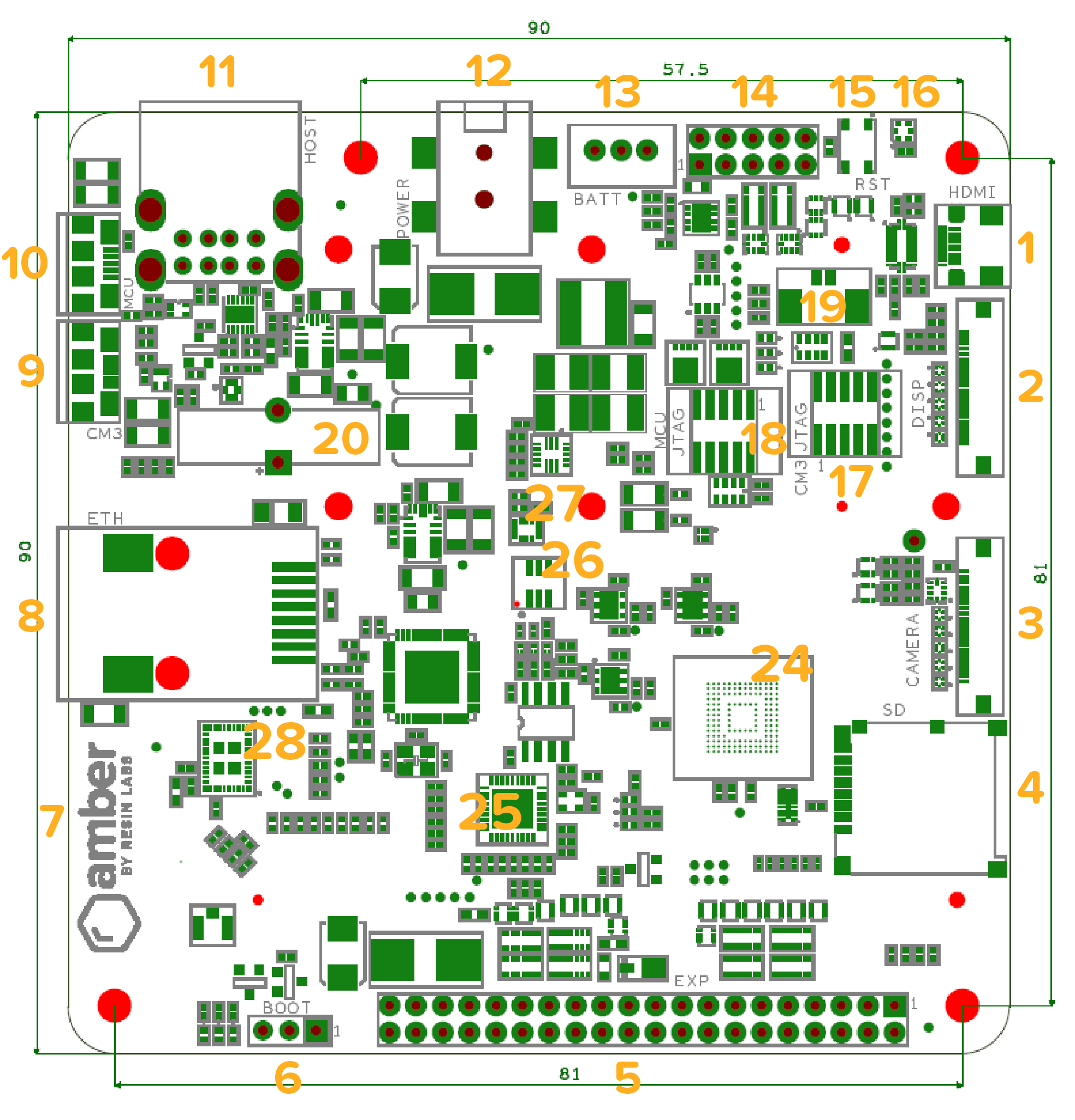
