MYD-YT113X QT Application Note

File status: [ ] Draft [√]Release	**FILE ID:**	MYIR-MYD-YT113X-SW-AN01-EN-L5.4.61
	VERSION:	V2.0[Doc]
	AUTHOR:	MSW0307
	RELEASE:	2024-08-09
	UPDATED:	2024-07-29

Revision History

VERSION	AUTHOR	PARTICIPANT	DATE	DESCRIPTION
V1.0[Doc]	MSW0202	MSW019	2023-06-08	Initial version
V2.0[Doc]	MSW0307	MSW041	2024-08-09	Updated software resources and operating procedures

CONTENT

1. Overview

Qt is a cross-platform graphical application development framework suitable for devices and platforms of various sizes, offering multiple versions for users to choose from. The MYD-YT113X series development board utilizes Qt version 5.12.5 for application development. During the Qt application development process, it is recommended to use the Qt Creator integrated development environment to develop Qt applications on Linux, while automatically cross-compiling for the ARM architecture of the development board.

This chapter uses the SDK tools built with Yocto as the cross-compilation toolchain, combined with Qt Creator to rapidly develop graphical applications.

2. Hardware Resources

- MYD-YT113X Development Board

3. Software Resources

- ubuntu 20.04 64bit
- qt-opensource-linux-x64-5.12.5.run
- Cross-compilation toolchain

4. Environment Preparation

4.1. Installing the QT Cross-Compilation Toolchain

This SDK provided by MYiR in addition to a variety of source code also provides the necessary QT cross-tool chain, the user needs to use the QT cross-compiler tool chain to establish an independent development environment.

- **create a new toolchain directory**

Unpack the toolchain to the specified directory according to the user's needs. The "*03-Tools/Complie Toolchain*" directory in the software directory of the download file stores the compilation chain file "*gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir-qt5.12.5-myir.tar.gz*" Copy the file to Ubuntu, and then unpack the file, the user can choose the directory to unpack, here take ~/opt directory as an example:

- **Decompress the compilation chain**

Unzip to the host's ~/opt directory:

```
 tar -xf gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir-qt5.12.5-myir.tar.gz -C ~/opt
```

- **Install Ubuntu20.04**

A Ubuntu desktop system is required; all subsequent operations will be carried out on the "*Ubuntu 20.04 64-bit*" version. Please install the desktop system on your own.

5. Installing Qt5.12.5 APP

Here,we need to download the installation package "*qt-opensource-linux-x64-5.12.5.run*",It includes Qt Creator.However,the Qt official website no longer provides offline installation packages for versions below "*Qt 5.15*",so please find the resources on your own for installation.

- **Installing Required Libraries for Qt**

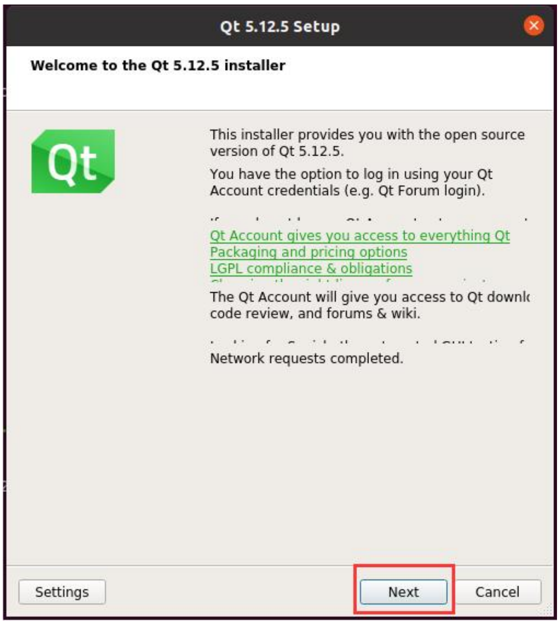
Before installing "*Qt Creator*", you need to configure the host environment. Please install the following resource packages:

```
PC$ sudo apt-get update
PC$ sudo apt install --reinstall libxcb-xinerama0
PC$ sudo apt install cmake build-essential libgl1-mesa-dev libglu1-mesa-dev freeglut3-dev cmake
```

- **Place the installation package in Ubuntu and run it.**

```
PC$ sudo ./qt-opensource-linux-x64-5.12.5.run
```

1. Run "*qt-opensource-linux-x64-5.12.5.run*" and the QT installer window will pop up, click next.。



Qt 5.12.5 Setup

Welcome to the Qt 5.12.5 installer

This installer provides you with the open source version of Qt 5.12.5.

You have the option to log in using your Qt Account credentials (e.g. Qt Forum login).

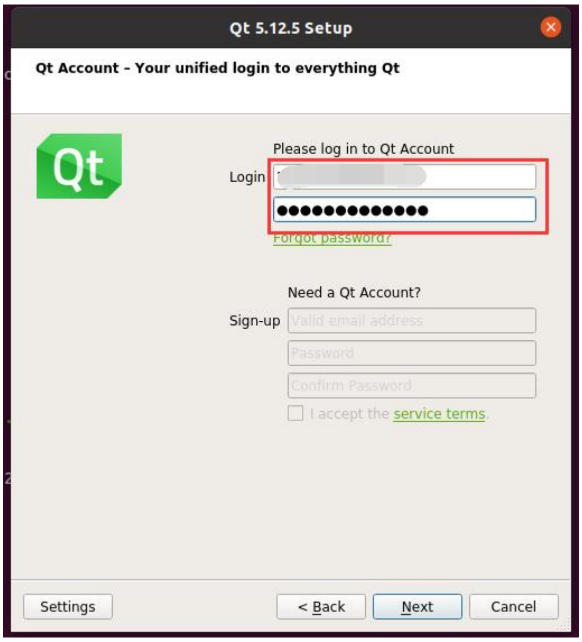
Qt Account gives you access to everything Qt
Packaging and pricing options
LGPL compliance & obligations

The Qt Account will give you access to Qt downlc code review, and forums & wiki.

Network requests completed.

Figure 5-1. QT installation program window

2. Enter your Qt account username and password. If you haven ‘t registered yet, you’ ll need to register first. Then click "Next"



Qt 5.12.5 Setup

Qt Account - Your unified login to everything Qt

Please log in to Qt Account

Login [redacted]

●●●●●●●●●●●●●

Forgot password?

Need a Qt Account?

Sign-up Valid email address

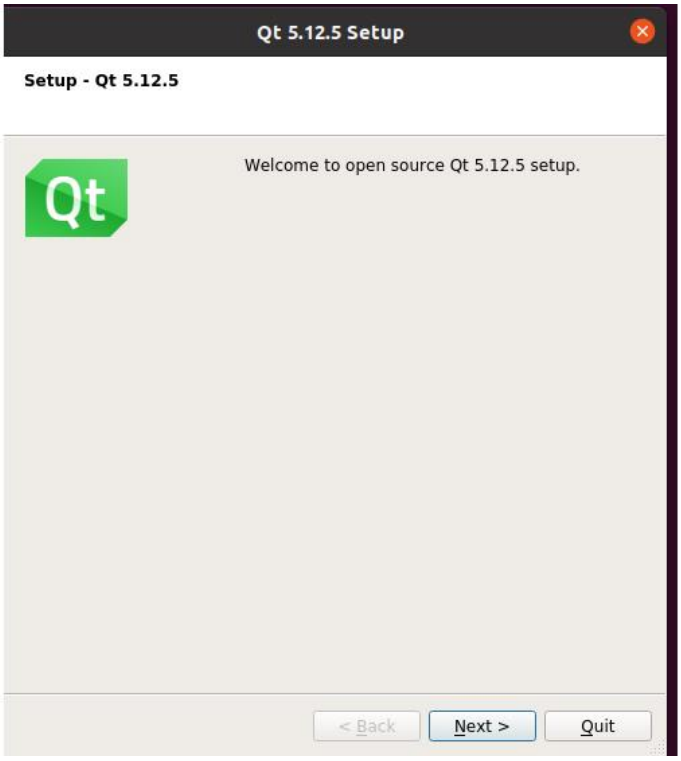
Password

Confirm Password

□ I accept the service terms.

Figure 5-2. Entering Account Number and Password

3. Continue to click next

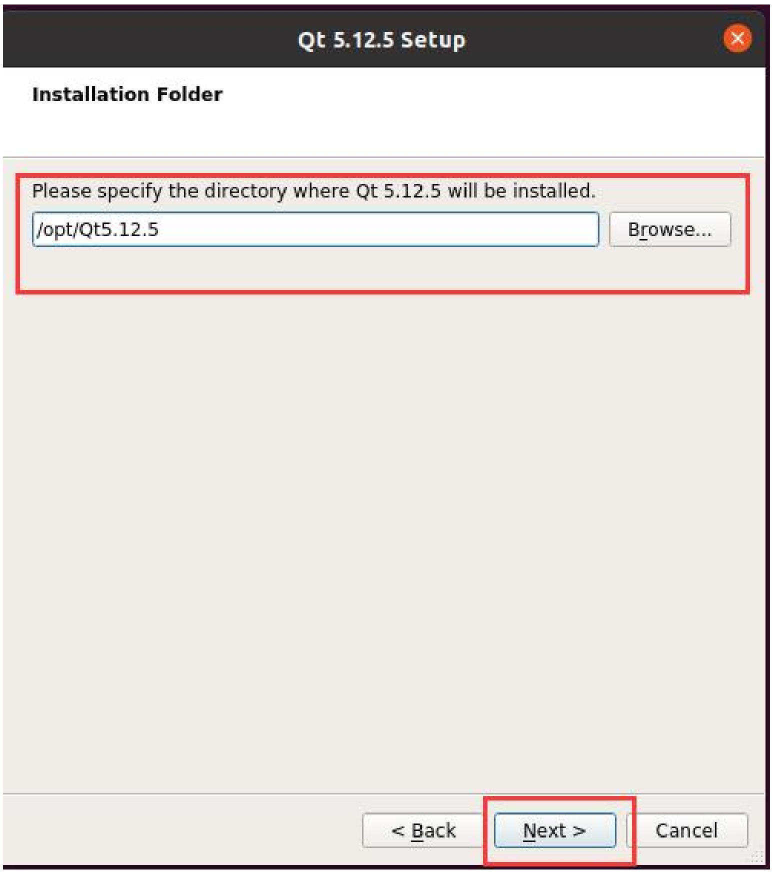


Setup - Qt 5.12.5

Welcome to open source Qt 5.12.5 setup.

