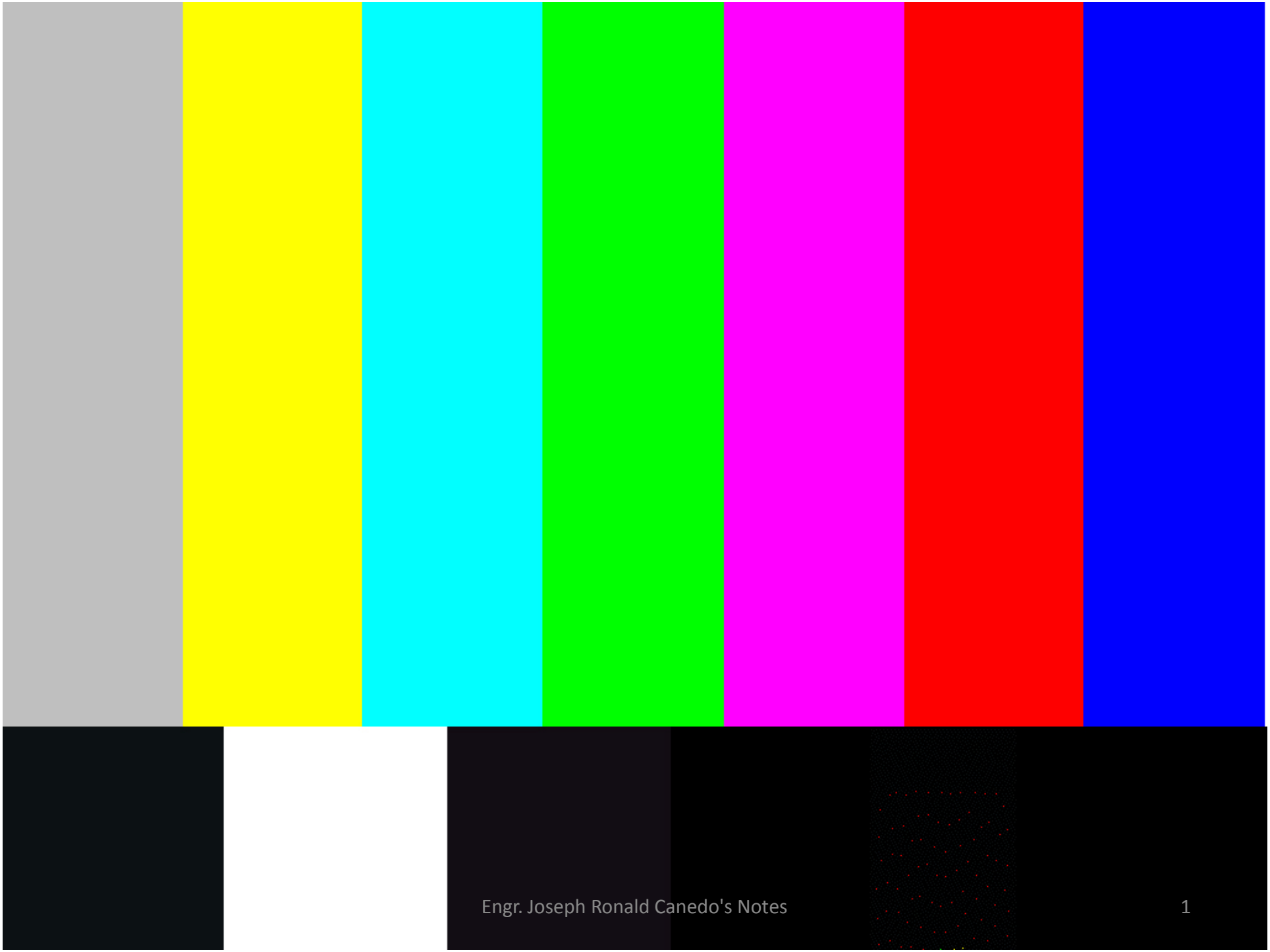


Serial Ports

by:
Joseph Ronald Cañedo

UARTs

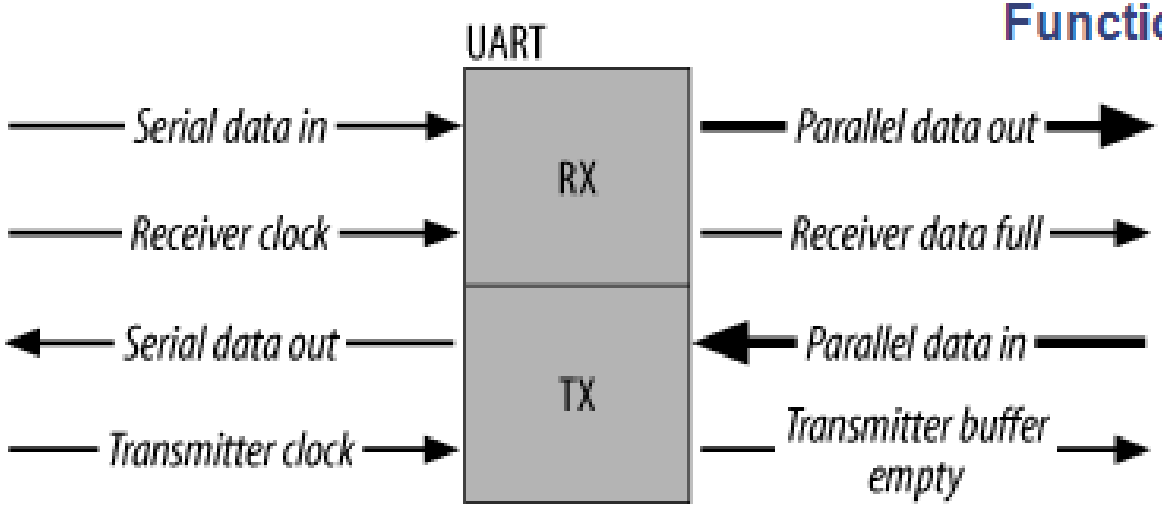
- Serial I/O involves the transfer of data over a single wire for each direction.
- All serial interfaces convert parallel data to a serial bit stream, and vice versa.
- Serial communication is employed when it is not practical, either in physical or cost terms, to move data in parallel between systems.
- Such serial communication may be between a computer and a terminal or printer, the infrared beamings of a Palm computer or remote control, or, in more advanced forms, high-speed network communication such as Ethernet.
- For embedded computers, a simple serial interface is the easiest and cheapest way to connect to a host computer, either as part of the application or merely for debugging purposes.

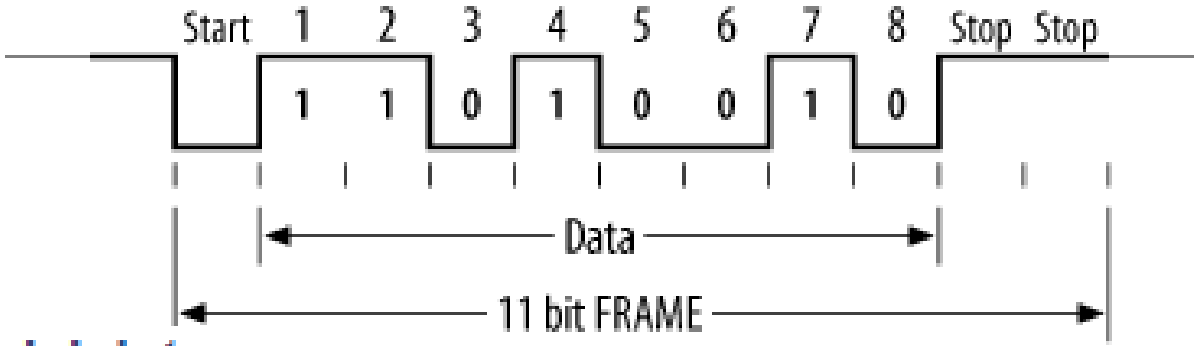
UARTs

- The simplest form of serial interface is that of the Universal Asynchronous Receiver Transmitter (UART).
- UARTs are also sometimes called Asynchronous Communication Interface Adapters (ACIAs).
- They are termed asynchronous because no clock is transmitted with the serial data. The receiver must lock onto the data and detect individual bits without the luxury of a clock for synchronization.

UARTs

Functional diagram of a UART





Asynchronous serial data

Error Detection

- In any transfer of data over a potentially noisy medium (such as a serial cable), the possibility of errors exists.
- To detect such errors, many serial systems implement parity as a simple check for the validity of the data.
- The parity bit of a byte to be transmitted is calculated by the sending UART and included with the byte as part of the transmission.
- The receiving UART also calculates the parity bit for the byte and compares this against the parity bit received.
- If they match, the receiver assumes that everything is fine. If they do not, the receiver then knows that something went amiss and that an error exists.

Error Detection

- There are several types of parity, the main two being even parity and odd parity.
- In any byte of data, there is either an even number of "1" bits or an odd number of "1" bits. An extra bit (the parity bit) is added to the byte to make the number of "1" bits even (even parity) or odd (odd parity).
- A good example of this is the *Cyclic Redundancy Check (CRC) algorithm*.

