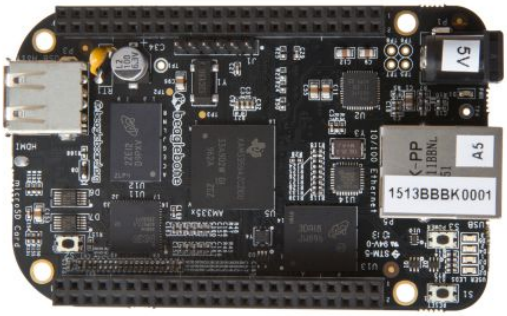
MAKER CORNER

(HTTP://WWW.MICHAELHLEONARD.COM)

This article explains in detail how to cross-compile for BeagleBone Black then easily run your program from within Eclipse.

Since I began developing the SensorCape for BeagleBone Black (http://www.michaelhleonard.com/projects/sensorcape/) I have been doing more and more work in the BeagleBone environment. One of the first things I learned is that it is possible to develop and build your applications directly on the BBB, but it is certainly not an efficient use of time. As your projects grow larger you will find yourself wanting to move away from the command line and be able to use an IDE. Additionally, the processor on the BBB will eventually begin to act as a limit on how quickly you can compile.



(http://i1.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/beaglebone-black-front.jpg)

Front Side of BeagleBone Black

To fix this problem, we would like to be able to develop and build our projects on another machine and then easily transfer to the BeagleBone Black. This doesn't sound so difficult, after-all why can't we just build on our local machine and then transfer the binary to the BeagleBone Black?

Need for Cross Compilation

The reason we can't do this is because your development platform most likely has a different processor architecture than the BeagleBone Black. Most modern computers use the Intel x86 instruction set, but many embedded systems use an ARM architecture. This essentially means that at the lowest level the two computers don't speak the same language. To get your processors speaking the same language we need to cross-compile for BeagleBone Black by using a compiler that is targeted at the ARM architecture. So let's get started!

Install a Virtual Machine on Non-Linux Systems

When I first started writing this article I wanted to be able to cover all operating systems, since it is generally a PITA to install a VM if you don't have to. Then I realized it was much worse trying to hack together a toolchain for other operating systems. In addition to this, after I started getting deeper into development I realized that it wouldn't even be practical to develop for the BeagleBone Black on a non-Linux system since it would be so difficult to obtain certain libraries and other useful programs.

So to sum it up, I know it sucks but just trust me, you don't want to deal with the headache of developing on a different platform than you deploy for. Linux distros are all basically the same under the hood, so if you develop from a Linux platform, it should be able to run on any embedded linux system.

There are quite a few virtualization options for both Windows and Mac but by far the two most popular are VirtualBox and VMWare Player. These two programs are the leader of the pack for good reasons. First they are free! Second, they both offer all of the features necessary for a casual user to fully emulate whatever environment they need.

For some unknown reason, I have always used VMWare Player on my Windows machines and VirtualBox for my Mac machines. I can't say I have a particular preference for either one really, my Windows box is much better at virtualization, but that is because it is about 2.5x the computer my MacBook Air is. In the interest of covering all the bases I will show you how to install a Linux VM using VMWare Player on Windows and I will show you how to install a Linux VM using VirtualBox on a Mac.

If you are already running a Linux system then you don't need to worry about a VM and can skip to the **"For Linux" section**. However, my instructions are for a Debian based system, so while you will definitely still be able to get this to work on other systems it will be easiest if you too are on a Debian system, that is up to you.

Choosing a Linux Distribution

There are a seemingly endless amount of Linux distributions (distros) which you could choose from. **This (slightly old) article from PCWorld (http://www.pcworld.com/article/204767/a_guide_to_todays_top_10_linux_distributions.html)** gives a quick rundown of the top 10 most popular distros. For our purposes you will want to choose a Debian distribution such as Ubuntu or Linux Mint. Ubuntu is easily the most popular distro but I think that they have gone too far in simplifying things and have made it more difficult for developers to use. Linux Mint, on the other hand, is based off of Ubuntu but does not attempt to hide the gritty technical details either.

For these reasons, I will be using Linux Mint during this tutorial but you can easily use Ubuntu without changing anything. You can download these distros from the following locations:

- **Ubuntu (http://www.ubuntu.com/download/desktop)**
- **Linux Mint (http://www.linuxmint.com/download.php)**

When choosing a download, you do not want any special installers or anything like that. After your download finishes and you have uncompressed the download you should end up with a .iso or similar disk image file once this has been verified you are ready to move on to installing your virtual machine.

Installing a Virtual Machine on Windows

A full walkthrough is coming, but in the meantime I have provided a link to the **"Getting Started" guide (http://www.vmware.com/pdf/vmware_player40.pdf)** provided by VMWare.

Skip to "For Linux" section

Installing a Virtual Machine on Mac OS

A full walkthrough is coming, in the meantime here is a very well written general guide from **Lifehacker on installing a VM using VirtualBox (http://lifehacker.com/5204434/the-beginners-guide-to-creating-virtual-machines-with-virtualbox)**.

