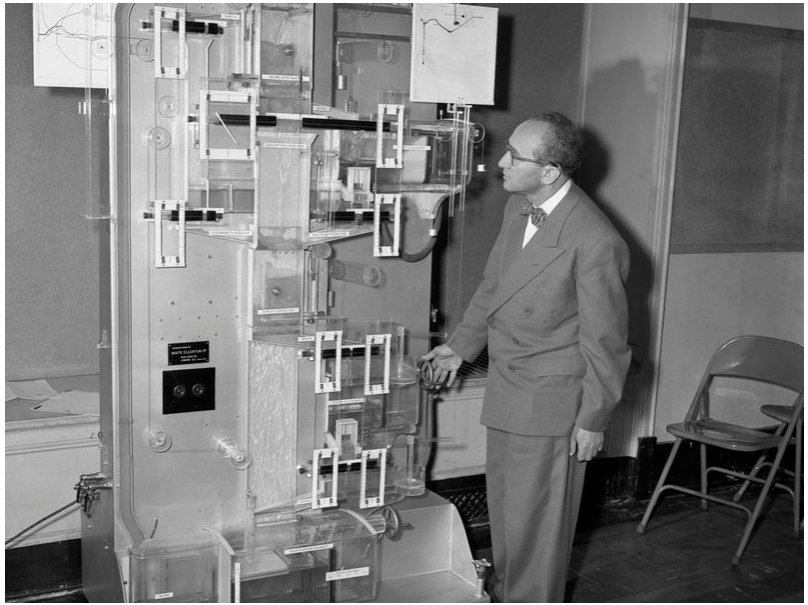
New Compilation Techniques for Reconfigurable Analog Devices

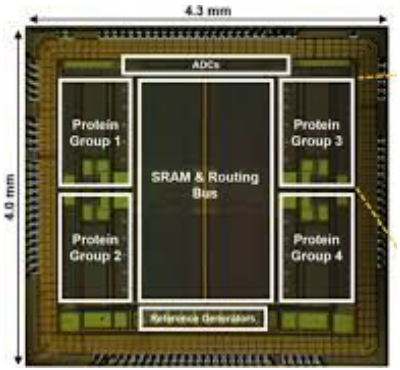
Sara Achour
Stanford CS & Stanford EE

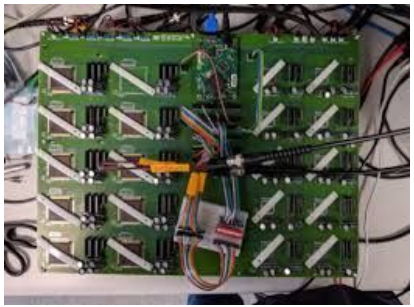
Automatically program analog computers to perform computation

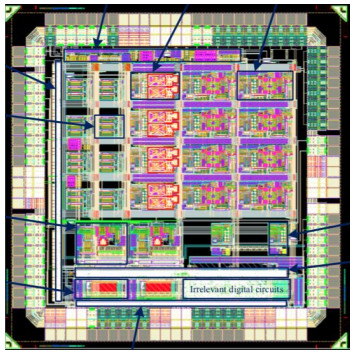


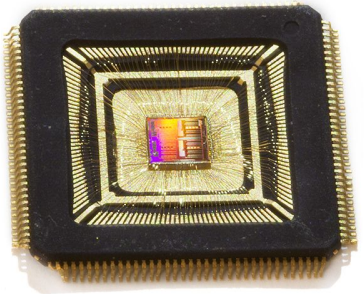
Ultra-Low Power Reconfigurable Analog Devices

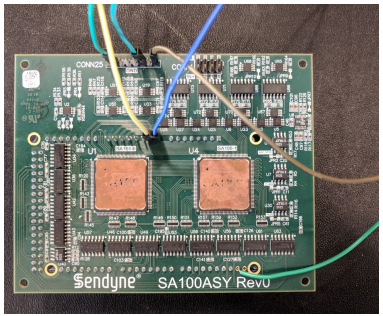
- CMOS Technology
- Digitally reconfigurable



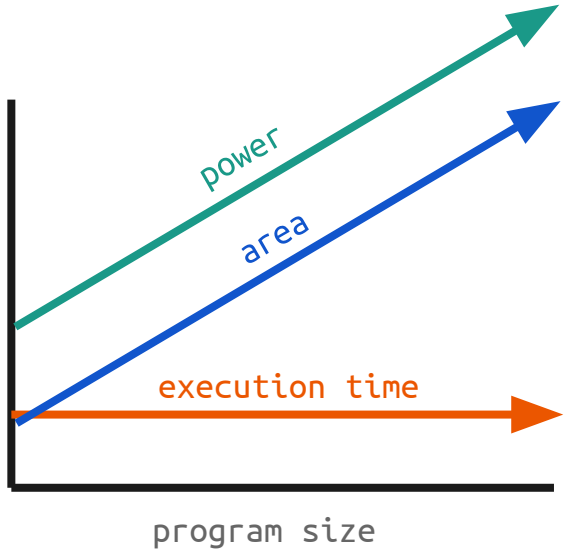


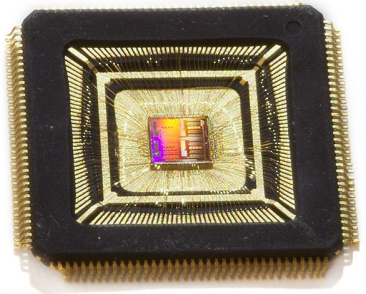






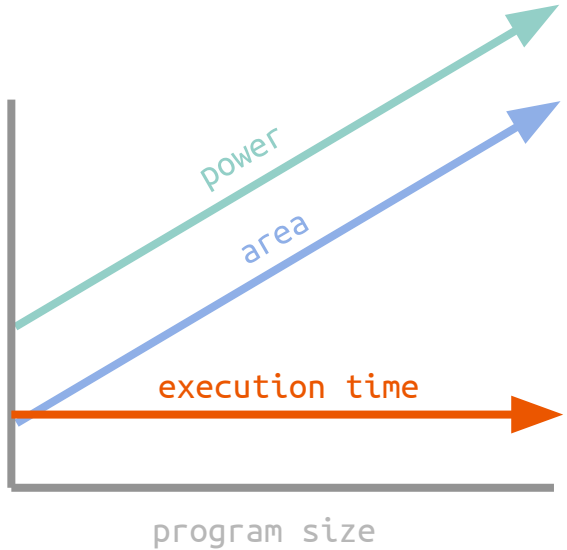
Ultra-Low Power Reconfigurable Analog Devices

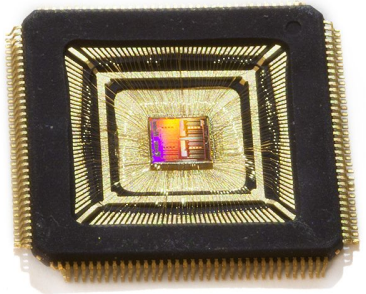




Perform computation using microjoules of energy!

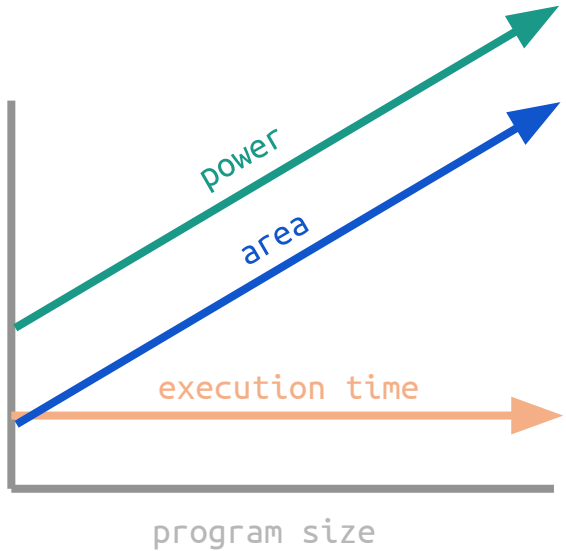
Ultra-Low Power Reconfigurable Analog Devices

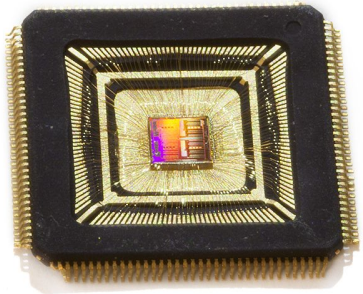




Execution time is constant w.r.t. complexity of computation

Ultra-Low Power Reconfigurable Analog Devices

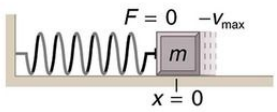


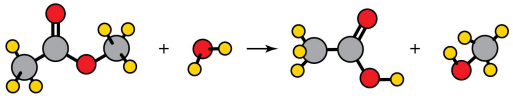


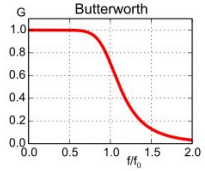
This target class of analog devices executes dynamical systems

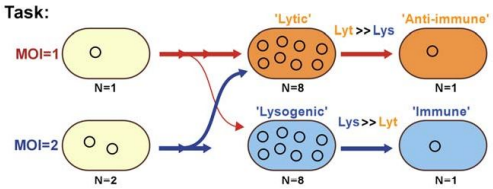
Differential-Equation Solving
Analog Devices

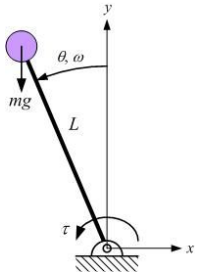
What are Dynamical Systems?









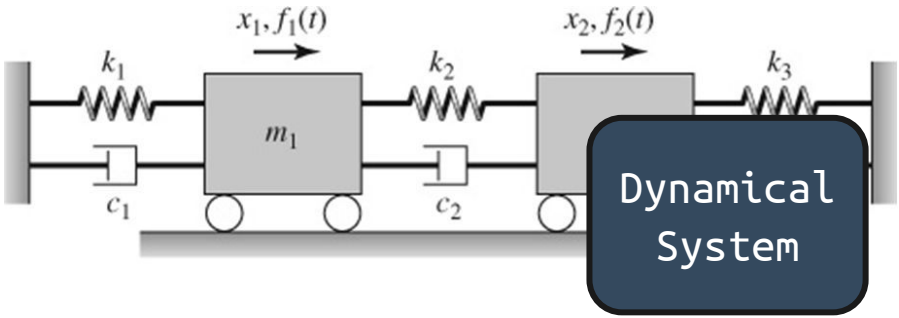


Differential-Equation Solving
Analog Devices

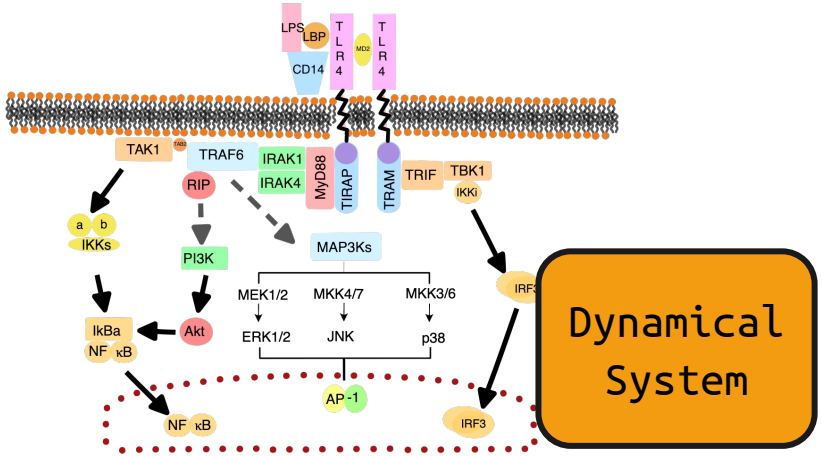
Dynamical Systems can be used to

Study physical phenomena

Two-Spring System



Biological Pathway

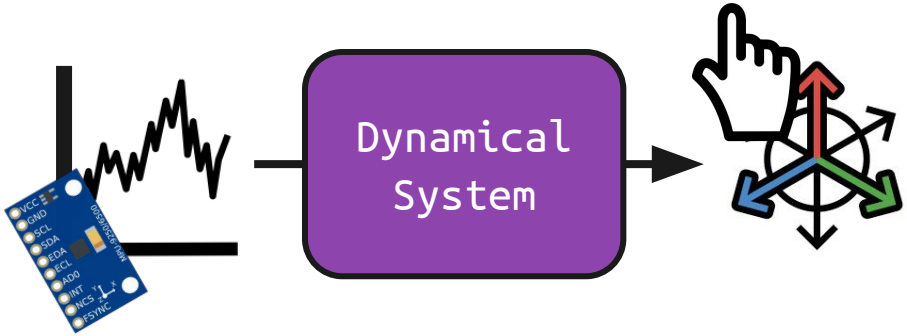


Differential-Equation Solving
Analog Devices

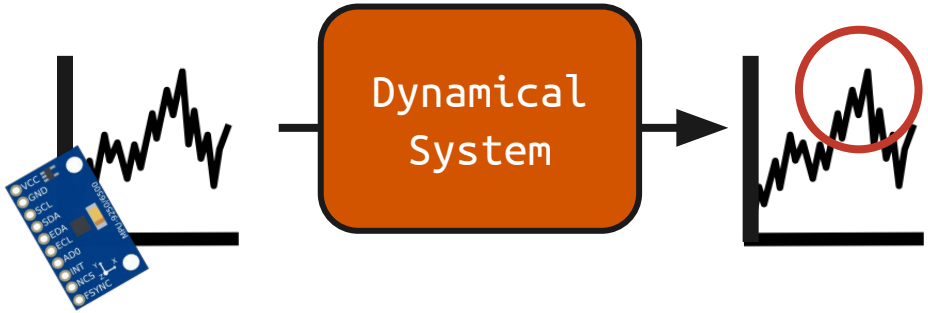
Dynamical Systems can be used to

Analyze surroundings

Position Tracking



Anomaly Detection

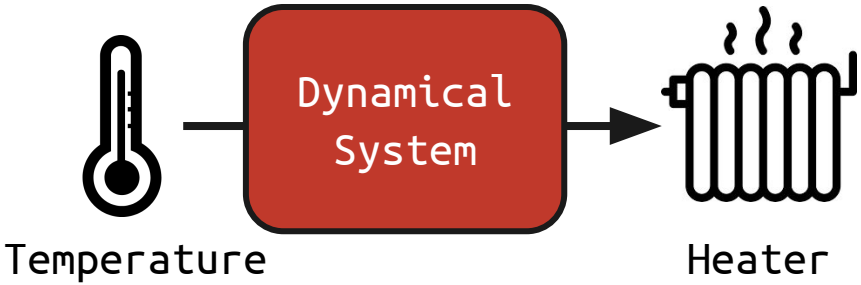


Differential-Equation Solving
Analog Devices

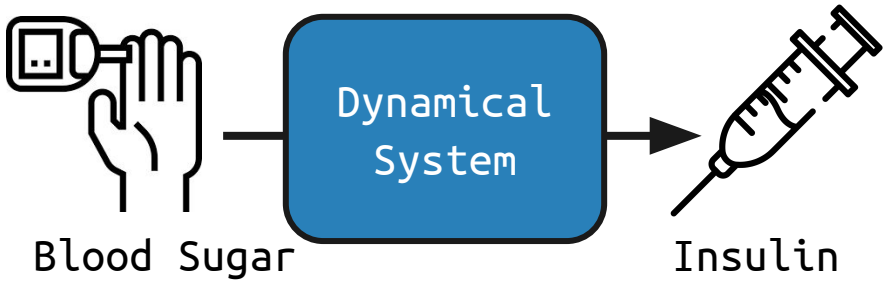
Dynamical Systems can be used to

React to the Environment

Thermostat



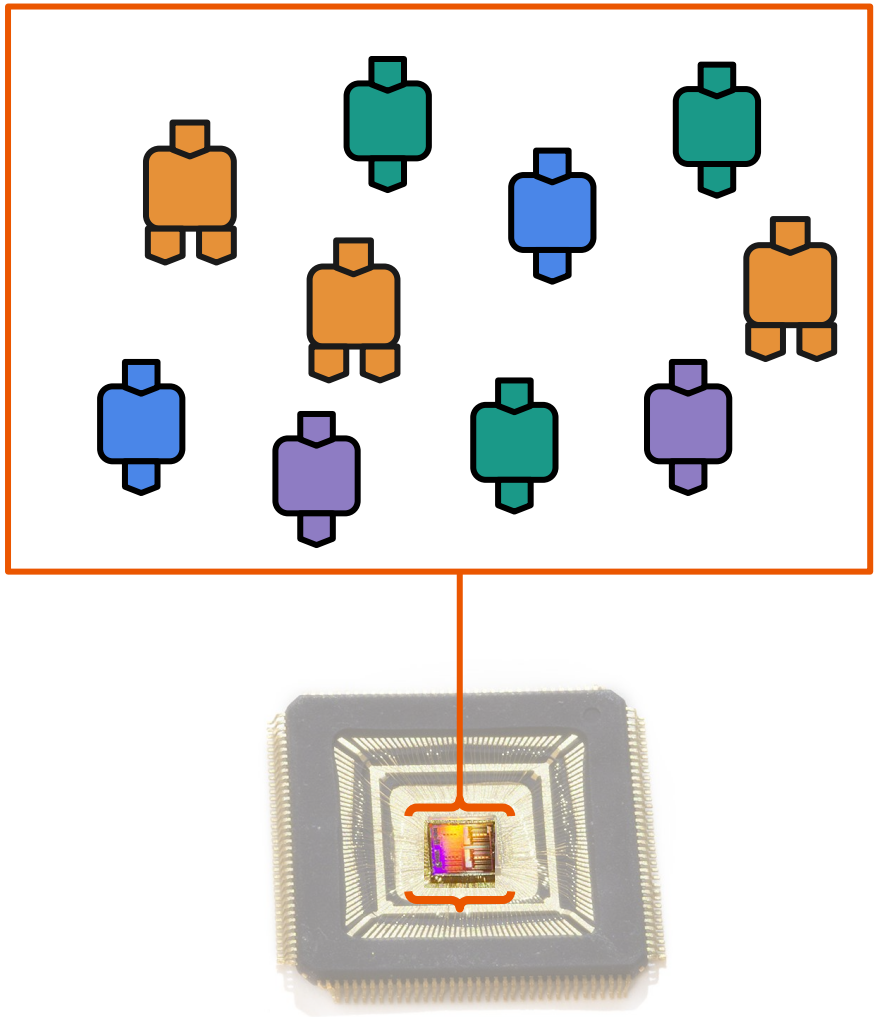
Insulin Pump



How do these analog devices work?

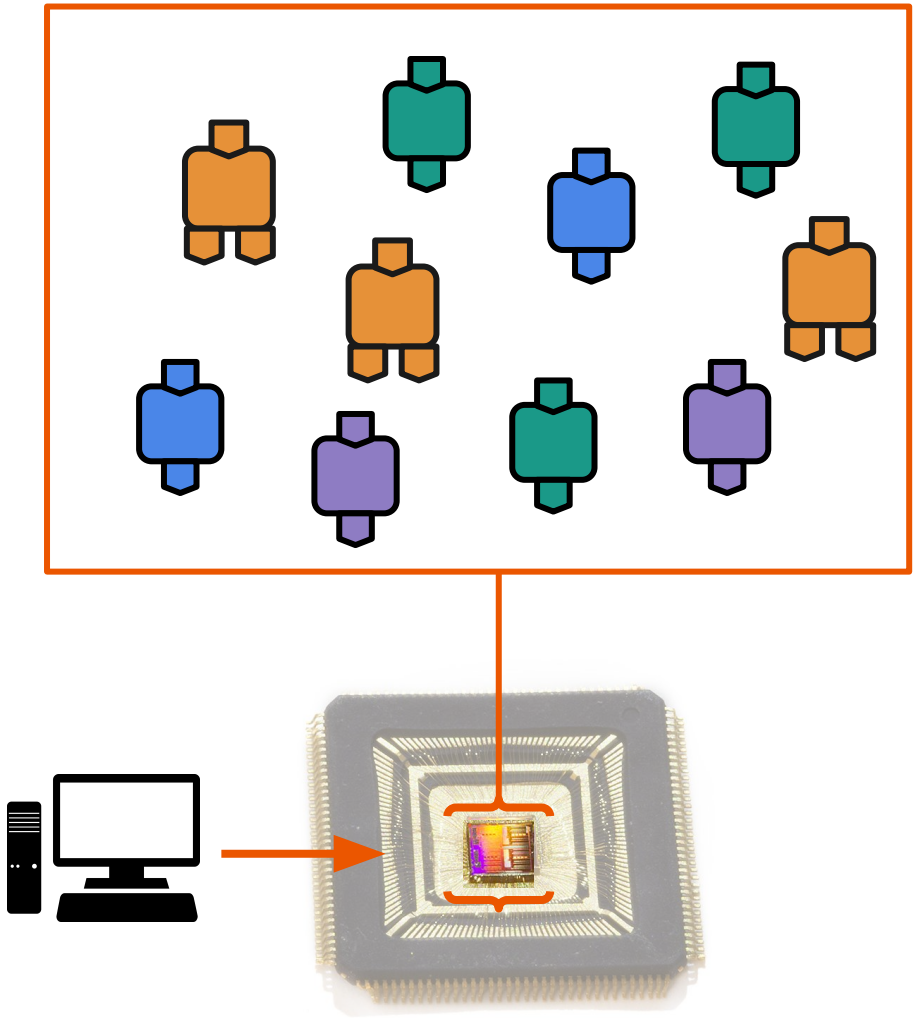
Analog Devices

Contain collections of computational blocks



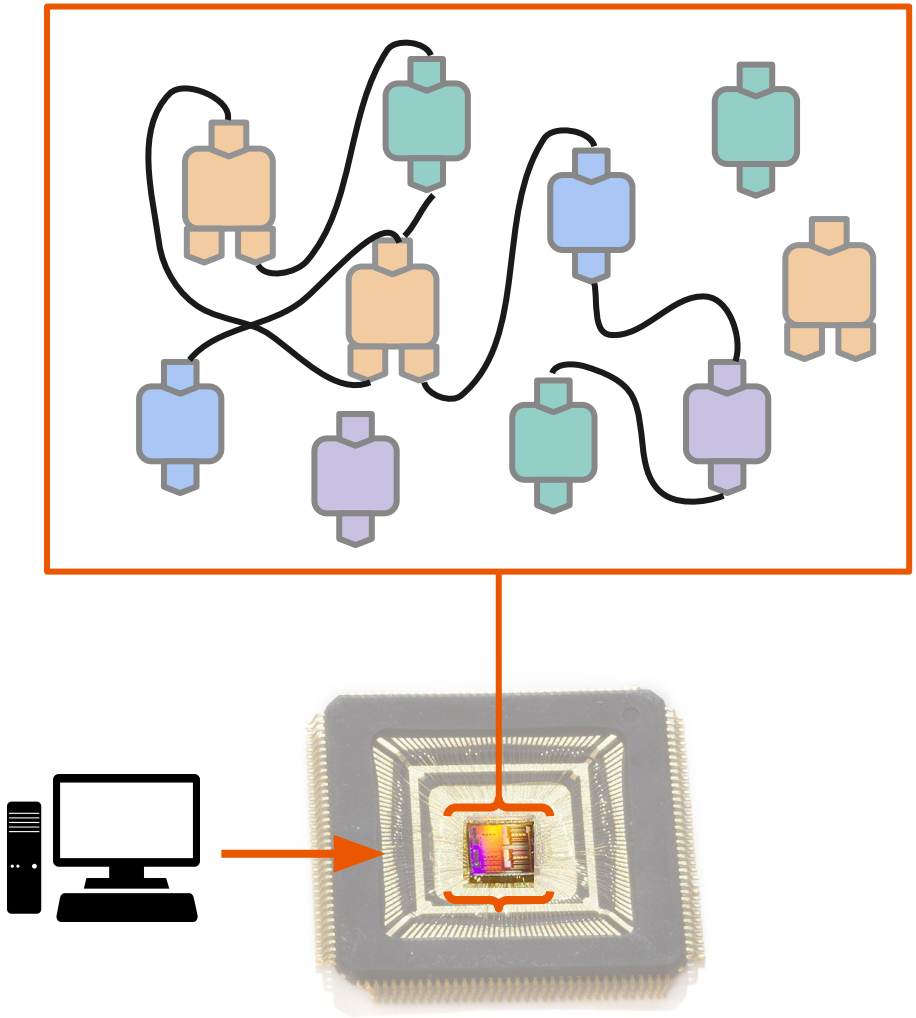
Analog Devices

Are programmable



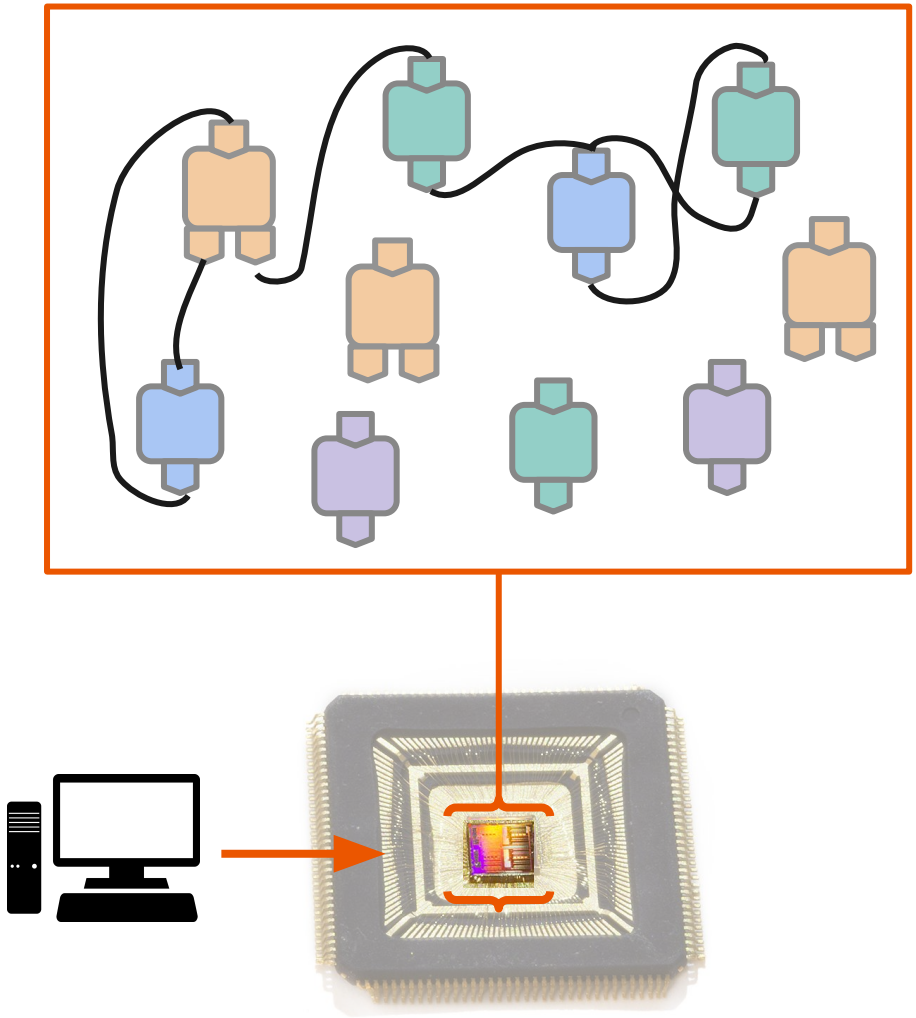
Analog Devices

Are programmable



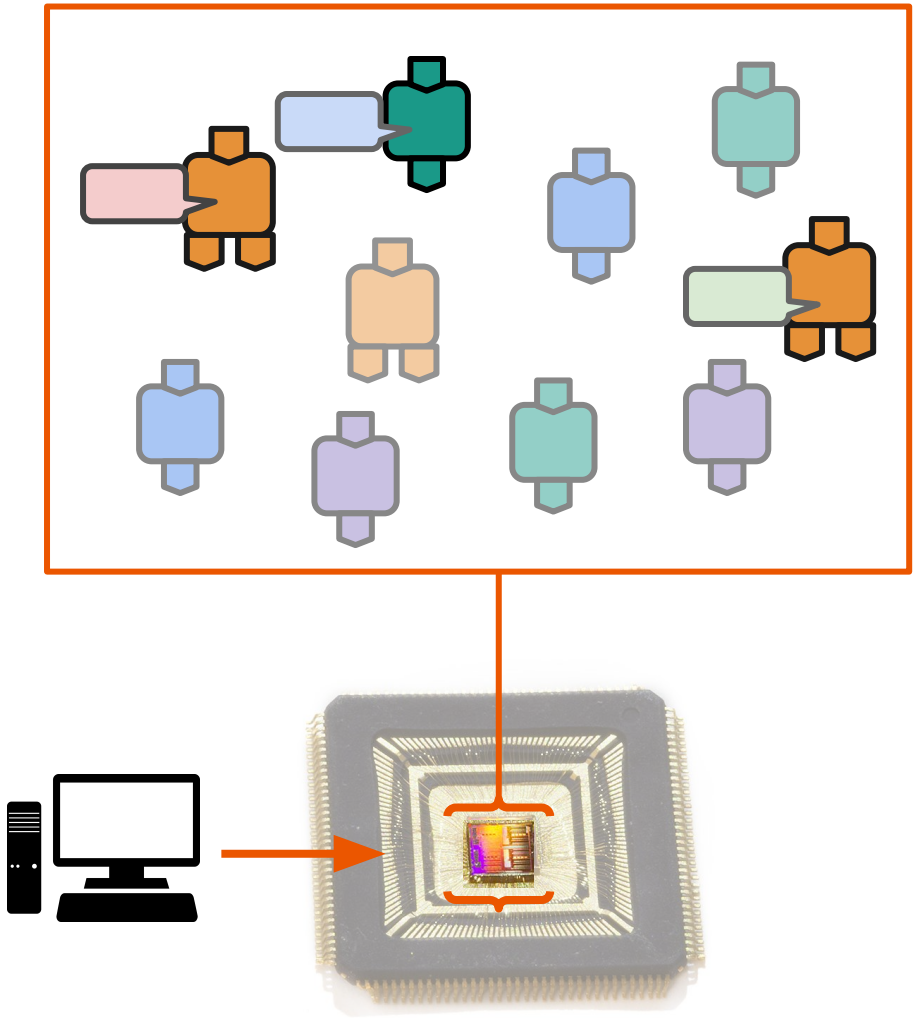
Analog Devices

Are programmable



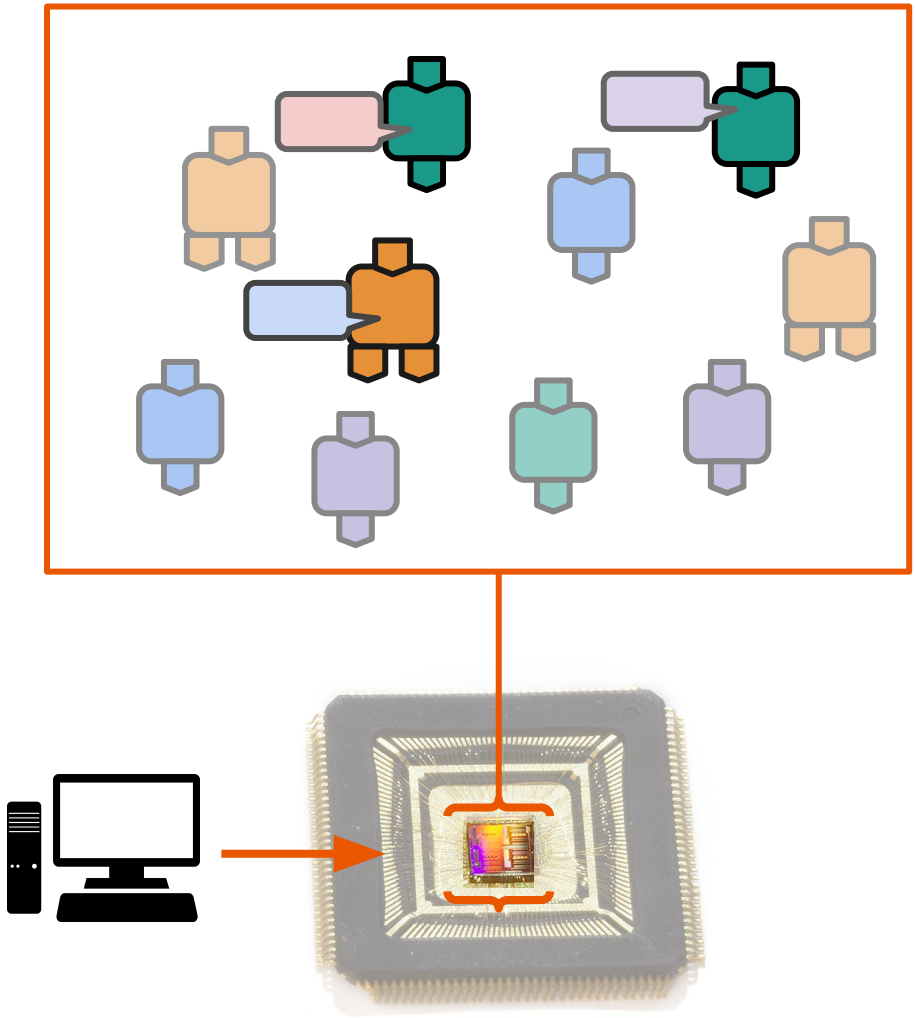
Analog Devices

Are programmable



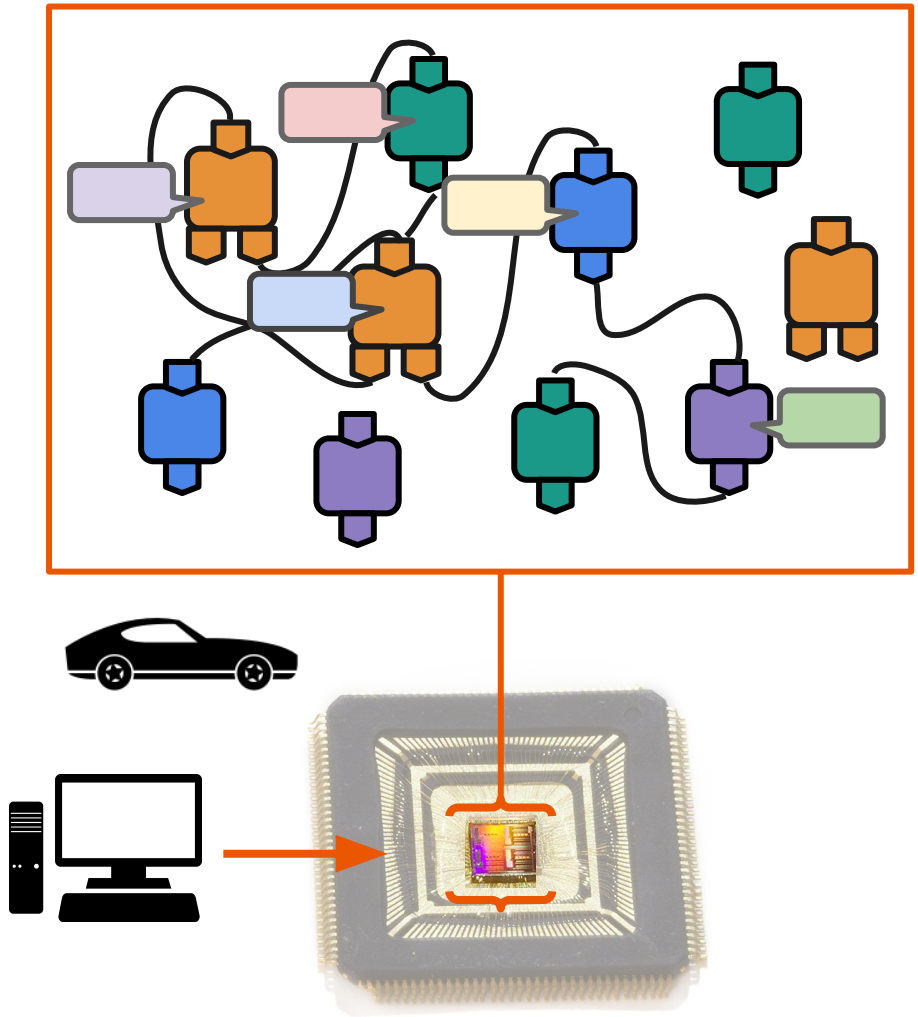
Analog Devices

Are programmable



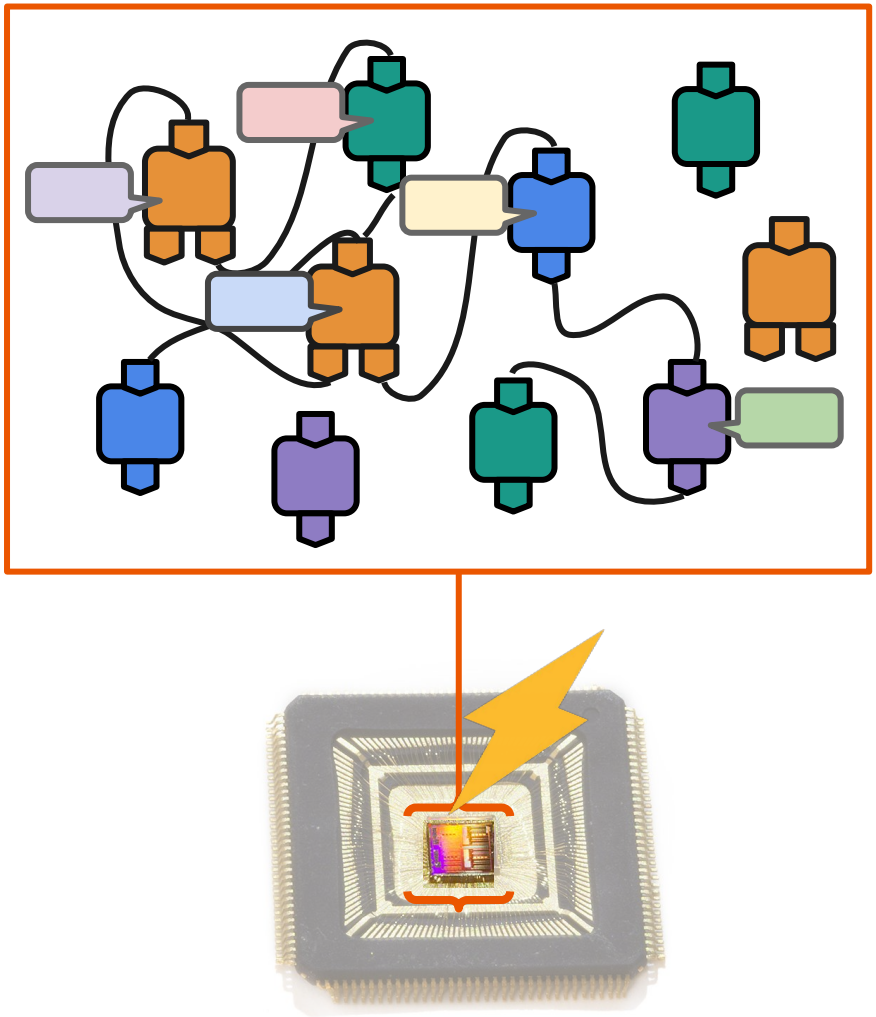
Analog Devices

Perform computation using device physics



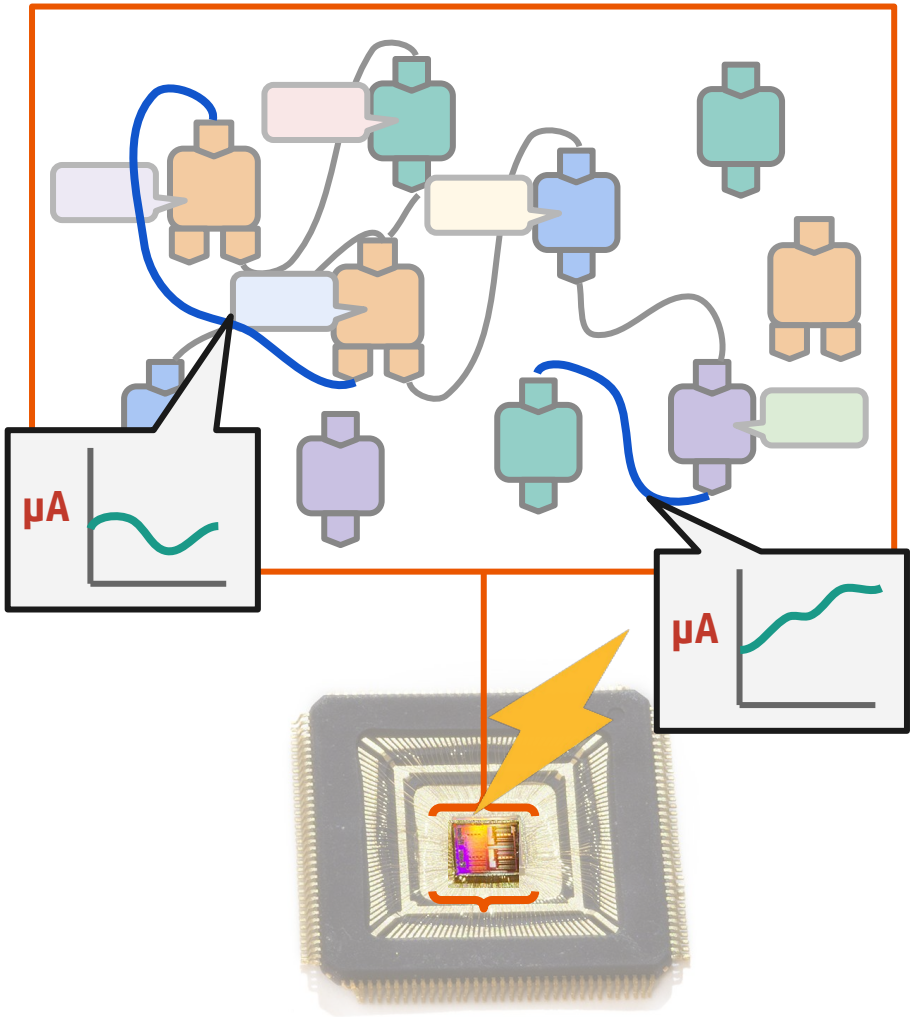
Analog Devices

Perform computation using device physics



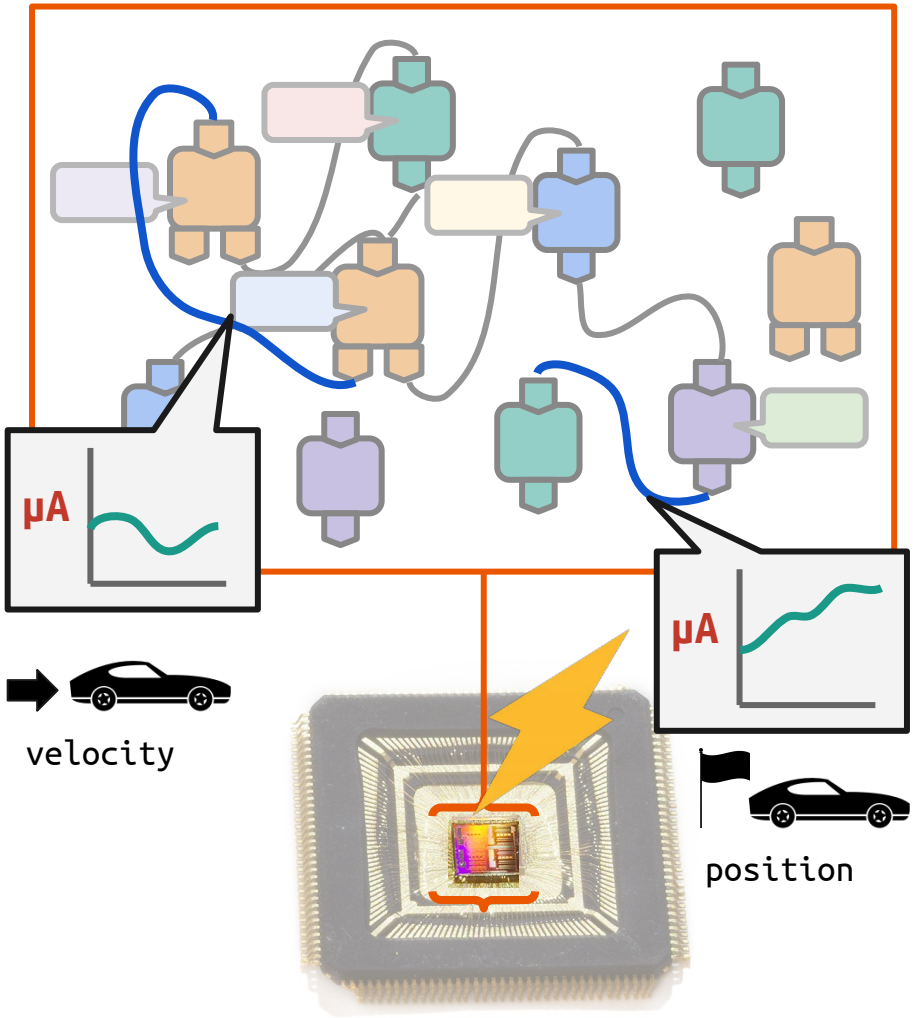
Analog Devices

Perform computation using device physics



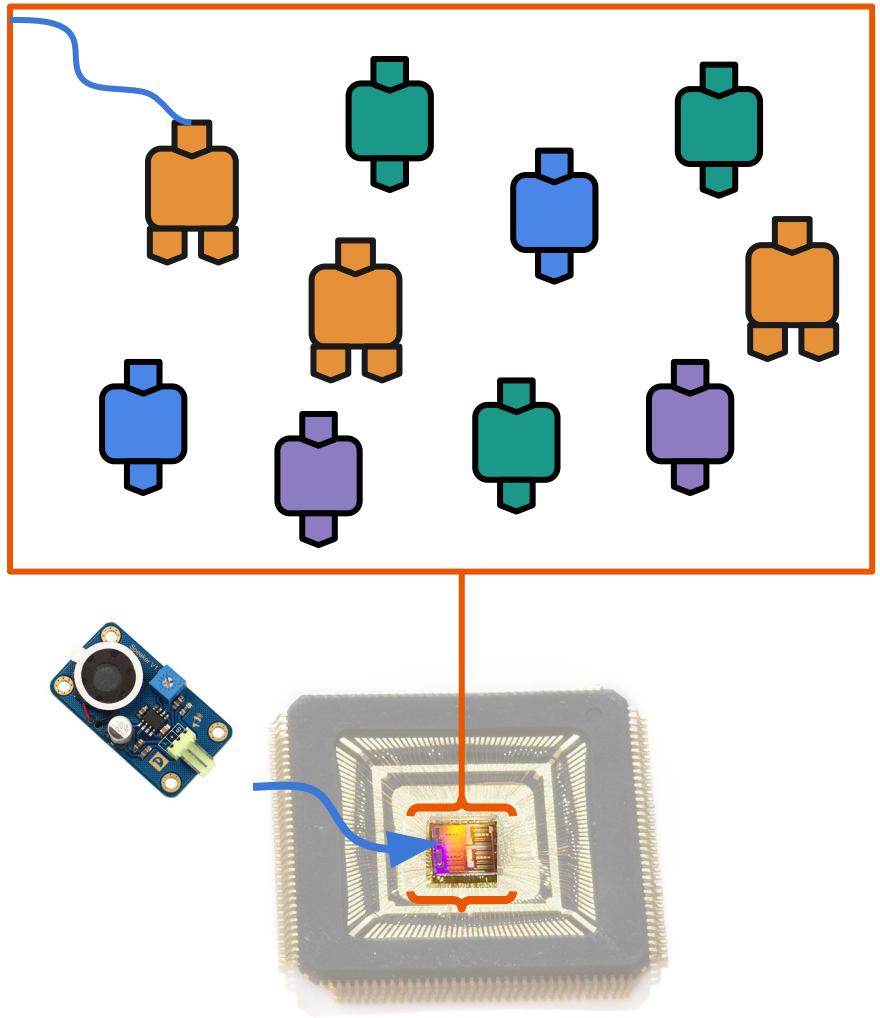
Analog Devices

Perform computation using device physics



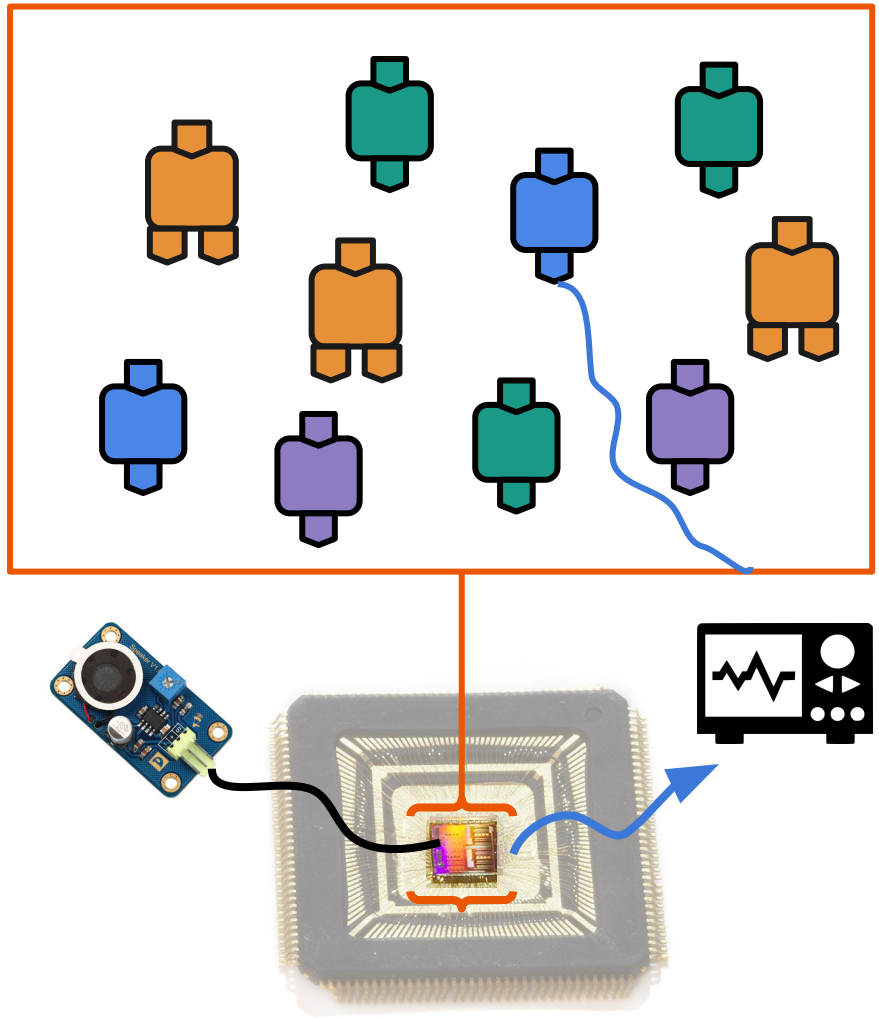
Analog Devices

Work with external signals



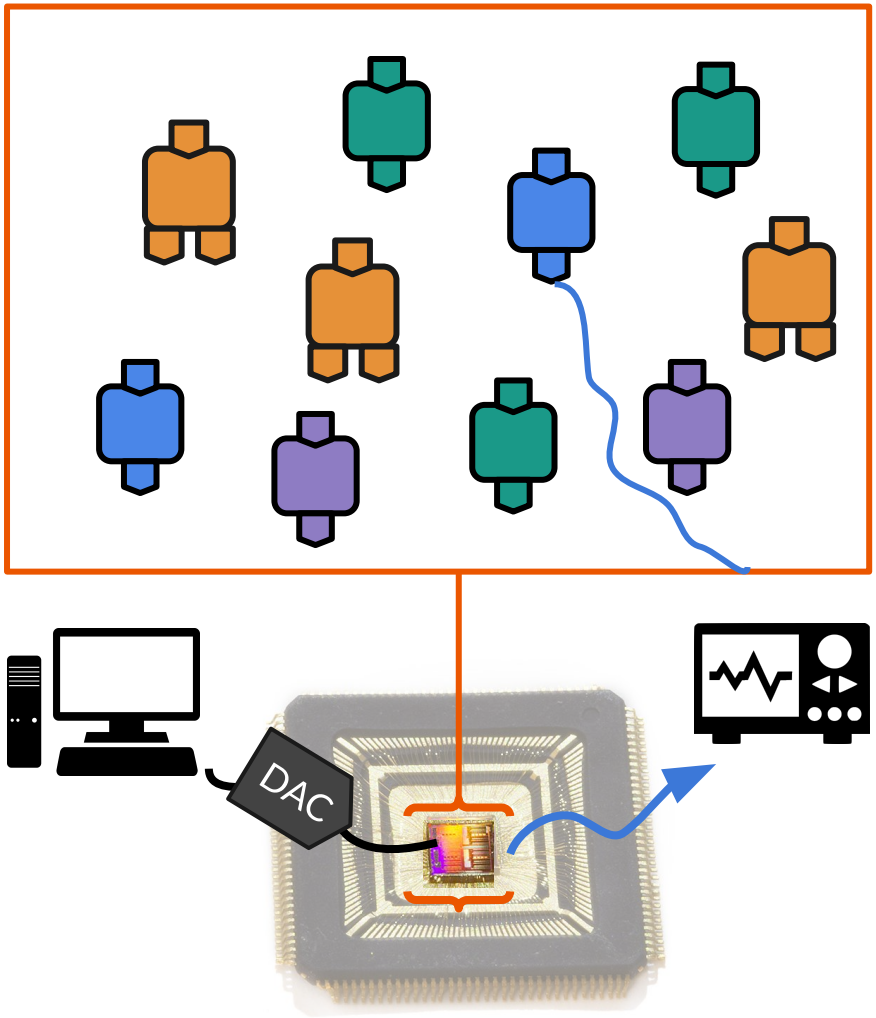
Analog Devices

Work with external signals



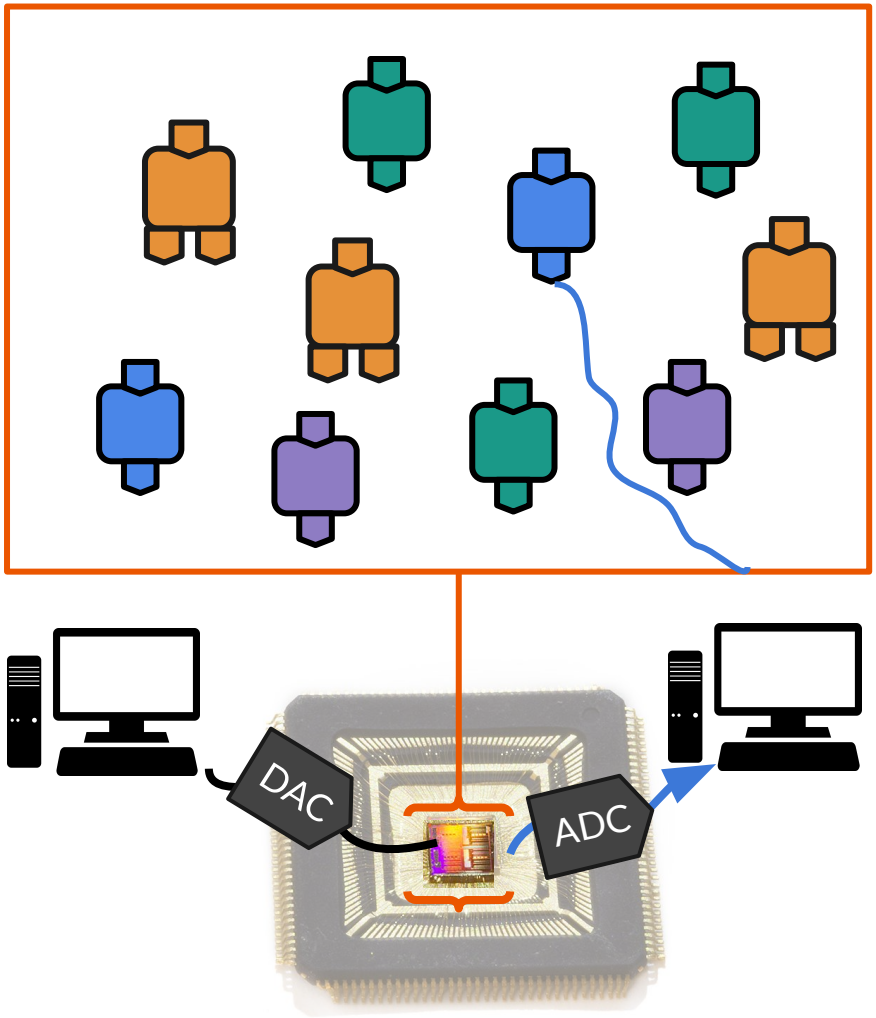
Analog Devices

Work with external signals



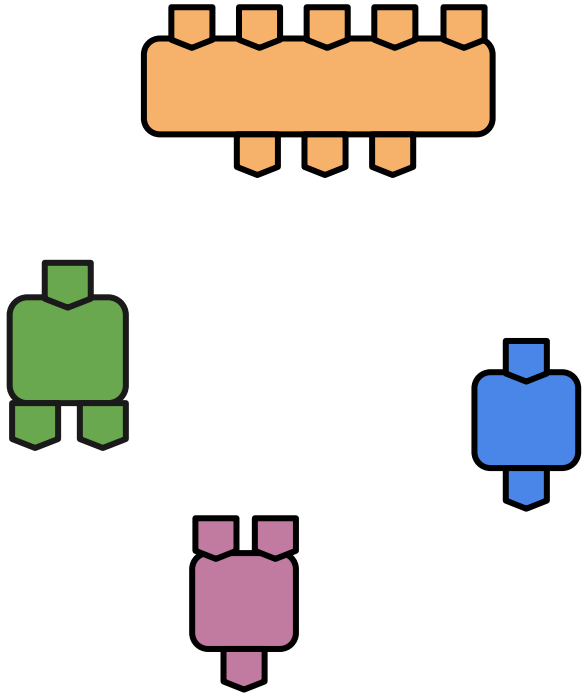
Analog Devices

Work with external signals

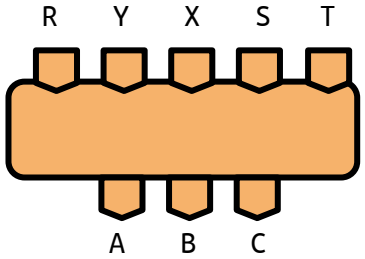


Characteristics of Analog Devices

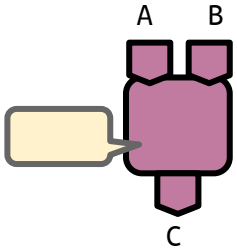
Wide variety of programmable analog blocks



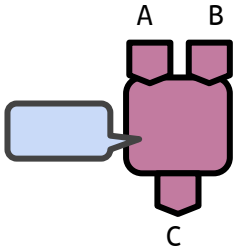
Analog blocks designed for efficiency and for usability



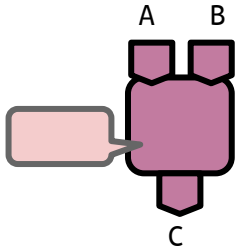
A=X-C
B=Y-C
dC/dt=∫ R*A*B-S*C
C(0)=T



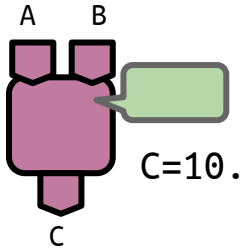
C=0.5*A*B



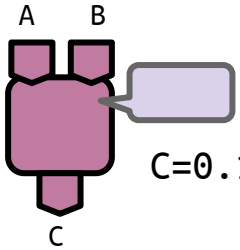
C=0.05*A*B



C=5*A*B

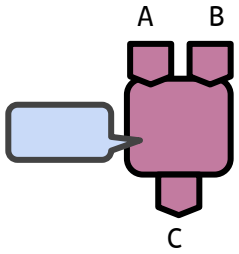


C=10.0*A



C=0.1*A

Analog blocks have analog behavior

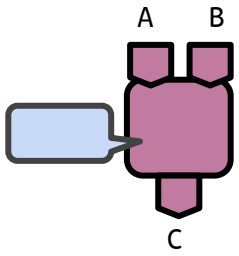


C=0.05*A*B

Analog blocks have analog behavior

Operating Range Limitations

Frequency Limitations



Quantization Error

Process Variations

C=0.05*A*B

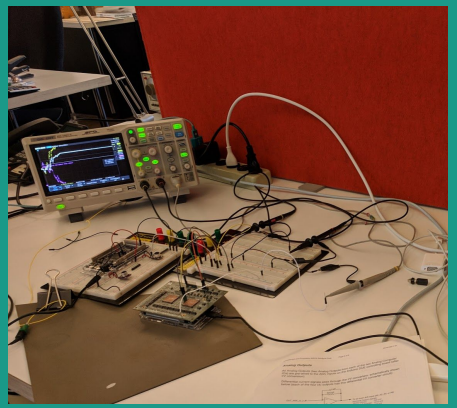
Analog Noise

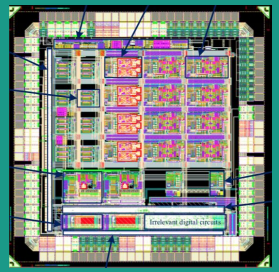
I developed compilation techniques for this class of devices

1. Time Dilation and Contraction for Programmable Analog Devices with Jaunt. ASPLOS 2018.
2. Configuration Synthesis for Programmable Analog Devices with Arco. PLDI 2016.

