Programming the BeagleBone

The BeagleBone is an ultra-small, cost-effective computer that comes with powerful hardware. It runs a fully-fledged Debian Linux OS and provides numerous electronics solutions. It gives you all the benefits of a Linux kernel, such as multitasking, multi-users, and extensive device driver support. It allows you to program in many languages, including high-level languages such as JavaScript and Python.

This book aims to exploit the hardware and software capabilities of the BeagleBone to create real-life electronics and Internet of Things (IoT) applications quickly. It is divided into two parts. The first part covers JavaScript programs. The second part provides electronics projects and IoT applications in Python. Starting with the basics needed to set up the BeagleBone and Cloud9 IDE, this book covers interfacing with various electronics components via simple programs. Finally, the book helps you create some real-life IoT applications.

Who this book is written for

This book is for anyone who wants to learn programming on embedded systems and understand key concepts such as GPIO, PWM, and I2C. It is intended for programming beginners who is willing to work on electronics projects and basic IoT applications. This book will be helpful to BeagleBone owner who wants to quickly implement small-scale home automation solutions. Some hands on is expected on C or Python. Some familiarity with electronics is helpful. However, it is not essential.

What you will learn from this book

- Connect your BeagleBone to a computer in different ways and get Cloud9 IDE running to quickly start programming on the BeagleBone
- Get to know BeagleBone extension pins, such as GPIO, and how to connect various electronic components to the BeagleBone
- Read and write to various electronic components, such as LEDs, push buttons, sensors, and motors
- Grasp in-depth theory on Analog, PWM, and BUS programming and the electronic components used in programs
- Handle data to and from various BUS-supporting modules, such as UART, I2C, and SPI, using the Adafruit BBIO Python library
- Write real-life IoT applications in JavaScript and Python, such as sending an e-mail on overheat and controlling a servo motor remotely
- Make use of free online cloud services to store and analyze sensor data collected on the BeagleBone
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 ’Cloud9 IDE’
- A synopsis of the book’s content
- More information on Programming the BeagleBone
Yogesh Chavan is a Linux enthusiast and open source promoter, living in Pune, India. He has now been teaching operating systems, device drivers, and embedded systems in colleges and institutes for more than three years. These include the computer science department of Pune University, Symbiosis Institute of Computer Studies and Research (SICSR) college, and the Centre for Development of Advanced Computing (C-DAC) institute. He has about six years of experience in the software industry. His previous industrial role was of a software maintenance engineer at Red Hat Software Services.

Yogesh has overseen many engineering projects under GEEP (http://www.geeksofpune.in). He has given many talks at the annual open source event, GNUnify (http://gnunify.in).

Yogesh enjoys cutting edge technologies. He is a hardware hobbyist who likes playing with smartphones, tablets, routers, and Arduino. He is big fan of Android and various Linux distributions. He has compiled and modified kernel and firmwares for many phones and routers. He wishes to be a humble contributor to the open source world.
Preface

The whole world is moving from desktop/computers to smartphones/embedded systems. We are at the door to Internet of Things (IoT). The number of Internet connected users crossed 3.2 billion in 2015, which is almost half of the total population. The number of connected devices has exceeded the number of humans already. IoT is going to bring a combination of solutions, mixing physical world and digital world. We will interact with the digital world in newer ways than keyboard and mouse. Everyday objects will have logic in them to do their work in a better way. Multiple things will work collectively in order to achieve better results. IoT will bring lot many possibilities and opportunities along.

IoT implementation needs power-efficient embedded system solutions. There has been an exponential rise in the demand of the embedded systems in the last few years. Therefore, there is a demand for embedded programming as well. Embedded development boards such as BeagleBone play a key role here. These boards provide opportunities to new programmers to get their hands on ARM-based embedded systems. BeagleBone comes as an ultra-small, cost-effective solution with a powerful hardware that runs Debian Linux. This book tries to explore the hardware and software capabilities of BeagleBone to create real-life electronics and IoT applications quickly.
Being an open source hardware board, BeagleBone is the perfect choice to study embedded systems. Board design files and in-depth datasheets are open for being studied. It comes with an Ethernet port that allows deploying the IoT projects without making any addition to the board. It provides plenty of GPIO, ADC, UART, I2C, and SPI pins that make it the right choice for electronic projects. One can install Debian, Ubuntu, Android, and many other Linux distributions. By default, it comes with a full-fledged Debian Linux OS running on it. This gives you all the benefits of Linux kernel such as multitasking, multiuser, and extensive device drivers support. It also allows you to do programming in many languages, including C, C++, JavaScript, Python, Ruby, Perl, and so on. This book uses BeagleBone as a tool to write useful applications on the embedded systems. Starting with the basics to set up BeagleBone and Cloud9 IDE, this book covers interfacing with various electronic components via simple programs. The electronics theory related to these components is explained in detail before using them in a program. Then, the book covers some real-life IoT applications.