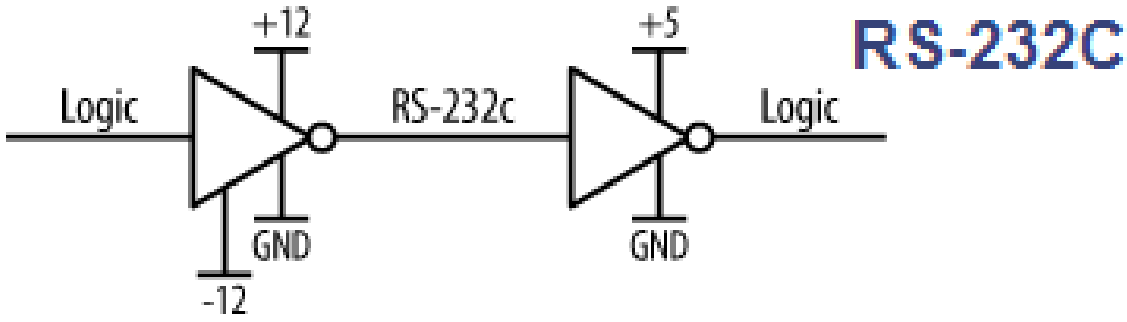
RS-232C

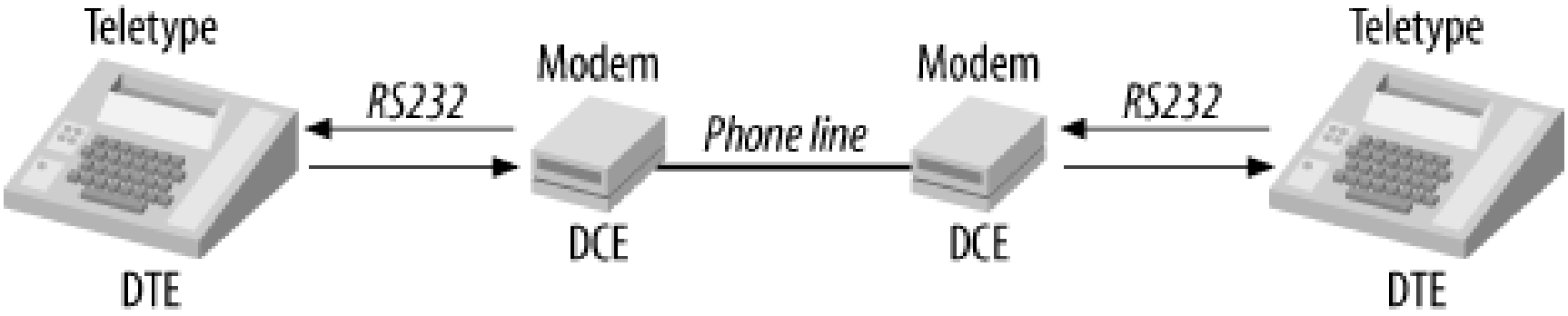
- RS-232C is a serial communication interface standard that has been in use, in one form or another, since the 1960s.
- RS-232C is used for interfacing serial devices over cable lengths of up to 25 meters and at data rates of up to 38.4 kbps.
- You can use it to connect to other computers, modems, and even old terminals (useful tools for monitoring status messages during debugging).
- In days of old, printers, plotters, and a host of other devices came with RS-232C interfaces. With the need to transfer large amounts of data rapidly, RS-232C is being supplanted as a connection standard by high-speed networks, such as Ethernet. However, it can still be a useful and (importantly) simple connection tool for your embedded system.

RS-232C

- RS-232C is unbalanced, meaning that the voltage level of a data bit being transmitted is referenced to local ground.
- A logic high for RS-232C is a signal voltage in the range of -5 to -15 V (typically -12 V), and a logic low is between +5 and +15 V (typically +12 V).
- So, just to make that clear, an RS-232C high is a negative voltage, and a low is a positive voltage, unlike the rest of your computer's logic.

RS-232C





Original use of RS232: connecting teletypes to modems

RS-232C

Signal	Function	25-pin	9-pin	Direction
Tx	Transmitted Data	2	3	From DTE to DCE
Rx	Received Data	3	2	To DTE from DCE
RTS	Request To Send	4	7	From DTE to DCE
CTS	Clear To Send	5	8	To DTE from DCE
DTR	Data Terminal Ready	20	4	From DTE to DCE
DSR	Data Set Ready	6	6	To DTE from DCE
DCD	Data Carrier Detect	8	1	To DTE from DCE
RI	Ring Indicator	22	9	To DTE from DCE
FG	Frame Ground (chassis)	1	-	Common
SG	Signal Ground	7	5	Common

