香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

# ECE3080 Microprocessors and Computer Systems

## Communication Interface


**Instructor：Tin Lun LAM**

E-mail: tllam@cuhk.edu.cn

URL: https://myweb.cuhk.edu.cn/tllam

# STM32F103xx Communication Interface

香 港 中 文 大 學（深 圳）
The Chinese University of Hong Kong, Shenzhen

life.augmented

## STM32F103xC, STM32F103xD, STM32F103xE

High-density performance line ARM®-based 32-bit MCU with 256 to 512KB Flash, USB, CAN, 11 timers, 3 ADCs, 13 communication interfaces

Datasheet − production data

### Features

- Core: ARM® 32-bit Cortex®-M3 CPU
  - 72 MHz maximum frequency, 1.25 DMIPS/MHz (Dhrystone 2.1) performance at 0 wait state memory access
  - Single-cycle multiplication and hardware division
- Memories
  - 256 to 512 Kbytes of Flash memory
  - up to 64 Kbytes of SRAM
  - Flexible static memory controller with 4 Chip Select. Supports Compact Flash, SRAM, PSRAM, NOR and NAND memories
  - LCD parallel interface, 8080/6800 modes
- Clock, reset and supply management
  - 2.0 to 3.6 V application supply and I/Os
  - POR, PDR, and programmable voltage detector (PVD)
  - 4-to-16 MHz crystal oscillator
  - Internal 8 MHz factory-trimmed RC
  - Internal 40 kHz RC with calibration
  - 32 kHz oscillator for RTC with calibration
- Low power
  - Sleep, Stop and Standby modes
  - $V_{BAT}$ supply for RTC and backup registers
- 3 × 12-bit, 1 μs A/D converters (up to 21 channels)
  - Conversion range: 0 to 3.6 V
  - Triple-sample and hold capability
  - Temperature sensor
- 2 × 12-bit D/A converters
- DMA: 12-channel DMA controller
  - Supported peripherals: timers, ADCs, DAC, SDIO, $I^2$Ss, SPIs, $I^2$Cs and USARTs
- Debug mode
  - Serial wire debug (SWD) & JTAG interfaces
  - Cortex®-M3 Embedded Trace Macrocell™
- Up to 112 fast I/O ports
  - 51/80/112 I/Os, all mappable on 16 external interrupt vectors and almost all 5 V-tolerant


- Up to 11 timers
  - Up to four 16-bit timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
  - 2 × 16-bit motor control PWM timers with dead-time generation and emergency stop
  - 2 × watchdog timers (Independent and Window)
  - SysTick timer: a 24-bit downcounter
  - 2 × 16-bit basic timers to drive the DAC
- Up to 13 communication interfaces
  - Up to 2 × $I^2$C interfaces (SMBus/PMBus)
  - Up to 5 USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
  - Up to 3 SPIs (18 Mbit/s), 2 with $I^2$S interface multiplexed
  - CAN interface (2.0B Active)
  - USB 2.0 full speed interface
  - SDIO interface
- CRC calculation unit, 96-bit unique ID
- ECOPACK® packages

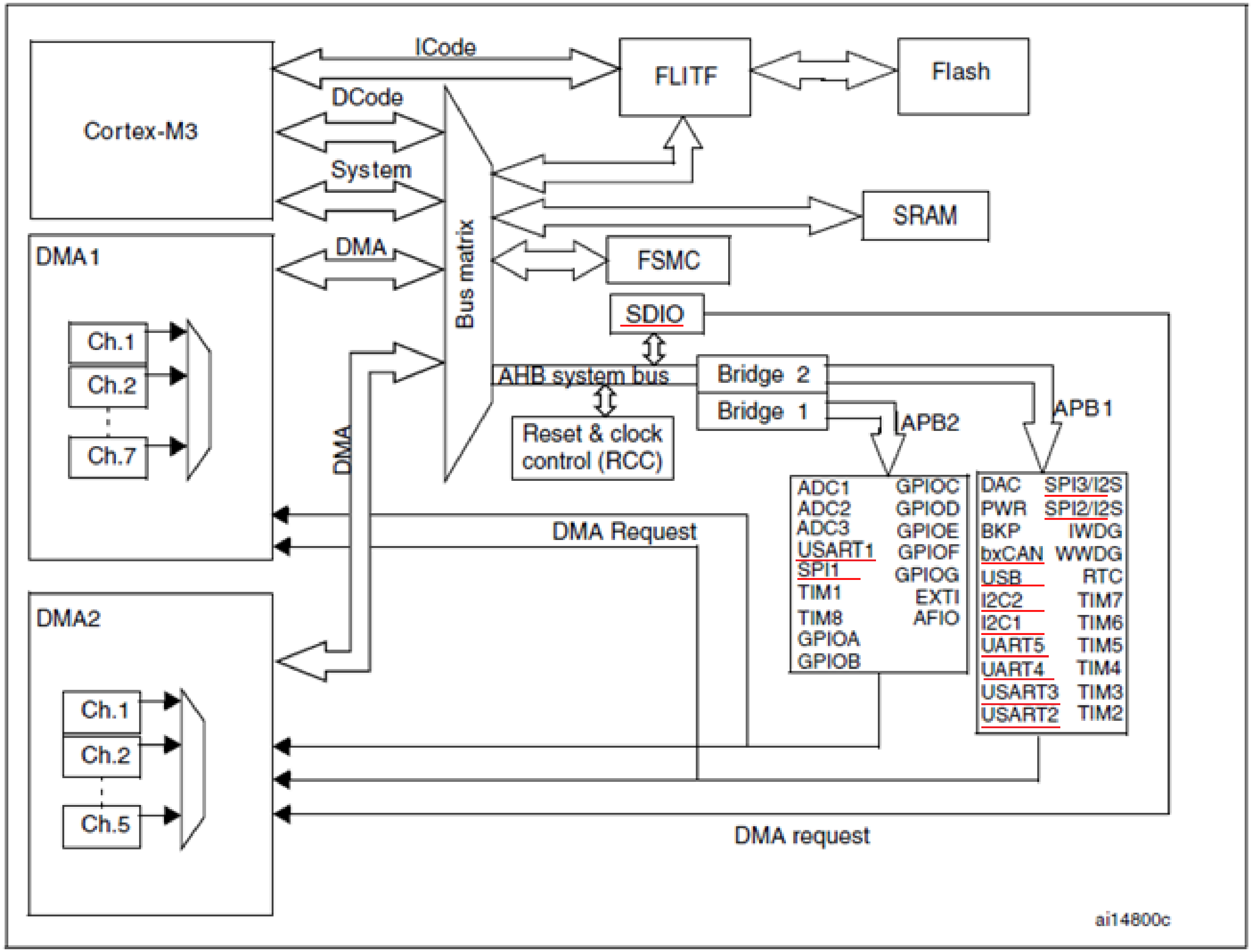
Table 1.Device summary

| Reference | Part number |
|---|---|
| STM32F103xC | STM32F103RC STM32F103VC STM32F103ZC |
| STM32F103xD | STM32F103RD STM32F103VD STM32F103ZD |
| STM32F103xE | STM32F103RE STM32F103ZE STM32F103VE |

