NuMicro® 8051

8-bit Microcontroller

8051 Family
KEIL™ µVision® Driver
User Manual

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

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

www.nuvoton.com

Table of Contents

1 OVERVIEW

The Nuvoton KEIL™ µVision® Driver allows the KEIL™ µVision®2, µVision®3, µVision®4 or µVision®5 IDE to communicate with Nuvoton in-circuit-emulation (ICE) or on-chip-debug (OCD) device. The µVision IDE can start and stop program execution, set breakpoints, check variables, inspect and modify memory contents, and perform single-step through programs running on your actual target hardware. This document describes how to install and use Nuvoton 8051 KEIL™ µVision® Driver with programs written by the KEIL™ C51/A51 compiler.

After installing the Nuvoton KEIL™ µVision® Driver after Rev. 2.06, it additionally provides Nuvoton own CPU database "Nuvoton 8051 Devices". It collects all Nuvoton 8051 MCU and benefits user easily building KEIL™ µVision® project. For ML51, MS51, N76E003, N76E885 and N76E616, the corresponding header file is also installed in KEIL™ general library. User can include it directly by writing e.g. "#include < ML51.h file >" without caring of its file location.

Open hyperlink to see supported 8051 devices: Link (search for 8051 series).

2 SYSTEM REQUIREMENTS

- Software: KEIL^{TM} µVision$^{®}$2, µVision$^{®}$3, µVision$^{®}$4 or µVision$^{®}$5.
- Hardware: Support evaluation board with Nu-Link interface.

3 INSTALL NUVOTON KEIL^{TM} µVISION® DRIVER

A complete set of files needed to run the Nuvoton 8051 KEIL^{TM} µVision® Driver inside the µVision®2, µVision®3, µVision®4 or µVision®5 interface is contained in the package.

All steps are performed automatically by setup program. Please follow the steps below:

- Install µVision® and the C51/A51 Compiler on your computer.
- Run Nu-Link_Keil_Driver.exe and then select the *<KEIL install path>*. The Hnew files will appear inside *<KEIL install path>\C51\BIN\ Nuvoton_8051_KeiL_uVision_Driver.DLL*.
- The latest Nu-Link Keil driver is available on the following web page: https://www.nuvoton.com/tool-and-software/ide-and-compiler/

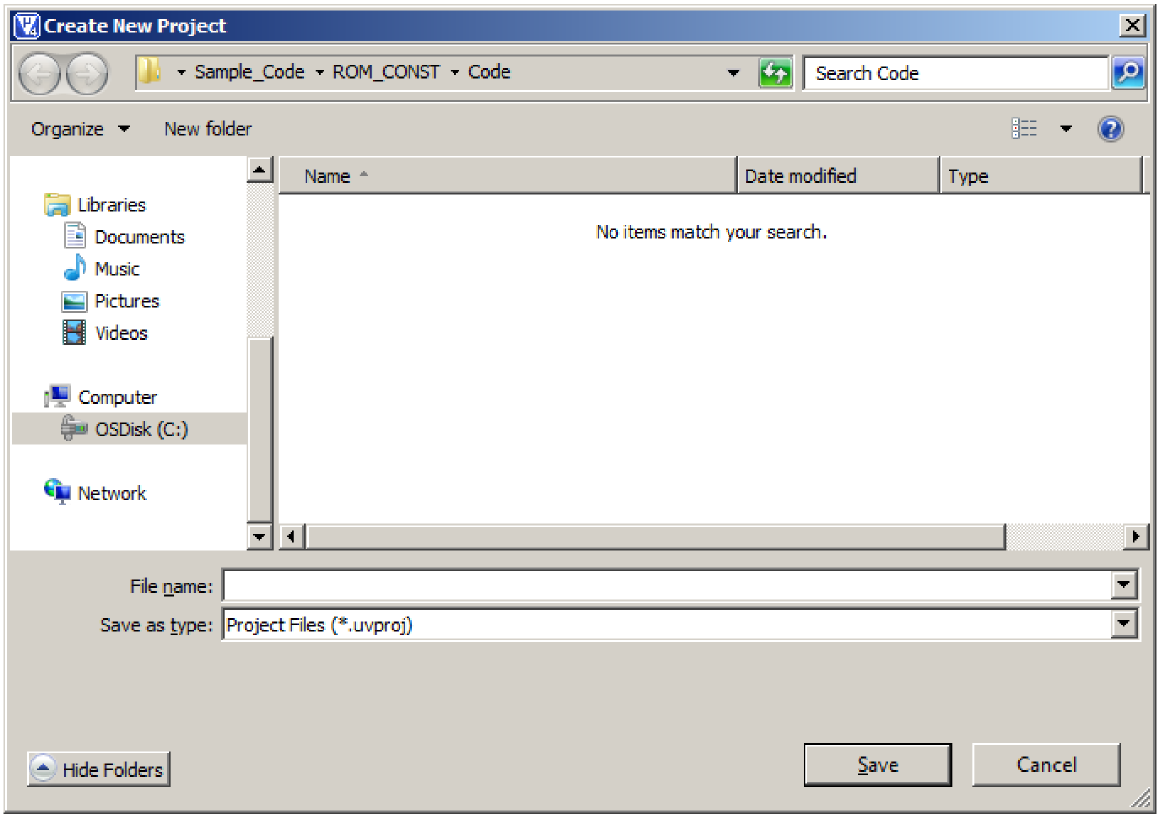
4 START TO WORK WITH µVISION® INTERFACE

4.1 Creating a New Project

To create a new project, please follow the steps below.

Step–1: Run the program and invoke “**Project – New µVision Project**” to open the *Create New Project* form.

Step–2: Input the new project name and click the “**Save**” button.

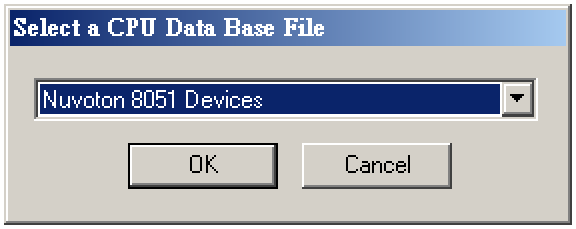


No items match your search.

Save as type: Project Files (*.uvproj)

Figure 4-1 Create a New Project

Step–3: Select "**Nuvoton 8051 Devices**" and click the "**OK**" button.

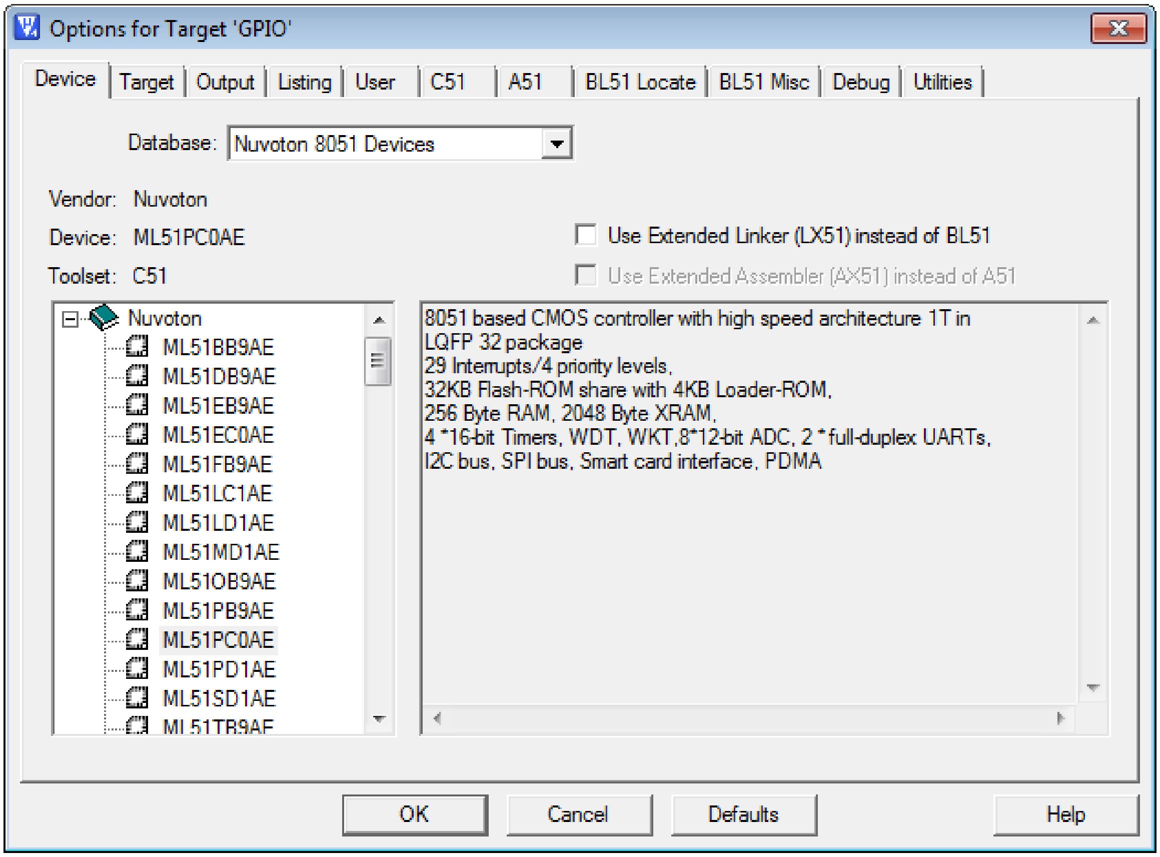


Select a CPU Data Base File

Nuvoton 8051 Devices

Figure 4-2 Select a CPU Database

Step–4: Select your device.



Database: Nuvoton 8051 Devices

Vendor: Nuvoton

Device: ML51PC0AE

Toolset: C51

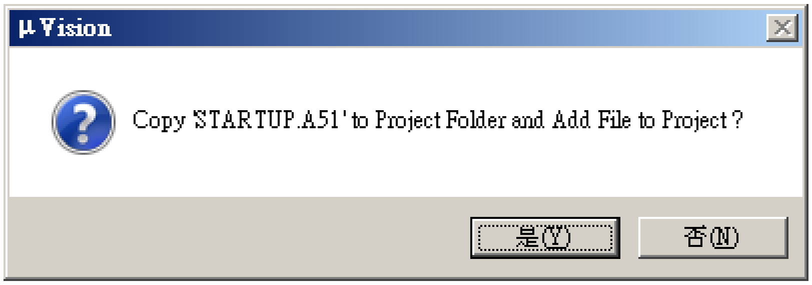
□ Use Extended Linker (LX51) instead of BL51

□ Use Extended Assembler (AX51) instead of A51

8051 based CMOS controller with high speed architecture 1T in
LQFP 32 package
29 Interrupts/4 priority levels,
32KB Flash-ROM share with 4KB Loader-ROM,
256 Byte RAM, 2048 Byte XRAM,
4 *16-bit Timers, WDT, WKT,8*12-bit ADC, 2 *full-duplex UARTs,
I2C bus, SPI bus, Smart card interface, PDMA

Figure 4-3 Select Your Device from the Database List

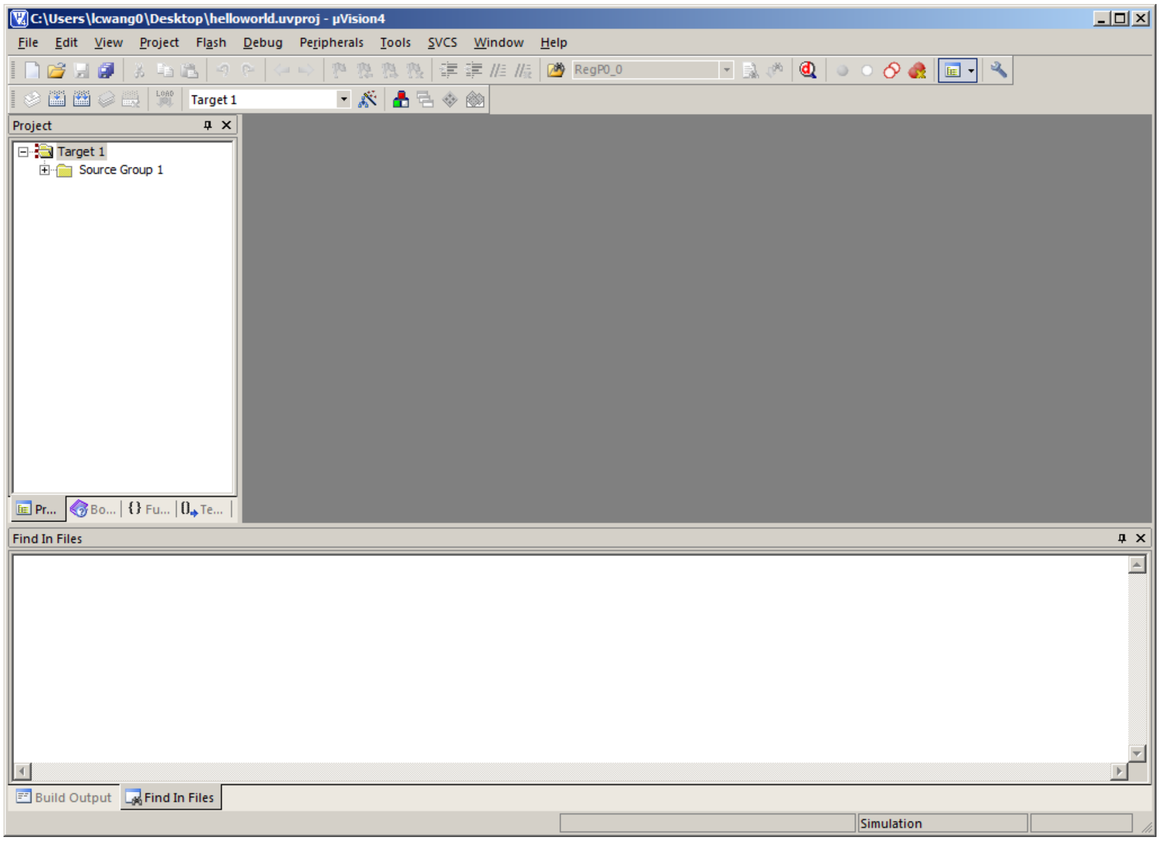
Step–5: Click “**Yes**” to copy “*STARTUP.A51*” to the project folder and complete the flow (as shown in *Figure 4-5*).



