MQ-LINK User Manual
V2.51

Contents

1. Change History

Version	Approved Date	Description
V2.51	2023/05/16	1. English version establish 2. Remove "5.9.3 How to enable and disable FreeRun and observe variable values ".
V2.5	2022/12/15	1. Adjusted the original chapter "5.11.2 SQ7617 EEPROM programming" to "6.5 EEPROM Writer", and added support for Flash ROM of some MQ MCU. 2. Adjust the original "5.11.1 Preparations before SQ7617EEPROM Program" chapter to "8.5 SQ7617 Preparation before EEPROM Program".
V2.4	2022/01/24	1. Added "5.10 SQ76 Series Flash Program Settings". 2. Added "5.11 SQ7617 EEPROM Writer Settings".
V2.3	2021/04/07	1. Added "6.4 Opening the Download Only function menu" instructions.
V 2.2	2020/3/2	1. Revise "CH2 Hardware Introduction", and remove the i87-IDE hardware introduction from the content. 2. Amended "3.1.1 Installation and Use Precautions ". 3. Add "5.9.3 How to enable and disable FreeRun and observe variable values ". 4. "8. Other Supplementary Notes" removes the precautions for using the MQ8603 multi-programming function.
V 2.1	2019/12/17	1 "5.9.1 Variable Observation Function" updated Figure 5-21 and Figure 5-22; added example 5. Use the mouse to select the variable to be observed in the IDE editing window, 6. Manually input the variable name to be observed 2. Add "5.9.2 Allow some variable name formats"
V2.0	2019/11/5	1. Change the file name from " i87-IDE User Manual " to " MQ-LINK User Manual ".

V1.9	2018/05/04	1. Remove the description of the Watch Window in "4.3.7 Configuration Menu". 2. Modify item 7. Observation variable description in " 4.3.3 View Menu". 3. "5.7.1 Chip Items and Corresponding Debug Target Chip Table" updated.
V1.8	2018/03/02	1. "4.3.7 Configuration Menu" added Watch Window instructions. 2. "5.6 Pin Settings" update. 3. "5.7.1 Chip Items and Corresponding Debug Target Chip Table" updated. 4. Added "6. iMQ i87-IDE programming function".

2. MQ-Link Hardware Introduction

MQ-Link is an IDE dedicated to MCU products (see Figure 2.1). MCU firmware on the host computer can directly download into the Flash on MCU through MQ-Link, and then execute debug functions, this will greatly increase the efficiency of user firmware development. There are two slots on the MQ-Link. The left port of the MQ-Link in Figure 2.1 is a Mini USB port, which can be connected to a PC (host computer); the right port is connected to the Evaluation Board(EV board) or the user's PCBA (Please see Figure 2-2). For the pin position definition of the slot, please refer to the label on the Q-Link (reserve 6 pins including VCC, OCDIO, OCDCK, RESET, GND, and MODE pin position).

When Connect to the user PCBA, pay attention to the following:

1. Because the MQ-Link default power supply is 5V (can be modified to 3.3V), so when connecting, do not plug in an external power supply on the PCB board.
2. Note that the OCDIO, OCDCK, RESET, and MODE pins of the IC must be confirmed to be floating before connecting to the MQ-Link, to avoid the circuit on the PCB from affecting the MQ-Link debugging signal.

Two LEDs on the front of the MQ-Link. When connected to the host computer, the green LED will start to flash, indicating that it is working (Active); if the red LED is on or the green LED is always on\not on, it means an error (Error). Note that since the label is located above the LED, the light from the LED will show through the label.

For the host computer software used by MQ-Link, please refer to the following chapters of this manual.

Figure 2.1 Front view of MQ-Link

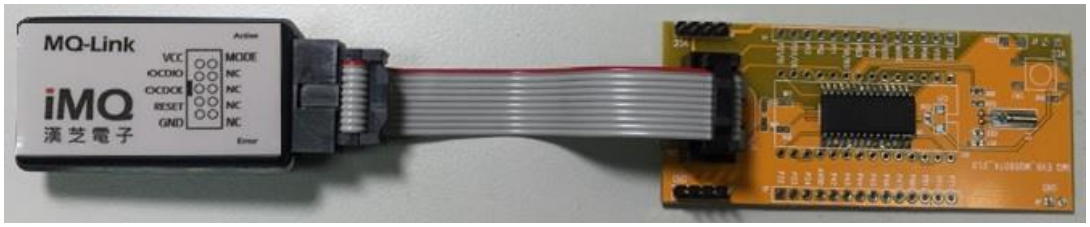


Figure 2.2 Connection between MQ-Link and EV Board (there is a product chip on the EV Board)

3. iMQ i87-IDE Software Installation and MQ-Link Firmware Update

3.1 iMQ i87-IDE Software Installation

3.1.1 Precautions

MQ-Link uses the iMQ i87-IDE host computer software; please refer to the instructions in the following chapters. Also, please note the following:

1. When using IDE software for program debugging, do not disable MCU "External Reset" pin, otherwise, IDE will not work normally.

2. When writing the assembled program, please be sure to use the correct assembly format that conforms to the i87 standard; if the format is not in compliance, the IDE may not work properly.

3. One should notice that the I/O port of MQ-LINK does not have a pull-down resistor. If you want to use the iMQ MQ8S series MCU(which has a pull-down resistor), whether each MCU has the function of a pull-down resistor (refer to MCU product specifications), please also connect an external pull-down resistor on the Target board.

Note 1): The MQ-LINK external pull-up resistor only provides the pull-up function, and there is no "Enable/ Disable" pull-up resistor function.

3.1.2 Install IDE Software Tool

As shown in Figure 3.1, move the cursor to the IDE software installation file (the file name usually contains version and date, for example, the V3.87 version released on November 25, 2022, the installation file name is "iMQ_IDE_v3.87(20221125).exe") click mouse right button. Please note that if there is a message indicating that you need to execute as a system administrator, please use "Run as administrator" to install; if not, click "Open" to install.

Note): As shown in Figure 3.1, move the cursor to the location of the "iMQ_IDE_V3.87(20221125).exe" installation file, and click the mouse right button. Please note that if there is a message indicating that you need to execute as a system administrator, please use "Run as administrator" to install; if not, click "Open" to install.

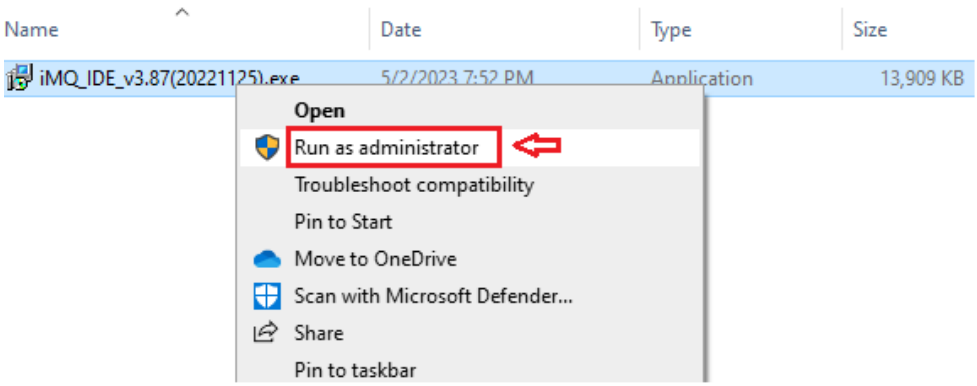


Figure 3.1

Windows system will pop up an alarm dialog, and click the “Yes” button, as shown in Figure 3.2.

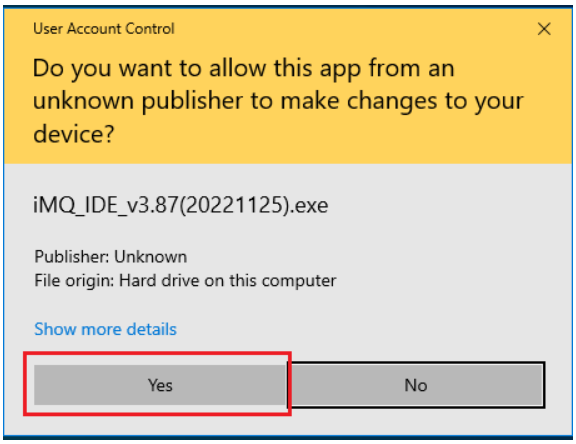


Figure 3.2

After that, select the file path to be installed and the name to be displayed in the Start menu, as shown in Figure 3.3 and Figure 3.4.

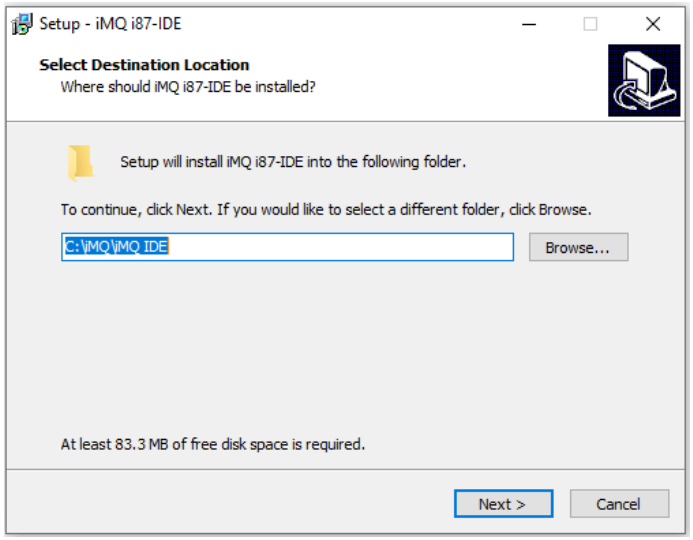


Figure 3.3

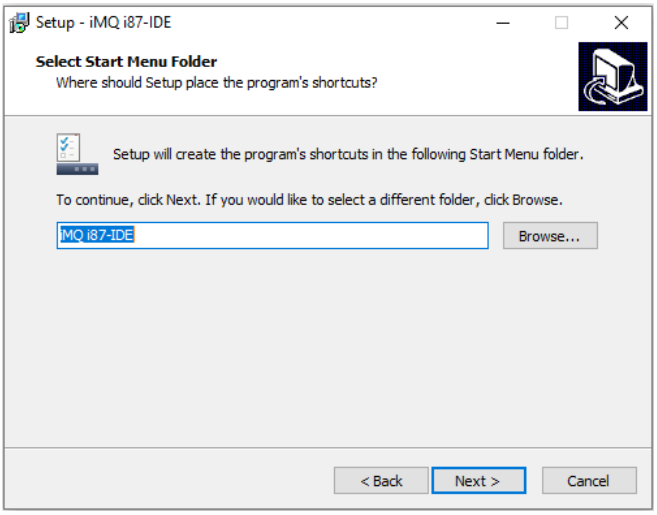


Figure 3.4

Move the mouse cursor to "Create a desktop icon" and check it. After the installation is complete, the IDE execution shortcut will automatically generate on the desktop of the computer. Then move the cursor to the "Next" position and press the left mouse button. Figure 3.5.

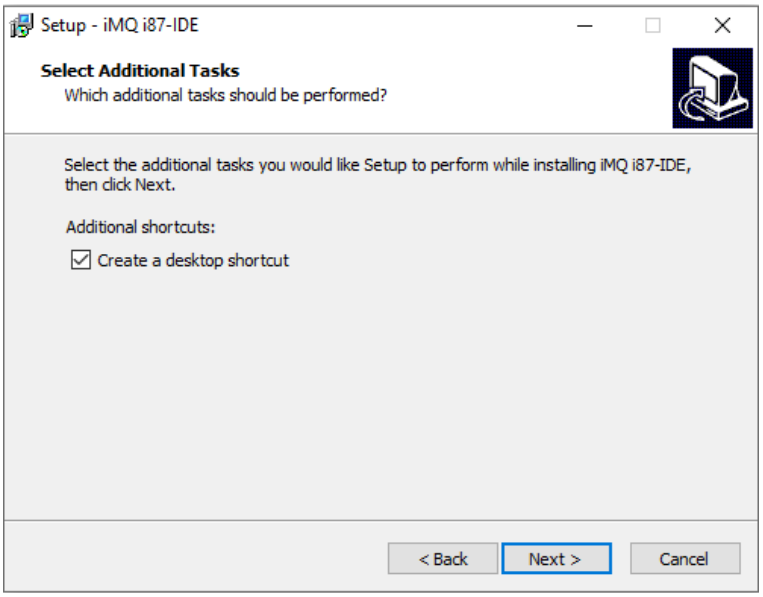


Figure 3.5

Move the cursor to the "Install" position, and press the left mouse button to start the installation, as shown in Figure 3.6.

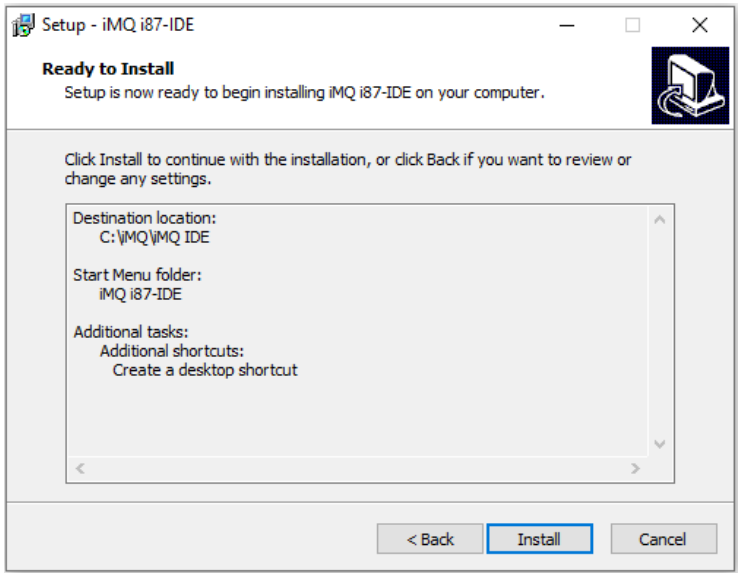


Figure 3.6

After the installation is complete, click "Finish".

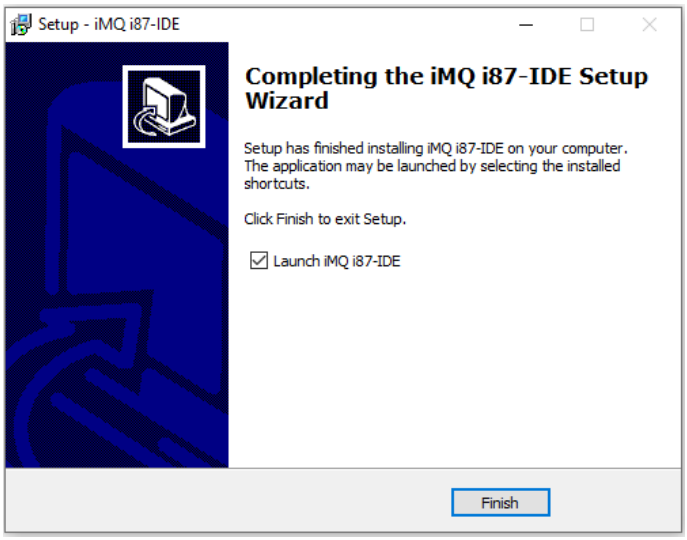


Figure 3.7

After that, a shortcut icon of the IDE will appear on the desktop of the computer, as shown in Figure 3.8.

Figure 3.8

Establish relation between IDE project files and IDE software

In the IDE project file, double-click .pmt file, and follow the dialog step to establish an open .pmt file and iMQ I87-IDE software relation, as shown in Figure 3.9.

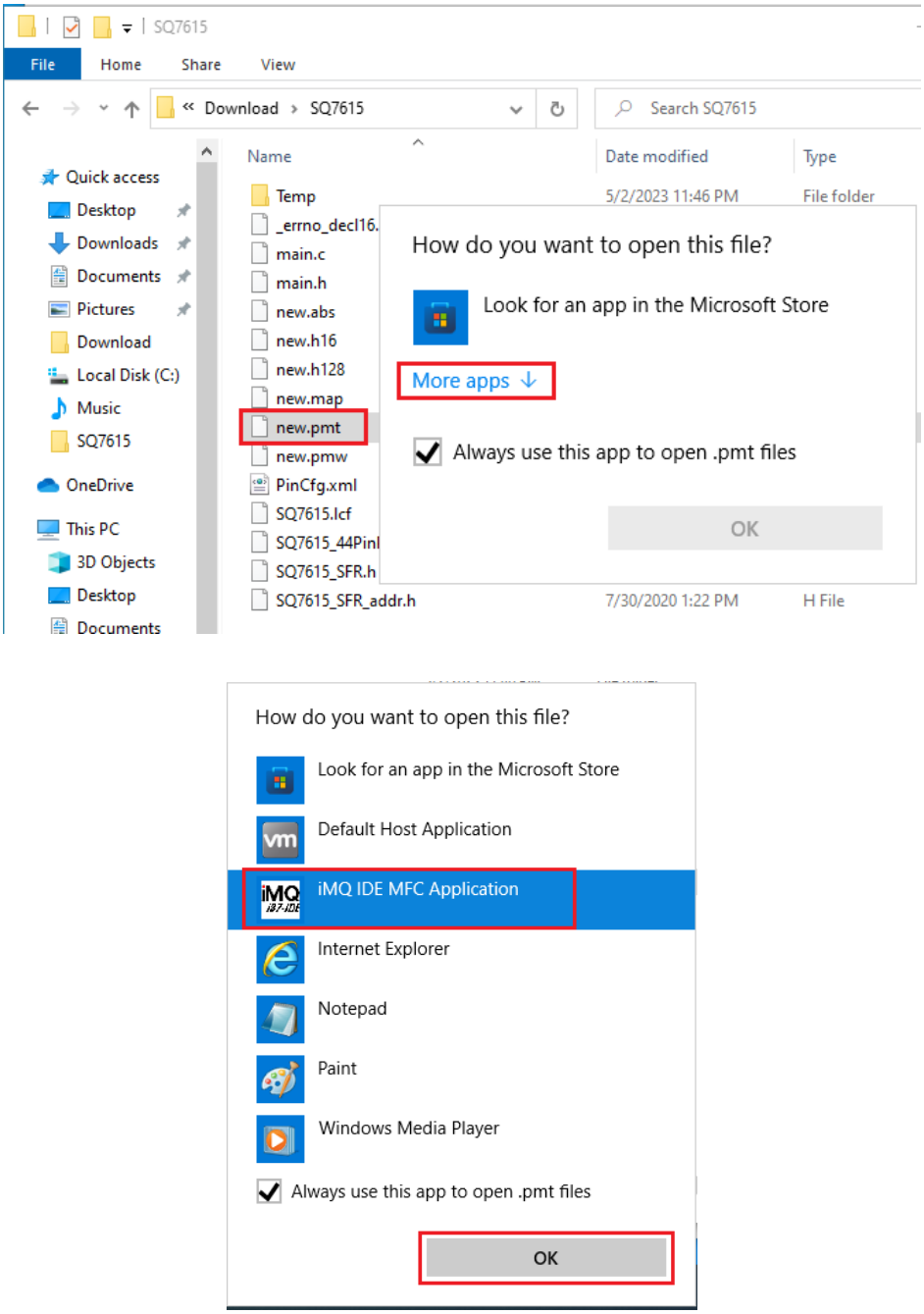


Figure 3.9

3.1.3 Uninstall IDE Software Tool

Open "Settings" in the Start menu, as shown in Figure 3.10. After the screen shown in Figure 3.11 appears, click the "Apps" button.

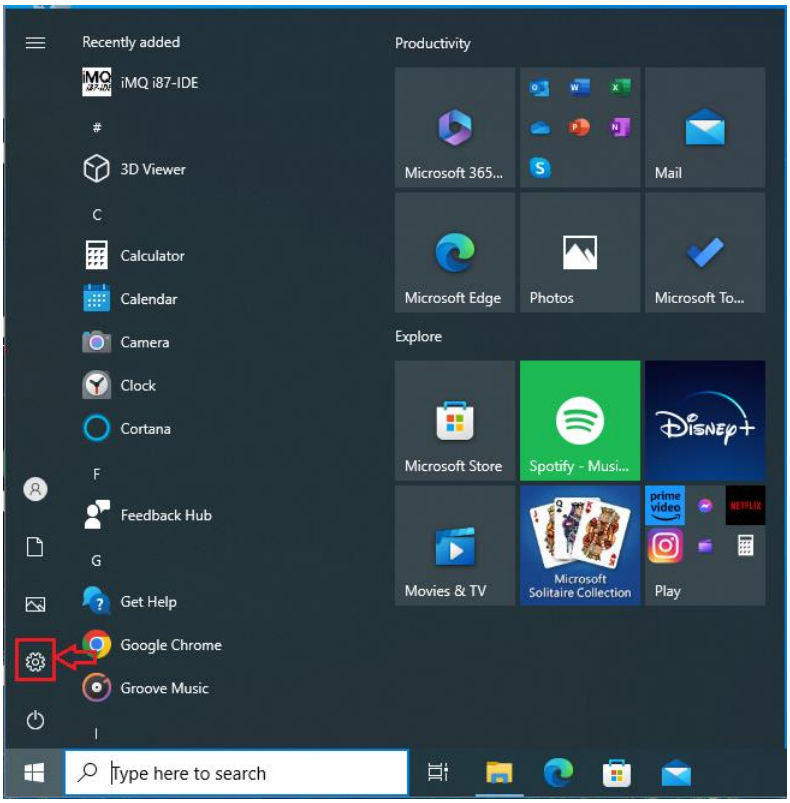


Figure 3.10

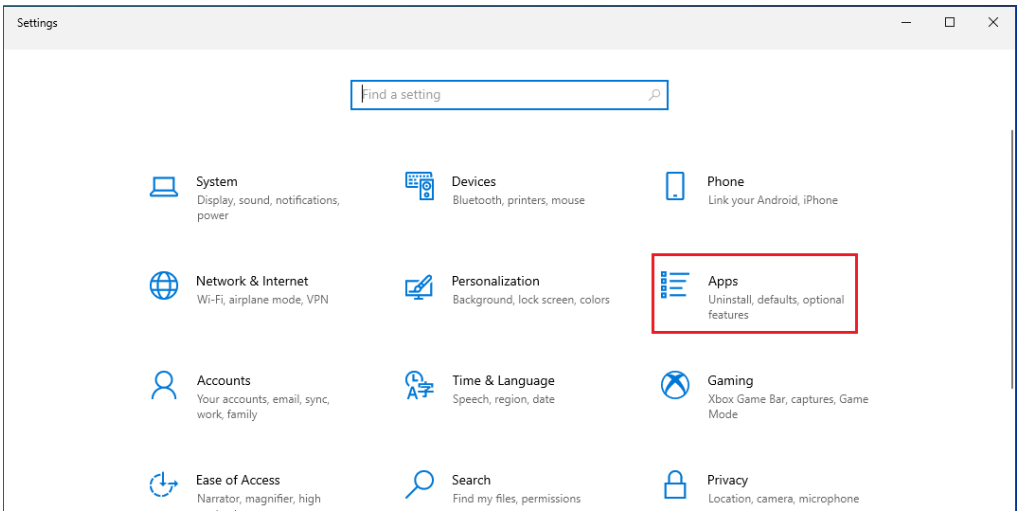


Figure 3.11

After the program list in Figure 3.12 appears, click the installer of iMQ i87-IDE (the name

usually includes the version, such as iMQ i87-IDE version V3.87), and press "Uninstall" to confirm the removal of the program. (Figure 3.13)