- Up to 13 communication interfaces
  - Up to 2 × $I^2$C interfaces (SMBus/PMBus)
  - Up to 5 USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
  - Up to 3 SPIs (18 Mbit/s), 2 with $I^2$S interface multiplexed
  - CAN interface (2.0B Active)
  - USB 2.0 full speed interface
  - SDIO interface

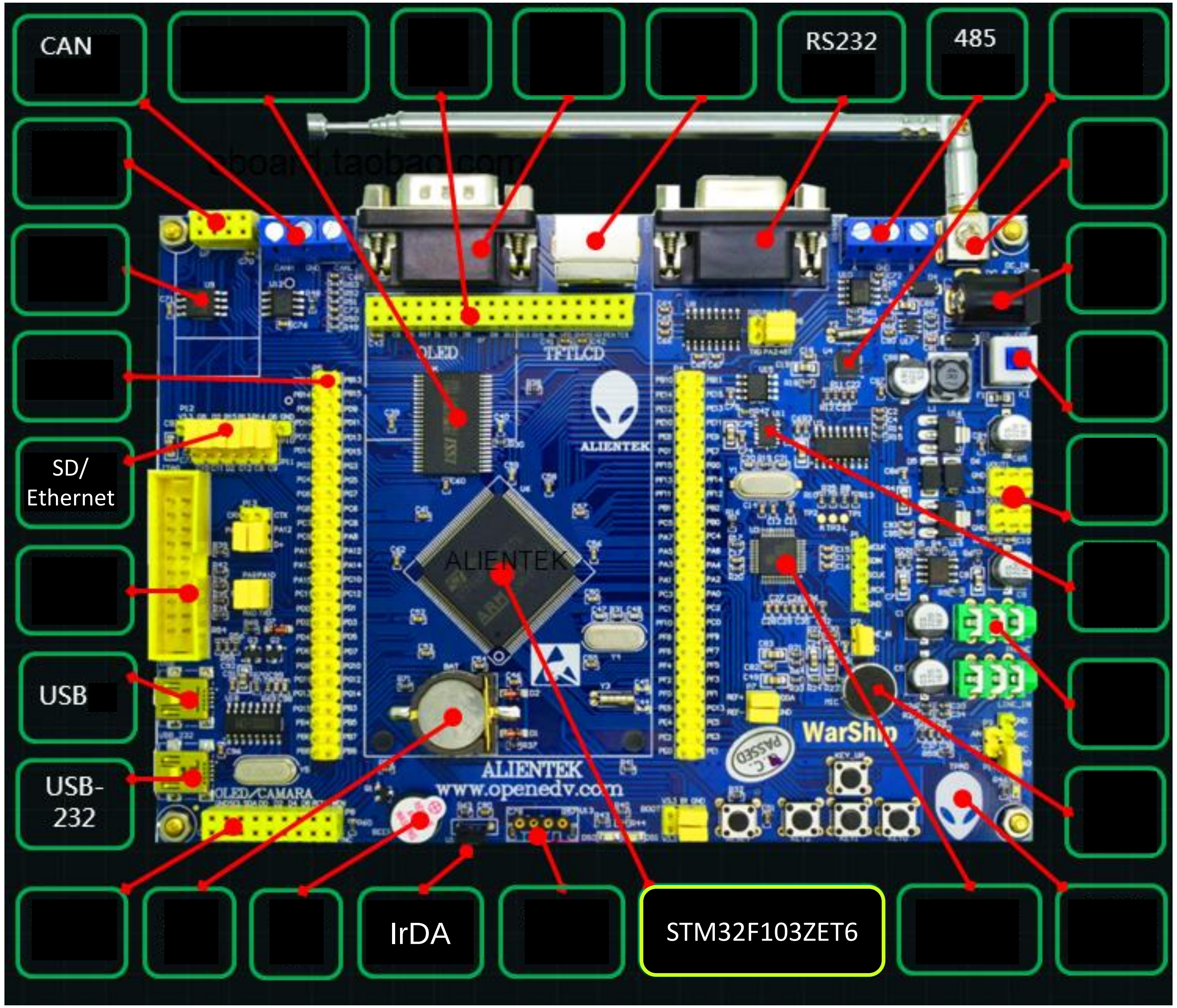
Contents from STM32F103xx data sheet

# STM32F103xx Communication Interface




- Up to 13 communication interfaces
  - Up to 2 × I$^2$C interfaces (SMBus/PMBus)
  - Up to 5 USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
  - Up to 3 SPIs (18 Mbit/s), 2 with I$^2$S interface multiplexed
  - CAN interface (2.0B Active)
  - USB 2.0 full speed interface
  - SDIO interface

