Development of ET-ARM STAMP LPC2119 Program with GCCARM and Keil uVision3

Generally, GCCARM program is C-Complier Program only but Text Editor Program is not included. So, if you want to develop ARM7 Program with GCCARM Program, you need to create Source Code with other Text Editor Program and save File in C Language. Next, Complier is translated by GCCARM Program via Command Line and users have to understand Complier's option well. Using GCCARM Program isn't convenient for users because they need to alternate program between Text Editor Program and Command Line to translate program. If there's any mistake, users have to reset and create new one. The good point of GCCARM Program is free download without charge.

For users that have Demo Keil-ARM Program can use Text Editor Program of Keil-ARM (Keil uVision3) for writing Source Code in C Language and link Command Line to GCCARM Program for translating Code. Users can use Keil uVision3 as like as Keil-ARM but its restriction is users have to write program in form of GCCARM only. The good point of using GCCARM Program is unlimited size of Output Hex File as well as Demo Keil-ARM Program.

Instructions for using Program

- You can Download Demo Keil-ARM Program from WWW.KEIL.COM and install program in folder named C:\keil as default value of program because of easy to use.

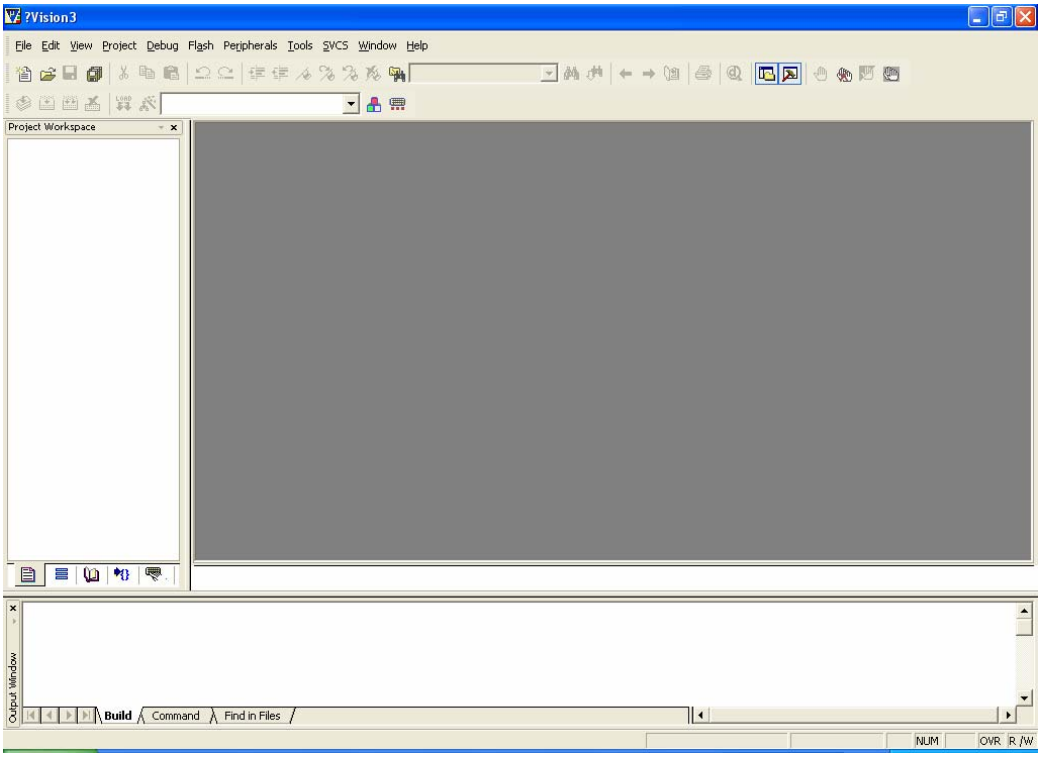
- You can download GCCARM Program from WWW.KEIL.COM and install program in folder named C:\Cygnus as default value of program because of easy to set option.

If you install program different from these, you have to change default value in those program and can refer some part of these instructions not all.

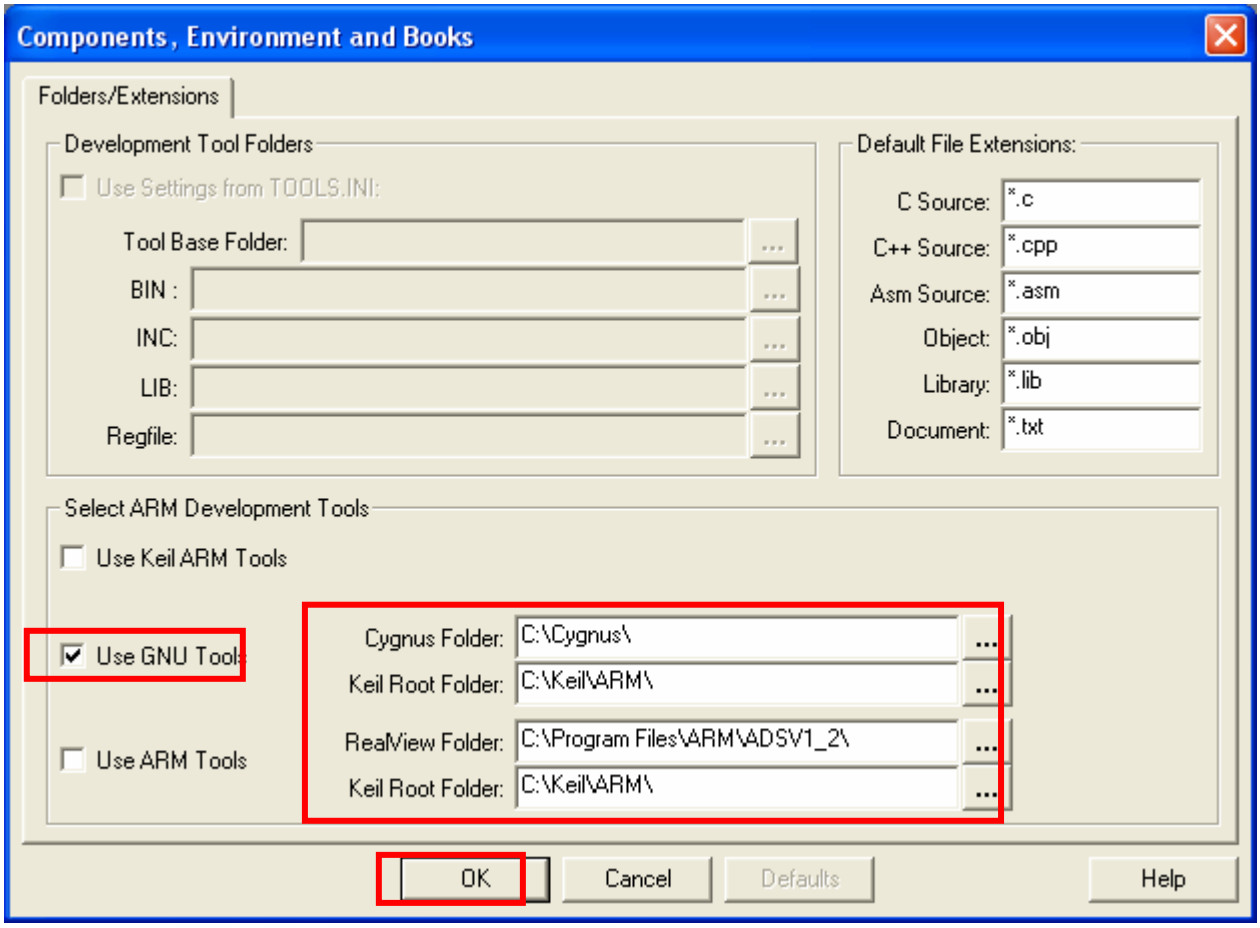
Using Keil uVision3 with GCCARM C-Complier

This section is described how to write C Language Program and translate Code via GCCARM Program under Text Editor Program of Keil (Keil uVision3) and how to set default value of Option to link Code only. If you want to find out more detail about using GCCARM Program, you can learn from user manual of GCCARM Program. Proceeding of setting default value of Keil uVision3 with GCCARM are;

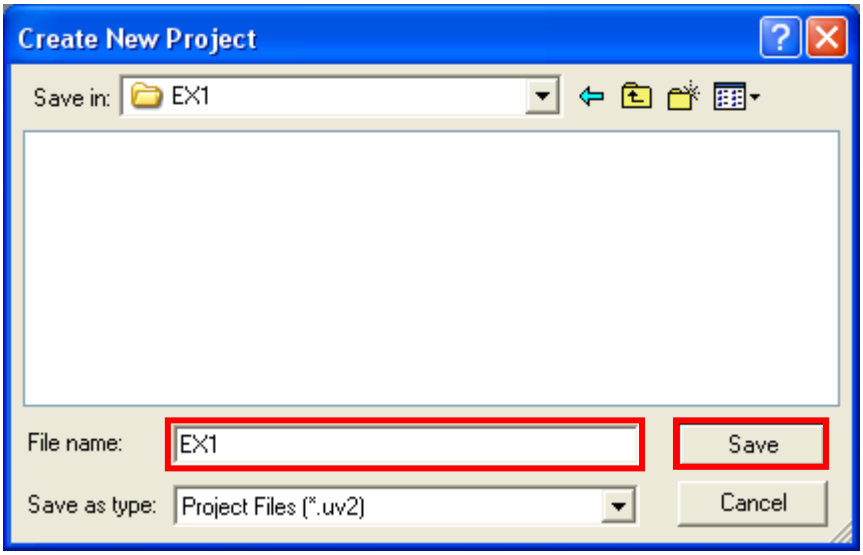
1. Opening Keil uVision3 Program which is Text Editor Program of Keil-ARM uses for writing C Language Source Code as in the picture.



