arm

CMSIS

CMSIS & ML Partner Event

embedded world 2026

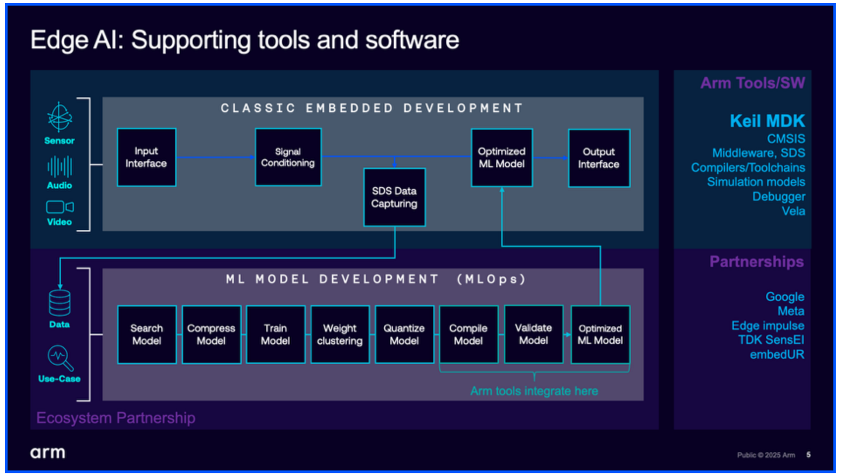
Arm MCU Tools/SW Team – Edge AI BU
Reinhard Keil

10 March 2026

Agenda

- Recap of 2025 and 2026 focus
- CMSIS Tool and Software Components
- Components (Tools, Software, Frameworks) for Edge AI
- Ecosystem partner ModelNova introducing Fusion Studio
- Compiler Commitment and Strategy for FuSa
- Summary

2025 Achievements + 2026 Focus



Edge AI: Supporting tools and software

Arm Tools/SW

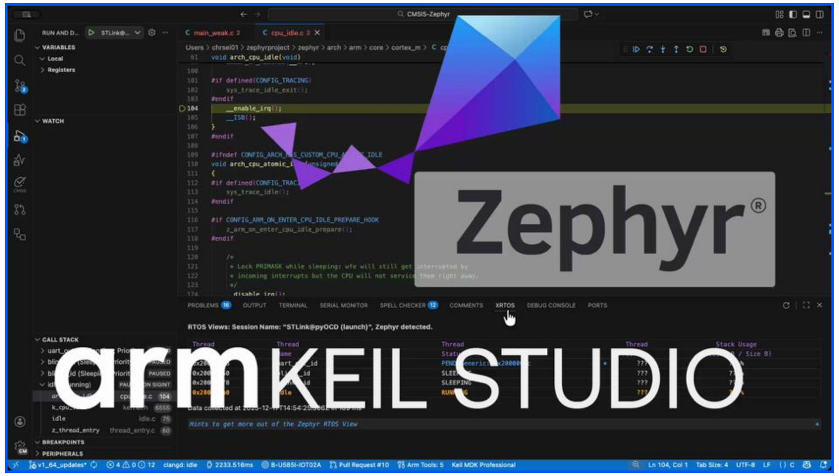
Keil MDK
CMSIS
Middleware, SDS
Compilers/Toolchains
Simulation models
Debugger
Vela

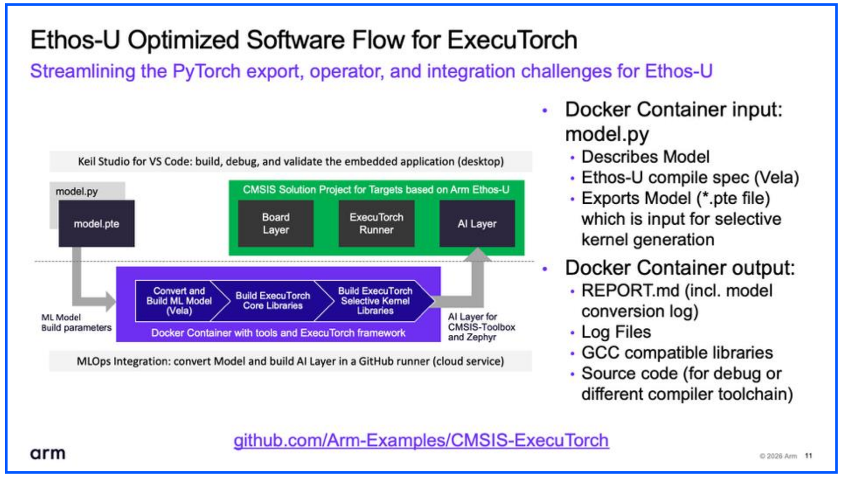
Partnerships

Google
Meta
Edge impulse
TDK SensEI
embedUR

arm

Public © 2025 Arm 5





Ethos-U Optimized Software Flow for ExecuTorch

Streamlining the PyTorch export, operator, and integration challenges for Ethos-U

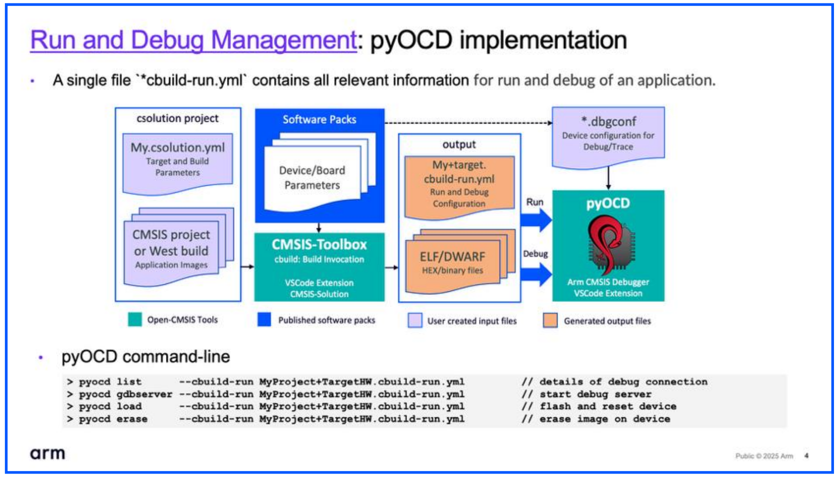
Keil Studio for VS Code: build, debug, and validate the embedded application (desktop)
model.py
model.pte
CMSIS Solution Project for Targets based on Arm Ethos-U
Board Layer
ExecuTorch Runner
AI Layer
ML Model Build parameters
Convert and Build ML Model (Vela)
Build ExecuTorch Core Libraries
Build ExecuTorch Selective Kernel Libraries
Docker Container with tools and ExecuTorch framework
AI Layer for CMSIS-Toolbox and Zephyr
MLOps Integration: convert Model and build AI Layer in a GitHub runner (cloud service)

- Docker Container input: model.py
  - Describes Model
  - Ethos-U compile spec (Vela)
  - Exports Model (*.pte file) which is input for selective kernel generation
- Docker Container output:
  - REPORT.md (incl. model conversion log)
  - Log Files
  - GCC compatible libraries
  - Source code (for debug or different compiler toolchain)

github.com/Arm-Examples/CMSIS-ExecuTorch

arm

© 2026 Arm 11



Run and Debug Management: pyOCD implementation

- A single file `*cbuild-run.yml` contains all relevant information for run and debug of an application.

- pyOCD command-line

```
> pyocd list      --cbuild-run MyProject+TargetHW.cbuild-run.yml        // details of debug connection
> pyocd gdbserver --cbuild-run MyProject+TargetHW.cbuild-run.yml        // start debug server
> pyocd load      --cbuild-run MyProject+TargetHW.cbuild-run.yml        // flash and reset device
> pyocd erase     --cbuild-run MyProject+TargetHW.cbuild-run.yml        // erase image on device
```

arm

Public © 2025 Arm 4

2026 Focus

- Ease-of-Use
- Edge AI
- Ecosystem
- Consistency
- VS Code Trace Views
- Zephyr
- Multi-core

Active Hardware IP – Edge AI BU

	Processors	
Cortex-A	Cortex-A55 Cortex-A53 Cortex-A35	Cortex-A720 Cortex-A520 Cortex-A320
Cortex-R	Cortex-R8 Cortex-R5	Cortex-R82
Cortex-M	Cortex-M33 Cortex-M23 Cortex-M0+, M4, M7	Cortex-M85 Cortex-M55 Cortex-M52
Ethos-U	Ethos-U55 Ethos-U65	Ethos-U85
Mali-ISP	Mali-C52 Mali-C32	Mali-C55
Corstone	Corstone-1000	Corstone-320/315 Corstone-300/310
System IP	DMA-250	DMA-350
	Available	

Edge AI is growing

Constrained Edge AI (RTX, FreeRTOS, Zephyr)

	Edge AI Deployed	Edge AI In Design
>1B Parameter		Cortex-A320 Ethos-U85 Mali-C55
<1B Parameter	Cortex-M55/M85 Ethos-U55/85 Mali-C55	

Linux Edge AI

	Edge AI Deployed	Edge AI In Design
Premium	Cortex-A78 Mali-C55 Mali-G57	Cortex-A720 Ethos-U85 Mali-C55 Mali-G510
Mainstream	Cortex-A55 Mali-C55 Mali-G31	Cortex-A520 Ethos-U85 Mali-C55 Mali-G310
Entry	Cortex-A35	Cortex-A320 Ethos-U85 Mali-C55
NPU	• Up to **256MACs/c**	• Up to **2048 MACs/c**

arm

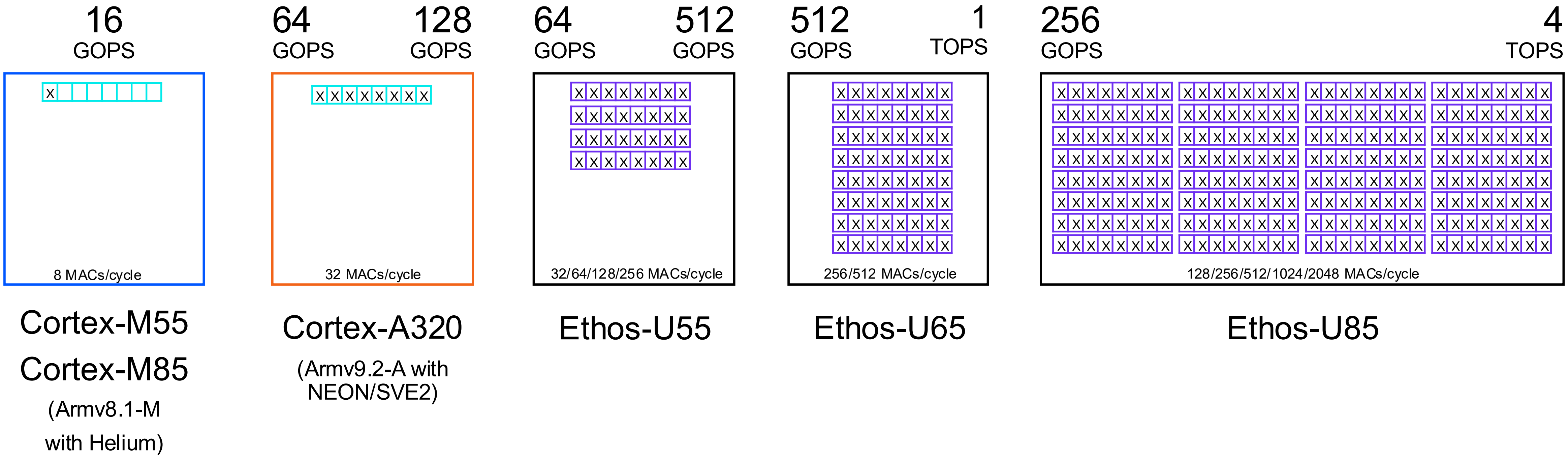
Arm IP: Unlocking the Full Potential of Edge AI

Accelerating implementation of higher performance devices

- Face recognition
- Image classification
- Keyword spotting

- Collaborative Robots
- Autonomous Visual Inspection
- Real-Time audio processing

- Small Language Models
- Transformers
- Multiple instances of same object



16 GOPS

8 MACs/cycle

Cortex-M55

Cortex-M85

(Armv8.1-M with Helium)

64 GOPS – 128 GOPS

32 MACs/cycle

Cortex-A320

(Armv9.2-A with NEON/SVE2)

64 GOPS – 512 GOPS

32/64/128/256 MACs/cycle

Ethos-U55

512 GOPS – 1 TOPS

256/512 MACs/cycle

Ethos-U65

256 GOPS – 4 TOPS

128/256/512/1024/2048 MACs/cycle

Ethos-U85

GOPS/TOPS expressed at iso-frequency (1 GHz)

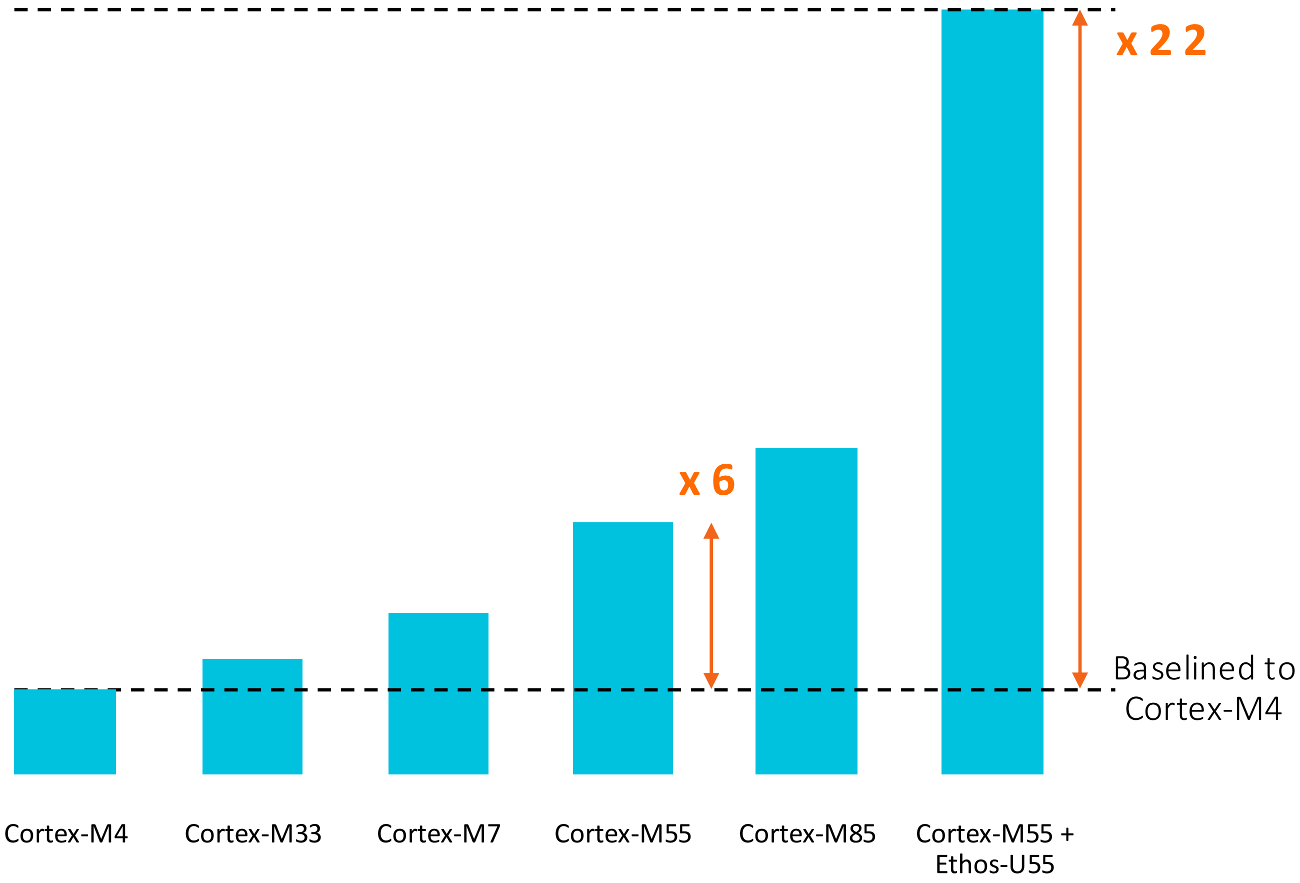
* Note that Cortex-A320 achieves significantly higher operating frequencies than Cortex-M CPUs