# Communication Interface on the Development Board



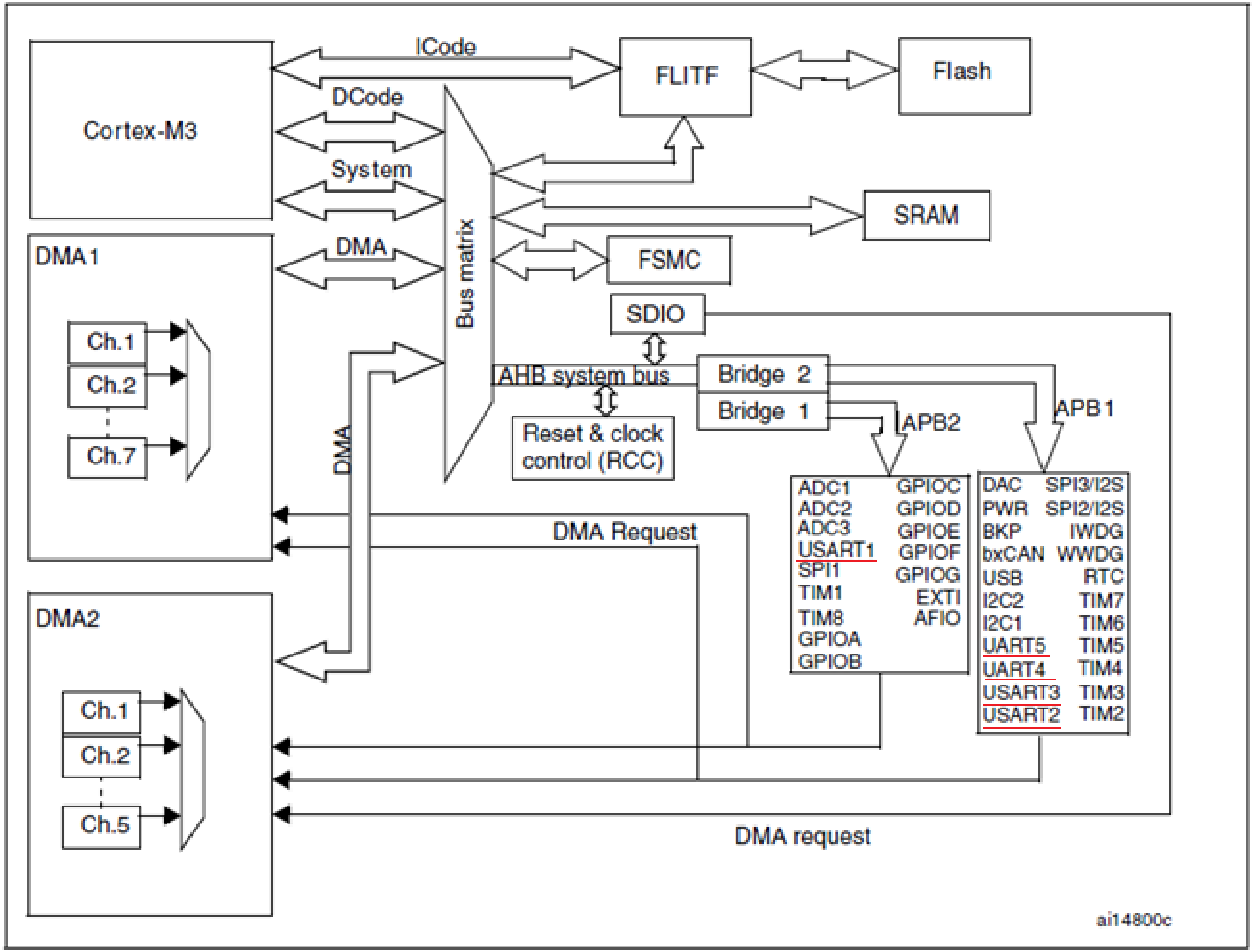

- Up to 13 communication interfaces
  - Up to 2 × $I^2C$ interfaces (SMBus/PMBus)
  - Up to 5 USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
  - Up to 3 SPIs (18 Mbit/s), 2 with $I^2S$ interface multiplexed
  - CAN interface (2.0B Active)
  - USB 2.0 full speed interface
  - SDIO interface

# Universal Synchronous Asynchronous Receiver Transmitter (USART)



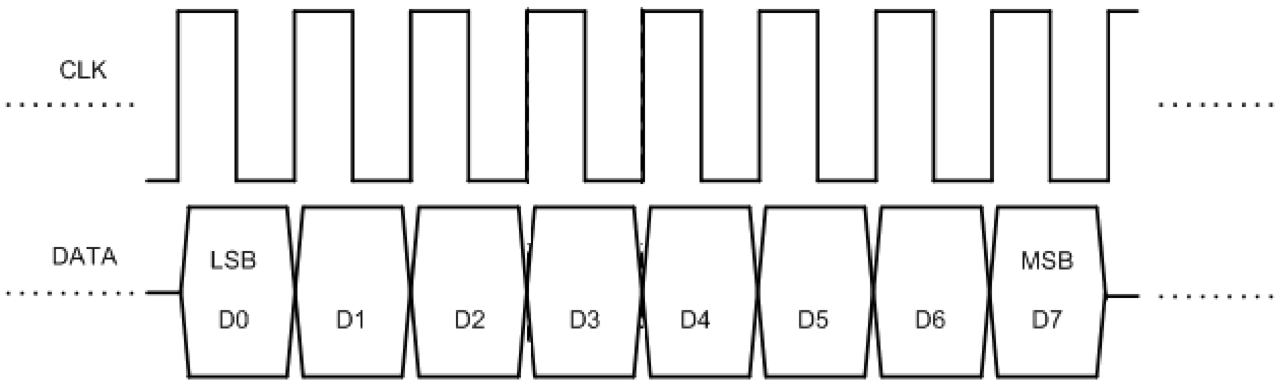
Images from http://en.wikipedia.org/wiki/I%C2%B2C

# USART in STM32



# USART

- A **Universal Synchronous-Asynchronous Receiver-Transmitter (USART)** is a type of serial interface that the microcontroller can be communicate synchronously or asynchronously with other devices.
- Serial transmission of digital information (bits) through a single wire is less expensive than parallel transmission through multiple wires.
- **In synchronous mode**, aside from the transmitted data, one device will generate a clock pulse and transmit it to the receiving device. Based on this clock pulse, the receiver can sample the data received, and get to know the information from the transmitter.



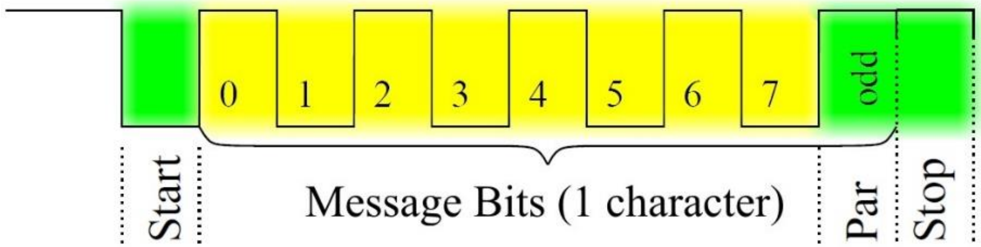

Synchronous serial transmission

# USART



**In asynchronous mode,** in another way, does not send out clock pulses. **The two devices must set up the same baud rate**. The receiver can retrieve the information from one single wire with the following protocol:

- **Start bit**: The signal is high when the line idles. If the receiver detects a low voltage, it will start receiving the next 8 bits as data. After that, there can be an optional parity bit. There are two typical types of parity, i.e., even or odd. The last bit is a stop bit, which is a high voltage.
- **Message bits**: In general, there are 8 bits data. But there can also be other lengths (5-9).
- **Parity bit (optional)**: In even parity, sum of the number of "1" s in data bits and the parity bit should be an even number. In odd parity, that sum should be an odd number.
- **Stop bit**: The signal is high when the transmission is completed.



