1. 移植 tslib

tslib 是电阻式触摸屏用于校准的一个开源程序软件库，能够为触摸屏驱动获得的采样提供诸如滤波、去抖、校准等功能，通常作为触摸屏驱动的适配层，为上层的应用提供了一个统一的接口。这里先编译安装 tslib，这样在后面编译 Qt 的时候才能打包编译进去。

1) 安装 autoconf、automake、libtool 包

```
apt-get install autoconf automake libtool libsysfs-dev g++
```

2) 下载文件 https://github.com/kergoth/tslib，解压并编译：

```
tar -xvf tslib-1.4.tar.bz2
```

```
gedit ./tests/ts_calibrate.c
```

go to line227 &&line 229,

```
    cal_fd = open (calfile, O_CREAT | O_RDWR,0777);
```

```
    cal_fd = open ("/etc/pointercal", O_CREAT | O_RDWR,0777);
```

```
./tslib.sh 2>&1 | tee makeLog.txt
```

查看一下生成的文件是否为 ARM 运行文件：file bin/ts_test

编译完成后在-prefix 指定的主机目录下生成可用文件，修改 tslib/etc/ts.conf，去掉#号和空格：# module_raw input -> module_raw input，然后将产生的整个 tslib 文件夹内容下载到开发板的对应路径下（/opt），并修改开发板环境变量。

4) 加载触摸屏驱动模块：modprobe ti_am335x_tsc.ko

允许校准程序进行校准：./ts_calibrate，校准后在 tslib 的 etc 目录下生成校准文件。

如果触摸不准，需要删除之前的校准文件，重新校准：rm -rf /etc/pointercal*

运行测试：　　./ts_print

2. 移植 qt

1)下载 qt-everywhere-opensource 和 qt-creator：

http://download.qt.io/archive/qt/

http://download.qt.io/new_archive/qt/

http://download.qt.io/official_releases/qt/

解压： tar -xvJf qt-everywhere-opensource-src-5.9.9.tar.xz -C /opt/

qt-everywhere-opensource-src-5.9.9

进入目录： cd qt-everywhere-opensource-src-5.9.9

2)安装工具包：apt-get install g++-multilib libx11-dev libxext-dev libxtst-dev zlib1g-dev

lib32ncurses5 lib32z1 libpng-dev autoconf automake libtool lsb-core libqt4-dev

lib32stdc++6 libgl1-mesa-dev libXrender-dev libjpeg-dev

3) 复制脚本和配置文件，执行编译：

cp qmake.conf

/opt/qt-everywhere-opensource-src-5.9.9/qtbase/mkspecs/linux-arm-gnueabi-g++

cp qt-arm.sh qt-desktop.sh /opt/qt-everywhere-opensource-src-5.9.9/

查看配置帮助，将结果输出到文本 helpdoc.txt 中： ./configure -help > helpdoc.txt

make clean

make confclean

./qt-arm.sh 2>&1 | tee confLog.txt

make -j4 2>&1 | tee makeLog.txt

make install -j4 2>&1 | tee installLog.txt

4)新建 fonts 文件夹并复制字体， QT 程序才能显示中文:

```
mkdir /opt/qt-arm-5.9.9/lib/fonts
```

```
cp STKAITI.TTF /opt/qt-arm-5.9.9/lib/fonts/
```

5)将编译好的/opt/qt-arm-5.9.9/bin 下 qmake 重命名为 arm-qmake,然后拷贝到虚拟机/usr/bin 下。

测试环境变量：qmake -v　　arm-qmake -v

此时在代码文件夹可以使用 make clean，make confclean，make 进行编译(需要 Makefile 文件)。

6)将编译好的/opt/qt-arm-5.9.9 中的所有文件打包，并在 BeagleBone 开发板中新建文件夹解压:

```
cd /opt/qt-arm-5.9.9/
```

```
tar -cvf qt5.tar ./*
```

```
# mkdir /opt/qt5
```

```
# tar -xvf qt5.tar -C /opt/qt5
```

