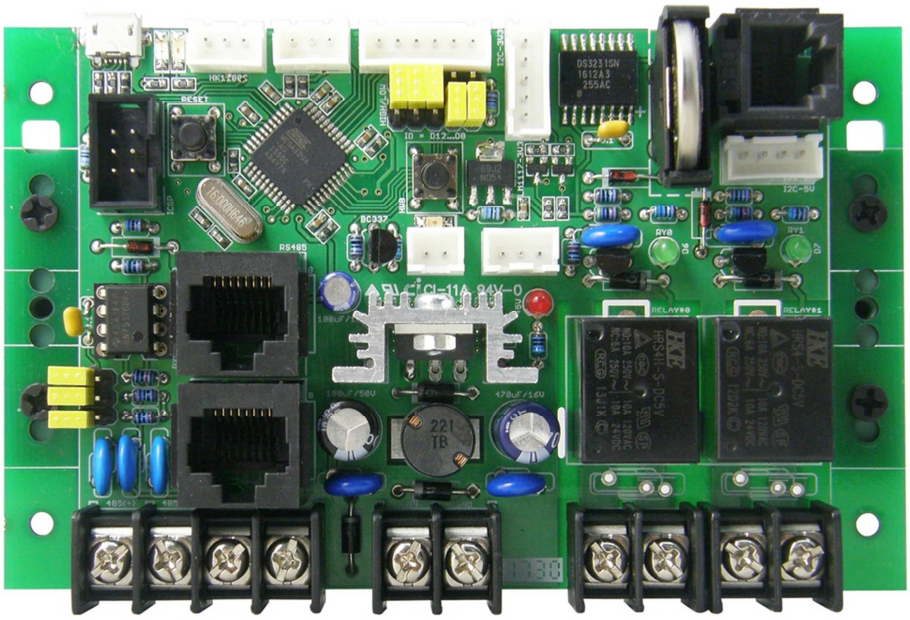
ET-MEGA32U4-RS485



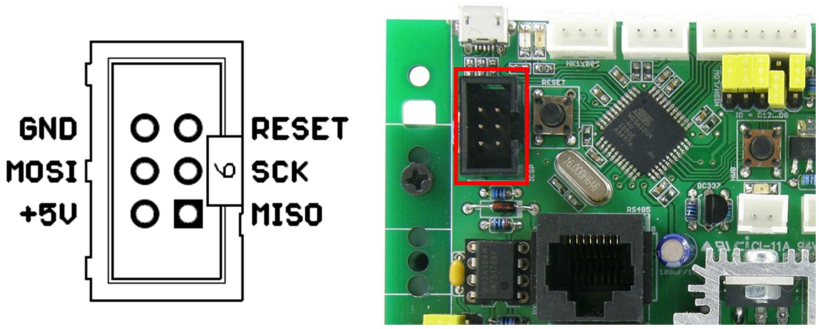
บอร์ด ET-MEGA32U4-RS485 เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR ซึ่งสามารถเลือกพัฒนาโปรแกรมด้วยภาษาต่างๆที่รองรับการทำงานของไมโครคอนโทรลเลอร์ตระกูล AVR เบอร์ ATMEGA32U4 หรือ C++ บนแพลตฟอร์มของ Arduino โดยบอร์ด ET-MEGA32U4-RS485 ถูกออกแบบให้เป็นบอร์ดคอนโทรลขนาดเล็กที่มีองค์ประกอบพื้นฐานครบถ้วนในบอร์ดเดียว เหมาะสมสำหรับนำไปประยุกต์ใช้งานในการควบคุมทั้งแบบที่ทำงานเดี่ยวๆ อิสระ Standalone หรือทำงานร่วมกันโดยการเชื่อมโยงเป็นเครือข่ายผ่านระบบสัญญาณสื่อสารแบบ RS485 Bus

- มีบัสสื่อสาร RS485 2-Wire Half Duplex พร้อมขั้วต่อแบบ Terminal 7.62mm และ RJ45 ขนานกันอย่างละ 2 ชุด เพิ่มความสะดวกในการต่อใช้งานเป็นเครือข่ายพ่วงกันแบบบัสได้โดยสะดวก ซึ่งบอร์ดออกแบบให้สามารถต่อร่วมกันในบัสได้มากถึง 32จุด โดยมี Jumper จำนวน 5บิตสำหรับเลือกกำหนดตำแหน่งแอดเดรสให้แต่ละบอร์ดมีตำแหน่งแอดเดรสไม่ซ้ำกัน และสามารถต่อได้ไกลเป็นระยะทางรวมกันได้ถึง 1200เมตร
- มีแหล่งจ่ายไฟแบบ Switching Regulate ขนาด 5V/1A รองรับแรงดัน Input ตั้งแต่ 7-30V พร้อมแผ่นระบายความร้อนทำให้สามารถใช้งานต่อเนื่องกันเป็นเวลานานๆได้อย่างไม่มีปัญหา
- มี Output RELAY ขนาด 10A จำนวน 2ชุด สำหรับใช้ทำหน้าที่เป็นสวิตช์ เปิด ปิด อุปกรณ์ไฟฟ้าต่างๆ ผ่านหน้าสัมผัสแบบ NO/COMMON คือหน้าสัมผัสเชื่อมต่อถึงกันเมื่อสั่ง ON RELAY พร้อมวงจรลดสัญญาณรบกวนที่เกิดจากการอาร์คของกระแสกระชากในขณะหน้าสัมผัสรีเลย์ตัดและต่อในกรณีนำหน้าสัมผัสไปใช้สั่งงานเปิดปิดอุปกรณ์จำพวกขดลวดเช่น มอเตอร์ โซลินนอยด์วาล์ว แมกเนติกซ์ ฯลฯ
- มี RTC(Real Time Clock) เบอร์ DS3231 พร้อมแบตเตอรี่ Backup

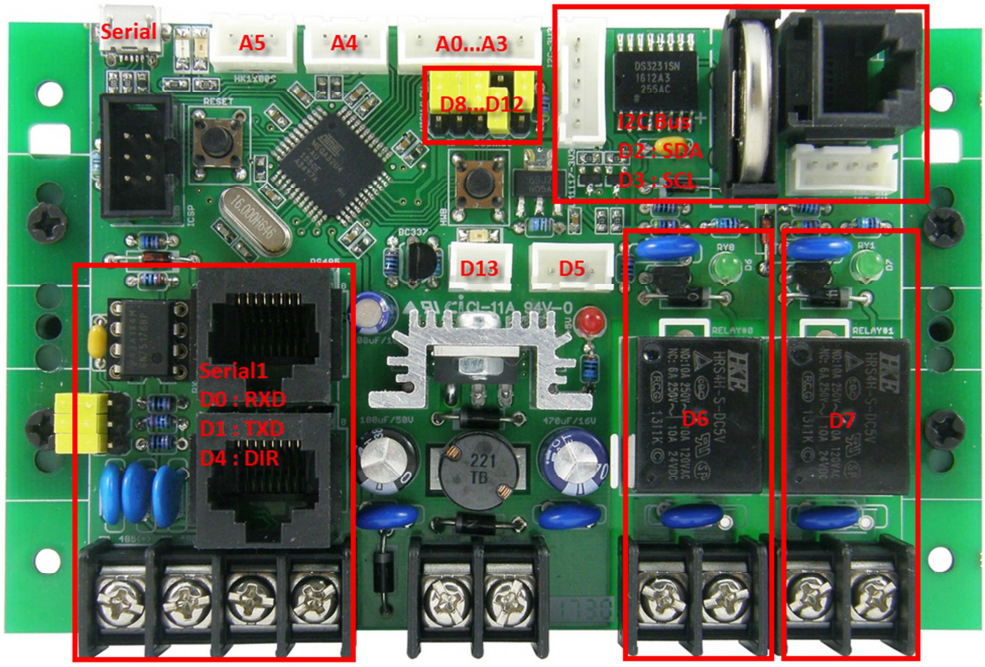
- มีบัสสื่อสาร I2C Bus ทั้งแบบใช้งานกับอุปกรณ์ที่เป็น 3V และ 5V สำหรับขยายอุปกรณ์ Input / Output แบบต่างๆผ่าน I2C Bus หรือ เชื่อมต่อกับอุปกรณ์เซ็นเซอร์ต่างๆที่เป็น I2C Bus
- มีขั้วต่อ I/O บัสแบบ Digital ขนาด 1บิต โดยใช้ขั้วแบบ 3 Pin สำหรับประยุกต์ใช้งานร่วมกับ Input/Output แบบต่างๆ เช่น Input สวิตช์ หรือ 1-Wire Sensor ต่างๆเช่น เซ็นเซอร์วัดอุณหภูมิเบอร์ DS18B20
- มีขั้วต่อสัญญาณแบบ Analog (A0-A3) โดยใช้งานเป็น Analog ADC จำนวน 4ช่อง หรือ ใช้เป็น Digital I/O จำนวน 4บิต ได้ตามต้องการ โดยใช้ขั้วต่อแบบ 6Pin
- มีขั้วต่อสัญญาณแบบ Analog A4 และ A5 โดยใช้งานเป็น Analog ADC ขนาด 1ช่อง จำนวน 2ชุด หรือ ใช้เป็น Digital I/O ขนาด 1บิต จำนวน 2ชุด ได้ตามต้องการ โดยใช้ขั้วต่อแบบ 3Pin
- มีขั้วต่อ ISP แบบ 6PIN มาตรฐาน ATMEL สำหรับใช้โปรแกรมผ่านเครื่องโปรแกรมแบบ ISP ตามมาตรฐาน ATMEL
- มีขั้วต่อ Micro USB สำหรับใช้ทำหน้าที่เป็น USART Download ผ่าน Bootloader โดยใช้แพลตฟอร์มการพัฒนาโปรแกรมแบบ Arduino
- รองรับการติดตั้งใช้งานบนราง DIN ขนาด 35มม.

