An Acorn on a Compute Blade at ps1: the cards, the cables, checking them, labels

Printed from the pages published at https://docs.fpgas.online/en/latest/, built from commit 98b174f182a5f7c015526deacbd056cbddb5143b of fpgas-online/fpgas.online-docs, fetched 2026-10-08. The published pages are the current ones; this is a copy.

1. Acorns at ps1, sheet 5 (boards/acorn/installations/ps1)
2. Taking pi20 at ps1’s old serial wiring off, sheet 11 (boards/acorn/installations/ps1-pi20-old-serial)
3. What fpgas.online ran on the ps1 blades, 7 October 2026, sheet 14 (boards/acorn/installations/ps1-ran-2026-10-07)
4. Compute Blade cables: overview, sheet 16 (boards/acorn/building/compute-blade/index: what-you-will-have, the-order-of-work, where-the-facts-come-from)
5. Compute Blade cables: parts and tools, sheet 20 (boards/acorn/building/compute-blade/bom)
6. Compute Blade cables: JTAG connector 1, prepare the wires, sheet 22 (boards/acorn/building/compute-blade/jtag-connector-1)
7. Compute Blade cables: JTAG connector 2, fill and check the housing, sheet 28 (boards/acorn/building/compute-blade/jtag-connector-2)
8. Compute Blade cables: UART connector 1, prepare the wires, sheet 33 (boards/acorn/building/compute-blade/uart-connector-1)
9. Compute Blade cables: UART connector 2, fill and check the housing, sheet 38 (boards/acorn/building/compute-blade/uart-connector-2)
10. Compute Blade cables: bench check, sheet 43 (boards/acorn/building/compute-blade/bench-check)
11. Compute Blade cables: fitting, sheet 47 (boards/acorn/building/compute-blade/fitting)
12. Compute Blade cables: verifying 1, run the check and read the result, sheet 49 (boards/acorn/building/compute-blade/verifying-1)
13. Compute Blade cables: verifying 2, when a test fails, sheet 54 (boards/acorn/building/compute-blade/verifying-2)
14. Compute Blade cables: verifying 2b, when the failure is not a wire, sheet 58 (boards/acorn/building/compute-blade/verifying-2b)
15. Compute Blade cables: verifying 3, what has been run on a Compute Blade, sheet 61 (boards/acorn/building/compute-blade/verifying-3)

16. Bootloader EEPROM on a Compute Module in a Compute Blade: does it need anything?, sheet 65 (setup/bootloader-eeprom-compute-module)
17. Bootloader EEPROM: upgrade and lock, sheet 67 (setup/bootloader-eeprom: what-the-compute-blade-s-maker-documents)
18. Labels for the Compute Blades at ps1, sheet 69

FOR CARL, AT PUMPING STATION: ONE. COPY 24, 8 OCT 2026

- This booklet is for making and fitting the two cables between an Acorn and its Compute Blade at ps1, and checking them. Every chapter but the last (the labels) is one page of https://docs.fpgas.online, or the named sections of one (the Overview, and the maker's documents in the bootloader chapter), printed as it stood on the date in its head line; the site is the current copy. The pages for a Raspberry Pi 5 are not in this copy. Where a chapter names another page, look for it in the list of chapters on the first two sheets: a page that is not in that list is not in this booklet.

THE ORDER OF WORK

1. Acorns at ps1: which card is on which blade, what state each is in, and what each still needs. Start here. Then Taking pi20 at ps1's old serial wiring off, and What fpgas.online ran on the ps1 blades, 7 October 2026 (the record of the tests).
2. Overview, then Parts and tools: what you will have at the end; tick off every line before you start.
3. JTAG connector 1 and 2: the cut, the reach check, then the P1 cable, from the bought cable to a checked housing.
4. UART connector 1 and 2: the P2 cable, with its resistor.
5. Bench check, then Fitting: a meter check with the power off, then the cables and the card go in.
6. Verifying 1, 2, 2b and 3: run the check on the blade, read the result, go from a failing line to the wire, and what to expect on a Compute Blade today.
7. Bootloader EEPROM on a Compute Module: whether it needs anything (for the two CM5 blades read: no; on the CM4 blades not known).
8. Labels: the last sheets, with a sheet before them saying which label is which blade's.

WHAT CANNOT BE DONE FROM THIS BOOKLET TODAY

- The cable steps are written from the design and have not been run by us on this hardware. Wire length and gauge are not measured: the guide uses each half of the bought cable at the length it has, and whether that reaches in a blade is not known. Whether a half reaches is checked in step 2 of JTAG connector 1, right after the cut; how much slack is needed is not measured.
- Which blade on the bench is pi14, pi16, pi18 or pi20 at ps1: we hold no record of where each sits. "Acorns at ps1" gives what to look for on each (not checked by eye) and a way that does not depend on looking (not yet tried).
- Anything installed on a blade is gone after it reboots (its root comes from the gateway, with an overlay in memory): install the check again after each boot.
- Converting a card to the fpgas.online design on a Compute Blade has no written steps yet. One attempt, test 6 on pi20 at ps1 on 7 Oct 2026, loaded our design into the FPGA's memory (its SRAM, not its flash) over JTAG and was stopped at step 1b, before any flash write (below). Until a card is converted, the check on a blade ends in "fail" even with perfect cables ("Verifying 1" says which lines to expect); the bench check with a meter is what the cables rest on.
- JTAG on a CM5 blade under kernel 6.18 (what the blades run now) needs the header's serial port turned off at boot, in the blade's boot files on the ps1 gateway. We ran it on pi20 at ps1 (a CM5) on 7 Oct 2026: with the port off, the check's jtag test read the FPGA's IDCODE and device DNA, and test 6 loaded our design into SRAM. A flash write over JTAG has not been run. On the CM4 blades: on 20 Sep 2026 JTAG on pi14 at ps1 got no response (TCK floating: its P1 cable unmated or its TCK wire open); on 7 Oct 2026 only logins, pin reads, a pull test (each pull set and put back) and vcgencmd (which hung). Whether JTAG there needs the port off is not known. The boot files serve twelve hosts at ps1, not only the blades: "Verifying 3" gives the two changes as run and two ways to make them (the shared directory, or one blade's own copy, as pi20 was tested), each with its undo. The change is yours to decide and to time; it is not needed for making or fitting the cables.
- After test 6 was stopped at step 1b (the PCIe re-probe after the load, which failed, and the reboot after it) at 15:52 on 7 Oct 2026 (Adelaide time; 05:22 UTC), pi20 at ps1 restarted every 2 to 3 minutes (one boot, from about 17:06, stayed up until we rebooted it at 17:20:12) until about 18:09, then stayed up; no power cycle was run by us; the cause is not known. Since 17:20 it boots your shared boot files again (the gateway's link was put back at 17:19:48; at 18:10 its serial port was on), with the card on the vendor image. The Acorn's flash was never written. config.txt and cmdline.txt in your shared boot directory were never changed (they read the same before and after): for the test, pi20's link pointed at a copy, and at 17:19:48 the link was put back and the copy removed; we rebooted pi20 into your files at 17:20:12.
- On the two Compute Module 4 blades (pi14 and pi18 at ps1): do not run vcgencmd (it hung there for good on 7 Oct 2026), so no label read and no bootloader read there; and the check has not been run there yet: run it on pi16 and pi20 at ps1 only.
- Where a page says "not yet run by us on this hardware" or "not verified by us", go carefully and tell Tim what you see; the same for "tell whoever gave you this guide" on any page.

LOGGING IN TO A BLADE

- ssh -p 11622 pi@ps1.fpgas.online is pi16 at ps1; the port is 1NN22 for piNN at ps1 (11422, 11622, 11822, 12022). The password is in the banner ssh prints before it asks for one; sudo asks for the same one.

Acorns at ps1

You look after the Acorns at ps1 (Pumping Station: One, Chicago) and want to know which card is where, what state it is in, and what it still needs. The hosts as hosts (addresses, switch ports, power) are on a site page that is not in this booklet; none of that is needed at the bench, because the visitor ports you need are listed below. What was read on each blade, with its date, is in the tables below (the longer record behind them is not in this booklet and is not needed at the bench); what we ran on the blades on 7 October 2026, and what followed, is on What fpgas.online ran on the ps1 blades, 7 October 2026.

The cards

Four Compute Blades; an Acorn was seen on three of them (on pi18 at ps1 none was seen: below). A card is named by its label once it has one: an Acorn's label carries its device DNA and its flash ID, and the flash ID is read through the fpgas.online design, so a card gets its label when it is converted. Until then a row says where the card was last read.

Which physical blade carries which name: we hold no record of where each one is. Nothing we have says which slot or position pi14 at ps1, pi16 at ps1, pi18 at ps1 or pi20 at ps1 is in. What ties a name to a blade is its Ethernet MAC address and its Compute Module's serial number, read on each blade on 7 October 2026 (they match the ps1 gateway's DHCP reservations):

Name	Compute Module	Serial number	Ethernet MAC
pi14 at ps1	CM4 Rev 1.1 4 GB	`10000000d00eb762`	`2c:cf:67:37:d4:bd`
pi16 at ps1	CM5 Lite 8 GB	`9fb8cfc7cb291e63`	`2c:cf:67:fb:91:e5`
pi18 at ps1	CM4 Rev 1.1 4 GB	`100000004e45f174`	`2c:cf:67:37:d5:08`
pi20 at ps1	CM5 Lite 8 GB	`d85a19afde59093d`	`2c:cf:67:fd:1e:be`

On a blade that is running, `ip link` prints its MAC (the `link/ether` line of `eth0`). How to read the MAC or the serial number off a blade in the hand, without running it (a sticker on the module or on the blade), is **not verified by us**.

Telling the blades apart on the bench: the plain answer. With what you hold, you cannot tell which blade is pi14 at ps1, pi16 at ps1, pi18 at ps1 or pi20 at ps1 by looking at it, and we hold no record that says. **Ask Tim which blade is which.** What we do hold:

- **Labels:** the only labels that name a blade are the two Compute Module labels of pi16 at ps1 and pi20 at ps1 (label sheets, last chapter of this booklet). They carry the MAC address, not a host name: the one with `2c:cf:67:fb:91:e5` is pi16 at ps1's, the one with `2c:cf:67:fd:1e:be` is pi20 at ps1's. Whether either is stuck on its blade is not recorded by us. If you find one on a module, its MAC against the table above names the blade. pi14 at ps1 and pi18 at ps1 have no label.

- **Looking:** the list below is all the records say each blade has (none of it checked by eye by us), and it splits the blades into two pairs only (pi14 at ps1 and pi18 at ps1 carry a Compute Module 4, pi16 at ps1 and pi20 at ps1 a Compute Module 5 Lite); nothing we hold separates the two blades within each pair.
- **Unplugging one blade:** the way after the list, not yet tried by us, and it needs Tim first.

What the records say each one has, to look for (none of these has been checked by eye by us):

- pi18 at ps1: no card seen in its M.2 slot (no PCIe device on 7 October 2026).
- pi14 at ps1 and pi18 at ps1 carry a Compute Module 4; pi16 at ps1 and pi20 at ps1 a Compute Module 5 Lite. Which marking on the module tells the two apart is not recorded by us.
- Between the two Compute Module 5 blades: which header pins pi20 at ps1's old serial wires sit on is not recorded by us (the steps for its old serial wiring), and neither is how pi16 at ps1's cables are wired, so the wires do not tell the two apart: use the way below.

A way that does not depend on looking, **not yet tried by us**. **Before you unplug a blade, ask Tim**: the blades are open to visitors, and unplugging one ends whatever is running on it, a visitor's session included. Then, with every other blade running, unplug one blade's PoE cable and wait a minute; the name whose visitor port then stops answering is that blade. The ports are 11422 for pi14 at ps1, 11622 for pi16 at ps1, 11822 for pi18 at ps1 and 12022 for pi20 at ps1 (`ssh -p 11622 pi@ps1.fpgas.online` and so on). Plug it back in and wait until its port answers again: pi20 at ps1 answered about three and a half minutes after a reboot (7 October 2026, 14:22, before our test of 15:52). Anything installed on it is gone (below).

All four answered on their visitor ports on 7 October 2026 (pi14 at ps1 and pi18 at ps1 had not on 5 October); pi20 at ps1 restarted every 2 to 3 minutes from 15:52 to about 18:09 that day (one boot, from about 17:06, stayed up until we rebooted it at 17:20:12), then stayed up (What fpgas.online ran on the ps1 blades, 7 October 2026). **If a blade's ssh port does not answer**, nothing can be run on it from this guide; first find out whether it is powered and its network link is up (its lights; the gateway's view of it): **not yet checked by us**. Its cables can still be made, bench-checked and fitted.

An install on a blade lasts until its next boot. The blades run their root file system from the gateway with an overlay in memory (`overlayroot=tmpfs` on the kernel command line, read 7 October 2026), so packages installed on a blade are gone after it reboots: an install of 5 October on pi16 at ps1 was gone on 7 October. Putting them into the shared root on the gateway is the gateway's owner's to do, and is not written here.

Converting a card means loading the fpgas.online design into it over JTAG and writing that design to the card's flash, once; after that the card runs it from every power-on. The steps are written for a Raspberry Pi 5 (on a page that is not in this booklet; converting is not part of this booklet's work) and have **not yet been run by us on a Compute Blade** to their end: begun once on pi20 at ps1 on 7 October 2026 and stopped before the flash write (What fpgas.online ran on the ps1 blades, 7 October 2026).

Card, by its label	On (last read)	Compute Module	What the card runs	PCIe	JTAG (P1)	P2 cable
Acorn CLE-101, no label yet (device DNA not read)	pi14 at ps1 (`1e24:0101` seen 2026-10-07)	CM4 Rev 1.1 4 GB	SQRL's factory image, `1e24:0101`	`0000:01`	2026-09-20: no response, TCK floating	not known

Card, by its label	On (last read)	Compute Module	What the card runs	PCIe	JTAG (P1)	P2 cable
Acorn CLE-101, no label yet (device DNA not read)	pi16 at ps1 (2026-10-07)	CM5 Lite Rev 1.0 8 GB	SQRL's factory image, `1e24:0101`	`0001:01`	2026-09-20: no response, TCK floating. 2026-10-05: cannot run (the serial driver holds GPIO14)	not known
Acorn CLE-101, device DNA `0x0028e5c45e304854`	pi20 at ps1 (2026-10-07)	CM5 Lite Rev 1.0 8 GB	a vendor XDMA sample image, `10ee:7011` (the flash never written: what we ran)	`0001:01`	answered (2026-09-20, kernel 6.12). Under kernel 6.18 (2026-10-07): fails with the header's serial port on (it holds GPIO14); passed with the port off at boot (what we ran)	K2 to GPIO15, J2 to GPIO14, no resistor (which header pins is not recorded: the steps); J5 and H5 not wired
none seen: no PCIe device (2026-10-07; the slot is empty, or a card has no link: `lspci` cannot tell)	pi18 at ps1 (2026-10-07)	CM4 Rev 1.1 4 GB				

What "P1 unmated" rests on: it is read off the TCK wire. The Acorn pulls TCK up; on pi20 at ps1 the Pi's pull-down cannot move it, while on pi14 at ps1 and pi16 at ps1 it floats, exactly as on pi18 at ps1, which has no card. (Read on 2026-09-20; on 2026-10-07 the TCK read was repeated on pi16 at ps1 only: a pull test was also run on pi14 at ps1 and pi18 at ps1 that day, with no result recorded by us.) Reseating P1 is the first thing to try.

No blade is wired to the guide yet. Only pi20 at ps1's wiring has been read: its P2 serial pair goes straight to GPIO14 (J2) and GPIO15 (K2), no resistor, and J5 and H5 are not wired (pin-ID read, 31 August 2026). How pi14 at ps1's and pi16 at ps1's P2 cables are wired is not known, because nothing can be loaded to read them.

What each blade still needs

The check on the two Compute Module 4 blades (pi14 at ps1 and pi18 at ps1) has not been run by us: it was run as written only on pi16 at ps1 and pi20 at ps1, both Compute Module 5 (7 October 2026). `vcgencmd` hung for good on the two CM4 blades that day. Until the check has been tried on a CM4 blade, make, bench-check and fit the cables of pi14 at ps1 and pi18 at ps1 as of the others, and run the check on the two CM5 blades only.

What to build. P2: four new cables by the guide, one for each blade; pi20 at ps1's present serial wiring is taken off (the steps). P1: three new cables by the guide, for pi14 at ps1, pi16 at ps1 and pi18 at ps1. pi18 at ps1 has none; on pi14 at ps1 and pi16 at ps1 TCK read as if no P1 cable were mated, and you cannot re-read it there now (the check is not run on pi14 at ps1, a Compute Module 4, and on pi16 at ps1 JTAG cannot run with the header's serial port on), so their old cables come off. pi20 at ps1's P1 answered on 2026-09-20: it is kept if it passes the meter check in the steps for its old serial wiring, else it is built new too. Every cable that goes in, new or kept, gets the bench check with the card out before it is fitted. So: four P2 cables and three P1 cables (four if pi20 at ps1's fails). Each bought Molex cable gives one P1 half and one P2 half, so that takes four Molex cables, with one P1 half spare (for pi20 at ps1 if needed). The terminals are 3 per P2 cable and 5 per P1 cable: 27 in all (32 with a new P1 for pi20 at ps1), and a few spare. pi18 at ps1 also needs a card; which card goes there is not recorded by us.

Before anything is fitted, refitted or reseated: ask Tim, then power the blade off (unplug its PoE cable, and a USB-C cable if one is plugged in). Over JTAG, run only the check's own `jtag` test (`sudo fpgas-acorn-verify --no-publish --test jtag`), and only as verifying 3 says: in a boot with the header's serial port off, after its steps that show GPIO14 is free and nothing on the card drives it. Do not run `openFPGALoader` by hand on a blade. **Loading a bitstream on a blade is not part of this guide**: on the one blade it was tried on, the card's PCIe endpoint did not come back after the load and the blade restarted for over two hours (what fpgas.online ran on the ps1 blades).

To reach the wiring of the Compute Blade building guide, from what the table above records. None of us has wired a blade this way or converted a card on one yet (one conversion was begun on pi20 at ps1 and stopped before the flash write): **not yet run by us on this hardware**.

Blade	Card	P1 (JTAG) cable	P2 (serial) cable	Host
pi14 at ps1	fitted, factory image: to be converted	not mated, or its TCK wire open (TCK follows the host's pull, as on the empty pi18 at ps1: 2026-09-20 and again 2026-10-07): take it off and build a new one by the guide	not known: build to the UART header, 470 Ω in the J2 wire, J5 and H5 cut back	as pi16 at ps1 (read 2026-10-07: the same serial-port settings); a Compute Module 4: (b) not yet run, and `gpioinfo` names no user for line 14 there (2026-10-07)
pi16 at ps1	fitted, factory image: to be converted	not mated, or its TCK wire open (TCK follows the host's pull, as on the empty pi18 at ps1: 2026-10-07): take it off and build a new one by the guide	not known: build to the UART header, 470 Ω in the J2 wire, J5 and H5 cut back	(a) for the serial-pair tests: the kernel console and the getty off `/dev/ttyAMA0`; (b) only for JTAG: the serial port off at boot (run on pi20 at ps1, 7 October 2026: what we ran)
pi18 at ps1	none seen: look; if the slot is empty, leave it empty and tell Tim (which card goes there is not recorded)	fit on the Extension Port	fit on the UART header, 470 Ω in the J2 wire, J5 and H5 cut back	as pi16 at ps1 (read 2026-10-07: the same serial-port settings); a Compute Module 4: (b) not yet run, and `gpioinfo` names no user for line 14 there (2026-10-07)

Blade	Card	P1 (JTAG) cable	P2 (serial) cable	Host
pi20 at ps1	fitted, vendor sample image in flash: to be converted	answered on 2026-09-20: kept if it passes the meter check in the steps for its old serial wiring. On 2026-10-07 JTAG could not run with the header's serial port on (it holds GPIO14), and read the IDCODE and DNA with the port off at boot, the Host column's (b) (What fpgas.online ran on the ps1 blades, 7 October 2026)	build a new one by the guide for the UART header (470 Ω in the J2 wire) and take the old serial wiring off (its pair is wired to GPIO14 and GPIO15; which header pins is not recorded: the steps)	as pi16 at ps1 (read 2026-10-05: the same kernel and serial-port settings): (a) for the serial-pair tests: the kernel console and the getty off `/dev/ttyAMA0`; (b) only for JTAG: the serial port off at boot (run on pi20 at ps1, 7 October 2026: what we ran)

The Host column is Carl's to do, on the gateway, not on a blade. The two files it means, `config.txt` and `cmdline.txt`, are in one directory on the ps1 gateway, `/srv/nfs/rpi/trixie/boot/`, which every netbooted host at ps1 boots from: on 7 October 2026 twelve hosts at ps1, not only the four blades (read on the ps1 gateway, 6 and 7 October 2026). For (b) there are two ways, and which to take is Carl's choice (verifying 3 has both, with their undo):

- **Change the shared directory.** At their next boot all twelve lose their console and login on the header's serial pins, and the firmware's boot messages there. They include pi21 at ps1, a Tiny Tapeout host by fpgas.online-infra's inventory (not seen by us), and seven hosts we cannot name. Undo: put the two files back (keep a copy of both first).
- **Change one blade only.** Copy the directory, make the change in the copy, and point only that blade's link in `/srv/tftp/` at the copy. Undo: point the link back at `/srv/nfs/rpi/trixie/boot`. This is how pi20 at ps1 was tested on 7 October 2026; its link points at the shared directory again (since 17:19:48 that day), so a change to the shared directory reaches pi20 at ps1 as it reaches the other hosts.

The order of work. The cables come first: making them, checking them on the bench and fitting them needs nothing from the gateway. Then the Host column's (a), which the serial-pair checks will need in a boot with the serial port on once a card is converted (on an unconverted card they are `not run`). Then (b), only when JTAG is to run on a blade, which is what converting a card needs: with the serial port off, `/dev/ttyAMA0` is not there, so the `p2-uart`, `p2-serial` and `scratch` tests cannot pass in that boot (verifying 3). Both are changes to the gateway's boot files, so they are Carl's to decide and to time, (b) for every host or for one blade only (above). A load over JTAG, which converting a card needs, is not part of this guide yet (the paragraph above the table).

What the Host column's (b) is, as run on pi20 at ps1 on 7 October 2026, and what followed: What fpgas.online ran on the ps1 blades, 7 October 2026.

The Host column's (a) has no written steps and has **not been tried by us**. What is known: on pi20 at ps1 (read 7 October 2026) the login prompt on `/dev/ttyAMA0` was there only because of the word `console=serial0,115200` in `cmdline.txt`; with that word out and `enable_uart=1` kept, the header's serial port would stay on for the serial-pair tests with nothing else using it. That is the expectation, not a run.

pi20 at ps1's present P2 wiring is not the guide's, and its old serial wires may share GPIO14 with P1's TMS wire: the steps for taking them off without losing P1 are on Taking pi20 at ps1's old serial wiring off.