Figure 5-3. Click Next

4. Select the folder where "*Qt5.12.5*" is installed and click next.



Qt 5.12.5 Setup

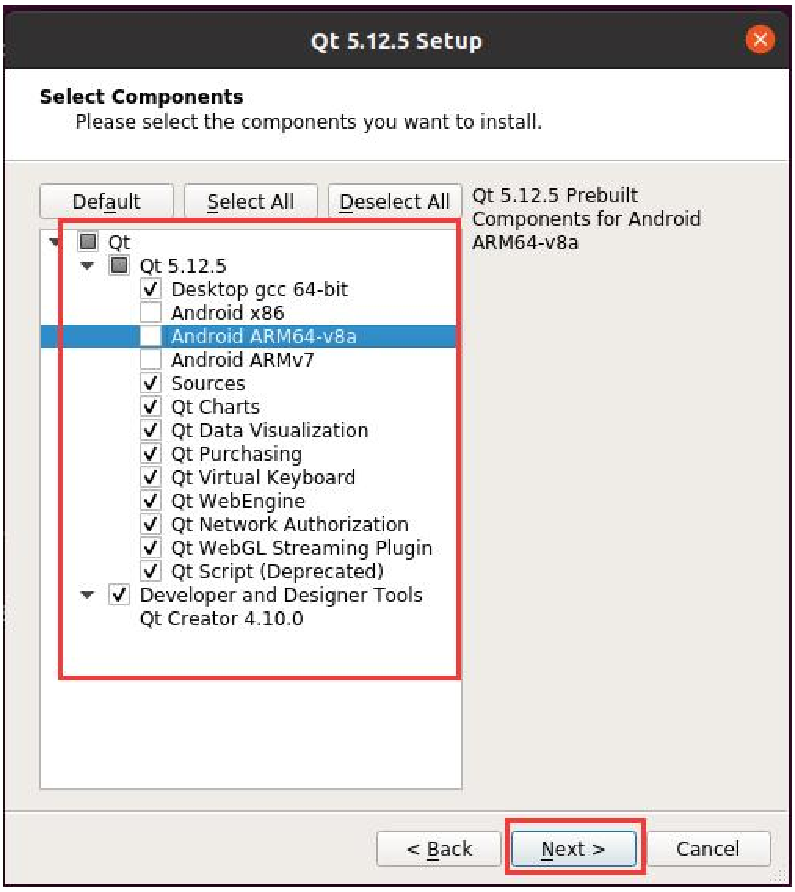
Installation Folder

Please specify the directory where Qt 5.12.5 will be installed.

/opt/Qt5.12.5

Figure 5-4. Select an installation directory

5. Select the components you want to install, and then click next.



Select Components

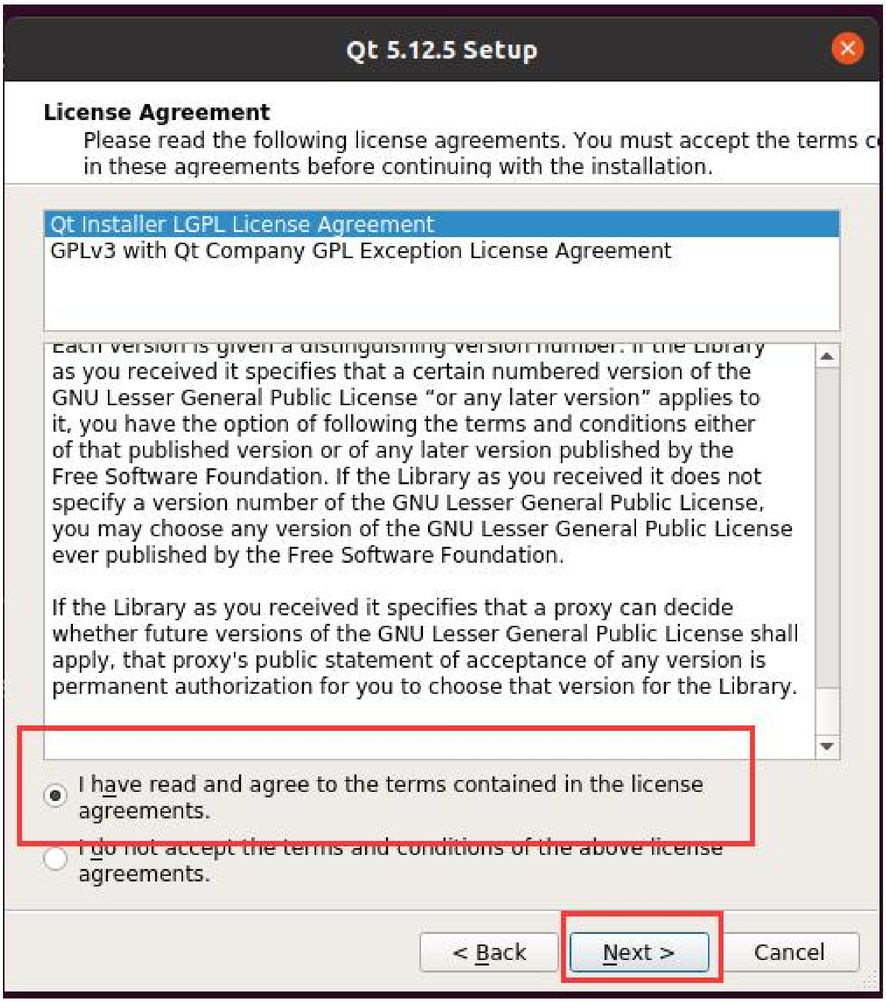
Please select the components you want to install.

- ▣ Qt
  - ▣ Qt 5.12.5
    - ☑ Desktop gcc 64-bit
    - □ Android x86
    - □ Android ARM64-v8a
    - □ Android ARMv7
    - ☑ Sources
    - ☑ Qt Charts
    - ☑ Qt Data Visualization
    - ☑ Qt Purchasing
    - ☑ Qt Virtual Keyboard
    - ☑ Qt WebEngine
    - ☑ Qt Network Authorization
    - ☑ Qt WebGL Streaming Plugin
    - ☑ Qt Script (Deprecated)
  - ☑ Developer and Designer Tools
    - Qt Creator 4.10.0

Qt 5.12.5 Prebuilt Components for Android ARM64-v8a

Figure 5-5. Select components

6. Read the agreement and select Agree, then click next



License Agreement

Please read the following license agreements. You must accept the terms c
in these agreements before continuing with the installation.

Qt Installer LGPL License Agreement
GPLv3 with Qt Company GPL Exception License Agreement

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

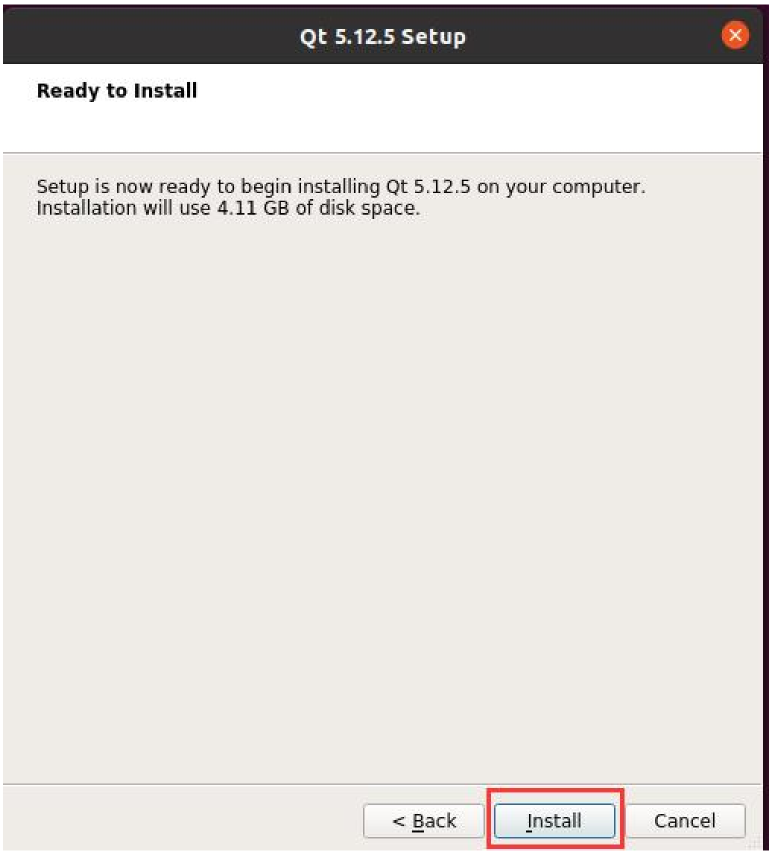
If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

● I have read and agree to the terms contained in the license agreements.

○ I do not accept the terms and conditions of the above license agreements.

Figure 5-6. Read the protocol

7. Click install to start the installation



Qt 5.12.5 Setup

Ready to Install

Setup is now ready to begin installing Qt 5.12.5 on your computer.
Installation will use 4.11 GB of disk space.