This book is divided in two parts: the first part covers programs in JavaScript and the second part of this book provides electronics projects and IoT applications in Python. Most of the physical computing theories and concepts are covered in the JavaScript part. Programs are explained in the explanation section, immediately after its source code. Troubleshooting steps are given wherever needed. Some programs have an execution section, which explains how a program works internally. Programming language conventions and error handling are loosely followed to make programs short and easy to understand. The language that is to be covered first is a tough choice. I choose JavaScript as it is energy-efficient, event-driven architecture. It is more suited as an IoT solution. Only sending the important information reduces the processing that is to be done on the Thing connected to the Internet. Javascript is a natural language of web. It comes preinstalled in the BeagleBone along with Node and Cloud9. You can start Javascript programming immediately after connecting BeagleBone. Let's start the journey of programming the BeagleBone.

What this book covers

Chapter 1, Cloud9 IDE, gives the essential information you need before you start programming on BeagleBone. It gives hardware and software-related information about the boards in the BeagleBone series. It provides you different ways to connect to BeagleBone and start Cloud9 IDE to program.

Chapter 2, Blinking Onboard LEDs, covers how to use Cloud9 IDE to write the first physical computing program—blinking onboard LED on the BeagleBone. It covers another program that creates a display pattern among the LEDs.
Chapter 3, Blinking External LEDs, explains the General Purpose Input/Output (GPIO) theory and how to attach the external LEDs to the GPIO pins. Then, it covers blinking LED and display pattern programs.

Chapter 4, Controlling LED Using a Push Button, teaches how to read from input components using polling. The interrupt method programs for each of these methods are covered.

Chapter 5, Reading from Analog Sensors, covers the theory about Analog I/O and how BeagleBone supports it. Then, it has programs to read from the TMP36 temperature sensor and light sensor.

Chapter 6, PWM – Writing Analog Information, explains how the Pulse Width Modulation (PWM) technique is used to write the analog information and how BeagleBone supports it. It has a program to fade-in LED and control the servo motor using PWM.

Chapter 7, Internet of Things with BeagleBone, explains how to implement IoT in JavaScript using BeagleBone. It covers important information about IoT. Then, there are three real-life examples. Two of them are programs to remotely control LED and servo motor. Another program is to shoot an e-mail alert when overtemperature is detected.

Chapter 8, Physical Computing in Python, explains rewriting all the programs that are covered from Chapter 3 to Chapter 6 in the Python language.

Chapter 9, UART, I2C, SPI Programming, covers popular buses in the embedded systems—UART, I2C, and SPI. All these protocols are explained here in detail. This chapter covers programs that communicate over each of these buses.

Chapter 10, Internet of Things Using Python, teaches IoT programs in Python. It has programs that were similar to the programs covered in Chapter 7. It has an additional program that uploads the temperature data over the Cloud website and we will receive the temperature graph over time for analysis.

Appendix A, GPIO Control in Bash, teaches how to set the direction and turn GPIO on/off directly by writing sysfs files.

Appendix B, BeagleBone Capes, provides information about BeagleBone add-on boards called capes.

Appendix C, Pinnux and the Device Tree, gives details about the new hardware description files that help the kernel to initialize BeagleBone. They are called device tree. It covers how to use the device tree filestoselecttheroleoftheBeagleBone pin among other possibilities.

This book does not cover BeagleBone Programmable Realtime Units (PRUs) and building/customising installable image.
BeagleBoard foundation has created a few low-power ARM-based open hardware boards. The most successful board among them was BeagleBone Black. Actually, it is a series of boards that match physical size with the same processor and slightly different hardware. There are three variants in this series—BeagleBone White, BeagleBone Black and BeagleBone Green. This book is about programming boards in this BeagleBone series. We will see details about BeagleBone hardware as the first topic of this chapter. Then we will learn how to connect and set up BeagleBone for our work. All BeagleBones come with a programming interface called BoneScript. It is a faster and easier way to deal with components of BeagleBone. We are going to use BoneScript for programming in the first part of this book. At the end of this chapter, we will learn about IDE (Integrated Development Environment), which helps us to do programming in BoneScript—Cloud9.

