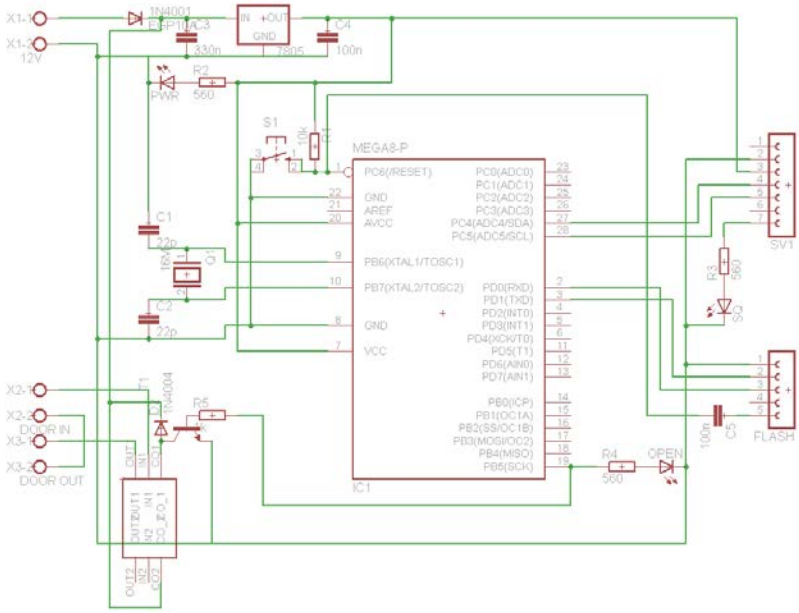
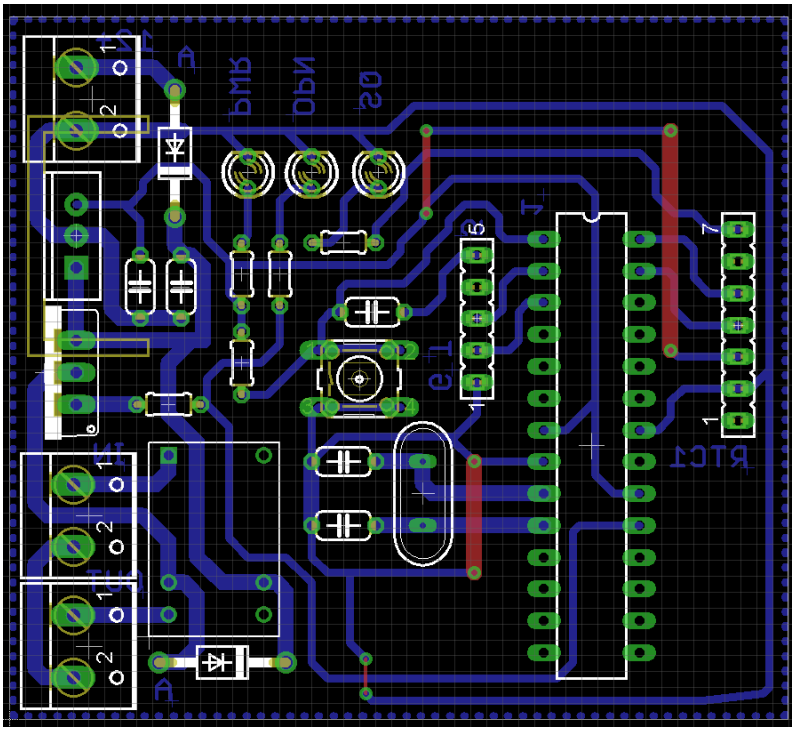
DOOR TIMER MANUAL

Schematic & Board



รูป 1 แผนผังวงจร



รูป 2 ลายวงจร

รายละเอียดอุปกรณ์

R1 = 10kΩ สำหรับ Pull-up ขา Reset

R2, R3, R4 = 560Ω สำหรับ LED แสดงสถานะ

R5 = 1k สำหรับจำกัดกระแสขา BASE ของ Transistor

C1, C2 = 22nF สำหรับสร้างความถี่ให้กับ Auduino

C3 = 330nF สำหรับกรองความถี่ในฝั่ง V source ของบอร์ด

C4, C5 = 100nF C4 สำหรับกรองความถี่ในฝั่ง 5 โวลล์ และ C5 สำหรับ DTR

D1, D2 = 1N4001 D1 ไว้สำหรับป้องกันการต่อไฟกลับขั้ว และ D2 สำหรับป้องกันกระแสย้อนกลับของ Relay

