Amir Mansouri Follow

Jan 13, 2017 · 15 min read

How to cross compile QT for Raspberry Pi 3 on Linux (Ubuntu) for Beginners!

Before we start the step by step of how to cross compile QT for Raspberry Pi on linux, let's discuss some of the basic topics.

Why Raspberry Pi?

Embedded devices are getting more and more mainstream. Every day we see new devices having a linux inside of them . From thermostat to your toaster oven, fridge, etc. The Raspberry Pi3 is affordable, reliable and fast.

Why QT?

I would love to talk for hours on how much I love QT. When it comes to writing GUI applications, there no tool more useful than QT. It's not just be400cause its typical GUI tools, QT team has taken C++ to a whole new level. You code in C++ but you don't have to deal with Visual Studio and everything that C++ sucks in. Also when it comes to running your GUI on Raspberry Pi, nothing works better than QT. I tried C# for weeks before landing on QT. I worked a lot with the serial communication capabilities of QT, sockets, and networking a lot more than GUI tools.

Why Cross Compiling ?

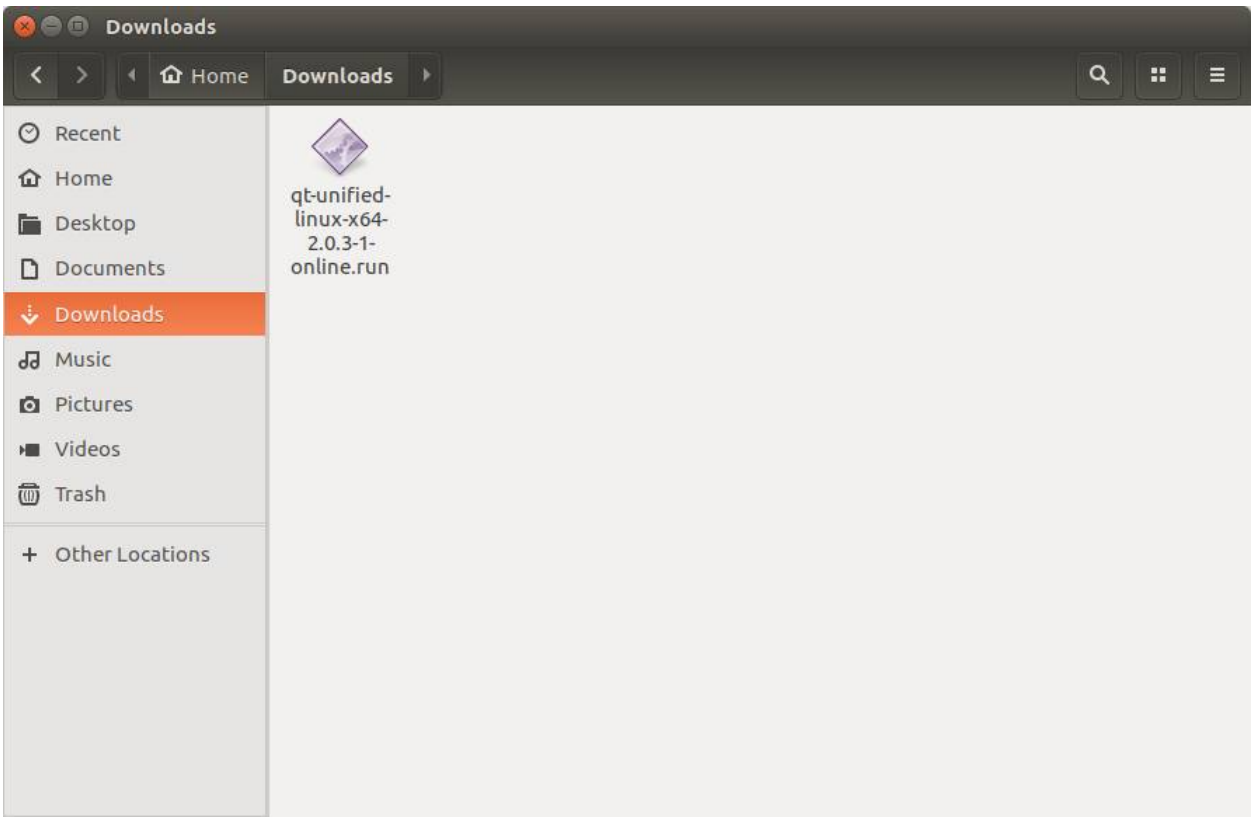
You might ask, why cross compiling, why can't I install QT creator

. . .

Step 2: Installing QT Creator on your Linux.

Now are we are going to install QT creator on our Ubuntu. Go to the QT creator website and download the installer. You can download the QT Creator Open Source if you are not doing any commercial development on QT.

Download then installer. You should have a file like this in your download folder:



Open up the terminal,

CD into the Download folder.

. . .

Step 3: Setting up the the toolchain and making the image ready.

