Exercise 1: Setting up Free toolchain For ARM® Cortex®-M Application development Eclipse GNU C/C++ GDB Server for Windows

Revision History

Revision Number	Date	Author	Description
0.1	04Oct2014	JW	First draft.
0.2	26Nov2014	JW	Generalized Eclipse installation. Added SEGGER J-Link GDB support.
0.3	04Feb2015	JW	Updated GNU tools section. Updated Eclipse section. Fixed table of contents links.
0.4	10Aug2016	JW	Updated SEGGER J-Link support to version 6.00e
0.5	13Nov2022	JW	Updated to latest Eclipse (Embedded) Updated link to ARM GNU Tool chain (GCC compiler/linker) Added NXP example project (LPC824) Added SDK for MCUXpresso Config tool
0.6	06Dec2022	JW	Added MAKE support (2.2.1 NOTE 8)

Audience

This document is intended for the ARM Cortex-M beginner and seasoned developers interested in evaluating ARM Cortex-M platforms.

Table of Contents

1.0 Installing GNU Tools for ARM Embedded Processors (arm-none-eabi-gcc)

The latest version of GNU tools for ARM embedded processors can be downloaded at [Arm GNU Toolchain](). Select the *Download Arm GNU Toolchain* button, follow instructions. Once downloaded, use the *gcc-arm-none-eabi-xx.x-20xx.xx-win32.exe* to install the toolchain. Do not include it in your PATH as developers can use many versions of the arm-none-eabi-gcc tools.

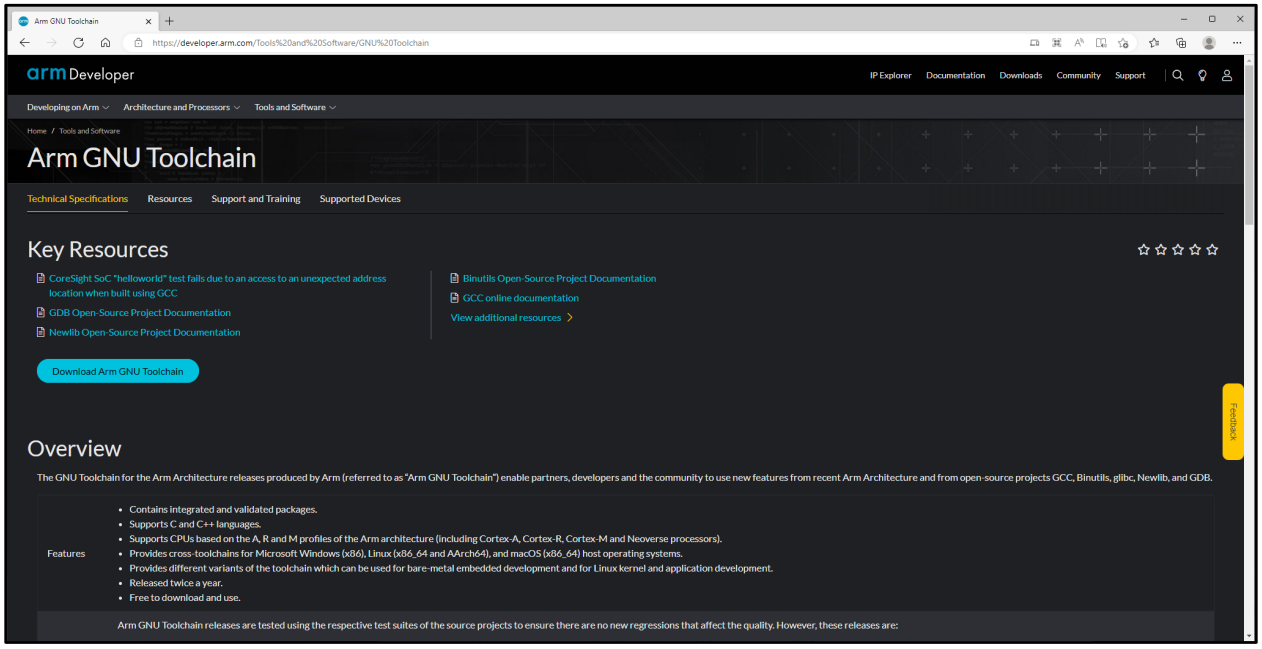


Figure 1 Arm GNU Toolchain

2.0 Installing Eclipse

The eclipse IDE requires the JAVA runtime engine so if you do not have JAVA RTE installed to go http://www.java.com/en/download/ and install it. Note, as the tool chains function in both 32-bit and 64-bit architecture, install both the 32-bit and 64-bit versions of JAVA.

Note, the Eclipse installer for 2022-09 R includes a JAVA Runtime Environment for Windows, as well as SEGGER J-link support.

Once you have verified that JAVA RTE has been installed go to https://www.eclipse.org/downloads/packages/release/2022-09/r/eclipse-ide-embedded-cc-developers and download the latest versions of ***Eclipse IDE for C/C++ Developers*** and install it (Eclipse IDE 2022-09, as of v 0.5 of this document). Select the *Download x86_64* button, follow instructions. Please do not forget to donate to support the Eclipse community.