µVision

Copy 'STARTUP.A51' to Project Folder and Add File to Project ?

Figure 4-4 Add “STARTUP.A51” to Project



C:\Users\lcwang0\Desktop\helloworld.uvproj - µVision4

Figure 4-5 Complete the Process to Create a New Project

4.2 Setting Debug Options

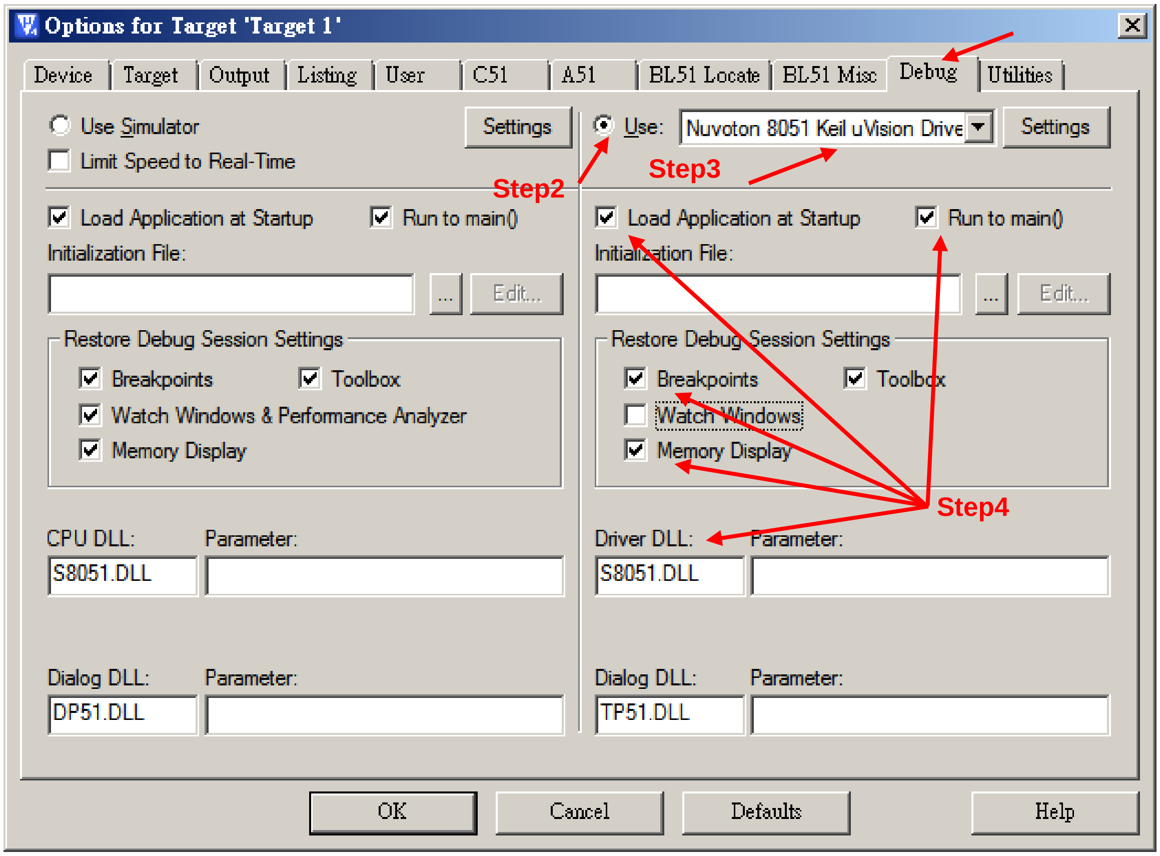
To set the debug options, please follow the steps below.

Step–1: Select the **Debug** tab in the “*Options for Target*” form.

Step–2: Make sure that the “**Use:**” option is checked.

Step–3: Select “**Nuvoton 8051 KEIL µVision Driver**”.

Step–4: Set options as shown in the figure below.



○ Use Simulator

☐ Limit Speed to Real-Time

☑ Load Application at Startup ☑ Run to main()

Initialization File:

Restore Debug Session Settings

☑ Breakpoints ☑ Toolbox

☑ Watch Windows & Performance Analyzer

☑ Memory Display

CPU DLL: S8051.DLL Parameter:

Dialog DLL: DP51.DLL Parameter:

● Use: Nuvoton 8051 Keil uVision Drive

☑ Load Application at Startup ☑ Run to main()

Initialization File:

Restore Debug Session Settings

☑ Breakpoints ☑ Toolbox

☐ Watch Windows

☑ Memory Display

Driver DLL: S8051.DLL Parameter:

Dialog DLL: TP51.DLL Parameter:

Figure 4-6 Set Debug Optins

Step–5: Select the **Utilities** tab in the “*Options for Target*” form.

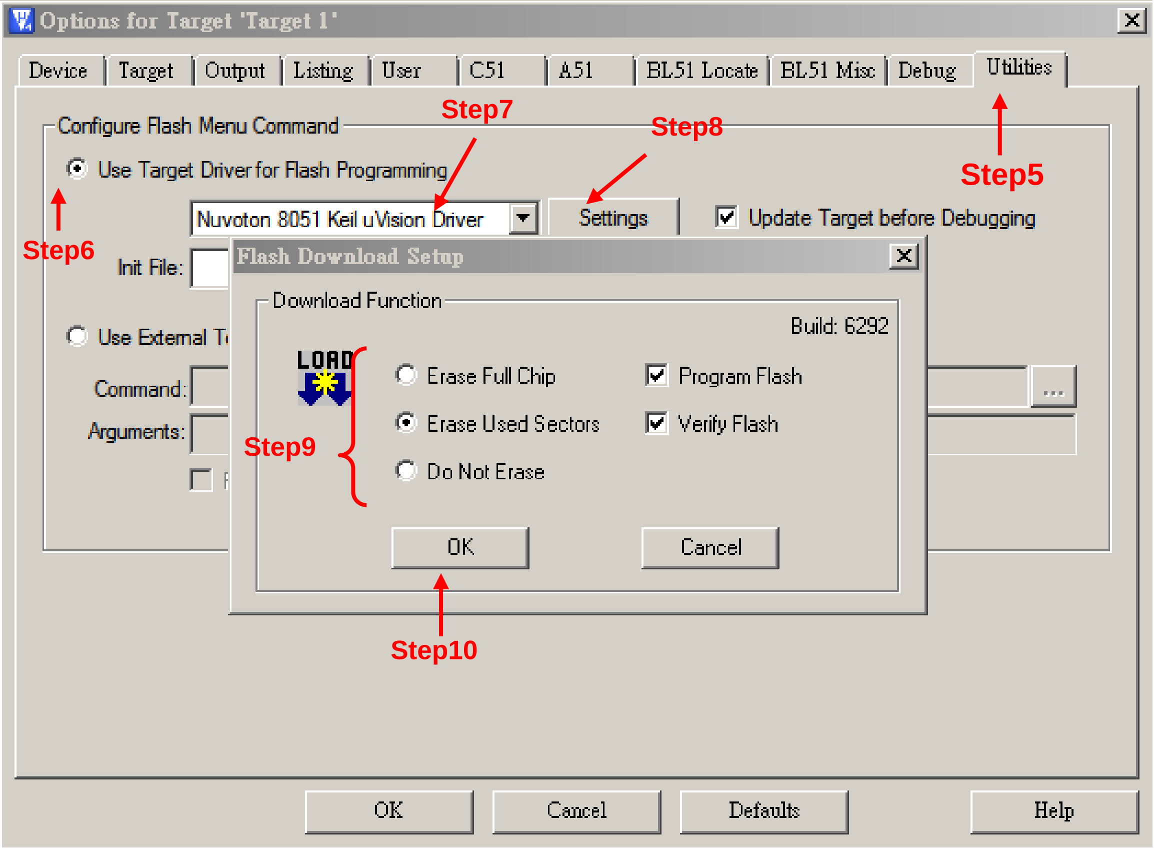
Step–6: Make sure that the “**Use Target Driver for Flash Programming:**” option is checked.

Step–7: Select “**Nuvoton 8051 KEIL µVision Driver**”.

Step–8: Click the “**Settings**” button.

Step–9: Check all options in the *Flash Download Setup* form.

Step–10: Click “**OK**” and complete the settings.



Options for Target 'Target 1'

Configure Flash Menu Command

● Use Target Driver for Flash Programming

Nuvoton 8051 Keil uVision Driver

☑ Update Target before Debugging

Init File:

○ Use External T

Command:

Arguments:

Flash Download Setup

Download Function

Build: 6292

○ Erase Full Chip

● Erase Used Sectors

○ Do Not Erase

☑ Program Flash

☑ Verify Flash

Figure 4-7 Set Download Options

5 USE THE DEBUG FEATURES

This chapter describes how to use Nuvoton 8051 KEIL^{TM} µVision$^{®}$ Driver with 8051 C51/A51 application based on the example project.

5.1 Debugging a New Project

Please follow the steps below after the installation had been performed.

Step–1: Invoke “**File – New**” to create a new file and save into C language.

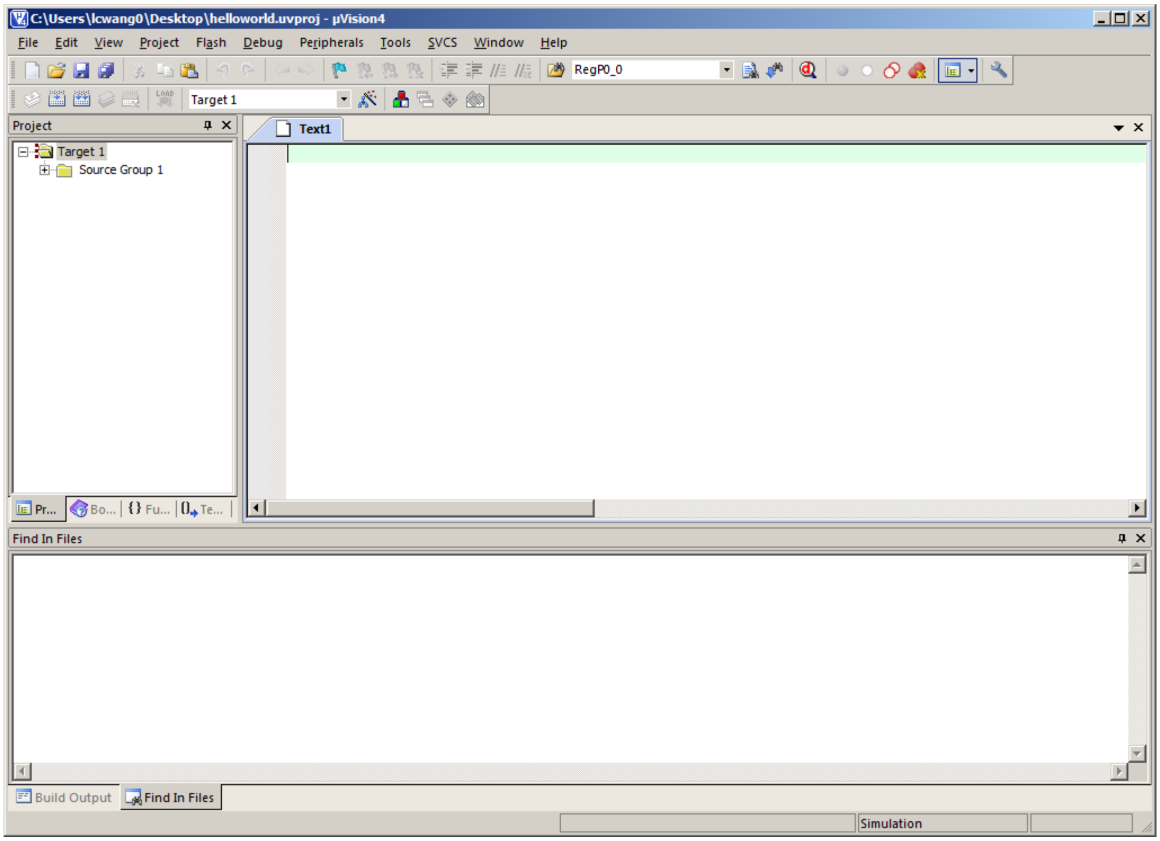
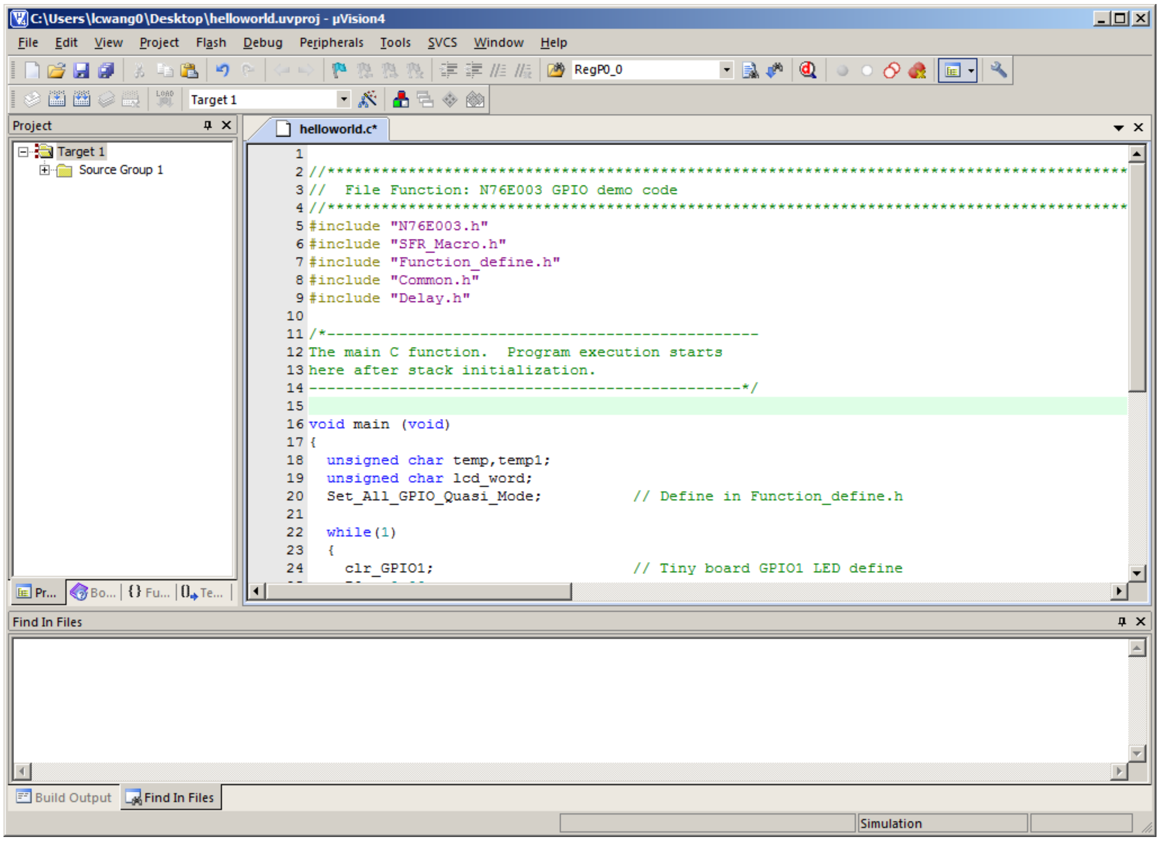


