**< 데비안 Qt 설명서 >**

**- 차례 -**

**1. Qt5.3.2 크로스 컴파일**

**2. 호스트 터미널 컴파일**

**3. VNC Qt Creator 크로스 컴파일**

**1) Qt5.3.2 크로스 컴파일**

Qt 크로스 컴파일을 하기 위해 데비안 컴파일러에 선행으로 추가해야 할 라이브러리가 있다.

**Zlib 추가:** (설치한 데비안 컴파일러의 경로를 알맞게 넣어준다)

$ wget http://prdownloads.sourceforge.net/libpng/zlib-1.2.11.tar.gz

$ tar zxvf zlib-1.2.11.tar.gz

$ cd zlib-1.2.11

$ mkdir release //결과 파일들을 복사할 디렉토리 생성

$ CC=~/var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-gcc ./configure --prefix=./release

$ make

$ make install

$ cd release

(컴파일된 결과물들을 bsp의 컴파일러 라이브러리 경로로 복사한다.)

$ cp -rf \* ~/var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/arm-linux-gnueabihf/

**tslib추가:** (zlib와 마찬가지로 경로에 주의)

$ git clone https://github.com/kergoth/tslib.git

$ cd tslib

$ mkdir build

$ ./autogen.sh

$ ./configure --prefix=$PWD/build/ --host=arm-linux-gnueabihf -enable-static=yes CC=~/var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/bin/arm-linux-gnueabihf-gcc

$ make

$ make install

$ cd build

$ sudo cp -rf bin/ include/ lib/ share/ ~/var\_som\_mx6\_debian/rootfs/usr/.

$ sudo cp etc/ts.conf ~/var\_som\_mx6\_debian/rootfs/etc/.

**Qt 크로스 컴파일 하기:**

(참고 : <http://git.geminit.co.kr/huvitz-vs-debian/Qt_jessie_5_3_2/wikis/home> )

먼저 컴파일 환경설정을 위해 데비안 라이브러리를 호스트쪽으로 복사한다.

$ sudo cp -ar ~/var\_som\_mx6\_debian/rootfs/lib/arm-linux-gnueabihf /lib/.

Gitlab에서 다운받는 방법 또는 디렉토리를 만들고 따로 첨부한 파일들을 복사한다.

(컴파일 설정 파일과 qt5.3.2 라이브러리 소스이다.)

$ git clone http://git.geminit.co.kr/huvitz-vs-debian/Qt\_jessie\_5\_3\_2.git

$ cd Qt\_jessie\_5\_3\_2

혹은 직접 소스를 받아도 된다. (첨부파일도 같은 디렉토리에 넣는다.)

$ wget https://download.qt.io/archive/qt/5.3/5.3.2/single/qt-everywhere-opensource-src-5.3.2.tar.gz

$ tar zxvf qt-everywhere-opensource-src-5.3.2.tar.gz

$ vi linux\_device\_pre.conf

(18 line: --sysroot 의 경로를 bsp의 rootfs 경로로 수정한다.)

데비안 환경에 맞게 설정한 파일들을 원본 Qt 소스에 덮어씌운다.

$ cp linux\_device\_pre.conf qt-everywhere-opensource-src-5.3.2/qtbase/mkspecs/devices/common/

$ cp qmake.conf qt-everywhere-opensource-src-5.3.2/qtbase/mkspecs/devices/linux-imx6-g++/

$ mkdir build

$ cd build

qmake 설정 ※(CROSS\_COMPILE 과 -sysroot 에서 경로를 알맞게 수정합니다. )

$ ../qt-everywhere-opensource-src-5.3.2/configure -v -tslib -opensource -confirm-license -no-pch -no-xcb -no-openssl -no-opengl -opengl es2 -no-largefile -qt-zlib -qt-libpng -qt-libjpeg -no-cups -gui -no-sse2 -no-iconv -release -make libs -device imx6 -device-option CROSS\_COMPILE=/home/user/var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/bin/arm-linux-gnueabihf- -sysroot /home/user/var\_som\_mx6\_debian/rootfs -no-gcc-sysroot -prefix /opt/qt5\_3\_2 -make examples -nomake tools -nomake tests

( Prefix는 컴파일된 qt라이브러리가 생성될 경로이다. )

$ make -j4

$ sudo make install

**2) 호스트 터미널 컴파일**

먼저 컴파일된 Qt5.3.2의 qmake 사용을 위해 PATH를 등록해준다.

