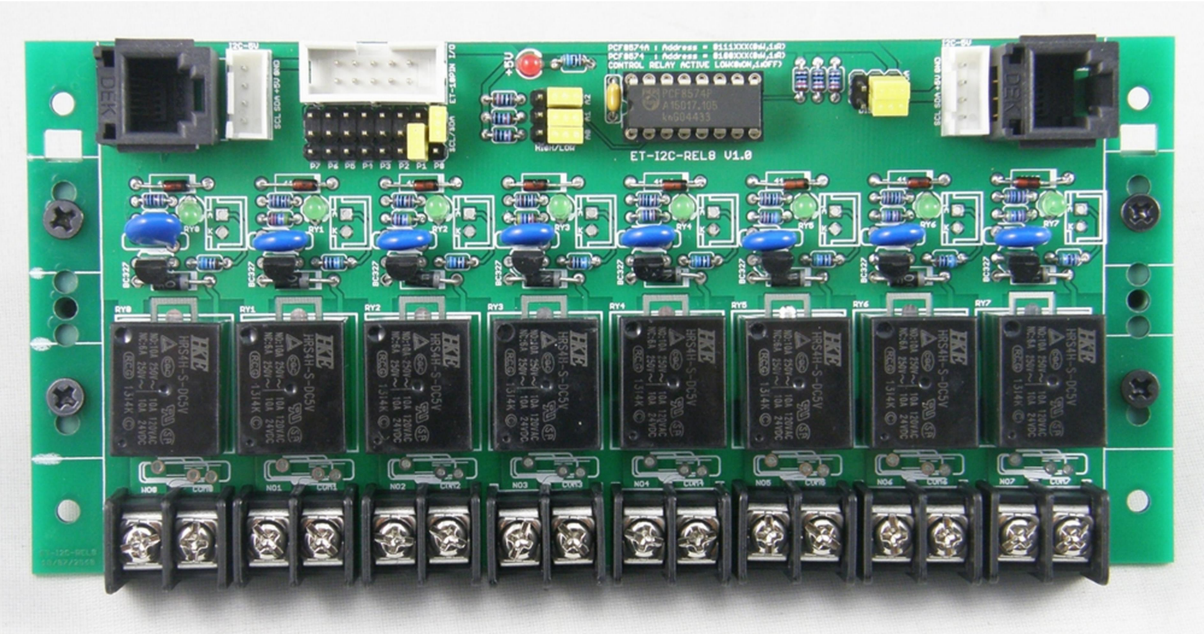
ET-I2C REL8



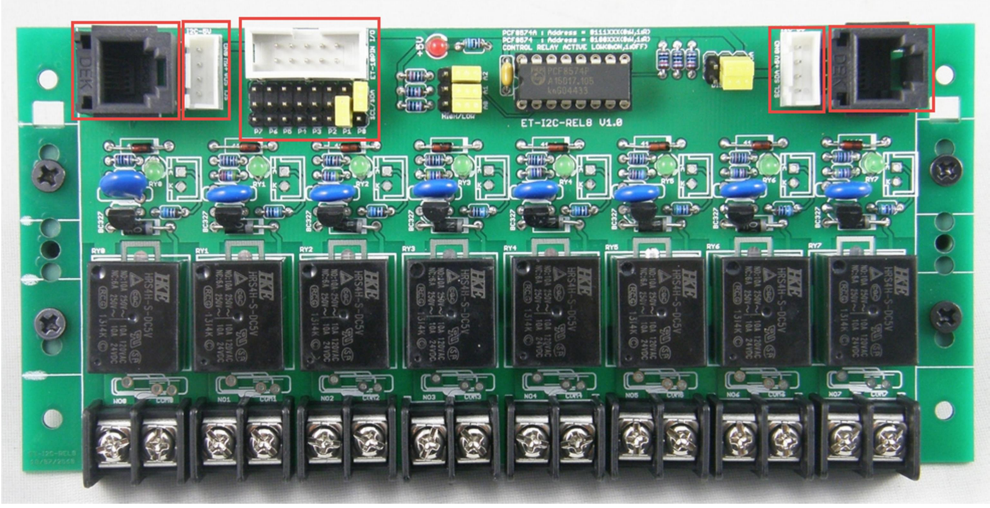
ET-I2C REL8 เป็นบอร์ดขยาย Output Relay ขนาด 8ช่อง ผ่านทาง I2C Bus โดยบอร์ดเลือกใช้ชิพ PCF8574/A เป็นตัวกลางในการสั่งงาน Output Relay จาก MCU ซึ่งสามารถเลือกกำหนดตำแหน่งแอดเดรสการติดต่อของชิพในบอร์ดจาก Jumper ให้มีความแตกต่างกันได้ 8 ตำแหน่ง จึงทำให้สามารถเลือกใช้บอร์ด ET-I2C REL8 ที่ติดตั้งชิพ PCF8574 ร่วมกันในบัสเดียวกันได้มากถึง 8บอร์ด และยังสามารถนำบอร์ด ET-I2C REL8 ที่ทำการติดตั้งชิพ PCF8574A มาต่อร่วมในบัสเดียวกันได้อีก 8บอร์ด รวมเป็น 16บอร์ดได้อีกด้วย

