BeagleV-Ahead

Release 1.0.20240509-wip

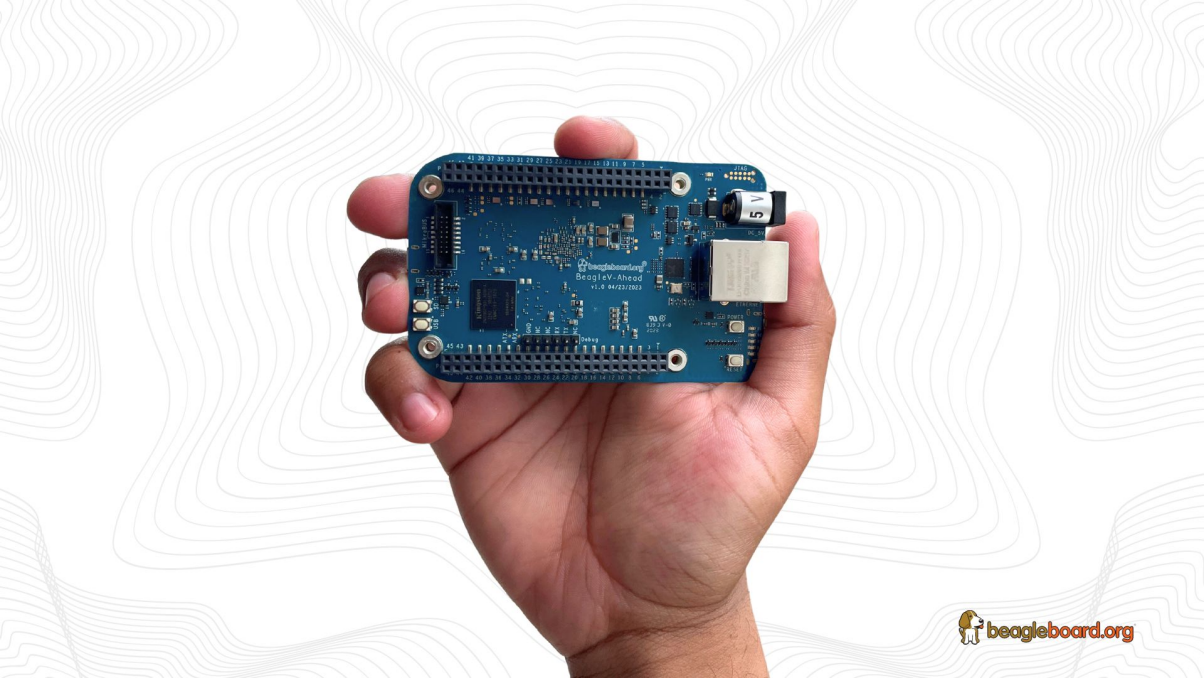
BeagleBoard.org Foundation
May 09, 2024

Table of contents

BeagleV-Ahead is a high-performance open-source RISC-V single board computer (SBC) built around the Alibaba TH1520 SoC. It has the same P8 & P9 cape header pins as BeagleBone Black allowing you to stack your favourite BeagleBone cape on top to expand it's capability. Featuring a powerful quad-core RISC-V processor BeagleV Ahead is designed as an affordable RISC-V enabled pocket-size computer for anybody who want's to dive deep into the new RISC-V ISA.

License Terms

- This documentation is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License
- Design materials and license can be found in the git repository
- Use of the boards or design materials constitutes an agreement to the boards-terms-and-conditions
- Software images and purchase links available on the board page
- For export, emissions and other compliance, see *Support*
- All support for BeagleV Ahead design is through BeagleBoard.org community at BeagleBoard.org forum.

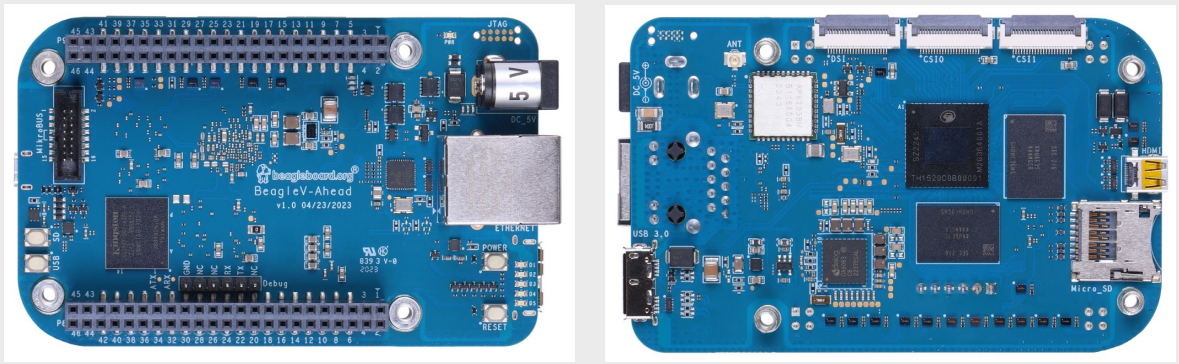


Important: This is a work in progress, for latest documentation please visit https://docs.beagleboard.org/latest/

Chapter 1

Introduction

BeagleV-Ahead is a high-performance open-source RISC-V single board computer (SBC) built around the Alibaba TH1520 SoC. It has the same P8 & P9 cape header pins as BeagleBone Black allowing you to stack your favourite BeagleBone cape on top to expand it's capability. Featuring a powerful quad-core RISC-V processor BeagleV Ahead is designed as an affordable RISC-V enabled pocket-size computer for anybody who want's to dive deep into the new RISC-V ISA.



1.1 Pinout Diagrams

Choose the cape header to see respective pinout diagram.

P8 cape header

P9 cape header

1.2 Detailed overview

BeagleV Ahead is build around T-Head TH1520 RISC-V SoC with quad-core Xuantie C910 processor clocked at 1.85GHz with a 4 TOPS NPU, support for 64-bit DDR, and audio processing using a single core C906.

Todo: remove "<To-Do>" items in the table below.

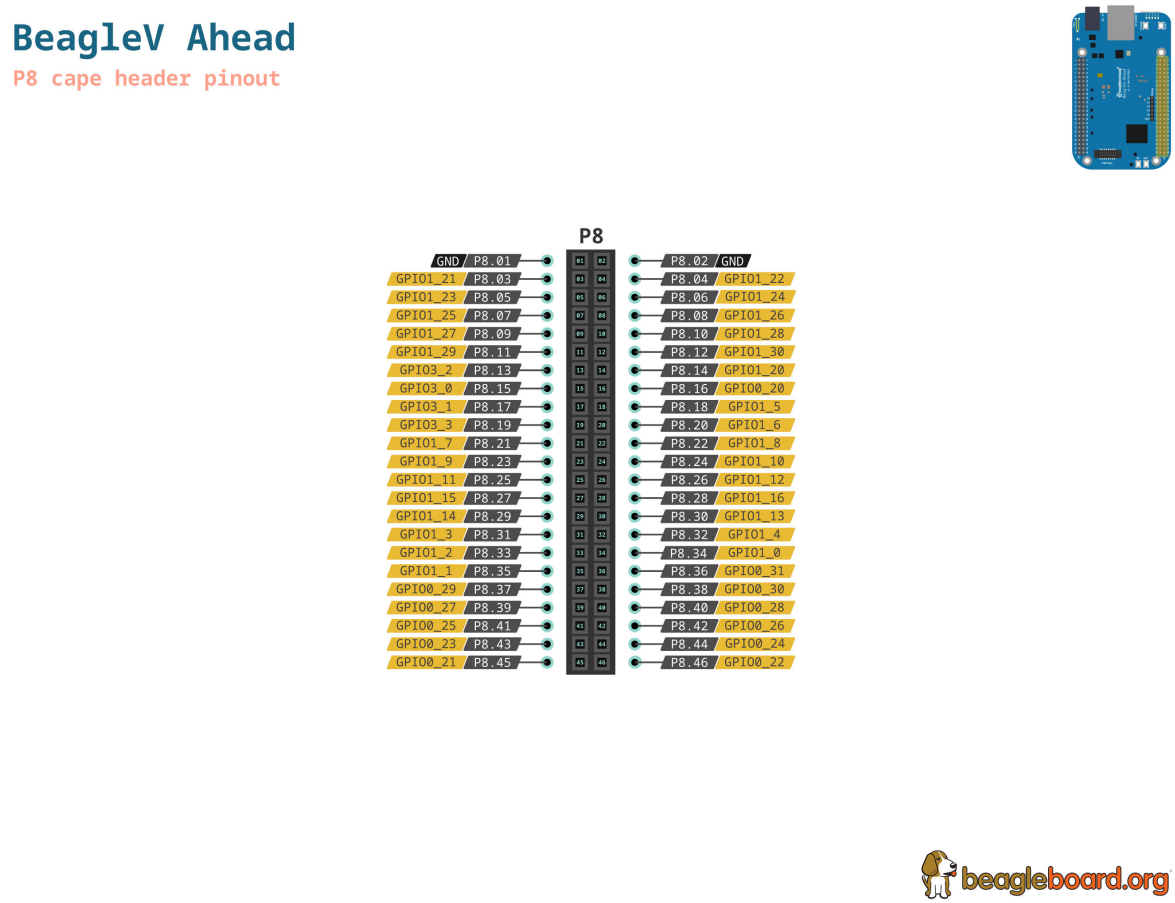


Fig. 1.1: BeagleV Ahead P8 cape header pinout

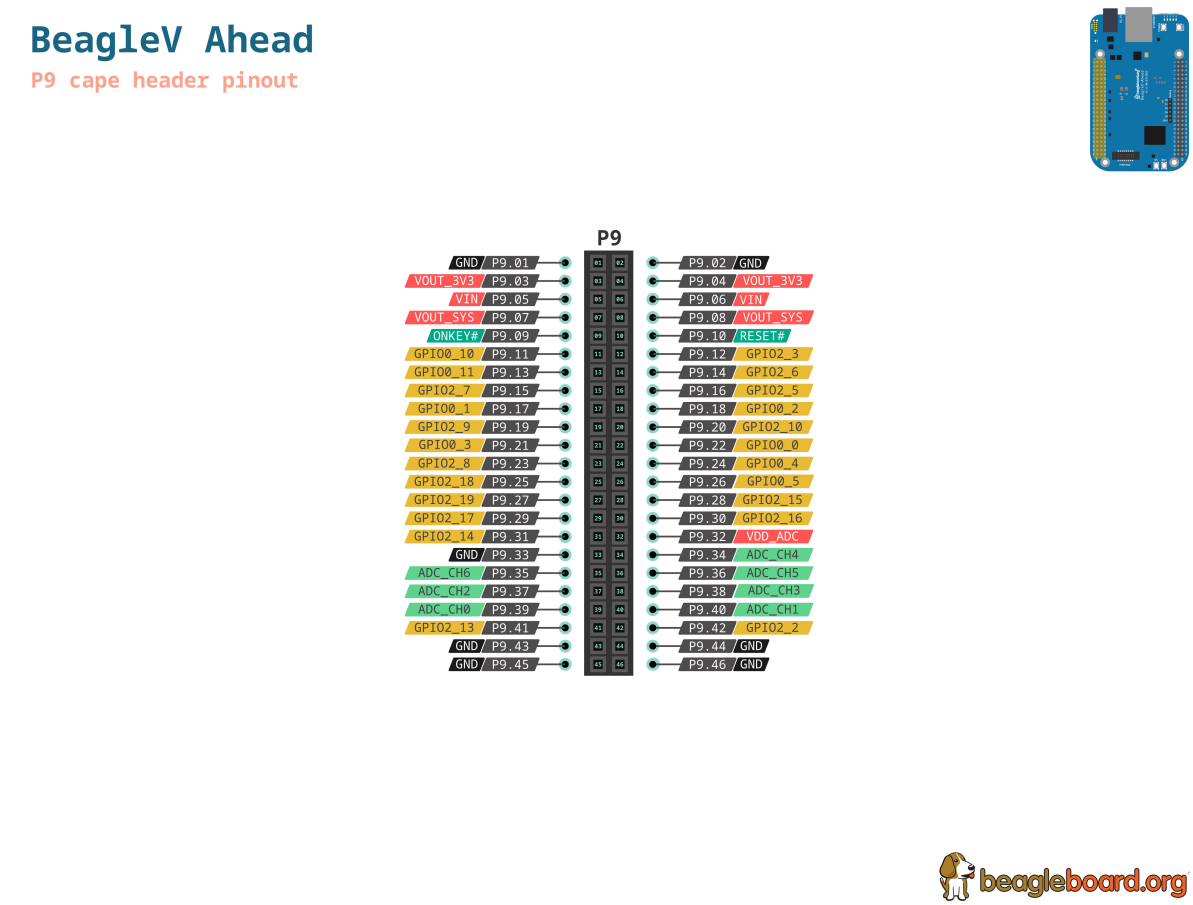


Fig. 1.2: BeagleV Ahead P9 cape header pinout

Table 1.1: BeagleV Ahead features

Feature	Description
Processor	T-Head TH1520 (quad-core Xuantie C910 processor)
PMIC	DA9063
Memory	4GB LPDDR4
Storage	16GB eMMC
WiFi/Bluetooth	• PHY: AP6203BM • Antennas: 2.4GHz & 5GHz
Ethernet	• PHY: Realtek RTL8211F-VD-CG Gigabit Ethernet phy • Connector: integrated magnetics RJ-45
microUSB 3.0	• Connectivity: USB OTG, Flash support • Power: Input: 5V @ <To-Do>, Output: 5V @ <To-Do>
HDMI	• Transmitter: TH1520 Video out system • Connector: Mini HDMI
Other connectors	• microSD • mikroBUS shuttle connector (I2C/UART/SPI/ADC/PWM/GPIO) • 2 x CSI connector compatible with BeagleBone AI-64, Raspberry Pi Zero / CM4 (22-pin) • DSI connector

1.3 Board components location

This section describes the key components on the board, their location and function.

1.3.1 Front components location

Table 1.2: BeagleV Ahead board front components location

Feature	Description
Power LED	Power (Board ON) indicator
JTAG (TH1520)	TH1520 SoC JTAG debug port
Barrel jack	Power input
GigaBit Ethernet	1Gb/s Wired internet connectivity
User LEDs	Five user LEDs, board-power-and-boot section provides more details. These LEDs are connect to the TH1520 SoC
Reset button	Press to reset BeagleV Ahead board (TH1520 SoC)
Power button	Press to shut-down (OFF), hold down to boot (ON)
P8 & P9 cape header	Expansion headers for BeagleBone capes.
UART debug header	6 pin UART debug header
USB boot button	Hold and reset board (power cycle) to flash eMMC via USB port
SD boot button	Hold and reset board (power cycle) to boot from SD Card
mikroBUS shuttle	16pin mikroBUS shuttle connector for interfacing mikroE click boards
16GB eMMC	Flash storage
RTL8211F	Gigabit IEEE 802.11 Ethernet PHY

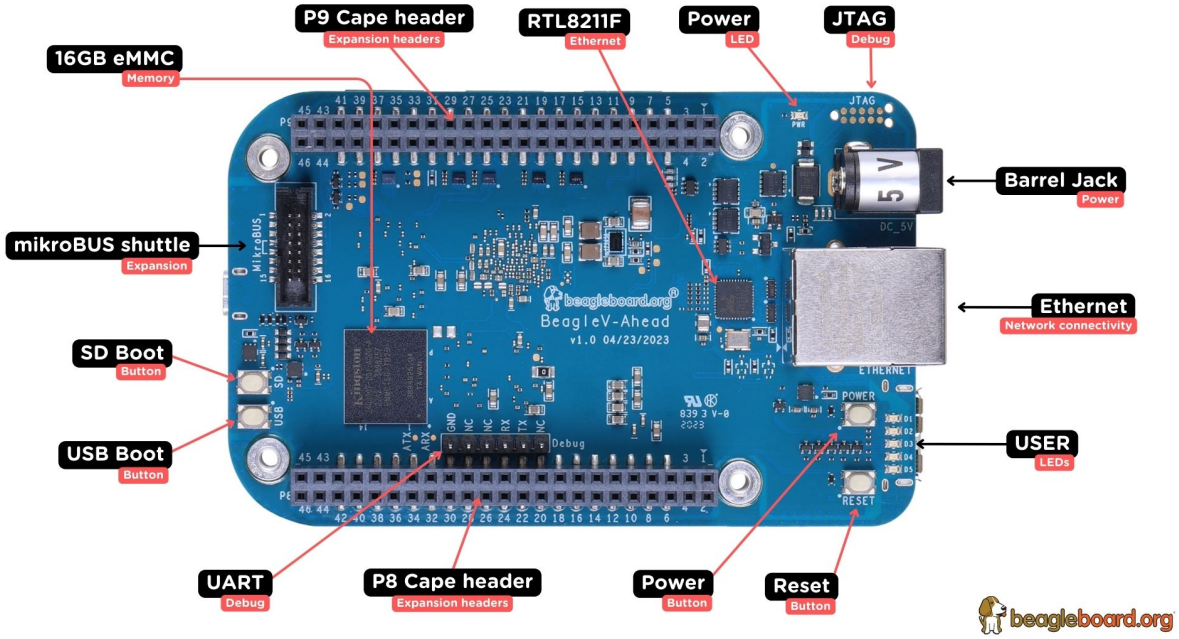


Fig. 1.3: BeagleV Ahead board front components location

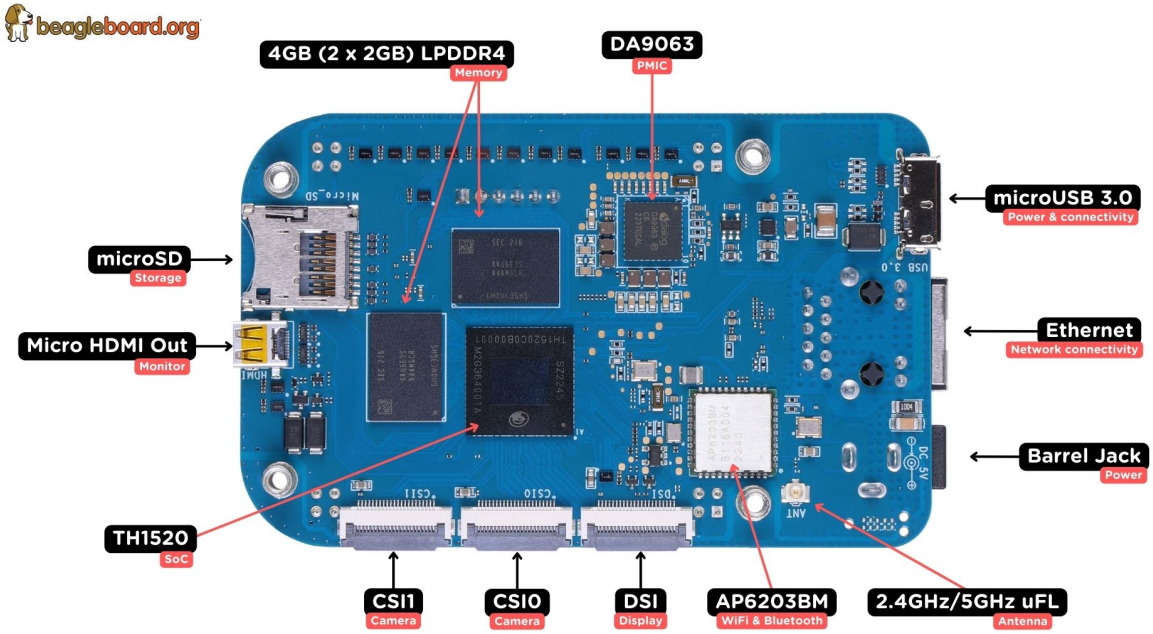


Fig. 1.4: BeagleV Ahead board back components location

1.3.2 Back components location

Table 1.3: BeagleV Ahead board back components location

Feature	Description
DA9063	Dialog semi Power Management Integrated Circuit (PMIC)
microUSB 3.0	Power & USB connectivity as client or Host (OTG)
Antenna connector	2.4GHz/5GHz uFL connector
AP6203BM	Ampak WiFi & BlueTooth combo
DSI	MIPI Display connector
CSI0 & CSI1	MIPI Camera connectors
TH1520	T-Head quad-core C910 RISC-V SoC
Mini HDMI	HDMI connector
microSD	Micro SD card holder
4GB RAM	2 x 2GB LPDDR4 RAM

