

# **Megawin**

# **8051 OCD ICE**

## **User Manual**

## Contents

1	Introduction .....	3
	Features .....	3
	Description .....	3
2	Hardware Setup .....	4
3	Software Setup .....	5
3.1	Install the USB Device Driver for the ICE Adapter .....	5
3.2	Install the Megawin 8051 Database in the Keil 8051 IDE Software .....	5
4	Keil IDE Setup .....	6
4.1	Options- Device .....	7
4.2	Options- Target .....	7
4.3	Options- Output .....	8
4.4	Options- C51 .....	8
4.5	Options- Debug .....	9
4.6	Options- Utilities .....	10
5	Start Debugging .....	11
5.1	Activate the <i>dScope</i> -Debugger Function .....	11
5.2	Introduction to the Debugger Environment .....	12
5.2.1	Reset/Run/Halt/Step/Run-to-Cursor .....	13
5.2.2	Source-Level Debugging .....	13
5.2.3	Breakpoint Setting .....	14
5.2.4	View/Edit the Contents of Peripherals' SFRs .....	15
5.2.5	View- Disassembly Window .....	16
5.2.6	View- Watch Window .....	17
5.2.7	View- Memory Window .....	18
6	Tools, Megawin ICP .....	19
6.1	About ICP .....	19
6.2	Use ICP .....	19
6.2.1	Update Programmer .....	20
6.2.2	Update Target .....	25
7	Special Notes .....	26
7.1	Register Definition Files .....	26
7.2	On-chip XRAM and External Data Memory .....	26
7.3	Code Optimization and Source-Level Debugging .....	27
7.4	"for-Loop" and Source-Level Debugging .....	28
7.5	Hardware Option Requirements During Debugging .....	28
7.6	Error Message .....	29
7.7	Properly Connect the ICE Adapter to a Host .....	30
	Revision History .....	31

## **1 Introduction**

### **Features**

- Megawin proprietary OCD (On-Chip-Debug) technology
- On-chip & in-system real-time debugging
- Two-pin dedicated serial interface for OCD, no target resource occupied
- Directly linked to the debugger function of the *Keil 8051 IDE Software*
- USB connection between target and host (PC)
- Helpful debug actions: *Reset, Run, Stop, Step* and *Run to Cursor*
- Programmable breakpoints, up to 4 breakpoints can be inserted simultaneously
- Several debug-helpful windows: Register/Disassembly/Watch/Memory Windows
- Source-level (*Assembly* or *C-language*) debugging capability

### **Description**

The all new “Megawin 8051 OCD ICE” is a powerful development tool for 8051 embedded system. By adopting the Megawin proprietary OCD (On-Chip-Debug) technology, this ICE provides on-chip and in-system real-time debugging. The user has no need to prepare any development board during developing, or the socket adapter used in the traditional ICE probe. All the thing the user needs to do is to reserve a 6-pin connector for the dedicated OCD interface: *VCC, OCD\_SDA, OCD\_SCL, RST, CLK* and *GND*.

In addition, the most useful feature is that it can directly connect the user’s target system to the *Keil 8051 IDE software* for debugging, which directly utilizes the Keil IDE’s *dScope-Debugger* function. Of course, all the advantages are based on your using *Keil 8051 IDE software*.

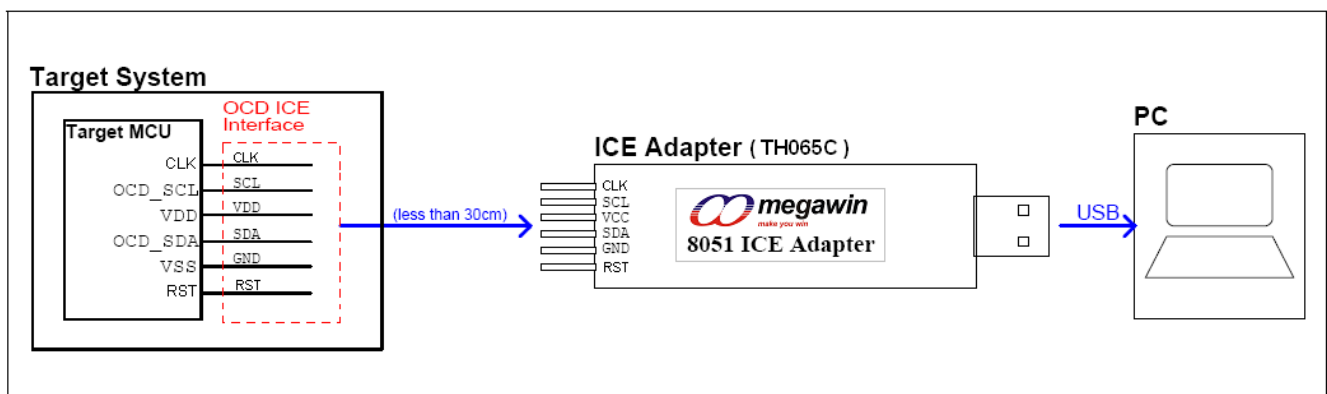
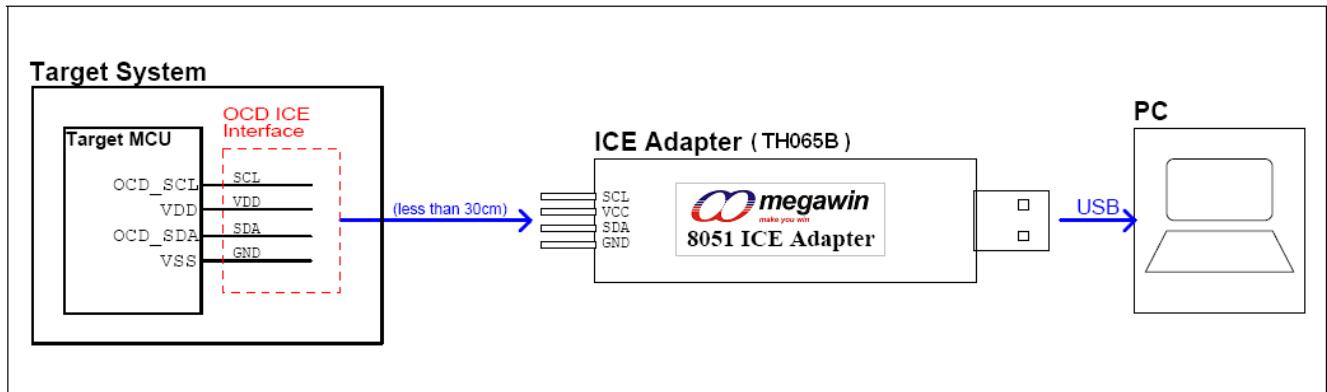
**Note:**

*“Keil” is the trade mark of “Keil Elektronik GmbH and Keil Software, Inc.”, and “Keil 8051 IDE software” is the most popular C51 compiler for 8051 embedded system development.*

## 2 Hardware Setup

For debugging, the user should connect the target system to a PC via the ICE adapter, as shown below. The ICE adapter is a bus-powered USB device, and therefore there is no need of a power adapter for it.

### Hardware Connection Diagram



Note: Refer to [Section 6.5](#) for more information.

### **3 Software Setup**

This section tell the user how to do software setup before using the OCD ICE.

#### **3.1 Install the USB Device Driver for the ICE Adapter**

The user just needs to plug the ICE adapter into any USB port in a PC. There is no need to install any device driver for the ICE adapter.

#### **3.2 Install the Megawin 8051 Database in the Keil 8051 IDE Software**

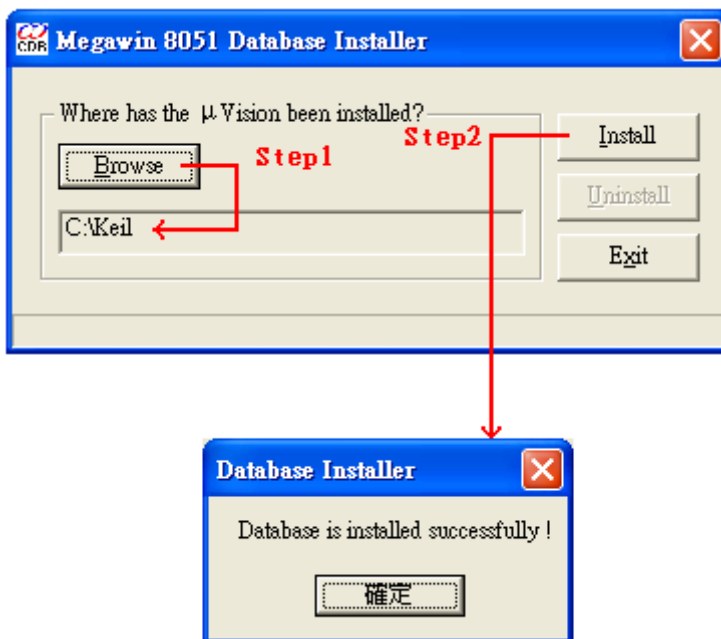
Activate the "Setup.exe" in the folder [Database Installer] to open the *Database Installer Application Program* to install the Megawin Database into the Keil 8051 IDE software. Of course, you should have installed the Keil 8051 IDE software, either  $\mu$ Vision2 or  $\mu$ Vision3, in your PC previously.

After opening the *Database Installer*, please follow the steps shown in the following GUI figure.

Step1) Click the **Browse** button to specify where the Keil software has been installed.  
(Normally, when you install the Keil 8051 IDE software, the default install-path is "C:\KEIL".)

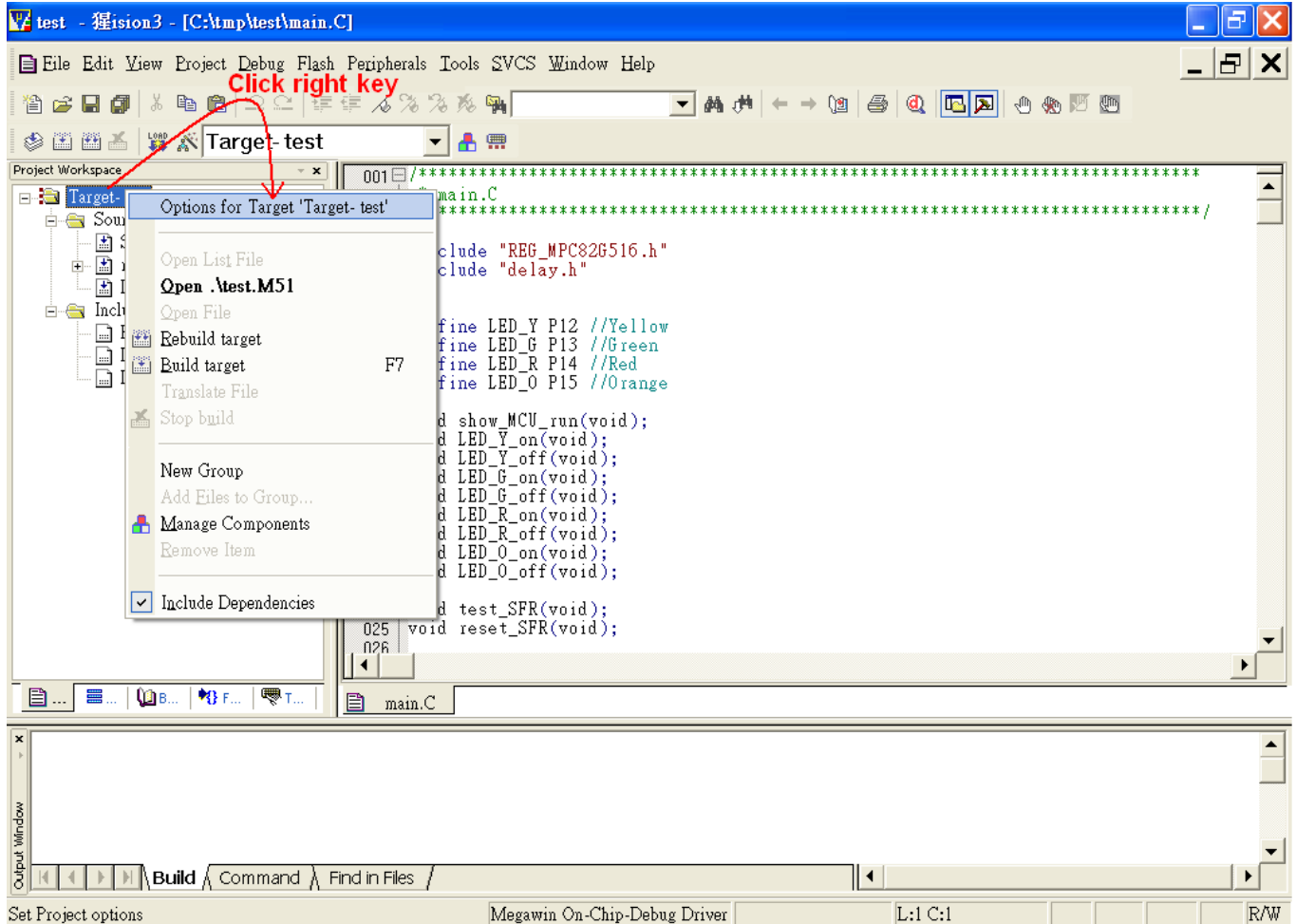
Step2) Click the **Install** button to start installing the Megawin Database into the Keil software.

GUI of the Database Installer



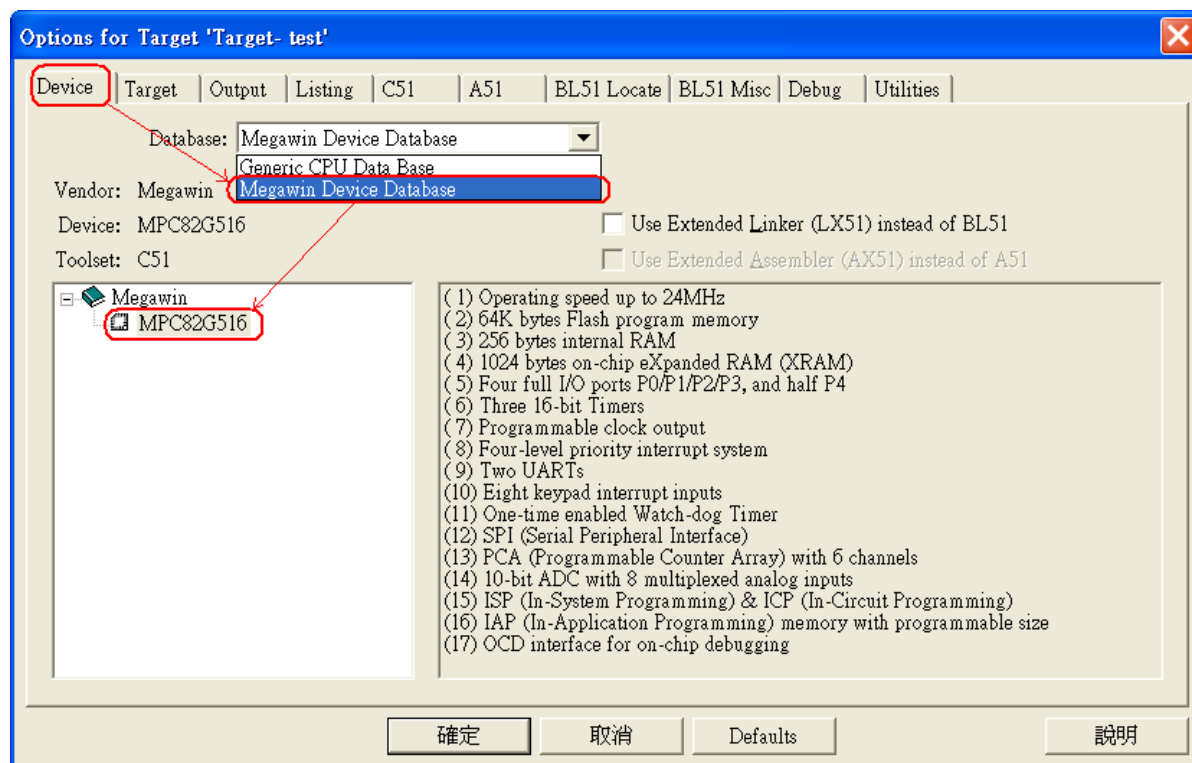
## 4 Keil IDE Setup

Before using the *dScope-Debugger* function of the Keil IDE, the user should do some proper settings in the Keil IDE. First, open the *µVision* project you would like to debug. Then, move cursor to “Target-..” and click the mouse’s right button to invoke the “**Options for Target**”, as shown below.