7)在目标板/etc 下的 profile 文件中添加环境变量 gedit /etc/profile

使环境变量生效： source /etc/profile

8)将编译器下 usr/lib，QT 缺失的库拷贝到目标文件系统对应目录下，加载触摸驱动，运行 QT 程序。

```
cp /mnt/libicuuc.so.62 /usr/lib
```

3. 安装 Qtcreator

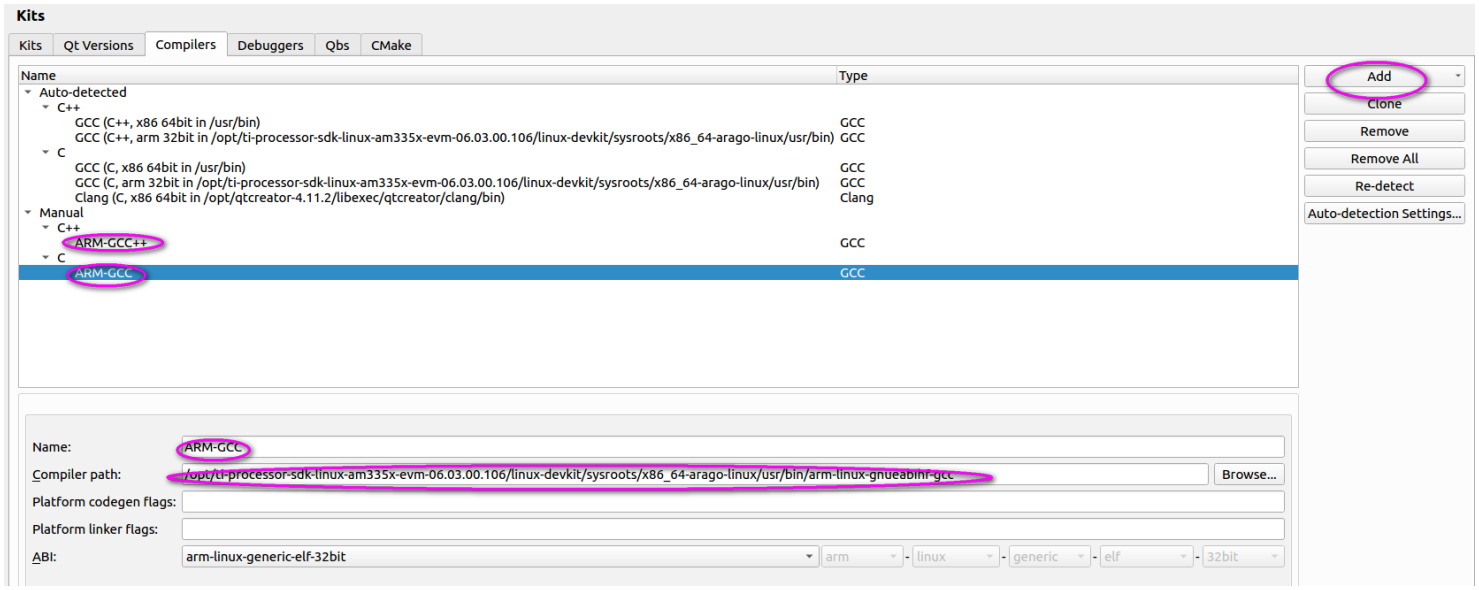
Qtcreator 安装：./qt-creator-opensource-linux-x86_64-4.11.2.run

