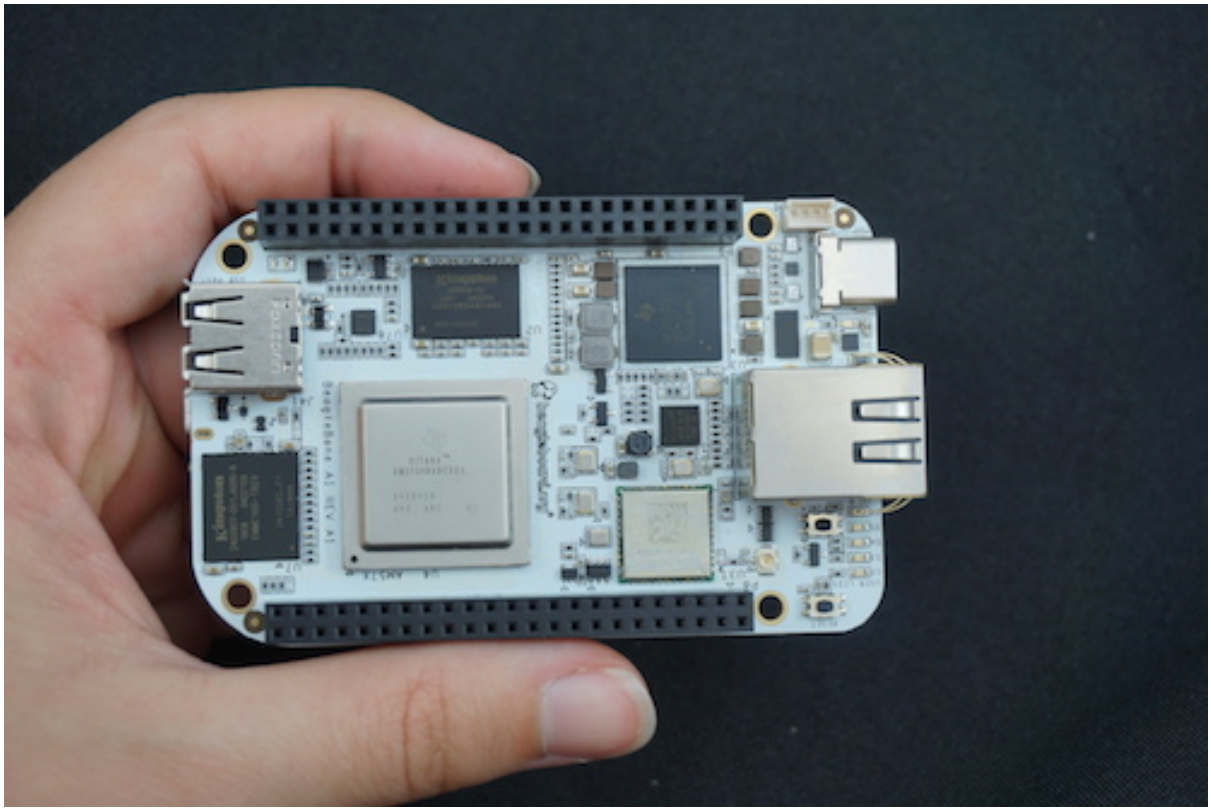
BeagleBone AI

BeagleBoard.org Foundation
Jan 02, 2025

Table of contents

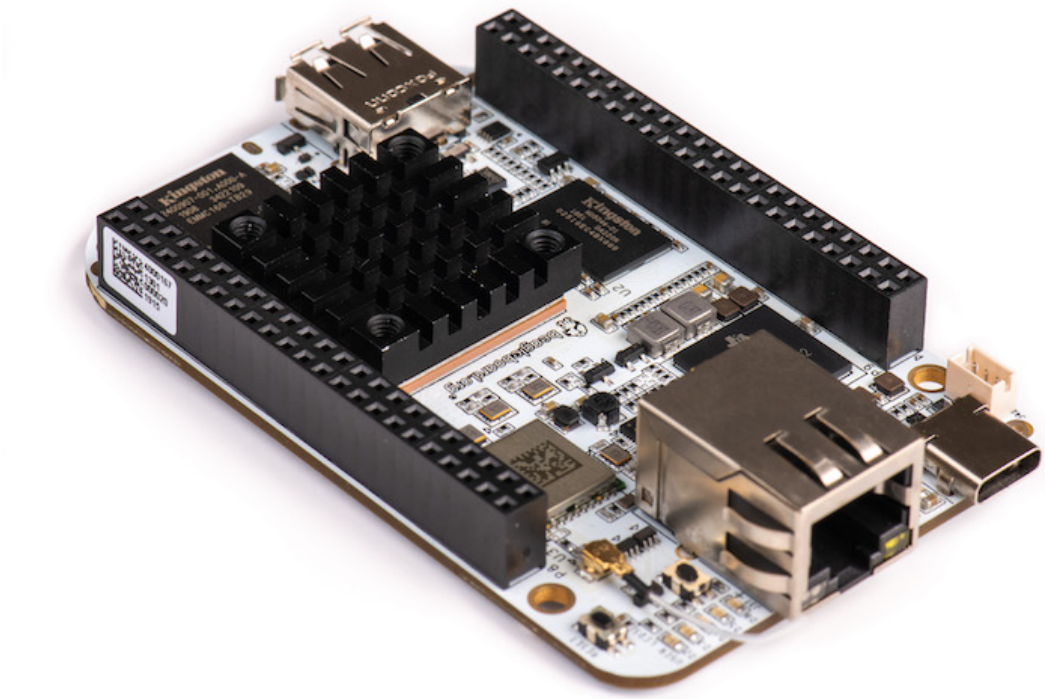
BeagleBone AI is based on the Texas Instruments AM5729 dual-core Cortex-A15 SoC with flexible BeagleBone Black header and mechanical compatibility. BeagleBone AI makes it easy to explore how artificial intelligence (AI) can be used in everyday life via the TI C66x digital-signal-processor (DSP) cores and embedded-vision-engine (EVE) cores supported through an optimized TIDL machine learning OpenCL API with pre-installed tools. Focused on everyday automation in industrial, commercial and home applications.



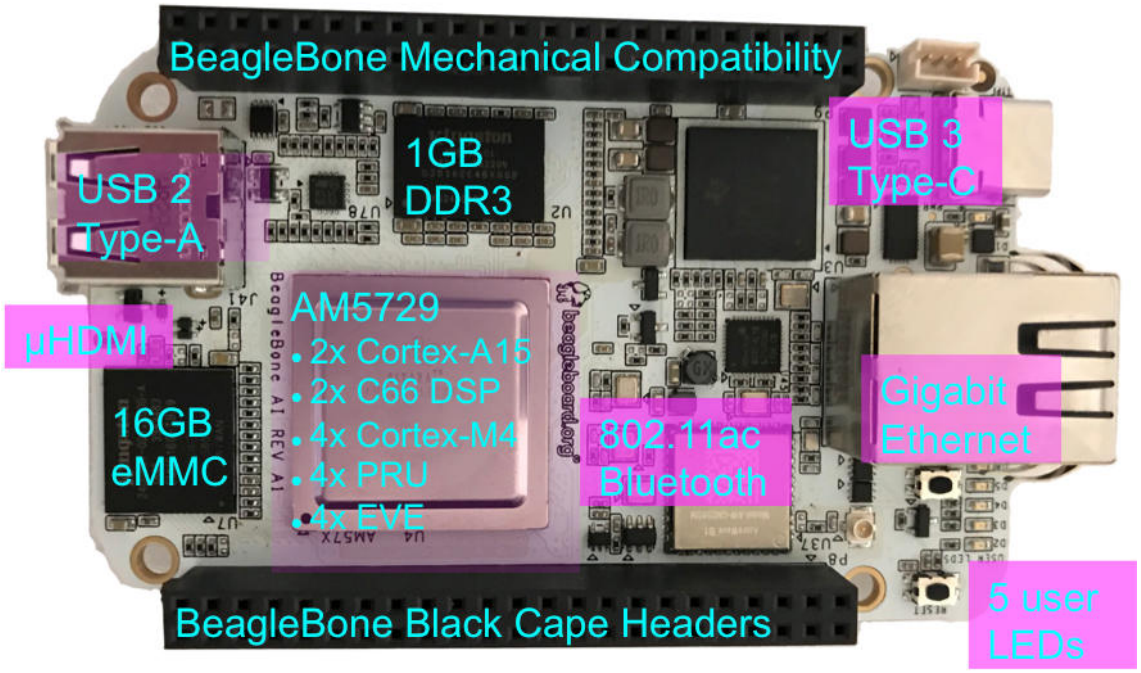
Chapter 1

Introduction

Built on the proven BeagleBoard.org® open source Linux approach, BeagleBone® AI fills the gap between small SBCs and more powerful industrial computers. Based on the Texas Instruments AM5729, developers have access to the powerful SoC with the ease of BeagleBone® Black header and mechanical compatibility. BeagleBone® AI makes it easy to explore how artificial intelligence (AI) can be used in everyday life via TI C66x digital-signal-processor (DSP) cores and embedded-vision-engine (EVE) cores supported through an optimized TIDL machine learning OpenCL API with pre-installed tools. Focused on everyday automation in industrial, commercial and home applications.



1.1 BeagleBone AI Overview



1.1.1 BeagleBone® AI Features

1.2 Main Processor Features of the AM5729 Within BeagleBone® AI

- Dual 1.5GHz ARM® Cortex®-A15 with out-of-order speculative issue 3-way superscalar execution pipeline for the fastest execution of existing 32-bit code
- 2 C66x Floating-Point VLIW DSP supported by OpenCL
- 4 Embedded Vision Engines (EVEs) supported by TIDL machine learning library
- 2x Dual-Core Programmable Real-Time Unit (PRU) subsystems (4 PRUs total) for ultra low-latency control and software generated peripherals
- 2x Dual ARM® Cortex®-M4 co-processors for real-time control
- IVA-HD subsystem with support for 4K @ 15fps H.264 encode/decode and other codecs @ 1080p60
- Vivante® GC320 2D graphics accelerator
- Dual-Core PowerVR® SGX544™ 3D GPU

1.3 Communications

- BeagleBone Black header and mechanical compatibility
- 16-bit LCD interfaces
- 4+ UARTs
- 2 I2C ports
- 2 SPI ports

- Lots of PRU I/O pins

1.4 Memory

- 1GB DDR3L
- 16GB on-board eMMC flash

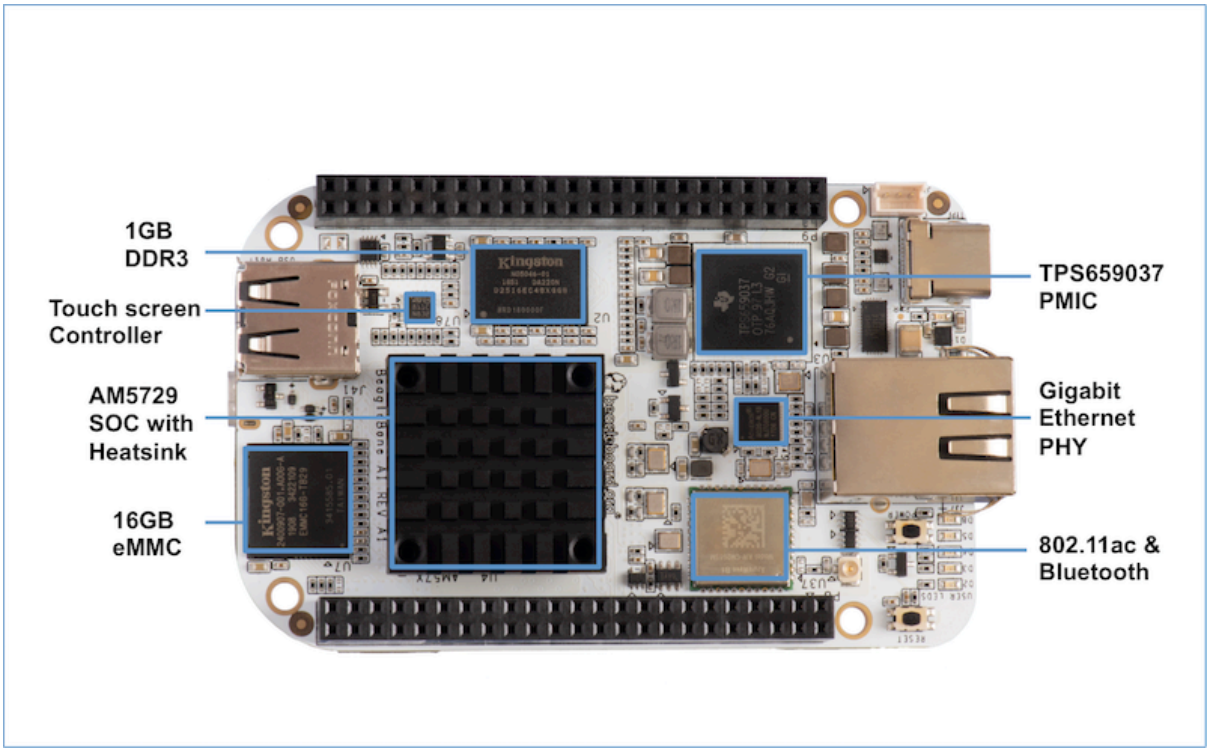
1.5 Connectors

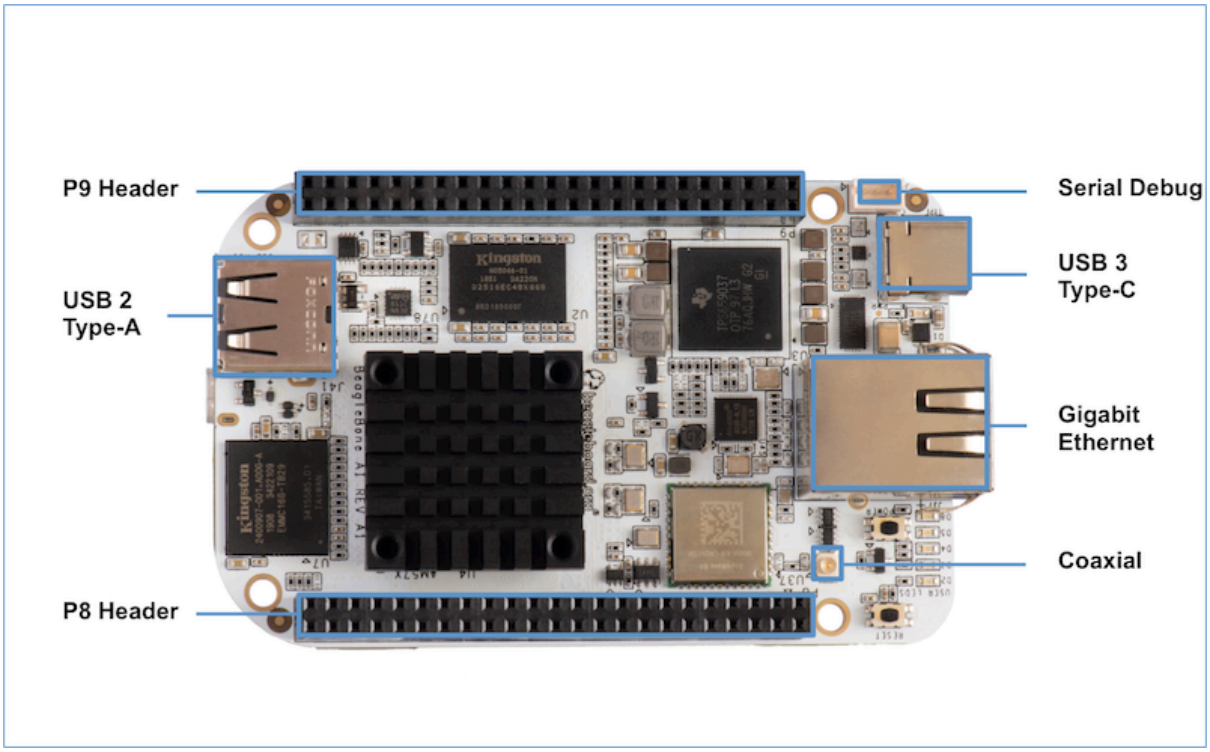
- USB Type-C connector for power and SuperSpeed dual-role controller
- Gigabit Ethernet
- 802.11ac 2.4/5GHz WiFi via the AzureWave AW-CM256SM

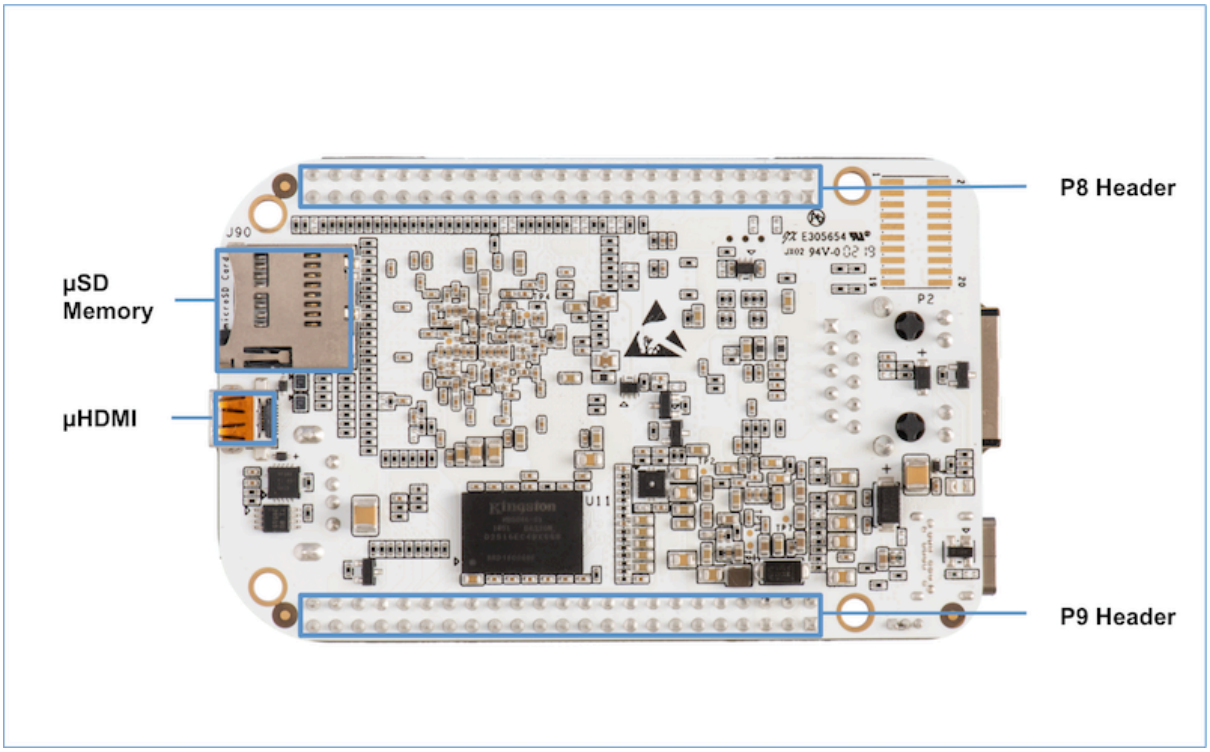
1.6 Out of Box Software

- Zero-download out of box software environment

1.6.1 Board Component Locations




P9 Header
USB 2
Type-A
P8 Header
Serial Debug
USB 3
Type-C
Gigabit
Ethernet
Coaxial


µSD
Memory
µHDMI
P8 Header
P9 Header

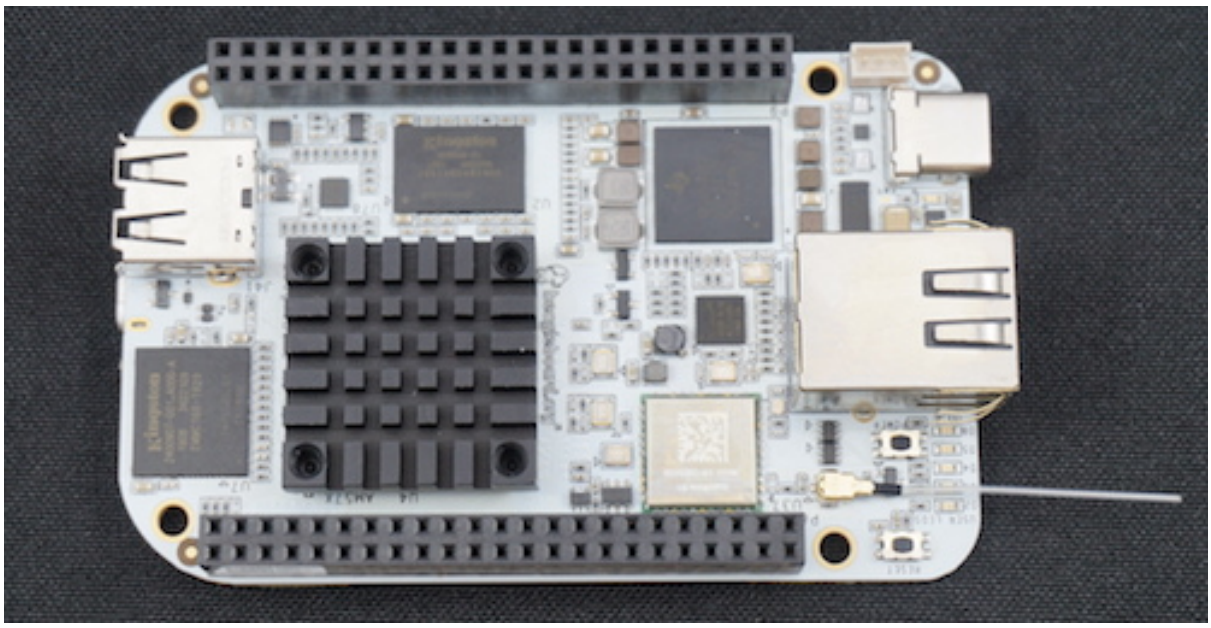
Chapter 2

Quick start

2.1 What's In the Box

BeagleBone® AI comes in the box with the heat sink and antenna already attached. Developers can get up and running in five minutes with no microSD card needed. BeagleBone® AI comes preloaded with a Linux distribution. In the box you will find:

- BeagleBone® AI
- Quick Start Guide



Tip: For board files, 3D model, regulatory docs and more, you can checkout BeagleBona-AI repository on OpenBeagle.

2.2 What's Not in the Box

You will need to purchase:

- USB C cable or USB C to USB A cable
- MicroSD Card (optional)

- Serial cable (optional)

More information or to purchase a replacement heat sink or antenna, please go to these websites:

- Antenna
- Heat Sink

2.3 Fans

The pre-attached heat sink has M3 holes spaced 20x20 mm. The height of the heat sink clears the USB type A socket, and all other components on the board except the 46-way header sockets and the Ethernet socket.

If you run all of the accelerators or have an older software image, you'll likely need fan. To find a fan, visit the link to fans in the FAQ.

> **Caution:** BeagleBone AI can run **HOT**! Even without running the accelerators, getting up to 70C is not uncommon.
>
> Read about *Fan update for BeagleBone AI on Forum <https://forum.beagleboard.org/t/fan-update-for-beaglebone-ai/1964>*.