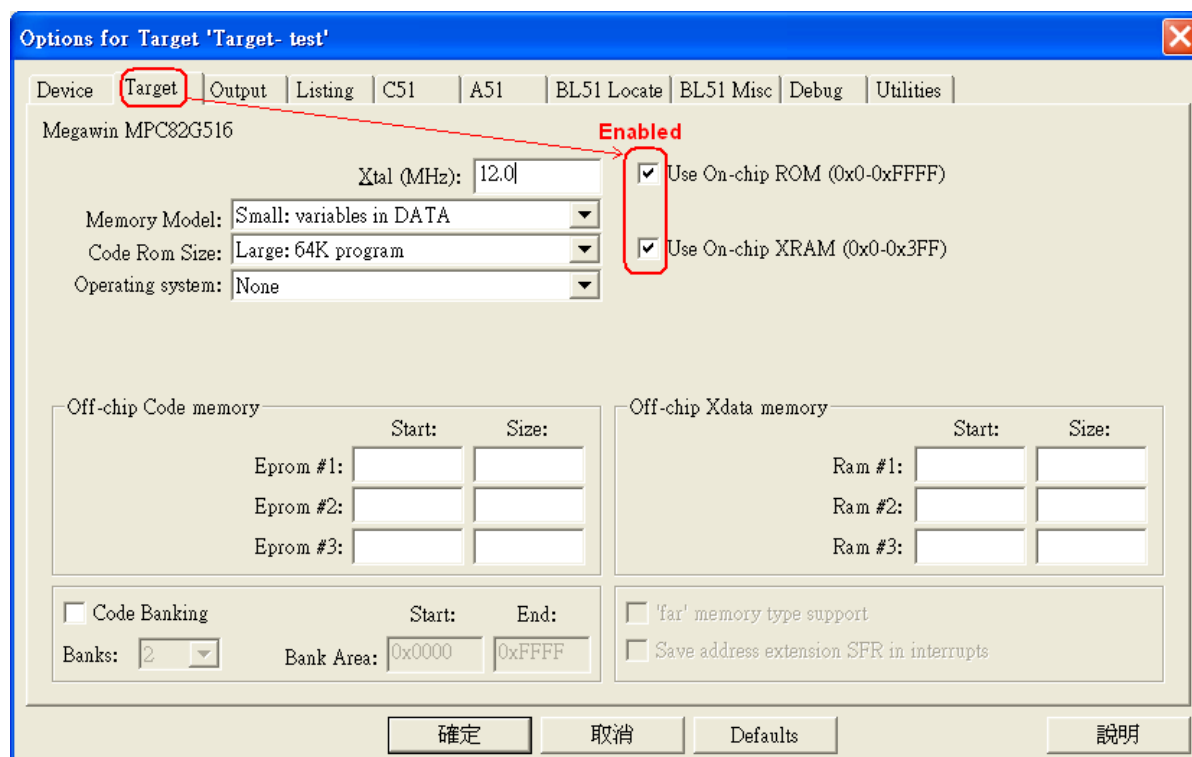
## 4.1 Options- Device

Select the “Megawin Device Database” and the target part number.



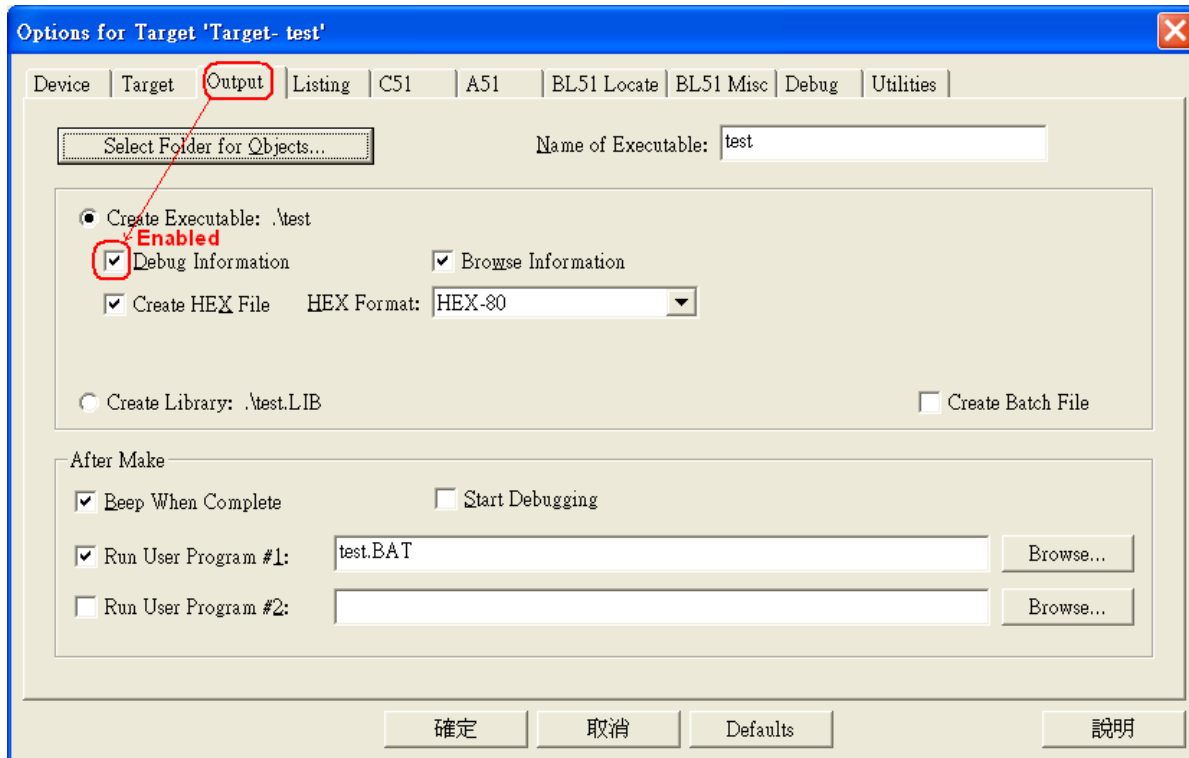
## 4.2 Options- Target

Enable the “Use on-chip ROM” and the “Use on-chip XRAM”.



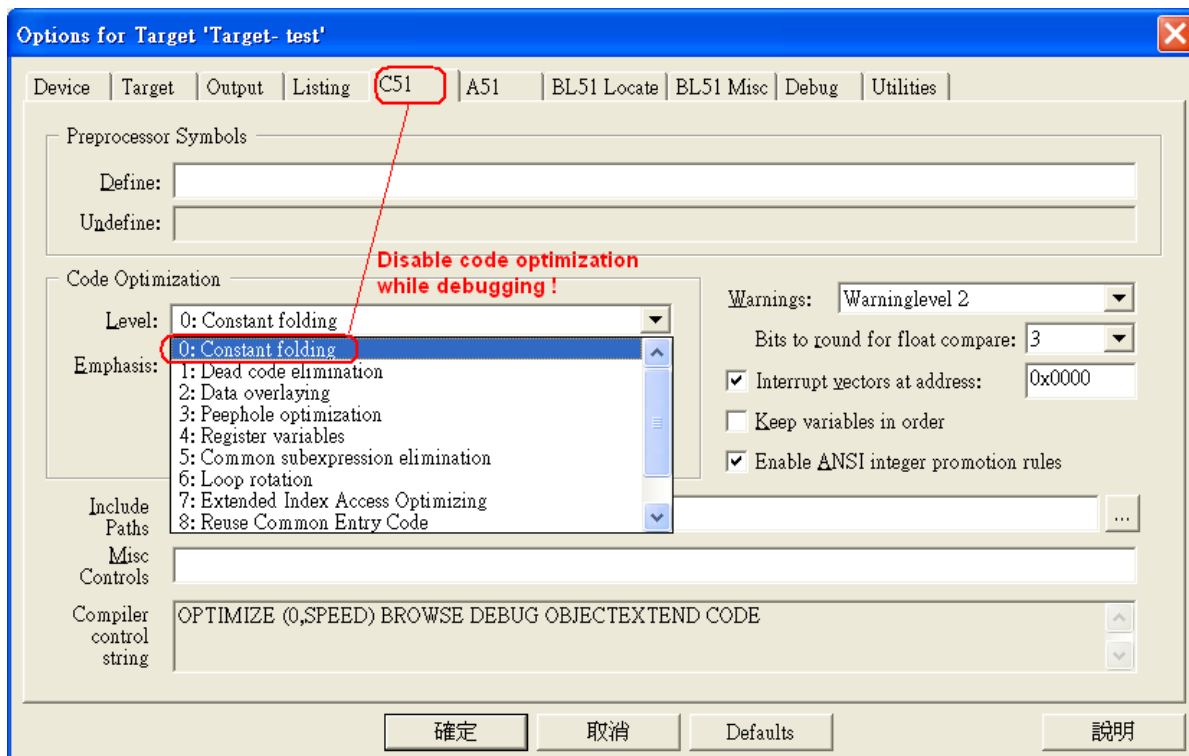
### 4.3 Options- Output

Enable the “Debug Information”. It is necessary for creating an absolute OMF file for source-level debugging.



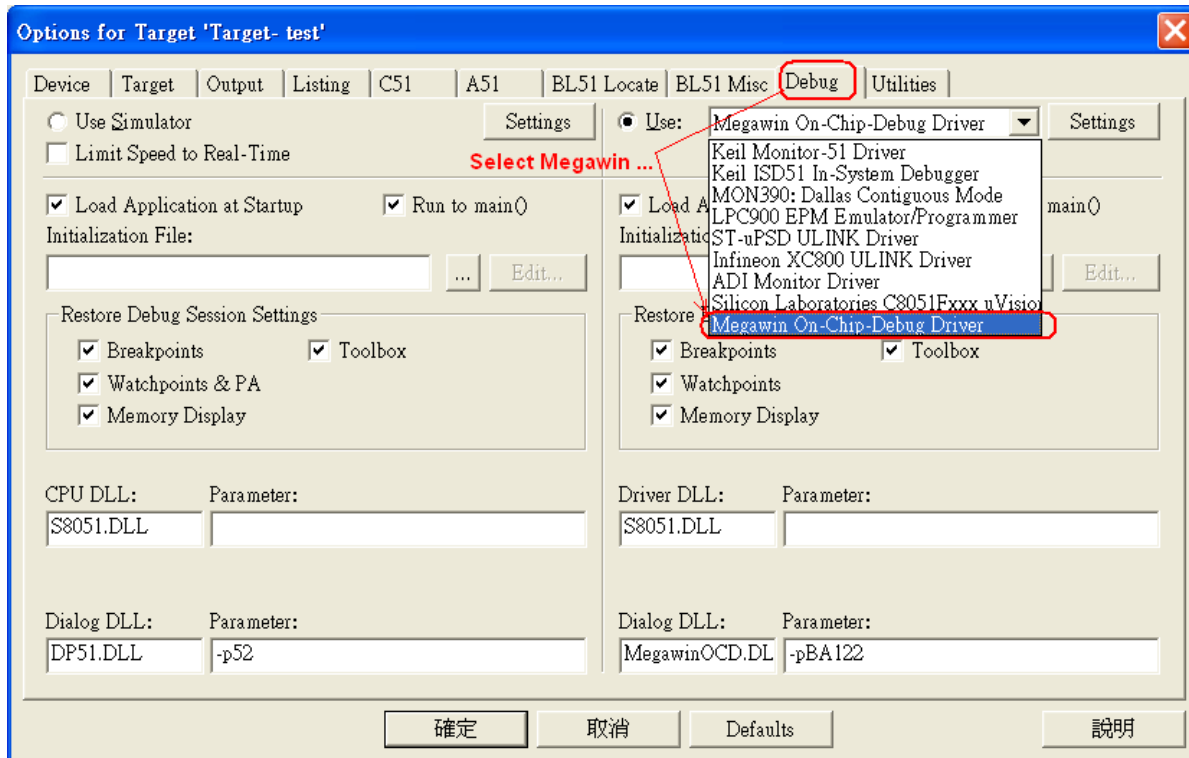
### 4.4 Options- C51

Disable the code optimization by selecting “Level 0: Constant folding”. Refer to [Section 6.3](#) for more information about this setting. *Note: This setting is optional.*

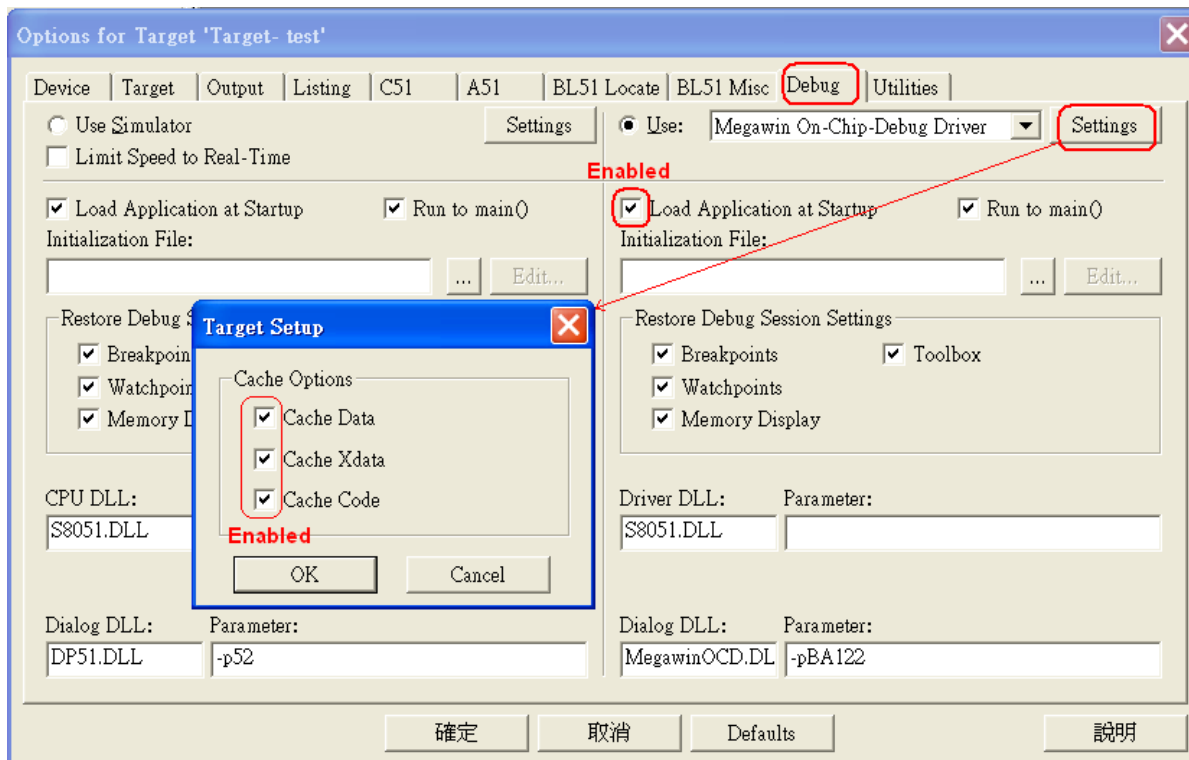


## 4.5 Options- Debug

Select the “Megawin On-Chip-Debug Driver”.



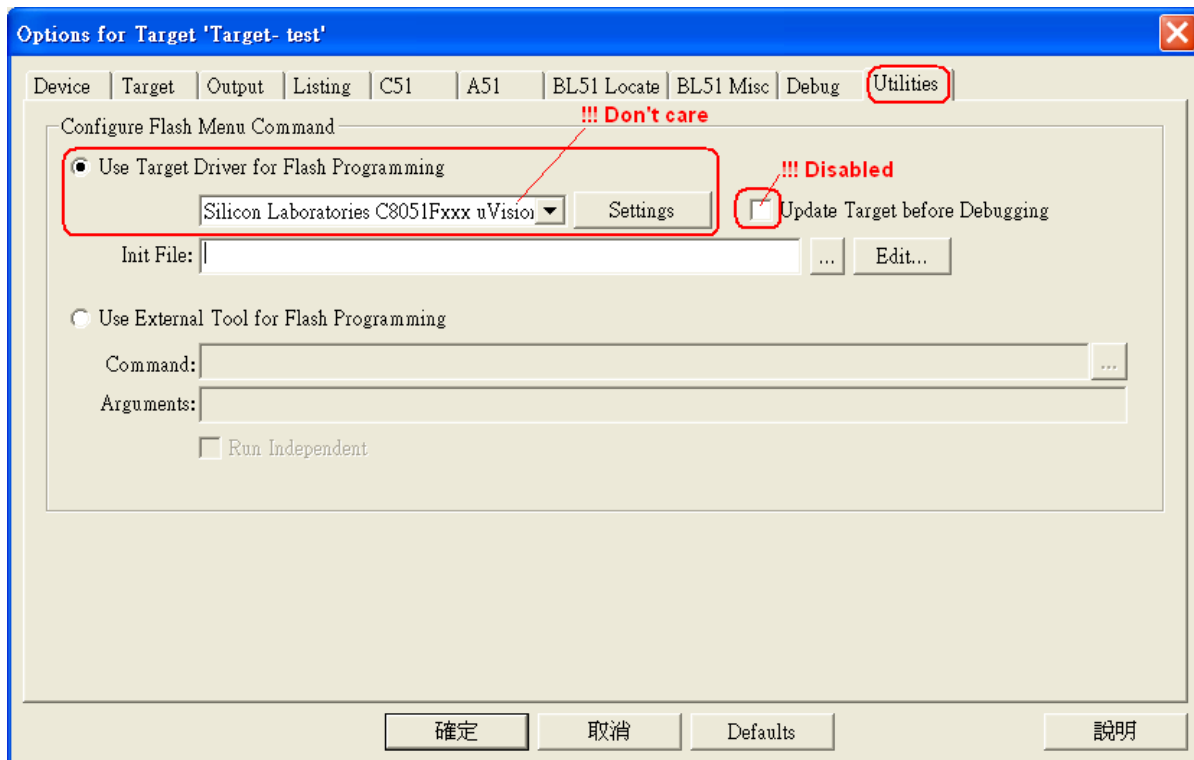
And, enable the “Load Application at Startup” and all the Cache Options.



