Hybrid Discrete-Continuous Computer Architectures for Post-Moore's-Law Era

Keynote at the Bi-annual HiPEAC Computing Systems Week Meeting
Barcelona, Spain
October 19th 2010

Prof. Simha Sethumadhavan
Columbia University
www.cs.columbia.edu/~simha

The FUTURE depends on you!

Executive Summary

- **Applications demand more**
  - Scientific & societal progress depends on better computers
- **Silicon scaling is slowing down**
  - Energy usage more important than hardware cost
- **Changing landscape demands new solutions**
  - Multi-cores and accelerators will not scale
- **A solution: Analog-Digital Computing**
  - Great match for emerging applications
  - "New and Improved" old technology!

What is Computing?

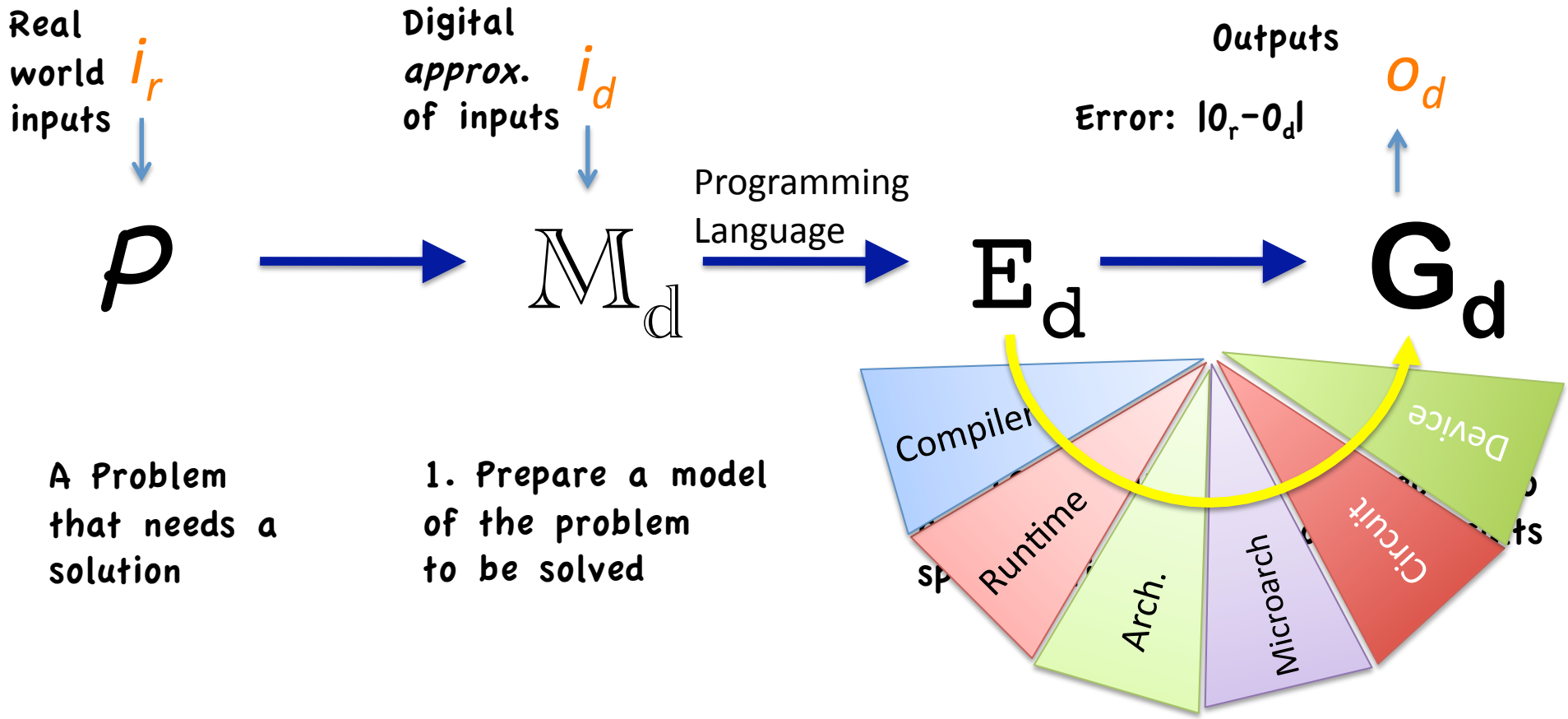
A Problem that needs a solution

1. Prepare a model of the problem to be solved

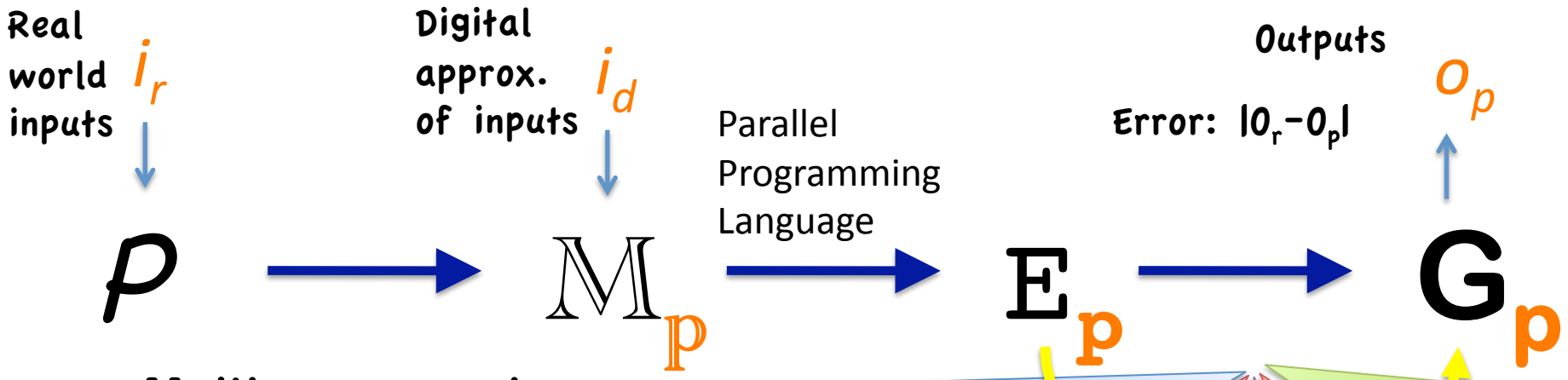
