

# tinyML® Talks

*Enabling Ultra-low Power Machine Learning at the Edge*

“CFU Playground: Customize Your ML Processor for Your Specific TinyML Model”

Tim Callahan - Google

January 11, 2022



[www.tinyML.org](http://www.tinyML.org)

# tinyML Talks Strategic Partners

AONdevIces

arm

Deeplite

EDGE IMPULSE

emza  
visual sense

GREENWAVES  
TECHNOLOGIES

Grovety Inc.

HOTC

imagimob

KLIKA·TECH  
GLOBAL IOT SOLUTIONS

LatentAI  
Adaptive AI for a Smarter Edge

maxim  
integrated. | NOW PART OF  
ANALOG  
DEVICES

Micro.ai

NXP

PROPHESÉE  
METAVISION FOR MACHINES

Qualcomm

Qeexo

RealityAI

RENESAS

REEXEN  
technology

SAP

seeed  
The IoT Hardware Enabler

SensiML™

life.augmented

SA STREAM ANALYZE

SynSense

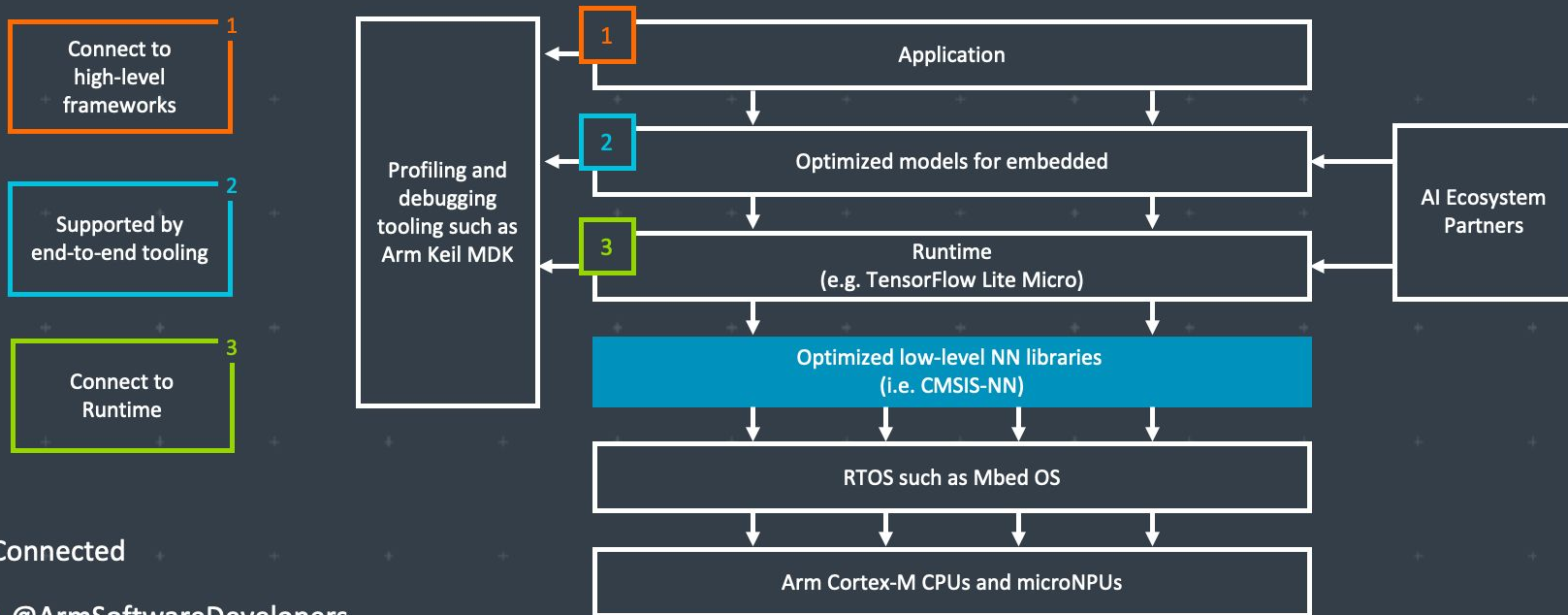
SYNTIANT

Additional Sponsorships available – contact [Olga@tinyML.org](mailto:Olga@tinyML.org) for info



# Executive Strategic Partners

# Arm: The Software and Hardware Foundation for tinyML



Stay Connected



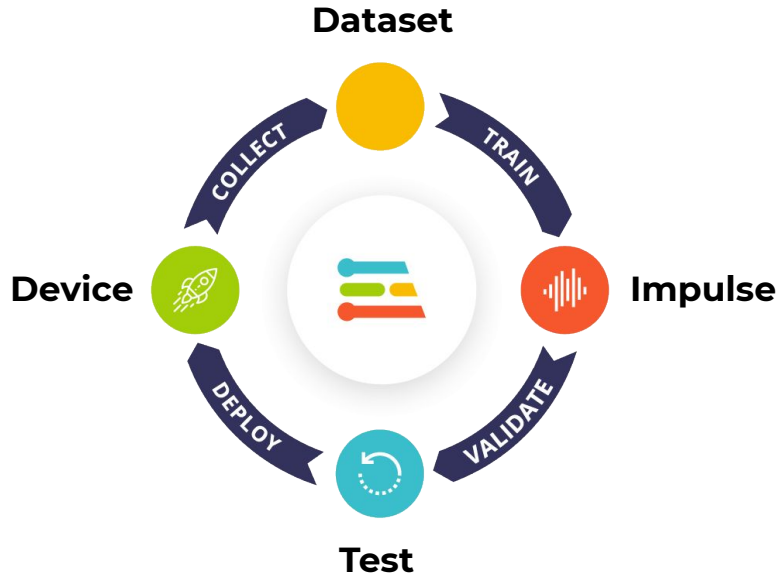
@ArmSoftwareDevelopers



@ArmSoftwareDev

Resources: [developer.arm.com/solutions/machine-learning-on-arm](https://developer.arm.com/solutions/machine-learning-on-arm)

# EDGE IMPULSE The leading edge ML platform



[www.edgeimpulse.com](http://www.edgeimpulse.com)

The screenshot displays the Edge Impulse web interface for configuring a model. The page title is "SPECTRAL FEATURES (CONTINUOUS GESTURES DEMO)". The interface includes a sidebar with navigation options: Dashboard, Devices, Data acquisition, Impulse design (with sub-options: Create impulse, Spectral features, Anomaly detection, NN Classifier), EON Tuner, Retrain model, Live classification, Model testing, Versioning, and Deployment. Below the sidebar is a "GETTING STARTED" section with links to Documentation and Forums.

The main content area shows the "Training set" configuration for a model. The parameters are:

Parameter	Value
Data in training set	18m 29s
Classes	6 (drink, fistbump, idle, snake, updown, wave)
Window length	2000 ms.
Window increase	120 ms.
Training windows	6,873

A "Generate features" button is visible below the training set parameters.

The "Feature explorer (6,819 samples)" section shows a 3D scatter plot with axes labeled "accX RMS", "accY RMS", and "accZ RMS". The plot displays data points for six classes: drink (blue), fistbump (red), idle (green), snake (orange), updown (purple), and wave (brown).

The "On-device performance" section shows:

Metric	Value
PROCESSING TIME	11 ms.
PEAK RAM USAGE	5 KB

**Qualcomm**  
AI research

# Advancing AI research to make efficient AI ubiquitous

## Power efficiency

Model design,  
compression, quantization,  
algorithms, efficient  
hardware, software tool

## Personalization

Continuous learning,  
contextual, always-on,  
privacy-preserved,  
distributed learning

## Efficient learning

Robust learning  
through minimal data,  
unsupervised learning,  
on-device learning

## A platform to scale AI across the industry



### Perception

Object detection, speech  
recognition, contextual fusion



### Reasoning

Scene understanding, language  
understanding, behavior prediction



### Action

Reinforcement learning  
for decision making



Edge cloud



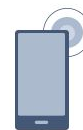
Cloud



IoT/IIoT



Automotive



Mobile

# SYNTIANT



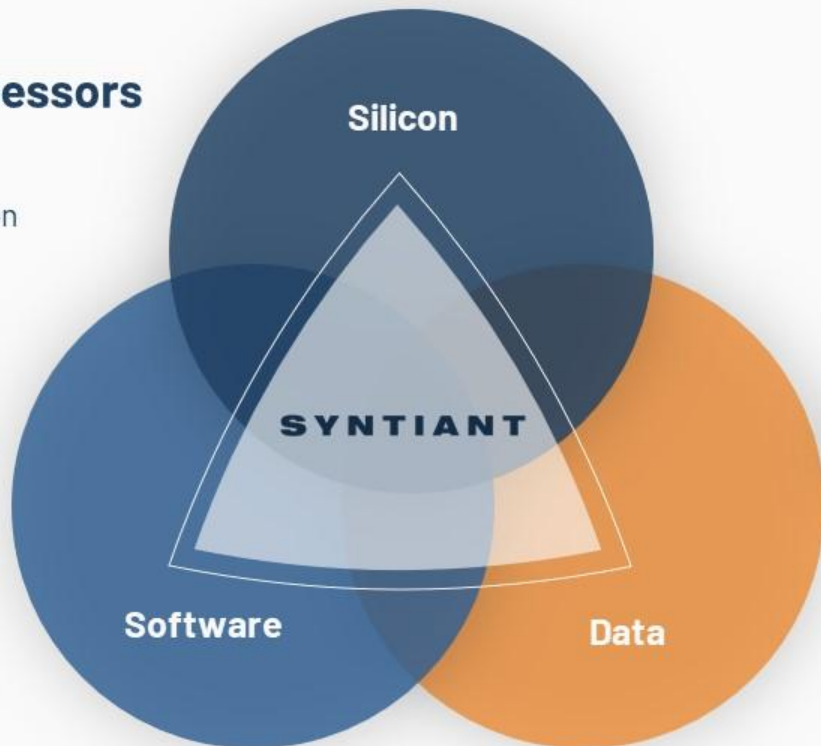
## Neural Decision Processors

- At-Memory Compute
- Sustained High MAC Utilization
- Native Neural Network Processing



## ML Training Pipeline

- Enables Production Quality Deep Learning Deployments



End-to-End  
Deep Learning  
Solutions

for

**TinyML & Edge AI**



## Data Platform

- Reduces Data Collection Time and Cost
- Increases Model Performance

**SYNTIANT**



[partners@syntiant.com](mailto:partners@syntiant.com)



[www.syntiant.com](http://www.syntiant.com)

# Platinum Strategic Partners





# WE USE AI TO MAKE OTHER AI FASTER, SMALLER AND MORE POWER EFFICIENT



**Automatically compress** SOTA models like MobileNet to <200KB with **little to no drop in accuracy** for inference on resource-limited MCUs



**Reduce** model optimization trial & error from weeks to days using Deeplite's **design space exploration**



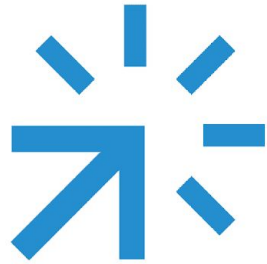
**Deploy more** models to your device without sacrificing performance or battery life with our **easy-to-use software**

BECOME BETA USER [bit.ly/testdeeplite](https://bit.ly/testdeeplite)

mobilityXlab

arm





**KLIKA · TECH**

GLOBAL IOT SOLUTIONS



# Add Advanced Sensing to your Product with Edge AI / TinyML

<https://reality.ai>

 [info@reality.ai](mailto:info@reality.ai)