การพัฒนาโปรแกรม ET-MEGA32U4 RS485

ในการพัฒนาโปรแกรมของบอร์ด ET-MEGA32U4 RS485 นั้น ถ้าผู้ใช้ต้องการพัฒนาโปรแกรมด้วยภาษาซี ของ AVR อย่าง WinAVR ผู้ใช้สามารถทำได้ตามปรกติโดยใช้งานร่วมกับเครื่องโปรแกรมตระกูล AVR ที่รองรับการโปรแกรม MCU เบอร์ ATMEGA32U4 แบบ Incircuit Serial Programmer ผ่านทางขั้วโปรแกรมแบบ AVRISP ขนาด 6PIN ตามมาตรฐานของ ATMEL



แต่สำหรับผู้ใช้ที่ต้องการพัฒนาโปรแกรมของบอร์ดด้วยแพล็ตฟอร์มแบบ Arduino ก็สามารถทำได้ทันทีโดยไม่จำเป็นต้องใช้เครื่องมือโปรแกรมใดๆจากภายนอกอีก ซึ่งบอร์ด ET-MAGA32U4 RS485 ชุดมาตรฐาน จาก อีทีที จะทำการบรรจุโปรแกรม Bootloader ของ Arduino Leonardo เตรียมไว้ให้เรียบร้อยแล้วผู้ใช้สามารถใช้การพัฒนาโปรแกรมของบอร์ดด้วยแพลตฟอร์ม Arduino ได้ทันทีโดยกำหนด Hardware ในการพัฒนาบนแพล็ตฟอร์ม Arduino เป็น Arduino Leonardo แล้ว Upload โปรแกรมผ่านทาง USB Serial ของบอร์ดได้ทันที ซึ่งจะทำให้อุปกรณ์ต่างๆบนบอร์ดมีสัญญาณการควบคุมและสั่งงานเป็นดังนี้

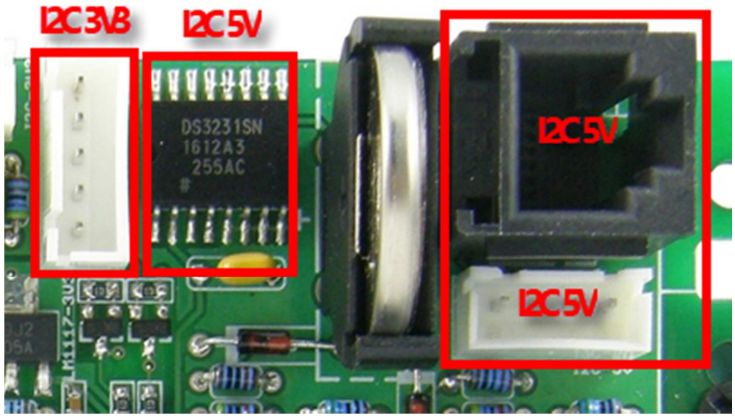


- RELAY#0 ใช้ Pin D6 เป็น Digital Output (LOW = ON RELAY, HIGH = OFF RELAY)
- RELAY#1 ใช้ Pin D7 เป็น Digital Output (LOW = ON RELAY, HIGH = OFF RELAY)
- LED Status ใช้ Pin D13 เป็น Digital Output (LOW = OFF LED, HIGH = ON LED)
- 1-Wire ใช้ Pin D5 เป็น 1-Wire Bus
- RS485 ใช้ Serial1
  - RXD ใช้ Pin D0 (Default Serial1 RXD)
  - TXD ใช้ Pin D1 (Default Serial1 TXD)
  - DIR ใช้ Pin D4 เป็น Digital Output (LOW = Receive RS485, HIGH = Send RS485)
- RS485 Slave Address
  - ID0 ใช้ Pin D8 เป็น Digital Input Pull Up (กำหนด pinMode = INPUT_PULLUP)
  - ID1 ใช้ Pin D9 เป็น Digital Input Pull Up (กำหนด pinMode = INPUT_PULLUP)
  - ID2 ใช้ Pin D10 เป็น Digital Input Pull Up (กำหนด pinMode = INPUT_PULLUP)
  - ID3 ใช้ Pin D11 เป็น Digital Input Pull Up (กำหนด pinMode = INPUT_PULLUP)
  - ID4 ใช้ Pin D12 เป็น Digital Input Pull Up (กำหนด pinMode = INPUT_PULLUP)
- I2C Bus ใช้ I2C Bus Default
  - SDA ใช้ Pin D2 เป็นสัญญาณในการเชื่อมต่อ
  - SCL ใช้ Pin D3 เป็นสัญญาณในการเชื่อมต่อ
- Analog#0 ใช้ Pin A0,A1,A2,A3 ในการเชื่อมต่อ
- Analog#1 ใช้ Pin A4 ในการเชื่อมต่อ
- Analog#2 ใช้ Pin A5 ในการเชื่อมต่อ
- Debug Serial(USB Serial) ใช้ Serial

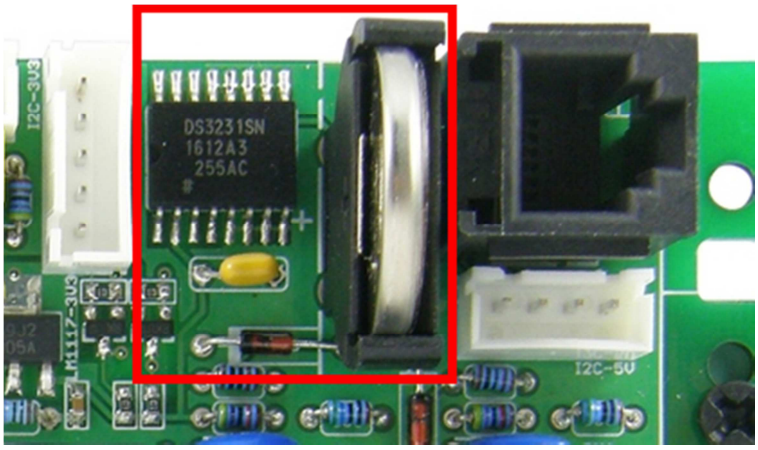
การใช้งาน **I2C Bus**

วงจรในส่วนของการเชื่อมต่อกับอุปกรณ์แบบ I2C Bus ของบอร์ด ET-MAGA32U4 RS485 จะใช้ขาสัญญาณ PD0(Arduino : D3) ทำหน้าที่เป็น SCL และใช้ PD1(Arduino : D2) ทำหน้าที่เป็นสัญญาณ SDA โดยภายในบอร์ดจะติดตั้งชิพ RTC เบอร์ DS3231 ซึ่งเป็น RTC แบบ I2C ทำงานที่ระดับ Logic TTL 5V และยังจัดให้มีขั้วต่อแบบ RJ11 6PIN กับ CPA 4PIN Block อย่างละชุด สำหรับทำหน้าที่เชื่อต่อกับอุปกรณ์ I2C Bus ที่เป็นสัญญาณ Logic TTL 5V จากภายนอกได้ตามต้องการอีกด้วย

สำหรับอุปกรณ์ที่เป็น I2C Bus แต่มีระดับสัญญาณเป็น Logic 3.3V นั้น บอร์ด ET-MEGA32U4 RS485 เองก็ได้จัดทำวงจรแปลงระดับสัญญาณ Logic จาก 5V เป็น 3.3V ของ I2C Bus จัดเตรียมไว้ให้ผู้ใช้ได้นำไปเชื่อมต่อประยุกต์ใช้งานได้ตามต้องการผ่านทางขั้วต่อแบบ CPA 5PIN Block ดังรูป

