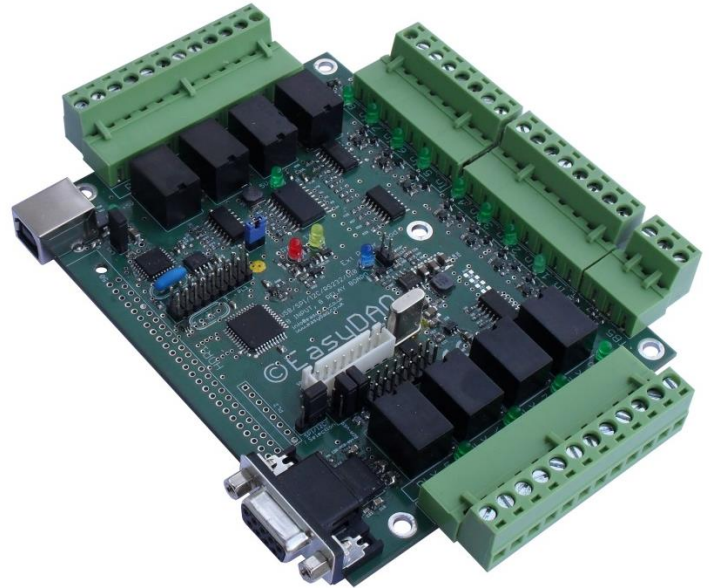


Product Datasheet 56

Features

- USB, I2C/SPI communications and RS232 communications
- Communications data activity LEDs
- 3 communications inputs can exist at the same time.
- 8 3A, 120V Opto-isolated signal relays + individual LEDs
- 8 Opto-isolated, wide input voltage sensors + individual LEDs. Max input 60V AC or DC.
- On board regulator with 7V to 24V input.
- Same board outline as USB8PR
- 2 part horizontal connectors
- 8 bit 5V logic DIO
- 8 bit 3V3 logic DIO on the I2C interface
- Optional bespoke software



Description

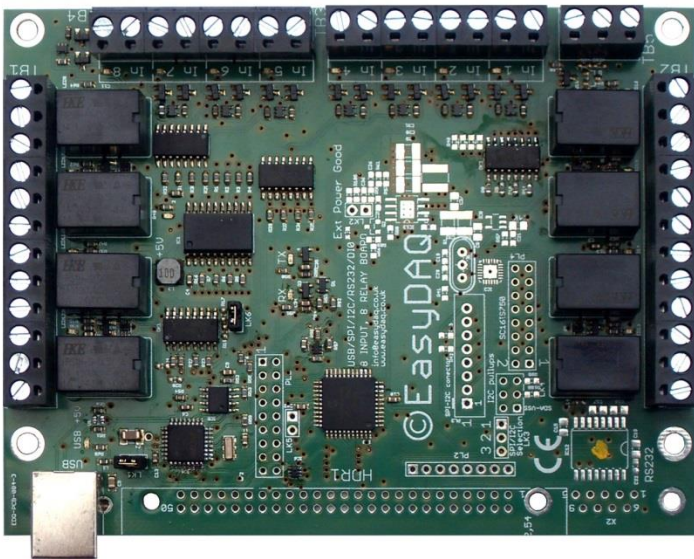
General purpose relay card with 8 opto-isolated, 120V, 3A signal relays and 8, 3V to 60V, AC/DC, opto-isolated voltage sensing inputs.

This card has 4 communication options:

- USB
- RS232
- I2C
- SPI

This card also has 3 powering options:

- USB power
- External 5V DC with reverse polarity protection.
- External 7 to 24V DC with reverse polarity protection and regulated 5V output on the external 5V terminal for powering your own hardware. 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 designed with future expansion in mind including access to 8, 5V DIO channels connected to the main controller and a further 8 3V3 DIO channels connected to the I2C/SPI port if this option is fitted. 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.

Product Datasheet 56

Specification

Power supply

- USB or external 5V powered (up to 8 relays @ 40mA per relay).
- External 7V – 24V switch-mode 6W maximum input power. Up to 500mA at 5V DC available to the user's hardware from the 5V terminal.

Relays

Rated voltage/current

Must operate/release voltage

Maximum contact ratings

Minimum recommended contact rating

Contact resistance

Operate/release time

Contact bounce period

Contact material

Operational life (min)

Contact arrangement

Control Interfaces

- USB 1, 2 or 3, Type B connector, hot pluggable.
- RS232 9 way D female.
- I2C or SPI on a JST XH 8pin header.