Skip to "For Linux" section

For Linux

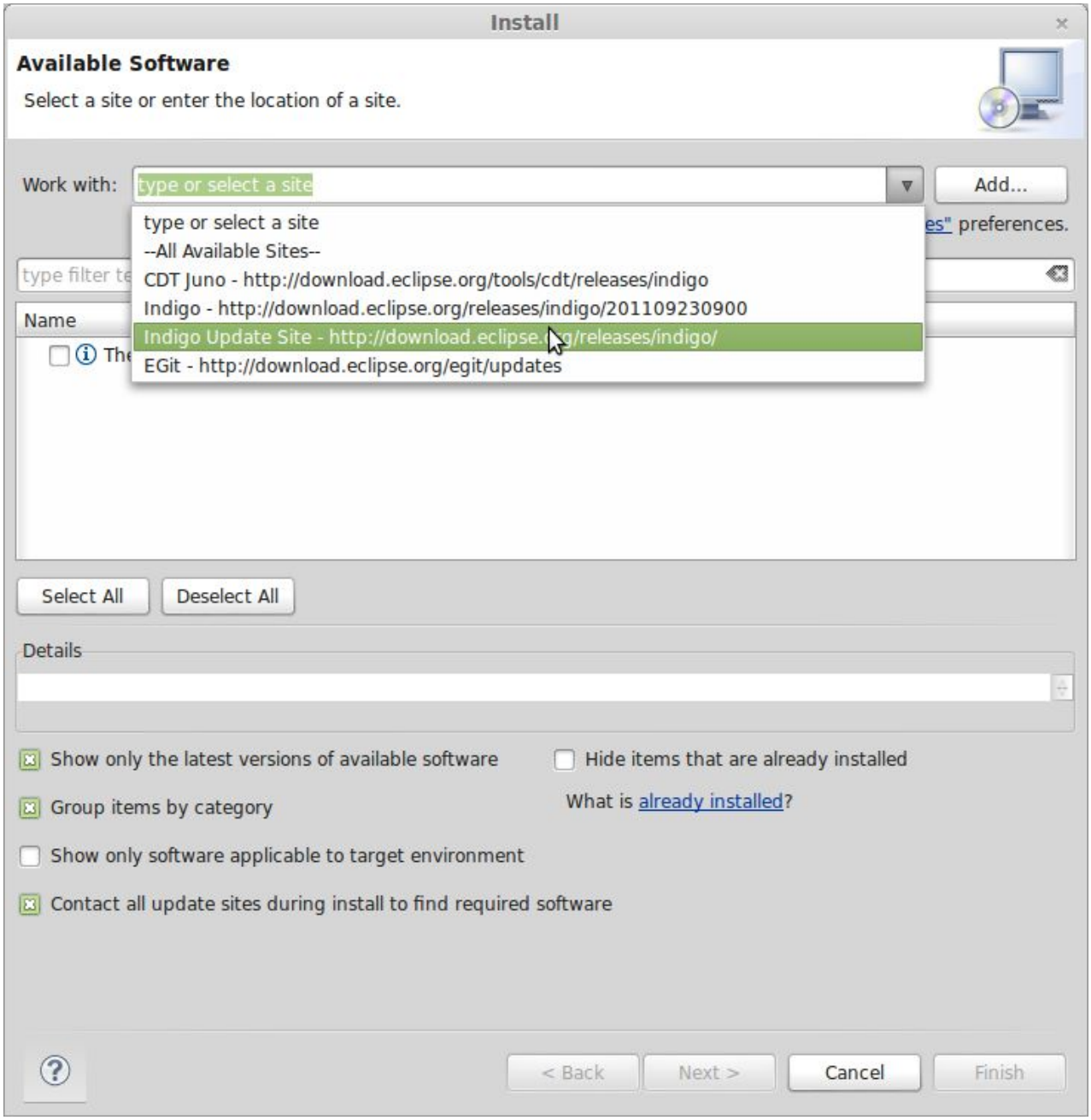
Install Eclipse, C/C++ Tools, and Remote Tools (Debian)

Open up the Terminal and enter:

```
sudo apt-get install eclipse eclipse-cdt g++ gcc
```

This command installs the main Eclipse package, the Eclipse C/C++ development tools (CDT), the GNU C++ Compiler/Linker (G++), and the GNU Compiler Collection (GCC). You most likely already have G++ and GCC installed, and they aren't even really necessary for this example, but it never hurts to verify.

After this installation finishes you need to open up Eclipse and install the "Remote System Explorer" (RSE) plugin. It is possible that RSE is included in the default Eclipse installation. To check this, go to Window -> Open Perspective -> Other... and choose "Remote System Explorer" from the Open Perspective dialog to open the RSE perspective. If it is not installed, navigate to Help -> Install New Software... Where the label says "Work with:" click on the down arrow to show a dropdown box. Select the update site for your version of Eclipse. For example my version of Eclipse is Indigo, so I choose the Indigo Update Site.