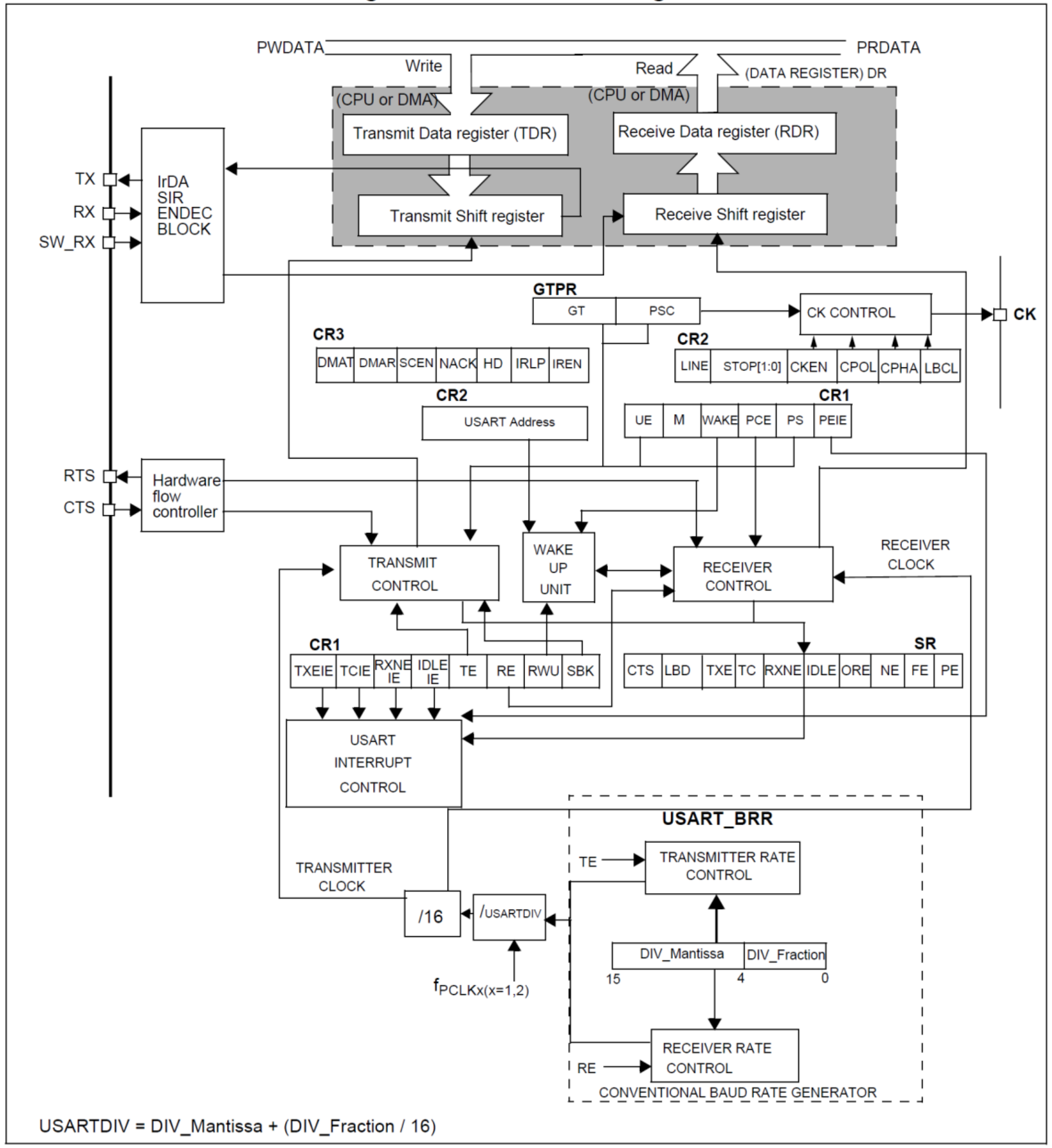

Asynchronous serial transmission

# USART

Any USART bidirectional communication requires a minimum of two pins: Receive Data In (RX) and Transmit Data Out (TX).

- **RX (Receive Data Input)**: It is the serial data input. Oversampling techniques are used for data recovery by discriminating between valid incoming data and noise.
- **TX (Transmit Data Output)**:
  - When the transmitter is disabled, the output pin returns to its I/O port configuration.
  - When the transmitter is enabled and nothing is to be transmitted, the TX pin is at high level.
  - In single-wire and smartcard modes, this I/O is used to transmit and receive the data (at USART level, data are then received on SW_RX).

Figure 279. USART block diagram




Contents from RM0008

# Major Limitations of USART



- **Speed:** USART is relatively slow compared to other communication protocols like SPI (Serial Peripheral Interface) or I2C (Inter-Integrated Circuit). The maximum baud rate of USART is typically limited to around 1-2 Mbps.
- **Distance:** USART signals (0-3.3V) are not suitable for long-distance transmission due to signal degradation and noise. The maximum distance for reliable USART communication is typically limited to around 1-5 meters at baud rate 115200bps.
- **Limited multi-drop capability:** USART is not designed for multi-drop applications, where multiple devices are connected to the same bus.
- **Limited compatibility:** USART is not compatible with all devices or systems, and may require additional hardware or software to interface with certain devices.

# UART Application
# RS-232 (EIA-232)



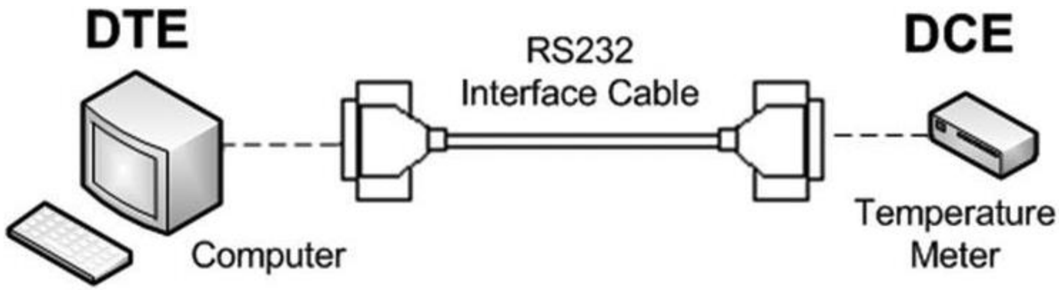
Images from http://en.wikipedia.org/wiki/I%C2%B2C


11

# RS-232

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

- **RS-232** (Recommended Standard 232) is a standard originally introduced in 1960 for serial communication transmission of data.
- It defines the **electrical and mechanical characteristics** of the interface (including voltage levels, signal connecting, and connector pinouts) between a **DTE (data terminal equipment)** such as a computer terminal or PC, and a **DCE (data circuit-terminating equipment or data communication equipment)**, such as a modem.
- The current version of the standard is *TIA-232-F Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange*, issued in 1997.
- The RS-232 standard had been commonly used with serial ports and serial cables. It is still widely used in industrial communication devices.



# RS-232

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

RS-232 is a widely used standard for serial communication and is commonly used in many applications, including:

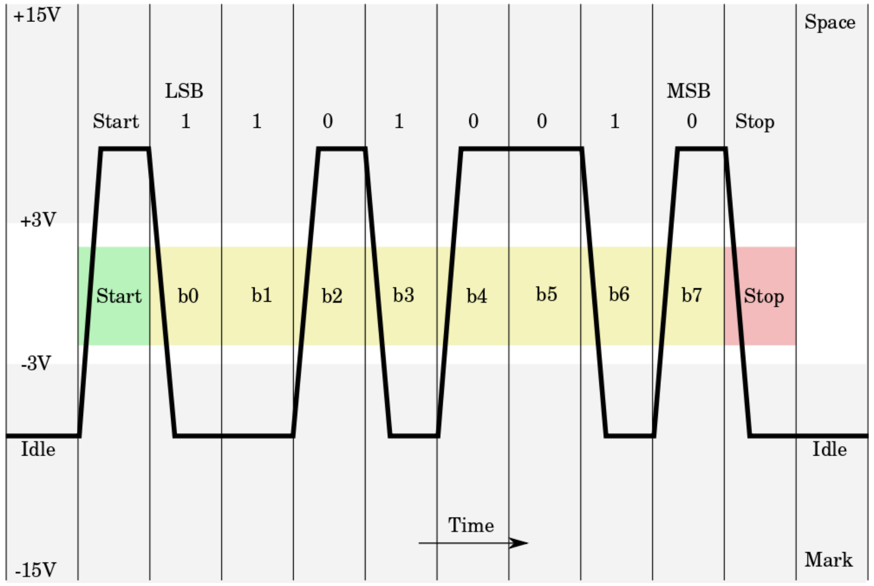
- **Computer-to-computer communication**: RS-232 is used to connect computers to each other, allowing them to exchange data and communicate.
- **Computer-to-peripheral communication**: RS-232 is used to connect computers to peripherals such as printers, modems, and serial mice.
- **Industrial control systems**: RS-232 is used in industrial control systems to connect devices such as programmable logic controllers (PLCs), sensors, and actuators.
- **Medical devices**: RS-232 is used in medical devices such as patient monitors, ventilators, and infusion pumps.