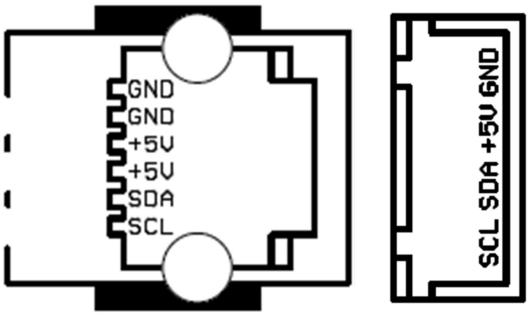
Output RELAY ของบอร์ด ET-I2C REL8 ถูกออกแบบให้ทำงานด้วย Logic LOW และหยุดทำงานของ Relay ด้วย Logic HIGH จึงทำให้ Output RELAY ไม่มีปัญหาทำงานเองในขณะเริ่มต้นจ่ายไฟเลี้ยงระบบในครั้งแรกจึงไม่มีปัญหาเรื่องความผิดพลาดจากอุปกรณ์ต่างๆทำงานเองโดยไม่ได้ตั้งใจในขั้นตอนเปิดเครื่องครั้งแรกก่อนที่ MCU จะหลุดพ้นจากสภาวะรีเซ็ตและเริ่มต้นทำงานเพื่อส่งคำสั่งเข้าไปควบคุม Output ของ PCF8574/A

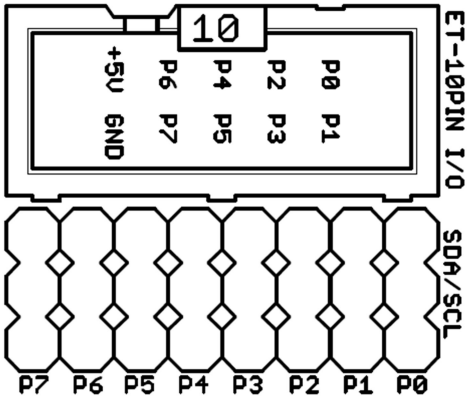
Specification

- Interface I2C Bus 5V
  - 1 x 10PIN IDE Connector
  - 2 x RJ11 6PIN Connector
  - 2 x CPA 4PIN Male Block Connector
- 8 Channel Relay Active LOW Logic
  - 8 x TERMINAL 7.62mm 2PIN Output Contact
  - NO/COMMON Contact Rating 10A/250VAC/24VDC
  - 8 x MOV(Varistor) for Contact Arcing Protection (Option)
- 1 x Internal Red LED +5V Power Indicator
- 8 x Internal Green LED Indicator ON/OFF Status
- 8 x CPA 2PIN For External LED Indicator ON/OFF Status(Option)
- 3 x Jumper For 8 Position I2C Address Setting
- 2 x Jumper For Enable/Disable Pull-Up I2C Signal
- Dimension Size 7.5cm x 16.5cm (3 Inch x 6.5 Inch)
- Din Rail 35mm (Option)

