NuMicro Quick Start Guide for Keil

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

2014年2月10日 版本1.20

Table of Contents

Feature of Contents

1 Introduction

1.1 About the Quick Start Guide

This Quick Start Guide will instruct you on how to use the Keil™ Microsoft Windows based software development tools with the NUC1XX development board. It gives you an overview of the most commonly used features in Keil MDK µVision4 and provides the necessary information for your own projects.

1.2 About NUC1xx series IC

The NUC1xx series include NUC100, NUC120, NUC130 and NUC140 series.

The NUC1xx series are 32-bit microcontrollers with embedded ARM® Cortex™-M0 core for industrial control and applications which need rich communication interfaces. The Cortex™-M0 is the newest ARM embedded processor with 32-bit performance and at a cost equivalent traditional 8-bit microcontroller.

The **NUC100** series embeds Cortex™-M0 core running up to 50MHz with 32K/64K/128K-byte embedded flash and 4K/8K/16K-byte embedded SRAM. It also equips with plenty of peripheral devices, such as Timers, Watchdog Timer, RTC, PDMA, UART, SPI/SSP, I^2C, I^2S, PWM Timer, GPIO, 12-bit ADC, Analog Comparator, Low Voltage Detector and Brown-out detector.

The **NUC120** series embeds Cortex™-M0 core running up to 50MHz with 32K/64K/128K-byte embedded flash and 4K/8K/16K-byte embedded SRAM. It also equips with plenty of peripheral devices, such as Timers, Watchdog Timer, RTC, PDMA, UART, SPI/SSP, I^2C, I^2S, PWM Timer, GPIO, USB 2.0 FS Device, 12-bit ADC, Analog Comparator, Low Voltage Detector and Brown-out detector.

The **NUC130** series embeds Cortex™-M0 core running up to 50MHz with 64K/128K-byte embedded flash and 8K/16K-byte embedded SRAM. It also equips with plenty of peripheral devices, such as Timers, Watchdog Timer, RTC, PDMA, UART, SPI/SSP, I^2C, I^2S, PWM Timer, GPIO, LIN, CAN, 12-bit ADC, Analog Comparator, Low Voltage Detector and Brown-out detector.

The **NUC140** series embeds Cortex™-M0 core running up to 50MHz with 64K/128K-byte embedded flash and 8K/16K-byte embedded SRAM. It also equips with plenty of peripheral devices, such as Timers, Watchdog Timer, RTC, PDMA, UART, SPI/SSP, I^2C, I^2S, PWM Timer, GPIO, LIN, CAN, USB 2.0 FS Device, 12-bit ADC, Analog Comparator, Low Voltage Detector and Brown-out detector.

1.3 About Keil™ µVision4 IDE

The µVision4 IDE is a Windows-based software development platform that combines a robust editor, project manager, and makes facility. The µVision4 integrates all tools including the C compiler, macro assembler, linker/locator, and HEX file generator. The µVision4 helps expedite the development process of your embedded applications by providing the following:

- Full-featured source code editor,
- Device database for configuring the development tool setting,
- Project manager for creating and maintaining your projects,
- Integrated make facility for assembling, compiling, and linking your embedded applications,
- Dialogs for all development tool settings,
- True integrated source-level Debugger with high-speed CPU and peripheral simulator,
- Advanced GUI interface for software debugging in the target hardware and for connection to KeilTM ULINK,
- Flash programming utility for downloading the application program into Flash ROM,
- Links to development tools manuals, device datasheets & user's guides.

2 Quick Start

2.1 Installing the Keil™ Software

You can download the Keil™ RealView® Microcontroller Development Kit Evaluation software from http://www.keil.com/. It contains the Keil™ µVision4 IDE. The evaluation version of the tools has a 32K bytes limit on images, but comes in a license-free version.

More information please reference Read Me First document from Keil™ about how to install Keil™ µVision4 software.

2.2 Connecting to your target

The target is powered via your PC, through its USB port or 5volt DC adaptor. The Keil™ ULINK family of adapter connects the USB port of your PC to the Serial Wire Debug (SWD) port of your target board allowing you to download and debug embedded programs running on your target hardware.

The ULINK2 connects to the NUC1xx using the 20-pin ribbon cable.

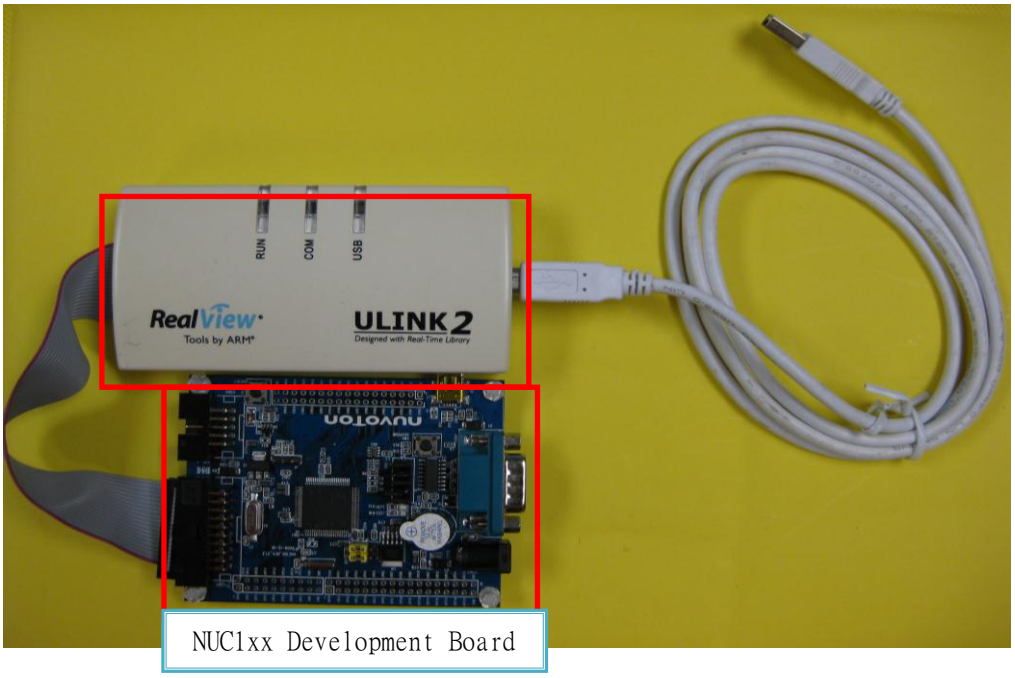


Figure 2-1

2.3 µVision4 Overview

µVision4 has two operating modes:

- **Build Mode**: Allows you to translate all the application files and to generate executable programs. The features of the Build Mode are described under Creating Applications.

- **Debug Mode**: Provides you with a powerful debugger for testing your application. The Debug Mode is described in Testing Programs.

