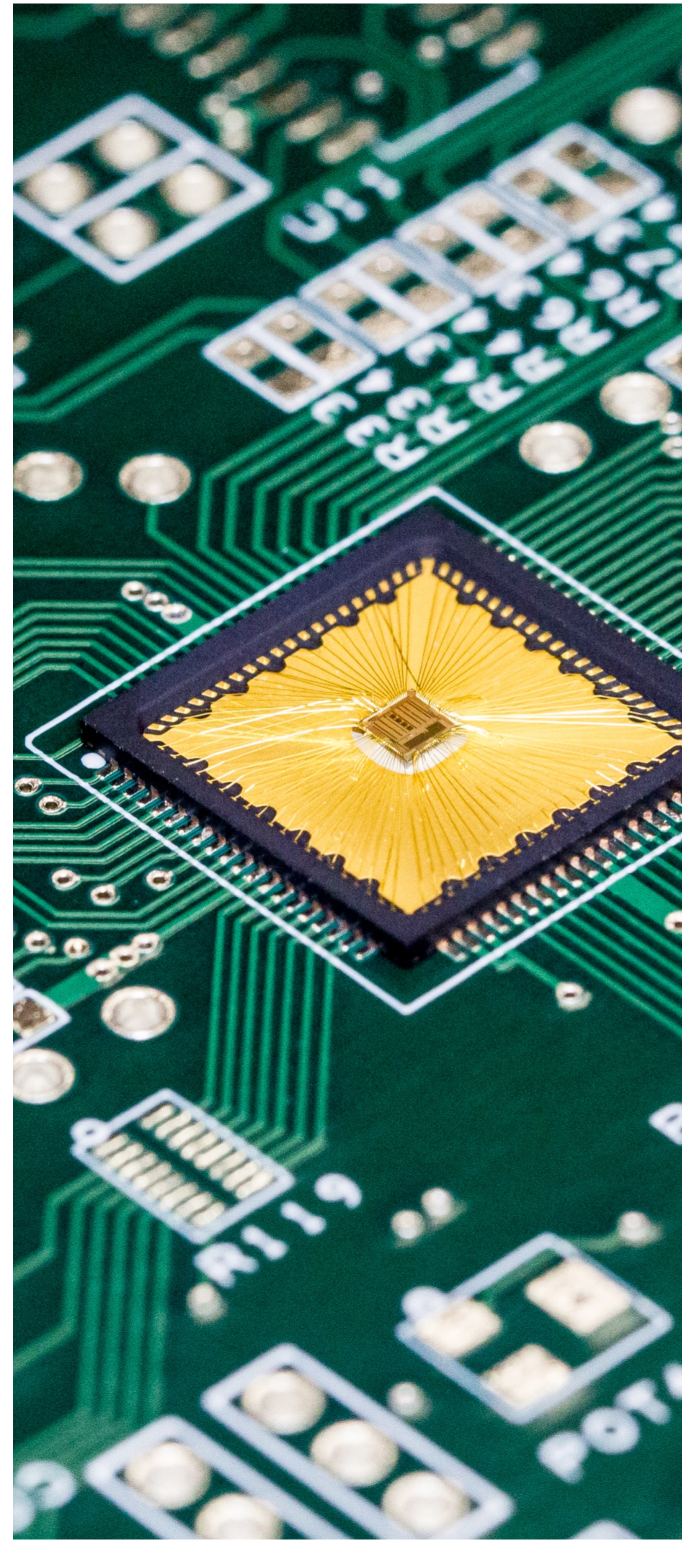
An Analog Accelerator for Linear Algebra

Yipeng Huang, Ning Guo, Mingoo Seok,
Yannis Tsividis, Simha Sethumadhavan

Columbia University



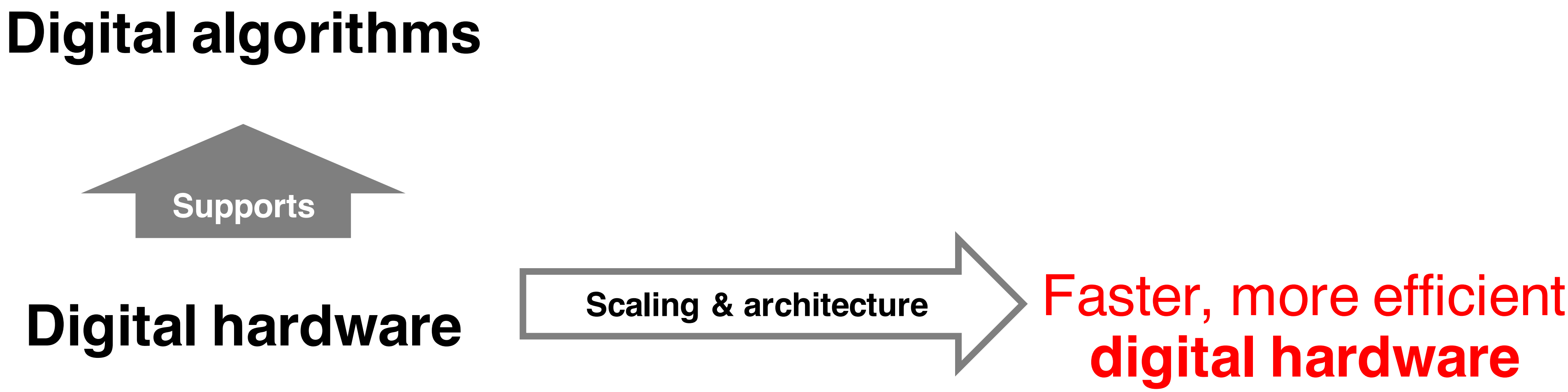
Why Analog?

Digital algorithms

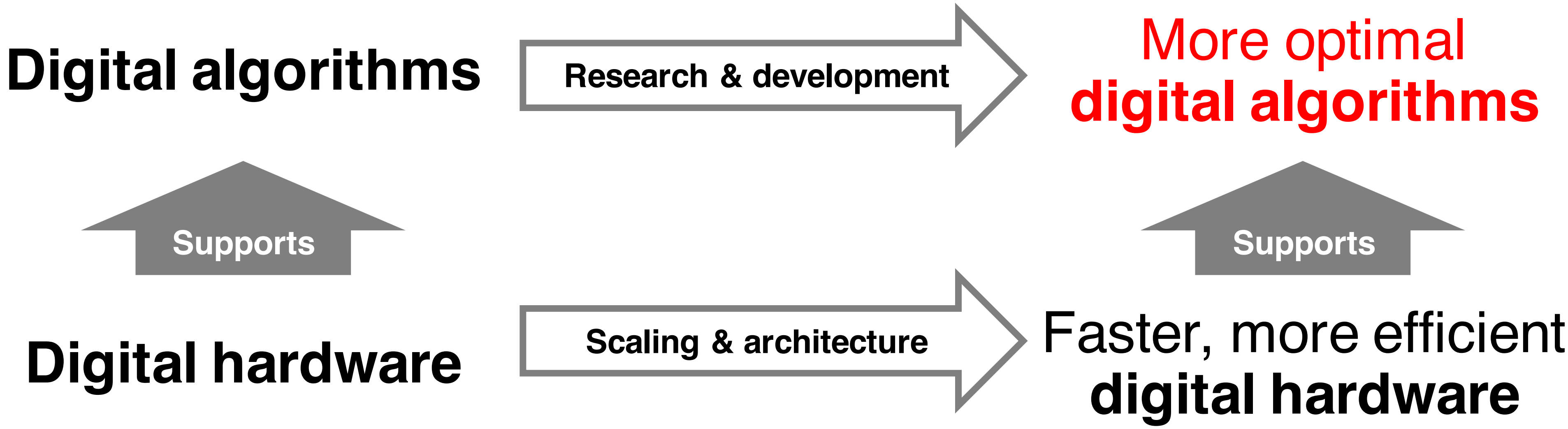
Digital hardware

- **Binary numbers**
- **Step-by-step operation**

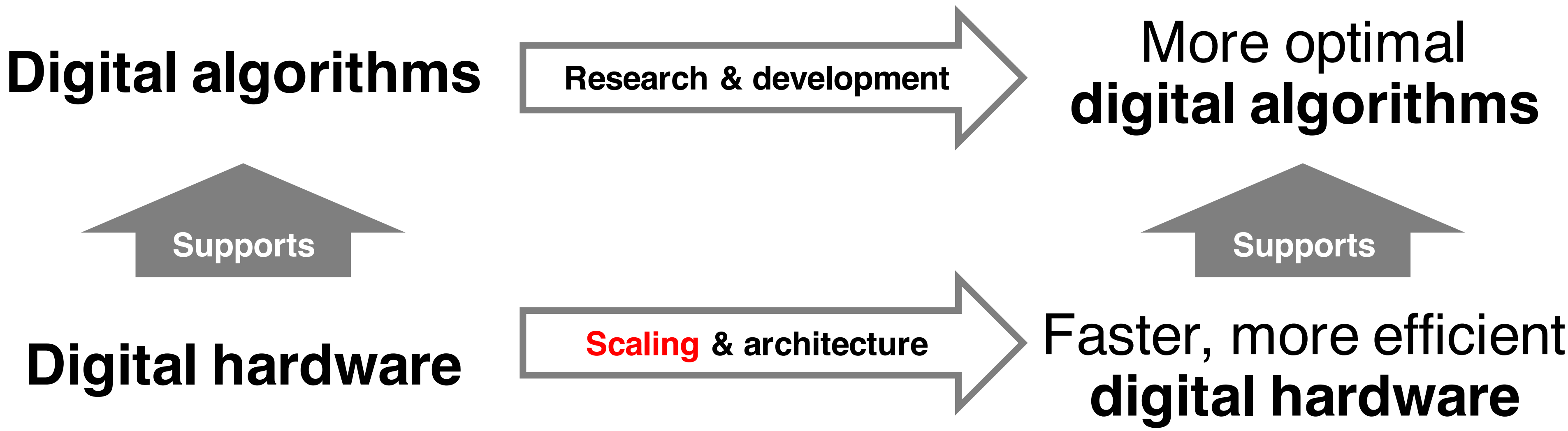
Why Analog?


Digital algorithms
Supports
Digital hardware
Scaling & architecture
Faster, more efficient
digital hardware

Why Analog?


Digital algorithms
Research & development
More optimal
digital algorithms
Supports
Supports
Digital hardware
Scaling & architecture
Faster, more efficient
digital hardware

Why Analog?

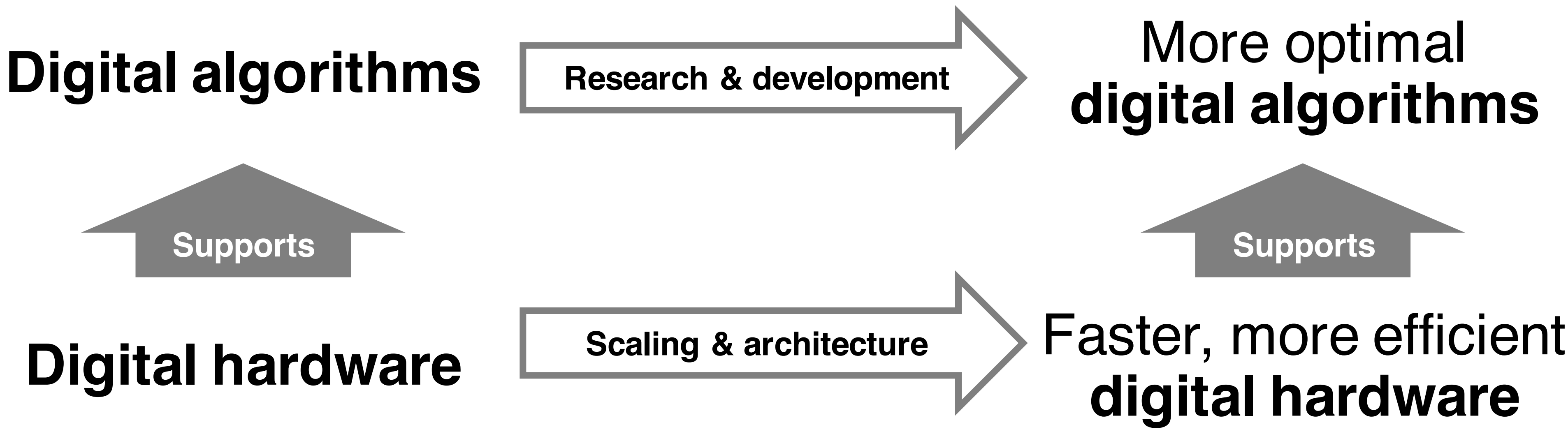


Gates' Simplification #1:

Dennard scaling ended, Moore scaling will end

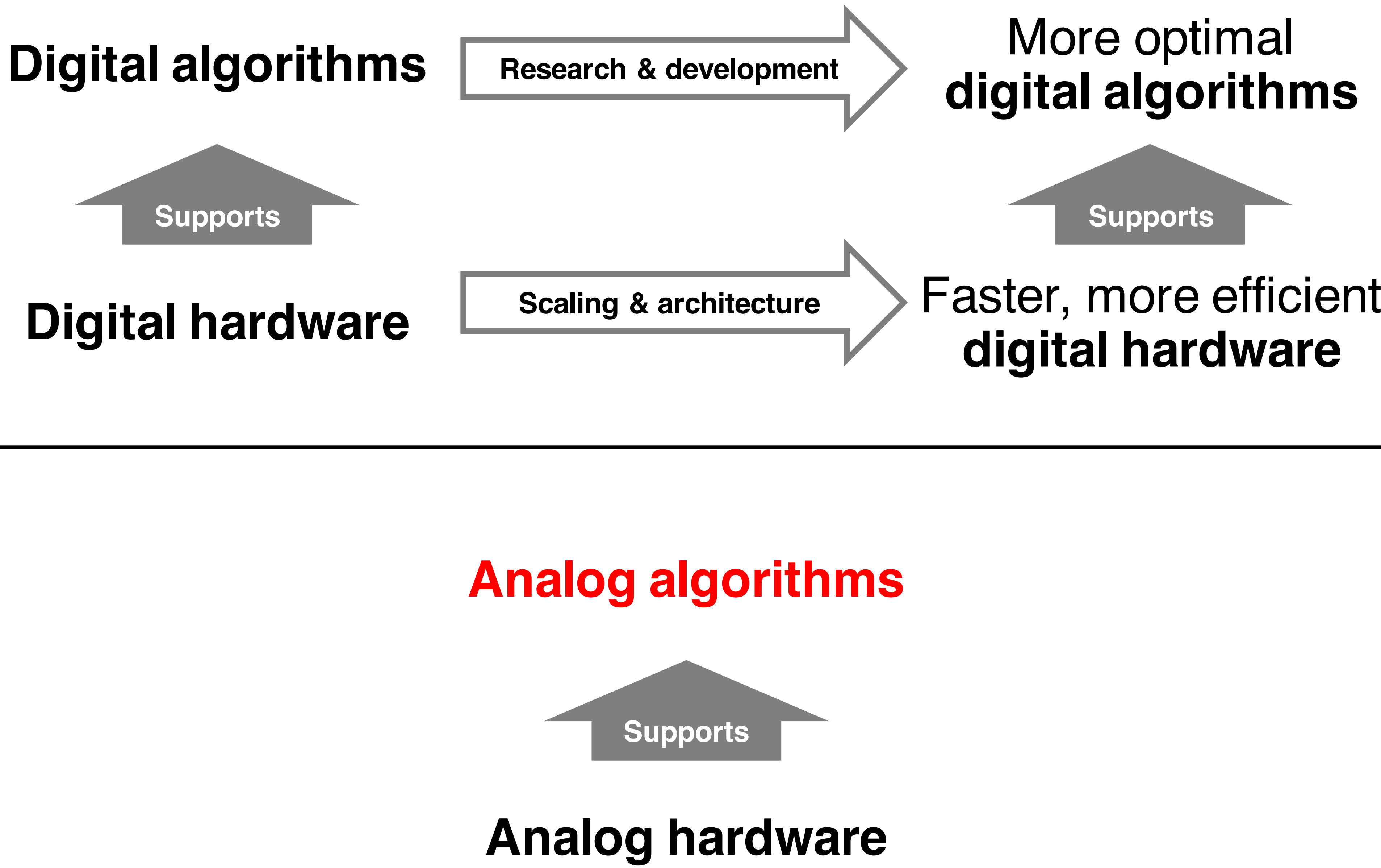
—Monday keynote speaker Doug Carmean

Why Analog?