Chapter 2

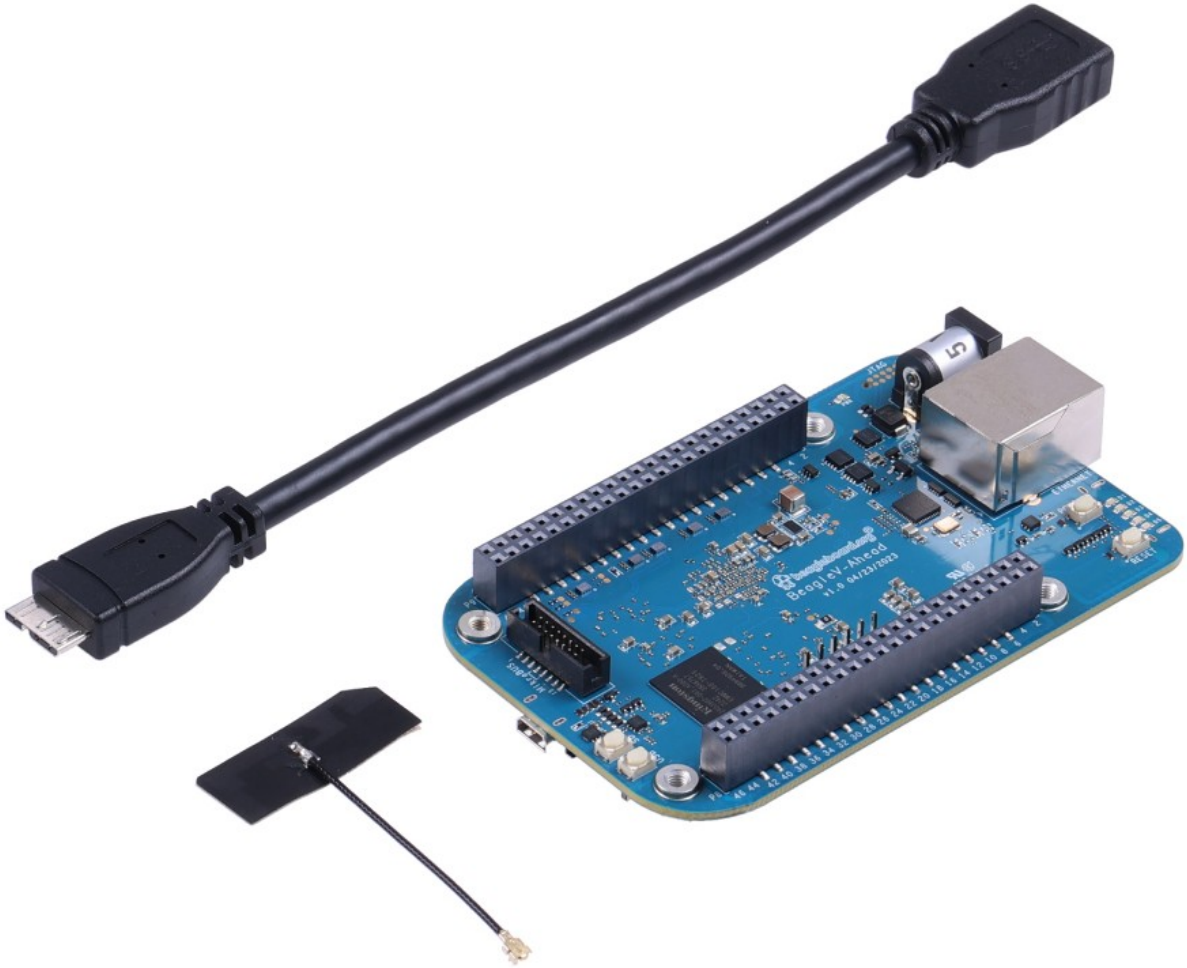
Quick Start

2.1 What's included in the box?

When you purchase a brand new BeagleV Ahead, In the box you'll get:

1. BeagleV Ahead board
2. One (1) 2.4GHz/5GHz antenna
3. USB super-speed micro-A plug to type-A receptacle cable (for connecting common USB type-A peripherals)
4. Quick-start card

Tip: For board files, 3D model, and more, you can checkout BeagleV-Ahead repository on OpenBeagle.



2.2 Unboxing

2.3 Antenna guide

Warning: uFL antenna connectors are very delicate and should be handled with care.

Connecting antenna

To use WiFi you are **required** to connect the 2.4GHz/5GHz antenna provided in BeagleV Ahead box. Below is a guide to connect the antenna to your BeagleV Ahead board.

Disconnecting antenna

If for some reason you want to disconnect the antenna from your BeagleV Ahead board you can follow the guide below to remove the antenna without beaking the uFL antenna connector.

2.4 Tethering to PC

To connect the board to PC via USB 3.0 port you can use either a standard high-speed micro-B cable or a USB 3.0 super-speed micro-B cable. Connection guide for both are shown below:

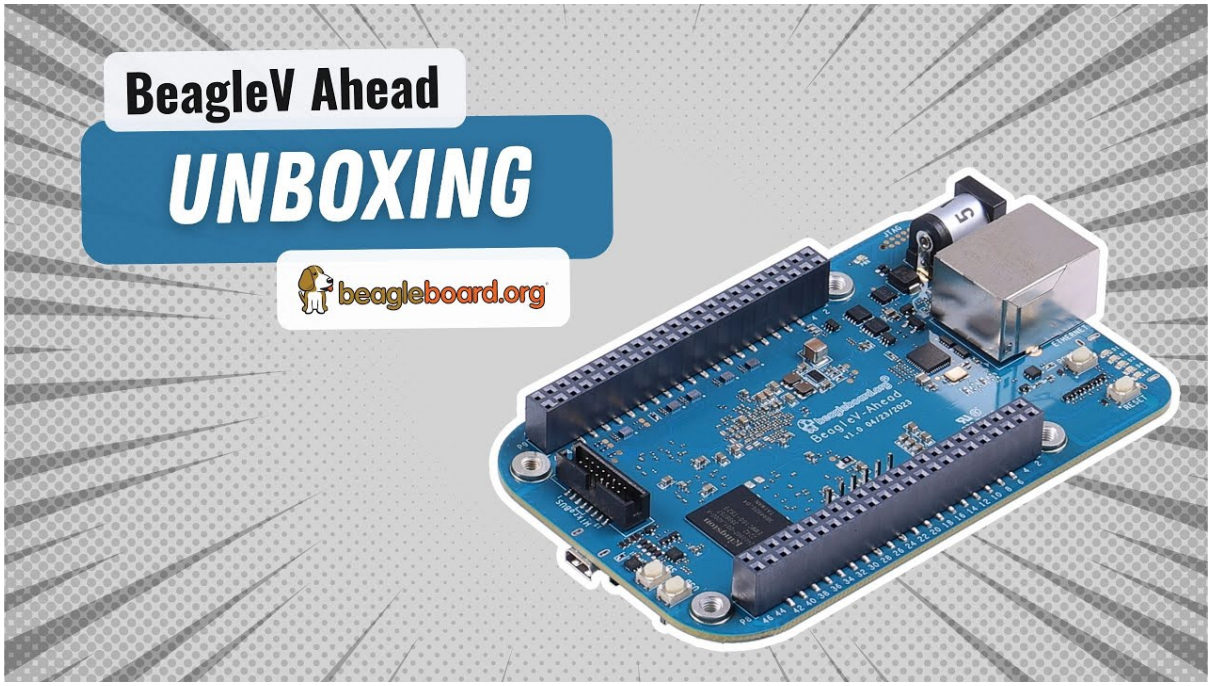


Fig. 2.1: https://youtu.be/SVC9peUUzE0

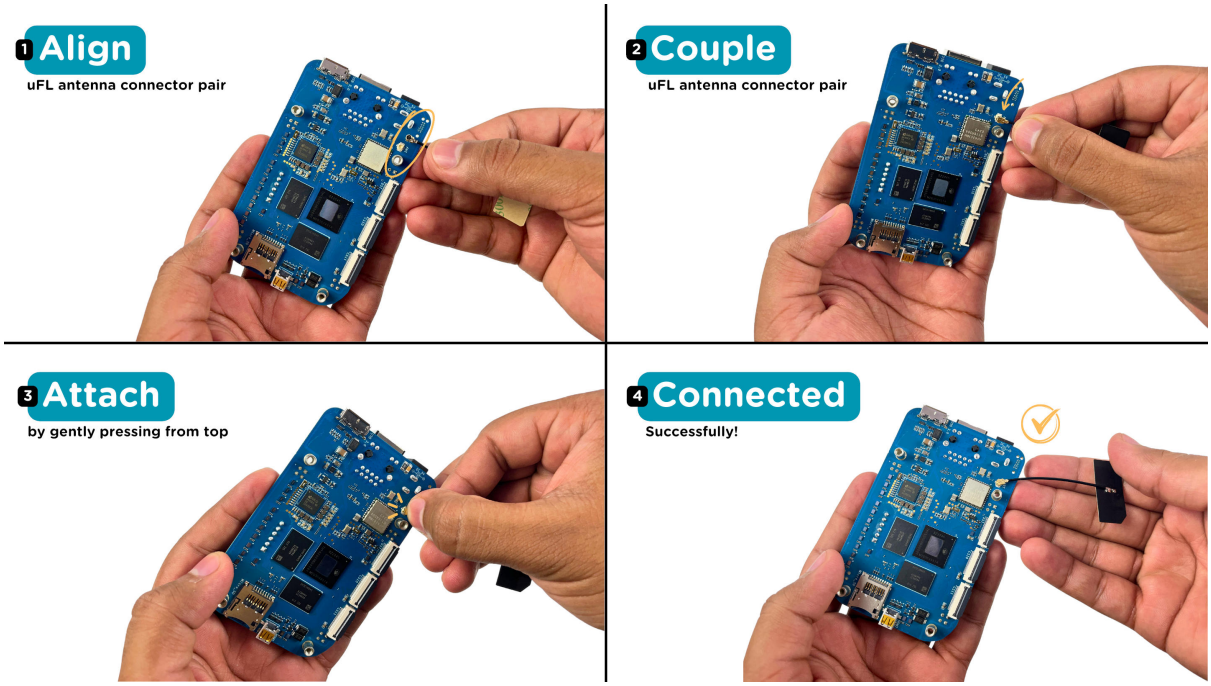


Fig. 2.2: Connecting 2.4GHz/5GHz antenna to BeagleV Ahead.

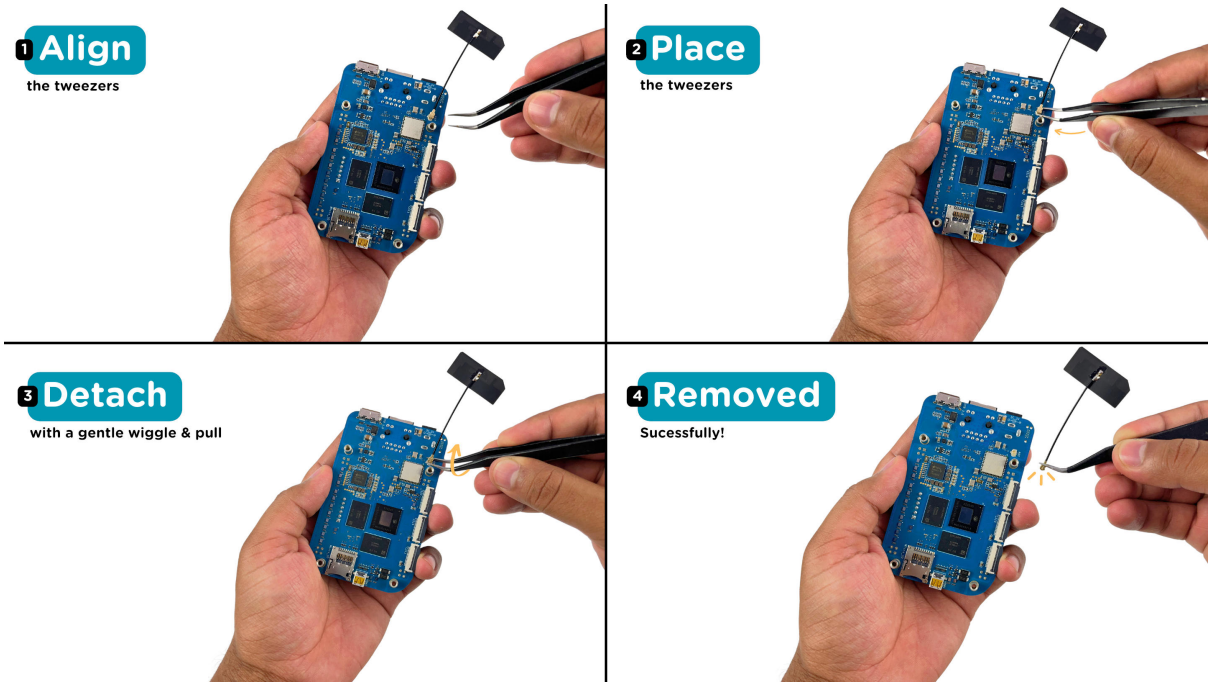


Fig. 2.3: Removing 2.4GHz/5GHz antenna to BeagleV Ahead.

Important: high-speed micro-B will support only USB 2.0 speed but super-speed micro-B cable will support USB 3.0 speed.

---

super-speed micro-B connection (USB 3.0)

For super speed USB 3.0 connection it's recommended to use super-speed micro-B USB cable. To get a super-speed micro-B cable you can checkout links below:

1. USB 3.0 Micro-B Cable - 1m (sparkfun)
2. Stewart Connector super-speed micro-B (DigiKey)
3. CNC Tech super-speed micro-B (DigiKey)
4. Assmann WSW Components super-speed micro-B (DigiKey)

Note: If you only have a high-speed micro-B cable you can checkout high-speed micro-B connection guide.

---

high-speed micro-B connection (USB 2.0)

For USB 2.0 connection it's recommended to use high-speed micro-B USB cable. To get a high-speed micro-B cable you can checkout links below:

1. USB micro-B Cable - 6 Foot (sparkfun)
2. Stewart Connector high-speed micro-B (DigiKey)
3. Assmann WSW Components high-speed micro-B (DigiKey)
4. Cvilux USA high-speed micro-B (DigiKey)

Note: Make sure the high-speed micro-B cable you have is a data cable as some high-speed micro-B cables are power only.

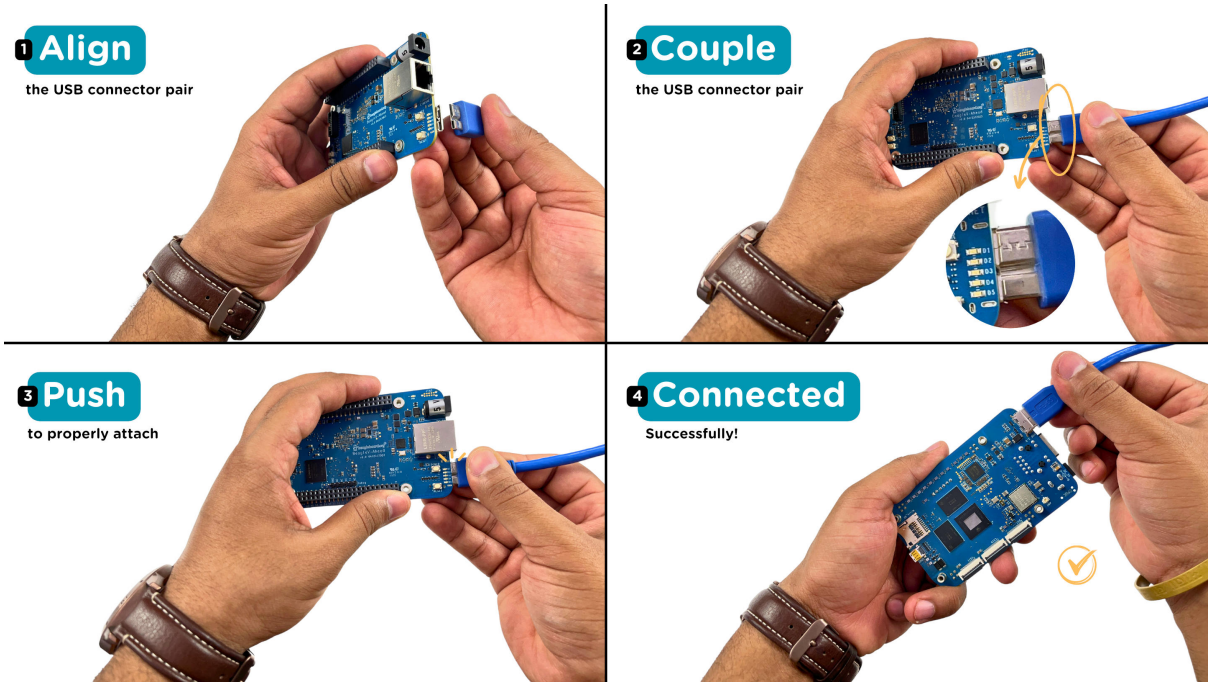


Fig. 2.4: super-speed micro-B (USB 3.0) connection guide for BeagleV Ahead.

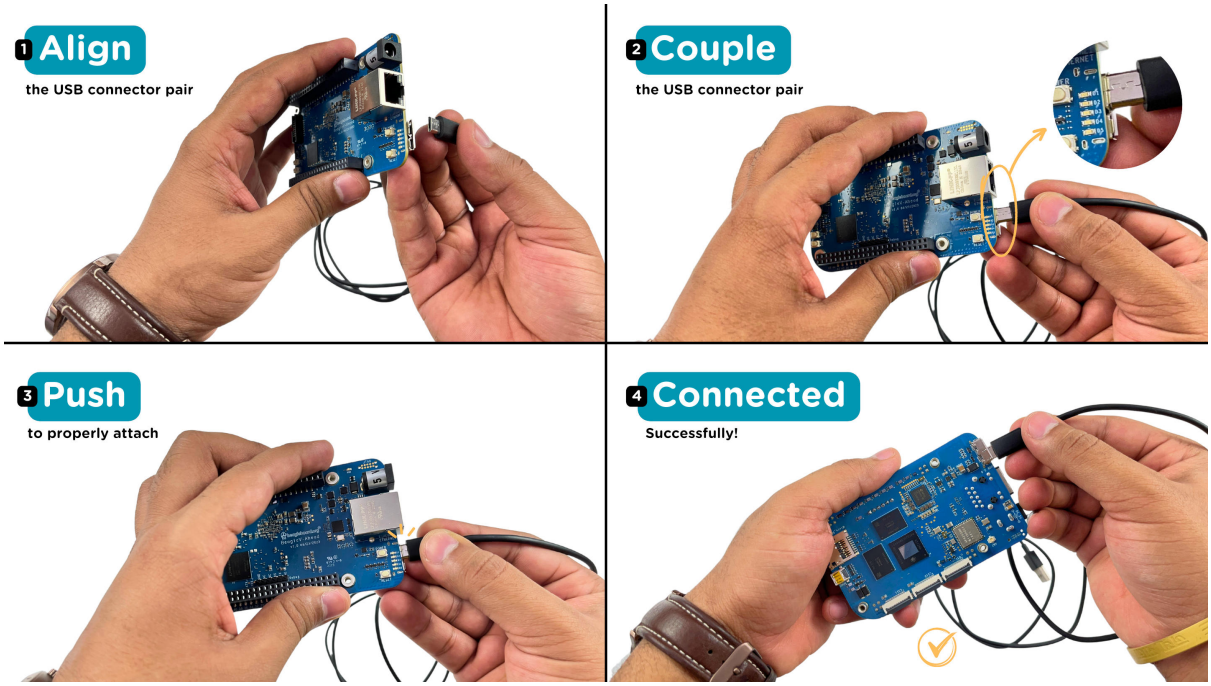


Fig. 2.5: high-speed micro-B (USB 2.0) connection guide BeagleV Ahead.

2.5 Flashing eMMC

Note: To flash your BeagleV Ahead you need either a super-speed micro-B or high-speed micro-B cable as shown in section above.

2.5.1 Download latest software image

To download the latest software image visit https://www.beagleboard.org/distros and search for BeagleV Ahead as shown below.

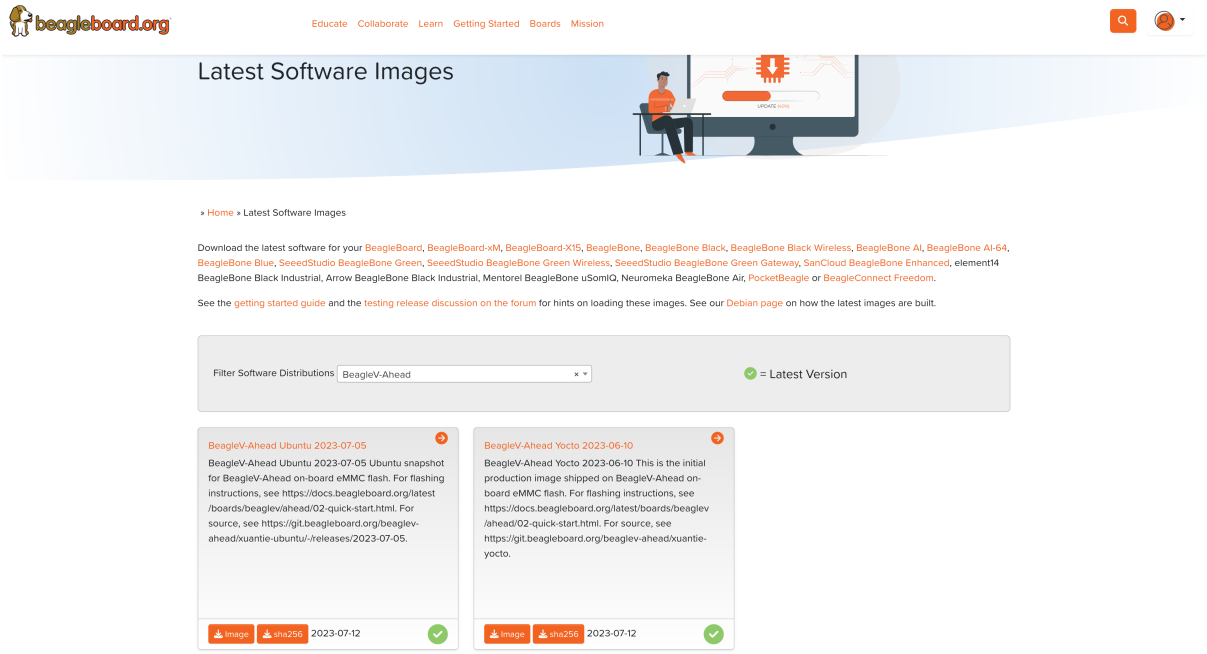


Fig. 2.6: Download latest software image for BeagleV Ahead board

2.5.2 Put BeagleV Ahead in USB flash mode

Note: Only super-speed micro-B is shown in graphic below but you can use a high-speed micro-B cable. Only difference will be lower flash speeds.