Q1 = 16MHz สำหรับสร้างความถี่ของวงจร

LED PWR = 5โวลล์ แสดงว่า มีกระแสไฟฟ้าเข้าระบบ

LED OPER = 5 โวลล์ แสดงว่า ระบบการใช้ Keycard ทำงาน

LED SQ = 5 โวลล์ แสดงว่า RTC ทำงาน โดยจะกระพริบที่ 1 Hz จาก RTC โดยตรง

Regulator = 7805 แหล่งจ่ายไฟ 5 โวลล์ให้กับวงจรทั้งหมด

Relay = HRS1H-S-DC12V ควบคุมการเปิด-ปิดการใช้ Keycard

Transistor = 2SC1061 ใช้ควบคุม Relay

RTC = DS1307 สร้างเวลาในการควบคุม

การคำนวณ

Power

Relay @12V	200 mW
Arduino Core @5V ,6 mA	30 mW
LED * 3 ea @5V, 20 mA each	300 mW
Pin control Relay @5V, 5 mA	25 mW
Pin I2C @5V, 5 mA	25 mW
RTC DS1307 @5V, 1.5 mA	7.5 mW

Power Consumption

ALL	587.5 mW (50 mA @12V)
12V	200 mW (17mA)
5V	387.5 mW (77.5 mA)

ข้อจำกัดในการจ่ายกระแส

Transistor

Relay 17 mA ค่ากำลังขยายของ 2SC1061 เป็น 35-200 ดังนั้น กระแสขา BASE เป็น 17/35 = 0.5 mA

ในวงจรเราได้จำกัดกระแสที่ไหลเข้าขา BASE ที่ 5 mA ดังนั้น เราสามารถ จ่ายกระแสได้ถึง 170 mA

Regulator

LM7805 จ่ายได้ถึง 1.5 A ในวงจรนี้เราได้ใช้ ที่ 50 mA ในกรณีที่ใช้กระแสเกิน 200 mA จำเป็นที่จะต้องใช้ Heatsink