Here are the topics that will be covered in this chapter:

- BeagleBone hardware
- Installing the Debian image on SD card
- Setting up BeagleBone
- bone101 page
- Cloud9
- Hello World program
BeagleBone hardware

The BeagleBoard foundation is a non-profit corporation promoting open source hardware and software. It has been releasing low power, hacker-friendly embedded boards since 2008. They have created a few powerful and educational single board computers. These boards are sold to the public under the Creative Commons share-alike license that encourages sharing. These boards are collectively called BeagleBoards. They have a GitHub page at https://github.com/beagleboard. Here you can find hardware information files and software related to released boards. Support for these boards comes from a very active developer community. The BeagleBoard group on Google has more than 10,000 members. You can view posts and join the group here: http://beagleboard.org/Community/Forums. Their IRC (Internet Relay Chat) channel #beagle on freenode is active. You can join the channel and ask questions at http://beagleboard.org/chat. There are more than 500 different projects registered with BeagleBoard at http://beagleboard.org/project.

There are two different series of released boards by beagleboard.org. First is the BeagleBoard series. This series has candidates—original BeagleBoard, BeagleBoard-xM and BeagleBoard-X15. These are comparatively big, square-sized boards. Their processors are slightly better in terms of performance and have an additional DSP (Digital Signal Processor) that can do better audio/video processing. These boards have many peripherals available onboard. They are more powerful and comparatively costly. These boards are perfect in scenarios where major audio/video processing is involved or performance is important.

Another series is called the BeagleBone series. This series has candidates—BeagleBone White, BeagleBone Black, BeagleBone Green. These boards are compact, lightweight and share the same physical size (3.4 inch × 2.1 inch). They all have the same Texas Instruments AM335x sitara ARM Cortex-A8 processor. These boards lack DSP and lag behind in scenarios with major audio/video processing. But these processors are fine in other scenarios. They come with fewer peripherals on board. You can attach many peripherals externally. They are far cheaper than boards in the BeagleBoard series. They provide many expansion i/o pins of type GPIO/I2C/SPI/PWM/UART/CAN/ADC. So you can connect lots of sensors, modules, electronic components, displays, and so on. to these boards. All these boards share the same expansion pins layout. This means if you study expansion of one board in the series, the knowledge applies to other boards in the series too. There are small differences in these boards. Please refer to the following table to learn the differences. All these characteristics make BeagleBone a popular choice among hardware hackers. Besides hardware, they can all boot up from the same firmware image. So all software stack is the same for them. These boards can be programmed using the same programming interface. All these boards come with preinstalled Cloud9 IDE, which allows you to write and deploy applications on that board in many programming languages remotely via web browser.
This book covers programming all boards in the BeagleBone series:

<table>
<thead>
<tr>
<th></th>
<th>BeagleBone White</th>
<th>BeagleBone Black</th>
<th>BeagleBone Green</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor</strong></td>
<td>720MHz ARM CortexA8</td>
<td>1GHz ARM Cortex-A8</td>
<td>1GHz ARM Cortex-A8</td>
</tr>
<tr>
<td><strong>RAM</strong></td>
<td>256MB DDR2</td>
<td>512MB DDR3</td>
<td>512MB DDR3</td>
</tr>
<tr>
<td><strong>Storage</strong></td>
<td>microSD slot only</td>
<td>2/4GB emmc, microSD slot</td>
<td>4GB emmc, microSD slot</td>
</tr>
<tr>
<td><strong>Display/Audio</strong></td>
<td>possible via external cape or USB port</td>
<td>onboard microHDMI</td>
<td>possible via external cape or USB port</td>
</tr>
<tr>
<td><strong>Power connector</strong></td>
<td>via MiniUSB or 5.5mm DC jack(5V)</td>
<td>via MiniUSB or 5.5mm DC jack(5V)</td>
<td>Micro USB only</td>
</tr>
<tr>
<td><strong>Boot debugging</strong></td>
<td>onboard serial and JTAG via MiniUSB</td>
<td>via external USB-serial to header pins</td>
<td>via external USB-serial to header pins</td>
</tr>
<tr>
<td><strong>Other connector</strong></td>
<td>one standard 2.0 USB host port, one MiniUSB client port, one Ethernet port</td>
<td>One standard 2.0 USB host port, one MiniUSB client port, one Ethernet port</td>
<td>one UART and one I2C grove connector, one standard 2.0 USB host, one Micro USB client port, one Ethernet port</td>
</tr>
<tr>
<td><strong>Expansion</strong></td>
<td>2x 46pin i/o expansion header</td>
<td>2x 46pin i/o expansion header</td>
<td>2x 46pin i/o expansion header</td>
</tr>
</tbody>
</table>

Comparison between different BeagleBones

**BeagleBone White (BBW)**

This is the first board released in the BeagleBone series. It was released at the end of 2011. At that time, it was just called BeagleBone. Now after more boards have been released in this series, it is called Original BeagleBone or BeagleBone White (BBW). BBW brought single cable development environment. One MiniUSB cable from PC to BBW gives power to BBW, access to storage of BBW and network-over-USB capabilities for communication. The same cable also gives serial access and JTAG access for debugging via FTDI chip. For more information about BeagleBone White, visit the following links:

- [http://beagleboard.org/bone](http://beagleboard.org/bone)
- [http://elinux.org/Beagleboard:BeagleBone](http://elinux.org/Beagleboard:BeagleBone)
The BeagleBone Black board is shown in the following image:

![BeagleBone Black Diagram](image)

**BeagleBone Black (BBB)**

BeagleBone Black was released in April 2013. It arrived with more processor speed, more RAM, onboard storage and Micro HDMI connectivity for a lesser price than BBW. BeagleBone Black was very cost-effective and became a huge success. It ranked second consecutively for 2014 and 2015 in a survey of the most popular single board computers conducted by Linux gizmos. BeagleBone Black does not have JTAG access via USB like BBW. You can get serial access via a USB port by help of USB-serial driver in BeagleBone Black only after the board boots. It has a serial pins header. A special USB-serial cable is needed to get serial access via this header to get boot time serial access. Like BBW, single MiniUSB cable from the PC is sufficient to give power to BBB, access to storage of BBB and network-over-USB capabilities for communication. For more information about BeagleBone Black, visit the following links:

- [http://beagleboard.org/black](http://beagleboard.org/black)
- [http://www.elinux.org/Beagleboard:BeagleBoneBlack](http://www.elinux.org/Beagleboard:BeagleBoneBlack)
BeagleBone Green (BBG)
Because BeagleBone is an open-source hardware design, anyone can modify design and create a BeagleBone clone (similar to Arduino). BeagleBone Green is a modified version of BeagleBone Black released in October 2015 by SeeedStudio. BeagleBone Green is even cheaper than BeagleBone Black. They removed the Micro HDMI and DC barrel jack. They replaced MiniUSB port with a more common Micro USB port. Internally they are just the same. Two grove connectors are included, which makes it easier to connect a large family of grove sensors and grove modules. For more information about BeagleBone Green, visit the following links:

- [http://www.beagleboard.org/green](http://www.beagleboard.org/green)

It is important to note that throughout this book we will use the word BeagleBone to represent any board (BBW or BBB or BBG) in the BeagleBone series. So BeagleBone setup steps in the book should work for BeagleBone White as well as BeagleBone Black as well as BeagleBone Green.

Installing Debian image on SD card
Now that we know about our hardware, let's install OS. BeagleBone Black and BeagleBone Green come preinstalled with Debian Linux on emmc storage. BeagleBone also supports booting from OS on an SD card. We can use a Linux setup on emmc for exercises in this book. But it is always better to install OS on SD card and boot from it. In case of wrong configuration or unexpected problems, we can install Debian image and start hacking again. Also, it is easier to copy-paste files on an SD card to and from a PC.

The BeagleBone wiki page has a list of OS known to be working on BeagleBone: [http://elinux.org/BeagleBone_Operating_Systems](http://elinux.org/BeagleBone_Operating_Systems). It includes Android, Ubuntu, Angstrom, Minix, WinCE, and so on. Officially Linux distribution Debian is supported. All exercises in this book are tested on a wheezy 7.9 Debian image provided by [beagleboard.org](http://beagleboard.org). Still there should be no problem running these exercises on other Debian versions. Debian is a popular Linux distribution in the embedded world. It is one of the oldest and largest Linux distributions. There are more than 100 Linux distributions derived from Debian. Debian's stable branch is known to be one of the best-tested and most bug-free distribution. There are many cases of machines that run for over a year without rebooting. This is important for unattended embedded systems. Debian has thousands of installable packages. It uses APT (Advanced Package Tool) using deb packages. You can get more information about Debian on [http://debian.org](http://debian.org).
When you connect the BeagleBone to PC using a USB-to-MiniUSB cable (Micro USB for BeagleBone Green), it gets detected as flash drive providing you with a local copy of the documentation and drivers. When it autoruns, you see the browser opened with the page Getting Started. If it does not run automatically, you can manually open the file START.htm or the README.htm file inside the BeagleBone flash drive. This HTML page has a link to "update to latest software" on the left. It provides you with a step-by-step procedure with screenshots to install the latest prebuilt Debian image on the SD card. An online version of this webpage is available at: http://beagleboard.org/getting-started#update.

Here are the important steps to install the latest Debian image on the BeagleBone:

1. Latest supported images of all BeagleBoards are available at http://beagleboard.org/latest-images. Download a Debian image for BeagleBone on your PC. The same image works on BeagleBone White, Black and Green.

2. These images come compressed. On a Linux system you can extract it by right-clicking to that file in File Manager and choosing Extract Here. On a Windows system, you can extract it using decompression software 7zip available at http://www.7-zip.org/download.html. You will get an .img file. This is an actual image file.