$ vi ~/.bashrc

맨 밑줄에 추가: export PATH=/opt/qt5\_3\_2/bin/:$PATH

$ source ~/.bashrc

$ qmake -version //버전확인.

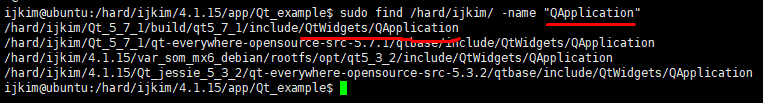
main.cpp 첨부.

또는 Gitlab 참고( <http://git.geminit.co.kr/huvitz-vs-debian/Qt_example/wikis/home> )

$ qmake –project //해당 디렉토리 이름으로 .pro 파일이 생성된다.

( qmake는 일종의 Makefile을 만들어주는 도구이다. )

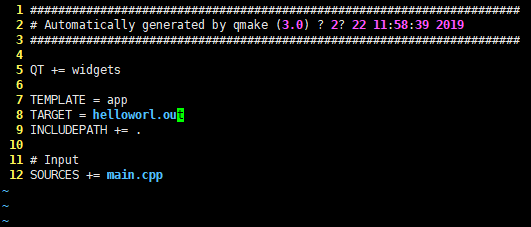
생성된 .pro 파일 내부에 예제소스에서 사용하는 모듈을 직접 추가해야 한다.



소스에서 인클루드 하고있는 QApplication는 QtWidgets 디렉토리에 있다.

‘Qt’는 빼고 소문자로 추가한다. QT += widgets

실행 파일 이름도 설정해준다 => helloworld.out



$ qmake //Makefile 생성됨.

$ make

결과물(helloworld.out)을 보드에 복사 후 실행한다.

# ./helloworld.out –platform linuxfb –plugin tslib

**3) VNC Qt Creator 크로스 컴파일**

**x11vnc 서버 설치:** (호스트에서 진행)

$ sudo apt-get install x11vnc

$ x11vnc -storepasswd //원격 접속 암호 설정.

$ vi /lib/systemd/system/x11vnc.service // vnc설정 편집.

------------------------------------------------------------------------------

[Unit]

Description=Start x11vnc at startup.

After=multi-user.target

[Service]

Type=simple

ExecStart=/usr/bin/x11vnc -auth guess -forever -loop -noxdamage -repeat -rfbauth /root/.vnc/passwd -rfbport 5900 -shared

ExecStop=/usr/bin/killall x11vnc

[Install]