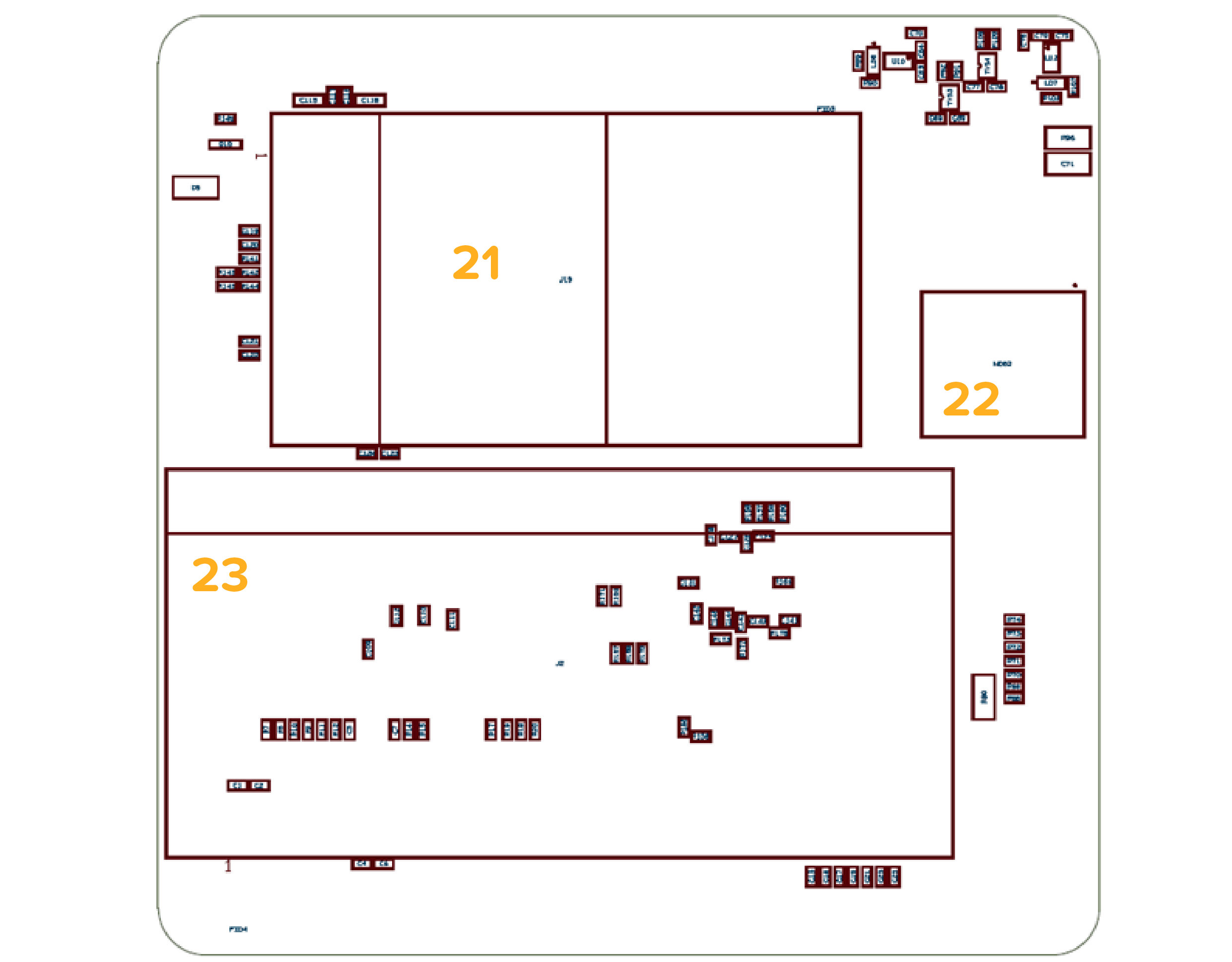


