การทดลองที่ 7 การใช้งาน Real Time Clock

<u>วัตถุประสงค์</u>

1. เพื่อให้นิสิตสามารถเขียนโปรแกรมภาษา C อย่างง่ายในการควบคุมไมโครคอนโทรลเลอร์ได้

3. เพื่อให้นิสิตสามารถใช้งาน Real Time Clock เบอร์ DS1307 ได้

<u>อุปกรณ์ในการทดลอง</u>

1. ชุดทดลองไมโครคอนโทรลเลอร์

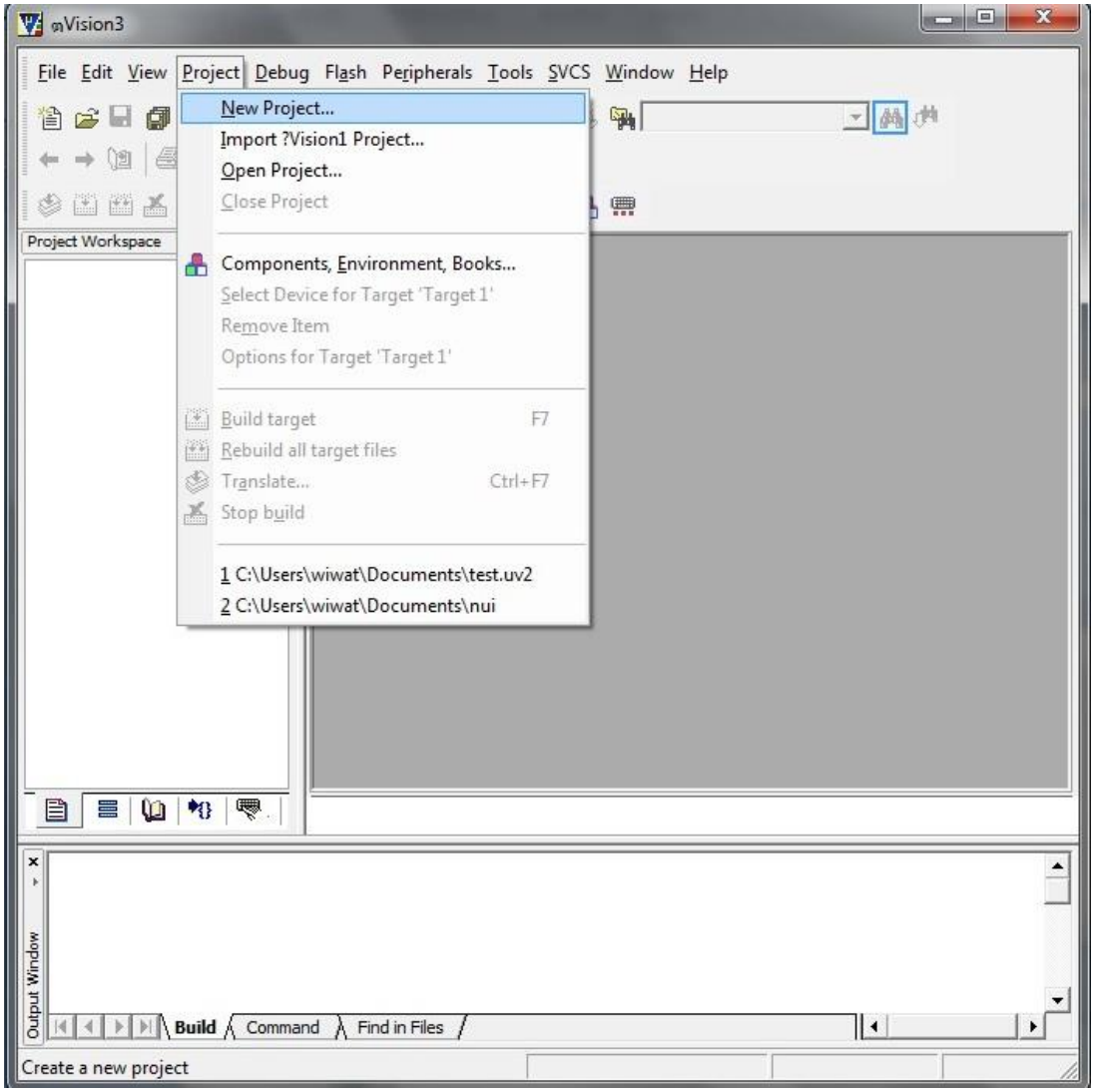
2. เครื่องคอมพิวเตอร์ PC พร้อมโปรแกรมสำหรับการเขียนและคอมไพล์ภาษา C - โปรแกรม Keil51 v.xx