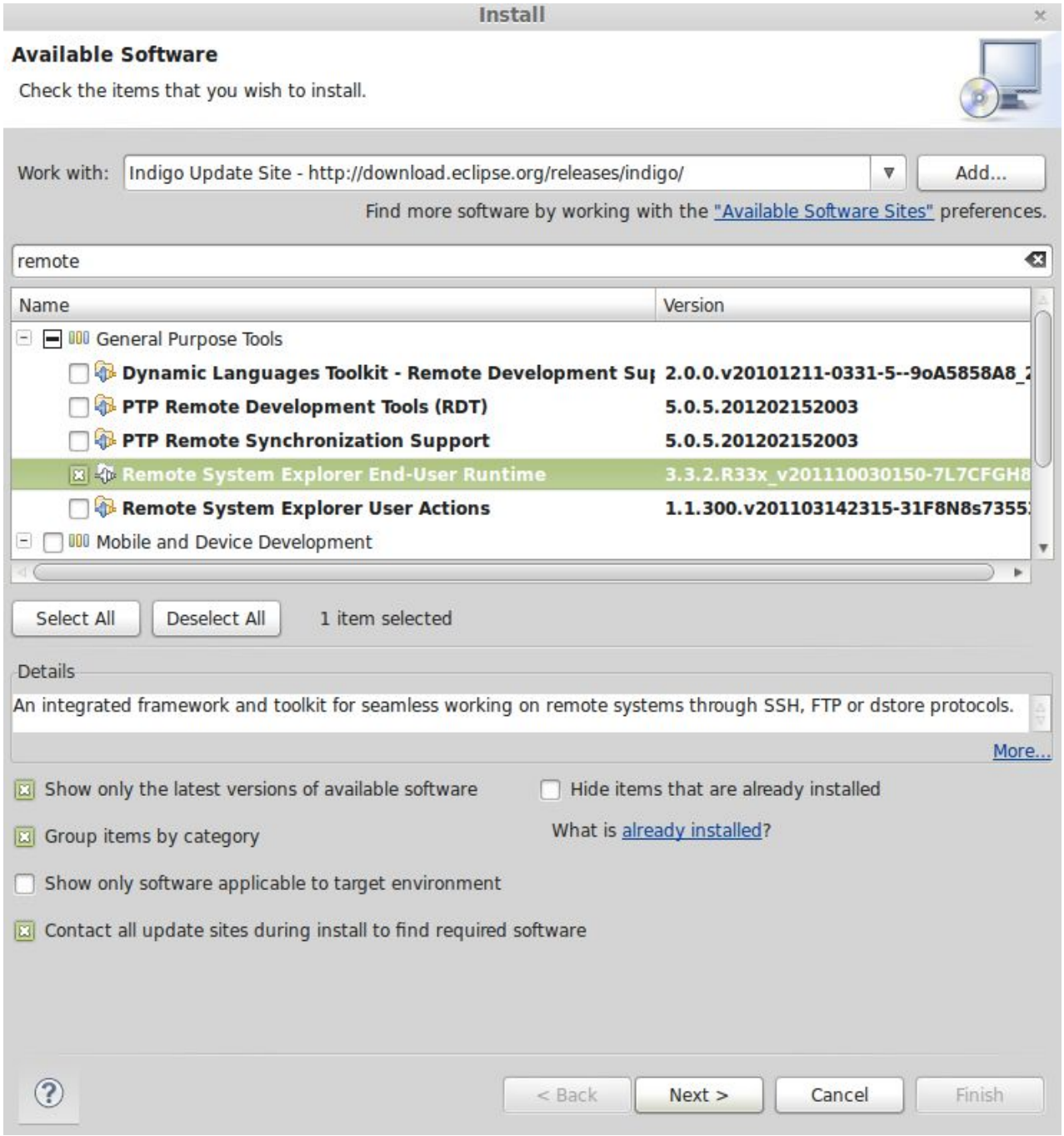


(http://i2.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/IndigoUpdate.png)

Selecting the Indigo Update Site

Then move your cursor to the search box below and enter “remote”. After a short wait you will be presented with a list of packages, one of which will be the “Remote System Explorer End-User Runtime” select this box. You will also want to scroll a bit further down under the “Mobile and Device Development” section and select the “C/C++ Remote Launch” plugin. After you have selected these packages, proceed with the wizard to finish the install. So again, be sure to install:

- Remote System Explorer End-User Runtime
- C/C++ Remote Launch



(http://i2.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/RSE.png)

Installing Remote System Explorer

Now that your basic Eclipse environment is set-up, we can move on to installing the BeagleBone Specific functionality.

Install Toolchain for BeagleBone Black

Plug your BeagleBone Black into your computer using the provided Mini-USB cable. If you are developing from a virtual machine make sure that the BeagleBone Black is mounted on your virtual machine and not on your host machine.

After the BeagleBone Black is mounted it should appear on your desktop as a removable drive. Navigate to this drive and double-click on START.htm. This will run the install scripts and verify that the BeagleBone Black is operating as expected. After a few seconds you should see the boxes for Step 1 and Step 2 turn green and earn a check-mark. This means that you are ready to move on to installation.

To install the proper toolchain for the BeagleBone Black we will pull the gcc-arm-linux-gnueabi package from the apt repository. This is included in the default Debian repositories so you can simply type...

```
sudo apt-get install gcc-arm-linux-gnueabi
```

After a short wait the ARM toolchain should install successfully and you are well on your way to being able to cross-compile for BeagleBone Black. The next step is to set up your Eclipse environment to make this process as automated as possible.

Create and Configure a Project to Build Using this Toolchain

This is where the magic happens, once we complete this section you should be able to create your own project that will compile on your local computer and then run on your BeagleBone Black with the push of a single button. Let's begin with something simple, a classic "Hello, World" application.

Creating a C++ Project

Navigate your cursor to the "Project Explorer" view. This is usually the left side-panel but if it isn't there then you can go to Window -> Show View -> Project Explorer. With your cursor positioned inside this view, right click and select New -> C++ Project.

A window will appear asking you to name your project and choose some project settings. You can name your project whatever you like, I will be naming mine "Test".

Then select ~~Executable -> Hello World C++ Project~~. If you have multiple Toolchain options just choose one, it doesn't matter since we will be changing this later.

At the next screen you will be asked to set up some of the template options, you can leave them as-is or change them to your liking. I will be changing the Hello World message to Hello BeagleBone. After you are done with this screen you can go to the next and skip that one, then select the "Finish" button.

After a moment Eclipse will finish preparing the project and you will be presented with the familiar "Hello World" program. Just to make sure there are no weird errors, go ahead and click the build button in the toolbar (it looks like a hammer). In your console at the bottom you should see a few compiler commands and it should end with a build summary. If you get errors at this point then there is most likely something wrong with your Eclipse installation, I recommend doing a few Google searches for help, then re-installing Eclipse if you can't find a solution. If all went well then you can move on to the next step.

