Disciplined Approximate Computing: From Language to Hardware, and Beyond

Luis Ceze University of Washington

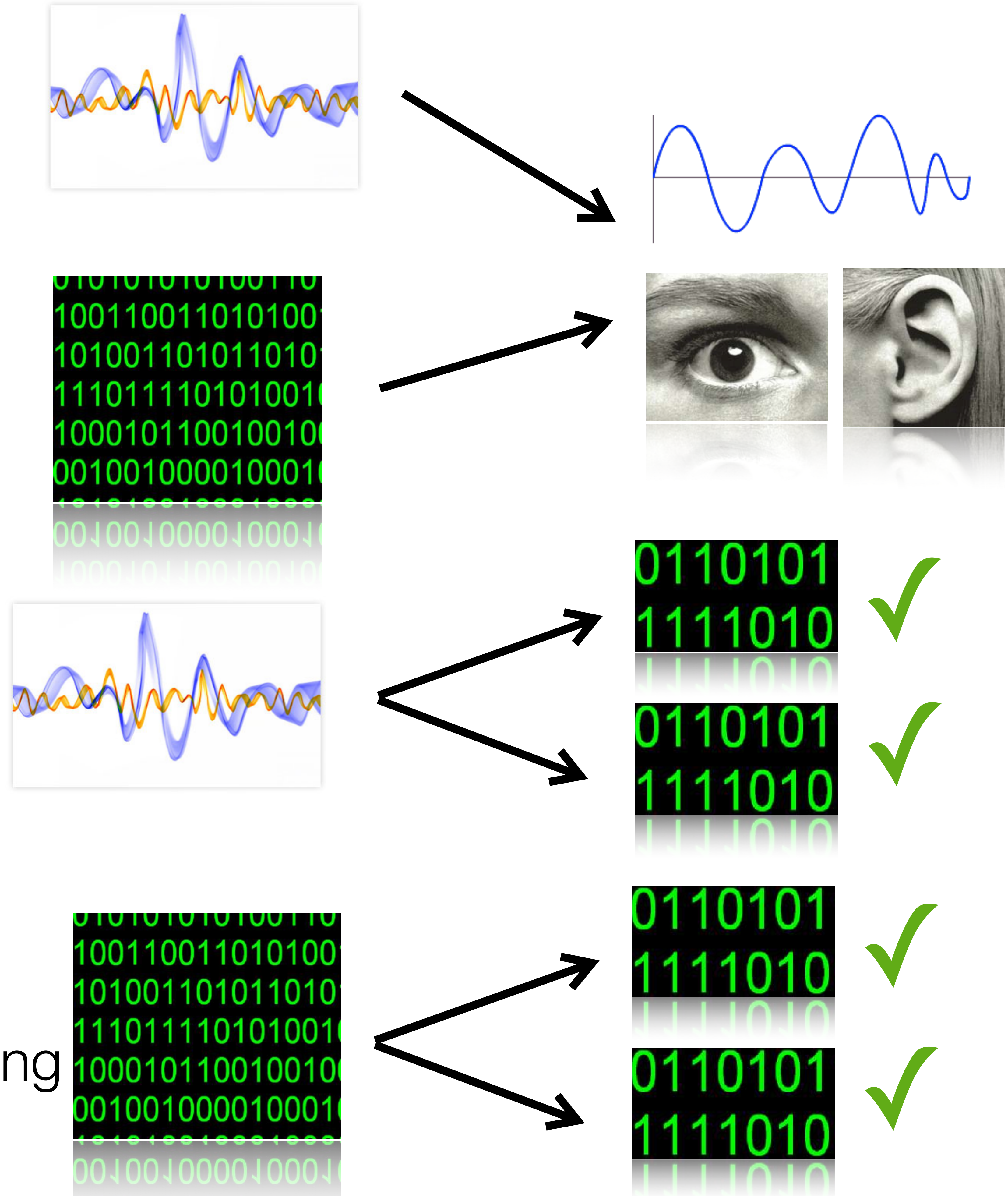
joint work with Adrian Sampson, Hadi Esmaelizadeh, Mike Ringenburg, Renee StAmant, Ben Ransford, Andre Baixo, Thierry Moreau, Dan Grossman, Mark Oskin (UW), Karin Strauss, Doug Burger, Todd Mytkowicz and Kathryn McKinley (Microsoft Research).

image, sound
and video processing

image rendering

sensor data analysis,
computer vision

simulations, games,
search, machine learning



image, sound
and video processing

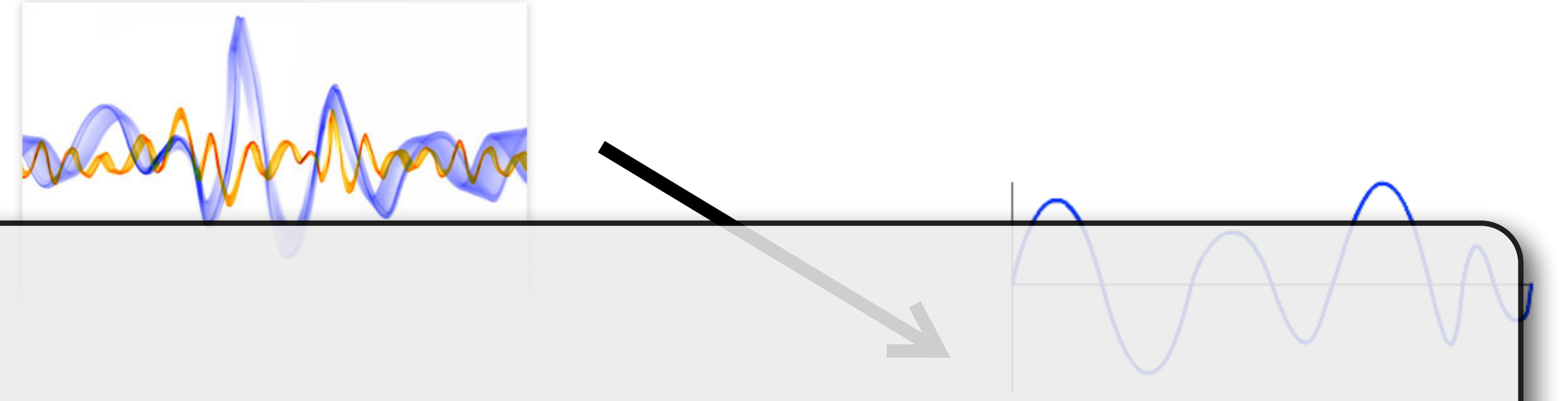


image rendering

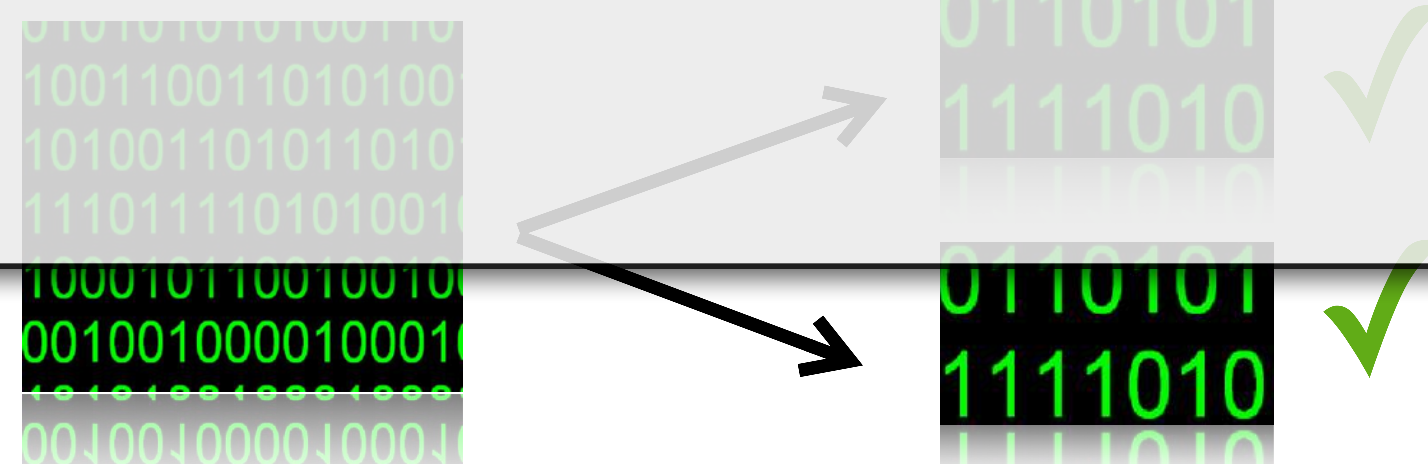
These applications consume a lot (most?)

Often input data is inexact by nature (from sensors)

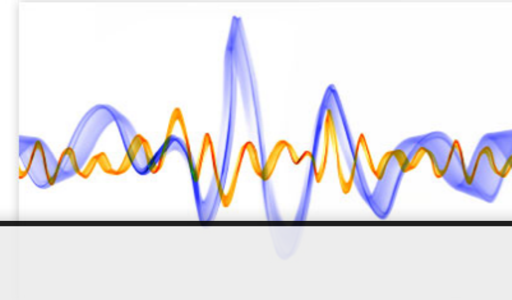
sensor data analysis,
computer vision

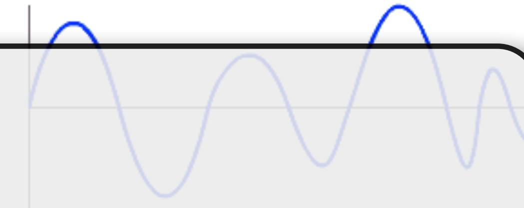
They have multiple acceptable outputs

simulations, games,
search, machine learning



image, sound
and video processing





These applications consume a lot (most?)

image rendering

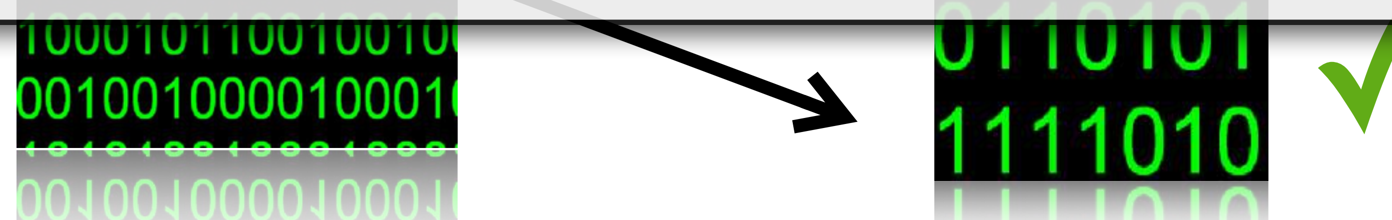
Often input data is inexact by nature (from sensors)

sensor data analysis,
computer vision

They have multiple acceptable outputs

They do not require “perfect execution”

simulations, games,
search, machine learning



Notions of “approximation” have been around for a long time...

Floating point

Lossy compression

Iterative algorithms

...

Notions of “approximation” have been around for a long time...

Floating point

Lossy compression

Iterative algorithms

...

Notions of "approximation" have been around for a long time...

So what is "Approximate computing" then?

Floating point

Lossy compression

Iterative algorithms

...

Notions of “approximation” have been around for a long time…

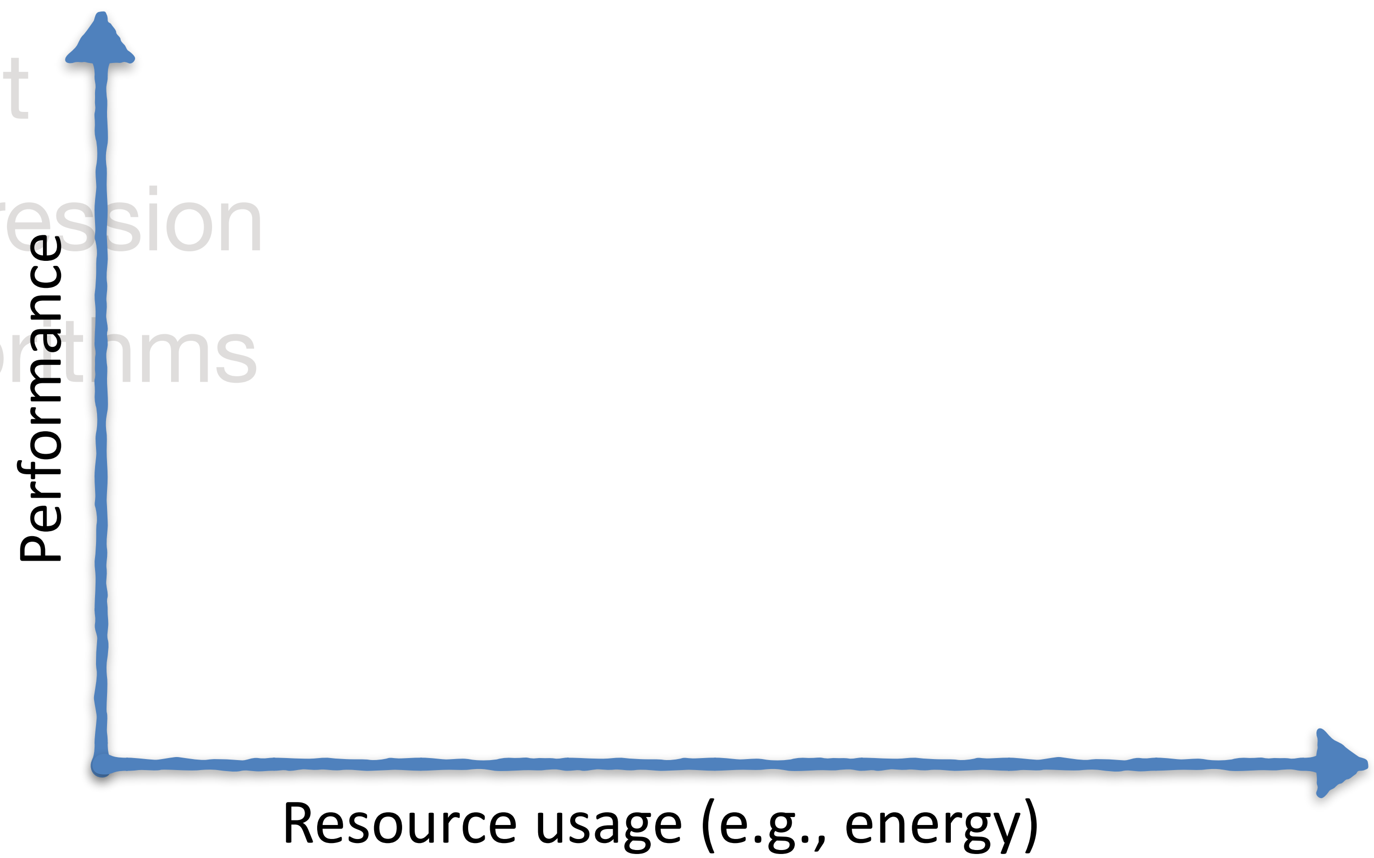
So what is “Approximate computing” then?

Floating point

Lossy compression

Iterative algorithms

…



Notions of "approximation" have been around for a long time…

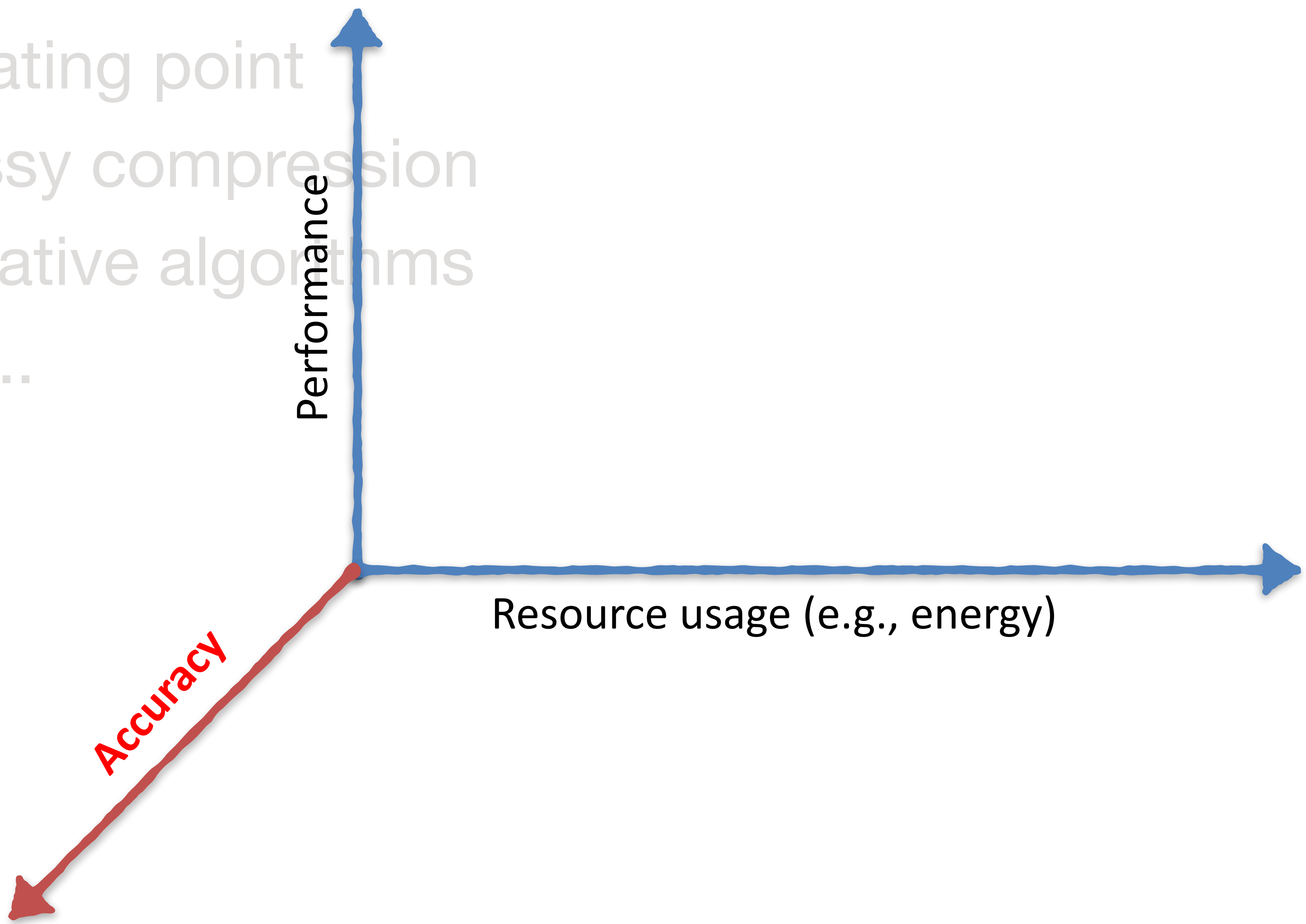
So what is "Approximate computing" then?

Floating point

Lossy compression

Iterative algorithms

…



Notions of "approximation" have been around for a long time…

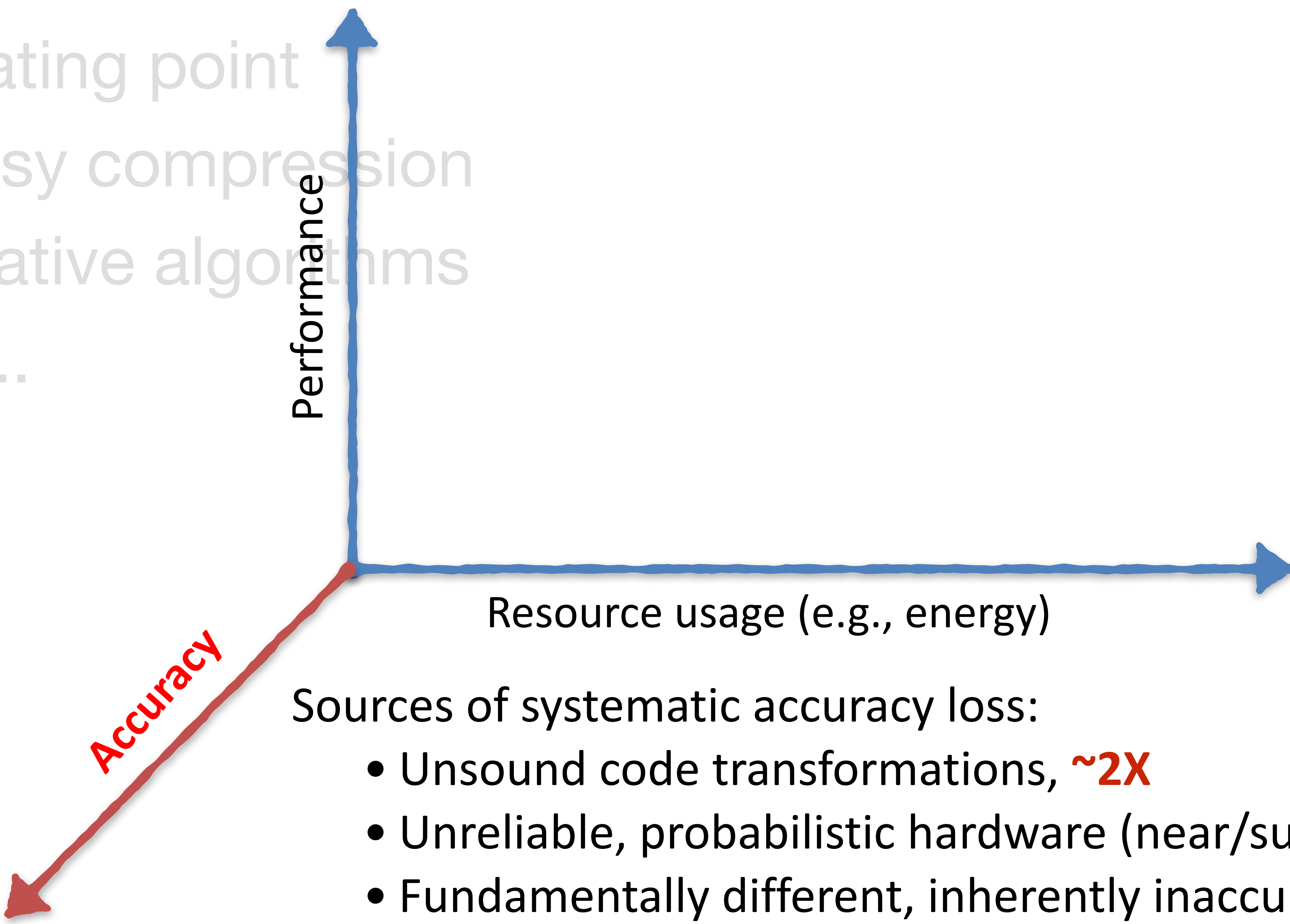
So what is "Approximate computing" then?

Floating point

Lossy compression

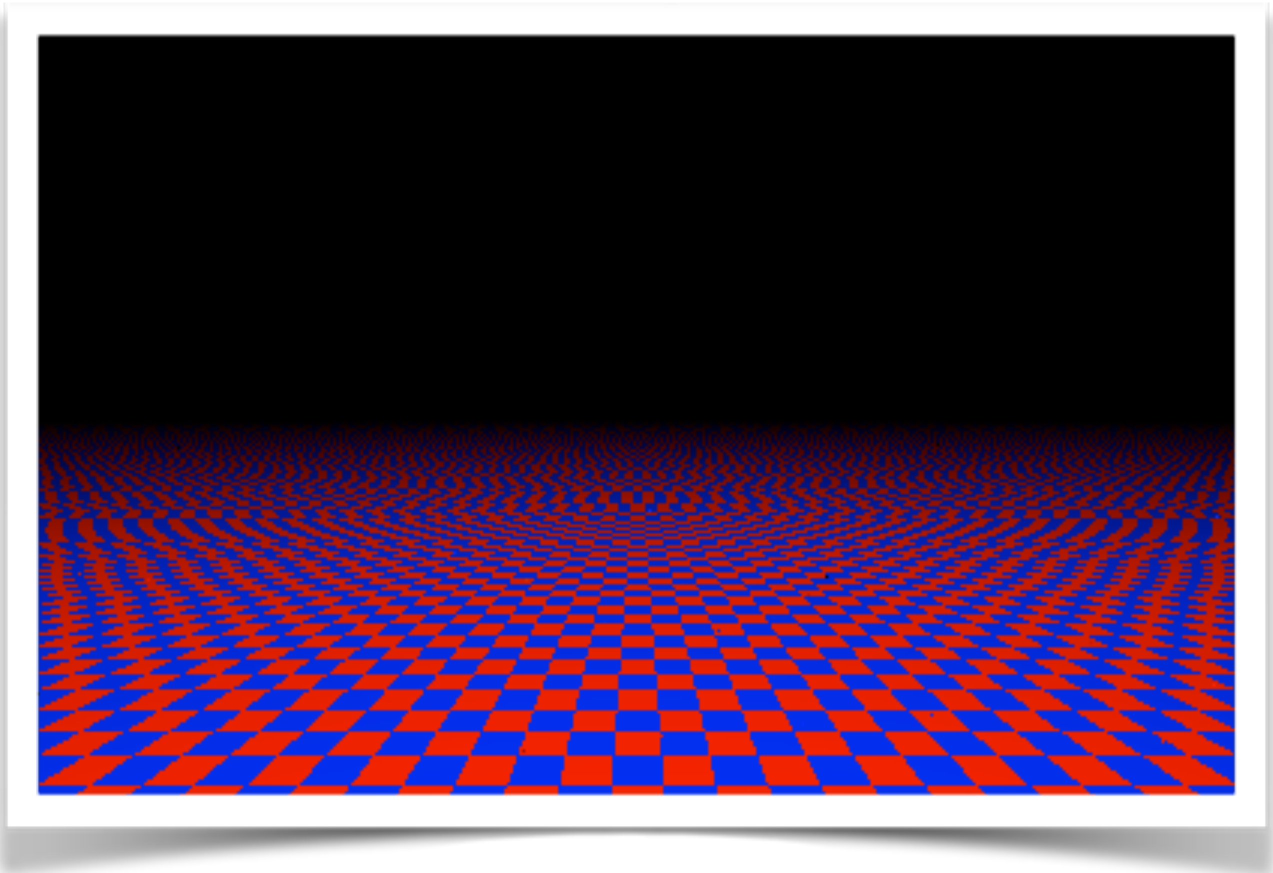
Iterative algorithms

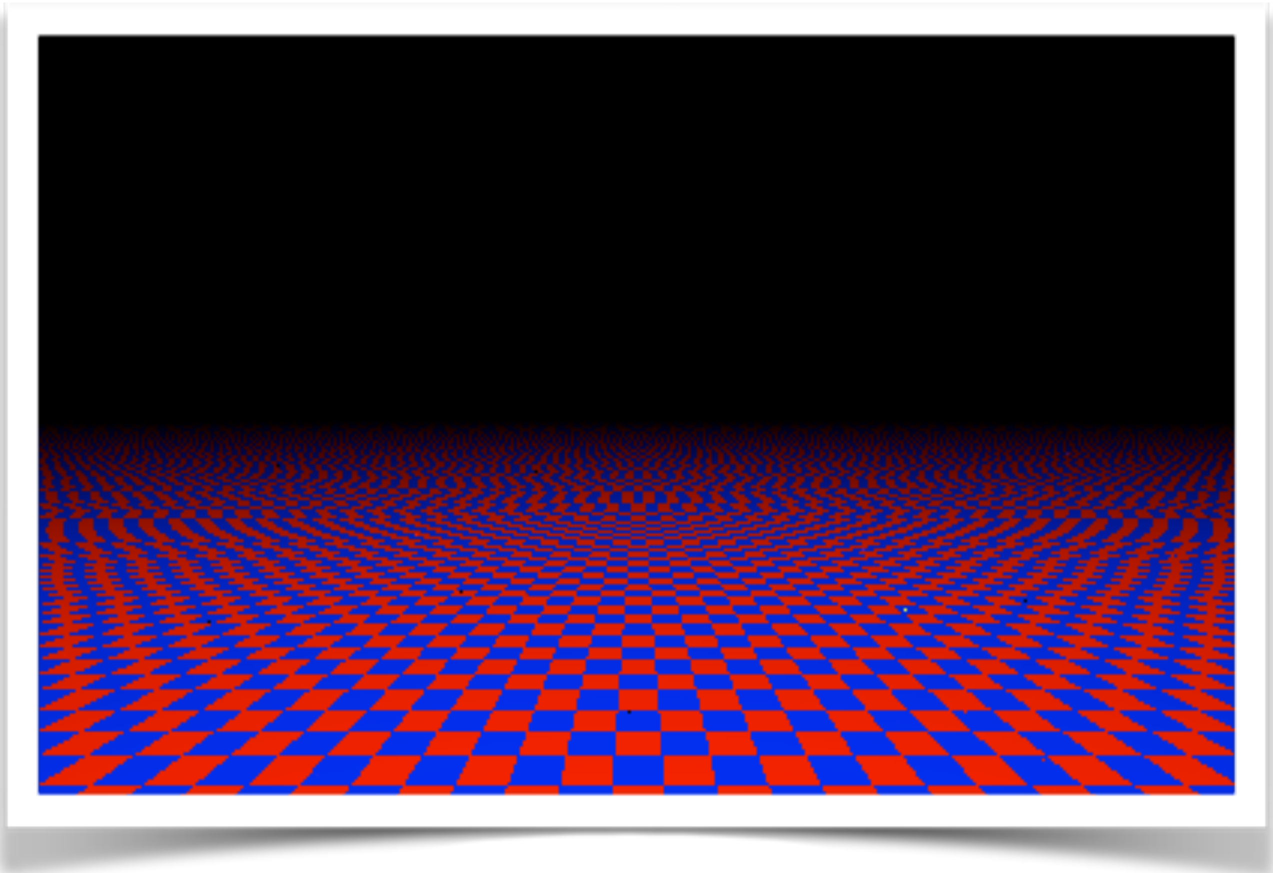
…

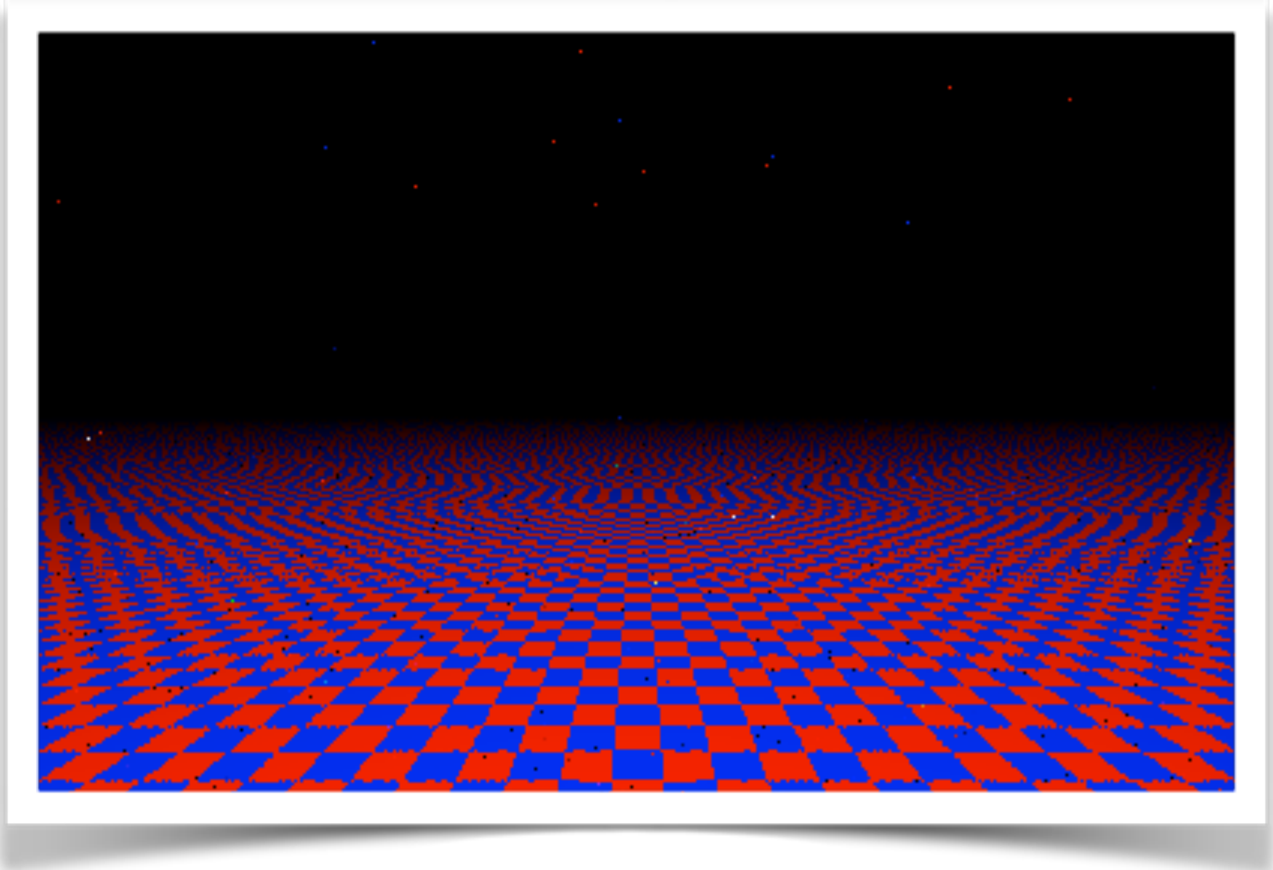


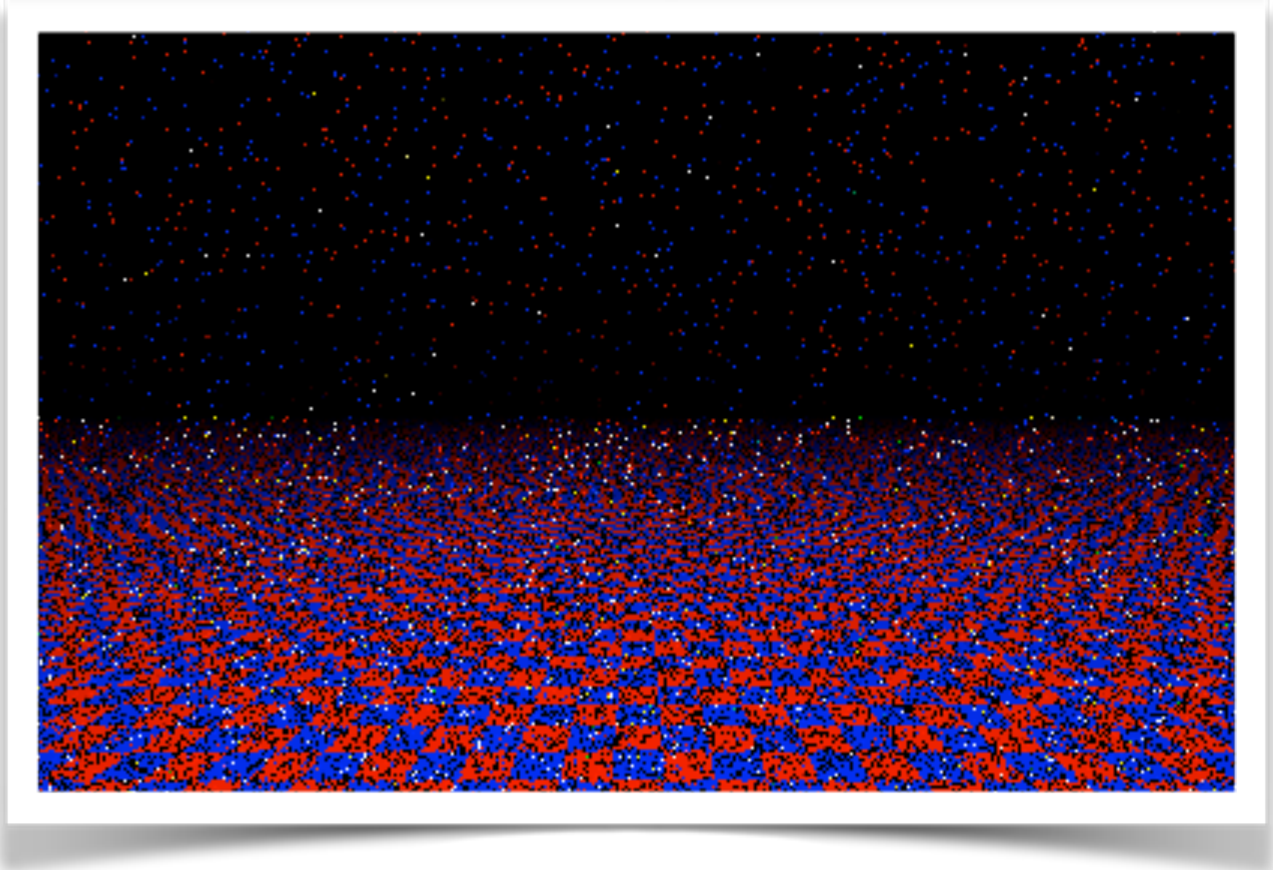
Sources of systematic accuracy loss:

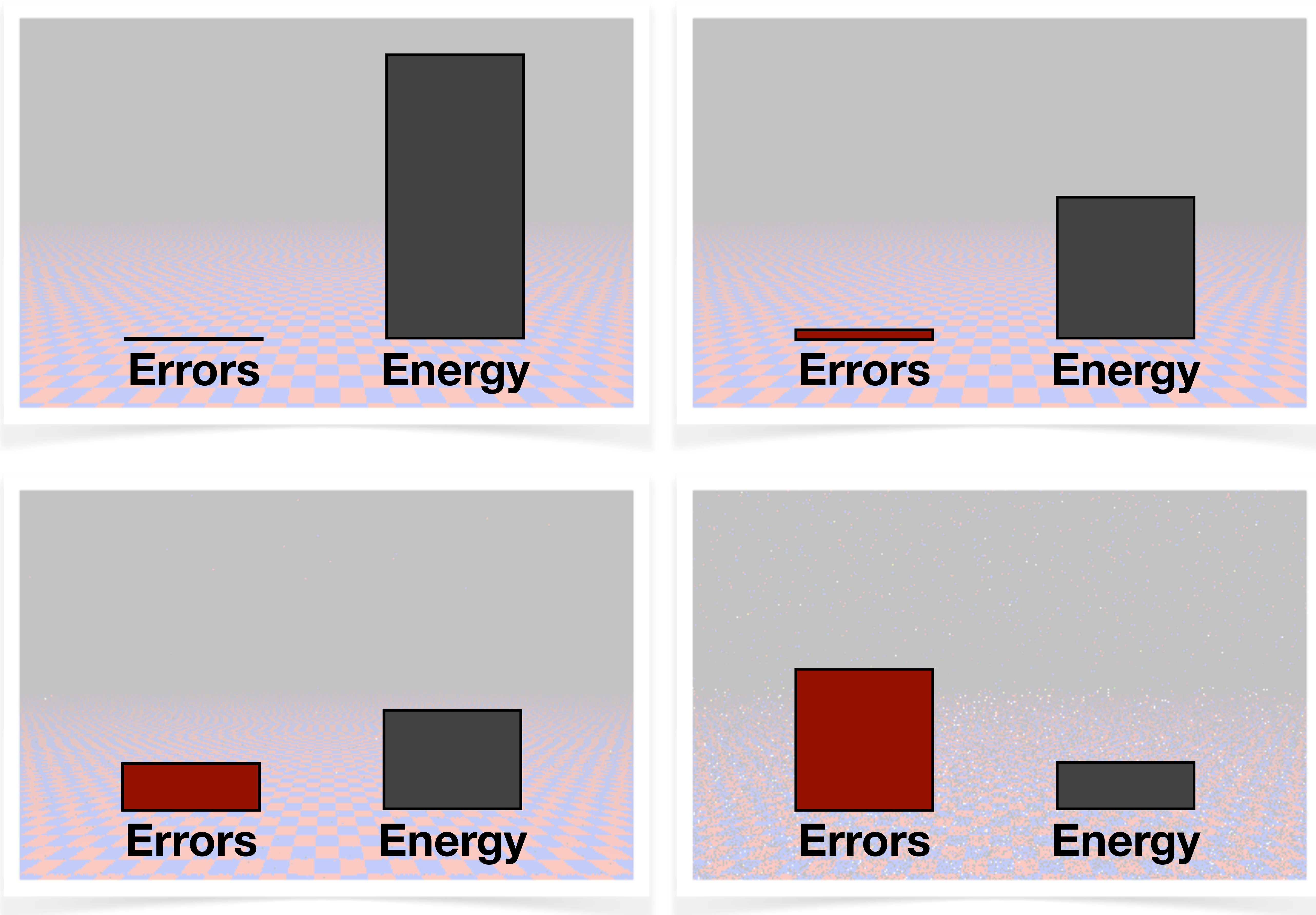
- Unsound code transformations, **~2X**
- Unreliable, probabilistic hardware (near/sub-threshold, etc.), **~5X**
- Fundamentally different, inherently inaccurate execution models, "**closer to physics**" (e.g., neural networks, analog computing), **~10-100X**





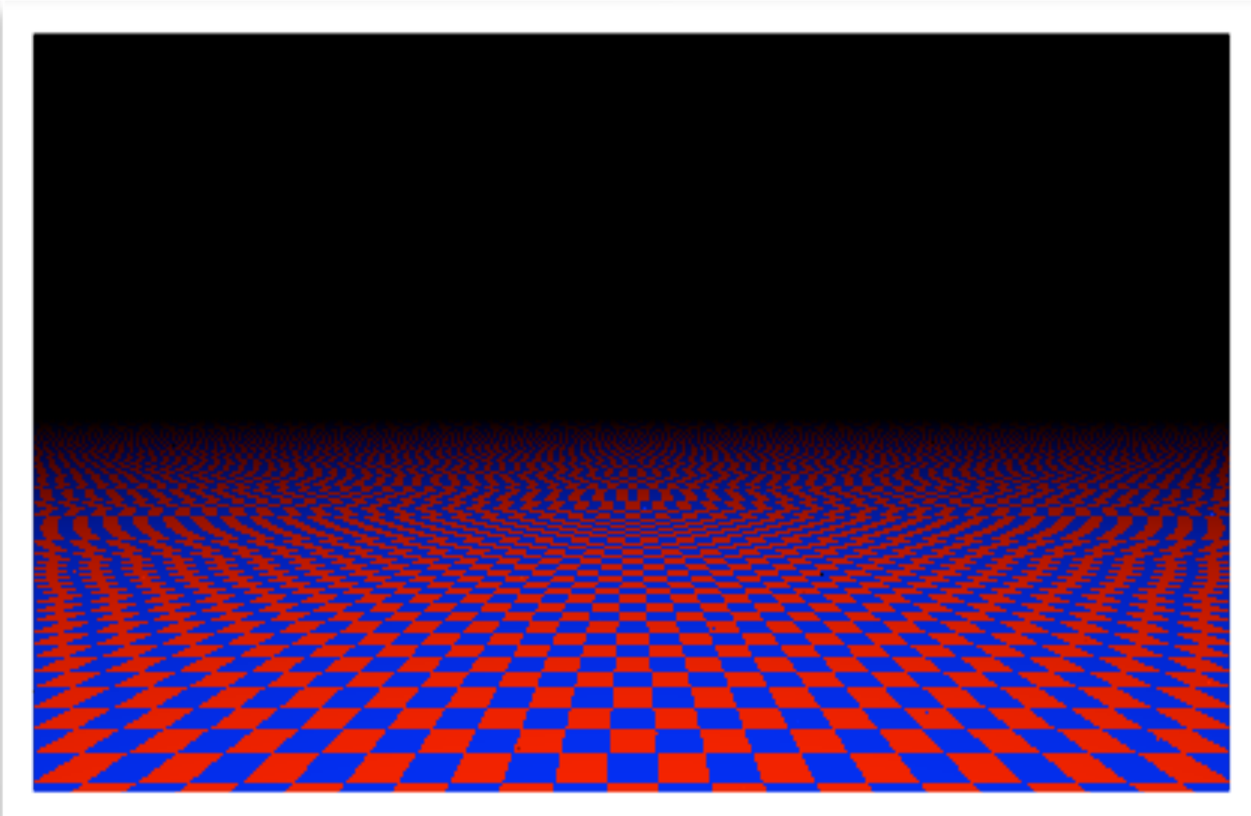


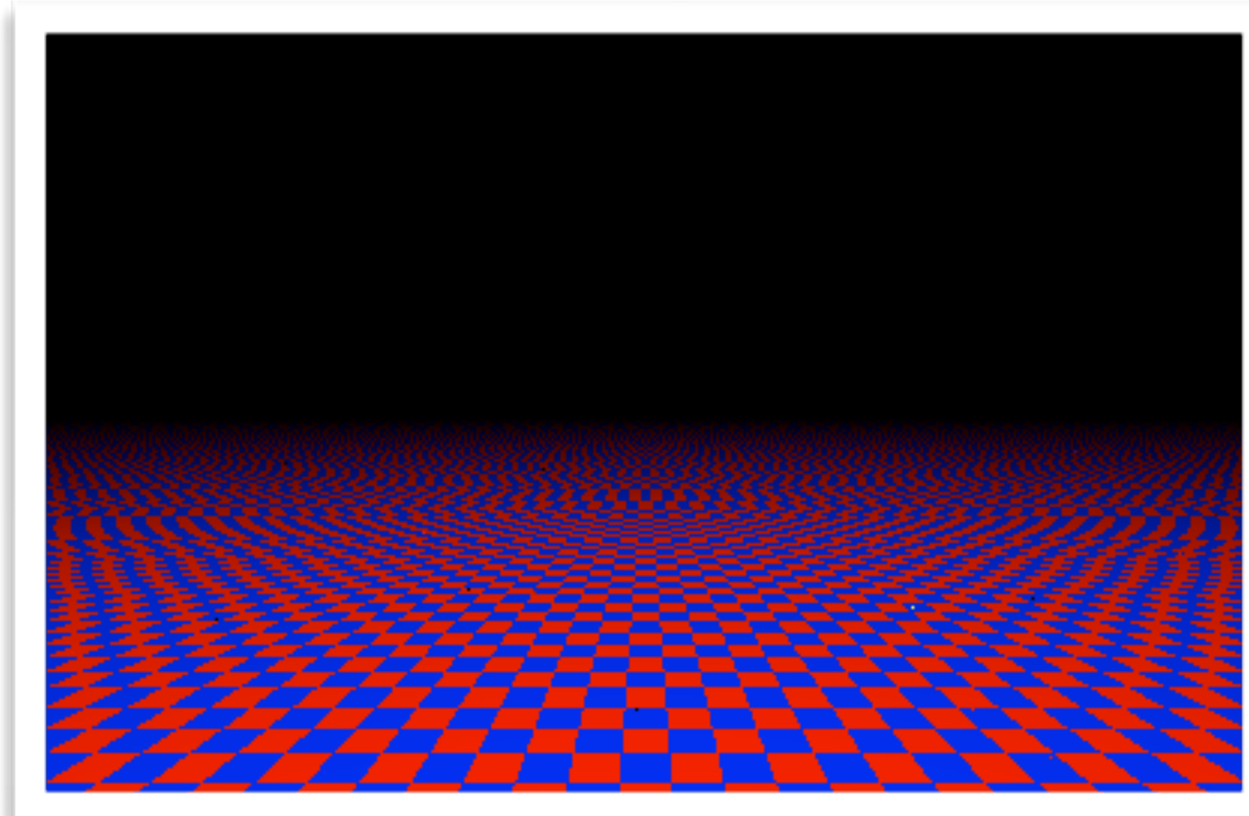


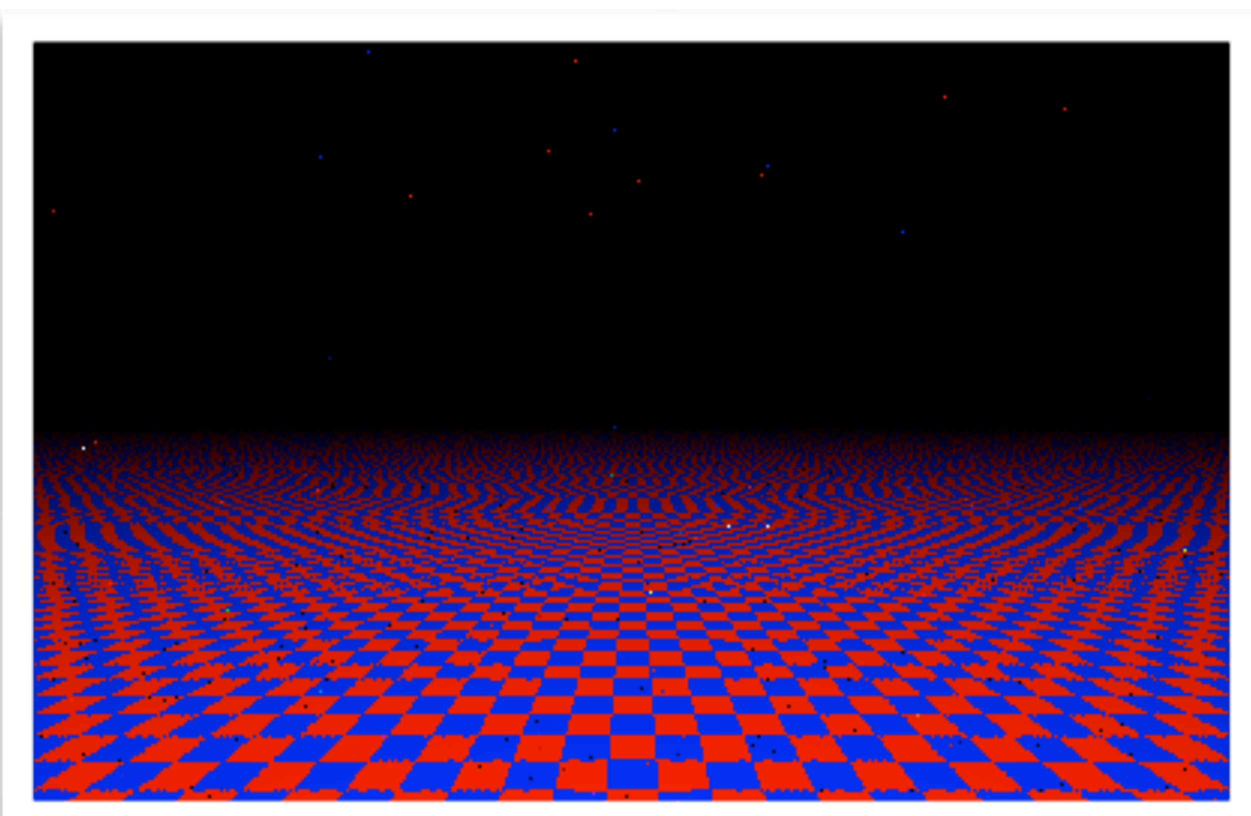

Errors
Energy
Errors
Energy
Errors
Energy
Errors
Energy

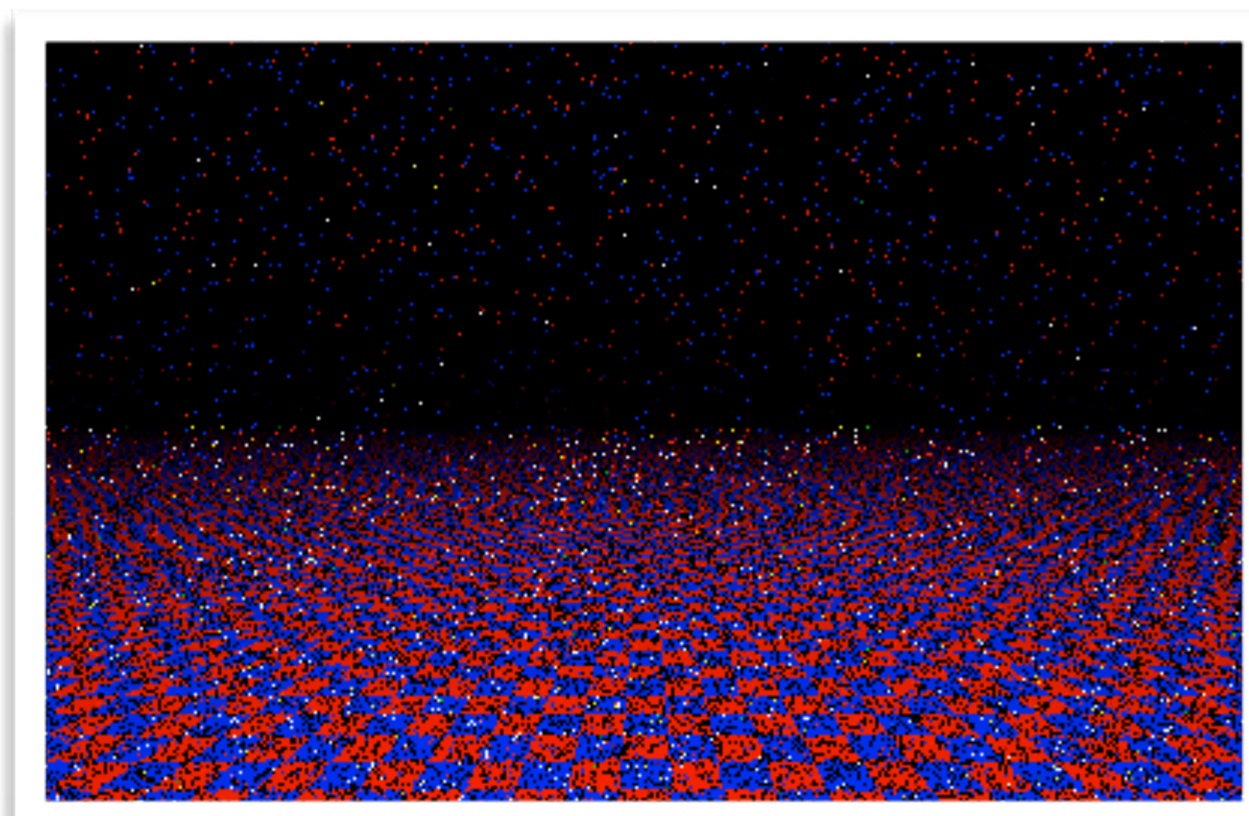

Errors
Energy
Errors
Energy
Errors
Energy
Errors
Energy

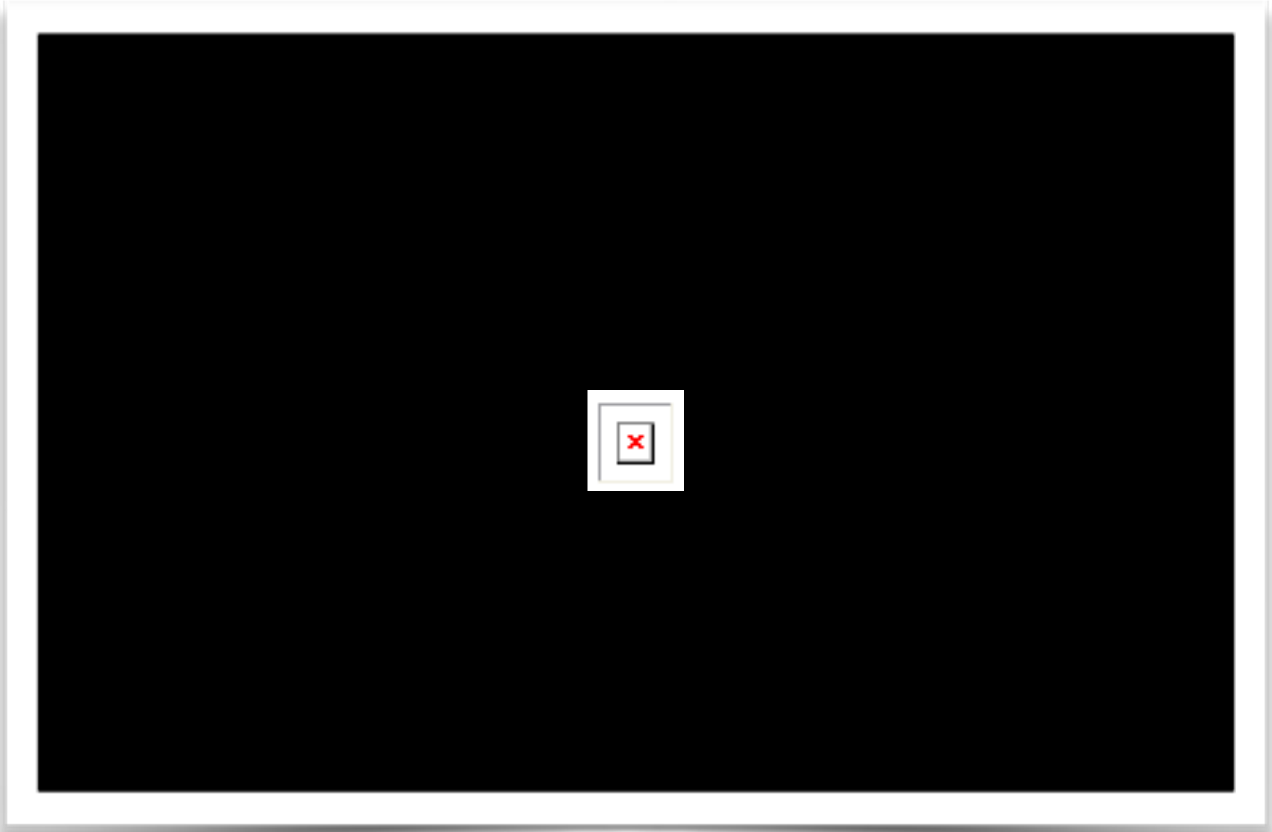
But approximation needs to be done carefully... or...

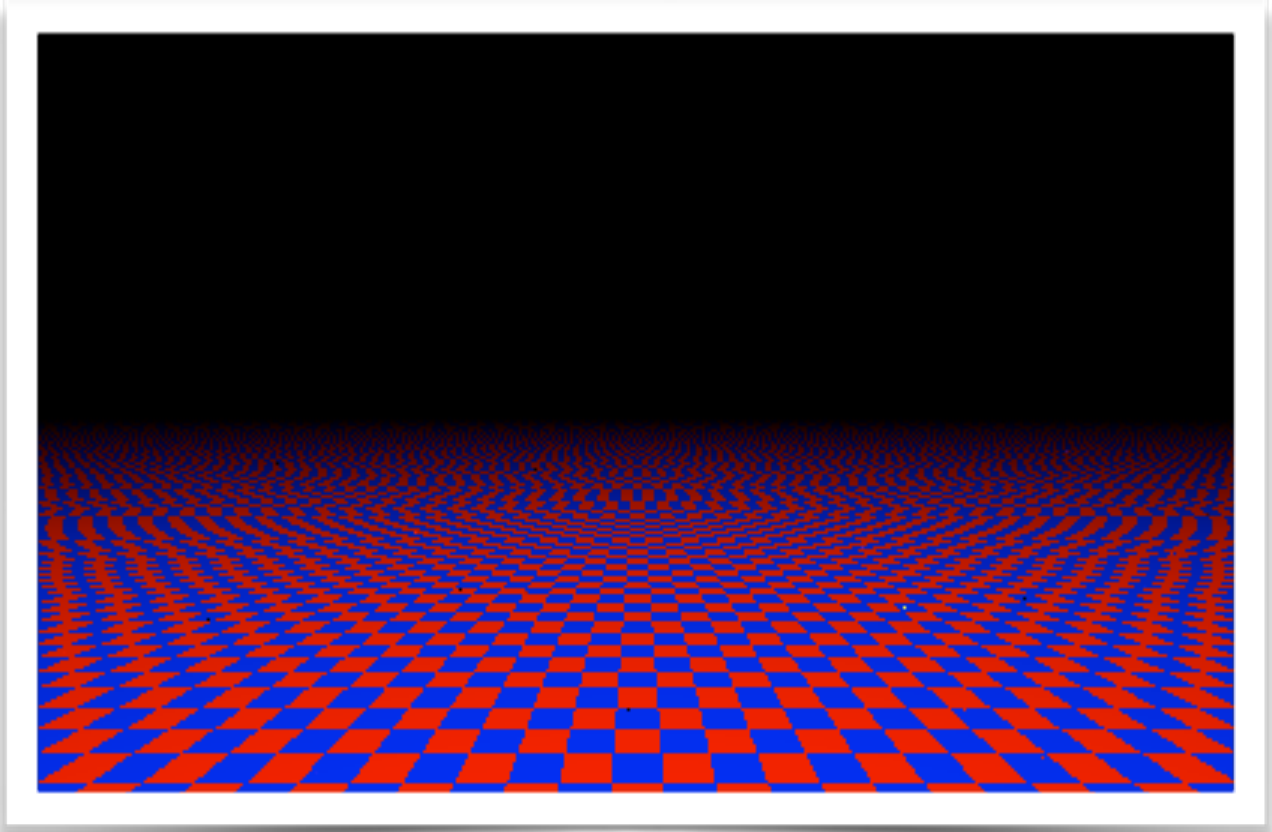


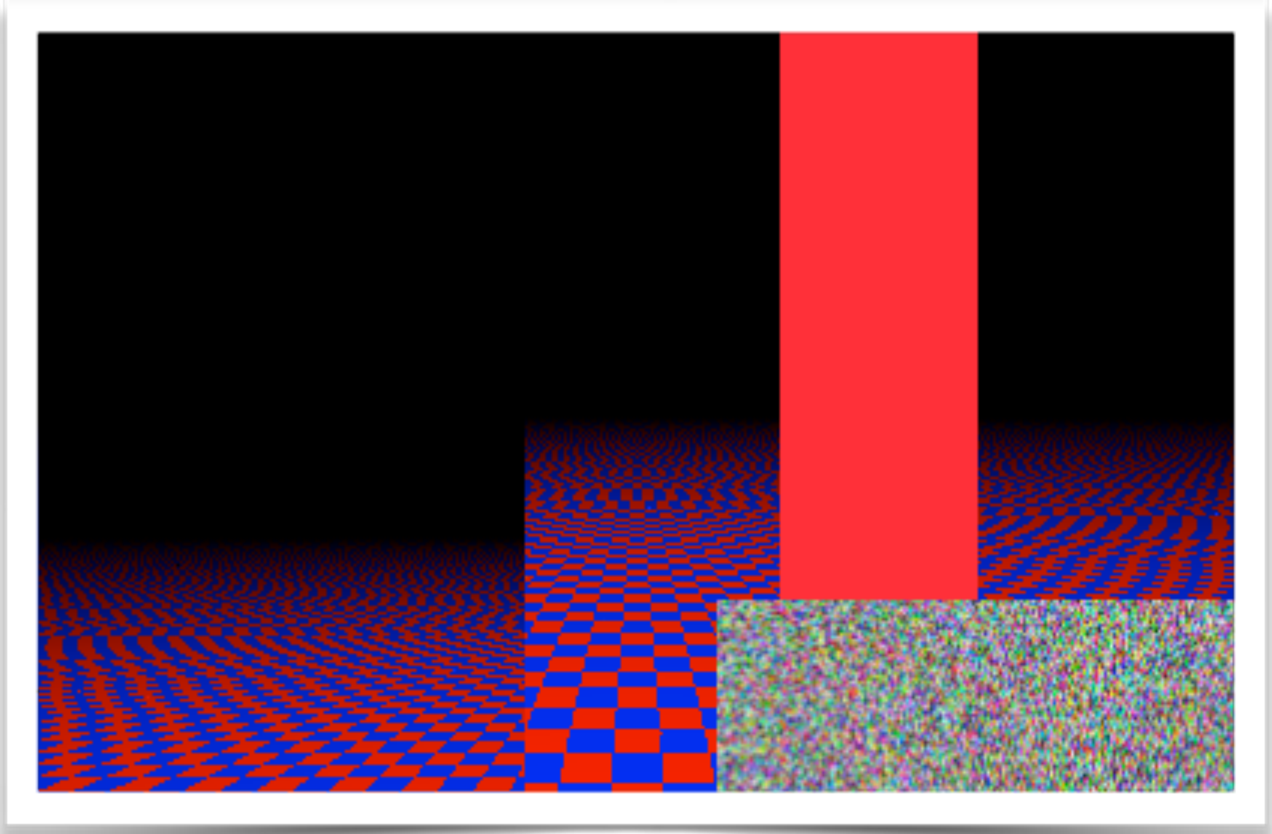


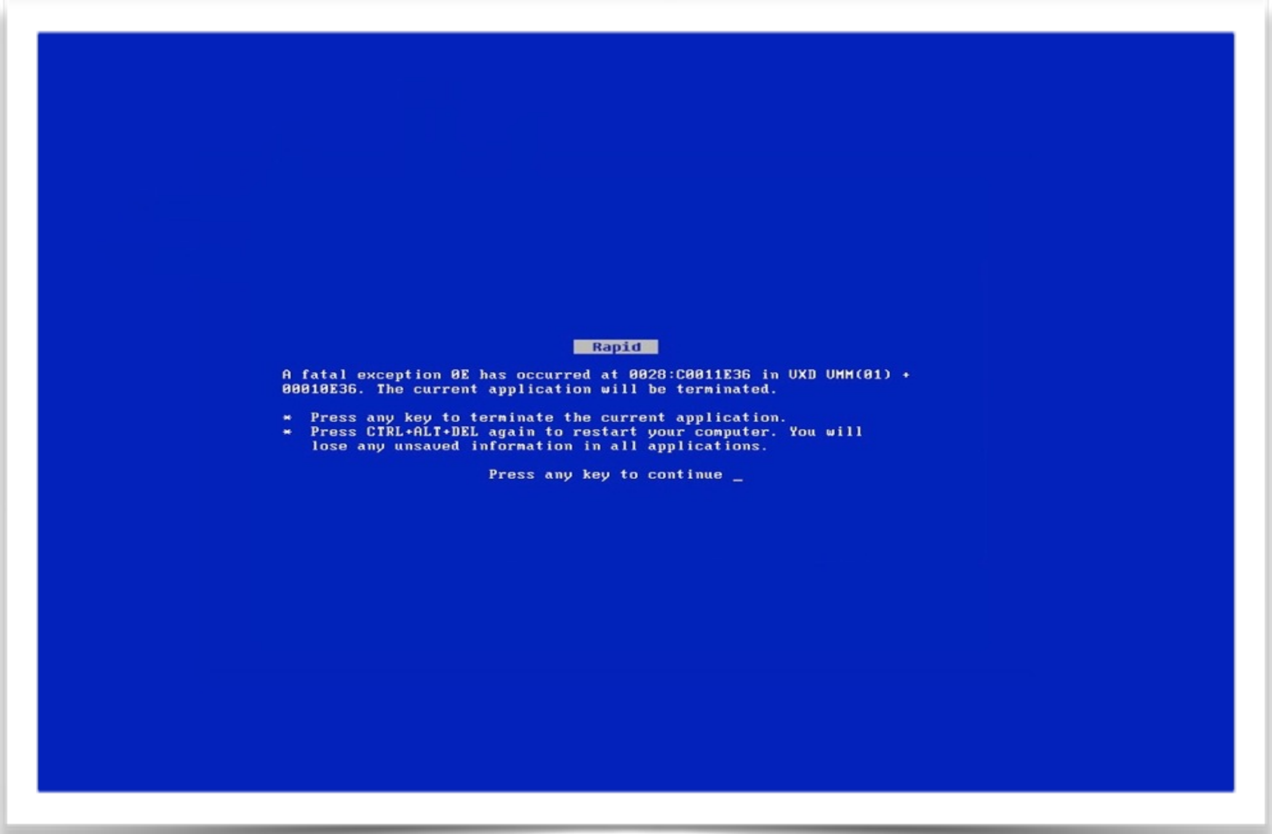








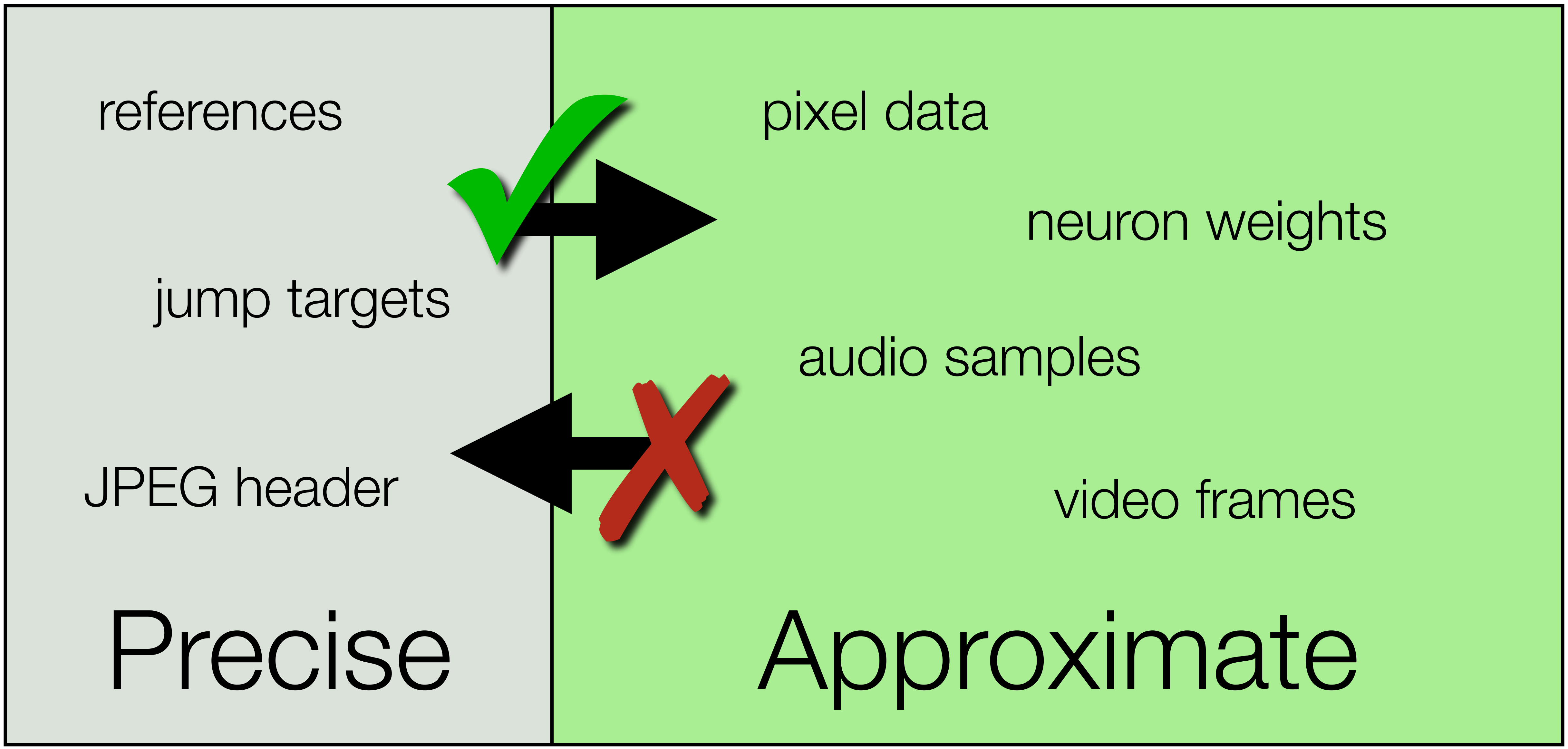



Rapid
A fatal exception 0E has occurred at 0028:C0011E36 in UXD UMM(01) + 00010E36. The current application will be terminated.
* Press any key to terminate the current application.
* Press CTRL+ALT+DEL again to restart your computer. You will lose any unsaved information in all applications.
Press any key to continue _

“Disciplined” approximate programming

Precise	Approximate
references	pixel data
jump targets	neuron weights
JPEG header	audio samples
	video frames

"Disciplined" approximate programming



- Programmer has direct control of approximate/precise and the flow
- System is free to approximate as long as rules are obeyed

Disciplined Approximate Programming
(EnerJ, EnerC,...)

```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
   a += func(2);
   @Approx int z;
   z = p * 2;
   p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

Disciplined Approximate Programming
(EnerJ, EnerC,...)

Relaxed Algorithms

```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

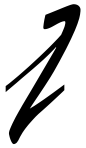
Disciplined Approximate Programming
(EnerJ, EnerC,...)