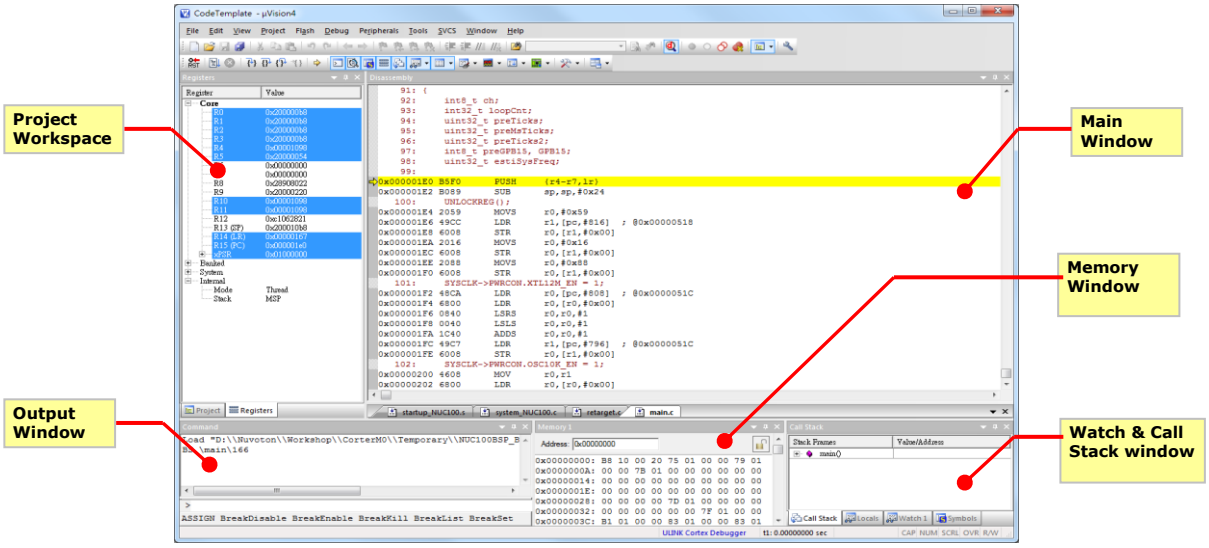


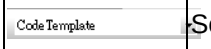
Figure 2-2

2.3.1 Build Process

The Build Target command runs the Compiler and Assembler. The tools automatically generate file dependencies so only those files that have changed are retranslated. You may enable additional Global Code Optimizations which are performed by incremental re-compilations of C modules and other utilities. The Project menu provides access to project files and dialogs for project management.

Build Toolbar

Only describe functions in common use as shown as below:

Command Option	Tool Button	Function Description	Hot key
Translate…		Translate current file	None
Build Target		Translate modified files and build application	F7
Rebuild Target		Re-translate all source files and build application	None
Batch Build		Execute build component on the selected project/targets of a Multi-project workspace	None
Stop Build		Stop current build process	None
Flash Download		Call Flash download utility as configure under options.	None
Target Option		Set project components, configure tool environment and manage books.	None
Select Current Project Target		Select current target	None
Manage Project		Set Project Component, Configure tool environment and manage books.	None

2.3.2 Debugger

The µVision4 IDE/Simulator/Debugger accelerates your learning efforts by providing a single environment for editing, simulating, and testing target hardware. Most debugger and editor functions may be quickly accessed from the toolbar.

You may use the context menu or the Editor Toolbar to insert breakpoints. Breakpoints you set while editing are activated when you start your debugging session. µVision4 marks the status of each source line in the Attributes column of the editor window. This provides a quick overview of the current breakpoint and execution status.

Debugger

Only describe functions in common use as shown as below:

Command Option	Tool Button	Function Description	Hot key
Reset CPU		Set CPU to reset state	Ctrl+F5
Go		Run until the next active breakpoint	F5
Halt Execution		Stop running	ESC
Single step into		Execute a single step into a function	F11
Step Over		Execute a single step over a function	F10
Step Out		Step out of current function	None
Run till current line		Execute until the current cursor line is reached	None
Show next statement		Show next executable statement /instruction	None
Disassembly		Show or hide Disassembly window	None
Watch & Call Stack window		Show or hide Watch & Call Stack window	None
Memory window		Show or hide Memory window	None

2.4 Step-by-Step

This section details all of the materials necessary to download code to an ARM-based development board for debug in the KeilTM µVision4 IDE using the JTAG debug agent.

2.4.1 Starting the software

Double-click on the Keil uVision4 icon to start the user interface. The compiler, assembler, linker and Scope will be called from within Keil uVision4 in this tutorial. After you invoke Keil uVision4, the window shown as below appears. From this window, you can create projects, edit files, configure the tool, assemble, link, and invoke the debugger.

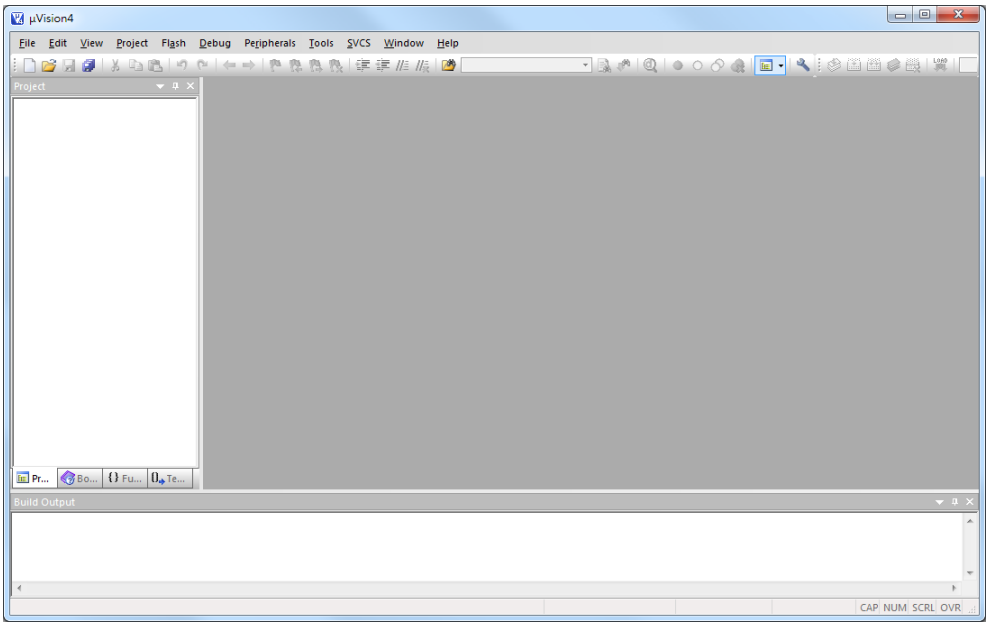


Figure 2-3

2.4.2 Creating a new project

Before writing any C-code, a project associated with our code needs to be created. This is done by first creating a new folder in the Keil directory in which your project will be saved. Next the Keil uV4 application can be launched and a new project is created. This is achieved by completing the following steps.