WantedBy=multi-user.target

------------------------------------------------------------------------------

\*설정 불러오기

$ systemctl daemon-reload

\*부팅 시 자동 실행 하기

$ systemctl enable x11vnc.service

\*서비스 시작

$systemctl start x11vnc.service

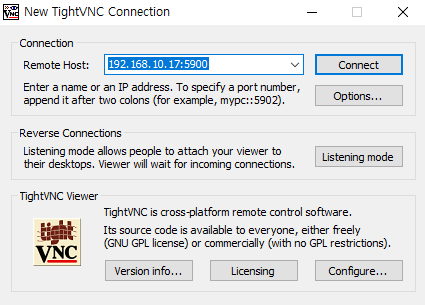
**vnc viewer 설치:**

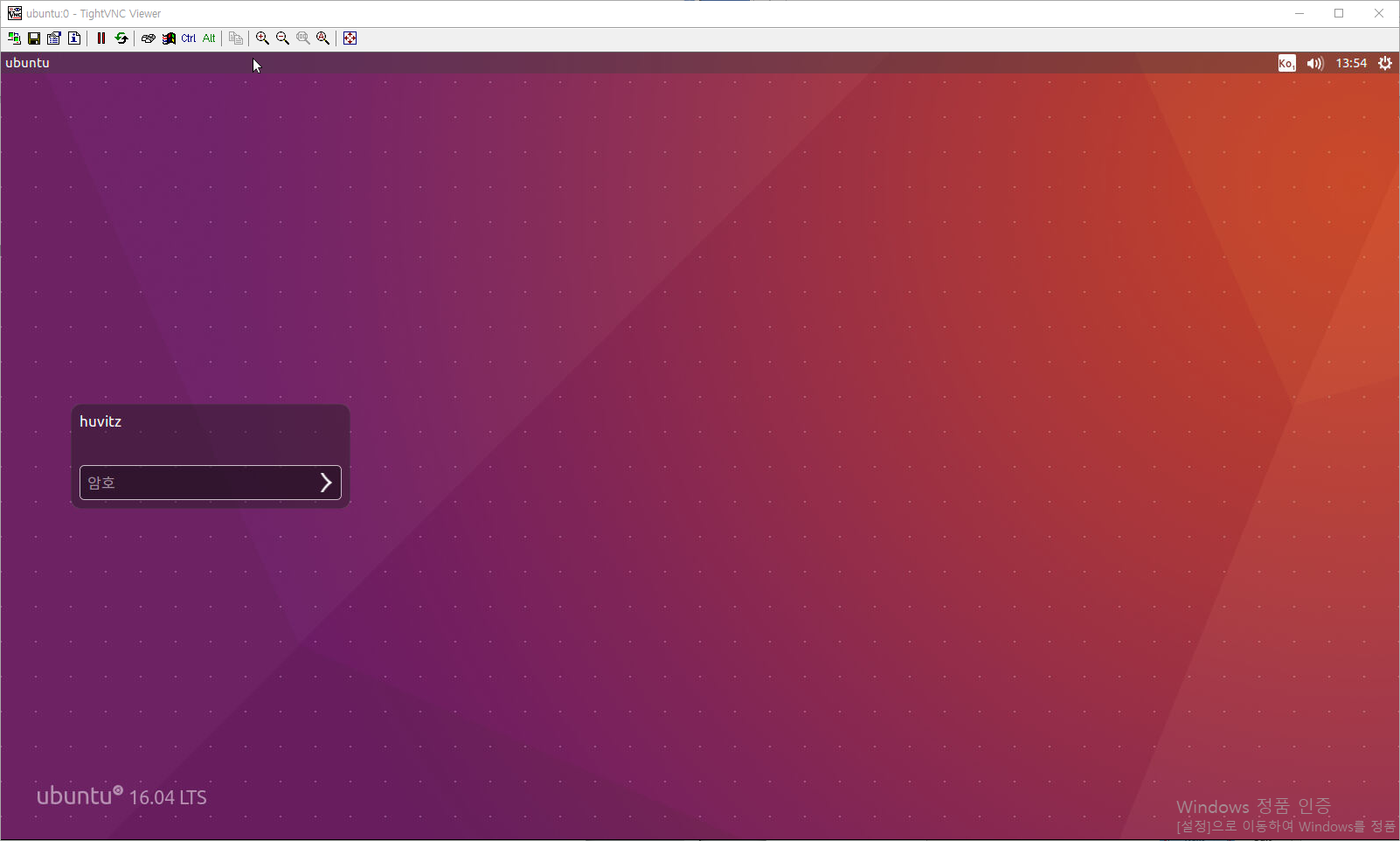
클라이언트쪽(보통 윈도우환경)에 vnc viewer를 설치한다. (아무거나 설치해도 무방하다.)

tightVNC 설치: <https://www.tightvnc.com/download.php>

환경에 맞는 버전으로 설치한다. => viewer만 선택하여 설치한다.

( 호스트 IP : 5900 ) 입력 -> Connect





**Qt Creator 설치:**

$ sudo apt-get update

$ sudo apt-get install build-essential

( ※만일 qt4 버전이 이미 설치되어있다면 삭제한다. )

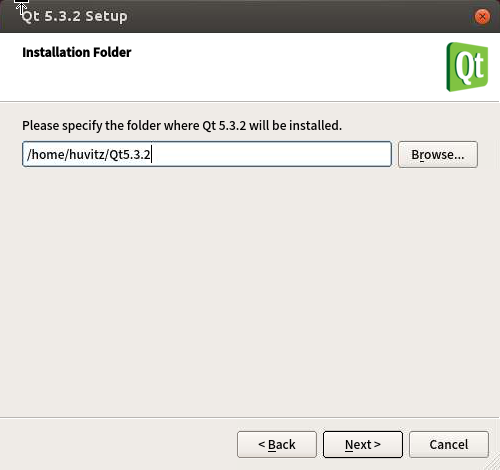
$ sudo apt-get purge --auto-remove libqt4-dev

$ wget https://download.qt.io/archive/qt/5.3/5.3.2/qt-opensource-linux-x64-5.3.2.run