# RS-232 – Data and Control Circuits

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

- Valid signals are either in the range of +3 to +15 volts or the range −3 to −15 volts with respect to the ground/common pin; consequently.
- Data transmission lines (e.g., TxD, RxD and their secondary channel equivalents): Logic One is defined as a negative voltage, the signal condition is called "mark". Logic Zero is positive and the signal condition is termed "space".
- Control signals lines (e.g., request to send (RTS), clear to send (CTS), data terminal ready (DTR), and data set ready (DSR)): have the opposite polarity – the asserted or active state is positive voltage and the deasserted or inactive state is negative voltage.

| Data circuits | Control circuits | Voltage |
|---|---|---|
| 0 (space) | Asserted | +3 to +15 V |
| 1 (mark) | Deasserted | −15 to −3 V |

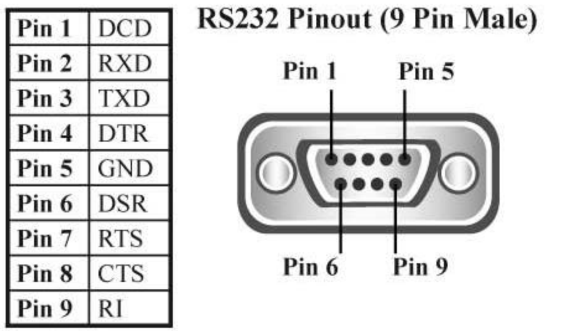


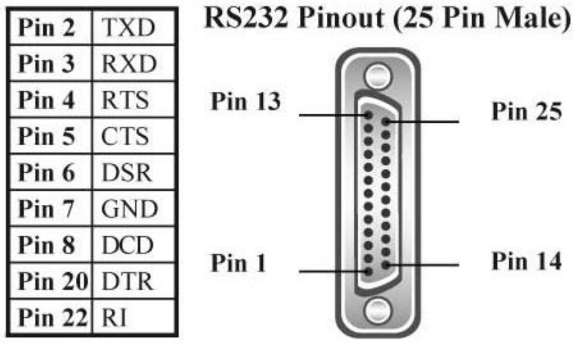

| Signal | Description |
|---|---|
| TxD | Transmitted Data - data transmitted from the DTE to the DCE |
| RxD | Received Data - data transmitted from the DCE to the DTE |
| RTS | Request To Send - set to 0 (*asserted*) by the DTE to prepare the DCE to receive data |
| CTS | Clear To Send - set to 0 (*asserted*) by the DCE to acknowledge RTS and allow the DTE to transmit |
| DTR | Data Terminal Ready - set to 0 (*asserted*) by the DTE to indicate that it is ready to be connected |
| DSR | Data Set Ready - set to 0 (*asserted*) by the DCE to indicate an active connection |

Contents from http://en.wikipedia.org/wiki/RS-232

# RS-232 – Connector and Pinout

| Circuit | | | Direction | | DB-25 pin | DE-9 pin |
|---|---|---|---|---|---|---|
| Name | Typical purpose | Abbreviation | DTE | DCE | | |
| Data Terminal Ready | DTE is ready to receive, initiate, or continue a call. | DTR | Out | In | 20 | 4 |
| Data Carrier Detect | DCE is receiving a carrier from a remote DCE. | DCD | In | Out | 8 | 1 |
| Data Set Ready | DCE is ready to receive and send data. | DSR | In | Out | 6 | 6 |
| Ring Indicator | DCE has detected an incoming ring signal on the telephone line. | RI | In | Out | 22 | 9 |
| Request To Send | DTE requests the DCE prepare to transmit data. | RTS | Out | In | 4 | 7 |
| Ready To Receive | DTE is ready to receive data from DCE. If in use, RTS is assumed to be always asserted. | RTR | Out | In | 4 | 7 |
| Clear To Send | DCE is ready to accept data from the DTE. | CTS | In | Out | 5 | 8 |
| Transmitted Data | Carries data from DTE to DCE. | TxD | Out | In | 2 | 3 |
| Received Data | Carries data from DCE to DTE. | RxD | In | Out | 3 | 2 |
| Common Ground | Zero voltage reference for all of the above. | GND | Common | | 7 | 5 |
| Protective Ground | Connected to chassis ground. | PG | Common | | 1 | — |