I built the first-ever compilation toolchain for a real-world device of this class

Noise-Aware Dynamical System Compilation for Analog Devices with Legno. ASPLOS 2020.



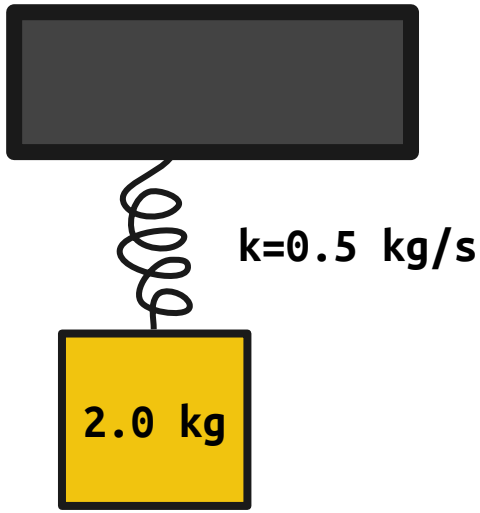


How does the compiler work?

Input

Dynamical System

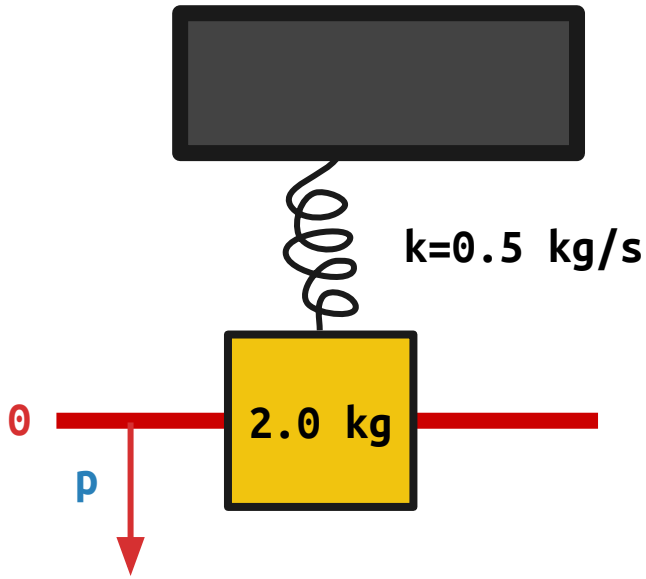
Harmonic Oscillator



Input

Dynamical System

Harmonic Oscillator

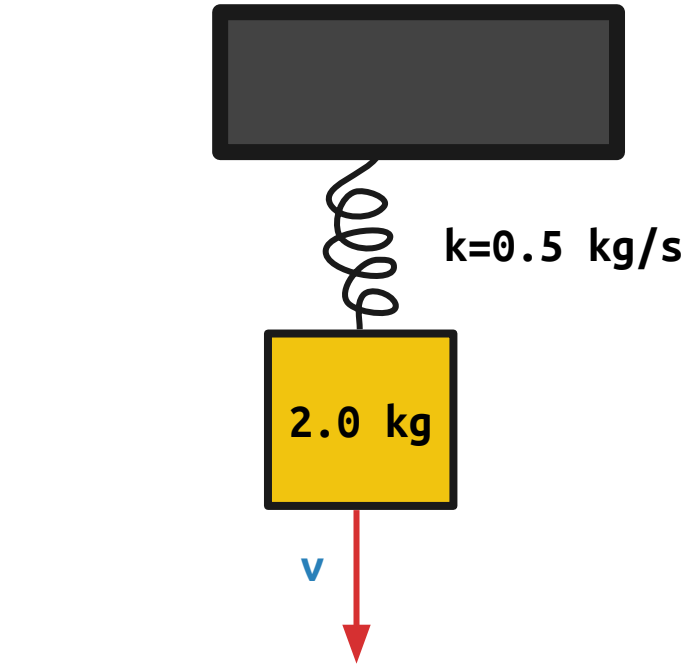


p: position in meters

Input

Dynamical System

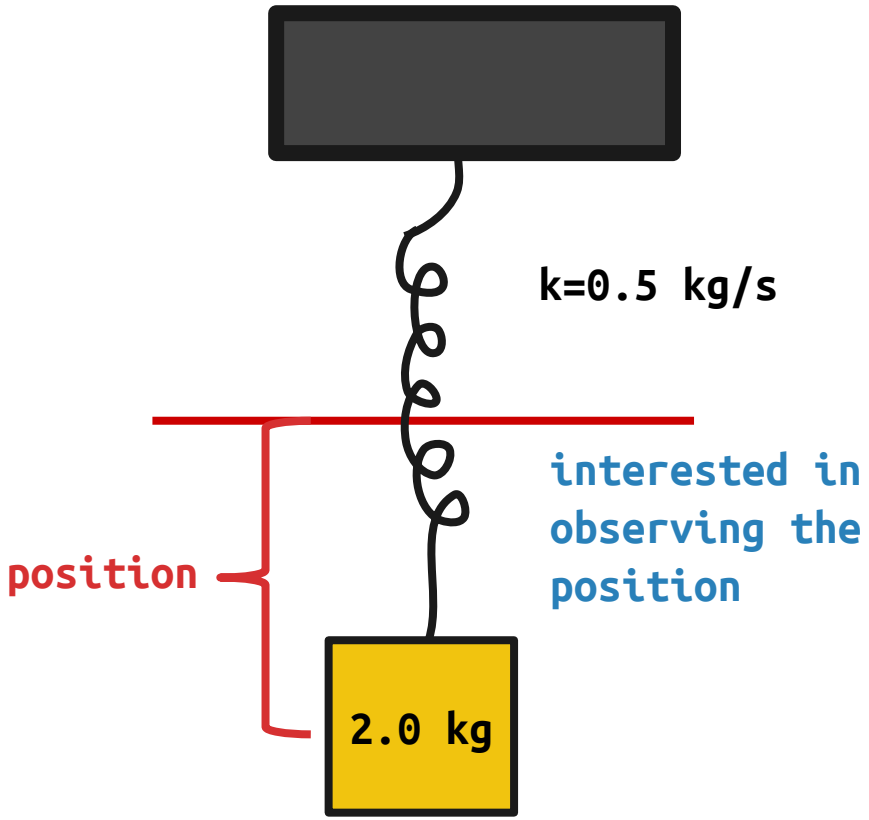
Harmonic Oscillator



Input

Dynamical System

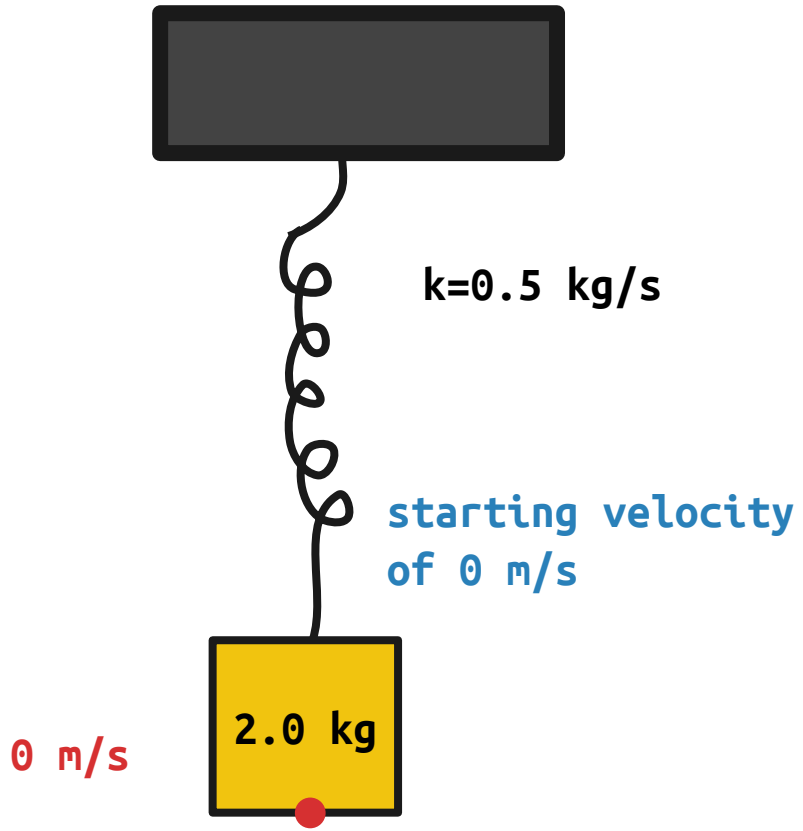
Harmonic Oscillator



Input

Dynamical System

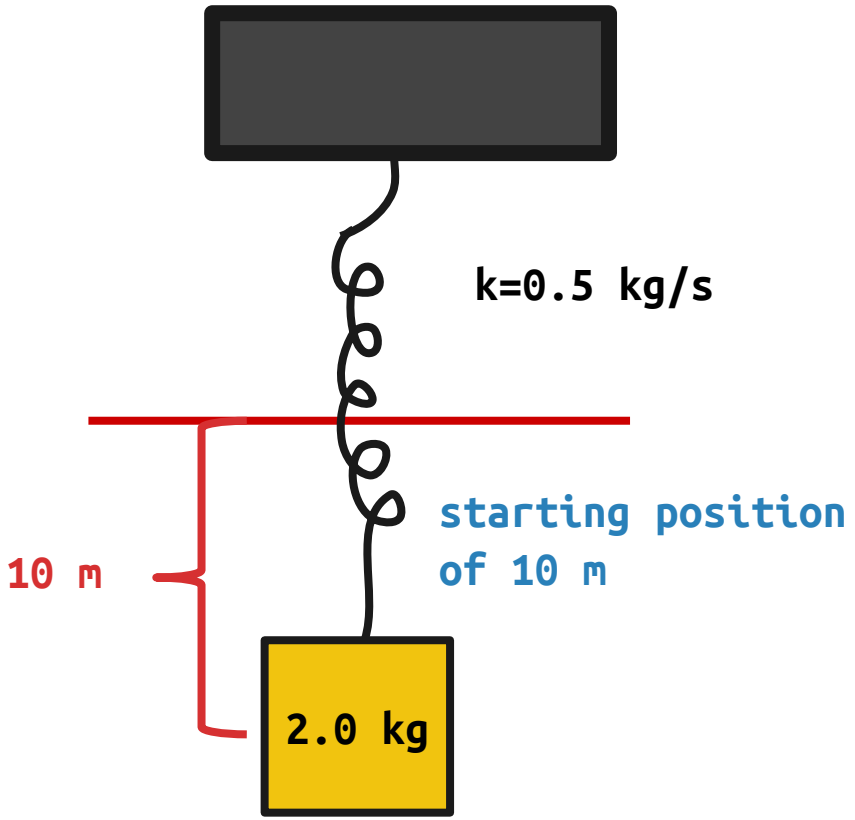
Harmonic Oscillator



Input

Dynamical System

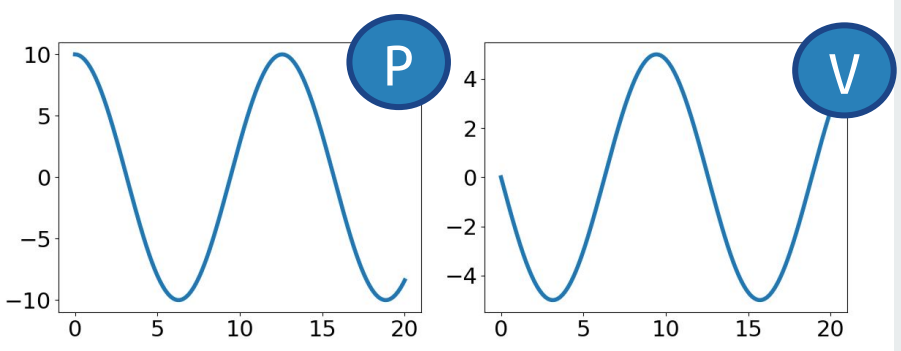
Harmonic Oscillator

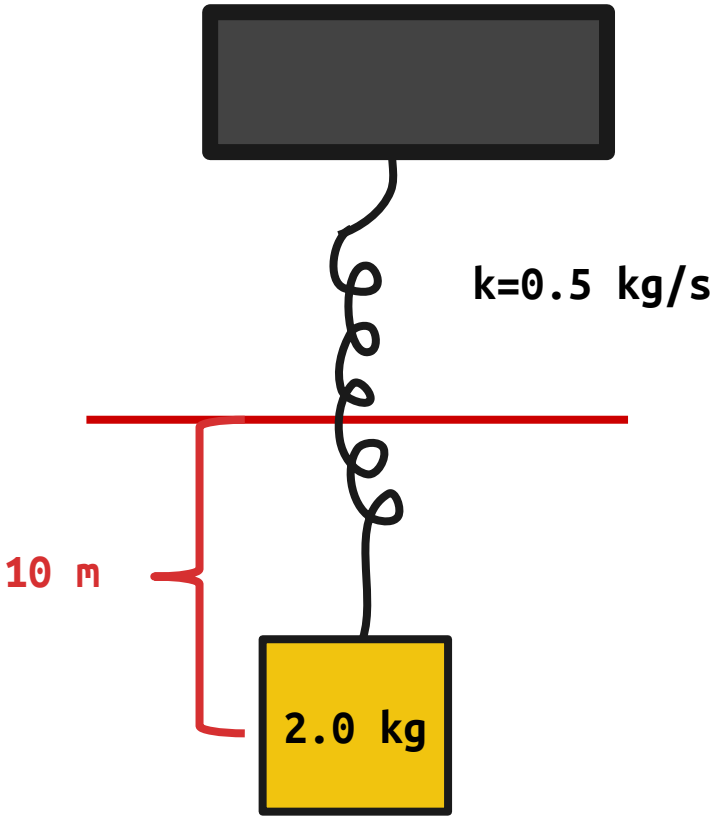


Input

Dynamical System

Harmonic Oscillator

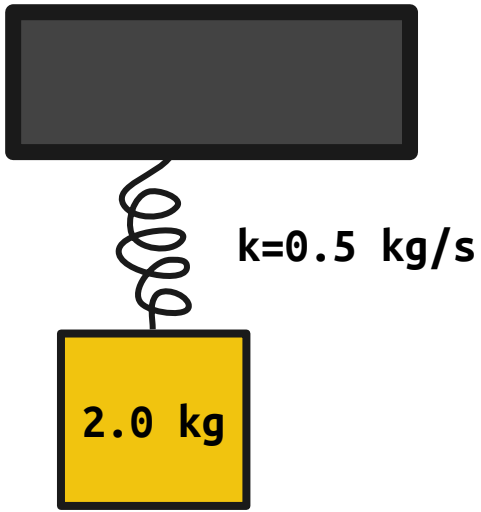




Input

Dynamical System

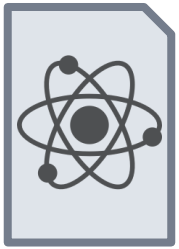
Harmonic Oscillator



$$v' = -0.25*p$$
$$p' = v$$

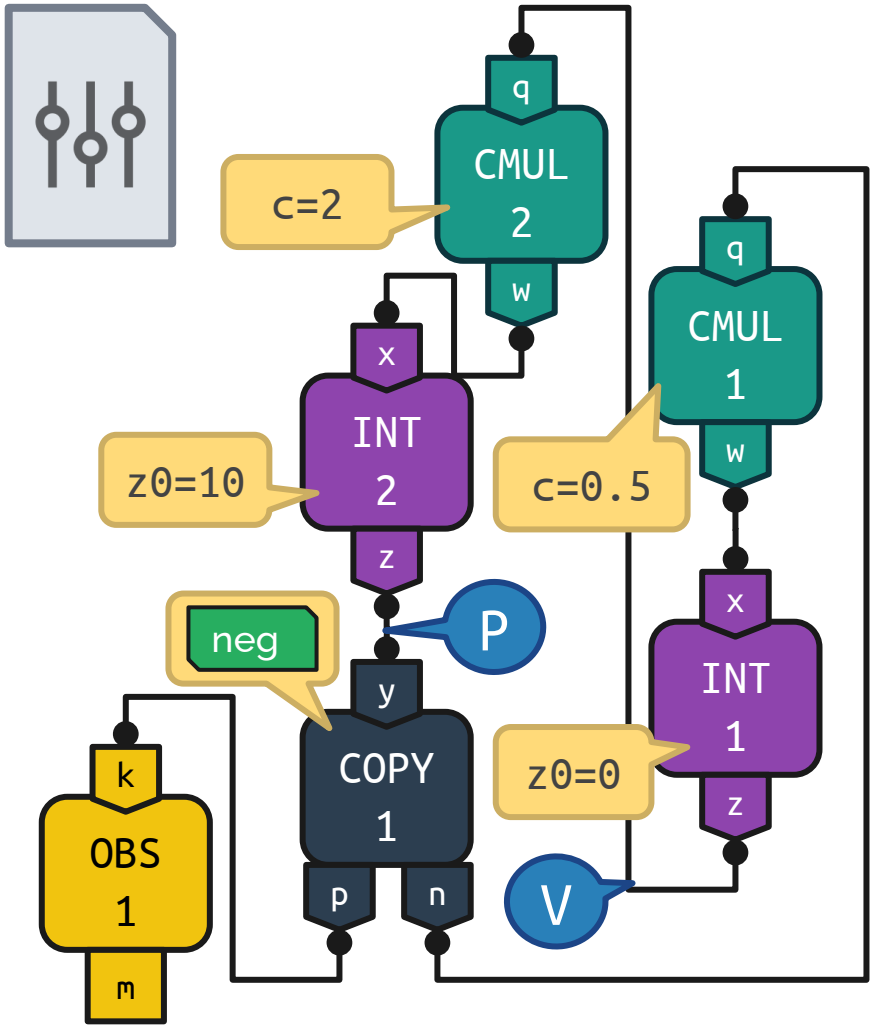
Input

Dynamical System Program



Oscillator

```
var v = ∫ -0.25*p dt_ds    v(0)=0
var p = ∫ v dt_ds          p(0)=10
observe p
```

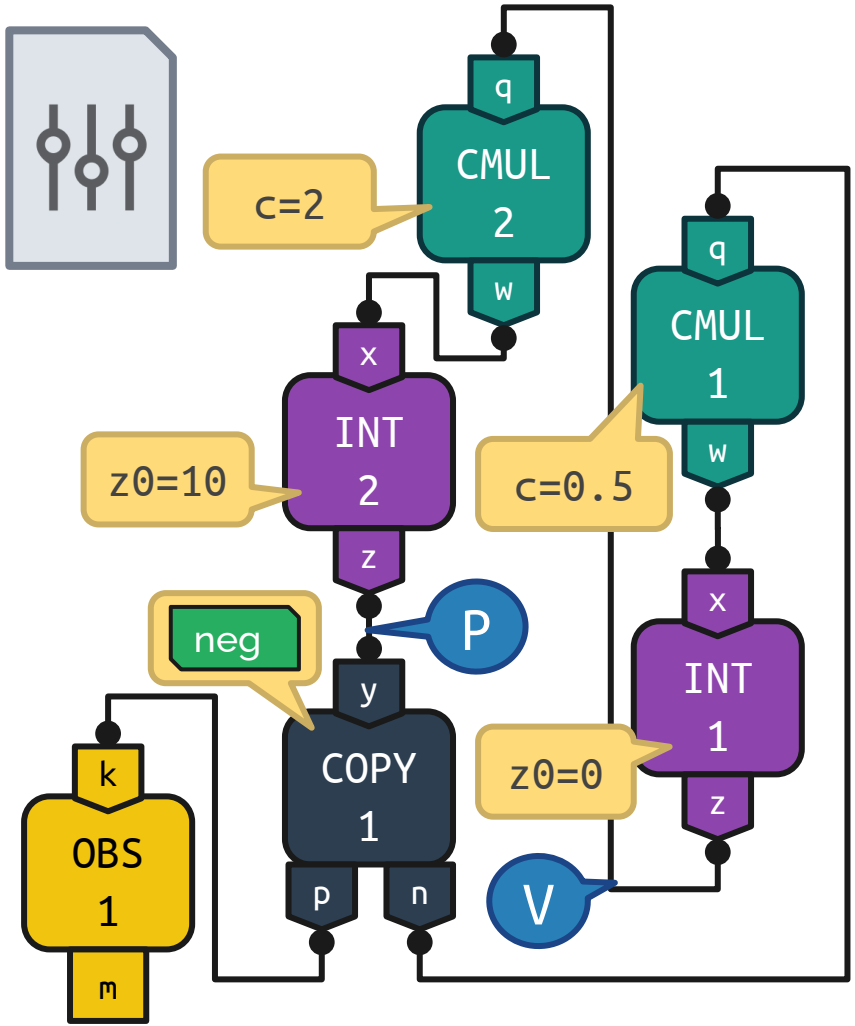


Oscillator

```
var v = ∫ -0.25*p dt_ds     v(0)=0
var p = ∫ v dt_ds           p(0)=10
observe p
```

Output

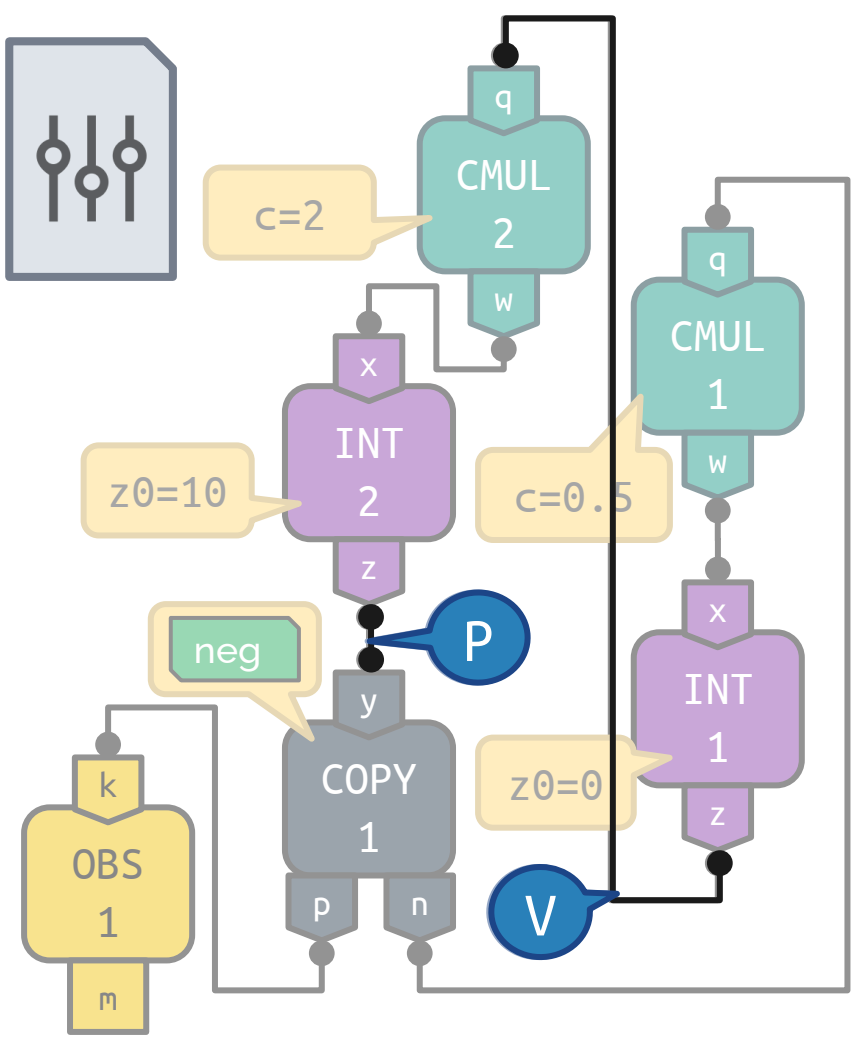
Configured Analog Device



Output

Configured Analog Device

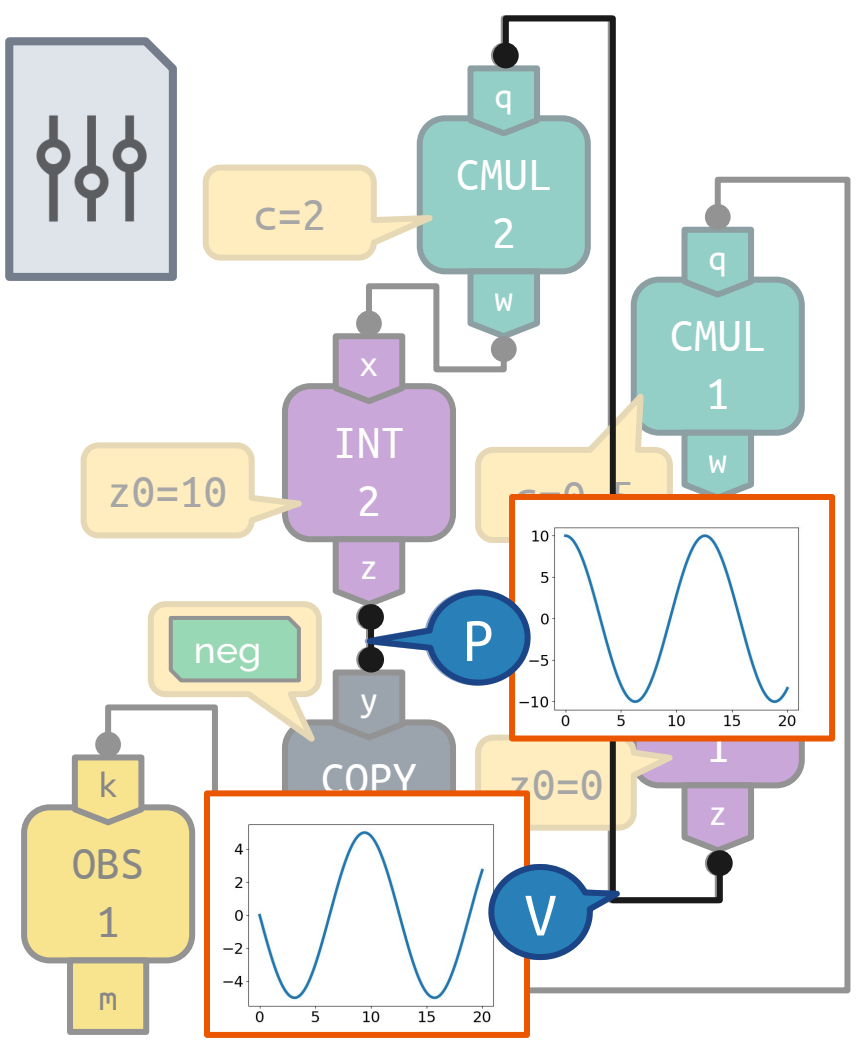
Analog currents map to position and velocity.



Output

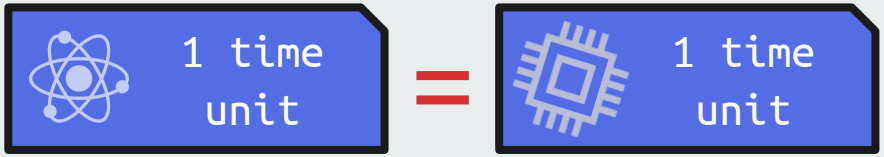
Configured Analog Device

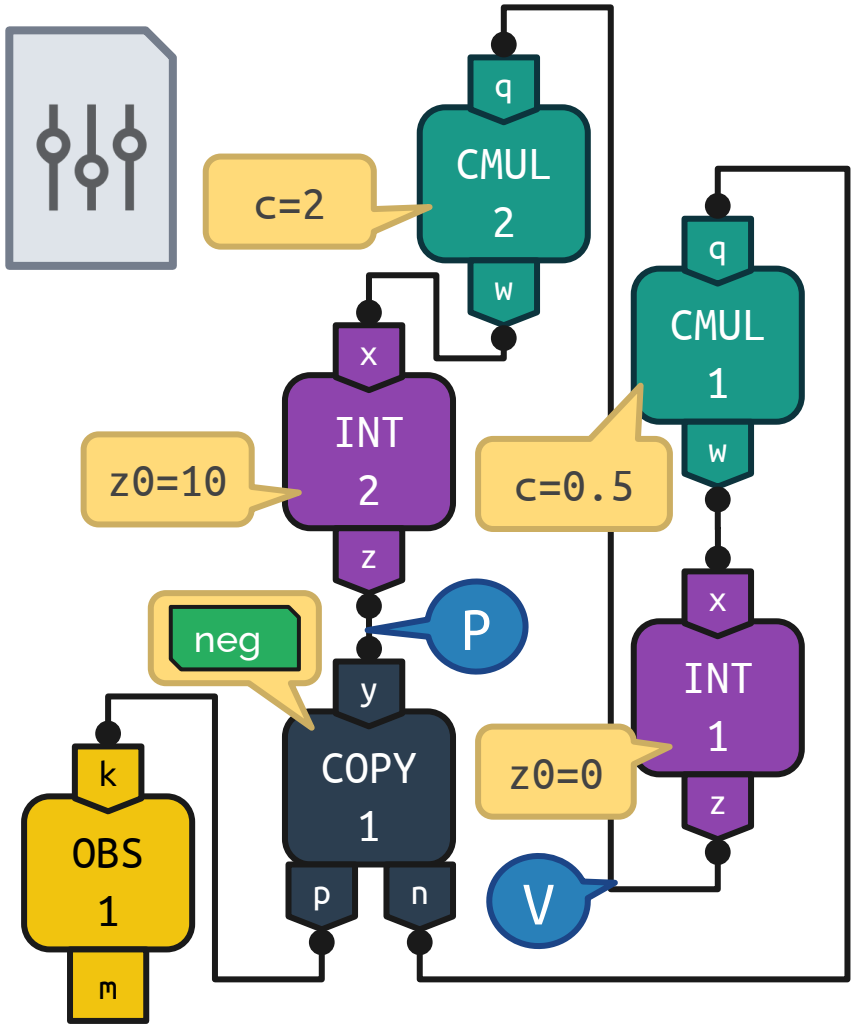
Analog currents map to position and velocity.



Output

Configured Analog Device



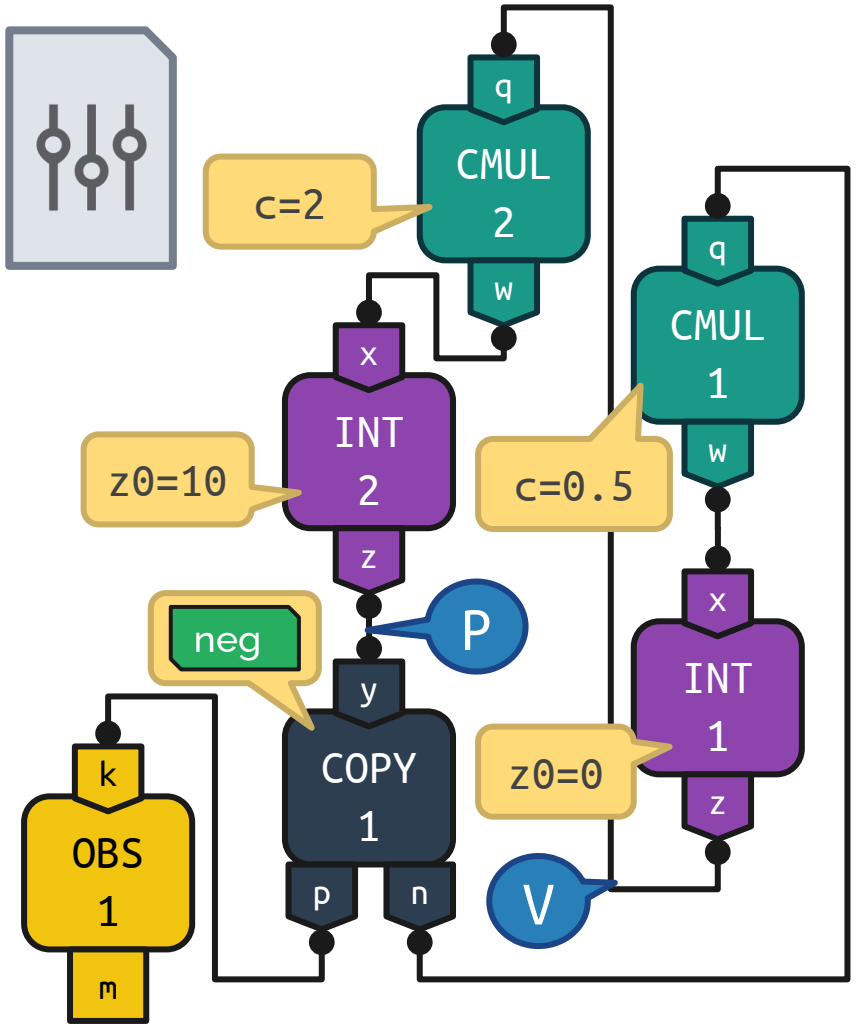


Output

Configured Analog Device

1 time unit = 1 time unit

7.93 µs

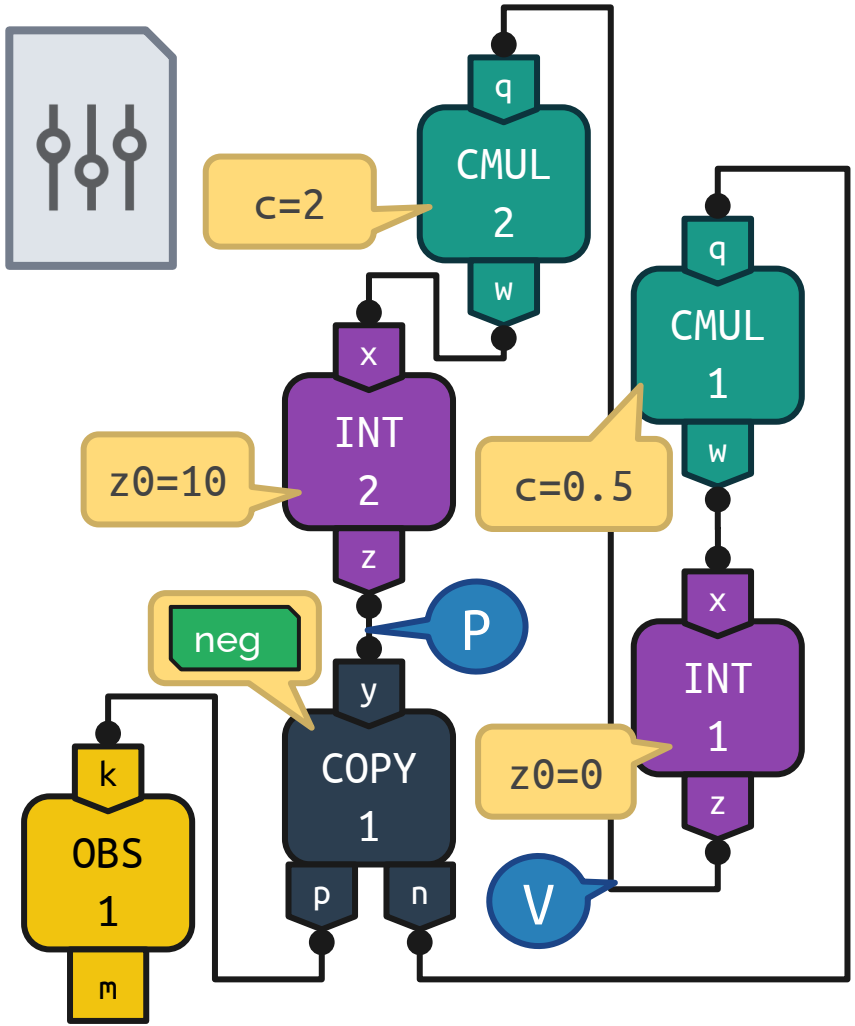


Output

Configured Analog Device

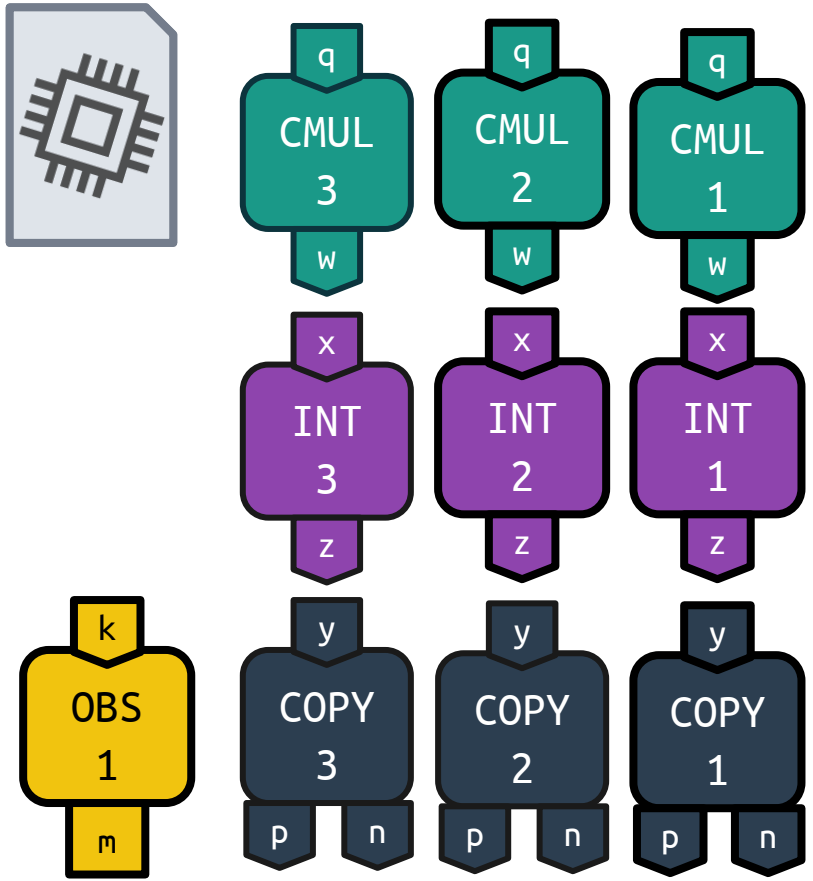
20 time units

158.6 µs



Inputs

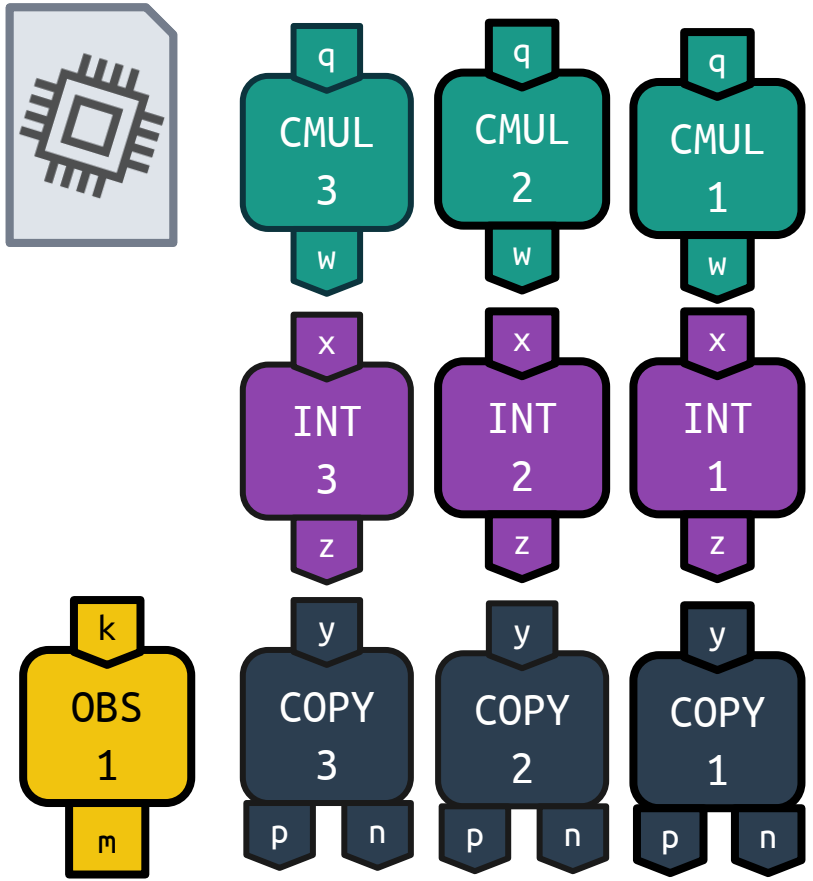
Analog Device



Inputs

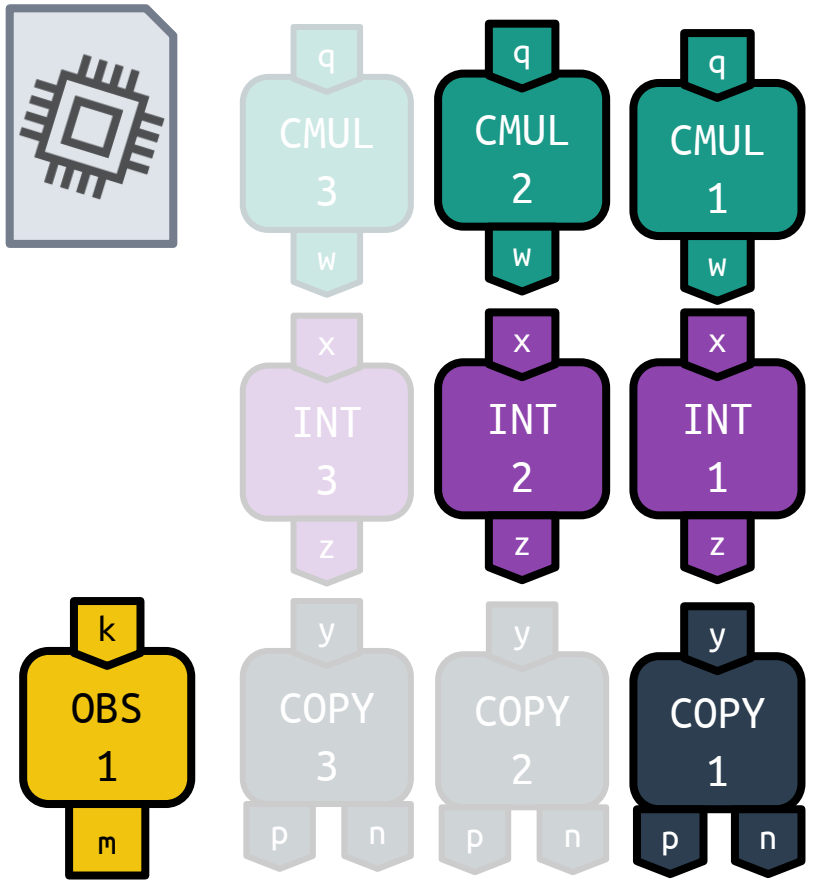
Analog Device

Any block can be connected to any other block.



Inputs

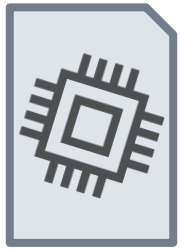
Analog Device



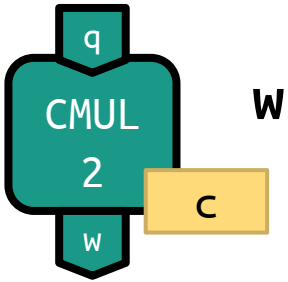
Inputs

Programming Interface

Digitally settable constants

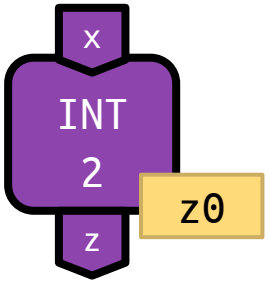


Multiplier



$$w = c*q$$

Integrator



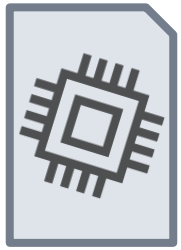
$$z=\int 0.5*x\ dt_{hw}$$

$$z(0)=z0$$

Inputs

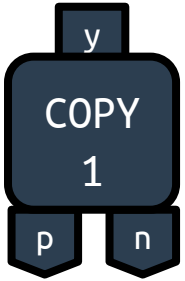
Programming Interface

Digitally settable block modes



Digitally Settable Block Modes

pos neg

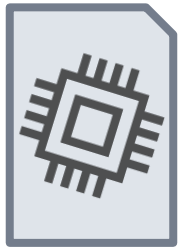



```
p = y
n = y
```

Inputs

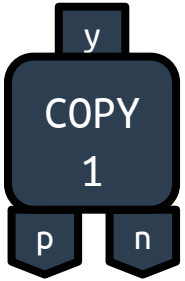
Programming Interface

Digitally settable block modes



Digitally Settable Block Modes

pos neg

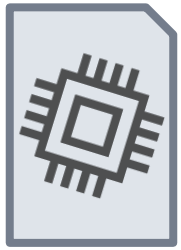


p = x

n = -x

Inputs

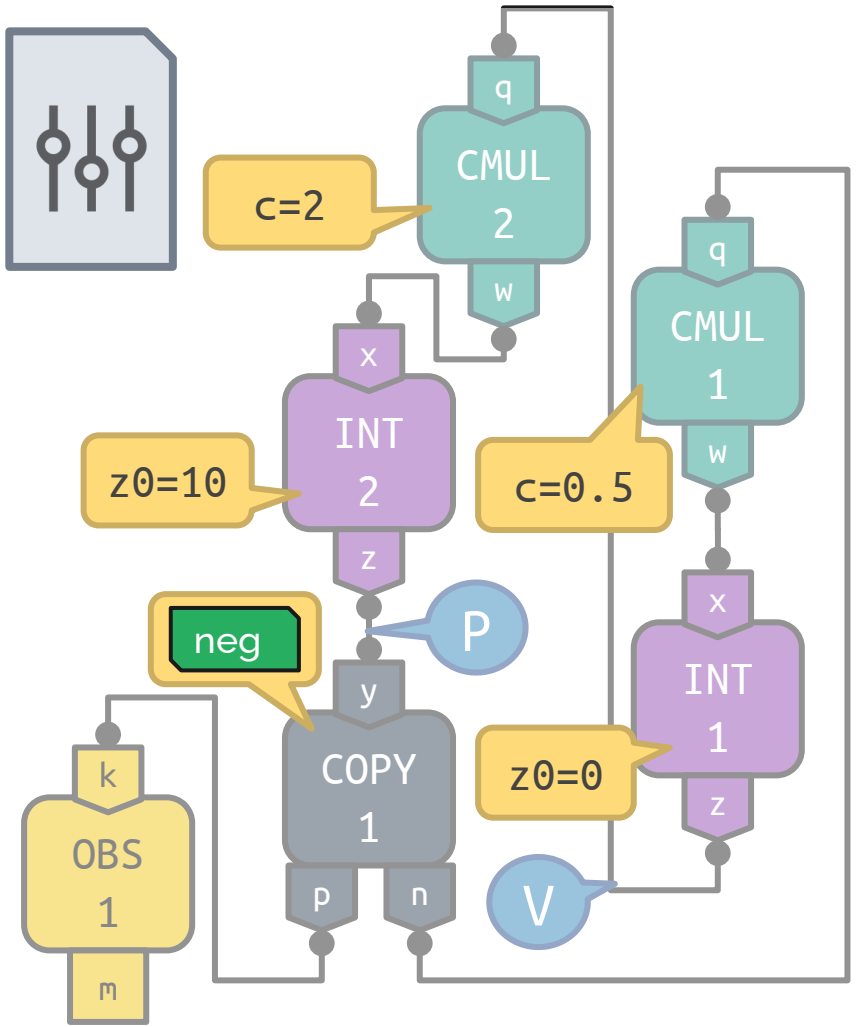
Observing Signals



Output

Analog Device Configuration

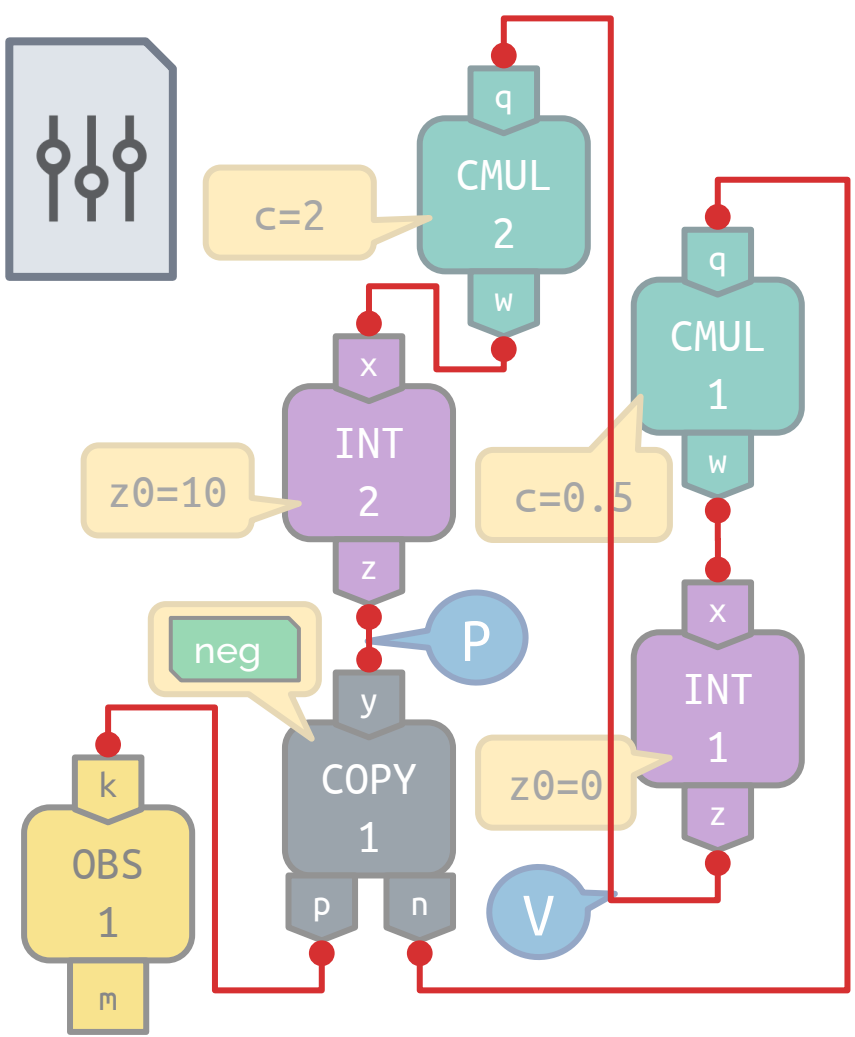
Programs each block in configuration



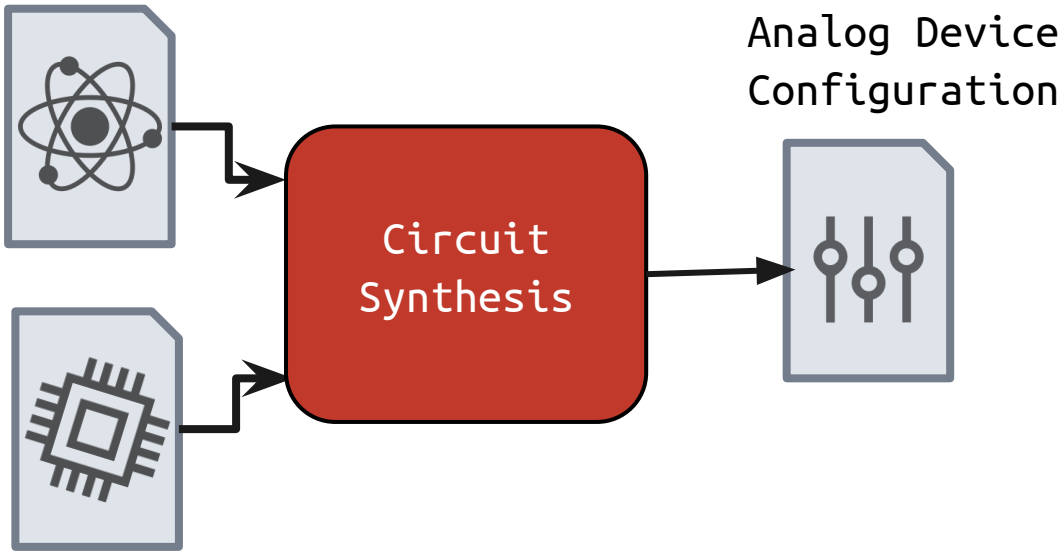
Output

Configured Analog Device

Selects connections to enable on device.

