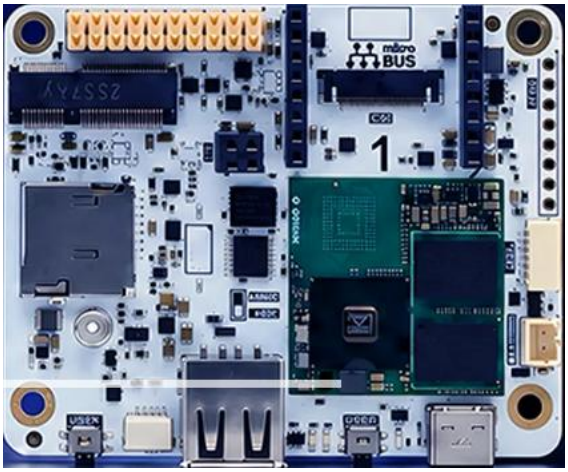
The Coralboard Talks Back. Now It Has to Earn a Syllabus.

Edge-AI Course Readiness, Part 1 · A series testing whether the Synaptics Coralboard is ready for edge-AI courses and labs



The Synaptics Coralboard SL. Image: DigiKey.

A developer kit isn't course-ready because it runs a demo. It's course-ready when an instructor can run it with 25 students and knows what "done" looks like.

My Synaptics Coralboard SL is on the bench, and the first three bring-up steps are done:

- ✅ **Received** from DigiKey
- ✅ **microSD card imaged** with Synaptics' out-of-box Linux image
- ✅ **Board talking to my laptop**
- ⏭️ **Next:** network over USB, the stock AI demos, and the open compiler toolchain

None of that is AI yet. It's the part that comes before AI, and it's where most hardware labs quietly lose a class period. This post covers what the board is, why it's worth testing, and what bring-up has already taught me about putting it in front of students.

Why put a board on the bench

Most students meet AI through a chat window. That hides the parts an engineer needs to see: the sensor, the model, the hardware that runs it, and the trade-offs between speed, power, and accuracy. A course that never puts a board on the bench teaches the interface, not the system.

That gap is what I work on at ML-EDU. I turn datasheets, app notes, GitHub demos, and vendor training into workshops and labs instructors can teach from. A recent example is a 3-hour, six-module instructor-led workshop for the Infineon PSOC™ Edge E84 AI Kit. For

more than 10 years, I've also organized and co-taught a 3-day, for-credit MIT IAP course on various topics including capacitive sensing, IoT, and Edge AI.

What the Coralboard is

The Coralboard SL is DigiKey part 720-KDVSL2619-11, listed at $99.79. Synaptics opened early access on October 14 with a limit of one board per account, and DigiKey began general orders on October 20.

Here's the hardware, from Synaptics' datasheet and Google's documentation:

- **Compute:** a Synaptics Astra SL2619 with two Arm Cortex-A55 cores at up to 2 GHz, an Arm Cortex-M52 for real-time work, and a 1 TOPS Torq NPU. The NPU is the first implementation of Google Research's open-source, RISC-V-based **Coral NPU**.
- **Memory and storage:** 2 GB DDR4, a microSD slot, and NOR flash.
- **I/O:** MIPI camera and display connectors, I^2S audio, a 20-pin GPIO header, mikroBUS, Qwiic, I3C, and an M.2 slot for Wi-Fi and Bluetooth.
- **Power:** 5 V USB-C, plus a 3.7 V LiPo connector with no on-board charging.
- **Software:** Yocto Linux from the Synaptics Astra SDK, with prebuilt images.

Know what it doesn't have before you plan a lab around it: one USB-A host port, no USB-C cable, no Ethernet jack, and no HDMI. (These are addons that can be purchased separately).

Why it's worth testing

An open compiler. Synaptics' Torq compiler and runtime are open source, built on IREE and MLIR, and accept models from PyTorch, ONNX, JAX, and TFLite/LiteRT. Most edge NPUs ship with closed compilers. Here, students can look under the hood. Synaptics has even published how it fits Gemma 3 270M onto the NPU using mixed 4- and 8-bit quantization, which makes a ready-made case study for an ML systems class.

A department-sized price. At $99.79, a 20-seat lab is about $2,000 in boards. That fits a department budget. It doesn't need a capital request.

A platform, possibly. On October 20, Synaptics announced a second board, the Coralboard SR, with a 400 GOPS NPU and the same Torq toolchain. Synaptics says the Coralboard "was designed as a platform, not a one-off board." If that holds, a lab built on the toolchain could outlive any single board. That's a vendor claim for now, and this series will test it.

Bring-up, so far

Step 1: Image the microSD card

You download Synaptics' out-of-box image (`sl2619_coralboard_oobe_scarthgap_wic_6.12_v2.6.1`) and write it to a microSD card.

