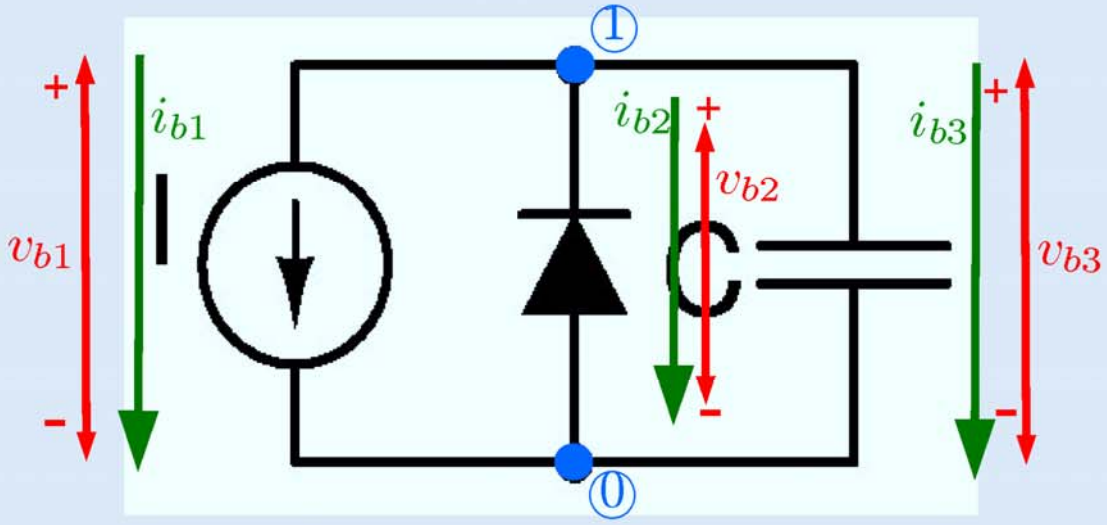
- Subsystems: ADC, DAC, microprocessor, …, wires
- Interaction rules (partial list):
  - Kirchoff's laws:
    - At every node of the circuit: sum(all currents) = 0

Bluetooth chip (Cambridge Silicon)



Courtesy of J. Roychowdhury, UC Berkeley

Diode-Capacitor (Nonlinear) Example



- 1 KCL eqn. at node 1: $i_{b1} + i_{b2} + i_{b3} = 0$
- 3 KVL equations: $v_{b1} = v_{b2} = v_{b3} = e_1$
- 3 BCRs: $i_{b1} = I(t); \quad i_{b2} = -\text{diode}(-v_{b2}; I_S, V_t); \quad i_{b3} = C\dfrac{dv_{b3}}{dt}$
- 7 unks, 7 eqns (incl. 1 differential, 1 nonlinear)
- eliminate by hand: ODEs
  - → **(ODEs not always possible)**
  - → **analytical soln not available!**
    - **numerical methods needed**

$$I(t) - \text{diode}(-e_1; I_S, V_t) + C\frac{de_1}{dt} = 0$$

$$\Rightarrow \boxed{\frac{de_1}{dt} = \frac{1}{C}\text{diode}(-e_1; I_S, V_t) - \frac{I(t)}{C}}$$

