



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

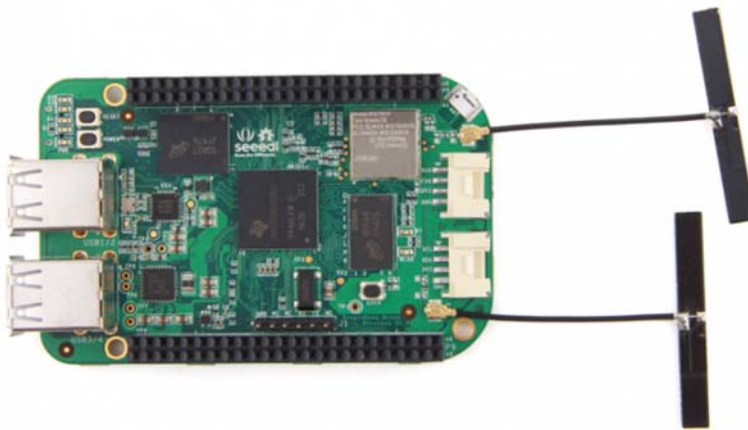
Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



BeagleBone Green Wireless

Introduction



SeedStudio BeagleBone Green Wireless (BBGW) is a joint effort by BeagleBoard.org and Seed Studio. It is based on the open-source hardware design of BeagleBone Black and developed into this differentiated version. SeedStudio BeagleBone Green Wireless has included a high-performance flexible WiFi/Bluetooth interface and two Grove connectors, making it easier to connect to the large family of Grove sensors. The on-board HDMI and Ethernet are removed to make room for these wireless features and Grove connectors.

Features

- **Fully Compatible with BeagleBone Black**
- **Processor: AM335x 1GHz ARM® Cortex-A8**
 - 512MB DDR3 RAM
 - 4GB 8-bit eMMC on-board flash storage
 - 3D graphics accelerator
 - NEON floating-point accelerator
 - 2x PRU 32-bit microcontrollers

- **Connectivity**

- USB client for power & communications
- USB host with 4-port hub
- WiFi 802.11 b/g/n 2.4GHz
- Bluetooth 4.1 with BLE
- 2x 46 pin headers
- 2x Grove connectors (I2C and UART)

- **Software Compatibility**

- Debian
- Android
- Ubuntu
- Cloud9 IDE on Node.js w/ BoneScript library
- plus much more

Specification

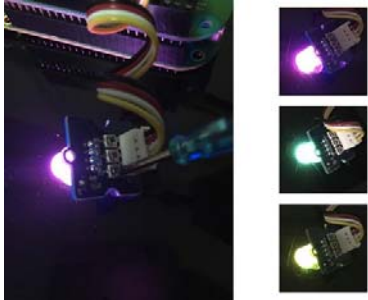
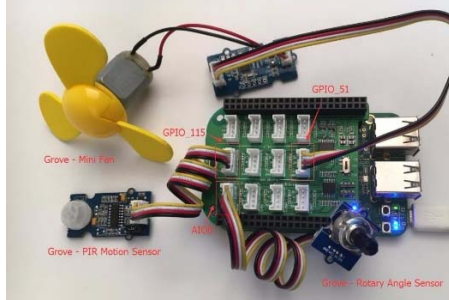
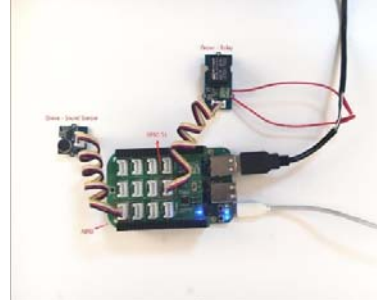


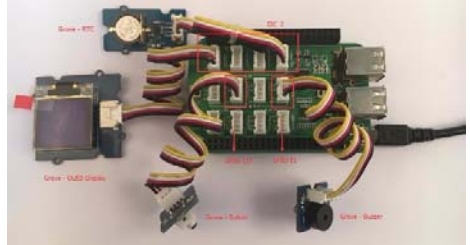
Item	Value
Processor	AM335x 1GHz ARM Cortex-A8
RAM	512MB DDR3
on-board Flash Storage	4GB eMMC
CPU Supports	NEON floating-point & 3D graphics accelerator
Micro USB Supports	powering & communications
USB	USB2.0 Host *4
Grove Connectors	2 (One I2C and One UART)
GPIO	2 x 46 pin headers
Ethernet	Wi-Fi 802.11b/g/n 2.4GHz and Bluetooth 4.1 LE
Operating Temperature	0 ~ 75

Application Ideas



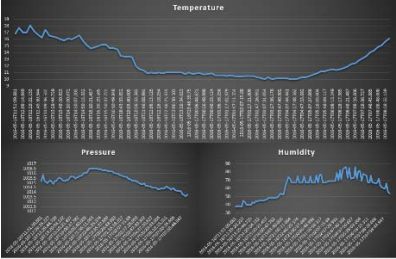
- Internet of Things
- Smart House
- Industrial
- Automation & Process Control
- Human Machine Interface
- Sensor Hub
- Robot