RS-232C

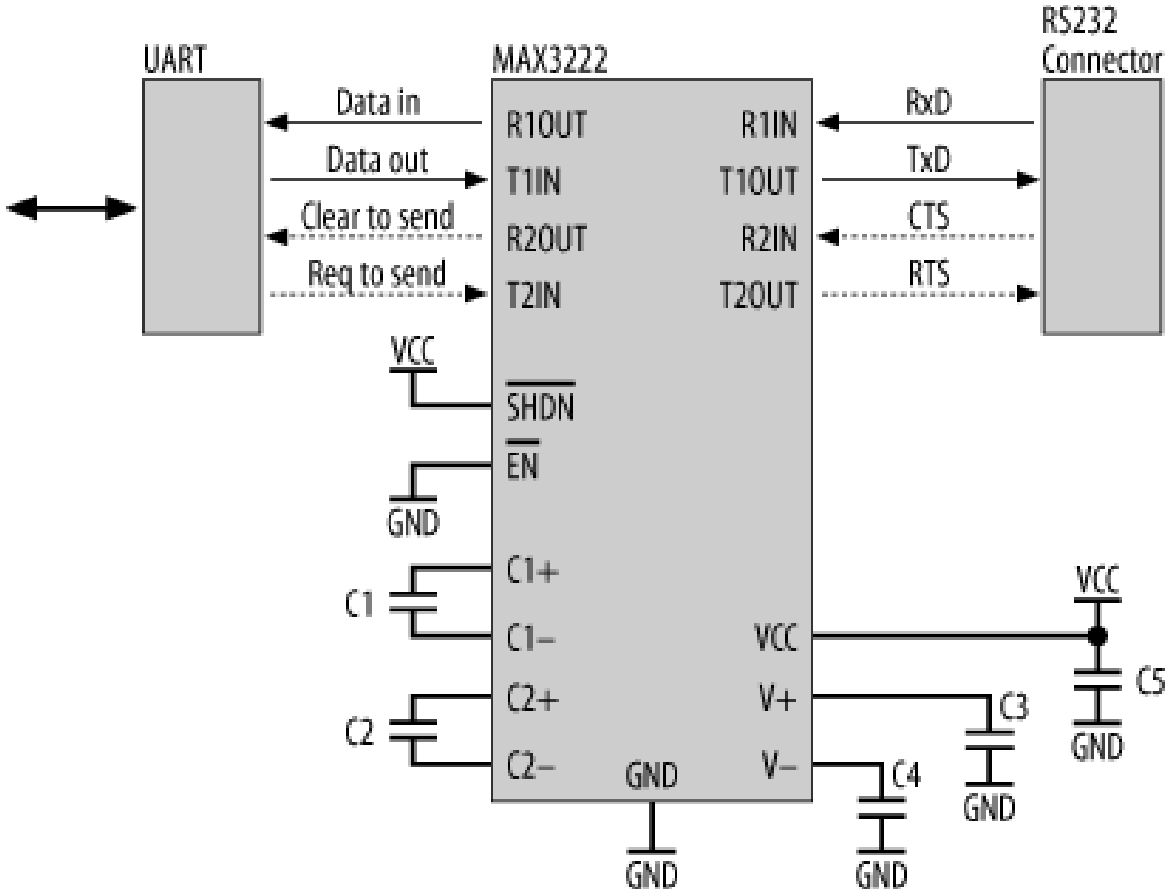
- **Shake Hands**
  - When two remote systems are communicating serially, there needs to be some way to prevent the transmitter from sending new data before the receiver has had a chance to process the old data.
  - This process is known as handshaking, or flow control. The way it works is simple. After transmitting a byte (or data packet), the transmitter will not send again until it has been given confirmation that the receiver is ready.
  - There are three forms of handshaking: hardware, software, and none.

RS-232C

- **Shake Hands**
  - Hardware handshaking in RS-232C uses two signals, RTS (Request To Send) and CTS (Clear To Send). When the transmitter wishes to send, it asserts RTS, indicating to the receiver that there is pending data. The receiver asserts CTS when it is ready, indicating to the transmitter that it may send. In this way, the flow of data is limited to the rate at which it may be processed.
  - Software handshaking, also known as XON/XOFF, is used where it is not possible to have hardware handshaking between the transmitter and receiver, such as when the transmission occurs over a phone line. Software handshaking chooses two characters to represent a request to "suspend transmission," and a "clear to resume."