2. Prepare an executable specification

3. Execute to obtain results

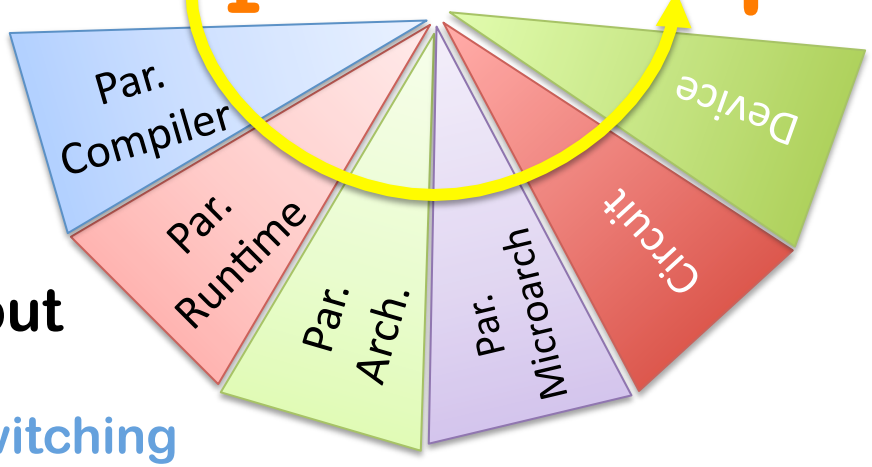
Digital Computing



Multicore Computing



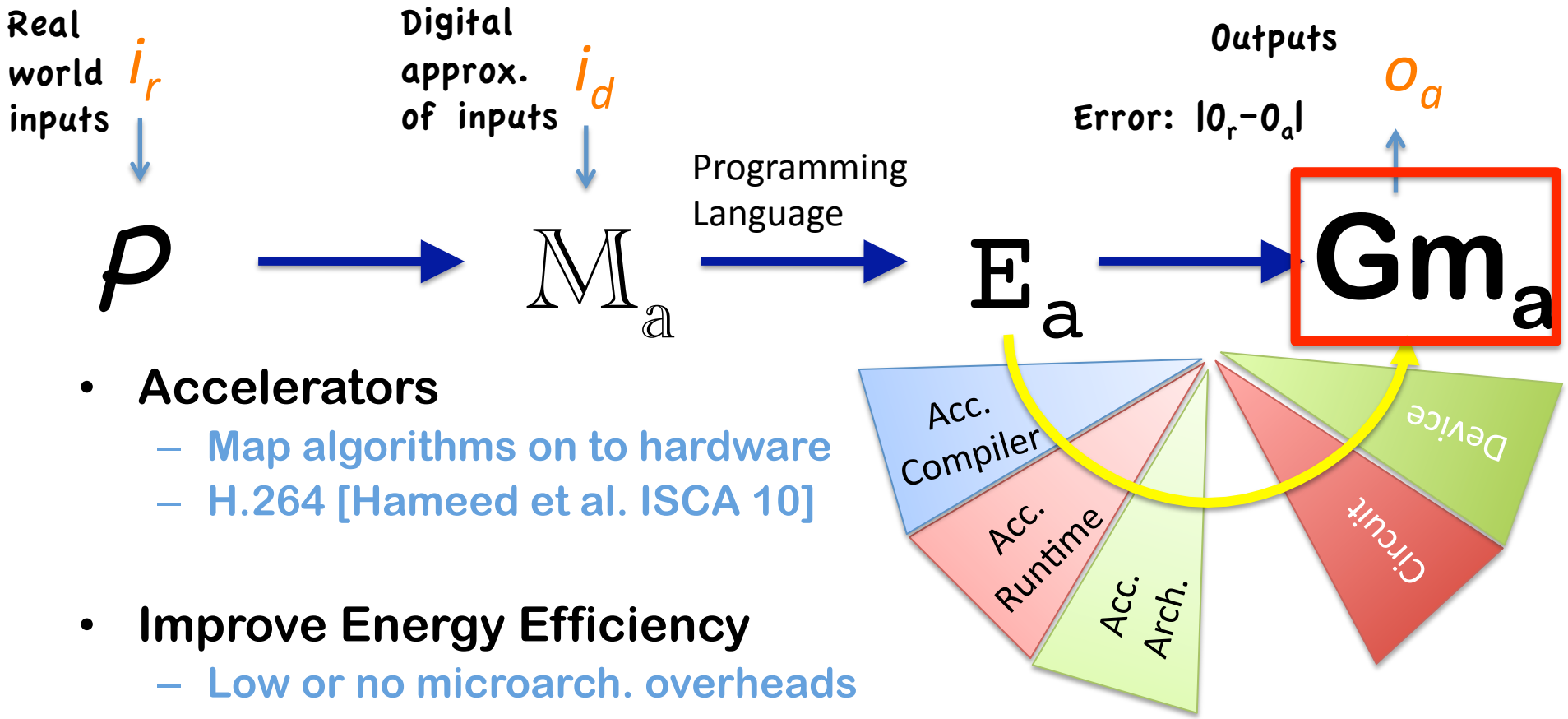
- **Multicore promise**
  - Reduce power
  - Same time increase throughput



- **Reality: Low power & throughput**
  - Serial regions limit throughput
  - Voltage scaling limits on-chip switching
  - Die &Wafer sizes are not growing ➔ cores cannot increase

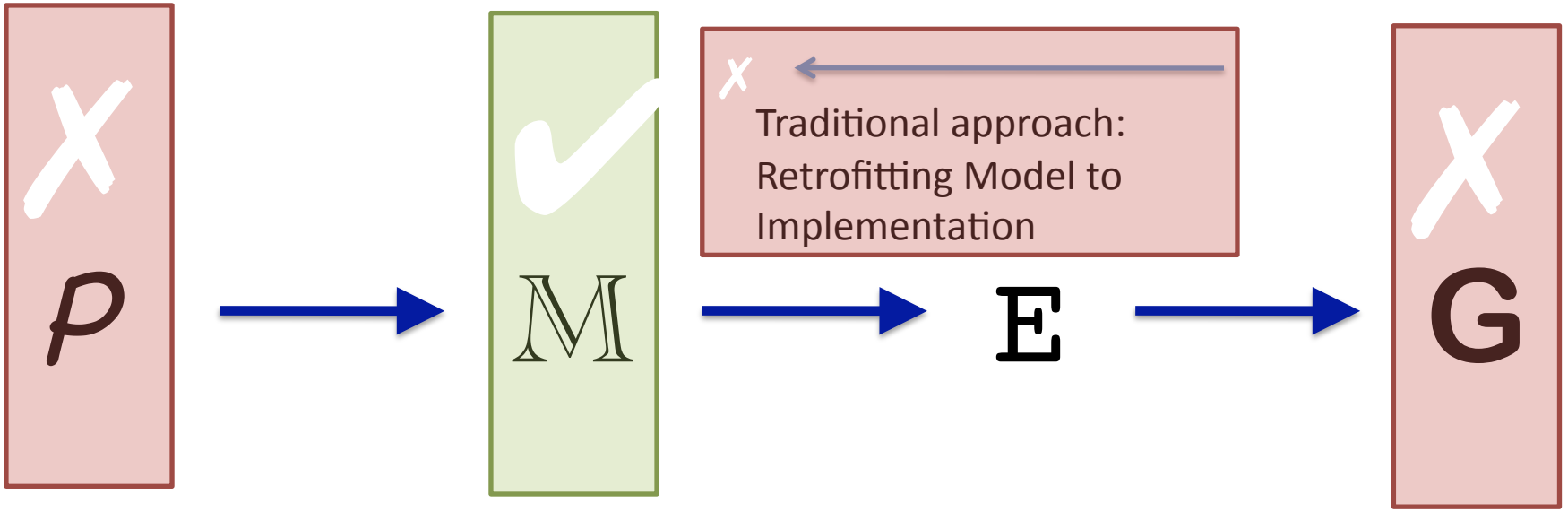
- **Multicores have merely postponed the "power wall"**

Accelerator Computing



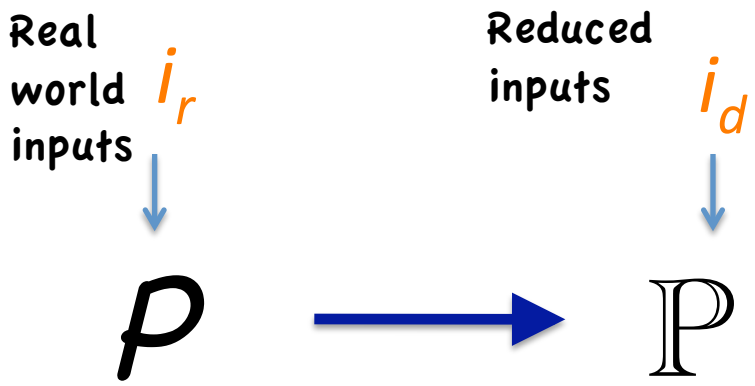
- **Accelerators**
  - Map algorithms on to hardware
  - H.264 [Hameed et al. ISCA 10]
- **Improve Energy Efficiency**
  - Low or no microarch. overheads
  - Some die area can go unused
- **Further energy efficiency improvements are difficult**

Next energy efficiency leap?



