USB-I2C-SPI-4VI4SRMx

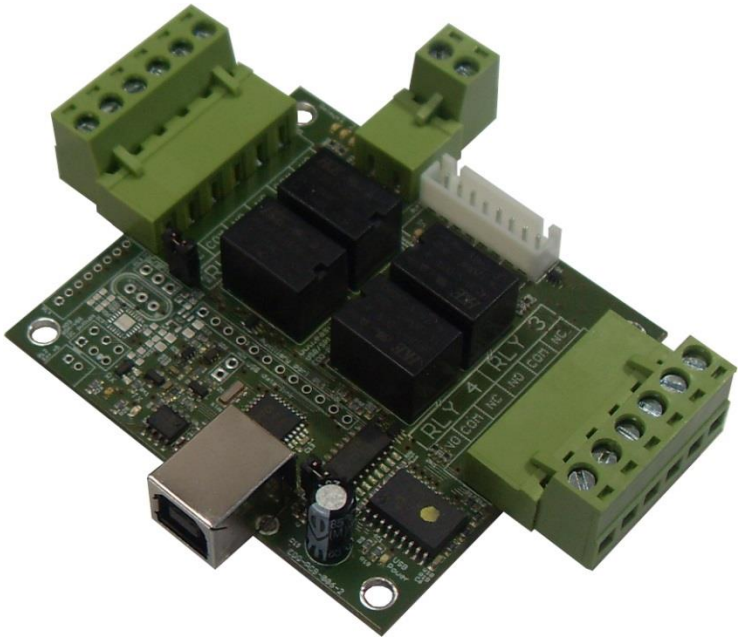
Low cost Data Acquisition & Control products

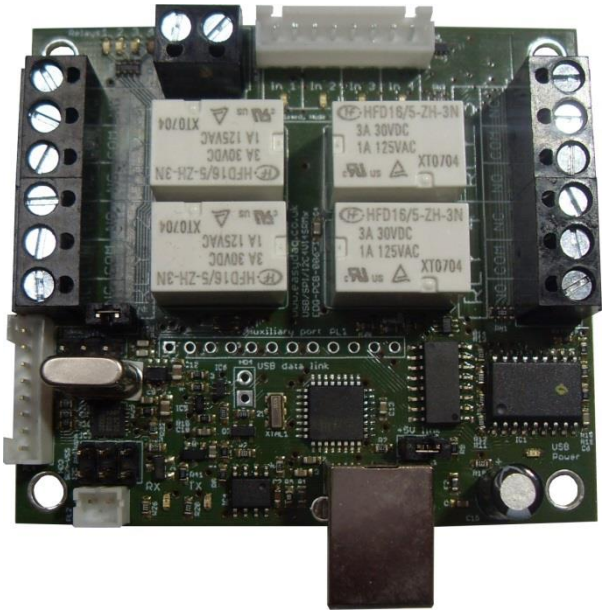
4 Signal Relay, 4 Wide Voltage Input, Multi-Protocol card

Product Datasheet 58

Features

- Control USB and/or I2C/SPI communications
- Communications data activity LEDs to add debugging and testing
- The 2 communications inputs can exist at the same time
- 4 off, 3A, 120V Opto-isolated signal relays + individual LEDs
- 4 Opto-isolated, wide input voltage sensors + individual LEDs. Max input 60V AC or DC.
- Same board outline as USB4SRMx
- 1 or 2 part horizontal connectors
- Optional 12V or 24V operating relays
- Optional bespoke software





Description

General purpose relay card with 4 opto-isolated, 120V, 3A signal relays and 4, 3V to 60V, AC/DC, opto-isolated voltage sensing inputs.

This card has 3 communication options:

- USB
- I2C
- SPI

This card also has 2 powering options:

- USB power
- External 5V DC with reverse polarity protection.

See detailed Power Supply description below.

Two LEDs indicate data transfers between your control hardware and the relay card for help with debugging and testing.

Each relay channel or voltage sensing input has its own LED to indicate status.

The card has been design with future expansion in mind including access up to 6, 5V DIO channels connected to the main controller. The programming pins for the control processor are also available to enable re-purposing of the card by loading application specific software.

The card is also available with reduced options fitted. See the order codes table.

The card is RoHS compliant and CE marked.

Specification

Power supply

- USB or external 5V powered (10mA + up to 4 relays @ 44mA per relay).

Control Interfaces

- USB 1, 2 or 3, Type B connector, hot pluggable.
- I2C or SPI on a JST XH 8pin header.