## 4.6 Options- Utilities

Always disable the “Update Target before Debugging”. It is because we have enabled the “Load Application at Startup” shown in [Section 4.5](#). And, leave the “Use Target Driver for Flash Programming” *don't-care*.

Note:  $\mu$ Vision2 doesn't have this selection item.

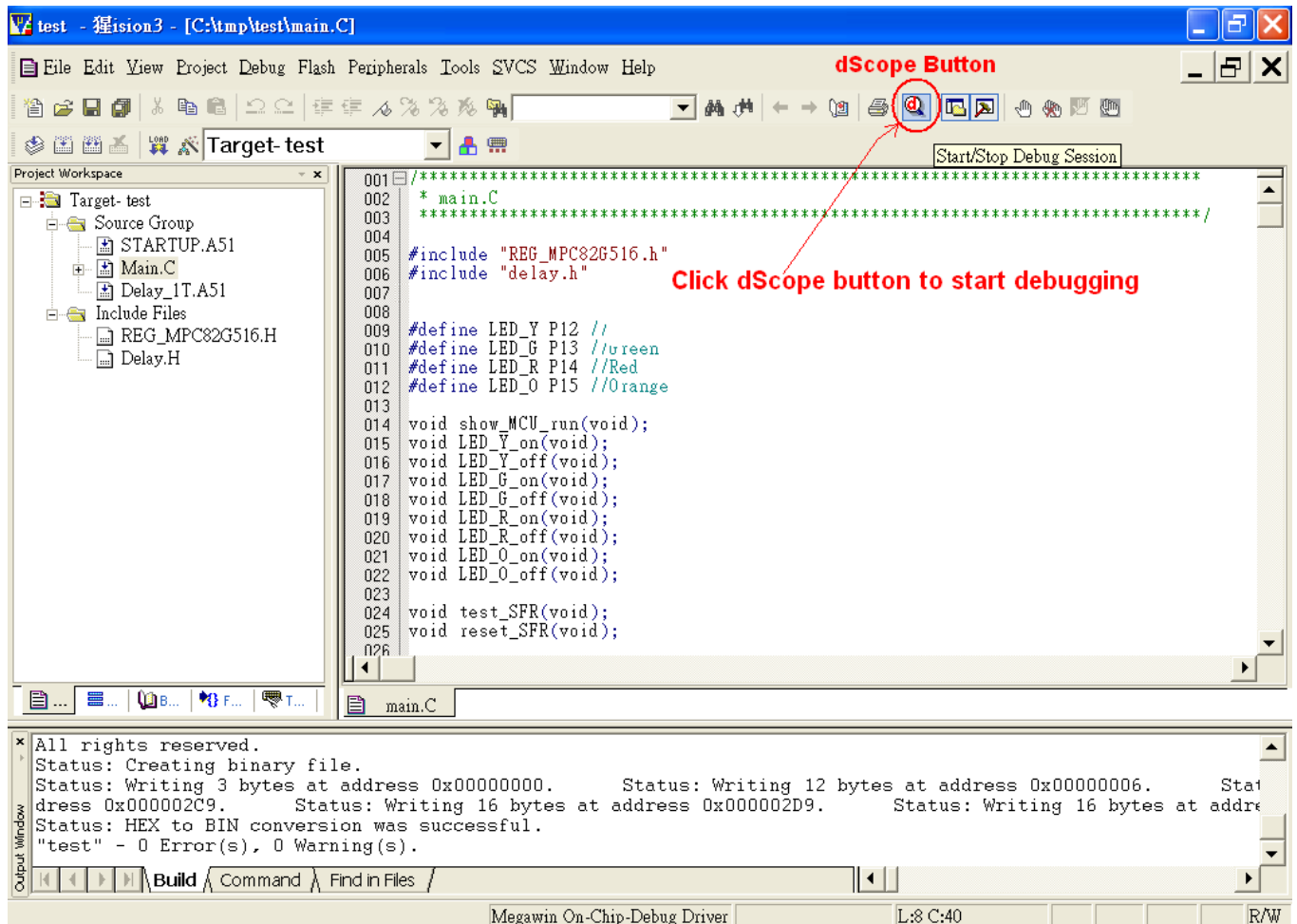


## 5 Start Debugging

After the tasks described in *Sections 2, 3 and 4* have been done, you can start debugging your  $\mu$ Vision project.

### 5.1 Activate the dScope-Debugger Function

After building the project (suppose no error), you can enter the Keil IDE's debugger mode by clicking the *dScope* button, as shown below. Now, the project code will be automatically downloaded into the target's Flash. It will take some time.



## 5.2 Introduction to the Debugger Environment

There are four basic windows regarding the debugging operation in the debugger environment. They are Register Window, Disassembly Window, Watch Window and Memory Window, as described below.

### Register Window

This window shows the contents of the current register bank (R0~R7), the system registers (A, B, SP, DTPR and the Program Counter) and the Program Status Word (PSW). The register with blue background means its content is just changed due to the instruction just executed.

### Disassembly Window

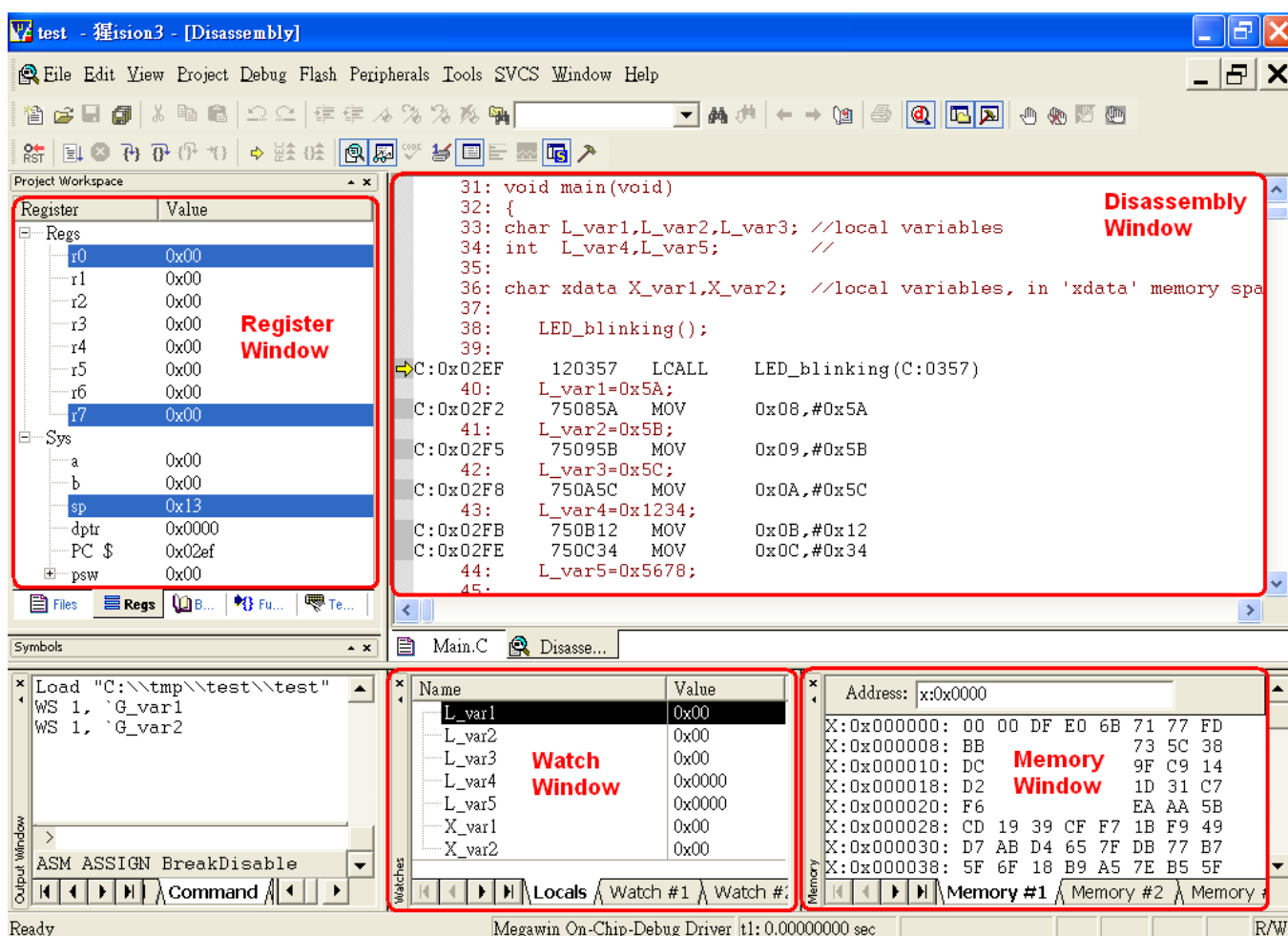
This window is the default window opened just when the debugger mode is entered. It shows the source-level code followed by its corresponding assembly code.

### Watch Window

This window automatically shows the local variables when **Locals** is clicked. The local variables are the variables declared within a function including the main() function. To view the global variables, click **Watch #1** or **Watch #2** and type <F2> key to edit and enter the variable name. The variable with blue background means its content is just changed due to the instruction just executed.

### Memory Window

This window shows the contents of the memory located at the **data/idata/xdata/code** memory space. The available commands are: d:0x00~d:0xFF, i:0x00~i:0xFF, x:0x0000~x:0xFFFF and c:0x0000~c:0xFFFF. The user can view any of the four memory by entering the corresponding command.



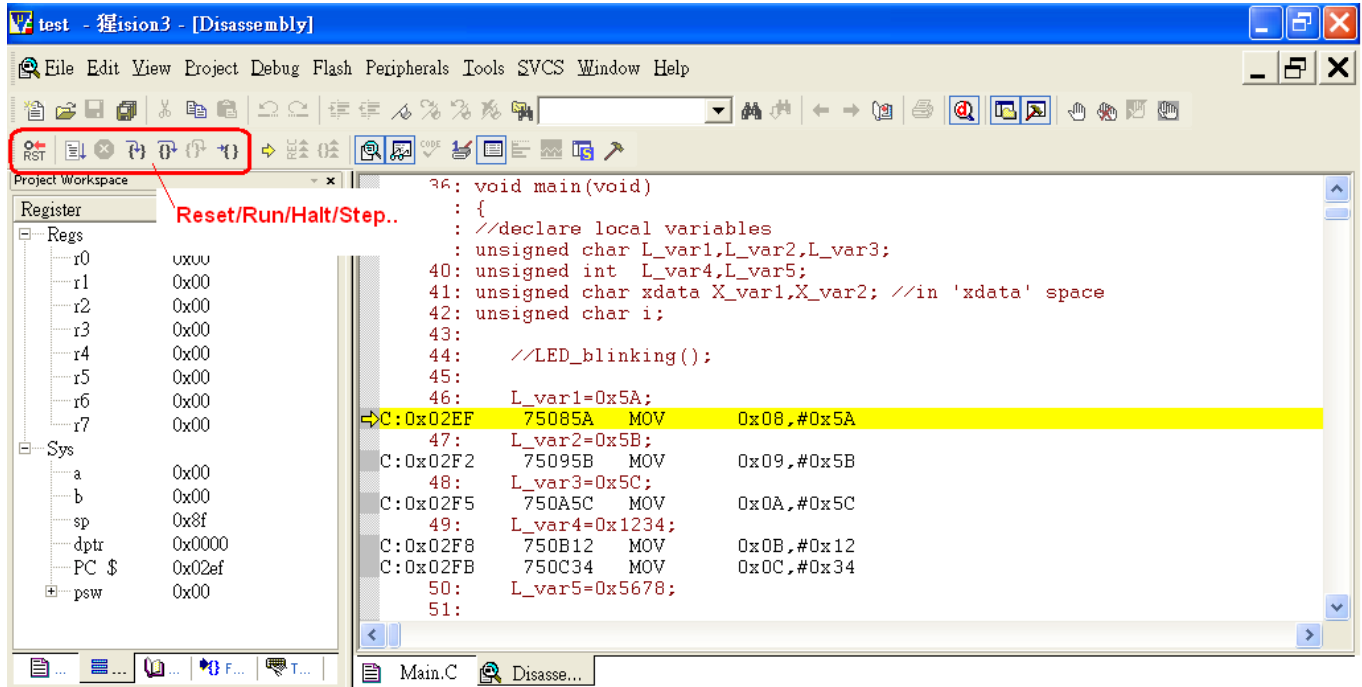
The screenshot displays the Megawin On-Chip-Debug Driver interface with the following windows:

- Register Window:** Shows the contents of registers r0-r7, Sys (a, b, sp, dptr, PC, PSW). The 'sp' register is highlighted with a blue background.
- Disassembly Window:** Shows source code and assembly instructions. The assembly instruction 'LED\_blinking(C:0357)' is highlighted.
- Watch Window:** Shows local variables L\_var1 through L\_var5 and X\_var1 through X\_var2. L\_var1 is highlighted with a blue background.
- Memory Window:** Shows memory contents starting from address x:0x0000. The memory address 'x:0x0000' is highlighted.

The status bar at the bottom indicates 'Ready' and 'Megawin On-Chip-Debug Driver t1: 0.00000000 sec'.

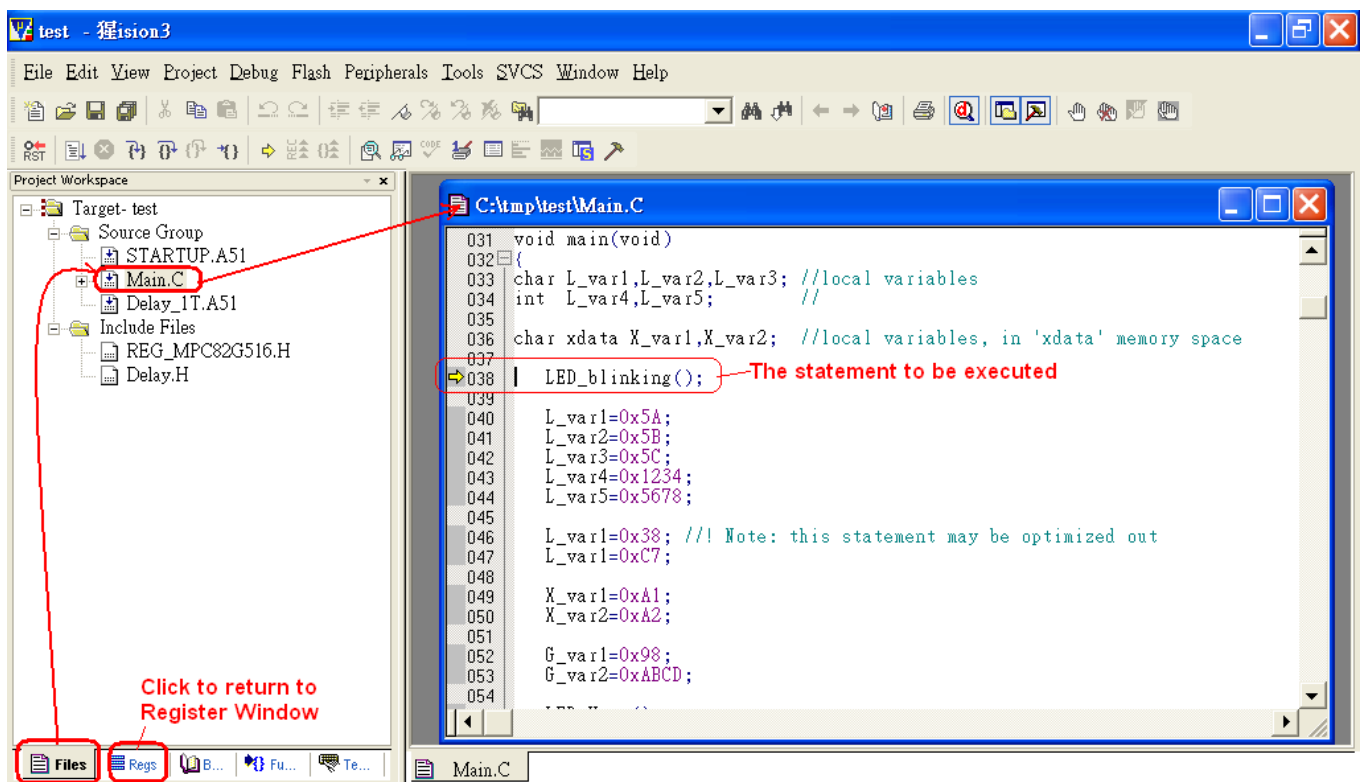
## 5.2.1 Reset/Run/Halt/Step/Run-to-Cursor

Reset, Run, Halt (Stop), Step and Run-to-Cursor are the basic debug actions. The user can easily invoke any of these actions by clicking the short-cut buttons in the debugger GUI, as shown below.