1. Official BeagleBone Fan Cape
2. *Coolerguys 25mm (25x25x10) USB Fan <https://www.coolerguys.com/en-in/collections/usb-fans/products/25mm-25x25x10-usb-fan>*

2.4 Main Connection Scenarios

This section will describe how to connect the board for use. The board can be configured in several different ways. Below we will walk through the most common scenarios. NOTE: These connection scenarios are dependent on the software image presently on your BeagleBone® AI. When all else fails, follow the instructions at *Upgrade BeagleBone AI software*.

- Tethered to a PC via USB C cable
- Standalone Desktop with powered USB hub, display, keyboard and mouse
- Wireless Connection to BeagleBone® AI

Tethered

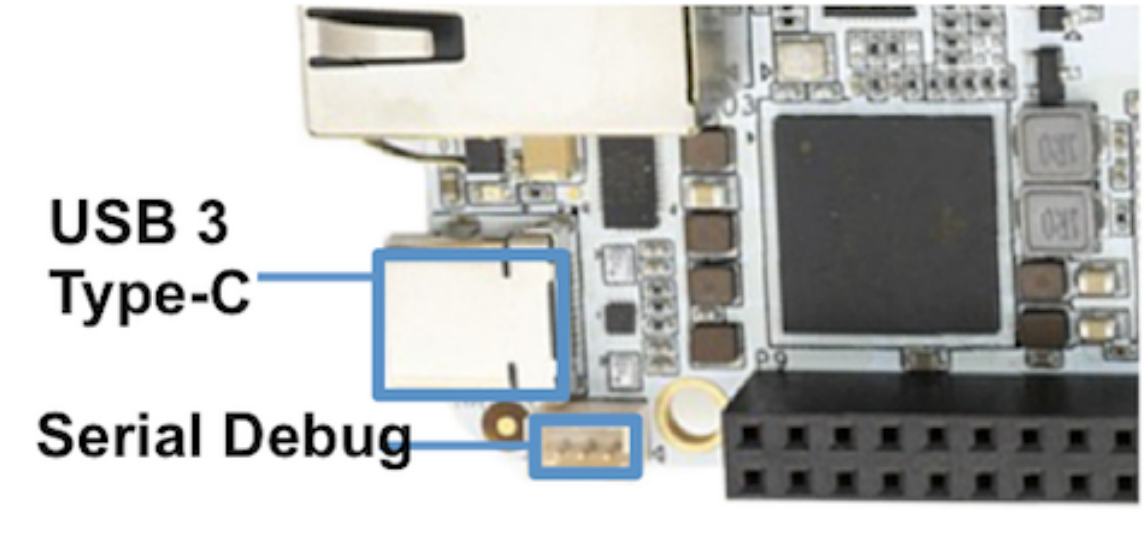
Tethered to a PC

The most common way to program BeagleBone® AI is via a USB connection to a PC. If your computer has a USB C type port, BeagleBone® AI will both communicate and receive power directly from the PC. If your computer does not support USB C type, you can utilize a powered USB C hub to power and connect to BeagleBone® AI which in turn will connect to your PC. You can also use a powered USB C hub to power and connect peripheral devices such as a USB camera. After booting, the board is accessed either as a USB storage device or via the browser on the PC. You will need Chrome or Firefox on the PC.

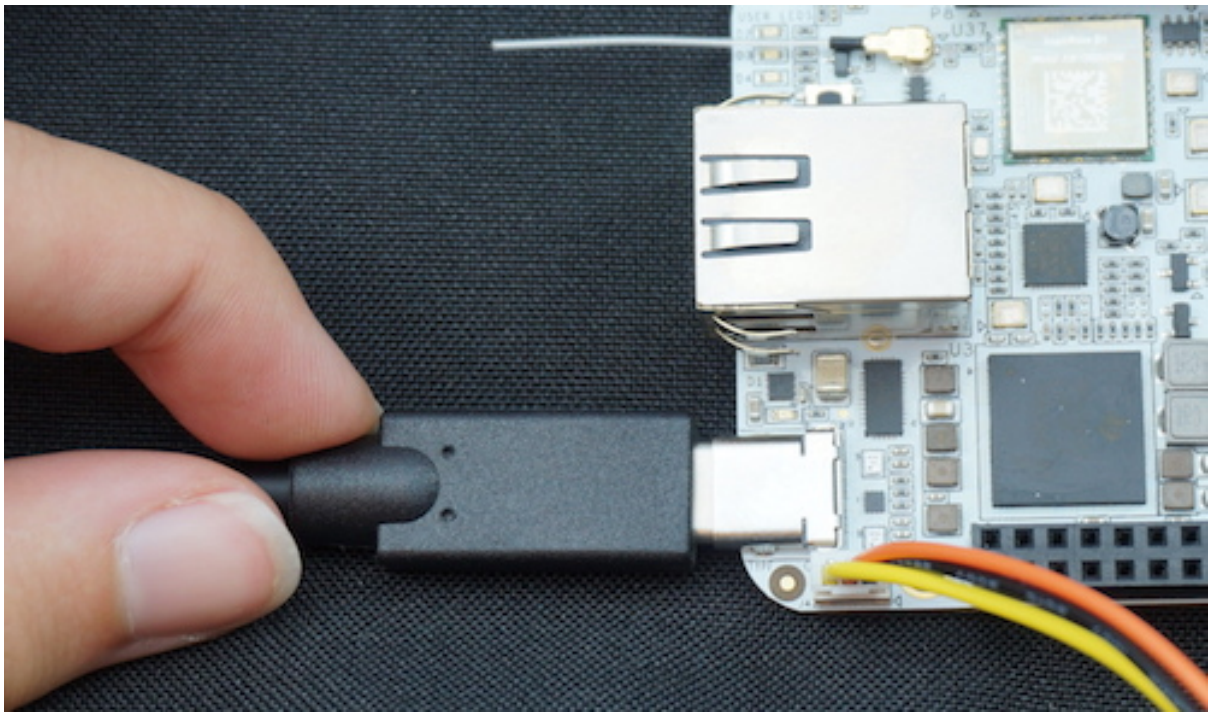
Note: Start with this image "am57xx-eMMC-flasher-debian-10.3-iot-tidl-armhf-2020-04-06-6gb.img.xz" loaded on your BeagleBone® AI.

---

1. Locate the USB Type-C connector on BeagleBone® AI

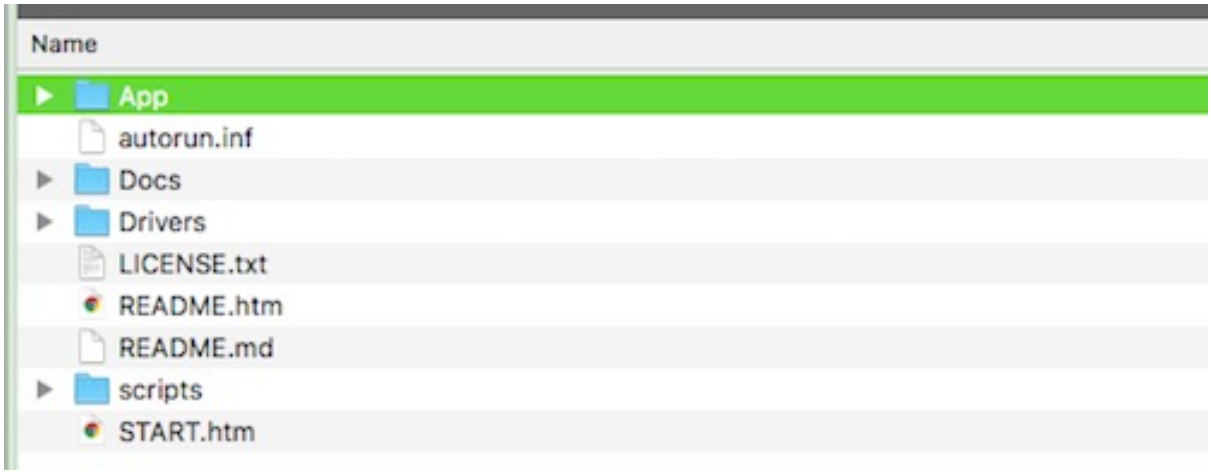


2. Connect a USB type-C cable to BeagleBone® AI USB type-C port.

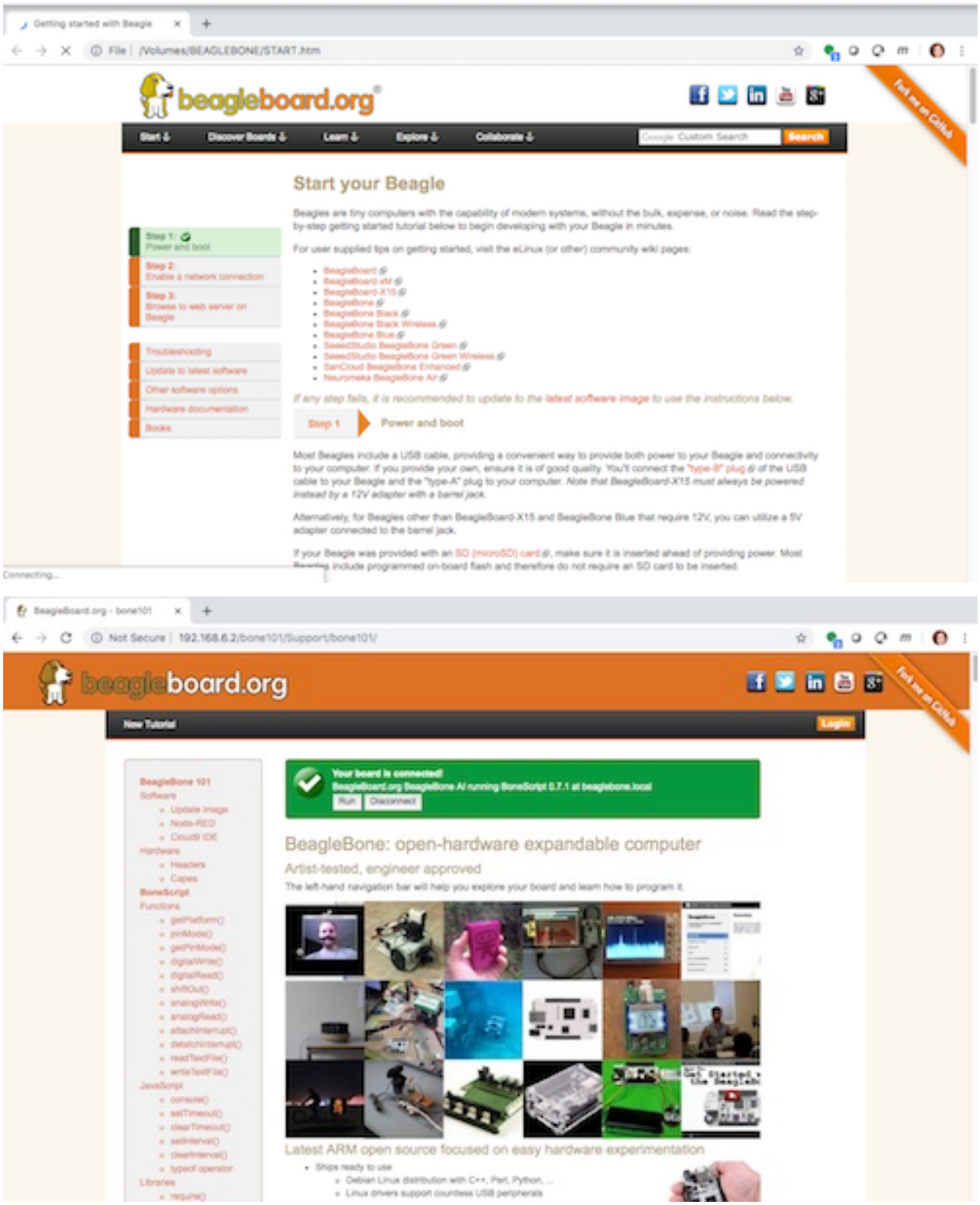


3. Connect the other end of the USB cable to the PC USB 3 port.

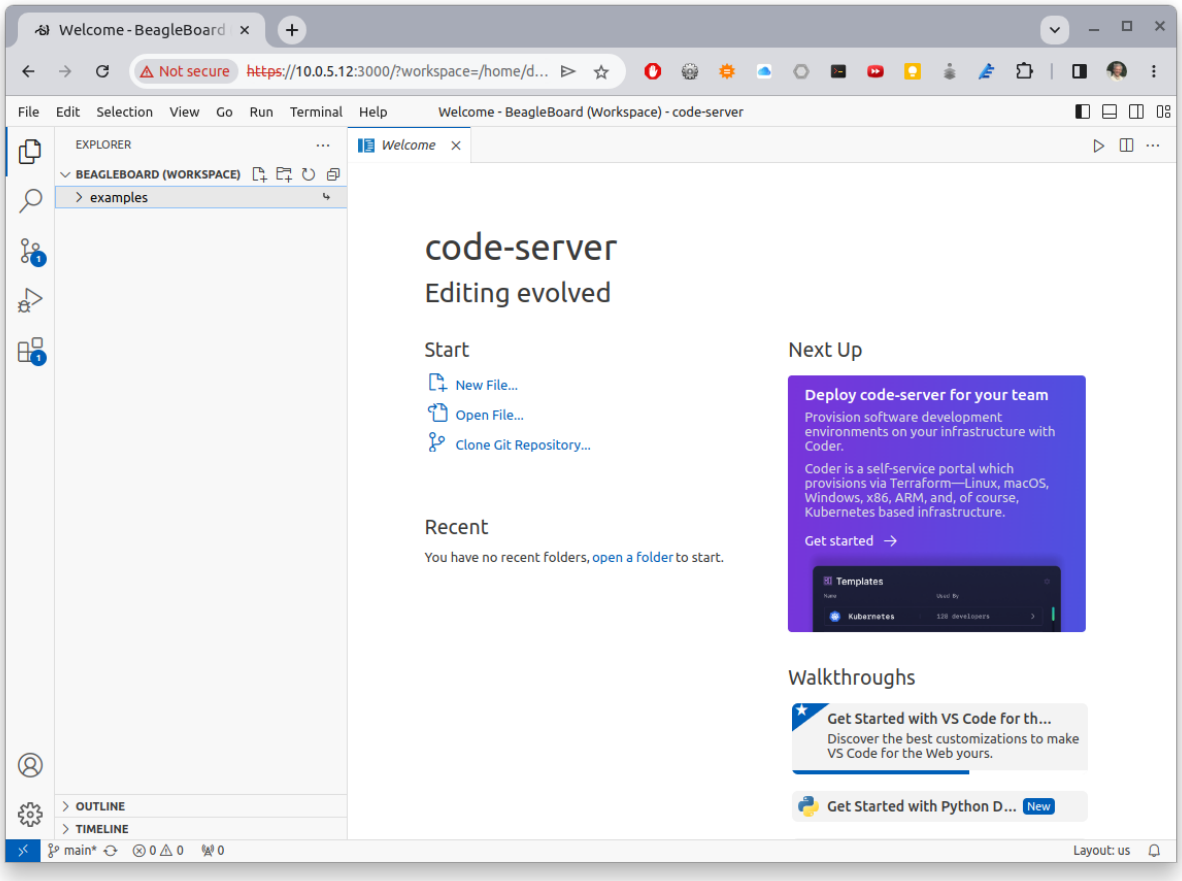
4. BeagleBone® AI will boot.
5. You will notice some of the 5 user LEDs flashing
6. Look for a new mass storage drive to appear on the PC.



7. Open the drive and open START.HTM with your web browser.



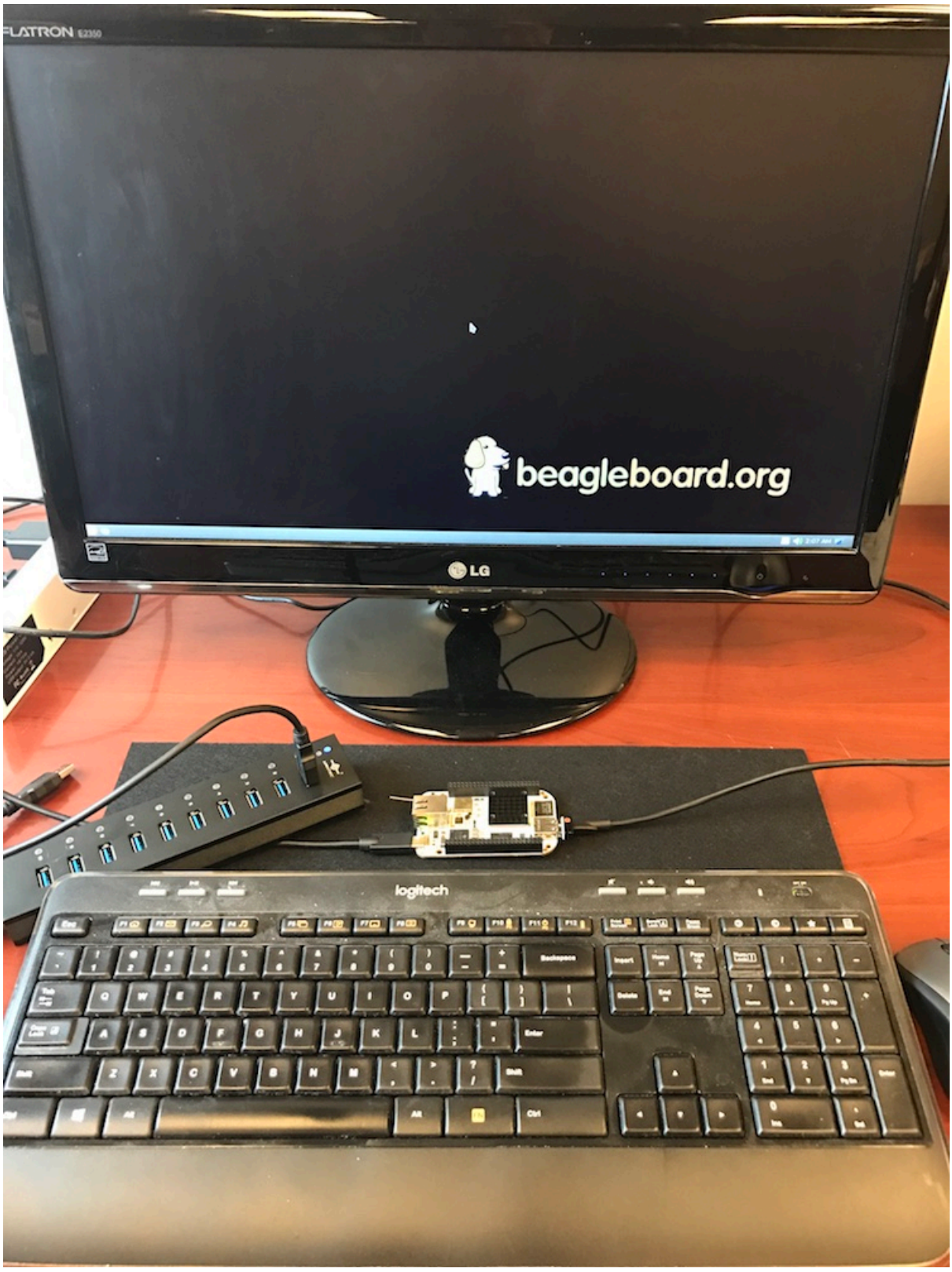
8. Follow the instructions in the browser window.



9. Go to Visual Studio Code IDE.

Standalone

Standalone w/Display and Keyboard/Mouse



Note: This configuration requires loading the latest debian 9 image from https://elinux.org/Beagleboard:Latest-images-testing

Load “am57xx-eMMC-flasher-debian-9.13-lxqt-tidl-armhf-2020-08-25-6gb.img.xz” image on the BeagleBone® AI

10. Connect a combo keyboard and mouse to BeagleBone® AI's USB host port.
11. Connect a microHDMI-to-HDMI cable to BeagleBone® AI's microHDMI port.
12. Connect the microHDMI-to-HDMI cable to an HDMI monitor.
13. Plug a 5V 3A USB type-C power supply into BeagleBone® AI's USB type-C port.
14. BeagleBone® AI will boot. No need to enter any passwords.
15. Depending on which software image is loaded, either a Desktop or a login shell will appear on the monitor.
16. Follow the instructions at https://beagleboard.org/upgrade

Wireless

Wireless Connection

Note: Start with this image "am57xx-eMMC-flasher-debian-10.3-iot-tidl-armhf-2020-04-06-6gb.img.xz" loaded on your BeagleBone® AI.

---

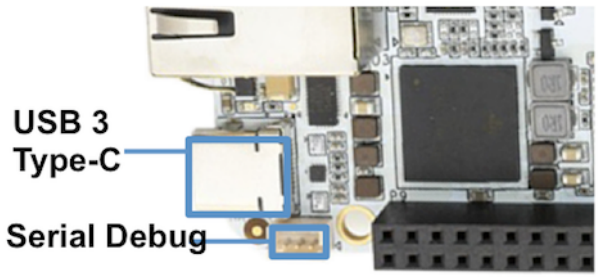
17. Plug a 5V 3A USB type-C power supply into BeagleBone® AI's USB type-C port.
18. BeagleBone® AI will boot.
19. Connect your PC's WiFi to SSID "BeagleBone-XXXX" where XXXX varies for your BeagleBone® AI.
20. Use password "BeagleBone" to complete the WiFi connection.
21. Open http://192.168.8.1 in your web browser.
22. Follow the instructions in the browser window.

2.5 Connecting a 3 PIN Serial Debug Cable

A 3 PIN serial debug cable can be helpful to debug when you need to view the boot messages through a terminal program such as putty on your host PC. This cable is not needed for most BeagleBone® AI boot up scenarios.