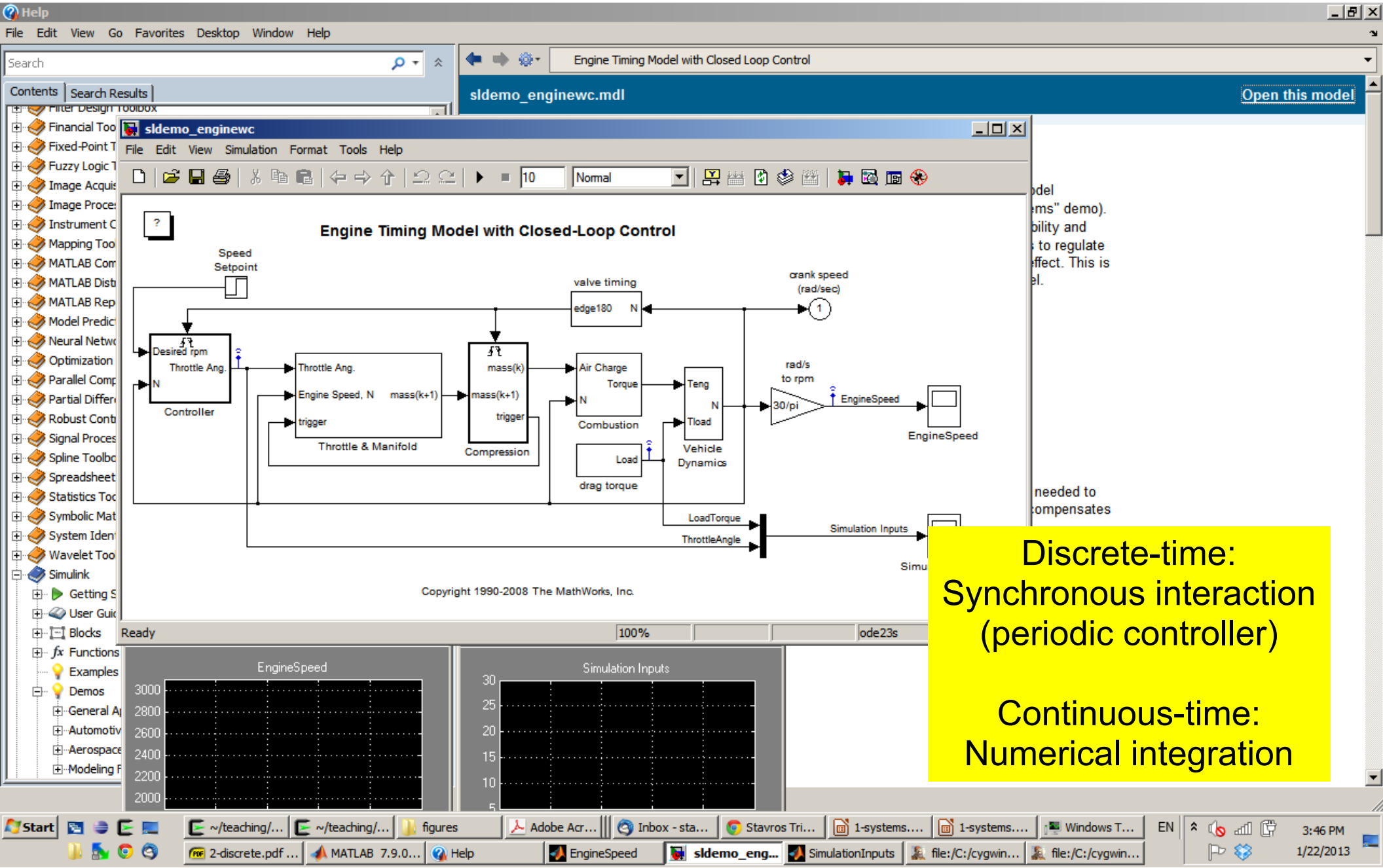
Example: concurrent software

- Multi-thread Java program:

Asynchronous interaction (*interleaving*)

