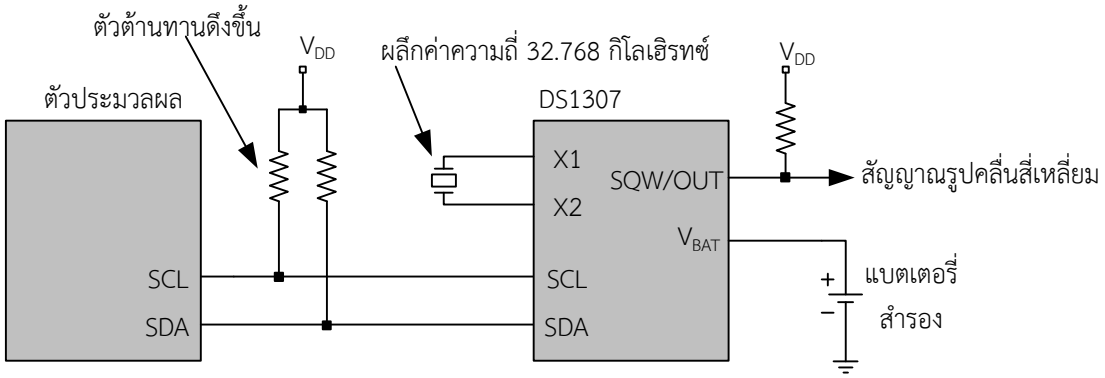
240-319 Embedded System Developer Module
LAB 14 : DS-1307 Real Time Clock

14.1 รายการอุปกรณ์ที่ใช้ในการทดลอง

- บอร์ด Arduino UNO
- บอร์ด CoE Arduino Shield
- มอดูล DS-1307
- มอดูล LCD 1602 แบบ I2C
- มอดูล PCF-8574
- คีย์แพ็ดแบบเมมเบรนชนิด 4x4
- สายเชื่อมต่อวงจร

14.2 ไอซีนาฬิกาเวลาจริง DS1307

DS1307 เป็นไอซีที่ถูกออกแบบมาสำหรับใช้เป็นวงจรอ้างอิงเวลาจริง ซึ่งสามารถเก็บค่าวันที่ วัน เดือน ปี ชั่วโมง นาที และวินาทีไว้ได้อย่างถูกต้องโดยใช้กระแสไฟฟ้าในการทำงานต่ำกว่า 500 นาโนแอมแปร์ ส่งผลให้สามารถใช้แบตเตอรี่สำรองขนาดเล็กเพื่อจ่ายไฟให้กับตัวไอซีให้ทำงานได้อย่างต่อเนื่องในกรณีที่ไม่มีไฟเลี้ยงจากแหล่งจ่ายไฟหลัก ไอซีถูกออกแบบมาให้สามารถปรับวันเวลาของปีได้ถูกต้องตามปฏิทินทั้งในกรณีของปีปกติสุรทิน ซึ่งมีจำนวนวัน 365 วัน และปีอธิกสุรทินซึ่งมีจำนวนวันเท่ากับ 366 วัน ไอซีถูกออกแบบมาให้เข้ากันได้กับการเชื่อมต่อแบบไอสแควร์ซี โดยมีรูปแบบการเชื่อมต่อกับตัวประมวลผลดังรูปที่ 14.6 ไอซี DS1307 มีความแตกต่างจากไอซีที่รองรับมาตรฐานการเชื่อมต่อแบบทู-ไวร์อื่น ๆ ตรงที่ไม่สามารถเลือกบิตเลขที่อยู่ A0-A2 โดยผู้ใช้ได้ ส่งผลให้บนบัสหนึ่ง ๆ สามารถมีไอซี DS1307 ได้เพียงตัวเดียวเท่านั้น



รูปที่ 14.6 วงจรใช้งานของไอซี DS1307 เมื่อทำการเชื่อมต่อกับตัวประมวลผลผ่านการเชื่อมต่อแบบไอสแควร์ซี

ไอซี DS1307 มีหน่วยความจำชนิดแรมสถิตภายในจำนวน 64 ตำแหน่ง แต่ละตำแหน่งเก็บข้อมูลได้หนึ่งไบต์ ตำแหน่งที่ 0-6 ถูกสงวนเอาไว้สำหรับเก็บค่าวันและเวลาของระบบ ดังแสดงในรูปที่ 14.7 ตำแหน่งที่ 7 ไว้สำหรับเก็บไบต์ควบคุมการทำงานของตัวไอซีเอง ส่วนตำแหน่งที่ 8-63 สามารถใช้เป็นหน่วยความจำไม่ลบเลือน (non volatile memory) ได้ แต่หน่วยความจำทั้ง 56 ไบต์นี้จะยังคงค่าอยู่ได้เมื่อมีแบตเตอรี่สำรองเท่านั้น

ค่าวันและเวลาที่เก็บในหน่วยความจำตำแหน่งที่ 0-6 จะถูกเข้ารหัสในรูปแบบของเลขบีซีดี ดังแสดงในรูปที่ 14.7 หน่วยความจำตำแหน่งที่ 0 ใช้บิตที่ 0-6 ในการเก็บค่าวินาที ส่วนบิตที่ 7 มีชื่อว่า CH (Clock Halt) ใช้ในการเปิดปิดการทำงานของวงจรออสซิลเลเตอร์ของวงจรนาฬิกา หากบิตนี้ถูกตั้งค่าให้เป็นตรรกะสูงจะส่งผลให้วงจรออสซิลเลเตอร์หยุดทำงานและนาฬิกาจะหยุดเดิน ค่าโดยปริยายของบิตนี้จะมีค่าตรรกะสูง ดังนั้นก่อนเริ่มใช้งานจะต้องมีการตั้งค่าบิตนี้ให้กลายเป็นตรรกะต่ำเสียก่อน

หน่วยความจำตำแหน่งที่ 2 ใช้ในการเก็บค่าหลักชั่วโมง โดยมีบิตที่ 6 เป็นตัวเลือกว่าจะให้ทำงานเป็นนาฬิกาแบบ 24 ชั่วโมง หรือแบบ 12 ชั่วโมง หากบิตนี้มีค่าเป็นตรรกะสูงจะหมายถึงการทำงานในแบบ 12 ชั่วโมง และจะใช้บิตที่ 5 เป็นตัวแสดงสถานะ AM/PM ของค่าชั่วโมง แต่หากบิตที่ 6 มีค่าเป็นตรรกะต่ำจะหมายถึงการทำงานในแบบ 24 ชั่วโมง

หน่วยความจำตำแหน่งที่ 3 ใช้ในการเก็บค่าวันในหนึ่งสัปดาห์ โดยค่าที่เป็นไปได้ คือ 1-7 โดยกำหนดให้เลข 1 แทนวันอาทิตย์ และเลข 2-7 แทนวันจันทร์-เสาร์ตามลำดับ