Figure 5-7. The installation starts

8. Wait for installation

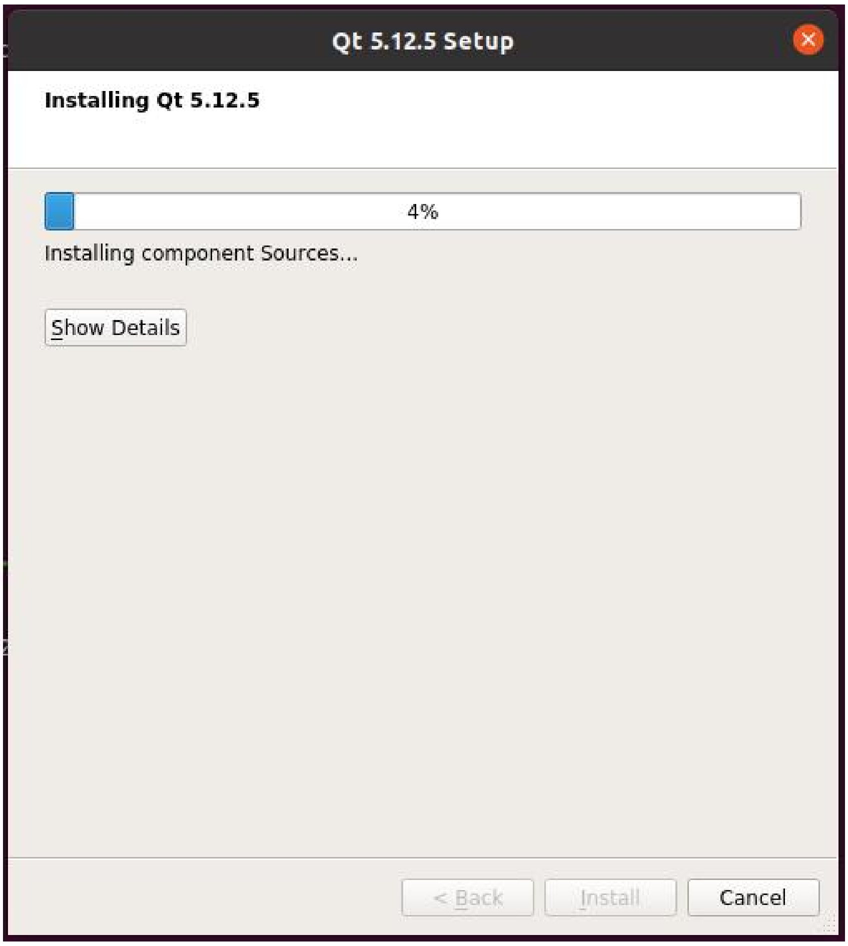
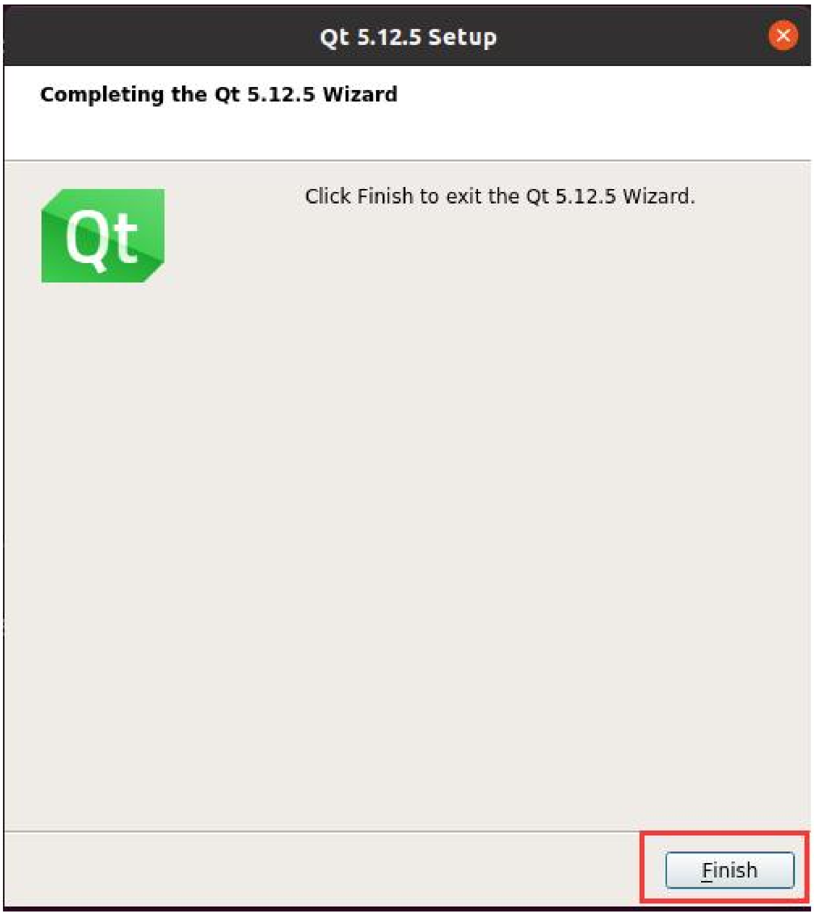


Figure 5-8. Wait for the installation

9. When the installation is complete, click finish



Qt 5.12.5 Setup

Completing the Qt 5.12.5 Wizard

Click Finish to exit the Qt 5.12.5 Wizard.

Figure 5-9. The installation is complete

6. Configure the Cross-Compilation Environment

1) Navigate to the "*Tools/QtCreator/bin*" folder within the Qt installation directory, and then execute the "*sudo ./qtcreator*" command to start "*Qt Creator*".

2) Once in "Qt Creator", select "*Tools->Options*" to access the configuration interface.

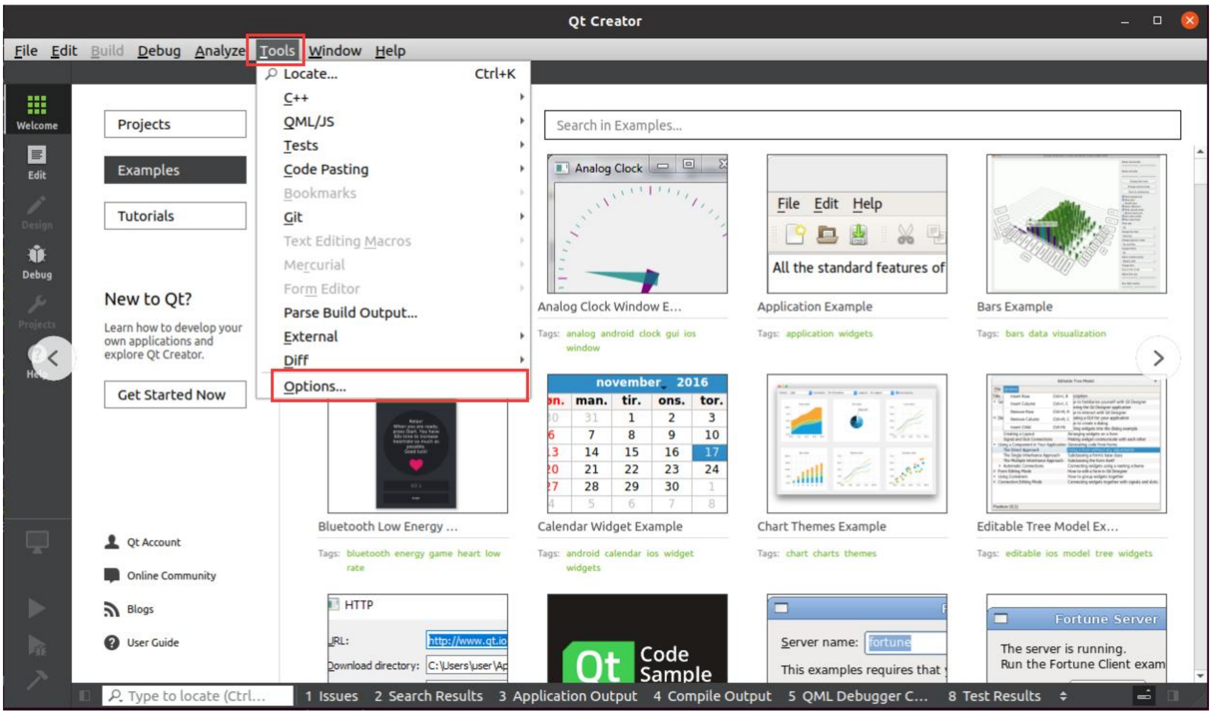
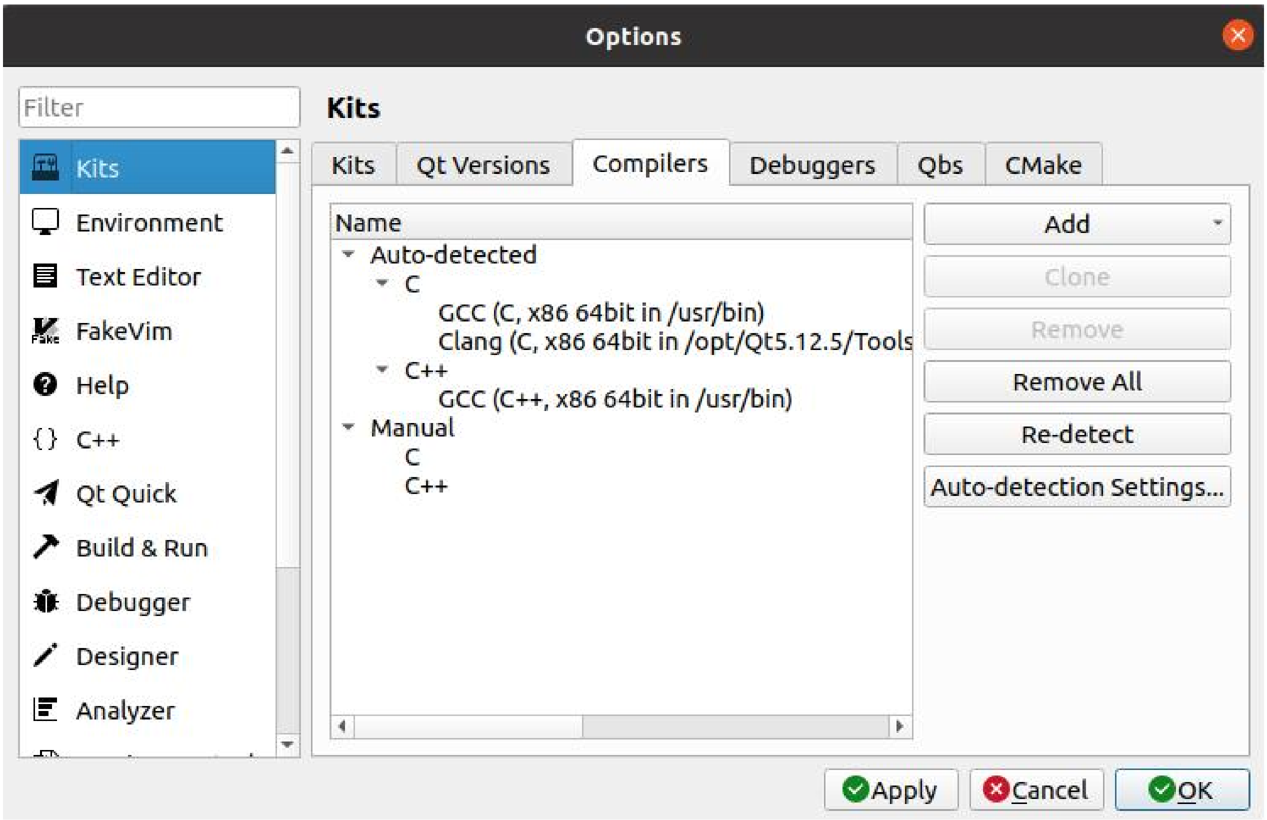


