Visual Studio Code User Setup Readme V1.4

About this Document

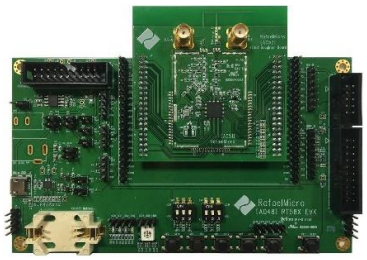
This document only supports "***VSCodeUserSetup-x64-1.62.3***". For any other version than v1.62.3, there is no guaranteed it can work properly.

Table of Contents

1. Introduction

The purpose of this document is to teach users about the installation and configuration of Visual Studio Code, Embedded IDE, and GNU ARM Embedded Toolchain. Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging, so users can develop applications on the Visual Studio Code development kit.

2. Visual Studio Code Development Environment



Rafael RT58x EVK

ARM J-Link Adapter

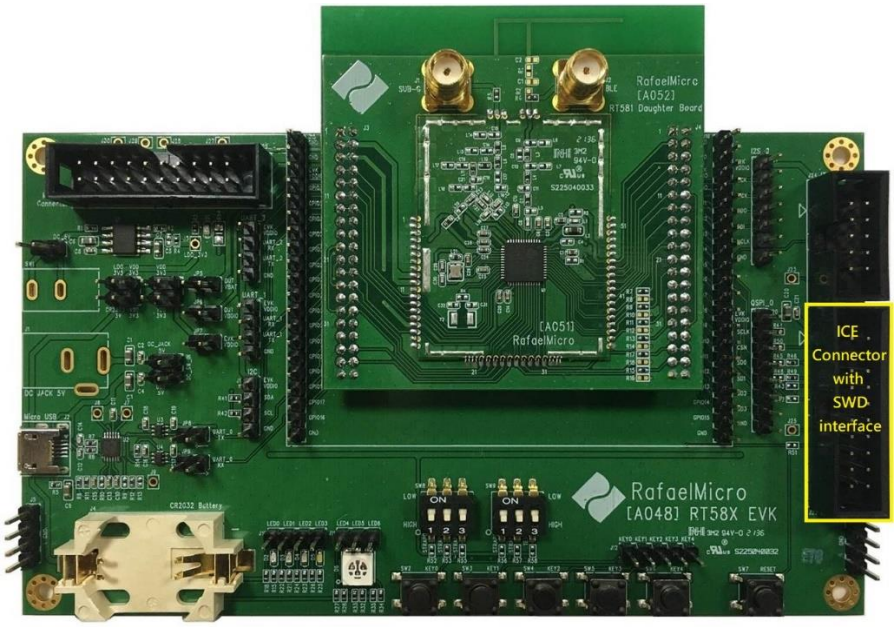
Visual Studio Code/EIDE

2.1 Debugger

- Install the ARM J-Link driver.
- Connect J-Link Adapter to RT58x EVK board via JTAG/SWD bus.
- Connect J-Link Adapter to PC via USB.

2.2 Development Board

The RT58x EVK provides an SWD interface with connector for use with an ICE debugger (J-Link Adapter) via 20pin IDE cable.



For detailed description of RT58x EVK board, please refer to RT58X EVK User Guide.

2.3 Visual Studio Code

Visual Studio Code features a lightning fast source code editor, perfect for day-to-day use. Visual Studio Code includes an interactive debugger, so you can step through source code, inspect variables, view call stacks, and execute command in the console. Visual Studio Code also integrates with build and scripting tools to perform common tasks making everyday workflows faster.