DE-9 Connector

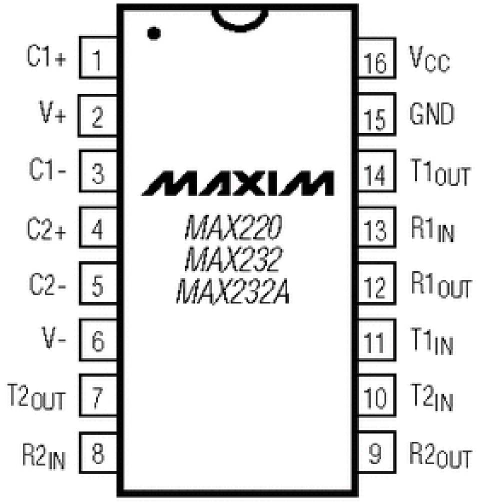

DB25 Connector

# RS-232 – Minimal Wire Connections



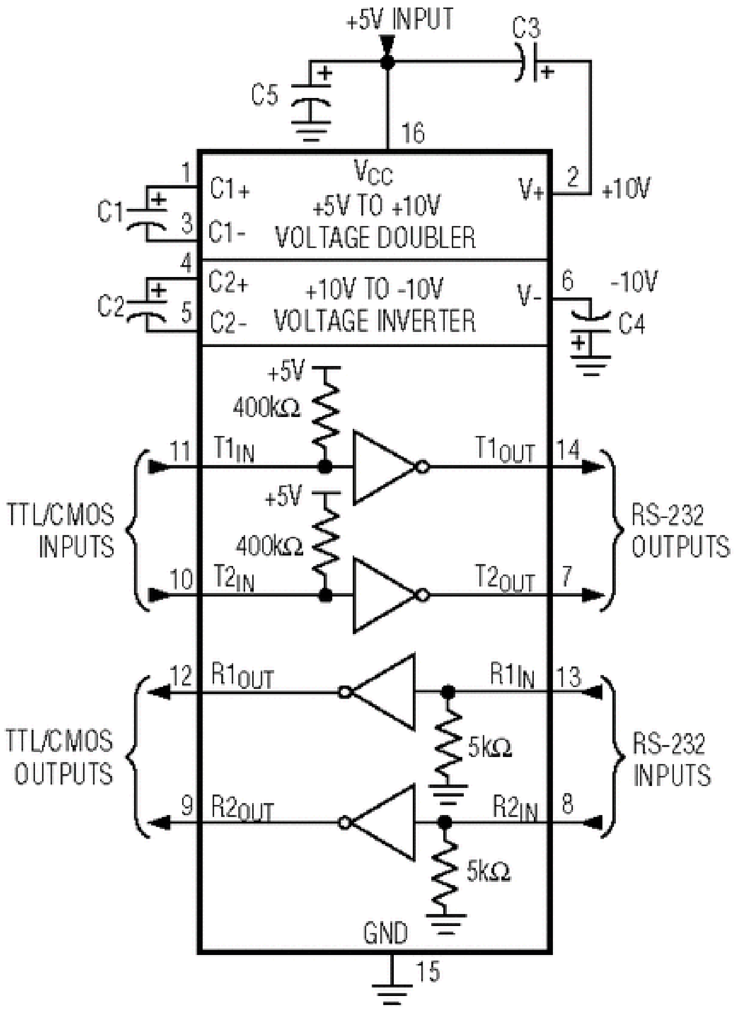
- A minimal "3-wire" RS-232 connection consisting only of transmit data, receive data, and ground, is commonly used when the full facilities of RS-232 are not required.
- Even a "two-wire" connection (data and ground) can be used if the data flow is one way.
- When only hardware flow control is required in addition to two-way data, the RTS and CTS lines are added in a "5-wire" version.

# UART – RS-232 Convertor

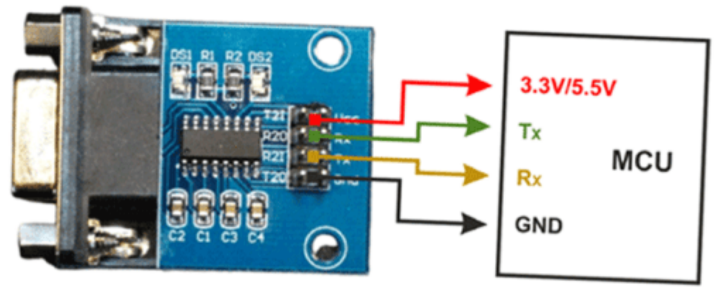



**DIP/SO**

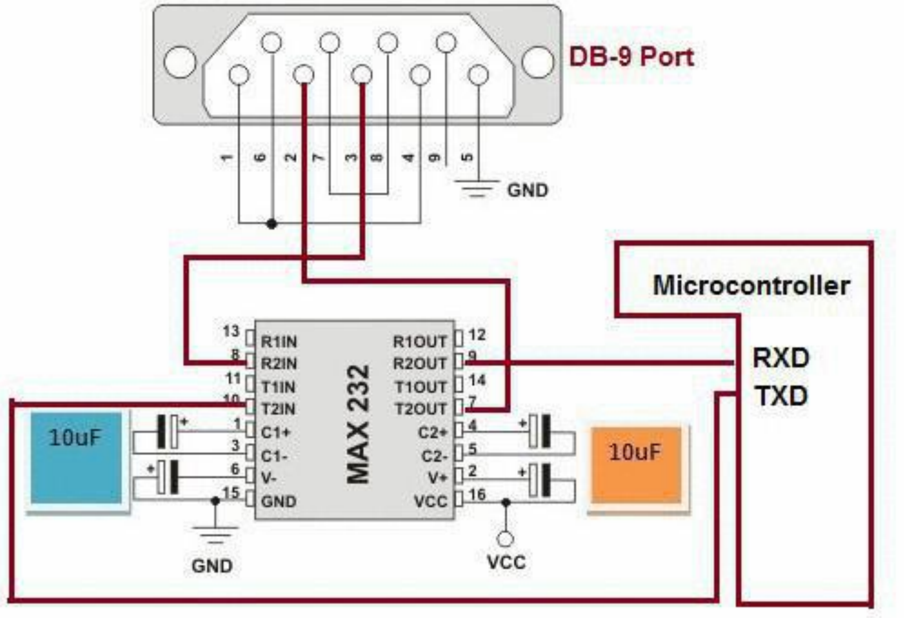
| CAPACITANCE (μF) | | | | | |
|---|---|---|---|---|---|
| **DEVICE** | C1 | C2 | C3 | C4 | C5 |
| **MAX220** | 0.047 | 0.33 | 0.33 | 0.33 | 0.33 |
| **MAX232** | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| **MAX232A** | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |




MAX232 RS232 Pinout

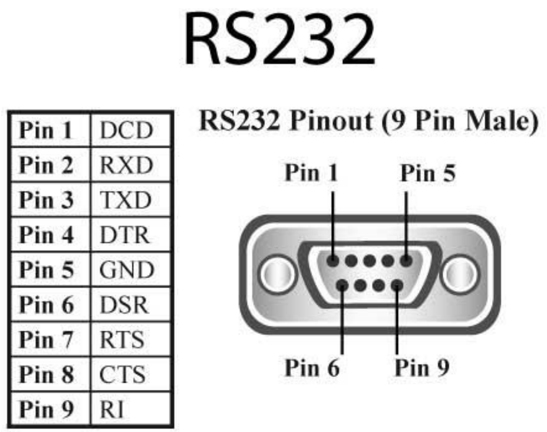



www.DatasheetHub.com



# RS-232 Summary

- Physical Media: Not specified
- Network Topology: Point-to-point
- Maximum Distance: ~15 metres (limited by cable capacitance)
- Mode of Operation: Single-end (reference to GND)
- Maximum Binary Rate: Not specified, hardware dependence
- Voltage Levels: 25V (maximum open-circuit Voltage)
- Mark (1): Negative voltages, -3V to -15V
- Space (0): Positive voltages, +3V to +15V
- Available Signals: TxD, RxD, CTS, DTR, DSR, RTS
- Connector types: D-subminiature connector, D-shell 26-pin "Alt A" connector




www.brainboxes.com

www.misco.co.uk

Contents from http://en.wikipedia.org/wiki/RS-232

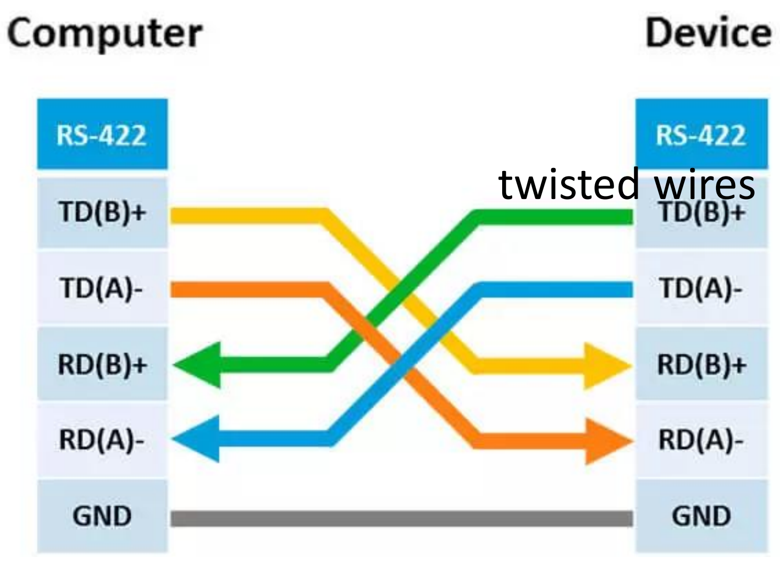
# UART Application
# RS-422 (EIA-422)