การเชื่อมต่อกับบอร์ด จะสามารถเลือกต่อสัญญาณควบคุม Relay ให้บอร์ดผ่านทางขั้ว I2C ซึ่งมีให้เลือกต่อได้ถึง 3แบบ คือ ขั้วต่อแบบ 10PIN IDE มาตรฐาน ET-10PIN BUS สำหรับใช้เชื่อมต่อกับบอร์ดของ อีทีที ที่มีขั้วสัญญาณเชื่อมต่อเป็นแบบ 10PIN IDE โดยวงจรในส่วนนี้จะมี Jumper สำหรับเลือกกำหนดได้ว่าจะใช้สัญญาณบิตจากขั้ว 10PIN IDE ทำหน้าที่เป็น SCL และ SDA ได้ตามต้องการอีกด้วย นอกจากนี้แล้วยังมีขั้วต่อแบบ RJ11 แบบ 6Pin และขั้วต่อแบบ CPA-4Pin Block ซึ่งมีให้อย่างละ 2 ชุด บนบอร์ด สำหรับรับสัญญาณเข้ามาควบคุมและต่อพ่วงสัญญาณออกไปควบคุมบอร์ดถัดไปในกรณีที่ต้องการจำนวน Input/Output แบบ I2C จำนวนหลายๆบอร์ดร่วมกันได้อีกด้วย