หน่วยความจำตำแหน่งที่ 6 ใช้ในการเก็บค่าปี โดยเก็บแค่ค่า 00-99 ซึ่งใช้แทนปีได้ตั้งแต่ปี 2000-2099 โดยไอซีถูกออกแบบมาให้สามารถปรับวันเวลาของปีได้ถูกต้องตามปฏิทินเกรกอเรียน (Gregorian calendar) ทั้งในกรณีของปีปกติสุรทินซึ่งมีจำนวนวัน 365 วัน และปีอธิกสุรทินซึ่งมีจำนวนวันเท่ากับ 366 วัน

หน่วยความจำตำแหน่งที่ 7 ใช้ในการควบคุมการทำงานสร้างสัญญาณรูปคลื่นสี่เหลี่ยมออกทางขา SQW/OUT โดยหากค่าในบิตที่ 4 มีค่าตรรกะสูงจะเป็นการเปิดทางให้วงจรสามารถสร้างความถี่ออกไปได้ ส่วนบิตที่ 7 เป็นค่าสถานะทางตรรกะปัจจุบันที่กำลังถูกส่งออก ณ ขา SQW/OUT ในขณะนั้น ส่วนบิตที่ 0-1 ใช้เป็นบิต RS0-RS1 สำหรับกำหนดค่าความถี่เอาต์พุตซึ่งเลือกได้ 4 ค่า โดยหากค่าใน RS1-RS0 เท่ากับ 00, 01, 10, และ 11 จะส่งผลให้ได้ค่าความถี่เอาต์พุตเท่ากับ 1, 4.096, 8.192, และ 32.768 กิโลเฮิรทซ์ตามลำดับ

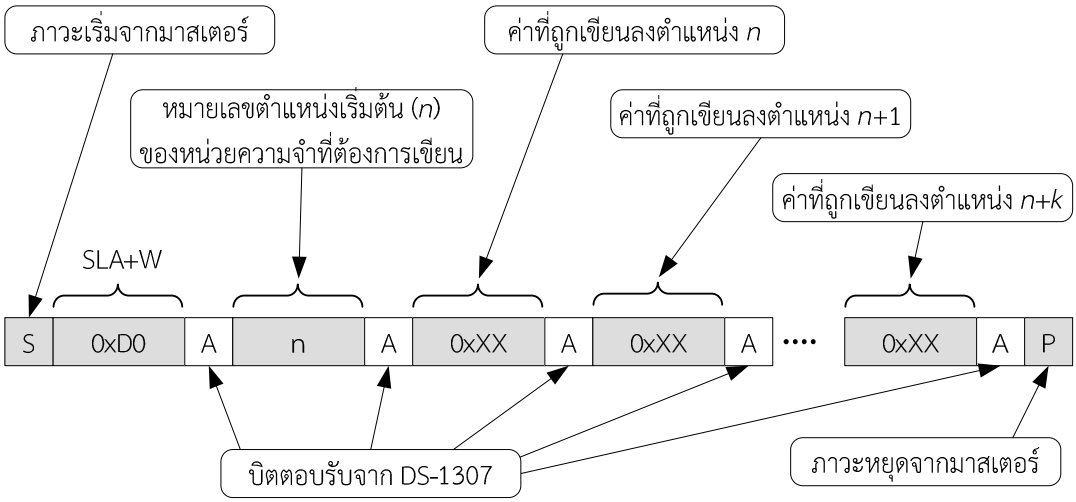
การทำงานของไอซี DS1307 ถูกออกแบบมาให้ใช้การเชื่อมต่อแบบไอสแควร์ซี มีหมายเลขที่อยู่ขนาด 7 บิตเท่ากับ 1101000_2 ดังนั้นการส่งหมายเลขที่อยู่ของไอซี DS1307 พร้อมบิตเขียน (SLA+W) จึงมีค่าเท่ากับ 11010000_2 หรือ 0xD0 ในขณะที่การส่งหมายเลขที่อยู่พร้อมบิตอ่าน (SLA+R) จะมีค่าเท่ากับ 11010001_2 หรือ 0xD1 ไอซีสามารถทำงานได้ 2 แบบวิธี ได้แก่ แบบวิธีสเลฟรีซีฟเวอร์ และแบบวิธีสเลฟทรานสมิตเตอร์

ในแบบวิธีสเลฟรีซีฟเวอร์ ฝั่งมาสเตอร์ (ซึ่งมักเป็นตัวประมวลผล) จะเริ่มต้นด้วยการส่งภาวะเริ่มตามด้วยหมายเลขที่อยู่ของไอซี DS1307 พร้อมบิตเขียน (SLA+W) ค่า 0xD0 ลงสู่บัสดังรูปที่ 14.8 หากพบว่ามีการตอบรับจากสเลฟ (ซึ่งในที่นี้คือ DS1307) ฝั่งมาสเตอร์จะส่งค่าตำแหน่งหน่วยความจำแรมสถิตที่ต้องการเขียนเป็นตำแหน่งแรกไปให้สแลฟ และตามด้วยค่าที่ต้องการเขียนลงในแรมสถิตตำแหน่งแรก ค่าถัดมาที่ส่งมาจากมาสเตอร์จะถูกสเลฟนำมาเขียนในแรมสถิตตำแหน่งถัดไป และเป็นเช่นนี้ไปจนกว่ามาสเตอร์จะส่งภาวะหยุดลงสู่บัส

ตำแหน่ง	บิตที่ 7	บิตที่ 6	บิตที่ 5	บิตที่ 4	บิตที่ 3	บิตที่ 2	บิตที่ 1	บิตที่ 0	ค่าที่เป็นไปได้
0	CH	วินาที (หลักสิบ)			วินาที (หลักหน่วย)				00-59
1	ไม่ใช้(0)	นาที (หลักสิบ)			นาที (หลักหน่วย)				00-59
2	ไม่ใช้(0)	24ชม.	ชั่วโมง (หลักสิบ)		ชั่วโมง (หลักหน่วย)				00-23
		12ชม.	AM/PM	ชั่วโมง (หลักสิบ)					(1-12) และ AM/PM
3	ไม่ใช้(0)	ไม่ใช้(0)	ไม่ใช้(0)	ไม่ใช้(0)	ไม่ใช้(0)	วัน (1=วันอาทิตย์)			01-07
4	ไม่ใช้(0)	ไม่ใช้(0)	วันที่ (หลักสิบ)		วันที่ (หลักหน่วย)				01-31
5	ไม่ใช้(0)	ไม่ใช้(0)	ไม่ใช้(0)	เดือน (หลักสิบ)	เดือน (หลักหน่วย)				01-12
6	ปี (หลักสิบ)				ปี (หลักหน่วย)				00-99
7	OUT	ไม่ใช้(0)	ไม่ใช้(0)	SQWE	ไม่ใช้(0)	ไม่ใช้(0)	RS1	RS0	0x00-0xFF