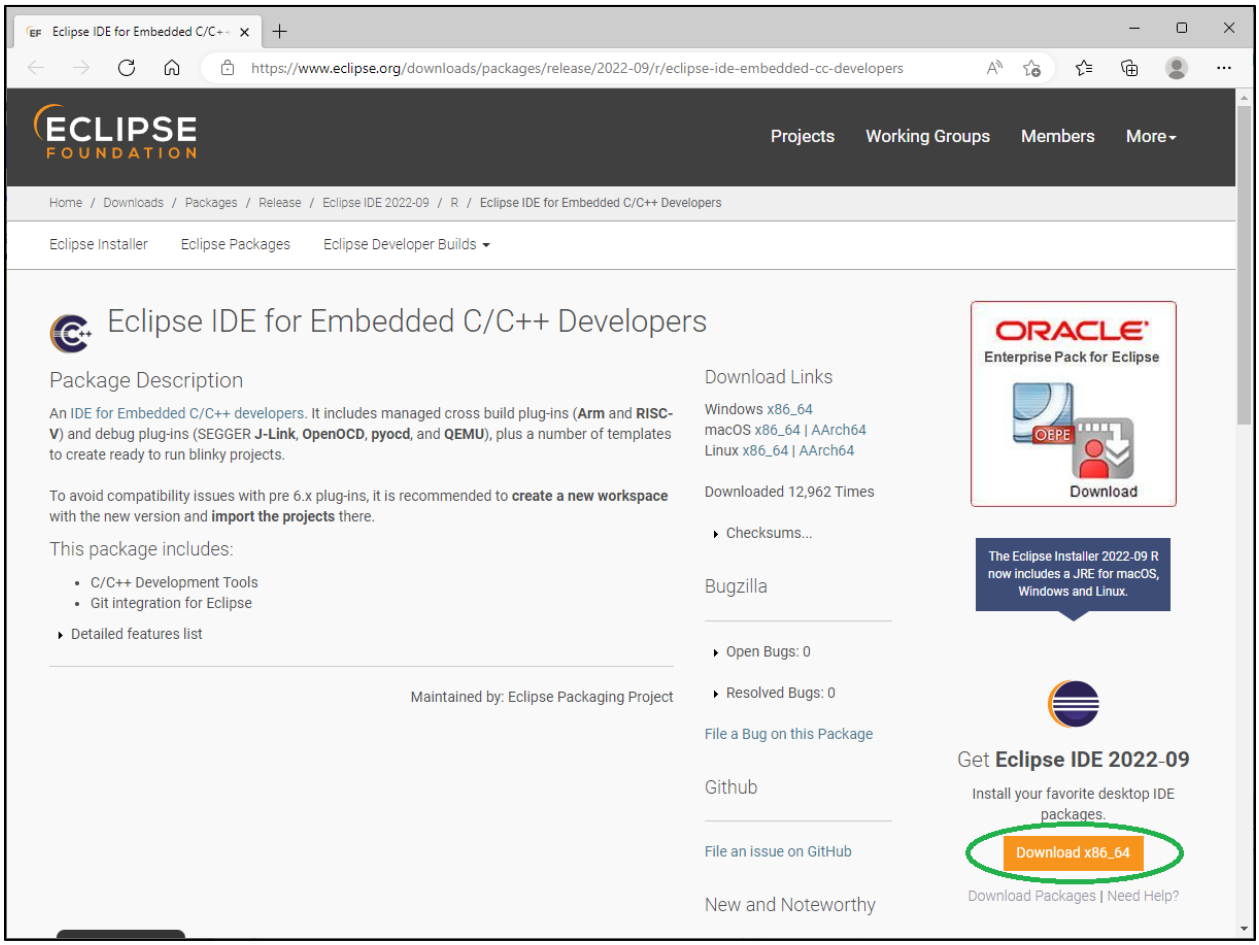


Figure 2 Eclipse Foundation CDT page,

Installing the Eclipse IDE is straight forward; launch the installer, eclipse-inst-jre-win64. Select Eclipse IDE for Embedded C/C++ Developers (this Eclipse release contains plug-ins that simplify ARM tools setup and embedded application development). As Eclipse is updated regularly, extract the archive in a directory suited to the version, i.e. for this exercise *C:\GCC-Eclipse-NXP-DEMO*.

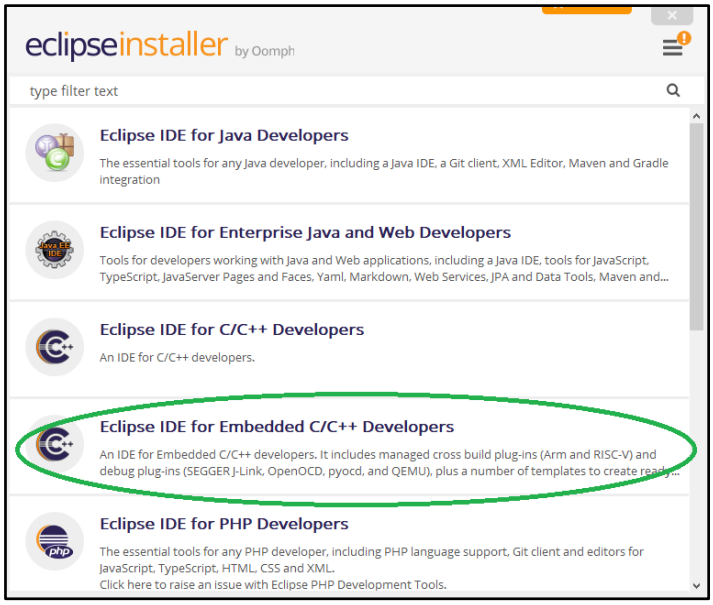


Figure 3 Select Eclipse IDE for Embedded C/C++ Developers

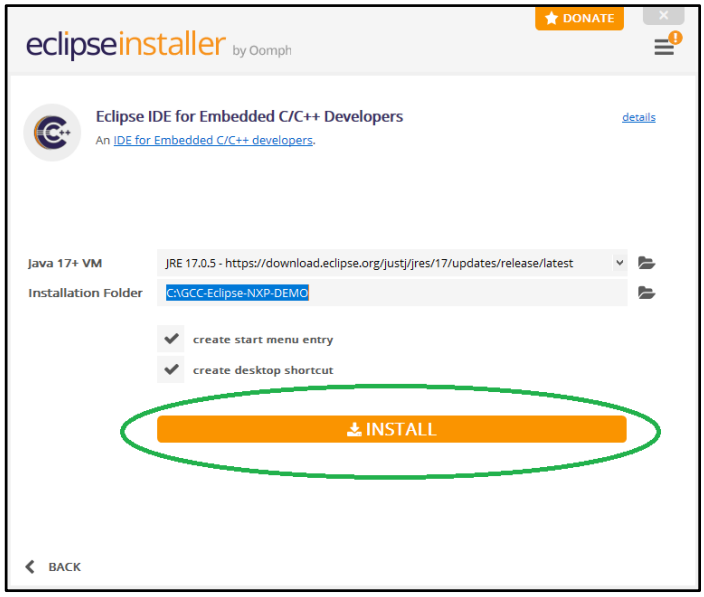


Figure 4 Eclipse Installer folders

2.1 Tool chain demo projects

For this exercise we use NXP's *LPCXpresso board for LPC824* (single CORE Cortex-M0+) PN: OM13071, an inexpensive and readily available eval board for ARM Cortex M0+ development. In this section we will describe how to create the Eclipse project "Hello World." Please note that the steps may be exhaustive however they are for the benefit of new users to the Eclipse development environment.

NOTEs:

- This project is limited to ARM® Cortex™-M0+ core registers and basic modification to the linker script for LPC824 memory map.
- Projects use a Segger J-LINK debug probe.
- Exercises are path dependent.

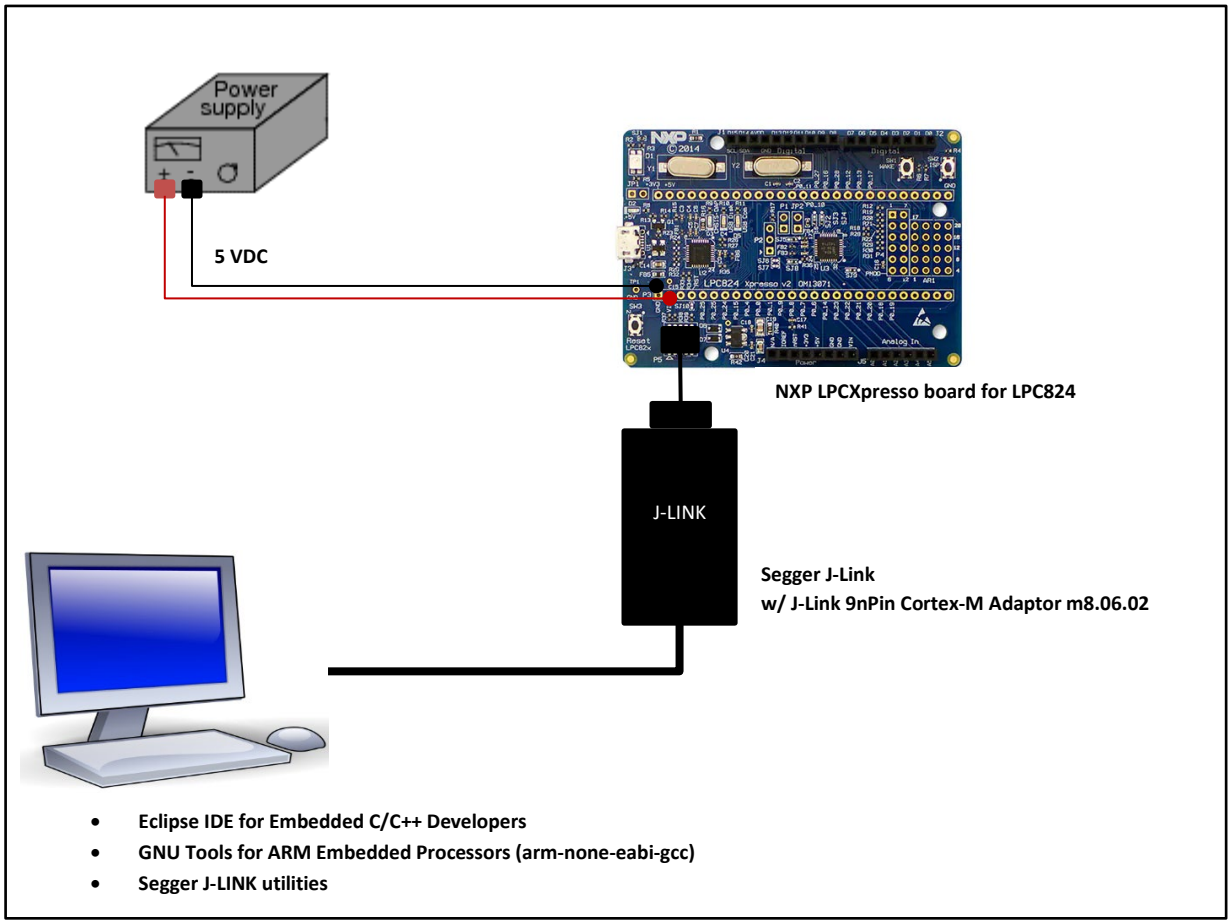
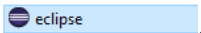