## 5.2.2 Source-Level Debugging

To do the source-level debugging, open the source file by clicking **Files** to open the Project Workspace and select the source files you want. Click **Regs** again to return to Register Window, as shown below.

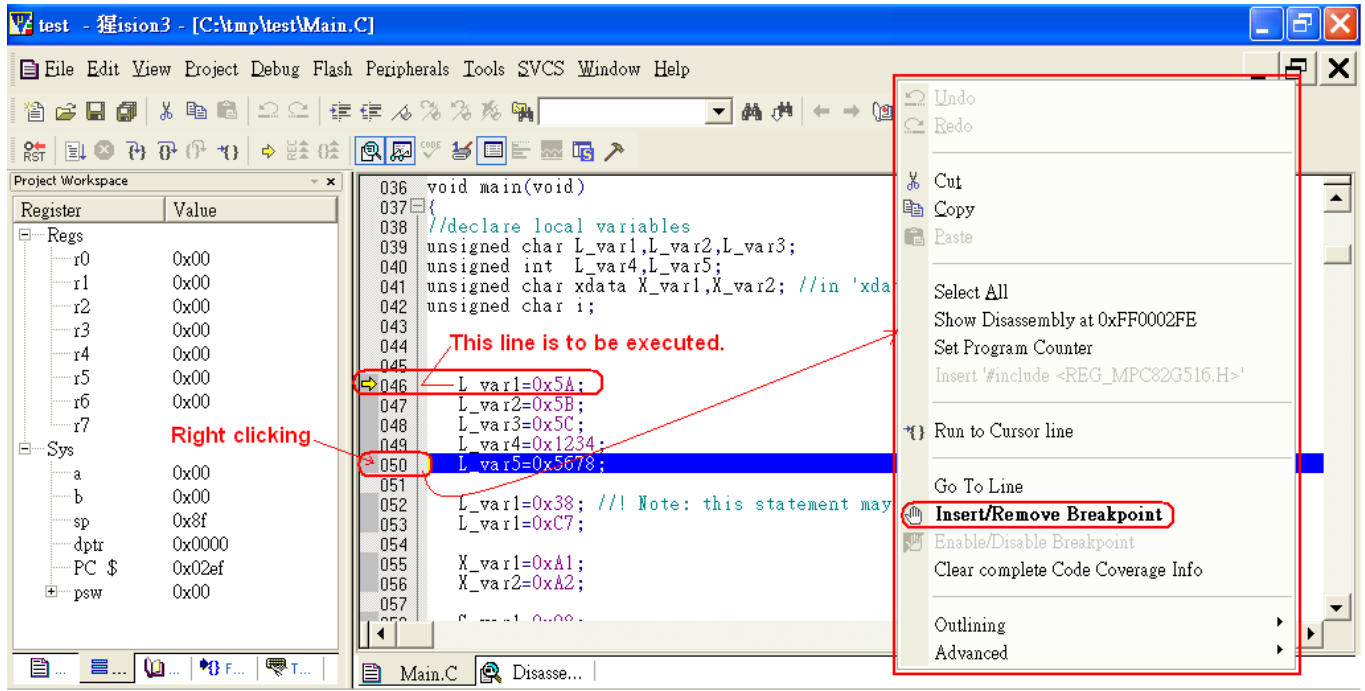


### 5.2.3 Breakpoint Setting

There are total four breakpoints available for debugging. Up to four breakpoints can be inserted simultaneously.

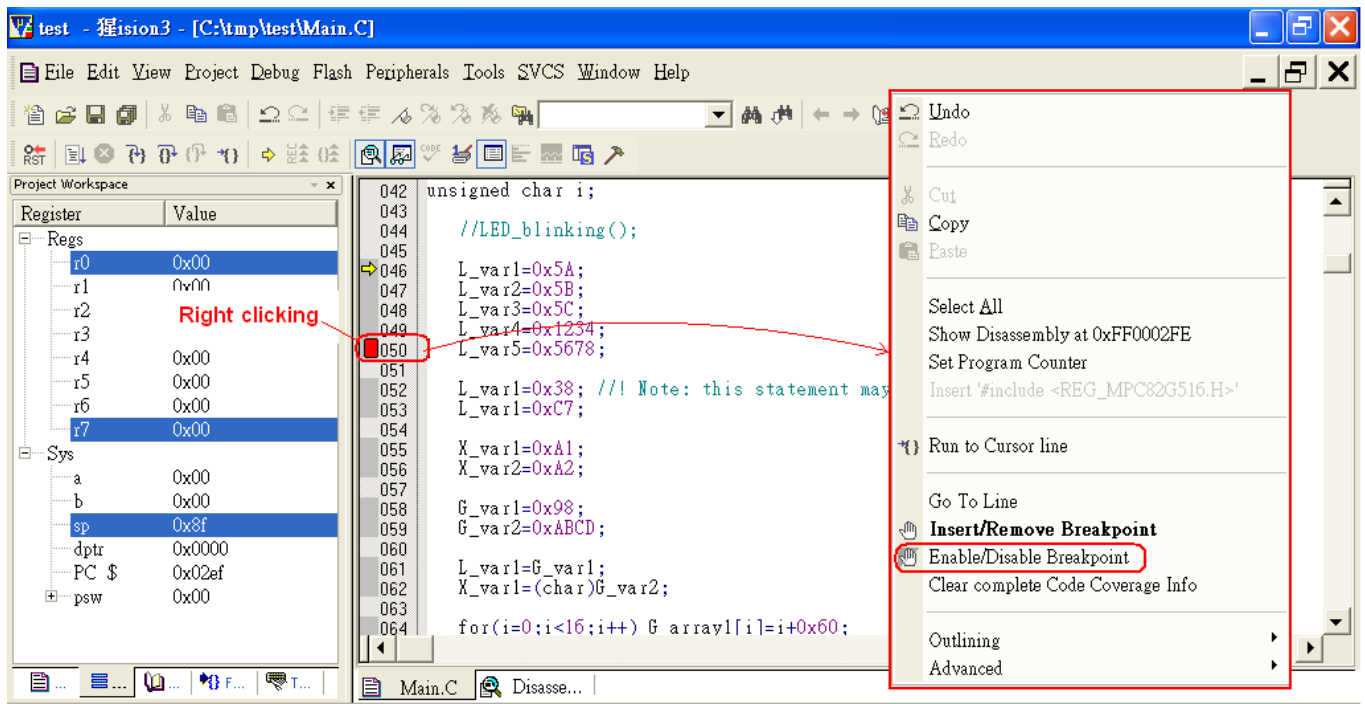
#### Insert/Remove a Breakpoint

Move the cursor to the front of the line and click the right key, then click **"Insert/Remove Breakpoint"** for toggling between Insert and Remove, as shown below.



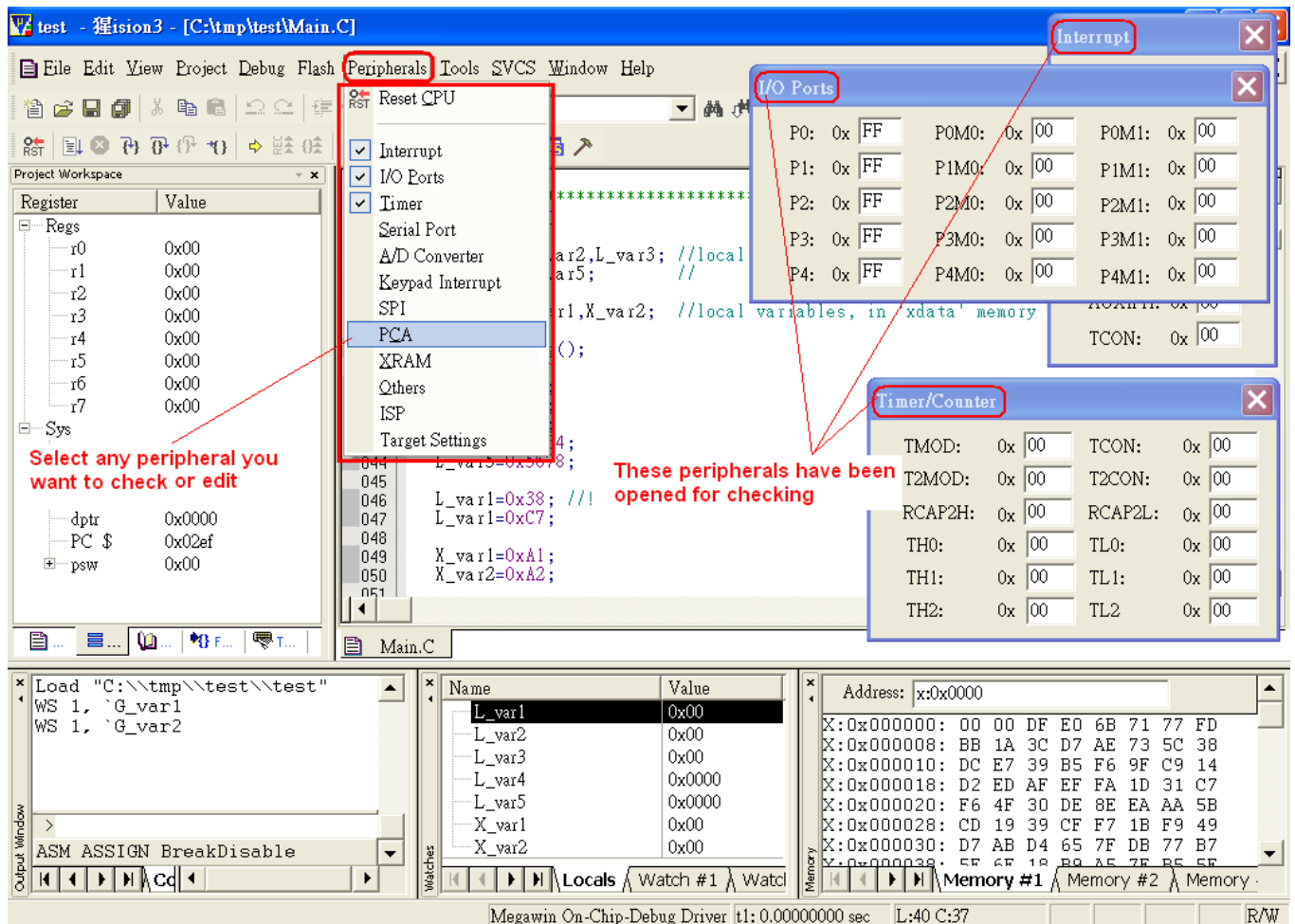
#### Enable/Disable a Breakpoint

Move the cursor to the front of the line and click the right key, then click **"Enable/Disable Breakpoint"** for toggling between Enable and Disable. Of course, this line should have been inserted a breakpoint previously.



## 5.2.4 View/Edit the Contents of Peripherals' SFRs

There are many peripheral SFRs that don't belong to the registers shown in the Register Window. To view or edit these registers, select the **Peripherals** item on the main menu. A pulled-down sub-menu will be displayed, and the user can select a peripheral to view or edit its corresponding SFRs, as shown below.



The screenshot shows the Megawin IDE interface with the **Peripherals** menu open. The menu options are: Reset CPU, Interrupt, I/O Ports, Timer, Serial Port, A/D Converter, Keypad Interrupt, SPI, PCA, XRAM, Others, ISP, and Target Settings. The **PCA** option is highlighted. Red arrows point from the text "Select any peripheral you want to check or edit" to the menu and from "These peripherals have been opened for checking" to the peripheral windows.

**Peripherals Menu:**

- Reset CPU
- Interrupt
- I/O Ports
- Timer
- Serial Port
- A/D Converter
- Keypad Interrupt
- SPI
- PCA
- XRAM
- Others
- ISP
- Target Settings

**I/O Ports Window:**

P0: 0x FF	P0M0: 0x 00	P0M1: 0x 00
P1: 0x FF	P1M0: 0x 00	P1M1: 0x 00
P2: 0x FF	P2M0: 0x 00	P2M1: 0x 00
P3: 0x FF	P3M0: 0x 00	P3M1: 0x 00
P4: 0x FF	P4M0: 0x 00	P4M1: 0x 00

**Timer/Counter Window:**

TMOD: 0x 00	TCON: 0x 00
T2MOD: 0x 00	T2CON: 0x 00
RCAP2H: 0x 00	RCAP2L: 0x 00
TH0: 0x 00	TL0: 0x 00
TH1: 0x 00	TL1: 0x 00
TH2: 0x 00	TL2: 0x 00

**Register Window:**

Register	Value
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00

**Output Window:**

```
Load "C:\\tmp\\test\\test"
WS 1, 'G_var1'
WS 1, 'G_var2'
ASM ASSIGN BreakDisable
```

**Locals Window:**

Name	Value
L_var1	0x00
L_var2	0x00
L_var3	0x00
L_var4	0x0000
L_var5	0x0000
X_var1	0x00
X_var2	0x00

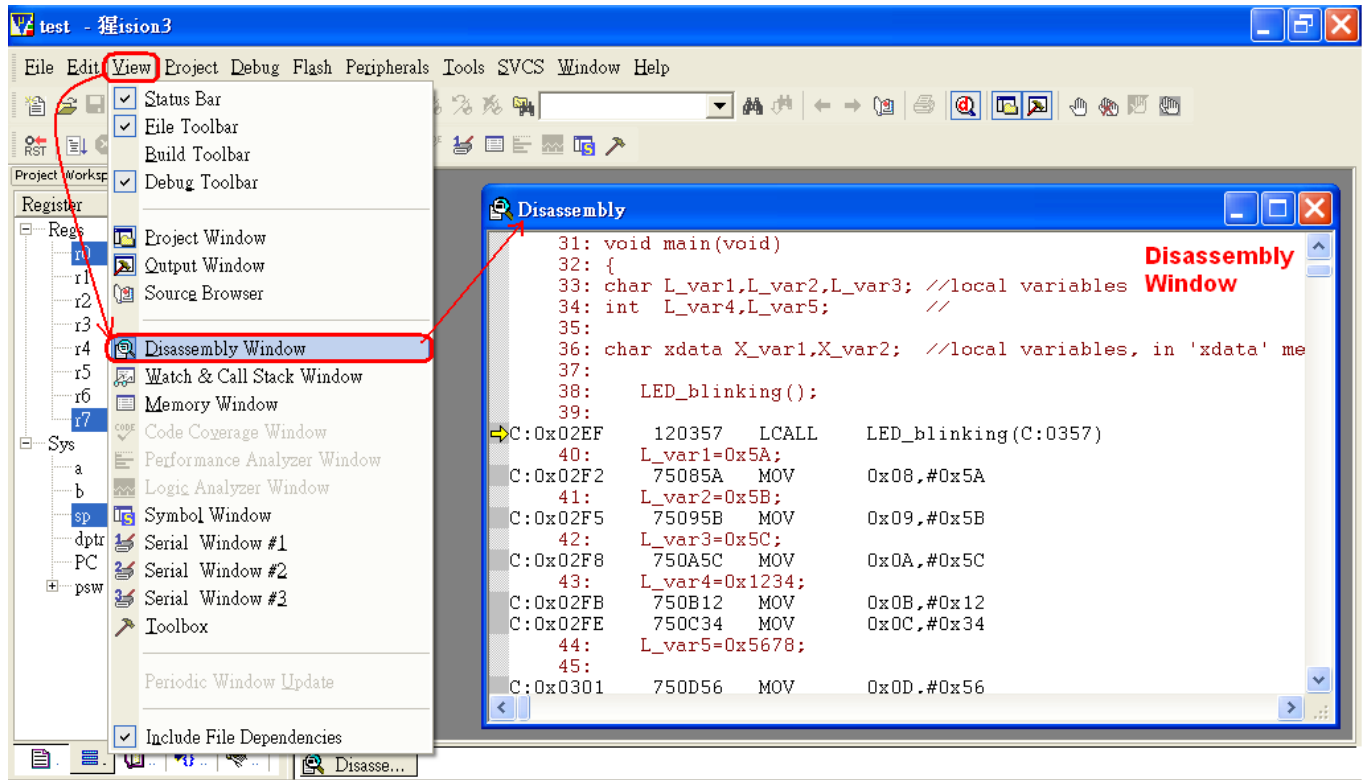
**Memory Window:**

Address	Value
0x000000	00 00 DF E0 6B 71 77 FD
0x000008	BB 1A 3C D7 AE 73 5C 38
0x000010	DC E7 39 B5 F6 9F C9 14
0x000018	D2 ED AF EF FA 1D 31 C7
0x000020	F6 4F 30 DE 8E EA AA 5B
0x000028	CD 19 39 CF F7 1B F9 49
0x000030	D7 AB D4 65 7F DB 77 B7
0x000038	5F 6F 18 B9 A5 7E B5 5F