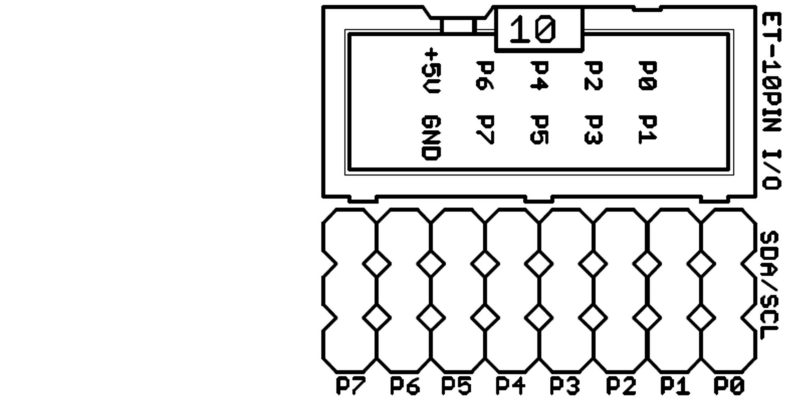


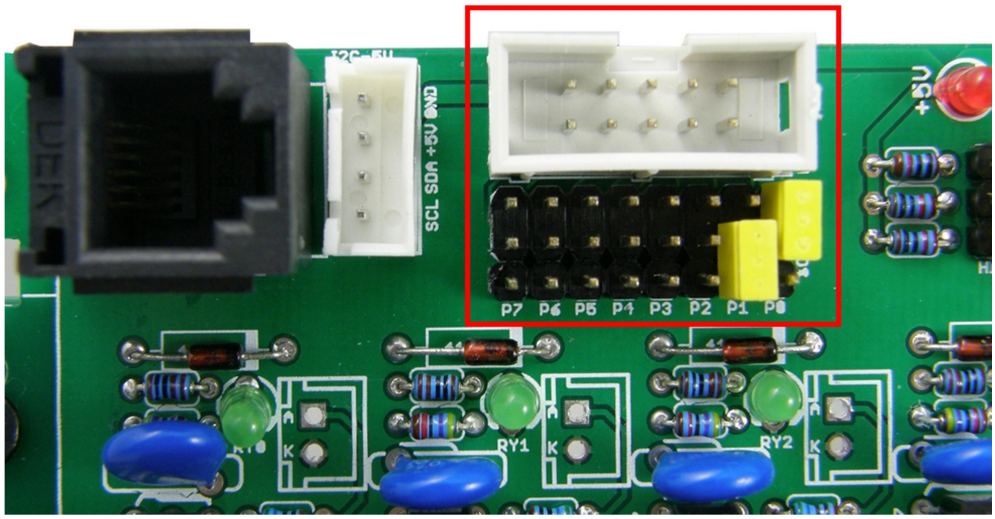


รูปแสดง ลักษณะและตำแหน่งขั้วสัญญาณที่ใช้ในการเชื่อมต่อ I2C Bus

การเชื่อมต่อ I2C Bus ผ่านขั้ว 10PIN IDE

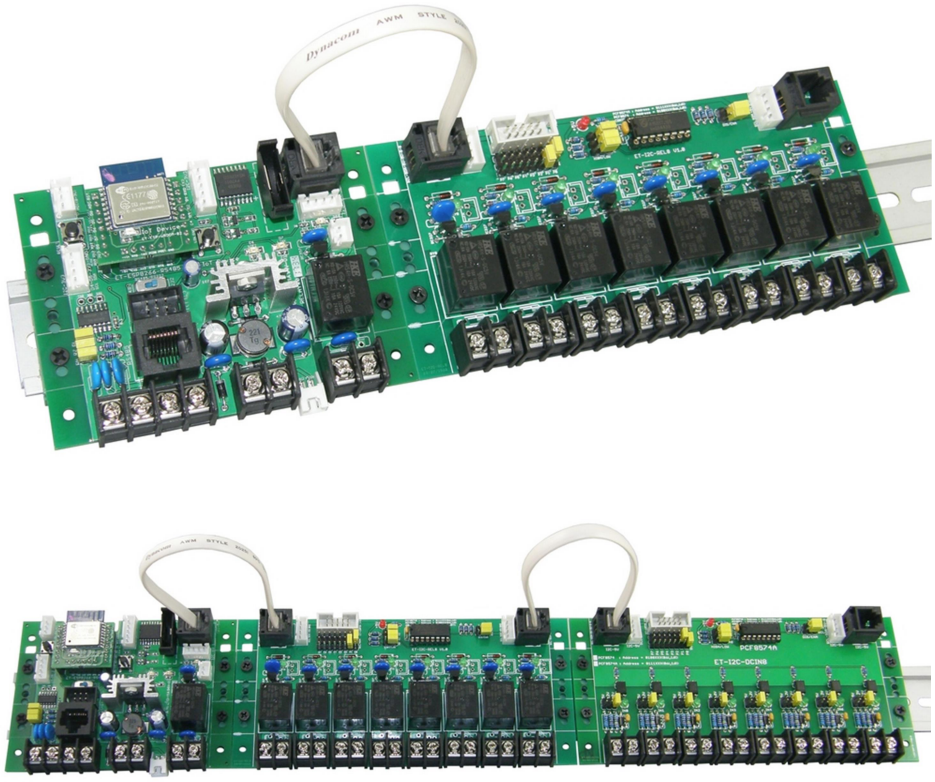
การเชื่อมต่อสัญญาณผ่านขั้ว 10PIN IDE จะกระทำผ่านสายแพร์ขนาด 10PIN ซึ่งในขั้ว 10PIN จะมีแหล่งจ่ายไฟ +5V กับ GND และ สัญญาณ 8เส้น รวมเป็น 10เส้น ซึ่งสัญญาณสามารถเลือกกำหนดได้ว่าต้องการใช้สัญญาณเส้นใด เป็น SCL และ SDA โดยให้ทำการเลือกกำหนด Jumper ของ Pin P0-P7 ที่ต้องการให้ทำหน้าที่เป็น SCL หรือ SDA ของ I2C Bus ได้ตามต้องการ





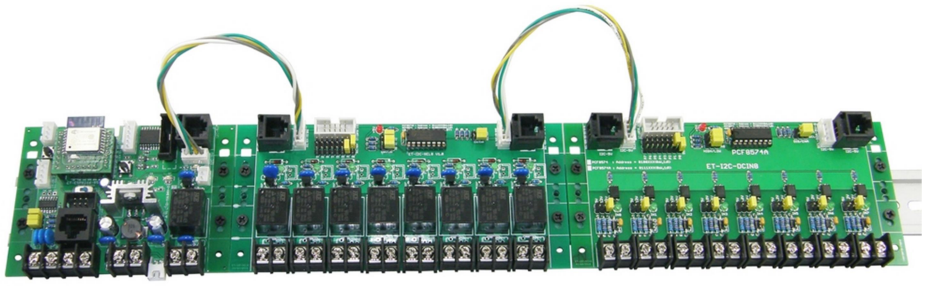
ตัวอย่างการเชื่อมต่อผ่าน 10PIN IDE และกำหนด Jumper ให้ P0 = SDA , P1 = SCL