$ chmod +x qt-opensource-linux-x64-5.3.2.run

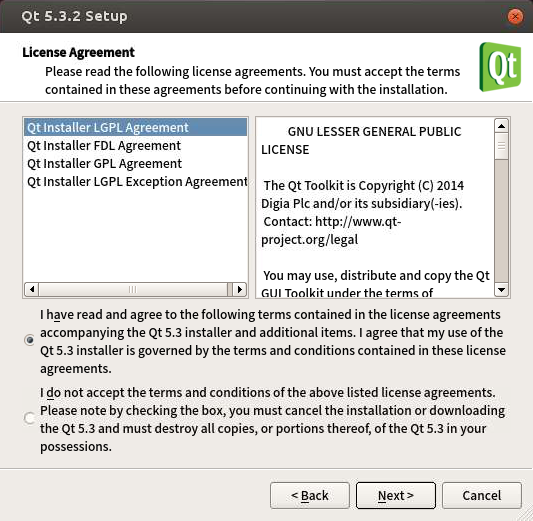
$ ./qt-opensource-linux-x64-5.3.2.run //installer 실행.

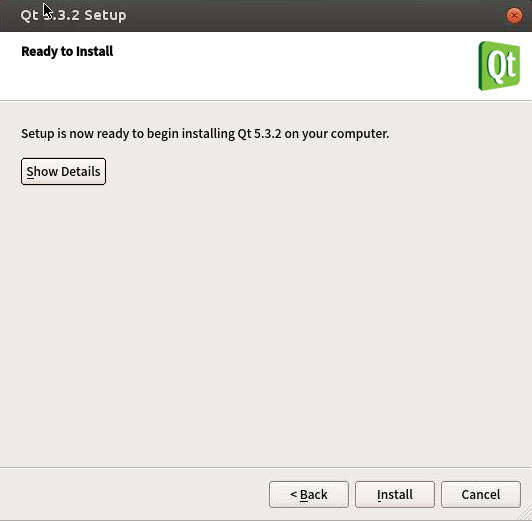






**옵션 선택**

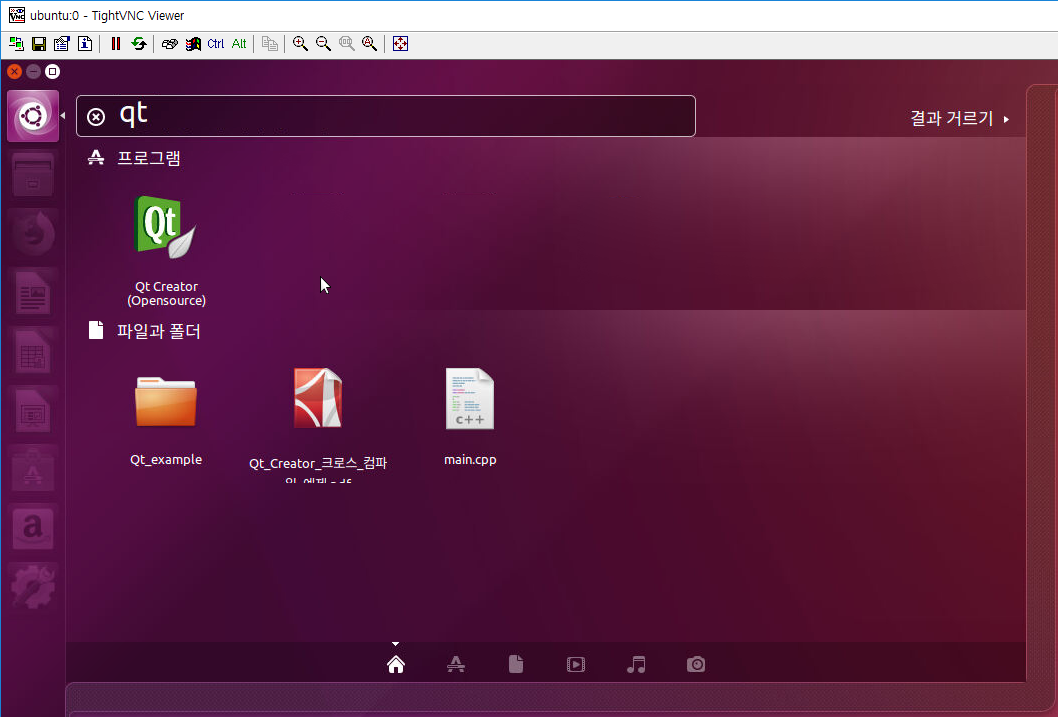