Figure 5-1 Create a New C File

Step–2: Write sample code as shown in *Figure 5-2* and save it.


C:\Users\lcwang0\Desktop\helloworld.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
Target 1
Project
Target 1
Source Group 1
Pr... Bo... {} Fu... 0↓ Te...
helloworld.c*
Find In Files
Build Output
Find In Files
Simulation

```
//***********************************************************************************************
//  File Function: N76E003 GPIO demo code
//***********************************************************************************************
#include "N76E003.h"
#include "SFR_Macro.h"
#include "Function_define.h"
#include "Common.h"
#include "Delay.h"

/*------------------------------------------------
The main C function.  Program execution starts
here after stack initialization.
------------------------------------------------*/

void main (void)
{
  unsigned char temp,temp1;
  unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;          // Define in Function_define.h

  while(1)
  {
    clr_GPIO1;                      // Tiny board GPIO1 LED define
```

Figure 5-2 Write Sample Code

Step–3: Invoke "**Project – Manage – Components, Environment, Books**". Add the C file that had been created through *Step–2*.

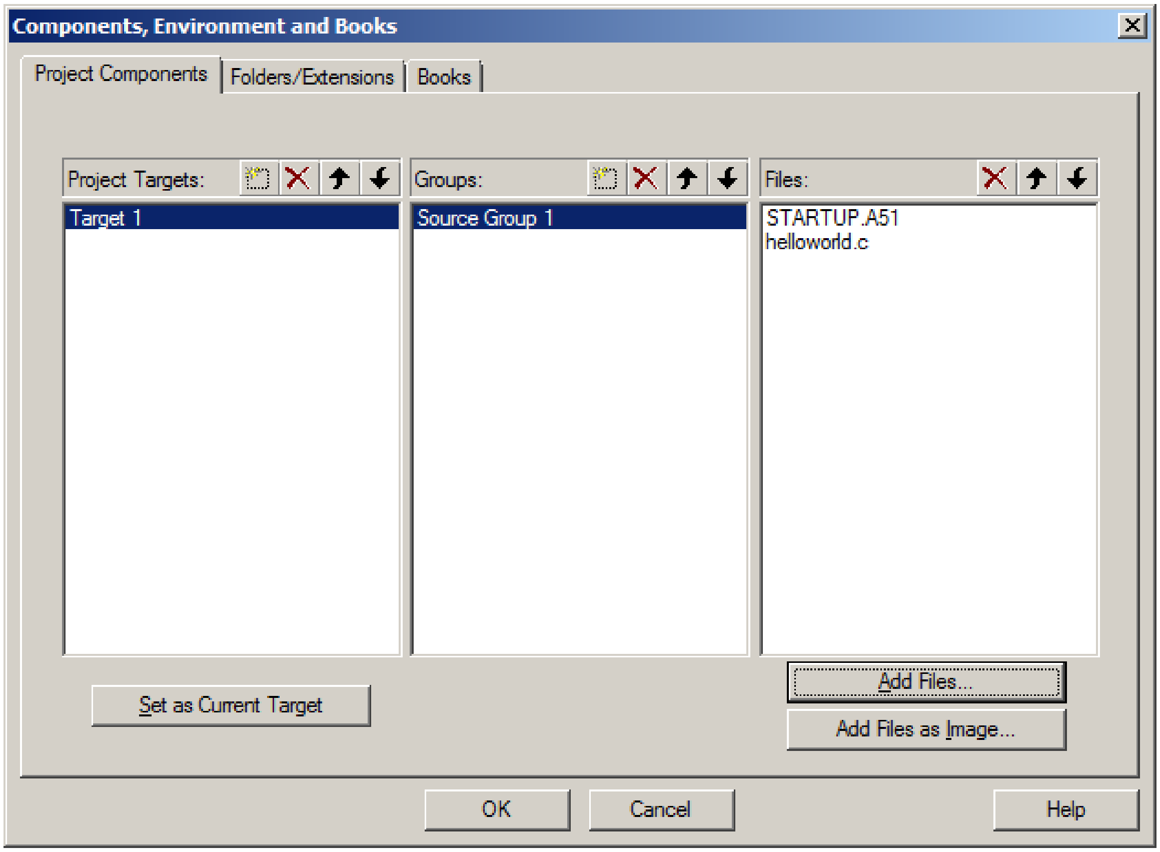
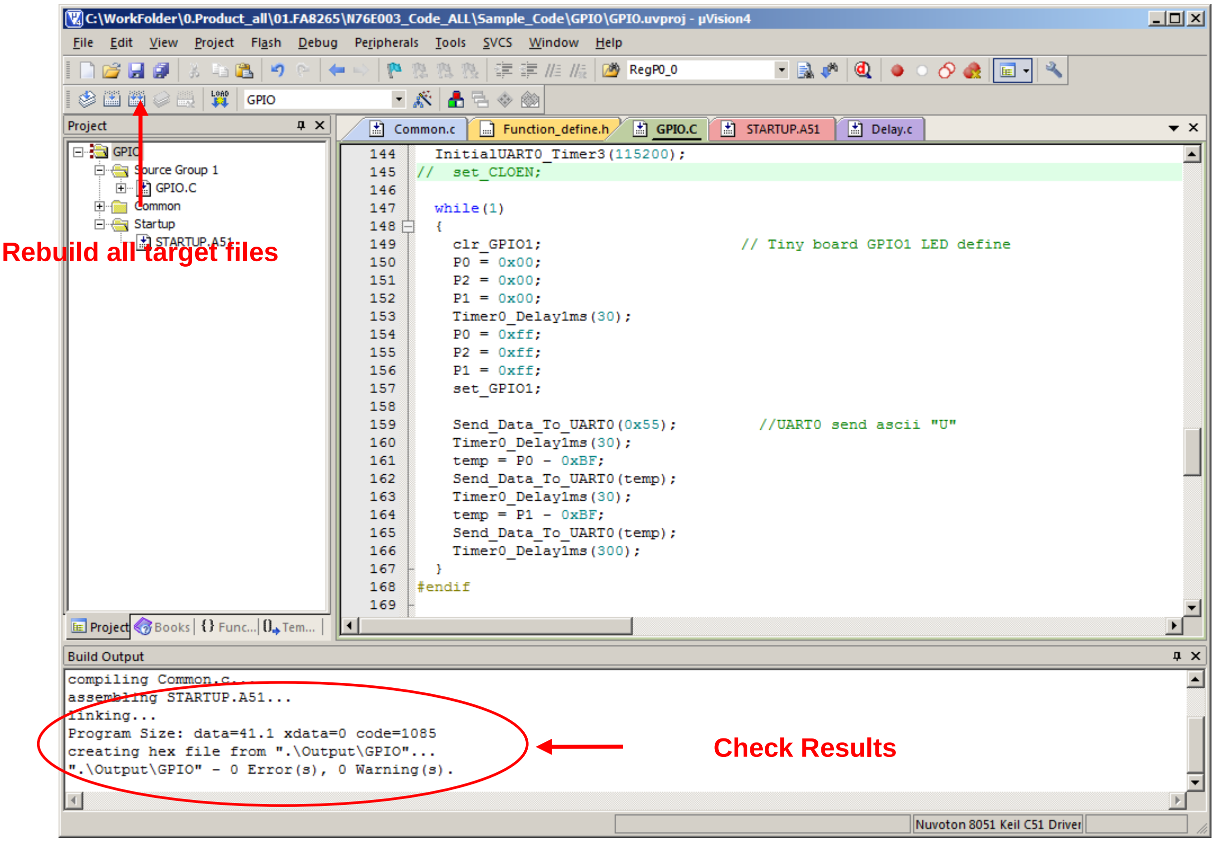


Figure 5-3 Add the C File in Your Project

Step–4: Invoke "**Rebuild all target files**" to build the project. Check if the *Build Output* window shows no warnings and errors.


C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
GPIO
Project
GPIO
Source Group 1
GPIO.C
Common
Startup
STARTUP.A51
Common.c
Function_define.h
GPIO.C
STARTUP.A51
Delay.c
Project
Books
{} Func...
0+ Tem...
Build Output
Nuvoton 8051 Keil C51 Driver

```
  InitialUART0_Timer3(115200);
//  set_CLOEN;

  while(1)
  {
    clr_GPIO1;                    // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
    P1 = 0x00;
    Timer0_Delay1ms(30);
    P0 = 0xff;
    P2 = 0xff;
    P1 = 0xff;
    set_GPIO1;

    Send_Data_To_UART0(0x55);       //UART0 send ascii "U"
    Timer0_Delay1ms(30);
    temp = P0 - 0xBF;
    Send_Data_To_UART0(temp);
    Timer0_Delay1ms(30);
    temp = P1 - 0xBF;
    Send_Data_To_UART0(temp);
    Timer0_Delay1ms(300);
  }
#endif
```

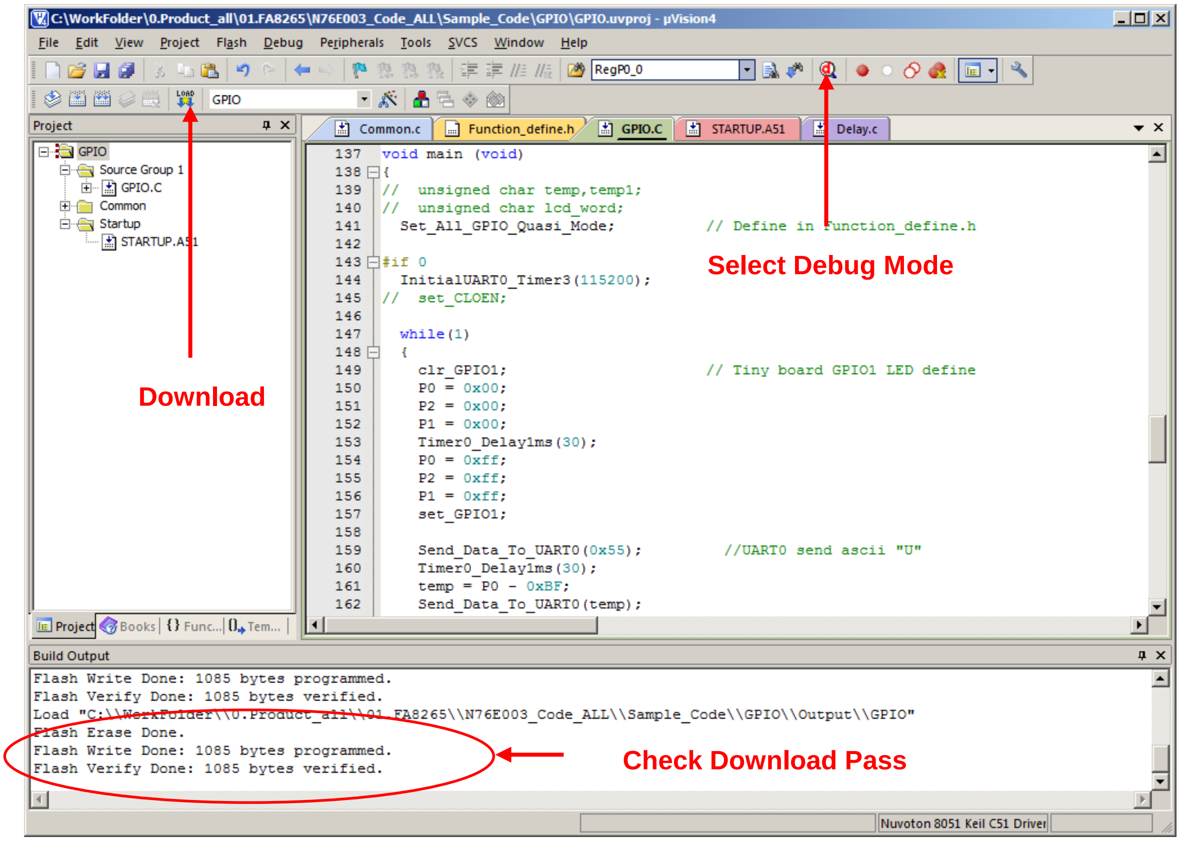
```
compiling Common.c...
assembling STARTUP.A51...
linking...
Program Size: data=41.1 xdata=0 code=1085
creating hex file from ".\Output\GPIO"...
".\Output\GPIO" - 0 Error(s), 0 Warning(s).
```

Rebuild all target files

Check Results

Figure 5-4 Build Target and Display Results

Step–5: Click “**Download**” to download the code to flash memory.