Figure 6-1. Enter Configuration Interface

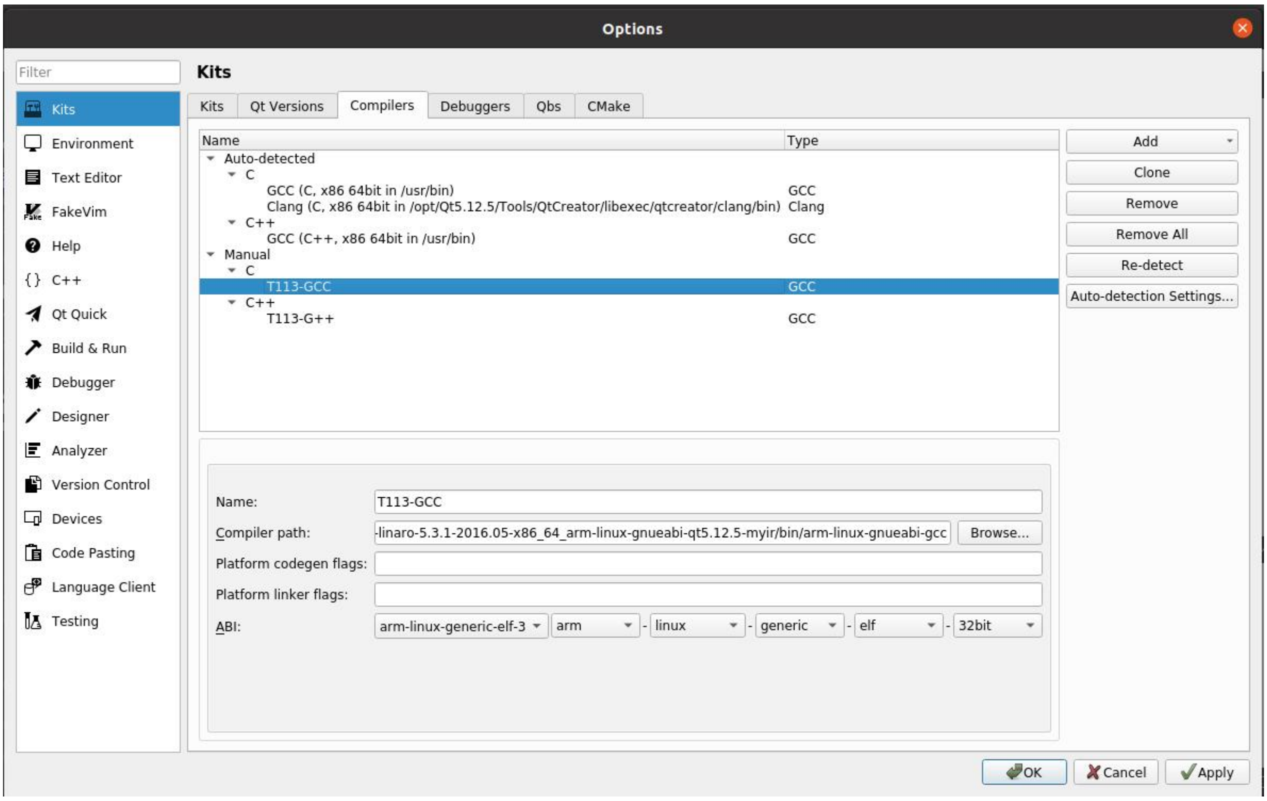
3) Select Compilers to configure the "*GCC*" and "*G++*" toolchain



```
Auto-detected
  C
    GCC (C, x86 64bit in /usr/bin)
    Clang (C, x86 64bit in /opt/Qt5.12.5/Tools
  C++
    GCC (C++, x86 64bit in /usr/bin)
Manual
  C
  C++
```

Figure 6-2. Compilers Configuration Interface

4) Select "*ADD->GCC->C*" and "*ADD->GCC->G++*" on the right side, add a customized compilation chain for " *GCC*" and "*G++*" respectively, then rename the "*GCC*" compilation chain you just added to "*T113-GCC*", and select "*/ opt/gcc-linaro-5. 3.1-2016.05-x86_64 _arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-gcc*"; rename the "*G++*" compilation chain just added to "*T113-G++*", select "*/opt/gcc-linaro-5.3.1-2016.05-x86_64_ arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-g++*". Once this is done click apply to apply.



Name	Type
Auto-detected	
C	
GCC (C, x86 64bit in /usr/bin)	GCC
Clang (C, x86 64bit in /opt/Qt5.12.5/Tools/QtCreator/libexec/qtcreator/clang/bin)	Clang
C++	
GCC (C++, x86 64bit in /usr/bin)	GCC
Manual	
C	
T113-GCC	GCC
C++	
T113-G++	GCC

Name: T113-GCC

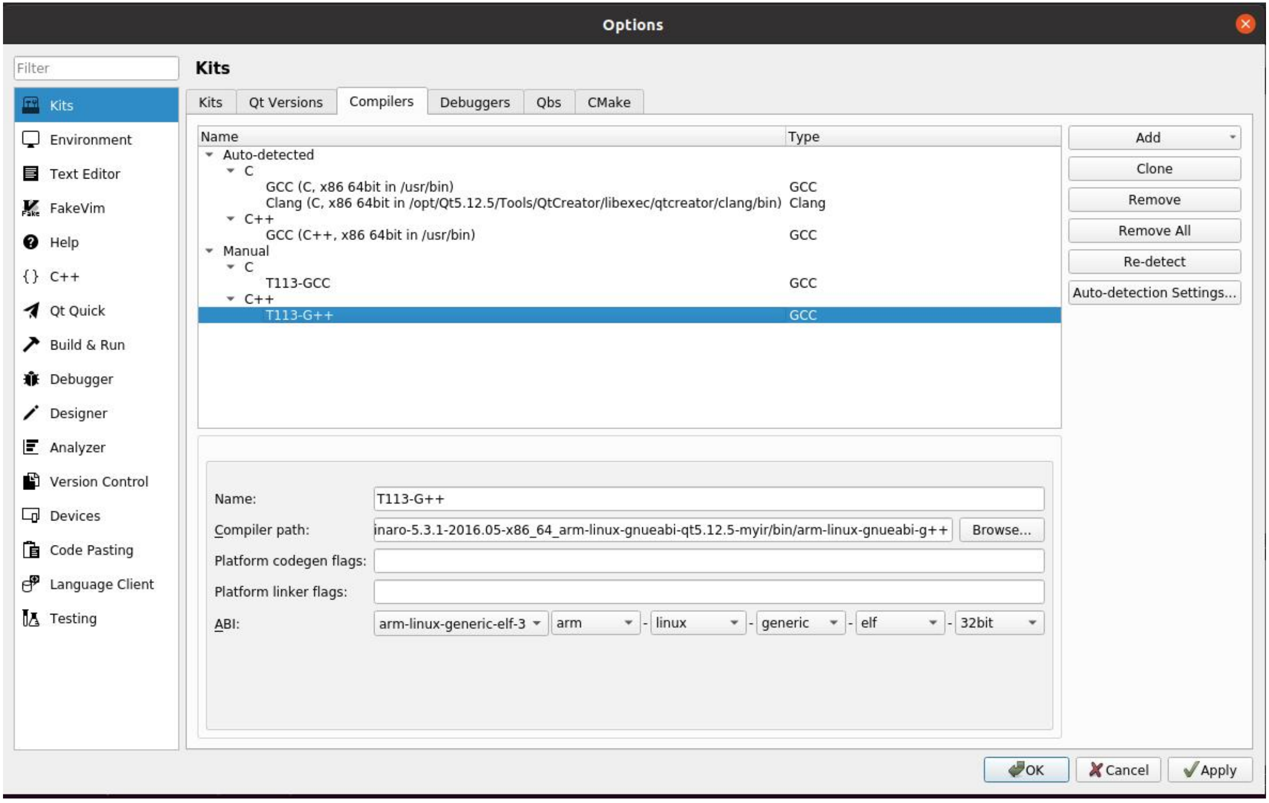
Compiler path: -linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-gcc

Platform codegen flags:

Platform linker flags:

ABI: arm-linux-generic-elf-3 arm - linux - generic - elf - 32bit

Figure 6-3. Configure GCC Toolchain



```
Name                                                                                    Type
▾ Auto-detected
    ▾ C
        GCC (C, x86 64bit in /usr/bin)                                                  GCC
        Clang (C, x86 64bit in /opt/Qt5.12.5/Tools/QtCreator/libexec/qtcreator/clang/bin) Clang
    ▾ C++
        GCC (C++, x86 64bit in /usr/bin)                                                GCC
▾ Manual
    ▾ C
        T113-GCC                                                                        GCC
    ▾ C++
        T113-G++                                                                        GCC
```

Name: T113-G++

Compiler path: inaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-g++

Platform codegen flags:

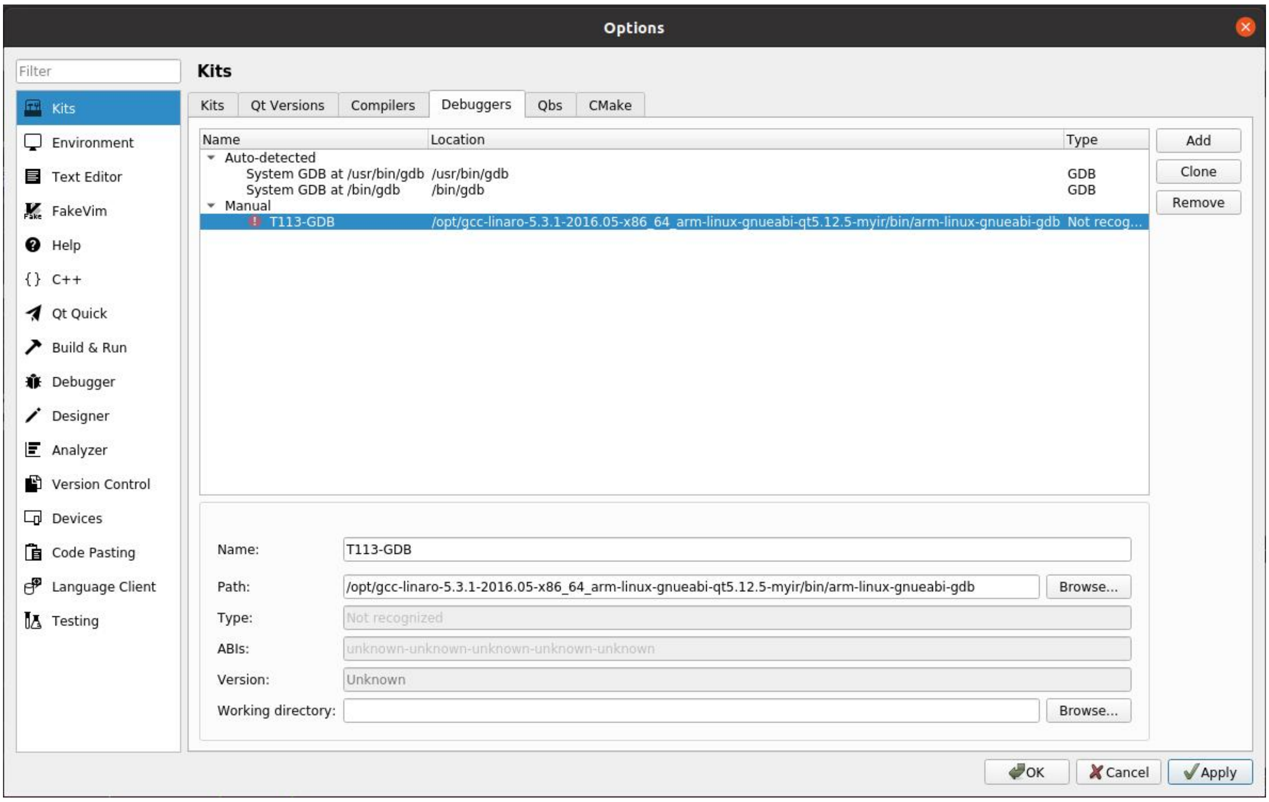
Platform linker flags:

ABI: arm-linux-generic-elf-3 ▾ arm ▾ - linux ▾ - generic ▾ - elf ▾ - 32bit ▾

Figure 6-4. Configure G++ Toolchain

5) Select "*Debugger*" to configure "*GDB*". Click Add on the right side to add a custom "*GDB*" configuration. Rename this "*GDB*" configuration to "*T113-GDB*", and select the path "*/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-gdb*". Once the configuration is completed, click Apply to "*apply*" the changes.

Note: The red exclamation mark in front of the T113-GDB here does not affect the final kits.



Name	Location	Type
Auto-detected		
System GDB at /usr/bin/gdb	/usr/bin/gdb	GDB
System GDB at /bin/gdb	/bin/gdb	GDB
Manual		
! T113-GDB	/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-gdb	Not recog...

Name: T113-GDB

Path: /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/bin/arm-linux-gnueabi-gdb

Type: Not recognized

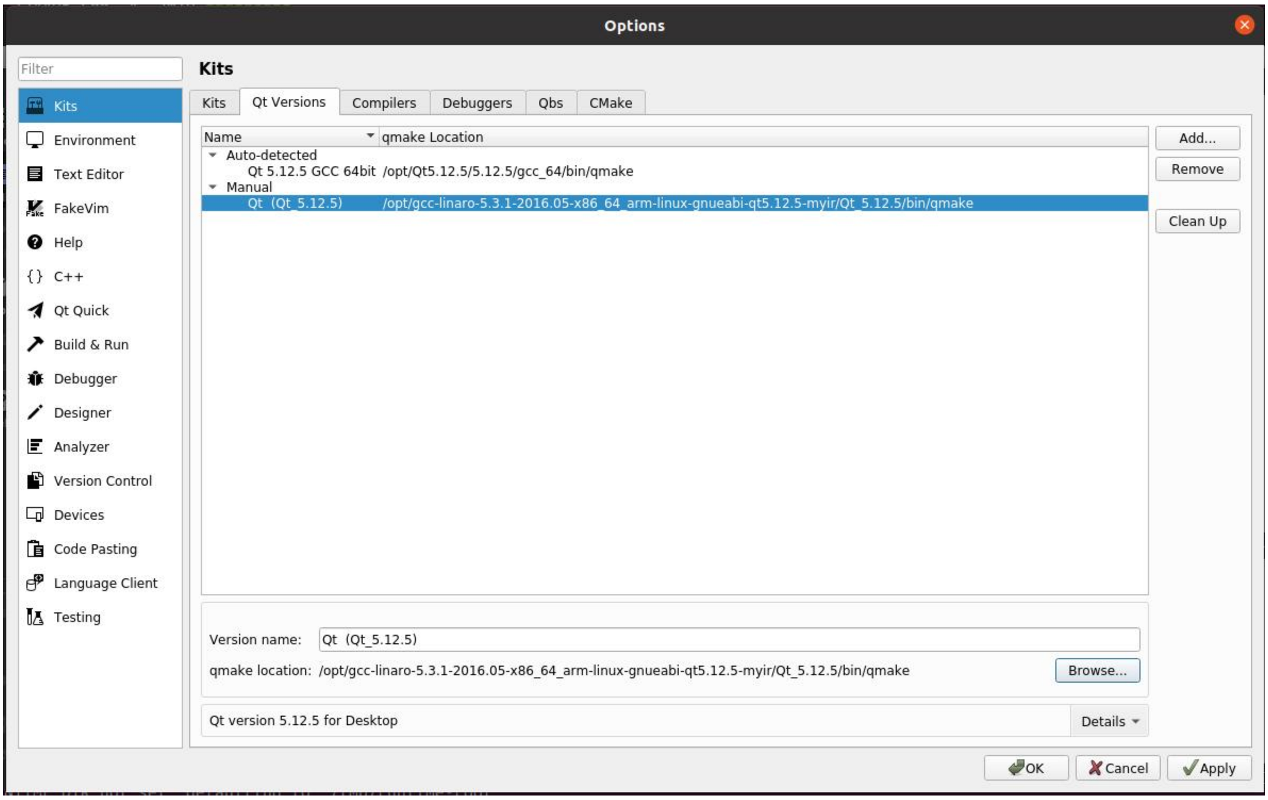
ABIs: unknown-unknown-unknown-unknown-unknown

Version: Unknown

Working directory:

Figure 6-5. Configure GDB

6) Select "*Qt Versions*" to configure "*QMake*". Here, simply click Add and select the path "/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/bin/qmake". Once the configuration is completed, click "*Apply*" to apply the changes.



Name	qmake Location
Auto-detected	
Qt 5.12.5 GCC 64bit	/opt/Qt5.12.5/5.12.5/gcc_64/bin/qmake
Manual	
Qt (Qt_5.12.5)	/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/bin/qmake