 [@SensorAI](https://twitter.com/SensorAI)

 [Reality AI](https://www.linkedin.com/company/reality-ai)

## Pre-built Edge AI sensing modules, plus tools to build your own

### Reality AI solutions

Prebuilt sound recognition models for  
indoor and outdoor use cases

Solution for industrial anomaly detection

Pre-built automotive solution that lets cars  
"see with sound"

### Reality AI Tools<sup>®</sup> software

Build prototypes, then turn them into  
real products

Explain ML models and relate the function  
to the physics

Optimize the hardware, including  
sensor selection and placement

# BROAD AND SCALABLE EDGE COMPUTING PORTFOLIO

## Microcontrollers & Microprocessors

### Arm® Core



Arm® Cortex®-M 32-bit MCUs  
Arm ecosystem, Advanced security, Intelligent IoT



Arm®-based High-end 32 & 64-bit MPUs  
High-resolution HMI, Industrial network & real-time control



Arm® Cortex®-M0+ Ultra-low Power 32-bit MCUs  
Innovative process tech (SOTB), Energy harvesting

**Renesas Synergy™** Arm®-based 32-bit MCUs for Qualified Platform  
Qualified software and tools

### Renesas Core



Ultra-low Energy 8 & 16-bit MCUs  
Bluetooth® Low Energy, SubGHz, LoRa®-based Solutions



High Power Efficiently 32-bit MCUs  
Motor control, Capacitive touch, Functional safety, GUI



40nm/28nm process Automotive 32-bit MCUs  
Rich functional safety and embedded security features

## Core technologies

### AI

A broad set of high-power and energy-efficient embedded processors

### Security & Safety

Comprehensive technology and support that meet the industry's stringent standards



### Digital & Analog & Power Solution

Winning Combinations that combine our complementary product portfolios

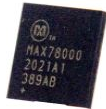
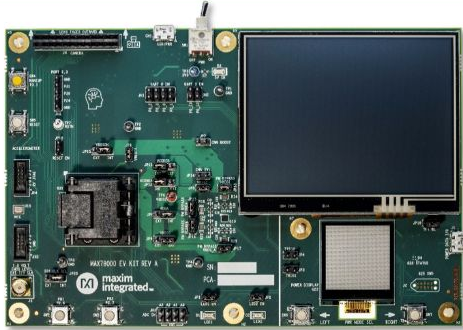
### Cloud Native

Cross-platforms working with partners in different verticals and organizations

# Gold Strategic Partners

## Maxim Integrated: Enabling Edge Intelligence

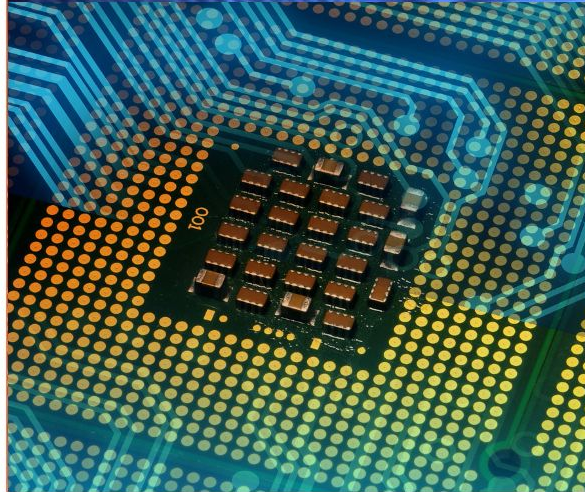
### Advanced AI Acceleration IC



The new MAX78000 implements AI inferences at low energy levels, enabling complex audio and video inferencing to run on small batteries. Now the edge can see and hear like never before.

[www.maximintegrated.com/MAX78000](http://www.maximintegrated.com/MAX78000)

### Low Power Cortex M4 Micros



Large (3MB flash + 1MB SRAM) and small (256KB flash + 96KB SRAM, 1.6mm x 1.6mm) Cortex M4 microcontrollers enable algorithms and neural networks to run at wearable power levels.

[www.maximintegrated.com/microcontrollers](http://www.maximintegrated.com/microcontrollers)

### Sensors and Signal Conditioning



Health sensors measure PPG and ECG signals critical to understanding vital signs. Signal chain products enable measuring even the most sensitive signals.

[www.maximintegrated.com/sensors](http://www.maximintegrated.com/sensors)



# Latent AI

Adaptive AI for the Intelligent Edge

[Latentai.com](https://latentai.com)



# Micr .ai





**NXP**



**seeed** studio  
**The IoT Hardware Enabler**

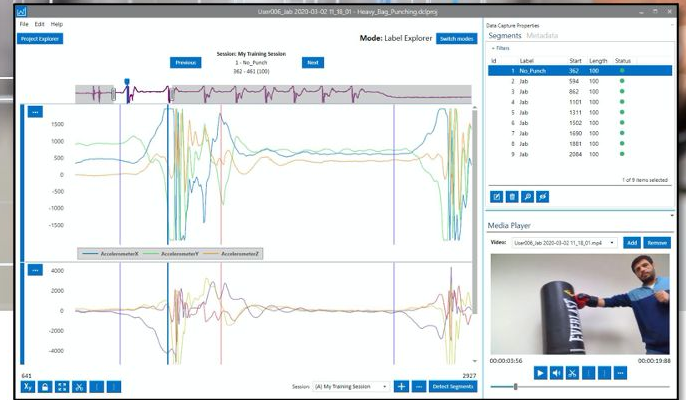


# Build Smart IoT Sensor Devices From Data

SensiML pioneered TinyML software tools that auto generate AI code for the intelligent edge.

- End-to-end AI workflow
- Multi-user auto-labeling of time-series data
- Code transparency and customization at each step in the pipeline

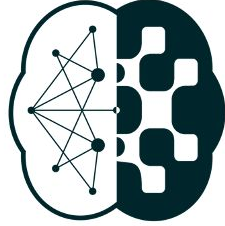
We enable the creation of production-grade smart sensor devices.



[sensiml.com](https://sensiml.com)



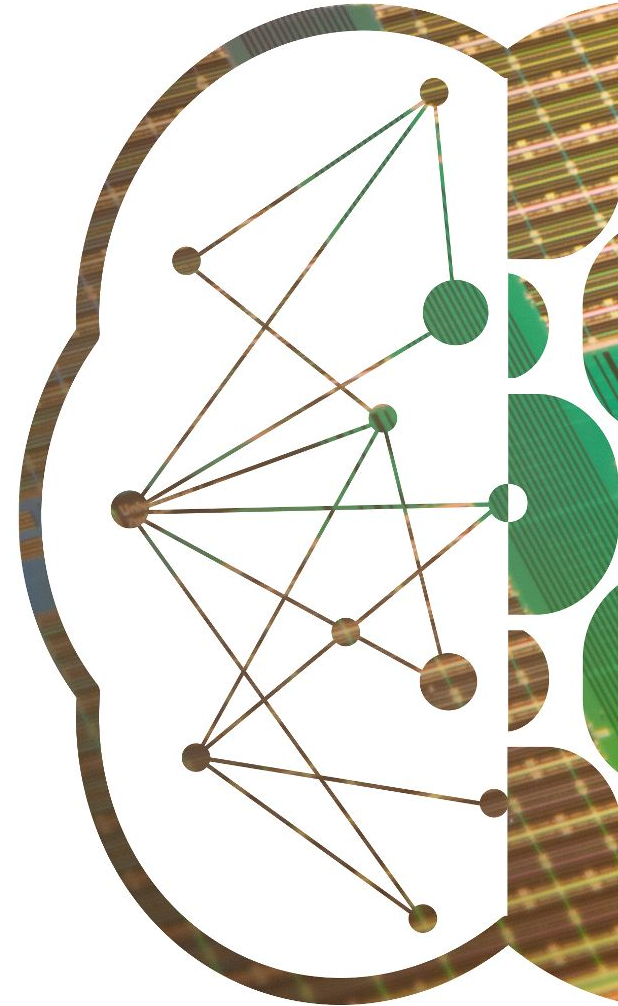
life.augmented



# SynSense

**SynSense** builds **sensing and inference** hardware for **ultra-low-power** (sub-mW) **embedded, mobile and edge** devices. We design systems for **real-time always-on smart sensing**, for audio, vision, IMUs, bio-signals and more.

<https://SynSense.ai>



# Silver Strategic Partners

AONdevices



The logo for Grovety Inc. features a green lightning bolt icon followed by the text "Grovety Inc." in a green, sans-serif font.





# tinyML Summit 2022

Miniature dreams can come true...

March 28-30, 2022

Hyatt Regency San Francisco Airport

<https://www.tinyml.org/event/summit-2022/>

*The Best Product of the Year and the Best Innovation of the Year awards are open for nominations between **November 15 and February 28.***

# tinyML Research Symposium 2022

March 28, 2022

<https://www.tinyml.org/event/research-symposium-2022>

More sponsorships are available: [sponsorships@tinyML.org](mailto:sponsorships@tinyML.org)



## Join Growing tinyML Communities:



7.7k members in  
40 Groups in 33 Countries

**tinyML - Enabling ultra-low Power ML at the Edge**

<https://www.meetup.com/tinyML-Enabling-ultra-low-Power-ML-at-the-Edge/>



2.5k members  
&  
4.3k followers

**The tinyML Community**

<https://www.linkedin.com/groups/13694488/>







Subscribe to  
**tinyML YouTube Channel**  
for updates and notifications  
*(including this video)*  
[www.youtube.com/tinyML](http://www.youtube.com/tinyML)