Relaxed Algorithms

ϕ **Aggressive Compilation**

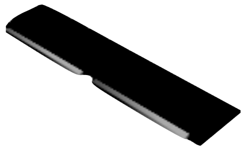
```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

Disciplined Approximate Programming
(EnerJ, EnerC,...)

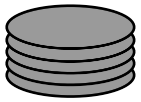


Relaxed Algorithms

ϕ **Aggressive Compilation**



Approximate Data Storage



```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

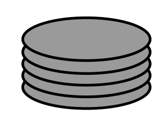
Disciplined Approximate Programming
(EnerJ, EnerC,...)

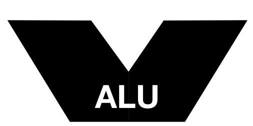
```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

Relaxed Algorithms

Φ **Aggressive Compilation**

Approximate Data Storage

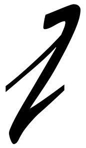




Variable-Accuracy ISA

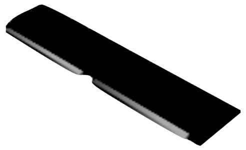
Disciplined Approximate Programming (EnerJ, EnerC,...)

```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

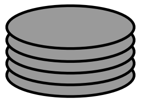


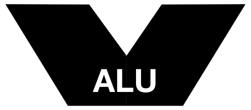
Relaxed Algorithms

Φ Aggressive Compilation

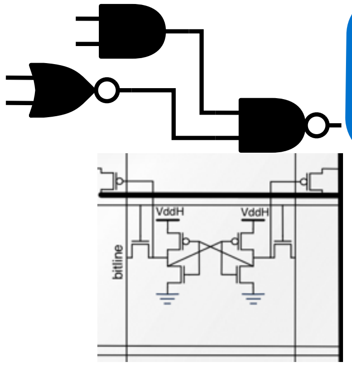


Approximate Data Storage





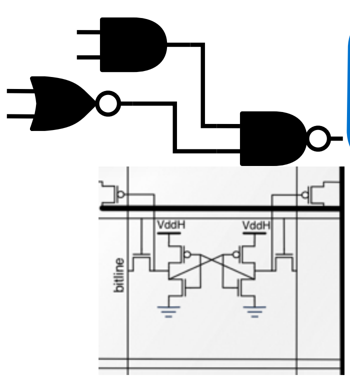
Variable-Accuracy ISA



Approximate Logic/Circuits

Disciplined Approximate Programming (EnerJ, EnerC,...)

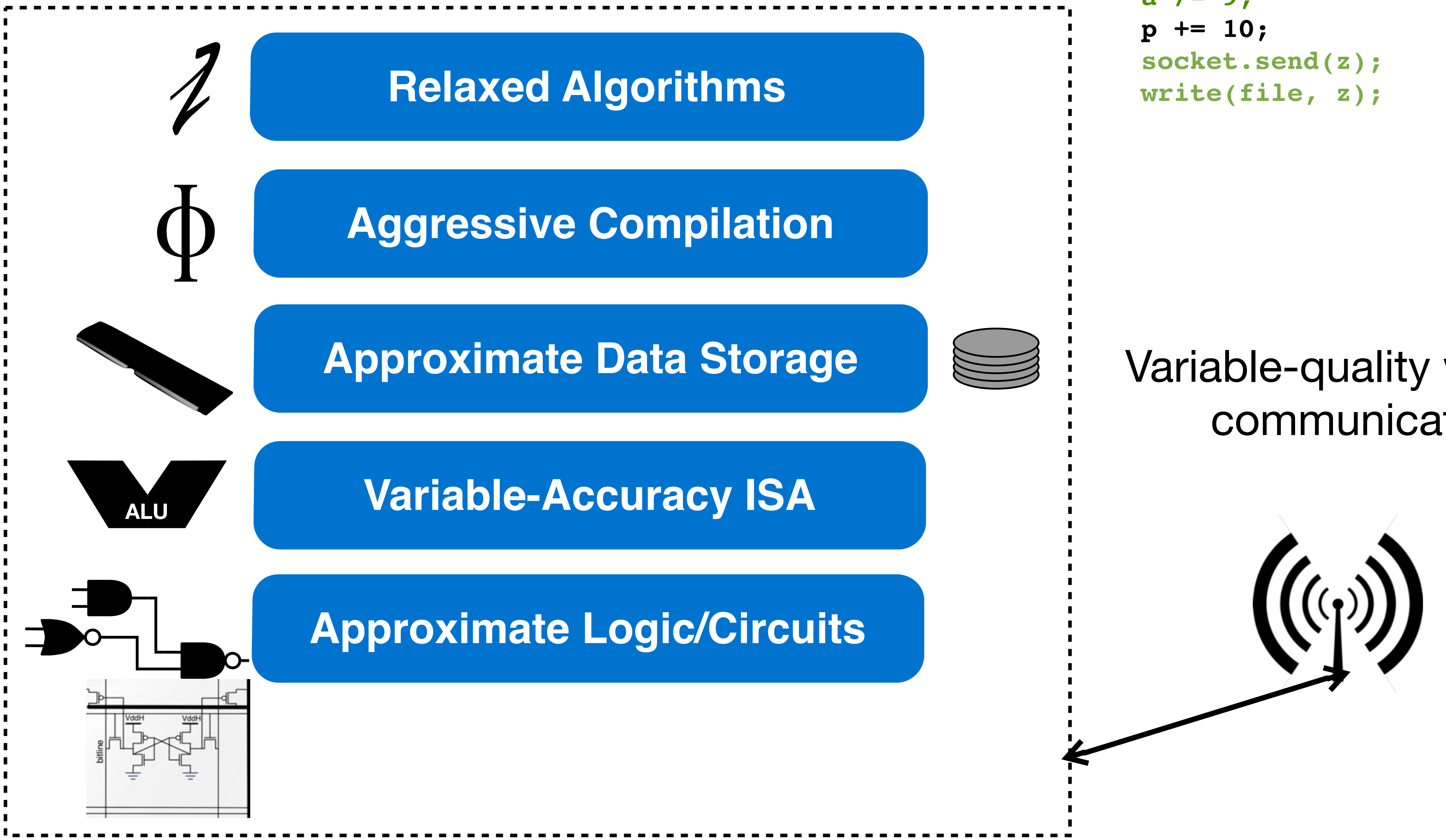
- Relaxed Algorithms
- Aggressive Compilation
- Approximate Data Storage
- Variable-Accuracy ISA
- Approximate Logic/Circuits



```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```

Disciplined Approximate Programming (EnerJ, EnerC,...)

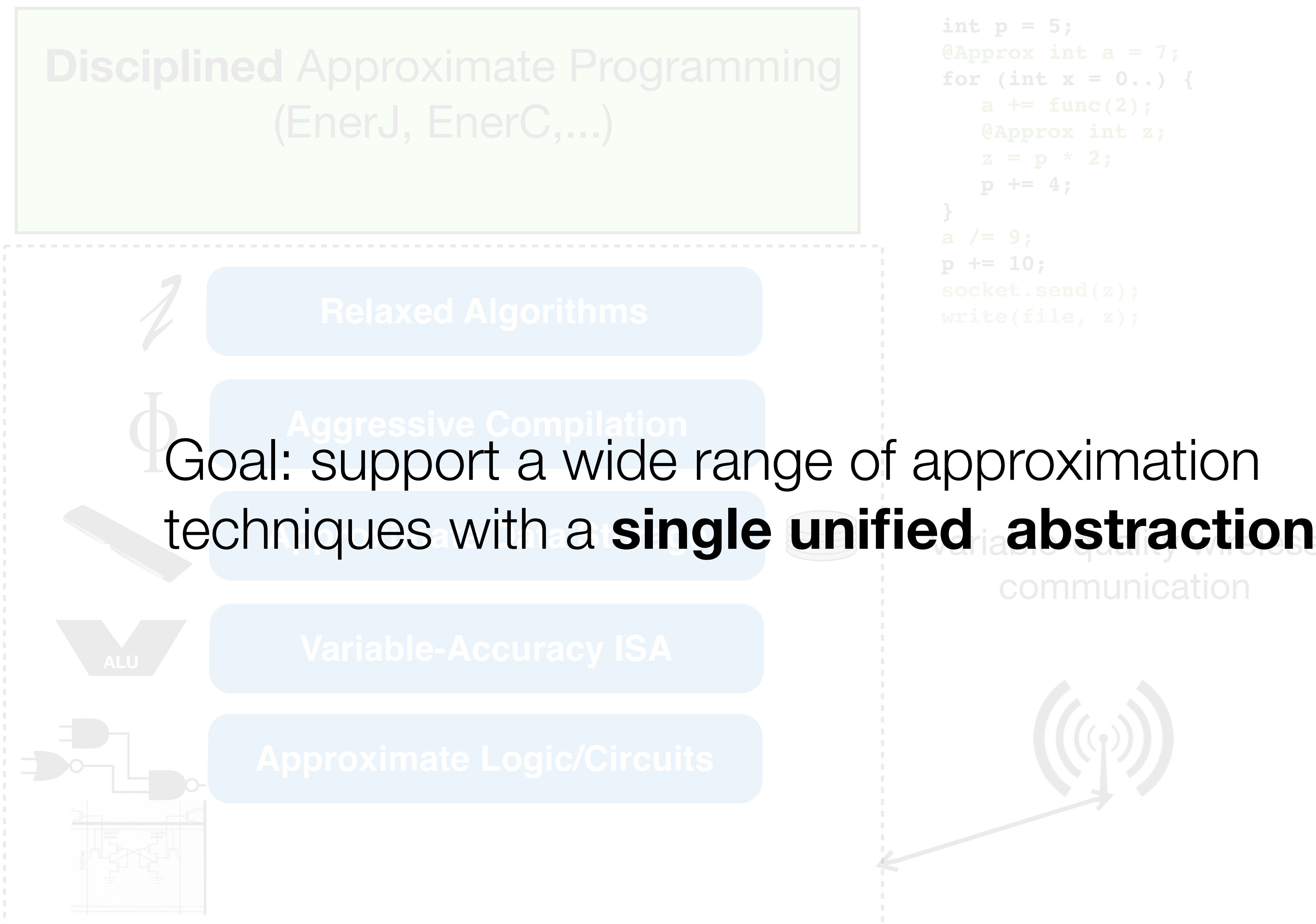
```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```



Disciplined Approximate Programming
(EnerJ, EnerC,...)

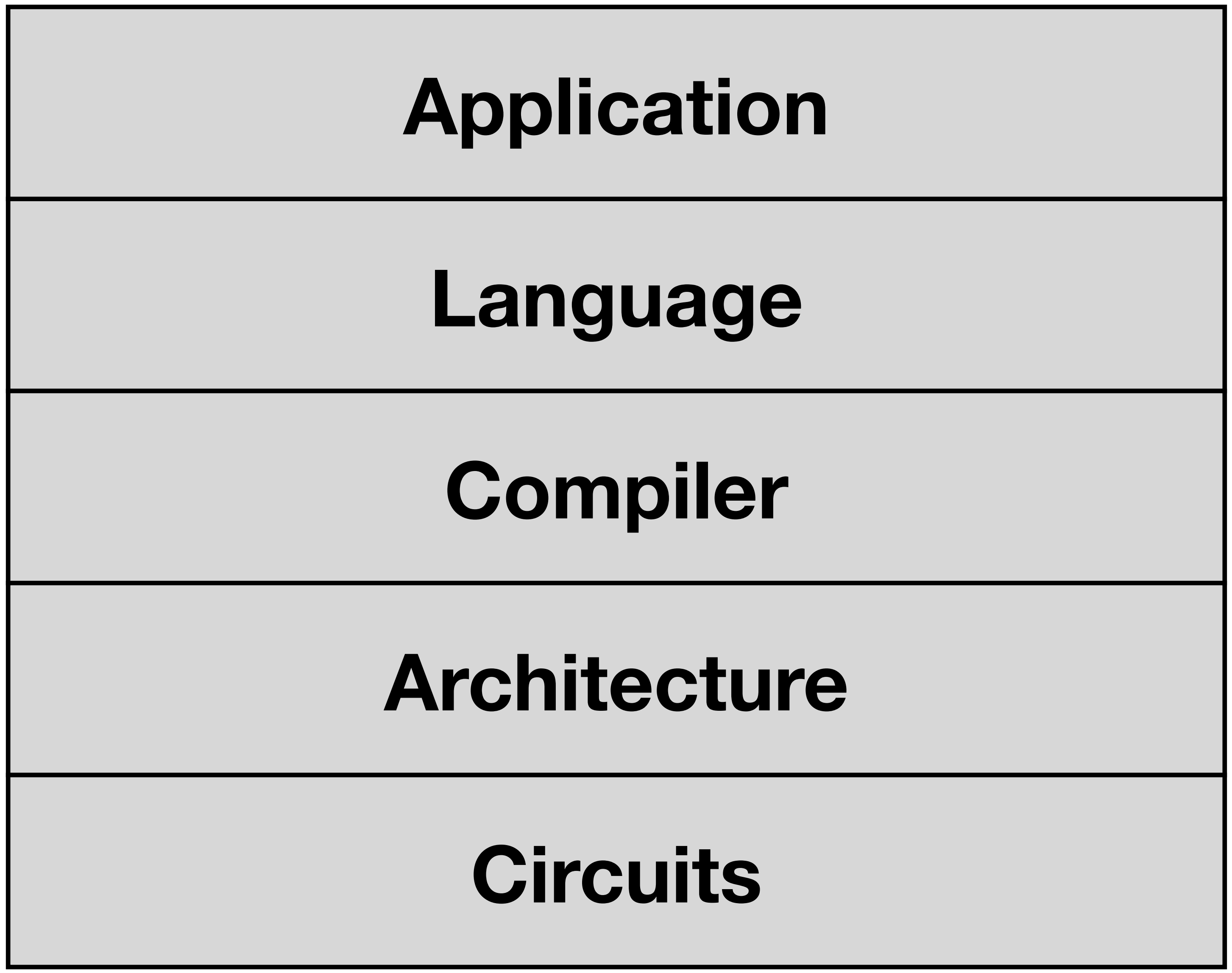
```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
   a += func(2);
   @Approx int z;
   z = p * 2;
   p += 4;
}
a /= 9;
p += 10;
socket.send(z);
write(file, z);
```


Variable-quality wireless communication

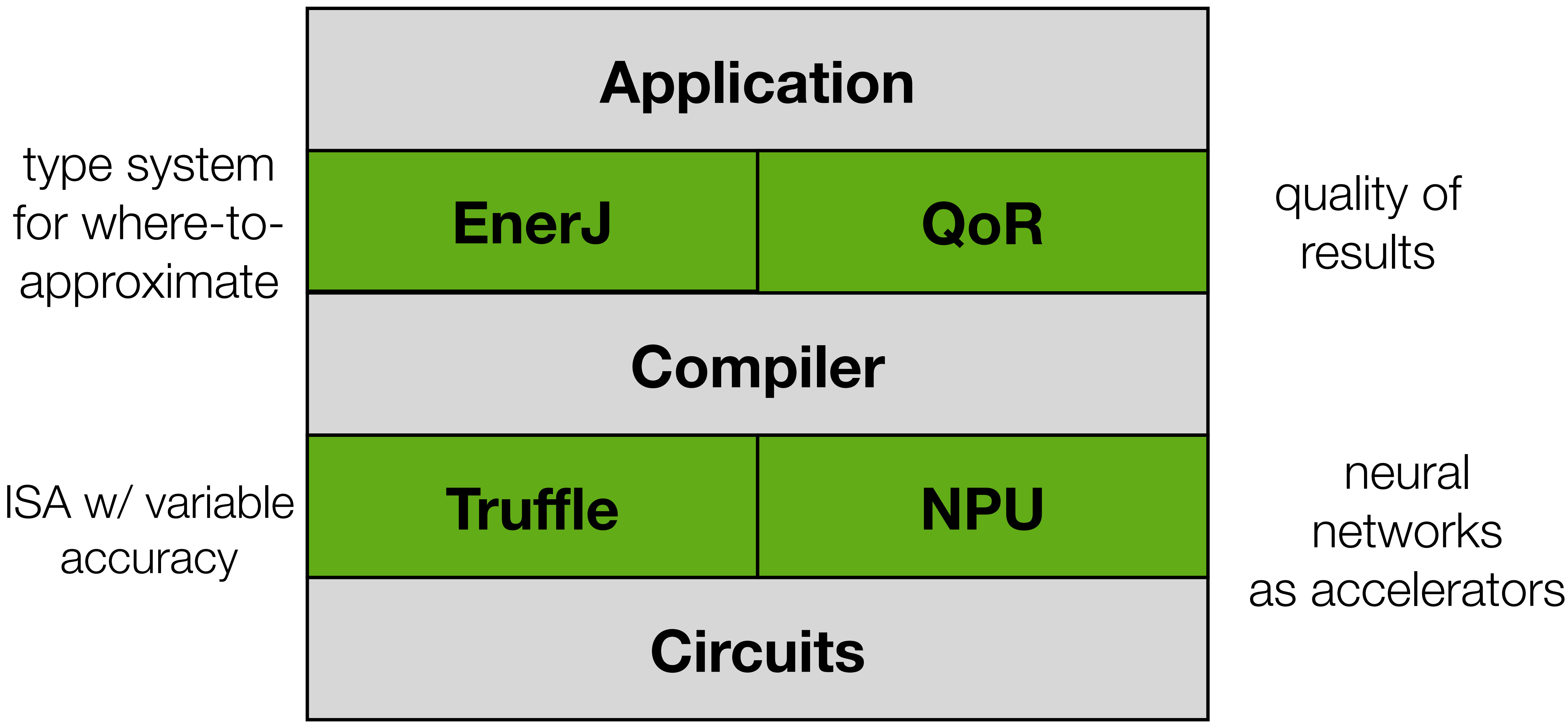


Goal: support a wide range of approximation techniques with a **single unified abstraction**.

The plan for this talk



The plan for the rest of this talk



Approximate Storage

Approximate Wireless

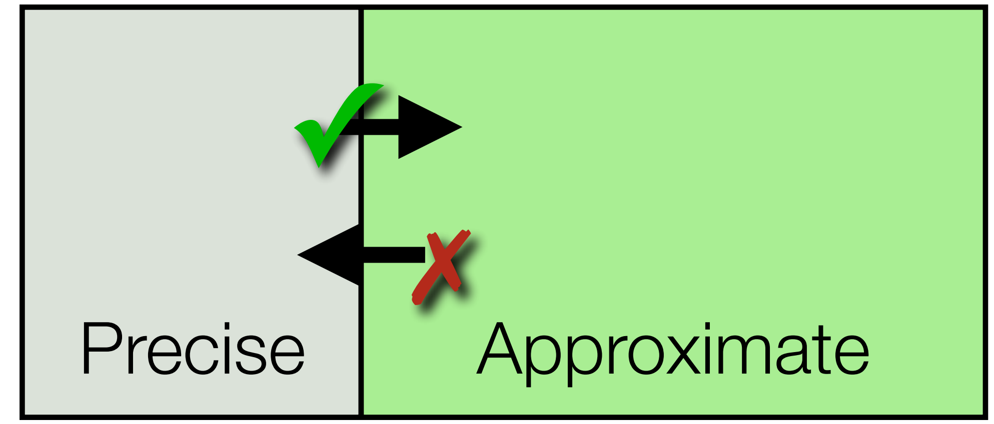
Prototype, etc

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

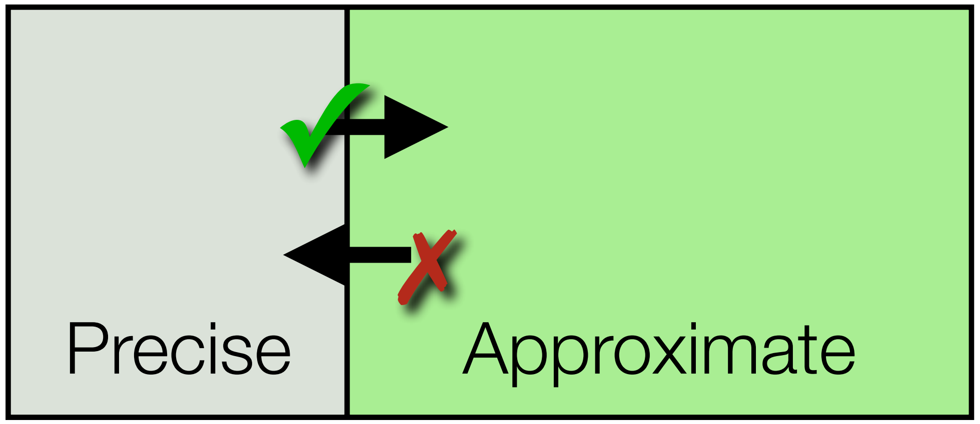


Separate critical and non-critical program components. *Analyzable statically.*

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

Separate critical and non-critical program components. *Analyzable statically.*

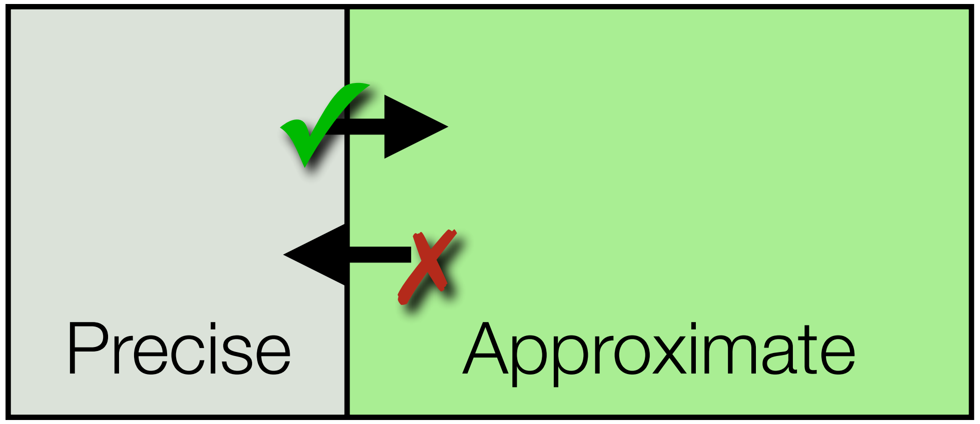



```
int a = ...;

int p = ...;
```

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011



Separate critical and non-critical program components. *Analyzable statically.*

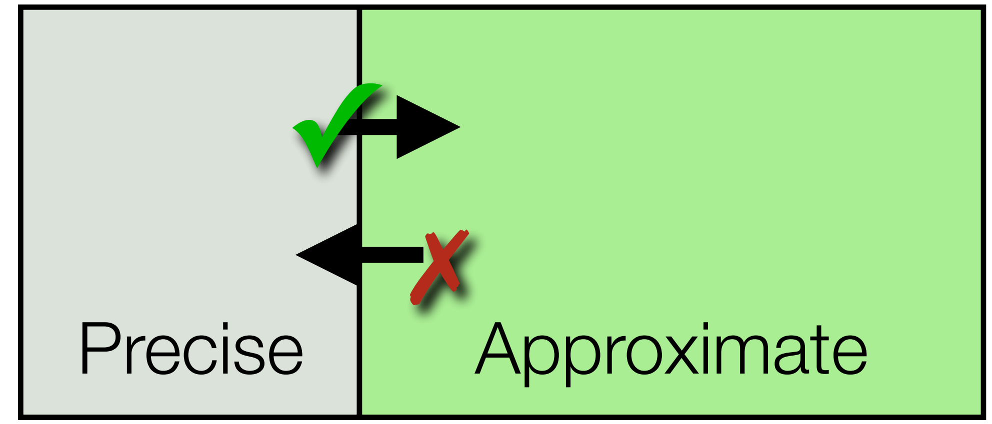
```
@Approx int a = ...;

@Precise int p = ...;
```

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

Separate critical and non-critical program components. *Analyzable statically.*




```
@Approx int a = ...;

@Precise int p = ...;

p = a;

a = p;
```

EnerJ


```
   @Approx   int a = ...;
 @Precise    int p = ...;
```

- Operator overloading for approximate operations:
- Endorsement of approximate values:
- Dealing with implicit flows in control:

EnerJ


```
@Approx   int a = ...;
@Precise  int p = ...;
```

- Operator overloading for approximate operations: `p + p;  p + a;  a + a;`
- Endorsement of approximate values:
- Dealing with implicit flows in control:

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

```
@Approx   int a = ...;
@Precise  int p = ...;
```

- Operator overloading for approximate operations: `p + p;` `p + a;` `a + a;`
- Endorsement of approximate values:
  ✓ `p = endorse(a);`
- Dealing with implicit flows in control:

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

```
@Approx   int a = ...;
@Precise  int p = ...;
```

- Operator overloading for approximate operations: `p + p;  p + a;  a + a;`

- Endorsement of approximate values:

```
p = endorse(a);
```

- Dealing with implicit flows in control:

```
if (a == 10)
    p = 2;
}
```

EnerJ

EnerJ: Approximate Data Types for Safe and General Low-Power Computation, PLDI 2011

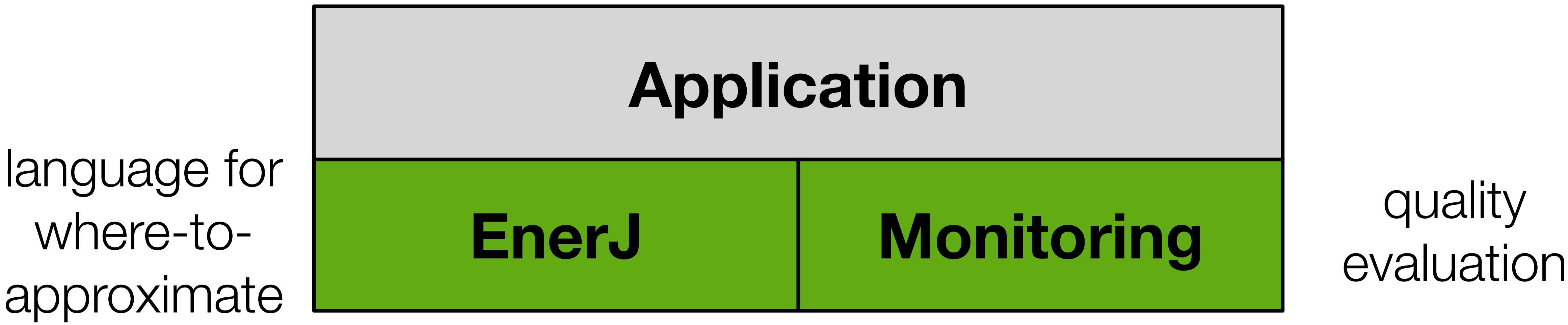
```
@Approx   int a = ...;
@Precise  int p = ...;
```

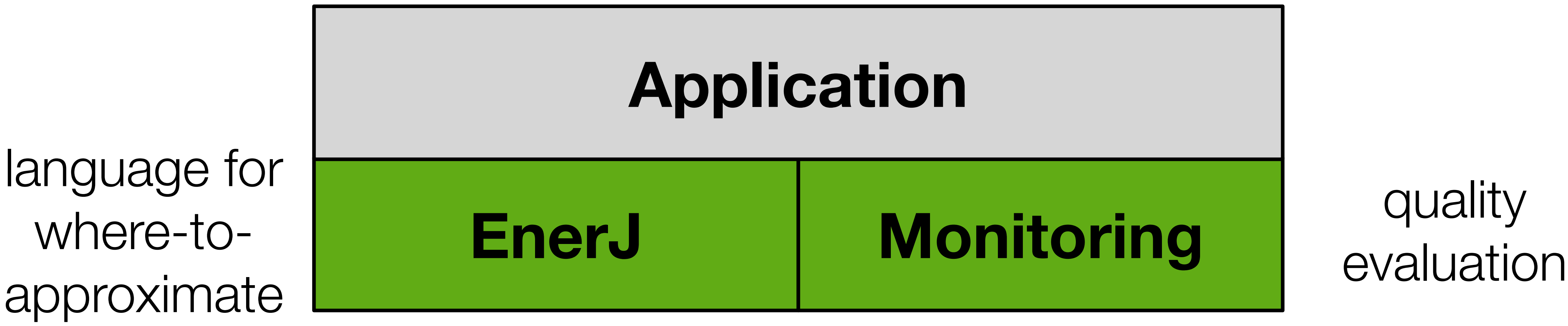
- Operator overloading for approximate operations: `p + p;  p + a;  a + a;`

- Endorsement of approximate values:
  ✓ `p = endorse(a);`

- Dealing with implicit flows in control:

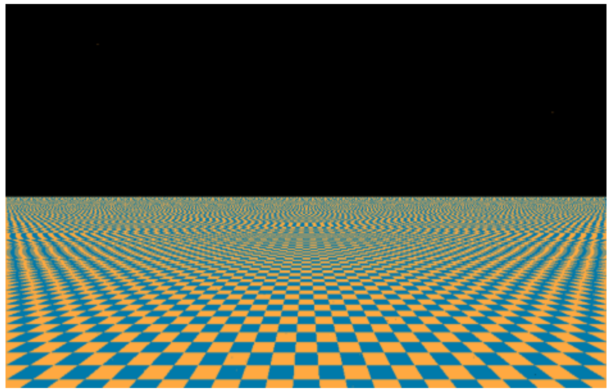
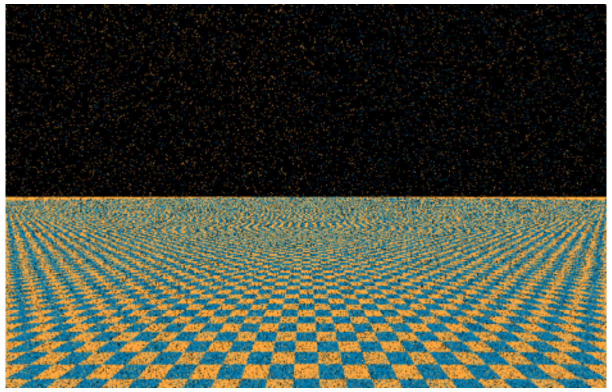
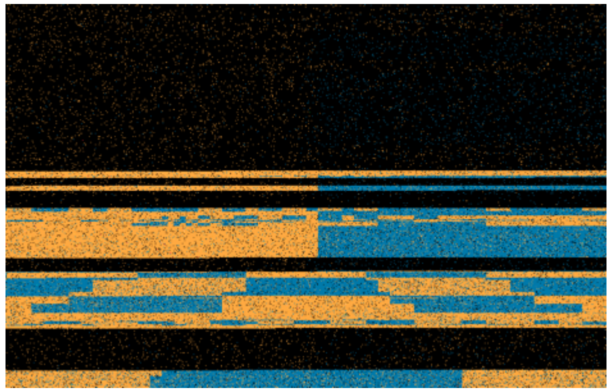
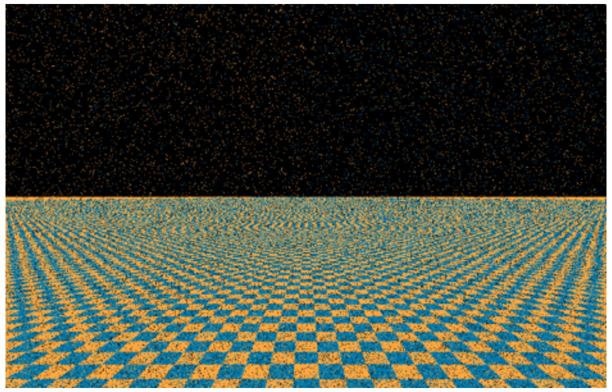
✓
```
if (endorse(a == 10)) {
    p = 2;
}
```


language for where-to-approximate
Application
EnerJ
Monitoring
quality evaluation


Application
EnerJ
Monitoring
language for
where-to-
approximate
quality
evaluation

How good is my final output?