ATTACK FUNDAMENTAL ALGORITHMIC OVERHEADS

Attempt 1: *Au naturel* Computing

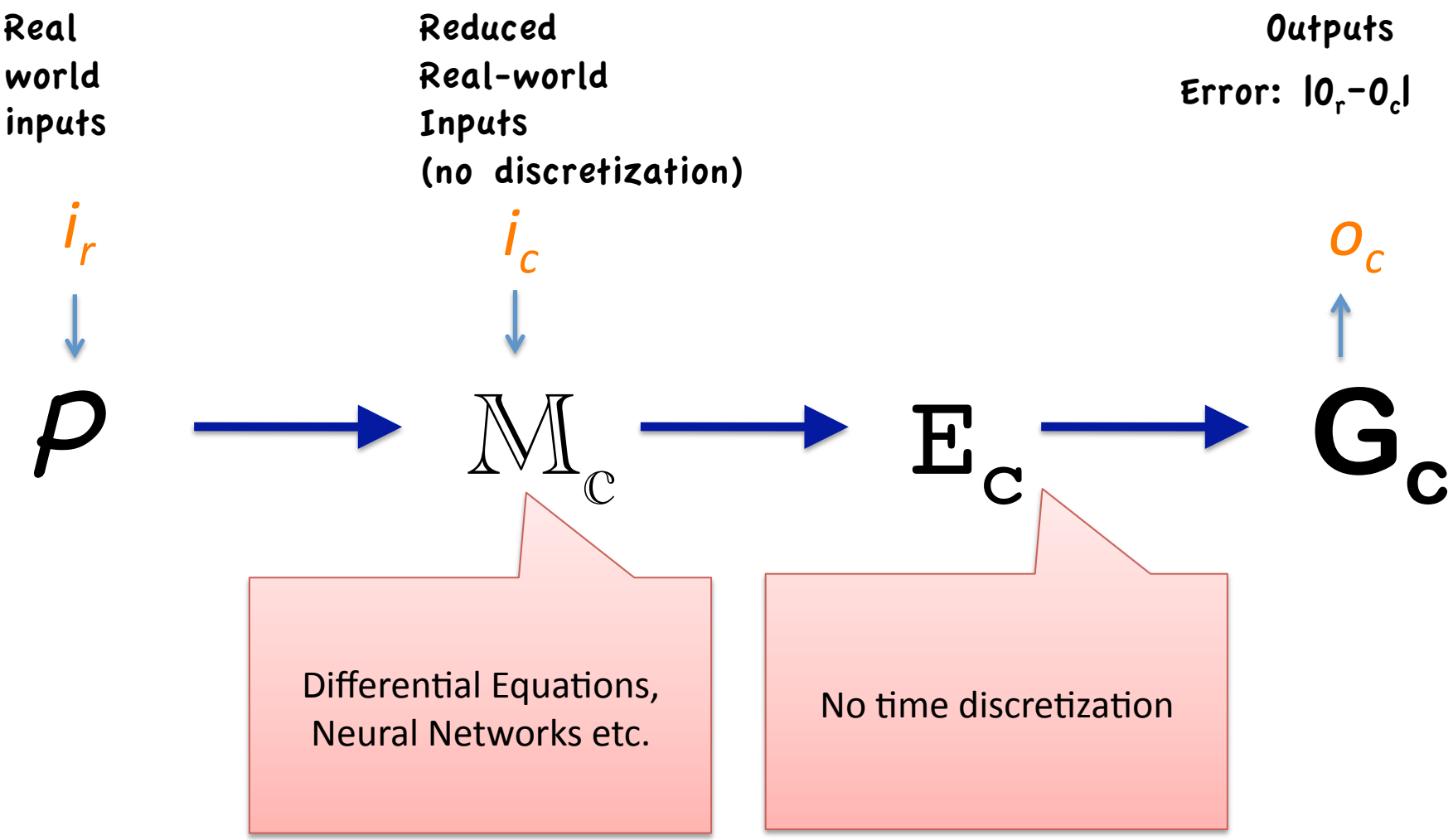


- **Example: Hurricane in a bottle**

- **Fast but *au naturel* is also unnatural**
  - For computer scientists
  - But this is what physical scientists do
  - Great if you can simulate the problem on a reduced scale

