# Remote Debugging with the GNU GDB Debugger using Qt Creator

MA35XX Series

江天文

2024/04/24

# Introduction

- Qt Creator is the de facto standard IDE for developing applications on embedded Linux.
- Qt Creator integrates with remote debugging facility through the GNU GDB Debugger and SSH client.

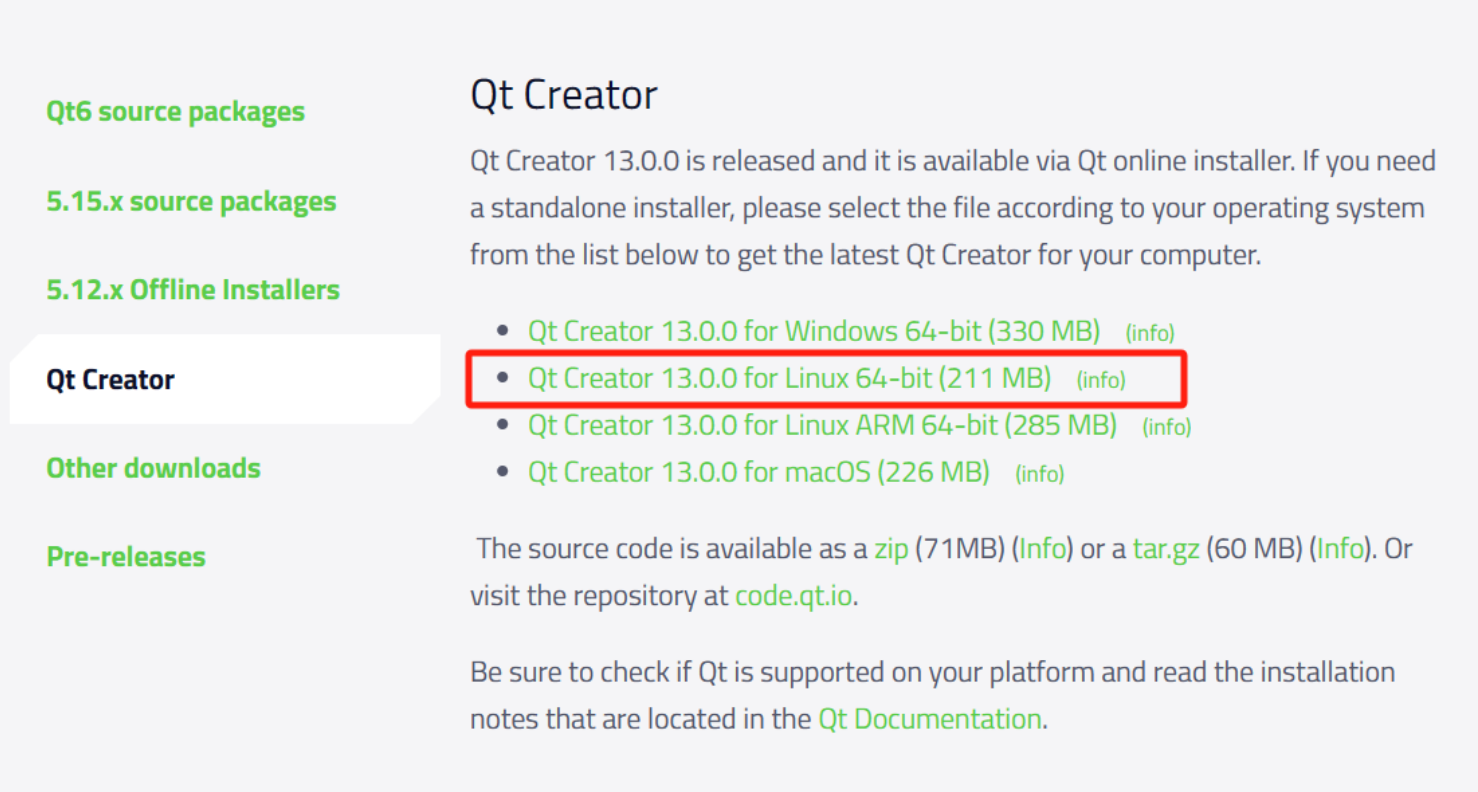
# Configure Buildroot

- Remote Debugging with Qt Creator requires the Buildroot configuration to meet development environment. Type ***make menuconfig*** to configure Buildroot, tailor the toolchain options and select the corresponding packages.

**→ Toolchain**
    [*] Build cross gdb for the host
        Python support (**Python 3**)
**→ Host utilities**
    [*] **host cmake**
**→ Target packages → Debugging, profiling and benchmark**
    [*] **gdb**
    -*- gdbserver
**→ Target packages → Networking applications**
    [*] **openssh**
    [*] **server**
    [*] **key utilities**
    [*] **rsync**

# Download Qt Creator Offline Installer

- At the time of writing the version of Qt Creator is 13.0.0.
- Browse the below site to fetch Qt Creator.
  https://www.qt.io/offline-installers



# Install Qt Creator

- Install package libxcb-cursor0
  $ sudo apt install libxcb-cursor0
- Change execution permission
  $ sudo chmod +x qt-creator-opensource-linux-x86_64-13.0.0.run
- Disconnect internet connection due to registration required during installation
- Install Qt Creator
  $ ./qt-creator-opensource-linux-x86_64-13.0.0.run

# Configure OpenSSH Server

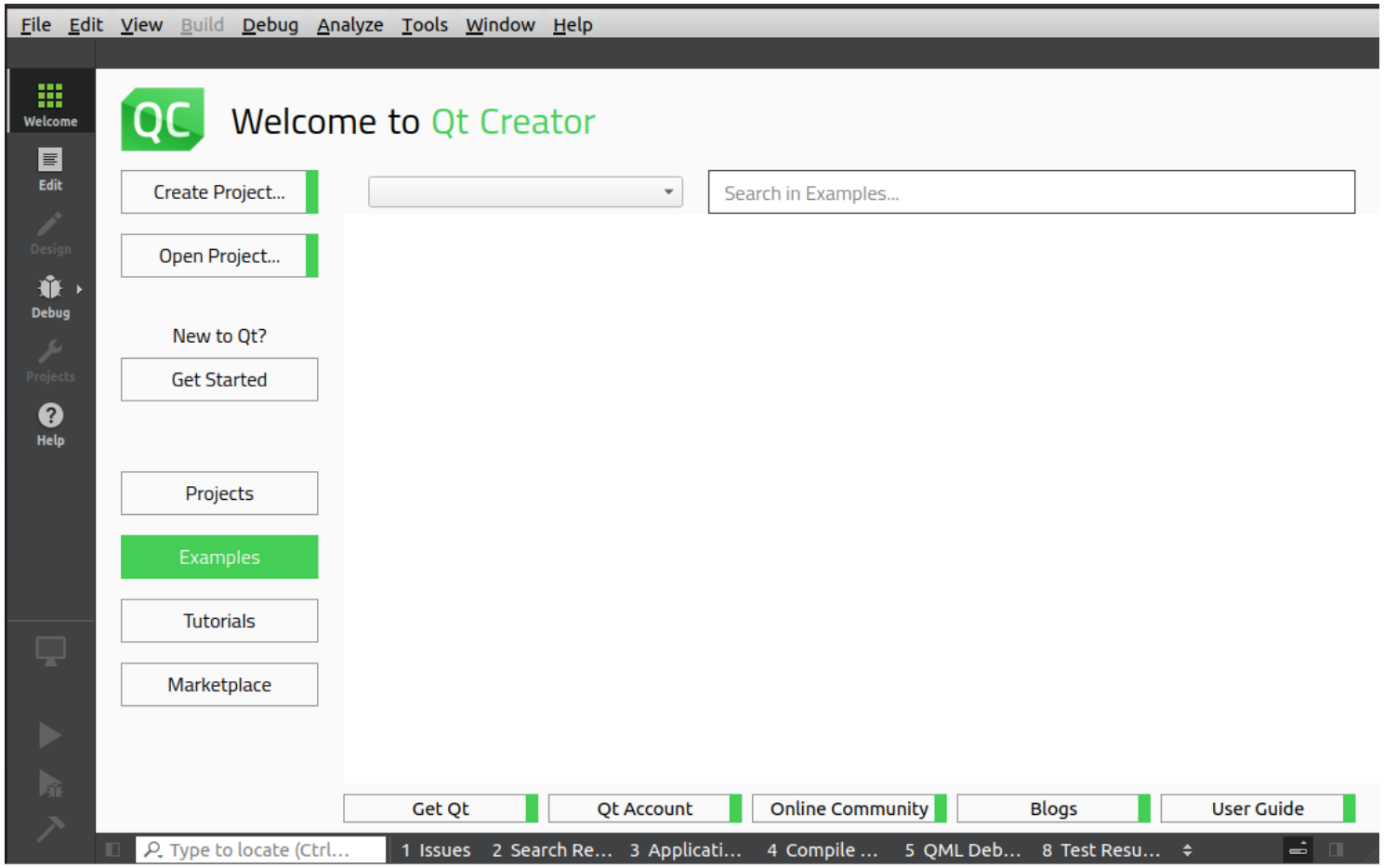
- Configure OpenSSH Server on target board.

```
# /etc/ssh/sshd_config

PermitRootLogin                 yes
PasswordAuthentication          yes
PermitEmptyPasswords            yes
```

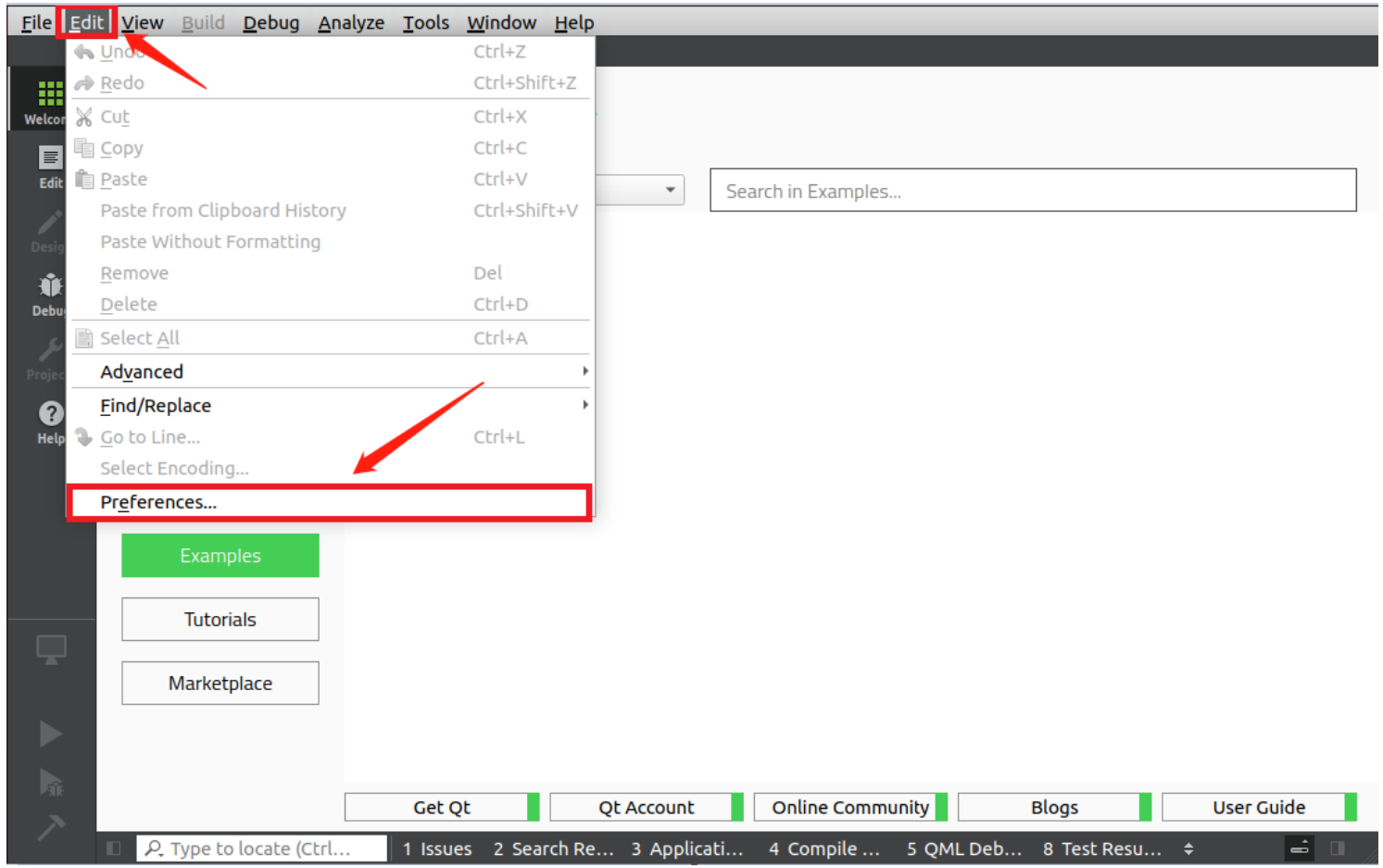
# Launch Qt Creator

- From Ubuntu application launch panel, find and launch Qt Creator.