ML on Edge Devices = compute + libraries + tools

Arm delivers tools, standardized software for scaling Edge AI

- Helium and Ethos-U Benefits
  - More computations
  - New DSP/ML kernels

- Foundation Tools & Software
  - Simplify development
  - CMSIS-DSP/NN libraries, Python wrapper
  - Stream-based processing techniques
  - Open software and tools platform



EEMBC AudioMark = DSP + NN Workload

Key open-source communities

LLVM

Compiler Technology

Linux Foundation

Zephyr®

Rich Operating Systems

Eclipse Foundation

CDT

Debug and IDE Technology

AI/ML

TensorFlow

Software Frameworks for Machine Learning

Linaro

linaro™

Trusted Firmware and Foundation Tools

CMSIS

CMSIS

Open-CMSIS Pack

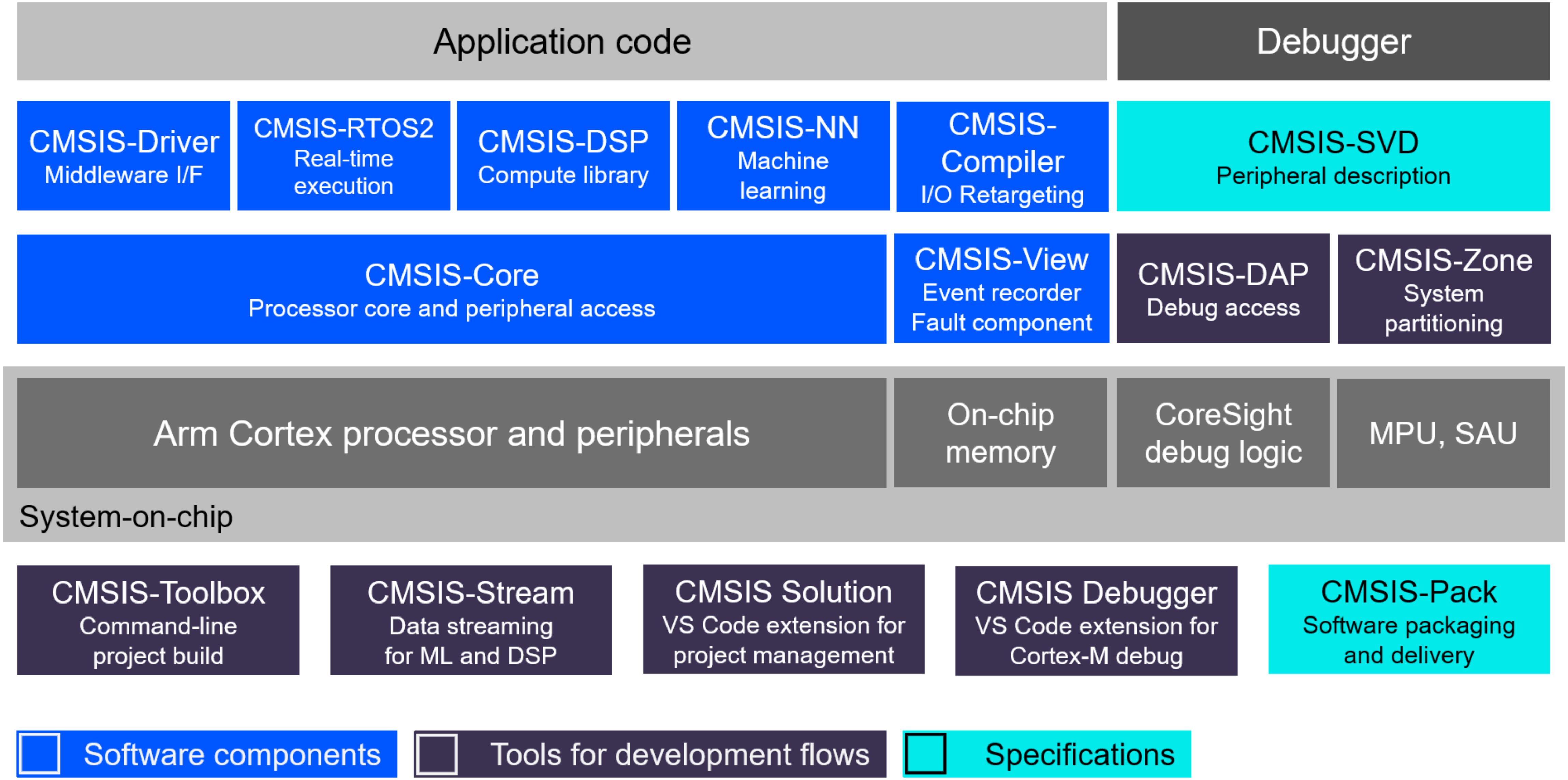
Foundation Software and Tools for Microcontroller

CMSIS Software and Tool Components

Joachim Krech

CMSIS Version 6

github.com/ARM-software/CMSIS_6



Application code

Debugger

CMSIS-Driver
Middleware I/F

CMSIS-RTOS2
Real-time execution

CMSIS-DSP
Compute library

CMSIS-NN
Machine learning

CMSIS-Compiler
I/O Retargeting

CMSIS-SVD
Peripheral description

CMSIS-Core
Processor core and peripheral access

CMSIS-View
Event recorder
Fault component

CMSIS-DAP
Debug access

CMSIS-Zone
System partitioning

Arm Cortex processor and peripherals

On-chip memory

CoreSight debug logic

MPU, SAU

System-on-chip

CMSIS-Toolbox
Command-line project build

CMSIS-Stream
Data streaming for ML and DSP

CMSIS Solution
VS Code extension for project management

CMSIS Debugger
VS Code extension for Cortex-M debug

CMSIS-Pack
Software packaging and delivery

🟦 Software components

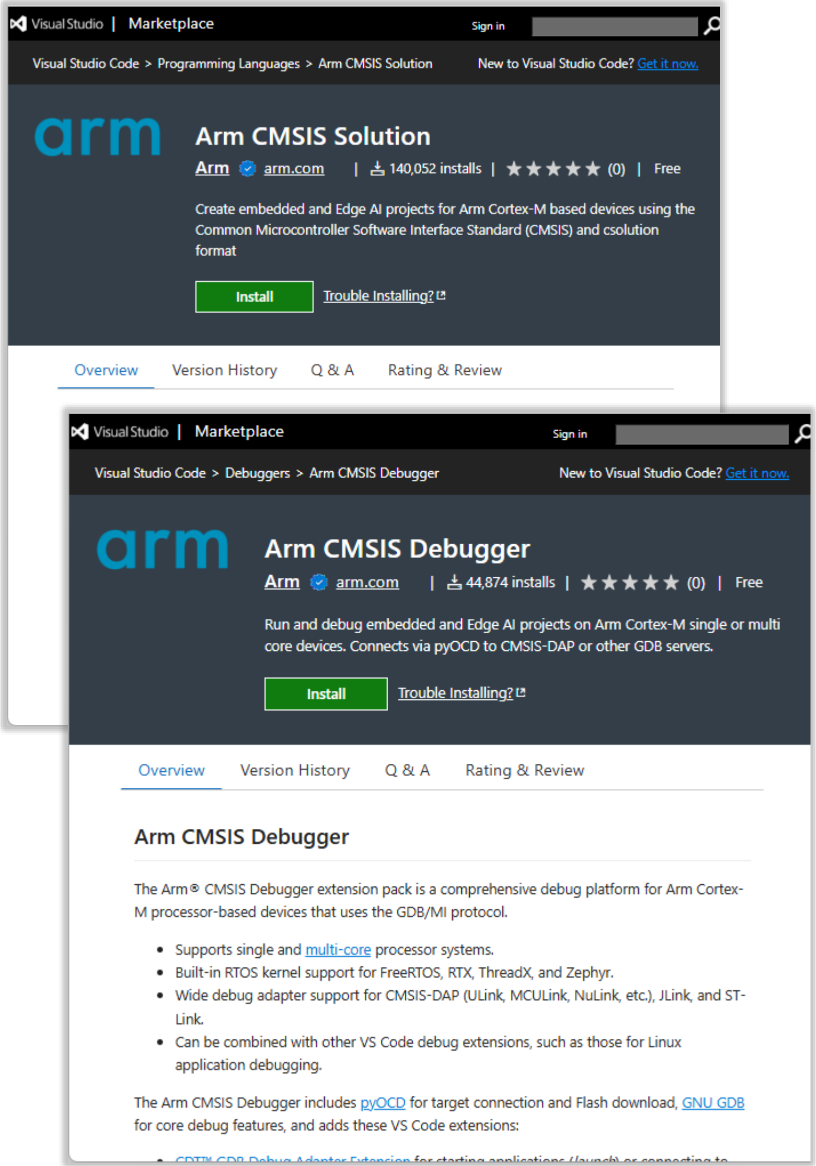
⬛ Tools for development flows

🟦 Specifications

Open-Source VS Code Extensions

Project & Build	Description	Provided by
CMSIS Solution	Create and Manage CMSIS based projects	Arm/Open-CMSIS-Pack
clangd	IntelliSense	LLVM
YAML	YAML Language Support	RedHat

Debug	Description	Provided by
CMSIS Debugger	Debug for Cortex-M processors	Arm/Open-CMSIS-Pack
CDT GDB Debug Adapter	Connector to GDB	Eclipse CDT Cloud
Memory Inspector	Memory Window	
Peripheral Inspector	SVD supported access to peripherals	



Visual Studio Code > Programming Languages > Arm CMSIS Solution

New to Visual Studio Code? Get it now.

Arm CMSIS Solution

Arm arm.com | 140,052 installs | ★★★★★ (0) | Free

Create embedded and Edge AI projects for Arm Cortex-M based devices using the Common Microcontroller Software Interface Standard (CMSIS) and csolution format

Visual Studio Code > Debuggers > Arm CMSIS Debugger

New to Visual Studio Code? Get it now.

Arm CMSIS Debugger

Arm arm.com | 44,874 installs | ★★★★★ (0) | Free

Run and debug embedded and Edge AI projects on Arm Cortex-M single or multi core devices. Connects via pyOCD to CMSIS-DAP or other GDB servers.

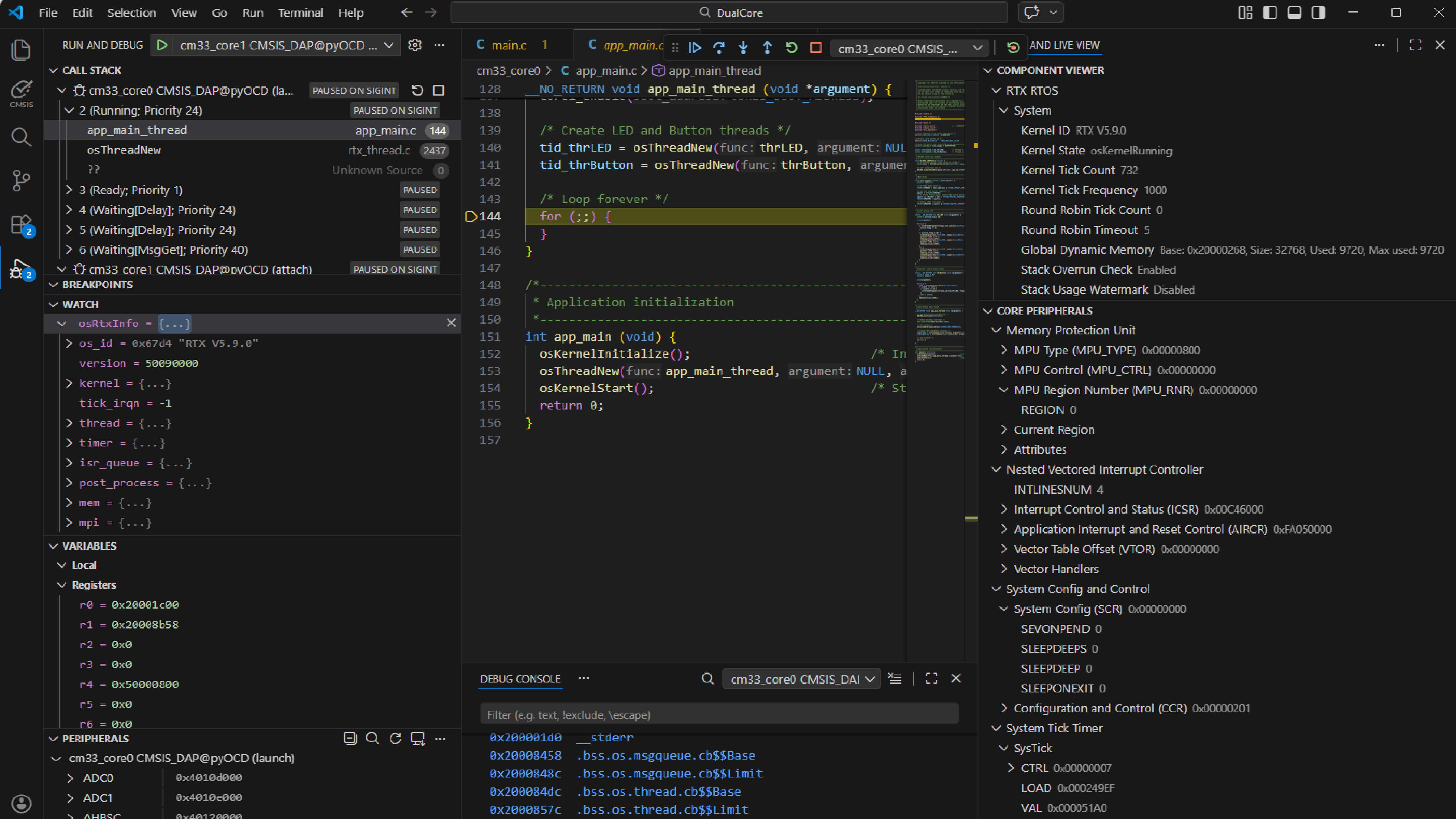
Arm CMSIS Debugger

The Arm® CMSIS Debugger extension pack is a comprehensive debug platform for Arm Cortex-M processor-based devices that uses the GDB/MI protocol.

- Supports single and multi-core processor systems.
- Built-in RTOS kernel support for FreeRTOS, RTX, ThreadX, and Zephyr.
- Wide debug adapter support for CMSIS-DAP (ULink, MCULink, NuLink, etc.), JLink, and ST-Link.
- Can be combined with other VS Code debug extensions, such as those for Linux application debugging.