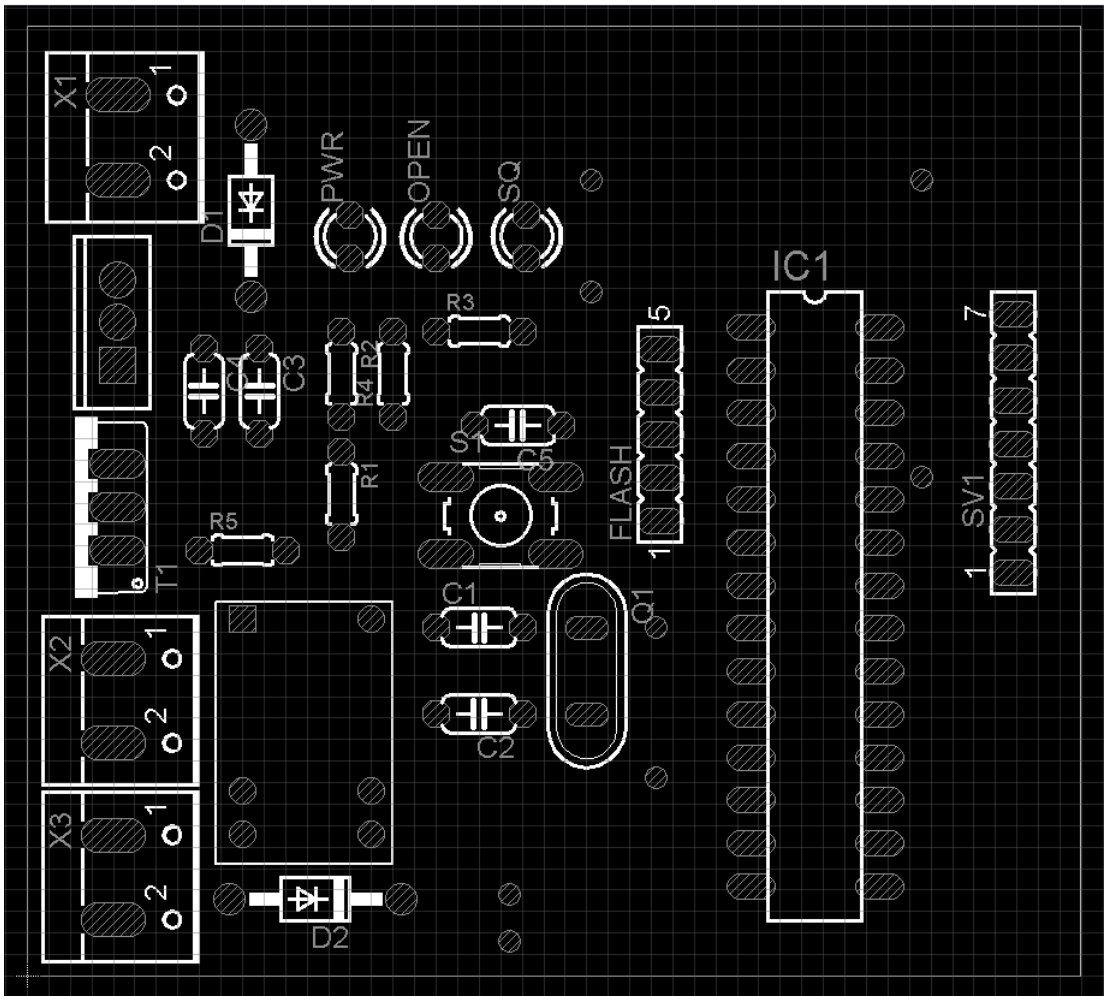
Relay HRS1H-S-DC12V

สามารถควบคุมแรงดันไฟฟ้าได้สูงสุดที่ 220VAC/30VDC

สามารถควบคุมกระแสไฟฟ้าได้สูงสุดที่ 2A

สามารถควบคุมพลังงานไฟฟ้าได้สูงสุดที่ 120VA,30W

Pinout



รูป 3 ผังแสดง **PINOUT**

PINOUT	INPUT/OUTPUT	คุณสมบัติ
X1 (1)	INPUT	รับกระแสไฟฟ้ากระแสตรงจากภายนอก แรงดัน 7-18 โวลล์
X1 (2)	GND	กราวด์
X2 (1)	INPUT	ต่อเข้ากับสายสัญญาณของเครื่อง Key card
X2 (2)	GND	กราวด์
X3 (1)	OUTPUT	ต่อเข้ากับแม่เหล็กล๊อคประตู
X3 (2)	GND	กราวด์
SV1 (1)	NC	ไม่ได้เชื่อมต่อ ซึ่งจะตรงกับขา Vbatt ของ RTC
SV1 (2)	GND	กราวด์
SV1 (3)	VCC	จ่ายไฟ 5 โวลล์ให้กับวงจร RTC
SV1 (4)	I/O	I2C : SDA ไม่มี R Pull-up
SV1 (5)	Output	I2C : SCL ไม่มี R Pull-up
SV1 (6)	NC	ไม่ได้เชื่อมต่อ ซึ่งจะตรงกับขา DS ของ RTC
SV1 (7)	INPUT	สัญญาณไฟกระพริบ 1 Hz จาก RTC
FLASH (1)	GND	กราวด์
FLASH (2)	OUTPUT	สัญญาณ Tx ของ Arduino ระดับลอจิกเป็น TTL 0-5 โวลล์
FLASH (3)	INPUT	สัญญาณ Rx ของ Arduino ระดับลอจิกเป็น TTL 0-5 โวลล์
FLASH (4)	NC	ไม่ได้เชื่อมต่อ
FLASH (5)	INPUT	สัญญาณ DTR จาก คอมพิวเตอร์