The parts are on Compute Blade cables: parts and tools. Once a blade is wired, check it.

Labels for the blades and cards at ps1, which exist and which wait, are on Labels at ps1; in the printed booklet the label sheets are the last chapter.

The blades have no page under `https://ps1.fpgas.online/fpgas/`: pi14 at ps1, pi16 at ps1, pi18 at ps1 and pi20 at ps1 all return 404 there (checked 2026-09-03 and again 2026-10-06).

Source: live probes (`lspci -nn`, `openFPGALoader --detect` and `--read-dna`, pull-up/pull-down on each P2 and JTAG line, the pin-ID check on pi20 at ps1); MACs and switch ports cross-checked against infra `host_vars/ps1.fpgas.online.yml`.

Taking pi20 at ps1’s old serial wiring off

You are about to fit the guide’s P2 cable on pi20 at ps1 and need its old serial wiring off without losing its P1 (JTAG) cable. What each blade still needs is on Acorns at ps1.

Its serial pair is wired straight to GPIO14 (J2) and GPIO15 (K2), with no resistor: read with the pin-ID design on 31 August 2026, when a design driving J2 stopped JTAG until a PoE cycle, so J2 and P1’s TMS share GPIO14 (fpgas.online-test-designs issue 4, the comment of 2026-08-31). GPIO14 is on Extension Port pin 9 and on UART header pin 3 (Uptime Lab’s GPIO guide).

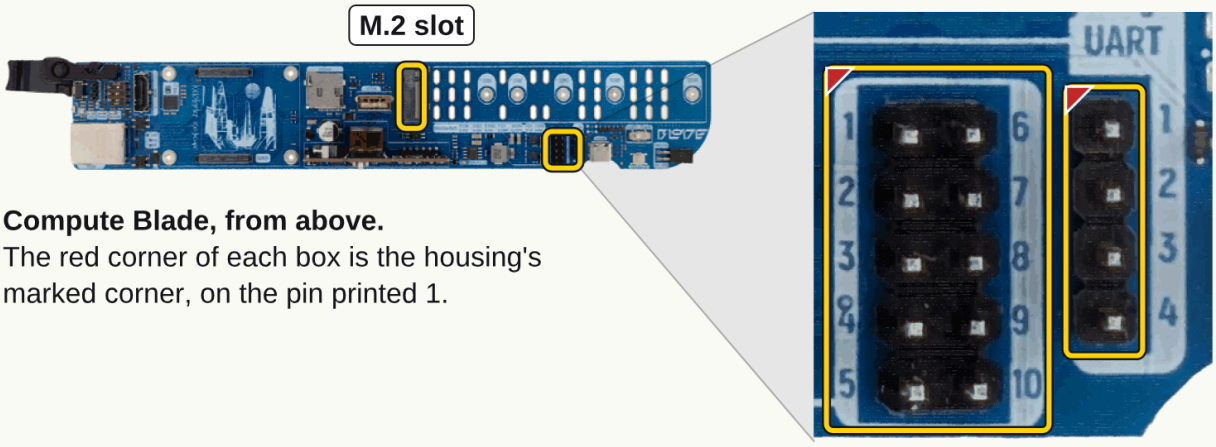
Which numbering these are. “Extension Port pin 9” and “UART header pin 3” are the Compute Blade’s own numbers, the ones printed on the blade beside its two headers (the silkscreen, seen in the picture below), and drawn on the right of the picture captioned “Compute Blade: the card, then the housings” on the Fitting sheet (headed “Compute Blade cables: fitting”; it is reproduced below). They are not the Raspberry Pi’s 40-pin numbers: on the Raspberry Pi’s 40-pin header GPIO14 is pin 8, and the blade’s pin 9 is that header’s pin 8. If you know the Pi’s numbers, do not count by them here. On the Extension Port, pin 1 is at the top of the left column and pin 6 at the top of the right, so pins 1 to 5 run down the left column and 6 to 10 down the right; pin 9 is the fourth from the top in the right column, and pin 10 the last. On the UART header pin 1 is the top one and pin 4 the bottom one, with pin 3 the third. Which GPIO each pin carries (pin 9 and GPIO14, for one) is from Uptime Lab’s GPIO guide, not measured by us.

Compute Blade: the card, then the housings

after the plugs are in their sockets

4. **Put the Acorn in the M.2 slot and fit its screw: the M2x2.5 standoff goes in the threaded insert for the card's size, with a 5 mm hex driver; put the M2 nylon washer on the M2x2.5 screw, insert the card at a 30° angle and press down, then secure the screw to the standoff with a PH1 driver (from Uptime Lab's assembly guide, not measured by us).**

↓



Compute Blade, from above.
The red corner of each box is the housing's marked corner, on the pin printed 1.

5. **Fit the P1 housing on the Extension Port, marked corner on pin 1, and the P2 housing on the UART, marked corner on pin 1.**

6. **Before powering on, look at both housings again, as on the bench check: the P1 housing's marked corner is on pin 1 of the Extension Port, and the P2 housing's on pin 1 of the UART. Turned round, the P1 housing puts 5 V on the TCK wire and the P2 housing puts 5 V on the K2 wire, above the most AMD's Artix 7 data sheet allows on an FPGA pin (DS181, Table 1: VCCO + 0.55 V, with VCCO at most 3.6 V), which can destroy the FPGA pin on the Acorn that wire reaches. Fitted, the housings get only this look; the meter check was the bench check's.**

Which Extension Port and UART pins the old serial wires sit on, and whether J2 shares a terminal or a housing with P1’s TMS wire, is not recorded by us. Build a new P2 cable by the guide (UART connector 1 and 2) first. Then ask Tim, and power the blade off (unplug its PoE cable, and a USB-C cable if one is

plugged in), take the card out of its M.2 slot (touch bare metal of the unplugged blade first, and hold the card by its edges, as on the cable pages), and look at Extension Port pins 9 and 10 and UART header pins 3 and 4: note which housing sits where.

1. Take the old serial wiring off the blade and the card. If a serial wire shares a housing, a terminal or a splice with P1's TMS wire, do not cut or pull it: take P1's cable off with it, and build a new P1 cable by the guide (JTAG connector 1 and 2).
2. Otherwise take P1's housing off its header and its plug out of the card's socket P1 too, and check it as JTAG connector 2 steps 3 and 4 do: each plug contact beeps to its cavity and to no other, and contact 6 (VCC, 3.3 V from the Acorn) beeps to no cavity at all: in the guide's cable its wire is cut off short and insulated (JTAG connector 1 step 5): look at it, a short stub with tube over its cut end. The bench check cannot tell whether wire 6 reaches a header pin, so this look and the meter are what keep its 3.3 V off the host. If any of this fails, or the housing is not the guide's 2×5, build a new P1 cable.
3. Run the bench check with the new P2 cable and the P1 cable, kept or new, the card still out, as that page says. Its ground beep is meant to show that no housing is turned round (that a turned housing would then stay silent is not tried by us, as that page says).
4. Then fit both cables and the card as Fitting does. Whether P1's TMS works is shown only by the check's `jtag` test, in a boot with the header's serial port off (verifying 3); the meter cannot reach it with the card fitted.

What fpgas.online ran on the ps1 blades, 7 October 2026

You look after the blades at ps1 and want to know what we ran on them on 7 October 2026, what it changed, and what followed. What each blade still needs is on Acorns at ps1; every time here is Adelaide time.

Test 4: JTAG with the header's serial port off (pi20 at ps1)

For this test pi20 at ps1's link in the gateway's `/srv/tftp/` was pointed at a copy of the shared boot directory, with two changes made in the copy only: in `config.txt`'s `[all]` section `enable_uart=1` became `enable_uart=0` and the line `uart_2ndstage=1` was taken out; in `cmdline.txt` the word `console=serial0,115200` was taken out (verifying 3 has the steps, both ways of making them and their undo). The blade was rebooted into the copy at 14:22 and was up again at 14:25.

In that boot GPIO14 had no user, the login prompt went to the Compute Module 5's own debug serial port (not on the header), so nothing more had to be taken off the header's pins, and the check's `jtag` test read the FPGA's IDCODE and device DNA (after checking that GPIO14 followed the Pi's pull both ways, verifying 3's step for a blade with no 470 Ω in its J2 wire, as pi20 at ps1 has). Nothing was loaded into the card in test 4. The IDCODE read `0x13631093`, the same part as on 20 September 2026 under kernel 6.12.75, when it read `0x3631093` (that reading left out the version digit); the DNA was `0x0028e5c45e304854`. Earlier on 7 October, with the serial port on, the check's `jtag` test had failed: GPIO14 is held by the serial port.

Test 6: converting pi20 at ps1's card, begun and stopped

Do not load a design into a card on a blade or convert it: that is not in this guide yet, because of what follows.

In the same boot, at 15:23, a load of our SoC into the FPGA (its SRAM, not its flash) over JTAG worked. The card's PCIe endpoint did not come back on a bus rescan. At step 1b a root-complex re-probe failed (the bind answered "No such device" and the root port was gone too), the run stopped there, and the blade was rebooted at 15:52.

From then on pi20 at ps1 restarted every 2 to 3 minutes until about 18:09, then stayed up; one boot, from about 17:06, stayed up until we rebooted it at 17:20:12. No power cycle was run by us; whether the load, the re-probe or the reboot caused it is not known. An `openFPGALoader --reset` at 17:19 put the vendor's sample image back in the FPGA, from the flash, and at 17:20:12 we rebooted the blade into the shared boot files (the next section). **The Acorn's flash was never written**: at 18:10 the card read as before the tests, `10ee:7011` at 5 GT/s x1.

The gateway's boot files

At 17:19:48 pi20 at ps1's link was put back at the shared directory `/srv/nfs/rpi/trixie/boot`, and the copy was removed. **`config.txt` and `cmdline.txt` in Carl's shared boot directory were never changed**: they read the same before and after (the two files we checked; the changes were made in

the copy only). Every boot of pi20 at ps1 since 17:20 has come from the shared directory (the gateway's TFTP and NFS logs), and at 18:10 its header's serial port was on again (`serial0` was `ttyAMA0`).

The Compute Module 4 blades

On 20 September 2026 JTAG on pi14 at ps1 got no response (TCK floating: its P1 cable unmated or its TCK wire open). On 7 October 2026 pi14 at ps1 and pi18 at ps1 had only logins, pin reads, a pull test on GPIO2, GPIO3 and GPIO4 (each pull set and put back) and `vcgencmd`, which hung for good on both. **No check and no JTAG has been run on a Compute Module 4 blade since 20 September.**

pi20 at ps1's missing resistor

pi20 at ps1's serial pair is direct on GPIO14 and GPIO15 without the resistor. From the check's code, not from a run on a blade: `fpgas-verify`'s `p2-serial` test never has both ends driving a wire at once (the Pi's pins are inputs while the FPGA drives, and the FPGA's are inputs while the Pi drives), so it does not rely on the resistor, and the fpgas.online design treats J2 as an input except while that test drives it. What the missing resistor does not survive is a design that drives J2, such as pin-ID.

Compute Blade cables: overview

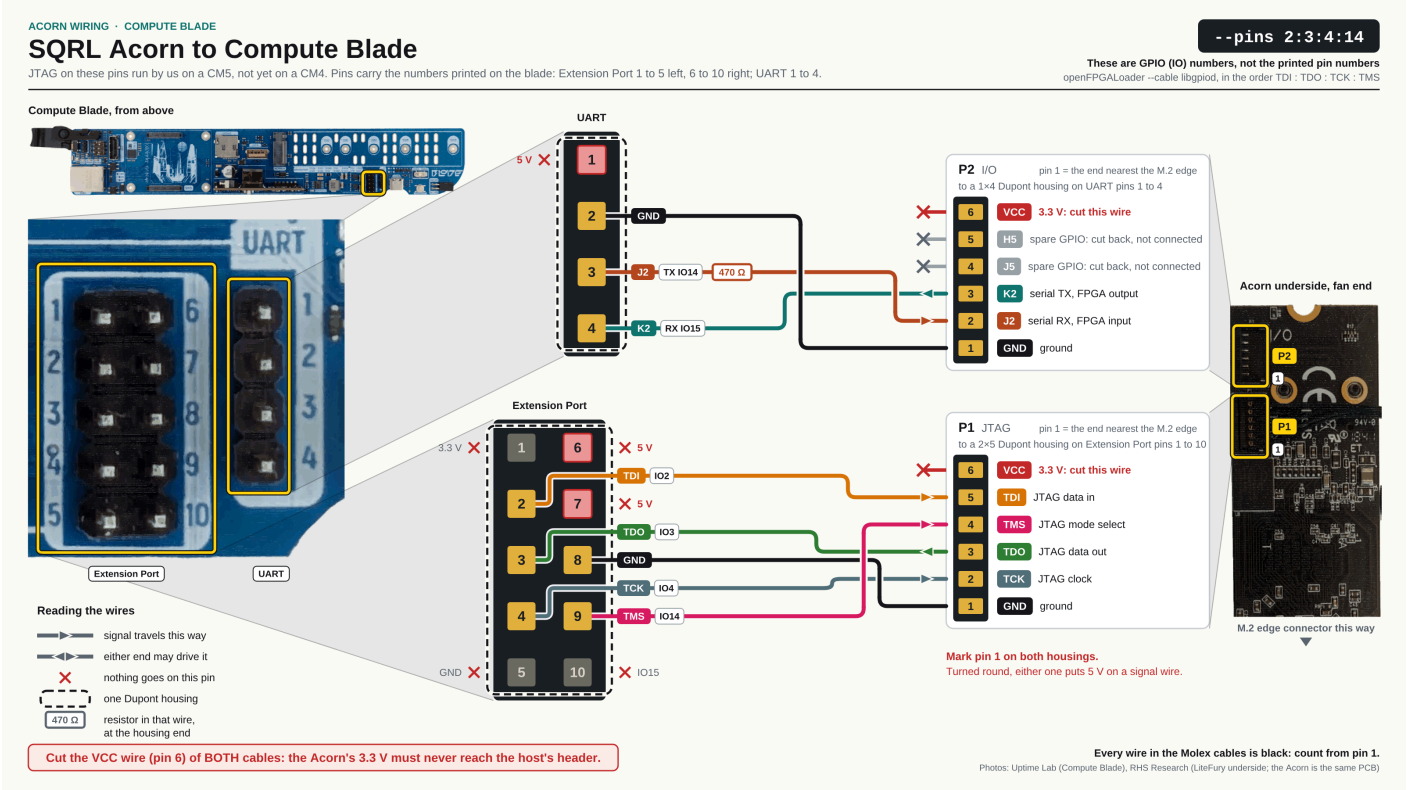
You have an Acorn and a Compute Blade, and want to build the two cables between them, fit them and check them. An Acorn on a Raspberry Pi 5 has its own guide; nothing there is for a Compute Blade.

Not yet run by us on this hardware: written from the design.

What you will have

Two short cables from the Acorn’s two connectors to the Compute Blade: the P1 cable carries JTAG, the P2 cable carries the serial port.

GND, TCK, TDO, TMS, TDI, J2 and K2 are the names on the pictures for each wire; J2 and K2 are the FPGA’s pin names.



[The finished wiring: Acorn to Compute Blade: full size on a landscape sheet of its own, sheet 19.]

Nothing in this guide cuts a wire to length. The bought cable is cut in half, once, as the first step (before the meter check, which needs the cut faces); each half is then used at the length it has, apart from the wires that are cut back at the plug and the one wire that is cut to take the resistor. Whether a half reaches from the card to the Compute Blade’s headers has not been measured by us: the guide checks it as its second step, before any wire is cut back or crimped.

The order of work

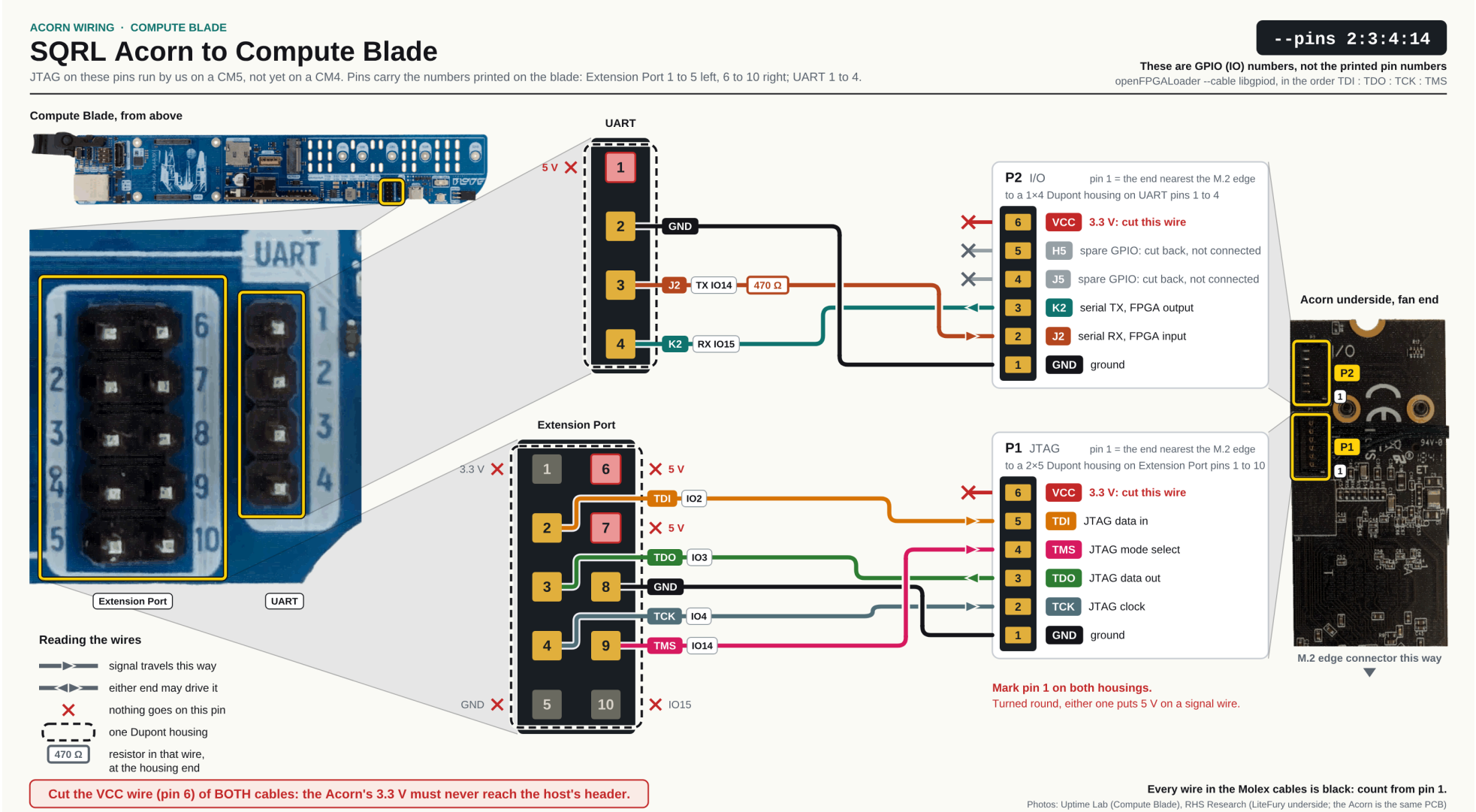
1. Parts and tools: the list to tick off before starting.
2. JTAG connector 1: the cable cut in half and each half checked for reach; the P1 cable’s wires flagged, checked with a meter, cut back and crimped.
3. JTAG connector 2: the P1 cable’s housing filled and checked.
4. UART connector 1: the P2 cable’s wires flagged, checked with a meter, cut back and crimped.
5. UART connector 2: the P2 cable’s housing filled and checked.
6. Bench check: both cables checked on the host before power, the card out of its slot.
7. Fitting: the plugs, the card and the housings go in.
8. Verifying: the check run on the host, and what a failing line means.

Where the facts come from

- The order of the pins in the Acorn’s two sockets (P1: GND, TCK, TDO, TMS, TDI, VCC; P2: GND, J2, K2, J5, H5, VCC), and that pin 1 is the end nearest the M.2 edge: the “Pico-EZmate connectors pinout on Acorn, Nite/LiteFury” legend on the LiteX Acorn CLE-215 wiki, and the silkscreen of RHS Research’s JTAG adapter. Which end of a plug is wire 1 is checked with a meter in the guide before anything is cut.
- That pin 6 (VCC) carries 3.3 V from the Acorn: not verified by us: taken from the same legend’s name for the pin. The guide cuts that wire back whatever it carries.
- How a card is put into the Compute Blade’s M.2 slot (standoff, washer, 30 degree angle, press down, PH1 driver): Uptime Lab’s assembly guide (https://raw.githubusercontent.com/uptime-lab/compute-blade/main/docs/docs/blade/getting-started/assembly.mdx) and its image page (docs.computeblade.com): not measured by us. How a card comes out is not given there, and is not recorded by us.
- The Compute Blade’s Extension Port and UART header (which pin is which): read off Uptime Lab’s board photo and their GPIO table; no Compute Blade has been wired this way by us yet.
- The Molex cable (part 0369200601: six circuits, a plug at each end, all six wires black, contacts 1.2 mm apart): not verified by us for this guide: from the part’s description and the cables in use at welland.
- That an Acorn CLE-215+, CLE-215 and CLE-101 are the same card, also sold as NiteFury and LiteFury, so that a LiteFury photo can stand for an Acorn: the LiteX Acorn CLE-215 wiki (“The Acorn CLE215+ is in fact the NiteFury board with more memory”), and RHS Research’s NiteFury-and-LiteFury repository, which gives the two boards’ FPGAs (NiteFury-II XC7A200T, LiteFury XC7A100T) and their size, 22 × 80 mm, but does not name the Acorn. That the CLE-101 is the LiteFury’s card: its FPGA answers as an XC7A100T (pi20 at ps1, 7 October 2026). That the CLE-215 is the same card as the CLE-215+: not verified by us. Not verified by us card against card.
- The card’s form, M.2 2280 with an M key: 22 × 80 mm from RHS Research’s repository (above); the M key: the Acorns at ps1 sit in the Compute Blades’ M.2 M-key slots (Uptime Lab’s M.2 guide) and their PCIe links come up (read on pi14, pi16 and pi20 at ps1, 7 October 2026).