The Arm CMSIS Debugger includes pyOCD for target connection and Flash download, GNU GDB for core debug features, and adds these VS Code extensions:

- CDT™ GDB Debug Adapter Extension for starting applications (launch) or connecting to



CALL STACK

```
cm33_core0 CMSIS_DAP@pyOCD (la...        PAUSED ON SIGINT
  2 (Running; Priority 24)               PAUSED ON SIGINT
    app_main_thread                      app_main.c  144
    osThreadNew                          rtx_thread.c  2437
    ??                                   Unknown Source  0
  3 (Ready; Priority 1)                  PAUSED
  4 (Waiting[Delay]; Priority 24)        PAUSED
  5 (Waiting[Delay]; Priority 24)        PAUSED
  6 (Waiting[MsgGet]; Priority 40)       PAUSED
cm33_core1 CMSIS_DAP@pyOCD (attach)      PAUSED ON SIGINT
```

BREAKPOINTS

WATCH

```
osRtxInfo = {...}
  os_id = 0x67d4 "RTX V5.9.0"
  version = 50090000
  kernel = {...}
  tick_irqn = -1
  thread = {...}
  timer = {...}
  isr_queue = {...}
  post_process = {...}
  mem = {...}
  mpi = {...}
```

VARIABLES

```
Local
Registers
  r0 = 0x20001c00
  r1 = 0x20008b58
  r2 = 0x0
  r3 = 0x0
  r4 = 0x50000800
  r5 = 0x0
  r6 = 0x0
```

PERIPHERALS

```
cm33_core0 CMSIS_DAP@pyOCD (launch)
  ADC0      0x4010d000
  ADC1      0x4010e000
  AHBSC     0x40120000
```

```c
__NO_RETURN void app_main_thread (void *argument) {

  /* Create LED and Button threads */
  tid_thrLED = osThreadNew(func: thrLED, argument: NUL
  tid_thrButton = osThreadNew(func: thrButton, argumer

  /* Loop forever */
  for (;;) {
  }
}

/*-----------------------------------------------------------
 * Application initialization
 *-----------------------------------------------------------

int app_main (void) {
  osKernelInitialize();                             /* In
  osThreadNew(func: app_main_thread, argument: NULL, a
  osKernelStart();                                  /* St
  return 0;
}
```

DEBUG CONSOLE

cm33_core0 CMSIS_DAI

Filter (e.g. text, !exclude, \escape)

```
0x200001d0  __stderr
0x20008458  .bss.os.msgqueue.cb$$Base
0x2000848c  .bss.os.msgqueue.cb$$Limit
0x200084dc  .bss.os.thread.cb$$Base
0x2000857c  .bss.os.thread.cb$$Limit
```

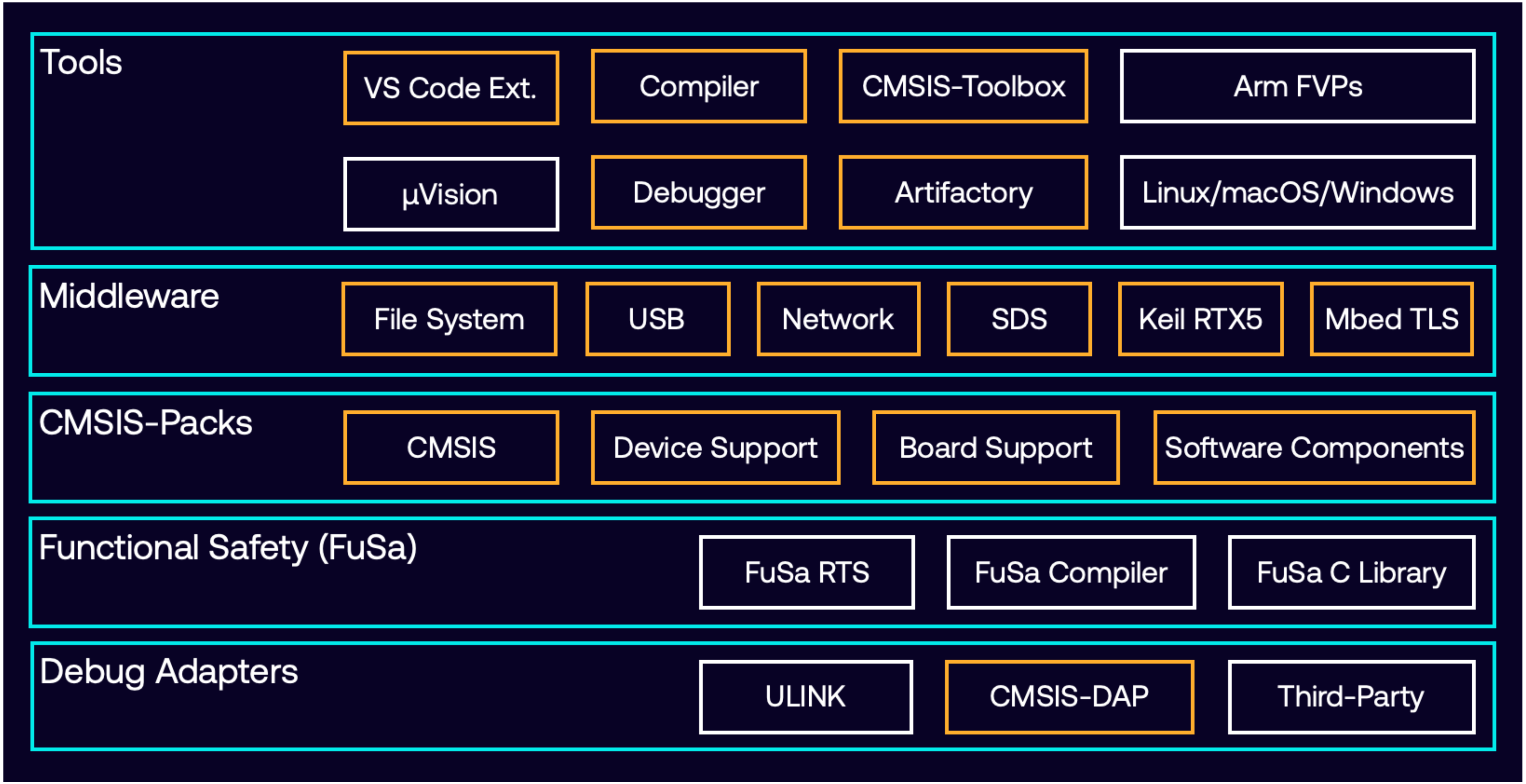
COMPONENT VIEWER

```
RTX RTOS
  System
    Kernel ID  RTX V5.9.0
    Kernel State  osKernelRunning
    Kernel Tick Count  732
    Kernel Tick Frequency  1000
    Round Robin Tick Count  0
    Round Robin Timeout  5
    Global Dynamic Memory  Base: 0x20000268, Size: 32768, Used: 9720, Max used: 9720
    Stack Overrun Check  Enabled
    Stack Usage Watermark  Disabled
```

CORE PERIPHERALS

```
Memory Protection Unit
  MPU Type (MPU_TYPE)  0x00000800
  MPU Control (MPU_CTRL)  0x00000000
  MPU Region Number (MPU_RNR)  0x00000000
    REGION  0
  Current Region
  Attributes
Nested Vectored Interrupt Controller
  INTLINESNUM  4
  Interrupt Control and Status (ICSR)  0x00C46000
  Application Interrupt and Reset Control (AIRCR)  0xFA050000
  Vector Table Offset (VTOR)  0x00000000
  Vector Handlers
System Config and Control
  System Config (SCR)  0x00000000
    SEVONPEND  0
    SLEEPDEEPS  0
    SLEEPDEEP  0
    SLEEPONEXIT  0
  Configuration and Control (CCR)  0x00000201
System Tick Timer
  SysTick
    CTRL  0x00000007
    LOAD  0x000249EF
    VAL  0x000051A0
```

Keil MDK Version 6 – Commitment for Professional Users



Tools

VS Code Ext. | Compiler | CMSIS-Toolbox | Arm FVPs

µVision | Debugger | Artifactory | Linux/macOS/Windows

Middleware

File System | USB | Network | SDS | Keil RTX5 | Mbed TLS

CMSIS-Packs

CMSIS | Device Support | Board Support | Software Components

Functional Safety (FuSa)

FuSa RTS | FuSa Compiler | FuSa C Library

Debug Adapters

ULINK | CMSIS-DAP | Third-Party

Open-source

Arm base software investment benefits the entire eco-system

Base software under permissive open-source license maintained by Arm

CMSIS-RTOS2 Real-time execution	CMSIS-Driver Middleware I/F	CMSIS-DSP Compute library	CMSIS-NN Machine learning	CMSIS-Core Device Access
Keil RTX RTOS Kernel	CMSIS-View Event recorder	CMSIS-Compiler I/O Retargeting	LiteRT (TFLu) TensorFlow Edge AI	ExecuTorch PyTourch Edge AI
CMSIS-FreeRTOS RTOS Kernel	MDK Middleware	mbedTLS Security	Trusted Firmware	Arm-2D Image Processing

■ 3rd party collaboration, ExecuTorch Q4'25

Extends FreeRTOS Kernel with FreeRTOS core libraries and AWS IoT libraries

Designed for resource-constrained devices, supporting hundreds of hardware platforms

12K+ Devices supported

Device Family Pack (DFP)
provided by Silicon Partners describes memory, peripherals, debug topology. Includes Flash algorithms and optional peripheral drivers.

Compiler & Debugger support for the entire ecosystem

CMSIS reference applications – board layer simplifies porting

MDK-Middleware contains software components for IPv4 and IPv6 networking, USB Host and Device communication, and file system for data storage.

Synchronous Data Stream (SDS) Framework implements a data stream management, provides methods and helper tools for developing and optimizing embedded applications that integrate DSP and ML algorithms.

Machine Learning Evaluation Kit contains examples that showcase audio and video ML models. Includes a template for user ML model development with instrumentation for performance measurement on hardware or FVP simulation.

MDK Middleware Examples

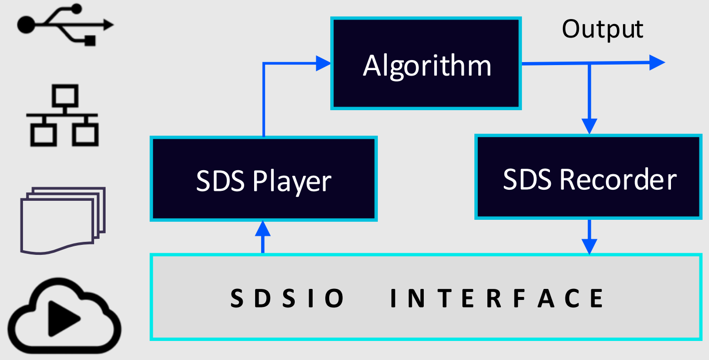
Reference Application

USB Device

USB Host

Network

File System



mlek_audio.csolution

Example and Template for ML model development

kws.cproject	user.cproject
Keyword Spotting	Framework for user algorithms

Board Support Pack (BSP)

extends a DFP with board specific software. Providing a board layer enables several different software stacks.

Driver APIs

Layer Type: Board
(<board-name>.clayer.yml)

Developer Choices: From bare-metal to RTOS-based systems

CMSIS components integrate into a diverse range of embedded systems

Bare-Metal

- Minimal overhead, highest determinism
- Small memory footprint
- Direct hardware control

Bare-metal offers maximum determinism and minimal overhead, but it does not scale well.

As scheduling and synchronization must be handled manually, making systems harder to maintain and debug over time.

FreeRTOS / Keil RTX (FuSa RTS)

- Predictable real-time behaviour
- Modular design – tasks are easier to isolate/maintain
- Rich ecosystem and MCU-friendly memory footprint

RTX or FreeRTOS adds some memory and runtime overhead and but simplifies task synchronization.

RTX is transparent to the interrupt system while FreeRTOS requires careful interrupt priority configuration.

Zephyr

- Scalable architecture with extensive device-driver support
- Unified build system and configuration
- Ideal for connected, feature-rich devices

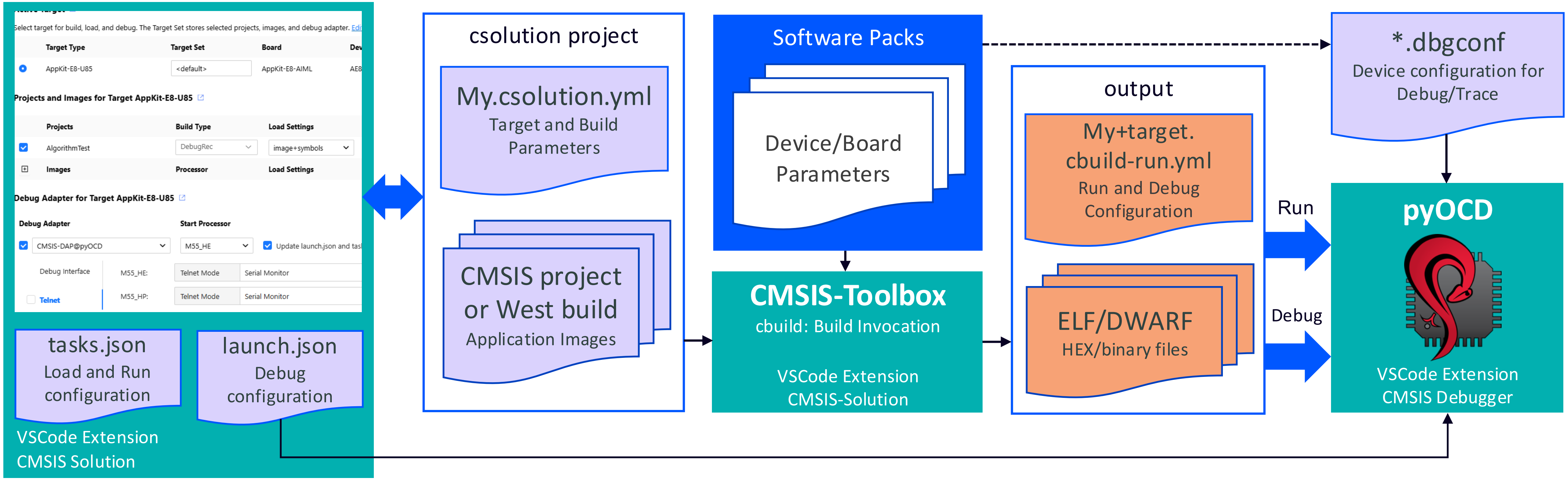
Zephyr provides strong structure and scalability, but with a larger footprint and steeper learning curve.

Its abstraction layers can feel heavyweight for small or tightly hardware-coupled applications.

github.com/Arm-Examples#rtos-applications

Run and Debug Management: VS Code + pyOCD implementation

- A single file `*cbuild-run.yml` contains all relevant information for run and debug of an application.



