You can see from this list that the latest g++ compiler number available that supports hard floats, at the time of writing, is `g++-4.7-arm-linux-gnueabihf`. Install the latest version available to you and any dependencies, which should be automatically identified. You may have already installed some of these dependencies in earlier steps (e.g., adding Guest Additions to the Debian VM), but here is the list that was required for this installation:

```
~# apt-get install build-essential libc6-armhf-cross libc6-dev-armhf-cross
~# apt-get install binutils-arm-linux-gnueabihf linux-libc-dev-armhf-cross
~# apt-get install libstdc++6-armhf-cross
~# apt-get install gcc-4.7-arm-linux-gnueabihf g++-4.7-arm-linux-gnueabihf
```

Please note that hard floats are being used here. If you wished to use soft floats for a different platform, then you would replace *armhf* with *armel* and *gnueabihf* with *gnueabi* in the preceding list. If you are having problems with dependencies, type `sudo apt-get autoremove` and try an older version of the gcc and g++ compilers.

If all goes well, you should be able to execute the cross-compiler at this point; however, you may have multiple versions installed, or you may wish to install multiple versions in the future. If you change directory to `/usr/bin` and perform `ls arm*` you might see something like the following:

```
root@debian:/usr/bin# ls arm*
arm-linux-gnueabihf-g++-4.7    arm-linux-gnueabi-readelf
arm-linux-gnueabihf-gcc-4.6    arm-linux-gnueabi-strings
arm-linux-gnueabihf-gcc-4.7    arm-linux-gnueabi-strip...
```

For convenience, you can create generic symbolic links to the version of the compilers that you wish to use. In this case symbolic links are created to the gXX-4.7 compilers. This generic symbolic link is used when the Eclipse build settings are configured in the next section:

```
.../usr/bin# ln -s arm-linux-gnueabihf-gcc-4.7 arm-linux-gnueabihf-gcc
.../usr/bin# ln -s arm-linux-gnueabihf-g++-4.7 arm-linux-gnueabihf-g++
```

You can test that these symbolic links work, from any location (as `/usr/bin` is in the PATH by default), by using the following:

```
molloyd@debian:~$ arm-linux-gnueabihf-g++ -v
gcc version 4.7.2 (Debian 4.7.2-5)...
```

Testing the Toolchain

At this point, you can build a test application to check that everything is working before you begin configuring Eclipse. You should write a small C++ program, build it on your desktop machine, and then deploy it to the BBB. For example, enter the following on the desktop machine:

```
molloyd@debian:~$ nano testARM.cpp
molloyd@debian:~$ more testARM.cpp
#include<iostream>
int main(){
   std::cout << "Testing Toolchain" << std::endl;
   return 0;
}
molloyd@debian:~$ arm-linux-gnueabihf-g++ testARM.cpp -o testARM
molloyd@debian:~$ ./testARM
bash: ./testARM: cannot execute binary file
```

At this point, the binary executable will not execute on your desktop machine, as it contains ARM hard float instructions. However, please note that if you install QEMU (in the Installing a Change Root section of this chapter), then the program will execute. Next, transfer the binary executable to the BBB:

```
molloyd@debian:~$ sftp molloyd@192.168.7.2
BeagleBoard.org BeagleBone Debian Image 2014-05-14
Connected to 192.168.7.2.
sftp> put testARM
Uploading testARM to /root/testARM...
sftp> exit
```

Finally, SSH to the BBB to confirm that the new binary works correctly:

```
molloyd@debian:~$ ssh root@192.168.7.2
BeagleBoard.org BeagleBone Debian Image 2014-05-14...
molloyd@beaglebone:~$ ./testARM
Testing Toolchain
```

Success! If you see this output, then you are able to build a binary on the desktop machine that can be executed directly on the BBB. The next section is an advanced topic and is required if you wish to use third-party libraries with your cross-compilation environment. If you wish, you can skip this section for the moment and go directly to the Cross-Compilation Using Eclipse section.

Cross-Compilation with Third-Party Libraries (Multiarch)

This section is not necessary in order to cross-compile C/C++ applications; however, it is likely that you will need to add third-party libraries in the future for tasks such as image and numerical processing. Traditionally, this has been