3. สายต่อพอร์ตอนุกรม

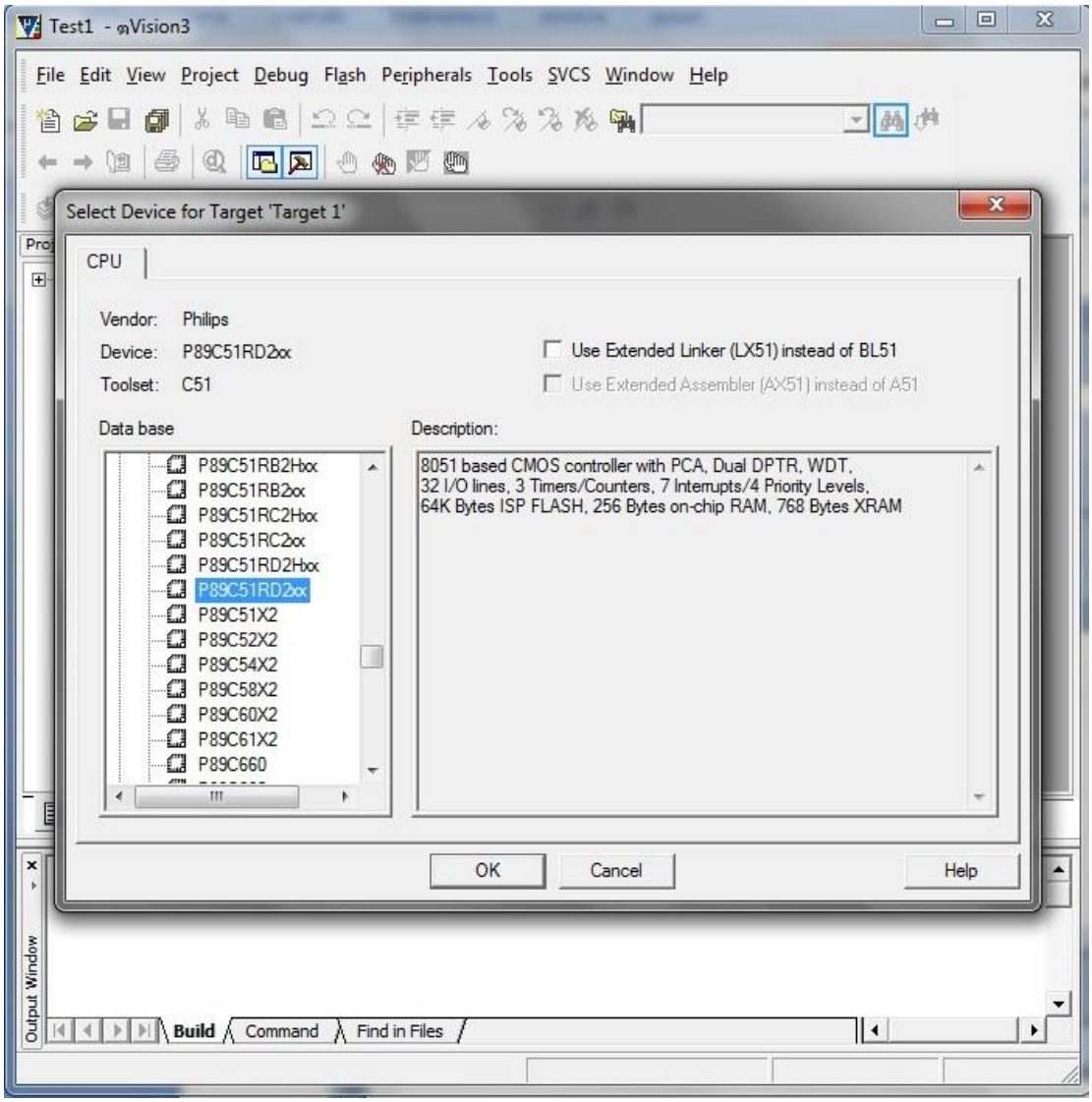
การใช้งาน Keil51

1. ใช้ IDE สำหรับพัฒนาชุดคำสั่งของ MCS-51 ด้วยภาษา C โดยใช้ Keil51เพื่อเขียนโปรแกรม Lab07_x.c