Configure the ARM Toolchain

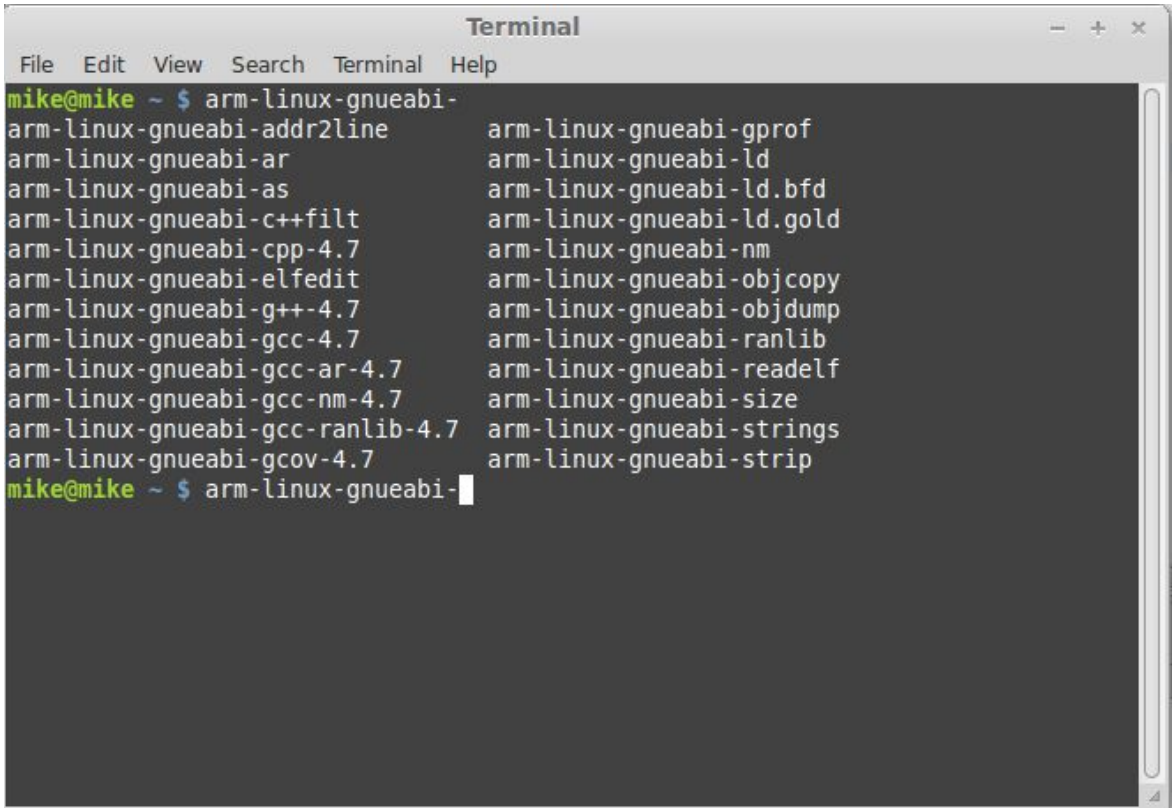
If you took a good look through your last build log you will notice that Eclipse used GCC and G++ to compile your program. This is normally fine, but in our case we want to deploy to an ARM architecture so we need to reconfigure our project to build using the toolchain we downloaded earlier.

To do this you can right click on your project name in the “Project Explorer” and select “Properties” from the menu. This will bring up a new window where you can modify the build settings of your project.

On the left sidebar of the properties window expand the “C/C++ Build” category and select “Settings”. This is where you can change your compilers, linker, and assembler, otherwise known as your toolchain. Make sure your active tab is “Tool Settings”, we will begin by changing your GCC Compiler. For my current set-up I change my compiler to:

```
arm-linux-gnueabi-gcc-4.7
```

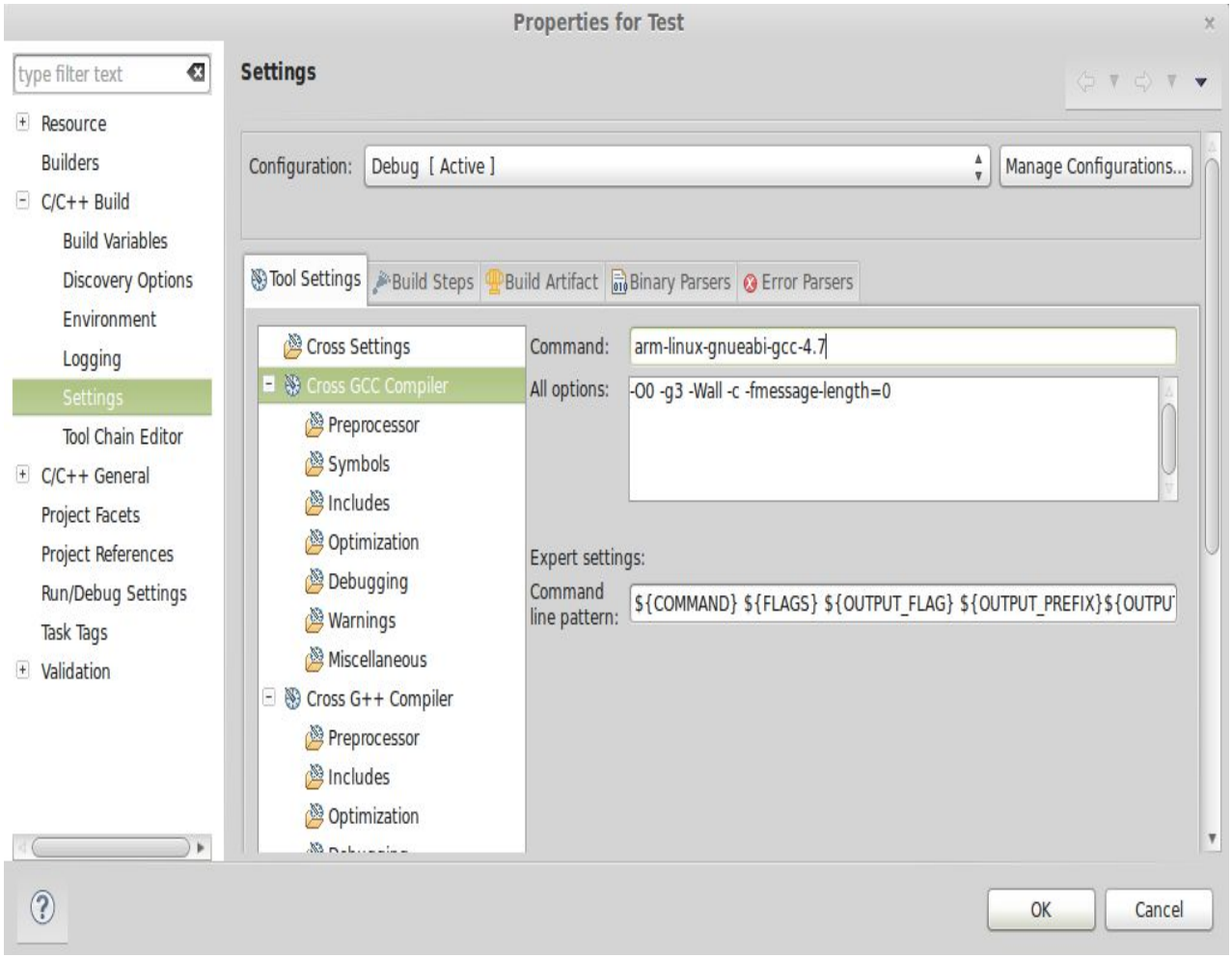
But this could be different for you, I recommend opening up a terminal window and typing arm-linux then hitting tab a few times to get a list of suggestions. This will show you what options you have available. To avoid messing up, I would just copy and paste from this list. This is what it looks like on my system.