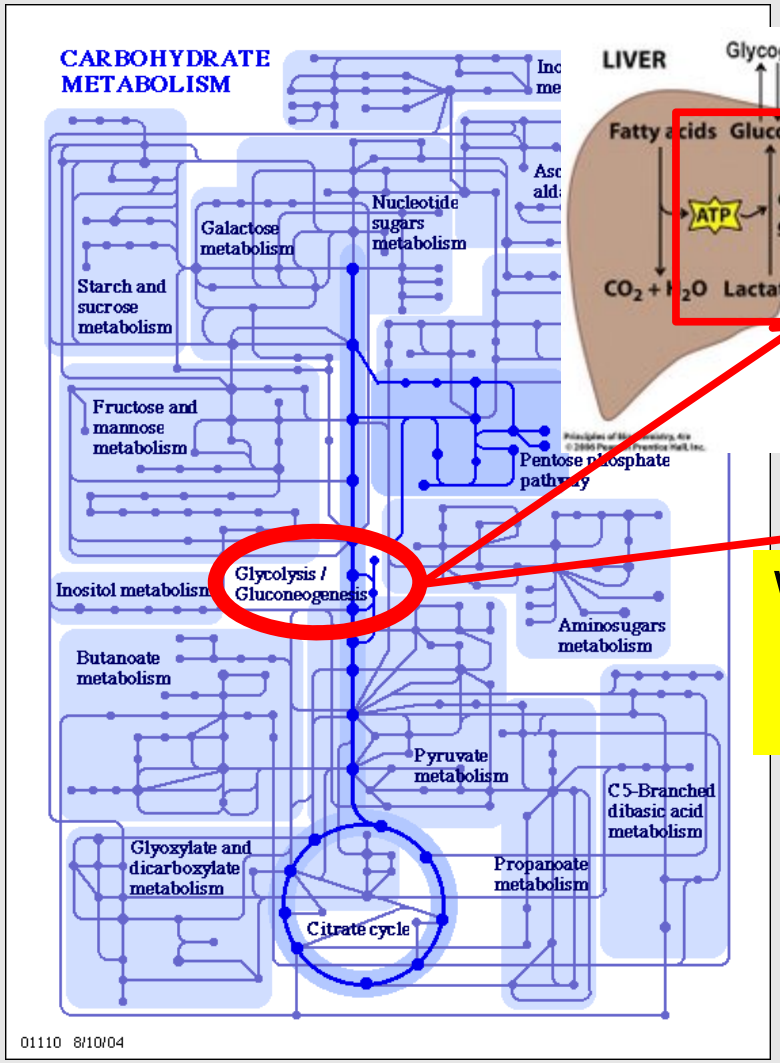
```
public class Deadlock {
    static class Friend {
        private final String name;
        public Friend(String name) {
            this.name = name;
        }
        public String getName() {
            return this.name;
        }
        public synchronized void bow(Friend bower) {
            System.out.format("%s: %s"
                + "  has bowed to me!%n",
                this.name, bower.getName());
            bower.bowBack(this);
        }
        public synchronized void bowBack(Friend bower) {
            System.out.format("%s: %s"
                + " has bowed back to me!%n",
                this.name, bower.getName());
        }
    }

    public static void main(String[] args) {
        final Friend alphonse =
            new Friend("Alphonse");
        final Friend gaston =
            new Friend("Gaston");
        new Thread(new Runnable() {
            public void run() { alphonse.bow(gaston); }
        }).start();
        new Thread(new Runnable() {
            public void run() { gaston.bow(alphonse); }
        }).start();
    }
}
```