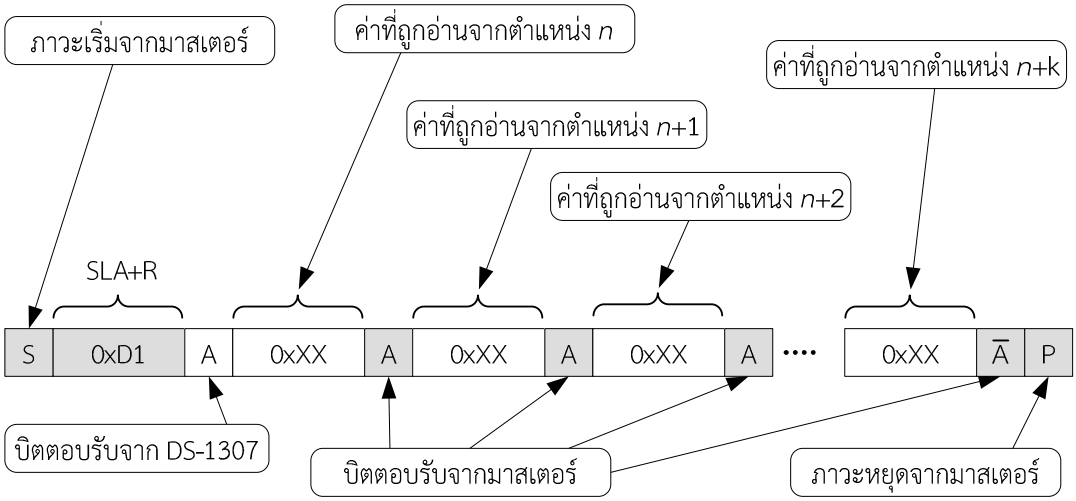
รูปที่ 14.7 ข้อมูลที่เก็บอยู่ในหน่วยความจำตำแหน่ง 0-7 ของไอซี DS1307

จะเห็นว่าก่อนที่จะทำการเขียนค่าลงสู่ DS1307 จะต้องมีการระบุก่อนว่าต้องการเริ่มต้นติดต่อกับหน่วยความจำแรมสถิตตำแหน่งใดเป็นลำดับแรก โดยในตัวไอซี DS1307 จะรับค่าหมายเลขตำแหน่งเริ่มต้นมาเก็บในเรจิสเตอร์ตัวชี้ตำแหน่งหน่วยความจำแรมสถิตภายใน และรับข้อมูลไบต์ถัดมาจากบัสมาเขียนลงในตำแหน่งที่กำลังถูกระบุในเรจิสเตอร์ตัวชี้ตำแหน่งดังกล่าว และจะทำการเพิ่มค่าในเรจิสเตอร์ตัวชี้ตำแหน่งนี้ครั้งละหนึ่งค่าโดยอัตโนมัติ



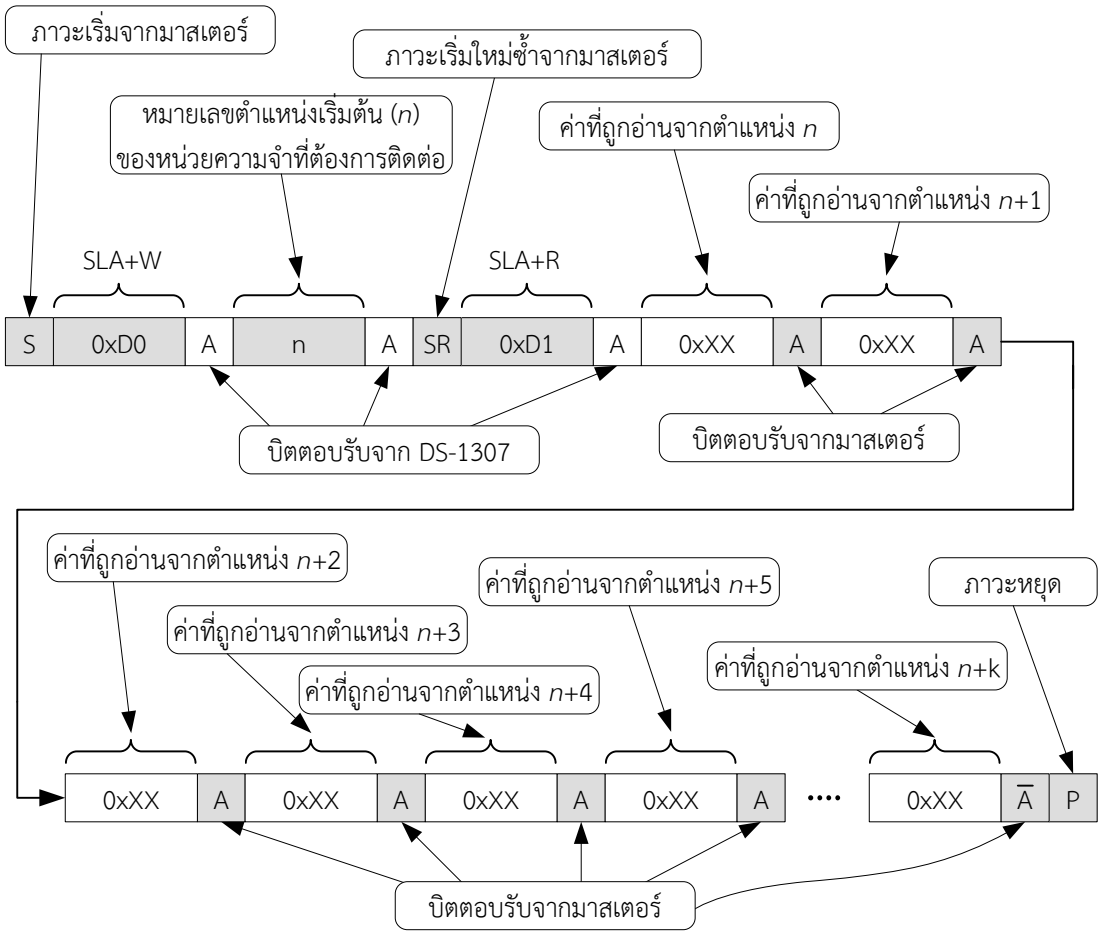
รูปที่ 14.8 การทำงานของไอซี DS1307 ในแบบวิธีสเลฟรีซีฟเวอร์