- *Quality-of-Result (QoR)*
- Application dependent
  – e.g, % of bad pixels, deviation from expected value, % of poorly classified images, car crashes, etc…

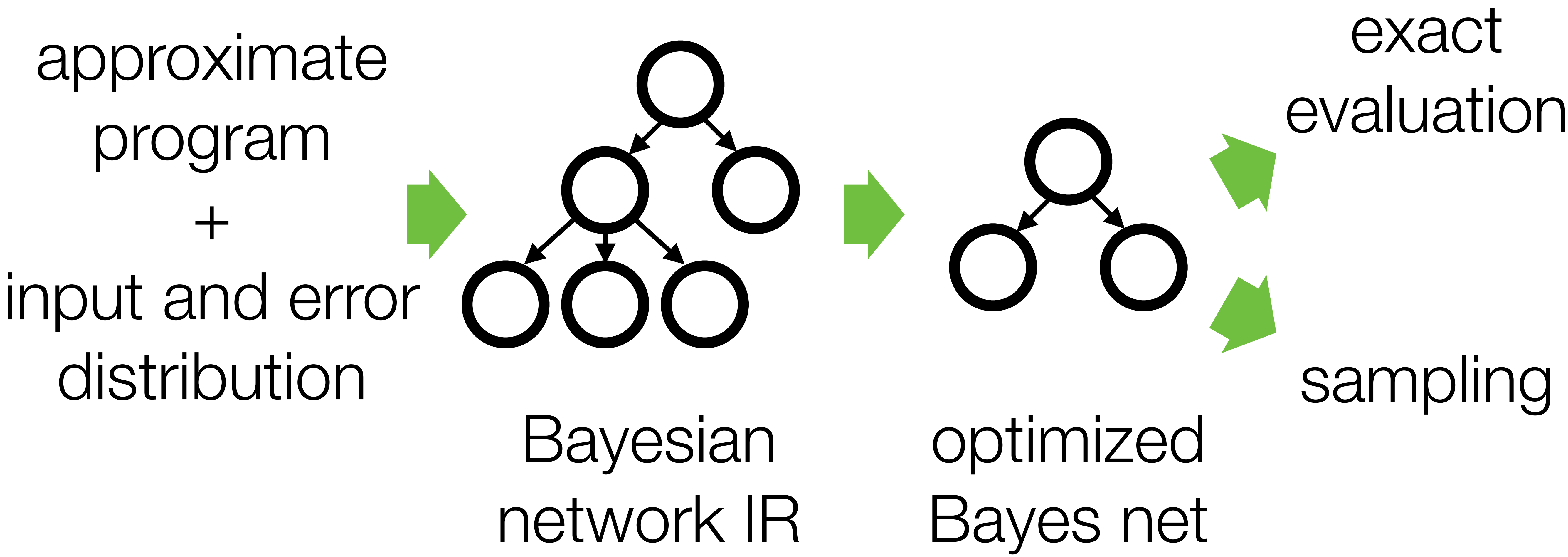
Specifying and checking QoR

Specifying and checking QoR

res = **computeSomething()**;
assert diff(res, **res'**) < 0.1;

precise version of the result

Verifying quality expressions



Expressing and Verifying Probabilistic Assertions, PLDI'14

Online QoR monitoring

Can *react* – recompute or reduce approximation

But needs to be cheap!

Online QoR monitoring

Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled

precise

re-execution

Online QoR monitoring

Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled

precise

re-execution

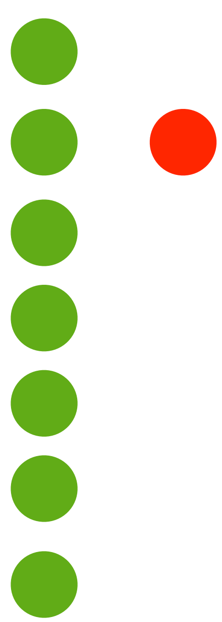
-
-
-
-
-
-
-

Online QoR monitoring

Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled
precise
re-execution



Online QoR monitoring

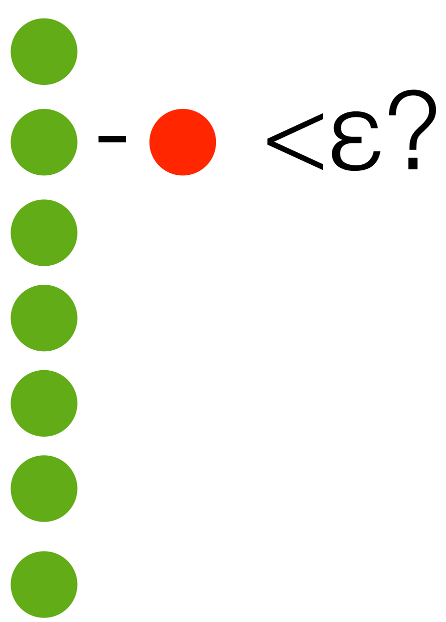
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled

precise

re-execution



Online QoR monitoring

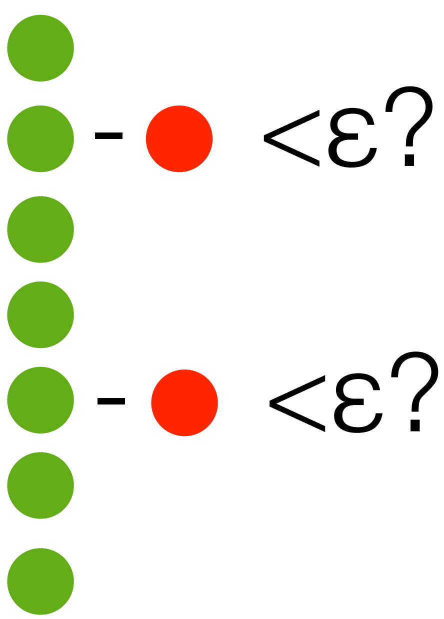
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled

precise

re-execution

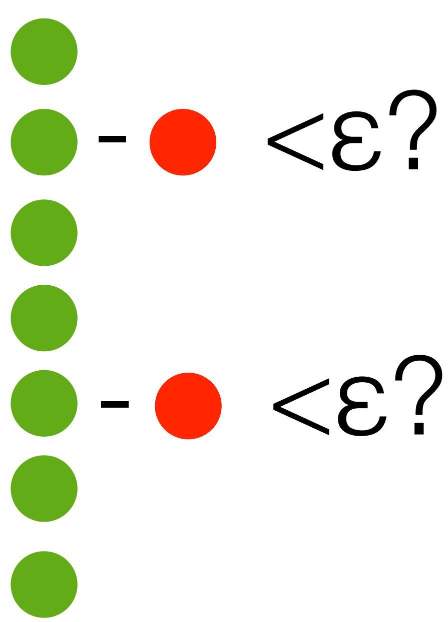


Online QoR monitoring

Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled precise re-execution



Online QoR monitoring

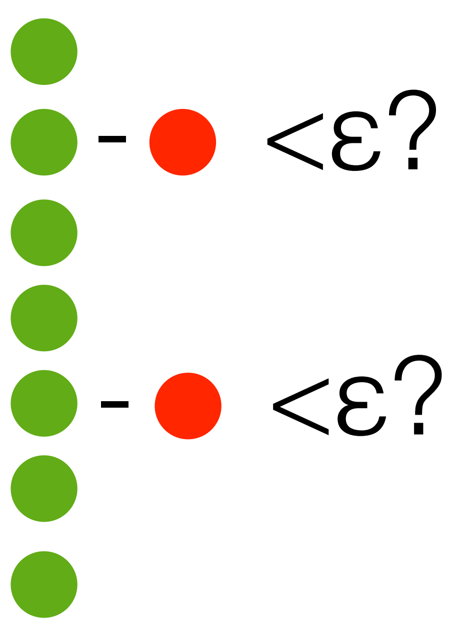
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled

precise

re-execution



Simple

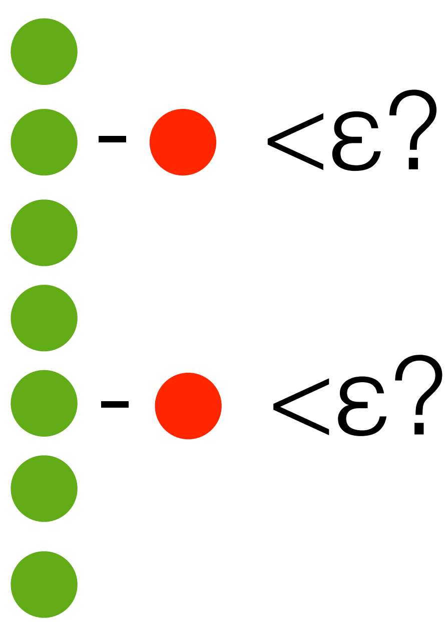
verification functions

Online QoR monitoring

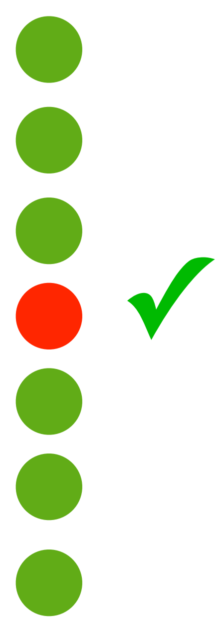
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled precise re-execution



Simple verification functions

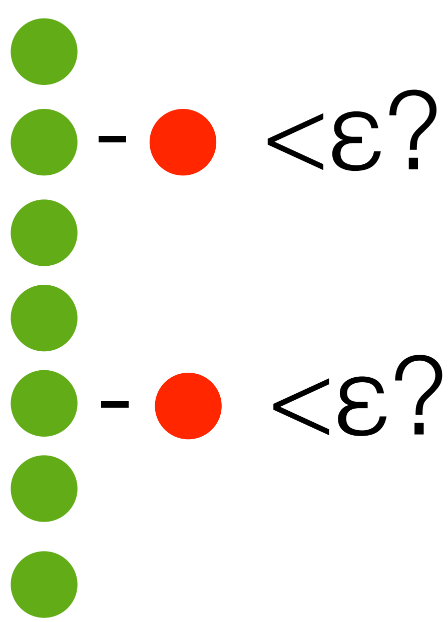


Online QoR monitoring

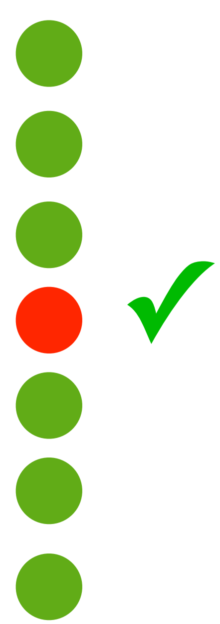
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled precise re-execution



Simple verification functions

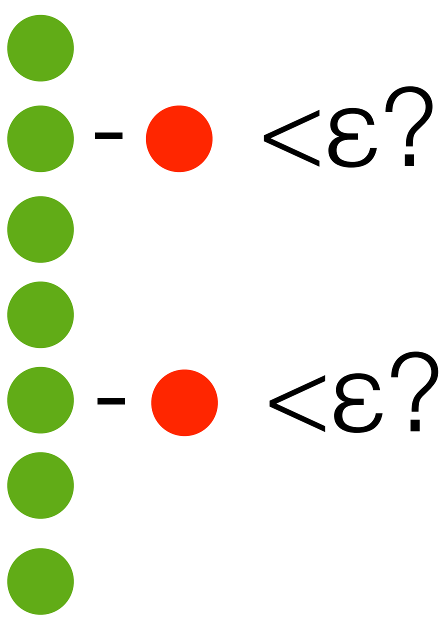


Online QoR monitoring

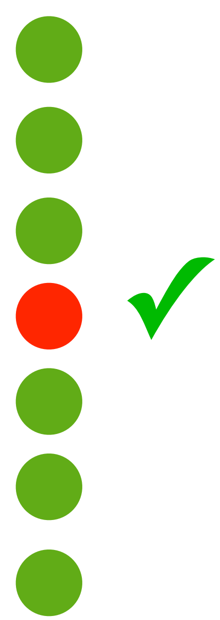
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled precise re-execution



Simple verification functions



Fuzzy Memoization

Online QoR monitoring

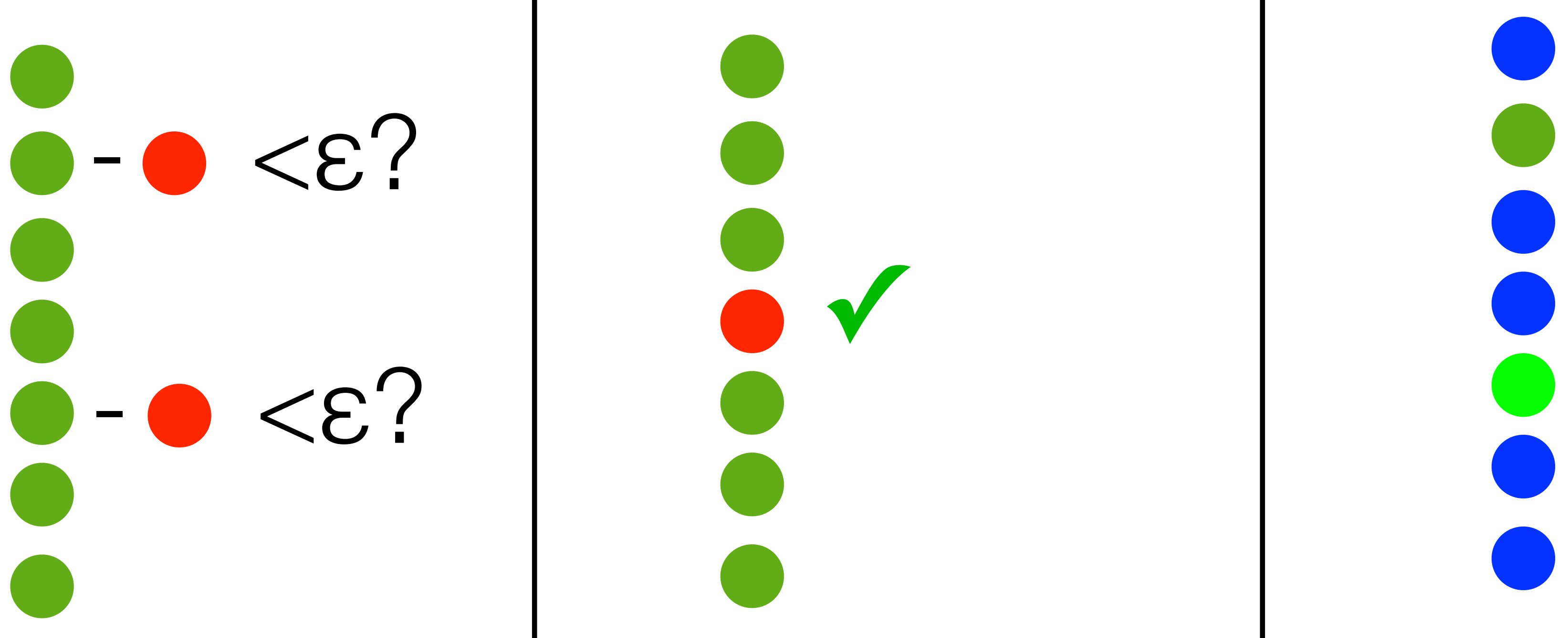
Can *react* – recompute or reduce approximation

But needs to be cheap!

Sampled precise re-execution

Simple verification functions

Fuzzy Memoization

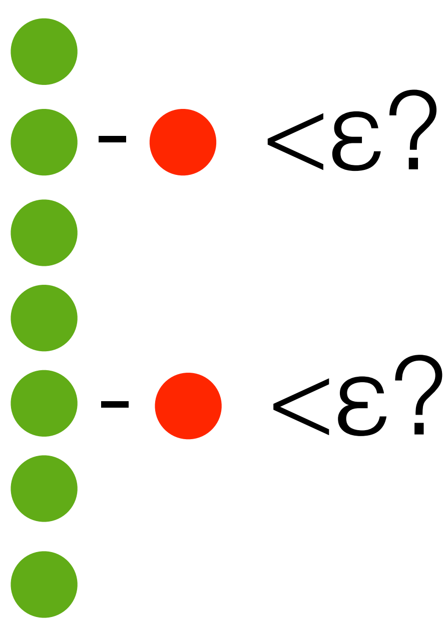


Online QoR monitoring

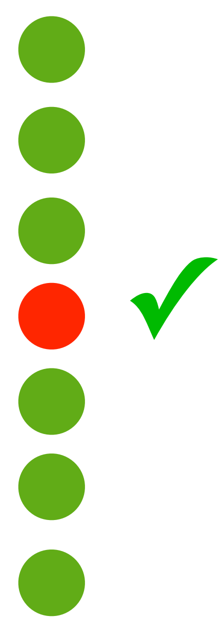
Can *react* – recompute or reduce approximation

But needs to be cheap!

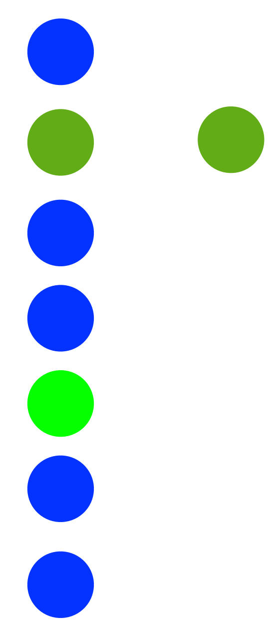
Sampled precise re-execution



Simple verification functions



Fuzzy Memoization

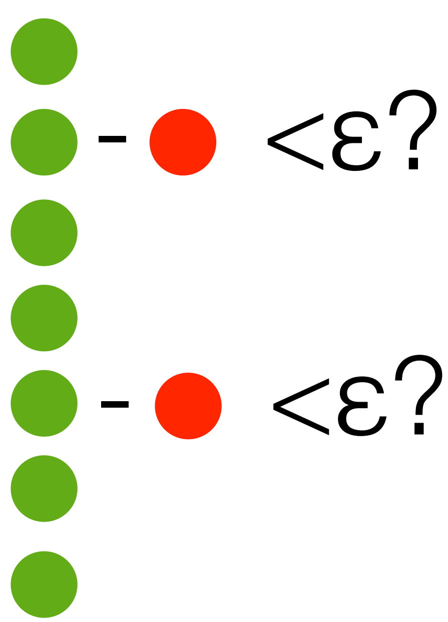


Online QoR monitoring

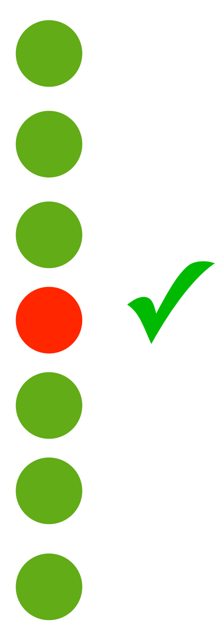
Can *react* – recompute or reduce approximation

But needs to be cheap!

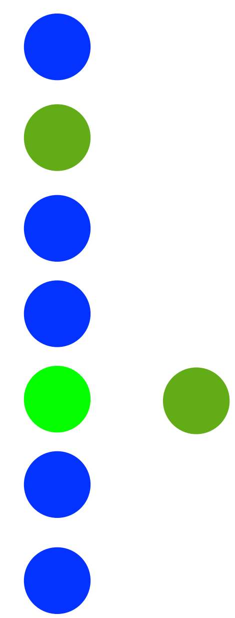
Sampled precise re-execution



Simple verification functions



Fuzzy Memoization

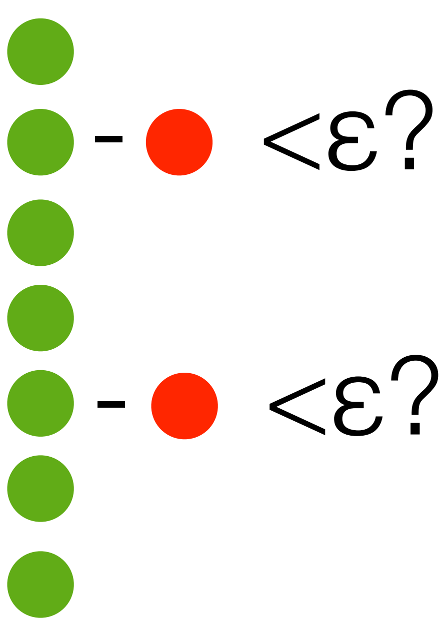


Online QoR monitoring

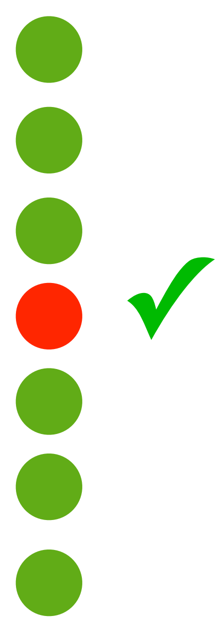
Can *react* – recompute or reduce approximation

But needs to be cheap!

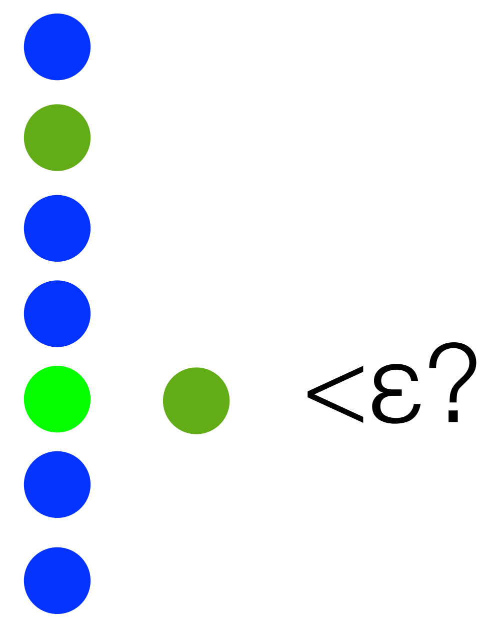
Sampled precise re-execution



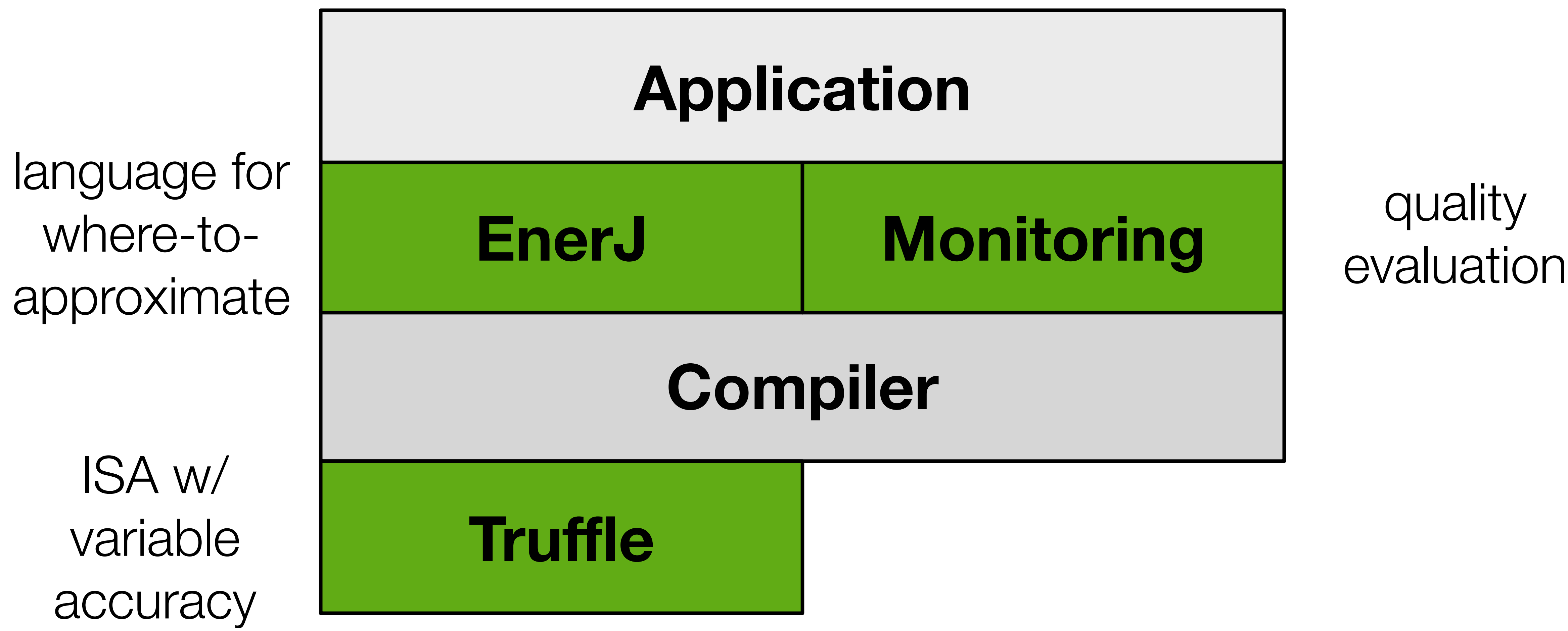
Simple verification functions



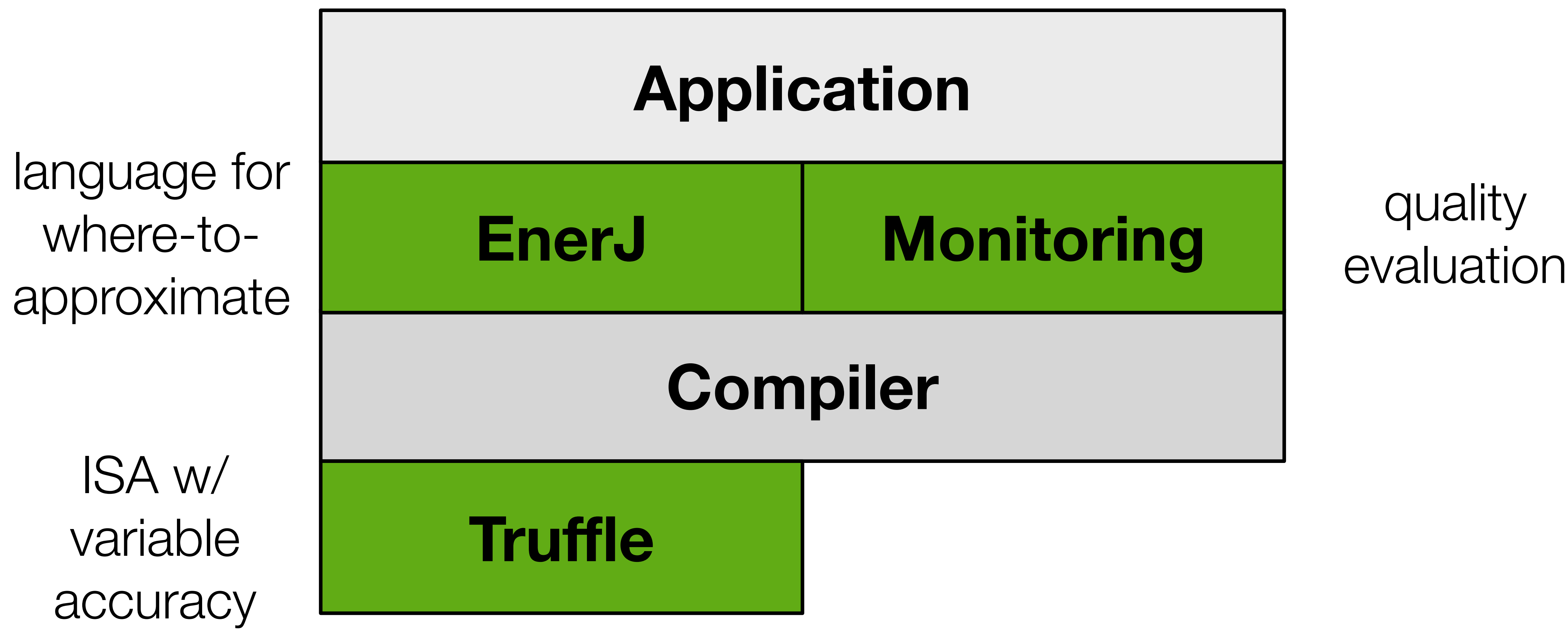
Fuzzy Memoization



What about *actual* approximate execution?

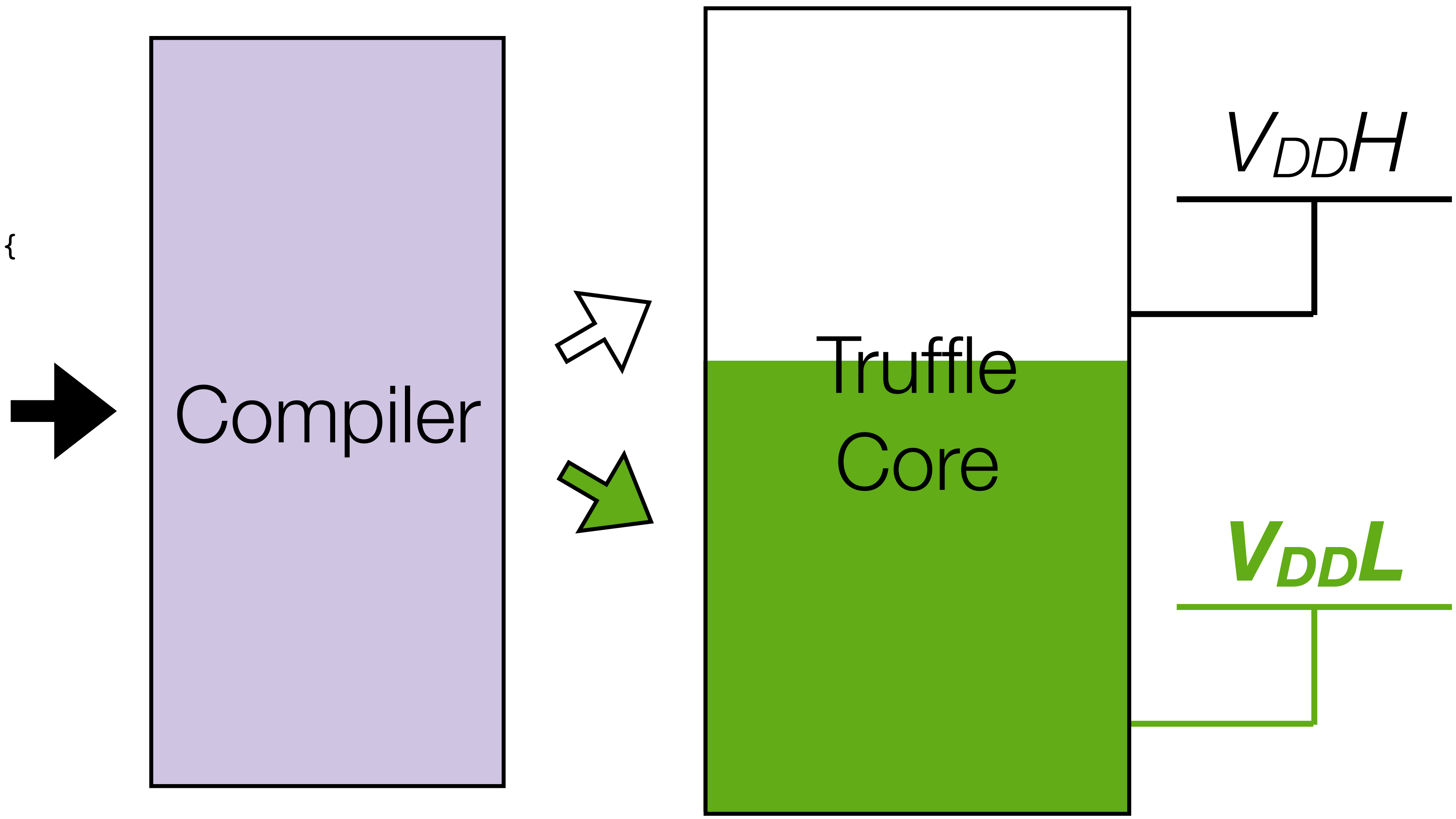


What about *actual* approximate execution?



