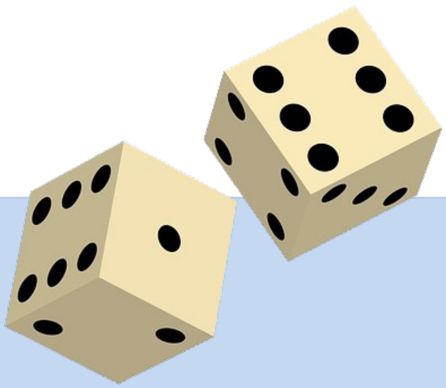
iCyPhy

Software Design for Cyber-Physical Systems

Edward A. Lee

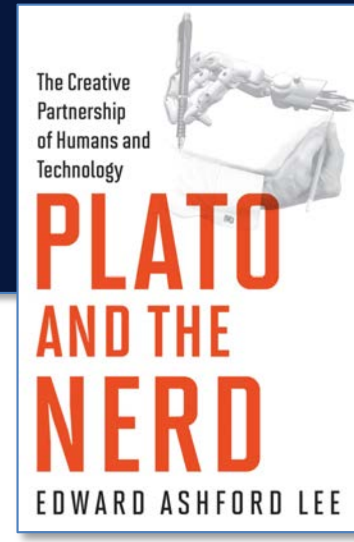


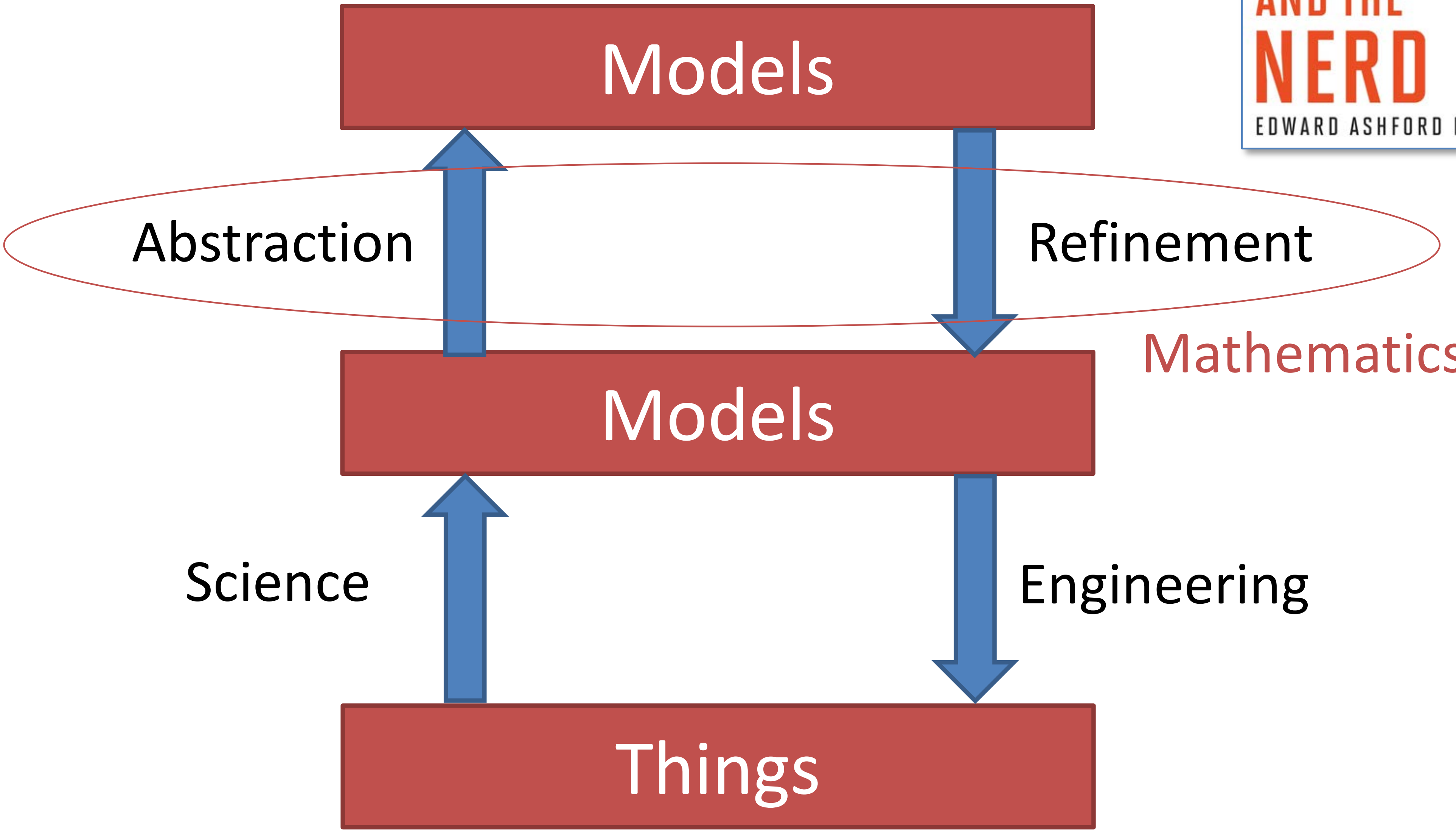
Module 11: Models Revisited

Technical University of Vienna
Vienna, Austria, May 2022

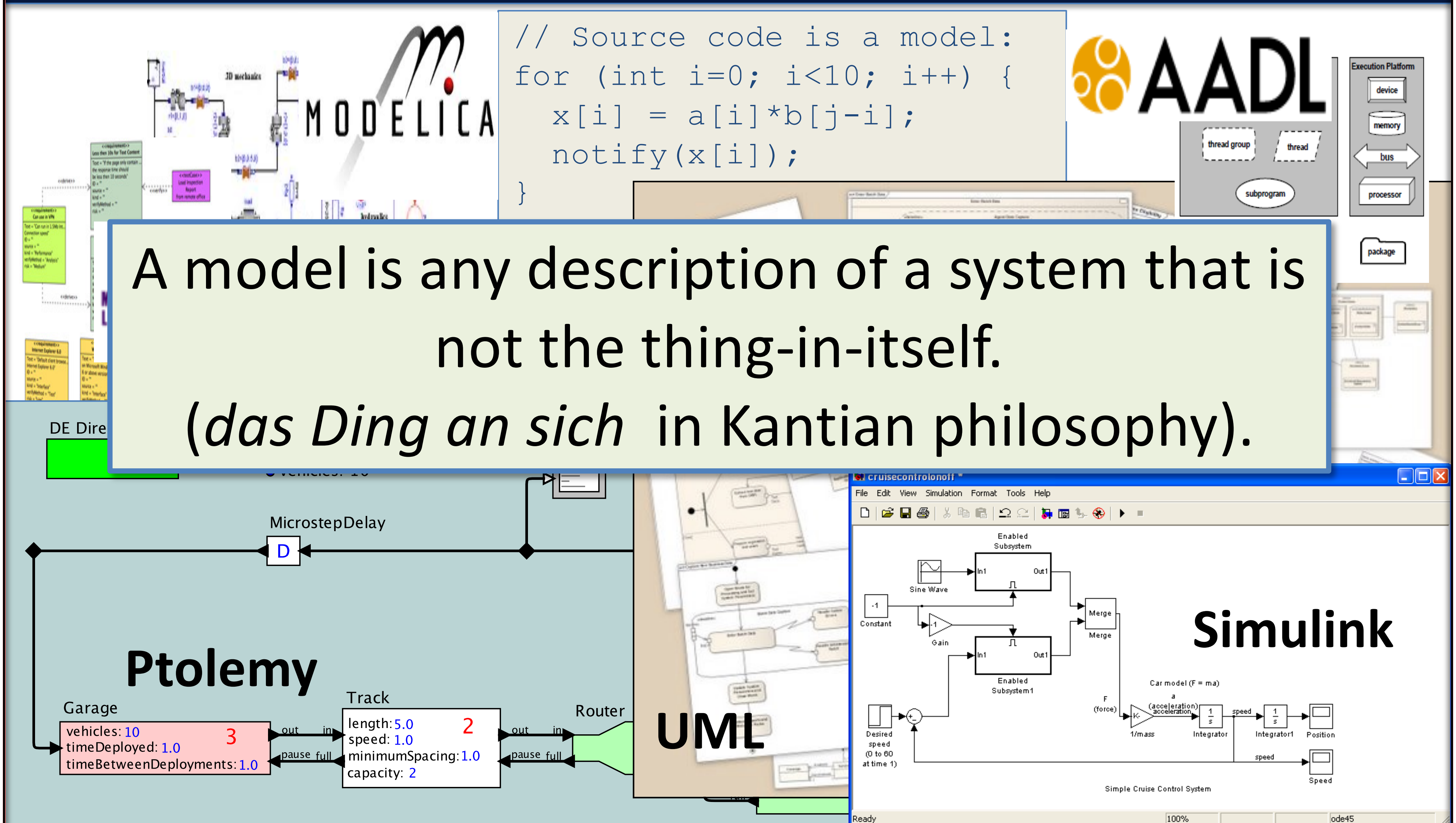
University of California, Berkeley

Science, Engineering, and Mathematics





What is a model?



A model is any description of a system that is not the thing-in-itself.

(*das Ding an sich* in Kantian philosophy).

Quiz: Is this a Model?

```
void foo(int32_t x) {
        if (x > 1000) {
                x = 1000;
        }
        if (x > 0) {
                x = x + 1000;
                if (x < 0) {
                        panic();
                }
        }
}
```

The physical system has many properties not represented in the model (e.g. timing, temperature, physical volume).


Ceci n'est pas une pipe.
Magritte

Purposes of Models

- Describe structure, weight, dimensions, …
- Describe energy needs, temperatures, …
