

N9H30 Non-OS BSP 使用手冊

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of N9H30 microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.



內容

- 1 N9H30 Non-OS BSP 簡介 3
 - 1.1 Keil 開發環境..... 3
 - 1.2 Eclipse 開發環境..... 4
 - 1.3 Eclipse 開發環境 (2025-09 版本以上)17
 - 1.4 開發板設置28
- 2 BSP 內容..... 29
 - 2.1 BSP 目錄架構29
 - 2.2 Non-OS BSP 內容29
- 3 NuWriter 30
- 4 版本歷史..... 31

1 N9H30 Non-OS BSP 簡介

這包 BSP 支持了 N9H30 系列芯片. 新唐科技的 N9H30 系列芯片是以 ARM926EJS 為核心的系統級單芯片. 包含了 16kB I-Cache 以及 16kB D-Cache 以及 MMU 記憶體管理模塊. 最高支援到 300MHz 的頻率, 並且提供了豐富的外設接口周邊. 有 USB 快速 Host/Device, SDHC, 支援 TFT LCD 介面, 和 I2S audio 介面, 有 11 組 UART... 等. 並可以由 NAND flash, SPI Flash 開機.

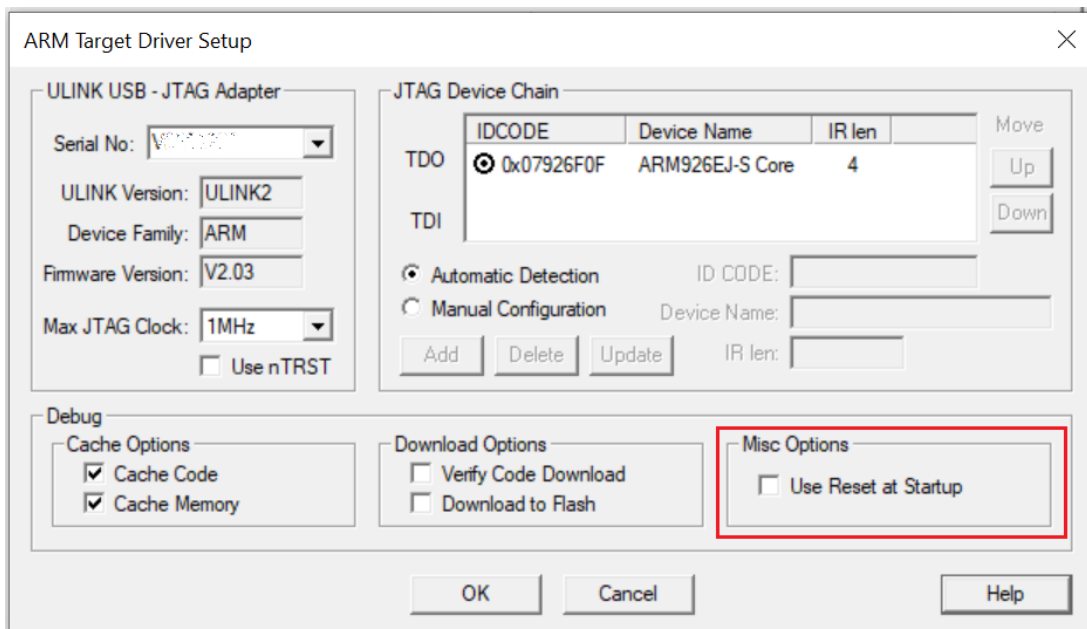
這包 Non-OS BSP 包含了以下內容:

- N9H30 Non-OS 驅動程式
- U-Boot 映像檔, 以及 N9H30 使用的驅動程式
- Windows 端燒錄程序 Nu-Writer, 以及所需的驅動
- 說明文檔

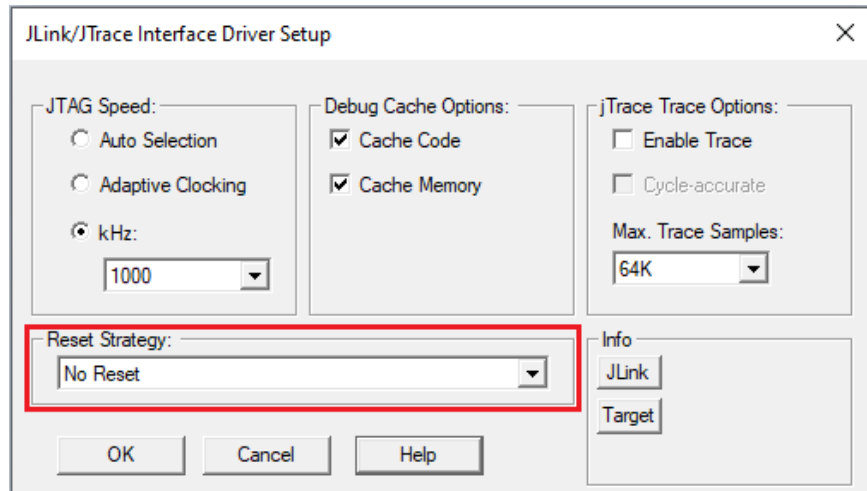
1.1 Keil 開發環境

Non-OS BSP 可以使用 Keil IDE 環境開發. 偵錯的 ICE 則是支援了 ULINK2. 開發環境的使用並不屬於本文件的介紹範圍. 若是需要使用說明, 可以至 Keil 的官方網站 <http://www.keil.com/> 查詢.

N9H30 支援 J-TAG 介面. 使用者可使用 ICE 透過 J-TAG 介面下載程式到 RAM 中進行偵錯. 但是 N9H30 在上電後, J-TAG 介面是關閉的, 除非讓 N9H30 新進入 USB 開機模式, 並使用 NuWriter 建立連線以後, 或者是 N9H30 不處於安全模式, 並且成功執行儲存於 SPI/NAND/eMMC 的程式. N9H30 的 J-TAG 介面才可使用. 請注意, ICE Reset 必須關閉, 否則 J-TAG 又會馬上被關閉. 以下是 ULINK2 關閉 Reset 的設置.



J-Link 也有類似的設置, 以下是關閉 J-Link Reset 的方式.



N9H30 Non-OS BSP 的 loader 跟 Linux BSP 使用了一套相同的開源 loader, U-Boot. U-Boot 的開發環境是在 Linux 系統中. 若是有開發需求, 可下載 N9H30 Linux BSP, 並參考 N9H30 Linux BSP 的使用手冊來架設開發環境. 或是可以直接使用 BSP 裡面已經預先編譯好的執行檔. 若是系統是從 SPI/eMMC 開機, 可以不使用 loader, 直接執行主程式. 但在 NAND boot 時, 需考慮壞塊, 的處理, 建議使用 U-Boot 開機.

1.2 Eclipse 開發環境

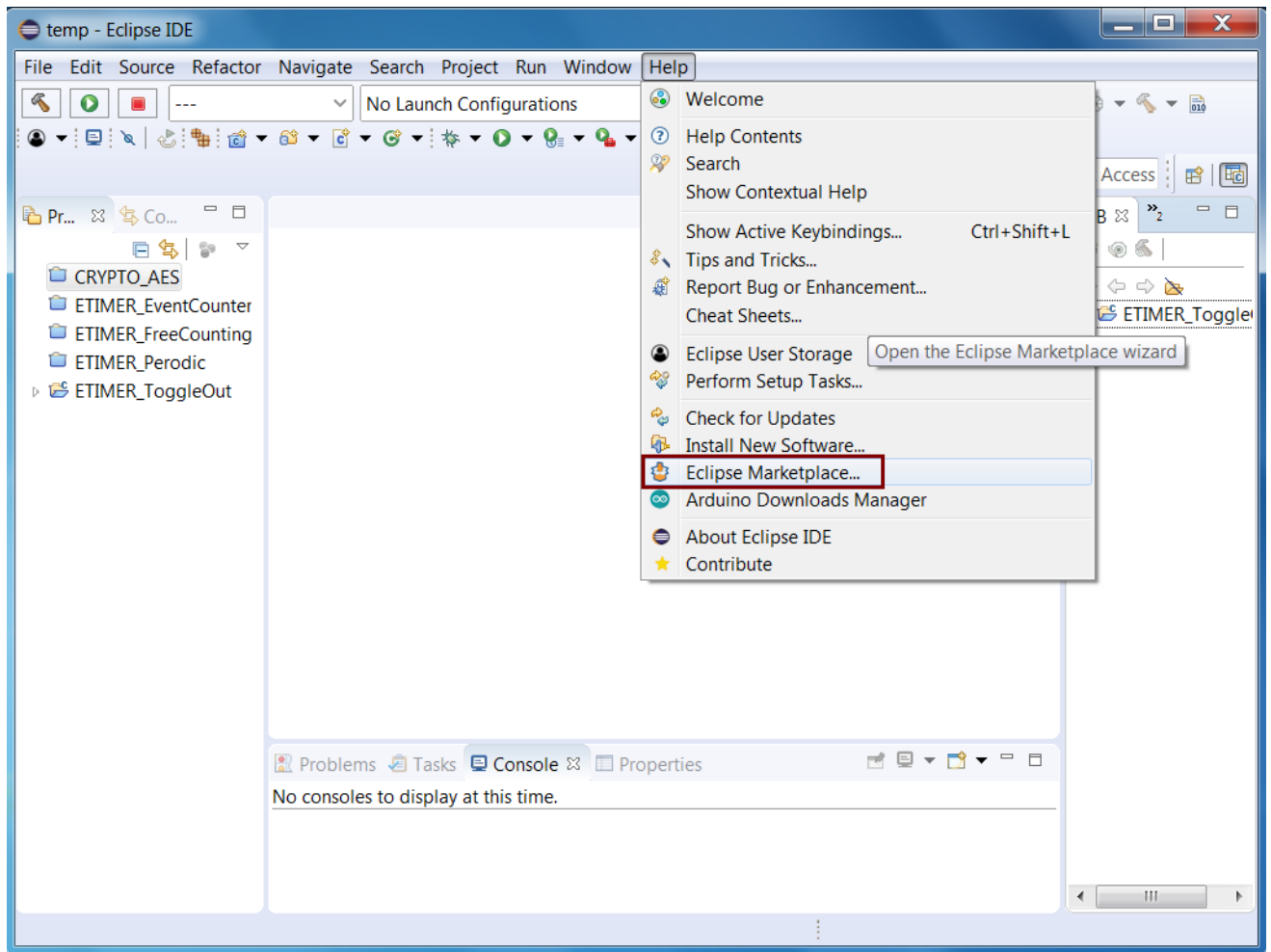
N9H30 BSP 也支援 Eclipse 開發環境. 本節將介紹Eclipse的安裝步驟. 首先, 使用者至Eclipse 官方網站<https://www.eclipse.org/downloads/> 下載Eclipse IDE for C/C++ Developers Tool 開發工具, 選擇適用作業系統及位元的版本, 然而Eclipse 是跑在 Java runtime environment(JRE) 下, 所以必須至 Java 網站下載 JRE或JDK 並安裝.