(http://i2.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/command_line.png)

Showing available ARM compilers

You will now go through all four options and change them to their appropriate ARM equivalents. My GCC compiler command looks like this after changing it:



(http://i0.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/arm-gcc-compiler-command.png)

ARM GCC Compiler Command

Now you will need to go ahead and update the commands for the G++ compiler, G++ linker, and the GCC assembler. Since this is fairly repetitive I won't show each step, but note that (at the time of writing) the assembler is the only command that does not have a 4.7 after it. If you need help feel free to post in the comments below.

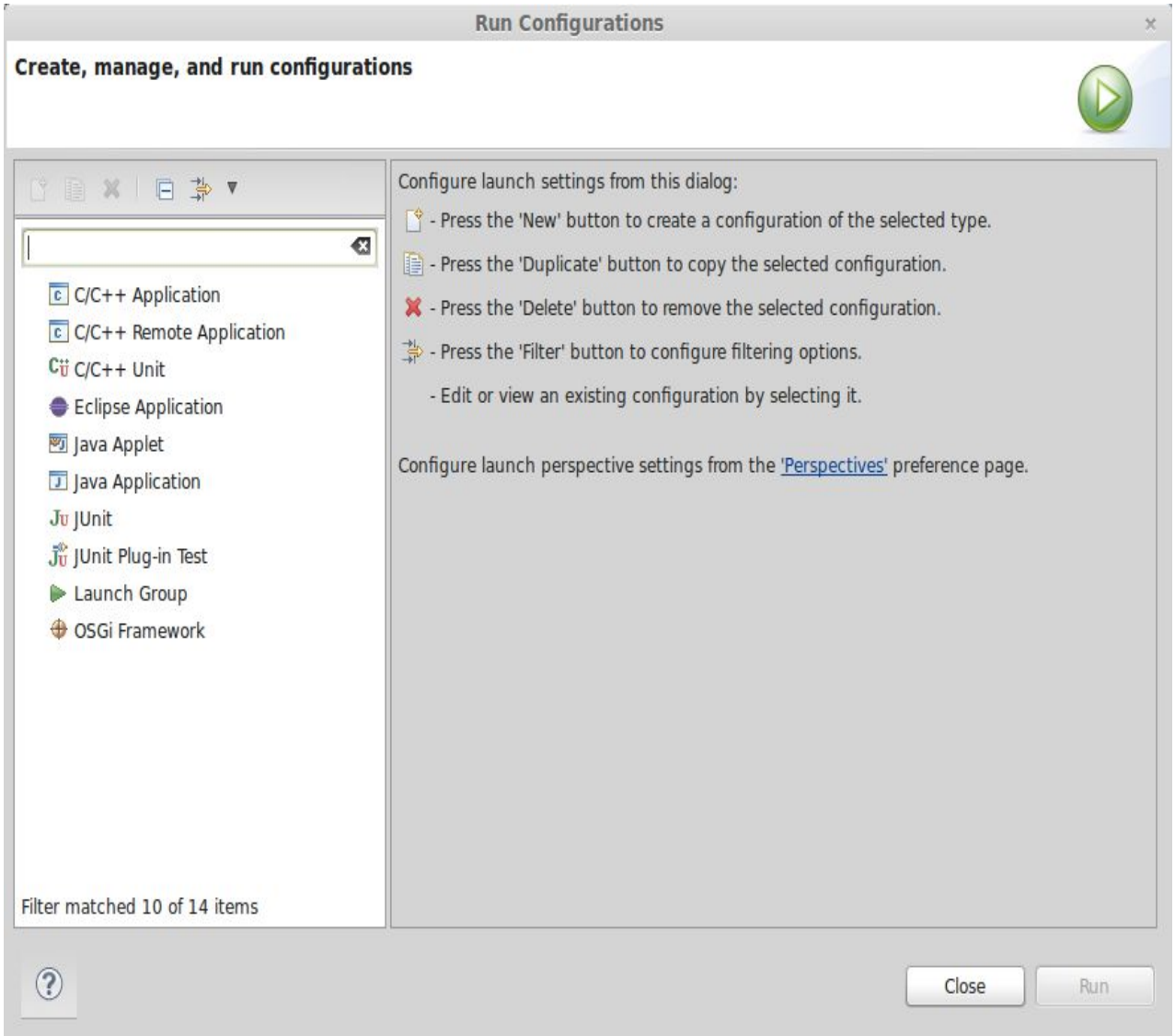
After you have updated your toolchain commands press "OK" and you will return to the main Eclipse view. To verify that everything worked as you hoped go ahead and build your project again. If it compiles without errors then congratulations, the hardest part is over and you have a binary that can run on your BeagleBone Black. This next step is just to make everything a bit easier. When we are done, you will be able to hit the run button and Eclipse will *automagically* compile your program, transfer it to the BeagleBone Black, and run it on the BeagleBone Black.

Set up Remote Deployment

Now that we have a project set up and know that it compiles, let's set it up to run on the BeagleBone Black. You will start by clicking on the down facing arrow next to the run button, as shown below.

(http://i2.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/Screen-Shot-2013-07-12-at-1.39.48-PM.png)

When you click on this arrow you will be presented with a dropdown menu, choose the "Run configurations..." option and you will see a new window that looks something like this.