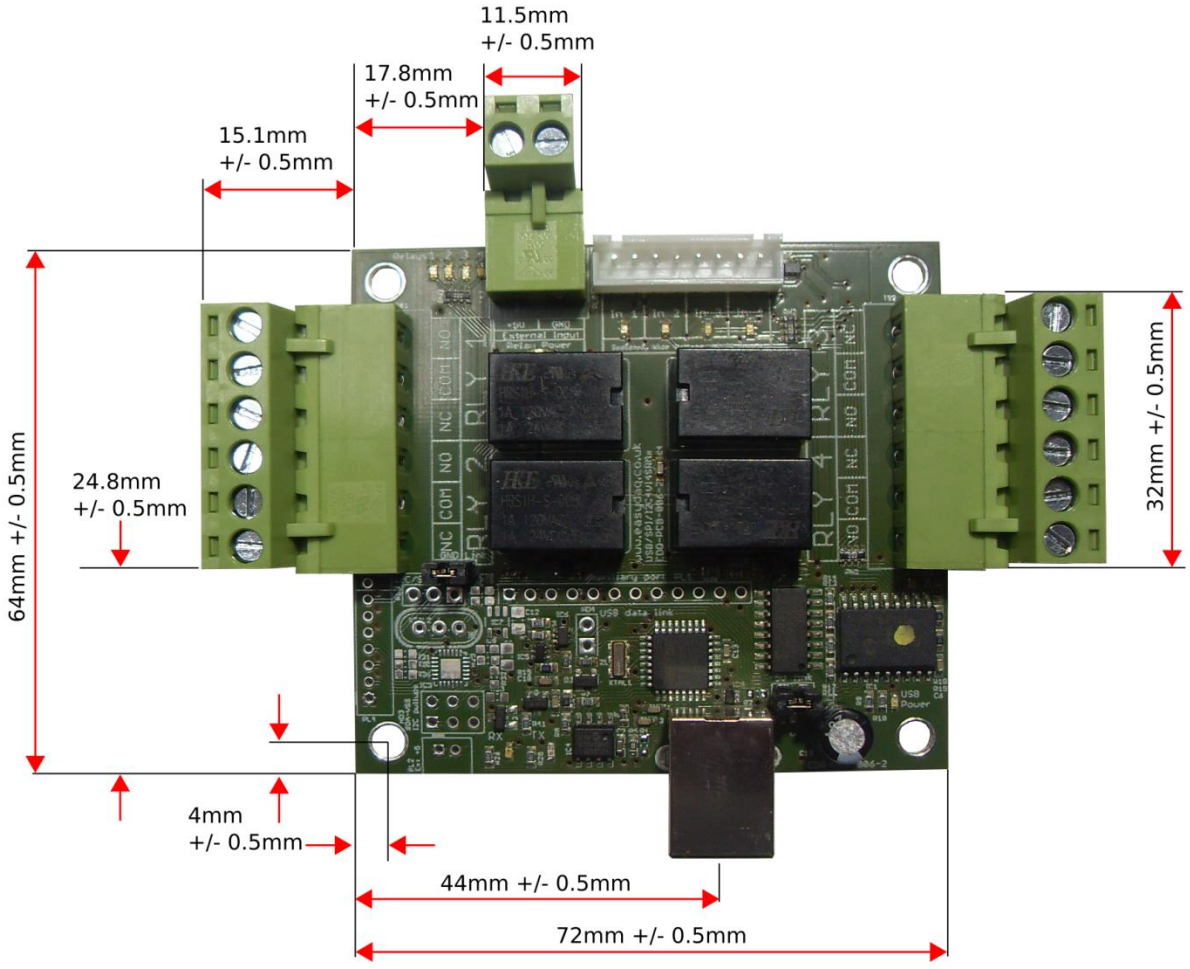
Operating temp range

-20 to +80^{0}C

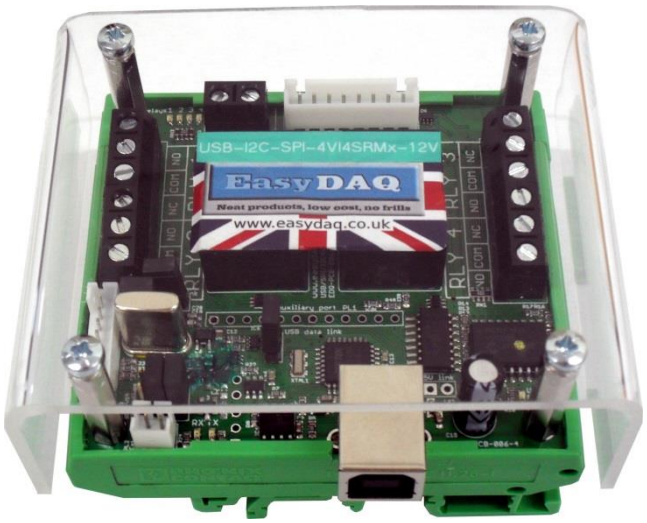
Relays

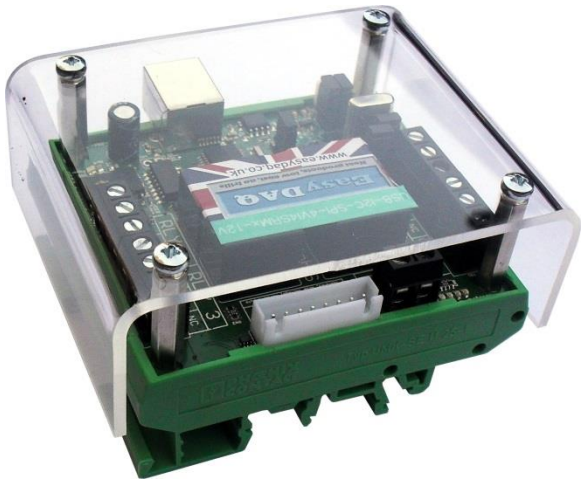
Coil Rated voltage/current	5VDC/44mA each, Standard 12VDC/19mA, -12V version 24VDC/9.2mA, -24V version
Must operate/release voltage	75%/10% of rated voltage
Maximum contact ratings	3A,120VAC or 3A,30VDC
Minimum recommended contact rating	5V @ 20mA
Contact resistance	100mΩ max
Operate/release time	5mS/5mS
Contact material	AuAg overlay, Ag alloy
Operational life (min)	Mechanical 10^7 / Electrical 10^5
Contact arrangement	SPDT, Form C