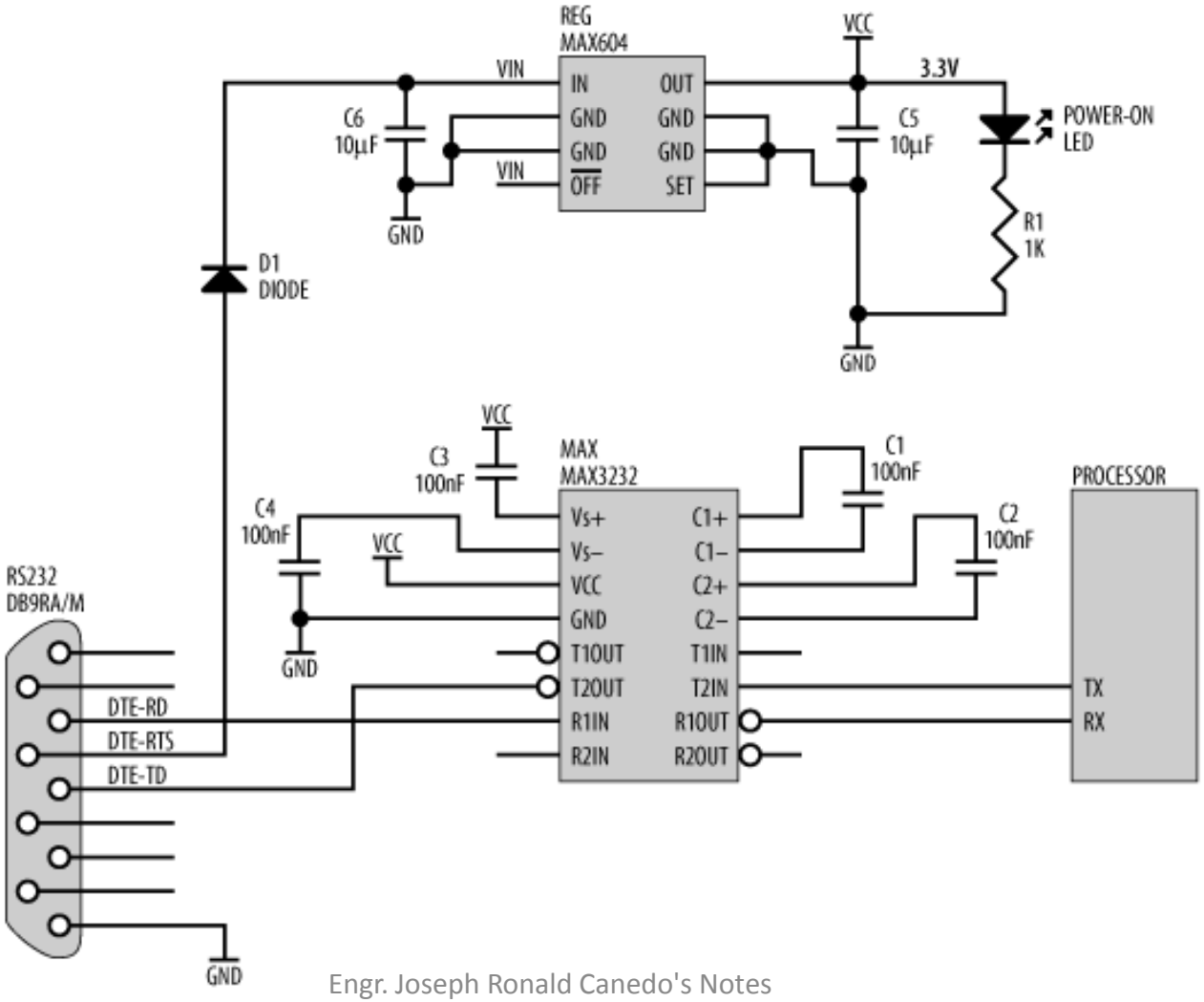
RS-232C

- **Implementing an RS-232C Interface**



RS-232C

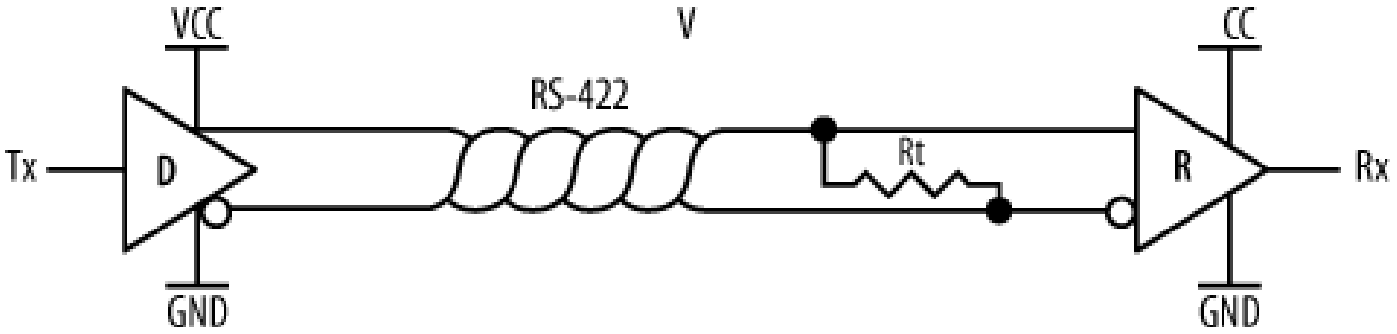
- **Using a Serial Port as a Power Supply**