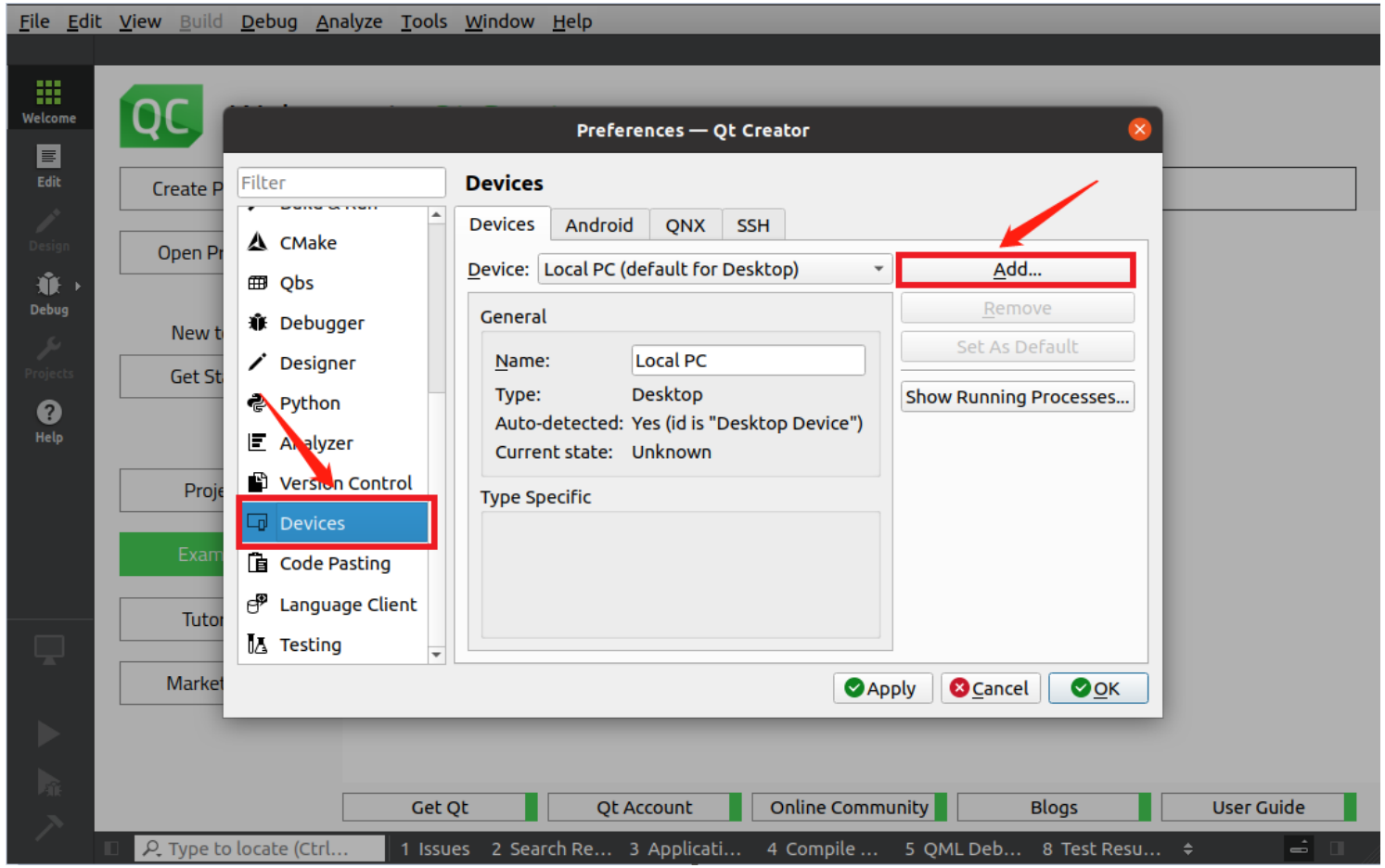
# Configure Qt Creator

- Click **Edit** → **Preferences**...



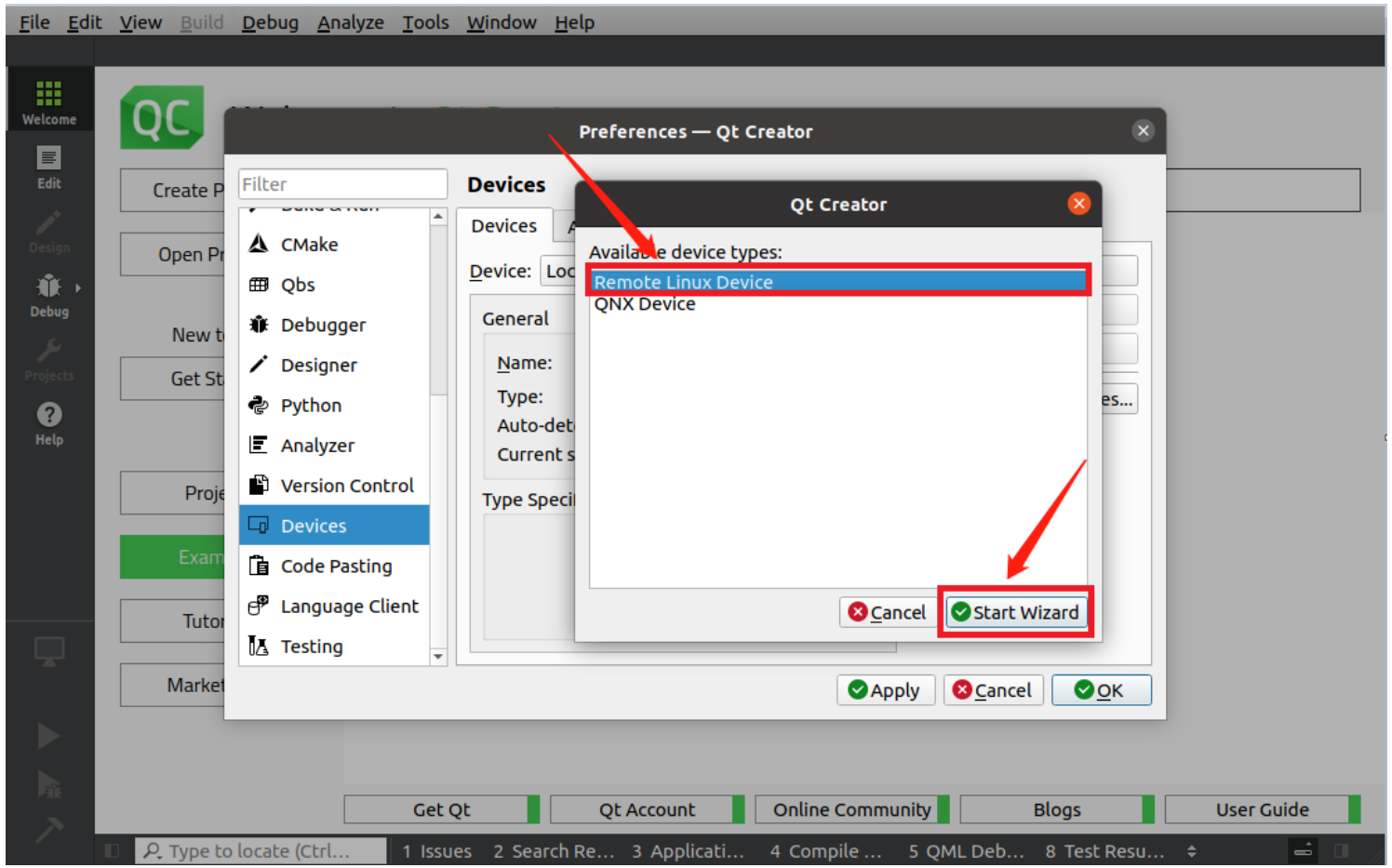
# Add Remote Devices

- From side list, click the **Devices** in dialog of **Preferences**, click **Add** to create a remote device.



