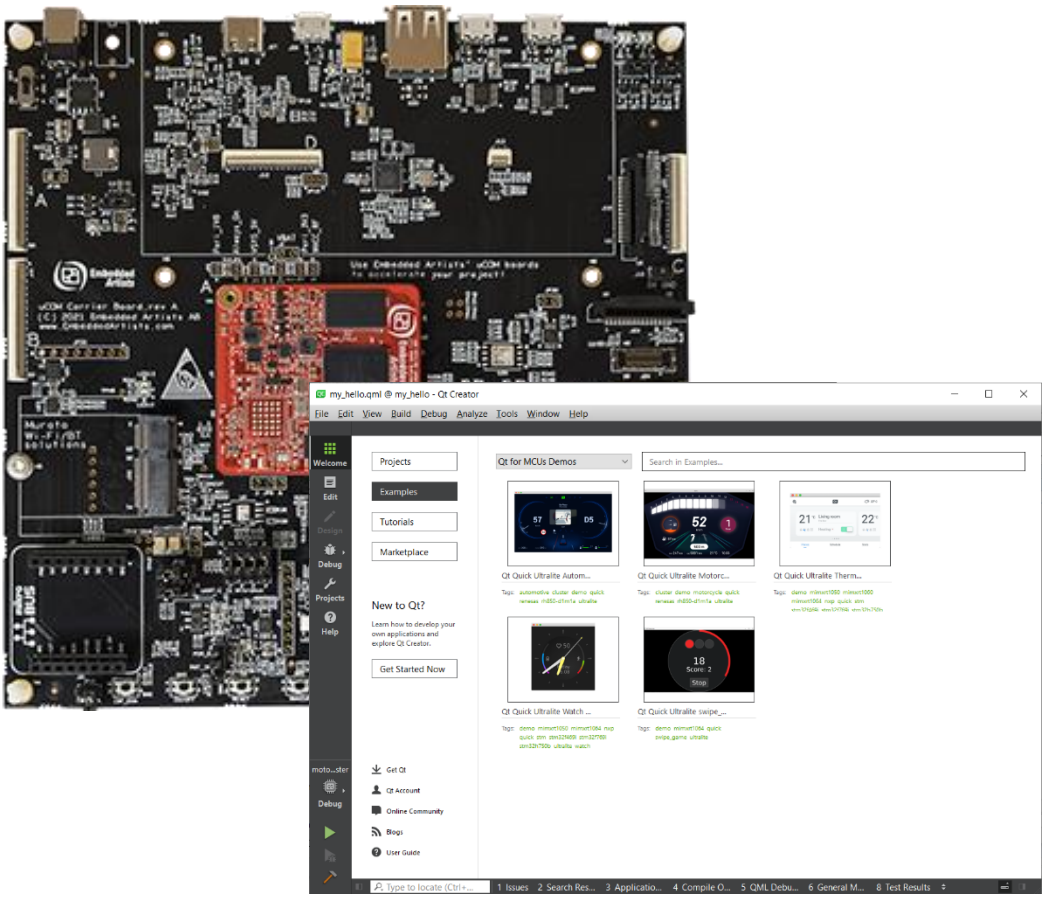
Qt for MCUs - Program Development



Get Up-and-Running Quickly and
Start Developing Your Application On Day 1!

Embedded Artists

Embedded Artists AB
Rundelsgatan 14
211 36 Malmö
Sweden

https://www.EmbeddedArtists.com

Disclaimer

Embedded Artists AB makes no representation or warranties with respect to the contents hereof and specifically disclaim any implied warranties or merchantability or fitness for any particular purpose. Information in this publication is subject to change without notice and does not represent a commitment on the part of Embedded Artists AB.

Feedback

We appreciate any feedback you may have for improvements on this document. Please send your comments to support@EmbeddedArtists.com.

Trademarks

All brand and product names mentioned herein are trademarks, services marks, registered trademarks, or registered service marks of their respective owners and should be treated as such.

Table of Contents

1 Document Revision History

Revision	*Date*	*Description*
A	2022-04-06	First release. Based on Qt for MCUs 2.0.0

2 Intro

Qt for MCUs is a complete graphics framework and toolkit with everything you need to design, develop, and deploy GUIs on embedded MCUs. Run your application on bare metal or a real-time operating system. Read more about it here: https://www.qt.io/product/develop-software-microcontrollers-mcu

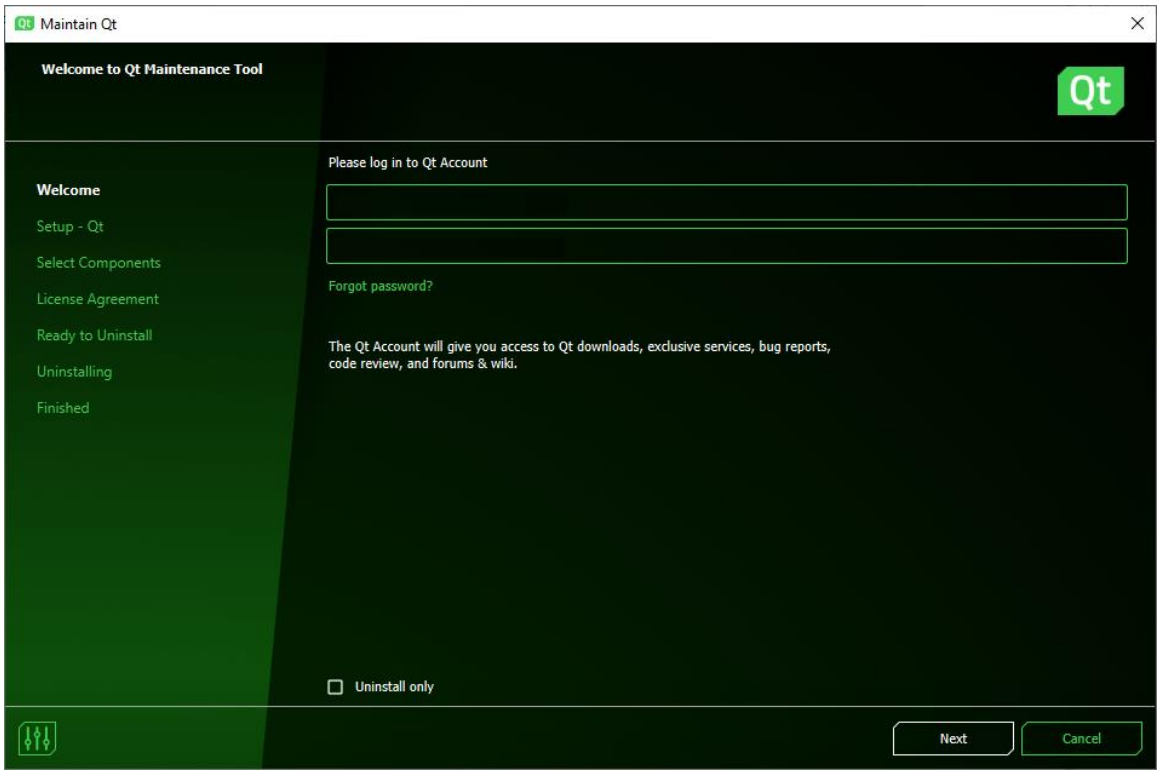
> As the time of writing this document the Qt for MCUs board support package (BSP) for Embedded Artists' iMX RT Developer Kit's is for Windows and the ARMGCC or IAR compilers only. The support is also only for the combination of Qt for MCUs 2.0.0 and NXP SDK 2.10.1.