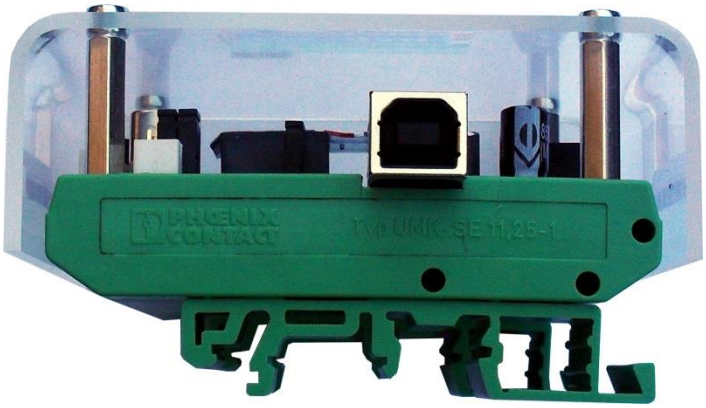
Dimensions



USB-I2C-SPI-4VI4SRMx housed in a CoverUSB4MxDIN





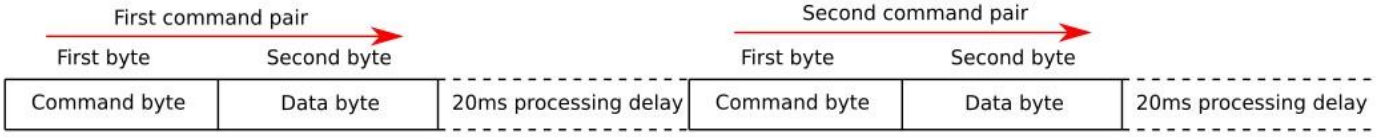


Detailed description and control

Command format

The card is commanded via simple single ASCII characters (+ status byte). This is a 2 byte pair. These are commands that address each port of the PIC processor device (Hex equivalent shown in brackets). The card can be controlled using a Terminal emulator if connected via USB or RS232 see below.

It is important to include a 20ms processing delay between command pairs.

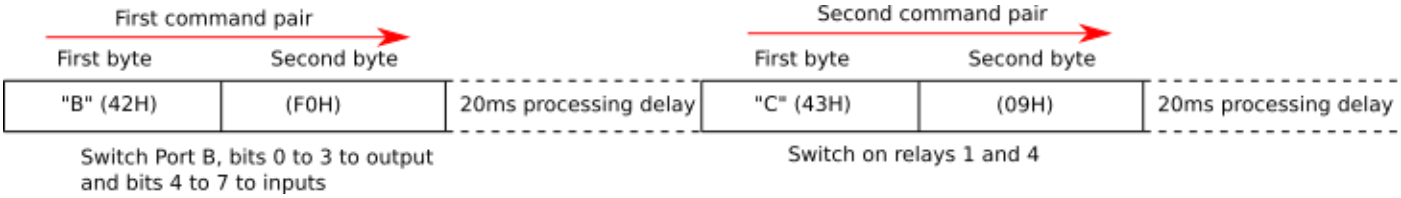


Port B (Channels 1-4) Relay commands:

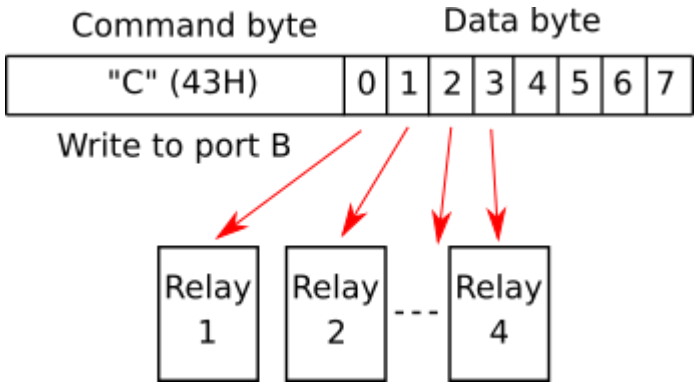
ASCII 'B' (42H), X Initialises the card (sets the port & channel I/O directions). Set direction of Port B, 1=Input, 0= output. (For example, where X=10111111 (AFH) = sets bit 7 as an output, the rest as inputs).

ASCII 'C' (43H), X Write data X to Port B (i.e. X=00000001 (01H), sets channel 1 to active). Valid data bytes are latched by the card until a further valid data byte is written to it.