- Describe dynamics
- Specify a design
- Simulate a behavior
- Verify that conformance with a requirement
- Specify a requirement
- …

Properties of Models

Formal?

A *formal model* is a model given in a well-defined, machine-readable syntax and can be mechanistically manipulated using well-defined rules to derive properties of the model.

Executable?

An *executable model* is a formal model describing the dynamic behavior of a system where a machine can use the model to simulate that dynamic behavior.

Faithful?

A *faithful model* is a model that reasonably accurately conforms to properties of the thing being modeled.

Determinism as a Property of Models

A **model** is *deterministic* if, given the initial *state* and the *inputs*, the model defines exactly one *behavior*.

Determinism in Physics: Laplace's Demon

Pierre Simon Laplace

Pierre-Simon Laplace (1749–1827).
Portrait by Joan-Baptiste Paulin Guérin, 1838

Laplace's Demon cannot exist.

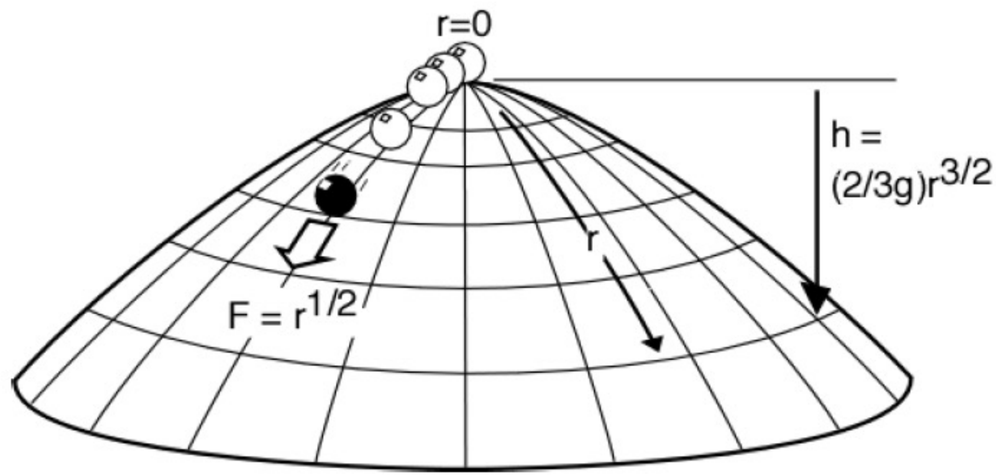
In 2008, David Wolpert proved that Laplace's demon cannot exist.