Figure 5 Demo system block diagram

2.1.1 Hello World

Go to the Eclipse install directory, our path "*C:\GCC-Eclipse-NXP-DEMO\eclipse*" and launch ***eclipse.exe***, 

.

1. In the Eclipse IDE Launcher dialog Workspace field enter "C:\GCC-Eclipse-NXP-DEMO\XPRESSO-LPC824 then select the *Launch* button.

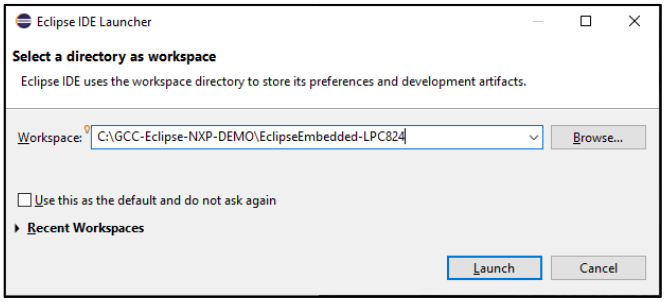


Figure 6 Project workspace

2. Select *File->New->C/C++ project:*

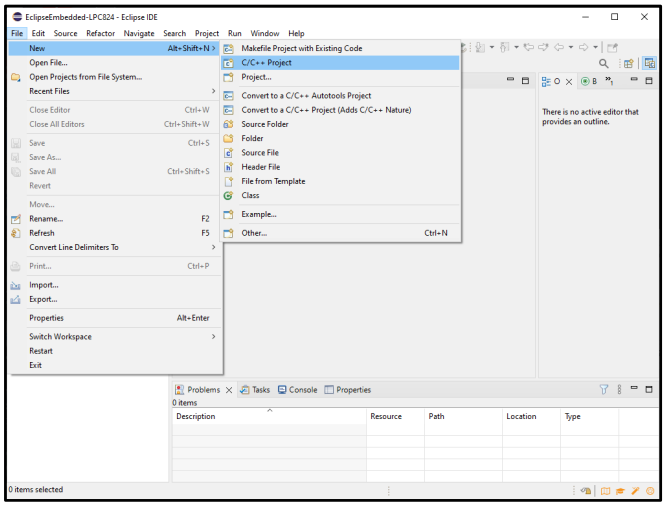


Figure 7 New Project

3. Use the *C Managed Build* Template then select *Next>* button:

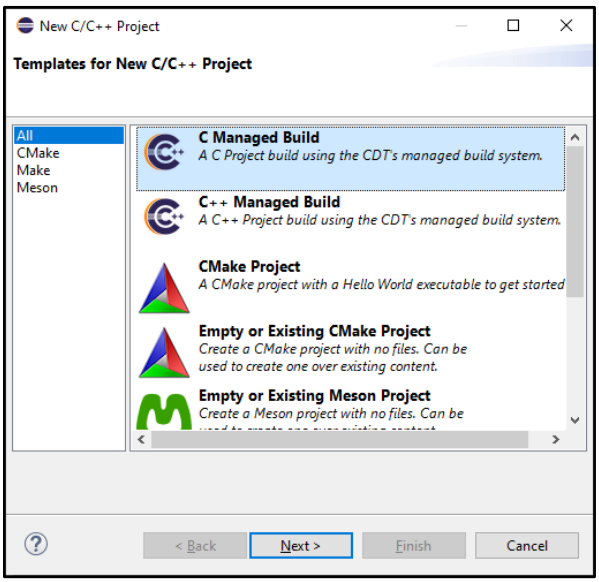


Figure 8 Eclipse managed C project

4. Project type *Hello World Arm Cortex-M C/C++* then select the *Next>* button"

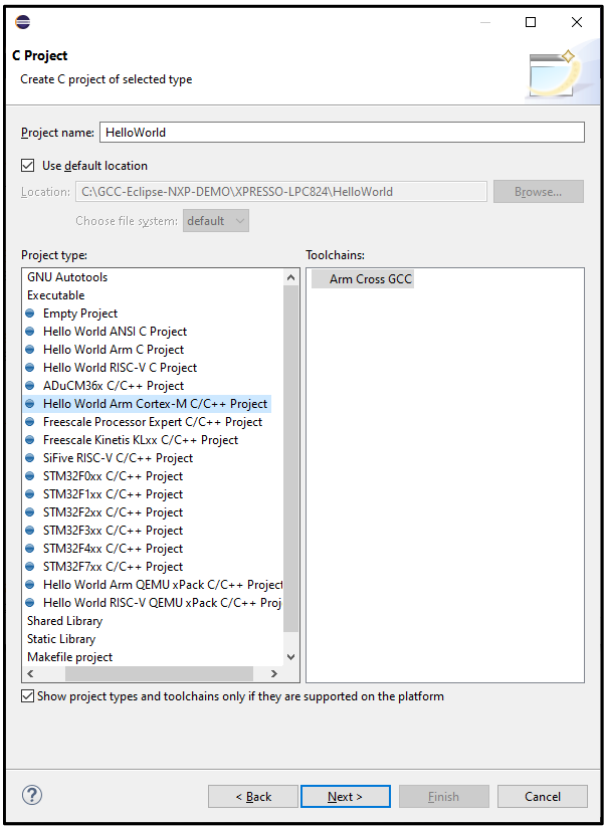


Figure 9 Default "Hello World!" Project

5. The LPC824 is a Cortex-M0+ device with 32K of FLASH, 8K of RAM and use the internal 12 MHz oscillator. The first four (4) fields will have to be set to those values below then select *Next>*:

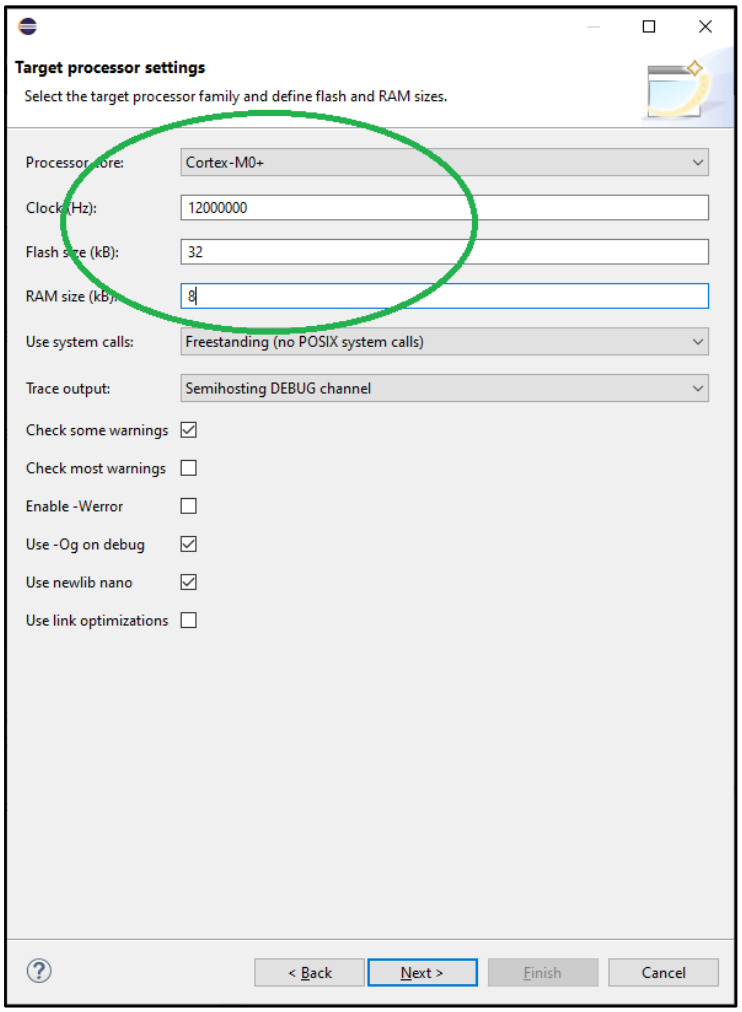


Figure 10 Platform configuration

6. Use the default folder options for this project select *Next>* and use the default configuration then select *Next>*:

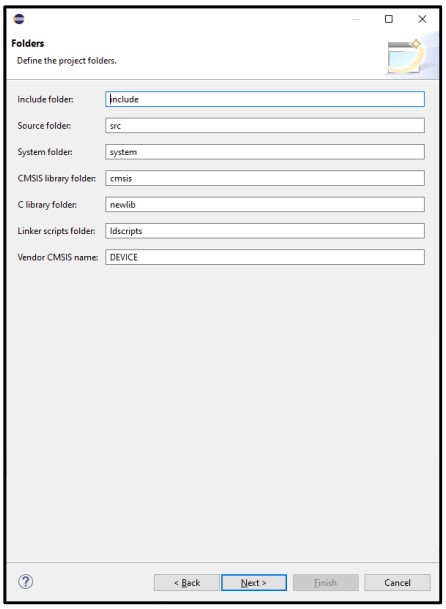


Figure 11 Default project folder

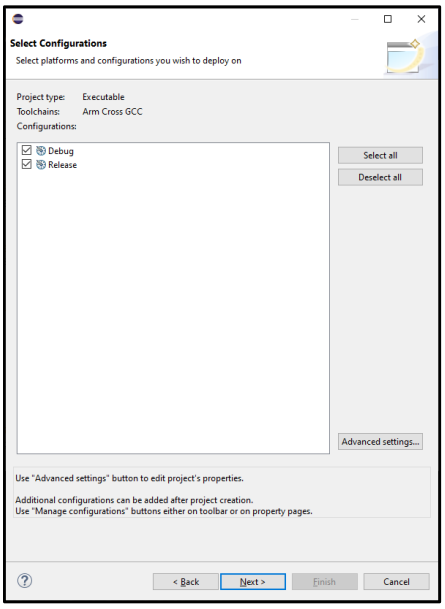


Figure 12 Build configurations

7. Ensure the correct Toolchain is correct and path to GCC:

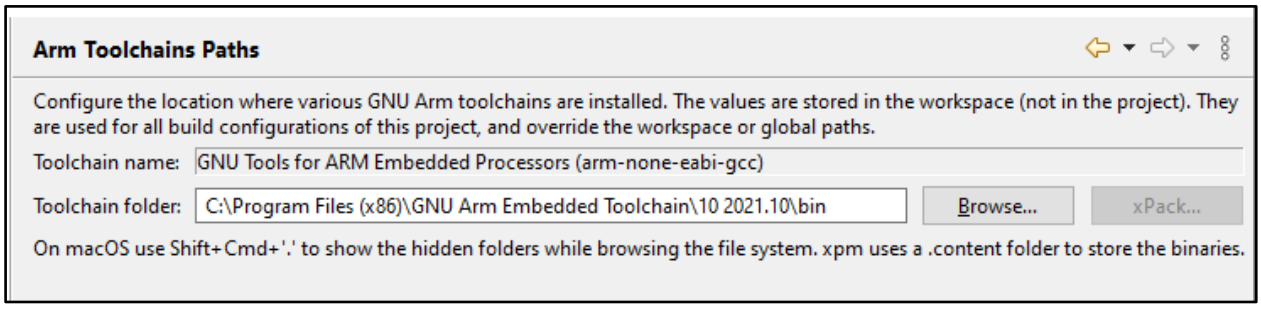


Figure 13 Tool chain paths

8. Additional note; As this is a managed MAKE project, *make.exe* must be present in the eclipse build tool path. It is up to the developer where and how to set the PATH variable; this can be done several ways, for ECLIPES managed projects we suggest *Properties →C/C++ Build →Settings [Toolchains TAB]* (part of the project settings). The down fall to this method when using REPOs, if the development team does not use similar *make.exe* path the project will fail to build… Alternatively, a "user" batch file for Windows can be created to set the environment and launch Eclipse. If make is not present, it can be downloaded from the web.
   Once available, set the Build tools folder for this exercise; *make.exe* is located in *C:\Program Files (x86)\GnuWin32\bin*.