Select target for build, load, and debug. The Target Set stores selected projects, images, and debug adapter. Edi

	Target Type	Target Set	Board	Dev
●	AppKit-E8-U85	<default>	AppKit-E8-AIML	AE8

Projects and Images for Target AppKit-E8-U85

	Projects	Build Type	Load Settings
☑	AlgorithmTest	DebugRec	image+symbols
⊞	**Images**	**Processor**	**Load Settings**

Debug Adapter for Target AppKit-E8-U85

Debug Adapter	Start Processor	
☑ CMSIS-DAP@pyOCD	M55_HE	☑ Update launch.json and tas

Debug Interface			
	M55_HE:	Telnet Mode	Serial Monitor
□ Telnet	M55_HP:	Telnet Mode	Serial Monitor

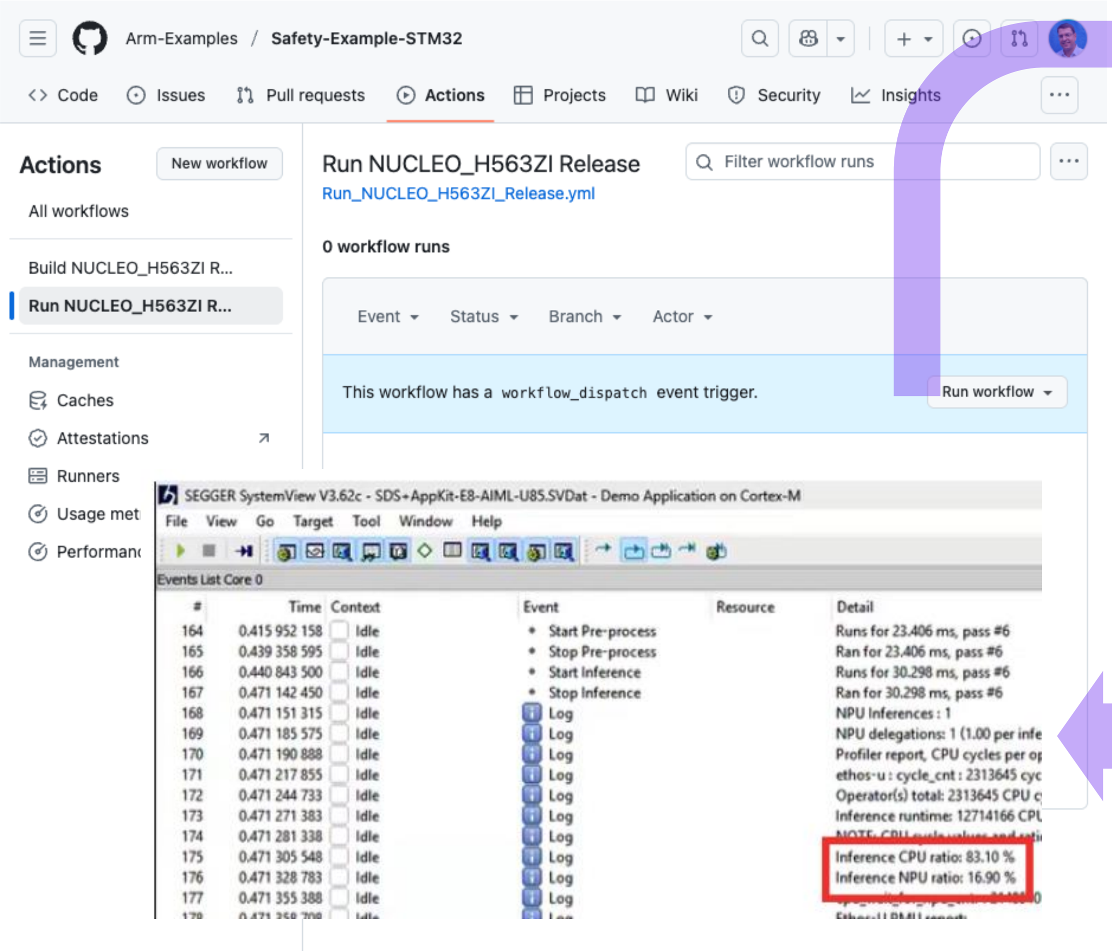
tasks.json
Load and Run configuration
launch.json
Debug configuration
VSCode Extension CMSIS Solution
csolution project
My.csolution.yml
Target and Build Parameters
CMSIS project or West build
Application Images
Software Packs
Device/Board Parameters
CMSIS-Toolbox
cbuild: Build Invocation
VSCode Extension CMSIS-Solution
output
My+target. cbuild-run.yml
Run and Debug Configuration
ELF/DWARF
HEX/binary files
*.dbgconf
Device configuration for Debug/Trace
Run
Debug
pyOCD
VSCode Extension CMSIS Debugger

- pyOCD command-line

```
> pyocd list      --cbuild-run MyProject+TargetHW.cbuild-run.yml   // details of debug connection (scan JTAG, SWD)
> pyocd gdbserver --cbuild-run MyProject+TargetHW.cbuild-run.yml   // start debug server, load command: flash
> pyocd run       --cbuild-run MyProject+TargetHW.cbuild-run.yml   // run application
> pyocd erase     --cbuild-run MyProject+TargetHW.cbuild-run.yml   // erase image on device
```

Continuous Integration (CI) and Hardware-in-the-Loop (HiL)

1. CI Build on GitHub-hosted Runner



Arm-Examples / **Safety-Example-STM32**

Actions

All workflows

Build NUCLEO_H563ZI R...

Run NUCLEO_H563ZI R...

Management

Caches

Attestations

Runners

Usage met

Performanc

Run NUCLEO_H563ZI Release
Run_NUCLEO_H563ZI_Release.yml

0 workflow runs

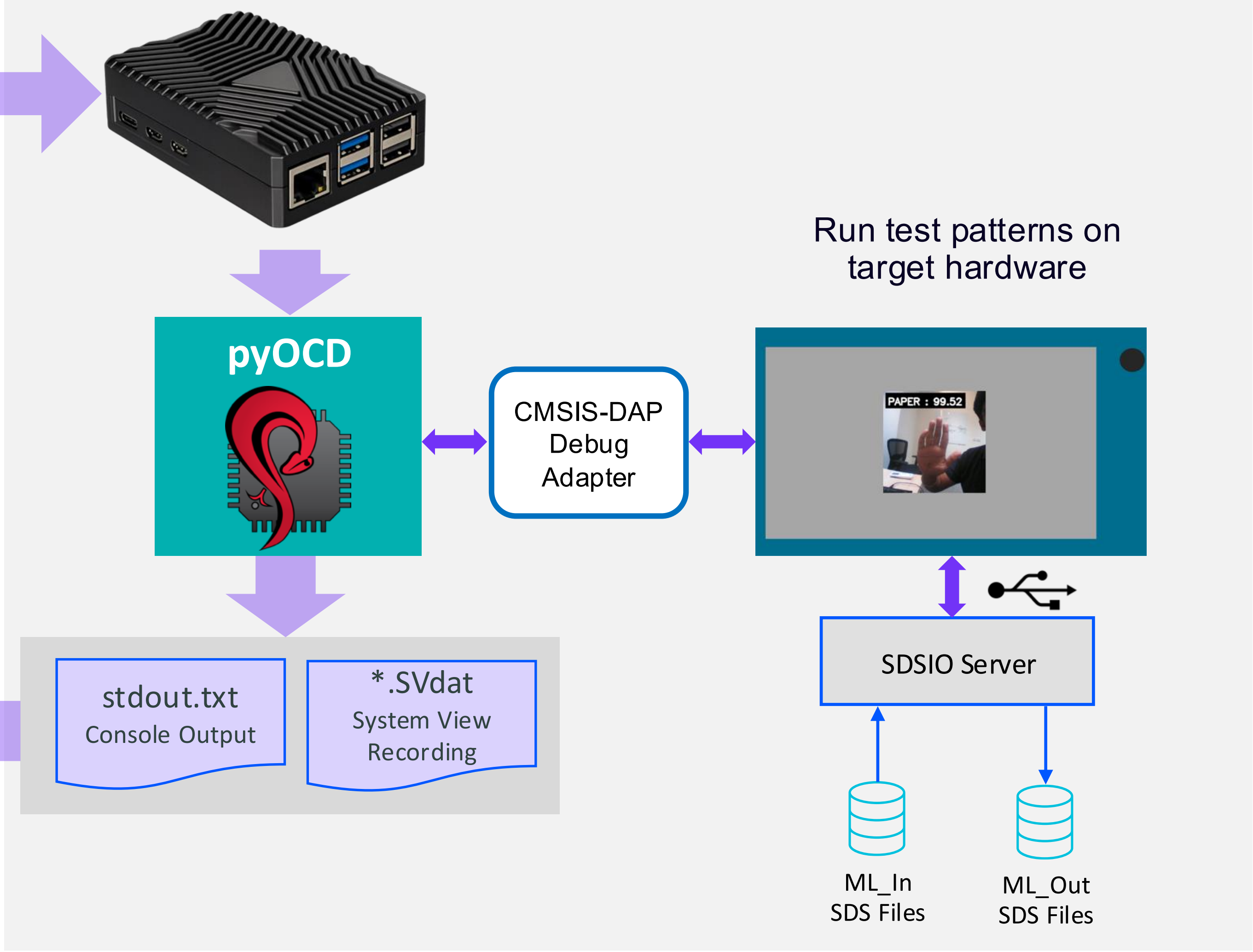
This workflow has a `workflow_dispatch` event trigger.

SEGGER SystemView V3.62c - SDS+AppKit-E8-AIML-U85.SVDat - Demo Application on Cortex-M

Events List Core 0

#	Time	Context	Event	Resource	Detail
164	0.415 952 158	Idle	Start Pre-process		Runs for 23.406 ms, pass #6
165	0.439 358 595	Idle	Stop Pre-process		Ran for 23.406 ms, pass #6
166	0.440 843 500	Idle	Start Inference		Runs for 30.298 ms, pass #6
167	0.471 142 450	Idle	Stop Inference		Ran for 30.298 ms, pass #6
168	0.471 151 315	Idle	Log		NPU Inferences : 1
169	0.471 185 575	Idle	Log		NPU delegations: 1 (1.00 per infe
170	0.471 190 888	Idle	Log		Profiler report, CPU cycles per op
171	0.471 217 855	Idle	Log		ethos-u : cycle_cnt : 2313645 cyc
172	0.471 244 733	Idle	Log		Operator(s) total: 2313645 CPU c
173	0.471 271 383	Idle	Log		Inference runtime: 12714166 CPU
174	0.471 281 338	Idle	Log		NOTE: CPU cycle values and rati
175	0.471 305 548	Idle	Log		Inference CPU ratio: 83.10 %
176	0.471 328 783	Idle	Log		Inference NPU ratio: 16.90 %
177	0.471 355 388	Idle	Log		[illegible]
178	0.471 358 708	Idle	Log		Ethos-U PMU report:

2. Custom Runner with pyOCD



Run test patterns on target hardware

Tools and Feature Roadmap

	H2'25	H1'26	H2'26
Keil MDK Version 6 Commercial software development solution for embedded, IoT, and Edge AI applications. Includes FuSa and support for VS Code, CMSIS, MDK-Middleware.	**µVision** 🟦 v5.43 **Tools Artifactory** 🟦 **FuSa RTS** 🟦 v1.2.0 **MDK Middleware** 🟦 v8.1.0	Release cadence: µVision on-demand, MDK Middleware, FuSa RTS 12 months Cortex-M Trace/LiveView combined with multi-core debug. Mature overall product. Work on examples and documentation.	Transition away from keil.com Cortex-M/Linux app development Introducing ATfE (LLVM)
VS Code Extensions CMSIS-Pack based software development with CLI and IDE workflows Compiler: AC6, GCC, IAR, LLVM. Free Integration in 3rd party tool solutions.	**CMSIS Toolbox** 🟦 v2.13 **CMSIS Solution** 🟦 v1.66 **CMSIS Debugger** 🟦 v1.4.1 Windows/Linux/macOS, DevOps	Release cadence: 2 months Add Component Viewer and start with core peripherals. Implement LiveView for system status, event recorder, SWO, and RTT.	Implement trace features. Investigate in Cortex-A debug and instruction trace features.
Compiler Broad support for all popular toolchains. Arm Compiler 6 continues in maintenance mode. New LLVM (based on open-source) compiler will be supported by Arm including FuSa qualifications.	**Arm Comp for Emb. (AC6)** 🟦 v6.24 **AC6 FuSa** 🟦 v6.22.2 **ATfE (LLVM)** 🟦 v21.1.1 **GCC** 🟦 v15.2.1	Release cadence: ATfE, GCC 6 months. AC6 FuSa depends on issues Provide ATfE with equal or better performance than AC6 for Helium and SVE2 (Cortex-A320) targets. Transition to LLVM libc and libc+.	Further code size and performance optimizations on ATfE. Helium optimized mem routines.
ML Embedded Platform SW Runtime: CMSIS-NN library Platforms: MLEK for FVP and hardware Vela: compiles ML models for Ethos-U System Analysis TOSA and ExecuTorch	**CMSIS-NN Runtime** 🟦 v7.0 **ML Platform** 🟦 25.11 **Vela** 🟦 v4.5.0	Release cadence: 4 months, CMSIS-NN 12 months Full Ethos support in ExecuTorch Ethos-U55 direct drive for Cortex-A Provisions for measuring performance	Zephyr integration CMSIS-NN ExecuTorch support Debug analysis improvements

🟦 Production version delivered

Components for Edge AI

Matthias Hertel

Edge AI Frameworks for Cortex-M and Ethos-U targets

TensorFlow LiteRT

LiteRT (TensorFlow Lite Runtime)

Production-grade inference runtime optimized for Cortex-M microcontrollers optional with Ethos-U

- Proven for Cortex-M only and Cortex-M + Ethos-U
- Optimized kernels for constrained memory
- Stable operator coverage for classic ML models
- Strong ecosystem, ready today

PyTorch

ExecuTorch (Lightweight PyTorch Runtime)

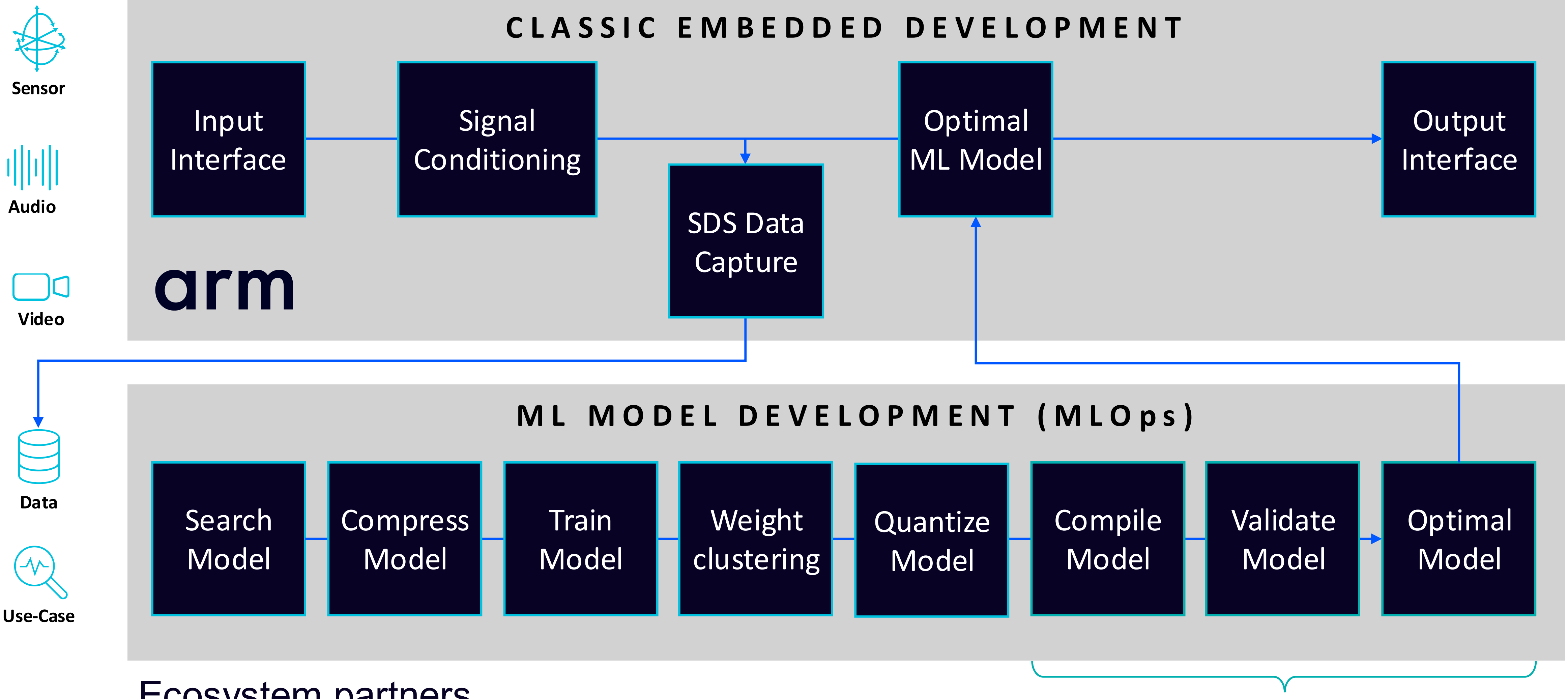
Dominant framework in research, making new ML model development and sharing easier.