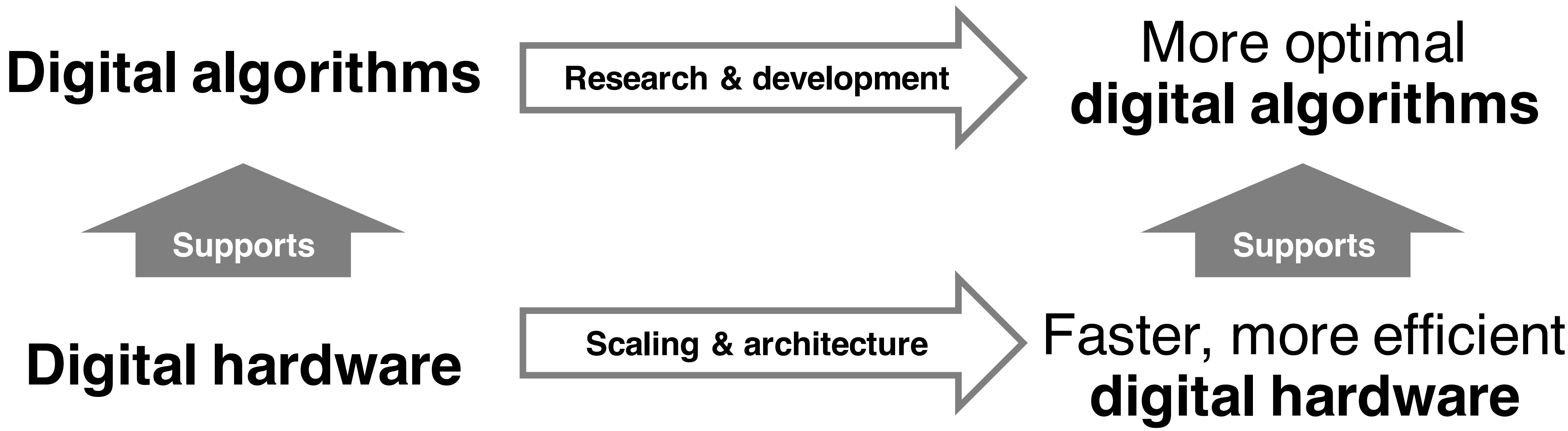


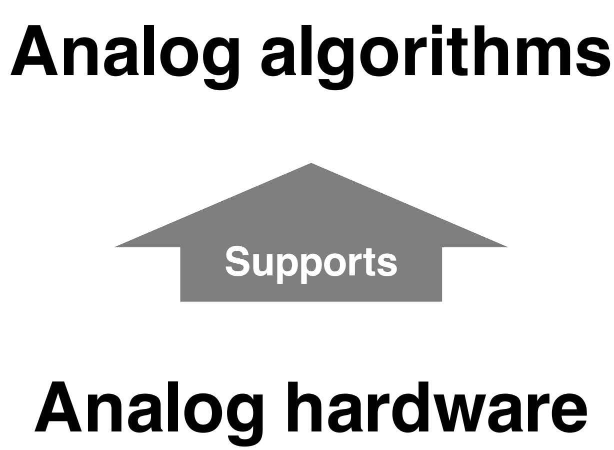
Analog hardware

Why Analog?


Digital algorithms
Research & development
More optimal
digital algorithms
Supports
Supports
Digital hardware
Scaling & architecture
Faster, more efficient
digital hardware
Analog algorithms
Supports
Analog hardware

Why Analog?





- **No binary numbers**
- **Continuous operation**

A continuous-time, analog computing model

- step-by-step algorithm → continuous-time algorithm
- continuous-time algorithm → analog accelerator hardware

Analog drawbacks: how to fix them

A prototype analog accelerator & evaluation

Continuous-time algorithm

Analog computing solves ordinary differential equations

Scientific computation phrased problems as ODEs

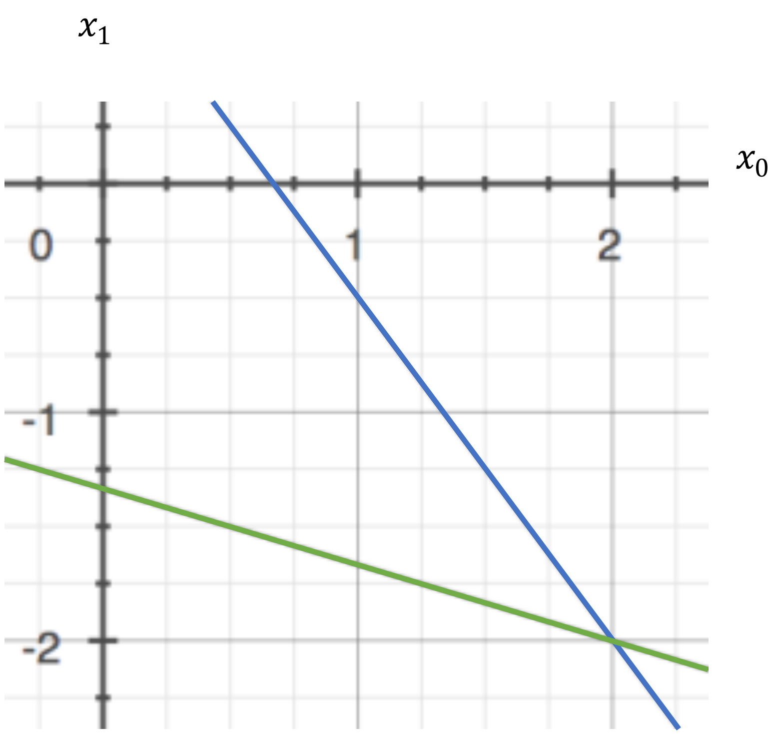
Modern problems are converted to linear algebra, not ODEs

Can we accelerate linear algebra using analog?

$$\boldsymbol{Ax} = \boldsymbol{b}$$

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

$$\begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$$

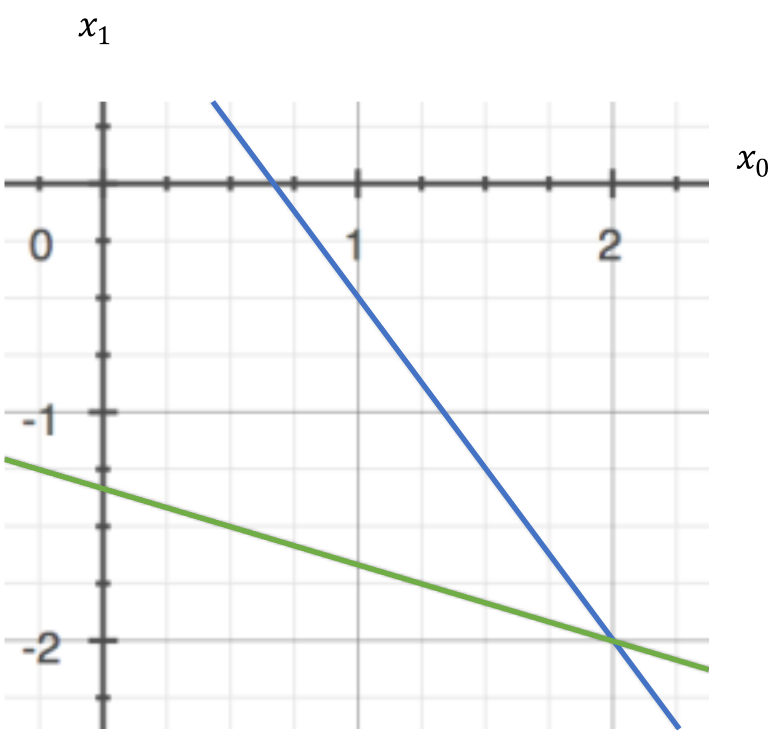


$$\boldsymbol{Ax} = \boldsymbol{b}$$

$$\begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$$

Direct methods

- E.g., Gaussian elimination



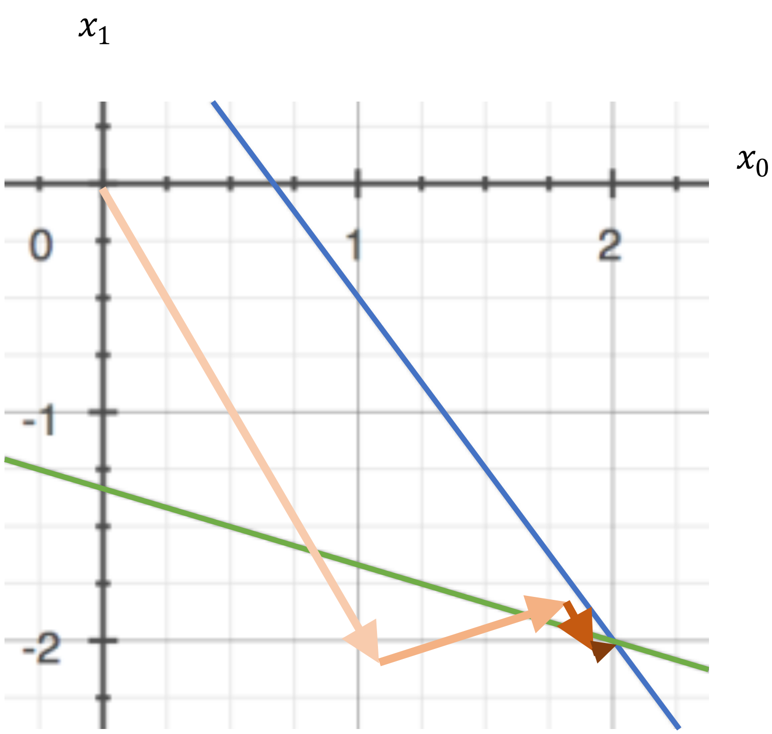
Iterative methods

$$\boldsymbol{Ax} = \boldsymbol{b}$$

$$\begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$$

Direct methods

- E.g., Gaussian elimination



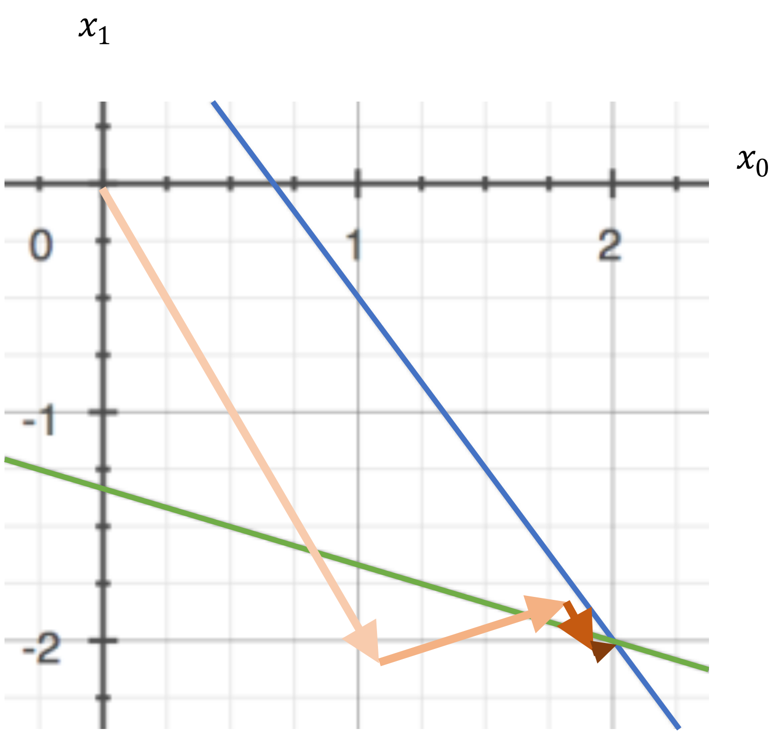
Iterative methods

- E.g., steepest gradient descent
- E.g., conjugate gradients

Solve $\begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$

Step-by-step

steepest gradient descent



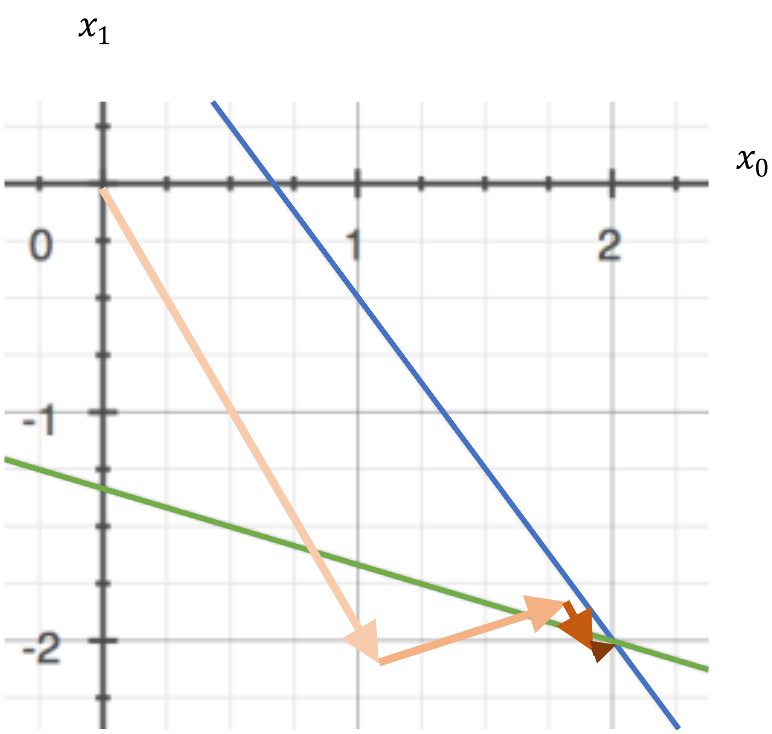
$$\text{Solve} \begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$$

Step-by-step

steepest gradient descent

recurrence relation

$$\begin{cases} x_0^{n+1} = x_0^n - s(a_{00}x_0^n + a_{01}x_1^n - b_0) \\ x_1^{n+1} = x_1^n - s(a_{10}x_0^n + a_{11}x_1^n - b_1) \end{cases}$$



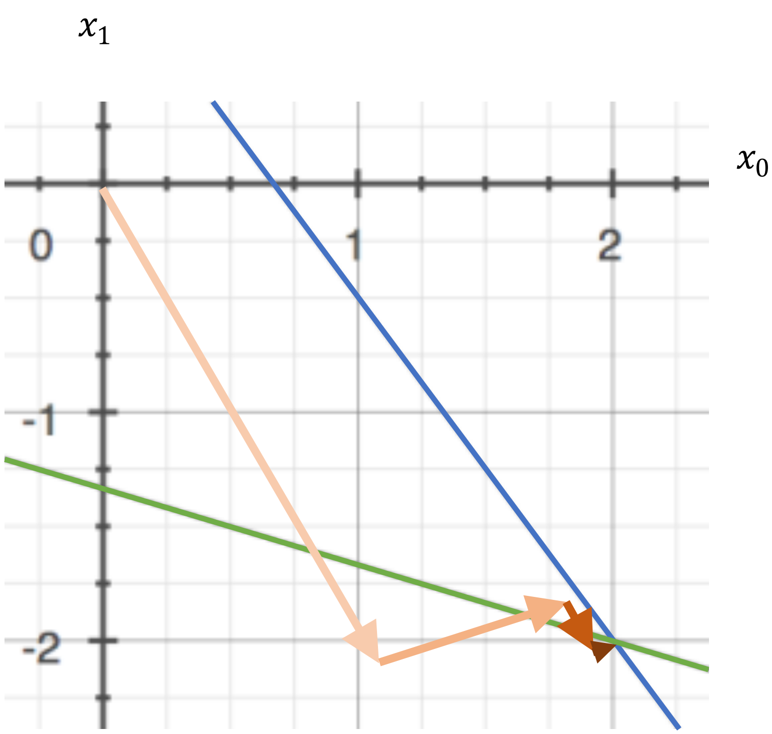
Solve $\begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$

Step-by-step

steepest gradient descent

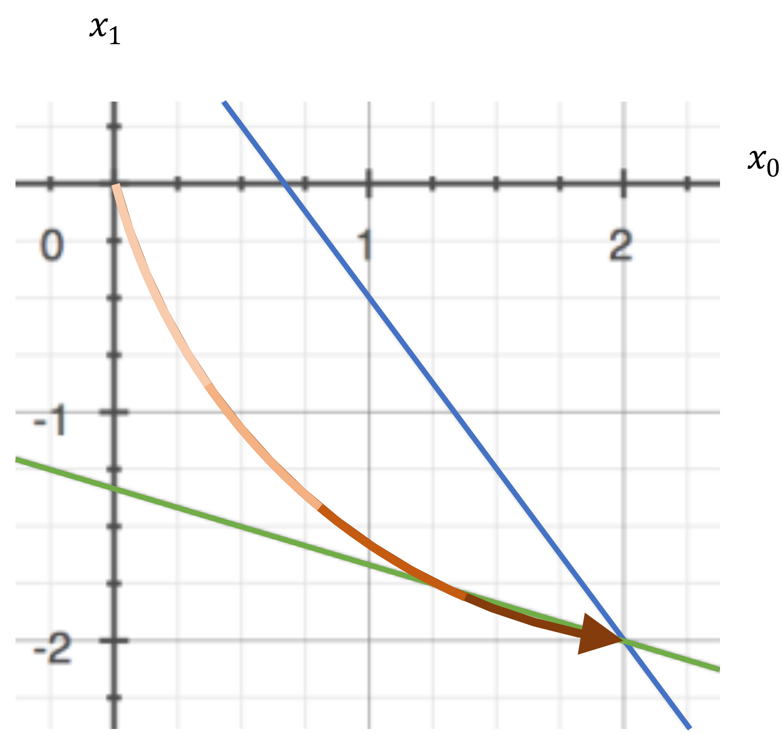
recurrence relation

$$\begin{cases} x_0^{n+1} = x_0^n - s(a_{00}x_0^n + a_{01}x_1^n - b_0) \\ x_1^{n+1} = x_1^n - s(a_{10}x_0^n + a_{11}x_1^n - b_1) \end{cases}$$