2. สร้างโปรเจ็คใหม่โดยเลือก New project ตั้งชื่อเป็น Lab07 แล้วกด ok

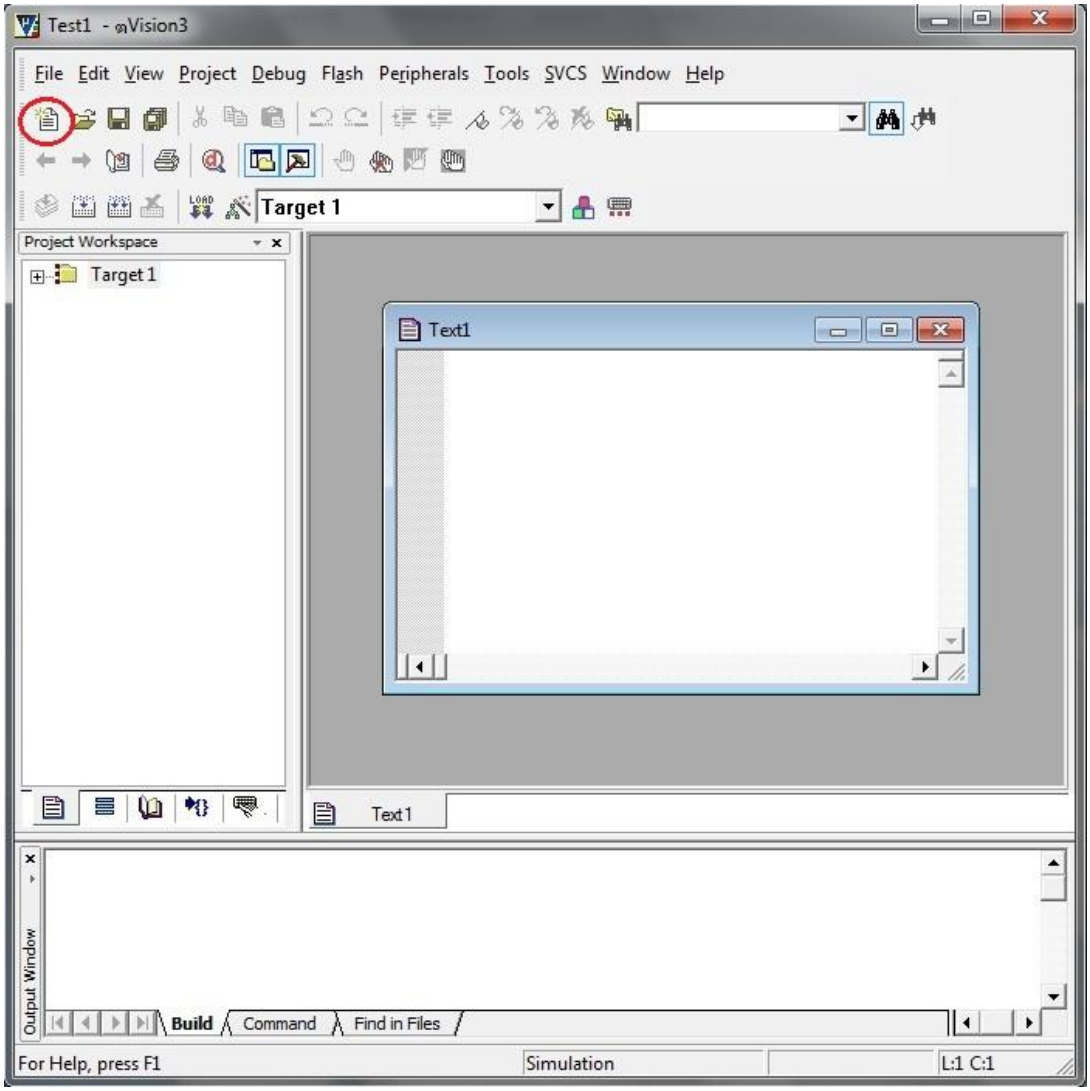

µVision3
File Edit View Project Debug Flash Peripherals Tools SVCS Window Help
New Project...
Import ?Vision1 Project...
Open Project...
Close Project
Project Workspace
Components, Environment, Books...
Select Device for Target 'Target 1'
Remove Item
Options for Target 'Target 1'
Build target F7
Rebuild all target files
Translate... Ctrl+F7
Stop build
1 C:\Users\wiwat\Documents\test.uv2
2 C:\Users\wiwat\Documents\nui
Output Window
Build Command Find in Files
Create a new project