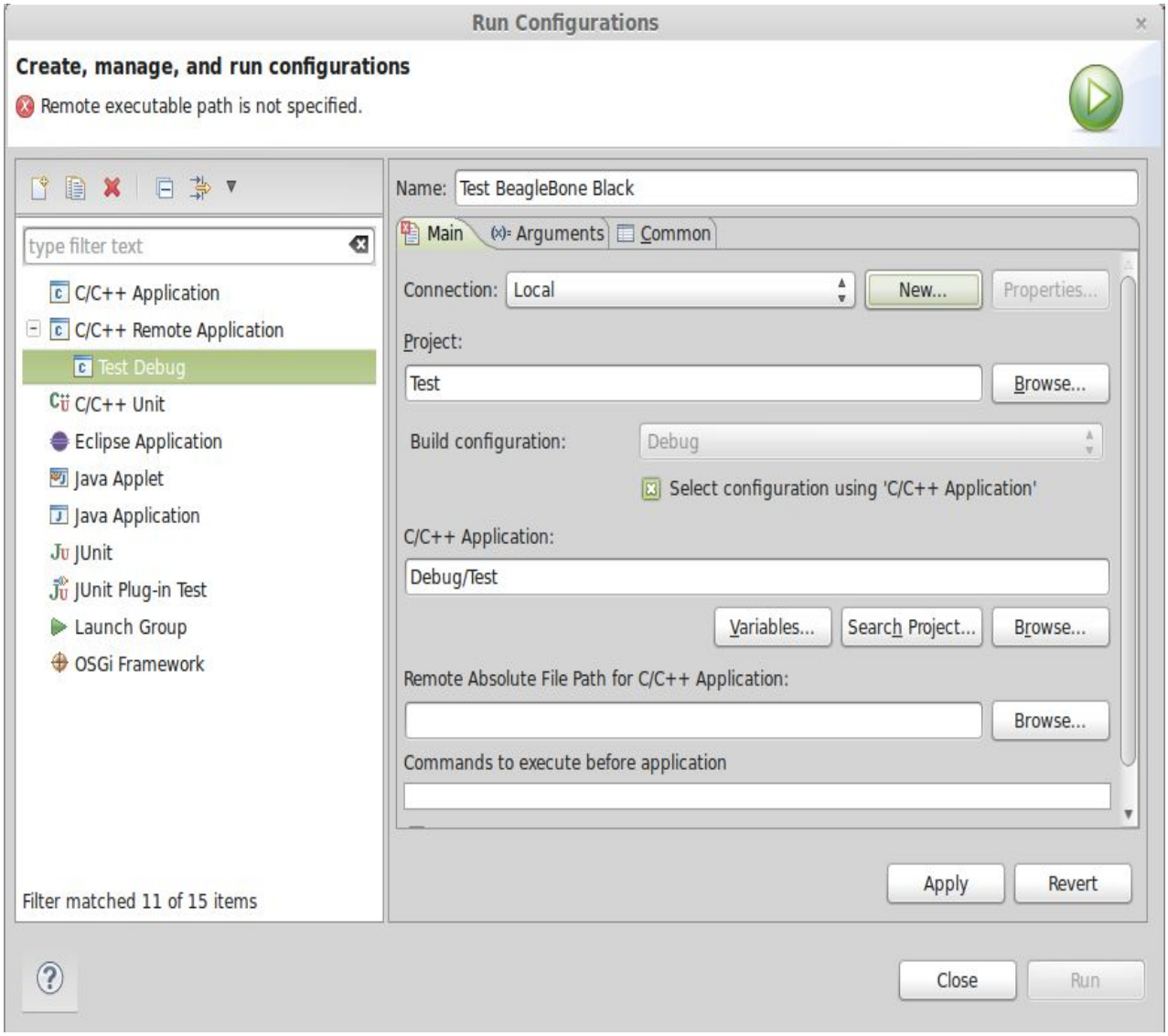


(http://i2.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/Screen-Shot-2013-07-12-at-1.40.14-PM.png)

You can see in the left sidebar that there is an option for "C/C++ Remote Application" if you do not see this on your system then you did not install all of the necessary plugins. Specifically, you forgot the C/C++ Remote Launch" plugin, return to the Eclipse marketplace and install this plugin. If you do see this option, then you

are ready to move on to setting up your connection.

To create a new run configuration just double-click on the "C/C++ Remote Application" item, after a moment Eclipse will have created a new run configuration and you should see a window that looks something like this.



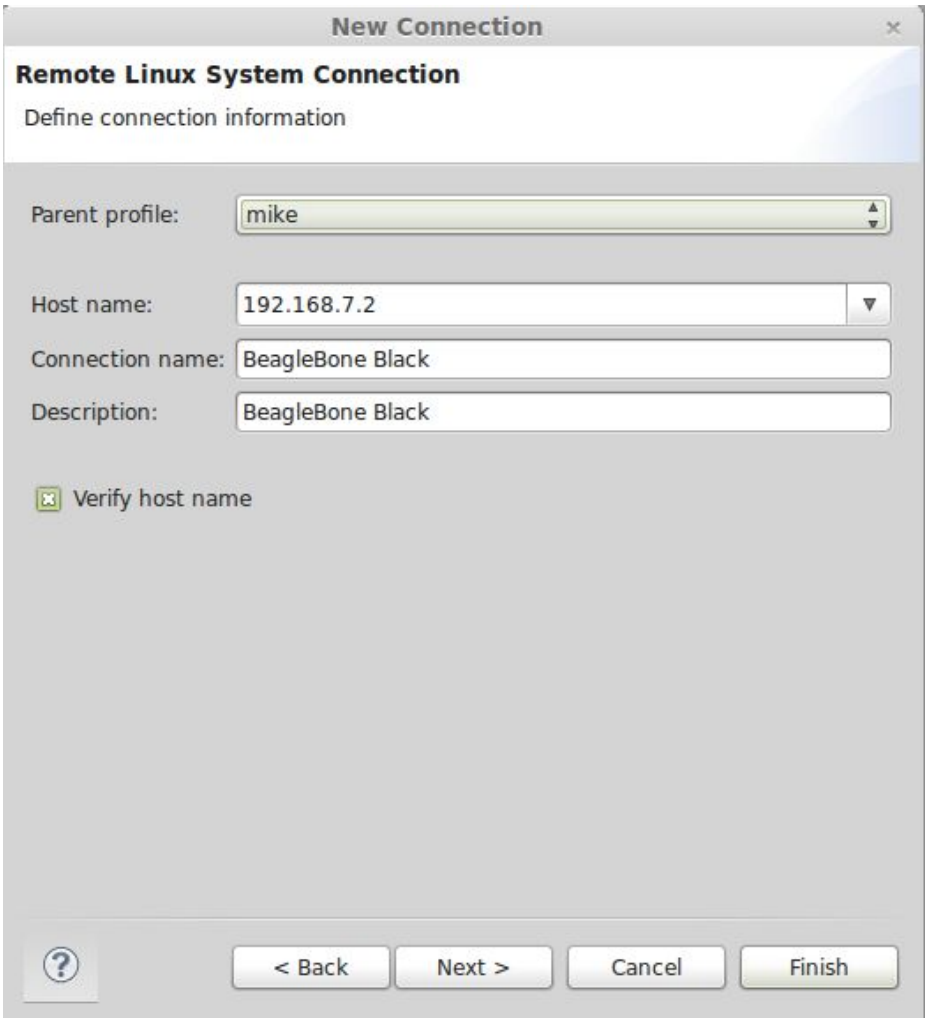
(http://i0.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/Screen-Shot-2013-07-12-at-1.41.53-PM.png)

