# RS485 using the EB062 E-block

*by Sean King, March 2011*

## Abstract

**RS485 is one of the most versatile communication standards and is widely used in data acquisition and control applications where multiple nodes communicate with each other. In this article Sean explains the fundamentals of the protocol and using Flowcode creates a program for use with the brand new RS485 E-block.**

## Requirements

**Software:**

- Professional licence of Flowcode v3 or v4 for any chip variant.

**Hardware:**

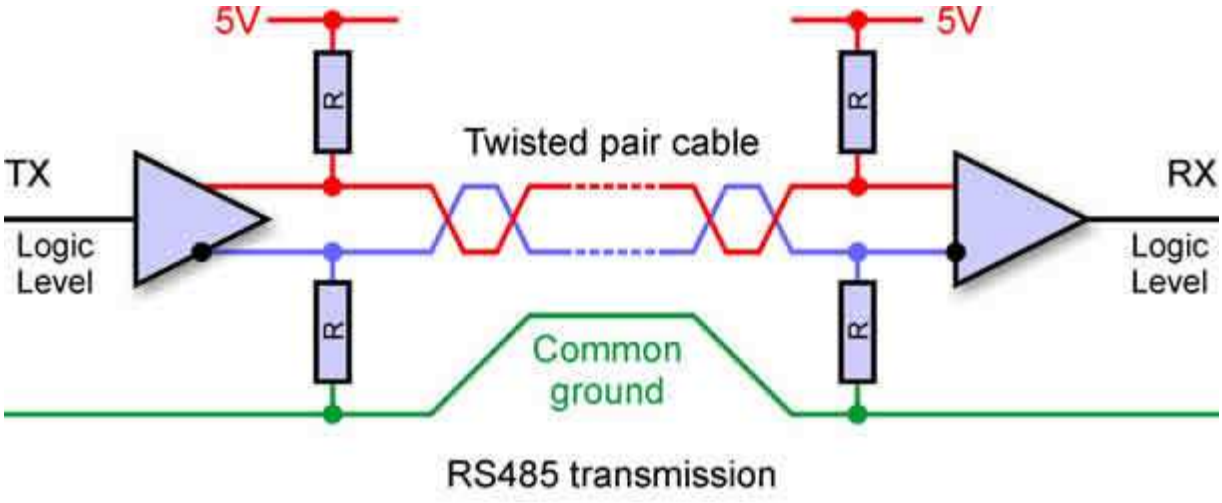
- RS485 E-block

## Introduction

In a previous article we discussed some of the features of the RS485 communications standard, developing a simple network protocol using the Flowcode RS232 (UART) component in 9-bit data mode. The original article was based on prototype hardware, but we now have the EB062 RS485 E-block which is capable of supporting this, and many other applications.


## Electrical specifications

RS485, and the similar RS422, are communication bus systems using two wire differential transmission of each signal. Differential transmission improves the signal's immunity to electrical noise, at the expense of the requirement for an extra conductor for each signal - usually in a twisted pair configuration. Specifications only define the electrical properties of the system (ISO physical layer). There are no specific communications protocols, though many have been adopted.




RS485 is often used for serial data communications. The lack of a specific protocol means that several pre-existing ones have been adopted (with some adaptations to allow additional benefits of RS485 to be included), and new ones developed for specific applications.

## EB062 RS485 E-Block

The EB062 has two communication channels (CH1 and CH2) and two independent RS485 transceivers. Each communication channel can be connected to the transmit or receive pin of either transceiver using a block of channel selection jumper links.

Jumper links are also available to configure each transceiver to be either a transmitter (TX), a receiver (RX), or have programmable direction (IO) using optional direction control signals (DDR1 and DDR2).

When a transceiver is configured for I0 direction control, a high level on the associated DDR line enables the transmitter. A low level disables the transmitter (freeing the signal lines for other transmitters to use) and enables the receiver.

## Common protocols

RS485 can often be used to directly replace RS232 links - extending the transmission range and baud rate available. Many commercial products are available to perform this function.
An extension of RS232, available in UARTs of many microcontrollers, is the transmission of a 9th data bit. This is used in many RS485 systems to differentiate between two types of transmissions:

9th bit = 1, 8 bits = node address
9th bit = 0, 8 bits = data

Unlike RS232, which can only allow communication between two devices (officially), RS485 can allow communications between a small network of devices (maximum 32) using a multi-drop bus. Each node on the bus must have a unique address to allow information to be targeted at it. The bus transmitters in each device can be disabled, allowing other devices to transmit on the bus without causing conflict. The use of 9 data bits can allow packets of bytes to be transmitted. The first byte can be highlighted as the address of the intended recipient node by setting its 9th bit.