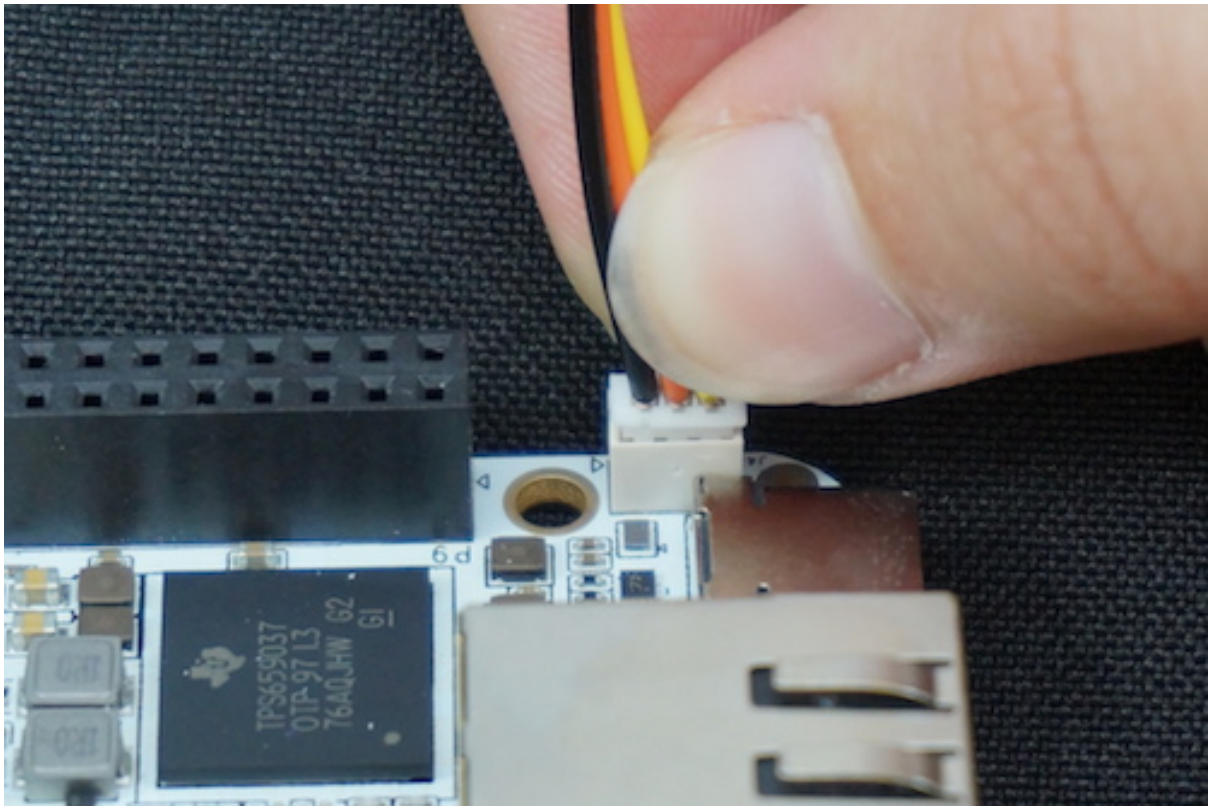
Cables: https://git.beagleboard.org/beagleboard/beaglebone-ai/-/wikis/Frequently-Asked-Questions#serial-cable

Locate the 3 PIN debug header on BeagleBone® AI, near the USB C connection.



Press the small white connector into the 3 PIN debug header. The pinout is:

- Pin 1 (the pin closest to the screw-hole in the board. It is also marked with a shape on the silkscreen): GND
- Pin 2: UART1_RX (i.e. this is a BB-AI input pin)
- Pin 3: UART1_TX (i.e. BB-AI transmits out on this pin)


TPS65903T
OTP-97 L3
76AQJHW G2
G1

Chapter 3

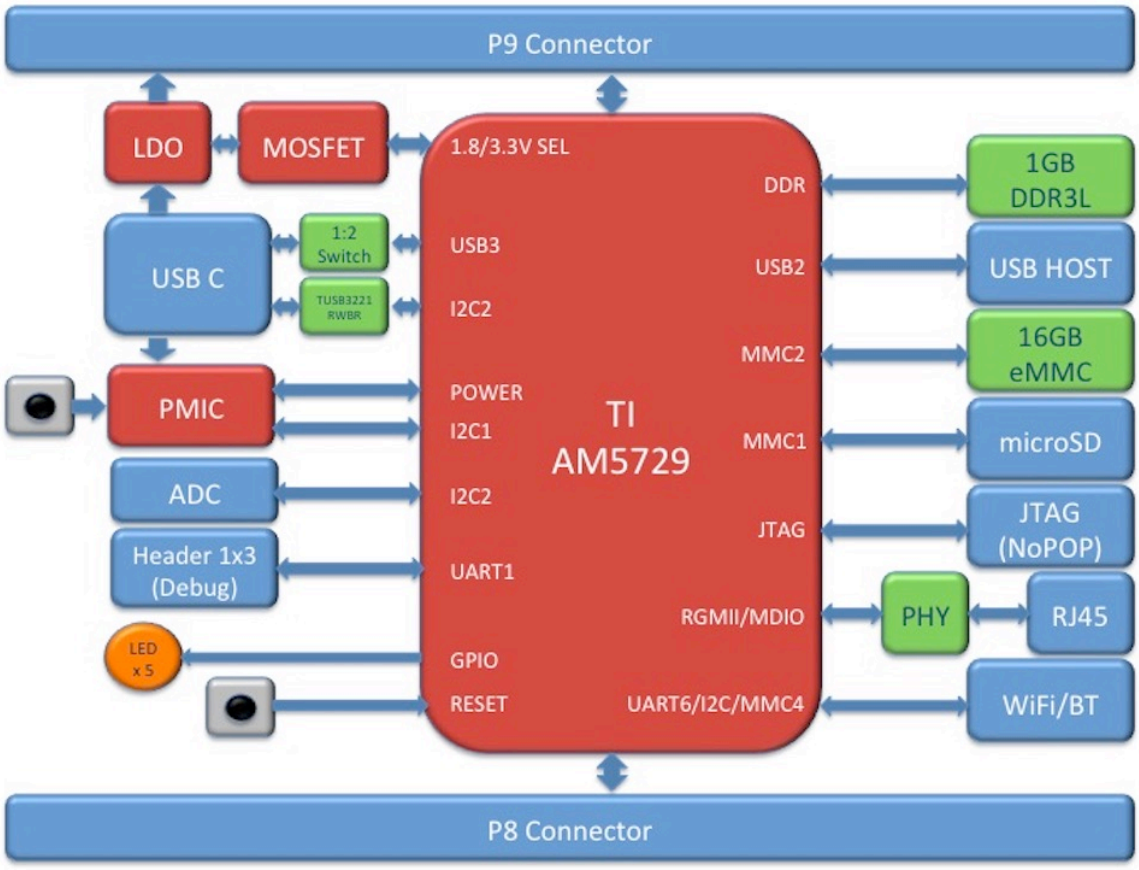
Design and specifications

This section provides a detailed description of the Hardware design. This can be useful for interfacing, writing drivers, or using it to help modify specifics of your own design.

The figure below is the high level block diagram of BeagleBone® AI. For those who may be concerned, this is the same figure found in section 5. It is placed here again for convenience so it is closer to the topics to follow.

3.1 Block Diagram

The figure below is the high level block diagram of BeagleBone® AI. For detailed layout information please check the schematics.

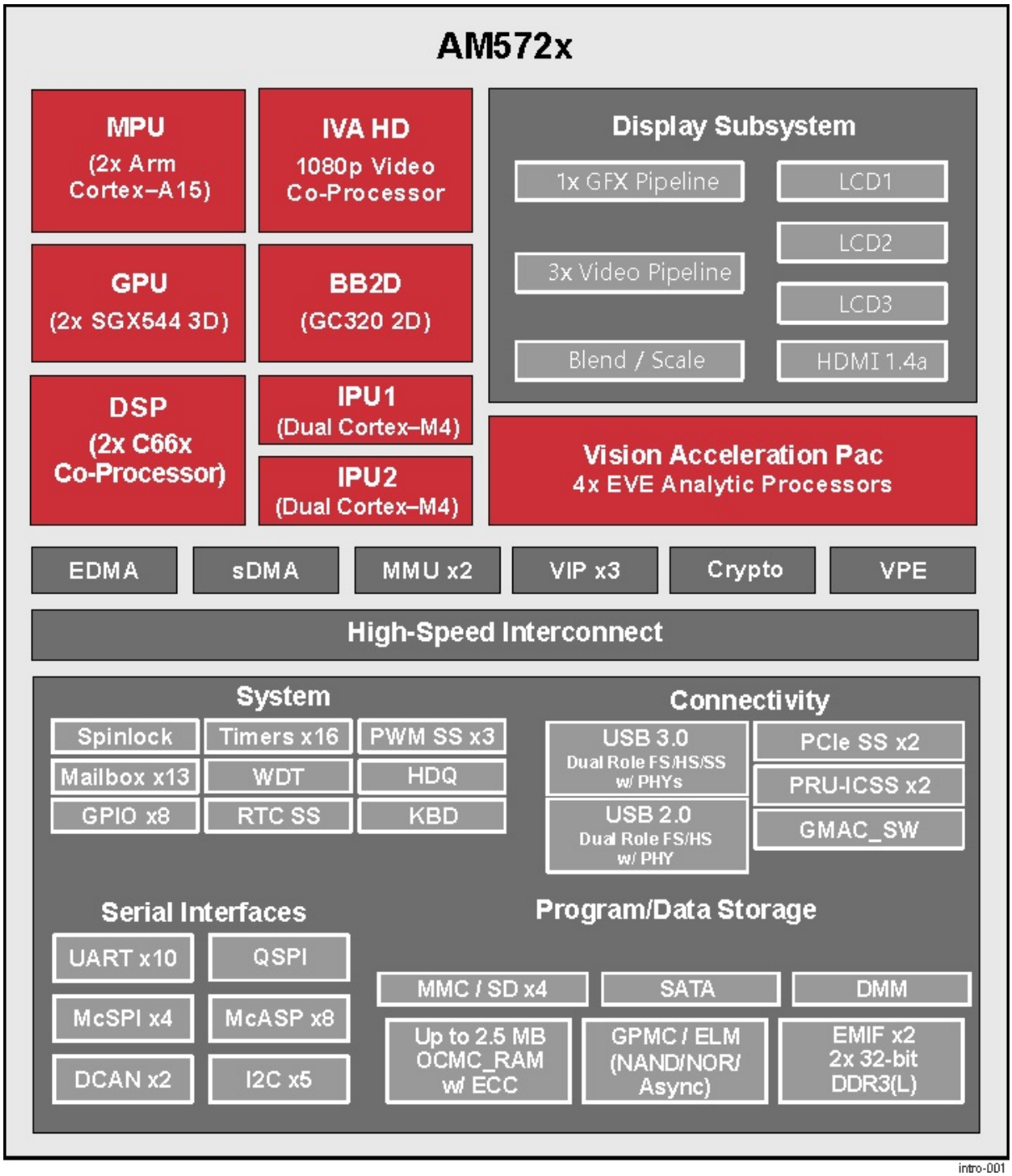


3.2 AM572x Sitara™ Processor

The Texas Instruments AM572x Sitara™ processor family of SOC devices brings high processing performance through the maximum flexibility of a fully integrated mixed processor solution. The devices also combine programmable video processing with a highly integrated peripheral set ideal for AI applications. The AM5729 used on BeagleBone® AI is the super-set device of the family.

Programmability is provided by dual-core ARM® Cortex®-A15 RISC CPUs with Arm® Neon™ extension, and two TI C66x VLIW floating-point DSP core, and Vision AccelerationPac (with 4x EVEs). The Arm allows developers to keep control functions separate from other algorithms programmed on the DSPs and coprocessors, thus reducing the complexity of the system software.

Texas Instruments AM572x Sitara™ Processor Family Block Diagram*



intro-001

MPU Subsystem The Dual Cortex-A15 MPU subsystem integrates the following submodules:

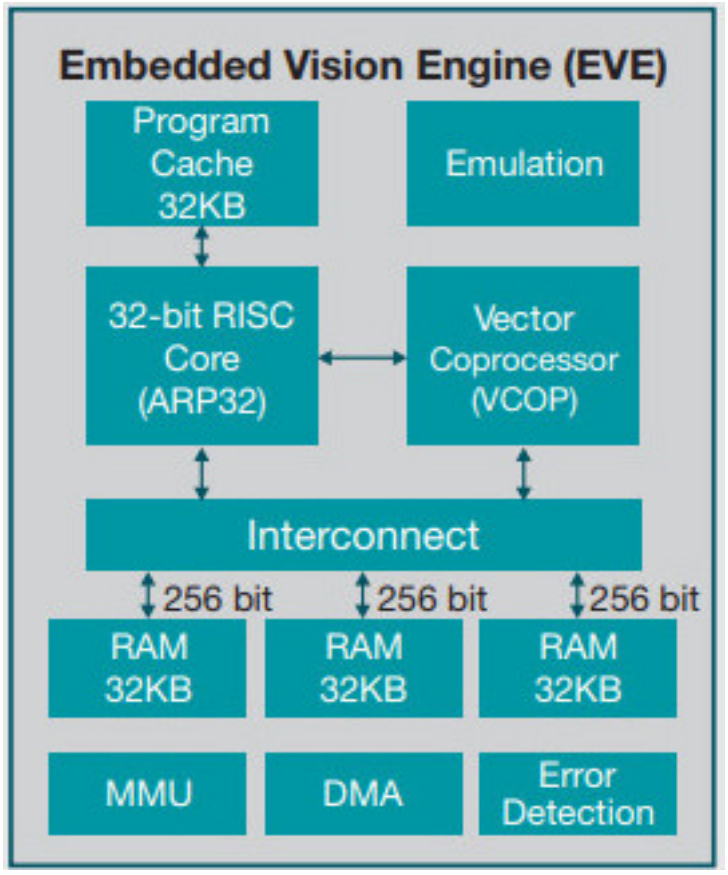
- ARM Cortex-A15 MPCore
  - Two central processing units (CPUs)
  - ARM Version 7 ISA: Standard ARM instruction set plus Thumb®-2, Jazelle® RCT Java™ accelerator, hardware virtualization support, and large physical address extensions (LPAE)
  - Neon™ SIMD coprocessor and VFPv4 per CPU
  - Interrupt controller with up to 160 interrupt requests
  - One general-purpose timer and one watchdog timer per CPU – Debug and trace features
  - 32-KiB instruction and 32-KiB data level 1 (L1) cache per CPU
- Shared 2-MiB level 2 (L2) cache
- 48-KiB bootable ROM
- Local power, reset, and clock management (PRCM) module
- Emulation features
- Digital phase-locked loop (DPLL)

DSP Subsystems There are two DSP subsystems in the device. Each DSP subsystem contains the following submodules:

- TMS320C66x™ Floating-Point VLIW DSP core for audio processing, and general-purpose imaging and video processing. It extends the performance of existing C64x+™ and C647x™ DSPs through enhancements and new features.
  - 32-KiB L1D and 32-KiB L1P cache or addressable SRAM
  - 288-KiB L2 cache
- 256-KiB configurable as cache or SRAM
- 32-KiB SRAM
- Enhanced direct memory access (EDMA) engine for video and audio data transfer
- Memory management units (MMU) for address management.
- Interrupt controller (INTC)
- Emulation capabilities
- Supported by OpenCL

EVE Subsystems

- 4 Embedded Vision Engines (EVEs) supported by TIDL machine learning library



The Embedded Vision Engine (EVE) module is a programmable imaging and vision processing engine. Software support for the EVE module is available through OpenCL Custom Device model with fixed set of functions. More information is available http://www.ti.com/lit/wp/spry251/spry251.pdf

PRU-ICSS Subsystems

- 2x Dual-Core Programmable Real-Time Unit (PRU) subsystems (4 PRUs total) for ultra low-latency control and software generated peripherals. Access to these powerful subsystems is available through through the P8 and P9 headers. These are detailed in Section 7.

IPU Subsystems There are two Dual Cortex-M4 IPU subsystems in the device available for general purpose usage, particularly real-time control. Each IPU subsystem includes the following components:

- Two Cortex-M4 CPUs
- ARMv7E-M and Thumb-2 instruction set architectures
- Hardware division and single-cycle multiplication acceleration
- Dedicated INTC with up to 63 physical interrupt events with 16-level priority
- Two-level memory subsystem hierarchy
  - L1 (32-KiB shared cache memory)
  - L2 ROM + RAM
- 64-KiB RAM
- 16-KiB bootable ROM
- MMU for address translation
- Integrated power management
- Emulation feature embedded in the Cortex-M4

IVA-HD Subsystem

- IVA-HD subsystem with support for 4K @ 15fps H.264 encode/decode and other codecs @ 1080p60 The IVA-HD subsystem is a set of video encoder and decoder hardware accelerators. The list of supported codecs can be found in the software development kit (SDK) documentation.

BB2D Graphics Accelerator Subsystem The Vivante® GC320 2D graphics accelerator is the 2D BitBlt (BB2D) graphics accelerator subsystem on the device with the following features:

- API support:
  - OpenWF™, DirectFB
  - GDI/DirectDraw
- BB2D architecture:
  - BitBlt and StretchBlt
  - DirectFB hardware acceleration
  - ROP2, ROP3, ROP4 full alpha blending and transparency
  - Clipping rectangle support
  - Alpha blending includes Java 2 Porter-Duff compositing rules
  - 90-, 180-, 270-degree rotation on every primitive
  - YUV-to-RGB color space conversion
  - Programmable display format conversion with 14 source and 7 destination formats
  - High-quality, 9-tap, 32-phase filter for image and video scaling at 1080p
  - Monochrome expansion for text rendering
  - 32K × 32K coordinate system

Dual-Core PowerVR® SGX544™ 3D GPU The 3D graphics processing unit (GPU) subsystem is based on POWERVR® SGX544 subsystem from Imagination Technologies. It supports general embedded applications. The GPU can process different data types simultaneously, such as: pixel data, vertex data, video data, and general-purpose data. The GPU subsystem has the following features:

- Multicore GPU architecture: two SGX544 cores.
- Shared system level cache of 128 KiB
- Tile-based deferred rendering architecture
- Second-generation universal scalable shader engines (USSE2), multithreaded engines incorporating pixel and vertex shader functionality
- Present and texture load accelerators
  - Enables to move, rotate, twiddle, and scale texture surfaces.
  - Supports RGB, ARGB, YUV422, and YUV420 surface formats.
  - Supports bilinear upscale.
  - Supports source colorkey.
- Fine-grained task switching, load balancing, and power management
- Programmable high-quality image antialiasing
- Bilinear, trilinear, anisotropic texture filtering
- Advanced geometry DMA driven operation for minimum CPU interaction
- Fully virtualized memory addressing for OS operation in a unified memory architecture (MMU)

3.3 Memory

3.3.1 1GB DDR3L

Dual 256M x 16 DDR3L memory devices are used, one on each side of the board, for a total of 1 GB. They will each operate at a clock frequency of up to 533 MHz yielding an effective rate of 1066Mb/s on the DDR3L bus allowing for 4GB/s of DDR3L memory bandwidth.

3.3.2 16GB Embedded MMC

A single 16GB embedded MMC (eMMC) device is on the board.

3.3.3 microSD Connector

The board is equipped with a single microSD connector to act as a secondary boot source for the board and, if selected as such, can be the primary booth source. The connector will support larger capacity microSD cards. The microSD card is not provided with the board.

3.4 Boot Modes

Todo: Need info on BBAI boot mode settings

3.5 Power Management

Todo: Need info on BBAI power management

3.6 Connectivity

Todo: Add WiFi/Bluetooth/Ethernet

BeagleBone® AI supports the majority of the functions of the AM5729 SOC through connectors or expansion header pin accessibility. See section 7 for more information on expansion header pinouts. There are a few functions that are not accessible which are: (TBD)

Todo: This text needs to go somewhere.

Table 3.1: On-board I2C Devices

Address	Identifier	Description
0x12	U3	TPS6590379 PMIC DVS
0x41	U78	STMPE811Q ADC and GPIO expander
0x47	U13	HD3SS3220 USB Type-C DRP port controller
0x50	U9	24LC32 board ID EEPROM
0x58	U3	TPS6590379 PMIC power registers
0x5a	U3	TPS6590379 PMIC interfaces and auxiliaries
0x5c	U3	TPS6590379 PMIC trimming and test
0x5e	U3	TPS6590379 PMIC OTP

3.7 Power Section

Figure ? is the high level block diagram of the power section of the board.

(Block Diagram for Power)

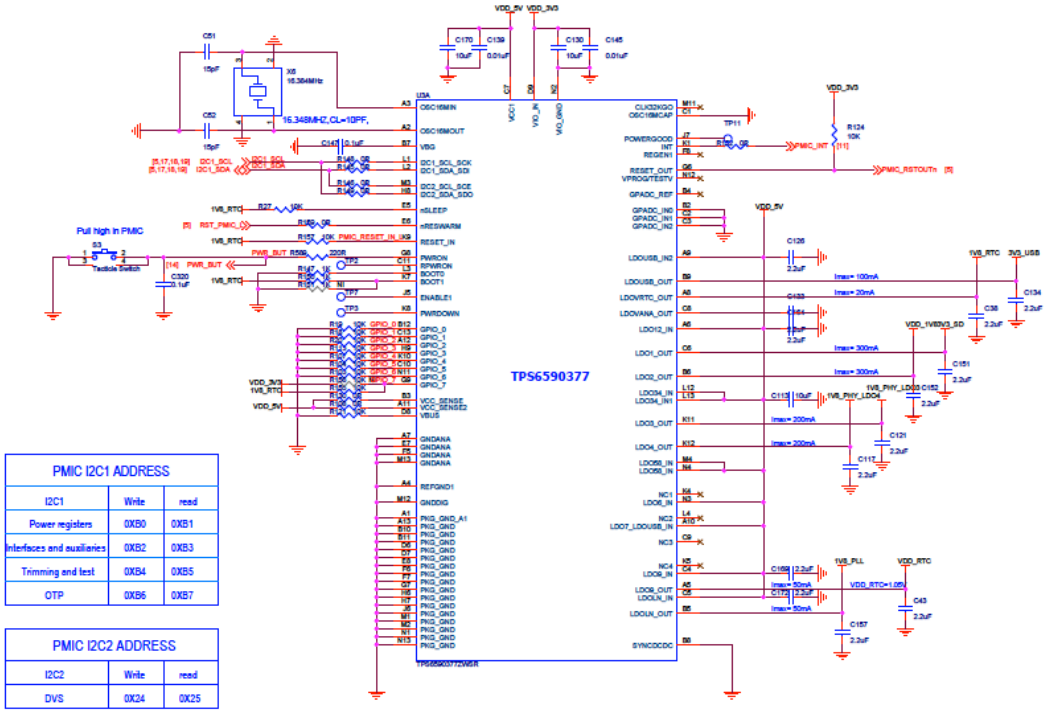
3.7.1 TPS6590379 PMIC

The Texas Instruments TPS6590379ZWSR device is an integrated power-management IC (PMIC) specifically designed to work well ARM Cortex A15 Processors, such as the AM5729 used on BeagleBone® AI. The datasheet is located here https://www.ti.com/lit/ds/symlink/tps659037.pdf

The device provides seven configurable step-down converters with up to 6 A of output current for memory, processor core, input-output (I/O), or preregulation of LDOs. One of these configurable step-down converters can be combined with another 3-A regulator to allow up to 9 A of output current. All of the step-down converters can synchronize to an external clock source between 1.7 MHz and 2.7 MHz, or an internal fallback clock at 2.2 MHz.

The TPS659037 device contains seven LDO regulators for external use. These LDO regulators can be supplied from either a system supply or a preregulated supply. The power-up and power-down controller is configurable and supports any power-up and power-down sequences (OTP based). The TPS659037 device includes a 32-kHz RC oscillator to sequence all resources during power up and power down. In cases where a fast start up is needed, a 16-MHz crystal oscillator is also included to quickly generate a stable 32-kHz for the system. All LDOs and SMPS converters can be controlled by the SPI or I2C interface, or by power request signals. In addition, voltage scaling registers allow transitioning the SMPS to different voltages by SPI, I2C, or roof and floor control.

One dedicated pin in each package can be configured as part of the power-up sequence to control external resources. General-purpose input-output (GPIO) functionality is available and two GPIOs can be configured as part of the power-up sequence to control external resources. Power request signals enable power mode control for power optimization. The device includes a general-purpose sigma-delta analog-to-digital converter (GPADC) with three external input channels.