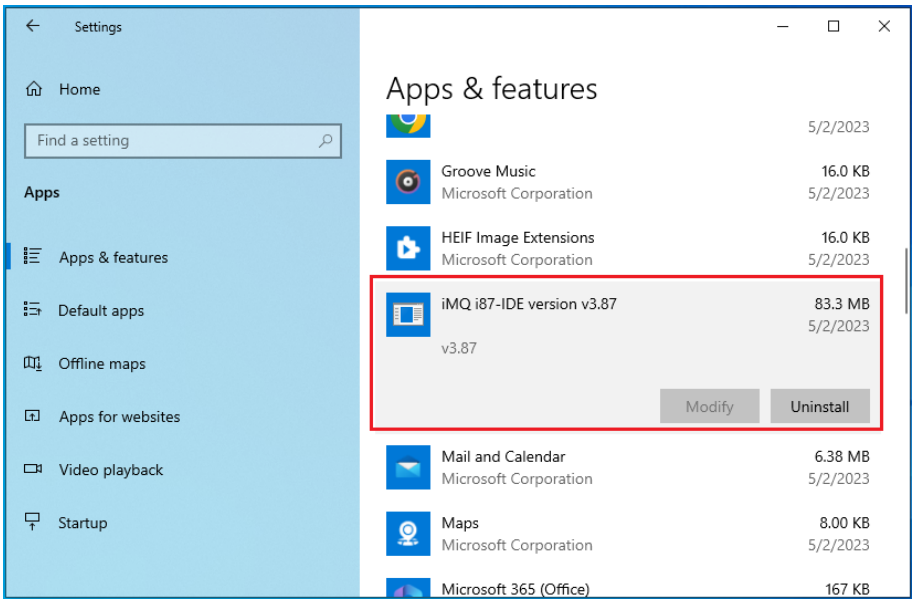
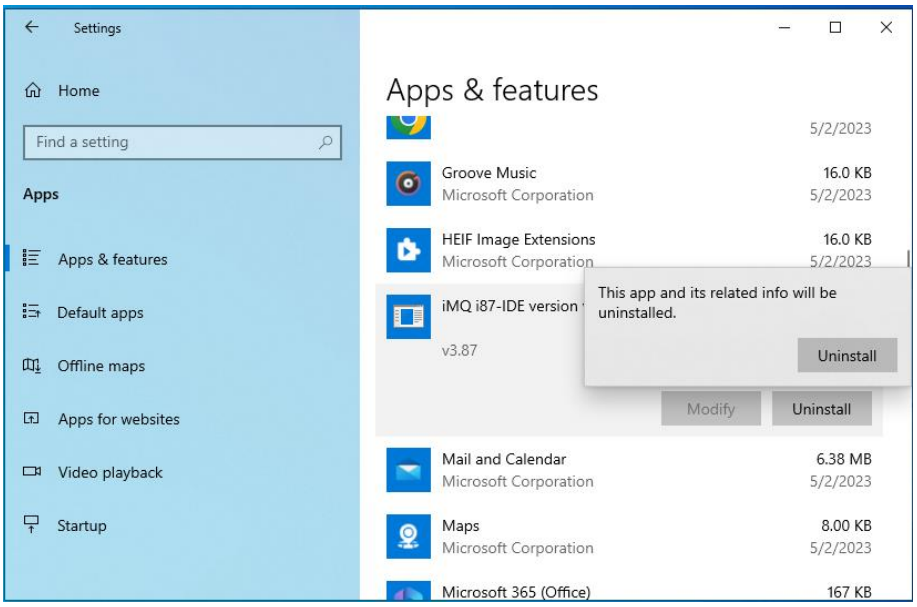
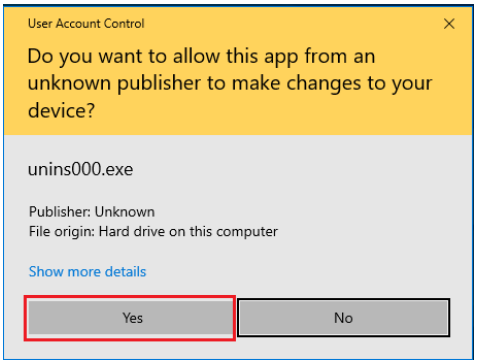


Figure 3.12





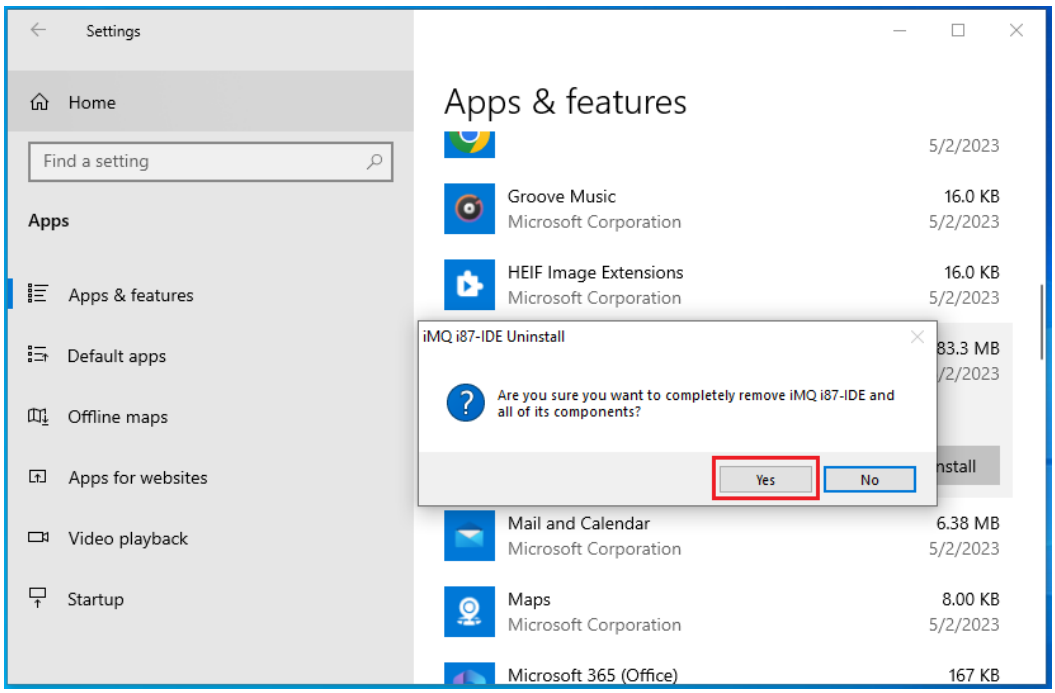


Figure 3.13

After the program is removed, the words "iMQ i87-IDE was successfully removed from your computer" will appear in the window. After pressing the "OK" button, the installed i87-IDE software will be removed, as shown in Figure 3.14.

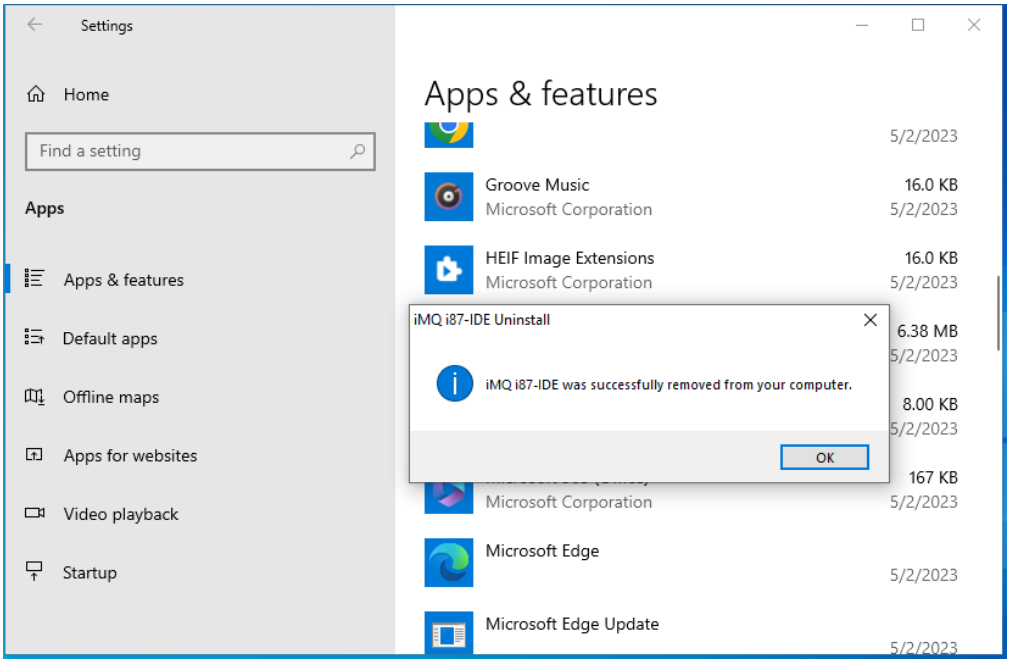


Figure 3.14

Then go to the "Program Files" subdirectory of the IDE installed disk(for example “C” disk), move the cursor to the subdirectory "iMQ" (Figure 3.15), click the right button of the mouse, and then Figure 3.16 will appear. Move the cursor to "Delete" and press the left button to complete the removal of all iMQ i87-IDE items.

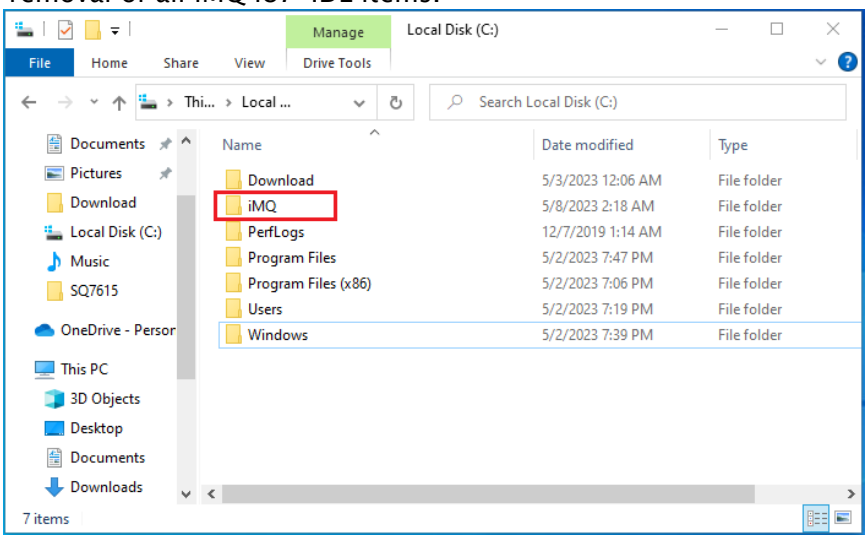


Figure 3.15

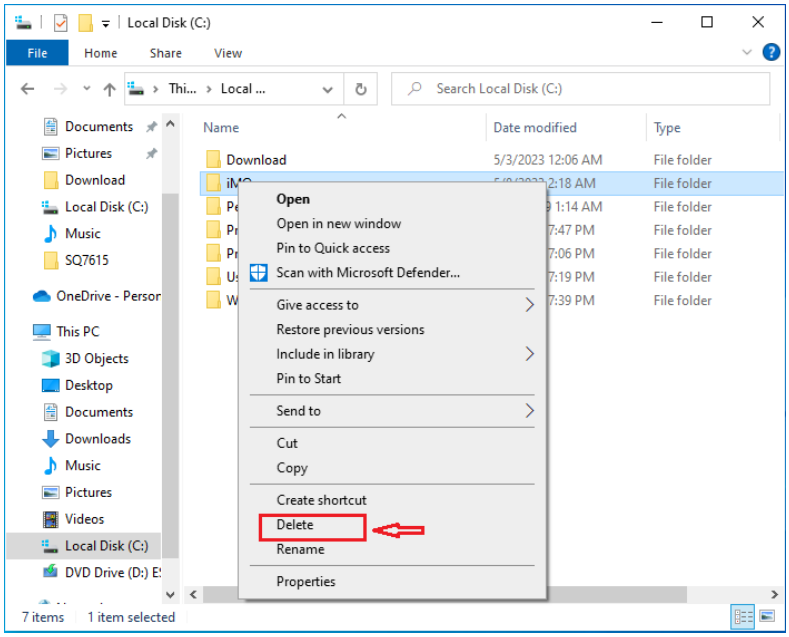


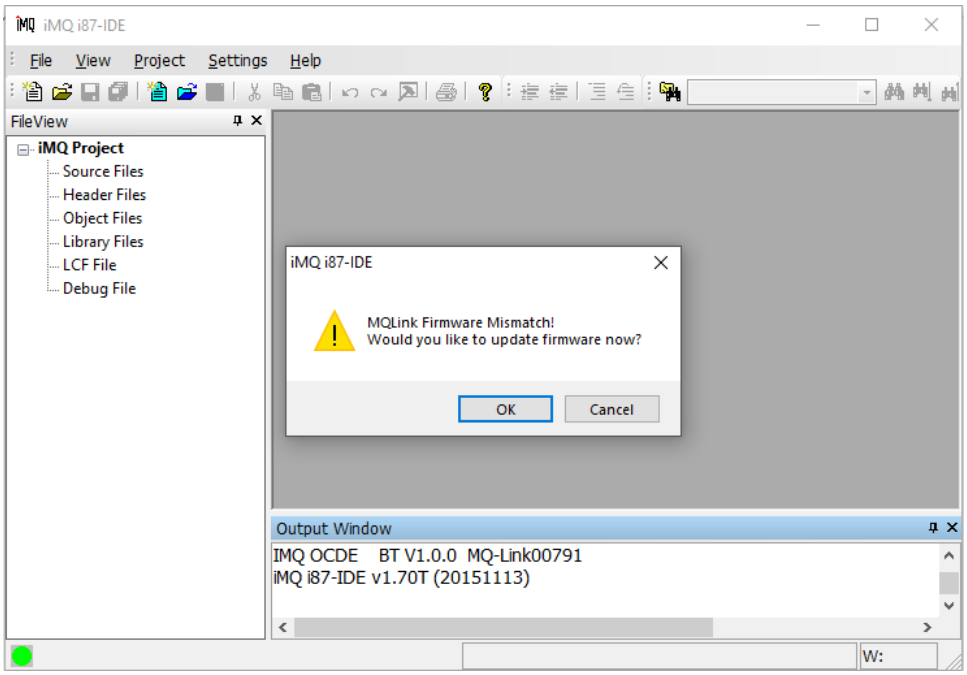
Figure 3.16

3.2 MQ-Link Firmware Update

3.2.1 Automatic Firmware Update

The firmware update is to adapt for new MCU support and to have the best performance and features.

After the MQ-Link hardware is connected to the IDE software of the host computer, it will check the MQ-Link firmware version; if the firmware version does not match with the IDE software it will automatically update the firmware.



3.2.2 Manual Firmware Update

Open the IDE software, and connect the MQ-Link hardware to the host computer through a USB cable. The dot on the lower left turns green to indicate that MQ-Link is connected and the following operations can be performed, In the IDE menu bar “settings”, select “Firmware Upgrade…”, will pop up the Firmware upgrade dialog, as shown in Figure 3.17.

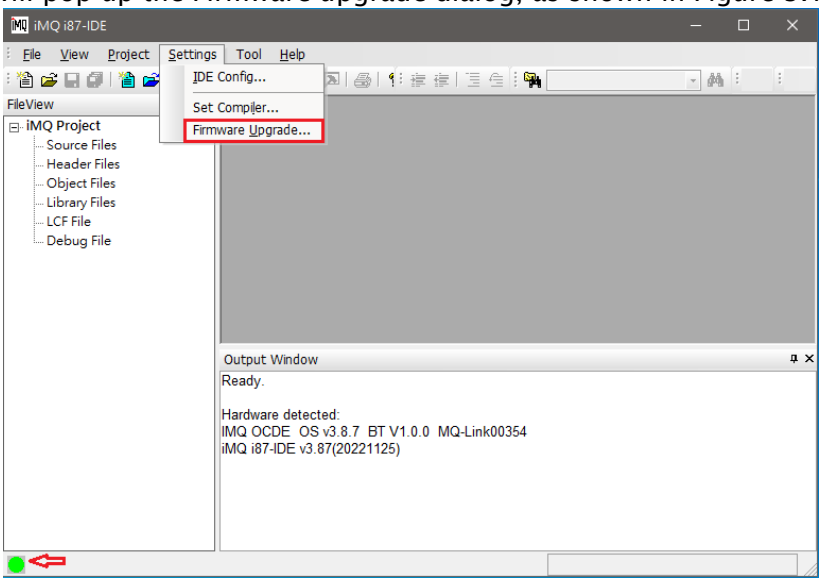


Figure 3.17 The dot on the lower left is green, indicating that the connection is successful

Click the “Browse(B)” button in the dialog box to select the firmware file to be upgraded, as shown in Figure 3.18 and Figure 3.19.

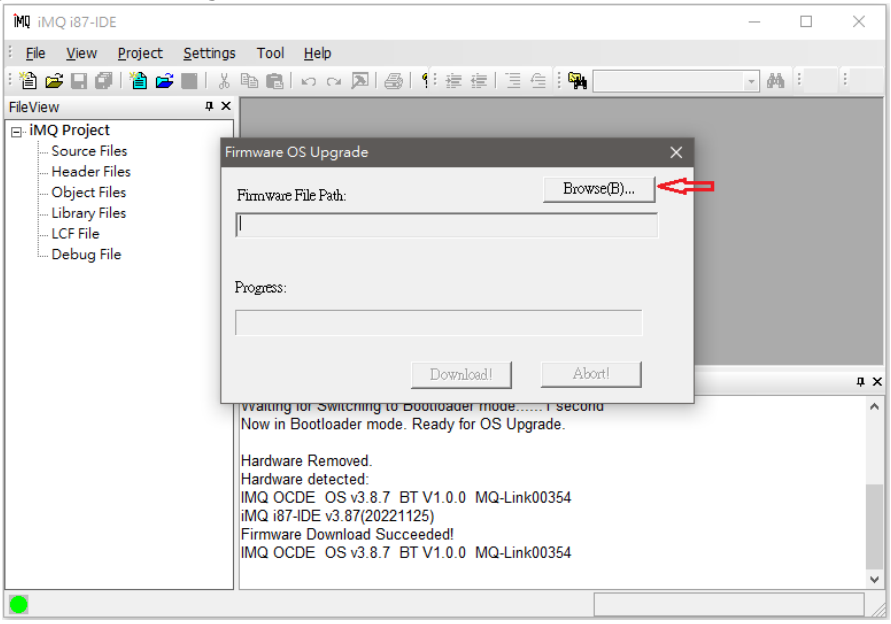


Figure 3.18 i87-IDE firmware upgrade dialog box

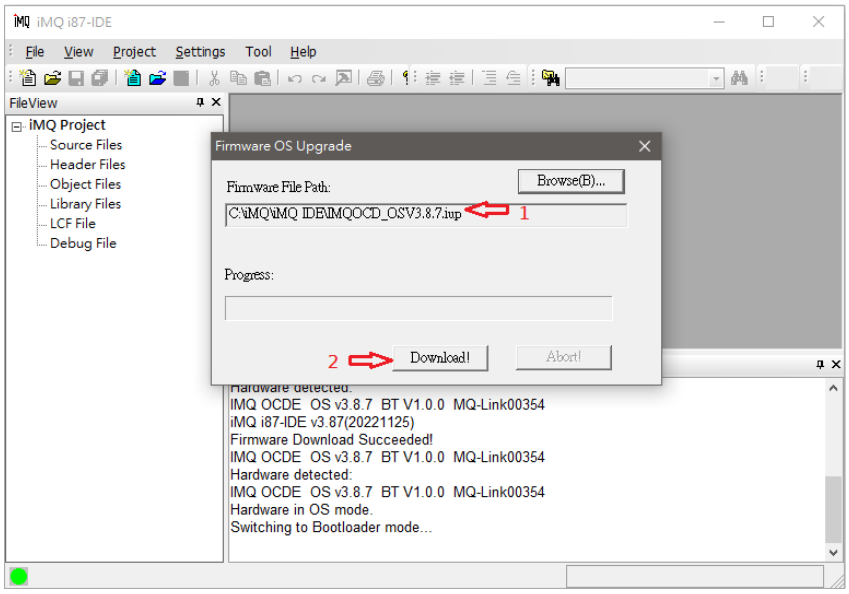


Figure 3.19 The firmware upgrade dialog, confirms that the firmware file to be upgraded has been selected

Click the Download icon indicated by the red arrow 2 in the dialog box in Figure 3.19. After clicking, a dialogue as shown in Figure 3.20 will appear. Please click OK directly to continue the firmware upgrade process.

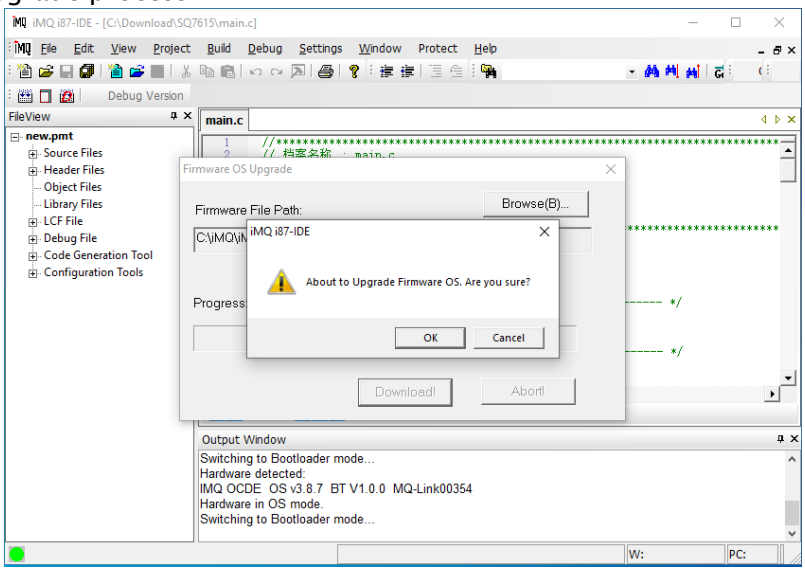


Figure 3.20 Dialog box for confirming firmware file download

After clicking OK, there will be a few seconds to wait for the upgrade. Afterward, if the dialog box information shown in Figure 3.21 appears, it means that the firmware upgrade has been completed. Otherwise, please repeat the above operations until the firmware version is upgraded successfully.

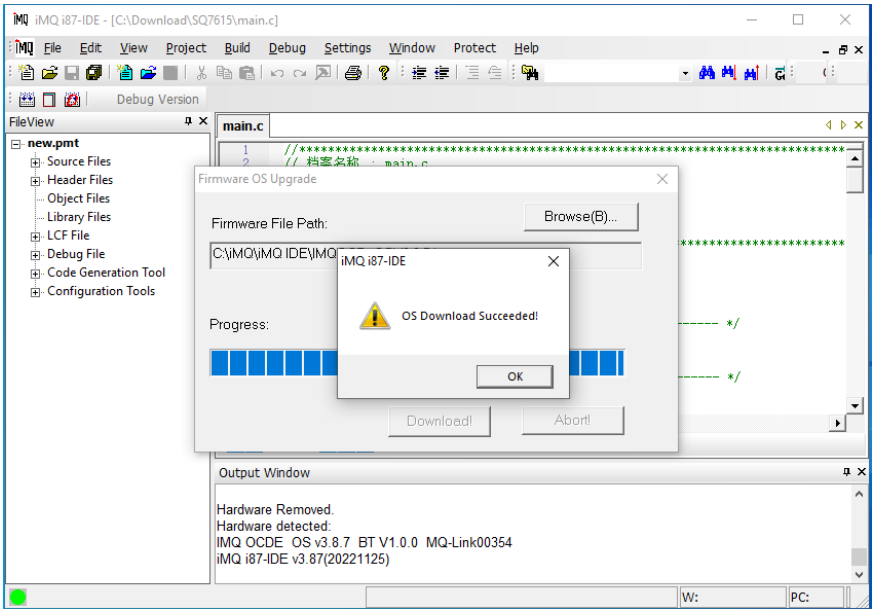


Figure 3.21 i87-IDE firmware upgrade successful dialog box

4. iMQ i87-IDE Software Function Description

4.1 General Description

iMQ i87-IDE is an integrated development environment that can complete functions such as code editing, compiling, and debugging during the development and debugging of various MCU application systems. Currently supported hardware is MQ68xx, MQ69x, and SQ series ICs. The software has two modes: Compile Mode and Debug Mode. The menus will be different in different modes. In the debug mode, the source code can also be edited, modified, and compiled.