To start program development, you need the following things, all of them:

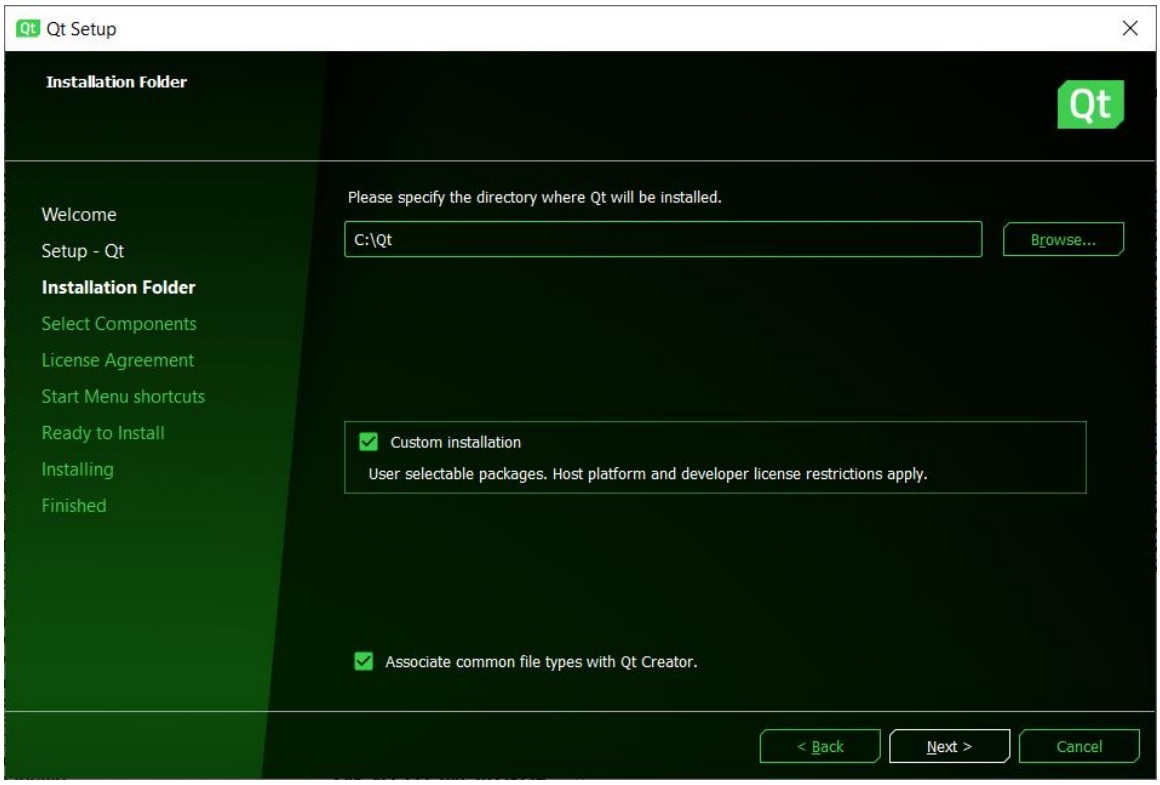
1. **Qt Installer** – The installer can be downloaded from https://www.qt.io/download after filling in a form for an evaluation license
2. **The board support package** – This is a zip file containing all the needed platform specific files to work with the *iMX RT Developer's Kit*. The zip-file can be downloaded from http://imx.embeddedartists.com.
3. **The patched SDK** – This is zip file containing the latest version of the NXP SDK patched to work with the *iMX RT Developer's Kit*. The zip-file can be downloaded from http://imx.embeddedartists.com.
4. **NXP MCUXpresso** – This is needed to flash the software
5. And of course, the ***iMX RT Developer's Kit***!

2.1 Download and install Qt for MCUs

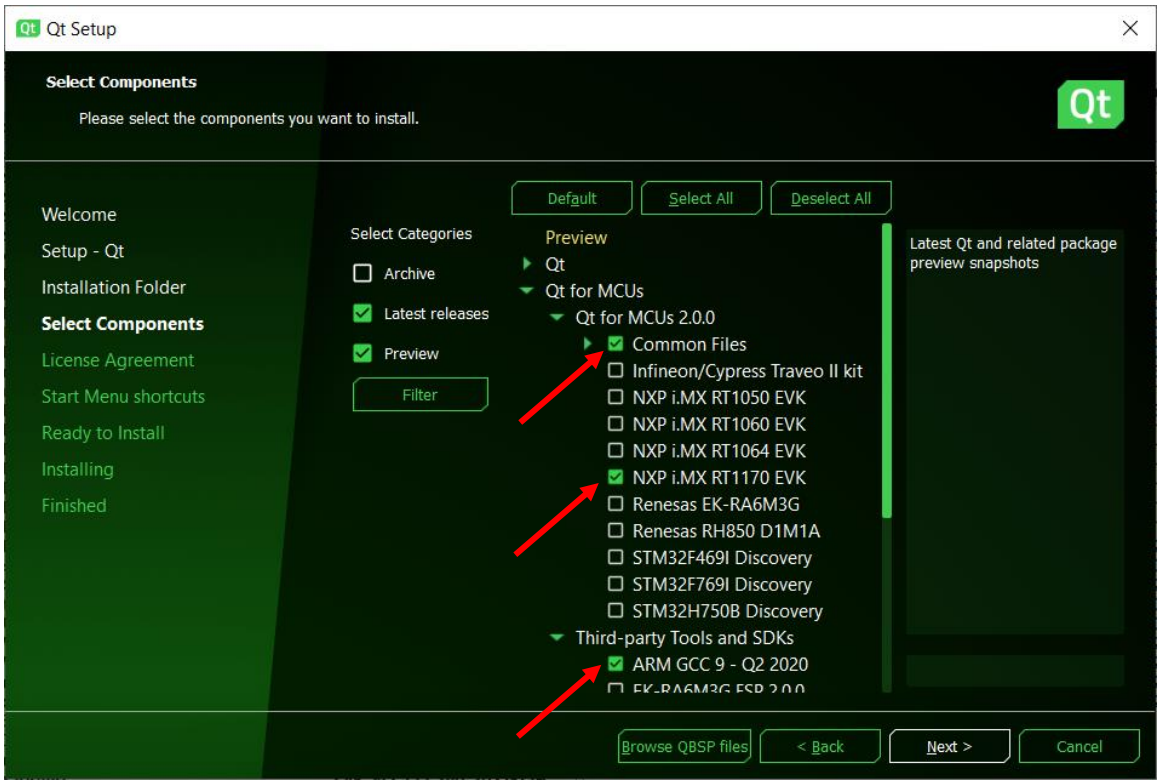
Download the installer from https://www.qt.io/download (need to fill out a registration form to start the evaluation). Start the installer, enter your credentials, and press Next.



Now make sure that *Custom installation* is selected and click Next to continue to the next page.