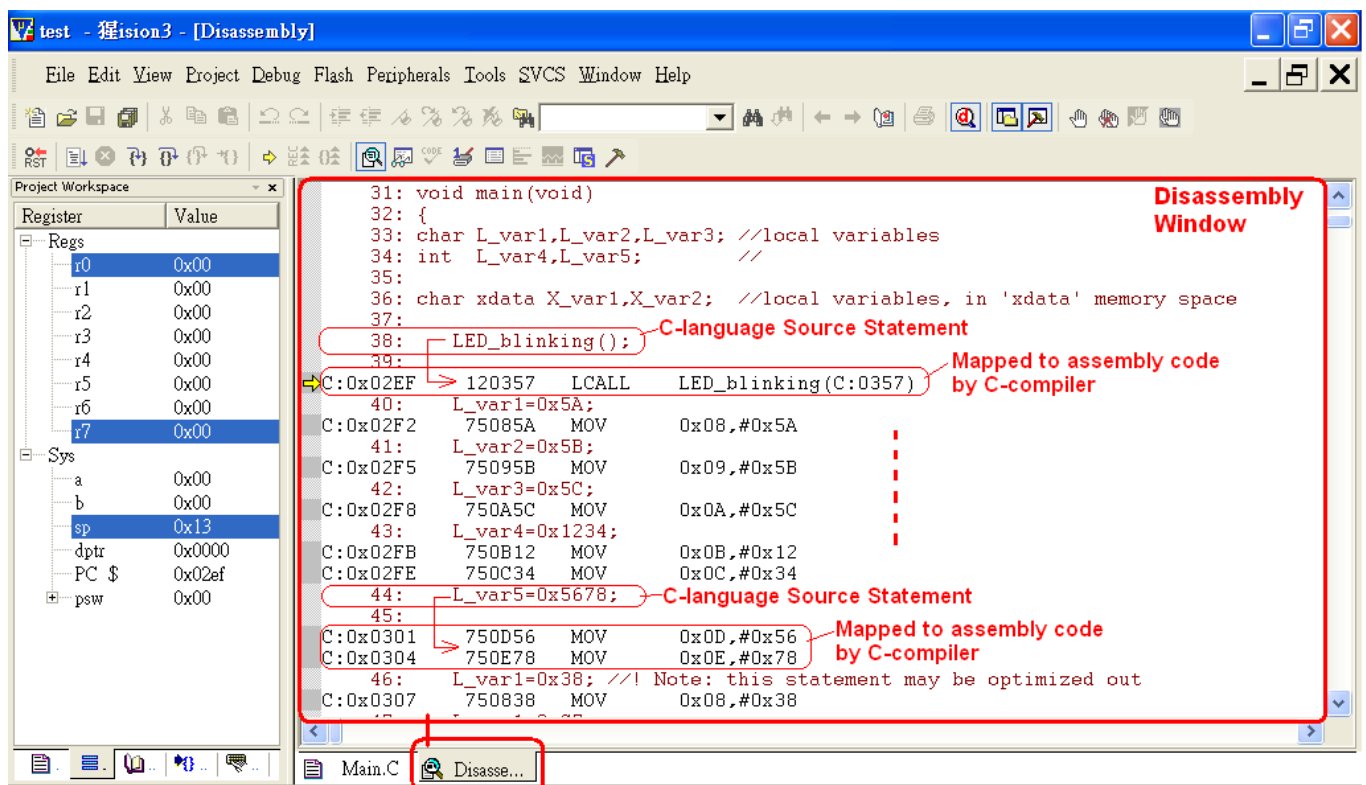
Megawin On-Chip-Debug Driver t1: 0.00000000 sec L:40 C:37 R/W

## 5.2.5 View- Disassembly Window

Disassembly Window displays source-level code followed by its corresponding assembly. To open this window, select the **View** item on the main menu. A pulled-down sub-menu will be displayed, and then select **Disassembly Window**, as shown below.

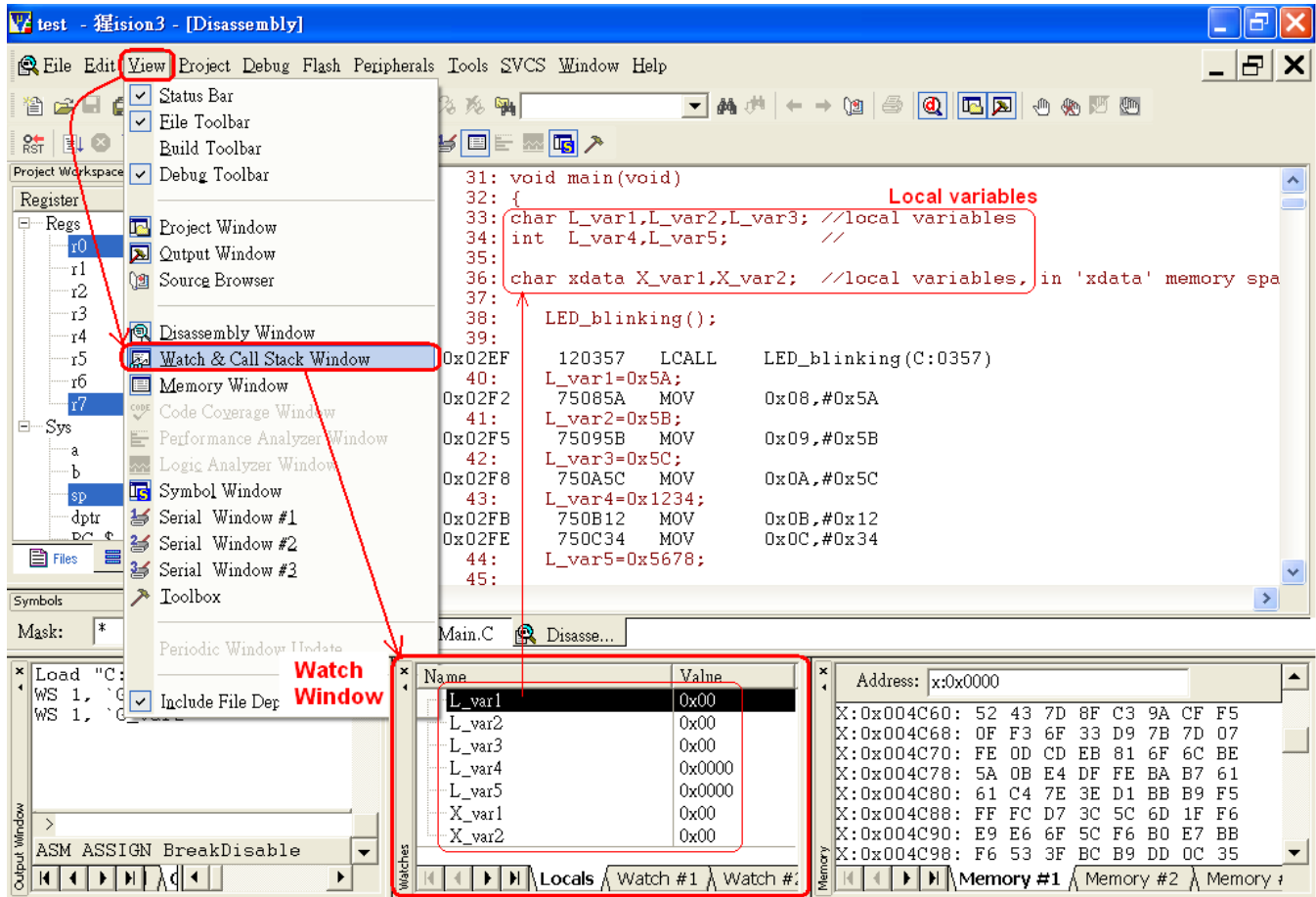


Maximize the Disassembly Window for detailed description:

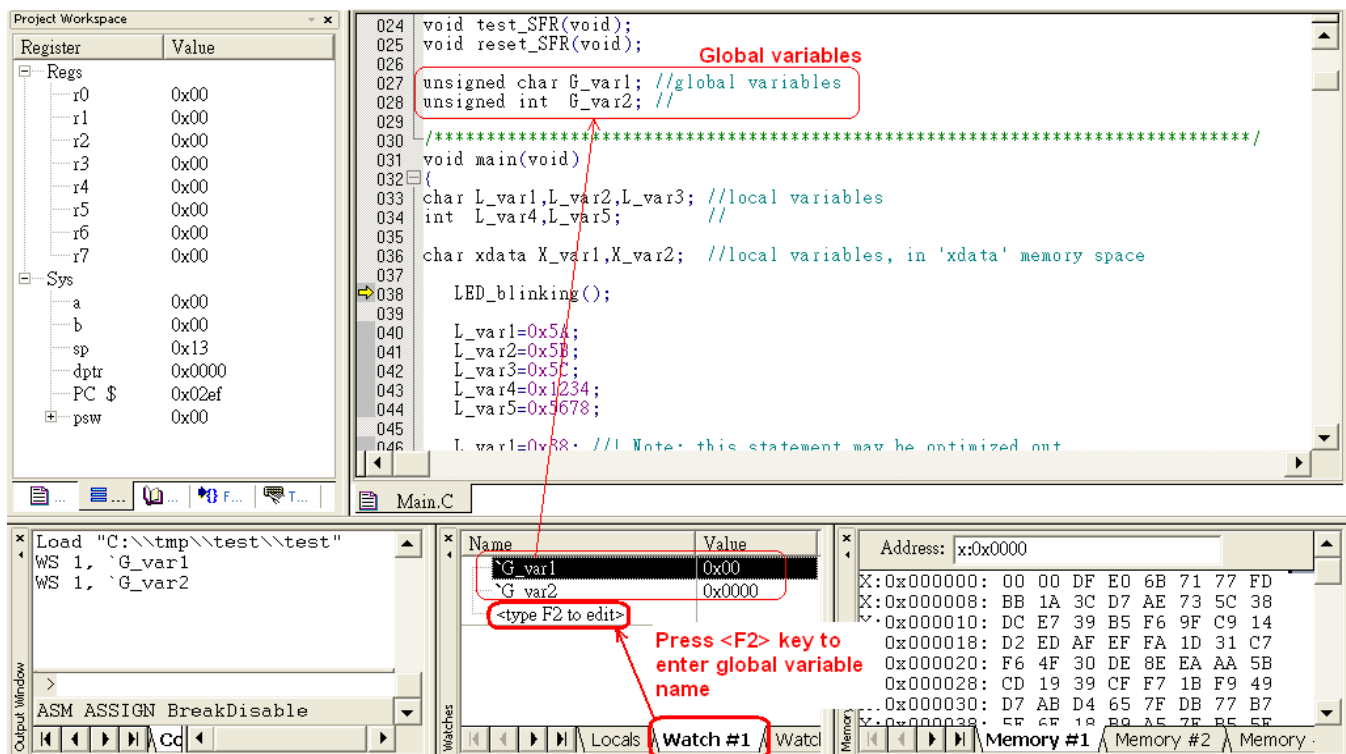


## 5.2.6 View- Watch Window

The Watch Window helps the user to check either local variables or global variables, as shown below.



To check the global variables, click **Watch #1** or **#2**, then type <F2> key to enter the variable name.

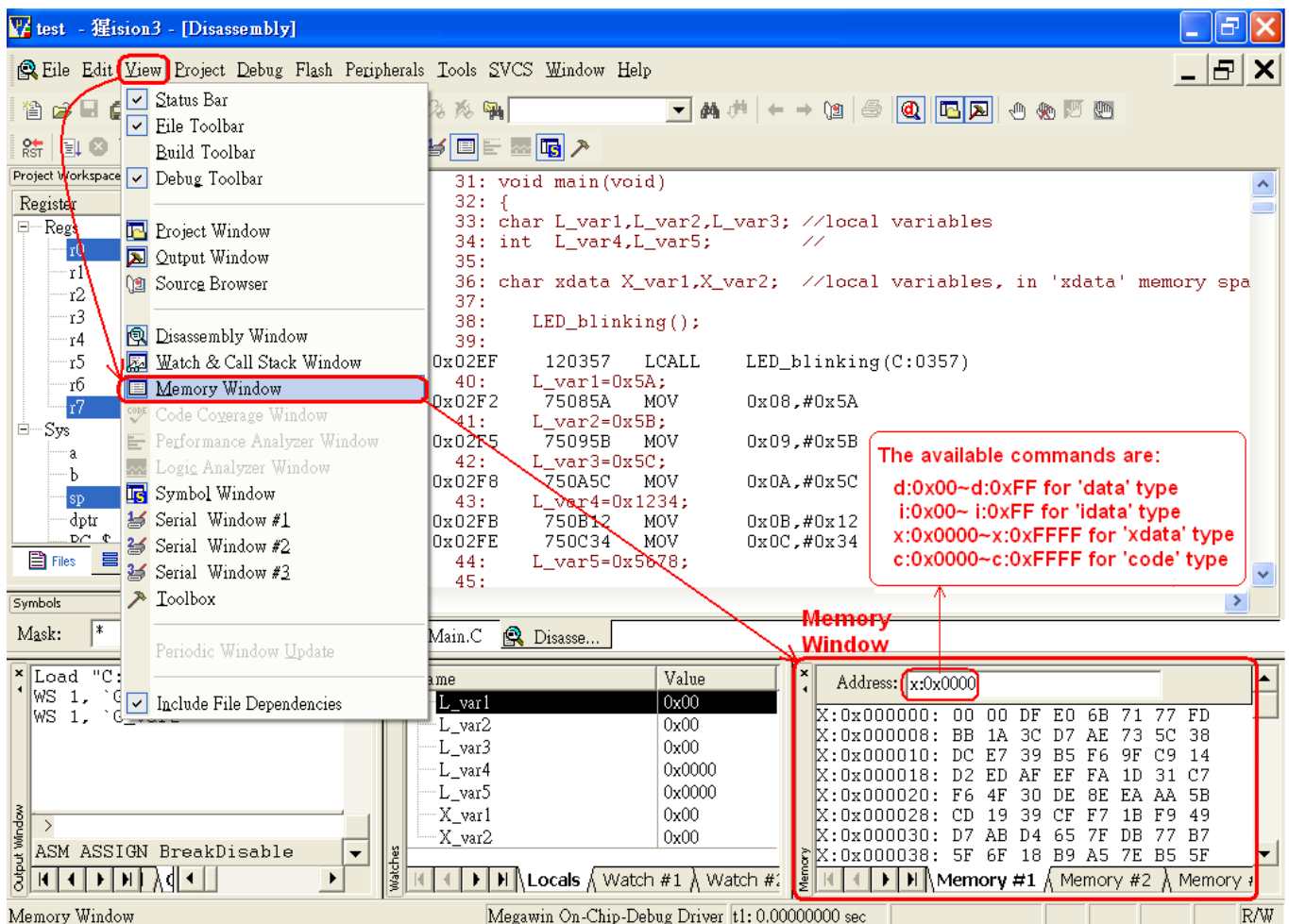


## 5.2.7 View- Memory Window

To open this window, select the **View** item on the main menu. A pulled-down sub-menu will be displayed, and then select **Memory Window**, as shown below. The available commands are:

- (1) d:0x00~d:0xFF, for 'data' type
- (2) i:0x00~i:0xFF, for 'idata' type
- (3) x:0x0000~x:0xFFFF, for 'xdata' type
- (4) c:0x0000~c:0xFFFF., for 'code' type

The user can view any of the four memory by entering the corresponding command. Refer to [Section 6.2](#) for how to display 'xdata' type variables.



