SHORT GUIDE FOR DEPLOYING

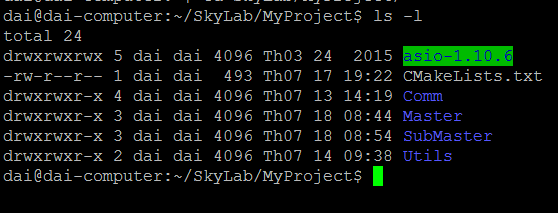
I have developed, debugged and preliminary tested the firmware on the PC running Ubuntu. The following is the steps for transferring the code to BBB and then compile, run the application on real Beagle-Bone Black.

1. First, I need to transfer the source code from my Ubuntu PC to BBB board. I assume that we have installed Debian Distribution on BBB. My BBB have IP address 10.10.1.144 and I will use default account in this guide.

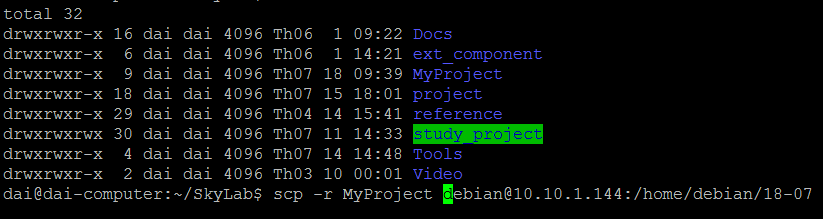
Login name: debian

Password: temppwd

This is the directory structure of project on my Ubuntu PC

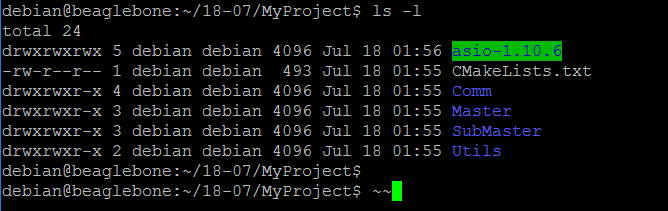


I will use the scp command to copy code to BBB



1. After transferring the source code to the BBB. It is ready to deploy the application on the real hardware. Again, I will use the putty from my Windows PC to do tasks

Showing the source code on the BBB



1. As of now, I have used cmake to re-compile the source code on the BBB. So there are some dependencies we need to install on the BBB: cmake, boost library (for some utilities)

The followings were the commands to install

debian@beaglebone:$ sudo apt-get update

debian@beaglebone:$ sudo apt-get install cmake

debian@beaglebone:$ sudo apt-get install cmake

debian@beaglebone:$ sudo apt-get install libboost-dev

1. When all dependencies were installed, we are ready to compile our project

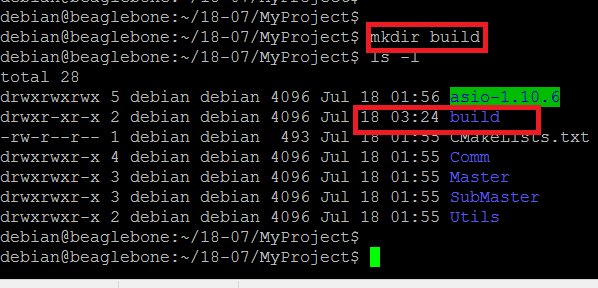
NOTE:

Before compiling the source code, you must update IP address and port settings on source code matching with current settings of your testing environment.

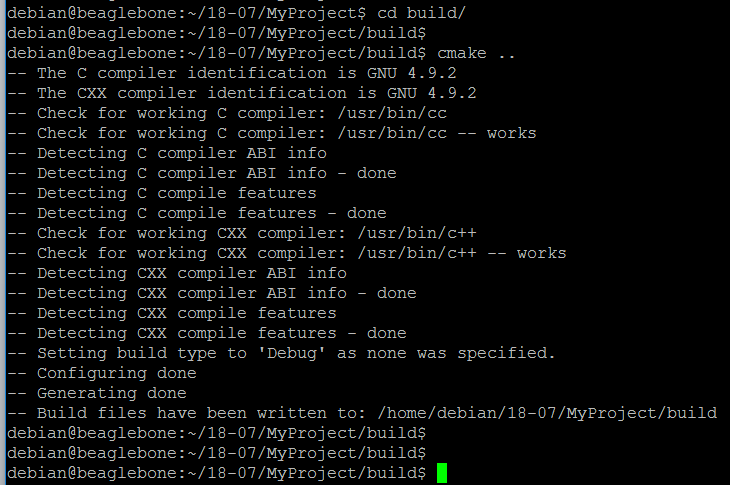
+ Open file main.cpp in “master” directory and update IP address and port information to permit it connects to submaster application

+ Open file main.cpp in “submaster” directory and update IP address setting to IP address of submaster itself.