เมื่อไอซี DS1307 ทำงานในแบบวิธีสเลฟทรานสมิตเตอร์ ตัวประมวลผลในฝั่งมาสเตอร์จะเริ่มต้นด้วยการส่งภาวะเริ่มแล้วตามด้วยหมายเลขที่อยู่ของไอซี DS1307 พร้อมบิตอ่าน (SLA+R) ค่า 0xD1 ดังรูปที่ 14.9 เมื่อไอซี DS1307 ตรวจสอบว่ามีสัญญาณเลขที่อยู่ของตนปรากฏที่บัสก็จะทำการส่งบิตตอบรับ และตามด้วยการอ่านค่าในหน่วยความจำแรมสถิตตำแหน่งที่ถูกเข้าถึงครั้งล่าสุดออกมาสู่บัส เมื่อมีการอ่านหน่วยความจำภายในเสร็จสิ้นหนึ่งตำแหน่ง ก็จะมีการเพิ่มค่าของเรจิสเตอร์ตัวชี้ตำแหน่งภายในไอซี DS1307 โดยอัตโนมัติและจะมีการอ่านค่าในตำแหน่งถัดไปส่งออกมาสู่บัสตราบเท่าที่ยังมีการตอบรับจากมาสเตอร์อยู่ เมื่อมาสเตอร์ต้องการหยุดรับข้อมูลจะต้องทำการส่งบิตตอบรับให้มีค่าเป็นตรรกะต่ำ เพื่อแสดงให้เห็นว่ามาสเตอร์มีความประสงค์ที่จะไม่ตอบรับข้อมูล (Not Acknowledge) อีกต่อไป และตามด้วยการส่งภาวะหยุด



รูปที่ 14.9 การทำงานของไอซี DS1307 ในแบบวิธีสเลฟทรานสมิตเตอร์

จะเห็นว่าการค่าจาก DS1307 ด้วยวิธีการที่แสดงในรูปที่ 14.9 จะเป็นการอ่านหน่วยความจำภายในตัวไอซีต่อจากตำแหน่งที่เคยมีการอ่านครั้งล่าสุด ซึ่งส่งผลให้อาจเกิดข้อยุ่งยากในการใช้งาน เนื่องจากในบางครั้งผู้ใช้อาจมีความประสงค์ที่จะอ่านค่าในตำแหน่งที่ระบุโดยตรงโดยไม่ต้องมีการอ่านผ่านตำแหน่งอื่นที่ไม่ต้องการ ซึ่งหากต้องการทำเช่นนี้ก็สามารถทำได้ดังวิธีอ่านซึ่งแสดงในรูปที่ 14.10



รูปที่ 14.10 การอ่านค่าจากไอซี DS1307 โดยวิธีการระบุตำแหน่งเริ่มต้นของหน่วยความจำ

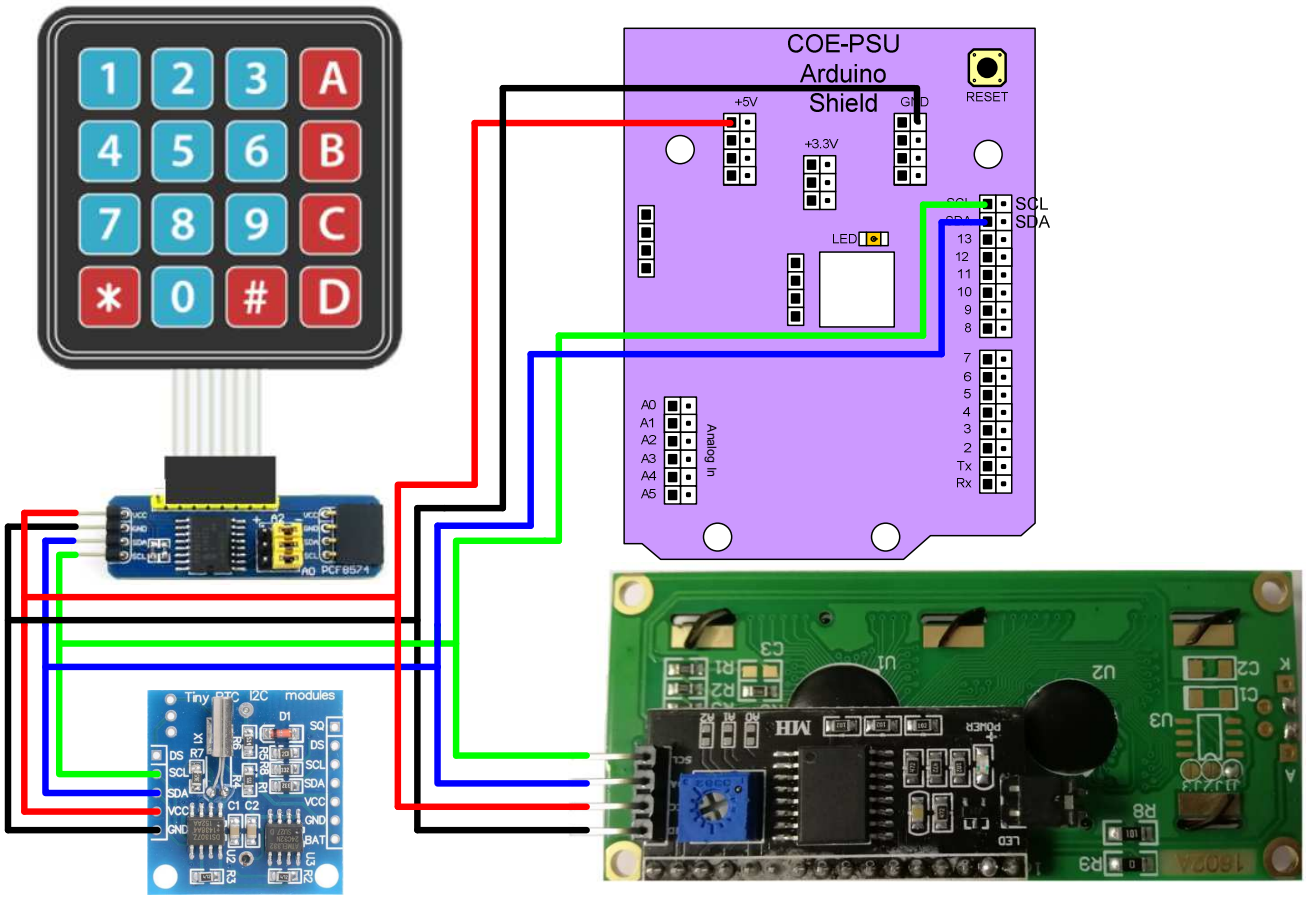
จากรูปที่ 14.10 จะเห็นว่าเพื่อที่จะระบุตำแหน่งหน่วยความจำ มาสเตอร์จะต้องส่งค่าหมายเลขที่อยู่ของไอซี DS1307 พร้อมบิตเขียน (SLA+W) ออกมาก่อน เมื่อไอซี DS1307 ตอบรับแล้วจึงส่งหมายเลขที่อยู่ (ค่า *n*) ของหน่วยความจำที่ต้องการติดต่อ หลังจากที่ไอซี DS1307 ตอบรับค่าหมายเลขที่อยู่นี้แล้วตัวมาสเตอร์จะต้องส่งภาวะเริ่มใหม่ซ้ำ (REPEATED START condition) ลงสู่บัสและตามด้วยหมายเลขที่อยู่ของไอซี DS1307 อีกครั้งแต่คราวนี้จะเป็นการระบุบิตอ่านแทน (SLA+R) ภายหลังจากไอซี DS1307 ตอบรับหมายเลขที่อยู่พร้อมบิตอ่านแล้ว มันก็จะทำการอ่านค่าในแรมสถิตภายในตำแหน่งที่ n ออกมาสู่บัส และเพิ่มค่าตัวชี้ตำแหน่งขึ้นครั้งละหนึ่งค่าและทำการอ่านตำแหน่งถัดไปส่งออกสู่บัส และจะเป็นเช่นนี้ไปเรื่อย ๆ จนกว่ามาสเตอร์จะส่งสัญญาณไม่ตอบรับ