To put your BeagleV Ahead board into eMMC flash mode you can follow the steps below:

1. Press and hold USB button.
2. Connect to PC with super-speed micro-B or high-speed micro-B cable.
3. Release USB button.

Important: If you want to put the board into eMMC flashing while it is already connected to a PC you can follow these steps:

1. Press and hold USB button.
2. Press reset button once.

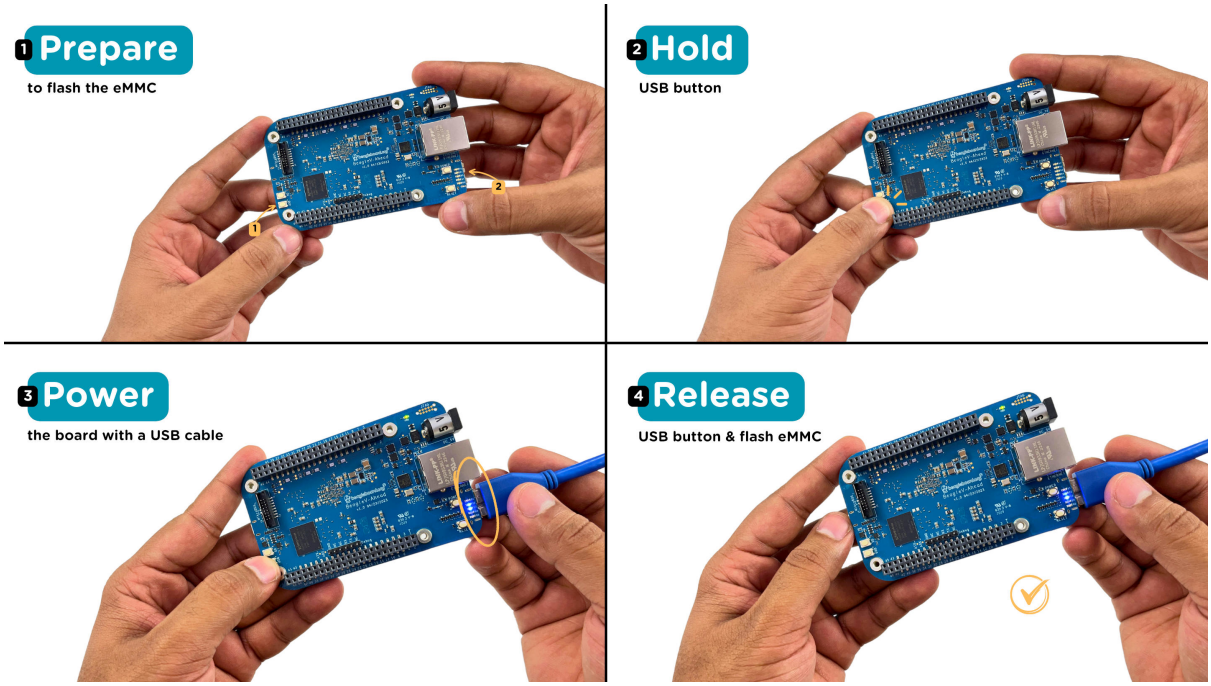


Fig. 2.7: Connecting BeagleV Ahead to flash eMMC

3. Release USB button.

2.5.3 Flash the latest image on eMMC

Linux

First you need to install android platform tools which includes *adb* and *fastboot*.

- Debian/Ubuntu-based Linux users can type the following command:

```
sudo apt-get install android-sdk-platform-tools
```

- Fedora/SUSE-based Linux users can type the following command:

```
sudo dnf install android-tools
```

Now unzip the latest software image zip file you have downloaded from https://www.beagleboard.org/distros which contains four files shown below:

```
[lorforlinux@fedora deploy] $ ls
boot.ext4  fastboot_emmc.sh  root.ext4  u-boot-with-spl.bin
```

Important: Make sure your board is in flash mode, you can follow the guide above to do that.

To flash the board you just have to exexute the script *fastboot_emmc.sh* as root and provide your passoword:

```
[lorforlinux@fedora deploy] $ sudo ./fastboot_emmc.sh
[sudo] password for lorforlinux:
```

Windows

Todo: add instructions for flashing in windows.

Mac

Todo: add instructions for flashing in Mac.

2.6 Access UART debug console

Note: It has been noticed that 6pin FTDI cables like this doesn't seem work with BeagleV Ahead debug port and there might be other cables/modules that will show garbage when connected to the board.

Some tested devices that are working good includes:

1. Adafruit CP2102N Friend - USB to Serial Converter
2. Raspberry Pi Debug Probe Kit for Pico and RP2040

To access a BeagleV Ahead serial debug console you can connected a USB to UART to your board as shown below:

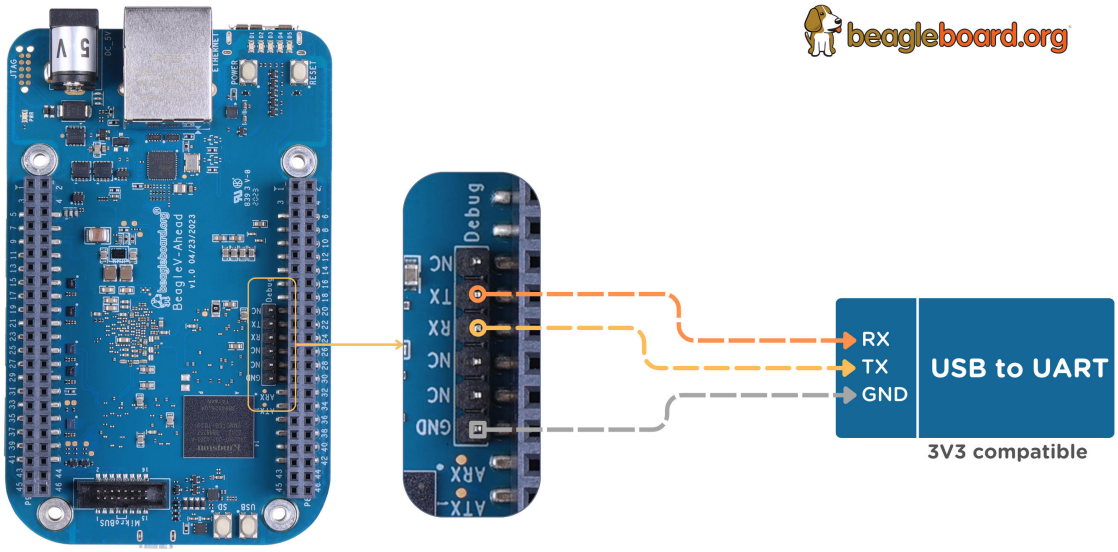


Fig. 2.8: BeagleV Ahead UART debug port connection

To see the board boot log and access your BeagleV Ahead's console you can use application like `tio` to access the conole. If you are using Linux your USB to UART converter may appear as `/dev/ttyUSB`. It will be different for Mac and Windows operatig systems. To find serial port for your system you can checkout this guide.

```
[lorforlinux@fedora ~] $ tio /dev/ttyUSB0
tio v2.5
Press ctrl-t q to quit
Connected
```

2.7 Connect USB gadgets

A super-speed micro-B (male) to USB A (female) OTG cable included in the box can be used to connect USB gadgets to your BeagleV Ahead board. When you do this, you'll be required to power the board via Barrel jack.

Important: To properly power the board and USB gadgets you must power the board with 5V @ 2A power supply.

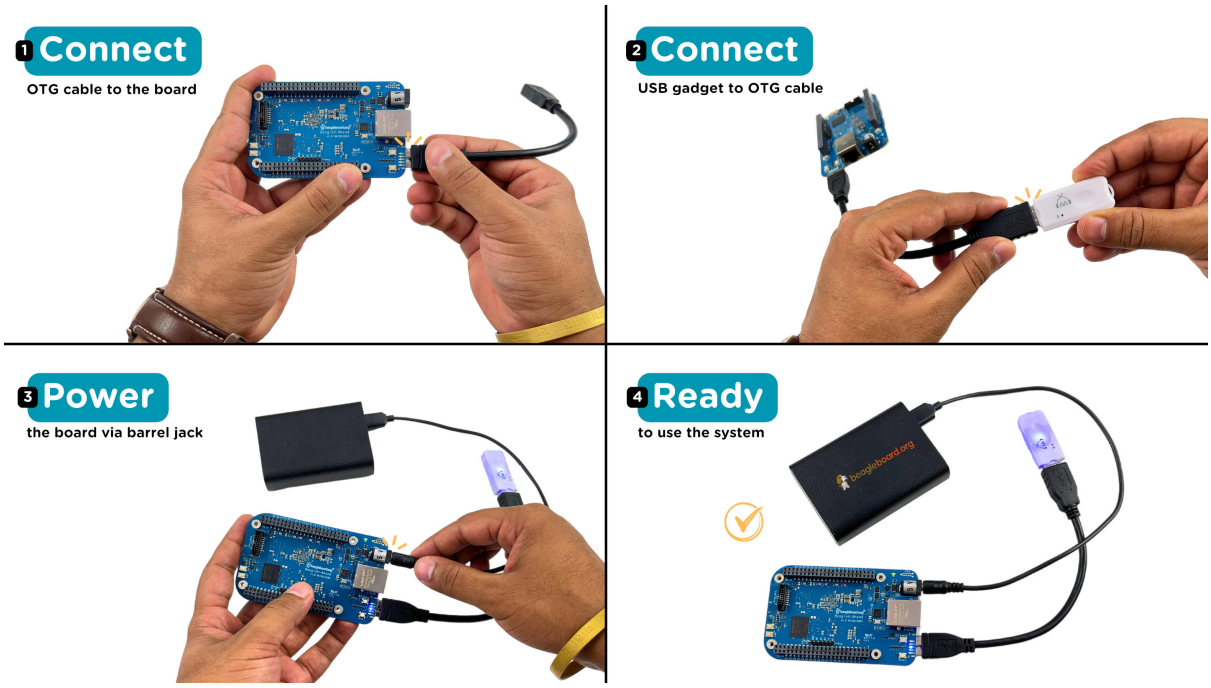


Fig. 2.9: USB OTG to connect USB gadgets to BeagleV Ahead board

2.8 Connect to WiFi

Yocto

After getting access to the UART debug console you will be prompted with,

```
THEAD C910 Release Distro 1.1.2 BeagleV ttyS0

BeagleV login:
```

Here you have to simply type `root` and press enter to start uisng your BeagleV Head board. Once you are in, to connect to any WiFi access point you have to edit the `/etc/wpa_supplicant.conf`

```
root@BeagleV:~# nano /etc/wpa_supplicant.conf
```

In the `wpa_supplicant.conf` file you have to provide `ssid` and `psk`. Here `ssid` is your WiFi access point name and `psk` is the passoword. It should look as shown below:

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
ap_scan=1
update_config=1
```

(continues on next page)

(continued from previous page)

```
network={
        ssid="My WiFi" ①
        psk="passoword" ②
        key_mgmt=WPA-PSK
}
```

① WiFi access point name

② WiFi passoword

Once you are done with editing the file you can save the file with `CTRL+O` and exit the nano editor with `CTRL+X`. Once you are back to terminal reconfigure the `wlan0` wireless interface which will trigger it to connect to the access point with the credentials you have added to `wpa_supplicant.conf`. Execute the command below to reconfigure `wlan0` wireless interface.

```
root@BeagleV:~# wpa_cli -i wlan0 reconfigure
OK
```

After executing this you can check if internet is working by executing `ping 8.8.8.8` as shown below:

```
root@BeagleV:~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=118 time=13.676 ms
64 bytes from 8.8.8.8: seq=1 ttl=118 time=17.050 ms
64 bytes from 8.8.8.8: seq=2 ttl=118 time=14.367 ms
64 bytes from 8.8.8.8: seq=3 ttl=118 time=19.320 ms
64 bytes from 8.8.8.8: seq=4 ttl=118 time=14.796 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 13.676/15.841/19.320 ms
```

Important: Due to a software issue Yocto might now assign any ip address to wlan0 wireless interface thus even if you are connected successfully to the access point of your choice you will still not be able to connect to the internet. Particularly If you are not getting any pings back when you execute `ping 8.8.8.8` you must execute the commands below:

1. `root@BeagleV:~# cp /lib/systemd/network/80-wifi-station.network.example /lib/systemd/network/80-wifi-station.network`
2. `root@BeagleV:~# networkctl reload`

this should fix the no internet issue on your BeagleV Ahead board!

2.9 Demos and Tutorials

- *Using CSI Cameras*

Chapter 3

Design & specifications

If you want to know how BeagleV Ahead board is designed and what are it's high-level specifications then this chapter is for you. We are going to discuss each hardware design element in detail and provide high-level device specifications in a short and crisp form as well.

3.1 Block diagram

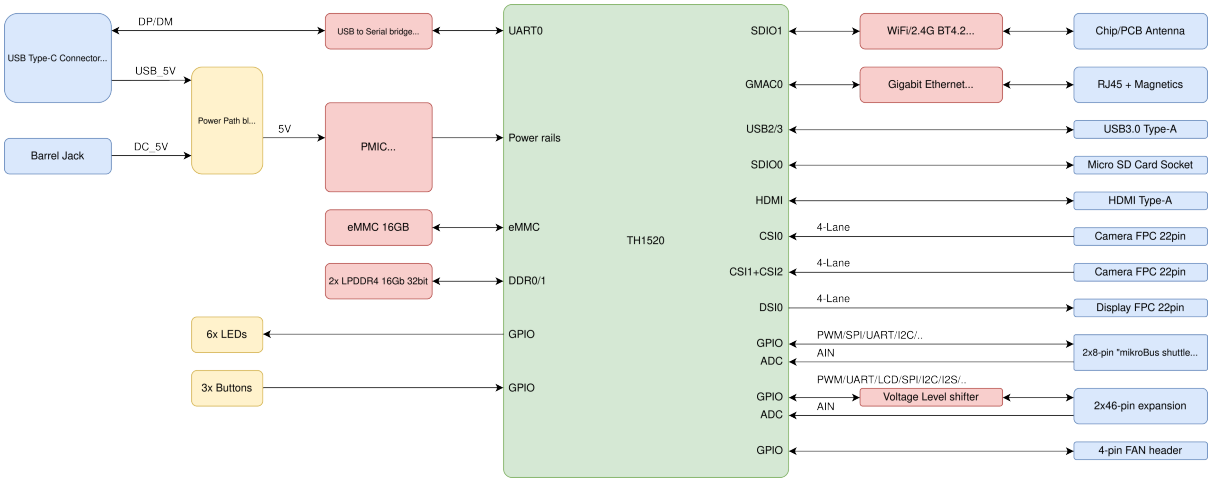


Fig. 3.1: System block diagram

3.2 System on Chip (SoC)

3.3 Power management

3.3.1 Barrel jack

3.3.2 0.8V DCDC buck

3.3.3 3.3V DCDC buck

3.3.4 1.8V LDO

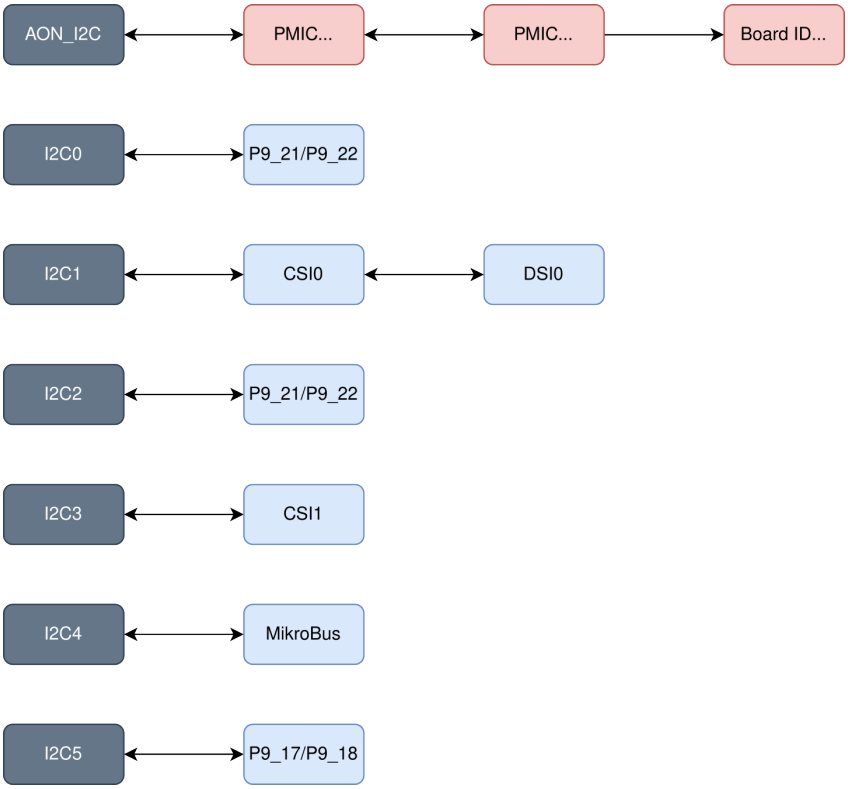


Fig. 3.2: I2C-Usage diagram

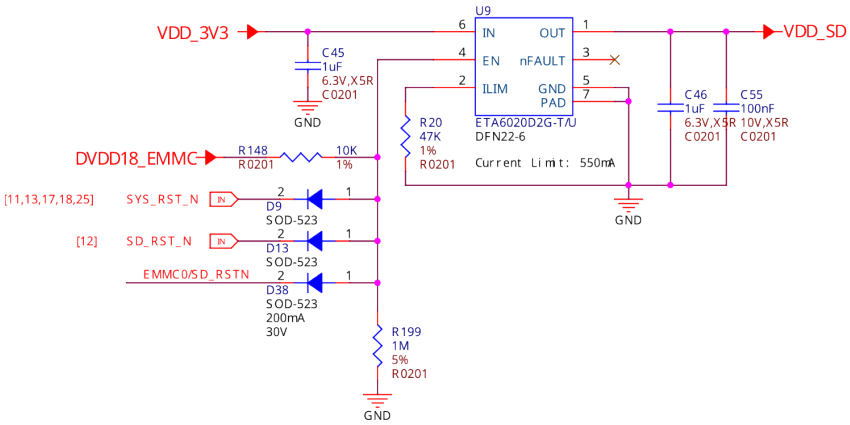


Fig. 3.3: SoC eMMC power switch