Example: To set all Port B bits 0 to 3 to output for controlling the relays, Bits 4 to 7 as inputs and switch on relays 1 and 4:

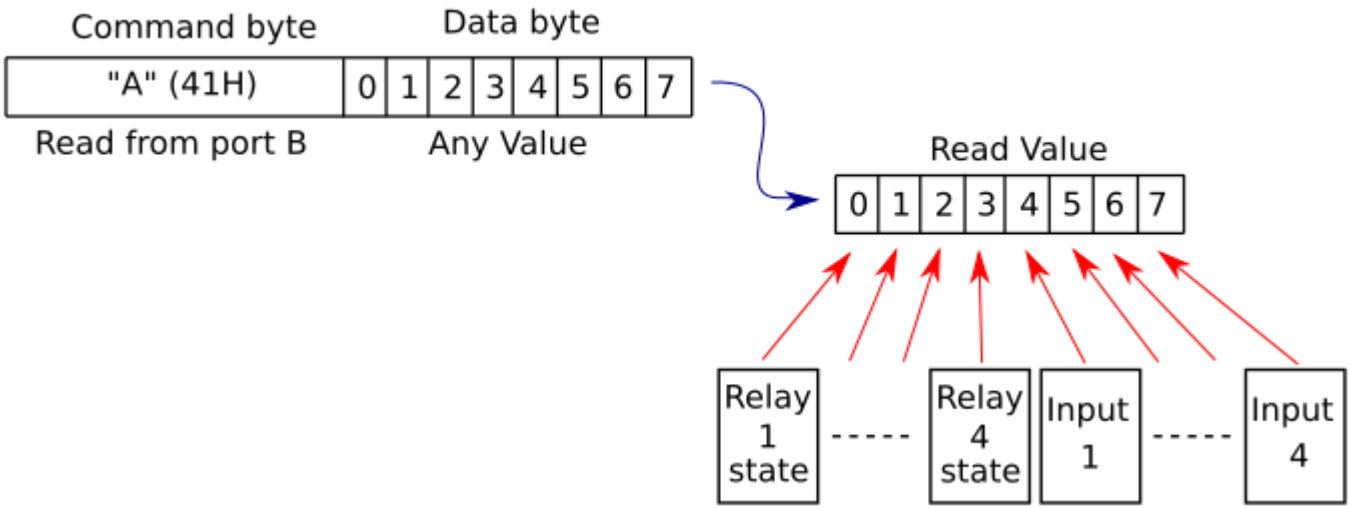


The first byte pair initialises the port to all output and the second and subsequent byte pairs are then used to control the relays. The 8 bits of the data byte represent the relays to be controlled. Relay 1 is controlled by Bit 0, relay 2 is controlled by Bit 1 and so on to relay 8 being controlled by Bit 7.



Port B (Channels 4-7) Input commands:

This is a two byte command with the second byte being any value and is ignored. The data byte will be transmitted after the dummy byte has been received.



The value of the 4 inputs is read from the top 4 bits of the returned value.

Note that this command also returns the state of the 4 relays on Port B.

Summary of EasyDAQ relay card commands.

Letter	HEX	Function	Note
A	41	Read Port B	
B	42	Set direction of Port B	
C	43	Write data to Port B	
D	44	Read Port C	Not used on USB4xxx boards
E	45	Set direction of Port C	
F	46	Write data to Port C	
G	47	Read Port D	
H	48	Set direction of Port D	
I	49	NOT USED	
J	4A	Write data to Port D	
K	4B	Read Port E	
L	4C	Set direction of Port E	
M	4D	Write data to Port E	

Serial Port settings

For the RS232 and USB interfaces the controlling system must be set to:

Baud rate: 9600
Parity: 0
Data: 8 bits
Stop bits: 1
Handshaking: None

Auto detection & com port assignment

When you connect this card to a USB port of your computer for the first time, it will be auto-detected and ask you to install drivers (downloadable from the 'downloads' section of our website). After installation, the card will appear as a 'virtual' COM port and be automatically assigned a COM port number by your OS. Following installation, the COM port number can be manually re-assigned via the control panel if required. Following reboots or disconnects of the USB card, the same COM port number will be assigned.

Using a Terminal Emulator

In order to test operation, the card can be connected to a serial port and controlled from a terminal emulator program such as "PuTTY" or "Realterm". See our "Data Sheet 50 (Using Terminal Emulators to control and test EasyDAQ cards)". Ensure port configuration is set as shown above, type (ASCII) characters shown above to achieve port direction and read or write command/data.

I2C/SPI control

If the I2C/SPI option is fitted then the card can be controlled directly from another processor such as an Arduino, Raspberry Pi, BeagleBone etc.

The communications via the I2C or SPI are managed by a NXP SC16IS750 I2C/SPI to serial converter.