- Strong for LLMs, vision, multimodal, generative AI
- ML developer friendly with modern tooling
- Rapidly growing eco-system momentum
- Maturing for Cortex-M + Ethos-U targets

Industry momentum shifts towards **PyTorch-first workflows**, with deployment maturing

Development flow for optimized Edge AI devices

Arm provides tools for classic embedded and back-end tools for MLOps



Sensor

Audio

Video

Data

Use-Case

CLASSIC EMBEDDED DEVELOPMENT

Input Interface

Signal Conditioning

SDS Data Capture

Optimal ML Model

Output Interface

arm

ML MODEL DEVELOPMENT (MLOps)

Search Model

Compress Model

Train Model

Weight clustering

Quantize Model

Compile Model

Validate Model

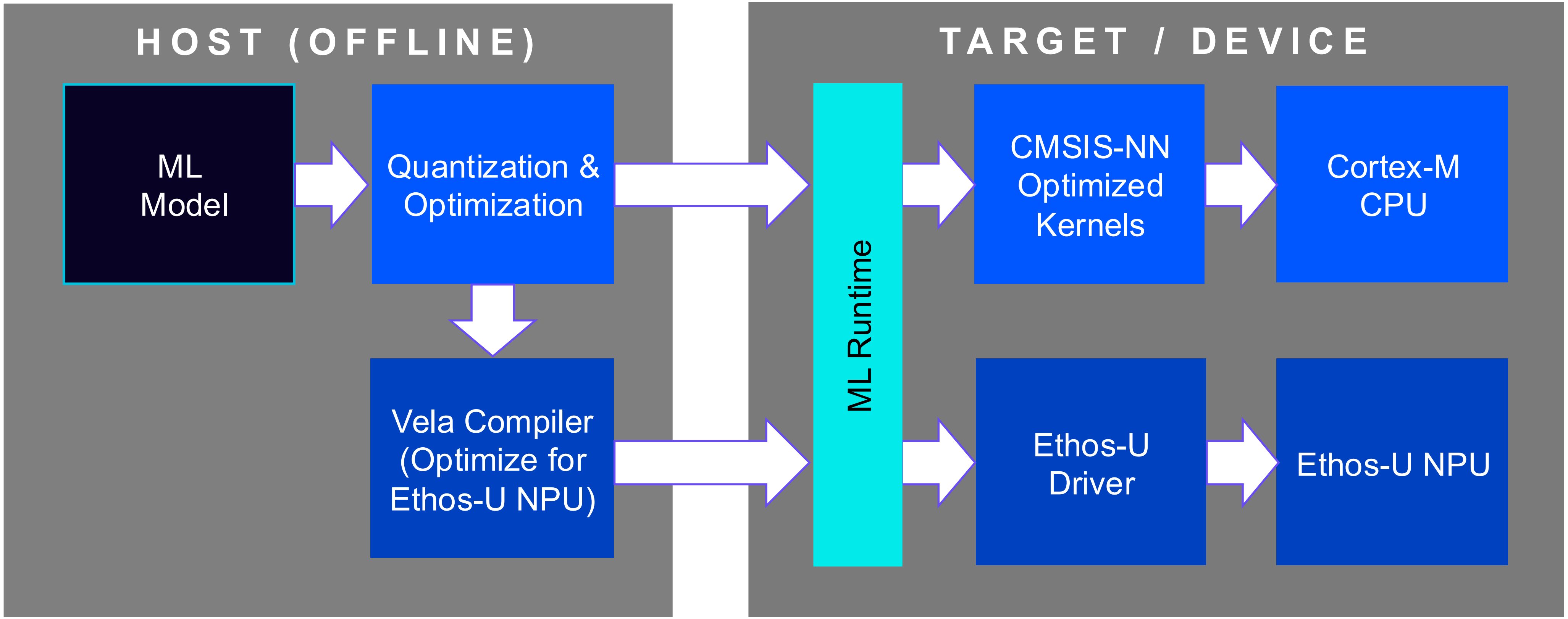
Optimal Model

Ecosystem partners

Arm tools integrate here

Ethos-U Optimized Software Flow for Tensorflow/PyTorch

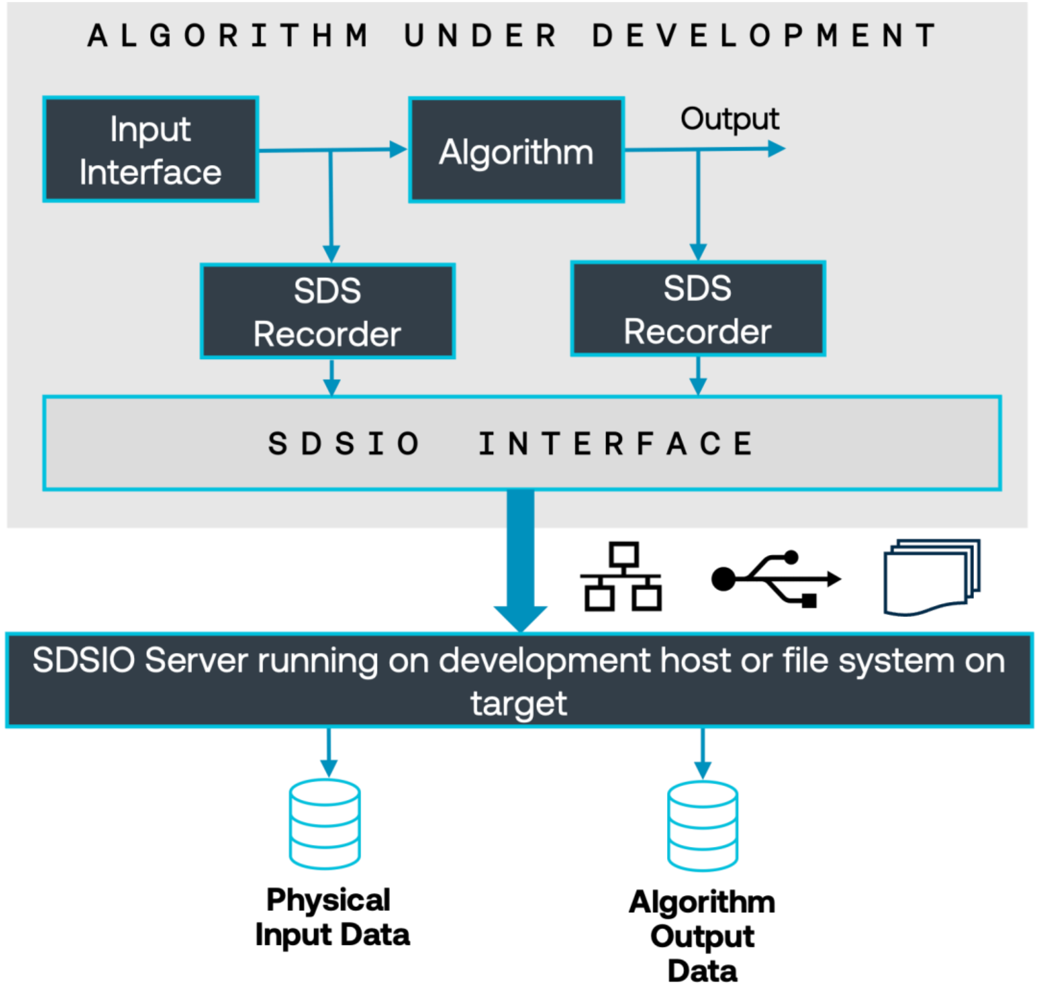
github.com/Arm-Examples#edge-ai-and-machine-learning


HOST (OFFLINE)
ML Model
Quantization & Optimization
Vela Compiler (Optimize for Ethos-U NPU)
TARGET / DEVICE
ML Runtime
CMSIS-NN Optimized Kernels
Cortex-M CPU
Ethos-U Driver
Ethos-U NPU