從 <https://github.com/xpack-dev-tools/windows-build-tools-xpack/releases> 下載編程工具。選擇 “GNU MCU Eclipse Windows Build Tools v2.12 20190422”, 然後下載 “gnu-mcu-eclipse-windows-build-tools-2.12-20190422-1053-win64.zip”, 並壓縮到本地 C:\eclipse 目錄下。

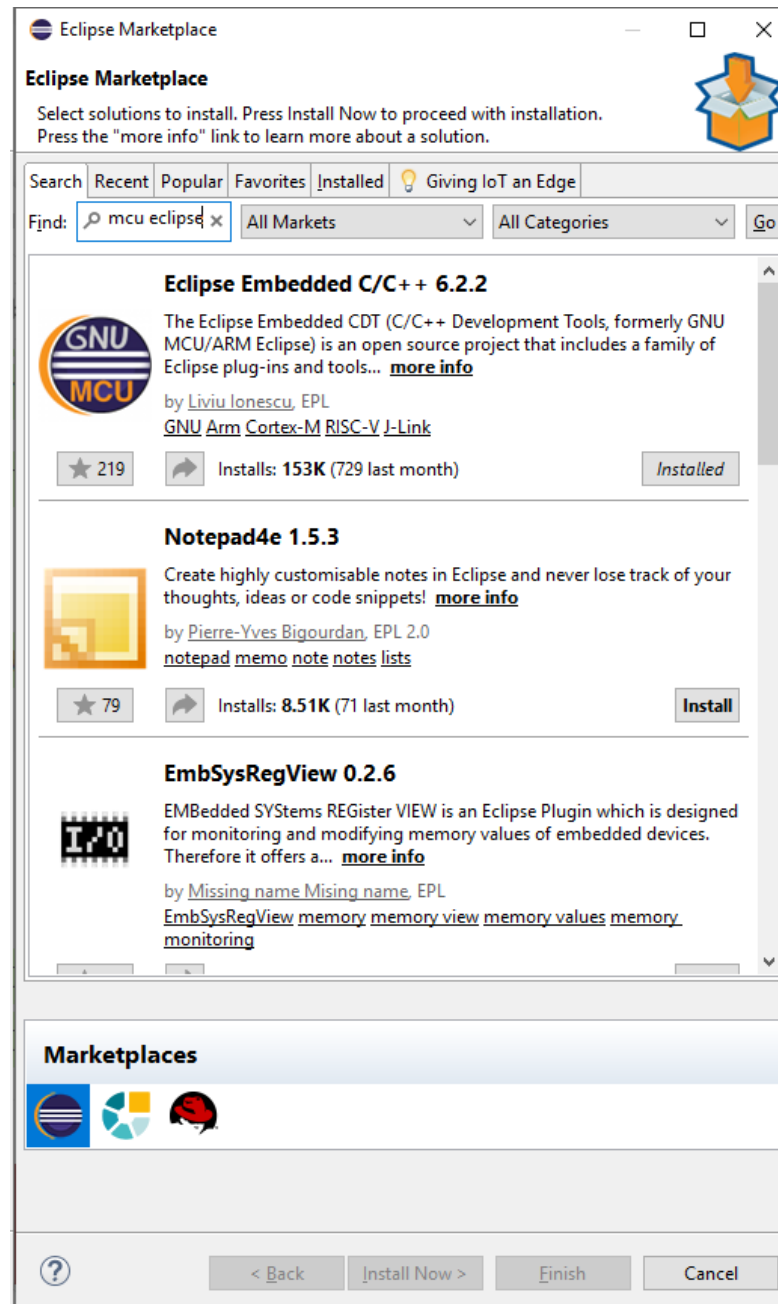
從 <https://developer.arm.com/downloads/-/gnu-rm> 下載GCC 工具鏈。選擇 “gcc-arm-none-eabi-10.3-2021.10-win32.zip” 下載, 並壓縮到本地 C:\eclipse 目錄下。

交叉編譯工具 - GNU ARM Embedded Toolchain 可以從網 <https://gnu-mcu-eclipse.github.io/plugins/install/> 下載。

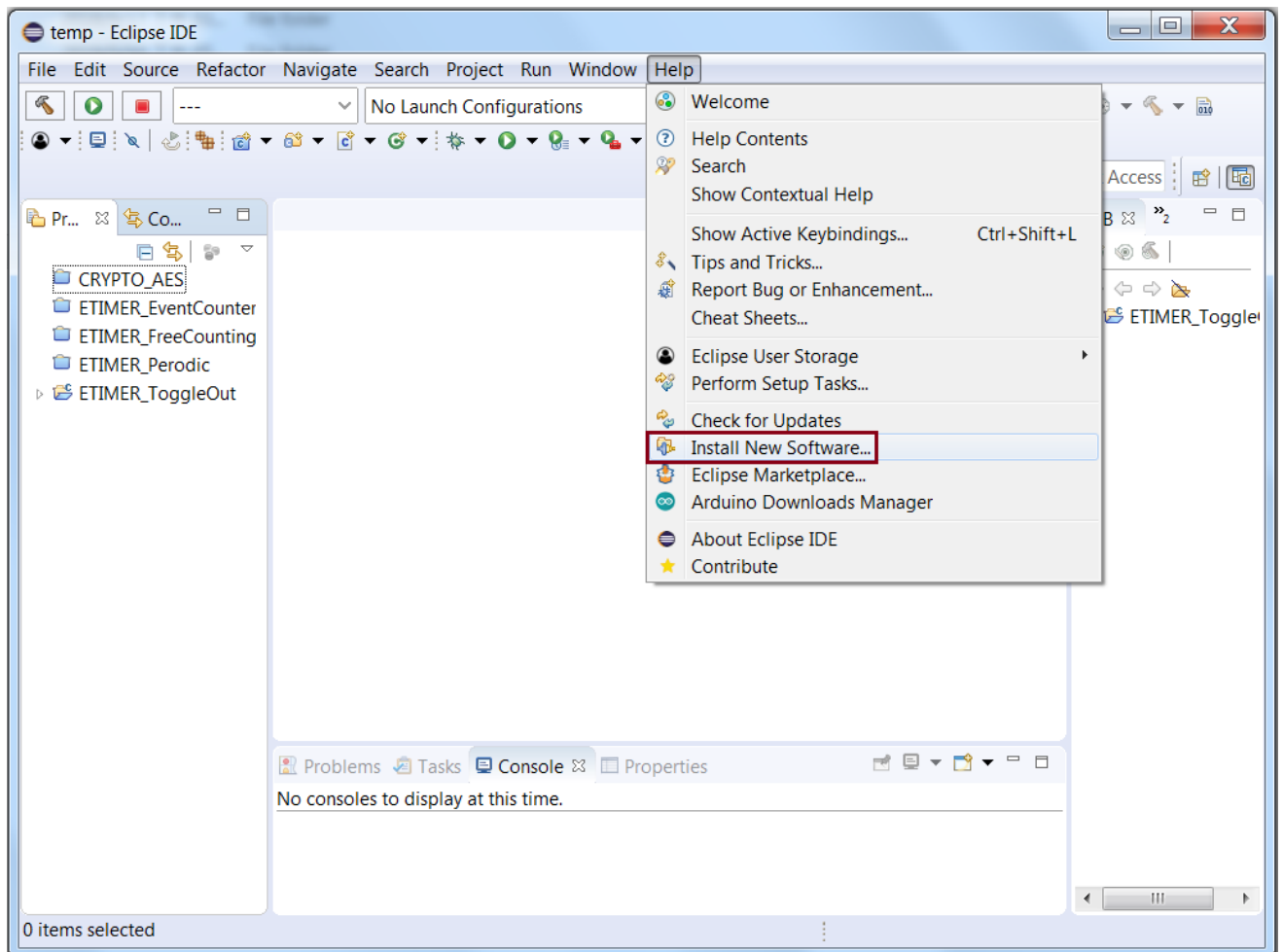
最後, 安裝完上述軟體後, 執行Eclipse 進入開發介面後, 點選 Help -> Eclipse Marketplace.



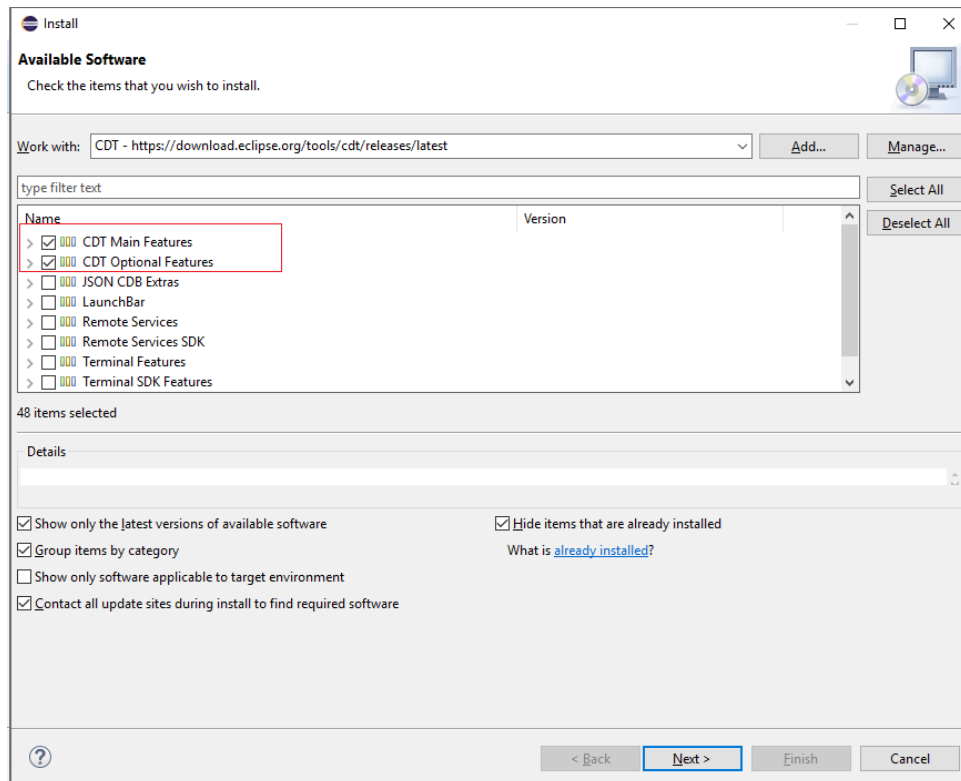
在Find欄位輸入gnu mcu eclipse, 搜尋結果顯示於下方視窗, 使用者選擇目前最新版本並按下 Install按鈕安裝, 補齊相關plug in套件(GNU MCU C/C++).