When the software is running in the editing mode, the user uses the editing and compiling functions of the software, such as opening or creating new files, copying, pasting, cutting, setting editing bookmarks, etc., without hardware support. When you click the "Debug" button or the shortcut key F5, the software runs in the debug mode. After that, the user can perform various debugging actions, such as running, single step, and variable observation and modification.

4.2 Software Interface

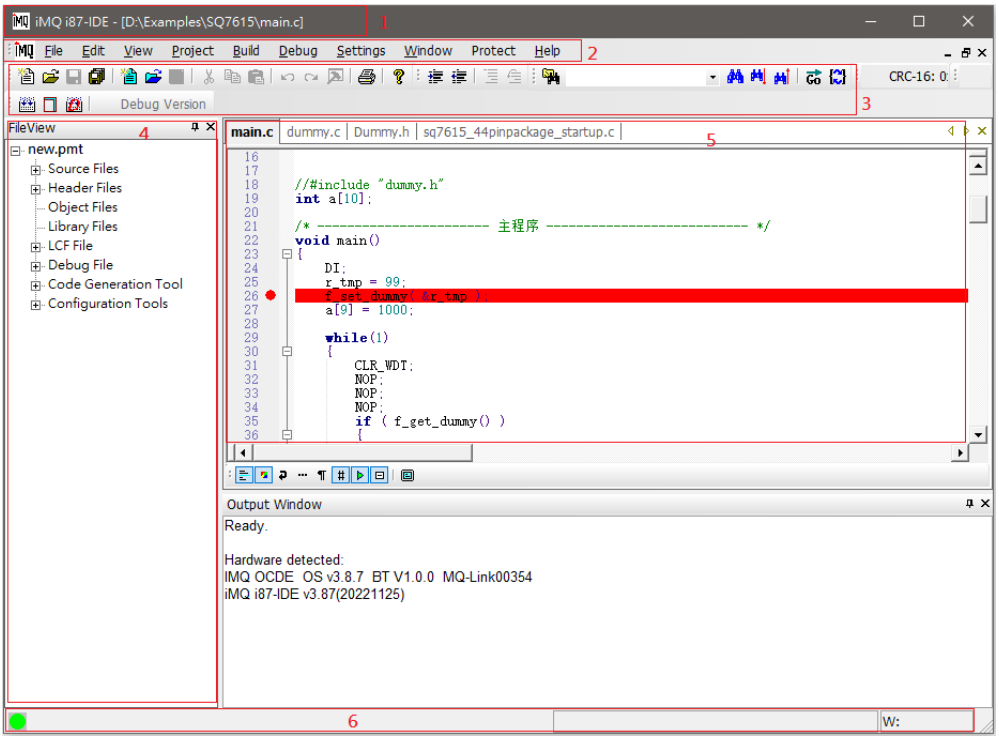


Figure 4.1 IDE software interface

The operation of the IDE interface is shown in Figure 4.1, which is divided into 6 parts, as follows. Different parts have different mouse right-click menus.

1. **Project Title** - The project title shows the path and name of the currently open file.

2. **Main Menu** - From the main menu, you can view information about the software, or complete all operations.

3. **Shortcut Toolbar** - The shortcut toolbar contains icons for some commonly used operations to speed up operations without the need to execute menu commands; the functions are the same as the commands in the menu.

4. **Project Window** - The Project (Project) window contains sub-windows such as Project, General Registers, Special Registers, and Stack.

5. **Source program window** - The source program window displays the source program interface, and the user can edit and modify the source program, and set breakpoints, and bookmarks in this window.

6. **Status bar** - The status bar displays the running status of the current tool and displays common information such as PC and running time.

4.3 Main Menu

In different modes, the main menu will be different, but the functions are the same. The main menu is divided into **File, Edit, View, Project, Build, Debug, Settings, Window, help, and** other submenus.

4.3.1 File Menu

The "File" menu includes: **New, Open, Close, Save, Save As, Print, Print Setup, Recently Files, Recently Projects,** and **Exit**. Among them, the recently files (projects) are used to reserve the names of the four recently opened files (or projects), and the user can directly open them here to speed up the operation.

4.3.2 Edit Menu

The "Edit menu includes: **Undo, Redo, Cut, Copy, Paste, Delete, Select All, Comment Selection, UnComment Selection, Find, Find Next, Find Previous, Replace, Go To Line, Find in Files, Match Brace, Select To Brace, Read Only, Toggle Bookmark, Next Bookmark, Previous Bookmark, Clear All Bookmarks.** Some of the special features are as follows:

1. Comment Selection - After you select a text block by dragging the mouse in the editor, execute this function from the main menu or right-click the menu, and you can set each line where the text block is located as a comment. The symbols used for comments depend on the language. The .asm file uses ";" as the comment symbol, and the C or C++ source file uses "//" as the comment symbol.

2. UnComment Selection - It is used to remove the comment mark of the text block that has been set as a comment. If the selected text block is an ordinary text, this command will not have any action.

3. Go To Line - each source code line has a corresponding line number, you can use this function to quickly jump to the code line you specify.

4. Match Brace - if brackets are used in the source code, such as {}, [ ] in c language, if there are many contents in the brackets, or there are many nesting layers of brackets, it is usually more difficult to locate the other side of the brackets Trouble, using the bracket matching function can help you quickly locate the other side of the bracket. The method of use is to position the editing cursor on one side of the bracket and then execute this function, and the editing cursor will immediately jump to the other side of the bracket.

5. Select bracket content - you can quickly select the content between a pair of brackets: put the editing cursor on one side of the brackets, and then execute this function, you can see that the content between the brackets has been selected, Then you can copy, cut, delete, etc.

6. Read-only - It is a toggle key, which can quickly set or cancel the read-only attribute of the file editing window. Read-only attributes are displayed in the status bar. Check here if you find that you cannot modify the file.

7. Advanced Bookmarks - The following are shortcuts for setting, canceling, and navigating to bookmarks 0 to 9:

Set/Cancel Bookmark n: Press "Ctrl+n"
Navigate to bookmark n: press "Alt + n"
(where n = 0 ~ 9)

4.3.3 View Menu

The view menu includes **Full Screen, Project, status bar, toolbars, Output, Debug Window,** and **Display Line Numbers**. It is used to call up various windows for observation.

1. Project Bar- Create a new project, open a project, and save a project in sequence.

2. Status Bar- Displays basic status information such as operating mode, program counter, running time, etc.

3. Toolbars - including **Standard bar, Find bar,** and **Text bar**. (Figure 4.2)
Standard bar: open file, create new file, save file, save all.
Find bar: search content input area, search, find next, find previous. Enter the content you want to find in the search input area and press "Enter". If a match is found, the cursor will be positioned at the first match; Find Next "F3" and Find Previous "Shift + F3" are used to find and position the next or previous match respectively. If you need more search conditions, you can click the Find button to set the search settings and execute the search in the search dialog box.

Figure 4.2 Standard Toolbar, Project Toolbar, Edit Toolbar, and Find Toolbar

Note): You can also search directly in the file, specify the search content, extension rules, and search directory to find files containing the specified content in the specified directory.

3. Debug Window -

ComReg window- In the Common register window, the hexadecimal value is displayed according to the address, see Figure 4.3. Among them, the gray background color is the register whose value has not changed, and the red background color is the register whose value has changed. It is divided into 5 levels, and the change of the register within 5 steps can be recorded. The earlier the change, the lighter the register color. When the mouse clicks on a register, its background color will be displayed as white. At the same time, the address and binary value of the register will be displayed at the bottom of the window. The

value of the register can be directly modified.

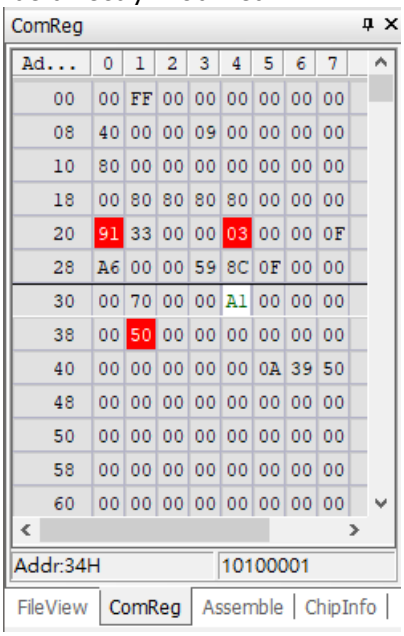


Figure 4.3 Common registers

SpecialReg window- in the special register window, the hexadecimal value of the register content is displayed after the special register name, when the mouse moves over the value, the binary result will be displayed, which is more convenient For observations, see Figure 4.4. At the same time, the address and a brief description of the register will be displayed at the bottom of the window. You can directly modify the value in the special register window, or click "..." in the value box to view the function description of the corresponding bit of the special register, and then configure the register according to the function.

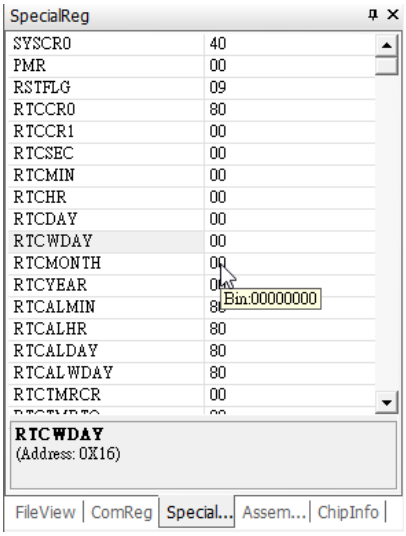


Figure 4.4 Special Registers

Dump Memory window- used to view chip memory data. Enter the specified address, and display a maximum of 64 pieces of data including the specified address.

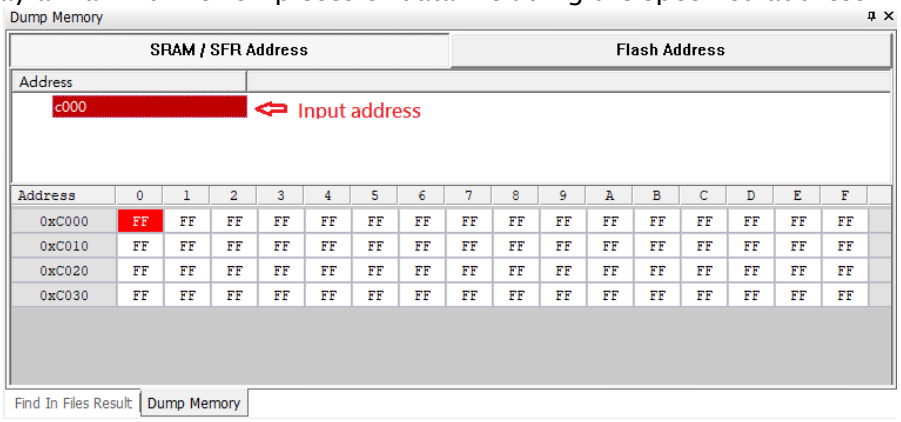


Figure 4.5 Memory observation window

Assembly window - The Assembly window displays the reverse generation of assembly code from machine code. This window provides the function of directly modifying assembly instructions or machine instructions, see Figure 4.6.

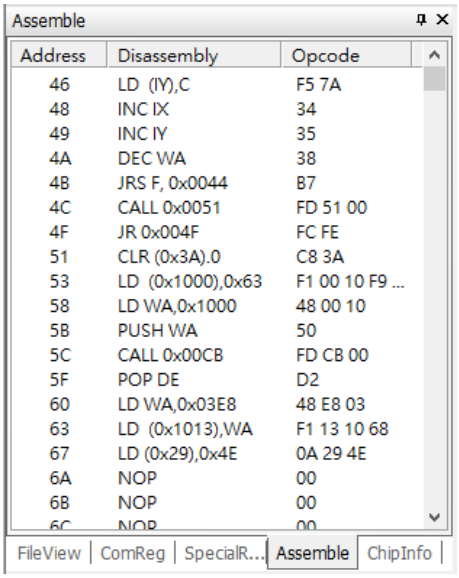


Figure 4.6 Assembly window

Watch window- The Watch window can observe various variables, including user-defined variables, special register variables, program labels, etc. If you use a high-level language such as C language, you can also observe complex data such as arrays and structures Structure, see Figure 4.7.

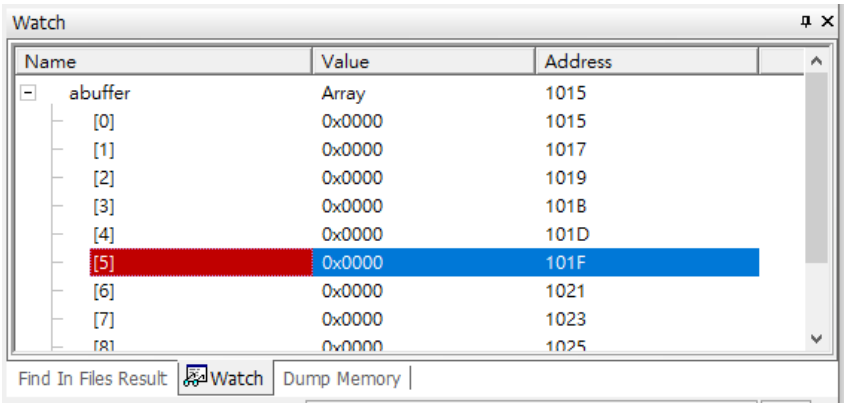


Figure 4.7 Observation variables

Note): Quick Start: In debug mode, double-click the blank line under Name, enter the variable name to be observed, and after confirmation, you can see the corresponding address and value in the name and value fields, unless the entered variable name is invalid or Not within the scope, you can change the observation properties of the variable through the right-click menu, such as switching the number system represented and setting all variable properties, etc., as shown in Figure 4.8.

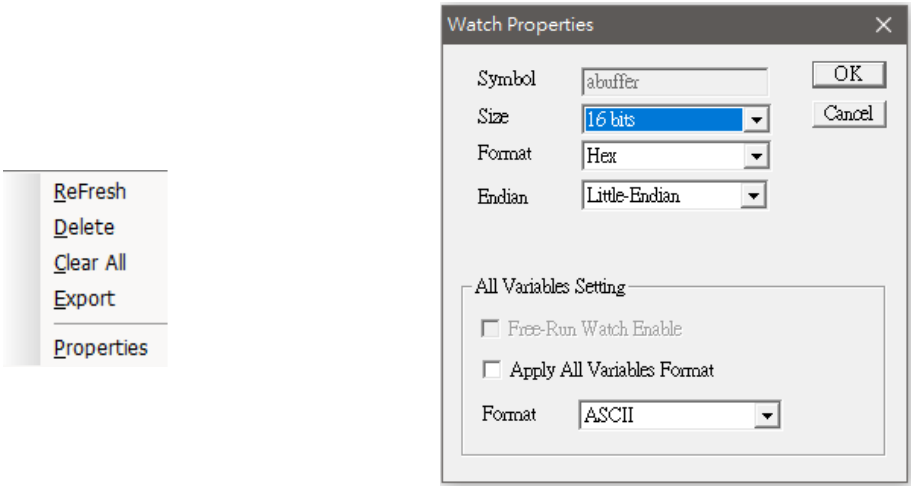


Figure 4.8 Watch Window property setting

Symbol window - The symbol window saves the debugging information of the symbol table. The symbols in the symbol table can be directly added to the watch window through the right-click menu, and the observed variable is based on the symbol table. If you want to confirm a variable name or symbol If the name can be observed, you can open the symbol window to check whether the symbol exists in the symbol window, if not, then it cannot be observed, see Figure 4.9.

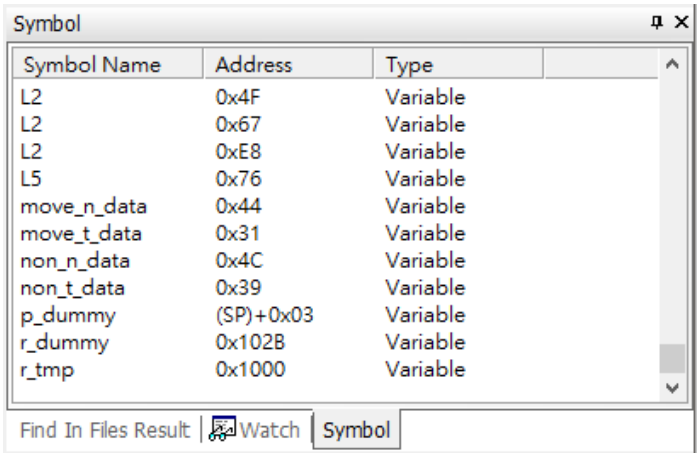


Figure 4.9 Symbol window

4.3.4 Project Menu

The project menu includes **new project, open project, close project, save project, save project as, project properties,** and **pin setting.** Project property is used to open the project property dialog box, which is used to modify project properties, such as changing compile options of assembler and C compiler, etc. For the specific set of project properties, see "5.4 Project Property Settings".

4.3.5 Build and Debug Menu

The Build menu includes **Release Ver., Debug ver.(**compile debug version, compile release version**), link, View link Map,** and **Download Only.**

The Debug menu includes **Start debug, End debug, Go, C Step Over, Run to Cursor, A Step In, A Auto Step, Pause, Reset,** and **Trace**. These functions, as the name suggests, are the same as general debug functions. Here are a few important commands.

1. C Step Over - Execute one C program line downwards, and jump into the subroutine to perform a single step when encountering a subroutine.

2. A Step Over - Execute one assembly program line downwards, and jump into the subroutine to perform a single step when encountering a subroutine.

3. A Auto Step - Automatically execute assembly single-step operations continuously. Using this function, you can intuitively observe the running track of the program, and the interval between single steps is 1 second.

4. Reset - Perform a reset operation, not a power-on reset, similar to the reset caused by the RST reset pin, but only modify the PC pointer and SFR.

4.3.6 Window Menu

The Window menu includes **New Window, Close All Documents, Tile Horizontally, Tile Vertically, Split Window, Window File List**, and **ReMap All Windows.** It mainly completes some operations such as tiling and splitting of the source program editing window.

4.3.7 Settings Menu

The Setting menu includes **IDE Config..., Set Compile...,** and **Firmware Upgrade.**

IDE Config...: IDE parameter settings include general settings (General), color and output window font settings (Colors and Fonts), grammar settings (Languages), etc. The following are some commonly used pages:

1. **General** - general settings (as shown in Figure 4.10)
   Maximize window - Whether to maximize the window when starting the software.
   Open Last Project - Whether to open the last project when starting the software.
   System language - software interface language, you can choose English or Chinese
   Check File Modification-Checks if the file has been modified by an external program.
   Mini Toolbar - Whether to display the mini editing toolbar.
   Copy on Mouse Selection - Whether to automatically copy the selected text.
   Save All Documents Without Prompting - Automatically save all documents on exit.

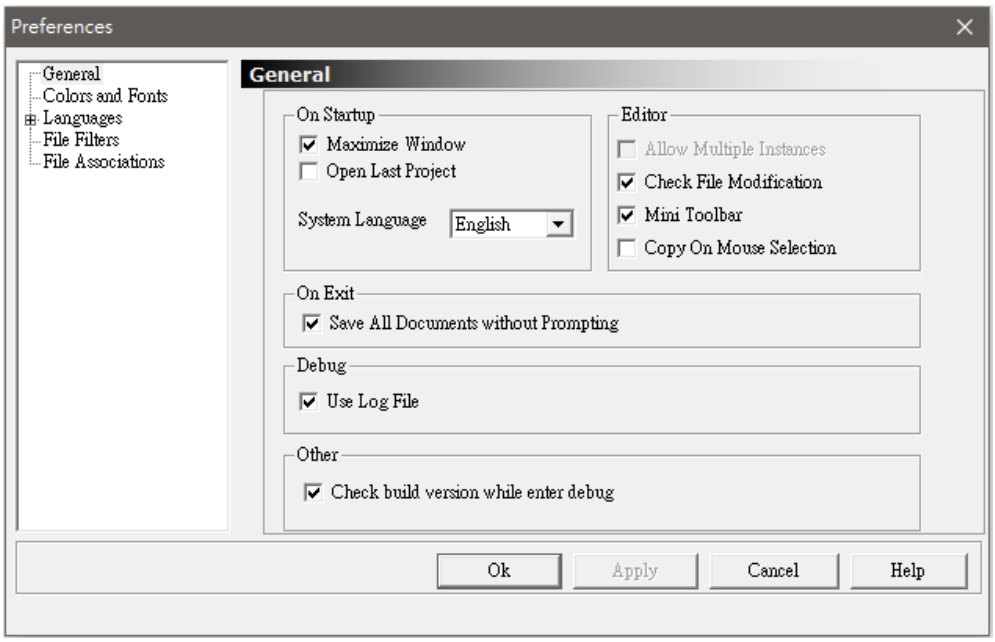


Figure 4.10 General settings (General)

2. Colors and Fonts - color and output window font settings (as shown in Figure 4.11)

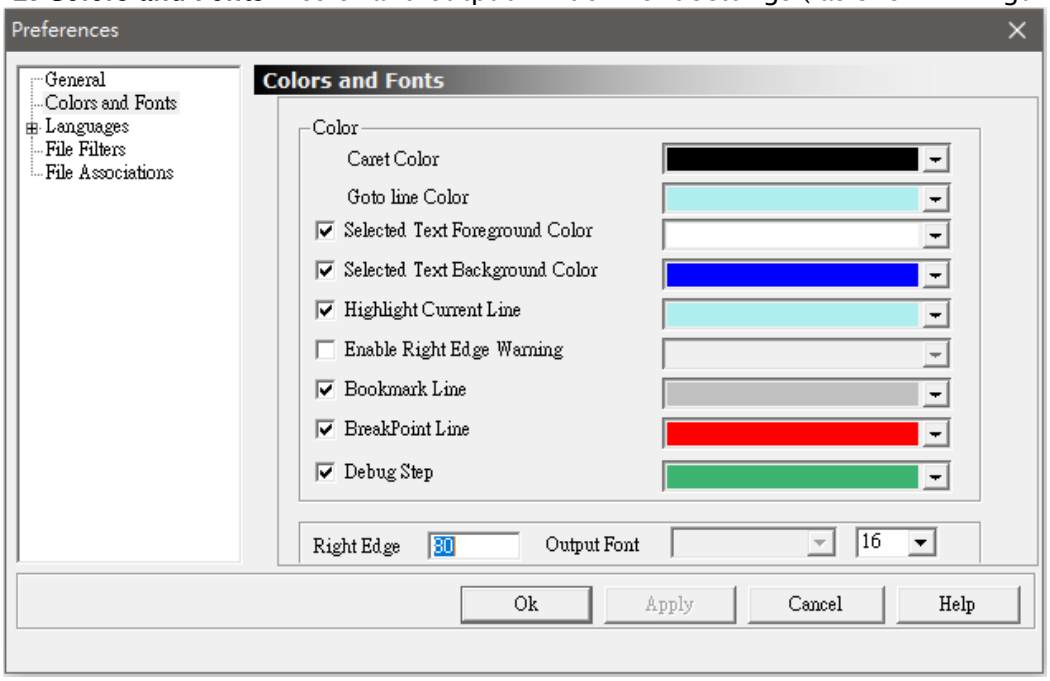


Figure 4.11 Colors and Output Window Font Settings (Colors and Fonts)

Caret Color - Sets the caret color.
Goto Line Color - Sets the color of the jump target line using the goto positioning dialog.
Selected Text Foreground Color - Sets the selected text foreground color.
Selected Text Background Color - Sets the selected text background color.
Highlight Current Line - Sets the color used for the current line highlight.
Enable Right Edge Warning - Sets the right margin warning line color.
Bookmark Line - Sets the line color of the bookmark.
Breakpoint Line - Sets the color of the line where the breakpoint is located.
Debug Step - Sets the PC line color when stepping.
Right Edge - Specifies the number of characters which the right margin warning line.
Output Font - Sets the font and font size used by the output window.