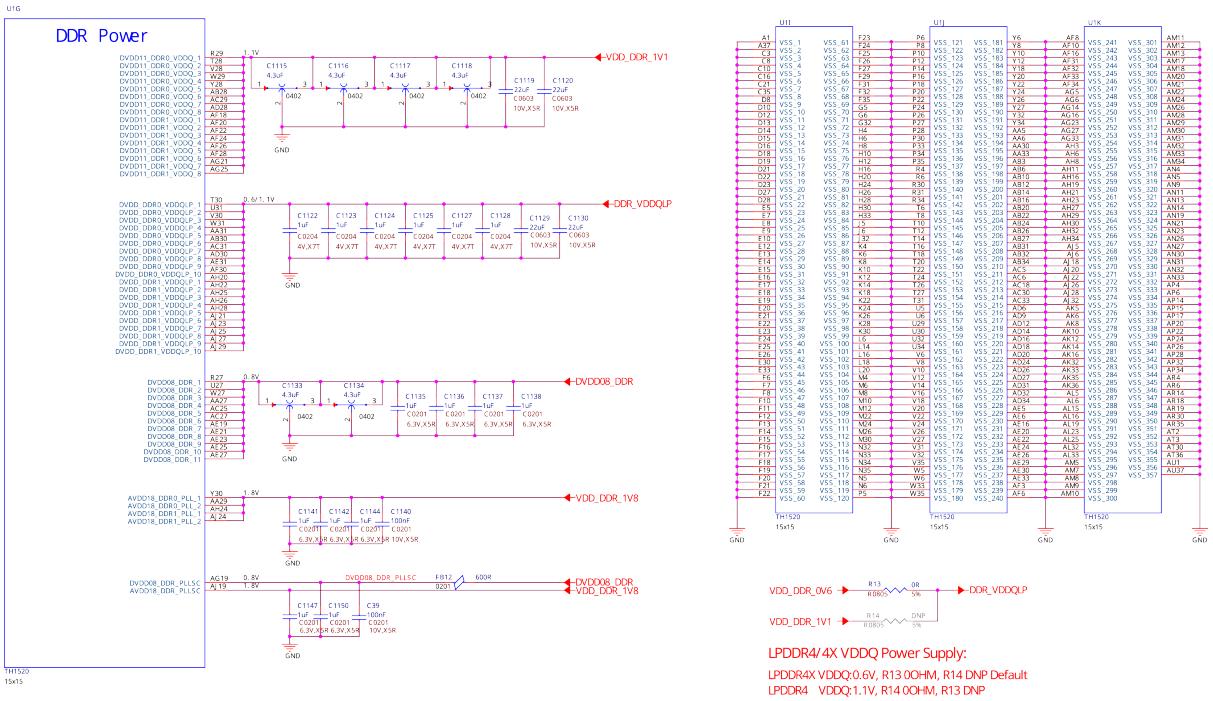


Fig. 3.4: SoC DDR Power

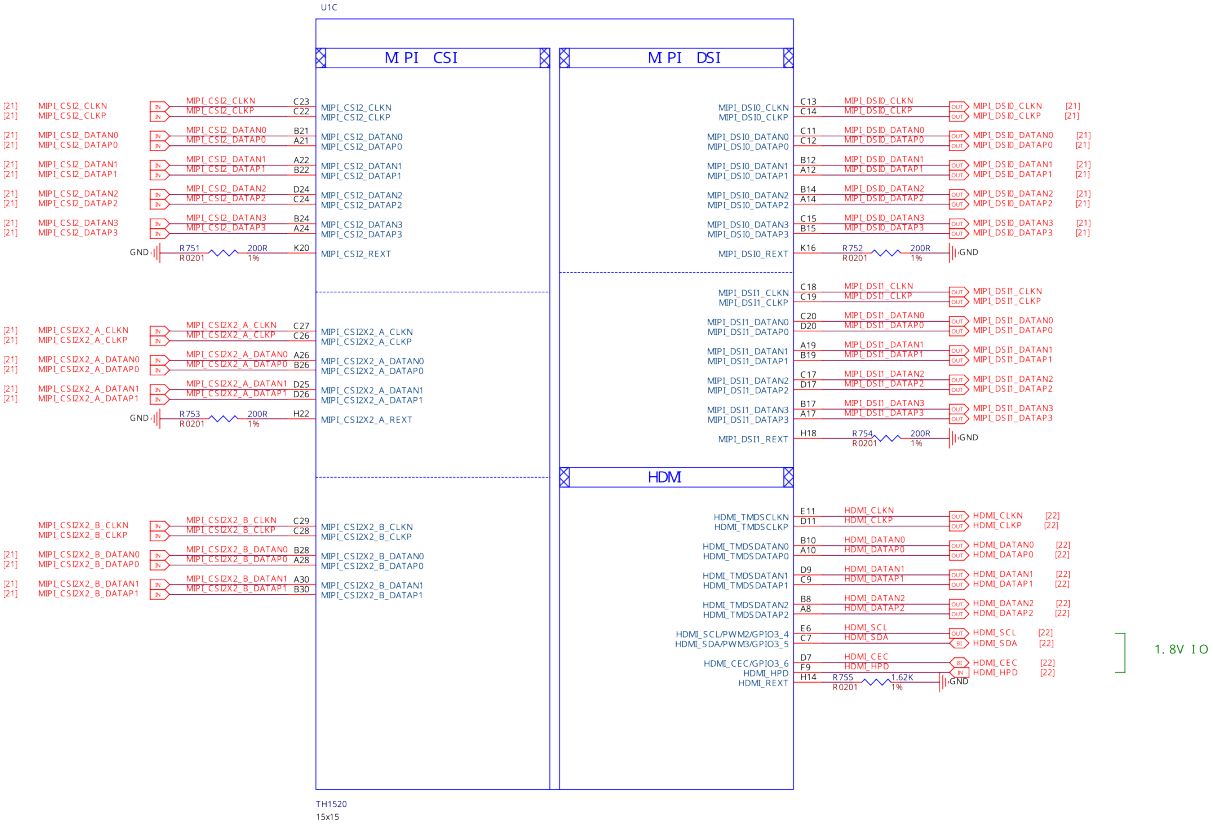


Fig. 3.5: SoC MIPI CSI DSI HDMI

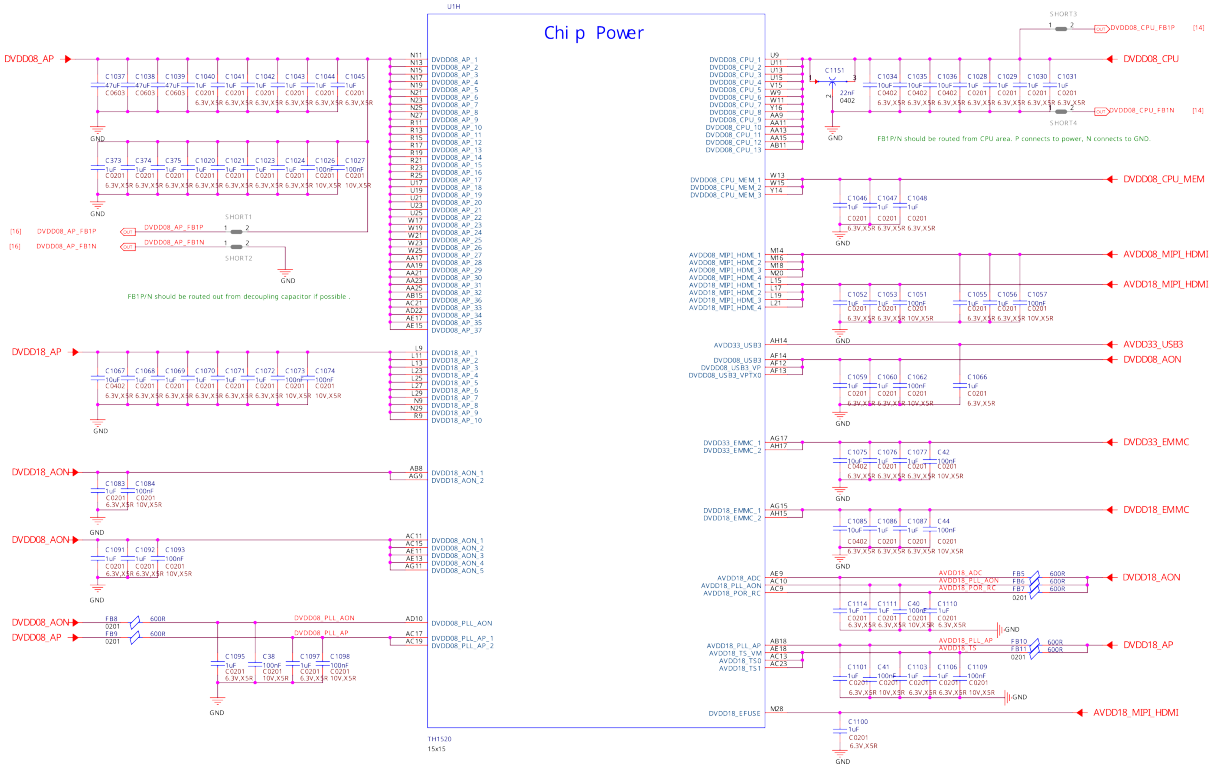


Fig. 3.6: SoC power

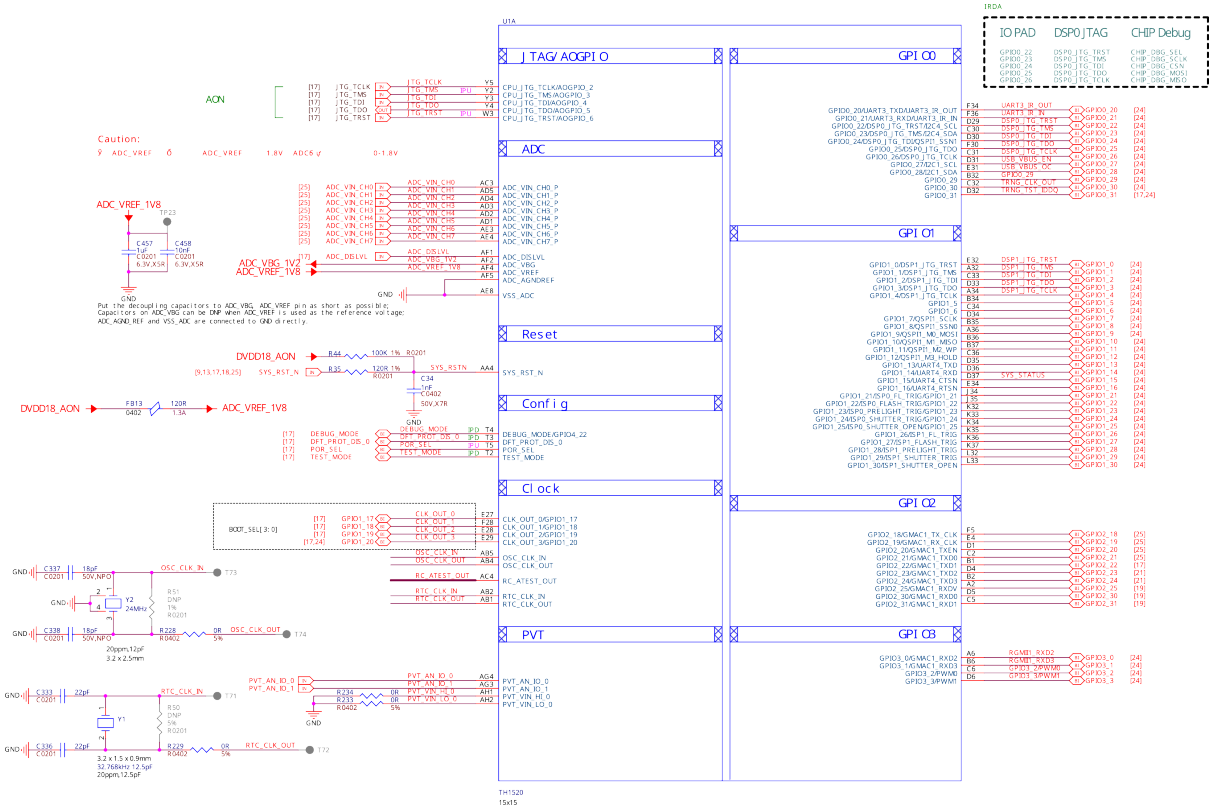


Fig. 3.7: SoC sys, ADC, and Clock

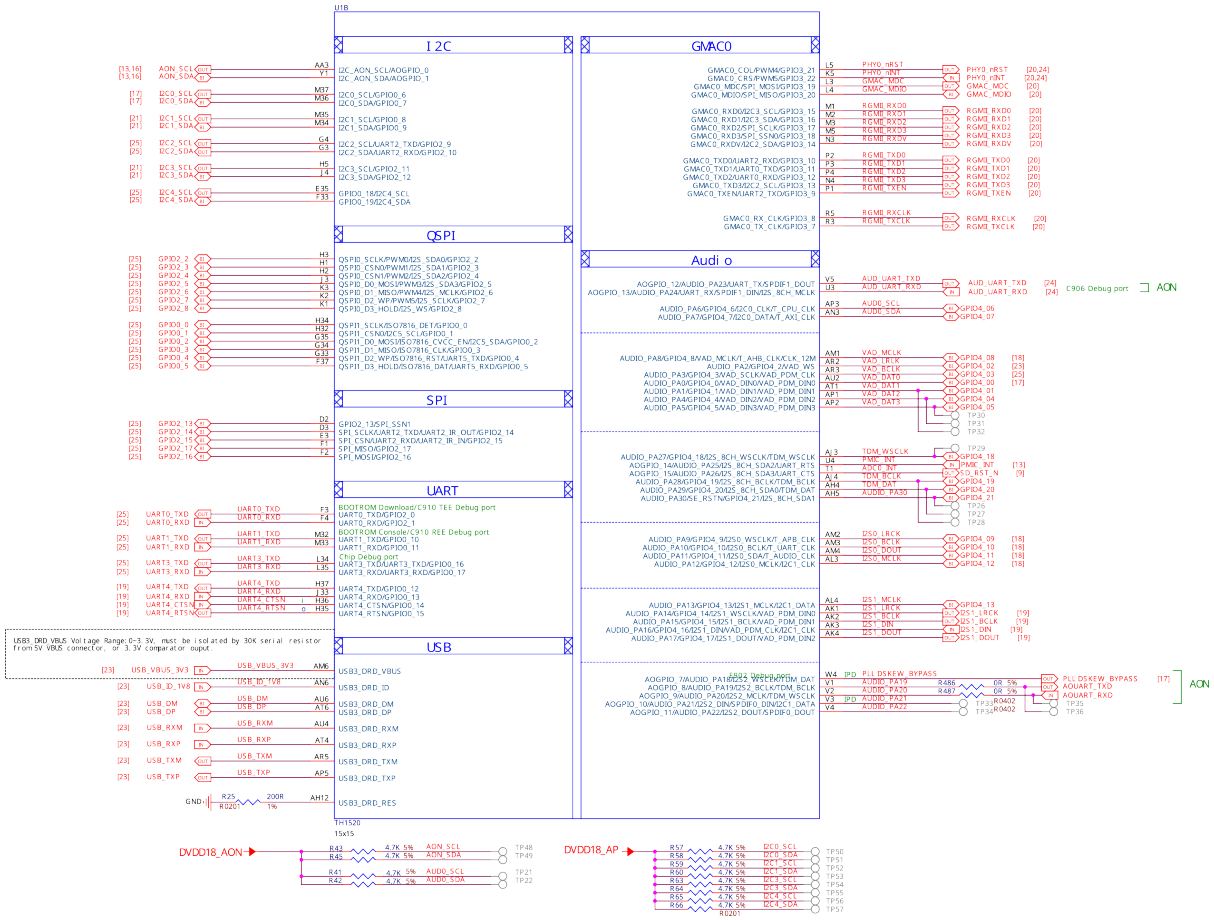


Fig. 3.8: SoC USB GMAC Audio

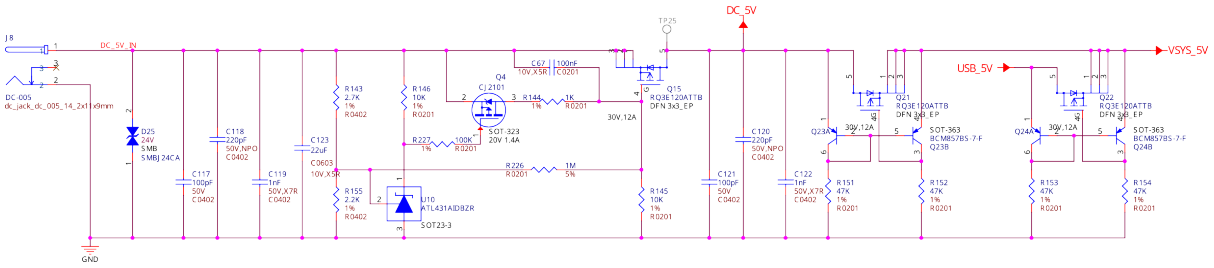


Fig. 3.9: Barrel jack power input

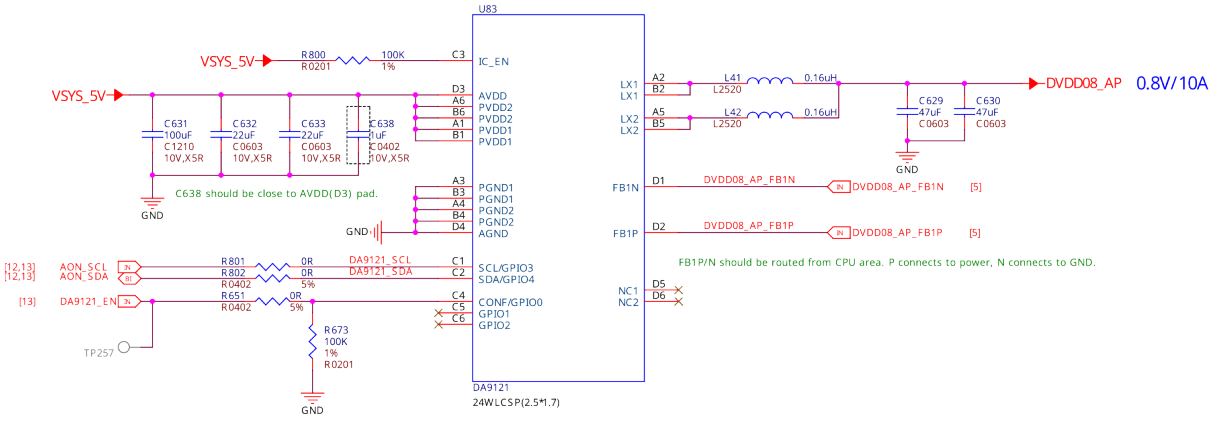


Fig. 3.10: 0.8V DCDC buck converter

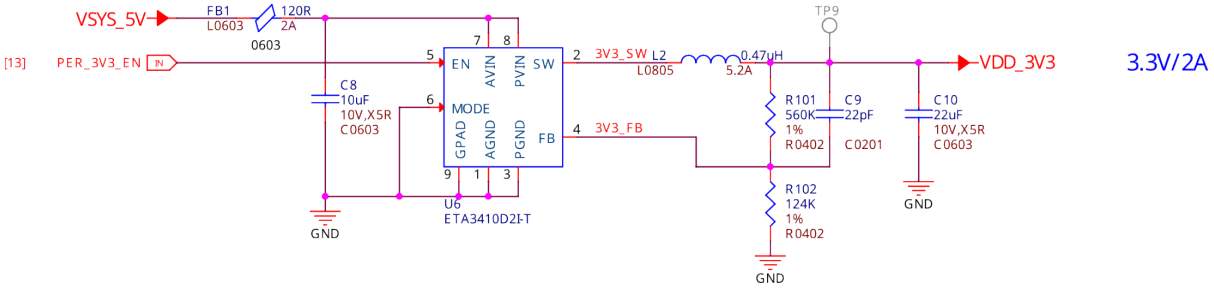


Fig. 3.11: 3.3V DCDC buck converter

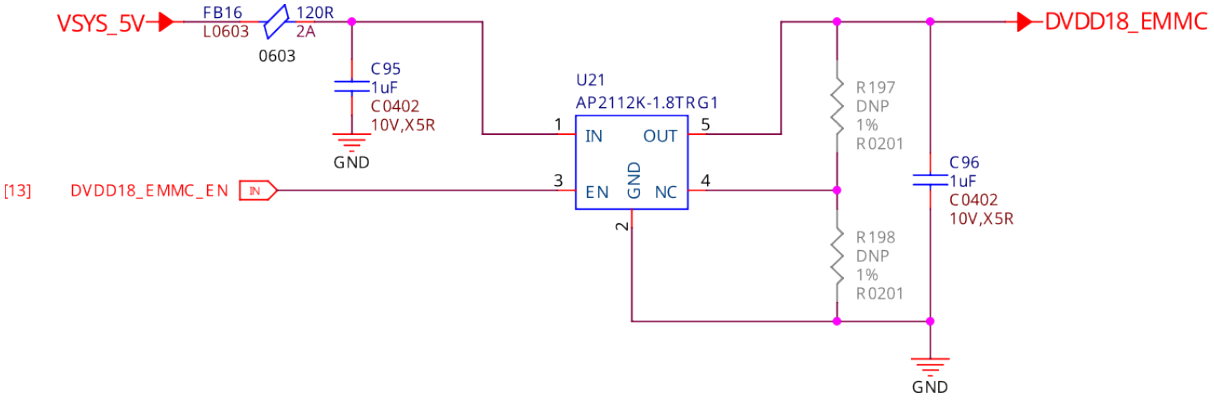


Fig. 3.12: 1.8V LDO regulator

3.3.5 PMIC

3.4 General Connectivity and Expansion

3.4.1 microUSB 3.0 port

3.4.2 P8 & P9 cape header pins

3.4.3 mikroBUS shuttle connector

3.4.4 P8, P9, and mikroBUS helper circuitry

3.5 Buttons and LEDs

3.5.1 Boot select buttons

3.5.2 User LEDs and Power LED

3.5.3 Power and reset button

3.6 Wired and wireless connectivity

3.6.1 Ethernet

3.6.2 WiFi & Bluetooth

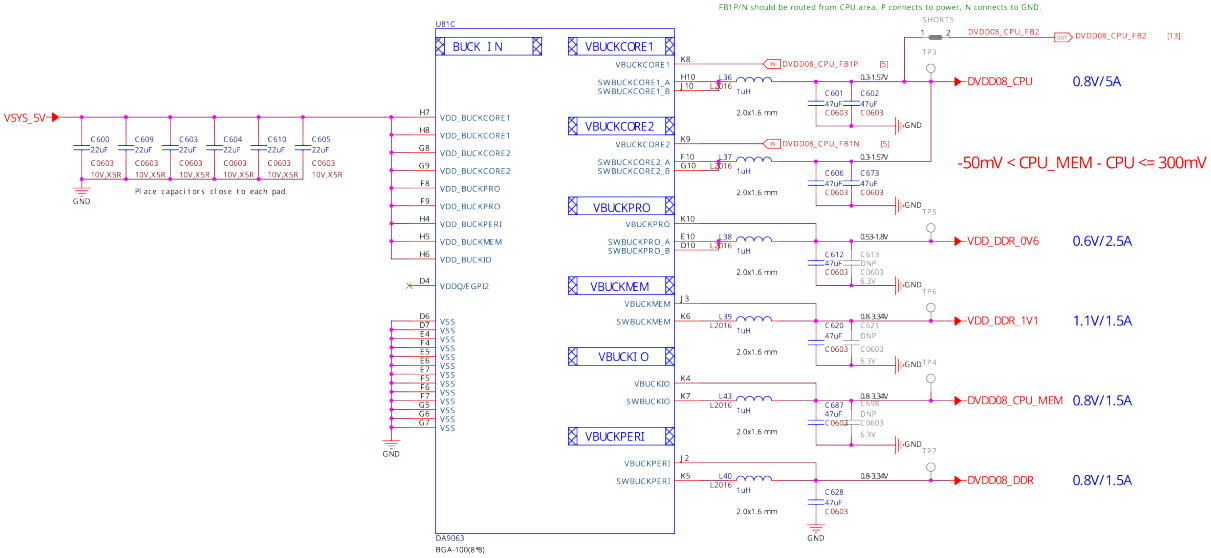


Fig. 3.13: PMIC Buck

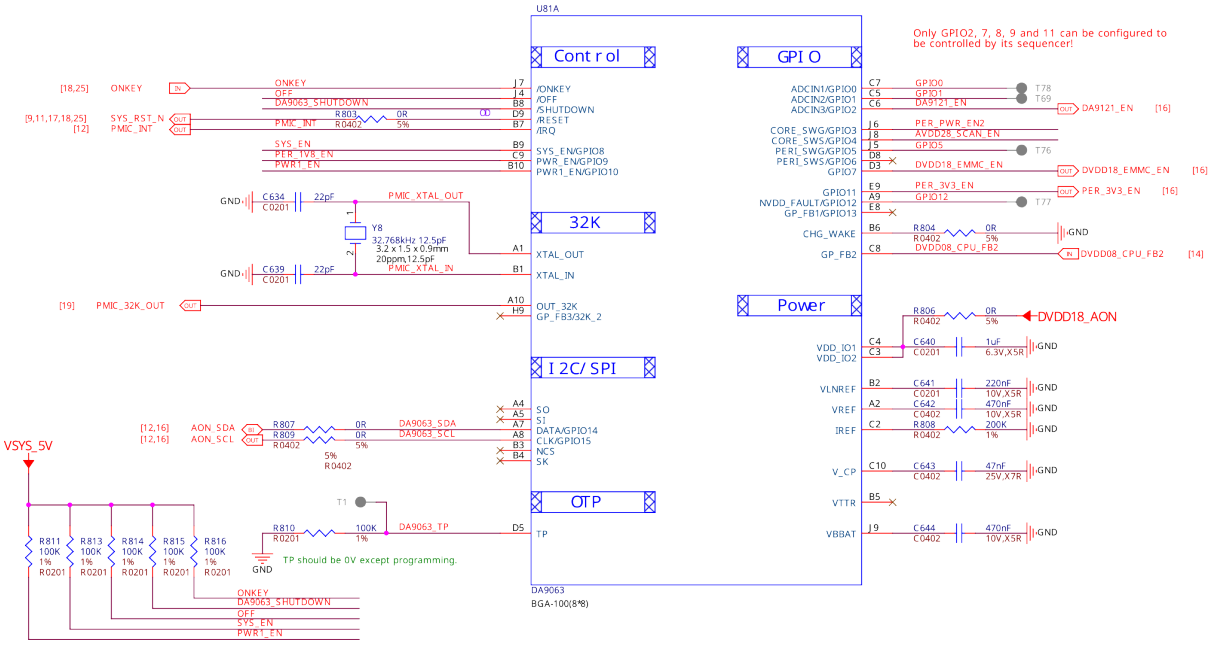


Fig. 3.14: PMIC Control

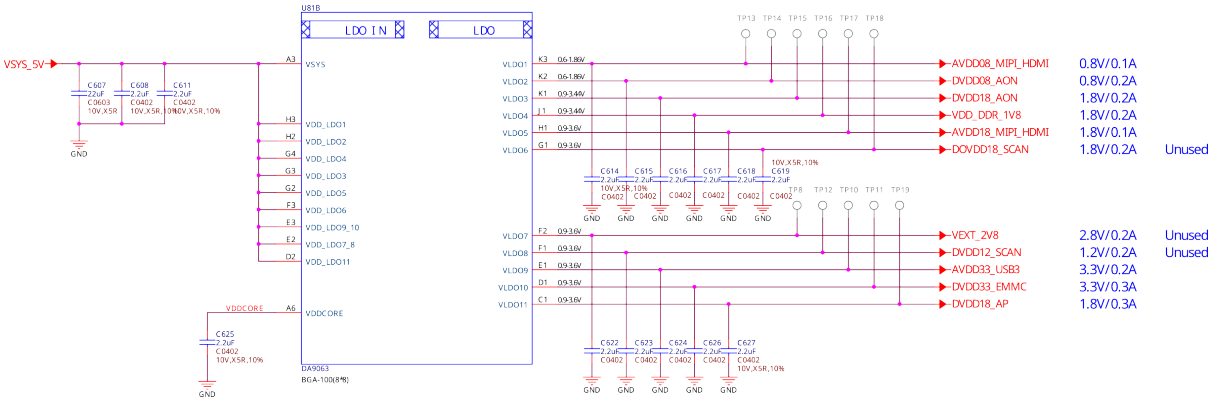


Fig. 3.15: PMIC LDO

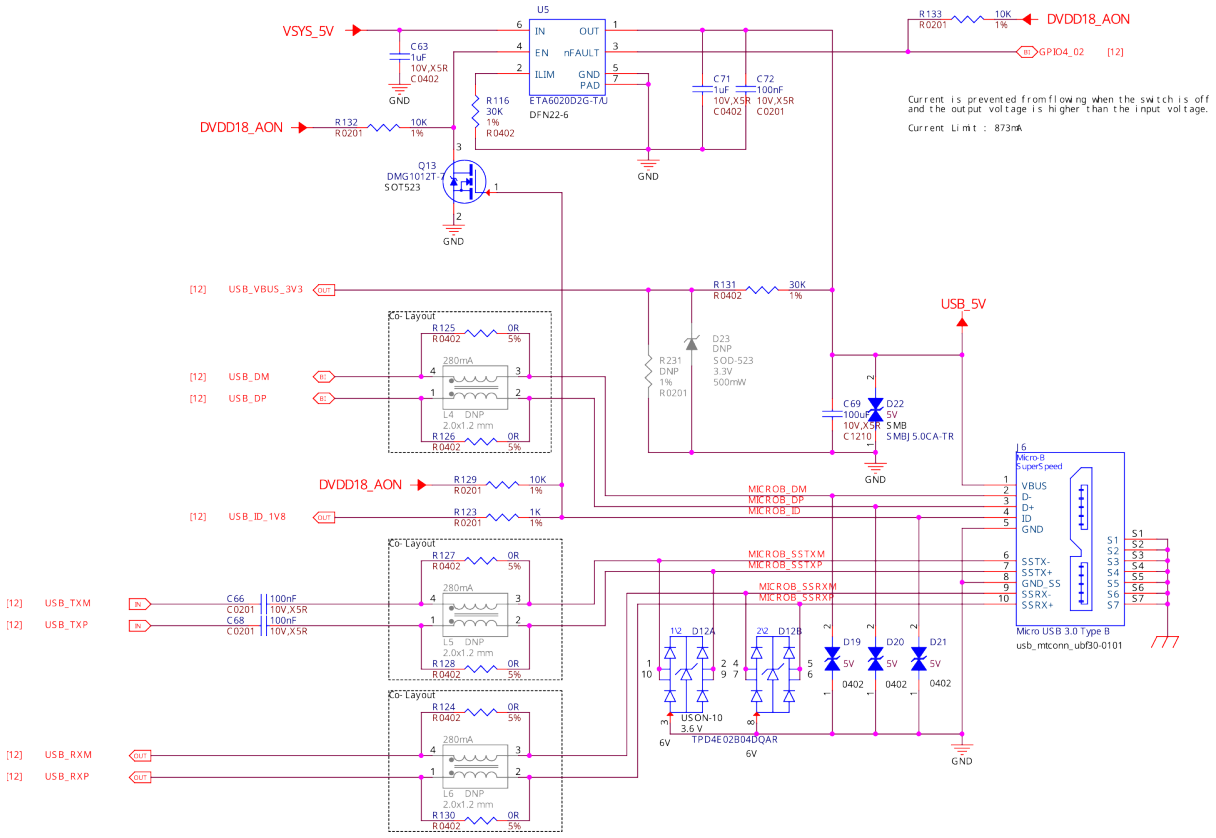


Fig. 3.16: microUSB 3.0 port

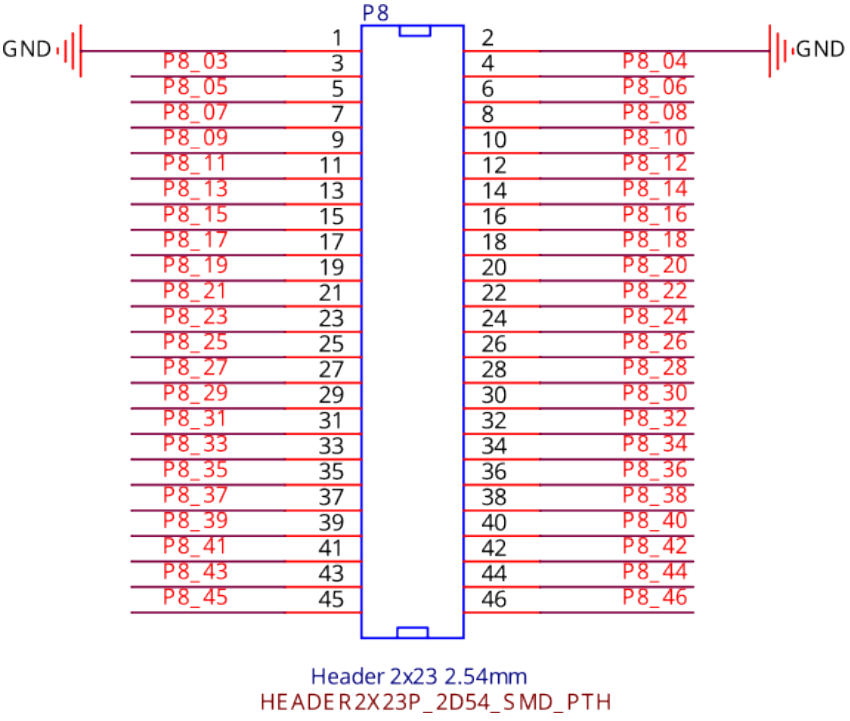


Fig. 3.17: P8 cape header

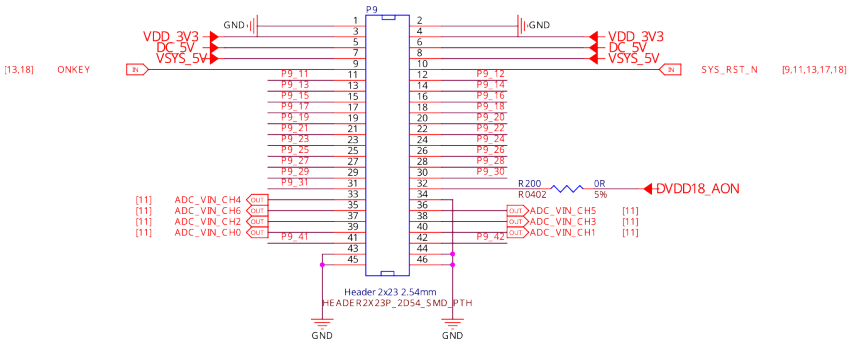
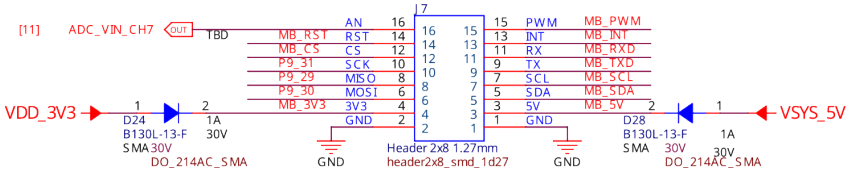
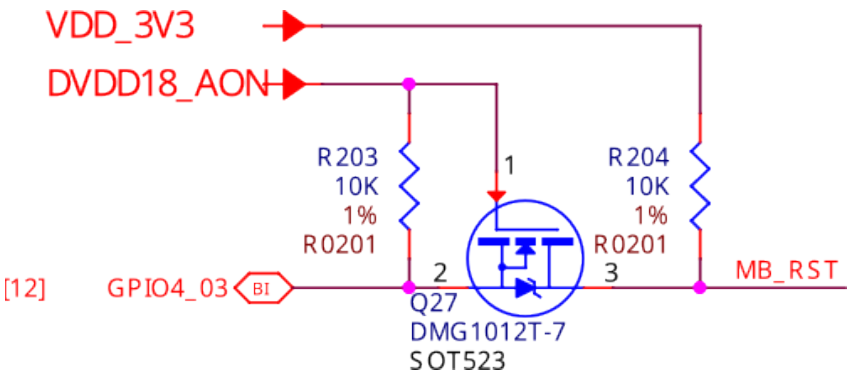


Fig. 3.18: P9 cape header





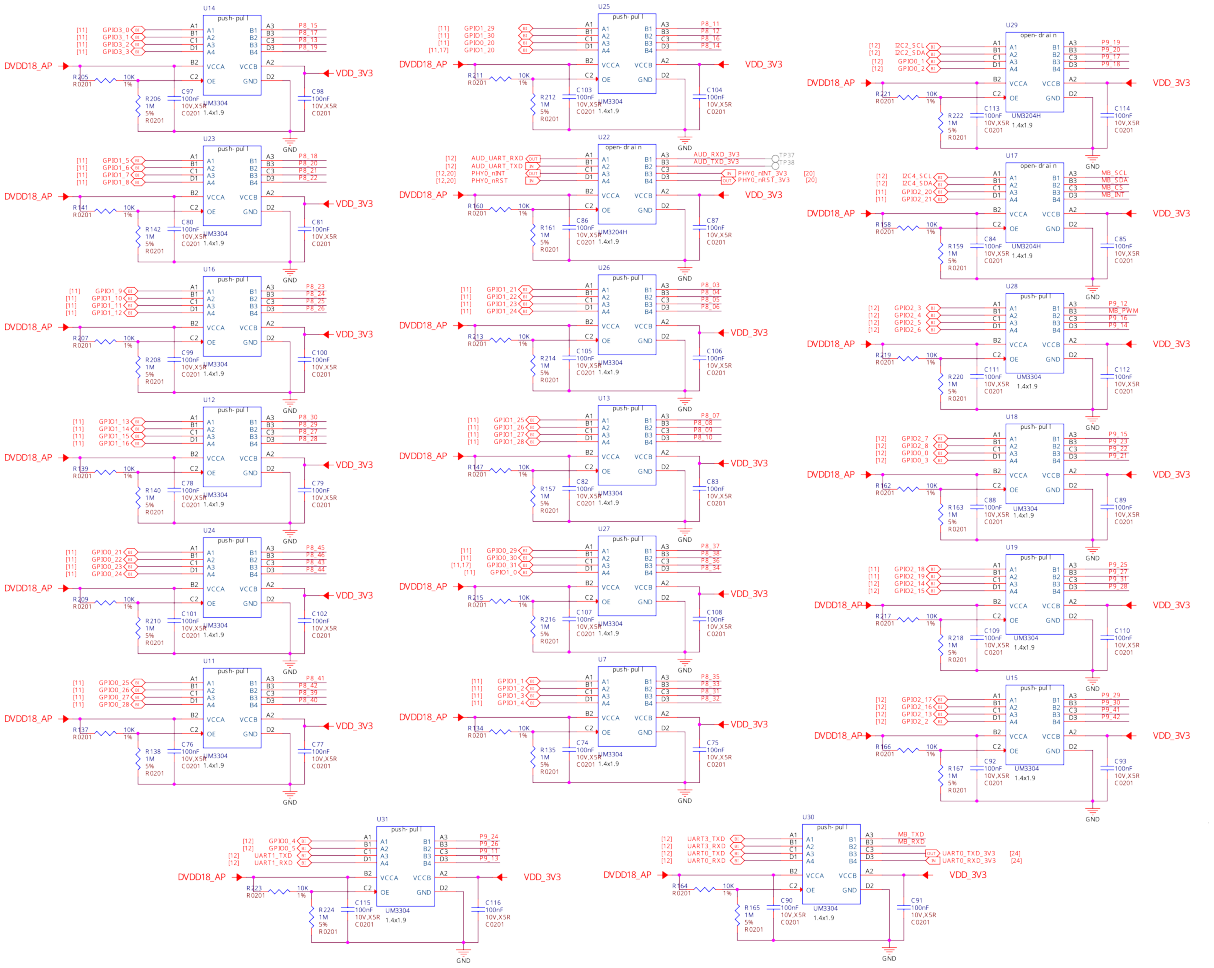
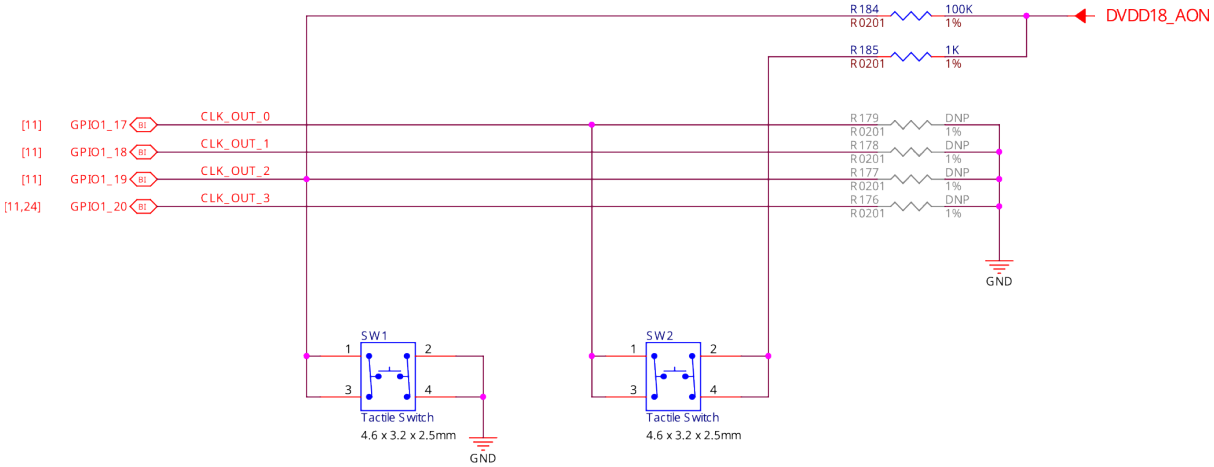


Fig. 3.19: P8, P9, and mikroBUS level shifters