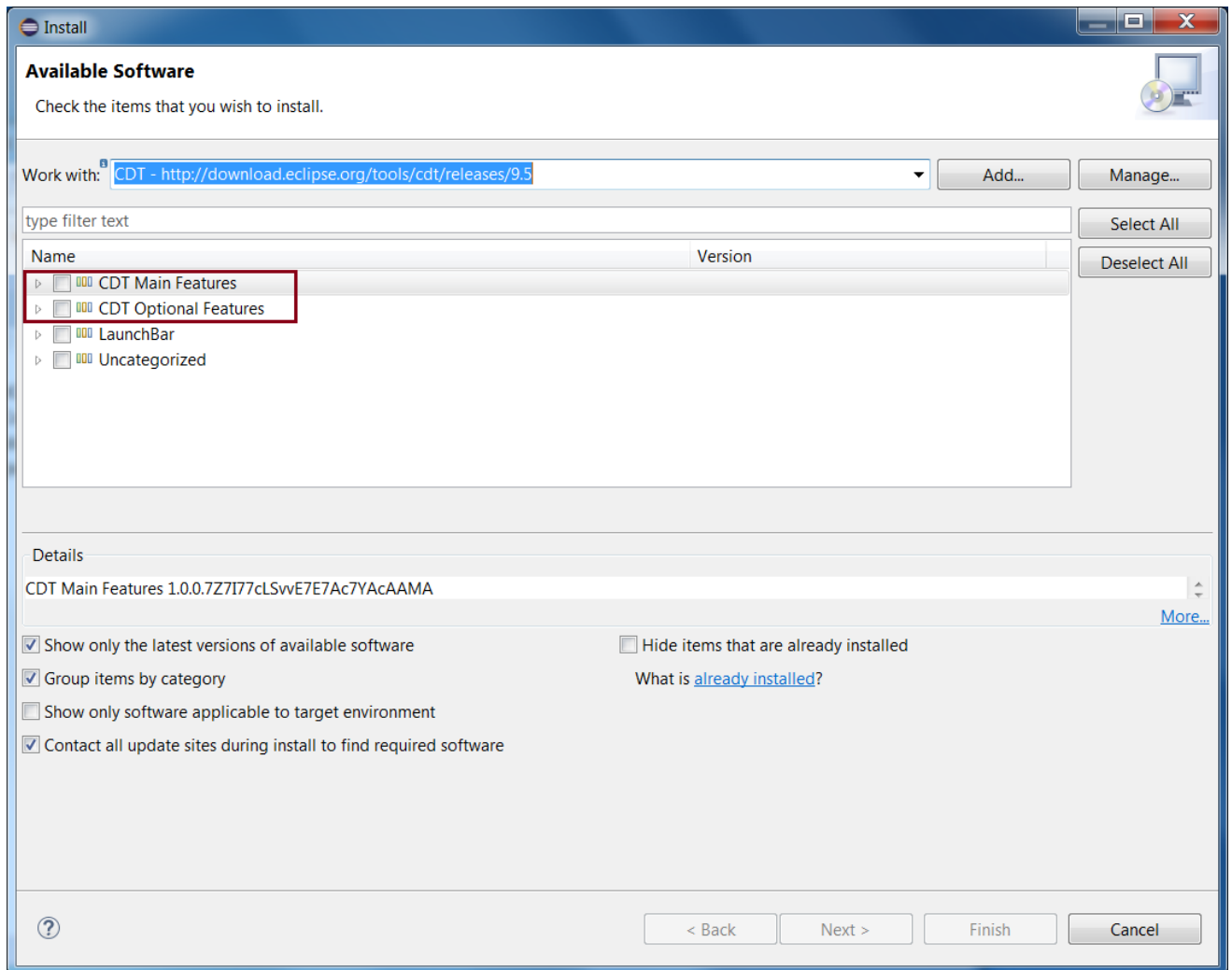
Eclipse需要另外安裝CDT才能支持C/C++開發程式，在Eclipse 的開發介面上點選Help -> Install New Software.



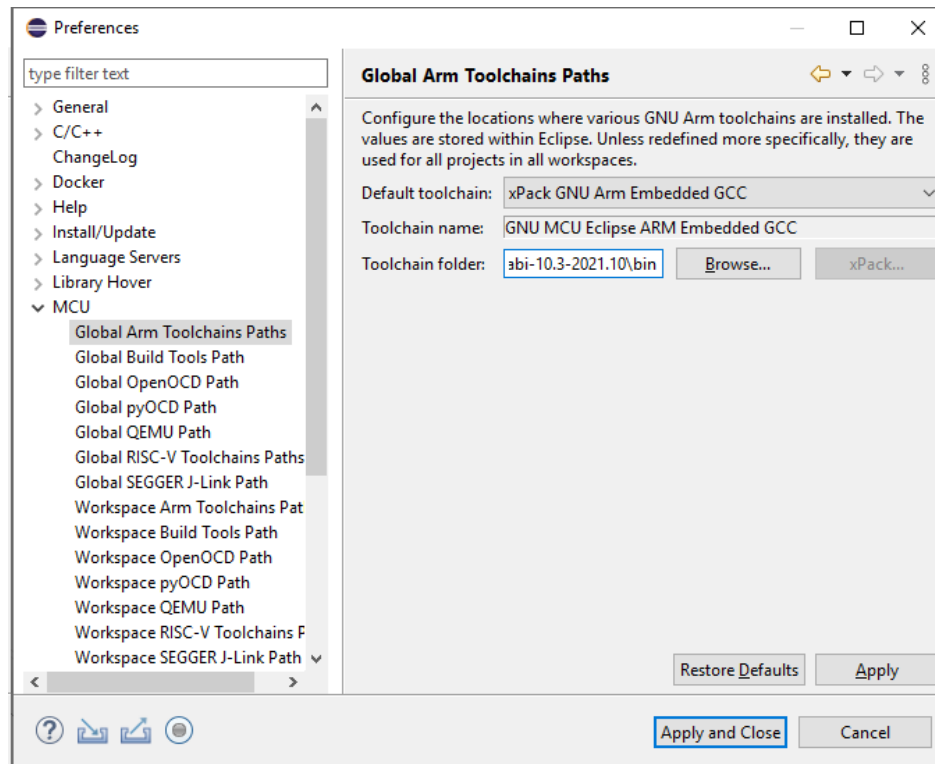
在work with欄位輸入CDT.



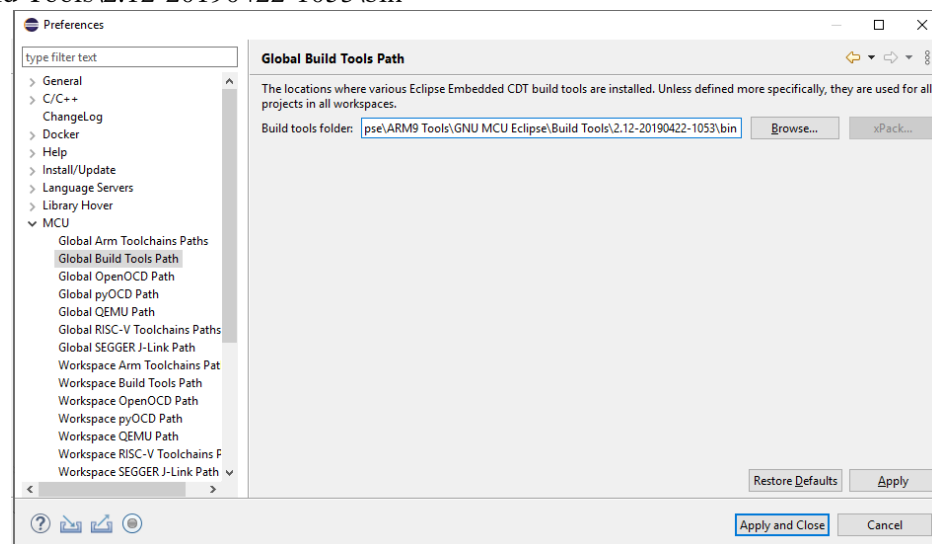
下方視窗顯示CDT Main Features和CDT Optional Features, 使用者可以依照需求勾選自訂的套件安裝或是全選安裝。



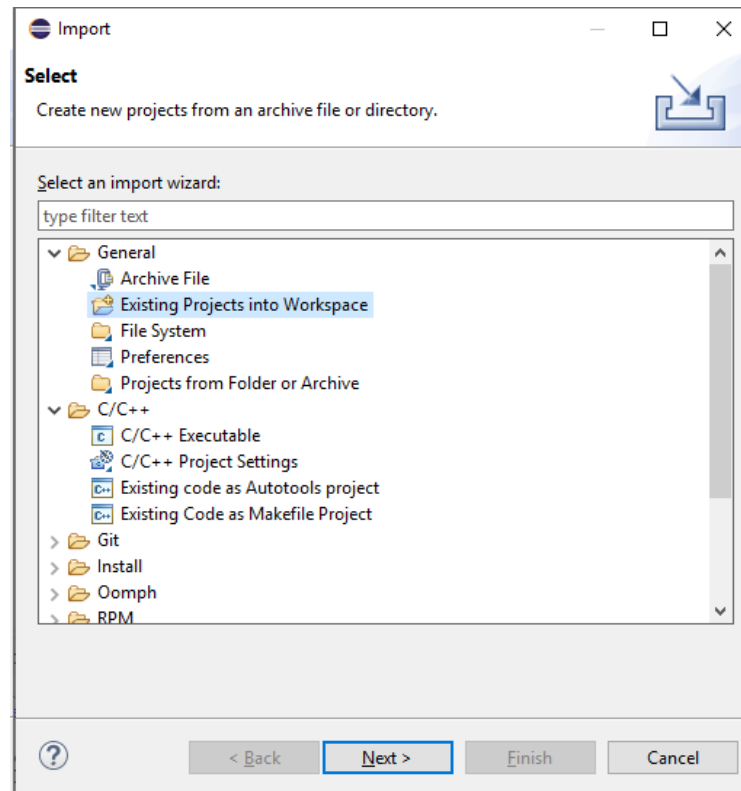
在安裝 CDT 之後，重啟 Eclipse。接著設定工具鏈，在前面步驟中已經下載解壓縮到本地 c:\eclipse 路徑下。在 Eclipse 主選單 “Windows” → “Preferences” 開啟 preference 視窗，然後選擇 “MCU” → “Global Arm Toolchains Paths”，然後選擇路徑 “C:\eclipse\gcc-arm-none-eabi-10.3-2021.10\bin”。



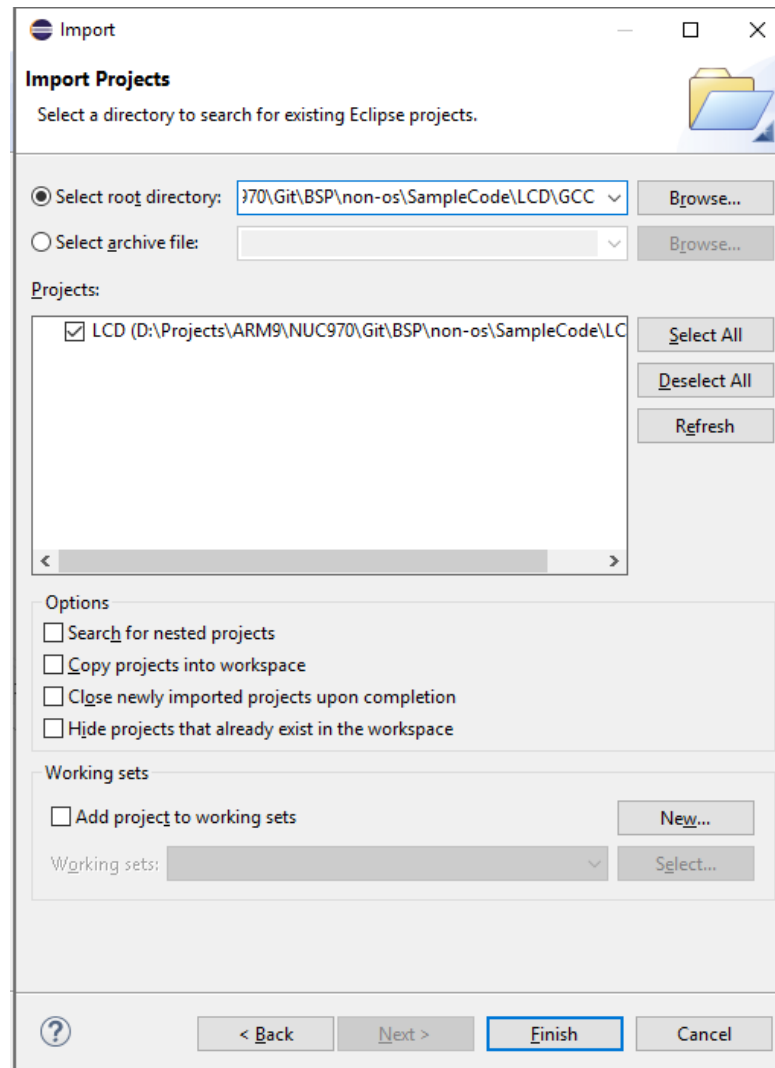
選擇下方的 “Global Build Tools Path” ，並瀏覽選擇到路徑 “C:\eclipse\GNU MCU Eclipse\Build Tools\2.12-20190422-1053\bin” 。