PMIC I2C1 ADDRESS		
I2C1	Write	read
Power registers	0XB0	0XB1
Interfaces and auxiliaries	0XB2	0XB3
Trimming and test	0XB4	0XB5
OTP	0XB6	0XB7

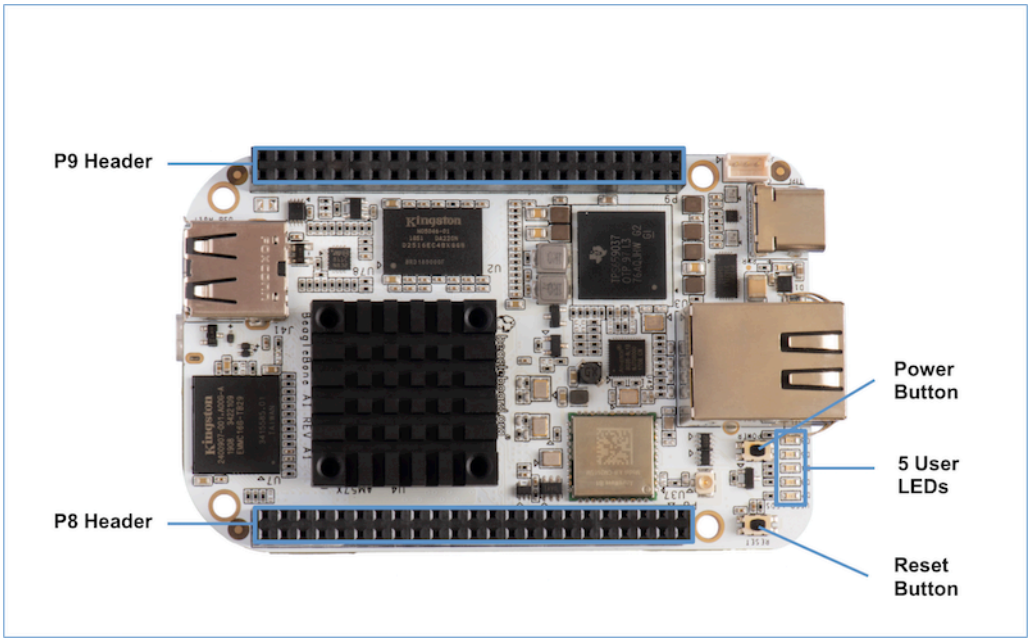
PMIC I2C2 ADDRESS		
I2C2	Write	read
DVS	0X24	0X25

3.7.2 USB-C Power

Below image shows how the USB-C power input is connected to the **TPS6590379**.

Todo: (Schematic screenshot)

3.7.3 Power Button



3.8 eMMC Flash Memory (16GB)

3.8.1 eMMC Device

3.8.2 eMMC Circuit Design

3.8.3 Board ID

A board identifier is placed on the eMMC in the second linear boot partition (/dev/mmcblk1boot1). Reserved bytes up to 32k (0x8000) are filled with "FF".

Table 3.2: Board ID

Name	Size (bytes)	Contents
Header	4	MSB 0xEE3355AA LSB (stored LSB first)
Board Name	8	Name for board in ASCII "BBONE-AI" = BeagleBone AI
Version	4	Hardware version code for board in ASCII "00A1" = rev. A1
Serial Number	14	Serial number of the board. This is a 14 character string which is: WWYYEMAInnnnnn where: • WW = 2 digit week of the year of production • YY = 2 digit year of production • EM = Embest • AI = BeagleBone AI • nnnnnn = incrementing board number

```
debian@beaglebone:~$ sudo hexdump -C /dev/mmcblk1boot1
00000000  aa 55 33 ee 42 42 4f 4e  45 2d 41 49 30 30 41 31  |.U3.BBONE-
↪AI00A1|
00000010  31 39 33 33 45 4d 41 49  30 30 30 38 30 33 ff ff  |1933EMAI000803..
↪|
00000020  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................
↪|
*
00008000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................
↪|
*
00400000
```

3.9 Wireless Communication: 802.11 ac & Bluetooth: AzureWave AW-CM256SM

Datasheet https://storage.googleapis.com/wzukusers/user-26561200/documents/5b7d0fe3c3f29Ct6k0QI/AW-CM256SM_DS_Rev%2015_CYW.pdf Wireless connectivity is provided on BeagleBone® AI via the AzureWave Technologies AW-CM256SM IEEE 802.11a/b/g/n/ac Wi-Fi with Bluetooth 4.2 Combo Stamp Module.

This highly integrated wireless local area network (WLAN) solution combines Bluetooth 4.2 and provides a complete 2.4GHz Bluetooth system which is fully compliant to Bluetooth 4.2 and v2.1 that supports EDR of 2Mbps and 3Mbps for data and audio communications. It enables a high performance, cost effective, low power, compact solution that easily fits onto the SDIO and UART combo stamp module.

Compliant with the IEEE 802.11a/b/g/n/ac standard, AW-CM256SM uses Direct Sequence Spread Spectrum (DSSS), Orthogonal Frequency Division Multiplexing (OFDM), BPSK, QPSK, CCK and QAM baseband modulation technologies. Compare to 802.11n technology, 802.11ac provides a big improvement on speed and range.

The AW-CM256SM module adopts a Cypress solution. The module design is based on the Cypress CYP43455 single chip.

3.9.1 WLAN on the AzureWave AW-CM256SM

High speed wireless connection up to 433.3Mbps transmit/receive PHY rate using 80MHz bandwidth,

- 1 antennas to support 1(Transmit) and 1(Receive) technology and Bluetooth
- WCS (Wireless Coexistence System)
- Low power consumption and high performance
- Enhanced wireless security
- Fully speed operation with Piconet and Scatternet support
- 12mm(L) x 12mm(W) x1.65mm(H) LGA package
- Dual - band 2.4 GHz and 5GHz 802.11 a/b/g/n/ac
- External Crystal

3.9.2 Bluetooth on the AzureWave AW-CM256S

- 1 antennas to support 1(Transmit) and 1(Receive) technology and Bluetooth
- Fully qualified Bluetooth BT4.2
- Enhanced Data Rate(EDR) compliant for both 2Mbps and 3Mbps supported
- High speed UART and PCM for Bluetooth

3.10 HDMI

The HDMI interface is aligned with the HDMI TMDS single stream standard v1.4a (720p @60Hz to 1080p @24Hz) and the HDMI v1.3 (1080p @60Hz): 3 data channels, plus 1 clock channel is supported (differential).

Todo: Verify it isn't better than this. Doesn't seem right.

3.11 PRU-ICSS

The Texas Instruments AM5729 Sitara™ provides 2 Programmable Real-Time Unit Subsystem and Industrial Communciation Subsystems. (PRU-ICSS1 and PRU-ICSS2).

Within each PRU-ICSS are dual 32-bit Load / Store RISC CPU cores: Programmable Real-Time Units (PRU0 and PRU1), shared data and instruction memories, internal peripheral modules and an interrupt controller. Therefore the SoC is providing a total of 4 PRU 32-bit RISC CPU's:

- PRU-ICSS1 PRU0
- PRU-ICSS1 PRU1
- PRU-ICSS2 PRU0
- PRU-ICSS2 PRU1

The programmable nature of the PRUs, along with their access to pins, events and all SoC resources, provides flexibility in implementing fast real-time responses, specialized data handling operations, peripheral interfaces and in off-loading tasks from the other processor cores of the SoC.

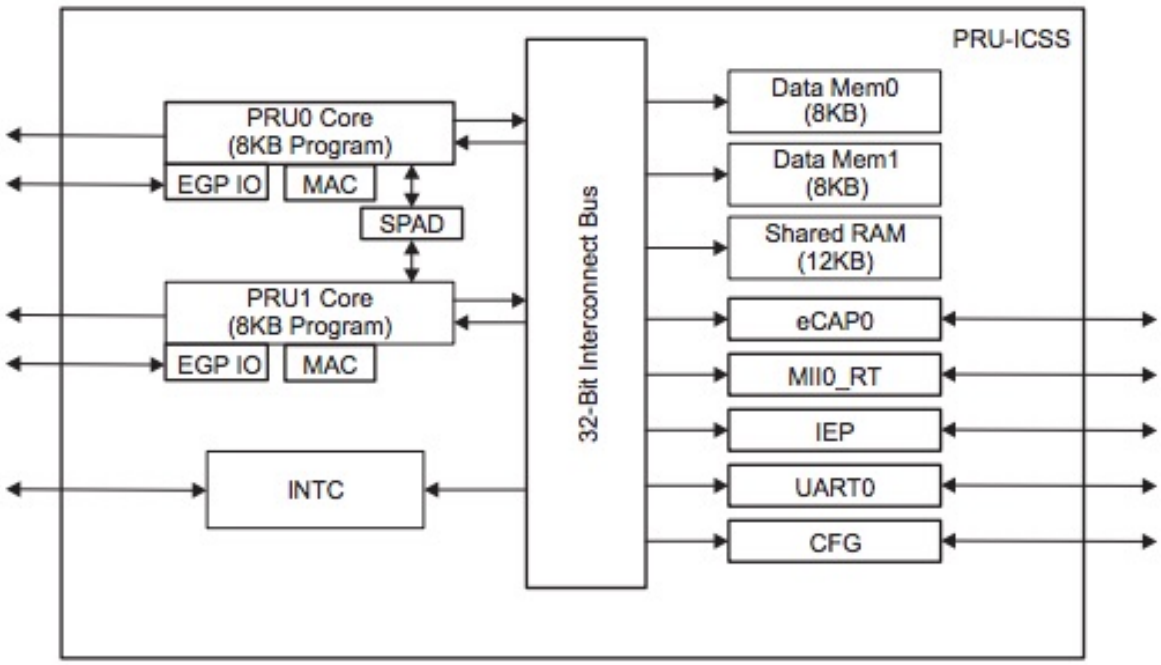
3.11.1 PRU-ICSS Features

Each of the 2 PRU-ICSS (PRU-ICSS1 and PRU-ICSS2) includes the following main features:

- 2 Independent programmable real-time (PRU) cores (PRU0 and PRU1)
- 21x Enhanced GPIs (EGPIs) and 21x Enhanced GPOs (EGPOs) with asynchronous capture and serial support per each PRU CPU core
- One Ethernet MII_RT module (PRU-ICSS_MII_RT) with two MII ports and configurable connections to PRUs
- 1 MDIO Port (PRU-ICSS_MII_MDIO)
- One Industrial Ethernet Peripheral (IEP) to manage/generate Industrial Ethernet functions
- 1 x 16550-compatible UART with a dedicated 192 MHz clock to support 12Mbps Profibus
- 1 Industrial Ethernet timer with 7/9 capture and 8 compare events
- 1 Enhanced Capture Module (ECAP)
- 1 Interrupt Controller (PRU-ICSS_INTC)
- A flexible power management support
- Integrated switched central resource with programmable priority
- Parity control supported by all memories

3.11.2 PRU-ICSS Block Diagram

Below is a high level block diagram of one of the PRU-ICSS Subsystems



3.12 PRU-ICSS Resources and FAQ's

Resources

- Great resources for PRU and BeagleBone® has been compiled here https://beagleboard.org/pru
- The PRU Cookbook provides examples and getting started information pru-cookbook-home
- Detailed specification is available at http://processors.wiki.ti.com/index.php/PRU-ICSS

FAQ

- Q: Is it possible to configure the Ethernet MII to be accessed via a PRU MII?
- A: TBD

3.12.1 PRU-ICSS1 Pin Access

The table below shows which PRU-ICSS1 signals can be accessed on BeagleBone® AI and on which connector and pins they are accessible from. Some signals are accessible on the same pins. Signal Names reveal which PRU-ICSS Subsystem is being addressed. pr1 is PRU-ICSS1 and pr2 is PRU-ICSS2

Table 3.3: PRU-ICSS1 Pin Access

SIGNAL NAME	DESCRIPTION	TYPE	PROC	HEADER_PIN	MODE	HEADER_PIN	MODE
pr1_pru0_gpo0	PRU0 General-Purpose Output	O	AH6	NA			
pr1_pru0_gpo1	PRU0 General-Purpose Output	O	AH3	NA			
pr1_pru0_gpo2	PRU0 General-Purpose Output	O	AH5	NA			
pr1_pru0_gpo3	PRU0 General-Purpose Output	O	AG6	P8_12	MODE13		
pr1_pru0_gpo4	PRU0 General-Purpose Output	O	AH4	P8_11	MODE13		
pr1_pru0_gpo5	PRU0 General-Purpose Output	O	AG4	P9_15	MODE13		
pr1_pru0_gpo6	PRU0 General-Purpose Output	O	AG2	NA			
pr1_pru0_gpo7	PRU0 General-Purpose Output	O	AG3	NA			
pr1_pru0_gpo8	PRU0 General-Purpose Output	O	AG5	NA			
pr1_pru0_gpo9	PRU0 General-Purpose Output	O	AF2	NA			
pr1_pru0_gpo10	PRU0 General-Purpose Output	O	AF6	NA			
pr1_pru0_gpo11	PRU0 General-Purpose Output	O	AF3	NA			
pr1_pru0_gpo12	PRU0 General-Purpose Output	O	AF4	NA			
pr1_pru0_gpo13	PRU0 General-Purpose Output	O	AF1	NA			
pr1_pru0_gpo14	PRU0 General-Purpose Output	O	AE3	NA			
pr1_pru0_gpo15	PRU0 General-Purpose Output	O	AE5	NA			
pr1_pru0_gpo16	PRU0 General-Purpose Output	O	AE1	NA			
pr1_pru0_gpo17	PRU0 General-Purpose Output	O	AE2	P9_26	MODE13		
pr1_pru0_gpo18	PRU0 General-Purpose Output	O	AE6	NA			
pr1_pru0_gpo19	PRU0 General-Purpose Output	O	AD2	NA			
pr1_pru0_gpo20	PRU0 General-Purpose Output	O	AD3	NA			
pr1_pru0_gpi0	PRU0 General-Purpose Input	I	AH6	NA			
pr1_pru0_gpi1	PRU0 General-Purpose Input	I	AH3	NA			
pr1_pru0_gpi2	PRU0 General-Purpose Input	I	AH5	NA			
pr1_pru0_gpi3	PRU0 General-Purpose Input	I	AG6	P8_12	MODE12		
pr1_pru0_gpi4	PRU0 General-Purpose Input	I	AH4	P8_11	MODE12		
pr1_pru0_gpi5	PRU0 General-Purpose Input	I	AG4	P9_15	MODE12		
pr1_pru0_gpi6	PRU0 General-Purpose Input	I	AG2	NA			
pr1_pru0_gpi7	PRU0 General-Purpose Input	I	AG3	NA			
pr1_pru0_gpi8	PRU0 General-Purpose Input	I	AG5	NA			
pr1_pru0_gpi9	PRU0 General-Purpose Input	I	AF2	NA			
pr1_pru0_gpi10	PRU0 General-Purpose Input	I	AF6	NA			
pr1_pru0_gpi11	PRU0 General-Purpose Input	I	AF3	NA			
pr1_pru0_gpi12	PRU0 General-Purpose Input	I	AF4	NA			
pr1_pru0_gpi13	PRU0 General-Purpose Input	I	AF1	NA			
pr1_pru0_gpi14	PRU0 General-Purpose Input	I	AE3	NA			

continues on next page

Table 3.3 – continued from previous page

SIGNAL NAME	DESCRIPTION	TYPE	PROC	HEADER_PIN	MODE	HEADER_PIN	MODE
pr1_pru0_gpi15	PRU0 General-Purpose Input	I	AE5	NA			
pr1_pru0_gpi16	PRU0 General-Purpose Input	I	AE1	NA			
pr1_pru0_gpi17	PRU0 General-Purpose Input	I	AE2	P9_26	MODE12		
pr1_pru0_gpi18	PRU0 General-Purpose Input	I	AE6	NA			
pr1_pru0_gpi19	PRU0 General-Purpose Input	I	AD2	NA			
pr1_pru0_gpi20	PRU0 General-Purpose Input	I	AD3	NA			
pr1_pru1_gpo0	PRU1 General-Purpose Output	O	E2	NA			
pr1_pru1_gpo1	PRU1 General-Purpose Output	O	D2	P9_20	MODE13		
pr1_pru1_gpo2	PRU1 General-Purpose Output	O	F4	P9_19	MODE13		
pr1_pru1_gpo3	PRU1 General-Purpose Output	O	C1	P9_41	MODE13		
pr1_pru1_gpo4	PRU1 General-Purpose Output	O	E4	NA			
pr1_pru1_gpo5	PRU1 General-Purpose Output	O	F5	P8_18	MODE13		
pr1_pru1_gpo6	PRU1 General-Purpose Output	O	E6	P8_19	MODE13		
pr1_pru1_gpo7	PRU1 General-Purpose Output	O	D3	P8_13	MODE13		
pr1_pru1_gpo8	PRU1 General-Purpose Output	O	F6	NA			
pr1_pru1_gpo9	PRU1 General-Purpose Output	O	D5	P8_14	MODE13		
pr1_pru1_gpo10	PRU1 General-Purpose Output	O	C2	P9_42	MODE13		
pr1_pru1_gpo11	PRU1 General-Purpose Output	O	C3	P9_27	MODE13		
pr1_pru1_gpo12	PRU1 General-Purpose Output	O	C4	NA			
pr1_pru1_gpo13	PRU1 General-Purpose Output	O	B2	NA			
pr1_pru1_gpo14	PRU1 General-Purpose Output	O	D6	P9_14	MODE13		
pr1_pru1_gpo15	PRU1 General-Purpose Output	O	C5	P9_16	MODE13		
pr1_pru1_gpo16	PRU1 General-Purpose Output	O	A3	P8_15	MODE13		
pr1_pru1_gpo17	PRU1 General-Purpose Output	O	B3	P8_26	MODE13		
pr1_pru1_gpo18	PRU1 General-Purpose Output	O	B4	P8_16	MODE13		
pr1_pru1_gpo19	PRU1 General-Purpose Output	O	B5	NA			
pr1_pru1_gpo20	PRU1 General-Purpose Output	O	A4	NA			
pr1_pru1_gpi0	PRU1 General-Purpose Input	I	E2	NA			
pr1_pru1_gpi1	PRU1 General-Purpose Input	I	D2	P9_20	MODE12		
pr1_pru1_gpi2	PRU1 General-Purpose Input	I	F4	P9_19	MODE12		
pr1_pru1_gpi3	PRU1 General-Purpose Input	I	C1	P9_41	MODE12		
pr1_pru1_gpi4	PRU1 General-Purpose Input	I	E4	NA			
pr1_pru1_gpi5	PRU1 General-Purpose Input	I	F5	P8_18	MODE12		
pr1_pru1_gpi6	PRU1 General-Purpose Input	I	E6	P8_19	MODE12		
pr1_pru1_gpi7	PRU1 General-Purpose Input	I	D3	P8_13	MODE12		
pr1_pru1_gpi8	PRU1 General-Purpose Input	I	F6	NA			
pr1_pru1_gpi9	PRU1 General-Purpose Input	I	D5	P8_14	MODE12		

continues on next page

Table 3.3 – continued from previous page

SIGNAL NAME	DESCRIPTION	TYPE	PROC	HEADER_PIN	MODE	HEADER_PIN	MODE
pr1_pru1_gpi10	PRU1 General-Purpose Input	I	C2	P9_42	MODE12		
pr1_pru1_gpi11	PRU1 General-Purpose Input	I	C3	P9_27	MODE12		
pr1_pru1_gpi12	PRU1 General-Purpose Input	I	C4	NA			
pr1_pru1_gpi13	PRU1 General-Purpose Input	I	B2	NA			
pr1_pru1_gpi14	PRU1 General-Purpose Input	I	D6	P9_14	MODE12		
pr1_pru1_gpi15	PRU1 General-Purpose Input	I	C5	P9_16	MODE12		
pr1_pru1_gpi16	PRU1 General-Purpose Input	I	A3	P8_15	MODE12		
pr1_pru1_gpi17	PRU1 General-Purpose Input	I	B3	P8_26	MODE12		
pr1_pru1_gpi18	PRU1 General-Purpose Input	I	B4	P8_16	MODE12		
pr1_pru1_gpi19	PRU1 General-Purpose Input	I	B5	NA			
pr1_pru1_gpi20	PRU1 General-Purpose Input	I	A4	NA			
pr1_mii_mt0_clk	MII0 Transmit Clock	I	U5	NA			
pr1_mii0_txen	MII0 Transmit Enable	O	V3	NA			
pr1_mii0_txd3	MII0 Transmit Data	O	V5	NA			
pr1_mii0_txd2	MII0 Transmit Data	O	V4	NA			
pr1_mii0_txd1	MII0 Transmit Data	O	Y2	NA			
pr1_mii0_txd0	MII0 Transmit Data	O	W2	NA			
pr1_mii0_rxdv	MII0 Data Valid	I	V2	NA			
pr1_mii_mr0_clk	MII0 Receive Clock	I	Y1	NA			
pr1_mii0_rxd3	MII0 Receive Data	I	W9	NA			
pr1_mii0_rxd2	MII0 Receive Data	I	V9	NA			
pr1_mii0_crs	MII0 Carrier Sense	I	V7	NA			
pr1_mii0_rxer	MII0 Receive Error	I	U7	NA			
pr1_mii0_rxd1	MII0 Receive Data	I	V6	NA			
pr1_mii0_rxd0	MII0 Receive Data	I	U6	NA			
pr1_mii0_col	MII0 Collision Detect	I	V1	NA			
pr1_mii0_rxlink	MII0 Receive Link	I	U4	NA			
pr1_mii_mt1_clk	MII1 Transmit Clock	I	C1	P9_41	MODE11		
pr1_mii1_txen	MII1 Transmit Enable	O	E4	NA			
pr1_mii1_txd3	MII1 Transmit Data	O	F5	P8_18	MODE11		
pr1_mii1_txd2	MII1 Transmit Data	O	E6	P8_19	MODE11		
pr1_mii1_txd1	MII1 Transmit Data	O	D5	P8_14	MODE11		
pr1_mii1_txd0	MII1 Transmit Data	O	C2	P9_42	MODE11		
pr1_mii_mr1_clk	MII1 Receive Clock	I	C3	P9_27	MODE11		
pr1_mii1_rxdv	MII1 Data Valid	I	C4	NA			
pr1_mii1_rxd3	MII1 Receive Data	I	B2	NA			
pr1_mii1_rxd2	MII1 Receive Data	I	D6	P9_14	MODE11		

continues on next page

Table 3.3 – continued from previous page

SIGNAL NAME	DESCRIPTION	TYPE	PROC	HEADER_PIN	MODE	HEADER_PIN	MODE
pr1_mii1_rxd1	MII1 Receive Data	I	C5	P9_16	MODE11		
pr1_mii1_rxd0	MII1 Receive Data	I	A3	P8_15	MODE11		
pr1_mii1_rxer	MII1 Receive Error	I	B3	P8_26	MODE11		
pr1_mii1_rxlink	MII1 Receive Link	I	B4	P8_16	MODE11		
pr1_mii1_col	MII1 Collision Detect	I	B5	NA			
pr1_mii1_crs	MII1 Carrier Sense	I	A4	NA			
pr1_mdio_mdclk	MDIO Clock	O	D3	P8_13	MODE11		
pr1_mdio_data	MDIO Data	IO	F6	NA			
pr1_edc_latch0_in	Latch Input 0	I	AG3 / E2	NA			
pr1_edc_latch1_in	Latch Input 1	I	AG5	NA			
pr1_edc_sync0_out	SYNC0 Output	O	AF2 / D2	P9_20	MODE11		
pr1_edc_sync1_out	SYNC1 Output	O	AF6	NA			
pr1_edio_latch_in	Latch Input	I	AF3	NA			
pr1_edio_sof	Start Of Frame	O	AF4 / F4	P9_19	MODE11		
pr1_edio_data_in0	Ethernet Digital Input	I	AF1 / E1	NA			
pr1_edio_data_in1	Ethernet Digital Input	I	AE3 / G2	NA			
pr1_edio_data_in2	Ethernet Digital Input	I	AE5 / H7	NA			
pr1_edio_data_in3	Ethernet Digital Input	I	AE1 / G1	NA			
pr1_edio_data_in4	Ethernet Digital Input	I	AE2 / G6	P9_26	MODE10	P8_34	MODE12
pr1_edio_data_in5	Ethernet Digital Input	I	AE6 / F2	P8_36	MODE12		
pr1_edio_data_in6	Ethernet Digital Input	I	AD2 / F3	NA			
pr1_edio_data_in7	Ethernet Digital Input	I	AD3 / D1	P8_15	MODE12		
p r1_edio_data_out0	Ethernet Digital Output	O	AF1 / E1	NA			
p r1_edio_data_out1	Ethernet Digital Output	O	AE3 / G2	NA			
p r1_edio_data_out2	Ethernet Digital Output	O	AE5 / H7	NA			
p r1_edio_data_out3	Ethernet Digital Output	O	AE1 / G1	NA			
p r1_edio_data_out4	Ethernet Digital Output	O	AE2 / G6	P9_26	MODE11	P8_34	MODE13
p r1_edio_data_out5	Ethernet Digital Output	O	AE6 / F2	P8_36	MODE13		
p r1_edio_data_out6	Ethernet Digital Output	O	AD2 / F3	NA			
p r1_edio_data_out7	Ethernet Digital Output	O	AD3 / D1	P8_15	MODE13		
pr1_uart0_cts_n	UART Clear-To-Send	I	G1 / F11	P8_45	MODE10		
pr1_uart0_rts_n	UART Ready-To-Send	O	G6 / G10	P8_34	MODE11	P8_46	MODE10
pr1_uart0_rxd	UART Receive Data	I	F2 / F10	P8_36	MODE11	P8_43	MODE10
pr1_uart0_txd	UART Transmit Data	O	F3 / G11	P8_44	MODE10		
pr1_ecap0_ ecap_capin_apwm_o	Capture Input/PWM Output	IO	D1 / E9	P8_15	MODE11	P8_41	MODE10

