NuMicro Cortex-M IAR EWARM ICE Driver User Manual

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

Table of Contents

List of Figures

1 Overview

1.1 Introduction

NuMicro Cortex-M IAR Driver allows IAR Embedded Workbench (EWARM) to communicate with Nuvoton on-chip debug logic.
In-system flash memory programming integrated into the driver allows the user to rapidly update target code. The IAR Embedded Workbench can be used to start and stop program execution, set breakpoints, check variables, inspect and modify memory contents, and single-step through programs to run your actual target hardware.
This document describes how to install and use NuMicro Cortex-M IAR Driver with programs written using IAR's compiling and flash tools.

1.2 Features

Nu-Link driver supports the following features. Some functions are triggered by IAR EWARM. The usage of these functions can be found in the IAR EWARM User Guide.

- Erase/program/verify Nuvoton chips. (via flash algorithm of Nu-Link IAR driver)
- Easy registers access of Nuvoton chips. (via the .ddf file of Nu-Link IAR driver)
- Support Hardware/Software/Flash breakpoints.
- Support Data breakpoints.
- Support Live Watch.
- Support various configurations for connection. (Reset options, SWD clock, etc.)

1.3 Supported Devices

Click to open the table of supported devices: Link of Supported devices.

2 Installing Nu-Link IAR Driver

2.1 System Requirements

- **Software:** IAR Embedded Workbench (EWARM 6.1 and higher version)
- **Hardware:** Nu-Link ICE Adapter.

2.2 Installation

Click installer file Nu-Link_IAR_Driver.exe.
The delivered package contains a complete set of files. The user needs to run NuMicro Cortex-M IAR EWARM Driver with C-Spy interface. The following directories and files can be found after package is installed successfully:

- **.\Samples:** The sample project that uses Nu-Link driver for IAR.
- **.\Nu-Link-IAR.dll:** The driver DLL.

3 Example – Create and Debug a Project

3.1 Start a New Project

This section describes how to start a new project based on NUC100 series chips. The fast and easy way to start a new project is open an existing IAR project. To make sure the user knows about all the steps to create an IAR project, this section will start with an empty project.

1. Open IAR Embedded Workbench, and click "**File**" → "**New**" → "**Workspace**".

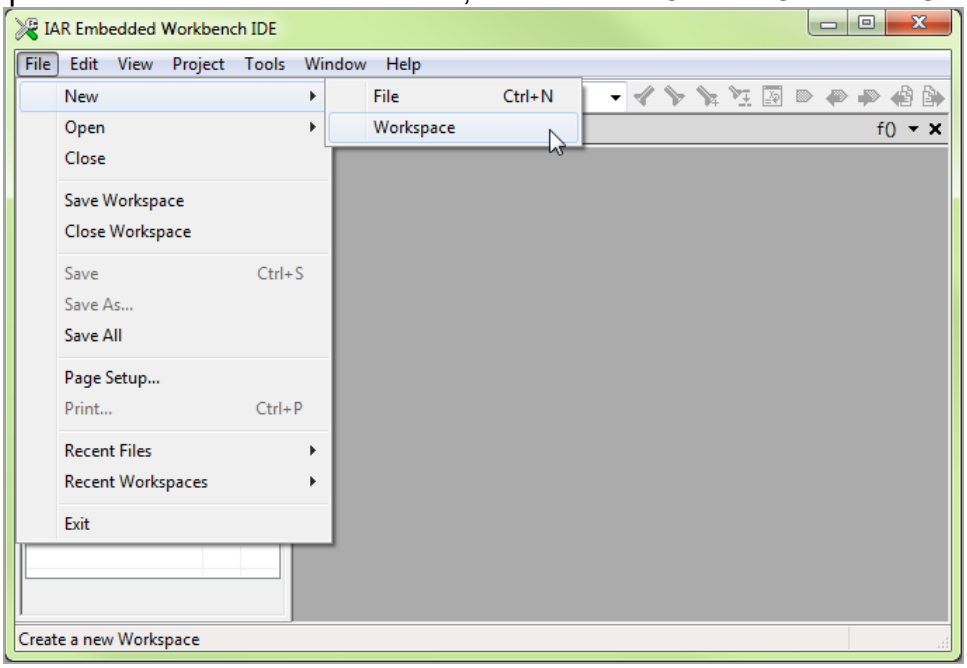

Figure 3-1 Create a New Workspace

2. Create a new project by clicking "**Project**" → "**Create New Project**".

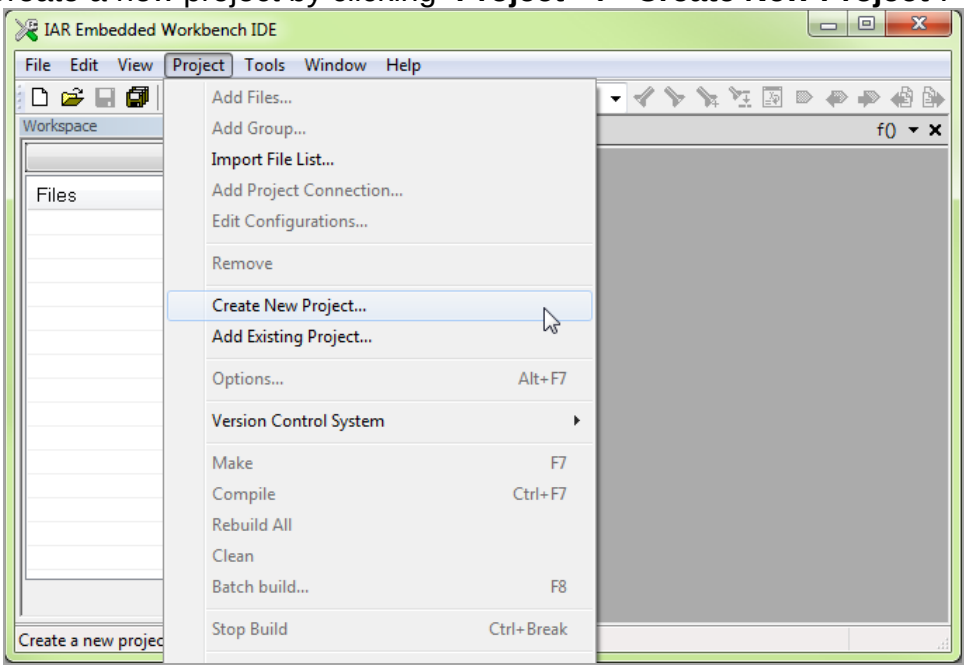

Figure 3-2 Create a New Project

3. Select "ARM" as the tool chain for this project, and then click "**OK**".

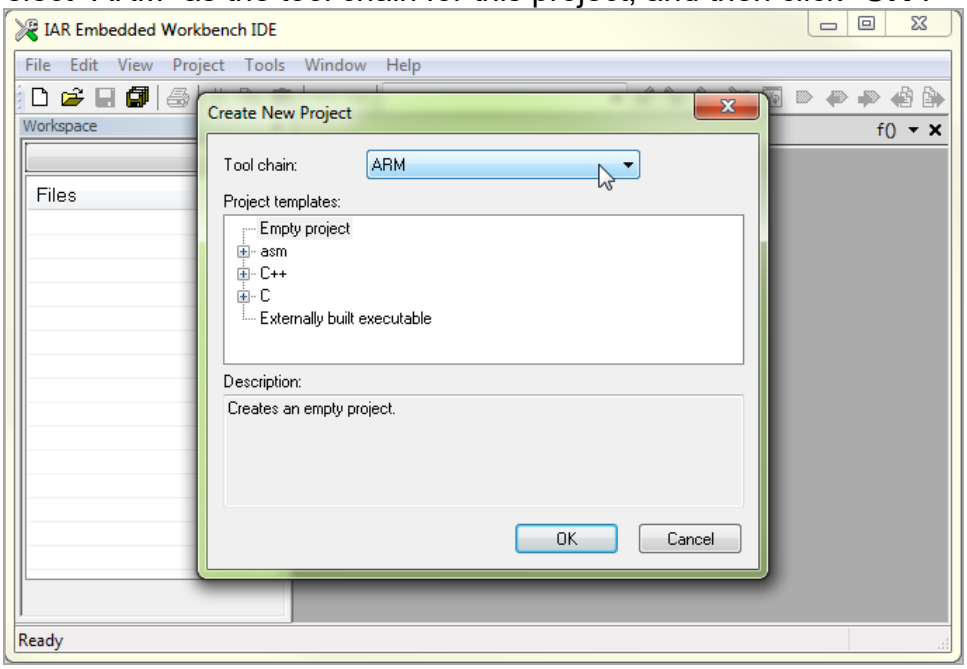

Figure 3-3 Select the Tool Chain for the Project

4. Now you'll be prompted to save the project. Select a folder and input a project name to save it.

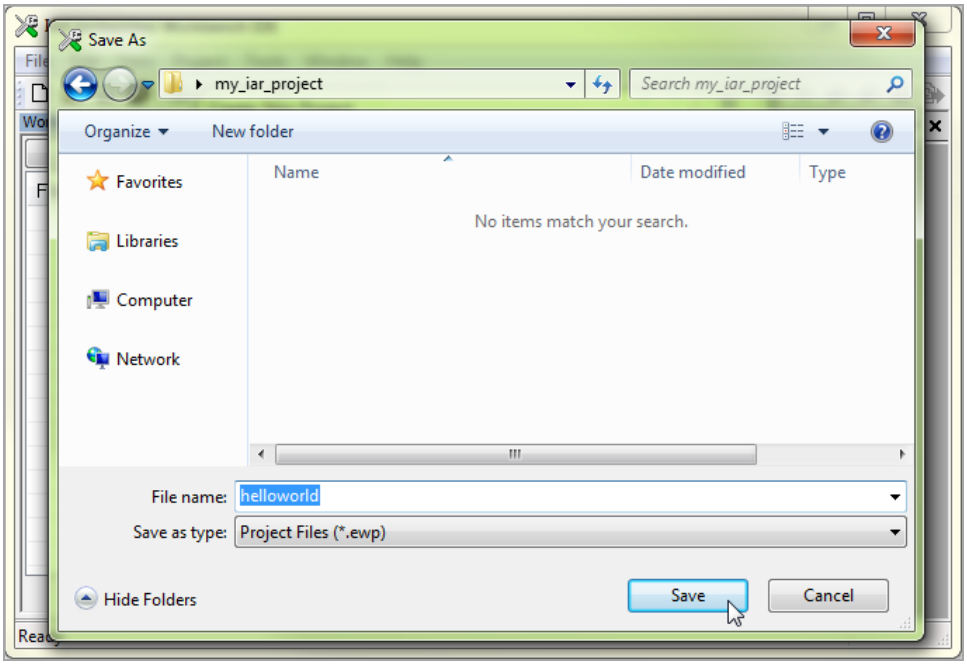

Figure 3-4 Save a Project

5. After the project is saved, right click on the project name of workspace area, and click "**Options**" to open the option setting form.