3. Languages - grammar settings, used to modify and add grammar files, it is not recommended for users to modify. Under it, there are three syntax setting dialog boxes Default, Cpp, and Assembler. Users can modify the options, and customize the editing settings such as the syntax keywords used in a certain language. The following is a preliminary interface introduction in Cpp (C/C++) language:

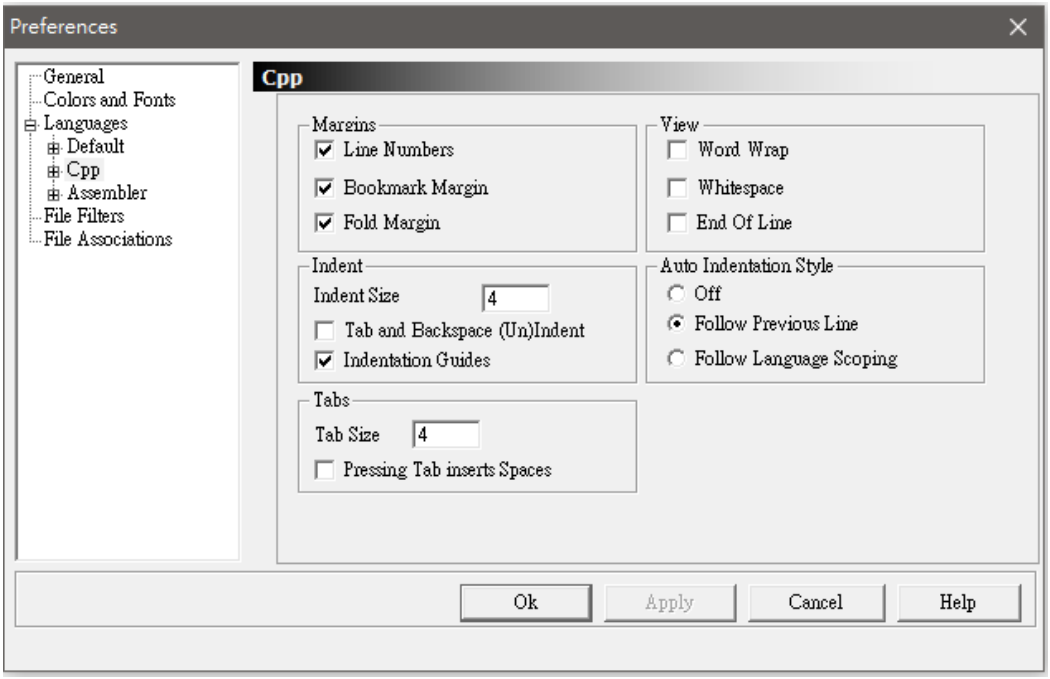


Figure 4.12 Language Settings (Languages) – the primary interface, taking C/C++ as an example

Line Numbers – Whether to display line number margins
Bookmark Margin – Whether to display the bookmark margin
Fold Margin – Whether to display the folding margin
Word Wrap – word wrapping
Whitespace – whether to display spaces
End Of Line – Whether to display line endings
Indent — Indent amount
Auto Indentation Style – Automatic indentation style
Tabs – TAB width setting

The following is an introduction to advanced dialog boxes in Assembler language, including dialog boxes such as Code, Color, Font, and Keywords. First is the Code dialog.

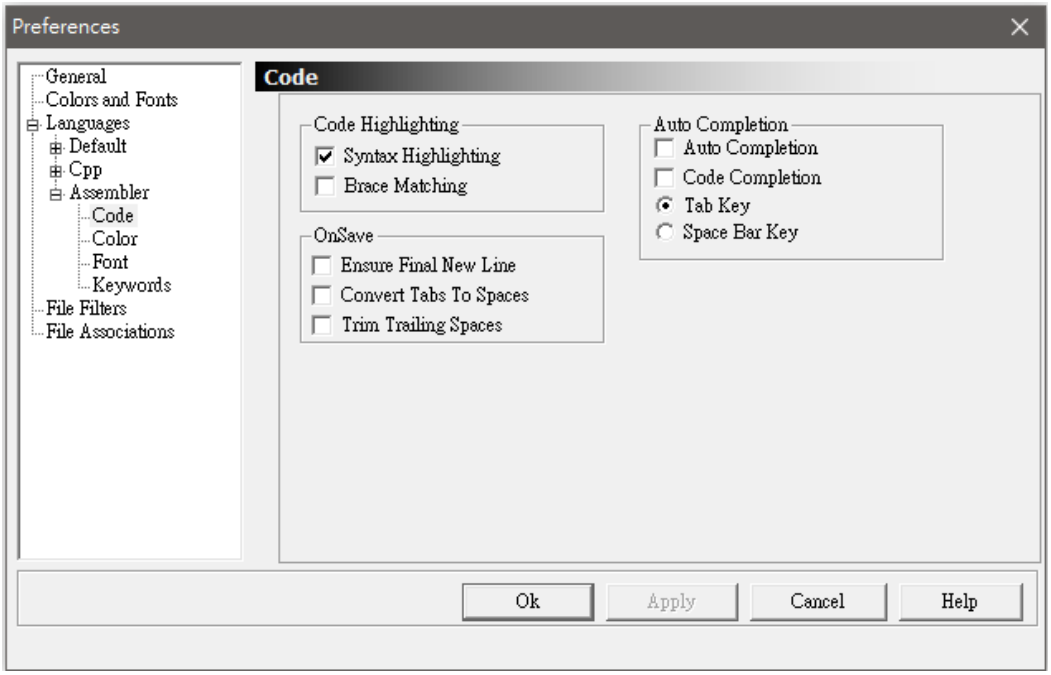


Figure 4.13 Language Settings (Languages) – Code dialog box, taking Assembler as an example

Syntax Highlighting – Whether to use the syntax highlighting function
Brace Matching – Whether to use the bracket matching function
Fold Ensure Final New Line – Ensure that there is a blank line at the end of the text
Convert Tabs To Spaces – convert TAB to spaces when saving
Trim trailing spaces – Trim trailing spaces when saving
Auto completion – not implemented

The Color dialog box is used to set the color of various components of the source code (such as spaces, comment blocks, single-line comments, numbers, keywords, strings, etc.) of the corresponding language. You can set foreground and background colors. "Reset All Colors" is used to restore the default settings. (See Figure 4.14)

The Font dialog box is used to set the font for various components of the source code (such as spaces, comment blocks, single-line comments, numbers, keywords, strings, etc.) of the corresponding language. "Apply to all styles" applies the current settings to all source code components. "Bold" is bold; "Italic" is italic; "Underline" is underlined. (See Figure 4.15)

As for the Keywords dialog box, if you need to add new keywords, you can add them here. It should be noted that lowercase letters should be used and spaces should be used to separate them. (See Figure 4.16)

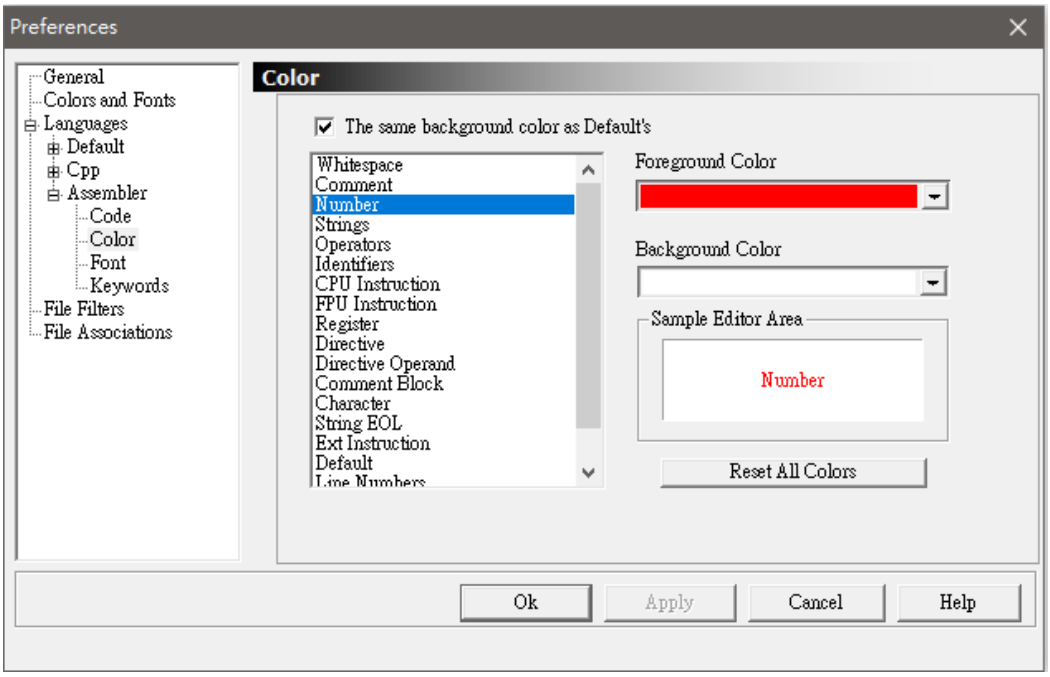


Figure 4.14 Language Settings (Languages) – Color dialog box, taking Assembler as an example

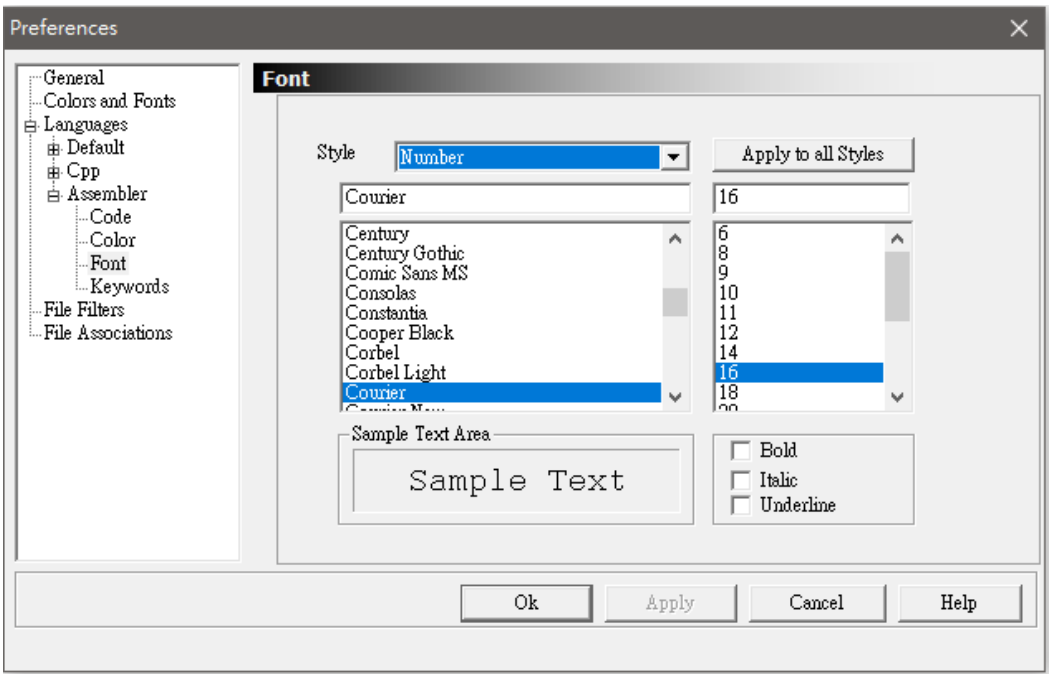


Figure 4.15 Language Settings (Languages) – Font dialog box, taking Assembler as an example

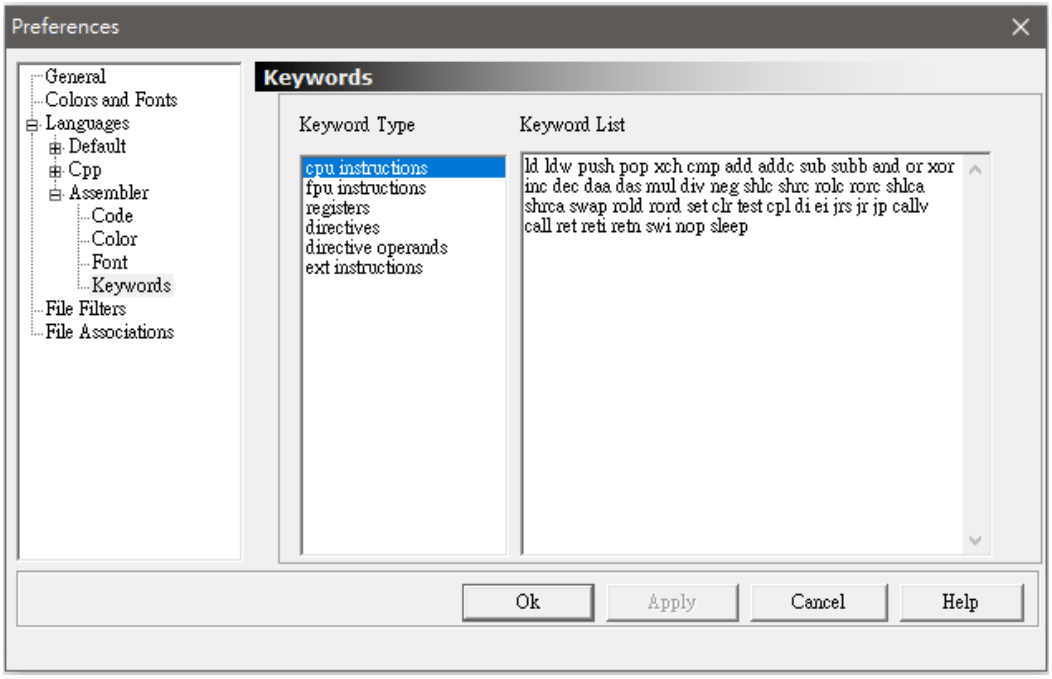


Figure 4.16 Languages – Keywords dialog box, taking Assembler as an example

4.3.8 Help Menu

The help menu includes **About IMQ-IDE**, which is mainly used to display the iMQ i87-IDE software version and copyright information.

4.4 Background on Debugging

The general flow of debugging using the IDE is shown in Figure 4.17.

The files that can be used for debugging by the IDE can only be object files, not source code. The object file is generated by compiling the source code through a compiler (including an assembler), and contains machine code and various debugging information, such as variable names and their addresses, associations between machine code and source code, tags, and so on.

iMQ i87-IDE currently supports abs files and hex files.

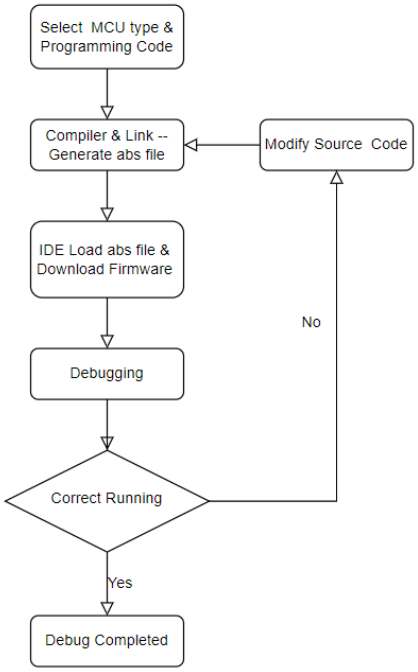


Figure 4.17 Debugging process

4.5 IDE Project Management

The user needs to set some necessary information during the debugging process. The IDE software manages the above information through the concept of project management and stores it in the project properties.

Before using the IDE to debug, the project must be established first. There are three types of projects:

- Empty Project
- C Template Project
- Asm Template Project

Selecting to create "C Template Project" and "Asm Template Project" will generate "startup", "main", "headfile", "interrupt table" and "LCF" files corresponding to the MCU as shown in Figure 4.18.

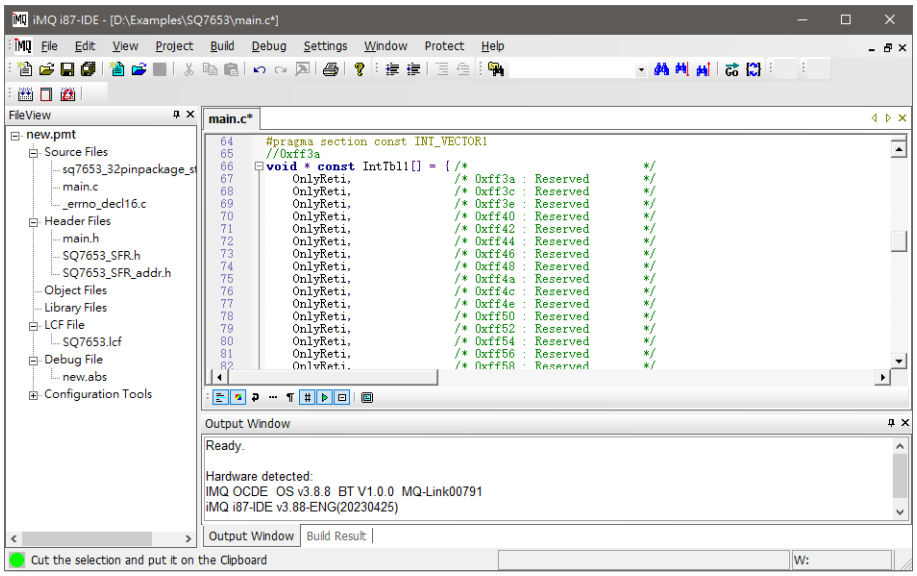


Figure 4.18

If you choose to create an "Empty Project", only "LCF" files will be generated, as shown in Figure 4.19.

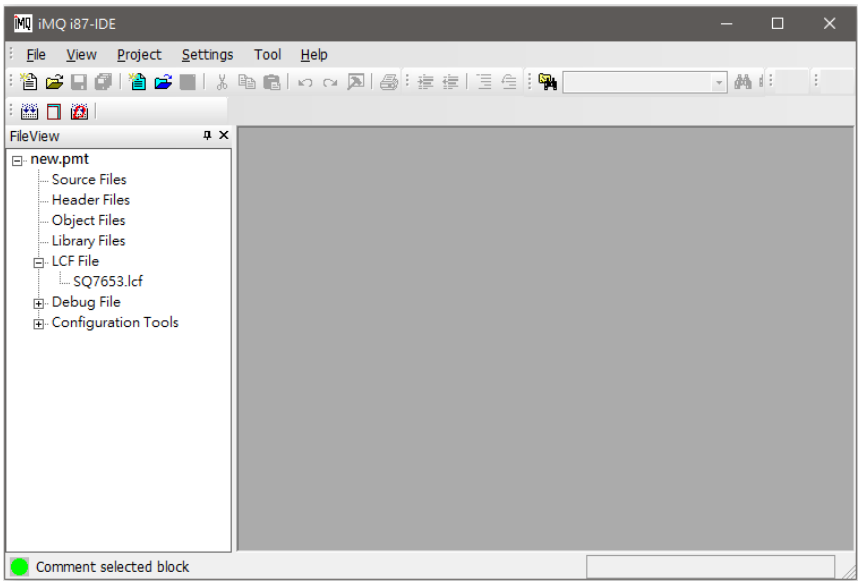


Figure 4.19

A complete project file generally includes the following:

File information: source program file or debug code file name and path

Compiler settings: the type of compiler used and its configuration

Debug settings: chip type, etc.

Window and other information: observe the setting of variables and breakpoints, the adjustment of user windows, … etc. When user exits, the information will be fully saved and reproduced the next time the project is opened.

5. iMQ i87-IDE Debugger

This chapter discusses how to use the IMQ OCDE debugger for development.

5.1 New Project

This IDE software provides a new project wizard for users to use.

5.1.1 Select Project Type

First, click the "Project" menu on the main menu and select "New Project". The pop-up window is as follows, select "Empty Project", "C Template Project" or "Asm Template Project". For the differences between the three projects, please refer to "4.5 iMQ i87-IDE Project".

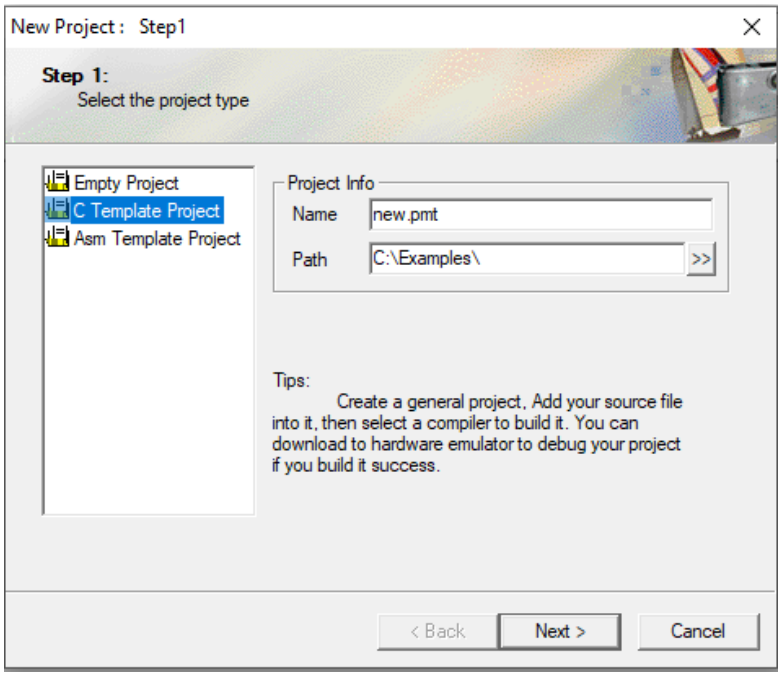


Figure 5.1 New project window

Fill in the project file name and path in turn. The project of iMQ i87-IDE is a file with a suffix of *.pmt to distinguish it from general project files (*.prj). After the setting is complete, click "Next" to continue.

5.1.2 Select Chip Manufacturer and Chip

The following window appears. In the drop-down menu of "Manufacturer", select the chip manufacturer, i.e. iMQ. Select the chip series in the "Series" on the right, i.e. i87. And Then select the chip model in the box below.

The user should pay attention to whether the chip model selected for the project can match the current hardware. After the setting is complete, click "Next" to continue (as shown in Figure 5.2).

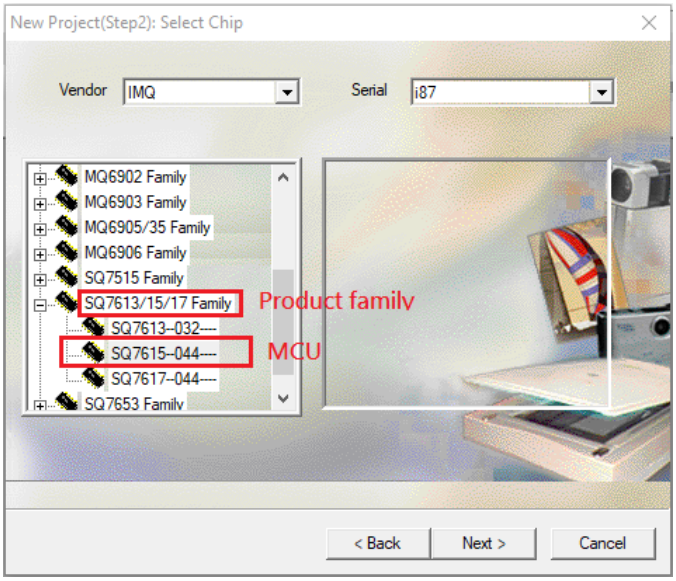


Figure 5.2 Select chip model

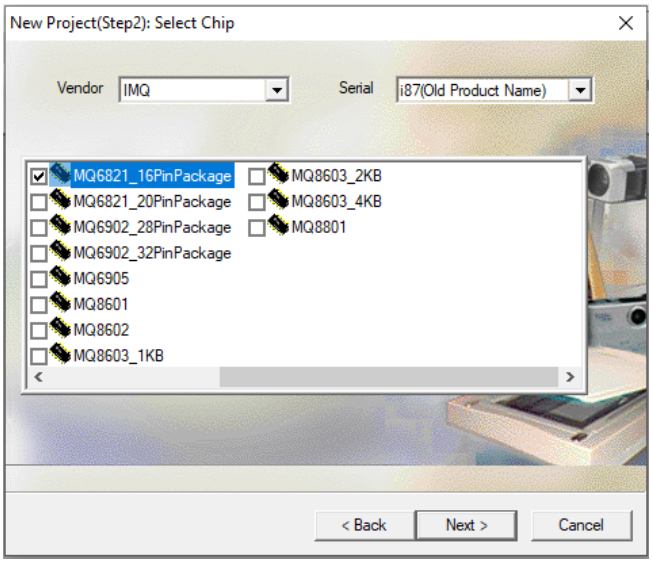


Figure 5.3 Select the chip model (compatible with the old product model)

5.1.3 Selecting Compiler

Use the drop-down menu to select a compiler. As shown in Figure 5.4.