安装目录选择：/opt/qtcreator-4.11.2

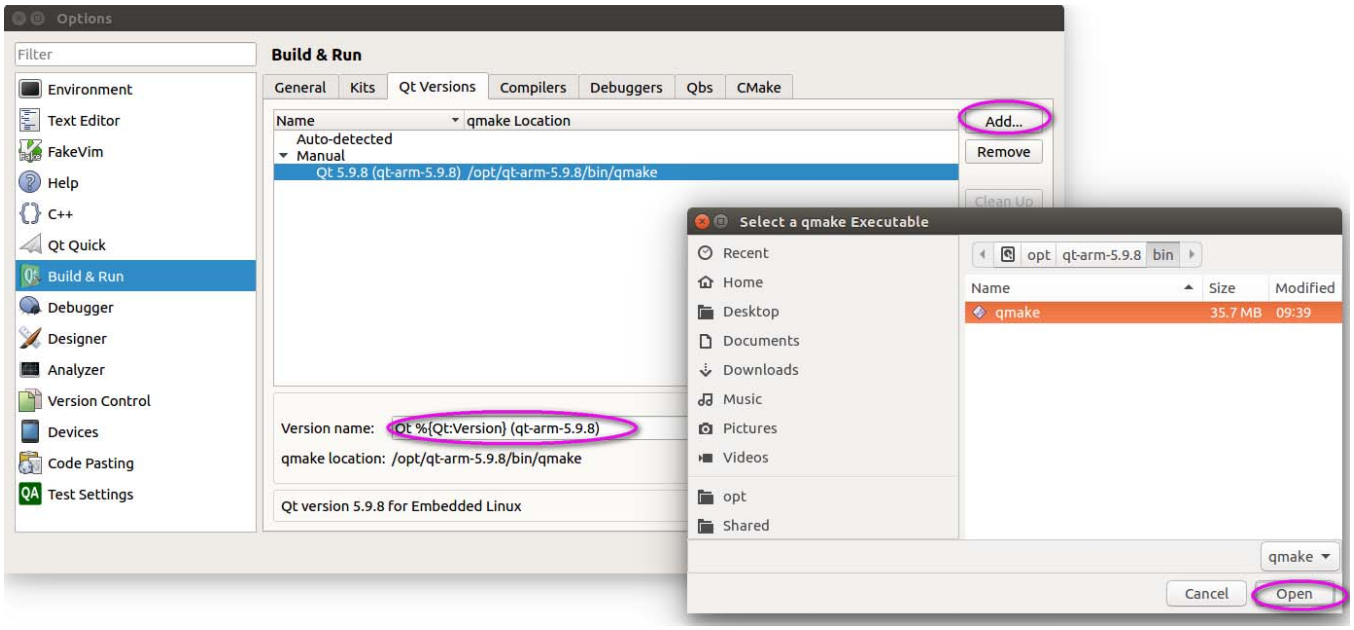
安装完成后，/opt/qtcreator-4.11.2/bin/qtcreator 即是可执行文件。