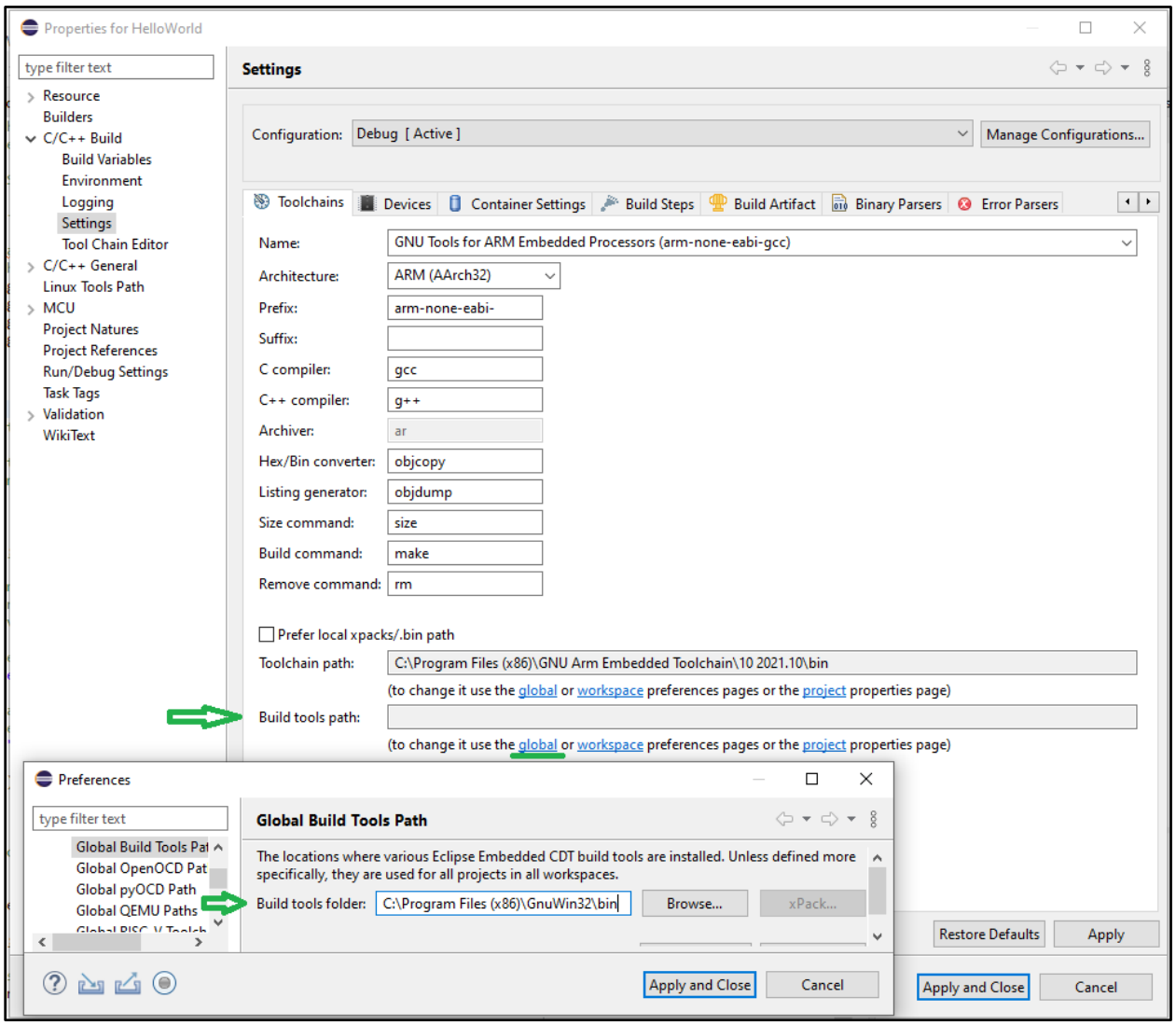


Figure 14 Setting make.exe path!

2.1.1.1 Building and debugger Project

The Eclipse Hello World ARM Cortex-M project provide basic CMSIS support, in our exercise the Cortex-M0+ Core as described in ARM® Cortex™-M0+ Devices, Generic User Guide.