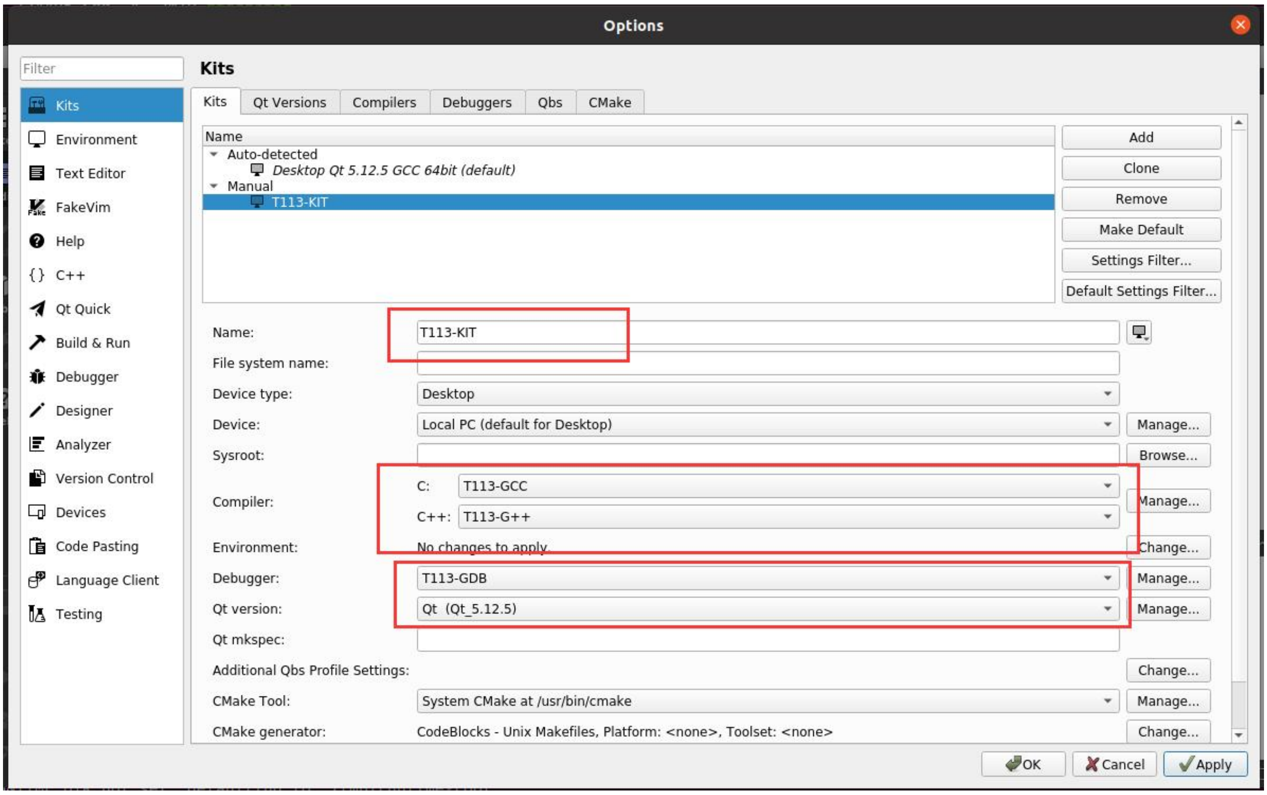
Version name: Qt (Qt_5.12.5)

qmake location: /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/bin/qmake

Qt version 5.12.5 for Desktop

Figure 6-6. Configure QMake

7) Select "*Kits*" to configure the cross-compilation toolchain. Click "*ADD*" on the right side to create a custom "*Kit*" configuration. Rename this "*Kit*" to "*T113-KIT*", and then choose the previously configured settings for "*Compiler*", "*Debugger*", and "*Qt Versions*". Once everything is set, click "*Apply*" to apply the changes. If no warnings or errors appear at this stage, the compilation toolchain configuration is complete.

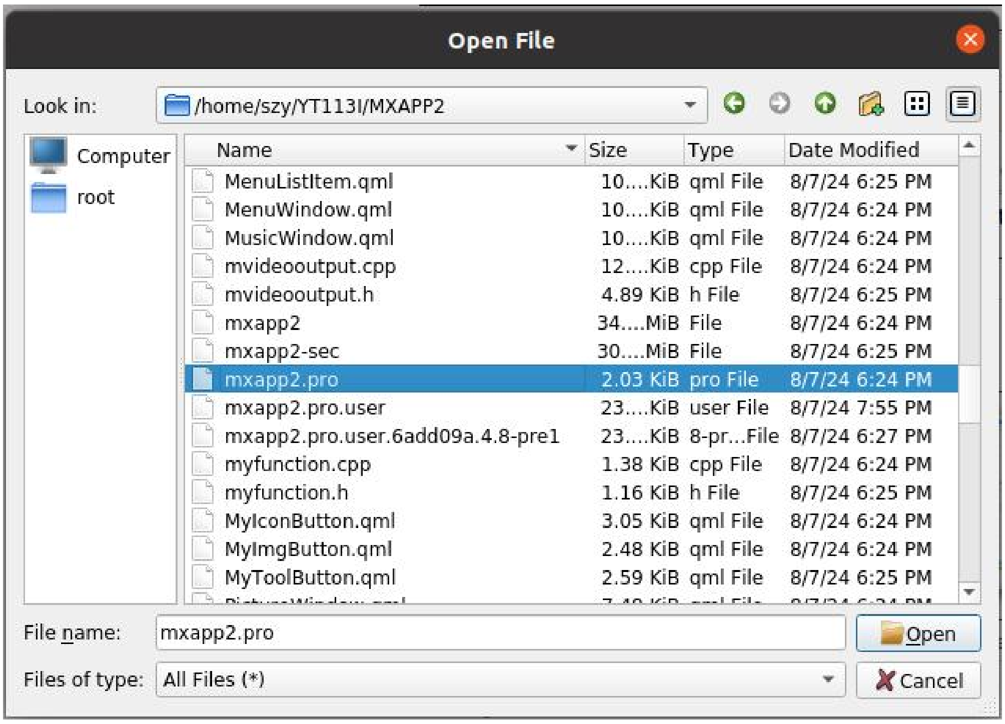


Name
- Auto-detected
  - *Desktop Qt 5.12.5 GCC 64bit (default)*
- Manual
  - T113-KIT

Name: T113-KIT
File system name:
Device type: Desktop
Device: Local PC (default for Desktop)
Sysroot:
Compiler: C: T113-GCC
C++: T113-G++
Environment: No changes to apply
Debugger: T113-GDB
Qt version: Qt (Qt_5.12.5)
Qt mkspec:
Additional Qbs Profile Settings:
CMake Tool: System CMake at /usr/bin/cmake
CMake generator: CodeBlocks - Unix Makefiles, Platform: <none>, Toolset: <none>

Figure 6-7. Configure Kit

8) Copy "*MXAPP.tar.gz*" to a working directory under "*Ubuntu*" and extract the source code. Configure it with the appropriate suite of compilation tools to compile the routine. Select "*File" -> "Open File or Project*" in the menu bar, in the dialog box that opens, browse to the directory where the "*MXAPP2*" routine is located, select the "*mxapp2.pro*" file, and click the "*Open*" button.



Open File

Look in: /home/szy/YT113I/MXAPP2

Computer
root

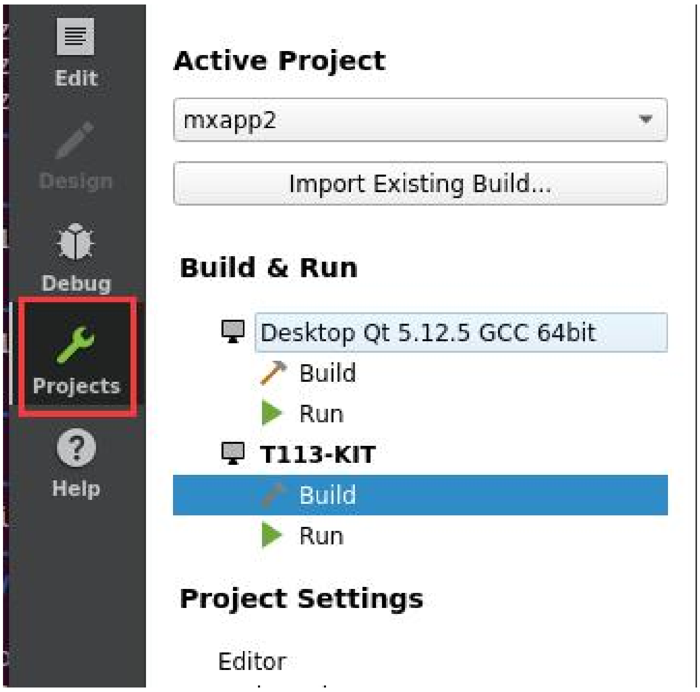
Name	Size	Type	Date Modified
MenuListItem.qml	10....KiB	qml File	8/7/24 6:25 PM
MenuWindow.qml	10....KiB	qml File	8/7/24 6:24 PM
MusicWindow.qml	10....KiB	qml File	8/7/24 6:24 PM
mvideooutput.cpp	12....KiB	cpp File	8/7/24 6:24 PM
mvideooutput.h	4.89 KiB	h File	8/7/24 6:25 PM
mxapp2	34....MiB	File	8/7/24 6:24 PM
mxapp2-sec	30....MiB	File	8/7/24 6:25 PM
mxapp2.pro	2.03 KiB	pro File	8/7/24 6:24 PM
mxapp2.pro.user	23....KiB	user File	8/7/24 7:55 PM
mxapp2.pro.user.6add09a.4.8-pre1	23....KiB	8-pr...File	8/7/24 6:27 PM
myfunction.cpp	1.38 KiB	cpp File	8/7/24 6:24 PM
myfunction.h	1.16 KiB	h File	8/7/24 6:25 PM
MyIconButton.qml	3.05 KiB	qml File	8/7/24 6:24 PM
MyImgButton.qml	2.48 KiB	qml File	8/7/24 6:24 PM
MyToolButton.qml	2.59 KiB	qml File	8/7/24 6:25 PM

File name: mxapp2.pro

Files of type: All Files (*)

Figure 6-8. select mxapp2

After opening the project, select the "*Projects*" icon from the left menu column.



Active Project

mxapp2

Import Existing Build...