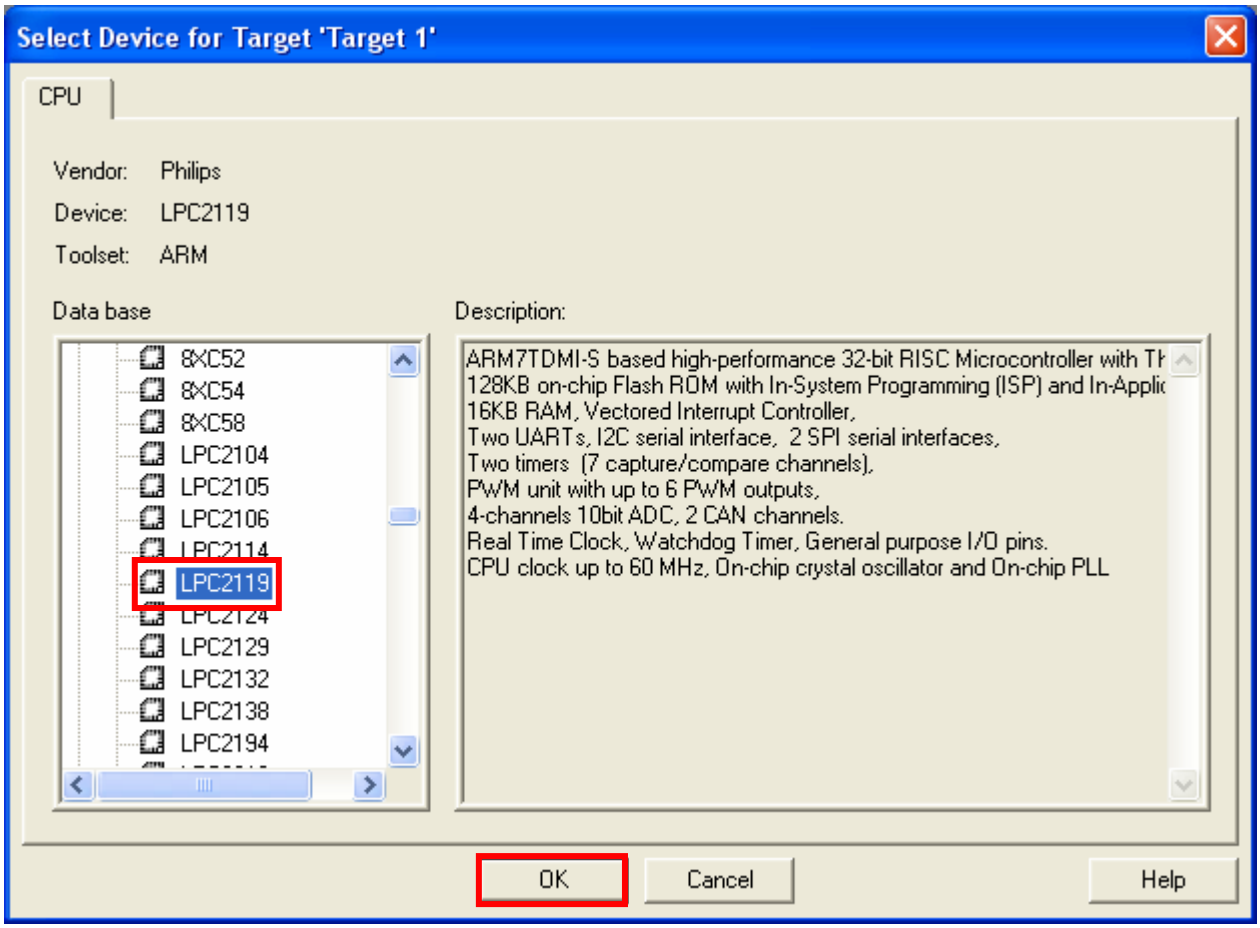
2. Setting default value to translate uVision3 Code to run with Keil uVision3 Program and GCCARM. Click **Project → Components, Environment, Books…** then select default value to use Complier titled **Select ARM Development Tools.** There's 3 default value of them – **Use Keil ARM Tools, Use GNU Tools,** and **Use ARM Tools**– in this case, selecting "**Use GNU Tools**". Next, setting to position of folder that is installed, default value of GCCARM is C:\Cygnus as in the picture (if it isn't this program, you have to change default value correctly.)



3. Creating new Project File by clicking **Project → New Project** then setting the position of folder that you want to save with its title. For example,if you want to create Project File named EX1 and save in folder named EX1, you can set them by yourself. After you fill its name in File name blank, clicking **Save** to save project File as in the picture.

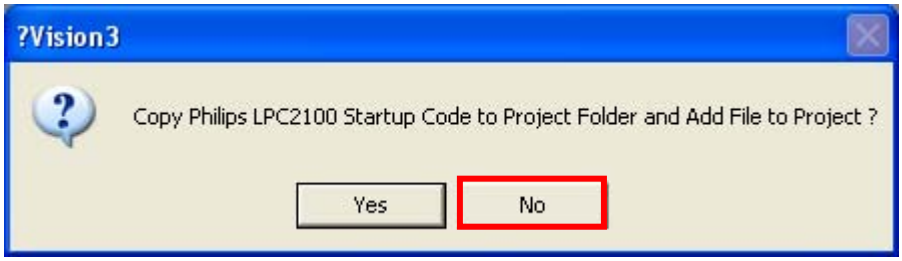


Program proceeding waits for users to set No. of MCU after you saved Project File. No.of Philips'MCU in ET-ARM STAMP LPC2119 Board is LPC2119 then click **OK** as in the picture.



In this step, users need to confirm to copy Startup File with Philips'Mcu or not because Startup File is the beginning of MCU running. For example, you should set the Stack value and setting default value into Phase-Lock-Loop before starting to run program, otherwise MCU is added orders to run by the written program.

Startup File of Keil-ARM is Assembly Language and both of Keil-ARM and GCCARM uses different Assembler Program, so, you can't use them together. Using GCCARM Program to translate orders, users need to create new Startup File to set default value as well as GCCARM form. In this case, select **"No"** to protect Keil uVision3 copy Startup File of Keil-ARM to use in this Project.



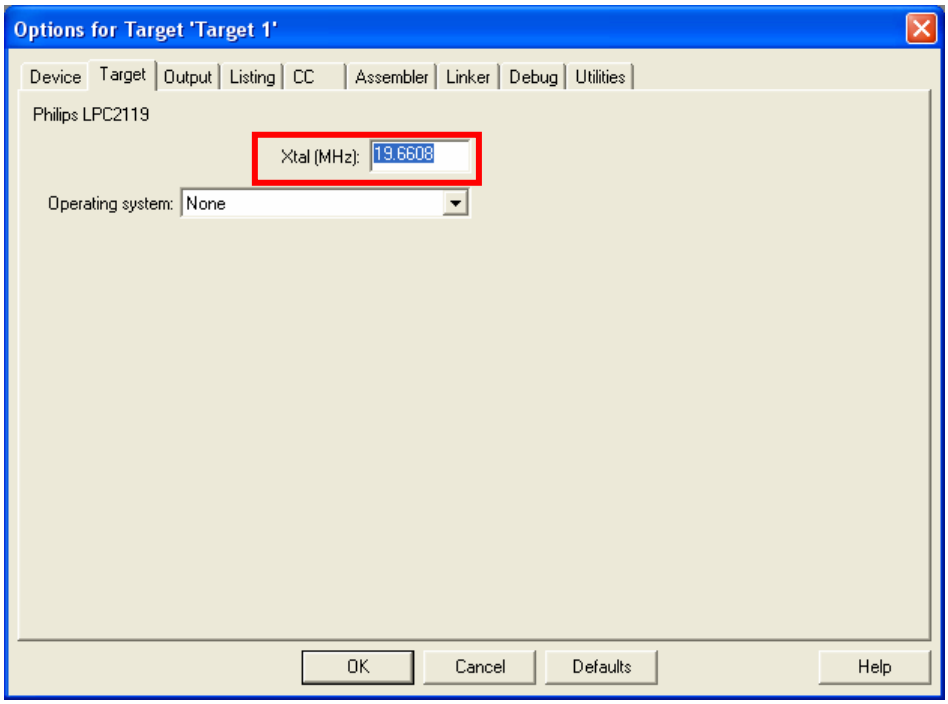
4. Copying File named "Flash.ld" and "Startup.S" that is in ETT's CD-ROM in folder named "SOURCE_GCC" into the same position of New Project File.