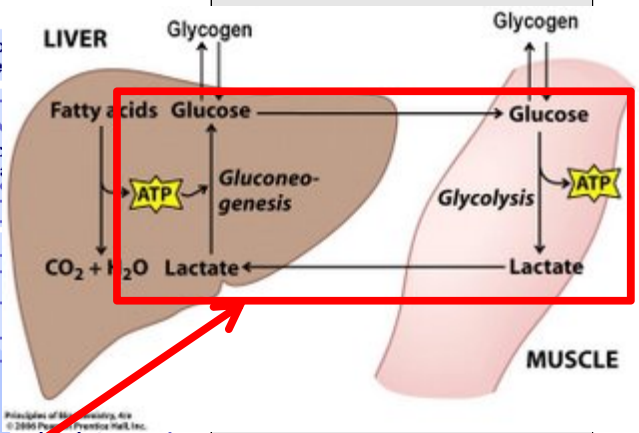
Example: plant + controller



Biochemical Systems

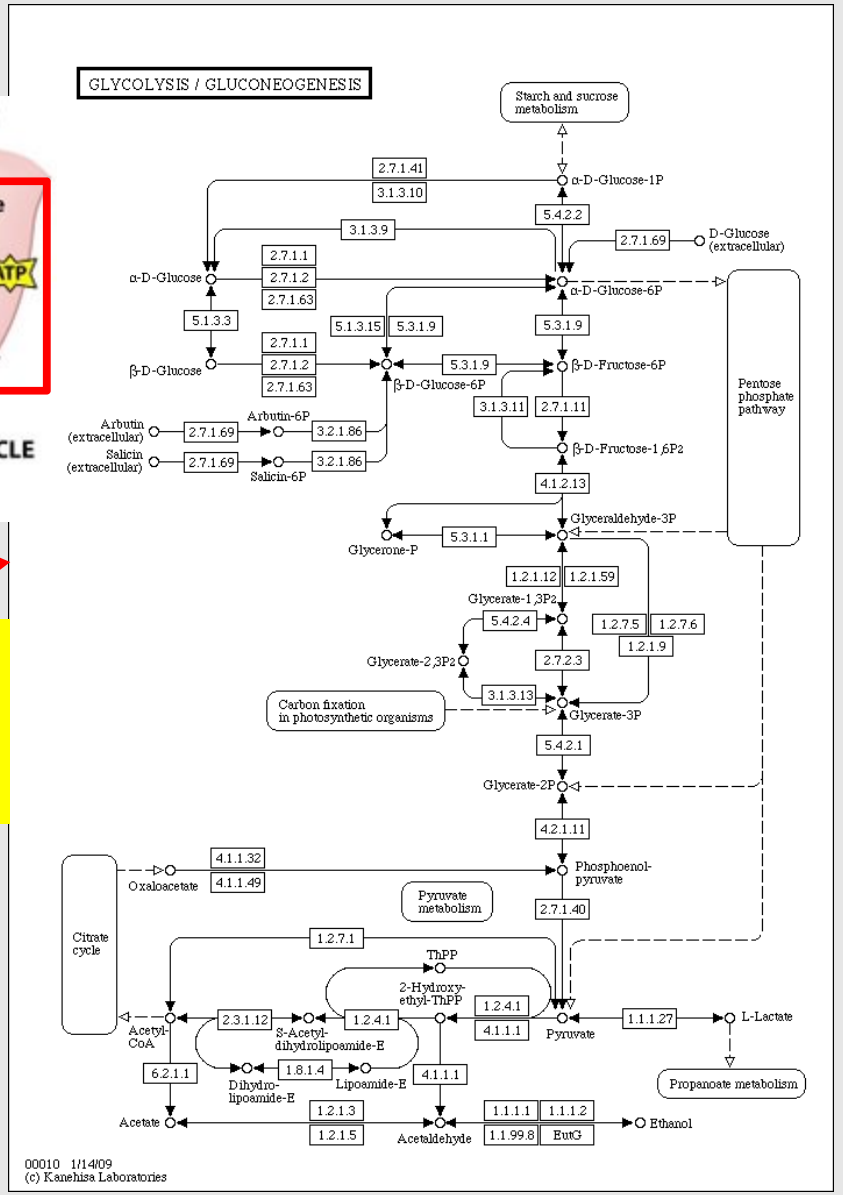
- Reactions governing conversion of glucose to ATP (energy) and back





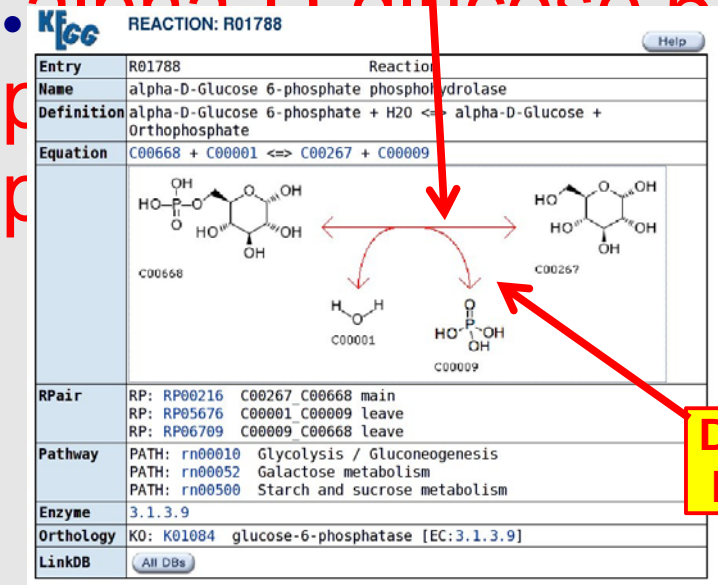
What do these diagrams mean?