His proof relies on the observation that such a demon, were it to exist, would have to exist in the very physical world that it predicts.

David Wolpert

Norton's Hill

Even without quantum physics, Newtonian physics is not deterministic.



Metastable system that obeys all of Newton's laws but is nondeterministic.

Norton, J. D. (2007). Causation as Folk Science. *In Causation, Physics, and the Constitution of Reality*, Oxford, Clarendon Press.

Did quantum physics dash this hope?

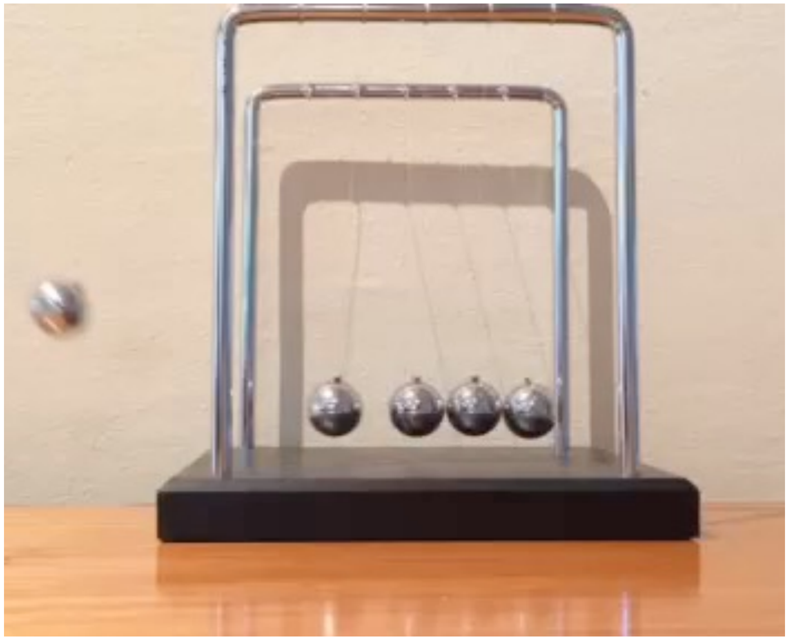
"At first, it seemed that these hopes for a complete determinism would be dashed by the discovery early in the 20th century that events like the decay of radioactive atoms seemed to take place at random. It was as if God was playing dice, in Einstein's phrase. But science snatched victory from the jaws of defeat by moving the goal posts and redefining what is meant by a complete knowledge of the universe."

(Stephen Hawking, 2002)

Determinism as a property of models, not things

$$x(t) = x(0) + \int_0^t v(\tau)d\tau$$
$$v(t) = v(0) + \frac{1}{m}\int_0^t F(\tau)d\tau,$$

Deterministic model



Deterministic system?

Determinism as a Property of Models

A **model** is *deterministic* if, given the initial *state* and the *inputs*, the model defines exactly one *behavior*.

Our most valuable models are *deterministic*.

Software as a Model

```c
void foo(int32_t x) {
        if (x > 1000) {
                x = 1000;
        }
        if (x > 0) {
                x = x + 1000;
                if (x < 0) {
                        panic();
                }
        }
}
```

This program defines exactly one behavior, given the input x.

The modeling framework defines state, input, and behavior.

The physical system has many properties not represented in the model (e.g. timing, temperature, …).

Architecture as a Model

Physical System

Image: Wikimedia Commons

Model

Integer Register-Register Operations

RISC-V defines several arithmetic R-type operations. All operations read the *rs1* and *rs2* registers as source operands and write the result into register *rd*. The *funct* field selects the type of operation.

31 — 27	26 — 22	21 — 17	16 — 7	6 — 0
rd	rs1	rs2	funct10	opcode
5	5	5	10	7
dest	src1	src2	ADD/SUB/SLT/SLTU	OP
dest	src1	src2	AND/OR/XOR	OP
dest	src1	src2	SLL/SRL/SRA	OP
dest	src1	src2	ADDW/SUBW	OP-32
dest	src1	src2	SLLW/SRLW/SRAW	OP-32

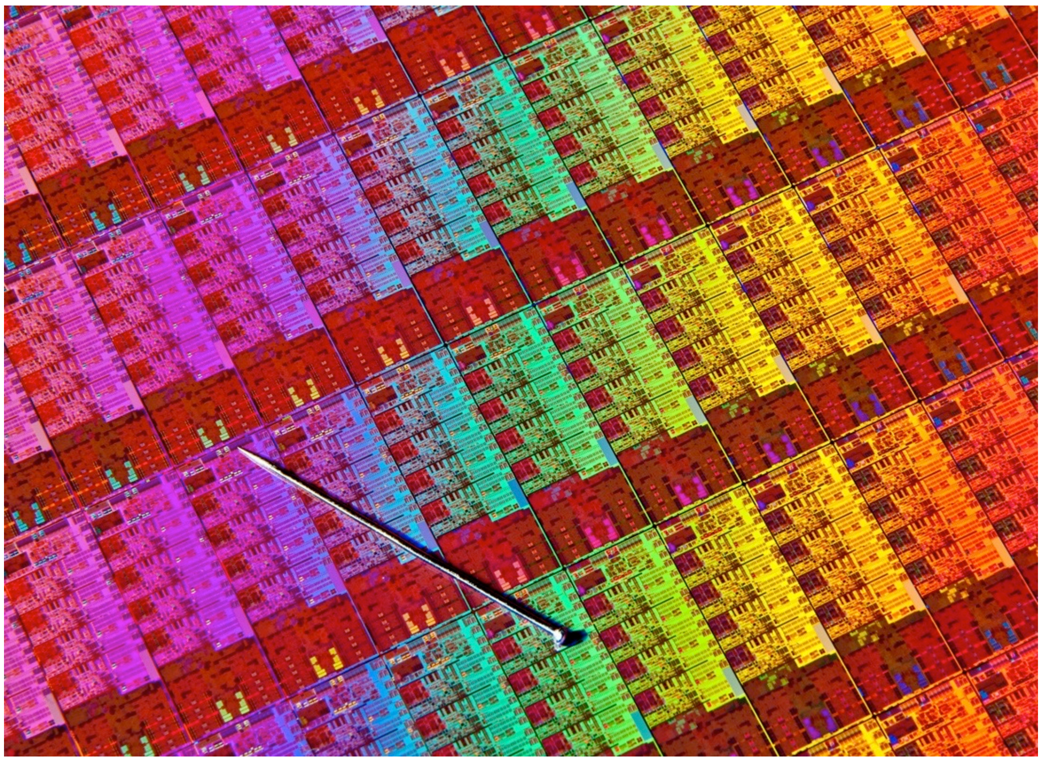
Waterman, et al., The RISC-V Instruction Set Manual, UCB/EECS-2011-62, 2011