Open up the terminal if it's not and make a folder and name it whatever you feel like. You can use my folder name if you are not creative 😜

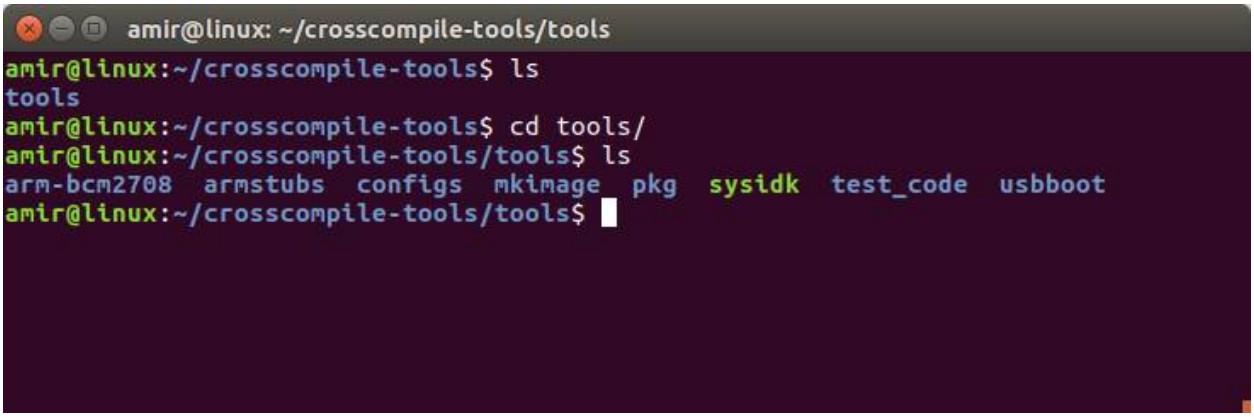
> *mkdir ~/crosscompile-tools*
>
> *cd ~/crosscompile-tools*
>
> *git clone https://github.com/raspberrypi/tools.git*

or if it didn't work

> *git clone https://github.com/amirhma/rpitools.git*
>
> *Cd tools*

Read noobie note 1 one more time if you are a noob.

Do a "ls" to see the content. It should look like this:



. . .

Setting Up and Building

We are going to fix all of the symbolic links in the Pi's root file system.

In terminal, make sure you are in the crosscompile-tools folder. You can alway use "pwd" command to check where you are. We are going to adjust symlinks to be relative and fix broken symbolic links.

> *sudo ./sysroot-relativelinks.py /mnt/rasp-pi-rootfs*

A successful results should takes a few seconds and turn a few pages of the terminal.

Initially I used the following script to fix the symbols.

> *./fixQualifiedLibraryPaths /mnt/rasp-pi-rootfs ~/crosscompile-tools/tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-**x64**/bin/arm-linux-gnueabihf-gcc*

I got an output that looked like the below:

Photo NEEDED

So if searching the term " XX" and looking for a solution to this output brought you to this pages, try the new scrip that I just included.

One last step before the main build. Let's export some variables to make our lives easier and save us some writing:

. . .

Setting Up Qt Creator

We are going to setup the QT creator and start developing program.

Make sure the following items are taken care of before proceeding:

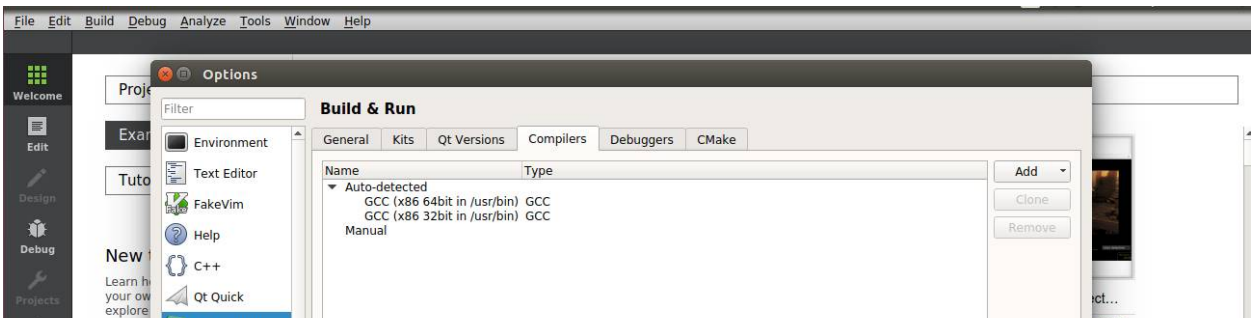
1. The custom image is burnt into the MicroSD card and placed into the Pi.
2. Your Pi 3 is connected to Internet using an Ethernet cable.
3. The raspbian.img is mounted on your linux computer already.

I ephasize you need that custom image of yours both burnt and installed on the Pi and already mounted on your cross compiler Linux. If you don't know how to mount the image, scroll up and find the instruction.

Open up the Qt Creator and go to the to

Tools > Options > Build and Run

then go to the Compiler tab



Raspberry Pi Technology Embedded Systems Qt