VS Code is available and free for download. It does not require a serial number or license key. For detailed introduction and download, please click the following link:
https://code.visualstudio.com

Users can also find the VS Code setup kit in the Rafael Micro download center.
https://support.rafaelmicro.com:8088/projects/download-center/files

The VS Code setup kit contains the following:

VS Code development kit:
[VS Code setup kit]\VSCode\VSCodeUserSetup-x64-1.62.3.exe

VS Code extension kit:
[VS Code setup kit]\VSCode\extensions.rar

Using EIDE(v2.15.0 extension) to development gcc project

GNU ARM Toolchain:
[VS Code setup kit]\ToolChain\GNU Arm Embedded Toolchain.rar

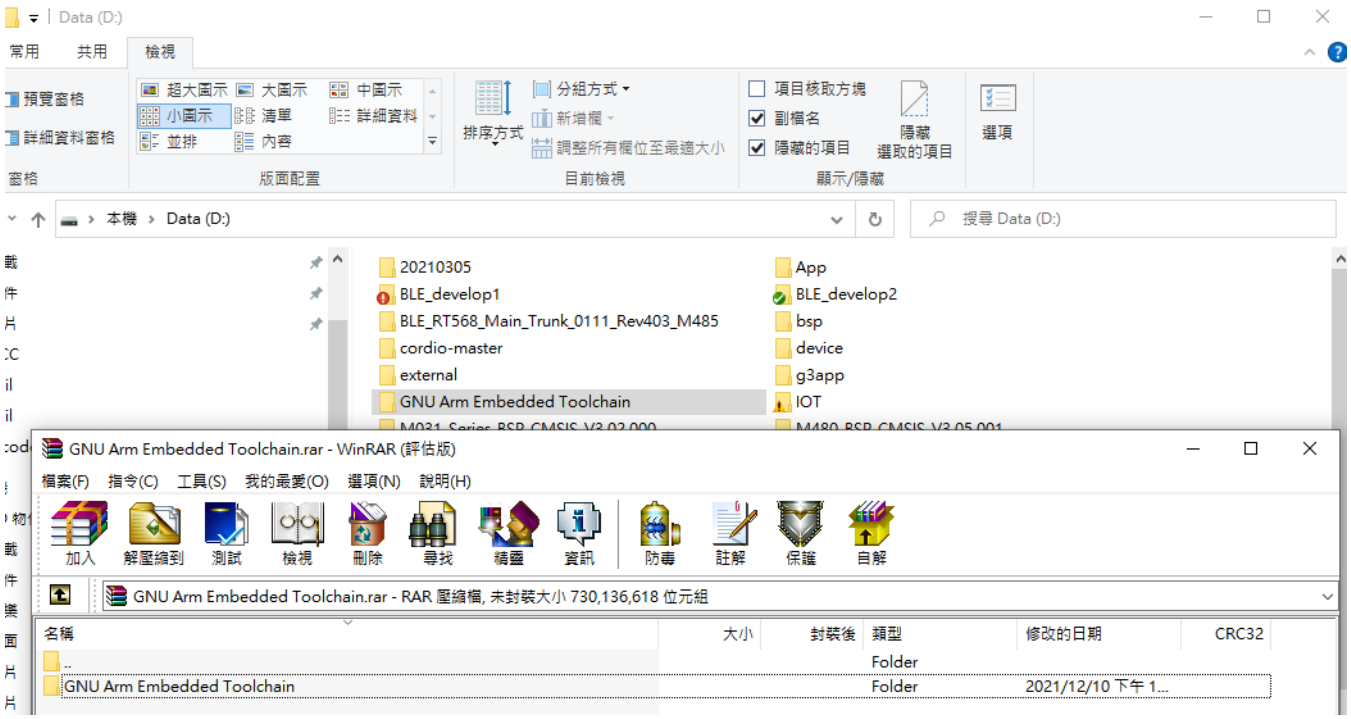
JLink extension kit:
[VS Code setup kit]\JLink.rar

3. Visual Studio Code Setup

Getting up and running with Visual Studio Code is quick and easy. The Visual Studio Code installer is a small download so users can run the installer (VSCodeUserSetup-{version}.exe) to easily install in a few minutes by following the installation prompts.