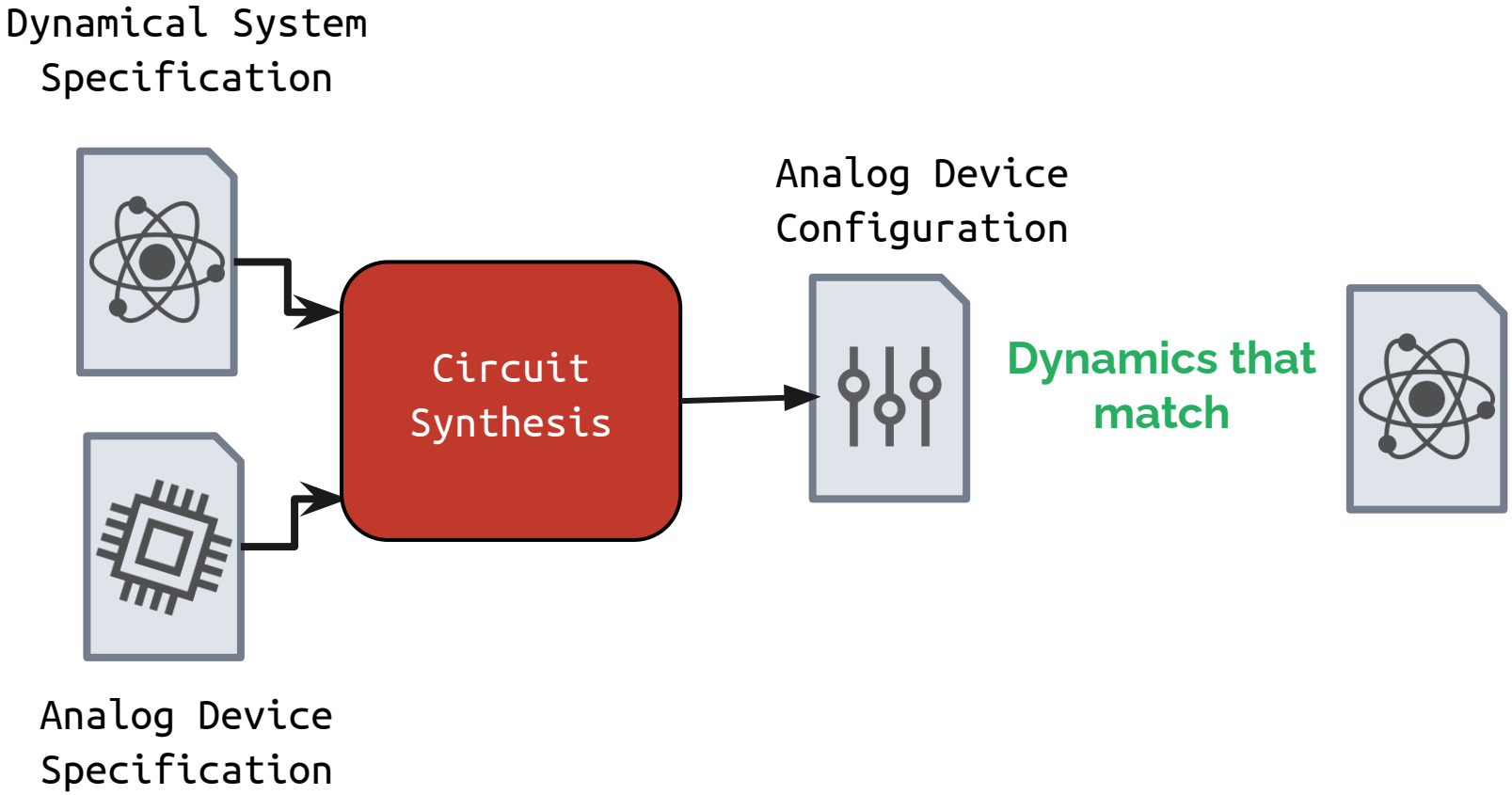


Dynamical System
Specification

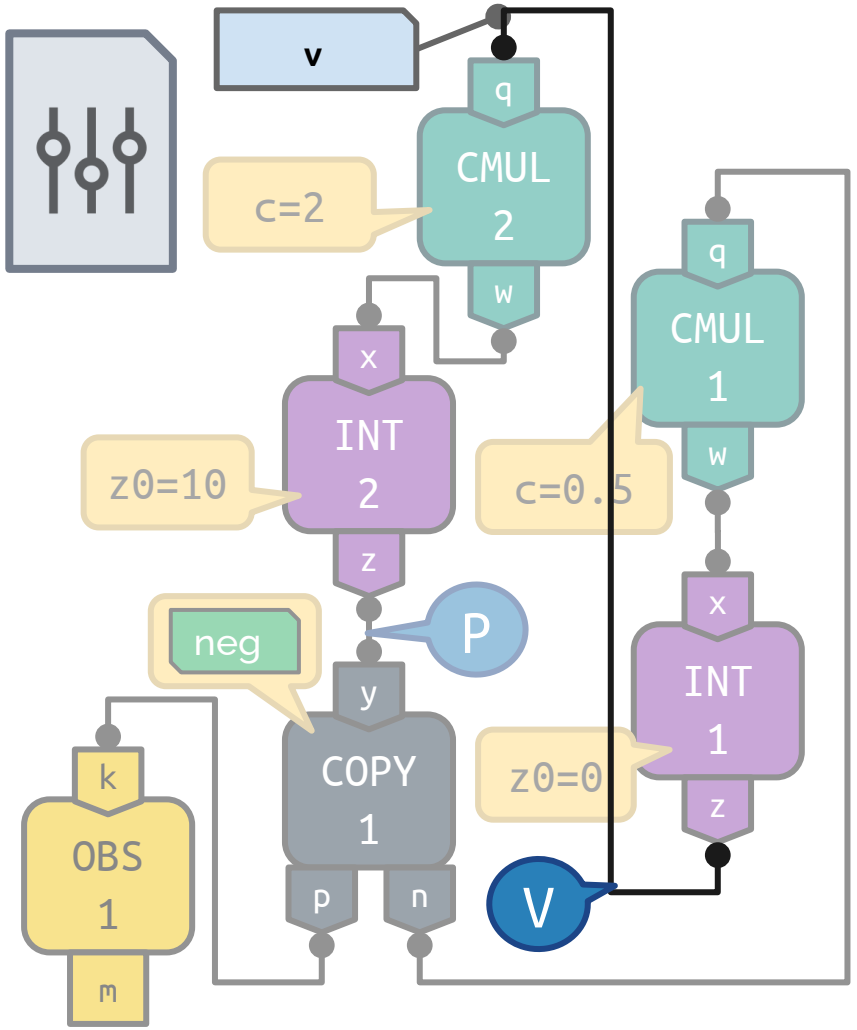


Analog Device
Configuration

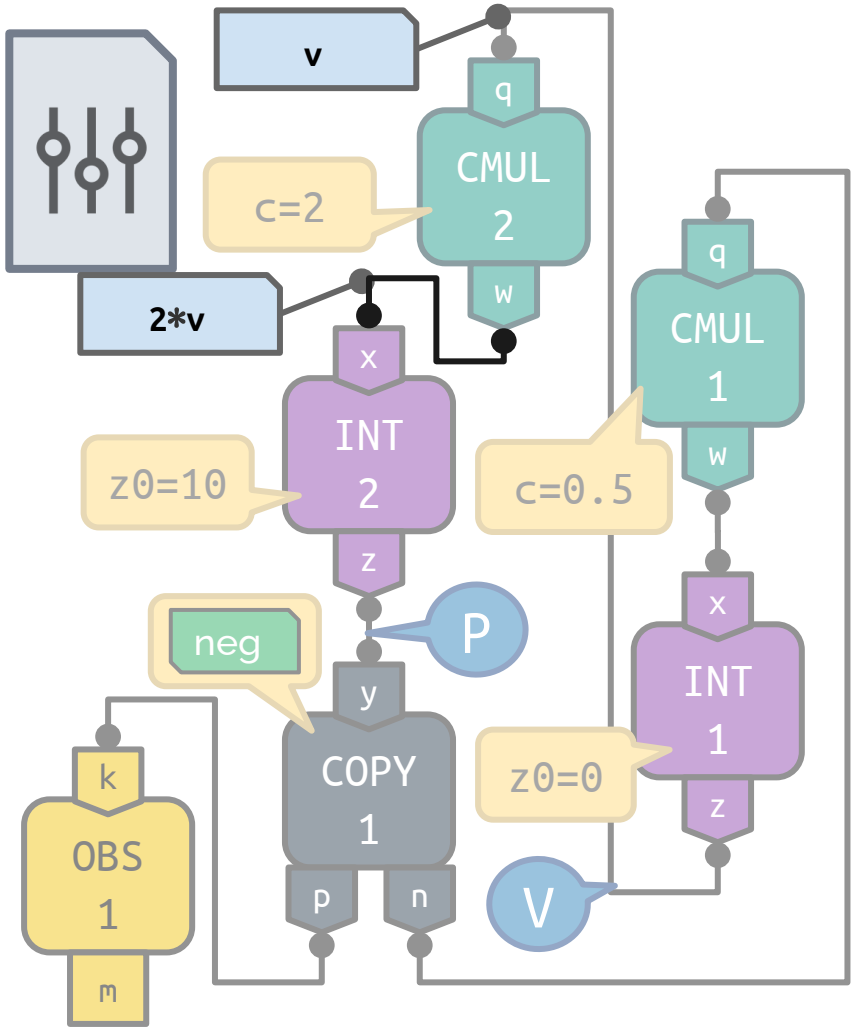
Analog Device
Specification


Dynamical System
Specification
Analog Device
Configuration
Circuit
Synthesis
Dynamics that
match
Analog Device
Specification

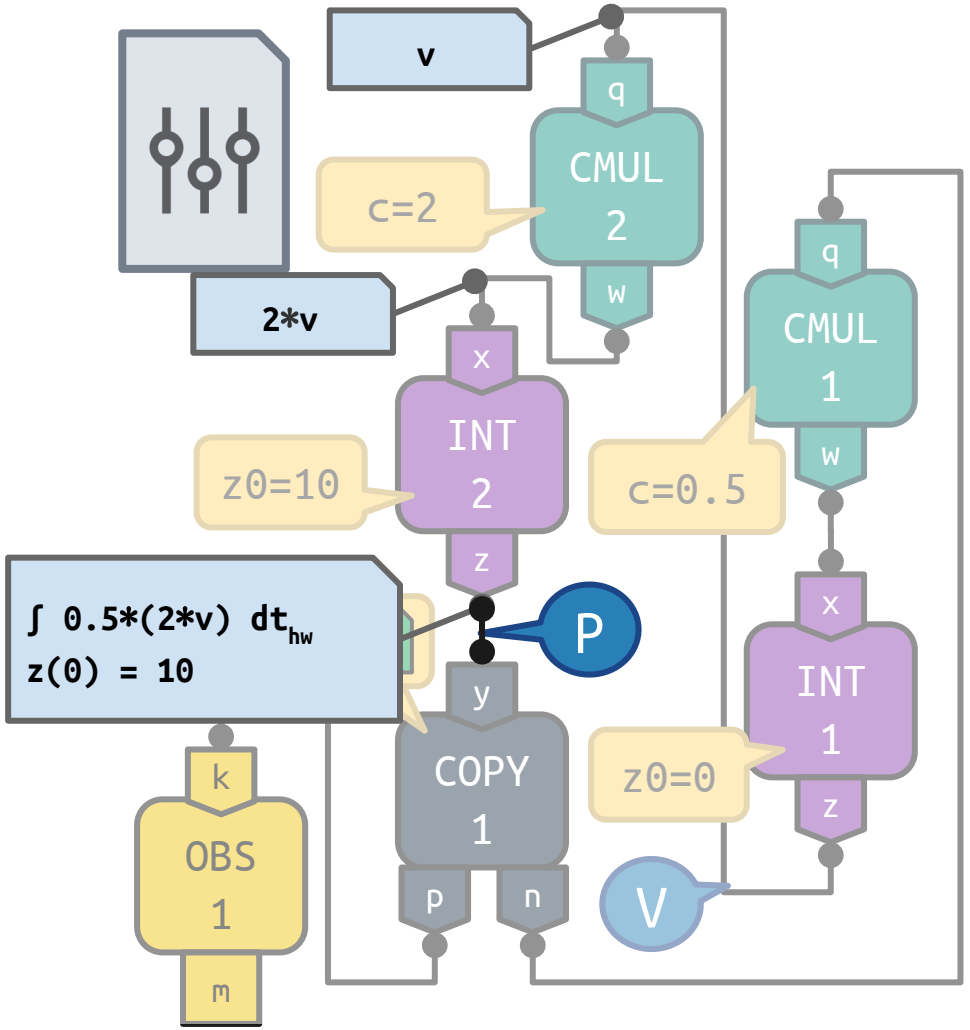
Matching Dynamics



Matching Dynamics



Matching Dynamics

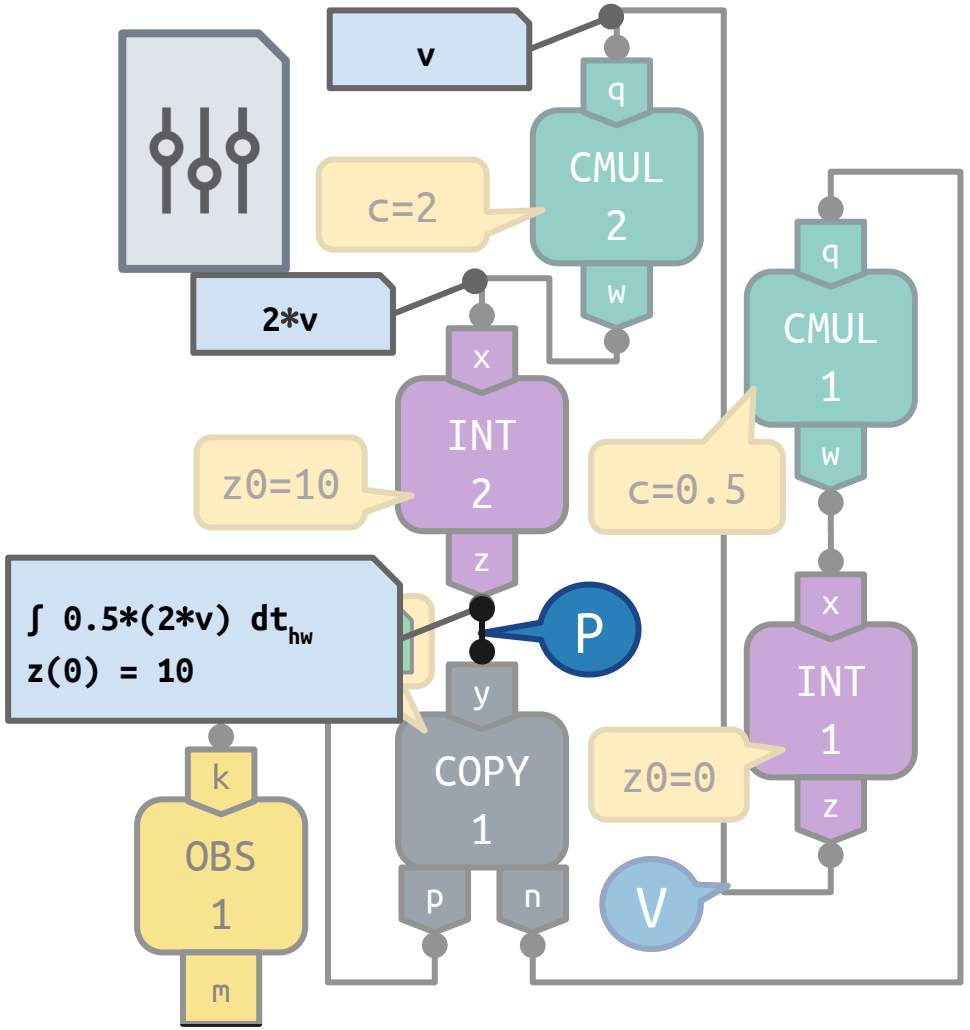


Matching Dynamics

P

$p=\int 0.5*(2*v)\ dt_{hw}$

$p(0)=10$



Matching Dynamics

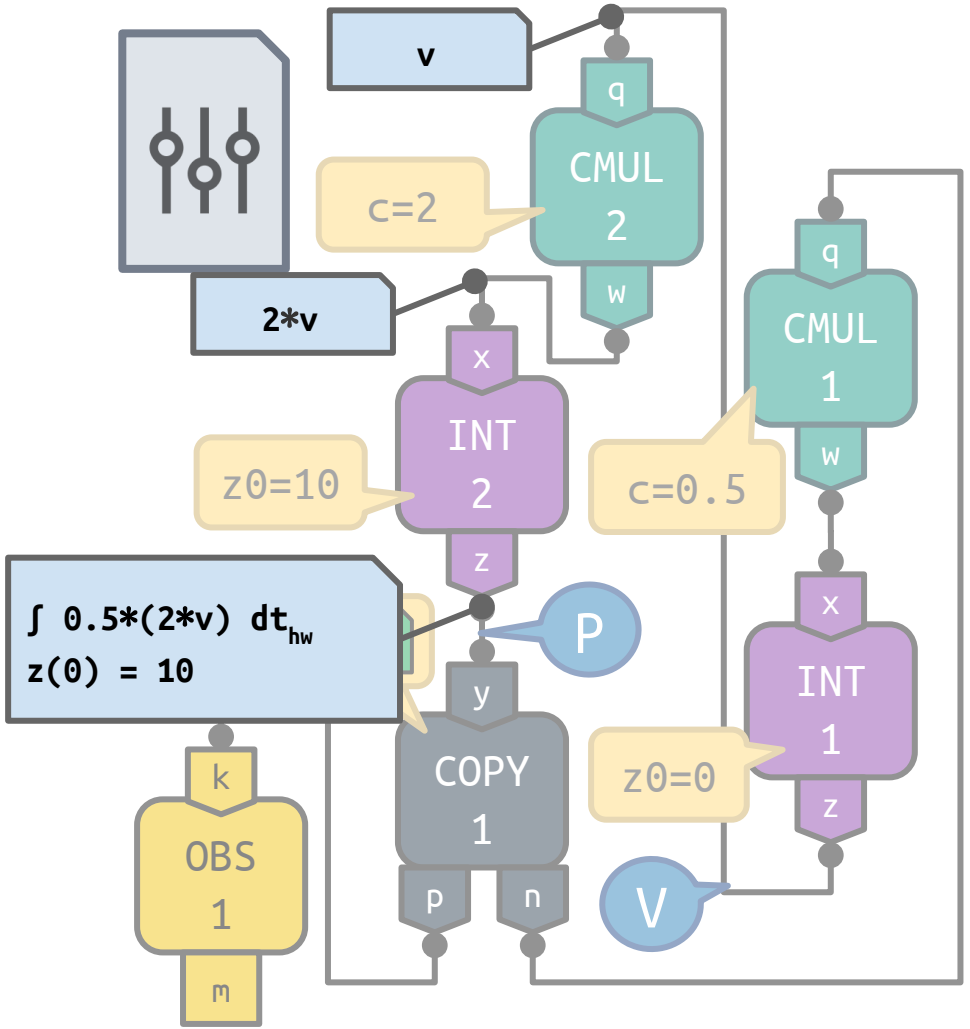
P

$p=\int 0.5*(2*v)\ dt_{hw}$

$p(0)=10$

$\text{var } p = \int v\ dt_{ds}$

$p(0)=10$



Matching Dynamics

P

$p=\int 0.5*(2*v)\ dt_{hw}$

$p(0)=10$

$\equiv$

$\text{var } p = \int v\ dt_{ds}$

$p(0)=10$

V

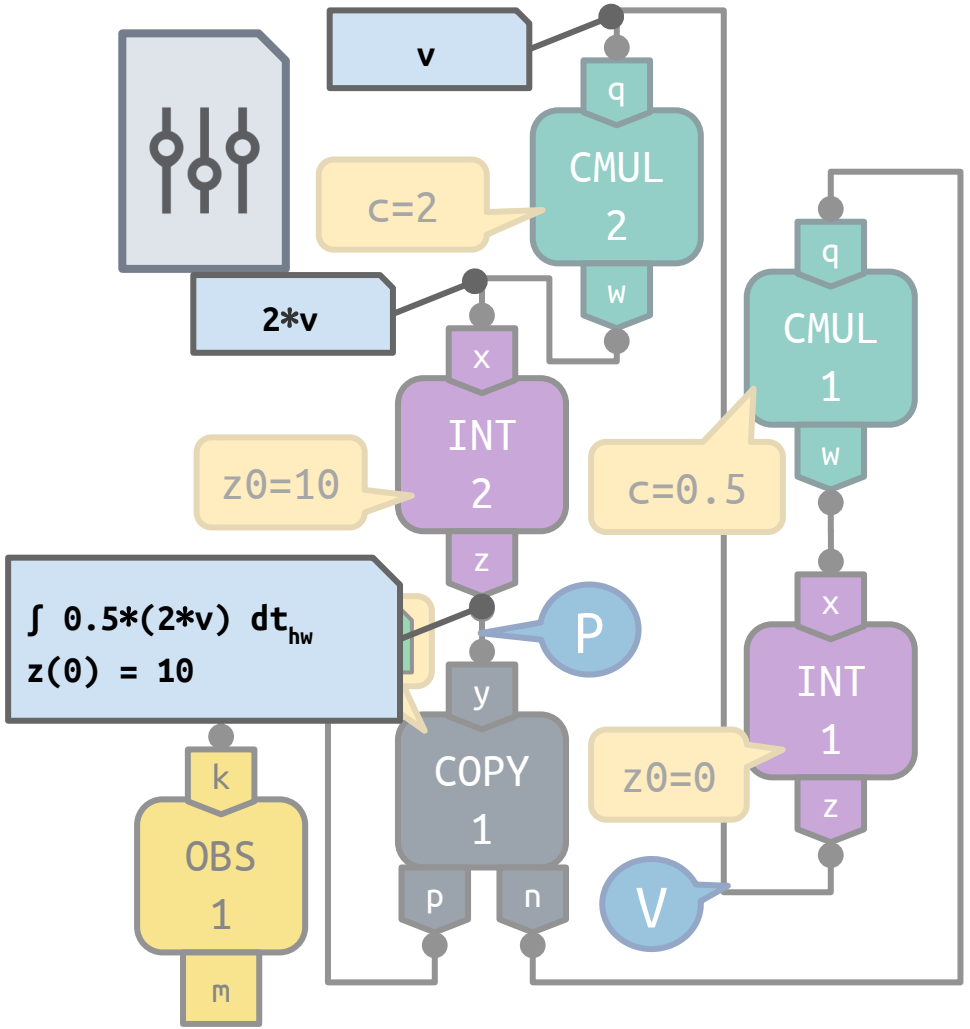
$v=\int 0.5*(0.5*(-p))\ dt_{hw}$

$v(0)=0$

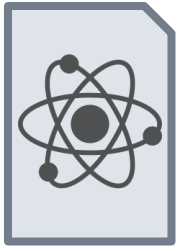
$\equiv$

$\text{var } v = \int -0.25*p\ dt_{ds}$

$v(0)=0$



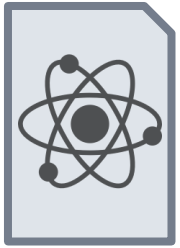
Circuit Synthesis



```
var v = ∫ -0.25*p dt_ds    v(0)=0
var p = ∫ v dt_ds          p(0)=10
observe p
```

Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route

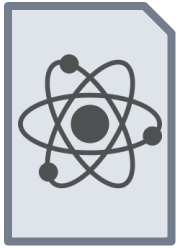


```
var v = ∫ -0.25*p dt_ds    v(0)=0
observe p
```

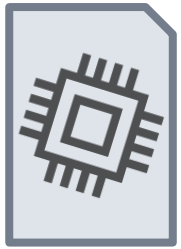
```
var p = ∫ v dt_ds          p(0)=10
```

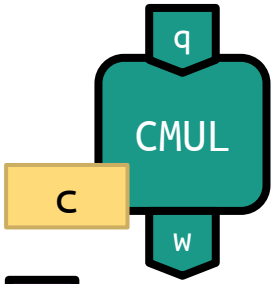
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route

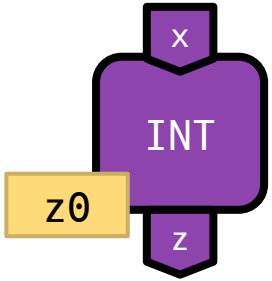


```
var v = ∫ -0.25*p dt_ds    v(0)=0
observe p
```






```
w =c*q
```

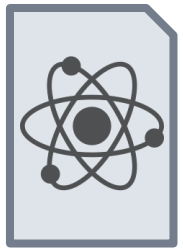



```
z=∫ 0.5*x dt_hw
z(0)=z0
```

```
var p = ∫ v dt_ds      p(0)=10
```

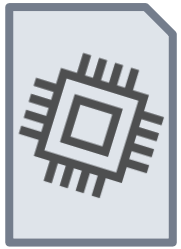
Circuit Synthesis

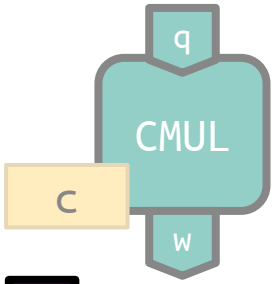
Subcircuit Synthesis → Assembly → Place and Route



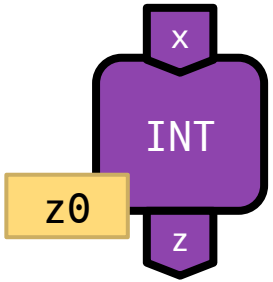
var v = ∫ -0.25*p dt_{ds} v(0)=0

observe p



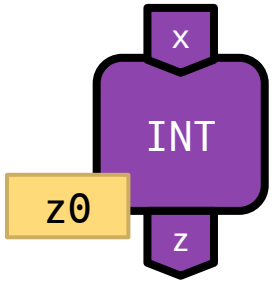


w =c*q



z=∫ 0.5*x dt_{hw}
z(0)=z0

Block Selection

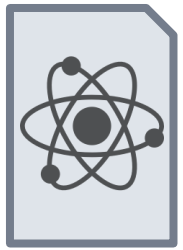


z=∫ 0.5*x dt_{hw}
z(0)=z0

var p = ∫ v dt_{ds} p(0)=10

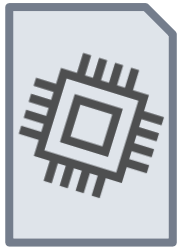
Circuit Synthesis

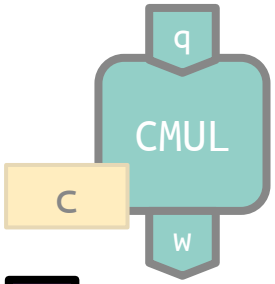
Subcircuit Synthesis → Assembly → Place and Route



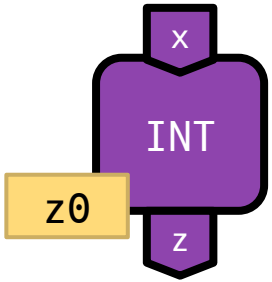
var v = ∫ -0.25*p dt_{ds} v(0)=0

observe p



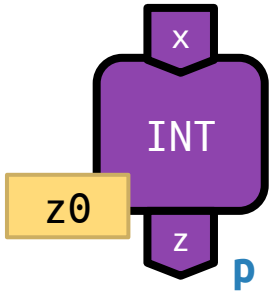


w =c*q



z=∫ 0.5*x dt_{hw}
z(0)=z0

Unification



z=∫ 0.5*x dt_{hw}
z(0)=z0

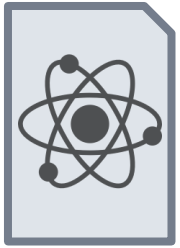
var p = ∫ v dt_{ds} p(0)=10

Goal

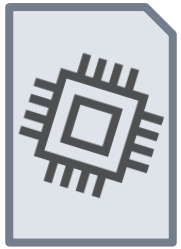
z = p

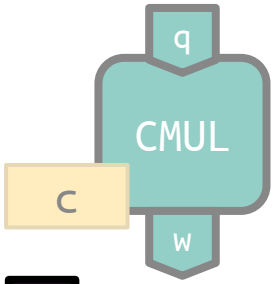
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route

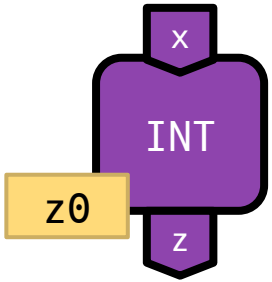


```
var v = ∫ -0.25*p dt_ds    v(0)=0
observe p
```



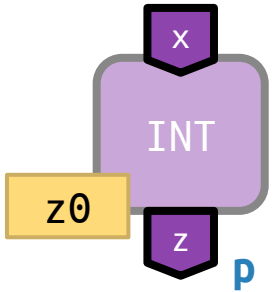



```
w =c*q
```




```
z=∫ 0.5*x dt_hw
z(0)=z0
```

Unification




```
z=∫ 0.5*x dt_hw
z(0)=z0
```

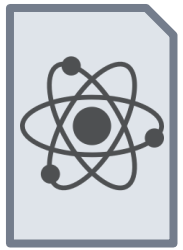
```
var p = ∫ v dt_ds        p(0)=10
```

Goal

```
z = p
```

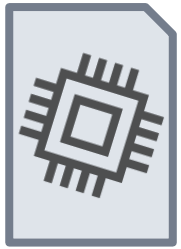
Circuit Synthesis

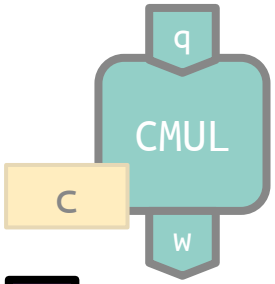
Subcircuit Synthesis → Assembly → Place and Route



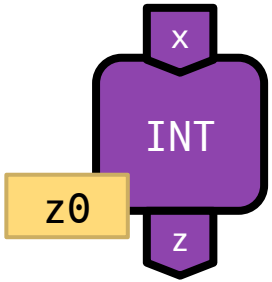
$$\text{var } v = \int -0.25*p \; dt_{ds} \qquad v(0)=0$$

$$\text{observe } p$$



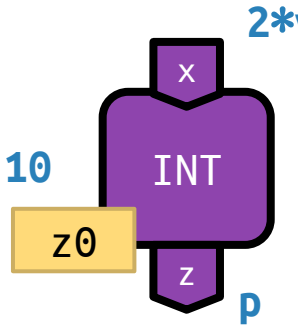


$$w = c*q$$



$$z = \int 0.5*x \; dt_{hw} \qquad z(0)=z0$$

Unification



$$z = \int 0.5*2*v \; dt_{hw} \qquad z(0)=10$$

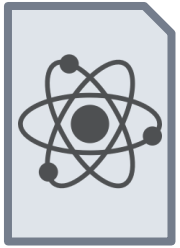
$$\text{var } p = \int v \; dt_{ds} \qquad p(0)=10$$

Goal

$$z = p$$

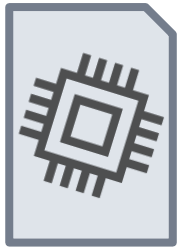
Circuit Synthesis

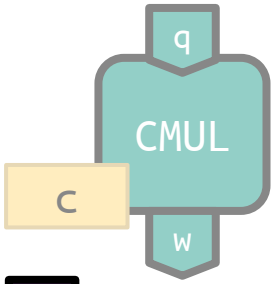
Subcircuit Synthesis → Assembly → Place and Route



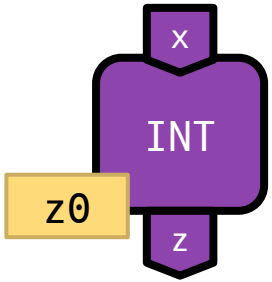
var v = ∫ -0.25*p dt_{ds} v(0)=0

observe p



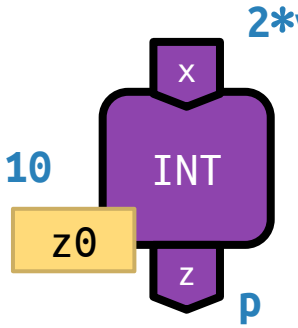


w =c*q



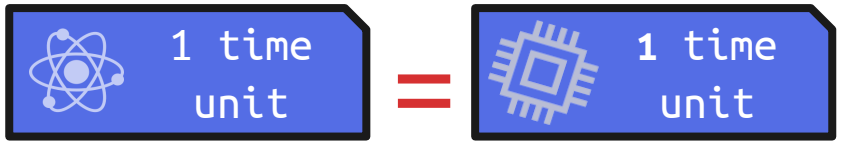
z=∫ 0.5*x dt_{hw}
z(0)=z0

Unification



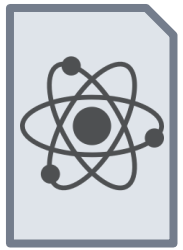
z=∫ 0.5*2*v dt_{hw}
z(0)=10

var p = ∫ v dt_{ds} p(0)=10



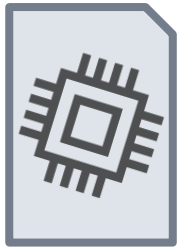
Circuit Synthesis

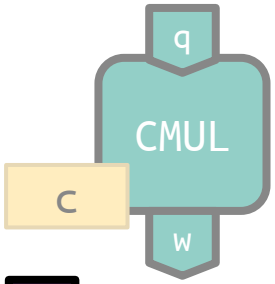
Subcircuit Synthesis → Assembly → Place and Route



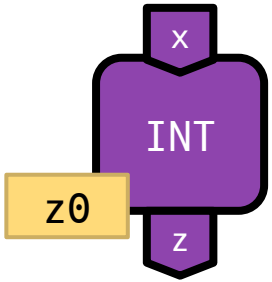
$$\text{var } v = \int -0.25*p \; dt_{ds} \qquad v(0)=0$$

$$\text{observe } p$$



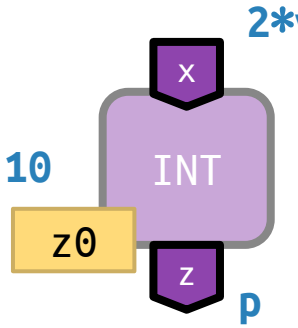


$$w = c*q$$



$$z = \int 0.5*x \; dt_{hw} \qquad z(0)=z0$$

Unification



$$z = \int 0.5*2*v \; dt_{ds} \qquad z(0)=10$$

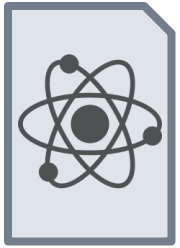
$$\text{var } p = \int v \; dt_{ds} \qquad p(0)=10$$

Goal

$$z = p$$

Circuit Synthesis

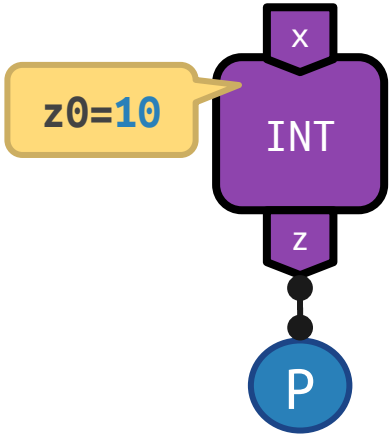
Subcircuit Synthesis → Assembly → Place and Route

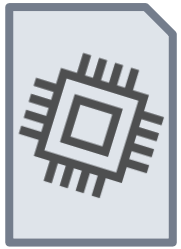


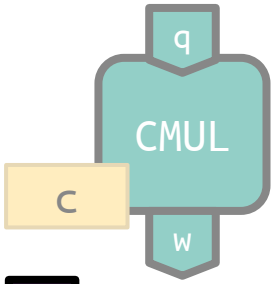
$$\text{var } v = \int -0.25*p \; dt_{ds} \qquad v(0)=0$$

observe p

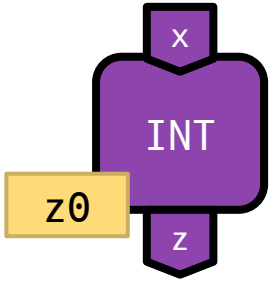
$$x = 2*v$$







$$w = c*q$$

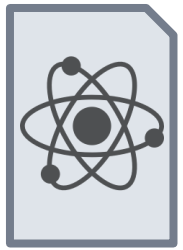


$$z = \int 0.5*x \; dt_{hw}$$

$$z(0)=z0$$

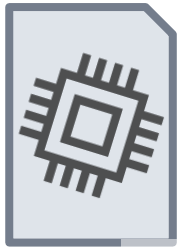
Circuit Synthesis

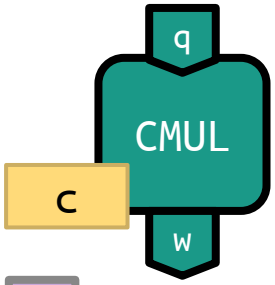
Subcircuit Synthesis → Assembly → Place and Route



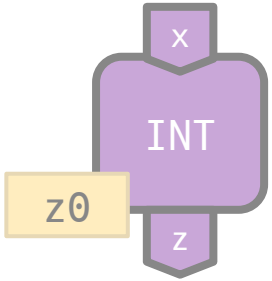
```
var v = ∫ -0.25*p dt_ds    v(0)=0
observe p
```

Block Selection



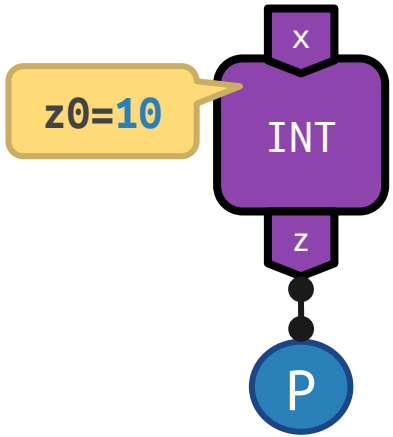



```
w =c*q
```



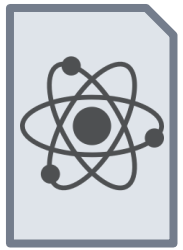

```
z=∫ 0.5*x dt_hw
z(0)=z0
```

```
x = 2*v
```

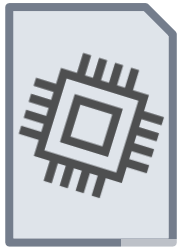


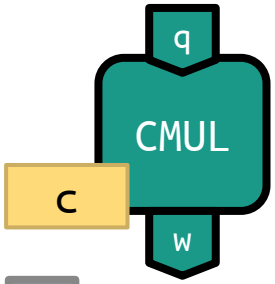
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route

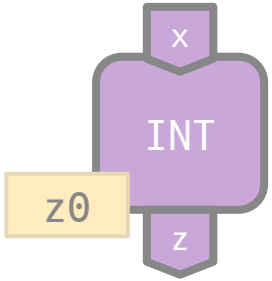


```
var v = ∫ -0.25*p dt_ds    v(0)=0
observe p
```





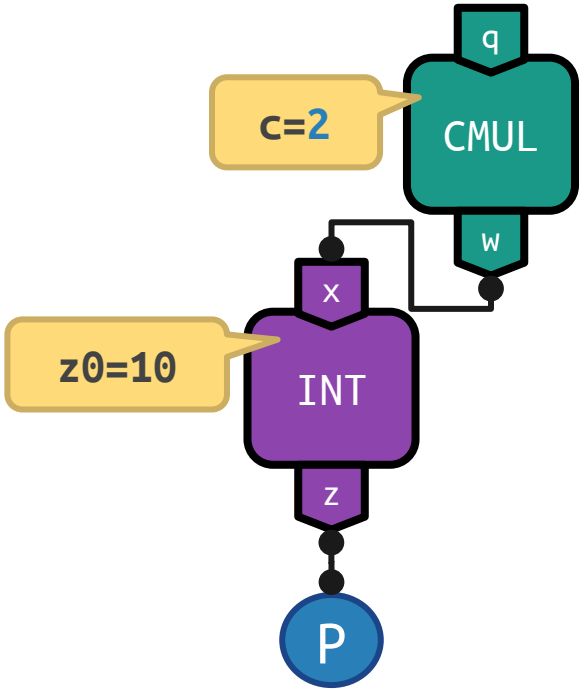

```
w =c*q
```




```
z=∫ 0.5*x dt_hw
z(0)=z0
```

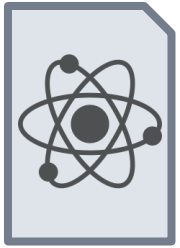
Unification

```
q = v
```

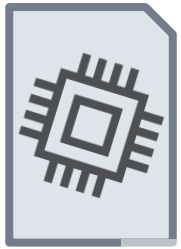


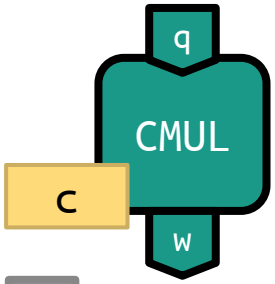
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route

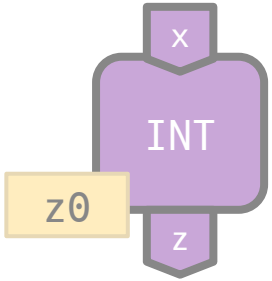


```
var v = ∫ -0.25*p dt_ds    v(0)=0
observe p
```



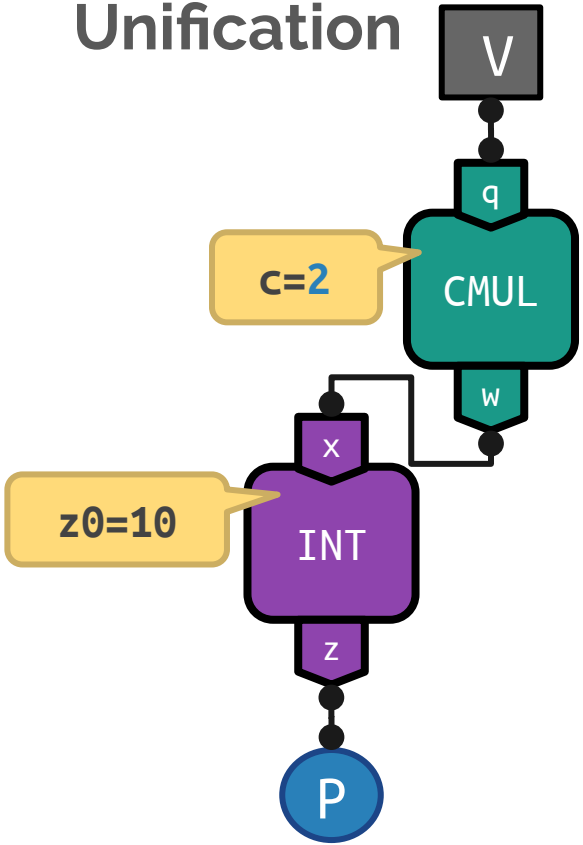



```
w =c*q
```



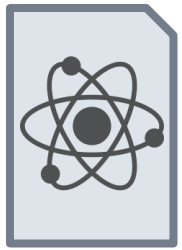

```
z=∫ 0.5*x dt_hw
z(0)=z0
```

Unification



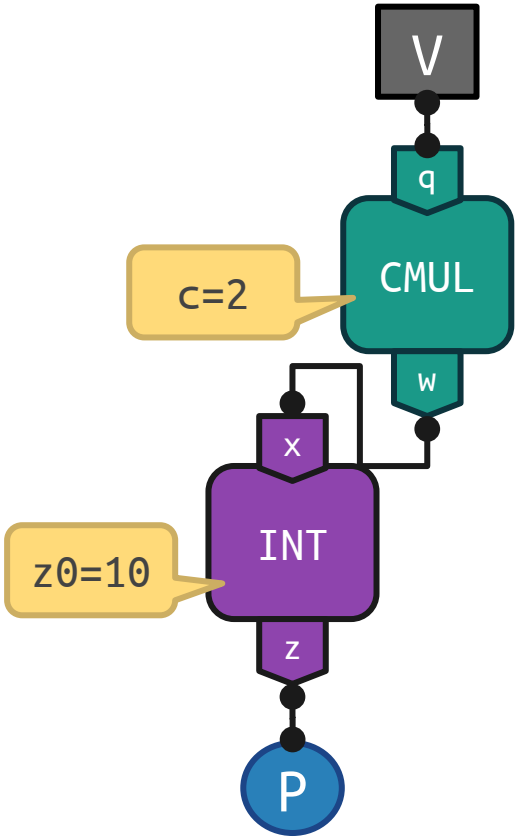
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



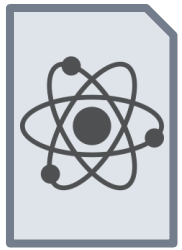
P

$$p = \int 0.5*(2*v)\ dt_{ds}$$
$$p(0)=10$$



Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



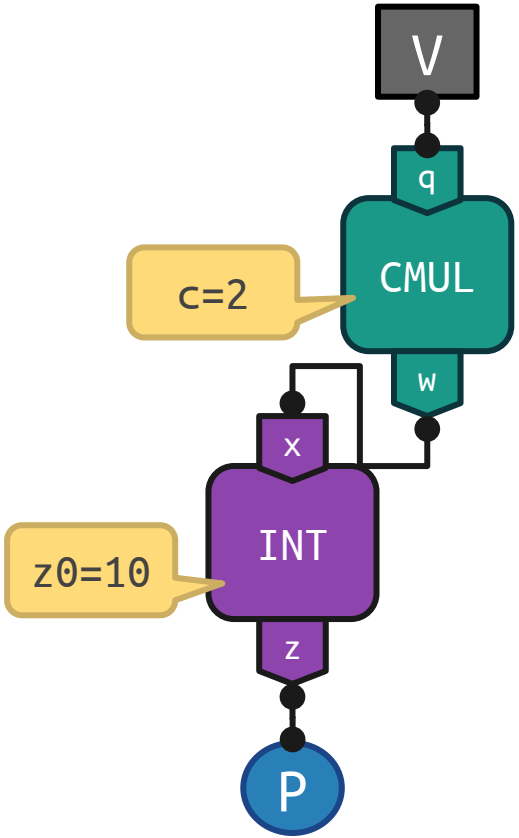
P

$p=\int 0.5*(2*v)\ dt_{ds}$

$p(0)=10$

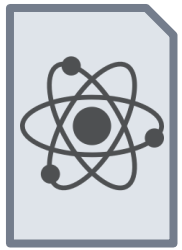
$p = \int v\ dt_{ds}$

$p(0)=10$



Circuit Synthesis

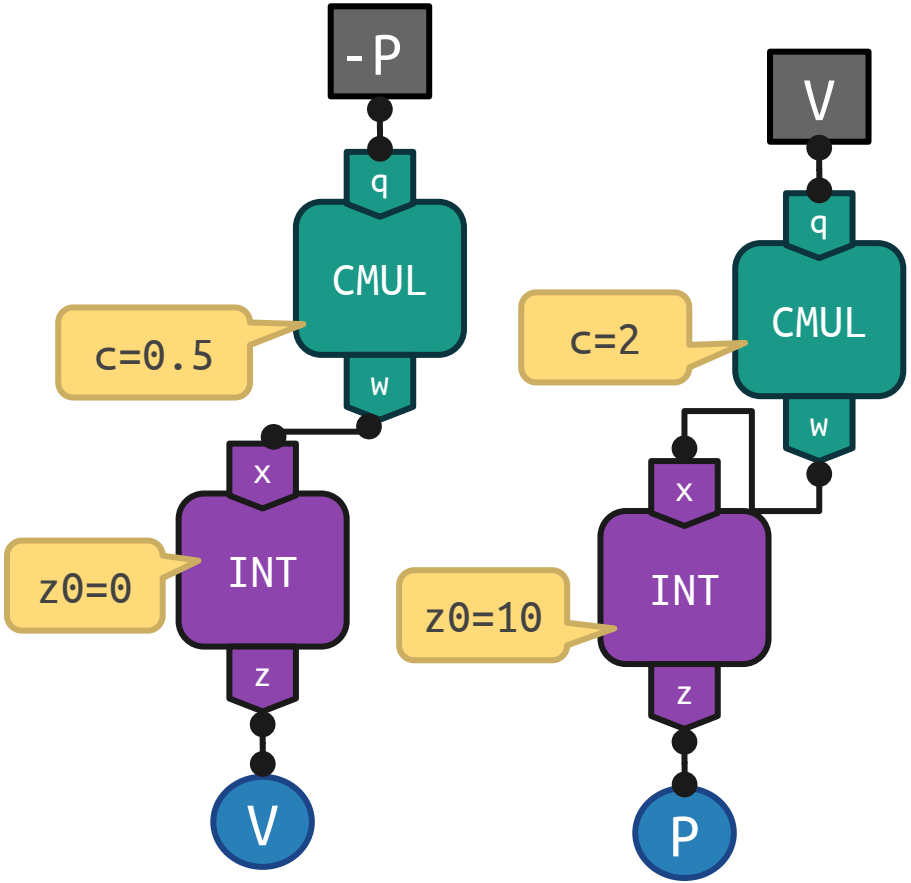
Subcircuit Synthesis → Assembly → Place and Route



V

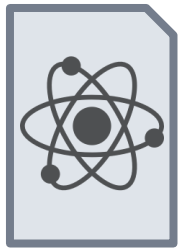
$v=\int 0.5*(0.5*(-p))\ dt_{ds}$

$v(0)=0$



Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



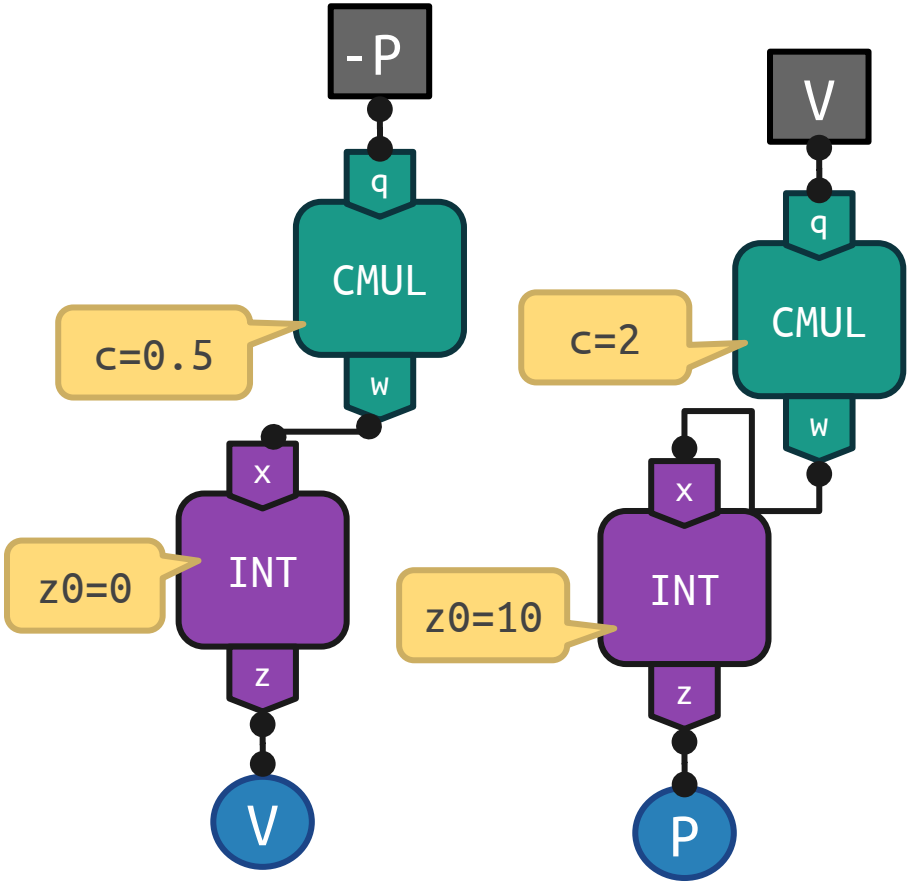
V

$v=\int 0.5*(0.5*(-p))\ dt_{ds}$

$v(0)=0$

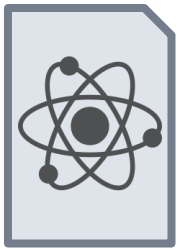
$v = \int -0.25*p\ dt_{ds}$

$v(0)=0$

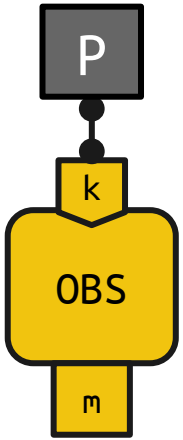


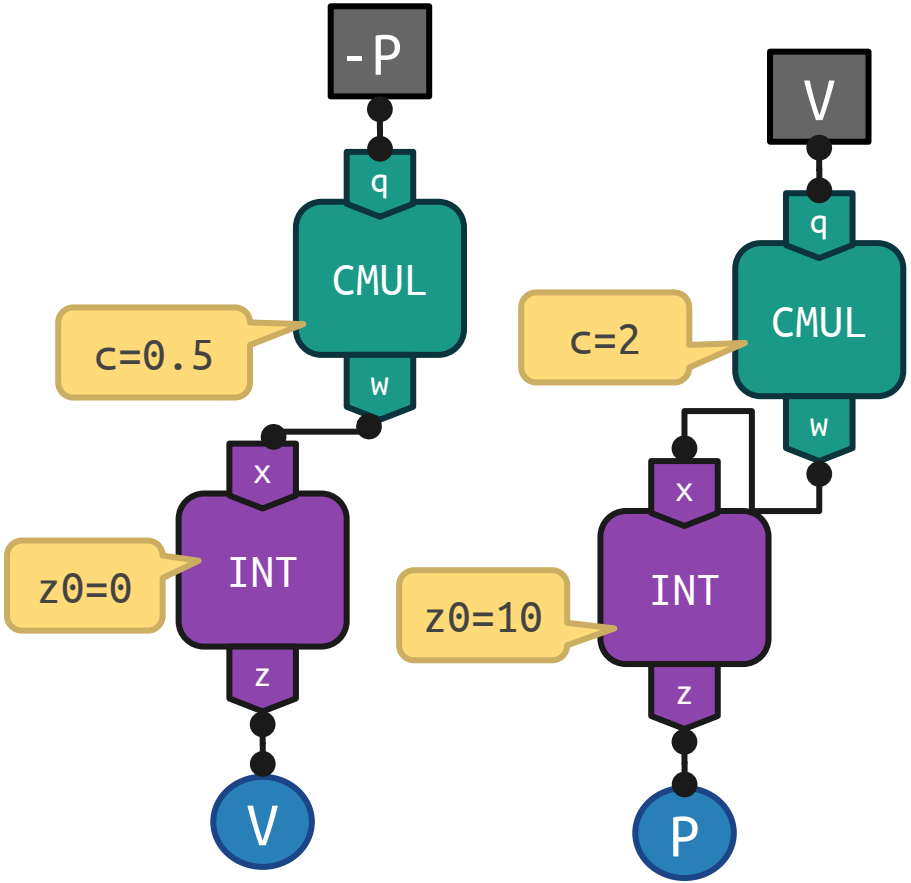
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