Continuous-time

continuous steepest descent



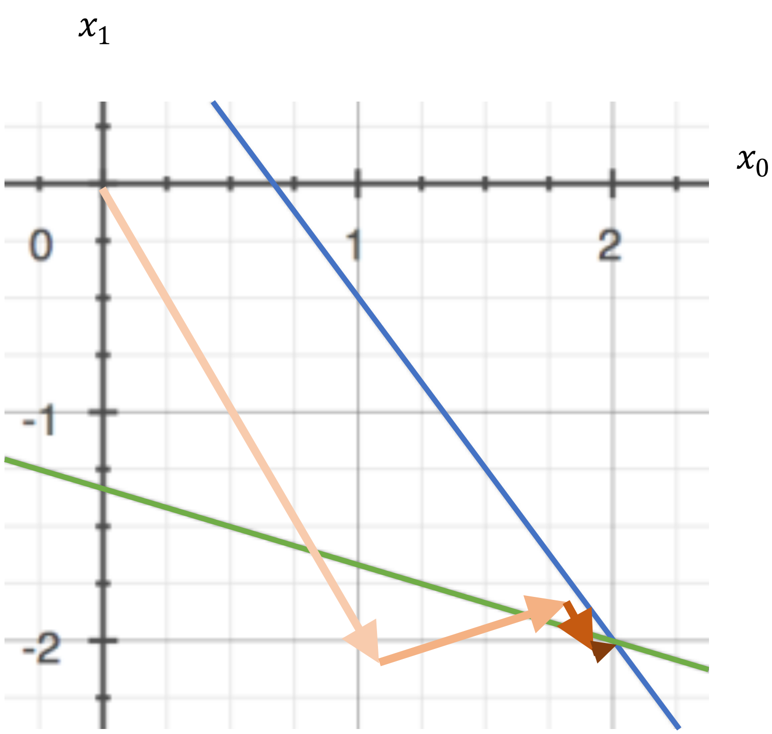
Solve $\begin{cases} a_{00}x_0 + a_{01}x_1 = b_0 \\ a_{10}x_0 + a_{11}x_1 = b_1 \end{cases}$

Step-by-step

steepest gradient descent

recurrence relation

$$\begin{cases} x_0^{n+1} = x_0^n - s(a_{00}x_0^n + a_{01}x_1^n - b_0) \\ x_1^{n+1} = x_1^n - s(a_{10}x_0^n + a_{11}x_1^n - b_1) \end{cases}$$

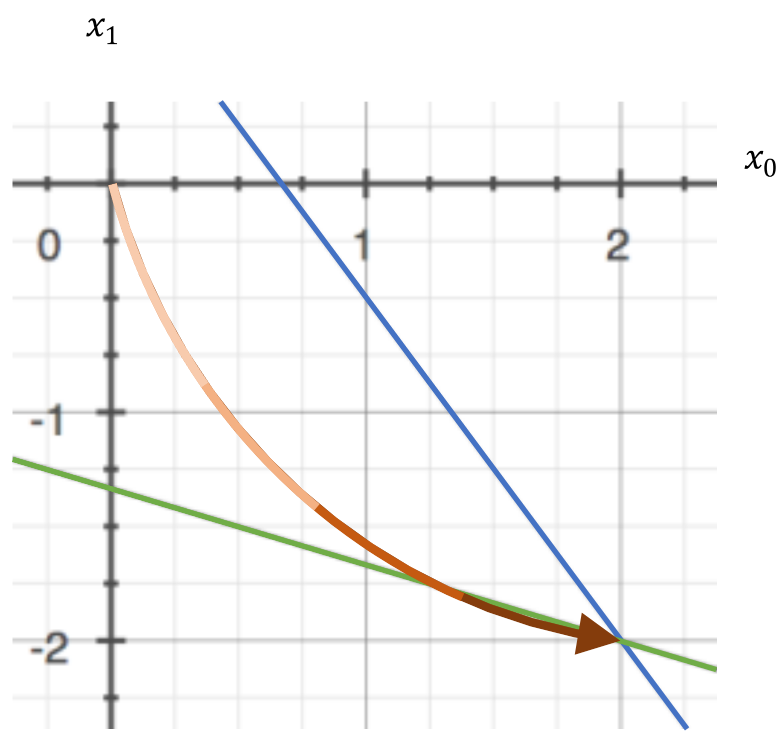


Continuous-time

continuous steepest descent

ordinary differential equation

$$\begin{cases} \mathrm{d}x_0/\mathrm{d}t = -a_{00}x_0 - a_{01}x_1 + b_0 \\ \mathrm{d}x_1/\mathrm{d}t = -a_{10}x_0 - a_{11}x_1 + b_1 \end{cases}$$



A continuous-time, analog computing model

CONTINUOUS-TIME

Potentially fast: not limited by step-by-step algorithm

A continuous-time, analog computing model

- step-by-step algorithm → continuous-time algorithm
- continuous-time algorithm → analog accelerator hardware

Analog drawbacks: how to fix them

A prototype analog accelerator & evaluation

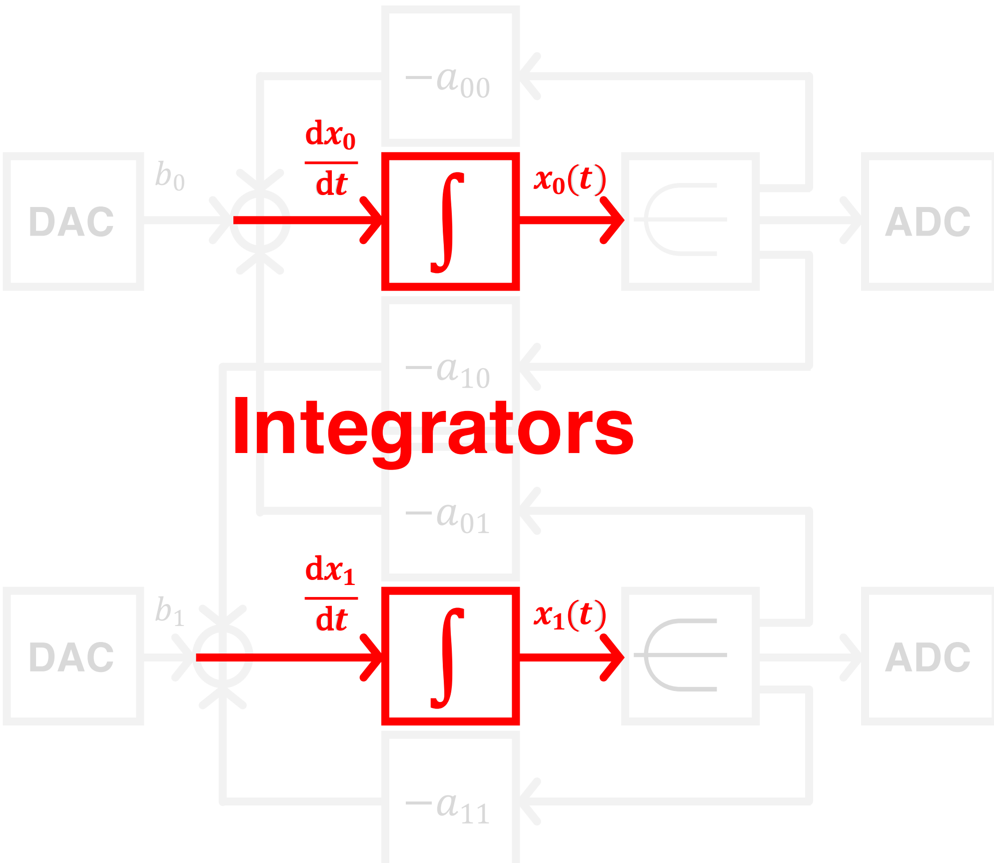
Analog accelerator hardware

Datapath: explicit data flow

Values: represented as analog current & voltage

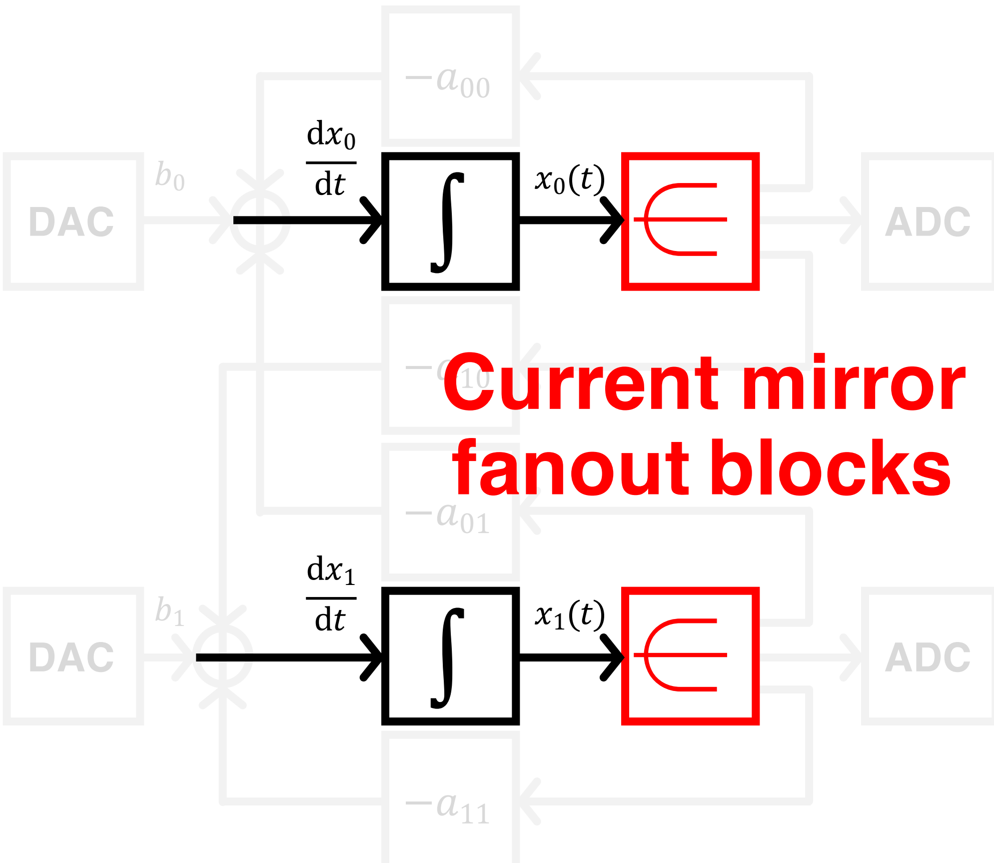
Functional units: analog arithmetic operators

Solve $\begin{cases} a_{00}\boldsymbol{x_0} + a_{01}\boldsymbol{x_1} = b_0 \\ a_{10}\boldsymbol{x_0} + a_{11}\boldsymbol{x_1} = b_1 \end{cases}$



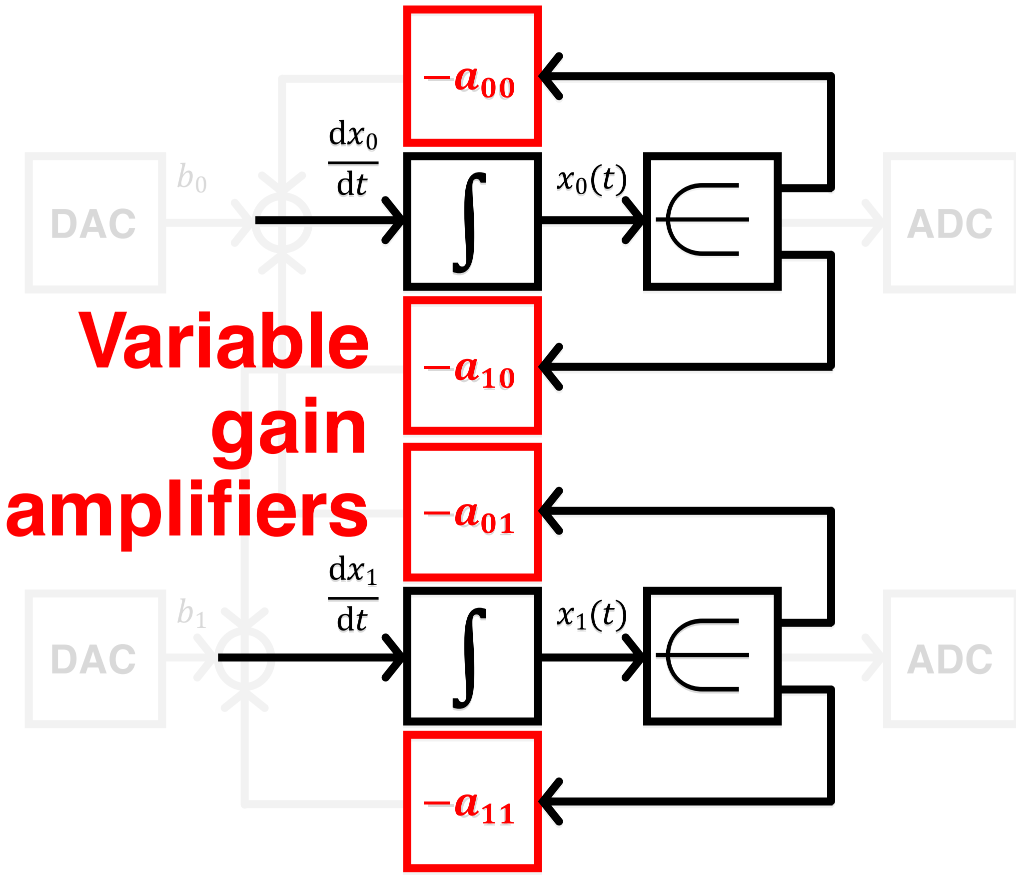
ordinary differential equation

$$\begin{cases} \mathrm{d}\boldsymbol{x_0}/\mathrm{d}t = -a_{00}\boldsymbol{x_0} - a_{01}\boldsymbol{x_1} + b_0 \\ \mathrm{d}\boldsymbol{x_1}/\mathrm{d}t = -a_{10}\boldsymbol{x_0} - a_{11}\boldsymbol{x_1} + b_1 \end{cases}$$



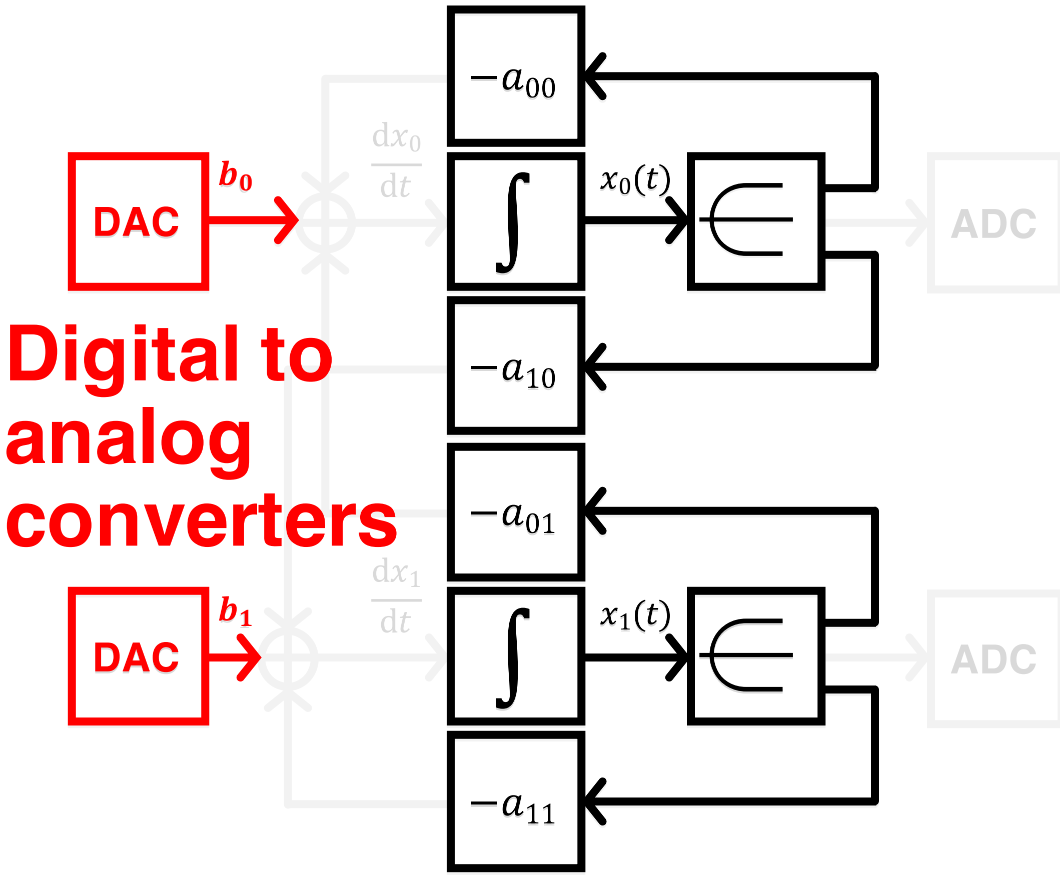
ordinary differential equation

$$\begin{cases} dx_0/dt = -a_{00}x_0 - a_{01}x_1 + b_0 \\ dx_1/dt = -a_{10}x_0 - a_{11}x_1 + b_1 \end{cases}$$