环境配置，在 tools->options 里面进行设置：

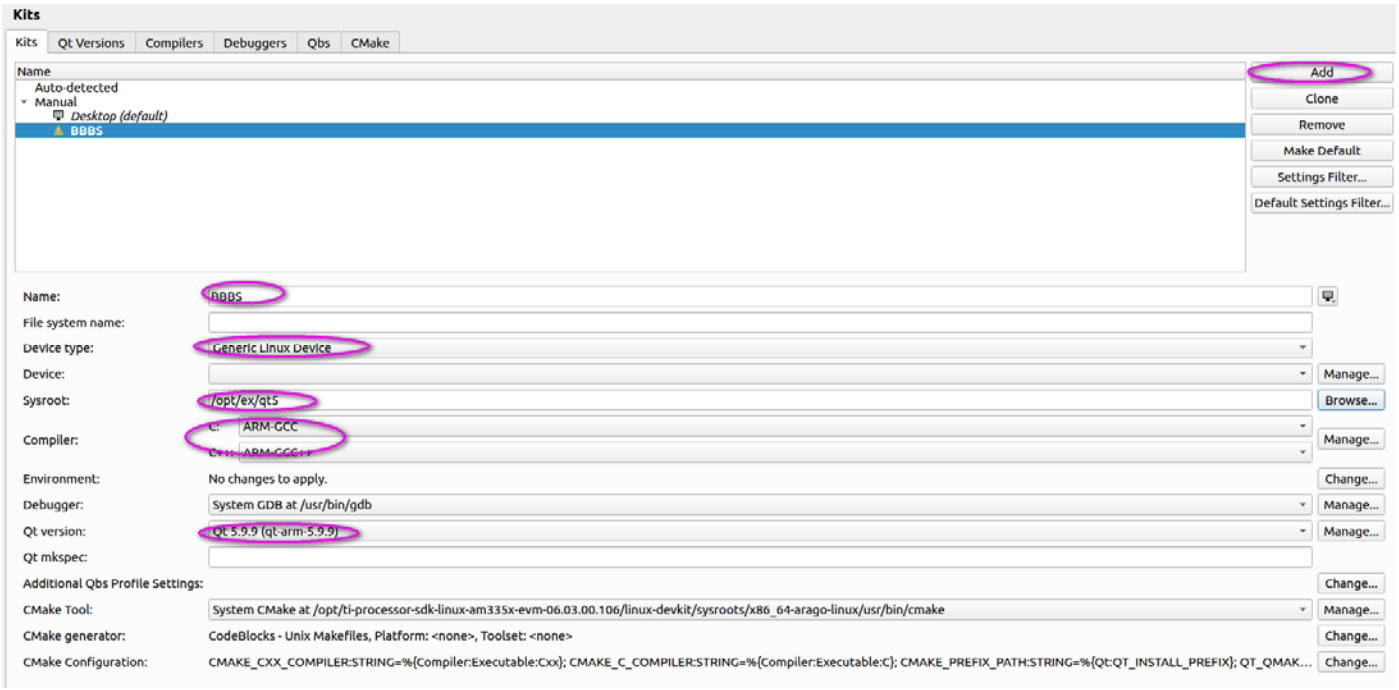
添加交叉编译器(arm-linux-gnueabihf-gcc)