- The 2.54 mm pitch of the housings: Uptime Lab’s GPIO guide says the Extension Port carries pins 1 to 10 of the Raspberry Pi GPIO pinout, a header of 2.54 mm (0.1 inch) pitch; not measured by us on a blade.
- The resistor’s 470 Ω and 1/8 W: our choice, not measured: 3.3 V across 470 Ω is at most 7 mA and 23 mW, well under 1/8 W (125 mW).
- That a Compute Module 4 takes the same wiring as a Compute Module 5: Uptime Lab’s assembly guide names both as modules for the Compute Blade, and its GPIO guide gives the blade’s pins by GPIO number and names no difference between the two; the wiring’s JTAG has been run by us on a CM5 only (pi20 at ps1, 7 October 2026). On a CM4: not run by us.
- The lengths (how far a wire is cut back, stripped, flagged): our choice, not measured.
- That a 2.54 mm Dupont crimp terminal crimps onto the Molex cable’s wire and holds: not verified by us: the wire’s gauge has not been measured or read from Molex’s drawing by us, and no cable has been crimped this way by us yet.
- That the 2×5 housing on the Extension Port and the 1×4 housing on the UART header fit side by side: not verified by us: Uptime Lab’s board photo shows the two headers next to each other; no housings have been fitted there by us.

Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).



The finished wiring: Acorn to Compute Blade (https://docs.fpgas.online/en/latest/_images/acorn-wiring-computeblade.png)

Compute Blade cables: parts and tools

You are about to build the two cables for an Acorn on a Compute Blade: tick off every line before you start.

The parts for **one** Compute Blade host; for several hosts, that many of each.

Have it	Qty	Part	Part number	What it is for
☐	1	Compute Blade (Uptime Lab)	—	its M.2 slot takes the card
☐	1	Raspberry Pi Compute Module 4 or 5	—	the host: it sits on the Compute Blade. JTAG on this wiring’s pins has been run by us on a CM5 only, not yet on a CM4
☐	1	SQRL Acorn CLE-215+, CLE-215 or CLE-101 (the same card is sold as LiteFury and NiteFury)	—	M.2 M-key, 2280
☐	1	Molex Pico-EZmate cable assembly, 6 circuits, a plug at each end	Molex 0369200601	cut in half: one half is the P1 cable, the other the P2 cable. All six wires are black
☐	1	heat-shrink tube, about 2 mm, a few centimetres	—	over each cut-back wire end
☐	1	Dupont housing, 2×5, 2.54 mm pitch	—	over Extension Port pins 1 to 10; 5 of its 10 cavities stay empty
☐	1	Dupont housing, 1×4, 2.54 mm pitch	—	over UART pins 1 to 4; 1 of its 4 cavities stays empty
☐	8	Dupont female crimp terminal, 2.54 mm	—	one for each connected wire: GND, TCK, TDO, TMS, TDI, GND, J2, K2; buy a few more than this, as spares
☐	1	resistor, 470 Ω, 1/8 W axial	—	in series with J2
☐	1	heat-shrink tube, about 3 mm, a few centimetres	—	over the resistor and both joints

The tools to build, fit and check its two cables, once for any number of hosts:

Have it	Tool	What it is for
☐	multimeter with a continuity buzzer	all six wires of the cable are black: each is checked from the plug to its cavity before the cable is fitted
☐	wire strippers for fine wire, and side cutters	to cut the cable in half, strip about 3 mm from each wire, and cut back the wires that are not connected
☐	crimping tool for 2.54 mm Dupont terminals	one crimp for each connected wire
☐	soldering iron and solder	to join the series resistor into its wire
☐	hot-air tool for the heat-shrink tube	to shrink the tube over each cut-back end (VCC always), and over the series resistor
☐	a fine probe tip for the multimeter, or a sewing pin to hold against a probe	the plug’s contacts are 1.2 mm apart, and a terminal is reached through a small opening
☐	masking tape and a fine marker pen, or a paint pen	to flag the six wires with their numbers, and to mark the pin 1 corner of each housing
☐	a ruler marked in millimetres	the steps give lengths in millimetres: where a flag goes, how much of a wire is left, cut back or stripped
☐	a PH1 (Phillips size 1) screwdriver	for the M2x2.5 screw that holds the card in the M.2 slot. Uptime Lab’s assembly guide for the Compute Blade names a PH1 driver for that screw (and a 5 mm hex driver for its standoff, if that is not fitted yet); not verified by us on the blades
☐	a computer with ssh, on a network that reaches the blade	to log in to the blade and run the check (the page “verifying 1”)

Compute Blade cables: JTAG connector 1, prepare the wires

You have the parts for an Acorn on a Compute Blade and are making the first of its two cables, from the Acorn's P1 socket (JTAG). On this page: cut the bought cable in half, find wire 1, check it with a meter, cut back the wires that are not used, crimp the rest.

Not yet run by us on this hardware: written from the design. Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).

What you need

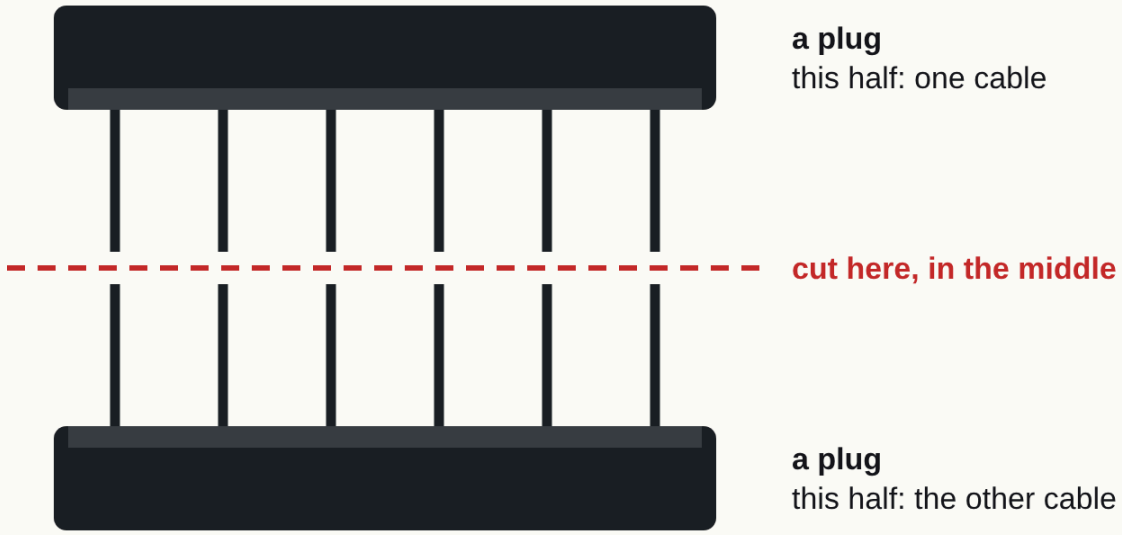
For this cable: one half of the Molex Pico-EZmate cable (a plug with six black wires); the 2×5 Dupont housing; 5 Dupont crimp terminals, and a few spare; 2 mm heat-shrink tube. Tools: a multimeter with a continuity buzzer, a fine probe tip for it or a sewing pin to hold against a probe, a ruler marked in millimetres, side cutters, wire strippers, the crimping tool, a hot-air tool, masking tape and a fine pen. The Acorn itself, out of any slot, is needed for the first steps: if it is fitted, ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. Then take the card out. Take its screw out first (an M2x2.5 screw with an M2 nylon washer, put in with a PH1 driver: from Uptime Lab's assembly guide, not measured by us). How the card then comes out of the slot is not recorded by us. The Acorn is a bare circuit board. Before you pick it up, touch bare metal of the host, or wear an anti-static wrist strap; hold the card by its edges.

Steps

1. Cut the Molex cable in half with side cutters. Each half is one cable. The cut comes before the reach check on purpose: a half is what goes between the card and the Compute Blade, so a half is what is checked for reach, below.

Cut the Pico-EZmate cable in half:

each half becomes one of the two cables



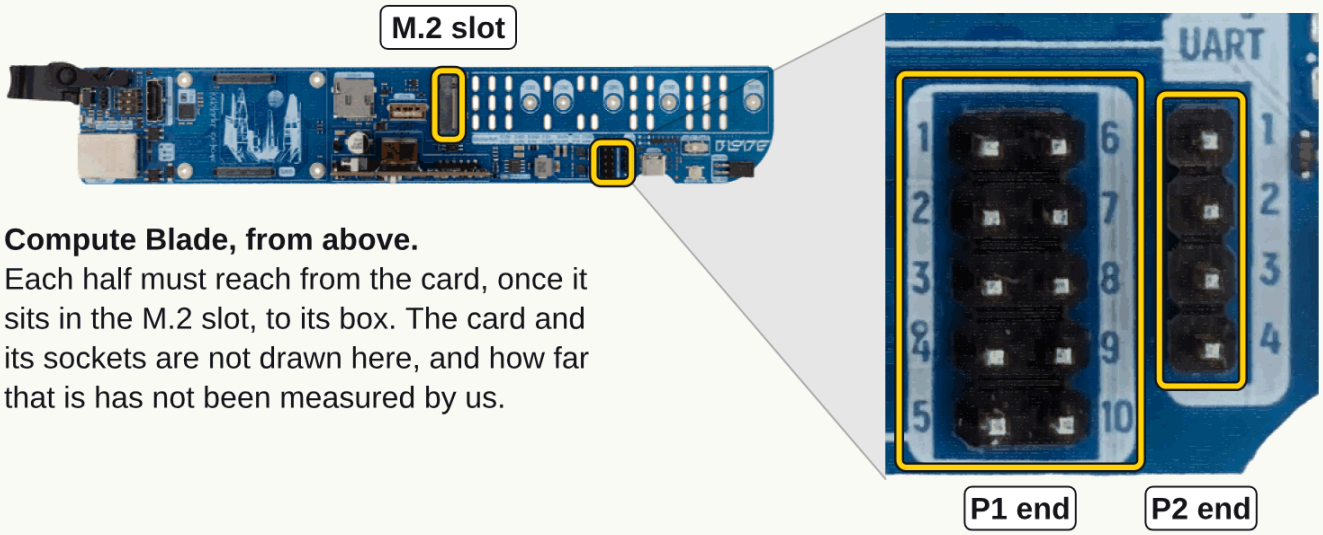
The two halves are alike. Either one can be the P1 cable or the P2 cable.

All six wires are black. Sketched, not from a photograph.

2. Check that each half reaches, before any wire is cut back or crimped. If the Compute Blade is on, ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. With the card out of its slot, hold the card over the slot where it will sit, and lay the half for P1 from socket P1 to the Extension Port and the half for P2 from socket P2 to the UART header, along the way each cable will run. Its cut end must reach the header with some slack left to bend into the housing; how much is needed has not been measured by us. If a half does not reach, stop: this guide uses each half at the length it has, and a longer cable is not written here. Tell Tim. **Not yet done by us on this hardware.**

Compute Blade: where each half must reach

from where the card will sit to the header



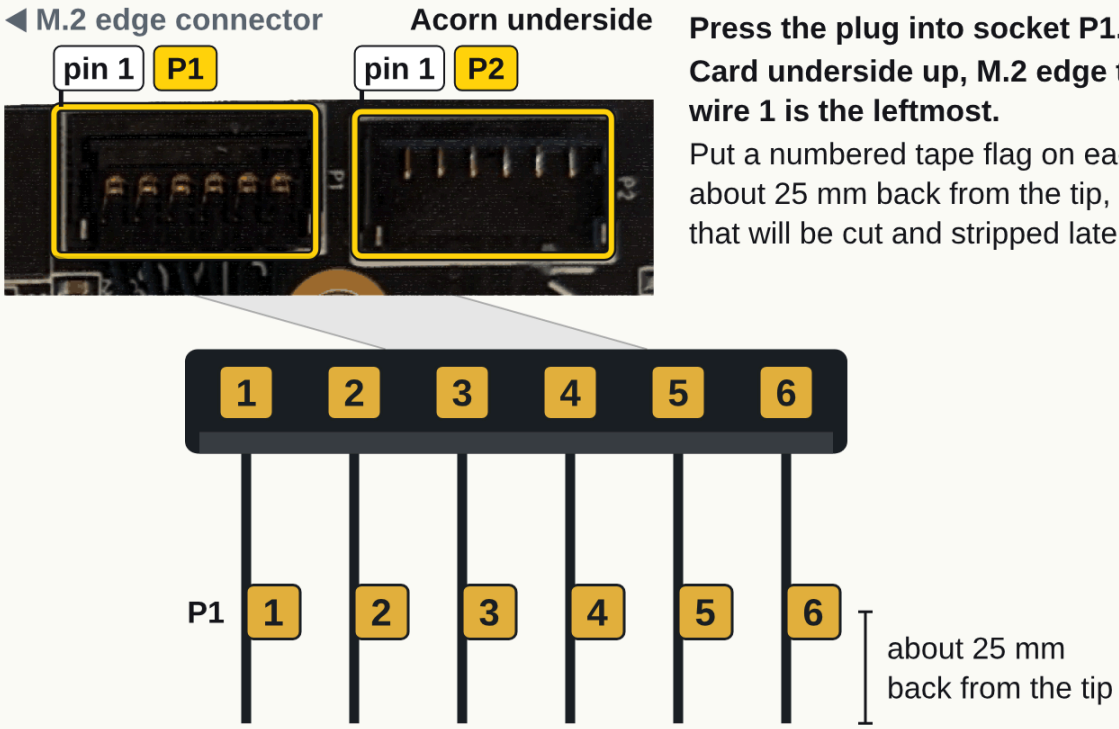
Compute Blade, from above.
Each half must reach from the card, once it sits in the M.2 slot, to its box. The card and its sockets are not drawn here, and how far that is has not been measured by us.

3. Find wire 1 of the P1 cable and flag the wires, before cutting any wire back.

1. Press the plug of one half into socket P1 on the underside of the Acorn. Hold the card underside up with the M.2 edge to your left: wire 1 is the leftmost.
2. Put a numbered tape flag on each of the 6 wires, 1 at the left to 6, about 25 mm back from the tip, clear of the end that will be cut and stripped later. Write P1 on the flag of wire 1 as well: off the card, the two halves look alike.

P1 cable (JTAG): flag the 6 wires

nothing is cut in this step



Press the plug into socket P1.

Card underside up, M.2 edge to your left: wire 1 is the leftmost.

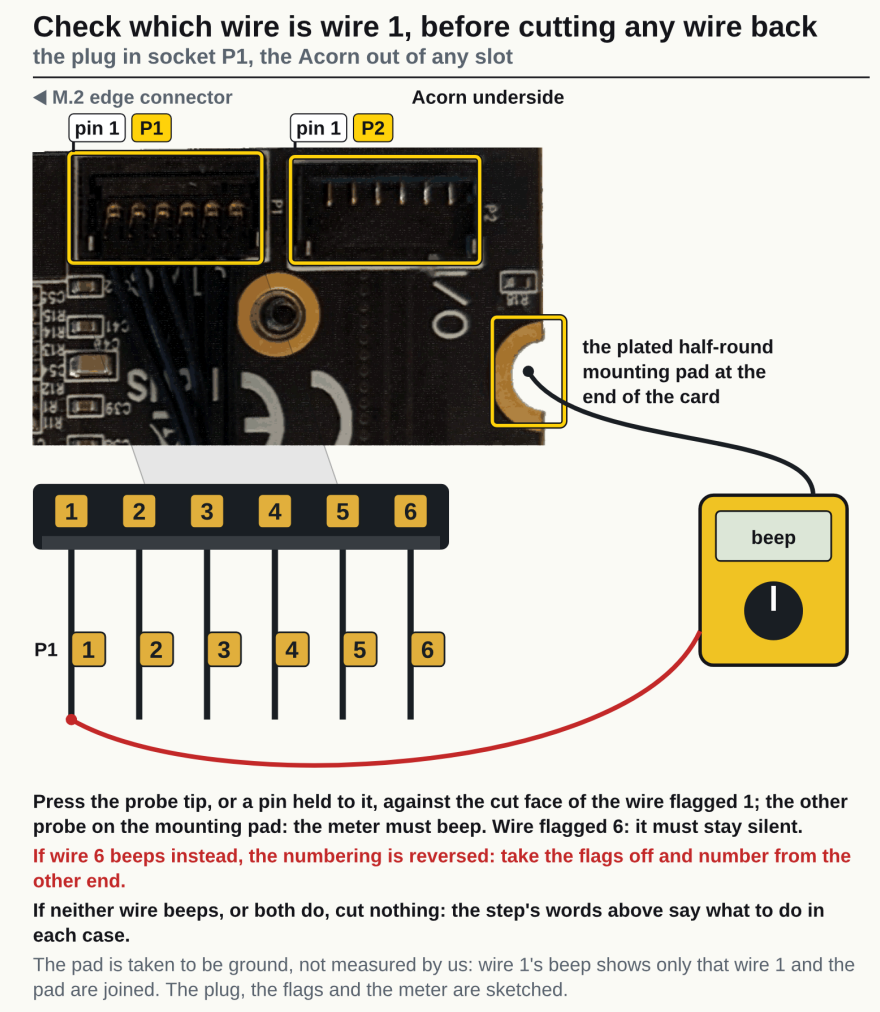
Put a numbered tape flag on each wire, 1 to 6, about 25 mm back from the tip, clear of the end that will be cut and stripped later.

All 6 wires stay whole in this step. Next: the meter check of wire 1.

All 6 wires are black. The plug and the flags are sketched.

4. Check which wire of the P1 cable is wire 1, with a meter, before cutting any wire back.

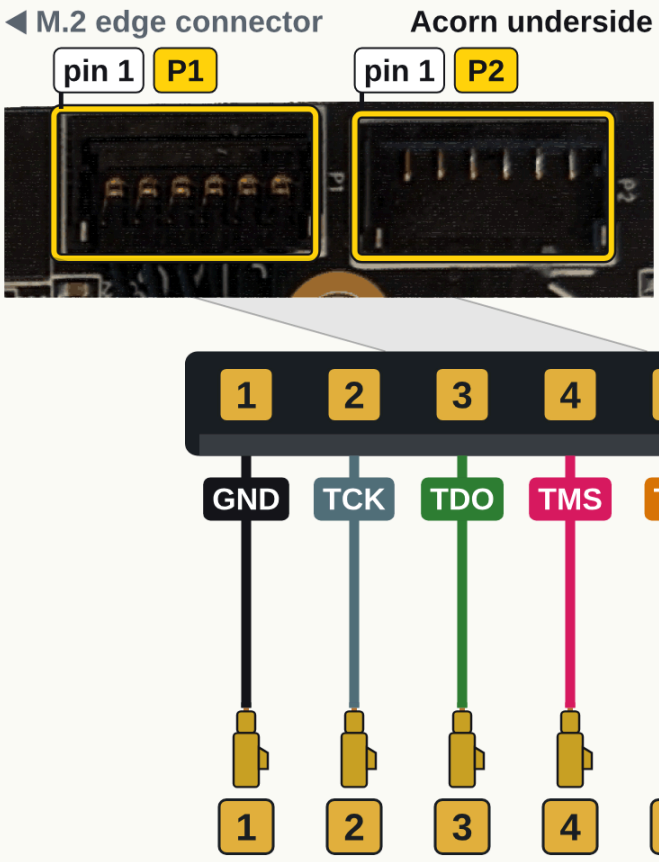
1. With the plug still in the socket (the Acorn out of any slot), set the meter to continuity. Press the probe tip, or a pin held to it, against the cut face of the wire flagged 1 (it is not stripped yet), and put the other probe on the plated half-round mounting pad at the end of the card: it must beep.
2. Do the same with the wire flagged 6: it must stay silent.
3. If wire 6 beeps instead, stop: the numbering is reversed; take the flags off and number from the other end. If neither wire beeps, strip about 3 mm from wires 1 and 6 and try again; if still neither beeps, stop: the ground point is not confirmed.
4. If both wires beep, first see that the two probes do not touch each other and that the cut faces of wires 1 and 6 do not touch. If both still beep, stop and cut nothing: the check cannot tell the wires apart. Leave the flags on and take the plug out. Wire 6 is the card's 3.3 V by the LiteX wiki's legend; that it stays silent to ground on a card with no power is expected and has not been measured by us. Set the meter to ohms, write down what each of the two wires reads to the pad, and send both readings to Tim.
5. Take the plug out again.



5. Cut wire 6 off about 10 mm from the plug and shrink a piece of the 2 mm tube over the cut end. It goes in no cavity. Wire 6 is VCC, 3.3 V from the Acorn (the LiteX wiki's legend names it so; not measured by us): it must never reach the host. Leave wires 1 to 5 at full length.

P1 cable (JTAG) for a Compute Blade:

which wires are cut back, which get a terminal



The plug is drawn under socket P1 to show which wire is which.

Card underside up, M.2 edge to your left: wire 1 is the leftmost.

Each wire keeps the flag you gave it before the wire 1 check.

The plug, seen from above as it sits in the socket.
Acorn turned underside up, M.2 edge to your left.

wire 6: cut off about 10 mm from the plug, heat shrink over the end
3.3 V from the Acorn: it must never reach the host.

Wires 1 to 5: leave at full length, strip about 3 mm, crimp a terminal on each.

Wire colours are for this picture only: the real wires are all black. Count from the plug's pin 1 end.

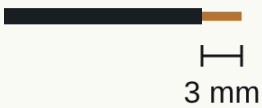
The plug and the terminals are sketched, not from a photograph.

6. Strip about 3 mm from wires 1 to 5. Crimp a Dupont terminal on each.

Crimp a terminal on each wire

Sketched, not from a photograph.

1. Strip about 3 mm.



2. Crimp the terminal on.



One crimp grips the bare wire, the other the insulation.

Compute Blade cables: JTAG connector 2, fill and check the housing

You have the P1 cable for an Acorn on a Compute Blade with its wires flagged and crimped, and are putting them into their housing. On this page: which wire goes in which cavity, and a meter check of every wire.