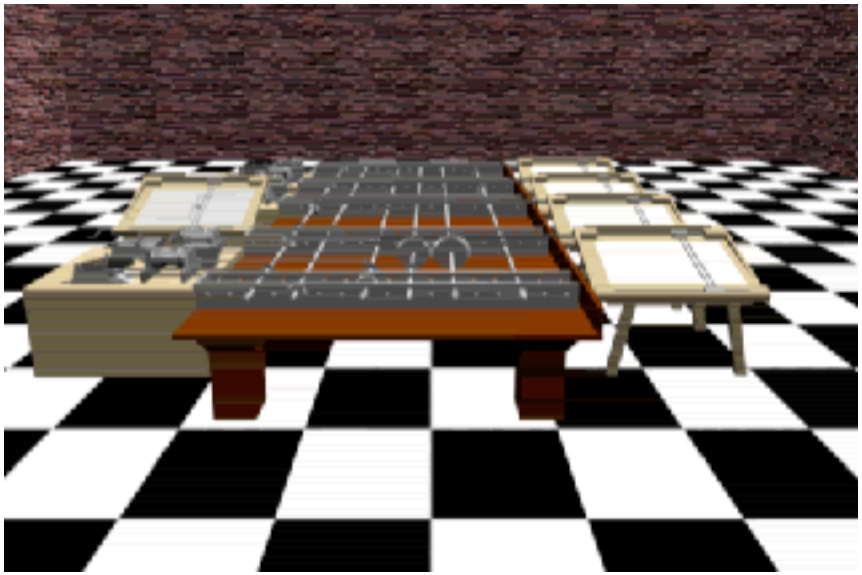
Improvement: Continuous Computing

- Use mathematical *formulations* used by scientists



Example Continuous Computer

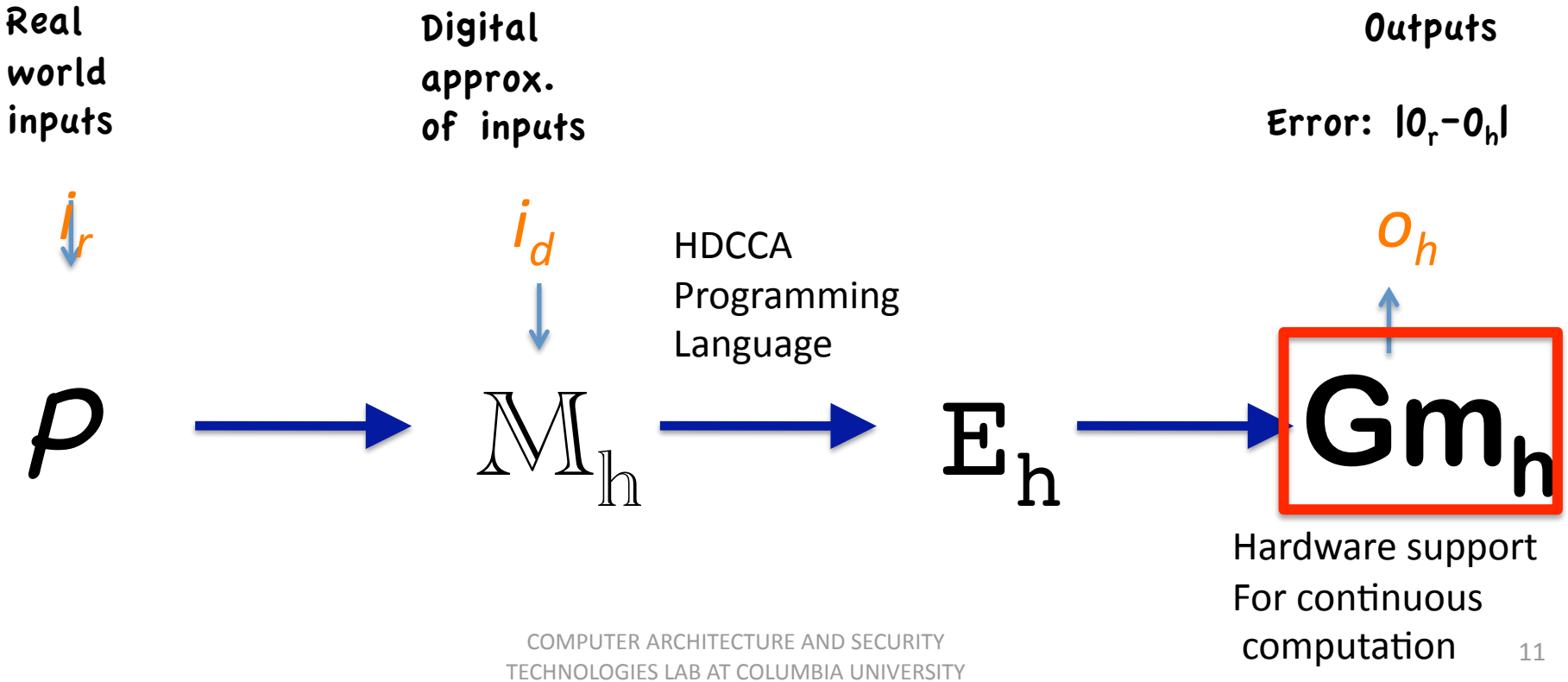
- **The Linear Differential Analyzer**
  - http://web.mit.edu/klund/www/analyzer/



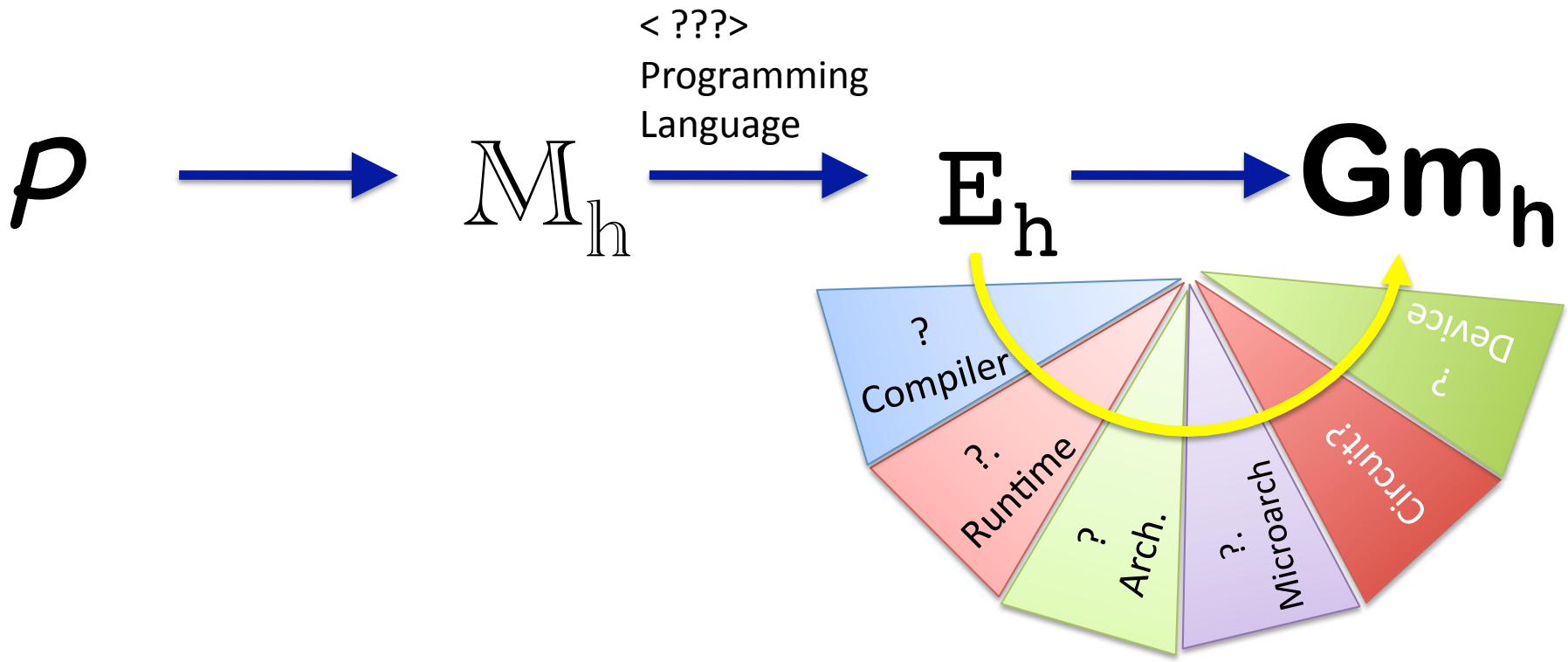
- **Problems:**
  - **Limited accuracy, not a practical general-purpose machine.**

The HYBRID Discrete-Continuous Model

- **Combine discrete and continuous models**
  - A more natural fit for computing
  - Discrete problem on discrete computer e.g., FSM
  - Continuous problem on continuous e.g., differential eqns.
- **Better for programmability, efficiency & accuracy**



HDCCA Research



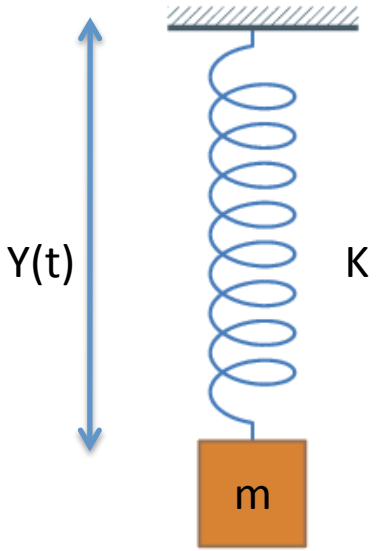
What should the HDCCA computing stack be?

