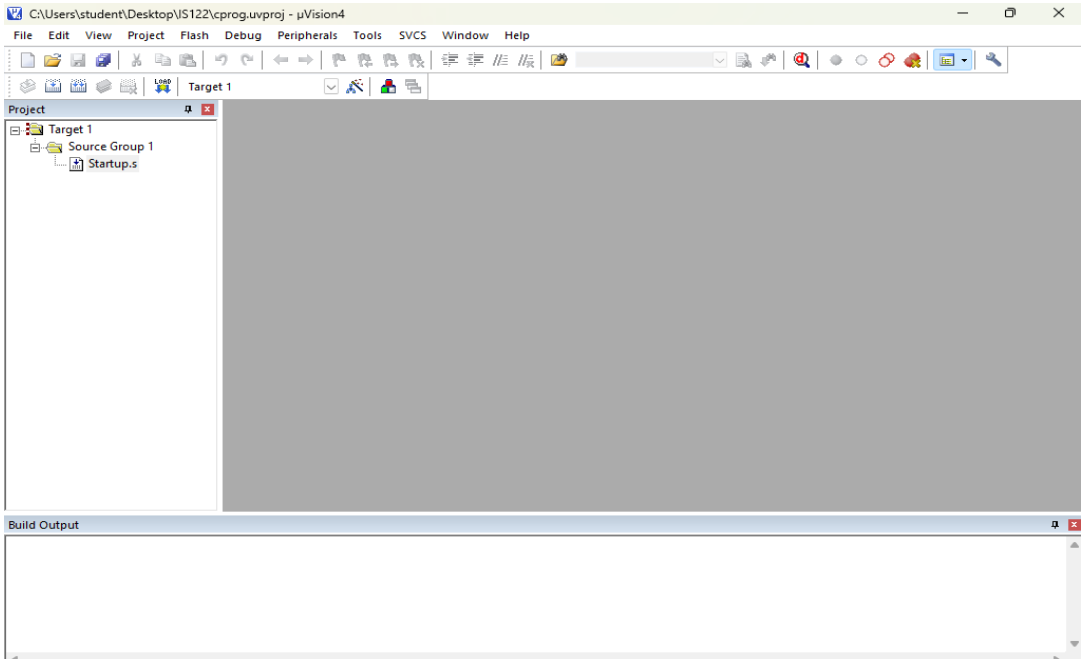
Part- B

1. C program to toggle the lowest pin of Port 0 with a delay between the two states. Observe and record the waveform obtained using the Logic Analyser in the Keil simulator.

Soln:

Step 1: Open the keil simulator

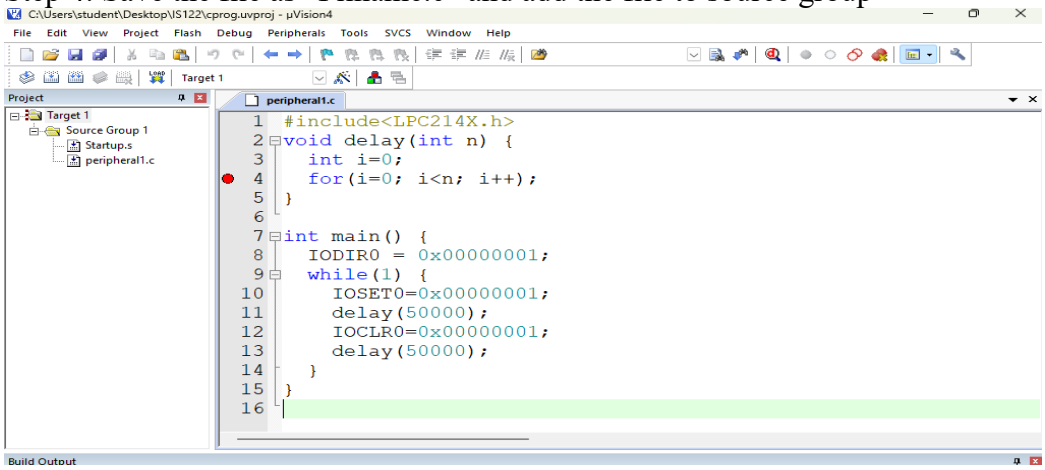
Step 2: Create the new project with LPC2148 and click " Yes" for the statup.s file, such that it will be linked to soruce group in project window.



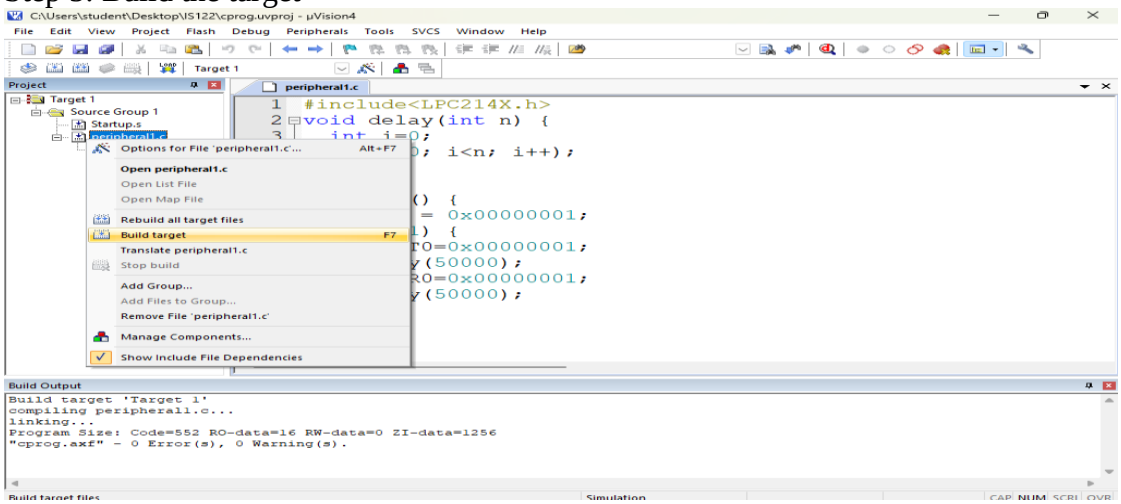
Step3: open the new file and type the program given below:

```
#include<LPC214X.h>
void delay(int n) {
	int i=0;
	for(i=0; i<n; i++);
}

int main() {
	IODIR0 = 0x00000001;
	while(1) {
		IOSET0=0x00000001;
		delay(50000);
		IOCLR0=0x00000001;
		delay(50000);
	}
}
```