- Create a folder named “CodeTemplate” in your setting path
- Launch the uV4 application. Start -> Programs -> Keil µVision4
- Create a new project. From the main window, choose the ‘Project’ menu and select New project . And then a new window appears as shown below
- Select the folder that you've created previously (CodeTemplate) and on the bottom of the window type the name of your new project, eg. CodeTemplate and press SAVE.

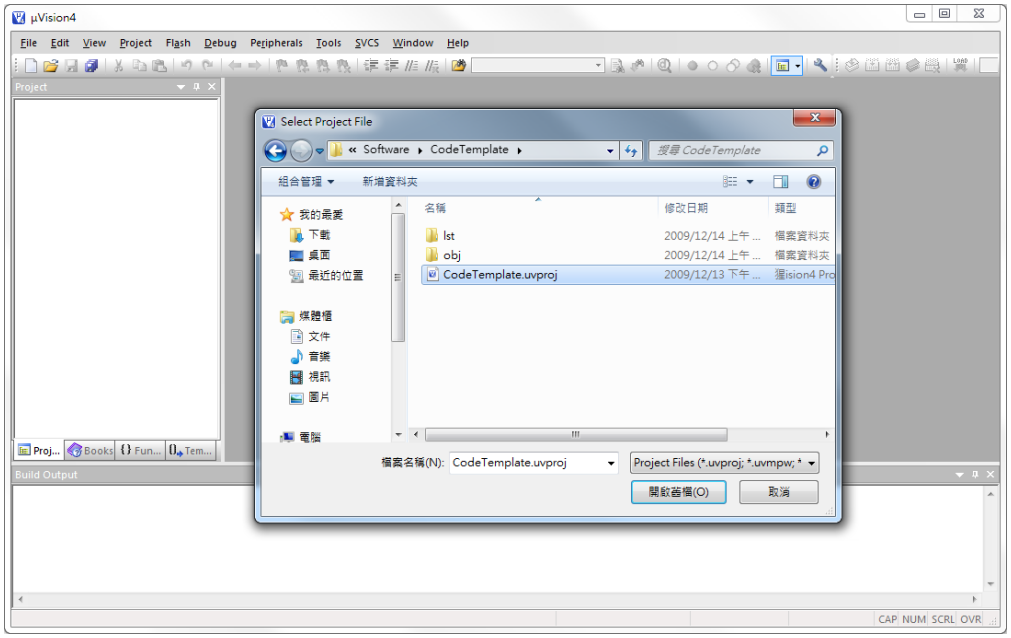


Figure 2-4

2.4.3 Device Support

A new window appears as shown below and you are now required to configure your setup to target the specific ARM device you wish to use (in this example we will be using the Cortex-M0) and the output file format generated after the compilation stage. This is achieved by completing the following steps.

- Open the ARM folder.
- Select the item on which you will be developing, in this tech note we will be using the Cortex-M0 as the target example.

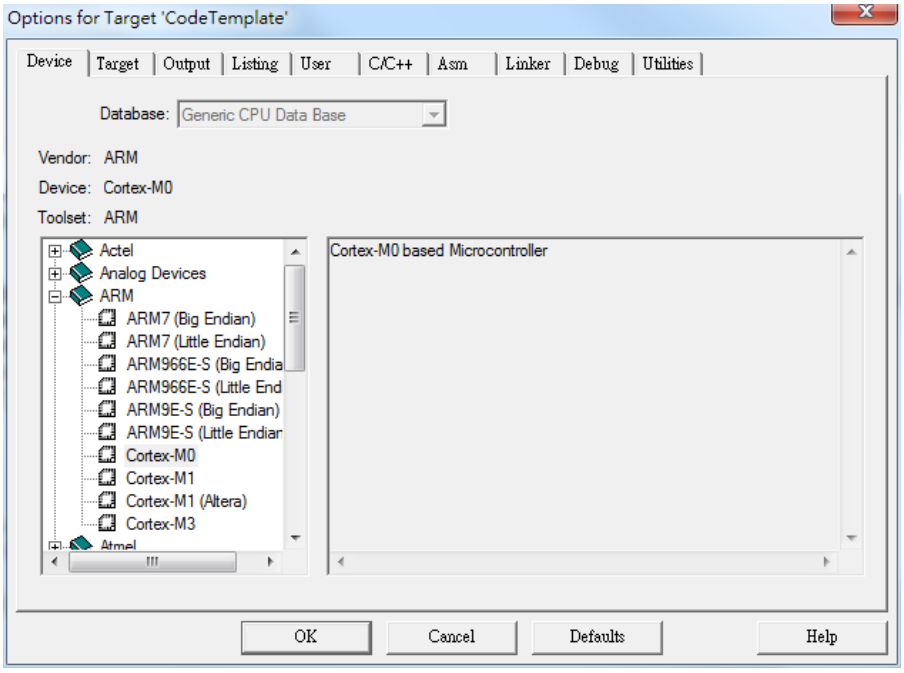


Figure 2-5

2.4.4 Project Management

Keil uVision4 ensures easy and consistent Project Management. A single project file stores source file names and saves configuration information for Compiler, Assembler, Linker, Debugger, Flash Loader, and other utilities. The Project menu provides access to project files and dialogs for project management.

Select the icon to open Management Project Component Setting

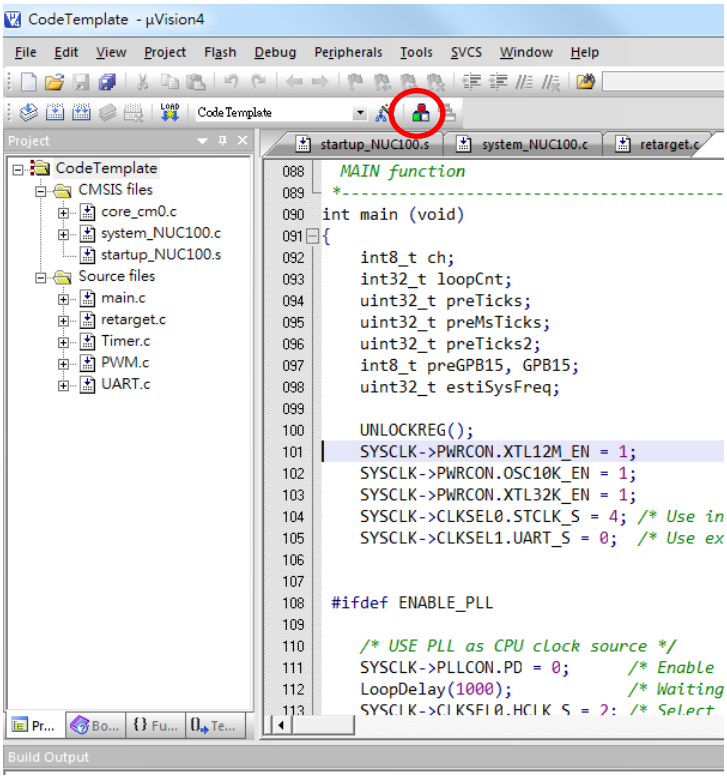


Figure 2-6

A Project Component Setting window appears as shown below. According to your assignment, create new group and link your source code.