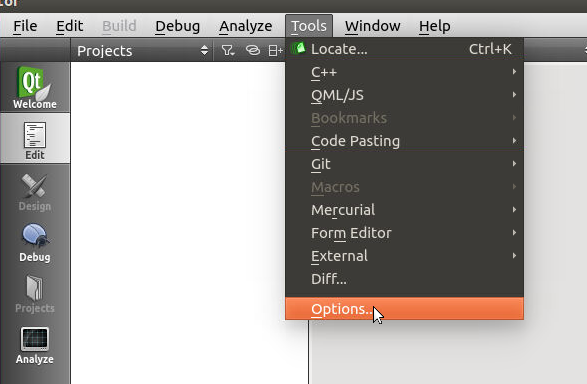


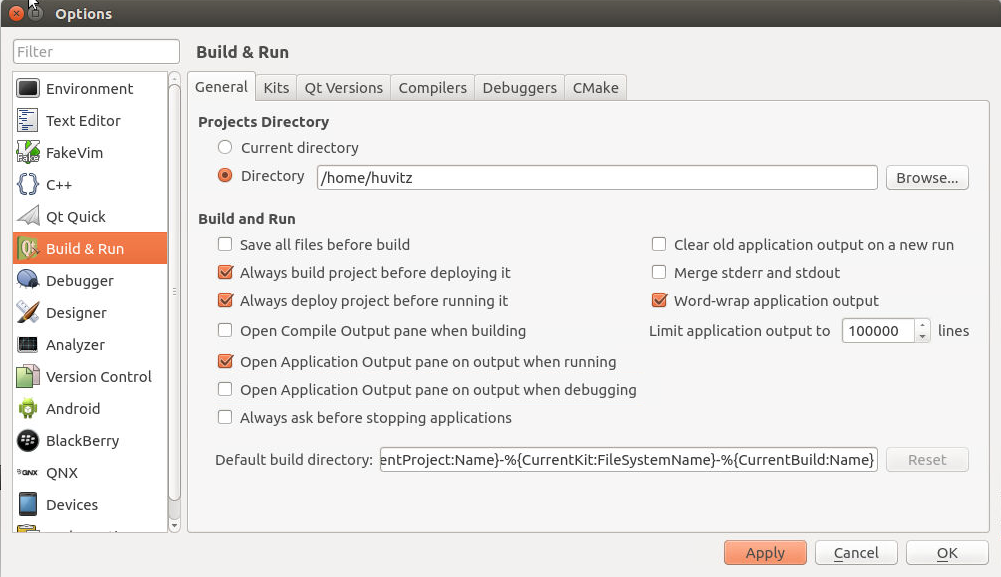
**체크 해제**

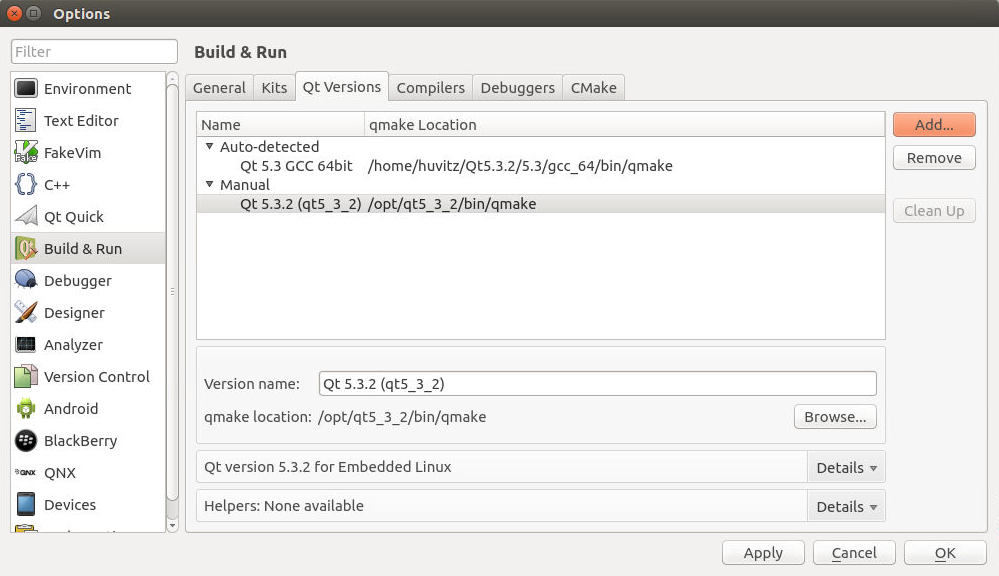
< Qt 실행 >



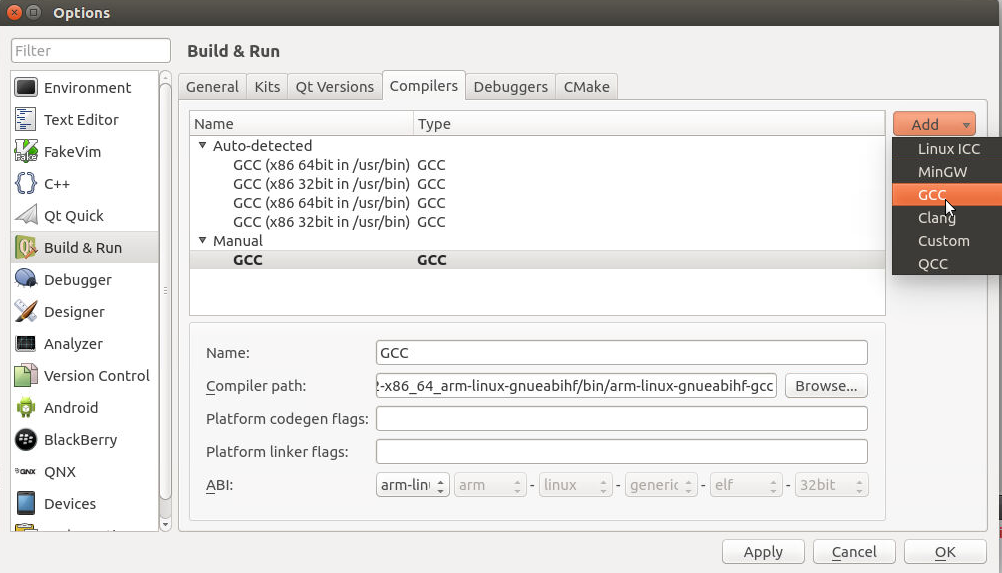
< Build tool kit 설정 >