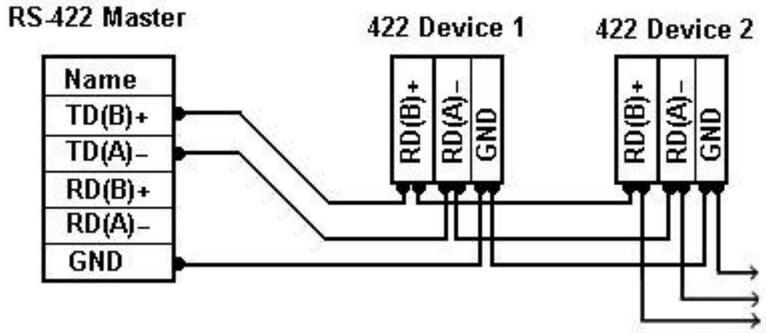
# RS-422

- RS-422, is also known as TIA/EIA-422. It was intended to replace the older RS-232 standard. It supports much **higher speed**, better **immunity from noise**, and **longer cable lengths**.
- The data transfer speed in RS-422 depends on the distance and can vary from **10 kbps (1200 meters)** to **10 Mbps (10 meters)**.
- The RS-422 interface is similar to RS-232. Allows you to simultaneously send and receive messages on separate lines (full duplex), but uses a **differential signal** for this, i.e. The potential difference between conductors A and B.
- The standard only defines signal levels. Other properties of a serial interface, such as electrical connectors and pin wiring, are set by the manufacturer of the device and is specified in the documentation for it.

# RS-422

- **Twisted wires:** The RS-422 line is 4 wires for data transmission (2 twisted wires for transmission and 2 twisted wires for receiving) and one common GND ground wire.
- **Differential signal:** Logical 1 corresponds to A>B by at least 200mV, while Logical 0 means B>A by at least 200mV.
- **Voltage**: The voltage on the data lines can be in the range from -6 V to +6 V.
- **Connections:** In the RS-422 network, there can only be one transmitting device and up to 10 receiving devices.