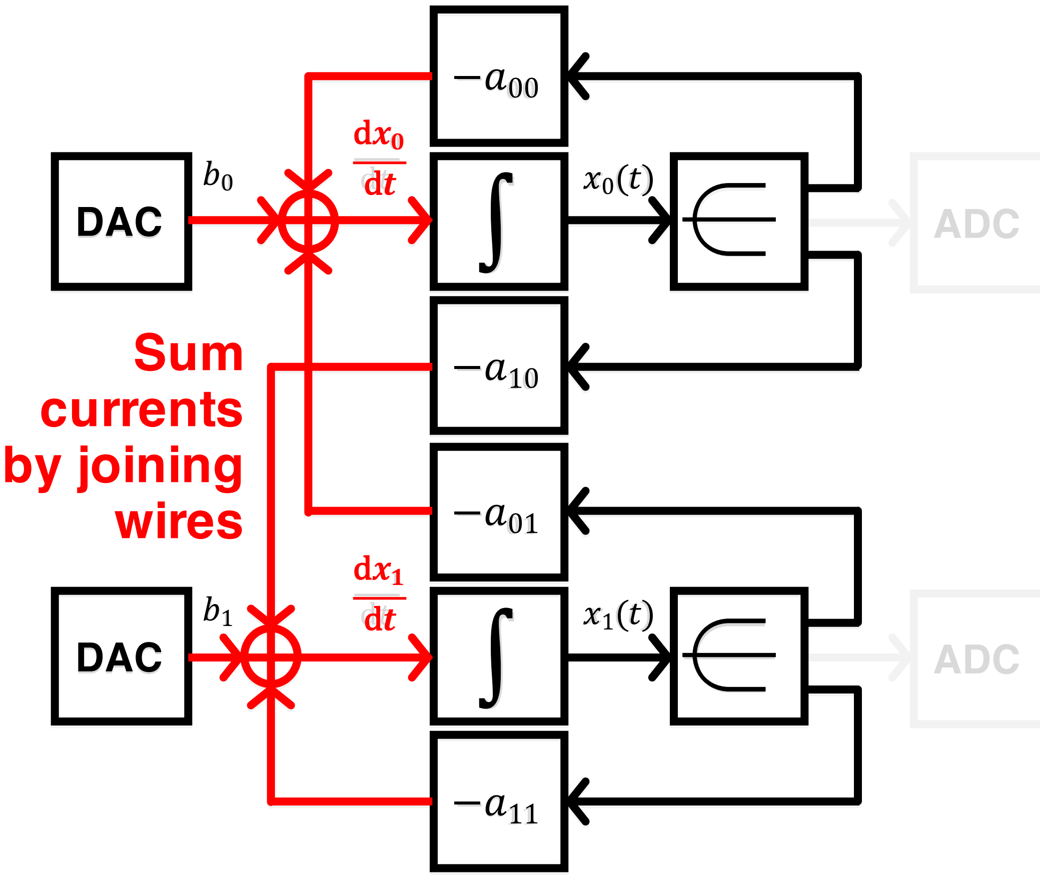
ordinary differential equation

$$\begin{cases} dx_0/dt = -a_{00}x_0 - a_{01}x_1 + b_0 \\ dx_1/dt = -a_{10}x_0 - a_{11}x_1 + b_1 \end{cases}$$



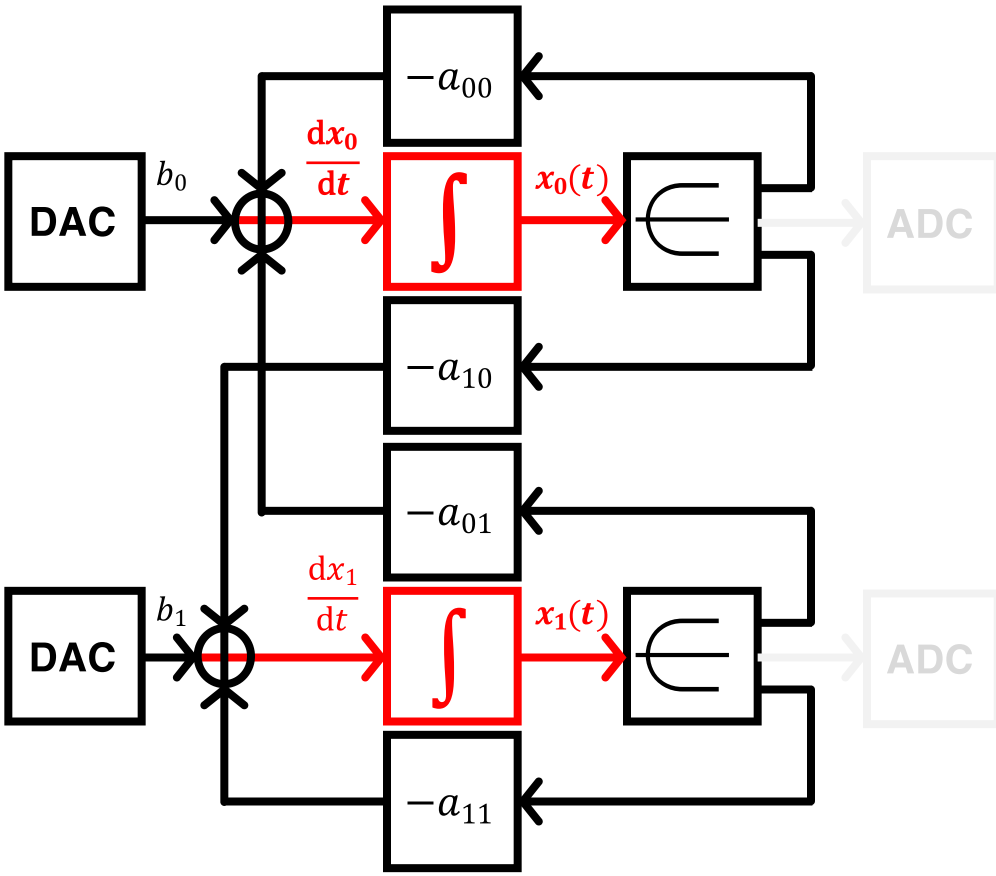
ordinary differential equation

$$\begin{cases} dx_0/dt = -a_{00}x_0 - a_{01}x_1 + \mathbf{b_0} \\ dx_1/dt = -a_{10}x_0 - a_{11}x_1 + \mathbf{b_1} \end{cases}$$



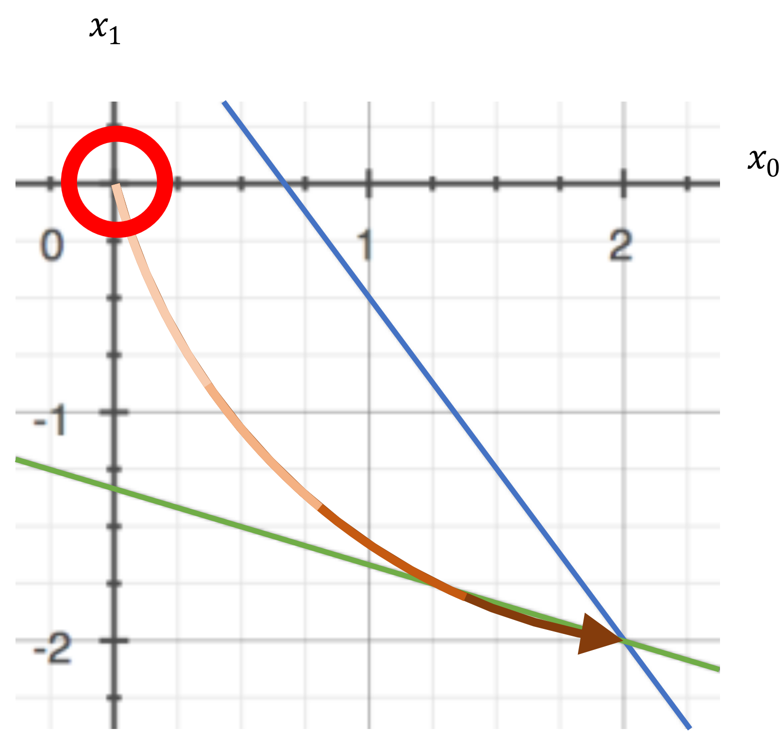
ordinary differential equation

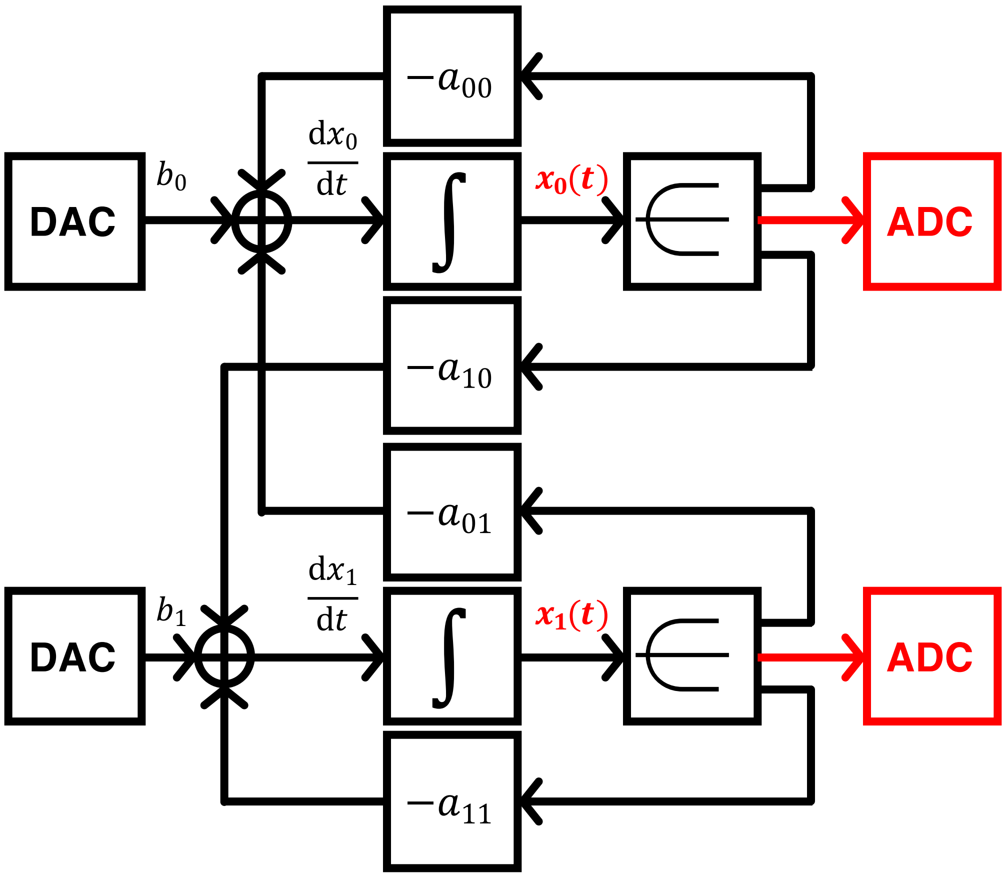
$$\begin{cases} dx_0/dt = -a_{00}x_0 - a_{01}x_1 + b_0 \\ dx_1/dt = -a_{10}x_0 - a_{11}x_1 + b_1 \end{cases}$$



ordinary differential equation

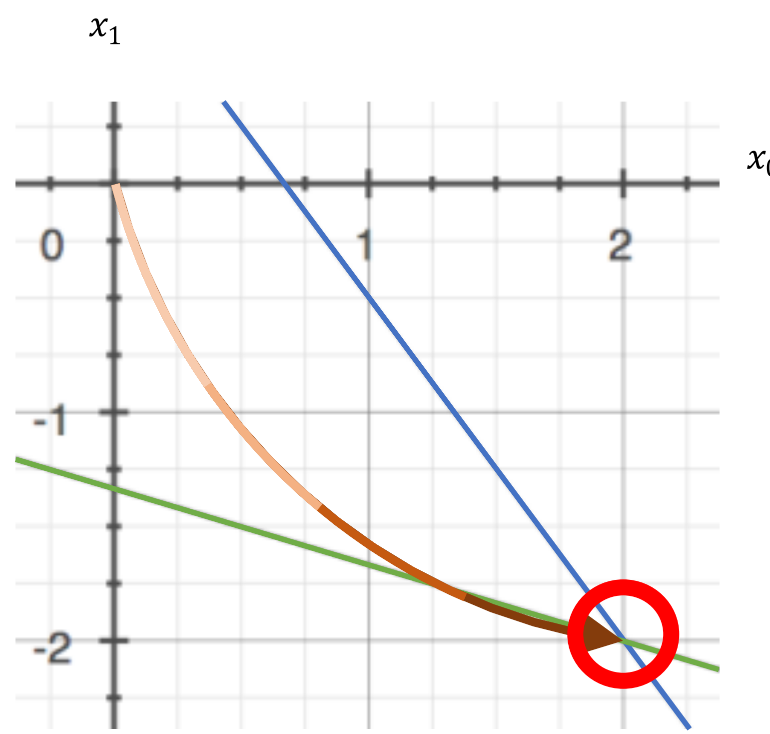
$$\begin{cases} dx_0/dt = -a_{00}x_0 - a_{01}x_1 + b_0 \\ dx_1/dt = -a_{10}x_0 - a_{11}x_1 + b_1 \end{cases}$$



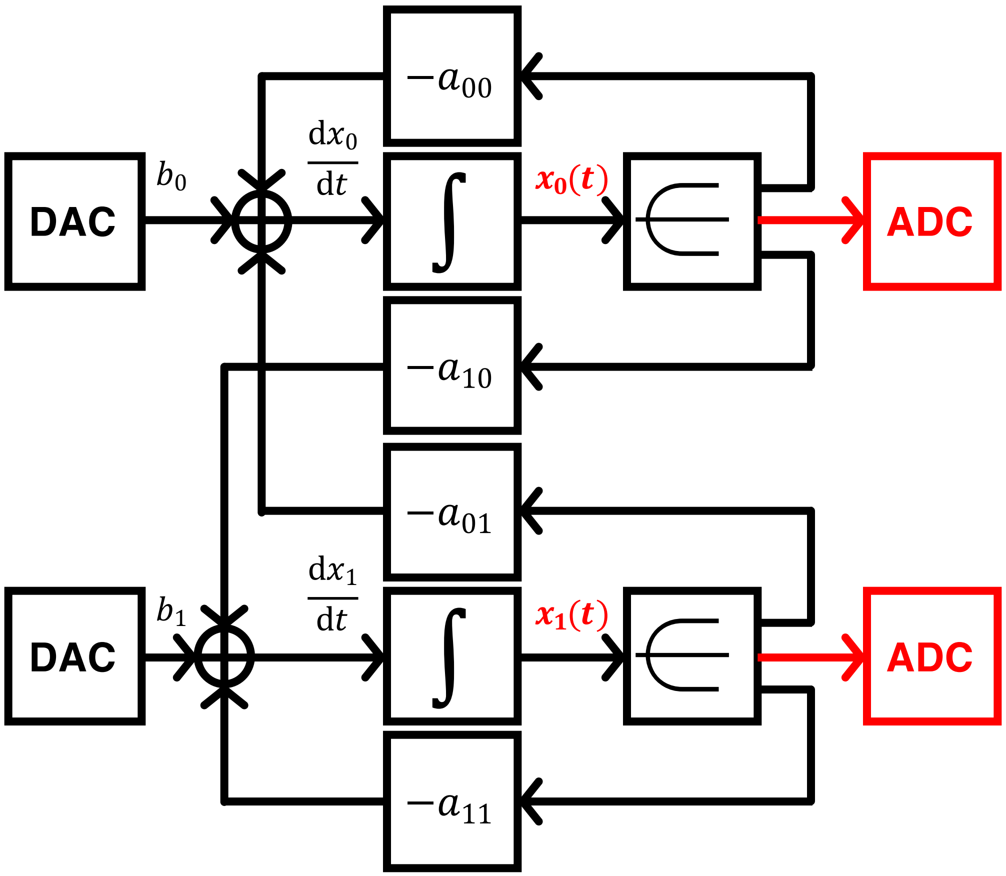


ordinary differential equation

$$\begin{cases} dx_0/dt = -a_{00}\boldsymbol{x_0} - a_{01}\boldsymbol{x_1} + b_0 \\ dx_1/dt = -a_{10}\boldsymbol{x_0} - a_{11}\boldsymbol{x_1} + b_1 \end{cases}$$