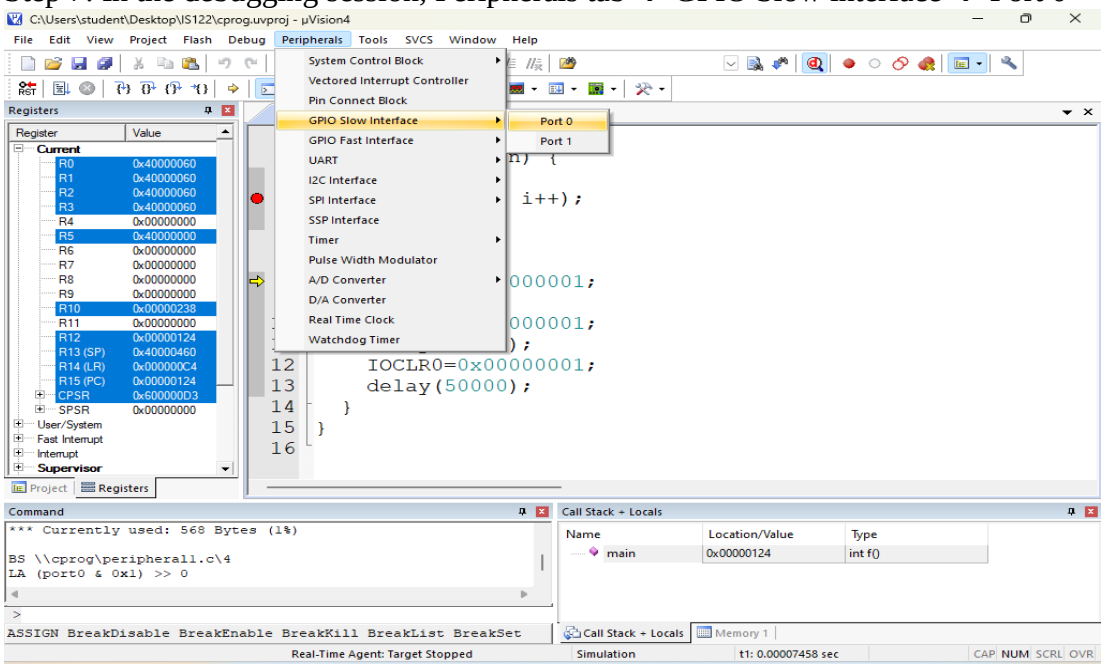
Step 4: Save the file as “Filname.c” and add the file to source group



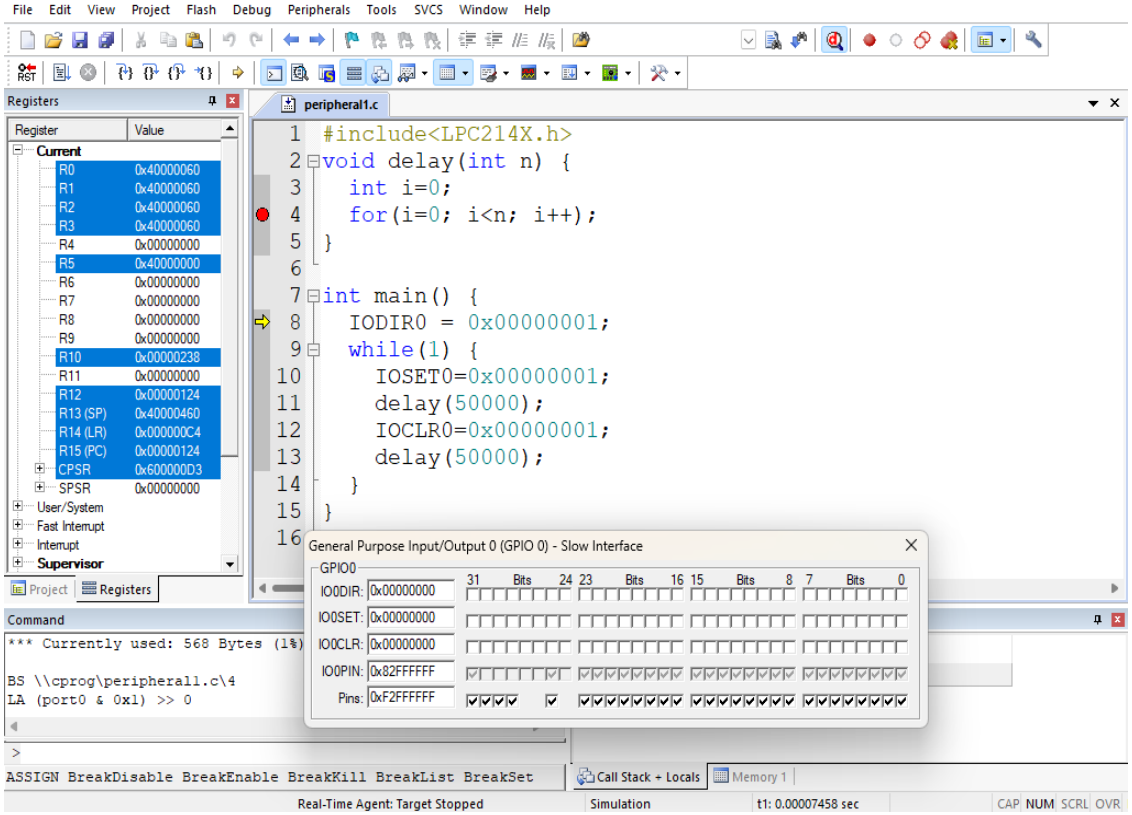
Step 5: Build the target



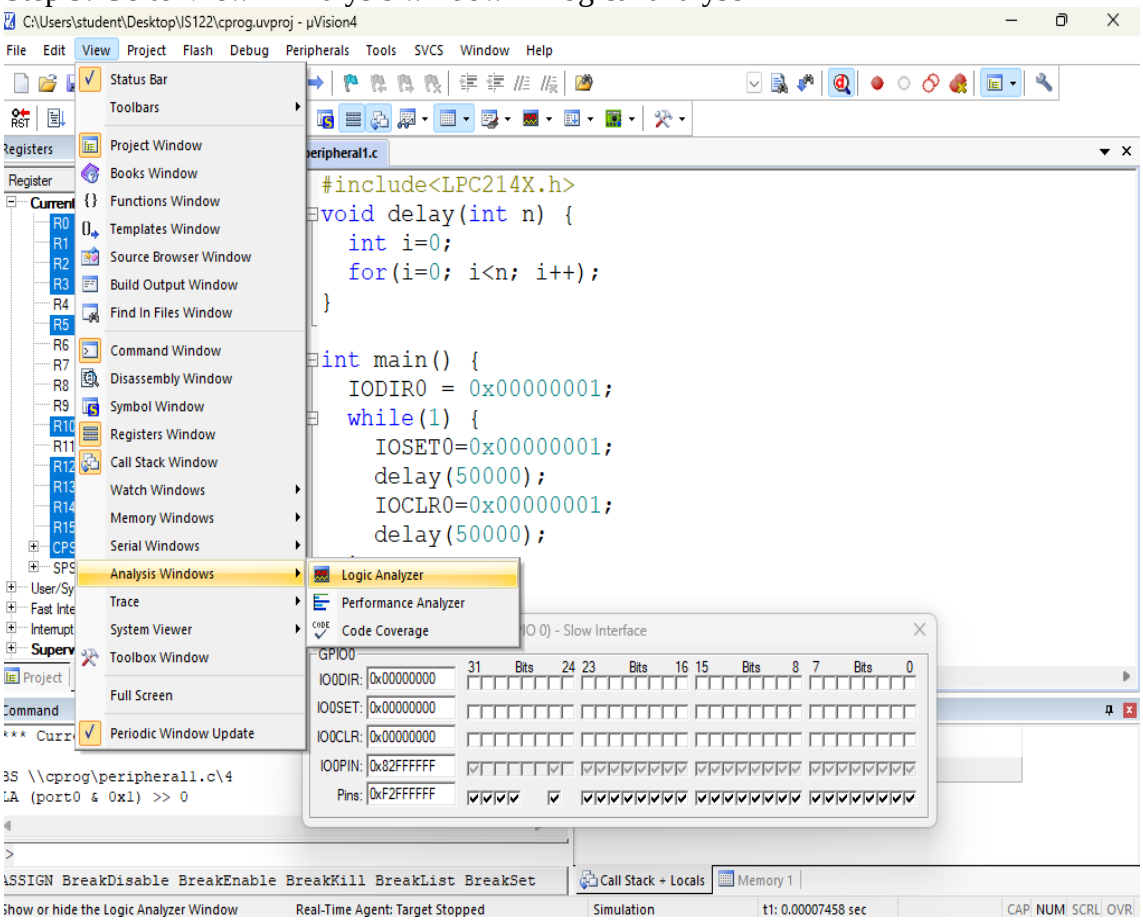
Step 6: After the build result with 0 errors, start the debugging session
Step 7: In the debugging session, Peripherals tab → GPIO Slow interface → Port 0