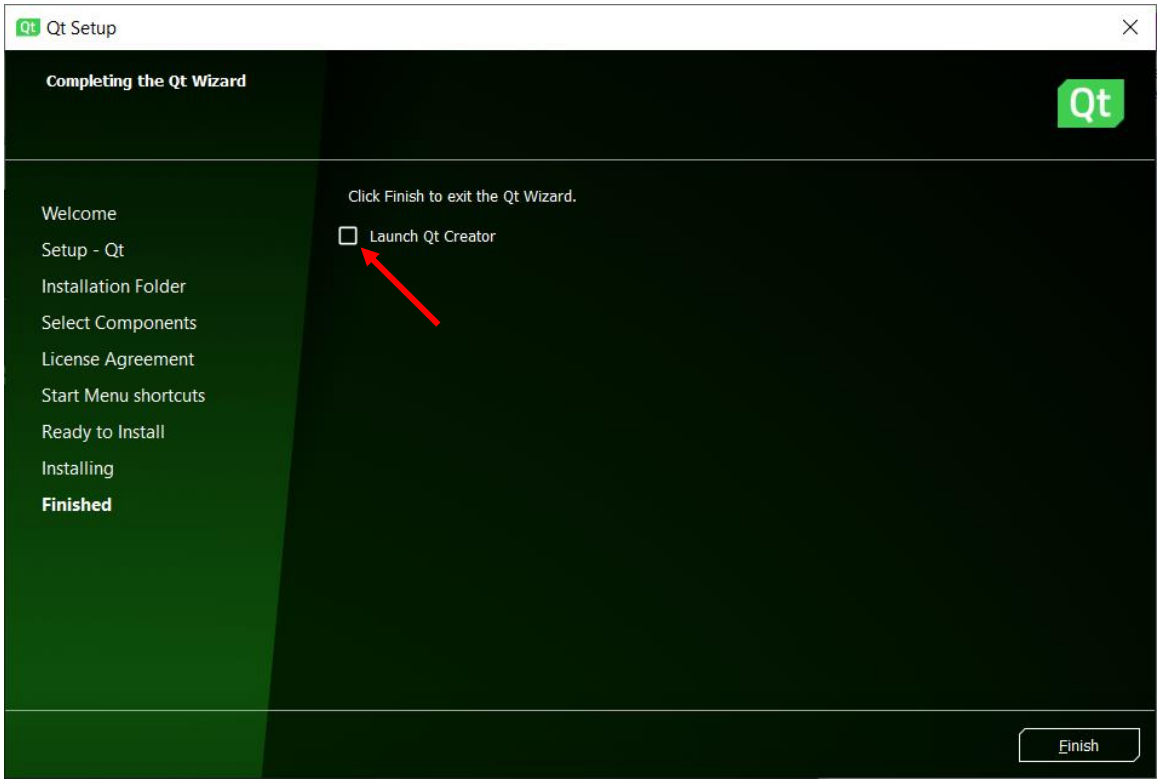


There are three things that must be selected on this page – *Common Files*, the *NXP i.MX RT1170 EVK* and the *ARM GCC 9 – Q2 2020* compiler.



After selecting them, press Next and on that page read and accept the License Agreement. Continue clicking through the guide to complete the installation.

There is an option on the last page of the installation wizard to launch QtCreator. Make sure to unselect it as there are some additional patching to do before launching QtCreator for the first time.



2.2 Add support for Embedded Artists iMX RT Developer's Kit

The iMX RT Developer's Kit was (at the time of writing this document) not included in the installer and must be installed separately. Start by downloading the platform-bsp archive from http://imx.embeddedartists.com. It will have a name like this:

qtformcus-platform-bsp-eaimxrt1176-freertos-2.0.0-<date>.7z

Before unpacking the archive, make sure that this folder exists on your computer: c:\Qt\QtMCUs\2.0.0\, it will be referred to as <QT_DIR> later in this document. If that folder does not exist, then **abort the patching** and look in section 2.1 for instructions on how to install the correct version.

Open the archive and extract all files into <QT_DIR>. A couple of files in the installation will be overwritten – please accept it when prompted during the extraction.

2.3 Downloading the SDK

Embedded Artists has published a version of the NXP SDK that has already been patched to work with the *iMX RT Developer's Kit*. The file can be downloaded from http://imx.embeddedartists.com and will have a filename like

`ea<cpu>_sdk_<version>_<date>.zip`

Pick version 2.10.1 or later.

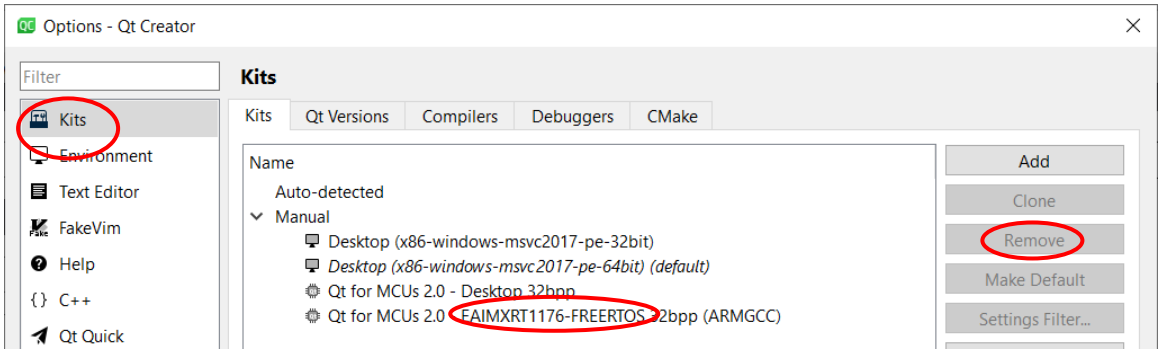
Unpack the archive somewhere with a short path. One suggestion is to create a folder with the same name as the archive in the root of the C:\ drive – e.g., for the iMX RT1176 Developer's Kit it could be c:\eaimxrt1176_sdk_2_10_1\. This folder will be referred to as <SDK_DIR> later in this document.

2.4 NXP MCUXpresso IDE

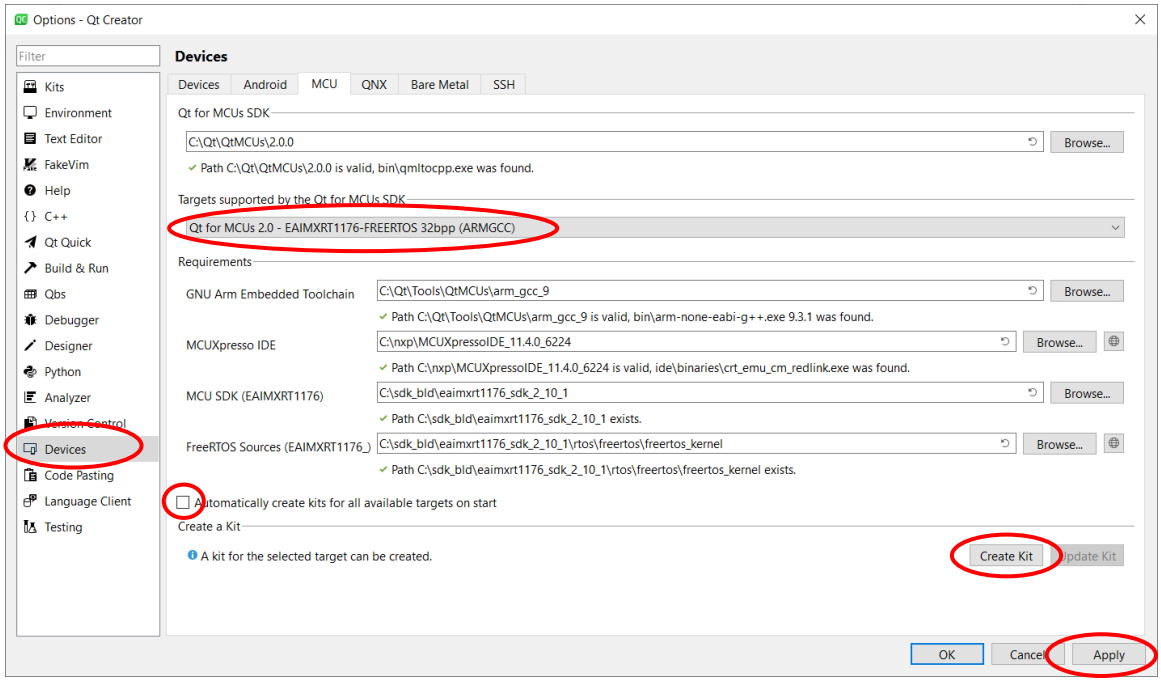
Qt for MCUs rely on an installation of NXP's MCUXpresso IDE for flashing the software onto the hardware. Download and install http://www.nxp.com/mcuxpresso/ide . After completing the installation, use Windows Explorer to find the location of the installation. If you made no changes during the installation, it should be under c:\nxp – for example version 11.4 installs under c:\nxp\MCUXpressoIDE_11.4.0_6237\. Remember this location, it will be referred to as <MCUX_DIR> later in this document.