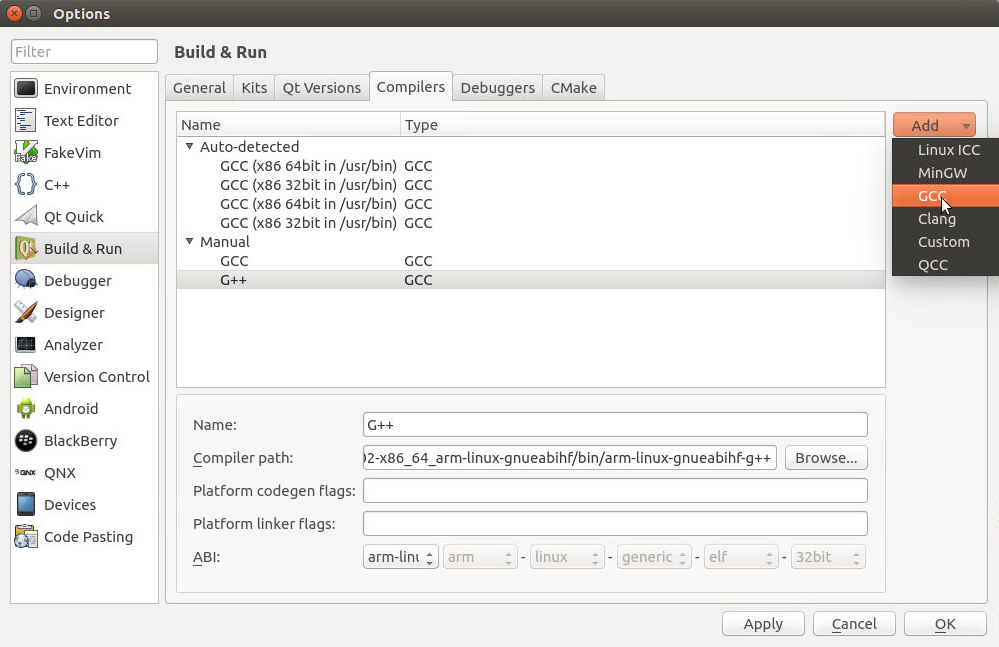




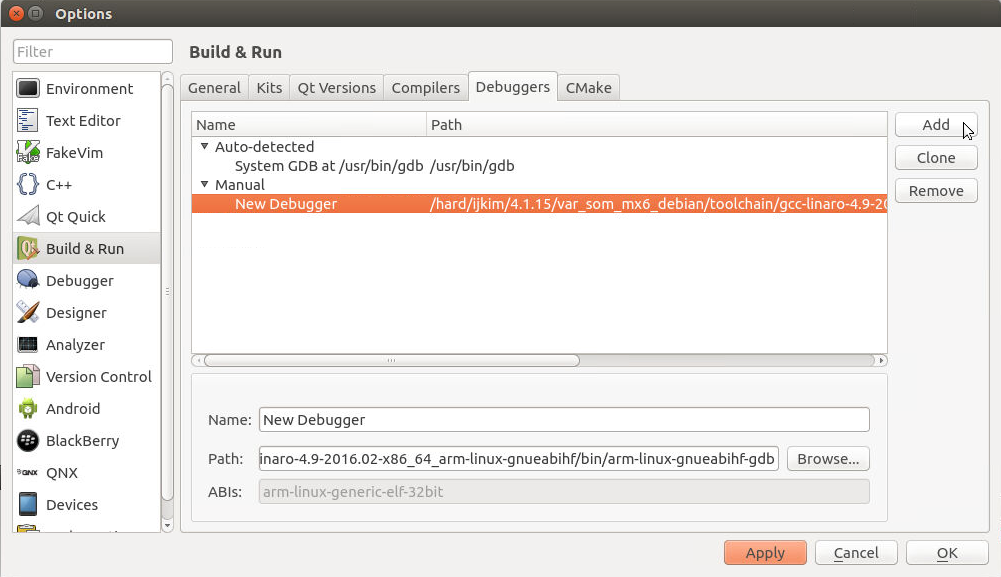
/opt/qt5\_3\_2/bin/qmake



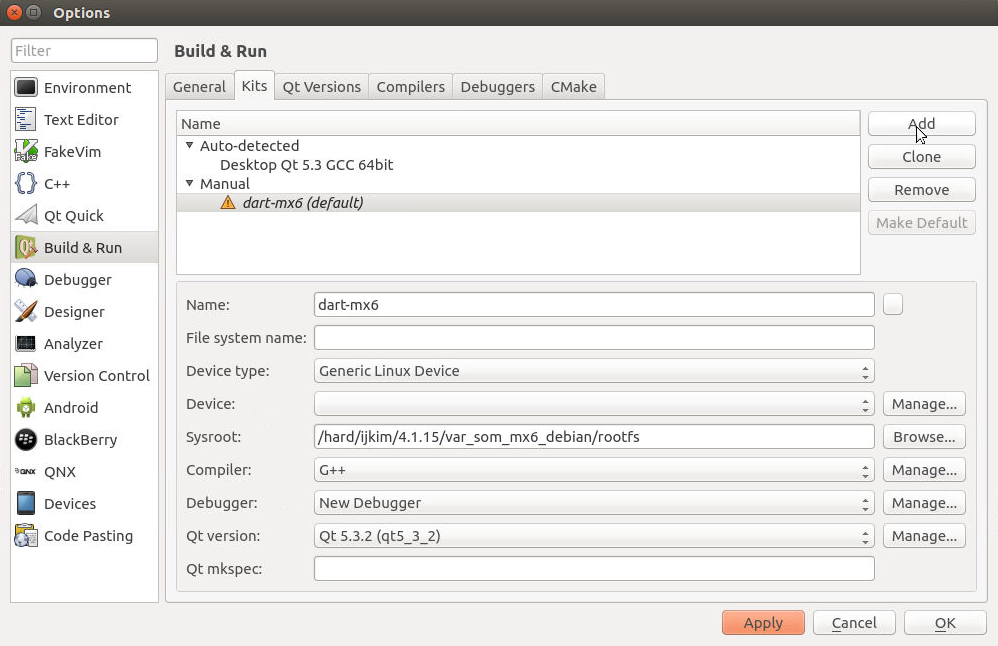
var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/  
bin/arm-linux-gnueabihf-gcc



var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/  
bin/arm-linux-gnueabihf-gcc



var\_som\_mx6\_debian/toolchain/gcc-linaro-4.9-2016.02-x86\_64\_arm-linux-gnueabihf/  
bin/arm-linux-gnueabihf-gdb