CLK_OUT3	CLK_OUT2	CLK_OUT1	CLK_OUT0	Boot Mode Setting
X	0	X	X	USB Download
X	1	0	0	eMMC Boot
X	1	0	1	SD Boot, SDIO0
X	1	1	0	SPI NAND boot, QSPI0, CS0
X	1	1	1	SPI NOR Boot, QSPI0, CS0

Fig. 3.20: Boot select buttons

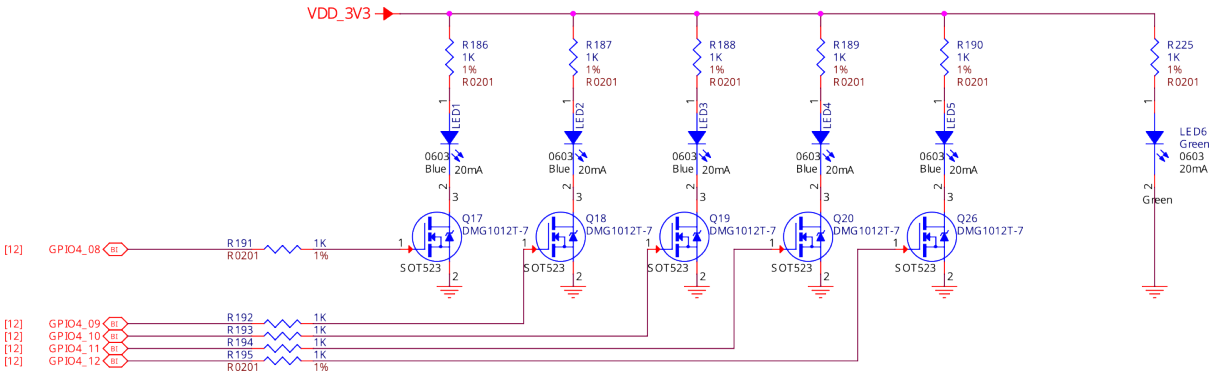


Fig. 3.21: User LEDs and power LED

Fig. 3.22: Power and reset button

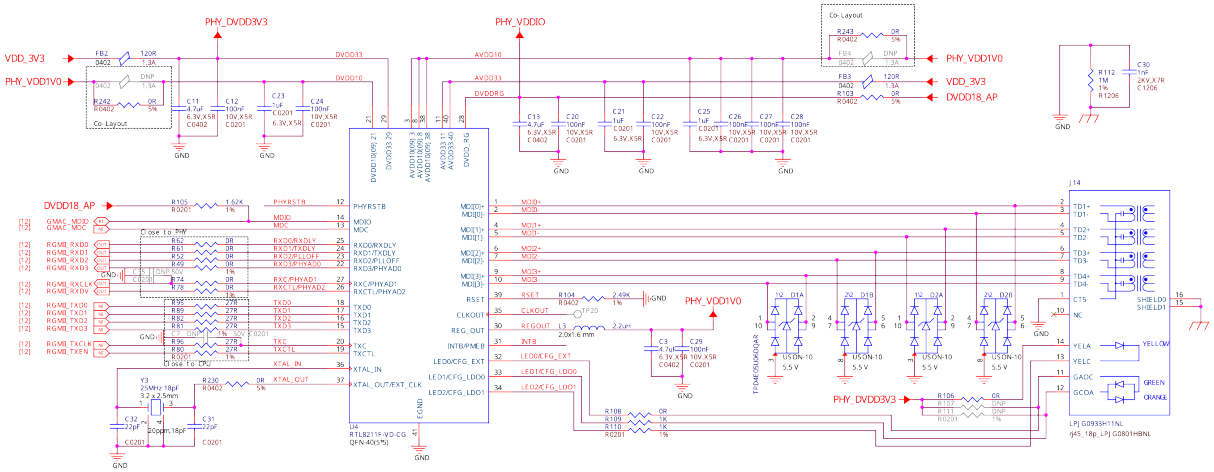


Fig. 3.23: Ethernet

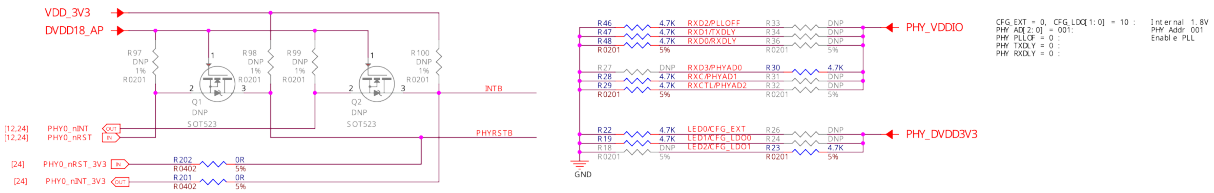


Fig. 3.24: Ethernet LevelShifter and Strapping

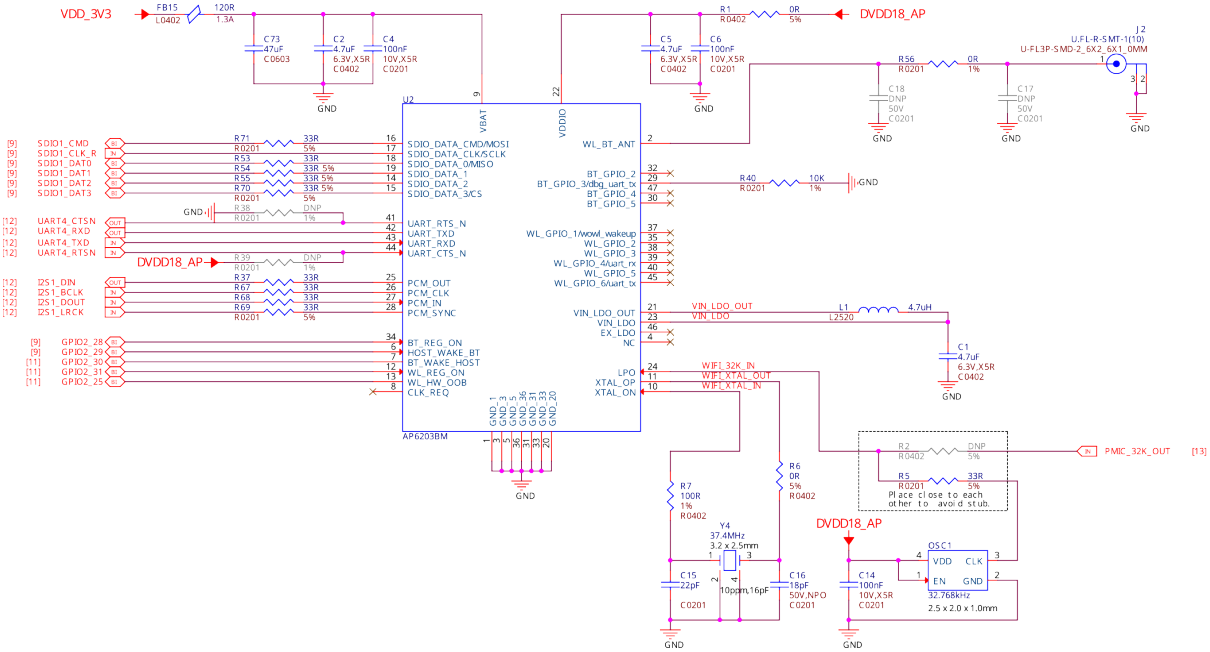


Fig. 3.25: WiFi and Bluetooth

3.7 Memory, Media and Data storage

3.7.1 DDR memory

3.7.2 eMMC

3.7.3 microSD

3.7.4 EEPROM

3.8 Multimedia I/O

3.8.1 CSI0

3.8.2 CSI1

3.8.3 DSI

3.8.4 CSI & DSI level shifter

3.8.5 HDMI

3.9 Debug

3.9.1 UART debug port

3.9.2 JTAG debug port

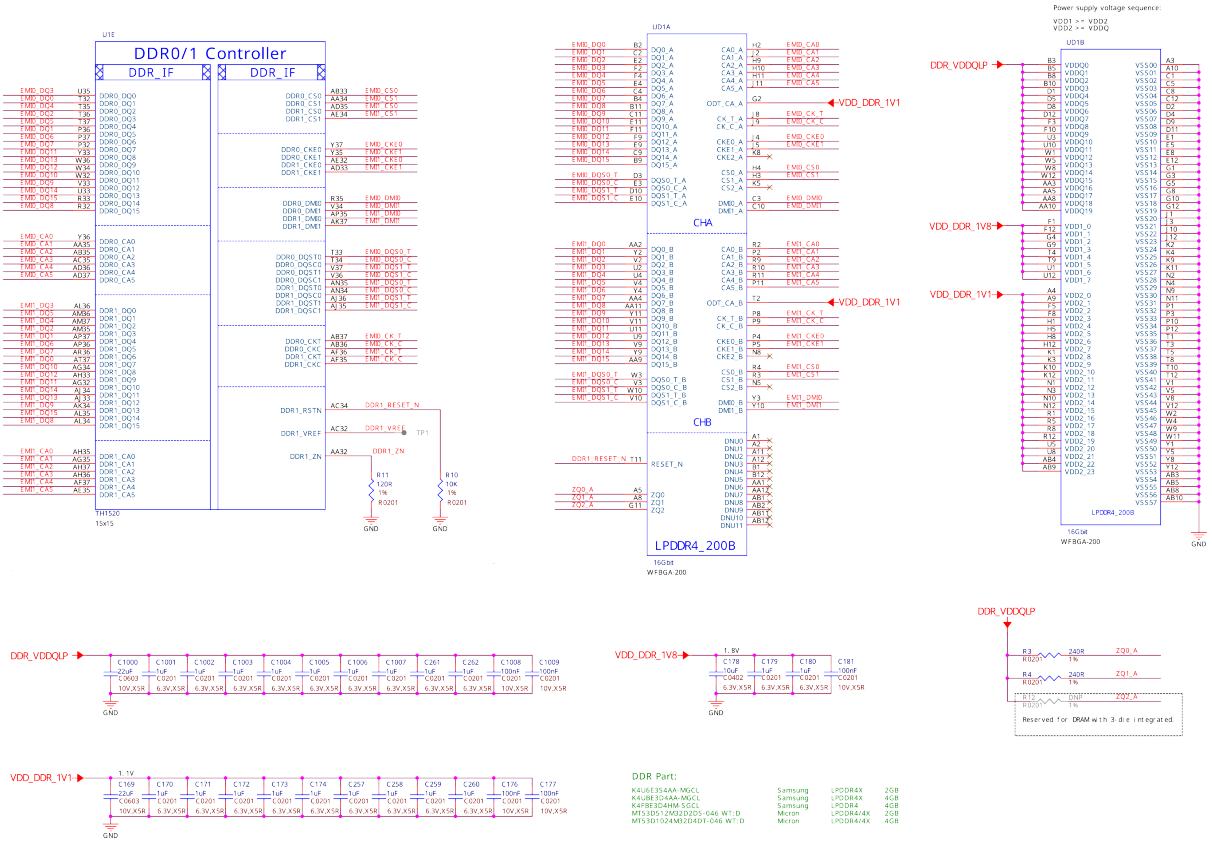


Fig. 3.26: 2GB DDR4 Memory chip1

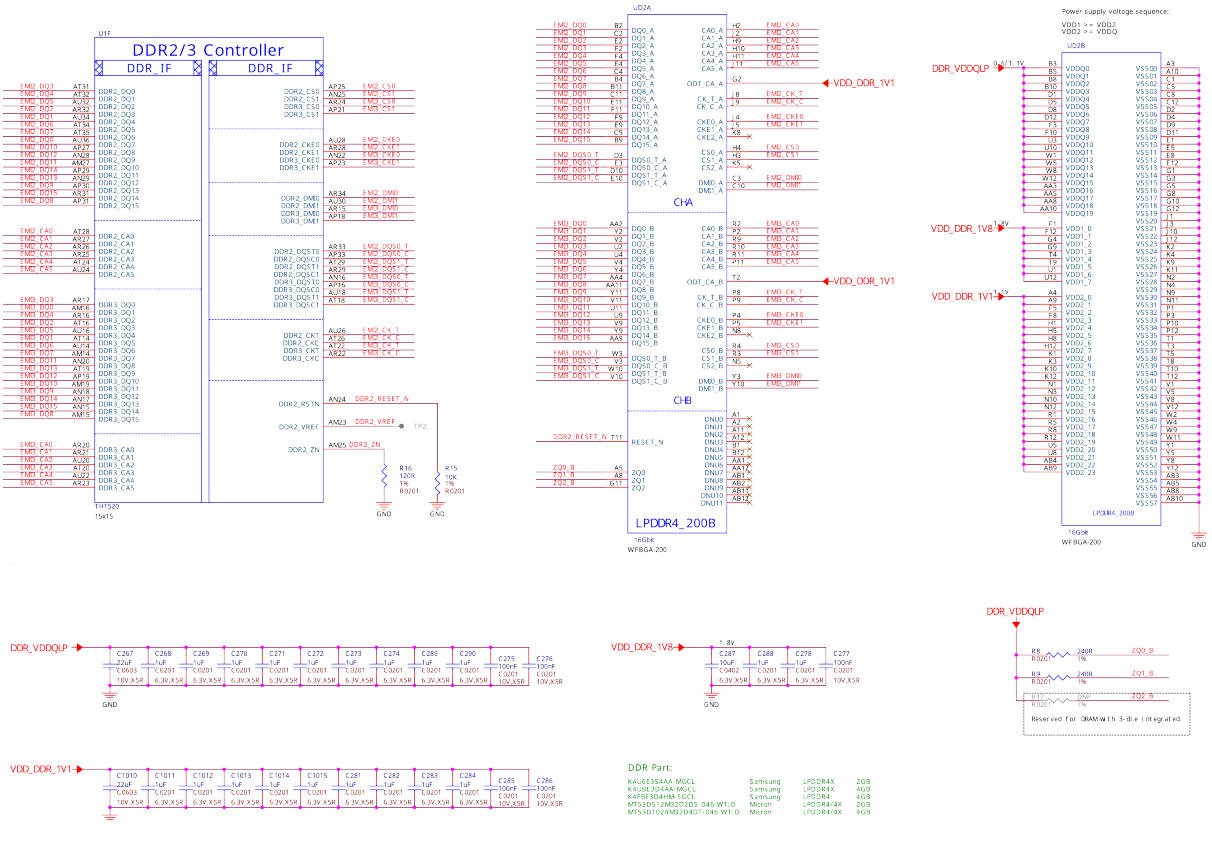


Fig. 3.27: 2GB DDR4 Memory chip2

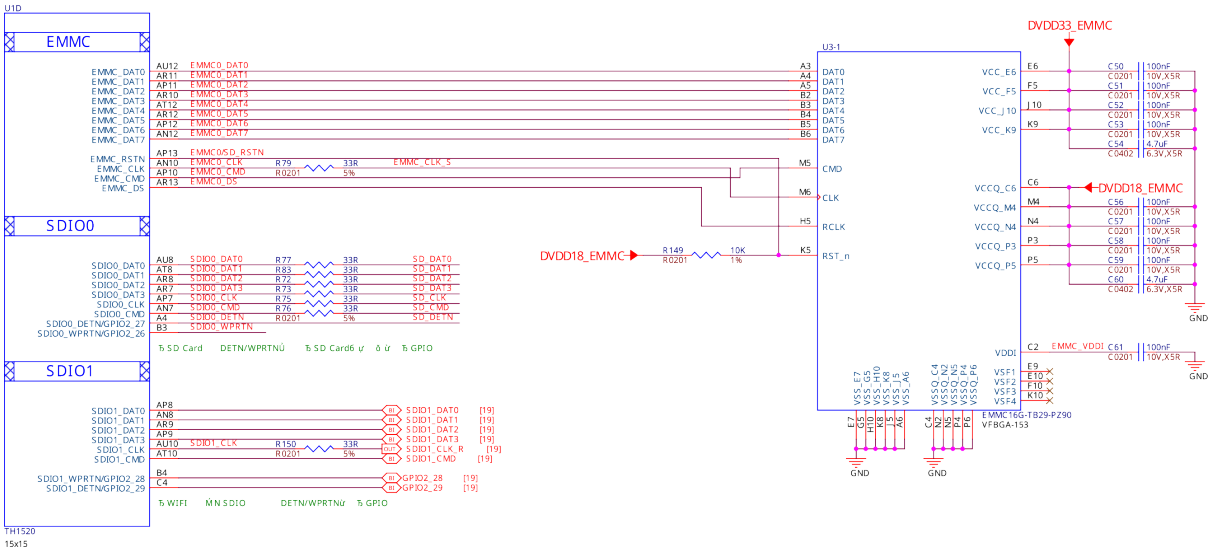


Fig. 3.28: 16GB eMMC

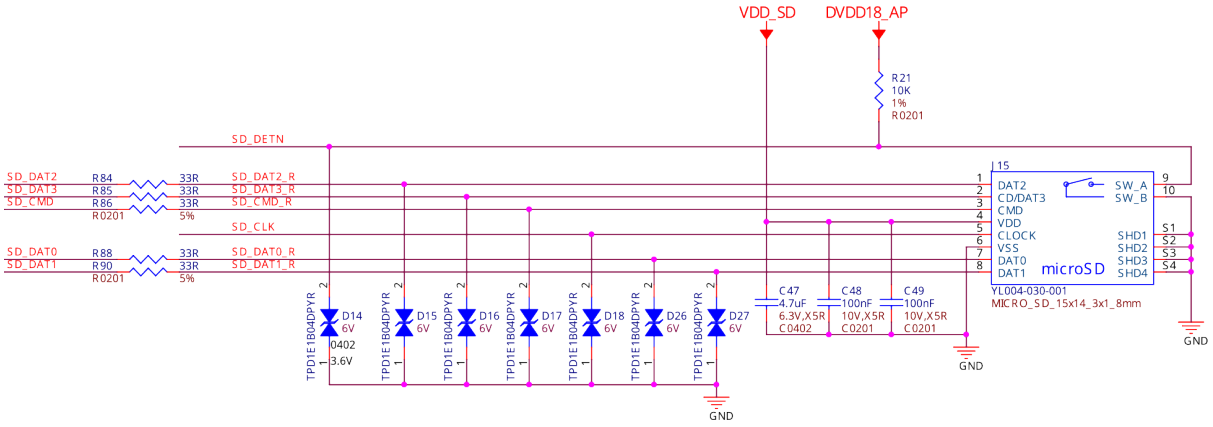


Fig. 3.29: microSD card connector

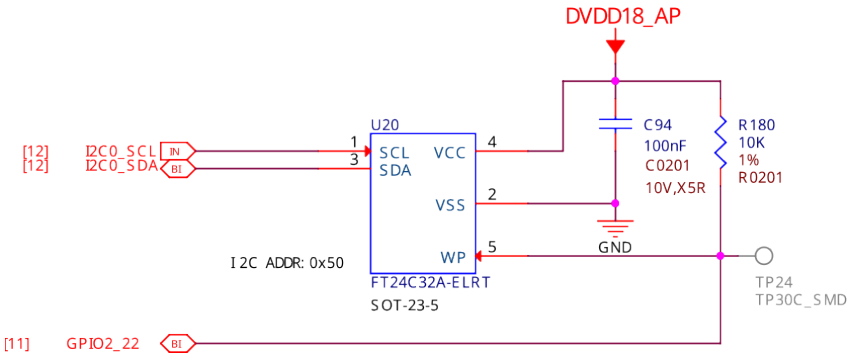


Fig. 3.30: 16GB EEPROM

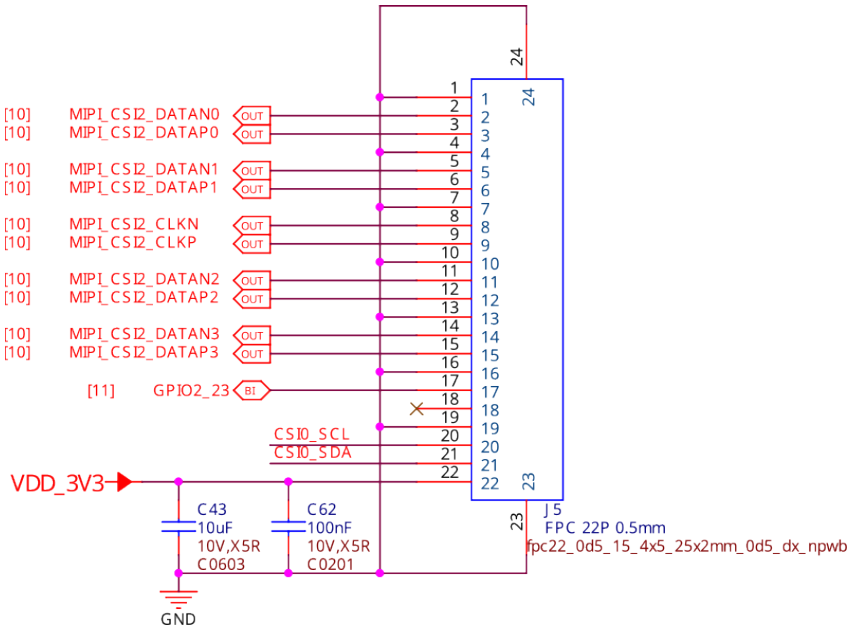


Fig. 3.31: CSI0 camera interface

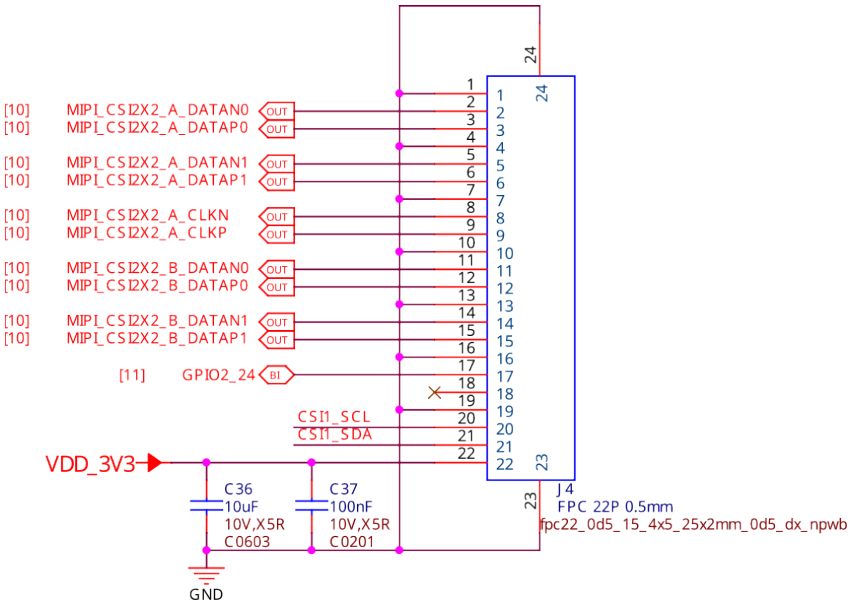


Fig. 3.32: CSI1 camera interface

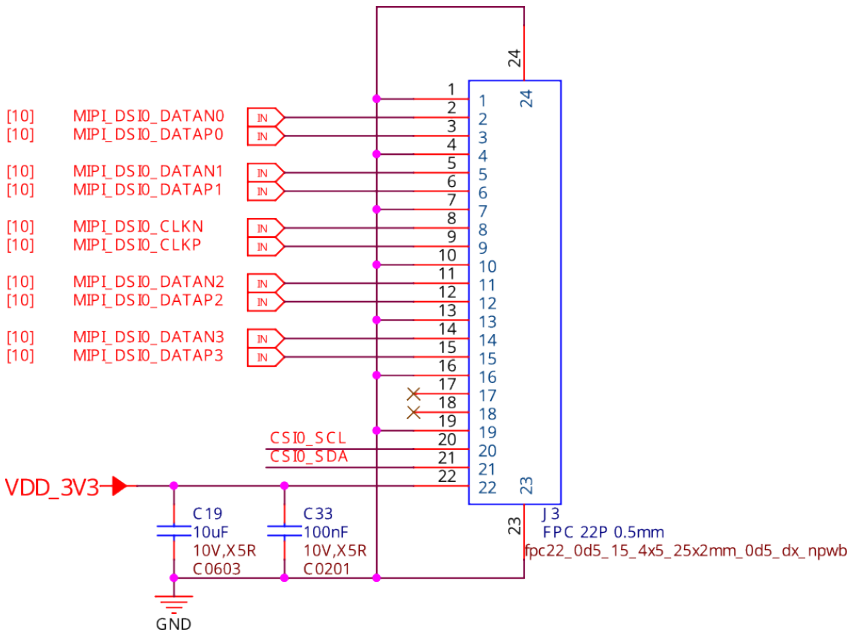


Fig. 3.33: DSI display interface

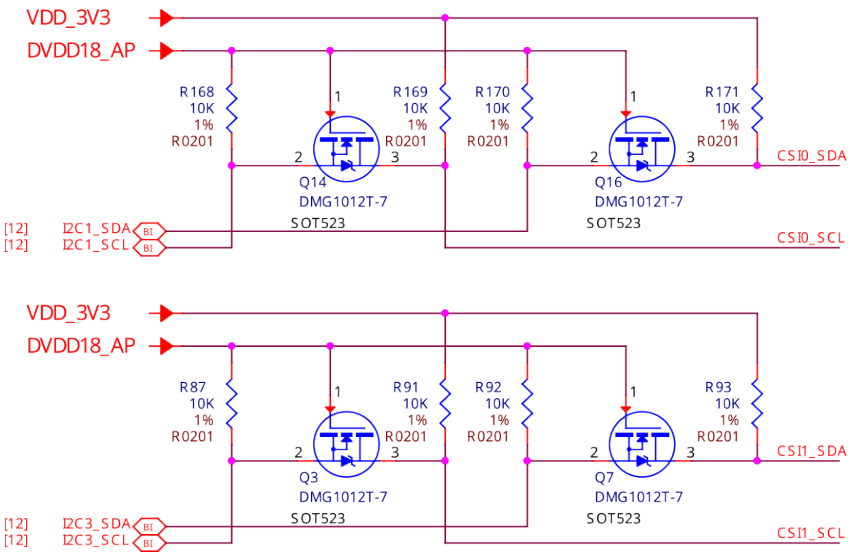


Fig. 3.34: CSI & DSI level shifter

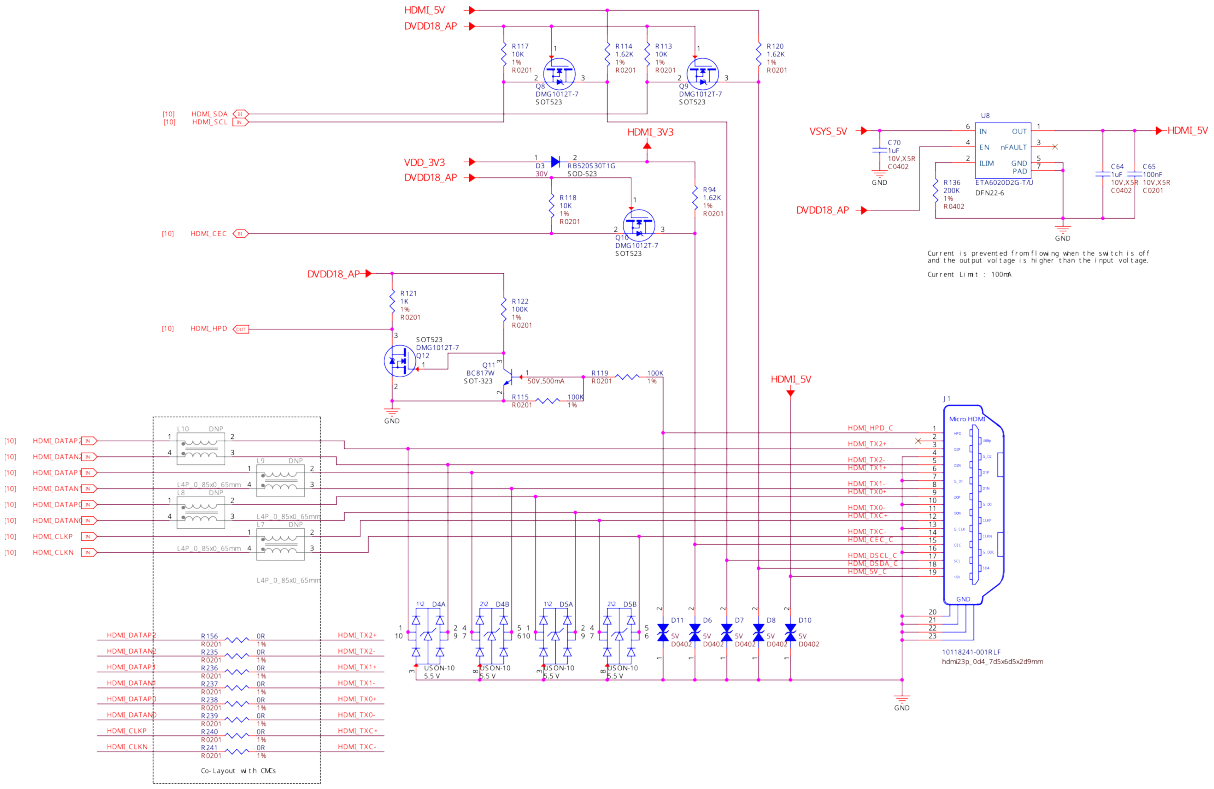


Fig. 3.35: HDMI display interface

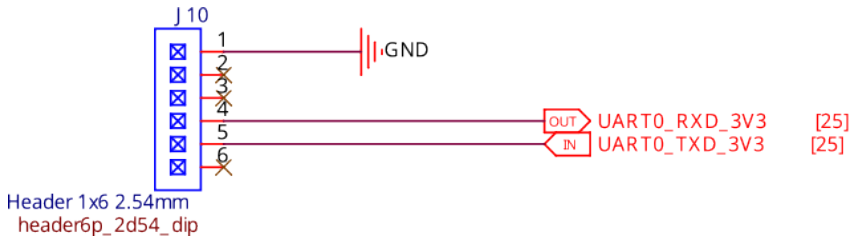


Fig. 3.36: UART Debug port

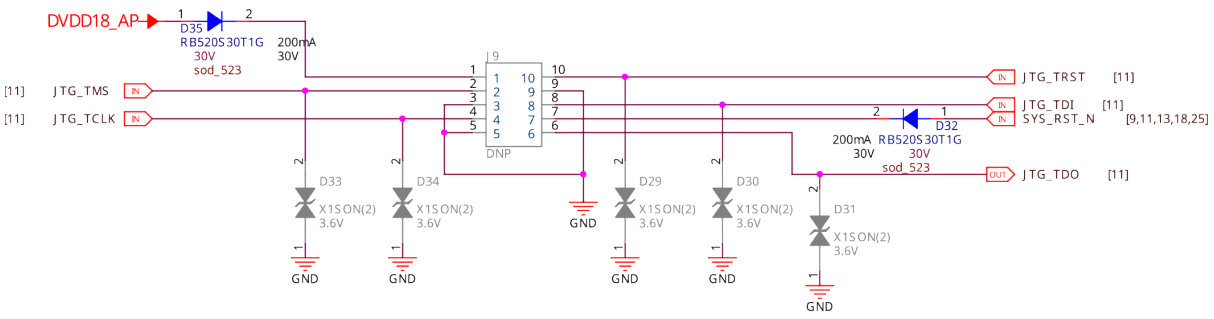