2.5 Setting up Qt Creator

Start Qt Creator and select the Tools->Options menu. Select Kits group in the left side and if there are any kits in the list with EAIMXRT1176 in the name (one is shown here) then select it/them and click the Remove button.



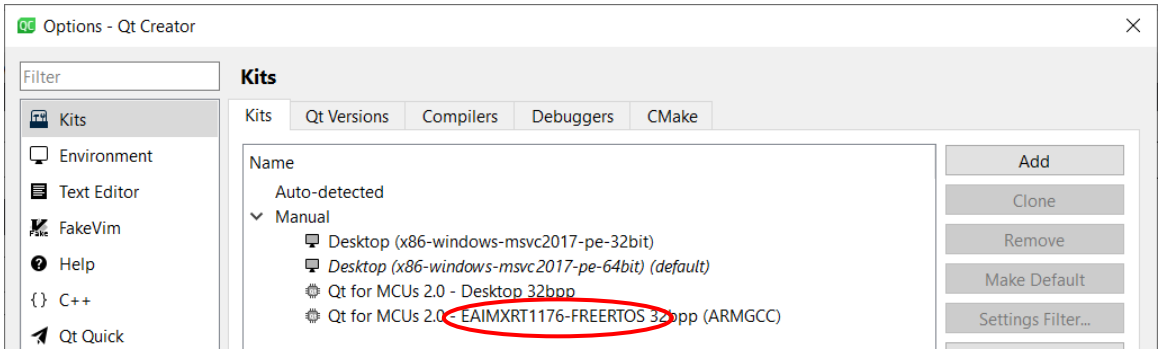
When all EAIMXRT1176 kits have been removed switch to the Devices group in the left side.



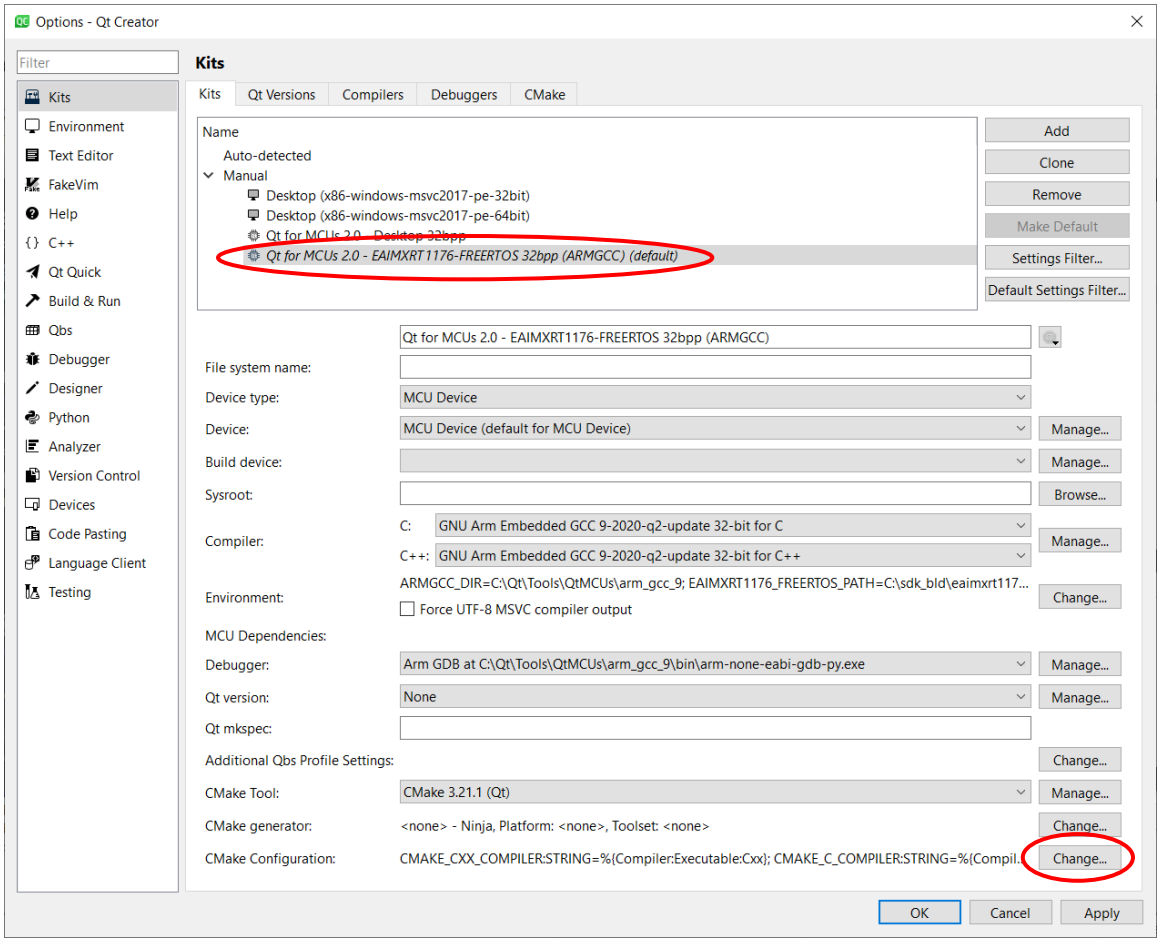
Start by selecting the EAIMXRT1176 target from the dropdown list – you can select either the ARMGCC or IAR. That should fill in the fields for the *GNU Arm Embedded Toolchain* but if it does not then browse to the location as shown in the image below. The warning about the path can be ignored.