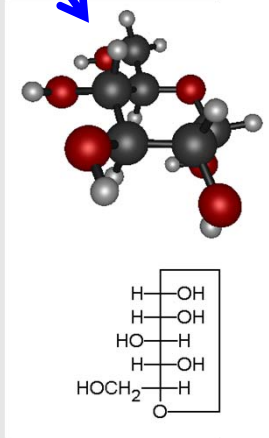
metabolic network

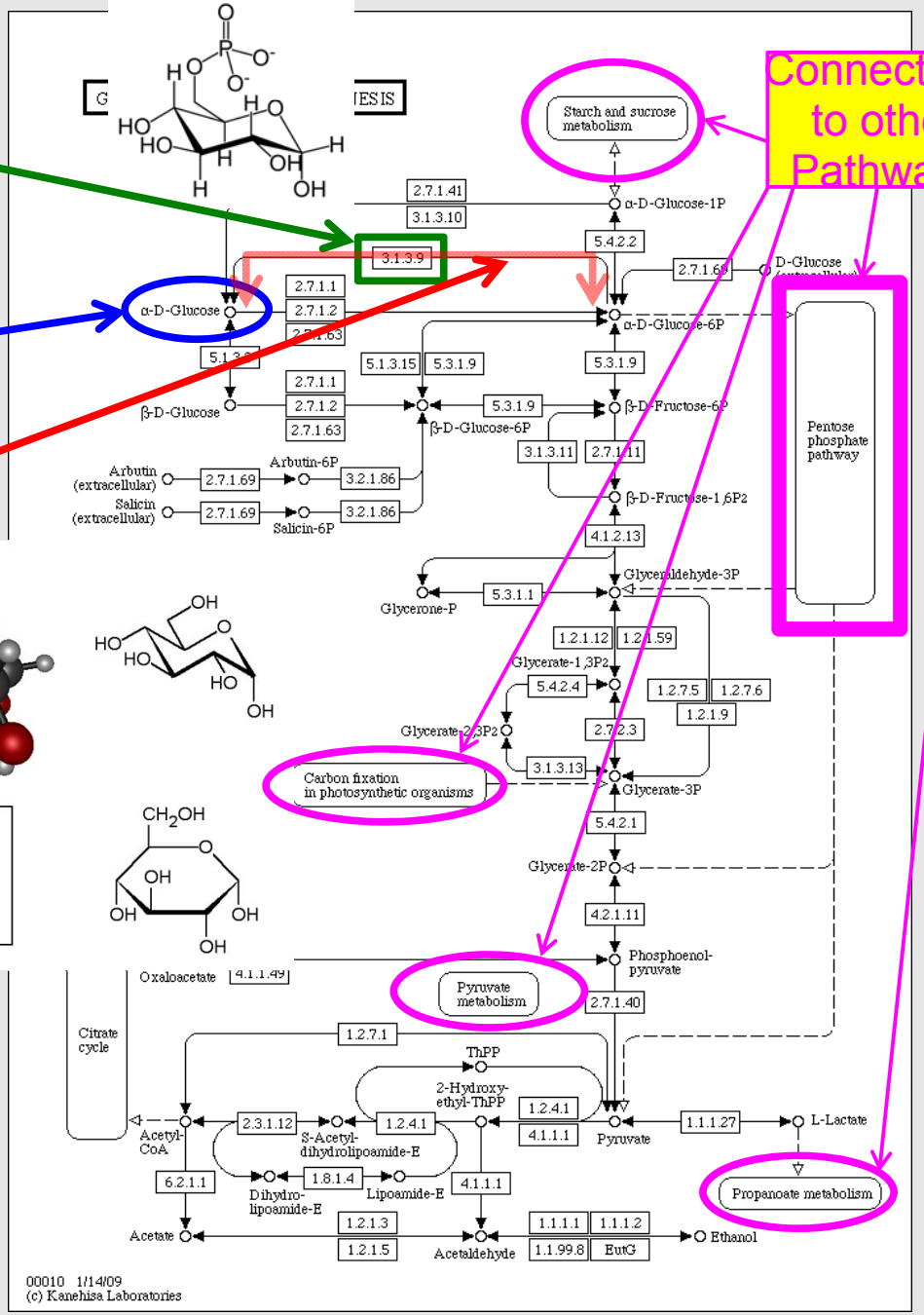


Pathway: Chain of Reactions

- **Enzyme** (protein)
  - glucose-6-phosphatase
- catalyzes reaction
- **Compound**
  - alpha-D-Glucose
- **Reaction**
  - alpha-D-glucose 6-
  - p
  - p