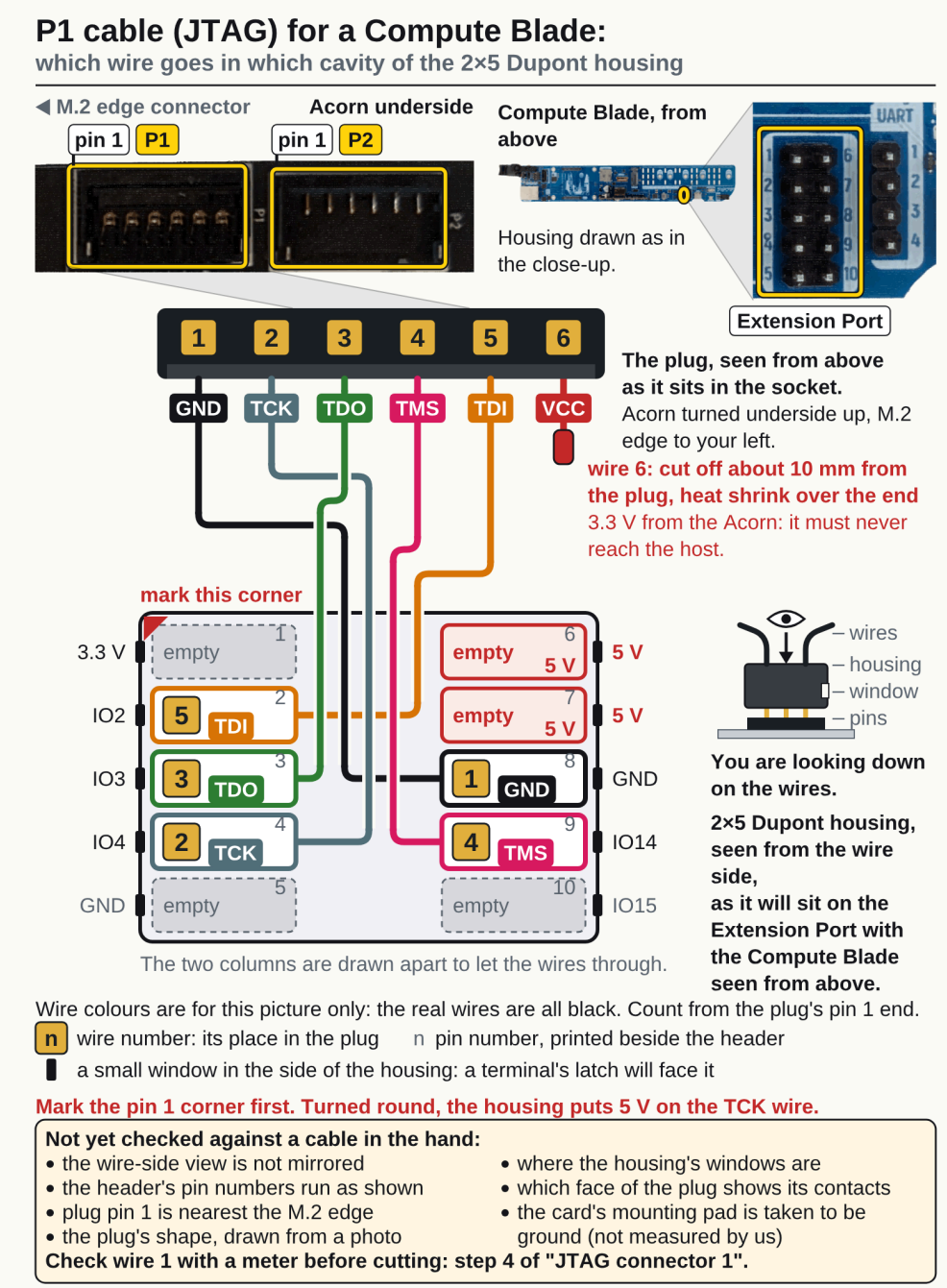
Not yet run by us on this hardware: written from the design. Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).

What you need

The P1 cable with its wires flagged and a terminal crimped on each (the page before this one), the empty 2×5 Dupont housing, a paint pen or a dot of tape, and a multimeter with a continuity buzzer and a fine probe or a sewing pin.

Steps

1. Hold the empty 2×5 housing with the wire openings facing you and its long side upright, as in the picture. Until it is marked, either way up is the same. Mark the top left corner with a paint pen or a dot of tape: that is the pin 1 corner. For each wire, read the number on its flag and find the same number in the picture: that is the cavity its terminal goes in. Turned round, the housing puts 5 V on the TCK wire, above the most AMD's Artix 7 data sheet allows on an FPGA pin (DS181, Table 1: VCCO + 0.55 V, with VCCO at most 3.6 V), which can destroy the FPGA pin on the Acorn that wire reaches. Wire 6 goes in no cavity.

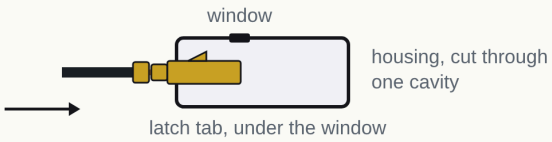


2. Push each terminal into its cavity, latch tab towards the window, until it clicks. Pull each wire gently: the terminal must stay in. The P1 cavity picture is shown again under the picture of how a terminal goes in, to read each wire's cavity from.

Push each terminal into its cavity

Sketched, not from a photograph.

1. Push it in, latch tab towards the window, until it clicks.

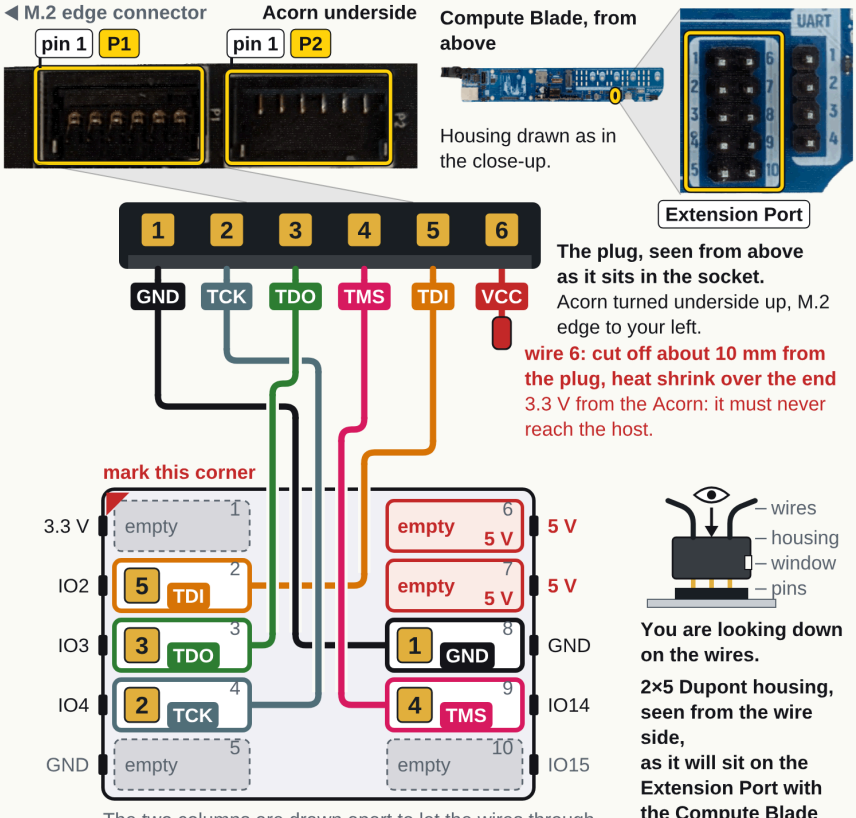


2. Pull the wire gently. The terminal must stay in.

▬ the window, as marked on the housing in the cavity picture

P1 cable (JTAG) for a Compute Blade:

which wire goes in which cavity of the 2×5 Dupont housing



Wire colours are for this picture only: the real wires are all black. Count from the plug's pin 1 end.

n wire number: its place in the plug n pin number, printed beside the header

▮ a small window in the side of the housing: a terminal's latch will face it

Mark the pin 1 corner first. Turned round, the housing puts 5 V on the TCK wire.

Not yet checked against a cable in the hand:

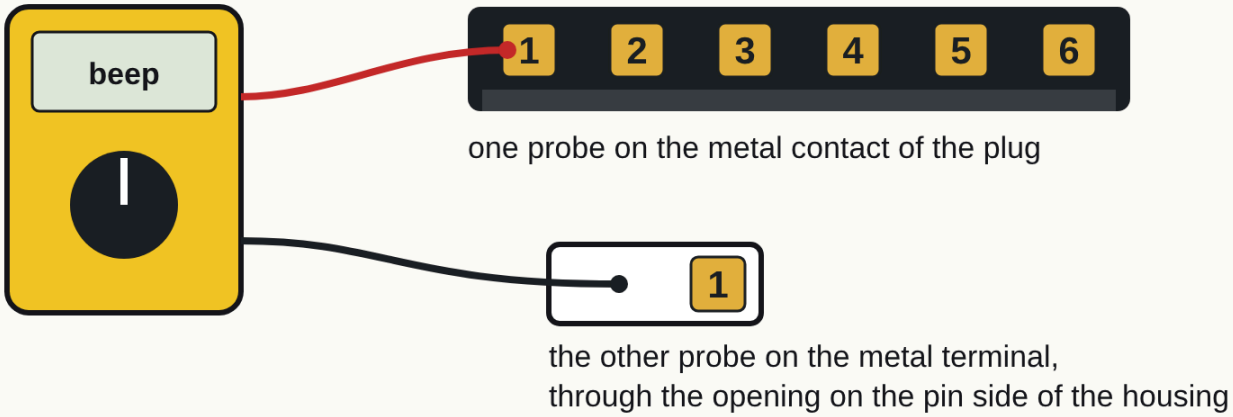
- the wire-side view is not mirrored
- the header's pin numbers run as shown
- plug pin 1 is nearest the M.2 edge
- the plug's shape, drawn from a photo
- where the housing's windows are
- which face of the plug shows its contacts
- the card's mounting pad is taken to be ground (not measured by us)

Check wire 1 with a meter before cutting: step 4 of "JTAG connector 1".

3. Check each wire with a meter on continuity. Set the meter to continuity. The probes go as in the picture: one on a contact of the plug, the other on a terminal, through the opening on the pin side of the housing. The plug’s contacts are 1.2 mm apart: use a fine probe or a sewing pin held to the probe.

Check every wire with a meter

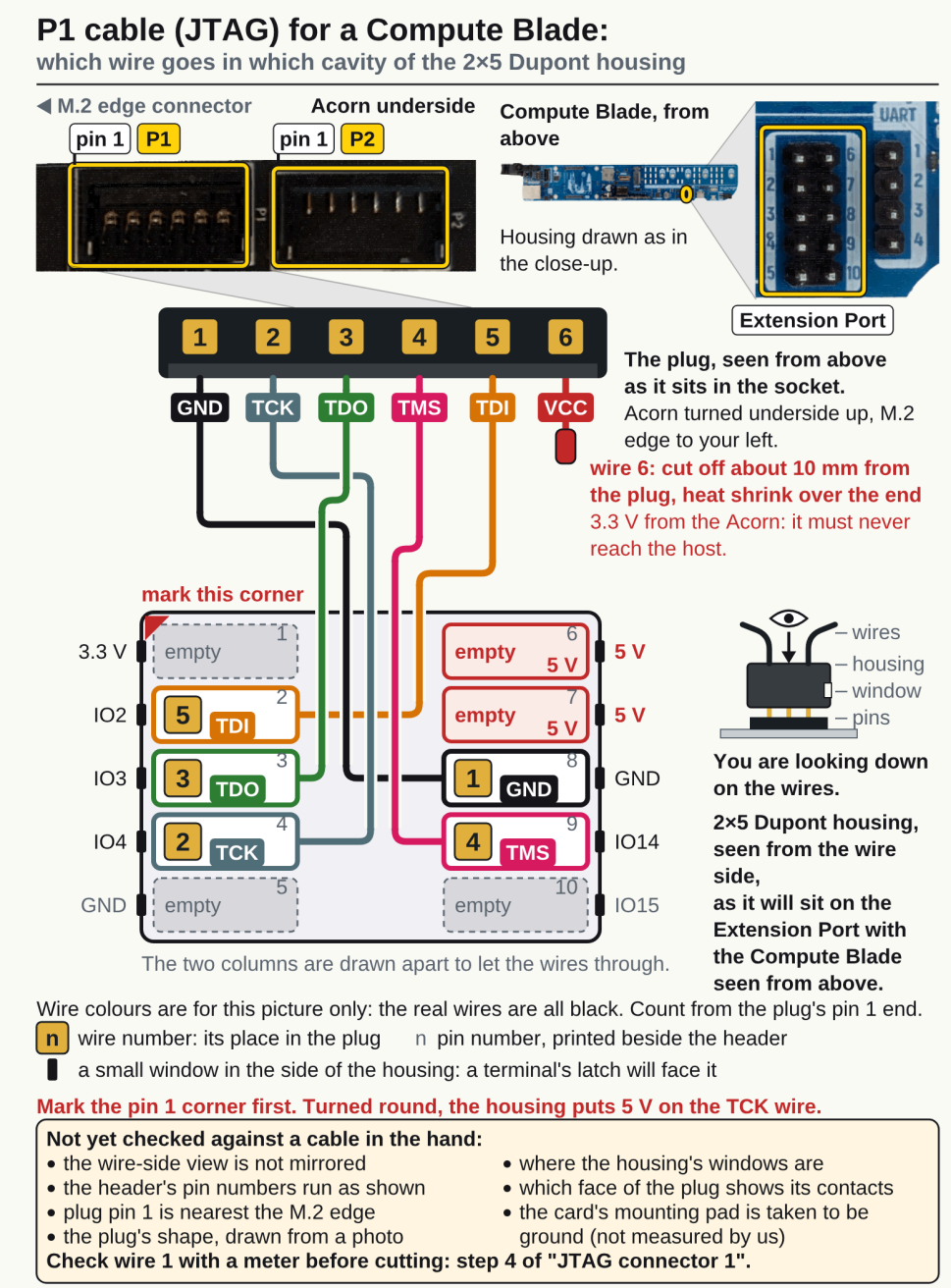
Sketched, not from a photograph.



Set the meter to continuity. For each wire, touch its contact on the plug and its terminal in the housing: the meter must beep.

Then try every other cavity: silent.

4. Now check each wire against its cavity. For each wire: read the number on its flag and find it in the picture; one probe on that wire's contact on the plug, the other on the terminal in that cavity: it must beep. Every other cavity must stay silent for that contact. The P1 cavity picture is shown again below, to read each wire's cavity from.



If a terminal is in the wrong cavity

A Dupont housing holds each terminal by a small plastic tab over its latch, in the window. Lift that tab a little with a pin and pull the wire gently: the terminal comes out, and can be pushed into the right cavity. (How these housings release; not yet done by us on these cables.)

Compute Blade cables: UART connector 1, prepare the wires

You have the parts for an Acorn on a Compute Blade and are making the second of its two cables, from the Acorn's P2 socket (the serial port). On this page: find wire 1, check it with a meter, cut back the wires that are not used, solder the resistor into the J2 wire, crimp the rest.

Not yet run by us on this hardware: written from the design. Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).

What you need

For this cable: the other half of the Molex Pico-EZmate cable, which was cut in half on the page "JTAG connector 1" (if it is still whole: cut it in the middle with side cutters; each half is one cable); the 1×4 Dupont housing; 3 Dupont crimp terminals, and a few spare; 2 mm heat-shrink tube; the 470 Ω resistor; 3 mm heat-shrink tube. Tools: a multimeter with a continuity buzzer, a fine probe tip for it or a sewing pin to hold against a probe, a ruler marked in millimetres, side cutters, wire strippers, the crimping tool, a hot-air tool, a soldering iron, masking tape and a fine pen. The Acorn itself, out of any slot, is needed for the first steps: if it is fitted, ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. Then take the card out. Take its screw out first (an M2x2.5 screw with an M2 nylon washer, put in with a PH1 driver: from Uptime Lab's assembly guide, not measured by us). How the card then comes out of the slot is not recorded by us. The Acorn is a bare circuit board. Before you pick it up, touch bare metal of the host, or wear an anti-static wrist strap; hold the card by its edges.

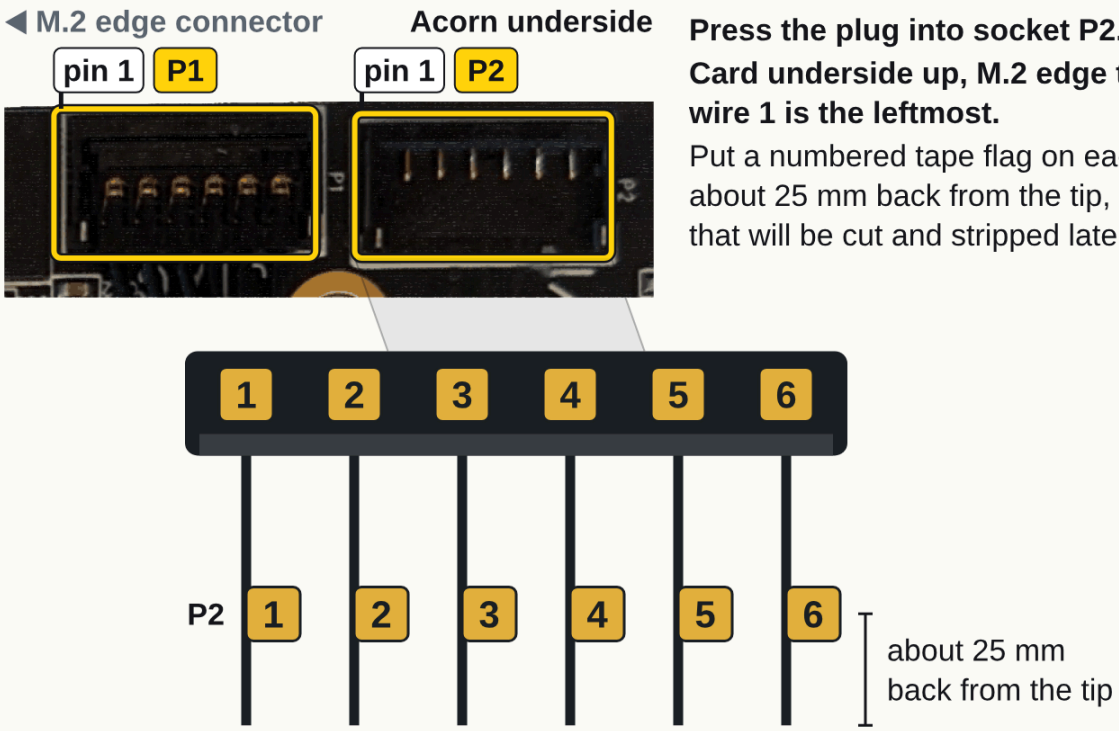
Steps

1. Find wire 1 of the P2 cable and flag the wires, before cutting any wire back.

1. Press the plug of the other half into socket P2 on the underside of the Acorn. Hold the card underside up with the M.2 edge to your left: wire 1 is the leftmost.
2. Put a numbered tape flag on each of the 6 wires, 1 at the left to 6, about 25 mm back from the tip, clear of the end that will be cut and stripped later. Write P2 on the flag of wire 1 as well: off the card, the two halves look alike.

P2 cable (I/O): flag the 6 wires

nothing is cut in this step



Press the plug into socket P2.

Card underside up, M.2 edge to your left: wire 1 is the leftmost.

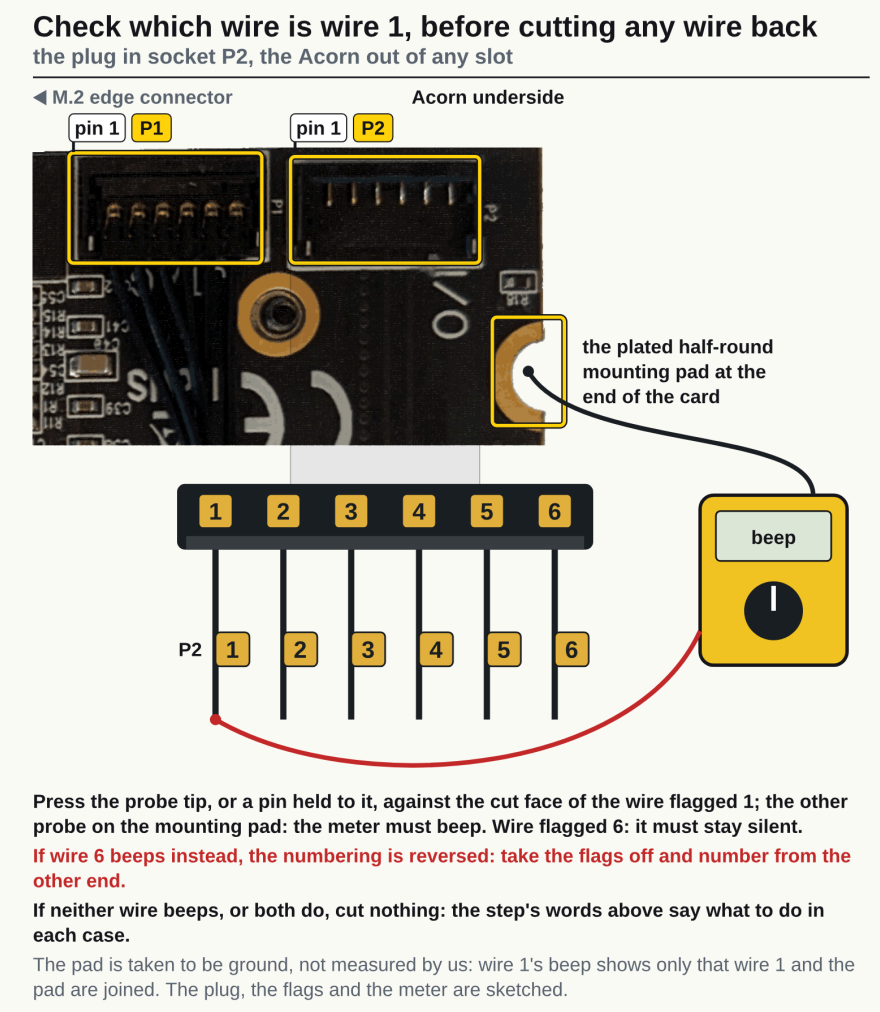
Put a numbered tape flag on each wire, 1 to 6, about 25 mm back from the tip, clear of the end that will be cut and stripped later.

All 6 wires stay whole in this step. Next: the meter check of wire 1.

All 6 wires are black. The plug and the flags are sketched.

2. Check which wire of the P2 cable is wire 1, with a meter, before cutting any wire back.

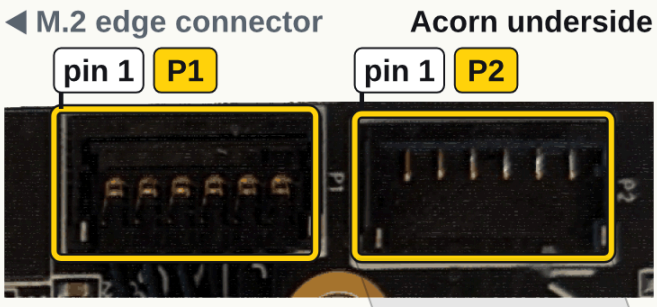
1. With the plug still in the socket (the Acorn out of any slot), set the meter to continuity. Press the probe tip, or a pin held to it, against the cut face of the wire flagged 1 (it is not stripped yet), and put the other probe on the plated half-round mounting pad at the end of the card: it must beep.
2. Do the same with the wire flagged 6: it must stay silent.
3. If wire 6 beeps instead, stop: the numbering is reversed; take the flags off and number from the other end. If neither wire beeps, strip about 3 mm from wires 1 and 6 and try again; if still neither beeps, stop: the ground point is not confirmed.
4. If both wires beep, first see that the two probes do not touch each other and that the cut faces of wires 1 and 6 do not touch. If both still beep, stop and cut nothing: the check cannot tell the wires apart. Leave the flags on and take the plug out. Wire 6 is the card's 3.3 V by the LiteX wiki's legend; that it stays silent to ground on a card with no power is expected and has not been measured by us. Set the meter to ohms, write down what each of the two wires reads to the pad, and send both readings to Tim.
5. Take the plug out again.