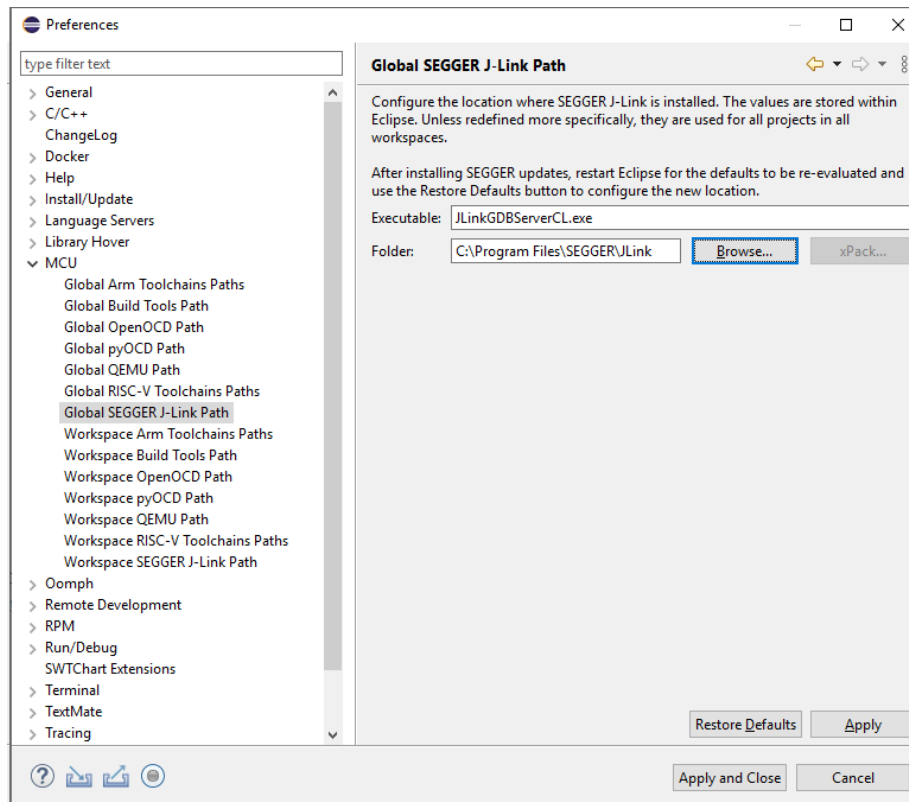
現在 Eclipse 環境應該已經設置好，可以進行 N9H30BSP 編譯。接下來可以從主選單 File → Import 導入一個 N9H30 範例專案。



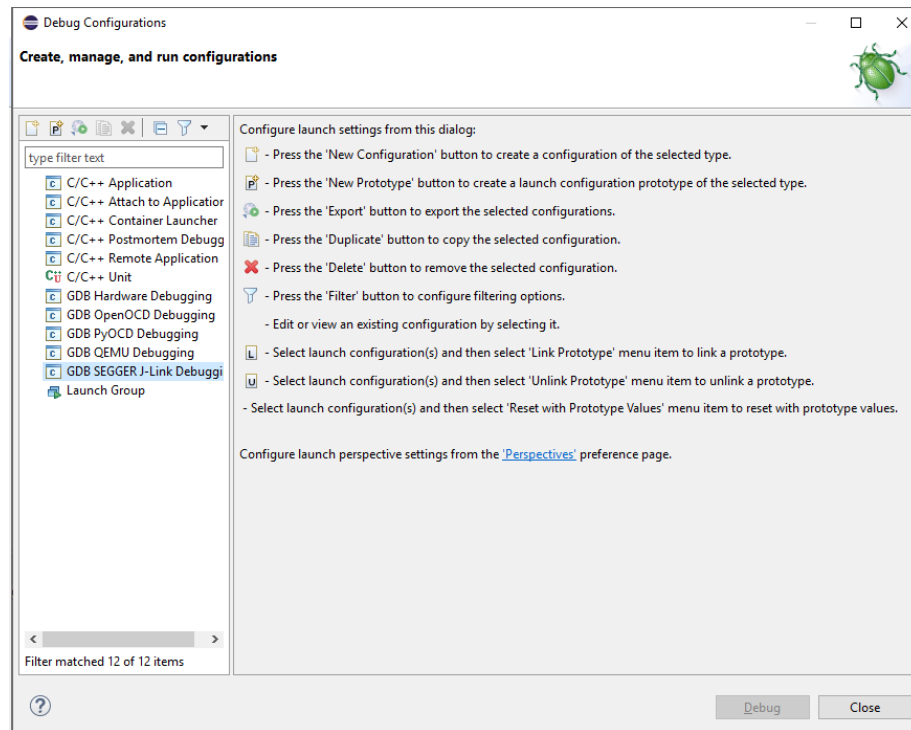
瀏覽選擇 GCC 專案，點擊 “Finish” 開啟專案。現在 Eclipse 應該可以編譯 N9H30 GCC 專案了。



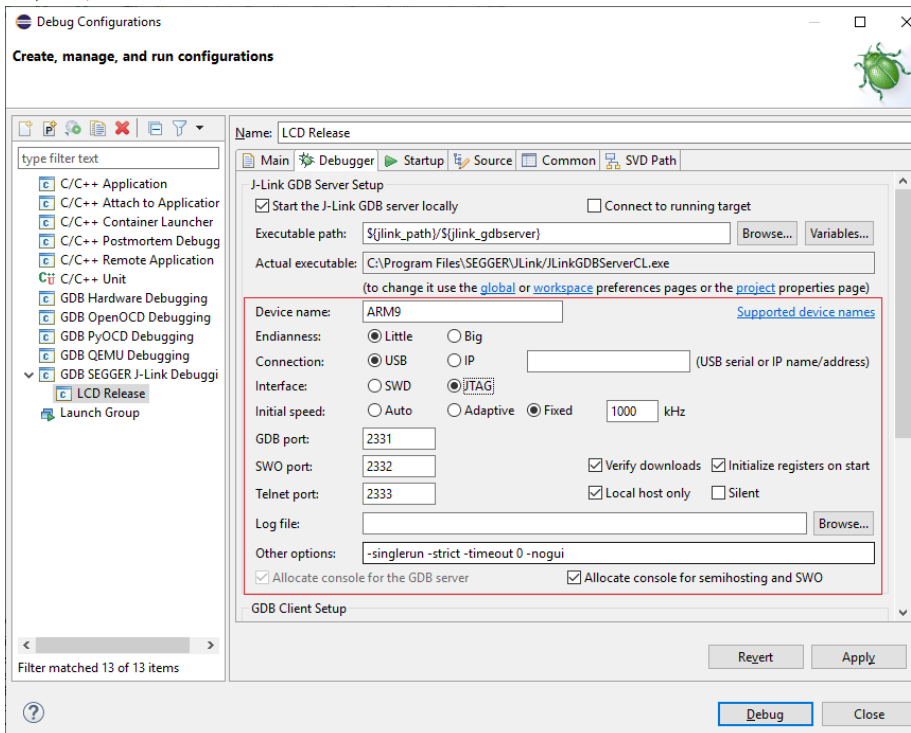
Eclipse 可搭配 J-Link 偵錯. 使用J-Link需要安裝GNU MCU C/C++ J-Link Debugging 套件, 安裝完成後, 進入開發介面後, 點選Windows->Preference->MCU中去設定J-Link Path , 設定後按下Apply按鈕即可完成J-Link Path設定.



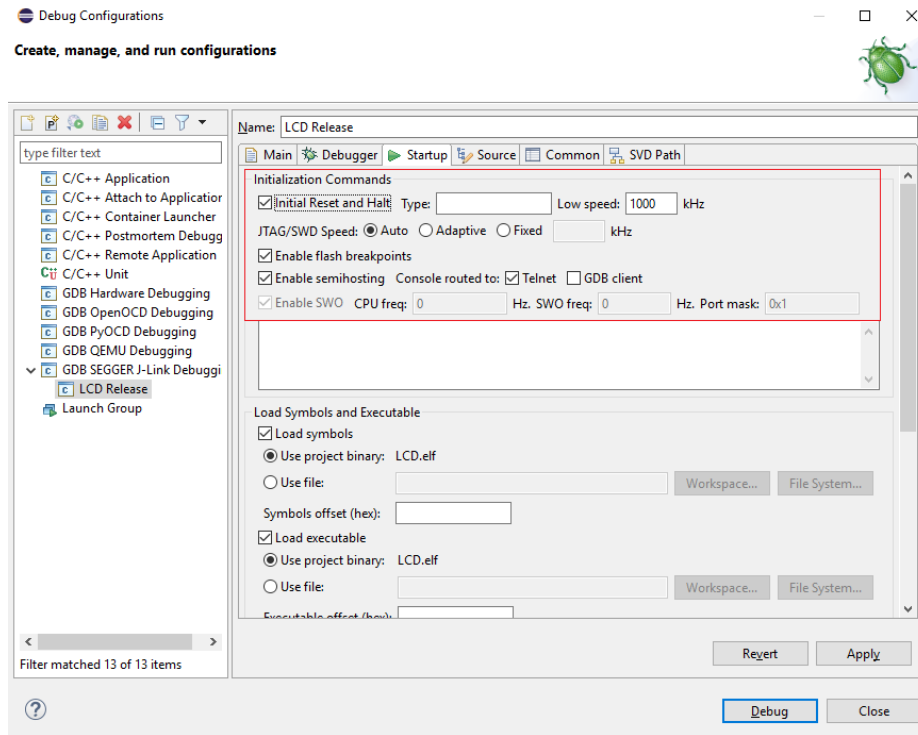
接下來，設定 Debug Configurations，在開發介面上點選Run-> Debug Configurations，接著滑鼠停在GDB SEGGER J-Link Debugging 選項上按兩次左鍵，