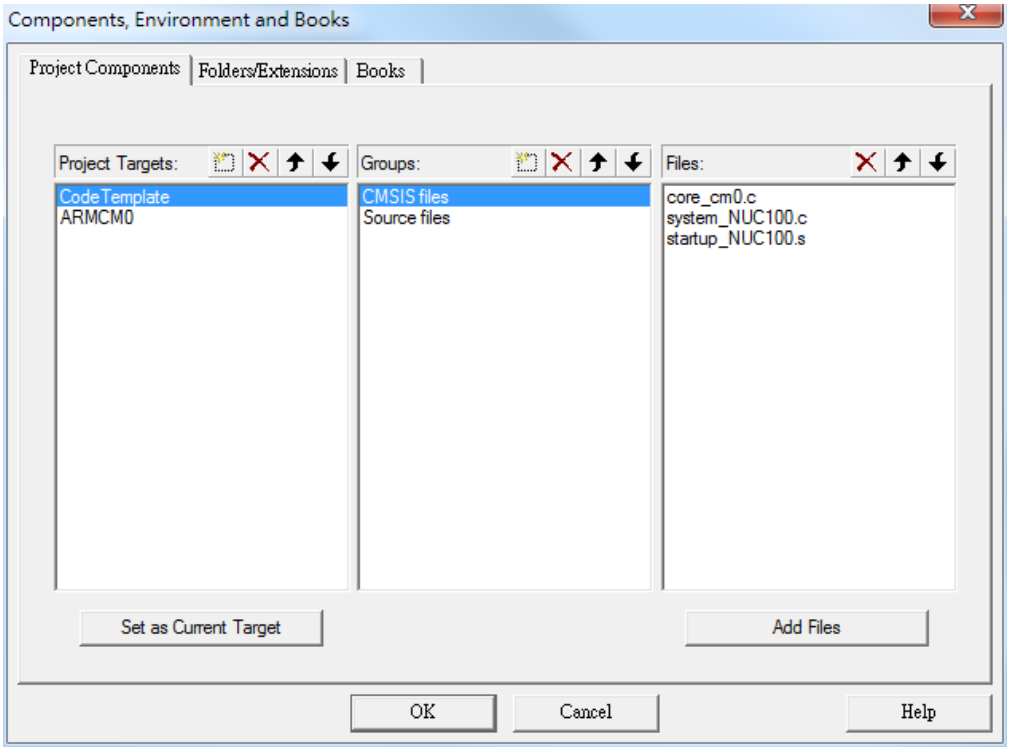


Figure 2-7

Finally, the project workspace will show as below:

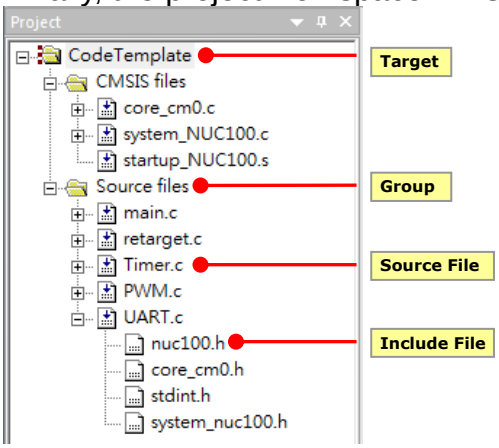


Figure 2-8

2.4.5 Creating a C program

Now you also can write your C program. In the main window, choose the File pull-down menu and select New. A new window named <text1> will appear on the screen and you can write your code to <text1>.

Once you've typed all the code, again choose the File pull-down menu and select Save. A new Save dialog window appears. Save your new file as “main.c” in the CodeTemplate folder you had created earlier. At this stage, before compiling the C-program, we need to include it in our project. To do this you must click with the right mouse button on ‘Source files ’and select Add Files to Group ‘Source Library’ as shown below. Select the file named “main.c“ that is in the CodeTemplate folder and click on Add and then on Close.

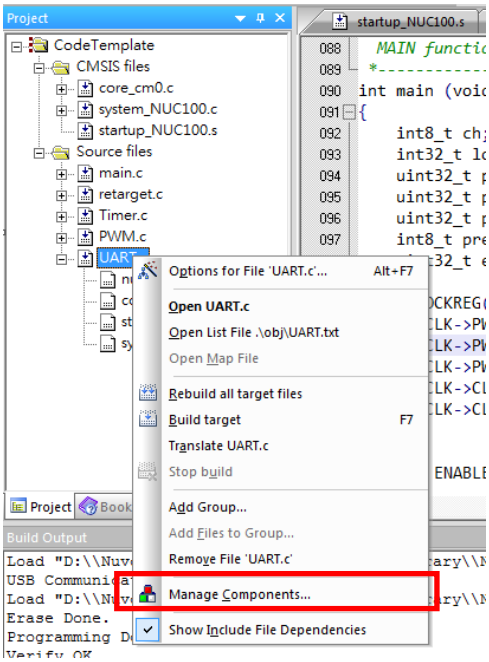


Figure 2-9

2.4.6 Compiling a C program

Select Rebuild all target files from the Project menu, or click on the Rebuild all button (icon).

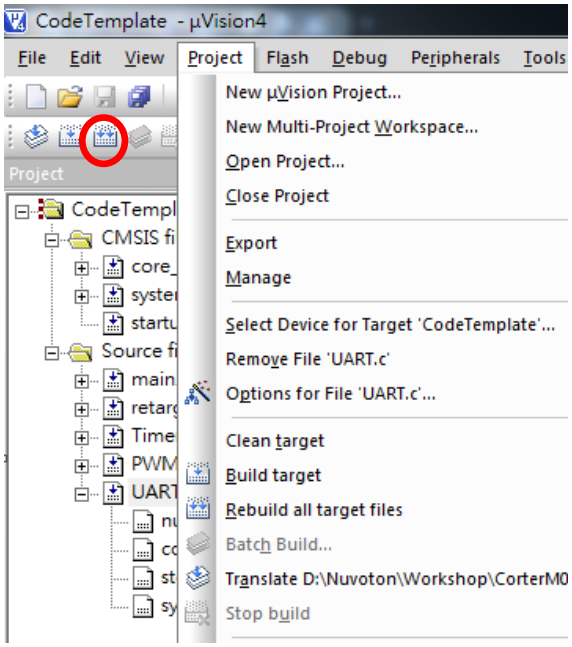


Figure 2-10

All of the source files are compiled and linked. The activity can be seen in the Build window at the bottom of the µVision4 IDE. (In this example, the process completes with an application named CodeTemplate.axf and CodeTemplate.bin built with no errors and no warnings.)