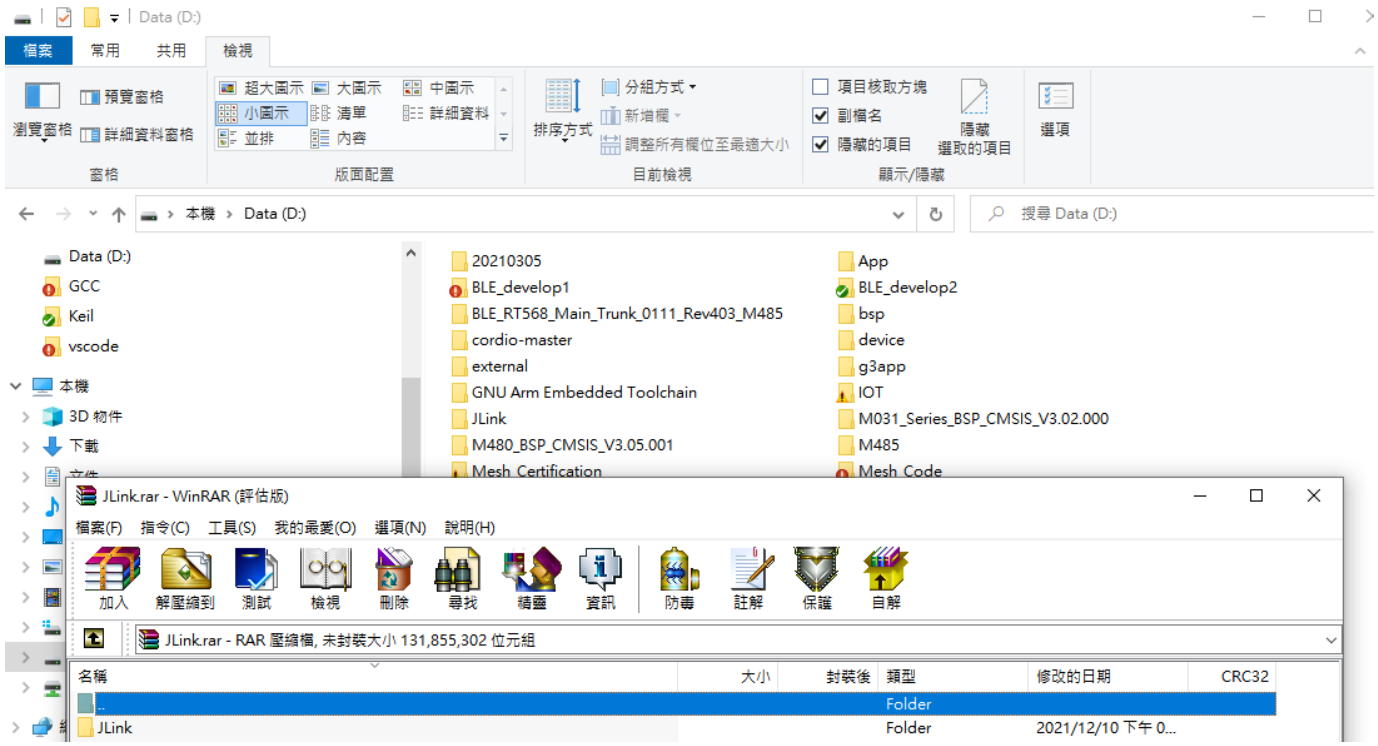
3.1 GNU ARM Toolchain Configuration

Open the compressed file GNU Arm Embedded Toolchain.rar and extract the GNU Arm Embedded Toolchain folder to D disk.



3.2 JLink Extension Configuration

Open the compressed file JLink.rar and extract the JLink folder to D disk.