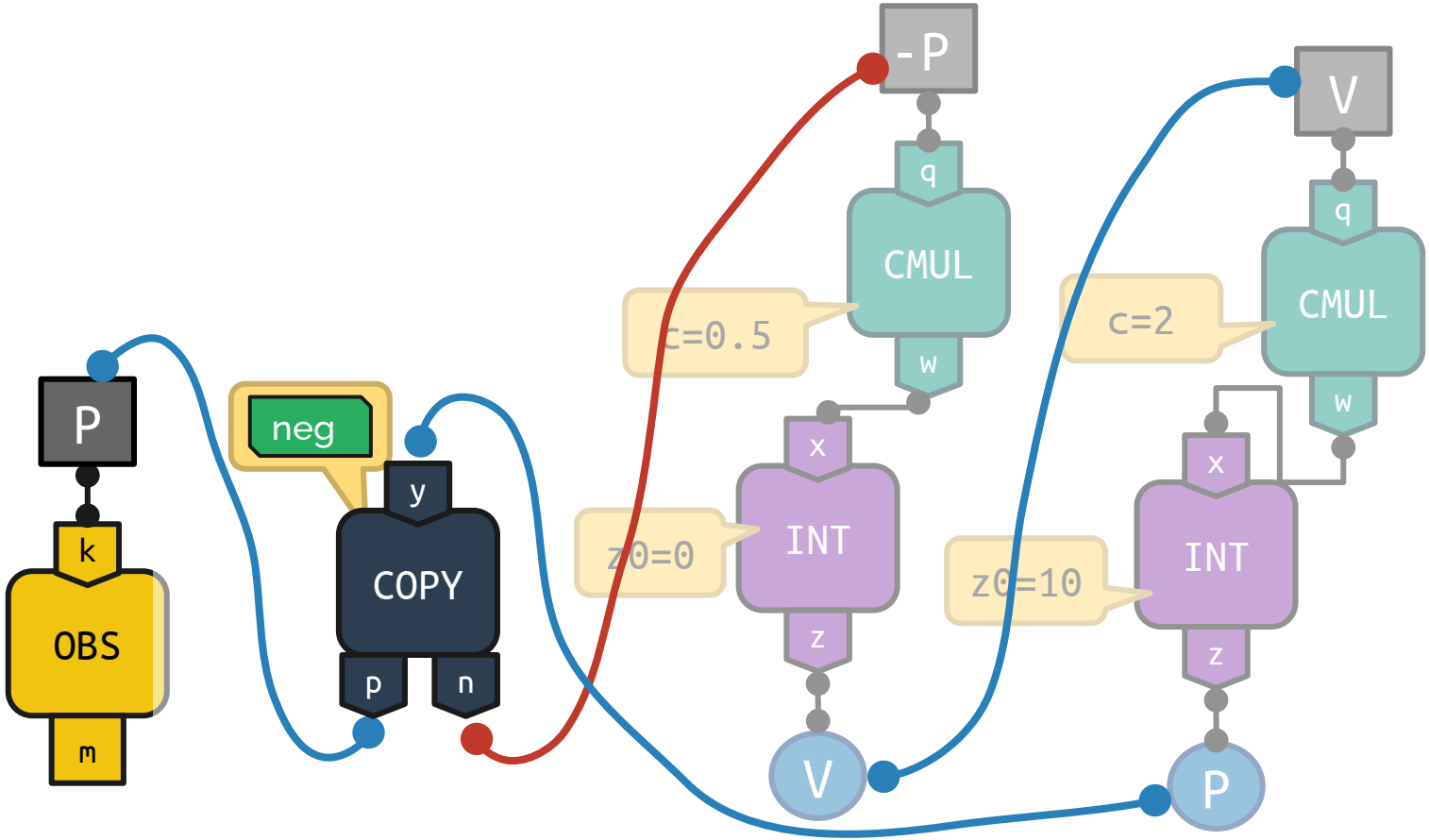
observe p





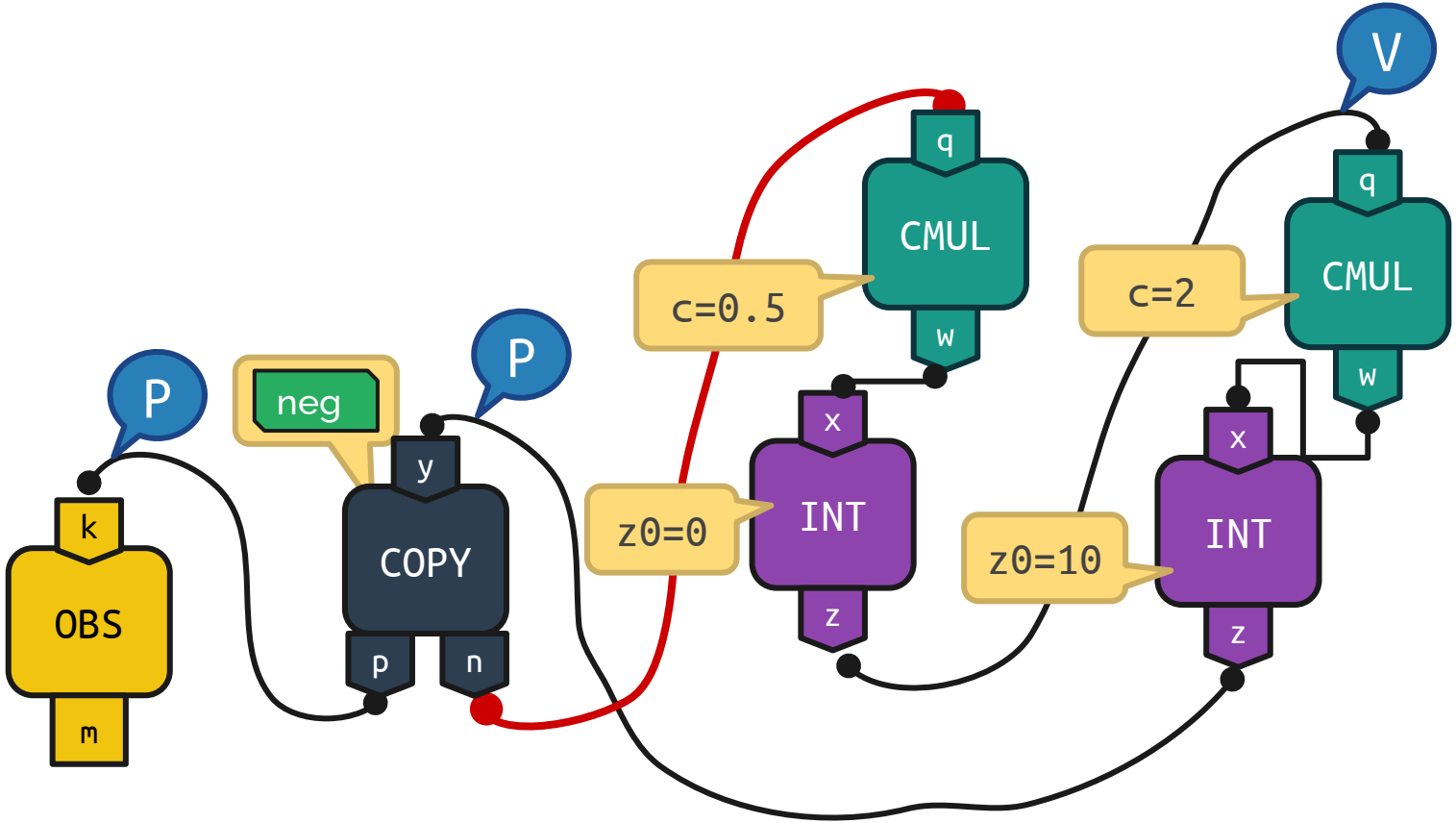
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



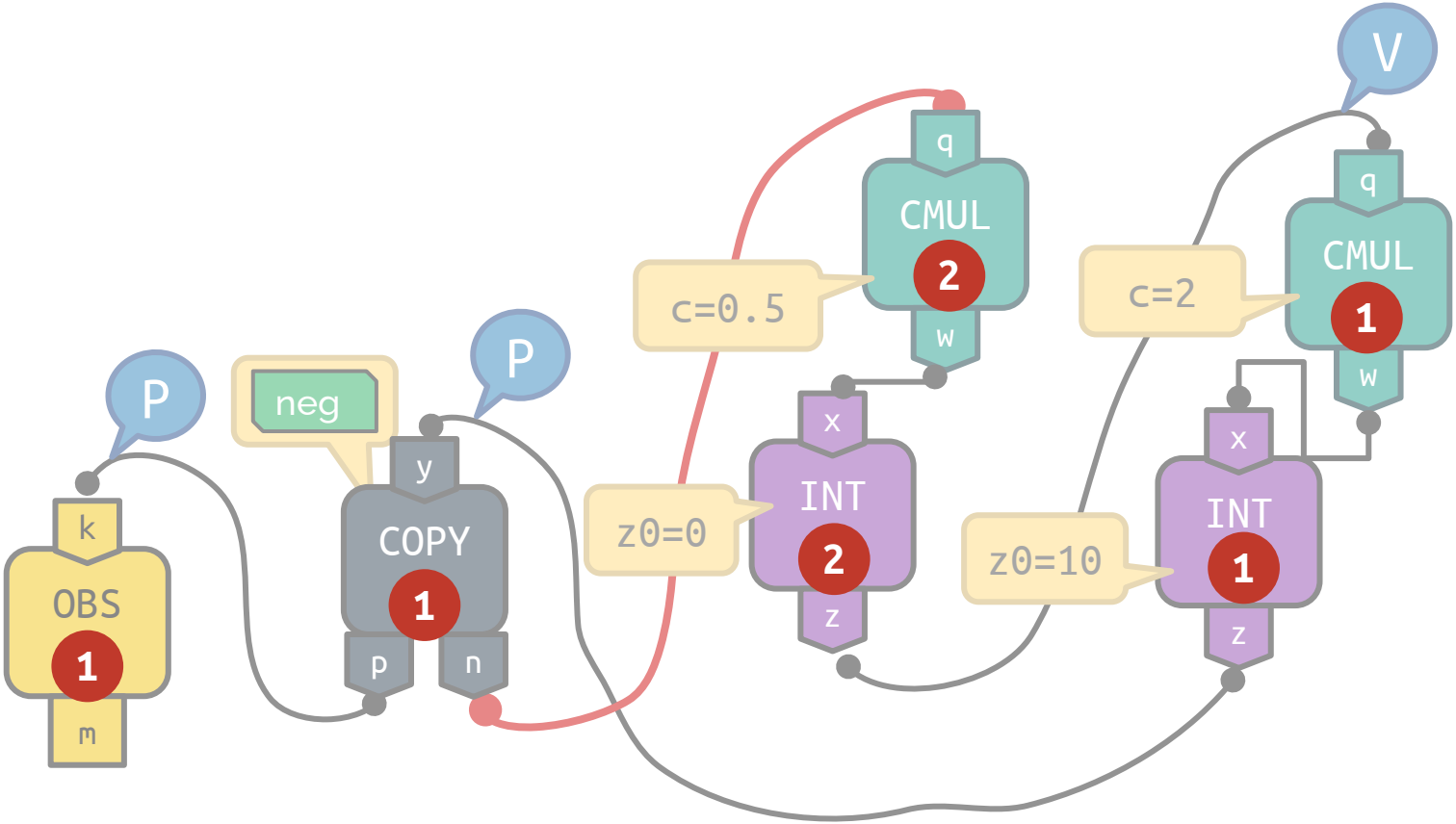
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



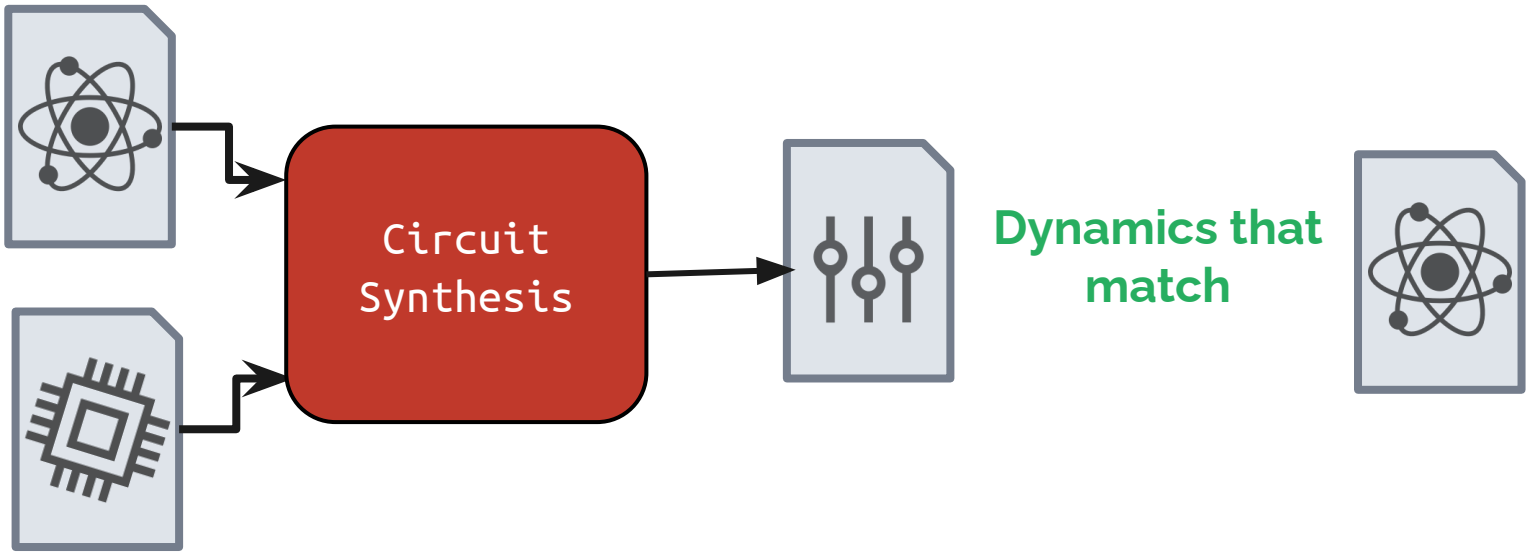
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route



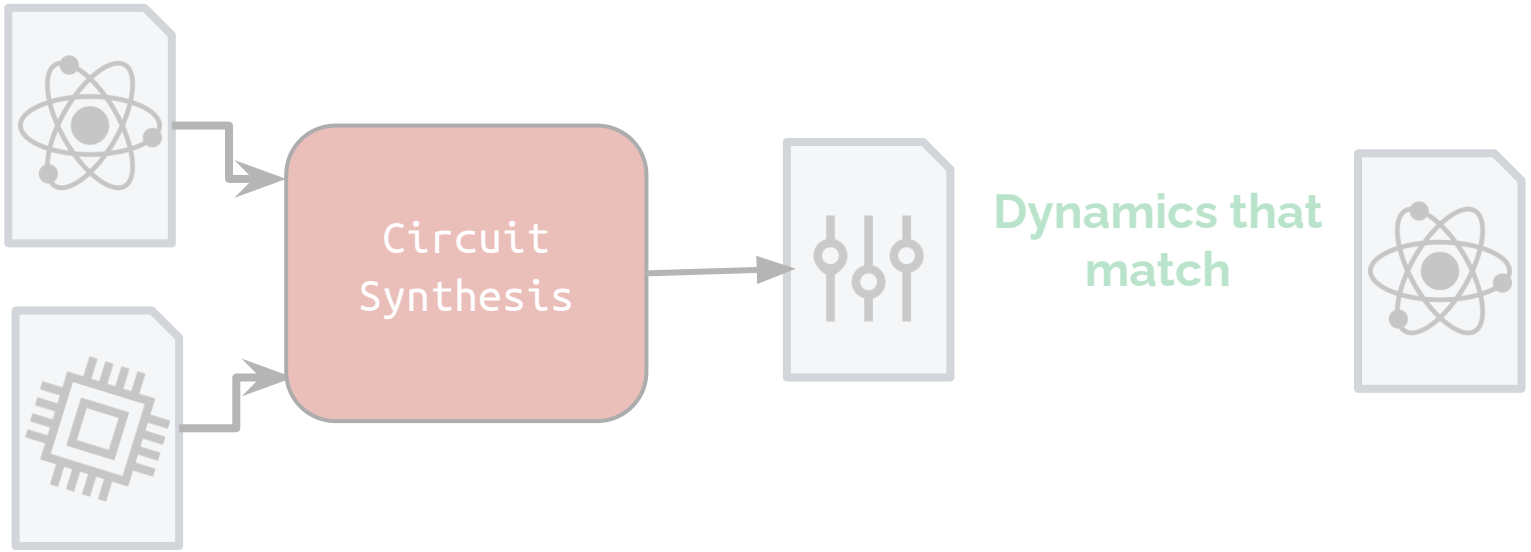
Circuit Synthesis

Subcircuit Synthesis → Assembly → Place and Route

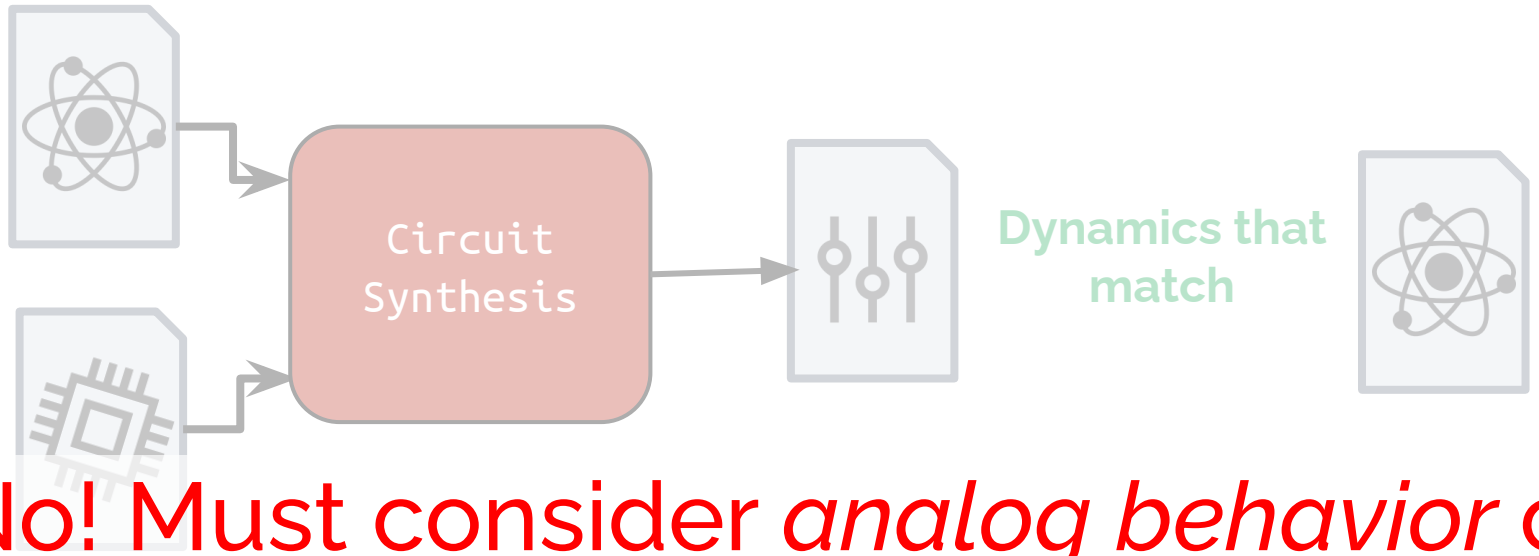


1. Configuration Synthesis for Programmable Analog Devices with Arco. PLDI 2016.
2. Noise-Aware Dynamical System Compilation for Analog Devices with Legno. ASPLOS 2020.

Are we done?



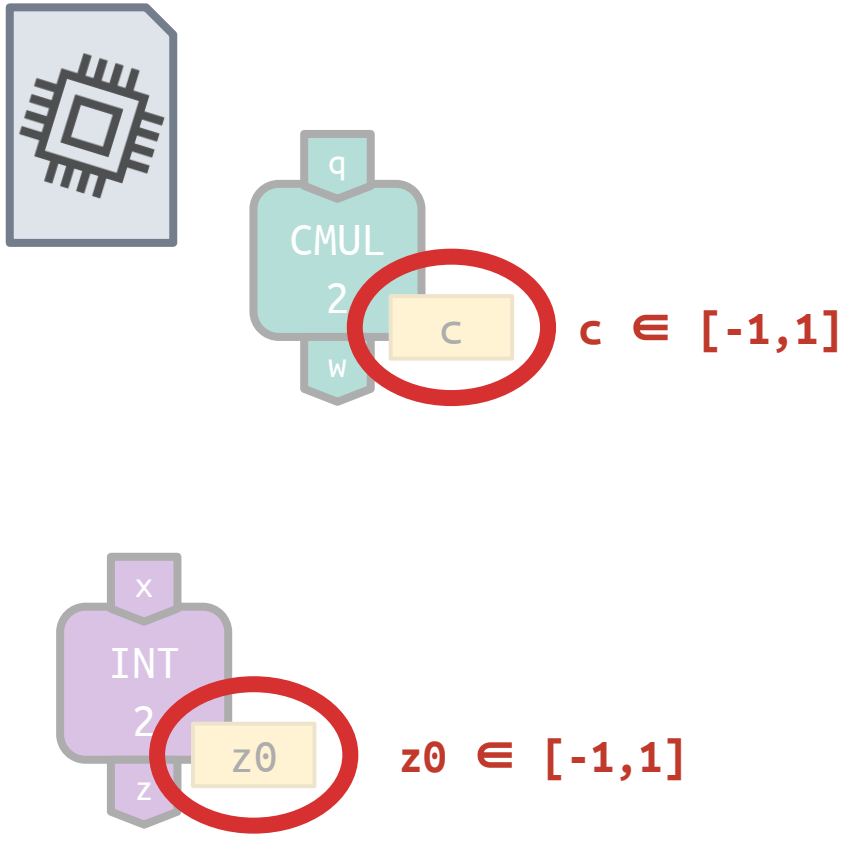
Are we done?



No! Must consider *analog behavior* of blocks

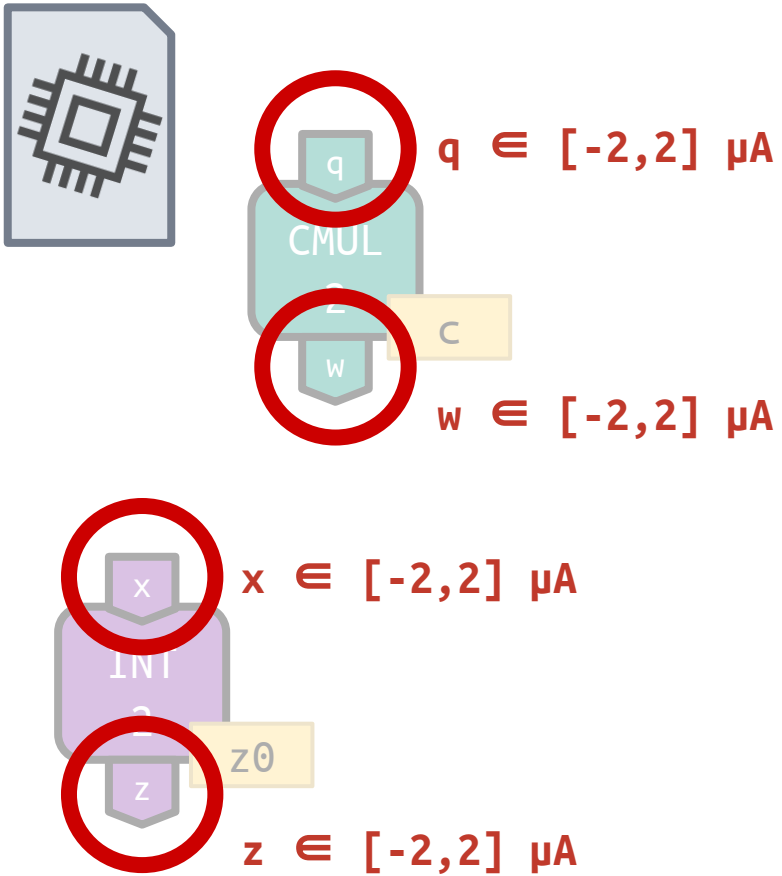
Physical Limitations

Operating range Limitations



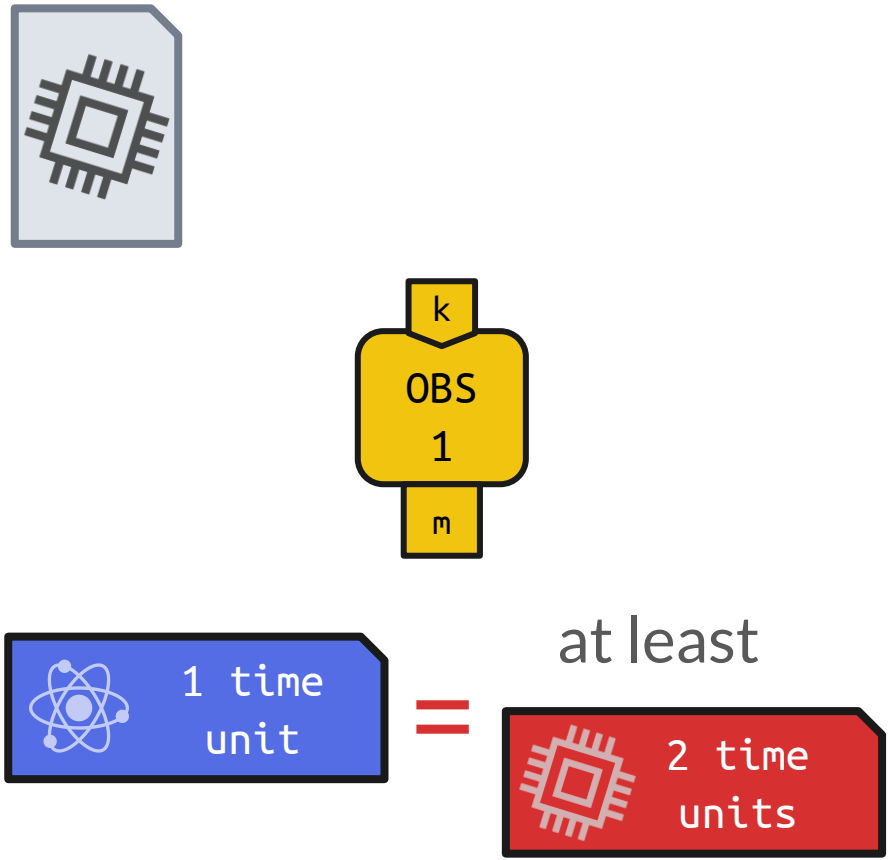
Physical Limitations

Operating range Limitations



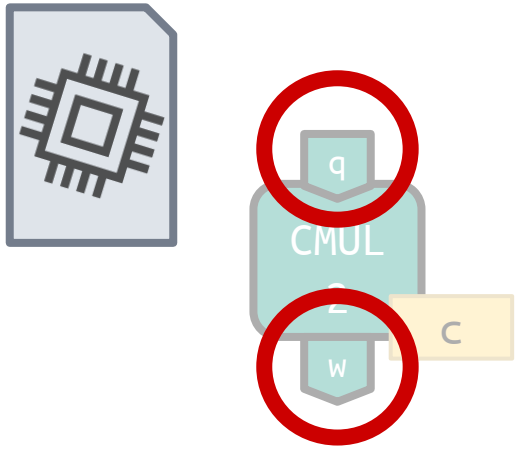
Inputs

Execution Speed Limitations

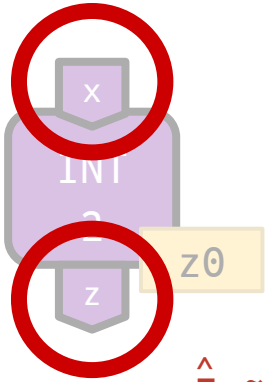


Physical Limitations

Analog Noise



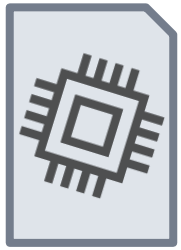
$\hat{w} \sim \text{Normal}(\bar{w}, 0.02\ \mu A)$

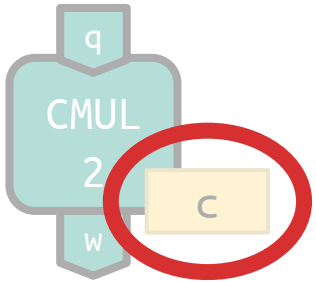


$\hat{z} \sim \text{Normal}(\bar{z}, 0.01\ \mu A)$

Physical Limitations

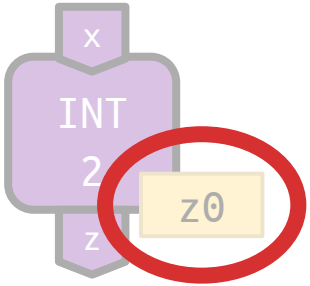
Quantization Error





$c \in [-1,-0.99...,0.99,1]$

$error = 0.01$

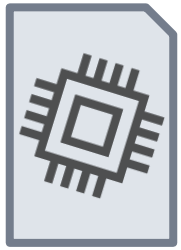


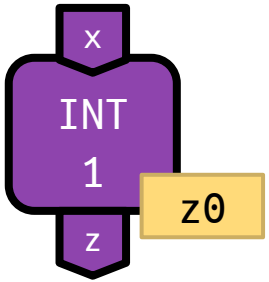
$z0 \in [-1,-0.99...,0.99,1]$

$error = 0.01$

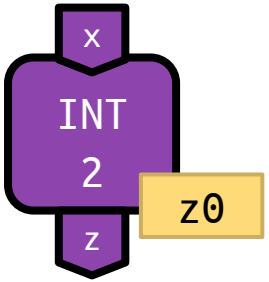
Inputs

Manufacturing Variations





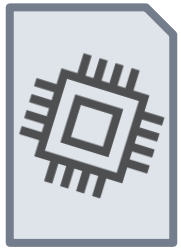
$$z = \int 0.64 * 0.5 * x \, dt_{hw}$$
$$z(0) = 0.80 * z0$$