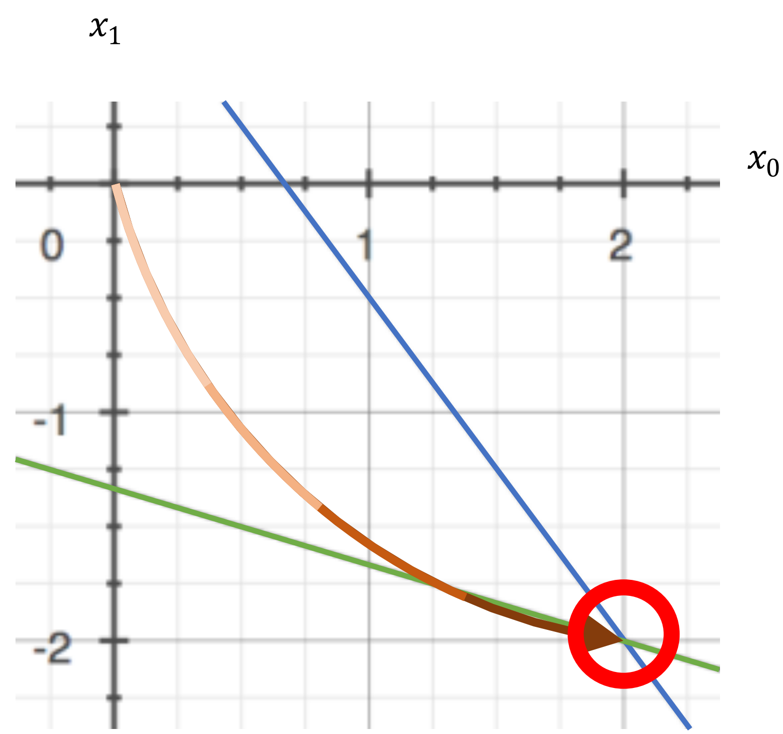


Solve $\begin{cases} a_{00}\boldsymbol{x_0} + a_{01}\boldsymbol{x_1} = b_0 \\ a_{10}\boldsymbol{x_0} + a_{11}\boldsymbol{x_1} = b_1 \end{cases}$



ordinary differential equation

$$\begin{cases} dx_0/dt = -a_{00}\boldsymbol{x_0} - a_{01}\boldsymbol{x_1} + b_0 \\ dx_1/dt = -a_{10}\boldsymbol{x_0} - a_{11}\boldsymbol{x_1} + b_1 \end{cases}$$



A continuous-time, analog computing model

CONTINUOUS-TIME

Potentially fast: not limited by step-by-step algorithm

ANALOG VALUES

Potentially efficient: one wire carries real number

A continuous-time, analog computing model

Analog drawbacks: how to fix them

Analog drawbacks:	how to fix them
• limited applications:	tackle key linear algebra kernel
• limited accuracy:	calibration & exceptions
• limited precision:	build precision with digital help
• limited scalability:	divide & conquere sparse matrix

A prototype analog accelerator & evaluation

Accuracy

Digital	**Analog**
Intermediate values unambiguously interpreted as 1 or 0	Process & temperature variation → computation result variation!

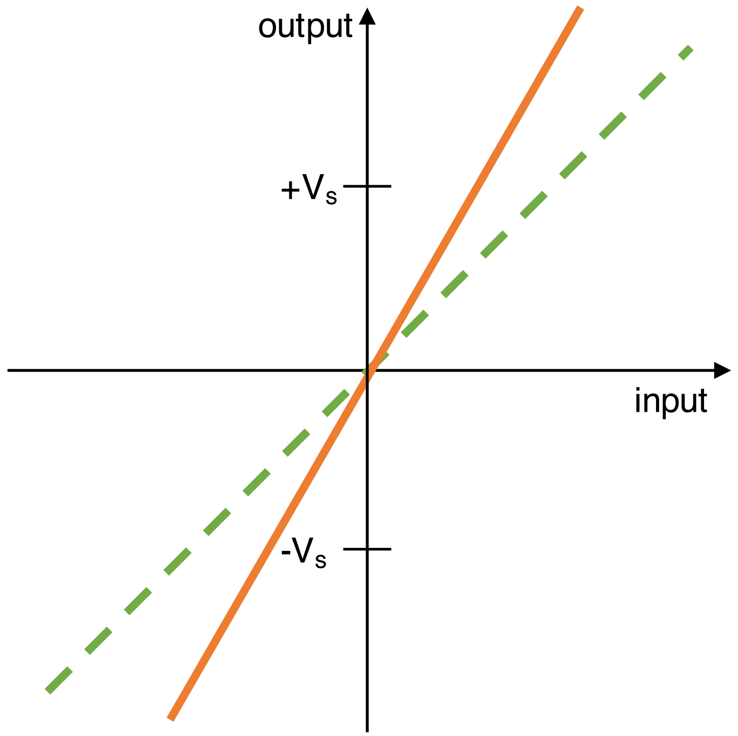
Accuracy

Digital	Analog
Intermediate values unambiguously interpreted as 1 or 0	Process & temperature variation → computation result variation!
Discrete math error correction possible	Within purely analog execution, no error correction

Accuracy: calibration & exceptions

Components are inaccurate:

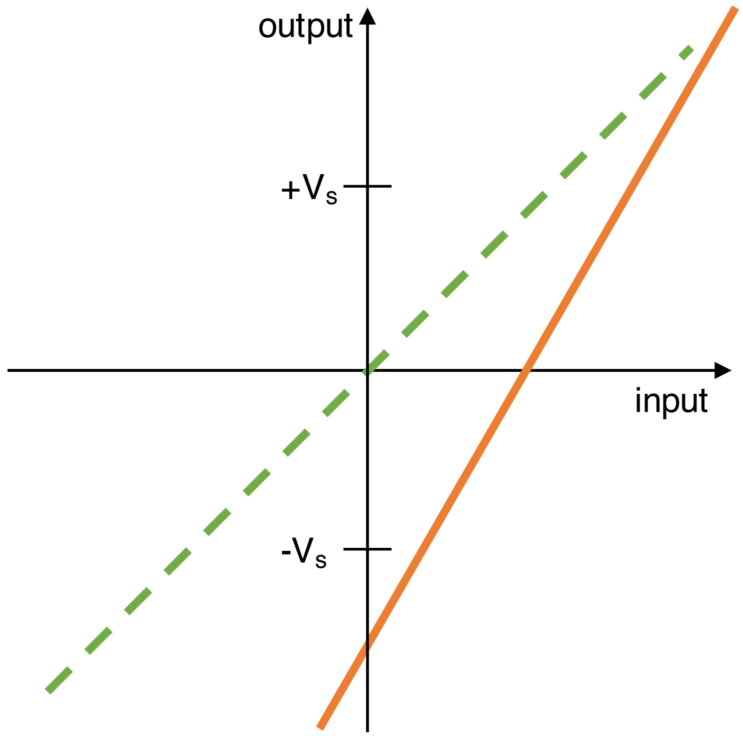
- Gain



Accuracy: calibration & exceptions

Components are inaccurate:

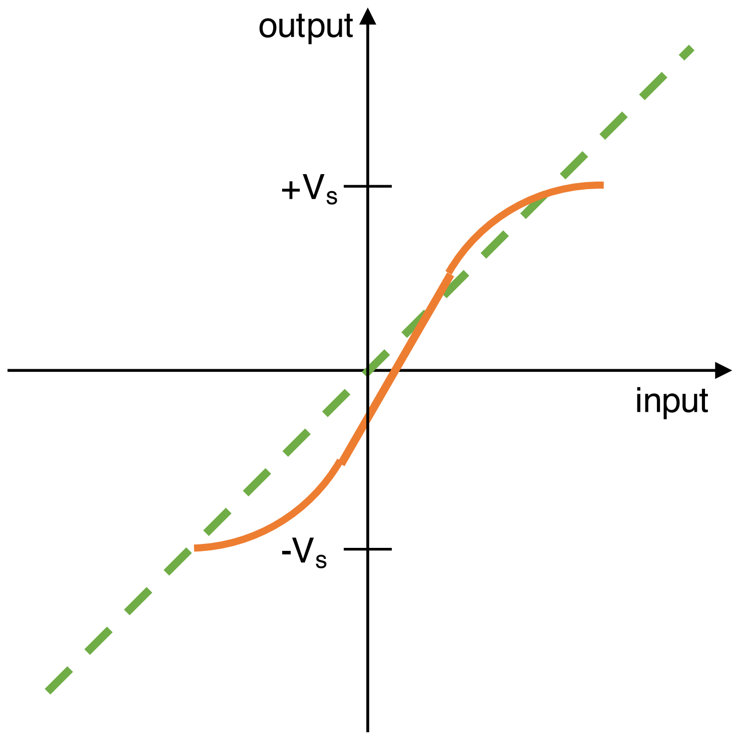
- Gain
- Offset



Accuracy: calibration & exceptions

Components are inaccurate:

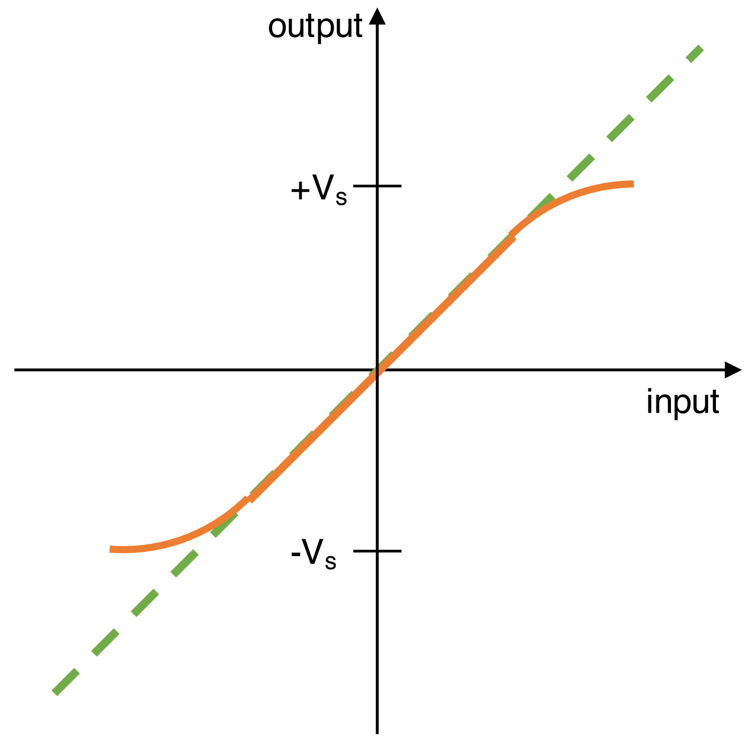
- Gain
- Offset
- Nonlinearity (clipping)



Accuracy: calibration & exceptions

Components are inaccurate:

- Gain
- Offset
- Nonlinearity (clipping)

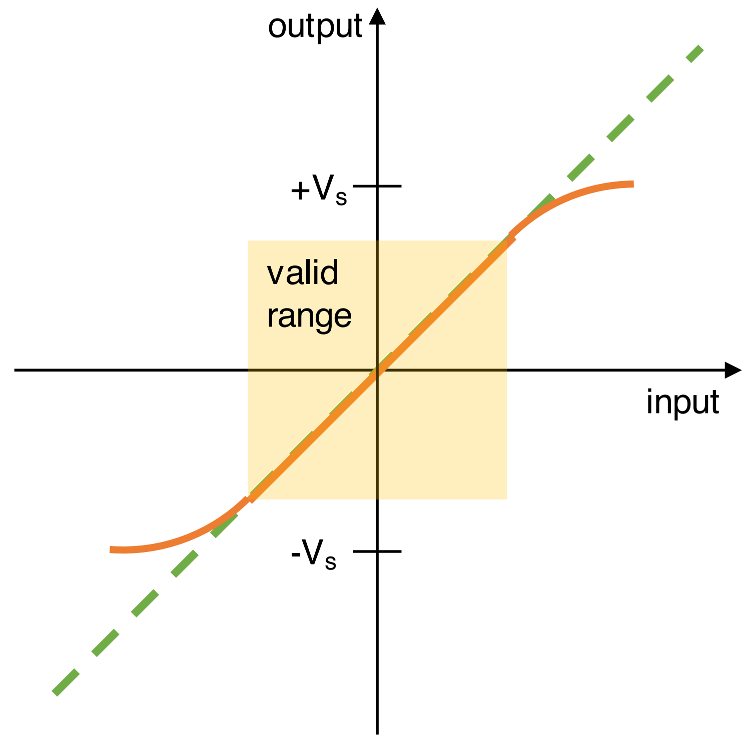


Calibrate: all components using additional DACs

Accuracy: calibration & exceptions

Components are inaccurate:

- Gain
- Offset
- Nonlinearity (clipping)



<u>Calibrate:</u> all components using additional DACs

<u>Exceptions:</u> catch values exceeding valid input range

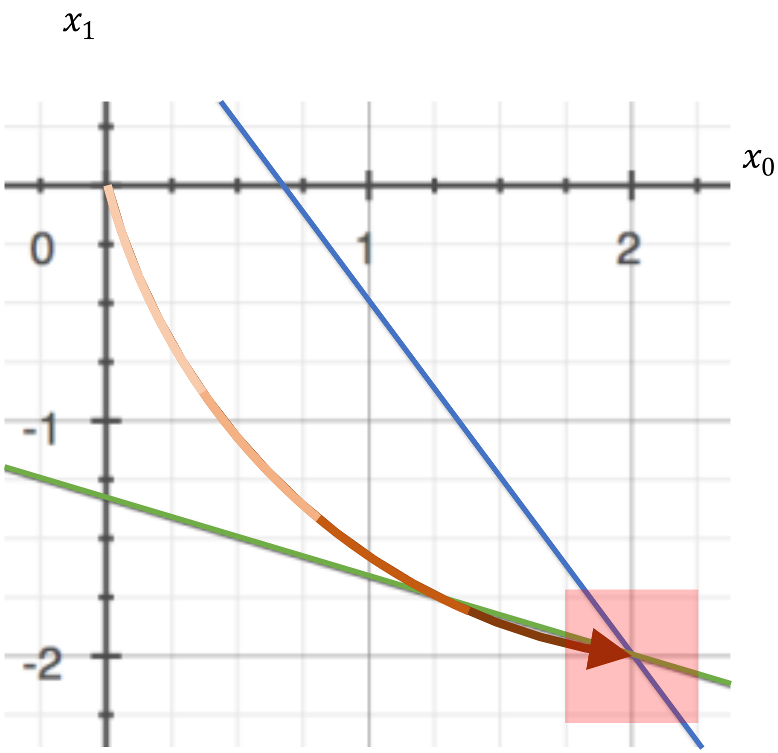
A continuous-time, analog computing model

Analog drawbacks: how to fix them

- limited applications: tackle key linear algebra kernel
- limited accuracy: calibration & exceptions
- limited precision: build precision with digital help
- limited scalability: divide & conquere sparse matrix