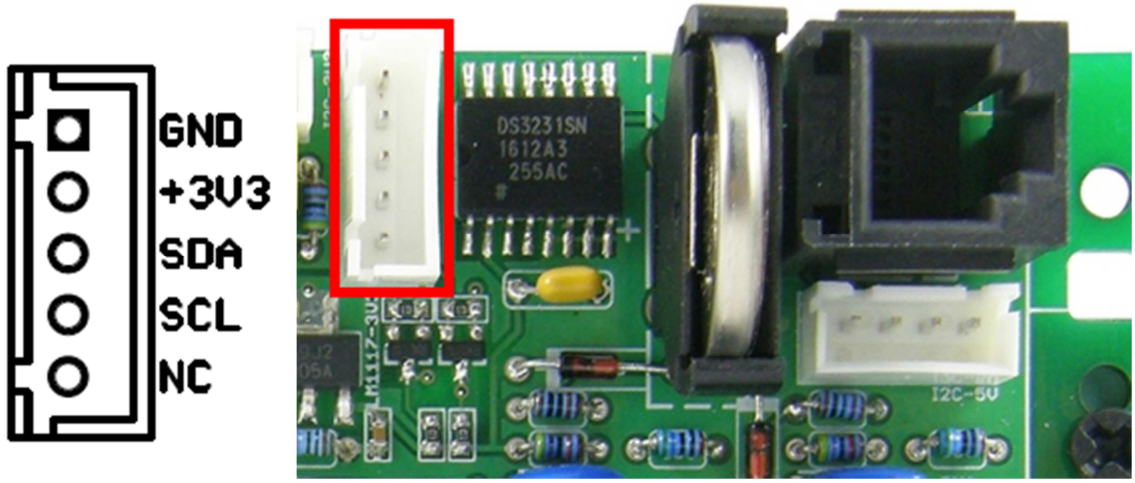


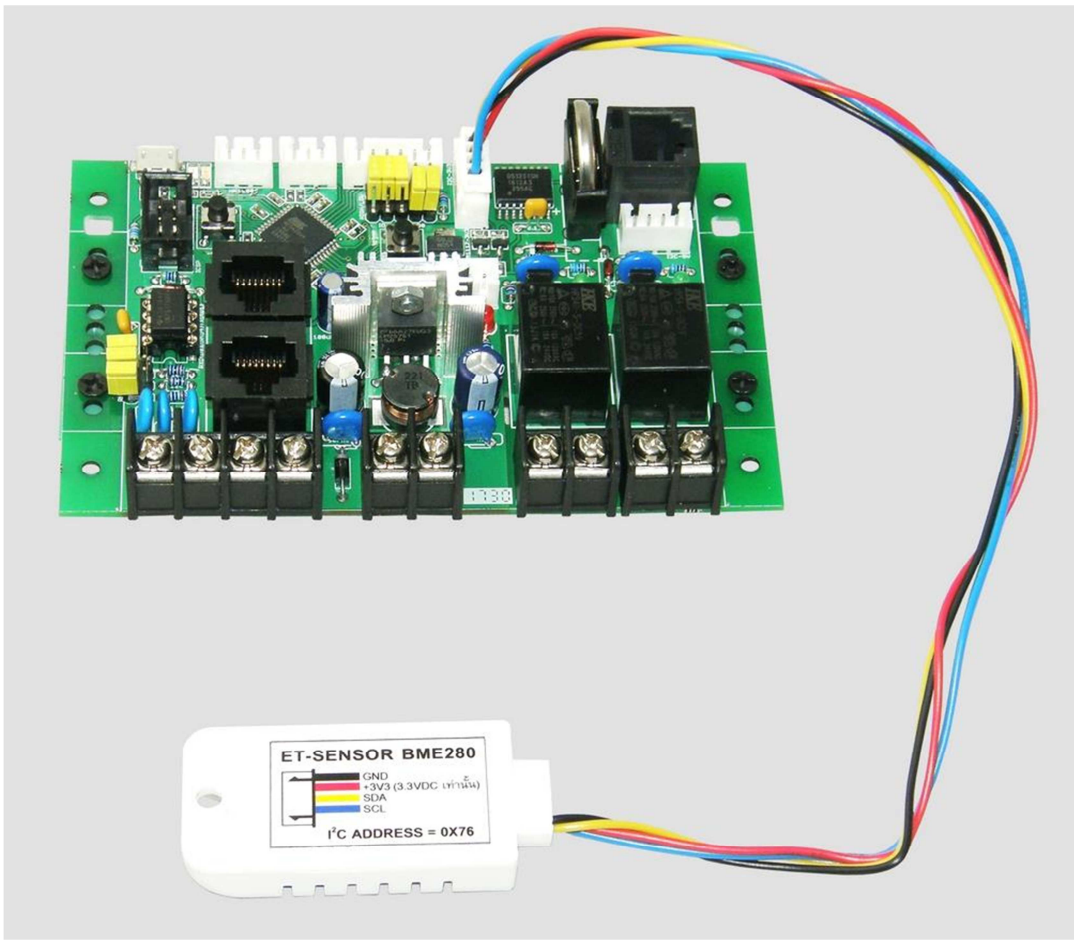
I2C RTC เบอร์ DS3231 เป็น Real Time Clock นาฬิกา สำหรับประยุกต์ใช้กับงานควบคุมต่างๆ เช่น ตั้งเวลาเปิด ปิด อุปกรณ์ไฟฟ้า โดยเชื่อมต่อกับ MCU ผ่านทาง I2C Bus



I2C Bus 3.3V

เป็นขั้ว I2C Bus แบบ CPA 5PIN Block ใช้สำหรับเชื่อมต่อกับอุปกรณ์ที่เชื่อมต่อผ่าน I2C Bus แบบที่รองรับการใช้งานกับแหล่งจ่ายไฟและระดับสัญญาณโลจิกในการเชื่อมต่อเป็น 3.3V เช่น Sensor ตรวจอากาศ ET-SENSOR BME280 หรือ ET-SENSOR SHT15 หรือ ET-SENSOR SHT31 หรือ ET-SENSOR AM2302 เป็นต้น

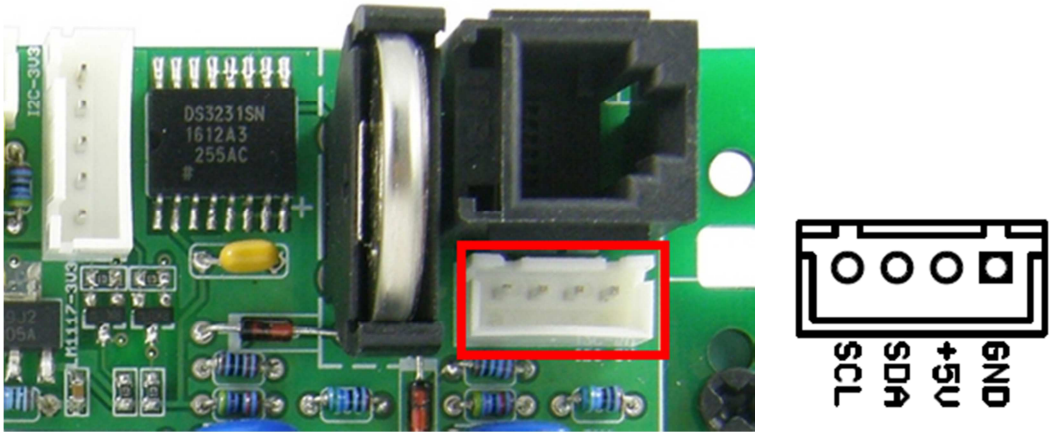


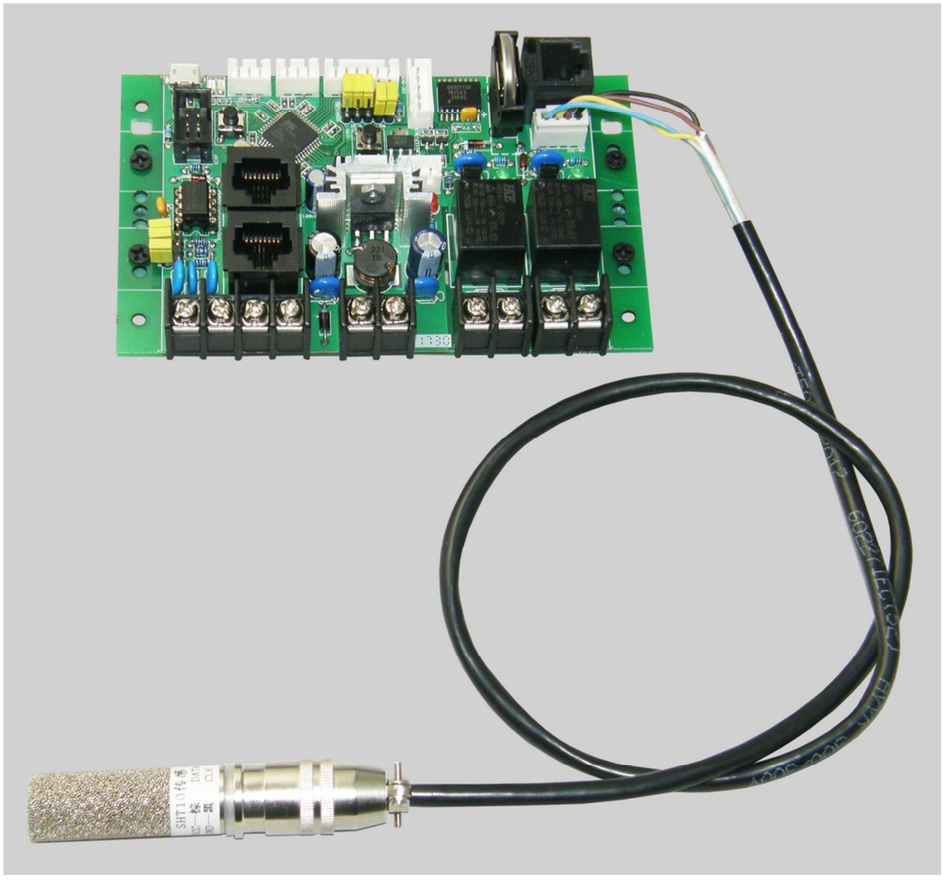


รูปแสดงตัวอย่างการเชื่อมต่อกับอุปกรณ์ I2C Bus 3.3V กับ ET-SENSOR BME280

I2C Bus 5V แบบ **4Pin Block**

เป็นขั้ว I2C Bus แบบ CPA 4PIN Block ใช้สำหรับเชื่อมต่อกับอุปกรณ์ที่เชื่อมต่อผ่าน I2C Bus แบบที่รองรับการใช้งานกับแหล่งจ่ายไฟและระดับสัญญาณโลจิกในการเชื่อมต่อเป็น 5V เช่น เซ็นเซอร์สำหรับตรวจวัดอุณหภูมิและความชื้นในอากาศและในดิน รุ่น ET-SHT10 WATER PROOF SENSOR หรือบอร์ดขยาย Input/Output ต่างๆ เช่น ET-I2C REL8 หรือ ET-I2C DCIN8 เป็นต้น