點選Debugger，並依照下圖設定。

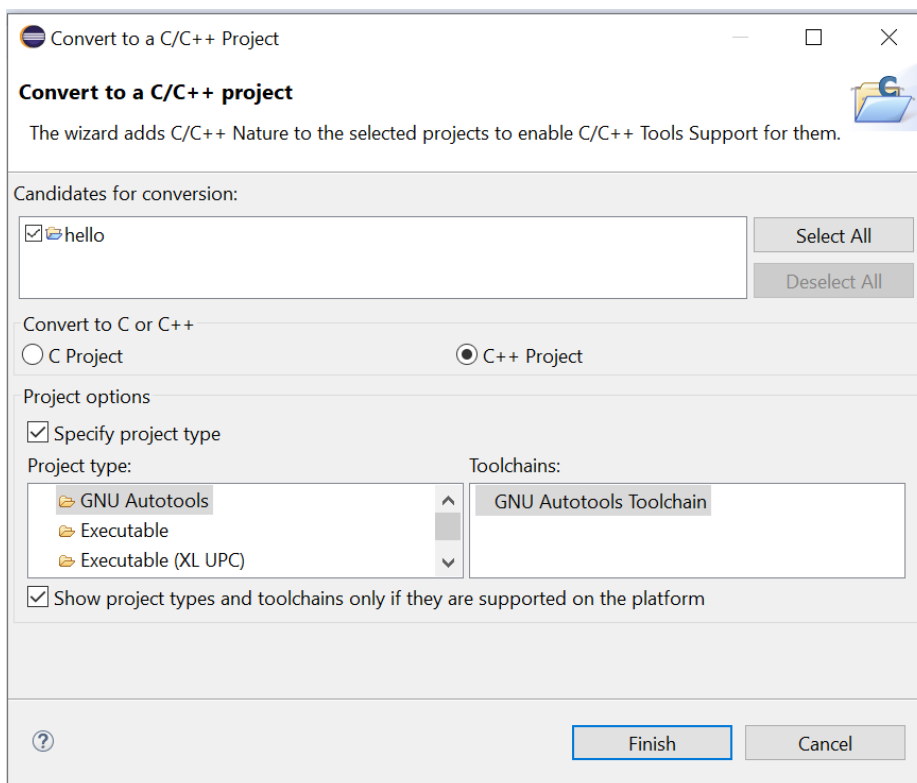
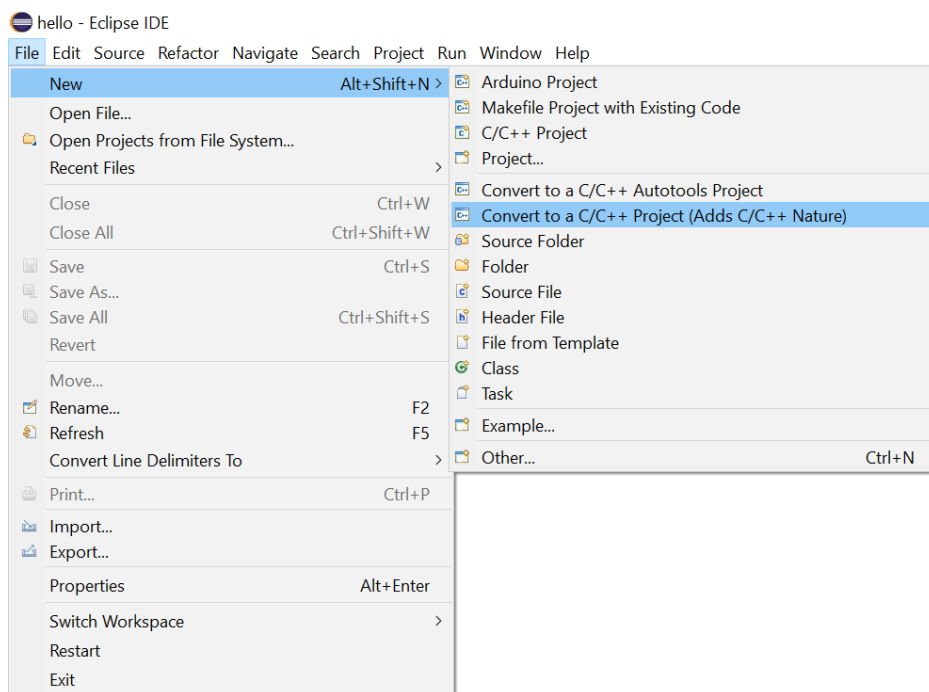


接下來點選Startup，此頁面的設定參考下圖，



完成上述設定後即可按下Debug 使用J-Link偵錯。

如果要將 C 專案轉換成 C++ 專案。首先，從 File ➔ Import 導入一個 Eclipse C 專案，然後選擇 convert C++ from File ➔ New ➔ Convert to a C/C++ Project (Adds C/C++ Nature)。然後在開啟窗依序完成即可。



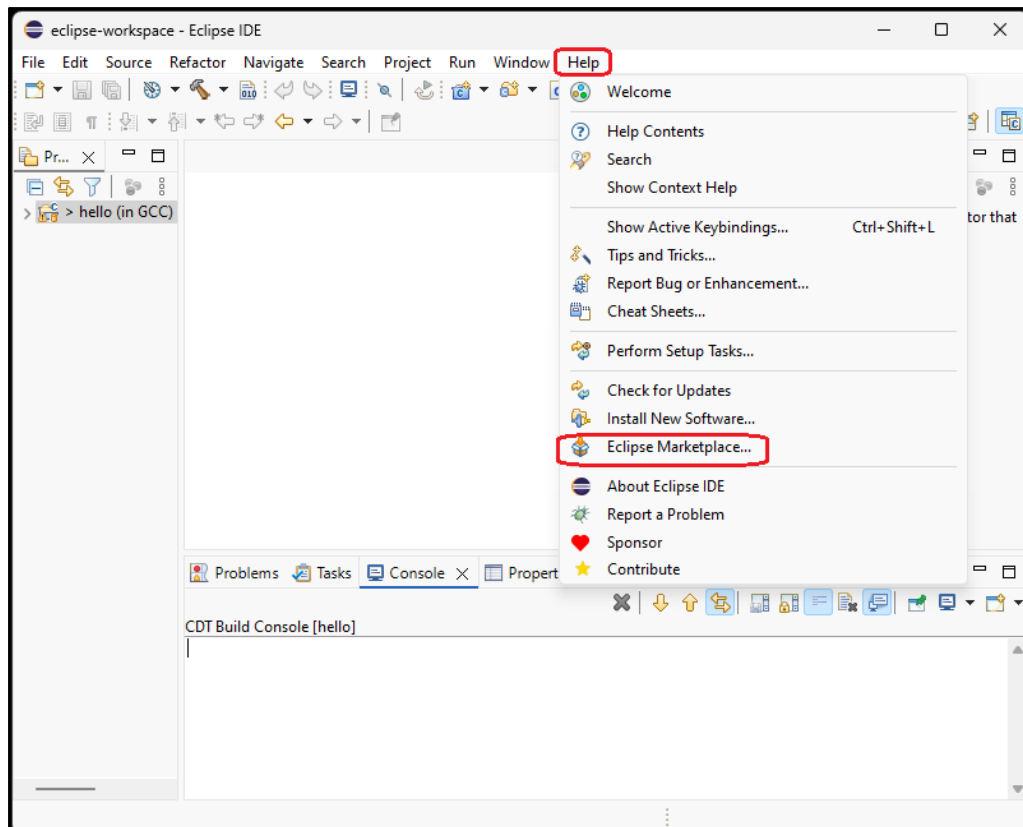
1.3 Eclipse 開發環境 (2025-09 版本以上)

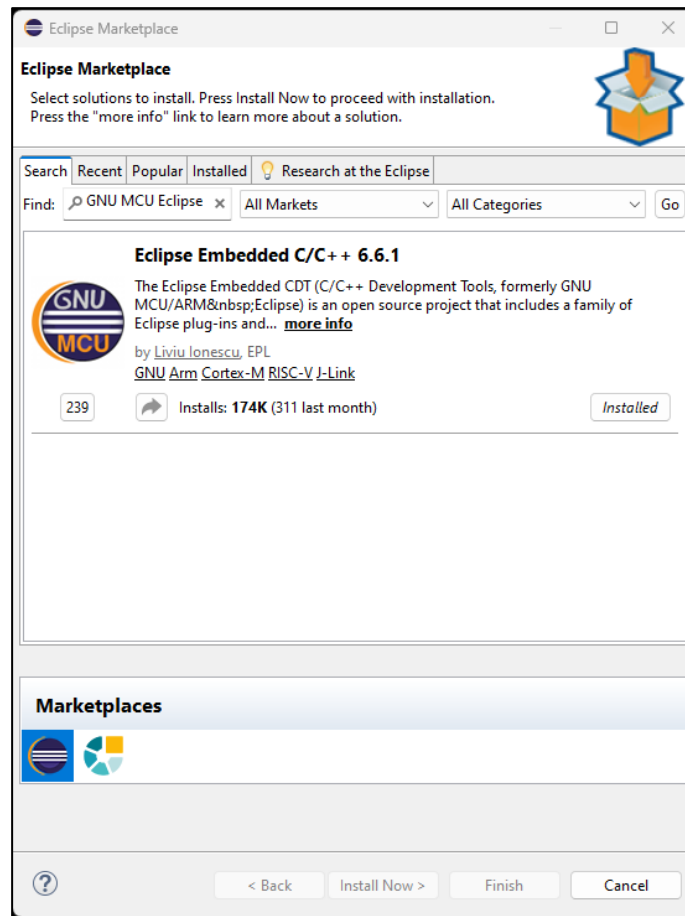
N9H30 BSP 也支援 Eclipse 開發環境，本節將介紹 Eclipse 的安裝步驟。首先，使用者至 Eclipse 官方網站<https://www.eclipse.org/downloads/> 下載 Eclipse IDE for Embedded C/C++ Developers Tool 開發工具，選擇適用作業系統及位元的版本。最近發行的 Eclipse 版本已包含 Java，因此不再需要單獨安裝 Java。

從 <https://github.com/xpack-dev-tools/windows-build-tools-xpack/releases> 下載編程工具。選擇 “GNU MCU Eclipse Windows Build Tools v2.12 20190422”，然後下載 “gnu-mcu-eclipse-windows-build-tools-2.12-20190422-1053-win64.zip”，並壓縮到本地 C:\eclipse 目錄下。

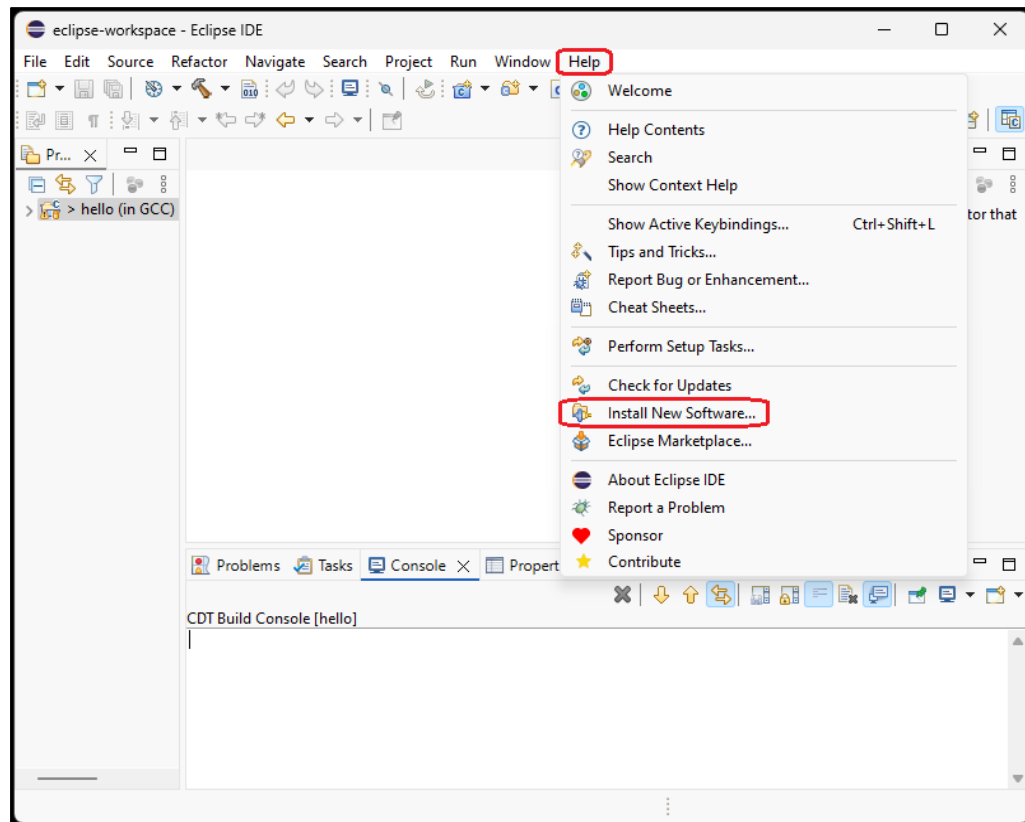
從 <https://developer.arm.com/downloads/-/gnu-rm> 下載 GCC 工具鏈。選擇 “gcc-arm-none-eabi-10.3-2021.10-win32.zip” 下載，並壓縮到本地 C:\eclipse 目錄下。