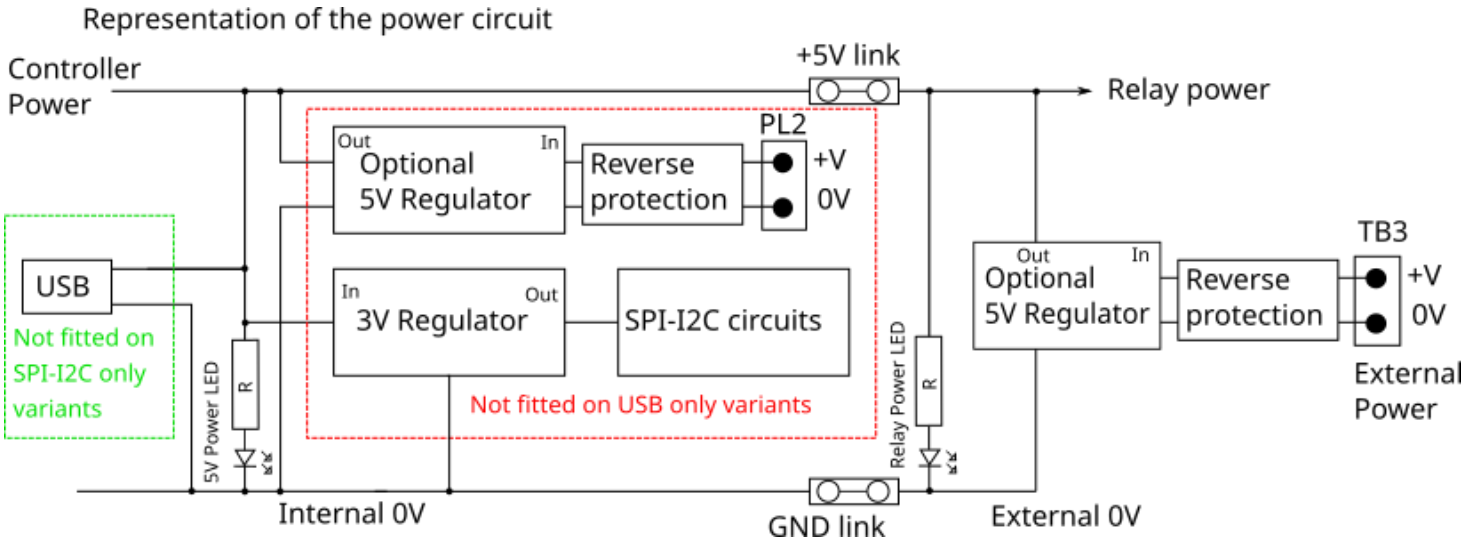
The SC15IS70 needs to be set up by the controlling system to enable communications with the processor on the relay card.

The following is an excerpt from an Arduino program that uses the "Wire.h" library to set up I2C communications:

```
#include <Wire.h>
byte Address = 0x48;          // Address of the SC16IS750 when A0 and A1 are pulled high.
                              // The value in the datasheet has to be shifter right by 1 bit to allow for
                                  // the R/W bit.
int delaytime = 20;           // 20ms minimum delay between instructions

// Register addresses shifted left 3 bits into bits 3:6. See NXP SC16IS740/750/760 datasheet Rev. 7 — 9 June 2011 section
10.4 "Use of subaddresses"

// SC16IS750 Registers :
#define THR         0x00 << 3      // Transmit Holding Register. Can only be written to.
#define RHR         0x00 << 3      // Receive Holding Register. Can only be read from.
#define IER         0x01 << 3
#define FCR         0x02 << 3
#define IIR         0x02 << 3
#define LCR         0x03 << 3      // Line Control Register
#define MCR         0x04 << 3      // Modem Control Register
#define LSR         0x05 << 3
#define MSR         0x06 << 3
#define SPR         0x07 << 3      // Scratch Pad Register
#define TXLVL       0x08 << 3      // Transmitter FIFO Level register (TXLVL) address (0x08)
#define RXLVL       0x09 << 3
#define DLAB        0x80 << 3
#define IODIR       0x0A << 3
#define IOSTATE     0x0B << 3
#define IOINTMSK    0x0C << 3
#define IOControl   0x0e << 3      // I/O Control register
#define EFCR        0x0f << 3      // Extra Features Control Register
#define DLL         0x00 << 3
#define DLH         0x01 << 3
#define EFR         0x02 << 3      // Enhanced Features Register
#define XON1        0x04 << 3
#define XON2        0x05 << 3
#define XOFF1       0x06 << 3
#define XOFF2       0x07 << 3

void setup() {
  Wire.begin();                    // join i2c bus (address optional for master)
  setupBRG();                      //
                                          // rest of your setup here
}

void setupBRG(){
  putByte(IOControl,B00010000);       // Software reset
  delay(100);
  putByte(LCR,0x80);                  // divisor latch enable
  putByte(DLL,12);                    // Lower byte of BRG  ((1.8432MHz/(12x16)). 16 clocks required per bit sent
  putByte(DLH,0x00);                  // Upper byte of BRG  ((1.8432MHz/(12x16)). 16 clocks required per bit sent

  putByte(LCR,0xBF);                  // Access EFR register
  putByte(EFR,0x10);                  // Enable Enhanced functions
  putByte(LCR,0x03);                  // Setup LCR for normal operation: 8 bit data, 1 stop bits, no parity
  putByte(FCR,0x06);                  // Reset FIFOs
  putByte(FCR,0x01);                  // Enable FIFOs
  Serial.println("BRG set up");
  Serial.println("");
}

void putByte(byte reg, byte data){
  Wire.beginTransmission(Address);    // transmit to device
  Wire.write(reg);                    // sends the register address
  Wire.write(data;                    // Setup LCR for normal operation: 8 bit data, 2 stop bits,
                                         // no parity
  Wire.endTransmission();             // stop transmitting
}
```

Complete examples can be found on the download page of our website.

Power options



The USB-I2C-SPI-4VI4SRMx is supplied with "GND Link" and "+5V Link" fitted which allows the board to be powered directly from the USB connector.