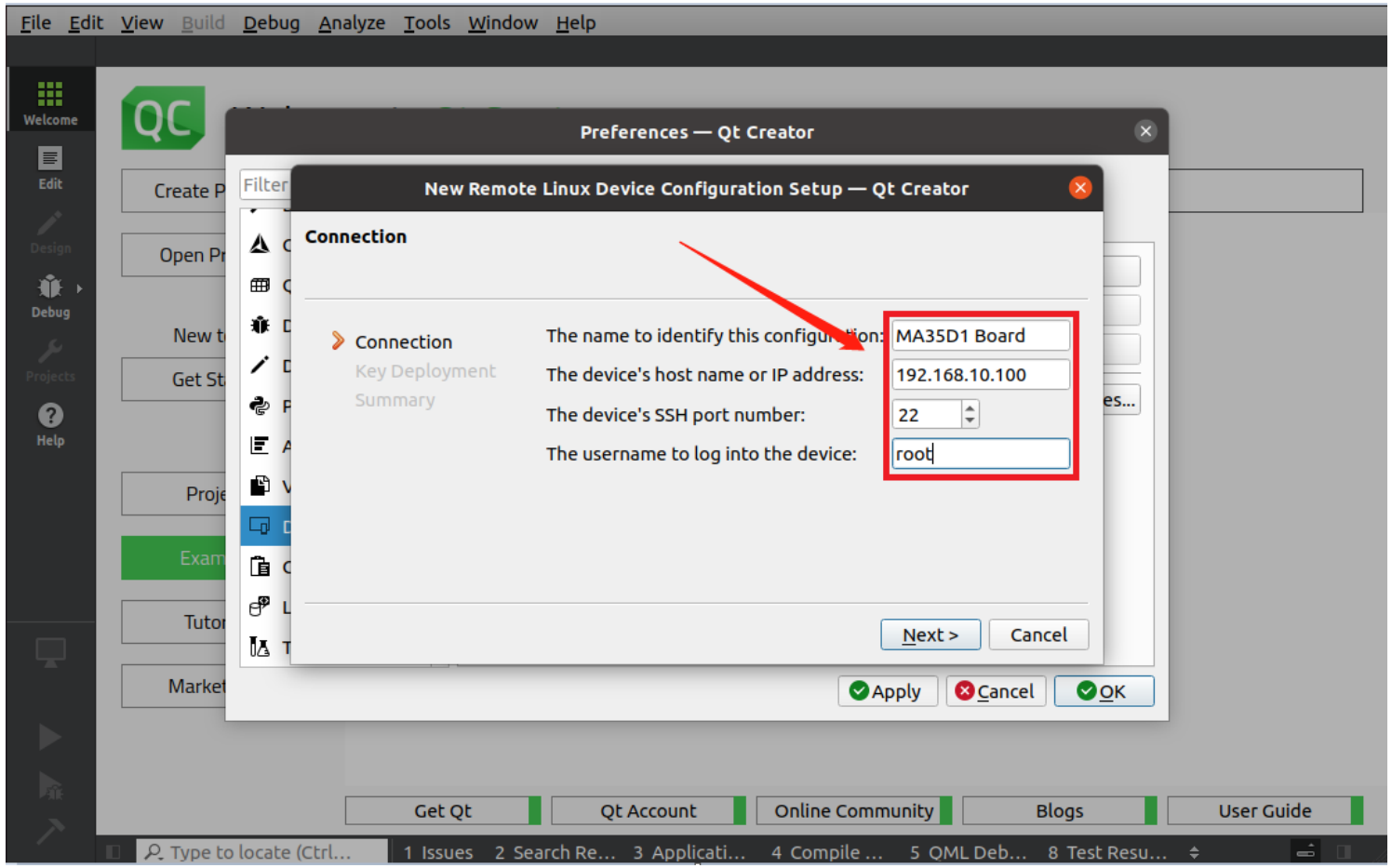
# Create Remote Linux Device

- Choose **Remote Linux Device** to **Start Wizard**



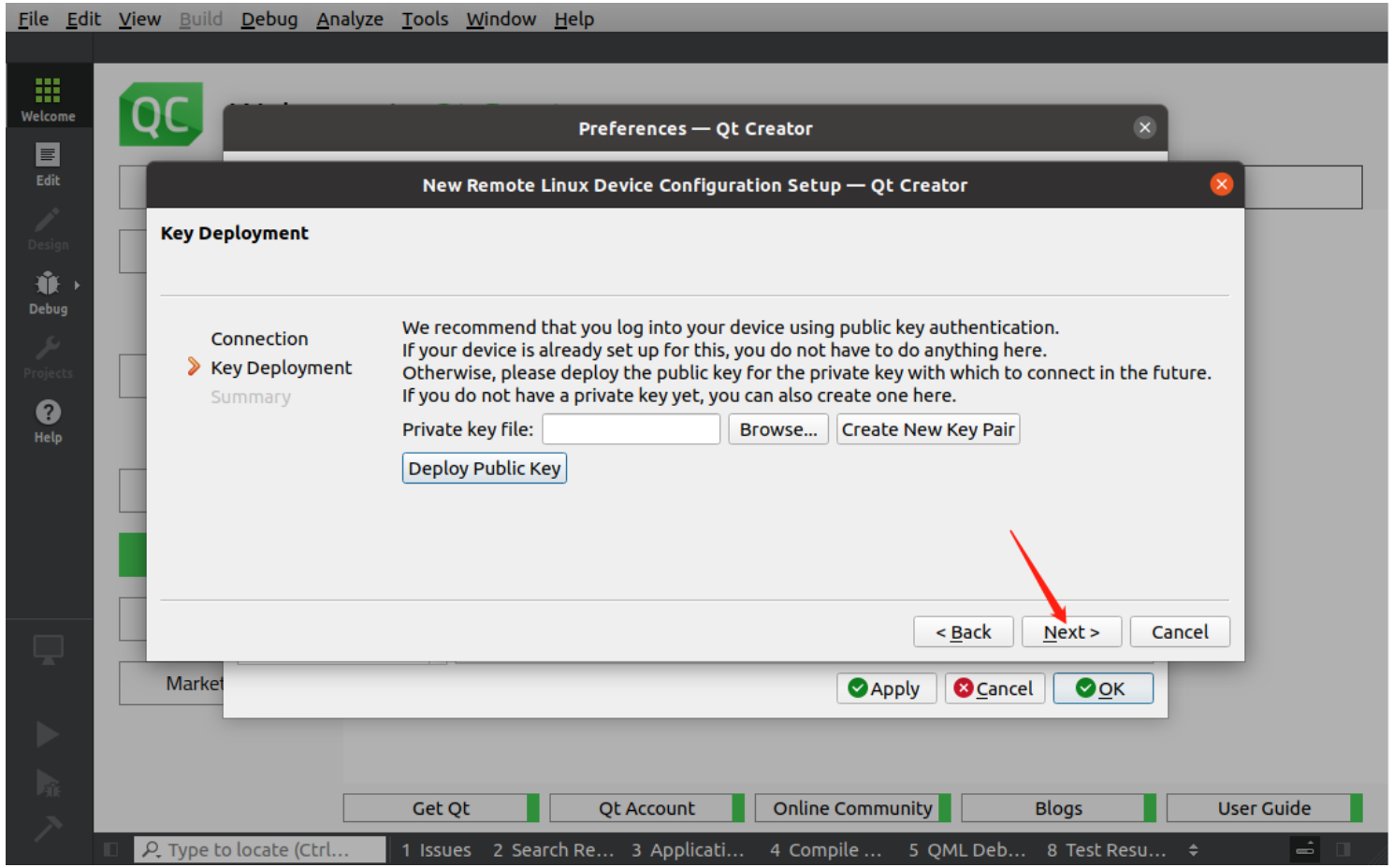
# Configure Remote Device

- Configuration name:
  **MA35D1 Board**
- IP address:
  **192.168.10.100**
- SSH port number:
  **22**
- username:
  **root**