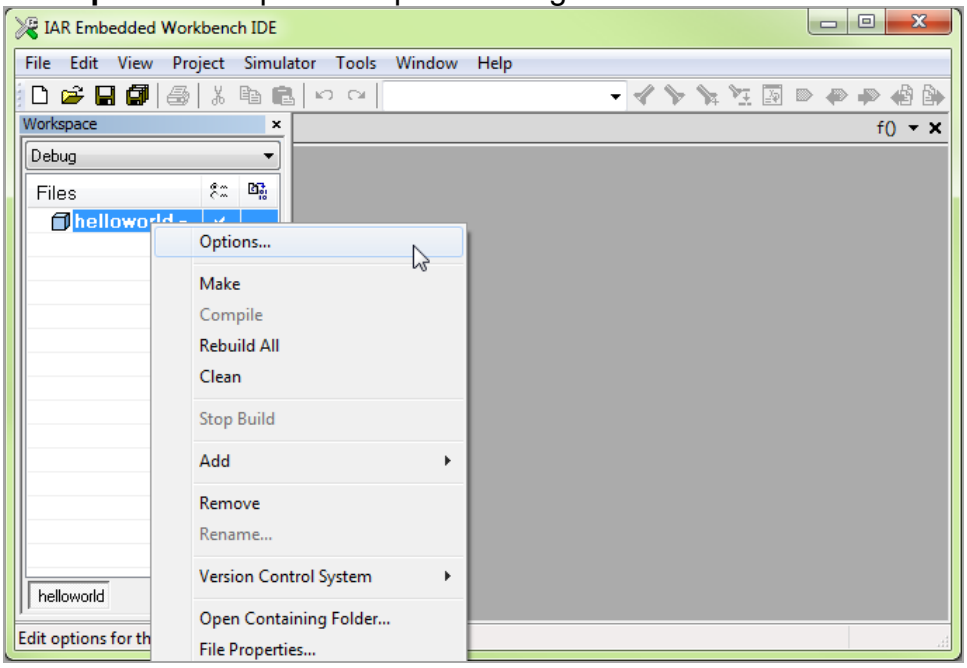

Figure 3-5 Open Project Options Form

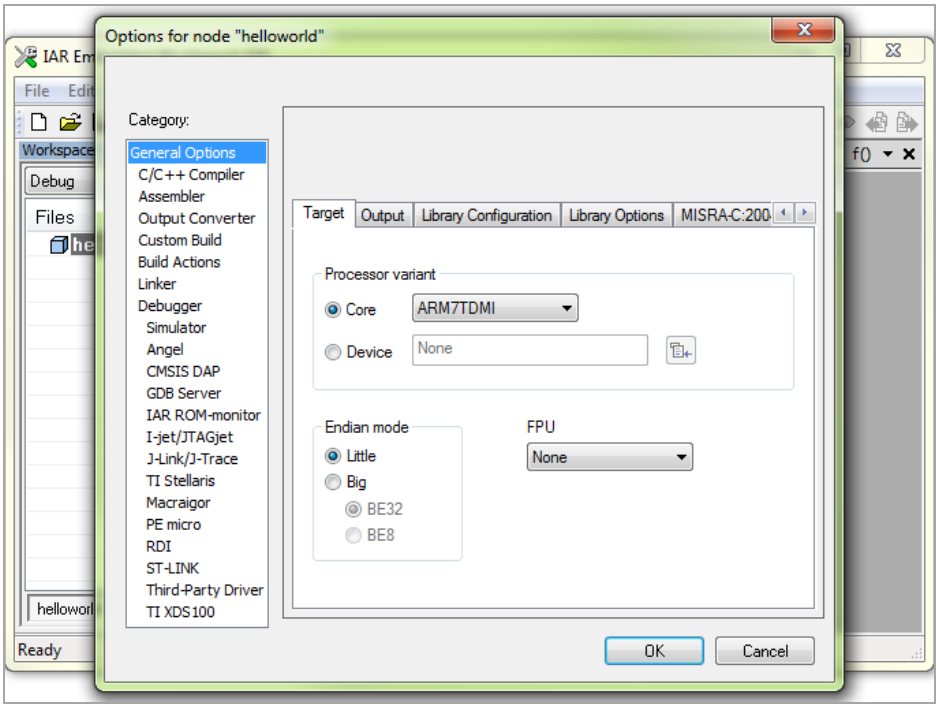

Figure 3-6 Options Form

6. The default options are neither for NUC100 series nor for Cortex-M0. Check the "**Device**" option and click the "Device" icon on the right to select the correct device name, such as "Nuvoton NUC100AN series (NUC100AN,NUC120AN)".

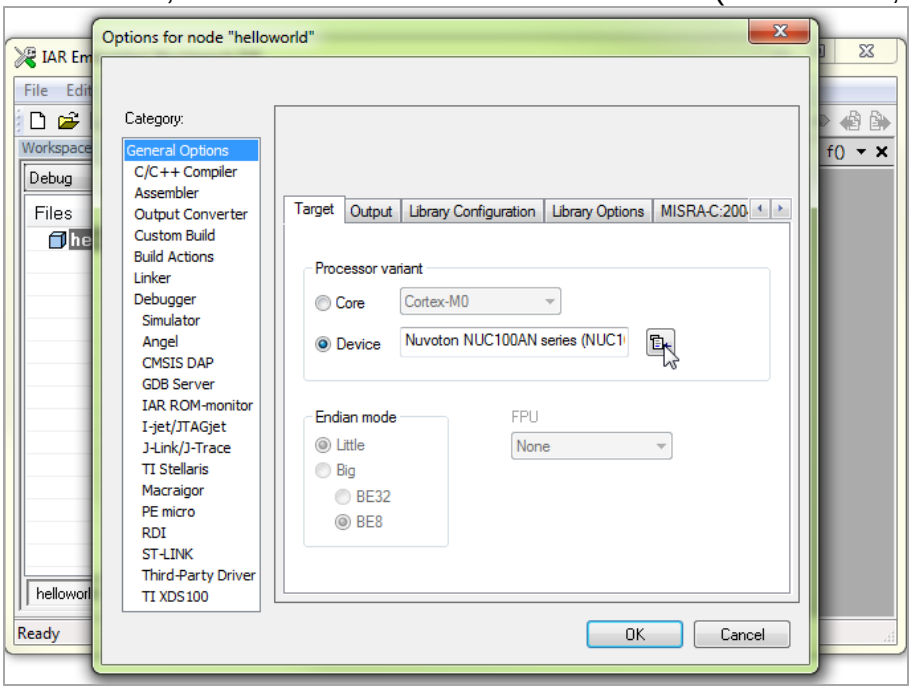

Figure 3-7 Select Target Devices

7. To link and run the program in flash memory, override the default link script for this project. In this example, simply specify the link script used in the sample project "*Samples\PWM*".

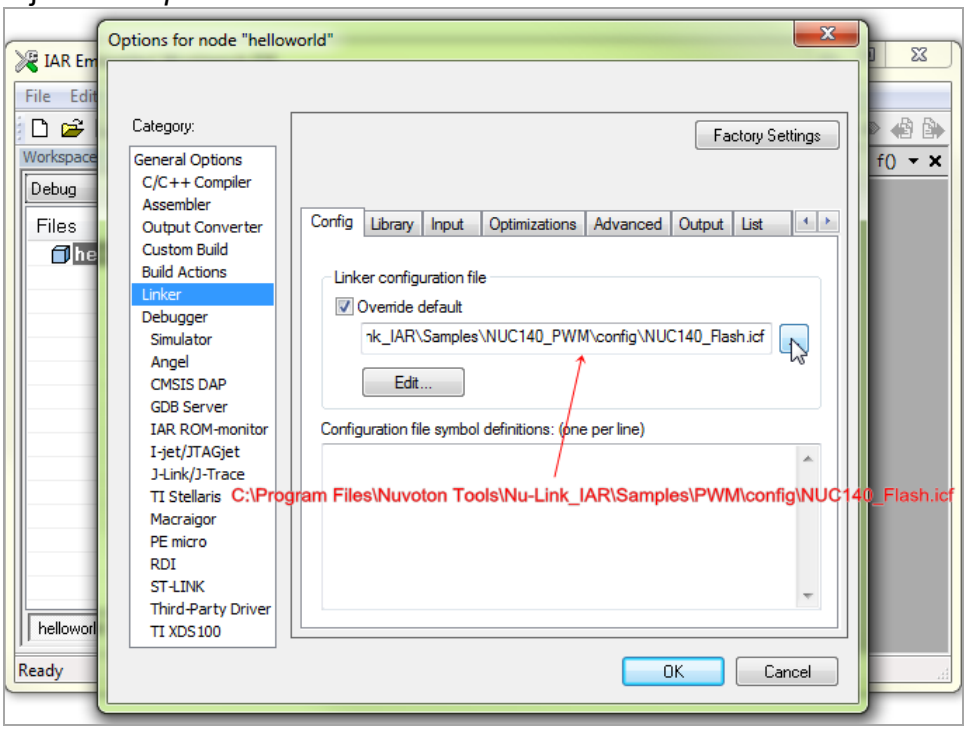

Figure 3-8 Select Linker Configuration File

(Note: 8051 series uses .xc file)

8. Select “Nu-Link” as the debugger driver for this project. On the “Debugger” page, select “**Third-Party Driver**”; and on the “Third-Party Driver” page, fill in the path of Nu_Link-IAR.dll.

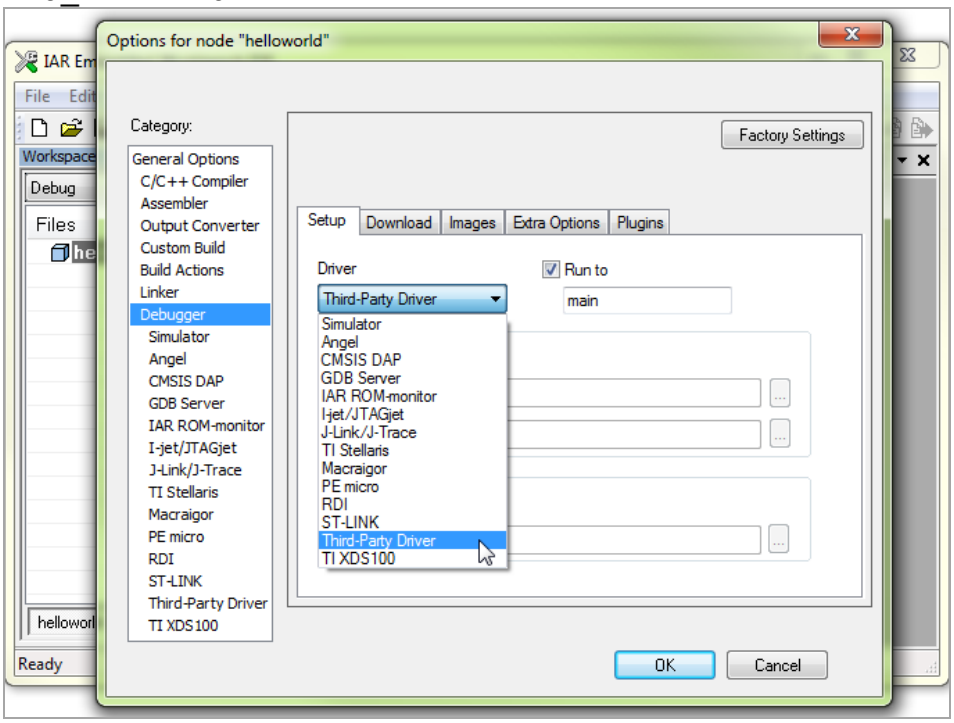

Figure 3-9 Select Debugger Driver

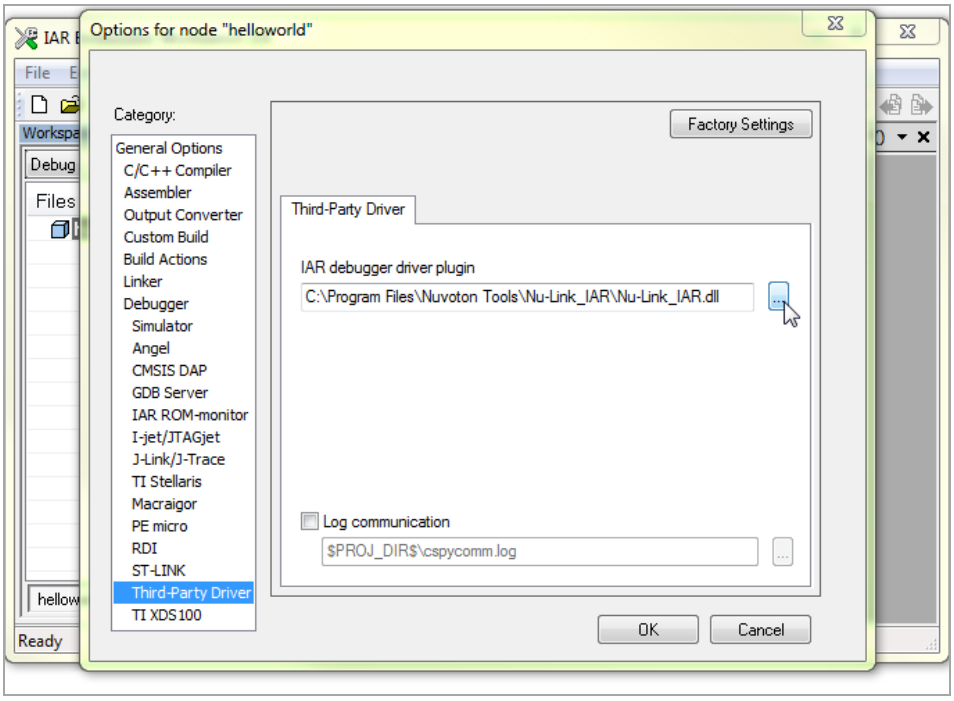

Figure 3-10 Use Nu-Link for Debugger Driver

9. Go back to the “Debugger” page, click the “Download” tab and enable the “**Use flash loader(s)**”. If the previous settings are correct, the default load file should be “*$TOOLKIT_DIR$\config\flashloader\Nuvoton\NUC100_APROM.board*”.