## **6 Tools, Megawin ICP**

### **6.1 About ICP**

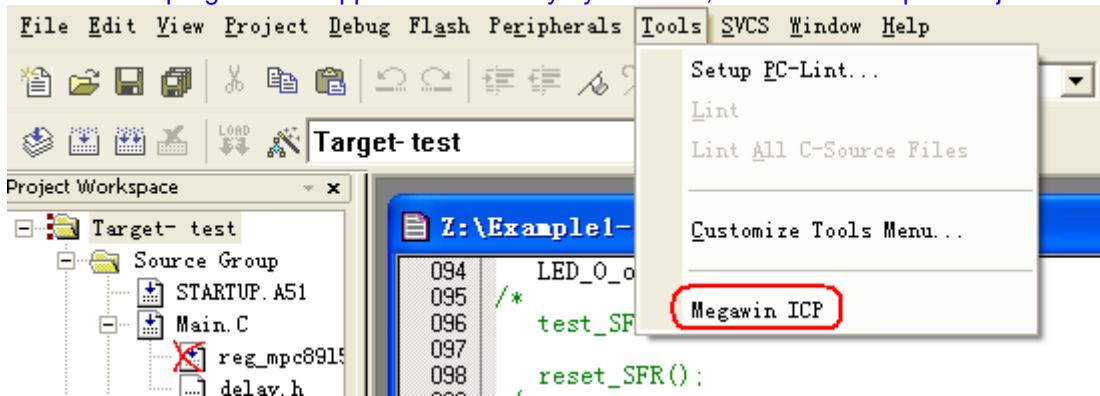
ICP is the acronym of In-Circuit Programming. Users can update the application code under the software control without removing the mounted MCU chip from the actual end product. In addition, because the programming data to be programmed to the target can be saved in the ICE adapter's non-volatile storage, this **stand-alone** programmer is able to work without host(PC) intervention. This feature is especially useful in the field without a PC.

### **6.2 Use ICP**

Here are the two ways of opening ICP application:

1. Execute "ICPProgrammer.exe" under "\C51\INC\Megawin\" of Keil's Install folder.
2. Click "Tools\Megawin ICP" from Keil's Menu bar.

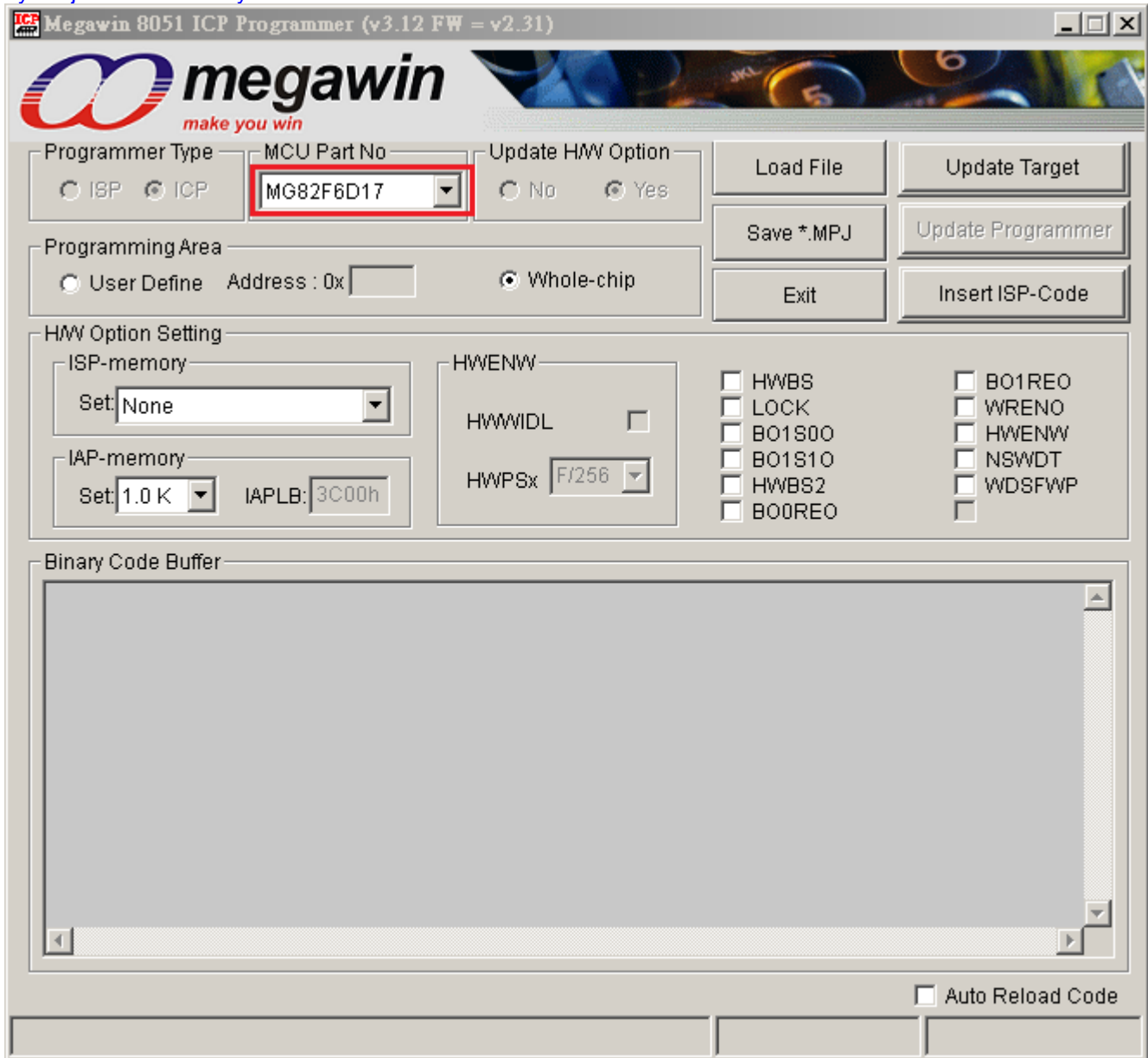
Attention: To program ICP application correctly by means 2, users have to open Project and Build first.



## 6.2.1 Update Programmer

Step 1: Choose a “MCU Part No.”

If users open ICP application by clicking Menu, Step 1 can be omitted. ICP application will choose MCU Part No. by Project automatically.



Megawin 8051 ICP Programmer (v3.12 FW = v2.31)

**megawin**  
*make you win*

Programmer Type: ☐ ISP ☒ ICP

MCU Part No: **MG82F6D17**

Update HW Option: ☐ No ☒ Yes

Load File

Update Target

Save \*.MPJ

Update Programmer

Exit

Insert ISP-Code

Programming Area: ☐ User Define Address: 0x  ☒ Whole-chip

HW Option Setting

ISP-memory: Set: **None**

IAP-memory: Set: **1.0 K** IAPLB: **3C00h**

HWENW

HWWIDL ☐

HWPSx: **F/256**

☐ HWBS ☐ BO1REO

☐ LOCK ☐ WRENO

☐ BO1S00 ☐ HWENW

☐ BO1S10 ☐ NSWDT

☐ HWBS2 ☐ WDSFWP

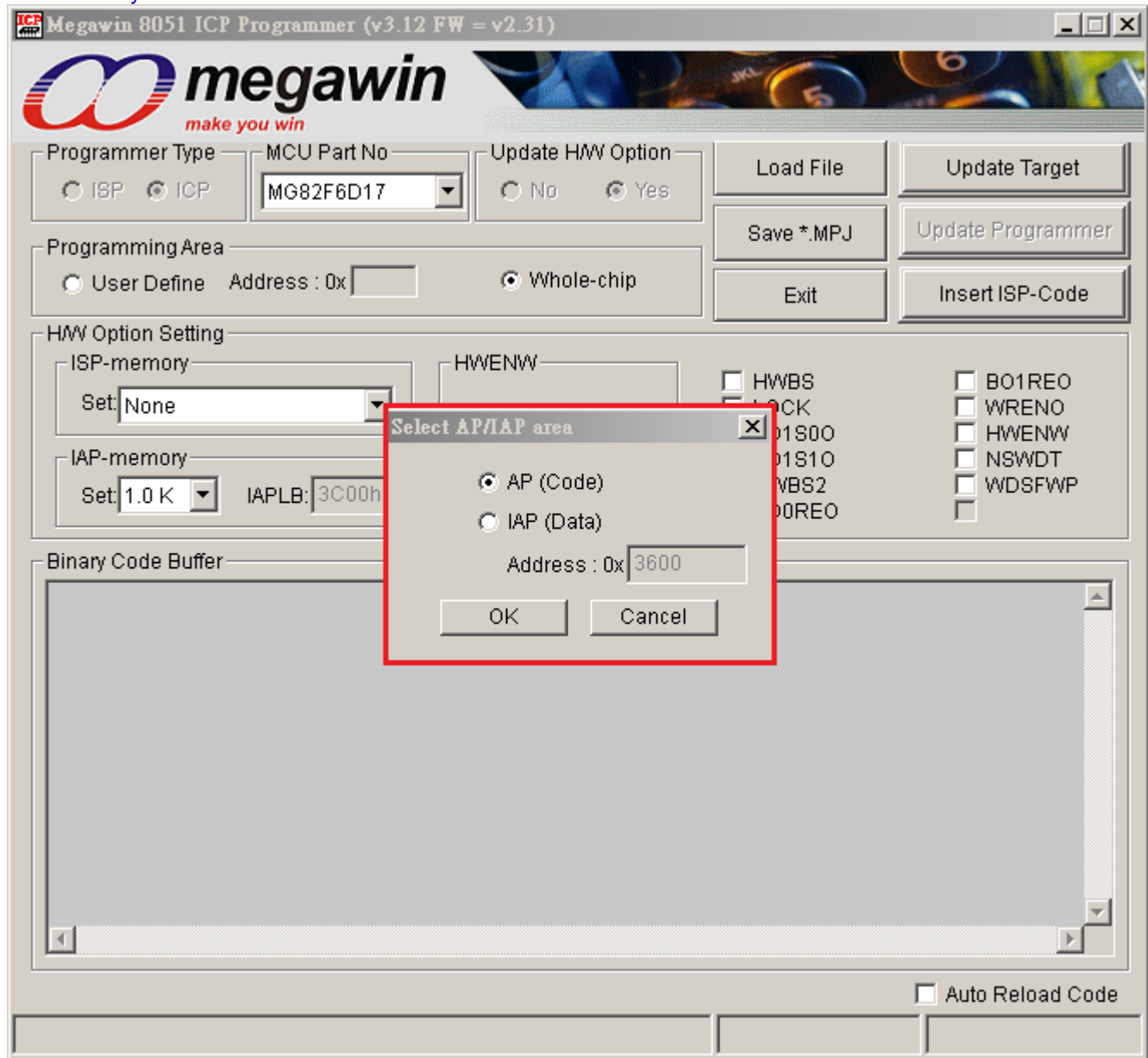
☐ BO0REO

Binary Code Buffer

☐ Auto Reload Code

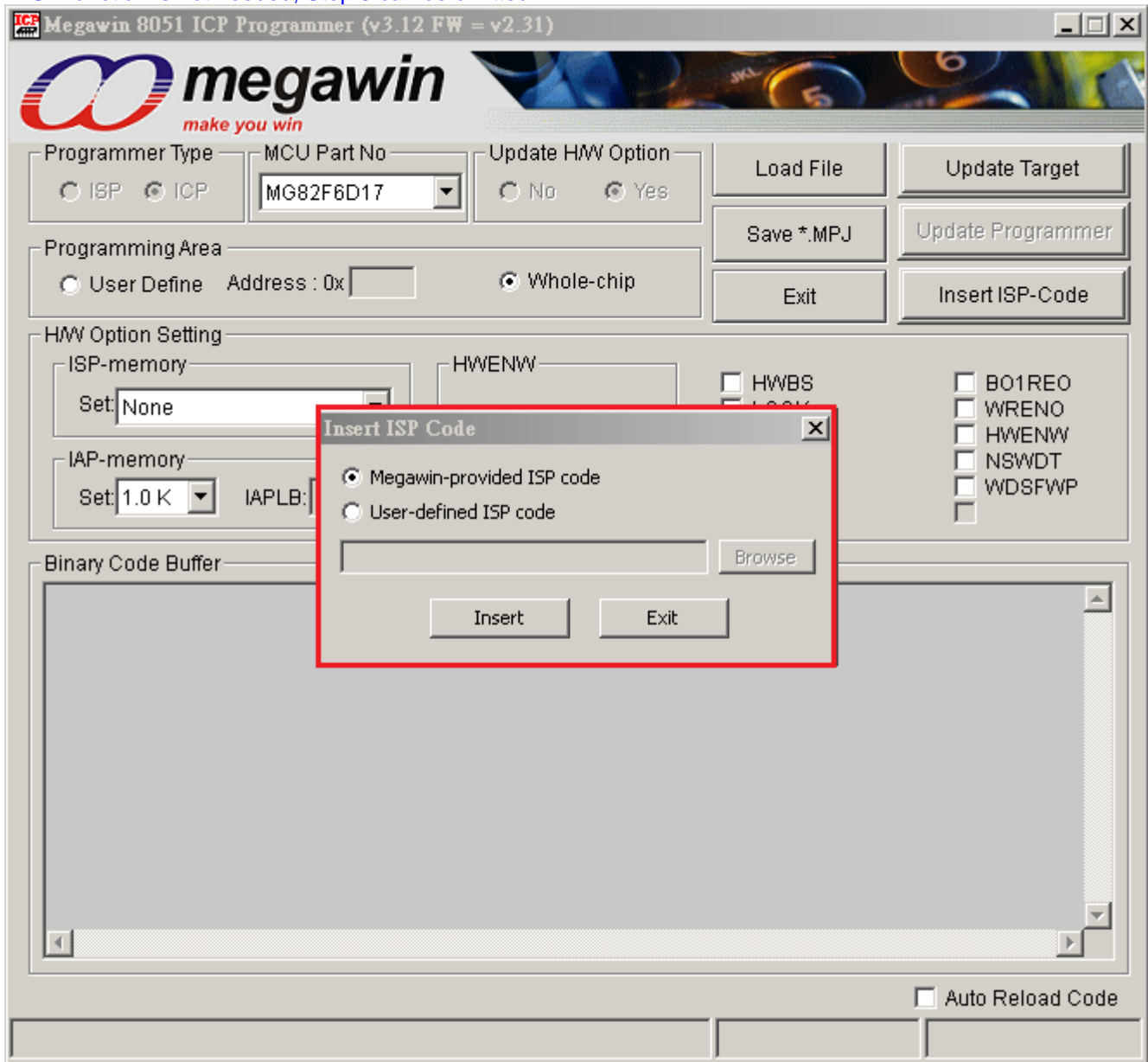
Step 2: Click "Load File" and choose loading AP(Code) or IAP(Data). "Load File" can be clicked repeatedly to load different files. While loading IAP(Data), users have to key in Address. HEX and BIN data formats are supported for file loading.

If users open ICP application by clicking Menu, Step 2 can be omitted. ICP application will load Target file automatically.



Step 3: Click “Insert ISP-Code” may choose to insert Megawin-provided ISP code or User-defined ISP code.

If ISP function is not needed, Step 3 can be omitted.