**tinyML**  
4.3K subscribers

**5.8k subscribers, 320+ videos with 166k views**

HOME VIDEOS PLAYLISTS COMMUNITY CHANNELS ABOUT

tinyML Summit 2021 Partner Session: Production...  
146 views • 1 month ago

tinyML Summit 2021 Partner Session: System...  
131 views • 1 month ago

tinyML Talks Massimo Banzi: tiny machine learnin...  
1.4K views • 1 month ago

tinyML Summit 2021 Partner Session: It's an SN...  
420 views • 1 month ago

tinyML Summit 2021 Partner Session: The...  
104 views • 1 month ago

tinyML Summit 2021 Partner Session: Innovativ...  
110 views • 1 month ago

Deploying AI in Embedded Systems  
430 views • 1 month ago

tinyML Summit 2021 Breaking News on...  
472 views • 1 month ago

tinyML Summit 2021 Partner Session: TinyML...  
160 views • 1 month ago

tinyML Summit 2021 Breaking News on...  
257 views • 1 month ago

tinyML Summit 2021 Breaking News on...  
299 views • 1 month ago

tinyML Summit 2021 Breaking News on...  
173 views • 1 month ago

tinyML Summit 2021 Partner Session: Machine...  
212 views • 1 month ago

tinyML Summit 2021 Partner Session: Tiny and...  
109 views • 1 month ago

tinyML Summit 2021 tiny Talks: Real-World...  
55 views • 1 month ago

tinyML Summit 2021 tiny Talks: Environmental Nois...  
173 views • 1 month ago

tinyML Summit 2021 Breaking News on...  
151 views • 1 month ago

tinyML Summit 2021 tiny Talks: Insights from a Mult...  
222 views • 1 month ago

tinyML Summit 2021 tiny Talks: TinyML Software...  
121 views • 1 month ago

tinyML Summit 2021 Keynote: Data-Free Mode...  
240 views • 1 month ago

tinyML Summit Partner Session: Pushing the AI...  
82 views • 1 month ago

tinyML Summit 2021 Partner Session: Low pow...  
129 views • 1 month ago

tinyML Summit 2021 Partner Session: How...  
95 views • 1 month ago

tinyML Summit 2021 Partner Session: Low-pow...  
168 views • 1 month ago

tinyML Summit 2021 Partner Session: TinyML is...  
115 views • 1 month ago

tinyML Summit 2021 Keynote: Efficient Audio...  
1.4K views • 1 month ago

tinyML Summit 2021 tiny Talks: An Introduction to a...  
180 views • 1 month ago

tinyML Summit 2021 tiny Talks: Low-precision...  
215 views • 1 month ago

tinyML Summit 2021 tiny Talks: Hardware Aware...  
75 views • 1 month ago

tinyML Summit 2021 tiny Talks: Neutrin0: A BlackBo...  
168 views • 1 month ago

tinyML Summit 2021 Panel Discussion: tinyML...  
1:01:15

tinyML Summit 2021 tiny Talks: Person Detection...  
18:26

tinyML Summit 2021 tiny Talks: Using Neural...  
19:00

tinyML Summit 2021 Keynote: Adaptive Neural...  
55:15

tinyML Summit 2021 Keynote: mililoules for...  
09:40

tinyML Summit 2021 Market Opportunities for Edge AI  
51:28



Next tinyML Talks

Date	Presenter	Topic / Title
Tuesday, January 18	Ashutosh Pandey, Infineon Technologies	Exploring techniques to build efficient and robust TinyML deployments

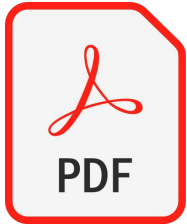
Webcast start time is 8:00 am Pacific time

Please contact [talks@tinymml.org](mailto:talks@tinymml.org) if you are interested in presenting

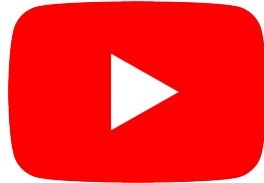


Reminders

Slides & Videos will be posted  
tomorrow



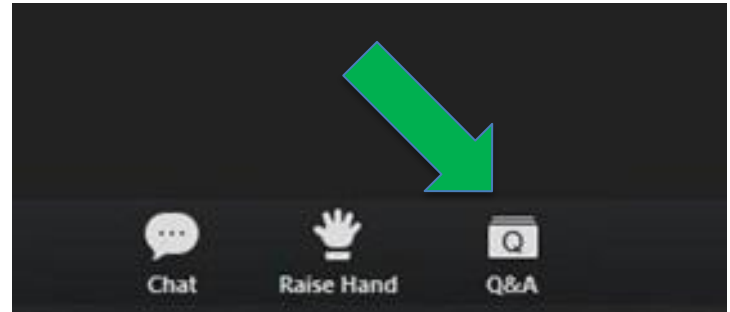
[tinyml.org/forums](https://tinyml.org/forums)



[youtube.com/tinyml](https://youtube.com/tinyml)



Please use the Q&A window for your  
questions





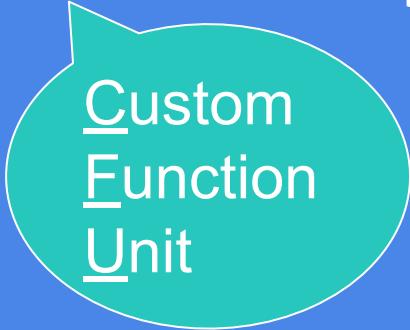
# Tim Callahan



Tim Callahan works at Google with the open source FPGA toolchain (SymbiFlow) team. His work is to help make FPGA development more accessible, fun, and rewarding. His research interests include anything that involves optimizing the hardware/software boundary. He has degrees from UC Berkeley, Cambridge University, and the University of Minnesota.

---

# CFU Playground



Custom  
Function  
Unit

Customize Your ML Processor for  
*Your* Specific TinyML Model

TinyML.org Talk  
January 11, 2022  
Tim Callahan, [tcal@google.com](mailto:tcal@google.com) presenting the work of many

Disclaimer: NOT an official Google project

*It's on GitHub: Fork it, fix it, send a PR!*

---

---

# Acknowledgements

**Googlers:** Alan Green, David Lattimore, Dan Callaghan,  
Tim Ansell,  
Interns Rachel Sugrono & Joey Bushagour  
TFLM (Pete Warden and team)

## **Antmicro**

**Harvard:** Prof Vijay Reddi, Shvetank Prakash, Colby Banbury

**Open source:** VexRiscv , LiteX, SymbiFlow, Yosys, Nextpnr,  
VTR, Migen, nMigen, Renode, Verilator, OpenOCD, ..

---

---

# Open Source Showcase!

- ML library                      TensorFlow Lite    -- open source
  - CPU ISA                         RISC-V                -- open
  - CPU design                     VexRiscv             -- open source
  - FPGA SoC/IP                  LiteX                  -- open source
  - FPGA synth/PnR               SymbiFlow, Yosys, -- open source  
   Nextpnr, VPR
    - FPGA vendor tools can be used if you wish
  - Python HW gen                Migen, nMigen      -- open source
  - Simulation                     Renode, Verilator -- open source
  
  - The only proprietary component is the **FPGA** itself
-

---

# Benefits of Open Source

- No licensing fees
- No license headaches in your build system
- You can fix bugs and address shortcomings
- No vendor lock-in
- Transparency -- open for inspection, both sw & hw
  - Do you trust a third-party blob of sw or hw integrated with your product?

→ see [betrusted.io](https://betrusted.io)

*“At its core lies not a CPU, but an FPGA, so that users are empowered to build their processors from scratch and know there are no flaws or backdoors in the architecture that could lead to security compromises.”*

---



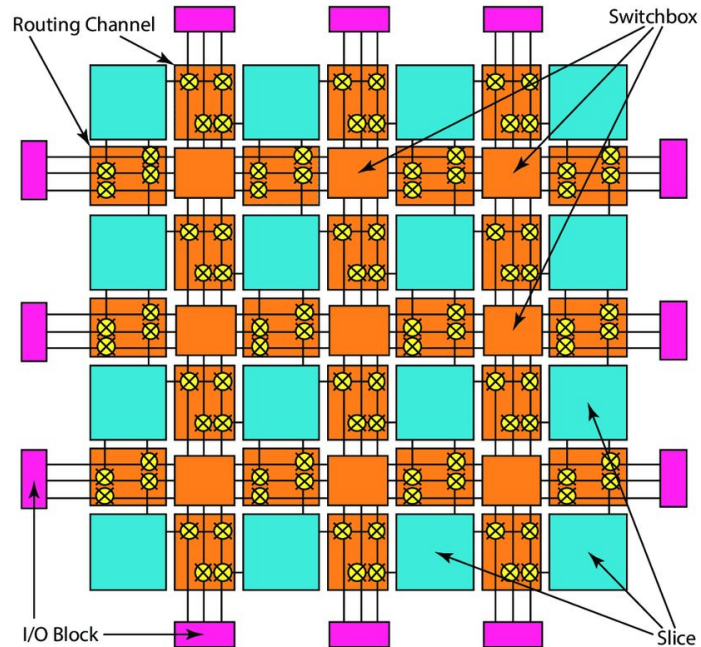
---

# CFU Playground Key Ideas

- **Specialize** the entire stack for your particular model – both the ML kernels AND the processor
  - **Make it easy to get started; allow quick iterations**  
→ rapid iterative design space exploration
-

---

# FPGA – programmable hardware



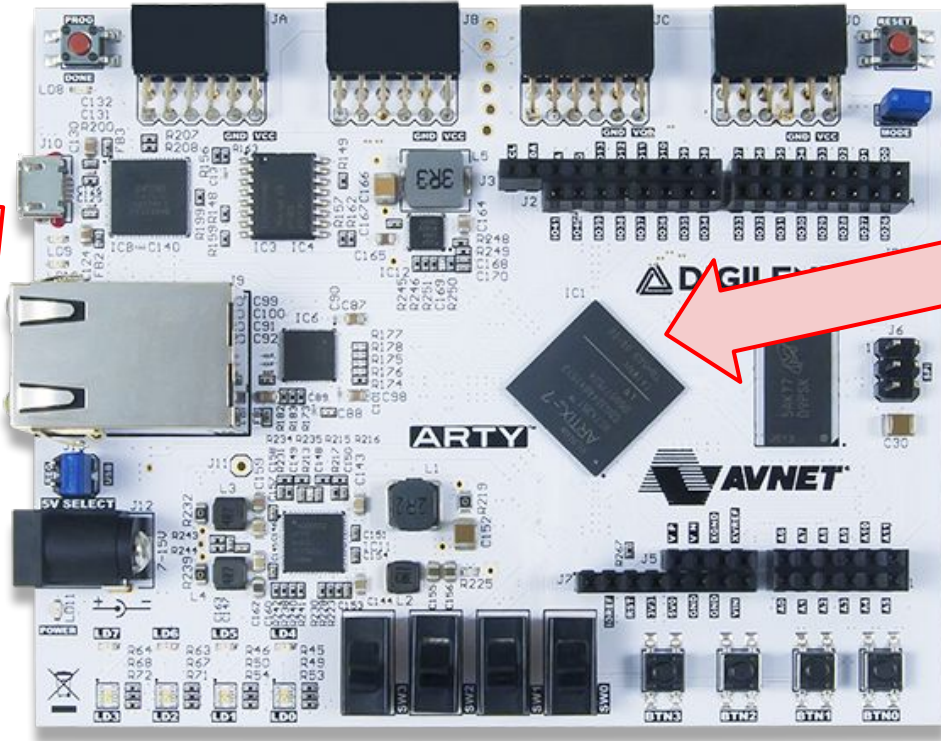
A “bitstream” or “configuration”

- determines the function of each configurable logic block
- and sets switches in the interconnect to connect them to each other and to I/O

Image source: “Parallel Programming for FPGAs”, Ryan Kastner, Janarбек Matai, Stephen Neuendorffer (CC 4.0)