Hardware support for **disciplined approximate execution**

```
int p = 5;
@Approx int a = 7;
for (int x = 0..) {
    a += func(2);
    @Approx int z;
    z = p * 2;
    p += 4;
}
a /= 9;
func2(p);
a += func(2);
@Approx int y;
z = p * 22 + z;
p += 10;
```



Architecture Support for Disciplined Approximate Programming, ASPLOS 2012

```
@Approx float[] nums;
⋮
@Approx float total = 0.0f;
for (@Precise int i = 0;
     i < nums.length;
     ++i)
  total += nums[i];
return total / nums.length;
```

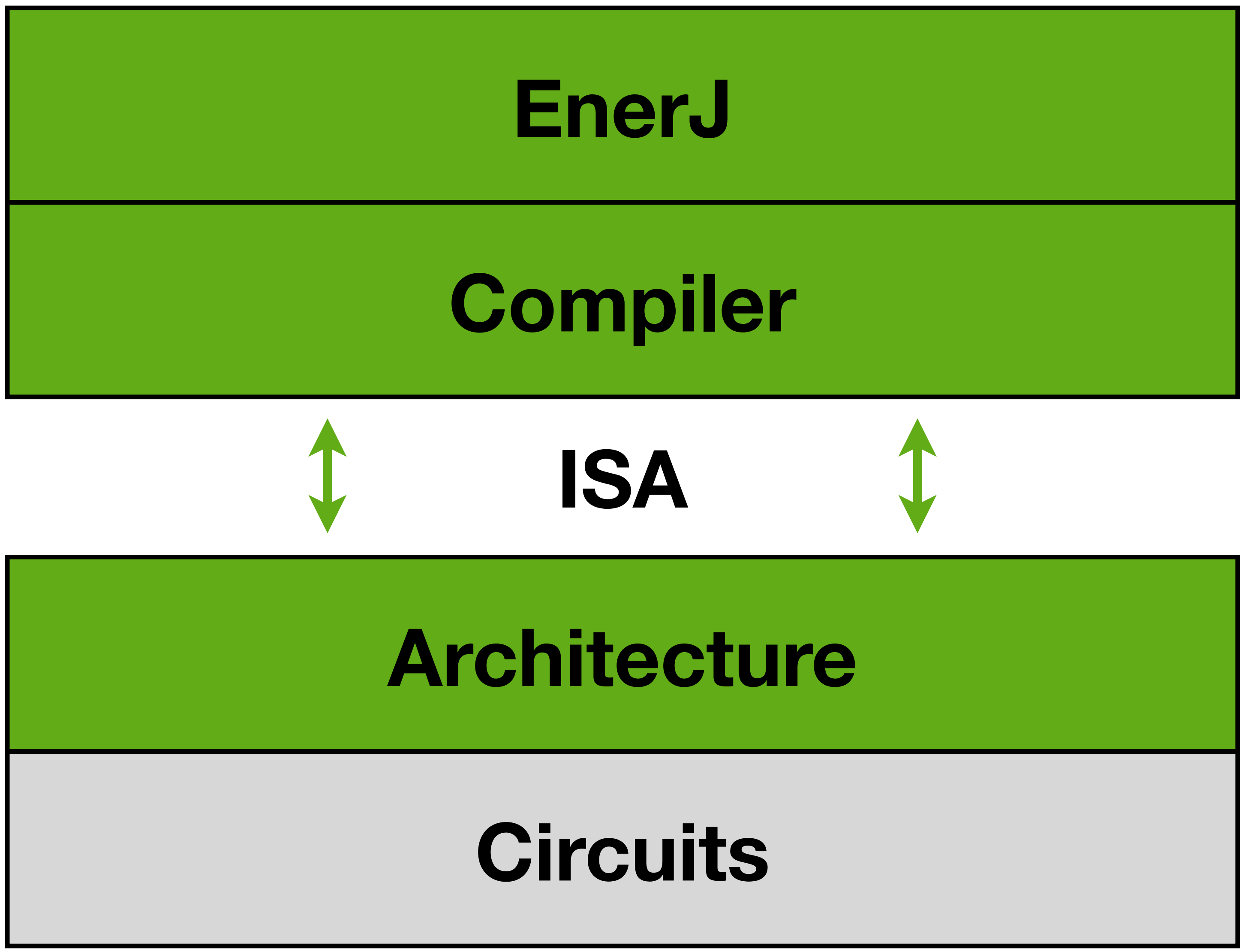
```
@Approx float[] nums;
⋮
@Approx float total = 0.0f;
for (@Precise int i = 0;
        i < nums.length;
        ++i)
    total += nums[i];
return total / nums.length;
```

approximate data storage

```
@Approx float[] nums;
⋮
@Approx float total = 0.0f;
for (@Precise int i = 0;
       i < nums.length;
       ++i)
    total += nums[i];
return total / nums.length;
```

approximate operations

Relaxing the hardware-software interface



Approximation-aware ISA

```
ld      0x04 r1
ld      0x08 r2
add     r1   r2   r3
st      0x0c r3
```

Approximation-aware ISA

```
ld     0x04 r1
ld     0x08 r2
add.a r1   r2   r3
st.a   0x0c r3
```

Approximation-aware ISA

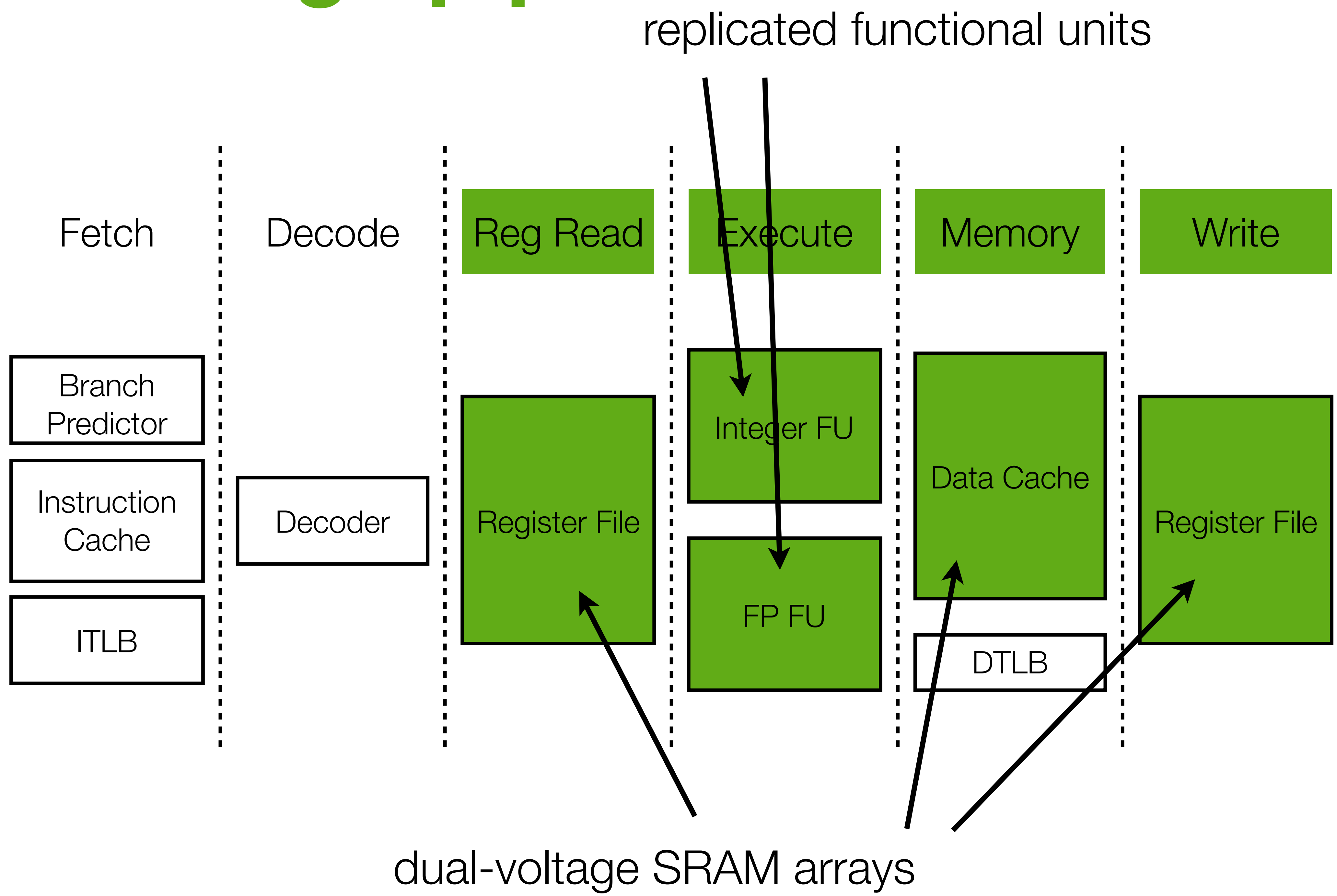
```
ld     0x04 r1
ld     0x08 r2
add.a  r1   r2    r3
st.a   0x0c r3
```

operations

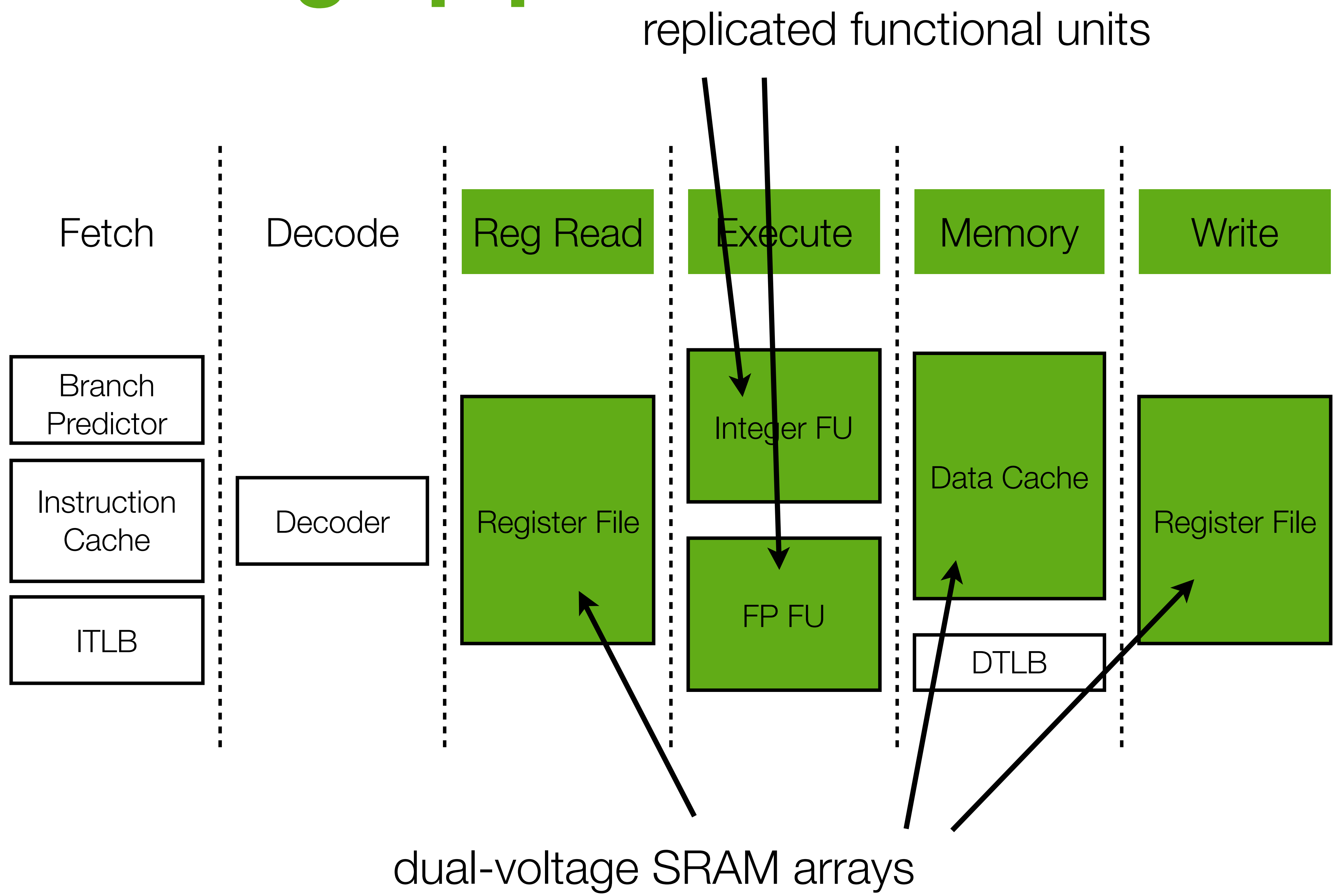
storage

registers
caches
main memory

Dual-voltage pipeline



Dual-voltage pipeline

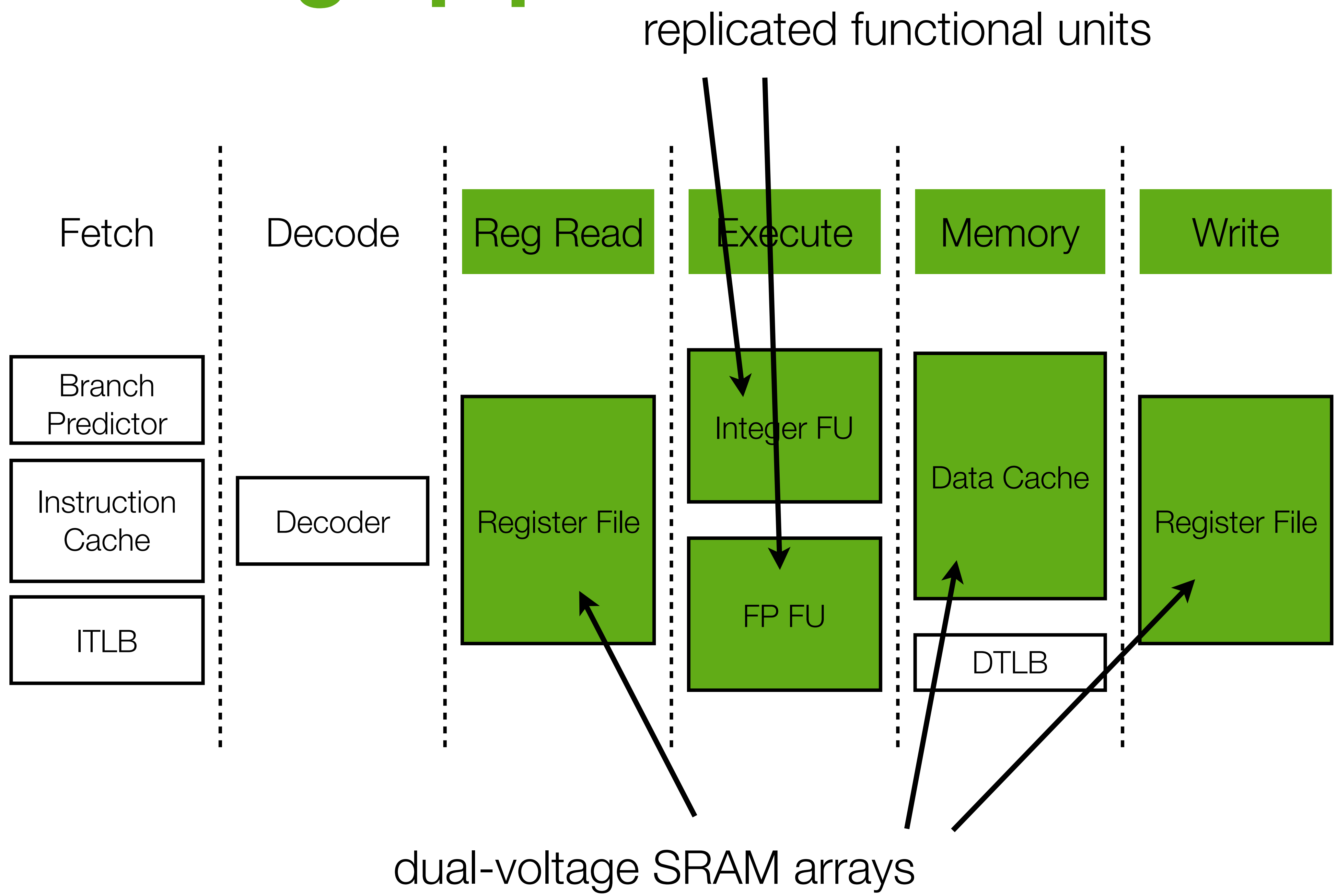


7–24% energy saved on average

(fft, game engines, raytracing, QR code readers, etc)

(scope: processor + memory)

Dual-voltage pipeline



7–24% energy saved on average

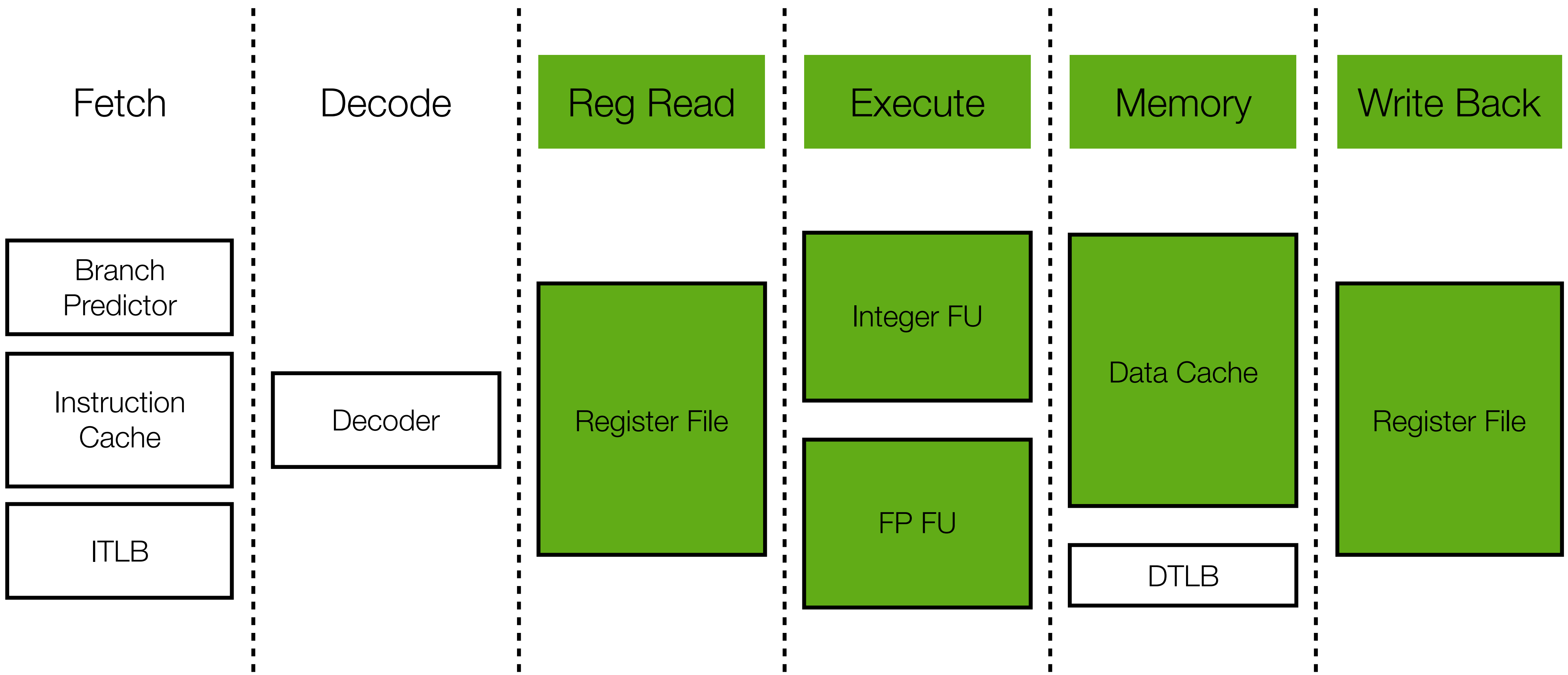
not good... :(

(though better implementations likely)

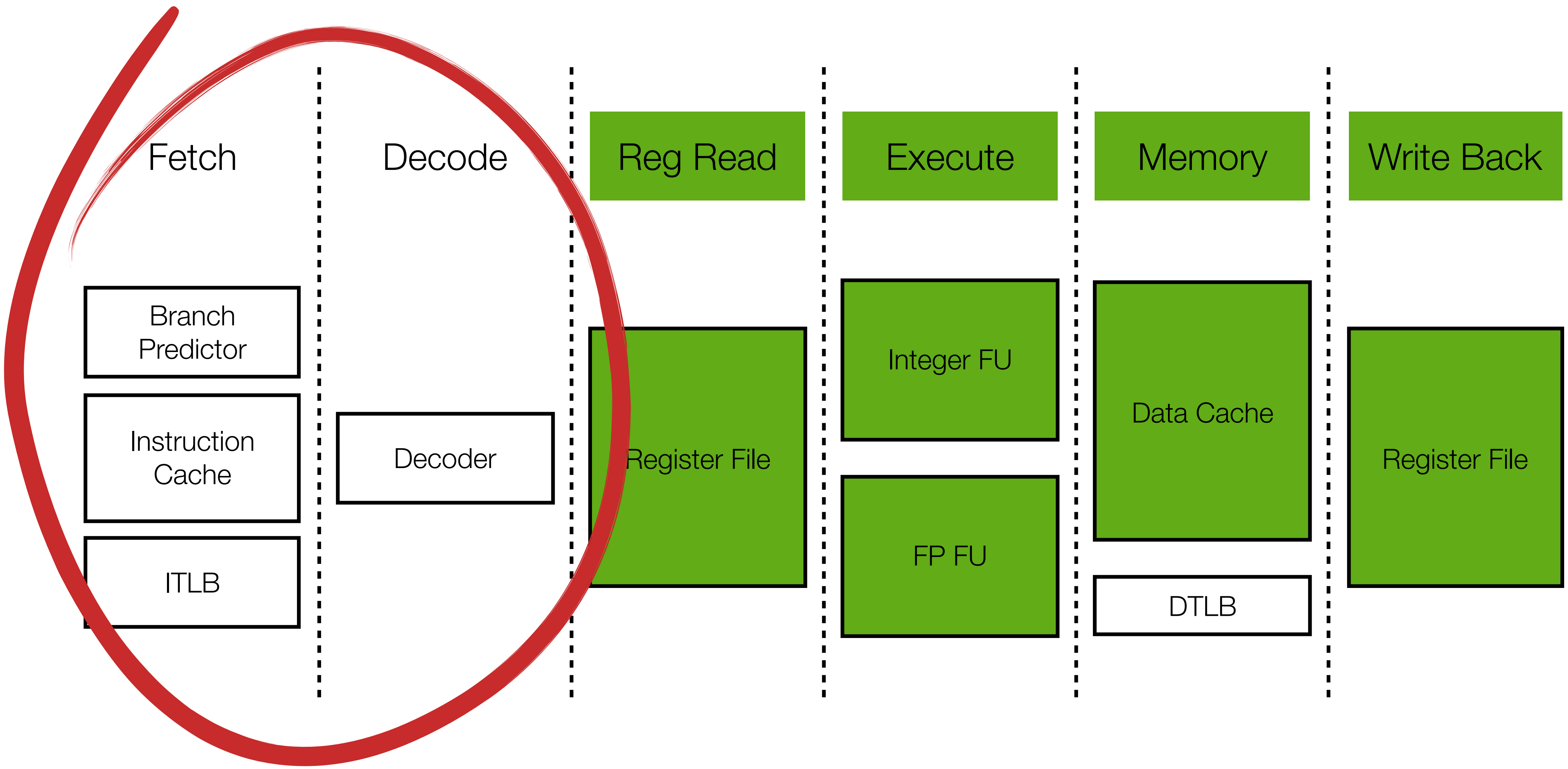
(fft, game engines, raytracing, QR code readers, etc)

(scope: processor + memory)

Amdahl's law... damn!



Amdahl's law... damn!

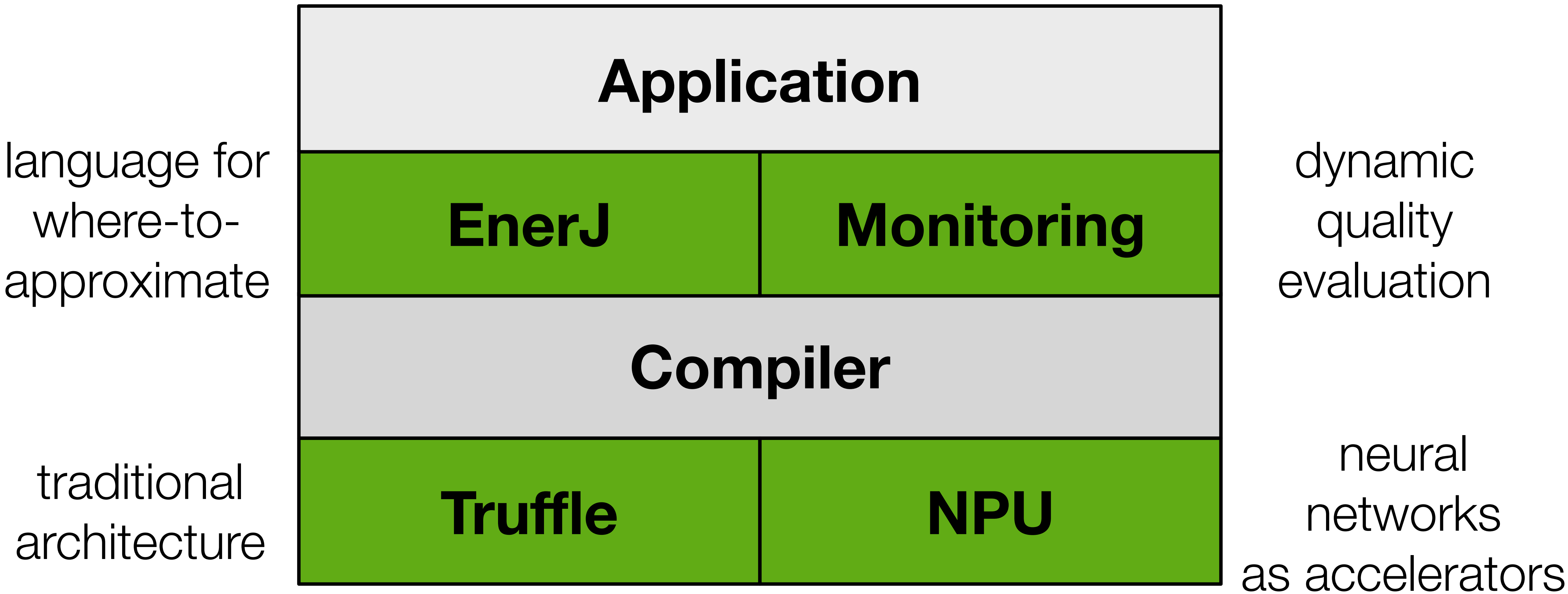


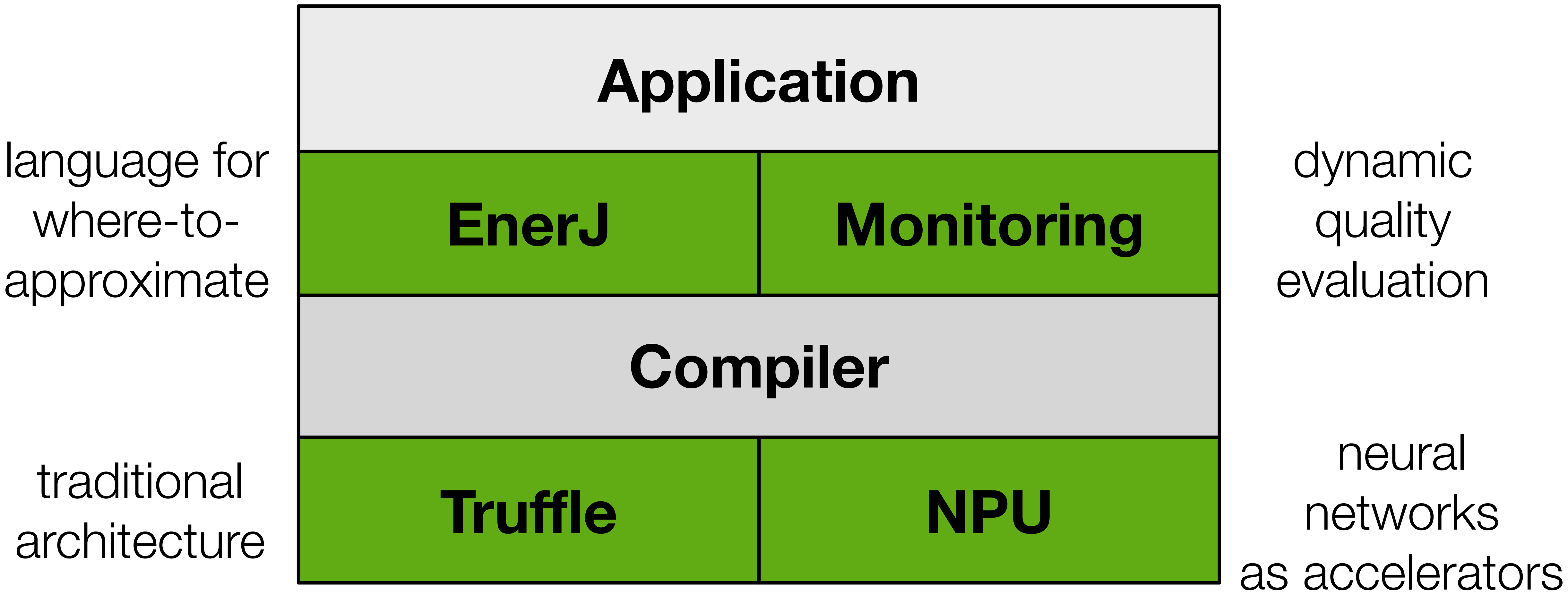
- Benefit limited to what can be approximated
- Instruction control can not be approximated

How can we get rid of exact instruction bookkeeping?

How can we get rid of exact instruction bookkeeping?

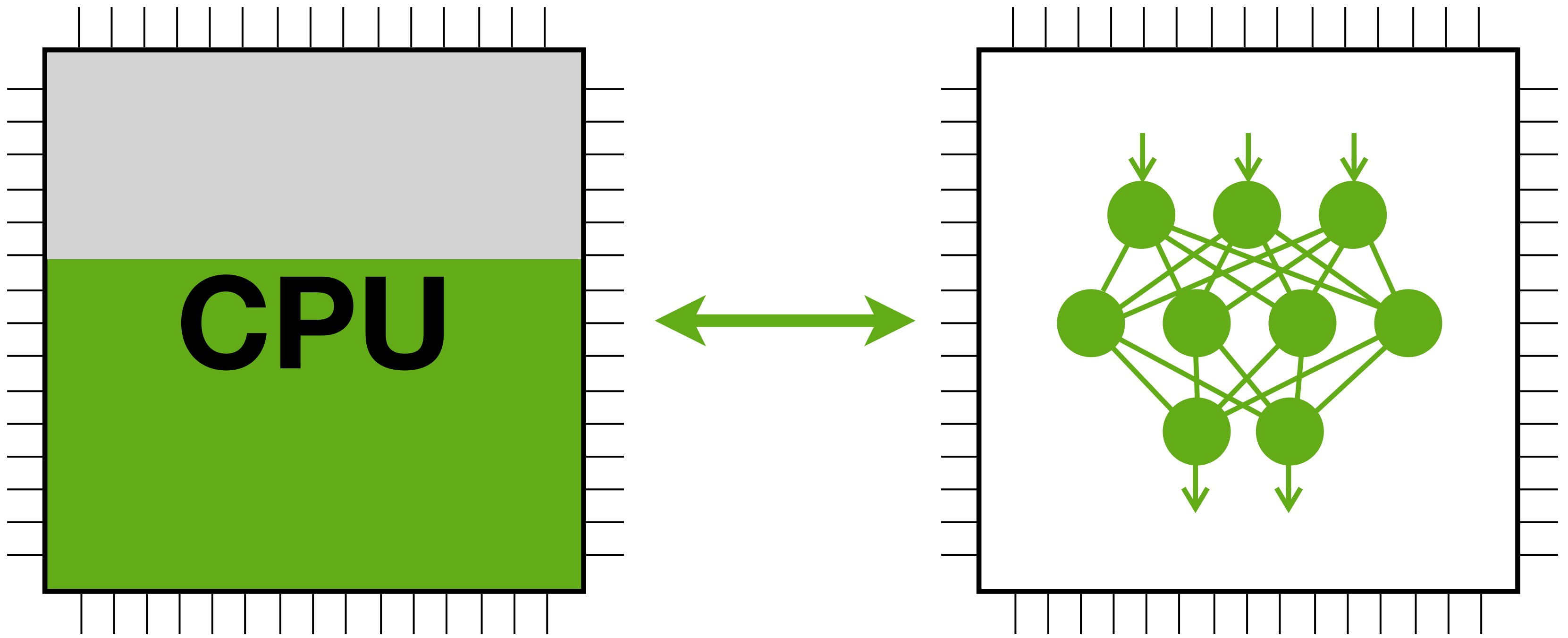
If behavior is approximate, why program it precisely?


language for
where-to-
approximate
traditional
architecture
Application
EnerJ
Monitoring
Compiler
Truffle
NPU
dynamic
quality
evaluation
neural
networks
as accelerators



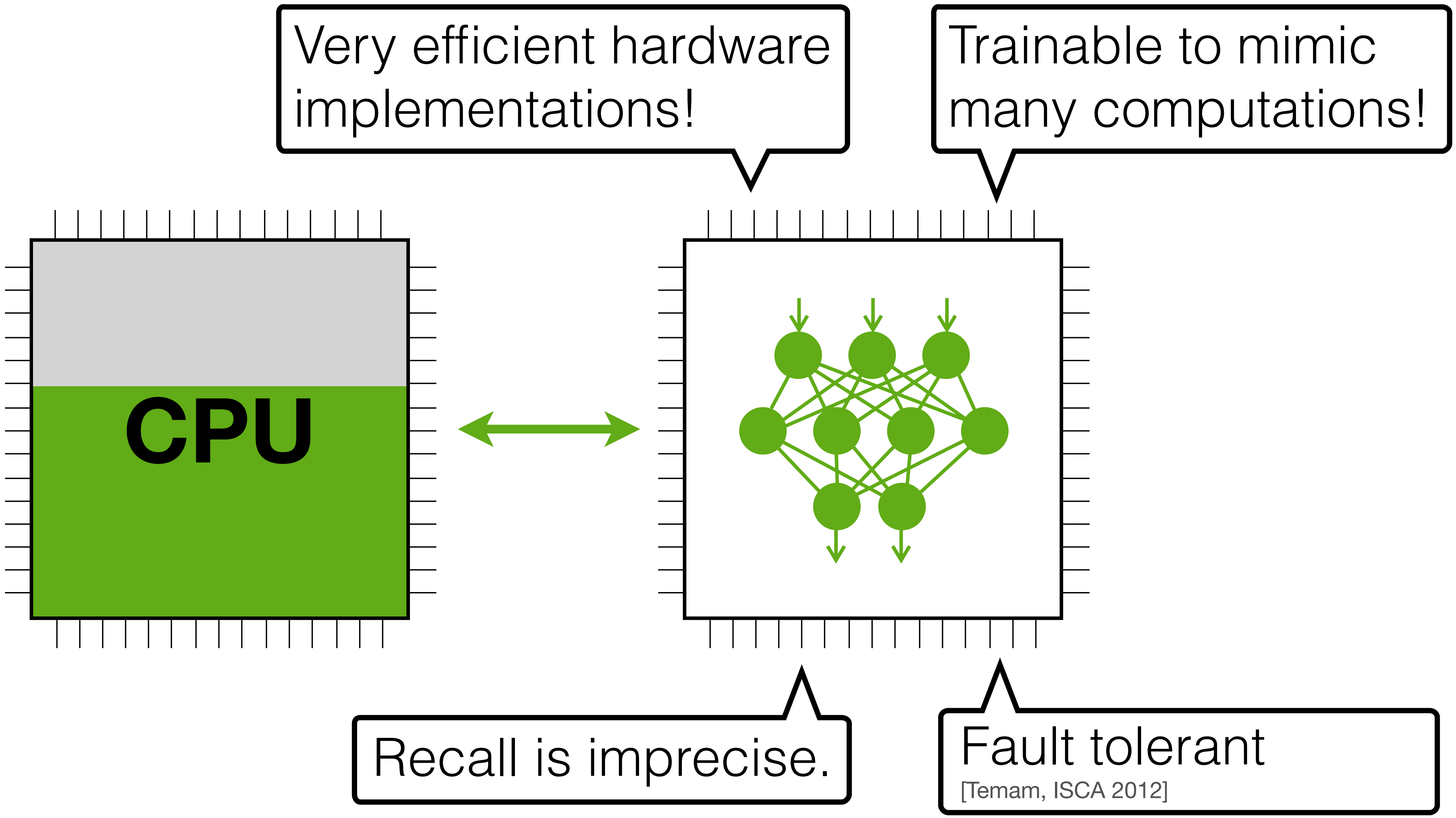
Application
language for where-to-approximate
EnerJ
Monitoring
dynamic quality evaluation
Compiler
traditional architecture
Truffle
NPU
neural networks as accelerators

Why Neural Networks as Approximate Accelerators?



Neural Acceleration of General-Purpose Approximate Programs, MICRO 2012
General-Purpose Code Acceleration with Limited-Precision Analog Computation, ISCA 2014

Why Neural Networks as Approximate Accelerators?