BBGW Starter Tutorial #1-#6

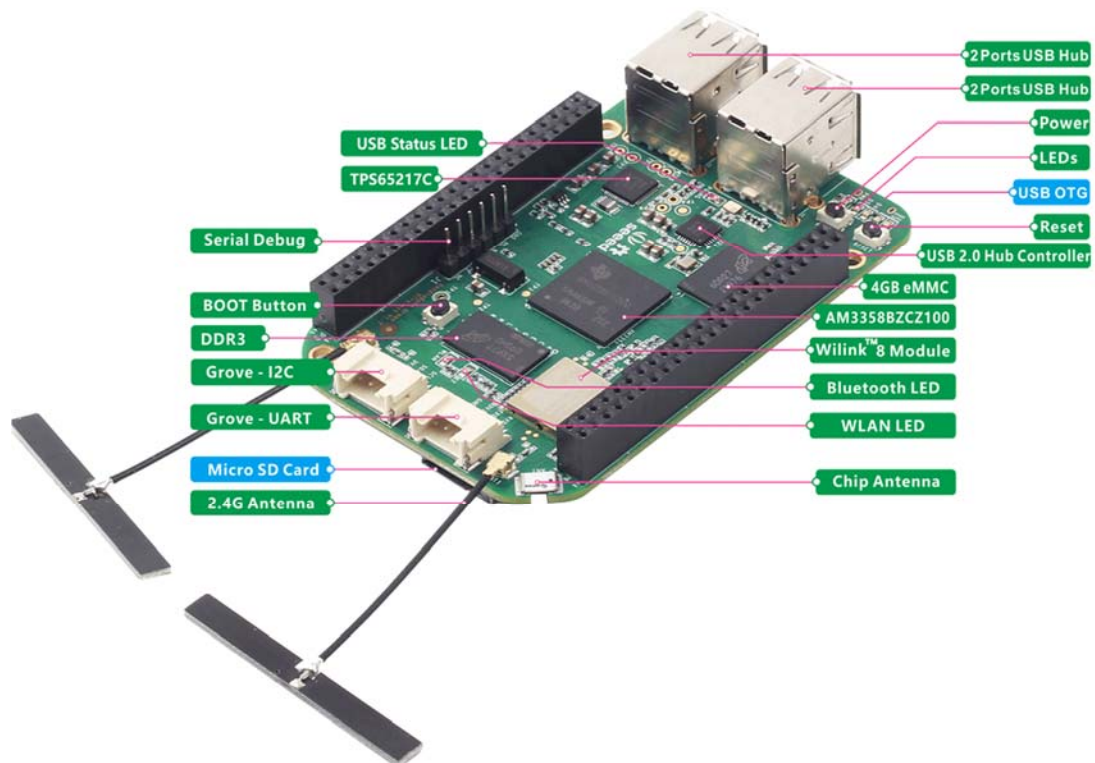
Here are some tutorials for the starters to use some Grove modules to realize their interesting ideas with BeagleBone Green Wireless(BBGW). The tutorials are based on Python and mraa/upm library.

<p>#1 The Breath LED</p>	<p>#2 Storm on your table</p>	<p>#3 Speak Louuuuudly</p>
		
<p>MAKE IT NOW!</p>	<p>MAKE IT NOW!</p>	<p>MAKE IT NOW!</p>
<p>#4 How hot is it today?</p>	<p>#5 Where are you?</p>	<p>#6 My Little alarm clock</p>
		
<p>MAKE IT NOW!</p>	<p>MAKE IT NOW!</p>	<p>MAKE IT NOW!</p>

Funny Projects

Bluetooth Device Detection	Home Control Center	SAP HCP IoT Service
 <p>Bluetooth Device Detection with the BeagleBone Green Wireless</p>		
<p>MAKE IT NOW!</p>	<p>MAKE IT NOW!</p>	<p>MAKE IT NOW!</p>

Hardware Overview



Pin map

Each digital I/O pin has 8 different modes that can be selected, including GPIO.

65 Possible Digital I/Os

Note

In GPIO mode, each digital I/O can produce interrupts.

Cape Expansion Headers

P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	MMC1_DAT6	3	4	MMC1_DAT7
VDD_5V	5	6	VDD_5V	MMC1_DAT2	5	6	MMC1_DAT3
SYS_5V	7	8	SYS_5V	GPIO_66	7	8	GPIO_67
PWR_BTN	9	10	SYS_RESETn	GPIO_69	9	10	GPIO_68
UART4_RXD	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
UART4_TXD	13	14	EHRPWM1A	EHRPWM2B	13	14	GPIO_26
GPIO_48	15	16	EHRPWM1B	GPIO_47	15	16	GPIO_46
SPI0_CS0	17	18	SPI0_D1	GPIO_27	17	18	GPIO_65
I2C2_SCL	19	20	I2C2_SDA	EHRPWM2A	19	20	MMC1_CMD
UART2_TXD	21	22	UART2_RXD	MMC1_CLK	21	22	MMC1_DAT5
GPIO_49	23	24	UART1_TXD	MMC1_DAT4	23	24	MMC1_DAT1
GPIO_117	25	26	UART1_RXD	MMC1_DAT0	25	26	GPIO_61
GPIO_115	27	28	GPIO_113	GPIO_86	27	28	GPIO_88
GPIO_111	29	30	GPIO_112	GPIO_87	29	30	GPIO_89
GPIO_110	31	32	VDD_ADC	GPIO_10	31	32	GPIO_11
AIN4	33	34	GNDA_ADC	GPIO_9	33	34	GPIO_81
AIN0	35	36	AIN5	GPIO_8	35	36	GPIO_80
AIN2	37	38	AIN3	GPIO_78	37	38	GPIO_79
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	ECAPPWM0	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	GPIO_70	45	46	GPIO_71

LEGEND	
 	Power/Ground/Reset
 	Available Digital
 	Available PWM
 	Shared I2C Bus
 	Reconfigurable Digital
 	Analog inputs (1.5V)

PWMs and Timers

Note

Up to 8 digital I/O pins can be configured with pulse-width modulators (PWM) to produce signals to control motors or create pseudo analog voltage levels, without taking up any extra CPU cycles.

8 PWMs and 4 timers

P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	GPIO_38	3	4	GPIO_39
VDD_5V	5	6	VDD_5V	GPIO_34	5	6	GPIO_35
SYS_5V	7	8	SYS_5V	TIMER4	7	8	TIMER7
PWR_BUT	9	10	SYS_RESETN	TIMER5	9	10	TIMER6
GPIO_30	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
GPIO_31	13	14	EHRPWM1A	EHRPWM2B	13	14	GPIO_26
GPIO_48	15	16	EHRPWM1B	GPIO_47	15	16	GPIO_46
GPIO_5	17	18	GPIO_4	GPIO_27	17	18	GPIO_65
I2C2_SCL	19	20	I2C2_SDA	EHRPWM2A	19	20	GPIO_63
EHRPWMOB	21	22	EHRPWMOA	GPIO_62	21	22	GPIO_37
GPIO_49	23	24	GPIO_15	GPIO_36	23	24	GPIO_33
GPIO_117	25	26	GPIO_14	GPIO_32	25	26	GPIO_61
GPIO_115	27	28	ECAPPWM2	GPIO_86	27	28	GPIO_88
EHRPWMOB	29	30	GPIO_112	GPIO_87	29	30	GPIO_89
EHRPWMOA	31	32	VDD_ADC	GPIO_10	31	32	GPIO_11
AIN4	33	34	GND_A_ADC	GPIO_9	33	34	EHRPWM1B
AIN6	35	36	AIN5	GPIO_8	35	36	EHRPWM1A
AIN2	37	38	AIN3	GPIO_78	37	38	GPIO_79
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	ECAPPWMO	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	EHRPWM2A	45	46	EHRPWM2B

Analog Inputs

Note

Make sure you don't input more than 1.8V to the analog input pins. This is a single 12-bit analog-to-digital converter with 8 channels, 7 of which are made available on the headers.