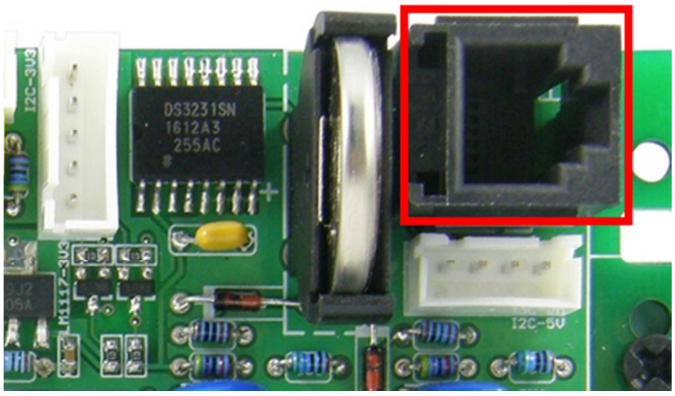


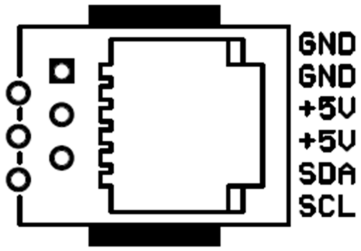


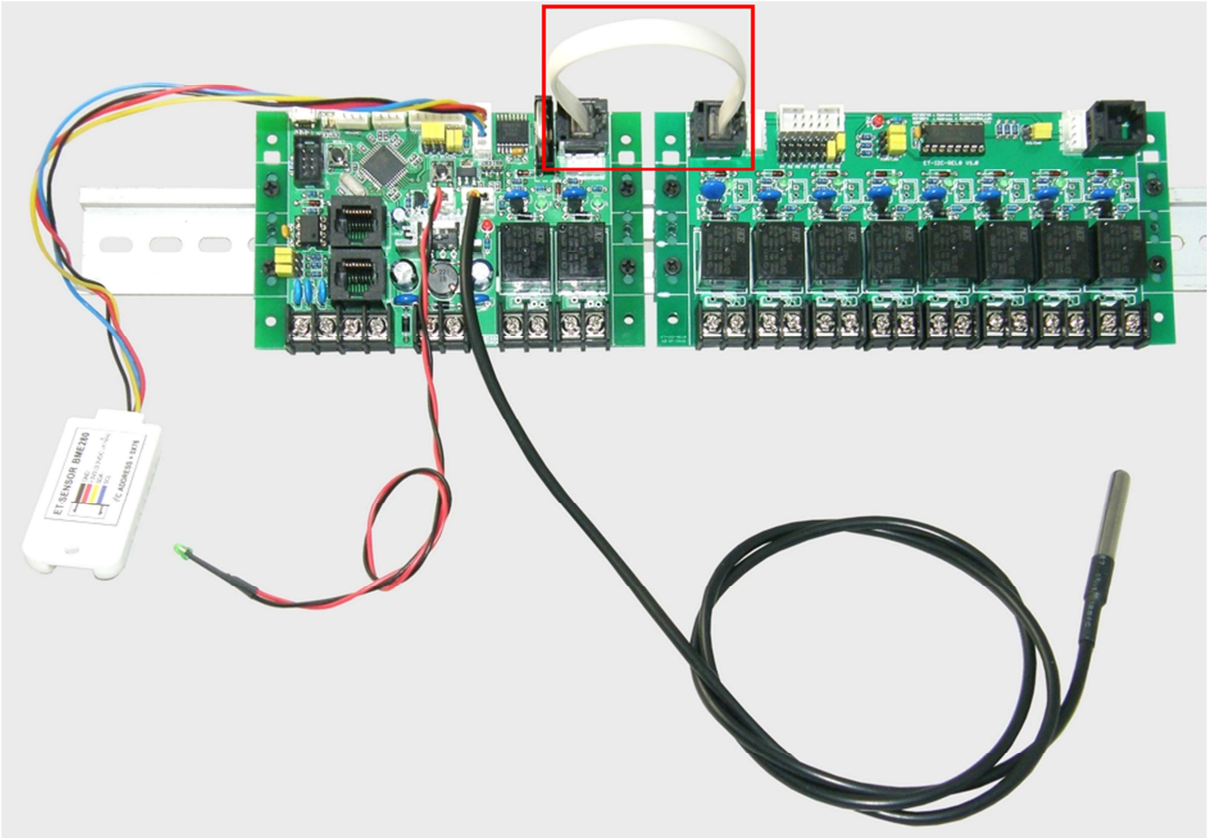
รูปตัวอย่างการเชื่อมต่อ I2C Bus(5V) แบบ 4 Pin กับ ET-SHT10 WATER PROOF SENSOR

I2C Bus แบบ **RJ11(5V)**

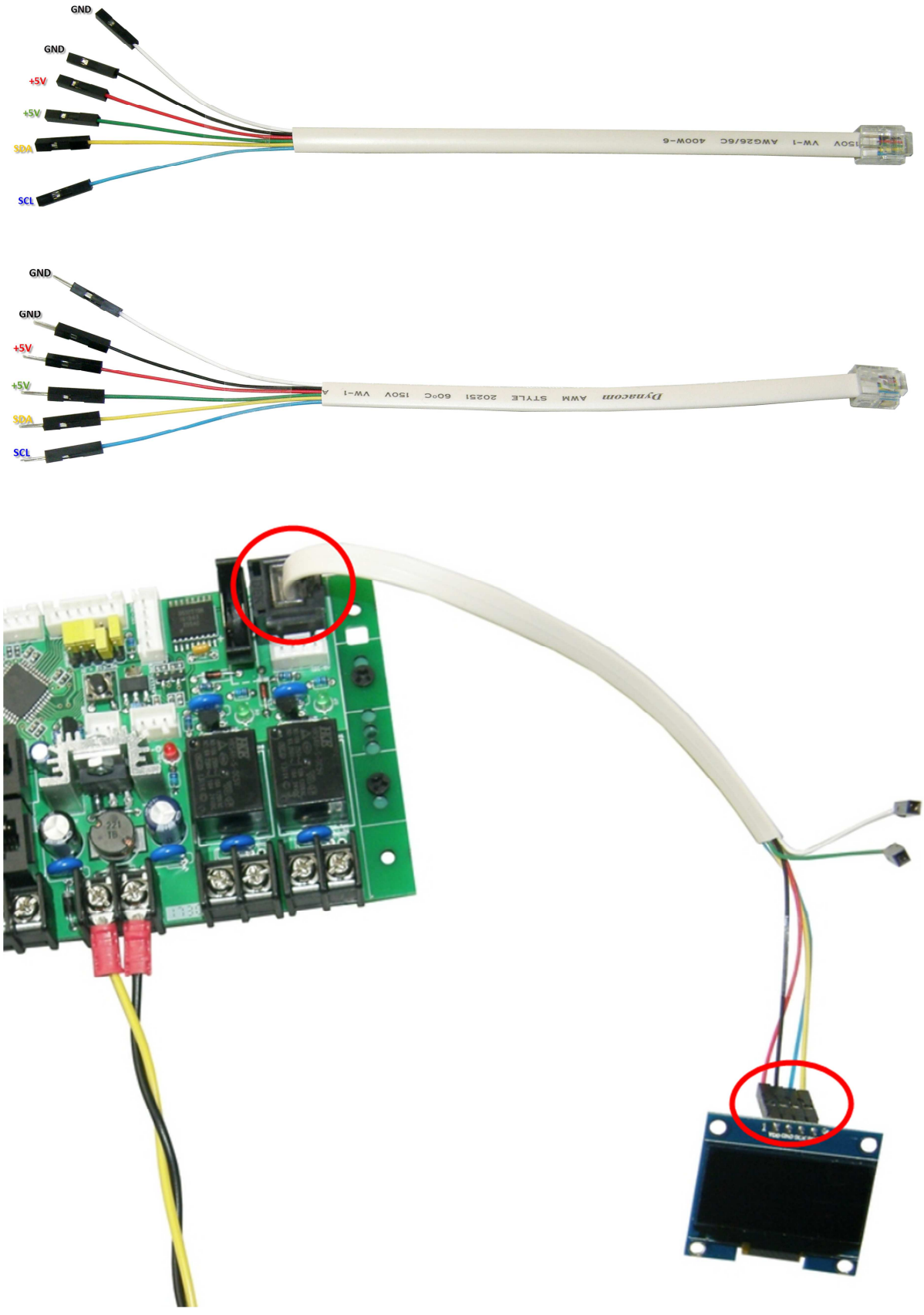
เป็นขั้ว I2C Bus แบบ RJ11 6PIN ใช้สำหรับเชื่อมต่อกับอุปกรณ์ที่เชื่อมต่อผ่าน I2C Bus แบบที่รองรับการใช้งานกับแหล่งจ่ายไฟและระดับสัญญาณโลจิกในการเชื่อมต่อเป็น 5V เช่น บอร์ดขยาย Input/Output ต่างๆ เช่น ET-I2C REL8 หรือ ET-I2C DCIN8 เป็นต้น