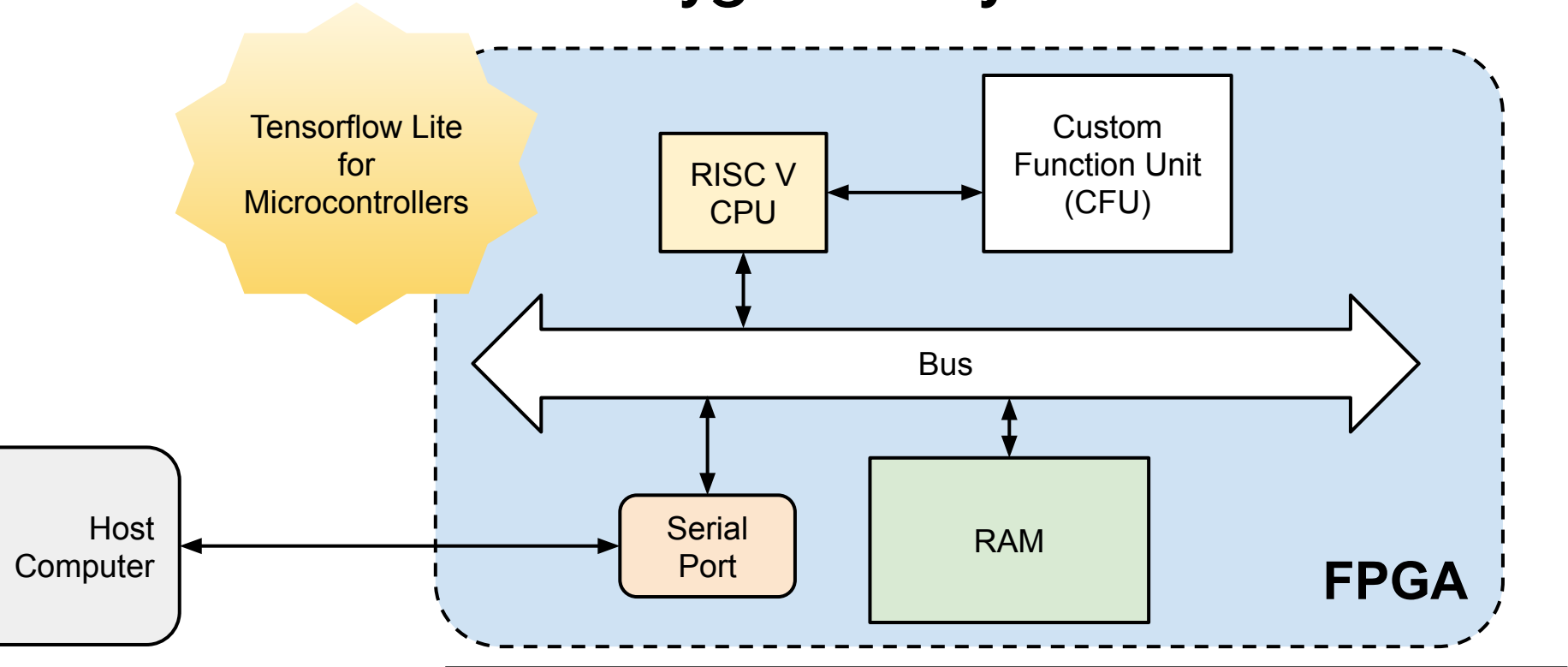
---

**USB  
Port**

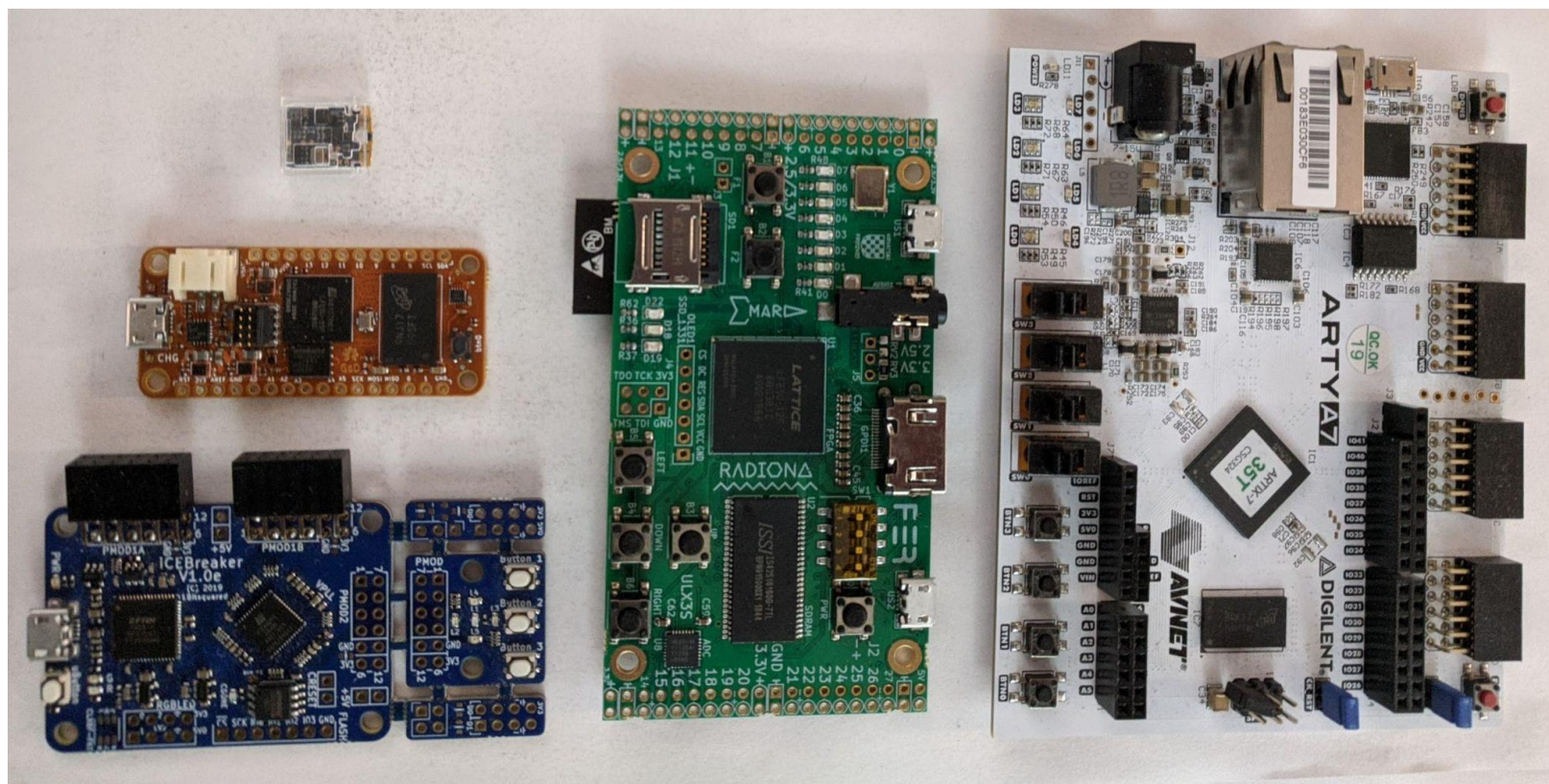


**FPGA**

# CFU Playground System Architecture





















# CFU Playground -- some of the tested boards



## Summary

### Jobs

- ✓ setup-matrix
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)
- ✓ test-projects (proj\_template, ...)

Name	Size	
 test-results-example_cfu-common_soc-1bitsquared_icebreaker	511 KB	
 test-results-example_cfu-common_soc-antmicro_lpddr4_test_board	511 KB	
 test-results-example_cfu-common_soc-camlink_4k	512 KB	
 test-results-example_cfu-common_soc-colorlight_i5	511 KB	
 test-results-example_cfu-common_soc-decklink_mini_4k	511 KB	
 test-results-example_cfu-common_soc-decklink_quad_hdmi_recorder	511 KB	
 test-results-example_cfu-common_soc-diligent_arty	511 KB	
 test-results-example_cfu-common_soc-diligent_arty_s7	511 KB	
 test-results-example_cfu-common_soc-diligent_genesys2	511 KB	

---

# CFU Playground Prerequisites:

- Linux host
- You will use:
  - git
  - C++ (mostly C)
  - Hardware design – Verilog or ???
    - unless you just want to experiment with CPU configuration
- A supported FPGA board
  - or you could just simulate with Renode or Verilator

Chisel

nMigen (Python)

XLS

SystemVerilog

---

---

# CFU Playground Uses

- **Deploy** a soft CPU+CFU on FPGA for tinyML
    - Firmware updates include a new ML model, new software, *and* a new CPU+CFU
  - **Prototype** a custom RISC-V-based ASIC
    - Yes you! [efabless.com](https://efabless.com) MPW will fab your open-source chip
  - **Learn** about ML software, hardware, and performance
  - **Research** new ML approaches
    - while co-designing the hardware to support it
-



---

# The Hardware Lottery



- Sara Hooker’s observation that the success of new ML approaches depends on their compatibility with downstream software and hardware
- Here you can “make your own luck”!

---

---

# RISC-V and CFU Basics

---

---

# The RISC-V Add Instruction

00720C33

add x24,x4,x7

# x24 = x4 + x7

---

---

# The RISC-V Add Instruction

00720C33

add x24,x4,x7

# x24 = x4 + x7

0000000

00111

00100

000

11000

0110011

---

---

# The RISC-V Add Instruction

00720C33

add x24,x4,x7

# x24 = x4 + x7

0000000	00111	00100	000	11000	0110011
					<b>opcode</b>
					<b>ALU</b>

---

---

0000000	00111	00100	000	11000	0110011
					<b>opcode</b>
					<b>ALU</b>



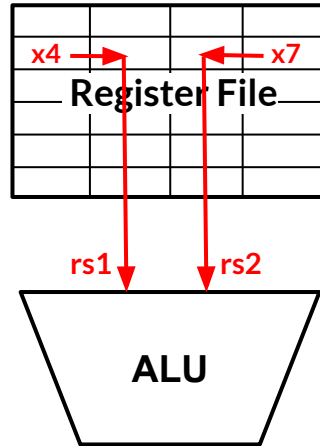
---

0000000	00111	00100	000	11000	0110011
funct7	rs2	rs1	funct3	rd	opcode
ADD	x7	x4	ADD	x24	ALU



---

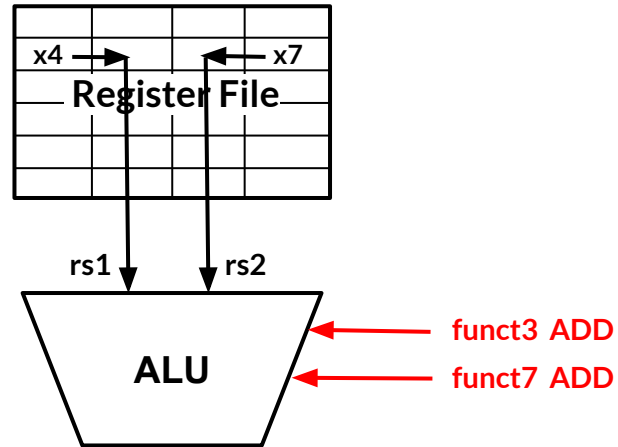
0000000	00111	00100	000	11000	0110011
funct7	rs2	rs1	funct3	rd	opcode
ADD	x7	x4	ADD	x24	ALU





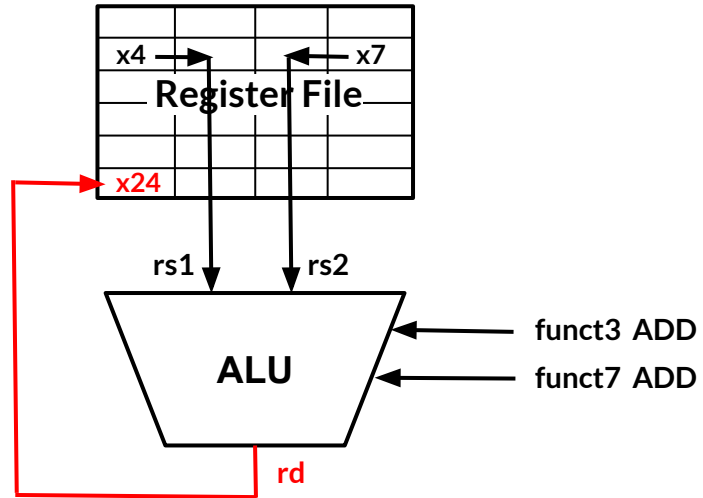
---

0000000	00111	00100	000	11000	0110011
funct7	rs2	rs1	funct3	rd	opcode
ADD	x7	x4	ADD	x24	ALU