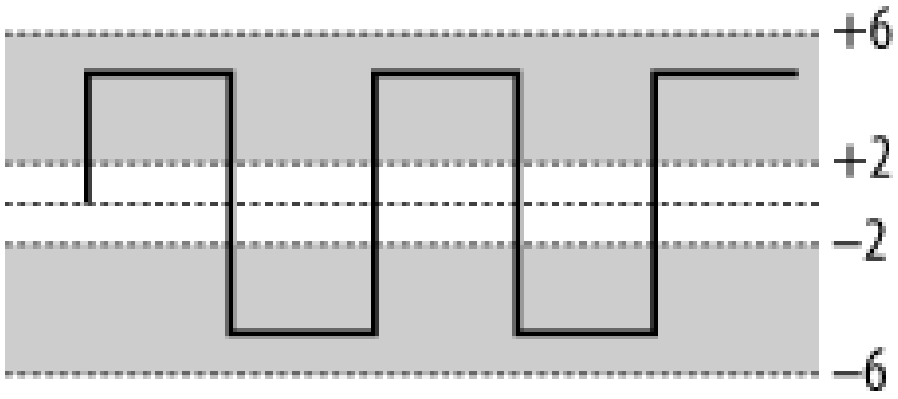


RS-422

- Unlike RS-232C, which is referenced to local ground, RS-422 uses the difference between two lines, known as a twisted pair or a differential pair, to represent the logic level.
- Thus, RS-422 is a balanced transmission, or, in other words, it is not referenced to local ground.
- Any noise or interference will affect both wires of the twisted pair, but the difference between them will be less affected.
- This is known as common-mode rejection. RS-422 can therefore carry data over longer distances and at higher rates with greater noise immunity than RS-232C. RS-422 can support data transmission over cable lengths of up to 1,200 meters (approximately 4,000 feet).