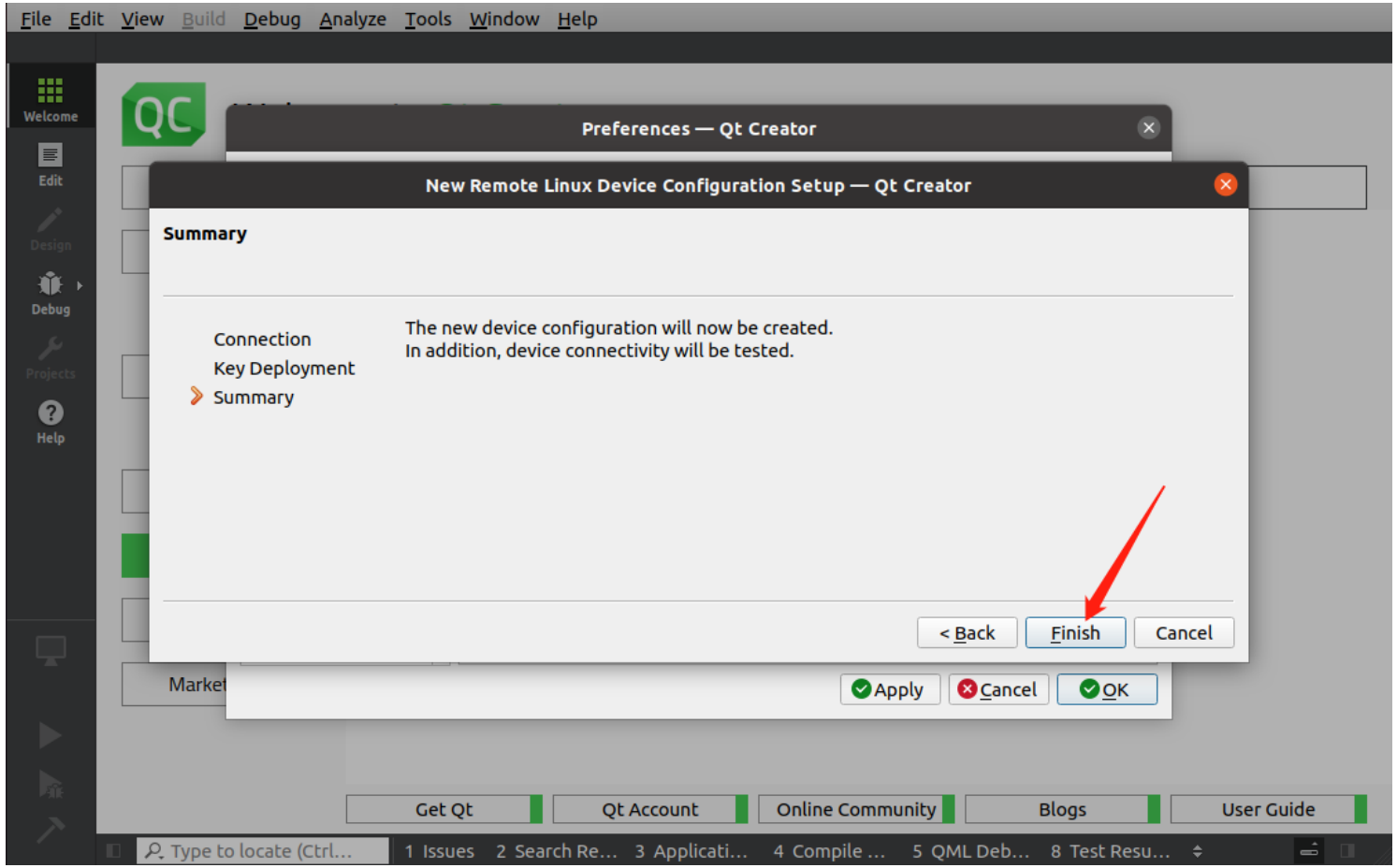
# Configure Remote Device

- Click **Next** to proceed



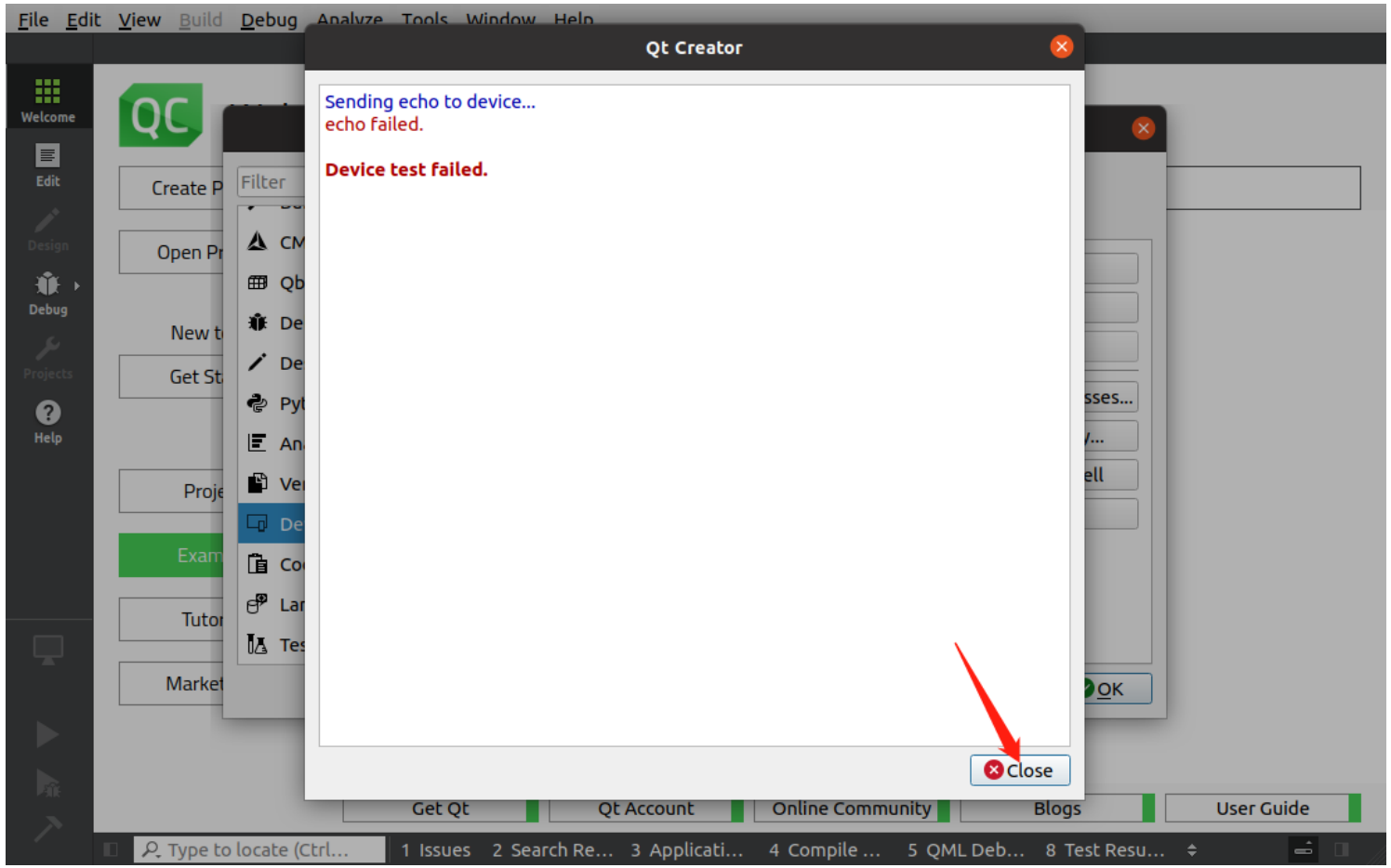
# Configure Remote Device

- Click **Finish**



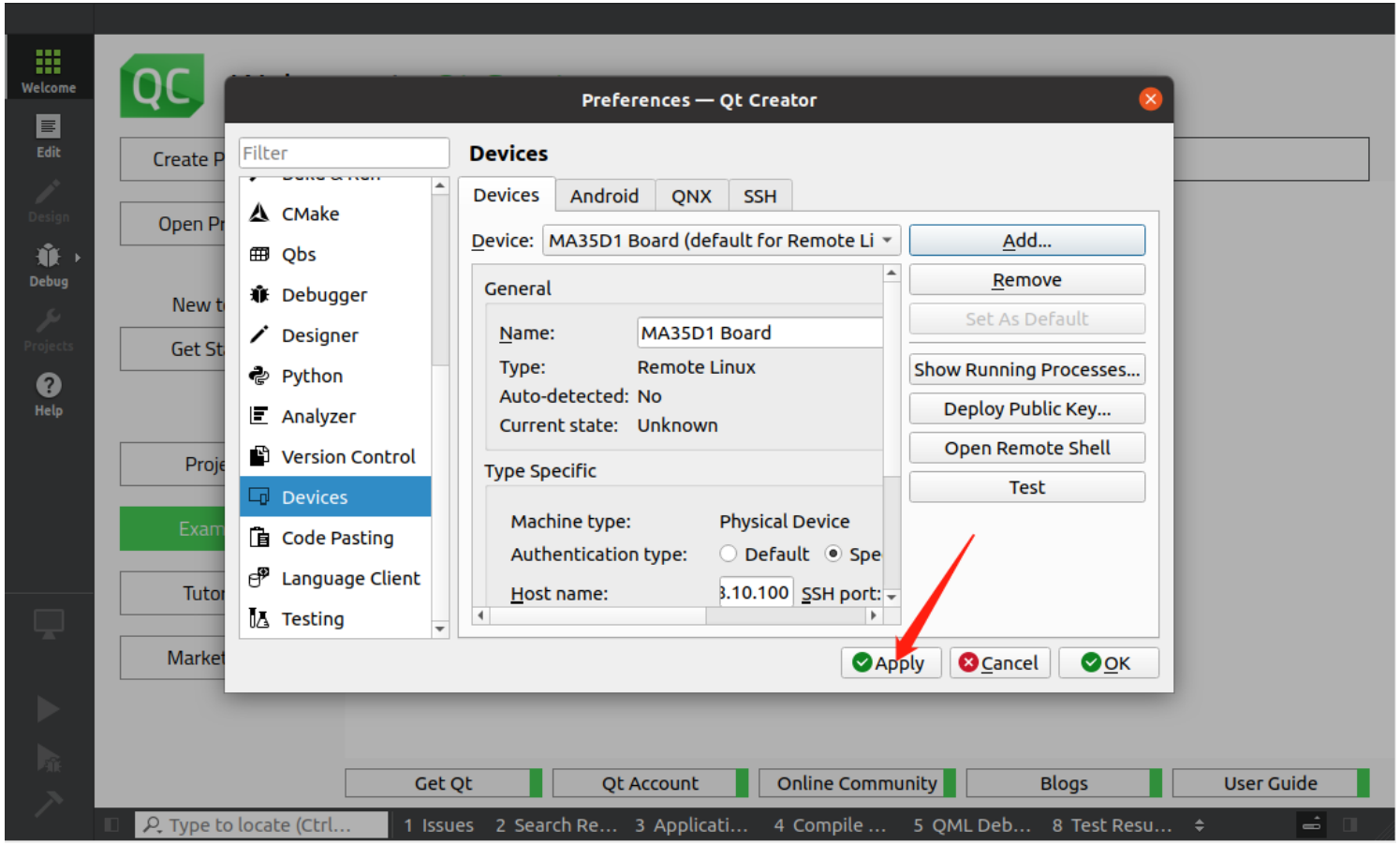
# Test SSH connection

- If SSH connection is established between PC and remote device, the dialog will show the result of test.



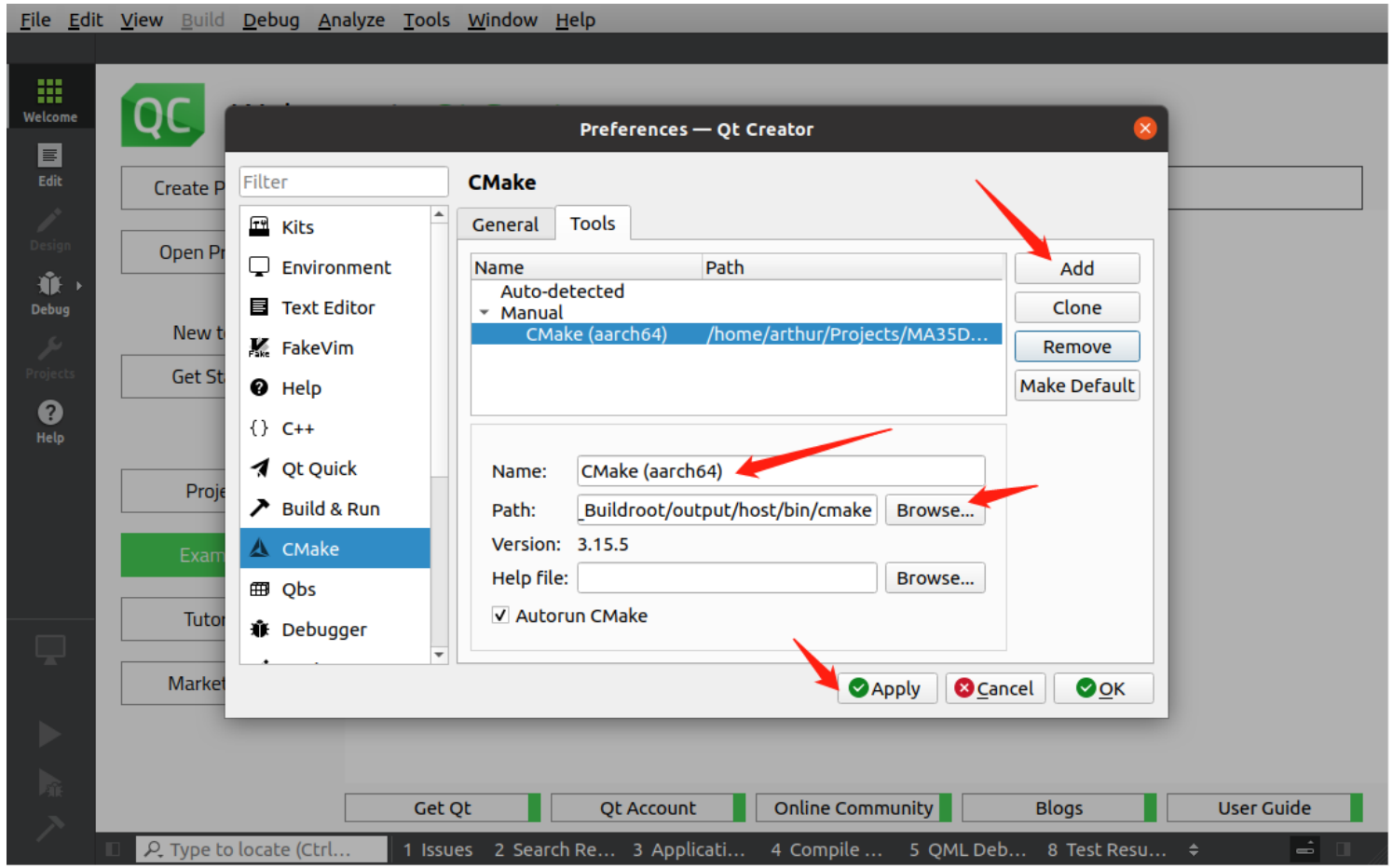
# Apply Remote Device

- Click **Apply** to complete remote device creation.