3. Cut wires 4, 5 and 6 off about 10 mm from the plug and shrink a piece of the 2 mm tube over each cut end. They go in no cavity. Wire 6 is VCC, 3.3 V from the Acorn (the LiteX wiki's legend names it so; not measured by us): it must never reach the host. Leave wires 1, 2 and 3 at full length.

P2 cable (I/O) for a Compute Blade:

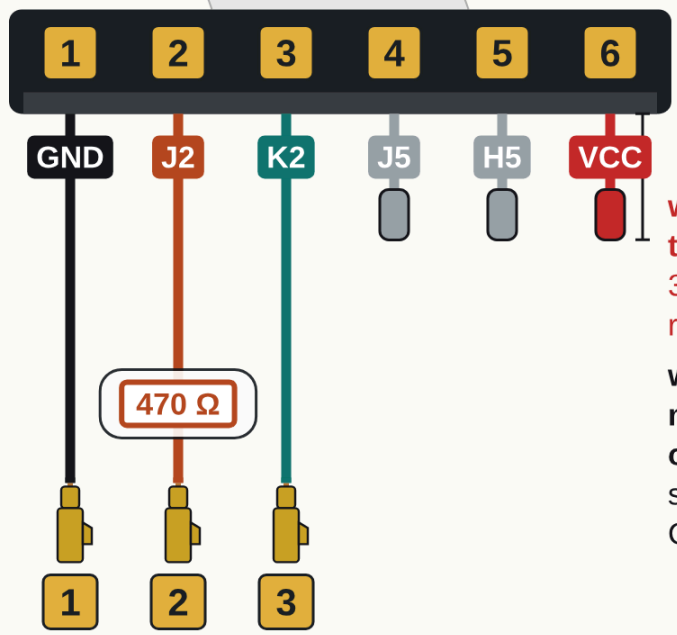
which wires are cut back, which get a terminal



The plug is drawn under socket P2 to show which wire is which.

Card underside up, M.2 edge to your left: wire 1 is the leftmost.

Each wire keeps the flag you gave it before the wire 1 check.



The plug, seen from above as it sits in the socket.
Acorn turned underside up, M.2 edge to your left.

wire 6: cut off about 10 mm from the plug, heat shrink over the end
3.3 V from the Acorn: it must never reach the host.

wires 4 and 5: cut off about 10 mm from the plug, heat shrink over the end
spare GPIO: not used on a Compute Blade.

Wires 1, 2 and 3: leave at full length, strip about 3 mm, crimp a terminal on each.

Wire 2 (J2): the 470 Ω resistor, in heat shrink, is soldered in before its terminal is crimped.

Wire colours are for this picture only: the real wires are all black. Count from the plug's pin 1 end.

The plug and the terminals are sketched, not from a photograph.

4. Cut wire 2 (J2) about 15 mm from its free end. Slide a piece of the 3 mm tube onto the wire, clear of the cut. Strip about 3 mm from each cut end. Trim the resistor's leads to about 5 mm each. Solder the 470 Ω resistor between the two cut ends. Slide the tube over the resistor and both joints and shrink it. Crimp the terminal only after this. The resistor is there because J2 lands on GPIO14, which is also JTAG TMS: with 470 Ω in the wire, JTAG still gets through if the FPGA drives J2 (designed so, not yet measured).

The 470 Ω resistor in wire 2 (J2) of the P2 cable

Sketched, not from a photograph.

1. Cut wire 2 about 15 mm from its free end.



2. Slide a piece of the 3 mm tube onto the wire, clear of the cut.



3. Strip 3 mm from each cut end. Trim the resistor's leads to 5 mm each. Solder it in.



4. Slide the tube over the resistor and both joints and shrink it.



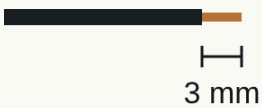
Crimp the terminal on this wire only after this.

5. Strip about 3 mm from wires 1, 2 and 3. Crimp a Dupont terminal on each.

Crimp a terminal on each wire

Sketched, not from a photograph.

1. Strip about 3 mm.



2. Crimp the terminal on.



One crimp grips the bare wire, the other the insulation.

Compute Blade cables: UART connector 2, fill and check the housing

You have the P2 cable for an Acorn on a Compute Blade with its wires flagged and crimped, and are putting them into their housing. On this page: which wire goes in which cavity, and a meter check of every wire.

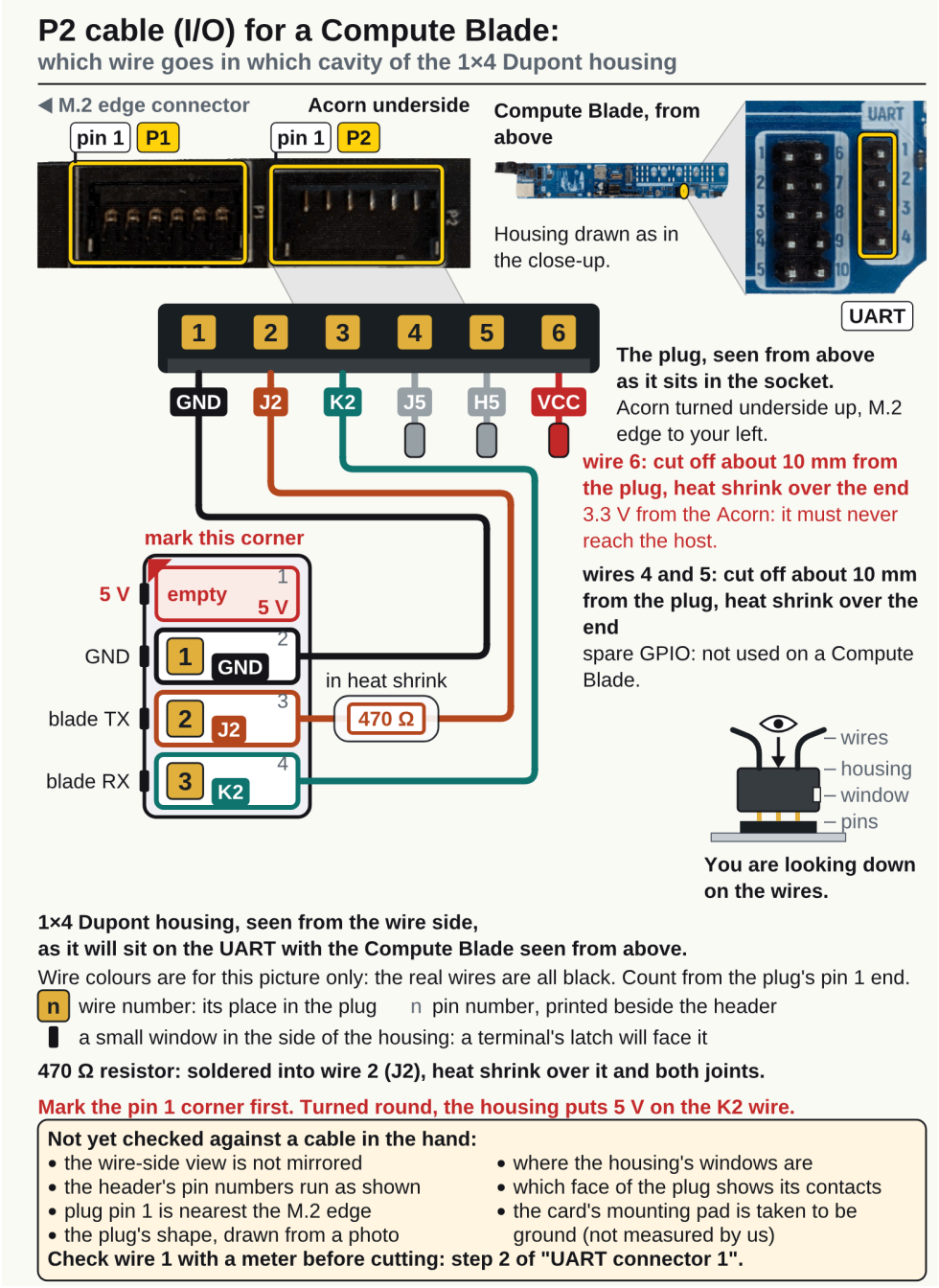
Not yet run by us on this hardware: written from the design. Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).

What you need

The P2 cable with its wires flagged and a terminal crimped on each (the page before this one), the empty 1×4 Dupont housing, a paint pen or a dot of tape, and a multimeter with a continuity buzzer and a fine probe or a sewing pin.

Steps

1. Hold the empty 1×4 housing with the wire openings facing you and its long side upright, as in the picture. Until it is marked, either way up is the same. Mark the top end with a paint pen or a dot of tape: that is the pin 1 corner. For each wire, read the number on its flag and find the same number in the picture: that is the cavity its terminal goes in. Turned round, the housing puts 5 V on the K2 wire, above the most AMD's Artix 7 data sheet allows on an FPGA pin (DS181, Table 1: VCCO + 0.55 V, with VCCO at most 3.6 V), which can destroy the FPGA pin on the Acorn that wire reaches. Wires 4, 5 and 6 go in no cavity.

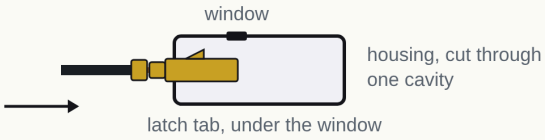


2. Push each terminal into its cavity, latch tab towards the window, until it clicks. Pull each wire gently: the terminal must stay in. The P2 cavity picture is shown again under the picture of how a terminal goes in, to read each wire's cavity from.

Push each terminal into its cavity

Sketched, not from a photograph.

1. Push it in, latch tab towards the window, until it clicks.

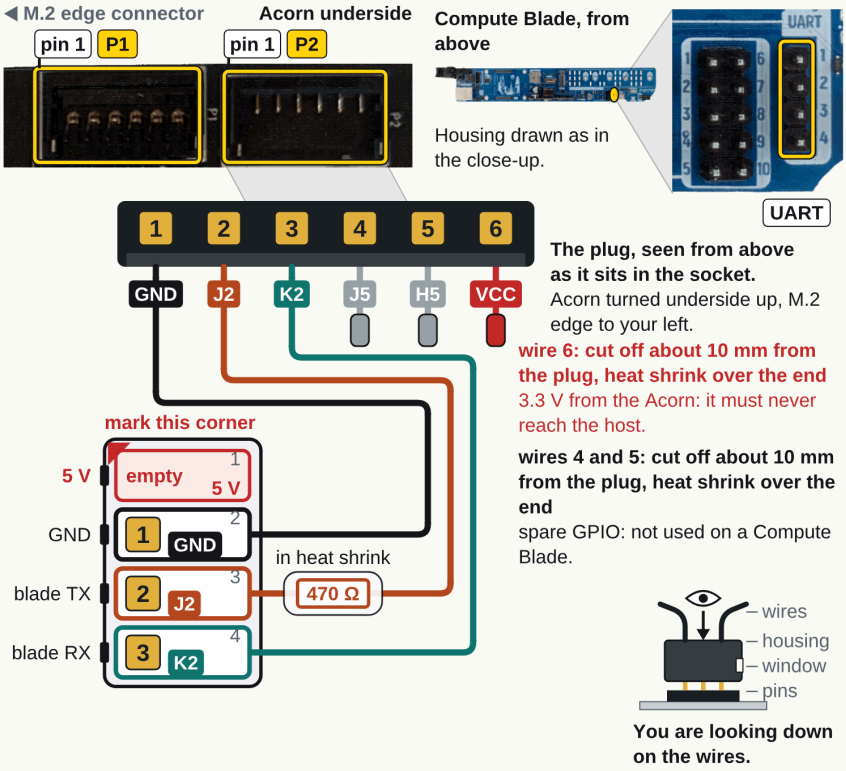


2. Pull the wire gently. The terminal must stay in.

▬ the window, as marked on the housing in the cavity picture

P2 cable (I/O) for a Compute Blade:

which wire goes in which cavity of the 1×4 Dupont housing



The plug, seen from above as it sits in the socket.
Acorn turned underside up, M.2 edge to your left.

wire 6: cut off about 10 mm from the plug, heat shrink over the end
3.3 V from the Acorn: it must never reach the host.

wires 4 and 5: cut off about 10 mm from the plug, heat shrink over the end
spare GPIO: not used on a Compute Blade.

You are looking down on the wires.

1×4 Dupont housing, seen from the wire side, as it will sit on the UART with the Compute Blade seen from above.

Wire colours are for this picture only: the real wires are all black. Count from the plug's pin 1 end.

n wire number: its place in the plug n pin number, printed beside the header

▮ a small window in the side of the housing: a terminal's latch will face it

470 Ω resistor: soldered into wire 2 (J2), heat shrink over it and both joints.

Mark the pin 1 corner first. Turned round, the housing puts 5 V on the K2 wire.

Not yet checked against a cable in the hand:

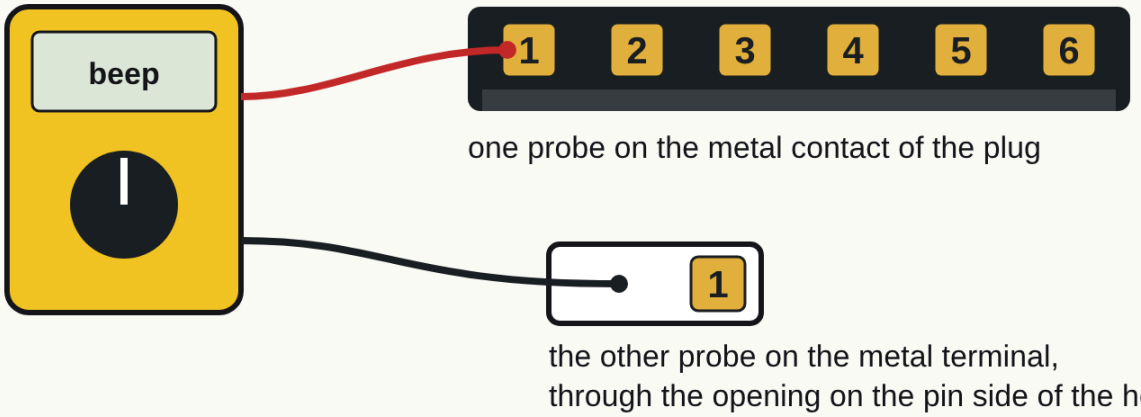
- the wire-side view is not mirrored
- the header's pin numbers run as shown
- plug pin 1 is nearest the M.2 edge
- the plug's shape, drawn from a photo
- where the housing's windows are
- which face of the plug shows its contacts
- the card's mounting pad is taken to be ground (not measured by us)

Check wire 1 with a meter before cutting: step 2 of "UART connector 1".

3. Check each wire with a meter on continuity. Set the meter to continuity. The probes go as in the picture: one on a contact of the plug, the other on a terminal, through the opening on the pin side of the housing. The plug’s contacts are 1.2 mm apart: use a fine probe or a sewing pin held to the probe.

Check every wire with a meter

Sketched, not from a photograph.

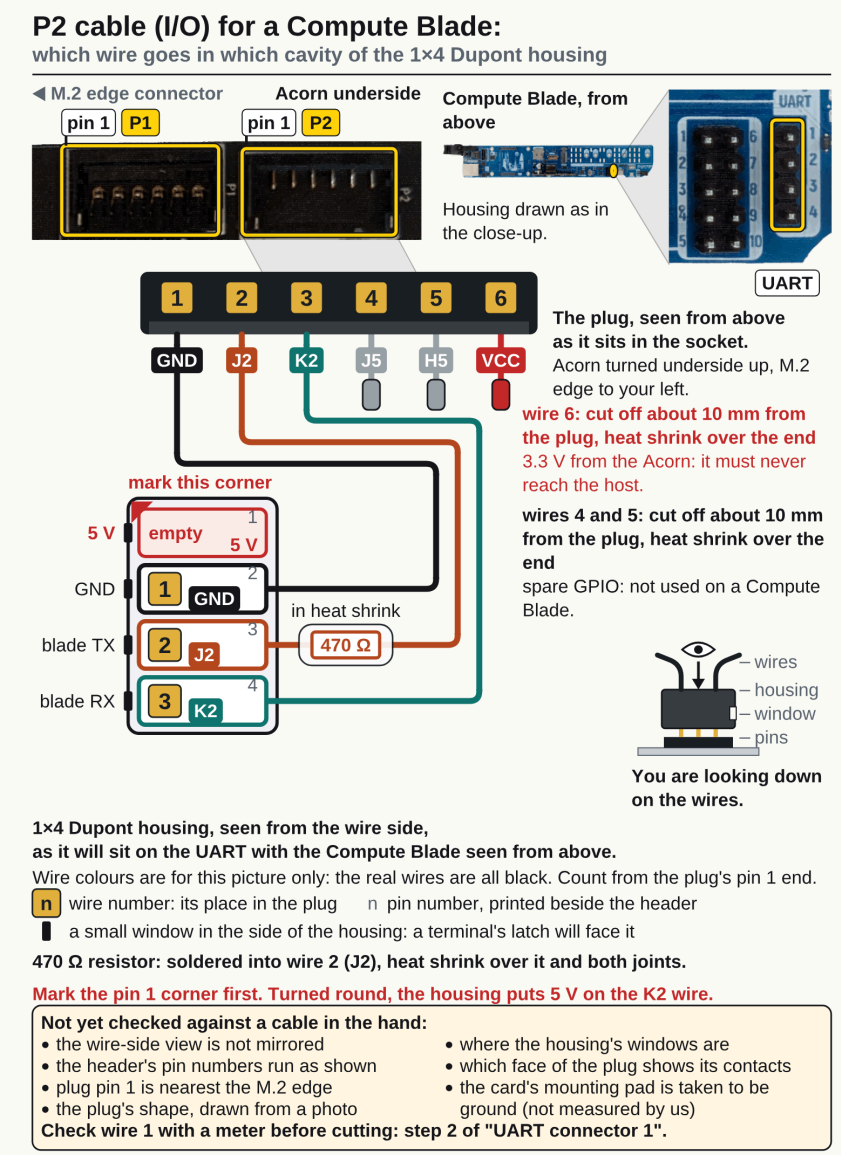


Set the meter to continuity. For each wire, touch its contact on the plug and its terminal in the housing: the meter must beep.

Then try every other cavity: silent.

Wire 2 has the 470 Ω resistor in it. It will not beep: read it in ohms, last, as the next step says.

4. Now check each wire against its cavity. For each wire: read the number on its flag and find it in the picture; one probe on that wire's contact on the plug, the other on the terminal in that cavity: it must beep. Every other cavity must stay silent for that contact. **Wire 2 is the exception: it has the 470 Ω resistor in it. Leave wire 2 until last and read it in ohms, not by the buzzer.** A continuity buzzer usually sounds only below some tens of ohms, so through the resistor it will usually stay silent (we have not tried your meter). Set the meter to ohms, on auto-range or the 2 kΩ range, and touch the two probes together first: it must read close to 0. Then, between wire 2's contact on the plug and its terminal, it must read close to 470 Ω. Close to 0 there means the resistor is bridged or was left out. Over-range (OL, or a lone 1 at the left of the display) or a value far from 470 Ω means a bad joint or the wrong wire. Between wire 2's contact and every other cavity it must show over-range. The P2 cavity picture is shown again below, to read each wire's cavity from.



If a terminal is in the wrong cavity

A Dupont housing holds each terminal by a small plastic tab over its latch, in the window. Lift that tab a little with a pin and pull the wire gently: the terminal comes out, and can be pushed into the right cavity. (How these housings release; not yet done by us on these cables.)

Compute Blade cables: bench check

You have built both cables for an Acorn on a Compute Blade. Before anything is powered, a check on the host with a meter that ground reaches the plugs and that the 3.3 V wire reaches nothing.

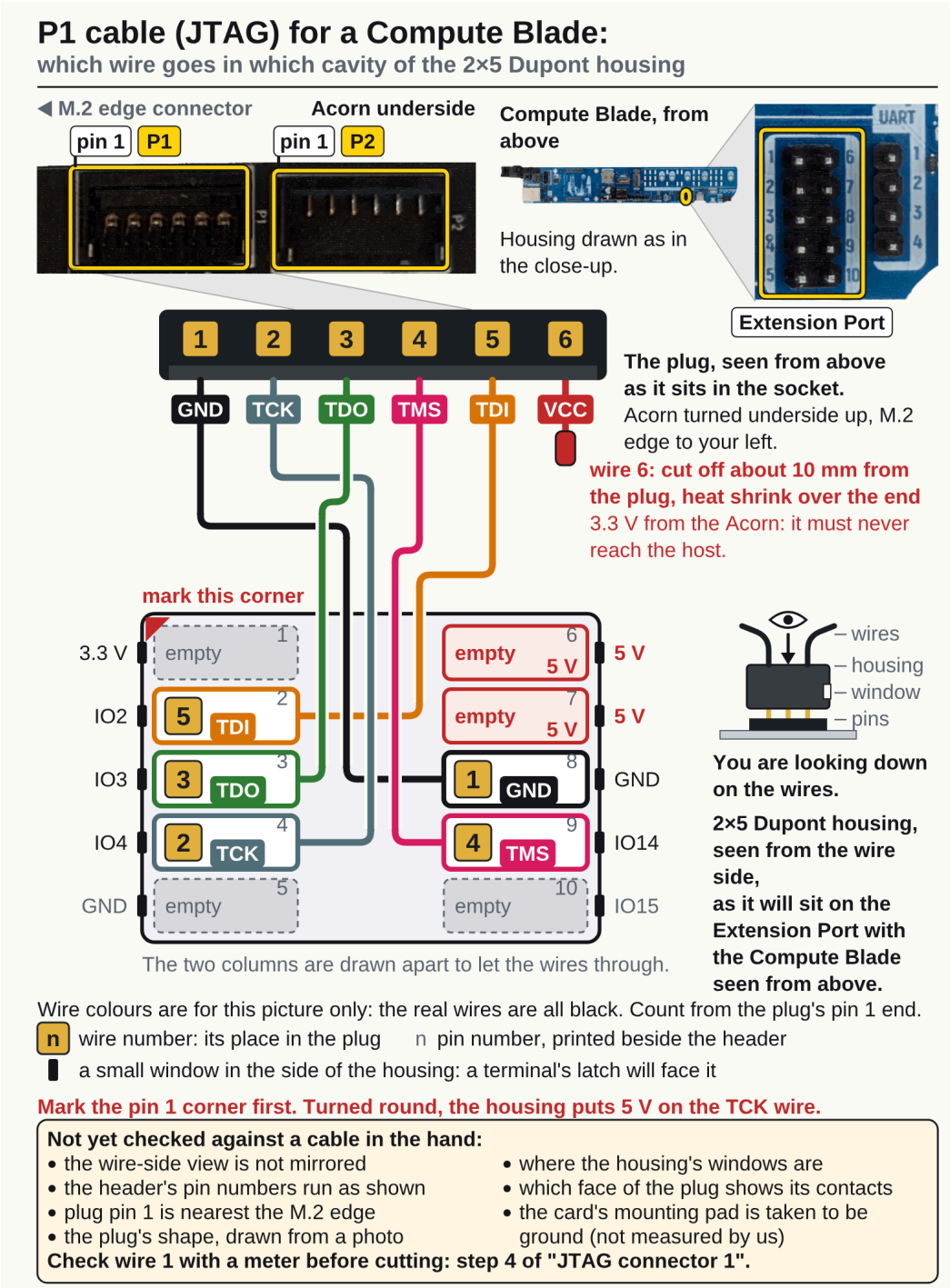
Not yet run by us on this hardware: written from the design. Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).