3. เลือก CPU : Philips P89C51RD2xx

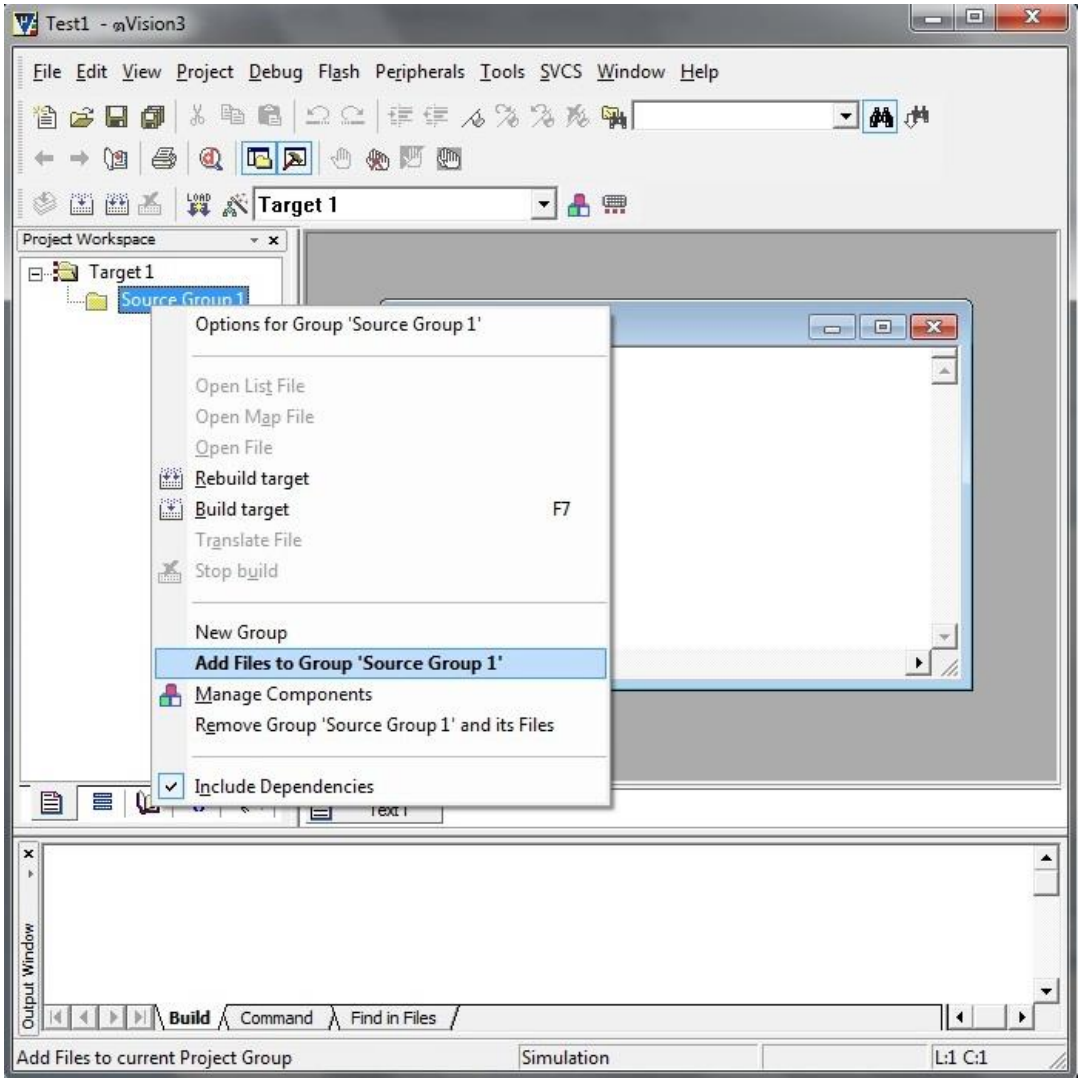


4. จากนั้นคลิ้กที่ปุ่มวงกลมสีแดง เพื่อสร้างหน้าต่างสำหรับเขียนโปรแกรม(หน้าต่าง Text1)

สร้างไฟล์ใหม่ เขียนโปรแกรมและ save โดยใช้ชื่อ Lab07_x.c (x แทนหมายเลขการทดลอง)

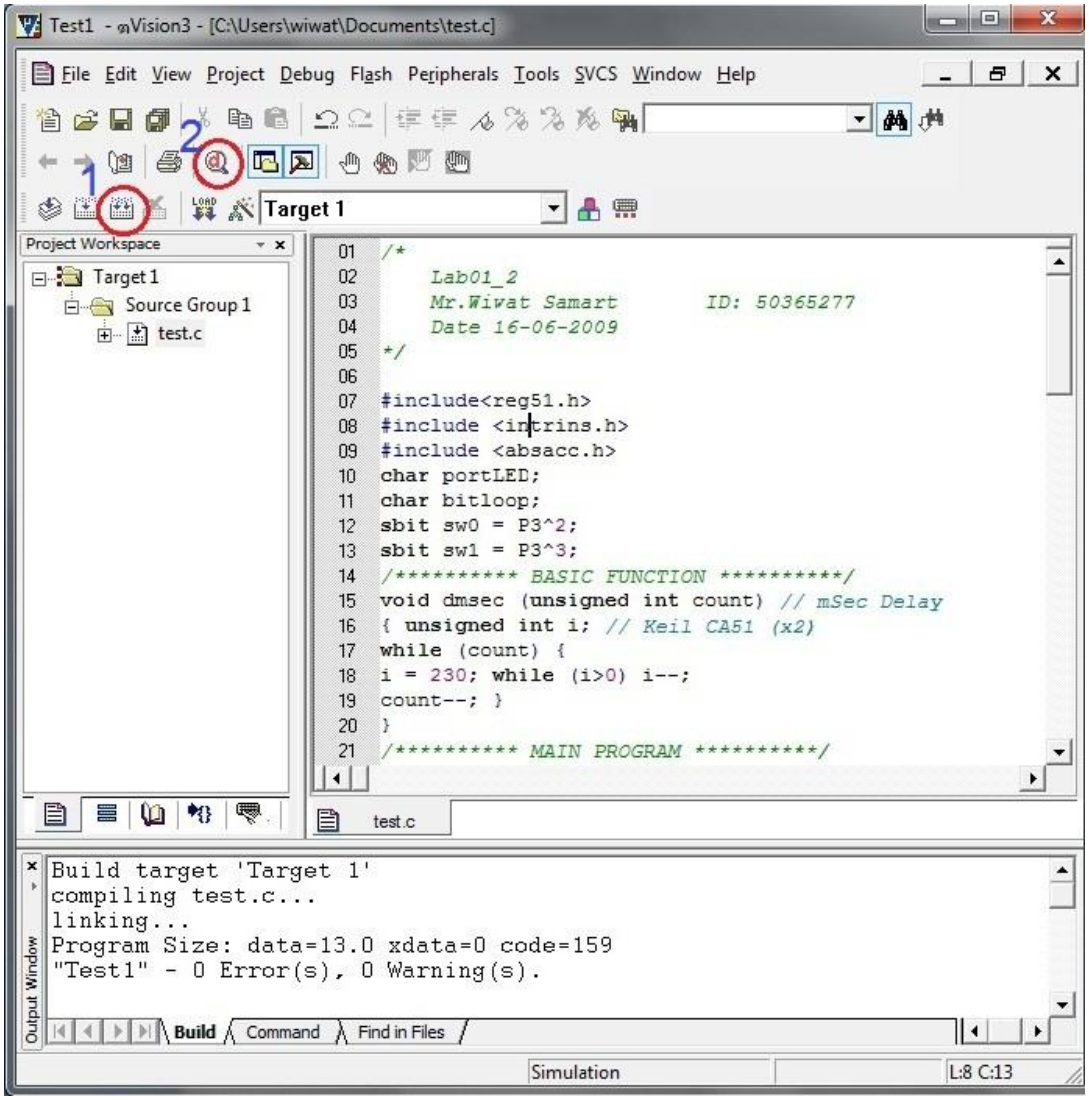


5. คลิ้กขวาที่ Source Group 1 แล้วเลือก Add file to group ‘Source Group 1’ แล้วเลือก File Lab07_x.c