Field	Comment
GNU Arm Embedded Toolchain	Only available if you select the ARMGCC target. It should be correct by default, and it should point to the ARMGCC version that was installed by the Qt installer.
IAR ARM Compiler	Only available if you select the IAR target. You must have installed the IAR compiler yourself (not covered by this guide) and make sure that the license is up to date as you will get some weird build errors otherwise. Use the Browse button to locate the installation dir.
MCUXpresso IDE	Browse to the folder where you installed MCUXpresso IDE in section 2.4 <MCUX_DIR>
MCU SDK	Browse to the folder where you unpacked the SDK in section 2.3 , <SDK_DIR>
FreeRTOS Sources	Browse to the <SDK_DIR>\rtos\freertos\freertos_kernel sub folder
Automatically create kits	Make sure the checkbox is not selected/marked as it will mess up your settings every time you start QtCreator

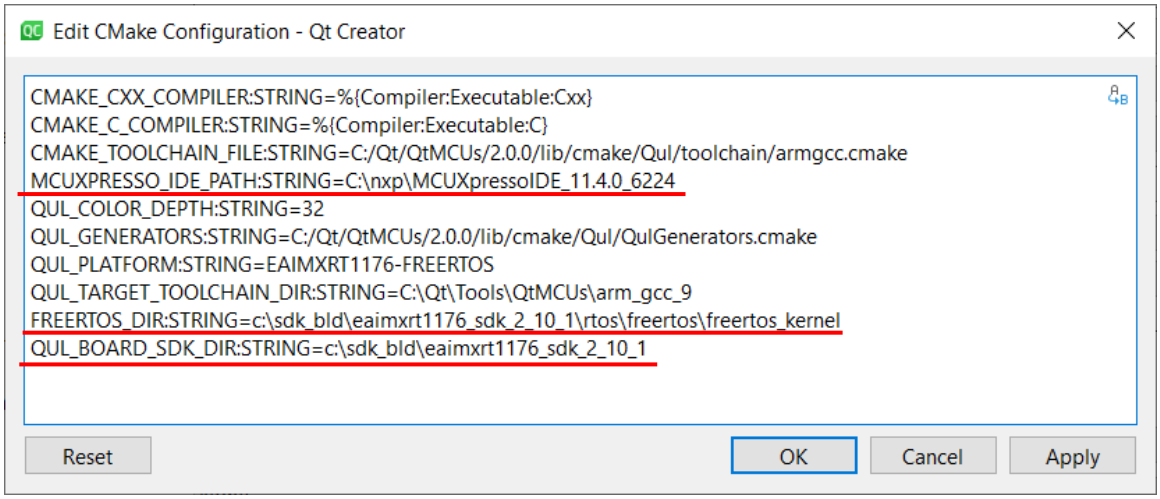
Press the *Apply* button to save the changes so far and then the *Create Kit* button to create the actual kit. After that go back to the *Kits* group in the left side where there will be an EAIMXRT1176 entry for the new kit like this:



Select the kit to bring up more settings:



Click the *Change* button for the *Cmake Configuration* to bring up this dialog:

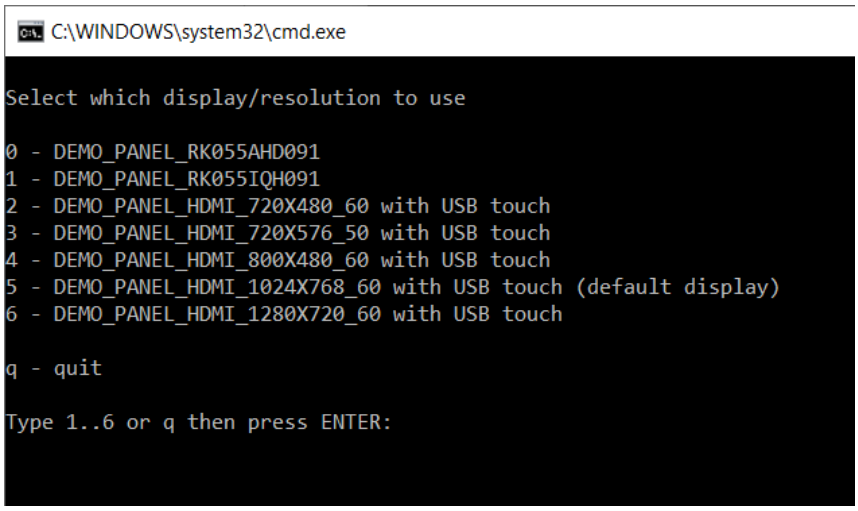


The three configuration flags above must be present – if they are not then add them. The values for the three options are the same as was used earlier in the *Devices* dialog: <MCUX_DIR>, <SDK_DIR>/rtos/freertos/freertos_kernel, <SDK_DIR>.

Close this dialog with the OK button and then do the same for the main Options window. The configuration of Qt should now be completed.

3 Select Display/Resolution to use

The iMX RT Developer's Kit supports several display options. The Qt for MCUs environment has been prepared for a number of these options but only one can be active at a time. To select which option to use, go to <QT_DIR>\lib\ and double click *select_display.bat* to bring up this menu (content may be different):



Enter a number to change to that display or press q to exit without making any changes.

The two displays at the top are NXP's 5.5 inch MIPI-DSI displays with touch support (I2C).

All the HDMI resolutions **should** work on most HDMI displays. The USB touch support has been tested on these two displays:

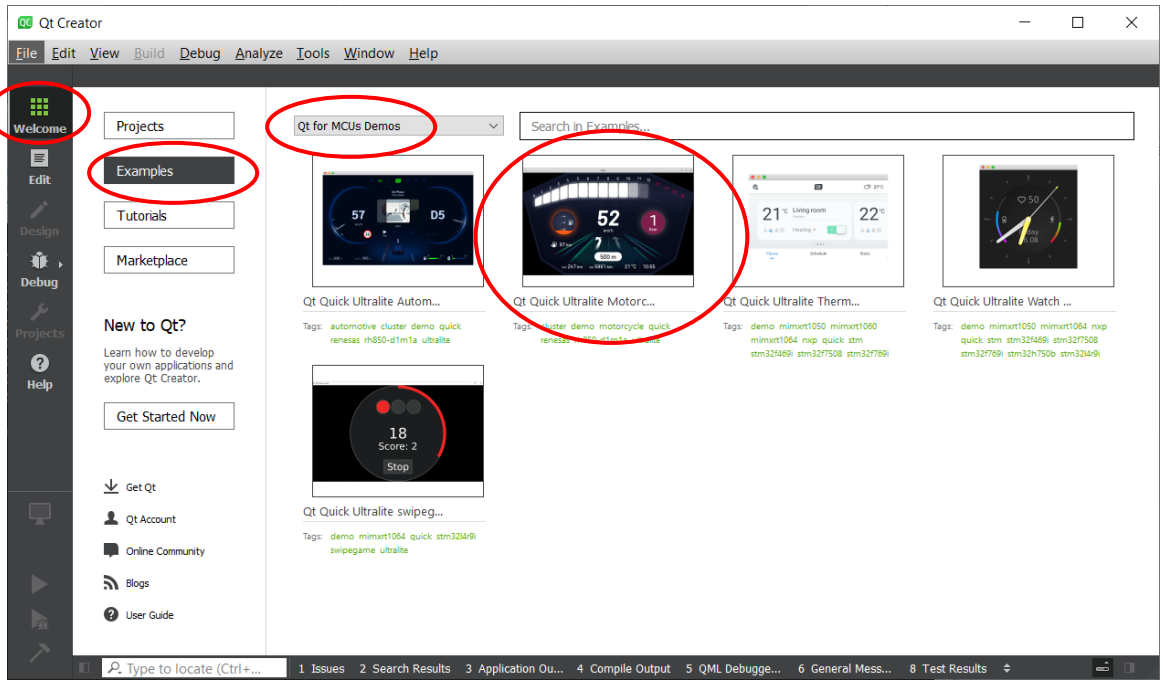
- Embedded Artists 7 inch HDMI Display Kit (EAD00363):
  https://www.embeddedartists.com/products/7-inch-hdmi-display-kit/

- NewHaven NHD-7.0-HDMI-N-RSXN-CTU:
  https://www.newhavendisplay.com/nhd70hdminrsxnctu-p-9552.html