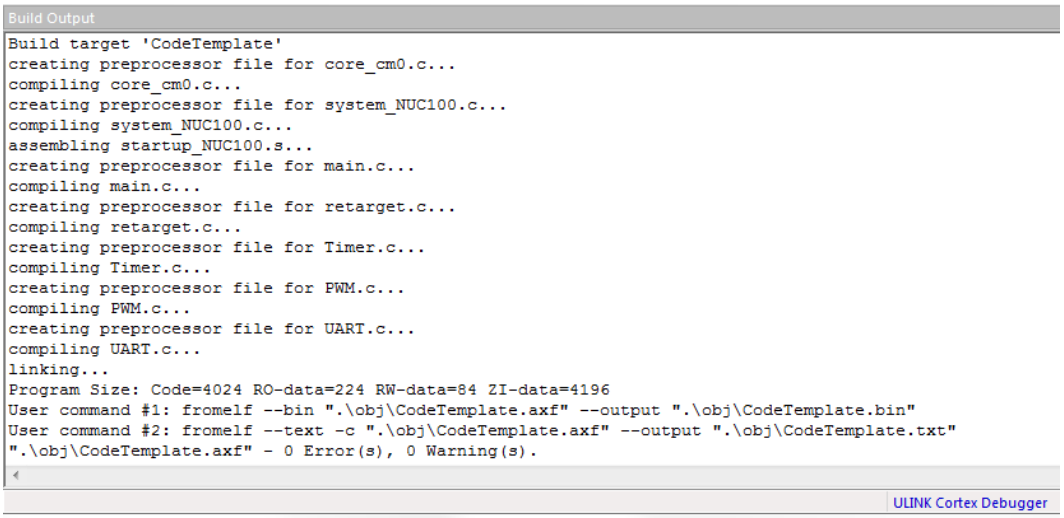


Figure 2-11

2.4.7 Connecting and Configuring the Hardware

Step-by-step to finished the section.

1. Click on Project => Options for Target => your_target_name. Or click on the Options for Target button (icon).

Figure 2-12

2. On Target tab, allows you to specify CPU and memory options. These are used to configure basic tool settings including those of the linker, debugger, and simulator.

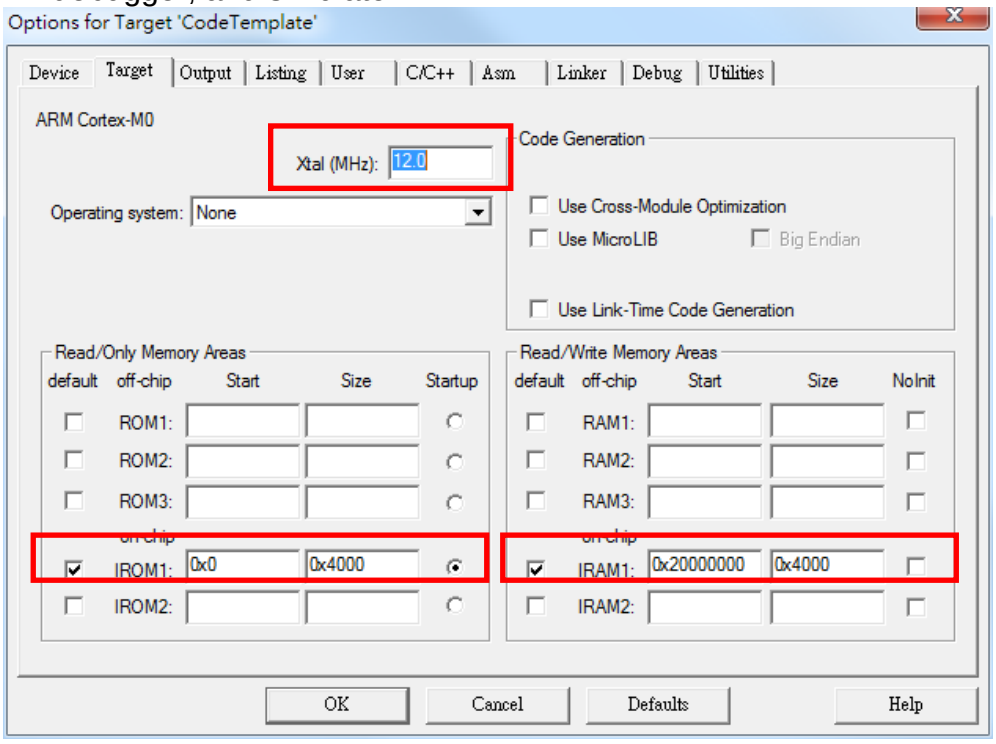


Figure 2-13

3. The setting dialog show up in a window on the center. Click Debug tab and Select ULINK Cortex Debugger. You might need to select this from the drop-down menu if it is not already selected. If you want to load the application when starting debug mode, please enable the check box of Load Application at Startup.