Outline

- Introduction to HDCCA
- A simple End-to-End example (mini tutorial)
  - Mini-tutorial goals
    - Solidify understanding of differences between discrete and continuous by studying differential equations.
    - Show continuous implementation with analog hardware
- Analog Old Vs. Analog New
- Research Challenges for HDCCA
- Tangible Benefits

A Simple End-to-End Example

- **Damped Harmonic Motion**
  - Toy, text book example
  - Spring attached to mass m
  - Spring constant k
  - Acted by external force F



- **We are trying to determine what the position at time T?**
  - Solution is given by the following equation:

$$m\frac{d^2y}{dt^2} + b\frac{dy}{dt} + ky = f(t)$$

Discrete Solution

- ***Step 1:*** **Write equations in matrix form**

$$\begin{bmatrix} u_1'(t) \\ u_2'(t) \end{bmatrix} = \begin{bmatrix} u_2(t) \\ -\frac{1}{m}[bu_2(t) + ku_1(t) - f(t)] \end{bmatrix}$$

$$u_1(t) = y(t)$$
$$u_2(t) = y'(t)$$

- ***Step 2***: **Compute values at time** `h` **based on Initial Values**

$$u_1(h) = u_1(0) + hu_1'(0)$$
$$u_2(h) = u_2(0) + hu_2'(0)$$

- ***Step 3 … n***: **Compute values at time** `2h` **based on value at** `h`**, and iterate until you converge based on some error bound.**

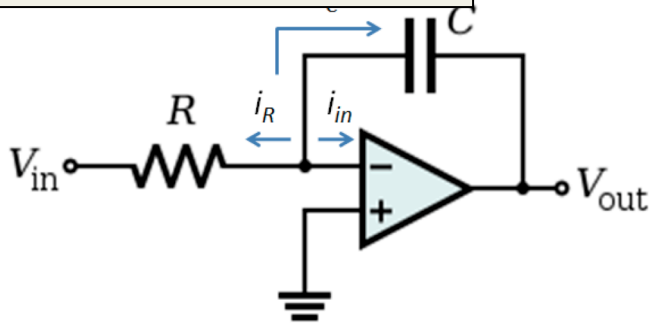
Continuous Solution

- **Integrate twice using op-amps**

$$m\frac{d^2y}{dt^2}+b\frac{dy}{dt}+ky=f(t) \qquad \frac{d^2y}{dt^2}=-\left[\frac{1}{m}\left(b\frac{dy}{dt}+ky-f(t)\right)\right] \qquad Vy=-\int_0^t\left(\frac{1}{m}f(t)-\frac{b}{m}y'-\frac{k}{m}y\right)dt=-\int_0^t y''dt=-y'$$

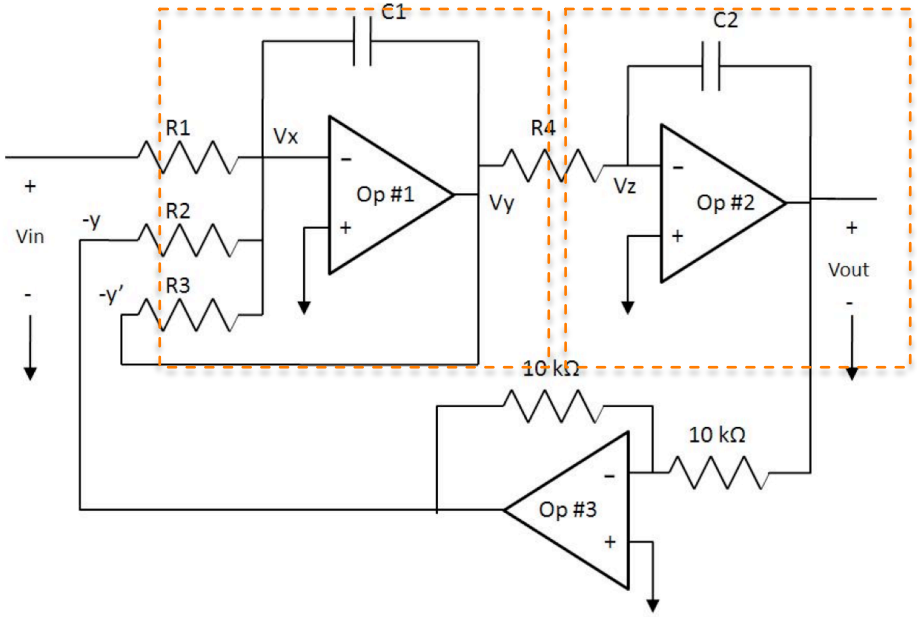
- **Scale down R,C value based on 1/m, b/m & k/m**
  - **Determines solution time**

Analog Circuit for Integration



$$V_{out}(t)=-\frac{1}{RC}\int_0^t V_{in}dt+V_{out}(0)$$

Analog Hardware Circuit



Comparison of Discrete vs. Continuous

- **Both produce approximate answers**
  - **Measured time to arrive at ±2.5% of analytical value**

Equation	Ode solver	Iterations	Run-time	Cycles	Maximum Estimated Speed-up
$10y''(t) + y(t) = 0$	ode23	198	0.01638	49140315	819
$y''(t) + 100y(t) = 0$	ode45	2557	0.080653	241957551	4032.65

- **Speed ups due to fact that we avoided discrete time stepping.**

Outline

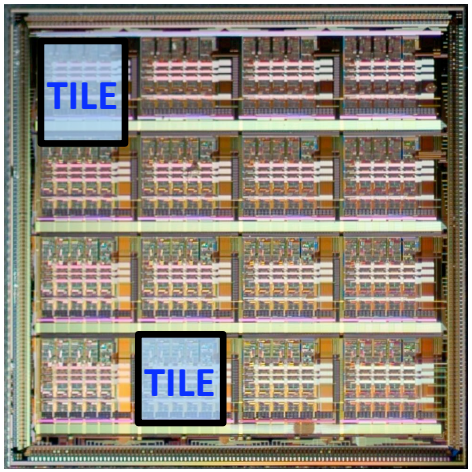
- Introduction to HDCCA
- A Simple End-to-End Example
- Analog Old Vs. Analog New
  - We are using analog to implement the continuous model
  - But, isn't analog dead?
  - Understand why digital superseded analog in the 70s.
  - Show that critical barriers to analog have been solved
- Research Challenges for HDCCA
- Tangible Benefits

Old Vs. New: Accuracy

- Old analog susceptible to noise and error
  - Cannot get more than 11-12 good bits
  - New analog devices are not much better
  - But the application landscape has changed
- #1: Many modern apps do not need high accuracy
  - Graphics, Optimization
- #2: Many modern apps are error-tolerant
  - Games, Learning
- #3: For high accuracy applications HDCCA is useful
  - Refine approximate analog values using digital solver!