|  |  |
| --- | --- |
| **Document Type** | Prototype Internal Manual |
| **Project Codename:** | AMBER-PI-PLUS |
| **Version** | 0.9.0 |
| **Author** | Carlo Maria Curinga |
| **State (Draft/Proposed/Approved)** | Proposed |
| **Mechanical** | https://drive.google.com/open?id=0BwMoybaHFhYKZDV2TkJ1MXFBSms |
| **Schematics** | https://drive.google.com/open?id=0BwMoybaHFhYKS2dvWlRTcldHQWc |





|  |  |
| --- | --- |
| 1 | Micro HDMI port |
| 2 | Raspberry Pi Display connector.  *NOTE: This connector does not work in the current revision and will be replaced with a full-size, vertical one.* |
| 3 | Raspberry Pi Camera connector.  *NOTE: This connector does not work in the current revision and will be replaced with a full-size, vertical one.* |
| 4 | Micro SDcard boot socket.  *NOTE: This socket was meant to be a fallback boot option if the internal eMMC did not work. It is only for hardware debugging purposes and will be removed in the next revision* |
| 5 | Raspberry Pi HAT connector |
| 6 | USB SLAVE BOOT ENABLE selector via jumper. Jumper position close to aerial (7) means boot from eMMC, far from aerial (7) means expose eMMC as mass storage via CM3L debug port (9) |
| 7 | 2.4GHz aerial (WiFi and Bluetooth). This aerial will be replaced with a double one (one trace on top layer, one trace on bottom layer) covering the same footprint in order to support 2.4 and 5 GHz bands. In the current revision there is an unpopulated uFL socket for adding an external antenna bypassing the embedded one, in future revisions this will come pre-soldered (one for each band). |
| 8 | RJ45 10/100 Ethernet port. This port is connected to the Raspberry Pi Compute Module 3 aka CM3L (23) via its USB controller. |
| 9 | CM3L debug port. When USB SLAVE BOOT ENABLE switch (6) is in flash mode, it allows to plug Amber to a host computer and flash its eMMC as a mass storage device. |
| 10 | MCU debug port. This port exposes a serial port to the Artik020 microcontroller (22).  *NOTE: It does not include a JTAG controller so in order to use it for flashing firmware on the module there is the pre-requisite of the module already running a DFU-over-UART firmware. There is also a JTAG port for direct programming (18)* |
| 11 | Dual USB type-A female port. This port is connected to the Raspberry Pi Compute Module 3 aka CM3L (23) via its USB controller. |
| 12 | POWER IN port. This is a Barrel Jack 5.5/2.1 female port that accepts Voltage input between 6V and 18V. |
| 13 | JST-XH 2-cells LiPo battery connector. This port allows to power Amber from a 2-cells LiPo battery. There is an integrated charge/balance/discharge/switch circuitry that allows the battery to be charged from POWER IN (12). The circuitry is identical to the BeagleBone Blue.  *NOTE: in the current revision the connector is mounted on the wrong orientation. If you need to use it you need to desolder its 3 pins, rotate it 180 degrees on the Z axis and solder it back. The connector polarity fins must face the inside of the board as shown in the picture below:* |
| 14 | MCU I/O pins. This pins expose the Artik020 (22) free I/O. UART, SPI, Digital and Analog are exposed. |
| 15 | RST tactile switch button. This button is connected to the START pins of the CM3L (23). Circuit is closed by default so the device will start on power input, unless the button is pressed which will prevent the CM3L (23) boot until release. Performing this operation allows to hard-reset the CM3L (23). |
| 16 | HTS221 Barometric sensor. This sensor is wired to the Artik020 (22).  *NOTE: this component will be removed in the next revision* |
| 17 | CM3L JTAG. This port allows to program the CM3L (23) with a JTAG programmer. *NOTE: This is very unusual and will likely be removed in the next revision* |
| 18 | MCU JTAG. This port allows to program the Artik020 with a JTAG programmer. |
| 19 | RST external connector. This connector allows to wire an external button for moving outside the board the RST (15) function. |
| 20 | RTC supercondenser. This element allows the RTC on the Artik020 (22) to keep running when Amber is powered off.  *NOTE: this element will be changed with a standard i2c quartz-based RTC wired to the CM3L (23)* |
| 21 | Mini PCIe subset socket. This socket exposes as much protocols as possible that match the PCIe standard. USB, UART and I2C are supported, while PCI bus is not due to missing controller on the CM3L (23). USB based cards like SDcard readers, USB ports are supported. USB/Serial based cards like RS232, RS485, zigbee adapters are supported. USB+I2C cards like GPRS/3G/4G adapters, LoRa WAN adapters are supported. *NOTE: we are currently performing compatibility tests on a wide range of cards, and will publish a list of known working devices.* |
| 22 | Artik020 MCU with embedded Bluetooth 4.2 and aerial.  *NOTE: due to a design mistake, the module is connected to the i2c-0 bus on the CM3L instead of i2c-1, which will be fixed in the next revision.* |
| 23 | SODIMM-200 socket for the CM3L |
| 24 | 8GB eMMC. Wired to the CM3L (23) SDIO as main internal storage, this will be also offered in 16GB and eventually 32GB in the next revision |
| 25 | USB Hub controller. This Hub can be dynamically configured via i2c, meaning that user can select which USB interface (LAN, mPCIe, USB1, USB2) has to be powered on. All the interfaces are ON by default. |
| 26 | RGB LED. This LED is connected to 3 Artik020 (22) GPIOs (one for each color) |
| 27 | LSM6DSL iNEMO. 3D digital accelerometer and a 3D digital gyroscope wired to the Artik020 (22) *NOTE: this component will be removed in the next revision* |
| 28 | Wifi+BT 4.1 embedded chip.  *NOTE: this component (CYW4343W) does not work on the current revision. You will need to add a USB dongle to Amber in order to get WiFi capabilities. The component will be also changed with another chip in the next revision.* |