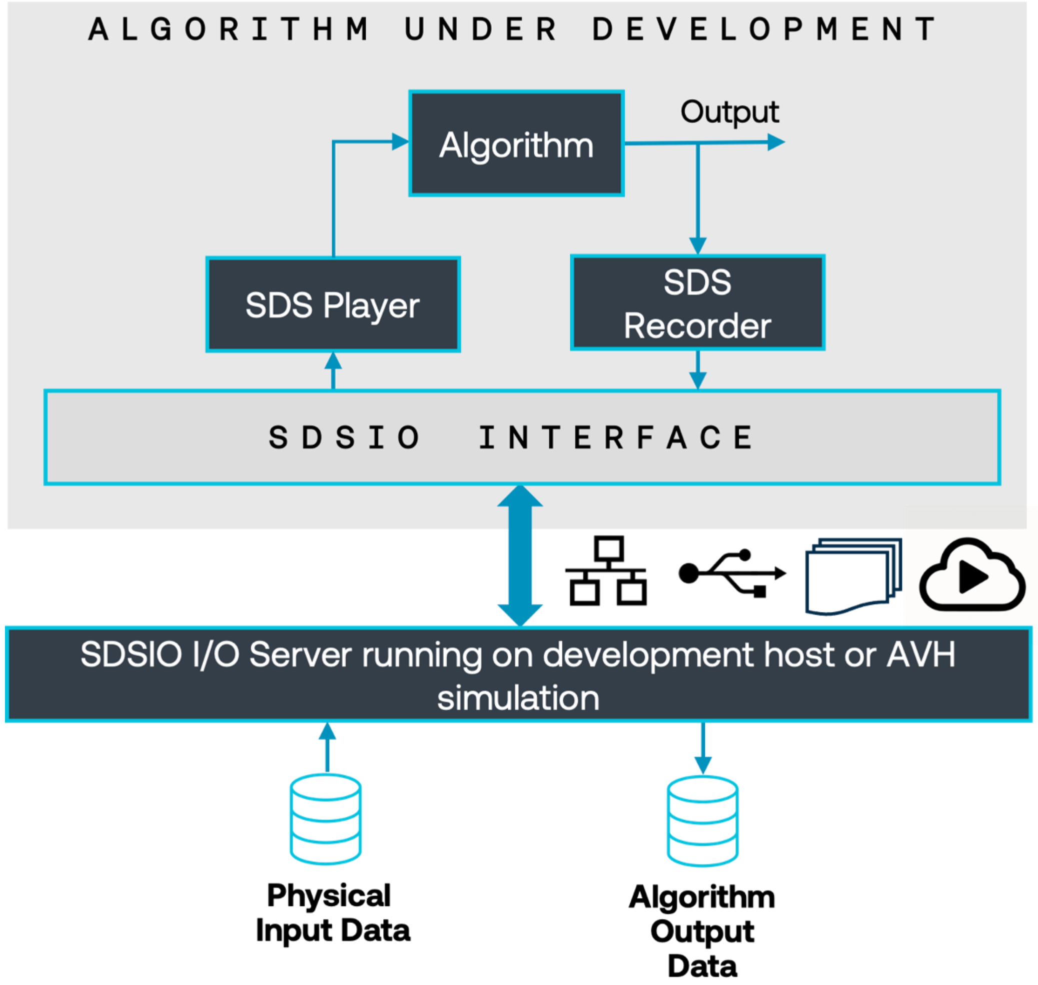
arm

SDS-Framework: Workbench for AI model and DSP development

Capture real-world data streams in hardware

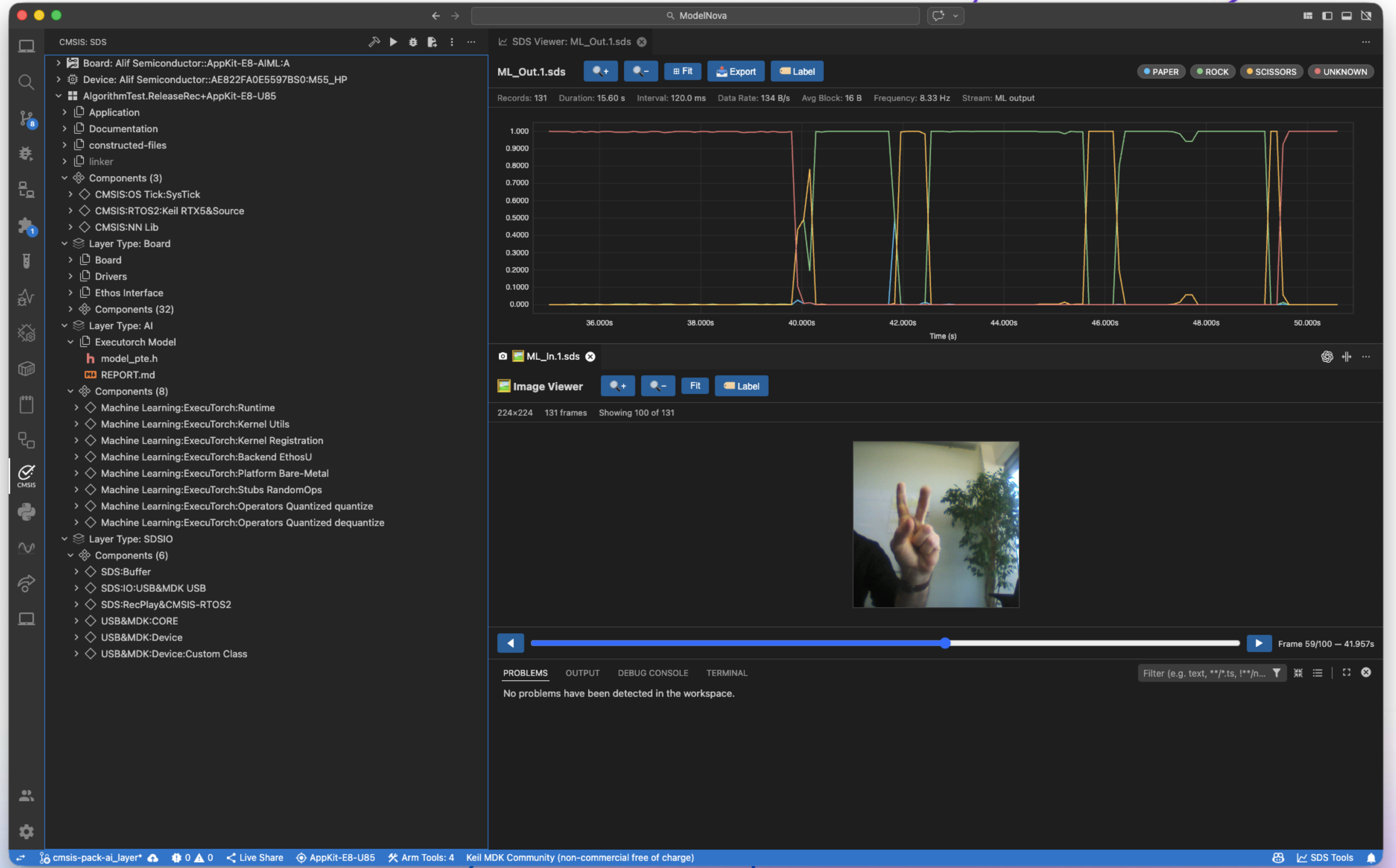


Use same data streams to repeat tests



github.com/ARM-software/SDS-Framework

Demo



- Board: Alif Semiconductor::AppKit-E8-AIML:A
- Device: Alif Semiconductor::AE822FA0E5597BS0:M55_HP
- AlgorithmTest.ReleaseRec+AppKit-E8-U85
  - Application
  - Documentation
  - constructed-files
  - linker
  - Components (3)
    - CMSIS:OS Tick:SysTick
    - CMSIS:RTOS2:Keil RTX5&Source
    - CMSIS:NN Lib
  - Layer Type: Board
    - Board
    - Drivers
    - Ethos Interface
    - Components (32)
  - Layer Type: AI
    - Executorch Model
      - model_pte.h
      - REPORT.md
    - Components (8)
      - Machine Learning:ExecuTorch:Runtime
      - Machine Learning:ExecuTorch:Kernel Utils
      - Machine Learning:ExecuTorch:Kernel Registration
      - Machine Learning:ExecuTorch:Backend EthosU
      - Machine Learning:ExecuTorch:Platform Bare-Metal
      - Machine Learning:ExecuTorch:Stubs RandomOps
      - Machine Learning:ExecuTorch:Operators Quantized quantize
      - Machine Learning:ExecuTorch:Operators Quantized dequantize
  - Layer Type: SDSIO
    - Components (6)
      - SDS:Buffer
      - SDS:IO:USB&MDK USB
      - SDS:RecPlay&CMSIS-RTOS2
      - USB&MDK:CORE
      - USB&MDK:Device
      - USB&MDK:Device:Custom Class

Records: 131 Duration: 15.60 s Interval: 120.0 ms Data Rate: 134 B/s Avg Block: 16 B Frequency: 8.33 Hz Stream: ML output

224×224 131 frames Showing 100 of 131

Frame 59/100 — 41.957s

No problems have been detected in the workspace.

Keil MDK Community (non-commercial free of charge)

ModelNova

Fusion Studio

John Marconi

Fusion Studio – Unified Edge AI Workflow for Arm

One unified workflow from dataset to deployment.

The Challenge

Fragmented tools, duplicated effort, and platform-specific rework slow Edge AI development.

The Solution

A unified workflow that removes fragmentation and accelerates development across Arm platforms.

What Fusion Studio Enables

End-to-End Lifecycle

Dataset → Training → Benchmarking → Deployment.
In one environment.

Arm-First

Deploy across Arm platforms without vendor lock-in or platform-specific rewrites.

Model Zoo Acceleration

Start from 150+ Edge-optimized models – not from scratch

Enterprise-Ready

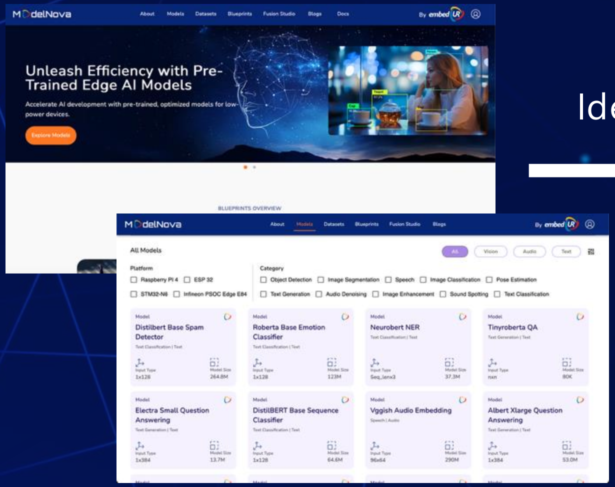
Local or self-hosted deployment, private datasets, secure by design

Develop faster. Optimize smarter.
Deploy confidently across the Arm ecosystem.

ModelNova
AN embedUR COMPANY

Fusion Studio – Your Competitive Edge AI Advantage

Start with ***150+*** *open-source models from ModelNova.ai*



Unleash Efficiency with Pre-Trained Edge AI Models

Accelerate AI development with pre-trained, optimized models for low-power devices.

BLUEPRINTS OVERVIEW

All Models

Platform

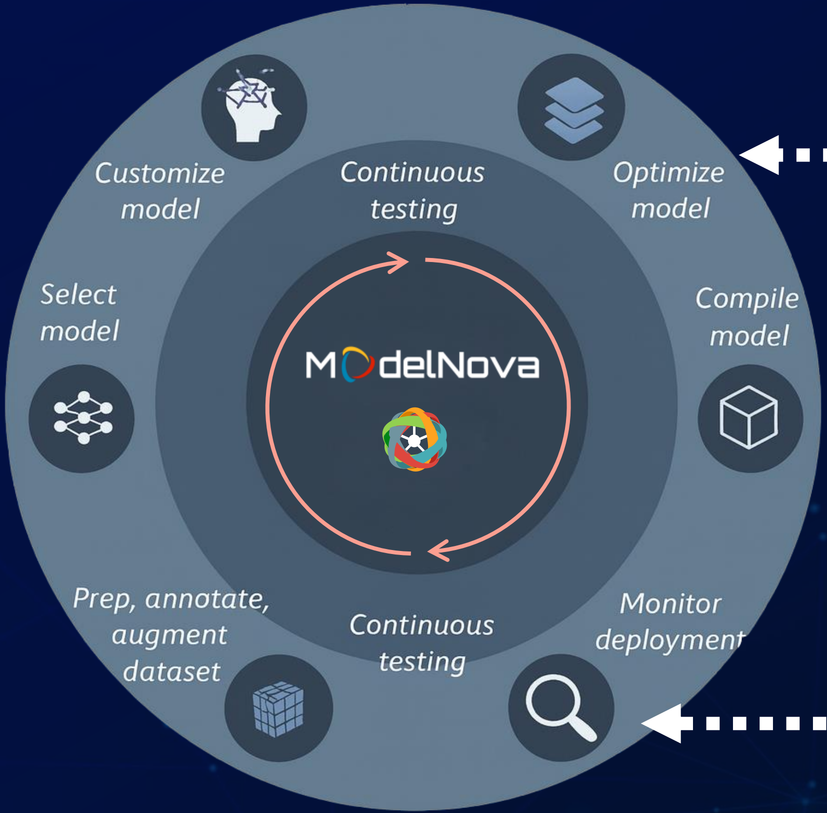
□ Raspberry PI 4 □ ESP 32
□ STM32-N6 □ Infineon PSOC Edge E84

Category

□ Object Detection □ Image Segmentation □ Speech □ Image Classification □ Pose Estimation
□ Text Generation □ Audio Denoising □ Image Enhancement □ Sound Spotting □ Text Classification

Model	Category	Input Type	Model Size
Distilbert Base Spam Detector	Text Classification | Text	1x128	264.8M
Roberta Base Emotion Classifier	Text Classification | Text	1x128	123M
Neurobert NER	Text Classification | Text	Seq_lenx3	37.3M
Tinyroberta QA	Text Generation | Text	nan	80K
Electra Small Question Answering	Text Generation | Text	1x384	13.7M
DistilBERT Base Sequence Classifier	Text Classification | Text	1x128	64.6M
Vggish Audio Embedding	Speech | Audio	96x64	290M
Albert Xlarge Question Answering	Text Generation | Text	1x384	53.0M