การเชื่อมต่อ I2C ผ่านขั้วต่อ RJ11 6PIN



ตัวอย่างการต่อบอร์ด ET-I2C REL8 โดยใช้ขั้วต่อแบบ RJ11 6Pin

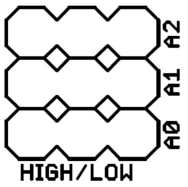
การเชื่อมต่อ I2C ผ่านทางขั้ว CPA 4Pin Block

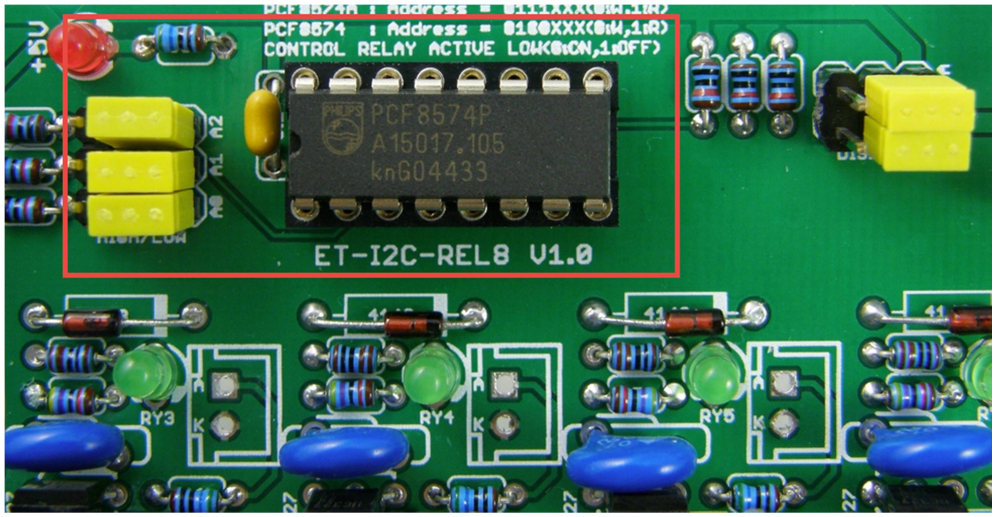


ตัวอย่างการต่อบอร์ด ET-I2C REL8 โดยใช้ขั้วต่อแบบ CPA 4Pin Block

การกำหนด Address I2C

บอร์ด ET-I2C REL8 สามารถเลือกติดตั้งชิพ PCF8574 หรือ PCF8574A สำหรับควบคุมการทำงานของ Relay ได้ แต่ตามปรกติแล้วบอร์ดมาตรฐานจาก อีทีที จะติดตั้งชิพเบอร์ PCF8574 มาให้ ซึ่ง สามารถเลือกกำหนดตำแหน่ง Address การเชื่อมต่อของชิพ ได้จาก Jumper A0,A1 และ A2 ภายในบอร์ดให้มีตำแหน่งการติดต่อสั่งงานที่ไม่ซ้ำกันได้ 8 ตำแหน่ง โดยการเลือกกำหนด Jumper ให้สัญญาณ A0,A1,A2 มีสถานะเป็น LOW หรือ HIGH ที่ไม่ซ้ำกันได้ดังตาราง





การกำหนด Jumper เลือก Address			ตำแหน่งแอดเดรส		
A2	A1	A0	Address	PCF8574	PCF8574A
LOW	LOW	LOW	0	0x40 : 0100 000(0:W)	0x70 : 0111 000(0:W)
LOW	LOW	HIGH	1	0x42 : 0100 001(0:W)	0x72 : 0111 001(0:W)
LOW	HIGH	LOW	2	0x44 : 0100 010(0:W)	0x74 : 0111 010(0:W)
LOW	HIGH	HIGH	3	0x46 : 0100 011(0:W)	0x76 : 0111 011(0:W)
HIGH	LOW	LOW	4	0x48 : 0100 100(0:W)	0x78 : 0111 100(0:W)
HIGH	LOW	HIGH	5	0x4A : 0100 101(0:W)	0x7A : 0111 101(0:W)
HIGH	HIGH	LOW	6	0x4C : 0100 110(0:W)	0x7C : 0111 110(0:W)
HIGH	HIGH	HIGH	7	0x4E: 0100 111(0:W)	0x7E : 0111 111(0:W)