7 analog inputs (1.8V)

P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	GPIO_38	3	4	GPIO_39
VDD_5V	5	6	VDD_5V	GPIO_34	5	6	GPIO_35
SYS_5V	7	8	SYS_5V	GPIO_66	7	8	GPIO_67
PWR_BUTTON	9	10	SYS_RESETN	GPIO_69	9	10	GPIO_68
GPIO_30	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
GPIO_31	13	14	GPIO_50	GPIO_23	13	14	GPIO_26
GPIO_48	15	16	GPIO_51	GPIO_47	15	16	GPIO_46
GPIO_5	17	18	GPIO_4	GPIO_27	17	18	GPIO_65
I2C2_SCL	19	20	I2C2_SDA	GPIO_22	19	20	GPIO_63
GPIO_3	21	22	GPIO_2	GPIO_62	21	22	GPIO_37
GPIO_49	23	24	GPIO_15	GPIO_36	23	24	GPIO_33
GPIO_117	25	26	GPIO_14	GPIO_32	25	26	GPIO_61
GPIO_115	27	28	GPIO_113	GPIO_86	27	28	GPIO_88
GPIO_111	29	30	GPIO_112	GPIO_87	29	30	GPIO_89
GPIO_110	31	32	VDD_ADC	GPIO_10	31	32	GPIO_11
AIN4	33	34	GND_ADC	GPIO_9	33	34	GPIO_81
AIN6	35	36	AIN5	GPIO_8	35	36	GPIO_80
AIN2	37	38	AIN3	GPIO_78	37	38	GPIO_79
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	GPIO_7	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	GPIO_70	45	46	GPIO_71

UART

Note

There is a dedicated header for getting to the UART0 pins and connecting a debug cable. Five additional serial ports are brought to the expansion headers, but one of them only has a single direction brought to the headers.

4 UARTs and 1 TX only

P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	GPIO_38	3	4	GPIO_39
VDD_5V	5	6	VDD_5V	GPIO_34	5	6	GPIO_35
SYS_5V	7	8	SYS_5V	GPIO_66	7	8	GPIO_67
PWR_BTN	9	10	SYS_RESETN	GPIO_69	9	10	GPIO_68
UART4_RXD	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
UART4_TXD	13	14	GPIO_50	GPIO_23	13	14	GPIO_26
GPIO_48	15	16	GPIO_51	GPIO_47	15	16	GPIO_46
GPIO_5	17	18	GPIO_4	GPIO_27	17	18	GPIO_65
UART1_RTSN	19	20	UART1_CTSN	GPIO_22	19	20	GPIO_63
UART2_TXD	21	22	UART2_RXD	GPIO_62	21	22	GPIO_37
GPIO_49	23	24	UART1_TXD	GPIO_36	23	24	GPIO_33
GPIO_117	25	26	UART1_RXD	GPIO_32	25	26	GPIO_61
GPIO_115	27	28	GPIO_113	GPIO_86	27	28	GPIO_88
GPIO_111	29	30	GPIO_112	GPIO_87	29	30	GPIO_89
GPIO_110	31	32	VDD_ADC	UART5_CTSN+	31	32	UART5_RTSN
AIN4	33	34	GNDA_ADC	UART4_RTSN	33	34	UART3_RTSN
AIN6	35	36	AIN5	UART4_CTSN	35	36	UART3_CTSN
AIN2	37	38	AIN3	UARR5_TXD+	37	38	UART5_RXD+
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	UART3_TXD	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	GPIO_70	45	46	GPIO_71

I2C

Note

The first I2C bus is utilized for reading EEPROMS on cape add-on boards and can't be used for other digital I/O operations without interfering with that function, but you can still use it to add other I2C devices at available addresses. The second I2C bus is available for you to configure and use.

2 I2C ports

P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	GPIO_38	3	4	GPIO_39
VDD_5V	5	6	VDD_5V	GPIO_34	5	6	GPIO_35
SYS_5V	7	8	SYS_5V	GPIO_66	7	8	GPIO_67
PWR_BTN	9	10	SYS_RESETN	GPIO_69	9	10	GPIO_68
GPIO_30	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
GPIO_31	13	14	GPIO_50	GPIO_23	13	14	GPIO_26
GPIO_48	15	16	GPIO_51	GPIO_47	15	16	GPIO_46
I2C1_SCL	17	18	I2C1_SDA	GPIO_27	17	18	GPIO_65
I2C2_SCL	19	20	I2C2_SDA	GPIO_22	19	20	GPIO_63
I2C2_SCL	21	22	I2C2_SDA	GPIO_62	21	22	GPIO_37
GPIO_49	23	24	I2C1_SCL	GPIO_36	23	24	GPIO_33
GPIO_117	25	26	I2C1_SDA	GPIO_32	25	26	GPIO_61
GPIO_115	27	28	GPIO_113	GPIO_86	27	28	GPIO_88
GPIO_111	29	30	GPIO_112	GPIO_87	29	30	GPIO_89
GPIO_110	31	32	VDD_ADC	GPIO_10	31	32	GPIO_11
AIN4	33	34	GND_ADC	GPIO_9	33	34	GPIO_81
AIN6	35	36	AIN5	GPIO_8	35	36	GPIO_80
AIN2	37	38	AIN3	GPIO_78	37	38	GPIO_79
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	GPIO_7	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	GPIO_70	45	46	GPIO_71

SPI

Note

For shifting out data fast, you might consider using one of the SPI ports.