Operating temp range

-20 to +80°C

5VDC/42mA each

75%/10% of rated voltage

1A/120VAC or 1A 30VDC

5V @ 20mA

100mΩ max

5mS/5mS

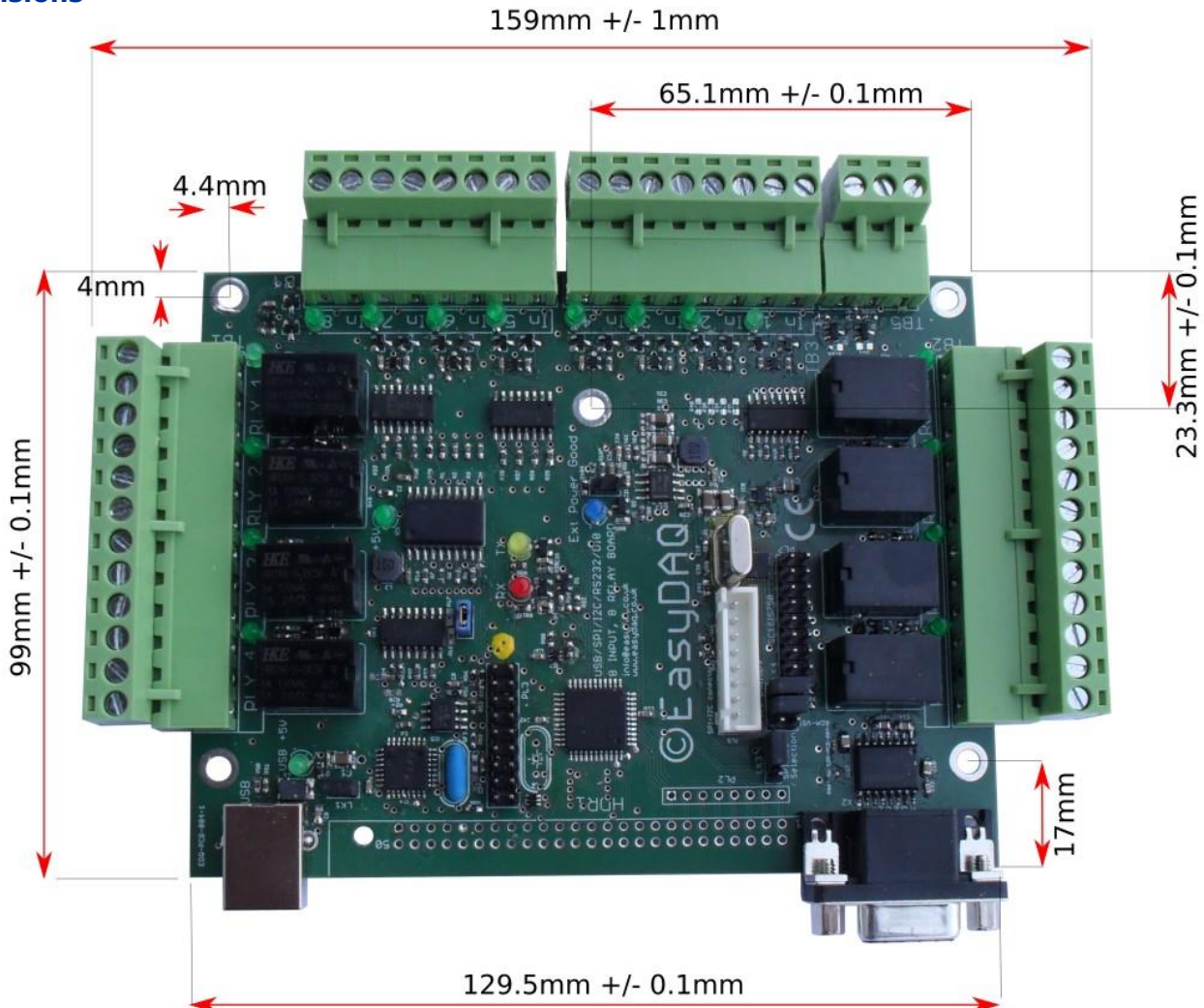
0.6mS operate/ 7.2mS release

AgAu

Mechanical 10⁷ / Electrical 10⁵

SPDT, Form C

Dimensions



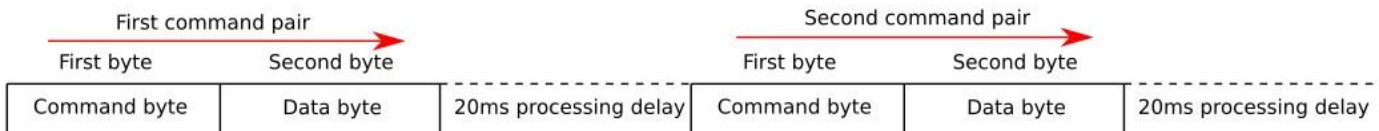
Product Datasheet 56

Detailed description and control

Command format

The card is commanded via simple single ASCII characters (+ status byte). I.e 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.

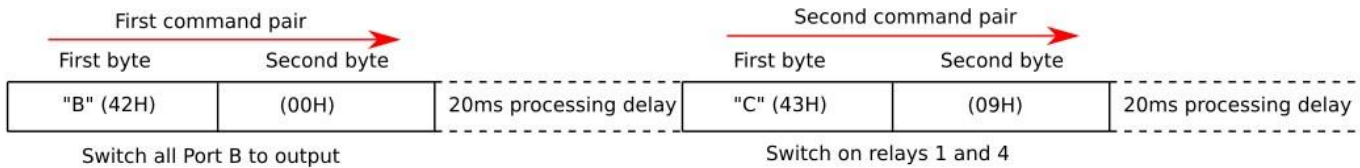


Port B (Channels 1-8) Relay commands:

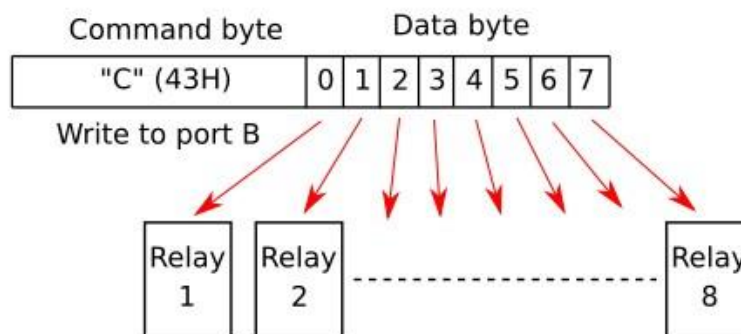
ASCII 'B' (42H), X Initialises the card (sets the port & channel I/O directions). Set direction of Port B, 1=Input, 0= output. (i.e. where X=10111111 (AFH) = sets bit 7 as an output, the rest as inputs).