File named "Flash.ld" which is Script File uses to keep the beginning and the end of default value with Flash Memory size and RAM in Philips' MCU No. LPC2119 (it is MCU of ET-ARM STAMP LPC2119).Script File orders Keil uVision3 Program how to translate some condition of Project File's order to GCCARM. If there's any mistake of Script File, HEX File is translated wrongly.

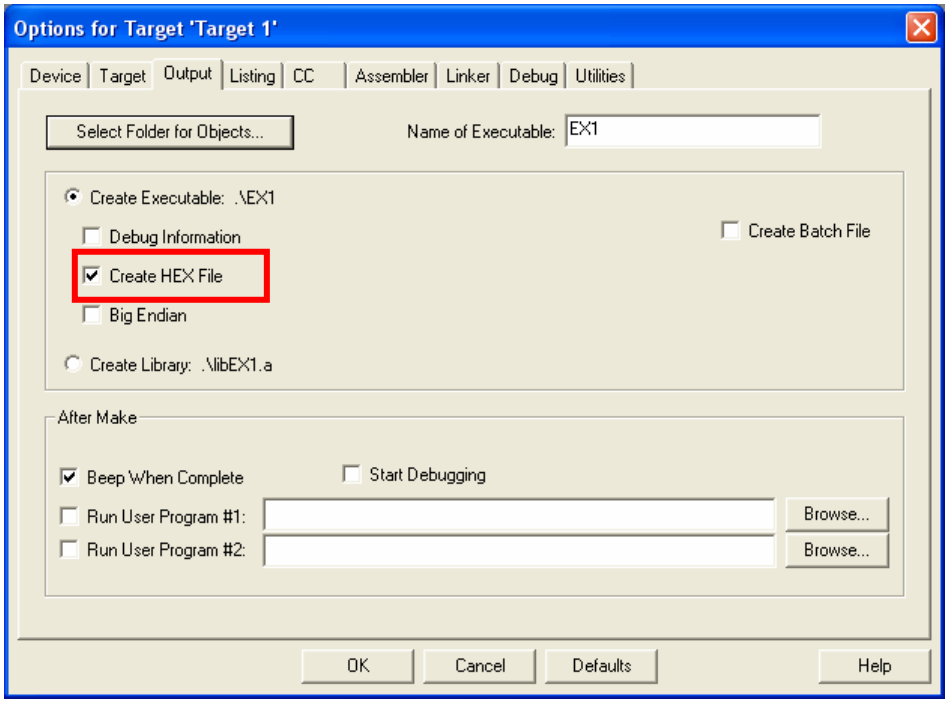
File named "Startup.S" which is contained orders of ARM7 in Assembly Language uses to set the necessary default value of MCU running such as setting Stack value to MCU, Initial Phase-Lock-Loop, MAM Function, and MCU's Vector.

5. Choosing Option of Project File by clicking **Project → Option for target 'target 1'** then choosing Tab of Target for setting MCU Target value. The proceeding is;

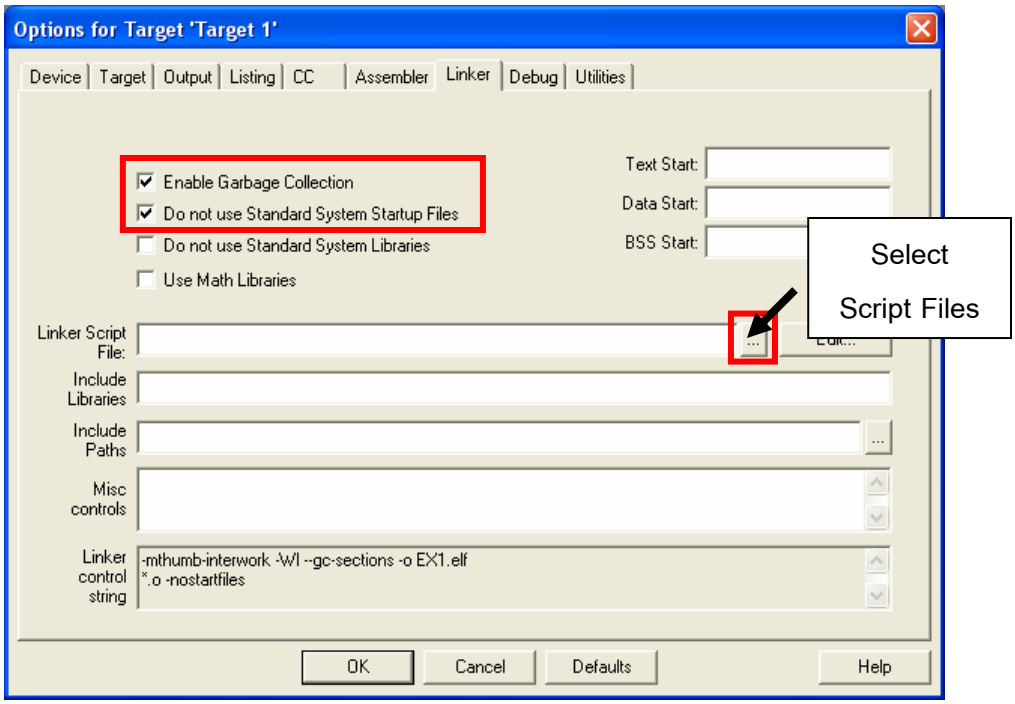
5.1 **X-TAL** is 19.6608 MHz

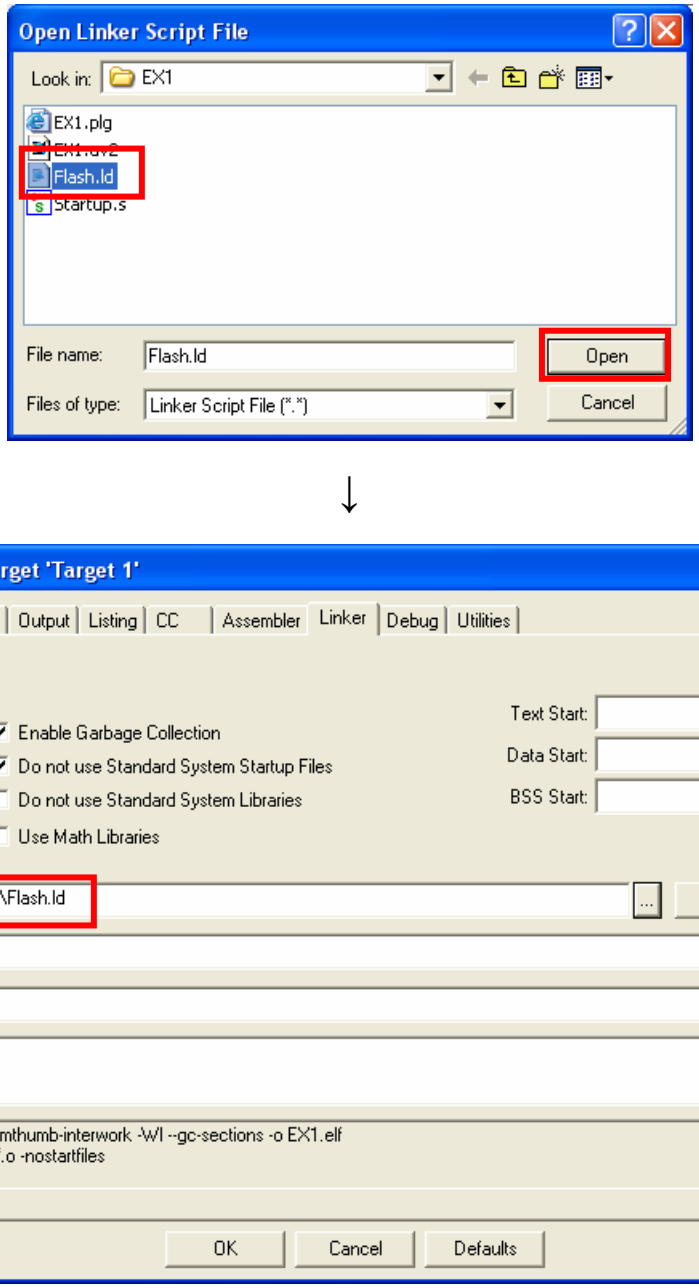


5.2 **Output,** click the default value of Create HEX File.