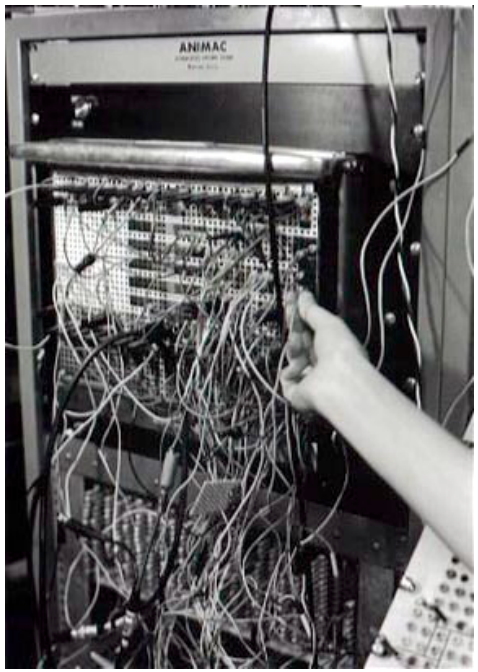
Old Vs. New: Design/Implementation

- Old: Lack of CAD tools, design methodology
- New: Still black art/engineering (like parallel prog?!)
  - Note that: Digital design complexity is approaching analog design complexity
  - But analog CAD is improving
- A recent successful analog design
  - Off-chip co-processor
  - Published in 2005 in ISSCC
  - Cowan and Tsividis @ Columbia



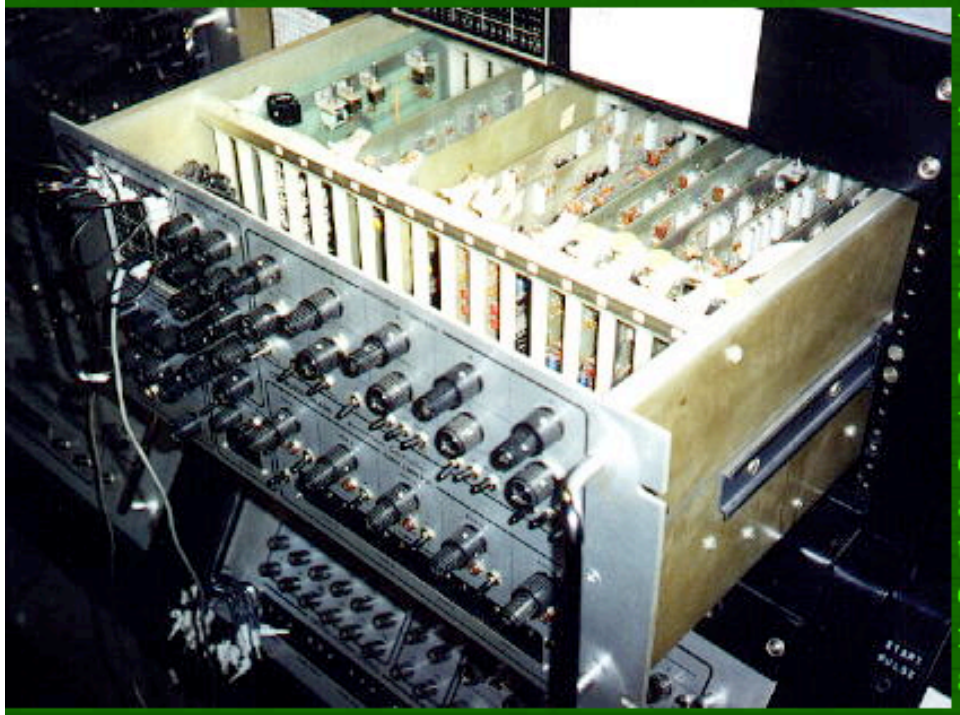
Old Vs. New: Programmability

- Old machines were large, clunky
- Scanimate graphics system [Siggraph 98]
  - "Most famous video produced by Scanimate – Death Star"



Old Vs. New: Programmability

Scanimate Tour: The CPU



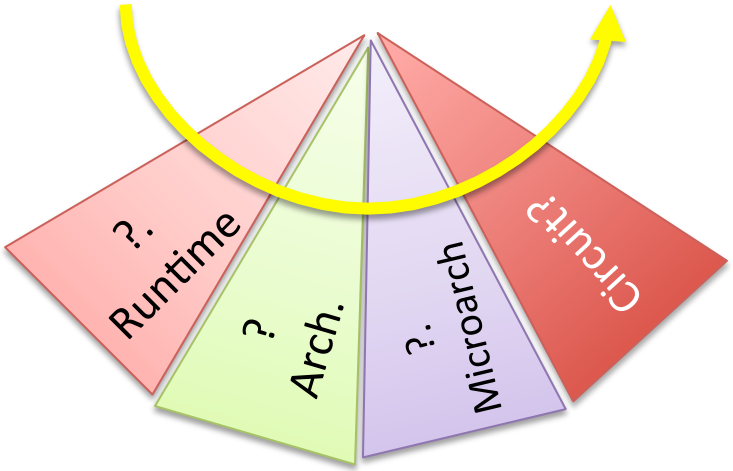
When I first got acquainted with the Scanimate in 1980, I had worked with microprocessors, building my own single board computer based on the RCA 1802... but that's another story. The point was that it took me a while to get used to calling a bunch of analog ramp generators, summing amps, multipliers, and phase-lock oscillators a CPU! But analog computers had been around a long time. The beauty of them is that you can have literally thousands of variables, each connected to a knob, and turning any or all of them produces an instantaneous (well, almost!) solution at the output(s). For animation, this was great, because you could show a client the element you were working on right then in finished form. He could suggest changes to the movement, appearance, whatever he wanted, and you could turn a few knobs, move a few wires, and there it was. Want it a little slower? Turn a knob. Want it exactly 10% slower? Forget it! Want exactly the same thing we did yesterday? You can forget that too! There is no such thing as duplicating a job on a scanimate. The phases of the moon, the earth's rotation, the cosmic karma of the gods of Analog all contributed to each masterpiece being necessarily unique! (That's the best spin on it I've ever read, if I do say so myself!)

Old Vs. New: Programmability

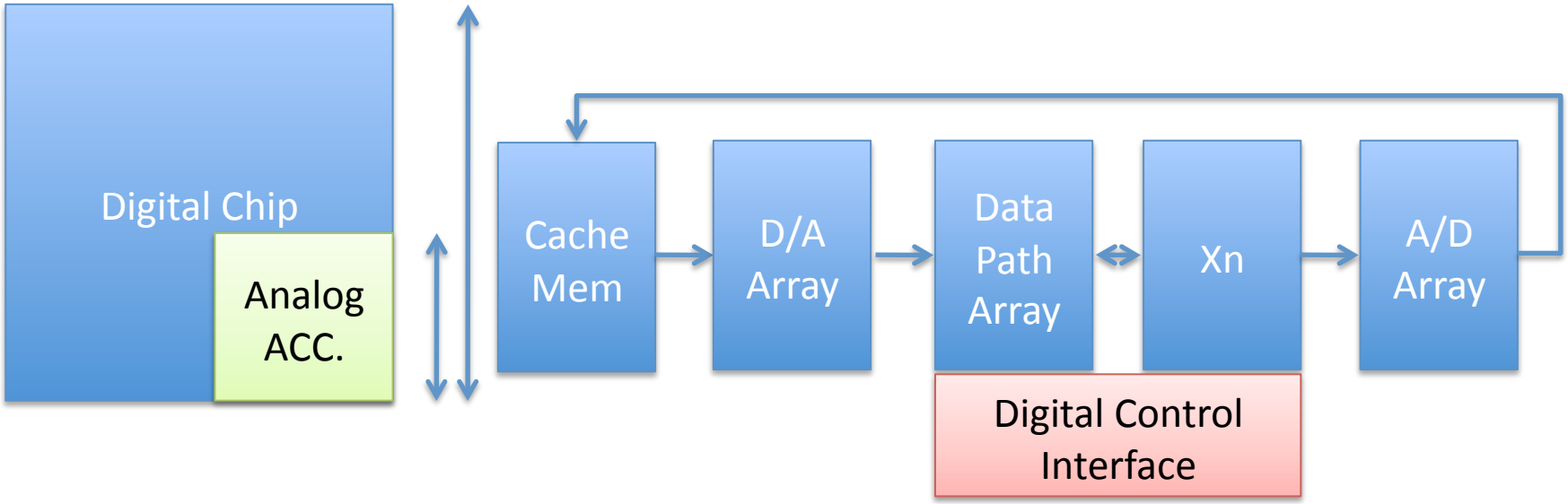
- **Old:**
  - Had to manually patch wires for programming
- **New:**
  - RC values can be programmed by digital
- **But more development is needed**
  - Compiler
  - Toolchain
  - Programming language
  - Development environment etc.,

Outline

- Introduction to HDCCA
- A simple End-to-End Example
- Analog Old Vs. Analog New
- Research Challenges for HDCCA



Research Challenges: Microarchitecture



Microarch	• How much on-chip area should be alllocated to Analog? • What type of functional units should be included? • Should the units be connected with circuit or packet switching? • How many input output channels to digital should be created? • Should the Datapath and ADCs be operated at different speeds?