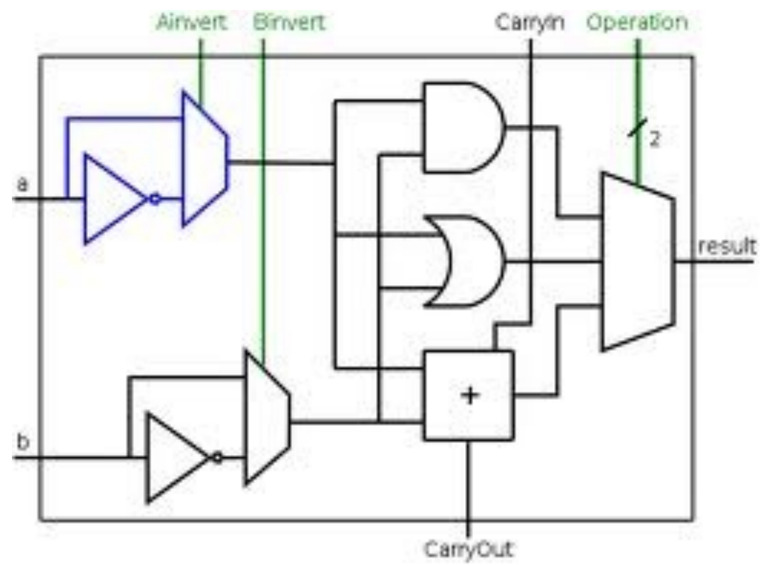
Instruction Set Architectures (ISAs) are deterministic models

Digital Circuits as Models

Physical System

Model





Synchronous digital logic
is a deterministic model

Physical Dynamics as a Model

Physical System

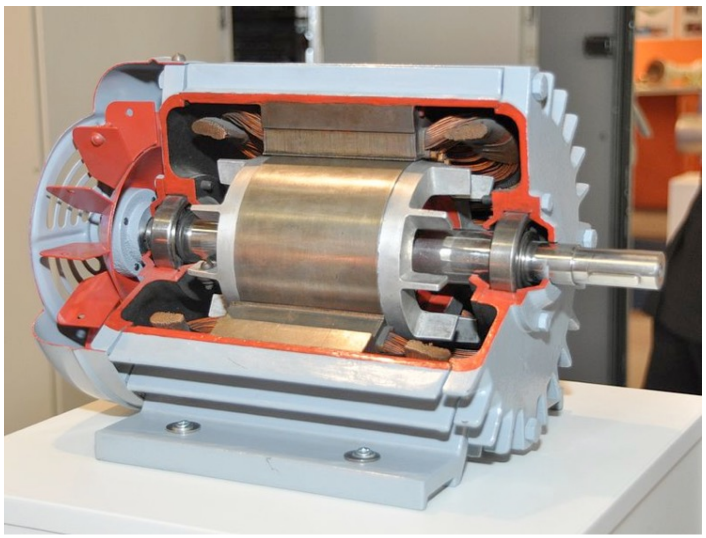


Image: Wikimedia Commons

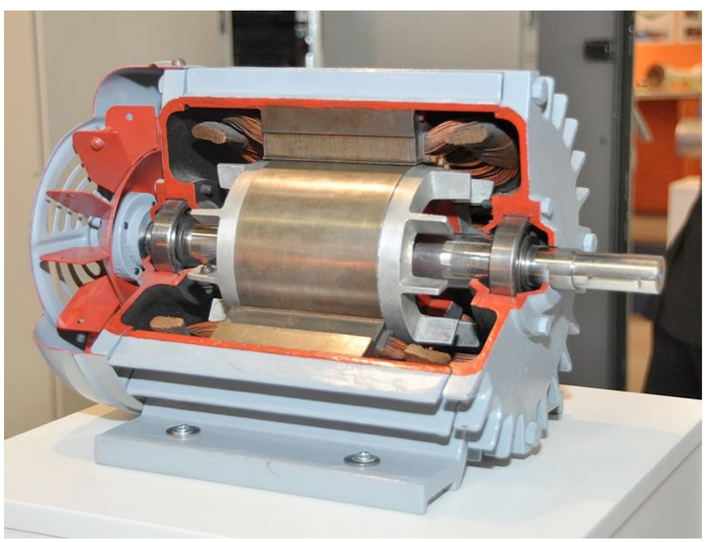
Model

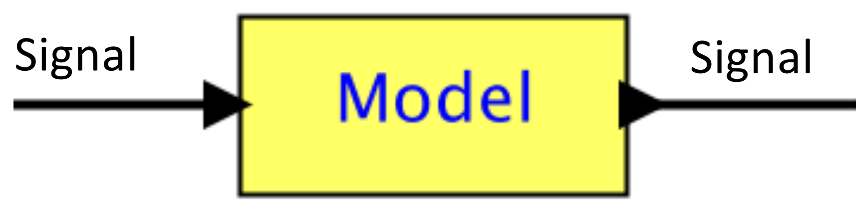
$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_{0}^{t} \mathbf{F}(\tau)d\tau$$