2 SPI ports

P9				P8			
DGND	1	2	DGND	DGND	1	2	DGND
VDD_3V3	3	4	VDD_3V3	GPIO_38	3	4	GPIO_39
VDD_5V	5	6	VDD_5V	GPIO_34	5	6	GPIO_35
SYS_5V	7	8	SYS_5V	GPIO_66	7	8	GPIO_67
PWR_BTN	9	10	SYS_RESETN	GPIO_69	9	10	GPIO_68
GPIO_30	11	12	GPIO_60	GPIO_45	11	12	GPIO_44
GPIO_31	13	14	GPIO_50	GPIO_23	13	14	GPIO_26
GPIO_48	15	16	GPIO_51	GPIO_47	15	16	GPIO_46
SPIO_CS0	17	18	SPIO_D1	GPIO_27	17	18	GPIO_65
SPI1_CS1	19	20	SPI1_CS0	GPIO_22	19	20	GPIO_63
SPIO_DO	21	22	SPIO_SCLK	GPIO_62	21	22	GPIO_37
GPIO_49	23	24	GPIO_15	GPIO_36	23	24	GPIO_33
GPIO_117	25	26	GPIO_14	GPIO_32	25	26	GPIO_61
GPIO_115	27	28	SPI1_CS0	GPIO_86	27	28	GPIO_88
SPI1_DO	29	30	SPI1_D1	GPIO_87	29	30	GPIO_89
SPI1_SCLK	31	32	VDD_ADC	GPIO_10	31	32	GPIO_11
AIN4	33	34	GND_ADC	GPIO_9	33	34	GPIO_81
AIN6	35	36	AIN5	GPIO_8	35	36	GPIO_80
AIN2	37	38	AIN3	GPIO_78	37	38	GPIO_79
AIN0	39	40	AIN1	GPIO_76	39	40	GPIO_77
GPIO_20	41	42	SPI1_CS1	GPIO_74	41	42	GPIO_75
DGND	43	44	DGND	GPIO_72	43	44	GPIO_73
DGND	45	46	DGND	GPIO_70	45	46	GPIO_71

Getting Started

Note

This chapter is writing under Win10. The steps are familiar for the other operate systems.

STEP1. Plug in your BBGW via USB

Use the provided micro USB cable to plug your BBGW into your computer. This will both power the board and provide a development interface. BBGW will boot Linux from the **on-board 2GB** or 4GB eMMC.

BBGW will operate as a flash drive providing you with a local copy of the documentation and drivers. Note that this interface may not be used to re-configure the microSD card with a new image, but may be used to update the boot parameters using the uEnv.txt file.

You'll see the PWR LED lit steadily. Within 10 seconds, you should see the other LEDs blinking in their default configurations.

- D2 is configured at boot to blink in a heartbeat pattern
- D3 is configured at boot to light during microSD card accesses
- D4 is configured at boot to light during CPU activity
- D5 is configured at boot to light during eMMC accesses

STEP2. Install Drivers

Install the drivers for your operating system to give you network-over-USB access to your Beagle. Additional drivers give you serial access to your board.

Operating System	USB Drivers	Comments
Windows (64-bit)	64-bit installer http://beagleboard.org/static/Drivers/Windows/BONE_D64.exe	
Windows (32-bit)	32-bit installer http://beagleboard.org/static/Drivers/Windows/BONE_DRV.exe	
Mac OS X	Network Serial http://beagleboard.org/static/Drivers/MacOSX/FTDI/EnergiaFTDIDrivers2.2.18.pkg	Install both sets of drivers.
Linux	mkudevrule.sh http://beagleboard.org/static/Drivers/Linux/FTDI/mkudevrule.sh	Driver installation isn't required, but you might find a few udev rules helpful.

Note

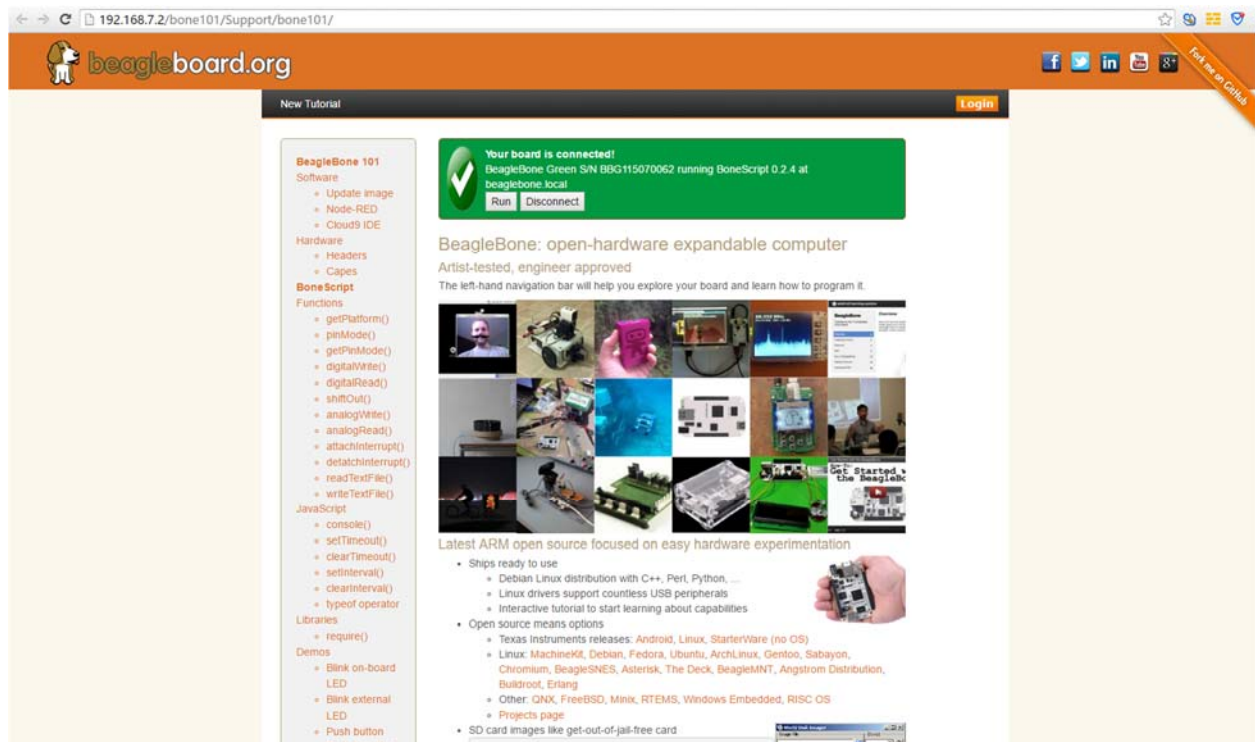
For window system, please note that:

- Windows Driver Certification warning may pop up two or three times. Click “Ignore”, “Install” or “Run”
 - To check if you’re running 32 or 64-bit Windows see [this](#).
 - <https://support.microsoft.com/en-us/help/827218/how-to-determine-whether-a-computer-is-running-a-32-bit-version-or-64-bit-version-of-the-windows-operating-system>
- On systems without the latest service release, you may get an error (0xc000007b). In that case, please [install](#) and retry:
 - <https://www.microsoft.com/en-us/download/confirmation.aspx?id=13523>
- You may need to reboot Windows.
- These drivers have been tested to work up to Windows 10

STEP3. Browse to your Beagle

Using either Chrome or Firefox (Internet Explorer will NOT work), browse to the web server running on your board. It will load a presentation showing you the capabilities of the board. Use the arrow keys on your keyboard to navigate the presentation.

Click <http://192.168.7.2> to launch to your BBGW. Older software images require you to EJECT the BEAGLE_BONE drive to start the network. With the latest software image, that step is no longer required.



The screenshot shows a web browser window at the URL <http://192.168.7.2/bone101/Support/bone101/>. The page features the BeagleBoard.org logo and a navigation menu on the left. A green notification box at the top center states "Your board is connected! BeagleBone Green S/N BBG115070062 running BoneScript 0.2.4 at beaglebone.local" with "Run" and "Disconnect" buttons. Below this, the main content area displays the heading "BeagleBone: open-hardware expandable computer" and "Artist-tested, engineer approved". A grid of images shows various BeagleBone boards and projects. The text below the grid reads "Latest ARM open source focused on easy hardware experimentation" and lists several features and options, including "Ships ready to use" (Debian Linux, Linux drivers), "Open source means options" (Texas Instruments releases, Linux distributions like MachineKit, Debian, Fedora, Ubuntu, ArchLinux, Gentoo, Sabayon, Chromium, BeagleSNES, Asterisk, The Deck, BeagleMNT, Angstrom Distribution, Buildroot, Erlang), "Other: QNX, FreeBSD, Minix, RTEMS, Windows Embedded, RISC OS", and "Projects page". A "Login" button is visible in the top right corner of the page.

STEP4. Cloud9 IDE

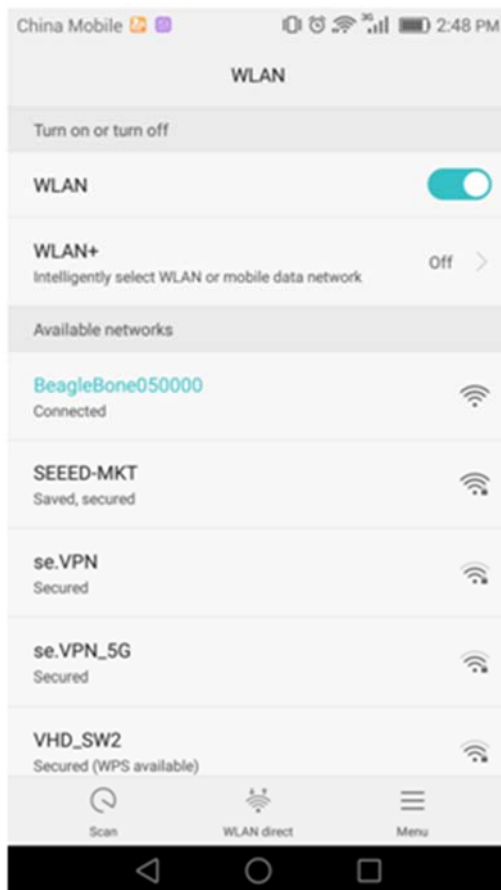
To begin editing programs that live on your board, you can use the Cloud9 IDE by click

Open Cloud9 IDE of BBG

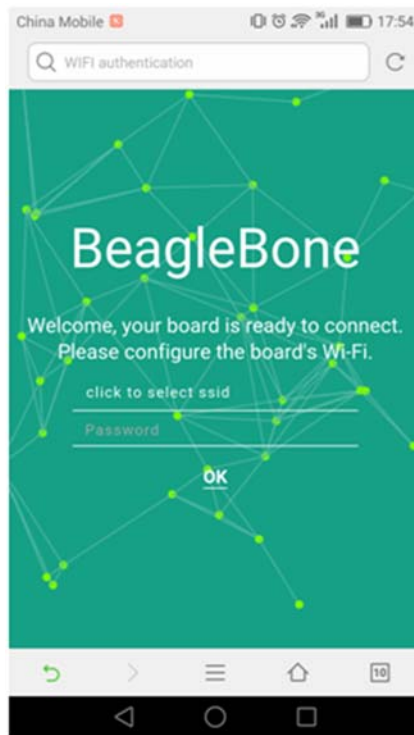
<http://192.168.7.2:3000/ide.html>

STEP5. Connect your BBGW to Wi-Fi

Using your smart phone or computer to scan local Wi-Fi network and connect to the AP named “BeagleBone XXX”



After connection succeeded, it will head to the login page automatically. Select the SSID of your Wi-Fi and enter the passwd, click OK.