C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
GPIO
Project
GPIO
Source Group 1
GPIO.C
Common
Startup
STARTUP.A51
Common.c
Function_define.h
GPIO.C
STARTUP.A51
Delay.c
Project
Books
{} Func...
0→ Tem...
Build Output
Nuvoton 8051 Keil C51 Driver

Download

Select Debug Mode

```
void main (void)
{
//  unsigned char temp,temp1;
//  unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;           // Define in Function_define.h

#if 0
  InitialUART0_Timer3(115200);
//  set_CLOEN;

  while(1)
  {
    clr_GPIO1;                       // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
    P1 = 0x00;
    Timer0_Delay1ms(30);
    P0 = 0xff;
    P2 = 0xff;
    P1 = 0xff;
    set_GPIO1;

    Send_Data_To_UART0(0x55);          //UART0 send ascii "U"
    Timer0_Delay1ms(30);
    temp = P0 - 0xBF;
    Send_Data_To_UART0(temp);
```

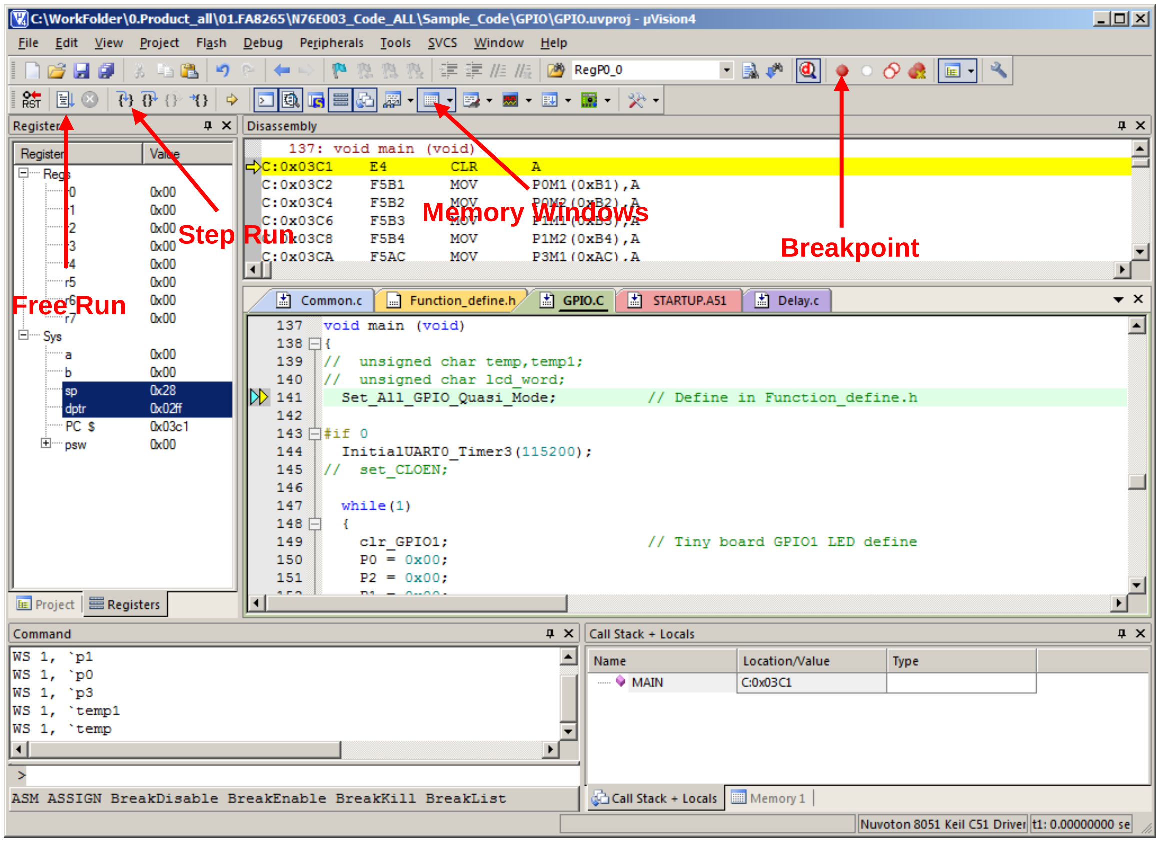
```
Flash Write Done: 1085 bytes programmed.
Flash Verify Done: 1085 bytes verified.
Load "C:\\WorkFolder\\0.Product_all\\01.FA8265\\N76E003_Code_ALL\\Sample_Code\\GPIO\\Output\\GPIO"
Flash Erase Done.
Flash Write Done: 1085 bytes programmed.
Flash Verify Done: 1085 bytes verified.
```

Check Download Pass

Figure 5-5 Download and Display Results

Step–6: Start debugging by clicking “**Start/Stop Debug Session**”.

Step–7: Select to free run, halt or run step by step. Breakpoints can be set or cleared, variables can be displayed and memory areas can be read, written or modified (refer to *Figure 5-6*).


C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
RST
Registers
Disassembly
Free Run
Step Run
Memory Windows
Breakpoint
Common.c
Function_define.h
GPIO.C
STARTUP.A51
Delay.c
Project
Registers
Command
Call Stack + Locals
Memory 1
Nuvoton 8051 Keil C51 Driver
t1: 0.00000000 se

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x28
dptr	0x02ff
PC $	0x03c1
psw	0x00

```
    137: void main (void)
C:0x03C1    E4       CLR      A
C:0x03C2    F5B1     MOV      P0M1(0xB1),A
C:0x03C4    F5B2     MOV      P0M2(0xB2),A
C:0x03C6    F5B3     MOV      P1M1(0xB3),A
C:0x03C8    F5B4     MOV      P1M2(0xB4),A
C:0x03CA    F5AC     MOV      P3M1(0xAC),A
```

```
void main (void)
{
//  unsigned char temp,temp1;
//  unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;          // Define in Function_define.h

#if 0
  InitialUART0_Timer3(115200);
//  set_CLOEN;

  while(1)
  {
    clr_GPIO1;                      // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
```

```
WS 1, `p1
WS 1, `p0
WS 1, `p3
WS 1, `temp1
WS 1, `temp
>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList
```

Name	Location/Value	Type
MAIN	C:0x03C1	

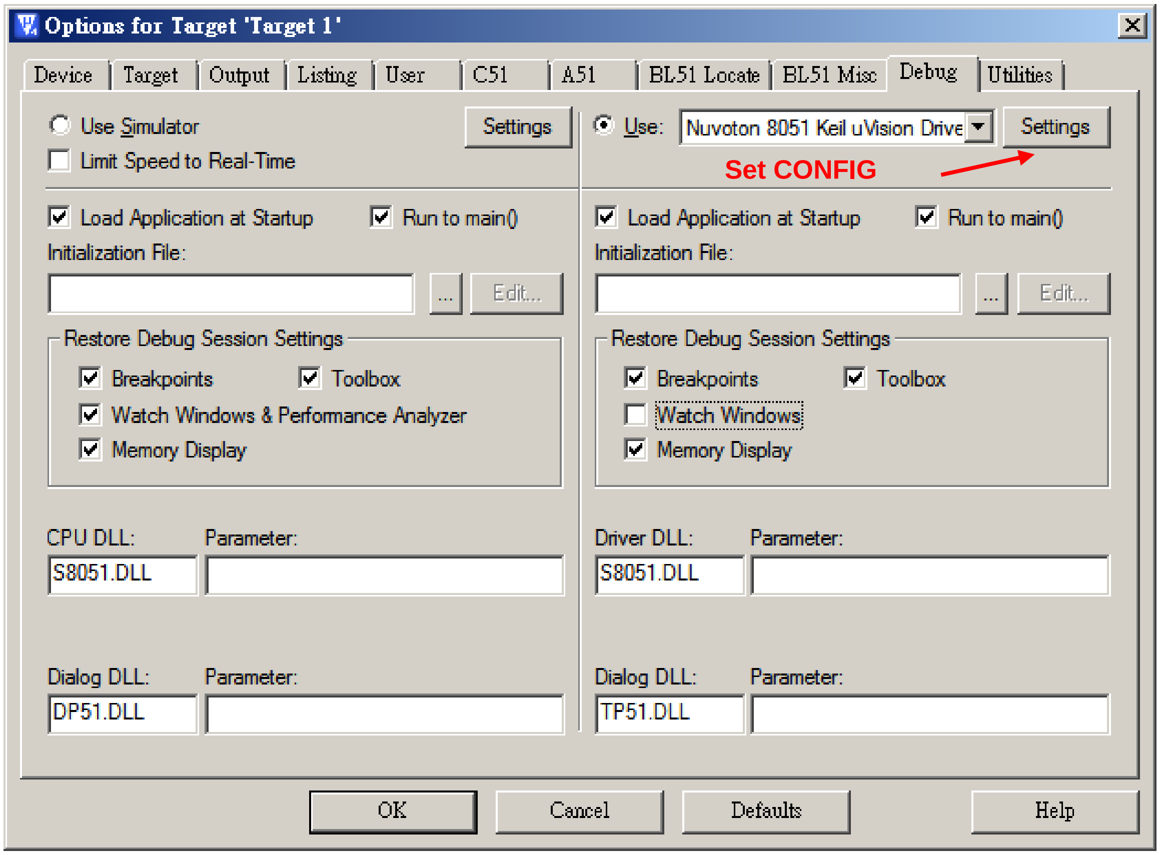
Figure 5-6 Execute Debug Mode

NOTE: Nuvoton ICE/OCD supports eight breakpoints. One breakpoint is reserved for the debug command, and the other seven breakpoints can be used.

5.2 Updating CONFIG

Step–1: Select the Debug tab in the “*Options for Target*” form.

Step–2: Click the “**Settings**” button.



Options for Target 'Target 1'

○ Use Simulator

□ Limit Speed to Real-Time

☑ Load Application at Startup ☑ Run to main()

Initialization File:

Restore Debug Session Settings

☑ Breakpoints ☑ Toolbox

☑ Watch Windows & Performance Analyzer

☑ Memory Display

CPU DLL: S8051.DLL Parameter:

Dialog DLL: DP51.DLL Parameter:

● Use: Nuvoton 8051 Keil uVision Drive

☑ Load Application at Startup ☑ Run to main()

Initialization File:

Restore Debug Session Settings

☑ Breakpoints ☑ Toolbox

□ Watch Windows

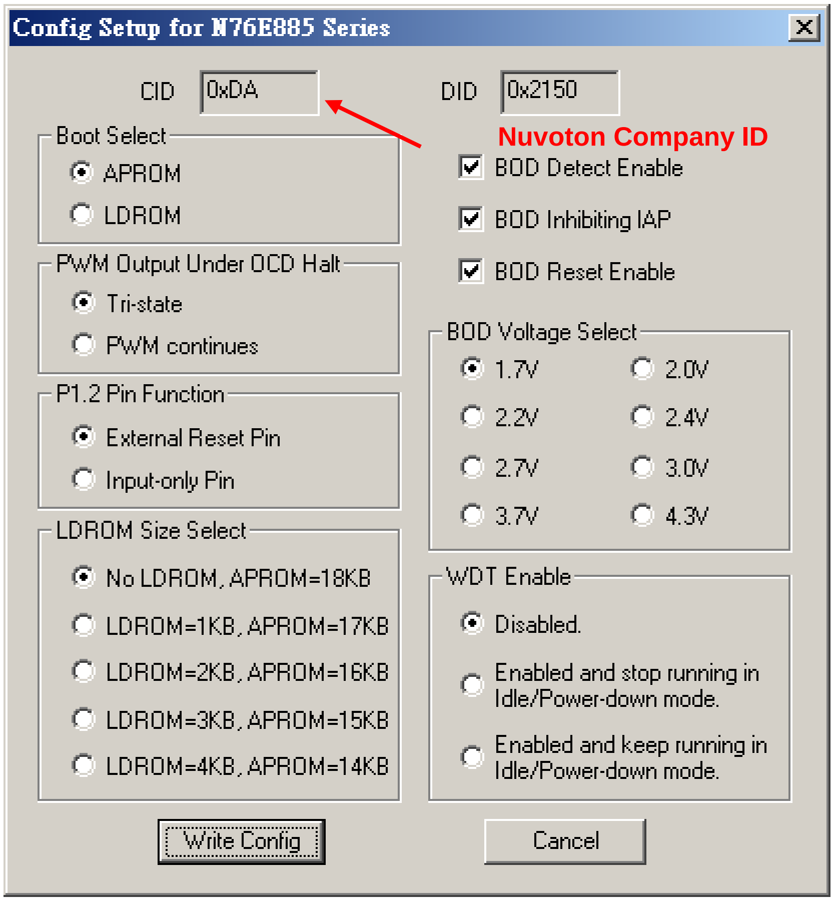
☑ Memory Display

Driver DLL: S8051.DLL Parameter:

Dialog DLL: TP51.DLL Parameter:

Figure 5-7 Set CONFIG Button

Step_3: Set *CONFIG* and click “**Write Config**“.