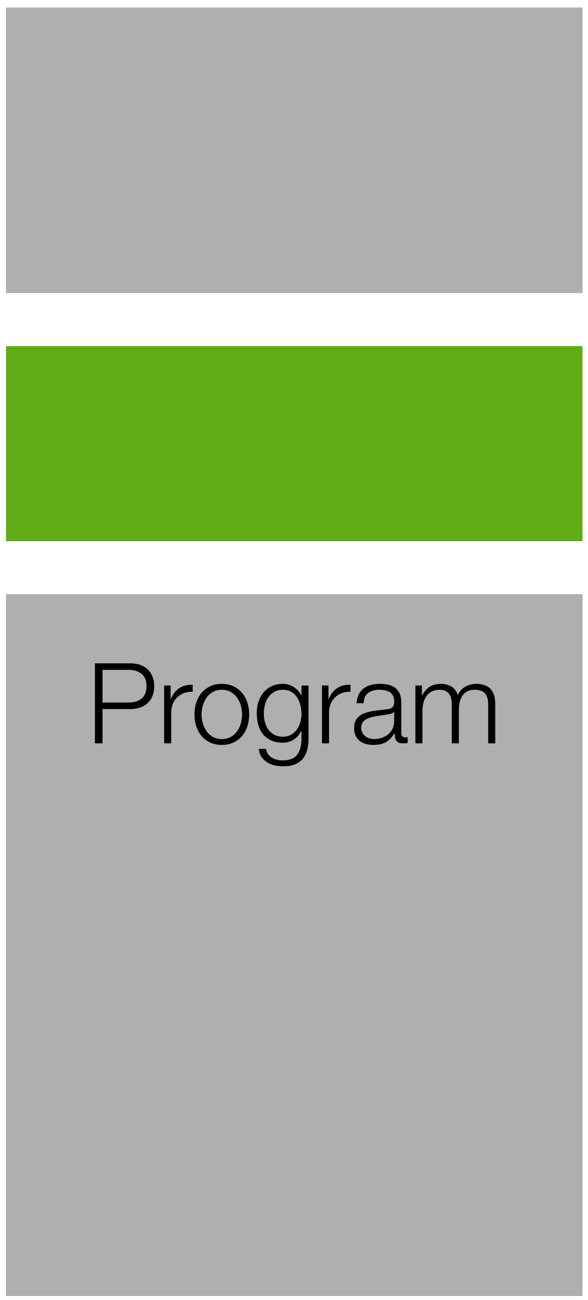
Neural Acceleration of General-Purpose Approximate Programs, MICRO 2012
General-Purpose Code Acceleration with Limited-Precision Analog Computation, ISCA 2014

Neural acceleration

Neural acceleration

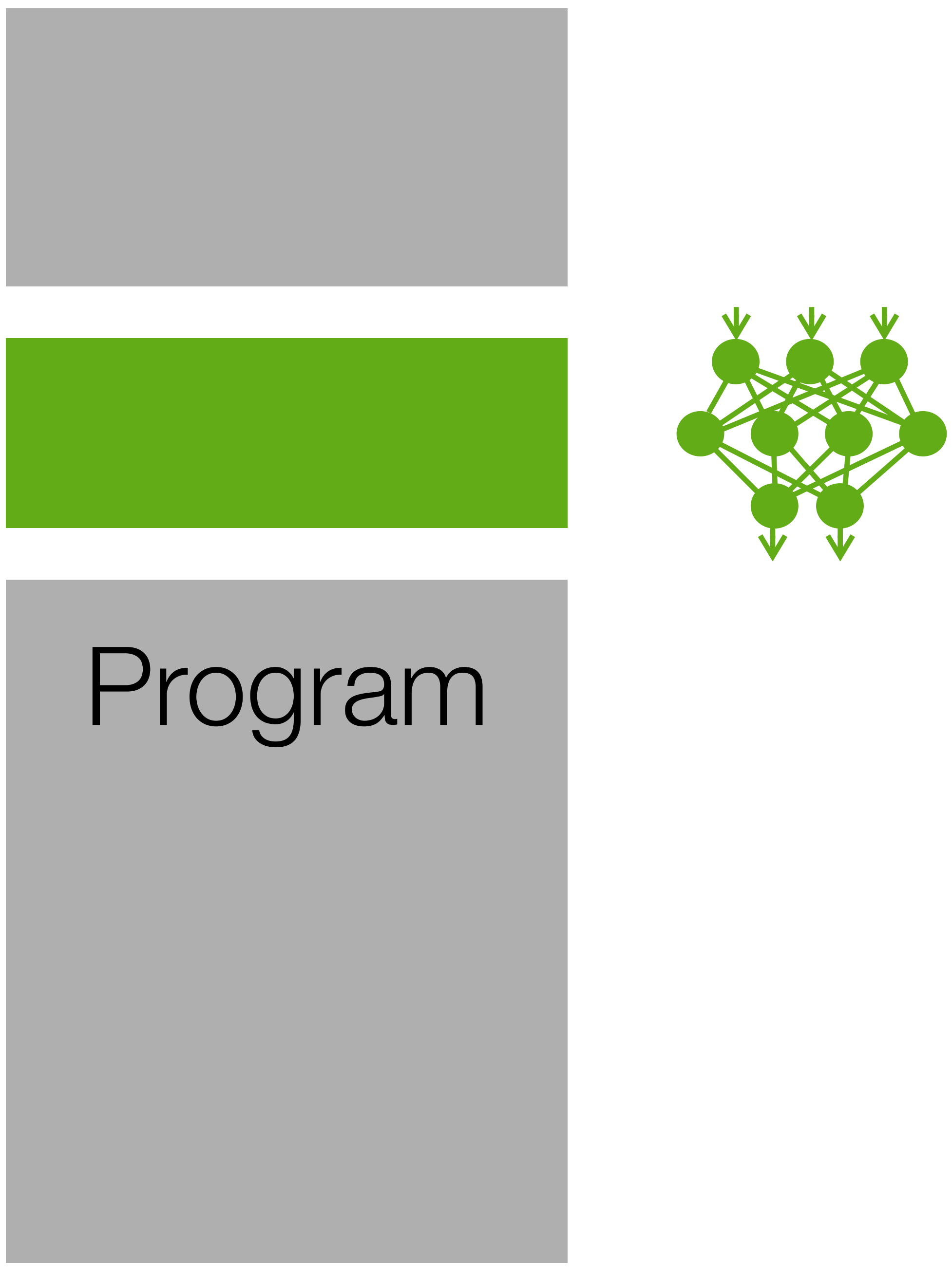
Find an approximate program component

Neural acceleration



Find an approximate program component

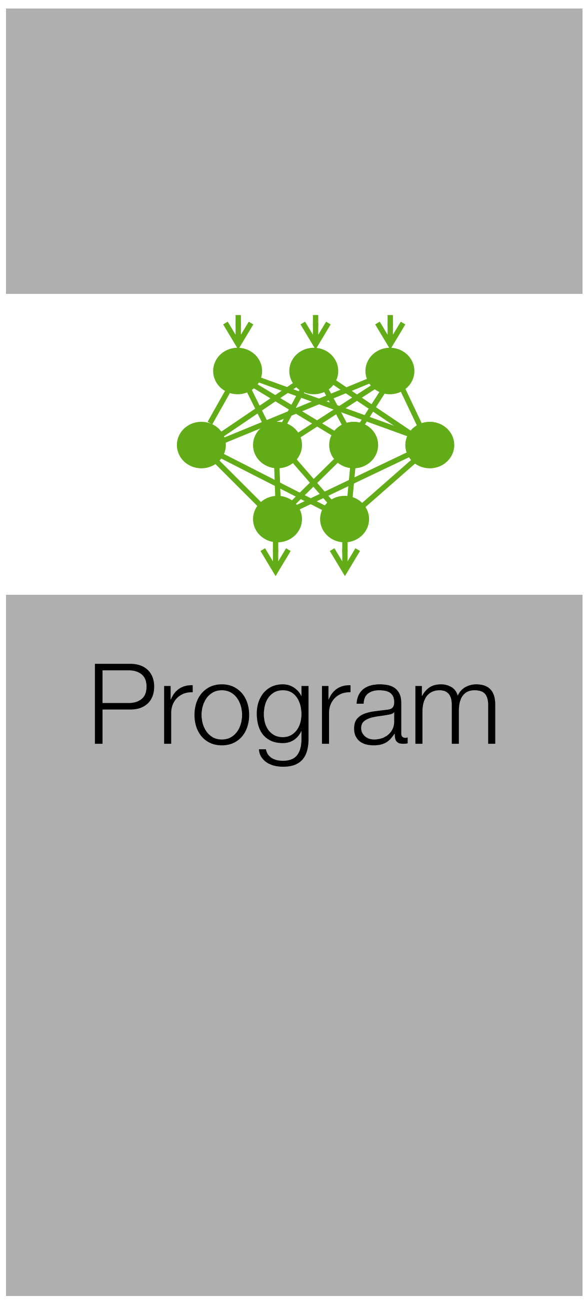
Neural acceleration



Find an approximate program component

Compile the program and train a neural network

Neural acceleration



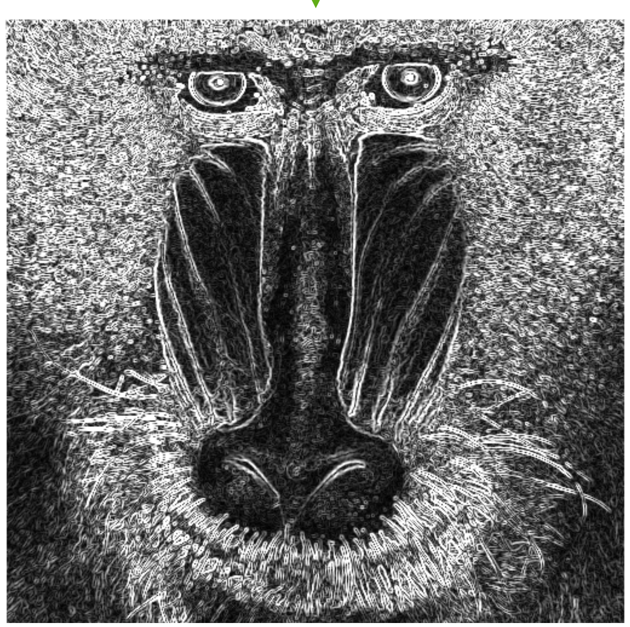
Find an approximate program component

Compile the program and train a neural network

Execute on a fast Neural Processing Unit (NPU)

An example: Sobel filter

edgeDetection()

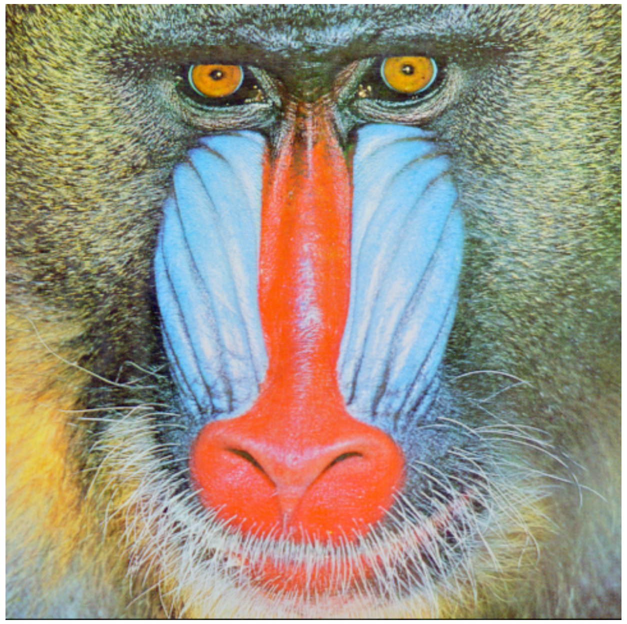


```
@approx float grad(approx float[3][3] p) {
  …
}
```

```
void edgeDetection(aImage &src,
                   aImage &dst) {
  for (int y = …) {
    for (int x = …) {
      dst[x][y] =
          grad(window(src, x, y));
    }
  }
}
```

```
@approx float dst[][];
```

An example: Sobel filter



edgeDetection()

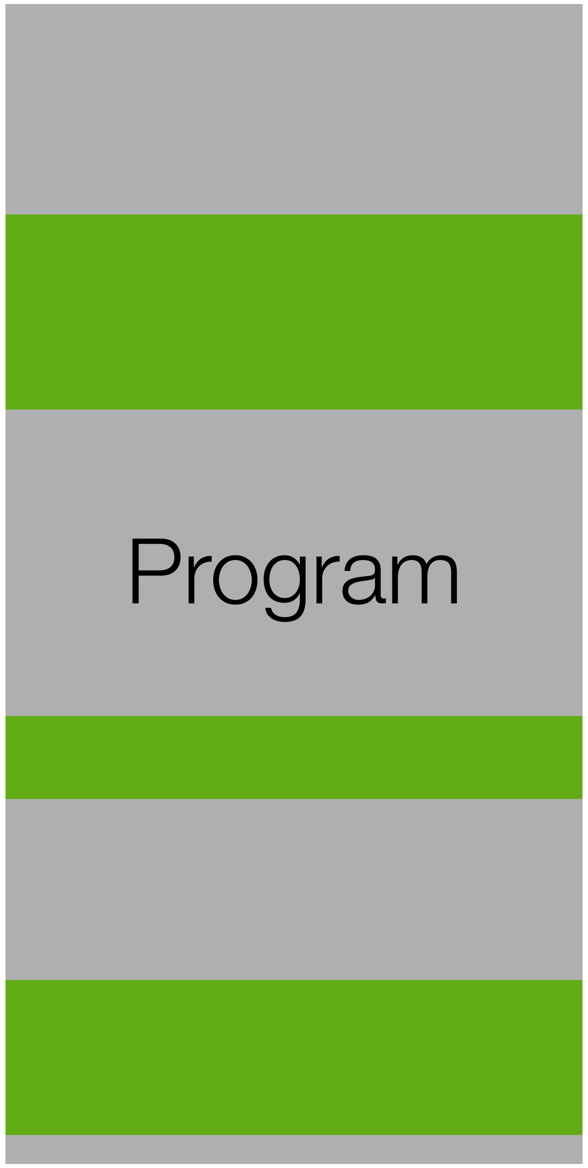
```
@approx float grad(approx float[3][3] p) {
   …
}
```

- ✓ Approximable
- ✓ Well-defined inputs and outputs

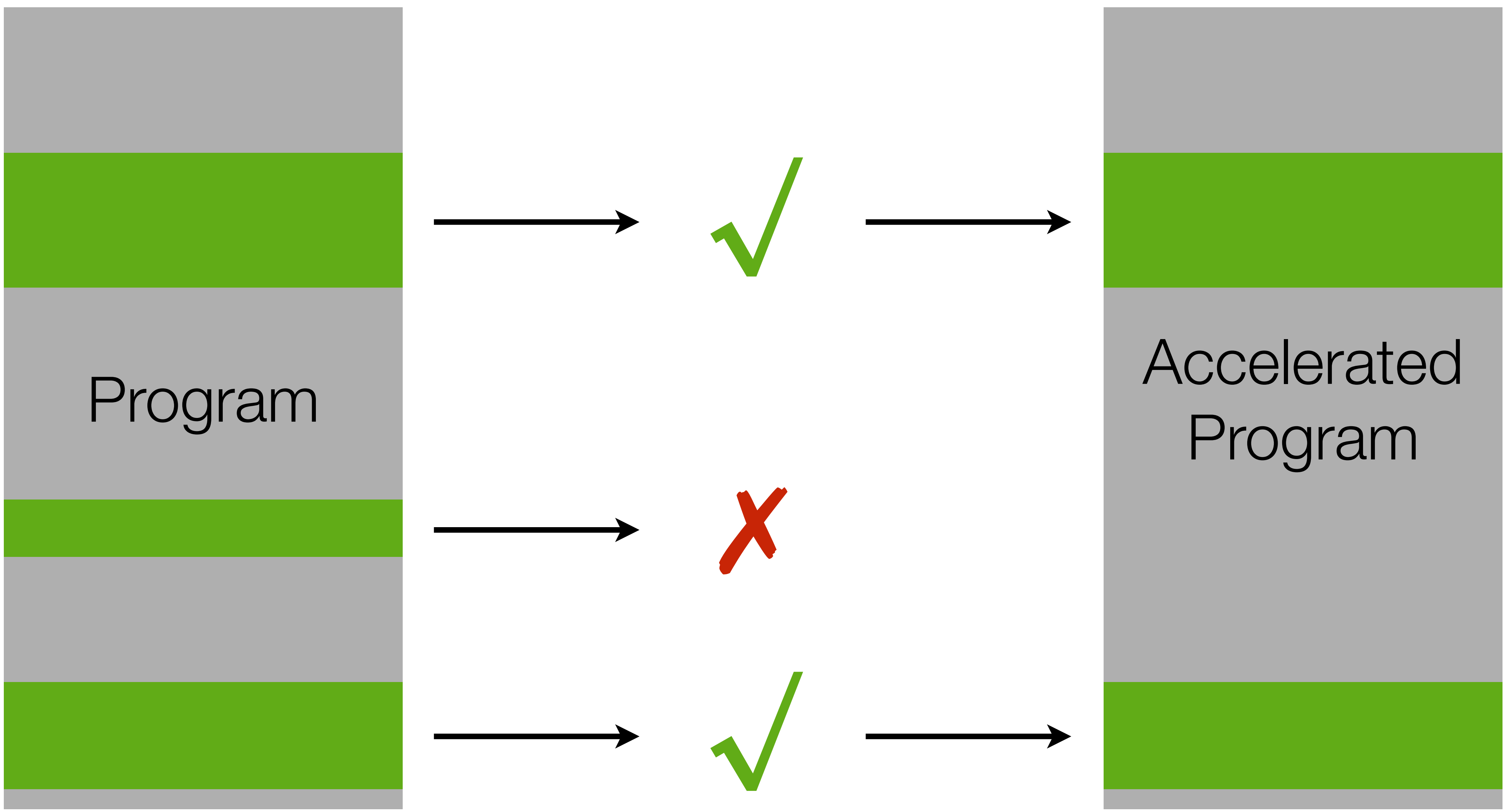
```
void edgeDetection(aImage &src,
                   aImage &dst) {
  for (int y = …) {
    for (int x = …) {
      dst[x][y] =
           grad(window(src, x, y));
    }
  }
}
```

```
@approx float dst[][];
```

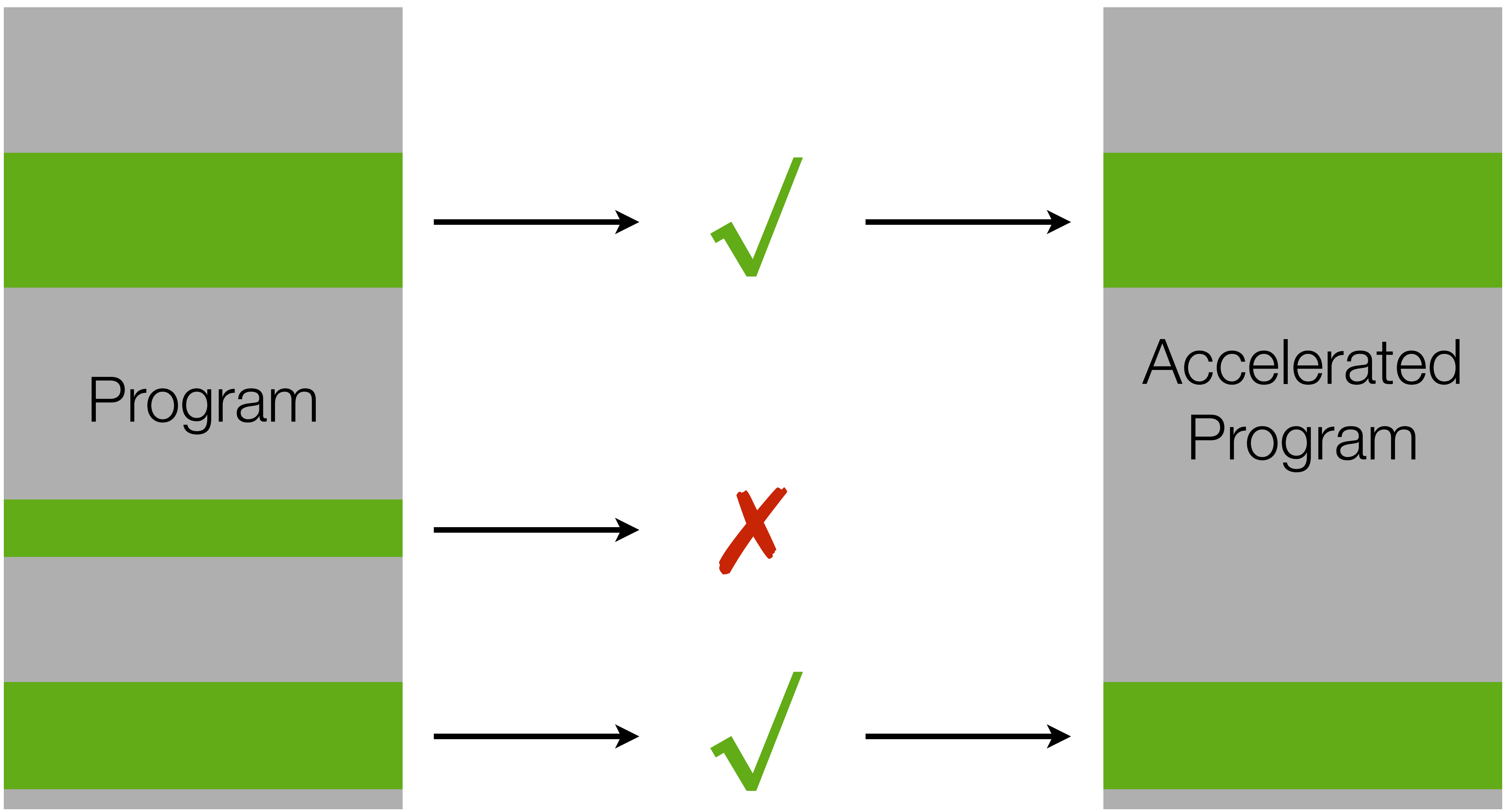
Empirically selecting target code



Empirically selecting target code

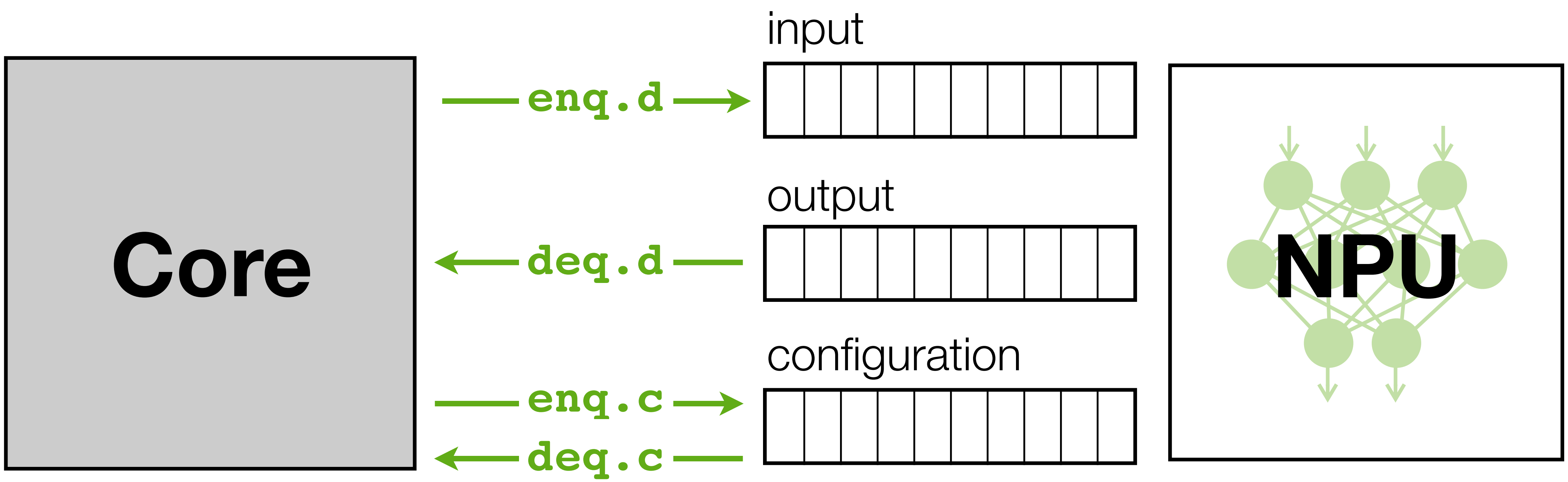


Empirically selecting target code

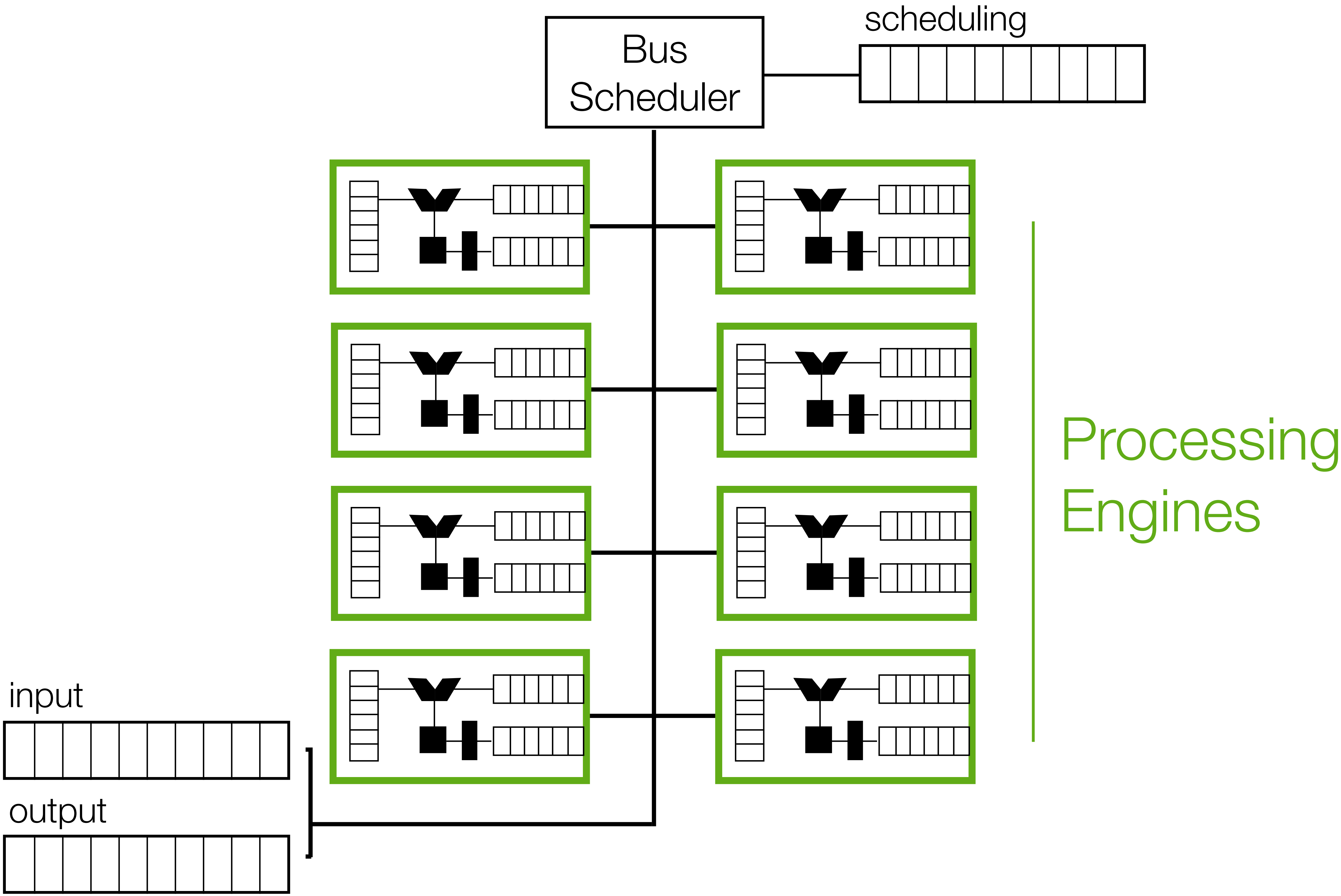


Each region of code leads to a different NN configuration.

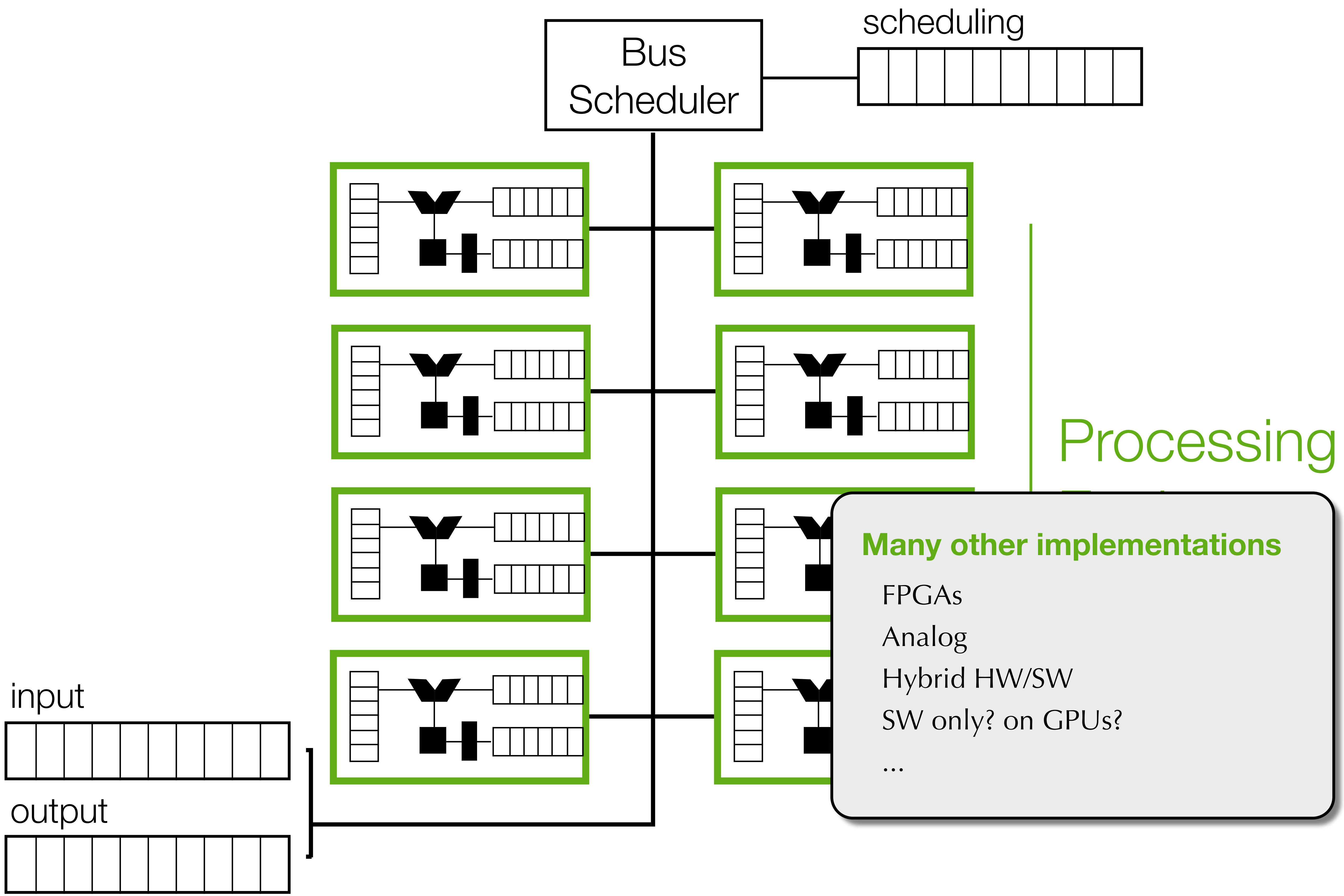
Neural Processing Unit



A digital NPU



A digital NPU



Many other implementations

FPGAs

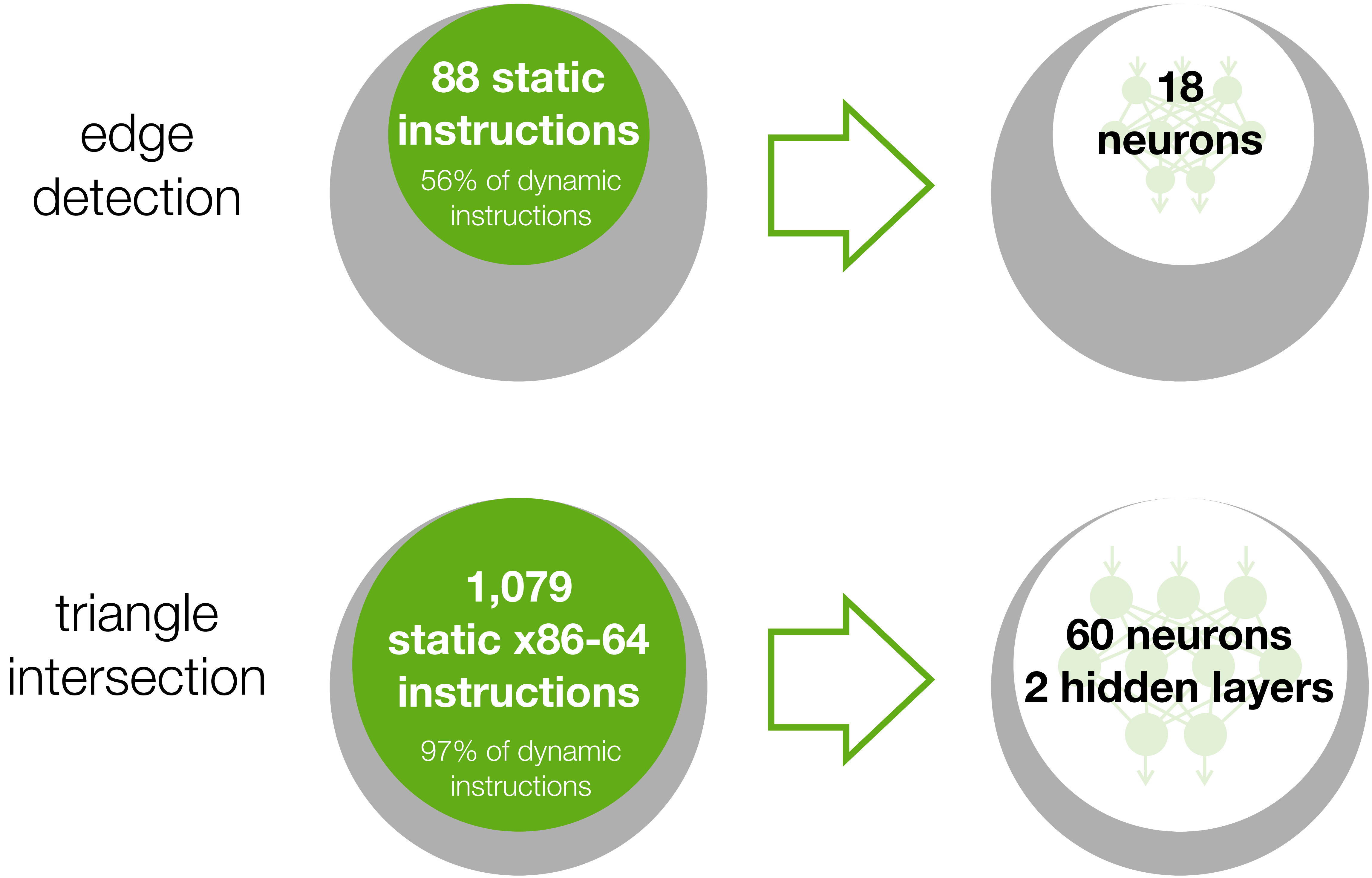
Analog

Hybrid HW/SW

SW only? on GPUs?