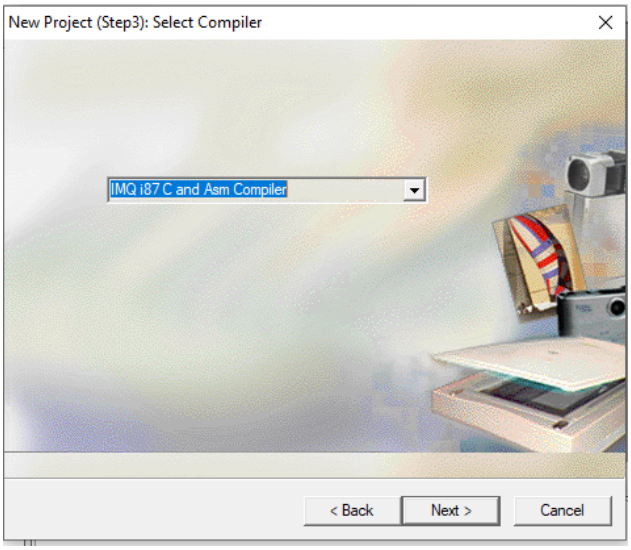


Figure 5.4 Select the compiler

The compiler type currently supported by iMQ i87-IDE is “iMQ i87 C and Asm compiler”. After the setting is complete, click "Next" to continue.

5.1.4 Complete New Project

At this time, a project summary window will pop up, which displays the information about the current project (as shown in Figure 5.5). Click Finish to finish and save the newly created project. You can see that the IDE has added some default files to your project. The file name corresponds to the one selected when building the project.

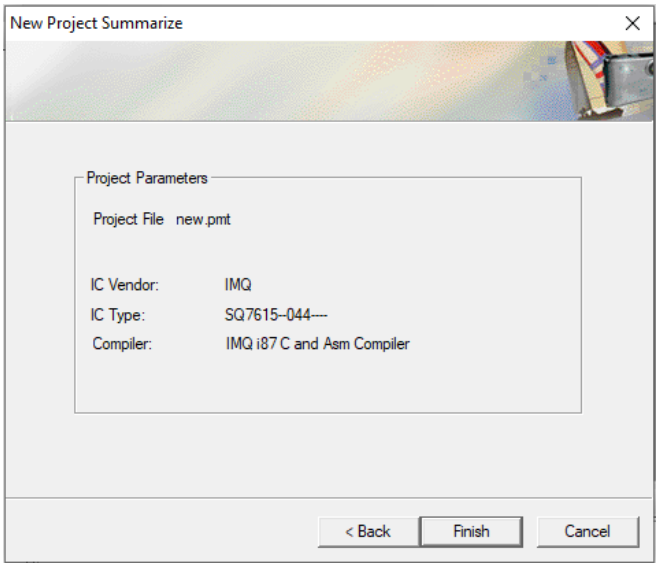


Figure 5.5 Complete the new project

5.2 Add Source Program

Click the right mouse button on the "Source Files" (source file) item on the left side of the main screen of the IDE, and a menu will pop up. Users can select "Add File". If there is no source code, you can select "New File", save it, and add it to the project.
In the "Header Files" of the project, you can also create or add header files. The header files added by the user should be placed in the same directory as other source files. Please refer to Figure 5.6. If necessary, you can add *.rel object module files under "Obj Files", or add library files that have been obtained by users in other ways under "Library Files" below.

Once complete, you can double-click a source file name to open the file editing window. When editing, we can use bookmarks to mark the places of focus, especially when the program is large, it is very convenient to jump to the bookmark.
IDE provides two bookmark functions: a common bookmark and an advanced bookmark. Use Ctrl+F2 to set or delete ordinary bookmarks on the line where the cursor is located, and use a single F2 to quickly jump between multiple labels. The advanced bookmark is a bookmark with a number, and the jump is in place in one step: use Ctrl+number (0~9) to set or delete (you can also execute the corresponding menu item repeatedly), and use Alt+number (0~9) to locate.

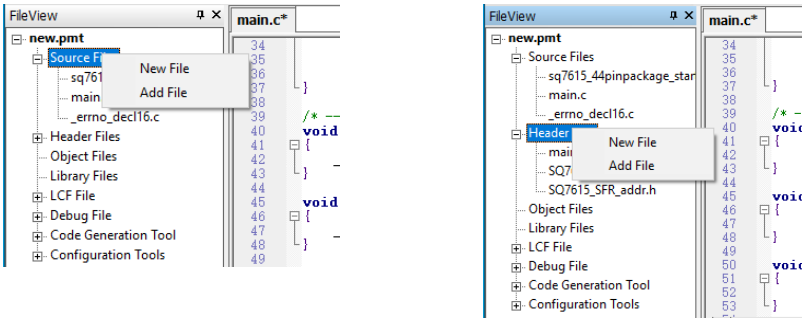


Figure 5.6 Options for creating or adding source programs (left) and header files (right)

5.3 Set Compiler

Since the embedded compiler is used, there is no need to set the compiler path. If you want to explicitly specify the compiler path, you must ensure that the corresponding compiler exists under the path. In the menu bar select "Settings" -> "Set Compiler...", and the system will pop up a "Compiler Settings" dialog, as shown in Figure 5.7.

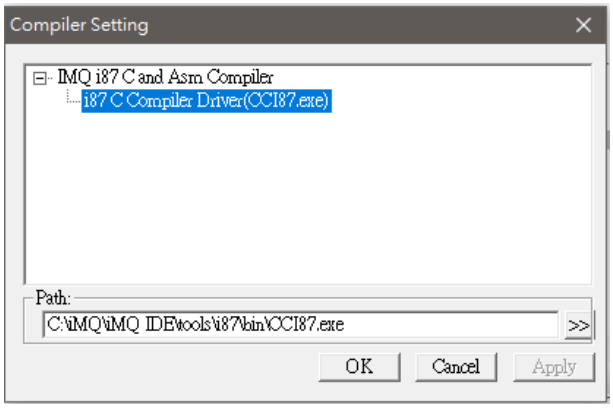


Figure 5.7 Set compiler path

5.4 Project Property Settings

Click "Project" on the main menu, and then select "Project Properties" to open the “Option” dialog. This setting item includes two parts: “General” and “Build Option”.

5.4.1 General Options

The “General” options are mainly set as shown in Figure 5.8, and there are the following types:

IC Type Setting - IC Type Setting
IC Vendor - Select the manufacturer name of the chip.
IC Serial - Select the series the chip belongs to.
IC Type - Chip Model
Trace Setting - Trace settings, this option has no effect

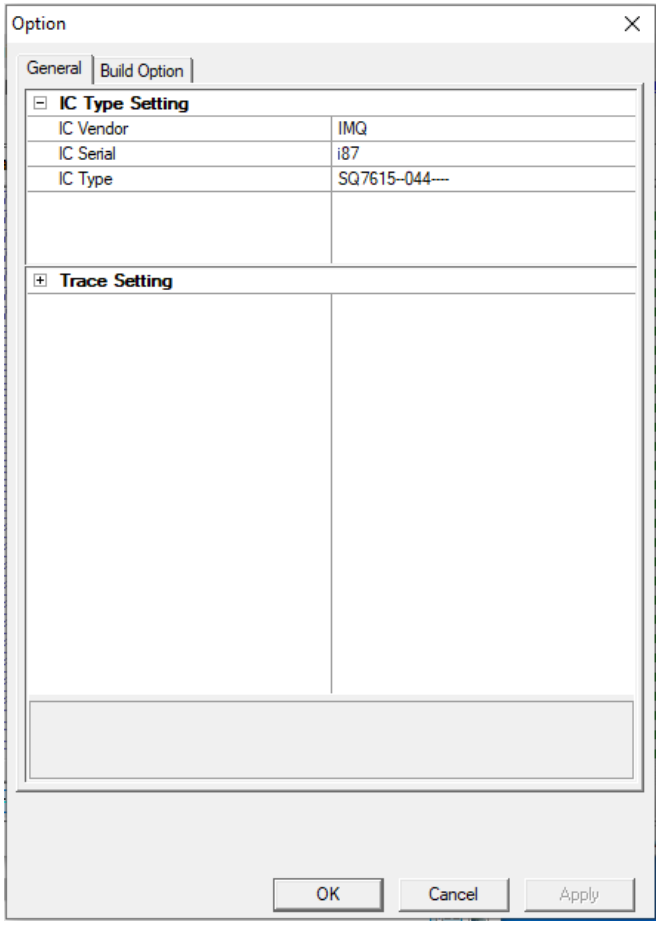


Figure 5.8 General Options for project properties

5.4.2 Build Options

The buil option mainly completes the selection and configuration of the compiler, as shown in Figure 5.9.

First, select the type of compiler you want to use from the drop-down list. Next, make the following compile settings:

Path Setting - path setting

Include Path - Path to include files used by the compiler, reserved for other compilers

Library Path - The library file path used by the compiler, reserved for other compilers

Linker Path - the linker file path used by the compiler, reserved for other compilers

Assemble Setting - assembler settings

C Compiler Setting - C language Compiler Settings

Linker Setting - Linker Settings

C Code static analysis - Enable or disable C language source code check

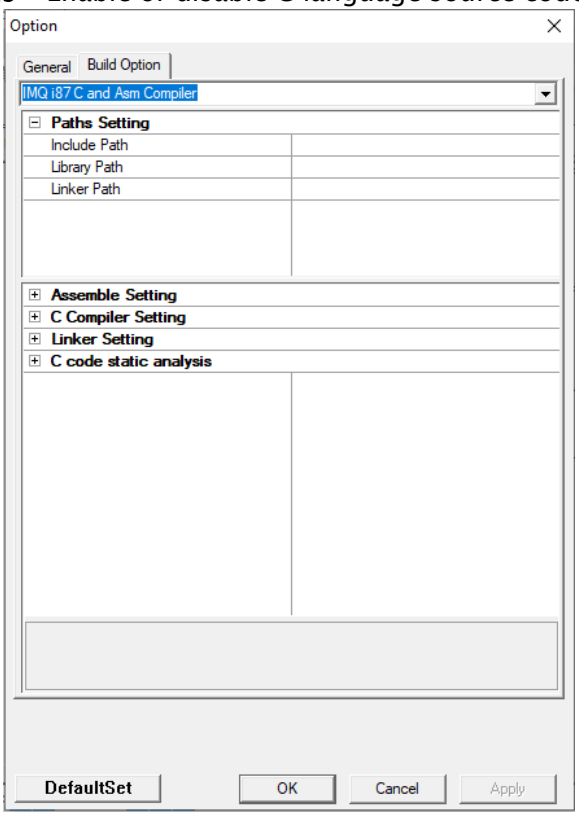


Figure 5.9 Build options for project properties

5.5 Compile Programs

After the above settings are completed, you can start compiling the program. There are two types of compilation in iMQ IDE: compiling release version and debug version. If you choose to compile the debug version, the system will automatically add _DEBU macro definition when compiling. The compile command can be invoked through the shortcut toolbar or menu bar. The compilation result will appear in the "Build Results" window.

If there is an error in the program, double-click the error line or warning line in the build result to directly locate the relevant line of the source code. Some errors, because they have nothing to do with a specific line of code, the compiler cannot locate them, such as link errors, etc.

5.6 Pin Setting

MQ series MCU may need pin settings of the different packages, select external high/low speed crystal oscillator or rolling code. The time to use this setting is to write the package pin information of the chip into the IC when using the i87 IDE to program the chip so that the function settings of each pin are correct to the IC package. If you use the flash writer (Writer 300) to program the chip, you don't need to consider this function.

Taking MQ6801 as an example, the setting method is:

1. Click the menu "Project -> Pin Setting" to open the Pin Setting dialog box.

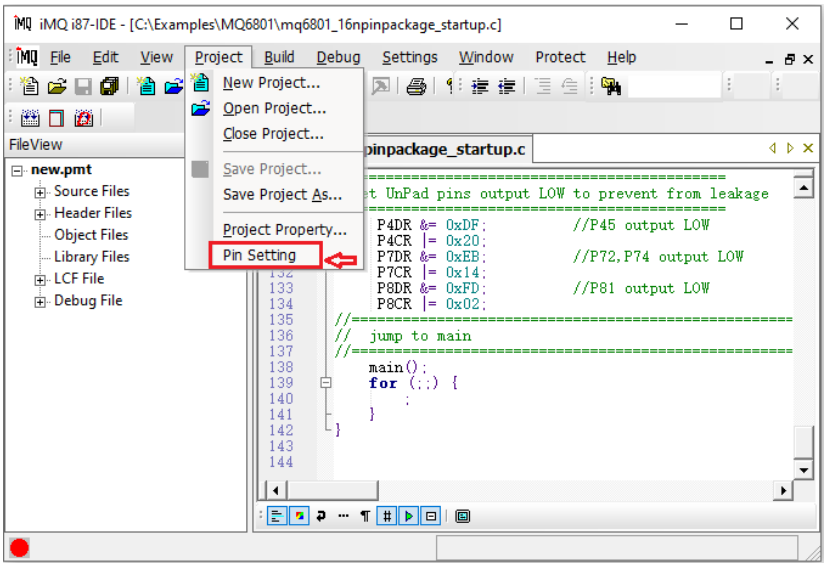


Figure 5.10 Select the pin function in the project menu

2. After opening the dialog box, you can choose the packages of MQ6801 with DP016, SP016, SP16N, SS016, DP020, SP020, and SS020.

Figure 5.11 Select the package

3. After selecting the pin package, click the drop-down list to set the pin function.
   The leftmost side "ext low OSC" determines whether the external input clock source uses a high-frequency or low-frequency crystal oscillator. If "ext low OSC" is checked, it is an external low-frequency clock input, and if it is not checked, it is an external high-frequency clock input.
   And if you check Protect, It can protect Flash content information.

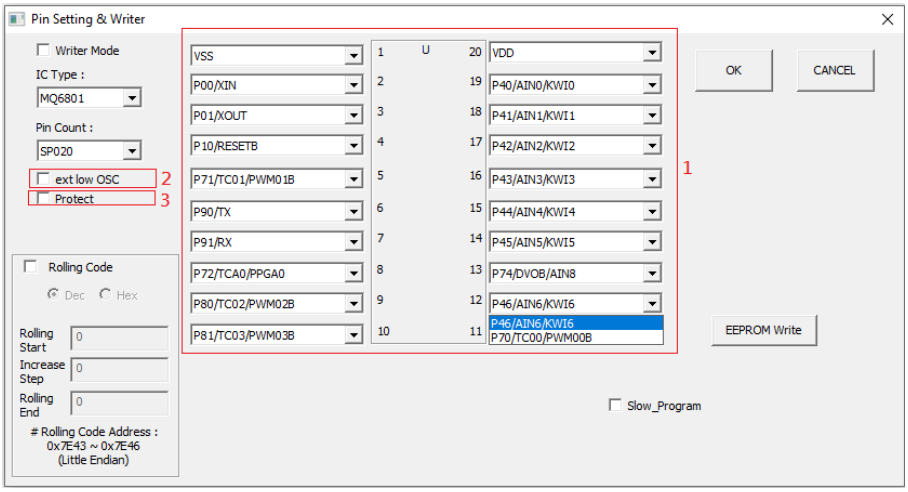


Figure 5.12 Pin setting and external clock source setting

4. After completing the setting, click "OK". It will pop up a "Type and Setting" dialog box, click "Yes" to save the setting, and the system will automatically save the setting file to the project path, and the file name is "<project name >.cfg". When the Pin Setting dialog box is opened next time, the system will read the last setting file from the project path.

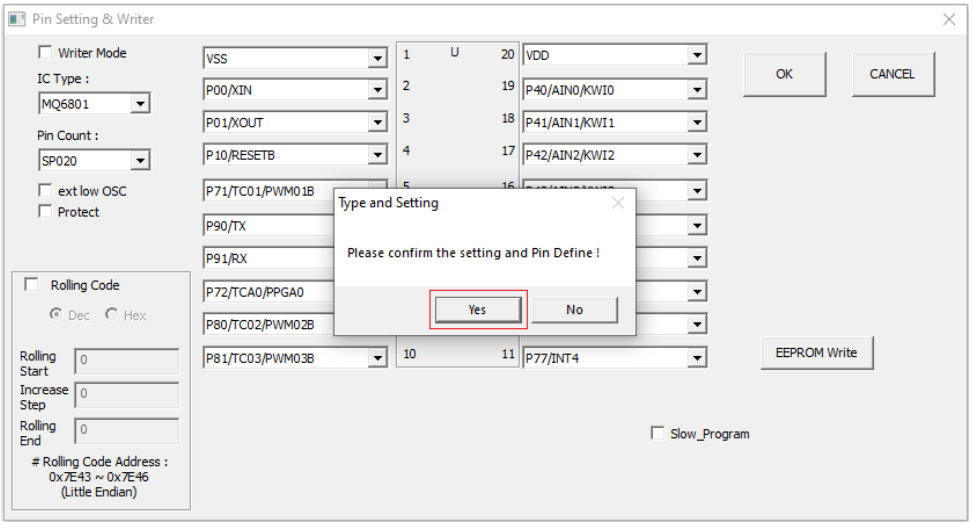


Figure 5.13 Complete the settings

5.6.1 Rolling Code setting

This function can burn data to a specific area. Take MQ6812 Figure 5.14 as an example, in

the pin setting screen: 1. Select PinCount 2. Check RollingCode 3. Select DEC (decimal)\HEX (hexadecimal) 4. Enter the rolling code start in the Rolling Start box Initial value (an example is 0x12345678) 5. Enter the increment of the rolling code value (this example increments by 1 each time) 6. The End value of rolling code (take this picture as an example, when the value is greater than 0x12345678A, it will not continue to increment) 7. Rolling code Code programming destination address 8. Click OK. The IDE will generate a CFG file to record the settings under the project path, and it will be burned into the chip when the chip is programmed. Currently, the IDE supports rolling code settings for MQ series MCU, for example, MQ6801, MQ6811, MQ6812, MQ6815, MQ6821, MQ6822, MQ6825... and other products.

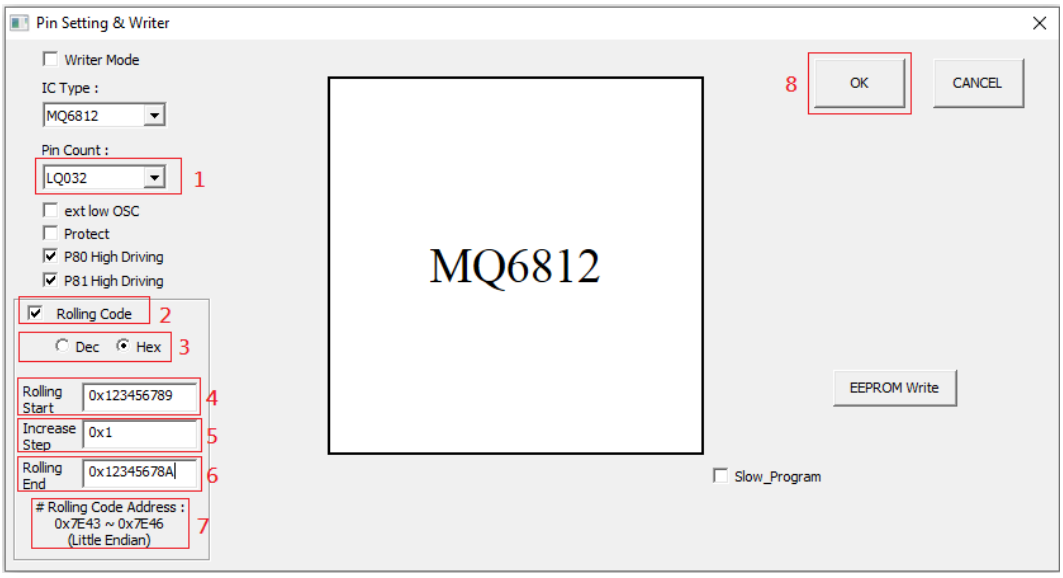


Figure 5.14 MQ6812 pin configuration diagram

After entering the debugging, you can use the memory observation window to confirm the burning data of the rolling code, as shown in Figure 5.15.

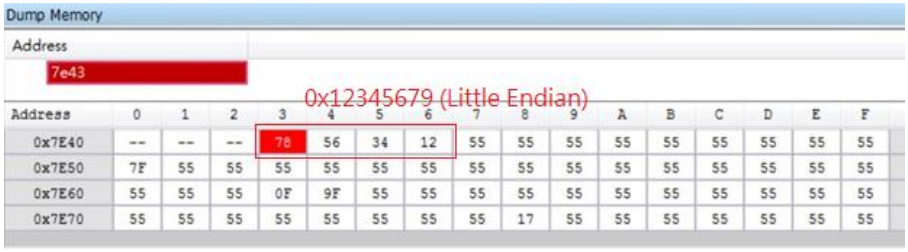


Figure 5.15 Rolling code storage address

5.7 Enter Debug Mode

After the above preparations are completed, click "Debug -> Start Debug" on the main menu, or directly click the icon on the shortcut toolbar to enter the debugging mode.

If you disable "Check build version while entering debug in the menu bar "Settings -> IDE Config..." (as shown in Figure 5.16). When you click "Start Debugging", it will not check whether the compiled version is the "debug version" or "Release Version".

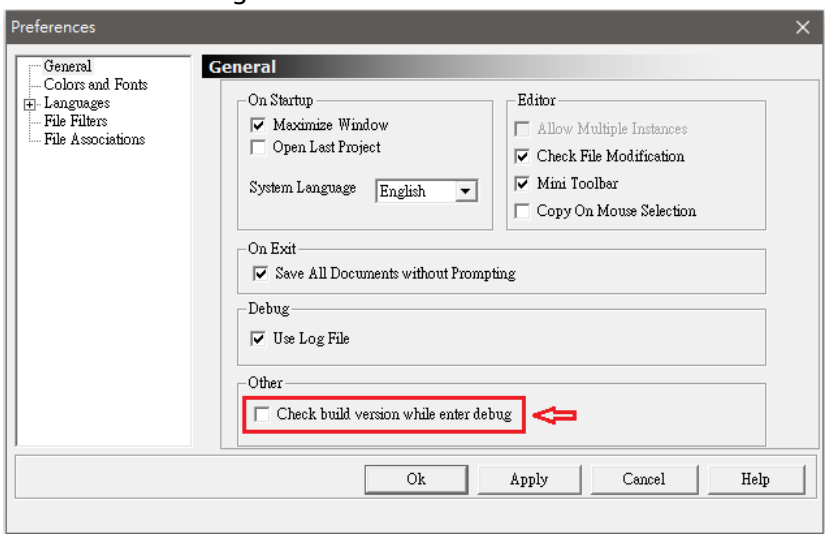


Figure 5.16

If "Check build version while enter debug" is enabled (as shown in Figure 5.17), the "debug version" will be checked when entering debugging. If the current compiled version is the "release version", a confirmation window will pop up (as shown in Figure 5.18).