3.12.2 PRU-ICSS2 Pin Access

The table below shows which PRU-ICSS2 signals can be accessed on BeagleBone® AI and on which connector and pins they are accessible from. Some signals are accessible on the same pins. Signal Names reveal which PRU-ICSS Subsystem is being addressed. pr1 is PRU-ICSS1 and pr2 is PRU-ICSS2

Table 3.4: PRU-ICSS2 Pin Access

SIGNAL NAME	DESCR IPTION	TYPE	PROC	HEAD ER_PIN	MODE	HEAD ER_PIN	MODE
p r2_pru 0_gpo0	PRU0 Gen eral-P urpose Output	O	G 11/AC5	P8_44	MODE13		
p r2_pru 0_gpo1	PRU0 Gen eral-P urpose Output	O	E9/AB4	P8_41	MODE13		
p r2_pru 0_gpo2	PRU0 Gen eral-P urpose Output	O	F9/AD4	P8_42	MODE13	P8_21	MODE13
p r2_pru 0_gpo3	PRU0 Gen eral-P urpose Output	O	F8/AC4	P8_39	MODE13	P8_20	MODE13
p r2_pru 0_gpo4	PRU0 Gen eral-P urpose Output	O	E7/AC7	P8_40	MODE13	P8_25	MODE13
p r2_pru 0_gpo5	PRU0 Gen eral-P urpose Output	O	E8/AC6	P8_37	MODE13	P8_24	MODE13
p r2_pru 0_gpo6	PRU0 Gen eral-P urpose Output	O	D9/AC9	P8_38	MODE13	P8_5	MODE13
p r2_pru 0_gpo7	PRU0 Gen eral-P urpose Output	O	D7/AC3	P8_36	MODE13	P8_6	MODE13
p r2_pru 0_gpo8	PRU0 Gen eral-P urpose Output	O	D8/AC8	P8_34	MODE13	P8_23	MODE13
p r2_pru 0_gpo9	PRU0 Gen eral-P urpose Output	O	A5/AD6	P8_35	MODE13	P8_22	MODE13
pr 2_pru0 _gpo10	PRU0 Gen eral-P urpose Output	O	C6/AB8	P8_33	MODE13	P8_3	MODE13
pr 2_pru0 _gpo11	PRU0 Gen eral-P urpose Output	O	C8/AB5	P8_31	MODE13	P8_4	MODE13
pr 2_pru0 _gpo12	PRU0 Gen eral-P urpose Output	O	C7/B18	P8_32	MODE13		
pr 2_pru0 _gpo13	PRU0 Gen eral-P urpose Output	O	B7/F15	P8_45	MODE13		
pr 2_pru0 _gpo14	PRU0 Gen eral-P urpose Output	O	B8/B19	P9_11	MODE13	P9_11	MODE13
pr 2_pru0 _gpo15	PRU0 Gen eral-P urpose Output	O	A7/C17	P8_17	MODE13	P9_13	MODE13
pr 2_pru0 _gpo16	PRU0 Gen eral-P urpose Output	O	A8/C15	P8_27	MODE13		
pr 2_pru0 _gpo17	PRU0 Gen eral-P urpose Output	O	C9/A16	P8_28	MODE13		
pr 2_pru0 _gpo18	PRU0 Gen eral-P urpose Output	O	A9/A19	P8_29	MODE13		
pr 2_pru0 _gpo19	PRU0 Gen eral-P urpose Output	O	B9/A18	P8_30	MODE13		
pr 2_pru0 _gpo20	PRU0 Gen eral-P urpose Output	O	A 10/F14	P8_46	MODE13	P8_8	MODE13
p r2_pru 0_gpi0	PRU0 Gen eral-P urpose Input	I	G 11/AC5	P8_44	MODE12		
p r2_pru 0_gpi1	PRU0 Gen eral-P urpose Input	I	E9/AB4	P8_41	MODE12		
p r2_pru 0_gpi2	PRU0 Gen eral-P urpose Input	I	F9/AD4	P8_42	MODE12	P8_21	MODE12
p r2_pru 0_gpi3	PRU0 Gen eral-P urpose Input	I	F8/AC4	P8_39	MODE12	P8_20	MODE12
p r2_pru 0_gpi4	PRU0 Gen eral-P urpose Input	I	E7/AC7	P8_40	MODE12	P8_25	MODE12
p r2_pru 0_gpi5	PRU0 Gen eral-P urpose Input	I	E8/AC6	P8_37	MODE12	P8_24	MODE12
p r2_pru 0_gpi6	PRU0 Gen eral-P urpose Input	I	D9/AC9	P8_38	MODE12	P8_5	MODE12
p r2_pru 0_gpi7	PRU0 Gen eral-P urpose Input	I	D7/AC3	P8_36	MODE12	P8_6	MODE12
p r2_pru 0_gpi8	PRU0 Gen eral-P urpose Input	I	D8/AC8	P8_34	MODE12	P8_23	MODE12
p r2_pru 0_gpi9	PRU0 Gen eral-P urpose Input	I	A5/AD6	P8_35	MODE12	P8_22	MODE12
pr 2_pru0 _gpi10	PRU0 Gen eral-P urpose Input	I	C6/AB8	P8_33	MODE12	P8_3	MODE12
pr 2_pru0 _gpi11	PRU0 Gen eral-P urpose Input	I	C8/AB5	P8_31	MODE12	P8_4	MODE12
pr 2_pru0 _gpi12	PRU0 Gen eral-P urpose Input	I	C7/B18	P8_32	MODE12		
pr 2_pru0 _gpi13	PRU0 Gen eral-P urpose Input	I	B7/F15	P8_45	MODE12		
pr 2_pru0 _gpi14	PRU0 Gen eral-P urpose Input	I	B8/B19	P9_11	MODE12	P9_11	MODE12

continues on next page

Table 3.4 – continued from previous page

SIGNAL NAME	DESCR IPTION	TYPE	PROC	HEAD ER_PIN	MODE	HEAD ER_PIN	MODE
pr 2_pru0 _gpi15	PRU0 Gen eral-P urpose Input	I	A7/C17	P8_17	MODE12	P9_13	MODE12
pr 2_pru0 _gpi16	PRU0 Gen eral-P urpose Input	I	A8/C15	P8_27	MODE12		
pr 2_pru0 _gpi17	PRU0 Gen eral-P urpose Input	I	C9/A16	P8_28	MODE12		
pr 2_pru0 _gpi18	PRU0 Gen eral-P urpose Input	I	A9/A19	P8_29	MODE12		
pr 2_pru0 _gpi19	PRU0 Gen eral-P urpose Input	I	B9/A18	P8_30	MODE12		
pr 2_pru0 _gpi20	PRU0 Gen eral-P urpose Input	I	A 10/F14	P8_46	MODE12	P8_8	MODE12
p r2_pru 1_gpo0	PRU1 Gen eral-P urpose Output	O	V1/D17	P8_32	MODE13		
p r2_pru 1_gpo1	PRU1 Gen eral-P urpose Output	O	U4/AA3	NA			
p r2_pru 1_gpo2	PRU1 Gen eral-P urpose Output	O	U3/AB9	NA			
p r2_pru 1_gpo3	PRU1 Gen eral-P urpose Output	O	V2/AB3	NA			
p r2_pru 1_gpo4	PRU1 Gen eral-P urpose Output	O	Y1/AA4	NA			
p r2_pru 1_gpo5	PRU1 Gen eral-P urpose Output	O	W9/D18	P9_25	MODE13		
p r2_pru 1_gpo6	PRU1 Gen eral-P urpose Output	O	V9/E17	P8_9	MODE13		
p r2_pru 1_gpo7	PRU1 Gen eral-P urpose Output	O	V7/C14	P9_31	MODE13		
p r2_pru 1_gpo8	PRU1 Gen eral-P urpose Output	O	U7/G12	P9_18	MODE13		
p r2_pru 1_gpo9	PRU1 Gen eral-P urpose Output	O	V6/F12	P9_17	MODE13		
pr 2_pru1 _gpo10	PRU1 Gen eral-P urpose Output	O	U6/B12	P9_31	MODE13		
pr 2_pru1 _gpo11	PRU1 Gen eral-P urpose Output	O	U5/A11	P9_29	MODE13		
pr 2_pru1 _gpo12	PRU1 Gen eral-P urpose Output	O	V5/B13	P9_30	MODE13		
pr 2_pru1 _gpo13	PRU1 Gen eral-P urpose Output	O	V4/A12	P9_26	MODE13		
pr 2_pru1 _gpo14	PRU1 Gen eral-P urpose Output	O	V3/E14	P9_42	MODE13		
pr 2_pru1 _gpo15	PRU1 Gen eral-P urpose Output	O	Y2/A13	P8_10	MODE13		
pr 2_pru1 _gpo16	PRU1 Gen eral-P urpose Output	O	W2/G14	P8_7	MODE13		
pr 2_pru1 _gpo17	PRU1 Gen eral-P urpose Output	O	E11	P8_27	MODE13		
pr 2_pru1 _gpo18	PRU1 Gen eral-P urpose Output	O	F11	P8_45	MODE13		
pr 2_pru1 _gpo19	PRU1 Gen eral-P urpose Output	O	G10	P8_46	MODE13		
pr 2_pru1 _gpo20	PRU1 Gen eral-P urpose Output	O	F10	P8_43	MODE13		
p r2_pru 1_gpi0	PRU1 Gen eral-P urpose Input	I	V1/D17	P8_32	MODE12		
p r2_pru 1_gpi1	PRU1 Gen eral-P urpose Input	I	U4/AA3	NA			
p r2_pru 1_gpi2	PRU1 Gen eral-P urpose Input	I	U3/AB9	NA			
p r2_pru 1_gpi3	PRU1 Gen eral-P urpose Input	I	V2/AB3	NA			
p r2_pru 1_gpi4	PRU1 Gen eral-P urpose Input	I	Y1/AA4	NA			
p r2_pru 1_gpi5	PRU1 Gen eral-P urpose Input	I	W9/D18	P9_25	MODE12		
p r2_pru 1_gpi6	PRU1 Gen eral-P urpose Input	I	V9/E17	P8_9	MODE12		
p r2_pru 1_gpi7	PRU1 Gen eral-P urpose Input	I	V7/C14	P9_31	MODE12		
p r2_pru 1_gpi8	PRU1 Gen eral-P urpose Input	I	U7/G12	P9_18	MODE12		
p r2_pru 1_gpi9	PRU1 Gen eral-P urpose Input	I	V6/F12	P9_17	MODE12		

continues on next page

Table 3.4 – continued from previous page

SIGNAL NAME	DESCR IPTION	TYPE	PROC	HEAD ER_PIN	MODE	HEAD ER_PIN	MODE
pr 2_pru1 _gpi10	PRU1 Gen eral-P urpose Input	I	U6/B12	P9_31	MODE12		
pr 2_pru1 _gpi11	PRU1 Gen eral-P urpose Input	I	U5/A11	P9_29	MODE12		
pr 2_pru1 _gpi12	PRU1 Gen eral-P urpose Input	I	V5/B13	P9_30	MODE12		
pr 2_pru1 _gpi13	PRU1 Gen eral-P urpose Input	I	V4/A12	P9_28	MODE12		
pr 2_pru1 _gpi14	PRU1 Gen eral-P urpose Input	I	V3/E14	P9_42	MODE12		
pr 2_pru1 _gpi15	PRU1 Gen eral-P urpose Input	I	Y2/A13	P8_10	MODE12		
pr 2_pru1 _gpi16	PRU1 Gen eral-P urpose Input	I	W2/G14	P8_7	MODE12		
pr 2_pru1 _gpi17	PRU1 Gen eral-P urpose Input	I	E11	P8_27	MODE12		
pr 2_pru1 _gpi18	PRU1 Gen eral-P urpose Input	I	F11	P8_45	MODE12		
pr 2_pru1 _gpi19	PRU1 Gen eral-P urpose Input	I	G10	P8_46	MODE12		
pr 2_pru1 _gpi20	PRU1 Gen eral-P urpose Input	I	F10	P8_43	MODE12		
pr2_e dc_lat ch0_in	Latch Input 0	I	F9	P8_42	MODE10		
pr2_e dc_lat ch1_in	Latch Input 1	I	F8	P8_39	MODE10		
pr2_e dc_syn c0_out	SYNC0 Output	O	E7	P8_40	MODE10		
pr2_e dc_syn c1_out	SYNC1 Output	O	E8	P8_37	MODE10		
pr2_e dio_la tch_in	Latch Input	I	D9	P8_38	MODE10		
pr2_ed io_sof	Start Of Frame	O	D7	P8_36	MODE10		
pr2 _uart0 _cts_n	UART C lear-T o-Send	I	D8	P8_34	MODE10		
pr2 _uart0 _rts_n	UART R eady-T o-Send	O	A5	P8_35	MODE10		
p r2_uar t0_rxd	UART R eceive Data	I	C6	P8_33	MODE10		
p r2_uar t0_txd	UART Tr ansmit Data	O	C8	P8_31	MODE10		
pr2 _ecap0 _ecap_ capin_ apwm_o	C apture Inp ut/PWM output	IO	C7	P8_32	MODE10		
pr2_e dio_da ta_in0	Et hernet D igital Input	I	B7	P8_45	MODE10		
pr2_e dio_da ta_in1	Et hernet D igital Input	I	B8	P9_11	MODE10		
pr2_e dio_da ta_in2	Et hernet D igital Input	I	A7	P8_17	MODE10		
pr2_e dio_da ta_in3	Et hernet D igital Input	I	A8	P8_27	MODE10		
pr2_e dio_da ta_in4	Et hernet D igital Input	I	C9	P8_28	MODE10		
pr2_e dio_da ta_in5	Et hernet D igital Input	I	A9	P8_29	MODE10		
pr2_e dio_da ta_in6	Et hernet D igital Input	I	B9	P8_30	MODE10		
pr2_e dio_da ta_in7	Et hernet D igital Input	I	A10	P8_46	MODE10		
pr2_ed io_dat a_out0	Et hernet D igital Output	O	B7	P8_45	MODE11		
pr2_ed io_dat a_out1	Et hernet D igital Output	O	B8	P9_11	MODE11		
pr2_ed io_dat a_out2	Et hernet D igital Output	O	A7	P8_17	MODE11		
pr2_ed io_dat a_out3	Et hernet D igital Output	O	A8	P8_27	MODE11		
pr2_ed io_dat a_out4	Et hernet D igital Output	O	C9	P8_28	MODE11		
pr2_ed io_dat a_out5	Et hernet D igital Output	O	A9	P8_29	MODE11		
pr2_ed io_dat a_out6	Et hernet D igital Output	O	B9	P8_30	MODE11		

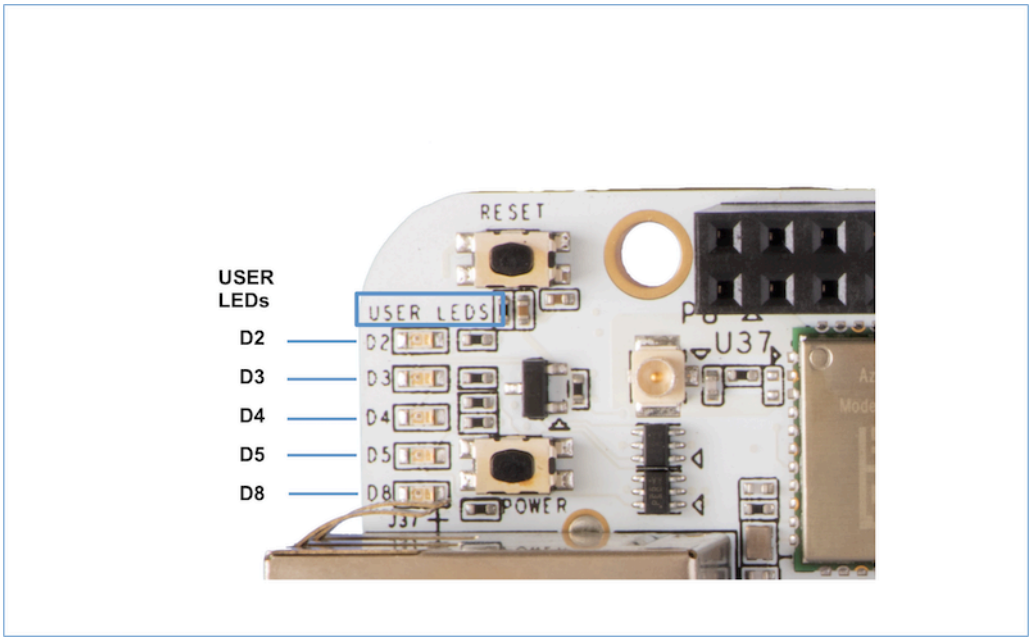
continues on next page

Table 3.4 – continued from previous page

SIGNAL NAME	DESCR IPTION	TYPE	PROC	HEAD ER_PIN	MODE	HEAD ER_PIN	MODE
pr2_ed io_dat a_out7	Et hernet D igital Output	O	A10	P8_46	MODE11		
pr2_mi i1_col	MII1 Col lision Detect	I	D18	P9_25	MODE11		
pr2_mi i1_crs	MII1 C arrier Sense	I	E17	P8_9	MODE11		
pr 2_mdio _mdclk	MDIO Clock	O	C 14/AB3	P9_31	MODE11		
p r2_mdi o_data	MDIO Data	IO	D 14/AA4	P9_29	MODE11		
p r2_mii 0_rxer	MII0 R eceive Error	I	G12	P9_18	MODE11		
pr2 _mii_m t0_clk	MII0 Tr ansmit Clock	I	F12	P9_17	MODE11		
p r2_mii 0_txen	MII0 Tr ansmit Enable	O	B12	P9_31	MODE11		
p r2_mii 0_txd3	MII0 Tr ansmit Data	O	A11	P9_29	MODE11		
p r2_mii 0_txd2	MII0 Tr ansmit Data	O	B13	P9_30	MODE11		
p r2_mii 0_txd1	MII0 Tr ansmit Data	O	A12	P9_28	MODE11		
p r2_mii 0_txd0	MII0 Tr ansmit Data	O	E14	P9_42	MODE11		
pr2 _mii_m r0_clk	MII0 R eceive Clock	I	A13	P8_10	MODE11		
p r2_mii 0_rxdv	MII0 Data Valid	I	G14	P8_7	MODE11		
p r2_mii 0_rxd3	MII0 R eceive Data	I	F14	P8_8	MODE11		
p r2_mii 0_rxd2	MII0 R eceive Data	I	A19	NA			
p r2_mii 0_rxd1	MII0 R eceive Data	I	A18	NA			
p r2_mii 0_rxd0	MII0 R eceive Data	I	C15	NA			
pr2 _mii0_ rxlink	MII0 R eceive Link	I	A16	NA			
pr2_mi i0_crs	MII0 C arrier Sense	I	B18	NA			
pr2_mi i0_col	MII0 Col lision Detect	I	F15	NA			
p r2_mii 1_rxer	MII1 R eceive Error	I	B19	P9_11	MODE11		
pr2 _mii1_ rxlink	MII1 R eceive Link	I	C17	P9_13	MODE11		
pr2 _mii_m t1_clk	MII1 Tr ansmit Clock	I	AC5	NA			
p r2_mii 1_txen	MII1 Tr ansmit Enable	O	AB4	NA			
p r2_mii 1_txd3	MII1 Tr ansmit Data	O	AD4	P8_21	MODE11		
p r2_mii 1_txd2	MII1 Tr ansmit Data	O	AC4	P8_20	MODE11		
p r2_mii 1_txd1	MII1 Tr ansmit Data	O	AC7	P8_25	MODE11		
p r2_mii 1_txd0	MII1 Tr ansmit Data	O	AC6	P8_24	MODE11		
pr2 _mii_m r1_clk	MII1 R eceive Clock	I	AC9	P8_5	MODE11		
p r2_mii 1_rxdv	MII1 Data Valid	I	AC3	P8_6	MODE11		
p r2_mii 1_rxd3	MII1 R eceive Data	I	AC8	P8_23	MODE11		
p r2_mii 1_rxd2	MII1 R eceive Data	I	AD6	P8_22	MODE11		
p r2_mii 1_rxd1	MII1 R eceive Data	I	AB8	P8_3	MODE11		
p r2_mii 1_rxd0	MII1 R eceive Data	I	AB5	P8_4	MODE11		
end	end	end	end	end	end	end	end

3.13 User LEDs

There are 5 User Programmable LEDs on BeagleBone® AI. These are connected to GPIO pins on the processor.