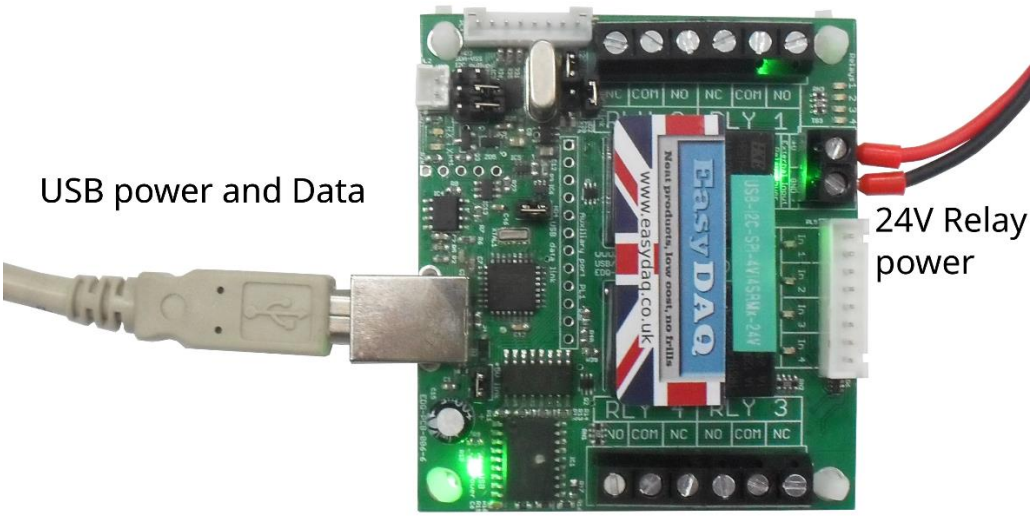
Full isolation can be achieved by removing both the "GND Link" and "+5V Link". In this case an external supply will be required connected to TB3.

The I2C-SPI-4VI4SRMx will require a +5V supply connected to PL2 to power the controller and the I2C- SPI circuitry.

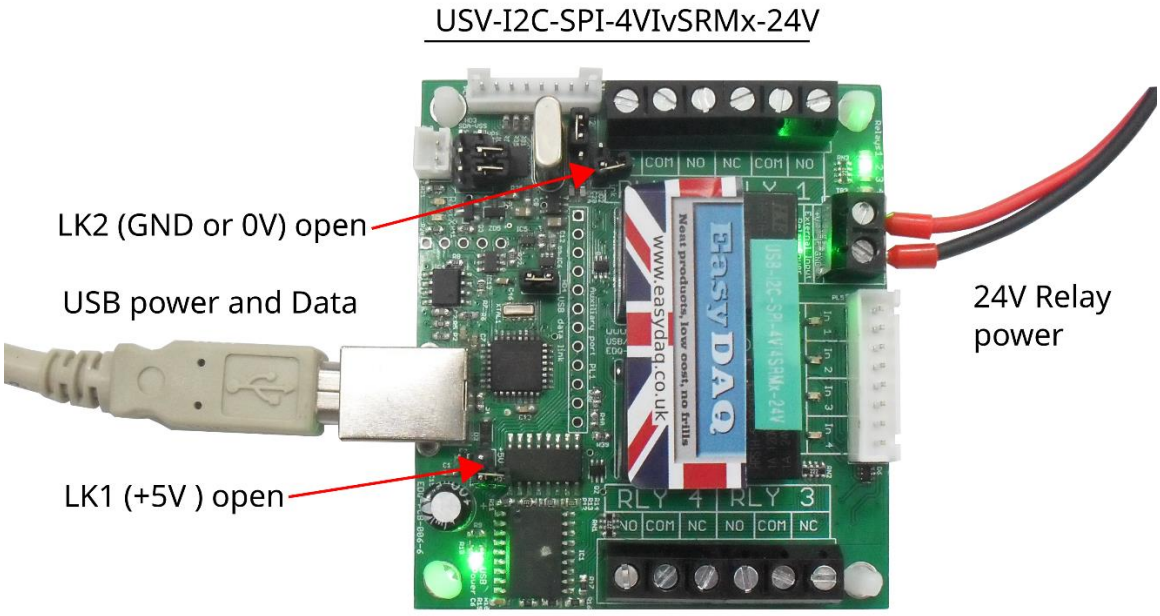
If 12V or 24V relay options are required then an external supply of the required voltage will be necessary on TB3. The 5V supply for the controller is then provided through the USB connector or PL2.

The following images show a USB-I2C-SPI-4VI4SRMx-24V in Non isolated and Isolated configurations:

USV-I2C-SPI-4VIvSRMx-24V



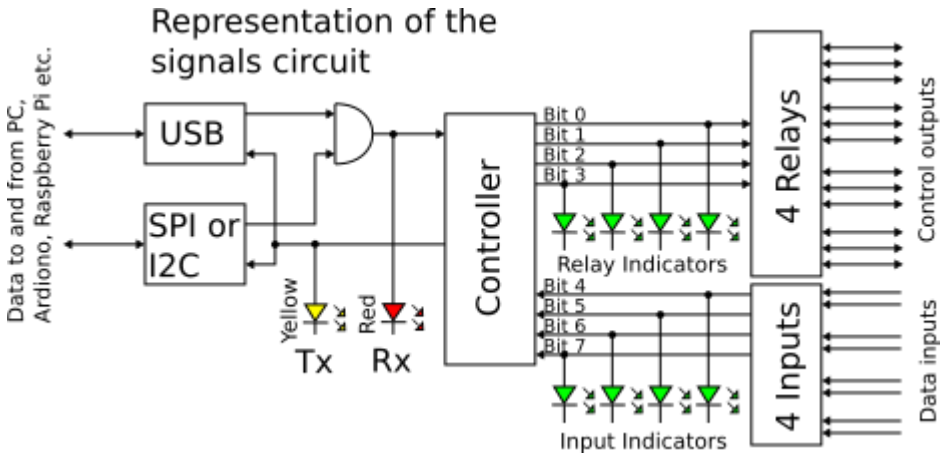
Non Isolated configuration with GND and +5V links