What you need

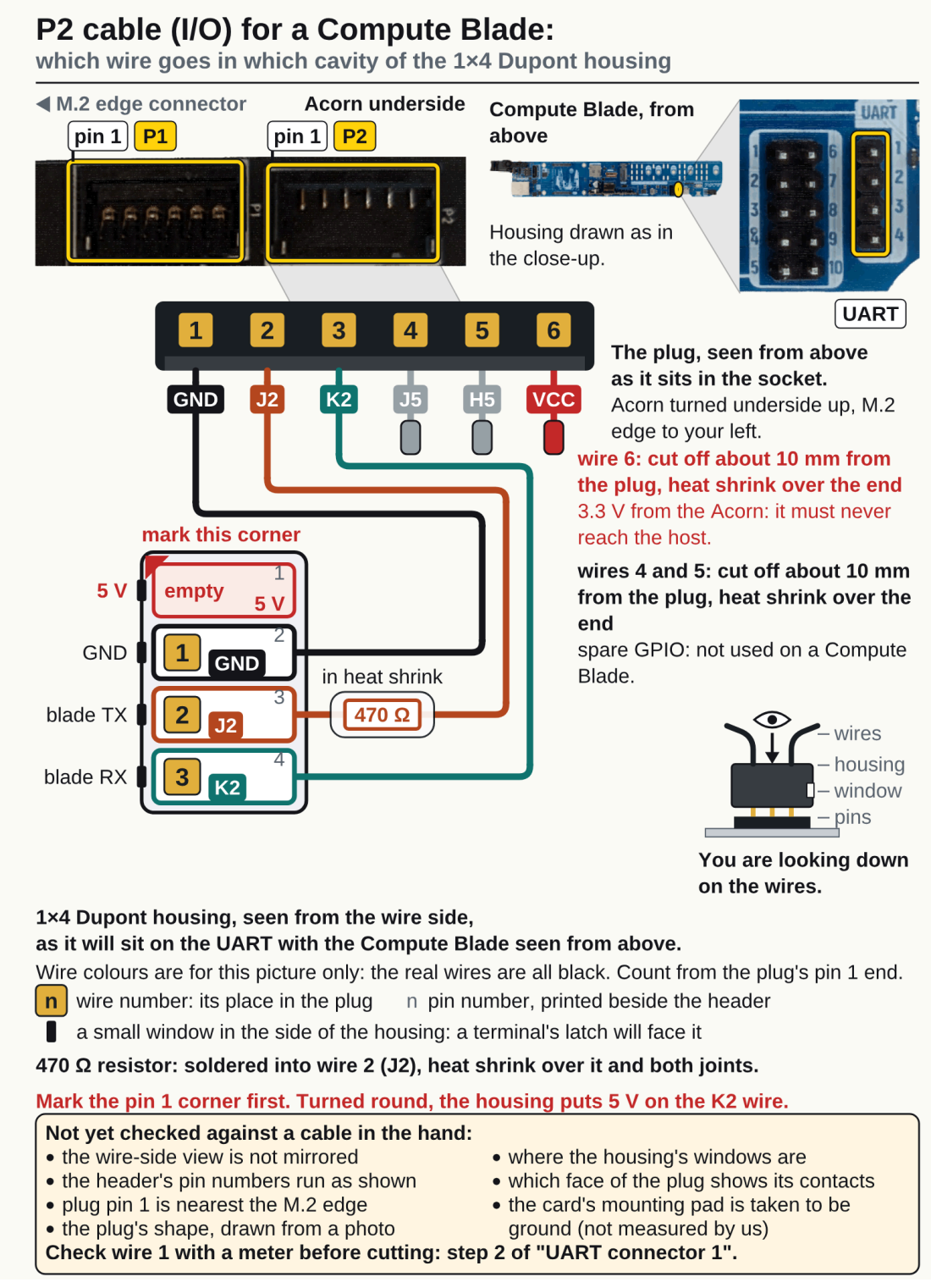
Both finished cables, the Compute Blade, and a multimeter with a continuity buzzer. If the Compute Blade is on, ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. The Acorn stays out of its slot; if it is fitted, take it out. Take its screw out first (an M2x2.5 screw with an M2 nylon washer, put in with a PH1 driver: from Uptime Lab’s assembly guide, not measured by us). How the card then comes out of the slot is not recorded by us.

Steps

1. This is a bench check; the housings come off again before the cables are fitted. If the Compute Blade is on, ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. Then fit the P1 housing on the Extension Port with its marked corner on the pin printed 1 beside the header. The P1 cavity picture is shown again below, for which wire goes in which cavity, and which corner is marked.



2. Fit the P2 housing on the UART with its marked corner on the pin printed 1 beside the header. The P2 cavity picture is shown again below, for which wire goes in which cavity, and which corner is marked.

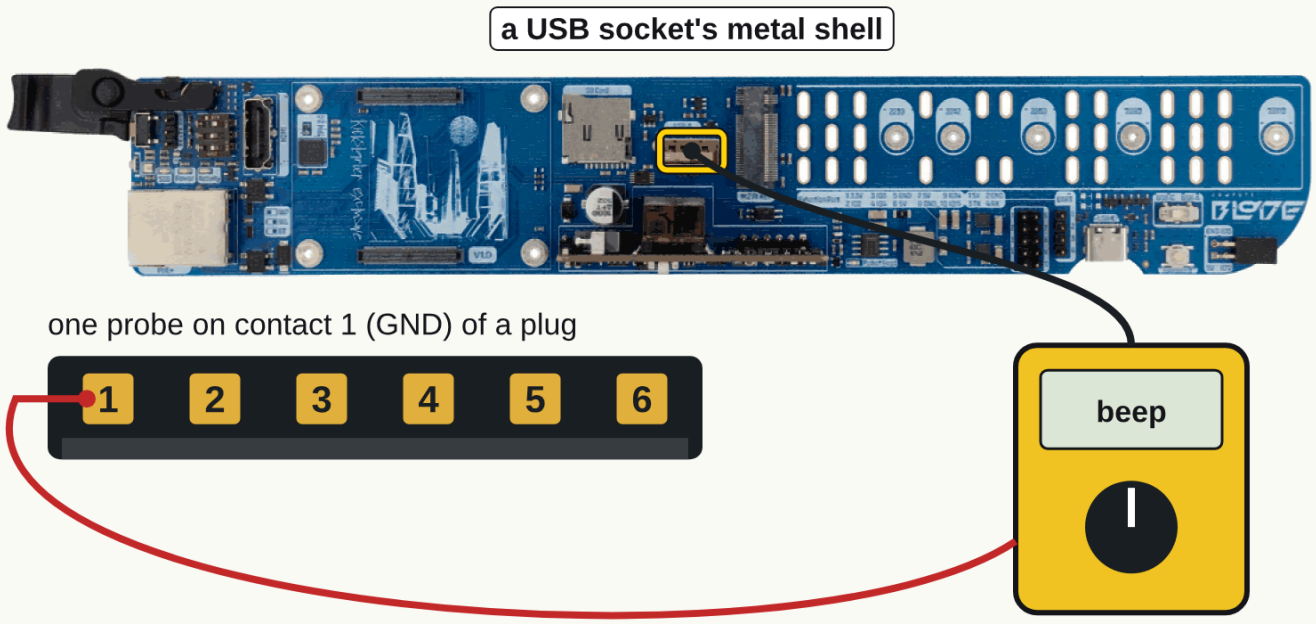


3. The Acorn is not in its slot and the plugs are free. Put one meter probe on contact 1 (GND) of a plug and the other on the metal shell of a USB socket on the Compute Blade: it must beep. Do the same for the other plug. Then contact 6 of each plug (VCC, the wire you cut back): against the shell and against every other contact it must be silent. That the shell is the host’s ground is taken as given, not measured by us: contact 1’s beep shows only that contact 1 and the shell are joined. The beep is also meant to show that the P1 housing is the right way round: turned round, its GND wire would sit on IO3, not a ground pin; that a turned housing would then stay silent is not tried by us. The beep is also meant to show that the P2 housing is the right way round: turned round, its GND wire would sit on TX, not a ground pin; that a turned housing would then stay silent is not tried by us.

Bench check on a Compute Blade, before power

housings on their headers, plugs free, card out of its slot

If the Compute Blade is on, ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in.



Contact 1 of each plug to the shell: the meter must beep.

Contact 6 of each plug (VCC, cut back): silent to the shell and to every other contact.

The plug and the meter are sketched. The shell being ground is not measured on this host.

If it fails

Do not fit the cables. A contact 1 that does not beep means that cable’s ground wire is open or in the wrong cavity; a contact 6 that beeps anywhere means the 3.3 V wire was not the one cut back. Go back to that cable’s second page, “JTAG connector 2” or “UART connector 2” (fill and check the housing), and check every wire again.

Compute Blade cables: fitting

Your two cables for an Acorn on a Compute Blade have passed the bench check, and you are fitting them and the card.

Not yet run by us on this hardware: written from the design. Photos: Uptime Lab (Compute Blade), RHS Research (LiteFury underside; the Acorn is the same PCB).

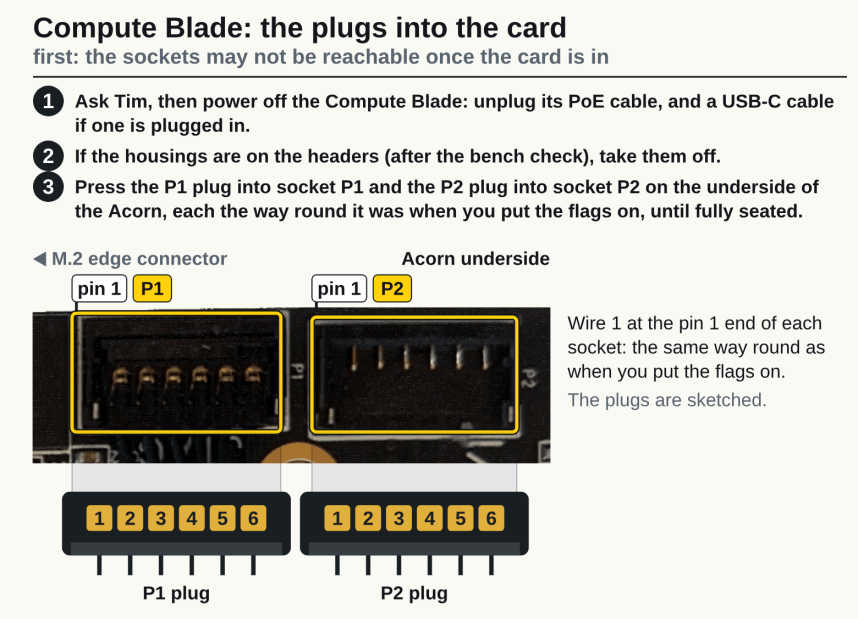
What you need

Both cables, checked on the bench (the page before this one), the Acorn and the Compute Blade. As before: after action 1 (step 1 below), touch bare metal of the host before you pick up the card, and hold it by its edges.

Steps

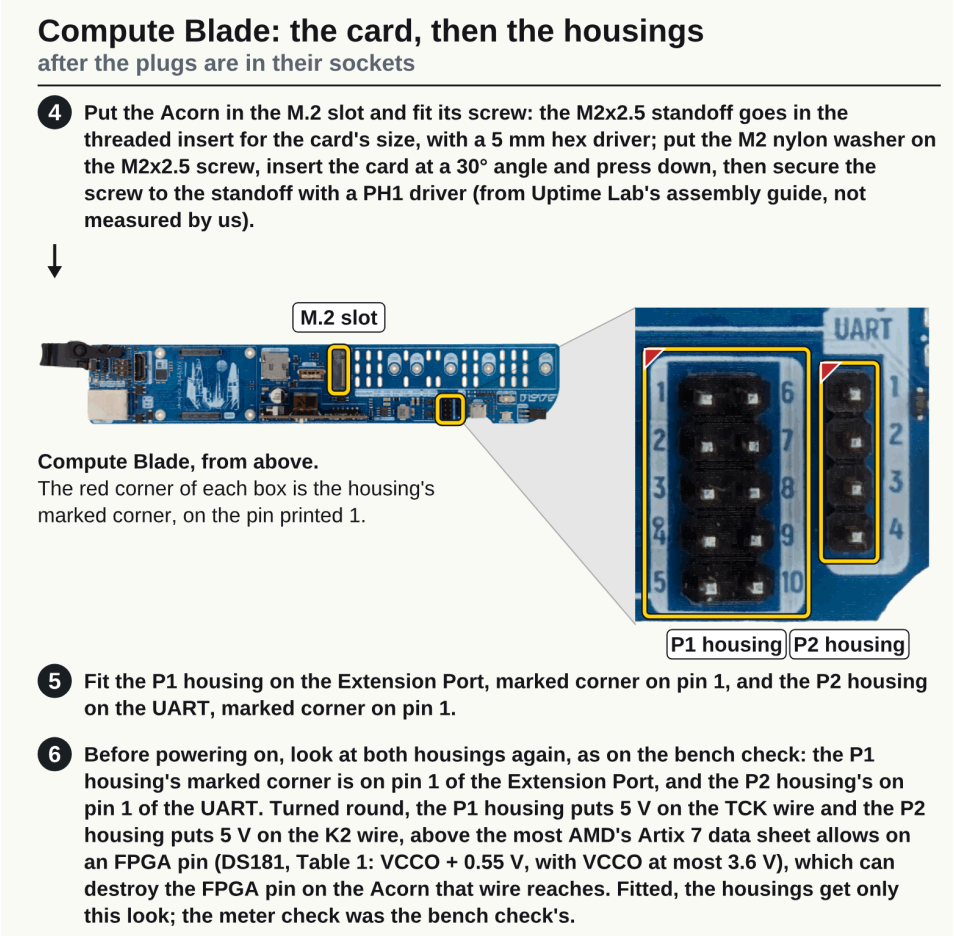
1. Fit the plugs into the card. The sockets are on the underside of the card and may not be reachable once it is in the slot, so the plugs go in first, in this order.

1. Ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in.
2. If the housings are on the headers (after the bench check), take them off.
3. Press the P1 plug into socket P1 and the P2 plug into socket P2 on the underside of the Acorn, each the way round it was when you put the flags on, until fully seated.



2. Fit the card, then the housings, in this order.

4. Put the Acorn in the M.2 slot and fit its screw: the M2x2.5 standoff goes in the threaded insert for the card’s size, with a 5 mm hex driver; put the M2 nylon washer on the M2x2.5 screw, insert the card at a 30° angle and press down, then secure the screw to the standoff with a PH1 driver (from Uptime Lab’s assembly guide, not measured by us).
5. Fit the P1 housing on the Extension Port, marked corner on pin 1, and the P2 housing on the UART, marked corner on pin 1.
6. Before powering on, look at both housings again, as on the bench check: the P1 housing’s marked corner is on pin 1 of the Extension Port, and the P2 housing’s on pin 1 of the UART. Turned round, the P1 housing puts 5 V on the TCK wire and the P2 housing puts 5 V on the K2 wire, above the most AMD’s Artix 7 data sheet allows on an FPGA pin (DS181, Table 1: VCCO + 0.55 V, with VCCO at most 3.6 V), which can destroy the FPGA pin on the Acorn that wire reaches. Fitted, the housings get only this look; the meter check was the bench check’s.



Next

Power the host on and run the check: the page “verifying 1”. At ps1: on pi16 and pi20 at ps1 only, not yet on the two Compute Module 4 blades.

Compute Blade cables: verifying 1, run the check and read the result

You have an Acorn on a Compute Blade, its two cables built and fitted, and want to know what the check on the blade says about the wiring. On a Compute Blade today it cannot yet prove the cables: the paragraph “What to expect on a Compute Blade today” below says why.

Log in to the blade first. At ps1:

From any machine with `ssh`, as the visitor. The port is `1NN22`, where `NN` is the number in the blade’s name: `11422` for pi14 at ps1, `11622` for pi16 at ps1, `11822` for pi18 at ps1, `12022` for pi20 at ps1.

```
$ ssh -p 11622 pi@ps1.fpgas.online
```

The password is in the banner that `ssh` prints when you connect, before it asks for one: the board pages of https://ps1.fpgas.online/fpgas/ say so under “Use your own ssh client” (read on 6 October 2026 on `pi3.html`, the page of pi3 at ps1, port `10322`). `sudo` asks for the same one. The blades have no board page of their own there. On 7 October 2026 this login was run as written on all four blades (pi14 at ps1, pi16 at ps1, pi18 at ps1 and pi20 at ps1), with the password from the banner.

What the check is

`fpgas-verify` is the program that checks an FPGA board from the machine it is attached to. For an Acorn its check doubles as a wiring test: each of its tests uses a known set of wires between the card and the blade, so which tests pass, and what a failing one says, point at the wire. An Acorn is sold as a CLE-215+, a CLE-215 or a CLE-101, and the check’s summary names the variant it found (`acorn cle-101`, for example); a CLE-215 has not been checked by us.

The check never writes the card’s flash and never loads a design into the FPGA. It drives the P1 and P2 wires, which is how it tests them, and puts the host’s pins back as it found them.

What to expect on a Compute Blade today. `pcie-link` passes when the card is seated, on SQRL’s factory image or on the vendor’s XDMA sample image (version 0.0.post1220 or newer). On a Compute Module 5 `jtag` cannot run in a boot that has the header’s serial port on (kernel 6.18): the serial port holds GPIO14, which is also the JTAG TMS wire, so `jtag` fails there whatever the wiring. With the port off at boot (a change to the gateway’s boot files, on the page “verifying 3”), `jtag` passes (it reads the IDCODE and device DNA): run on pi20 at ps1 on 7 October 2026. On a Compute Module 4 it has not been run by us under this kernel (on pi14 at ps1 on 20 September 2026, under kernel 6.12, JTAG got no response, TCK floating). Every other test but `rp1-pio` is `not run` until the card is converted to the fpgas.online design, and converting a card on a blade is not in this guide yet: do not load a design into the card or convert it (the page “verifying 3” says why). So the result today is `fail` even with perfect cables: it shows that the card is seated and its PCIe link is up, and it cannot yet show that the two cables are right. The bench check with a meter, before fitting, is what the cables rest on until then.

Install it and run it

The Compute Blades at ps1 boot from the network with their root file system in memory (`overlayroot=tmpfs`): what you install is gone at the next boot, and so is the check that would run at boot. So after each boot, install and run by hand. **Ask Tim before you reboot a blade:** a reboot ends a visitor's session on it, the same harm as a power-off.

Not on pi14 at ps1 or pi18 at ps1 yet. Those two carry a Compute Module 4; the check has been run as written only on the two Compute Module 5 blades (pi16 and pi20 at ps1, 7 October 2026), and `vcgencmd` hung for good on both CM4 blades that day. Until the check has been tried on a CM4 blade, run it on pi16 at ps1 and pi20 at ps1 only.

The check runs `jtag` too, whose TMS wire is GPIO14. In a boot with the header's serial port on, `jtag` fails without running the JTAG tool (the result printed below). In a boot with the header's serial port off (the page "verifying 3"), run the check only after steps 2 and 3 of the list "Before JTAG runs on a blade" on that page pass (check that GPIO14 is free; check that nothing on the card drives GPIO14). If either fails, do not run the check in that boot: something holds or drives the TMS wire.

A line below is too long for the sheet: where a line starts with ↪, it continues the line above. Type the two as one line, leaving out the arrow and the one space after it.

```
# 1. The two apt repositories the packages come from.
sudo install -d -m0755 /etc/apt/keyrings
curl -fsSL https://apt.fpgas.online/apt.gpg | sudo tee /etc/apt/keyrings/apt.gpg > /dev/null
echo "deb [signed-by=/etc/apt/keyrings/apt.gpg] https://apt.fpgas.online/$(. /etc/os-release; echo $VERSION_CODENAME)/ ./" \
  | sudo tee /etc/apt/sources.list.d/apt.list
curl -fsSL https://fpgas.online/fpgas.online-fpga-tools/fpgas.online-fpga-tools.gpg \
  | sudo tee /etc/apt/keyrings/fpgas.online-fpga-tools.gpg > /dev/null
echo "deb [signed-by=/etc/apt/keyrings/fpgas.online-fpga-tools.gpg] https://fpgas.online/fpgas.online-fpga-tools/$(. /etc/os-
↪ release; echo $VERSION_CODENAME)/ ./" \
  | sudo tee /etc/apt/sources.list.d/fpgas.online-fpga-tools.list

# 2. The Acorn's packages.
sudo apt update
sudo apt install fpgas-online-acorn

# 3. What is installed, and which board this host is set up to check.
fpgas-verify --list

# 4. The check. Nothing is sent anywhere unless a file on the host says `publish = on`; --no-publish makes sure.
sudo fpgas-acorn-verify --no-publish
```

`fpgas-verify` checks whichever board this host is set up for, as the check at boot does; `fpgas-acorn-verify` checks the Acorn whatever the host is set up for. On a host set up for an Acorn the two print the same.

Read the result

There is one result, **pass** or **fail**, and only a pass exits 0. The summary on the terminal lists every test in the order it ran with its result; for a check that did not pass it ends with `RESULT:`, a `failed:` line for each failed test, a `not run:` line for the tests that did not run and why, and `What to do:`.

No Compute Blade has passed the whole check yet. This is what one prints today: a Compute Blade with a Compute Module 5 and an Acorn CLE-101 still on the image it was sold with (pi16 at ps1, 7 October 2026, installed by the steps above, version 0.0.post1216). Two things are wrong and neither is the wiring or the installation: the card has not been converted to the fpgas.online design, and in this boot the JTAG test cannot have its TMS pin, which the header's serial port holds. Its `What to do:` lines are left out here: that version's advice was for a Raspberry Pi 5. **On a Compute Blade, do not load a**

design into the card or convert it; that is not in this guide yet (the page “verifying 3” says why). From version 0.0.post1284 the check says so itself, and says that changing the gateway’s shared boot files or giving one blade its own copy is the gateway owner’s choice.