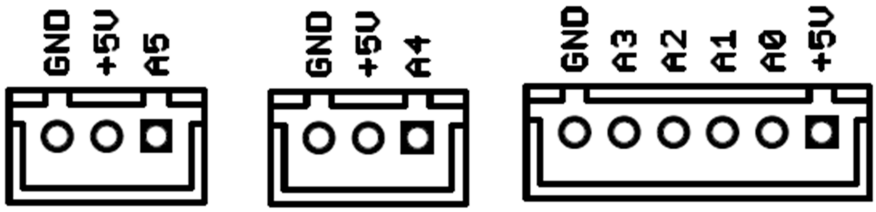
รูปตัวอย่างการต่อขยาย Output Relay ด้วย I2C Bus 5V กับบอร์ด ET-I2C REL8

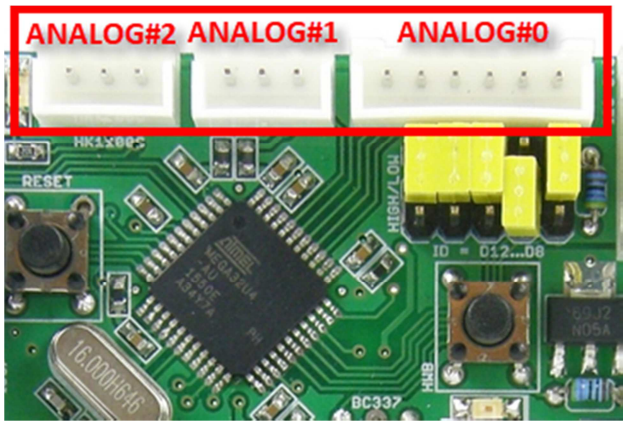


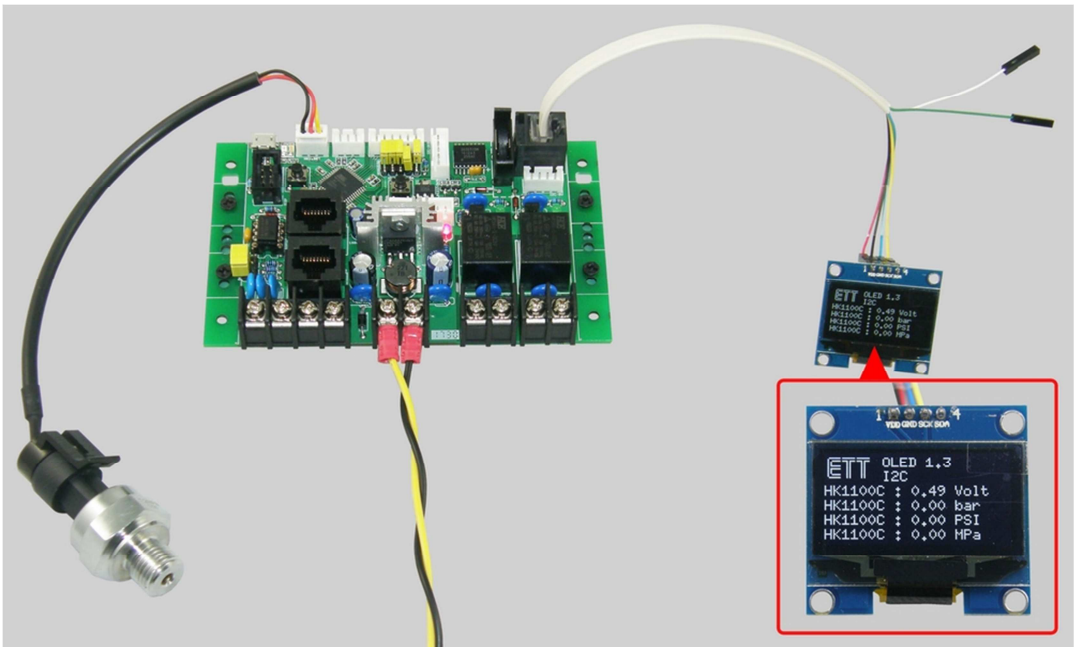
รูปตัวอย่างการเชื่อมต่อขั้ว I2C Bus แบบ RJ11 กับ I2C Display แบบ OLED

ANALOG(AN0..AN5)

บอร์ด ET-MEGA32U4 RS485 มีสัญญาณ ANALOG ให้ใช้งานจำนวน 6 ช่อง โดย จัดเรียงสัญญาณเชื่อมต่อผ่านขั้ว Connector จำนวน 3 ชุด คือ ขั้วต่อ ANALOG#0 จำนวน 4ช่อง (A0…A3) ขั้วต่อ ANALOG#1 จำนวน 1ช่อง (A4) และขั้วต่อ ANALOG#2 จำนวน 1ช่อง (A5) ดังนี้



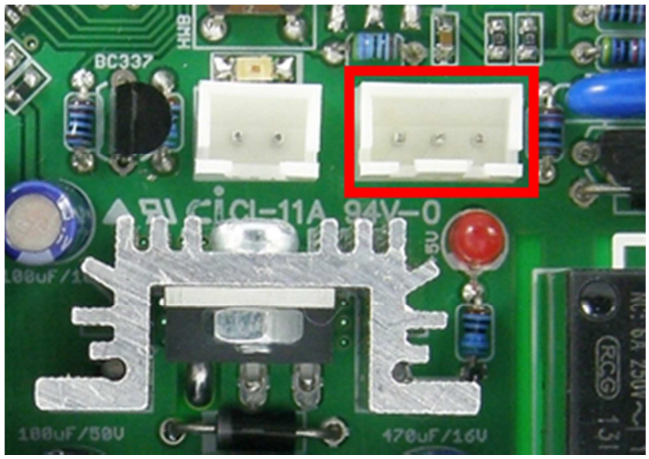


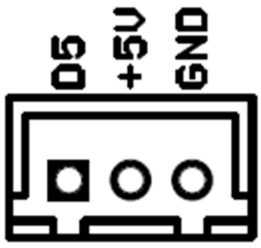


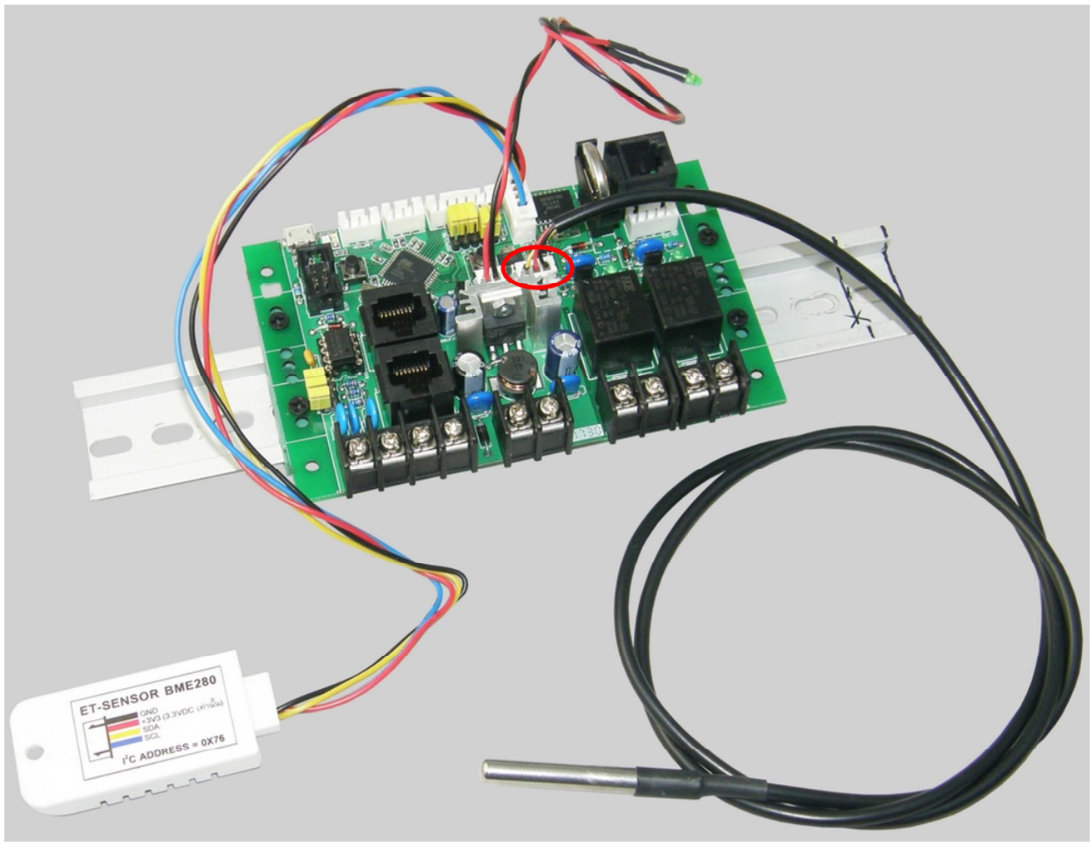
รูปตัวอย่างการต่อใช้งาน ANALOG#2 กับเซ็นเซอร์ HK1100C เพื่อวัดความดัน 0-12บาร์

1-Wire Bus

1-Wire จะใช้สัญญาณ Digital I/O Pin D5(PC6) ซึ่งสามารถนำไปประยุกต์ใช้งานเป็น Digital Input หรือ Digital Output หรือ 1-Wire Bus สำหรับเชื่อมต่อกับอุปกรณ์ 1-Wire เช่น DS18B20