3.3 JLink Edit JLinkDevices.xml

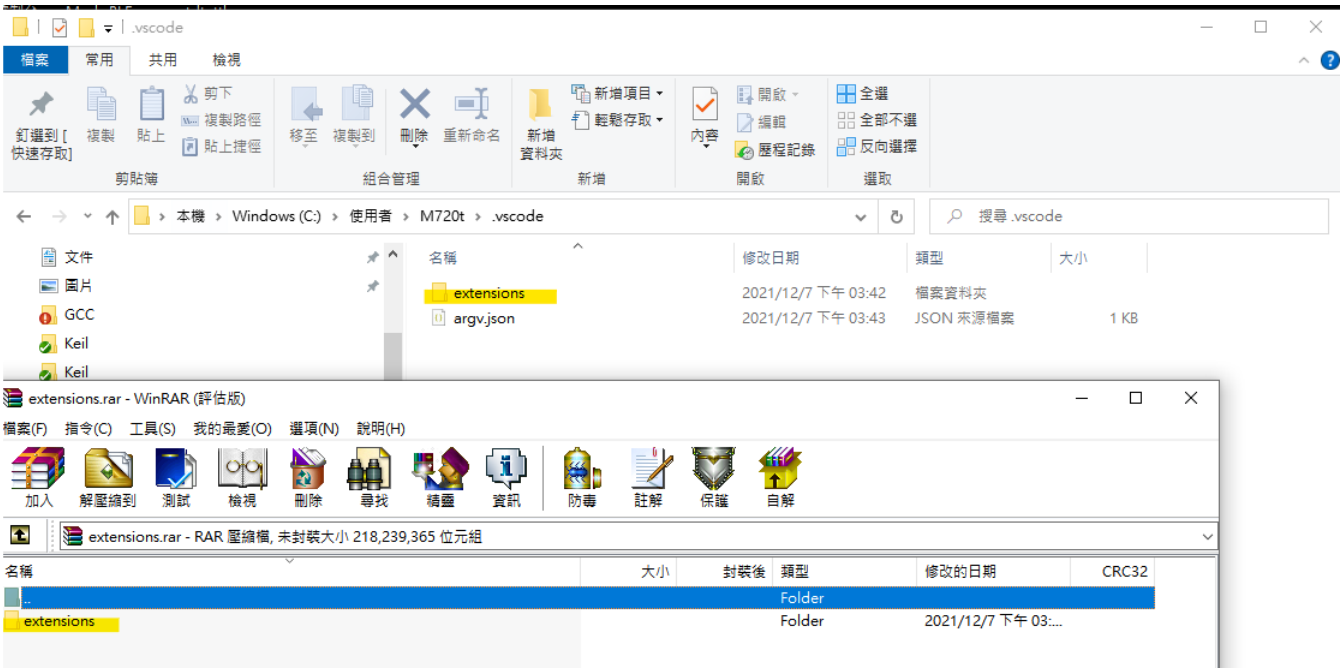
Edit Jlinkdevices.xml that the JLink for support different flash size MCU.

1. Copy Flash algorithm (RT58X_1MB.FLM/RT58x_512KB.FLM/RT58x_2MB.FLM) into “D:\JLink\”
2. Edit D:\JLink\JlinkDevices.xml file add new device

```
<DataBase>
  <!--                    -->
  <!-- Rafael -->
  <!--                    -->
  <Device>
   <ChipInfo Vendor="Rafael"
             Name="ARMCM3"
             Core="JLINK_CORE_CORTEX_M3"
             WorkRAMAddr="0x20000000"
             WorkRAMSize="0x1000" />
   <FlashBankInfo Name="Internal Flash"
                  BaseAddr="0x00007000"
                  MaxSize="0x000F7000"
                  Loader="RT58x_1MB.FLM"
                  LoaderType="FLASH_ALGO_TYPE_CMSIS"
                  AlwaysPresent="1" />
  </Device>
  <Device>
   <ChipInfo Vendor="Rafael"
             Name="RT58x_512KB"
             Core="JLINK_CORE_CORTEX_M3"
             WorkRAMAddr="0x20000000"
             WorkRAMSize="0x1000" />
   <FlashBankInfo Name="Internal Flash"
                  BaseAddr="0x00007000"
                  MaxSize="0x00077000"
                  Loader="RT58x_512KB.FLM"
                  LoaderType="FLASH_ALGO_TYPE_CMSIS"
                  AlwaysPresent="1" />
  </Device>
  <Device>
   <ChipInfo Vendor="Rafael"
             Name="RT58x_2MB"
             Core="JLINK_CORE_CORTEX_M3"
             WorkRAMAddr="0x20000000"
             WorkRAMSize="0x1000" />
   <FlashBankInfo Name="Internal Flash"
                  BaseAddr="0x00007000"
                  MaxSize="0x001F7000"
                  Loader="RT58x_2MB.FLM"
                  LoaderType="FLASH_ALGO_TYPE_CMSIS"
                  AlwaysPresent="1" />
  </Device>
</DataBase>
```