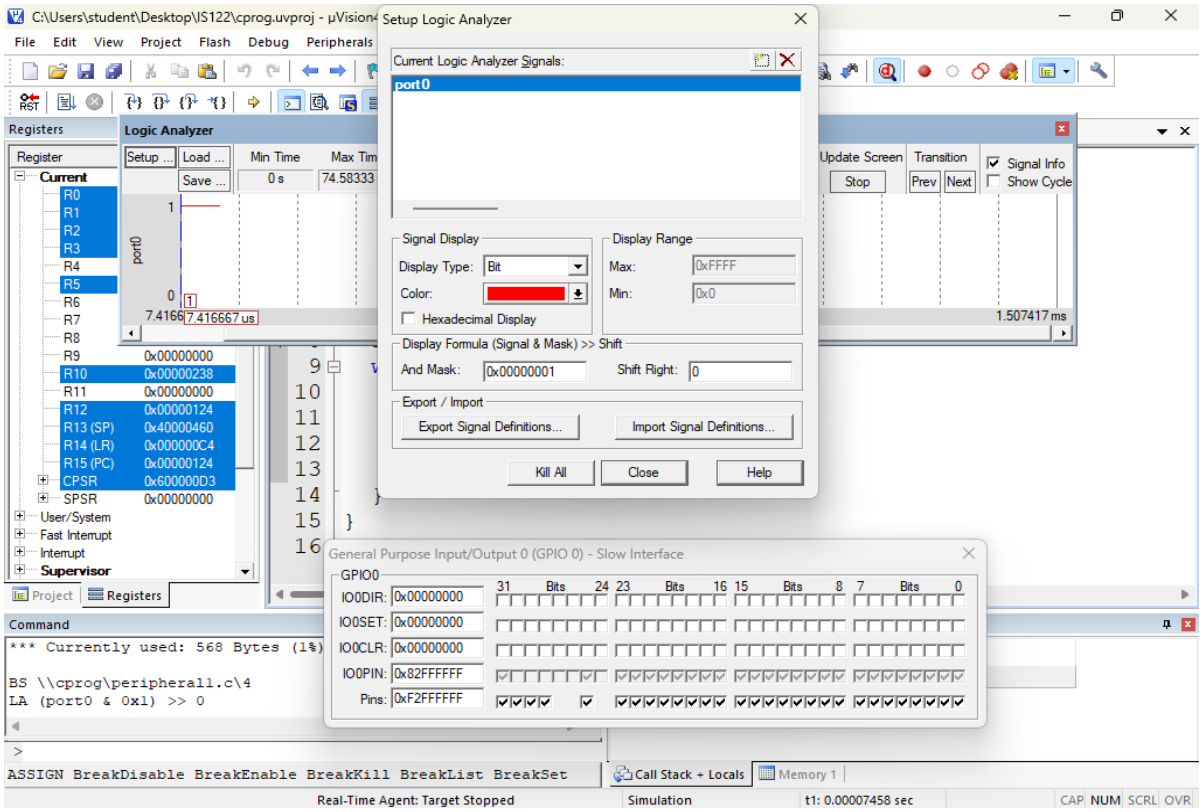
Step 8: On port0 slow interface box opens



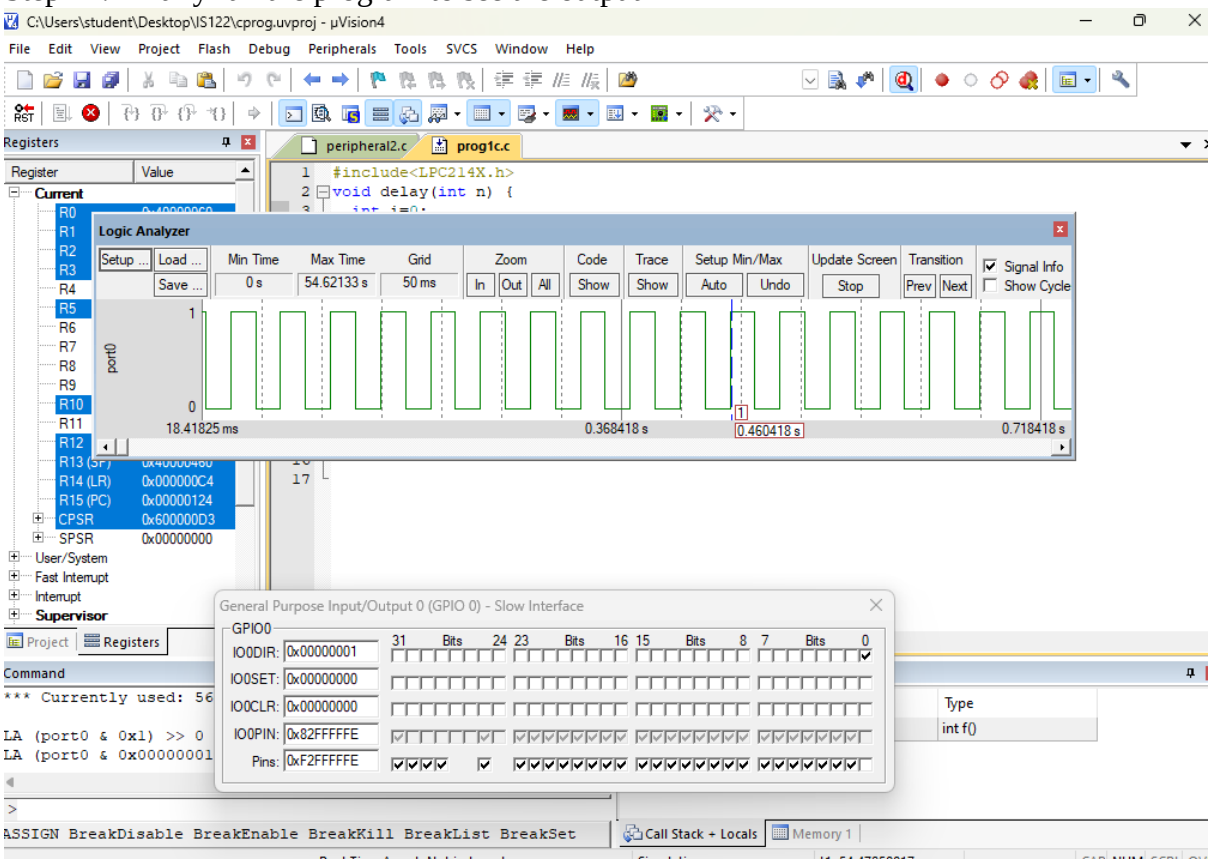
Step 9: Go to View→Analysis window → logical analyser