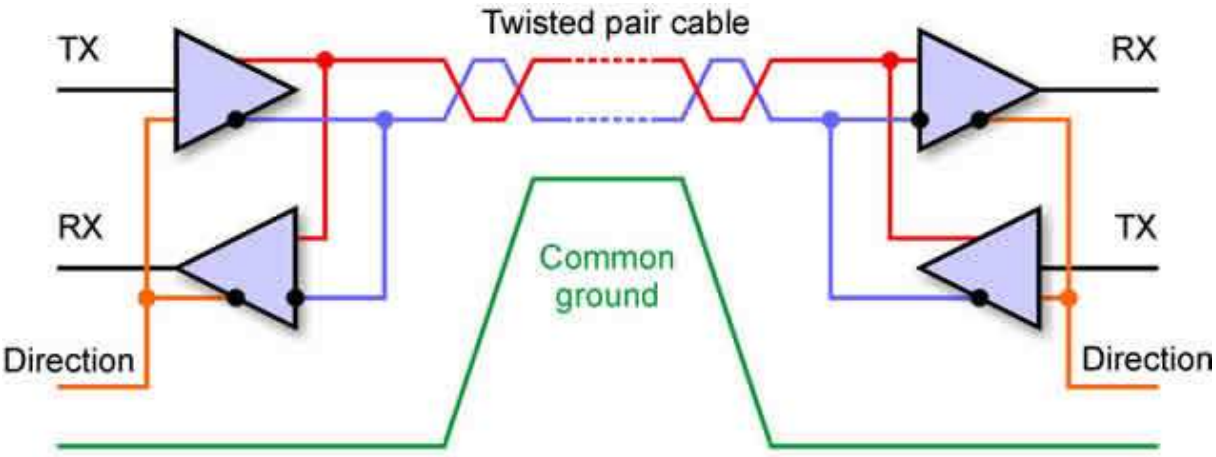
For simplicity, many RS485 networks have a single master node and multiple slave nodes. The master node initiates all communications and indicates the slave node being accessed by including its node addresses in the message.
RS485 does not have the in-built message collision detection and correction available in bus systems like CAN, but there are techniques to allow support for multi-master networks and unsolicited messages.

Common configurations and applications include:

**Half duplex (Peer-to-Peer or Network)**

Uses Transceiver1 only to transmit CH1 and receive CH2 using DDR1 for direction control

**J16 CH1 = Transceiver1 TX**
**J17 CH2 = Transceiver1 RX**
**J6 Transceiver1 direction control = IO**
**Transceiver1 direction control signal = DDR1**




Half-duplex RS485 transmission

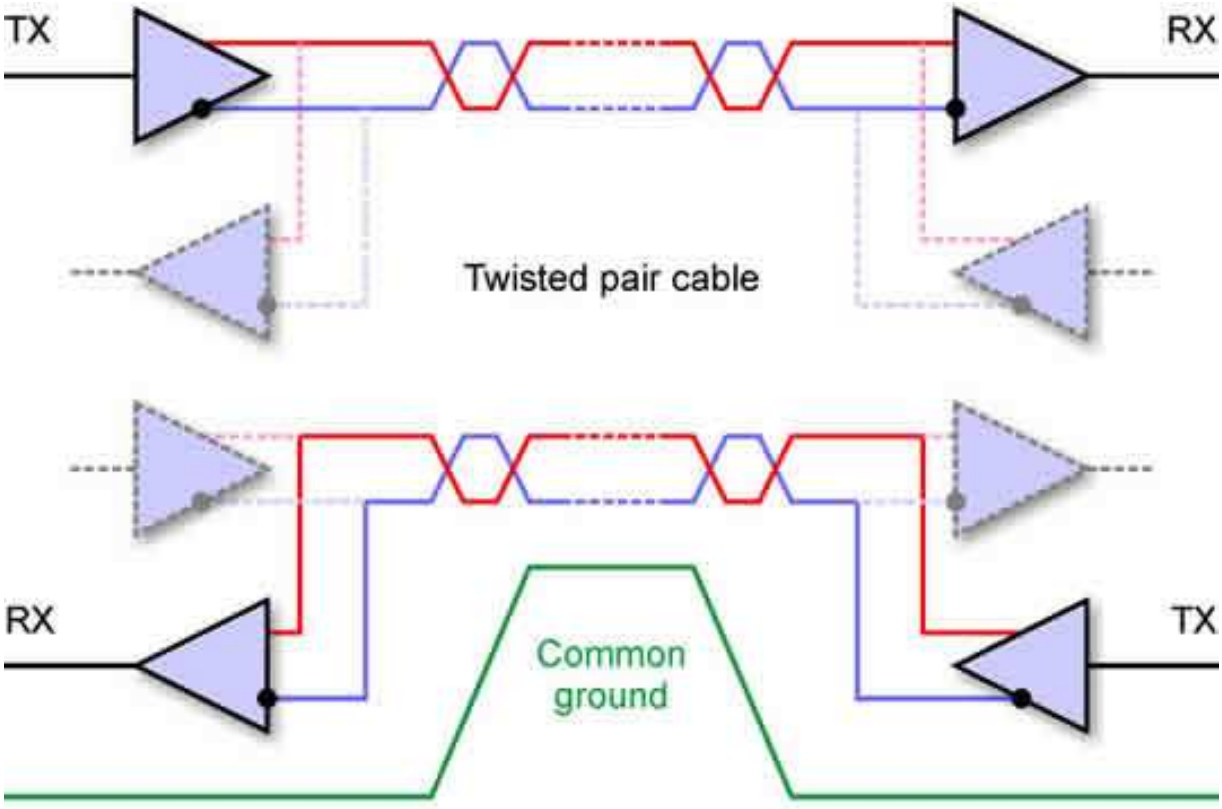
This configuration can be used as the basis for several types of networks. Data transmitted by one node is available to all other nodes. The network protocol must ensure that no more than one node is configured as a transmitter at any time.