...

How do the NNs look like in practice?



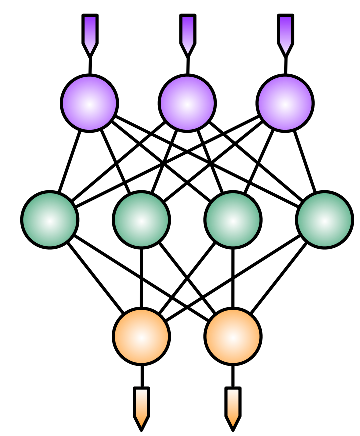
How do the NNs look like in practice?

edge detection

88 static instructions

56% of dynamic instructions

18 neurons



$$o_j = sigmoid(\sum_i w_{ji} x_{ji})$$

triangle intersection

1,079 static x86-64 instructions

97% of dynamic instructions

60 neurons
2 hidden layers

Summary of results

2.3x average speedup

Ranges from 0.8x to 11.1x

3.0x average energy reduction for digital, ~**10x** for analog

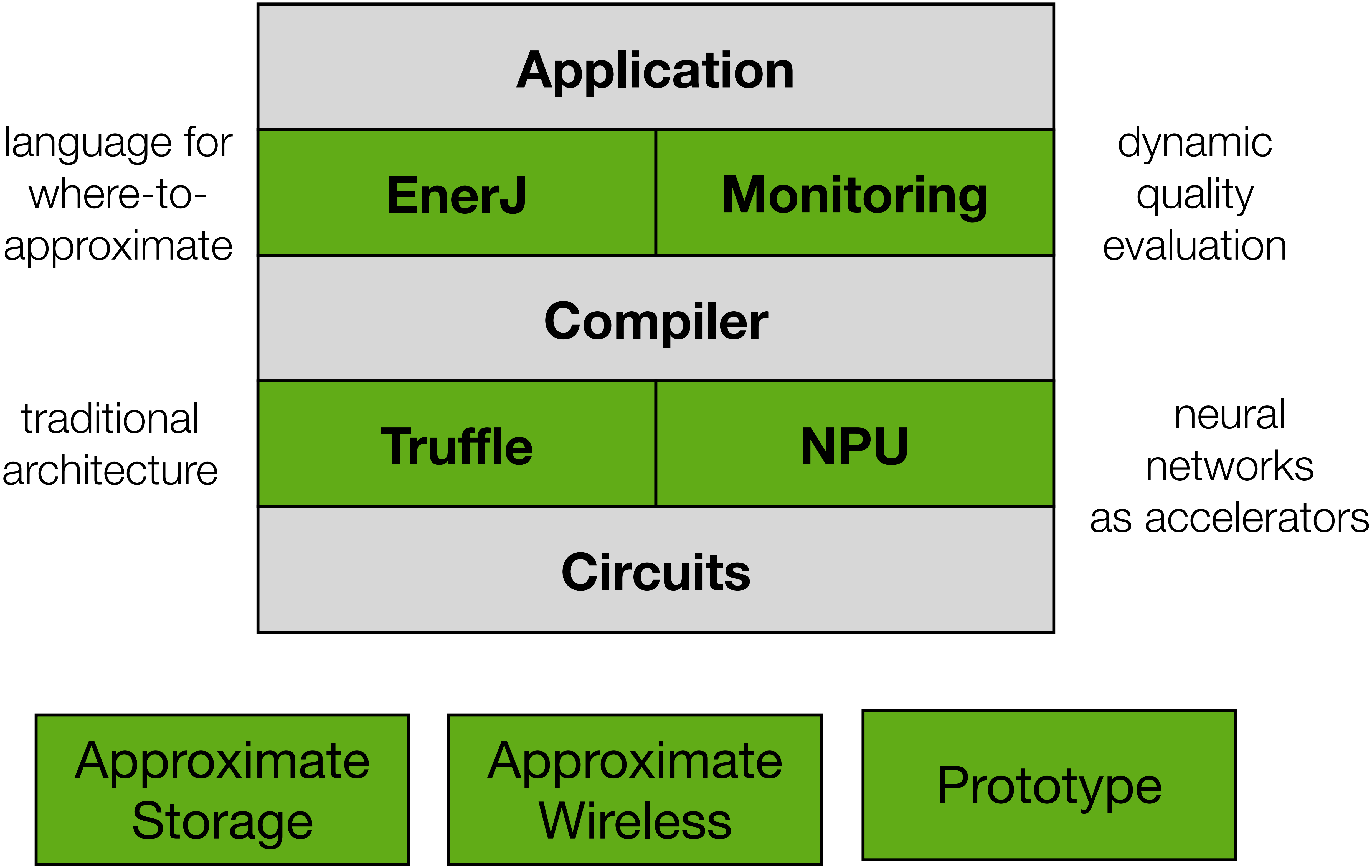
All benchmarks benefit

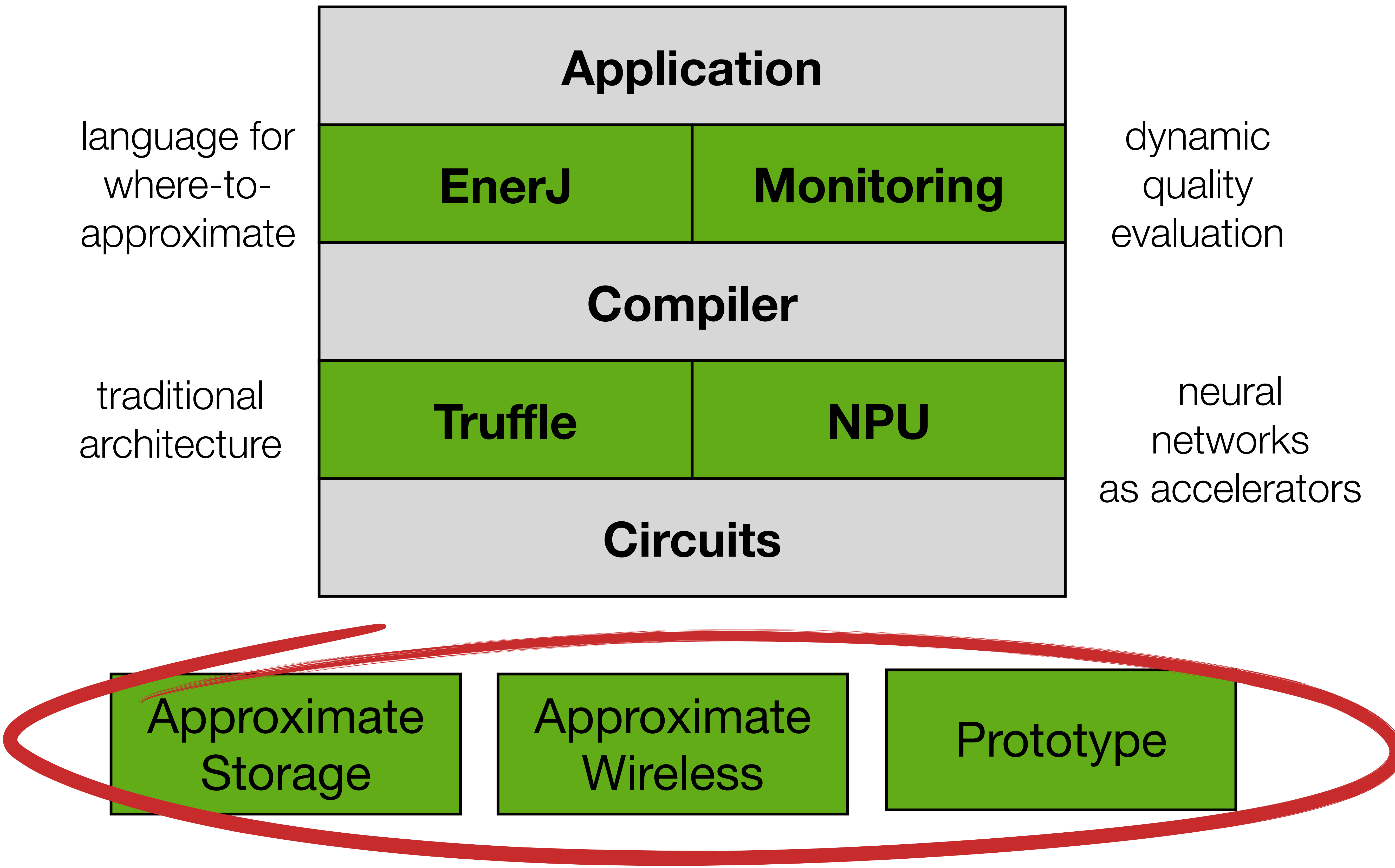
Quality loss below 10% in all cases

Based on application-specific quality metrics

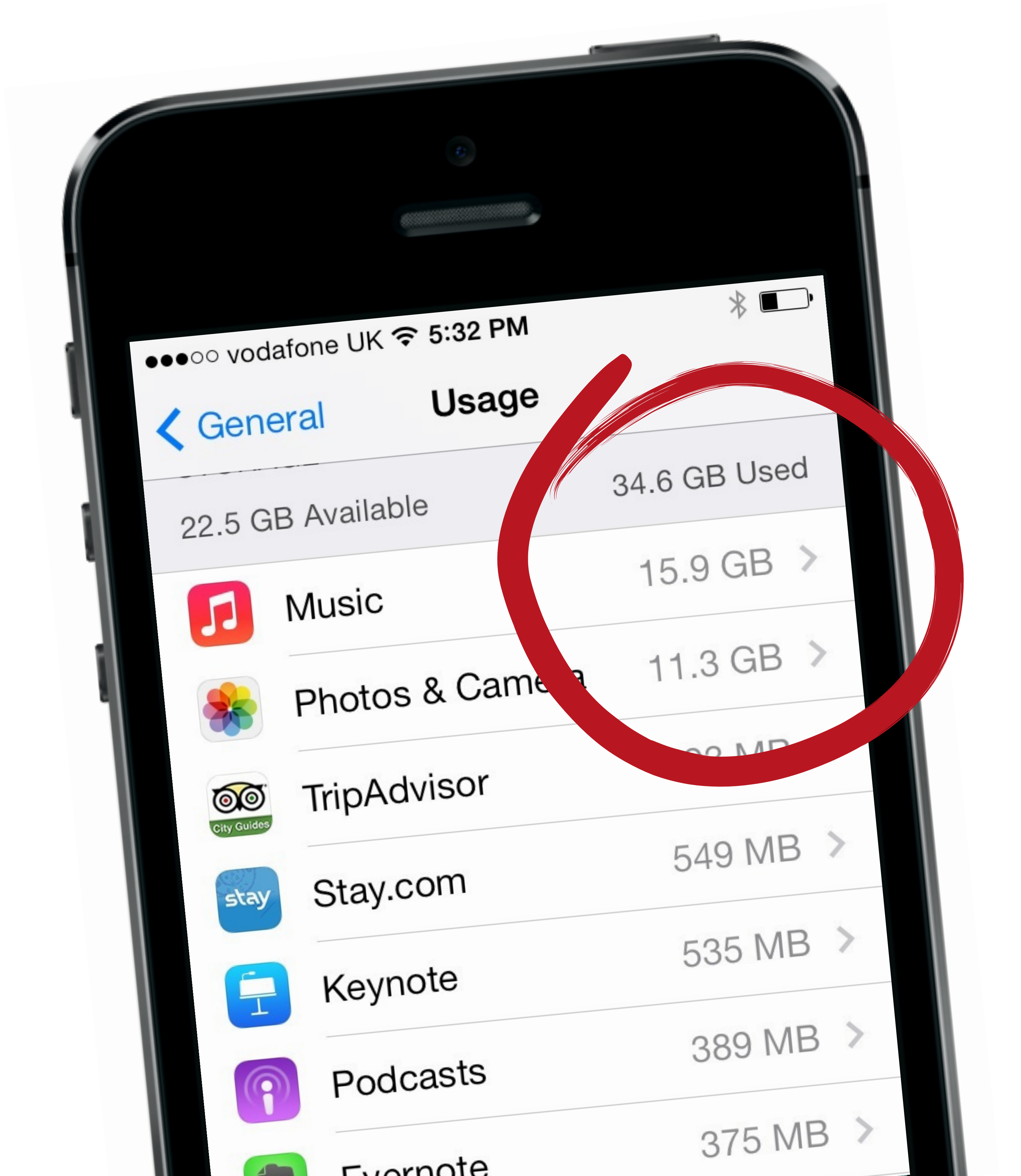
Just one possible design. Many others possible. Analog is where the big gains are likely (~10x+).

Key here is algorithmic transformation that enables new more efficient execution models.


Application
language for where-to-approximate
EnerJ
Monitoring
dynamic quality evaluation
Compiler
traditional architecture
Truffle
NPU
neural networks as accelerators
Circuits
Approximate Storage
Approximate Wireless
Prototype



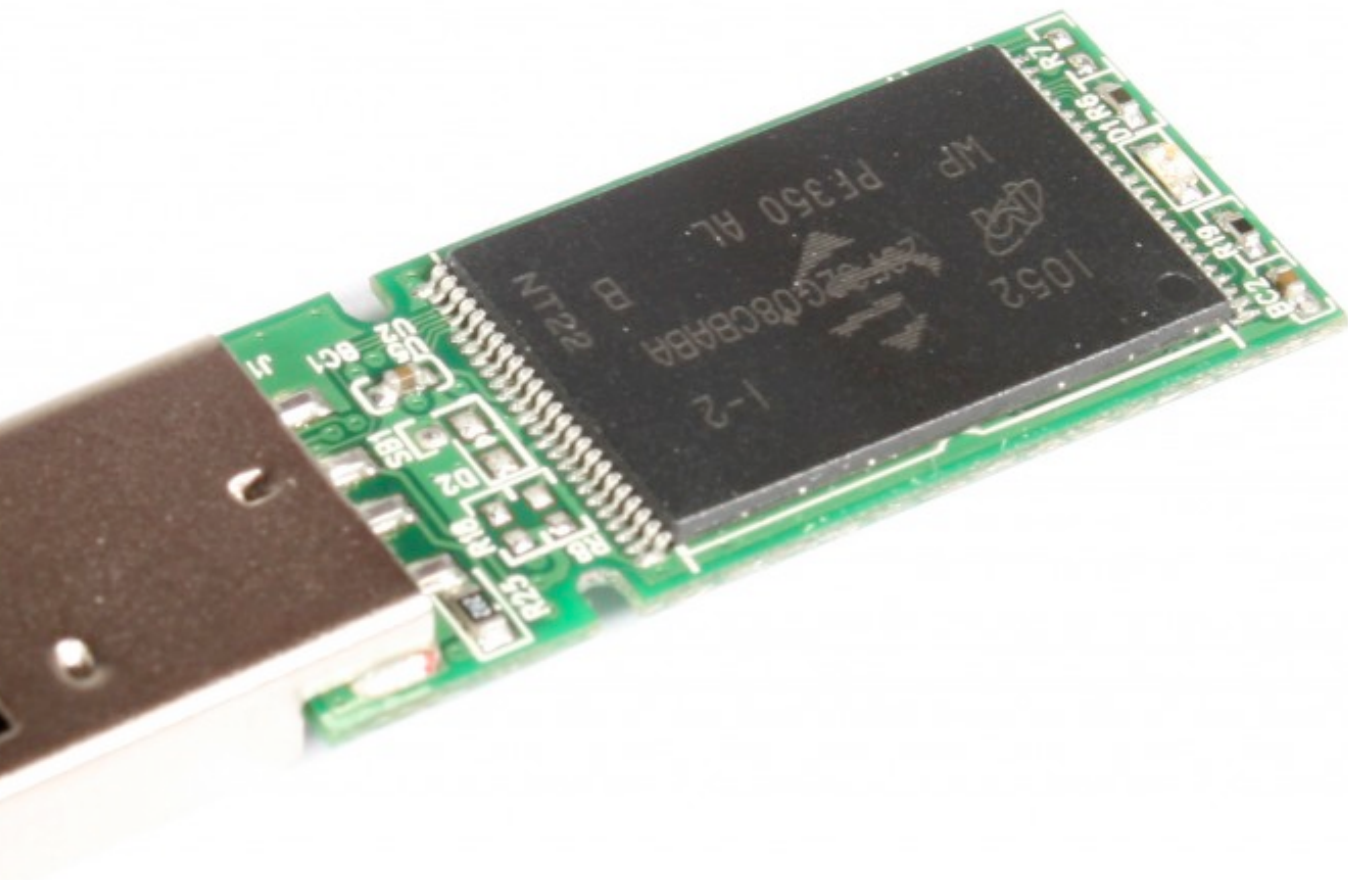
Application
language for where-to-approximate
EnerJ
Monitoring
dynamic quality evaluation
Compiler
traditional architecture
Truffle
NPU
neural networks as accelerators
Circuits
Approximate Storage
Approximate Wireless
Prototype



vodafone UK
5:32 PM
General
Usage
22.5 GB Available
34.6 GB Used
Music
15.9 GB
Photos & Came
11.3 GB
TripAdvisor
City Guides
549 MB
stay
Stay.com
535 MB
Keynote
389 MB
Podcasts
375 MB

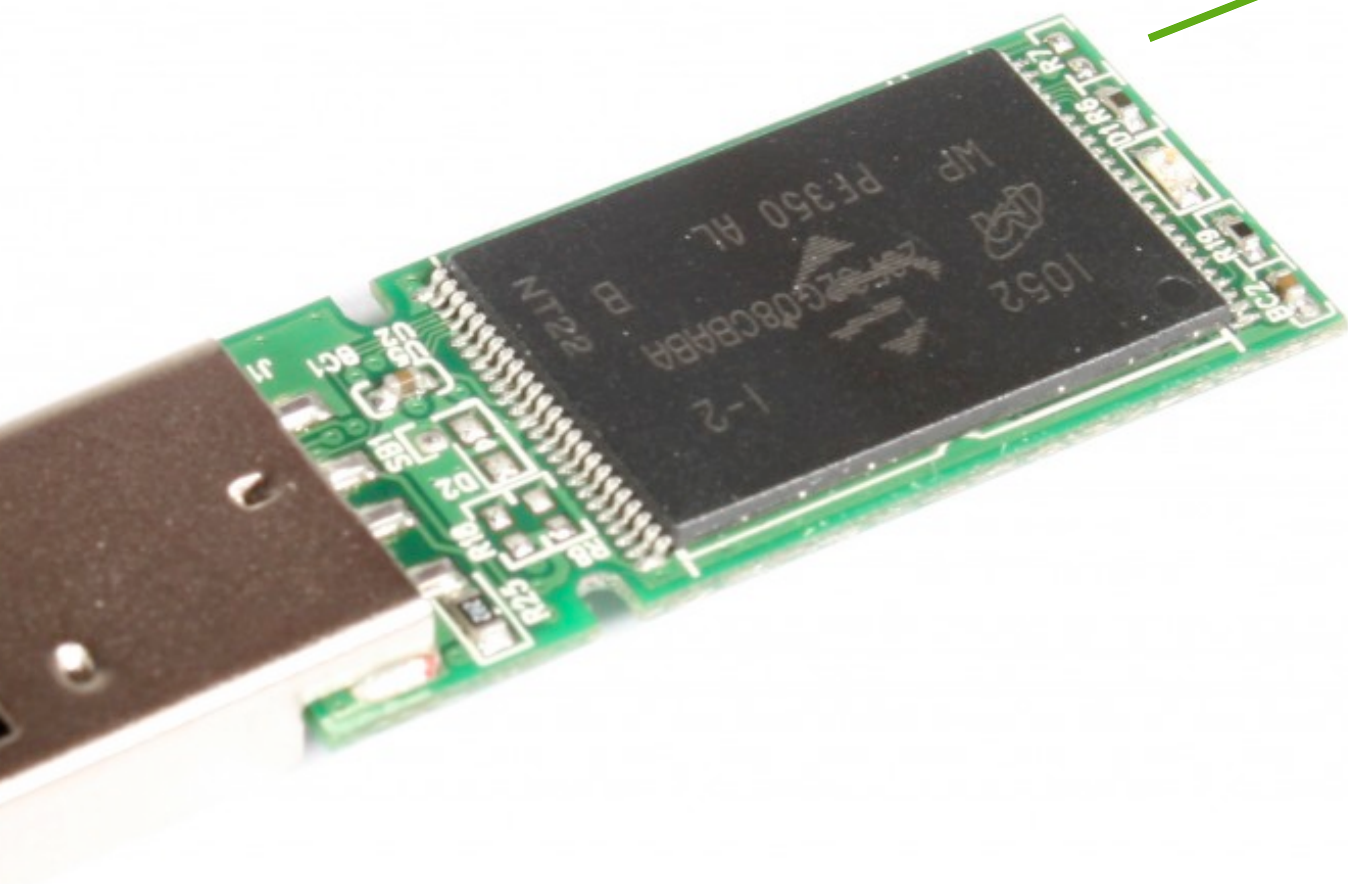
Approximate mass storage with Flash and PCM

Approximate Storage in Solid State Memories [MICRO'13]



Approximate mass storage with Flash and PCM

Approximate Storage in Solid State Memories [MICRO'13]

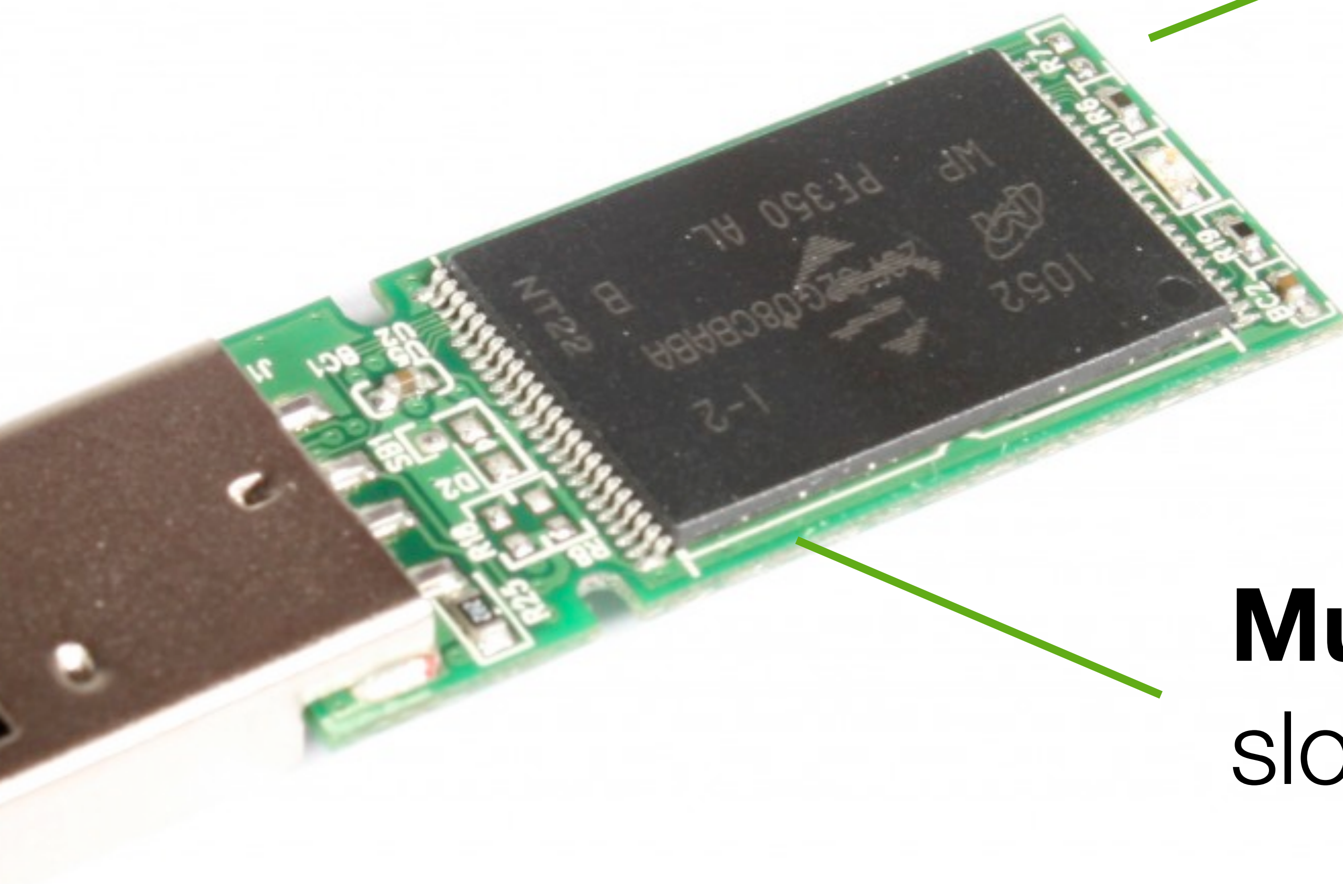


Cells **wear out** over time

Approximate mass storage with Flash and PCM

Approximate Storage in Solid State Memories [MICRO'13]

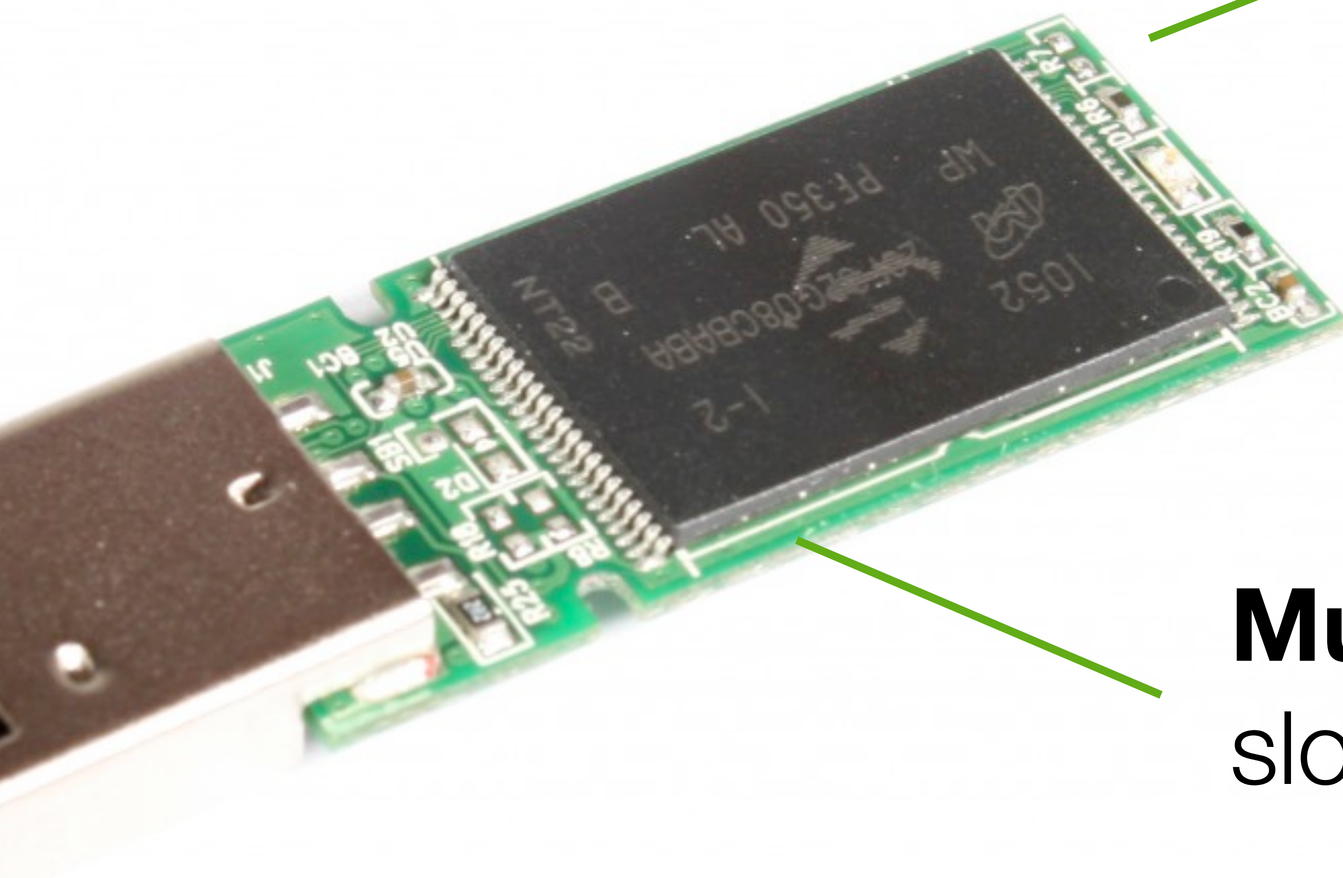
Cells **wear out** over time



Multi-level cells are slow or unreliable

Approximate mass storage with Flash and PCM

Approximate Storage in Solid State Memories [MICRO'13]



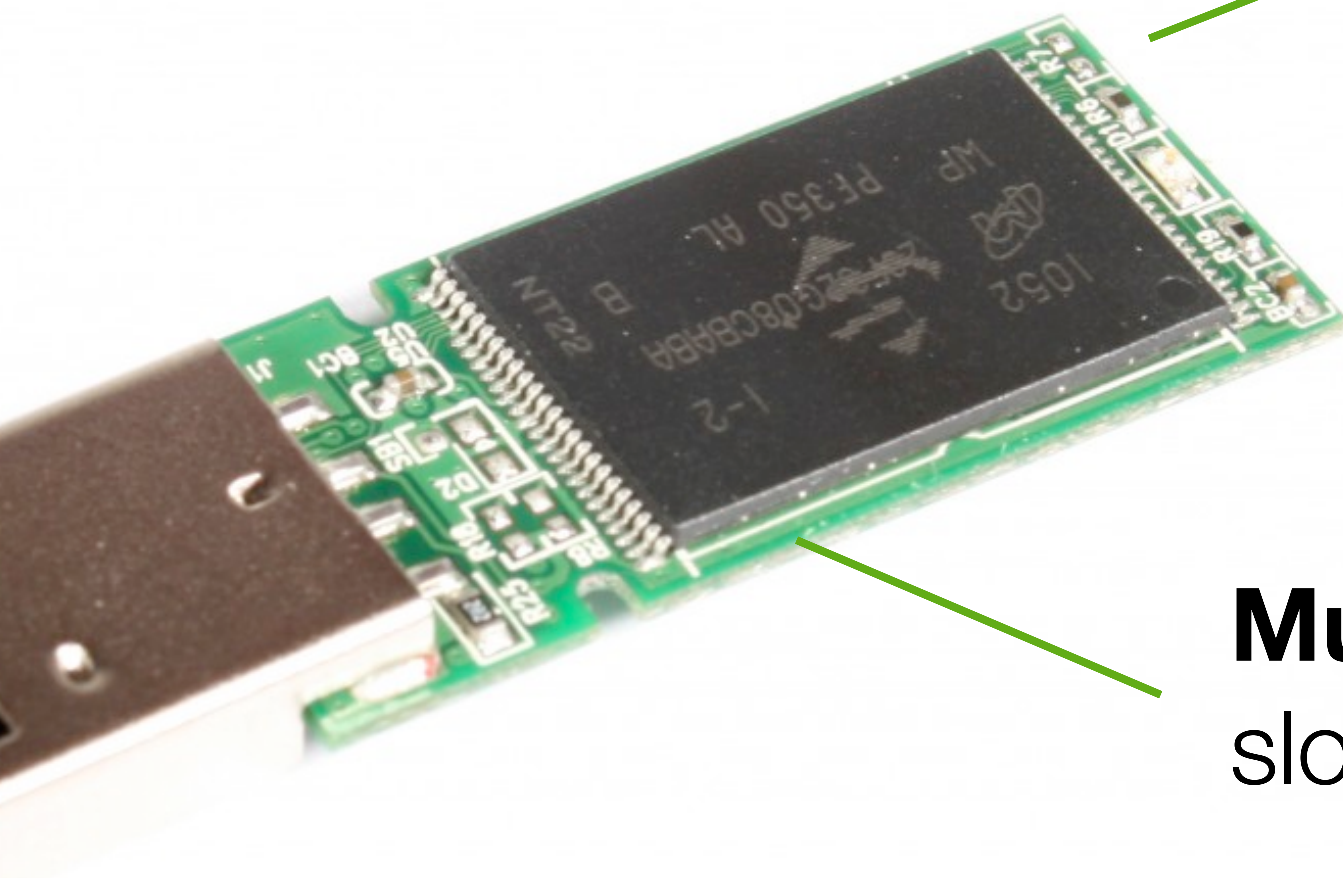
Cells **wear out** over time

Use **worn-out** memory for **approximate** data instead of throwing it away.

Multi-level cells are slow or unreliable

Approximate mass storage with Flash and PCM

Approximate Storage in Solid State Memories [MICRO'13]



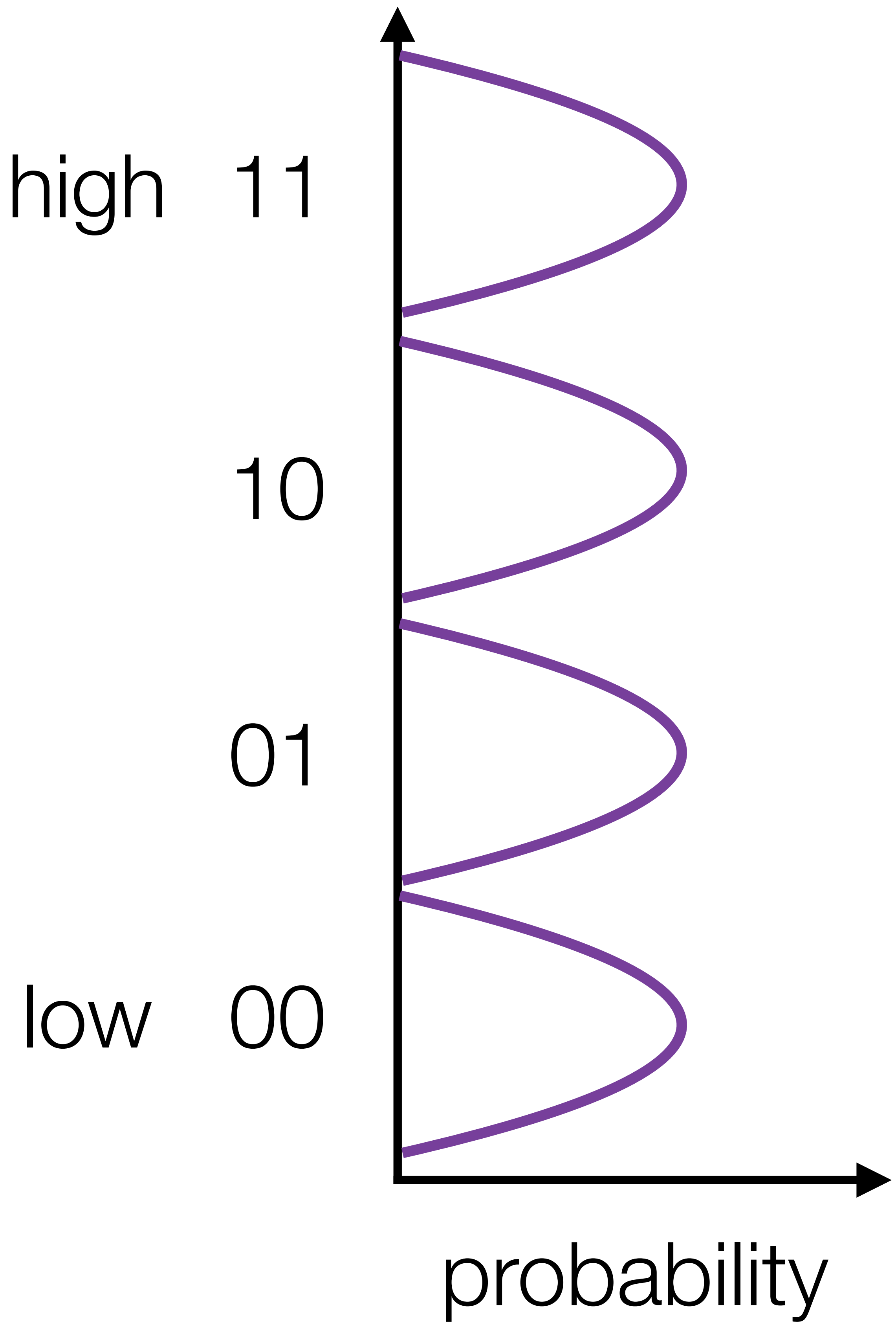
Cells **wear out** over time

Use **worn-out** memory for **approximate** data instead of throwing it away.