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 to output for controlling the relays 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 C (Channels 9-16) Input commands:

The 8, 3V to 60V, AC/DC, opto-isolated voltage sensing inputs need to be enabled before use. This requires bit 2 of port "E" to be set to an output and pulled low. To do this use the following commands:

ASCII 'L' (4CH), 0x0B Initialises the port (sets the port & channel I/O directions) so that E2 is an output.

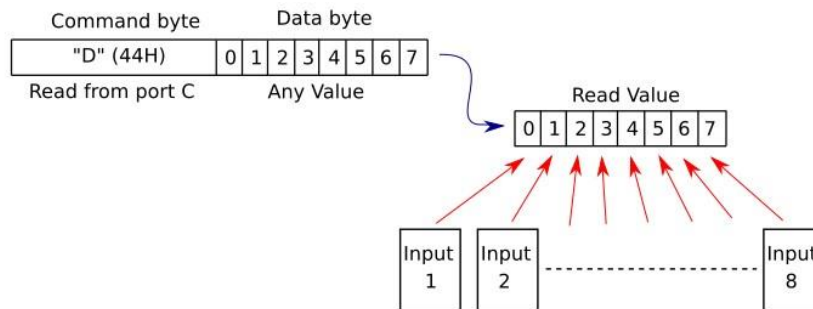
ASCII 'M' (4DH), 0x00 Ensures that the pin E2 is low which will enable the input buffer.

ASCII 'E' (45H), 0xFF Initialises port C as input.

ASCII 'D' (44H), X Read Port C data.

Product Datasheet 56

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.