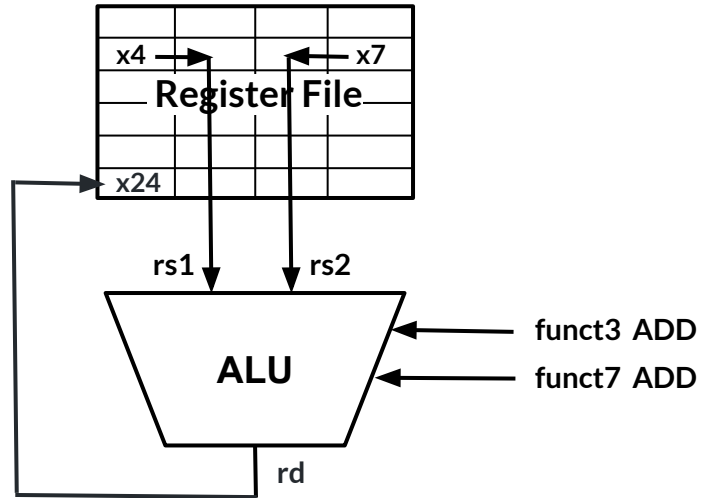


---

0000000	00111	00100	000	11000	0110011
funct7	rs2	rs1	funct3	rd	opcode
ADD	x7	x4	ADD	x24	ALU

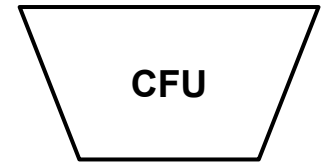


0000000	00111	00100	000	11000	0110011
funct7	rs2	rs1	funct3	rd	opcode
ADD	x7	x4	ADD	x24	ALU



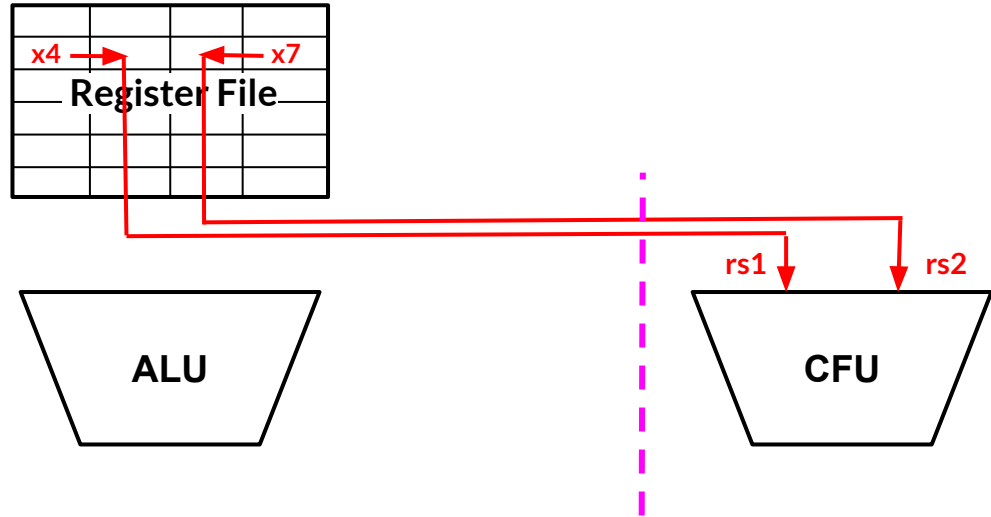
---

0000000	00111	00100	000	11000	0001011
					opcode
					CUSTOM

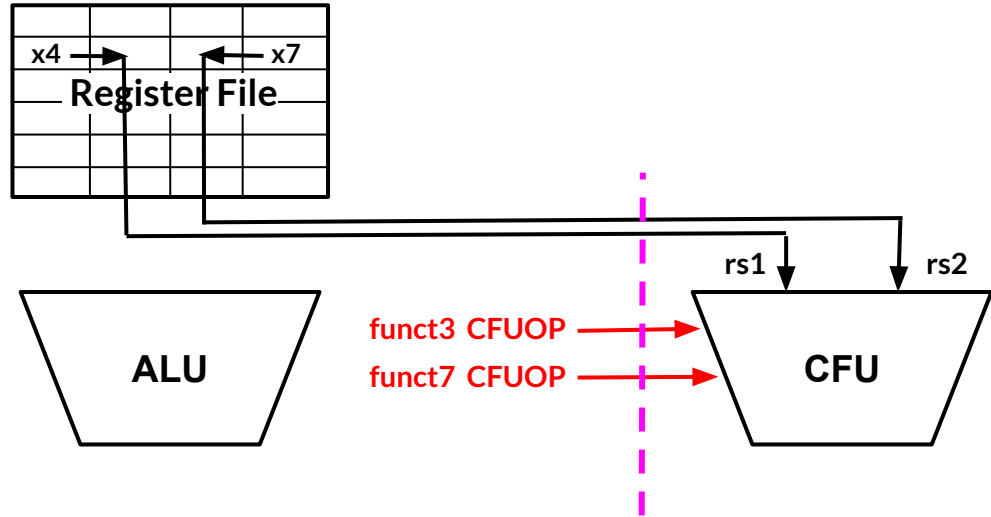


---

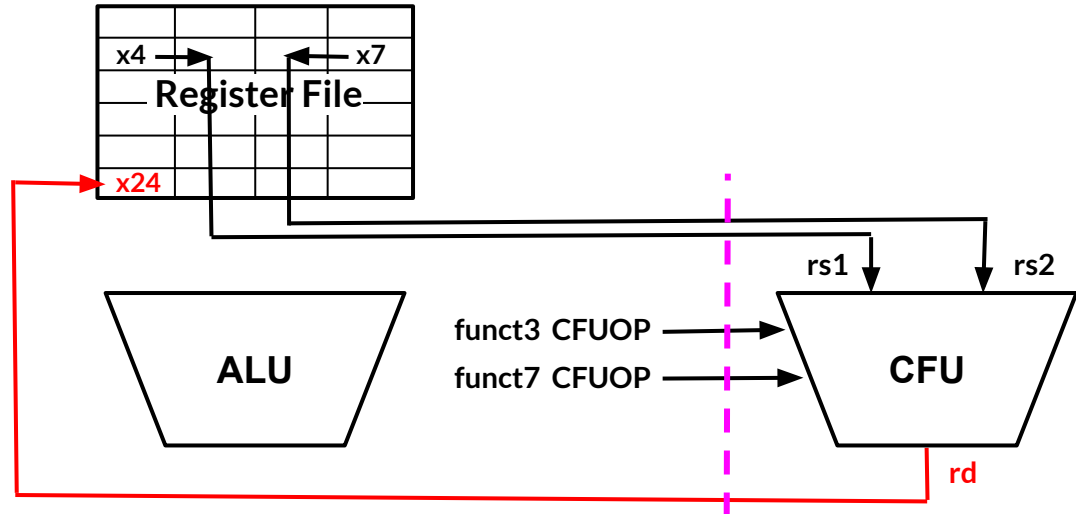
0000000	00111	00100	000	11000	0001011
funct7	rs2	rs1	funct3	rd	opcode
CFUOP	x7	x4	CFUOP	x24	CUSTOM



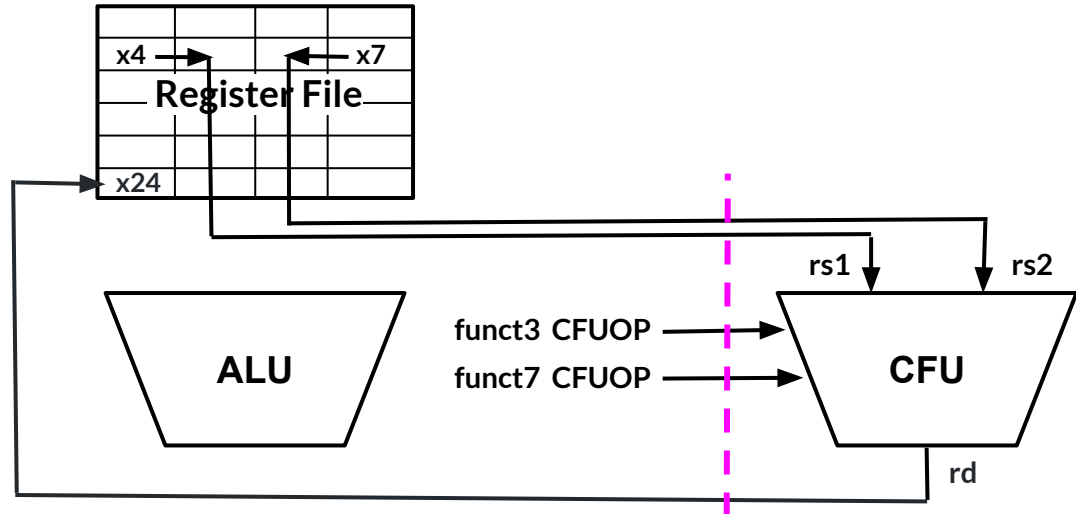
0000000	00111	00100	000	11000	0001011
funct7	rs2	rs1	funct3	rd	opcode
CFUOP	x7	x4	CFUOP	x24	CUSTOM



0000000	00111	00100	000	11000	0001011
funct7	rs2	rs1	funct3	rd	opcode
CFUOP	x7	x4	CFUOP	x24	CUSTOM



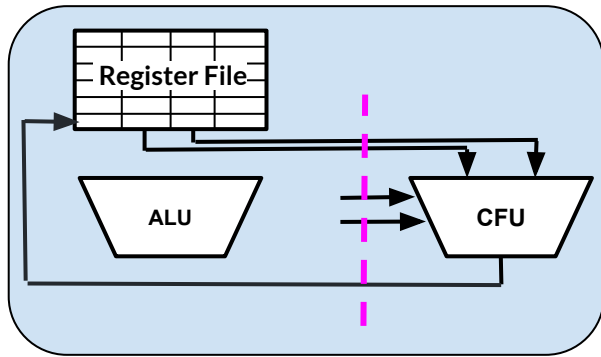
0000000	00111	00100	000	11000	0001011
funct7	rs2	rs1	funct3	rd	opcode
CFUOP	x7	x4	CFUOP	x24	CUSTOM





---

# CFU details



- Two operands from the regfile, one result written back
  - Multiple/variable cycles; can refuse new inputs while working → pipelined or iterative computation
  - NO direct connection between CFU & the memory hierarchy
  - CFU cannot otherwise affect CPU state (branch etc.)
  - CFU can contain state -- registers and memories that persist
  - CFU can contain its own sequencer
  - One CFU, multiple instructions that can access the shared state
-

---

## So how do we use the CFU?

- Do we build a nifty compiler? NO!
  - YOU are the compiler..it's up to you to insert uses of your new instructions into the code
  - After all, you're the one who just designed the new instruction to speed up your code, so you know exactly where it will be used
-

---

# How to use the CFU from software

- Access the new instruction using function call syntax:

```
rslt = cfu_op(funct3, funct7, op1, op2);
```

*Compile-time constants*

*C / C++ variables / expressions*

---

---

# How to use the CFU from software

- Access the new instruction using function call syntax:

```
rslt = cfu_op(funct3, funct7, op1, op2);
```

*Compile-time constants*

*C / C++ variables / expressions*

```
rslt = cfu_op(0, 1, op1, op2);
```

---

---

# How to use the CFU from software

- Access the new instruction using function call syntax:

```
rslt = cfu_op(funct3, funct7, op1, op2);
```

*Compile-time constants*

*C / C++ variables / expressions*

```
#define HAMMING_DISTANCE(x,y) cfu_op(0, 1, (x), (y));
```

```
rslt = HAMMING_DISTANCE(op1, op2);
```

---

---

# How to use the CFU from software

- Custom instruction macros intermix with regular C code:

```
t1 = *x;  
t2 = cfu_op(0, 0, t1, b);  
t3 = cfu_op(1, 0, t2, b);  
*x = t3;
```

No overhead!