1. The linker script memory map must be updated to point to the base address of on-board RAM, set:

RAM (xrw) : ORIGIN = 0x10000000, LENGTH = 8K

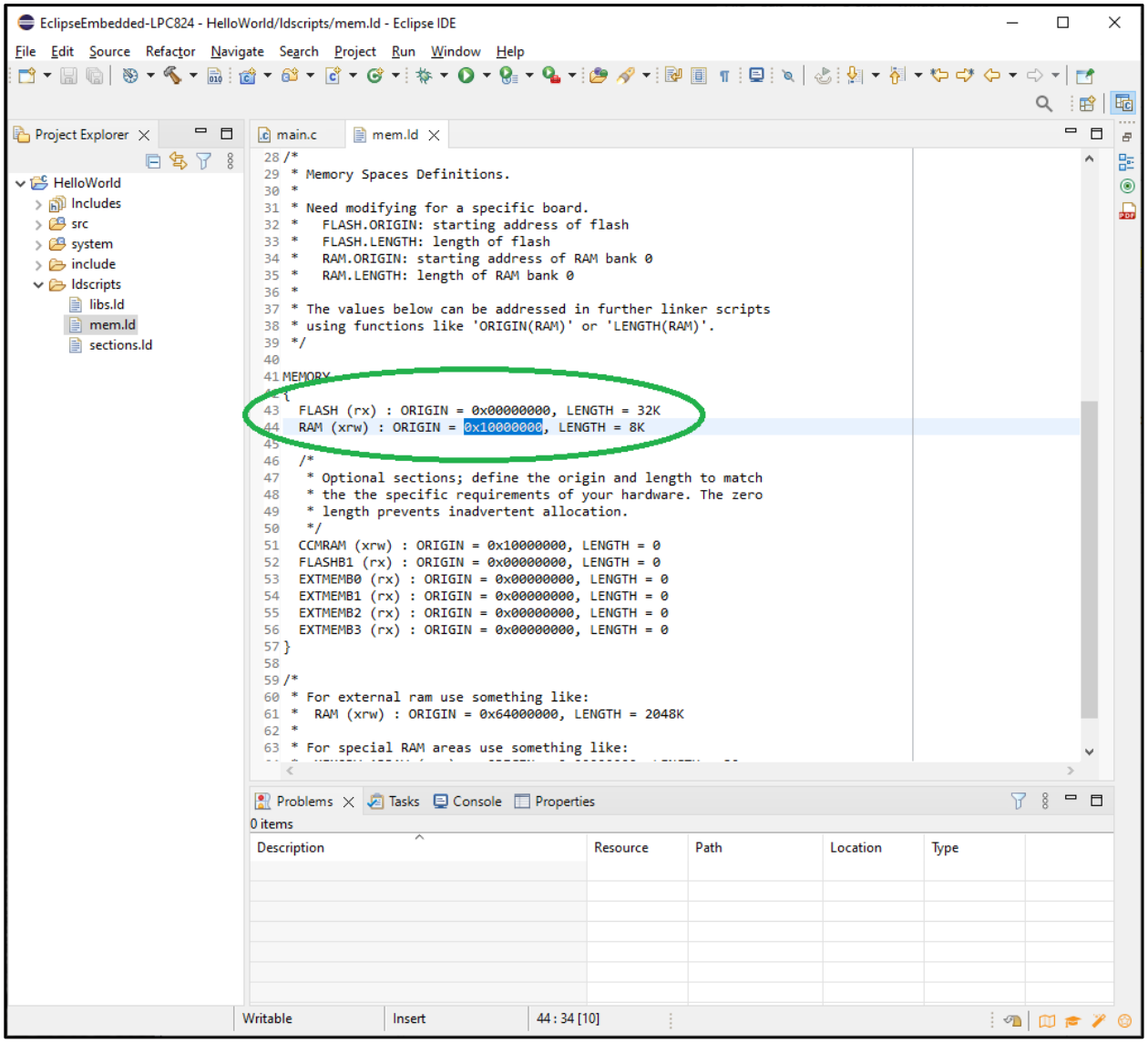


Figure 15 Platform memory map

2. Finally build the project, in the **Project Explorer** select *HelloWorld->Build Project*:

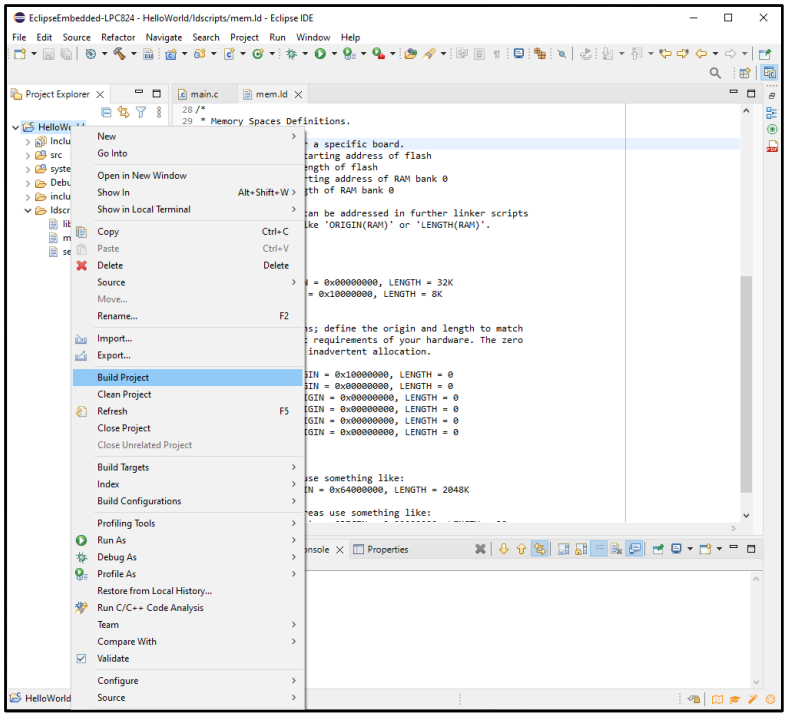


Figure 16 Build Project

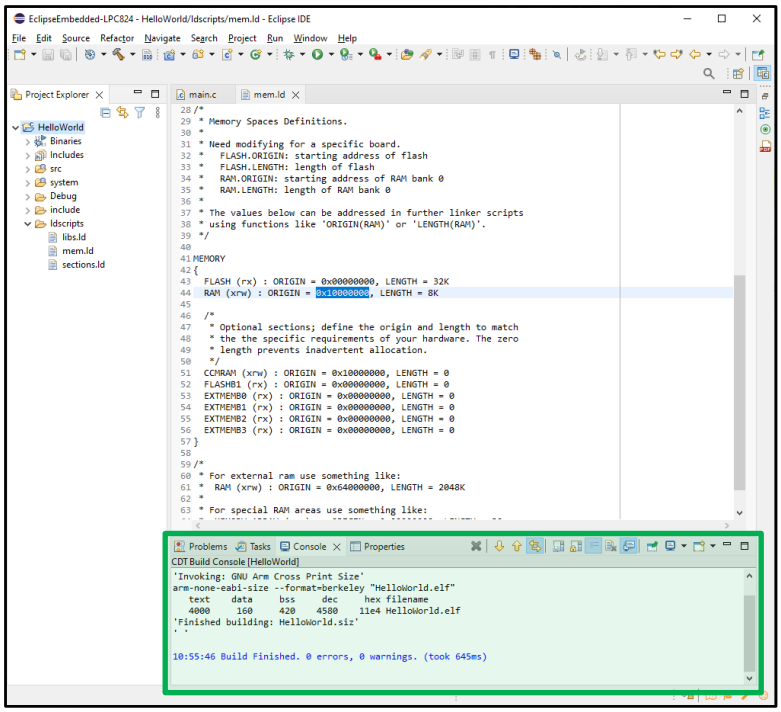


Figure 17 Successful build

3. The hardware debug interface must be configured before used. From the button bar select the debug ICON,  drop list and from the list box select *Debug Configurations…*

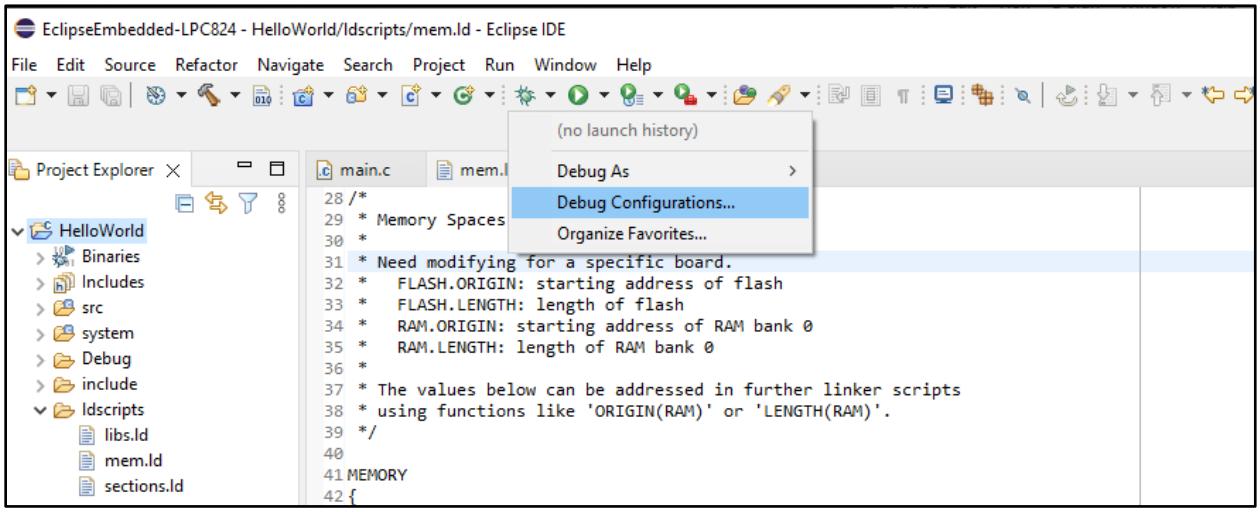


Figure 18 GDB Hardware Configuration

4. From the items list select *GDB SEGGER J-Link Debugging* and double click it to bring up the HelloWorld debug configuration dialog:

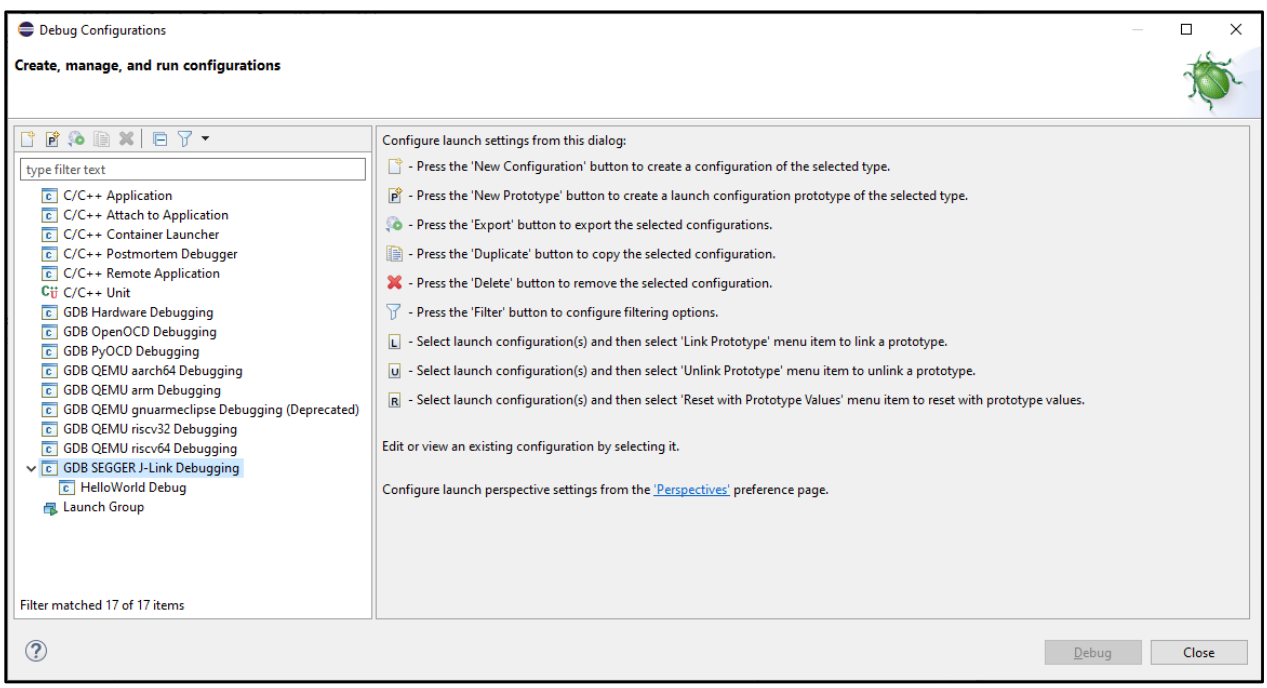


Figure 19 Select SEGGER J-LINK GDB configuration

5. From the List Box select *HelloWorld Debug* then select the *Debugger* tab. In the *Device name* field enter **LPC824M201** and in the *USB serial or IP name/address* field enter your J-Link product serial number. Select the *Debug* button and agree to the debug perspective.
   The developer is encouraged to investigate other tabs for available configuration.