Port D (Channels 17-24) DIO commands:

ASCII 'H' (48H), X Initialises the card (sets the port & channel I/O directions). Set directions of individual bits of Port D

ASCII 'J' (4AH), X Write data X to Port D (i.e. X=00000001 (01H), sets channel 17 to input and the rest to output).

ASCII 'G' (47H), X Read Port D. X is a dummy value. The data byte is returned after the dummy byte has been received.

Port E commands:

3 bits of Port E are controllable.

Bit 0 is free for user applications.

Bit 1 is connected to "Power Good" via a 100K resistor.

Bit 2 is used to enable the wide input buffer on port C

ASCII 'L' (4CH), X Initialises the port as inputs and outputs. Usually this is set to 0BH, Bits 0 and 1 inputs and bit 2 output to control the Port C buffer.

ASCII 'M' (4DH), X Write data X to Port E. Usually X = 0H to enable the Port C buffer.

ASCII 'K' (4BH), X Read Port E. X is a dummy value. The data byte is returned after the dummy byte has been received.

Summary of EasyDAQ relay card commands.

Letter	HEX	Function
A	41	Read Port B
B	42	Set direction of Port B
C	43	Write data to Port B
D	44	Read Port C
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



Product Datasheet 56

Serial Port settings

For the RS232 and the 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 SC16IS750 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
```

Product Datasheet 56

```
#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

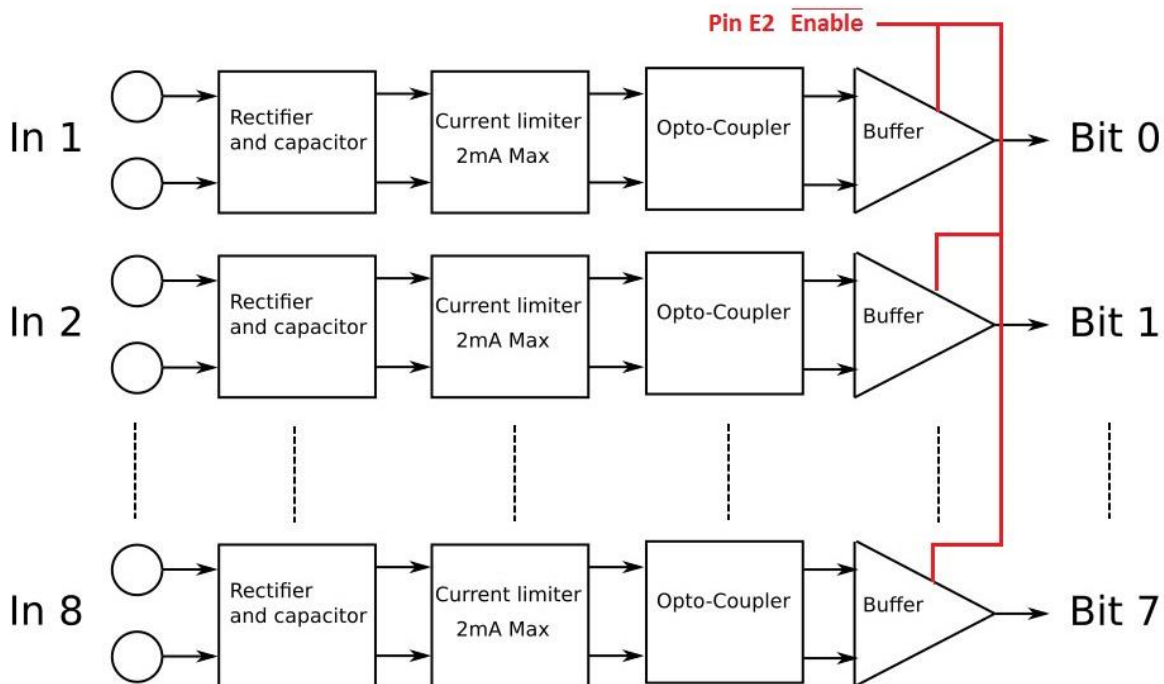
  putByte(LCR,0xBF);          // Access EFR register
  putByte(EFR,0x10);          // Enable Enhanced functions
  putByte(LCR,0x03);          // Setup LCR for normal operation: 8 bit data, 1 stop bits, no parity
  putByte(FCR,0x06);          // Reset FIFOs
  putByte(FCR,0x01);          // Enable FIFOs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

Detailed electrical specification of inputs:

This card has opto-coupled, wide voltage range, current limited, AC or DC inputs. To use these inputs the buffer must be enabled by setting Port E, Pin 2 to output and taking this output low.