5.3 **Linker,** click **Enable Garbage collection** and **Do not use Standard System Startup File,** then set Linker Script File as "Flash.ld" by click the order button […] is between File name of Script File and "Edit…"button and then click Icon of "Flash.ld" and select open. It shows name of Script File in the blank immediately.

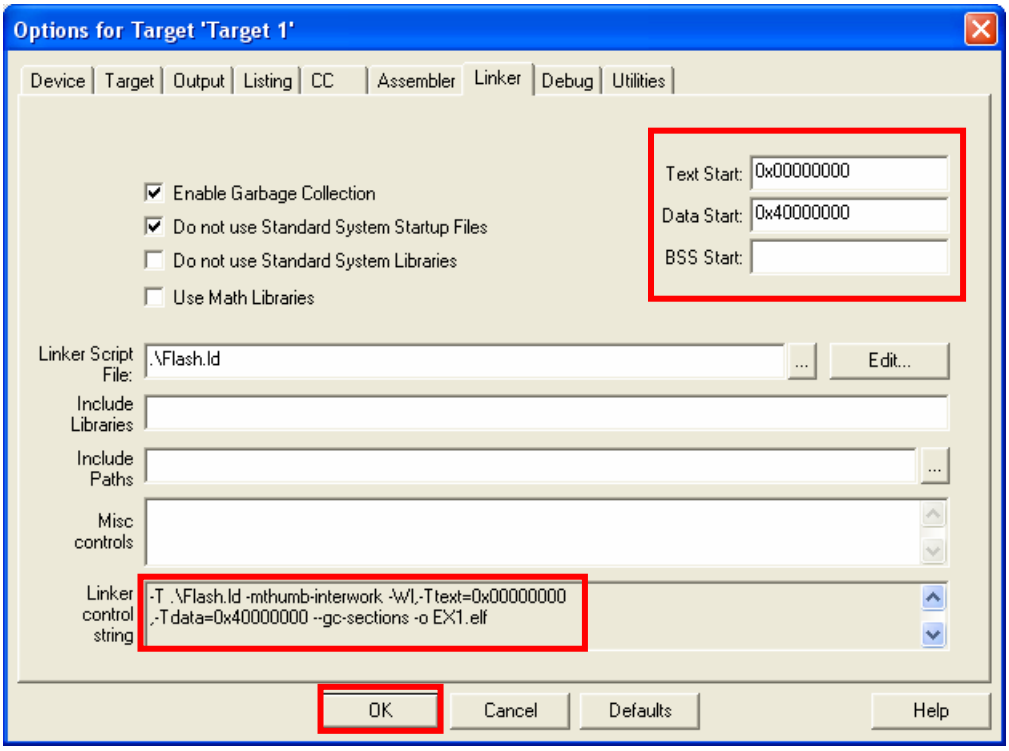




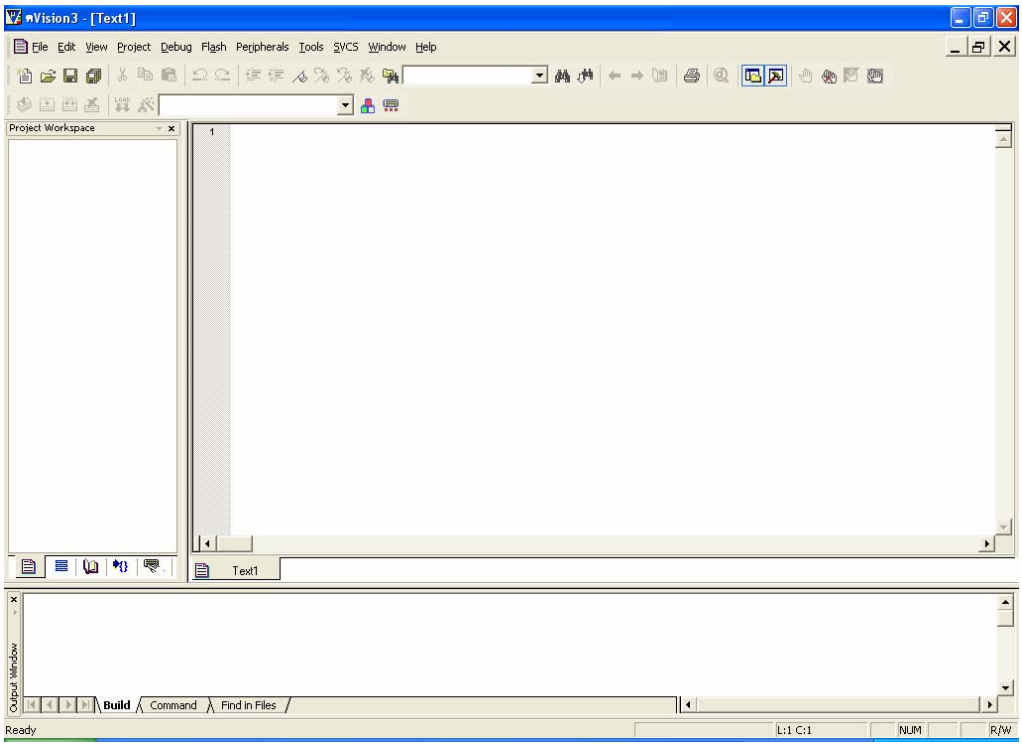
After finished to set Script File to Project, next proceeding is setting the beginning and the end value of address position in memory to save program and data. To set the value of LPC2119 is

- Text Start is 0x0000 0000
- Data Start is 0x4000 0000

Note: Keil uVision3 program displays its option that is translated through Linker Control String blank and its values change up to our conditions as in the picture.



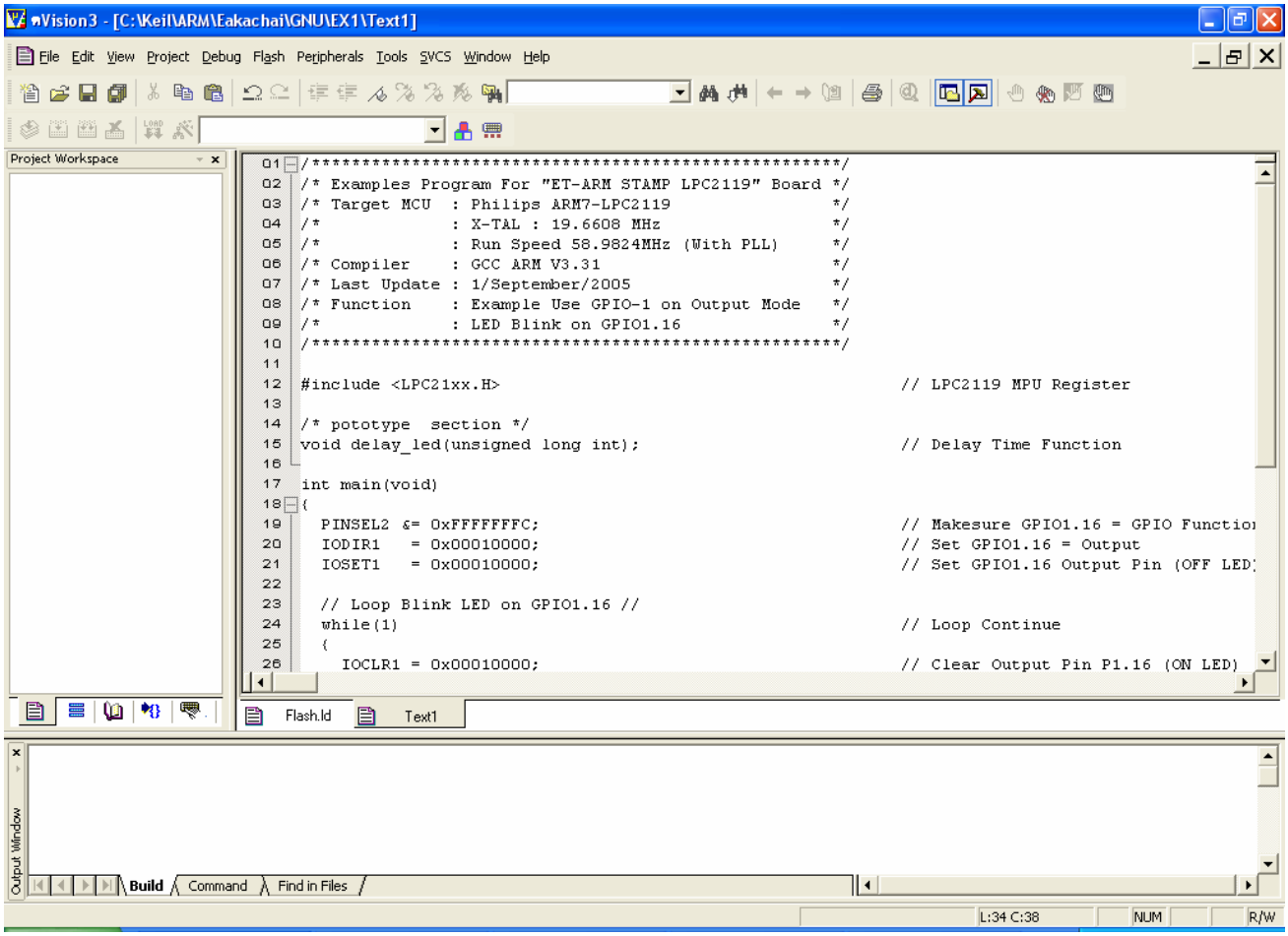
6. Start to write Source Code with C Language by click **File → New**… The Text File is displayed. In the first time, you should set its name as "Text1" as in the picture.