Developer Instructions

# Flashing CM3L on Amber

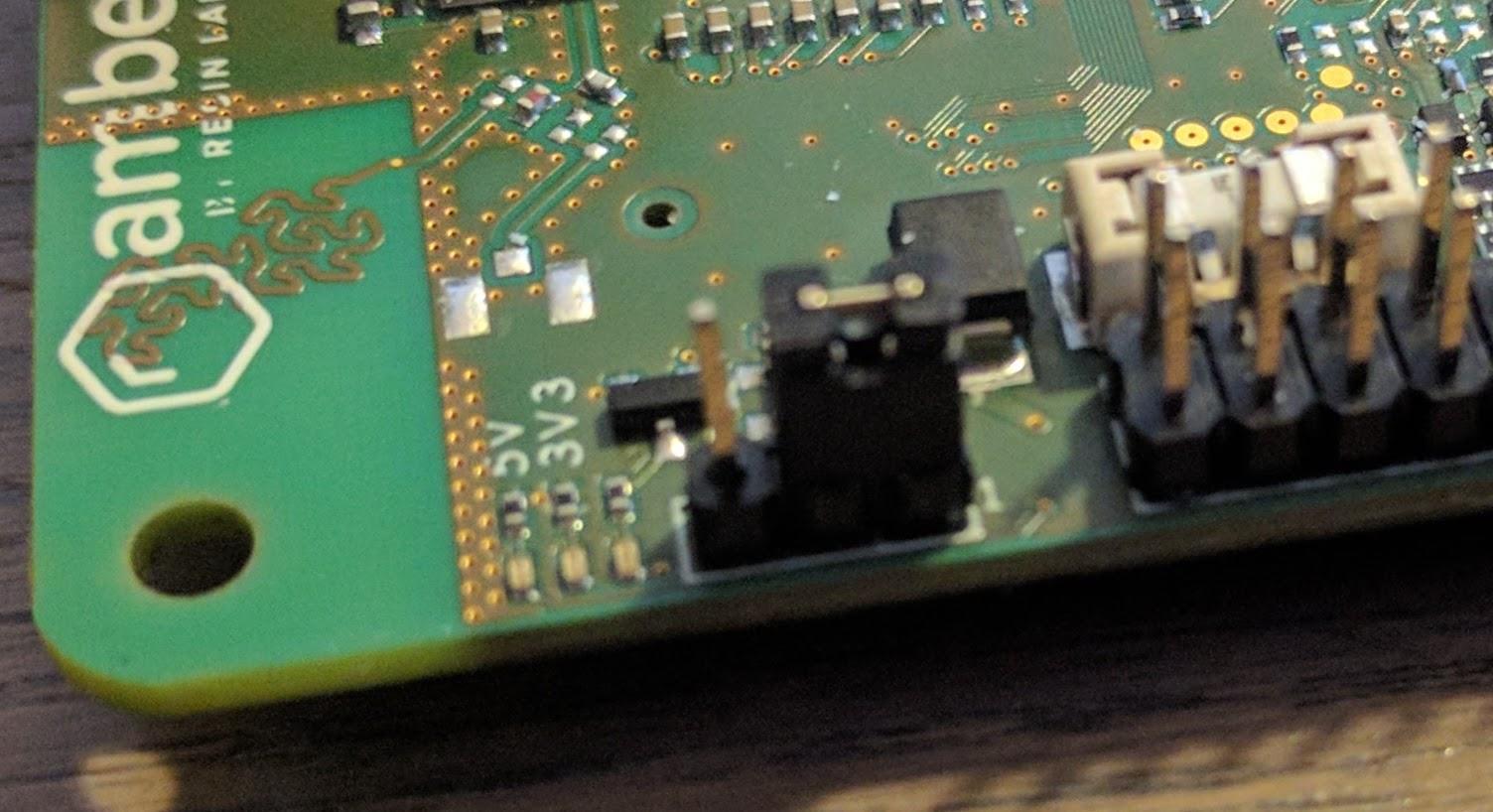
Amber supports the Raspberry Pi Compute Module 3 **lite** (and future new versions).