3.4 VS Code Extension Configuration

Open the compressed file extensions.rar, extract the extensions folder and overwrite it to the location of the VS Code extension kit. By default, VS Code extension kit is installed under C:\Users\your_PC_name\.vscode.



3.5 VS Code Configuration

Execute VSCode, press the key “Ctrl + Shift + P”, type the text “settings”, and select the option “Preferences: Open Settings(JSON)”.

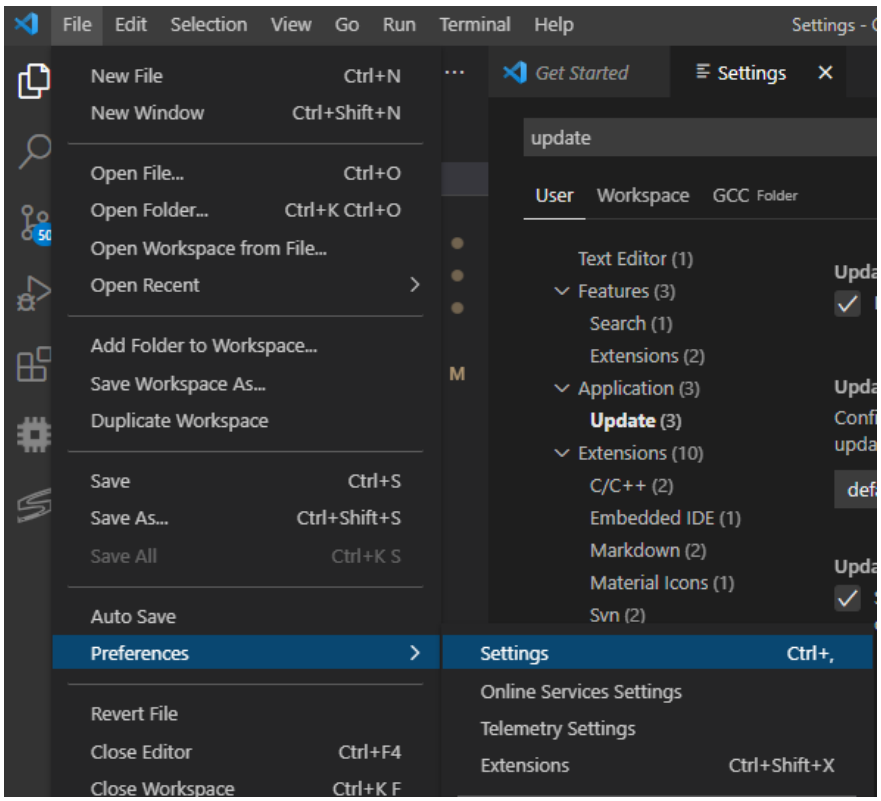
Open settings.json, add the following settings at the end, save and close the settings file.

"EIDE.JLink.InstallDirectory": "D:\\JLink",
"EIDE.JLink.DeviceXmlPath": "D:\\JLink",
"EIDE.ARM.GCC.InstallDirectory": "D:\\GNU Arm Embedded Toolchain\\10 2021.10",
"cortex-debug.armToolchainPath": "D:\\GNU Arm Embedded Toolchain\\10 2021.10\\bin",
"cortex-debug.JLinkGDBServerPath": "D:\\JLink\\JLinkGDBServerCL.exe",