Step 10: In the logical analyser select “set” and box will open and set as “port 0.0” and display type as “Bit”.



Step 11: Finally run the program to see the output

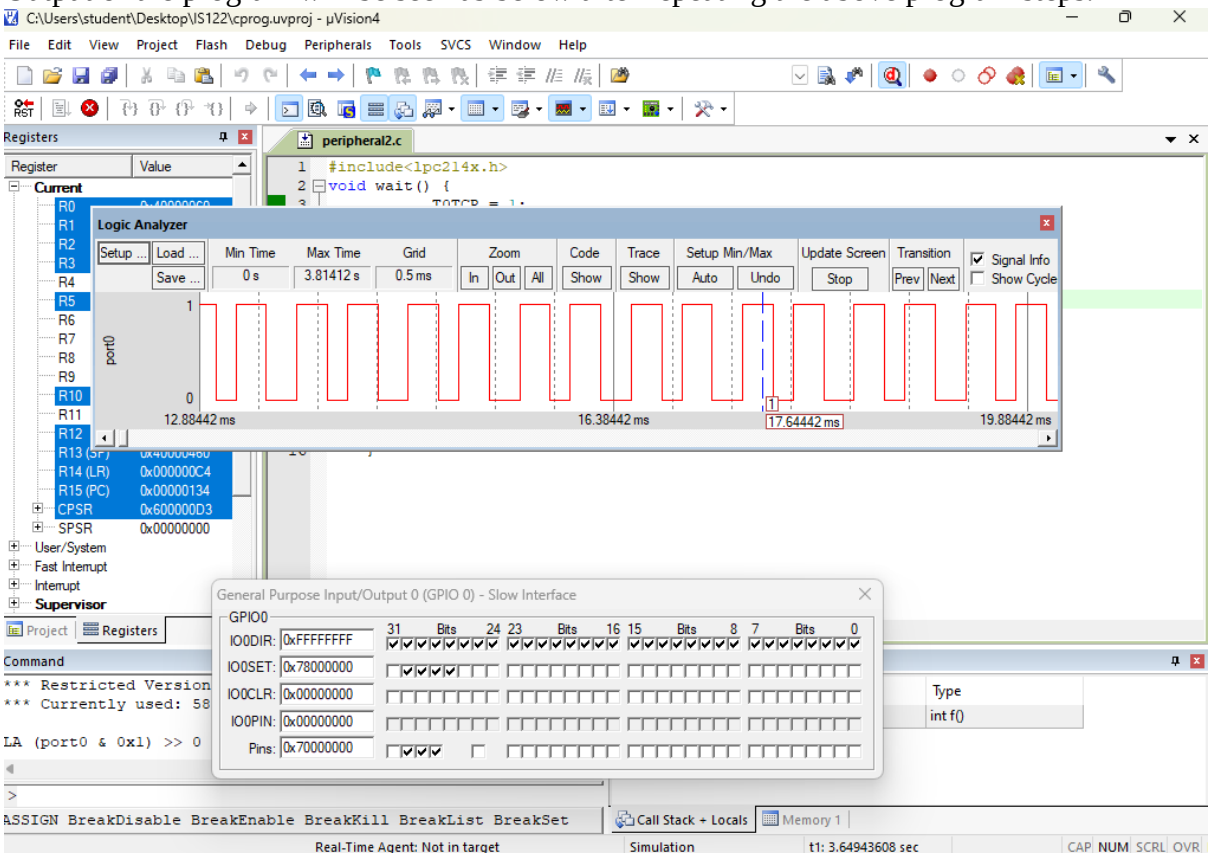


2. C program to generate an asymmetric square wave of 120Hz and having a duty cycle of 25% using the Timer0 module.

Soln: Repeat the same steps as above for the program given below

```
#include<lpc214x.h>
void wait() {
        T0TCR = 1;
                while(!(T0TC == T0MR0));
                }

                int main() {
                        T0MR0 = 0x00000100;
                         T0MCR = 0x2;
                            while(1) {
                                    IODIR0 = 0xFFFFFFFF;
                                    IOPIN0 = ~IOPIN0;
                                        wait();
                                    }
}
```

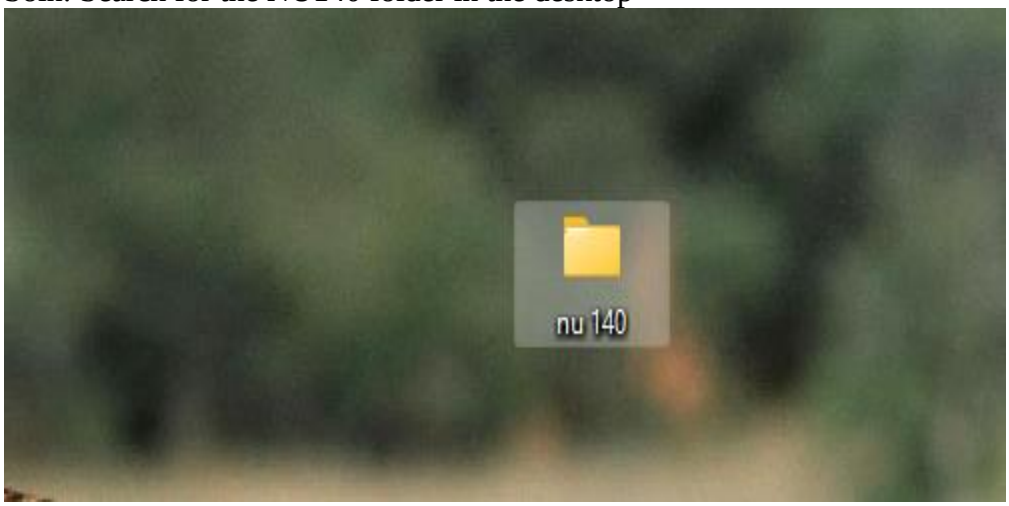
Output of the program will be seen as below after repeating the above program steps.