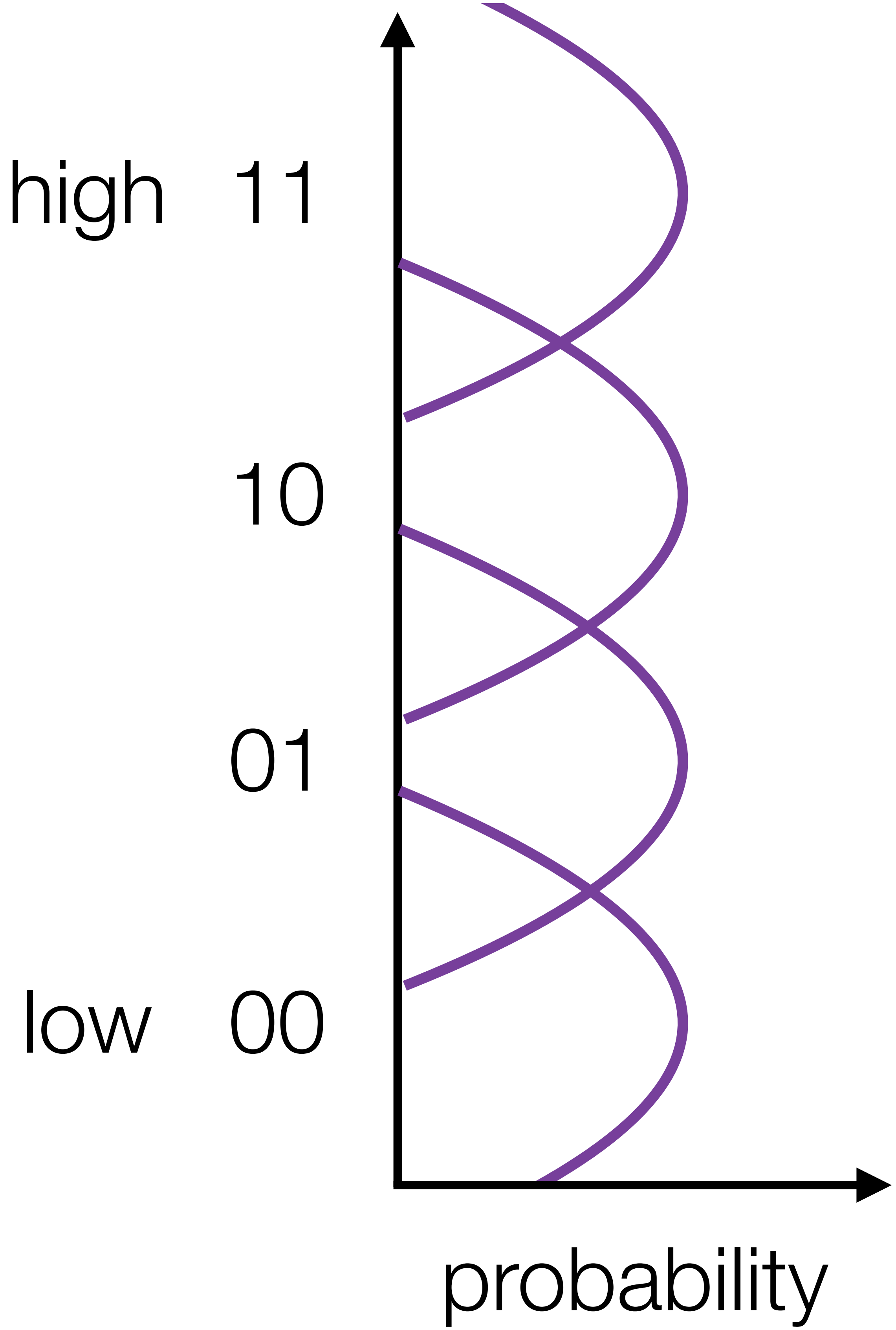
Multi-level cells are slow or unreliable

Trade off accuracy for performance/density in **multi-level cell** accesses.

Precise Multi-level Cells



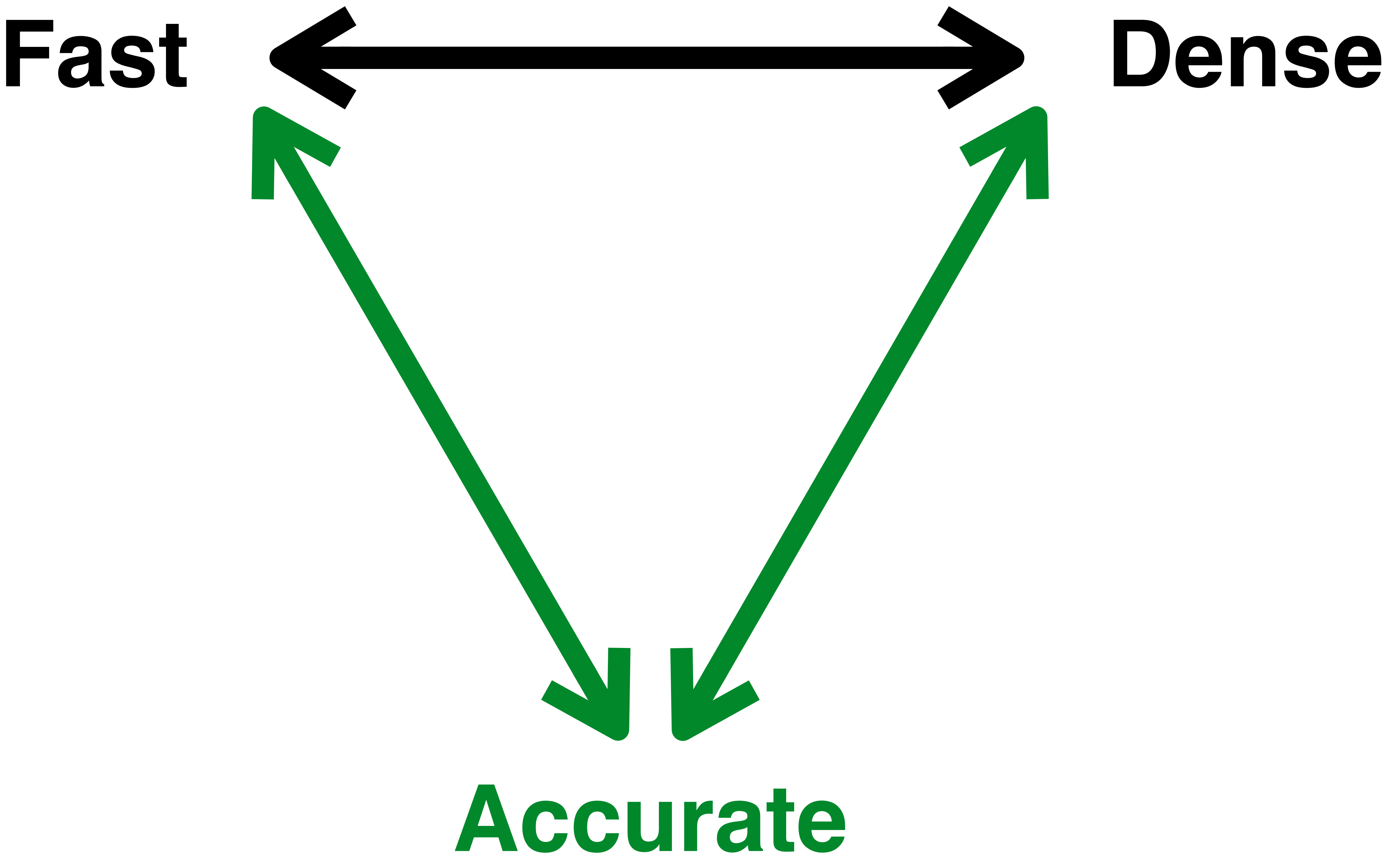
Approximate Multi-level Cells



Typical Trade-off in Multi-Level Cells

Fast ↔ **Dense**

Adding a New Trade-Off Axis



Approximate Wireless Communication


The New York Times – Bre ×

www.nytimes.com

m | blank | 29 c | Pin It | >r2k | Meal Train | wol | money | aed | umass | research | lang | wallace | cs up | OED | » | Other Bookmarks

HOME PAGE | TODAY'S PAPER | VIDEO | MOST POPULAR | U.S. Edition

Subscribe to Home Delivery | sielken... | Help

The New York Times

Thursday, April 11, 2013 Last Update: 12:32 PM ET

Search

Follow Us | Subscribe to Home Delivery | Personalize Your Weather

WORLD
U.S.
POLITICS
NEW YORK
BUSINESS
DEALBOOK
TECHNOLOGY
SPORTS
SCIENCE
HEALTH
ARTS
STYLE
OPINION

Autos
Blogs
Books
Cartoons
Classifieds
Crosswords
Dining & Wine
Education
Event Guide
Fashion & Style
Home & Garden
Jobs
Magazine
Movies
Music
Obituaries
Public Editor
Real Estate
Sunday Review

Senate Votes 68-31 to Allow Debate on Gun Legislation

By JENNIFER STEINHAUER
57 minutes ago

Lawmakers on Thursday thwarted a threatened filibuster, clearing the way for debate on the first piece of major gun control legislation to be considered in the Senate in decades.

Post a Comment | Read (10)

For Swing-State Democrats, Perils on Gun Control

By JONATHAN WEISMAN
8:58 AM ET

For years, guns have been an issue for Democrats from swing states to avoid. But the politics have changed.

Move to Widen Help for Syrian Rebels Gains Speed in West

By MICHAEL R. GORDON and MARK LANDLER

The United States is poised to increase its nonlethal aid to

Doug Mills/The New York Times

Family members of victims of gun violence gathered at the Capitol.

THE LEDE

Latest Updates on the Gun Debate

By JENNIFER PRESTON 7 minutes ago

While the vote Thursday signaled a victory for advocates of gun control, the legislation's fate remained uncertain.

- Loopholes Allow Buyers to Skirt Checks

N.H.L. Announces Effort in Support of Gay Athletes

By JEFF Z. KLEIN 9 minutes ago

The hockey league has formed a partnership with an advocacy group pledged to fight homophobia in sports.

South Korea Moves to Defuse Tensions With North

By CHOE SANG-HUN

The Opinion Pages

'Accidental Racist' and Lyrical Provocation

A song has raised hackles. Should popular culture have such power?

ROOM for DEBATE

THOMAS B. EDSALL

In Campaigns, Do You Get What You Pay For?

Strategists don't like what they see in the campaign money landscape.

- Editorial: The President's Budget
- Collins: Yipes, It's Congress on the Move
- Blow: Rand Paul Goes to Howard
- Yu: In China, Feudal Fixes for Modern Problems
- Op-Ed: Child Migrants, Alone in Court
- Fixes: The Power of Talking to Your Baby

MARKETS » At 12:35 PM ET

S.&P. 500	Dow	Nasdaq
1,594.05	14,868.09	3,298.12
+6.32	+65.85	+0.87
+0.40%	+0.44%	+0.03%

GET QUOTES | My Portfolios »

Stock, ETFs, Funds | Go

THURSDAY STYLES »

Picky, Picky: A Critic's Spring Fashion Guide

Cathy Horyn's practical, and ornery, guide to spring shopping (but don't hold her to it).

HOME & GARDEN »

Q&A

A Creative Mind in Motion

Ferruccio Laviani on his creative process and his latest work for Fratelli Boffi.

Approximate Wireless Communication


The New York Times – Bre

www.nytimes.com

m | blank | c | Pin It | >r2k | Meal Train | wol | money | aed | umass | research | lang | wallace | cs up | OED | Other Bookmarks

HOME PAGE | TODAY'S PAPER | VIDEO | MOST POPULAR | U.S. Edition

Subscribe to Home Delivery | sielken... | Help

The New York Times

Thursday, April 11, 2013 Last Update: 12:32 PM ET

Search

Subscribe to Home Delivery | Personalize Your Weather

WORLD
U.S.
POLITICS
NEW YORK
BUSINESS
DEALBOOK
TECHNOLOGY
SPORTS
SCIENCE
HEALTH
ARTS
STYLE
OPINION

Autos
Blogs
Books
Cartoons
Classifieds
Crosswords
Dining & Wine
Education
Event Guide
Fashion & Style
Home & Garden
Jobs
Magazine
Movies
Music
Obituaries
Public Editor
Real Estate
Sunday Review

Senate Votes 68-31 to Allow Debate on Gun Legislation
By JENNIFER STEINHAUER
57 minutes ago
Lawmakers on Thursday thwarted a threatened filibuster, clearing the way for debate on the first piece of major gun control legislation to be considered in the Senate in decades.
Post a Comment | Read (10)

Doug Mills/The New York Times
Family members of victims of gun violence gathered at the Capitol.

For Swing-State Democrats, Perils on Gun Control
By JONATHAN WEISMAN
8:58 AM ET
For years, guns have been an issue for Democrats from swing states to avoid. But the politics have changed.

THE LEDE
Latest Updates on the Gun Debate
By JENNIFER PRESTON 7 minutes ago
While the vote Thursday signaled a victory for advocates of gun control, the legislation's fate remained uncertain.
• Loopholes Allow Buyers to Skirt Checks

Move to Widen Help for Syrian Rebels Gains Speed in West
By MICHAEL R. GORDON and MARK LANDLER
The United States is poised to increase its nonlethal aid to

N.H.L. Announces Effort in Support of Gay Athletes
By JEFF Z. KLEIN 9 minutes ago
The hockey league has formed a partnership with an advocacy group pledged to fight homophobia in sports.

South Korea Moves to Defuse Tensions With North
By CHOE SANG-HUN

The Opinion Pages

'Accidental Racist' and Lyrical Provocation
A song has raised hackles. Should popular culture have such power?
ROOM for DEBATE

THOMAS B. EDSALL
In Campaigns, Do You Get What You Pay For?
Strategists don't like what they see in the campaign money landscape.

• Editorial: The President's Budget
• Collins: Yipes, It's Congress on the Move
• Blow: Rand Paul Goes to Howard
• Yu: In China, Feudal Fixes for Modern Problems
• Op-Ed: Child Migrants, Alone in Court
• Fixes: The Power of Talking to Your Baby

MARKETS » At 12:35 PM ET

S.&P. 500	Dow	Nasdaq
1,594.05	14,868.09	3,298.12
+6.32	+65.85	+0.87
+0.40%	+0.44%	+0.03%

GET QUOTES My Portfolios »
Stock, ETFs, Funds Go

THURSDAY STYLES »
Picky, Picky: A Critic's Spring Fashion Guide
Cathy Horyn's practical, and ornery, guide to spring shopping (but don't hold her to it).

HOME & GARDEN »
Q&A
A Creative Mind in Motion
Ferruccio Laviani on his creative process and his latest work for Fratelli Boffi.

precise

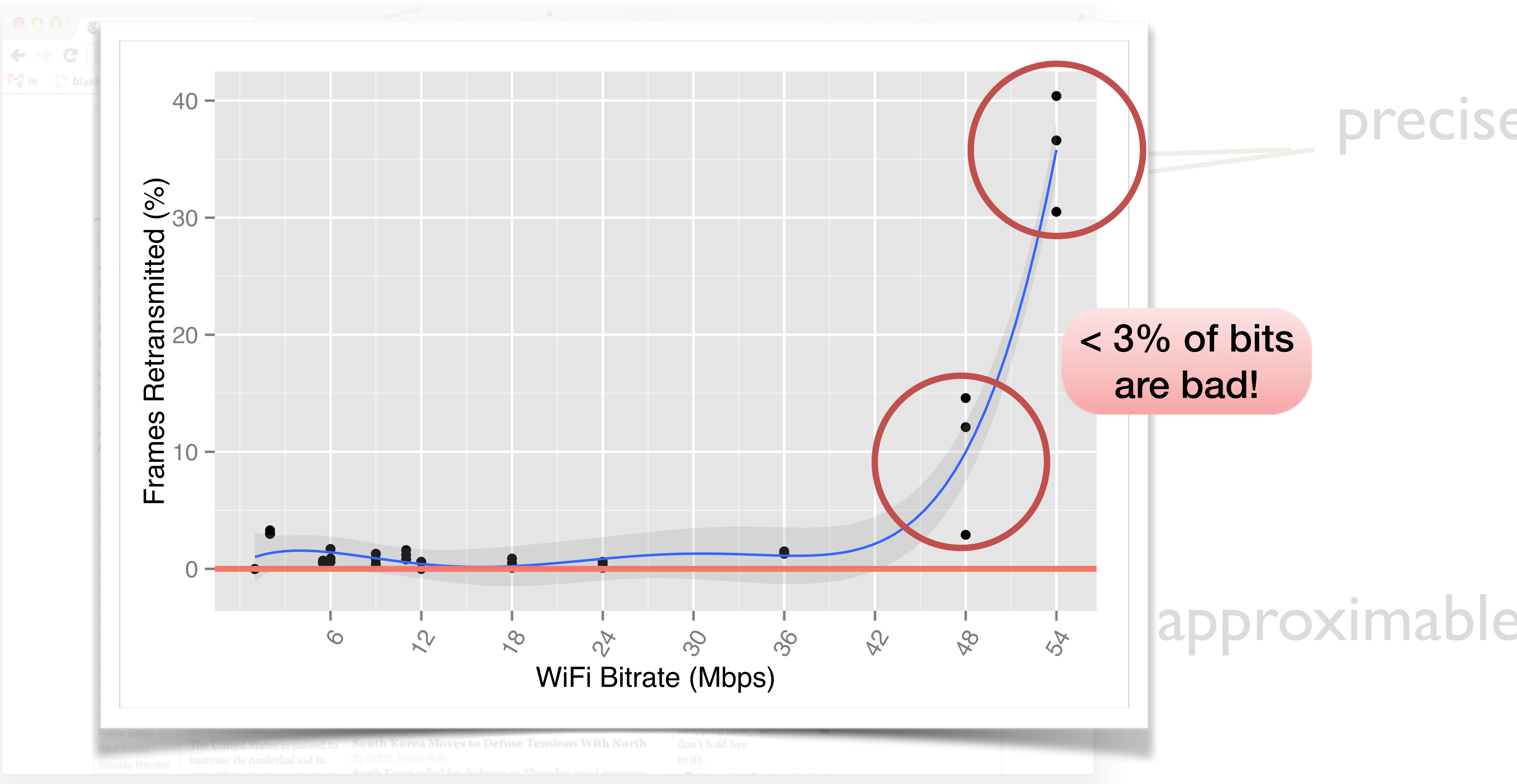
Approximate Wireless Communication

precise

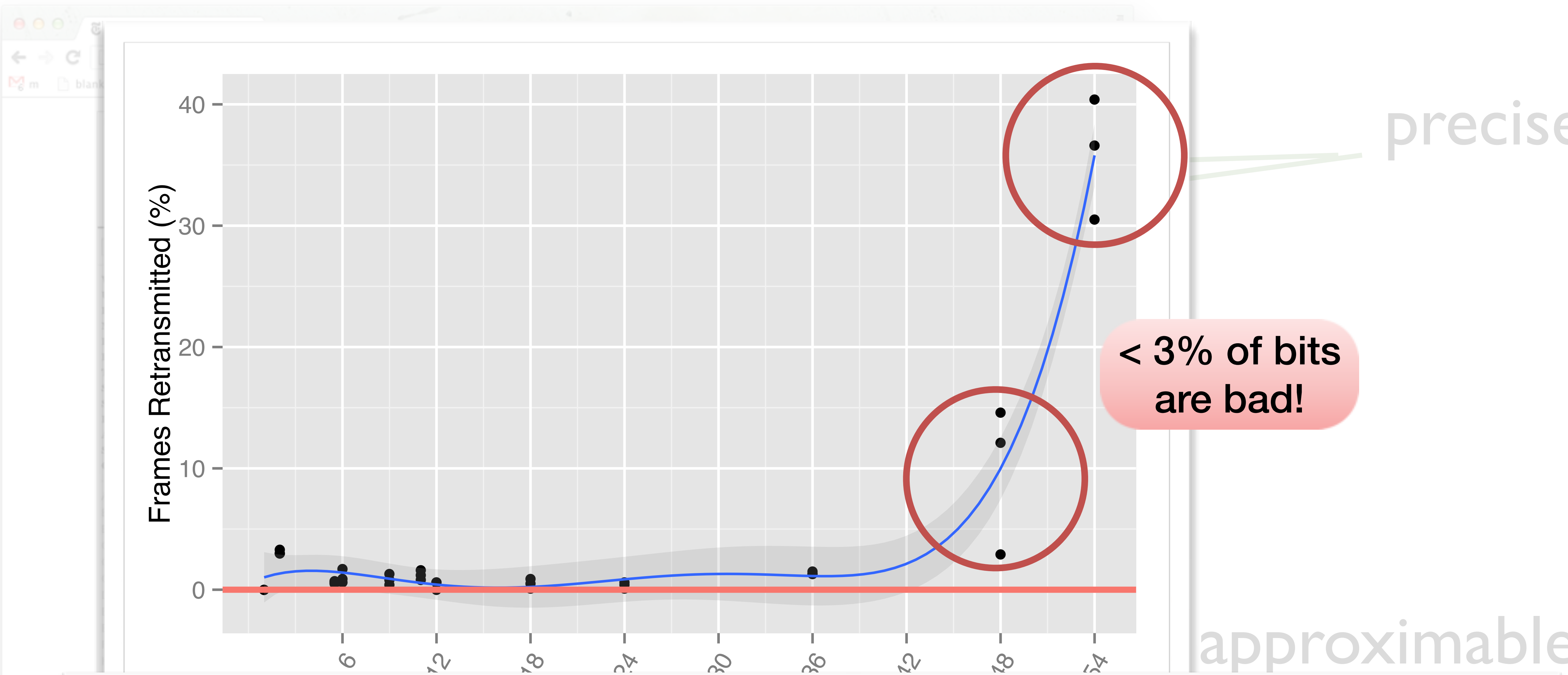
approximable

Approximate Wireless Communication

Approximate Wireless Communication

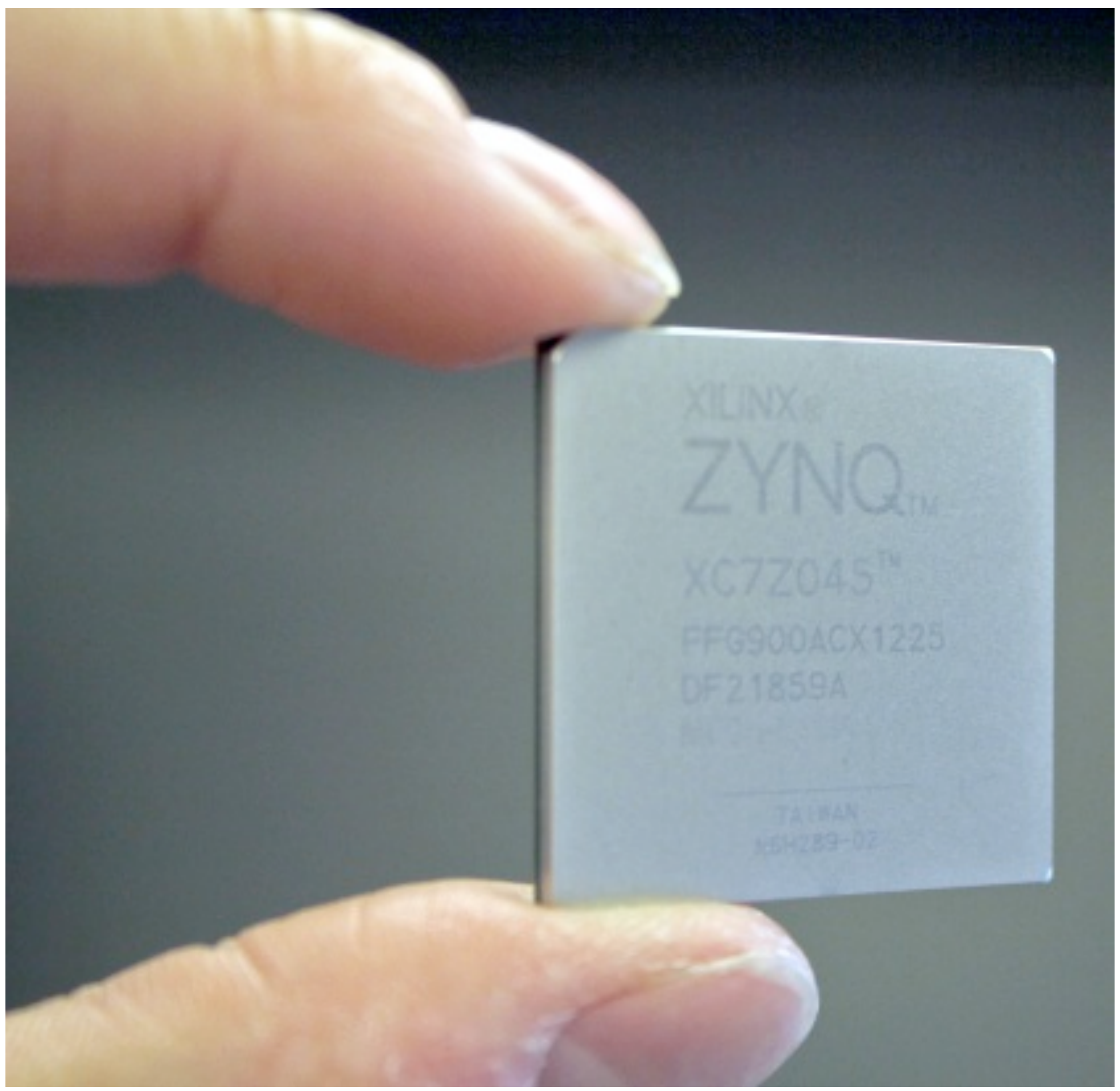


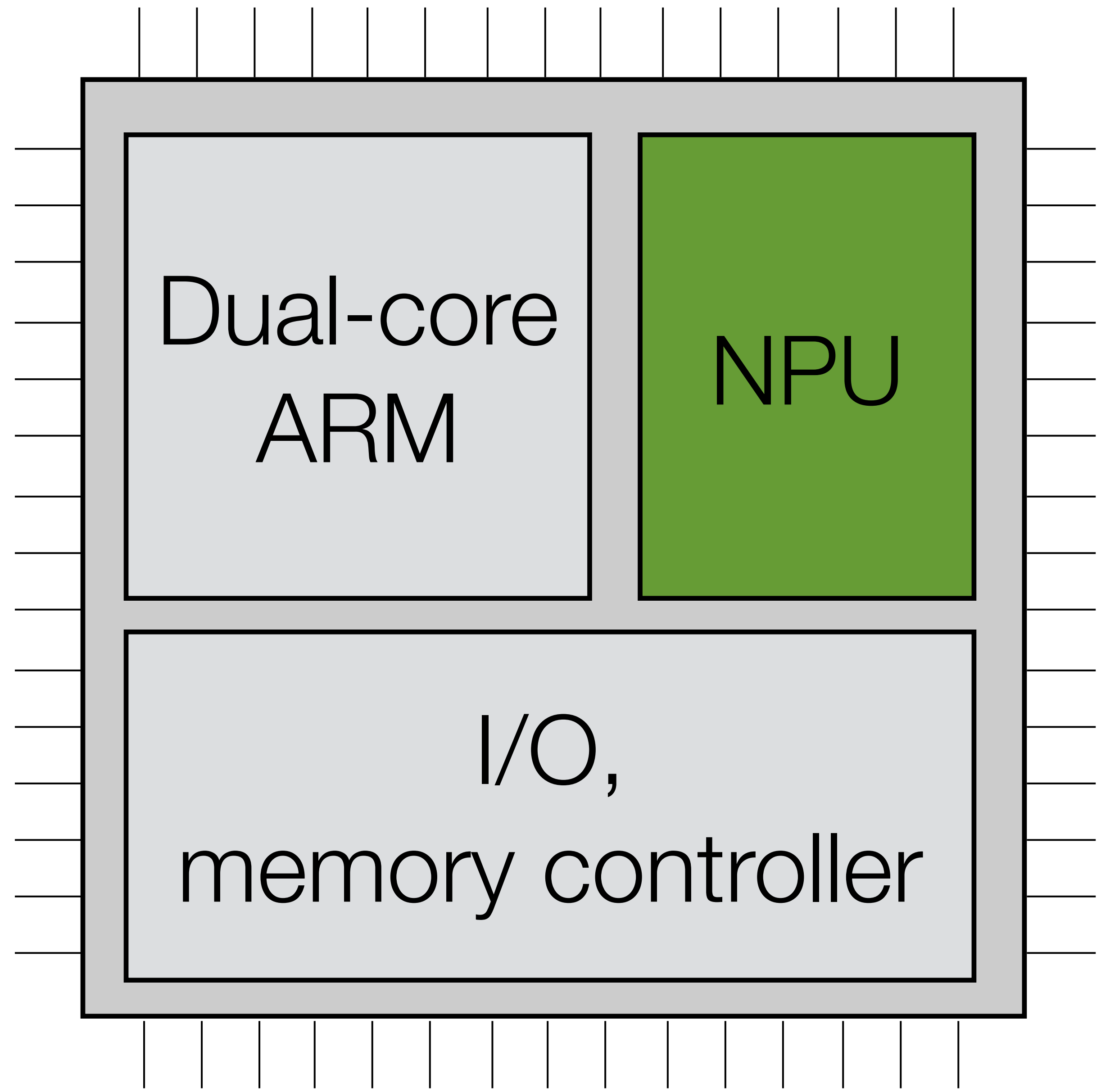
Approximate Wireless Communication



Configurable-quality wireless protocol. Quality automatically set by the data type.

Neural Acceleration on a programmable SoC





Showing End-to-End benefit

Mobile Vision/Augmented Reality
Linux on Zynq SoC (ARM CPU + FPGA)

Neural Accelerator + Compiler Support + Approx. FPGA

Measure
Energy Savings

Measure
Speedup

Evaluate
User Experience

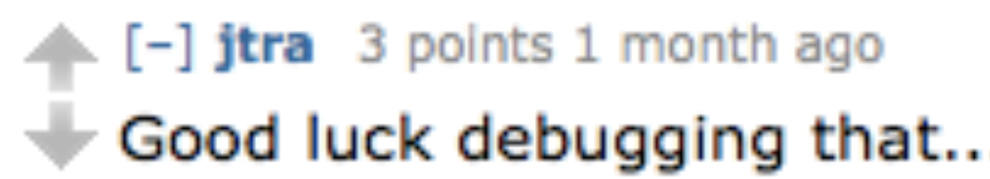
Evaluate
Programmer Effort

How will approximate computing fail?

- Applications can't take advantage of approximation opportunities
- Programmers aren't able to write/debug/test approximate code
- Quality assurance problems
- Marketing reasons: "buy my flaky system!"

How will approximate computing fail?

- Applications can't take advantage of approximation opportunities
- Programmers aren't able to write/debug/test approximate code
- Quality assurance problems
- Marketing reasons: "buy my flaky system!"



How will approximate computing fail?

- Applications can't take advantage of approximation opportunities
- Programmers aren't able to write/debug/test approximate code
- Quality assurance problems
- Marketing reasons: "buy my flaky system!"

[–] **jtra** 3 points 1 month ago

Good luck debugging that...

[–] **MorePudding** 9 points 1 month ago

Oh god, this sounds awful .. and it has enough potential to actually make things worse, if hardware vendors end up shoving it down our throats by force (i.e. people recalculating things over and over again, just to be safe; non-standard/"unofficial" hardware that tries to work around the limitations in embedded devices, and other things like that).

How will approximate computing fail?

- Applications can't take advantage of approximation opportunities
- Programmers aren't able to write/debug/test approximate code
- Quality assurance problems
- Marketing reasons: "buy my flaky system!"

[–] jtra 3 points 1 month ago

Good luck debugging that...

[–] MorePudding 9 points 1 month ago

Oh god, this sounds awful .. and it has enough potential to actually make things worse, if hardware vendors end up shoving it down our throats by force (i.e. people recalculating things over and over again, just to be safe; non-standard/"unofficial" hardware that tries to work around the limitations in embedded devices, and other things like that).

[–] MorePudding 16 points 1 month ago

There was a somewhat related post to this a few weeks ago here.

> The basic idea was simple: if hardware suffers more transient failures as it gets smaller, why not allow software to detect erroneous computations and re-execute them? This idea seemed promising until John realized THAT IT WAS THE WORST IDEA EVER. Modern software barely works when the hardware is correct, so relying on software to correct hardware errors is like asking Godzilla to prevent Mega-Godzilla from terrorizing Japan.

Other ongoing effort

- **Understanding** specialization vs. approximation benefits
- **Compiler-only** approximation w/ unsound transformations
- **HCI aspects**: how do measure user satisfaction? do incentives matter in choosing quality?
- **Language support** for QoR (quality of results, probabilistic assertions)
- **Tools** to help programmers w/ porting, testing and debugging
- Exploring uses in **energy-harvesting**-based devices
- **approxbench.org**

Conclusion

We need to exploit application properties and co-design hardware-software for better efficiency.

Getting closer to physics might lead to very big efficiency gains.

Our goal is to exploit ***approximate computing across the system***. (compute, storage, communication)

Key aspect is co-designing programming model with approximation techniques: ***disciplined*** approximate programming.

Early results encouraging. Approximate computing can potentially save our bacon in a post-Dennard era and be in the survival kit for dark silicon.

Thanks!

Luis Ceze
University of Washington
luisceze@cs.washington.edu