The first finding came before the board ever powered on, and it's about documentation. Google's getting-started page still describes the Google I/O limited edition, which boots from onboard eMMC and ships with a sensor board. The clearest instructions for the retail

board's SD card were in a DigiKey forum post that uses `bmaptool`, a Linux command-line tool. That's fine for an engineer. A first-year student on a Windows laptop would stall here.

Step 2: Get the board talking to a computer

With the card in, the board boots Yocto Linux. The documented ways in are a USB serial console at 115200 baud or Android's `adb` tool. You log in as `root` with no password, and on first boot the filesystem expands to fill the card.

It worked, and my laptop and the board are now talking. But look at what this step needed that the board didn't supply: a card-imaging tool, a serial terminal or `adb` installed on the host, and a USB-C cable that carries data, not just power. Every one of those is a separate place for a student to get stuck.

The stall log so far

Step	What a student hits	What the instructor guide needs
Find the right guide	The top getting-started page describes a different edition of the board	Link straight to the Coralboard SL instructions
Image the card	The clearest steps use a Linux command-line tool	Pre-image the cards, or give a Windows and Mac path
First contact	A serial terminal or `adb` has to be installed on the host first	Install host tools before lab day, and spec a data-capable USB-C cable

None of these is a dealbreaker. All of them are fixable in an instructor guide. That's the point: a board becomes teachable when someone does this work once, so 25 students don't have to do it 25 times.

Coming next week

Part 2 picks up where the board starts doing AI:

- Sharing the laptop's network over USB so the board can pull the example code
- Running the stock examples: object detection, image classification, Gemma translation, function calling, and Jellectronica
- Timing power-on to first working demo, with every stall logged
- A first go/no-go read against four questions:
    1. **Can a student finish it?** Not just run a demo, but change it and get it working again.
    2. **Is it a sequence?** Do the examples build from bring-up to deployment?
    3. **Is it lab-anchored?** Can students measure something real, like latency or power?
    4. **Can a professor teach it?** Or would an instructor have to rebuild the lab from datasheets?

Where it could fit (untested)

- **Intro courses:** Vision, audio, and small language models on one board could anchor a "one board, many modalities" unit.
- **Science fair projects:** I sit on the board of the Santa Clara Valley Science and Engineering Fair Association, which handles roughly 800 applications a year. Qwiic and mikroBUS connectors, a camera interface, and battery input could make low-power field projects more approachable.
- **College labs:** The open compiler, the Yocto build, and the split between NPU, Arm cores, and microcontroller add up to a real systems-engineering lab.

For hardware companies

The same test applies to your board. Booths create visibility. Syllabi create repeated use. Count the instructors who teach the module on their own, the courses that list your product in the syllabus, and the students who finish the lab. The stall log above is the cheapest adoption investment you can make: every row you fix in your own documentation is a class period a professor doesn't lose.

Before ML-EDU, I grew the Cypress University Alliance from zero to 1,900 universities across the USA, China, India, Japan, and EMEA. The lesson: technical content becomes valuable when it becomes teachable. If your university strategy stops at booths, start with an **ML-EDU University Adoption Sprint**. It ends with a go/no-go before you invest in scale. Message me on LinkedIn or email me at patrick@ml-edu.org to find out more.

Faculty and program leads: if you teach embedded systems, computer vision, or edge AI, what breaks first when a new board arrives: setup, software, lab time, or assessment? Reply with one constraint, and I'll test it in Part 2.

— Patrick Kane, ML-EDU

Sources

- Synaptics, Coralboard SL datasheet. https://developer.synaptics.com/docs/developer-kits/coralboard/sl/datasheet
- Synaptics, "Synaptics and Google Research Expand the Coralboard™ Ecosystem" (October 20, 2026). https://www.edge-ai-vision.com/2026/10/synaptics-and-google-research-expand-the-coralboardtm-ecosystem/
- Google for Developers, Getting started with Coralboard. https://developers.google.com/coral/products/SL2610-get-started
- DigiKey TechForum, "Getting started with Synaptics Coralboard SL." https://forum.digikey.com/t/getting-started-with-synaptics-coralboard-sl/72305
- Google Research, "Coral NPU: A full-stack platform for Edge AI" (October 15, 2025). https://research.google/blog/coral-npu-a-full-stack-platform-for-edge-ai/
- Synaptics Torq compiler documentation. https://synaptics-torq.github.io/torq-compiler/v/latest

- Synaptics, “The Edge LLM Offload Story: How Synaptics Torq Enables High-Efficiency Gemma Inference” (August 2026). https://www.edge-ai-vision.com/?p=58869
- Synaptics, sl2610-examples repository. https://github.com/synaptics-astra-demos/sl2610-examples
- DigiKey, Synaptics Coralboard SL2619 listing (part 720-KDVSL2619-11). https://www.digikey.com/en/products/detail/synaptics-coralboard/720-KDVSL2619-11/29784108