You now have the skeleton of a remote run configuration but need to inform Eclipse of which device you would like to deploy to. In order to do this you can click on the "New..." button to create a new connection which will present you with the new connection window. You will first be asked to define the remote system type, for the BeagleBone Black you will choose "Linux" but as you can see there are many other methods of connected to devices.

On the next screen you will tell Eclipse what IP address to connect to as well as what SSH profile to use. Setting up the SSH profile is outside of the scope of this tutorial, but Eclipse should be able to handle that for you.

For my BeagleBone Black (and I believe all of them) connected by USB the IP address is 192.168.7.2. Enter this number in the "Host Name" section. If you BeagleBone Black has a different IP you will want to use that instead.

Finally, you can give the connection an easy to remember name and description, or you can leave these fields blank, that's up to you. My completed connection information is as follows.



(http://i0.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/Screen-Shot-2013-07-12-at-1.42.41-PM.png)

Click the next button and proceed to defining your connection protocols. These next four prompts allow you to define what protocols eclipse should use to communicate with your BeagleBone Black's subsystem. You will choose the following options:

1. ssh.files
2. processes.shell.linux
3. ssh.shells
4. ssh.terminals

To verify that your connection settings are correct, go to the Remote System Explorer view and right click on your new connection. In the dropdown menu and select "Connect..." after this you will be prompted for a user name and password. This should be the user name and password for the account you want to log-in to the BeagleBone Black with. If you haven't set up any non-default accounts or know that you will need root access you should just log in with the default root account by using the following credentials.

(http://i0.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/Screen-Shot-2013-07-12-at-1.44.56-PM.png)

Hit OK and you now have a working connection to the BeagleBone Black through Eclipse, you can even use Eclipse as a remote terminal now, but we won't get in to that right now. The next and final step is to configure your remote paths and to give your program execute permissions. If you don't understand what that means, don't worry, I am about to walk you through it.

Return to the "Run Configurations" window you need to begin by setting the remote path of your executable. Since you haven't yet uploaded an executable to the BBB you need to define this as the program name in your root directory. So for example, in my case the program is called "Test" and my root directory is /home/root/ this means that my remote path will be:

```
/home/root/Test
```

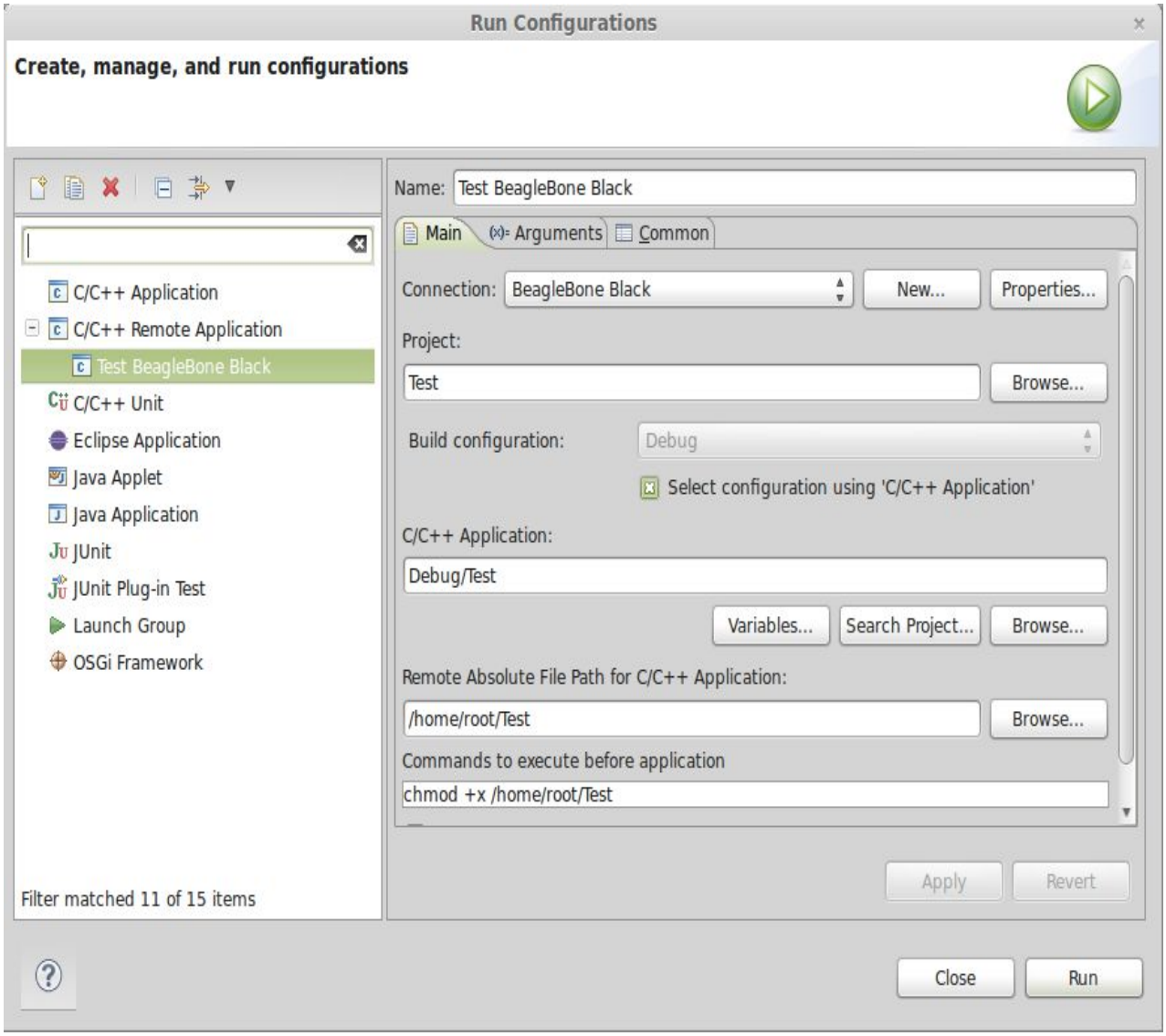
In general you will define this as:

```
/home/root/{ProjectName}
```