6. เขียนโปรแกรมให้เสร็จแล้วคลิ้กที่ วงกลมที่ 1 เพื่อทำการ compile ถ้าหากว่าไม่พบ Error โดยที่ถ้าเกิด error ขึ้น โปรแกรมจะแสดงว่า error กี่ที่ (ดู windows output ด้านล่างของโปรแกรม) คลิ้กที่ วงกลมที่ 2 เพื่อให้ keil ตรวจสอบและทำการ debug พร้อมทั้งตรวจสอบขนาดของโปรแกรมที่เราสร้างขึ้นและพร้อมสำหรับการทดสอบการทำงานโปรแกรม

7. Build โปรแกรมที่เขียน และให้เลือก option→ create hex file โดยเลือกที่ check box Hex file ที่ ได้จะมีชื่อเหมือกับชื่อโปรเจ็ค คือ Lab07.hex



6. บันทึกผลการทดลอง และอธิบายการทำงานของแต่ละคำสั่งในโปรแกรม และทำออกมาในรูปแบบรายงาน

7. ให้นิสิตทำการทดลองที่เหลือ โดยเมื่อทำการทดลองใหม่ให้สร้างไฟล์ใหม่ เขียนโปรแกรมและ save เป็นชื่อใหม่ เช่น สร้างไฟล์ใหม่ชื่อ Lab07_2.c (2แทนหมายเลขการทดลองที่ 2)

8. จากนั้น Click ที่ Source group แล้ว click ขวาที่ Lab07_1.c ซึ่งอยู่ใน Source group เลือก Remove file ' Lab07_1.c' เพื่อเอา file Lab07_1.c ออกจาก project