Compiled and disassembled:

```
400001a0:      00812783      lw      a5,8(sp)  
400001a4:      00d7878b      cfu[0,0] a5, a5, a3  
400001a8:      00d7978b      cfu[0,1] a5, a5, a3  
400001ac:      00f12423      sw      a5,8(sp)
```

Objdump can't disassemble the custom instructions,  
so we wrote a helper script..

---

---

# Why use a CFU approach for tinyML?

- With ML inference, the hot spots are small & very hot
  - A SMALL amount of custom hardware to exploit the bit-level flexibility of FPGA can accelerate a LARGE fraction of the runtime
  - Leave complexity, setup, outer loops in SW
  - Iterative approach: easy to take first step, then next step, ...
  - In our experience, the CFU will incrementally grow to become *almost* a full-blown “accelerator”
-

---

---

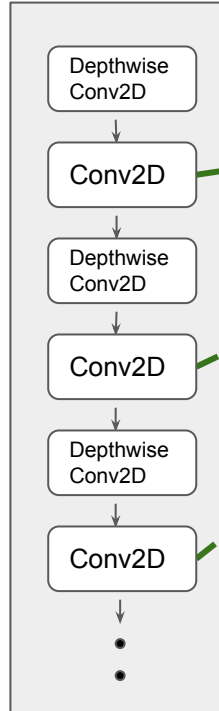
# TensorFlow Lite for Microcontrollers (TFLM)

---



# TensorFlow Lite for Microcontrollers

TFLite model



All instances use the same kernel...

```
Conv2D(.....)
{
  if (filter.shape() is 1x1xC) {
    Conv2D_1x1(.....)
    return;
  }
  for ( ... ) {
    for ( ... ) {
      for ( ... ) {
        ....
      }
    }
  }
}
```

...but a kernel can dispatch to a **specialized** version

```
Conv2D_1x1(.....)
{
  for ( ... ) {
    for ( ... ) {
      for ( ... ) {
        ....
      }
    }
  }
}
```

---

---

# CFU Playground

---

---

# CFU Playground Build

```
$ cd proj/proj_template_v/ # establish project
$ make prog 3.5min # build gateway using ./cfu.v
                    and put the bitstream on the FPGA
$ make -j8 software <1min # build software w/ local overrides
$ make load # load and execute the software
```

...then interact with the software running on the board

---

```
----- Liftoff! -----  
Hello, World!  
  
CFU Playground  
-----  
1: TfLM Models menu  
2: Functional CFU Tests  
3: Project menu  
4: Performance Counter Tests  
5: TFLite Unit Tests  
6: Benchmarks  
7: Util Tests  
main> 1  
  
Running TfLM Models menu  
  
TfLM Models  
-----  
1: Person Detection int8 model  
2: Micro Speech  
3: MLCommons Tiny V0.1 Keyword Spotting  
x: eXit to previous menu  
models> 3
```

```
Tests for kws model  
-----  
0: Run with "down" input  
1: Run with "go" input  
2: Run with "left" input  
g: Run golden tests (check for expected outputs)  
x: eXit to previous menu  
kws> g  
  
Running Run golden tests (check for expected outputs)  
...  
  
Copied 490 bytes at 0x400dd590  
Running kws  
.....  
"Event", "Tag", "Ticks"  
0, CONV_2D, 19841  
1, DEPTHWISE_CONV_2D, 4306  
2, CONV_2D, 11690  
3, DEPTHWISE_CONV_2D, 4226  
4, CONV_2D, 11662  
5, DEPTHWISE_CONV_2D, 4230  
6, CONV_2D, 11050  
7, DEPTHWISE_CONV_2D, 4757  
8, CONV_2D, 11905  
9, AVERAGE_POOL_2D, 237  
10, RESHAPE, 3  
11, FULLY_CONNECTED, 16  
12, SOFTMAX, 26  
Perf counters not enabled.  
86M ( 85996663) cycles total  
OK Golden tests passed
```

---

# CFU Playground Development

- Start a new project by copying a template:
    - `cp -r proj/proj_template proj/my_first_project`
  - Choose or bring the TFLite model
  - Use built-in profiling to identify time-consuming TF ops
  - Look at tensor shapes and other parameters, find opportunities to specialize/simplify the kernel(s)
  - Then, look for CFU acceleration opportunities
    - Design CFU, alter TFLM kernels to use the custom instructions
  - Measure speedup, repeat!
-

---

---

# Examples

---

# Simple Example

This Conv2D kernel consumes 76% of the execution time with model “pdti8”, so go after the computation in the innermost loop.

```
const int32_t input_offset = params.input_offset; // r = s(q - Z)

for (int batch = 0; batch < batches; ++batch) {
  for (int out_y = 0; out_y < output_height; ++out_y) {
    const int in_y_origin = (out_y * stride_height) - pad_height;
    for (int out_x = 0; out_x < output_width; ++out_x) {
      const int in_x_origin = (out_x * stride_width) - pad_width;
      for (int out_channel = 0; out_channel < output_depth; ++out_channel) {
        int32_t acc = 0;
        for (int filter_y = 0; filter_y < filter_height; ++filter_y) {
          const int in_y = in_y_origin + dilation_height_factor * filter_y;
          for (int filter_x = 0; filter_x < filter_width; ++filter_x) {
            const int in_x = in_x_origin + dilation_width_factor * filter_x;

            // Zero padding by omitting the areas outside the image.
            const bool is_point_inside_image =
              (in_x >= 0) && (in_x < input_width) && (in_y >= 0) &&
              (in_y < input_height);

            if (!is_point_inside_image) {
              continue;
            }

            for (int in_channel = 0; in_channel < input_depth; ++in_channel) {
              int32_t input_val = input_data[Offset(input_shape, batch, in_y,
                                                    in_x, in_channel)];

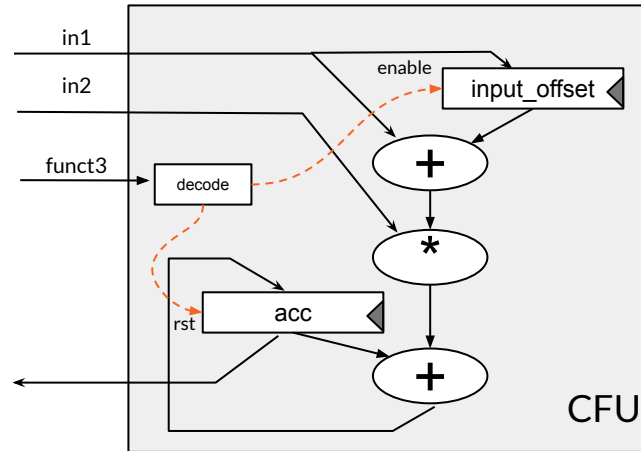
              int32_t filter_val = filter_data[Offset(
                filter_shape, out_channel, filter_y, filter_x, in_channel)];

              acc += filter_val * (input_val + input_offset);
            }
          }
        }
      }
    }
  }
}
(use acc)
```

# Simple Example

Loop invariant

`acc += filter_val * (input_val + input_offset)`





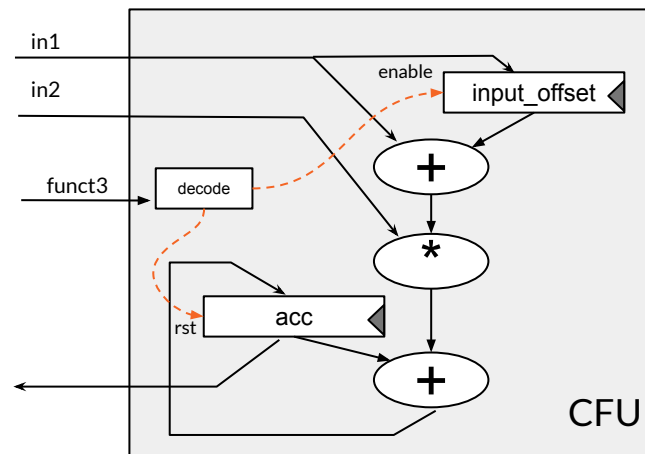
# Simple Example

```
//  
// The CFU computation  
//  
reg signed [31:0] input_offset; // state  
reg signed [31:0] acc; // state  
  
wire signed [31:0] filt_val = in1;  
wire signed [31:0] input_val = in2;  
  
wire signed [31:0] newacc = acc + (filt_val * (input_val + input_offset) );  
assign out = acc;  
  
always @(posedge clk) begin  
  if (cmd_valid) begin  
    if (funct3 == 3'b000) begin  
      input_offset <= in1;  
    end else if (funct3 == 3'b001) begin  
      acc <= in1;  
    end else if (funct3 == 3'b010) begin  
      acc <= newacc;  
    end  
  end  
end  
end
```

Verilog!

$acc += filter\_val * (input\_val + input\_offset)$

*Loop invariant*



# Simple Example

31% cycle reduction for CONV\_2D  
24% cycle reduction overall  
(model "pdti8")

```
const int32_t input_offset = params.input_offset; // r = s(q - Z)
// CFU: copy input_offset into the CFU
cfu_op(0, 0, input_offset, 0);
```

```
for (int batch = 0; batch < batches; ++batch) {
  for (int out_y = 0; out_y < output_height; ++out_y) {
    const int in_y_origin = (out_y * stride_height) - pad_height;
    for (int out_x = 0; out_x < output_width; ++out_x) {
      const int in_x_origin = (out_x * stride_width) - pad_width;
      for (int out_channel = 0; out_channel < output_depth; ++out_channel) {
```

```
        //int32_t acc = 0;
        // CFU: set the CFU internal acc to ZERO
        cfu_op(1, 0, 0, 0);
```

```
        for (int filter_y = 0; filter_y < filter_height; ++filter_y) {
          const int in_y = in_y_origin + dilation_height_factor * filter_y;
          for (int filter_x = 0; filter_x < filter_width; ++filter_x) {
            const int in_x = in_x_origin + dilation_width_factor * filter_x;
```

...

```
        for (int in_channel = 0; in_channel < input_depth; ++in_channel) {
          int32_t input_val = input_data[Offset(input_shape, batch, in_y,
                                               in_x, in_channel)];
          int32_t filter_val = filter_data[Offset(
            filter_shape, out_channel, filter_y, filter_x, in_channel)];
```

```
          // acc += filter_val * (input_val + input_offset);
          // CFU: add-multiply-accumulate in the CFU
          cfu_op(2, 0, filter_val, input_val);
        }
      }
    }
  }
```

```
    // CFU: retrieve final acc value from the CFU
    int32_t acc = cfu_op(3, 0, 0, 0);
```