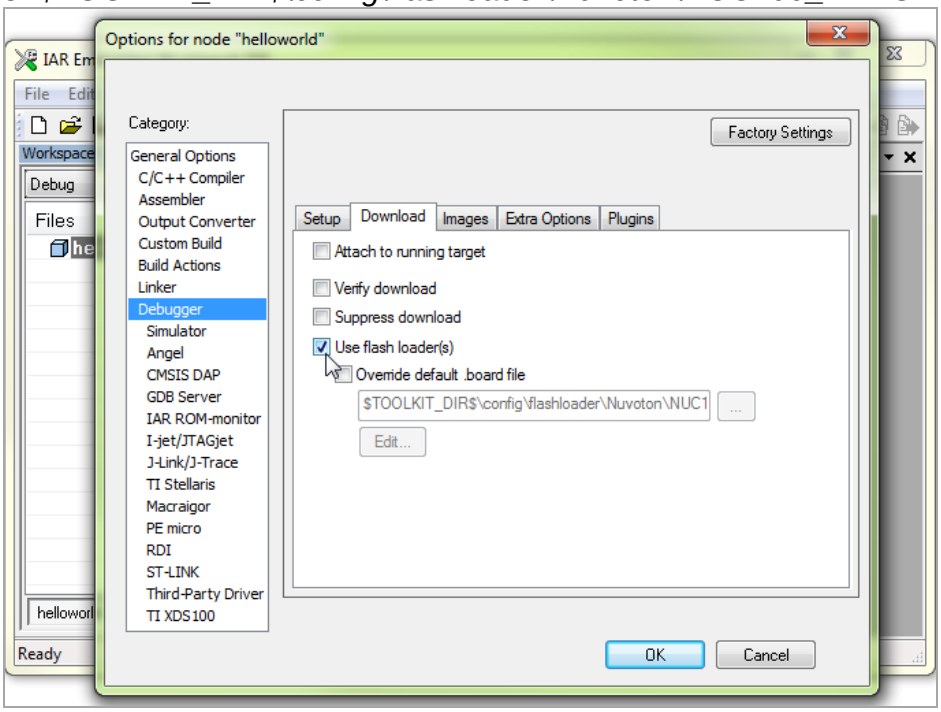

Figure 3-11 Select “Use flash loader” Options

(Note: 8051 series do not need to set flash loader)

10. After the option settings are done, click the “**OK**” button and save the project.

Since the project is initially empty and has no source code, an ASM startup code is required for a Cortex-M project. Simply copy the file “*cstartup_M.s*“ from the installation folder of Nu-Link driver to the project’s directory.

To build the project, you should add the startup code and user application code to the project. Please follow the steps below:

1. Add the *cstartup_M.s* to the project.
   The default path of the startup ASM is:
   "C:\Program Files (x86)\Nuvoton Tools\Nu-Link_IAR\Samples\NUC140_PWM\cstartup_M.s"
   (Nuvoton BSP for IAR project uses file *startup_seriesName.s* as the startup ASM file, which has the same functions as *cstartup_M.s*).

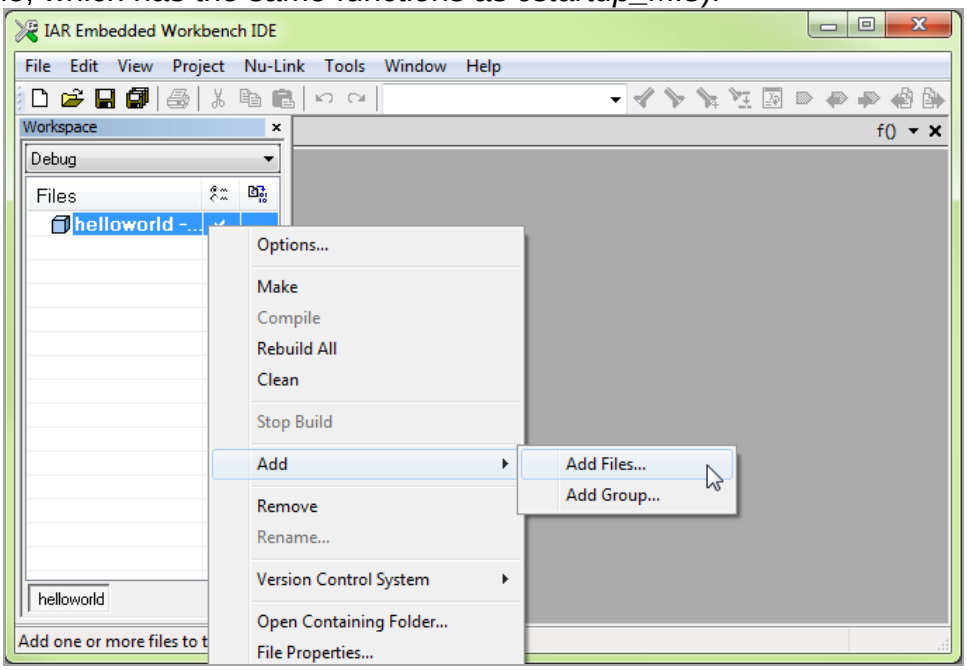

Figure 3-12 Add Files to the Project

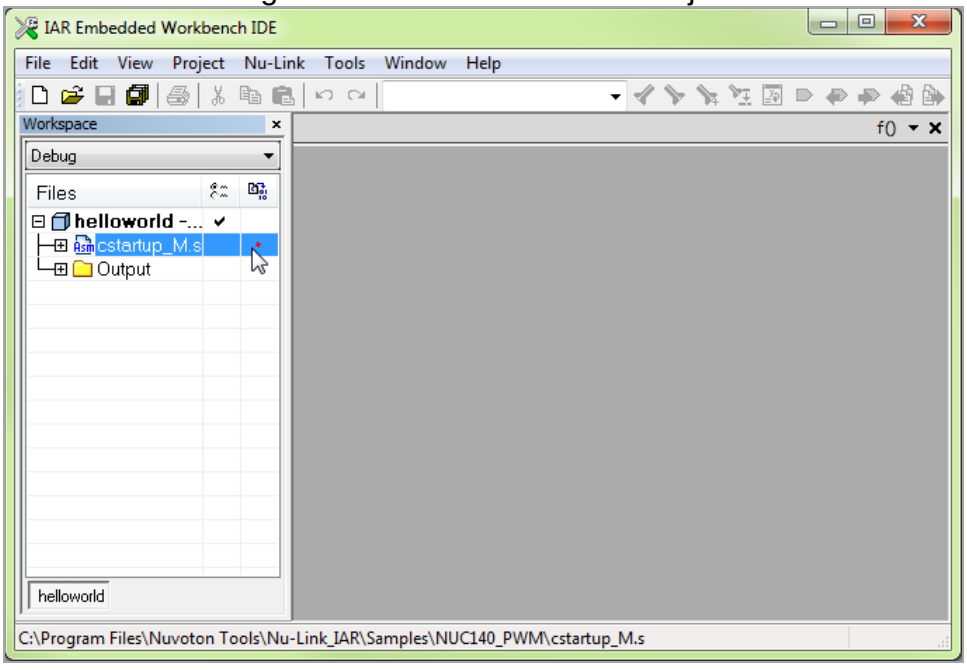

Figure 3-13 Add “Cstartup_M.s” File to the Project

2. Now add the main function to this project. Click “**File**” → ”**New**” → “**File**”, and input the main function in the new file’s editing pane.

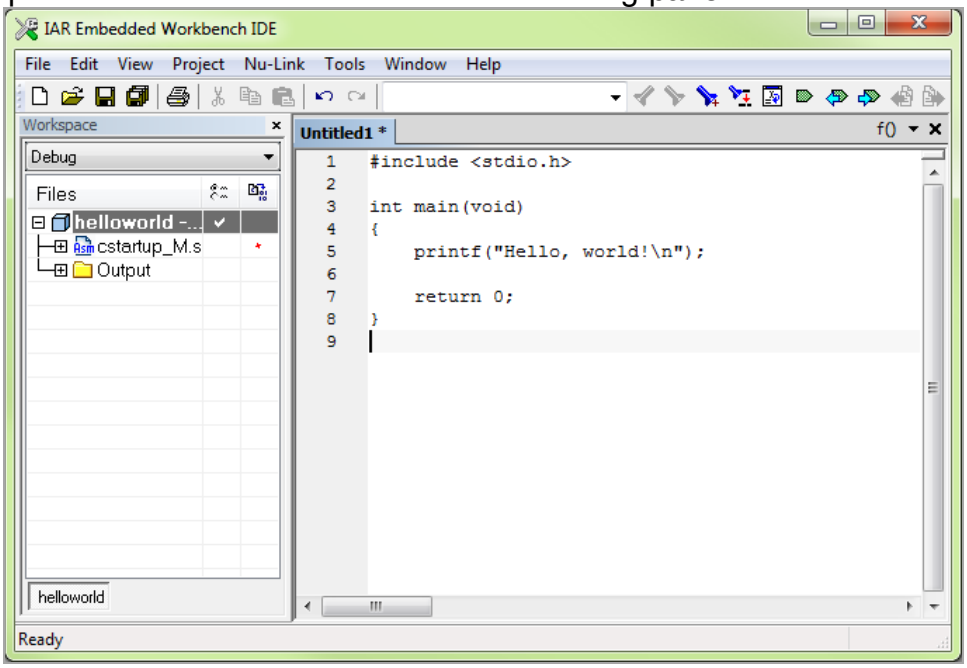

Figure 3-14 Add the Main Function in a New File

3. Click “**File**” → “**Save**” to save the new text file as “*main.c*”.

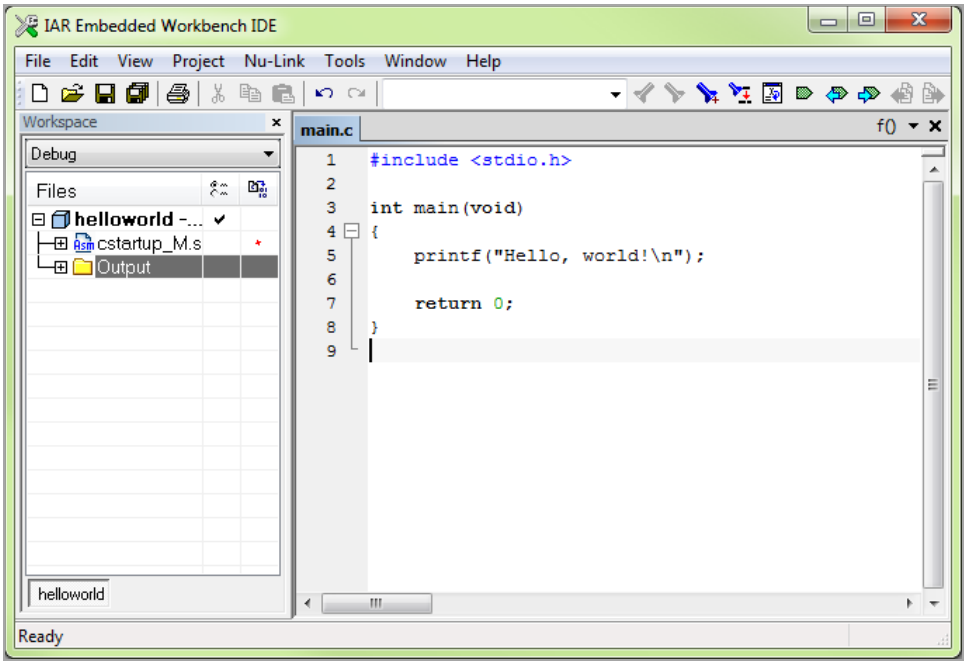

Figure 3-15 Save the New File and Rename to “main.c”