Differential Equations
are deterministic models

CPS combinations of deterministic models are nondeterministic

```
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
}
```





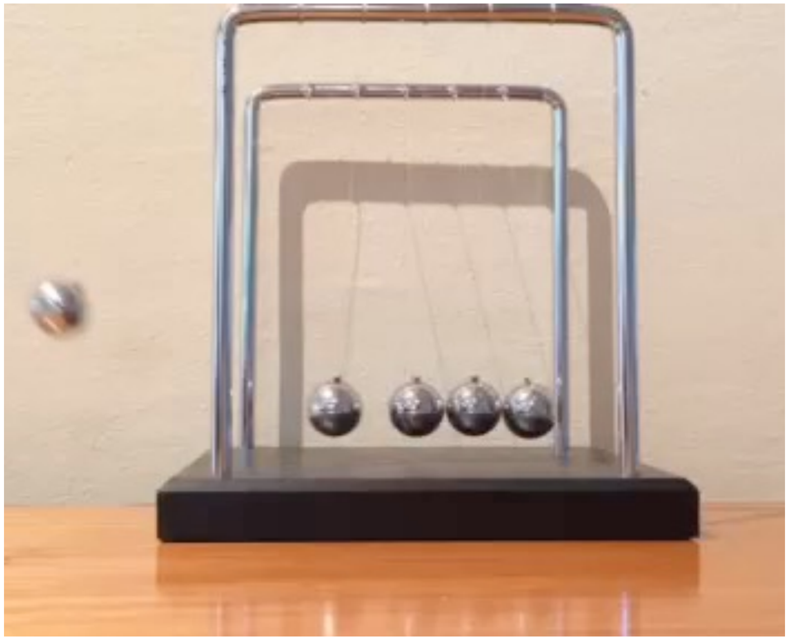
$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \mathbf{F}(\tau)d\tau$$

Image: Wikimedia Commons

Models vs. Reality

$$x(t) = x(0) + \int_0^t v(\tau)d\tau$$
$$v(t) = v(0) + \frac{1}{m}\int_0^t F(\tau)d\tau,$$

The model



The target (the thing being modeled).

In this example, the *modeling framework* is calculus and Newton's laws.