Research Challenges: Architecture

Analog Interfaces	Functionality
Configuration	Configure the datapath for a sub problem
Calibration	To query processor state when computation is carried out.
Compute	Export types of functional units available
Completion	When should the output values be sampled?

Arch.	**• What is the machine model?** **• Should we use the accelerator as a slave or standalone?** **• What are the semantics of the instructions?**
Microarch	• How much on-chip area should be alllocated to Analog? • What type of functional units should be included? • Should the units be connected with circuit or packet switching? • How many input output channels to digital should be created? • Should the Datapath and ADCs be operated at different speeds?

Research Challenges: Compiler/PL

PL	• What should the continuous languages primitives be?
Compiler	• Do we need a separate static and dynamic compiler?
Arch.	• What is the machine model? • Should we use the accelerator as a slave or standalone? • What are the semantics of the instructions?
Microarch	• How much on-chip area should be alllocated to Analog? • What type of functional units should be included? • Should the units be connected with circuit or packet switching? • How many input output channels to digital should be created? • Should the Datapath and ADCs be operated at different speeds?

Research Challenges: Algorithms

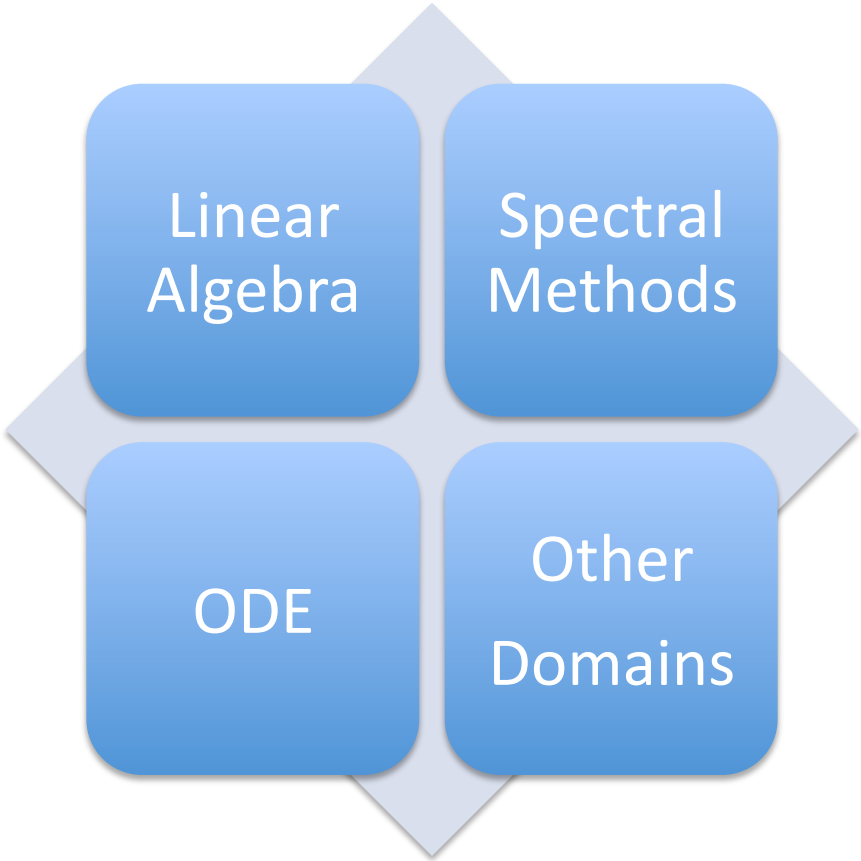
Algorithm Developer	• Development of algorithms to decompose tasks? • Can we come up with a formal theory of how errors should be handled?
PL	• What should the continuous languages primitives be?
Compiler	• Do we need a separate static and dynamic compiler?
Arch.	• What is the machine model? • Should we use the accelerator as a slave or standalone? • What are the semantics of the instructions?
Microarch	• How much on-chip area should be alllocated to Analog? • What type of functional units should be included? • Should the units be connected with circuit or packet switching? • How many input output channels to digital should be created? • Should the Datapath and ADCs be operated at different speeds?

Outline

- Introduction to HDCCA
- HDCCA mini tutorial
- Analog Old Vs. Analog New
- Research Challenges for HDCCA
- Tangible Benefits of HDCCA
  - Analysis of existing benchmark suites
  - Mapping a non-straightforward problem on to HDCCA

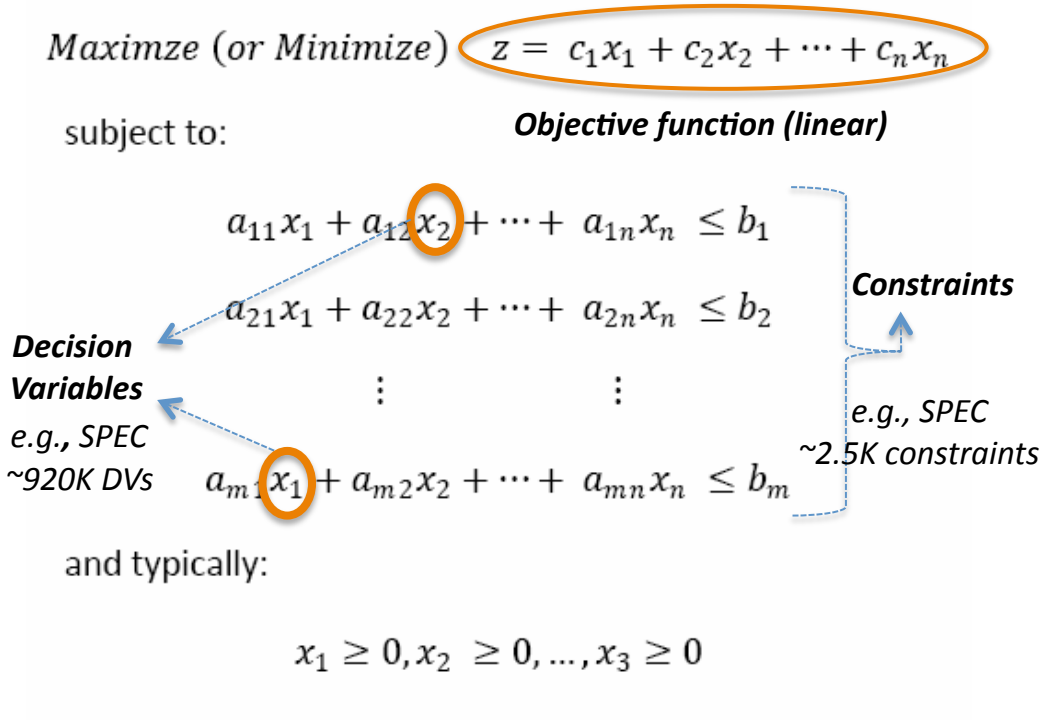
Analog Accelerator Utility

- **Examined three benchmark suites**
  - SPEC CFP
  - Intel RMS
  - Berkeley Dwarfs
- **Categorize**
  - Based on problem
  - Not algorithm
- **40 % map to HDCCA**