4. Add “*main.c*” to the current project. After “*main.c*” has been added, it will be listed in the “Workspace” pane.

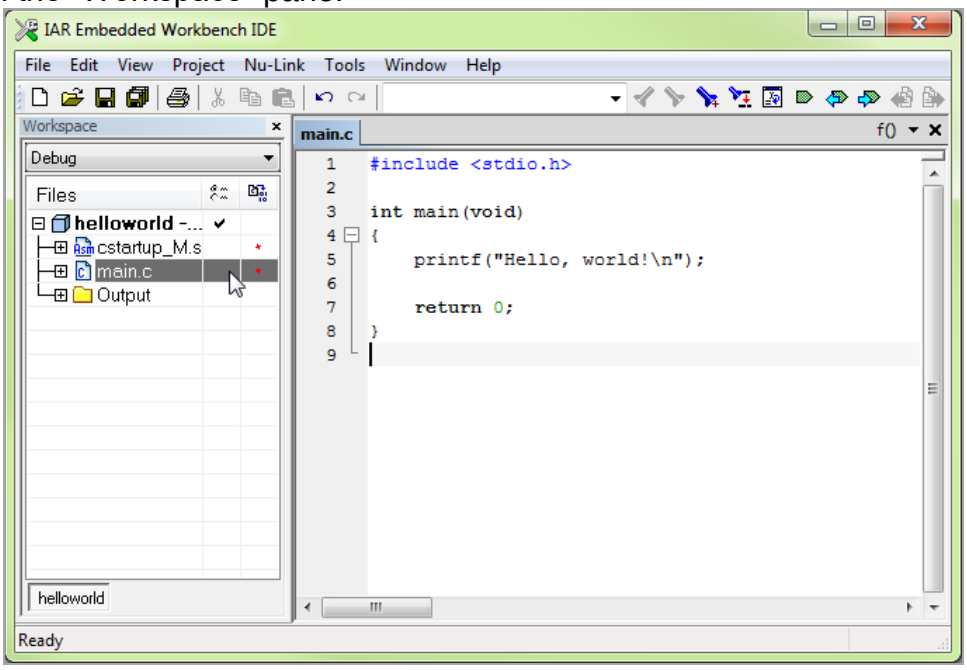

Figure 3-16 Add “main.c” File to the Project

5. Click “**File**” → “**Save Workspace**”. In the file dialog, input a name for this workspace and save it.

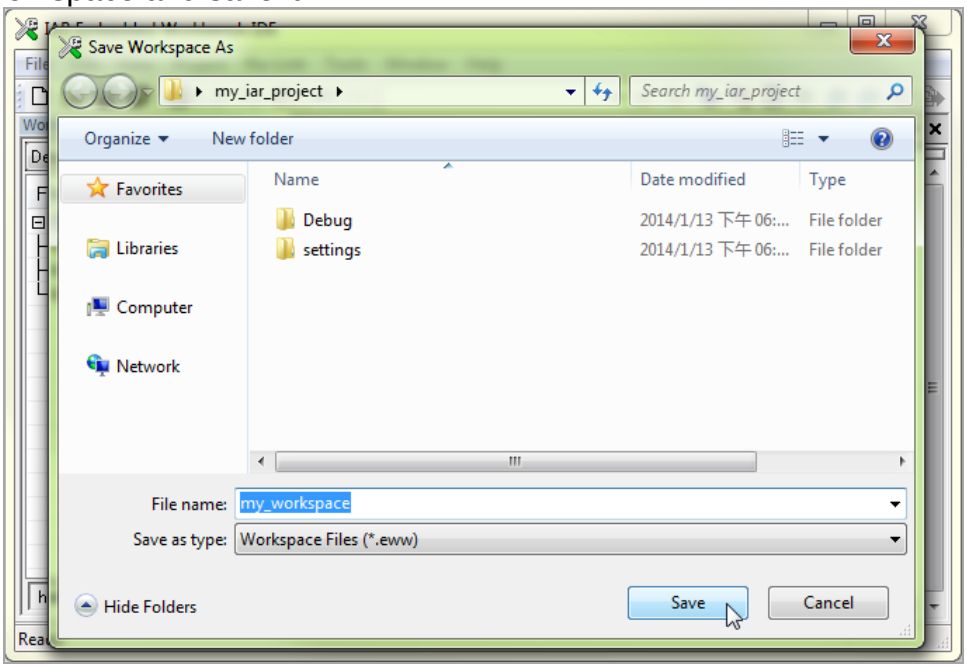

Figure 3-17 Save the Workspace

6. Now start to build the project by clicking "**Project**" → "**Rebuild All**" or "**Make**".

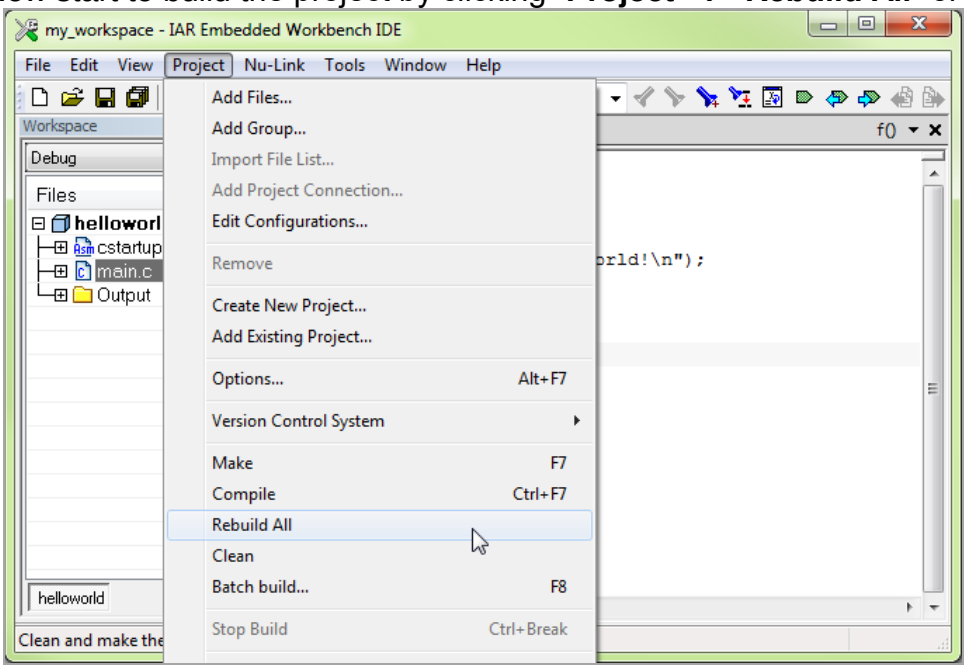

Figure 3-18 Rebuild the Project

7. The project is created successfully.

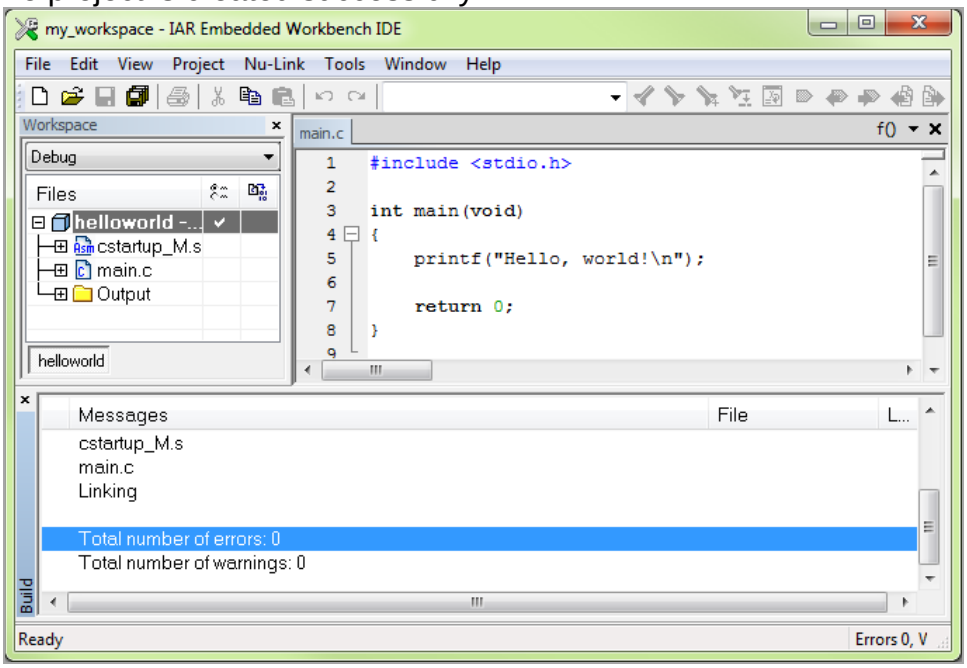

Figure 3-19 Build the Project Successfully

3.2 Debug a project

IAR EWARM is a tool for users to debug the project easily. After the project is created successful and the target device and Nu-Link debugger is correctly connected. (Note: 8051 series didn't support the features after section 3.2.1)

- Click "**Project**" → "**Download and Debug**" to start to download and debug the project. Using the project described 3.1, the program will pause at function main().

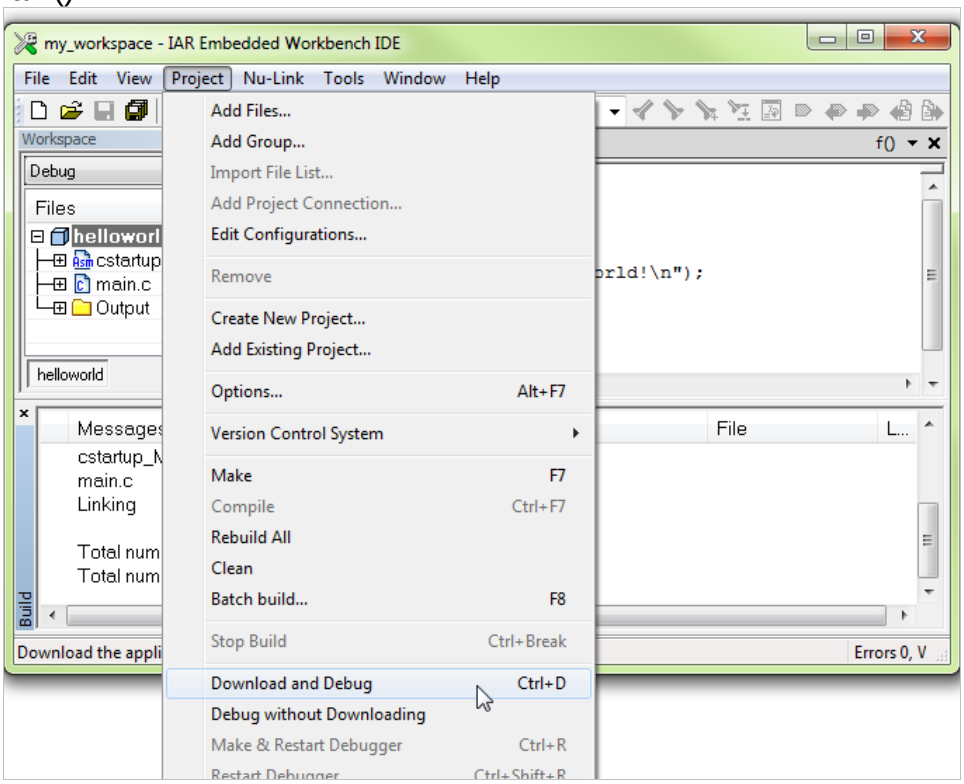

Figure 3-20 Download Code and Debug

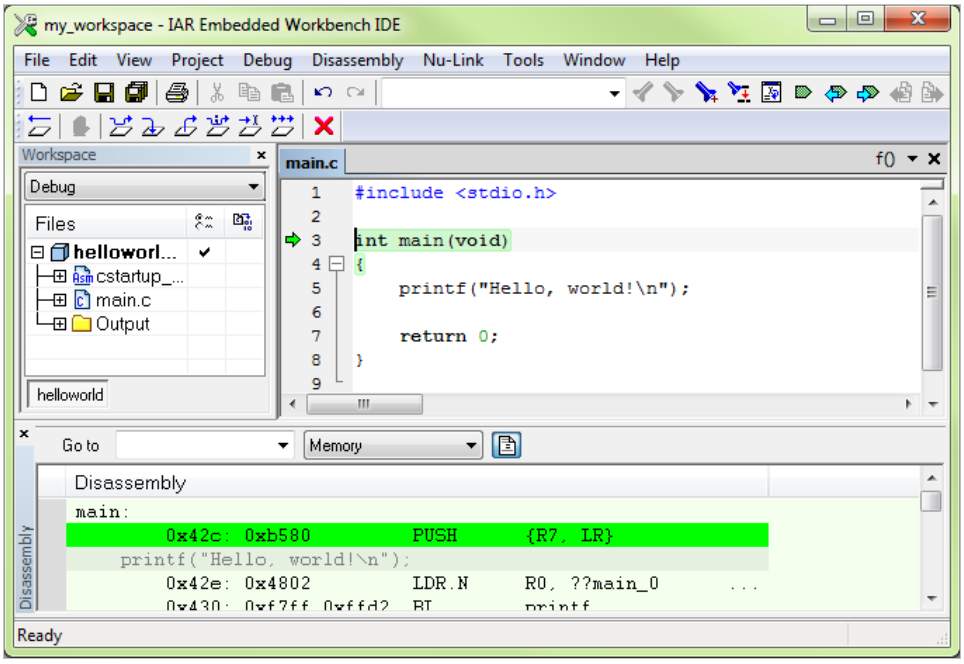

Figure 3-21 Pause at Main Function while Debugging

The following section introduces the debug features using the project described above.

- Move the cursor to the line that contains “printf”, press “**F9**” to set a breakpoint, and then press “**F5**” to run the program until one breakpoint occurs. Now the program pauses at the breakpoint line.