คำอธิบาย

Library #include "DS1307.h" สามารถโหลดได้จาก Reference RTC

โปรแกรมนี้ทำหน้าที่ เปิด-ปิด ตัวล๊อคระบบแม่เหล็กของประตู โดยใช้หลักการ Open/Close Circuit ของชุดแม่เหล็ก โดยอ้างอิงเวลามาจากวงจร External RTC เทียบกับกฎที่ได้กำหนดไว้

ในวงจรได้กำหนดขาควบคุมการเปิด-ปิด ด้วย ขาดิจิตอลที่ 13 (หรือขาที่ 19 ของตัวถัง)

ในโปรแกรมได้ออกแบบให้มีการใช้ Watchdog ในการป้องกันความผิดพลาดของการทำงาน โดยตั้งเวลาไว้ที่ 500 มิลลิวินาที ในคำสั่ง wdt_enable(WDTO_500MS);

ในการตั้งกฎเวลาการเปิด-ปิด จะใช้ struct Rule ซึ่งจะประกอบตัว time_nop (เวลาที่ตัดการใช้ตัวล๊อคแม่เหล็ก) และ time_opr (เวลาที่เปิดใช้ตัวล๊อคแม่เหล็ก) ซึ่งได้ประกาศไว้ทั้งหมด 7 ตัว (0-6) ตามจำนวนวันใน 1 สัปดาห์ โดยกำหนดให้ วันอาทิตย์ (rule[0]) และวันเสาร์ rule[6] เปิดการทำงานตลอดทั้งวัน และในวันธรรมดา rule[1-5] ให้ตัดการทำงานในช่วงเวลา 0830 ถึง 1630

ในการอ่านเวลาจาก Exteranl RTC จะใช้ คำสั่ง RTC.get(rtc,true); ซึ่งเวลาเราจะอยู่ในตัวแปร rtc ซึ่งจะมีสมาชิกทั้งหมด 7 ตัวคือ rtc[0](วินาที; 0-59), rtc[1](นาที; 0-59), rtc[2](ชั่วโมง; 0-23), rtc[3](วัน; 1-7, อาทิตย์-เสาร์), rtc[4](วันที่ ;1-31), rtc[5](เดือน; 1-12)และ rtc[6](ปี; 0-99 คือ 2000 ถึง 2099)

การควบคุมทำงานโดยตรวจสอบช่วงเวลาว่าอยู่ในช่วงเวลาที่ตั้งไว้หรือไม่ โดยคำสั่ง if(((rtc[2]*100 + rtc[1]) >= rule[rtc[3]-1].time_nop) && ((rtc[2]*100 + rtc[1]) < rule[rtc[3]-1].time_opr)) ซึ่งจะแบ่งได้เป็น