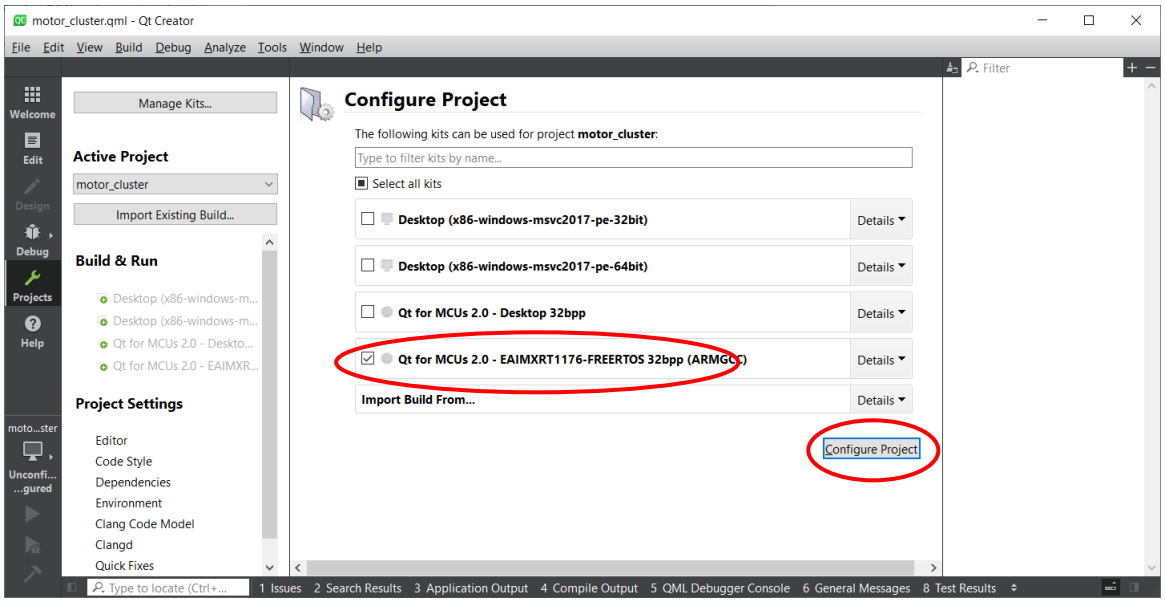
If you are using another HDMI display with USB touch then you will have to update the driver to support it, see section 7.2

4 Build Examples/Demos in Qt Creator

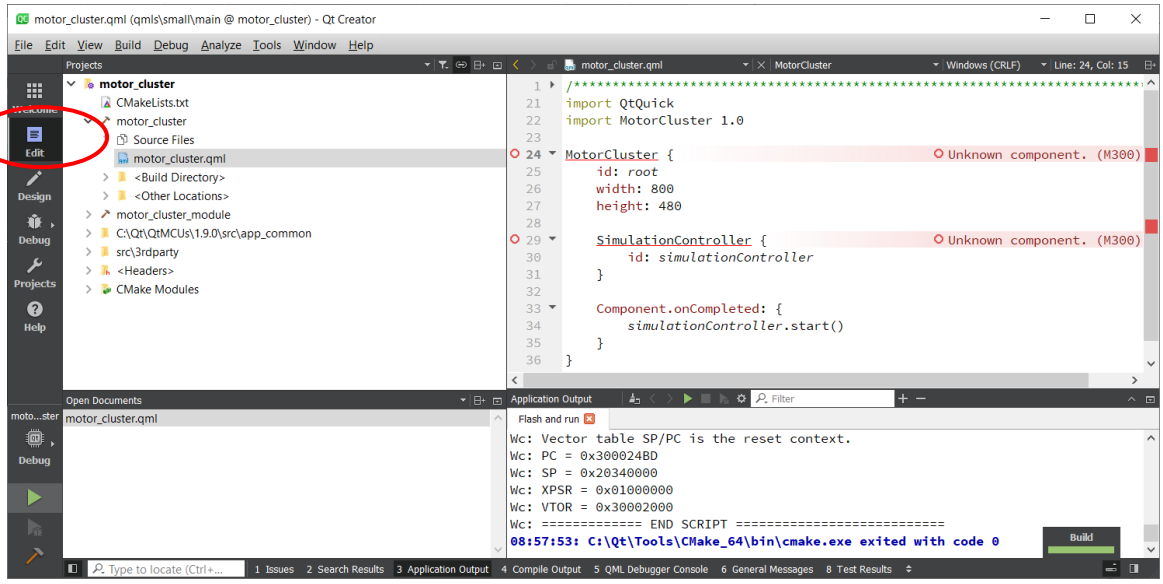
Qt Creator separates Demos and Examples, but both can be found in Qt Creator by clicking on the *Welcome* button and then *Examples*. From there you can use the dropdown menu to switch between the two.



Start by clicking the *Qt Quick Ultralite Motor Cluster demo*. It will bring up a dialog with information about what the demo does, its files and other useful information but for now we focus on the main window.



Select the Kit that you created during installation, it has EAIMXRT1176 in the name, and then press the Configure Project button. This brings up the Edit view.



Ignore the two errors. The project will compile anyway.

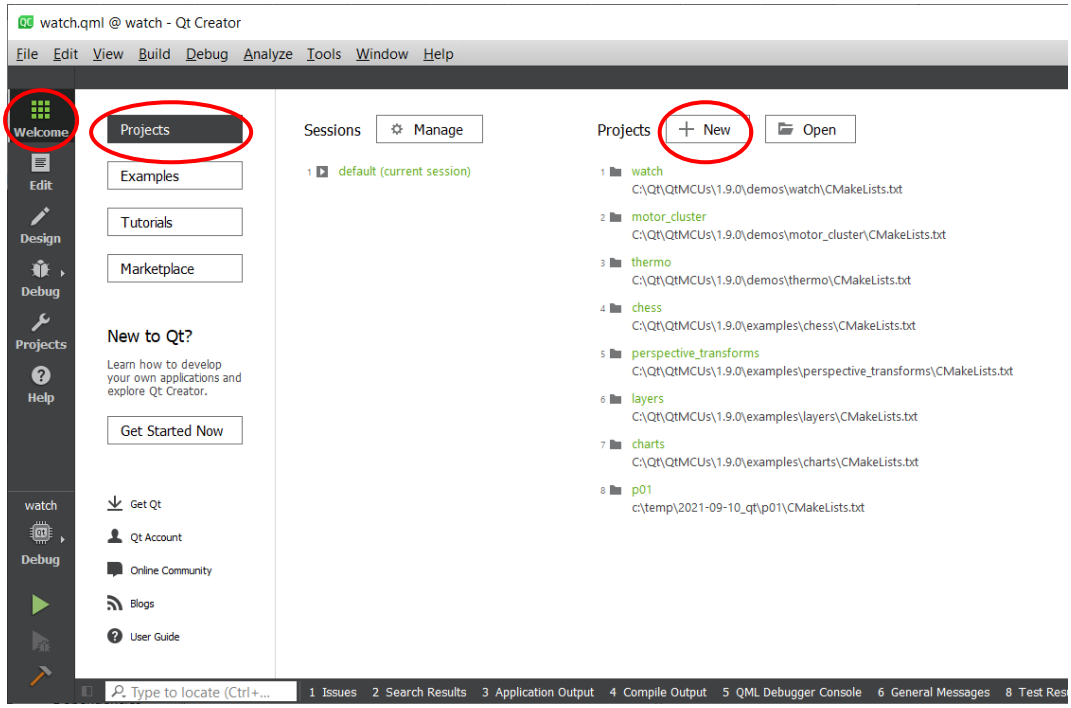
Now it is time to prepare and turn on the hardware. Follow the instructions here: https://www.embeddedartists.com/getting-started/

When the hardware is turned on, press the Run button to build and flash the demo.