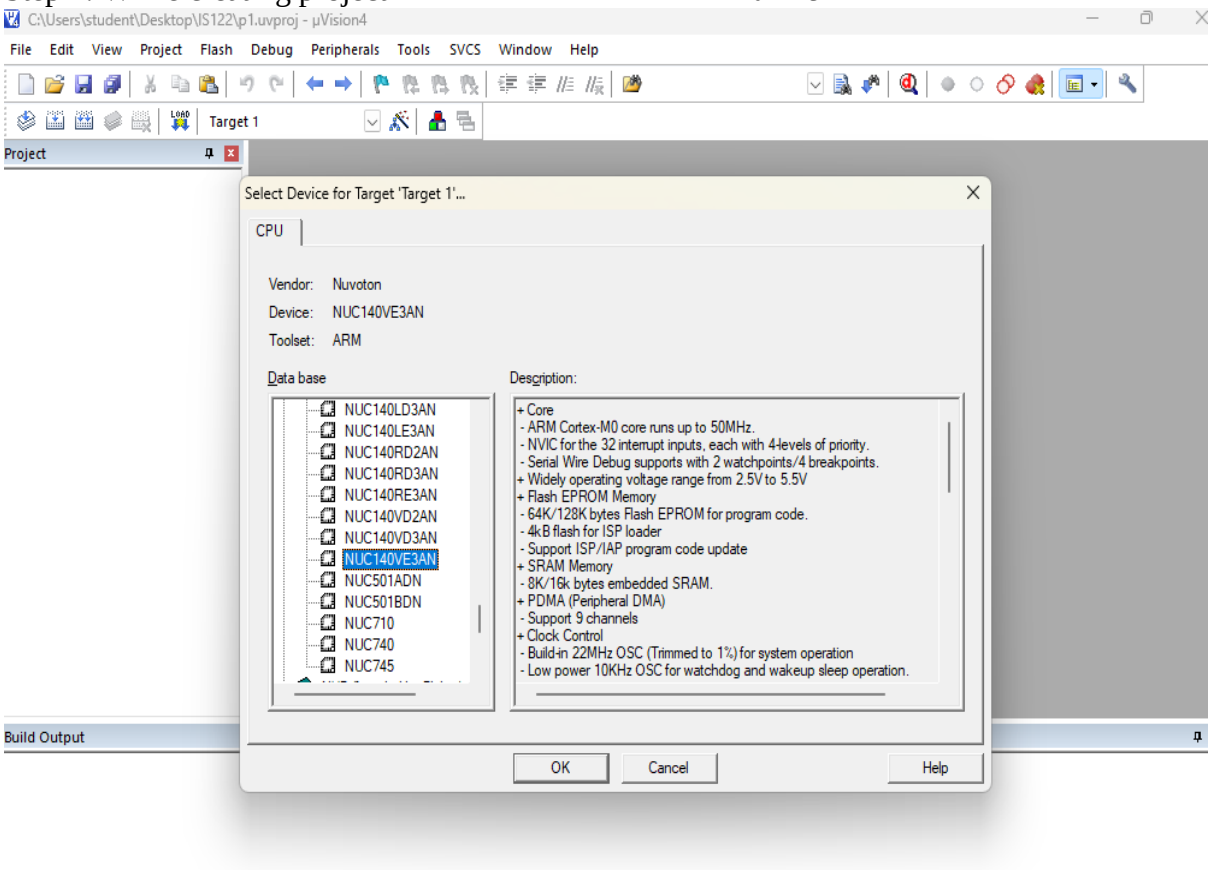
Hardware Programs

1. Write a C program to Interface NuMicro MCU Learning Board to Light a RGB LED connected to port A12-14.

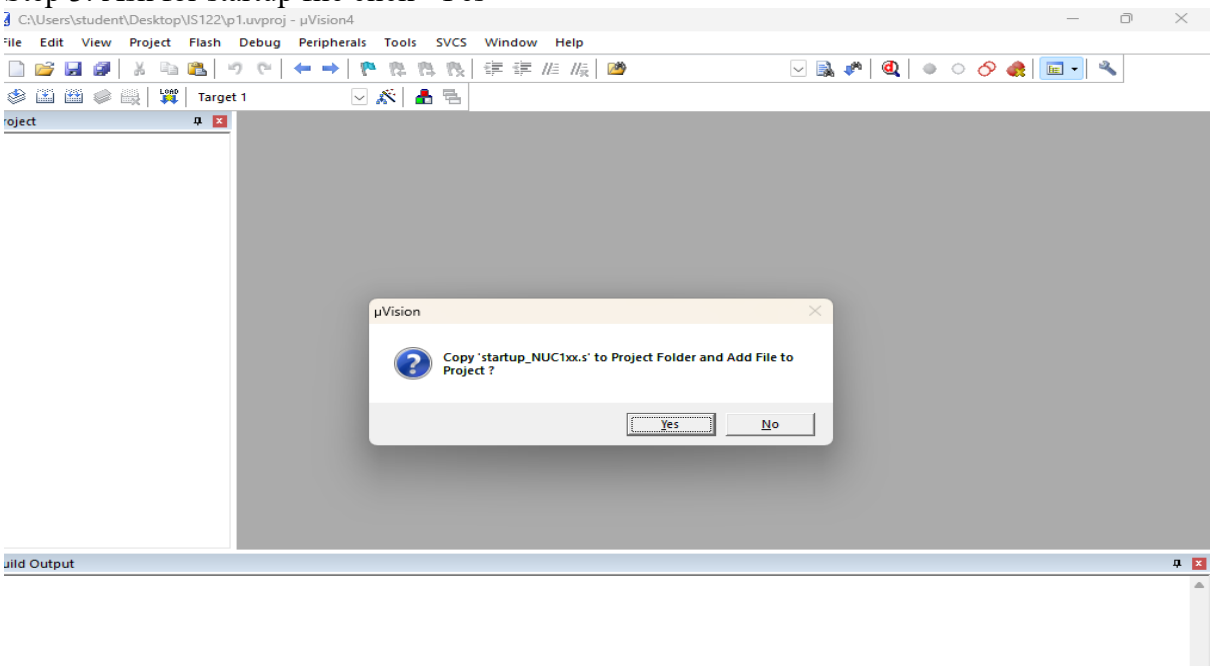
Soln: Search for the NU140 folder in the desktop