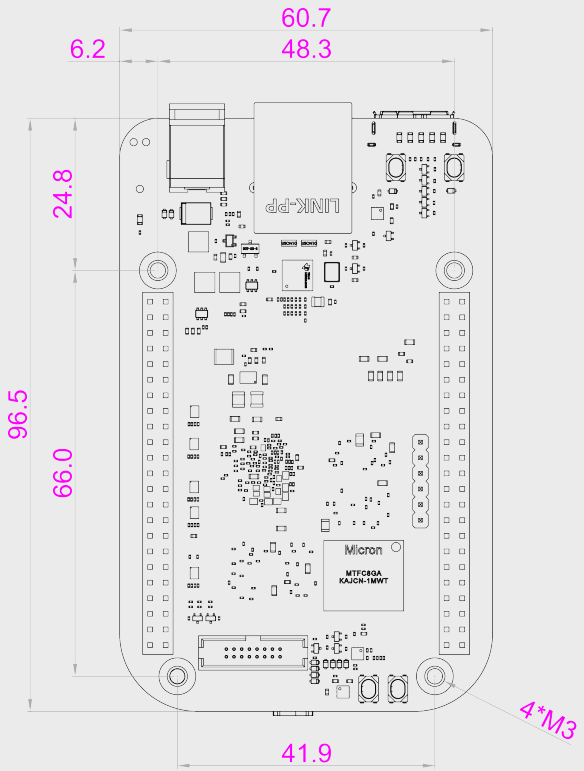
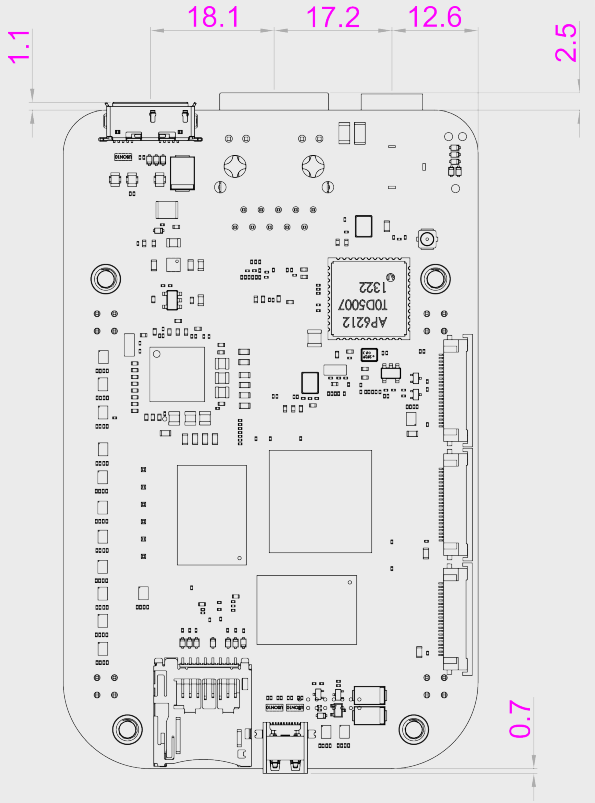


Fig. 3.37: JTAG debug port

3.10 Mechanical Specifications

Top	Bottom
	

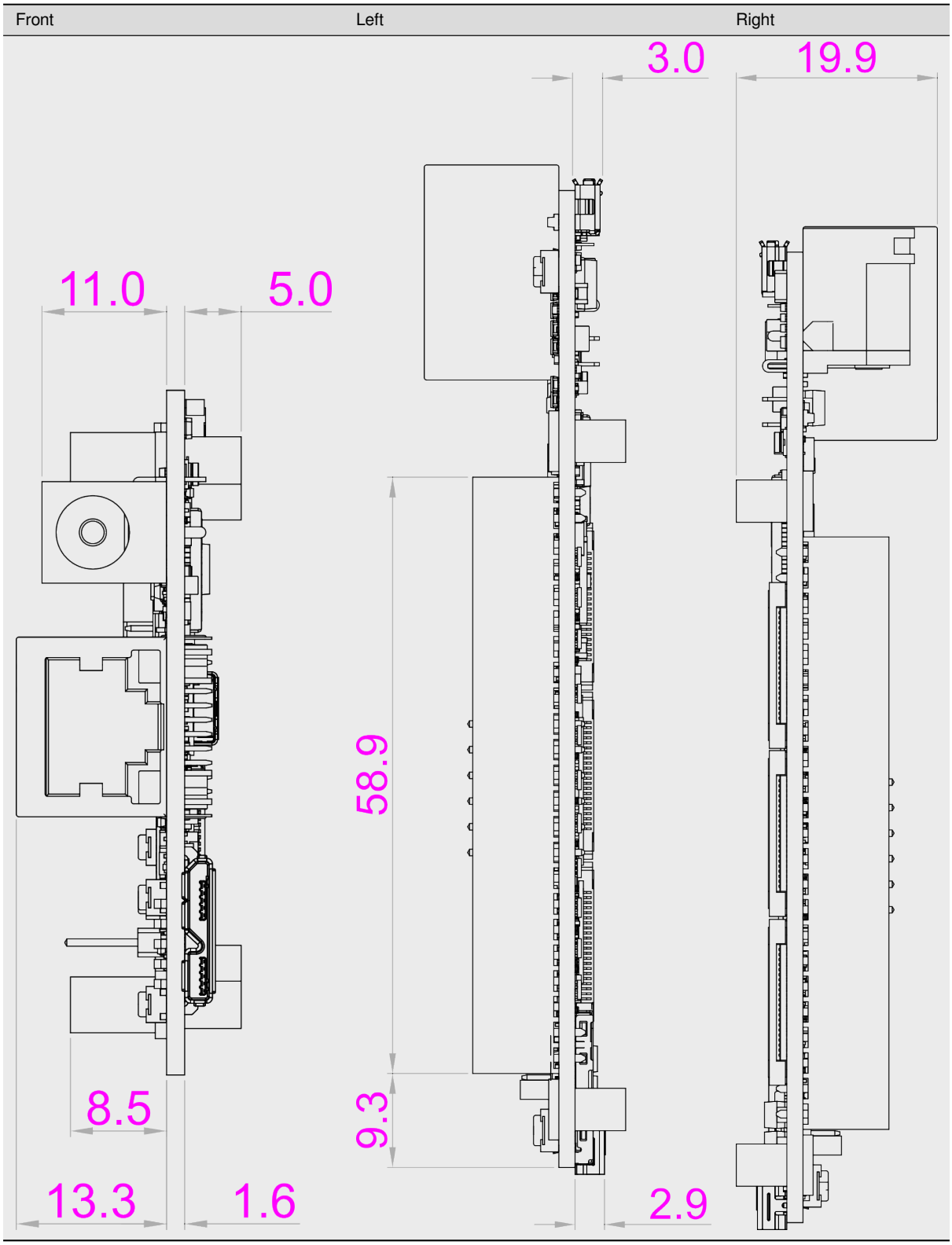

Front
Left
Right
3.0
19.9
11.0
5.0
58.9
9.3
8.5
13.3
1.6
2.9

Table 3.1: Dimensions & weight

Parameter	Values
Size	96.5×60.7×19.9mm
Max heigh	21.1mm
PCB Size	96.5x60.5*1.6mm
PCB Layers	10 layers
PCB Thickness	1.6mm
RoHS compliant	yes
Gross Weight	128.8g
Net weight	49.7g

Chapter 4

Expansion

4.1 Cape Headers

The expansion interface on the board is comprised of two headers P8 (46 pin) & P9 (46 pin). All signals on the expansion headers are **3.3V** unless otherwise indicated.

Note: Do not connect 5V logic level signals to these pins or the board will be damaged.

Note: DO NOT APPLY VOLTAGE TO ANY I/O PIN WHEN POWER IS NOT SUPPLIED TO THE BOARD. IT WILL DAMAGE THE PROCESSOR AND VOID THE WARRANTY.

NO PINS ARE TO BE DRIVEN UNTIL AFTER THE SYS_RESET LINE GOES HIGH.

4.1.1 Connector P8

The following tables show the pinout of the **P8** expansion header. The SW is responsible for setting the default function of each pin. Refer to the processor documentation for more information on these pins and detailed descriptions of all of the pins listed. In some cases there may not be enough signals to complete a group of signals that may be required to implement a total interface.

The column heading is the pin number on the expansion header.

The **GPIO** row is the expected gpio identifier number in the Linux kernel.

Each row includes the gpiochipX and pinY in the format of *X Y*. You can use these values to directly control the GPIO pins with the commands shown below.