Fidelity is how well the model and its target match

A Model

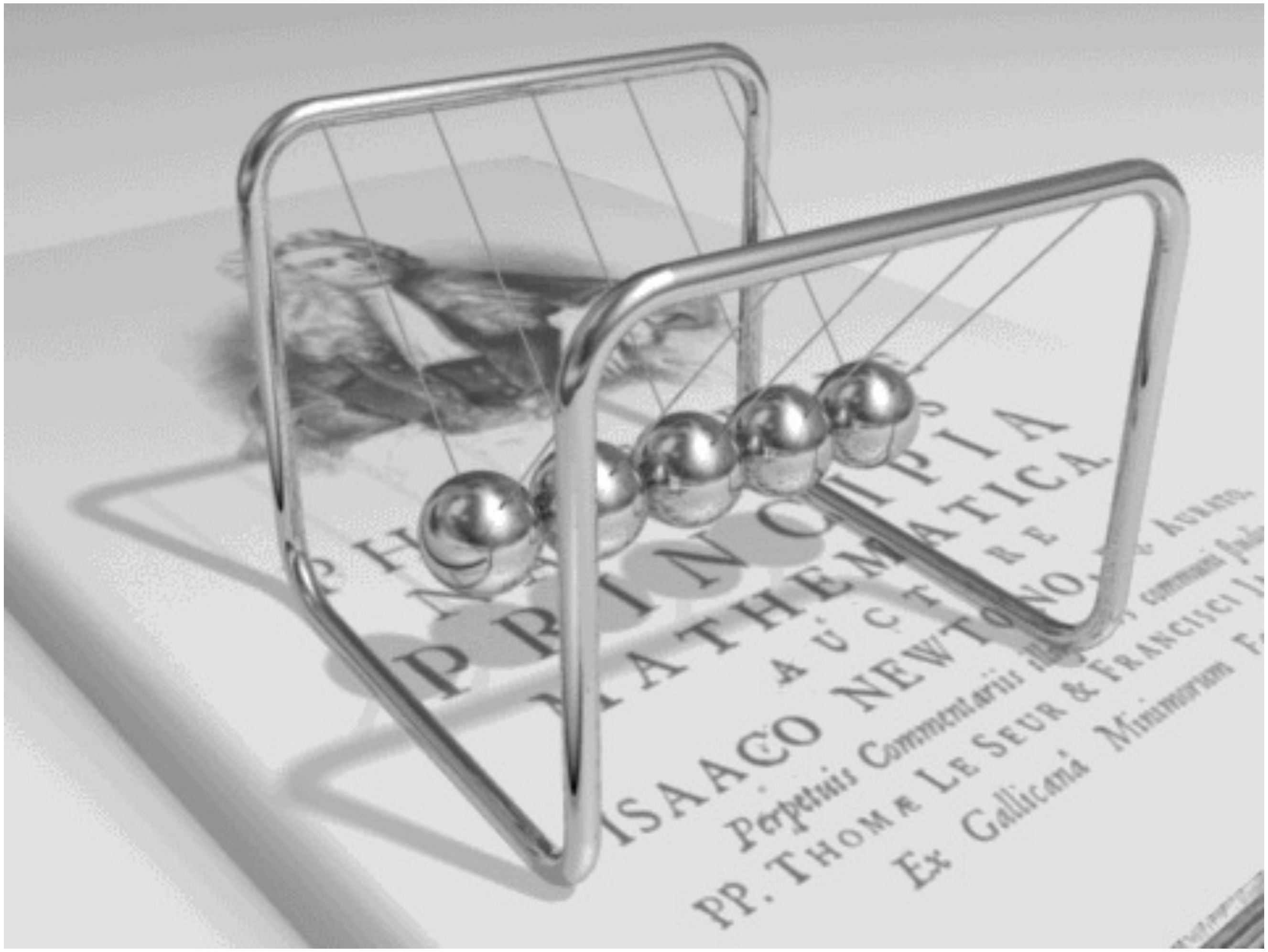
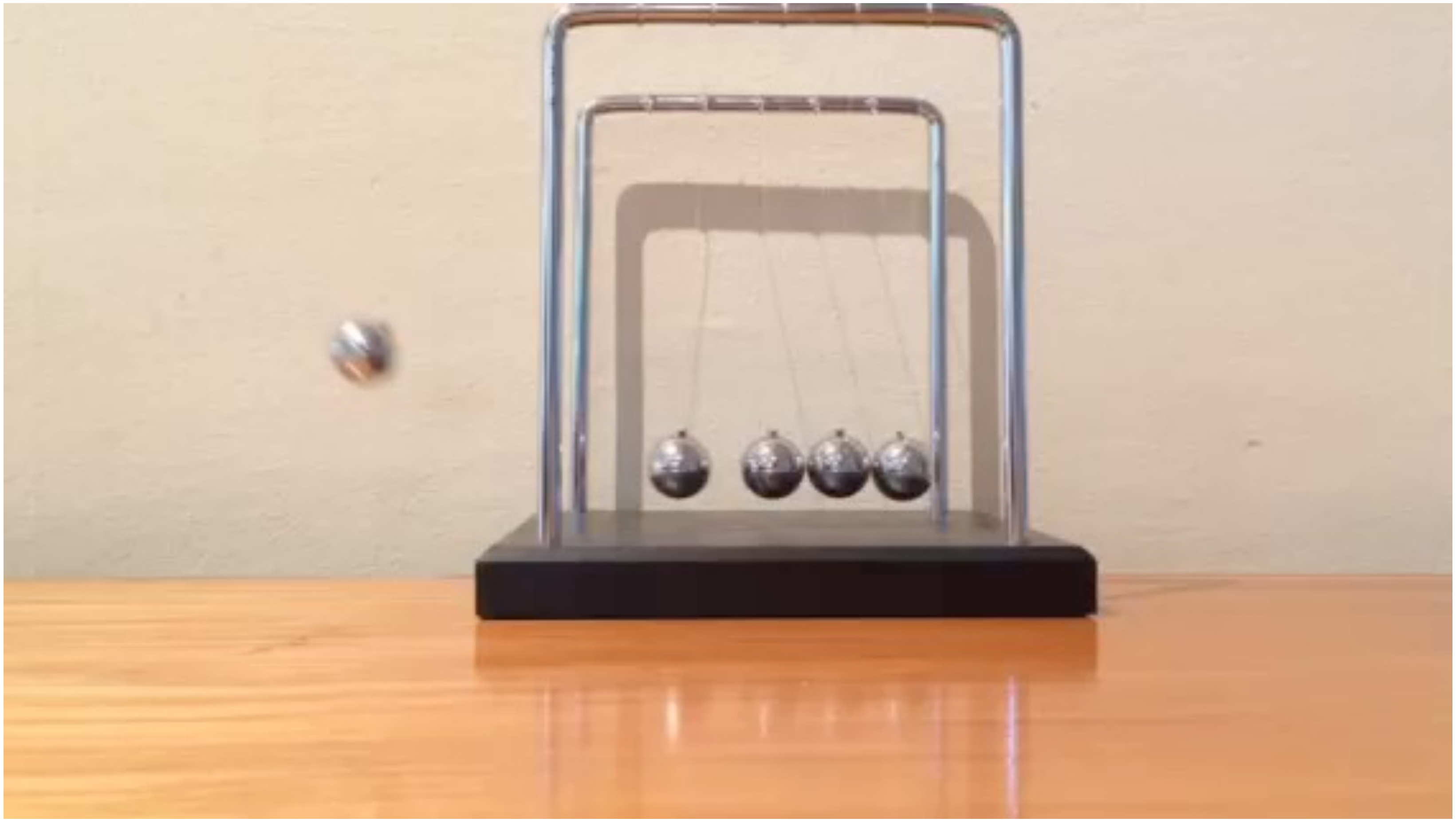


Image by Dominique Toussaint, GNU Free Documentation License, Version 1.2 or later.

A Physical Realization



The Value of Simulation

"Simulation is doomed to succeed."

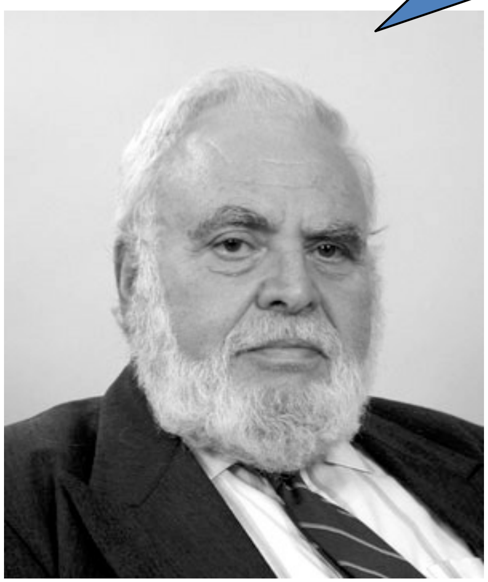
Could this statement be confusing engineering and scientific models?

Figure 1: Three scenes generated from a single ~20-line SCENIC scenario representing bumper-to-bumper traffic.

[Fremont, et al., Scenic: Language-Based Scene Generation, Arxiv.org, Sept. 2018]

Engineers often confuse the model with the thing being modeled

You will never strike oil by drilling through the map!