添加 qmake 编译工具

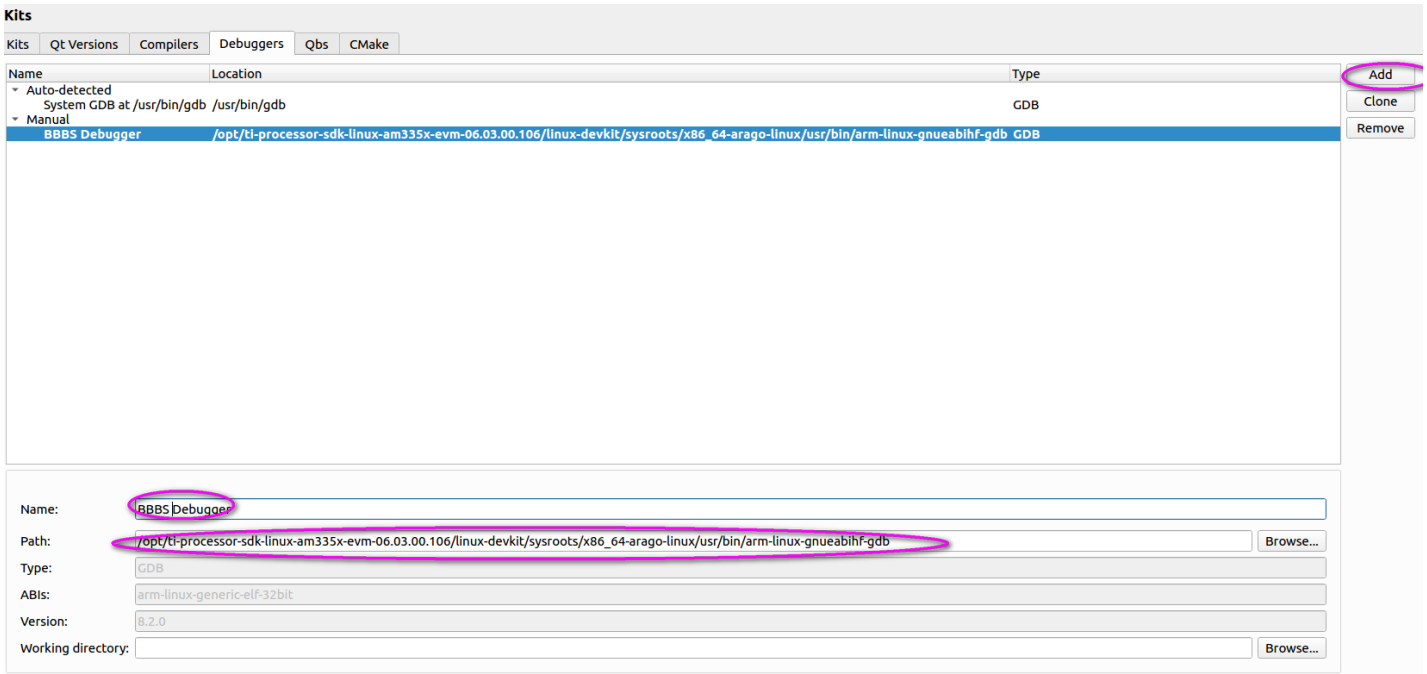


添加构建套件（Kit）



添加 Debuggers，点击"Debuggers -> Browse…"，选择对应平台 GDB，具体 路 径 为

/TI-SDK/linux-devkit/sysroots/x86_64-arago-linux/usr/bin/arm-linux-gnueabihf-gdb



注：QtCreator 使用 QT 开发，在 QtCreator 菜单 -> 帮助 -> 关于 QtCreator 就可以查看对应版本。

4. 虚拟机中运行 QT 程序

```
make clean

make confclean

./qt-desktop.sh 2>&1 | tee confLog.txt
```