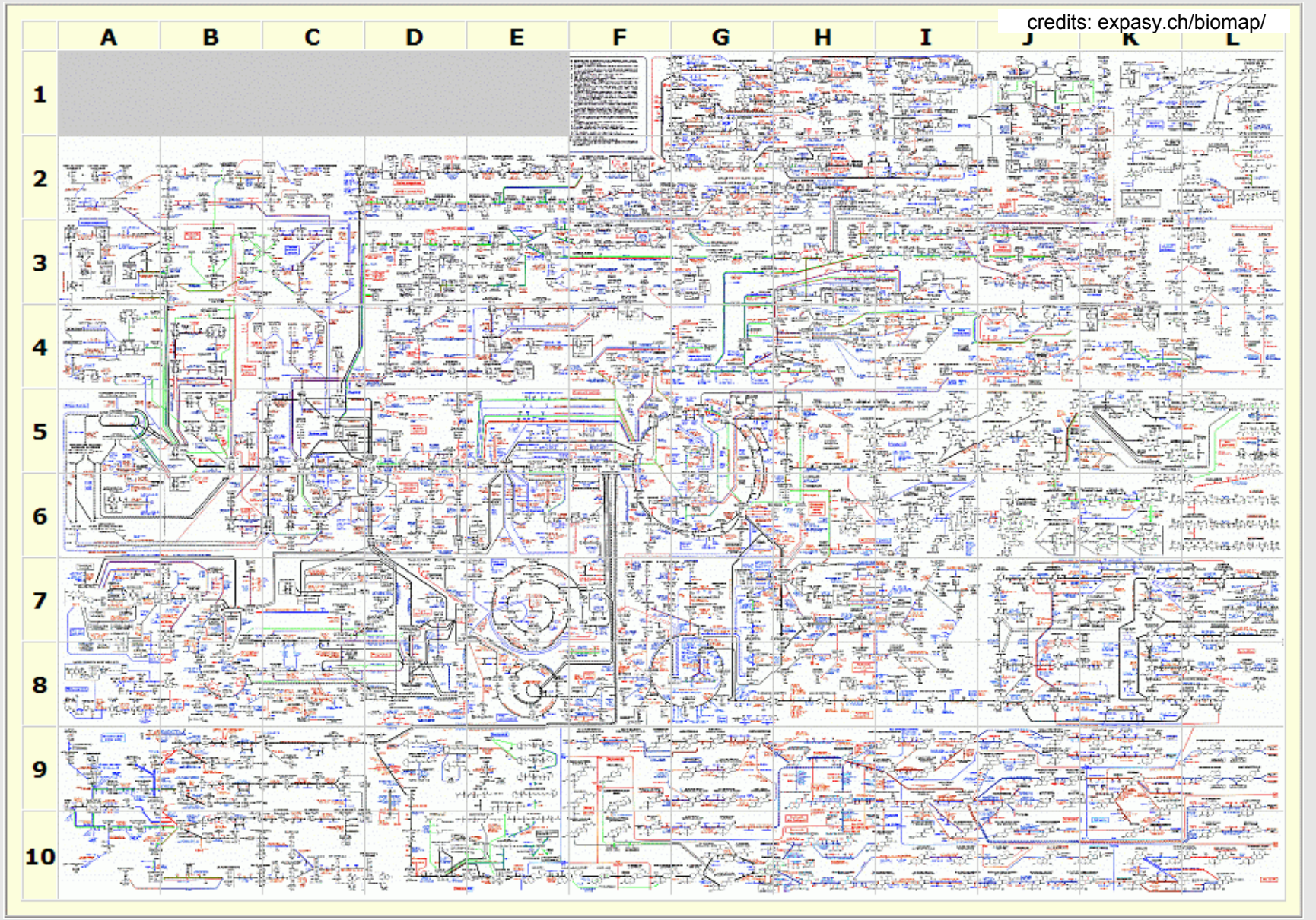






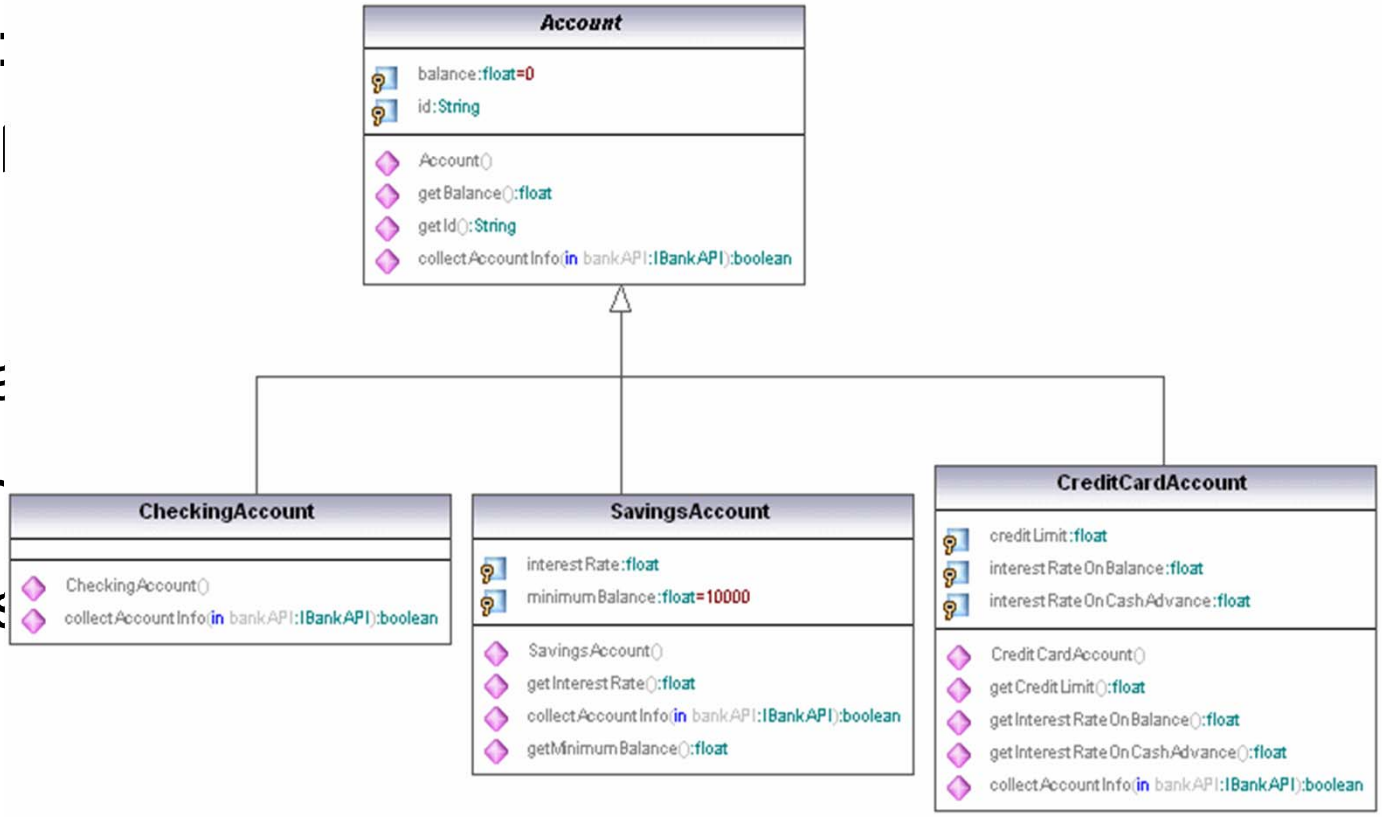
Differential Equations

Metabolic Pathway Maps



Systems: structure + behavior

- Structure:
  - What the system is made of, its parts, sub-systems, …
  - Some modeling languages focus on structure: e.g., UML class diagrams
- Behavior:
  - What t
- The two a
- This cour
- But also s es for synch



Stavros Tripakis

Adjunct Assoc. Prof. and Researcher

UC Berkeley

- Past:
  - Undergrad: University of Crete, Greece, 1992
  - PhD: Verimag Laboratory, Grenoble, France, 1998
  - Postdoc: Berkeley, 1999 – 2001
  - Research Scientist: CNRS, Verimag, France, 2001 – 2006
  - Research Scientist: Cadence Design Systems, Berkeley, 2006 -- 2008

Looking for 2 phd students funded by NSF CPS project!

- Current research interests
  - Rigorous system design, cyber-physical systems (CPS)
  - System modeling, verification, and synthesis
  - Compositionality and interfaces

Sanjit A. Seshia

Associate Professor, EECS
B.Tech, IIT Bombay
M.S. & Ph.D., Carnegie Mellon Univ.