Build & Run

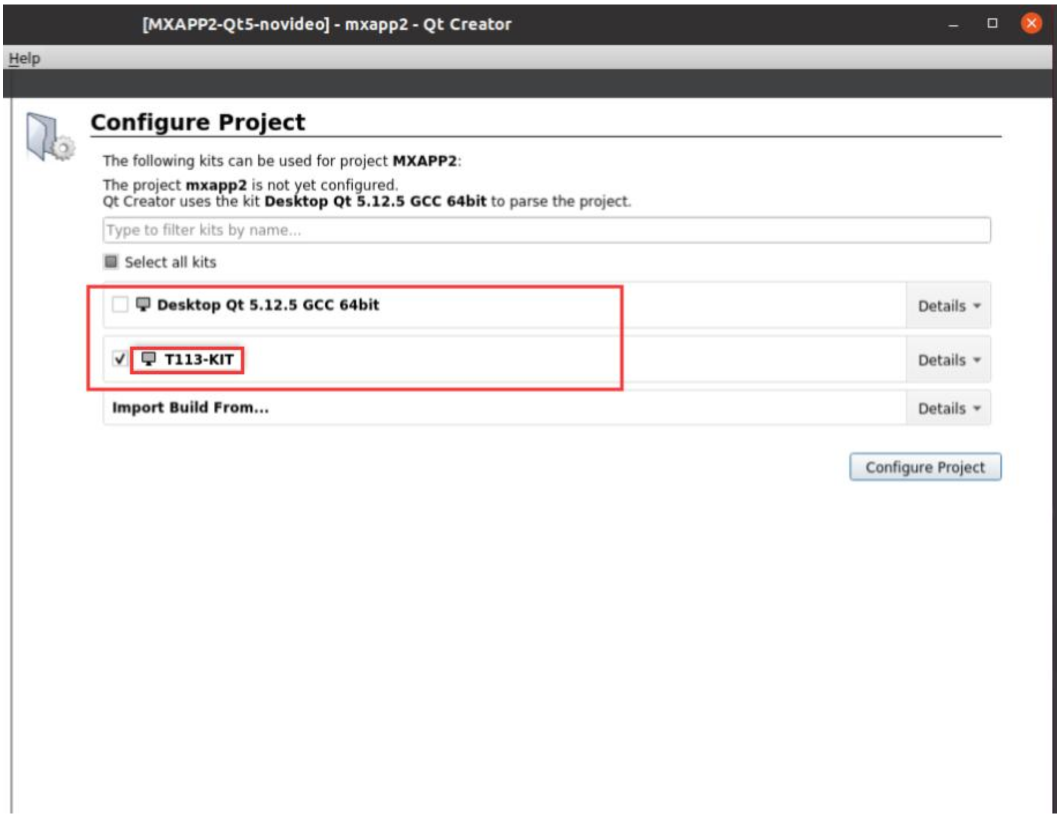
- Desktop Qt 5.12.5 GCC 64bit
  - Build
  - Run
- T113-KIT
  - Build
  - Run

Project Settings

Editor

Figure 6-9. Click on Projects

Switch to manage kits interface on the right side, under the "*Build & Run*" tab, select the kit with "*T113-KIT*" option, so that the project will use the "*T113-KIT*" related configuration kit to build the application. Then right click and select "*mxapp2*" and click Build.



[MXAPP2-Qt5-novideo] - mxapp2 - Qt Creator

Configure Project

The following kits can be used for project **MXAPP2**:

The project **mxapp2** is not yet configured.
Qt Creator uses the kit **Desktop Qt 5.12.5 GCC 64bit** to parse the project.

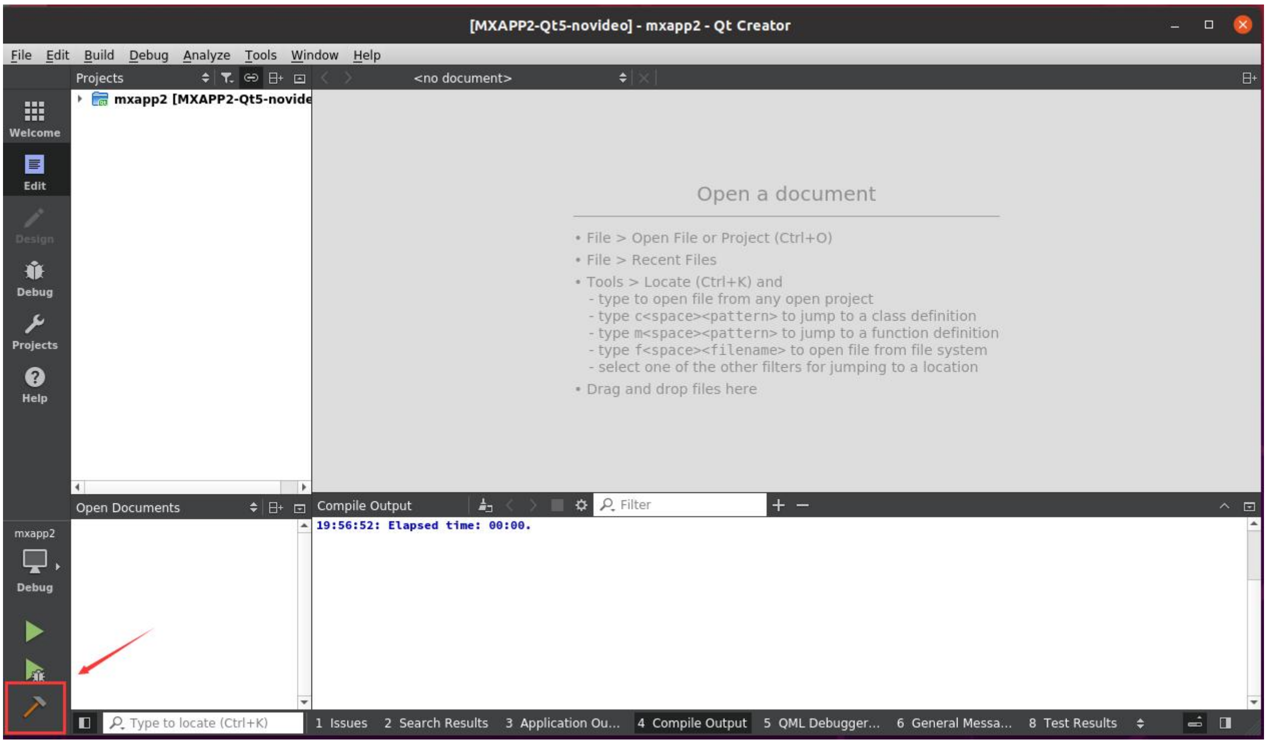
☐ Select all kits

☐ **Desktop Qt 5.12.5 GCC 64bit**

☑ **T113-KIT**

Import Build From...

Figure 6-10. Select T113-KIT for compilation

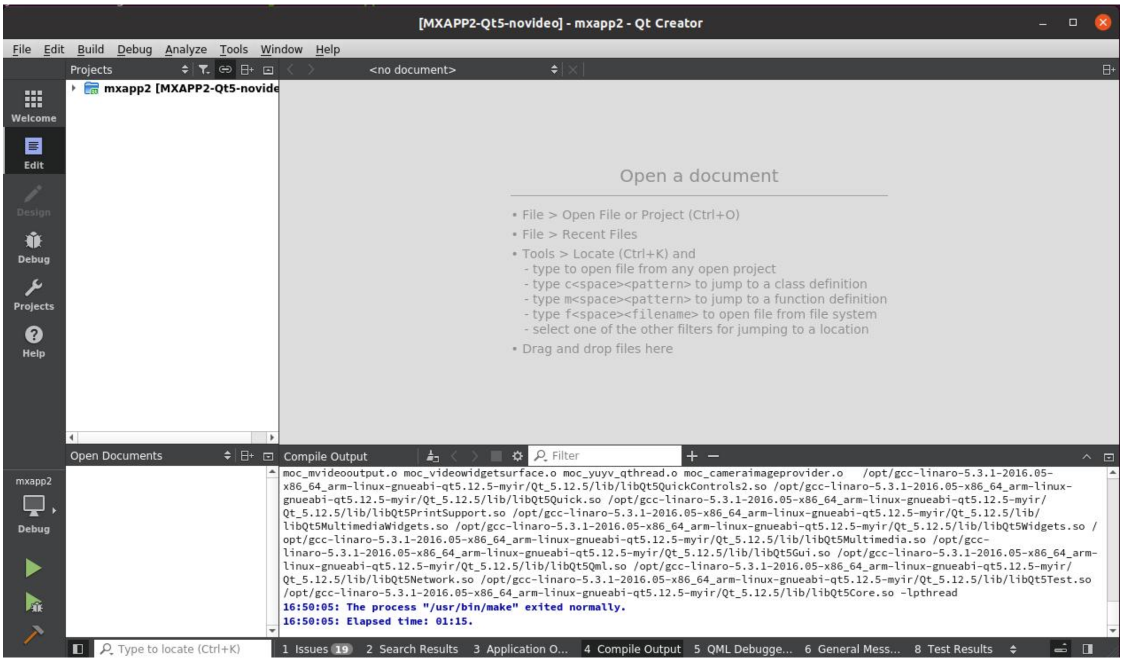


[MXAPP2-Qt5-novideo] - mxapp2 - Qt Creator

```
19:56:52: Elapsed time: 00:00.
```

File Edit Build Debug Analyze Tools Window Help
Projects
mxapp2 [MXAPP2-Qt5-novide
<no document>
Welcome
Edit
Design
Debug
Projects
Help
Open a document
• File > Open File or Project (Ctrl+O)
• File > Recent Files
• Tools > Locate (Ctrl+K) and
- type to open file from any open project
- type c<space><pattern> to jump to a class definition
- type m<space><pattern> to jump to a function definition
- type f<space><filename> to open file from file system
- select one of the other filters for jumping to a location
• Drag and drop files here
Open Documents
Compile Output
Filter
mxapp2
Debug
Type to locate (Ctrl+K)
1 Issues 2 Search Results 3 Application Ou... 4 Compile Output 5 QML Debugger... 6 General Messa... 8 Test Results