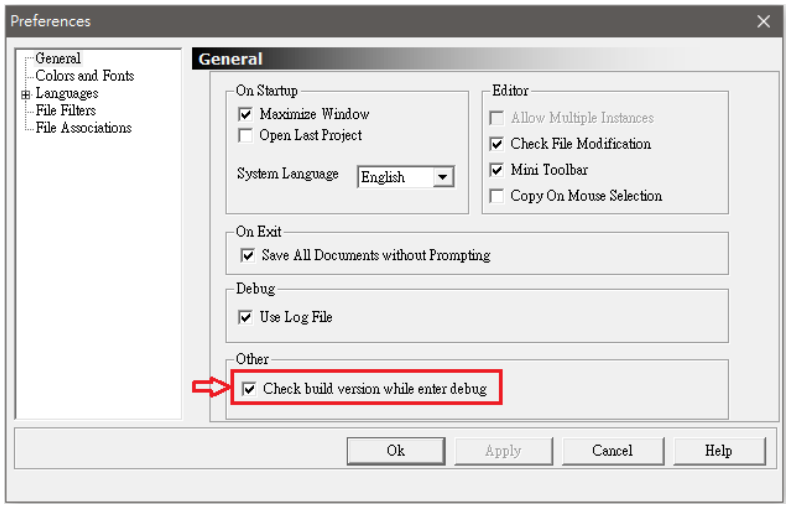


Figure 5.17

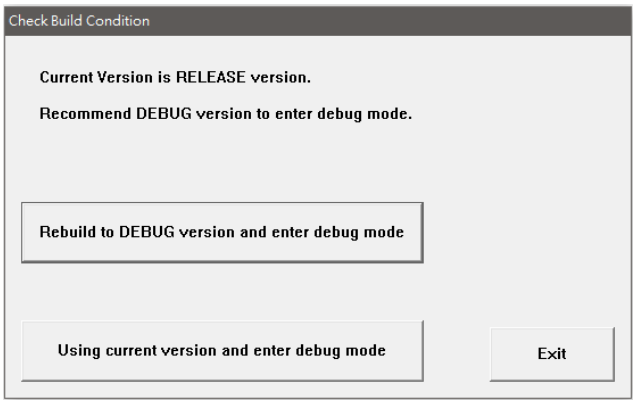


Figure 5.18

If the HEX file is older than the original code file, the IDE judges that it needs to be recompiled to update the HEX file. At this time, it will automatically compile it into a "debug version" and then enter the debug mode.

If the MQ-Link hardware is not connected or there is a problem with the connection, the software will prompt "Fail to connect".
If the MQ-link hardware is connected, but the MCU device does not exist or is not connected, then when entering debug mode may hold MQ-Link hardware(the “Active” LED not blinking). If this situation happens, the user needs to re-plug the USB cable into Host PC.

After entering the debugging mode, you can start debugging the program, modify the source code according to the debugging situation, and then recompile. Commonly used debugging methods are generally:

1. Set a breakpoint, then go run until the program stops at the breakpoint.
2. Focus on the area to do a single step, observe variables, etc.,

5.7.1 Chip Project and Corresponding Chip List

When entering the debugging mode, the IDE software will compare the projected chip with the connected chip. The following table shows some of the chip projects and the corresponding connectable target chip/evaluation board.

Chip project name	Connectable chip	Evaluation board
MQ6801--020----	MQ6801 Family chip	MQ6801 EVBoard
MQ6801--016----	MQ6801 Family chip	MQ6801 EVBoard
MQ6801—16N——	MQ6801 Family chip	MQ6801 EVBoard
MQ6811--010----	MQ6801 Family chip, MQ6811 Family chip	MQ6801 EVBoard

MQ6812--032----	MQ6812 Family chip	MQ6812 EVBoard
MQ6812--028----	MQ6812 Family chip	MQ6812 EVBoard
MQ6821--020----	MQ6812 Family chip, MQ6821 Family chip	MQ6812 EVBoard
MQ6821--016----	MQ6812 Family chip, MQ6821 Family chip	MQ6812 EVBoard
MQ6832 — 032- ---	MQ6832 Family chip	MQ6832 EVBoard
MQ6832 — 016- ---	MQ6832 Family chip	MQ6832 EVBoard
MQ6815--048----	MQ6815 Family chip	MQ6825 EVBoard
MQ6815--044----	MQ6815 Family chip	MQ6825 EVBoard
MQ6825--048----	MQ6825 Family chip	MQ6825 EVBoard
MQ68 3 5--048----	MQ68 3 5 Family chip	MQ6835 EVBoard
MQ6 902 --0 28 ----	MQ6 902 Family chip	MQ6807 EVBoard
MQ6801	MQ6801 Family chip	MQ6801 EVBoard
MQ6801_16PinPackage	MQ6801 Family chip	MQ6801 EVBoard
MQ6811_10PinPackage	MQ6801 Family chip, MQ6811 Family chip	MQ6801 EVBoard
MQ6812_28PinPackage	MQ6812 Family chip	MQ6812 EVBoard
MQ6812_32PinPackage	MQ6812 Family chip	MQ6812 EVBoard
MQ6821_16PinPackage	MQ6812 Family chip, MQ6821 Family chip	MQ6812 EVBoard
MQ6821_20PinPackage	MQ6812 Family chip, MQ6821 Family chip	MQ6812 EVBoard
SQ7615—044----	SQ7615 Family chip	SQ7615 EVBoard
SQ7655—032----	SQ7653 Family chip	SQ7655 EVBoard
SQ7705.ADM—064----	SQ7705 Family chip	SQ7705 EVBoard
SQ7705.USR—064----	SQ7705 Family chip	SQ7705 EVBoard

5.8 Breakpoints Management

5.8.1 Program Breakpoints

Program breakpoint is the most commonly used breakpoint, that is, it is set at a certain address in the program, and once the program runs here, it will stop. The i87 IDE uses the breakpoint management window to manage various breakpoints in a unified manner. Double-click the program breakpoint item in the “BreakPoint” window, and you can quickly jump to the corresponding program line, which is very convenient for positioning. In the window, breakpoints can be edited, such as adding, deleting, enabling or disabling, adding judgment conditions, and so on.

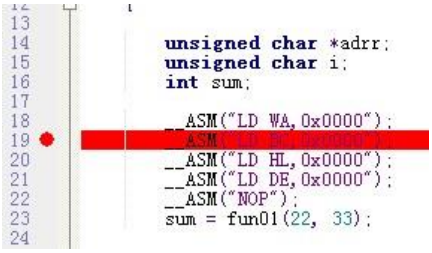
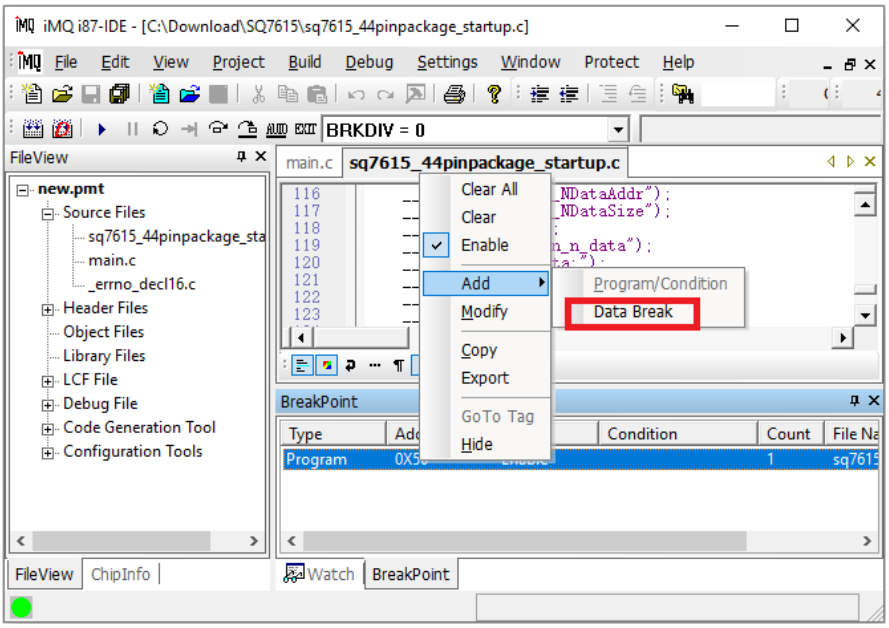


Figure 5.19 Program breakpoint example

5.8.2 Data Breakpoints

The data breakpoint function is to generate a breakpoint when writing/reading the specified address or to generate a breakpoint when the program executes the specified address. Click the right mouse button in the “breakpoint window” and select "Add” -> “Data Break” to set the data breakpoint.



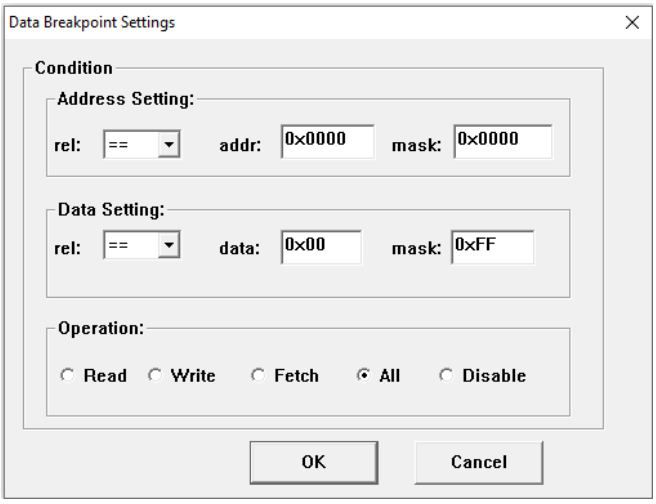


Figure 5.20 Data breakpoint function dialog

1. **Relationship (rel)**: The default is "==".
2. **Address**: specified address
3. **data**: specified data
4. **Mask**: If a bit mask is 1, then the bit is 0 or 1 all valid; the mask is 0. It does not affect the address\data setting.

   Example: address mask condition valid address

   0xf123 0x000f 0xf120 ~ 0xf12f

   0xf123 0x0003 0xf120~ 0xf123

   0xf123 0xf123 0xf123

   data mask condition valid data

   0x45 0x0f 0x40~0x4f

   0x45 0x00 0x45
5. **Operation**: Set the conditions for the establishment of the action

   Read: When the chip reads the specified data at the specified address, a breakpoint is triggered.

   Example: Address: 0x0040 Mask: 0x001f

   Data: 0x08 Mask: 0x0f

   Operation: read

   The action of this setting is that when the chip reads from 0x0040~0x005f to 0x8~0xf, a breakpoint will be generated.

   Write: Trigger a breakpoint when the chip writes the specified data at the specified address.

   Example: Address: 0x0040 Mask: 0x001f

   Data: 0x08 Mask: 0x0f

   Operation: write

   The action of this setting is that when the chip writes from 0x0040~0x005f to 0x8~0xf, a breakpoint will be generated.

   Instruction Fetch: When the chip runs to the specified address, a breakpoint is triggered.

   Example: Address: 0xf000 Mask: 0x0010

   Operation: Fetch

   The action of this setting is that when the chip runs to 0xf000 or 0xf010, a breakpoint will be generated.

5.9 Variable Watch

5.9.1 Variable Watch and Adding Variables

iMQ i87-IDE provides a variable watch function. You can manually enter the variable name in the “watch window” to add the variable; the case of the input variable name must be consistent with that in the source program. If the variable is invalid, the system will prompt "Symbol not found".

For C language, this software not only supports its basic data types, such as char, unsigned char, int, short int, long, unsigned long, float, etc. but also supports various complex variable formats such as multidimensional arrays, structures, pointers, etc. Several viewing methods, such as binary, decimal, hexadecimal, ASCII codes, etc. If you see a variable value displayed as a strange symbol, don't worry, it's because the variable's display attribute is set to ASCII code, just change its attribute to HEX or Decimal.
The following uses C language as an example to illustrate the method of adding observation variables:

1. Create or open an existing project, edit the source program, enter debug mode, and open the watch variable (as shown in Figure 5.21). then Watch Variables.

2. The general method of adding watch variables is: to double-click the blank line in the watch window name field to enter the editing state and enter the variable name. But the conventional method is error-prone, it is recommended to use the following simple method.

3. The simple way to add observation variables is: Open the symbol window, which lists all symbols, including special register names, address labels, user-defined RAM variable names, etc., and you can easily add them to the watch by using the right-click menu select “Add watch”.

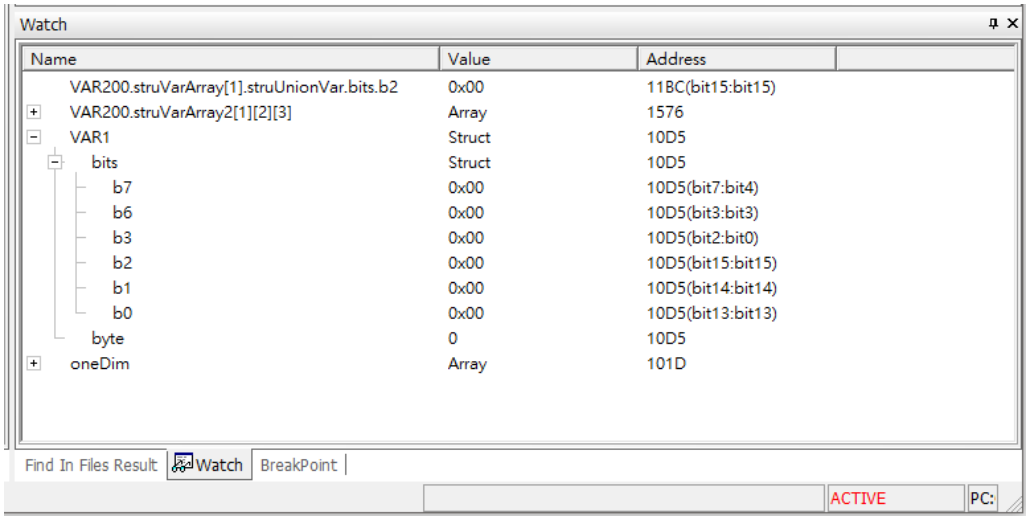


Figure 5.21 Variable watch window

4. The method to modify the display property of the value is: to right-click on the

Value field, and the field will display red, select "Property", and select the type you want from the "Data Format" drop-down list (as shown in Figure 5.22).

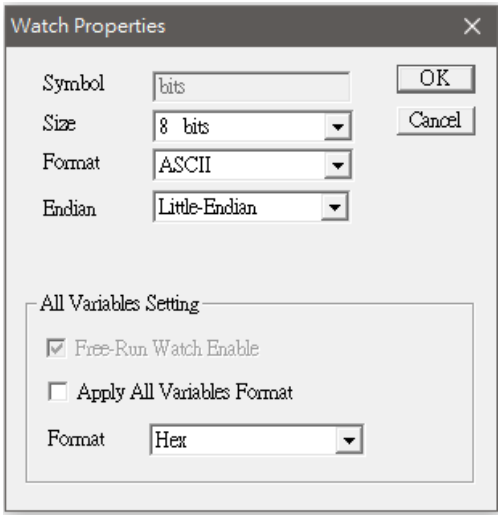


Figure 5.22 Watch Item Properties

5. Use the mouse to select the variable to be observed in the IDE editing window. Note: When using the mouse to make a selection, the selection must start from the front end of the variable name, as shown in the figure below :

Step 1: Use the mouse to select variables from the beginning

Step 2: Press the right mouse button and the popup menu window will appear and select " Add Watch"

Step 3: The selected variable will appear in the "Watch window "

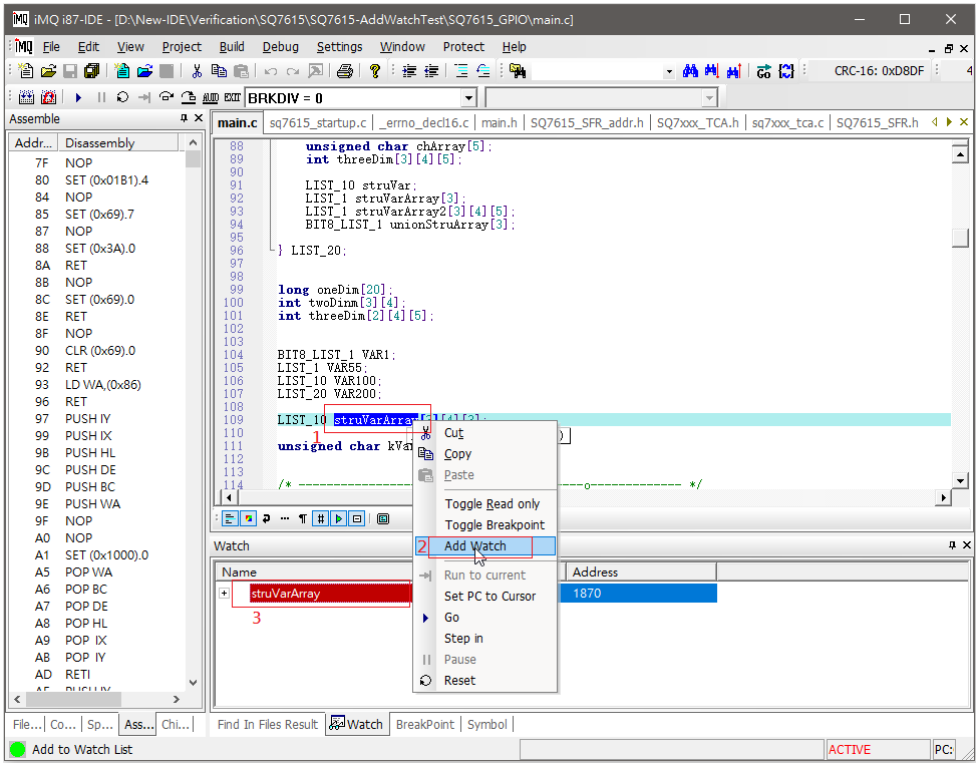


Figure 5.23 Use the mouse to select the variable to be watched

6. Manually enter the name of the variable to watch. In the " Watch Window ", you can directly input the variable name by keyboard manually.

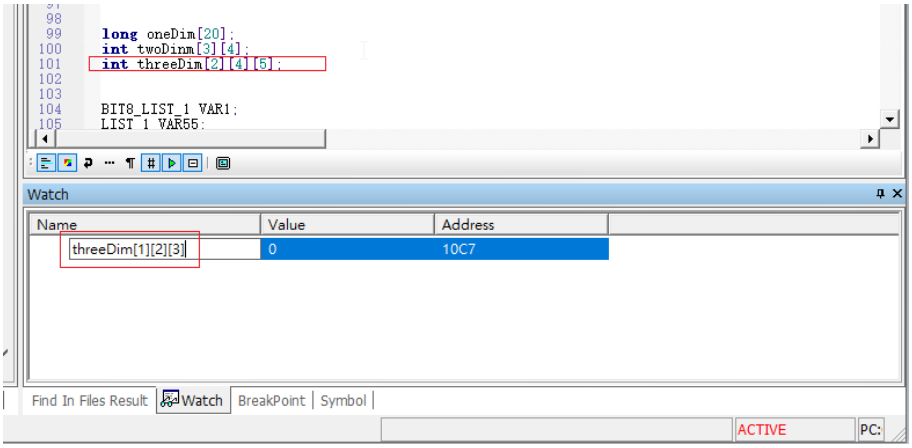


Figure 5.24 Manually enter the variable name directly in the "Watch Window "

5.9.2 Allow Partial Variable Name Format

5.9.2.1 Simple Arrays

Variables not only allow full variable names but also partial variable names, for example, a three-dimensional array of integers

Int threeDim[][][];

It can manually enter the range of variables to be observed in the watch window. Several formats are allowed:

1. threeDim[1][2][3] -- watch single variable value
2. threeDim -- all variables of all dimensions can be observed
3. threeDim[1] -- all variables under threeDim[1] can be observed
4. threeDim[1][2] -- all variables under threeDim[1][2] can be observed

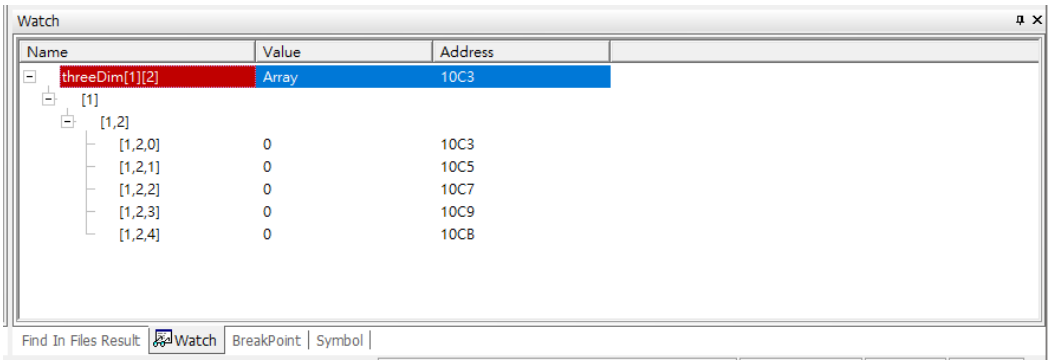


Figure 5.25 Manually enter the range of variables to be observed in the "Watch window"

5.9.2.2 Simple Structure (Union)

The same is true for structures, such as a Union structure

```
typedef union
{
    struct {
        unsigned char  b7:4;
        unsigned char  b6:1;
        unsigned char  b3:3;
        unsigned char  b2:1;
        unsigned char  b1:1;
        unsigned char  b0:1;
    }bits;

    unsigned int byte;
} BIT8_LIST_1;

BIT8_LIST_1 VAR1;
```

Figure 5.26 Enter the structure variable to be observed in the "watch window"

1. It can only enter the root variable name (here VAR1) to "watch Window" to observe

the values of all elements in the complete variable structure

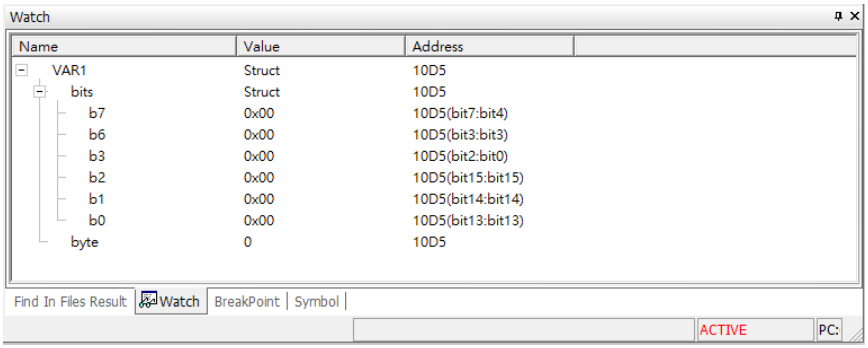


Figure 5.27 Enter the structure variable name in the "Watch window "

2. It can enter variable names such as

   VAR1.byte

   VAR1.bits.b1 → The value of a single-bit value

   To observe the value of the specified variable element

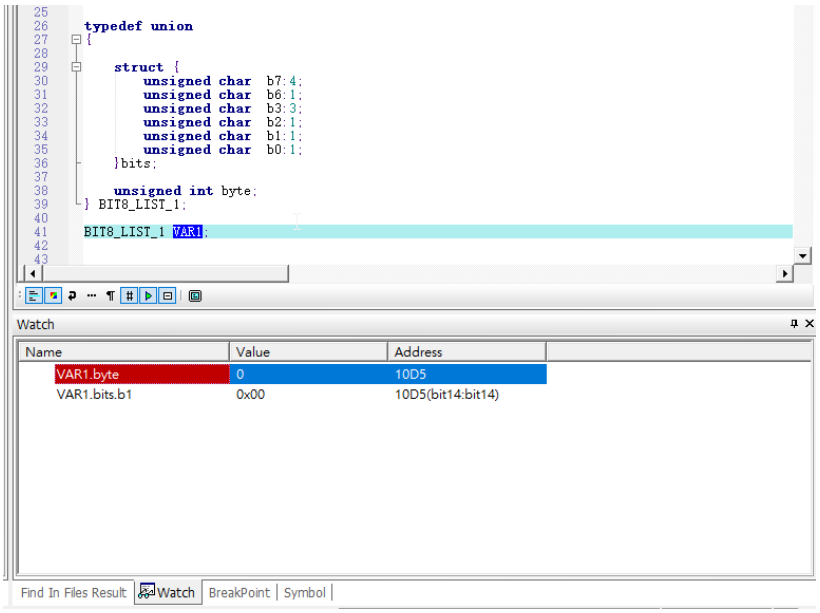


Figure 5.28 Enter the structure field name in the "Watch window "

3. It can also enter part of the variable name to observe the interesting part of the structure, for example

   VAR1.bits → Only observe the bits part of the structure

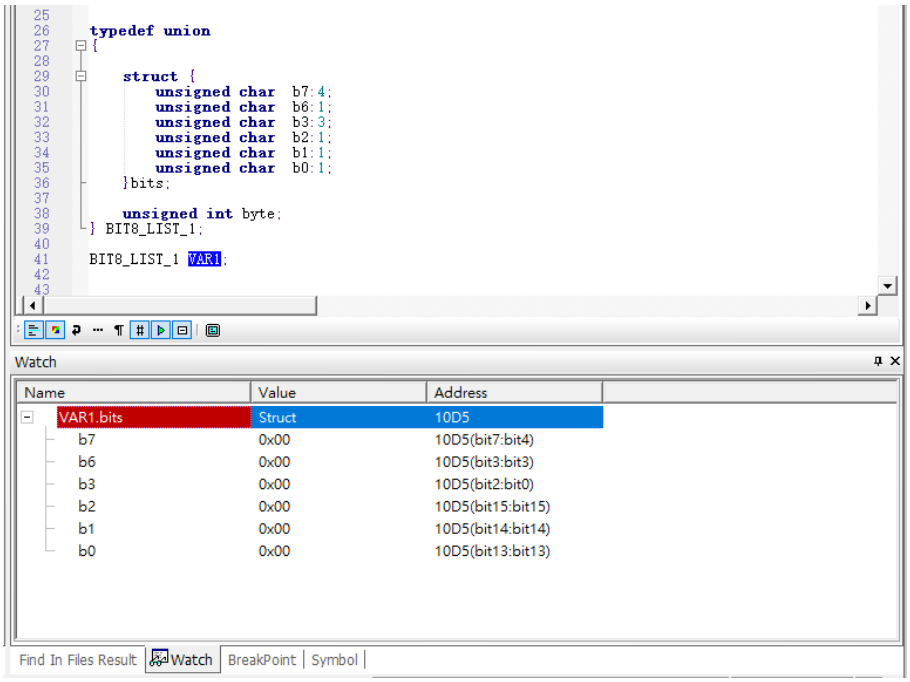


Figure 5.29 Enter part of the structure name in the "Watch window "

5.9.2.3 Complex Structure (Union)

A complex structure, including structure in structure, structure array, etc., can also input part of the name to observe the part of interest.