Config Setup for N76E885 Series

CID 0xDA

DID 0x2150

Nuvoton Company ID

Boot Select
- ● APROM
- ○ LDROM

PWM Output Under OCD Halt
- ● Tri-state
- ○ PWM continues

P1.2 Pin Function
- ● External Reset Pin
- ○ Input-only Pin

LDROM Size Select
- ● No LDROM, APROM=18KB
- ○ LDROM=1KB, APROM=17KB
- ○ LDROM=2KB, APROM=16KB
- ○ LDROM=3KB, APROM=15KB
- ○ LDROM=4KB, APROM=14KB

☑ BOD Detect Enable

☑ BOD Inhibiting IAP

☑ BOD Reset Enable

BOD Voltage Select
- ● 1.7V
- ○ 2.0V
- ○ 2.2V
- ○ 2.4V
- ○ 2.7V
- ○ 3.0V
- ○ 3.7V
- ○ 4.3V

WDT Enable
- ● Disabled.
- ○ Enabled and stop running in Idle/Power-down mode.
- ○ Enabled and keep running in Idle/Power-down mode.

Figure 5-8 Set CONFIG Content

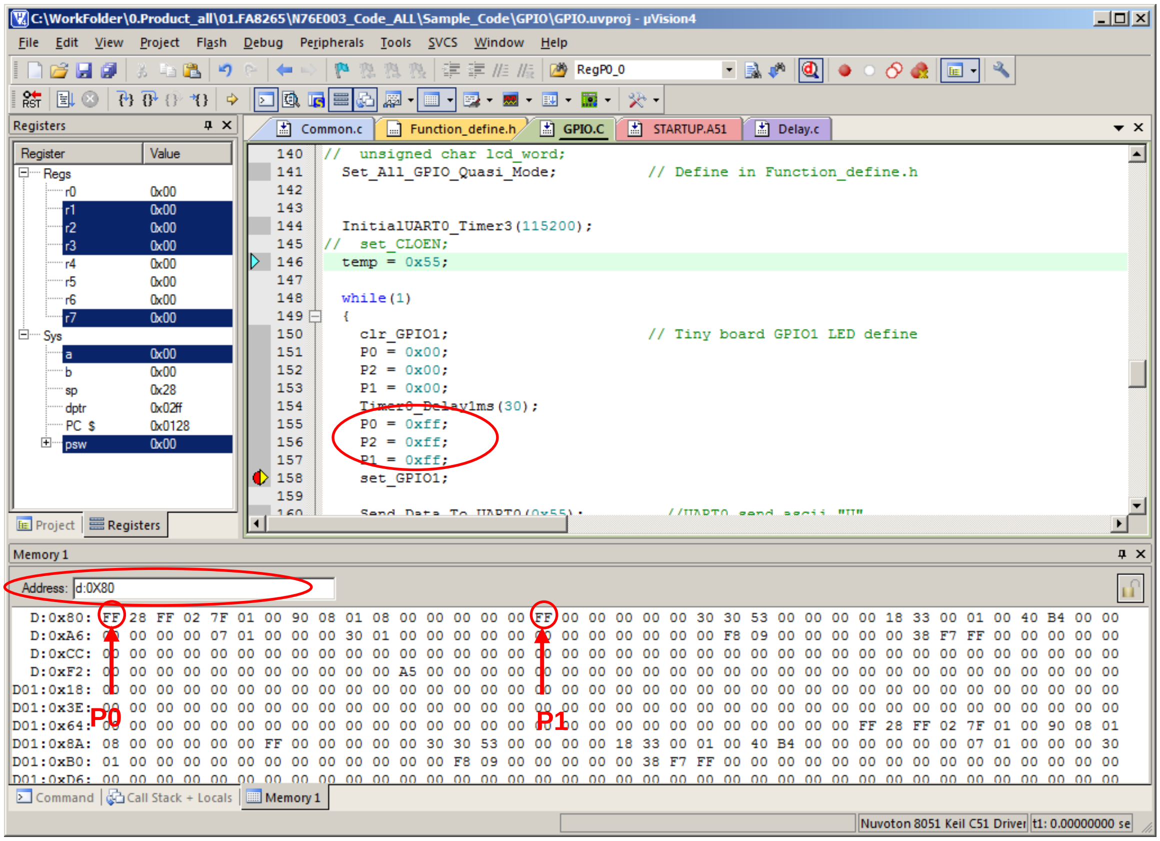
NOTE: **a.** Flash should be re-downloaded after clicking “**Write Config**”.

b. CID must be “**0xDA**” which is the “**Company ID**” (*Nuvoton* ID).

c. DID and CONFIG contents vary from different part numbers.

5.3 Read or Write SFR

Under debug mode, enter “**D: 0x80**” into the Address text field in the *Memory* window. “**D: 0x80**” means that the SFR address start 0x80 to 0xFF. SFRs is generally read or written.



C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
Registers
Common.c
Function_define.h
GPIO.C
STARTUP.A51
Delay.c
Project
Registers
Memory 1
Command
Call Stack + Locals
Memory 1
Nuvoton 8051 Keil C51 Driver
t1: 0.00000000 se
P0
P1

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x28
dptr	0x02ff
PC $	0x0128
psw	0x00

```
//   unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;              // Define in Function_define.h


  InitialUART0_Timer3(115200);
//  set_CLOEN;
  temp = 0x55;

  while(1)
  {
    clr_GPIO1;                          // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
    P1 = 0x00;
    Timer0_Delay1ms(30);
    P0 = 0xff;
    P2 = 0xff;
    P1 = 0xff;
    set_GPIO1;

    Send_Data_To_UART0(0x55);             //UART0 send ascii "U"
```

Address: d:0X80

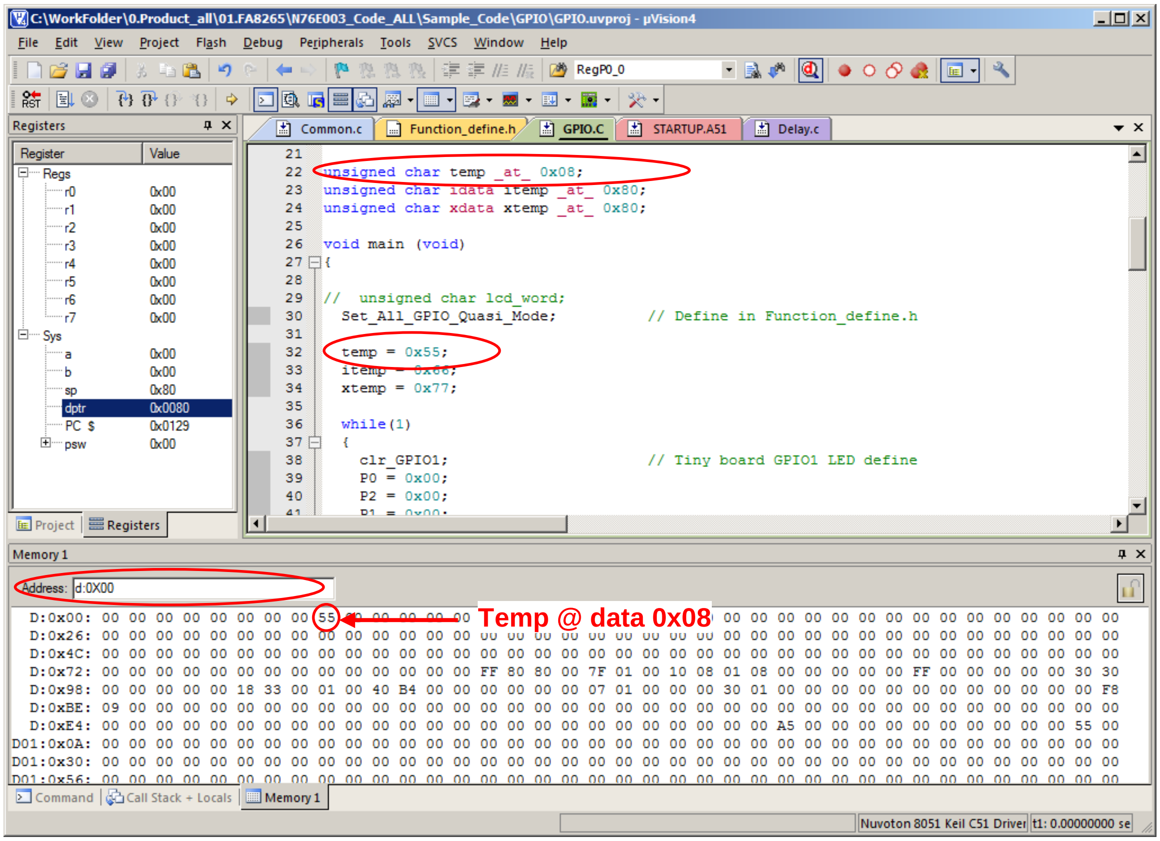
```
  D:0x80: FF 28 FF 02 7F 01 00 90 08 01 08 00 00 00 00 00 FF 00 00 00 00 00 30 30 53 00 00 00 00 18 33 00 01 00 40 B4 00 00
  D:0xA6: 00 00 00 00 07 01 00 00 00 30 01 00 00 00 00 00 00 00 00 00 00 00 00 F8 09 00 00 00 00 00 38 F7 FF 00 00 00 00 00
  D:0xCC: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  D:0xF2: 00 00 00 00 00 00 00 00 00 00 00 A5 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D01:0x18: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D01:0x3E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D01:0x64: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 28 FF 02 7F 01 00 90 08 01
D01:0x8A: 08 00 00 00 00 00 FF 00 00 00 00 00 30 30 53 00 00 00 00 18 33 00 01 00 40 B4 00 00 00 00 00 00 07 01 00 00 00 30
D01:0xB0: 01 00 00 00 00 00 00 00 00 00 00 00 00 F8 09 00 00 00 00 00 38 F7 FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D01:0xD6: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Z

Figure 5-9 Read/Write SFRs

5.4 Read or Write Direct RAM

Under debug mode, enter “**D: 0x00**” into the Address text field in the *Memory* window. “**D: 0x00**” means that the direct RAM address start 0x00 to 0x7F. Direct RAM can be read or written.



C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
Registers
Common.c
Function_define.h
GPIO.C
STARTUP.A51
Delay.c
Project
Registers
Memory 1
Command
Call Stack + Locals
Memory 1
Nuvoton 8051 Keil C51 Driver
t1: 0.00000000 se
Temp @ data 0x08

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x80
dptr	0x0080
PC $	0x0129
psw	0x00

```c

unsigned char temp _at_ 0x08;
unsigned char idata itemp _at_ 0x80;
unsigned char xdata xtemp _at_ 0x80;

void main (void)
{

//  unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;          // Define in Function_define.h

  temp = 0x55;
  itemp = 0x66;
  xtemp = 0x77;

  while(1)
  {
    clr_GPIO1;                      // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
    P1 = 0x00;
```

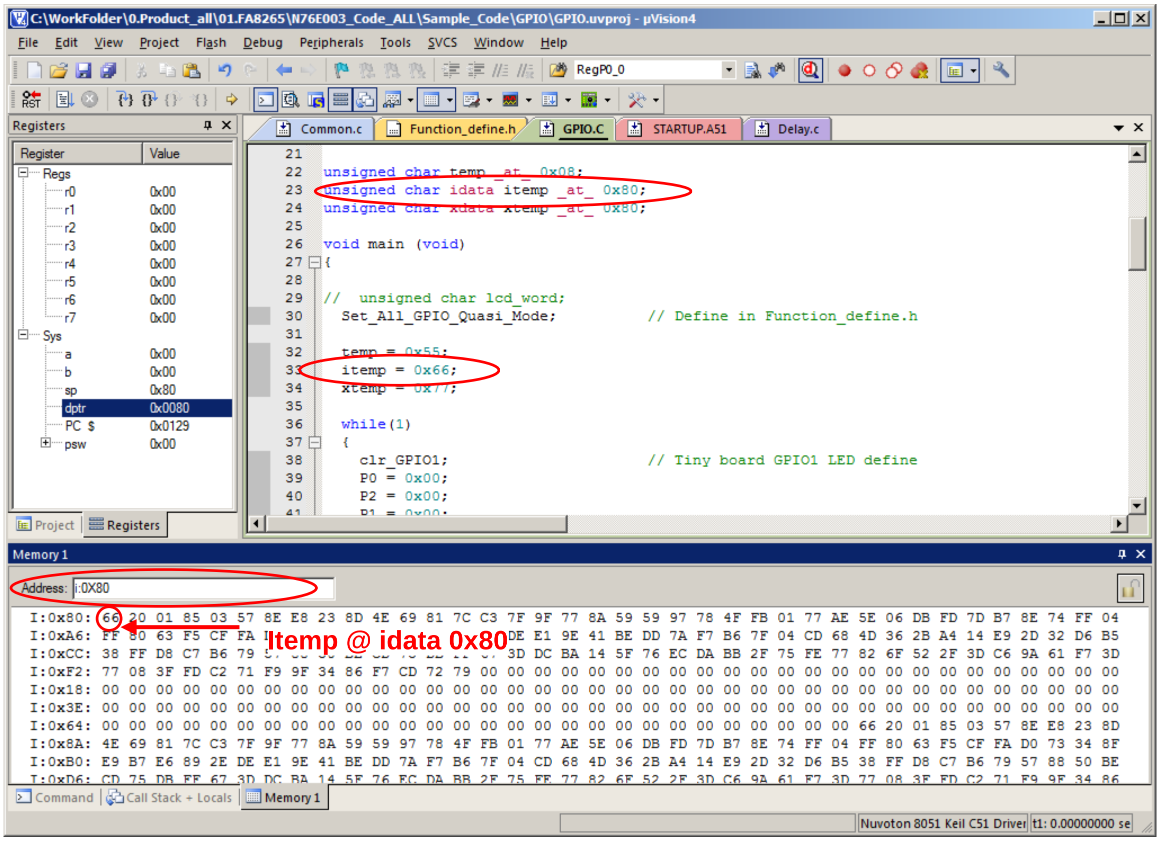
Address: d:0X00