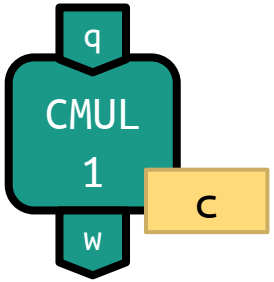


$$z = \int 0.93 * 0.5 * x \, dt_{hw}$$
$$z(0) = 0.77 * z0$$

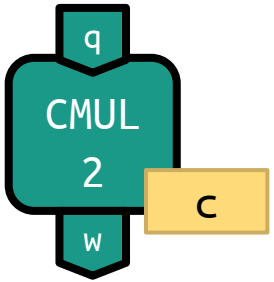
Inputs

Manufacturing Variations

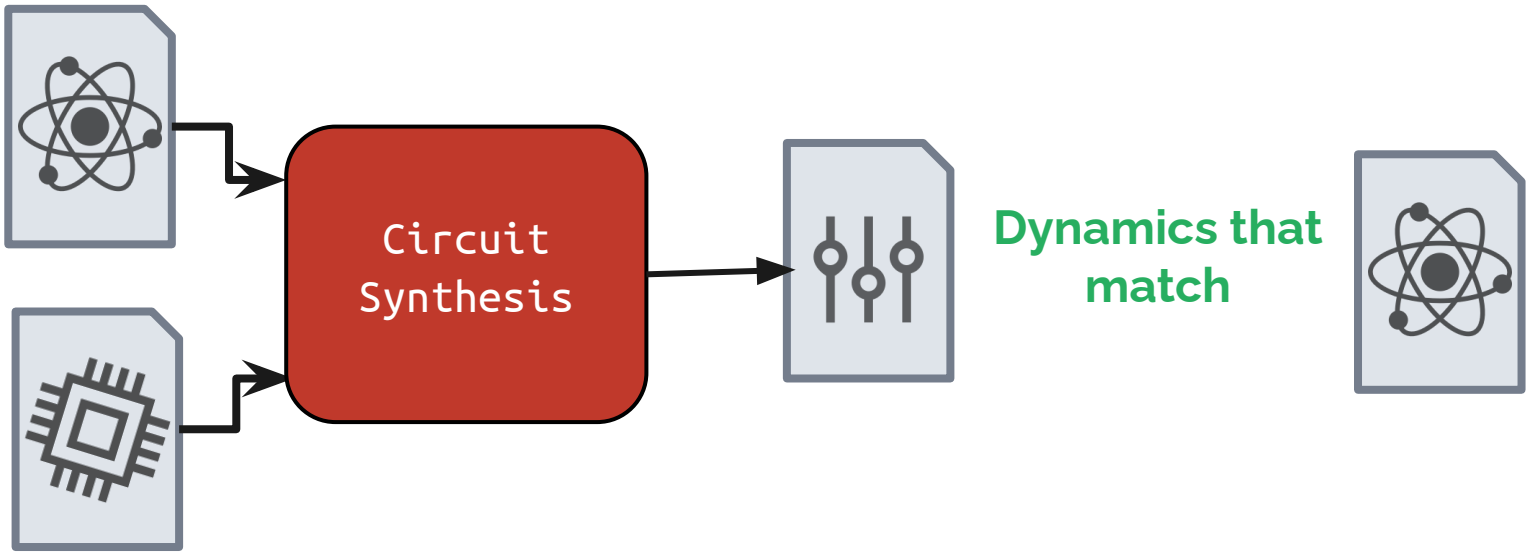


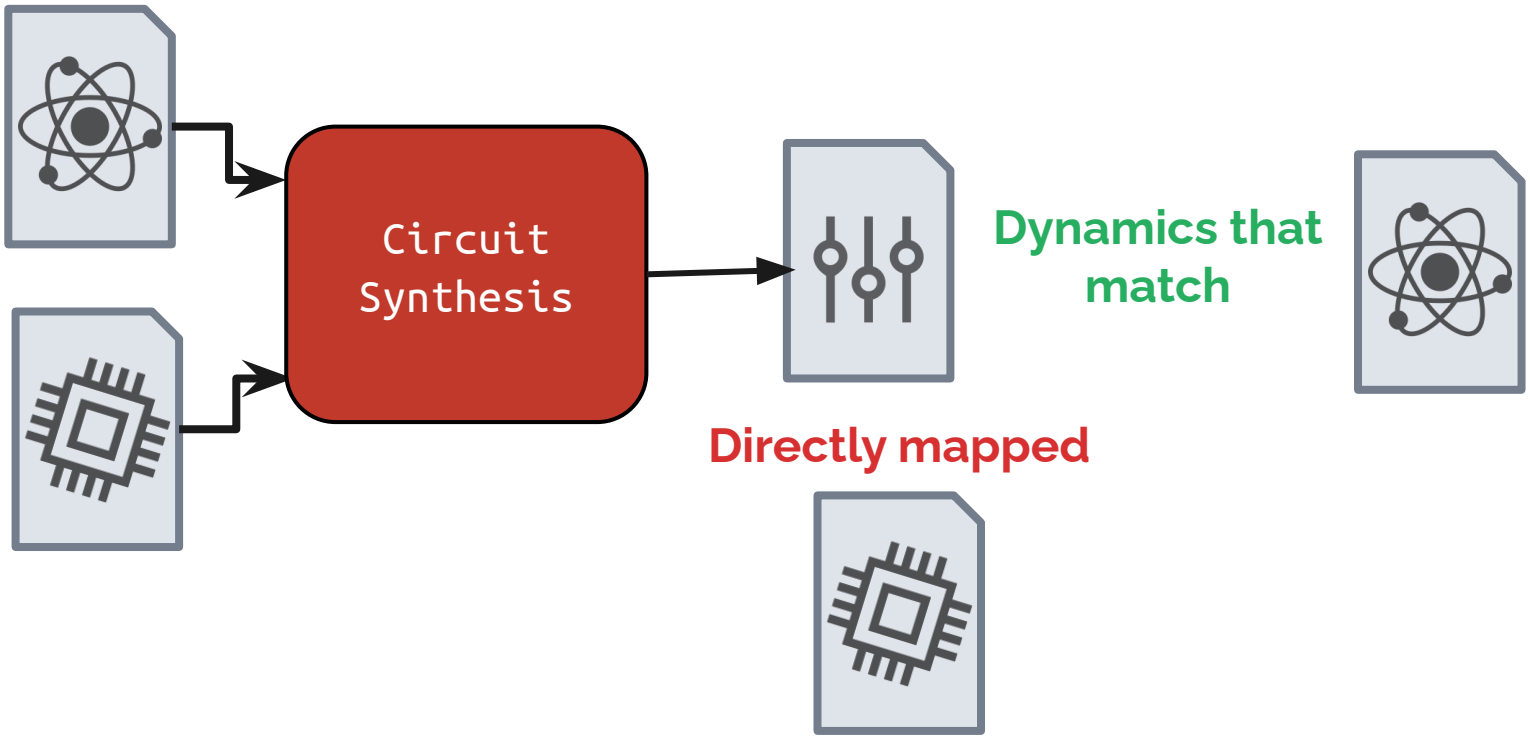


w =0.76*c*q



w =1.12*c*q

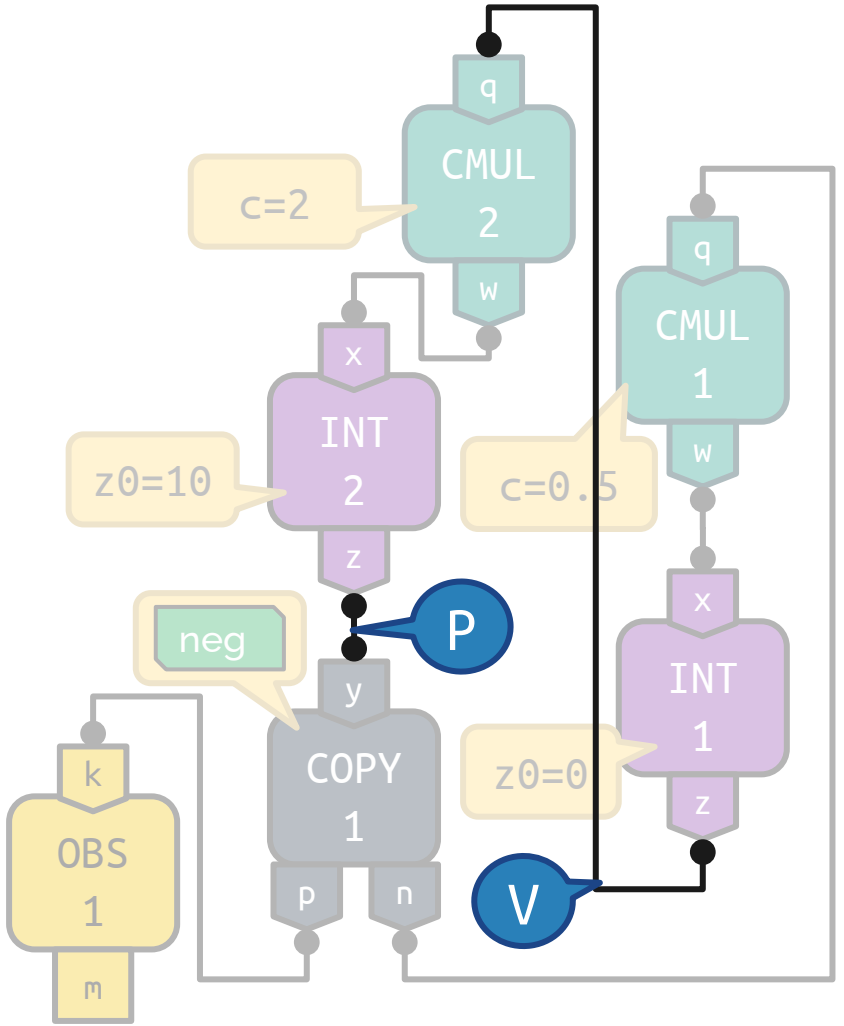

Circuit
Synthesis
Dynamics that
match


Circuit
Synthesis
Dynamics that
match
Directly mapped

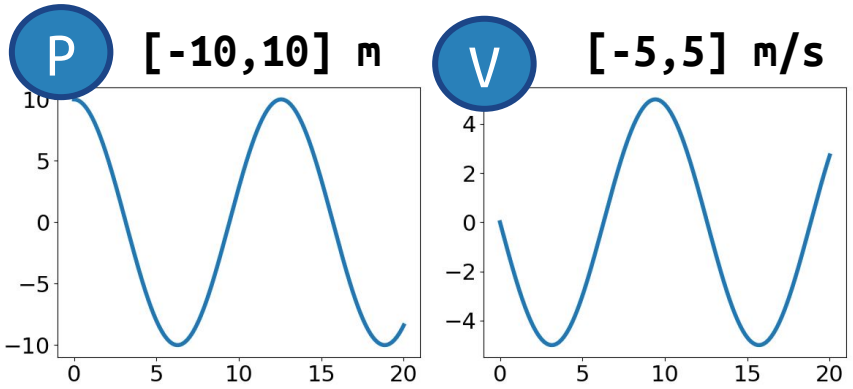
Directly Mapped Configuration

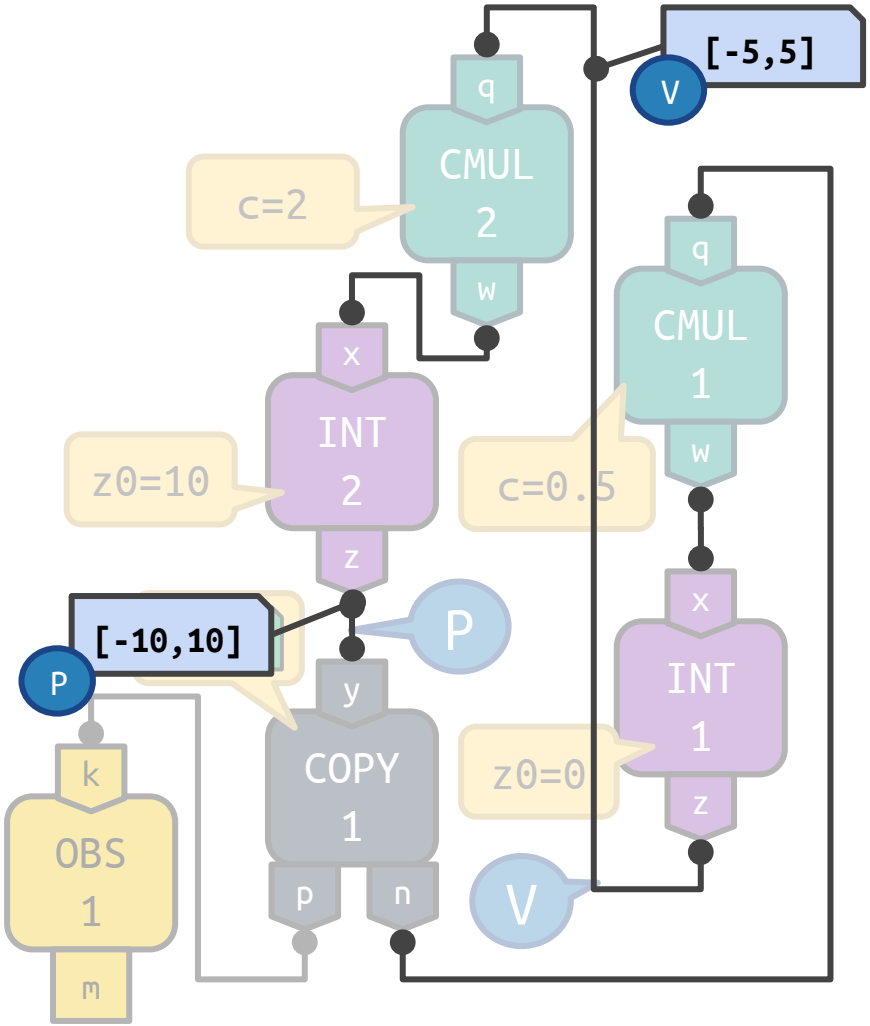
P `1 µA = 1 meter`

V `1 µA = 1 meter/second`

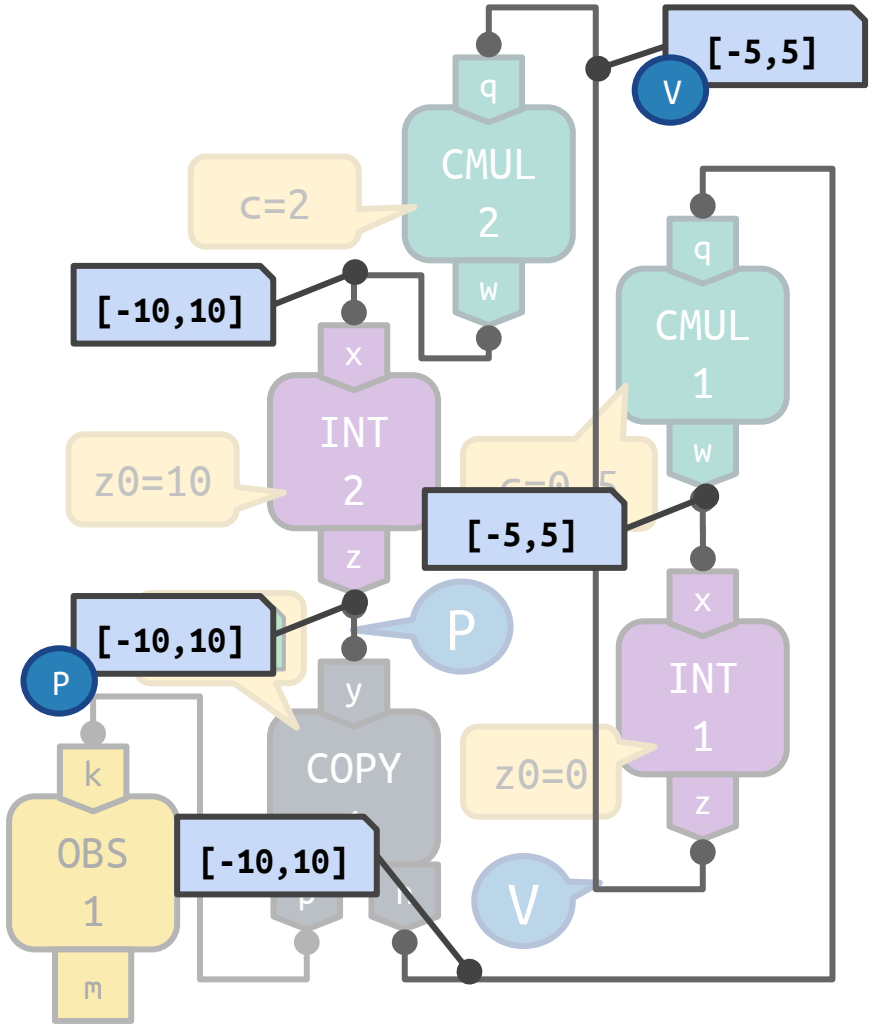


Directly Mapped Configuration



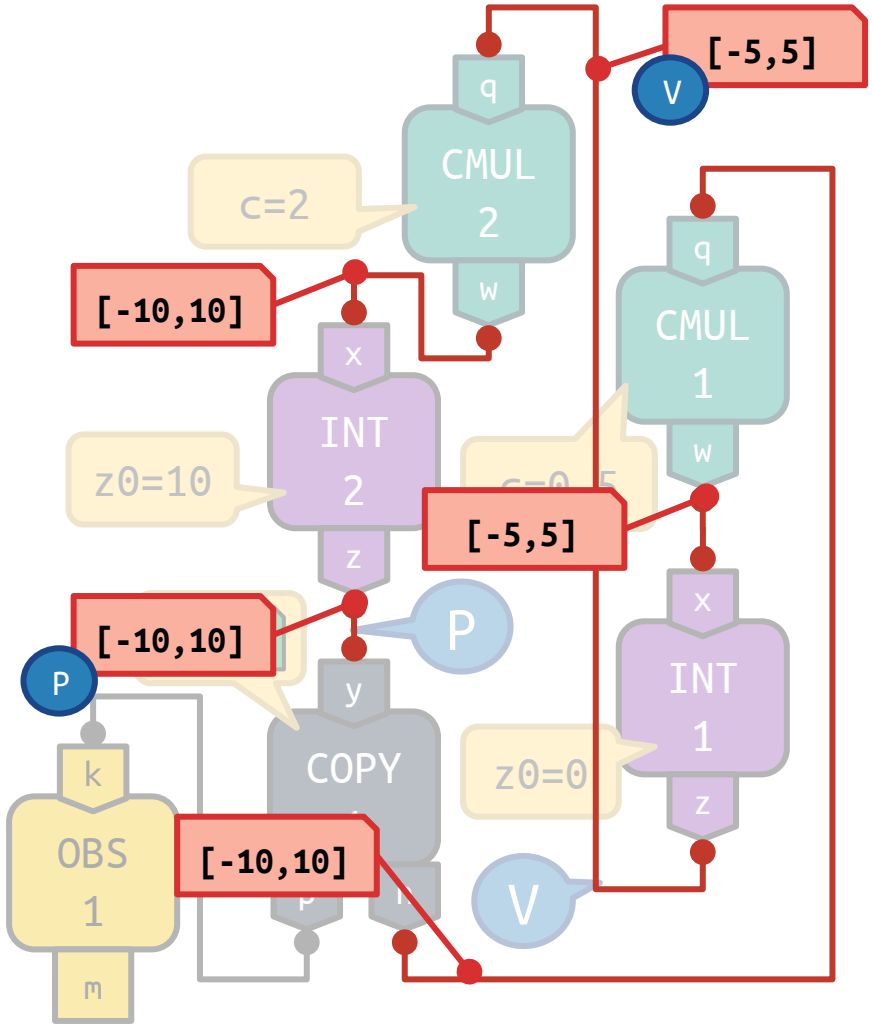


Directly Mapped Configuration



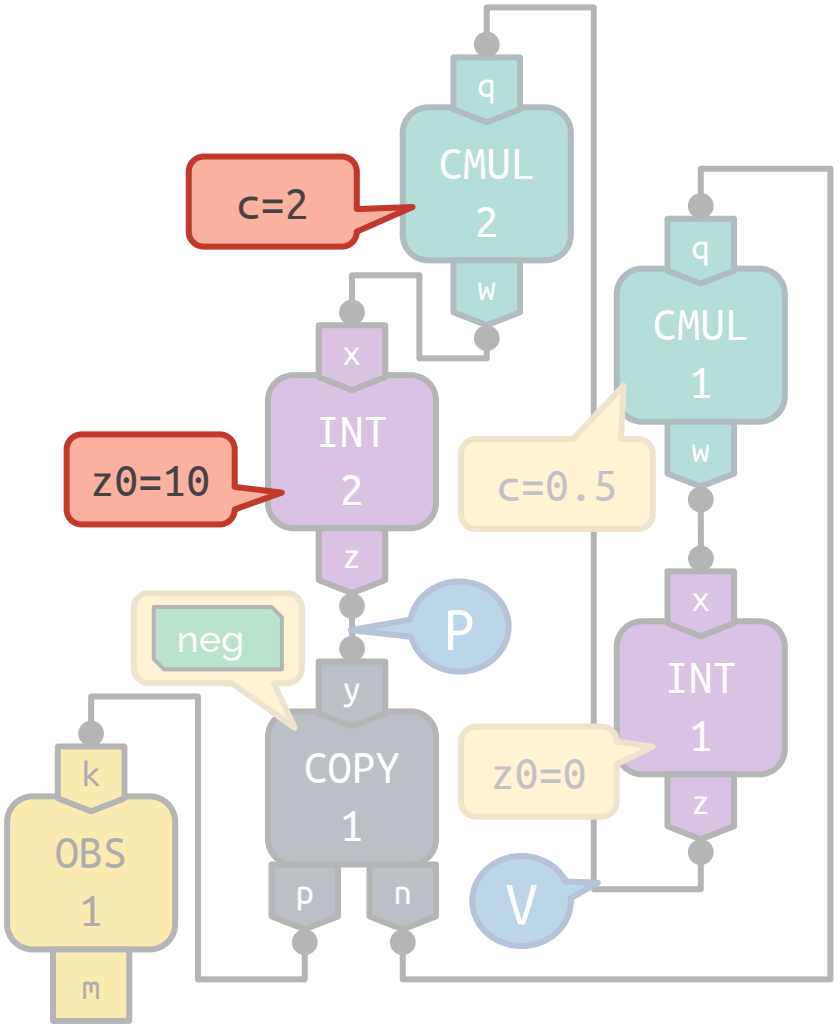
Operating Range Limitation Violations

$x,z,q,w \in [-2,2]\ \mu A$

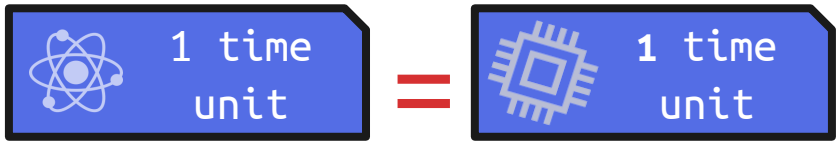


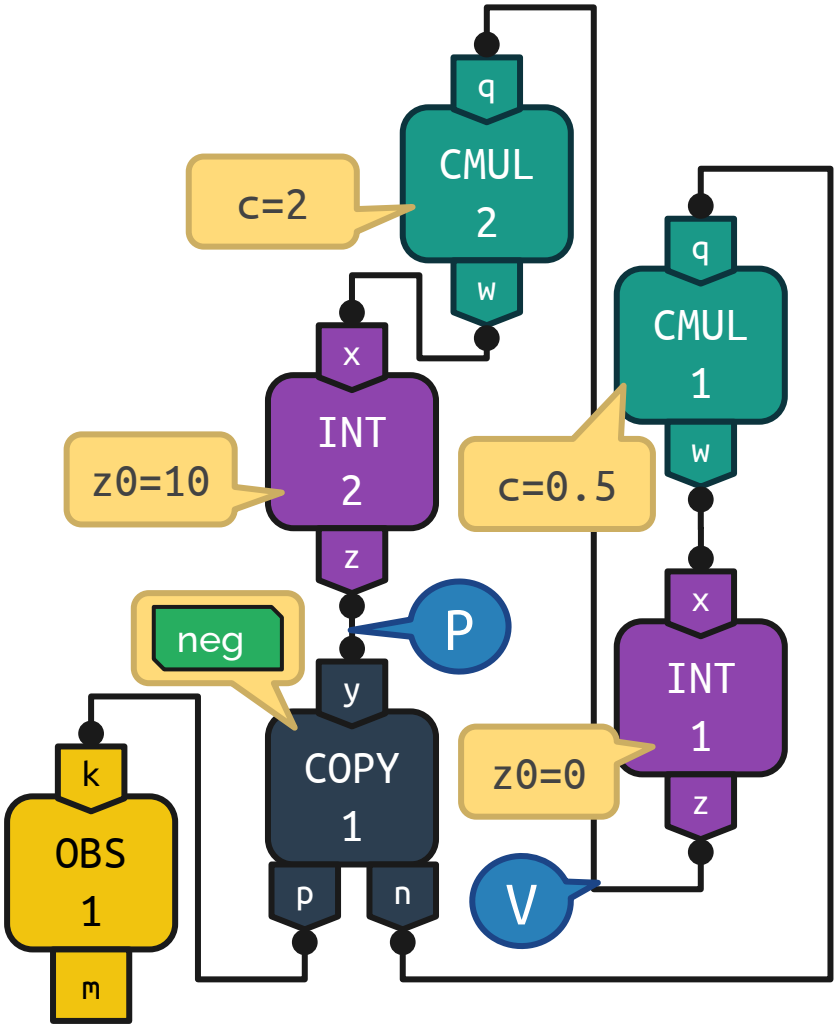
Operating Range Limitation Violations

$$c, z0 \in [-1,1]$$



Directly Mapped Configuration



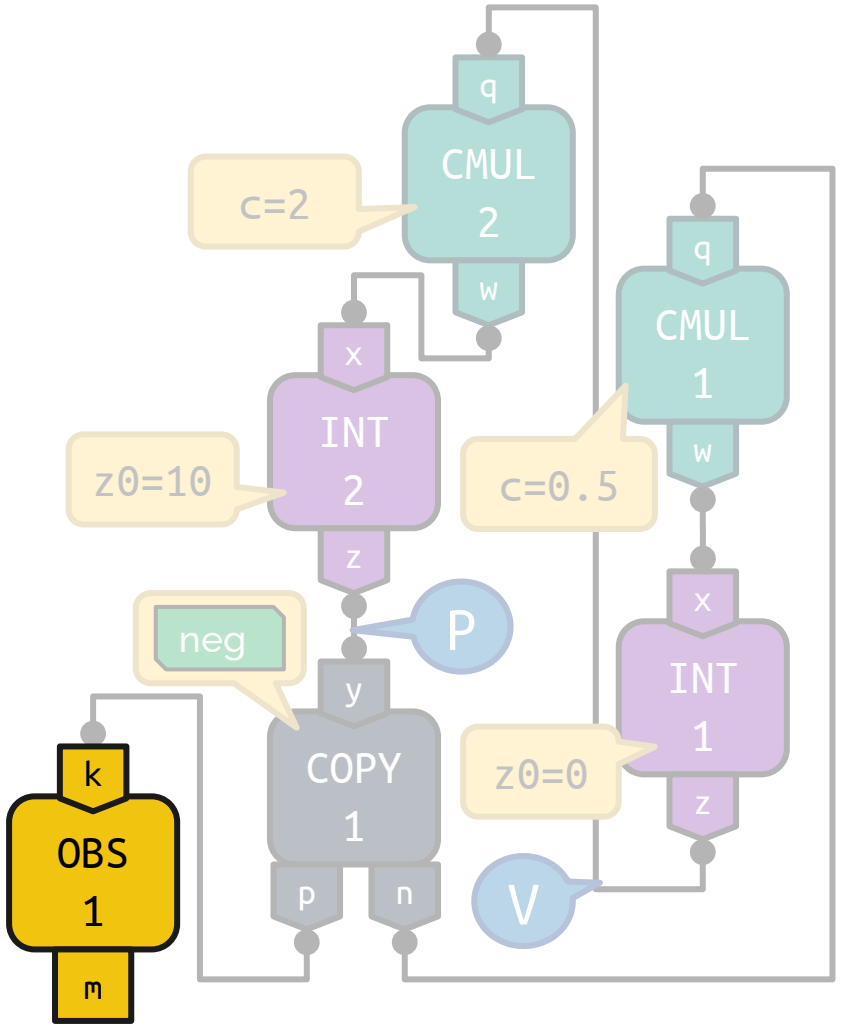


Execution Speed Limitations

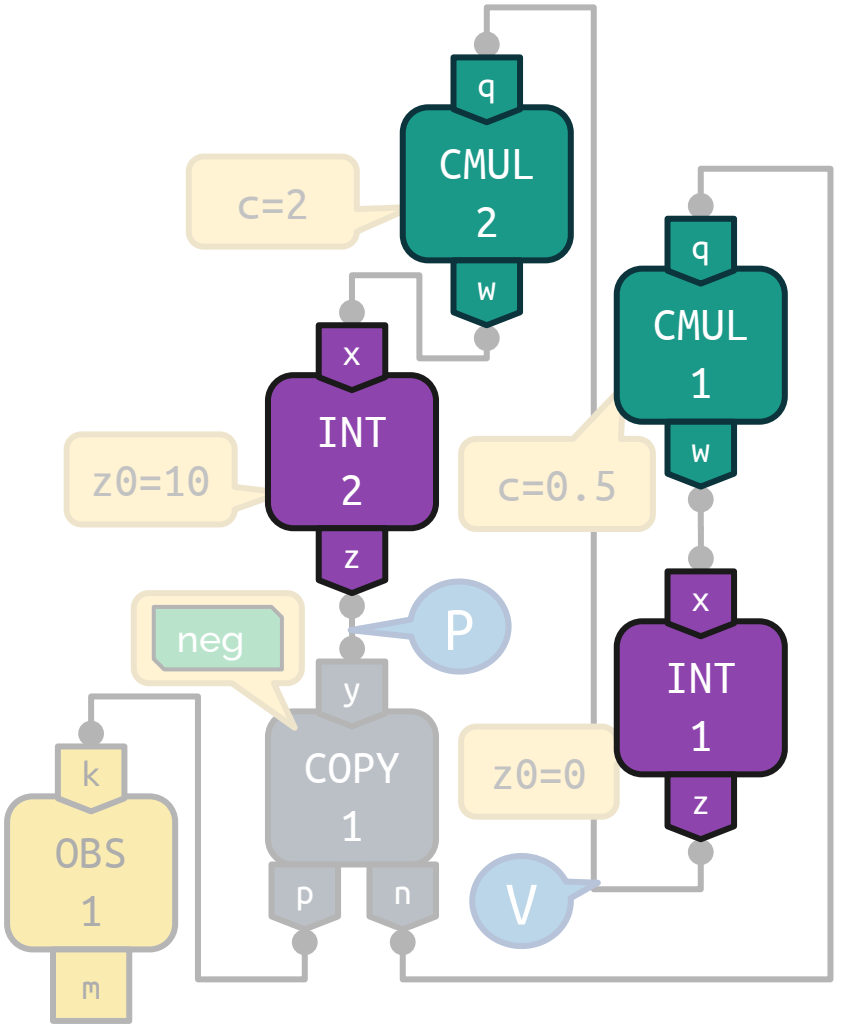
1 time unit = 1 time unit

Must be at at least

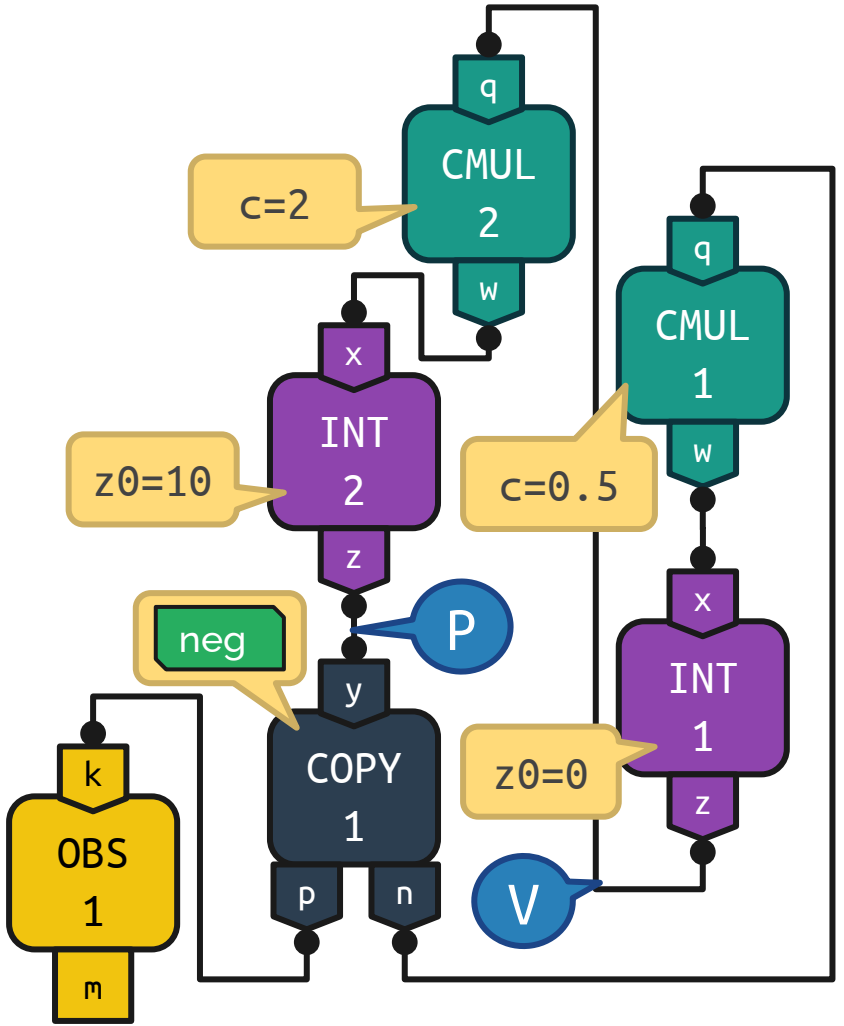
2 time units



Manufacturing Variations



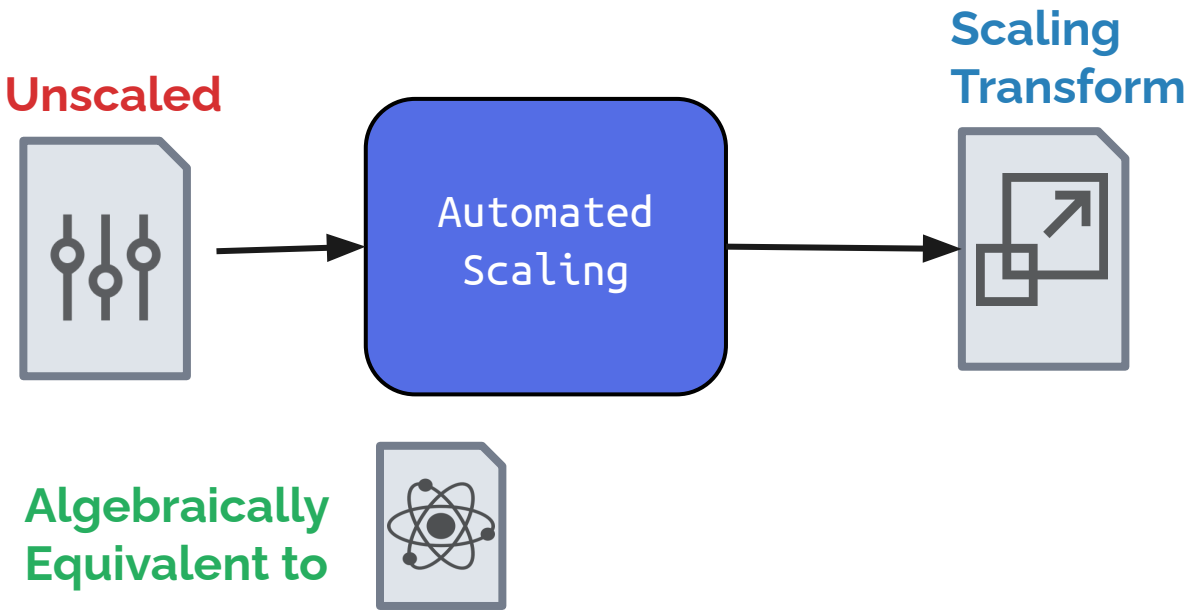
Noise and Quantization Error



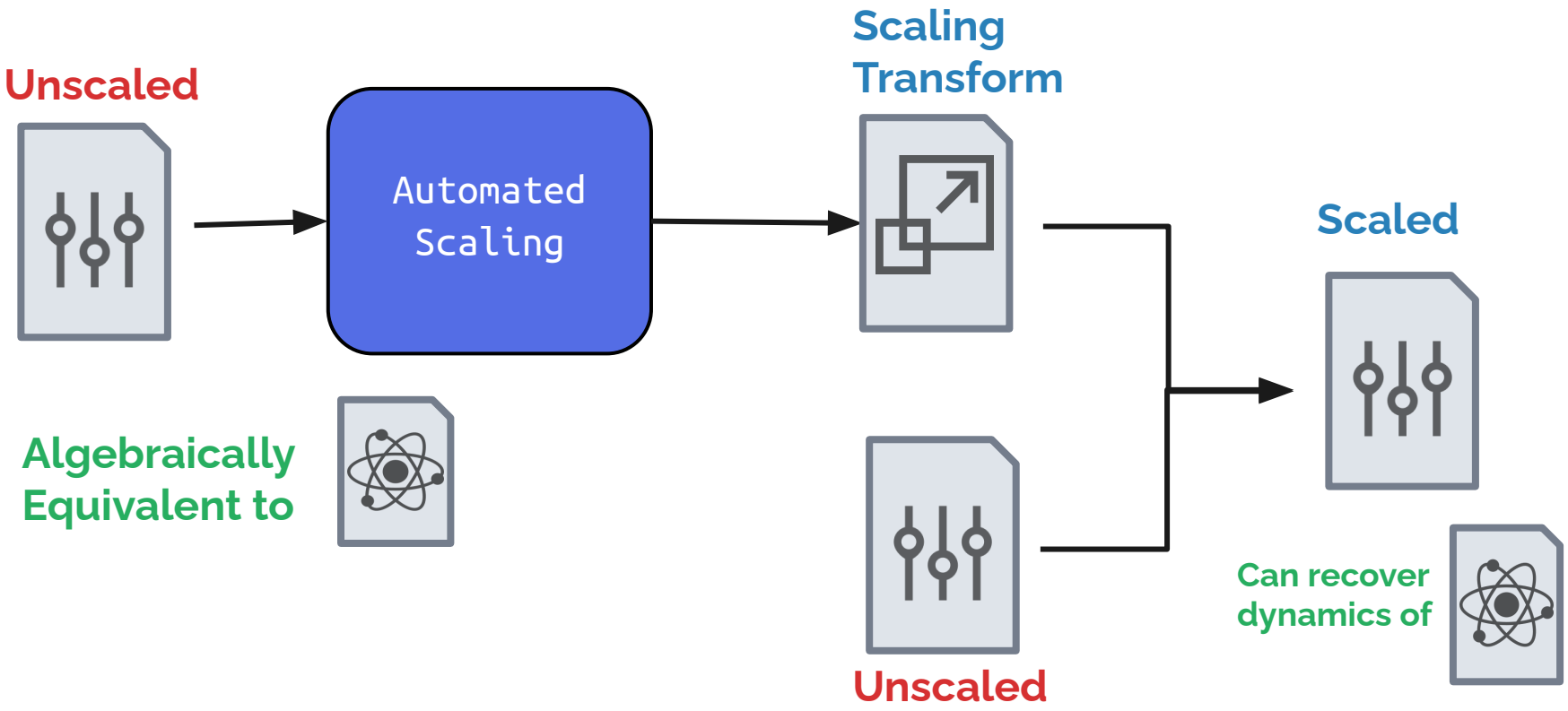
How do we mitigate these behaviors?

How do we mitigate these behaviors?

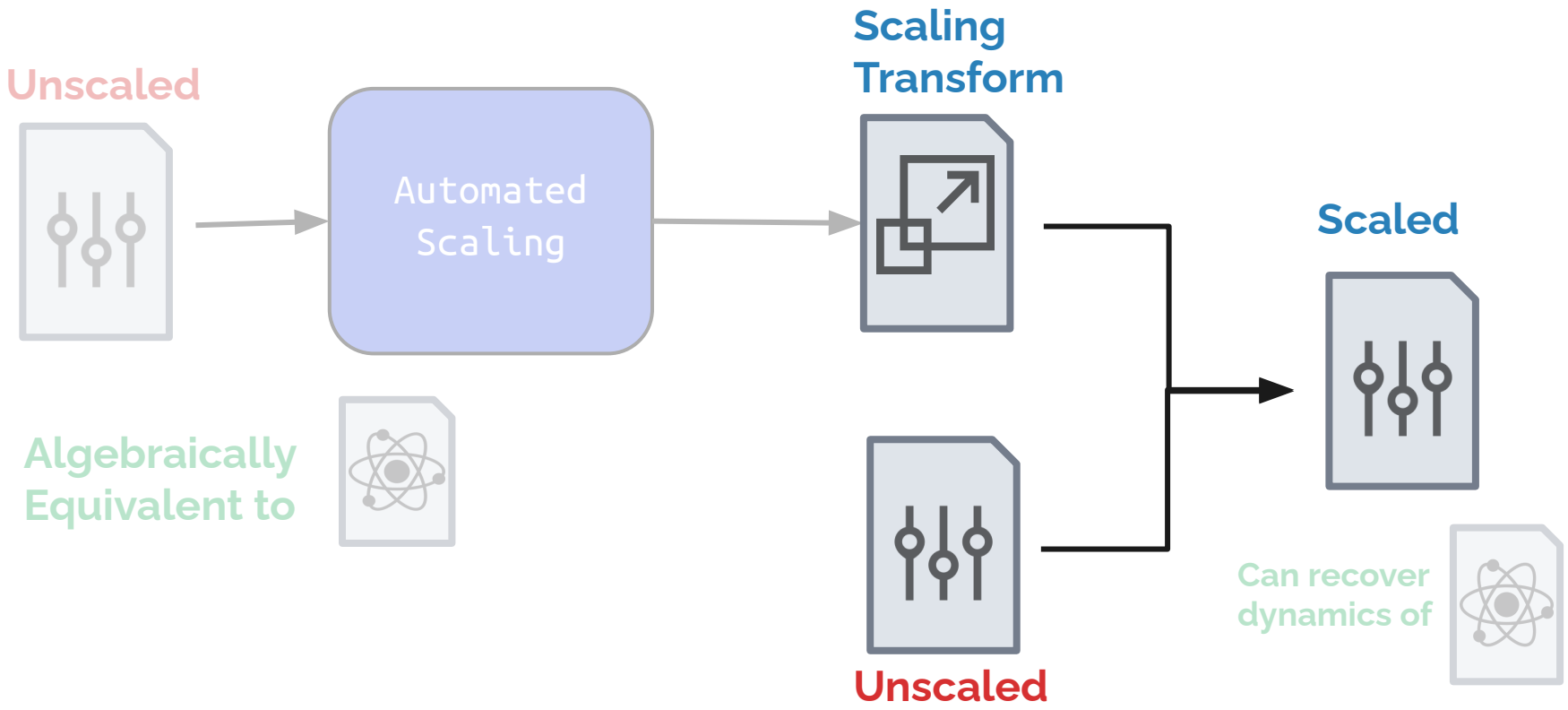
We Scale the Analog Device Configuration