9. Click ขวาที่ Source group เลือก Add file to group แล้วเลือก File Lab07_2.c

10. ทำเช่นนี้จนครบทุกการทดลอง

การทดลองที่ 7.1

```
/*******************************************/
/* Example Program For ET-BASE51 V3.0(ED2) */
/* MCU      : AT89C51ED2(XTAL=29.4912 MHz) */
/*          : Frequency Bus = 58.9824 MHz  */
/* Compiler : Keil C51 (V7.50)             */
/* Write By : Eakachai Makarn(ETT CO.,LTD.)*/
/*******************************************/
// Demo I2C RTC(DS1307) Interrupt Mode
// SCL = P3.7
// SDA = P3.6
// INT = P3.3(INT1)

/* Include  Section */
#include <reg52.h>
                    // Standard 8052 SFR : File
#include <stdio.h>          // For printf I/O functions

/* AT89C51ED2 SFR */
sfr CKCON = 0x8F;
                    // Clock Control

/* Timer1 Baudrate (29.4912 MHz x 2 )
   Reload = 256 - [58.9824 MHz / (384 x Baud)]
   1200   = 0x80(256-128=128)
   2400   = 0xC0(256-94=192)
   4800   = 0xE0(256-32=224)
   9600   = 0xF0(256-16=240)
   19200  = 0xF8(256-8=248)
   38400  = 0xFC(256-4=252)
*/
sbit scl = P3^7;
                    // I2C SCL Signal
sbit sda = P3^6;
                    // I2C SDA Signal

/* Pototype  Section */
void ds1307_write_byte(unsigned char,unsigned
char);              // Setup RTC
unsigned char ds1307_read_byte(unsigned char);
          // Read RTC
void i2c_send_byte(unsigned char);
          // Write Byte I2C
unsigned char i2c_get_byte(void);
                              // Read Byte I2C
void delay_i2c(void);
          // Short Delay For I2C Bus Interface
```

```
void delay(unsigned long);

        // Delay Time Function(1..4294967295)

/*---------------------------------------------

The main C function.  Program execution Here

---------------------------------------------*/

void main (void)

{

  unsigned char j;

        unsigned char last_second;


  CKCON = 0x01;

                // Initial X2 Mode (58.9824 MHz)


        /* Initial MCS51 Serial Port */

  TMOD &= 0x0F;

        // Reset old Timer1 Mode Config

  TMOD |= 0x20;

        // Update Timer1 = 8 Bit Auto Reload

  SCON  = 0x50;

                // Serial Port Mode 1 (N,8,1)

  ES    = 0;

        // Disable Serial Interupt

  ET1   = 0;

        // Disable Timer1 Interrupt

  PCON &= 0x7F;

        // SMOD1 = 0 (Disable Double Baudrate)

  TH1   = 0xF0;

// Setup Timer1 Baudrate 9600BPS / 58.9824 MHz

  TL1   = 0xF0;

  TR1   = 1;

        // Start Timer1 Generate Baudrate

  TI    = 1;

        // Set TI to send First char of UART

        scl = 1;

        // Makesure I2C Stop Condition

        sda = 1;


        ds1307_write_byte(7,0x80);

        // Disable Square Wave Output

        ds1307_write_byte(2,0);

        // Set Hour          = 00

        ds1307_write_byte(1,0);

        // Set Minute = 00
```

```
        ds1307_write_byte(0,0);

        // Start RTC Clock + Set Second = 00


        printf ("Hello This is ET-BASE51
V3.0(ED2)\n");              // Display Start Menu
        printf ("Demo Test RTC:DS1307 Polling
Read\n");
        printf ("HR:MN:SC\n");
        delay(100);

        while(1)
        {
    printf("\r");
                // Restart Print
    j = ds1307_read_byte(2);
                // Get Hour
    printf("%02BX",j);
    printf(":");

         j = ds1307_read_byte(1);
                // Get Minute
         printf("%02BX",j);
    printf(":");

         j = ds1307_read_byte(0);
                // Get Second
         last_second = j;
                // Save Second Reference
         printf("%02BX",j);
         do

// Repeat Get Second until Second Change
           {
             j = ds1307_read_byte(0);
        // Get Second
       }
     while(last_second == j);
        // Repeat if Second Not Change
        }
}



/*****************************/
/* Write Data 1-Byte to DS1307 */
/*****************************/
void ds1307_write_byte(unsigned char
ds1307_address,unsigned char ds1307_data)
```

```
{
  sda = 0;
                    // I2C Start Condition
  scl = 0;
  delay_i2c();

  i2c_send_byte(0xD0);

// Send ID Code DS1307,Write (1101000+0)
  sda = 1;
                          // Release SDA
  scl = 1;
                          // Start ACK Clock
  delay_i2c();
  while(sda) {}
  scl = 0;
                    // End ACK Clock
  delay_i2c();

  i2c_send_byte(ds1307_address);
        // Send DS1307 Address
  sda = 1;
                    // Release SDA
  scl = 1;
                    // Start ACK Clock
  delay_i2c();
  while(sda) {}
  scl = 0;
                    // End ACK Clock
  delay_i2c();

  i2c_send_byte(ds1307_data);
        // Send DS1307 Data
  sda = 1;
                          // Release SDA
  scl = 1;
                    // Start ACK Clock
  delay_i2c();
  while(sda) {}
  scl = 0;
                    // End ACK Clock
  delay_i2c();
```

```
  sda = 0;
                          // Stop Bit(End of Data)

  scl = 1;
                                  // I2C Stop
Condition
  delay_i2c();
  sda = 1;

 return;
}

/*******************************/
/* Read Data 1-Byte From DS1307 */
/*******************************/
unsigned char ds1307_read_byte(unsigned char
ds1307_address)
{
  unsigned char ds1307_data;
        // Dummy Byte

  sda = 0;
                          // I2C Stat condition
  scl = 0;
  delay_i2c();

  i2c_send_byte(0xD0);

// Send ID Code DS1307,Write (1101000+0)
  sda = 1;
                                  // Release SDA
  scl = 1;
                          // Start ACK Clock
  delay_i2c();
  while(sda) {}
  scl = 0;
                          // End ACK Clock
  delay_i2c();

  i2c_send_byte(ds1307_address);
        // Send DS1307 Address
  sda = 1;
                                  // Release SDA
  scl = 1;
                          // Start ACK Clock
```

```
  delay_i2c();
  while(sda) {}

  scl = 0;
                              // End ACK Clock
  delay_i2c();

  scl = 1;
                                    // I2C Stop
Condition
  delay_i2c();
  sda = 1;

  // New Start For Read //
  sda = 0;
                                    // I2C Stat
condition
  scl = 0;
  delay_i2c();

  i2c_send_byte(0xD1);

// Send ID Code DS1307,Read (1101000+1)
  sda = 1;
                                    // Release SDA
  scl = 1;
                                    // Start ACK Clock
  delay_i2c();
  while(sda) {}

  scl = 0;
                                    // End ACK Clock
  delay_i2c();

  ds1307_data = i2c_get_byte();
                                    //
Read 1-Byte From DS1307

  sda = 1;
        // Send Stop Bit (End of Read Data)
  scl = 1;
                    // Start Stop Bit Clock
  delay_i2c();
  scl = 0;
                                    // End Stop Bit Clock
  delay_i2c();
```

```
  scl = 1;

                                    // I2C Stop
Condition
  delay_i2c();
  sda = 1;

  return ds1307_data;
}

/*******************************/
/* Send Data 8 Bit to I2C Bus */
/*******************************/
void i2c_send_byte(unsigned char i)
{
 char j;

                              // Bit Counter Send

 for(j = 0;j < 8;j++)

                      // 8-Bit Counter Send Data
 {

  if((i & 0x80) == 0x80)

          // Send MSB First
   {sda = 1;}

          // Send Data = 1
  else
   {sda = 0;}

                  // Send Data = 0

  scl = 1;

                  // Release SDA
  delay_i2c();

  scl = 0;

                              // Next Bit Send
  delay_i2c();

  i <<= 1;

          // Shift Data For Send (MSB <- LSB)
 }
  return;
}
```

```
/*******************************/
/* Get Data 8 Bit From I2C Bus */
/*******************************/
unsigned char i2c_get_byte(void)
{
 unsigned char i;
                    // Result Byte Buffer
 char j;
                    // Bit Counter Read Data

 for(j = 0; j < 8; j++)
                    // 8-Bit Counter Read Data
 {
  i <<= 1;
                                    // Shift Result
Save (MSB <- LSB)

  sda = 1;
                            // Release Data
  scl = 1;
                    // Strobe Read SDA
  delay_i2c();

  if(sda == 1)
   {
            i |= 0x01;
                                // Save Bit Data = 1
          }
  else
   {
            i &= 0xFE;
                                // Save Bit Data = 0
          }
    scl = 0;
                        // Next Bit Read
  delay_i2c();
 }
  return i;
}

/*************************************/
/* Delay For I2C Bus Device Interface */
/*************************************/
void delay_i2c(void)
{
```

```
  unsigned char i;
  i = 100;
                                  // Delay Counter
  while(i > 0) {i--;}
              // Loop Decrease Counter
  return;
}
```

```
/*****************************************/
/* Long Delay Time Function(1..4294967295) */
/*****************************************/
void delay(unsigned long i)
{
  while(i > 0) {i--;}
              // Loop Decrease Counter
  return;
}
```