Isolated configuration with open GND and +5V links

Signals overview:

The signal paths for controlling relays and sensing inputs are shown in the following diagram.



Link configurations:

The USB-I2C-SPI-4VI4SRMx has sets of 5 links that are used to configure the board.

LK1 and LK2 isolate the +5V and GND (0V) rails between the signal and relay side of the board.

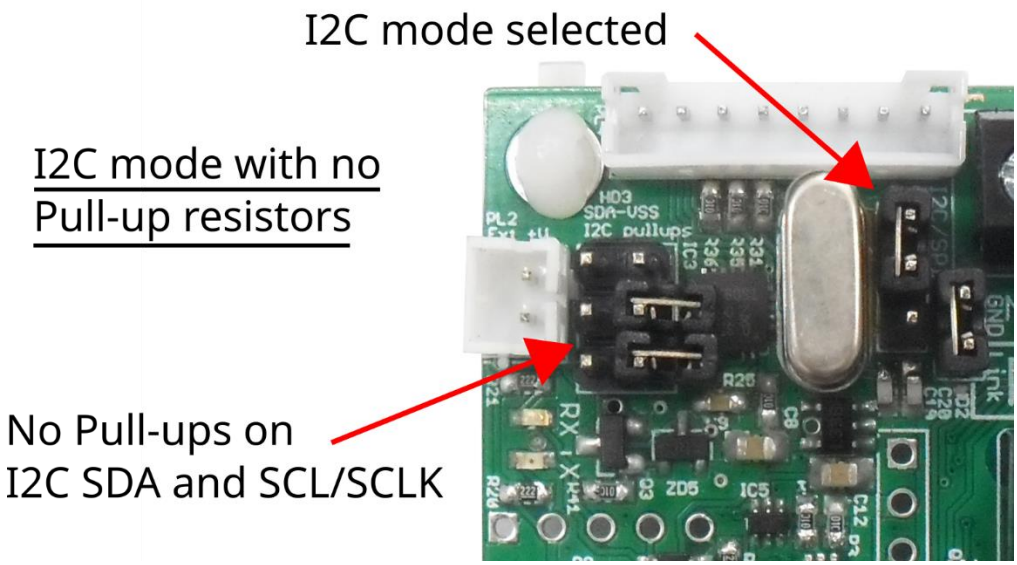
PL3 selects between SPI and I2C mode.

HD3 is used to configure the I2C signal pull-up resistors and SPI SDA Pull down.

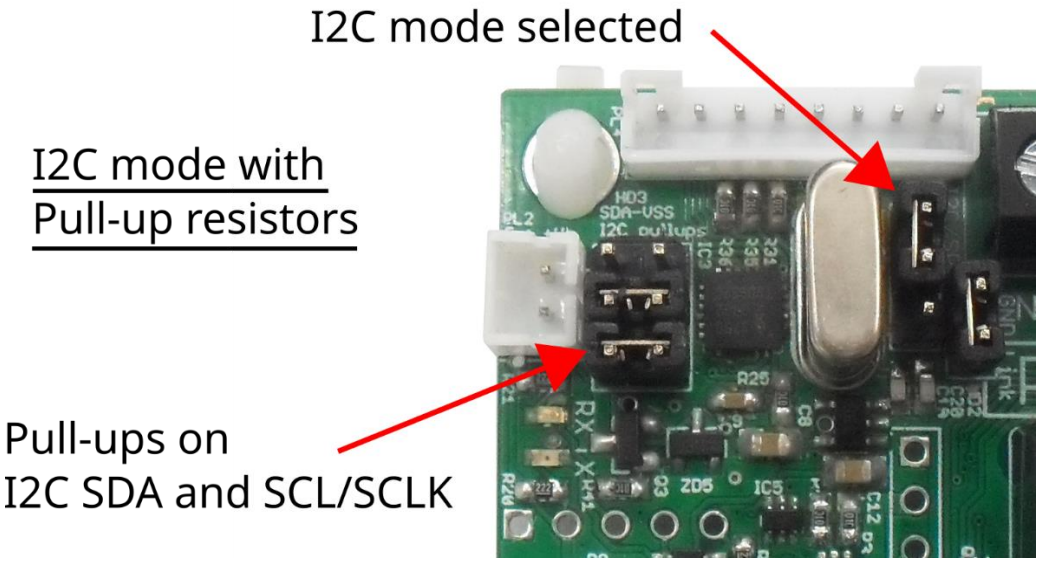
HD4 disables data from the USB port and is used when the USB is not powered.