“Lite” means that the module itself has the eMMC socket unpopulated and the traces are exposed via SODIMM-200. This is very important since the standard CM3 has a fixed 4GB eMMC which was rated as not enough for a good multi-container experience. Instead, with CM3L, Amber can expose variable storage sizes via its embedded eMMC wired to the CM3L via the SODIMM-200 pins. The current revision uses a 8GB eMMC but the same footprint is also offered with 16 and 32GB from Microchip, the manufacturer of the eMMC module.

*NOTE: Any recent OS distribution (2017+) for the Raspberry Pi 3 Model B should be compatible with Amber. This includes of course Raspbian and ResinOS 2.x targeting the “raspberrypi3” device-type.*

The first thing you need to do is install a tool on your computer that allows Amber to present its eMMC as a mass storage device. The tool is provided by the Raspberry Pi Foundation itself and is also compatible with standard Raspberry Pis. Please follow their instructions [here](https://www.raspberrypi.org/documentation/hardware/computemodule/cm-emmc-flashing.md) until the actual flashing step (we have a better solution than *dd*).

Once you have everything set up on your computer, set the USB SLAVE BOOT ENABLE selector (6) on the flash position, with the jumper closer to the HAT connector as shown in the picture below.



Then run rpiboot on your computer, connect Amber via CM3L Debug port (9) and power it from the POWER IN port (12).