The table shows the signals used to control the LEDs from the processor. Each LED is user programmable. However, there is a Default Functions assigned in the device tree for BeagleBone® AI:

LED	GPIO SIGNAL	DEFAULT FUNCTION
D2	GPIO3_17	Heartbeat When Linux is Running
D3	GPIO5_5	microSD Activity
D4	GPIO3_15	CPU Activity
D5	GPIO3_14	eMMC Activity
D8	GPIO3_7	WiFi/Bluetooth Activity

Chapter 4

Expansion

4.1 Expansion Connectors

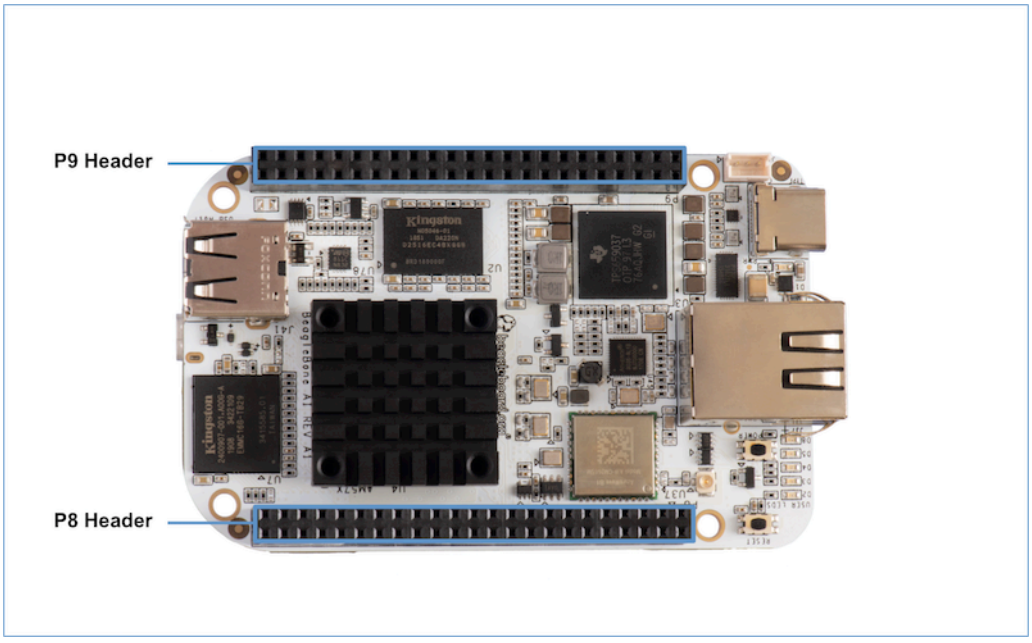
The expansion interface on the board is comprised of two 46 pin connectors, the P8 and P9 Headers. All signals on the expansion headers are **3.3V** unless otherwise indicated.

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.

Figure ? shows the location of the expansion connectors.



The location and spacing of the expansion headers are the same as on BeagleBone Black.

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.

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.

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.

Table 4.1: P8.01-P8.02

P8.01	P8.02
GND	GND

Table 4.2: P8.03-P8.05

	P8.03	P8.04	P8.05
GPIO	24	25	193
BALL	AB8	AB5	AC9
REG	0x179C	0x17A0	0x178C
MODE 0	mmc3_dat6	mmc3_dat7	mmc3_dat2
1	spi4_d0	spi4_cs0	spi3_cs0
2	uart10_ctsn	uart10_rtsn	uart5_ctsn
3			
4	vin2b_de1	vin2b_clk1	vin2b_d3
5			
6			
7			
8			
9	vin5a_hsync0	vin5a_vsync0	vin5a_d3
10	ehrpwm3_tripzone_input	eCAP3_in_PWM3_out	eQEP3_index
11	pr2_mii1_rxd1	pr2_mii1_rxd0	pr2_mii_mr1_clk
12	pr2_pru0_gpi10	pr2_pru0_gpi11	pr2_pru0_gpi6
13	pr2_pru0_gpo10	pr2_pru0_gpo11	pr2_pru0_gpo6
14	gpio1_24	gpio1_25	gpio7_1
15	Driver off	Driver off	Driver off

Table 4.3: P8.06-P8.09

	P8.06	P8.07	P8.08	P8.09
GPIO	194	165	166	178
BALL	AC3	G14	F14	E17
REG	0x1790	0x16EC	0x16F0	0x1698
MODE0	mmc3_dat3	mcasp1_axr14	mcasp1_axr15	xref_clk1
1	spi3_cs1	mcasp7_aclkx	mcasp7_fsx	mcasp2_axr9
2	uart5_rtsn	mcasp7_aclkr	mcasp7_fsr	mcasp1_axr5
3				mcasp2_ahclkx
4	vin2b_d2			mcasp6_ahclkx
5				
6				
7		vin6a_d9	vin6a_d8	vin6a_clk0
8				
9	vin5a_d2			
10	eQEP3_strobe	timer11	timer12	timer14
11	pr2_mii1_rxdv	pr2_mii0_rxdv	pr2_mii0_rxd3	pr2_mii1_crs
12	pr2_pru0_gpi7	pr2_pru1_gpi16	pr2_pru0_gpi20	pr2_pru1_gpi6
13	pr2_pru0_gpo7	pr2_pru1_gpo16	pr2_pru0_gpo20	pr2_pru1_gpo6
14	gpio7_2	gpio6_5	gpio6_6	gpio6_18
15	Driver off	Driver off	Driver off	Driver off

Table 4.4: P8.10-P8.13

	P8.10	P8.11	P8.12	P8.13
GPIO	164	75	74	107
BALL	A13	AH4	AG6	D3
REG	0x16E8	0x1510	0x150C	0x1590
MODE 0	mcasp1_axr13	vin1a_d7	vin1a_d6	vin2a_d10
1	mcasp7_axr1			
2				
3		vout3_d0	vout3_d1	mdio_mclk
4		vout3_d16	vout3_d17	vout2_d13
5				
6				
7	vin6a_d10			
8				
9				kbd_col7
10	timer10	eQEP2B_in	eQEP2A_in	ehrpwm2B
11	pr2_mii_mr0_clk			pr1_mdio_mdclk
12	pr2_pru1_gpi15	pr1_pru0_gpi4	pr1_pru0_gpi3	pr1_pru1_gpi7
13	pr2_pru1_gpo15	pr1_pru0_gpo4	pr1_pru0_gpo3	pr1_pru1_gpo7
14	gpio6_4	gpio3_11	gpio3_10	gpio4_11
15	Driver off	Driver off	Driver off	Driver off

Table 4.5: P8.14-P8.16

	P8.14	P8.15	P8.16
GPIO	109	99	125
BALL	D5	D1	B4
REG	0x1598	0x1570	0x15BC
MODE 0	vin2a_d12	vin2a_d2	vin2a_d21
1			
2			vin2b_d2
3	rgmii1_txc		rgmii1_rxd2
4	vout2_d11	vout2_d21	vout2_d2
5		emu12	vin3a_fld0
6			vin3a_d13
7			
8	mii1_rxclk	uart10_rxd	mii1_col
9	kbd_col8	kbd_row6	

continues on next page

Table 4.5 – continued from previous page

	P8.14	P8.15	P8.16
10	eCAP2_in_PWM2_out	eCAP1_in_PWM1_out	
11	pr1_mii1_txd1	pr1_ecap0_ecap_capin_apwm_o	pr1_mii1_rxlink
12	pr1_pru1_gpi9	pr1_edio_data_in7	pr1_pru1_gpi18
13	pr1_pru1_gpo9	pr1_edio_data_out7	pr1_pru1_gpo18
14	gpio4_13	gpio4_3	gpio4_29
15	Driver off	Driver off	Driver off
2nd BALL		A3	
2nd REG		0x15B4	
2nd MODE 0		vin2a_d19	
2nd 1			
2nd 2		vin2b_d4	
2nd 3		rgmii1_rxctl	
2nd 4		vout2_d4	
2nd 5			
2nd 6		vin3a_d11	
2nd 7			
2nd 8		mii1_txer	
2nd 9			
2nd 10		ehrpwm3_tripzone_input	
2nd 11		pr1_mii1_rxd0	
2nd 12		pr1_pru1_gpi16	
2nd 13		pr1_pru1_gpo16	
2nd 14		gpio4_27	
2nd 15		Driver off	

Table 4.6: P8.17-P8.19

	P8.17	P8.18	P8.19
GPIO	242	105	106
BALL	A7	F5	E6
REG	0x1624	0x1588	0x158C
MODE 0	vout1_d18	vin2a_d8	vin2a_d9
1			
2	emu4		
3	vin4a_d2		
4	vin3a_d2	vout2_d15	vout2_d14
5	obs11	emu18	emu19
6	obs27		
7			
8		mii1_rxd3	mii1_rxd0
9		kbd_col5	kbd_col6
10	pr2_edio_data_in2	eQEP2_strobe	ehrpwm2A
11	pr2_edio_data_out2	pr1_mii1_txd3	pr1_mii1_txd2
12	pr2_pru0_gpi15	pr1_pru1_gpi5	pr1_pru1_gpi6
13	pr2_pru0_gpo15	pr1_pru1_gpo5	pr1_pru1_gpo6
14	gpio8_18	gpio4_9	gpio4_10
15	Driver off	Driver off	Driver off

Table 4.7: P8.20-P8.22

	P8.20	P8.21	P8.22
GPIO	190	189	23
BALL	AC4	AD4	AD6
REG	0x1780	0x177C	0x1798
MODE 0	mmc3_cmd	mmc3_clk	mmc3_dat5
1	spi3_sclk		spi4_d1
2			uart10_txd
3			
4	vin2b_d6	vin2b_d7	vin2b_d0
5			
6			
7			
8			
9	vin5a_d6	vin5a_d7	vin5a_d0
10	eCAP2_in_PWM2_out	ehrpwm2_tripzone_input	ehrpwm3B
11	pr2_mii1_txd2	pr2_mii1_txd3	pr2_mii1_rxd2
12	pr2_pru0_gpi3	pr2_pru0_gpi2	pr2_pru0_gpi9
13	pr2_pru0_gpo3	pr2_pru0_gpo2	pr2_pru0_gpo9
14	gpio6_30	gpio6_29	gpio1_23
15	Driver off	Driver off	Driver off

Table 4.8: P8.23-P8.26

	P8.23	P8.24	P8.25	P8.26
GPIO	22	192	191	124
BALL	AC8	AC6	AC7	B3
REG	0x1794	0x1788	0x1784	0x15B8
MODE 0	mmc3_dat4	mmc3_dat1	mmc3_dat0	vin2a_d20
1	spi4_sclk	spi3_d0	spi3_d1	
2	uart10_rxd	uart5_txd	uart5_rxd	vin2b_d3
3				rgmii1_rxd3
4	vin2b_d1	vin2b_d4	vin2b_d5	vout2_d3
5				vin3a_de0
6				vin3a_d12
7				
8				mii1_rxer
9	vin5a_d1	vin5a_d4	vin5a_d5	
10	ehrpwm3A	eQEP3B_in	eQEP3A_in	eCAP3_in_PWM3_out
11	pr2_mii1_rxd3	pr2_mii1_txd0	pr2_mii1_txd1	pr1_mii1_rxer
12	pr2_pru0_gpi8	pr2_pru0_gpi5	pr2_pru0_gpi4	pr1_pru1_gpi17
13	pr2_pru0_gpo8	pr2_pru0_gpo5	pr2_pru0_gpo4	pr1_pru1_gpo17
14	gpio1_22	gpio7_0	gpio6_31	gpio4_28
15	Driver off	Driver off	Driver off	Driver off

Table 4.9: P8.27-P8.29

	P8.27	P8.28	P8.29
GPIO	119	115	118
BALL	E11	D11	C11
REG	0x15D8	0x15C8	0x15D4
MODE 0	vout1_vsync	vout1_clk	vout1_hsync
1			
2			
3	vin4a_vsync0	vin4a_fld0	vin4a_hsync0
4	vin3a_vsync0	vin3a_fld0	vin3a_hsync0
5			
6			
7			
8	spi3_sclk	spi3_cs0	spi3_d0
9			

continues on next page

Table 4.9 – continued from previous page

	P8.27	P8.28	P8.29
10			
11			
12	pr2_pru1_gpi17		
13	pr2_pru1_gpo17		
14	gpio4_23	gpio4_19	gpio4_22
15	Driver off	Driver off	Driver off
2nd BALL	A8	C9	A9
2nd REG	0x1628	0x162C	0x1630
2nd MODE0	vout1_d19	vout1_d20	vout1_d21
2nd 1			
2nd 2	emu15	emu16	emu17
2nd 3	vin4a_d3	vin4a_d4	vin4a_d5
2nd 4	vin3a_d3	vin3a_d4	vin3a_d5
2nd 5	obs12	obs13	obs14
2nd 6	obs28	obs29	obs30
2nd 7			
2nd 8			
2nd 9			
2nd 10	pr2_edio_data_in3	pr2_edio_data_in4	pr2_edio_data_in5
2nd 11	pr2_edio_data_out3	pr2_edio_data_out4	pr2_edio_data_out5
2nd 12	pr2_pru0_gpi16	pr2_pru0_gpi17	pr2_pru0_gpi18
2nd 13	pr2_pru0_gpo16	pr2_pru0_gpo17	pr2_pru0_gpo18
2nd 14	gpio8_19	gpio8_20	gpio8_21
2nd 15	Driver off	Driver off	Driver off

Table 4.10: P8.30-P8.32

	P8.30	P8.31	P8.32
GPIO	116	238	239
BALL	B10	C8	C7
REG	0x15CC	0x1614	0x1618
MODE 0	vout1_de	vout1_d14	vout1_d15
1			
2		emu13	emu14
3	vin4a_de0	vin4a_d14	vin4a_d15
4	vin3a_de0	vin3a_d14	vin3a_d15
5		obs9	obs10
6		obs25	obs26
7			
8	spi3_d1		
9			
10		pr2_uart0_txd	pr2_ecap0_ecap_capin_apwm_o
11			
12		pr2_pru0_gpi11	pr2_pru0_gpi12
13		pr2_pru0_gpo11	pr2_pru0_gpo12
14	gpio4_20	gpio8_14	gpio8_15
15	Driver off	Driver off	Driver off
2nd BALL	B9	G16	D17
2nd REG	0x1634	0x173C	0x1740
2nd MODE 0	vout1_d22	mcasp4_axr0	mcasp4_axr1
2nd 1			
2nd 2	emu18	spi3_d0	spi3_cs0
2nd 3	vin4a_d6	uart8_ctsn	uart8_rtsn

continues on next page

Table 4.10 – continued from previous page

	P8.30	P8.31	P8.32
2nd 4	vin3a_d6	uart4_rxd	uart4_txd
2nd 5	obs15		
2nd 6	obs31	vout2_d18	vout2_d19
2nd 7			
2nd 8		vin4a_d18	vin4a_d19
2nd 9		vin5a_d13	vin5a_d12
2nd 10	pr2_edio_data_in6		
2nd 11	pr2_edio_data_out6		
2nd 12	pr2_pru0_gpi19		pr2_pru1_gpi0
2nd 13	pr2_pru0_gpo19		pr2_pru1_gpo0
2nd 14	gpio8_22		
2nd 15	Driver off	Driver off	Driver off

Table 4.11: P8.33-P8.35

	P8.33	P8.34	P8.35
GPIO	237	235	236
BALL	C6	D8	A5
REG	0x1610	0x1608	0x160C
MODE 0	vout1_d13	vout1_d11	vout1_d12
1			
2	emu12	emu10	emu11
3	vin4a_d13	vin4a_d11	vin4a_d12
4	vin3a_d13	vin3a_d11	vin3a_d12
5	obs8	obs6	obs7
6	obs24	obs22	obs23
7		obs_dmarq2	
8			
9			
10	pr2_uart0_rxd	pr2_uart0_cts_n	pr2_uart0_rts_n
11			
12	pr2_pru0_gpi10	pr2_pru0_gpi8	pr2_pru0_gpi9
13	pr2_pru0_gpo10	pr2_pru0_gpo8	pr2_pru0_gpo9
14	gpio8_13	gpio8_11	gpio8_12
15	Driver off	Driver off	Driver off
2nd BALL	AF9	G6	AD9
2nd REG	0x14E8	0x1564	0x14E4
2nd MODE0	vin1a_fld0	vin2a_vsync0	vin1a_de0
2nd 1	vin1b_vsync1		vin1b_hsync1
2nd 2			
2nd 3		vin2b_vsync1	vout3_d17
2nd 4	vout3_clk	vout2_vsync	vout3_de
2nd 5	uart7_txd	emu9	uart7_rxd
2nd 6			
2nd 7	timer15	uart9_txd	timer16
2nd 8	spi3_d1	spi4_d1	spi3_sclk
2nd 9	kbd_row1	kbd_row3	kbd_row0
2nd 10	eQEP1B_in	ehrpwm1A	eQEP1A_in
2nd 11		pr1_uart0_rts_n	
2nd 12		pr1_edio_data_in4	
2nd 13		pr1_edio_data_out4	
2nd 14	gpio3_1	gpio4_0	gpio3_0
2nd 15	Driver off	Driver off	Driver off

Table 4.12: P8.36-P8.38

	P8.36	P8.37	P8.38
GPIO	234	232	233
BALL	D7	E8	D9
REG	0x1604	0x15FC	0x1600
MODE 0	vout1_d10	vout1_d8	vout1_d9
1			
2	emu3	uart6_rxd	uart6_txd
3	vin4a_d10	vin4a_d8	vin4a_d9
4	vin3a_d10	vin3a_d8	vin3a_d9
5	obs5		
6	obs21		
7	obs_irq2		
8			
9			
10	pr2_edio_sof	pr2_edc_sync1_out	pr2_edio_latch_in
11			
12	pr2_pru0_gpi7	pr2_pru0_gpi5	pr2_pru0_gpi6
13	pr2_pru0_gpo7	pr2_pru0_gpo5	pr2_pru0_gpo6
14	gpio8_10	gpio8_8	gpio8_9
15	Driver off	Driver off	Driver off
2nd BALL	F2	A21	C18
2nd REG	0x1568	0x1738	0x1734
2nd MODE 0	vin2a_d0	mcasp4_fsx	mcasp4_aclkx
2nd 1		mcasp4_fsr	mcasp4_aclkr
2nd 2		spi3_d1	spi3_sclk
2nd 3		uart8_txd	uart8_rxd
2nd 4	vout2_d23	i2c4_scl	i2c4_sda
2nd 5	emu10		
2nd 6		vout2_d17	vout2_d16
2nd 7	uart9_ctsn		
2nd 8	spi4_d0	vin4a_d17	vin4a_d16
2nd 9	kbd_row4	vin5a_d14	vin5a_d15
2nd 10	ehrpwm1B		
2nd 11	pr1_uart0_rxd		
2nd 12	pr1_edio_data_in5		
2nd 13	pr1_edio_data_out5		
2nd 14	gpio4_1		
2nd 15	Driver off	Driver off	Driver off

Table 4.13: P8.39-P8.41

	P8.39	P8.40	P8.41
GPIO	230	231	228
BALL	F8	E7	E9
REG	0x15F4	0x15F8	0x15EC
MODE 0	vout1_d6	vout1_d7	vout1_d4
1			
2	emu8	emu9	emu6
3	vin4a_d22	vin4a_d23	vin4a_d20
4	vin3a_d22	vin3a_d23	vin3a_d20
5	obs4		obs2
6	obs20		obs18
7			
8			
9			
10	pr2_edc_latch1_in	pr2_edc_sync0_out	pr1_ecap0_ecap_capin_apwm_o
11			
12	pr2_pru0_gpi3	pr2_pru0_gpi4	pr2_pru0_gpi1
13	pr2_pru0_gpo3	pr2_pru0_gpo4	pr2_pru0_gpo1
14	gpio8_6	gpio8_7	gpio8_4
15	Driver off	Driver off	Driver off

Table 4.14: P8.42-P8.44

	P8.42	P8.43	P8.44
GPIO	229	226	227
BALL	F9	F10	G11
REG	0x15F0	0x15E4	0x15E8
MODE 0	vout1_d5	vout1_d2	vout1_d3
1			
2	emu7	emu2	emu5
3	vin4a_d21	vin4a_d18	vin4a_d19
4	vin3a_d21	vin3a_d18	vin3a_d19
5	obs3	obs0	obs1
6	obs19	obs16	obs17
7		obs_irq1	obs_dmarq1
8			
9			
10	pr2_edc_latch0_in	pr1_uart0_rxd	pr1_uart0_txd
11			
12	pr2_pru0_gpi2	pr2_pru1_gpi20	pr2_pru0_gpi0
13	pr2_pru0_gpo2	pr2_pru1_gpo20	pr2_pru0_gpo0
14	gpio8_5	gpio8_2	gpio8_3
15	Driver off	Driver off	Driver off

Table 4.15: P8.45-P8.46

	P8.45	P8.46
GPIO	224	225
BALL	F11	G10
REG	0x15DC	0x15E0
MODE 0	vout1_d0	vout1_d1
1		
2	uart5_rxd	uart5_txd
3	vin4a_d16	vin4a_d17
4	vin3a_d16	vin3a_d17
5		
6		
7		
8	spi3_cs2	
9		

continues on next page

Table 4.15 – continued from previous page

	P8.45	P8.46
10	pr1_uart0_cts_n	pr1_uart0_rts_n
11		
12	pr2_pru1_gpi18	pr2_pru1_gpi19
13	pr2_pru1_gpo18	pr2_pru1_gpo19
14	gpio8_0	gpio8_1
15	Driver off	Driver off
2nd BALL	B7	A10
2nd REG	0x161C	0x1638
2nd MODE 0	vout1_d16	vout1_d23
2nd 1		
2nd 2	uart7_rxd	emu19
2nd 3	vin4a_d0	vin4a_d7
2nd 4	vin3a_d0	vin3a_d7
2nd 5		
2nd 6		
2nd 7		
2nd 8		spi3_cs3
2nd 9		
2nd 10	pr2_edio_data_in0	pr2_edio_data_in7
2nd 11	pr2_edio_data_out0	pr2_edio_data_out7
2nd 12	pr2_pru0_gpi13	pr2_pru0_gpi20
2nd 13	pr2_pru0_gpo13	pr2_pru0_gpo20
2nd 14	gpio8_16	gpio8_23
2nd 15	Driver off	Driver off

Todo: Notes regarding the resistors on muxed pins.

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.

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.

In the table are the following notations:

PWR_BUT is a 5V level as pulled up internally by the TPS6590379. It is activated by pulling the signal to GND.

Todo: (Actually, on BeagleBone AI, I believe PWR_BUT is pulled to 3.3V, but activation is still done by pulling the signal to GND. Also, a quick grounding of PWR_BUT will trigger a system event where shutdown can occur, but there is no hardware power-off function like on BeagleBone Black via this signal. It does, however, act as a hardware power-on.)

Todo: (On BeagleBone Black, SYS_RESET was a bi-directional signal, but it is only an output from BeagleBone AI to capes on BeagleBone AI.)