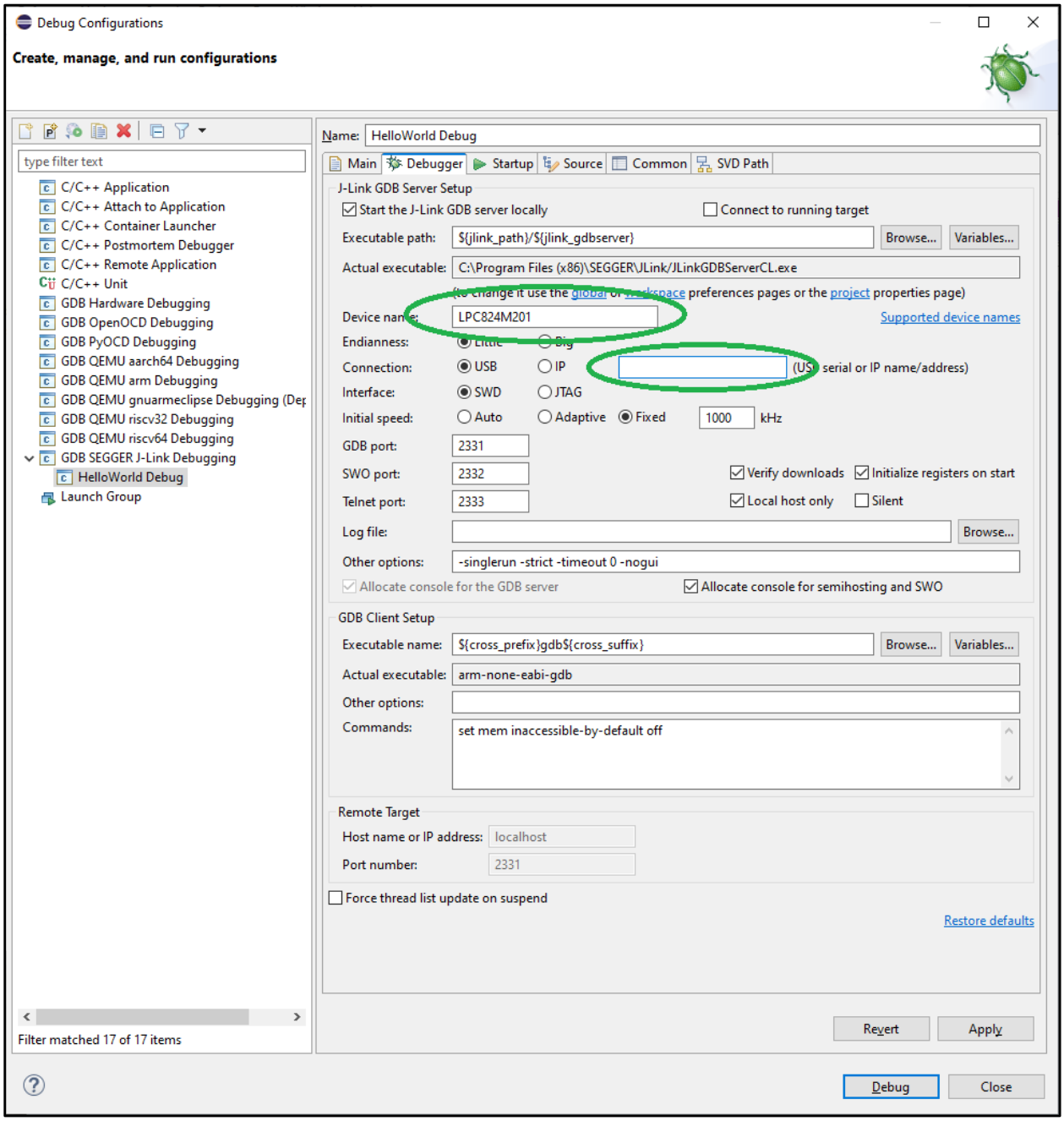


Figure 20 SEGGER J-Link hardware configuration for GDB.

6. Finally run the project from the button bar select the resume button or F8 to run the project. Note the output in the console window:

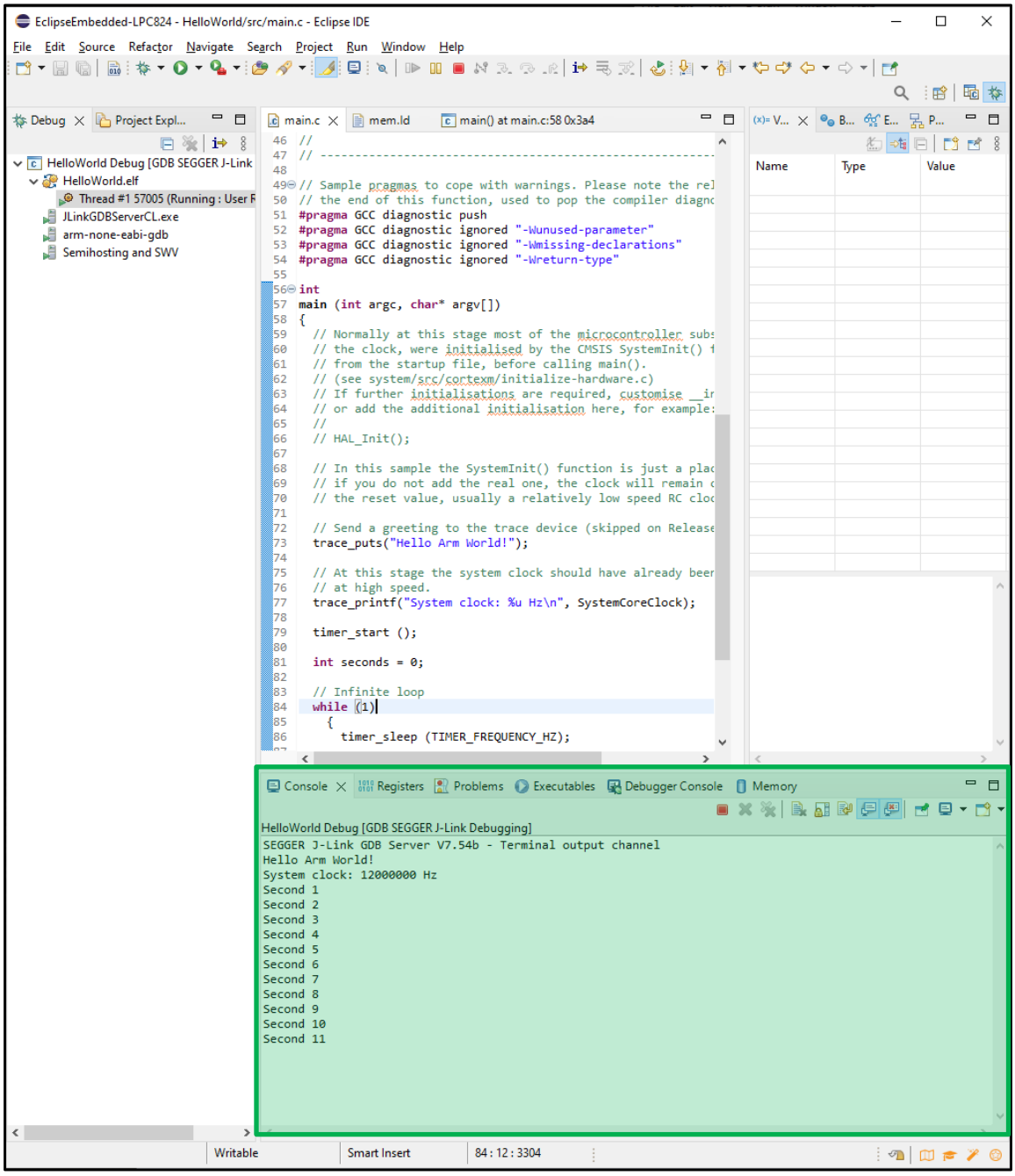


Figure 21 Debug Console output

3.0 Installing SEGGER J-Link utilities

The SEGGER J-Link hardware and software utilities allow developers the means to download firmware to the target platform as well as debug applications in the target platform. As of version 0.4 of this document we will describe how to install and set up SEGGERS's J-Link ARM versions 6.00e. The process outlined below is the same regardless of SEGGER J-Link utilities version.