#### Step 4: H/W Option Setting

**Megawin 8051 ICP Programmer (v3.12 FW = v2.31)**

**megawin**  
make you win

Programmer Type: ☐ ISP ☒ ICP    MCU Part No: MG82G5E32    Update H/W Option: ☐ No ☒ Yes

Programming Area: ☐ User Define Address: 0x    ☒ Whole-chip

Buttons: Load File, Update Target, Save \*.MPJ, Update Programmer, Exit, Insert ISP-Code

---

**H/W Option Setting**

ISP-memory: Set: 1.5K

IAP-memory: Set: 1.0 K    IAPLB: 7600h

HWENW:    HWWIDL:    HWPSx: F/256

☒ HWBS    ☐ BO1REO  
☐ LOCK    ☐ WRENO  
☐ BO1S00    ☐ HWENW  
☐ BO1S10    ☐ NSWDT  
☐ HWBS2    ☐ WDSFWP  
☐ BO0REO    ☐

---

**Binary Code Buffer**

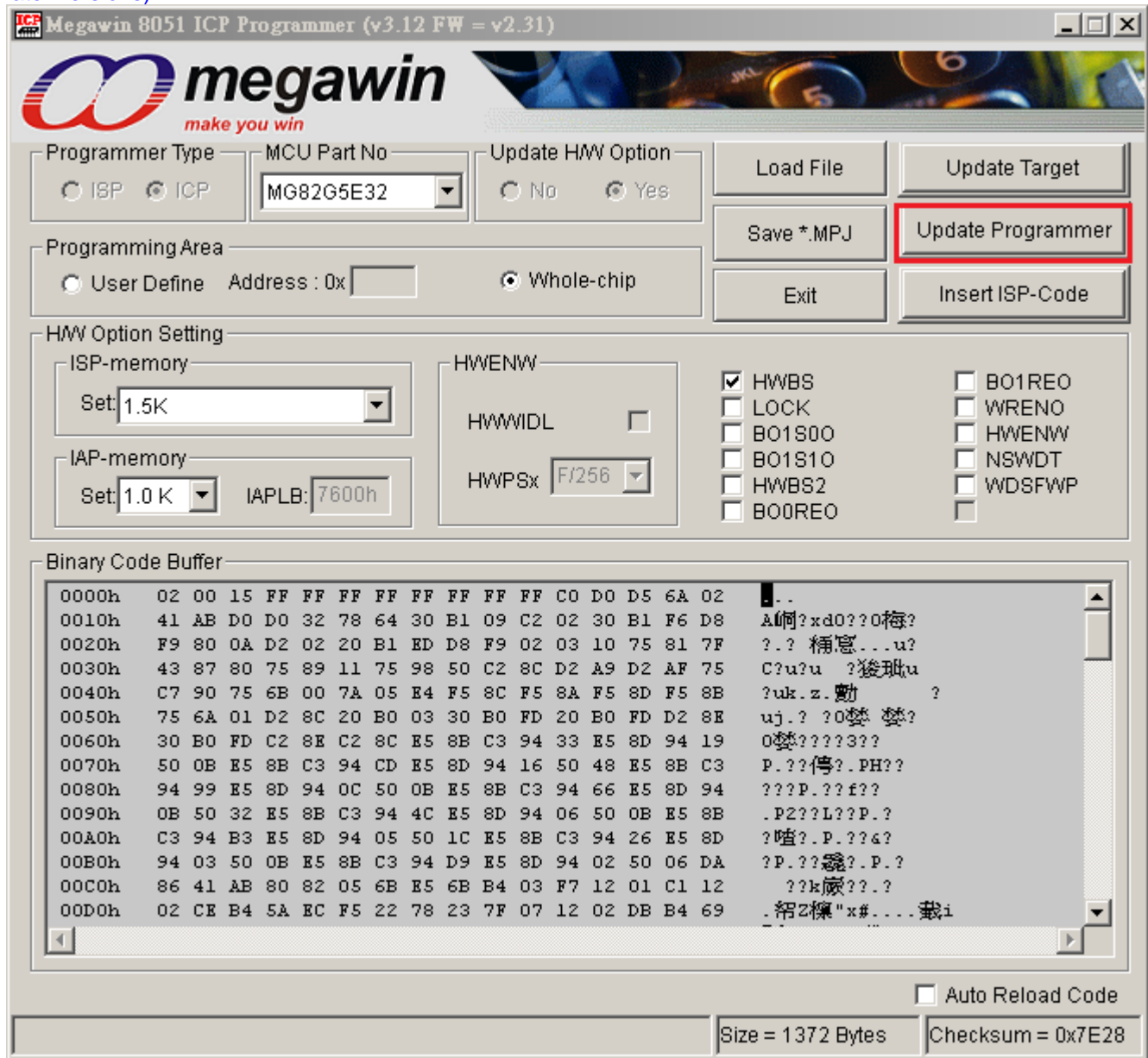
0000h	02 00 15 FF FF FF FF FF FF FF FF C0 D0 D5 6A 02	...
0010h	41 AB D0 D0 32 78 64 30 B1 09 C2 02 30 B1 F6 D8	A啊?xd0??0梅?
0020h	F9 80 0A D2 02 20 B1 ED D8 F9 02 03 10 75 81 7F	??.? 桶富...u?
0030h	43 87 80 75 89 11 75 98 50 C2 8C D2 A9 D2 AF 75	C?u?u ?後班u
0040h	C7 90 75 6B 00 7A 05 E4 F5 8C F5 8A F5 8D F5 8B	?uk.z. 勤 ?
0050h	75 6A 01 D2 8C 20 B0 03 30 B0 FD 20 B0 FD D2 8E	uj.? ?0婪 婪?
0060h	30 B0 FD C2 8E C2 8C E5 8B C3 94 33 E5 8D 94 19	0婪????3??
0070h	50 0B E5 8B C3 94 CD E5 8D 94 16 50 48 E5 8B C3	P.??停?.PH??
0080h	94 99 E5 8D 94 0C 50 0B E5 8B C3 94 66 E5 8D 94	???P.??f??
0090h	0B 50 32 E5 8B C3 94 4C E5 8D 94 06 50 0B E5 8B	.P2??L??P.?
00A0h	C3 94 B3 E5 8D 94 05 50 1C E5 8B C3 94 26 E5 8D	?啥?.P.??&?
00B0h	94 03 50 0B E5 8B C3 94 D9 E5 8D 94 02 50 06 DA	?P.??露?.P.?
00C0h	86 41 AB 80 82 05 6B E5 6B B4 03 F7 12 01 C1 12	??k廠???
00D0h	02 CE B4 5A EC F5 22 78 23 7F 07 12 02 DB B4 69	.紹2標"x#....裁i

☐ Auto Reload Code

Size = 1372 Bytes    Checksum = 0x7E28

Step 5: Click "Update Programmer" to download programming data to the ICE adapter.

"Update Programmer" function can be chosen only when connecting an ICE adapter (Only support TH065C or later versions).



The screenshot shows the Megawin 8051 ICP Programmer interface. The title bar indicates the version is v3.12 FW = v2.31. The interface includes several sections:

- Programmer Type:** Radio buttons for ISP and ICP (selected).
- MCU Part No:** A dropdown menu showing MG82G5E32.
- Update HW Option:** Radio buttons for No and Yes (selected).
- Buttons:** Load File, Update Target, Save \*.MPJ, Update Programmer (highlighted with a red box), Exit, and Insert ISP-Code.
- Programming Area:** Radio buttons for User Define and Whole-chip (selected).
- HW Option Setting:**
  - ISP-memory:** Set to 1.5K.
  - IAP-memory:** Set to 1.0 K, IAPLB: 7600h.
  - HWENW:** HWWIDL (unchecked), HWPSx: F/256.
  - Checkboxes:** HWBS (checked), LOCK (unchecked), BO1S00 (unchecked), BO1S10 (unchecked), HWBS2 (unchecked), BO0REO (unchecked), BO1REO (unchecked), WRENO (unchecked), HWENW (unchecked), NSWDT (unchecked), WDSFWP (unchecked).
- Binary Code Buffer:** A table showing hexadecimal data and its ASCII representation.
 

Address	Hex Data	ASCII
0000h	02 00 15 FF FF FF FF FF FF FF C0 D0 D5 6A 02	...
0010h	41 AB D0 D0 32 78 64 30 B1 09 C2 02 30 B1 F6 D8	A啊?xd0??0梅?
0020h	F9 80 0A D2 02 20 B1 ED D8 F9 02 03 10 75 81 7F	?.? 桶意...u?
0030h	43 87 80 75 89 11 75 98 50 C2 8C D2 A9 D2 AF 75	C?u?u ?後班u
0040h	C7 90 75 6B 00 7A 05 E4 F5 8C F5 8A F5 8D F5 8B	?uk.z. 勳 ?
0050h	75 6A 01 D2 8C 20 B0 03 30 B0 FD 20 B0 FD D2 8E	uj.? ?0焚 焚?
0060h	30 B0 FD C2 8E C2 8C E5 8B C3 94 33 E5 8D 94 19	0焚????3??
0070h	50 0B E5 8B C3 94 CD E5 8D 94 16 50 48 E5 8B C3	P.??停?.PH??
0080h	94 99 E5 8D 94 0C 50 0B E5 8B C3 94 66 E5 8D 94	???P.??f??
0090h	0B 50 32 E5 8B C3 94 4C E5 8D 94 06 50 0B E5 8B	.P2??L??P.?
00A0h	C3 94 B3 E5 8D 94 05 50 1C E5 8B C3 94 26 E5 8D	?喳?.P.??4?
00B0h	94 03 50 0B E5 8B C3 94 D9 E5 8D 94 02 50 06 DA	?P.??魏?.P.?
00C0h	86 41 AB 80 82 05 6B E5 6B B4 03 F7 12 01 C1 12	??k厥???
00D0h	02 CE B4 5A EC F5 22 78 23 7F 07 12 02 DB B4 69	.帮2操"x#....裁1
- Footer:** Auto Reload Code (unchecked), Size = 1372 Bytes, Checksum = 0x7E28.

### **6.2.2 Update Target**

How to update the target? Users may

1. click "Update Target" to program on-line update, referring to steps 1 through 4 of 6.2.1 Update Programmer, or
2. click "Downloading" of ICE adapter to program off-line update, referring to 6.2.1 Update Programmer.



## 7 Special Notes

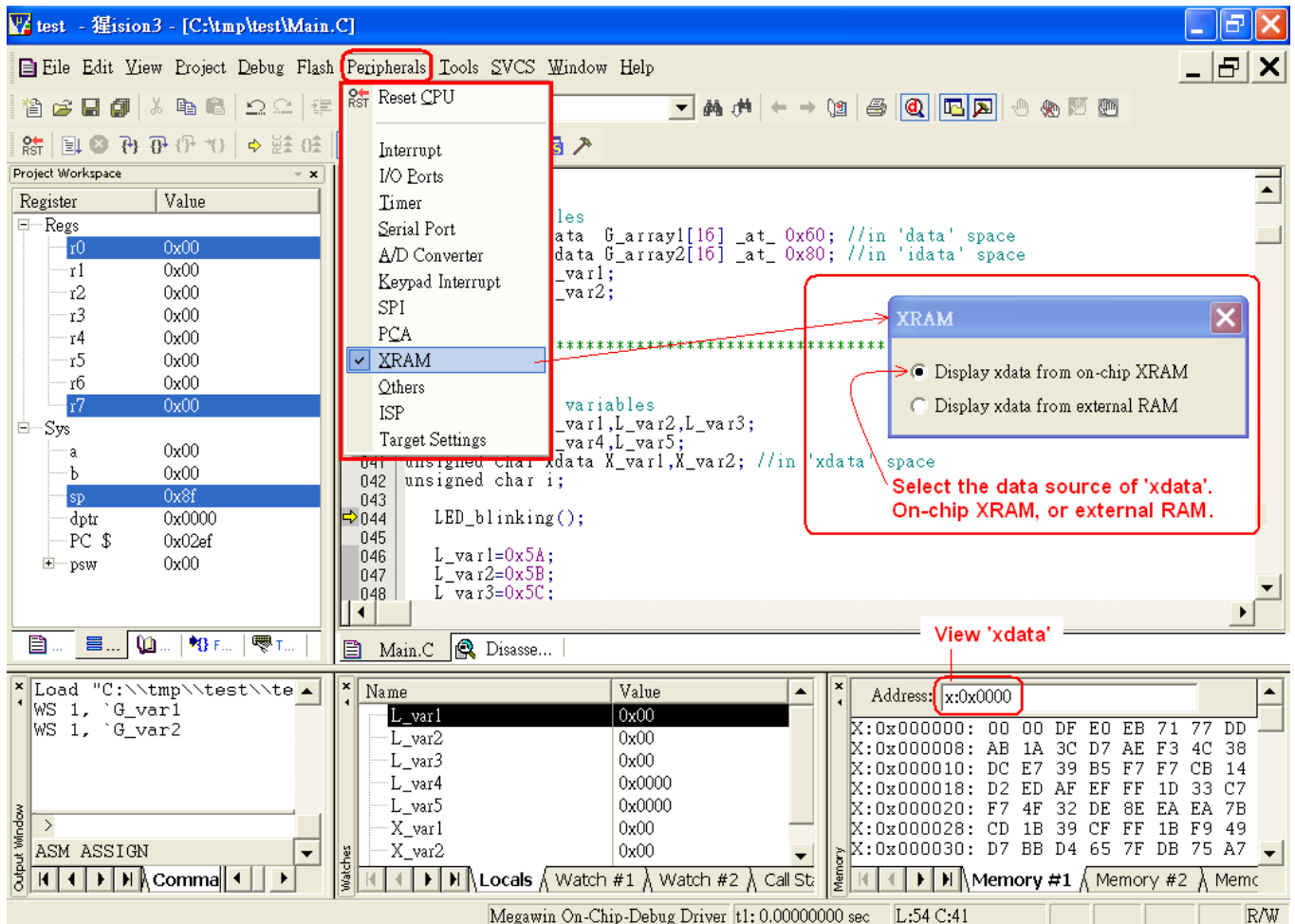
### 7.1 Register Definition Files

Register definition files *REG\_MPC82G516.INC* and *REG\_MPC82G516.H* define all Special Function Registers (SFRs) and bit-addressable control/status bits. They are installed into the default search path used by the Keil 8051 IDE software when you do the Software Setup (described in Section 2). Therefore, when using the Keil 8051 tools, you can include them by `$INCLUDE (REG_MPC82G516.INC)` and `#include <REG_MPC82G516.H>`. It is not necessary to copy a register definition file to each project's file directory.

### 7.2 On-chip XRAM and External Data Memory

Megawin 8051 devices provide on-chip XRAM (eXpanded RAM), which is accessed with the same instructions as the traditional external data memory. The size of on-chip XRAM in MPC82G516 is 1024 bytes with addresses 0x0000 to 0x03FF. That is, the address space of on-chip XRAM overlaps that of the external data memory. So, there must be a control bit used to distinguish these two physical memories during access. The ERAM bit (bit-1 in register AUXR) plays this role. *Because the C51 Compiler won't take care which physical memory the user wants to access, the user must manually clear this bit before accessing on-chip XRAM and set this bit before accessing external data memory.* By default, this control bit is '0' after powered on or chip reset for on-chip XRAM accessing.

The C51 Compiler offers two different memory types that access external data: *xdata* and *pdata*. (The *xdata* memory specifier refers to any location in the 64K-byte address space of external data memory. The *pdata* memory type specifier refers to only one page or 256 bytes of external data memory.) When the user want to view the variables declared by *xdata* or *pdata* directly in the Memory Window rather than in the Watch Window, he should select “**Display xdata from on-chip XRAM**” or “**Display xdata from external RAM**” under menu **Peripherals- XRAM**, as shown in the following figure.



The following example code shows how to use both on-chip XRAM and external RAM in an application. To view G\_array1[ ], select “Display xdata from on-chip XRAM”; and to view G\_array2[ ], select “Display xdata from external RAM”.

#### Example of using both on-chip XRAM and external RAM

```
unsigned char xdata G_array1[512] _at_ 0x0000; // in 'xdata' space, will use on-chip XRAM
unsigned char xdata G_array2[512] _at_ 0x0000; // in 'xdata' space, will use ext. RAM
unsigned int i;

AUXR&=0xFD; //clear AUXR.1 for on-chip XRAM
for (i=0; i<512; i++) G_array1[i]=0x5A; // fill XRAM with 0x5A

AUXR|=0x02; //set AUXR.1 for external RAM
for (i=0; i<512; i++) G_array2[i]=0xA5; // fill ext. RAM with 0xA5
```

Note that there will be a linking warning listed below. However, it doesn't matter because we intentionally declare G\_array1 and G\_array2 in the same address space. In fact, we access to the different physical memory controlled by bit-1 of AUXR.

```
linking...
*** WARNING L6: XDATA SPACE MEMORY OVERLAP
FROM: 0000H
TO: 01FFH
```

### **7.3 Code Optimization and Source-Level Debugging**

As shown in the following source code, the C51 compiler won't generate any machine code for “*L\_var1=0x38;*” because this statement becomes meaningless due to its following statement “*L\_var1=0xC7;*”. For code optimization, “*L\_var1=0x38;*” will be optimized out unless the code optimization is disabled as described in [Section 4.4](#).

```
unsigned char L_var1;

L_var1=0x38; // ! Note: this statement may be optimized out by the C51 compiler
L_var1=0xC7;
```

So, during source-level debugging, *L\_var1* will never show *0x38* but may show a random number when this statement is just executed. In fact, there is no machine code for this statement. The user should pay attention to it!

Sometimes, for debugging purpose, the user may disable the compiler's code optimization. Note that once the compiler's code optimization is disabled, there may be some linking errors which won't occur when the code optimization is enabled. For example, refer to the following linking error message, it means the variables you use exceed the RAM an MCU has. To make this error disappear, the only way is to enable the compiler's code optimization to let the compiler make more efficient use of the RAM.

```
linking...
*** ERROR L107: ADDRESS SPACE OVERFLOW
SPACE: DATA
SEGMENT: ?DT?_VP_DISPLAYMODE?VP
LENGTH: 0001H
```

## **7.4 “for-Loop” and Source-Level Debugging**

The following two statements are fully the same for the 8051 CPU to execute them. During source-level debugging, there is no problem to apply *Step* action on Statement 1. However, it will take so much time if the user apply *Step* action on Statement 2. We think it is caused by unknown processing in the Keil debugger function. Before we getting the reply from Keil, we suggest using Statement 1 instead of Statement 2 in the source code if you want to do step-debugging in such statement. Another solution for Statement 2 is: move cursor to Line2 and click left key, then click *Run-to-Cursor* button to fly over Line 1.