This mode requires one twisted-pair cable for signalling

**Full duplex (Peer-to-Peer or Network master )**

Uses Transceiver1 to transmit CH1 and Transceiver2 to receive CH2

**J16 CH1 = Transceiver1 TX**
**J19 CH2 = Transceiver2 RX**
**J6 Transceiver1 direction control = TX**
**J10 Transceiver2 direction control = RX**




Full-duplex RS485 transmission

This configuration can be used as a direct replacement for a standard RS232 connection, allowing greater distances and higher Baud rates. It can also be used as the master node in a simple network..

This mode requires two twisted-pair cables for signalling.

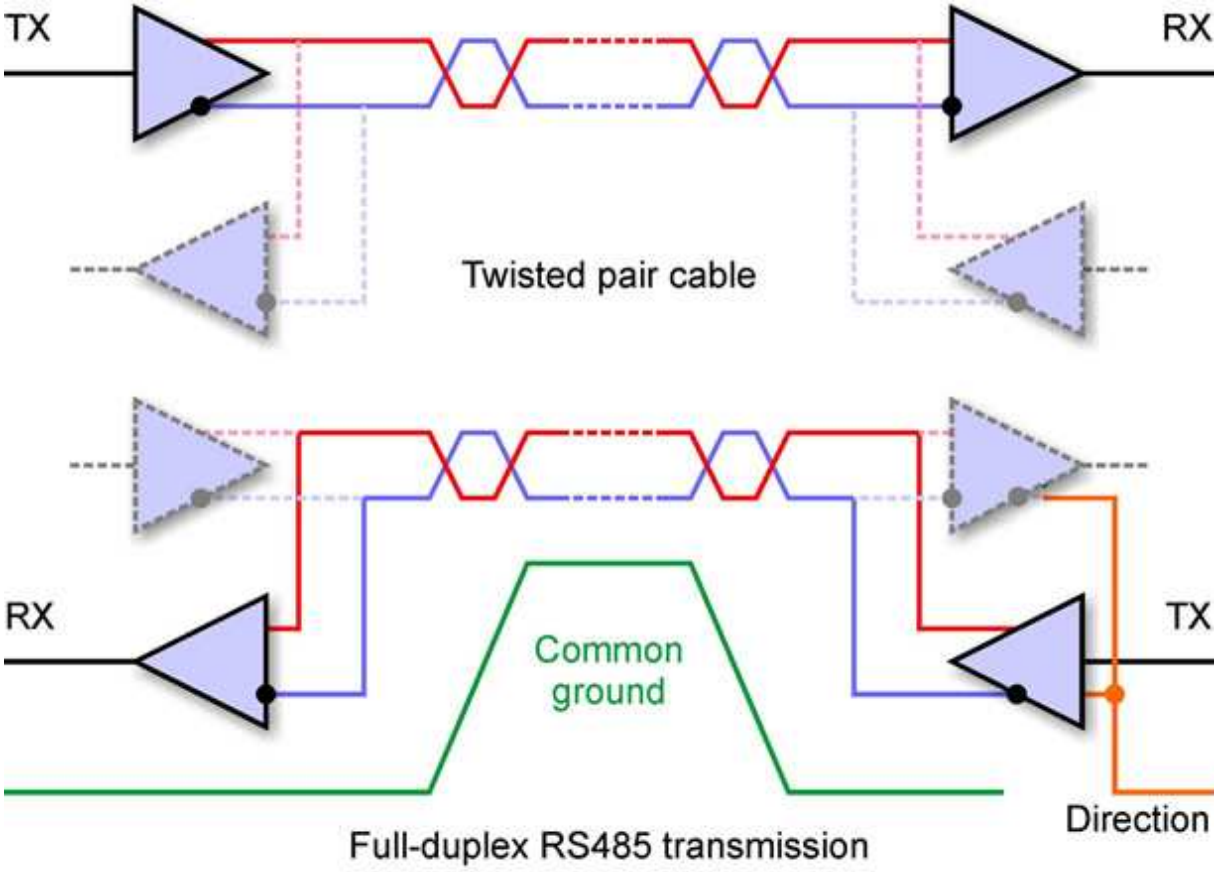
In a Peer-to-Peer system the transmitter in each node is connected only to the receiver in the other node, so can be enabled at all times.