Automated Scaling



Automated Scaling

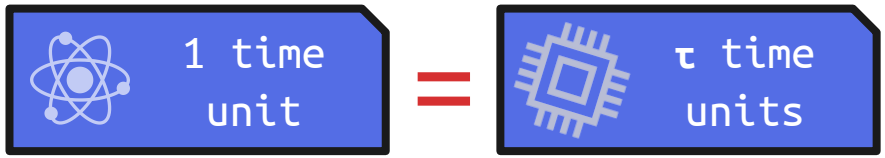


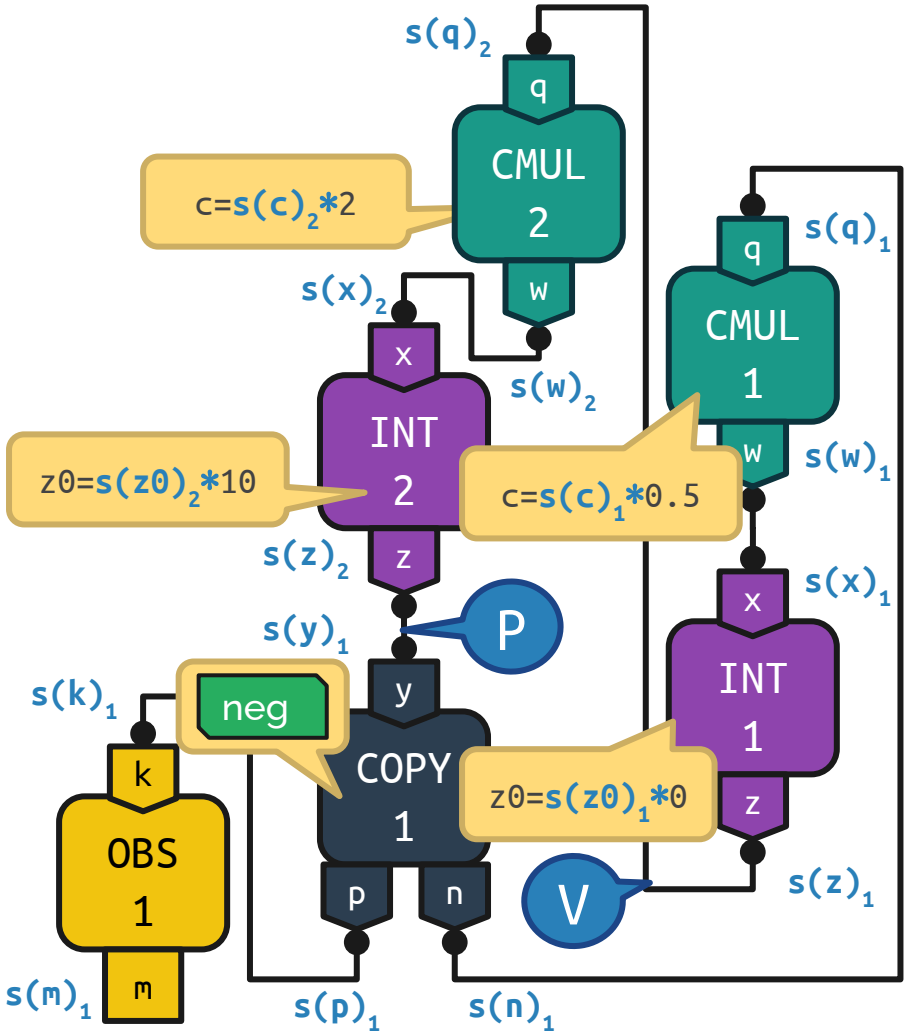
Automated Scaling

The Scaling Transform

$s(z)_2$
$s(c)_1$ Signal scale factors

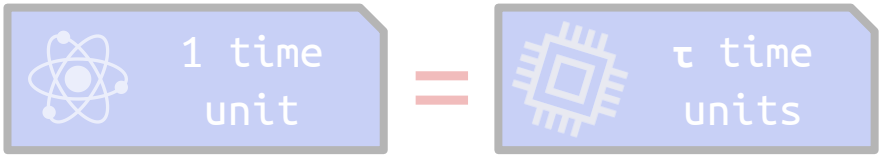
τ Time scale factor

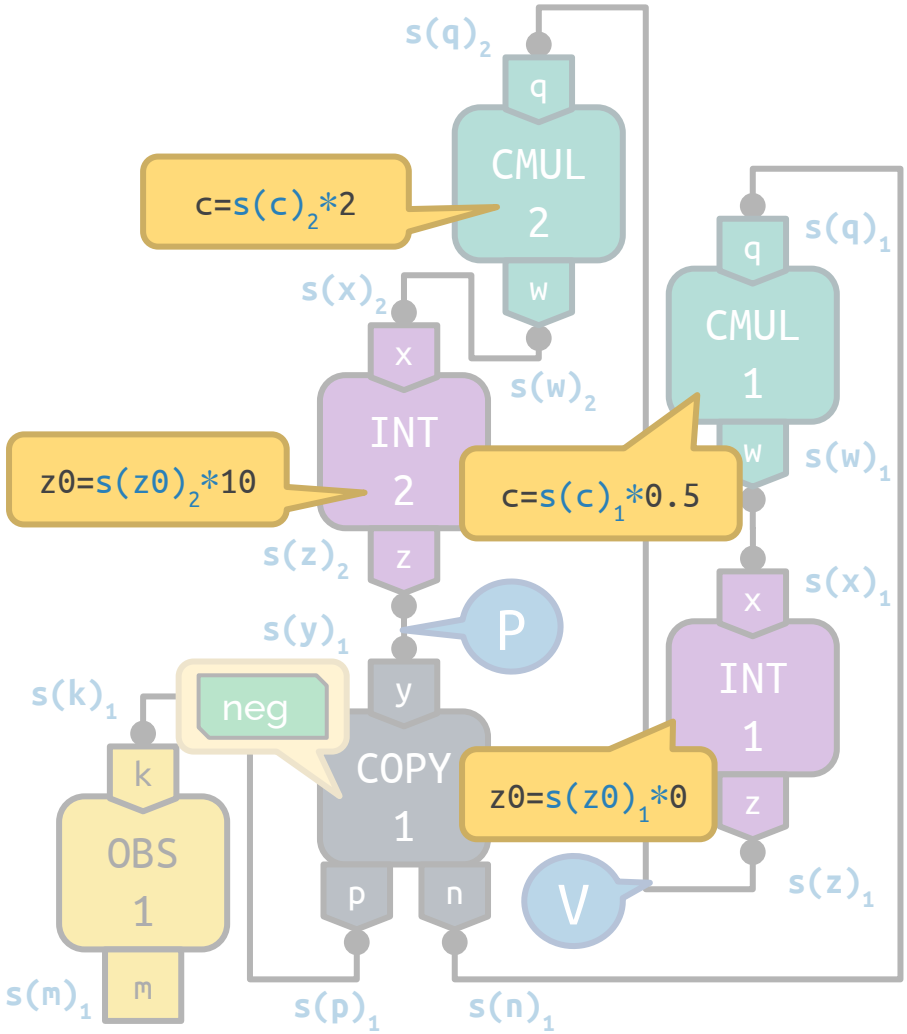




Applying the Scaling Transform

Multiply each digital value with its signal scale factor

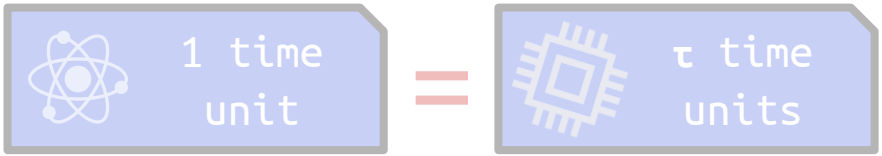


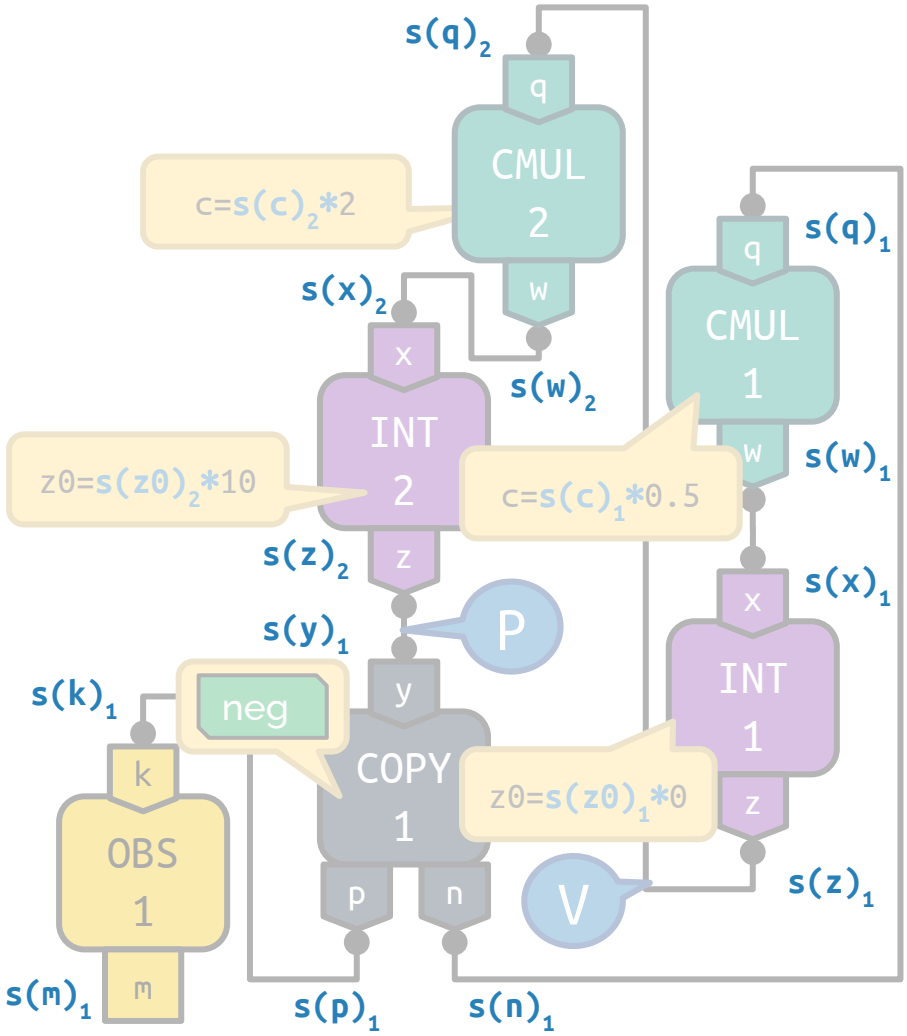


Applying the Scaling Transform

The scaled signals propagate through the analog device

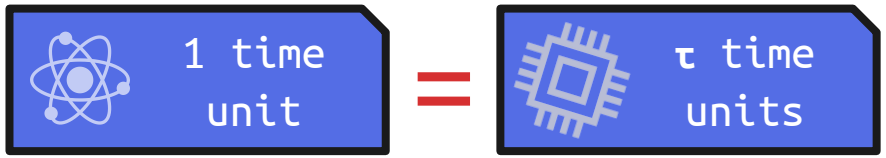
Respects physical restrictions

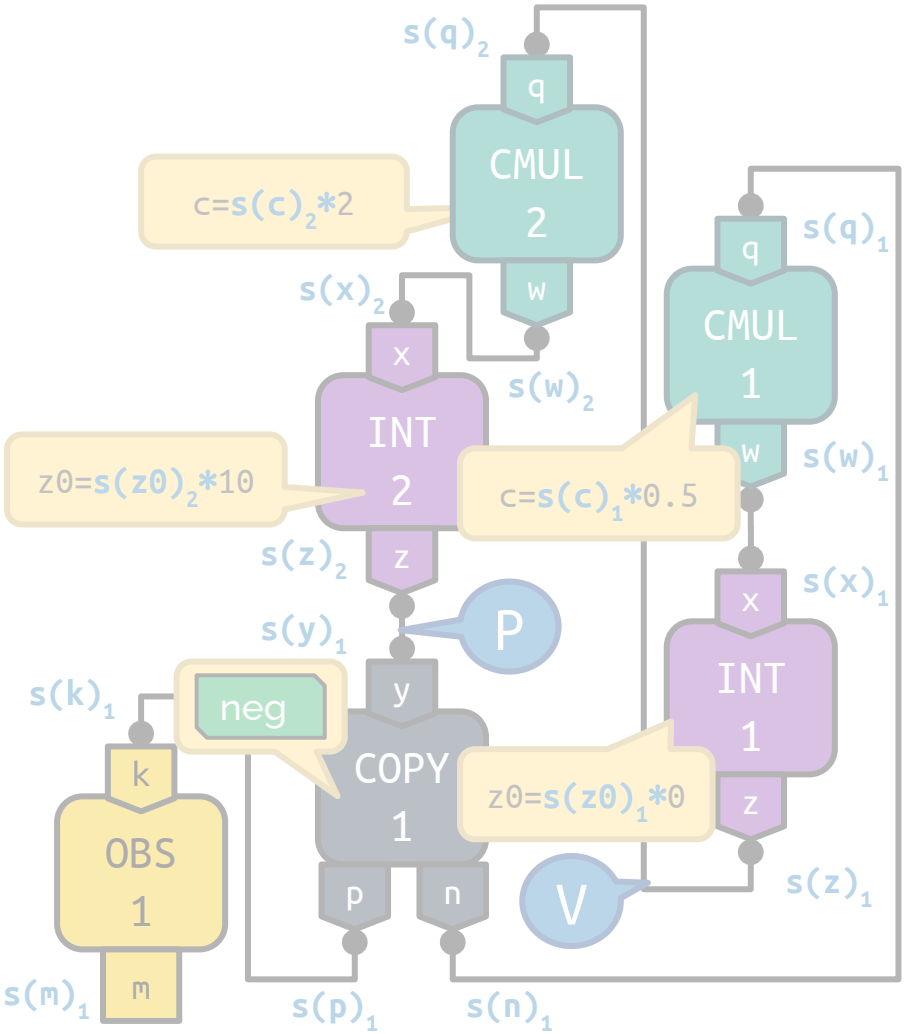


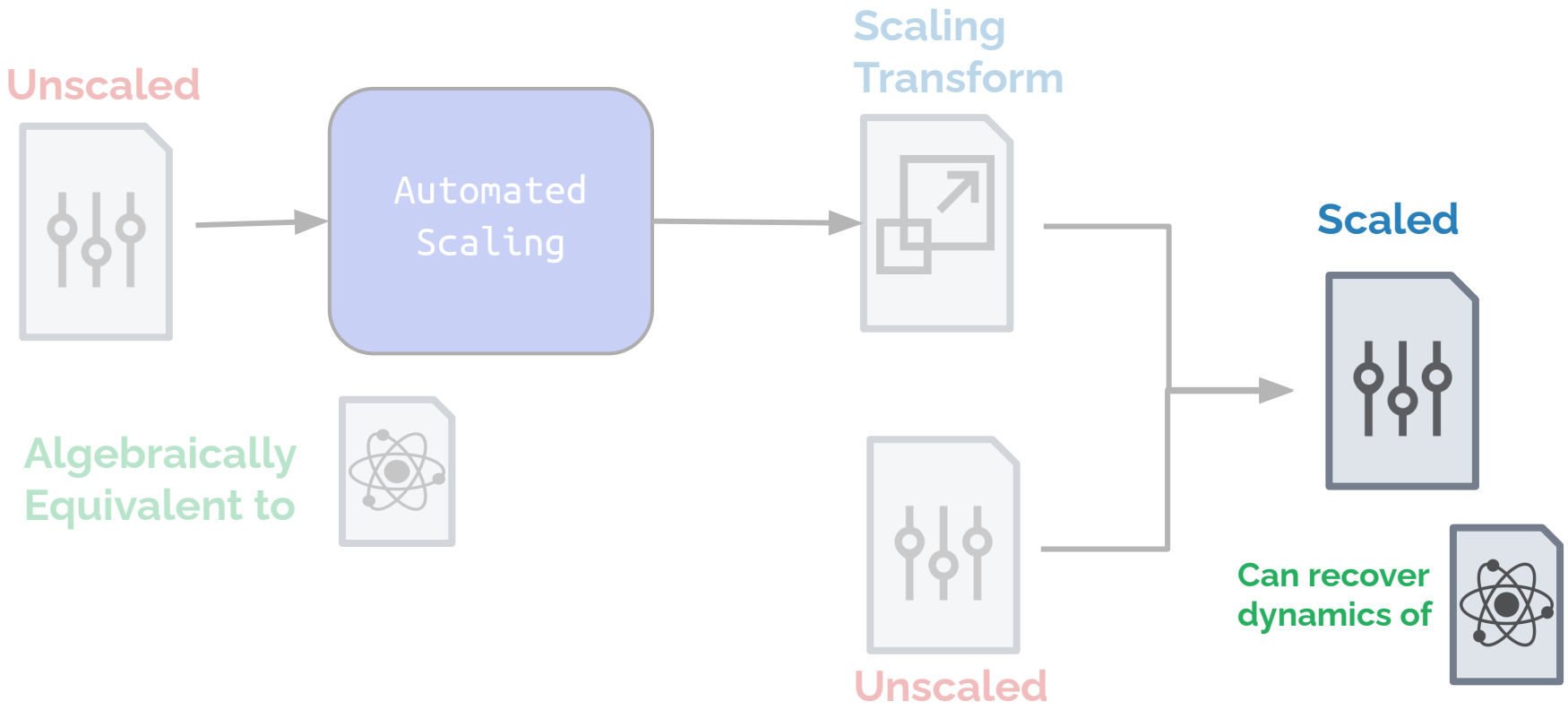


Applying the Scaling Transform

This automatically changes the execution speed of the dynamical system

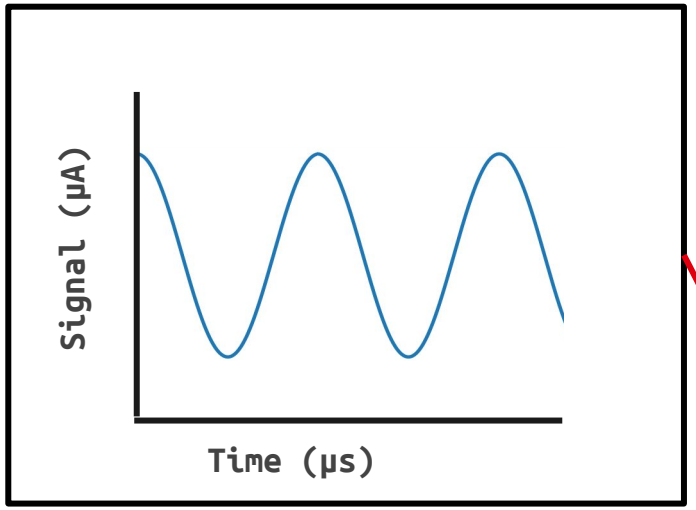


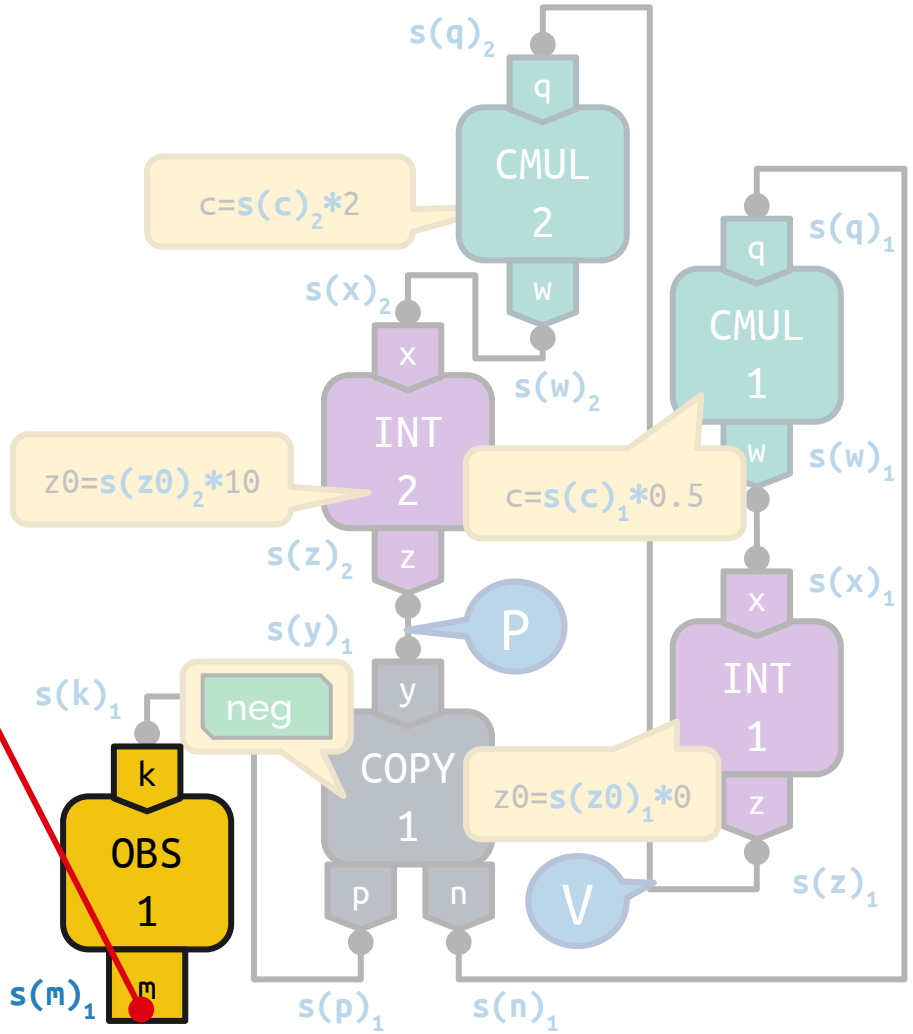




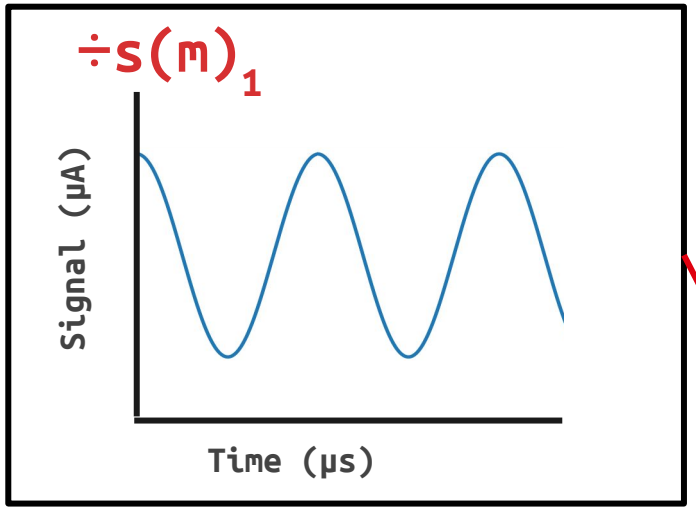
Automated Scaling

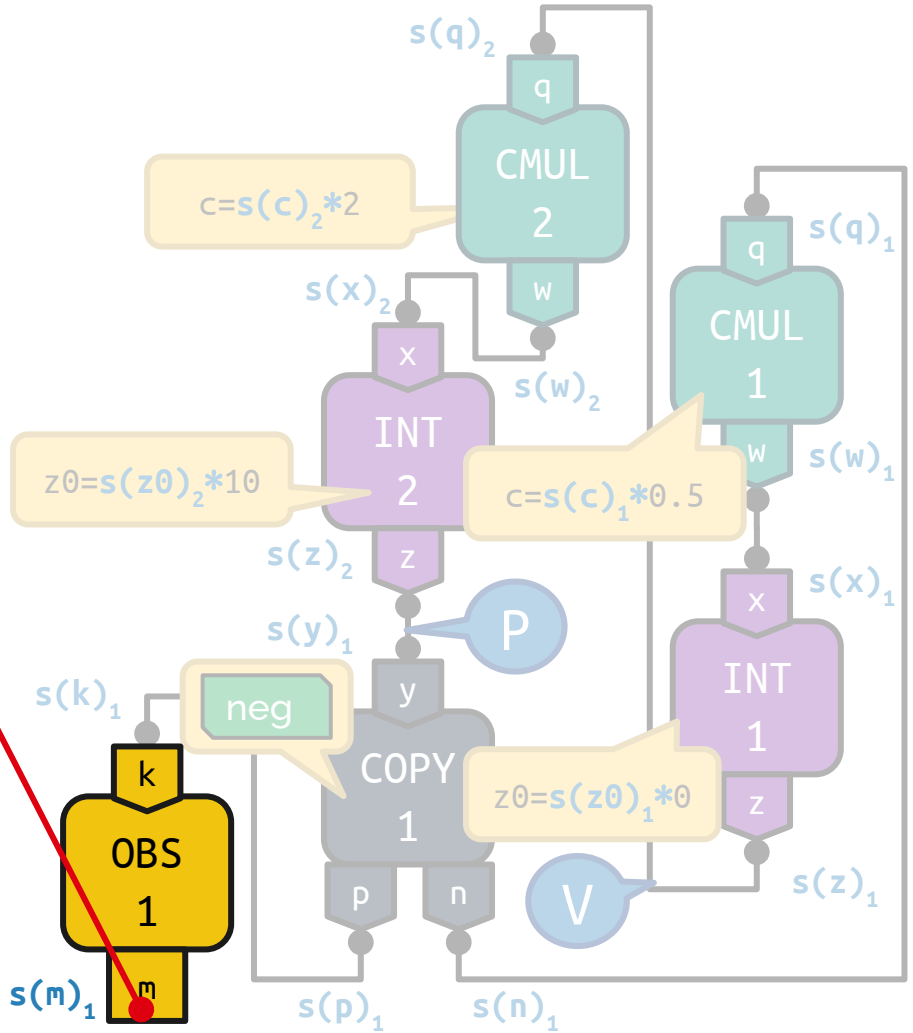
Recovering the Original Dynamics



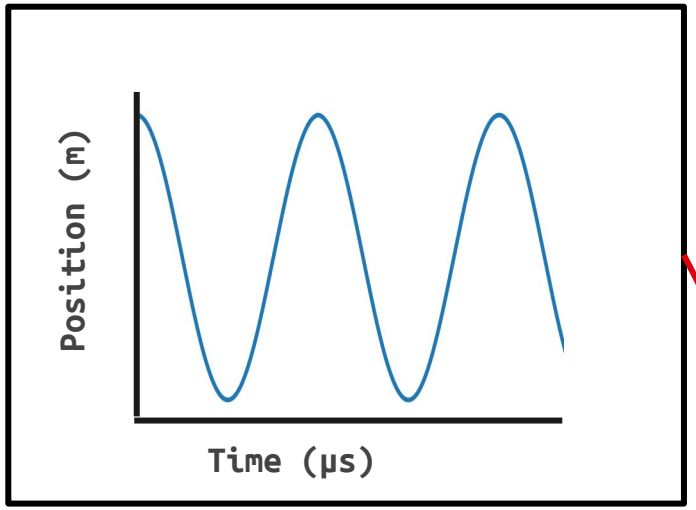


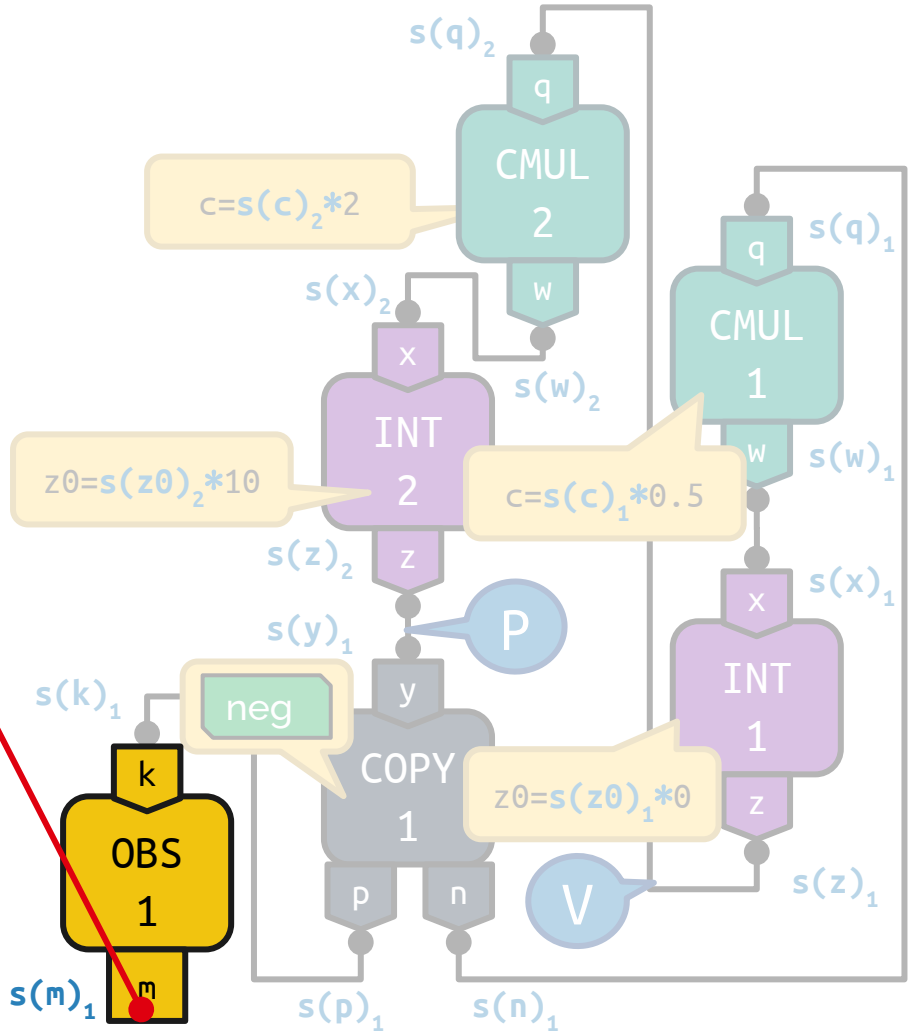
Recovering the Original Dynamics



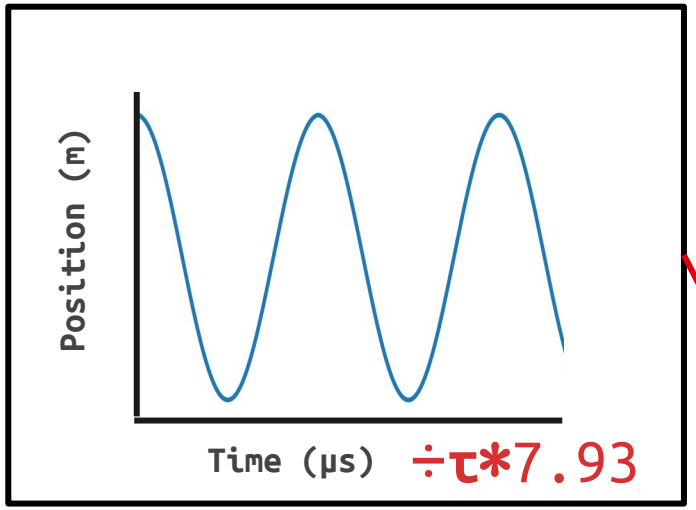


Recovering the Original Dynamics

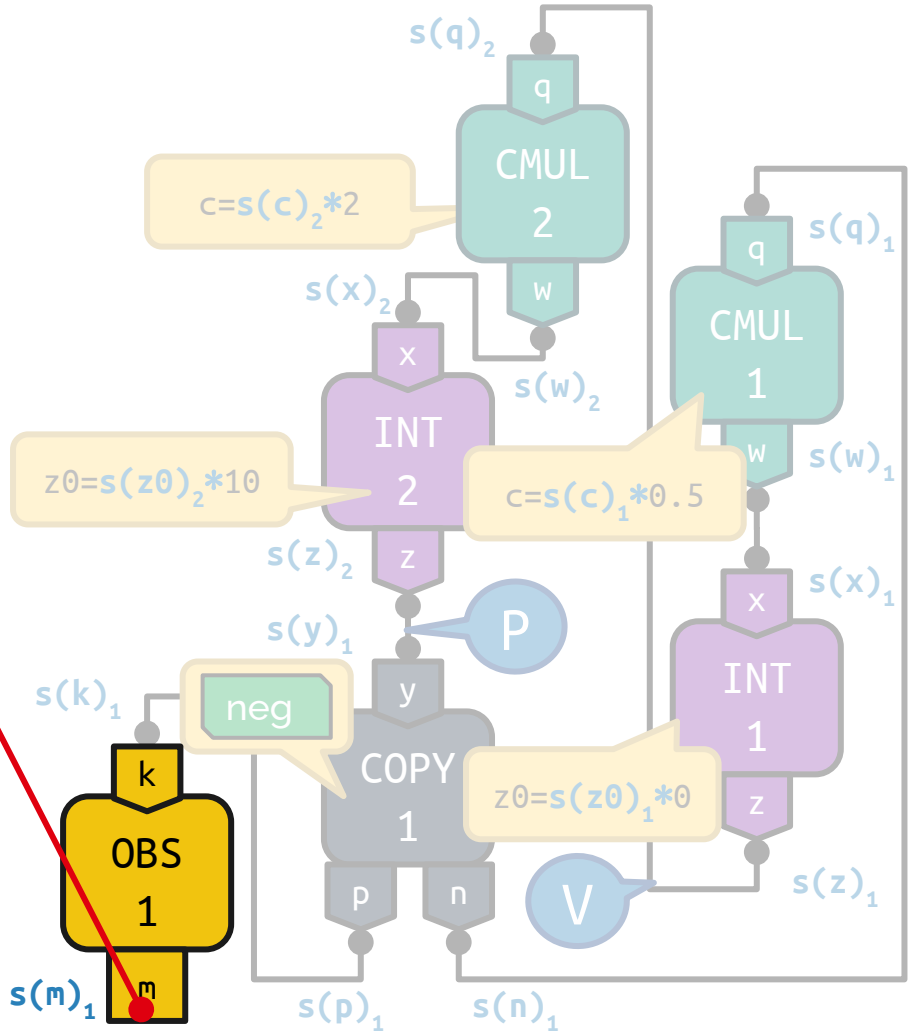




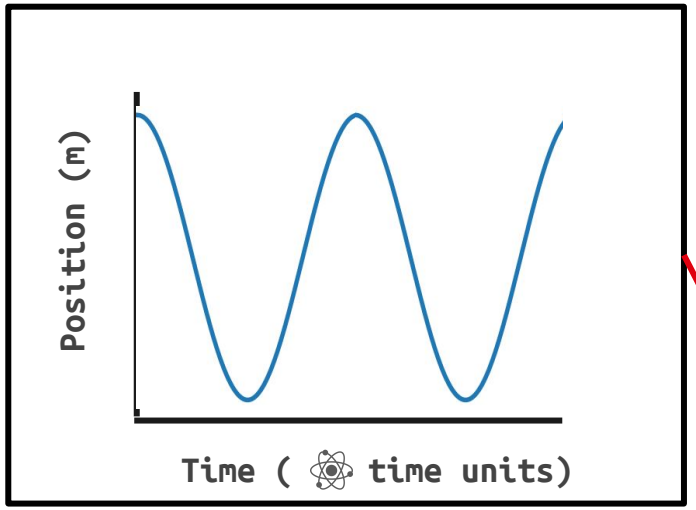
Recovering the Original Dynamics

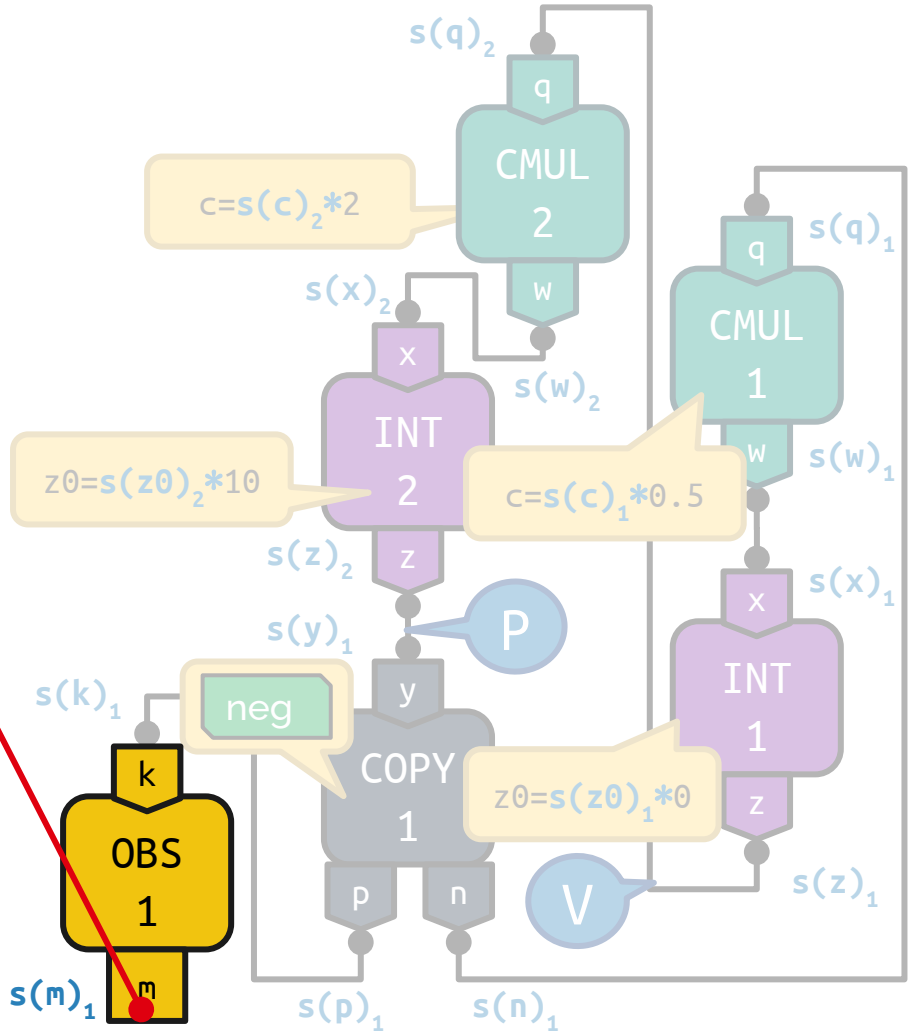


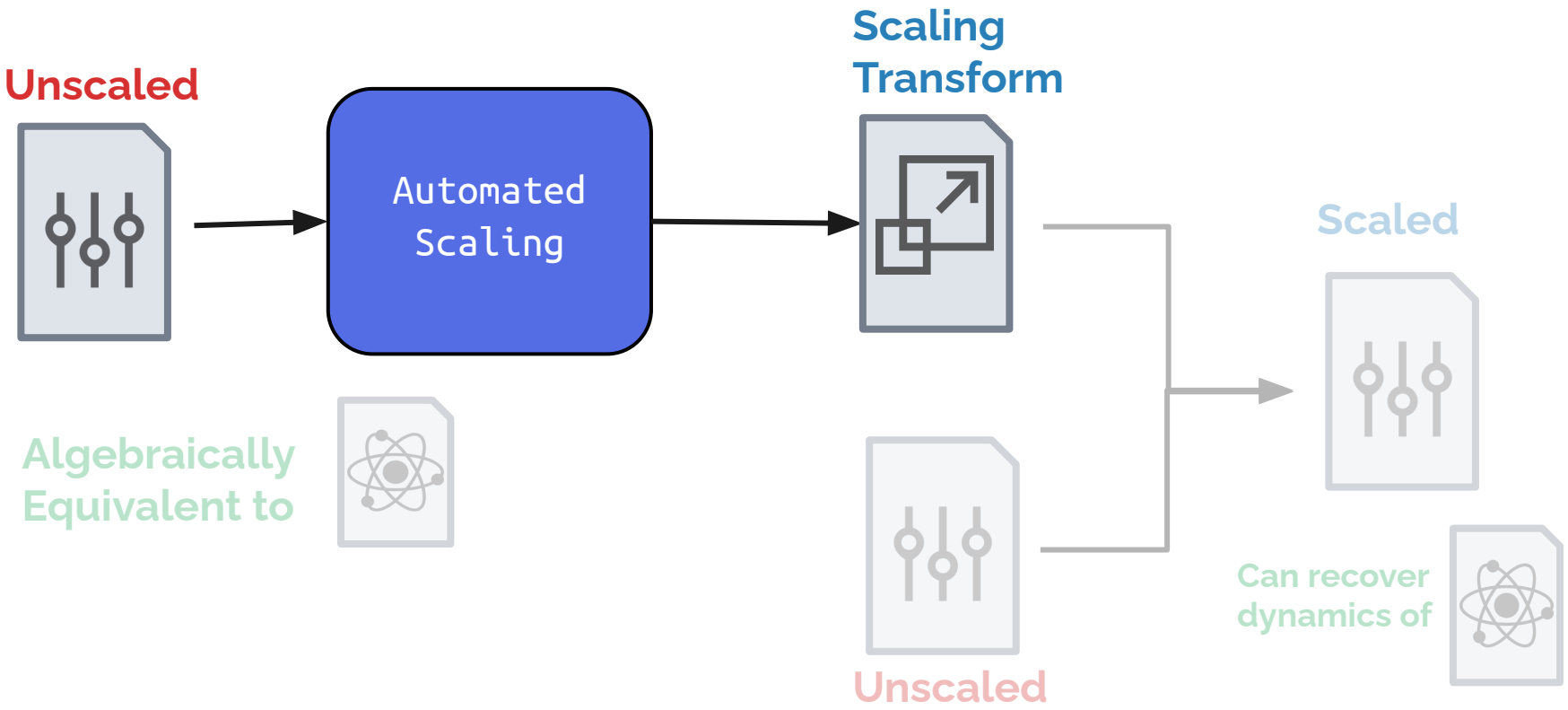
1 time unit = τ*7.93 μs



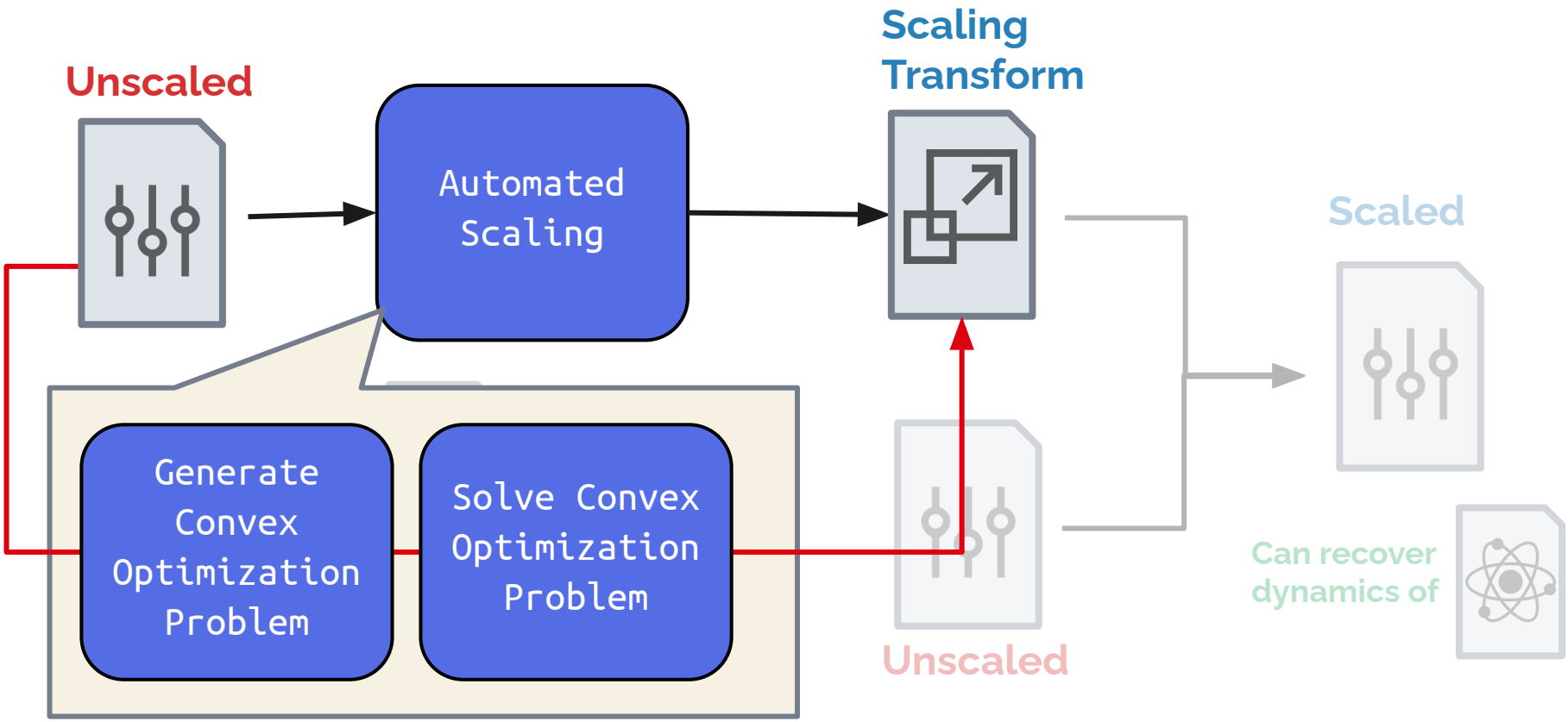
Recovering the Original Dynamics



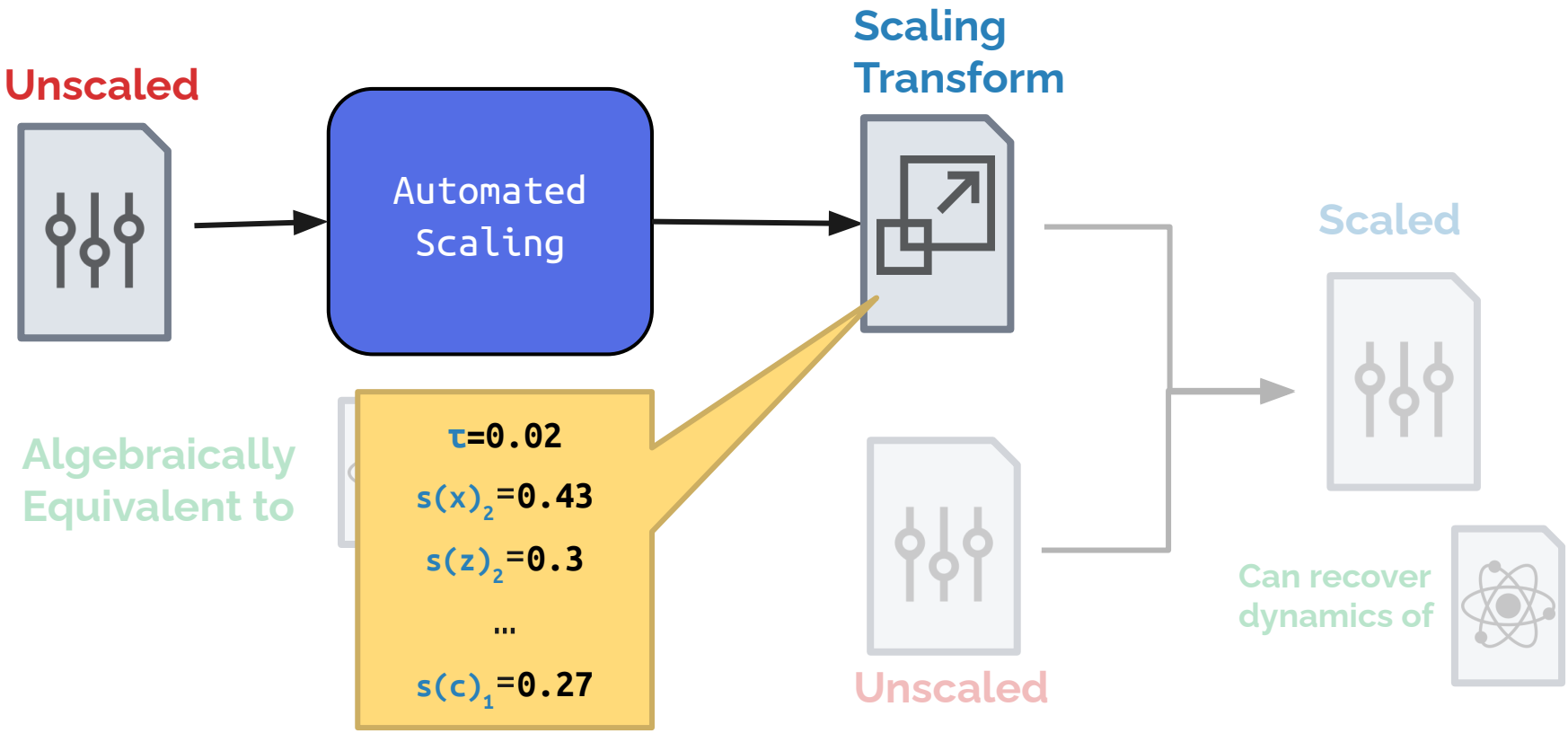




Automated Scaling



Automated Scaling



Automated Scaling

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, \ldots, s(c)$

Objective: Maximize execution speed

$$\text{Minimize } \tau$$

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, \ldots, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

- Operating Range Constraints
- Execution Speed Constraints
- Quality Constraints
- Connection Constraints
- Factor Constraints

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, \ldots, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

Operating Range Constraints

Operating Range Restrictions

Execution Speed Constraints

Execution Speed Restrictions

Quality Constraints

Noise and Quantization Error

Connection Constraints

Factor Constraints

Manufacturing Variations

Constraints encode physical limitations of analog device

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, \ldots, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

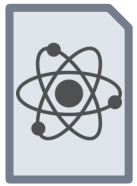
Operating Range Constraints

Execution Speed Constraints

Quality Constraints

Connection Constraints

Factor Constraints



Constraints ensure original dynamical system is recoverable

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, \ldots, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

Operating Range Constraints

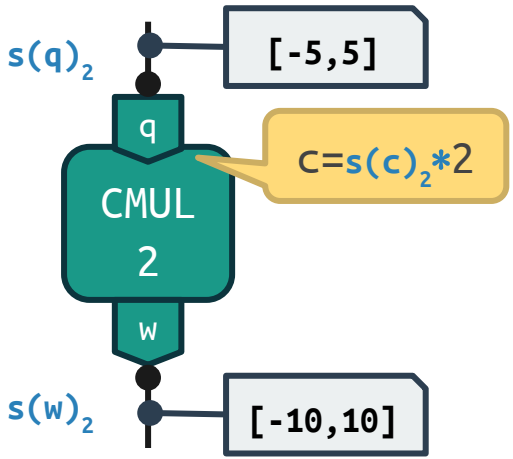
Execution Speed Constraints

Quality Constraints

Connection Constraints

Factor Constraints

Operating Range Constraints



$$s(q)_2 * [-5,5] \subseteq [-2,2]\ \mu A$$

$$s(w)_2 * [-10,10] \subseteq [-2,2]\ \mu A$$

$$s(c)_2 * 2 \subseteq [-1,1]$$

$$q,w \in [-2,2]\ \mu A$$

$$c \in [-1,1]$$

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, s(c)$

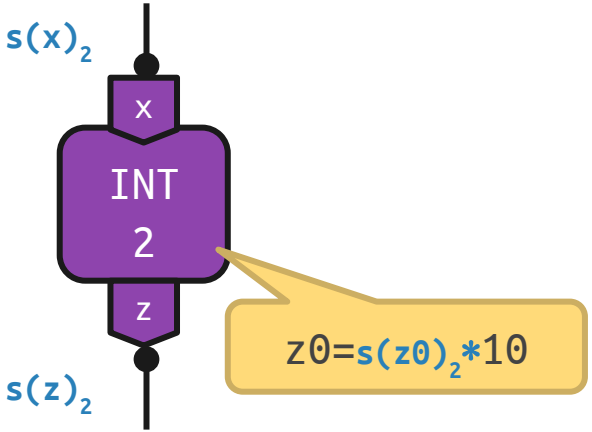
Objective: Maximize execution speed

Minimize τ

Subject to:

- Operating Range Constraints
- Execution Speed Constraints
- Quality Constraints
- Connection Constraints
- **Factor Constraints**

Factor Constraints

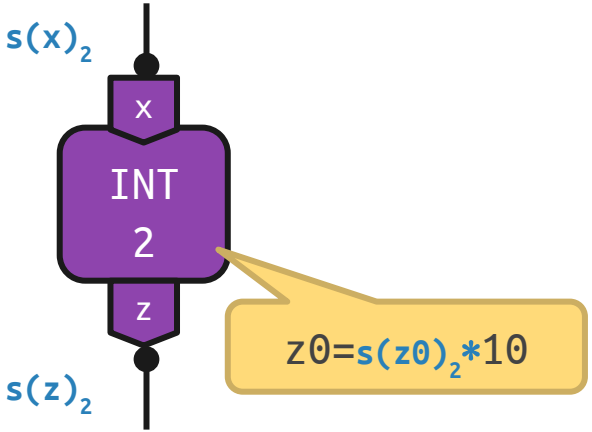


$$s(z)_2 * z = \int 0.93 * 0.5 * (s(x)_2 * x)\ \tau^{-1}\ dt_{hw}$$

$$s(z)_2 * z(0) = 0.77 * s(z0)_2 * 10$$

Goal: Factor out and eliminate **scaling factors** and **manufacturing variations** from **dynamics (black)**

Factor Constraints

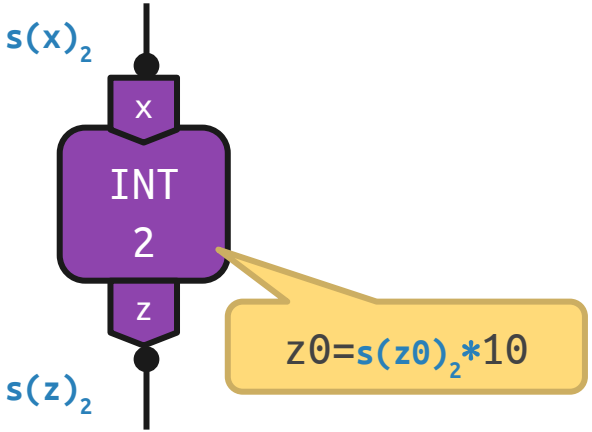


$$s(z)_2 * z = (0.93 * s(x)_2 * \tau^{-1}) \int 0.5 * (x)\, dt_{hw}$$

$$s(z)_2 * z(0) = (0.77 * s(z0)_2)\ 10$$

Goal: Factor out and eliminate **scaling factors** and **manufacturing variations** from **dynamics (black)**

Factor Constraints



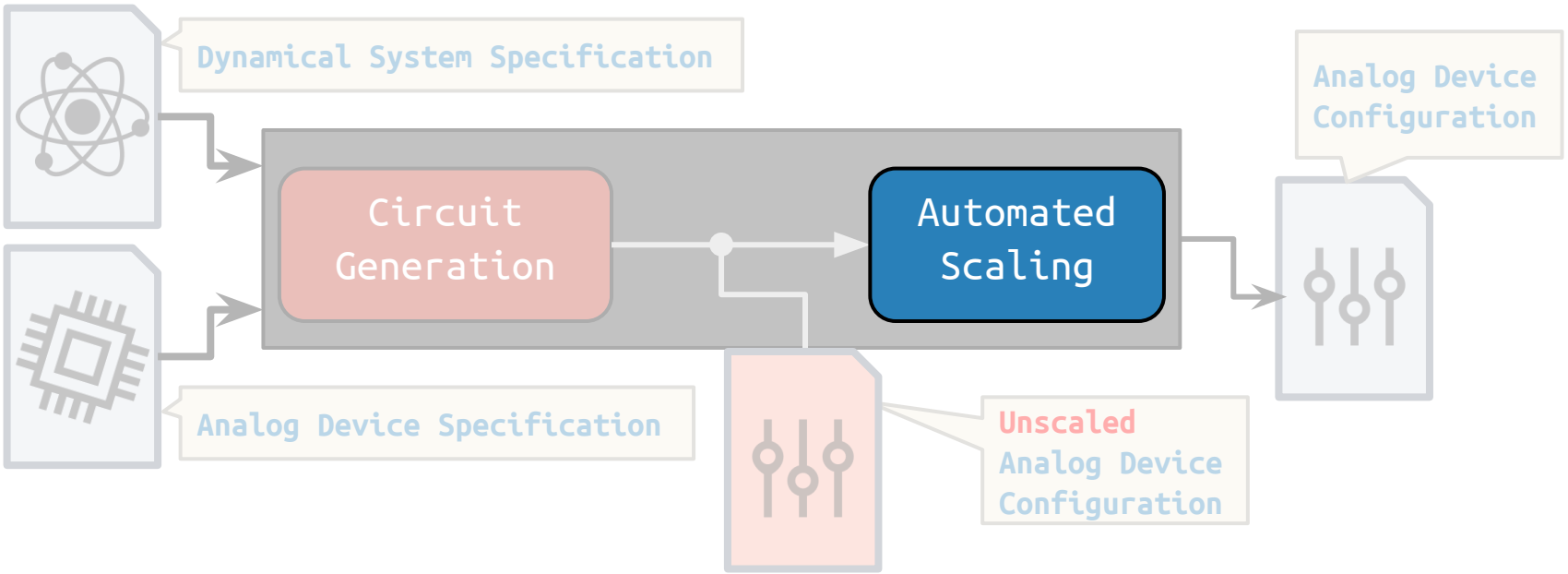
$z = \int 0.5*(x)\ dt_{hw}$

$s(z)_2 = 0.93*s(x)_2*\tau^{-1}$

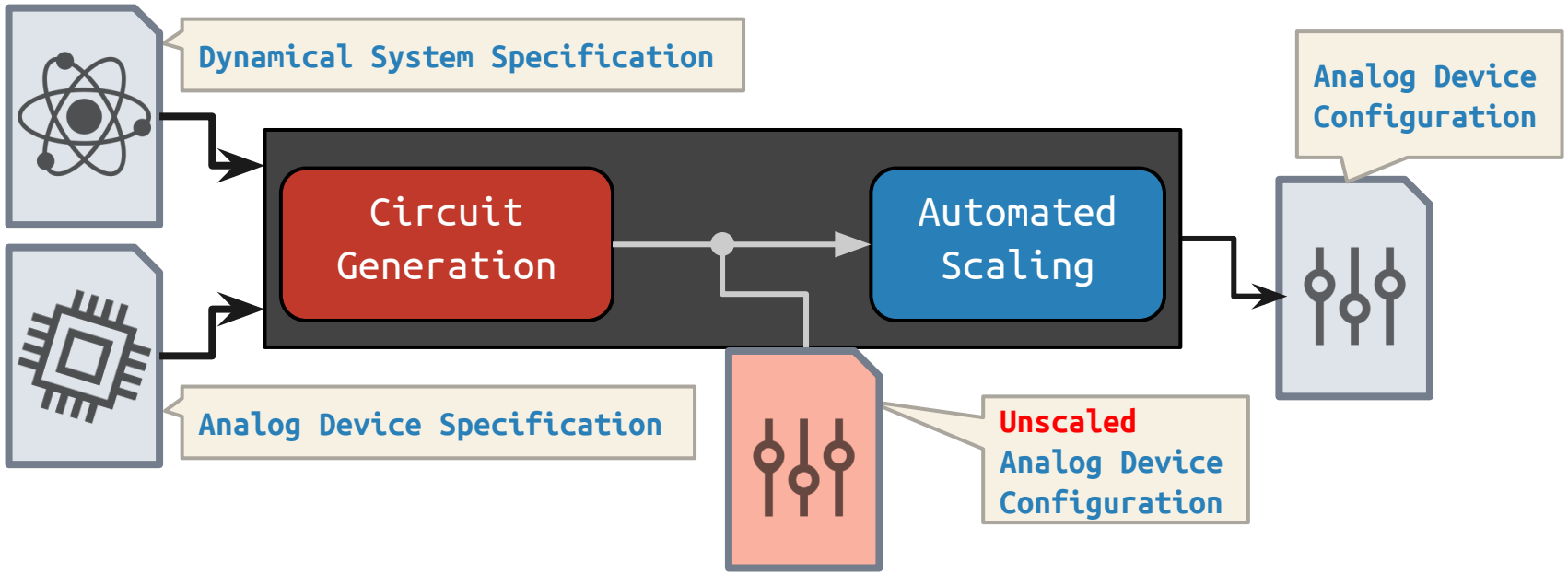
$z(0) = 10$

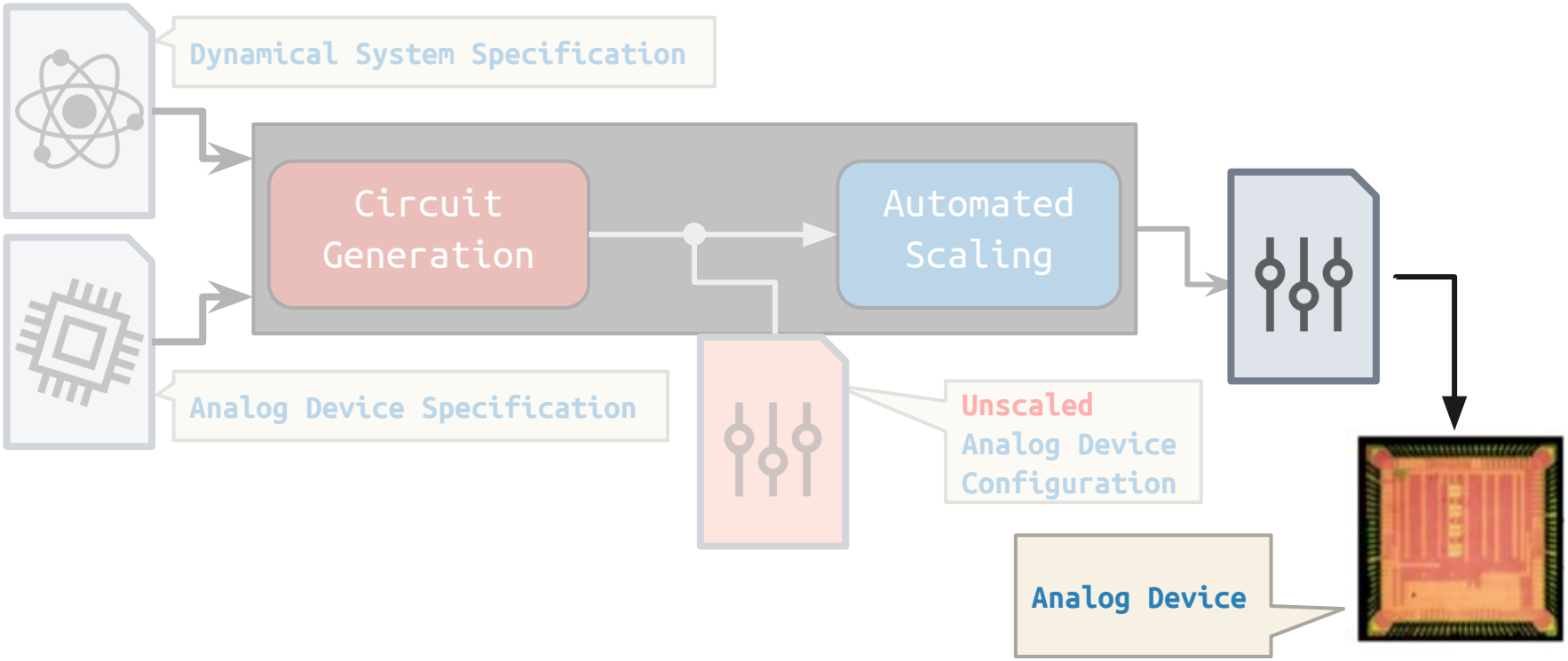
$s(z)_2 = 0.77*s(z0)_2$

Goal: Factor out and eliminate **scaling factors** and **manufacturing variations** from **dynamics (black)**

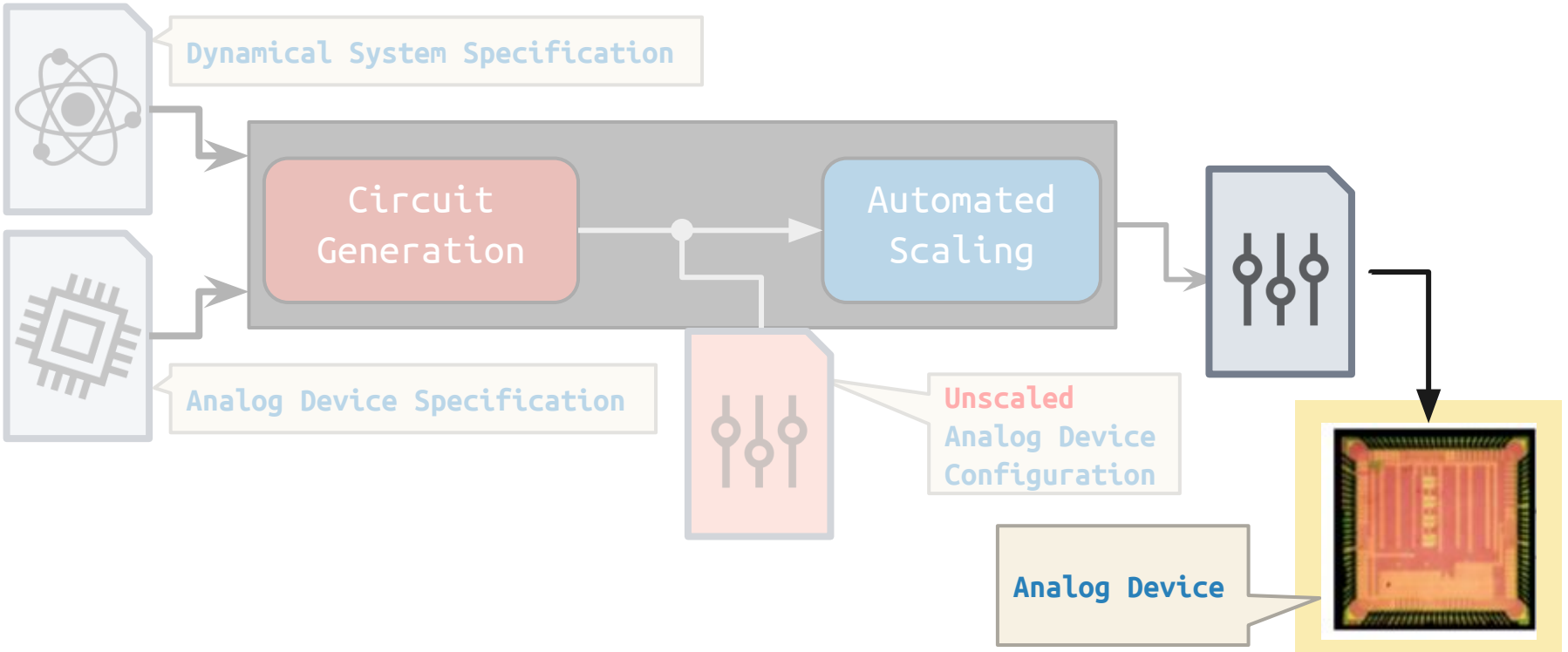


1. Time Dilation and Contraction for Programmable Analog Devices with Jaunt. ASPLOS 2018.
2. Noise-Aware Dynamical System Compilation for Analog Devices with Legno. ASPLOS 2020.


Dynamical System Specification
Analog Device Specification
Circuit Generation
Automated Scaling
Unscaled Analog Device Configuration
Analog Device Configuration


Dynamical System Specification
Circuit Generation
Automated Scaling
Analog Device Specification
Unscaled Analog Device Configuration
Analog Device

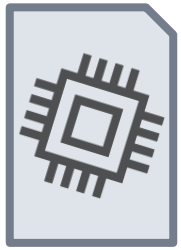
Compiler

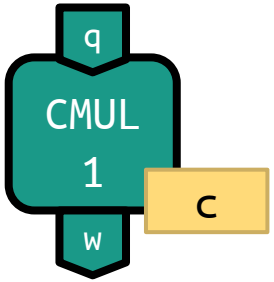

Dynamical System Specification
Circuit Generation
Automated Scaling
Analog Device Specification
Unscaled Analog Device Configuration
Analog Device

Compiler

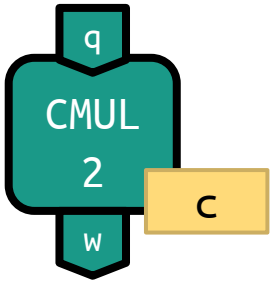
Inputs

Manufacturing Variations

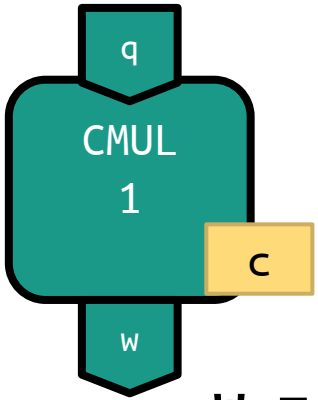




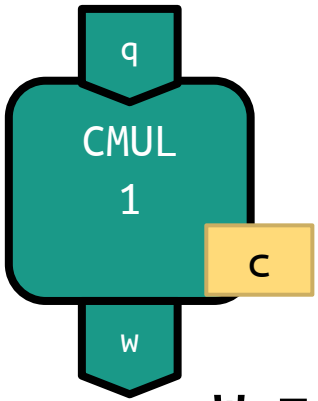
w =0.86*c*q



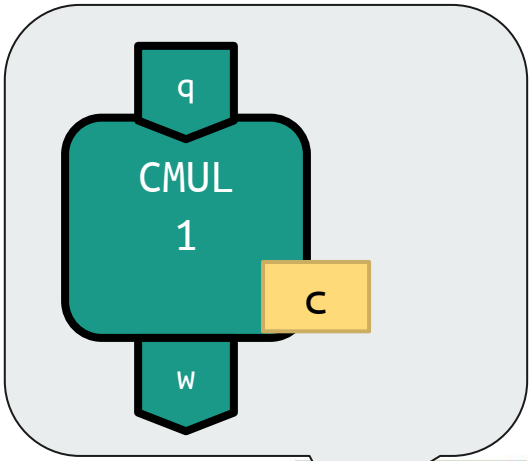
w =1.11*c*q

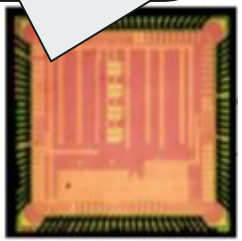


w = c*q



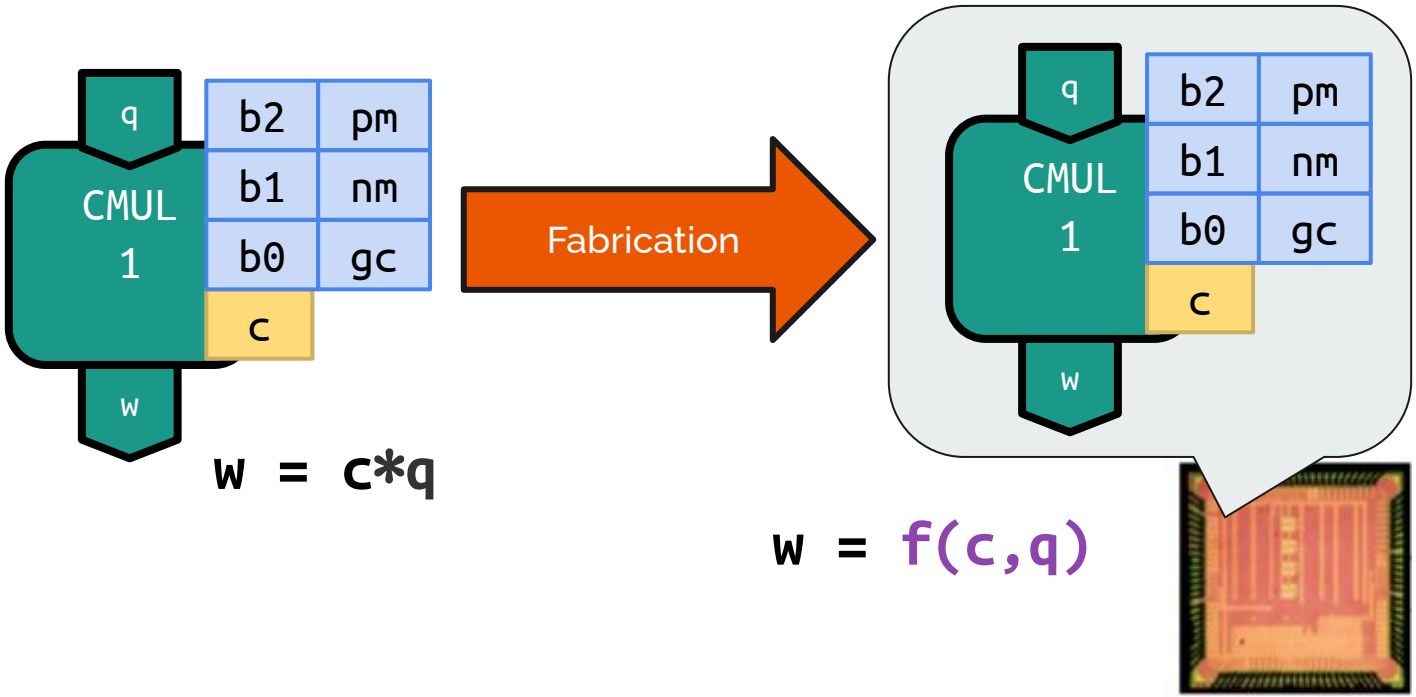
Fabrication

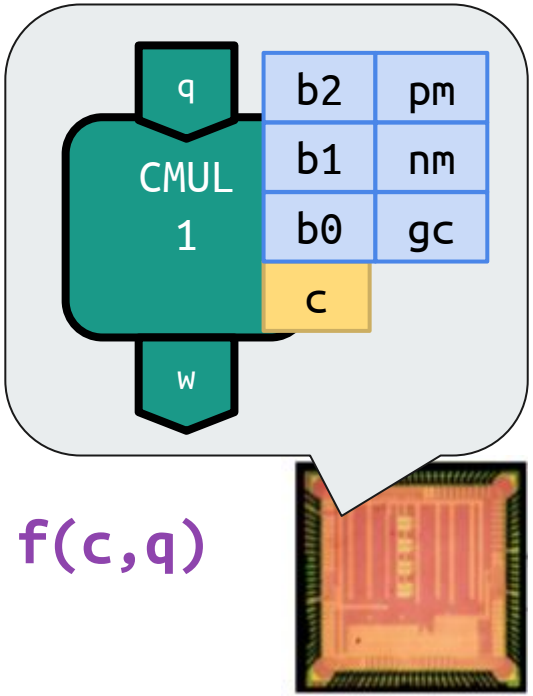




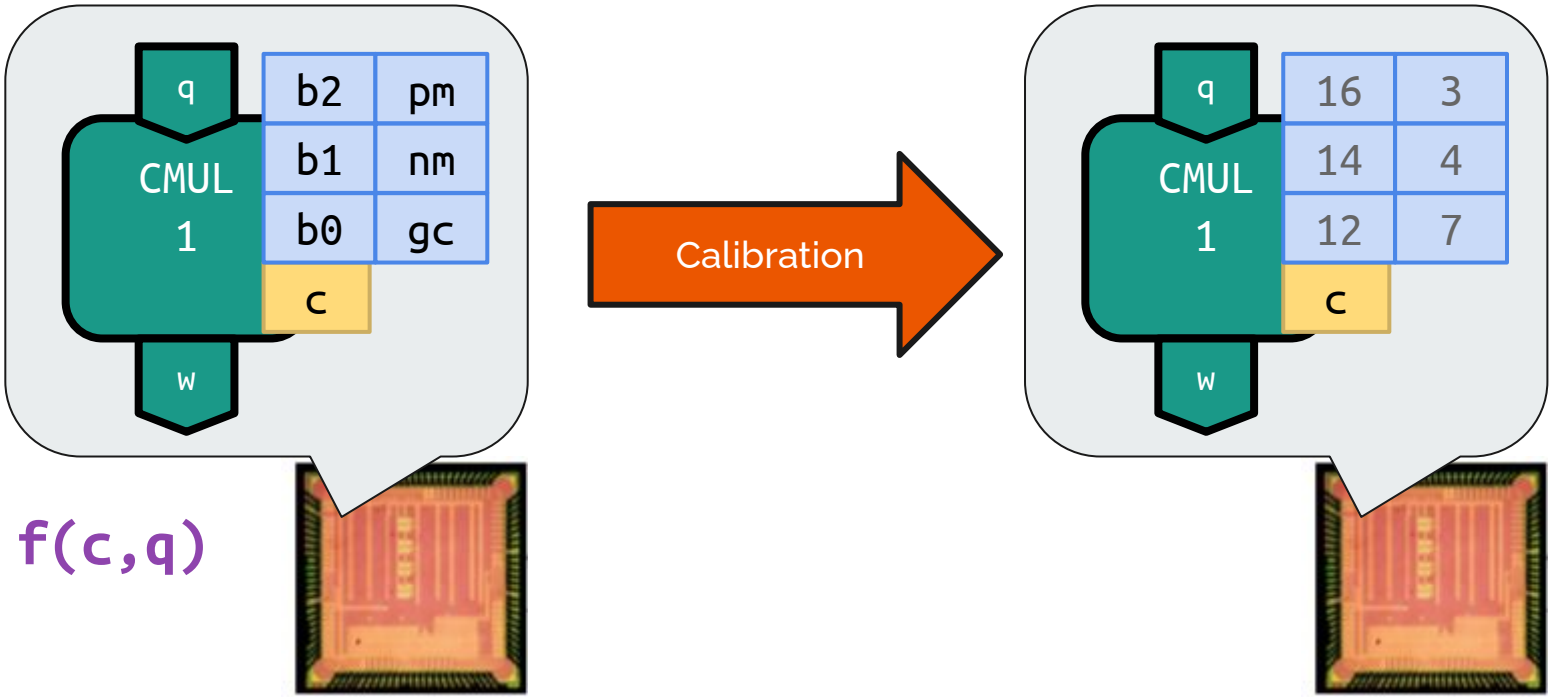
`w = c*q`

`w = f(c,q)`


q
CMUL
1
b2
pm
b1
nm
b0
gc
c
w
Fabrication
q
CMUL
1
b2
pm
b1
nm
b0
gc
c
w
w = c*q
w = f(c,q)

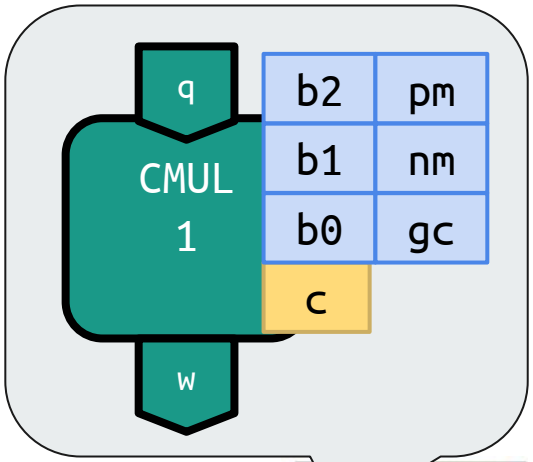


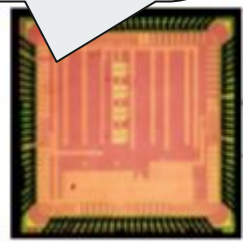
$w = f(c,q)$



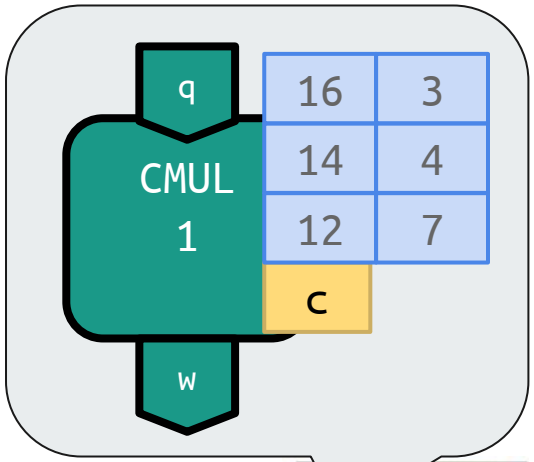
$w = f(c,q)$

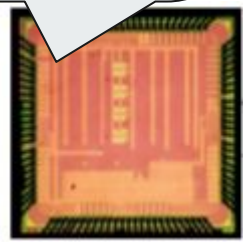
Objective: Minimize the error between expected, observed dynamics





Calibration





$w = f(c,q)$

$\text{minimize } \|f(c,q) - c*q\|$

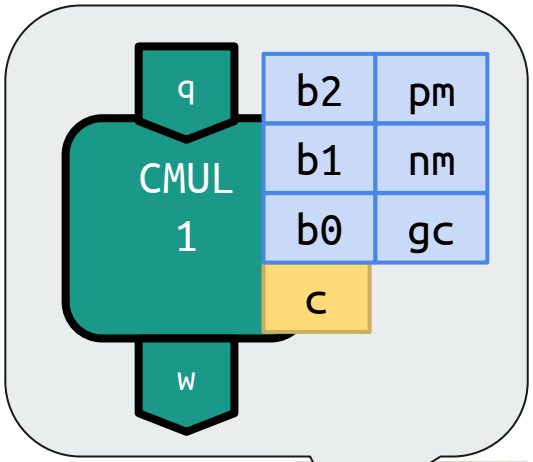
Objective: Minimize the error between expected, observed dynamics

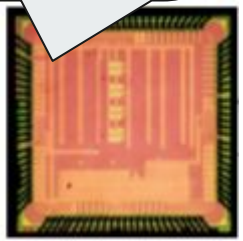
This is not the best calibration objective

$w = f(c,q)$

$\text{minimize } \|f(c,q) - c*q\|$

Objective: Minimize the error between expected, observed dynamics

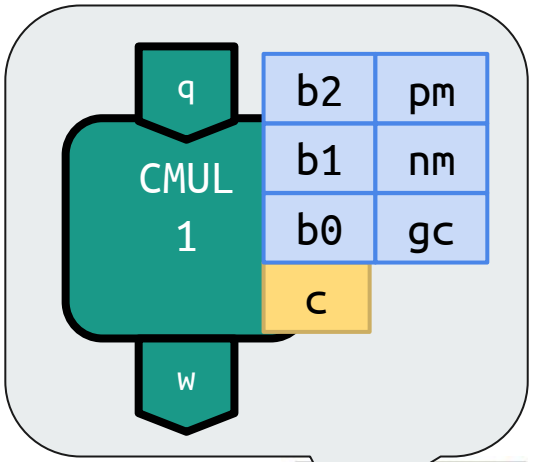


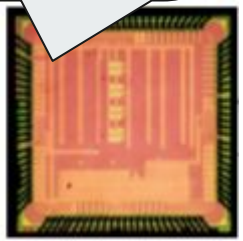


$w = f(c,q)$

$$\underset{b2,b1,b0,pm,nm,gc}{\operatorname{argmin}} \|f(c,q) - c*q\|$$

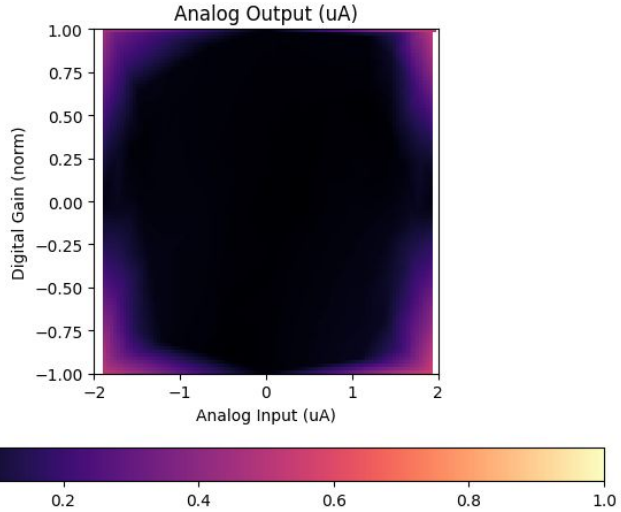
Objective: Minimize the error between expected, observed dynamics





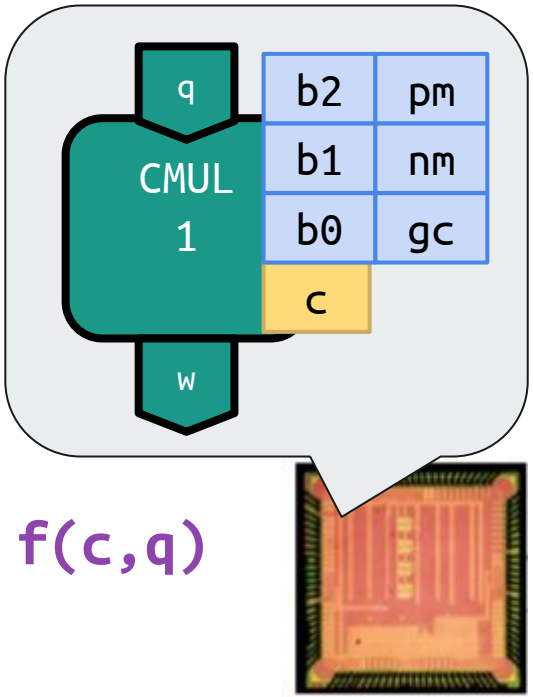
$w = f(c,q)$

$\underset{b2,b1,b0,pm,nm,gc}{\operatorname{argmin}} \|f(c,q) - c*q\|$



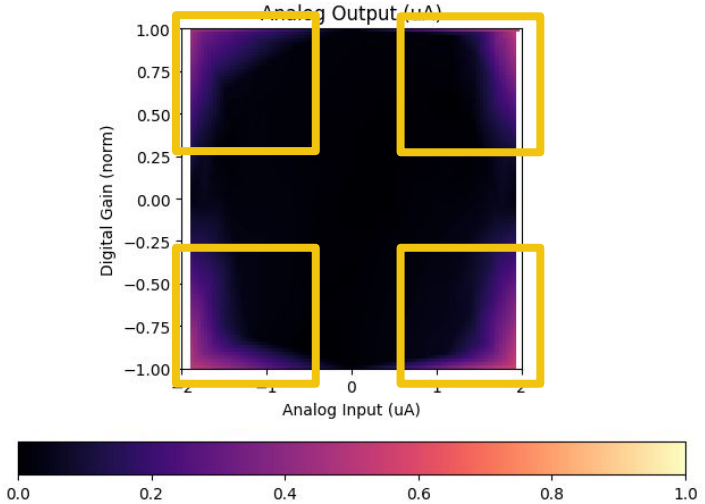
$|f(c,q) - c*q|$

Objective: Minimize the error between expected, observed dynamics

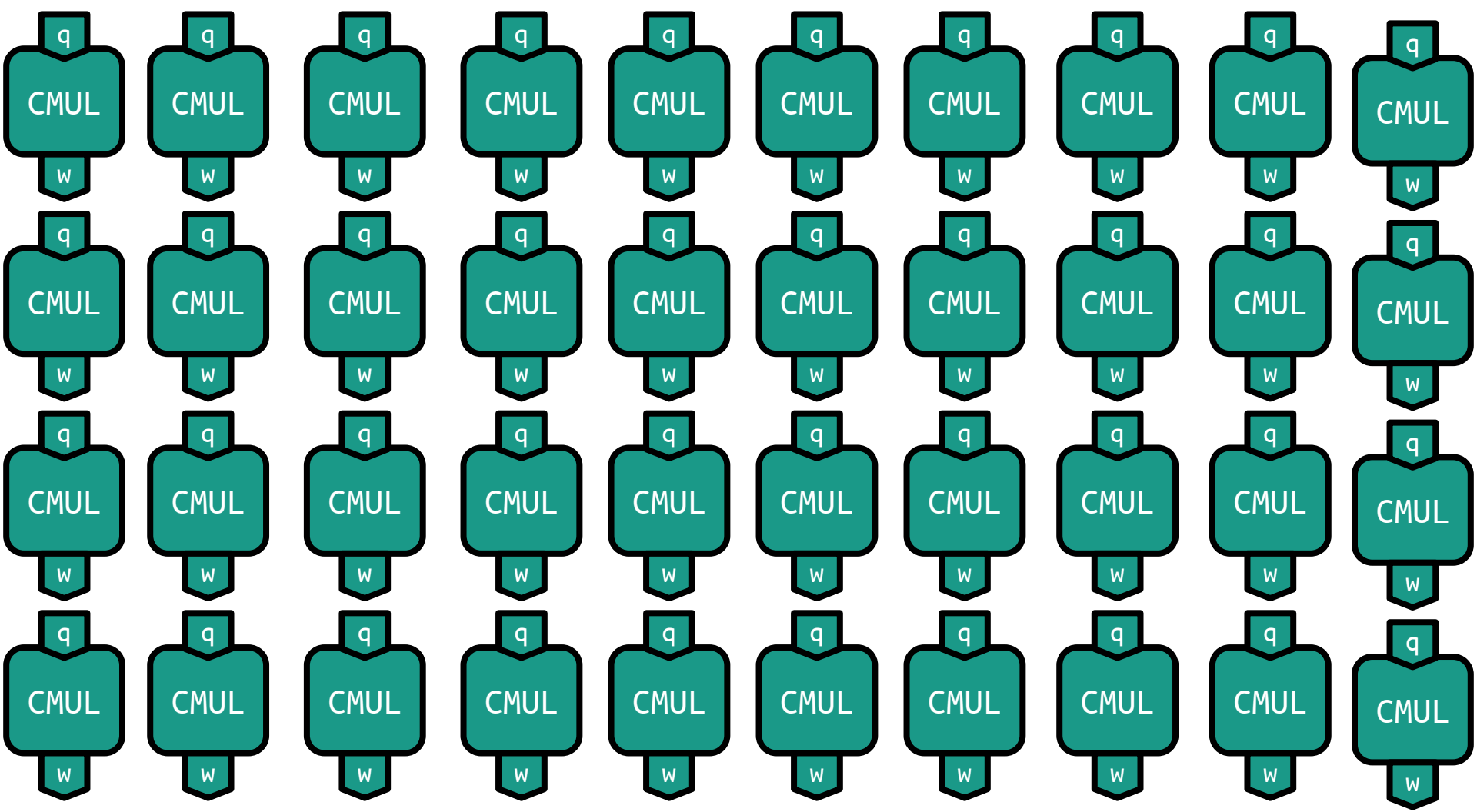


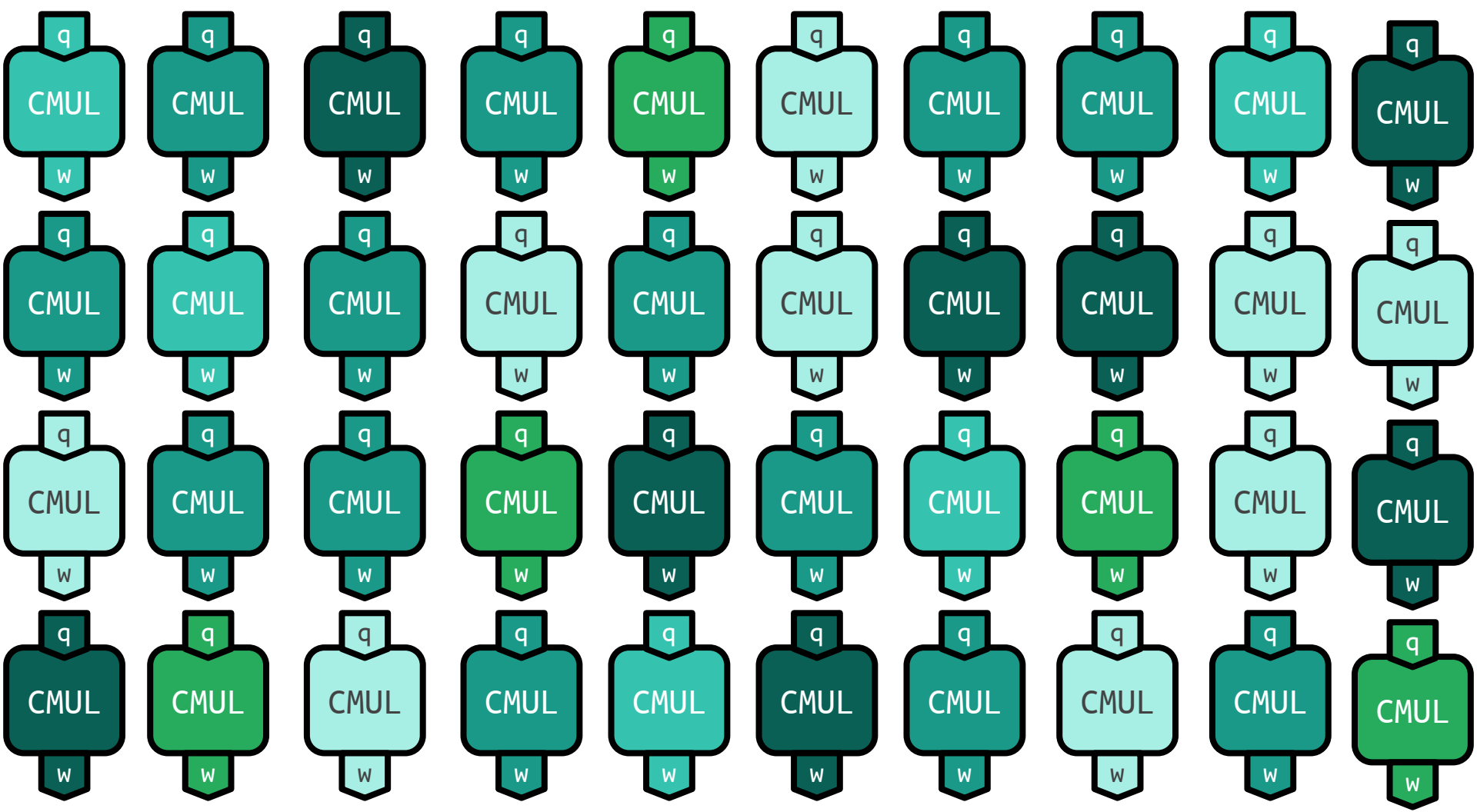
$w = f(c,q)$

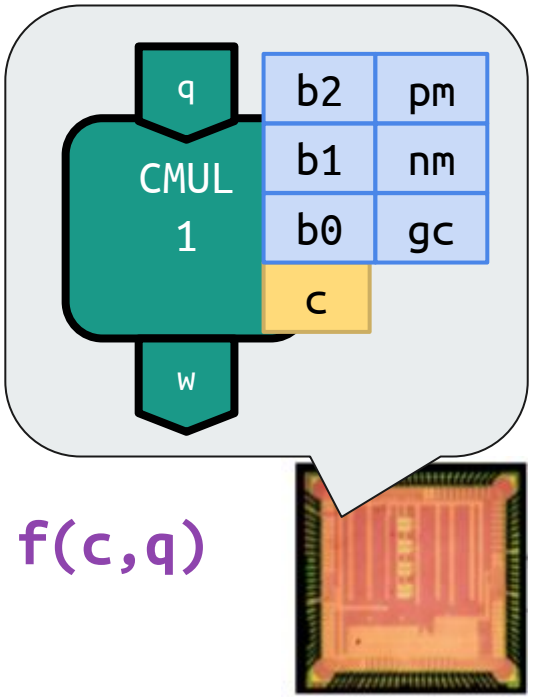
$\text{argmin}_{b2,b1,b0,pm,nm,gc} \|f(c,q) - c*q\|$