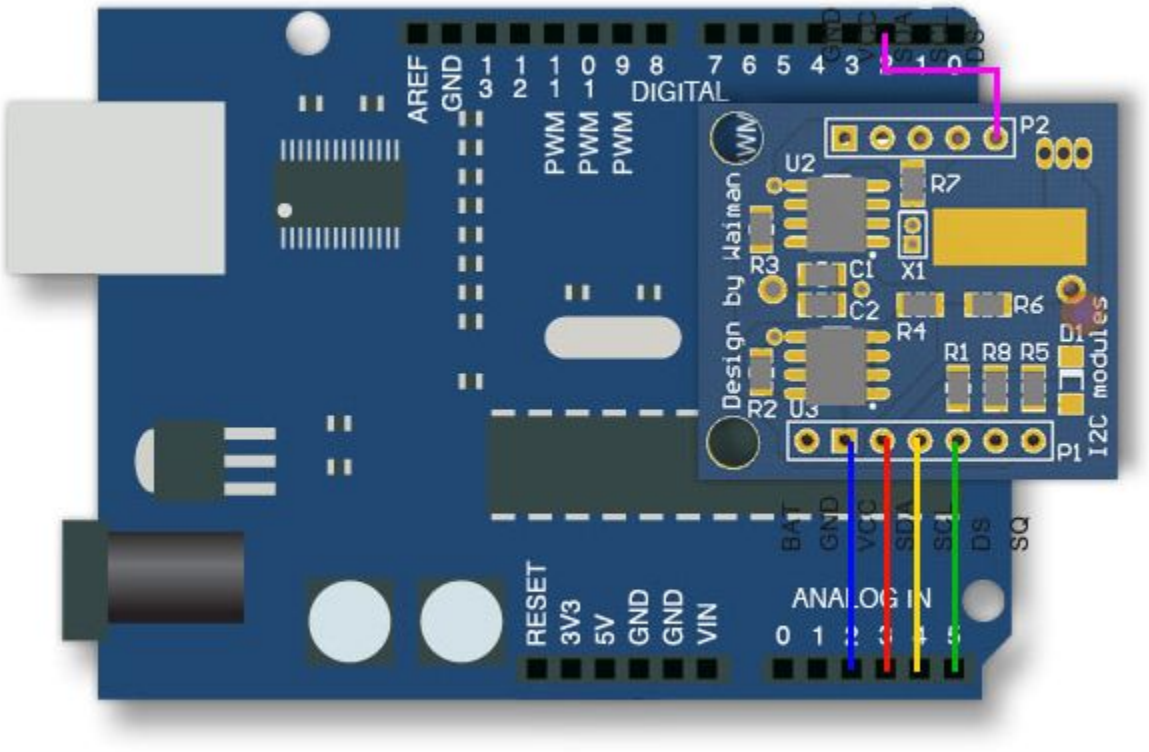
1) การตรวจสอบว่า "อยู่หลังช่วงเวลาที่ตัดการทำงาน" หรือไม่ ด้วยเงื่อนไข (((rtc[2]*100 + rtc[1]) >= rule[rtc[3]-1].time_nop) ซึ่งค่า rtc[3] จะมาจาก External RTC โดยจะเป็นค่า 1-7 (1 คือ วันอาทิตย์ จนถึง 7 คือวันเสาร์) แต่ในกฎของเราได้ตั้ง 0-6 (0 คือ วันอาทิตย์ จนถึง 6 คือวันเสาร์) จึงได้มีการ "ลบ 1"

2) ตรวจสอบว่า "อยู่ในช่วงเวลาที่เปิดการใช้ตัวล๊อค" หรือไม่ ด้วยเงื่อนไข ((rtc[2]*100 + rtc[1]) < rule[rtc[3]-1].time_opr)

ถ้าเหตุการณ์เป็นไปตามเงื่อนไขที่เราได้กำหนดไว้ จะทำการตัดการใช้ตัวล๊อค ด้วยคำสั่ง KeyCardInActive();

และถ้าเหตุการณ์อยู่นอกเหนือจากเงื่อนไขจะทำการใช้ตัวล๊อคแม่เหล็ก ด้วยคำสั่ง KeyCardActive();

การต่อวงจรสำหรับการตั้งเวลา



รูป 4 การต่อวงจรสำหรับตั้งเวลา

ที่มา :

http://www.emartee.com/product/42059/Tiny%20RTC%20DS1307%20Shield%20V2.0%20%20Arduino%20Compatible

คำอธิบาย

ในคำสั่งตั้งเวลานี้ จะกำหนดให้ External RTC มีการกระพริบที่ขา SQ ด้วยความถี่ 1 Hz ในบริเวณของ /*Config Square Wave*/ และตั้งค่าเวลาที่ /*Setup Time*/

และจะแสดงผลเวลาผ่านทาง Terminal

การนำไปใช้งานตั้งเวลา

1. ตั้งเวลาในโปรแกรมล่วงหน้าประมาณ 1 นาที

2. ทำการอัพโหลดโปรแกรมไปยังบอร์ด Arduino Atmega328

3. เชื่อมต่อวงจรดังรูป 4

4. รอจนถึงเวลาที่ได้กำหนดไว้

5. กดปุ่ม RESET บนบอร์ด

6. เปิด Terminal เพื่อตรวจสอบเวลาว่าตรงกับปัจจุบันหรือไม่

Reference

Relay http://www.es.co.th/Schemetic/PDF/HRS1-RELAY.PDF

Transistor http://www.datasheetcatalog.org/datasheet/WINGS/2SC1061.pdf

RTC

http://www.emartee.com/product/42059/Tiny%20RTC%20DS1307%20Shield%20V2.0%20%20Arduino%20Compatible