另外可以透過 Eclipse Marketplace 或是 Eclipse Install New Software，將外掛程式安裝到現有的 Eclipse 基礎上。使用 Eclipse Marketplace 安裝 Eclipse Embedded CDT 外掛。請依序點選 Eclipse 功能表 > Help > Eclipse Marketplace。在 Find 欄位中輸入 GNU MCU Eclipse，使用者在搜尋結果中選擇最新版本，然後點選 Install 安裝，補齊相關 plug in 套件 (GNU MCU C/C++)。

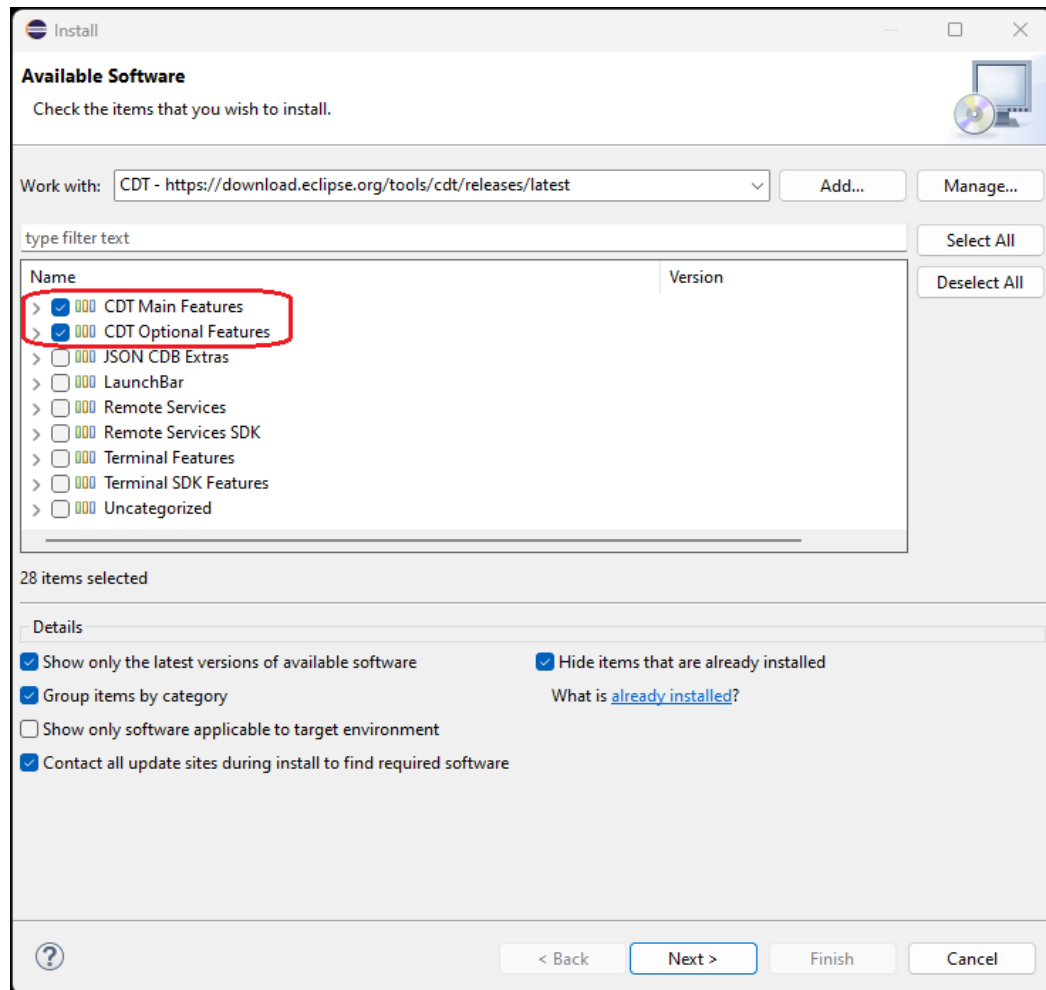




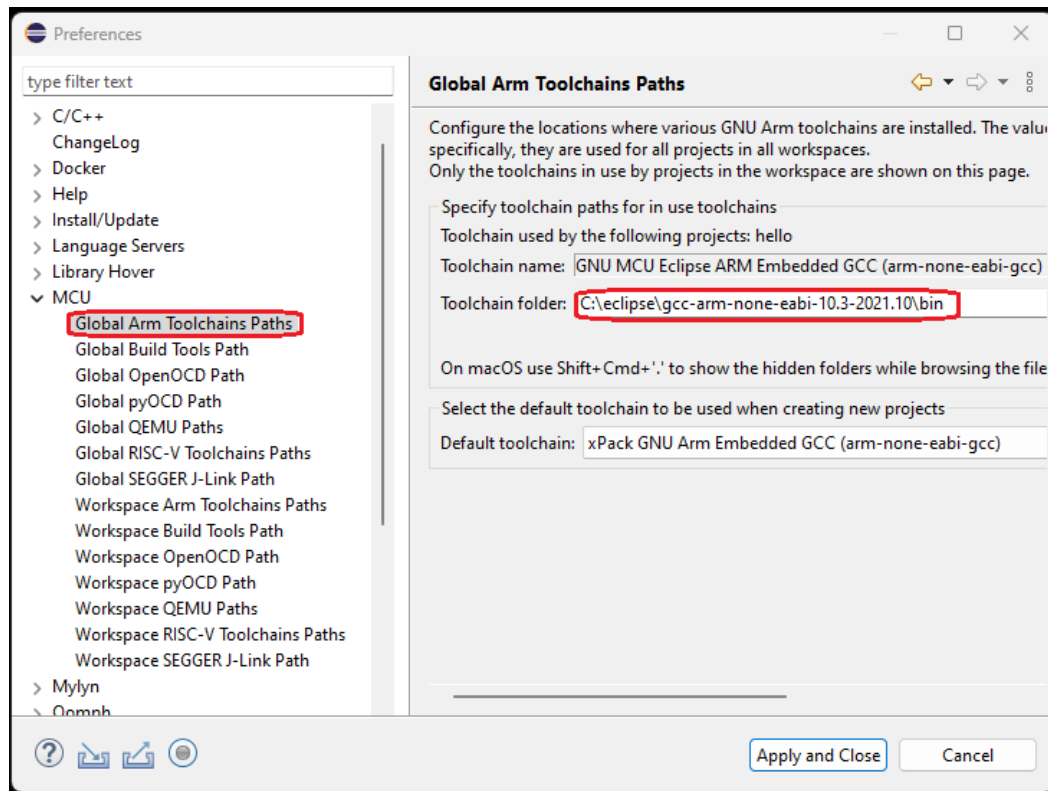
Eclipse 需要另外安裝 CDT 才能支持 C/C++ 開發程式，在 Eclipse 的開發介面上點選 **Help > Install New Software**。



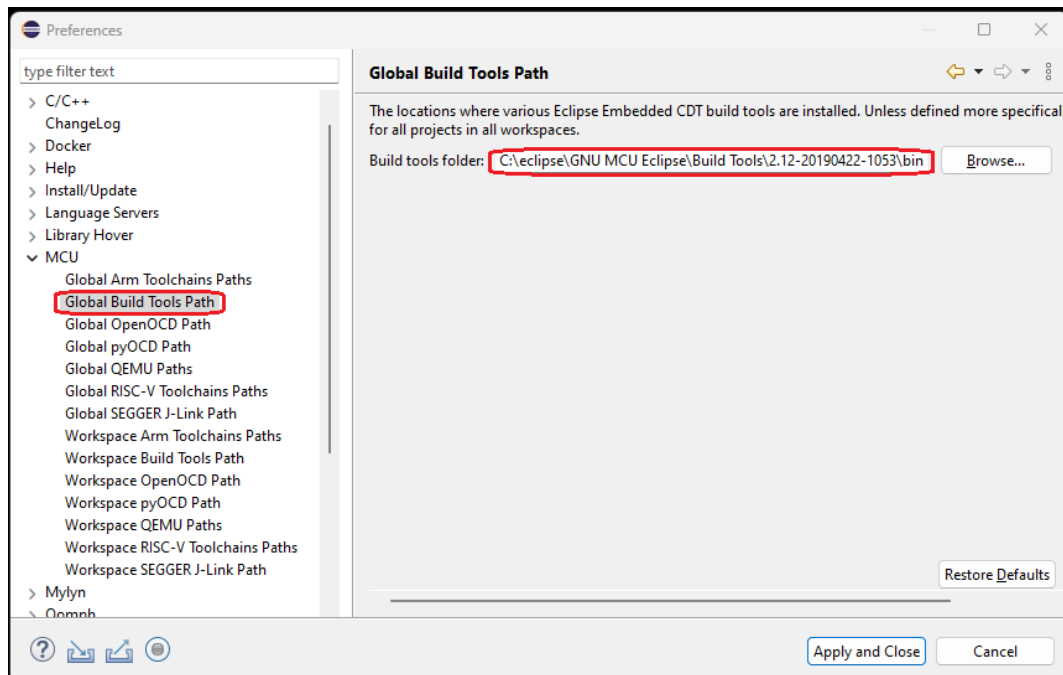
在 Work with 欄位輸入 **CDT**，下方視窗顯示 CDT Main Features 和 CDT Optional Features，使用者可以依照需求勾選自訂的套件安裝或是全選安裝。安裝 CDT 之後，請重啟 Eclipse。



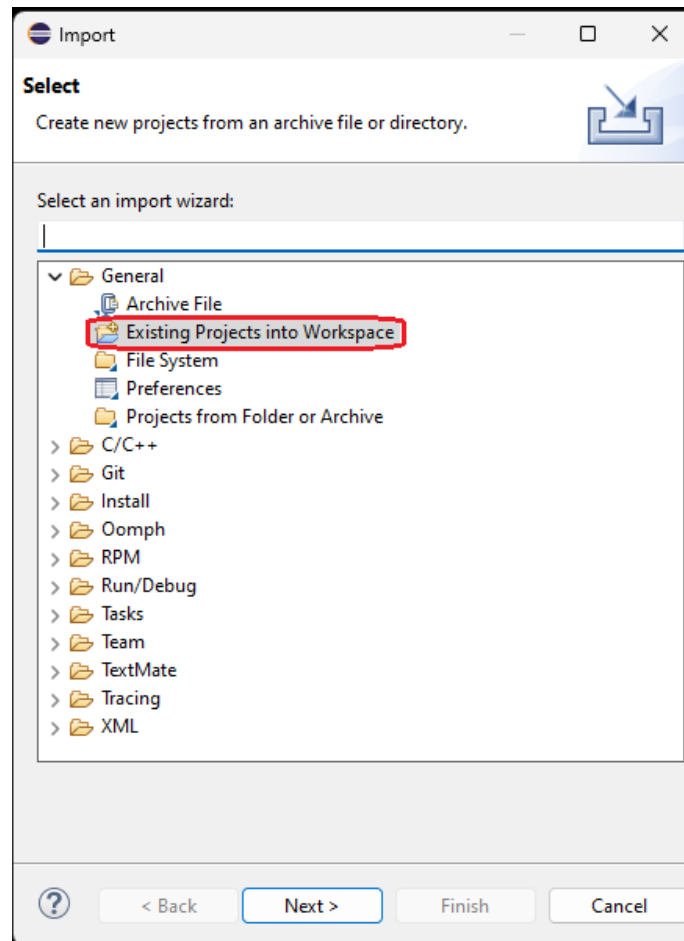
接著設定工具鏈，在前面步驟中已經下載解壓縮到本地 `c:\eclipse` 路徑下。在 Eclipse 主選單 **Windows > Preferences** 開啟 preference 視窗，然後選擇 **MCU > Global Arm Toolchains Paths**，設定路徑 “`C:\eclipse\gcc-arm-none-eabi-10.3-2021.10\bin`”。