$|f(c,q) - c*q|$


q
CMUL
w

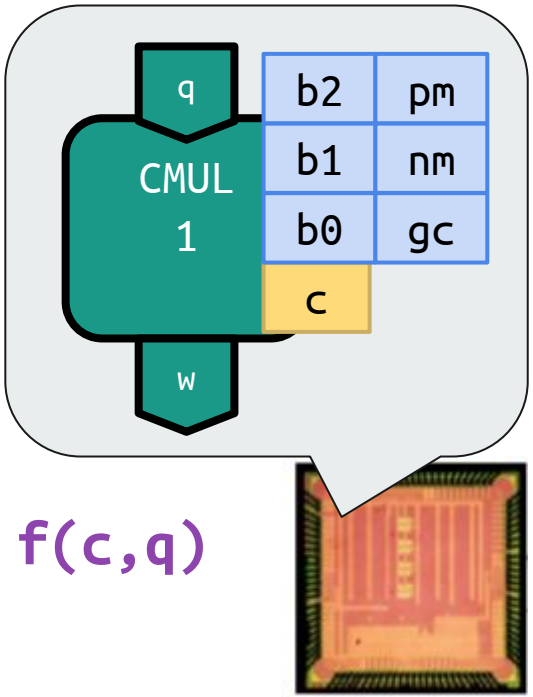

q
CMUL
w



$w = f(c,q)$

We allow calibrated multipliers to scale the result

$$w = \alpha * c * q$$



$w = f(c,q)$

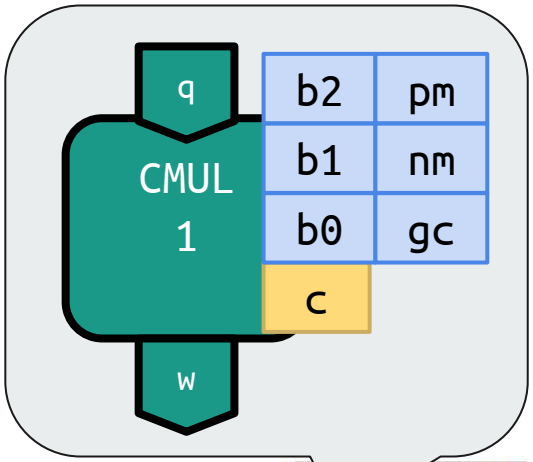
We allow calibrated multipliers to scale the result

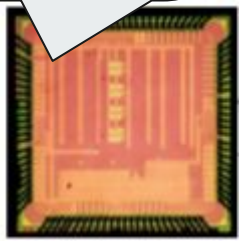
$$w = \alpha * c * q$$

α is a empirically derived parameter

$$\underset{\alpha}{\text{argmin}} \; \| f(c,q) - \alpha * c * q \|$$

Objective: Find the codes that produce the best fit for α

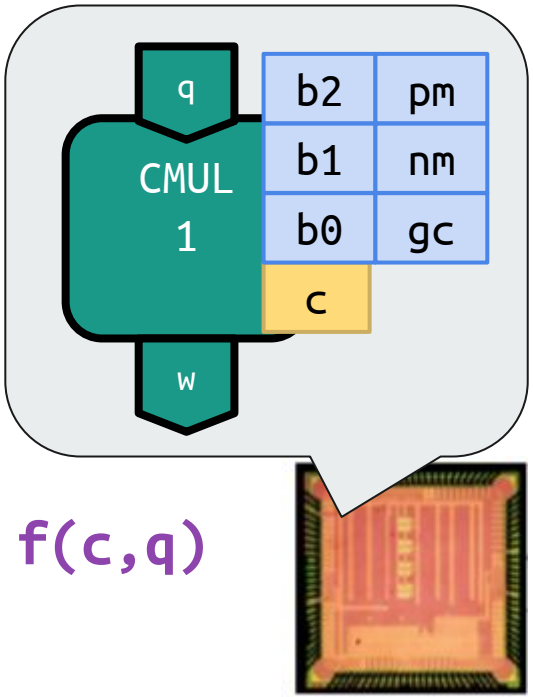




$w = f(c,q)$

$$\underset{b2,b1,b0,pm,nm,gc}{\operatorname{argmin}} \; \| f(c,q) - \alpha * c * q \|$$

Objective: Find the codes that produce the best fit for α



$$w = f(c,q)$$

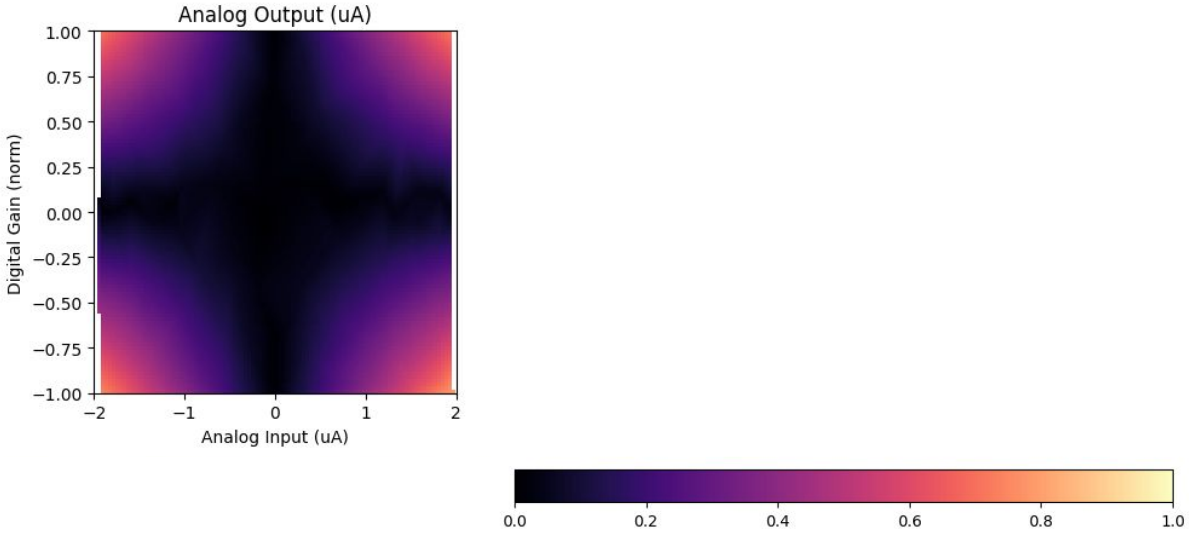
$$\underset{b2,b1,b0,pm,nm,gc}{\operatorname{argmin}} \; \| f(c,q) - \alpha * c * q \|$$

For each combination of calibration codes, we find the best α

$$\underset{\alpha}{\operatorname{argmin}} \; \| f(c,q) - \alpha * c * q \|$$

Calibration Objective

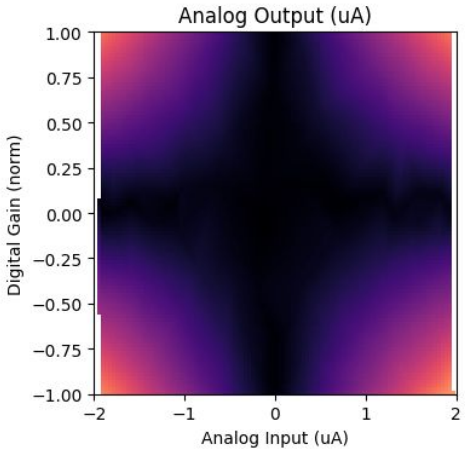
Best fitting α



$|f(c,q) - c*q|$

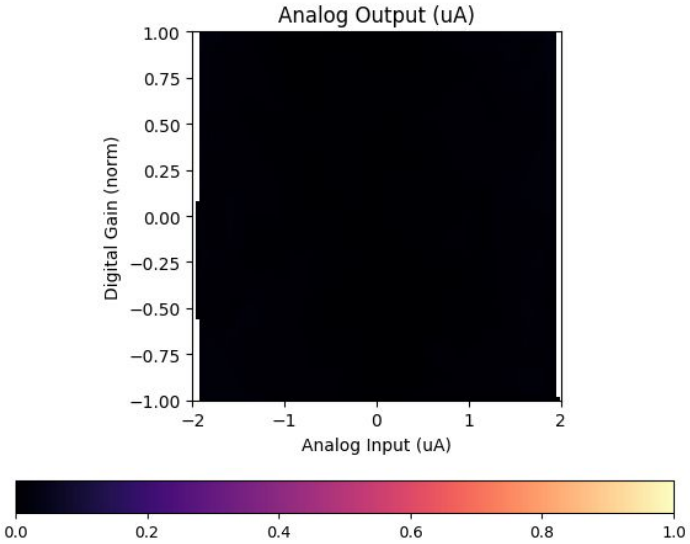
Calibration Objective

Best fitting α



$|f(c,q) - c*q|$

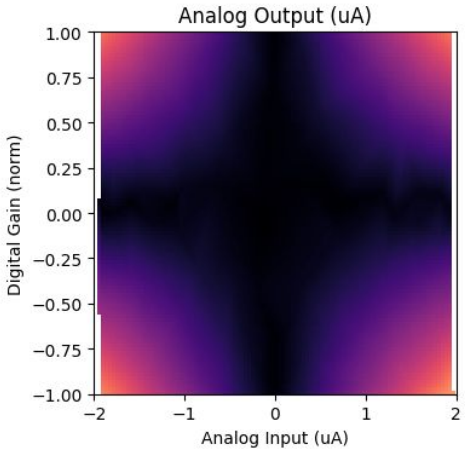
Best fitting α



$|f(c,q) - \alpha*c*q|$

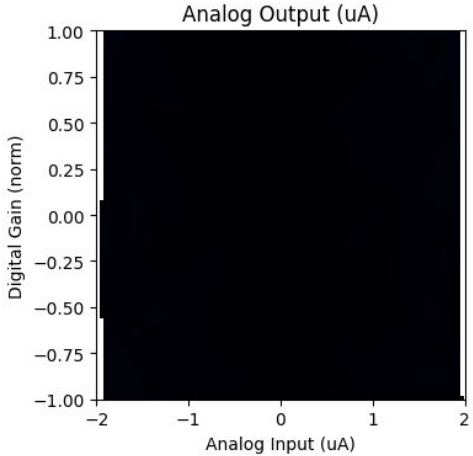
Calibration Objective

Best fitting α



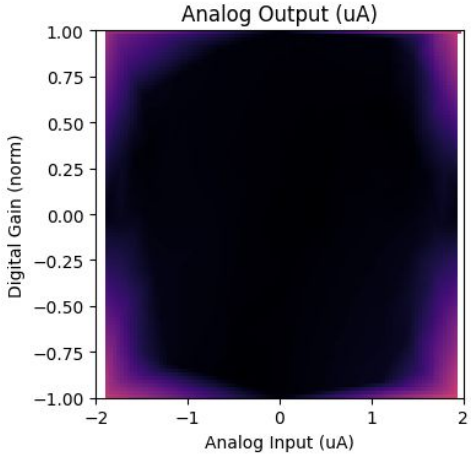
$|f(c,q) - c*q|$

Best fitting α

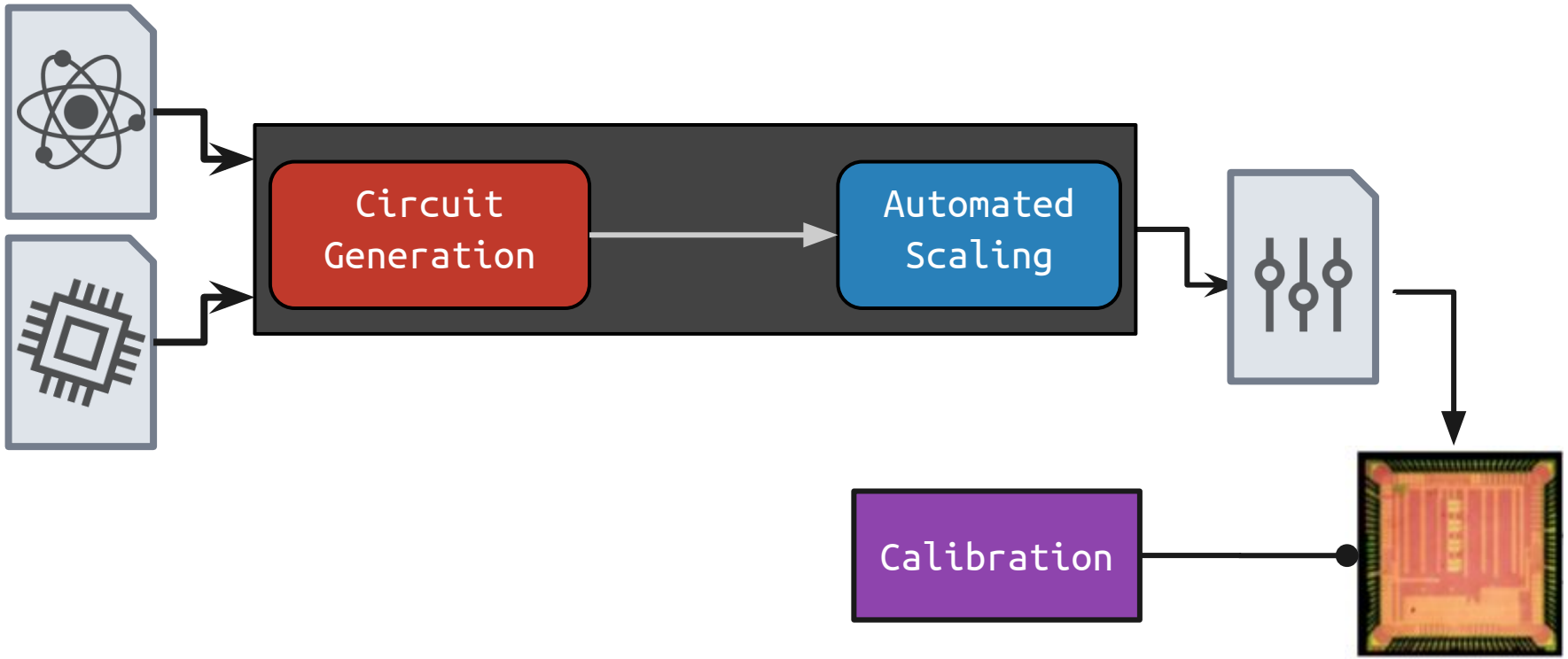


$|f(c,q) - \alpha*c*q|$

Minimize error

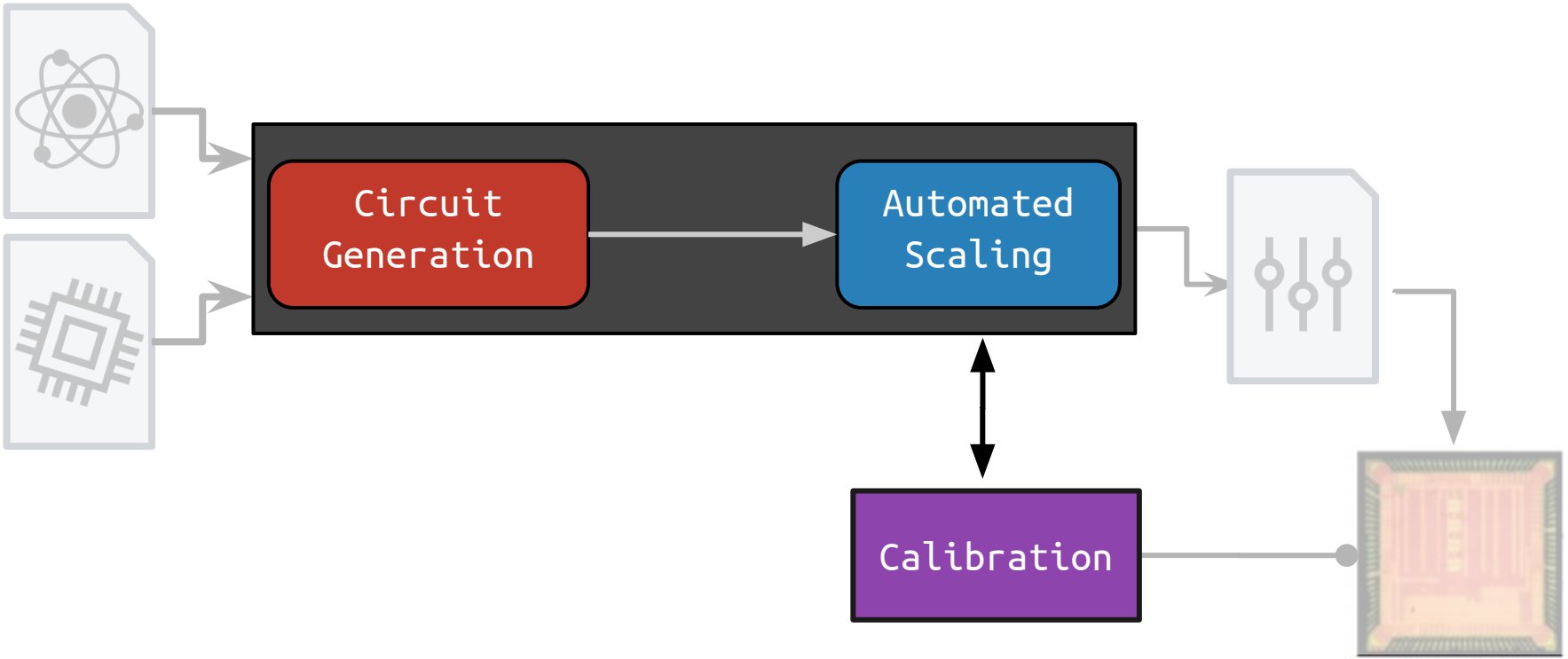


$|f(c,q) - c*q|$

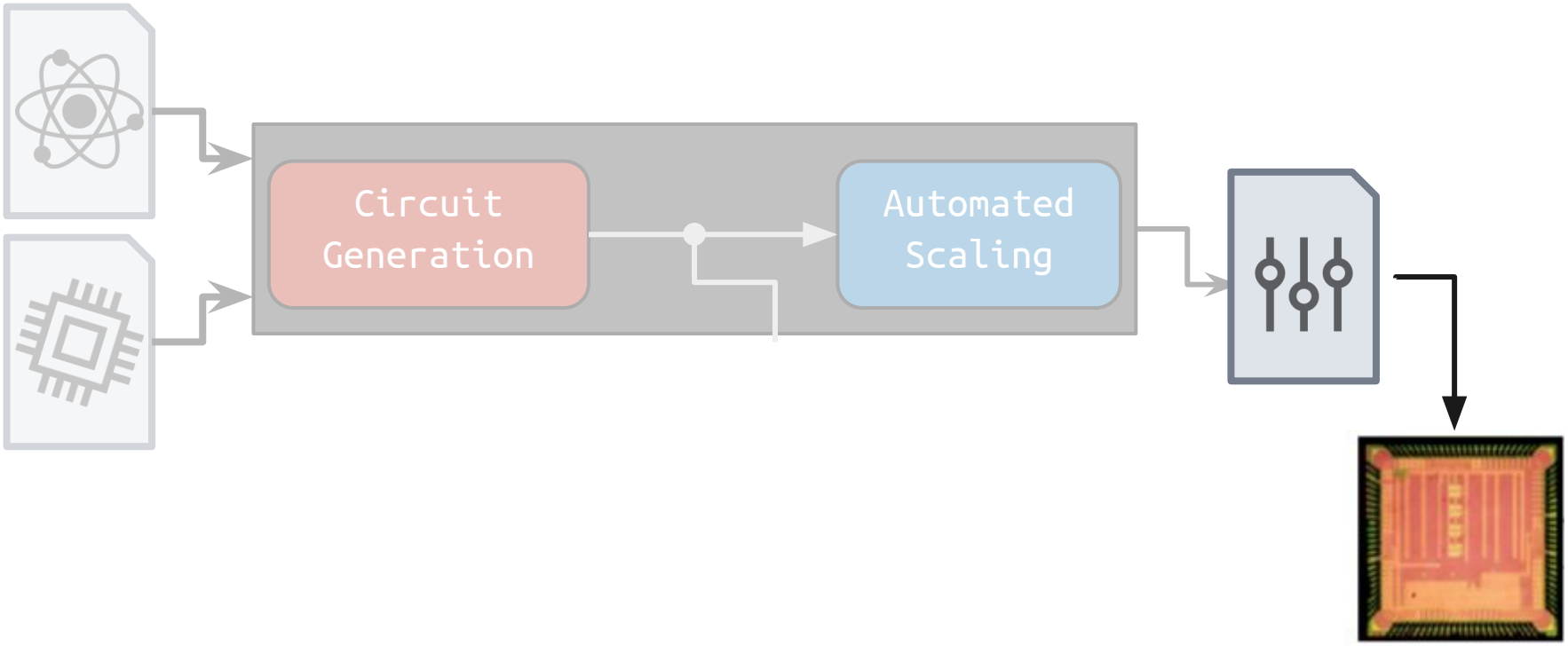


Compiler

Synergy between compilation, calibration



Compiler

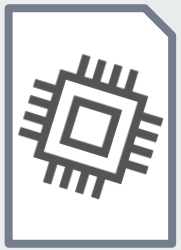


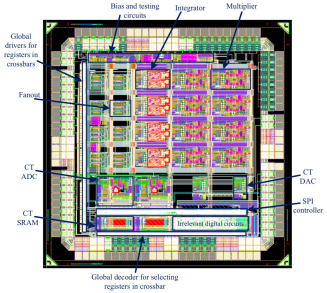
Compiler

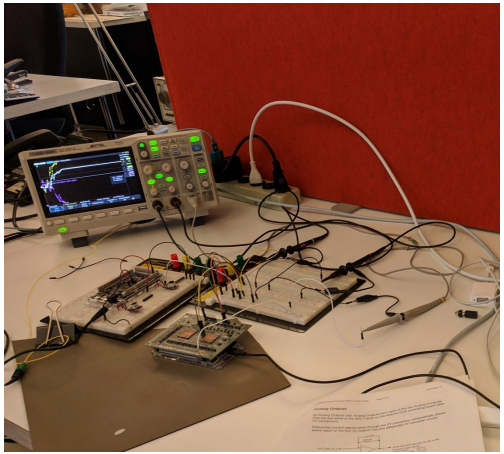
Results

Experimental Setup

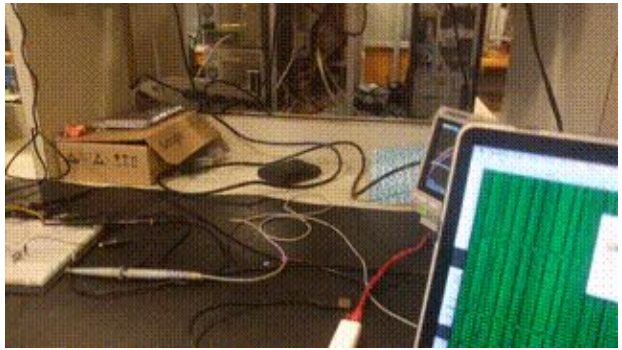
HCDCv2 Analog Device







COLUMBIA UNIVERSITY



Guo, Ning, Investigation of Energy-Efficient Hybrid Analog/Digital Approximate Computation in Continuous Time. Columbia University. 2017

Dampened spring oscillator

Botulism neurotoxin model

Kalman filter

PI controller

Vanderpol oscillator with forcing function

Dampened oscillator

Pendulum

Oscillator

Vanderpol oscillator

Genetic toggle switch

Movement of heat

Michaelis menten reaction

Dampened spring oscillator

Botulism neurotoxin model

Kalman filter

PI controller

Vanderpol oscillator with forcing function

Dampened oscillator

Pendulum

Oscillator

Vanderpol oscillator

Genetic toggle switch

Movement of heat

Michaelis menten reaction

Dampened spring oscillator

Botulism neurotoxin model

Kalman filter

PI controller

Vanderpol oscillator with forcing function

Dampened oscillator

Pendulum

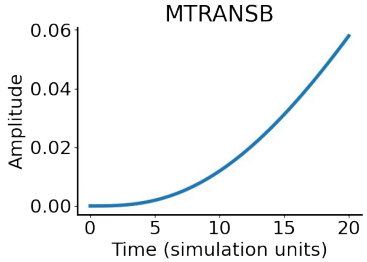
Oscillator

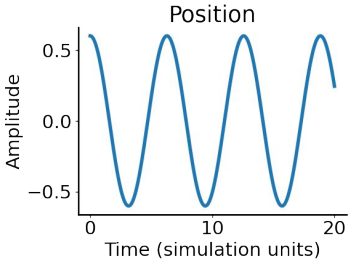
Vanderpol oscillator

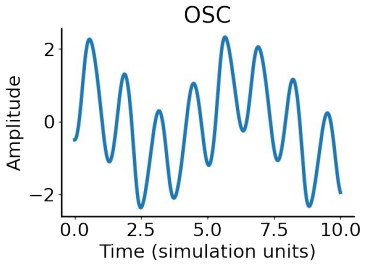
Genetic toggle switch

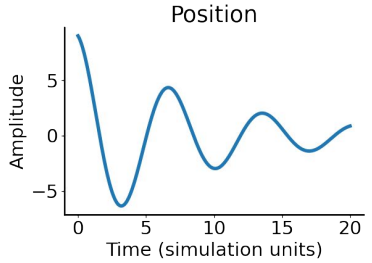
Movement of heat

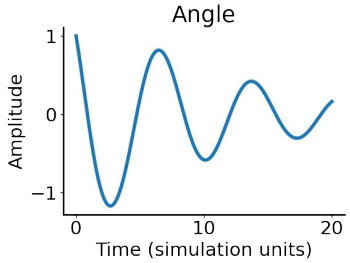
Michaelis menten reaction

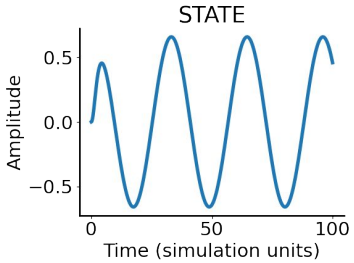

MTRANSB
Amplitude
Time (simulation units)

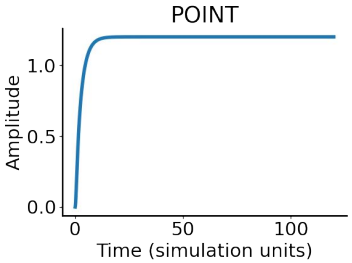

Position
Amplitude
Time (simulation units)

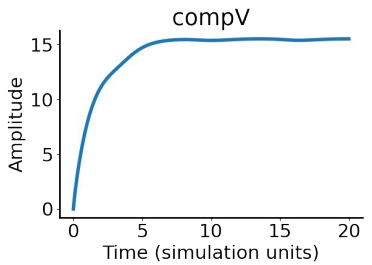

OSC
Amplitude
Time (simulation units)

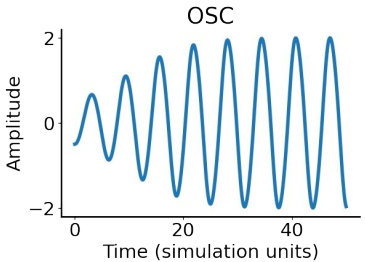

Position
Amplitude
Time (simulation units)

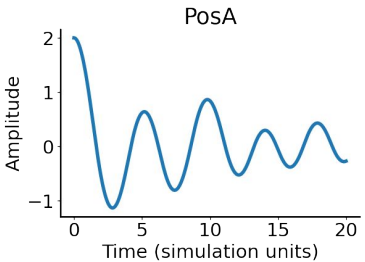

Angle
Amplitude
Time (simulation units)

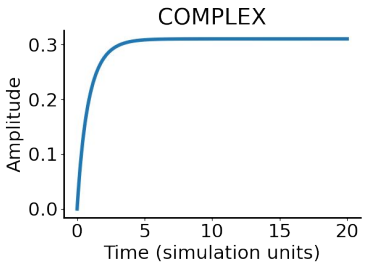

STATE
Amplitude
Time (simulation units)

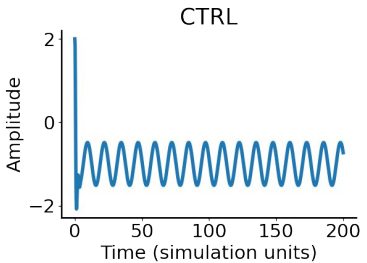

POINT
Amplitude
Time (simulation units)

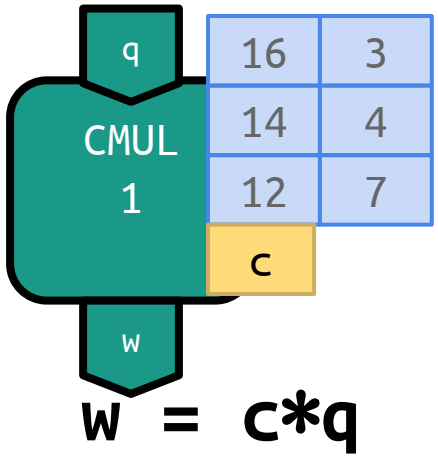

compV
Amplitude
Time (simulation units)


OSC
Amplitude
Time (simulation units)

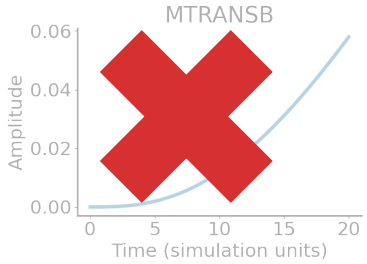

PosA
Amplitude
Time (simulation units)

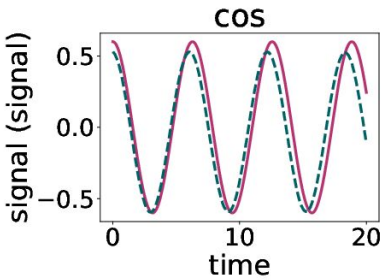

COMPLEX
Amplitude
Time (simulation units)

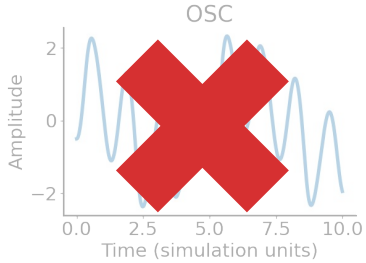

CTRL
Amplitude
Time (simulation units)

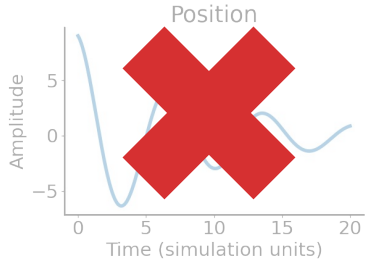


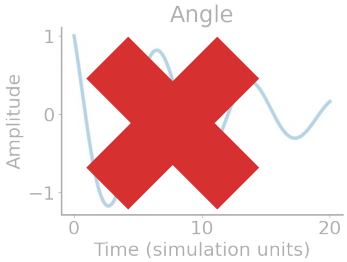
Synthesize configurations without scaling

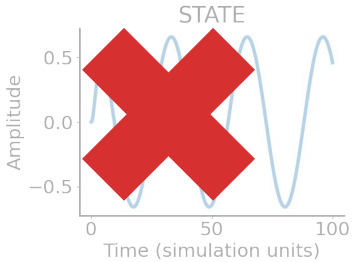

MTRANSB
Amplitude
Time (simulation units)

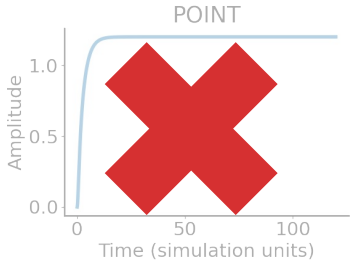

cos
signal (signal)
time

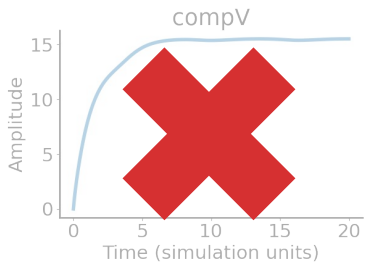

OSC
Amplitude
Time (simulation units)

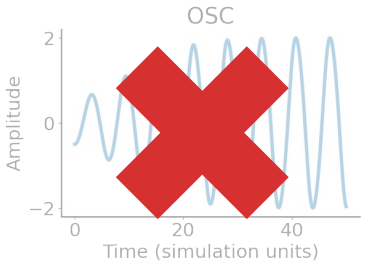

Position
Amplitude
Time (simulation units)


Angle
Amplitude
Time (simulation units)

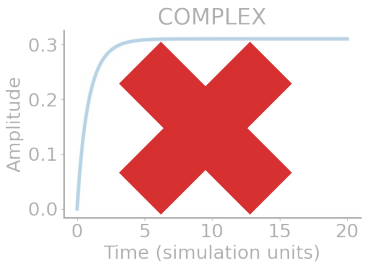

STATE
Amplitude
Time (simulation units)

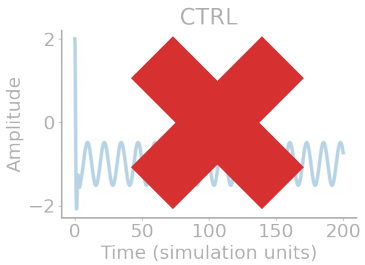

POINT
Amplitude
Time (simulation units)

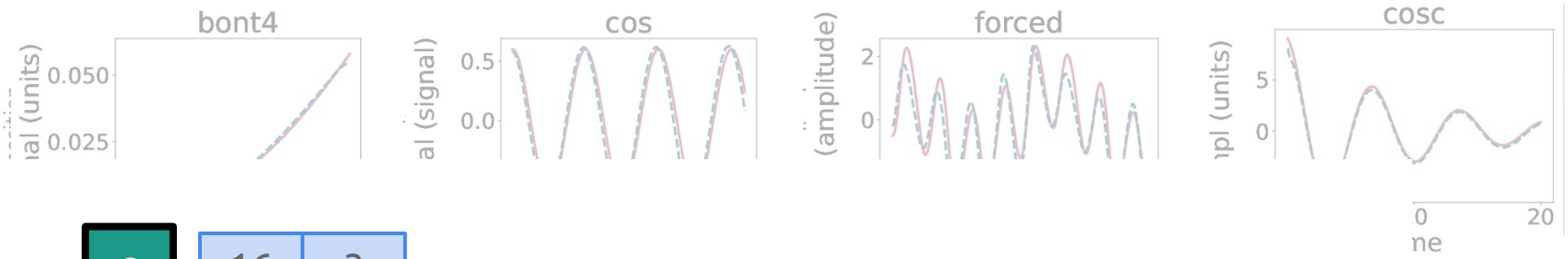

compV
Amplitude
Time (simulation units)

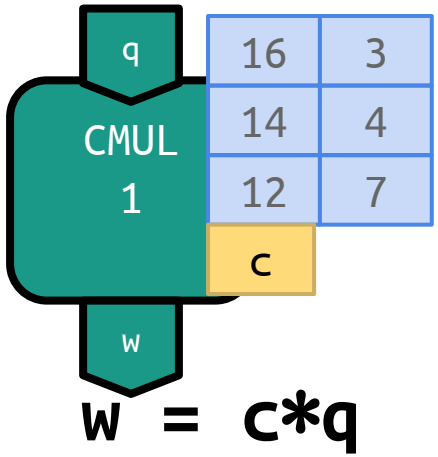

OSC
Amplitude
Time (simulation units)


PosA
Amplitude
Time (simulation units)


COMPLEX
Amplitude
Time (simulation units)


CTRL
Amplitude
Time (simulation units)

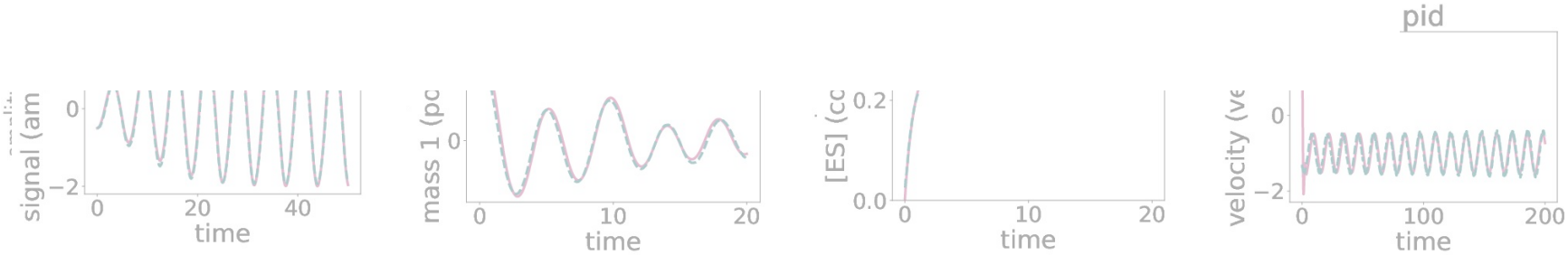


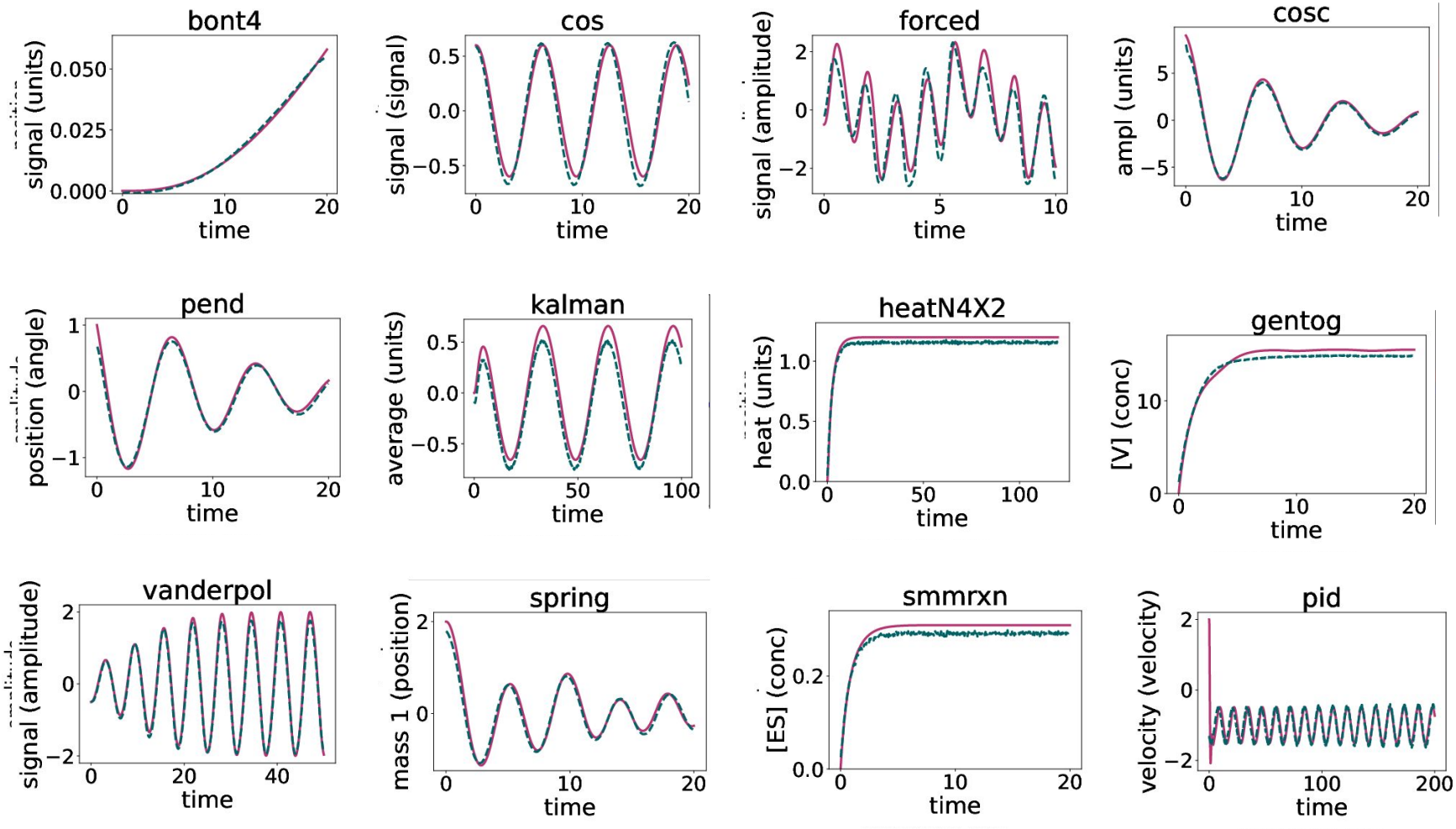


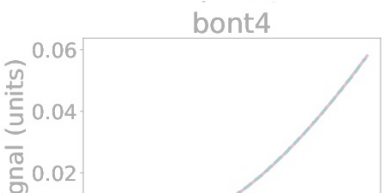
Calibration Objective

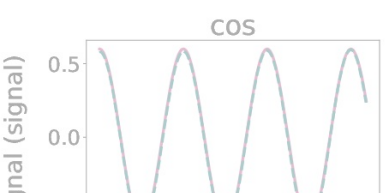
Minimize Error

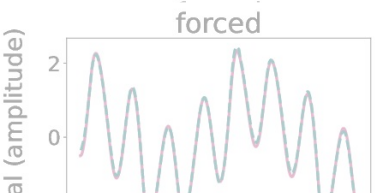
Assume calibrated blocks adhere to specification

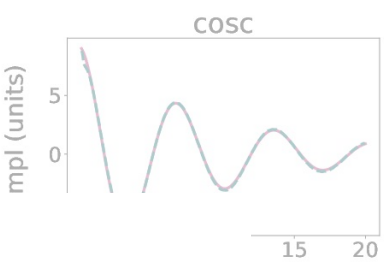


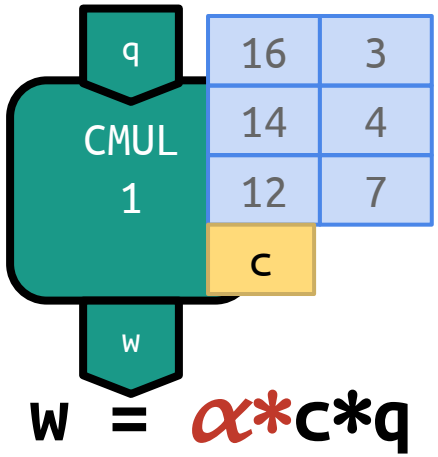

bont4
signal (units)
time
cos
signal (signal)
time
forced
signal (amplitude)
time
cosc
ampl (units)
time
pend
position (angle)
time
kalman
average (units)
time
heatN4X2
heat (units)
time
gentog
[V] (conc)
time
vanderpol
signal (amplitude)
time
spring
mass 1 (position)
time
smmrxn
[ES] (conc)
time
pid
velocity (velocity)
time







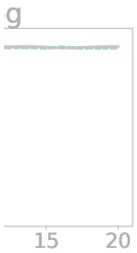


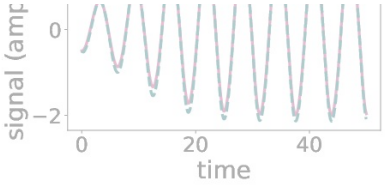


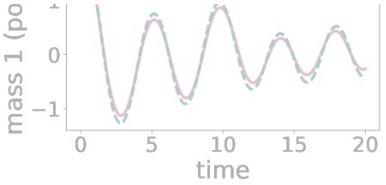
Calibration Objective

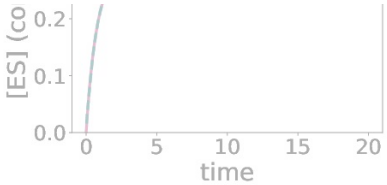
Find best fitting model

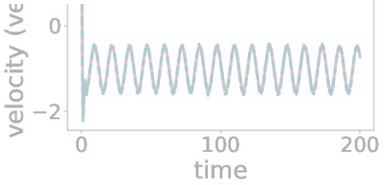
Allow calibrated blocks to deviate from specification

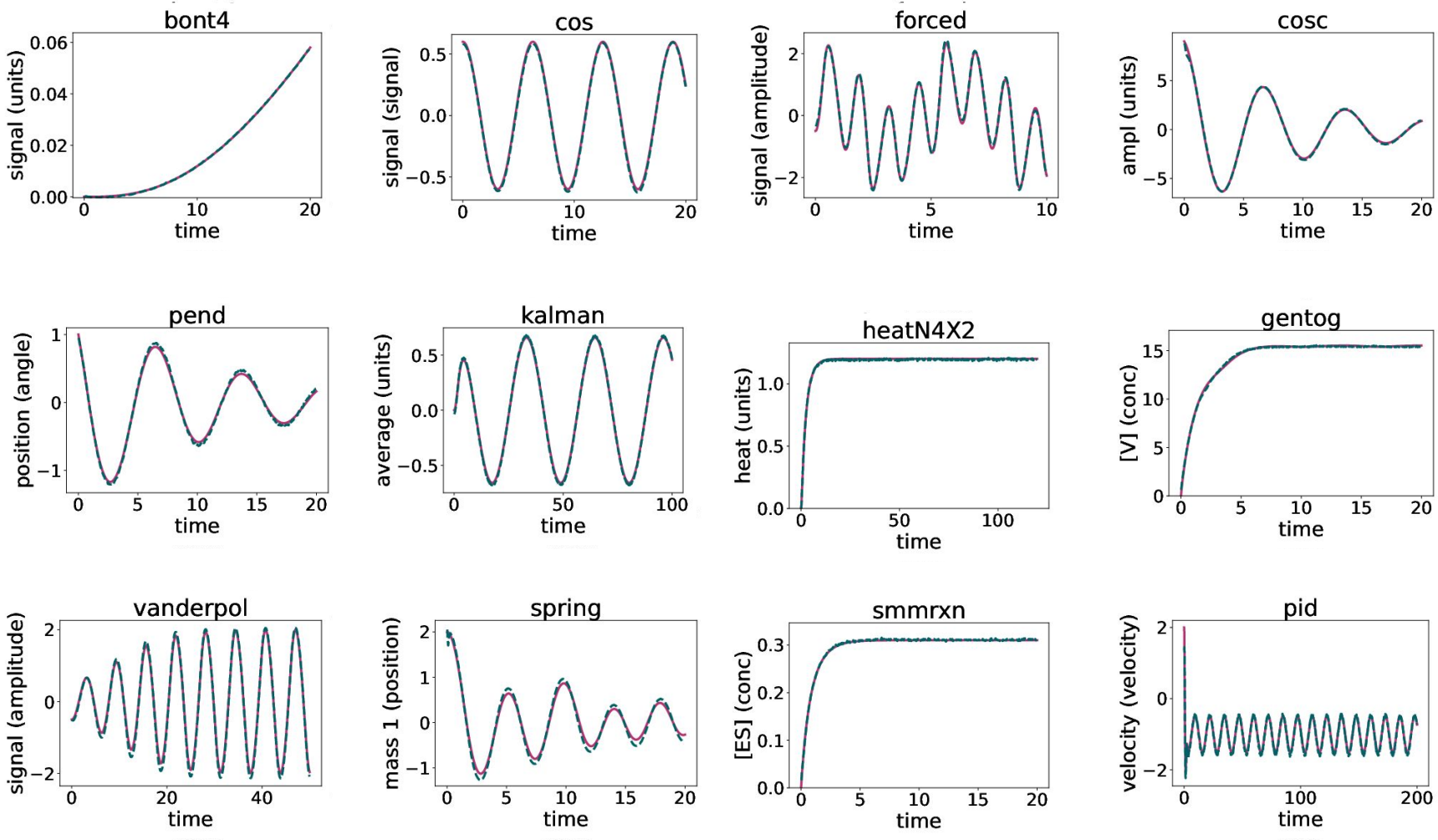











bont4
signal (units)
time
cos
signal (signal)
time
forced
signal (amplitude)
time
cosc
ampl (units)
time
pend
position (angle)
time
kalman
average (units)
time
heatN4X2
heat (units)
time
gentog
[V] (conc)
time
vanderpol
signal (amplitude)
time
spring
mass 1 (position)
time
smmrxn
[ES] (conc)
time
pid
velocity (velocity)
time

Dampened spring oscillator

Botulism neurotoxin model

Kalman filter

PI controller

Vanderpol oscillator with forcing function

Dampened oscillator

Pendulum

Oscillator

Vanderpol oscillator

Genetic toggle switch

Movement of heat

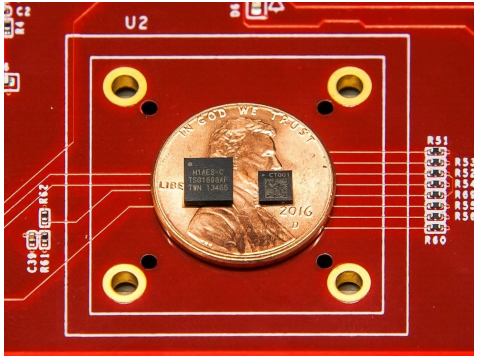
Michaelis menten reaction

0.913 mW	0.823 mW	0.864 mW	0.861 mW
0.722 mW	0.395 mW	0.554 mW	0.199 mW
0.849 mW	0.804 mW	0.556 mW	0.526 mW

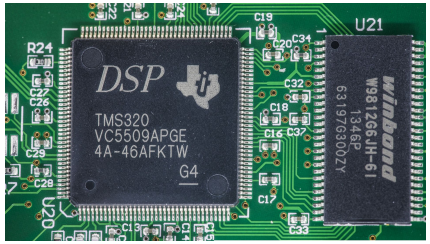
1.50 ms	0.50 ms	3.32 ms	6.58 ms
3.97 ms	1.34 ms	0.50 ms	1.59 ms
1.25 ms	0.50 ms	9.52 ms	0.52 ms

1.37 µJ	0.41 µJ	2.87 µJ	5.67 µJ
3.37 µJ	0.53 µJ	0.28 µJ	0.32 µJ
0.90 µJ	0.40 µJ	5.30 µJ	0.28 µJ