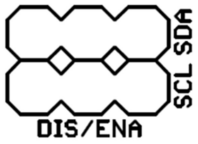
ตารางแสดง ตำแหน่งแอดเดรส I2C Bus ของบอร์ดในกรณีใช้เป็น Control Word ของ I2C

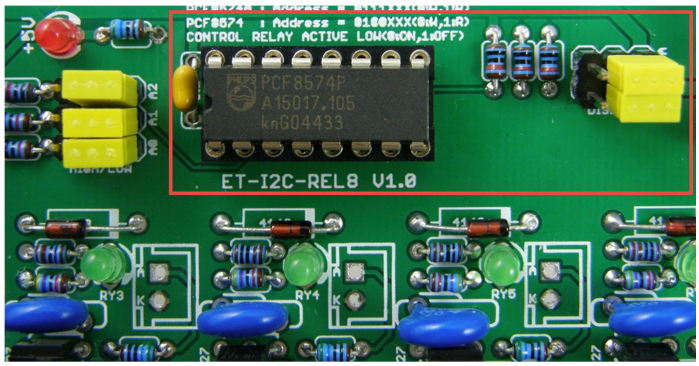
การกำหนด Jumper เลือก Address			ตำแหน่งแอดเดรส		
A2	A1	A0	Address	PCF8574	PCF8574A
LOW	LOW	LOW	0	0x20 : 0010 0000(0:W)	0x38 : 0011 1000(0:W)
LOW	LOW	HIGH	1	0x21 : 0010 0001(0:W)	0x39 : 0011 1001(0:W)
LOW	HIGH	LOW	2	0x22 : 0010 0010(0:W)	0x3A: 0011 1010(0:W)
LOW	HIGH	HIGH	3	0x23 : 0010 0011(0:W)	0x3B : 0011 1011(0:W)
HIGH	LOW	LOW	4	0x24 : 0010 0100(0:W)	0x3C : 0011 1100(0:W)
HIGH	LOW	HIGH	5	0x25 : 0010 0101(0:W)	0x3D : 0011 1101(0:W)
HIGH	HIGH	LOW	6	0x26 : 0010 0110(0:W)	0x3E : 0011 1110(0:W)
HIGH	HIGH	HIGH	7	0x27 : 0010 0111(0:W)	0x3F : 0011 1111(0:W)

ตารางแสดง ตำแหน่งแอดเดรส I2C Bus ของบอร์ดในกรณีใช้กับ Library ของ Arduino

การเลือกกำหนด Pull-Up ให้ I2C Bus

ปรกติแล้วสัญญาณ I2C จะถูกขับผ่าน MOSFET ที่เป็น Open Drain ซึ่งมีความจำเป็นต้องต่อวงจร Pull-Up ให้กับสัญญาณของ I2C Bus ทั้ง 2 เส้น คือ SCL และ SDA ในบัสไว้ด้วยเสมอเพื่อให้สัญญาณมีระดับโลจิกเท่ากับระดับโลจิกของขาสัญญาณ MCU ที่นำมาขับบัส ซึ่งบอร์ด MCU ที่นำมาต่อเพื่อขับบัสนั้น บางวงจรอาจทำการติดตั้งตัวต้านทานสำหรับ Pull-Up ไว้อยู่แล้วภายในบอร์ดของ MCU เอง ซึ่งในกรณีนี้ผู้ใช้ก็ต้องทำการปลดการ Pull-Up ภายในบอร์ดของ ET-I2C REL8 ออกโดยการเลือก Jumper ของ SDA และ SCL ไว้ทางด้าน DIS(Disable) แต่ถ้าบอร์ด MCU ที่เป็นตัวขับบัส ไม่ได้ต่อวงจร Pull-Up ไว้ให้ ผู้ใช้ก็ต้องทำการเชื่อมต่อการ Pull-Up ภายในบอร์ด ET-I2C REL8 ที่จัดเตรียมไว้ในบอร์ด โดยเลือก Jumper ของ SDA และ SCL ไว้ทางด้าน ENA(Enable) ด้วย