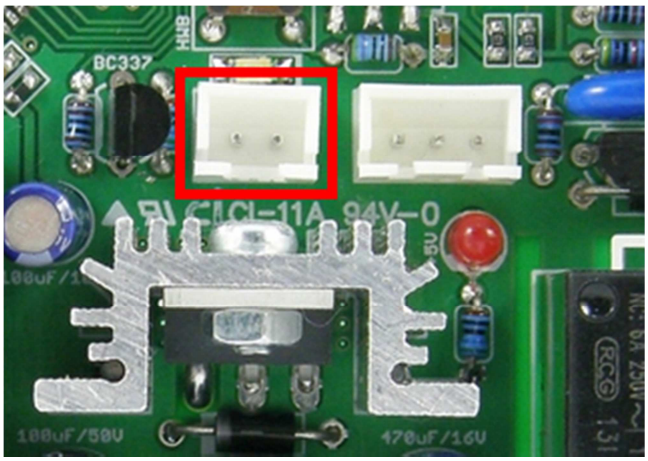


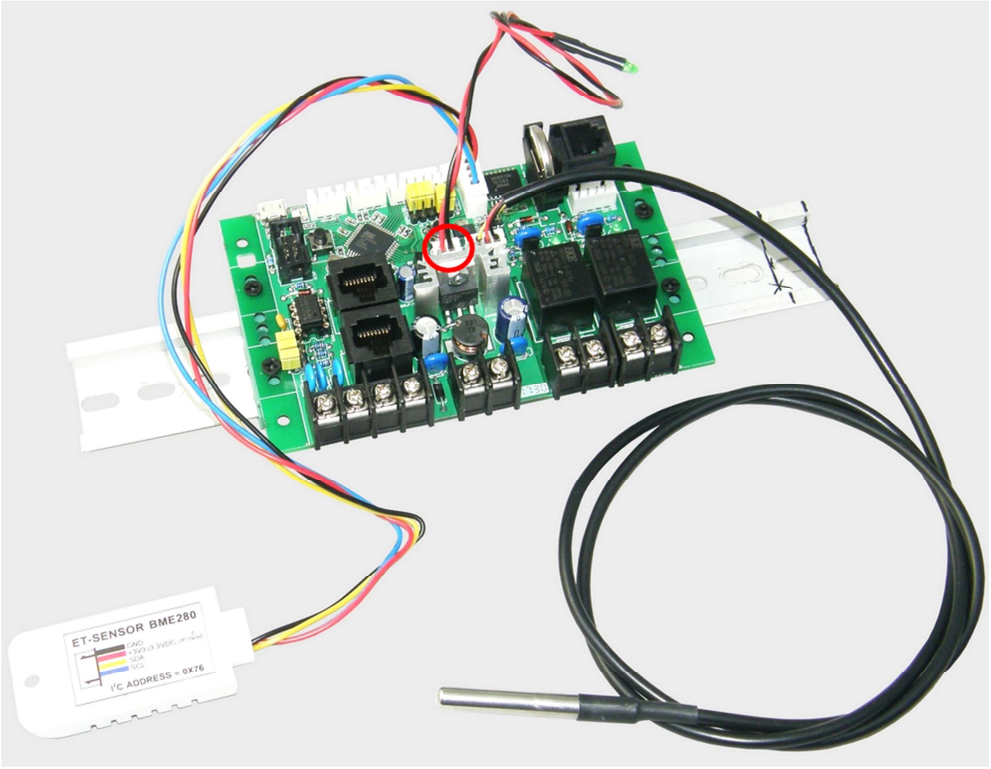


รูปแสดงตัวอย่างการต่อเซ็นเซอร์วัดอุณหภูมิแบบ 1-Wire เบอร์ DS18B20

External LED Status

LED Status จะใช้สัญญาณ Digital I/O Pin D13(PC7) สำหรับสั่งงานควบคุมการติดดับของ LED โดยจะมีทรานซิสเตอร์ช่วยขับกระแสและมีขั้วต่อสำหรับต่อกับ LED ภายนอกได้อีก เช่นเมื่อต้องการติดตั้งเป็น LED Status แสดงการทำงานติดไว้ภายนอกกล่องโดยการต่อผ่าน Connector มาเชื่อมกับบอร์ดได้



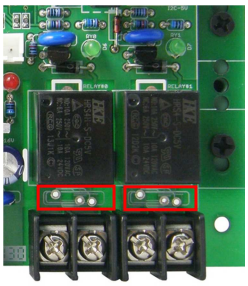


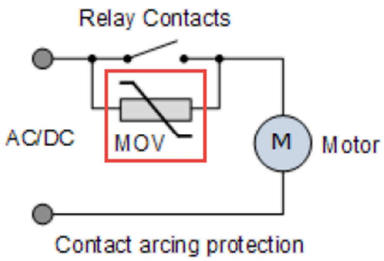
รูปแสดงตัวอย่างการเชื่อมต่อ LED Status สำหรับนำไปแสดงผลนอกบอร์ด

การใช้งาน **Output Relay**

บอร์ด **ET-MEGA32U4-RS485** จะมี **Output Relay** จำนวน **2**ช่อง แต่ละช่องทำงานอิสระต่อกัน โดย **Output** แต่ละชุดจะมีขั้วต่อแบบ **Terminal 7.62mm** ขนาด **2Pin** เป็นจุดเชื่อมต่อใช้งาน โดยจะเป็นจุดต่อหน้าสัมผัส **Relay** ชนิด **NO(Normal Open)** โดยหน้าสัมผัสแต่ละชุดสามารถรับกระแสได้สูงสุด 10แอมป์ ซึ่งหน้าสัมผัสจะมีคุณสมบัติเหมือน สวิตช์ เปิด ปิด อุปกรณ์ไฟฟ้า โดยในสภาวะปรกติตอนที่ **Relay** ยังไม่ทำงาน หน้าสัมผัสนี้จะยังไม่ต่อเชื่อมถึงกันเหมือนการปิดสวิตช์ แต่เมื่อสั่งให้ **Relay** ทำงาน หน้าสัมผัสนี้จึงจะเชื่อมต่อเข้าถึงกันเหมือนการเปิดสวิตช์ ดังนั้นเราจึงสามารถนำหน้าสัมผัสของ **Relay** นี้ไปใช้สั่ง เปิด ปิด อุปกรณ์ไฟฟ้าต่างๆแทนสวิตช์ได้ เพียงแต่หน้าสัมผัส **Relay** นี้จะมีความพิเศษกว่าหน้าสัมผัสสวิตช์ทั่วๆตรงที่ ไม่ต้องใช้มือกดเพื่อสั่ง เปิด ปิด เอง แต่เราสามารถสั่ง เปิด ปิด สวิตช์นี้ได้จากโปรแกรมโดยกำหนดเงื่อนไขต่างๆได้เอง โดยสามารถสั่ง **ON Relay** ได้โดยกำหนด **Logic Output** ของ **GPIO Output** ให้เป็น **LOW** และสั่ง **OFF Relay** ได้โดยการกำหนด **Logic Output** ของ **GPIO Output** ให้เป็น **HIGH**

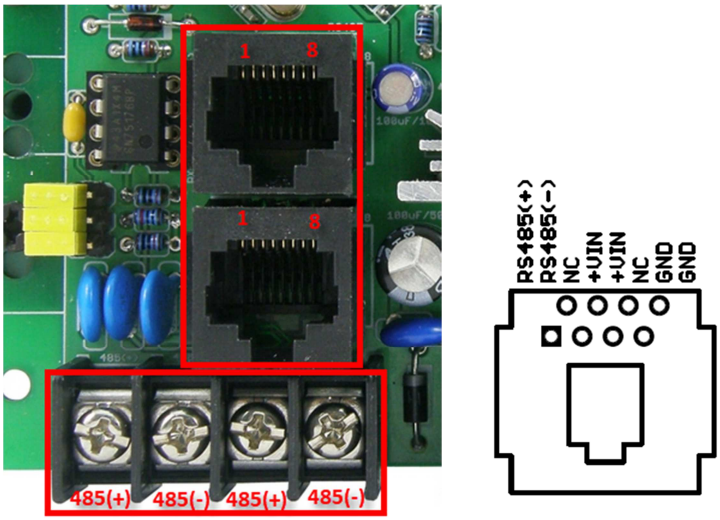
ในกรณีที่นำหน้าสัมผัสรีเลย์ไปใช้เปิดปิดอุปกรณ์ไฟฟ้าที่มีขนาดพิกัดกระแสสูงๆ โดยเฉพาะอุปกรณ์ไฟฟ้าที่เป็นขดลวด เช่น วาล์วไฟฟ้า และ มอเตอร์ ซึ่งอุปกรณ์เหล่านี้จะดึงกระแสผ่านตัวเองในพิกัดที่สูงกว่าปรกติ 2-3เท่าตัว เพื่อใช้ในการสตาร์ทและเริ่มต้นทำงาน ซึ่งในขณะที่ ON และ OFF มักจะเกิดการกระชากอย่างรุนแรงผ่านหน้าสัมผัส ซึ่งจะทำให้เกิดการอาร์คและเกิดสัญญาณรบกวนให้กับอุปกรณ์ไฟฟ้าต่างๆที่ต่อใช้งานร่วมกันอยู่ในระบบไฟฟ้าเดียวกันได้ ซึ่งเราสามารถลดการกระชากป้องกันการอาร์คที่หน้าสัมผัสนี้ได้โดยการติดตั้ง MOV(Varistor) คร่อมเข้าไปที่หน้าสัมผัสได้ โดยที่ไกล้ๆขั้วต่อของหน้าสัมผัสแต่ละชุดของบอร์ด ET-MEGA32U4-RS485 ผู้ใช้สามารถติดตั้ง MOV สำหรับป้องกันการอาร์คที่หน้าสัมผัสเมื่อสั่ง เปิด ปิด หน้าสัมผัสได้ ซึ่งสามารถเลือกใช้ MOV ขนาดต่างๆให้เหมาะสมกับขนาดและประเภทแรงดันไฟฟ้าทำนำไปใช้งานงานเปิดปิดอุปกรณ์ไฟฟ้าทั้ง กระแสตรง และ กระแสสลับ