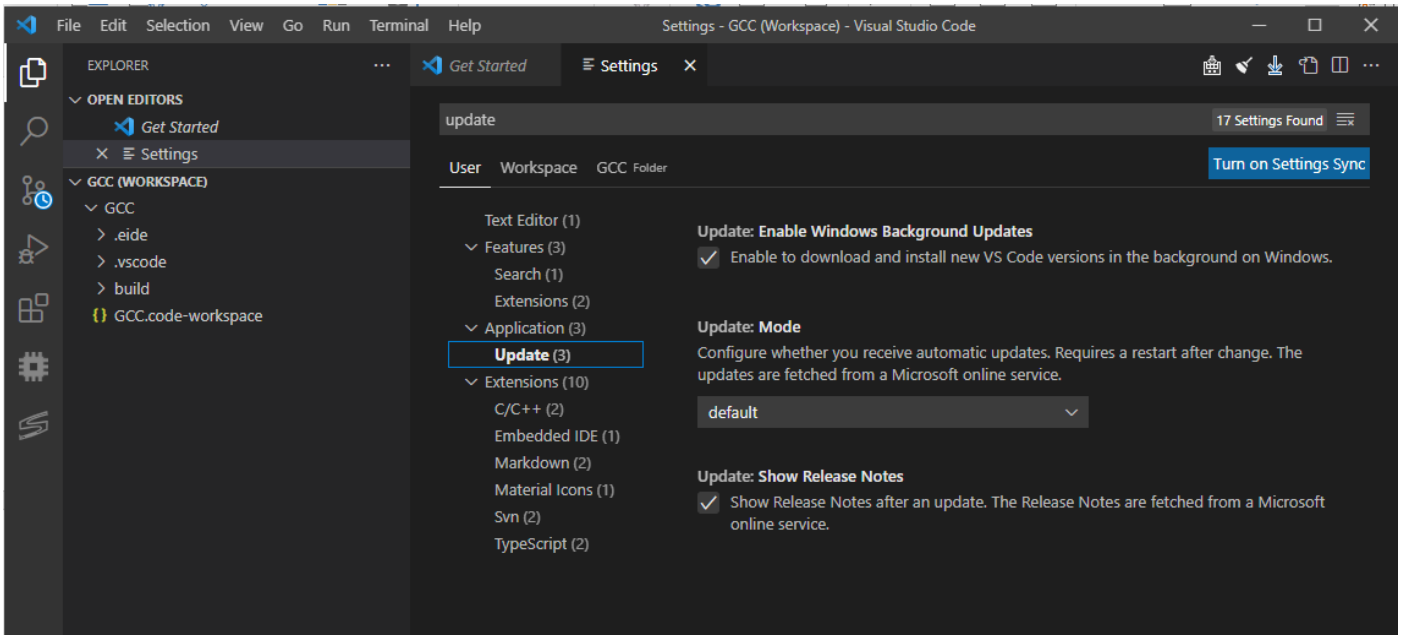
```
25
26      "EIDE.JLink.InstallDirectory": "D:\\JLink",
27      "EIDE.JLink.DeviceXmlPath": "D:\\JLink",
28      "EIDE.ARM.GCC.InstallDirectory": "D:\\GNU Arm Embedded Toolchain\\10 2021.10",
29      "cortex-debug.armToolchainPath": "D:\\GNU Arm Embedded Toolchain\\10 2021.10\\bin",
30      "cortex-debug.JLinkGDBServerPath": "D:\\JLink\\JLinkGDBServerCL.exe",
31  }
```