Go to http://www.segger.com/jlink-software.html and from the J-Link software & documentation pack for Windows select the Download button. Download the archive and extract it in the directory of your choice then launch the setup allocation. Follow instruction to completion.

Once installation has completed connect the J-Link device to the PC, USB enumeration should take place. Once Windows has enumerated the device, connect the J-Link to the target platform (in this case the NXP LPC824 eval board) and launch the J-Link GDB server (type J-Link GDB Server in the windows search box). Alternatively select the windows start Icon the scroll to SEGGER – J-Link Vx.xxx and choose J-Link GDB Server Vx.xxx.

The GDB Server Configuration dialog will appear, see (Figure 21 SEGGER GDB server Configuration). Ensure the following are set:

- Connection to J-Link: USB
- Target interface: SWD
- Speed: Auto selection
- Target device: Note, the developer must select the device, in our example the NXP LPC284 Xpresso evaluation board is the target so we will select the LPC824 MCU. Click the navigation button and the device explorer dialog will appear see (Figure 22 Target Device Settings: NXP LPC284).

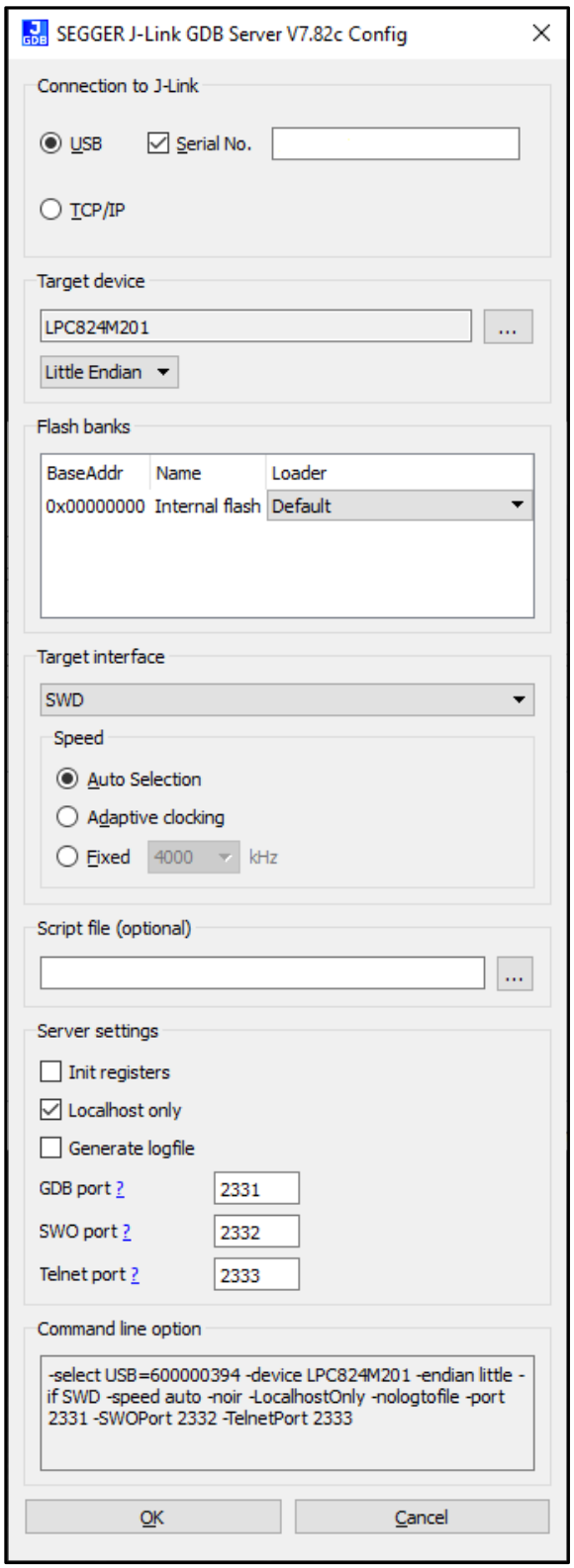


Figure 22 SEGGER GDB server Configuration

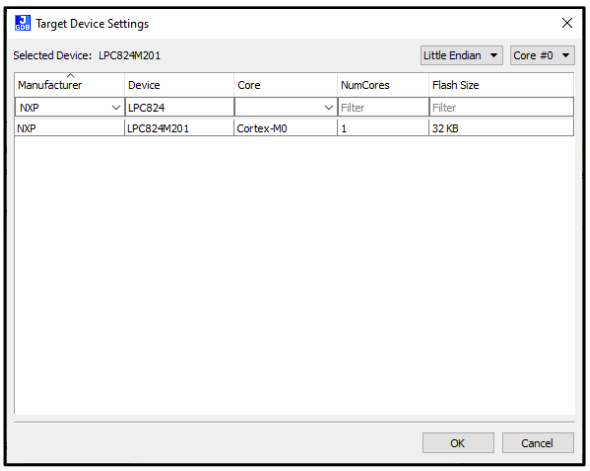


Figure 23 Target Device Settings: NXP LPC284

Once the correct device has been selected select *OK* on the Target device setting dialog then select the *OK* button on the GDB server configuration dialog. If the J-Link device successfully connects to the target the SEGGER J-Link GDB Server dialog will appear see (Figure 23 SEGGER J-Link GDB Server: Connection StatusDialog). Congratulation you are ready to debug…

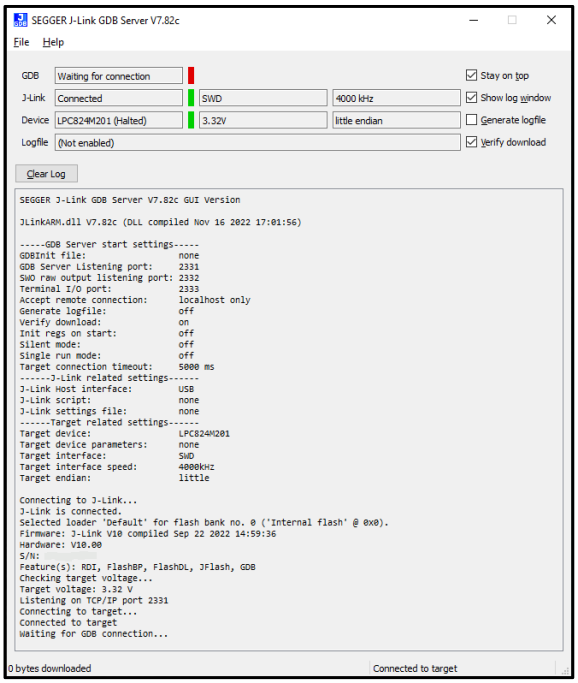


Figure 24 SEGGER J-Link GDB Server: Connection Status