3. Now, attach a USB SD card reader to your PC. On a Linux system, you can dump an .img file on a SD card using the following command. You need to find the name of the SD card file. You can get that in dmesg command output after you connect the SD card. Put that in place of sdx in command.
   ```
sudo dd if=<image_file_path> of=/dev/sdx bs=1M ; sync
   ```
   On a Windows system, you can use image writer software like win32diskimager, available at: http://sourceforge.net/projects/win32diskimager/files/latest/download. Select .img file and correct the SD card drive and click the write button to dump an image on the SD card.

   **Warning:** Be careful when selecting and formatting SD card files/drives. The wrong selection for an SD card can damage data on other storage.

4. Push this SD card in the BeagleBone SD card holder. Give power supply to the board and you should see user LEDs blinking, which is sign of life. In case you want to make sure you are booting from the SD card and not emmc, get shell access (covered in the next topic) and edit files /etc/issue and /etc/issue.net on the SD card to print that it is booted from the SD card. The next the shell access login will show that message.
Setting up BeagleBone

Embedded boards lack rich programming environment like x86 Desktop PC. Often they are connected to x86 Desktop for programming. Before starting programming, we need to connect BeagleBone to PC and set up the working environment. Let us see different ways to connect to BeagleBone and how to start Cloud9 programming IDE and the useful bone101 page. If you connect the BeagleBone via any one of the following ways successfully, you can skip others. You will need a little Linux commands knowledge here. If any of these connection steps do not work, try the steps given in Troubleshooting section. The most common problem is the power supply. USB 2.0 is designed to supply max 500mA current, which is fine if you are not attaching peripherals to BeagleBone. But if you are connecting Ethernet, HDMI or USB devices, you should use a 5V 2A power adapter with 5.5mm DC barrel jack to power up BeagleBone. You can get it at https://www.sparkfun.com/products/12889. On BeagleBone Green, you will have to use a 5V 2A Micro USB power adapter. You can get it at: https://www.sparkfun.com/products/12890.

By default, you can login as username debian and password temppwd on default Debian images for BeagleBone. This user has sudo access to all the commands in the current Debian image. The board has to be secured before starting to use it. Once logged in, you should change the password of debian user using the command passwd. You can also login as root with no password when asked. You should login as a root user and change the root password too using the same command.

Direct connection to monitor and keyboard

BeagleBone Black comes with a Micro HDMI port. You can connect it to HDMI monitor/TV/display using Micro HDMI-HDMI cable. You can attach USB keyboard and mouse to USB port directly or using USB hub. There is no need of a host laptop/desktop PC for this connectivity. After GUI login, you can get shell access by opening terminal emulator like lxterminal. You can browse local files on BeagleBone. You can open the browser and visit http://beaglebone.local to see the bone101 page and http://beaglebone.local:3000 to open Cloud9 IDE. There are HDMI/DVI/VGA/LCD add-on cape boards that fit on the BeagleBone and help you to achieve similar connectivity.
Ethernet over USB

For this connection, you need to connect the BeagleBone to the host laptop/desktop PC using a USB to MiniUSB cable (Micro USB in case of BeagleBone Green). This is a popular way of accessing BeagleBone and programming on Cloud9 IDE. BeagleBone treats its MiniUSB port as a virtual Ethernet port using a special driver. With the help of this driver, one can attach an IP address to MiniUSB port and access it like the Ethernet. By default, it has the static address 192.168.7.2. In order to communicate with it, you need to have usbnet/RNDIS driver installed on a connected PC. It comes preinstalled in most of the Linux systems. In that case, you will get a new Ethernet entry in output of command ifconfig. On a Windows system, you need to install RNDIS drivers manually. BeagleBone has a RNDIS driver installable in its fat storage. Please refer to the Installing Debian image on SD card section, to open the Getting Started webpage. This page has a local link to an executable file that installs RNDIS and other drivers. Click on the correct driver link on the Getting Started page according to your PC OS. You can also install the driver manually from the Drivers folder in BeagleBone's fat storage.

After installing the RNDIS2 driver, you will see the new network card interface in network settings. It should get IP address 192.168.7.1 via dhcp. If not, manually give the IP address to new network interface on your PC—192.168.7.1. You can open the bone101 page by putting http://192.168.7.2 in the browser address bar and Cloud9 IDE with http://192.168.7.2:3000/. You can get shell access on BeagleBone by running command ssh root@192.168.7.2 in the Linux/Mac terminal. On Windows systems, you will need terminal emulator like putty for this. You need to connect BeagleBone to the router via an Ethernet or USB-Wi-Fi in order to get Internet access. You can use a VNC connection to get BeagleBone GUI access. It is covered in a later section.