Idea



Customize model

Continuous testing

Optimize model

Select model

Compile model

M delNova

Prep, annotate, augment dataset

Continuous testing

Monitor deployment

Real Data from Device Sensors

Deploy

Inference Output Tuned for Accuracy and Performance

20 Years of Know-How

We've solved the hard problems already

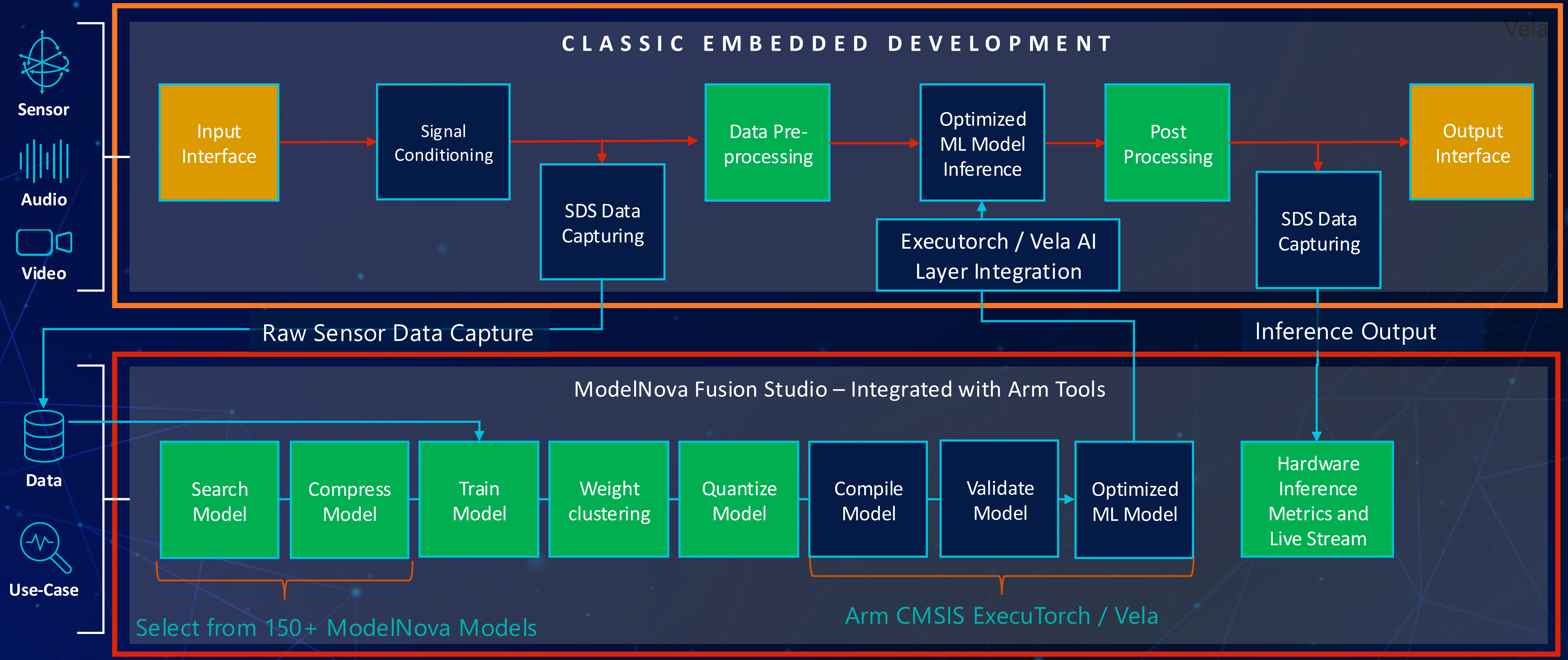
Deploy Anywhere

Cloud, on-premise, or hybrid - your choice

Enterprise Ready

Complete solutions, not just tools

ModelNova Fusion Studio + Arm Development Flow



Sensor

Audio

Video

CLASSIC EMBEDDED DEVELOPMENT

Vela

Input Interface

Signal Conditioning

SDS Data Capturing

Data Pre-processing

Optimized ML Model Inference

Executorch / Vela AI Layer Integration

Post Processing

SDS Data Capturing

Output Interface

Raw Sensor Data Capture

Inference Output

Data

Use-Case

ModelNova Fusion Studio – Integrated with Arm Tools

Search Model

Compress Model

Train Model

Weight clustering

Quantize Model

Compile Model

Validate Model

Optimized ML Model

Hardware Inference Metrics and Live Stream

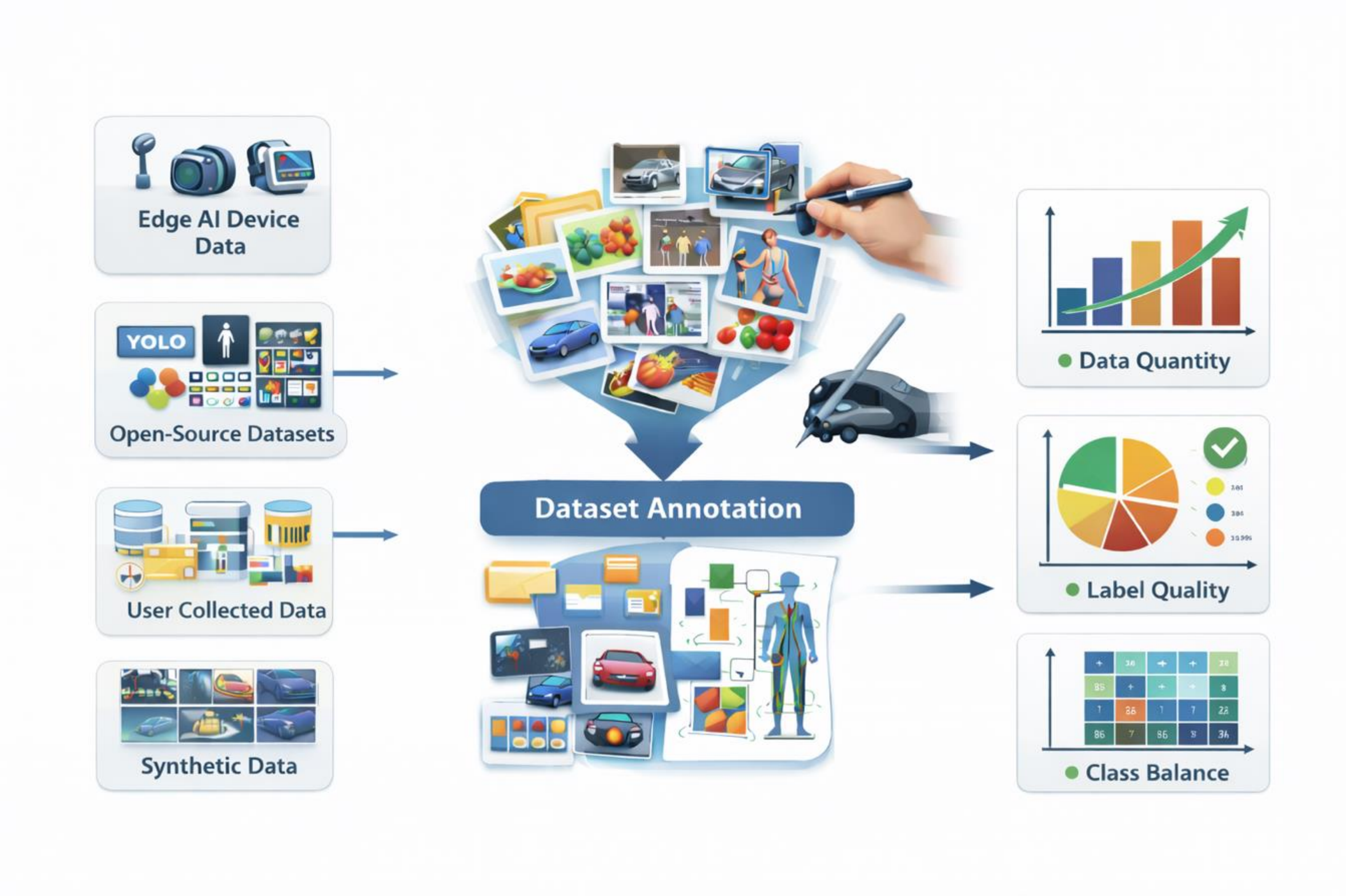
Select from 150+ ModelNova Models

Arm CMSIS ExecuTorch / Vela

ModelNova

AN embedUR COMPANY

Fusion Studio – Dataset collection, annotation, and analysis



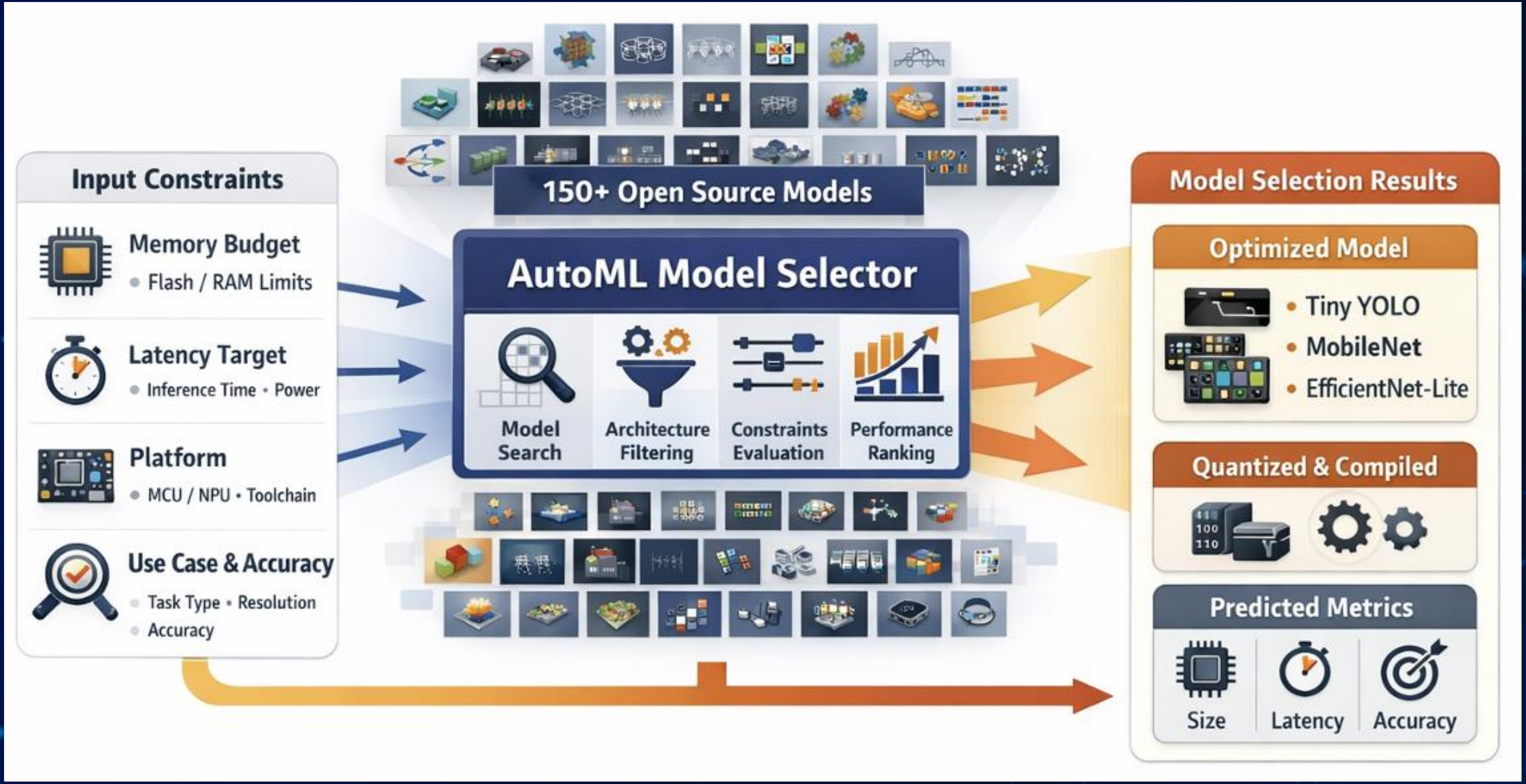
Full Dataset Lifecycle Management

- Collect data from any source
- Manual + AI annotation
- Secure cloud backup
- Pre-training dataset validation
- Augmentations (coming soon)

Fusion Studio – AutoML for Open-source Models and Your Platform

AI AutoML Assistant

Analyzes user-defined requirements to select and optimize models for the target platform.



"I need a model that can detect and count people at VGA 10 FPS on my Cortex M55 with Ethos-U85 running Zephyr with Executorch"

Fusion Studio – Arm Platform integration

ModelNova Fusion Studio

ModelNova Open-source Model
Dataset
Local and Cloud CUDA / MPS / CPU Training
Device Management Service

arm Keil Studio (part of MDK)

CMSIS
Executorch / Vela
SDS
Compilers (AC6, GCC, LLVM)
Debugger

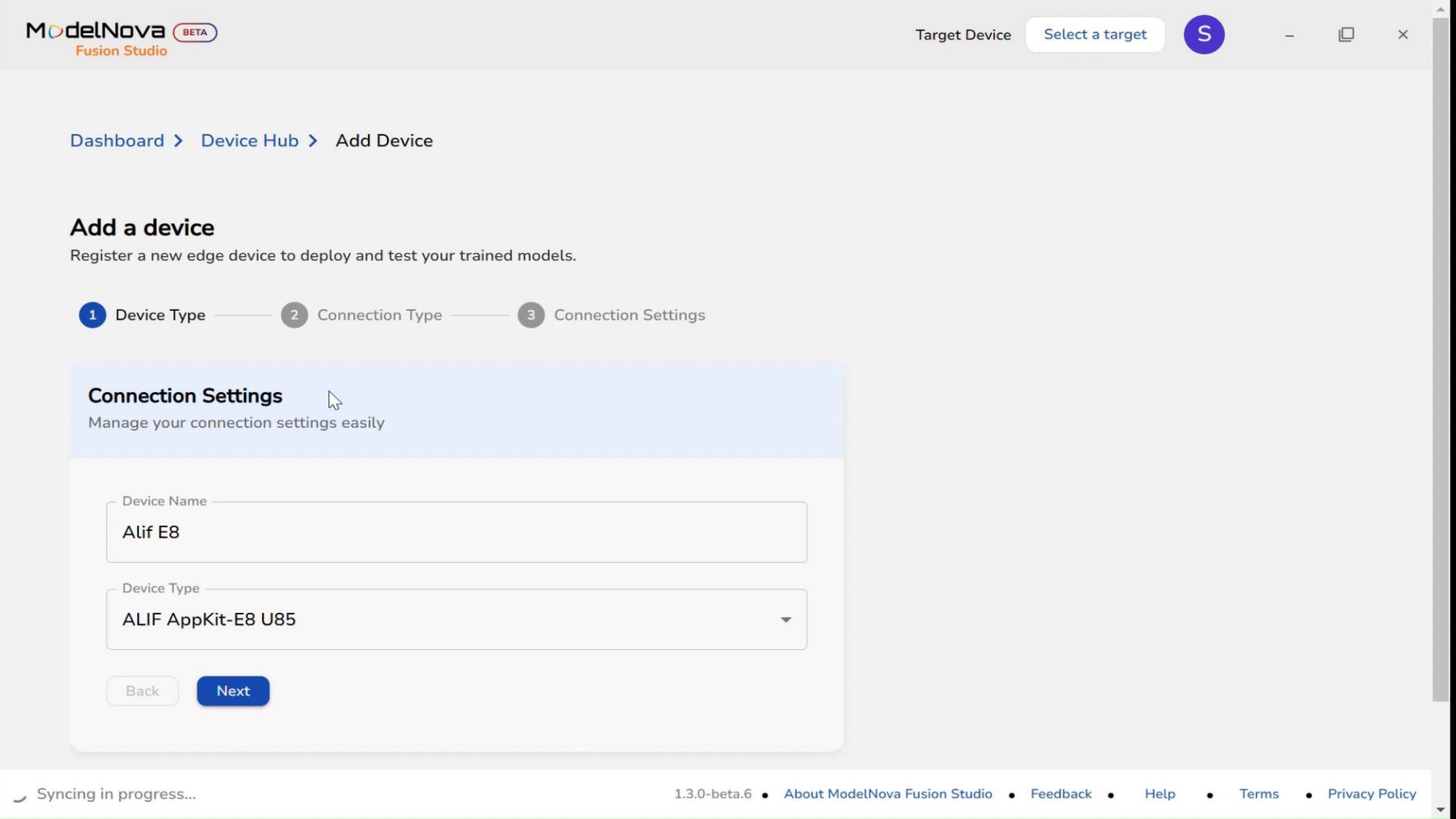
Fusion Studio Platforms

Raspberry Pi 4/5
Arm Ethos-U55 (FVP and SoC)
- Alif E7

Arm Ethos-U85 (FVP and SoC)
- Alif E8

ALIF SEMICONDUCTOR SE Tools

ModelNova AN embedUR COMPANY



Add a device

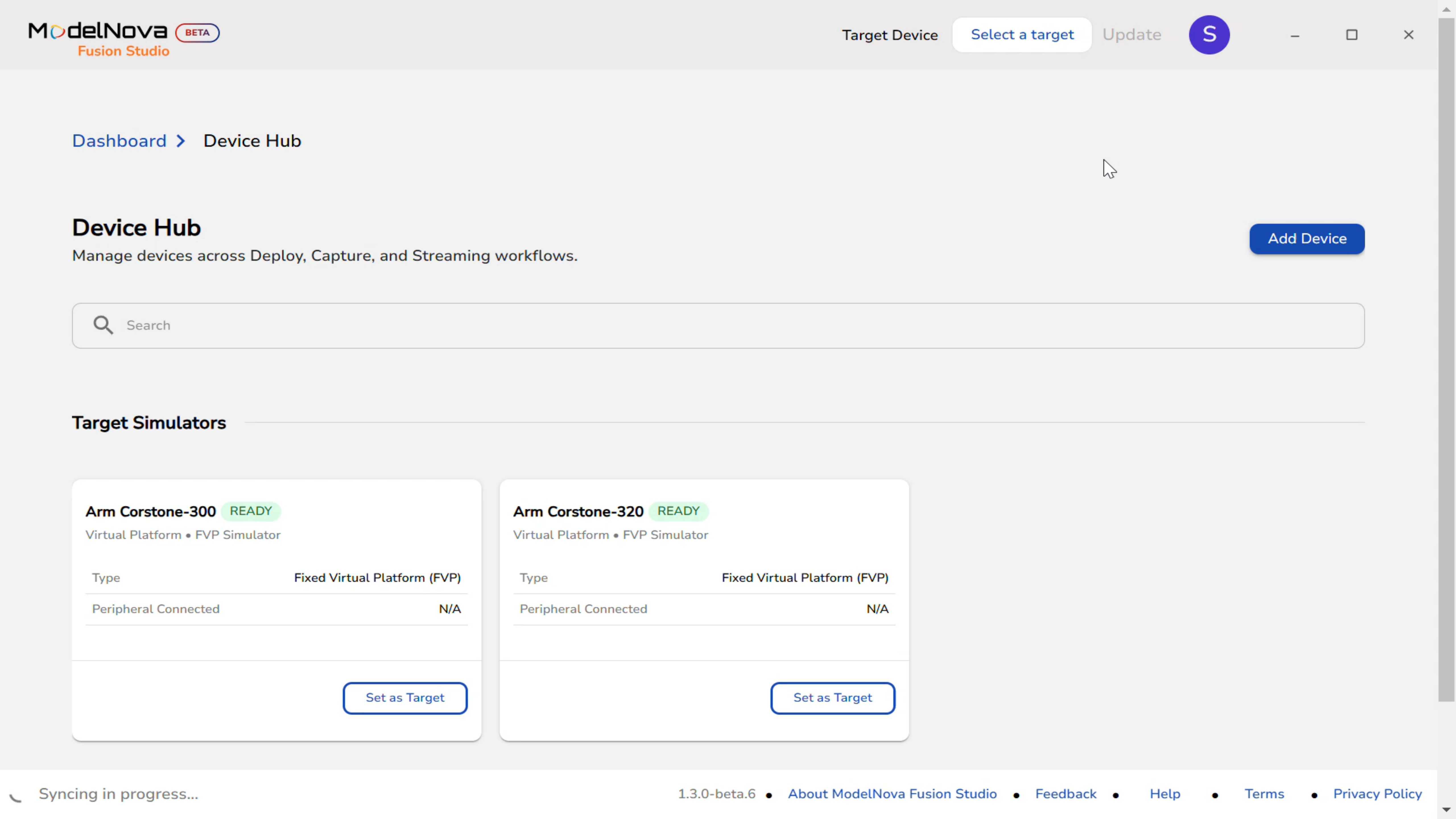
Register a new edge device to deploy and test your trained models.

Connection Settings

Manage your connection settings easily

Device Name: Alif E8

Device Type: ALIF AppKit-E8 U85



Device Hub

Manage devices across Deploy, Capture, and Streaming workflows.

Target Simulators

Arm Corstone-300 READY

Virtual Platform • FVP Simulator

Type	Fixed Virtual Platform (FVP)
Peripheral Connected	N/A

Arm Corstone-320 READY

Virtual Platform • FVP Simulator

Type	Fixed Virtual Platform (FVP)
Peripheral Connected	N/A

Syncing in progress...