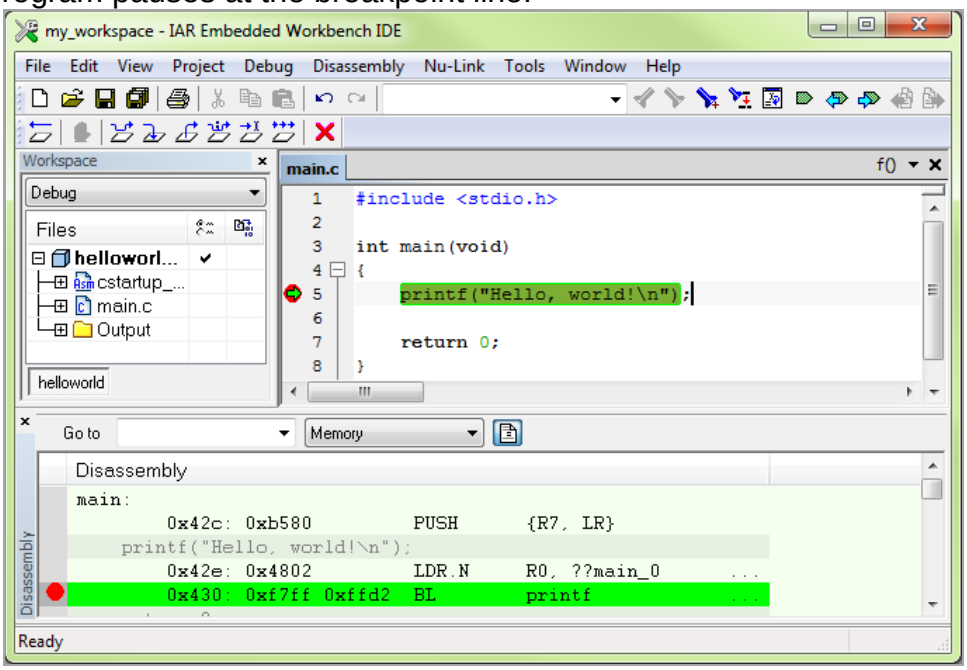

Figure 3-22 Pause the Program by Setting Breakpoints

- Open the “Terminal I/O” pane by clicking “**View**” → “**Terminal I/O**”. Press “**F10**” to step over current source line. The printf will output the message in the “Terminal I/O” pane in a semihosting way.

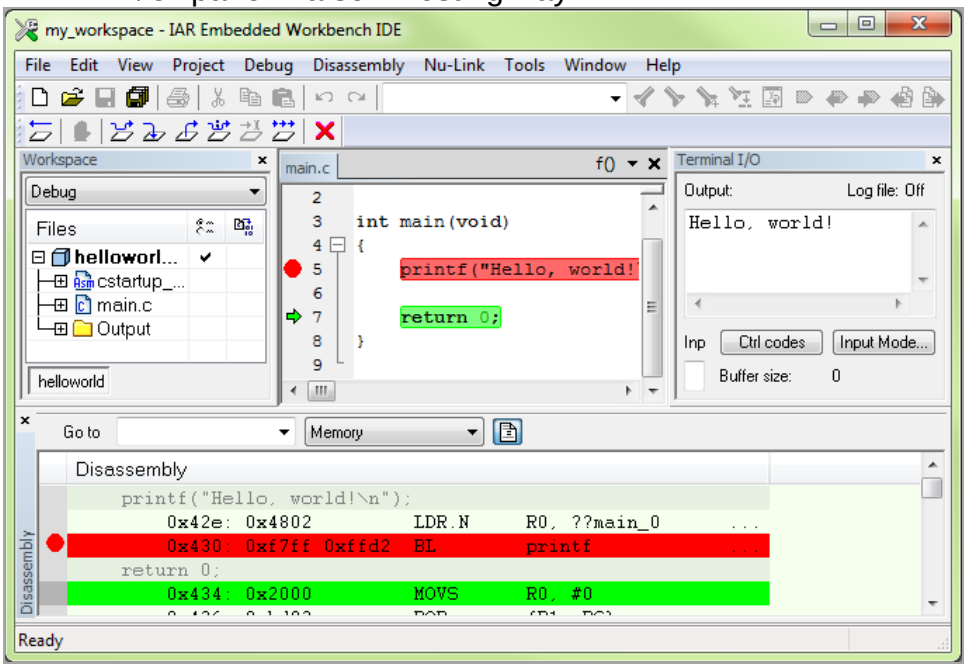

Figure 3-23 Terminal I/O Pane

- Click “**View**” → “**Memory**” to open the Memory pane, in which user can view or edit the memory on the target device.

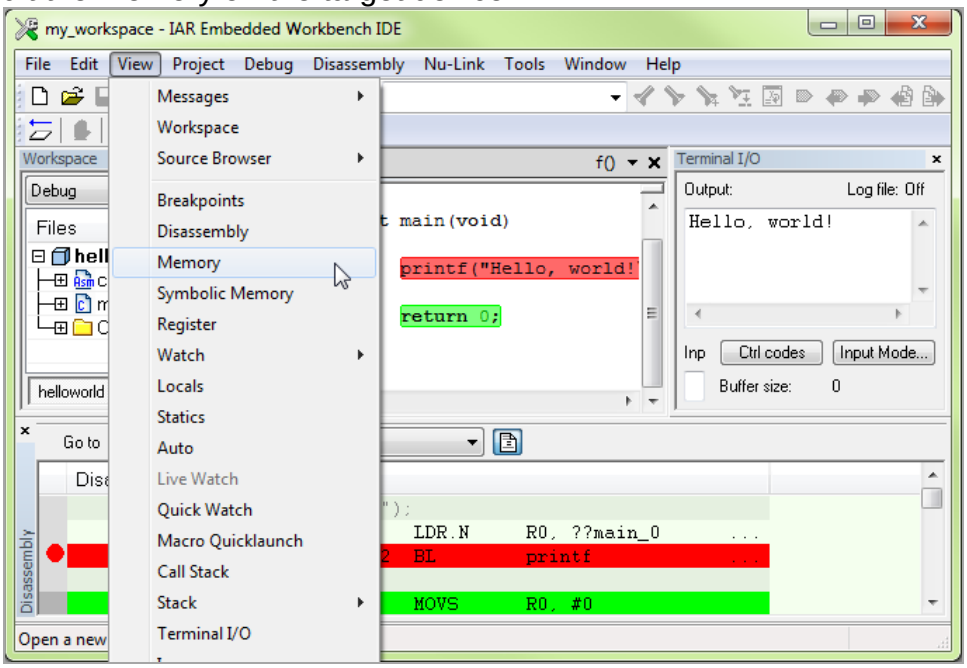

Figure 3-24 Open Memory Pane

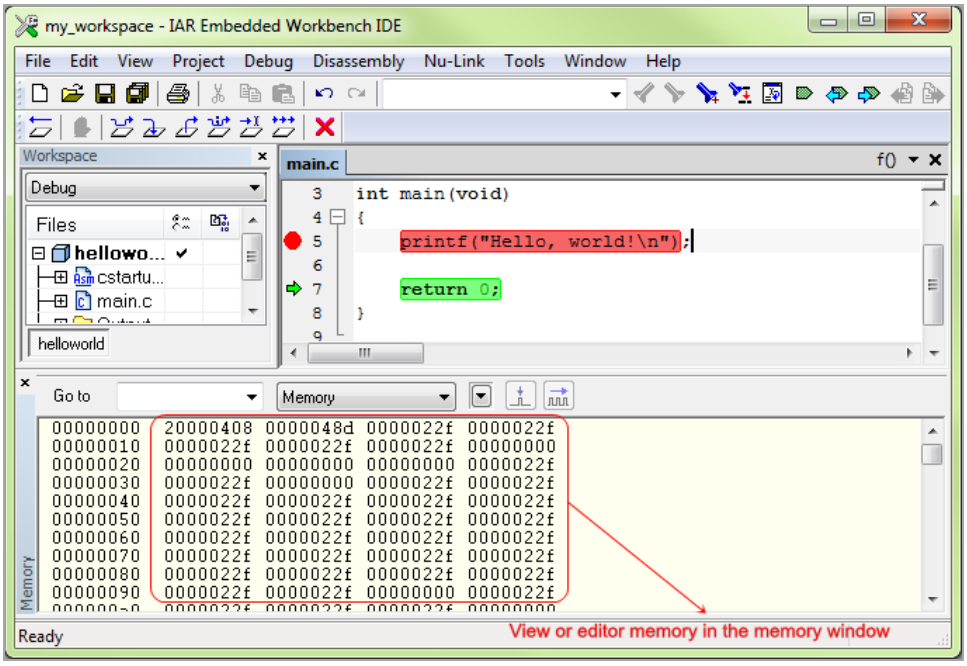

Figure 3-25 Memory Pane

- Click “**View**” → “**Register**” to open the Register pane, in which user can view or edit the registers on the target device.

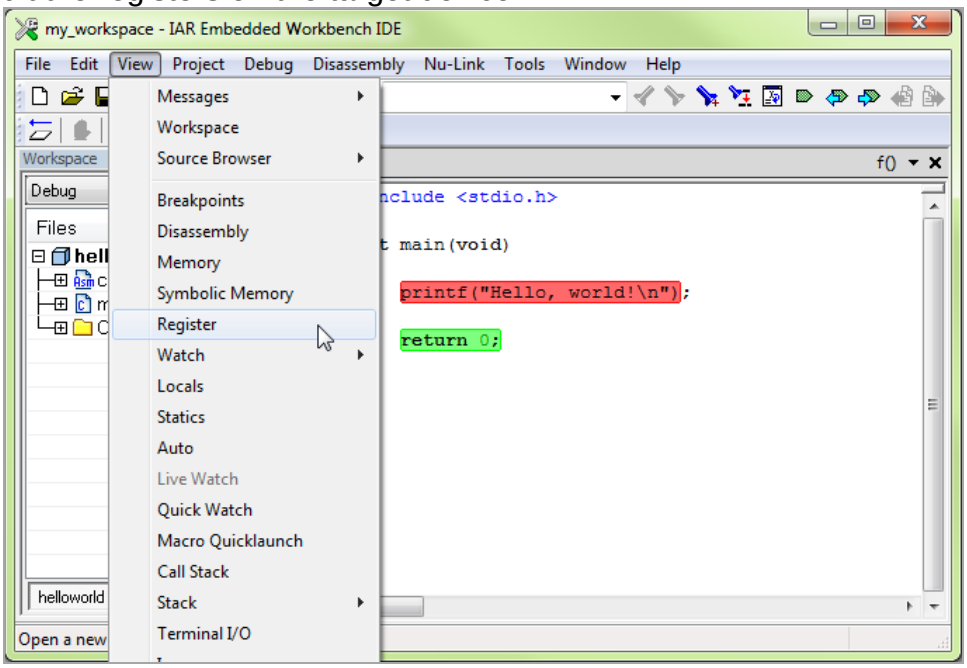

Figure 3-26 Open Register Pane

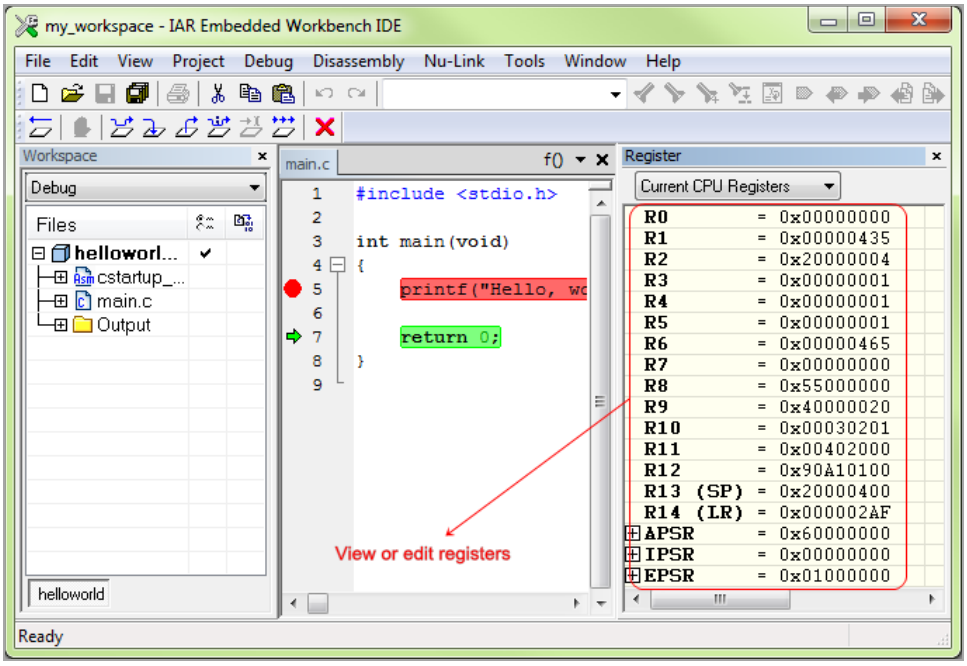

Figure 3-27 Register Pane

- In the same Register pane, the user can also access registers other than CPU registers.