Troubleshooting

- Use Firefox/Chrome only for connection. Internet Explorer is not supported.
- Sometimes, BeagleBone's usb0 (Ethernet over USB) interface does not get turned on automatically. It can be turned on by doing shell access to BeagleBone and executing the command:
  ```bash
  sudo ifconfig usb0 up.
  ```
- Refer to the table at end of this topic to learn different ways you can get shell access. After getting shell access to BeagleBone, you can use utilities like ifconfig, ping and dmesg to debug further.
- If you were not able to install the RNDIS driver from the BeagleBone flash storage, you can install drivers from http://beagleboard.org/static/Drivers.
Ethernet port or USB Wi-Fi adapter

BeagleBone comes with an Ethernet port. You can connect a laptop/desktop PC and BeagleBone via a Ethernet cable directly or preferably via a router. By default, the Ethernet port of BeagleBone is configured to get automatic IP address via dhcp. There should be a dhcp server configured in your network to get the IP address assigned to the BeagleBone Ethernet port. Ethernet port connection with the Internet-connected router gives your BeagleBone access to the Internet. Then you can install/upgrade packages on the Internet. Many programs in this book rely on Internet connection to the BeagleBone. This is the most preferred way of connection as far as this book is concerned.

Another option is to connect the USB Wi-Fi adapter to BeagleBone and then connect your PC to BeagleBone via its IP address. You need to install the correct kernel firmware package according to your Wi-Fi adapter chipset using the command `apt-get`. Once configured, this connection works the same as the Ethernet connection.

If you connect it via Ethernet or a USB-Wi-Fi adapter, you can browse 
http://<<Beaglebone's ip address>> for the bone101 page and 
http://<<Beaglebone's ip address>>:3000 for Cloud9 IDE. BeagleBone Ethernet gets its IP address via DHCP. You can find out the IP address of BeagleBone from the router web login. You can also get shell access and run the command `ifconfig` to find out the IP address of the BeagleBone. Different methods to get BeagleBone shell access are given in the table at the end of this topic. So, if the BeagleBone IP address is found to be 192.168.1.2, then the bone101 page will be accessible at http://192.168.1.2. You can also ssh to root@<<Beaglebone's ip address>> to get shell access via Ethernet. You can do VNC connection to get BeagleBone GUI access, which is covered in the next section.

Troubleshooting

- Troubleshooting Ethernet/USB-Wi-Fi steps follows the same steps as Ethernet over USB troubleshooting. But for the Ethernet port, the IP addressing scheme will be different than 192.168.7.x.
- In case of a problem with a USB-Wi-Fi adapter, make a serial or Ethernet over USB connection to BeagleBone to get shell access and check whether it is getting detected by the command `dmesg`. If it is detected correctly, you can configure it using `iwconfig` and `ifconfig`. 
VNC
You can also connect via VNC (Virtual Network Computing). It is a remote desktop connection. You need to connect via the Ethernet over USB or Ethernet or USB-Wi-Fi method first to do a VNC connection. Debian for BeagleBone comes with tightvncserver preinstalled. To set up VNC get shell access to BeagleBone and run the command vncserver on it as a Debian user. Download VNC client software on PC and connect to server beaglebone:1. You will get a similar GUI like you get on an HDMI connection. You will be able to get shell access via the lxterminal terminal emulator, browse the bone101 page at http://beaglebone.local and browse cloud9 IDE at http://beaglebone.local:3000. As BeagleBone White and BeagleBone Green do not have an HDMI port, this option is useful if you want GUI access.

Troubleshooting
- The vncserver display does not always run on :1. Command vncserver prints the name of the display it is running on. Note it down and use it when connecting from the VNC viewer.
- Sometimes the IP address works instead of the name. Specify <beaglebone ip address>:1 as the server display.
- The VNC session might not use the lxde window manager. Currently Debian uses openbox window manager for the VNC session of the debian user. It shows the desktop with no icon or panel. You get shell and browser access by right-clicking on the desktop.

Serial Connection
Serial connection gives you a booting log and then a SSH-like command line interface. Please note that you cannot access Cloud9 or bone101 page or GUI using serial connection. BeagleBone provides two types of serial access. One is using a pseudo USB-serial driver on a USB port. You get a SSH-like shell access by this method. As a USB-serial driver starts working after kernel booting, you do not get boot logs by this method. BeagleBone Black on MiniUSB port and BeagleBone Green on Micro USB port provide this type of serial access. When you connect by these methods, BeagleBone appears as serial device on the connected PC. On Windows, it appears as serial COM port in the device manager. Note down the COM port number in the device manager and connect to that port via putty or hyperterminal specifying baud rate 115200.
On the Linux system, check kernel messages of the PC using the command `dmesg` to find out the name of the device it got detected as. It gets detected as `/dev/ttyUSB0` by default. You can use this command to connect it serially:

```
sudo screen /dev/ttyUSB0 115200
```

When you are done, press Ctrl + A then K to exit the screen.