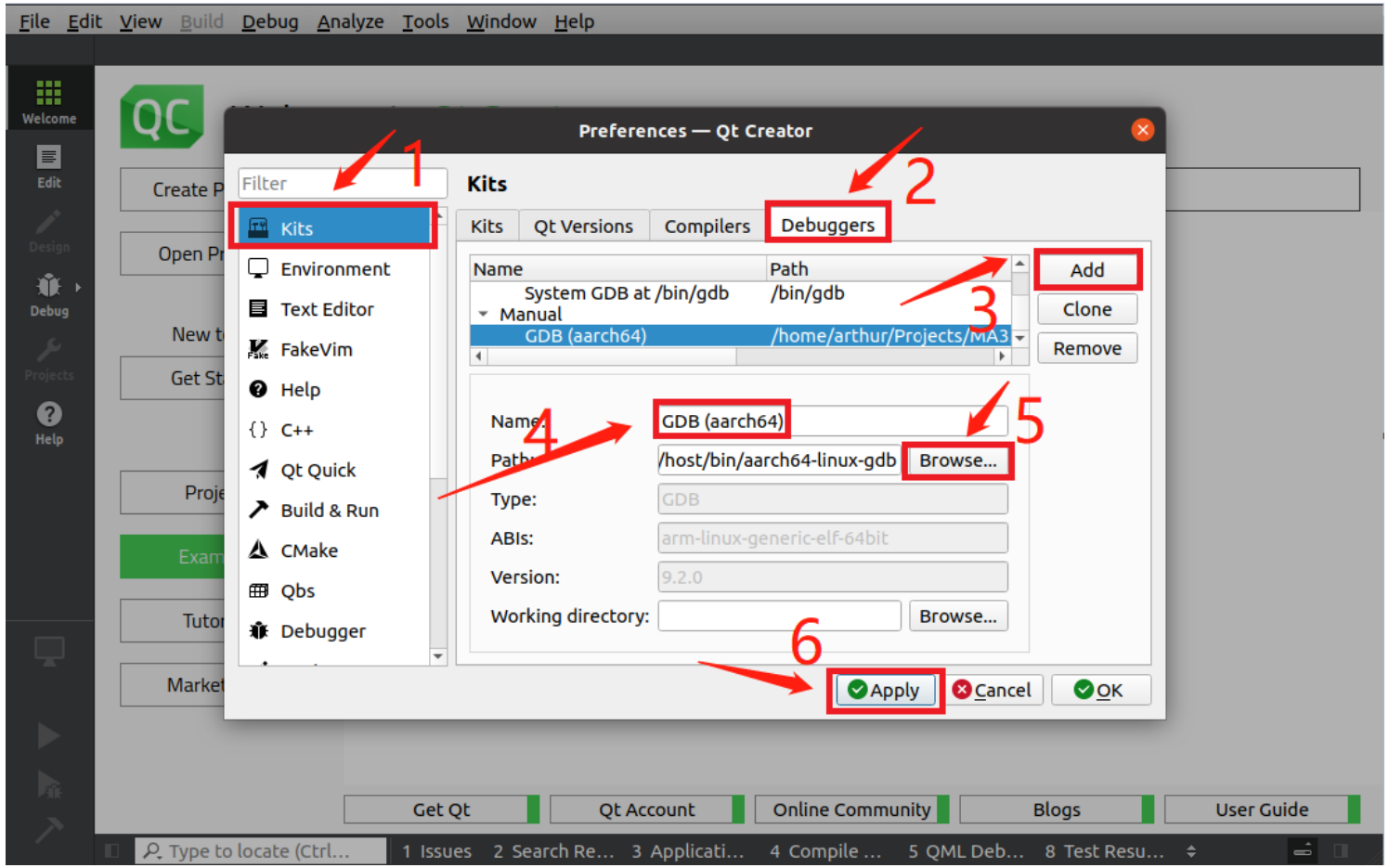
# Configure CMake

- In dialog of **Preferences**, choose **CMake** from left side panel, click Add to set CMake to ***${BR2_DIR}/output/host/bin/cmake***. Then click **Apply** to forward.



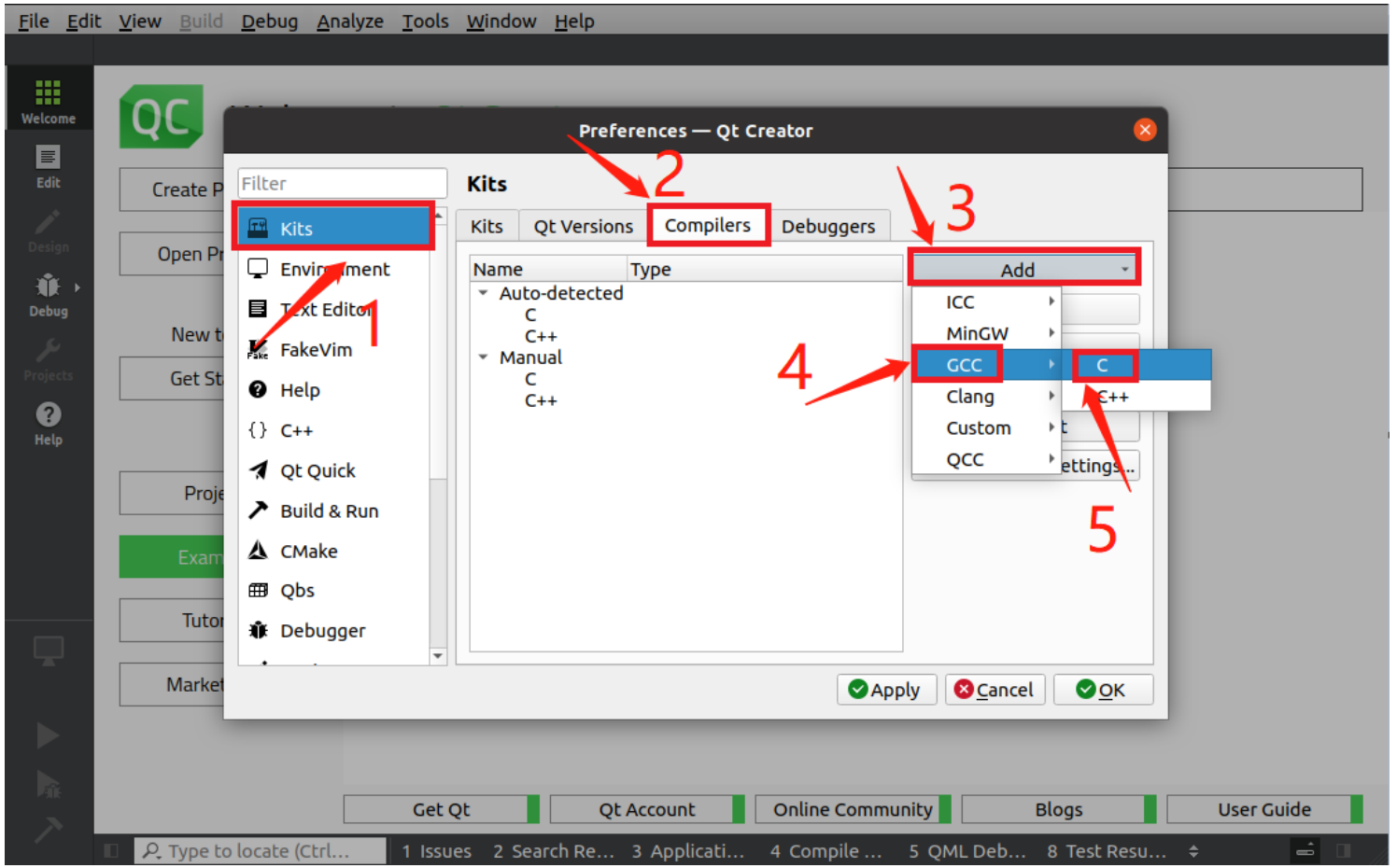
# Configure Kits (Debuggers)

- In dialog of **Preferences**, choose **Kits** from left side panel, Add **Debuggers** as ***${BR2_DIR}/output/host/bin/aarch64-linux-gdb***.



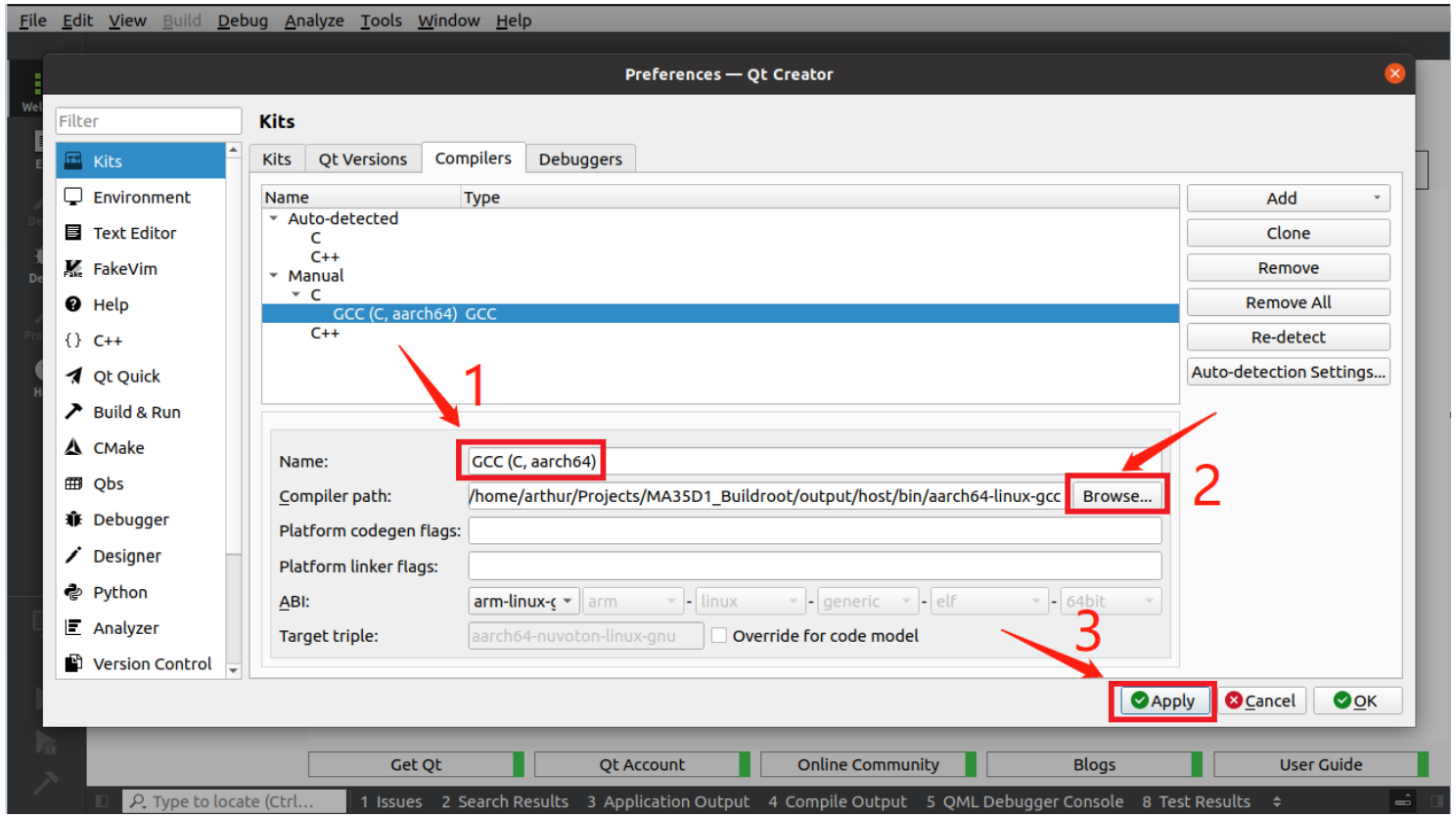
# Configure Kits (Compilers)

- In dialog of **Preferences**, choose **Kits** from left side panel, Add **Compilers (GCC: C)**