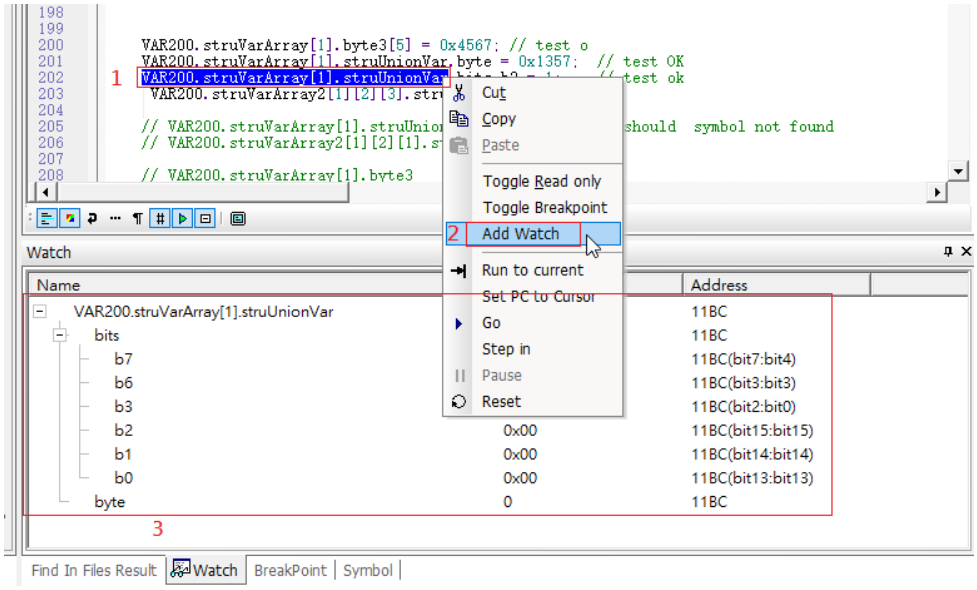


Figure 5.30 Enter the Complex structure variable in the "Watch window"

5.10 SQ76 Series Flash Program Setting

SQ76 series products (SQ7613, SQ7615, SQ7617, and SQ7653... etc.) download the firmware by default to the entire flash area, and you can also choose the area to be downloaded when downloading firmware in debug mode.

When the project is started, the “Flash configuration” option will appear below the FileView, and the setting screen will appear by double-clicking the mouse :

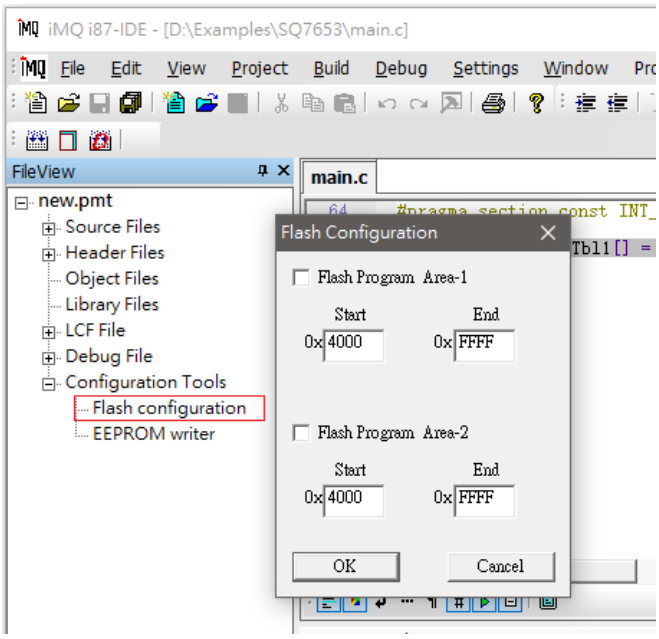


Figure 5.31

Here can select two firmware sections to write in. When the checkbox of Flash Program Area-1 or Flash Program Area-2 is checked, you can specify the area to be programmed, such as the setting in the figure below.

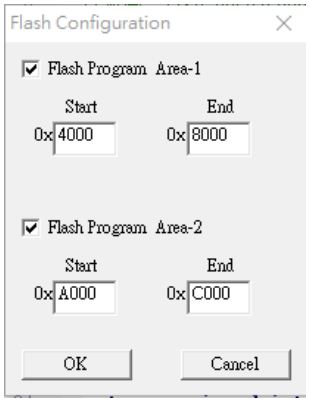


Figure 5.32

It is to select the program codes in the areas 0x4000 ~ 0x8000 and 0xA000 ~ 0xC000 in the original compiled firmware to be programmed into the chip, while the rest of the firmware will not be programmed (the original area in this area of the chip code will be maintained and will not be erased).

6. iMQ i87-IDE Writer Function

This chapter will help you understand the operation of the writer mode function in the IDE.

6.1 Open Writer Function

There are two ways to open the writer mode:

For All series chip in “Tool -> Writer” in the initial menu to open it, as shown in Figure 6.1.

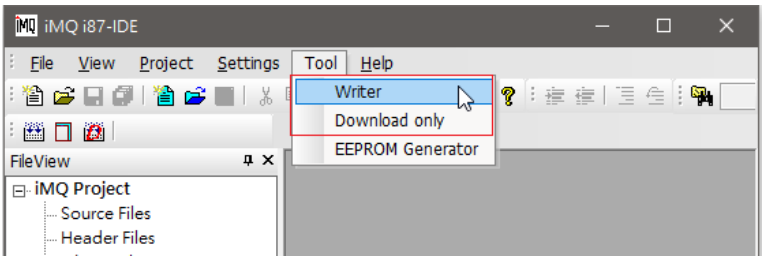


Figure 6. 1 Writer mode

Figure 6.2 Writer initial page

For MQ68/69 series chip, there has a second way is “Project -> Pin Setting”

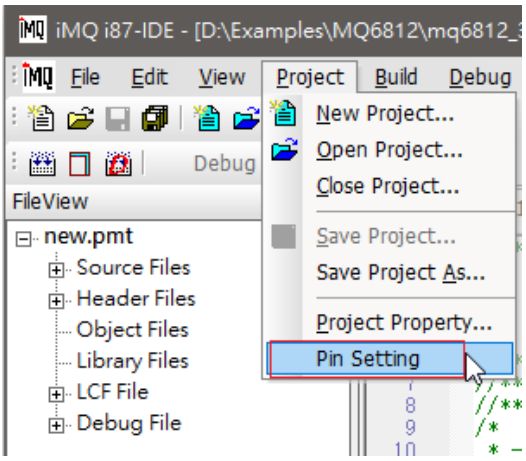


Figure 6.3 Pin Setting menu

After entering the “Pin Setting” dialog, check Writer Mode in the upper left corner to switch the writing mode, (as shown in Figure 6.4). and it will switch to general writer mode dialog (as shown in Figure 6.5).

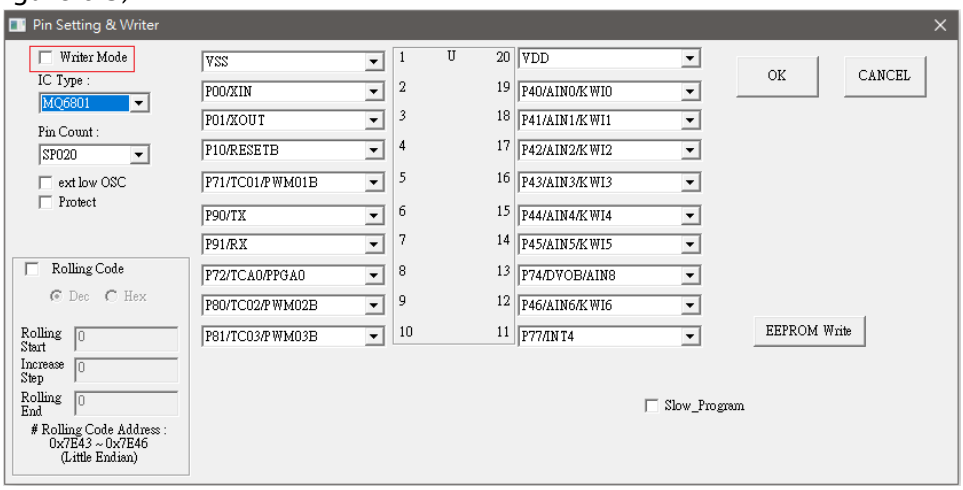


Figure 6. 4 Check Writer Mode

Figure 6.5 Switch to general Writer Mode

6.2 Writer Function Introduction

This section divides the Writer programming interface into 11 blocks, as shown in Figure 6.6, and each functional block will be divided into subsections for explanation.

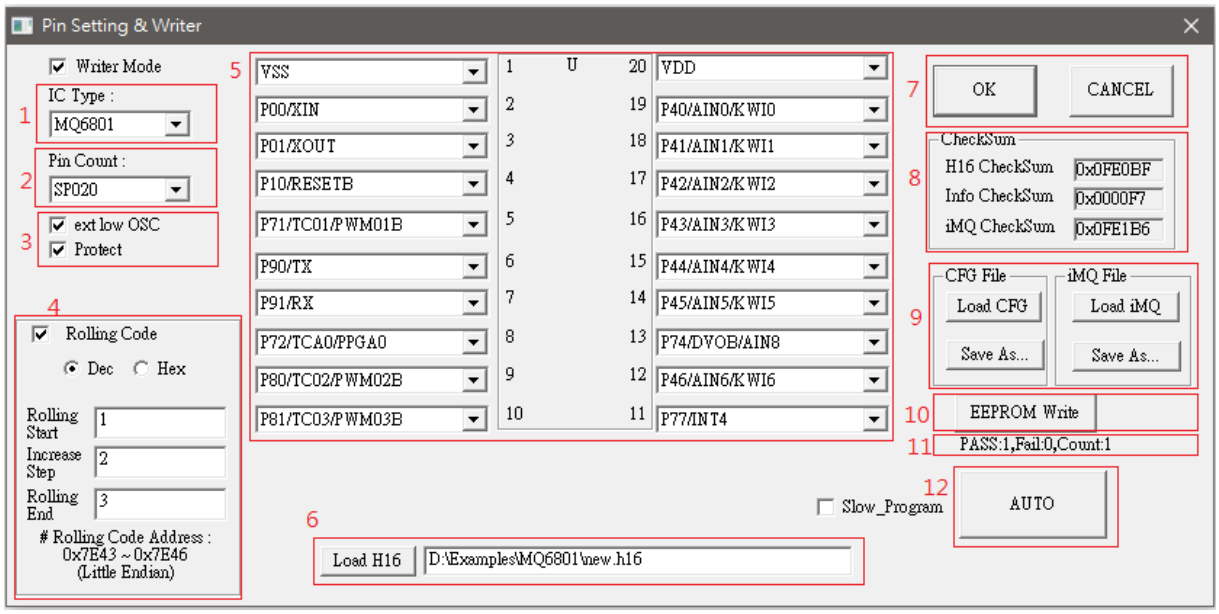


Figure 6.6 Writer function

6.2.1. IC Type Selection

In the first area, all currently supported IC models can be selected, as shown in Figure 6.7.

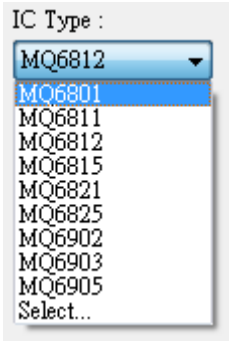


Figure 6.7 Select the pre-programmed IC model

6.2.2. Package Pin Selection

After selecting the IC model, you can pre-program the package pins of the IC model in the second area. For example, MQ6801 has different package pins such as SP016 and SS016, as shown in Figure 6.8.

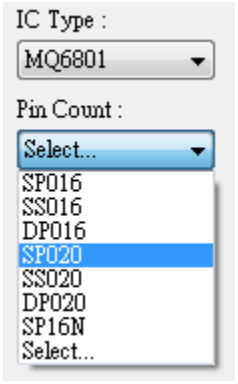


Figure 6.8. Select Package Pins

After selection, according to different package pins, the 3rd to 5th functional blocks will appear, as shown in Figure 6.6.

6.2.3. Extra Function Selection

The third area is the extra function selection area, which mainly has the following four functions, as shown in Figure 6.9, ticking "ext low OSC" means using an external low-frequency clock. "Protect" can protect Flash content. "P80 High Driving" and "P81 High Driving" are to set P80 or P81 from general I/O to high driving I/O. Note that MQ6812 series P80 and P81 are preset as high driving I/O. If you want to switch to Generally I/O can be unchecked it. In addition, the increase or decrease may vary due to different IC models. Not all IC models can set the above 4 functions.

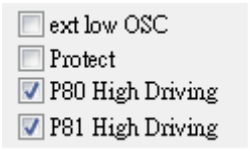


Figure 6.9 Functional block

6.2.4. Rolling Code

The fourth area can set the rolling code value and can choose Dec (decimal) or Hex (hexadecimal) representations. "Rolling Start" is the starting value of the rolling code. "Increase Step" is the incremental value of the rolling code. "Rolling End" is the end value of the rolling code. As shown in Figure 6.10, the rolling code value starts counting from 1 and increments by 2 each time, and the final rolling code value is 3. Note!! The maximum range of the rolling code is 0~4294967280 (decimal) or 0~0xFFFFFFFF (hexadecimal).

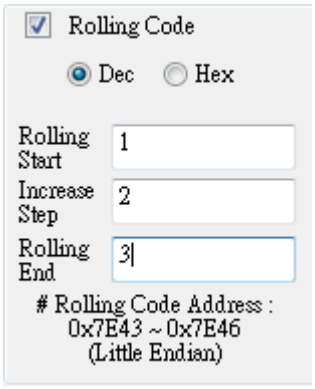


Figure 6.10. Rolling code setting

6.2.5. Pin Setting

In the fifth area, only the IC model of MQ6801 can switch some specific pins to different functions according to the user's application, as shown in Figure 6.11, taking MQ6801 SP020 as an example. The rest of the remaining IC models do not support this function, and only the picture of the IC model will appear, as shown in Figure 6.12.

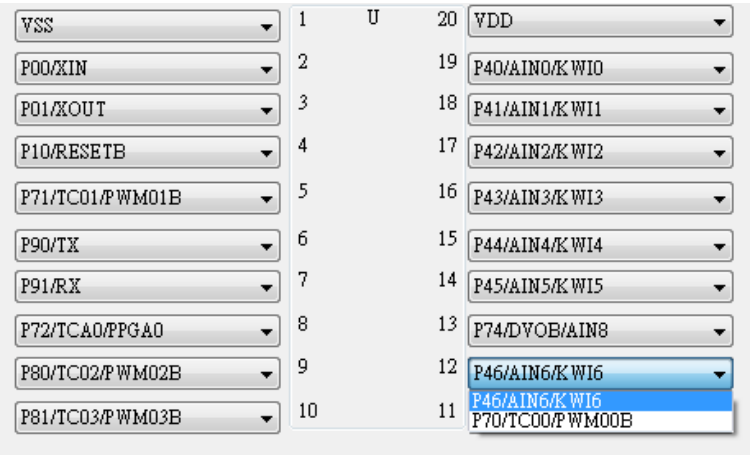


Figure 6.11. Pin setting switch

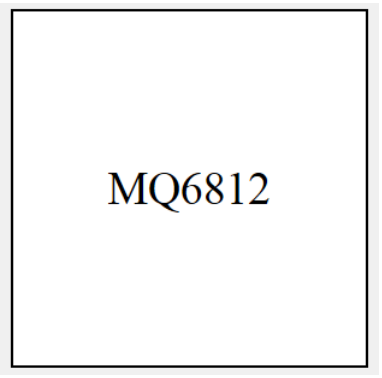


Figure 6.12. IC Diagram

6.2.6. Load Programming File

Press the Load H16 button to load the H16 programming file after selecting the path, as shown in Figure 6.13.

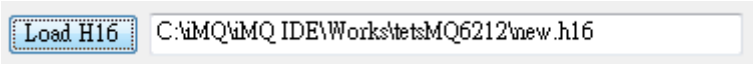


Figure 6.13 Load H16 file path

6.2.7. Confirm Settings and Start Programming

After setting areas 1 to 6 or by loading old or newly saved settings in area 9, press the OK button, and a window will appear for final confirmation, as shown in Figure 6.14, confirming that it is correct and then clicking "Yes".

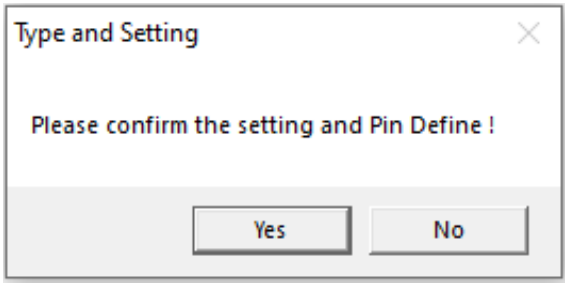


Figure 6.14 Confirmation window

Then, the currently stored Checksum information will be displayed in the eighth area, as shown in Figure 6.15, "H16 Checksum" is the checksum value loaded into the programming file H16. "Info Checksum" sets the checksum value for zone 1~7 for the user. "iMQ Checksum" is the check value of the sum of the above two values, and this information is used for checking the programming result.

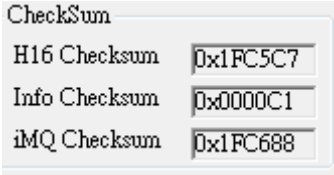


Figure 6.15 CheckSum information

Finally, you can press the “AUTO” button in the 12th area. After programming, if the PASS window appears, the programming is successful, as shown in Figure 6.16. If it fails, please refer to Section 6.3 for an explanation of error messages. The 11 area is to record the number of successes, failures, and total of the current programming, as shown in Figure 6.17.

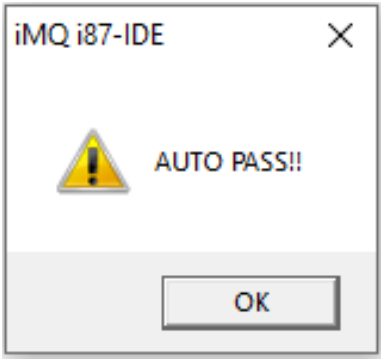


Figure 6.16 Programming success

Figure 6.17 Programming record

6.3 Writer Error Message

error message	description

Rolling Code over your Setting, please reset your Rolling Code!!	The Rolling Code has reached the upper limit and needs to be reset
Enter PROM Mode Fail!! Enter PROM Mode Fail. Please check your device or try it again !!	Failed to enter PROM Mode, it may be because the IC is not placed properly, or there is a transmission problem, you can try again
Check Info Block Error, error code=0x02 Program InfoBlock Error, error code=0x02, Please try again !!	The possible reasons are that the IC on the programmer is not placed properly, the transfer board is incorrect, the packaging is bad (open short), the IC version is incorrect, or there is no IC to be programmed.
ProgramRollingCode Error, error code=0x05, Please try again !!	This error code will appear when the frequency of the internal crystal oscillator (HIRC) after trimming cannot be within the specification, or the frequency cannot be measured.
Check Info Block Error, error code=0x06	This error code will appear when the frequency of the internal crystal oscillator (HIRC) after trimming cannot be within the specification, or the frequency cannot be measured.
Program InfoBlock Error, error code=0x09, Please try again !!	After data is written into the MCU internal memory data block (Info Block), the comparison with the original data fails. Or this error code will appear when the usage times of the OTP MCU internal memory data block (Info Block) exceeds the limit.
Check Info Block Error, error code=0x0A	The number of pins of the IC does not match the setting of the programming file
Check Info Block Error, error code=0x0E	This IC failed the CP test IC.
Match Fail!!Please check IC Type or Device !!	The reason is that the programmed IC does not match the configuration or the device is not properly placed
Blank Check Fail!!	If the Blank Check fails, it may be because there is not enough space in the Program Memory for programming, or there is a problem with the OTP.
Erase Fail!!	An error occurred while wiping the IC
Write Fail!!	An error occurred during IC programming
Verify Fail	The Flash content in the programmed IC does not match the loaded H16
Program Info Block Fail!!	An error occurred while writing to the InfoBlock

6.4 Open Download Only Function Menu

Figure 6.18 Enable Download only programming function

"Tool -> Download only", this function will program the h16 file only, and not update the information of the info block.

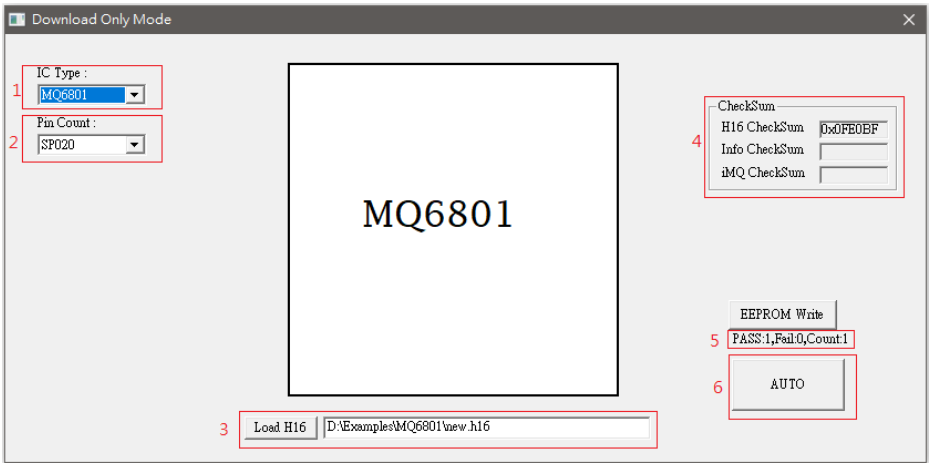


Figure 6.19 Download only programming function

6.4.1. IC Type Selection

In the first area, all currently supported IC models can be selected, as shown in Figure 6.20.