# Simple Example

Add aliases for the custom instructions, for readability

31% cycle reduction for CONV\_2D  
24% cycle reduction overall  
(model "pdti8")

```
const int32_t input_offset = params.input_offset; // r = s(q - Z)
// CFU: copy input_offset into the CFU
cfu_init_offset(input_offset);

for (int batch = 0; batch < batches; ++batch) {
  for (int out_y = 0; out_y < output_height; ++out_y) {
    const int in_y_origin = (out_y * stride_height) - pad_height;
    for (int out_x = 0; out_x < output_width; ++out_x) {
      const int in_x_origin = (out_x * stride_width) - pad_width;
      for (int out_channel = 0; out_channel < output_depth; ++out_channel) {

        //int32_t acc = 0;
        // CFU: set the CFU internal acc to ZERO
        cfu_clear_acc();

        for (int filter_y = 0; filter_y < filter_height; ++filter_y) {
          const int in_y = in_y_origin + dilation_height_factor * filter_y;
          for (int filter_x = 0; filter_x < filter_width; ++filter_x) {
            const int in_x = in_x_origin + dilation_width_factor * filter_x;

            ...

            for (int in_channel = 0; in_channel < input_depth; ++in_channel) {
              int32_t input_val = input_data[Offset(input_shape, batch, in_y,
                                                    in_x, in_channel)];
              int32_t filter_val = filter_data[Offset(
                filter_shape, out_channel, filter_y, filter_x, in_channel)];

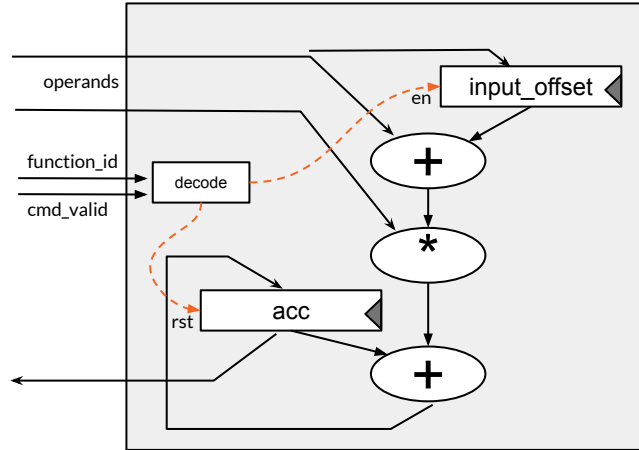
              // acc += filter_val * (input_val + input_offset);
              // CFU: add-multiply-accumulate in the CFU
              cfu_macc_with_offset(filter_val, input_val);
            }
          }
        }
      }
    }
  }

  // CFU: retrieve final acc value from the CFU
  int32_t acc = cfu_get_acc();
}
```

# Typical CFU evolution

Loop invariant

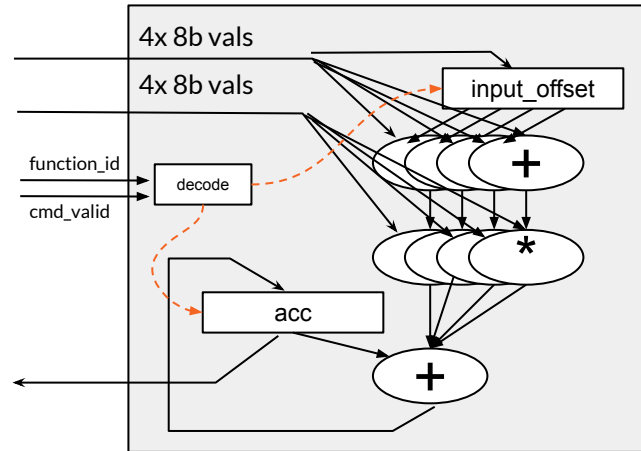
$acc += filter\_val * (input\_val + input\_offset)$



# Typical CFU evolution

*Loop invariant*

`acc += filter_val * (input_val + input_offset)`

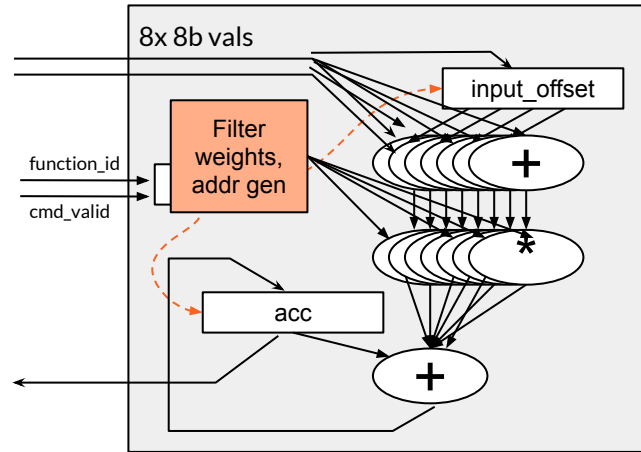


Exploit SIMD

# Typical CFU evolution

Loop invariant

$acc += filter\_val * (input\_val + input\_offset)$



Move re-used data (filter weights) into the CFU

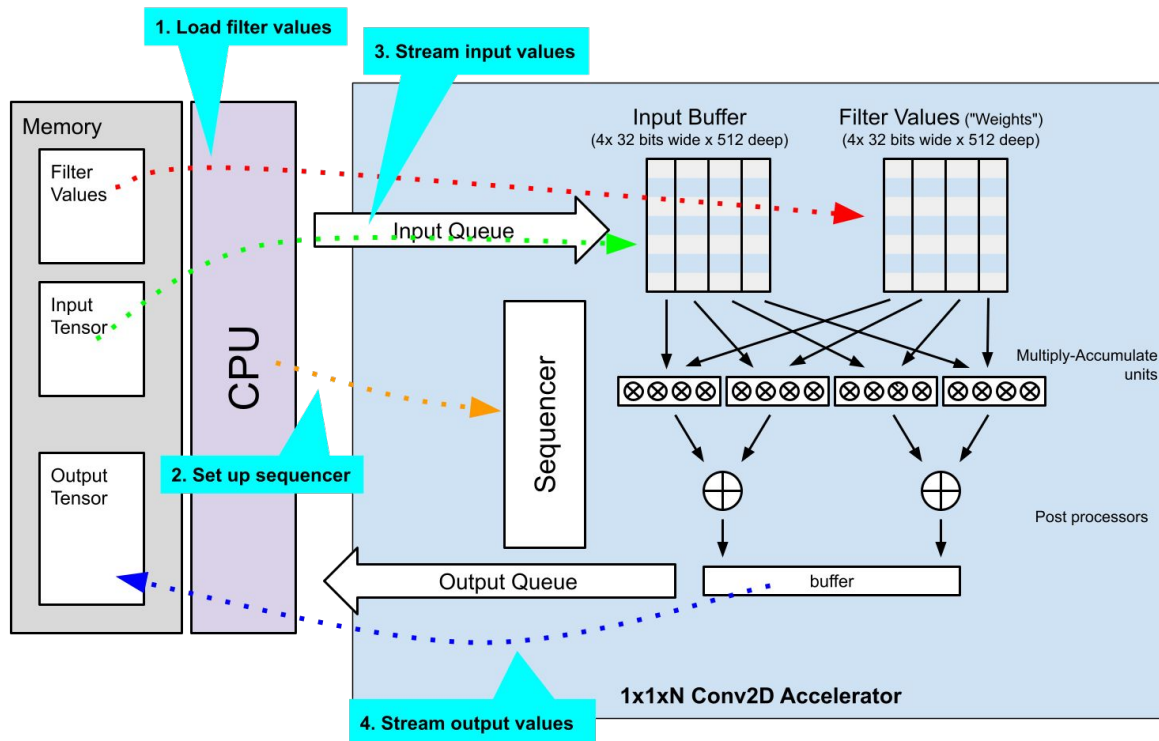
---

# Typical CFU evolution

And after a few more iterations, you've added:

- Local memories for both `input_vals` and `filter_vals`
  - Increased MACC parallelism
  - Sequencer for complete matrix x vector computation
  - Activation function & scaling
  - Double buffering at input and output buffers to pipeline data transfer and computation
-

# Typical CFU evolution





---

# MobileNet v2

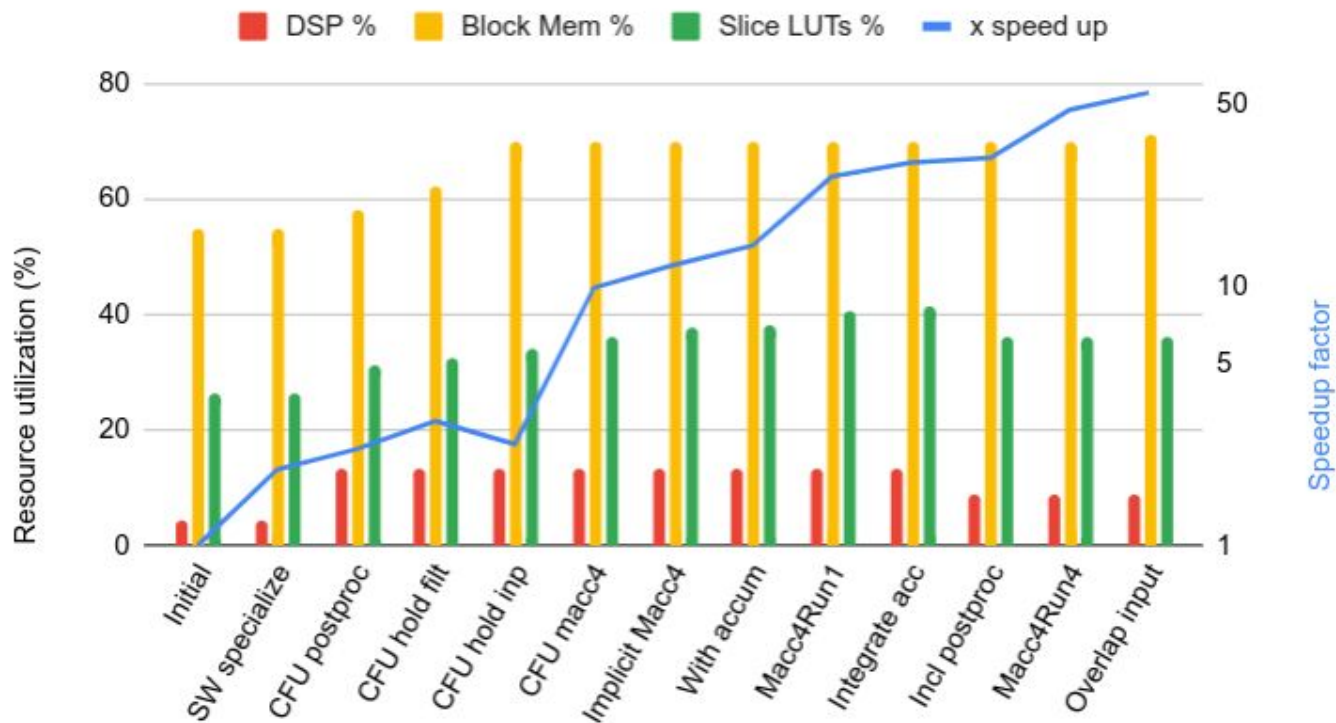
Before speeding up conv\_2d (1x1):