```
# to set the GPIO pin state to HIGH
debian@BeagleBone:~$ gpioset X Y=1

# to set the GPIO pin state to LOW
debian@BeagleBone:~$ gpioset X Y=0

For Example:

+---------+----------+
| Pin     | P8.03    |
+=========+==========+
| GPIO    | 1 20     |
+---------+----------+
```

(continues on next page)

(continued from previous page)

```
Use the commands below for controlling this pin (P8.03) where X = 1 and Y =␣
↪20

# to set the GPIO pin state to HIGH
debian@BeagleBone:~$ gpioset 1 20=1

# to set the GPIO pin state to LOW
debian@BeagleBone:~$ gpioset 1 20=0
```

The **BALL** row is the pin number on the processor.

The **REG** row is the offset of the control register for the processor pin.

The **MODE #** rows are the mode setting for each pin. Setting each mode to align with the mode column will give that function on that pin.

NOTES:

DO NOT APPLY VOLTAGE TO ANY I/O PIN WHEN POWER IS NOT SUPPLIED TO THE BOARD. IT WILL DAMAGE THE PROCESSOR AND VOID THE WARRANTY.

NO PINS ARE TO BE DRIVEN UNTIL AFTER THE SYS_RESET LINE GOES HIGH.

P8.01-P8.02

P8.01	P8.02
GND	GND

P8.03-P8.05

Pin	P8.03	P8.04	P8.05
Name	GPIO1_21	GPIO1_22	GPIO1_23
BALL	J34	J35	K32
GPIO	1 21	1 22	1 23
REG	GPIO1_21_MUX	GPIO1_22_MUX	GPIO1_23_MUX
Mode 0	GPIO1_21	GPIO1_22	GPIO1_23
MODE 1	~	~	~
MODE 2	ISP0_FL_TRIG	ISP0_FLASH_TRIG	ISP0_PRELIGHT_TRIG
MODE 3	GPIO1_21	GPIO1_22	GPIO1_23
MODE 4	~	~	~
MODE 5	~	~	~

P8.06-P8.09

Pin	P8.06	P8.07	P8.08	P8.09
Name	GPIO1_24	GPIO1_25	GPIO1_26	GPIO1_27
BALL	K33	K34	K35	K36
GPIO	1 24	1 25	1 26	1 27
REG	GPIO1_24_MUX	GPIO1_25_MUX	GPIO1_26_MUX	GPIO1_27_MUX
Mode 0	GPIO1_24	GPIO1_25	GPIO1_26	GPIO1_27
MODE 1	~	~	~	~
MODE 2	ISP0_SHUTTER_TRIG	ISP0_SHUTTER_OPEN	ISP1_FL_TRIG	ISP1_FLASH_TRIG
MODE 3	GPIO1_24	GPIO1_25	~	~
MODE 4	~	~	~	~
MODE 5	~	~	~	~

P8.10-P8.13

Pin	P8.10	P8.11	P8.12	P8.13
Name	GPIO1_28	GPIO1_29	GPIO1_30	GPIO3_2
BALL	K37	L32	L33	C6
GPIO	1 28	1 29	1 30	3 2
REG	GPIO1_28_MUX	GPIO1_29_MUX	GPIO1_30_MUX	GPIO3_2_MUX
MODE 0	GPIO1_28	GPIO1_29	GPIO1_30	GPIO3_2
MODE 1	~	~	~	PWM0
MODE 2	ISP1_PRELIGHT_TRIG	ISP1_SHUTTER_TRIG	ISP1_SHUTTER_OPEN	~
MODE 3	~	~	~	~
MODE 4	~	~	~	~
MODE 5	~	~	~	~

P8.14-P8.16

Pin	P8.14	P8.15	P8.16
Name	CLK_OUT_3	GPIO3_0	GPIO0_20
BALL	E29	A6	F34
GPIO	1 20	3 0	0 20
REG	CLK_OUT_3_MUX	GPIO3_0_MUX	GPIO0_20_MUX
MODE 0	BOOT_SEL3	GPIO3_0	GPIO0_20
MODE 1	CLK_OUT_3	GMAC1_RXD2	UART3_TXD
MODE 2	~	~	UART3_IR_OUT
MODE 3	GPIO1_20	~	~
MODE 4	~	~	~
MODE 5	~	~	~

P8.17-P8.19

Pin	P8.17	P8.18	P8.19
Name	GPIO3_1	GPIO1_5	GPIO3_3
BALL	B6	B34	D6
GPIO	3 1	1 5	3 3
REG	GPIO3_1_MUX	GPIO1_5_MUX	GPIO3_3_MUX
MODE 0	GPIO3_1	GPIO1_5	GPIO3_3
MODE 1	GMAC1_RXD3	~	PWM1
MODE 2	~	~	~
MODE 3	~	~	~
MODE 4	~	DPU_COLOR_16	~
MODE 5	~	DPU1_COLOR_16	~

P8.20-P8.22

Pin	P8.20	P8.21	P8.22
Name	GPIO1_6	GPIO1_7	GPIO1_8
BALL	C34	D34	B35
GPIO	1 6	1 7	1 8
REG	GPIO1_6_MUX	GPIO1_7_MUX	GPIO1_8_MUX
MODE 0	GPIO1_6	GPIO1_7	GPIO1_8
MODE 1	~	QSPI1_SCLK	QSPI1_SSN0
MODE 2	~	~	~
MODE 3	~	~	~
MODE 4	DPU_COLOR_17	DPU_COLOR_18	DPU_COLOR_19
MODE 5	DPU1_COLOR_17	DPU1_COLOR_18	DPU1_COLOR_19

P8.23-P8.26

Pin	P8.23	P8.24	P8.25	P8.26
Name	GPIO1_9	GPIO1_10	GPIO1_11	GPIO1_12
BALL	A36	B36	B37	C36
GPIO	1 9	1 10	1 11	1 12
REG	GPIO1_9_MUX	GPIO1_10_MUX	GPIO1_11_MUX	GPIO1_12_MUX
MODE 0	GPIO1_9	GPIO1_10	GPIO1_11	GPIO1_12
MODE 1	QSPI1_M0_MOSI	QSPI1_M1_MISO	QSPI1_M2_WP	QSPI1_M3_HOLD
MODE 2	~	~	~	~
MODE 3	~	~	~	~
MODE 4	DPU_COLOR_20	DPU_COLOR_21	DPU_COLOR_22	DPU_COLOR_23
MODE 5	DPU1_COLOR_20	DPU1_COLOR_21	DPU1_COLOR_22	DPU1_COLOR_23

P8.27-P8.29

Pin	P8.27	P8.28	P8.29
Name	GPIO1_15	GPIO1_16	GPIO1_14
BALL	D37	E34	D36
GPIO	1 15	1 16	1 14
REG	GPIO1_15_MUX	GPIO1_16_MUX	GPIO1_14_MUX
MODE 0	GPIO1_15	GPIO1_16	GPIO1_14
MODE 1	UART4_CTSN	UART4_RTSN	UART4_RXD
MODE 2	~	~	~
MODE 3	~	~	~
MODE 4	DPU_VSYNC	DPU_PIXELCLK	DPU_HSYNC
MODE 5	DPU1_VSYNC	DPU1_PIXELCLK	DPU1_HSYNC

P8.30-P8.32

Pin	P8.30	P8.31	P8.32
Name	GPIO1_13	GPIO1_3	GPIO1_4
BALL	D35	D33	A34
GPIO	1 13	1 3	1 4
REG	GPIO1_13_MUX	GPIO1_3_MUX	GPIO1_4_MUX
MODE 0	GPIO1_13	GPIO1_3	GPIO1_4
MODE 1	UART4_TXD	DSP1_JTG_TDO	DSP1_JTG_TCLK
MODE 2	~	~	~
MODE 3	~	~	~
MODE 4	DPU_COLOR_EN	DPU_COLOR_14	DPU_COLOR_15
MODE 5	DPU1_COLOR_EN	DPU1_COLOR_14	DPU1_COLOR_15

P8.33-P8.35

Pin	P8.33	P8.34	P8.35
Name	GPIO1_2	GPIO1_0	GPIO1_1
BALL	C33	E32	A32
GPIO	1 2	1 0	1 1
REG	GPIO1_2_MUX	GPIO1_0_MUX	GPIO1_1_MUX
MODE 0	GPIO1_2	GPIO1_0	GPIO1_1
MODE 1	DSP1_JTG_TDI	DSP1_JTG_TRST	DSP1_JTG_TMS
MODE 2	~	~	~
MODE 3	~	~	~
MODE 4	DPU_COLOR_13	DPU_COLOR_11	DPU_COLOR_12
MODE 5	DPU1_COLOR_13	DPU1_COLOR_11	DPU1_COLOR_12

P8.36-P8.38

Pin	P8.36	P8.37	P8.38
Name	GPIO0_31	GPIO0_29	GPIO0_30
BALL	D32	B32	C32
GPIO	0 31	0 29	0 30
REG	GPIO0_31_MUX	GPIO0_29_MUX	GPIO0_30_MUX
MODE 0	GPIO0_31	GPIO0_29	GPIO0_30
MODE 1	~	~	~
MODE 2	~	~	~
MODE 3	~	~	~
MODE 4	DPU_COLOR_10	DPU_COLOR_8	DPU_COLOR_9
MODE 5	DPU1_COLOR_10	DPU1_COLOR_8	DPU1_COLOR_9

P8.39-P8.41

Pin	P8.39	P8.40	P8.41
Name	GPIO0_27	GPIO0_28	GPIO0_25
BALL	D31	E31	F30
GPIO	0 27	0 28	0 25
REG	GPIO0_27_MUX	GPIO0_28_MUX	GPIO0_25_MUX
MODE 0	GPIO0_27	GPIO0_28	GPIO0_25
MODE 1	~	~	DSP0_JTG_TDO
MODE 2	I2C1_SCL	I2C1_SDA	~
MODE 3	~	~	~
MODE 4	DPU_COLOR_6	DPU_COLOR_7	DPU_COLOR_4
MODE 5	DPU1_COLOR_6	DPU1_COLOR_7	DPU1_COLOR_4

P8.42-P8.44

Pin	P8.42	P8.43	P8.44
Name	GPIO0_26	GPIO0_23	GPIO0_24
BALL	C31	C30	D30
GPIO	0 26	0 23	0 24
REG	GPIO0_26_MUX	GPIO0_23_MUX	GPIO0_24_MUX
MODE 0	GPIO0_26	GPIO0_23	GPIO0_24
MODE 1	DSP0_JTG_TCLK	DSP0_JTG_TMS	DSP0_JTG_TDI
MODE 2	~	I2C4_SDA	QSPI1_SSN1
MODE 3	~	~	~
MODE 4	DPU_COLOR_5	DPU_COLOR_2	DPU_COLOR_3
MODE 5	DPU1_COLOR_5	DPU1_COLOR_2	DPU1_COLOR_3

P8.45-P8.46

Pin	P8.45	P8.46
Name	GPIO0_21	GPIO0_22
BALL	F36	D29
GPIO	0 21	0 22
REG	GPIO0_21_MUX	GPIO0_22_MUX
MODE 0	GPIO0_21	GPIO0_22
MODE 1	UART3_RXD	DSP0_JTG_TRST
MODE 2	UART3_IR_IN	I2C4_SCL
MODE 3	~	~
MODE 4	DPU_COLOR_0	DPU_COLOR_1
MODE 5	DPU1_COLOR_0	DPU1_COLOR_1

4.1.2 Connector P9

The following tables show the pinout of the **P9** expansion header. The SW is responsible for setting the default function of each pin. Refer to the processor documentation for more information on these pins and detailed descriptions of all of the pins listed. In some cases there may not be enough signals to complete a group of signals that may be required to implement a total interface.

The column heading is the pin number on the expansion header.

The **GPIO** row is the expected gpio identifier number in the Linux kernel.

Each row includes the gpiochipX and pinY in the format of *X Y*. You can use these values to directly control the GPIO pins with the commands shown below.

```
# to set the GPIO pin state to HIGH
debian@BeagleBone:~$ gpioset X Y=1

# to set the GPIO pin state to LOW
debian@BeagleBone:~$ gpioset X Y=0

For Example:

+---------+----------+
| Pin     | P9.11    |
+=========+==========+
| GPIO    | 1 1      |
+---------+----------+

Use the commands below for controlling this pin (P9.11) where X = 1 and Y = 1

# to set the GPIO pin state to HIGH
debian@BeagleBone:~$ gpioset 1 20=1

# to set the GPIO pin state to LOW
debian@BeagleBone:~$ gpioset 1 20=0
```

The **BALL** row is the pin number on the processor.

The **REG** row is the offset of the control register for the processor pin.

The **MODE #** rows are the mode setting for each pin. Setting each mode to align with the mode column will give that function on that pin.

If included, the **2nd BALL** row is the pin number on the processor for a second processor pin connected to the same pin on the expansion header. Similarly, all row headings starting with **2nd** refer to data for this second processor pin.

NOTES:

DO NOT APPLY VOLTAGE TO ANY I/O PIN WHEN POWER IS NOT SUPPLIED TO THE BOARD. IT WILL DAMAGE THE PROCESSOR AND VOID THE WARRANTY.

NO PINS ARE TO BE DRIVEN UNTIL AFTER THE SYS_RESET LINE GOES HIGH.

P9.01-P9.05

P9.01	P9.02	P9.03	P9.04	P9.05
GND	GND	VOUT_3V3	VOUT_3V3	VIN

P9.06-P9.10

P9.06	P9.07	P9.08	P9.09	P9.10
VIN	VOUT_SYS	VOUT_SYS	ONKEY#	RESET#

P9.11-P9.13

Pin	P9.11	P9.12	P9.13
Name	UART1_TXD	QSPI0_CSN0	UART1_RXD
BALL	M32	H1	M33
GPIO	0 10	2 3	0 11
REG	UART1_TXD_MUX	QSPI0_CSN0_MUX	UART1_RXD_MUX
MODE 0	UART1_TXD	QSPI0_SSN0	UART1_RXD
MODE 1	~	PWM1	~
MODE 2	~	I2S_SDA1	~
MODE 3	GPIO0_10	GPIO2_3	GPIO0_11
MODE 4	~	~	~
MODE 5	~	~	~

P9.14-P9.16

Pin	P9.14	P9.15	P9.16
Name	QSPI0_D1_MISO	QSPI0_D2_WP	QSPI0_D0_MOSI
BALL	K3	K2	J3
GPIO	2 6	2 7	2 5
REG	QSPI0_D1_MISO_MUX	QSPI0_D2_WP_MUX	QSPI0_D0_MOSI_MUX
MODE 0	QSPI0_M1_MISO	QSPI0_M2_WP	QSPI0_M0_MOSI
MODE 1	PWM4	PWM5	PWM3
MODE 2	I2S_MCLK	I2S_SCLK	I2S_SDA3
MODE 3	GPIO2_6	GPIO2_7	GPIO2_5
MODE 4	~	~	~
MODE 5	~	~	~

P9.17-P9.19

Pin	P9.17	P9.18	P9.19
Name	QSPI1_CSN0	QSPI1_D0_MOSI	I2C2_SCL
BALL	H32	G35	G4
GPIO	0 1	0 2	2 9
REG	QSPI1_CSN0_MUX	QSPI1_D0_MOSI_MUX	I2C2_SCL_MUX
MODE 0	QSPI1_SSN0	QSPI1_M0_MOSI	I2C2_SCL
MODE 1	~	ISO7816_CVCC_EN	UART2_TXD
MODE 2	I2S_MCLK	I2C5_SDA	~
MODE 3	GPIO0_1	GPIO0_2	GPIO2_9
MODE 4	EFUSE_SPI_NSS	EFUSE_SPI_SI	~
MODE 5	~	~	~

P9.20-P9.22

Pin	P9.20	P9.21	P9.22
Name	I2C2_SDA	QSPI1_D1_MISO	QSPI1_SCLK
BALL	G3	G34	H34
GPIO	2 10	0 3	0 0
REG	I2C2_SDA_MUX	QSPI1_D1_MISO_MUX	QSPI1_SCLK_MUX
MODE 0	I2C2_SDA	QSPI1_M1_MISO	QSPI1_SCLK
MODE 1	UART2_RXD	ISO7816_CLK	ISO7816_DET
MODE 2	~	~	~
MODE 3	GPIO2_10	GPIO0_3	GPIO0_0
MODE 4	~	EFUSE_SPI_SO	EFUSE_SPI_CLK
MODE 5	~	~	~

P9.23-P9.25

Pin	P9.23	P9.24	P9.25
Name	QSPI0_D3_HOLD	QSPI1_D2_WP	GPIO2_18
BALL	K1	G33	F5
GPIO	2 8	0 4	2 18
REG	QSPI0_D3_HOLD_MUX	QSPI1_D2_WP_MUX	GPIO2_18_MUX
MODE 0	QSPI0_M3_HOLD	QSPI1_M2_WP	GPIO2_18
MODE 1	~	ISO7816_RST	GMAC1_TX_CLK
MODE 2	I2S_WS	UART5_TXD	~
MODE 3	GPIO2_8	GPIO0_4	~
MODE 4	~	EFUSE_BUSY	~
MODE 5	~	~	~

P9.26-P9.28

Pin	P9.26	P9.27	P9.28
Name	QSPI1_D3_HOLD	GPIO2_19	SPI_CSN
BALL	F37	E4	E3
GPIO	0 5	2 19	2 15
REG	QSPI1_D3_HOLD_MUX	GPIO2_19_MUX	SPI_CSN_MUX
MODE 0	QSPI1_M3_HOLD	GPIO2_19	SPI_SSN0
MODE 1	ISO7816_DAT	GMAC1_RX_CLK	UART2_RXD
MODE 2	UART5_RXD	~	UART2_IR_IN
MODE 3	GPIO0_5	~	GPIO2_15
MODE 4	~	~	~
MODE 5	~	~	~

P9.29-P9.31

Pin	P9.29	P9.30	P9.31
Name	SPI_MISO	SPI_MOSI	SPI_SCLK
BALL	F1	F2	D3
GPIO	2 17	2 16	2 14
REG	SPI_MISO_MUX	SPI_MOSI_MUX	SPI_SCLK_MUX
MODE 0	SPI_MISO	SPI_MOSI	SPI_SCLK
MODE 1	~	~	UART2_TXD
MODE 2	~	~	UART2_IR_OUT
MODE 3	GPIO2_17	GPIO2_16	GPIO2_14
MODE 4	~	~	~
MODE 5	~	~	~

P9.32-P9.40

P9.32	P9.34
VDD_ADC	GND

P9.33	P9.35	P9.36	P9.37	P9.38	P9.39	P9.40
ADC_VIN_CH4	ADC_VIN_CH6	ADC_VIN_CH5	ADC_VIN_CH2	ADC_VIN_CH3	ADC_VIN_CH0	ADC_VIN_CH1

P9.41-P9.42

Pin	P9.41	P9.42
Name	GPIO2_13	QSPI0_SCLK
BALL	D2	H3
GPIO	2 13	2 2
REG	GPIO2_13_MUX	QSPI0_SCLK_MUX
MODE 0	GPIO2_13	QSPI0_SCLK
MODE 1	SPI_SSN1	PWM0
MODE 2	~	I2S_SDA0
MODE 3	~	GPIO2_2
MODE 4	~	~
MODE 5	~	~

P9.43-P9.46

P9.43	P9.44	P9.45	P9.46
GND	GND	GND	GND

mikroBUS

Pin	mikroBUS port		Pin
ADC_VIN_CH7	**AN**	**PWM**	QSPI0_CSN1 (MODE1:PWM2)
AUDIO_PA3 (MODE3:GPIO4_3)	**RST**	**INT**	GPIO2_21 (MODE0:GPIO2_21)
GPIO2_20 (MODE0:GPIO2_20)	**CS**	**RX**	UART3_RXD (MODE1:UART3_RXD)
SPI_SCLK (MODE0:SPI_SCLK)	**SCK**	**TX**	UART3_TXD (MODE1:UART3_TXD)
SPI_MISO (MODE0:SPI_MISO)	**MISO**	**SCL**	GPIO0_18 (MODE1:I2C4_SCL)
SPI_MOSI (MODE0:SPI_MOSI)	**MOSI**	**SDA**	GPIO0_19 (MODE1:I2C4_SDA)
3.3V supply	**3V3**	**5V**	5V supply
Ground	**GND**	**GND**	Ground

Chapter 5

Demos

Todo: We need a CSI capture and DSI display demos

Todo: We need a cape compatibility layer demo

Important: This document is a work on progress.

5.1 Using CSI Cameras

Note: CSI support is only available in Yocto image for BeagleV Ahead, to flash latest Yoctor image on your BeagleV Ahead you can checkout *Flashing eMMC* section.

5.1.1 Hardware

IMX219 camera modules has been tested to work well with BeagleV Ahead, some of them are listed below:

1. Raspberry Pi Camera Board v2 - 8 Megapixels (Adafruit)
2. Raspberry Pi NoIR Camera Board v2 - 8 Megapixels (Adafuit)
3. Arducam IMX219 (Robu.in)

In addition to the camer you'll need a 15pin to 22pin cable as well:

1. Raspberry Pi Zero FPC Camera Cable (Adafruit)
2. Raspberry Pi Zero v1.3 Camera Cable (Adafuit)
3. Raspberry Pi Zero V1.3 Camera Cable (Robu.in)

5.1.2 Software

There are several demo applications available for testing CSI, execute commands below to test your IMX219 camera on CSI0 & CSI1 ports:

1. Change directory to demo application location using: `cd /usr/share/vi/isp/test`

2. Set environment variable `export ISP_LOG_LEVEL=3`
3. To test CSI0 execute: `./camera_demo1 2 0 1 0 1920 1080 1 30 7`
4. To test CSI1 execute: `./camera_demo1 0 0 1 0 1920 1080 1 30 7`

When you execure *camera_demo1* then you should see something like this on your console:

```
...
...
IMX219: IMX219_IsiExposureControlIss: g=168.960999, Ti=0.050000
CAMERIC-MI-IRQ: isp mi frame out (59)  fps[0]: 19.74
IMX219: IMX219_IsiExposureControlIss: g=168.960999, Ti=0.050000
CAMERIC-MI-IRQ: isp mi frame out (60)  fps[0]: 19.73
IMX219: IMX219_IsiExposureControlIss: g=168.960999, Ti=0.050000
CAMERIC-MI-IRQ: isp mi frame out (61)  fps[0]: 19.72
IMX219: IMX219_IsiExposureControlIss: g=168.960999, Ti=0.050000
CAMERIC-MI-IRQ: isp mi frame out (62)  fps[0]: 19.72
IMX219: IMX219_IsiExposureControlIss: g=168.960999, Ti=0.050000
CAMERIC-MI-IRQ: isp mi frame out (63)  fps[0]: 19.71
...
...
```

The output above indicates your CSI camera is working well.

Important: Usage of other demo applications will be added to this page as well.

Source for these demo application can be found here

Chapter 6

Support

All support for BeagleV Ahead design is through BeagleBoard.org community at BeagleBoard.org forum.

6.1 Production board boot media

- BeagleV-Ahead Rev 1.0

6.2 Certifications and export control

6.2.1 Export designations

Todo: update details

- HS:
- US HS:
- EU HS:

6.2.2 Size and weight

Todo: update details

- Bare board dimensions:
- Bare board weight:
- Full package dimensions:
- Full package weight:

6.3 Additional documentation

6.3.1 Hardware docs

For any hardware document like schematic diagram PDF, EDA files, issue tracker, and more you can checkout the BeagleV Ahead design repository.

6.3.2 Software docs

For BeagleV Ahead specific software projects you can checkout all the BeagleV Ahead project repositories group.

6.3.3 Support forum

For any additional support you can submit your queries on our forum, https://forum.beagleboard.org/c/beaglev

6.3.4 Pictures

6.4 Change History

Note: This section describes the change history of this document and board. Document changes are not always a result of a board change. A board change will always result in a document change.

6.4.1 Board Changes

For all changes, see https://git.beagleboard.org/beaglev-ahead/beaglev-ahead. Versions released into production are noted below.

Table 6.1: BeagleV Ahead board change history

Rev	Changes	Date	By
		2023-03-08	