Research theme: Formal Methods
Algorithms + Modeling for Dependable Computing

Theory	+	**Practice**
Algorithms (constraint solving, optimization, learning), Logic & Automata Theory (model checking, computational logic)		CAD for VLSI & Synthetic Biology, Computer Security, Embedded / Cyber-Physical Systems

Examples:

- UCLID: One of the first SMT solvers and SMT-based verifiers
- GameTime: Timing analysis of embedded software by game-theoretic online learning

Round of introductions

Your name, research/professional interests, grad/undergrad student, ...

Quiz

- Express the following in your favorite mathematical formalism:
  - You can fool some people sometimes
  - You can fool some of the people all of the time
  - You can fool some people sometimes but you can't fool all the people all the time [Bob Marley]
  - You can fool some of the people all of the time, and all of the people some of the time, but you can not fool all of the people all of the time [Abraham Lincoln]

This course

- How can we design complex systems?
- *Fundamental Algorithms for System Modeling, Analysis, and Optimization*
- 3 elements:
  - Systems
    - System-oriented view: look at the system as a whole, then focus on its parts
  - Algorithms
    - Focus on computer-aided design techniques
  - Modeling, Analysis, and Optimization
    - Focus on model-based design

Topics we will study in this course

Design Flows (very briefly, next lecture)

- Methodology of going from a model to implementation

Fundamental, Cross-Cutting Algorithms

- Timing analysis (graph algos.)
- Retiming (graph + constraint solving, optimization)

Algorithms for Discrete Models

- Boolean algebra, Boolean function representation and manipulation (circuit representations, Binary Decision Diagrams)
- Optimizing Boolean representations (logic synthesis & optimization)
- Logic: propositional logic, Satisfiability(SAT) solving, temporal logic
- Observability and Controllability: fault localization and testing
- Automata/Transition systems: representation, reachability analysis, model checking, game solving, …
- Symbolic Execution

Topics we will study in this course

Algorithms for Continuous Models

- Continuous system simulation
- Solving differential (algebraic) equations

Algorithms for Cyber-Physical Models

<everything above, plus…>

- Synchronous systems, fixed-point computations
- Scheduling and performance analysis for dataflow systems
- Discrete-event simulation for timed systems
- Simulation/analysis of hybrid systems
- Stochastic systems: probabilistic analyses
- Controller synthesis

From Algorithms to Software

- How to write well-structured, efficient code

CAD at Berkeley: History

- CAD research at Berkeley: design tools with an impact

late 60s and 70s

CANCER, SPICE (Rohrer, Nagel, Cohen, Pederson, ASV, Newton, etc.)

SPLICE (Newton)

80s

MAGIC (Ousterhout et. al.)

Espresso (Brayton, ASV, Rudell, Wang et. al.)

MIS (Brayton, ASV, et. al.)

90s

SIS, HSIS (Brayton, ASV et. al.)

VIS (Brayton, ASV, Somenzi et. al.)

Ptolemy (Lee et. al.)

2000-date

MVSIS (Brayton, Mishchenko et. al.)

BALM (Mishchenko, Brayton et. al.)

ABC (Mishchenko, Brayton et. al.)

MetroPolis, Metro II, Clotho (ASV et. al.)

Ptolemy II, HyVisual (Lee et. al.)

UCLID, GameTime, Beaver (Seshia et. al.)

Course Logistics

Webpage, Books, etc.

The course webpage is the definitive source of information

http://embedded.eecs.berkeley.edu/eecsx44/

We'll also use bSpace

No textbook. Readings will be posted / handed out for each set of lectures.

Some references will be placed on reserve in Engineering library.

GSI: Mike Zimmer

See webpage for office hours, etc.

Format of Lectures

2 1.5 hour lectures per week (TuTh 3:30 – 5 pm)

1 hr Discussion section / lecture each Fri 10-11

- usually a topic supporting homeworks/projects; sometimes an extension of lectures

Grading

2 Midterms (20% each)

8-10 homework assignments (total ~ 25%)

1 course project (~ 35%)

Course project:

- ❑ Graduate students (244): Must investigate a novel research idea
- ❑ Undergraduates (144): Encouraged to do 244-style project (join with grads!); also permitted to do implementation projects or literature surveys

This course

Modeling classes	Discrete	Continuous	Dataflow	Timed	Stochastic
Analysis techniques					
Composition					

EECS 144/244
Fundamental Algorithms for System Modeling, Analysis, and Optimization

Lecture 1: Introduction, Systems

Sanjit Seshia, Stavros Tripakis

UC Berkeley

Fall 2013