查看 log，可知使用 mkspecs/linux-g++下的 qmake.conf

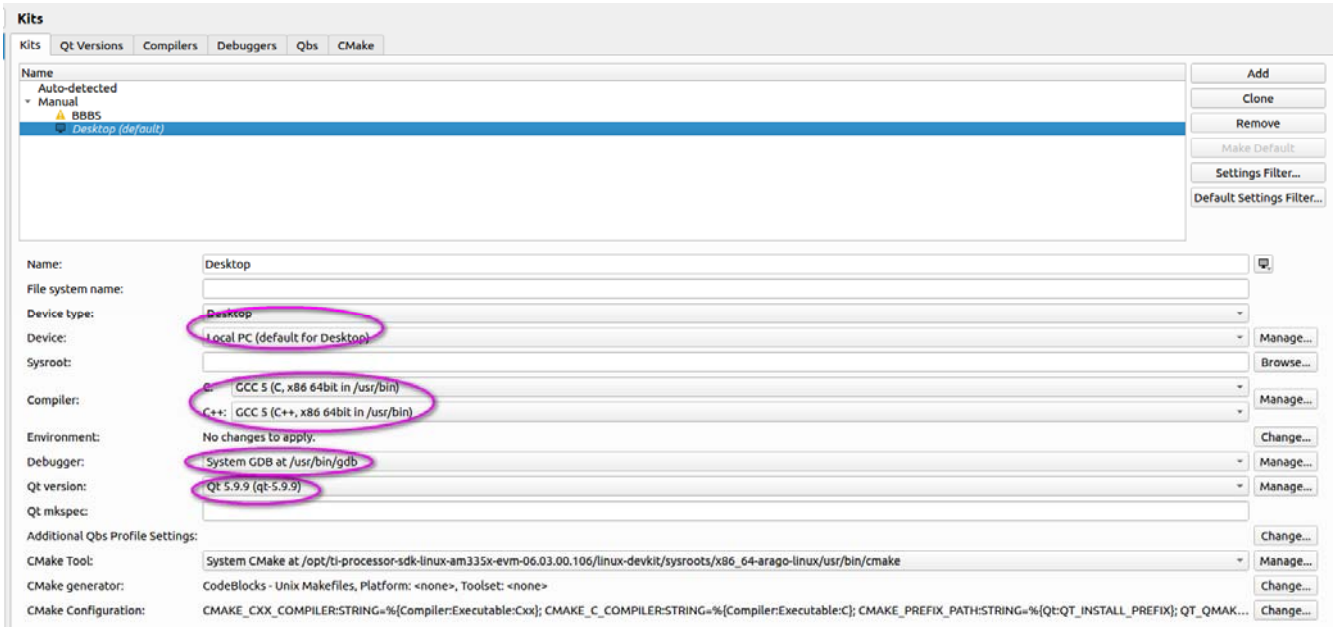
如果编译出错，有可能系统的 gcc 版本不够，需要升级为 gcc 8。

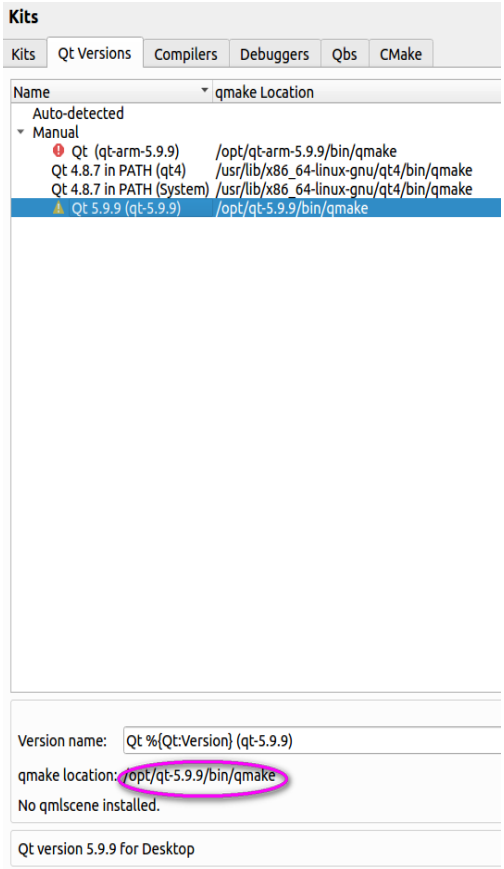
```
make -j4 2>&1 | tee makeLog.txt

make install -j4 2>&1 | tee installLog.txt
```

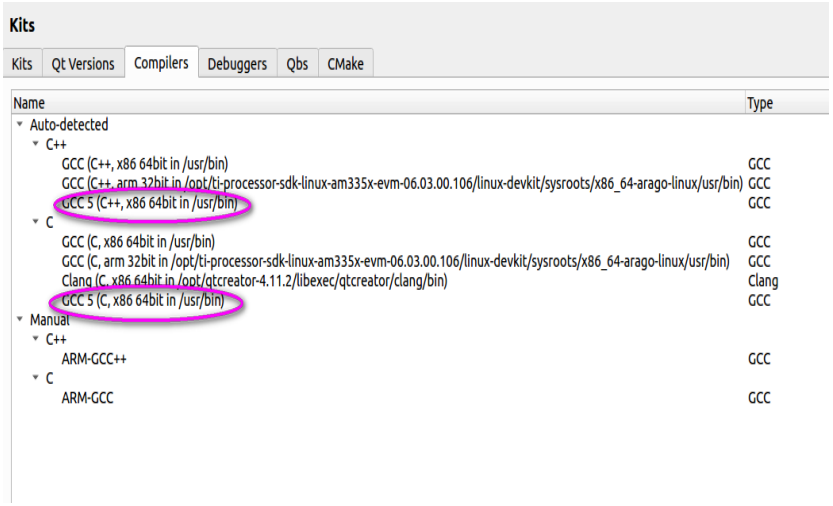
编译成功后，需要拷贝字体

```
mkdir /opt/qt-5.9.9/lib/fonts

cp STKAITI.TTF /opt/qt-5.9.9/lib/fonts/
```