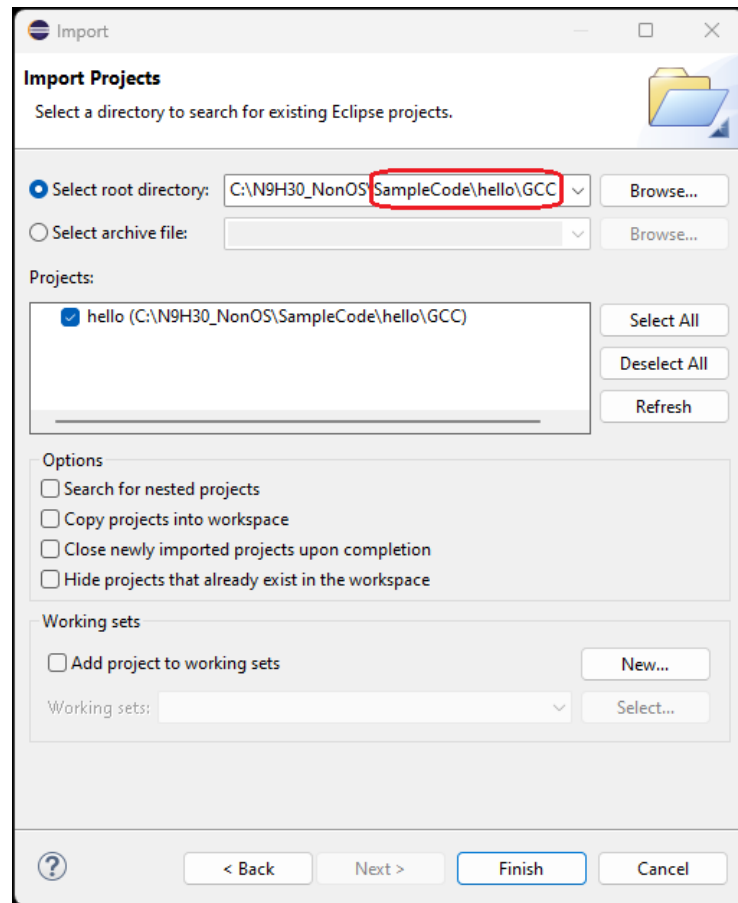
選擇下方的 **Global Build Tools Path**，並瀏覽選擇到路徑 “C:\eclipse\GNU MCU Eclipse\Build Tools\2.12-20190422-1053\bin”。



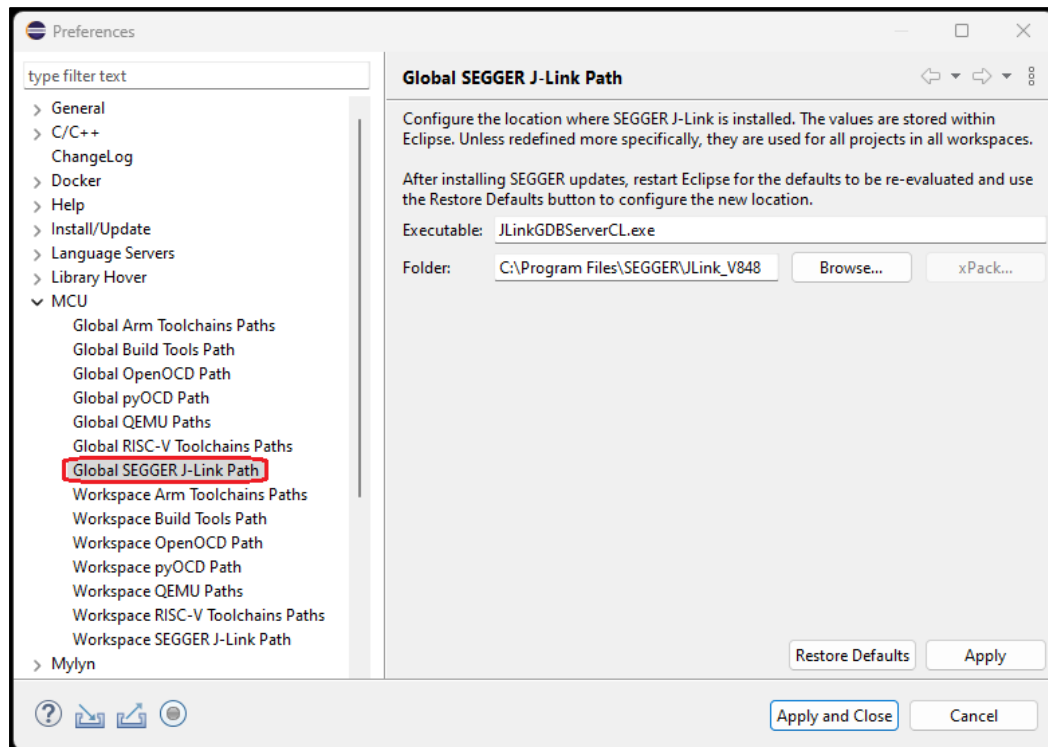
現在 Eclipse 環境已經設置好，可以進行 N9H30 BSP 編譯。接下來從主選單 **File > Import** 導入一個 N9H30 範例專案，選擇 **General > Existing Project into Workspace**，然後點選 **Next**。



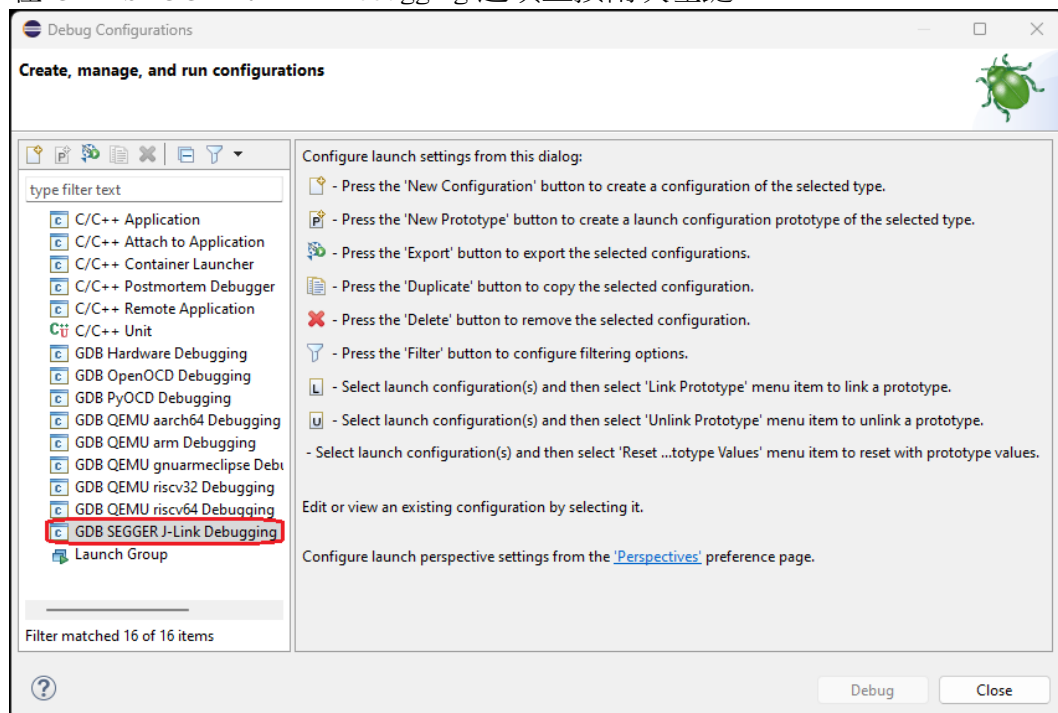
瀏覽並選擇 GCC 專案，點擊 **Finish** 開啟專案。現在 Eclipse 可以編譯 N9H30 GCC 專案了。



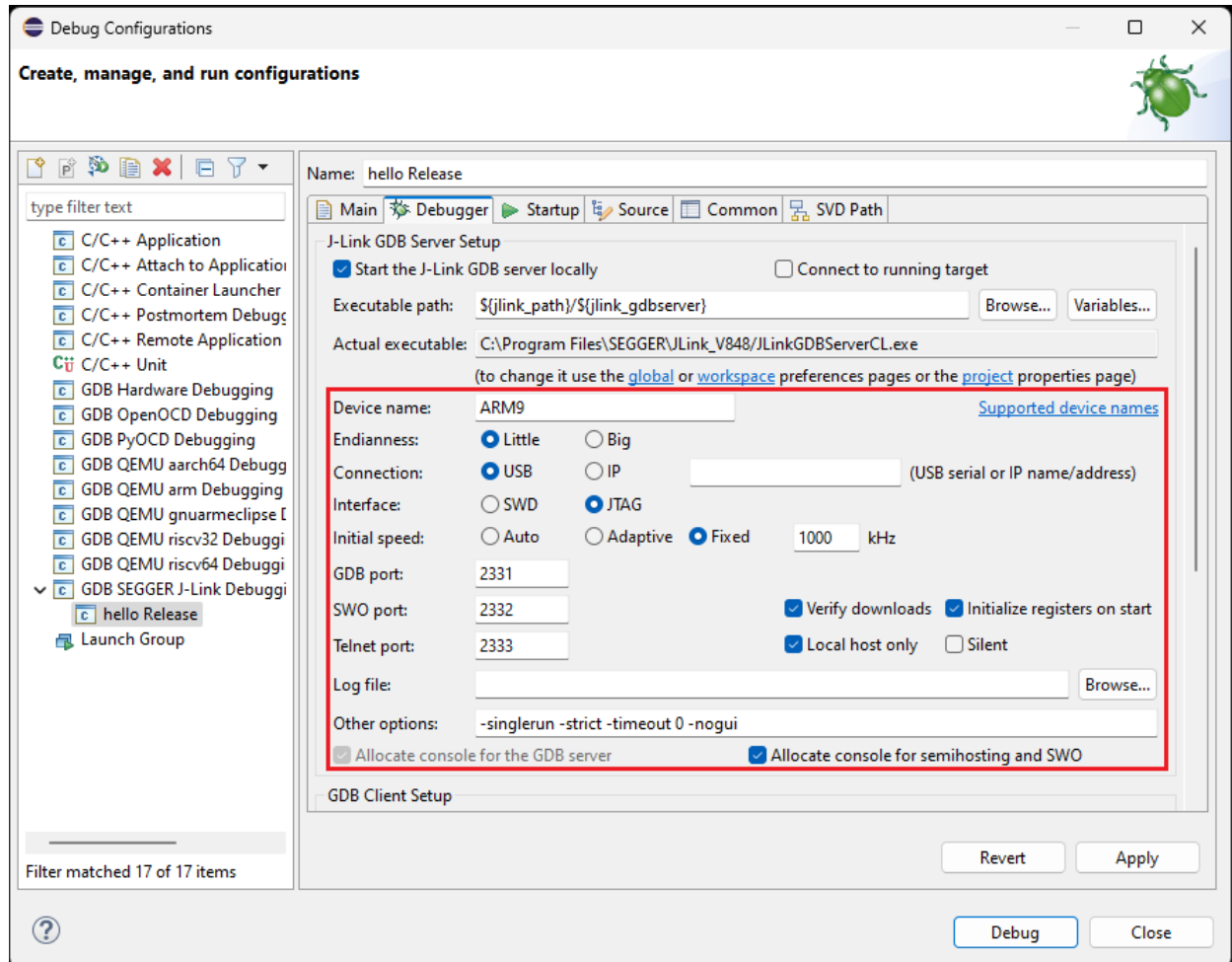
Eclipse 可搭配 J-Link 進行除錯。在開始除錯之前，請先從以下網站下載並安裝 J-Link 軟體套件：<https://www.segger.com/downloads/jlink/>。安裝完成後，透過 Eclipse 功能表依序點選 **Window > Preferences > MCU > Global SEGGER J-Link Path** 設定 J-Link 路徑，然後點擊 **Apply** 完成設定。