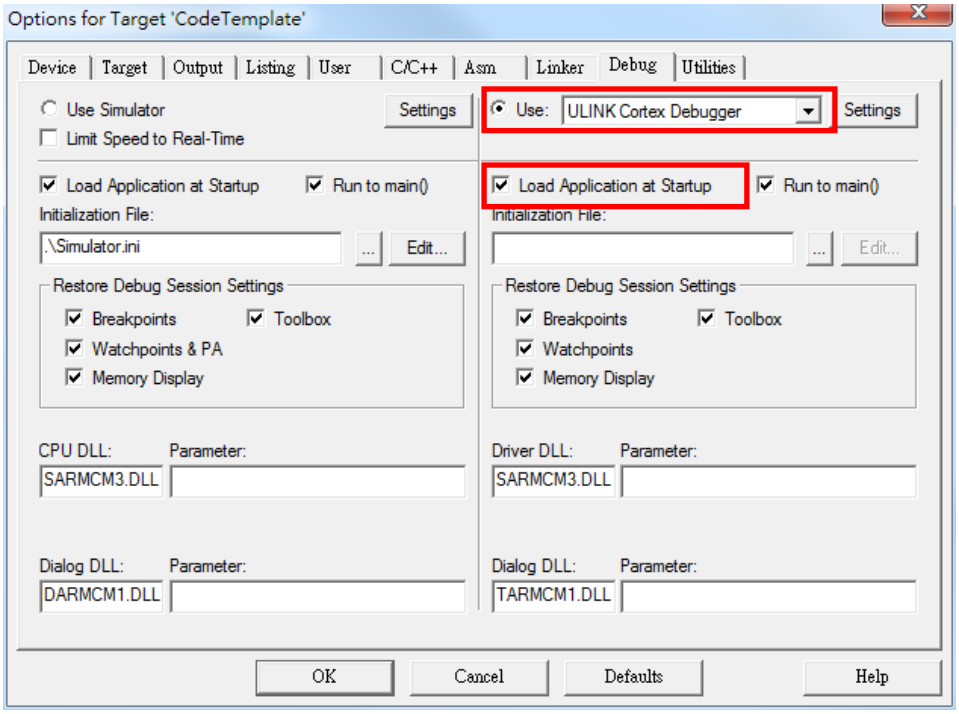


Figure 2-14

2.4.8 Simulating your source code

Another powerful feature of the uVision4 IDE is that it allows you to run your code. To start a simulation session you simply click on the on **Start/Stop Debug Session** option available from the **Debug** pull-down menu. Alternatively you can press <Cntrl+F7> or the 'Debug' icon available in the 'File' toolbar as shown as below.

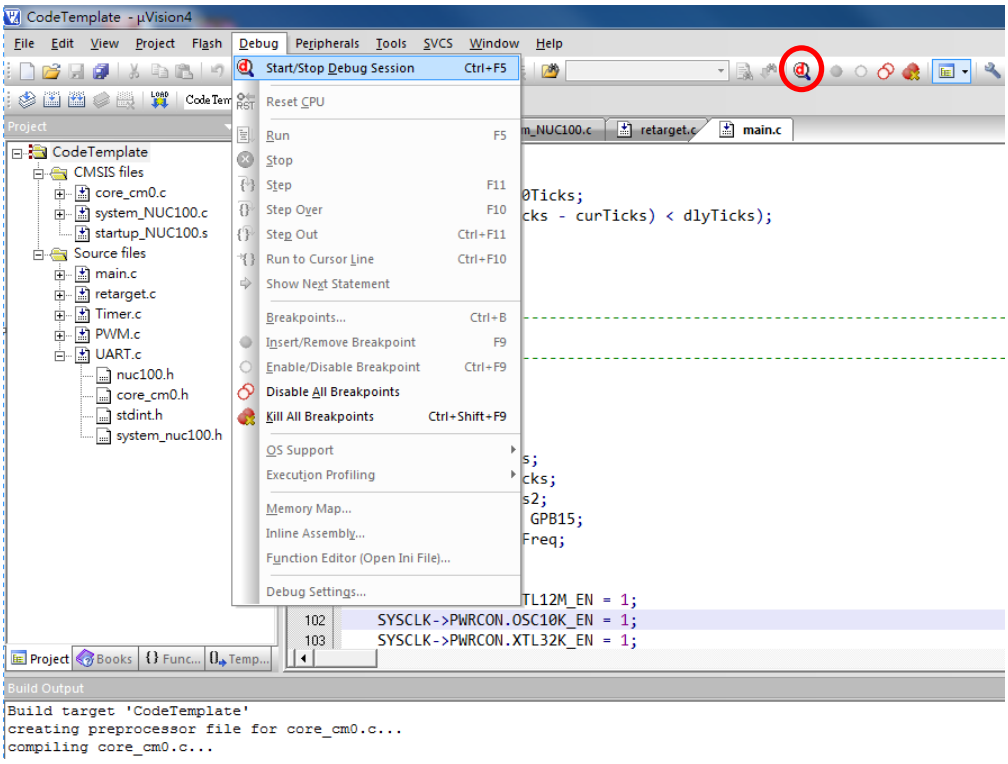


Figure 2-15

The IDE switches to debugging mode. The processor registers show up in a window on the left, the debugger command window is visible at the bottom, and the main window shows the source code being debugged.

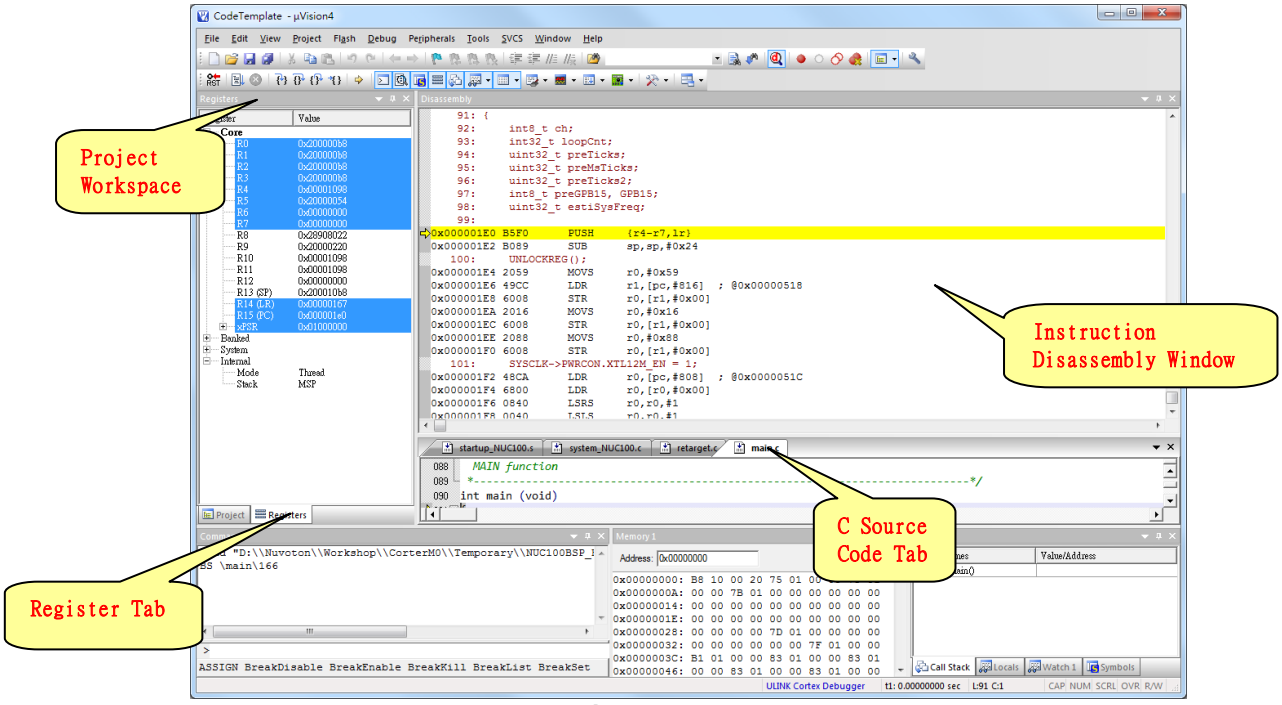


Figure 2-16

From here, you can examine and modify memory, program variables, and processor registers, set breakpoints, single step, and all other typical debugging activities. To run the program, select Run from the Debug menu, or click on the Run button (icon).