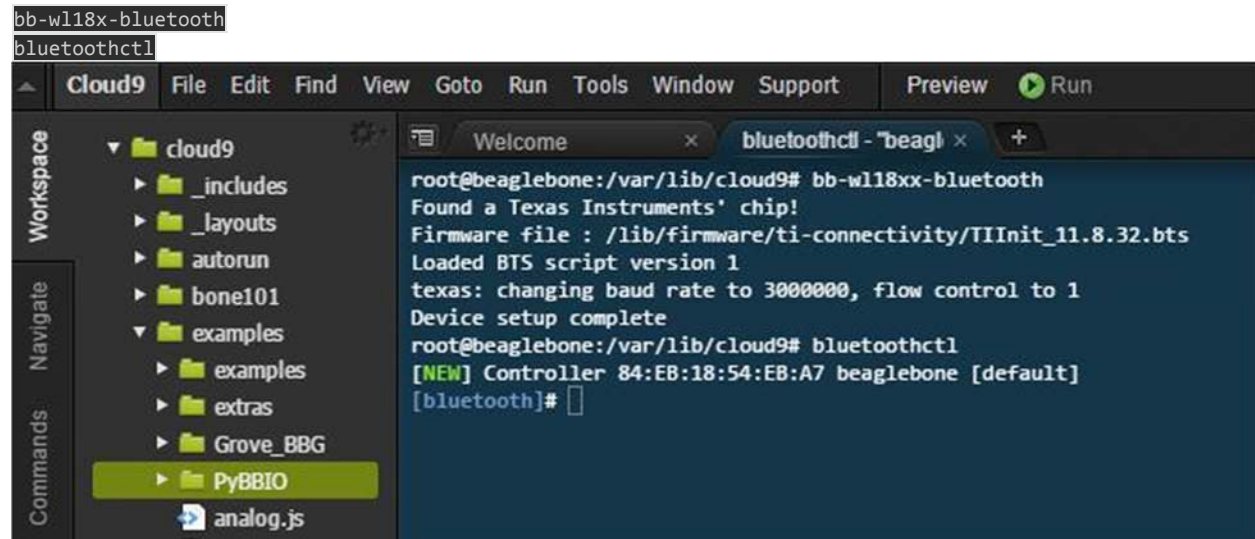
Now your BBGW is connected to Wi-Fi.



STEP6. Connect your BBGW to your Bluetooth Device

Connect to Cloud9 IDE and start a new terminal. Start the bluetooth config with the command:

```
bb-wl18x-bluetooth
bluetoothctl
```



The screenshot shows the Cloud9 IDE interface. The terminal window displays the following output:

```
root@beaglebone:/var/lib/cloud9# bb-wl18x-bluetooth
Found a Texas Instruments' chip!
Firmware file : /lib/firmware/ti-connectivity/TIInit_11.8.32.bts
Loaded BTS script version 1
texas: changing baud rate to 3000000, flow control to 1
Device setup complete
root@beaglebone:/var/lib/cloud9# bluetoothctl
[NEW] Controller 84:EB:18:54:EB:A7 beaglebone [default]
[bluetooth]#
```

Type `scan on` to scan local bluetooth devices. My device named “jy” is found.

```
[bluetooth]# scan on
Discovery started
[CHG] Controller 84:EB:18:54:EB:A7 Discovering: yes
[NEW] Device 00:11:11:11:11:07 LAPTOP-Q2IJ7NUQ
[NEW] Device 10:10:10:10:10:0A 红米手机
[NEW] Device 0C:0C:0C:0C:0C:0B 0C-0C-0C-0C-0B
[NEW] Device 49:49:49:49:49:47 49-f-49-49-47
[CHG] Device 10:10:10:10:10:0A RSSI: -79
[CHG] Device 0C:0C:0C:0C:0C:0B RSSI: -59
[CHG] Device 0C:0C:0C:0C:0C:0B Name: jy
[CHG] Device 0C:0C:0C:0C:0C:0B Alias: jy
[CHG] Device 0C:0C:0C:0C:0C:0B UUIDs:
00001200-0000111f-0000112f-0000110a-0000110c-00001132-00000000-2d8d2466
[NEW] Device 00:11:11:11:11:C6 E3
[bluetooth]#
```

Copy the device mac address, then connect to the device with the command:

```
pair 0C:xx:xx:xx:xx:0B
trust 0C:xx:xx:xx:xx:0B
connect 0C:xx:xx:xx:xx:0B

[bluetooth]# pair 0C:xx:xx:xx:xx:0B
Attempting to pair with 0C:xx:xx:xx:xx:0B
[CHG] Device 0C:xx:xx:xx:xx:0B Connected: yes
[CHG] Device 0C:xx:xx:xx:xx:0B Modalias: bluetooth:v004Cp6F02d0930
[CHG] Device 0C:xx:xx:xx:xx:0B UUIDs:
00000000-00001000-0000110a-0000110c-0000110e-00001116-0000111f-0000112f-00001132-00001200-
[CHG] Device 0C:xx:xx:xx:xx:0B Paired: yes
Pairing successful
[CHG] Device 0C:xx:xx:xx:xx:0B Connected: no

[bluetooth]# trust 0C:xx:xx:xx:xx:0B
[CHG] Device 0C:xx:xx:xx:xx:0B Trusted: yes
Changing 0C:xx:xx:xx:xx:0B trust succeeded
[bluetooth]# connect 0C:xx:xx:xx:xx:0B
Attempting to connect to 0C:xx:xx:xx:xx:0B
[CHG] Device 0C:xx:xx:xx:xx:0B Connected: yes
Connection successful
```

Now your BBGW is connected to your bluetooth device. Type `quit` back to the terminal. Play music on BBGW, then you will hear music on your bluetooth speaker device.

Update to latest software

You need to update the board to latest software to keep a better performance, here we will show you how to make it step by step.

STEP 1. Download the latest software image

First of all, you have to download the suitable image here.

<https://www.dropbox.com/s/9qsa75cazhjgb1x/BBGW-blank-debian-8.4-seeed-iot-armhf-2016-06-27-4gb1.zip?dl=0>

Note

Due to sizing necessities, this download may take about 30 minutes or more.

The file you download will have an **.img.xz** extension. This is a compressed sector-by-sector image of the SD card.

STEP2. Install compression utility and decompress the image

Download and install **7-zip**. <http://www.7-zip.org/download.html>

Note

Choose a version that suitable for your system.