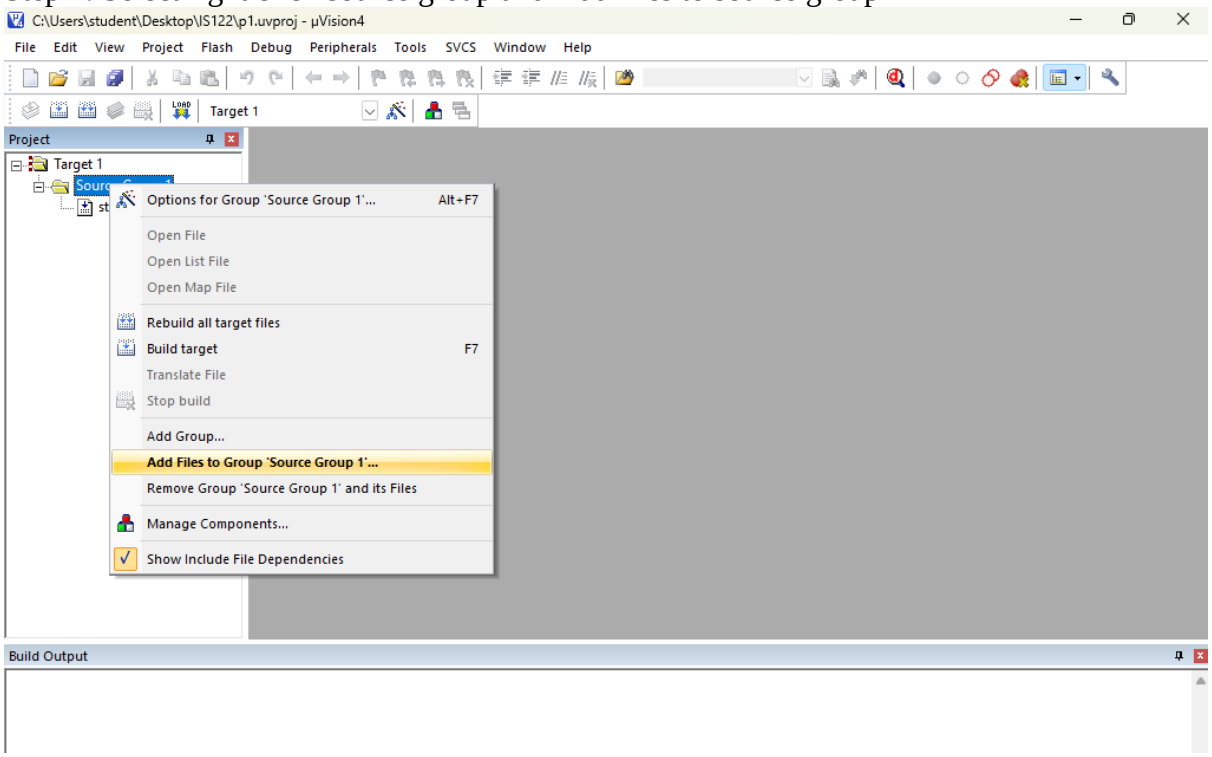
Step1: Open the keil software to create new project
Step 2: While creating project select "Nuvoton → NUC140VE3AN" and click OK



Step 3: Ask for startup file click “Yes”



Step 4: Select right click source group and Add files to source group



Step 5: Click NU140folder→CMSIS→coresupport→core_cm0 file and then Add

Step 6: Click U140folder→CMSIS→devicesupport→Nuvoton→NUC1xx→system_NUC1.x file click and then Add

Step 7: Click NU140folder→NuvotonPlatform_keil→Src→Driver→DrvGPIO file click and Add.

Step 8: Click NU140folder→NuvotonPlatform_keil→Src→Driver→DrvSYS file click and Add.

Step 9: Click NU140folder→NuvotonPlatform_keil→Src→Driver→DrvUART file click and Add.

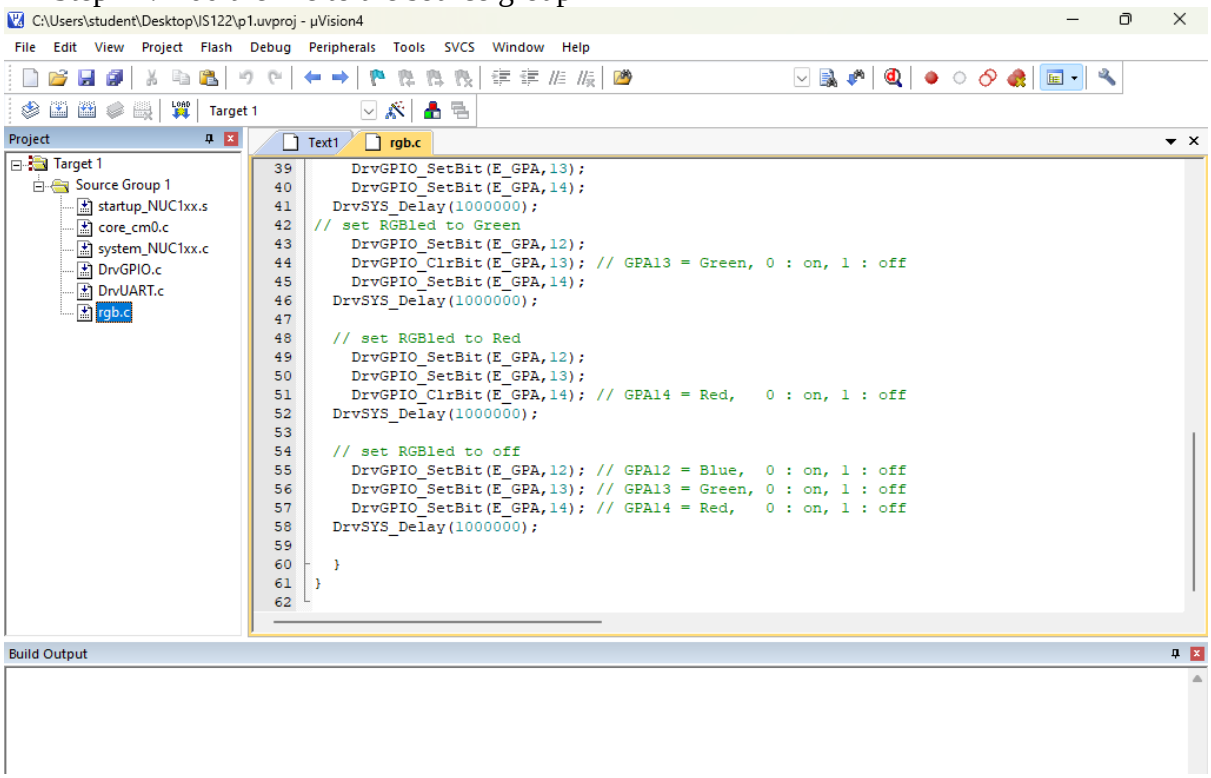
Step10: Open new file and type the below program

```
// Smpl_GPIO_RGBled : GPA12,13,14 output control RGB LED
//  output low to enable LEDs
#include <stdio.h>
#include "NUC1xx.h"
#include "Driver\DrvGPIO.h"
#include "Driver\DrvUART.h"
#include "Driver\DrvSYS.h"

// Initial GPIO pins (GPA 12,13,14) to Output mode
void Init_LED()
{
  // initialize GPIO pins
  DrvGPIO_Open(E_GPA, 12, E_IO_OUTPUT); // GPA12 pin set to output mode
  DrvGPIO_Open(E_GPA, 13, E_IO_OUTPUT); // GPA13 pin set to output mode
  DrvGPIO_Open(E_GPA, 14, E_IO_OUTPUT); // GPA14 pin set to output mode
  // set GPIO pins output Hi to disable LEDs
  DrvGPIO_SetBit(E_GPA, 12); // GPA12 pin output Hi to turn off Blue  LED
  DrvGPIO_SetBit(E_GPA, 13); // GPA13 pin output Hi to turn off Green LED
  DrvGPIO_SetBit(E_GPA, 14); // GPA14 pin output Hi to turn off Red   LED
}

int main (void)
{
  UNLOCKREG();                          // unlock register for programming
   DrvSYS_Open(48000000);     // set System Clock to run at 48MHz (PLL with
12MHz crystal input)
  LOCKREG();                            // lock register from programming

  Init_LED();

  while (1)
  {
  // GPA12 = Blue,  0 : on, 1 : off
  // GPA13 = Green, 0 : on, 1 : off
  // GPA14 = Red,   0 : on, 1 : off

  // set RGBled to Blue
   DrvGPIO_ClrBit(E_GPA,12); // GPA12 = Blue,  0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,13);
   DrvGPIO_SetBit(E_GPA,14);
  DrvSYS_Delay(1000000);
```