Statement 1:

```
Line1: for (i=0; i<16; i++) {  
Line2:     G_array1[i]=i+0x60;  
Line3: }
```

Statement 2:

```
Line1: for (i=0; i<16; i++) G_array1[i]=i+0x60;  
Line2: ...  
Line3: ...
```

## **7.5 Hardware Option Requirements During Debugging**

There are two requirements regarding the hardware option in the *dScope-Debugger* mode:

Requirement 1: The debugged chip must be in un-locked state

It is because if the debugged chip is locked, the downloading of the user's application code in the *dScope-Debugger* mode will cause the chip to be whole-chip erased, and therefore all the chip's hardware options will be disabled. Thus the debugged chip may not work well owing to losing its original hardware options. For example, for a locked chip with IAP-memory configured, after downloading the user's application code when entering the *dScope-Debugger* mode, its IAP-memory will disappear (i.e., disabled). So, the chip cannot work well.

Requirement 2: The ISP function of the debugged chip must be disabled

It is because if the ISP function is enabled, the debugged chip will always boot from the ISP-memory and run the ISP-code when the chip receives the *Reset* command in the *dScope-Debugger* mode. It will cause a problem. That is, the code the MCU runs (i.e., the ISP-code) is different from the code of the opened Keil project (i.e., the user's application code). So, during debugging, the user needs to disable the ISP function by having the hardware option *HWBS* disabled temporarily.

*Note:*

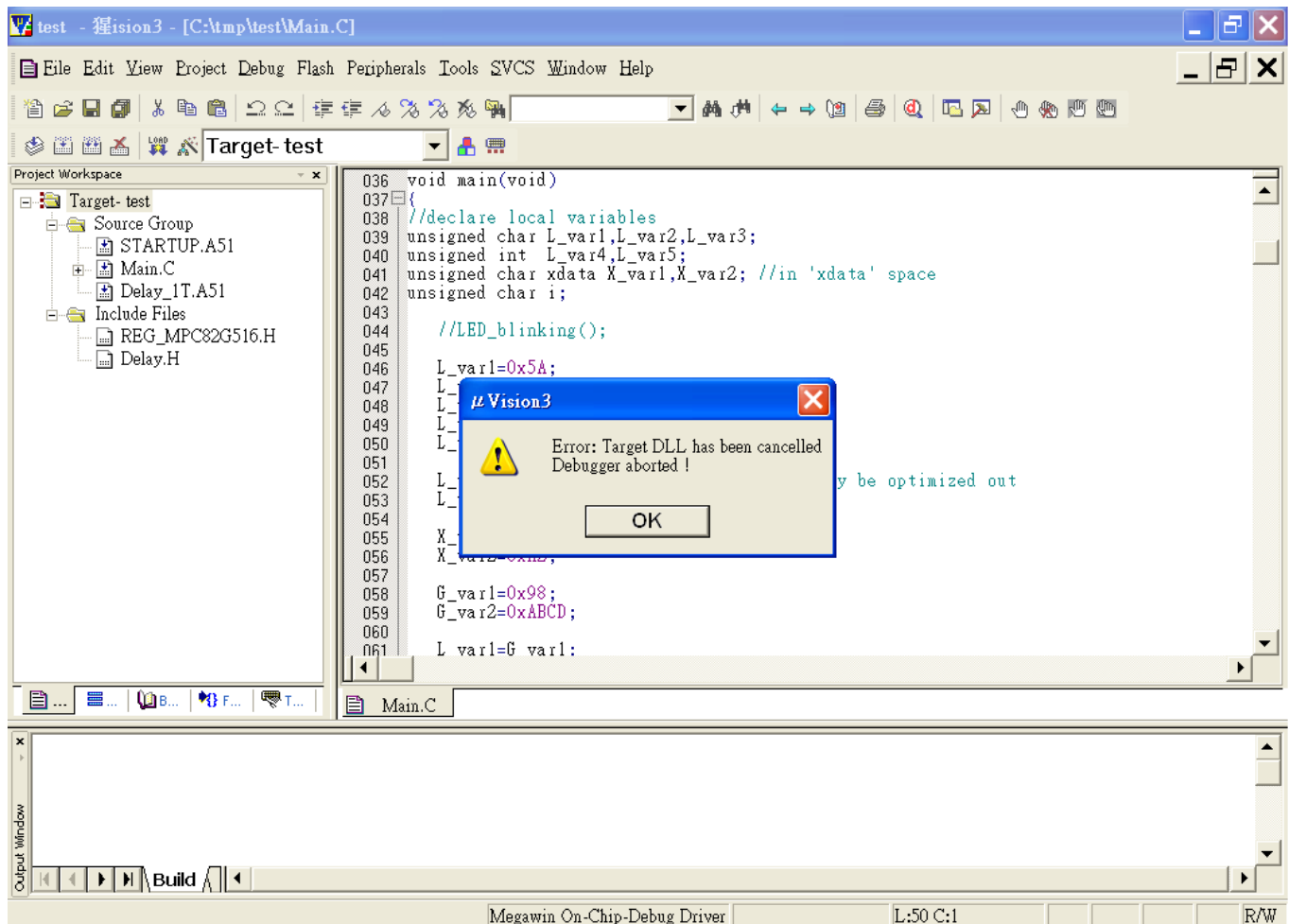
*After the application code is debugged completely, the user may use the “Megawin 8051 ICP Programmer” to restore the original hardware option.*

## 7.6 Error Message

There will be an error message “**Error: Target DLL has been cancelled. Debugger aborted !**” shown in following figure if:

- (1) ICE adapter hardware fails, or
- (2) Target MCU doesn't work (for example, not powered on), or
- (3) Cable error or improper connection between ICE adapter and the Target MCU.

Once the error message pops out, click “**OK**”. Then, check the above possible causes to solve the problem.



## 7.7 Properly Connect the ICE Adapter to a Host

The data transfer rate of the ICE adapter will be slowed down severely if it is connected to a host via a USB2.0 hub. So, to speed up the downloading when clicking *dScope* button to enter the debugger mode, the user had better directly plug the ICE adapter into the host's USB port, as shown in Figure 6.7.1. Don't plug into a hub and then to the host, as shown in Figure 6.7.2.

Figure 6.7.1 Directly plug into the host's USB port

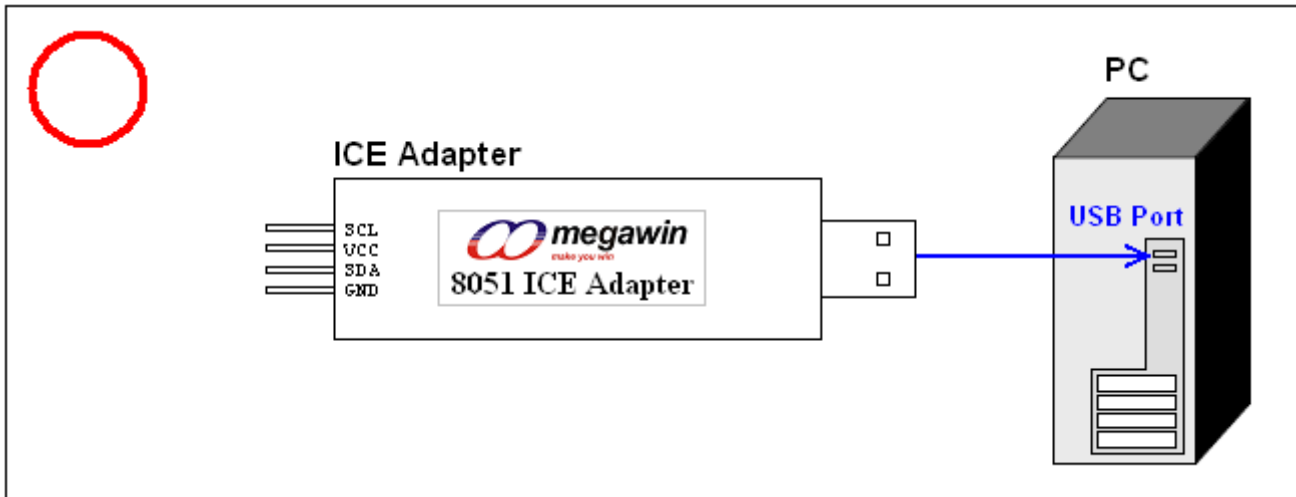
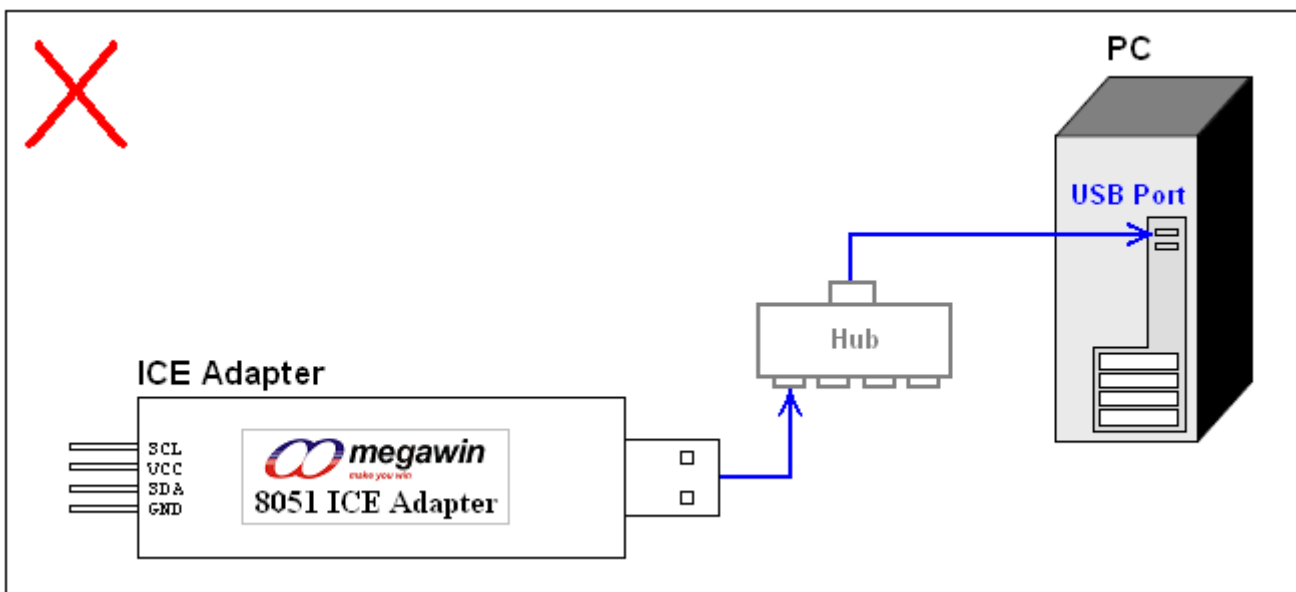


Figure 6.7.2 Don't plug into a hub and then to the host's USB port



## Revision History

Revision	Description	Date
v1.00	The first release for beta-site test.	2007/08/15
v1.01	Add notes when installation fails. (Section 3.2)	2007/08/24
v1.02	Change to manually specify the installation path of the Keil software. (Section 3.2)	2007/08/27
v2.00	Add the notification of default installation path of Keil 8051 IDE software. (Section 3.2)	2007/08/29
	Update the Keil IDE Setup. (Section 4.4)	2007/10/08
	Update the Special Notes. (Section 6)	2007/10/08
	The formal released version.	2007/10/08
v2.10	(1) Improve the defect of breakpoint setting. (2) Fix the bug of wrong erasing range when downloading the application code.	2007/12/26
V2.20	(1) Update the data base for all series of MCU in Driver Installer. (2) Removed the function of detecting the ICE adapter when install Driver.	2009/02/27
V2.21	Change the folder name of Driver Install to Database install	2009/04/01
V2.30	(1) Supported MG82FL(E)532 and MG82FL(E)564 (2) Supported ICP function	2010/05/10
V2.31	Update "Database Installer "	2010/05/21
V2.32	Support uVision4	2010/06/02
V2.33	Update "IcpProgrammer.exe" in Database Installer	2010/08/25
V2.40	Supported MG84FG516	2011/05/02
V2.41	Update "IcpProgrammer.exe" in Database Installer	2011/06/01
V2.50	Support Off-Line Mode programming	2011/10/20
V2.51	Support H/W ver.TH065E to prevent to damage the MG84FG516	2012/04/01
V2.52	Fix the bug on ICP function for MG84FG516	2012/05/01
V2.53	Update "IcpProgrammer.exe" in Database Installer	2012/05/15
V2.54	(1) Supported "Maximum Counter" in Off-Line Mode programming (2) Supported " Serial Number " in Off-Line Mode programming (3) Improve the performance on Off-Line Mode programming.	2012/07/12
V2.55	Fix the bug on ICP function	2012/09/28
V2.56	Fix the bug on ICE function	2012/10/08
V2.60	(1) Supported "MPJ" file (2) Database support MG86FL(E)104 and MG86FL(E)508 (3) Supported MG82FG5A64 (4) Update " warning message " when OCD ICE in update processing	2012/12/10
V2.61	(1) Fix a bug for MG84FG516 at access P6M0 in debug mode (2) Update " Megawin.dat "	2013/01/10
V2.62	Update "IcpProgrammer.exe" in Database Installer	2013/01/14
V2.63	Supported MG82FG5A32	2013/06/27
V2.64	Update the Hardware Setup. (Section 2)	2013/09/27
V2.70	(1) Supported MG82FG5B(32/16) (2) Supported MG20FL(E)809	2103/11/15
V2.71	Supported MG82FG5B(24/08)	2104/04/09
V2.72	Update H and INC files in H and INC folder	2014/05/15
V2.90	Supported MG82FG5C(64/32)	2015/04/15

V2.91	Update "IcpProgrammer.exe" in Database Installer	2015/05/21
V2.92	Update "IcpProgrammer.exe" in Database Installer	2015/05/22
V2.93	Update "MegawinOCD.dll" in Database Installer	2015/08/24
V2.94	Update "MegawinOCD.dll" in Database Installer	2015/12/14
V2.95	Support " Auto Reload Code " in IcpProgrammer	2016/09/01
V2.96	Update "IcpProgrammer.exe" in Database Installer	2015/09/26
V3.00	Supported MG82FG5D(08/16)	2017/06/09
V3.01	Update "IcpProgrammer.exe" in Database Installer	2017/11/16
V3.02	Update "MegawinOCD.dll" in Database Installer	2017/12/28
V3.03	Add HexEdit.dll in Database Installer and modify IcpProgrammer.exe	2018/01/23
V3.04	Modify the sample codes	2018/03/26
V3.05	Supported MG82G5E32	2018/05/10
V3.08	Fix Messages when reset to AP on ICP Programmer	2018/09/10
V3.09	Modify SFR's "PDTCLR" to "PDTCLRA"	2018/11/07
V3.10	Update "MegawinOCD.dll", Fix bug	2018/12/14
V3.11	Fix ICP Programmer bug, Off-line program error	2019/01/08
V3.12	Supported MG82F6D17	2019/03/11
V3.13	Fix ICP Programmer bug. H/W Options error	2019/04/10
V3.14	Modify Header files for MG82F6D17	2019/05/20
V3.15	Fix ICP Programmer for win10 bug when off-line mode	2019/06/12
V3.16	Fix DLL bug when update SFR dialog	2019/09/12
V3.17	Modify DLL, fix bug: not write default OR when while-chip erase	2019/11/05
V3.18.0.0	Supported MG82F6D64/6D32	2020/04/17
V3.18.0.1	Add TREN0, TRLC0, TSPC0 to SFR dialog	2020/07/02
V3.18.0.2	Fix ICP Programmer body error	2020/08/05
V3.18.0.3	Modify Header files for MG82F6D64/32	2020/09/07
V3.18.0.4	Modify Header files for MG82F6D64/32	2020/09/24
V3.19.0.0	Support MG82F6D16	2021/03/02
V3.19.0.1	Fix ICP Programmer for MG82FG6D16 bug when offline mode	2021/06/30
V3.19.0.4	1. Fix ICP Programmer, LOCK unchecked for MG82FG6D16 H/W Option Setting 2. Fix ADCON0.CH3 to CHS3 for MG82FG5Bxx and MG82FG5Cxx HeaderFiles	2021/08/23
V3.20.0.0	1. Update MG84FG516 ISP(DFU) code 2. Update MG82F6D17 default OR 3. Modify 5A/5B/5C HWPS list for ICP Programmer 4. Add MG82F5Bxx	2022/03/01
V3.20.0.1	1. Fix MG82F5Bxx update error for ICP Programmer off-line button run	2022/11/22
V3.21.0.0	1. Debug : MG82FG5Bxx, MA82F5Bxx info	2023/06/20