I2C mode

The next diagram shows I2C mode without pull up resistors on SDA and SCLK/SCLK:

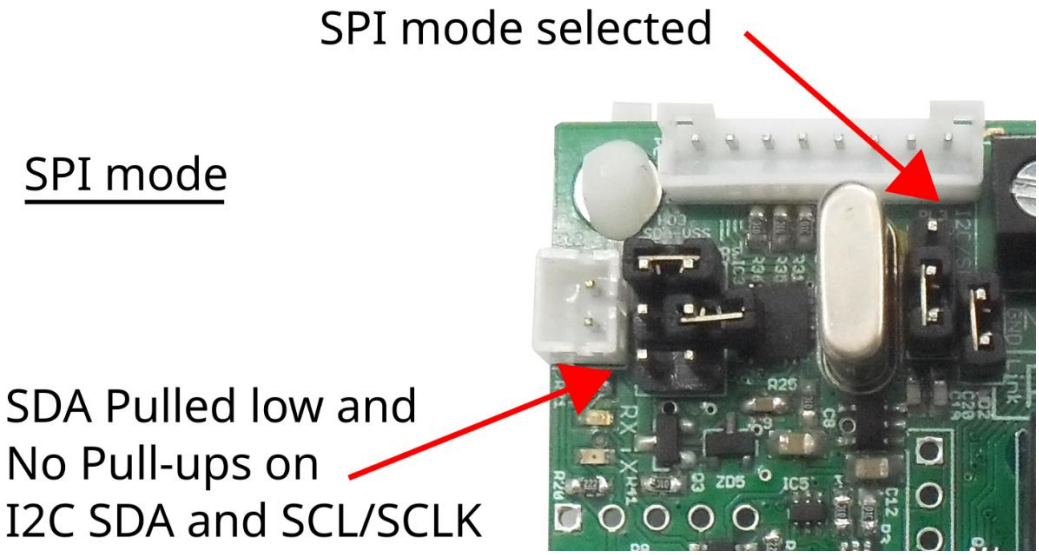


To enable the I2C pull up resistors on SDA and SCLK/SCLK the links on HD3 need to be moved to the positions shown in the next diagram:



SPI mode

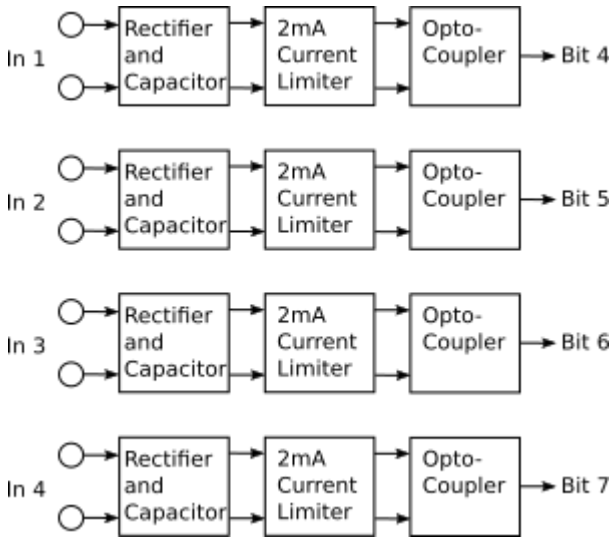
SPI mode is selected by setting the links as shown in the next diagram:



SDA has to be pulled low to ensure correct operation of the SC16IS750 I2C/SPI Interface chip in SPI mode.

Detailed electrical specification of inputs:

This card has opto-coupled, wide voltage range, current limited, AC or DC inputs. These inputs appear on the top 4 bits of port B. Port B has to be set to input to read these signals.



Each input pair is individually electrically isolated.

Input specification:

Parameter	Value	Notes
Max input voltage	60V DC or AC Peak	
Minimum detection voltage	3V DC or AC peak	
Maximum input to card GND voltage	60V DC or AC Peak	
Max input current	+/- 2mA	Current limited input.
Capacitance	100nF	

Order codes

USB-I2C-SPI-4VI4SRMx	The full specification board with all communications options, single part connector.
USB4VI4SRMx4	As above, but without the I2C/SPI.
I2C-SPI-4VI4SRMx	As USB4VI4SRMx4 above, but fitted with just the I2C and SPI communication circuit. No USB. (Note these will be built special to order)
-N	Add –N for No opto-isolation on the relays. (Note these will be built special to order)
-2H or -2V	Add -2H for 2 part, right angle or -2V for 2 part vertical connectors. (Note these will be built special to order)
-12V or -24V	Add -12V or -24V for 12V and 24V relays respectively. (Note these will be built special to order)
-NO-TB	Add -NO-TB for no terminal blocks fitted. (Note these will be built special to order)

NOTE.
Information in this document is believed to be accurate and reliable. However, EasyDAQ does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

Also, the USB-I2C-SPI-4VI4SRMx is quite a complex board so please contact EasyDAQ via or email admin@easydaq.co.uk if ever you need any help with these cards.

Please also contact us if you need a special variant of these boards.

Document versions

Version number	Date	Notes
Draft	20th March 2020	Draft document
1.0	27th January 2021	First release
1.1	24th September 2021	Added pictures of a CoverUSB4MxDIN housing a 12V relay version.
1.2	4th June 2022	This version. Updated power supply information and add I2C/SPI selection details for EDQ-006-6 PCBs.