Totals

SOFTMAX	11510	(0.0M)
RESHAPE	21887	(0.0M)
FULLY_CONNECTED	48284	(0.0M)
AVERAGE_POOL_2D	487497	(0.5M)
ADD	4564330	(4.6M)
MUL	7236662	(7.2M)
SUB	16517877	(16.5M)
CONV_2D_3x3	95241303	(95.2M)
DEPTHWISE_CONV_2D	197214007	(197.2M)
CONV_2D_1x1	559817289	(559.8M)

---

# MobileNet v2



55x speedup!  
(on just conv2d 1x1)

But Amdahl's law...  
overall speedup is just  
2.8x

---

# MobileNet v2

After speeding up conv\_2d (1x1):

Totals

SOFTMAX	11503	(0.0M)
RESHAPE	21886	(0.0M)
FULLY_CONNECTED	50183	(0.1M)
AVERAGE_POOL_2D	494322	(0.5M)
ADD	4577562	(4.6M)
MUL	7233115	(7.2M)
CONV_2D_1x1	10263213	(10.3M)
SUB	16517493	(16.5M)
CONV_2D_3x3	76621863	(76.6M)
DEPTHWISE_CONV_2D	199485469	(199.5M)



Originally  
560M

---

---

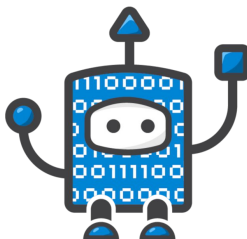
Example #2

Keyword Spotting  
on Fomu

---

# About ~~my~~ Intern Joey Bushagour's project

- Adding support for tiny FPGAs in CFU-Playground
- Fitting TensorFlow Lite for Microcontrollers on tiny FPGAs
- Making inference on tiny FPGAs faster



fomu.im

# 75× speedup on model inference

## How it ~~started~~:

```
Running MLCommons Tiny V0.1 Keyword Spotting
Error_reporter OK!
Input: 490 bytes, 4 dims: 1 49 10 1

Tests for kws model
=====
0: Run with "down" input
1: Run with "go" input
2: Run with "left" input
g: Run golden tests (check for expected outputs)
x: eXit to previous menu
kws> █
```

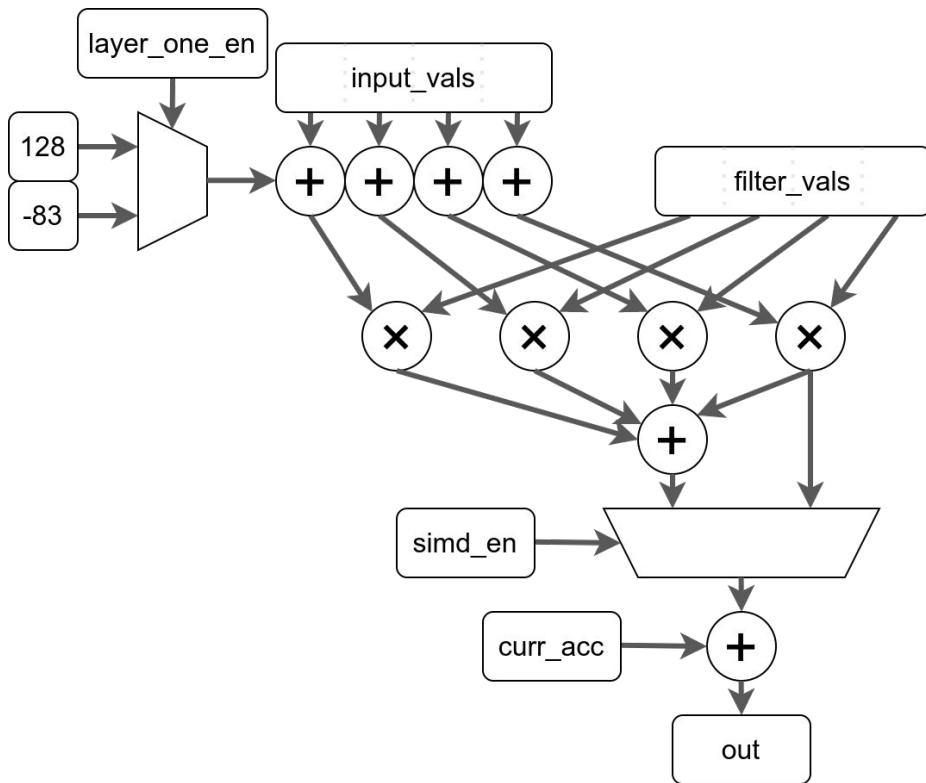
## How it's going:

```
Running MLCommons Tiny V0.1 Keyword Spotting
Error_reporter OK!
Input: 490 bytes, 4 dims: 1 49 10 1

Tests for kws model
=====
0: Run with "down" input
1: Run with "go" input
2: Run with "left" input
g: Run golden tests (check for expected outputs)
x: eXit to previous menu
kws> █
```

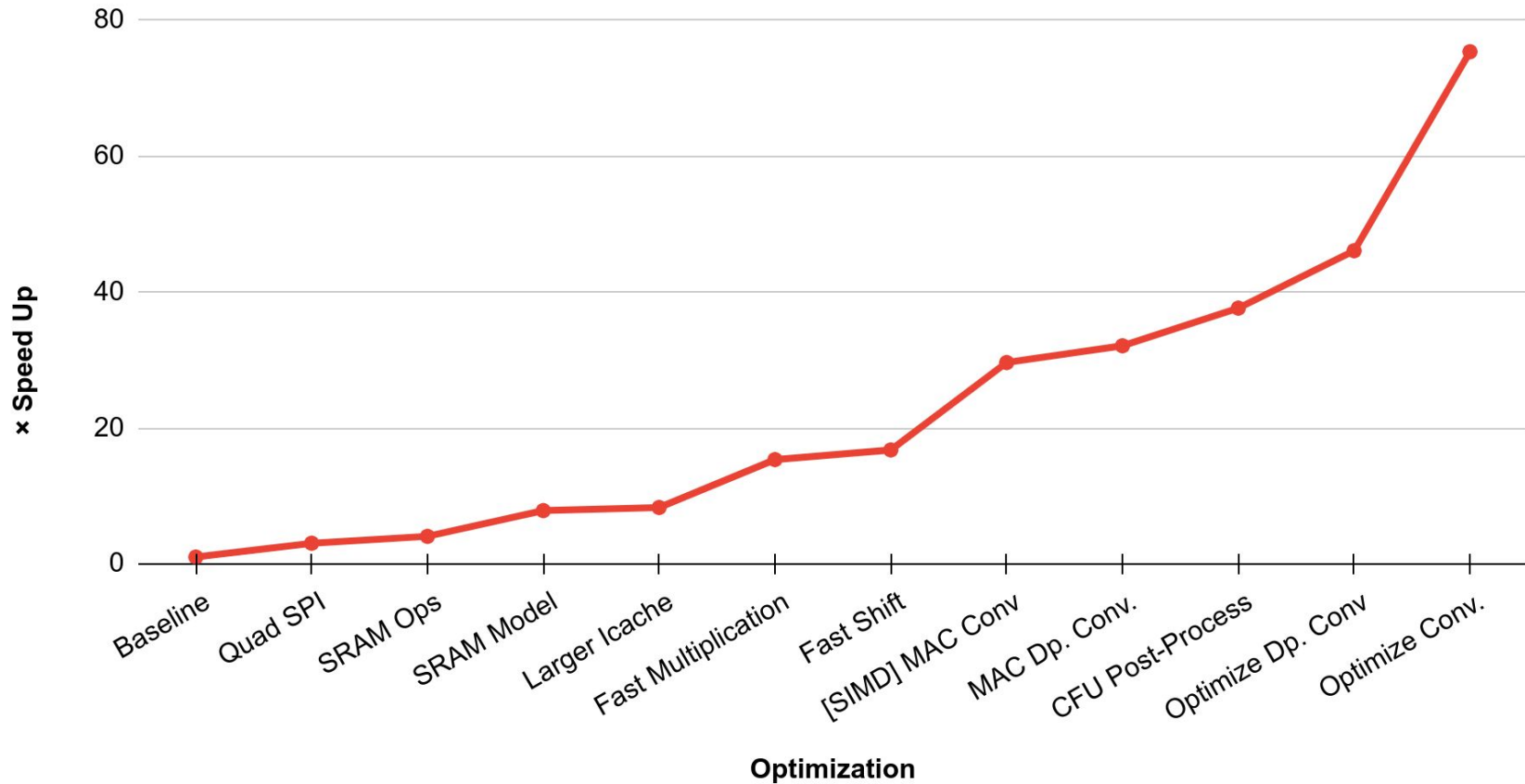
# Optimization: Use CFU SIMD MAC in TFLM ops

- Convolution and depthwise convolution are mostly multiply and accumulate
- Sometimes these 8 bit multiply and accumulates are contiguous in memory
- Our registers are 32 bits wide, we can vectorize where possible



Cumulative speedup: **32.10x**

## × Speed Up Over Optimizations





# What's left

- We have 75× speedup
- If we could find room to fit a branch predictor and bypass we'd have 102×
- Optimizing depthwise convolution more

```
Running MLCommons Tiny V0.1 Keyword Spotting
Error_reporter OK!
Input: 490 bytes, 4 dims: 1 49 10 1

Tests for kws model
=====
0: Run with "down" input
1: Run with "go" input
2: Run with "left" input
g: Run golden tests (check for expected outputs)
x: eXit to previous menu
kws> █
```

---

---

Wrap up

---

---

# Generating hardware from Python?

- Yes! We used nMigen (now “Amaranth HDL”) to build our large CFUs.
  - It is a Python library for generating HW
  - You still need to understand HW
  - But you can use Python conveniences (functions, loops, dictionaries, etc)
  - Google “`pyconline au 2021 cfu`” to find Alan’s presentation on YouTube.
-

---

# Join In The Fun

Clone it: [github.com/google/CFU-Playground](https://github.com/google/CFU-Playground)

- plenty of sample code and models to accelerate
- works with many LiteX supported boards (**check the wiki!**)

Docs: [cfu-playground.readthedocs.io](https://cfu-playground.readthedocs.io)

- introductions to nmigen, step-by-step guides, detailed documentation

Contact us:

- raise a github issue
  - mail us: [tcal@google.com](mailto:tcal@google.com), [avg@google.com](mailto:avg@google.com)
  - chat us: <https://gitter.im/CFU-Playground/community>
-

---

---

End

---



# Copyright Notice

This multimedia file is copyright © 2022 by tinyML Foundation. All rights reserved. It may not be duplicated or distributed in any form without prior written approval.

tinyML<sup>®</sup> is a registered trademark of the tinyML Foundation.

**[www.tinyml.org](http://www.tinyml.org)**



# Copyright Notice

This presentation in this publication was presented as a tinyML® Talks webcast. The content reflects the opinion of the author(s) and their respective companies. The inclusion of presentations in this publication does not constitute an endorsement by tinyML Foundation or the sponsors.

There is no copyright protection claimed by this publication. However, each presentation is the work of the authors and their respective companies and may contain copyrighted material. As such, it is strongly encouraged that any use reflect proper acknowledgement to the appropriate source. Any questions regarding the use of any materials presented should be directed to the author(s) or their companies.

tinyML is a registered trademark of the tinyML Foundation.

[www.tinyML.org](http://www.tinyML.org)