Table 4.16: P9.01-P9.05

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

Table 4.17: P9.06-P9.10

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

Table 4.18: P9.11-P9.13

	P9.11	P9.12	P9.13
GPIO	241	128	172
BALL	B19	B14	C17
REG	0x172C	0x16AC	0x1730
MODE 0	mcasp3_axr0	mcasp1_aclkr	mcasp3_axr1
1		mcasp7_axr2	
2	mcasp2_axr14		mcasp2_axr15
3	uart7_ctsn		uart7_rtsn
4	uart5_rxd		uart5_txd
5			
6		vout2_d0	
7	vin6a_d1		vin6a_d0
8		vin4a_d0	
9			vin5a_fld0
10		i2c4_sda	
11	pr2_mii1_rxer		pr2_mii1_rxlink
12	pr2_pru0_gpi14		pr2_pru0_gpi15
13	pr2_pru0_gpo14		pr2_pru0_gpo15
14		gpio5_0	
15	Driver off	Driver off	Driver off
2nd BALL	B8		AB10**
2nd REG	0x1620		0x1680
2nd MODE 0	vout1_d17		usb1_drvvbus
2nd 1			
2nd 2	uart7_txd		
2nd 3	vin4a_d1		
2nd 4	vin3a_d1		
2nd 5			
2nd 6			
2nd 7			timer16

continues on next page

Table 4.18 – continued from previous page

	P9.11	P9.12	P9.13
2nd 8			
2nd 9			
2nd 10	pr2_edio_data_in1		
2nd 11	pr2_edio_data_out1		
2nd 12	pr2_pru0_gpi14		
2nd 13	pr2_pru0_gpo14		
2nd 14	gpio8_17		gpio6_12
2nd 15	Driver off		Driver off

Table 4.19: P9.14-P9.16

	P9.14	P9.15	P9.16
GPIO	121	76	122
BALL	D6	AG4	C5
REG	0x15AC	0x1514	0x15B0
MODE 0	vin2a_d17	vin1a_d8	vin2a_d18
1		vin1b_d7	
2	vin2b_d6		vin2b_d5
3	rgmii1_txd0		rgmii1_rxc
4	vout2_d6	vout3_d15	vout2_d5
5			
6	vin3a_d9		vin3a_d10
7			
8	mii1_txd2		mii1_txd3
9		kbd_row2	
10	ehrpwm3A	eQEP2_index	ehrpwm3B
11	pr1_mii1_rxd2		pr1_mii1_rxd1
12	pr1_pru1_gpi14	pr1_pru0_gpi5	pr1_pru1_gpi15
13	pr1_pru1_gpo14	pr1_pru0_gpo5	pr1_pru1_gpo15
14	gpio4_25	gpio3_12	gpio4_26
15	Driver off	Driver off	Driver off

Table 4.20: P9.17-P9.19

	P9.17	P9.18	P9.19
GPIO	209	208	195
BALL	B24	G17	R6
REG	0x17CC	0x17C8	0x1440
MODE 0	spi2_cs0	spi2_d0	gpmc_a0
1	uart3_rtsn	uart3_ctsn	
2	uart5_txd	uart5_rxd	vin3a_d16
3			vout3_d16
4			vin4a_d0
5			
6			vin4b_d0
7			i2c4_scl
8			uart5_rxd
9			
10			
11			
12			
13			
14	gpio7_17	gpio7_16	gpio7_3
15	Driver off	Driver off	Driver off
2nd BALL	F12	G12	F4
2nd REG	0x16B8	0x16B4	0x157C

continues on next page

Table 4.20 – continued from previous page

	P9.17	P9.18	P9.19
2nd MODE 0	mcasp1_axr1	mcasp1_axr0	vin2a_d5
2nd 1			
2nd 2			
2nd 3	uart6_txd	uart6_rxd	
2nd 4			vout2_d18
2nd 5			emu15
2nd 6			
2nd 7	vin6a_hsync0	vin6a_vsync0	
2nd 8			uart10_rtsn
2nd 9			kbd_col2
2nd 10	i2c5_scl	i2c5_sda	eQEP2A_in
2nd 11	pr2_mii_mt0_clk	pr2_mii0_rxer	pr1_edio_sof
2nd 12	pr2_pru1_gpi9	pr2_pru1_gpi8	pr1_pru1_gpi2
2nd 13	pr2_pru1_gpo9	pr2_pru1_gpo8	pr1_pru1_gpo2
2nd 14	gpio5_3	gpio5_2	gpio4_6
2nd 15	Driver off	Driver off	Driver off

Table 4.21: P9.20-P9.22

	P9.20	P9.21	P9.22
GPIO	196	67	179
BALL	T9	AF8	B26
REG	0x1444	0x14F0	0x169C
MODE 0	gpmc_a1	vin1a_vsync0	xref_clk2
1		vin1b_de1	mcasp2_axr10
2	vin3a_d17		mcasp1_axr6
3	vout3_d17		mcasp3_ahclkx
4	vin4a_d1	vout3_vsync	mcasp7_ahclkx
5		uart7_rtsn	
6	vin4b_d1		vout2_clk
7	i2c4_sda	timer13	
8	uart5_txd	spi3_cs0	vin4a_clk0
9			
10		eQEP1_strobe	timer15
11			
12			
13			
14	gpio7_4	gpio3_3	gpio6_19
15	Driver off	Driver off	Driver off
2nd BALL	D2	B22	A26
2nd REG	0x1578	0x17C4	0x17C0
2nd MODE 0	vin2a_d4	spi2_d1	spi2_sclk
2nd 1		uart3_txd	uart3_rxd
2nd 2			
2nd 3			
2nd 4	vout2_d19		
2nd 5	emu14		
2nd 6			
2nd 7			
2nd 8	uart10_ctsn		
2nd 9	kbd_col1		
2nd 10	ehrpwm1_synco		
2nd 11	pr1_edc_sync0_out		

continues on next page

Table 4.21 – continued from previous page

	P9.20	P9.21	P9.22
2nd 12	pr1_pru1_gpi1		
2nd 13	pr1_pru1_gpo1		
2nd 14	gpio4_5	gpio7_15	gpio7_14
2nd 15	Driver off	Driver off	Driver off

Table 4.22: P9.23-P9.25

	P9.23	P9.24	P9.25
GPIO	203	175	177
BALL	A22	F20	D18
REG	0x17B4	0x168C	0x1694
MODE 0	spi1_cs1	gpio6_15	xref_clk0
1		mcasp1_axr9	mcasp2_axr8
2	sata1_led	dcan2_rx	mcasp1_axr4
3	spi2_cs1	uart10_txd	mcasp1_ahclkx
4			mcasp5_ahclkx
5			
6		vout2_vsync	
7			vin6a_d0
8		vin4a_vsync0	hdq0
9		i2c3_scl	clkout2
10		timer2	timer13
11			pr2_mii1_col
12			pr2_pru1_gpi5
13			pr2_pru1_gpo5
14	gpio7_11	gpio6_15	gpio6_17
15	Driver off	Driver off	Driver off

Table 4.23: P9.26-P9.29

	P9.26	P9.27	P9.28	P9.29
GPIO	174	111	113	139
BALL	E21	C3	A12	A11
REG	0x1688	0x15A0	0x16E0	0x16D8
MODE 0	gpio6_14	vin2a_d14	mcasp1_axr11	mcasp1_axr9
1	mcasp1_axr8		mcasp6_fsx	mcasp6_axr1
2	dcan2_tx		mcasp6_fsr	
3	uart10_rxd	rgmii1_txd3	spi3_cs0	spi3_d1
4		vout2_d9		
5				
6	vout2_hsync			
7			vin6a_d12	vin6a_d14
8	vin4a_hsync0	mii1_txclk		
9	i2c3_sda			
10	timer1	eQEP3B_in	timer8	timer6
11		pr1_mii_mr1_clk	pr2_mii0_txd1	pr2_mii0_txd3
12		pr1_pru1_gpi11	pr2_pru1_gpi13	pr2_pru1_gpi11
13		pr1_pru1_gpo11	pr2_pru1_gpo13	pr2_pru1_gpo11
14	gpio6_14	gpio4_15	gpio4_17	gpio5_11
15	Driver off	Driver off	Driver off	Driver off
2nd BALL	AE2	J14		D14
2nd REG	0x1544	0x16B0		0x16A8
2nd MODE 0	vin1a_d20	mcasp1_fsr		mcasp1_fsx
2nd 1	vin1b_d3	mcasp7_axr3		
2nd 2				
2nd 3				

continues on next page

Table 4.23 – continued from previous page

	P9.26	P9.27	P9.28	P9.29
2nd 4	vout3_d3			
2nd 5				
2nd 6	vin3a_d4	vout2_d1		
2nd 7				vin6a_de0
2nd 8		vin4a_d1		
2nd 9	kbd_col5			
2nd 10	pr1_edio_data_in4	i2c4_scl		i2c3_scl
2nd 11	pr1_edio_data_out4			pr2_mdio_data
2nd 12	pr1_pru0_gpi17			
2nd 13	pr1_pru0_gpo17			
2nd 14	gpio3_24	gpio5_1		gpio7_30
2nd 15	Driver off	Driver off		Driveroff

Table 4.24: P9.30-P9.31

	P9.30	P9.31
GPIO	140	138
BALL	B13	B12
REG	0x16DC	0x16D4
MODE 0	mcasp1_axr10	mcasp1_axr8
1	mcasp6_aclkx	mcasp6_axr0
2	mcasp6_aclkr	
3	spi3_d0	spi3_sclk
4		
5		
6		
7	vin6a_d13	vin6a_d15
8		
9		
10	timer7	timer5
11	pr2_mii0_txd2	pr2_mii0_txen
12	pr2_pru1_gpi12	pr2_pru1_gpi10
13	pr2_pru1_gpo12	pr2_pru1_gpo10
14	gpio5_12	gpio5_10
15	Driver off	Driver off
2nd BALL		C14
2nd REG		0x16A4
2nd MODE 0		mcasp1_aclkx
2nd 1		
2nd 2		
2nd 3		
2nd 4		
2nd 5		
2nd 6		
2nd 7		vin6a_fld0
2nd 8		
2nd 9		
2nd 10		i2c3_sda
2nd 11		pr2_mdio_mdclk
2nd 12		pr2_pru1_gpi7
2nd 13		pr2_pru1_gpo7
2nd 14		gpio7_31
2nd 15		Driver off

Todo: This table needs entries

Table 4.25: P9.32-P9.40

	P9.32	P9.33	P9.34	P9.35	P9.36	P9.37	P9.38	P9.39	P9.40
Row 1	P9.32	P9.33	P9.34	P9.35	P9.36	P9.37	P9.38	P9.39	P9.40

Table 4.26: P9.41-P9.42

	P9.41	P9.42
GPIO	180	114
BALL	C23	E14
REG	0x16A0	0x16E4
MODE 0	xref_clk3	mcasp1_axr12
1	mcasp2_axr11	mcasp7_axr0
2	mcasp1_axr7	
3	mcasp4_ahclkx	spi3_cs1
4	mcasp8_ahclkx	
5		
6	vout2_de	
7	hdq0	vin6a_d11
8	vin4a_de0	
9	clkout3	
10	timer16	timer9
11		pr2_mii0_txd0
12		pr2_pru1_gpi14
13		pr2_pru1_gpo14
14	gpio6_20	gpio4_18
15	Driver off	Driver off
2nd BALL	C1	C2
2nd REG	0x1580	0x159C
2nd MODE 0	vin2a_d6	vin2a_d13
2nd 1		
2nd 2		
2nd 3		rgmii1_txctl
2nd 4	vout2_d17	vout2_d10
2nd 5	emu16	
2nd 6		
2nd 7		
2nd 8	mii1_rxd1	mii1_rxdv
2nd 9	kbd_col3	kbd_row8
2nd 10	eQEP2B_in	eQEP3A_in
2nd 11	pr1_mii_mt1_clk	pr1_mii1_txd0
2nd 12	pr1_pru1_gpi3	pr1_pru1_gpi10
2nd 13	pr1_pru1_gpo3	pr1_pru1_gpo10
2nd 14	gpio4_7	gpio4_14
2nd 15	Driver off	Driver off

Todo: Table entries needed

Table 4.27: P9.43-P9.46

	P9.43	P9.44	P9.45	P9.46
Row 1	P9.43	P9.44	P9.45	P9.46

4.2 Serial Debug

Todo: Need info on BealgeBone AI serial debug

4.3 USB 3 Type-C

Todo: Need info on BealgeBone AI USB Type-C connection

4.4 USB 2 Type-A

Todo: Need info on BealgeBone AI USB Type-A connection

4.5 Gigabit Ethernet

Todo: Need info on BealgeBone AI USB Gigabit Ethernet connection

4.6 Coaxial

Todo: Need info on BealgeBone AI u.FL antenna connection

4.7 microSD Memory

Todo: Need info on BealgeBone AI uSD card slot

4.8 microHDMI

Todo: Need info on BealgeBone AI uHDMI connection

Chapter 5

Cape Board Support

There is a Cape Headers Google Spreadsheet which has a lot of detail regarding various boards and cape add-on boards.

See also beaglebone-cape-interface-spec

Todo: Add BeagleBone-AI content

5.1 BeagleBone® Black Cape Compatibility

Todo: Add BeagleBone-AI `BeagleBone® Black Cape Compatibility` section content

See beaglebone-cape-interface-spec for now.

5.2 EEPROM

Todo: Add BeagleBone-AI `EEPROM` section content

5.3 Pin Usage Consideration

Todo: Add BeagleBone-AI `Pin Usage Consideration` section content

5.4 GPIO

Todo: Add BeagleBone-AI `GPIO` section content

5.5 I2C

Todo: Add BeagleBone-AI I2C section content

5.6 UART or PRU UART

This section is about both UART pins on the header and PRU UART pins on the headers we will include a chart and later some code

Table 5.1: UART

Function	Pin	ABC Ball	Pinctrl Register	Mode
uart3_txd	P9.21	B22	0x17C4	1
uart3_rxd	P9.22	A26	0x17C0	1
uart5_txd	P9.13	C17	0x1730	4
uart5_rxd	P9.11	B19	0x172C	4
uart5_ctsn	P8.05	AC9	0x178C	2
uart5_rtsn	P8.06	AC3	0x1790	2
uart8_txd	P8.37	A21	0x1738	3
uart8_rxd	P8.38	C18	0x1734	3
uart8_ctsn	P8.31	G16	0x173C	3
uart8_rtsn	P8.32	D17	0x1740	3
uart10_txd	P9.24	F20	0x168C	3
uart10_rxd	P9.26	E21	0x1688	3
uart10_ctsn	P8.03	AB8	0x179C	2
uart10_rtsn	P8.04	AB5	0x17A0	2
uart10_txd	P9.24	F20	0x168C	3
uart10_rxd	P9.26	E21	0x1688	3
uart10_ctsn	P9.20	D2	0x1578	8
uart10_rtsn	P9.19	F4	0x157C	8

Table 5.2: PRU UART

Function	Pin	ABC Ball	Pinctrl Register	Mode
pr2_uart0_txd	P8.31	C8	0x1614	10
pr2_uart0_rxd	P8.33	C6	0x1610	10
pr2_uart0_cts_n	P8.34	D8	0x1608	10
pr2_uart0_rts_n	P8.35	A5	0x160C	10
pr1_uart0_rxd	P8.43	F10	0x15E4	10
pr1_uart0_txd	P8.44	G11	0x15E8	10
pr1_uart0_cts_n	P8.45	F11	0x15DC	10
pr1_uart0_rts_n	P8.46	G10	0x15E0	10

Todo: Add BeagleBone-AI content

5.7 SPI

Todo: Add BeagleBone-AI SPI section content

5.8 Analog

Todo: Add BeagleBone-AI Analog section content

5.9 PWM, TIMER, eCAP or PRU PWM/eCAP

Todo: Add BeagleBone-AI PWM, TIMER, eCAP or PRU PWM/eCAP section content

5.10 eQEP

Todo: Add BeagleBone-AI eQEP section content

5.11 CAN

Todo: Add BeagleBone-AI CAN section content

5.12 McASP (audio serial like I2S and AC97)

Todo: Add BeagleBone-AI McASP (audio serial like I2S and AC97) section content

5.13 MMC

Todo: Add BeagleBone-AI MMC section content

5.14 LCD

Todo: Add BeagleBone-AI LCD section content

5.15 PRU GPIO

Todo: Add BeagleBone-AI `PRU GPIO` section content

5.16 CLKOUT

Todo: Add BeagleBone-AI `CLKOUT` section content

5.17 Expansion Connector Headers

Todo: discuss header options for working with the expansion connectors per https://git.beagleboard.org/beagleboard/beaglebone-black/-/wikis/System-Reference-Manual#section-7-1

5.18 Signal Usage

Todo: Add BeagleBone-AI `Signal Usage` section content

5.19 Cape Power

Todo: Add BeagleBone-AI `Cape Power` section content

5.20 Mechanical

Todo: Add BeagleBone-AI `Mechanical` section content

Chapter 6

Demos & Tutorials

6.1 Upgrade BeagleBone AI software

Important: This documentation is very old and your feedback is requested, ideally via the discussion mailing-list. Otherwise, visit /about/jkridner to contact me directly to provide feedback, because it is important this page be reasonably easy to use.

Note: By default, the boards may run warm and shutdown. Please keep significant ventilation on your board and update your software per these instructions.

There are 4 main steps to updating the software on BeagleBone AI:

1. Get connected to the Internet
2. Update the boot-up scripts and Linux kernel
3. Update distribution components
4. Update examples in the Cloud9 IDE workspace

The distribution components includes all of the on-board software utilizing Debian package management. This is the vast majority of the on-board software.

The examples in the Cloud9 IDE workspace are managed with a version control tool called git. This is so that it is possible to edit, record changes and revert changes to any of the examples. The Cloud9 IDE is integrated with this version control and history can be seen using the “Changes” tab on the far-left.

The boot-up scripts and Linux kernel updates are managed separately from rest of the system to simplified maintanence. Get connected to the Internet

There are many ways to get BeagleBone AI onto the Internet. Ethernet, WiFi and USB-based methods are described below. Getting an Internet connection and performing the distribution, examples and kernel updates below is the fastest way to get BeagleBone AI up-to-date. Ethernet

Just connect BeagleBone AI to your router and it will automatically DHCP an IP address for access to the Internet.

```
debian@beaglebone:/var/lib/cloud9$ ifconfig eth0
eth0: flags=-28605  mtu 1500
        inet 192.168.0.174  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::8a3f:4aff:fe82:c102  prefixlen 64  scopeid 0x20
        ether 88:3f:4a:82:c1:02  txqueuelen 1000  (Ethernet)
        RX packets 17763  bytes 12611619 (12.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7744  bytes 1010400 (986.7 KiB)
```

(continues on next page)

(continued from previous page)

```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 126

debian@beaglebone:/var/lib/cloud9$
```

6.1.1 WiFi (without an Enterprise Login)

Utilize the connman command-line app or the cmst graphical utility to connect to your WiFi network.

```
debian@beaglebone:/var/lib/cloud9$ sudo connmanctl
[sudo] password for debian:temppwd
connmanctl> scan wifi
Scan completed for wifi
connmanctl> services
       MyWifi                 wifi_1234567890_1234567890123456_managed_psk
connmanctl> agent on
Agent registered
connmanctl> connect wifi_1234567890_1234567890123456_managed_psk
Agent RequestInput wifi_1234567890_1234567890123456_managed_psk
       Passphrase = [ Type=psk, Requirement=mandatory, Alternates=[ WPS ] ]
       WPS = [ Type=wpspin, Requirement=alternate ]
Passphrase? MySecretPassphrase
Connected wifi_1234567890_1234567890123456_managed_psk
connmanctl> quit
debian@beaglebone:/var/lib/cloud9$ ifconfig wlan0
wlan0: flags=-28605  mtu 1500
        inet 192.168.0.193  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::82c5:f2ff:fe7f:722d  prefixlen 64  scopeid 0x20
        ether 80:c5:f2:7f:72:2d  txqueuelen 1000  (Ethernet)
        RX packets 24  bytes 2734 (2.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 56  bytes 10405 (10.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

debian@beaglebone:/var/lib/cloud9$
```

6.1.2 WiFi with Enterprise Login

For networks utilizing 802.1x Enterprise Login requirements, such as Eduroam, the creation of an additional configuration file can enable access.

Content taken from librobotcontrol documentation

Many wifi networks such as those found at universities and enterprises, require a user login instead of a shared passphrase. To demonstrate how to configure connman to connect to such networks, we will use the UCSD campus-wide network as an example.

Start with a normal scan and look for the desired enterprise network.

```
debian@beaglebone:/var/lib/cloud9$ sudo connmanctl
[sudo] password for debian:temppwd
connmanctl> scan wifi
Scan completed for wifi
connmanctl> services
        UCSD-PROTECTED       wifi_000f540aa884_554353442d50524f544543544544-
↪ieee8021x
        ATT5363              wifi_ec1127bffa51_41545435333633_managed_psk
        2WIRE407             wifi_ec1127bffa51_3257495245343037_managed_psk
        ATT8fHHhfi           wifi_ec1127bffa51_4154543866484868669_managed_
```

(continues on next page)

(continued from previous page)

```
↪psk
connmanctl> quit
```

Note how the type of network is listed as ieee8021x indicating that it uses Network Access Control instead of a typical passkey (psk) as you would find in a consumer home network.

Make a new file in the /var/lib/connman/ directory with a name matching what is listed during the scan. For this example, the name would be 000f540aa884_554353442d50524f544543544544-ieee8021x.config

Fill in this file as follows, replacing the service name, SSID, Identity, and Passphrase with your own details. Your enterprise network may also use an authentication method other than PEAP and MSCHAPV2. Consult the IT help desk for your enterprise for details on that configuration.

```
debian@beaglebone:/var/lib/cloud9$ sudo nano /var/lib/connman/wifi_
↪000f540aa884_554353442d50524f544543544544-ieee8021x.config
[sudo] password for debian:temppwd

Enter your information into the new config file like so:

[service_wifi_000f540aa884_554353442d50524f544543544544_managed_ieee8021x]
Type = wifi
SSID = 554353442d50524f544543544544
EAP = peap
Phase2 = MSCHAPV2
Identity= USERNAME
Passphrase= PASSWORD

Restart the connman service and check if the connection was successful

debian@beaglebone:/var/lib/cloud9$ sudo systemctl restart connman
debian@beaglebone:/var/lib/cloud9$ ifconfig wlan0
wlan0: flags=-28605  mtu 1500
        inet 192.168.0.193  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::82c5:f2ff:fe7f:722d  prefixlen 64  scopeid 0x20
        ether 80:c5:f2:7f:72:2d  txqueuelen 1000  (Ethernet)
        RX packets 24  bytes 2734 (2.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 56  bytes 10405 (10.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

debian@beaglebone:/var/lib/cloud9$
```

6.1.3 USB via Internet Connection Sharing

You need to first establish a shell connection different than the USB network connection you plan on using to get to the Internet.

In your host operating system, you'll need to share your Internet connection back to the board. With an Ubuntu host, use the utility "nm-connection-editor".

```
sudo ip addr flush dev usb0
```

```
sudo dhclient usb0
```

Notes: How to find MAC address and correct connection?