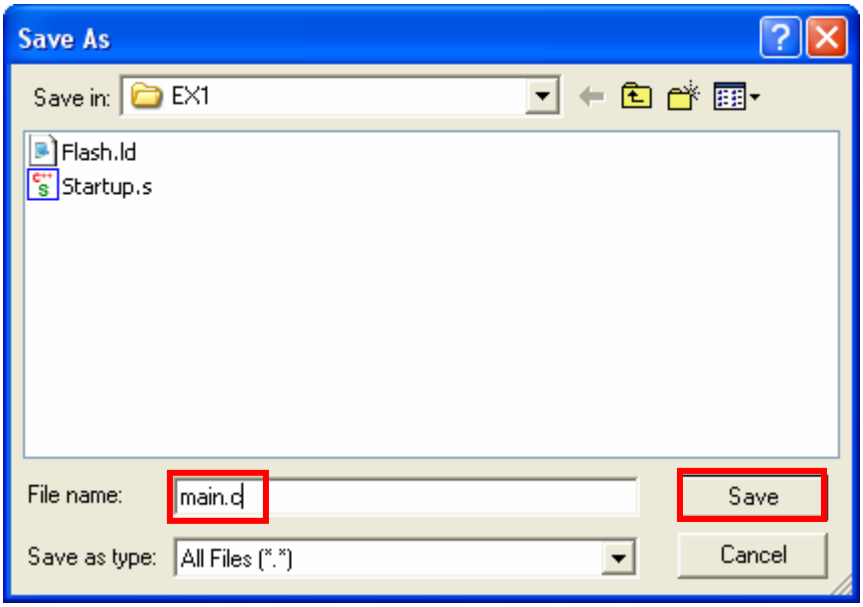
In this step, you should write Source Code with C Language in the GCCARM's form as in the example

```
/*****************************************************/
/* Examples Program For "ET-ARM STAMP LPC2119" Board */
/* Target MCU  : Philips ARM7-LPC2119                */
/*           : X-TAL : 19.6608 MHz                   */
/*             : Run Speed 58.9824MHz (With PLL)     */
/* Compiler    : GCC ARM V3.31                       */
/* Last Update : 1/September/2005                    */
/* Function    : Example Use GPIO-1on Output Mode    */
/*             : LED Blink on GPIO1.16               */
/*****************************************************/

#include <LPC21xx.H>               // LPC2119 MPU Register

/* pototype  section */
void delay_led(unsigned long int); // Delay Time Function

int main(void)
{
  PINSEL2 &= 0xFFFFFFFC;           // Set GPIO1.16 = GPIO Function
  IODIR1   = 0x00010000;           // Set GPIO1.16 = Output
  IOSET1   = 0x00010000;           // Set GPIO1.16 Pin (OFF LED)

  // Loop Blink LED on GPIO1.16 //
  while(1)                         // Loop Continue
  {
     IOCLR1 = 0x00010000;          // Clear Pin P1.16 (ON LED)
     delay_led(1500000);           // Display LED Delay
     IOSET1 = 0x00010000;          // Set Pin P1.16 (OFF LED)
     delay_led(1500000);           // Display LED Delay
  }
}

/***********************/
/* Delay Time Function */
/*    1-4294967296     */
/***********************/
void delay_led(unsigned long int count1)
{
  while(count1 > 0) {count1--;}    // Loop Decrease Counter
}
```

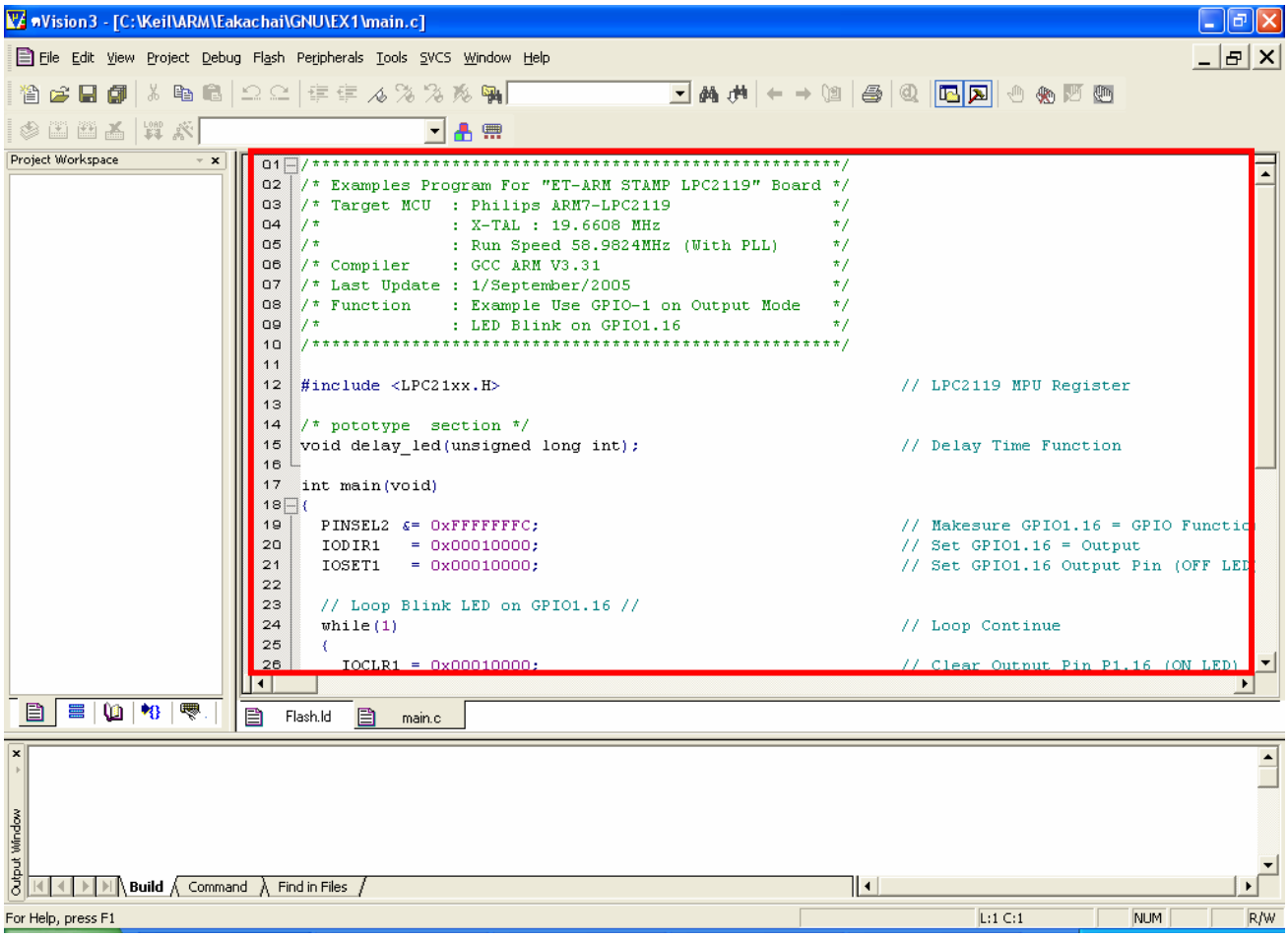
An example of a blinker Program at GPIO1.16



After finished to write program in C Language, you click save this file with its surname as ".C". Click **File → Save As…** and then set its file name as ".main.c" as in the picture.



After saved file, colour of letters in this program are changed with its function such as Comment, Variable, and Order. It is the good point of Keil uVision3 program and users use it easily.