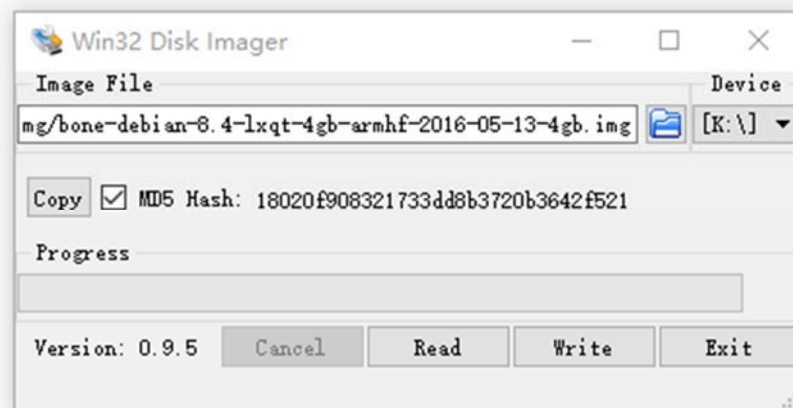
Use 7-zip to decompress the SD card **.img file**

STEP3. Install SD card programming utility

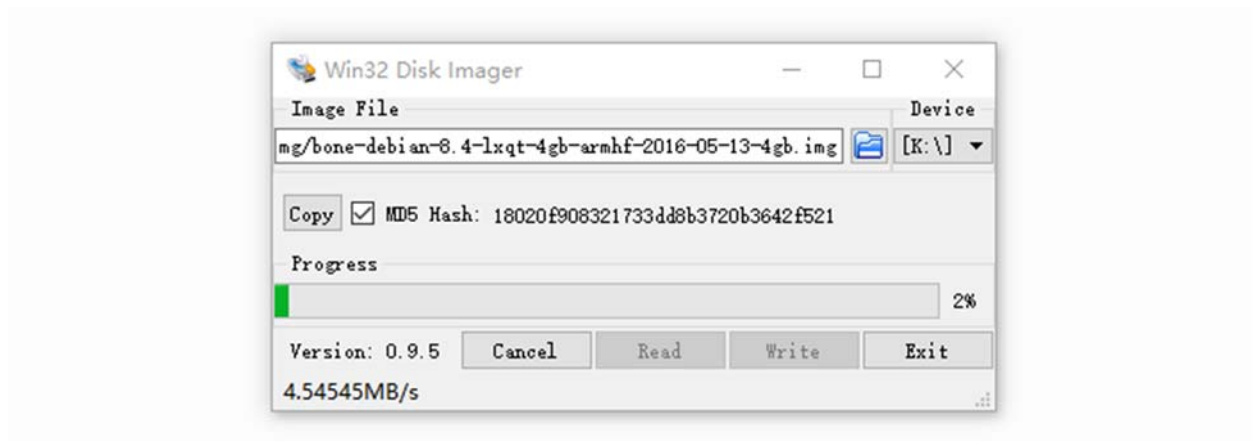
Download and install **Image Writer for Windows**. Be sure to download the binary distribution. <https://sourceforge.net/projects/win32diskimager/files/latest/download>

STEP4. Write the image to your SD card

You need a SD adapter to connect your microSD card to your computer at the first. Then use the software Image Write for Windows to write the decompressed image to your SD card.



Click on **Write** button, then the process is started.



Note

- You may see a warning about damaging your device. This is fine to accept as long as you are pointing to your SD card for writing.
- You should not have your BeagleBone connected to your computer at this time.
- This process may need up to 10 minutes.

STEP5. Boot your board off of the SD card

Insert SD card into your (powered-down first) board. Then the board will boot from the SD card.

Note

If you don't need to write the image to your on-board eMMC, you don't need to read the last of this chapter. Otherwise please go ahead.

If you desire to write the image to your on-board eMMC, you need to launch to the board, and modify a file.

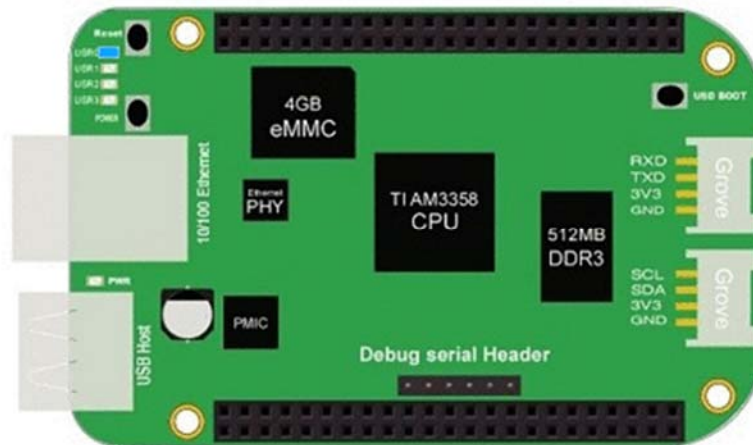
In **/boot/uEnv.txt**:

```
#Enable Generic eMMC Flasher:  
#Make sure, these tools are installed: dosfstools rsync  
#cmdline=init=/opt/scripts/tools/eMMC/init-eMMC-flasher-v3.sh
```

Change to:

```
#Enable Generic eMMC Flasher:  
#Make sure, these tools are installed: dosfstools rsync  
cmdline=init=/opt/scripts/tools/eMMC/init-eMMC-flasher-v3.sh
```

Then you will find the 4 user led light as below:



Note

If you don't find the upper tracing light, please power down and power up the board.

When the flashing is complete, all 4 USRx LEDs will be **off**. The latest Debian flasher images automatically power down the board upon completion. This can take up to **10 minutes**. Power-down your board, remove the SD card and apply power again to be complete.

Program Grove Module with Mraa and UPM

We have provided Mraa library and UPM library to make it easy for developers and sensor manufacturers to map their sensors & actuators on top of supported hardware and to allow control of low level communication protocol by high level languages & constructs.

What are Mraa and UPM?

Mraa is a C/C++ library with bindings to Python, Javascript and Java to interface with the I/O on BBB, BBGW and other platforms, with a structured and sane API where port names/numbering matches the board that you are on. Use of Mraa does not tie you to specific hardware with board

detection done at runtime you can create portable code that will work across the supported platforms. UPM is a high level repository for sensors that use MRAA. Each sensor links to MRAA and are not meant to be interlinked although some groups of sensors may be. Each sensor contains a header which allows to interface with it. Typically a sensor is represented as a class and instantiated. The constructor is expected to initialise the sensor and parameters may be used to provide identification/pin location on the board.