A line below is too long for the sheet: where a line starts with ↪, it continues the line above. Type the two as one line, leaving out the arrow and the one space after it.

```
$ sudo fpgas-acorn-verify --no-publish

********************************************************************************
*** FPGA VERIFY: FAIL ********************************************************
fpgas-verify: fail (mode acorn, configured: acorn)
  not published: --no-publish
  acorn cle-101: fail
    unconverted: runs SQRL's factory image, not the fpgas.online design
    pcie-link  pass
    rp1-pio    pass
    jtag       fail: P1 JTAG could not be probed: GPIO14 (TMS) is held by 1f00030000.serial (uart0): the kernel does not hand
↪  out a pin that is held, so the JTAG chain cannot be scanned
    pcie-bar0  not run: unconverted: runs SQRL's factory image, not the fpgas.online design
    flash      not run: unconverted: runs SQRL's factory image, not the fpgas.online design
    ddr        not run: unconverted: runs SQRL's factory image, not the fpgas.online design
    p2-uart    not run: the board does not run a known build
    p2-serial  not run: unconverted: runs SQRL's factory image, not the fpgas.online design
    scratch    not run: unconverted: runs SQRL's factory image, not the fpgas.online design
    p2-gpio    not run: J5 and H5 are not wired on the Compute Blade setup
  state recorded (first run) in /var/lib/fpgas-online/verify-state.json

RESULT: FAIL: a board did not pass.
  acorn cle-101: fail (2 tests passed, 1 failed, 7 not run)
    fault: unconverted: runs SQRL's factory image, not the fpgas.online design
    failed: jtag: P1 JTAG could not be probed: GPIO14 (TMS) is held by 1f00030000.serial (uart0): the kernel does not hand ou
↪ t a pin that is held, so the JTAG chain cannot be scanned
    not run: pcie-bar0, flash, ddr, p2-serial, scratch: unconverted: runs SQRL's factory image, not the fpgas.online design
    not run: p2-uart: the board does not run a known build
    not run: p2-gpio: J5 and H5 are not wired on the Compute Blade setup
What to do: (left out here; see the text above)
```

On the blades at ps1 every `sudo` first prints `sudo: unable to resolve host pi16: Name or service not known` (with the blade’s own name). It did no harm on the blades it was seen on (7 October 2026), and is left out above.

A card still on the vendor’s XDMA sample image (pi20 at ps1, 7 October 2026, version 0.0.post1220) gets the same tests as one on SQRL’s image: `pcie-link` and `rp1-pio` pass, `jtag` fails on GPIO14 as above, the rest are `not run`. Its summary line reads `acorn -: fail` (no variant is named), and the first `What to do:` line says the board runs Xilinx’s XDMA sample design (on a Compute Blade, from version 0.0.post1284, not to convert it). That was on pi20 at ps1, installed by the steps above as version 0.0.post1216 and updated to 0.0.post1220 the same day; version 0.0.post1216 printed `acorn: fail (no test ran)` there (fpgas.online-test-designs issue 155, a card on the vendor XDMA sample image got no test at all).

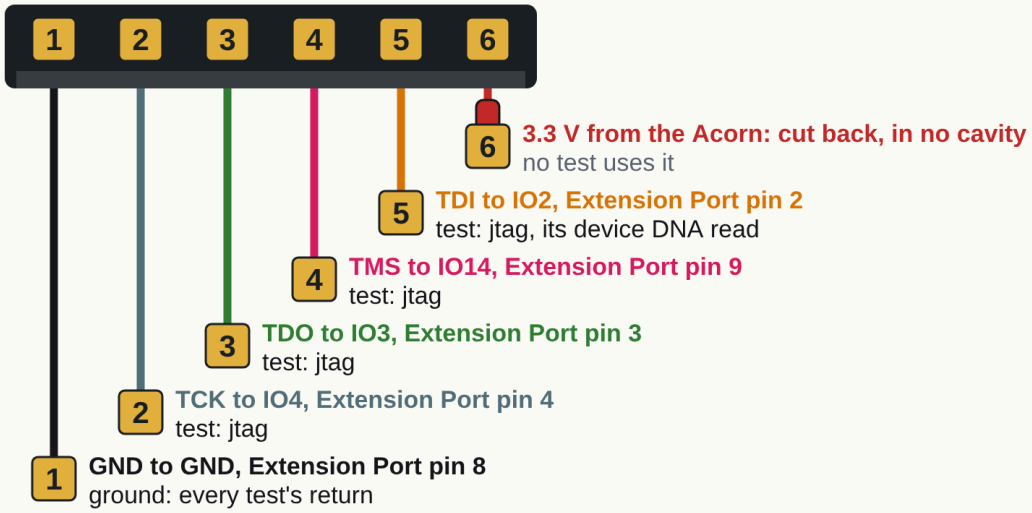
A pass will list every test with `pass` and end there, with no `RESULT:` part; on a Compute Blade `p2-gpio` stays `not run`, because J5 and H5 are not wired.

Which test uses which wire

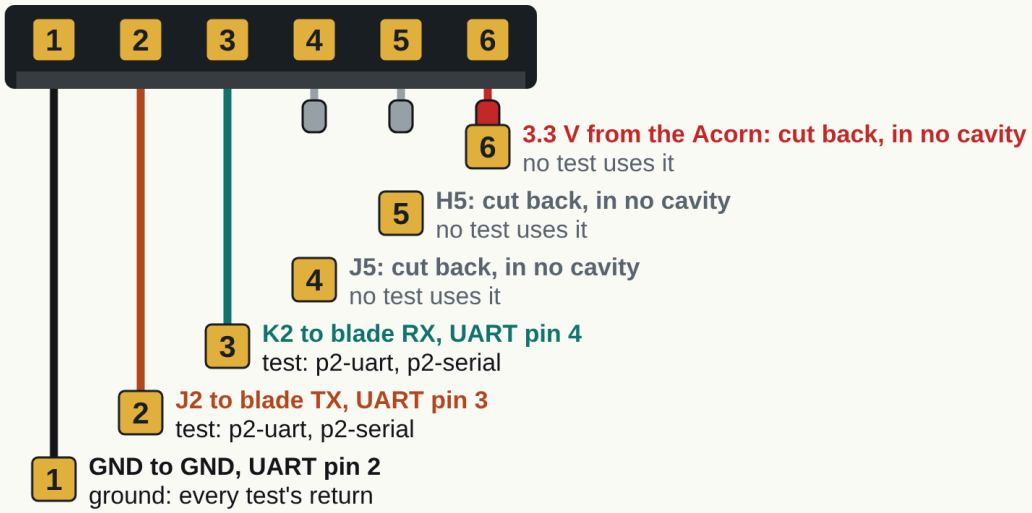
Which test uses which wire: Acorn on a Compute Blade

the wire numbers are the flags you put on

P1 cable (JTAG)



P2 cable (I/O)



pcie-link and pcie-bar0 go through the M.2 slot; rp1-pio, flash and ddr use no wire of the cables; scratch uses the serial pair as well.

p2-gpio is never run on a Compute Blade: J5 and H5 are not wired.

Wire colours are for this picture only: the real wires are all black. Count from the plug's pin 1 end.

The plugs are sketched, not from a photograph.

Test	Wires it uses	A pass shows	Runs on a card still on SQRL's image
`pcie-link`	the M.2 slot	the card is seated and the PCIe link is at the setup's speed and width	yes
`jtag`	P1: TCK, TMS, TDO for the IDCODE read; TDI as well for the device DNA read	all four JTAG wires, and that the FPGA is the variant's part	yes
`pcie-bar0`	the M.2 slot	the fpgas.online design is running and answers over PCIe	no: the board gets `fault: unconverted: …`; this test, `p2-uart`, `p2-serial`, `p2-gpio`, `flash`, `ddr` and `scratch` are listed as `not run` (`pcie-link`, `jtag` and `rp1-pio` still run)
`p2-uart`	P2: K2 (FPGA transmit) to the Pi's RXD (GPIO15), J2 (FPGA receive) from the Pi's TXD (GPIO14)	the serial pair, in the right direction	no
`p2-serial`	the same two wires, driven and read as plain pins in both directions	each of J2 and K2 on its own, so a crossed pair or one open wire is told apart	no
`p2-gpio`	none: J5 and H5 are not wired on a Compute Blade	nothing: it is listed as `not run`	never runs
`rp1-pio`	no wire: `/dev/pio0` on a Pi 5 or CM5 (the test passed on a Pi 5 at welland, 2 October 2026, and on pi16 and pi20 at ps1, 7 October 2026)	nothing about the wiring (not run on other hosts)	yes
`flash`, `ddr`, `scratch`	no wire of the cable, except that `scratch` also goes over the serial pair	nothing about the wiring	no

So on a card that has not been converted yet, `pcie-link` and `jtag` are the wiring tests; the P2 wires can only be tested once the card runs the fpgas.online design (converting a card; on a Compute Blade that is not in this guide yet: do not).

Compute Blade cables: verifying 2, when a test fails

The check of your Acorn on a Compute Blade printed a failing line, and you want to know which wire it means.

From a failing line to the wire

Find the failing line in the table, then the wire in the two cavity pictures under it: the number in a cavity is the number on the wire's flag.

Before you touch a cable: Ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. After moving a wire, the cable goes through the same checks as a new one before any boot: take the card out and pull both plugs from its sockets, check that cable's plug contacts against its housing with the meter as steps 3 and 4 of "JTAG connector 2" or "UART connector 2" do, run "the bench check" with the card out, and fit the cables as "Fitting" does. Then ask Tim and boot, and run the check again (on a blade at ps1 the install is gone after the boot: install again, as on verifying 1; and not on pi14 or pi18 at ps1 yet, as verifying 1 says). One test can be run on its own, `sudo fpgas-acorn-verify --no-publish --test p2-serial` or `sudo fpgas-acorn-verify --no-publish --test jtag`; it prints the usual summary, then the whole report as JSON. Run `--test jtag` only as the page "verifying 3" says: in a boot with the header's serial port off, and only after steps 2 and 3 of the list "Before JTAG runs on a blade" on the page "verifying 3" pass (check that GPIO14 is free; check that nothing on the card drives GPIO14). In a boot with the header's serial port off, the same goes for the whole check, which runs `jtag` too (the page "verifying 1").

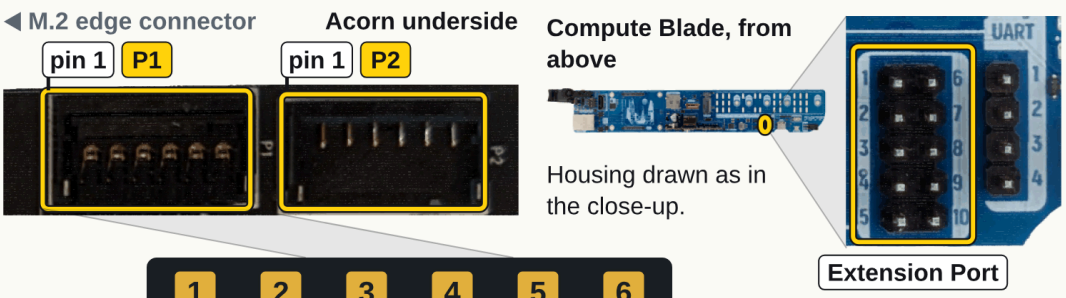
The failing line	Look at
`pcie-link fail: link is x2, expected x1` (or a speed)	the M.2 seat; or the setup's expected figures are not this host's
`jtag fail: no device on the P1 JTAG chain`	the P1 cable is not plugged in, or TCK, TMS or TDO is open or on the wrong pin
`jtag fail: P1 JTAG chain has …, expected one …`	another device answers, or the card is not the variant the host expects
`jtag fail: device DNA over P1 JTAG reads 0x0: the DNA port is not being read` (or `reads 0x1fffffffffffff`), `openFPGALoader --read-dna read no device DNA over P1 JTAG`, or `device DNA over JTAG … is not the one over BAR0 …`	TDI: the IDCODE read worked without it
`jtag fail: … GPIO14 (TMS) is held by … (uart0): the kernel does not hand out a pin that is held …`	not a wire: the serial port holds GPIO14, which is also TMS, so JTAG cannot run in a boot that has the header's serial port on (kernel 6.18). The page "what has been run on a Compute Blade" says what has been tried

The failing line	Look at
`p2-uart fail: no UARTBone reply on /dev/ttyAMA0 (P2 K2/J2)`	the serial pair: open, or crossed; `p2-serial` says which
`p2-serial fail: J2 -> GPIO14: the FPGA drove 1, the Pi read 0; K2 -> GPIO15: the FPGA drove 0, the Pi read 1; …` with the `01` and `10` lines swapped and `00` and `11` right	J2 and K2 are **crossed**: wires 2 and 3 of the P2 cable are in each other's cavity. Take both terminals out of the housing and put each in the other's cavity (the P2 cavity picture below). The 470 Ω resistor stays in wire 2 (J2)
`p2-serial` naming one of its two signals only (J2 or K2)	that one wire is **open**, or on the wrong pin. Before the FPGA drives, the test sets the host's pull against the level to come, so an open wire reads the opposite of what was driven. When you then check the J2 wire with a meter, set it to ohms: a good J2 wire reads close to 470 Ω end to end, and a continuity buzzer usually stays silent through that

The two cavity pictures are the ones the cables were built from, shown again to find a wire's cavity. Their notes about cutting, marking and the meter check belong to building the cables.

P1 cable (JTAG) for a Compute Blade:

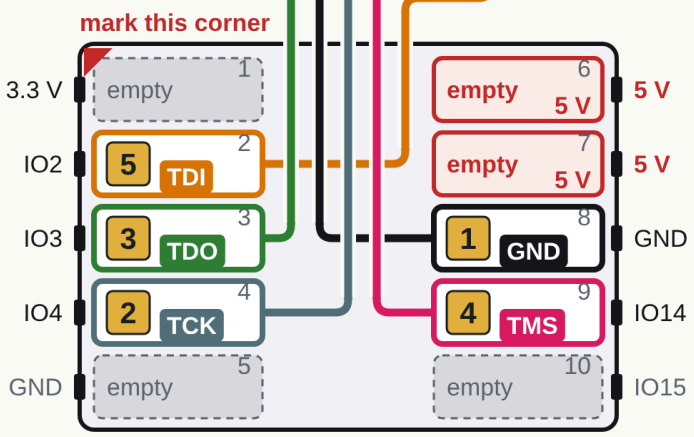
which wire goes in which cavity of the 2×5 Dupont housing



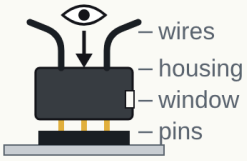


The plug, seen from above as it sits in the socket.
Acorn turned underside up, M.2 edge to your left.

wire 6: cut off about 10 mm from the plug, heat shrink over the end
3.3 V from the Acorn: it must never reach the host.



The two columns are drawn apart to let the wires through.



You are looking down on the wires.

2×5 Dupont housing, seen from the wire side, as it will sit on the Extension Port with the Compute Blade seen from above.

Wire colours are for this picture only: the real wires are all black. Count from the plug's pin 1 end.

n wire number: its place in the plug n pin number, printed beside the header

a small window in the side of the housing: a terminal's latch will face it

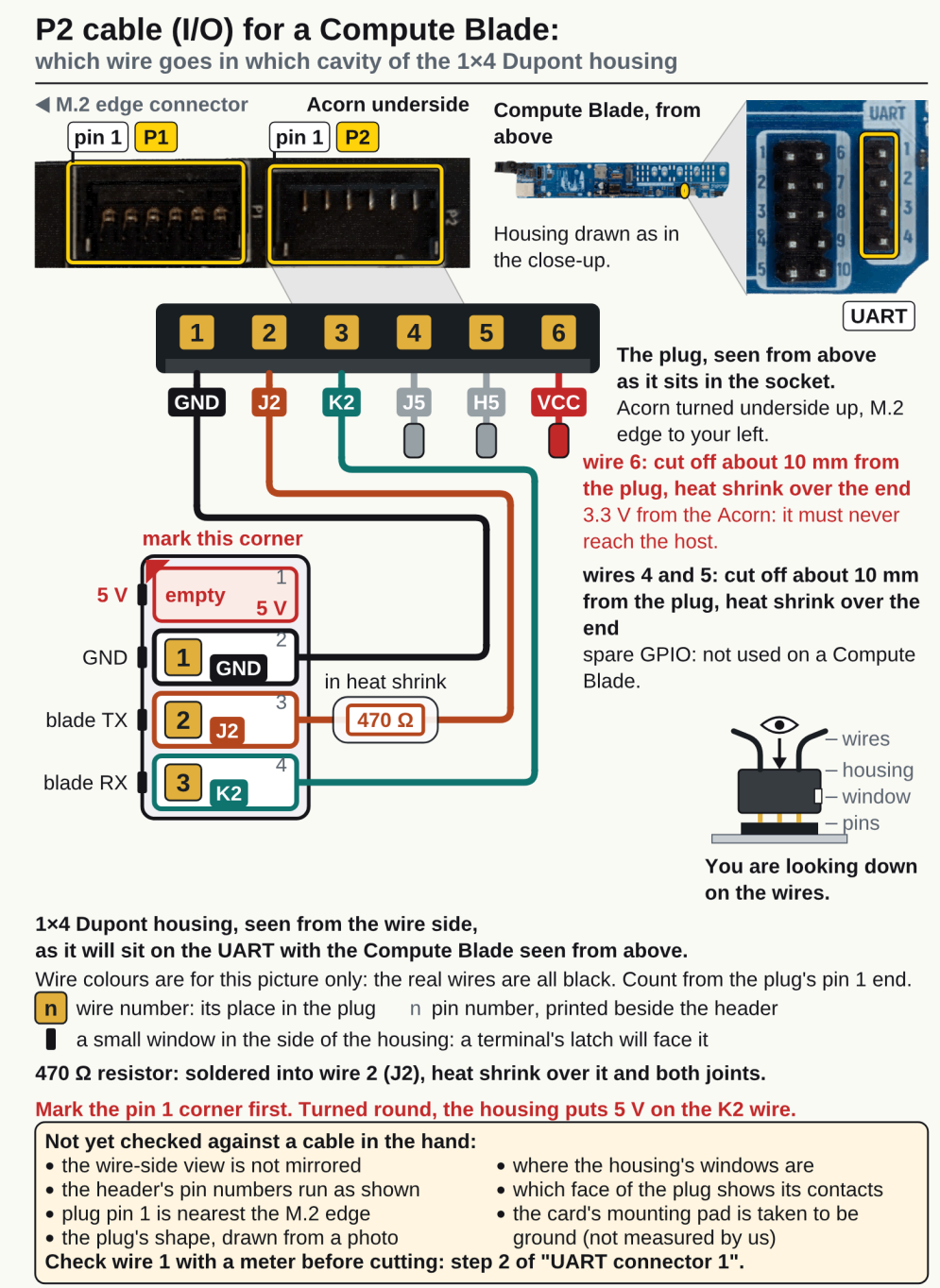
Mark the pin 1 corner first. Turned round, the housing puts 5 V on the TCK wire.

Not yet checked against a cable in the hand:

- the wire-side view is not mirrored
- the header's pin numbers run as shown
- plug pin 1 is nearest the M.2 edge
- the plug's shape, drawn from a photo
- where the housing's windows are
- which face of the plug shows its contacts
- the card's mounting pad is taken to be ground (not measured by us)

Check wire 1 with a meter before cutting: step 4 of "JTAG connector 1".

Compute Blade cables: verifying 2, “From a failing line to the wire”, “the two cavity pictures are the ones the cables were built from”, continued from the sheet before



`p2-serial` prints what was driven and what was read, eight lines for two wires. The two digits are the two signals: the right-hand digit is J2, the left-hand one K2.

A correctly wired pair reads back what was driven: `FPGA drives 01: Pi reads GPIO14=1 GPIO15=0`, `Pi drives 01: FPGA reads 01`, and so on for every pattern.

A failing line that is not in the table above is not about a wire of the cables: the page “verifying 2b” has every other message the check gives about an Acorn.

Compute Blade cables: verifying 2b, when the failure is not a wire

The check of your Acorn on a Compute Blade printed a line that is not about one of the cables' wires (the card's image, its memory, the tool itself), and you want to know what it means.

Every other message about an Acorn

The check's own words, from the tool's list of common failures, which has the other boards' too. A wire of the cables is behind the `jtag` and `p2-…` lines only; for those, the page "verifying 2" goes from the line to the wire.