ภาคผนวก

Code ของการนำไปใช้งาน

CODE

```
#include <WProgram.h>
#include <Wire.h>
#include "DS1307.h"
#include "OneWire.h"
#include <avr/wdt.h>

#define PIN_DOOR_CTRL 13                    //กำหนดให้ขาควบคุมเป็นขาที่ 13
#define STATE_KEY_OPR     HIGH              //กำหนดว่าเมื่อมีการทำงานจ่ายลอจิก HIGH
#define STATE_KEY_NOP     LOW               //กำหนดว่าเมื่อไม่ทำงานจ่ายลอจิก LOW

/*ใน struct Rule ตั้งค่าเวลาเป็นนาฬิกา เช่น เวลา 8 โมง จะต้องตั้งว่า 800 หรือเวลา 8โมง 30 นาทีจะต้องตั้งเป็น 830 เหลือเวลา บ่าย3 30 นาที จะต้องตั้งเป็น 1530 */
typedef struct
{
      unsigned int time_nop; // time_nop คือ เวลาที่ ตัด การทำงานของระบบ Key Card
      unsigned int time_opr; //time_opr คือ เวลาที่ระบบ Key Card ทำงาน ปกติ
}Rule;

void KeyCardInActive(void );    //คำสั่งให้ ตัด ระบบการใช้ Key Card
void KeyCardActive(void );      //คำสั่งให้ เปิด ระบบการใช้ Key Card

int rtc[7];                     //สร้างตัวแปรสำหรับอ่านค่าจาก RTC
Rule rule[7];                   //สร้างตัวแปร กฎ

void setup()
{
      wdt_enable(WDTO_500MS);               //ตั้งระบบ Watchdog ไว้ที่ 500 มิลลิวินาที
      pinMode(PIN_DOOR_CTRL, OUTPUT);  //กำหนดขาควบคุมเป็นแบบ OUTPUT
```

//กำหนดให้วันอาทิตย์ใช้งาน Key Card ***ทั้งวัน***

rule[0].time_nop = 2500;

rule[0].time_opr = 0000;

//กำหนดให้วันจันทร์ ***ตัด*** ระบบ Key Card ในช่วงเวลา 0830 ถึง 1630

rule[1].time_nop = 830;

rule[1].time_opr = 1630;

//กำหนดให้วันอังคาร ***ตัด*** ระบบ Key Card ในช่วงเวลา 0830 ถึง 1630

rule[2].time_nop = 830;

rule[2].time_opr = 1630;

//กำหนดให้วันพุธ ***ตัด*** ระบบ Key Card ในช่วงเวลา 0830 ถึง 1630

rule[3].time_nop = 830;

rule[3].time_opr = 1630;

//กำหนดให้วันพฤหัสบดี ***ตัด*** ระบบ Key Card ในช่วงเวลา 0830 ถึง 1630

rule[4].time_nop = 830;

rule[4].time_opr = 1630;

//กำหนดให้วันศุกร์ ***ตัด*** ระบบ Key Card ในช่วงเวลา 0830 ถึง 1630

rule[5].time_nop = 830;

rule[5].time_opr = 1630;

//กำหนดให้วันเสาร์ใช้งาน Key Card ***ทั้งวัน***

rule[6].time_nop = 2500;

rule[6].time_opr = 0000;