Solomon Wolf Golomb

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

Lee, Berkeley

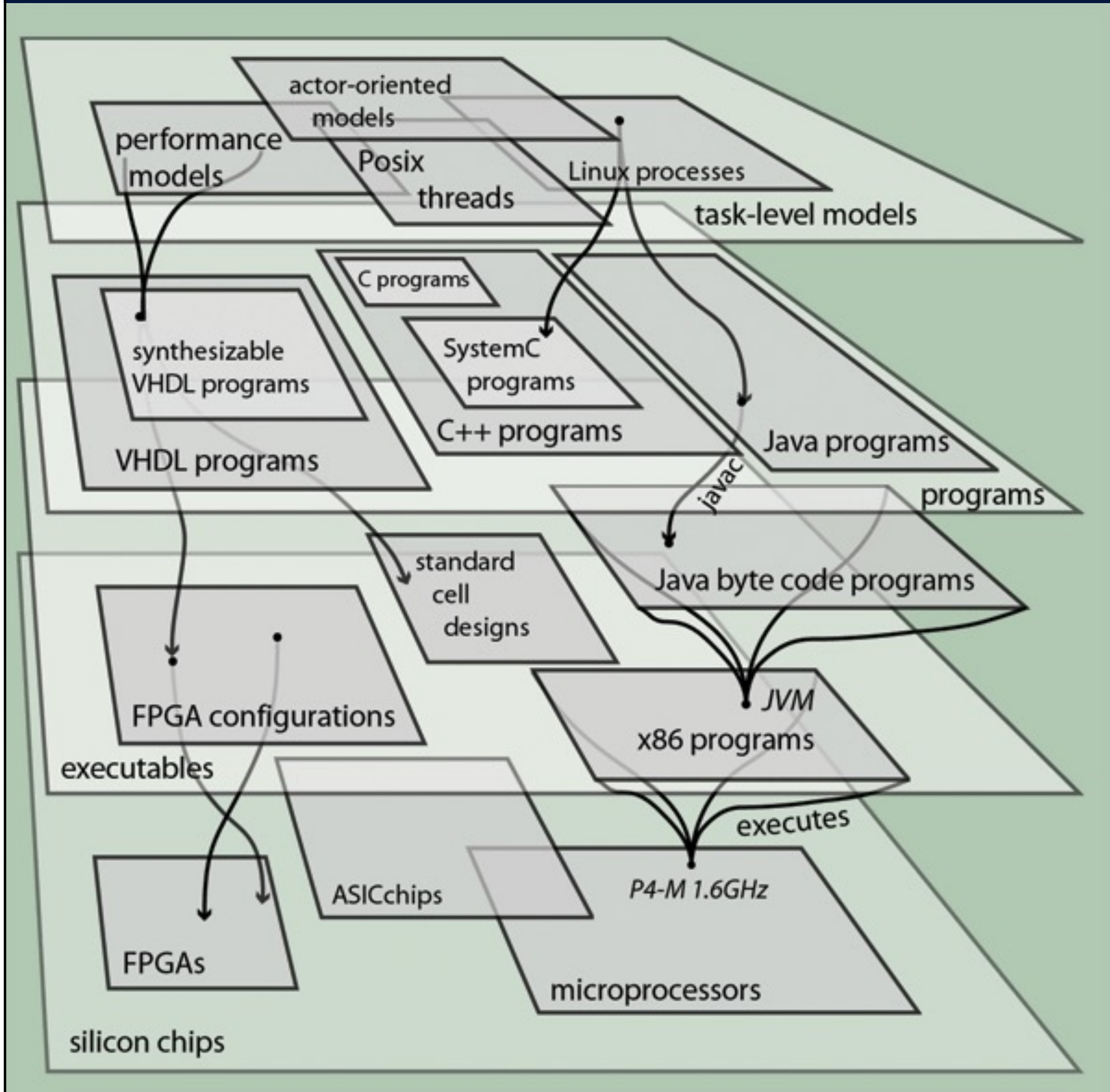
Recall Avionics

In “fly by wire” aircraft, computers control the plane, mediating pilot commands.

CCA 2.0
Boeing Dreamscape

Abstraction Layers

All of which are models except the bottom



The purpose of an abstraction is to hide details of the implementation below and provide a platform for design from above.

Abstraction Layers

All of which are models except the bottom

Every abstraction layer has failed for the aircraft designer.

The design *is* the implementation.

Frozen Designs

Everything about the design, down to wire lengths and microprocessor chips, must be frozen at the time of design.

CCA 2.0
Boeing Dreamscape

Contrast with correctness criteria in software

We can safely assert that line 8 does not execute, *regardless of the choice of microprocessor*!

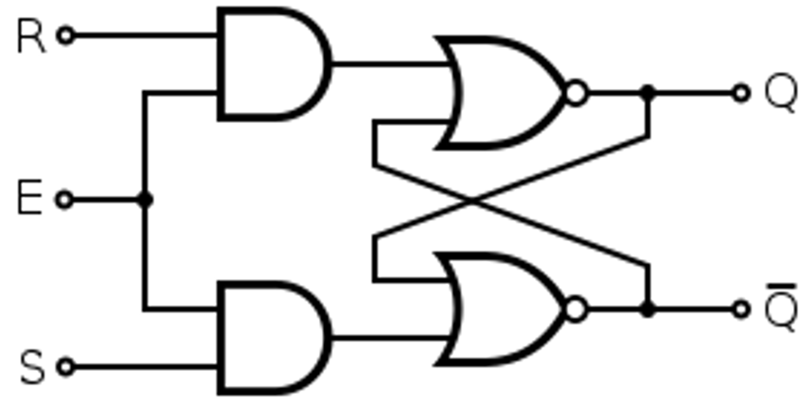
```
1  void foo(int32_t x) {
2          if (x > 1000) {
3                  x = 1000;
4          }
5          if (x > 0) {
6                  x = x + 1000;
7                  if (x < 0) {
8                          panic();
9                  }
10         }
11 }
```

We can develop **absolute confidence** in the software, in that only a **hardware failure** is an excuse.

But not with regards to timing!!

Hardware is Good at Timing

Synchronous digital logic delivers precise, repeatable timing.