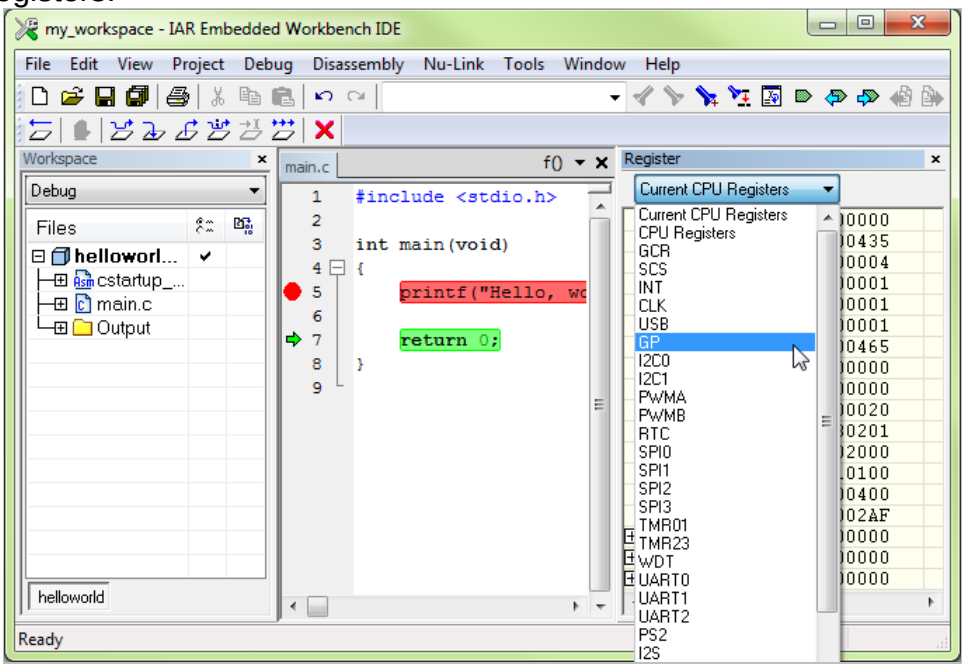

Figure 3-28 Select Other Registers

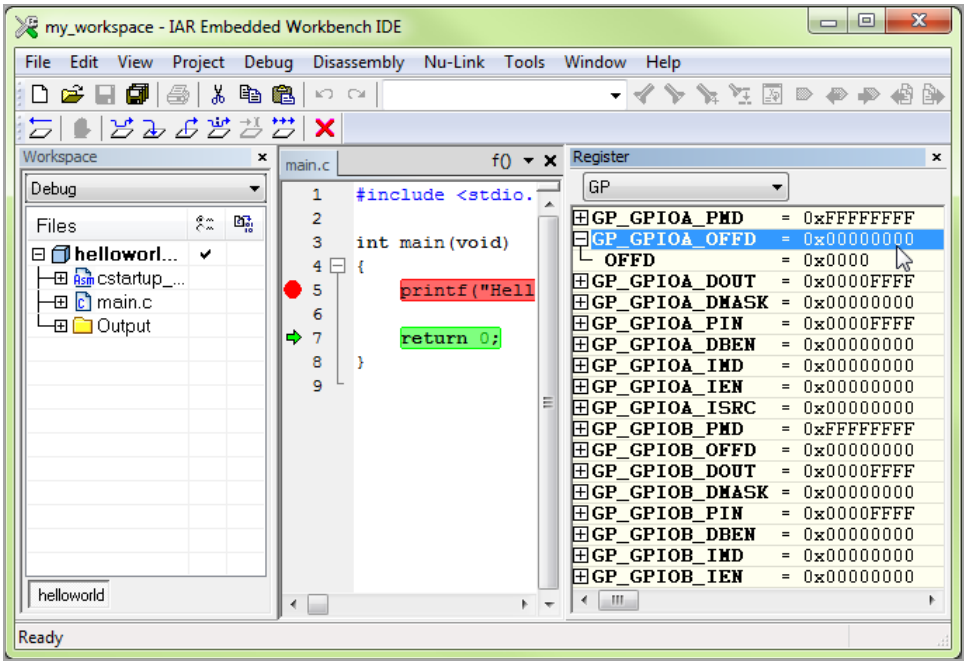

Figure 3-29 Access Other Registers

3.2.1 Live Watch

Nu-Link supports IAR “Live Watch” function which can periodically update the static or global variables and display the values in the “Live Watch” pane while target CPU running.

- In Debug mode, Select “**View**” → “**Live Watch**”, and then drag and drop the variables you want to observe in the “Live Watch” pane.

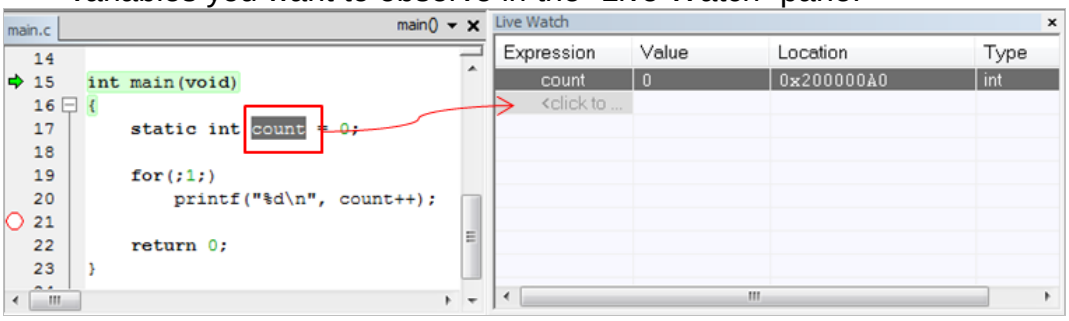

Figure 3-30 Live Watch Pane

- Right-click on the “Build” window, select “**Options**” → “**Debugger**”, and you can set the update frequency.

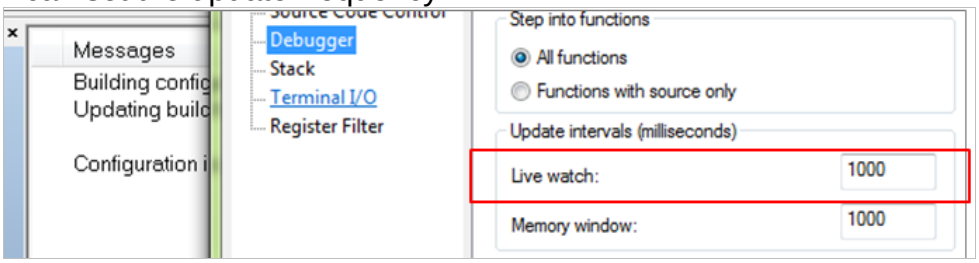

Figure 3-31 Set Live Watch Update Frequency

- Click the “**Go**” icon, and you can observe the variables. The automatic update will be performed periodically.

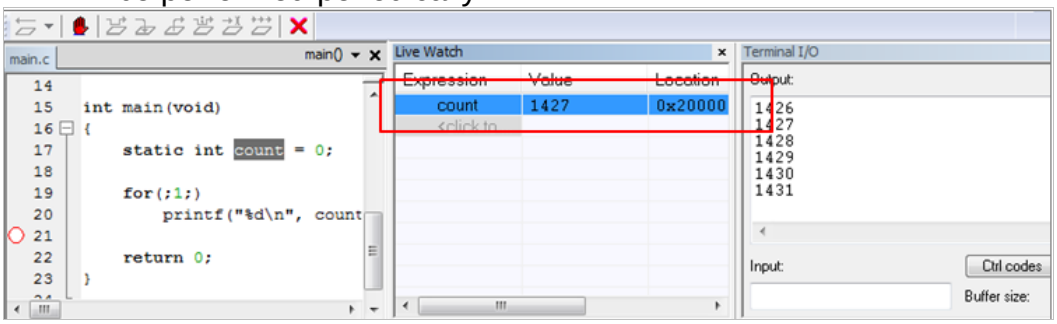

Figure 3-32 Live Watch Example

3.2.2 PinView Plug-in

In Debug mode, user can call “NuTool – PinView” from “**Nu-Link**” → “**NuTool – PinView**”, and then check the correctness of pin assignment through GUI.

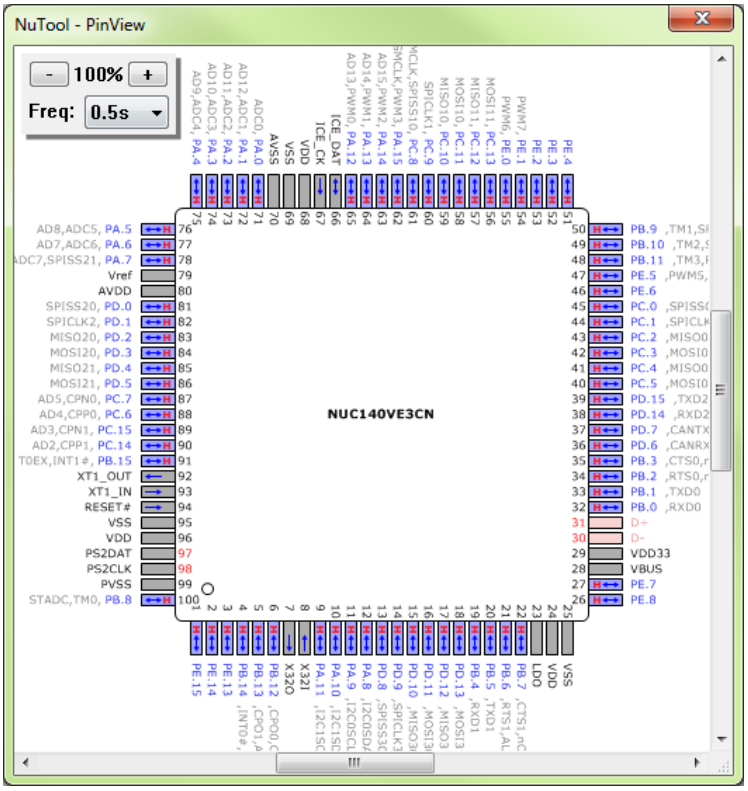

Figure 3-33 NuTool - PinView

3.2.3 Semihosting

The Nu-Link supports some semihosting functions for user to easily output messages in IAR EWARM IDE. The default option settings support semihosting library. As the example project above, printf outputs the message in the “Terminal I/O” pane.

The only problem in this example is that it lacks the support of outputting message on real target device. If the program does not run in ICE debug mode, but run directly in flash memory, it would be halted by function printf.

To solve the problem described above, please follow the steps below:

1. Open the "*C:\Program Files (x86)\Nuvoton Tools\Nu-Link_IAR\Samples\NUC140_PWM*" directory and copy the following two files into your project folder:

   - *SH_retarget.c*
   - *SH_startup_NUC1xx.s*

2. Then open your project, and

   - Replace *retarget.c* (if exists) with *SH_retarget.c*.
   - Replace *startup_NUC1xx.s* or *startup_M.s* (if exists) with *SH_startup_NUC1xx.s*.

3. To reduce the code size, you can undefine the macros in *SH_retarget.c* to disable the real UART or semihosting terminal or both.

   - #define DEBUG_ENABLE_UART
   - #define DEBUG_ENABLE_SEMIHOST

4. Build and run again. Now you can test the semihosting function again.

   - If debugging with Nu-Link:

     Select "**View**" → "**Terminal I/O**" to open the "Terminal I/O" pane.

     When the program calls printf, it will show messages in the "Terminal I/O" pane.

   - If running program in flash directly without the debugger being connected:

     When program calls printf, it will show messages in real UART if DEBUG_ENABLE_UART is defined.

3.2.4 NuConsole Plug-in

In addition to semihosting, the Nu-Link provides another I/O mechanism without affecting the target's real time behavior. By using standard debug port SWD, It doesn't need any additional pin or hardware. The only requirement is that the target application should reserve a buffer space (hereinafter referred to as **InfoBlock**) in SRAM in order to store control settings and communicate I/O data between NuConsole and target. To use the NuConsole functions, please follow the steps below:

1. Open the "*C:\Program Files (x86)\Nuvoton Tools\Nu-Link_IAR\NuConsole_Sample*" directory and copy the following files into your project folder:
   - *NuConsole.h/.c*
   - *NuConsole_Config.h*
   - *NuConsole_Retarget.c*
2. Set up the project
   - Replace *retarget.c* (if exists) with *NuConsole_Retarget.c.*
   - Add *NuConsole.c* to the project and include *NuConsole.h* in the corresponding files.
3. Configure InfoBlock
   - In *NuConsole_Config.h*, adjust the appropriate size of TX/RX buffers according to the application requirements and hardware limitation. Also, the TX buffer can be configured to be blocking or non-blocking.
   - Call **NuConnsole_Init()** function to initialize InfoBlock before doing I/O operations (e.g. printf()).
4. Enable the "*Generate linker map file*" option and build the project
   - In the linker map "*project_name.map*" file under "*project_path/Debug/List*" directory, find the value of symbol **NuConsole_InfoBlock** variable declared in NuConsole.c to get the memory address of InfoBlock.
5. Download and run

6. In Debug mode, select "**Nu-Link**" → "**NuConsole**" to invoke the control dialog. Set up the address of InfoBlock, and then click the start button to process I/O data.