*NOTE: a CM3L module needs to be inserted in Amber!*

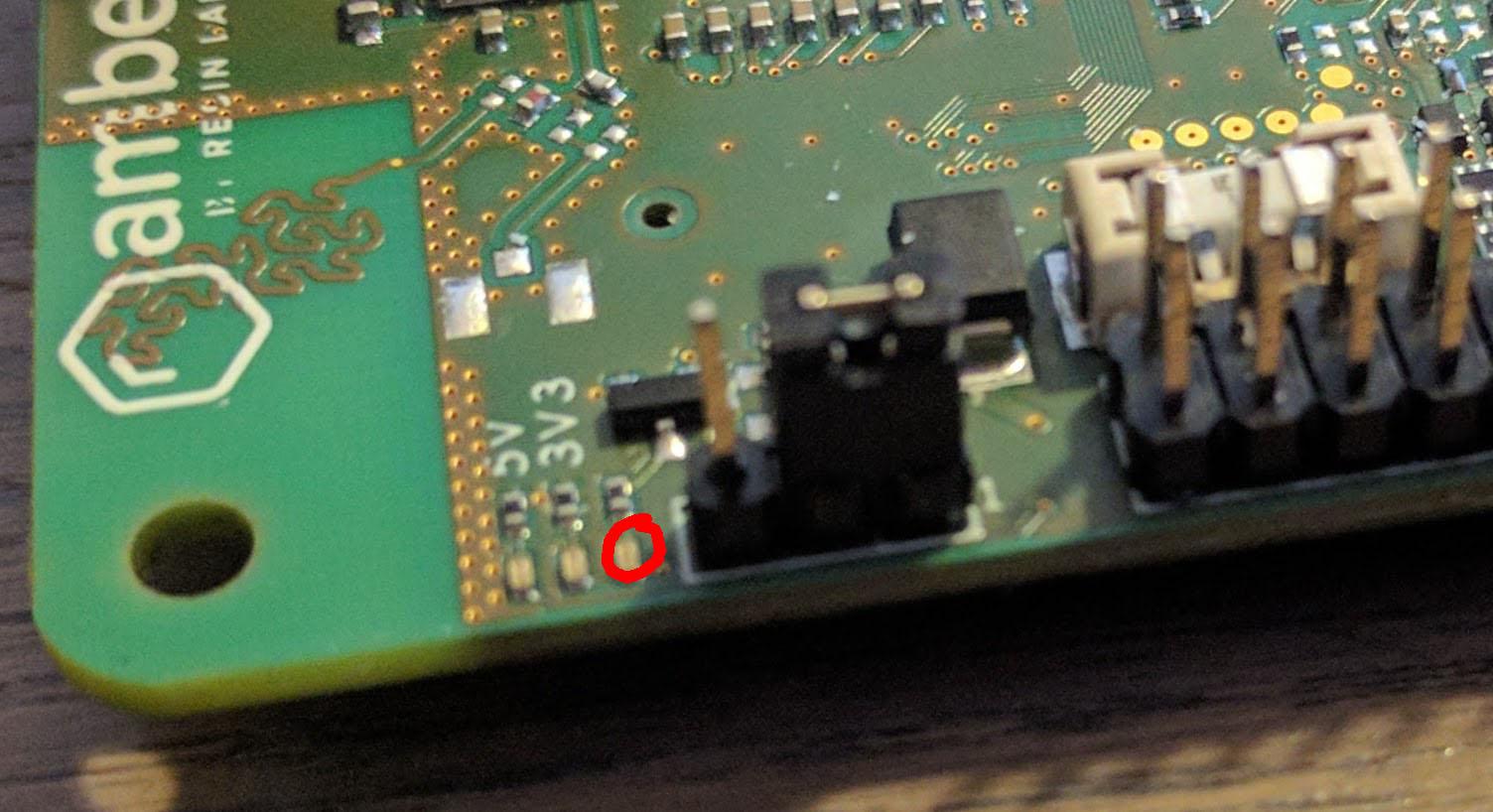
After a couple of seconds, the Amber eMMC should be mounted as mass storage on your computer. You can then start Etcher, which will list the eMMC as a 7.7GB device named “0001” (the name might change in the future). Proceed as usual selecting the image you want to write and press the “Flash!” button.op

*NOTE: the slow speed of around 5.8Mbps is due to the usbboot implementation, not the eMMC itself - you’ll notice how faster it is compared to the best microSD card when booting Amber*



After flash is complete, power off Amber, put the USB SLAVE BOOT ENABLE selector (6) on the boot position, with the jumper closer to the aerial. Powering Amber on will result in the device booting from the freshly-written eMMC.