As a network master the transmitter connected to several slave node receivers, so can be enabled at all times. The receiver is connected to several slave node transmitters (see Full duplex (Network slave)).

**Full duplex (Network slave)**

Uses Transceiver1 to transmit CH1 and Transceiver2 to receive CH2

**J16 CH1 = Transceiver1 TX**
**J19 CH2 = Transceiver2 RX**
**J6 Transceiver1 direction control = IO**
**J10 Transceiver2 direction control = RX**
**Transceiver1 direction control signal = DDR1**




Full-duplex RS485 transmission

This configuration can be used by slave nodes in a multi-node network.

This mode requires two twisted-pair cables for signalling. Several slave node transmitters may be connected to the master node receiver, so each must release control of the signal lines when not transmitting - by using the DDR1 signal to turn the transmitter into a receiver.

**Dual output**

Uses Transceiver1 to transmit CH1 and Transceiver2 to transmit CH2

**J16 CH1 = Transceiver1 TX**
**J19 CH2 = Transceiver2 TX**
**J6 Transceiver1 direction control = TX**
**J10 Transceiver2 direction control = TX**

Many industrial stepper motor drives use RS485 receivers for step and direction control. In this case the CH1 and CH2 would be connected to logic/clock outputs rather than UART pins.

**Dual input**

Uses Transceiver1 to receive CH1 and Transceiver2 to receive CH2

**J16 CH1 = Transceiver1 RX**
**J19 CH2 = Transceiver2 RX**
**J6 Transceiver1 direction control = RX**
**J10 Transceiver2 direction control = RX**

Many industrial incremental encoders use RS485 transmitters for the quadrature position signals. In this case the CH1 and CH2 would be connected to logic/counter inputs rather than UART pins.

## Termination

Long cable runs must be terminated with appropriate resistors to prevent signal attenuation and reflections. RS485 connections should be made with twisted-pair cable, with a characteristic impedance of 60 ohms. When several nodes are connected together, the cables must be routed to form a chain, rather than a star or loop. The chain must have two distinct end nodes. The end nodes only must provide termination resistors of 120 ohms each, connected between the signal lines.

The EB062 provides termination resistors for both transceivers (selected by jumpers J4 and J14).

An alternative termination option also uses bias resistors connected to 0V and 5V. These are also available for both transceivers on the EB062 and selected using jumpers J5 and J15.

## Example programs

The two example programs attached to this article represent a Master and one Slave node of a simple, half-duplex network. Other slave nodes can be added by changing the value of the MyAddress variable in each copy.

The Master node initiates all communications.

The Flowcode RS232 components are configured for 9-bit data mode, with the most significant bit ( Bit 8 ) indicating whether a value represents Control/Address (=1) or Data (=0).

The data rate is set to 38400 Baud. This is relatively high, but should be achievable over long distances using suitable cable and termination.

Bit 7 of a Control/Address message indicates whether data is to be written from the Master to the Slave node (=0) or read by the Master node from the Slave (=1).

The network can be connected using pairs of signal wires connected to the Transceiver 1 terminals of each EB062, plus a common 0V. Suitable twisted pair cable should be used for long distances at high Baud rates.

Connections should be daisy-chained, with the termination links connected at the two end nodes.

Configuration jumper settings for all boards should be for the Half duplex option detailed above.

The main connection jumper on each EB062 should be on option B

## Further reading

Below are some links to other resources and articles on related subjects, and technical documentation relating to the hardware used for this project...

| | |
|---|---|
| Flowcode: | http://www.matrixmultimedia.com/flowcode.php |
| Learning Centre: | http://www.matrixmultimedia.com/lc_index.php |
| User Forums: | http://www.matrixmultimedia.com/mmforums |
| Product Support: | http://www.matrixmultimedia.com/sup_menu.php |