```
  D:0x00: 00 00 00 00 00 00 00 00 55 00 00 00 00 00                            00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  D:0x26: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  D:0x4C: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  D:0x72: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 80 80 00 7F 01 00 10 08 01 08 00 00 00 00 00 FF 00 00 00 00 00 30 30
  D:0x98: 00 00 00 00 00 18 33 00 01 00 40 B4 00 00 00 00 00 00 07 01 00 00 00 30 01 00 00 00 00 00 00 00 00 00 00 00 00 F8
  D:0xBE: 09 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  D:0xE4: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 A5 00 00 00 00 00 00 00 00 00 00 00 55 00
D01:0x0A: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D01:0x30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D01:0x56: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Figure 5-10 Read/Write Direct RAM

5.5 Read or Write Indirect RAM

Under debug mode, enter “**I: 0x00**” into the Address text field in the *Memory* window. “**I: 0x00**” means that the indirect RAM address start 0x00 to 0xFF. Indirect RAM can be read or written.



C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x80
dptr	0x0080
PC $	0x0129
psw	0x00

```

unsigned char temp _at_ 0x08;
unsigned char idata itemp _at_ 0x80;
unsigned char xdata xtemp _at_ 0x80;

void main (void)
{

//  unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;          // Define in Function_define.h

  temp = 0x55;
  itemp = 0x66;
  xtemp = 0x77;

  while(1)
  {
    clr_GPIO1;                      // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
    P1 = 0x00;
```

Address: i:0X80

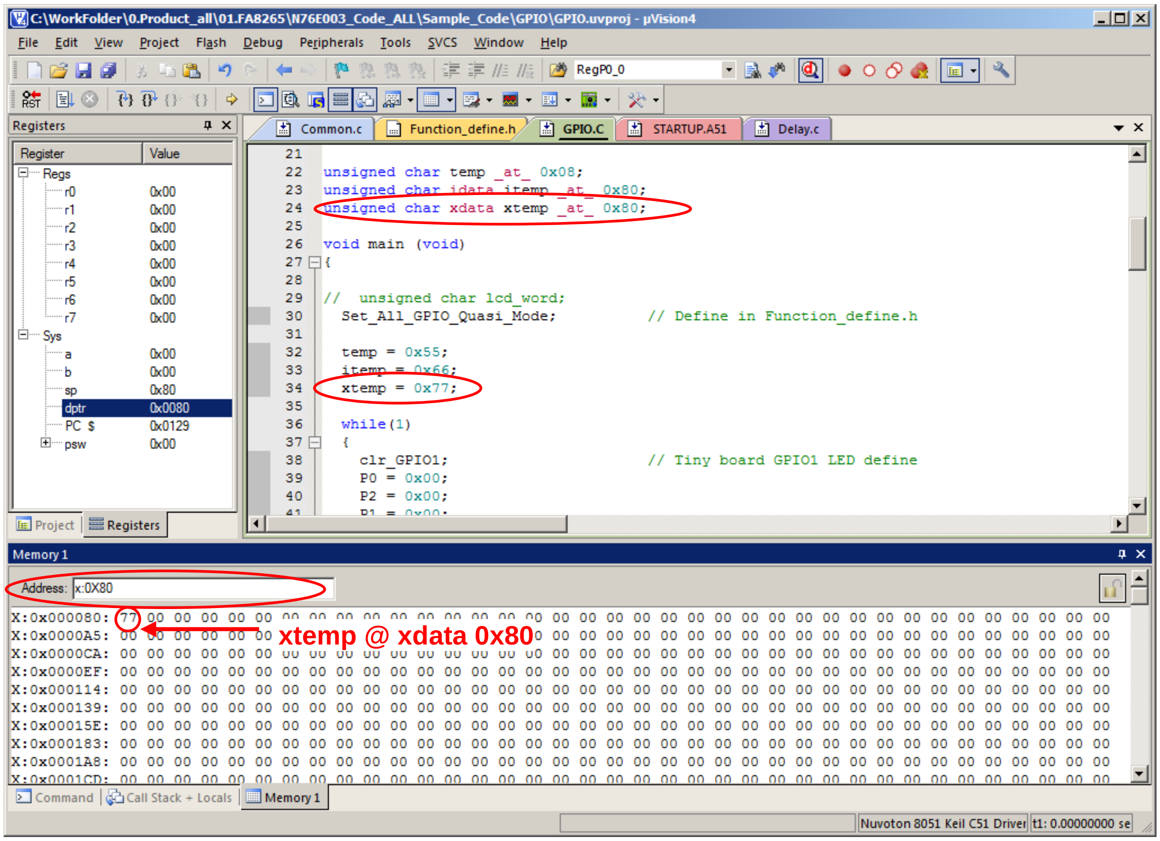
```
I:0x80: 66 20 01 85 03 57 8E E8 23 8D 4E 69 81 7C C3 7F 9F 77 8A 59 59 97 78 4F FB 01 77 AE 5E 06 DB FD 7D B7 8E 74 FF 04
I:0xA6: FF 80 63 F5 CF FA [illegible] DE E1 9E 41 BE DD 7A F7 B6 7F 04 CD 68 4D 36 2B A4 14 E9 2D 32 D6 B5
I:0xCC: 38 FF D8 C7 B6 79 [illegible] 3D DC BA 14 5F 76 EC DA BB 2F 75 FE 77 82 6F 52 2F 3D C6 9A 61 F7 3D
I:0xF2: 77 08 3F FD C2 71 F9 9F 34 86 F7 CD 72 79 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x18: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x3E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x64: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 66 20 01 85 03 57 8E E8 23 8D
I:0x8A: 4E 69 81 7C C3 7F 9F 77 8A 59 59 97 78 4F FB 01 77 AE 5E 06 DB FD 7D B7 8E 74 FF 04 FF 80 63 F5 CF FA D0 73 34 8F
I:0xB0: E9 B7 E6 89 2E DE E1 9E 41 BE DD 7A F7 B6 7F 04 CD 68 4D 36 2B A4 14 E9 2D 32 D6 B5 38 FF D8 C7 B6 79 57 88 50 BE
I:0xD6: CD 75 DB FF 67 3D DC BA 14 5F 76 EC DA BB 2F 75 FE 77 82 6F 52 2F 3D C6 9A 61 F7 3D 77 08 3F FD C2 71 F9 9F 34 86
```

Nuvoton 8051 Keil C51 Driver t1: 0.00000000 se

Figure 5-11 Read/Write Indirect RAM

5.6 Read or Write XRAM

Under debug mode, enter “**X: 0x0000**” into the Address text field in the *Memory* window. “**X: 0x0000**” means that the XRAM start address at 0x0000. XRAM can be read or written.


C:\WorkFolder\0.Product_all\01.FA8265\N76E003_Code_ALL\Sample_Code\GPIO\GPIO.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
RegP0_0
Registers
Common.c
Function_define.h
GPIO.C
STARTUP.A51
Delay.c
Project
Registers
Memory 1
Command
Call Stack + Locals
Memory 1
Nuvoton 8051 Keil C51 Driver
t1: 0.00000000 se
xtemp @ xdata 0x80

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x00
b	0x00
sp	0x80
dptr	0x0080
PC $	0x0129
psw	0x00

```

unsigned char temp _at_ 0x08;
unsigned char idata itemp _at_ 0x80;
unsigned char xdata xtemp _at_ 0x80;

void main (void)
{

//  unsigned char lcd_word;
  Set_All_GPIO_Quasi_Mode;          // Define in Function_define.h

  temp = 0x55;
  itemp = 0x66;
  xtemp = 0x77;

  while(1)
  {
    clr_GPIO1;                      // Tiny board GPIO1 LED define
    P0 = 0x00;
    P2 = 0x00;
    P1 = 0x00;
```

Address: x:0X80

```
X:0x000080: 77 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x0000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x0000CA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x0000EF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x000114: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x000139: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x00015E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x000183: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x0001A8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
X:0x0001CD: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

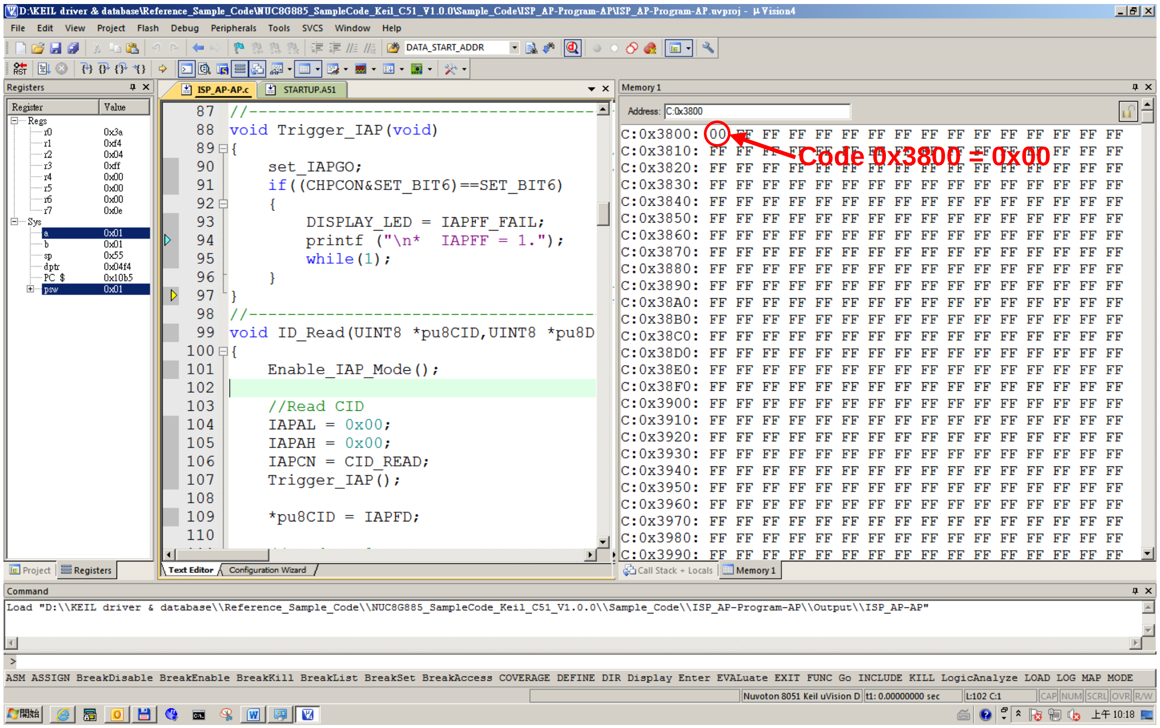
Figure 5-12 Read/Write XRAM

NOTE: If the MCU does not have XRAM, the setting is invalid.

5.7 Read Code

Under debug mode, enter “**C: 0x0000**” into the Address text field in the *Memory* window. “**C:0x0000**” means that the CODE start address at 0x0000. CODE is read only. Reading CODE area benefits debugging especially when Data Flash storage is used during execution.

NOTE: This feature is valid for ML51 series, MS51 series and N76 series.


D:\KEIL driver & database\Reference_Sample_Code\NUC8G885_SampleCode_Keil_C51_V1.0.0\Sample_Code\ISP_AP-Program-AP\ISP_AP-Program-AP.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
DATA_START_ADDR
RST
Registers
ISP_AP-AP.c
STARTUP.A51
Memory 1
Project
Registers
Text Editor
Configuration Wizard
Call Stack + Locals
Memory 1
Command
Code 0x3800 = 0x00
開始
上午 10:18

Register	Value
Regs	
r0	0x3a
r1	0xf4
r2	0x04
r3	0xff
r4	0x00
r5	0x00
r6	0x00
r7	0x0e
Sys	
a	0x01
b	0x01
sp	0x55
dptr	0x04f4
PC $	0x10b5
psw	0x01

```
//-----------------------------------
void Trigger_IAP(void)
{
    set_IAPGO;
    if((CHPCON&SET_BIT6)==SET_BIT6)
    {
        DISPLAY_LED = IAPFF_FAIL;
        printf ("\n*  IAPFF = 1.");
        while(1);
    }
}
//-----------------------------------
void ID_Read(UINT8 *pu8CID,UINT8 *pu8D
{
    Enable_IAP_Mode();