3.6 VS Code Disable Update

File→Preference→Setting



Select Update Option



Update Mode

Default : Auto Update

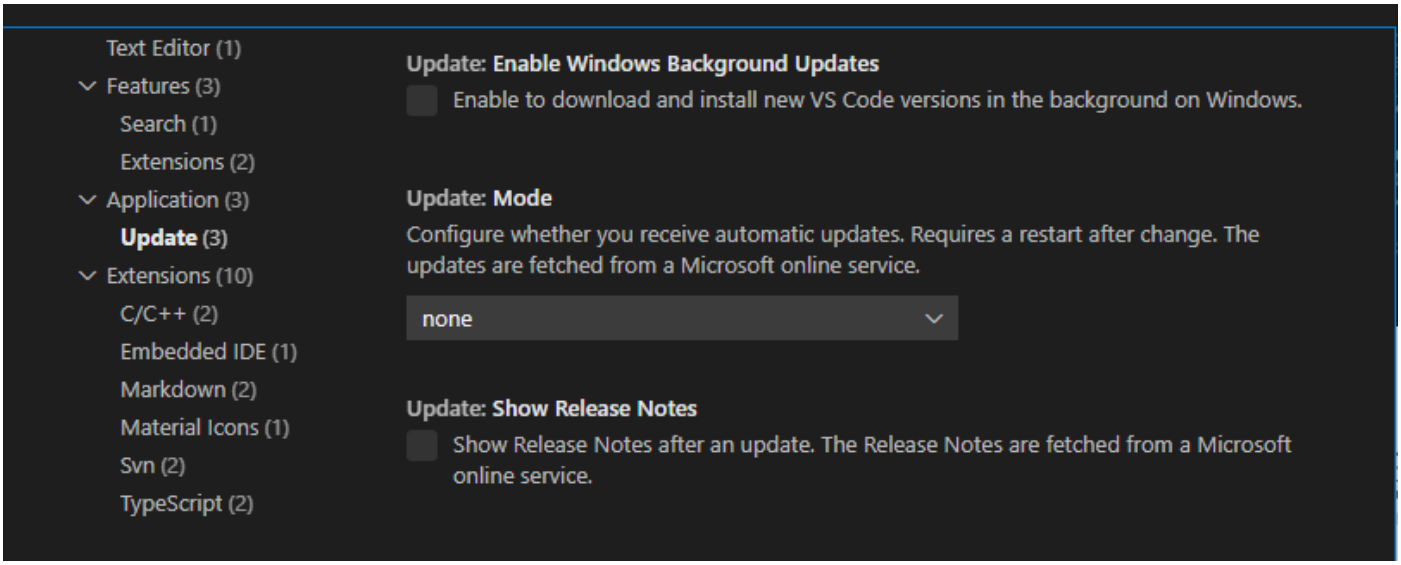
None : Disable Update

Manual : User Update

Start : Check Update

Update : Enable Windows Background Updates

Update : Show Release Notes



3.7 Project Builder Configurations

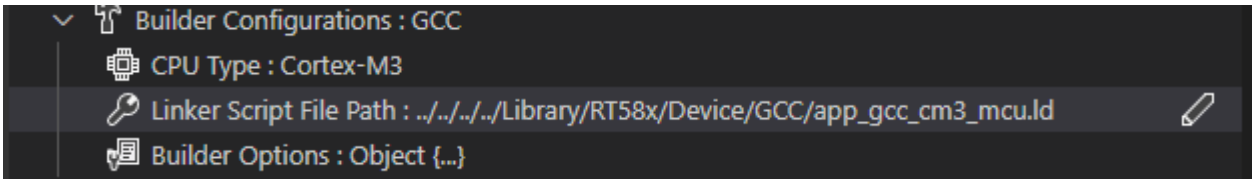
Default Linker Script File Path: Library\RT58x\Device\GCC\app_gcc_cm3_mcu.ld.

According to the RafaelMicro MCU flash size to select the link script file.

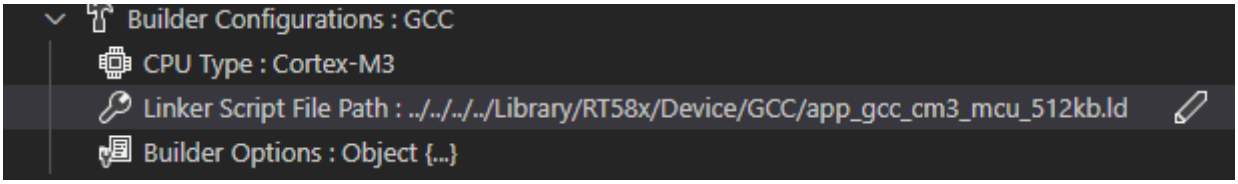
If you are using 512K Flash MCU, modify the path of the link script file to "Library RT58x Device GCC app_gcc_cm3_mcu_512KB.ld".