A prototype analog accelerator & evaluation

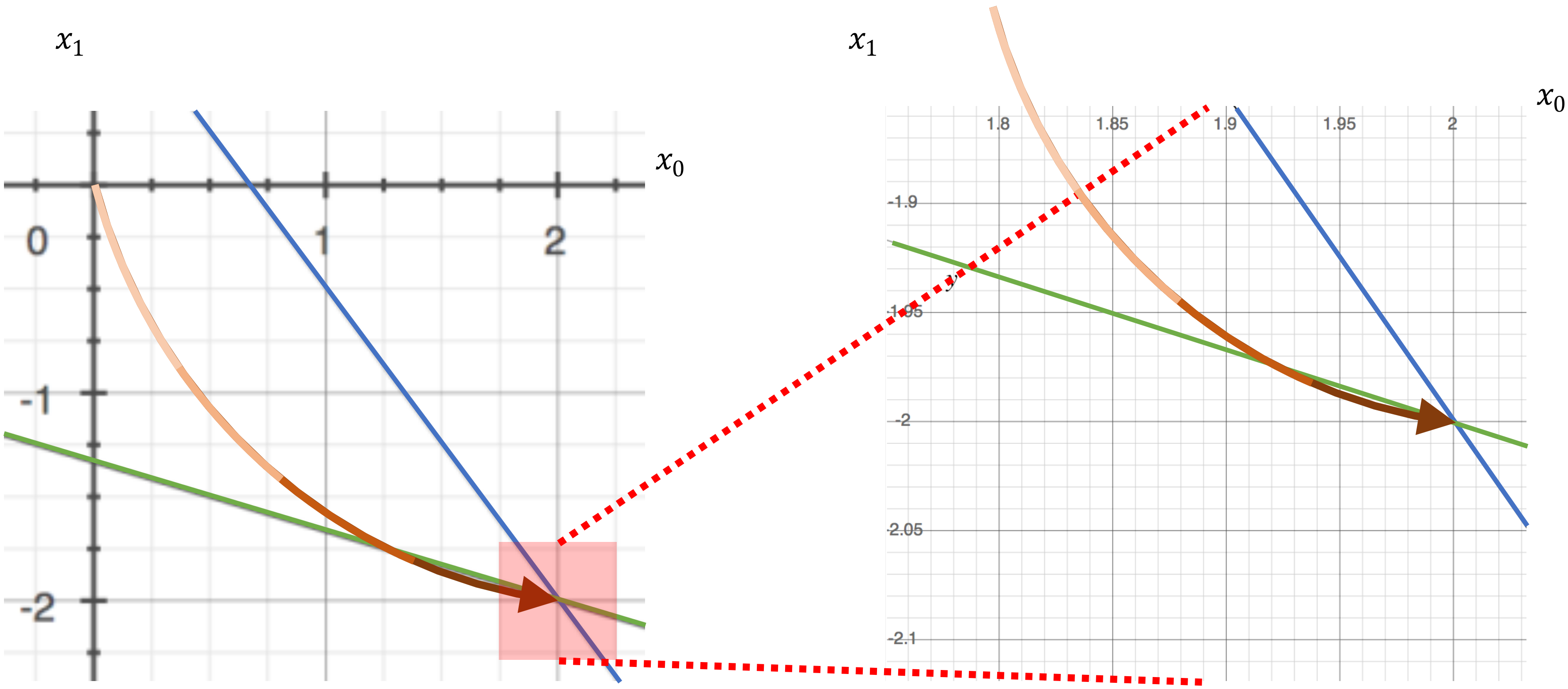
Precision: build precision w/ digital help



Limitation in sampling resolution

→ limited precision result

Precision: build precision w/ digital help



Limitation in sampling resolution
→ limited precision result

Find residual, rescale problem,
& solve again in analog for precision

A continuous-time, analog computing model

Analog drawbacks: how to fix them

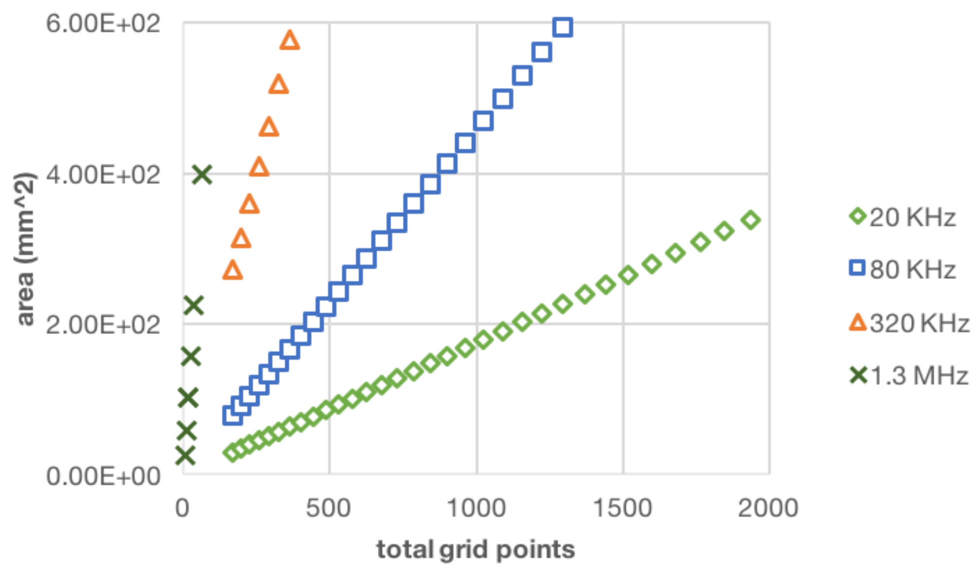
- limited applications: tackle key linear algebra kernel
- limited accuracy: calibration & exceptions
- limited precision: build precision with digital help
- limited scalability: divide & conquere sparse matrix

A prototype analog accelerator & evaluation

Scalability: divide & conquer sparse matrix

Without time multiplexing, analog hardware needed for all variables

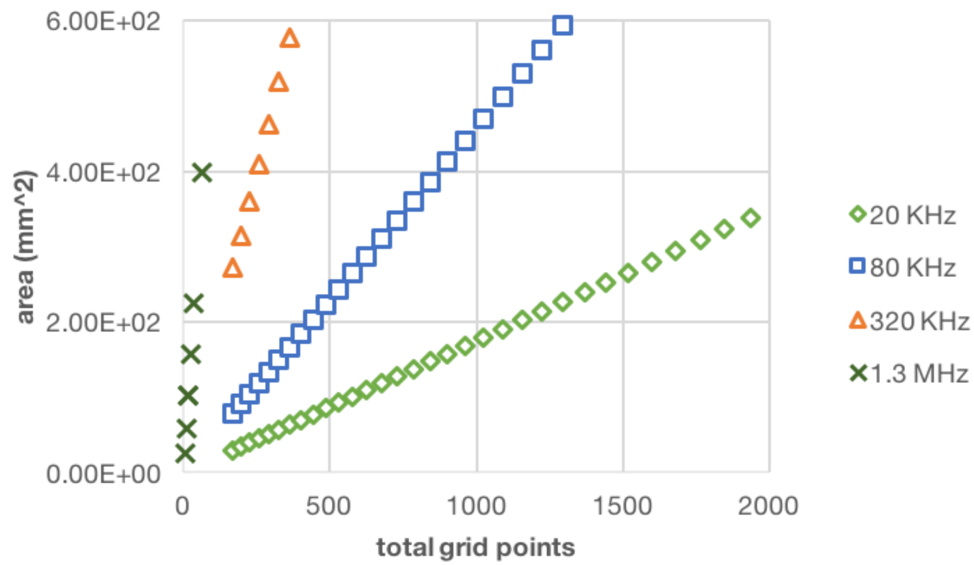
- Impractical to build analog hardware for whole problems



Scalability: divide & conquer sparse matrix

Without time multiplexing, analog hardware needed for all variables

- Impractical to build analog hardware for whole problems



Solve subproblems of smaller size in analog

- Possible because many problems have sparse matrices

$$A\mathbf{u} = \mathbf{b}$$

$$A = \frac{1}{\frac{1}{3^2}} \begin{bmatrix} 4 & -1 & & -1 & & & & & \\ -1 & 4 & -1 & & -1 & & & & \\ & -1 & 4 & & & -1 & & & \\ -1 & & & 4 & -1 & & -1 & & \\ & -1 & & -1 & 4 & -1 & & -1 & \\ & & -1 & & -1 & 4 & & & -1 \\ & & & -1 & & & 4 & -1 & \\ & & & & -1 & & -1 & 4 & -1 \\ & & & & & -1 & & -1 & 4 \end{bmatrix}$$

$$\mathbf{u} = [u_0, u_1, \ldots, u_8]^\top, \mathbf{b} = [b_0, b_1, \ldots, b_8]^\top$$

A continuous-time, analog computing model

Analog drawbacks: how to fix them

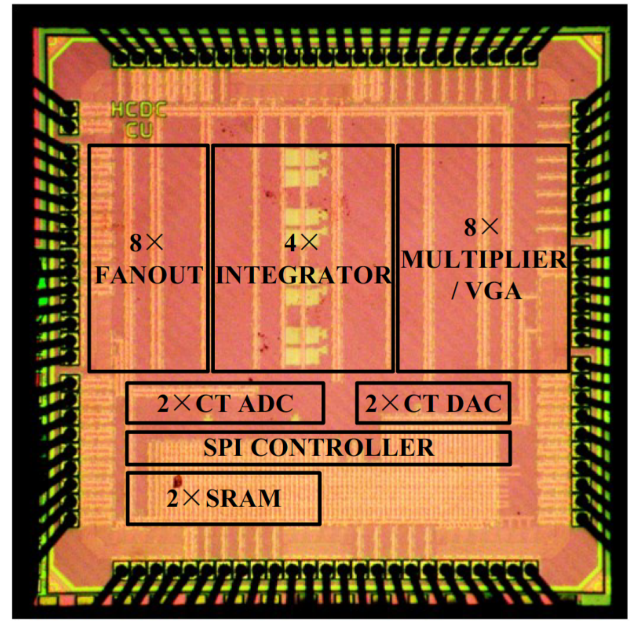
A prototype analog accelerator & evaluation

- microarchitecture
- architecture & programming
- energy
- performance

Prototype analog accelerator: why & how

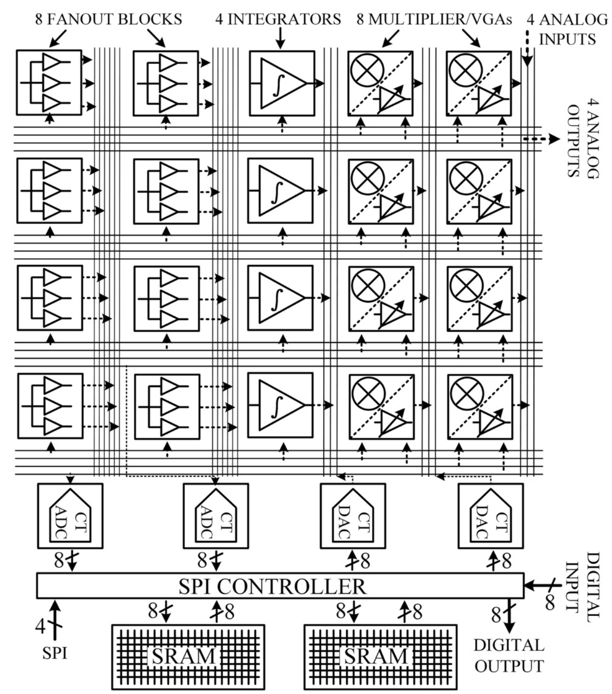
Why prototype?

- validate analog circuits
- physical measurement of components
- prototype helps developing analog applications

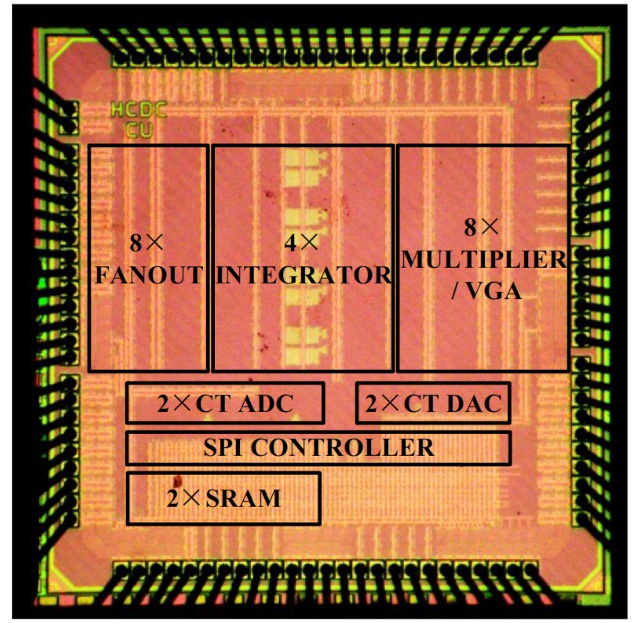


Engineering process

- 2 full-time PhD students, 4 project students
- 3 years
- full custom analog
- synthesized digital
- I worked on validation, digital interface, applications

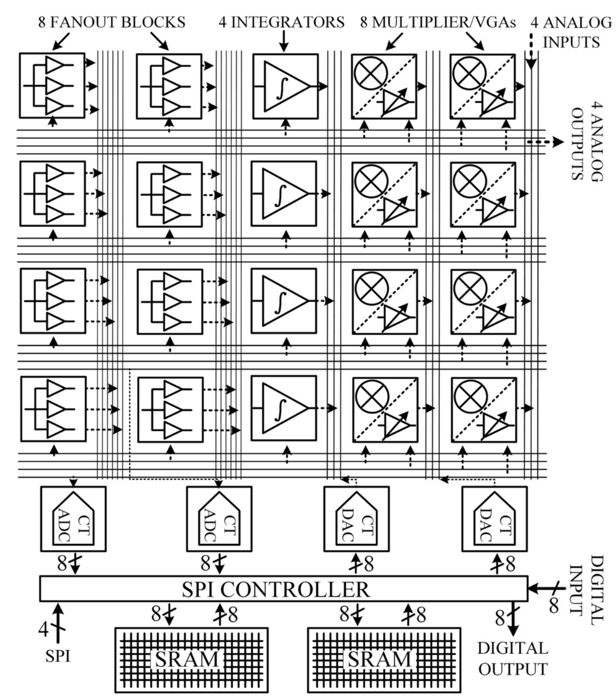


Prototype analog accelerator: µArch.



Components

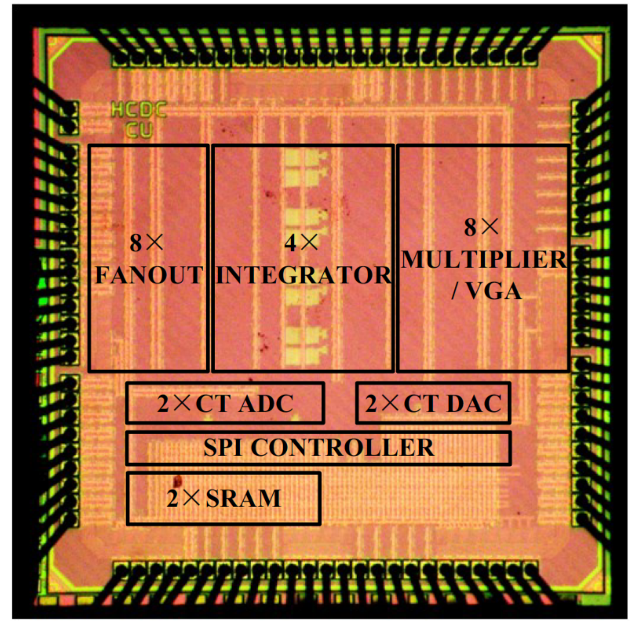
- 20 KHz analog bandwidth
- 4 integrators
- 8 multipliers
- Other features: nonlinear lookup



Fabrication

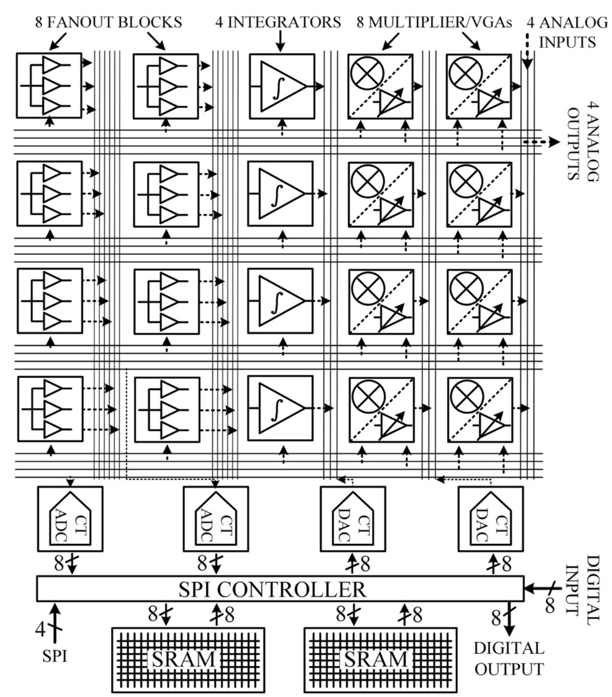
- 1.2 V 65nm TSMC
- Low power density
  - 0.06 W/cm^2 at full power
  - 2.0 mm^2 active area
  - 1.2 mW at full power

Prototype analog accelerator: interface



Architecture

- 8-bit A/D/A conversion
- Configurable analog crossbar
- Calibration & exceptions on all analog