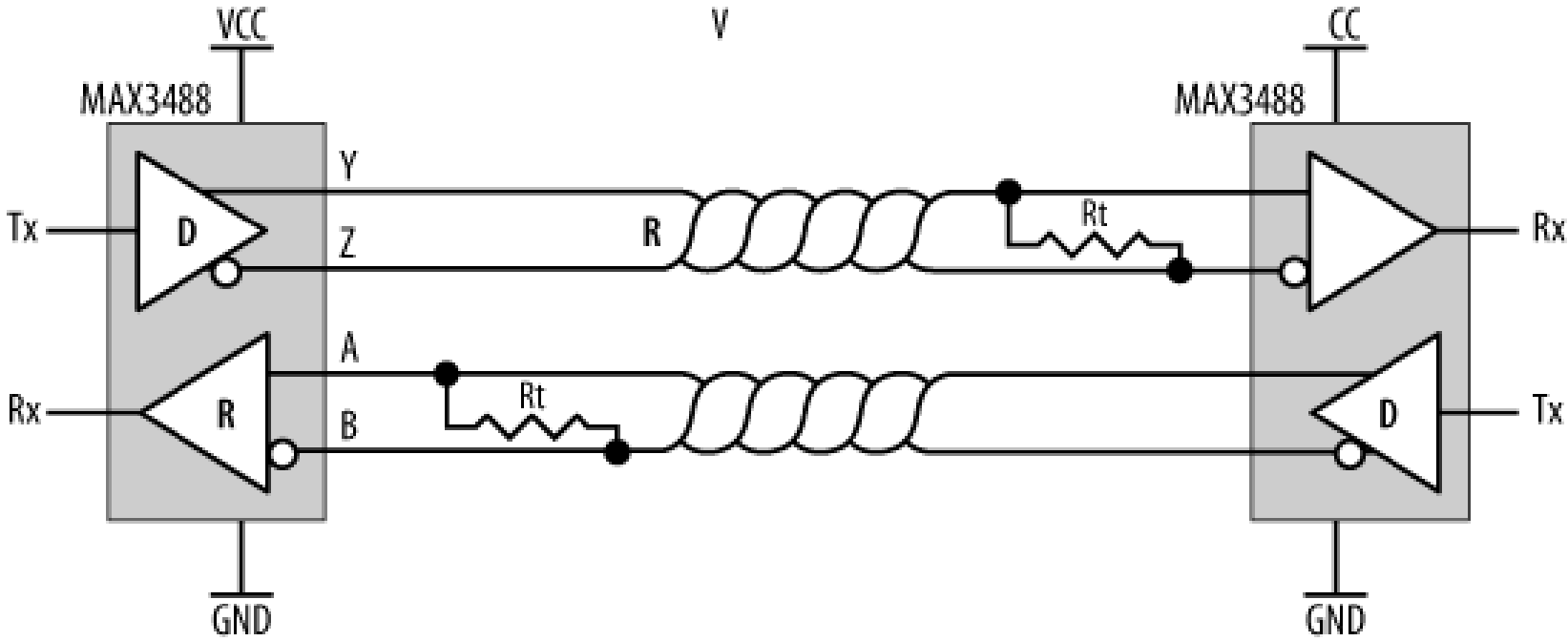
RS-422





RS-422

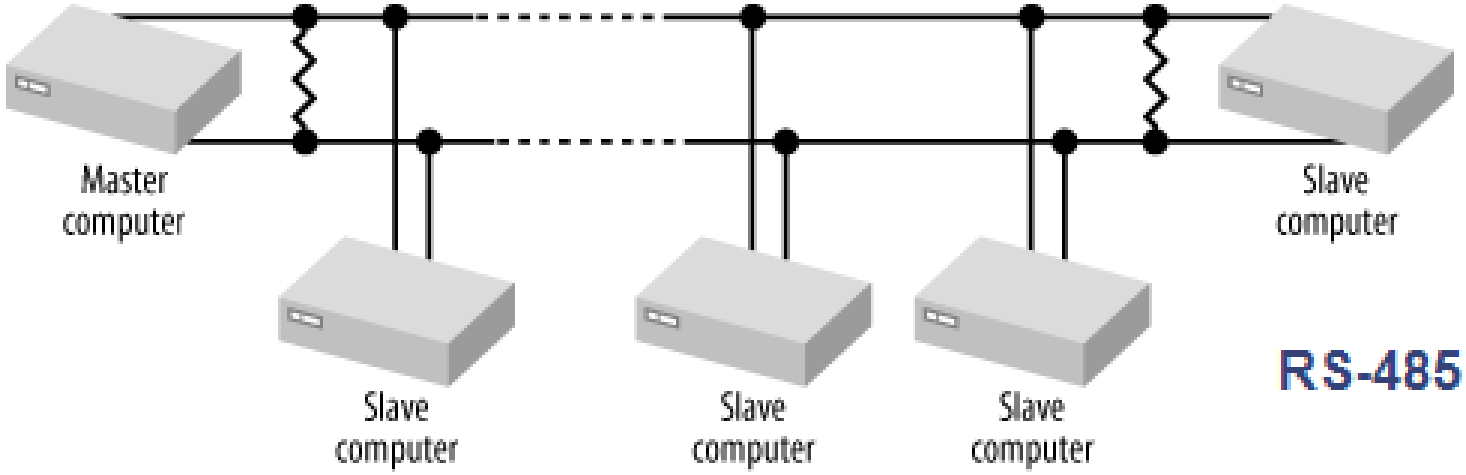
- **Bidirectional RS-422 interface**



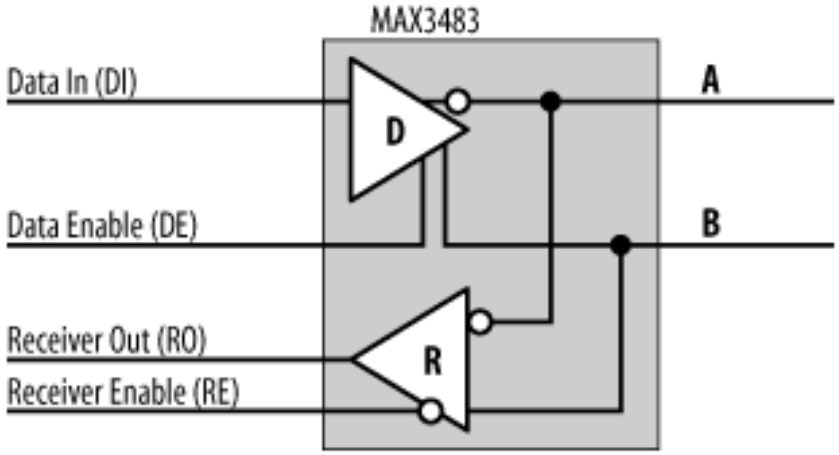
RS-485

- RS-485 is a variation on RS-422 that is commonly used for low-cost networking and in many industrial applications. It is one of the simplest and easiest networks to implement. It allows multiple systems (nodes) to exchange data over a single twisted pair.
- RS-485 is based on a master-slave architecture. All transactions are initiated by the master, and a slave will transmit only when specifically instructed to do so. There are many different protocols that run over RS-485, and often people will do their own thing and create a protocol specific to the application at hand.