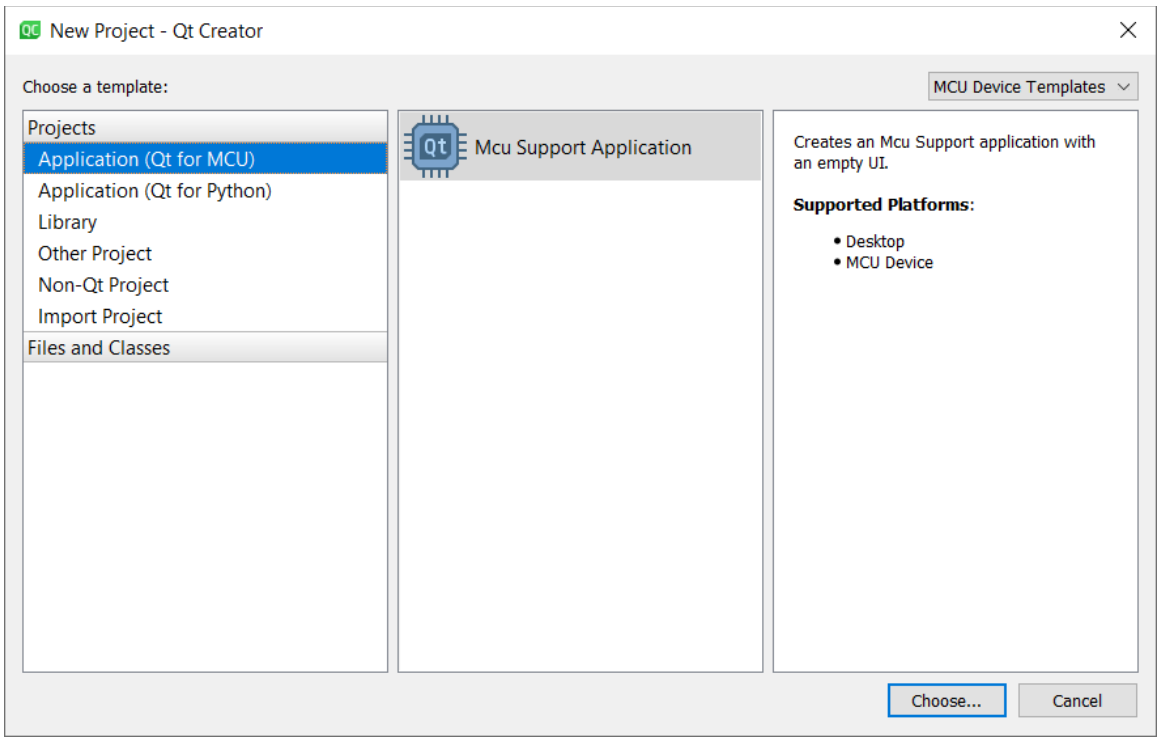
The display should now show something like this:

5 Create a New Project

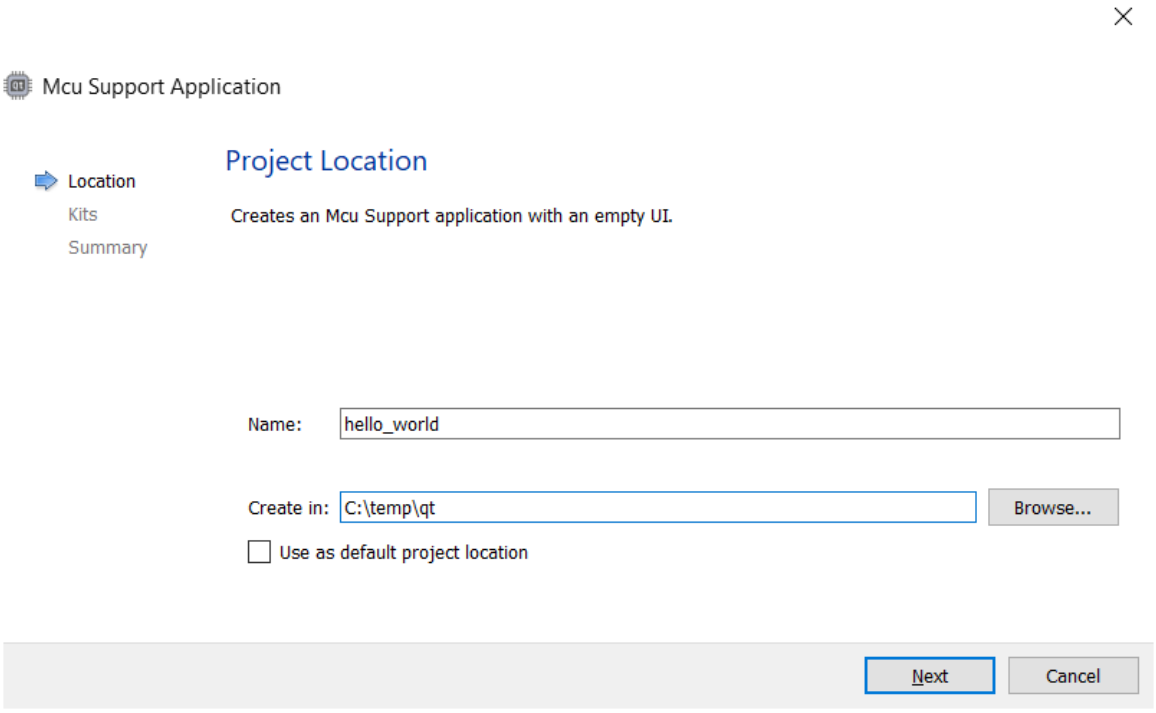
To create a new project in Qt Creator, start from the welcome screen and select *Projects* and then *New*.



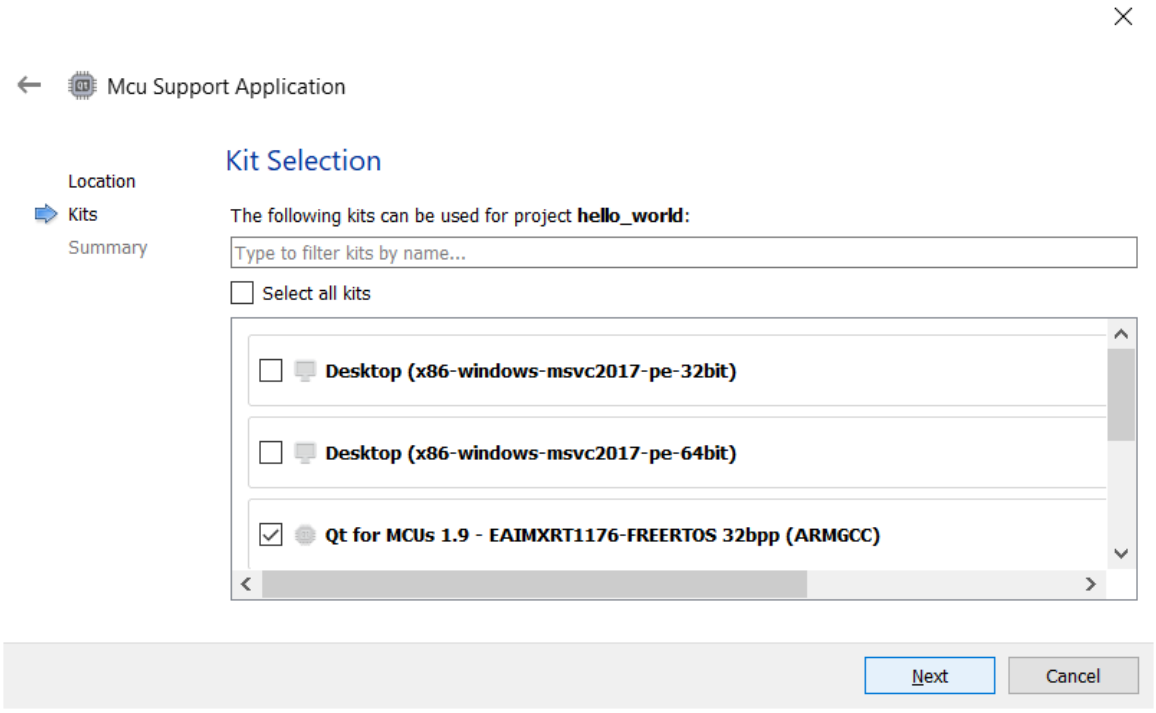
Select *Application (Qt for MCUs)* in the dialog



Select a name and location of the project



Select which Kit the project is for (use the EAIMXRT1176 one).



