Integrating qt and Rpi3.

To begin development of applications using Qt and make it compatible to deploy its application on Rpi, we need to follow the given procedure.

Some prerequisite required are:
First, we should have rpi with us and Os being installed on it, here Rpi 3 is used, There are many OS but we start with Raspbian for desktop.
Secondly, Make sure you have Qt installed in your system and working properly.

Open Qt and go to tools from toolbar - open options given at last,
You will see a pop up window in that move on to kit,
We need to make a kit named Rpikit, on which we can deploy our developed applications to be used in Rpi.
So for that lets begin:

Steps:
1. On Rpi - first of all enable ssh on Rpi and connect it with Wi-Fi. To connect with ssh go to terminal and run following commands,

sudo raspi-config

Goto interfacing options enable ssh.
Note: make sure you connect your Rpi with the same Wifi with which your system is connected.

Then run below commands to update rpi,
sudo apt update
sudo apt upgrade
reboot

2. On Rpi - Edit sources list in */etc/apt/sources.list* with use of your favorite editor (nano / vi) and uncomment the **deb-src** line:

sudo nano /etc/apt/sources.list
sudo apt-get update

3. On Rpi - Update your system and install required libraries:

sudo apt-get build-dep qt5-qmake
sudo apt-get build-dep libqt5webengine-data
sudo apt-get build-dep qt4-x11
sudo apt-get build-dep libqt5gui5

sudo apt-get install libboost1.58-all-dev libudev-dev libinput-dev libts-dev libmtdev-dev libjpeg-dev libfontconfig1-dev
sudo apt-get install libssl-dev libdbus-1-dev libglib2.0-dev libxkbcommon-dev libegl1-mesa-dev libgbm-dev libgles2-mesa-dev mesa-common-dev
sudo apt-get install libasound2-dev libpulse-dev gstreamer1.0-omx libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev gstreamer1.0-alsa
sudo apt-get install libvpx-dev libsnappy-dev libnss3-dev
sudo apt-get install libsrtp0-dev
sudo apt-get install "^libxcb.*"
sudo apt-get install libfreetype6-dev libicu-dev libsqlite3-dev libxslt1-dev libavcodec-dev libavformat-dev libswscale-dev
sudo apt-get install libgstreamer0.10-dev gstreamer-tools libraspberrypi-dev libx11-dev libglib2.0-dev
sudo apt-get install freetds-dev libsqlite0-dev libpq-dev libiodbc2-dev firebird-dev libjpeg9-dev libgst-dev libxext-dev libxcb1 libxcb1-dev libx11-xcb1
sudo apt-get install libxcb-sync1 libxcb-sync-dev libxcb-render-util0 libxcb-render-util0-dev libxcb-xfixes0-dev libxrender-dev libxcb-shape0-dev libxcb-randr0-dev
sudo apt-get install libxcb-glx0-dev libxi-dev libdrm-dev libssl-dev libxcb-xinerama0 libxcb-xinerama0-dev
sudo apt-get install libatspi-dev libssl-dev libxcursor-dev libxcomposite-dev libxdamage-dev libfontconfig1-dev
sudo apt-get install libxss-dev libxtst-dev libpci-dev libcap-dev libsrtp0-dev libxrandr-dev libnss3-dev libdirectfb-dev libaudio-dev

4. On Rpi - make the following directory on rpi system:

sudo mkdir /usr/local/qt5pi
sudo chown pi:pi /usr/local/qt5pi

5. On PC - Then on the host Pc make directories and install the tools from given link, this contains debugger and compiler,

mkdir ~/raspi
cd ~/raspi
git clone https://github.com/raspberrypi/tools

6. On PC - Make directory sysroot inside raspi made in previous step, inside which make two directories usr and opt, as shown below:

mkdir sysroot sysroot/usr sysroot/opt

Now, run following commands to sync Rpi and Host Pc.

rsync -avz pi@raspberrypi.local:/lib sysroot

rsync -avz pi@raspberrypi.local:/usr/include sysroot/usr
rsync -avz pi@raspberrypi.local:/usr/lib sysroot/usr
rsync -avz pi@raspberrypi.local:/opt/vc sysroot/opt

7. On PC - Get the given file from link and then execute it, as shown below.

wget
https://raw.githubusercontent.com/Kukkimonsuta/rpi-buildqt/master/scripts/utils/sysroot-relativelinks.py
chmod +x sysroot-relativelinks.py
./sysroot-relativelinks.py sysroot

8. On PC - Now, It's time to install some big stuff, there are two ways either follow the below commands, if they work fine, else we may download the file directly from link then extract it at desired location, that is in raspi directory.