7. Now when you execute the program and run the I/O statements, you can see debug information in the NuConsole dialog as shown below:

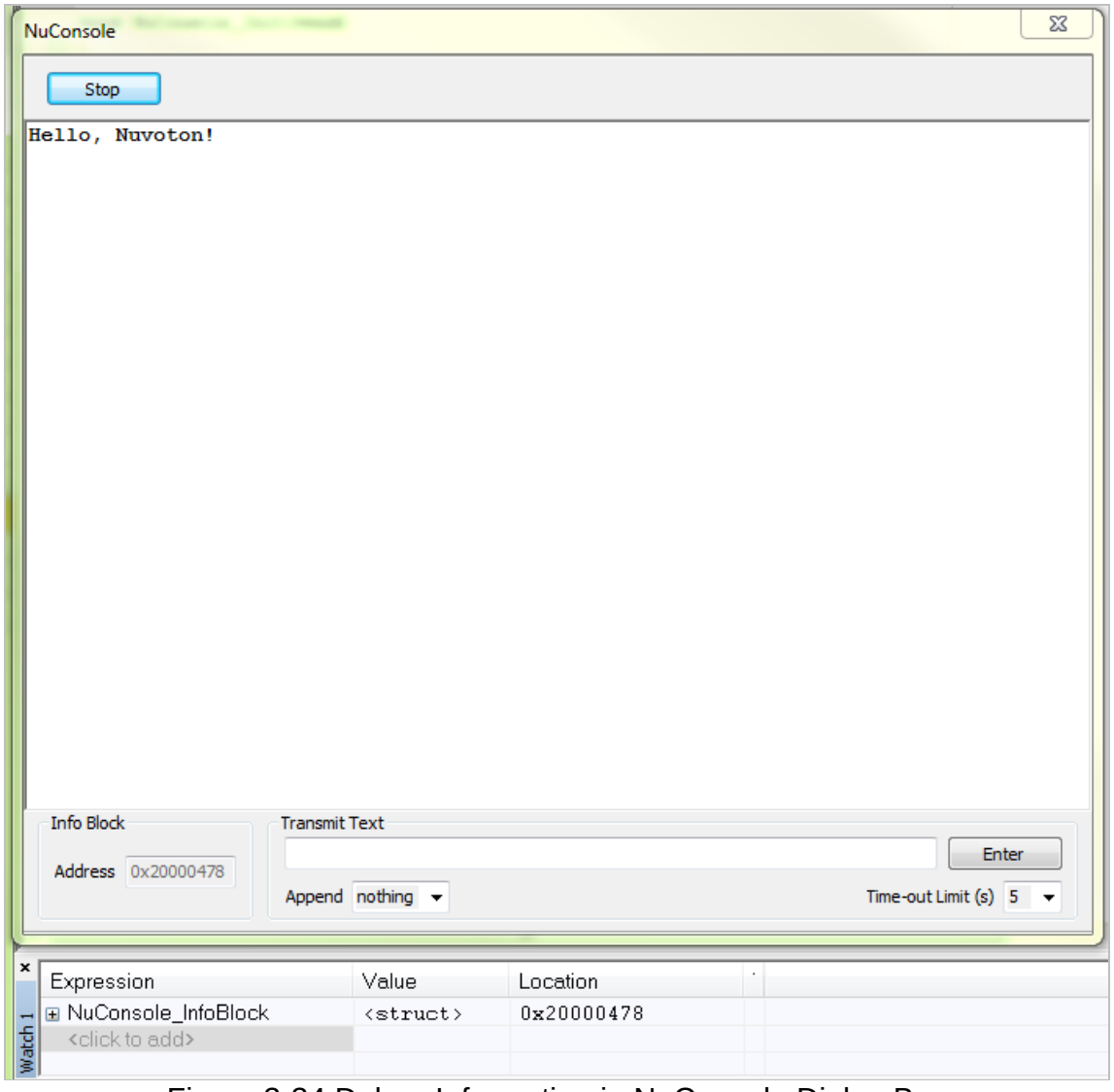

Figure 3-34 Debug Information in NuConsole Dialog Box

3.2.5 ETM Trace

To start Embedded Trace Macrocell (ETM) tracing on Nuvoton Cortex®-M4/M23 devices, please connect to the device using the Nu-Link2-Pro with 20-pin connector and follow the steps below.

1. Configure the NuTrace.
   - Select "**NuTrace**" from "**Nu-Link**" menu.
   - In "**Trace Port**" select **Sync Trace Port with 4 bit data**. It is possible to use other bit sizes but best to use the largest to increase the bandwidth.
   - In "**Capture Mode**", specify whether trace data is collected before or after a trigger.
     - **Trace After**: Capture the trace information after the trigger point and stop capturing when trace buffer is full.
     - **Trace Before**: Capture the most recent trace information before CPU is stopped.
   - **Select Trace Enable** and **ETM Trace Enable**.
   - **Click OK** to save the changes.

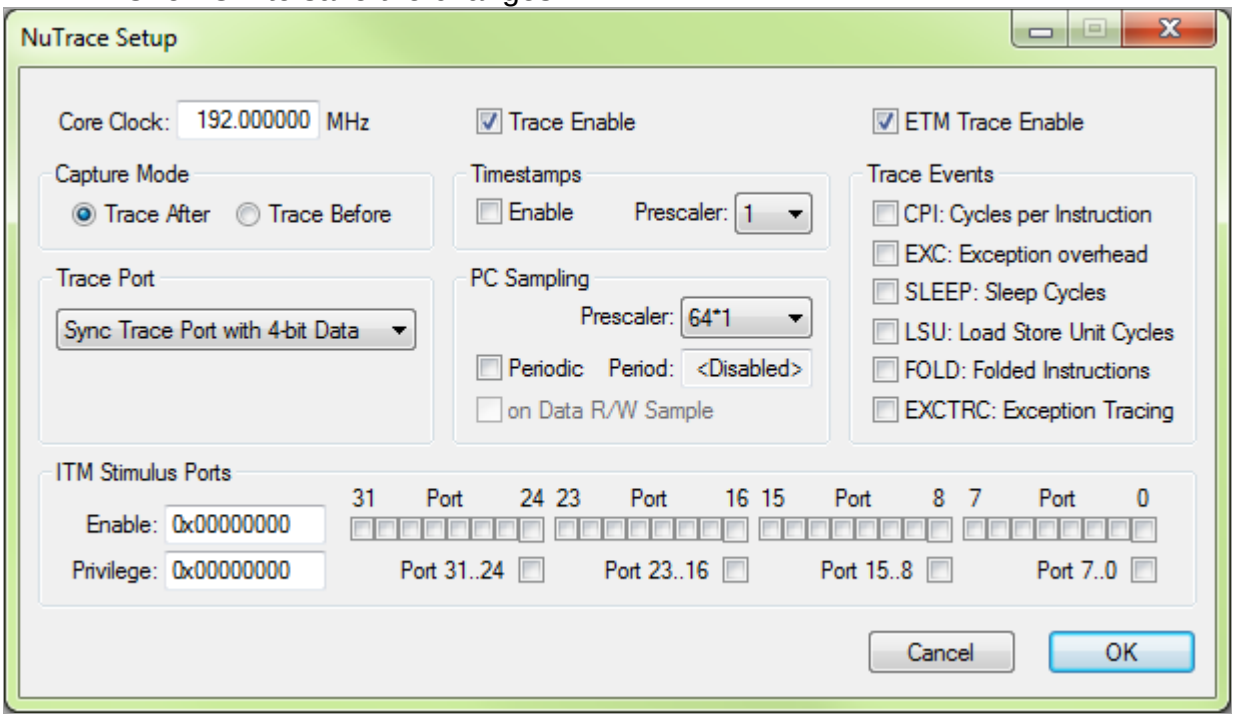

Figure 3-35 Trace Setup with ETM

2. In **Setup macros**, please insert the script file to initialize the device's trace pins when starting the debugger. The following is an example script file.

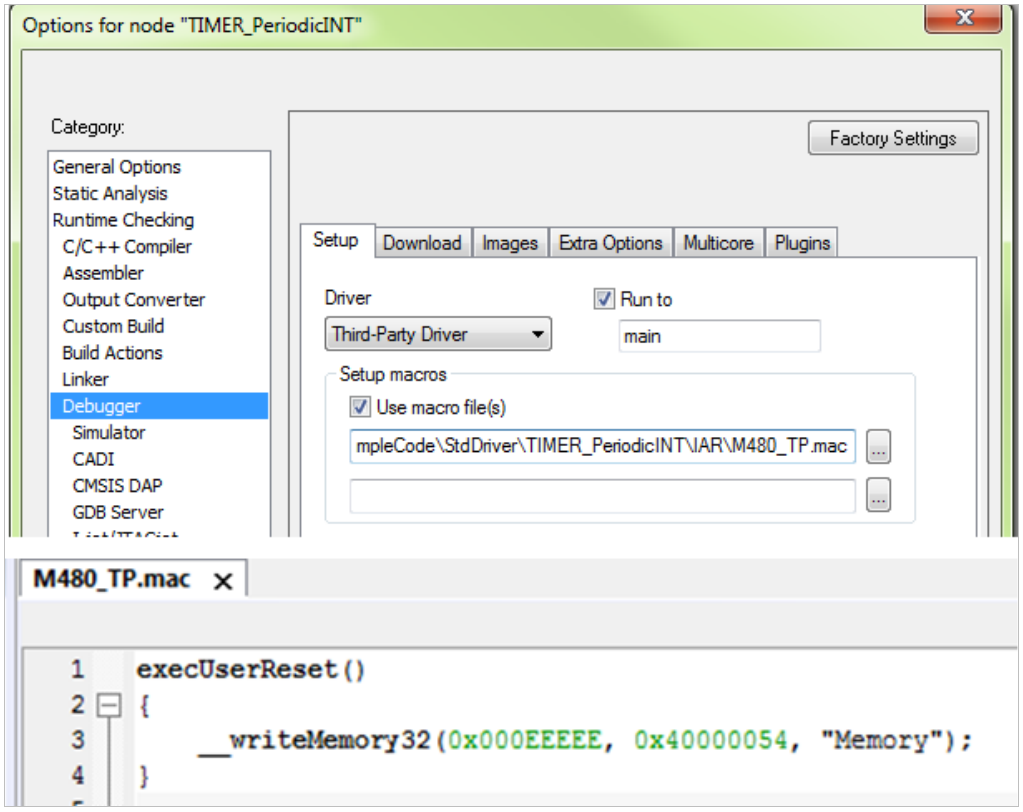

Figure 3-36 Initialize File for Trace Pins

Note: The Nu-Link driver with the version v2.07 or later will automatically setup the trace pins when starting the debugger. The user does not need to do the above configuration.

3. After doing above settings, user must start the debugger. In Debug mode, please select "Nu-Link" → "NuTrace" to invoke the tracing information dialog, and it will show every single executed instruction in the current application as shown below.

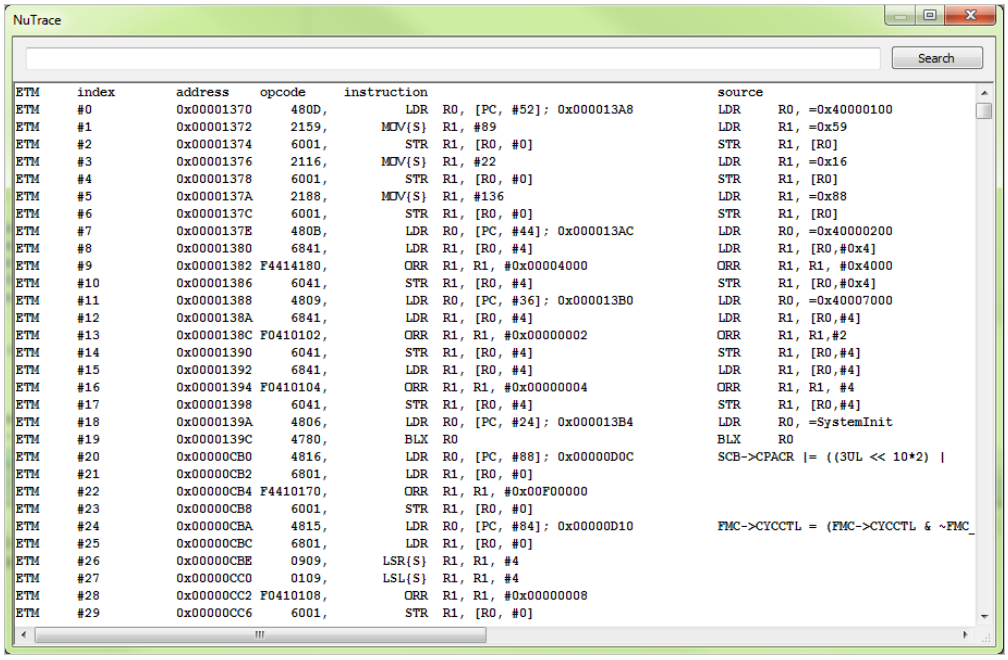

Figure 3-37 Tracing Information Dialog

3.2.6 LCDView Plug-in

In Debug mode, user can call “NuTool – LCDView” from “**Nu-Link**” → “**NuTool – LCDView**”. For details, please refer to the following link to the user manual:

https://www.nuvoton.com/resource-download.jsp?tp_GUID=UG1320220524115039

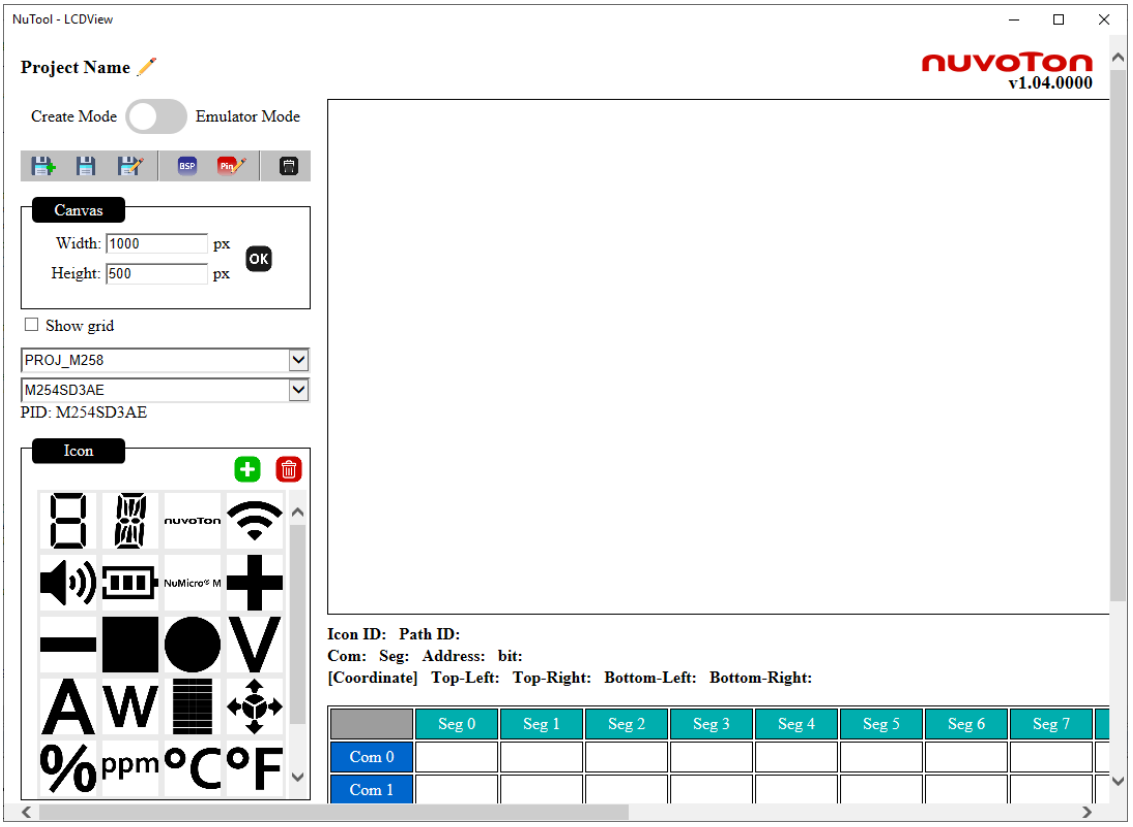

Figure 3-38 NuTool - LCDView

4 Firmware Update

When trying to debug a project using IAR EWARM, it will check the firmware version first. If the current firmware version is not consistent with the installed Nu-Link IAR Driver, a dialog box will pop up as follows:

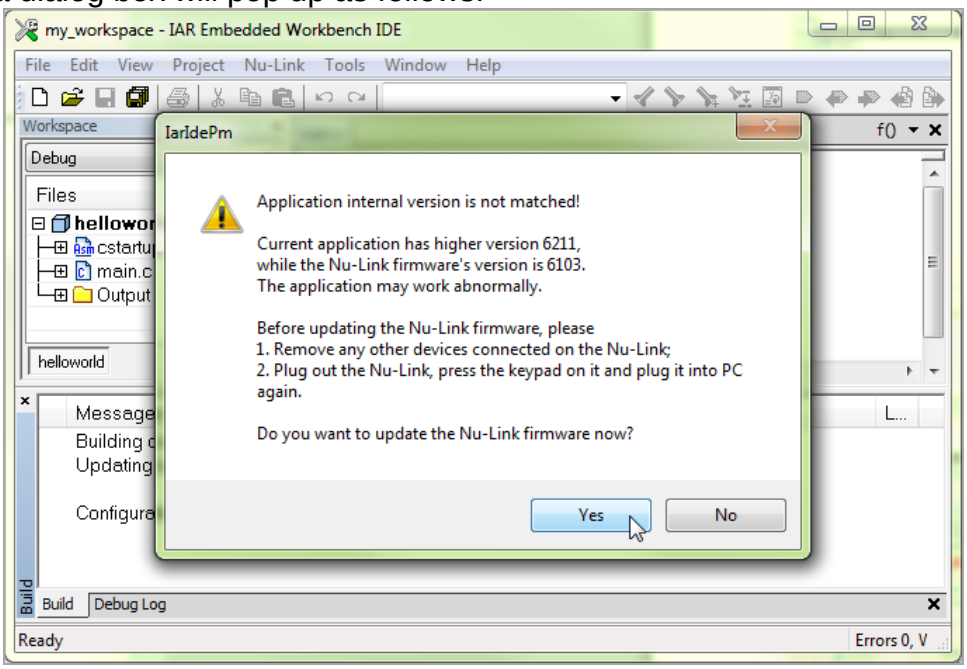

Figure 4-1 Firmware Update Selection Dialog Box

Click “**Yes**” to update firmware or click “**No**” to cancel.

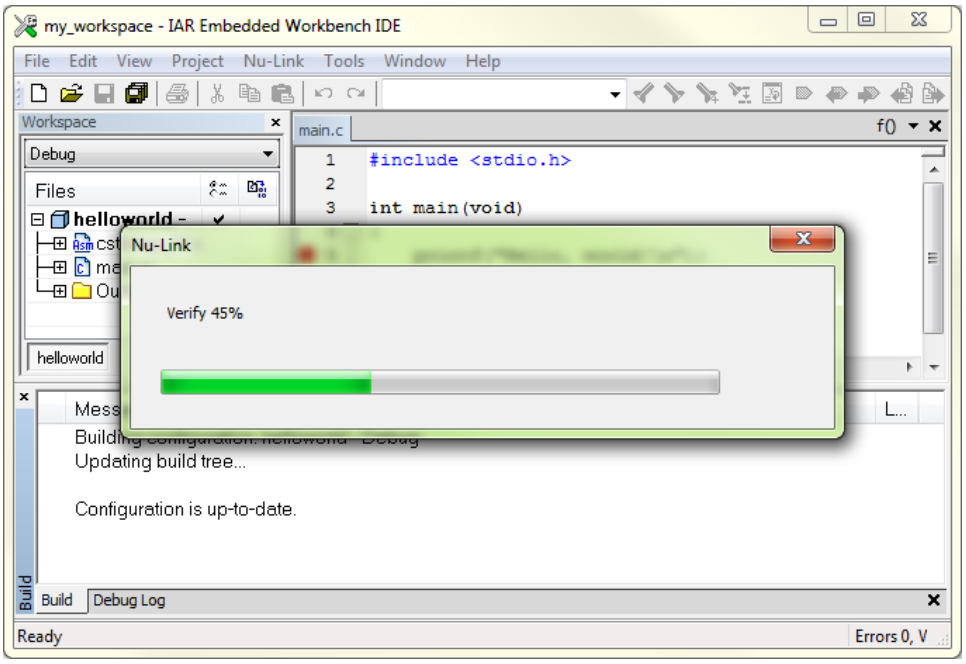

Figure 4-2 Updating Firmware

When update is complete, it is necessary to recreate a connection between Nu-Link and PC. Please plug out the Nu-Link from PC and plug in again.

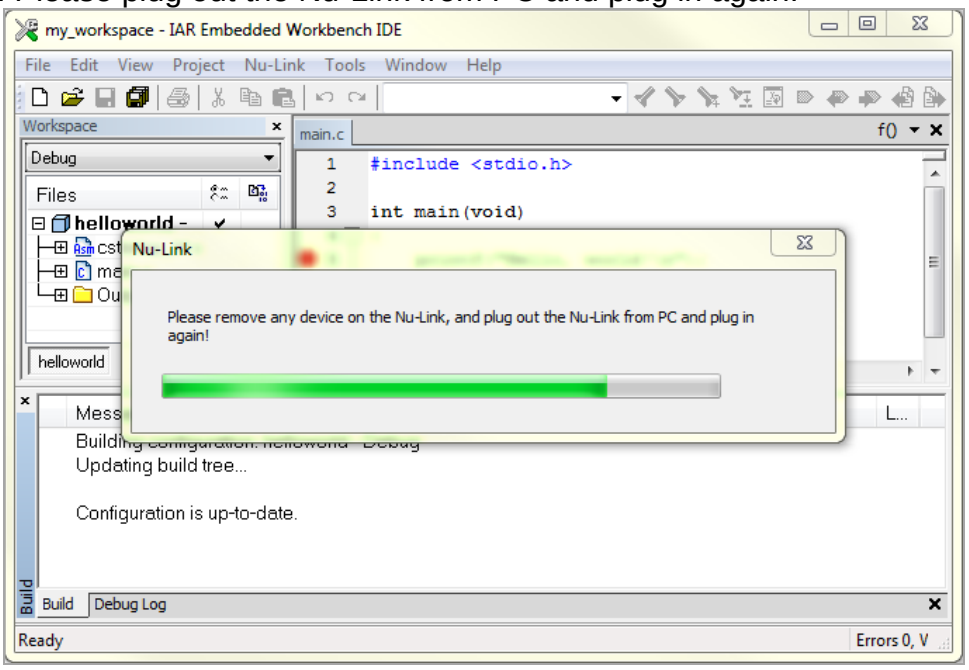

Figure 4-3 Re-connect Nu-Link to Complete Firmware Update

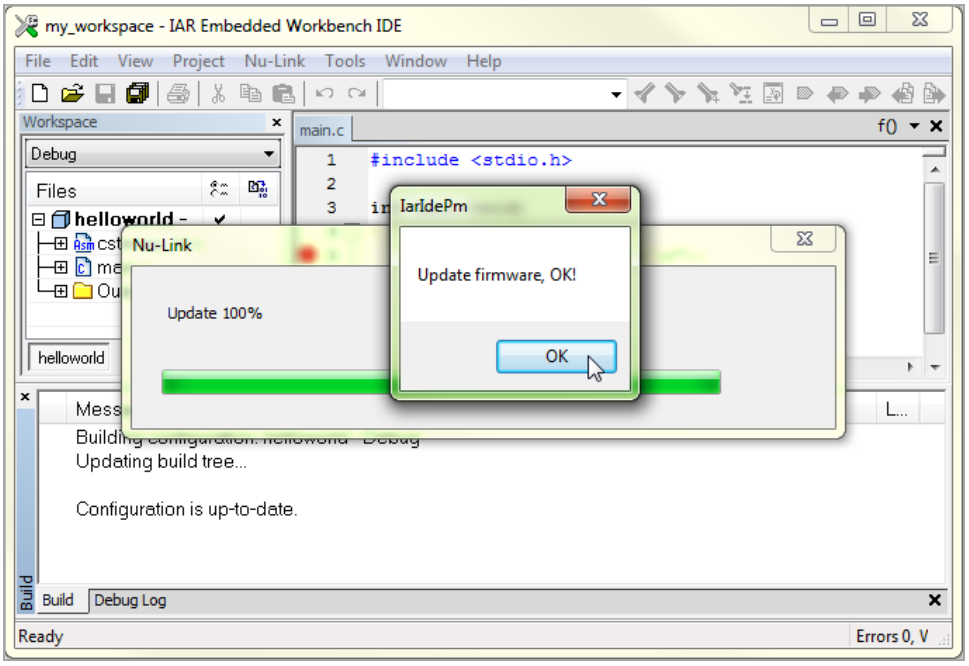

Figure 4-4 Update Firmware Completely

If you use Nu-Link2 adapter, you can also use drag and drop to upgrade firmware. Press the button on Nu-Link2 adapter and plug in USB cable, you will see a disk name “Nu-Link2”. Drag and drop bin file into it will upgrade Nu-Link2 firmware. (Note: if you see disk name “NuMicro MCU” it will upgrade the target device firmware instead of Nu-Link2 itself.)

5 Revision History

Revision	Date	Description
1.02	2010/07/07	Added the IAR's project settings.
1.03	2010/07/22	Added the M05x series.
1.17	2011/08/12	Used new configuration for NUC100 series.
1.18	2013/07/01	Added the "Nuvoton Announcement" and "Firmware Update" chapters.
1.19	2014/02/24	Changed document format, and updated all figures.
1.20	2014/10/17	Changed document and figure format.
2.01	2017/02/23	Added the "NuConsole" chapter.
2.07	2019/04/02	Added the "ETM Trace" chapter.