การทดลองที่ 7.2

```
/*****************************************/
/* Example Program For ET-BASE51 V3.0(ED2) */
/* MCU      : AT89C51ED2(XTAL=29.4912 MHz) */
/*          : Frequency Bus = 58.9824 MHz  */
/* Compiler : Keil C51 (V7.50)             */
/* Write By : Eakachai Makarn(ETT CO.,LTD.)*/
/*****************************************/
// Demo I2C RTC(DS1307) Interrupt Mode
// SCL = P3.7
// SDA = P3.6
// INT = P3.3(INT1)
```

```
/* Include  Section */
#include <reg52.h>
// Standard 8052 SFR : File
#include <stdio.h>
// For printf I/O functions
/* AT89C51ED2 SFR */
sfr CKCON = 0x8F;
// Clock Control
/* Timer1 Baudrate (29.4912 MHz x 2 )
   Reload = 256 - [58.9824 MHz / (384 x Baud)]
   1200   = 0x80(256-128=128)
   2400   = 0xC0(256-94=192)
```

```
  4800   = 0xE0(256-32=224)

  9600   = 0xF0(256-16=240)

  19200  = 0xF8(256-8=248)

  38400  = 0xFC(256-4=252)

*/



sbit scl = P3^7;

// I2C SCL Signal

sbit sda = P3^6;

// I2C SDA Signal

// Bit Boolean 0=False,1=True //

bit int1_status;

                              // INT1 Interrupt Flag Status

/* Pototype  Section */

void ds1307_write_byte(unsigned char,unsigned char);

                        // Setup RTC

unsigned char ds1307_read_byte(unsigned char);

                              // Read RTC

void i2c_send_byte(unsigned char);

                                          // Write Byte I2C

unsigned char i2c_get_byte(void);

                                    // Read Byte I2C
```

```
void delay_i2c(void);

                              // Short Delay For I2C Bus
Interface



/*-----------------------------------------

The main C function.  Program execution Here

-----------------------------------------*/

void main (void)

{

   unsigned char j;



   CKCON = 0x01;

         // Initial X2 Mode (58.9824 MHz)



         /* Initial MCS51 Serial Port */

   TMOD &=  0x0F;

                              // Reset old Timer1 Mode Config

   TMOD |=  0x20;

                     // Update Timer1 = 8 Bit Auto Reload

   SCON  =  0x50;

                              // Serial Port Mode 1 (N,8,1)

   ES    =  0;

                              // Disable Serial Interupt
```

```
ET1   =  0;
                        // Disable Timer1 Interrupt
PCON &=  0x7F;
                // SMOD1 = 0 (Disable Double Baudrate)
TH1   =  0xF0;
        // Setup Timer1 Baudrate 9600BPS / 58.9824 MHz
TL1   =  0xF0;
TR1   =  1;
                        // Start Timer1 Generate Baudrate
TI    =  1;
                // Set TI to send First char of UART

        /* Setup INT1 Interrupt Control */
        EX1   =  1;
                // Enable External INT1
        IT1   =  1;

        // INT1 Trig on Falling Edge only
        EA    =  1;

                // Enable Global Interrupt

        scl = 1;

        // Makesure I2C Stop Condition
        sda = 1;


        ds1307_write_byte(7,0x90);
                // Enable Square Wave Output = 1Hz
        ds1307_write_byte(2,0);
                // Set Hour          = 00
        ds1307_write_byte(1,0);
                // Set Minute = 00
        ds1307_write_byte(0,0);
                // Start RTC Clock + Set Second = 00


        printf ("Hello This is ET-BASE51 V3.0(ED2)\n");
                // Display Start Menu
        printf ("Demo Test RTC:DS1307 Interrupt Read\n");
        printf ("HR:MN:SC\n");

        while(1)
        // Loop Continue
        {
         while(int1_status == 1)
                        // Wait Interrupt Complete
         {
           printf("\r");
                                // Restart Print
```

```
        j = ds1307_read_byte(2);
                                        // Get Hour
        printf("%02BX",j);
        printf(":");
        j = ds1307_read_byte(1);
                                        // Get Minute
        printf("%02BX",j);
        printf(":");
              j = ds1307_read_byte(0);
                                        // Get Second
              printf("%02BX",j);
                    int1_status = 0;
                                        // Reset Status
             }
            }
}

/*******************************/
/* Write Data 1-Byte to DS1307 */
/*******************************/
void ds1307_write_byte(unsigned char
ds1307_address,unsigned char ds1307_data)
{
sda = 0;

                    // I2C Start Condition

scl = 0;
delay_i2c();

i2c_send_byte(0xD0);
          // Send ID Code DS1307,Write (1101000+0)
sda = 1;

                    // Release SDA
scl = 1;

          // Start ACK Clock
delay_i2c();
while(sda) {}
scl = 0;

          // End ACK Clock