1.3.0-beta.6 • About ModelNova Fusion Studio • Feedback • Help • Terms • Privacy Policy

Get started today!

github.com/Arm-Examples/ModelNova

Visit our booths

Arm: Hall 4/booth 504

- See live demos of CMSIS Executorch + Fusion Studio Models

ModelNova: Hall 4/booth 600

- Get details on Fusion Studio and the ModelNova platform

Speaking Session

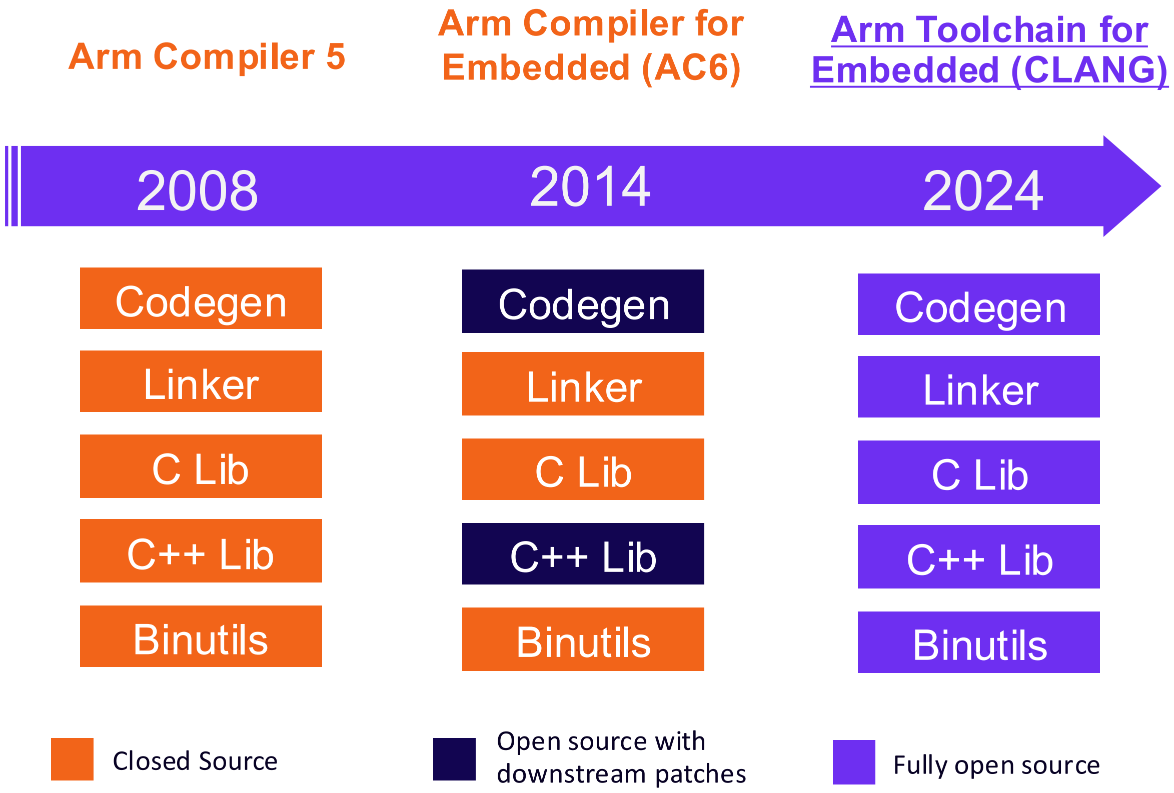
From model to market: Accelerated AI development on Arm

- Wed, March 11 | 9:30 a.m. | Hall 5

Reinhard Keil

Compilers are a long-term commitment

Arm Toolchain for Embedded

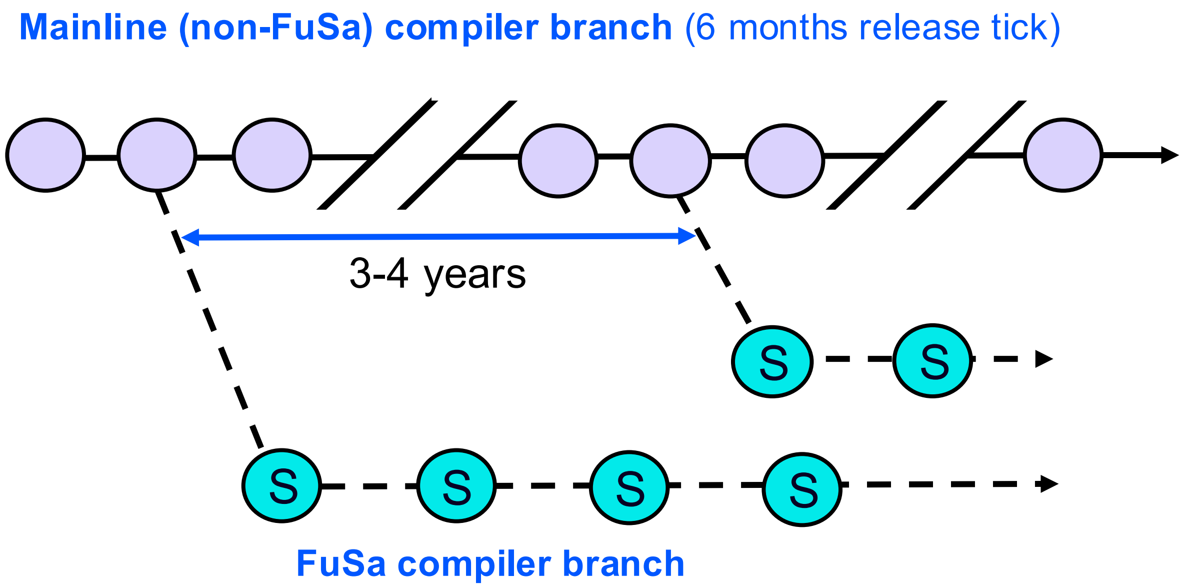


Arm Compiler 5	Arm Compiler for Embedded (AC6)	Arm Toolchain for Embedded (CLANG)
2008	2014	2024
🟧 Codegen	⬛ Codegen	🟪 Codegen
🟧 Linker	🟧 Linker	🟪 Linker
🟧 C Lib	🟧 C Lib	🟪 C Lib
🟧 C++ Lib	⬛ C++ Lib	🟪 C++ Lib
🟧 Binutils	🟧 Binutils	🟪 Binutils

🟧 Closed Source

⬛ Open source with downstream patches

🟪 Fully open source



Mainline (non-FuSa) compiler branch (6 months release tick)

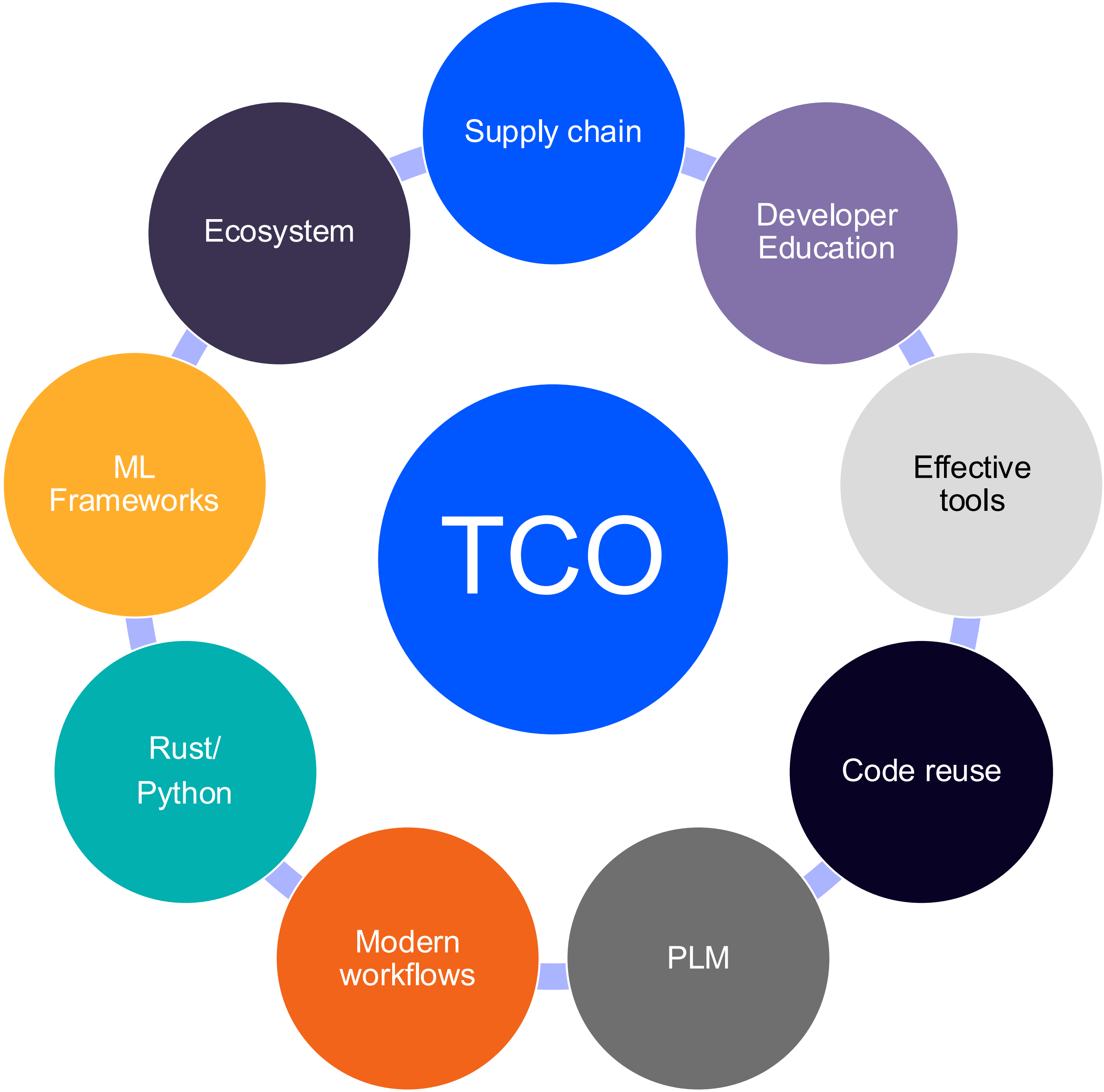
3-4 years

FuSa compiler branch

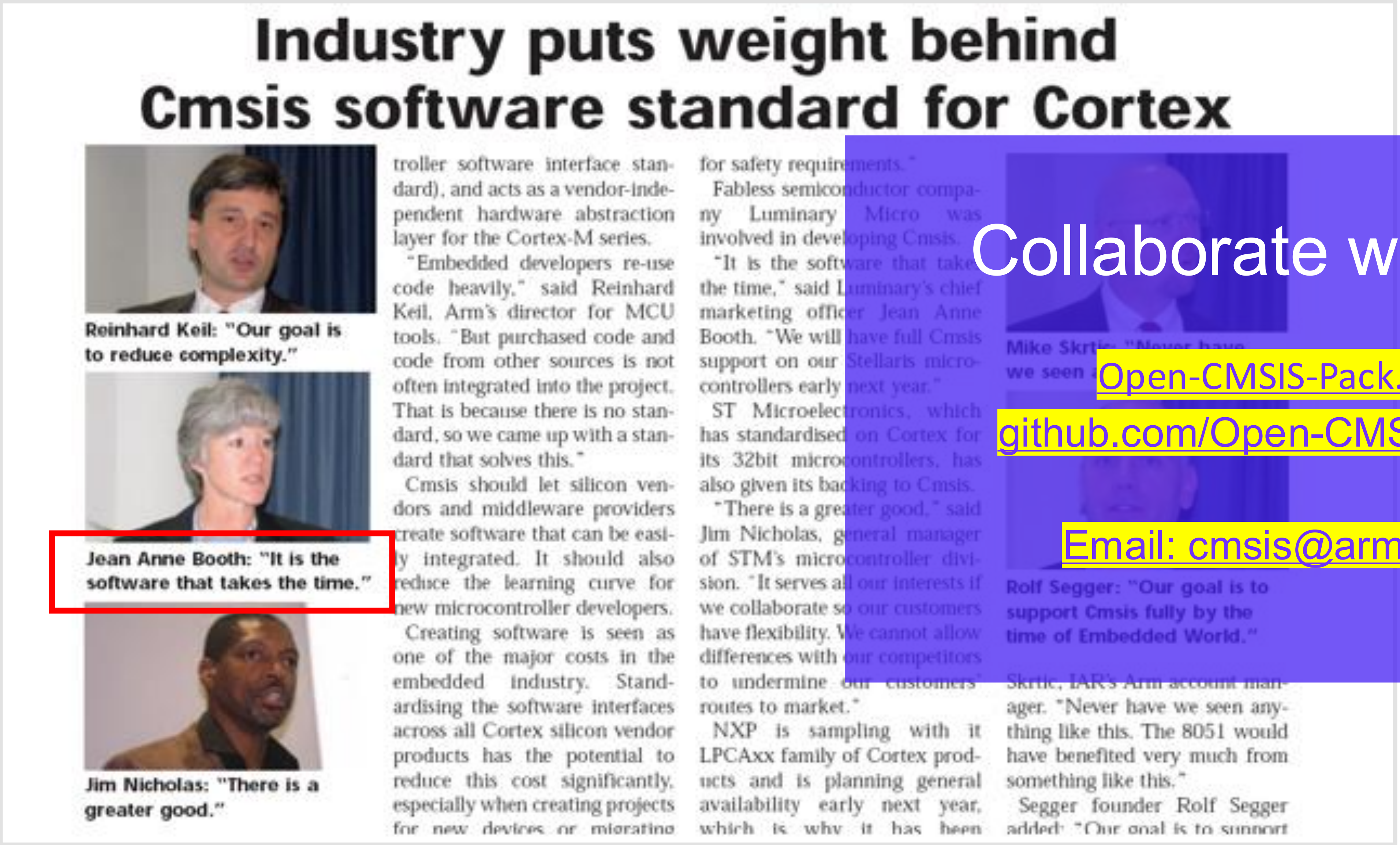
FuSa compiler	Initial Release	Last Release
AC5.04.x	Oct 2013	Sep 2018
AC5.06.x	May 2016	Jan 2021
AC6.6.x	May 2017	Feb 2023
AC6.16.x	Oct 2021	Long-term support To Oct 2025
AC6.22.x	Oct 2024	Long-term support To Oct 2029

Why LLVM Technology?

- It's more than just a compiler (sanitizer, clangd, etc.)
- Base for other languages (RUST)
- MLIR Codegen used for AI model deployment


TCO
Supply chain
Developer Education
Effective tools
Code reuse
PLM
Modern workflows
Rust/ Python
ML Frameworks
Ecosystem

CMSIS Press Conference 2008: **It’s the software that takes the time!**



Industry puts weight behind Cmsis software standard for Cortex

Reinhard Keil: "Our goal is to reduce complexity."

Jean Anne Booth: "It is the software that takes the time."

Jim Nicholas: "There is a greater good."

Mike Skrtic: "Never have we seen

Rolf Segger: "Our goal is to support Cmsis fully by the time of Embedded World."

troller software interface standard), and acts as a vendor-independent hardware abstraction layer for the Cortex-M series.

"Embedded developers re-use code heavily," said Reinhard Keil, Arm's director for MCU tools. "But purchased code and code from other sources is not often integrated into the project. That is because there is no standard, so we came up with a standard that solves this."

Cmsis should let silicon vendors and middleware providers create software that can be easily integrated. It should also reduce the learning curve for new microcontroller developers.

Creating software is seen as one of the major costs in the embedded industry. Standardising the software interfaces across all Cortex silicon vendor products has the potential to reduce this cost significantly, especially when creating projects for new devices or migrating

for safety requirements."

Fabless semiconductor company Luminary Micro was involved in developing Cmsis.

"It is the software that takes the time," said Luminary's chief marketing officer Jean Anne Booth. "We will have full Cmsis support on our Stellaris microcontrollers early next year."

ST Microelectronics, which has standardised on Cortex for its 32bit microcontrollers, has also given its backing to Cmsis.

"There is a greater good," said Jim Nicholas, general manager of STM's microcontroller division. "It serves all our interests if we collaborate so our customers have flexibility. We cannot allow differences with our competitors to undermine our customers' routes to market."

NXP is sampling with it LPCAxx family of Cortex products and is planning general availability early next year, which is why it has been

Skrtic, IAR's Arm account manager. "Never have we seen anything like this. The 8051 would have benefited very much from something like this."

Segger founder Rolf Segger added: "Our goal is to support

Collaborate with us!

Open-CMSIS-Pack.org

github.com/Open-CMSIS-Pack

Email: cmsis@arm.com

arm

Tack
ಧನ್ಯವಾದಗಳು
Merci
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Thank you
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה
ధన్యవాదములు
Köszönöm