14.3 การคิดคะแนน

การทดลองนี้ประกอบไปด้วย 3 Checkpoint โดยแบ่งคะแนนดังนี้

- Checkpoint 1 35 %
- Checkpoint 2 45 %
- Checkpoint 3 20 %

ก่อนการทำ Checkpoint ให้ทำการทดลองที่ 14.1-14.5 ให้เสร็จสิ้นเสียก่อน เพื่อเตรียมฮาร์ดแวร์ให้พร้อมและศึกษาวิธีการเขียนโปรแกรมติดต่อกับมอดูล DS-1307, มอดูลคีย์แพ็ด และมอดูล LCD ชนิด I2C



รูปที่ 14.11 วงจรสำหรับการทดลองที่ 14.1-14.5 และ Checkpoint 1-2

การทดลองที่ 14.1

ทำการเชื่อมต่อวงจรดังรูปที่ 14.11 จากนำโปรแกรมในรูปที่ 14.12 มาเปิดในหน้าต่าง Editor ของ Arduino IDE ทำการอัพโหลดโปรแกรมลงบนบอร์ด สังเกตการทำงานของฟังก์ชัน I2C_bus_scan เปิดโปรแกรม Serial Monitor จากนั้นสังเกตอุปกรณ์ที่สแกนพบบนบัส I^2C ซึ่งในที่นี่มี 4 รายการ ได้แก่

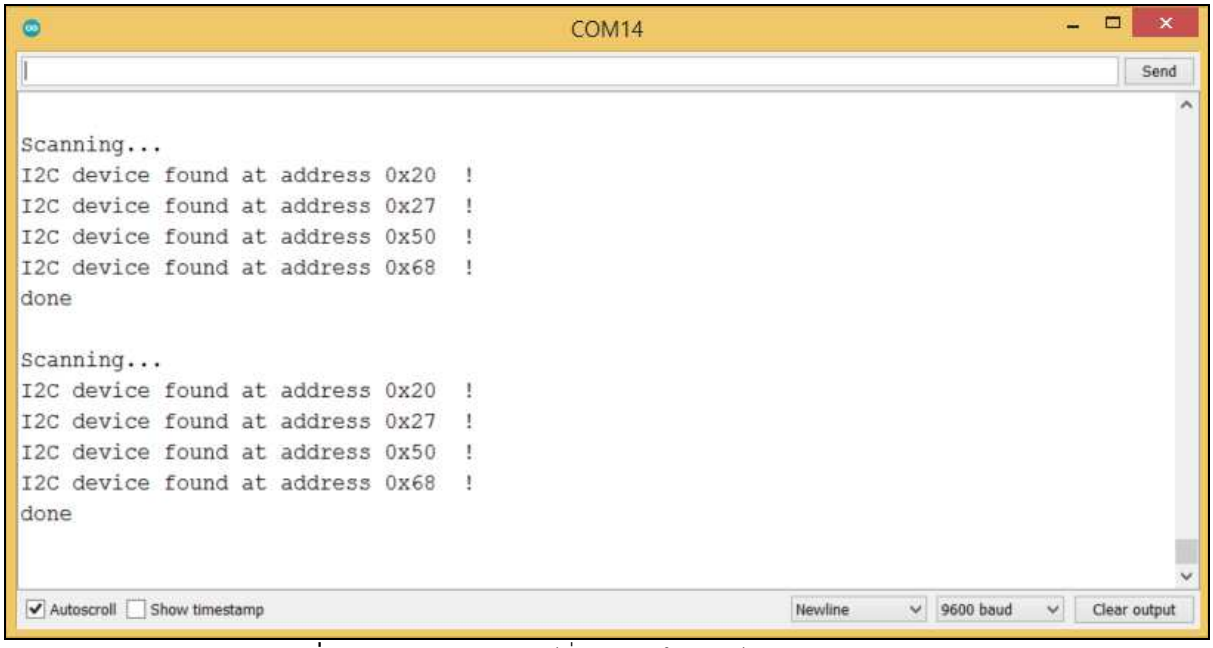
- ตำแหน่ง 0x20 บอร์ด PCF-8574
- ตำแหน่ง 0x27 ของบอร์ด LCD
- ตำแหน่ง 0x50 เป็นของไอซี 24C32 ซึ่งอยู่บนบอร์ด RTC
- ตำแหน่ง 0x68 เป็นของไอซี DS-1307

```
#include <Keypad_I2C.h>
#include <Keypad.h>
#include <Wire.h>

#define I2CADDR 0x20
const byte ROWS = 4;
const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
```