Another type of serial connection possible on BeagleBone is via using a hardware chip that is responsible for making a serial connection. As an external chip is doing the work, this method gives boot time logs as well as shell access when booting is done. This connection is the only way to get logs when BeagleBone is failing to boot properly. It is useful to configure Ethernet, and debug problems. BeagleBone White has FTDI USB-to-serial conversion chip onboard. It gives serial access over a MiniUSB port. For BeagleBone Black and BeagleBone Green, the onboard USB-serial chip is absent. But there are serial header pins available onboard. You need a special USB-serial TTL cable to attach to these header pins. Any FTDI or PL2303 or cp210x chip based USB TTL cable can be used here. The FTDI cable from Adafruit is popular which can be found at https://www.adafruit.com/products/70. You will have to install drivers corresponding to the chip used in the cable. Drivers for these chips come preinstalled in most of the Linux systems and the serial port gets detected as `/dev/ttyUSB0` by default. You can connect using the `screen` command we covered in previous paragraph. Windows drivers for FTDI or PL2303 of CP210x are available on vendor websites. Once you install the driver, you will be able to locate the COM port in the device manager and connect via the hyperterminal or putty:

<table>
<thead>
<tr>
<th>Physical ports</th>
<th>GUI</th>
<th>Shell</th>
<th>bone101</th>
<th>Cloud9</th>
<th>Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct monitor and keyboard connection</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDMI &amp; regular USB port</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Possible via Ethernet or USB-Wi-Fi</td>
</tr>
<tr>
<td>Ethernet over USB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mini/Micro USB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>via VNC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>via SSH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ethernet or USB-Wi-Fi</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ethernet or regular USB port</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>via VNC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>via SSH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mini/Micro USB or Ethernet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial connection</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mini/Micro USB or serial-header</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

[11]
Cloud9 IDE

Besides all these options, Cloud9 IDE has one tab inside it, which gives you a root shell. So, if you have Cloud9 access, there is no need to connect via terminal emulator or SSH to get BeagleBone shell access. Here is a helpful link on BeagleBone wiki about connecting BeagleBone: http://elinux.org/Beagleboard:Terminal_Shells.

The bone101 page

We have seen several ways to connect to BeagleBone and access the bone101 webpage. This webpage can be accessed on web server running on port 80 on Beaglebone. This page has information about the board and some examples of BoneScript code:
The top-most green frame tells you that the board is connected. If your board is not connected, this frame will be of orange color and it will ask you to enter the IP address of the board. The left side pane has links to supported BoneScript and JavaScript functions. The remaining page gives miscellaneous information including available OS, upgrading OS, Cloud9 quickstart information, and expansion I/O pin details.

You can always get help about BoneScript library functions from the left frame on the bone101 page. These function help information have example code snippets. These code snippets come with a run and restore button. If your board is showing a green frame indicating that you are connected, then you can run these examples directly from the bone101 page. You can modify the code on the webpage and run new code by pressing the run button. If you want to go back to the unmodified original example code, press the restore button. You can even open this page from a smartphone browser with the correct BeagleBone IP address and run code from there.

If you are connected to BeagleBone via Ethernet over USB, you get connected to the topmost green frame automatically. If you are connected to the Ethernet port, you will see an orange frame asking for the IP address. You can get the IP address of the Ethernet port from your router weblogin or by getting shell access and running ifconfig. You can enter that IP and you will get green frame saying You are connected. Then you can explore online help and execute the example code.

A later part in the webpage gives a step-by-step procedure with screenshots to start Cloud9 IDE and run the first BoneScript program. The last part of this page gives details about BeagleBone Black hardware. It has links to the online hardware design files and wiki page. This section gives pin details of expansion headers like which pins have GPIO, analog input, PWM, UART, I2C and SPI capabilities. We will need this information throughout the book. To get updated help information and examples of BoneScript, visit http://beagleboard.org/Support/BoneScript/. This page allows us to connect BeagleBone and run BoneScript example code on BeagleBone similar to the bone101 page.
Cloud9 IDE runs on port number 3000 on BeagleBone. The Setting up BeagleBone section covered many ways to connect to BeagleBone and open Cloud9 IDE. After connecting to BeagleBone and opening Cloud9 via any one of these ways, we are ready to explore BeagleBone's default programming IDE.