Each input pair is individually electrically isolated.



USB-RS232-I2C-SPI-8VI8SRMx-2

Low cost Data Acquisition & Control products

8 Signal Relay, 8 Wide Voltage Input, Multi-Protocol card

Product Datasheet 56

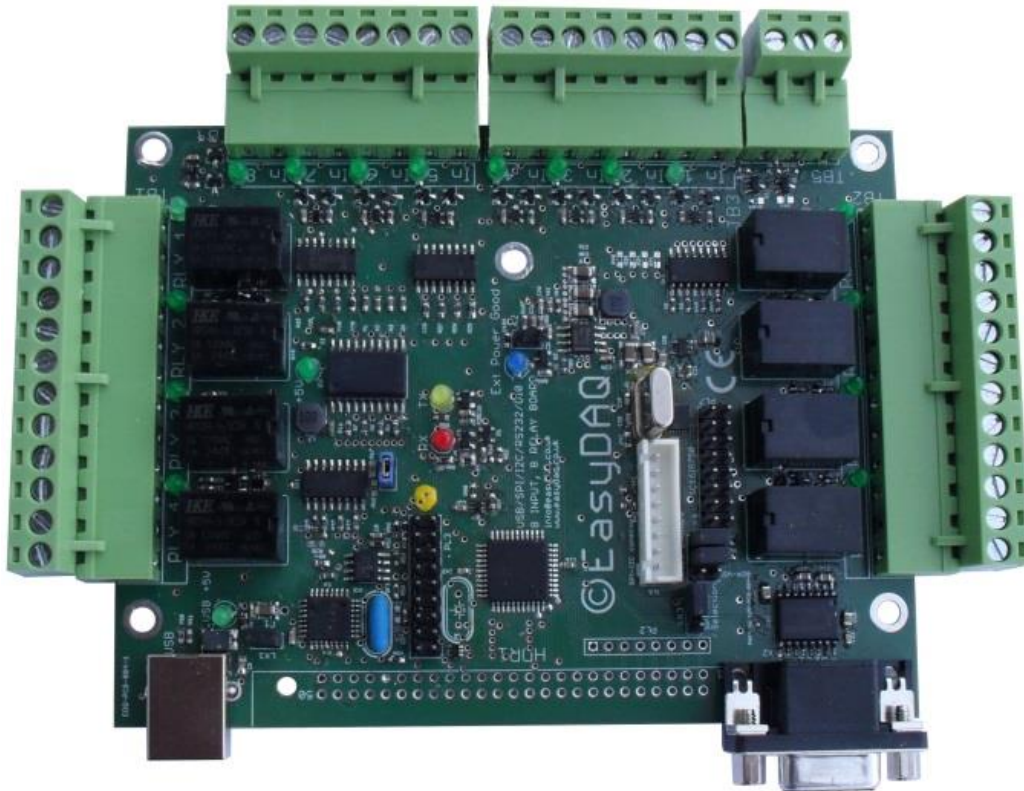
Input specification

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	

Product Datasheet 56

Order codes

USB-RS232-I2C-SPI-8VI8SRMx-2	The full specification board with all options
USB-RS232-I2C-SPI-8VI8SRMx	As above with single part connectors.
USB-8VI8SRMx-2	As above, but without the RS232, I2C/SPI.
USB-8VI8SRMx	As above with single part connectors.
I2C/SPI-8VI8SRMx-2	As USB-8VI8SRMx-2 above, but fitted with just the I2C and SPI communication circuit. No USB or RS232.
I2C/SPI-8VI8SRMx	As above with single part connectors.
RS232-8VI8SRMx-2	As USB-8VI8SRMx-2 above, but fitted with just the I2C and SPI communication circuit. No USB or RS232.
RS232-8VI8SRMx	As above with single part connectors.
-P	Add -P if no on board PSU is required. (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)



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.

Document versions

Version	Date	Notes
Draft	20 th March 2020	Draft document
1.0	27 th January 2021	Released
1.1	9 th February 2021	Corrections and updates
1.2	16 th December 2021	Added more information about port E