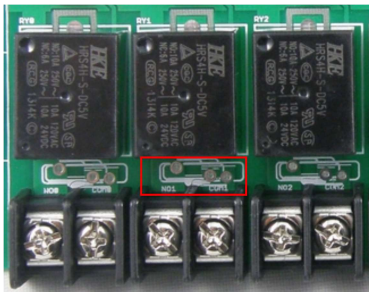


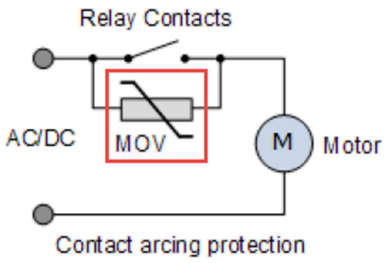
ตัวอย่างการกำหนด Jumper เพื่อ Enable การ Pull-Up สัญญาณ SDA และ SCL

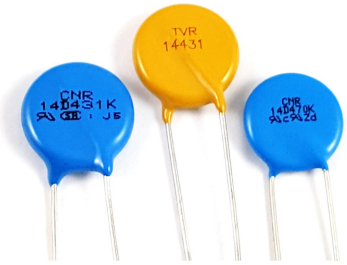
การใช้งาน Output Relay

บอร์ด ET-I2C REL8 1ชุด จะมี Output Relay จำนวน 8ช่อง แต่ละช่องทำงานอิสระต่อกัน โดย Output แต่ละชุดจะมีขั้วต่อแบบ Terminal 7.62mm ขนาด 2Pin เป็นจุดเชื่อมต่อใช้งาน โดยจะเป็นจุดต่อหน้าสัมผัส Relay ชนิด NO หรือ Normal Open โดยหน้าสัมผัสแต่ละชุดสามารถรับกระแสได้สูงสุด 10แอมป์ โดยหน้าสัมผัสจะมีคุณสมบัติเหมือนสวิตช์เปิดปิดอุปกรณ์ไฟฟ้า ซึ่งในสภาวะปรกติตอนที่ Relay ยังไม่ทำงาน หน้าสัมผัสนี้จะยังไม่ต่อถึงกันเหมือนการปิดสวิตช์ แต่เมื่อสั่งให้ Relay ทำงาน หน้าสัมผัสนี้จึงจะต่อเข้าหน้ากันเหมือนการเปิดสวิตช์ ดังนั้นเราจึงสามารถนำหน้าสัมผัสของ Relay นี้ไปใช้ เปิด ปิด อุปกรณ์ไฟฟ้าต่างๆแทนสวิตช์ได้ เพียงแต่หน้าสัมผัส Relay นี้จะมีความพิเศษกว่าหน้าสัมผัสสวิตช์ทั่วๆตรงที่ ไม่ต้องใช้มือกดเพื่อสั่ง เปิด ปิด เอง แต่เราสามารถสั่ง เปิด ปิด สวิตช์นี้ได้จากโปรแกรมโดยกำหนดเงื่อนไขต่างๆได้เอง โดยสามารถสั่ง ON Relay ได้โดยกำหนด Logic Output ของ PCF8574/A ให้เป็น LOW และสั่ง OFF Relay ได้โดยการกำหนด Logic Output ของ PCF8574/A ให้เป็น HIGH