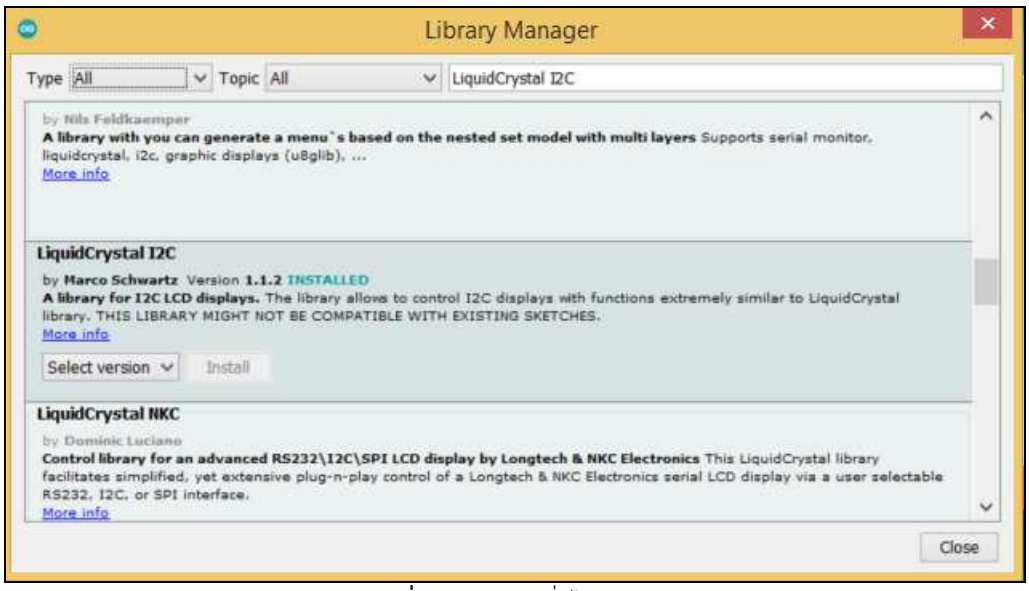
```
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {7, 6, 5, 4};
byte colPins[COLS] = {3, 2, 1, 0};

Keypad_I2C keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS, I2CADDR);
void I2C_bus_scan(void);

void setup(){
  Wire.begin( );
  keypad.begin( );
  Serial.begin(38400);

  I2C_bus_scan();
}

void loop(){
  char key = keypad.getKey();
  if (key != NO_KEY){
    Serial.print(key);
    Serial.println(" is pressed");
  }
}

void I2C_bus_scan(void)
{
  Serial.println ();
  Serial.println ("www.9arduino.com ...");
  Serial.println ("I2C scanner. Scanning ...");
  byte count = 0;

  Wire.begin();
  for (byte i = 8; i < 120; i++)  // Loop ค้นหา Address
  {
    Wire.beginTransmission (i);
    if (Wire.endTransmission () == 0)
      {
      Serial.print ("Found address: ");
      Serial.print (i, DEC);
      Serial.print (" (0x");
      Serial.print (i, HEX);
      Serial.println (")");
      count++;
```

```
      delay (1);
      }
  }
  Serial.println ("Done.");
  Serial.print ("Found ");
  Serial.print (count, DEC);
  Serial.println (" device(s).");
}
```

รูปที่ 14.12 โค้ดสำหรับการทดลองที่ 14.1



รูปที่ 14.13 ตัวอย่างอุปกรณ์ที่สแกนพบโดยฟังก์ชัน I2C_bus_scan



รูปที่ 14.14 การเพิ่มไลบรารี LiquidCrystal I2C

การทดลองที่ 14.2

สร้างโปรเจ็กต์ใหม่ใน Arduino IDE จากนั้นนำโค้ดในรูปที่ 14.15 มาใส่ในหน้าต่าง Editor ของ Arduino IDE ทำการเพิ่มไลบรารี LiquidCrystal I2C ดังรูปที่ 14.14 จากนั้นคอมไพล์โปรแกรม และอัพโหลดโปรแกรมลงบอร์ด Arduino UNO จากนั้นสังเกตการเปลี่ยนแปลงบนหน้าจอแอลซีดี

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);   //Module IIC/I2C Interface บางรุ่นอาจจะใช้ 0x3f

void setup()
{
  lcd.init();
  lcd.backlight();       // เปิด backlight
  lcd.home();
  lcd.print("Hello, world!");
  lcd.setCursor(0, 1);
  lcd.print("CoE PSU");
}

void loop()
{
}
```

รูปที่ 14.15 โค้ดสำหรับการทดลองที่ 14.2

การทดลองที่ 14.3

สร้างโปรเจ็กต์ใหม่ใน Arduino IDE จากนั้นนำโค้ดในรูปที่ 14.16 มาใส่ในหน้าต่าง Editor ของ Arduino IDE ทำการเพิ่มไลบรารี I2CKeypad ดังรูปที่ 14.17 จากนั้นคอมไพล์โปรแกรม และอัพโหลดโปรแกรมลงบอร์ด Arduino UNO จากนั้นสังเกตการเปลี่ยนแปลงบนหน้าจอแอลซีดี ทดลองกดปุ่มบนคีย์แพ็ด จากนั้นสังเกตการแปลี่ยนแปลงบนหน้าจอแอลซีดี

```
#include <LiquidCrystal_I2C.h>
#include <Keypad_I2C.h>
#include <Keypad.h>
#include <Wire.h>

#define I2CADDR 0x20
const byte ROWS = 4;
const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
```

```
byte rowPins[ROWS] = {7, 6, 5, 4};
byte colPins[COLS] = {3, 2, 1, 0};
unsigned long last_time;
bool blink_txt;

LiquidCrystal_I2C lcd(0x27, 16, 2);   //Module IIC/I2C Interface บางรุ่นอาจจะใช้ 0x3f
Keypad_I2C keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS, I2CADDR);


void setup()
{
  blink_txt = false;
  last_time =0;
  Wire.begin();
  keypad.begin();
  lcd.init();
  lcd.backlight();       // เปิด backlight

  lcd.home();
  lcd.print("CoE PSU");
  lcd.setCursor(0, 1);
  lcd.print("Press Keypad...");
}

void loop()
{
  int i;
  char key = keypad.getKey();
  if (key != NO_KEY){
    lcd.setCursor(0,1);
    for (i=0;i<16;i++)
      lcd.print(" ");
    lcd.setCursor(0,1);
    lcd.print(key);
    lcd.print(" key pressed");
  }

  //---ตรวจสอบว่าเวลาผ่านไป 1 วินาทีหรือยัง
  if( millis() - last_time > 1000)
  {
    last_time = millis();
    if(blink_txt)
    {
```

```
      blink_txt = false;
      lcd.setCursor(0,0);
      for(i=0;i<7;i++)
        lcd.print(" ");
    }
    else
    {
      blink_txt = true;
      lcd.setCursor(0,0);
      lcd.print("CoE PSU");
    }
  }
}
```

รูปที่ 14.16 โค้ดสำหรับการทดลองที่ 14.3

รูปที่ 14.17 การเพิ่มไลบรารี I2CKeypad

การทดลองที่ 14.4

สร้างโปรเจ็กต์ใหม่ใน Arduino IDE จากนั้นนำโค้ดในรูปที่ 14.18 มาใส่ในหน้าต่าง Editor ของ Arduino IDE ทำการเพิ่มไลบรารี DS1307 และไลบรารี Time ดังรูปที่ 14.19 จากนั้นคอมไพล์โปรแกรม และอัพโหลดโปรแกรมลงบอร์ด Arduino UNO เปิดโปรแกรม Serial Monitor จากนั้นสังเกตการเปลี่ยนแปลงบนโปรแกรม Serial Monitor