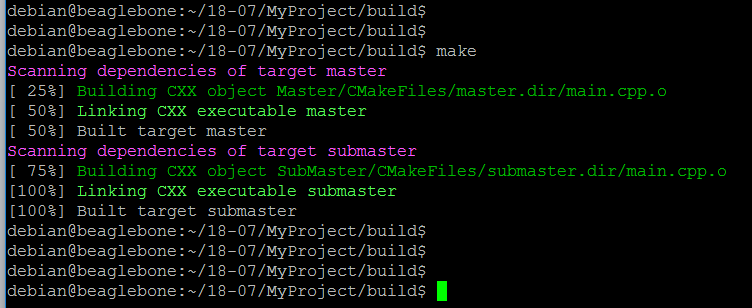
* Make sure we were in the root directory of project
* Make new directory for building progress, for example “build”



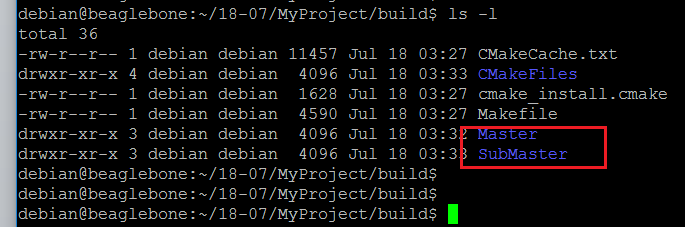
* Move to just-created “build” directory and run “cmake ..” command. If everything was right, we will have similar screen as below

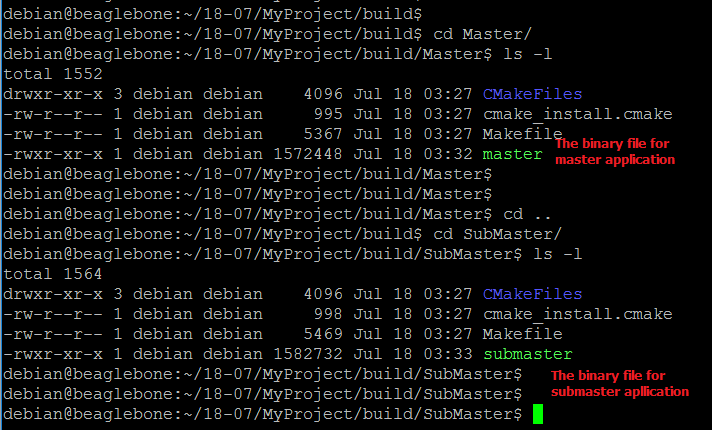


* Next, run “make” command and wait for a while (2 or 3 minutes) before compiling progress to finish
* Again, If everything was right, we will have similar screen as below



The compile will create 2 sub-directories under “build” directory. The one was used to store binary file for master application, the other was used to store binary file for submaster application.





Now, we have binary files for both master and submaster applications. You can test them with the simulator and real external serial devices.

JSON FORMAT OF SETTINGS

{

"version": "1.0.0",

"self\_id": 0,

"units": [

{

"unit\_type": "master",

"unit\_id": 0,

"unit\_ipaddr": "192.168.1.1"

},

{

"unit\_type": "slave",

"unit\_id": 1,

"unit\_ipaddr": "192.168.1.2"

},

.

.

.

{

"unit\_type": "submaster",

"unit\_id": 9,

"unit\_ipaddr": "192.168.1.10"

}

]

}

The application on each BBB should have its own settings.json file was on the same directory with binary file of the application

+ The "self\_id" element in JSON string used to identify the unit

+ The settings.json file for Master should have "self\_id" assigned to 0

+ The settings.json file for Submaster should have "self\_id" assigned to 9

+ The settings.json file for Slave should have "self\_id" assigned to 1 – 8

+ The "units" array stores all information of all units in details and this array should be the same on all BBBs