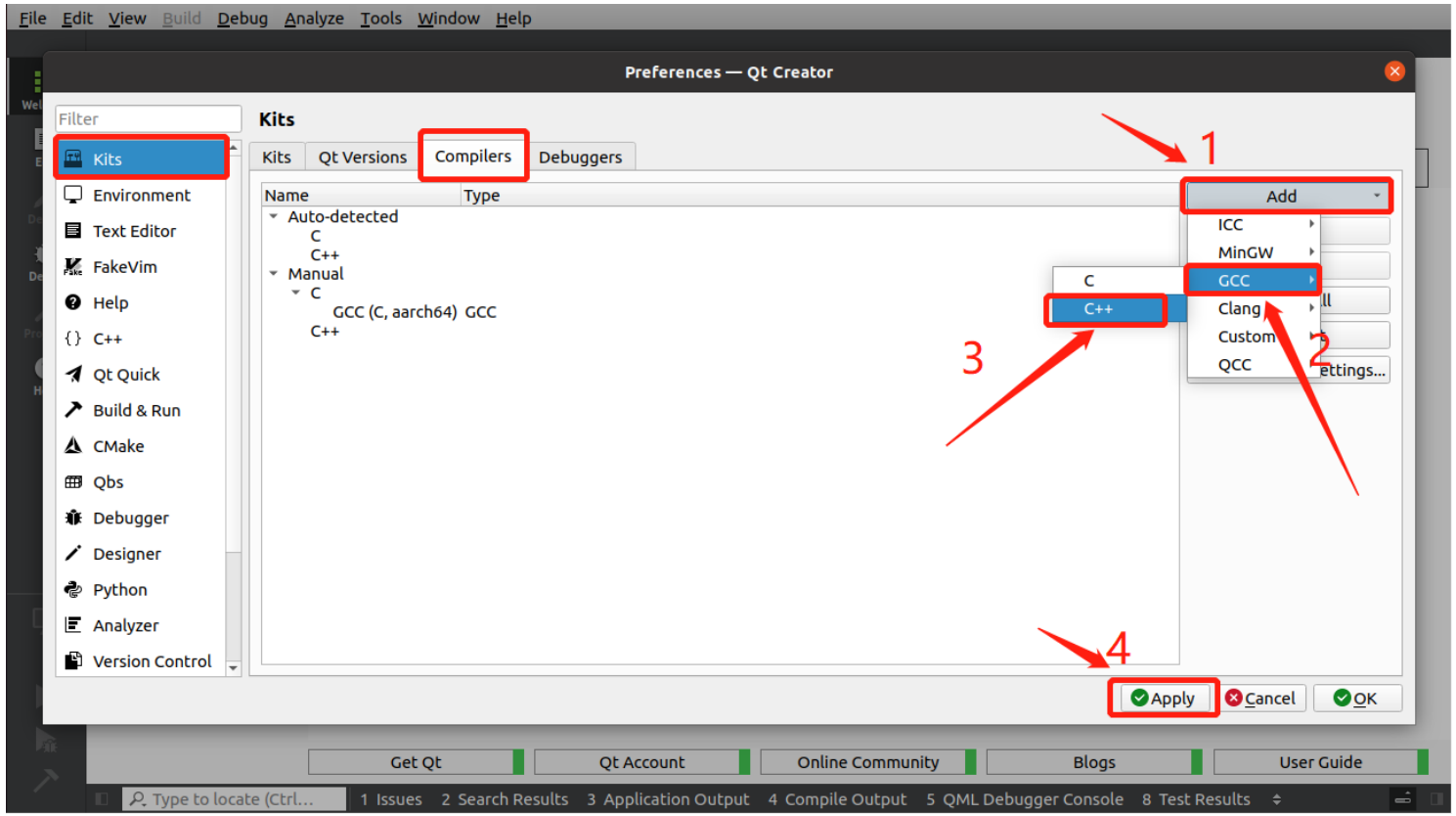
# Configure Kits (Compilers)

- Add **GCC C compiler**: ***${BR2_DIR}/output/host/bin/aarch64-linux-gcc***



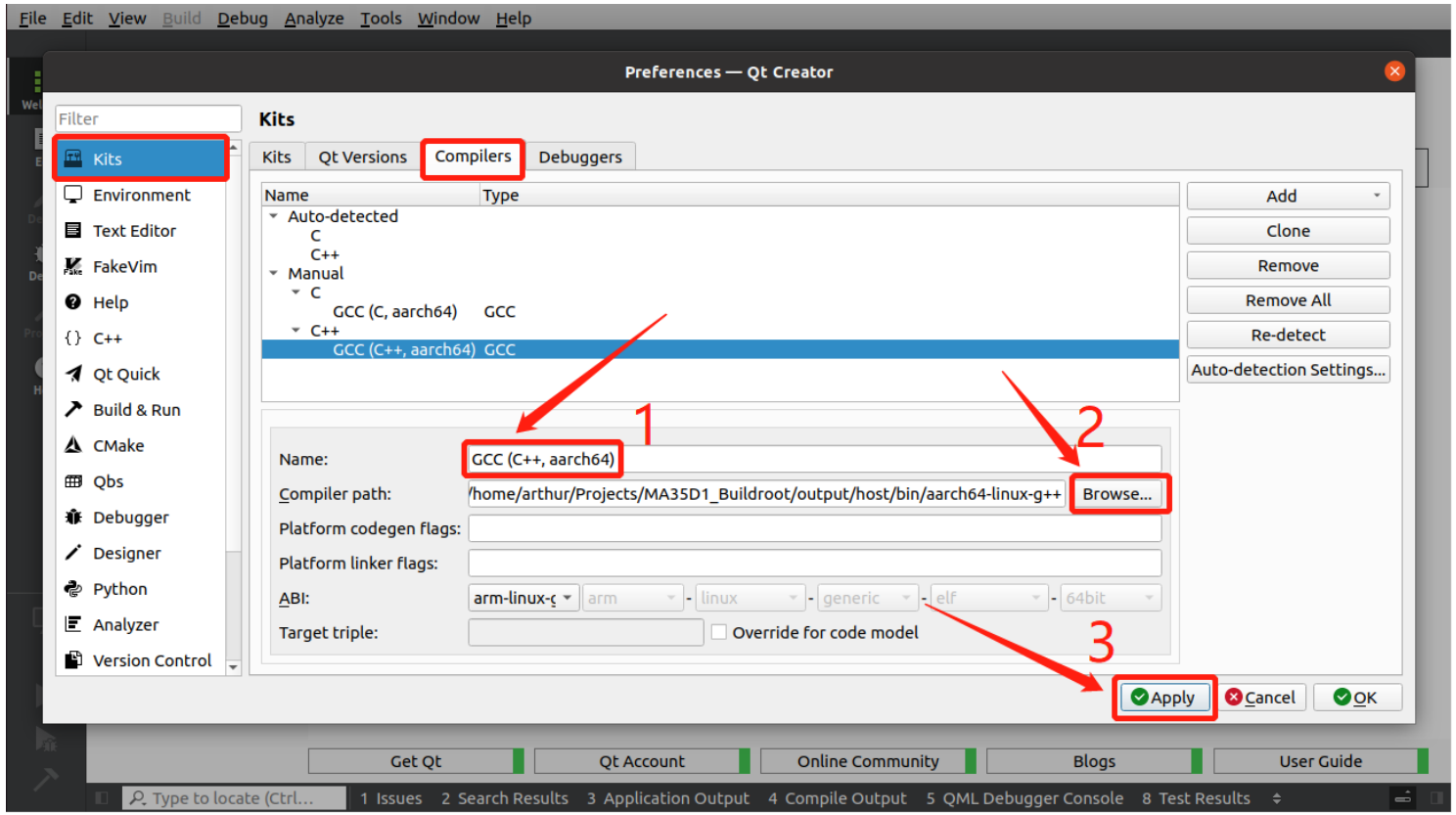
# Configure Kits (Compilers)

- Add **GCC C++ compiler**



# Configure Kits (Compilers)

- Add **GCC C++ compiler**: ***${BR2_DIR}/output/host/bin/aarch64-linux-g++***



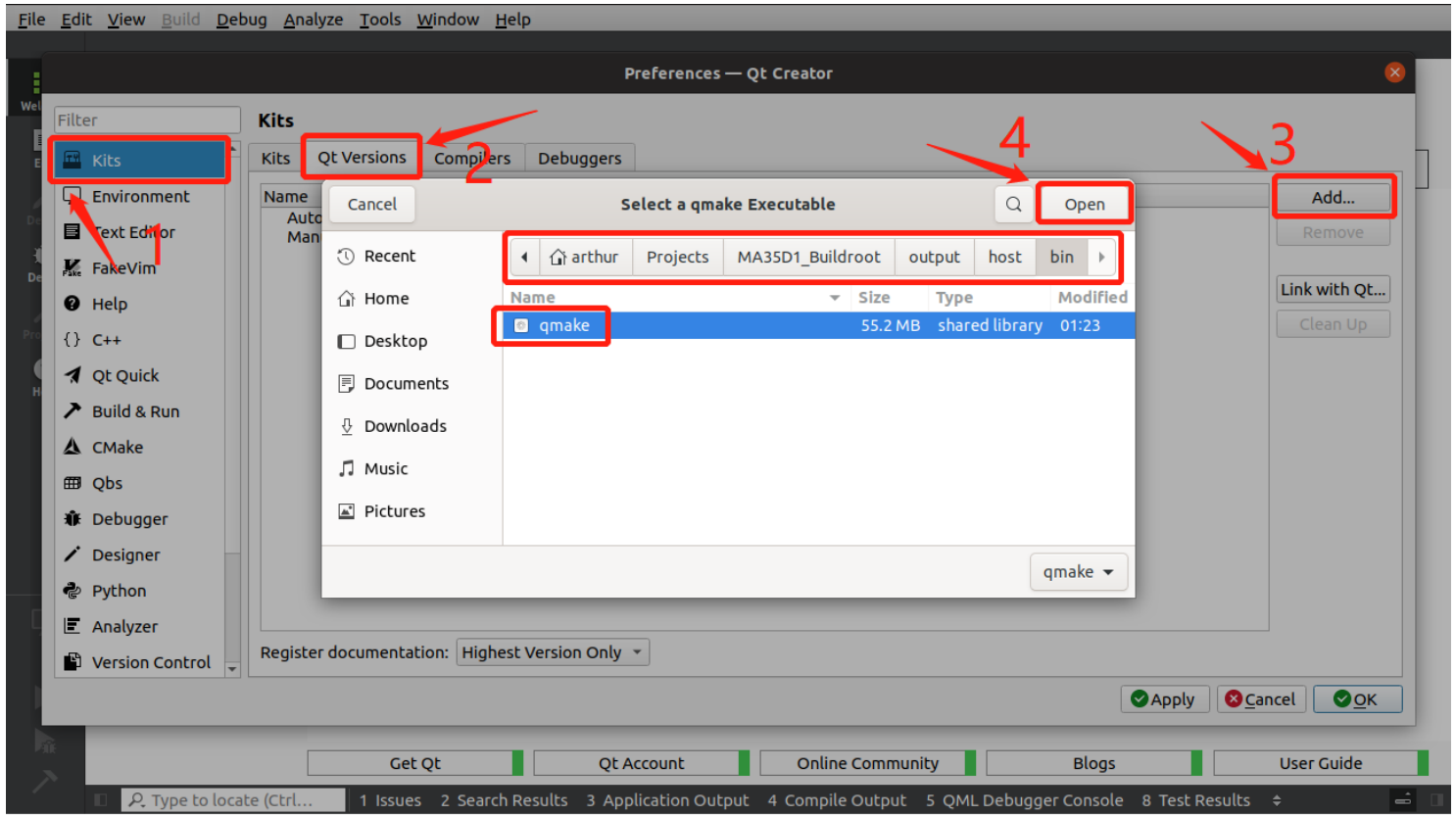




NuDeveloper Ecosystem – Make the engineers' job easier.