It says	Meaning, and what to do
`fail: unconverted: …`	an Acorn on SQRL's factory image (or the XDMA sample). **Do not** load a design into the card or convert it; that is not in the guide yet. The only attempt (pi20 at ps1, 7 October 2026) lost the card's PCIe endpoint (a bus rescan did not bring it back, and a root-complex re-probe failed), and the reboot after it was followed by about two hours of restarts, cause not known
`fail: … is not a design we built`	a Xilinx PCIe design the Acorn check does not know; its flash is not read
`fail: running the golden image`	the Acorn's operational slot did not boot; it fell back to golden
`fail: link is x2, expected x1`	the Acorn's PCIe link is not the setup's (`expected.toml`)
`fail: no device on the P1 JTAG chain` / `no UARTBone reply on /dev/ttyAMA0`	an Acorn's JTAG or P2 UART cable is off or miswired
`fail: P1 JTAG could not be probed: GPIO14 (TMS) is held by 1f00030000.serial (uart0): the kernel does not hand out a pin that is held…`	an Acorn's JTAG pin is in use by another driver, so the scan was not tried. Seen on a Compute Blade with a CM5 (kernel 6.18): TMS and the serial port's TX are both GPIO14, and with the serial port on the kernel does not lend the pin. JTAG cannot be tested there while the serial port is on. With the header's serial port off at boot (in `config.txt` `enable_uart=0` and no `uart_2ndstage=1`, in `cmdline.txt` no `console=serial0`; on a netbooted blade these files are on the gateway, where they may be shared by several hosts: change the shared files, which changes every host that boots from them, or give the one blade its own copy; which is the gateway owner's choice), the pin was free and the JTAG test passed on one Compute Blade with a CM5 (pi20 at ps1, 7 October 2026; fpgas.online-test-designs issue 127: on a Compute Blade the JTAG test cannot have GPIO14 while the serial port holds it)

It says	Meaning, and what to do
`fail: … gpiod_line_request_set_values_subset: Assertion 'request' failed`	openFPGALoader could not have one of the JTAG pins, because a driver holds it. Seen on a Compute Blade with a CM5 (kernel 6.18), where the serial port holds GPIO14, which is also JTAG TMS: the JTAG test cannot run there while the serial port is on; with it off at boot the JTAG test passed on one CM5 blade (pi20 at ps1, 7 October 2026; fpgas.online-test-designs issue 127: on a Compute Blade the JTAG test cannot have GPIO14 while the serial port holds it)
`fail: openFPGALoader printed no raw IDCODE scan (needs --verbose-level 2 output)`	the tool's version, not the board: openFPGALoader exited 0 but printed no `- 0 -> 0x...` lines at `--verbose-level 2`, so it is older than v0.9.0. Its last lines are in the report's `output`
`fail: … failed (exit N) before scanning the JTAG chain: …`	the scan tool exited with an error and printed no scan: the cable or gpiochip would not open, say. The reason ends with its last line of output; more is in the report's `output`
`fail: device DNA over P1 JTAG reads 0x…: the DNA port is not being read`	the DNA read over JTAG is all zeros or all ones, which is no chip's DNA. The IDCODE read worked without TDI, so check the P1 TDI wire first. Ask Tim, then power off the Compute Blade: unplug its PoE cable, and a USB-C cable if one is plugged in. Check the TDI wire with the card out and both plugs pulled, as steps 3 and 4 of "JTAG connector 2" do; fit the cables again as "Fitting" does, and run only the check's `jtag` test (`sudo fpgas-acorn-verify --no-publish --test jtag`), in a boot with the header's serial port off, and only after steps 2 and 3 of the list "Before JTAG runs on a blade" on the page "verifying 3" pass (check that GPIO14 is free; check that nothing on the card drives GPIO14); not `openFPGALoader` by hand
`fail: device DNA over JTAG … is not the one over BAR0`	the P1 TDI wire does not carry, or the DNA readout is wrong. Only a good BAR0 DNA is compared: one of all zeros or all ones is pcie-bar0's own fault (`device DNA over BAR0 reads 0x0: the DNA port is not being read`)
`fail: K2 -> GPIO15: …` / `GPIO14 -> J2: …`	a P2 serial wire is cut or miswired
`fail: DRAM write … MB/s, below the … MB/s expected` / `… words wrong in the … half`	the DRAM is slow or broken; `selftest.py` shows more

It says	Meaning, and what to do
`error: this host (…) is not an Acorn setup in wiring.toml`	an Acorn on a host neither setup has: add the host to `wiring.toml` if it is a real setup
`error: … does not match its manifest` / `manifest.json is missing`	`sudo apt install --reinstall fpgas-online-<board>-bitstreams`
`changed`	the board or its flash differs from the recorded state. Meant it? Run the check again with `--update` to record what it finds. Run `sudo fpgas-verify --no-publish --update`. It runs the whole check, `jtag` too: in a boot with the header's serial port off, run it only after steps 2 and 3 of the list "Before JTAG runs on a blade" on the page "verifying 3" pass (check that GPIO14 is free; check that nothing on the card drives GPIO14)

Compute Blade cables: verifying 3, what has been run on a Compute Blade

Your Compute Blade's check fails at `jtag` although the wiring is right, or you want to know what to expect before you start: what has and has not been run on a blade, and the pin JTAG shares with the serial port.

What has been run on a Compute Blade

What has been run on a Compute Blade, and what has not, as of 7 October 2026:

	State
Installing the packages and running the check (Raspberry Pi OS trixie, CM5)	run at ps1, by the page's own steps (pi16 and pi20 at ps1, 7 October 2026, version 0.0.post1216; on pi20 at ps1 updated to 0.0.post1220 the same day, the version every later run there used): the failing run printed on the page "verifying 1" is the result from pi16 at ps1, with 0.0.post1216. An install lasts until the blade's next boot
`pcie-link`	run at ps1: passes (5.0 GT/s, x1); on a card on the vendor's XDMA sample image too, from version 0.0.post1220 (pi20 at ps1, 7 October 2026)
`jtag` with the serial port on, kernel 6.18	run at ps1: **cannot work**. TMS is GPIO14, which is also the serial port's TX; that kernel does not lend a pin a driver has, and the serial driver cannot be detached while the system runs (read on pi16 at ps1, 5 October 2026: the driver has no `unbind` file, and the port is a kernel console). From 0.0.post1111 the test fails saying so, without running the tool (fpgas.online-test-designs issue 127, the Acorn check on a Compute Blade cannot have GPIO14 while the serial port holds it). Seen on the two CM5 blades, where `gpioinfo` names the kernel as line 14's user; on the two CM4 blades `gpioinfo` names no user for line 14 although the pin is the serial port's TXD0 (7 October 2026): whether JTAG can have it there is **not yet run by us**
`jtag` with the serial port off	run at ps1 on a CM5 (pi20 at ps1, 7 October 2026, our test 4, in the boot from 14:25 with the two changes below): **passes**: the test reads the IDCODE, `0x13631093` (an XC7A100T: AMD's 7 Series configuration user guide, UG470, Table 1-1, gives `X3631093` for the 7A100T, X being the revision digit), and the device DNA, `0x0028e5c45e304854` (the same as read from this card on 20 September 2026). Test 4 loaded no design and wrote no flash; later in the same boot test 6 loaded our design into the FPGA's SRAM (below the table), and the flash was never written. **Not yet run on a CM4 blade**
`jtag` under kernel 6.12, serial port on	recorded as working on pi20 at ps1 on 20 September 2026 (`--pins 2:3:4:14`), before these packages existed; on pi14 at ps1, a CM4, the same attempt that day got no response (TCK floating). Not run with these packages
The `p2-uart` and `p2-serial` tests	**not yet run by us on this hardware**: they need a converted card

	State
Converting a card on a Compute Blade	**not in this guide yet; do not load a design into a card on a blade or convert it.** Begun once, on pi20 at ps1 on 7 October 2026 (our test 6), and stopped at its step 1b, before the flash write: the paragraph below the table. The written steps are for the Pi 5 setup
A Compute Blade that passes the whole check	**not yet seen**. With the serial port off, pi20 at ps1 (7 October 2026) passed `pcie-link`, `rp1-pio` and `jtag`; the other seven were not run, and the check failed only on `unconverted: runs the vendor XDMA sample image`
The `p2-serial` test on a blade whose J2 wire has no 470 Ω resistor (pi20 at ps1's present serial pair: J2 straight to GPIO14, K2 to GPIO15, read with the pin-ID design on 31 August 2026; on which header pins it sits is not recorded by us)	**not yet run by us on this hardware**. From the code: while it runs to its end or raises an error, the test never has both ends of a wire driving at once (the Pi's pins are made inputs before the FPGA drives, and the FPGA's outputs are switched off before the Pi drives), so it does not rely on the resistor. What the resistor guards against is a design that drives J2 while JTAG or the serial port drives GPIO14; the fpgas.online design leaves J2 an input except while the host has switched J2/K2 to GPIO mode and enabled J2's output, which is what this test does, with the Pi's GPIO14 an input at that moment

JTAG and the serial pair share GPIO14 on a Compute Blade (in the guide's wiring J2 reaches it through 470 Ω, so that JTAG wins if both drive it: by design, not measured by us). Under kernel 6.18 they cannot both be had from one boot: with the header's serial port on, the kernel keeps GPIO14 for it. **With the header's serial port off at boot, JTAG reads the FPGA's IDCODE and device DNA**: run on a Compute Module 5 blade, pi20 at ps1, on 7 October 2026 (our test 4).

Test 6, the one attempt to convert a card on a blade, and how it ended (pi20 at ps1, 7 October 2026, in the same boot; times are Adelaide time). At 15:23 the card's PCIe endpoint was removed and our design was loaded into the FPGA's SRAM over JTAG; the load worked. A bus rescan did not bring the endpoint back. At step 1b the root-complex re-probe (unbind and bind of the blade's PCIe controller) failed: the bind answered "No such device", and the PCIe root port was gone too. The run stopped there, and the blade was rebooted at 15:52. From then on it restarted by itself every 2 to 3 minutes until about 18:09 (one boot, from about 17:06, stayed up until we rebooted it at 17:20:12); the cause is not known, and no power cycle was run by us. At 17:19 an `openFPGALoader --reset` over JTAG made the FPGA reload the vendor's sample image from its flash, which was never written, and at 17:20:12 we rebooted the blade into the shared boot files; in the boot from about 18:09 the card was back as before (`10ee:7011`, 5 GT/s x1). So a bus rescan and a root-complex re-probe both failed to restore the endpoint after a load on a blade, and a reboot was followed by the restarts: **do not load a design into a card on a blade or convert it; that is not in this guide yet.** A flash write over JTAG has **not been run** on a blade.

On the two Compute Module 4 blades: on 20 September 2026 a JTAG attempt on pi14 at ps1 got no response (TCK floating). On 7 October 2026 pi14 and pi18 at ps1 had logins, pin reads (`pinctrl get`, `gpioinfo`), a pull test that set the Pi's pull-down and then its pull-up on GPIO2, GPIO3 and GPIO4 (TDI, TDO, TCK) with `pinctrl`, read each pin and put each back as it was (`pinctrl get` read the same before and after, on both blades), and `vcgencmd`, which hung there: **no check, and no JTAG since 20 September**.

The two changes, as run:

- in `config.txt`, in its `[all]` section: `enable_uart=1` becomes `enable_uart=0`, and the line `uart_2ndstage=1` (the firmware's own boot messages on the header's serial pins) is taken out;

- in `cmdline.txt` (one line): the word `console=serial0,115200` is taken out.

What that boot then looks like (read on pi20 at ps1, 7 October 2026): there is no `/dev/ttyAMA0`; `serial0` names `ttyAMA10`, the Compute Module 5's own debug serial port, and the login prompt goes there; it does not use the header's pins, which in that boot have no function (below). So **there is no login and no console on the header's serial pins in that boot**, and no program has to be stopped from using them. GPIO14 and GPIO15 are switched to no function, pulled down, with no user. The root file system is not changed: it stays the gateway's `/srv/nfs/rpi/trixie/root`.

These two files are not on the blade, and the change is not yours to make from the blade. At ps1 the netbooted hosts fetch them from the gateway: its TFTP root (`/srv/tftp/`) has one entry for each host, a symbolic link named by the last eight characters of the host's serial number (on the host: `tr -d '\0' < /proc/device-tree/serial-number`), and those links point at one directory, `/srv/nfs/rpi/trixie/boot/`. On 7 October 2026 twelve hosts at ps1 were pointed at it, not only the four blades: among them pi21 at ps1, a Tiny Tapeout host by fpgas.online-infra's inventory (not seen by us), and seven hosts we cannot name. So there are two ways, and the choice belongs to whoever runs the gateway (at ps1: Carl):

1. **Change the one shared directory.** Every host that boots from it, all twelve, loses its console and login on the header's serial pins, and the firmware's boot messages there, from its next boot. To undo it, put the two files back as they were (keep a copy of both before the change); the hosts go back at their next boot.
2. **Give one blade its own copy.** Copy the directory (any name), make the two changes in the copy, and point that blade's link in `/srv/tftp/` at the copy. Only that blade is changed, from its next boot. To undo it, point the link back at `/srv/nfs/rpi/trixie/boot` and, after asking Tim, boot the blade again. This is how pi20 at ps1 was tested on 7 October 2026: its link, `/srv/tftp/de59093d`, was pointed at a copy for the test, and the changes were made in the copy only. The link was put back at 17:19:48 and the copy was removed then, and the shared `config.txt` and `cmdline.txt` read the same as before it. Every boot of pi20 at ps1 since 17:20 has come from the shared directory; at 18:10 its header's serial port was on (`serial0` was `ttyAMA0`).

Which of the two to take is the gateway owner's choice, not this page's: the first changes every host that boots from the shared directory, the second only the one blade.

Before JTAG runs on a blade

Ask Tim, then boot the blade again (a reboot ends a visitor's session on it, the same harm as a power-off), and in that boot (the one with the serial port off):

1. Install the packages again, as on the page "verifying 1": the install is gone after the boot.
2. Check that GPIO14 is free, with the commands at the end of this page: it must show no consumer.
3. Check that nothing on the card drives GPIO14. Do this on each blade you run JTAG on (at ps1: pi16 and pi20 only, as the page "verifying 1" says). Always run this step: the 470 Ω resistor in J2's wire is designed to protect against a card that drives GPIO14, not yet measured by us (pi20 at ps1's present serial pair, read on 31 August 2026, has none). Note what `pinctrl get 14` prints, then:

   ```
   sudo pinctrl set 14 ip pd; pinctrl lev 14   # must print 0
   sudo pinctrl set 14 ip pu; pinctrl lev 14   # must print 1
   ```

   Then put the pull back as `pinctrl get 14` showed it: on pi20 at ps1 on 7 October 2026 it showed `no    pd` before and after, so `sudo pinctrl set 14 no pd`. Both reads were as above there.

4. Only if steps 2 and 3 pass, run `sudo fpgas-acorn-verify --no-publish --test jtag`. If either fails, do not run JTAG: something holds or drives the TMS wire.

With the serial port off, `/dev/ttyAMA0` is not there, so the `p2-uart`, `p2-serial` and `scratch` tests cannot pass in that boot; what a Compute Blade's check should count as its result in each of the two configurations is not settled.

To see who has the JTAG pins on a host, without running anything on the card:

A line below is too long for the sheet: where a line starts with ↪, it continues the line above. Type the two as one line, leaving out the arrow and the one space after it.

```
pinctrl get 2,3,4,14,15     # the function each pin is switched to
gpiodetect                  # the header's chip: `pinctrl-rp1` on a CM5, `pinctrl-bcm2711` on a CM4 (as read at ps1, 7 Octobe
↪ r 2026)
gpioinfo -c gpiochip0 | grep -E 'line +(2|3|4|14):'   # with that chip's name (gpiod 2; gpiod 1: `gpioinfo gpiochip0`)
# a line shown with a consumer (`consumer="kernel"`) or `[used]` is one the kernel will not hand out
```

Bootloader EEPROM on a Compute Module in a Compute Blade: does it need anything?

You have a Compute Module 4 or 5 in a Compute Blade (at ps1, each Acorn sits on one) **and want to know whether its bootloader needs anything before or after the rest of the work.** For the two Compute Module 5 Lite blades we read (pi16 at ps1 and pi20 at ps1), the answer is no; for the two Compute Module 4 blades at ps1 it is not known (below). A Raspberry Pi 5 has its own page; nothing there is for a blade.

> **WARNING**
>
> **There are no upgrade steps for a Compute Module in a Compute Blade, because we have not done one.** What this page gives you is how to tell whether your blade needs anything. Both Compute Module 5 Lite blades we read (pi16 at ps1 and pi20 at ps1) are fine as they are and need nothing.

Does this blade need anything?

Log in to the blade the way you normally do and run these two. They change nothing and need no root.

At ps1, do not run them on the two Compute Module 4 blades (pi14 at ps1 and pi18 at ps1). There, on 7 October 2026, `vcgencmd get_config enable_uart` hung in the firmware and did not return: it could not be stopped, and `timeout 10` in front of it did not help (it was still waiting 15 minutes later). A reboot is expected to clear it; not tried. Why it hangs is not known. So for those two blades the answer to this page's question is **not known**, and so are their labels, which wait on the same read (Labels). On a Compute Module 5 blade run them as written (they answered on both; `timeout 10` in front does no harm, and whether it would help there is not known).

```
$ timeout 10 vcgencmd bootloader_version
2025/12/08 19:29:54
version 2226a853bb9f5fd80392e3a4a89e457aeca88008 (release)
timestamp 1765222194
update-time 1774980226
capabilities 0x0000007f
$ timeout 10 vcgencmd bootloader_config
[all]
BOOT_UART=1
# Default BOOT_ORDER for provisioning
# SD -> NVMe -> USB -> Network
BOOT_ORDER=0xf2461
```

That is what one blade of ours (a Compute Module 5 Lite) printed while it was running from the network. The Compute Module 4 blades at ps1 did not answer `vcgencmd` (above).

This blade's BOOT_ORDER, read from the LAST digit

0xf2461: SD card, NVMe, USB, then the network (the 2), then round again.

f	2	4	6	1
start again: then	network: tried 4th	USB: tried 3rd	NVMe: tried 2nd	SD card: tried 1st

IMPORTANT

What to do

- Your blade starts from the network, and its `BOOT_ORDER` has a `2` in it (as `0xf2461` has; `2` is network boot in Raspberry Pi's `BOOT_ORDER` table): **do nothing.** Its bootloader is fine as it is.
- Your blade does not start from the network, or its `BOOT_ORDER` has no `2` in it: **do not try to change it from this page.** If the blade is part of fpgas.online, open an issue at fpgas-online/fpgas.online-docs with the two outputs above. If it is your own, this page cannot help you further: what the blade's maker and Raspberry Pi document is quoted, untested by us, under What the Compute Blade's maker documents.
- Your blade is meant to start from its own storage (an SSD or a card) and does: it is not broken, and nothing on this page applies to it.

A date in the first line that is older or newer than the one above is not a reason to act, and neither is "UPDATE AVAILABLE" from `rpi-eeprom-update`.

Why there are no upgrade steps

Raspberry Pi documents two ways to write a Compute Module's bootloader (Compute Module EEPROM bootloader):

- over USB from another computer with `rpiboot`, with `EEPROM_nWP` not pulled low. Raspberry Pi's bootloader page says this "is also the only option available for CM4 and CM4S". On a blade it needs parts that the blade's maker says only the Dev model of the blade has;
- "self-update" mode, from USB mass storage or network boot, which overrides the Compute Module default of the `rpi-eeprom-update` service being off. Raspberry Pi warns that it "does not update the bootloader atomically. If a power failure occurs during an EEPROM update, you could corrupt the EEPROM."

Neither has been tried by us on a blade. That is why a blade that works is left alone.

Bootloader EEPROM: upgrade and lock

Every Raspberry Pi since the Pi 4 keeps its bootloader and the bootloader's settings in a small flash chip on the board, the bootloader EEPROM. On a fleet Pi that chip is locked, so that nobody with root on the Pi can change how it boots; a new bootloader therefore cannot simply be installed. Which machine do you have?

- **A Raspberry Pi 5: check, upgrade and lock.** Is its bootloader the fleet's, and is it locked? (The check.) If not, six steps for someone with the Pi in hand (the upgrade).
- **A Compute Module 4 or 5 in a Compute Blade: does it need anything?** How to tell (the two blades we read need nothing). There are no upgrade steps for it.

The rest of this page is what has been run and measured, what is not known and what does not work.

What is behind these pages

What the Compute Blade's maker documents

> **WARNING**
> Untested by us. Do not follow these as steps.

Only the **Dev** model of the Compute Blade has the parts for the USB route. The maker says so where it describes writing an operating system to a module's eMMC, in a guide written for the Compute Module 4 (image guide); the same port and button are what a bootloader flash over USB needs:

> eMMC can only be imaged on Dev blade as it has USB Type-C port, USB switch, and nRPIBOOT button

and, for putting the module into USB boot:

> From the usbboot directory run `sudo ./rpiboot`
>
> Move the USB switch to the USB Type-C position. Then, while holding down the nRPIBOOT button on the blade connect the USB Type-C cable.
>
> The Device will reconnect several times. On the last time it will appear as a USB media device.

The USB switch is there because "The compute module can only operate one USB port at a time." The maker's usbboot guide covers building `rpiboot` and says the configuration is edited in `/usbboot/<firmware directory>/config.txt`; for the flash itself it points to Raspberry Pi's "Flash Compute Module bootloader EEPROM", which is the procedure printed again below.

The write-protect pin is on a DIP switch, again on the Dev model only. From the maker's DIP switch guide:

> DIP Switch is ONLY populated on Dev model Compute Blades.
>
> Only change switch positions when the Blade is unplugged from power.

> To enable write protection, changes are required to `config.txt`. It can be found in the directory `/boot/firmware/`, or flash the EEPROM using usbboot adding the following line to the `config.txt` file `eeprom_write_protect=1` This will pull the `EEPROM_nWP` pin low. This will enable the DIP switch.

Read with Raspberry Pi's description (quoted below), that is the same two-part lock: `eeprom_write_protect=1` sets the flash's status register, and switch 1 is what holds `EEPROM_nWP` low. Raspberry Pi requires `EEPROM_nWP` not to be low while the bootloader is flashed, and the maker's table calls the left position "Disabled", so left is the position for an upgrade. That last step is our inference from the two documents, not something either states or we have tried.

LABELS FOR THE COMPUTE BLADES AT PS1

- Which label is which blade's. A label does not print its host's name, so tell the two apart by the "eth" line:
- eth 2c:cf:67:fb:91:e5 is the Compute Module of pi16 at ps1.
- eth 2c:cf:67:fd:1e:be is the Compute Module of pi20 at ps1.
- Both were made from reads of 5 Oct 2026, and checked against the modules on 7 Oct 2026 (every field matches). "HAT none" on them means the firmware reported no HAT; the header's pins were deliberately not scanned, because on a Compute Blade they are JTAG wires.

THE TWO SHEETS AFTER THIS ONE

- First: plain US Letter paper, labels at true size (63.5 x 38.1 mm) with a line to cut along.
- Second: laid out for Avery 5163 / 8163 stickers (4 x 2 inch, ten to the sheet), each label in the middle of its sticker. This layout has not been tried on a real sticker sheet: print it on plain paper first and hold it against the sticker sheet.
- Print both at 100 %, with no scaling.

LABELS THAT DO NOT EXIST YET

- The Acorn (FPGA) labels. An Acorn label carries the card's flash ID as well as its device DNA, and the flash ID is read over PCIe through the fpgas.online design, so each card's label waits for that card's conversion.
- Labels for pi14 at ps1 and pi18 at ps1: they did not answer on 5 Oct 2026 when these were made. Both answered on 7 Oct 2026, but the label read asks the firmware (vcgencmd), which hung for good on both that day: do not run it there; their labels wait until that is settled.

Plain US Letter: labels at true size (63.5 x 38.1 mm). Print at 100 %, no scaling; cut along the lines.

Raspberry Pi Compute Module 5 Lite

8 GB · Rev 1.0 · rev code d041a0

HAT none

serial
9fb8cfc7
cb291e63

eth MAC
2c:cf:67:fb:91:e5

wlan MAC
radio disabled, not readable

Raspberry Pi Compute Module 5 Lite

8 GB · Rev 1.0 · rev code d041a0

HAT none

serial
d85a19af
de59093d

eth MAC
2c:cf:67:fd:1e:be

wlan MAC
radio disabled, not readable

Avery 5163 / 8163 (4 x 2 in), grid from Avery's template. Not yet tried on real stock: print on plain paper first and hold it against the sheet. Print at 100 %, no scaling.

Raspberry Pi Compute Module 5 Lite

8 GB · Rev 1.0 · rev code d041a0

HAT none

serial
9fb8cfc7
cb291e63

eth MAC
2c:cf:67:fb:91:e5

wlan MAC
radio disabled, not readable

Raspberry Pi Compute Module 5 Lite

8 GB · Rev 1.0 · rev code d041a0

HAT none

serial
d85a19af
de59093d

eth MAC
2c:cf:67:fd:1e:be

wlan MAC
radio disabled, not readable