ในกรณีที่นำหน้าสัมผัสรีเลย์ไปใช้เปิดปิดอุปกรณ์ไฟฟ้าที่มีขนาดพิกัดกระแสสูงๆ โดยเฉพาะอุปกรณ์ไฟฟ้าที่เป็นขดลวด เช่น วาล์วไฟฟ้า และ มอเตอร์ ซึ่งอุปกรณ์เหล่านี้จะดึงกระแสผ่านตัวเองในพิกัดที่สูงกว่าปรกติ 2-3เท่าตัว เพื่อใช้ในการสตาร์ทและเริ่มต้นทำงาน ซึ่งในขณะที่ ON และ OFF มักจะเกิดการกระชากอย่างรุนแรงผ่านหน้าสัมผัส ซึ่งจะทำให้เกิดการอาร์คและเกิดสัญญาณรบกวนให้กับอุปกรณ์ไฟฟ้าต่างๆที่ต่อใช้งานร่วมกันอยู่ในระบบไฟฟ้าเดียวกันได้ ซึ่งเราสามารถลดการกระชากป้องกันการอาร์คที่หน้าสัมผัสนี้ได้โดยการติดตั้ง MOV(Varistor) คร่อมเข้าไปที่หน้าสัมผัสได้ โดยที่ใกล้ๆขั้วต่อของหน้าสัมผัสแต่ละชุดของบอร์ด ET-I2C REL8 ผู้ใช้สามารถติดตั้ง MOV สำหรับป้องกันการอาร์คที่หน้าสัมผัสเมื่อสั่ง เปิด ปิด หน้าสัมผัสได้ ซึ่งสามารถเลือกใช้ MOV ขนาดต่างๆให้เหมาะสมกับขนาดและประเภทแรงดันไฟฟ้าทำนำไปใช้งานงานเปิดปิดอุปกรณ์ไฟฟ้าทั้ง กระแสตรง และ กระแสสลับ







รูปแสดง ตำแหน่งและวงจรการติดตั้ง MOV (Varistor) เพื่อป้องกันการอาร์คที่หน้าหน้าสัมผัส Relay

ตัวอย่างโปรแกรม สั่งงานบอร์ด ET-I2C-REL8 **ด้วย** ET-MEGA32U4-RS485

```
#include <Wire.h>                // I2C Bus
#include "pcf8574.h"             // PCF8574/A

//====================================================================
#define SDA_I2C       D2        // PD1
#define SCL_I2C       D3        // PD0
//====================================================================
PCF8574 PCF8574_RELAY_EXP0(0x20);   // PCF8574  = 0100,000+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP1(0x21);   // PCF8574  = 0100,001+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP2(0x22);   // PCF8574  = 0100,010+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP3(0x23);   // PCF8574  = 0100,011+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP4(0x24);   // PCF8574  = 0100,100+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP5(0x25);   // PCF8574  = 0100,101+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP6(0x26);   // PCF8574  = 0100,110+(0:W,1:R)
PCF8574 PCF8574_RELAY_EXP7(0x27);   // PCF8574  = 0100,111+(0:W,1:R)
//====================================================================

//====================================================================
byte relay_external_dev0   = 0xFF;
//====================================================================
const byte RELAY0_ON_MASK   = 0xFE;  // 1111 1110
const byte RELAY1_ON_MASK   = 0xFD;  // 1111 1101
const byte RELAY2_ON_MASK   = 0xFB;  // 1111 1011
const byte RELAY3_ON_MASK   = 0xF7;  // 1111 0111
const byte RELAY4_ON_MASK   = 0xEF;  // 1110 1111
const byte RELAY5_ON_MASK   = 0xDF;  // 1101 1111
const byte RELAY6_ON_MASK   = 0xBF;  // 1011 1111
const byte RELAY7_ON_MASK   = 0x7F;  // 0111 1111
//====================================================================
const byte RELAY0_OFF_MASK  = 0x01;  // 0000 0001
const byte RELAY1_OFF_MASK  = 0x02;  // 0000 0010
```

```
const byte RELAY2_OFF_MASK  = 0x04;  // 0000 0100

const byte RELAY3_OFF_MASK  = 0x08;  // 0000 1000

const byte RELAY4_OFF_MASK  = 0x10;  // 0001 0000

const byte RELAY5_OFF_MASK  = 0x20;  // 0010 0000

const byte RELAY6_OFF_MASK  = 0x40;  // 0100 0000

const byte RELAY7_OFF_MASK  = 0x80;  // 1000 0000

//====================================================================


void setup()

{

 Wire.begin();                 // Initial I2C Bus

 PCF8574_RELAY_EXP0.write8(relay_external_dev0);

}


void loop()

{

 relay_external_dev0 &= RELAY0_ON_MASK;                                  // ON Relay[0]

 PCF8574_RELAY_EXP0.write8(relay_external_dev0);

 delay(1000);


 relay_external_dev0 |= RELAY0_OFF_MASK;                                 //OFF Relay[0]

 PCF8574_RELAY_EXP0.write8(relay_external_dev0);

 delay(1000);

}
```