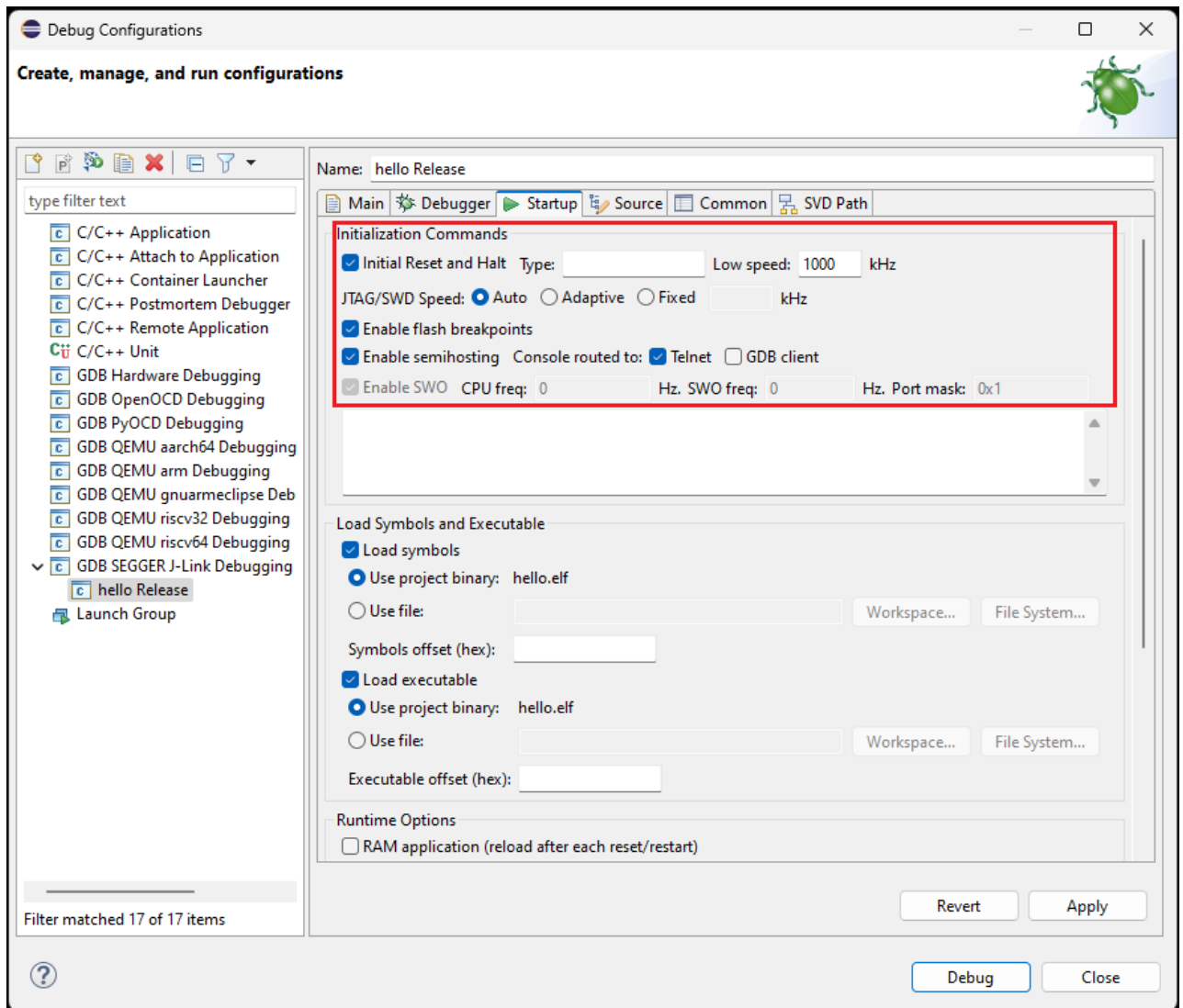
接下來，設定 Debug Configurations。在開發介面上點選 **Run > Debug Configurations**，接著滑鼠停在 **GDB SEGGER J-Link Debugging** 選項上按兩次左鍵。



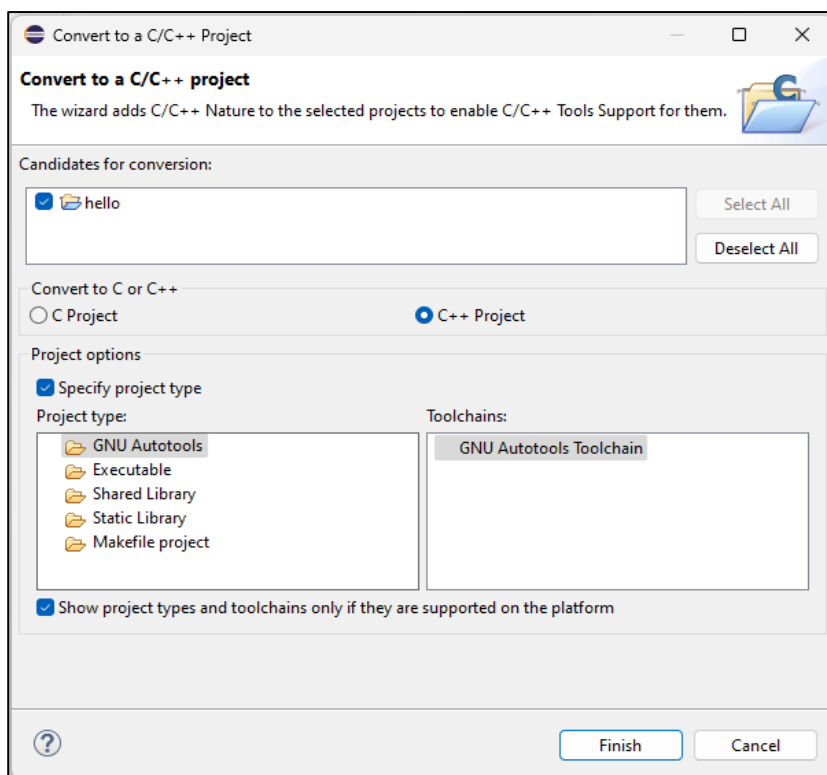
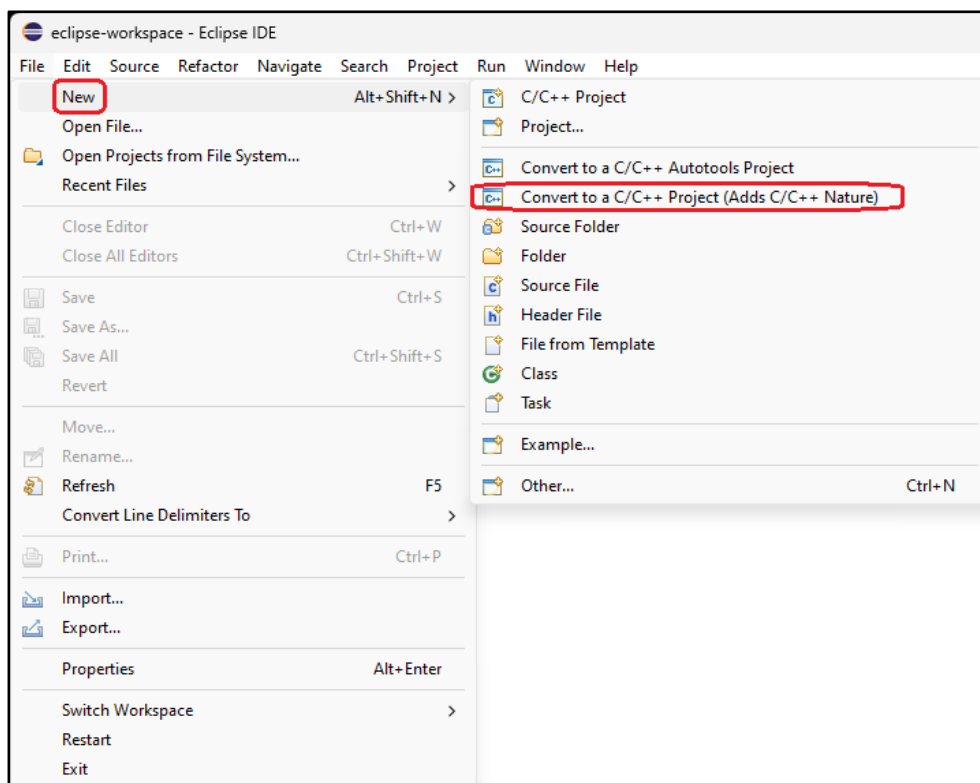
點選 **Debugger**，並依照下圖設定。



接下來點選 **Startup**，此頁面的設定參考下圖。完成所有設定後，點擊 **Apply** 使設定生效。接著，確認 N9H30 開發板已設定為從 USB 開機，並且已連接至 NuWriter 以進行 DDR 初始化。完成這些步驟後，點擊 **Debug** 開始使用 J-Link 進行除錯。



繼續在 Eclipse 中進行操作，將 C 專案轉換成 C++ 專案。首先，從 **File > Import** 導入一個 Eclipse C 專案，然後選擇 **File > New > Convert to a C/C++ Project (Adds C/C++ Nature)**。最後點擊 **Finish** 以完成專案轉換。



1.4 開發板設置

N9H30 系列芯片支持不同的開機模式, 可從 SPI, NAND, eMMC 開機, 或是進入 USB ISP 模式. 這些設置是透過 PA[1:0] 的 power on setting 控制. 請參考開發版的文件來做系統相應的設置.

2 BSP 內容

2.1 BSP 目錄架構

Non-OS BSP 包含了四個目錄, 各目錄的內容列在下表:

目錄名稱	內容
BSP	目錄下包含了 Non-OS 驅動程式, 第三方軟體, 以及範例程式.
Documents	BSP 相關文件
Images	預先編譯好的 U-Boot 映像檔.
Tools	Windows 上的燒錄工具以及驅動程式

2.2 Non-OS BSP 內容

BSP 目錄下有以下內容:

目錄名稱	內容
Driver	N9H30 各個周邊的驅動程式. 個驅動程式的 API 說明請參考在 Document 目錄下的 N9H30 Non-OS BSP Driver Reference Guide.chm 文件.
Library	N9H30 使用的函數庫. 例如USB Host.
SampleCode	驅動相關範例程式
Script	包含了Keil 連結時的 link script. 以及要進入偵錯模式使用的腳本.
ThirdParty	第三方軟體, 包含了FATFS 文件系統, 以及 LwIP 開源TCP/IP協議棧.

3 NuWriter

NuWriter 工具能幫助使用者透過 USB ISP模式, 將映像檔寫入儲存體中, 例如: SPI Flash, NAND Flash, 或是 eMMC. 使用方式請參考 N9H30 NuWriter User Manual 文件.

4 版本歷史

版本號	日期	描述
1.00	Jan. 31, 2018	初版發布
1.01	Jul. 5, 2018	內容更新
1.02	Dec. 24, 2018	內容更新
1.10	May 31, 2018	描述 Eclipse 開發環境
1.20	Mar. 7, 2022	移除 emWin
1.30	Aug. 8, 2022	更新 Eclipse 開發環境說明
1.31	Dec. 16, 2022	新增 C/C++ 專案轉換說明
1.32	Sep. 23, 2025	更新 Eclipse 開發環境說明 (2025-09 版本以上)

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, “Insecure Usage”.

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer’s risk, and in the event that third parties lay claims to Nuvoton as a result of customer’s Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton

*Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.*