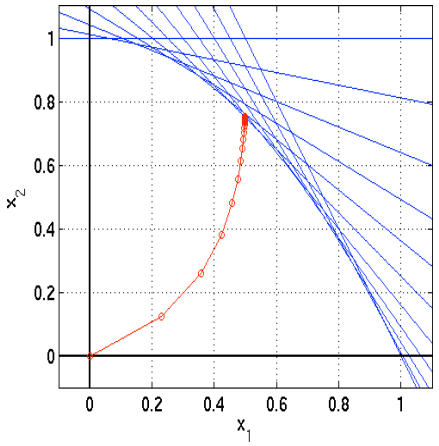


Solving Linear Programming

- **Soplex in SPEC solves LP using Simplex**
  - **Cannot be directly mapped on to analog accelerator**
  - **Alternate formulation of the problem is needed**

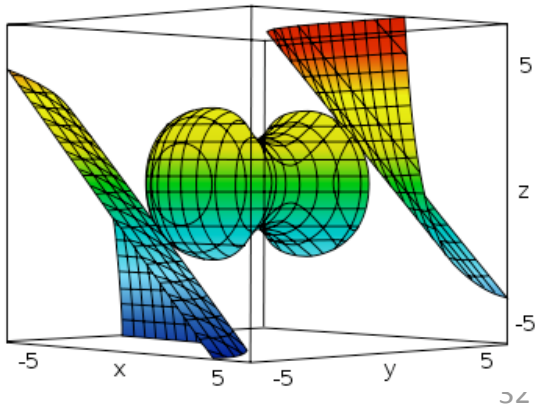


Analog Mapping

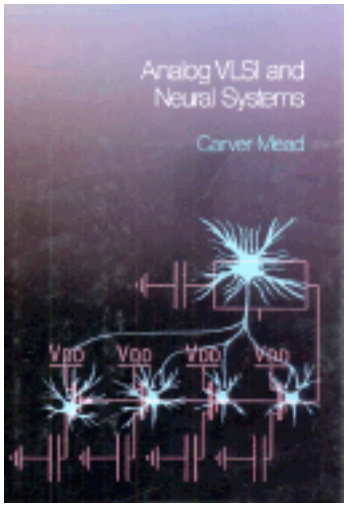


- **Solved using gradient descent instead of simplex**
  - Generate a moving in the solution space
  - Move the point based on the objective function
  - When P is maximum of minimum the gradient goes to zero
  - P is the solution

- **Also works for non-linear constraints!**



Related Work



Several Thesis

* Received by the IRE, June 28, 1961; revised manuscript received, November 20, 1961.
† Electrical Engineering Dept., University of Arizona, Tucson, Ariz.

The Impact of Hybrid Analog-Digital Techniques on the Analog-Computer Art*

GRANINO A. KORN†

EXPERIENCE WITH HYBRID COMPUTATION

E. M. King, Jr., and R. Gelman
Missile and Space Division
General Electric Company
Philadelphia 1, Pennsylvania

1942

The Theory and Design of Linear Differential Equation Machines*

Claude E. Shannon

Table of Contents

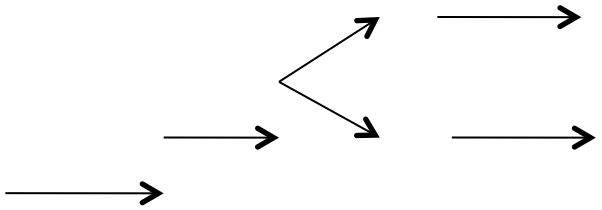
A HYBRID ANALOG-DIGITAL DIFFERENTIAL ANALYZER SYSTEM

John V. Wait
Department of Electrical Engineering
University of Arizona, Tucson, Arizona

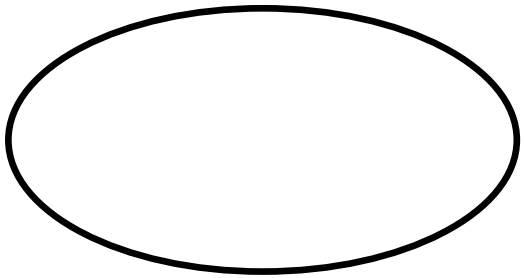
CONCLUDING REMARKS

Philosophy of Engineering Innovations

LINEAR THEORY OF INVENTION



CYCLIC THEORY OF INVENTION



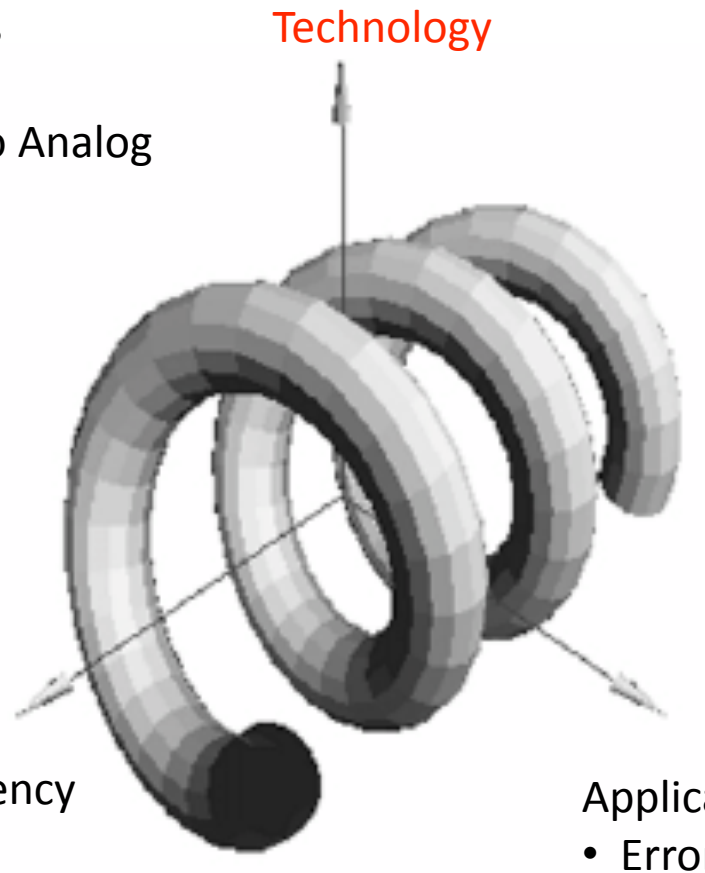
Vector, SIMD MMX
Multiprocessor, Multicores
Networks, Networks on Chip

Use Models

Simha's Philosophy of Engineering Innovations

The right symbolism for computer engineering innovation is at least a helix.

Analog improvements
- Integration
- Digital interfacing to Analog
- Some Automation



New emphasis on energy-efficiency

Application Changes
- Error-tolerant
- Reduced accuracy applications

To CITE This talk use CUCS-026-10

Other Research at CASTL

I. Proactive Security Project

- Current Approach to Security : Patch flaws reactively

- **What if we took a ground up approach?**
  - Secure hardware first
  - Build hardware primitives to support SW security
  - Build SW security countermeasures using HW primitives

II. Accelerating Discovery in Computer Systems

- We are seeing rapid application development
- Traditional methods are too slow to keep pace

- **Can we use Machine Learning and Crowd sourcing to build better systems?**