Install and update

Mraa and UPM are already installed in the system image of BBGW, so **you don't need to install it**. However if you want to update the library, or want to upgrade the library, use `apt-get update` and `apt-get upgrade` please. Refer to <https://github.com/intel-iot-devkit/mraa> and <https://github.com/intel-iot-devkit/upm> for more information.

Mraa Example

- light a led

```
import mraa
import time
mraa.gpio60 = P9_14 = GPIO_50
led = mraa.Gpio(60)
led.dir(mraa.DIR_OUT)

while True:
    led.write(1)
    time.sleep(1)
    led.write(0)
    time.sleep(1)
```

- Grove - PIR Sensor

```
import mraa
import time
mraa.gpio73 = P9_27 = GPIO_115
pir = mraa.Gpio(73)
pir.dir(mraa.DIR_IN)

while True:
    print (pir.read())
    time.sleep(1)
```

- Grove - Rotary Angle Sensor

```
import mraa
import time
mraa.ai01 = AIN0
rotary = mraa.Aio(1)
```

```

while True:
    print(rotary.read())
    time.sleep(1)

```

- More Tutorials

Grove - 3-Axis Digital Accelerometer($\pm 16g$) Grove - Variable Color LED

Grove - Mini Fan Grove - PIR Motion Sensor Grove - Rotary Angle Sensor

Grove - Relay Grove - Sound Sensor

Grove - OLED Display 0.96" Grove - Light Sensor Grove - Temperature Sensor

Grove - GPS Grove - Button(P) Grove - Buzzer Grove - RTC v2.0

Mraa Map for BBGW

BBGW Mraa Gpio					
Mraa	phy	GPIO	Mraa	phy	GPIO
7	P8_07	GPIO_66	42	P8_42	GPIO_75
8	P8_08	GPIO_67	43	P8_43	GPIO_72
9	P8_09	GPIO_69	44	P8_44	GPIO_73
10	P8_10	GPIO_68	45	P8_45	GPIO_70
13	P8_13	GPIO_23	46	P8_46	GPIO_71
19	P8_19	GPIO_22	57	P9_11	GPIO_30
27	P8_27	GPIO_86	59	P9_13	GPIO_31
28	P8_28	GPIO_88	60	P9_14	GPIO_50
29	P8_29	GPIO_87	61	P9_15	GPIO_48
30	P8_30	GPIO_89	62	P9_16	GPIO_51
31	P8_31	GPIO_10	63	P9_17	GPIO_5
32	P8_32	GPIO_11	64	P9_18	GPIO_4
33	P8_33	GPIO_9	67	P9_21	GPIO_3
34	P8_34	GPIO_81	68	P9_22	GPIO_2
35	P8_35	GPIO_8	69	P9_23	GPIO_49
36	P8_36	GPIO_80	70	P9_24	GPIO_15
37	P8_37	GPIO_78	71	P9_25	GPIO_117
38	P8_38	GPIO_79	72	P9_26	GPIO_14
39	P8_39	GPIO_76	73	P9_27	GPIO_115
40	P8_40	GPIO_77	87	P9_41	GPIO_20
41	P8_41	GPIO_74	88	P9_42	GPIO_7

BBGW Mraa I2C			
Mraa	I2C	PIN	FUN
0	I2C1	P9_17	I2C1_SCL
		P9_18	I2C1_SDA
1	I2C2	P9_19	I2C2_SCL
		P9_20	I2C2_SDA

BBGW Mraa PWM		
Mraa	PWM	PIN
68	EHRPWM0A	P9_22
67	EHRPWM0B	P9_21
36 60	EHRPWM1A	P8_36
		P9_14
62 34	EHRPWM1B	P9_16
		P8_34
19 45	EHRPWM2A	P8_19
		P8_45
13 46	EHRPWM2B	P8_13
		P8_46
88	ECAPPWM0	P9_42

BBGW Mraa ADC	
Mraa	ADC
1	AIN0
2	AIN1
3	AIN2
4	AIN3
5	AIN4
6	AIN5
7	AIN6

BBGW Mraa UART				
MRAA	PIN	FUN	UART	DEV
0	P9_24	UART1_TXD	UART1	/dev/ttyO1
	P9_26	UART1_RXD		
1	P9_21	UART2_TXD	UART2	/dev/ttyO2
	P9_22	UART2_RXD		
2			UART3	/dev/ttyO3
3	P9_13	UART4_TXD	UART4	/dev/ttyO4
	P9_11	UART4_RXD		
4	P8_37	UART5_TXD+	UART5	/dev/ttyO5
	P8_38	UART5_RXD+		

References and Resources

References

There're many references to help you to get more information about the board.

- BeagleBoard Main Page
- BeagleBone Green Wireless info at BeagleBoard page
- BeagleBoard Getting Started
- Troubleshooting
- Hardware documentation
- Projects of BeagleBoard

Resources

- BeagleBone_Green_Wireless Schematic(pdf)
https://github.com/SeedDocument/BeagleBone_Green_Wireless/blob/master/resources/BeagleBone_Green%20Wireless_V1.0_SCH_20160314.pdf

Help us make it better

Thank you for choosing Seeed. A couple of months ago we initiated a project to improve our documentation system. What you are looking at now is the first edition of the new documentation system. Comparing to the old one, here is the progresses that we made:

- Replaced the old documentation system with a new one that was developed from Mkdocs, a more widely used and cooler tool to develop documentation system.
- Integrated the documentation system with our official website, now you can go to Bazaar and other section like Forum and Community more conveniently.
- Reviewed and rewrote documents for hundreds of products for the system's first edition, and will continue migrate documents from old wiki to the new one.

An easy-to-use instruction is as important as the product itself. We are expecting this new system will improve your experience when using Seeed's products. However since this is the first edition, there are still many things need to improve, if you have any suggestions or findings, you are most welcome to submit the amended version as our contributor or give us suggestions in the survey below, Please don't forget to leave your email address so that we can reply.

Happy hacking