Note: make sure you are working with Qt version 5.12.9, any other version requires the link to be changed.

wget
https://download.qt.io/official_releases/qt/5.12/5.12.9/single/qt-everywhere-src-5.12.9.tar.xz
tar -xvf qt-everywhere-src-5.12.9.tar.xz
rm -d qt-everywhere-src-5.12.9.tar.xz
cd qt-everywhere-src-5.12.9

9. On PC - Once you have downloaded the required directory, run the below command. Below command will change if you use another version of Rpi; it's only for Rpi3.

./configure -release -opengl es2 -device *linux-rasp-pi3-g*++ -device-option CROSS_COMPILE=~/raspi/tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-x64/bin/arm-linux-gnueabihf- -sysroot ~/raspi/sysroot -opensource -confirm-license -make libs -prefix /usr/local/qt5pi -extprefix ~/raspi/qt5pi -hostprefix ~/raspi/qt5 -v -no-use-gold-linker

10. On PC - Run the “make” command, we may also run “make -j4‘, where 4 is no of cores used we may change it accordingly. It's most time consuming and may take an hour.

make -j4
make install

11. On PC - Deploy Qt to the device. We simply sync everything from *~/raspi/qt5pi* to the prefix we configured above.

cd ..

rsync -avz qt5pi pi@raspberrypi.local:/usr/local

12. On Rpi - Update the device to let the linker find the Qt libs:

```
echo /usr/local/qt5pi/lib | sudo tee /etc/ld.so.conf.d/qt5pi.conf
sudo ldconfig
```

13. On Rpi - Run these commands to set egl libraries and fix its error, if command doesn't work then use f with s.

```
sudo mv /usr/lib/arm-linux-gnueabihf/libEGL.so.1.0.0
/usr/lib/arm-linux-gnueabihf/libEGL.so.1.0.0_backup
sudo mv /usr/lib/arm-linux-gnueabihf/libGLESv2.so.2.0.0
/usr/lib/arm-linux-gnueabihf/libGLESv2.so.2.0.0_backup
sudo ln -s /opt/vc/lib/libEGL.so /opt/vc/lib/libEGL.so.1
sudo ln -s /opt/vc/lib/libGLESv2.so /opt/vc/lib/libGLESv2.so.2
sudo ln -s /opt/vc/lib/libEGL.so /usr/lib/arm-linux-gnueabihf/libEGL.so.1.0.0
sudo ln -s /opt/vc/lib/libGLESv2.so /usr/lib/arm-linux-gnueabihf/libGLESv2.so.2.0.0
sudo ln -s /opt/vc/lib/libbrcmEGL.so /opt/vc/lib/libEGL.so
sudo ln -s /opt/vc/lib/libbrcmGLESv2.so /opt/vc/lib/libGLESv2.so

sudo ln -s libbrcmEGL.so libEGL.so.1
sudo ln -s libbrcmGLESv2.so libGLESv2.so.2
echo 'export LD_LIBRARY_PATH=/opt/vc/lib' >> ~/.bashrc
sudo ldconfig
```

14. Set up the required things for the Rpi kit.
To make any of these click add by going to respective places.

Make device,
Goto tools -> options -> devices
Select generic linux devices in devices
Provide Ip address of Rpi and username

Make compiler,
Goto tools -> options -> kits -> compiler
Provide location of compiler
Hint:
/home/rishabh/raspi/tools/arm-bcm2708/gcc-linaro-arm-linux-gnueabihf-raspbian-x64/bin/arm-linux-gnueabihf-gcc

Make debugger
Goto tools -> options -> kits -> debugger
Provide location of debugger

Hint:
/home/rishabh/raspi/tools/arm-bcm2708/arm-rpi-4.9.3-linux-gnueabihf/bin/arm-linux-gnueabihf-gdb

Make qt version
Goto tools -> options -> kits -> qt version
Provide location of qmake
Hint: /home/rishabh/raspi/qt5/bin/qmake

Location of sysroot

15. Develop a RPIKIT on pc and set the developed devices, compiler, debugger, Qt Version and sysroot location.

Now let's begin with an example and see how to deploy on rpi.

Open your project,
Make sure you have selected the release not debug while clicking the play/run button from left bottom,
Change .pro file little by changing target at last to
else: unix:!android: target.path = /home/pi/$${TARGET}

Run the application,
You will see your application in home/pi of your Rpi.
Go to it and run it.
If you face any problem run following commands on Rpi and then retry,

export QT_QPA_PLATFORMTHEME=qt5ct
export DISPLAY=:0.0
export XAUTHORITY=/home/pi/.xauthority
export XDG_SESsion_TYPE=x11

16. If we want our file to run automatically on Rpi we need to set some variables and its value in build section on left side,

Variable		value
QT_QPA_PLATFORMTHEME	=	qt5ct
DISPLAY	=	:0.0
XAUTHORITY	=	/home/pi/.xauthority
XDG_SESsion_TYPE	=	x11

Now if you follow the 15 step it will run the application also.