```
#include <LiquidCrystal_I2C.h>
#include <Keypad_I2C.h>
#include <Keypad.h>
#include <Wire.h>
#include <TimeLib.h>
```

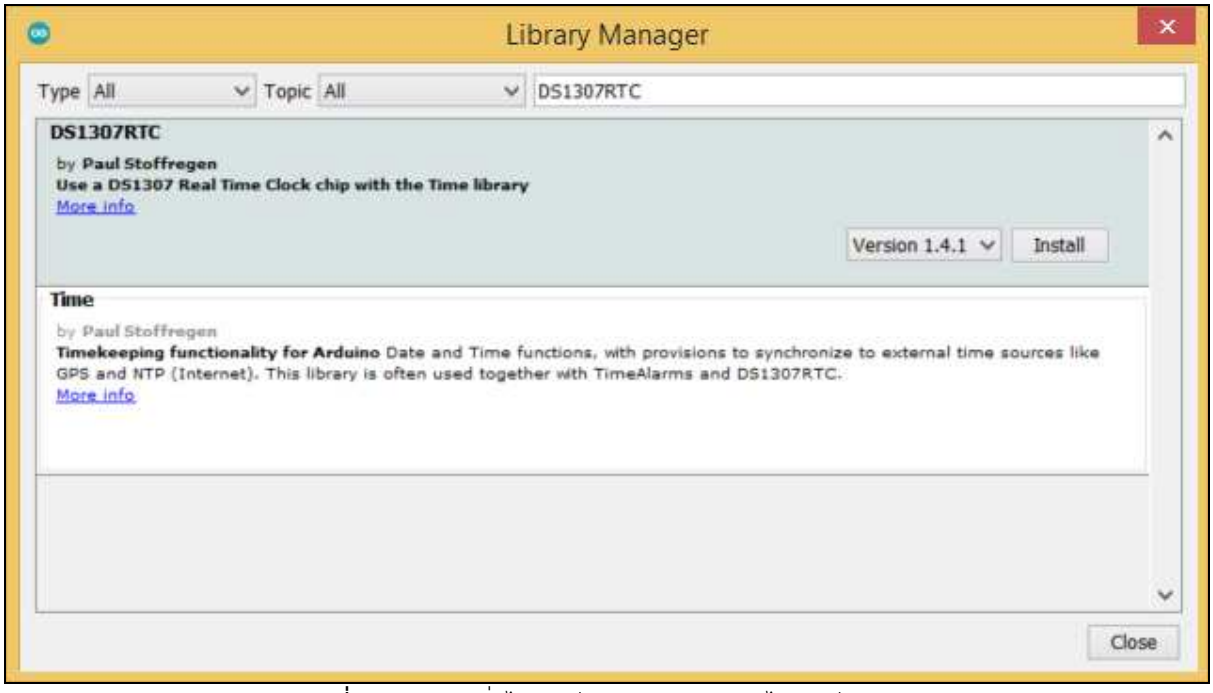
```
#include <DS1307RTC.h>

#define I2CADDR 0x20


const byte ROWS = 4;
const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

byte rowPins[ROWS] = {7, 6, 5, 4};
byte colPins[COLS] = {3, 2, 1, 0};
unsigned long last_time;
bool blink_txt;

LiquidCrystal_I2C lcd(0x27, 16, 2);   //Module IIC/I2C Interface บางรุ่นอาจจะใช้ 0x3f
Keypad_I2C keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS, I2CADDR);
tmElements_t tm;    //for DS1307


void print2digits(int number) {
  if (number >= 0 && number < 10) {
    Serial.write('0');
  }
  Serial.print(number);
}


void setup()
{
  blink_txt = false;
  last_time =0;
  Serial.begin(38400);
  Wire.begin();
  keypad.begin();
  lcd.init();
  lcd.backlight();       // เปิด backlight

  lcd.home();
```

```
  lcd.print("CoE PSU");
  lcd.setCursor(0, 1);
  lcd.print("Press Keypad...");
}

void loop()
{
  int i;
  if (RTC.read(tm)) {
    Serial.print("Ok, Time = ");
    print2digits(tm.Hour);
    Serial.write(':');
    print2digits(tm.Minute);
    Serial.write(':');
    print2digits(tm.Second);
    Serial.print(", Date (D/M/Y) = ");
    Serial.print(tm.Day);
    Serial.write('/');
    Serial.print(tm.Month);
    Serial.write('/');
    Serial.print(tmYearToCalendar(tm.Year));
    Serial.println();
  } else {
    if (RTC.chipPresent()) {
      Serial.println("The DS1307 is stopped.  Please run the SetTime");
      Serial.println("example to initialize the time and begin running.");
      Serial.println();
    } else {
      Serial.println("DS1307 read error!  Please check the circuitry.");
      Serial.println();
    }
    delay(9000);
  }
  delay(1000);
}
```

รูปที่ 14.18 โค้ดสำหรับการทดลองที่ 14.4

การทดลองที่ 14.5

สร้างโปรเจ็กต์ใหม่ใน Arduino IDE จากนั้นนำโค้ดในรูปที่ 14.20 มาใส่ในหน้าต่าง Editor ของ Arduino IDE จากนั้นคอมไพล์โปรแกรม และอัพโหลดโปรแกรมลงบอร์ด Arduino UNO เปิดโปรแกรม Serial Monitor จากนั้นสังเกตการเปลี่ยนแปลงบนโปรแกรม Serial Monitor



รูปที่ 14.19 การเพิ่มไลบรารี DS1307RTC และไลบรารี Time

```
#include <LiquidCrystal_I2C.h>
#include <Keypad_I2C.h>
#include <Keypad.h>
#include <Wire.h>
#include <TimeLib.h>
#include <DS1307RTC.h>

#define I2CADDR 0x20


const byte ROWS = 4;
const byte COLS = 4;

char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

const char *monthName[12] = {
  "Jan", "Feb", "Mar", "Apr", "May", "Jun",
  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
```

```
};


byte rowPins[ROWS] = {7, 6, 5, 4};
byte colPins[COLS] = {3, 2, 1, 0};
unsigned long last_time;
bool blink_txt;

LiquidCrystal_I2C lcd(0x27, 16, 2);   //Module IIC/I2C Interface บางรุ่นอาจจะใช้ 0x3f
Keypad_I2C keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS, I2CADDR);
tmElements_t tm;    //for DS1307

bool getTime(const char *str)
{
  int Hour, Min, Sec;

  if (sscanf(str, "%d:%d:%d", &Hour, &Min, &Sec) != 3) return false;
  tm.Hour = Hour;
  tm.Minute = Min;
  tm.Second = Sec;
  return true;
}

bool getDate(const char *str)
{
  char Month[12];
  int Day, Year;
  uint8_t monthIndex;

  if (sscanf(str, "%s %d %d", Month, &Day, &Year) != 3) return false;
  for (monthIndex = 0; monthIndex < 12; monthIndex++) {
    if (strcmp(Month, monthName[monthIndex]) == 0) break;
  }
  if (monthIndex >= 12) return false;
  tm.Day = Day;
  tm.Month = monthIndex + 1;
  tm.Year = CalendarYrToTm(Year);
  return true;
}


void print2digits(int number) {
  if (number >= 0 && number < 10) {
    Serial.write('0');
```