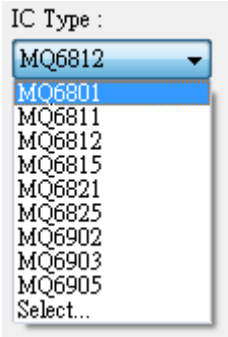


Figure 6.20 Select the IC model

6.4.2. Package Pin Selection

After selecting the IC model, you can program the package pins of the IC model in the

second area. For example, MQ6801 has different package pins such as SP016 and SS016, as shown in Figure 6.21.

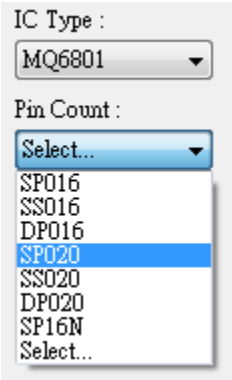


Figure 6.21. Select Package Pins

6.4.3. Load Programming File

Press the Load H16 button in the third area, and after selecting the path, the H16 programming file can be loaded, as shown in Figure 6.22.

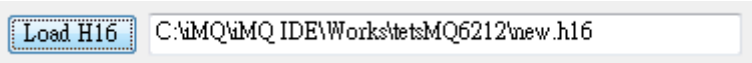


Figure 6.22 Load H16 file path

The current Checksum information will be displayed in the fourth area, as shown in Figure 6.23, "H16 Checksum" is the checksum value of the loaded programming file H16.

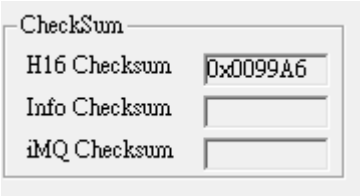


Figure 6.23 CheckSum information

6.4.4. Programming

Press the AUTO button in the sixth area to start programming. If the PASS window appears, the programming is successful, as shown in Figure 6.24. And the fifth area is to record the number of successes, failures, and total count of the current programming, as shown in Figure 6.25.

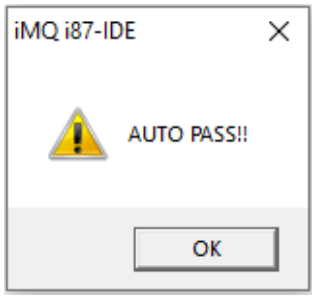


Figure 6.24 Programming success

Figure 6.25 Programming record

6.5 EEPROM Writer

6.5.1 EEPROM Writer Function

The EEPROM writer function can program the SQ series EEPROM and MQ series user info ROM area.

There are two ways to start the EEPROM writer:

Note: This chapter uses SQ7617 as an example.

(1) After opening the project, the option "EEPROM Writer" will appear at the bottom of the FileView window, double-click with the left mouse button :

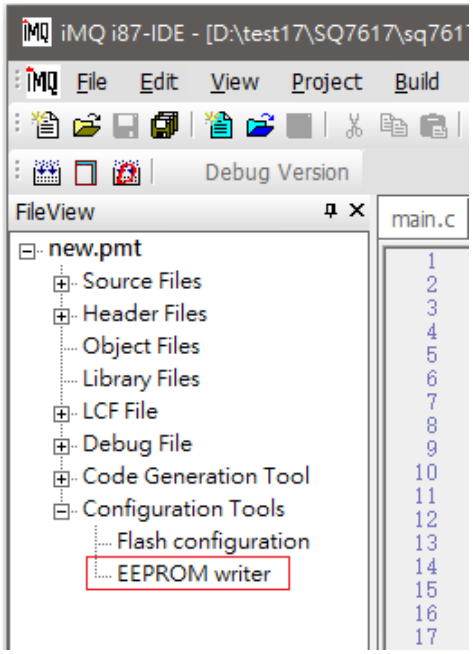


Figure 6. 26 FileView window

(2) Open the Writer dialog from Writer mode (menu Tool → Writer or Download only), and follow the steps below:

(i) Select IC Type

(ii) Select Pin Count and it will appear

(iii) The button of "EEPROM Write":

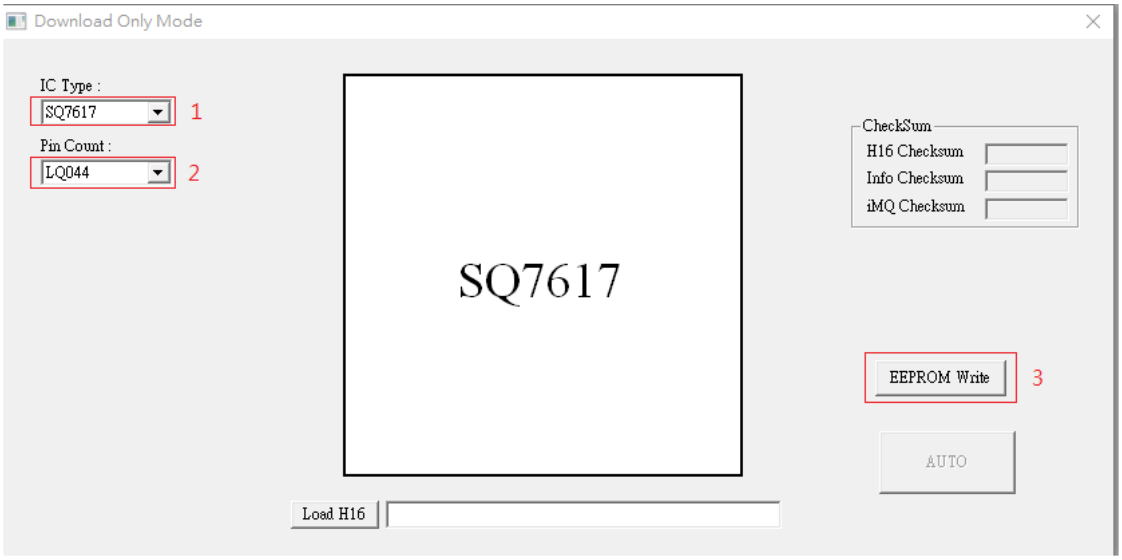


Figure 6.27 Set IC Type, Pin Count, EEPROM Write

EEPROM Writer dialog box that can be opened using the above two methods

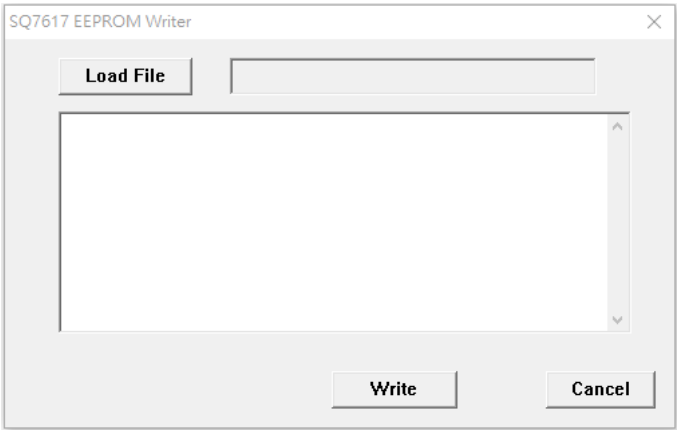


Figure 6.28 Load the file to be written into EEPROM

Press the "Load File" button to load the file to be written into the EEPROM

Press "Write" to write the loaded file into SQ7617 EEPROM.

6.5.2 EEPROM File Generator

The EEPROM writer uses a fixed format as follows:

This file can be edited by the general editor or by IDE “EEPROM Generator” tool.

EEPROM writer file format:

```
################################################
# 0x0000: AREA-1
################################################
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF 16 00 0E C8 00 00 89 C8 C3
################################################
# 0x0110: AREA2
################################################
0A 11 22 33 44 55 66 16 00 0E C8 00 00 89 C8 C3
0B 22 FF FF FF FF FF 16 00 0E C8 00 00 89 C8 C3
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```

Format specification:

: Those starting with this will be ignored

0x0110 : Specify the starting address of this block, followed by Area-2 as a comment, which can be any string

0A 11 …. : Fixed 16 Byte data to be programmed (Hex format)

Open “Tool->EEPROM Generator” to generate the EEPROM file, as shown in Figure 6.29.

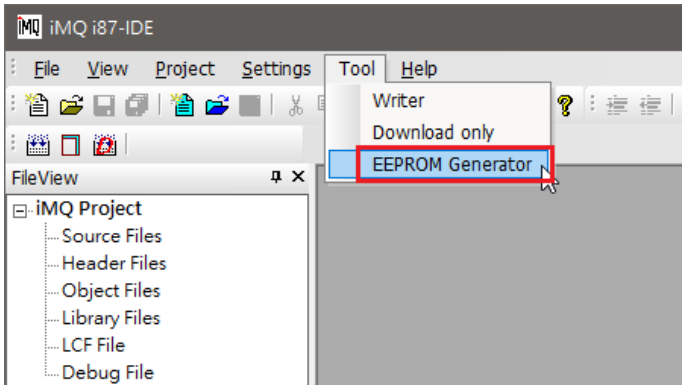


Figure 6.29 Open the “EEPROM Generator” tool

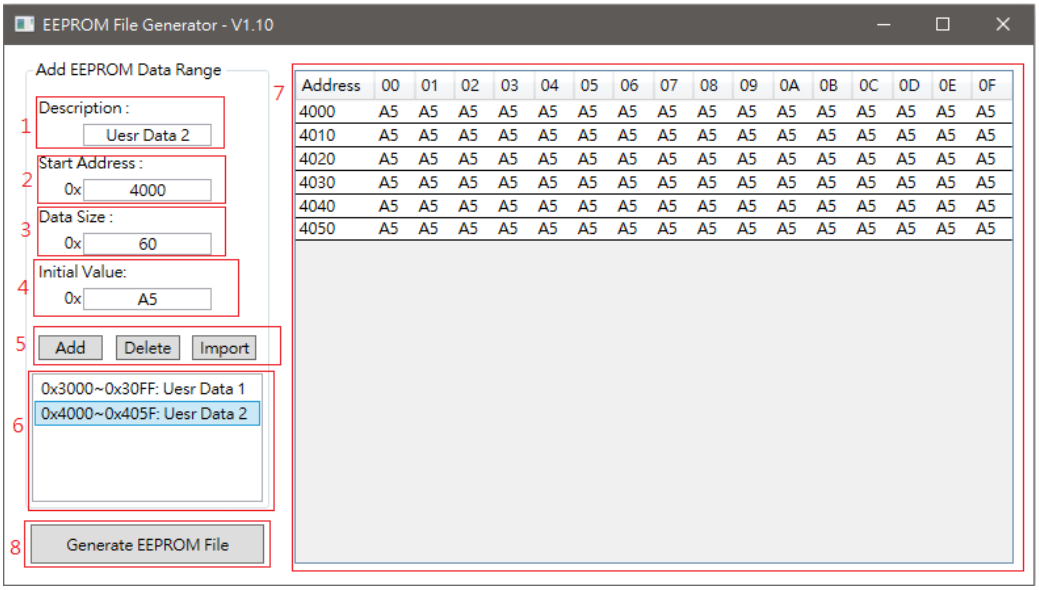
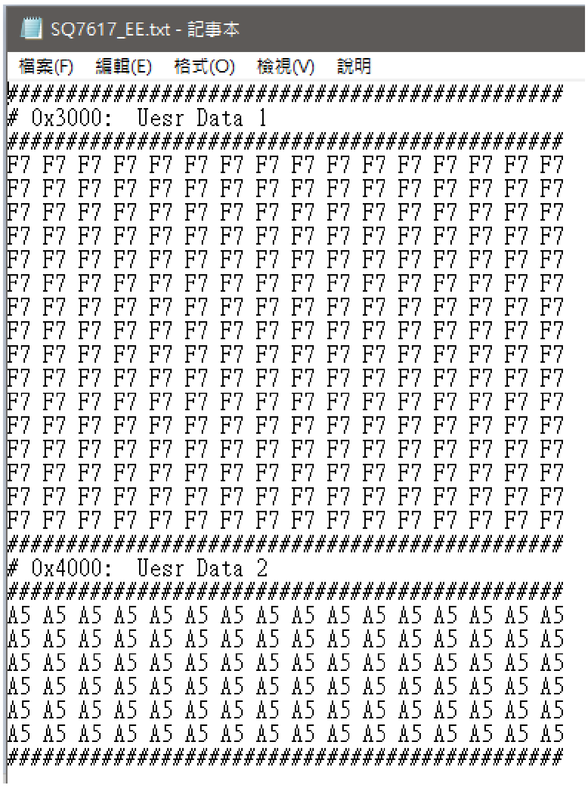


Figure 6.30 EEPROM Generator tool

1. Description: section area description
2. Start Address: section starting address(must 16 bytes alignment)
3. Data Size: section size(must multiple of 16)
4. Initial value: section initial value
5. Add, Delate, Import: add/delete a section. It can add multiple sections in one EEPROM file.
6. Section display area.
7. Section editor area: It can edit section content.
8. Generate EEPROM file: generate single/multiple section data into EEPROM file.

The generated EEPROM file is as Figure 6.31.



SQ7617_EE.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
################################################
# 0x3000:  Uesr Data 1
################################################
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7 F7
################################################
# 0x4000:  Uesr Data 2
################################################
A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5
A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5
A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5
A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5
A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5
A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5 A5
################################################
```

Figure 6.31 Multiple section EEPROM File

7. iMQ i87-IDE Error Message

pop-up window error message	description
Empty Hex File or Invalid Hex Data on 0xfffa, 0xfffb. 0xffff expected.	1. Code compilation failed 2. 0xfffa and 0xfffb in the code are values other than 0xff
Program Device Failed. Cannot enter debug mode.	1. The debugger failed to connect to the target chip 2. The programming code failed
Fail download.	Programming code failed
Program Fail File CheckSum: "Number" Chip CheckSum: "Number"	After programming, the CHECKSUM comparison failed
Load h16 file failed. Exit PROM Mode.	Failed to read the programed file
Enter PROM Mode failed.	The debugger failed to connect to the target chip
Error occur while sending PSW to OCD board!	Password transmission error before entering debug mode
Can't find Password from Hex file	The code file is not generated the debug password, and there is no manual setting
Enter debug mode failed!	Failed to connect to debug mode
Can't Find IC Serail NO. in "Chip Type".ini	The IDE software does not support the chip name in the project file
Unsupported CHIP	The IDE software does not support the chip name in the project file
CopyFile "filename" failed	1. When saving the project, the code file is missing. 2. When creating a new project, the example code file is missing. 3. When creating a new project, the target address is above 2 subdirectories.
The file had existed in this project already!	When adding a code file, a file with the same path and name already exists
Project IC Type and .Cfg file mismatch.	The configuration file (.CFG) does not match the chip of the project file
Fail to load dll file: "file name", Please make sure that it is exist!	The dynamic library file in the bin folder is missing
Only support one event/data breakpoint.	This chip only supports 1 data breakpoint
Read Info Table Fail!	A data inconsistency error occurred when reading the chip info table
Info Table fail before download!	An info table exception is detected before

	programming the chip
Info Table fail After download!	An info table exception is detected after programming the chip
Trim code corrupted!	After programming the chip, the Trim Code is lost
Trim Code CRC Check Fail!	Chip Trim Code Abnormal

8. Other Supplementary Notes

8.1 Lightweight Adjustment

When debugging, if the general register window and the special register window are not opened, all the memory of the MCU will not be read out during debugging.

In this way, when automatic single-stepping is performed, the execution time can be saved.

8.2 Other Reference Documents

Under the path where the IDE software is installed (the installation path is customized by the user, please refer to "3.1.2. The procedure for installing i87-IDE tool software"), if the user customizes the installation directory in C:\iMQ IDE, enter C:\iMQ IDE \tools\i87\manual, you can see the following three files:

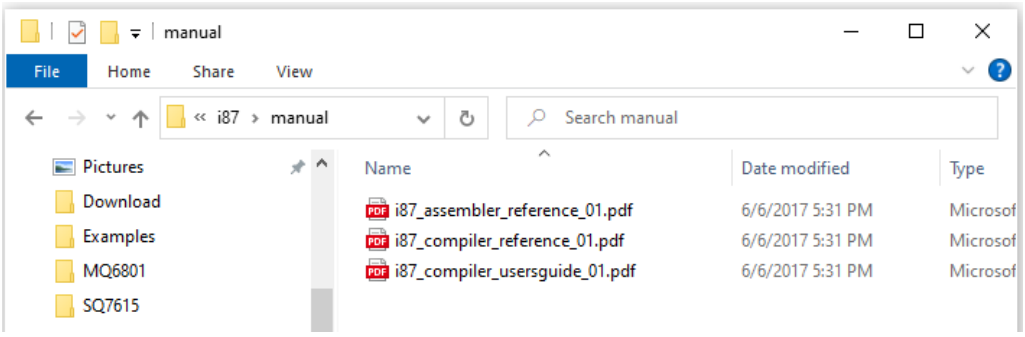


Figure 8.1 Other reference files in the IDE software installation subdirectory

The contents of these three documents are briefly described as follows, and users can refer to them by themselves:

file name	Description of content
i87_assembler_reference_01.pdf	Explain the assembly syntax and the relevant .lst file content after C language compilation
i87_compiler_reference_01.pdf	Meaning of Error Messages During Compilation
i87_compiler_usersguide_01.pdf	Explain the process of compiling the program, and related files

8.3 MQ-Link Hardware Signal Voltage Switching 5V/3.3V

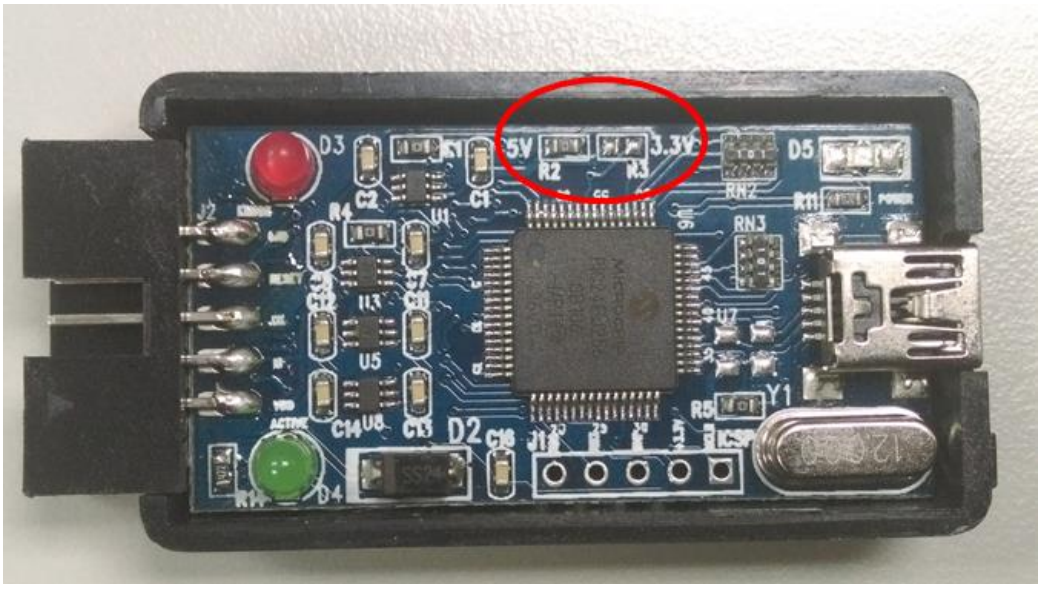


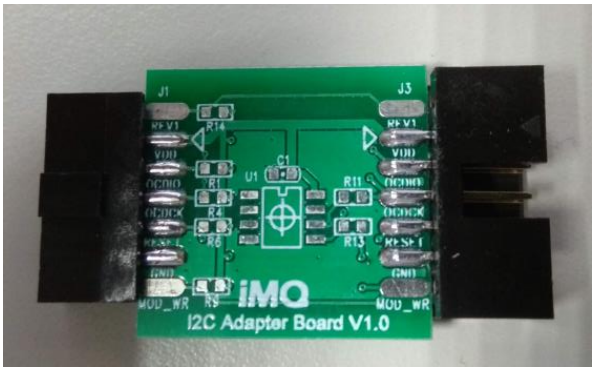
Figure 8.2 Diagram on MQ-Link PCBA

Remove the MQLink shell, solder a 0-ohm resistor to R2 or R3, and set the MQLink signal voltage to 5V or 3.3V

MQLink signal voltage	R2	R3
5V	Solder 0-ohm resistor	open circuit
3.3V	open circuit	Solder 0-ohm resistor

8.4 Preparations Before SQ7617 EEPROM Write

SQ7617 allows external programming of EEPROM. At this time, both Pin12 and Pin13 of SQ7617 must be connected to Vcc with a pull-up resistor of 4.7~10K. This pull-up resistor can be added by the user or provided by the I2C Adapter Board (as shown in the figure below -- provided by iMQ).



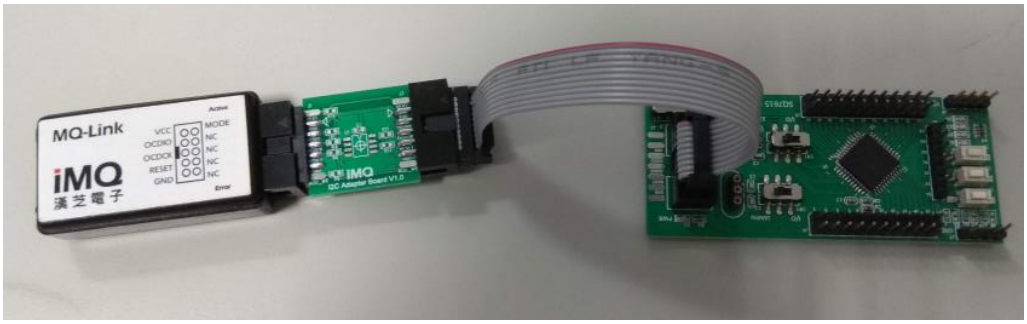


Figure 8.3 I2C Adapter Board and EVB connection diagram

Pin6 and pin8 of MQ-Link should be connected to pin13 (SDA) and pin12 (SCL) of SQ7617, and SQ7617 can also be connected through iMQ I2C Adapter Board pin6, pin8. When there is no I2C Adapter Board, you can also directly connect pin6 and pin8 of MQ-Link to pin13 (SDA) and pin12 (SCL) of SQ7617.

Note: EEPROM programming function, refer to " 6.5 EEPROM Writer " setting for details

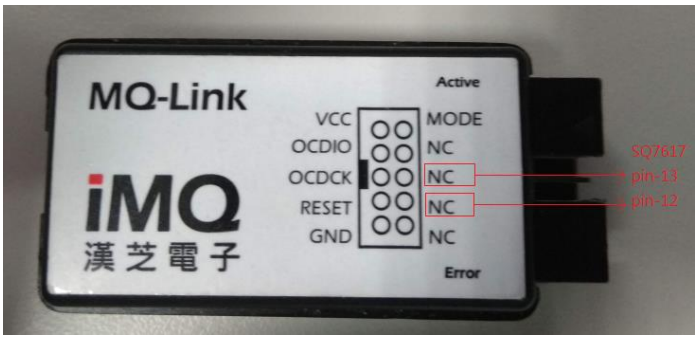


Figure 8.4 Connect MQ-Link Pin6, pin8 to SQ7617pin13 and pin12