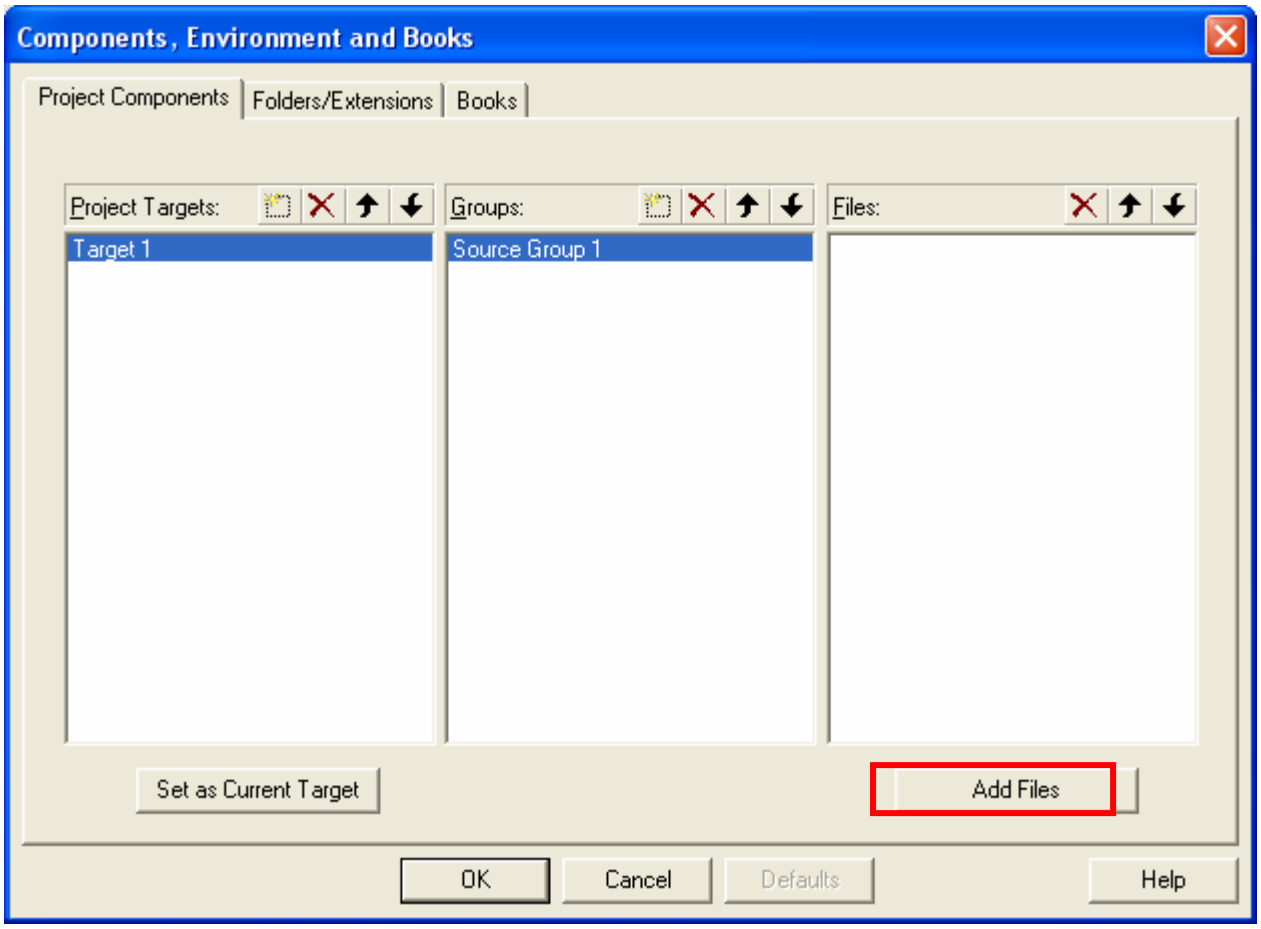


7. Add File with Project File by click **Project → Components, Environment, BOOks…** then choose **Tab Project Components** and then select Add File

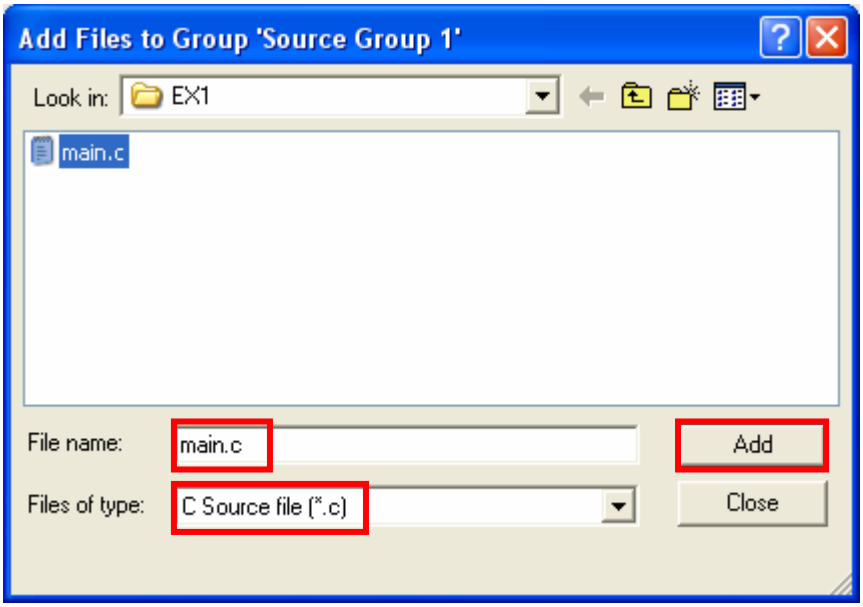
In the first time, you should select Files of type as "C Source files(*.c)" because it can show other files name in C Source Code. Click icon file named "main.c" then select **Add** file named "Startup.s" to Project File

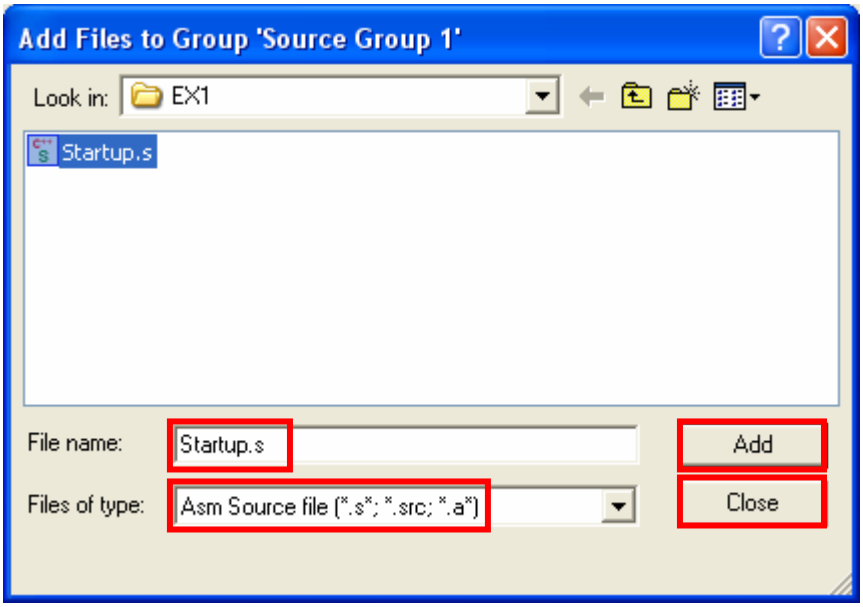
Then set new File of type as "ASM Source files(*.s*;*.src;*.a*). Its file named Startup.s is shown in file name blank, then click icon file "Startup.s", and then select **Add**

After Add file named "main.c" and "Startup.s" to Project File completely, select **Close** as in the picture