**컴파일러**

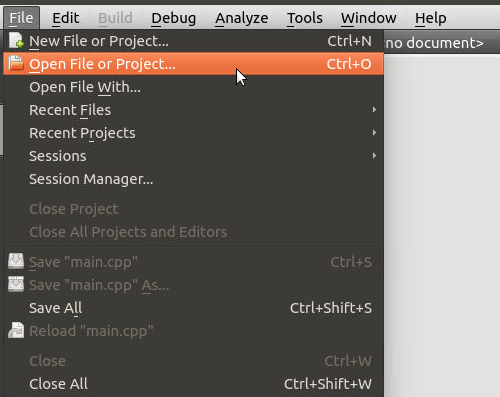
**디버거**

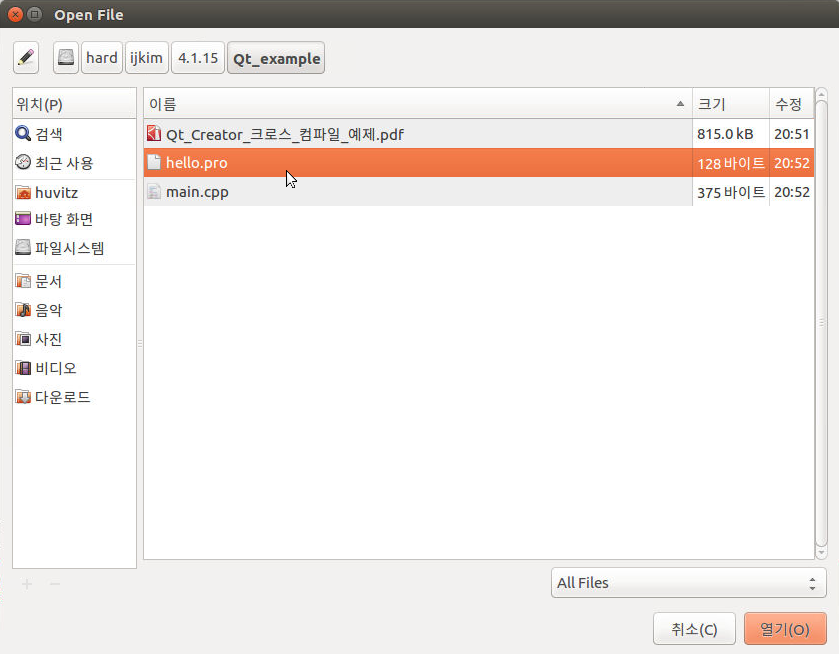
**qmake**

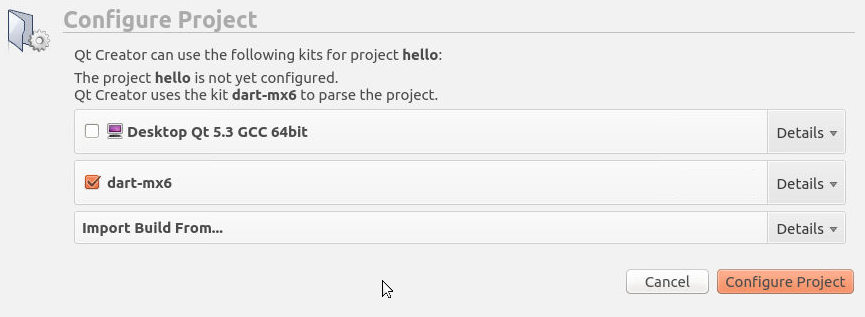
**rootfs 경로**

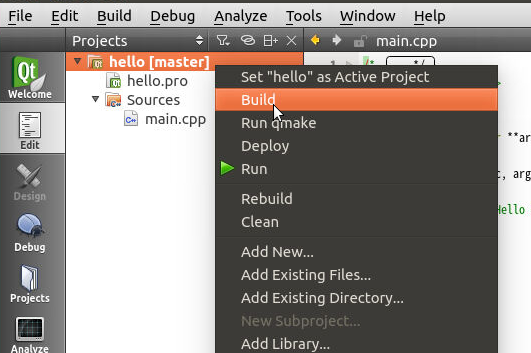
**Kit 이름**

* Open project





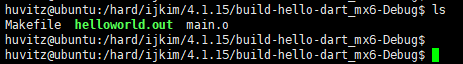




빌드 클릭.

build-hello-dart\_mx6-Debug 이름의 디렉토리가 생성됩니다.

helloworld.out 실행파일 보드로 복사하여 실행.



실행

# ./helloworld.out –platform linuxfb –plugin tslib