delay_i2c();

i2c_send_byte(ds1307_address);
                              // Send DS1307 Address
```

```
  sda = 1;

                    // Release SDA

  scl = 1;

          // Start ACK Clock

  delay_i2c();

  while(sda) {}

  scl = 0;

          // End ACK Clock

  delay_i2c();


  i2c_send_byte(ds1307_data);
                                  // Send DS1307 Data

  sda = 1;

                    // Release SDA

  scl = 1;

          // Start ACK Clock

  delay_i2c();

  while(sda) {}

  scl = 0;

          // End ACK Clock

  delay_i2c();

  sda = 0;

                    // Stop Bit(End of Data)


  scl = 1;

                    // I2C Stop Condition

  delay_i2c();

  sda = 1;


 return;

}


/*********************************/

/* Read Data 1-Byte From DS1307 */

/*********************************/

unsigned char ds1307_read_byte(unsigned char
ds1307_address)

{

  unsigned char ds1307_data;

// Dummy Byte

  sda = 0;
```

```
// I2C Stat condition
  scl = 0;
  delay_i2c();
  i2c_send_byte(0xD0);
// Send ID Code DS1307,Write (1101000+0)
  sda = 1;
                    // Release SDA
  scl = 1;
          // Start ACK Clock
  delay_i2c();
  while(sda) {}
  scl = 0;
          // End ACK Clock
  delay_i2c();

  i2c_send_byte(ds1307_address);
                          // Send DS1307 Address
  sda = 1;
                    // Release SDA
  scl = 1;
          // Start ACK Clock
delay_i2c();
while(sda) {}
scl = 0;
          // End ACK Clock
delay_i2c();


scl = 1;
                    // I2C Stop Condition
delay_i2c();
sda = 1;

// New Start For Read //
sda = 0;
                    // I2C Stat condition
scl = 0;
delay_i2c();
```

```
i2c_send_byte(0xD1);
        // Send ID Code DS1307,Read (1101000+1)
sda = 1;
                // Release SDA
scl = 1;
        // Start ACK Clock
delay_i2c();
while(sda) {}
scl = 0;
        // End ACK Clock
delay_i2c();
ds1307_data = i2c_get_byte();
                        // Read 1-Byte From DS1307

sda = 1;
                // Send Stop Bit (End of Read Data)
scl = 1;
        // Start Stop Bit Clock
delay_i2c();
scl = 0;
        // End Stop Bit Clock
delay_i2c();

scl = 1;
                // I2C Stop Condition
delay_i2c();
sda = 1;

return ds1307_data;
}

/*****************************/
/* Send Data 8 Bit to I2C Bus */
/*****************************/
void i2c_send_byte(unsigned char i)
{
 char j;
        // Bit Counter Send
```

```
for(j = 0;j < 8;j++)
                        // 8-Bit Counter Send Data
{
  if((i & 0x80) == 0x80)
                        // Send MSB First
    {sda = 1;}
                        // Send Data = 1
  else
    {sda = 0;}
                                // Send Data = 0

  scl = 1;

        // Release SDA
  delay_i2c();

  scl = 0;

        // Next Bit Send
  delay_i2c();

  i <<= 1;

        // Shift Data For Send (MSB <- LSB)
}
  return;
}

/*******************************/
/* Get Data 8 Bit From I2C Bus */
/*******************************/
unsigned char i2c_get_byte(void)
{
 unsigned char i;
                        // Result Byte Buffer
 char j;

        // Bit Counter Read Data

 for(j = 0; j < 8; j++)
                        // 8-Bit Counter Read Data
 {
  i <<= 1;

                // Shift Result Save (MSB <- LSB)
  sda = 1;

        // Release Data
```

```
  scl = 1;
                                                    //
Strobe Read SDA
  delay_i2c();


  if(sda == 1)
   {
              i |= 0x01;

           // Save Bit Data = 1
            }
  else
   {
              i &= 0xFE;

           // Save Bit Data = 0
            }

  scl = 0;
                                  // Next Bit Read
  delay_i2c();
}


 return i;
}

/*************************************/
/* Delay For I2C Bus Device Interface */
/*************************************/
void delay_i2c(void)
{
  unsigned char i;
  i = 100;

                          // Delay Counter
  while(i > 0) {i--;}
                                    // Loop Decrease Counter
  return;
}



/*********************************/
/* External INT1 Service Function */
/* Trig By RTC:DS1307 Every 1Hz   */
/*********************************/
void external1 (void) interrupt 2 using 1
                                    // External-INT1 : Bank-1
{
```

```
  int1_status = 1;                    // Set Interrupt Status
}
```

การบ้าน

1. เขียนโปรแกรมนับความเร็วรอบมอเตอร์โดยใช้ Real Time Clock เป็นฐานเวลา แสดงผลทางจอ LCD

การส่งงาน

ส่ง code ภาษา C มาทาง email และ เขียนอธิบาย program พอเข้าใจส่งพร้อม Flow chart และแสดงผล

Reference

http://www.keil.com/support/man/docs/c166/c166_libref.htm