Future Directions

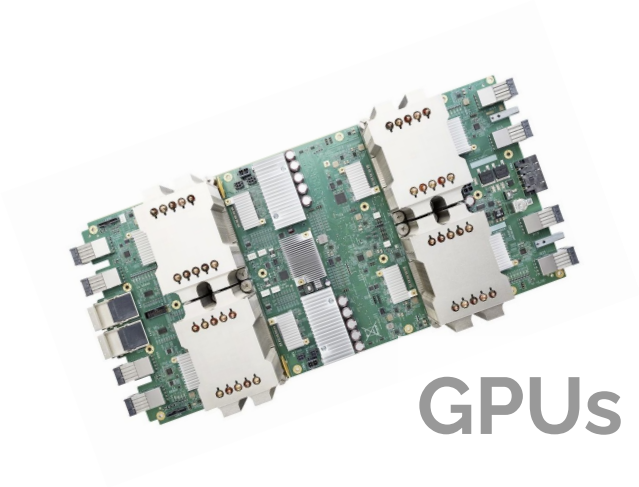


Secure Co-Processors

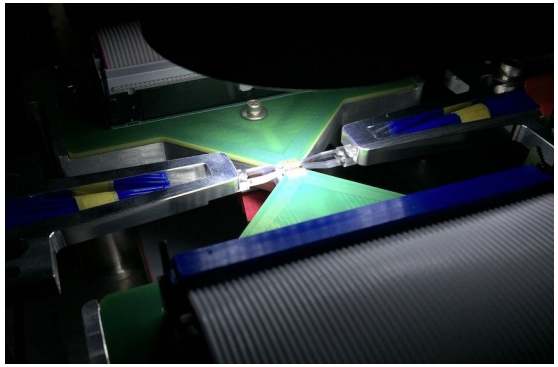


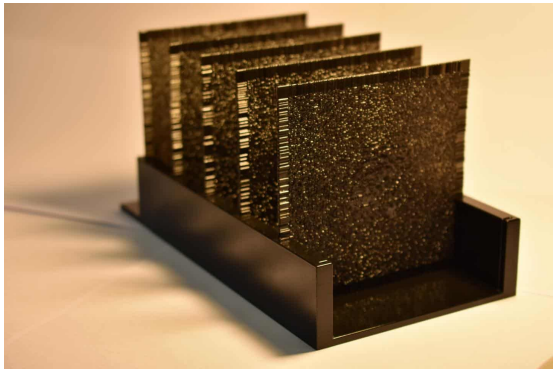
DSPs

TPUs

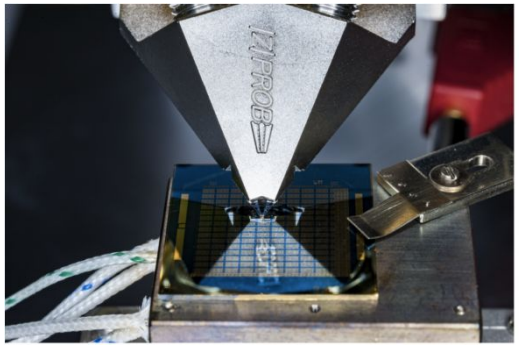


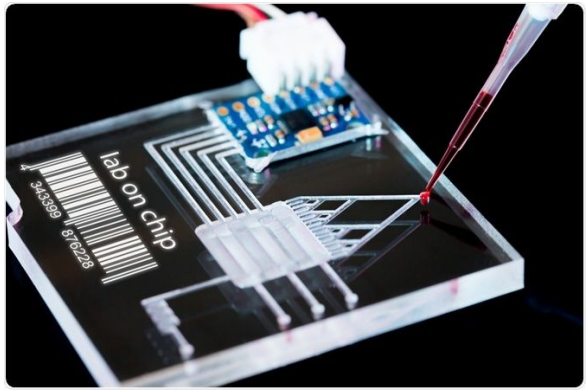
GPUs

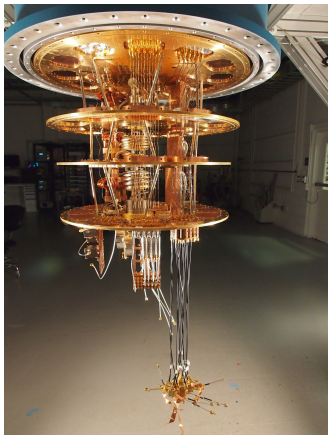


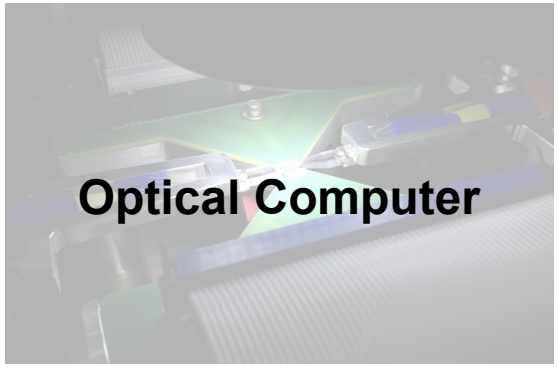


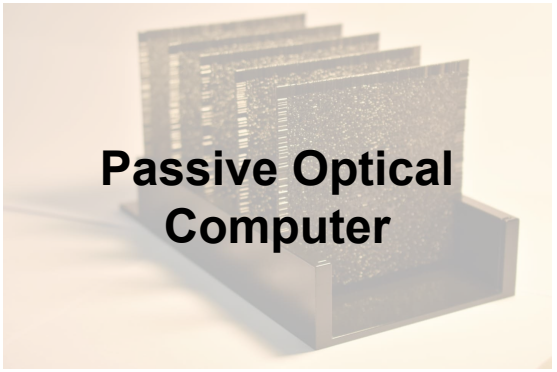
Many exciting special-purpose hardware platforms on the horizon



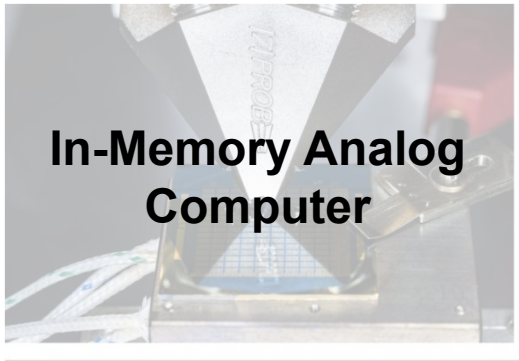


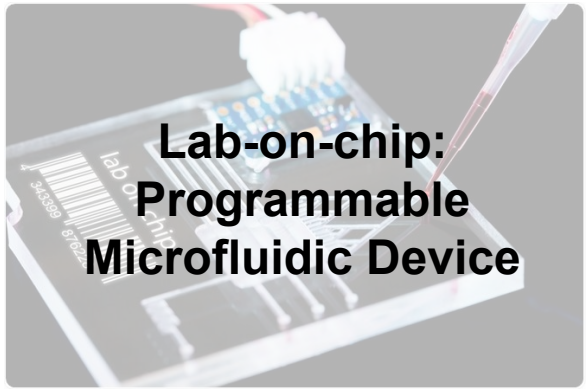


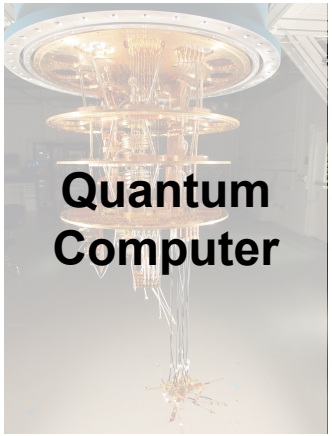


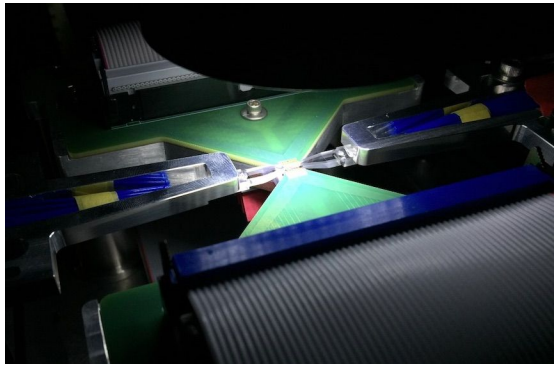


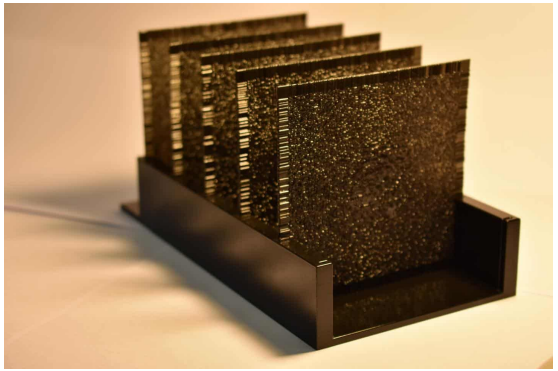
Many exciting special-purpose hardware platforms on the horizon



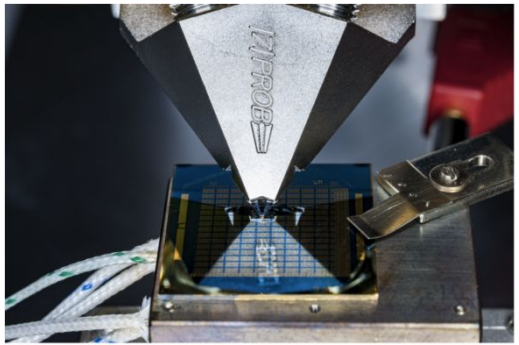


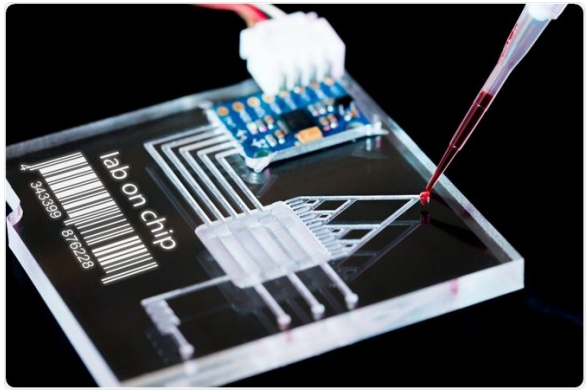


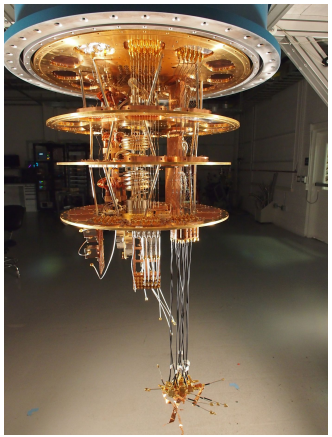


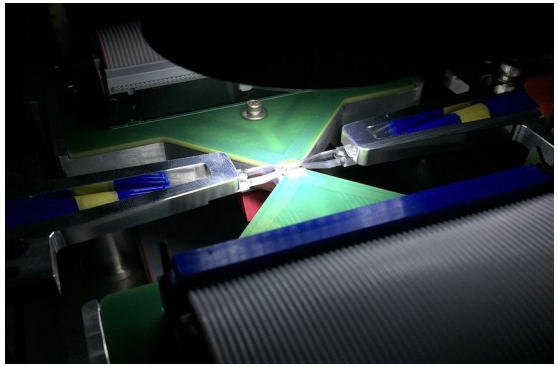


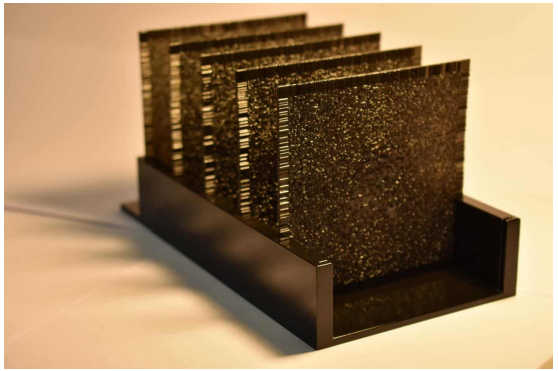
Common Theme: Analog Behavior!



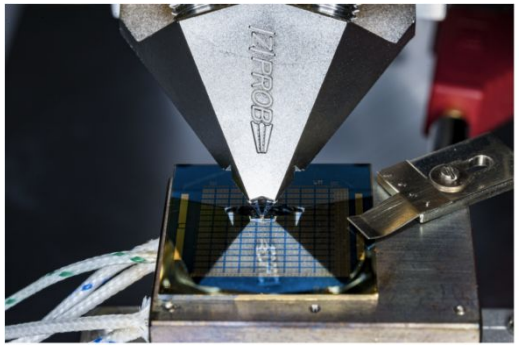


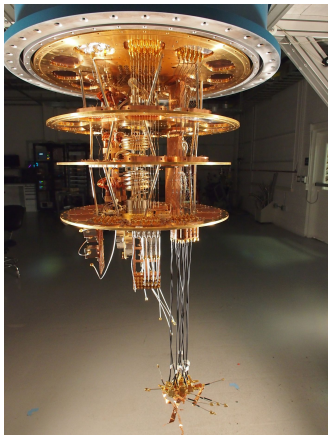


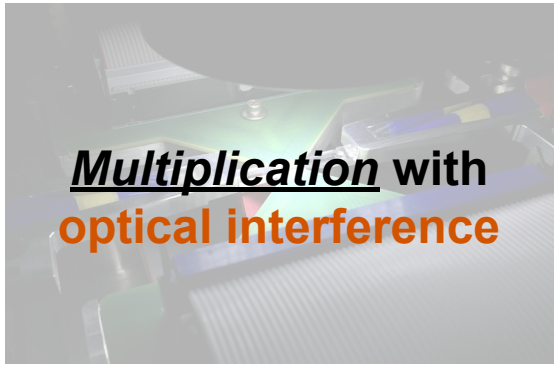


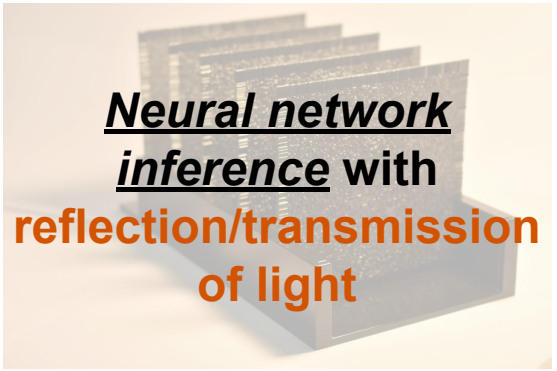


Some devices leverage analog behavior to perform computation

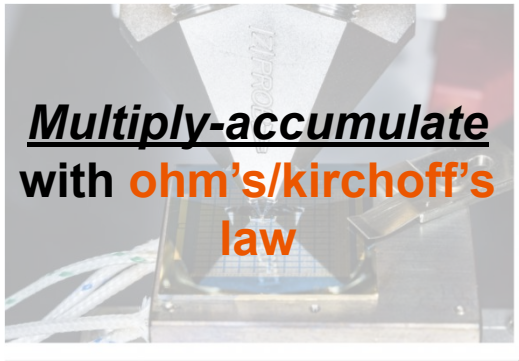


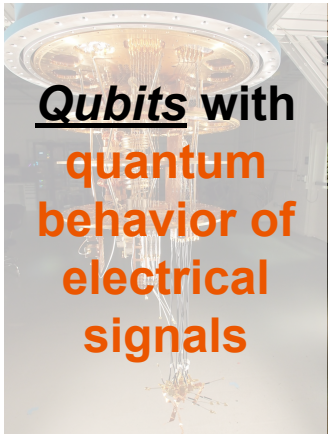




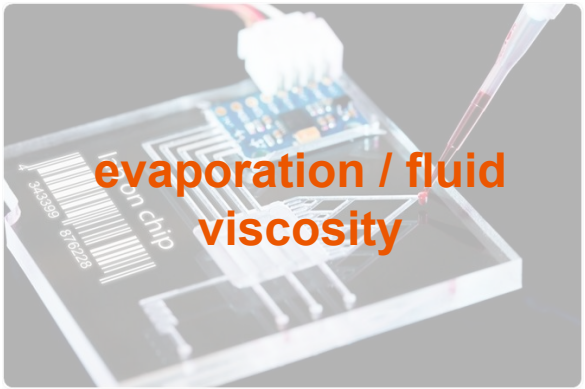


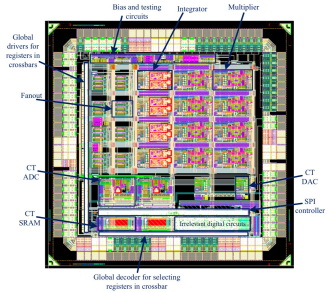
Some devices leverage analog behavior to efficiently perform computation



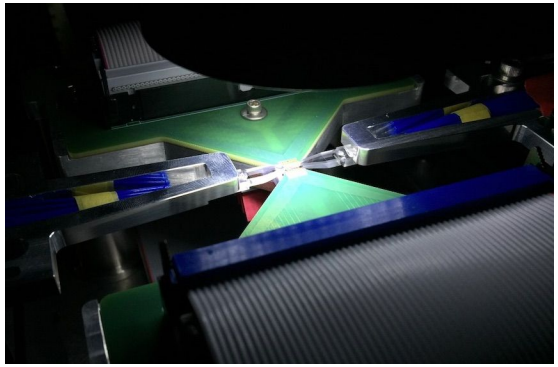


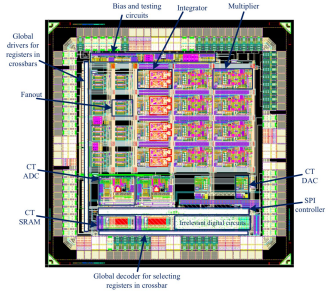
Some computational models/technologies are adversely affected by analog behavior

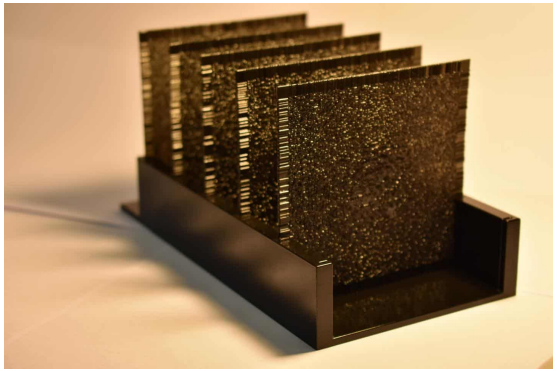




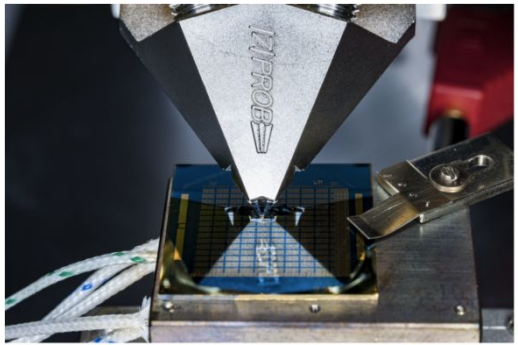
Some devices both leverage and are adversely affected by analog behavior

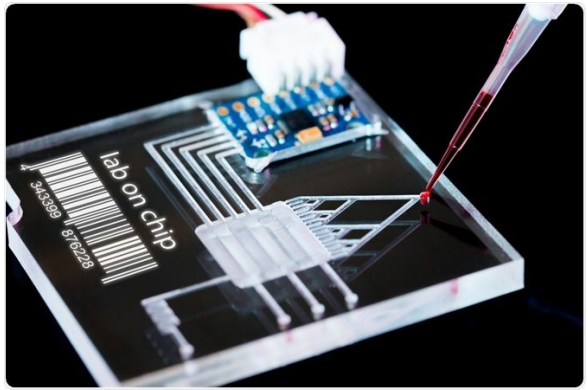


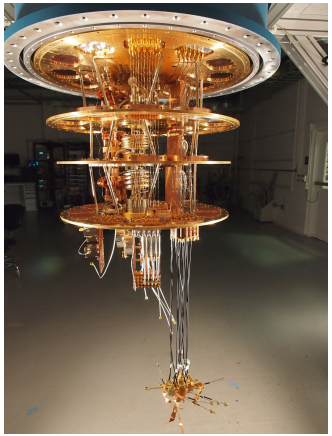


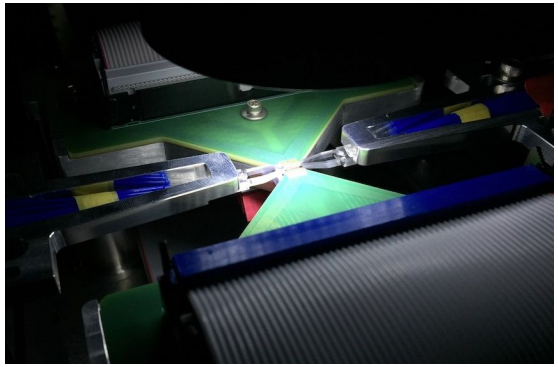


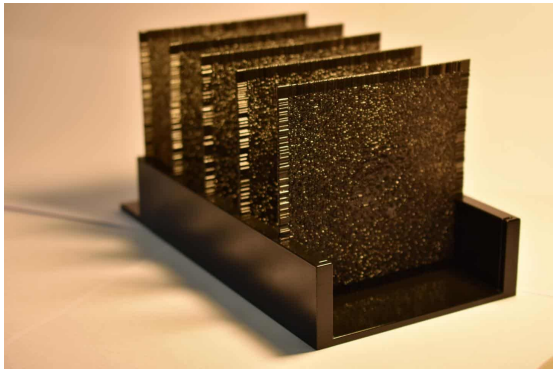
Some devices both leverage and are adversely affected by analog behavior

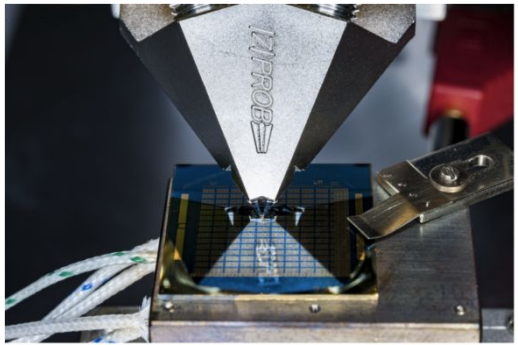


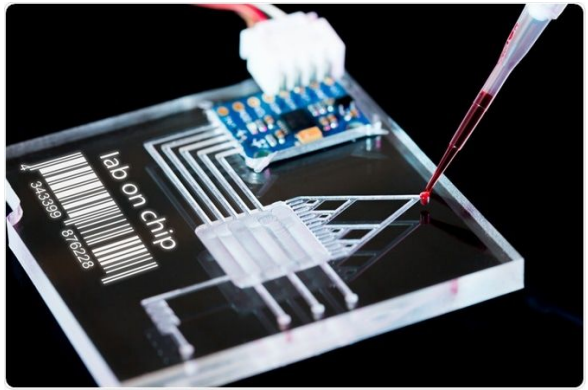


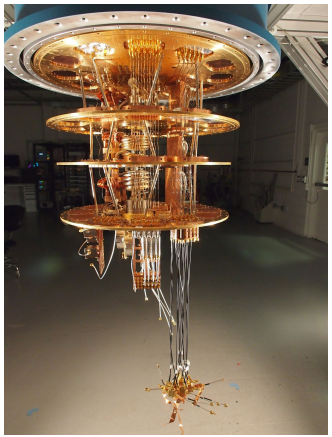









lab on chip



Contributions

- New compilation techniques for reconfigurable analog devices
  - Circuit synthesis techniques
  - Automated scaling techniques
- First-ever compilation toolchain for a real-world analog device of this class
- Experimental results evaluating effectiveness of compilation toolchain

Backup

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, \ldots, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

Operating Range Constraints

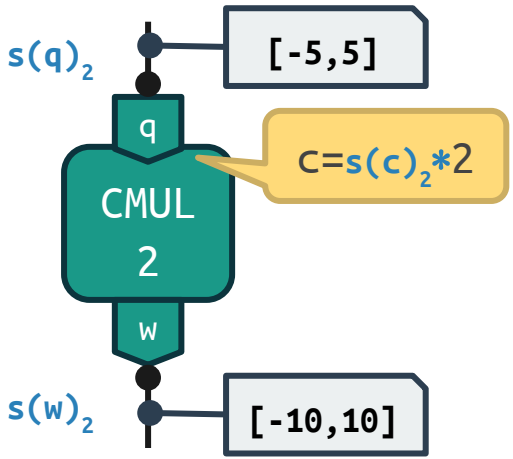
Execution Speed Constraints

Quality Constraints

Connection Constraints

Factor Constraints

Operating Range Constraints



$$s(q)_2 * [-5,5] \subseteq [-2,2]\ \mu A$$

$$s(w)_2 * [-10,10] \subseteq [-2,2]\ \mu A$$

$$s(c)_2 * 2 \subseteq [-1,1]$$

$$q,w \in [-2,2]\ \mu A$$

$$c \in [-1,1]$$

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

- Operating Range Constraints
- Execution Speed Constraints
- **Quality Constraints**
- Connection Constraints
- Factor Constraints

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

Operating Range Constraints

Execution Speed Constraints

Quality Constraints

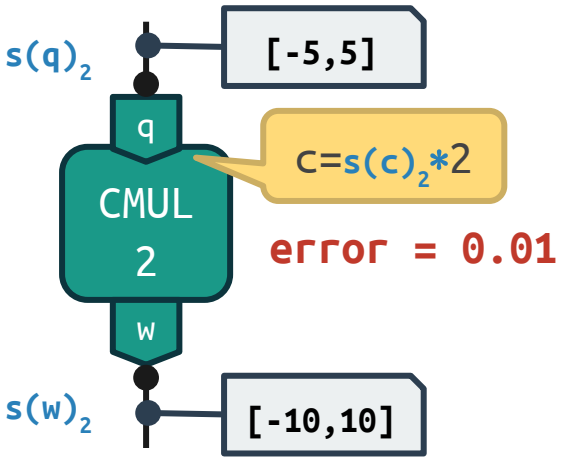
Connection Constraints

Factor Constraints

minimum analog and digital SNR

Digital SNR

Analog SNR



w ~ Normal(w,0.02 μA)

Quality Constraints

$s(w)_2 * \|[-10,10]\| \ (0.02\ \mu A)^{-1} \geq$ Analog SNR

$s(c)_2 * 2 \ (0.01)^{-1} \geq$ Digital SNR

Automatic Scaling

Convex Optimization

Formulated as geometric programming problem.

Variables: $\tau, s(x)_2, s(z)_2, s(c)$

Objective: Maximize execution speed

Minimize τ

Subject to:

Operating Range Constraints

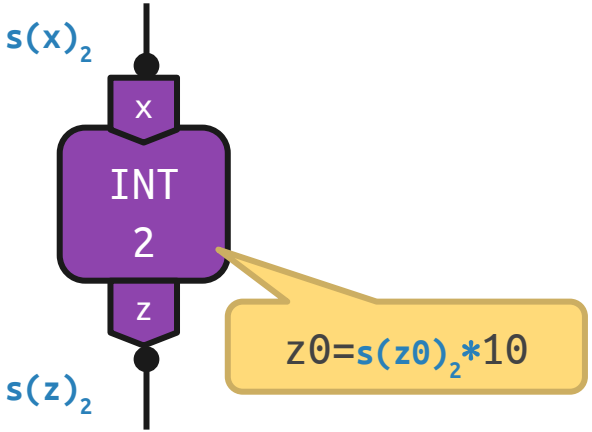
Execution Speed Constraints

Quality Constraints

Connection Constraints

Factor Constraints

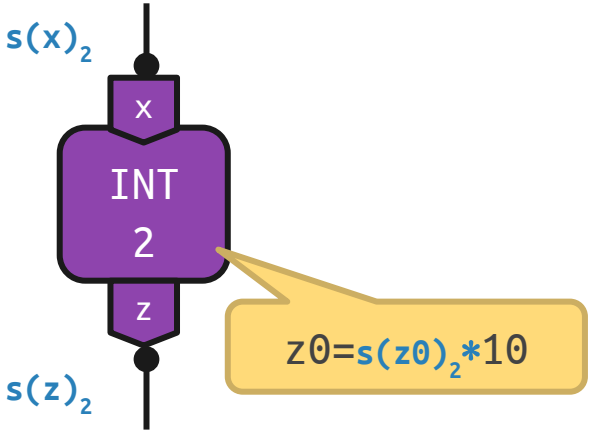
Factor Constraints



$$s(z)_2 * z = \int 0.93 * 0.5 * (s(x)_2 * x)\ \tau^{-1}\ dt_{hw}$$

$$s(z)_2 * z(0) = 0.77 * s(z0)_2 * 10$$

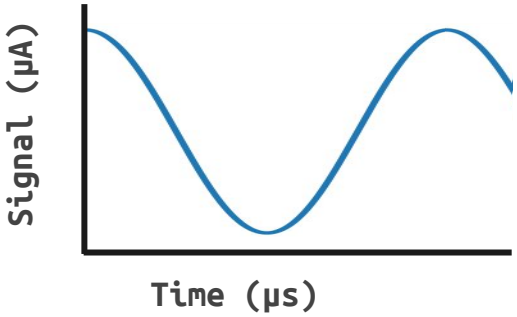
Factor Constraints



$$s(z)_2 * z = \int 0.93 * 0.5 * (s(x)_2 * x) \; \tau^{-1} \; dt_{hw}$$

$$s(z)_2 * z(0) = 0.77 * s(z0)_2 * 10$$

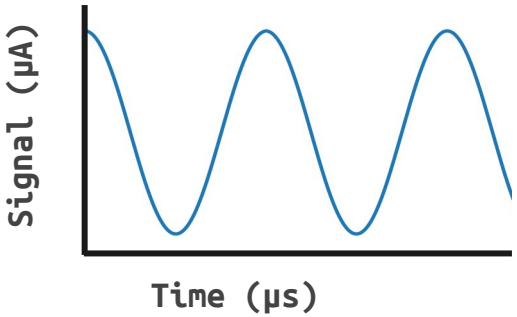
Time Scaling Differential Equations



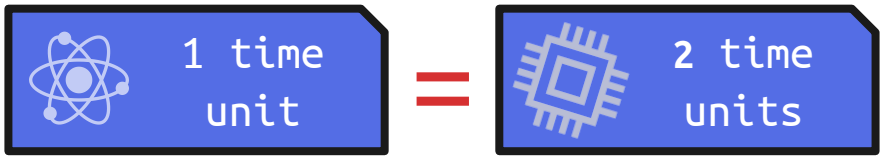
$\tau = 2.0$

$$p = \int v \, \tau^{-1} * dt_{hw}$$

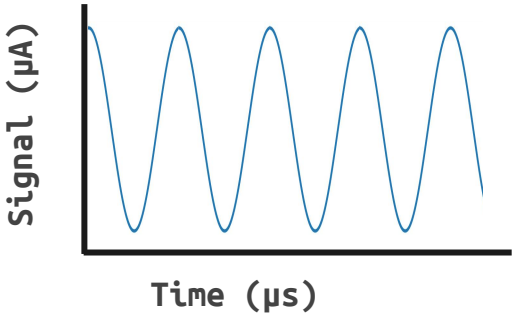
$$v = \int -0.25 * p \, \tau^{-1} * dt_{hw}$$



$\tau = 1.0$



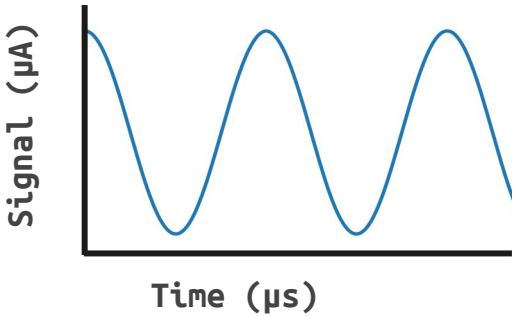
Time Scaling Differential Equations



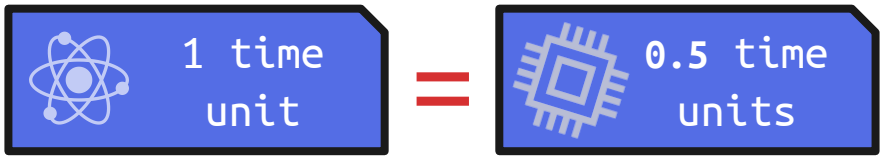
$\tau = 0.5$

$$p = \int v \, \tau^{-1} * dt_{hw}$$

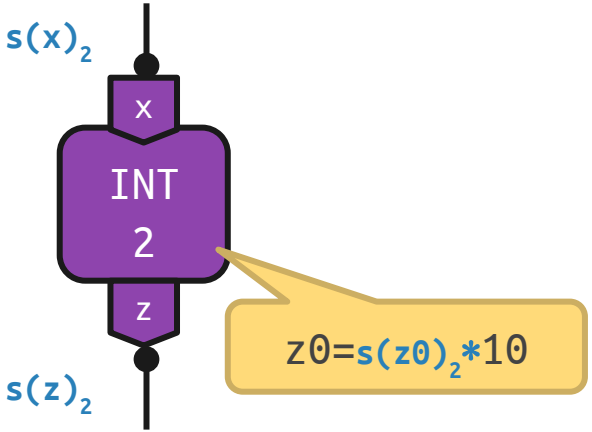
$$v = \int -0.25 * p \, \tau^{-1} * dt_{hw}$$



$\tau = 1.0$



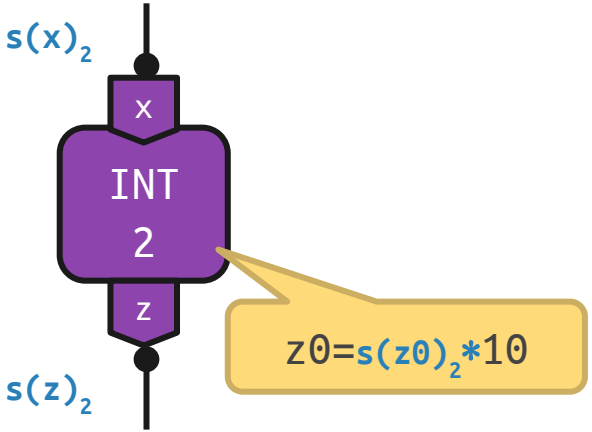
Factor Constraints



$$s(z)_2 * z = \int 0.93 * 0.5 * (s(x)_2 * x)\ \tau^{-1}\ dt_{hw}$$

$$s(z)_2 * z(0) = 0.77 * s(z0)_2 * 10$$

Factor Constraints

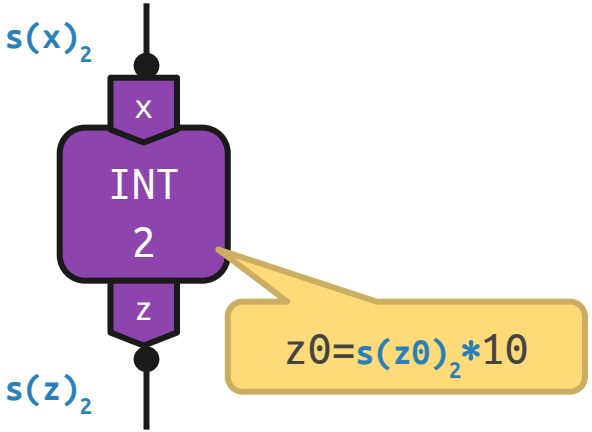


$$s(z)_2 * z = \int 0.93 * 0.5 * (s(x)_2 * x)\ \tau^{-1}\ dt_{hw}$$

$$s(z)_2 * z(0) = 0.77 * s(z0)_2 * 10$$

Goal: Factor out and eliminate **scaling factors** and **manufacturing variations** from **dynamics (black)**

Factor Constraints

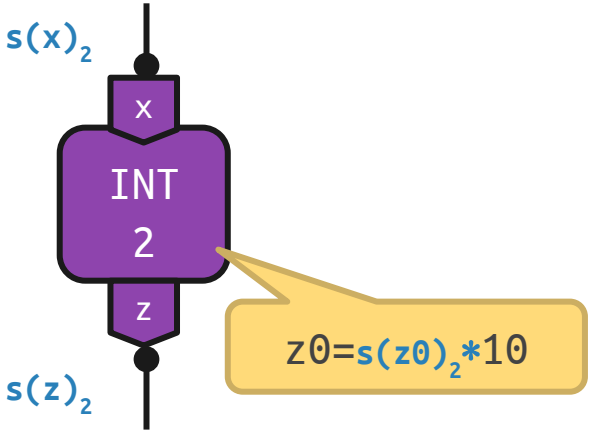


$$s(z)_2 * z = (0.93 * s(x)_2 * \tau^{-1}) \int 0.5 * (x)\, dt_{hw}$$

$$s(z)_2 * z(0) = (0.77 * s(z0)_2)\ 10$$

Goal: Factor out and eliminate **scaling factors** and **manufacturing variations** from **dynamics (black)**

Factor Constraints



$z = \int 0.5*(x)\ dt_{hw}$

$s(z)_2 = 0.93*s(x)_2*\tau^{-1}$

$z(0) = 10$

$s(z)_2 = 0.77*s(z0)_2$

Goal: Factor out and eliminate **scaling factors** and **manufacturing variations** from **dynamics (black)**

Unscaled

Algebraically Equivalent to

Unscaled

Algebraically Equivalent to

Convex Optimization Problem

Minimize τ

Subject to:

- Operating Range Constraints
- Execution Speed Constraints
- Quality Constraints
- Connection Constraints
- Factor Constraints

Unscaled

Algebraically Equivalent to

Convex Optimization Problem

Minimize τ

Subject to:

- Operating Range Constraints
- Execution Speed Constraints
- Quality Constraints
- Connection Constraints
- Factor Constraints

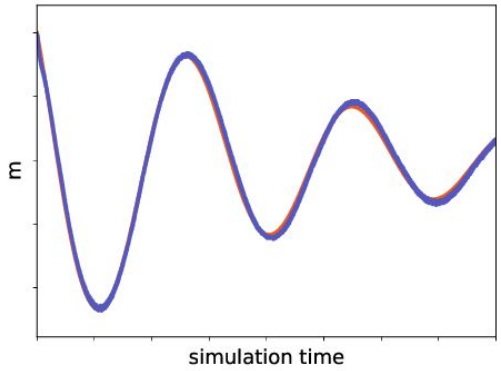
Solved using Convex Solver

What can go wrong?

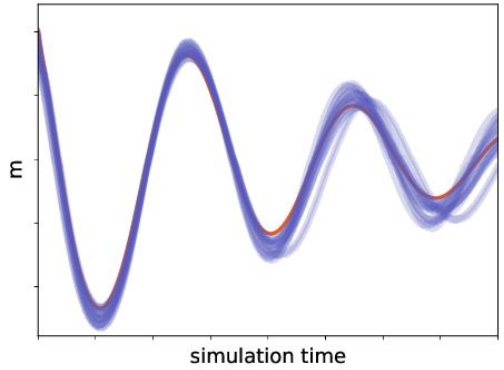
Challenges

Not all generated configurations deliver good results

Best Pendulum Execution



All Pendulum Executions



Challenges

Not all generated configurations deliver good results

Some configurations are actually better than others

Design method for ranking device configurations

Challenges

Not all generated configurations deliver good results

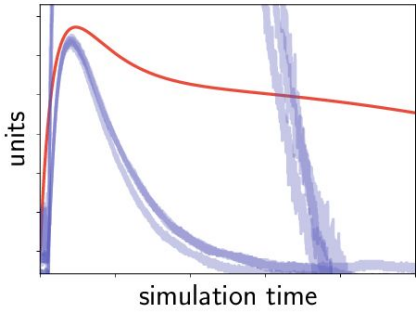
Not all manufactured block behaviors captured by compiler

Profile and statically compensate for other block behaviors.

Challenges

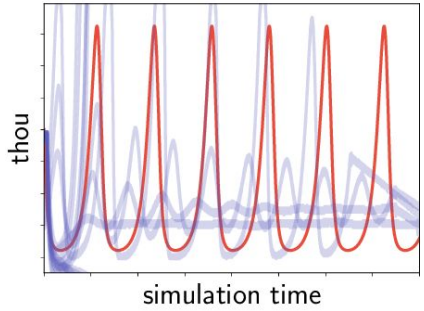
Not all dynamical systems can be accurately executed on device

Dynamic Range too High



Erythropoietin receptor model

System is Sensitive to Error



Predator-Prey simulation

Challenges

Not all dynamical systems can be accurately executed on device

Automatically characterize dynamical systems

Challenges

Not all dynamical systems can be accurately executed on device

Automatically characterize dynamical systems

How much error can the system tolerate?

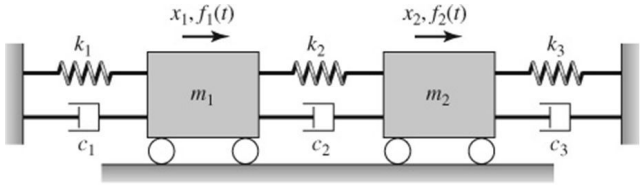
Challenges

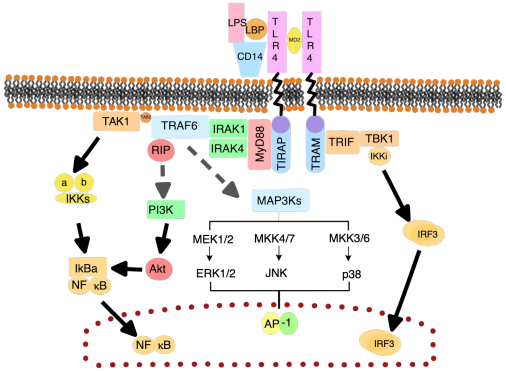
Not all dynamical systems can be accurately executed on device

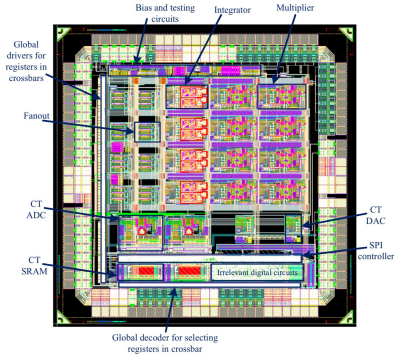
Automatically characterize dynamical systems.

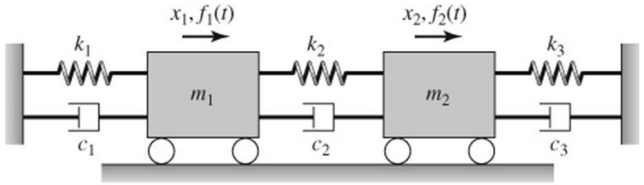
How much error can the system tolerate?

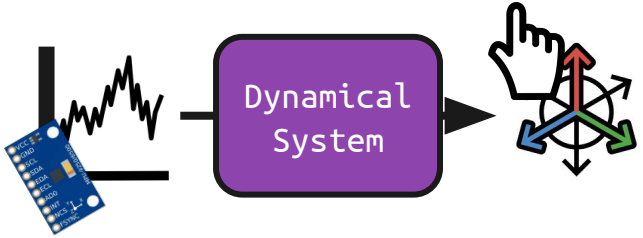
Where in the computation does it tolerate error?

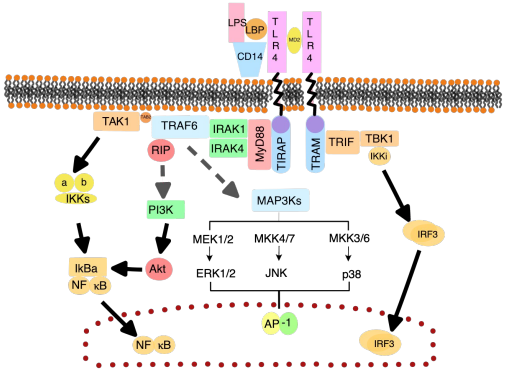

$x_1, f_1(t)$
$x_2, f_2(t)$
k_1
k_2
k_3
m_1
m_2
c_1
c_2
c_3

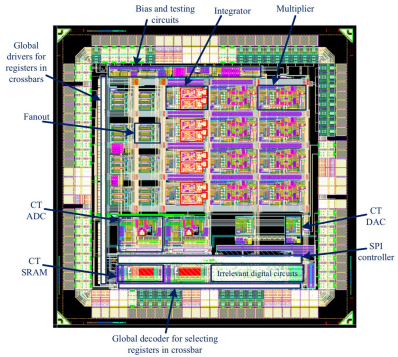

LPS
LBP
CD14
TLR4
MD2
TLR4
TAK1
TRAF6
IRAK1
IRAK4
MyD88
TIRAP
TRAM
TRIF
TBK1
IKKi
RIP
a
b
IKKs
PI3K
MAP3Ks
MEK1/2
MKK4/7
MKK3/6
ERK1/2
JNK
p38
IRF3
IkBa
NF κB
Akt
AP-1
NF κB
IRF3

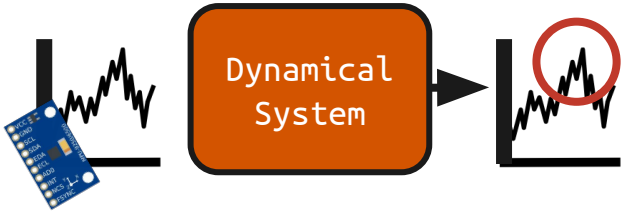

Bias and testing circuits
Integrator
Multiplier
Global drivers for registers in crossbars
Fanout
CT ADC
CT DAC
SPI controller
CT SRAM
Irrelevant digital circuits
Global decoder for selecting registers in crossbar


$x_1, f_1(t)$
$x_2, f_2(t)$
k_1
k_2
k_3
m_1
m_2
c_1
c_2
c_3


Dynamical System

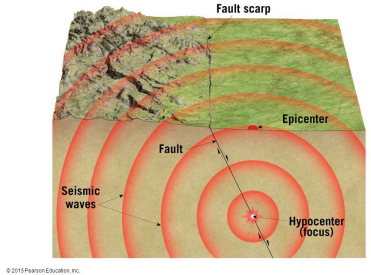

LPS
LBP
CD14
TLR4
MD2
TLR4
TAK1
TRAF6
IRAK1
IRAK4
MyD88
TIRAP
TRAM
TRIF
TBK1
IKKi
RIP
a
b
IKKs
PI3K
MAP3Ks
MEK1/2
MKK4/7
MKK3/6
ERK1/2
JNK
p38
IRF3
IkBa
NF κB
Akt
AP-1
NF κB
IRF3


Bias and testing circuits
Integrator
Multiplier
Global drivers for registers in crossbars
Fanout
CT ADC
CT SRAM
CT DAC
SPI controller
Irrelevant digital circuits
Global decoder for selecting registers in crossbar

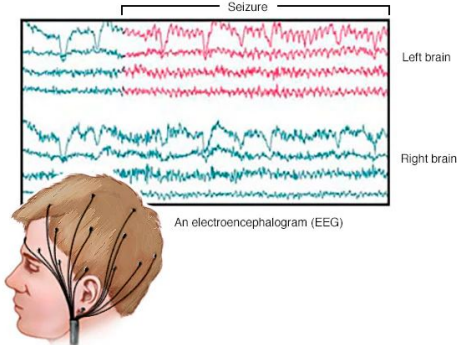

Dynamical System


Dynamical System

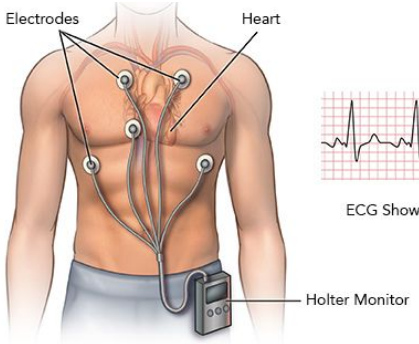
Many interesting signals are slow-evolving



0.2 Hz to 20 Hz



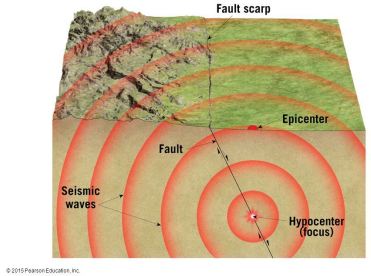
1 Hz–100 Hz



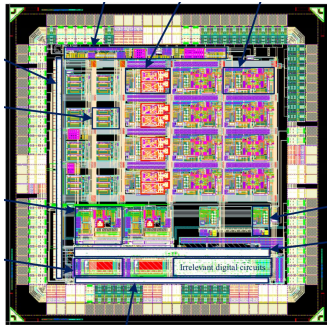
ECG Showing Heart Rhythm

0.1 Hz–1 kHz

Many interesting signals are slow-evolving



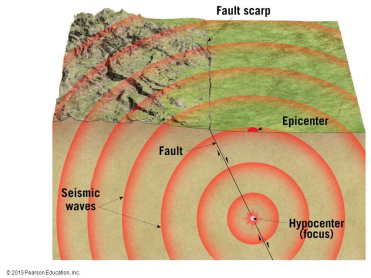
0.2 Hz to 20 Hz

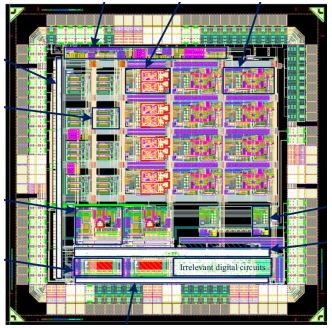


126,000 Hz

Slow signals are difficult to work with in pure analog

Many interesting signals are slow-evolving





0.2 Hz to 20 Hz

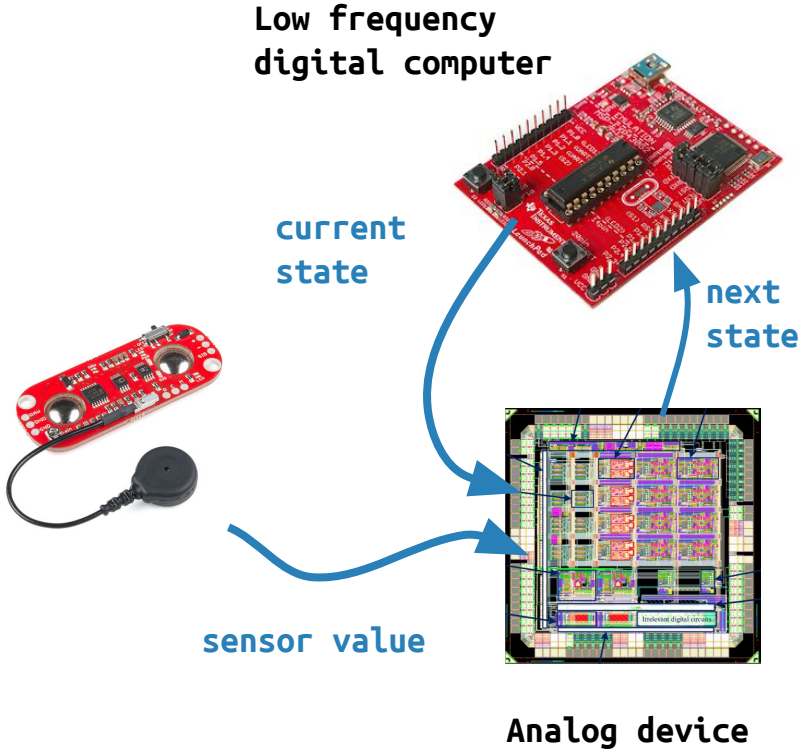
126,000 Hz

Slow signals are difficult to work with in pure analog

Dramatically slow down dynamical system to process signal

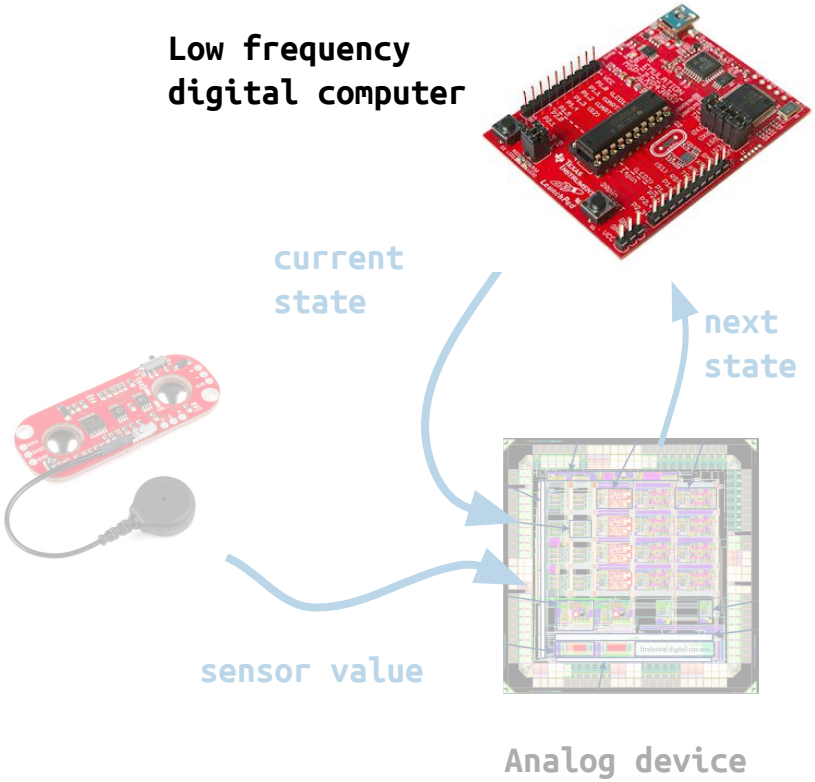
Sensitive to noise and error

Hybrid digital-analog computation



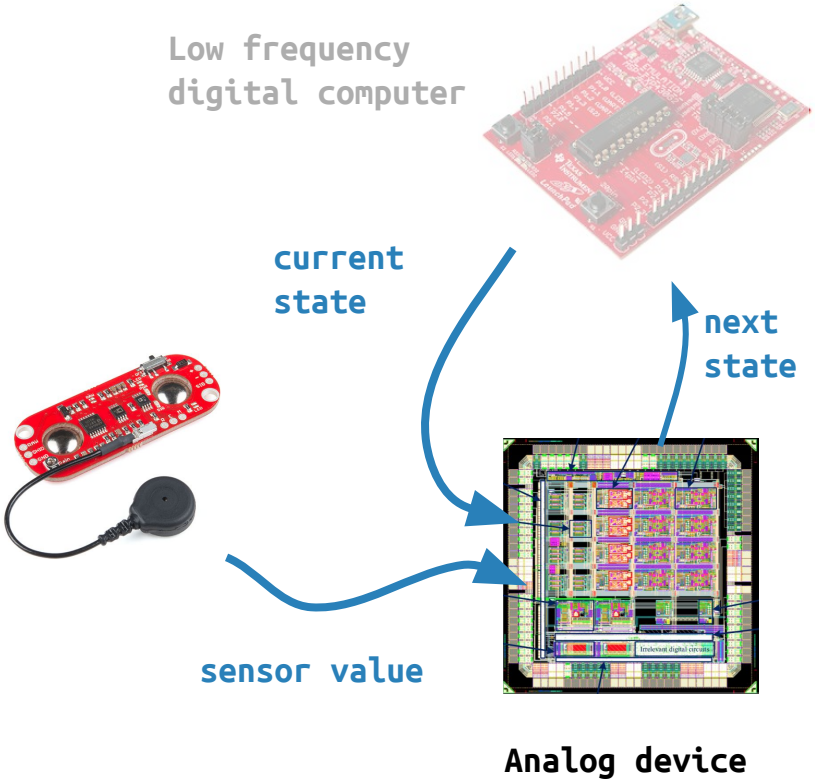
Hybrid digital-analog computation

Digitally store long-running state



Hybrid digital-analog computation

Process external input and update state in analog



Hybrid digital-analog computation