    //Read CID
    IAPAL = 0x00;
    IAPAH = 0x00;
    IAPCN = CID_READ;
    Trigger_IAP();

    *pu8CID = IAPFD;

```

Address: C:0x3800

```
C:0x3800: 00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3810: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3820: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3830: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3840: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3850: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3860: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3870: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3880: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3890: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x38A0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x38B0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x38C0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x38D0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x38E0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x38F0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3900: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3910: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3920: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3930: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3940: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3950: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3960: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3970: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3980: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
C:0x3990: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```

```
Load "D:\\KEIL driver & database\\Reference_Sample_Code\\NUC8G885_SampleCode_Keil_C51_V1.0.0\\Sample_Code\\ISP_AP-Program-AP\\Output\\ISP_AP-AP"
>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE DEFINE DIR Display Enter EVALuate EXIT FUNC Go INCLUDE KILL LogicAnalyze LOAD LOG MAP MODE
```

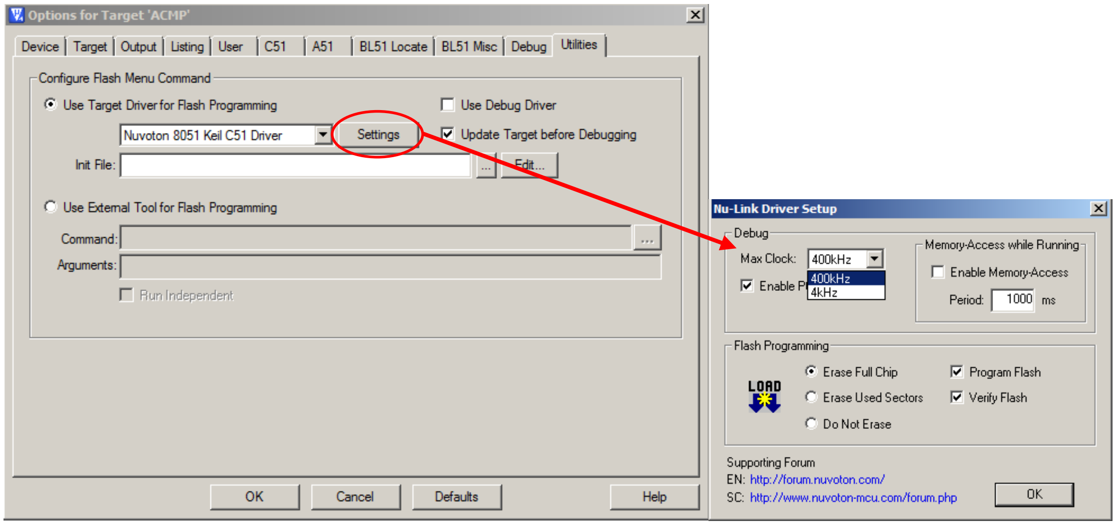
```
Nuvoton 8051 Keil uVision D | t1: 0.00000000 sec | L:102 C:1 | CAP NUM SCRL OVR R/W
```

Figure 5-13 Read/Write Code

5.8 ICE Clock Setting

If target MCU use LIRC or system clock divider under 1MHz, need define the ICE max Clock to 4kHz to confirm the ICE read / write data correctly.

NOTE: This feature is valid for ML51 series, MS51 series and N76 series.



Configure Flash Menu Command

● Use Target Driver for Flash Programming

Nuvoton 8051 Keil C51 Driver

☐ Use Debug Driver

☑ Update Target before Debugging

Init File:

○ Use External Tool for Flash Programming

Command:

Arguments:

☐ Run Independent

Debug

Max Clock: 400kHz

400kHz
4kHz

☑ Enable P

Memory-Access while Running

☐ Enable Memory-Access

Period: 1000 ms

Flash Programming

● Erase Full Chip

○ Erase Used Sectors

○ Do Not Erase

☑ Program Flash

☑ Verify Flash

Supporting Forum

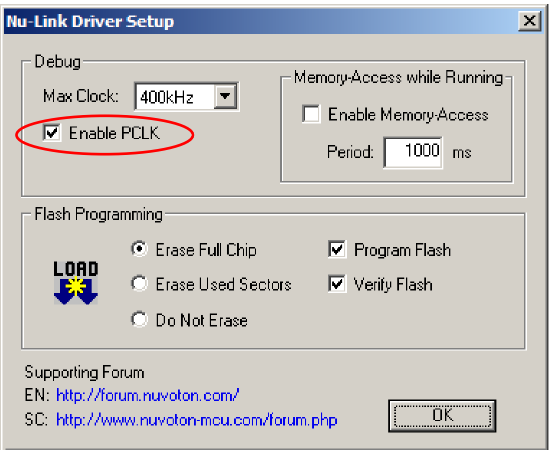
EN: http://forum.nuvoton.com/

SC: http://www.nuvoton-mcu.com/forum.php

5.9 Timer Clock Keep Run Under OCD Halt

For some MCU such as N76E003, CONFIG setting support PWM continues run under OCD halt, means if break point or step run, PWM keep output as normal setting, this function also need ICE driver setup “Enable PCLK” selected.

NOTE: When this function is enabled, not only the PWM clock keep running and output, all Timer include Timer 0/1/2/3 & WKT) also keep count. This feature is valid for ML51 series, MS51 series and N76 series.



Nu-Link Driver Setup

Debug

Max Clock: 400kHz

☑ Enable PCLK

Memory-Access while Running

☐ Enable Memory-Access

Period: 1000 ms

Flash Programming

● Erase Full Chip

○ Erase Used Sectors

○ Do Not Erase

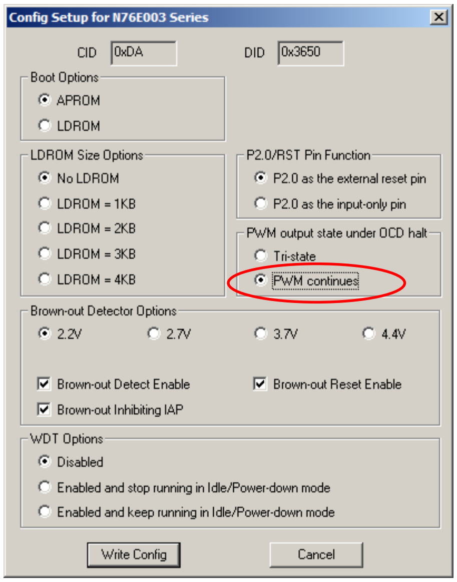
☑ Program Flash

☑ Verify Flash

Supporting Forum

EN: http://forum.nuvoton.com/

SC: http://www.nuvoton-mcu.com/forum.php



Config Setup for N76E003 Series

CID 0xDA DID 0x3650

Boot Options

● APROM

○ LDROM

LDROM Size Options

● No LDROM

○ LDROM = 1KB

○ LDROM = 2KB

○ LDROM = 3KB

○ LDROM = 4KB

P2.0/RST Pin Function

● P2.0 as the external reset pin

○ P2.0 as the input-only pin

PWM output state under OCD halt

○ Tri-state

● PWM continues

Brown-out Detector Options

● 2.2V ○ 2.7V ○ 3.7V ○ 4.4V

☑ Brown-out Detect Enable

☑ Brown-out Reset Enable

☑ Brown-out Inhibiting IAP

WDT Options

● Disabled

○ Enabled and stop running in Idle/Power-down mode

○ Enabled and keep running in Idle/Power-down mode

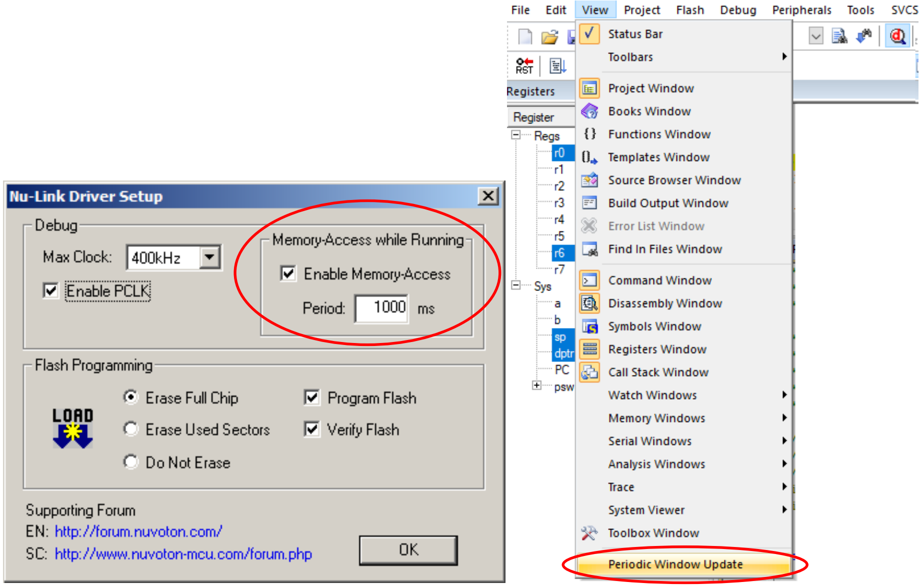
5.10 Memory Access while Runing

When driver setup window with memory access running function item means support SFR and RAM value keep update when ICE in free run mode.

Customer can define the period of refresh timing, the minima value is 100ms.

Setting as following and enable “Periodic Windows Updated” in debug mode.

NOTE: Since the refresh process is stop the MCU running, update memory windown then running MCU again, so this maybe cause some timing issue not match the real clock when this function is enabled. This feature is valid for ML51 series, MS51 series and N76 series.


Nu-Link Driver Setup

Debug

Max Clock: 400kHz

☑ Enable PCLK

Memory Access while Running

☑ Enable Memory Access

Period: 1000 ms

Flash Programming

● Erase Full Chip ☑ Program Flash

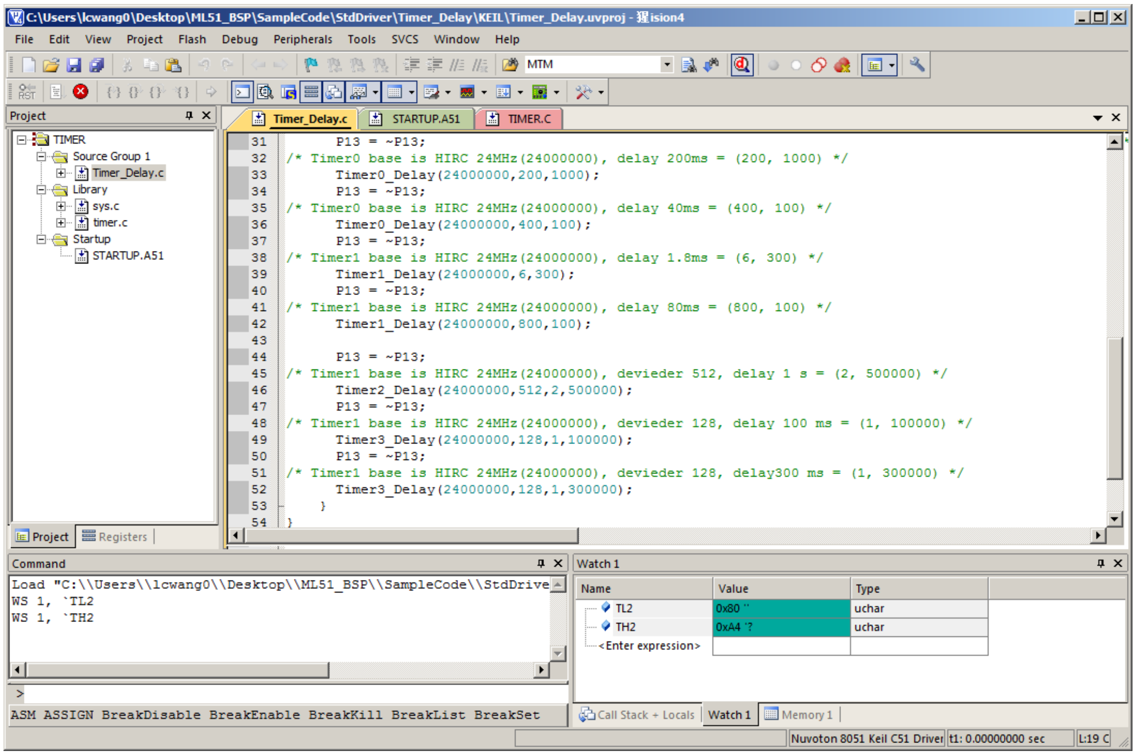
○ Erase Used Sectors ☑ Verify Flash

○ Do Not Erase

Supporting Forum

EN: http://forum.nuvoton.com/

SC: http://www.nuvoton-mcu.com/forum.php


C:\Users\lcwang0\Desktop\ML51_BSP\SampleCode\StdDriver\Timer_Delay\KEIL\Timer_Delay.uvproj - µVision4

```
        P13 = ~P13;
/* Timer0 base is HIRC 24MHz(24000000), delay 200ms = (200, 1000) */
        Timer0_Delay(24000000,200,1000);
        P13 = ~P13;
/* Timer0 base is HIRC 24MHz(24000000), delay 40ms = (400, 100) */
        Timer0_Delay(24000000,400,100);
        P13 = ~P13;
/* Timer1 base is HIRC 24MHz(24000000), delay 1.8ms = (6, 300) */
        Timer1_Delay(24000000,6,300);
        P13 = ~P13;
/* Timer1 base is HIRC 24MHz(24000000), delay 80ms = (800, 100) */
        Timer1_Delay(24000000,800,100);

        P13 = ~P13;
/* Timer1 base is HIRC 24MHz(24000000), devieder 512, delay 1 s = (2, 500000) */
        Timer2_Delay(24000000,512,2,500000);
        P13 = ~P13;
/* Timer1 base is HIRC 24MHz(24000000), devieder 128, delay 100 ms = (1, 100000) */
        Timer3_Delay(24000000,128,1,100000);
        P13 = ~P13;
/* Timer1 base is HIRC 24MHz(24000000), devieder 128, delay300 ms = (1, 300000) */
        Timer3_Delay(24000000,128,1,300000);
    }
}
```

Command

```
Load "C:\\Users\\lcwang0\\Desktop\\ML51_BSP\\SampleCode\\StdDrive
WS 1, `TL2
WS 1, `TH2
>
ASM ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet
```

Watch 1

Name	Value	Type
TL2	0x80 ''	uchar
TH2	0xA4 '?'	uchar
<Enter expression>		

6 PROJECT WORKING FLOW

The chapter provides flows for users to download or execute debug mode in KEIL^{TM} C51 environment.