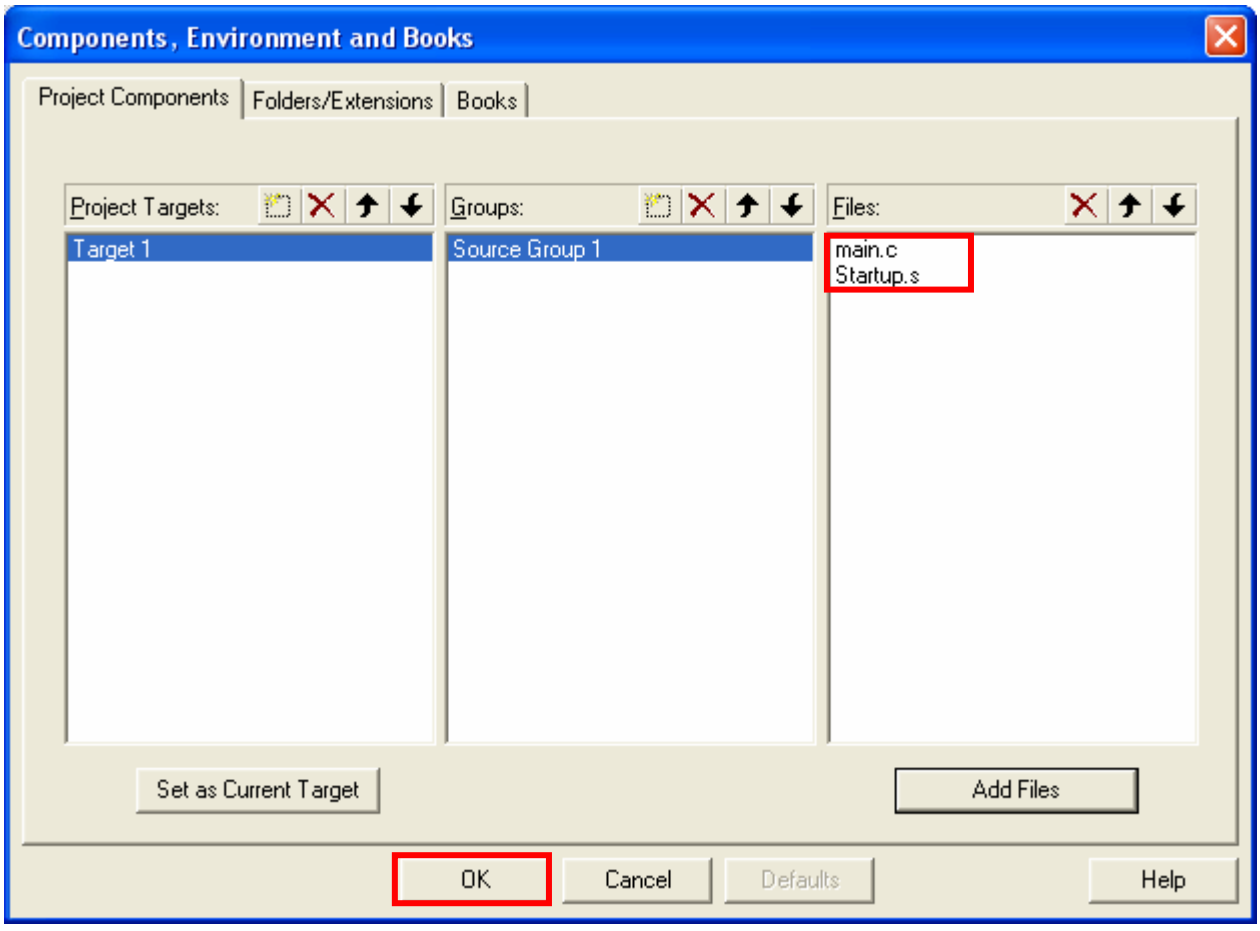


↓



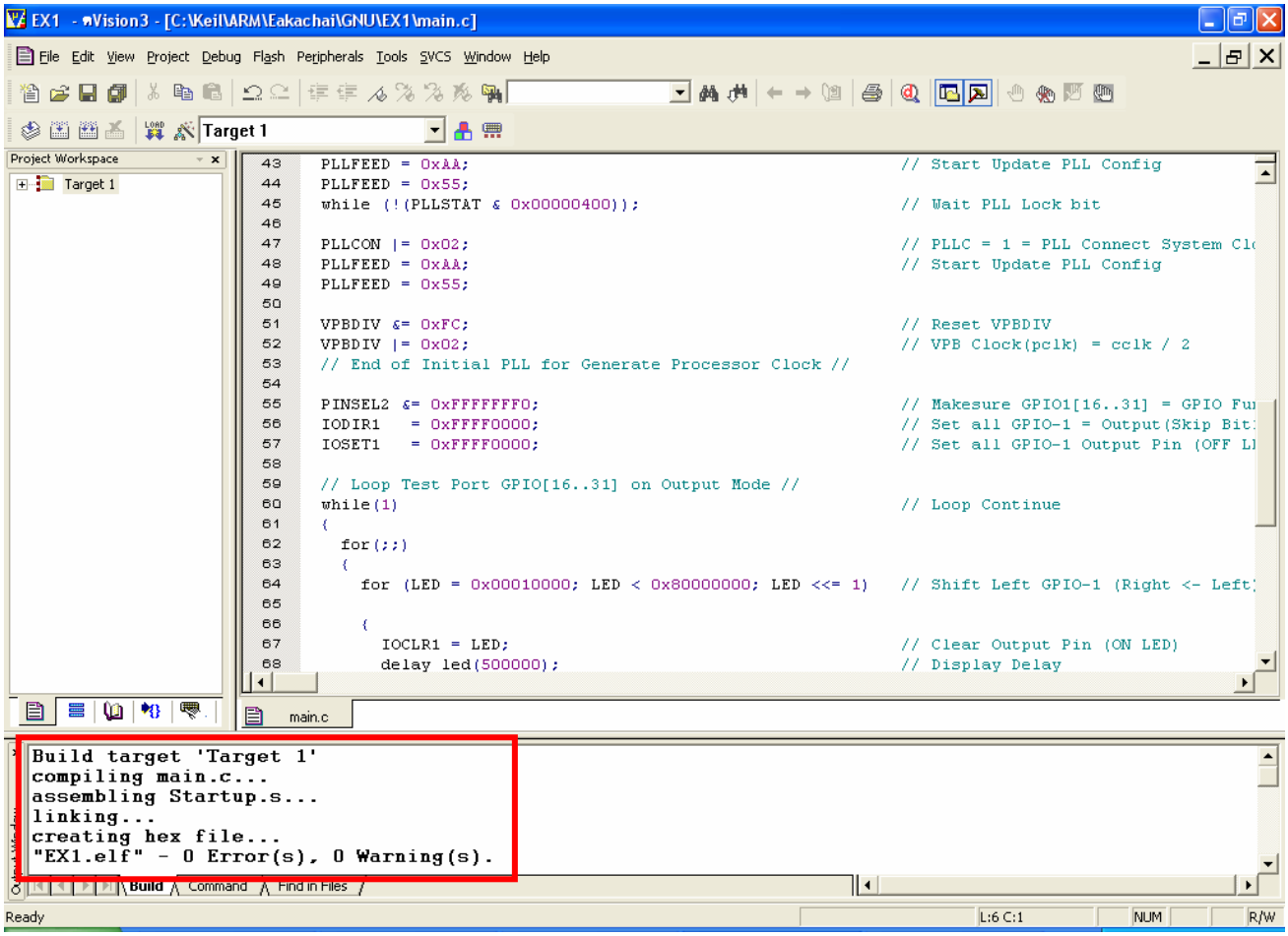


↓



After both of file named "main.c" and "Startup.s" are added to Project File , in the Tab of File displays theirs names

8. Translate written program by click **Projects → Rebuild all target files.** Keil uVision3 Program can order GCCARM Program to translate program immediately



After translated program correctly without any mistake (0 Error and 0 Warning), you will get the HEX File name as same as the Project File name. Users can download HEX File to MCU immediately

The Instructions of Initial MCU before Main Program Starting to Work

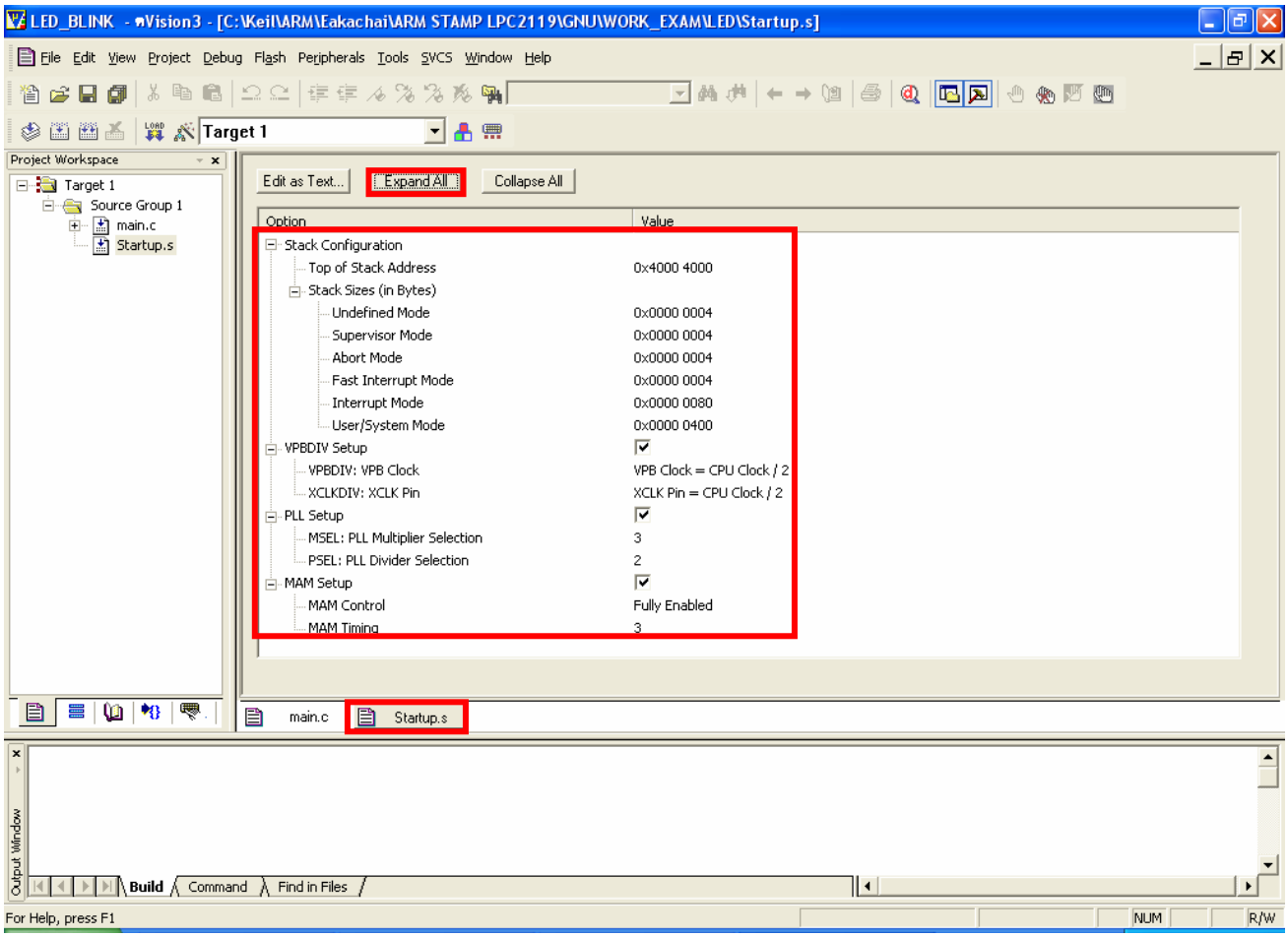
Your MCU will be the most efficient in high speed to collect data and running program, if you set default values as;