Using the 2MB Flash MCU, modify the path of the link script file to "Library RT58x Device GCC app_gcc_cm3_mcu_2mb.ld".

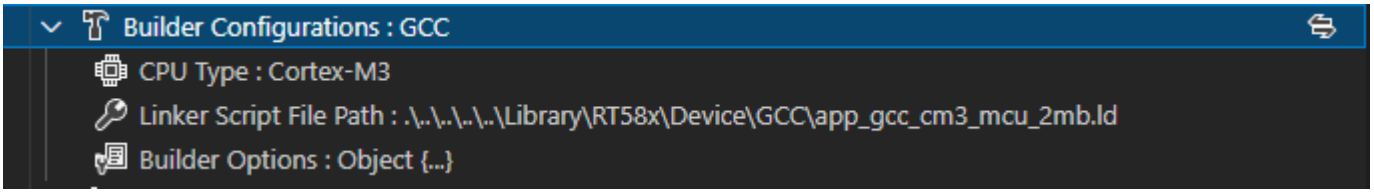
Default Linker Script File Path:



512K Flash MCU Linker Script File Path:



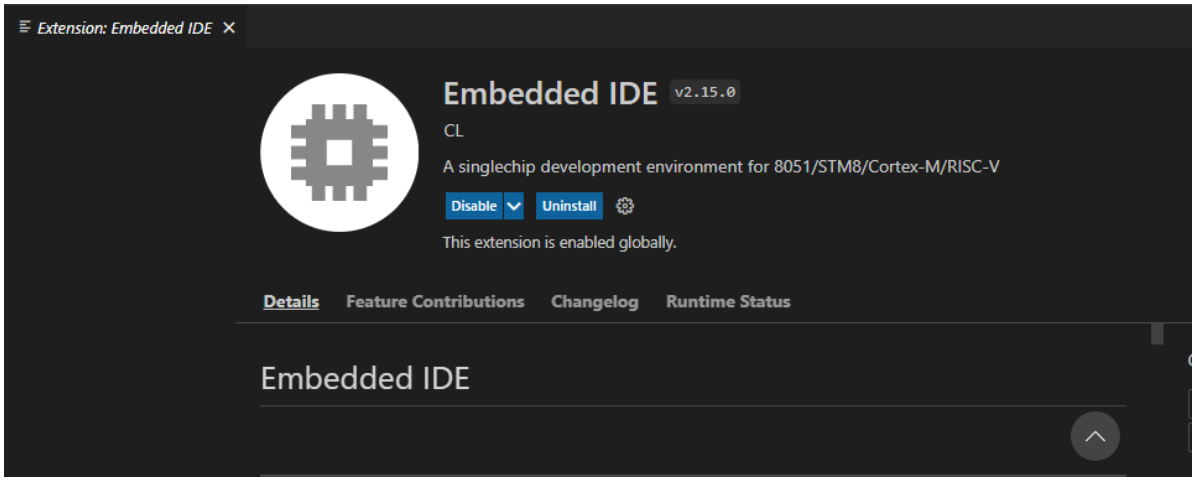
2MB Flash



3.8 Development GCC Project Extension Tool

Using EIDE(v2.15.0)extension tool to development GCC project.

reference website : https://em-ide.com/docs/intro/



Revision History

Revision	Description	Owner	Date
1.0	Initial version.	Chiaho Hu	2022/02/15
1.1	Revise vs code kit download description	Ives Lee	2022/07/15
1.2	Add edit JLinkdevices.xml for download code Add change linker script file for different flash size MCU	Ives Lee	2022/07/18
1.3	Revise JLinkdevices.xml and linker script description Add disable Vs code update	Ives Lee	2022/05/03
1.4	Add development GCC project extension tool version and reference website.	Ives Lee	2024/08/19