Finally, you will need to grant your program execute permissions. This is essentially just a flag that tells the BeagleBone Black that it's okay to run the file through the processor. You can do this by using the chmod command with the +x flag. For my project it looks like this:

```
chmod +x /home/root/Test
```

The final run configuration is:



(http://i1.wp.com/www.michaelhleonard.com/wp-content/uploads/2013/07/Screen-Shot-2013-07-12-at-1.47.54-PM.png)

Apply your configuration using the “Apply” button, then run your project and check your console, you should see an output that looks something like this...

```
root@beaglebone:~# echo $PWD'>'
/home/root>
root@beaglebone:~# chmod +x /home/root/Test;/home/root/Test;exit
Hello BeagleBone
logout
```

This shows the commands Eclipse used to run the application and shows a successful "Hello BeagleBone" message. There are any number of things that could go wrong in these steps so if you run into a problem feel free to comment below, I will do my best to help you and update this tutorial to be more clear.

References

This was a complex article and took quite a bit of research on my part. I came across several other articles that would cover a part of the process but not the whole flow, so I decided to combine all these articles as well as my own personal experiences into a unified place. I hope you have been able to learn everything you came for but if not feel free to comment below or check out these articles I read along the way.

- fortune datko – Cross-compiling applications for the BeagleBone (http://datko.net/2013/05/06/cross-compiling-applications-for-the-beaglebone/) - I don't think I actually used anything directly from this post but I remember reading over it a few times, and I think it is helpful for understanding what is going on.
- Lakeview Research – Using Eclipse to Cross-Compile Applications for Embedded Systems (http://www.lvr.com/eclipse1.htm) – A very useful series of posts showing how to set up cross-compilation in Eclipse. Whenever I first read it I didn't realize there was a tutorial on setting up remote deployment, but there definitely is. So I guess if you had a hard time understanding my post, go read this one.
- Derek Malloy - Beaglebone: C/C++ Programming Introduction for ARM Embedded Linux Development using Eclipse CDT (https://www.youtube.com/watch?v=vFv_-ykLppo) – Possibly the single most helpful resource in this list. This video essentially shows you how to do everything in this tutorial except it was originally made for the BeagleBone, and it is in video format. I don't know about you but I prefer being able to read through the instructions rather than watching a video. Anyways, if my tutorial wasn't of any use to you and the Lakeview Research site didn't help, try this video.

That's all! If you have any questions or would like to share your project ideas please leave your thoughts in the comments below. If you found this post helpful be sure to subscribe!

Google+

 (https://plus.google.com/+MichaelLeonard)Michael Leonard (https://plus.google.com/+MichaelLeonard)

Posted in: Programming (http://www.michaelhleonard.com/category/technology/programming-technology/)
Tagged: beaglebone (http://www.michaelhleonard.com/tag/beaglebone/), beaglebone black (http://www.michaelhleonard.com/tag/beaglebone-black/), coding (http://www.michaelhleonard.com/tag/coding/), programming (http://www.michaelhleonard.com/tag/programming/)

24 comments

Add a comment as Anthony Antoun

Top comments

BeagleBoard.org 8 months ago - BeagleBoard.org (Tutorials/Guides)
How to cross-compile under **#Eclipse** for BeagleBone Black

+4 1

Michael Leonard 8 months ago - Make: Forum (Raspberry Pi)

+1 1

Michael Leonard 8 months ago (edited)
I wrote this for the BeagleBone Black but most of the instructions should work pretty much the same for th Raspberry Pi.

Joshua Datko 8 months ago - Shared publicly
Glad my post was helpful! This is a very detailed post, thanks. I haven't used Eclipse for any C/C++ projects but heard of people using it for embedded development and always wondered how they set it up. If I ever have to do the same, I now know where to look.

1 · Reply

Michael Leonard 8 months ago - Beaglebone Black (Discussion)

Michael Leonard originally shared this

+5 1

Arun Kapur 1 month ago - Shared publicly
Thanks for the tutorial. I am new to this and I am lost after your step where I go to the" Run Configurations" and define what protocols eclipse should use to communicate with the BeagleBone Black's subsystem.

SEARCH THIS SITE

To search type and hit enter...

SUBSCRIBE

Enter your email address below to subscribe to this blog and receive notifications of new posts by email or subscribe using RSS to read new posts in your favorite RSS reader.

Join 58 other subscribers

Email Address

Subscribe by Email

(http://www.youtube.com

(https://www...

/Micha... QCUWCu66rZGl0nND7Pzw)