Press *Next* to complete the wizard and start working on your first program. Use the play button to build and flash the program onto the hardware.

6 More Information

This document is just a quick guide to get you started but it does not cover any of the details and possibilities of the complete Qt for MCUs framework.

We suggest visiting https://doc.qt.io/QtForMCUs/index.html for the most up-to-date information and having a look at the installed documentation found under c:\Qt\QtMCUs\2.0.0\docs\quickultralite\index.html (or <QT_DIR>\docs\quickultralite\index.html if not installed in the default location).

7 Known Issues

7.1 HDMI Resolution X is not Working

We have tested the HDMI resolutions available to the Qt for MCUs port (seen in section 3 above) on a number of different HDMI displays and the only resolution that we have found not to work is 800x480 on EAD00363 (7 inch HDMI Display kit from Embedded Artists) but there are likely other displays out there with other problems. It is impossible to test for all combinations.

7.2 Touch is not Working

At the time of writing this document we had implemented USB touch (single finger) for two different displays: (seen in section 3 above). Adding support for more fingers or more displays is a huge task.

> The functions below require detailed knowledge of your touch display and how it encodes its HID reports. This is out of scope of this document.

What we have done is left two hooks into the USB driver, allowing you to decode your own touch display. To do this first add this block to the main cpp file to declare the functions and get the skeleton code:

```
extern "C" {
   typedef bool (* VidPidFunction_t)( uint16_t vid, uint16_t pid );
   typedef bool (* ExtractFunction_t)( uint32_t vidpid,
       const uint8_t* buff, uint32_t len, uint16_t* x,
       uint16_t* y, bool* pressed );
   void BOARD_RegisterUSBTouchCallbacks(VidPidFunction_t vp,
ExtractFunction_t e);

    bool my_extractor(uint32_t vidpid, const uint8_t* buff,
                      uint32_t len, uint16_t* x,
                      uint16_t* y, bool* pressed)
    {
        Qul::PlatformInterface::log("in my_extractor\r\n");
        return false;
    }

    bool my_acceptor(uint16_t vid, uint16_t pid)
    {
        Qul::PlatformInterface::log("in my_acceptor\r\n");
        return false;
    }
}
```

And then add this line to main() to register the two callback functions:

```
int main()
{
    BOARD_RegisterUSBTouchCallbacks(my_acceptor, my_extractor);
    ...
}
```

Compile and run your program and you should only see the “in my_acceptor” printout as it prevents all touch displays by returning false.

Change the implementation of my_acceptor to this to allow your VID/PID (assuming VID=0x1234 and PID=0x5678):

```
bool my_acceptor(uint16_t vid, uint16_t pid)
{
    return (vid==0x1234 && pid==0x5678);
}
```

Compile and run your program and you should now see the "in my_extractor" printout as you touch the display. However, the coordinates are not reported to Qt for MCUs yet. To do that, modify the my_extractor function. The following assumes that a HID report from the display is 5 bytes long, first byte indicates a press followed by two bytes for the x coordinate and two bytes for the y coordinate.

```
bool my_extractor(uint32_t vidpid, const uint8_t* buff,
                  uint32_t len, uint16_t* x,
                  uint16_t* y, bool* pressed)
{
    if (len == 5) {
        *pressed = buff[0];
        uint32_t x_tmp = buff[1] + (buff[2]<<8);
        uint32_t y_tmp = buff[3] + (buff[4]<<8);

        /* Scale to fit display size. Assumes that touch
           screen reports in 10000x10000 resolution and
           the display is 1024x768 */
        *x = (uint16_t)((1024*x_tmp)/10000);
        *y = (uint16_t)((768*y_tmp)/10000);

        Qul::PlatformInterface::log("x=%u  y=%u\r\n", *x, *y);
        return true;
    }
    return false;
}
```

Returning true tells the driver to pass the values on to Qt for MCUs.

7.3 Example/Demo X is not Working

Some of the examples or demos will not work. At the time of writing these examples/demos were not working:

- Camera – The program will start and show a welcome screen but when the Start Camera button is pressed a black screen will appear instead of the camera image. This is because the connection to the camera hardware has not been ported for the iMX RT117x MCUs.
- Interrupt Handler – Shows nothing on the display and nothing in the terminal. This program has not been ported for the iMX RT117x MCUs.

In general, if you have problems with a demo/example, look at the archive with prebuilt binaries and the accompanying document. Test to flash one of those binaries to see if you get a different result or if that document has any updated status information.

8 Troubleshooting

If you experience problems with flashing or debugging have a look at the troubleshooting suggestions in *iMX RT Developer's Kit Program Development Guide*

9 Disclaimers

Embedded Artists reserves the right to make changes to information published in this document, including, without limitation, specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Customer is responsible for the design and operation of their applications and products using Embedded Artists' products, and Embedded Artists accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the Embedded Artists' product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. Customer is required to have expertise in electrical engineering and computer engineering for the installation and use of Embedded Artists' products.

Embedded Artists does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using Embedded Artists' products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). Embedded Artists does not accept any liability in this respect.

Embedded Artists does not accept any liability for errata on individual components. Customer is responsible to make sure all errata published by the manufacturer of each component are taken note of. The manufacturer's advice should be followed.

Embedded Artists does not accept any liability and no warranty is given for any unexpected software behavior due to deficient components.

Customer is required to take note of manufacturer's specification of used components, for example MCU, SDRAM and FLASH. Such specifications, if applicable, contains additional information that must be taken note of for the safe and reliable operation. These documents are stored on Embedded Artists' product support page.

All Embedded Artists' products are sold pursuant to Embedded Artists' terms and conditions of sale: http://www.embeddedartists.com/sites/default/files/docs/General_Terms_and_Conditions.pdf

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by Embedded Artists for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN EMBEDDED ARTISTS' TERMS AND CONDITIONS OF SALE EMBEDDED ARTISTS DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF EMBEDDED ARTISTS PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY THE CEO OF EMBEDDED ARTISTS, PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, NUCLEAR, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE.

Resale of Embedded Artists' products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by Embedded Artists

for the Embedded Artists' product or service described herein and shall not create or extend in any manner whatsoever, any liability of Embedded Artists.

This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

9.1 Definition of Document Status

Preliminary – The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. Embedded Artists does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information. The document is in this state until the product has passed Embedded Artists product qualification tests.

Approved – The information and data provided define the specification of the product as agreed between Embedded Artists and its customer, unless Embedded Artists and customer have explicitly agreed otherwise in writing.