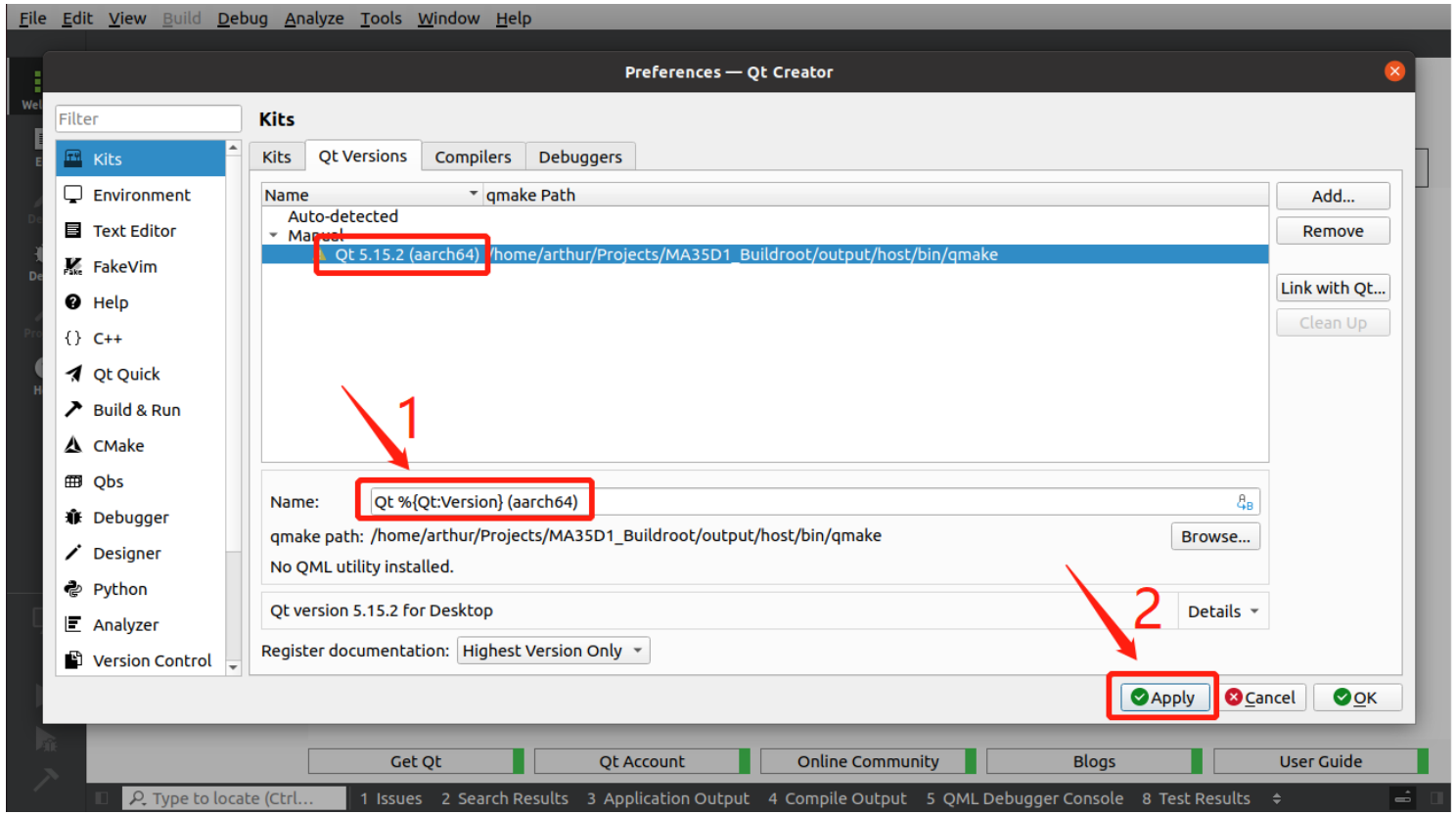
# Configure Kits (Qt Versions)

- Add **qmake**



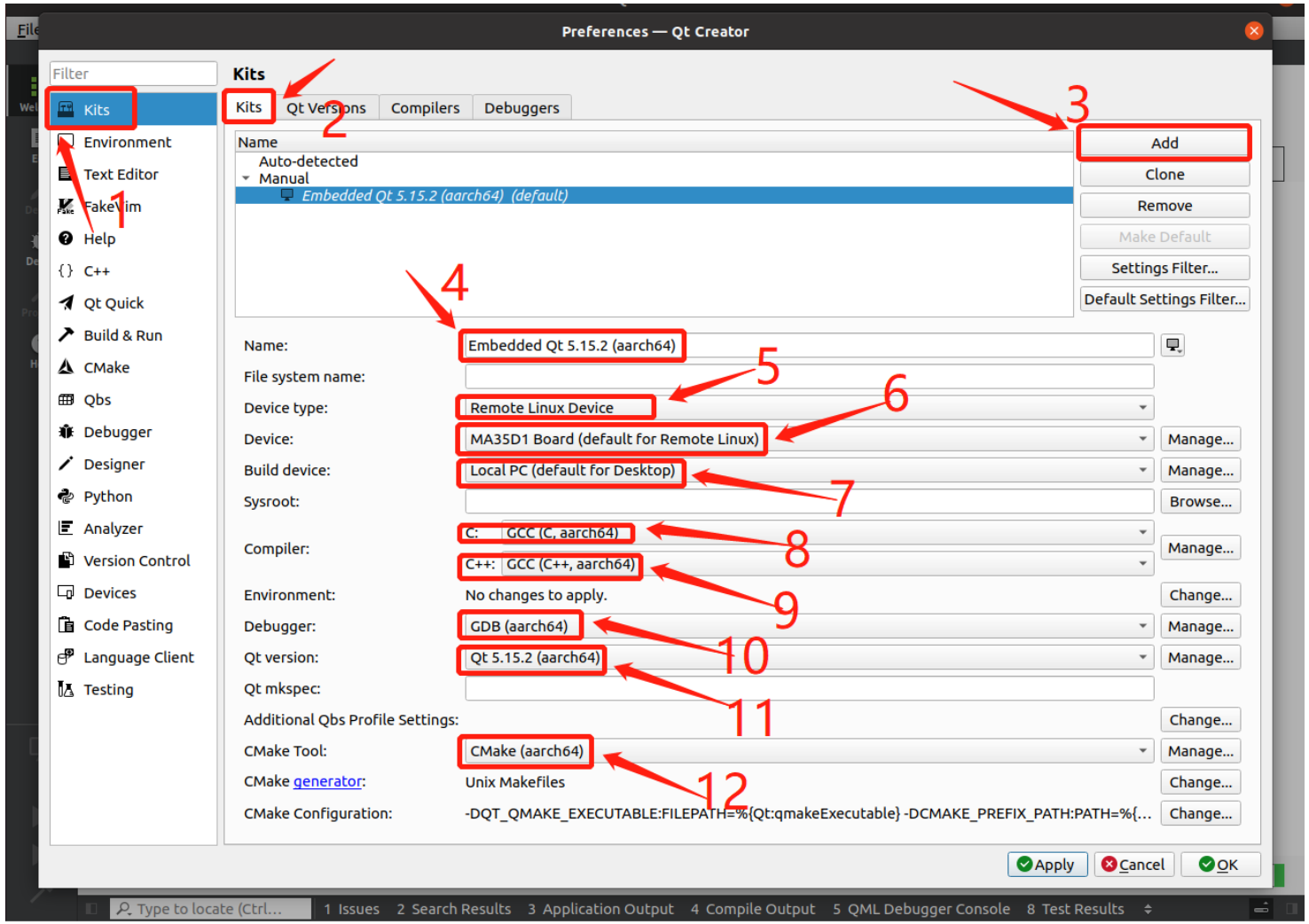
# Configure Kits (Qt Versions)

- Set **qmake** to ***${BR2_DIR}/output/host/bin/qmake***.



# Configure Kits

- Set Kits to
  - **Name**: Embedded Qt 5.15.2 (aarch64)
  - **Device type**: Remote Linux Device
  - **Device**: MA35D1 Board
  - **Build device**: Local PC
  - **Compiler C**: GCC (C, aarch64)
  - **Compiler C++**: GCC (C++, aarch64)
  - **Debugger**: GDB (aarch64)
  - **Qt version**: Qt 5.15.2 (aarch64)
  - **CMake Tool**: CMake (aarch64)





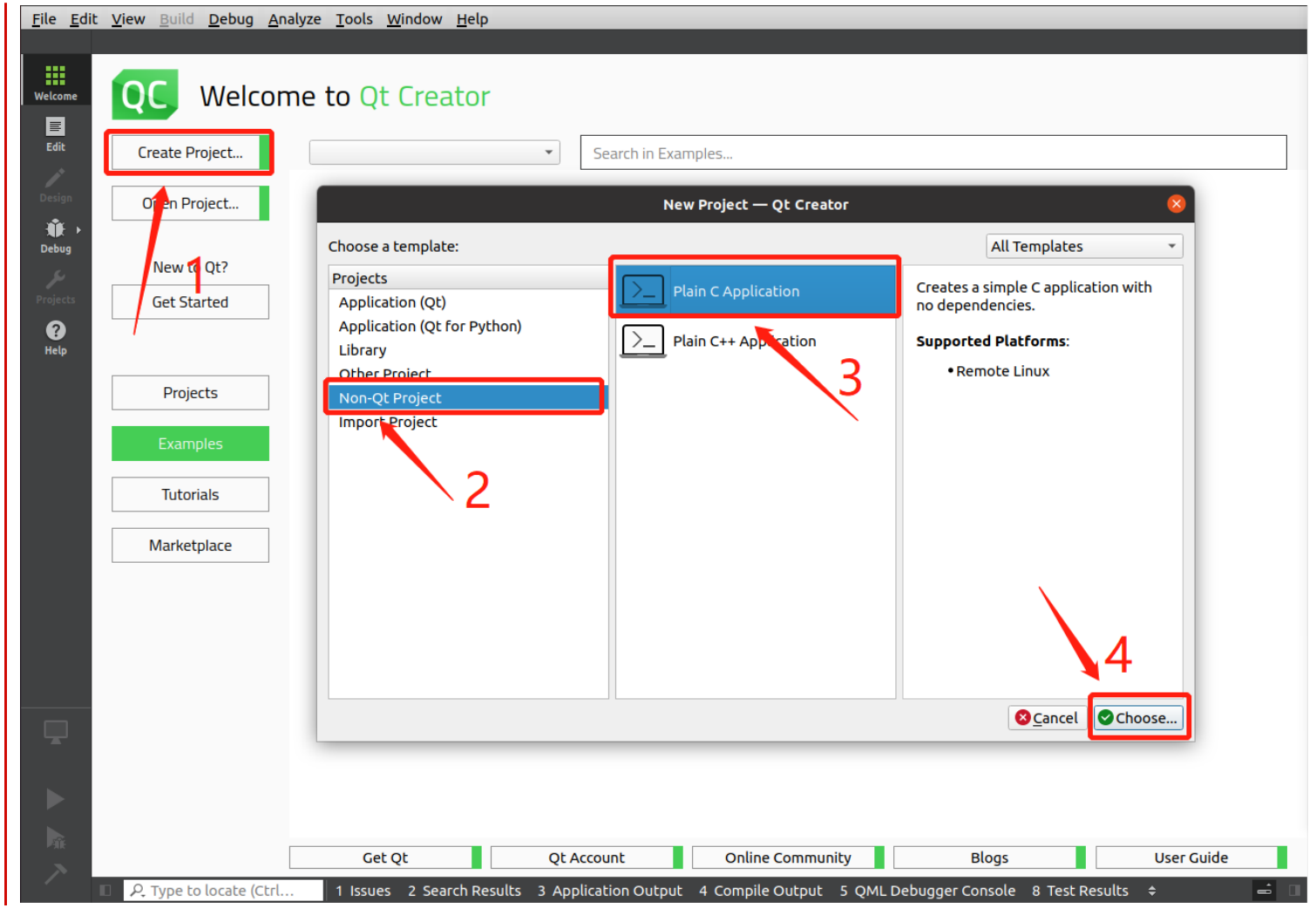


NuDeveloper Ecosystem – Make the engineers' job easier.