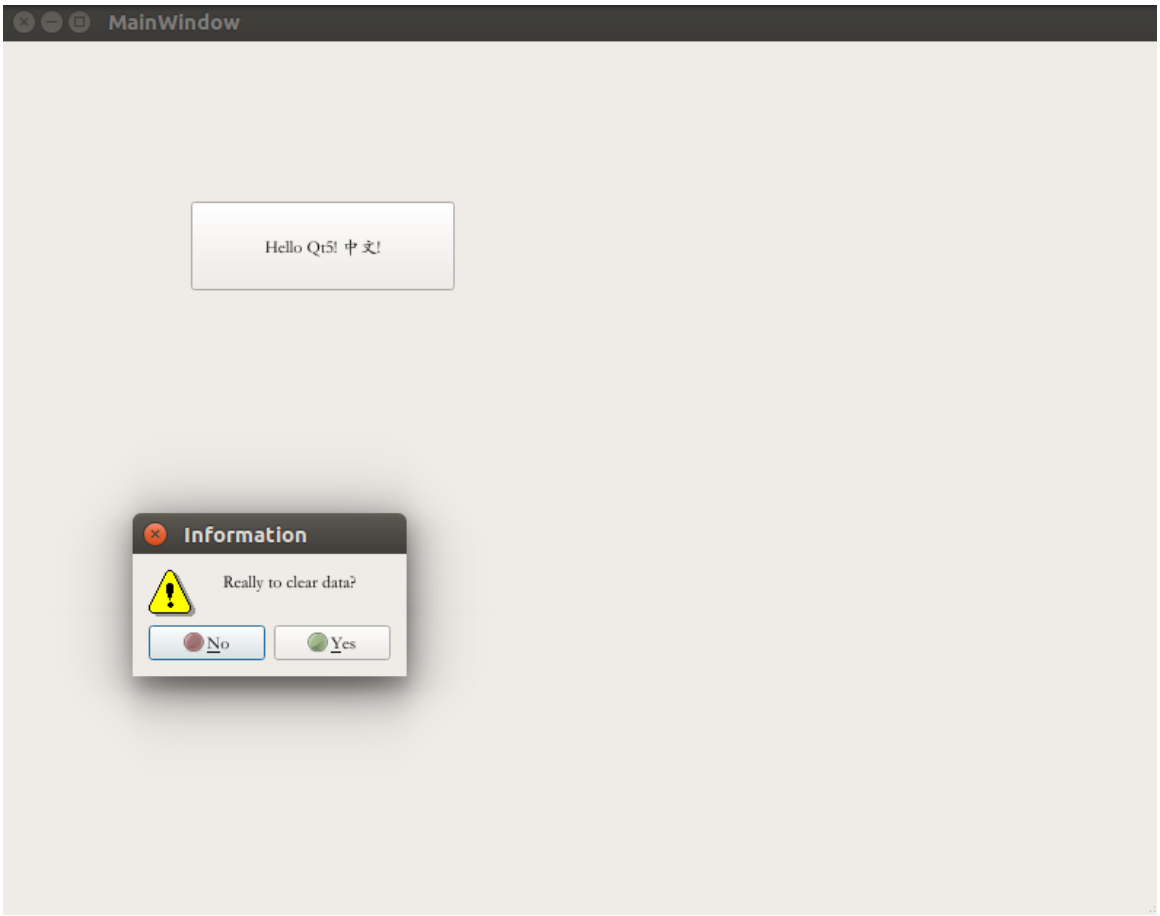
设置 Kit 如下:




Kits
Kits
Qt Versions
Compilers
Debuggers
Qbs
CMake
Name
qmake Location
Auto-detected
Manual
Qt (qt-arm-5.9.9) /opt/qt-arm-5.9.9/bin/qmake
Qt 4.8.7 in PATH (qt4) /usr/lib/x86_64-linux-gnu/qt4/bin/qmake
Qt 4.8.7 in PATH (System) /usr/lib/x86_64-linux-gnu/qt4/bin/qmake
Qt 5.9.9 (qt-5.9.9) /opt/qt-5.9.9/bin/qmake
Version name:
Qt %{Qt:Version} (qt-5.9.9)
qmake location:
/opt/qt-5.9.9/bin/qmake
No qmlscene installed.
Qt version 5.9.9 for Desktop


Kits
Kits
Qt Versions
Compilers
Debuggers
Qbs
CMake
Name
Type
Auto-detected
C++
GCC (C++, x86 64bit in /usr/bin) GCC
GCC (C++, arm 32bit in /opt/ti-processor-sdk-linux-am335x-evm-06.03.00.106/linux-devkit/sysroots/x86_64-arago-linux/usr/bin) GCC
GCC 5 (C++, x86 64bit in /usr/bin) GCC
C
GCC (C, x86 64bit in /usr/bin) GCC
GCC (C, arm 32bit in /opt/ti-processor-sdk-linux-am335x-evm-06.03.00.106/linux-devkit/sysroots/x86_64-arago-linux/usr/bin) GCC
Clang (C, x86 64bit in /opt/qtcreator-4.11.2/libexec/qtcreator/clang/bin) Clang
GCC 5 (C, x86 64bit in /usr/bin) GCC
Manual
C++
ARM-GCC++ GCC
C
ARM-GCC GCC

在虚拟机的运行效果：


MainWindow
Hello Qt5! 中文!
Information
Really to clear data?
No
Yes