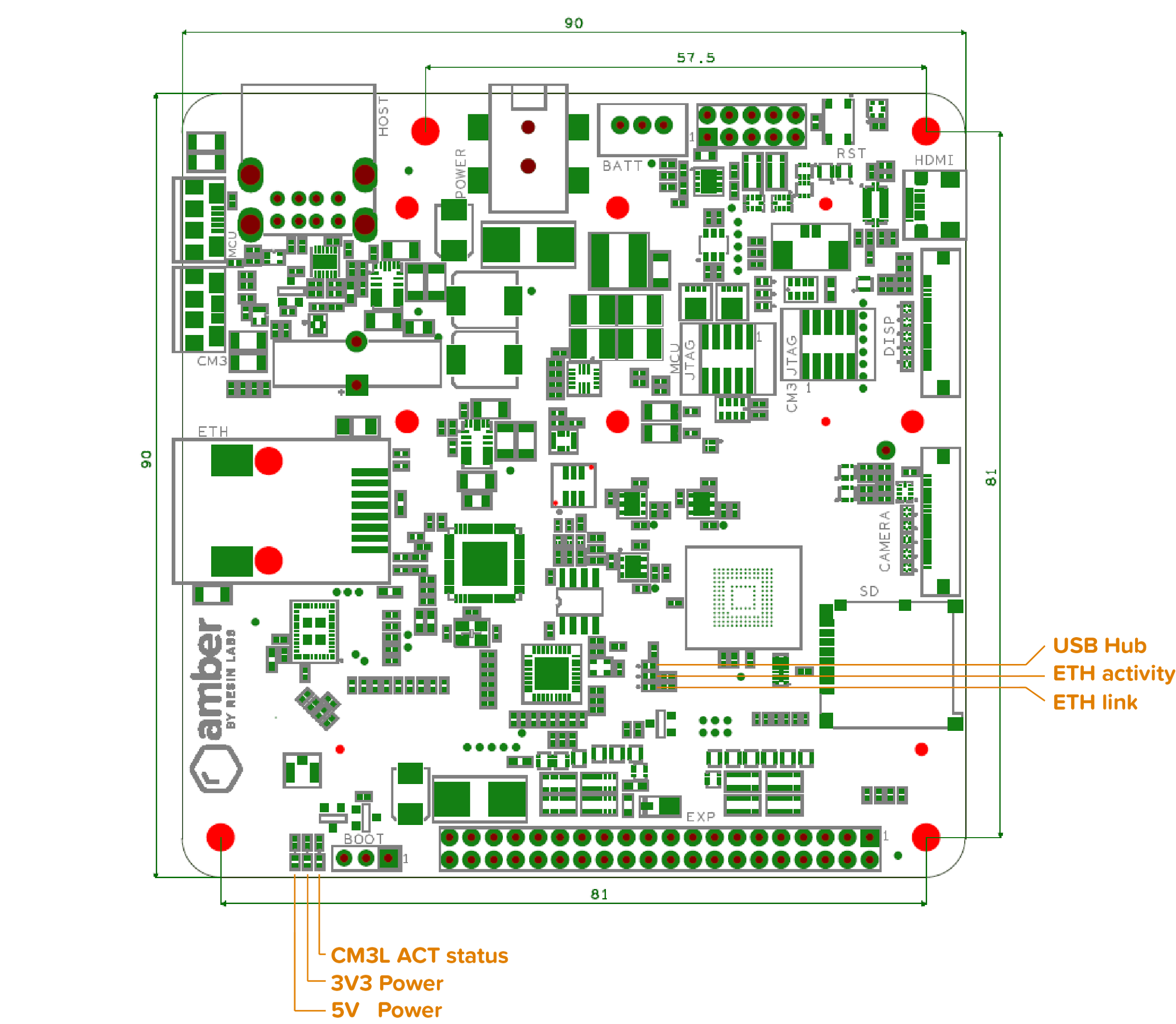
*NOTE: On managed ResinOS, If you configured networking correctly on the resin.io dashboard while downloading the image to flash, this is the point when the device provisions and shows up in your application dashboard. If the device fails to connect to the configured WiFi SSID, the usual 4-blinks pattern will be showed on the CM3L ACT status LED (the green LED next to USB SLAVE BOOT ENABLE selector (6), below the 3v3 LED)*



# Enabling i2c-0 on the CM3L for Artik020 access

Even if this is a design mistake and will get fixed in the next iteration, you can still work on the communication between CM3L and Artik020 enabling the i2c-0 bus with a device tree parameter: **dtparam=i2c0=on**

# Status LEDs on Amber



# 

# 

# Enclosing the prototype

There is a 3d-printable case for this revision, that can be found [here](https://drive.google.com/drive/folders/0BwMoybaHFhYKNGZnSmUwWndwbkk?usp=sharing). Only 4 x 40mm M3 screws and 4 x M3 nuts are required (better if both nylon)