DDRC |= _BV(2) | _BV(3); //กำหนดให้ขา 2 และ 3 เป็นขา Power

PORTC |= _BV(3); // กำหนดให้ขา 3 เป็นขา Vcc

```
	Wire.begin();				//กำหนดให้มีการใช้ I2C
}

void loop()
{
	RTC.get(rtc,true); //อ่านค่าจาก RTC เข้ามาเก็บในตัวแปร rtc

	/*ตรวจสอบช่วงเวลา; TIME_OPEN < TIME < TIME_CLOSE*/
	if(((rtc[2]*100 + rtc[1] ) >= rule[rtc[3]-1].time_nop) && ((rtc[2]*100 + rtc[1] ) < rule[rtc[3]-
1].time_opr)) KeyCardInActive();		//ถ้าอยู่ ใน ช่วงเวลา "ตัดการทำงานของ Key Card"
	/*ในกฎมีทั้งหมด 7 กฎ โดยเริ่มจาก 0-6 (0 คือวันอาทิตย์ จนถึง 6 คือวันเสาร์) แต่ตัวแปร rtc ตัวที่ 3
เก็บวันของสัปดาห์เป็น 1-7 (1 คือวันอาทิตย์ จนถึง 7 คือวันเสาร์) ดังนั้นจึงจำเป็นที่จะต้อง ลบออกด้วย 1
จากวันที่เราอ่านได้เพื่อให้ตรงกับกฎ*/
	else KeyCardActive()};		//ถ้าอยู่ นอก ช่วงเวลา "เปิดการทำงานของ Key Card"
	wdt_reset();				//รีเซตการนับเวลาของ Watchdog
}

void KeyCardInActive(void){ digitalWrite(PIN_DOOR_CTRL, STATE_KEY_NOP);}
void KeyCardActive(void){ digitalWrite(PIN_DOOR_CTRL, STATE_KEY_OPR);}
```

Code ของการตั้งเวลา

CODE

```
#include <WProgram.h>
#include <Wire.h>
#include "DS1307.h"
#include "OneWire.h"
#include <avr/wdt.h>
void setup(){
	DDRC  |= _BV(2) | _BV(3);		//กำหนดให้ขา 2 และ 3 เป็นขา Power
	PORTC |= _BV(3);			// กำหนดให้ขา 3 เป็นขา Vcc

	Serial.begin(9600);			//กำหนดให้ UART มี Baud rate ที่ 9600
```

```
	Wire.begin();				//กำหนดให้มีการใช้ I2C

	/*ตั้งค่าให้ RTC สร้างสัญญาณ 1 Hz ออกทางขา SQ*/
	Wire.beginTransmission(DS1307_CTRL_ID);	//ติดต่อกับ RTC ผ่านทาง I2C
	Wire.send(0x07);				//ติดต่อไปยัง รีจิสเตอร์ CONTROL
	Wire.send(0x10);				//กำหนดให้สร้างสัญญาณ 1Hz
	Wire.endTransmission();				//ส่ง Bit Stop

	/*ตั้งค่าเวลาให้กับ RTC*/
	RTC.stop();				//ตัดการทำงานของ RTC ในกรณีทำงานอยู่
	RTC.set(DS1307_SEC,0);			//กำหนดเวลาในหน่วย วินาที (0-59)
	RTC.set(DS1307_MIN,50);			//กำหนดเวลาในหน่วย นาที (0-59)
	RTC.set(DS1307_HR,13);			//กำหนดเวลาในหน่วย ชั่วโมง (0-23)
	RTC.set(DS1307_DOW,2);			//กำหนดวันของสัปดาห์ โดย 1 คือ วันอาทิตย์,...,6 คือ วันเสาร์
	RTC.set(DS1307_DATE,3);			//กำหนดเวลาในหน่วย วันที่
	RTC.set(DS1307_MTH,9);			//กำหนดเวลาในหน่วย เดือน
	RTC.set(DS1307_YR,12);			//กำหนดเวลาในหน่วย ปี
	RTC.start();				//เริ่มการนับเวลา
}

void loop(){
	RTC.get(rtc,true);			//อ่านค่าเวลาจาก RTC
	/*แสดงเวลาเป็น HH:MM Day Date*/
	Serial.print(rtc[2], DEC);		//แสดง ชั่วโมง
	Serial.print(":");
	Serial.print(rtc[1], DEC);
	Serial.print(" ");
	Serial.print(rtc[3], DEC);
	Serial.print(" ");
	Serial.print(rtc[4], DEC);
	Serial.println(" ");
}
```