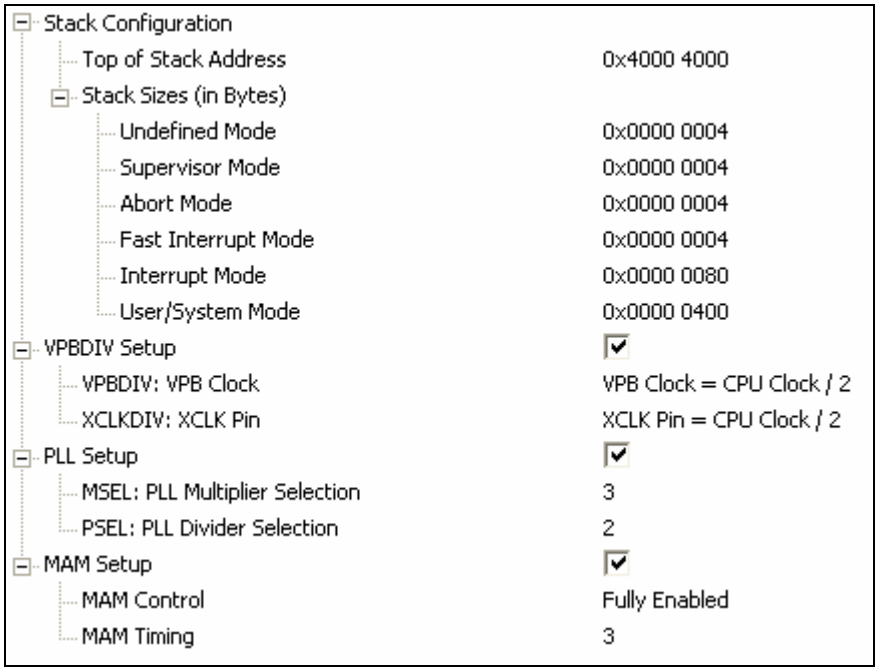
- Default value of PLL is Processor Clock (cclk)=58.9824MHz, in case of, using XTAL=19.6608MHz, default value is M(Multiply)=3 and P(Divide)=2 and FFCO=235.9296MHz
- Default value of VPB Clock(pclk) is a half of cclk (29.49MHz)
- Default value of MAM Timing is 3 cycle of cclk (MAMTIM=0x03)
- Default value of MAM Mode is Full Enable (MAMCR=0x02)

There's two methods to set default value as in the example. Firstly, writing Code in all written program by yourself and secondaly,copying Startup File and Script File as ready-made program then Add Startup File and Script File in your Project File. There's two ways to verify and correct the default values of Startup File and Script File. First, correct Code directly and second, set default value of Startup File and Script File from Keil uVision3 Program. In this case, correcting default value of them by Keil uVision3 Program is easier

Verification of Startup File

Function of Startup File is contained beginning orders of MCU before start to run with written program by yourself. Function of Program in Startup File is Initial MCU in the necessary part and then jump to run in written Program in C Language. The method to verify Startup is clicking Tab of Startup File then selecting "Expand All". The correct default value of Startup File is like in the picture.

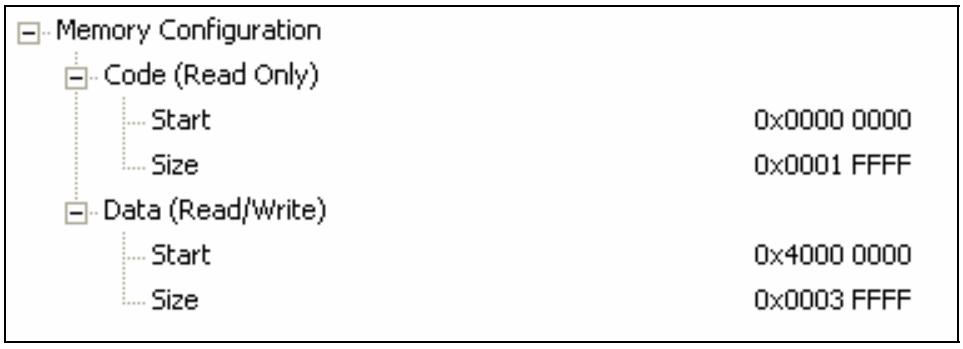




Picture shows setting the default value of Startup File in LPC2119

Verification of Script File

Function of Script File is kept default value of position and Flash Memory size in MCU or Target Board uses with written Program by yourself.The beginning position of 128K Code Flash Memory in LPC2119 is 00000000H-0001FFFFH and 16Kbyte Data RAM Memory is 40000000H-40003FFFH. Configuration value of Script File named "Flash.ld" that ETT wrote in GCCARM's C Language is correct for LPC2119. If you correct or change its default value, there's some mistake for running program. To verify default value of Configuration in Script File by opening file named "Flash.ld" in the same folder of Project File through Keil uVision3 Program or opening File in Tab File to "Flash.ld" and then selecting "Expand All". After these, you can verify them and the correct default values is like in the picture



Picture shows setting the default value of Script File in LPC2119

An example of GCCARM's Code C Language for Initial LPC2119

If you want to writ Initial Program for running MCU by yourself, you should add orders in the beginning of Main Program as in the example

```
// Initial PLL & VPB Clock For ET-ARM7 STAMP LPC2119
// Start of Initial PLL for Generate Processor Clock
// PLL Configuration Setup
// X-TAL = 19.6608MHz
// M(Multiply) = 3
// P(Divide) = 2
// Processor Clock(cclk) = M x OSC
//                       = 3 x 19.6608MHz
//                       = 58.9824MHz
// FCCO = cclk x 2 x P
//      = 58.9824 x 2 x P
//      = 235.9296 MHz
// VPB Clock(pclk) = 29.4912MHz
// Start of Initial PLL for Generate Processor Clock
PLLCFG &= 0xE0;                    // Reset MSEL0:4
PLLCFG |= 0x02;                    // MSEL(PLL Multiply) = 3
PLLCFG &= 0x9F;                    // Reset PSEL0:1
PLLCFG |= 0x20;                    // PSEL(PLL Devide) = 2

PLLCON &= 0xFC;                    // Reset PLLC,PLLE
PLLCON |= 0x01;                    // PLLE = 1 = Enable PLL

PLLFEED = 0xAA;                    // Start Update PLL Config
PLLFEED = 0x55;
while (!(PLLSTAT & 0x00000400));   // Wait PLL Lock bit

PLLCON |= 0x02;                    // PLLC = 1 (Connect PLL Clock)
PLLFEED = 0xAA;                    // Start Update PLL Config
PLLFEED = 0x55;

VPBDIV &= 0xFC;                    // Reset VPBDIV
VPBDIV |= 0x02;                    // VPB Clock(pclk) = cclk / 2
// End of Initial PLL for Generate Processor Clock

// Start of Initial MAM Function
MAMCR = 0x00;                      // Disable MAM Function
MAMTIM = 0x03;                     // MAM Timing = 3 Cycle of cclk
MAMCR = 0x02;                      // Enable MAM = Full Function
// End of Initial MAM Function

// Start of Main Function Here
.
```

An example of Code for Initial LPC2119 before Main Program start to work