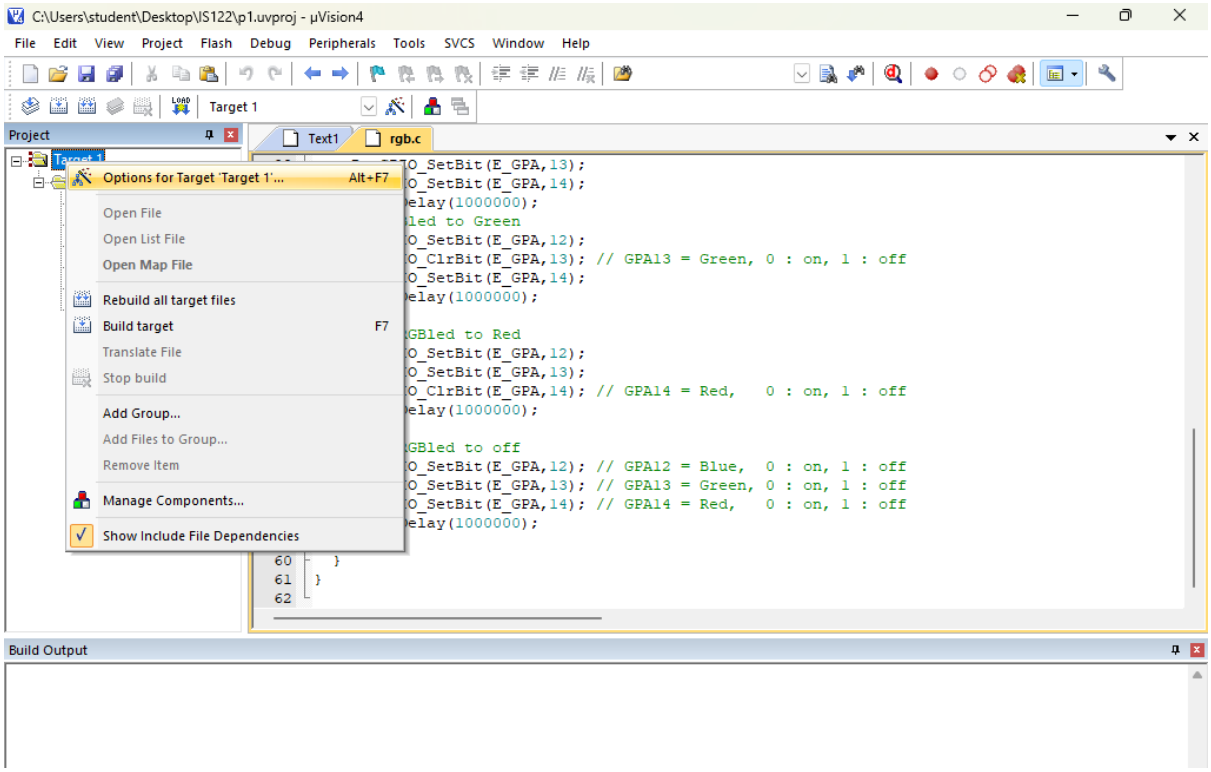
```
// set RGBled to Green
   DrvGPIO_SetBit(E_GPA,12);
   DrvGPIO_ClrBit(E_GPA,13); // GPA13 = Green, 0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,14);
  DrvSYS_Delay(1000000);

  // set RGBled to Red
   DrvGPIO_SetBit(E_GPA,12);
   DrvGPIO_SetBit(E_GPA,13);
   DrvGPIO_ClrBit(E_GPA,14); // GPA14 = Red,   0 : on, 1 : off
  DrvSYS_Delay(1000000);

  // set RGBled to off
   DrvGPIO_SetBit(E_GPA,12); // GPA12 = Blue,  0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,13); // GPA13 = Green, 0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,14); // GPA14 = Red,   0 : on, 1 : off
  DrvSYS_Delay(1000000);

  }
}
```

Step11: Save the file as “Filename.c”

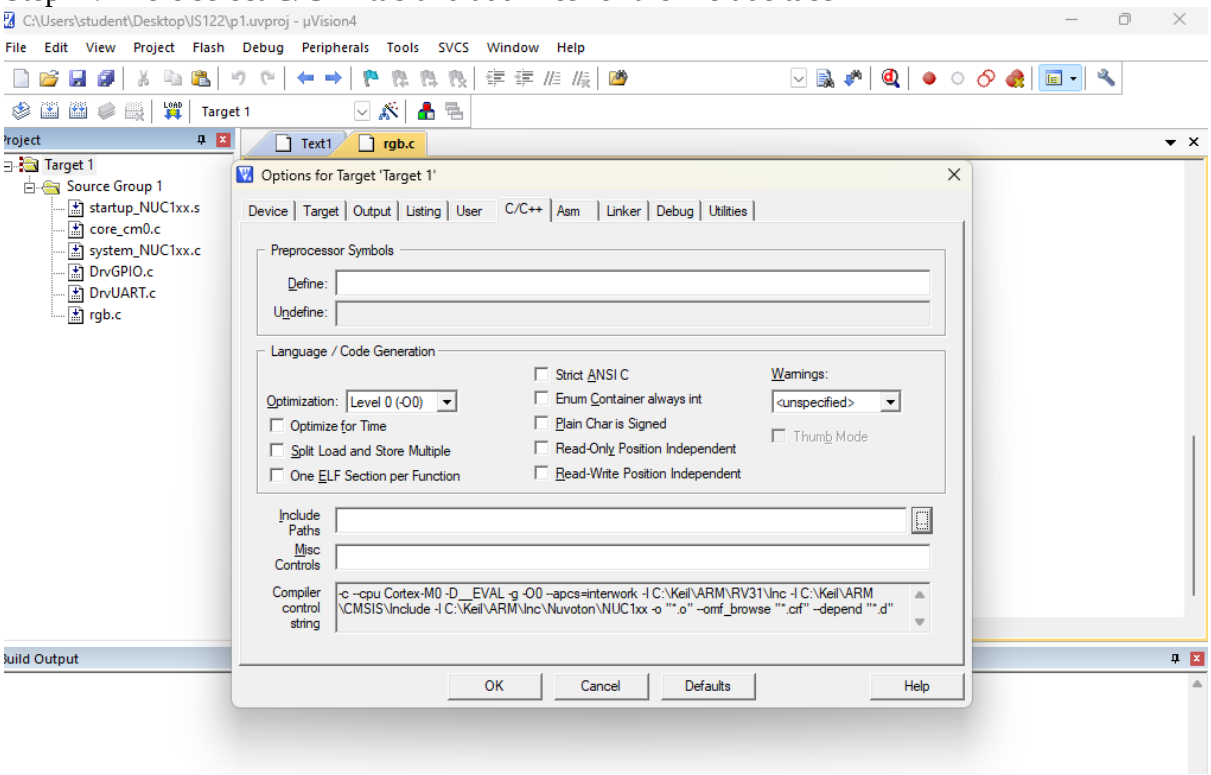
Step 12: Add the file to the source group