nuvoTon

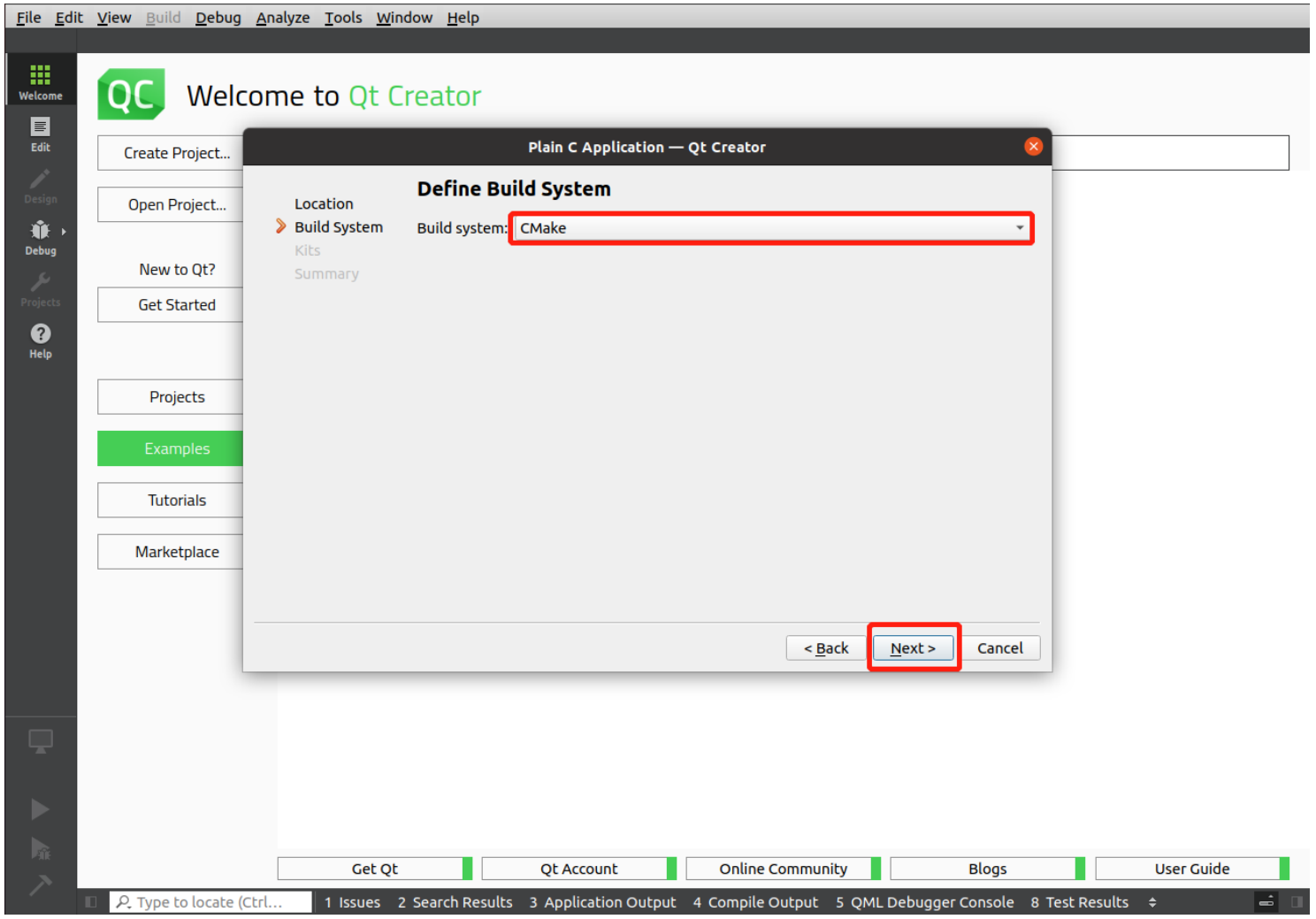
# Create a Non-Qt Project Plain C Application

- Choose **Non-Qt Project**, **Plain C Application**



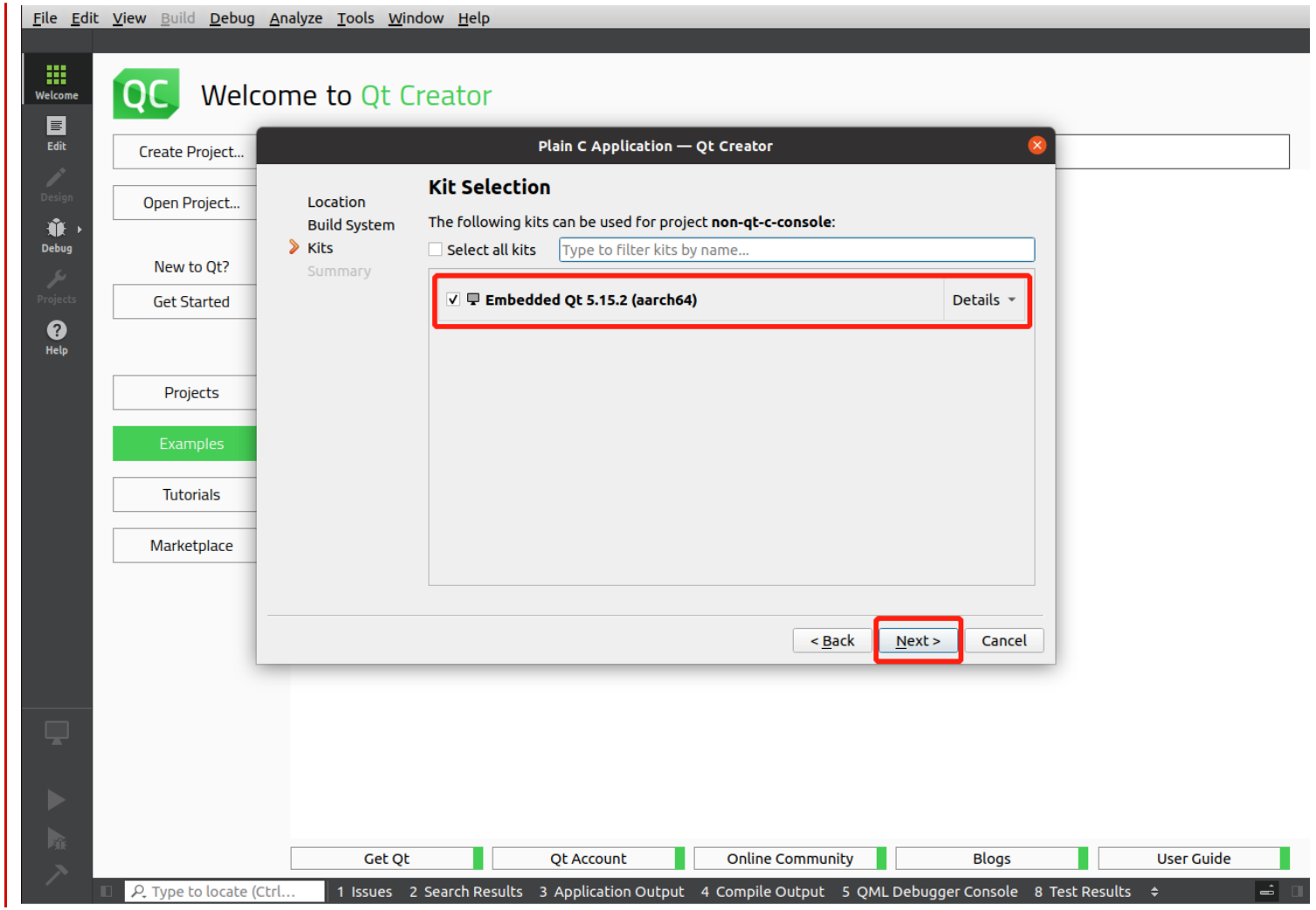
# Create a Non-Qt Project Plain C Application

- Select **Build system** as ***CMake***



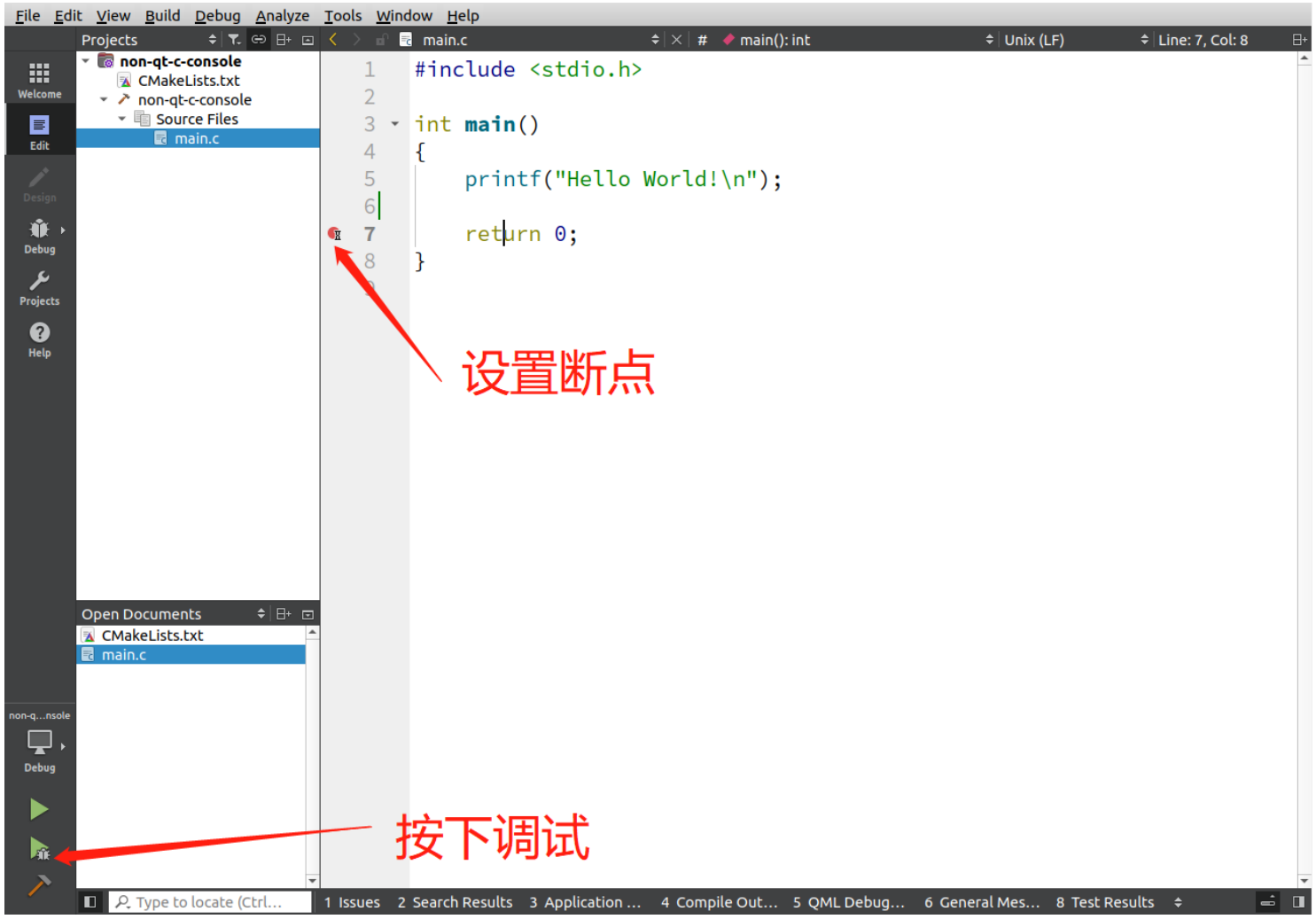
# Create a Non-Qt Project Plain C Application

- Select **Kit** as ***Embedded Qt 5.15.2 (aarch64)***



# Create a Non-Qt Project Plain C Application

- **Double click** the corresponding line to set ***breakpoint***.



Joy of innovation

NUVOTON

谢谢

謝謝

Děkuji

Bedankt

Thank you

Kiitos

Merci

Danke

Grazie

ありがとう

감사합니다

Dziękujemy

Obrigado

Спасибо

Gracias

Teşekkür ederim

Cảm ơn