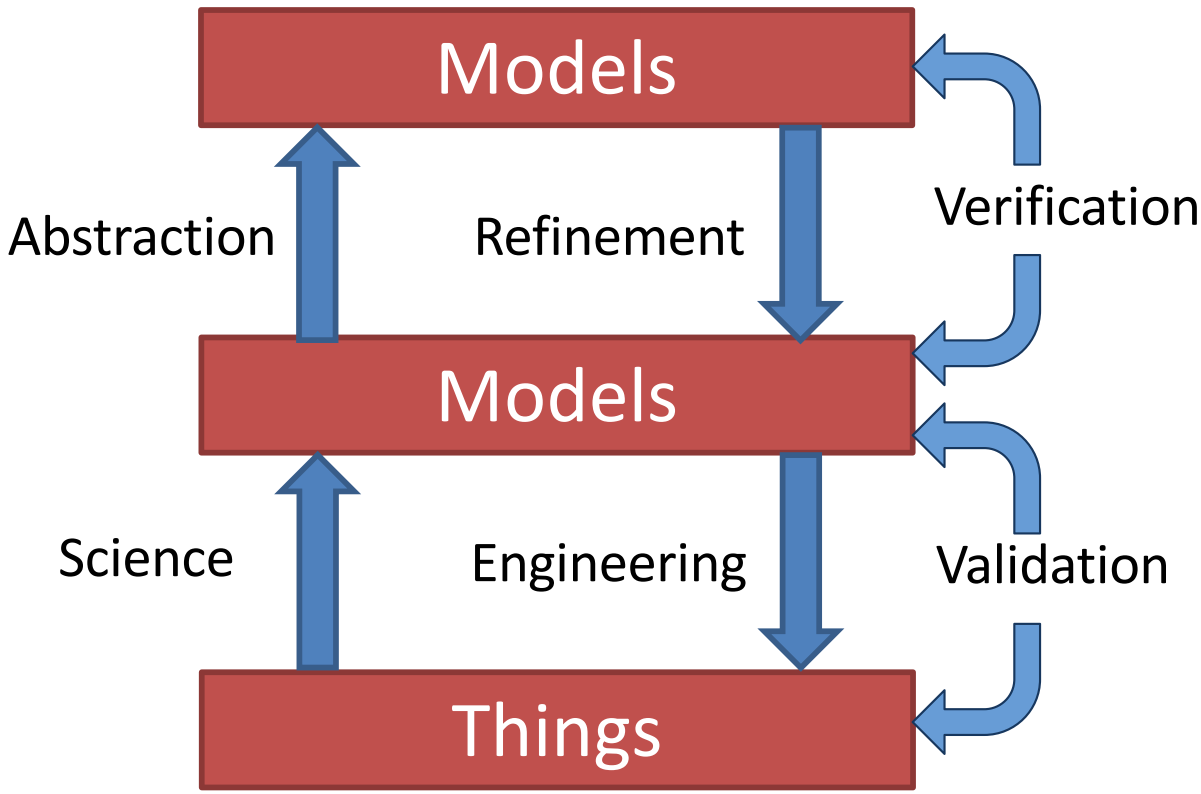
… but the overlaying software abstractions discard timing.

```
// Perform the convolution.
for (int i=0; i<10; i++) {
  x[i] = a[i]*b[j-i];
  // Notify listeners.
  notify(x[i]);
}
```

Verification and Validation

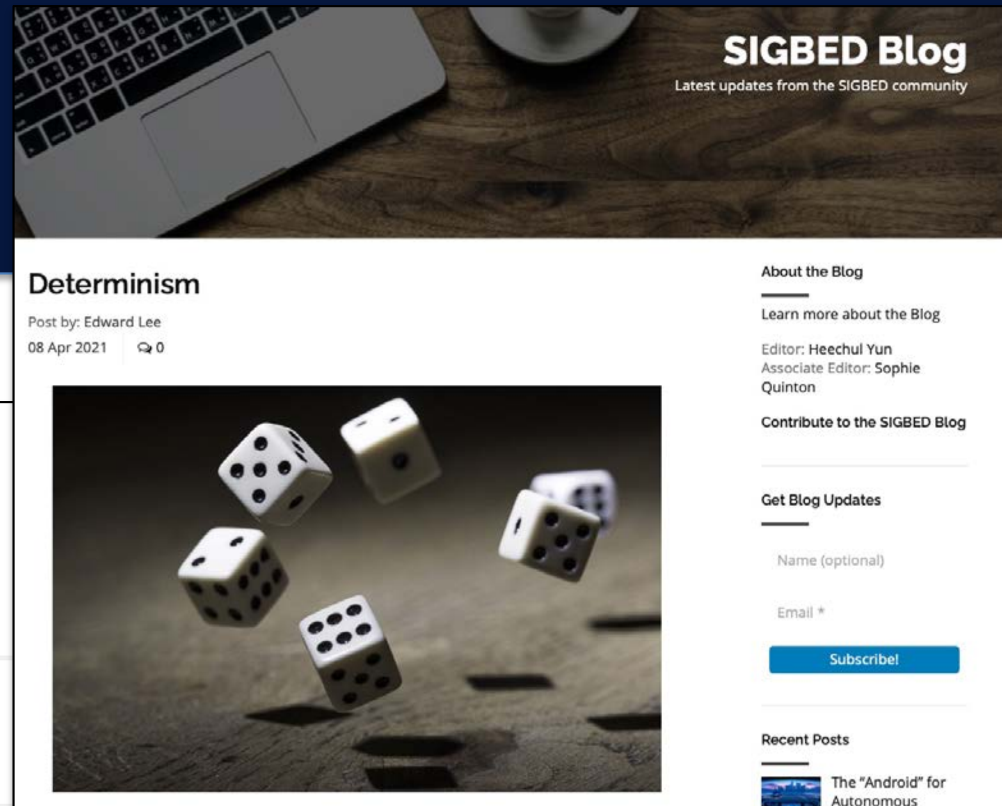
Per Boehm:

- Am I building the right product? (validation)
- Am I building the product right? (verification)



Formal methods can only address the Verification question.

References



RESEARCH-ARTICLE OPEN ACCESS

Determinism

Author: Edward A. Lee Authors Info & Affiliations

ACM Transactions on Embedded Computing Systems, Volume 20, Issue 5 • July 2021 • Article No.: 38, pp 1–34 • https://doi.org/10.1145/3453652

http://ptolemy.org/~eal
eal@berkeley.edu

http://repo.lf-lang.org
Lingua Franca