Programming

- Library for configuration
- ODE syntax parser and compiler

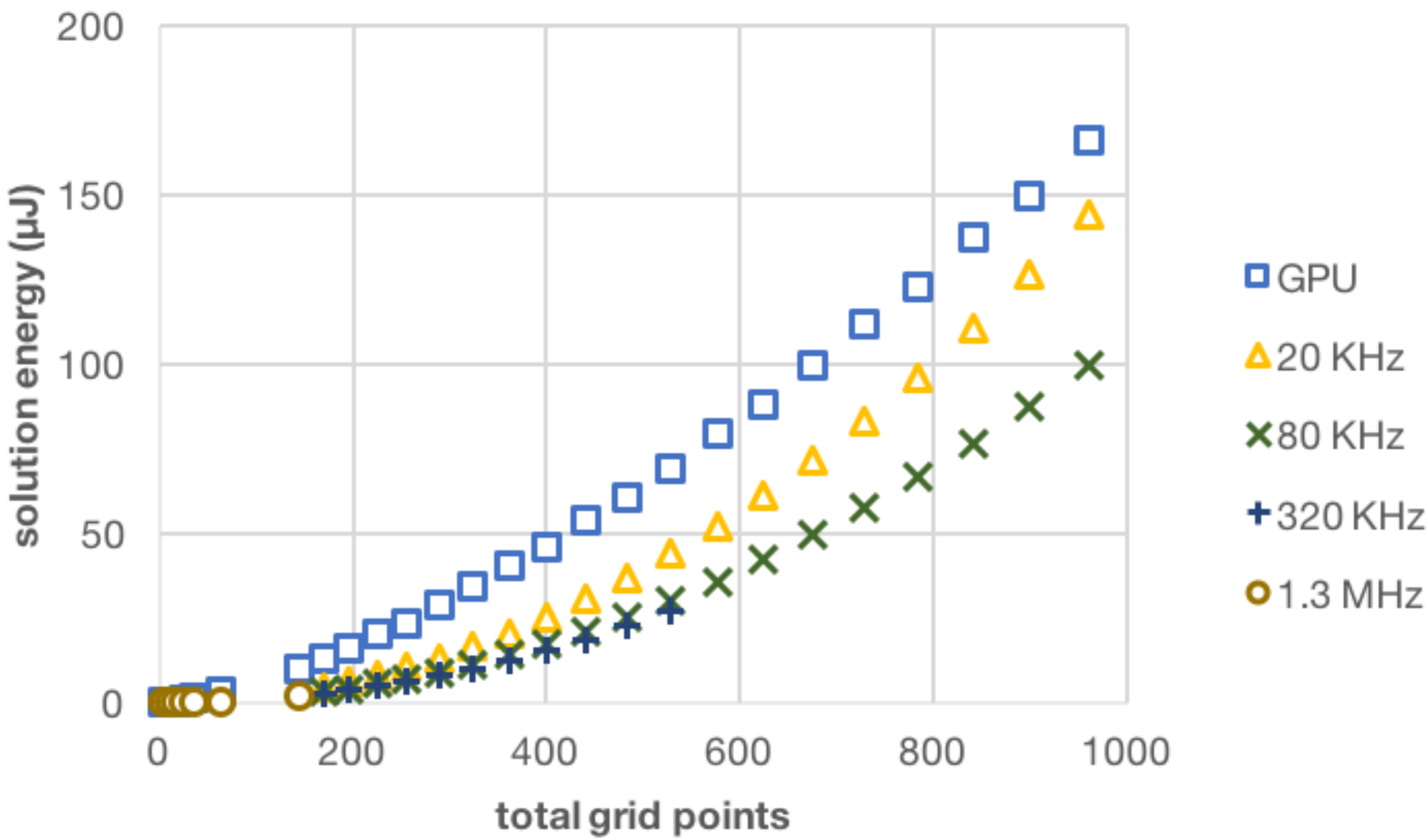
A continuous-time, analog computing model

Analog drawbacks: how to fix them

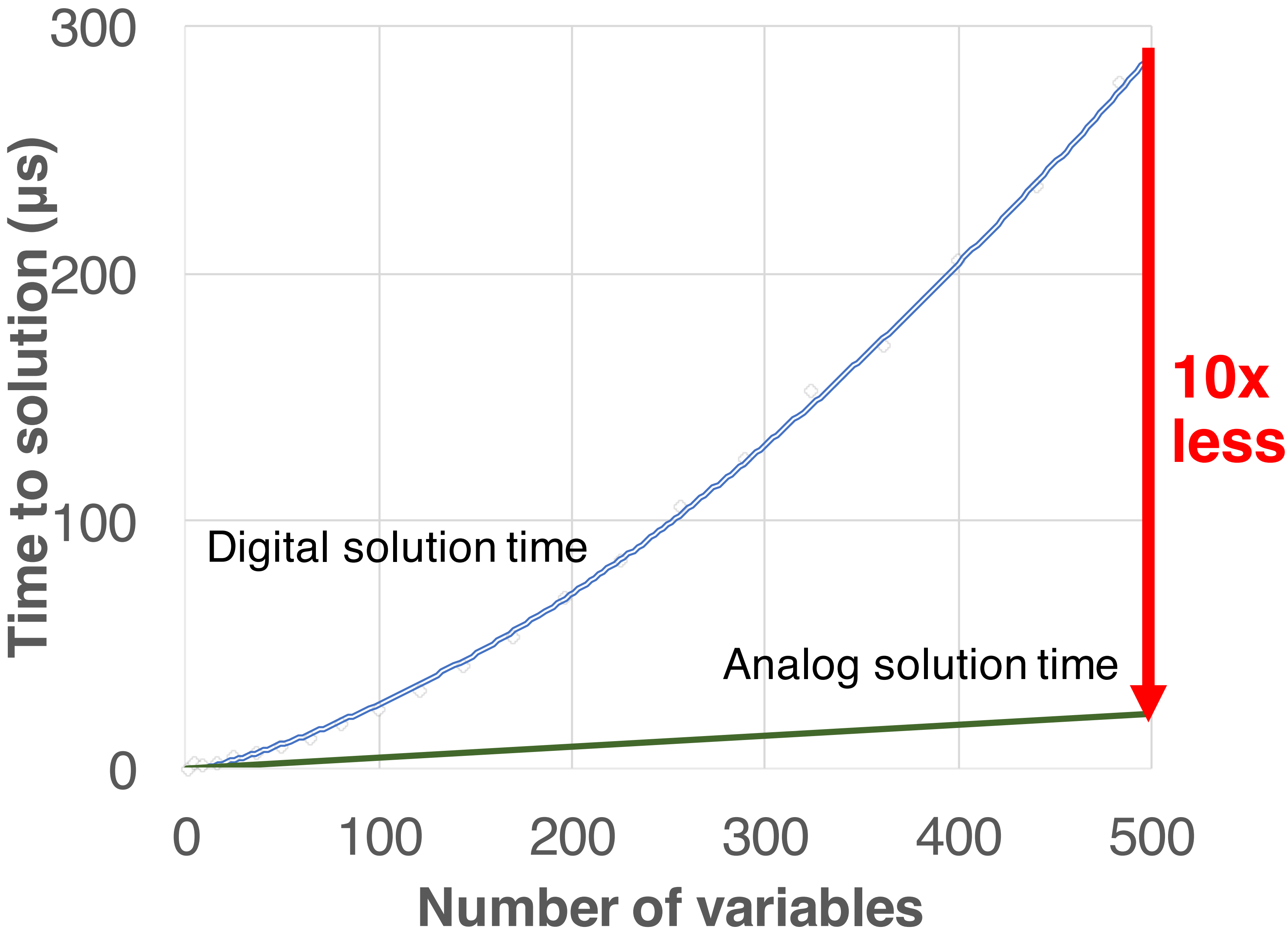
A prototype analog accelerator & evaluation

- microarchitecture
- architecture & programming
- energy
- performance

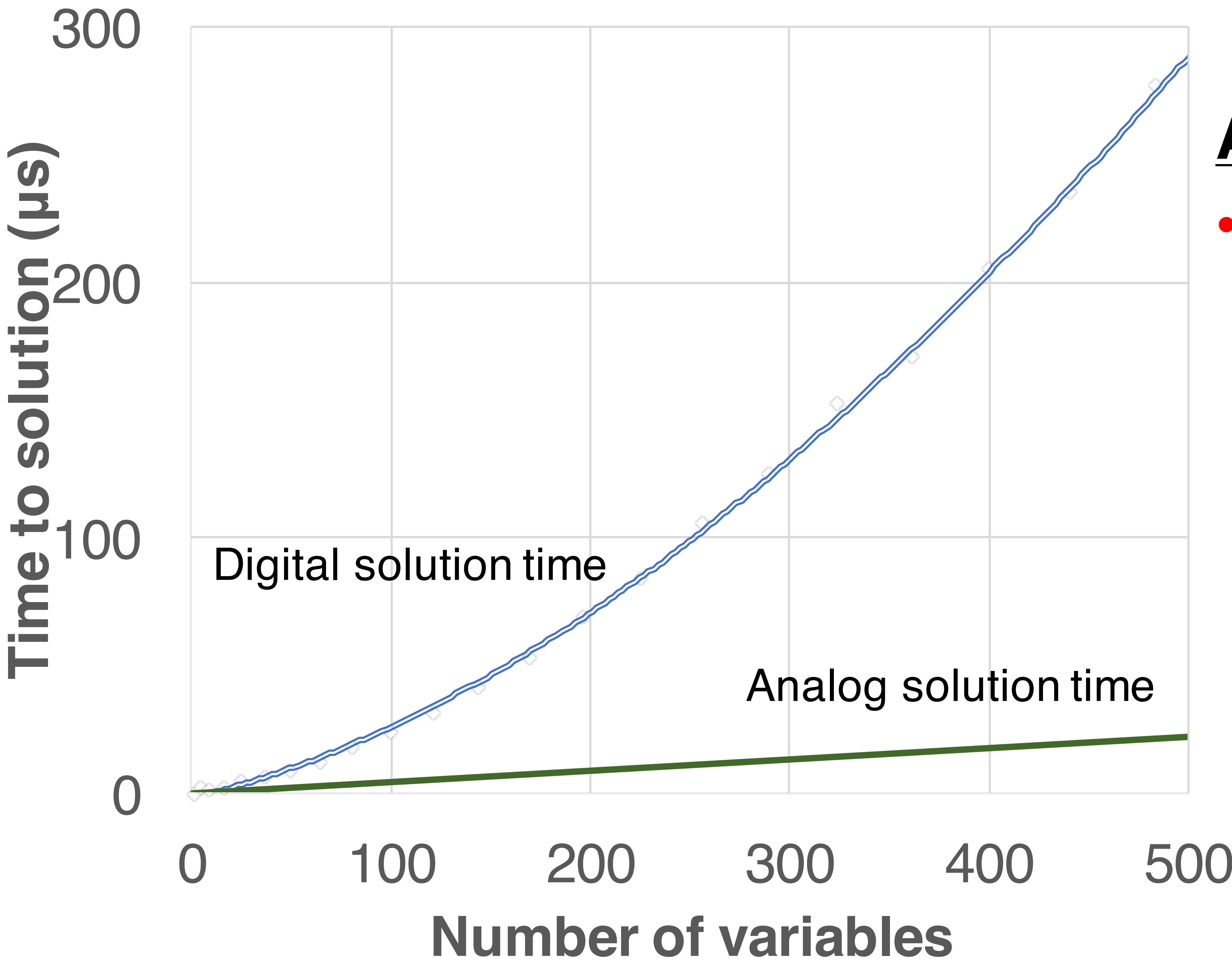
Solution energy: analog HW vs. digital SW



Solution time: analog HW vs. digital SW



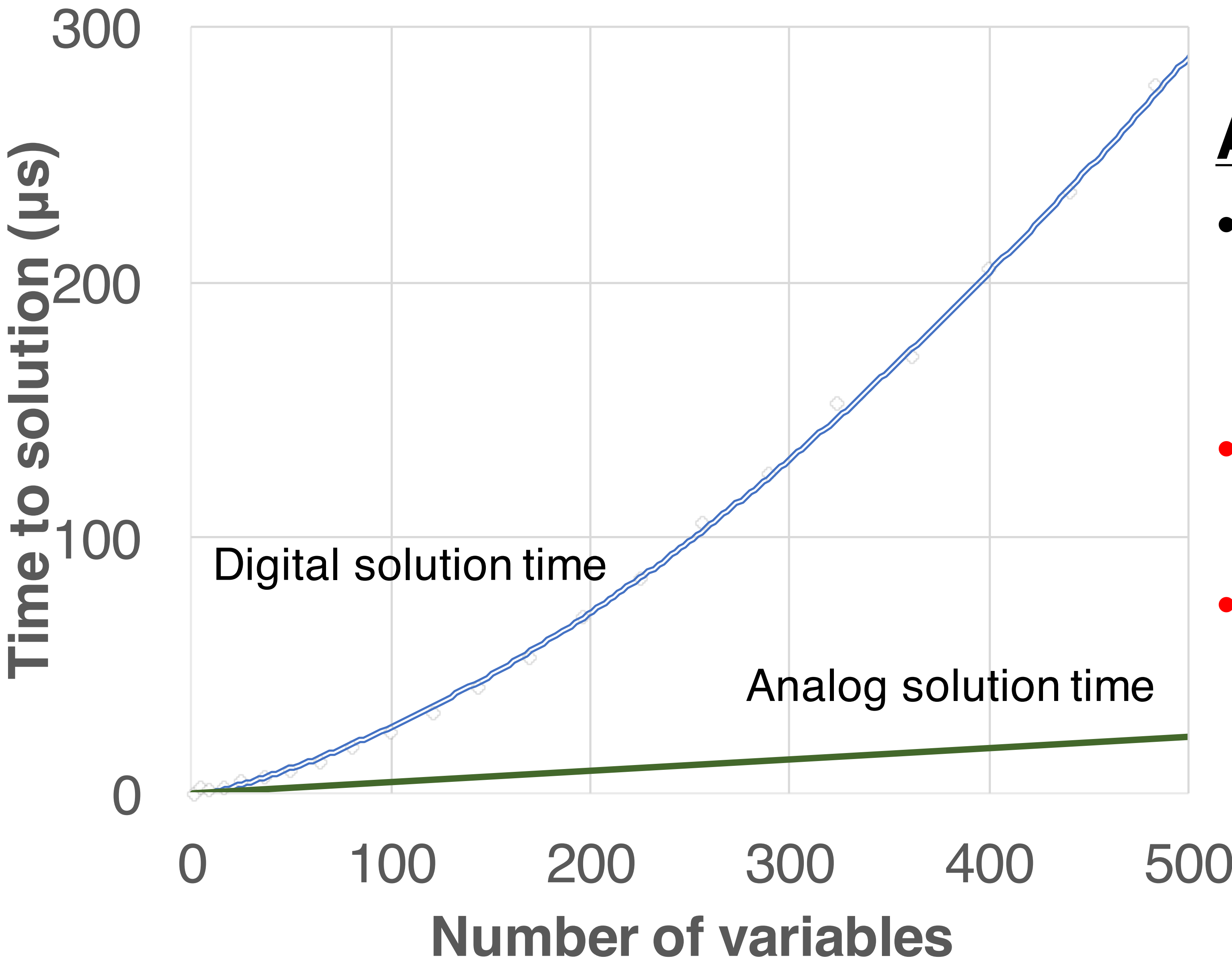
Solution time: analog HW vs. digital SW



Analog:

- **Explicit dataflow architecture**

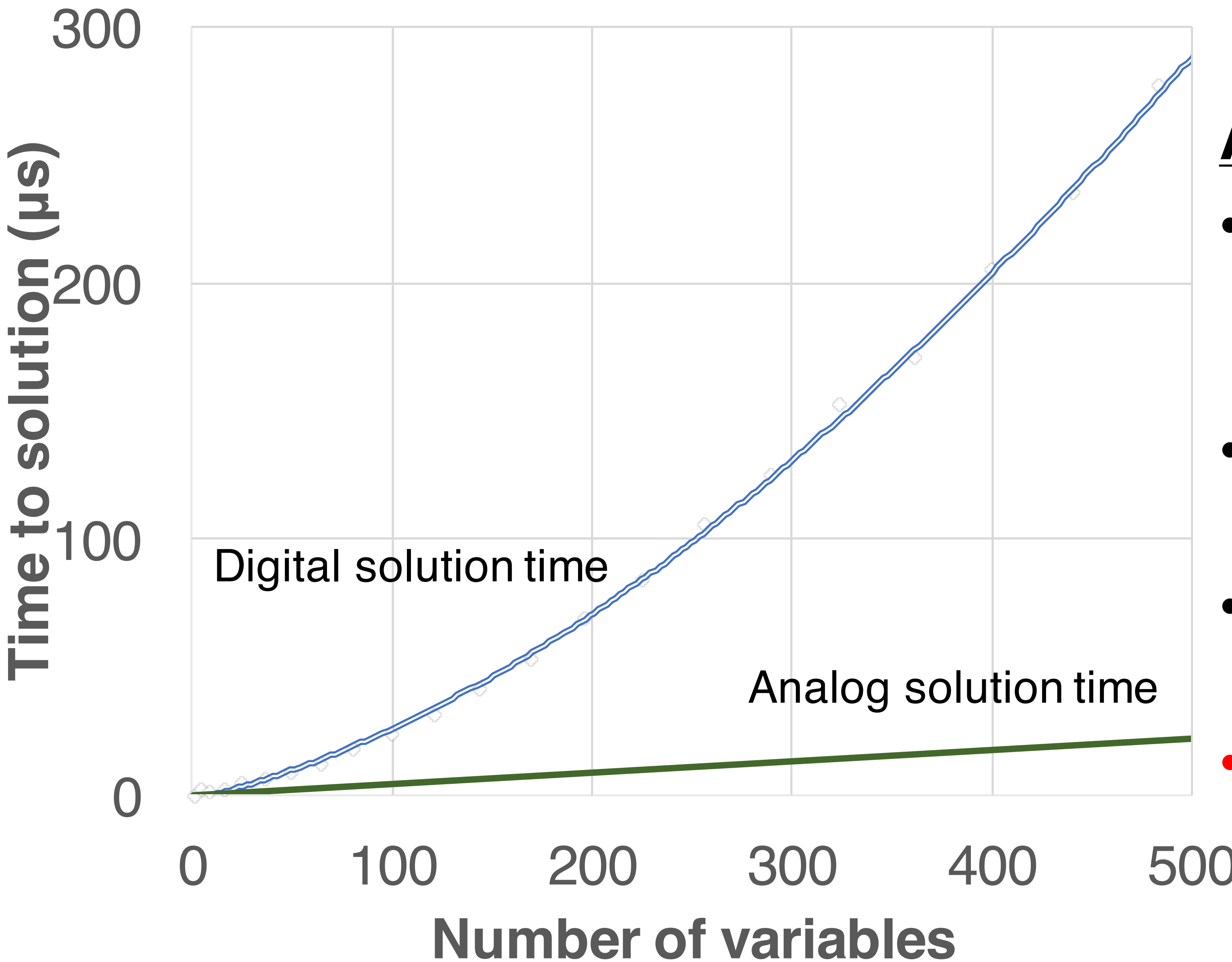
Solution time: analog HW vs. digital SW



Analog:

- **Explicit dataflow architecture**
- **Continuous-time speed**
- **Analog efficiency**

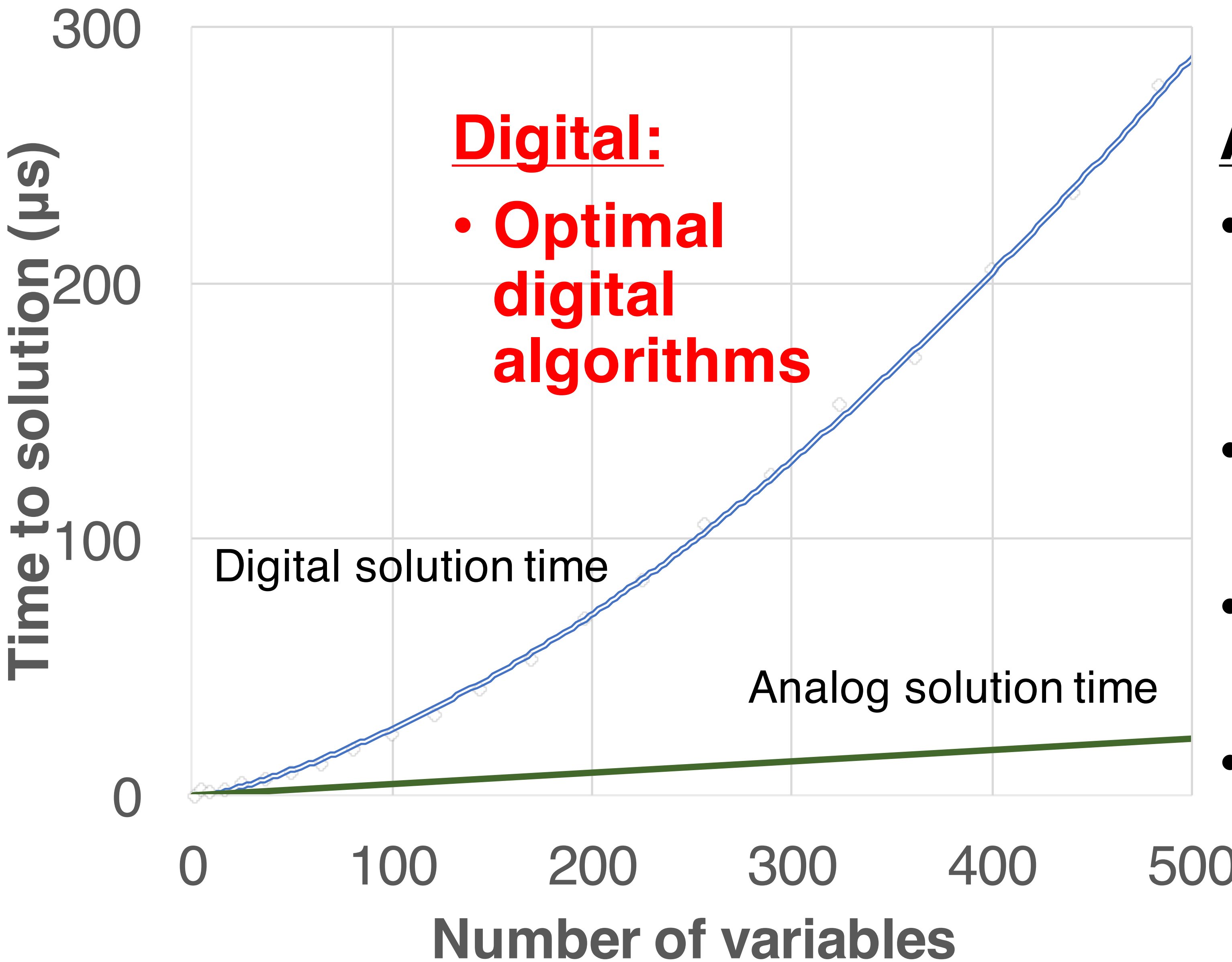
Solution time: analog HW vs. digital SW



Analog:

- **Explicit dataflow architecture**
- **Continuous-time speed**
- **Analog efficiency**
- **High analog area cost**

Solution time: analog HW vs. digital SW



Digital:

- **Optimal digital algorithms**

Analog:

- **Explicit dataflow architecture**
- **Continuous-time speed**
- **Analog efficiency**
- **High analog area cost**

An Analog Accelerator for Linear Algebra

Continuous-time + analog offers alternative abstractions to digital

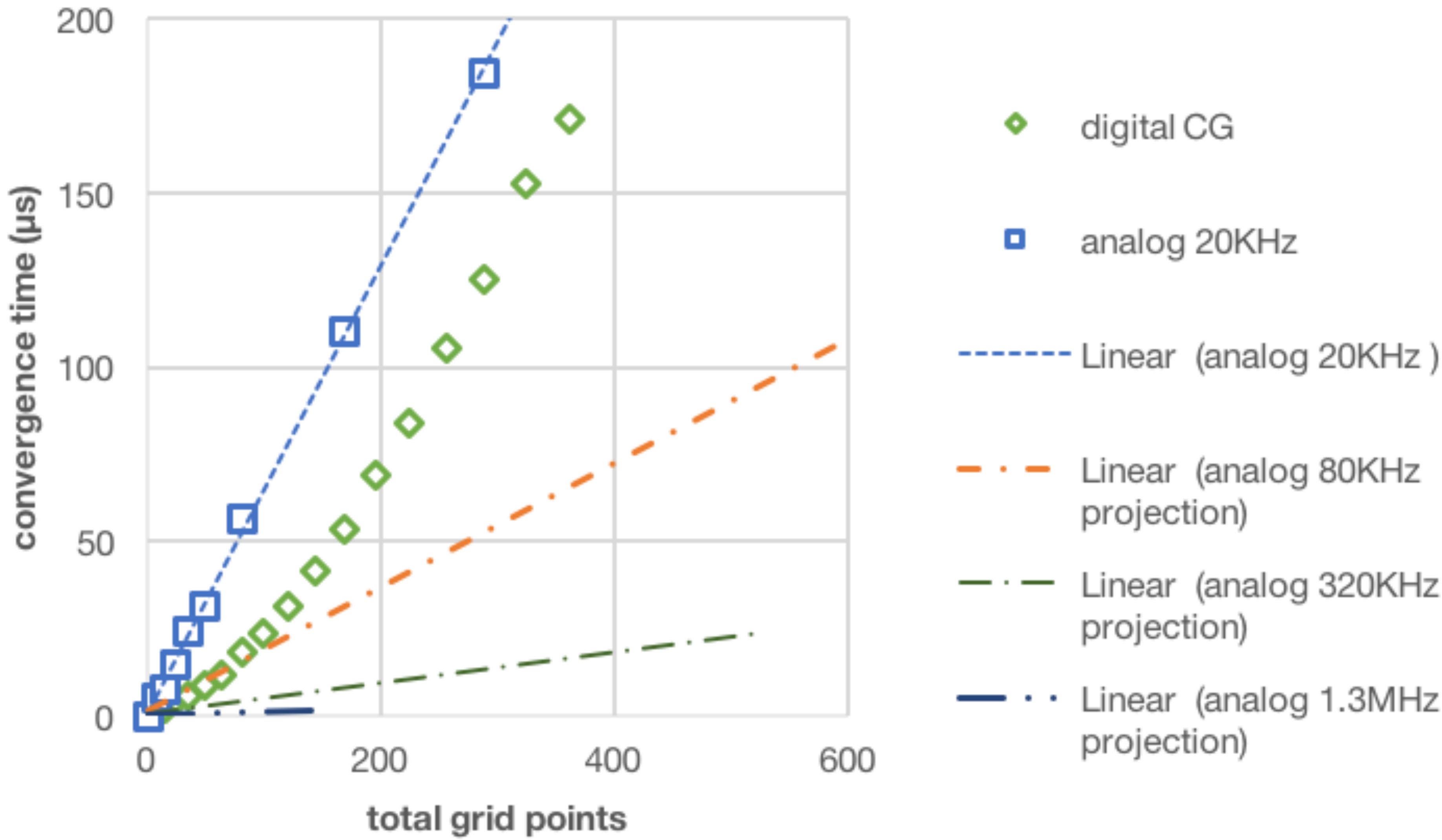
Tackled analog drawbacks: generality, accuracy, precision, scalability

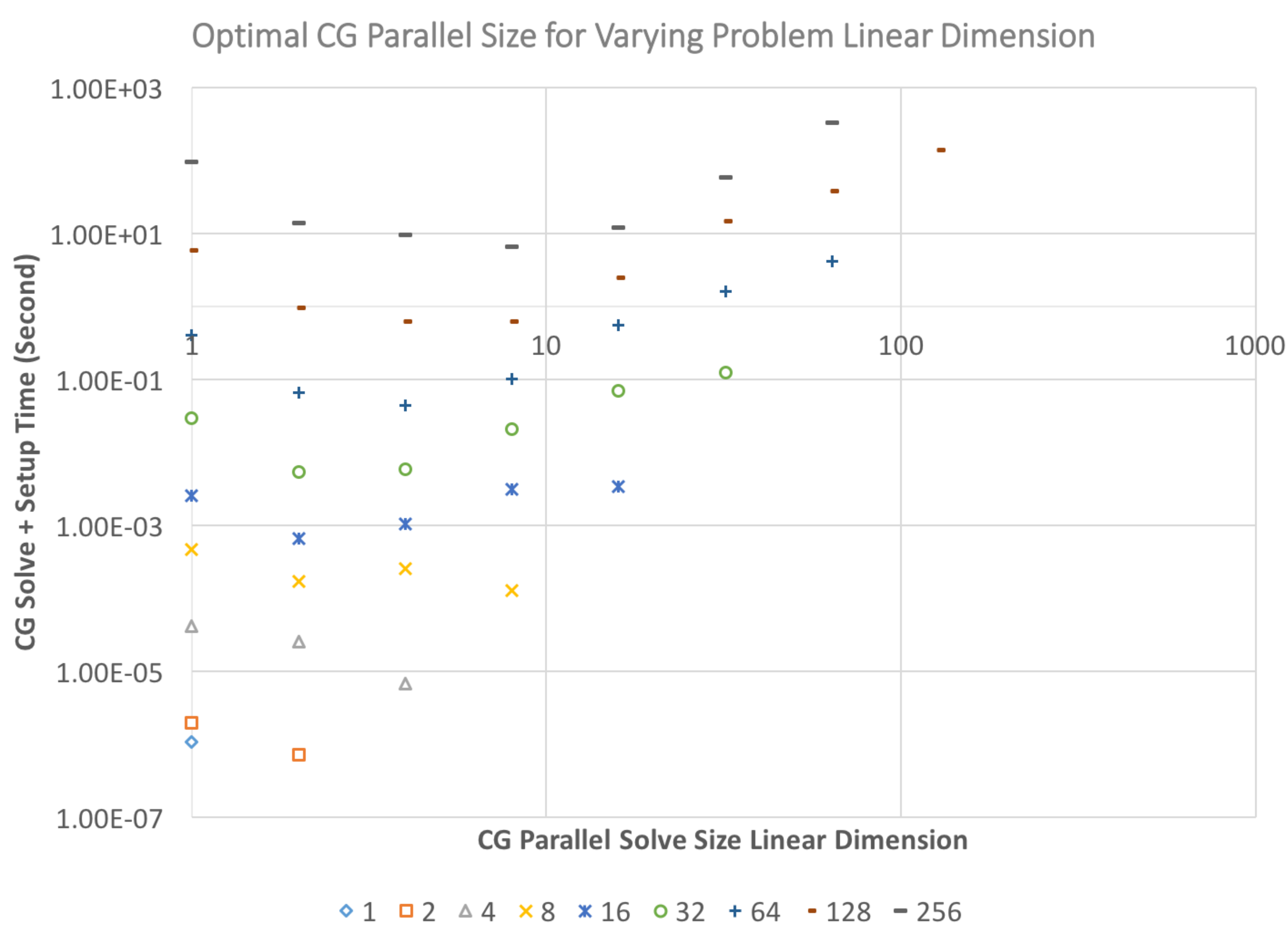
Analog prototype: ISA & hardware; speed, area, energy analysis

Should we use analog to accelerate linear algebra?

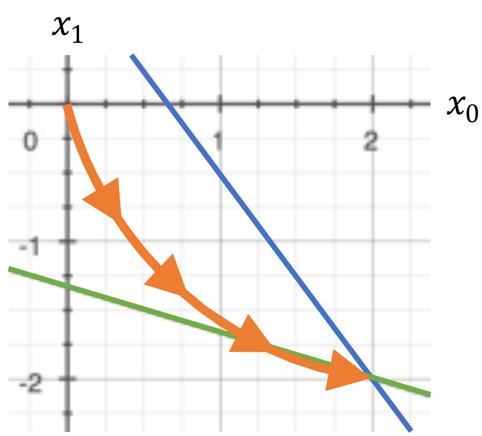
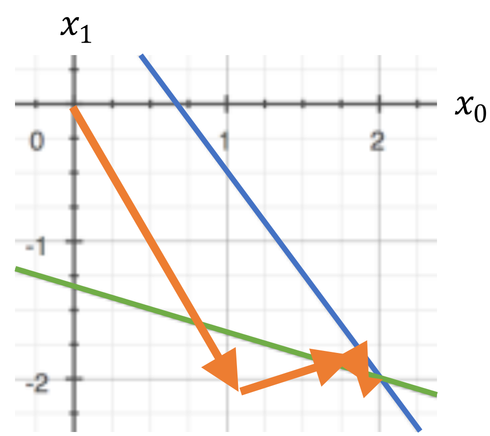
Some gains, but digital algorithms are optimized!

Analog promises greater advantage in other problems: nonlinear?


convergence time (µs)
200
150
100
50
0
0
200
400
600
total grid points
digital CG
analog 20KHz
Linear (analog 20KHz)
Linear (analog 80KHz projection)
Linear (analog 320KHz projection)
Linear (analog 1.3MHz projection)



Optimal CG Parallel Size for Varying Problem Linear Dimension
1.00E+03
1.00E+01
1.00E-01
1.00E-03
1.00E-05
1.00E-07
1
10
100
1000
CG Solve + Setup Time (Second)
CG Parallel Solve Size Linear Dimension
1
2
4
8
16
32
64
128
256

Analog	Digital
	
Continuous time Ordinary differential equation Advantage: fast Advantage: low power b/c no clock	**Discrete time** Recurrence relation **Advantage: allows complex algorithms** Advantage: allows time multiplexing
Continuous value Current & voltage Advantage: fast and efficient operations Advantage: one wire carries real number	**Discrete value** Integers & floating point **Advantage: high dynamic range** Advantage: high signal-to-noise ratio