Notes: On Ubuntu, the IPv4 Settings terminology "Shared to other computers" is what you apply to the connection to your board (ie., downlink) not to your Internet-connected WiFi or Ethernet (ie., uplink). Update the boot-up scripts and Linux kernel

```
debian@beaglebone:/var/lib/cloud9$ cd /opt/scripts
debian@beaglebone:/opt/scripts$ git pull
Already up-to-date.
debian@beaglebone:/opt/scripts$ sudo tools/update_kernel.sh
[sudo] password for debian:temppwd
info: checking archive
2019-09-06 02:29:22 URL:https://rcn-ee.com/repos/latest/stretch-armhf/LATEST-
 ↪ti [168/168] -> "LATEST-ti" [1]
-----------------------------
Kernel Options:
ABI:1 LTS41 4.1.30-ti-r70
ABI:1 LTS44 4.4.155-ti-r155
ABI:1 LTS49 4.9.147-ti-r121
ABI:1 LTS414 4.14.108-ti-r116
ABI:1 LTS419 4.19.59-ti-r26
-----------------------------
Kernel version options:
-----------------------------
LTS44: --lts-4_4
LTS49: --lts-4_9
LTS414: --lts-4_14
LTS419: --lts-4_19
STABLE: --stable
TESTING: --testing
-----------------------------
info: you are running: [4.14.108-ti-r113], latest is: [4.14.108-ti-r116]␣
 ↪updating...
Ign:1 http://deb.debian.org/debian stretch InRelease
Get:2 http://deb.debian.org/debian stretch-updates InRelease [91.0 kB]
.
.
.
(Reading database ... 109903 files and directories currently installed.)
Preparing to unpack .../ti-sgx-jacinto6evm-modules-4.14.108-ti-r116_1stretch_
 ↪armhf.deb ...
Unpacking ti-sgx-jacinto6evm-modules-4.14.108-ti-r116 (1stretch) ...
Setting up ti-sgx-jacinto6evm-modules-4.14.108-ti-r116 (1stretch) ...
update-initramfs: Generating /boot/initrd.img-4.14.108-ti-r116
debian@beaglebone:/opt/scripts$ sudo shutdown -r now

Update distribution components

debian@beaglebone:/var/lib/cloud9$ sudo apt update
[sudo] password for debian:temppwd
Ign:1 http://deb.debian.org/debian stretch InRelease
Hit:2 http://deb.debian.org/debian stretch-updates InRelease
Hit:3 http://deb.debian.org/debian-security stretch/updates InRelease
.
.
.
debian@beaglebone:/var/lib/cloud9$ sudo apt upgrade
.
.
.
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libpq5␣
 ↪linux-cpupower linux-libc-dev nginx nginx-common nginx-full tzdata
23 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
Need to get 10.3 MB of archives.
After this operation, 41.0 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://deb.debian.org/debian stretch-updates/main armhf tzdata all␣
 ↪2019b-0+deb9u1 [275 kB]
```

(continues on next page)

(continued from previous page)

```
Get:2 http://repos.rcn-ee.com/debian stretch/main armhf bonescript armhf 0.7.
 ↪3-git20190822.0-0rcnee1~stretch+20190903 [5,463 kB]
Get:3 http://deb.debian.org/debian-security stretch/updates/main armhf␣
 ↪libcpupower1 armhf 4.9.168-1+deb9u5 [637 kB]
.
.
.
Setting up libiio-utils (0.16-1rcnee0~stretch+20190812) ...
Setting up libnginx-mod-http-echo (1.10.3-1+deb9u3) ...
Setting up linux-cpupower (4.9.168-1+deb9u5) ...
Setting up nginx-full (1.10.3-1+deb9u3) ...
[ ok ] Upgrading binary: nginx.
Setting up nginx (1.10.3-1+deb9u3) ...
Processing triggers for initramfs-tools (0.130) ...
update-initramfs: Generating /boot/initrd.img-4.14.108-ti-r116
debian@beaglebone:/var/lib/cloud9$ sudo apt install -y ti-tidl mjpg-streamer-
 ↪opencv-python

Update examples in the Cloud9 IDE workspace

debian@beaglebone:/var/lib/cloud9$ cd /var/lib/cloud9
debian@beaglebone:/var/lib/cloud9$ git pull
Already up-to-date.
debian@beaglebone:/var/lib/cloud9$

Test installed versions

debian@beaglebone:/var/lib/cloud9$ sudo /opt/scripts/tools/version.sh
[sudo] password for debian:temppwd
git:/opt/scripts/:[5b2e16aa1e5c0f627f1d48a6dd1c13b446b9f53b]
model:[BeagleBoard.org_BeagleBone_AI]
dogtag:[BeagleBoard.org Debian Image 2019-08-02]
kernel:[4.14.108-ti-r116]
nodejs:[v6.17.0]
pkg check: to individually upgrade run: [sudo apt install --only-upgrade ]
pkg:[bb-cape-overlays]:[4.4.20190812.0-0rcnee0~stretch+20190812]
pkg:[bb-wl18xx-firmware]:[1.20190227.1-0rcnee0~stretch+20190227]
pkg:[kmod]:[23-2rcnee1~stretch+20171005]
pkg:[librobotcontrol]:[1.0.4-git20190227.1-0rcnee0~stretch+20190327]
pkg:[firmware-ti-connectivity]:[20180825+dfsg-1rcnee1~stretch+20181217]
groups:[debian : debian adm kmem dialout cdrom floppy audio dip video␣
 ↪plugdev users systemd-journal i2c bluetooth netdev gpio pwm eqep␣
 ↪remoteproc admin spi tisdk weston-launch xenomai cloud9ide]
cmdline:[console=ttyS0,115200n8 root=/dev/mmcblk1p1 ro rootfstype=ext4␣
 ↪rootwait coherent_pool=1M net.ifnames=0 rng_core.default_quality=100 quiet]
dmesg | grep remote
[    2.945344] remoteproc remoteproc0: 4b234000.pru is available
[    2.946253] remoteproc remoteproc1: 4b238000.pru is available
[    2.962679] remoteproc remoteproc2: 4b2b4000.pru is available
[    2.965359] remoteproc remoteproc3: 4b2b8000.pru is available
[    6.569222] remoteproc remoteproc4: 58820000.ipu is available
[    6.598088] remoteproc remoteproc5: 55020000.ipu is available
[    6.606271] remoteproc remoteproc6: 40800000.dsp is available
[    6.627725] remoteproc remoteproc7: 41000000.dsp is available
[    6.634220] remoteproc remoteproc4: powering up 58820000.ipu
[    6.634239] remoteproc remoteproc4: Booting fw image dra7-ipu1-fw.xem4,␣
 ↪size 6867360
[    6.662443] remoteproc remoteproc4: registered virtio0 (type 7)
[    6.662449] remoteproc remoteproc4: remote processor 58820000.ipu is now␣
 ↪up
[    6.676794] remoteproc remoteproc5: powering up 55020000.ipu
```

(continues on next page)

(continued from previous page)

```
[    6.676819] remoteproc remoteproc5: Booting fw image dra7-ipu2-fw.xem4,␣
↪size 3751356
[    6.842752] Modules linked in: omap_remoteproc virtio_rpmsg_bus rpmsg_
↪core usb_f_ecm usb_f_mass_storage iptable_nat nf_conntrack_ipv4 nf_defrag_
↪ipv4 nf_nat_ipv4 nf_nat nf_conntrack usb_f_rndis u_ether libcomposite␣
↪iptable_mangle iptable_filter cmemk(O) uio_pdrv_genirq uio spidev pruss_
↪soc_bus pru_rproc pruss pruss_intc ip_tables x_tables
[    6.843887] Modules linked in: omap_remoteproc virtio_rpmsg_bus rpmsg_
↪core usb_f_ecm usb_f_mass_storage iptable_nat nf_conntrack_ipv4 nf_defrag_
↪ipv4 nf_nat_ipv4 nf_nat nf_conntrack usb_f_rndis u_ether libcomposite␣
↪iptable_mangle iptable_filter cmemk(O) uio_pdrv_genirq uio spidev pruss_
↪soc_bus pru_rproc pruss pruss_intc ip_tables x_tables
[    6.849561] Modules linked in: omap_remoteproc virtio_rpmsg_bus rpmsg_
↪core usb_f_ecm usb_f_mass_storage iptable_nat nf_conntrack_ipv4 nf_defrag_
↪ipv4 nf_nat_ipv4 nf_nat nf_conntrack usb_f_rndis u_ether libcomposite␣
↪iptable_mangle iptable_filter cmemk(O) uio_pdrv_genirq uio spidev pruss_
↪soc_bus pru_rproc pruss pruss_intc ip_tables x_tables
[    6.919311] remoteproc remoteproc5: registered virtio1 (type 7)
[    6.919319] remoteproc remoteproc5: remote processor 55020000.ipu is now␣
↪up
[    6.926824] remoteproc remoteproc7: powering up 41000000.dsp
[    6.926842] remoteproc remoteproc7: Booting fw image dra7-dsp2-fw.xe66,␣
↪size 20998684
[    6.936607] remoteproc remoteproc6: powering up 40800000.dsp
[    6.936623] remoteproc remoteproc6: Booting fw image dra7-dsp1-fw.xe66,␣
↪size 20998684
[    7.001835] remoteproc remoteproc7: registered virtio2 (type 7)
[    7.001842] remoteproc remoteproc7: remote processor 41000000.dsp is now␣
↪up
[    7.011099] remoteproc remoteproc6: registered virtio3 (type 7)
[    7.011106] remoteproc remoteproc6: remote processor 40800000.dsp is now␣
↪up
dmesg | grep pru
[    2.941572] pruss 4b200000.pruss: creating PRU cores and other child␣
↪platform devices
[    2.945344] remoteproc remoteproc0: 4b234000.pru is available
[    2.945394] pru-rproc 4b234000.pru: PRU rproc node /ocp/pruss_soc_
↪bus@4b226004/pruss@0/pru@34000 probed successfully
[    2.946253] remoteproc remoteproc1: 4b238000.pru is available
[    2.946307] pru-rproc 4b238000.pru: PRU rproc node /ocp/pruss_soc_
↪bus@4b226004/pruss@0/pru@38000 probed successfully
[    2.947598] pruss 4b280000.pruss: creating PRU cores and other child␣
↪platform devices
[    2.962679] remoteproc remoteproc2: 4b2b4000.pru is available
[    2.962733] pru-rproc 4b2b4000.pru: PRU rproc node /ocp/pruss_soc_
↪bus@4b2a6004/pruss@0/pru@34000 probed successfully
[    2.965359] remoteproc remoteproc3: 4b2b8000.pru is available
[    2.965409] pru-rproc 4b2b8000.pru: PRU rproc node /ocp/pruss_soc_
↪bus@4b2a6004/pruss@0/pru@38000 probed successfully
[    6.842752] Modules linked in: omap_remoteproc virtio_rpmsg_bus rpmsg_
↪core usb_f_ecm usb_f_mass_storage iptable_nat nf_conntrack_ipv4 nf_defrag_
↪ipv4 nf_nat_ipv4 nf_nat nf_conntrack usb_f_rndis u_ether libcomposite␣
↪iptable_mangle iptable_filter cmemk(O) uio_pdrv_genirq uio spidev pruss_
↪soc_bus pru_rproc pruss pruss_intc ip_tables x_tables
[    6.843887] Modules linked in: omap_remoteproc virtio_rpmsg_bus rpmsg_
↪core usb_f_ecm usb_f_mass_storage iptable_nat nf_conntrack_ipv4 nf_defrag_
↪ipv4 nf_nat_ipv4 nf_nat nf_conntrack usb_f_rndis u_ether libcomposite␣
↪iptable_mangle iptable_filter cmemk(O) uio_pdrv_genirq uio spidev pruss_
↪soc_bus pru_rproc pruss pruss_intc ip_tables x_tables
[    6.849561] Modules linked in: omap_remoteproc virtio_rpmsg_bus rpmsg_
↪core usb_f_ecm usb_f_mass_storage iptable_nat nf_conntrack_ipv4 nf_defrag_
```

(continues on next page)

(continued from previous page)

```
↪ipv4 nf_nat_ipv4 nf_nat nf_conntrack usb_f_rndis u_ether libcomposite␣
↪iptable_mangle iptable_filter cmemk(O) uio_pdrv_genirq uio spidev pruss_
↪soc_bus pru_rproc pruss pruss_intc ip_tables x_tables
[    9.175815] pruss_uio_shmem 4b200000.pruss_shmem: Allocating gdev
[    9.175825] pruss_uio_shmem 4b200000.pruss_shmem: Allocating info
[    9.175832] pruss_uio_shmem 4b200000.pruss_shmem: Requesting resource
[    9.175853] pruss_uio_shmem 4b200000.pruss_shmem: Mapping resource
[    9.179197] pruss_uio_shmem 4b200000.pruss_shmem: Registering with uio␣
↪driver
[    9.179745] pruss_uio_shmem 4b200000.pruss_shmem: Saving platform data
[    9.179858] pruss_uio_shmem 4b280000.pruss_shmem: Allocating gdev
[    9.179864] pruss_uio_shmem 4b280000.pruss_shmem: Allocating info
[    9.179870] pruss_uio_shmem 4b280000.pruss_shmem: Requesting resource
[    9.179886] pruss_uio_shmem 4b280000.pruss_shmem: Mapping resource
[    9.179899] pruss_uio_shmem 4b280000.pruss_shmem: Registering with uio␣
↪driver
[    9.180137] pruss_uio_shmem 4b280000.pruss_shmem: Saving platform data
dmesg | grep pinctrl-single
[    0.914771] pinctrl-single 4a003400.pinmux: 282 pins at pa fc003400 size␣
↪1128
dmesg | grep gpio-of-helper
lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 046d:0825 Logitech, Inc. Webcam C270
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
END
debian@beaglebone:/var/lib/cloud9$
```

Chapter 7

Additional Support Information

All support for BeagleBone AI design is through BeagleBoard.org community at BeagleBoard.org forum.

Todo: Reference https://docs.beagleboard.org/latest/intro/support/index.html and https://beagleboard.org/resources

Related TI documentation: http://www.ti.com/tool/BEAGLE-3P-BBONE-AI

7.1 Production board boot media

Todo: Add production boot media link in `_static/epilog/production.image` and reference it here.

7.2 REGULATORY, COMPLIANCE, AND EXPORT INFORMATION

- Country of origin: PRC
- FCC: 2ATUT-BBONE-AI
- CE: TBD
- CNHTS: 8543909000
- USHTS: 8473301180
- MXHTS: 84733001
- TARIC: 8473302000
- ECCN: 5A992.C
- CCATS: Z1613110/G180570
- RoHS/REACH: TBD
- Volatility: TBD

7.3 Mechanical Information

- Board Dimensions: 8.9cm x 5.4cm x 1.5cm

- Board Net Weight 48g
- Packaging Dimensions: 13.8cm x 10cm x 4cm
- Gross Weight (including packaging): 110g
- 3D STEP model: https://git.beagleboard.org/beagleboard/beaglebone-ai/-/tree/master/Mechanical

7.4 Change History

7.4.1 Rev A0

Initial prototype revision. Not taken to production. eMMC flash image provided by Embest.

7.4.2 Rev A1

Second round prototype.

- Fixed size of mounting holes.
- Added LED for WiFi status.
- Added microHDMI.
- Changed eMMC voltage from 3.3V to 1.8V to support HS200.
- Changed eMMC from 4GB to 16GB.
- Changed serial debug header from 6-pin 100mil pitch to 3-pin 1.5mm pitch.
- Switched expansion header from UART4 to UART5. The UART4 pins were used for the microHDMI.

eMMC flash image provided by Embest.

7.4.3 Rev A1a

Alpha pilot-run units and initial production.

- Added pull-down resistor on serial debug header RX line.

Alpha pilot-run eMMC flash image: https://debian.beagleboard.org/images/bbai-pilot-20190408.img.xz

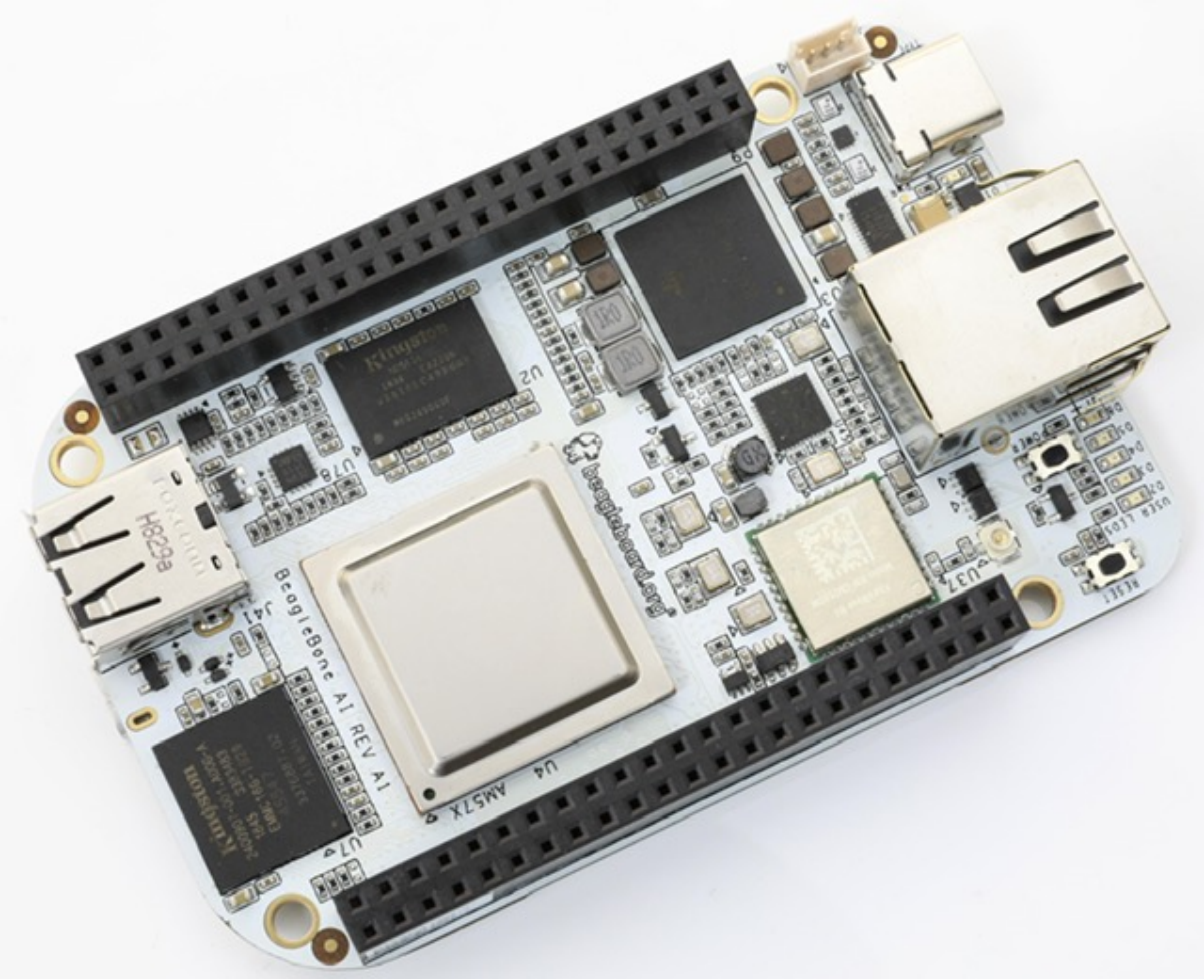
Production eMMC flash image: http://debian.beagleboard.org/images/am57xx-eMMC-flasher-debian-9.9-lxqt-armhf-2019-08-03-4gb.img.xz

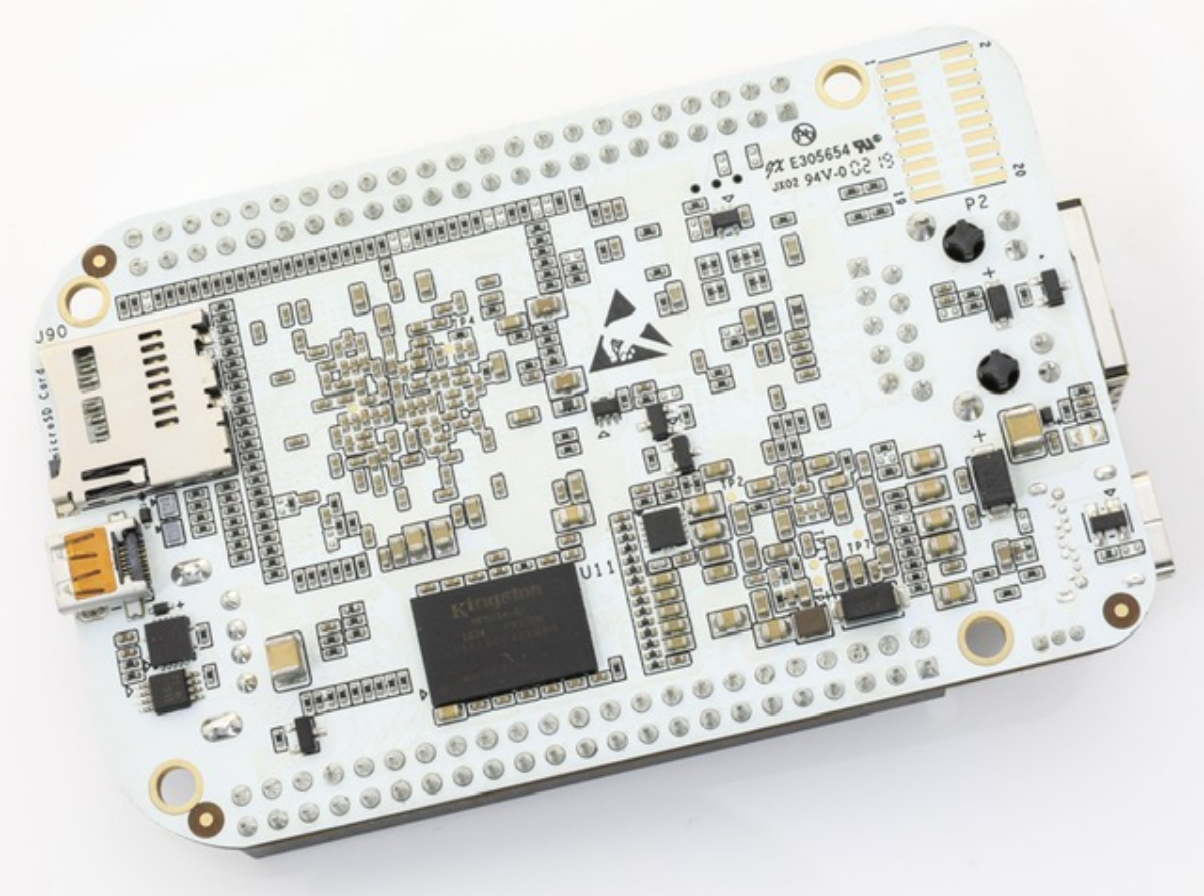
7.4.4 Rev A2

Proposed changes.

- HW: need pull-down on console uart RX line.
- HW: position of microSD may impact existing case designs.
- HW: P9.13 does not have a GPIO.
- HW: HDMI hotplug detection not working.
- HW: add extra DCAN.
- HW: wire mods required to enable JTAG.
- HW: Small I2C nvmem/eeprom for board identifier.

7.5 Pictures




J90
P2
U11
Kingston