6.1 Program Code Download Flow

Step–1: Create a new project (refer to *Figure 4-1* ~ *Figure 4-5*).

Step–2: Set debug options (refer to *Figure 4-6* ~ *Figure 4-7*).

Step–3: Create a new C file and write sample code (refer to *Figure 5-1* ~ *Figure 5-2*).

Step–4: Add C file in your project (refer to *Figure 5-3*).

Step–5: Set CONFIG (refer to *Figure 5-7* ~ *Figure 5-8*).

Step–6: Build target and display results (refer to *Figure 5-4*).

Step–7: Download and display results (refer to *Figure 5-5*).

6.2 Program Code Debug Flow

Step–1: Create a new project (refer to *Figure 4-1* ~ *Figure 4-5*).

Step–2: Set debug options (refer to *Figure 4-6* ~ *Figure 4-7*).

Step–3: Create a new C file and write sample code (refer to *Figure 5-1* ~ *Figure 5-2*).

Step–4: Add C file in your project (refer to *Figure 5-3*).

Step–5: Build target and display results (refer to *Figure 5-4*).

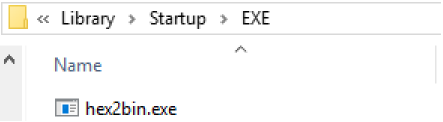
Step–6: Execute debug mode (refer to *Figure 5-5* ~ *Figure 5-6*)

Step–7: Read or write SFRs, direct RAM, indirect RAM and XRAM (refer to *Figure 5-9* ~ *Figure 5-12*).

6.3 Create .bin file

Follow below steps to create bin file from .hex.

Step–1: Make sure hex2bin.exe is in BSP.

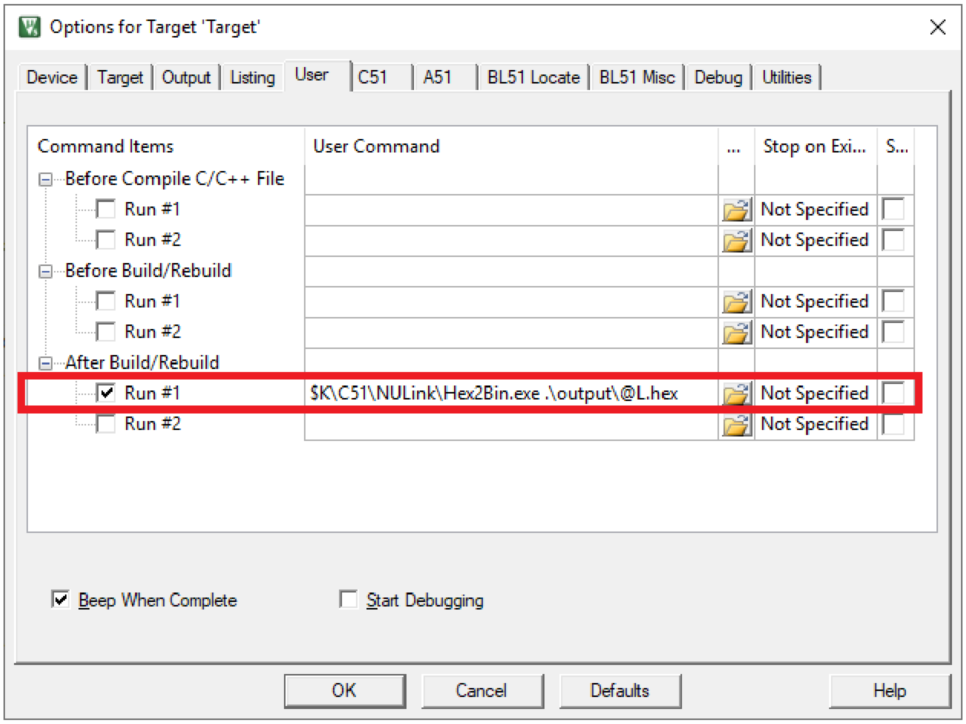


« Library › Startup › EXE

Name

hex2bin.exe

Step–2: Add following setting



Options for Target 'Target'

Command Items	User Command	...	Stop on Exi...	S...
⊟ Before Compile C/C++ File				
□ Run #1			Not Specified	□
□ Run #2			Not Specified	□
⊟ Before Build/Rebuild				
□ Run #1			Not Specified	□
□ Run #2			Not Specified	□
⊟ After Build/Rebuild				
☑ Run #1	$K\C51\NULink\Hex2Bin.exe .\output\@L.hex		Not Specified	□
□ Run #2			Not Specified	□

☑ Beep When Complete □ Start Debugging

Step–3: Example of arguments.

$K\C51\NULink\Hex2Bin.exe .\output\@L.hex	By default, unused location will be filled with 0xFF
$K\C51\NULink\Hex2Bin.exe .\output\@L.hex -P 0x00	With parameter –P or –p filled with data defined following. In this case will be filled with 00.
$K\C51\NULink\Hex2Bin.exe .\output\@L.hex -p 0xFF	Same as –P. in this case filled with FF
$K\C51\NULink\Hex2Bin.exe .\output\@L.hex 0x0 0x1000 0xFF80 0x80	Create two bin file. Unused location filled with 0xFF. The first file <project>.bin data create start from 0x00 and file size is0x1000. The second file <project name1>.bin create start from 0xFF80 file size is 0x80.
$K\C51\NULink\Hex2Bin.exe .\output\@L.hex -P 0x00 0x0 0x1000 0xFF80 0x80	Create two bin file. Unused location filled parameter after –P defined. In this case will be filled with 00.

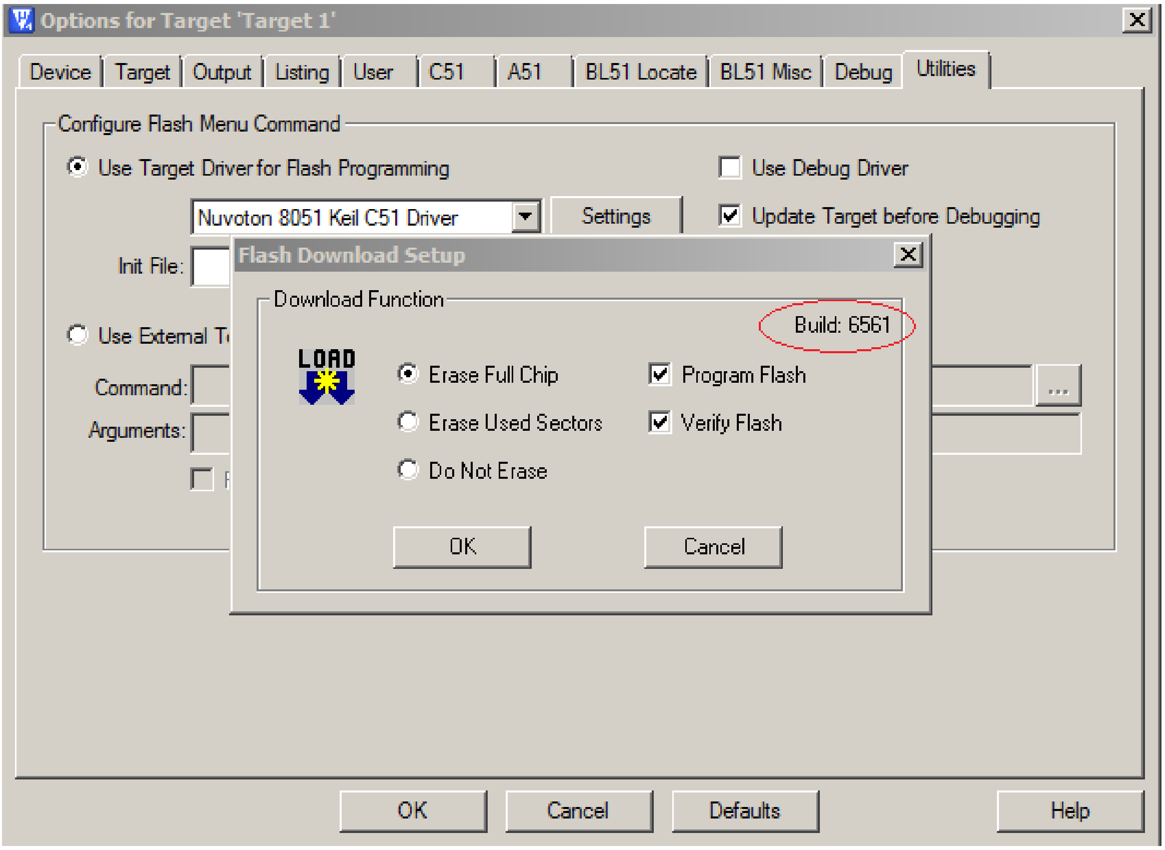
Step-4: Checksum information

>>> [Project Name].bin Checksum = 0xXXXX<<<	By default, one bin file checksum format
>>> [Project Name].bin Checksum = 0xXXXX <<< >>> [Project Name]1.bin Checksum = 0xXXXX <<<	Tow bin file checksum format base on customer range defined.

7 CHECK VERSION

7.1 Nu-Link Bridge Firmware Version

The ICE/OCD debug tool uses Nu-Link as a bridge interface to communicate with KEIL^{TM} debug environment and the target MCU. The main chip of Nu-Link is a Nuvoton NUC12SR MCU. The firmware version programmed in this chip can be checked in “Flash Download Setup” window by clicking “Settings” in “Utilities” page.



Options for Target 'Target 1'

Configure Flash Menu Command

● Use Target Driver for Flash Programming

Nuvoton 8051 Keil C51 Driver

□ Use Debug Driver

☑ Update Target before Debugging

Init File:

○ Use External T

Command:

Arguments:

Flash Download Setup

Download Function

Build: 6561

● Erase Full Chip

○ Erase Used Sectors

○ Do Not Erase

☑ Program Flash

☑ Verify Flash

Figure 7-1 Check Bridge Firmware Version

7.2 Appendix Limitations of KEIL^{TM} C51 Evaluation Edition

KEIL^{TM} development tools without a current product license run as an Evaluation edition and have the following restrictions:

- The 8051 compiler, assembler, linker, and debugger are limited to 2 Kbytes of object code. Source code may be of any size.
- Programs that generate more than 2 Kbytes of object code will not compile, assemble, or link.
- The debugger supports programs that are 2 Kbytes or smaller.
- The startup code generated includes LJMPs. Code generated cannot be used in single-chip devices that support 2 Kbytes or less of program space.
- Programs start at offset 0x0800. Programs generated with the evaluation software may not be programmed into single-chip devices with less than 2 Kbytes of on-chip ROM.
- No hardware support for multiple DPTR registers is provided.
- No support for floating-point arithmetic and no support for user libraries are provided.
- No support for in-line assembly using #pragma ASM.
- The following components which are present in the PK51 Full Version are not included in the Evaluation Version: Linker for Code Banking, Library Manager, and RTX51 Tiny Real-time Operating System.

8 REVISION HISTORY

Date	Revision	Description
1.00	Nov 09,2011	Initial release.
1.01	Nov 17,2011	1. Revised W79E82J CONFIG. 2. Added the free-run button.
1.02	Nov 21,2011	Revised W79E85J CONFIG.
1.03	Mar 05,2012	Supported W79E659.
1.04	Apr 11,2012	1. Supported N79E37J. 2. Added Project Working Flow. 3. Updated format and descriptions.
1.05	Dec 8, 2014	1. Supported NUC8G885. 2. Supported Nuvoton’s own CPU database and header files. 3. Supported CODE area read for NUC8G885. 4. Added how to check version of the debug environment.
1.06	Jan 14, 2015	Renamed NUC8G885 as N76E885.
1.07	Apr 15, 2015	1. Minor modified for N76E885 flash programming. 2. Supported N76E616.
1.08	Oct 08, 2015	Fixed a compile issue for KEIL™ µVision®5.
2.00	Nov 15, 2016	1. Add N76E003 supporting 2. Modify Nu-link-Me as bridge support
2.01	Nov 8, 2017	1. Modify the name of keil driver. 2. Modify RAM picture and description.
2.06	Jan 21, 2019	1. Added ML51 and MS51 in chapter 2 support series table 2. Added chapter 5.8~5.10

Notice: Using this software indicates your acceptance of the disclaimer hereunder:

THIS SOFTWARE IS FOR YOUR REFERENCE ONLY AND PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. YOUR USING THIS SOFTWARE/FIRMWARE IS BASED ON YOUR OWN DISCRETION, IN NO EVENT SHALL THE COPYRIGHT OWNER OR PROVIDER BE LIABLE TO ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.