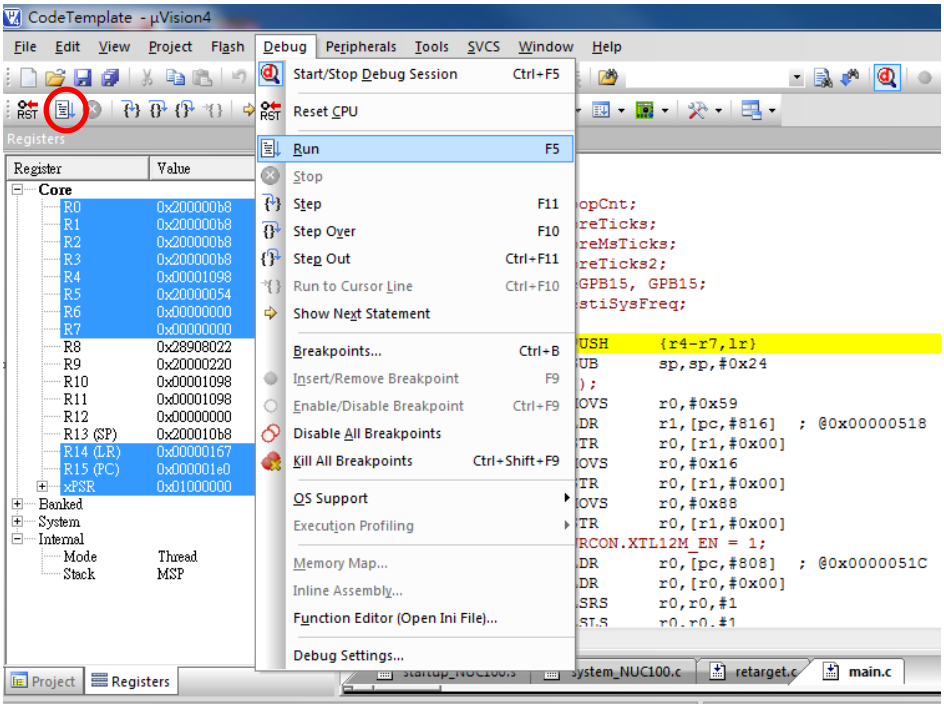


Figure 2-17

2.4.9 Flash Tool

Keil tool chain supports to download image file to NUC1xx DEV Board through ICE interface. A specified flash tool file should be provided to let Keil flash download function work. For NUC1xx series IC, the flash tool is called NUC1xx_AP_128.FLM NUC1xx_AP_64.FLM NUC1xx_AP_32.FLM NUC1xx_LD_4.FLM for different flash size and this file could be found at \FlashTool. To install the NUC1xx_AP_128.FLM, we need to copy it to the install directory of Keil, i.e. C:\ARM\Flash. After copy NUC1xx_AP_128.FLM to the specified directory, we can go back to the Keil IDE, open the Options for Targets dialog, select Utilities tab and open the Settings dialog.

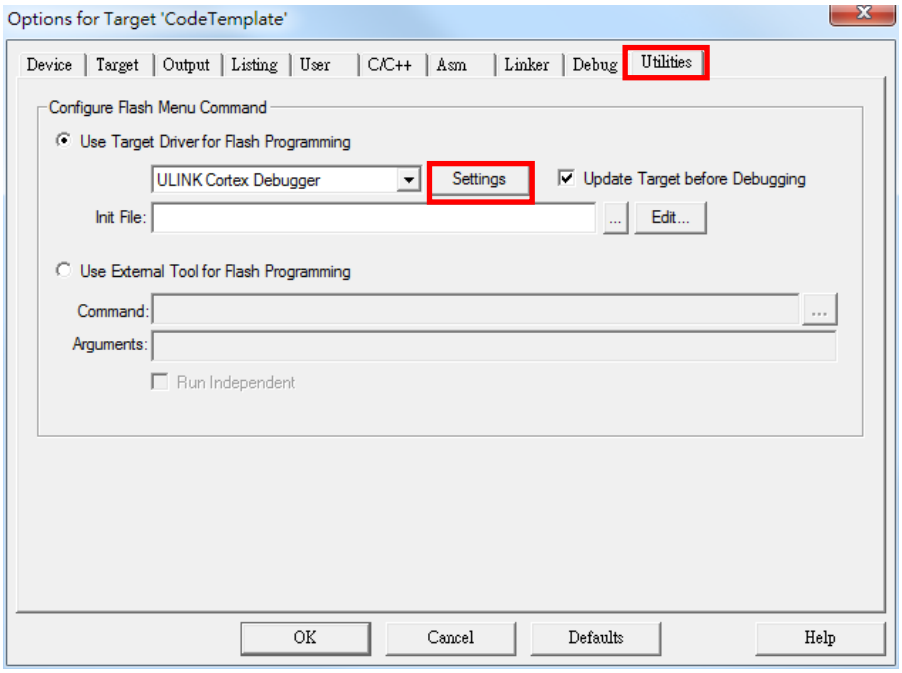


Figure 2-18

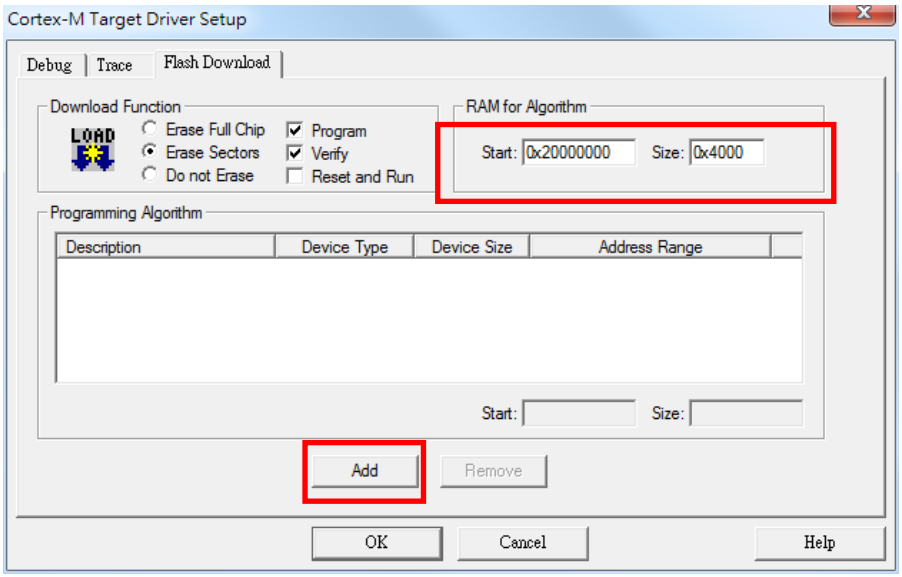


Figure 2-19

After the Settings button has been pressed, the Flash Download Setup dialog would be opened. Then we can set the start address to be 0x20000000, size to be 0x4000 in RAM for Algorithm setting. Finally, we press the Add button to add the flash tool. If the NUC1xx_AP_128.FLM was copy to \Keil\ARM\Flash, we should be able to find NUC1xx 128kB Flash AP item in Add dialog:

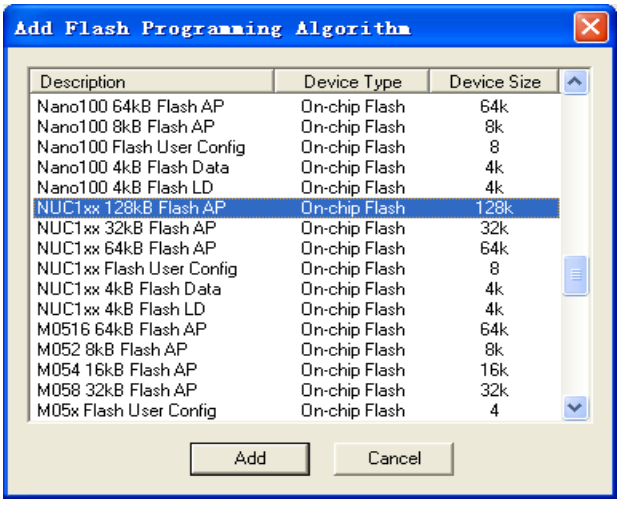


Figure 2-20

After find the NUC1xx 128kB Flash AP item, press Add button to add it to Flash Download Setup dialog.

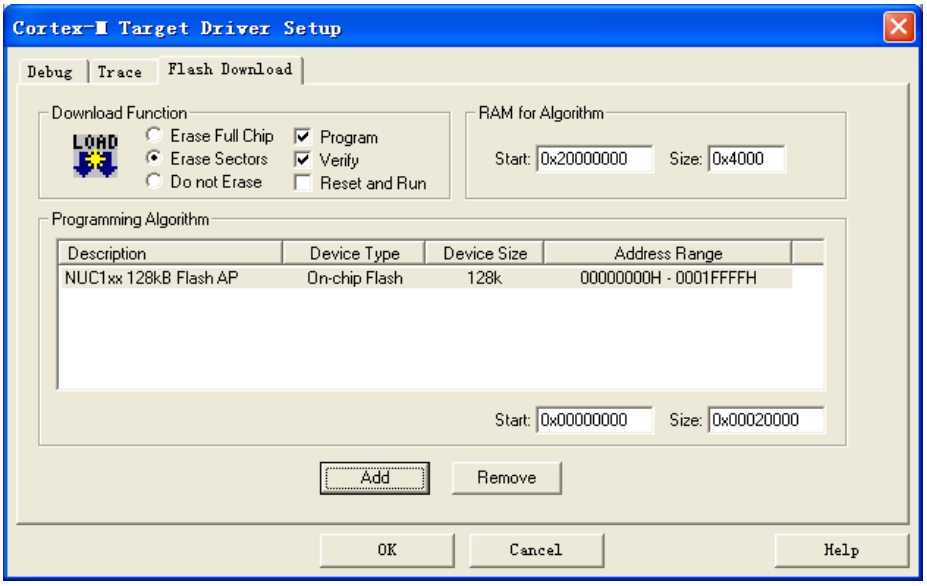


Figure 2-21

Now, we can press ok to finish the setting.

Note: Due to the flash base address is 0x00000000, we should set the RO base of linker according to the flash base address.

2.4.10 Conclusion

You have now installed the KeilTM RealView® Microcontroller Development Kit, and used it to build, load, and run a demonstration application on your Nuvoton® Development Board. From here, you can experiment with the debugger or start creating your own application using the CodeTemplate program as an example. If CodeTemplate project sample starts running, and you should see some text output to the hyper terminal display as shown as below:

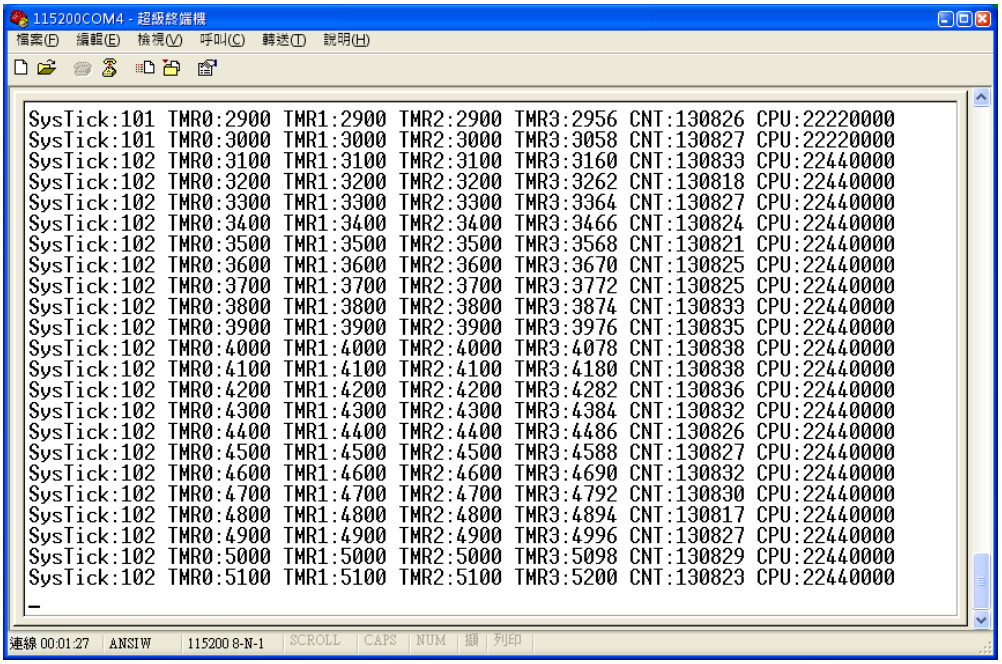


Figure 2-22

3 Revision History

Revision	Date	Description
V1.01	Jan. 28, 2010	● Created.
V1.02	Jun. 22, 2010	● To change NUC100.FLM file which depends on flash size by using NUC1xx_128kB.FLM NUC1xx_64kB.FLM NUC1xx_32kB
V1.03	Jul. 22, 2010	● BTo change NUC1xx_128kB.FLM NUC1xx_64kB.FLM NUC1xx_32kB.FLM to NUC1xx_AP_128.FLM NUC1xx_AP_64.FLM NUC1xx_AP_32.FLM and Add NUC1xx_LD_4.FLM
V1.04	Feb. 10, 2014	● Document format modified