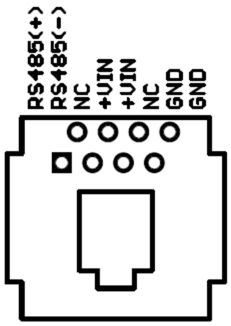


รูปแสดง ตำแหน่งและวงจรการติดตั้ง **MOV (Varistor)** กับหน้าสัมผัส **Relay** ในบอร์ด **ET-MEGA32U4-RS485**

RS485 Bus



RS485 Bus เป็นบัสการสื่อสาร USART แบบ Half Duplex สามารถประยุกต์ใช้สื่อสารรับส่งข้อมูลเป็นระบบเครือข่ายในระยะทางที่ห่างไกลกันได้เป็นอย่างดี โดยจะใช้ D1(PD3) เป็น TXD ใช้ D0(PD2) เป็น RXD และใช้ D4(PD4) เป็น DIR Direction สำหรับเลือกทิศทางการรับส่งข้อมูลใน RS485 Bus โดยต้องกำหนดให้ D4(DIR) ทำหน้าที่เป็น Digital Output Pin ถ้ากำหนดให้เป็น Logic LOW(“0”) จะเป็นการกำหนดทิศทางเป็นฝ่ายรับข้อมูลจาก RS485 Bus ถ้ากำหนดเป็น Logic HIGH(“1”) จะเป็นการกำหนดทิศทางเป็นฝ่ายส่งข้อมูลออกไปใน RS485 Bus จุดเชื่อมต่อสัญญาณ RS485 Bus จะมี 2แบบ คือ Terminal ขนาด 7.62มม. และขั้วต่อ Connector แบบ RJ45 อย่างละ 2ชุด ให้ผู้ใช้เลือกใช้ได้ตามความสะดวกและเหมาะสม

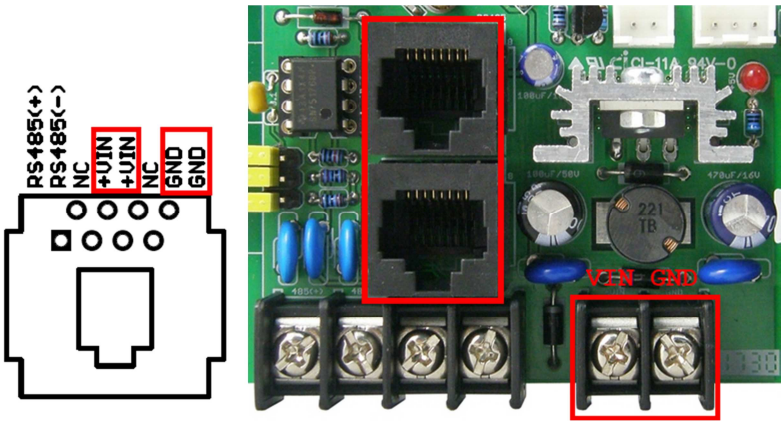


1	2	3	4	5	6	7	8
RS485(+)	RS485(-)	NC	+V(7-30V)	+V(7-30V)	NC	GND	GND

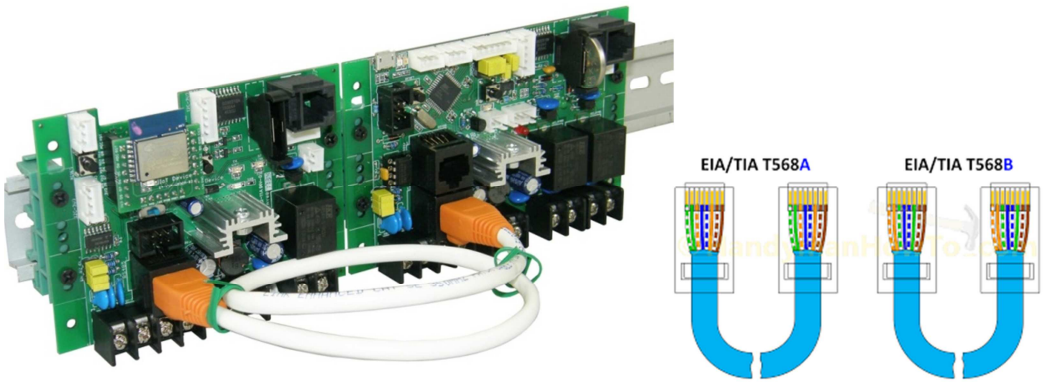
รูปแสดง การจัดตำแหน่งสัญญาณของ RS485 ในขั้ว RJ45

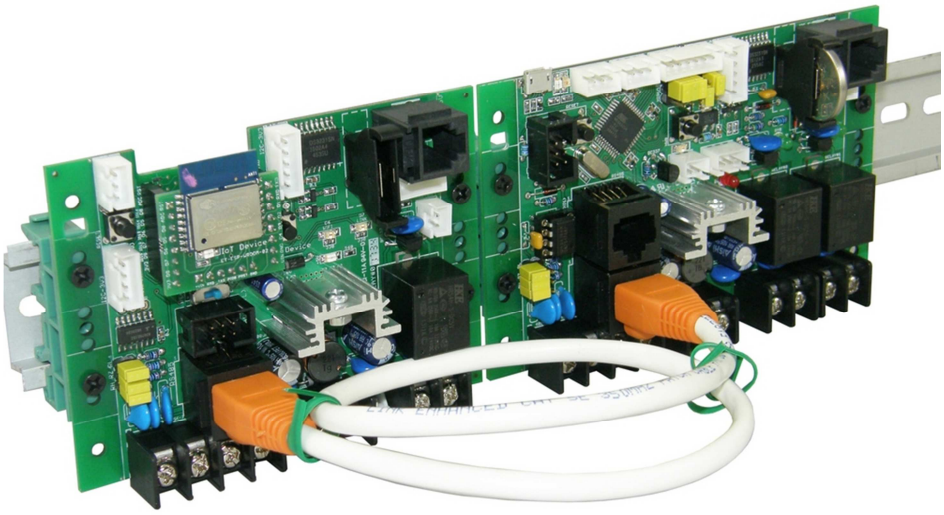
แหล่งจ่ายไฟเลี้ยง **Power Supply**

บอร์ด **ET-MEGA32U4-RS485** มีจุดรับไฟเลี้ยงวงจรเพื่อใช้เป็นแหล่งจ่ายไฟให้อุปกรณ์ในบอร์ด ซึ่งรองรับแรงดันไฟฟ้ากระแสตรงได้ในย่าน 7-30VDC โดยมีจุดเชื่อมต่อแหล่งจ่ายไฟเลี้ยงวงจร 3ช่องทาง คือ ขั้ว Terminal 7.62mm ขนาด 2Pin และ Connector RJ45 อีก 2ชุด

