Compiling Boost Library for BeagleBone Black

by: Jitin Dhillon, Amandeep Dhothar, Indy Sekhon

This document guides the user through:

1. Downloading and configuring Boost for ARM processors
2. Using the library in C++ applications

TABLE OF CONTENTS

Guide Differences:
This guide is based on some online resources for the library. It combines the information from multiple guides into a easy to follow single guide. It also provides instructions on how to compile for ARM.

Formatting:

1. Commands starting with $ are Linux console commands on the host PC:
   $ echo "Hello PC world"
2. Commands starting with # are Linux console commands on the target board:
   # echo "Hello Target world"
3. Commands are case-sensitive in Linux

Description:

Boost is a set of libraries for the **C++** programming language. It is open sourced and peer reviewed and provides a lot of missing functionality. The libraries are documented nicely and the license allows inclusion in open and closed source projects.[1] The library provides support for tasks and structures such as linear algebra, pseudorandom number generation, multithreading image processing, regular expressions, and unit testing. It contains over eighty individual libraries which can be used individually.[2]

[1] http://stackoverflow.com/questions/125580/what-are-the-advantages-of-using-the-c-boost-libraries
[2] https://en.wikipedia.org/wiki/Boost_(C%2B%2B_libraries)

Install Boost Library

1. **Download Boost library on host**

   $ wget http://sourceforge.net/projects/boost/files/boost/1.59.0/boost_1_59_0.tar.gz/download

2. **Extract the downloaded file to /usr/local/**

   $ tar -xvf download -C /usr/local/

   Note: Most of the included libraries (excluding Chrono, Context, Filesystem, GraphParallel, IOStreams, Locale, MPI, ProgramOptions, Python, Regex, Serialization, Signals, System, Thread, Timer, and Wave) are able to utilized on the host and target without anymore configuration.

The library is used by including it in a header or c++ file.

#include <boost/somefile.hpp> or
#include "boost/somefile.hpp"

3. **Example of a library that does not need to be configured[3]:**

```
#include <boost/lambda/lambda.hpp>
#include <iostream>
#include <iterator>
#include <algorithm>

int main() {
       using namespace boost::lambda;
       typedef std::istream_iterator<int> in;

      std::for_each(
            in(std::cin), in(), std::cout << (_1 * 3) << " " );
}
```

To compile:

$ c++ -I path/to/boost_1_59_0 example.cpp -o example

[3] taken from Boost Getting Started on Unix Variants
http://www.boost.org/doc/libs/1_59_0/more/getting_started/unix-variants.html#link-your-program-to-a-boost-library

In this case:

$ arm-linux-gnueabi-g++ -I /usr/local/ example.cpp -o output

```
adhothar@ubuntu:~/cmpt433/work/project/boost_test$ cat boost_demo.cpp
#include <boost/lambda/lambda.hpp>
#include <iostream>
#include <iterator>
#include <algorithm>

int main()
{
    using namespace boost::lambda;
    typedef std::istream_iterator<int> in;

    std::for_each(
        in(std::cin), in(), std::cout << (_1 * 3) << " " );
}
adhothar@ubuntu:~/cmpt433/work/project/boost_test$ arm-linux-gnueabi-g++ -I /usr
/local/ boost_demo.cpp -o example
adhothar@ubuntu:~/cmpt433/work/project/boost_test$
```

The application can be run on the target with

 # echo 1 2 3 | ./example

Troubleshooting:

If you experience *"error while loading shared libraries: libstdc++.so.6: cannot open shared object file: No such file or directory"* you must statically link the standard c++ library such as :

```
arm-linux-gnueabi-g++ -I /usr/local/ -static-libstdc++ example.cpp -o
output
```

```
root@adhothar-beagle:/mnt/remote/myApps# echo 45 | ./example
./example: error while loading shared libraries: libstdc++.so.6: cannot open sha
red object file: No such file or directory
root@adhothar-beagle:/mnt/remote/myApps# echo 45 | ./example
135 root@adhothar-beagle:/mnt/remote/myApps#
```

Configuring and Building Libraries for ARM[4]

1. **Change directory to Boost folder and configure installation[5]**

   $ cd /usr/local/boost_1_59_0

   $ gedit project-config.jam

   and replace

   using gcc ;

   with

   using gcc : arm : arm-none-linux-gnueabi-g++ ;

2. **Build libraries**

   The libraries are installed through

   ***$** ./bootstrap.sh*

   Unless you have write permissions in /usr/local/ you must use

   ***$** ./bootstrap.sh **--prefix**=/usr/local/*

   Write permissions can be obtained from

   $ sudo chown -R $(whoami) /usr/local

   Finally

   $ ./bjam install toolset=gcc-arm

3. **Example**

```
#include <boost/regex.hpp>
#include <iostream>
#include <string>

int main() {
      std::string line;
      boost::regex pat( "^Subject: (Re: |Aw: )*(.*)" );
      while (std::cin) {
                  std::getline(std::cin, line);
                  boost::smatch matches;
                  if (boost::regex_match(line, matches, pat))
                              std::cout << matches[2] << std::endl;
      }
}
```

[4] Taken from http://www.boost.org/doc/libs/1_59_0/more/getting_started/unix-variants.html#prepare-to-use-a-boost-library-binary

[5] Taken from http://www.boost.org/build/doc/html/bbv2/tasks/crosscompile.html

Compile with

```
$ arm-linux-gnueabi-g++ -I /usr/local/ boost_demo_linked.cpp -o
demo \ -static-libstdc++ /usr/local/lib/libboost_regex.a
```

Now given a text file jayne.txt

```
To: George Shmidlap
From: Rita Marlowe
Subject: Will Success Spoil Rock Hunter?
---
See subject.
```

```
# demo < jayne.txt
```

will return

```
Will Success Spoil Rock Hunter?
```