Figure 6-11 Click to start compiling



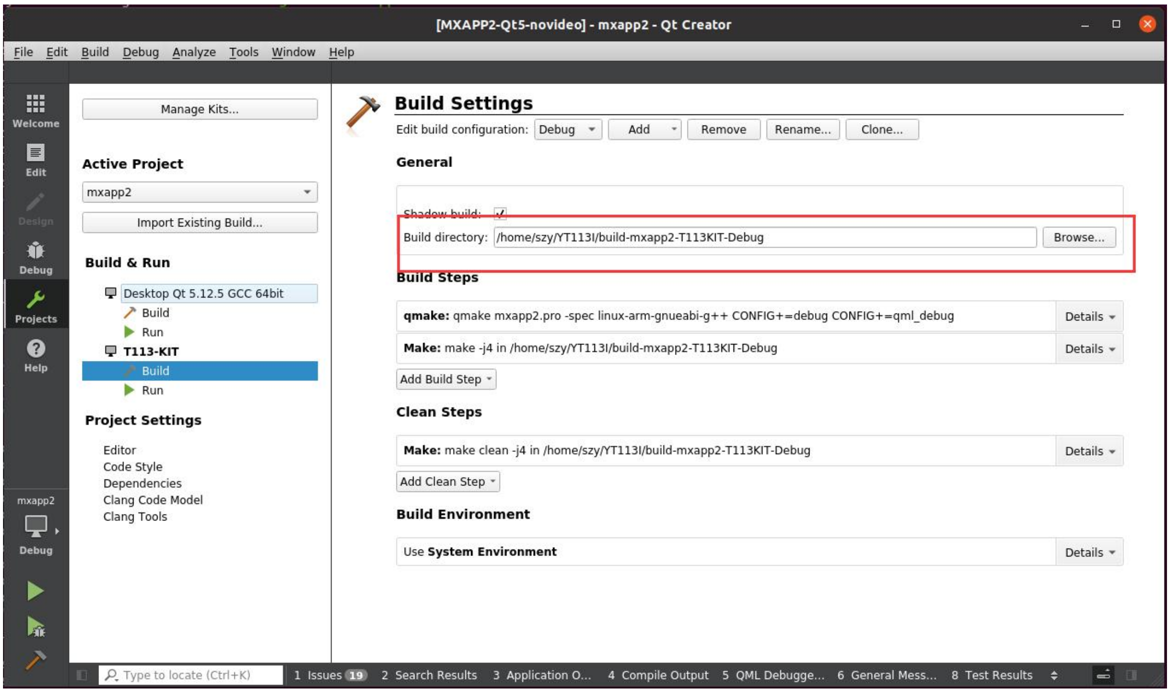
Open a document

- File > Open File or Project (Ctrl+O)
- File > Recent Files
- Tools > Locate (Ctrl+K) and
  - type to open file from any open project
  - type c<space><pattern> to jump to a class definition
  - type m<space><pattern> to jump to a function definition
  - type f<space><filename> to open file from file system
  - select one of the other filters for jumping to a location
- Drag and drop files here

```
moc_mvideooutput.o moc_videowidgetsurface.o moc_yuyv_qthread.o moc_cameraimageprovider.o   /opt/gcc-linaro-5.3.1-2016.05-
x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5QuickControls2.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-
gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Quick.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/
Qt_5.12.5/lib/libQt5PrintSupport.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/
libQt5MultimediaWidgets.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Widgets.so /
opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Multimedia.so /opt/gcc-
linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Gui.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-
linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Qml.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/
Qt_5.12.5/lib/libQt5Network.so /opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Test.so
/opt/gcc-linaro-5.3.1-2016.05-x86_64_arm-linux-gnueabi-qt5.12.5-myir/Qt_5.12.5/lib/libQt5Core.so -lpthread
16:50:05: The process "/usr/bin/make" exited normally.
16:50:05: Elapsed time: 01:15.
```

Figure 6-12. Compilation completed

After "*QtCreator*" builds the "*mxapp*" project, the compiled binary file is stored in the specified directory. Then copy the "*mxapp*" file to the development board and run it.



Active Project: mxapp2

Build & Run
- Desktop Qt 5.12.5 GCC 64bit
  - Build
  - Run
- T113-KIT
  - Build
  - Run

Project Settings
- Editor
- Code Style
- Dependencies
- Clang Code Model
- Clang Tools

Build Settings

Edit build configuration: Debug

General

Shadow build: ☑

Build directory: /home/szy/YT113I/build-mxapp2-T113KIT-Debug

Build Steps

qmake: qmake mxapp2.pro -spec linux-arm-gnueabi-g++ CONFIG+=debug CONFIG+=qml_debug

Make: make -j4 in /home/szy/YT113I/build-mxapp2-T113KIT-Debug

Clean Steps

Make: make clean -j4 in /home/szy/YT113I/build-mxapp2-T113KIT-Debug

Build Environment

Use **System Environment**

Figure 6-13. Save Path

7. References

- **Ubuntu Desktop Download Address:**

  https://ubuntu.com/download/desktop

- **QT Development Guide**

  https://www.qt.io/

Appendix A

Warranty & Technical Support Services

MYIR Electronics Limited is a global provider of ARM hardware and software tools, design solutions for embedded applications. We support our customers in a wide range of services to accelerate your time to market.

MYIR is an ARM Connected Community Member and work closely with ARM and many semiconductor vendors. We sell products ranging from board level products such as development boards, single board computers and CPU modules to help with your evaluation, prototype, and system integration or creating your own applications. Our products are used widely in industrial control, medical devices, consumer electronic, telecommunication systems, Human Machine Interface (HMI) and more other embedded applications. MYIR has an experienced team and provides custom design services based on ARM processors to help customers make your idea a reality.

The contents below introduce to customers the warranty and technical support services provided by MYIR as well as the matters needing attention in using MYIR's products.

Service Guarantee

MYIR regards the product quality as the life of an enterprise. We strictly check and control the core board design, the procurement of components, production control, product testing, packaging, shipping and other aspects and strive to provide products with best quality to customers. We believe that only quality products and excellent services can ensure the long-term cooperation and mutual benefit.

Price

MYIR insists on providing customers with the most valuable products. We do not pursue excess profits which we think only for short-time cooperation. Instead, we hope to establish long-term cooperation and win-win business with customers. So we will offer reasonable prices in the hope of making the business greater with the customers together hand in hand.

Delivery Time

MYIR will always keep a certain stock for its regular products. If your order quantity is less than the amount of inventory, the delivery time would be within three days; if your order quantity is greater than the number of inventory, the delivery time would be always four to six weeks. If for any urgent delivery, we can negotiate with customer and try to supply the goods in advance.

Technical Support

MYIR has a professional technical support team. Customer can contact us by email (support@myirtech.com), we will try to reply you within 48 hours. For mass production and customized products, we will specify person to follow the case and ensure the smooth production.

After-sale Service

MYIR offers one year free technical support and after-sales maintenance service from the purchase date. The service covers:

Technical support service

MYIR offers technical support for the hardware and software materials which have provided to customers;

- To help customers compile and run the source code we offer;
- To help customers solve problems occurred during operations if users follow the user manual documents;
- To judge whether the failure exists;
- To provide free software upgrading service.

However, the following situations are not included in the scope of our free technical support service:

- Hardware or software problems occurred during customers' own development;
- Problems occurred when customers compile or run the OS which is tailored by themselves;
- Problems occurred during customers' own applications development;
- Problems occurred during the modification of MYIR's software source code.

After-sales maintenance service

The products except LCD, which are not used properly, will take the twelve months free maintenance service since the purchase date. But following situations are not included in the scope of our free maintenance service:

- The warranty period is expired;
- The customer cannot provide proof-of-purchase or the product has no serial number;
- The customer has not followed the instruction of the manual which has caused the damage the product;
- Due to the natural disasters (unexpected matters), or natural attrition of the components, or unexpected matters leads the defects of appearance/function;
- Due to the power supply, bump, leaking of the roof, pets, moist, impurities into the boards, all those reasons which have caused the damage of the products or defects of appearance;

- Due to unauthorized weld or dismantle parts or repair the products which has caused the damage of the products or defects of appearance;
- Due to unauthorized installation of the software, system or incorrect configuration or computer virus which has caused the damage of products.

Warm tips

1. MYIR does not supply maintenance service to LCD. We suggest the customer first check the LCD when receiving the goods. In case the LCD cannot run or no display, customer should contact MYIR within 7 business days from the moment get the goods.

2. Please do not use finger nails or hard sharp object to touch the surface of the LCD.

3. MYIR suggests user purchasing a piece of special wiper to wipe the LCD after long time use, please avoid clean the surface with fingers or hands to leave fingerprint.

4. Do not clean the surface of the screen with chemicals.

5. Please read through the product user manual before you using MYIR's products.

6. For any maintenance service, customers should communicate with MYIR to confirm the issue first. MYIR's support team will judge the failure to see if the goods need to be returned for repair service, we will issue you RMA number for return maintenance service after confirmation.

Maintenance period and charges

- MYIR will test the products within three days after receipt of the returned goods and inform customer the testing result. Then we will arrange shipment within one week for the repaired goods to the customer. For any special failure, we will negotiate with customers to confirm the maintenance period.
- For products within warranty period and caused by quality problem, MYIR offers free maintenance service; for products within warranty period but out of free maintenance service scope, MYIR provides maintenance service but shall charge some basic material cost; for products out of warranty period, MYIR provides maintenance service but shall charge some basic material cost and handling fee.

Shipping cost

During the warranty period, the shipping cost which delivered to MYIR should be responsible by user; MYIR will pay for the return shipping cost to users when the product is repaired. If the warranty period is expired, all the shipping cost will be responsible by users.

Products Life Cycle

MYIR will always select mainstream chips for our design, thus to ensure at least ten years continuous supply; if meeting some main chip stopping production, we will inform customers in time and assist customers with products updating and upgrading.

Value-added Services

1. MYIR provides services of driver development base on MYIR's products, like serial port, USB, Ethernet, LCD, etc.
2. MYIR provides the services of OS porting, BSP drivers' development, API software development, etc.
3. MYIR provides other products supporting services like power adapter, LCD panel, etc.
4. ODM/OEM services.

MYIR Electronics Limited

Room 04, 6th Floor, Building No.2, Fada Road,
Yunli Inteiligent Park, Bantian, Longgang District.
Support Email: support@myirtech.com
Sales Email: sales@myirtech.com
Phone: +86-755-22984836
Fax: +86-755-25532724
Website: www.myirtech.com