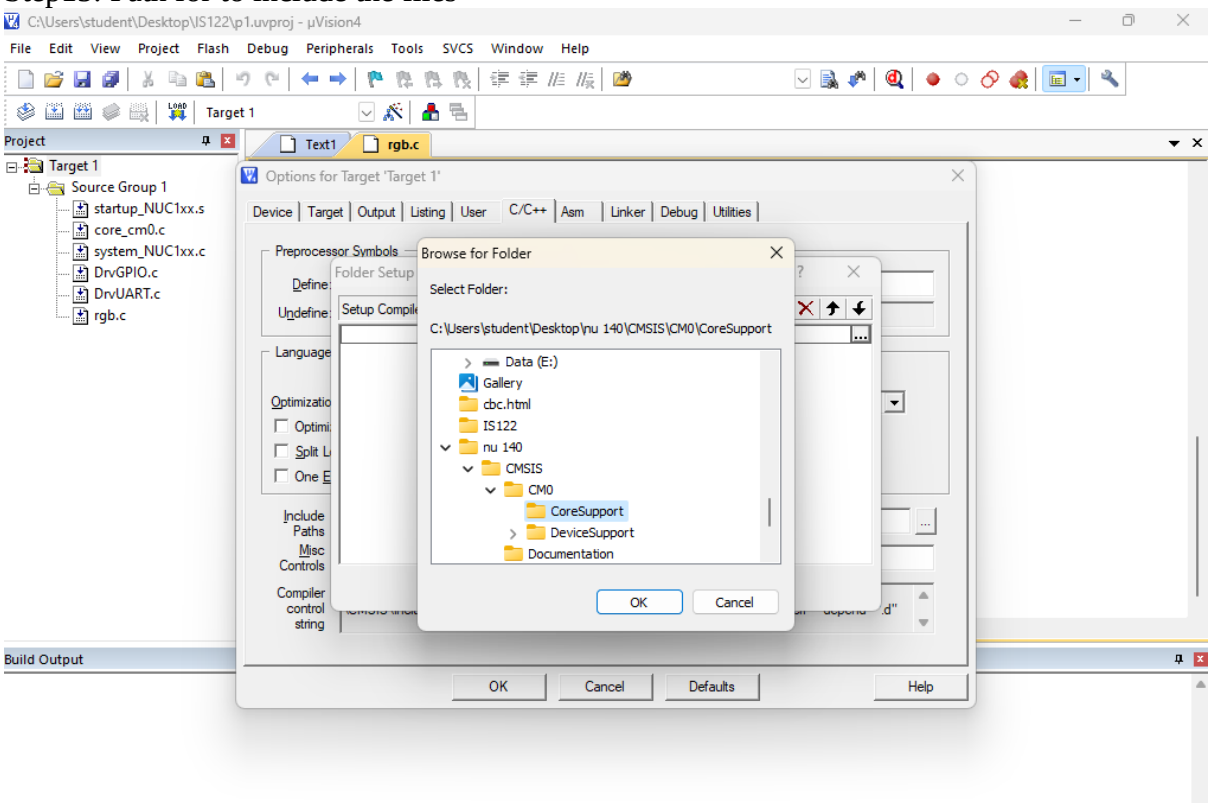
Step 13: select and right click the target in project window and select Options



Step14: There select C/C++ tab and add files for the Include tabs



Step15: Path for to include the files



Step 16: Select Debug tab and chage to Use→ NU Link Debugger

Step 17: Select Utilities and use target as NULink Debugger

Step 18: Connect the hardware

Step 19: Build the target and Run

2. Write a C program to Interface NuMicro MCU Learning Board to beep a buzzer connected to port B11.
   Soln : Same as above procedure

   Program is:

```
//
// Smpl_GPIO_Buzzer : GPB11 low-active output control Buzzer
// Note: Nu-LB-NUC140 R1 should be 0 ohm
//
#include &lt;stdio.h&gt;
#include &quot;NUC1xx.h&quot;
#include &quot;Driver\DrvSYS.h&quot;
#include &quot;Driver\DrvGPIO.h&quot;
#include &quot;Driver\DrvADC.h&quot;
int main (void)
{
UNLOCKREG(); // unlock register for programming
DrvSYS_Open(48000000); // set System Clock to run at 48MHz
LOCKREG(); // lock register from programming
DrvGPIO_Open(E_GPB, 11, E_IO_OUTPUT); // initial GPIO pin GPB11 for
controlling Buzzer
while(1) {
DrvGPIO_ClrBit(E_GPB,11); // GPB11 = 0 to turn on Buzzer
DrvSYS_Delay(100000); // Delay
DrvGPIO_SetBit(E_GPB,11); // GPB11 = 1 to turn off Buzzer
DrvSYS_Delay(100000); // Delay
}
}
```

3. Write a C program to Interface NuMicro MCU Learning Board to a even segment to display the values from 0 to 9999.
   Soln:

```
//
// Smpl_7seg : counting from 0 to 9999 and display on 7-segment LEDs
//
#include <stdio.h>
#include "NUC1xx.h"
#include "Driver\DrvSYS.h"
#include "Seven_Segment.h"
#include "DrvGPIO.h"
#include "DrvSYS.h"

// display an integer on four 7-segment LEDs
void seg_display(int16_t value)
{
  int8_t digit;
        digit = value / 1000;
        close_seven_segment();
        show_seven_segment(3,digit);
        DrvSYS_Delay(5000);

        value = value - digit * 1000;
        digit = value / 100;
        close_seven_segment();
        show_seven_segment(2,digit);
        DrvSYS_Delay(5000);

        value = value - digit * 100;
        digit = value / 10;
        close_seven_segment();
        show_seven_segment(1,digit);
        DrvSYS_Delay(5000);

        value = value - digit * 10;
        digit = value;
        close_seven_segment();
        show_seven_segment(0,digit);
        DrvSYS_Delay(5000);
}

int32_t main (void)
{
        char TEXT1[16];
                                int val;
                                val=0000;
        UNLOCKREG();
        DrvSYS_Open(48000000);
        LOCKREG();

        while(1)
```

```
        {           DrvSYS_Delay(500);
         val=val++ ;

        seg_display(val);                               // write 1 to clear the flag

        }
}
```