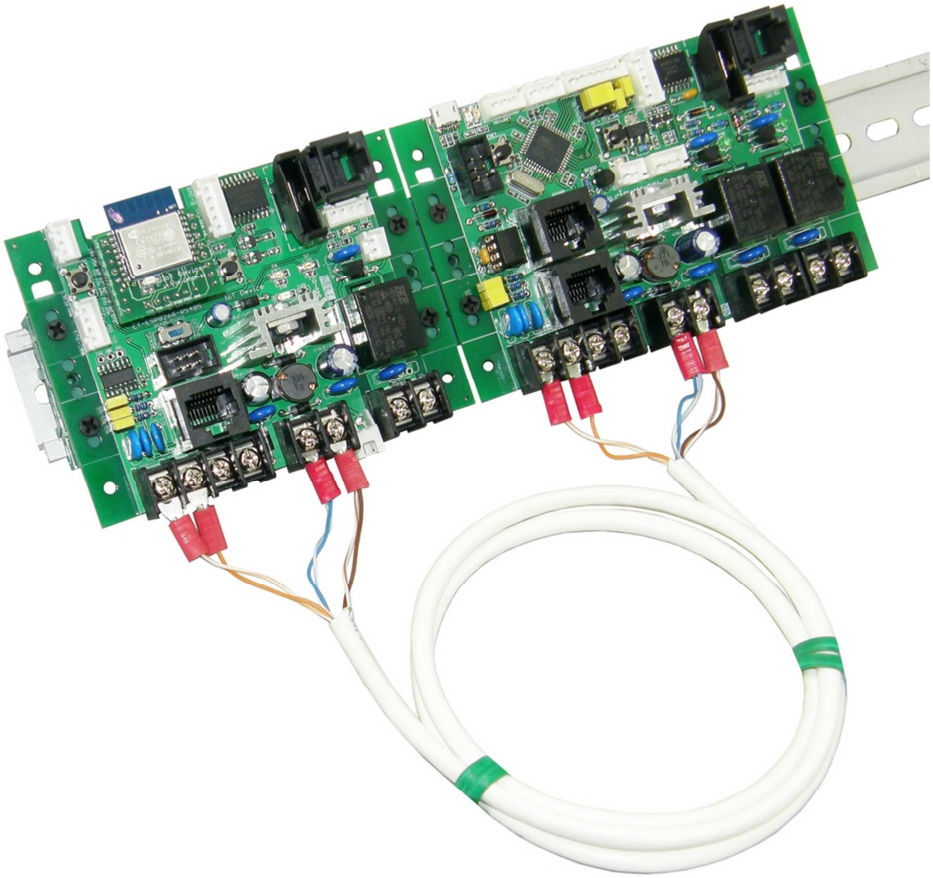


โดยในกรณีที่ใช้การเชื่อมต่อแบบ RJ45 นั้น สามารถใช้สาย UTP ที่ใช้กับเครือข่ายระบบ LAN แบบ Direct ตามมาตรฐาน EIA/TIA T568A หรือ EIA/TIA T568B มาใช้เป็นสายสื่อสารและแหล่งจ่ายไฟ Power Supply ให้กับอุปกรณ์ในบอร์ดไปพร้อมๆกันในสายสัญญาณเส้นเดียวกันได้ แต่อย่างไรก็ตามในกรณีที่จุดใช้งานมีอุปกรณ์อื่นๆที่ต้องต่อใช้งานเพิ่มเติมมากกว่าอุปกรณ์ในบอร์ดและอุปกรณ์นั้นมีความต้องการใช้กระแสมาก ขนาดสายและหน้าสัมผัสของขั้วต่อ RJ45 อาจไม่สามารถรองรับการใช้การในลักษณะอย่างนี้ได้ ผู้ใช้จำเป็นต้องแยกสายสำหรับใช้เป็นคู่สายของ Power Supply ในขนาดสายที่รองรับพิกัดกระแสไฟฟ้าได้สูงเพียงพอกับความต้องการของอุปกรณ์ไฟฟ้าที่จะใช้งานเองด้วย





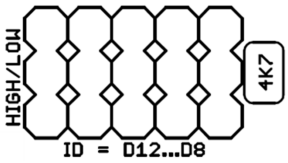
ตัวอย่างการต่อสาย **RS485** โดยใช้ขั้ว **RJ45** โดยใช้สายแลนด์มาตรฐาน **EIA/TIA T568A/B**

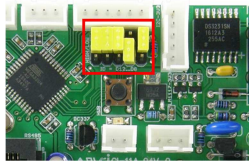


ตัวอย่างการต่อสาย **RS485** และ แหล่งจ่าย **Power 7-30V** โดยใช้ขั้ว **Terminal 7.62**มม.

การกำหนด Address ของ RS485

บอร์ด ET-MEGA32U4-RS485 ออกแบบให้มี Jumper สำหรับเลือกกำหนด ID Code ของบอร์ดจำนวน 5ชุด ซึ่งจะสามารถเลือกกำหนดค่า Address ของบอร์ดให้มีความแตกต่างไม่ซ้ำกันได้ 32ตำแหน่ง โดย Jumper แต่ละตัวจะเลือกกำหนดตำแหน่งเป็น LOW หรือ HIGH ได้ตามต้องการดังตาราง





การกำหนด Jumper เพื่อตั้งค่า ID Code					ID Code
ID4(D12)	ID3(D11)	ID2(D10)	ID1(D9)	ID0(D8)	
LOW	LOW	LOW	LOW	LOW	00
LOW	LOW	LOW	LOW	HIGH	01
LOW	LOW	LOW	HIGH	LOW	02
LOW	LOW	LOW	HIGH	HIGH	03
LOW	LOW	HIGH	LOW	LOW	04
LOW	LOW	HIGH	LOW	HIGH	05
LOW	LOW	HIGH	HIGH	LOW	06
LOW	LOW	HIGH	HIGH	HIGH	07
LOW	HIGH	LOW	LOW	LOW	08
LOW	HIGH	LOW	LOW	HIGH	09
LOW	HIGH	LOW	HIGH	LOW	10
LOW	HIGH	LOW	HIGH	HIGH	11
LOW	HIGH	HIGH	LOW	LOW	12
LOW	HIGH	HIGH	LOW	HIGH	13
LOW	HIGH	HIGH	HIGH	LOW	14
LOW	HIGH	HIGH	HIGH	HIGH	15
HIGH	LOW	LOW	LOW	LOW	16
HIGH	LOW	LOW	LOW	HIGH	17
HIGH	LOW	LOW	HIGH	LOW	18
HIGH	LOW	LOW	HIGH	HIGH	19
HIGH	LOW	HIGH	LOW	LOW	20
HIGH	LOW	HIGH	LOW	HIGH	21
HIGH	LOW	HIGH	HIGH	LOW	22
HIGH	LOW	HIGH	HIGH	HIGH	23
HIGH	HIGH	LOW	LOW	LOW	24
HIGH	HIGH	LOW	LOW	HIGH	25
HIGH	HIGH	LOW	HIGH	LOW	26
HIGH	HIGH	LOW	HIGH	HIGH	27
HIGH	HIGH	HIGH	LOW	LOW	28
HIGH	HIGH	HIGH	LOW	HIGH	29
HIGH	HIGH	HIGH	HIGH	LOW	30
HIGH	HIGH	HIGH	HIGH	HIGH	31

ตารางแสดง การเลือกกำหนดรหัส **ID Code** ของบอร์ด **ET-MEGA32U4-RS485**

ตัวอย่างการกำหนดฟังก์ชั่นใช้งาน Serial USB และ Serial RS485 ของบอร์ด ET-MEGA32U4-RS485

```
/* Demo ET-MEGA32U4-RS485 Serial Port Interface
 * MCU    : ATMEGA32U4(Arduino Leonado)
 *        : Bootloader
 *        : -> .../caterina/Caterina-Leonardo.hex
 *        : Fuse Bit
 *        : -> low_fuses      = 0xFF
 *        : -> high_fuses     = 0xD8
 *        : -> extended_fuses = 0xCB(0xFB)
 *        : Lock Bit
 *        : -> 0x2F(0xEF)
 * RS485  : RS485 RXD:D0
 *        : RS485 TXD:D1
 *        : RS485 Direction(D4 : LOW = RXD, HIGH = TXD)
 */
#define SerialDebug Serial                    // USB Serial
#define SerialRS485 Serial1                   // Serial1(D1=TXD,D0=RXD)
//================================================================
const int RS485_DIRECTION_PIN =     4;        // RS485 Direction
const int RS485_RXD_SELECT    =     LOW;
const int RS485_TXD_SELECT    =     HIGH;
//================================================================
const int RS485_ID0_PIN       =     8;        // Slave ID LSB
const int RS485_ID1_PIN       =     9;
const int RS485_ID2_PIN       =     10;
const int RS485_ID3_PIN       =     11;
const int RS485_ID4_PIN       =     12;        // Slave ID MSB
//================================================================
int SlaveAddress = 0;
char SlaveID[2];
//================================================================
```

```
void setup(void)
{
  pinMode(RS485_DIRECTION_PIN, OUTPUT);
  digitalWrite(RS485_DIRECTION_PIN, RS485_RXD_SELECT);
  pinMode(RS485_ID0_PIN, INPUT_PULLUP);
  pinMode(RS485_ID1_PIN, INPUT_PULLUP);
  pinMode(RS485_ID2_PIN, INPUT_PULLUP);
  pinMode(RS485_ID3_PIN, INPUT_PULLUP);
  pinMode(RS485_ID4_PIN, INPUT_PULLUP);
  //================================================================
  SlaveAddress = 0;
  if(digitalRead(RS485_ID0_PIN) == HIGH) SlaveAddress += 1;
  if(digitalRead(RS485_ID1_PIN) == HIGH) SlaveAddress += 2;
  if(digitalRead(RS485_ID2_PIN) == HIGH) SlaveAddress += 4;
  if(digitalRead(RS485_ID3_PIN) == HIGH) SlaveAddress += 8;
  if(digitalRead(RS485_ID4_PIN) == HIGH) SlaveAddress += 16;
  String StringID = String(SlaveAddress,DEC);
  if(SlaveAddress < 10)
  {
    SlaveID[0] = '0';
    SlaveID[1] = StringID[0];
  }
  else
  {
    SlaveID[0] = StringID[0];
    SlaveID[1] = StringID[1];
  }
  //================================================================
  SerialDebug.begin(115200);                    // Debug USART
  SerialRS485.begin(115200);                    // RS485 USART
  while(!SerialDebug);                          // wait for USB serial connect.
}
```

```
void loop(void)
{
  //=========================================================================
  SerialDebug.print("My Slave ID : ");
  SerialDebug.write(SlaveID[0]);
  SerialDebug.write(SlaveID[1]);
  //=========================================================================
  digitalWrite(RS485_DIRECTION_PIN, RS485_TXD_SELECT);
  SerialRS485.print("My RS485 Slave ID:");
  SerialRS485.write(SlaveID[0]);
  SerialRS485.write(SlaveID[1]);
  SerialRS485.flush();
  digitalWrite(RS485_DIRECTION_PIN, RS485_RXD_SELECT);
  //=========================================================================
  While(1)
  {
    …
    …
    …
  }
  //=========================================================================
}
```