RS-485



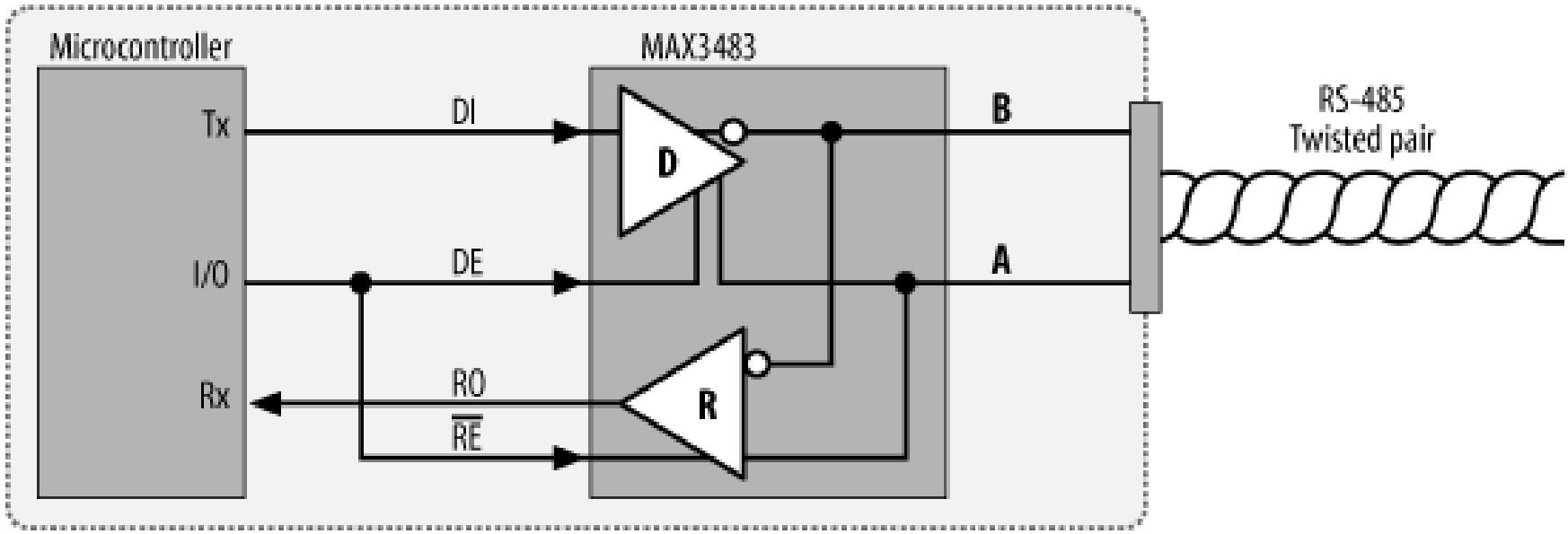
RS-485 network



RS-485 transceiver

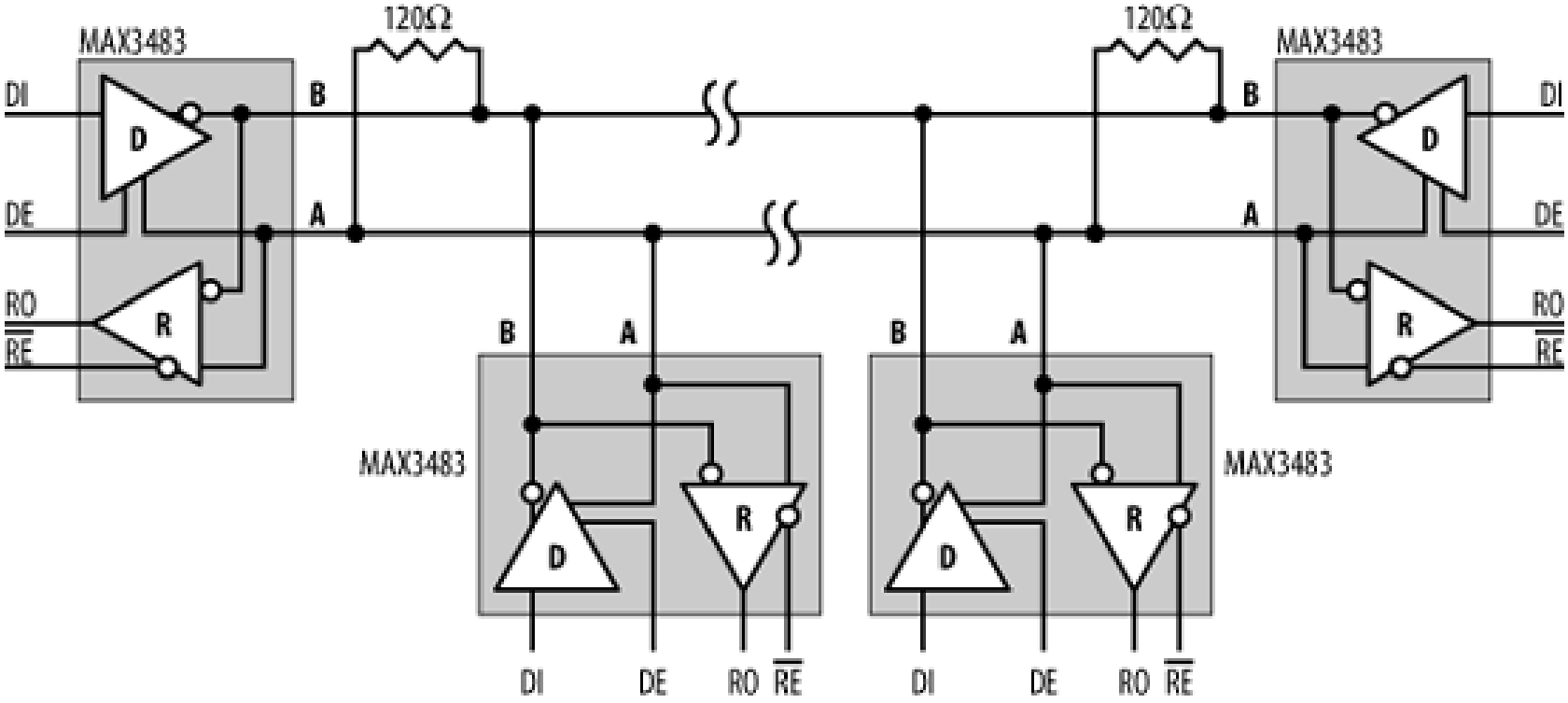
RS-485

- **Connecting a MAX3483 to a microcontroller**



RS-485

- **Half-duplex RS-485**



RS-485

- **Full-duplex RS-485**