```
  }
  Serial.print(number);
}


void setup()
{
    bool parse=false;
  bool config=false;

  // get the date and time the compiler was run
  if (getDate(__DATE__) && getTime(__TIME__)) {
    parse = true;
    // and configure the RTC with this info
    if (RTC.write(tm)) {
      config = true;
    }
  }

  Serial.begin(38400);
  while (!Serial) ; // wait for Arduino Serial Monitor
  delay(200);
  if (parse && config) {
    Serial.print("DS1307 configured Time=");
    Serial.print(__TIME__);
    Serial.print(", Date=");
    Serial.println(__DATE__);
  } else if (parse) {
    Serial.println("DS1307 Communication Error :-{");
    Serial.println("Please check your circuitry");
  } else {
    Serial.print("Could not parse info from the compiler, Time=\"");
    Serial.print(__TIME__);
    Serial.print("\", Date=\"");
    Serial.print(__DATE__);
    Serial.println("\"");
  }
}

void loop()
{
}
```

รูปที่ 14.20 โค้ดสำหรับการทดลองที่ 14.5

Checkpoint 1

จงเขียนโปรแกรมเพื่ออ่านค่าเวลาจากบอร์ด DS-1307 และนำมาแสดงผลบนจอแอลซีดีซึ่งเชื่อมต่อกับบอร์ด UNO ผ่านบัส I^2C โดยกำหนดให้มีการแสดงผลดังรูปแบบตามตัวอย่างในรูปที่ 14.21 จากนั้นทดลองปลดไฟเลี้ยงวงจรทิ้งไว้อย่างน้อย 10 วินาที แล้วจ่ายไฟเลี้ยงให้กับวงจรอีกครั้ง สังเกตค่าเวลาที่ขึ้นบนหน้าจออีกครั้ง รอให้เวลาผ่านไป 1 นาทีแล้วสังเกตการเปลี่ยนแปลงของเวลาหลักนาที

รูปที่ 14.21 ตัวอย่างหน้าจอของผลลัพธ์ที่ต้องการใน Checkpoint 1

Checkpoint 2

จงเขียนโปรแกรมเพื่อสร้างนาฬิกาแสดงผลวัน/เดือน/ปี และเวลาในแบบ 24 ชั่วโมง (00.00-23.59) และกำหนดให้มีการใช้ปุ่มบนคีย์แพ็ดในการตั้งค่าเวลา โดยมีปุ่มควบคุม ดังนี้

- ปุ่ม B ใช้กดเพื่อเข้าสู่โหมดการเลื่อนเวลาหลักชั่วโมง และหากกดปุ่ม B ซ้ำอีกครั้งก็จะเลื่อนไปเป็นโหมดการปรับเวลาหลักนาที หากกดปุ่ม B ซ้ำอีก ก็จะสลับไปเป็นการปรับเวลาหลักชั่วโมง
- ปุ่ม C ใช้ในการกดเพื่อเพิ่มค่าเวลา ขึ้นอยู่กับว่าปรับหลักชั่วโมงหรือหลักนาที
- ปุ่ม D ใช้ในการกดเพื่อลดค่าเวลา ขึ้นอยู่กับว่าปรับหลักชั่วโมงหรือหลักนาที
- ปุ่ม # ใช้ในการออกจากโหมดตั้งเวลาและทำการ Save ค่าเวลาใหม่ลงใน DS-1307

หมายเหตุ ค่าเวลาในหลักชั่วโมงที่ตั้งจะต้องอยู่ในช่วง 00-23 และค่าเวลาในหลักนาทีจะต้องอยู่ในช่วง 00-59

ให้แสดงค่าชั่วโมงและนาทีในรูปแบบเลขสองหลักเสมอ ยกตัวอย่างเช่น หากเป็นเวลา 7 โมง หนึ่งนาที ก็ให้แสดงผลเวลาเป็นค่า 07:01

ให้นักศึกษาดูคลิปสาธิตการทำงานของ Checkpoint 2 ได้ที่ลิงก์

https://youtu.be/lfk-i8IjzNM

ในการตรวจ ผู้ตรวจจะให้นักศึกษาลองปรับเวลาใหม่ จากนั้นให้ปลดไฟเลี้ยงออกอย่างน้อย 10 วินาที จากนั้นให้จ่ายไฟเลี้ยงเข้าไปใหม่ และดูว่าค่าเวลาที่ปรับใหม่ยังอยู่หรือไม่เมื่อได้รับไฟเลี้ยงครั้งใหม่

Checkpoint 3

จงนำโปรแกรมใน checkpoint 2 มาเขียนเพิ่มเติมให้มีการรับปุ่ม A สำหรับการปรับค่าวัน/เดือน/ปี กำหนดให้มีการใช้ปุ่มบนคีย์แพ็ดในการตั้งค่าเวลา โดยมีปุ่มควบคุม ดังนี้

- ปุ่ม A ใช้กดเพื่อเข้าสู่โหมดการเลื่อนวันที่ และหากกดปุ่ม A ซ้ำอีกครั้งก็จะเลื่อนไปเป็นโหมดการเลื่อนเดือน หากกดปุ่ม A ซ้ำอีก ก็จะสลับไปเป็นการปรับปี ค.ศ. หากกดปุ่ม A ซ้ำอีก ก็จะเปลี่ยนกลับไปเป็นการเลื่อนวันที่
- ปุ่ม C ใช้ในการกดเพื่อเพิ่มค่า ว/ด/ป ขึ้นอยู่กับว่าปรับค่าใด
- ปุ่ม D ใช้ในการกดเพื่อลดค่า ว/ด/ป ขึ้นอยู่กับว่าปรับค่าใด
- ปุ่ม # ใช้ในการออกจากโหมดตั้ง ว/ด/ป และทำการ Save ค่าวันที่ใหม่ลงใน DS-1307

ให้นักศึกษาดูคลิปสาธิตการทำงานของ Checkpoint 3 ได้ที่ลิงก์

https://youtu.be/dNTH_OmHw30

ในการตรวจ ผู้ตรวจจะให้นักศึกษาลองปรับ ว/ด/ป ใหม่ จากนั้นให้ปลดไฟเลี้ยงออกอย่างน้อย 10 วินาที จากนั้นให้จ่ายไฟเลี้ยงเข้าไปใหม่ และดูว่าค่า ว/ด/ป ที่ปรับใหม่ยังอยู่หรือไม่เมื่อได้รับไฟเลี้ยงครั้งใหม่

หมายเหตุ

โปรแกรมในการทดลองที่ 14.1-14.5 สามารถดาวน์โหลดได้ใน LMS2