Cloud9 is a web-based IDE to develop primarily JavaScript and Node.js applications. It also supports many other programming languages including PHP, Java, Python and Ruby. Cloud9 runs on the web which means it works inside a browser. There is no difference if you are using Windows, Linux or MAC OS on your PC or even from your smartphone. If you are using a supported Firefox or Chrome browser on a remote PC/smartphone, you can run Cloud9 and do programming on BeagleBone. Cloud9 allows many developers to code simultaneously on single project through the web. It supports instant deployment to many well-known cloud platforms like Microsoft Azure, Google App Engine and OpenShift. It is open source software and maintained by a company called Cloud9 IDE, Inc. You can get the source code here: https://github.com/c9/core:
Cloud9 is made up of multiple child windows inside. In the leftmost windows you can select a project related window among **Workspace** or **Navigate** or **Commands**. The **Workspace** window has a tree view that expands and collapses project files. You can add new, rename and delete project files from this window. At the bottom, there is a console window to display program output, compilation errors, and `console.log()` messages. When you run any program, you get output printed in this window. You can add a new JavaScript immediate tab in this window. It is useful to evaluate expressions, execute statements and print variables values. You can also add a bash shell in this window tab. You can run commands directly on BeagleBone from here.

At the right, there is a place for toolbars. **Debugger** and **Outline** toolbars take place in this area. Expand the **debugger** toolbar when you want to debug a program line by line. The middle empty part is the code editor area. Here actual code gets written. It has very small pane area at the left side. It shows line numbers and breakpoints, errors, and warning signs corresponding to that line. Cloud9 supports multiple tabs to edit many files simultaneously. It provides code completion features for snippets and identifiers. It has rich debugging features like setting breakpoints and step into/step over. It provides dragging and dropping of files from computer to code area. This way, you can add new ready-made code to your project. There are many customization options available with a menu bar at the uppermost side.

We are going to use Cloud9 for all BoneScript programs. There are some ready-made files with code in the demo directory. But they need some hardware setup. So, we will skip them. The best way to get help is to visit the bone101 page or [http://beagleboard.org/Support/BoneScript/](http://beagleboard.org/Support/BoneScript/).

### Hello World program

Let’s do a quick and dirty **Hello World** JavaScript program. In this program, we are just printing **Hello World** in the console view provided by Cloud9:

1. Go to the **File** menu. Click on the **New File** button. You will see a new code editor tab opened with title **Untitled1**.
2. Write the following code in the code editor view:

   ```javascript
   // My first hello world program.
   console.log("Hello World");
   ```

---

**Downloading the example code**

You can download the example code files from your account at [http://www.packtpub.com](http://www.packtpub.com) for all the Packt Publishing books you have purchased. If you purchased this book elsewhere, you can visit [http://www.packtpub.com/support](http://www.packtpub.com/support) and register to have the files e-mailed directly to you.
3. Go to the **File** menu again and save as `hello.js`. Now, you should see the tab title is changed from `Untitled1` to `hello.js`. Now Cloud9 knows this is JavaScript program. It will highlight the program with different colors.

4. Click on the **run** button at the top of screen. You will see the **Debugger** toolbar from right side is expanded to the **Debug** pane. Click on the **Resume** button on expanded pane or press the `F8` key. You should see the **Hello World** text in the output view at the bottom.

**Explanation**

In this tiny code, the first line is the comment. JavaScript supports C and C++ style commenting. So the pattern `/* ... */` is used in multi-line commenting and the pattern `//` is used for single line commenting. The second line is calling the `log()` method on the object `console`. We have given the string **Hello World** as a parameter to the `log()` method. A semicolon `;` denotes a terminated statement. Here, the `console` object provides access to the browser's debugging console. `console.log()` is a method to print string. It prints **Hello World** in the console.

As we are using a built-in JavaScript object, we have not loaded a JavaScript module/library. You can try more methods provided by the console object. You can get a detailed document about JavaScript at: [http://www.w3schools.com/js/](http://www.w3schools.com/js/) and [https://developer.mozilla.org/en-US/docs/Web/JavaScript](https://developer.mozilla.org/en-US/docs/Web/JavaScript).

**Summary**

This chapter covered all the prerequisites to get programming IDE access on BeagleBone with the latest software. In this chapter we learned about different boards available in the BeagleBone series and their hardware differences. Then we learned how to install the latest Debian image on BeagleBone and various ways to connect to BeagleBone with troubleshooting steps. Once connected, we can visit the `bone101` HTML page, which has lots of information about BeagleBone. It also provides help pages for JavaScript and BoneScript functions. Then we started using Cloud9 IDE. It is a web-based IDE that runs on port number `3000` on BeagleBone. In the end, we created our first JavaScript program that prints **Hello World**.

Now, we are ready with IDE and we know how to compile and run a program on BeagleBone. In the next chapters, we will write programs that interact with the physical environment.
Where to buy this book

You can buy Programming the BeagleBone from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.