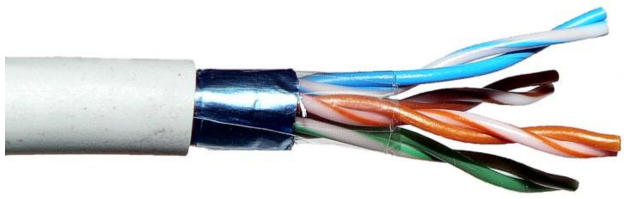


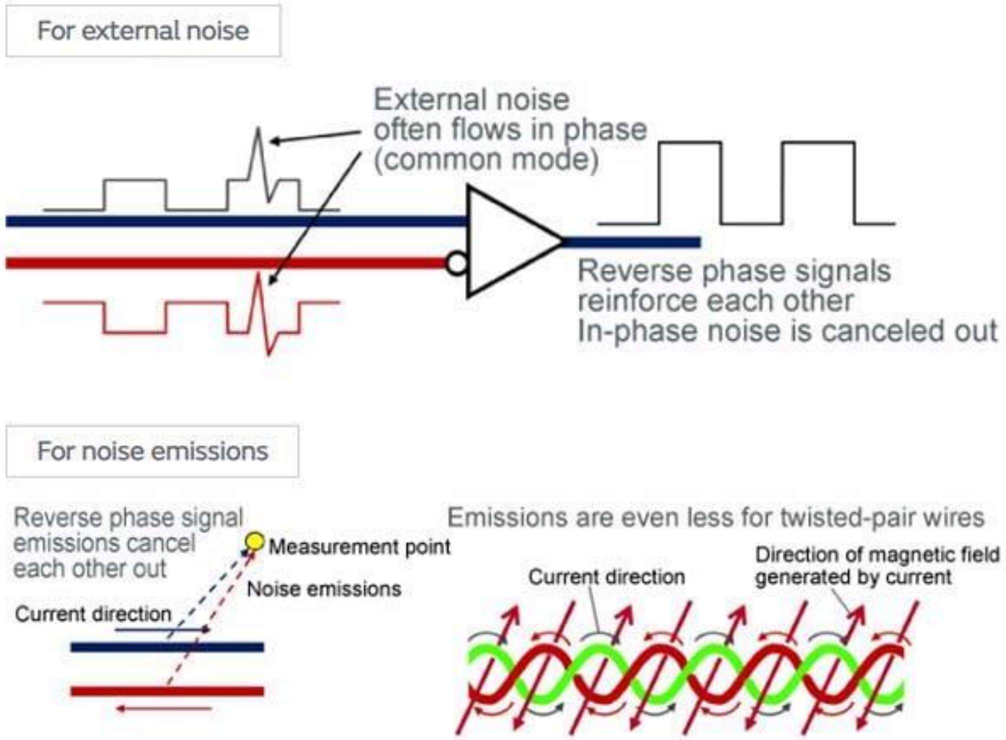



# Twisting wires (Twisted pair)



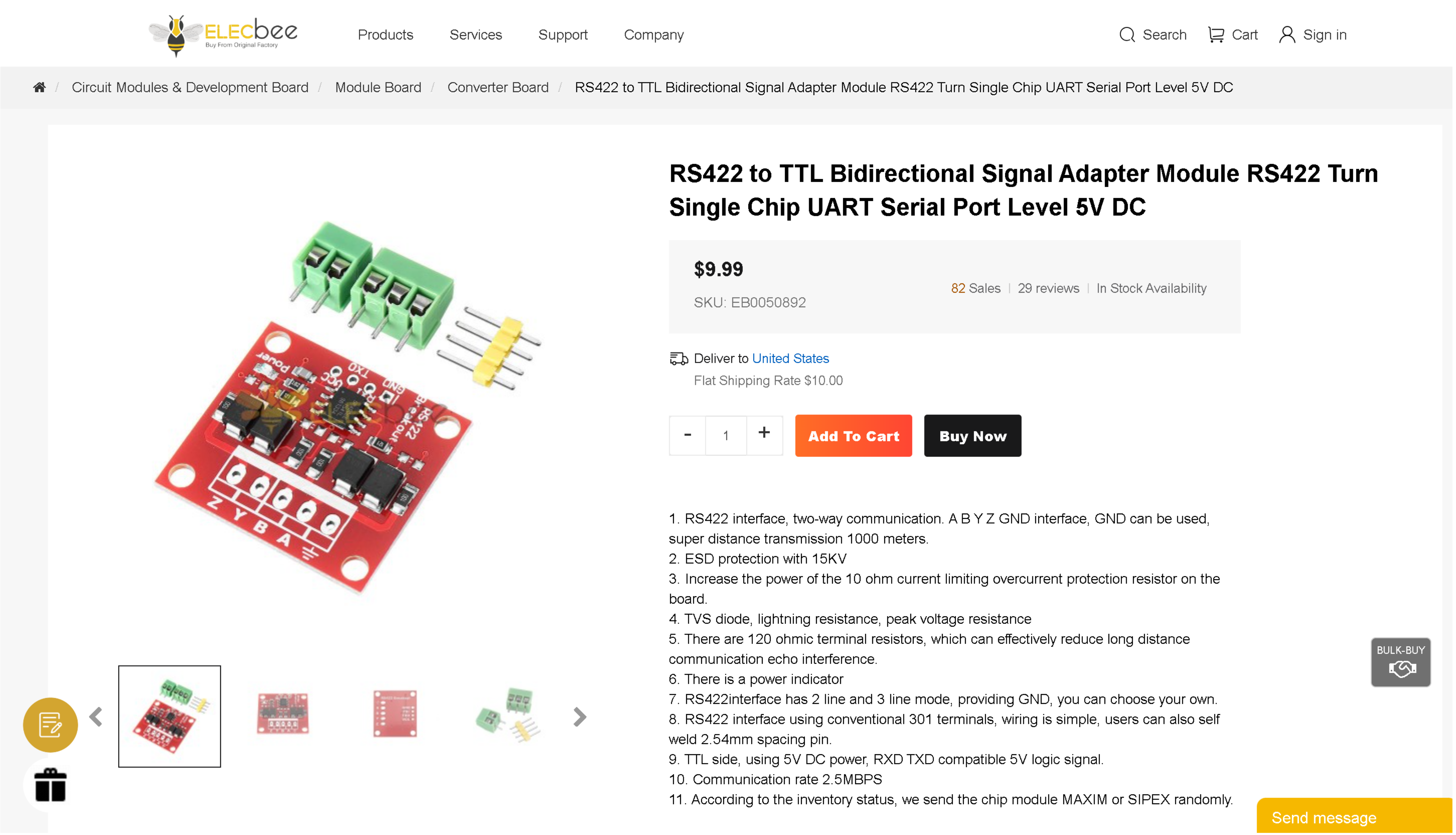
- Twisting wires with each other allows you to get rid of **external interference**, because the interference acts equally on both wires, and the information is extracted from the potential difference between the conductors A and B of one line.
- Twisting wires' reverse phase signal can also cancel out the **noise emissions**.





# UART – RS422 Convertor






https://www.elecbee.com/en-30586-RS422-to-TTL-Bidirectional-Signal-Adapter-Module-RS422-Turn-Single-Chip-UART-Serial-Port-Level-5V-DC

# RS-422 Summary

- Physical Media: Twisted Pair
- Network Topology: Point-to-point, Multi-dropped
- Maximum Devices: 10 (1 driver & 10 receivers)
- Maximum Distance: 1500 meters (4,900 ft)
- Mode of Operation: Differential
- Maximum Binary Rate: 100 kbit/s – 10 Mbit/s
- Voltage Levels : −6V to +6V (maximum differential Voltage)
- Mark (1): Negative voltages
- Space (0): Positive voltages
- Available Signals: Tx+, Tx−, Rx+, Rx− (Full Duplex)
- Connector types: Not specified


Contents from http://en.wikipedia.org/wiki/RS-422

# UART Application
# RS-485 (EIA-485)

# RS-485

- **RS-485** (EIA-485) was designed to address the multi-drop limitation of RS-422, allowing up to 32 devices to communicate.
- In RS-485 each driver can be switched off allowing multiple units to send data. The Output data drivers from the masters are disabled unless data is coming out of a master.
- In one segment of the RS-485 network there can be up to 32 devices, but with the help of additional repeaters and signal amplifiers up to 256 devices. At one time, only one transmitter can be active.
- In industry, the most common interface is RS-485, because the RS-485 uses a multi-point topology, which allows you to connect several receivers and transmitters.

Contents from http://en.wikipedia.org/wiki/RS-485

# RS-485

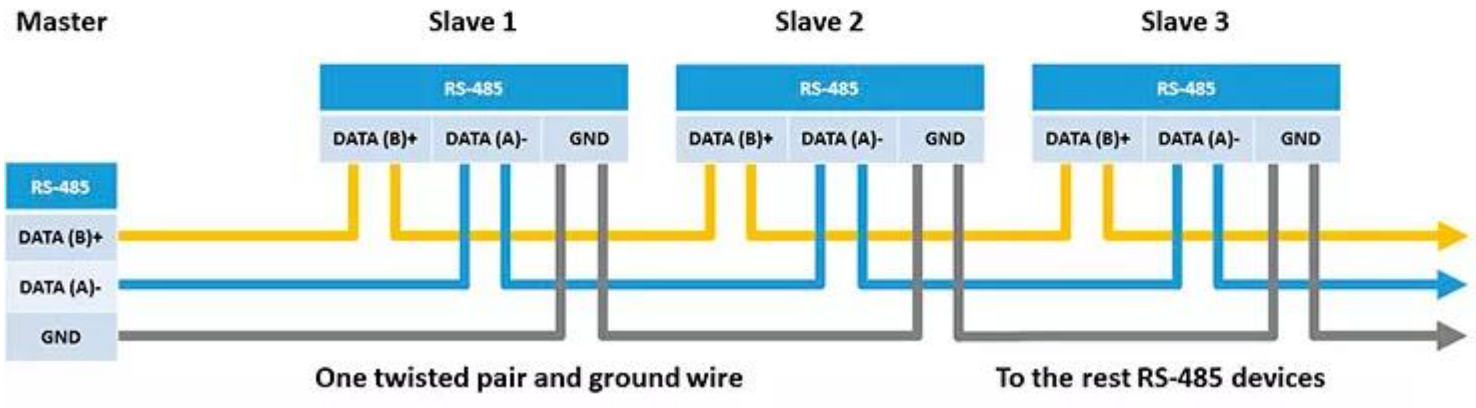


- **RS-485** is a standard defining the electrical characteristics of drivers (a.k.a. transmitter, generator) and receivers for use in balanced digital multipoint systems.
- The RS-485 standard does not define a specific type of connector, which depends on the manufacturer of the device and is specified in the documentation for it.
- The voltage on the lines is in the range from -7 V to +12 V.
- Since it uses a differential (signaling), balanced line over twisted pair (like RS-422), it can span relatively large distances (up to 4,000 ft (1,200 m)).
- A rule of thumb is that the speed (in bit/s) × the length (in meters) should not exceed $10^8$. Thus a 50 meter cable should not signal faster than 2 Mbit/s.

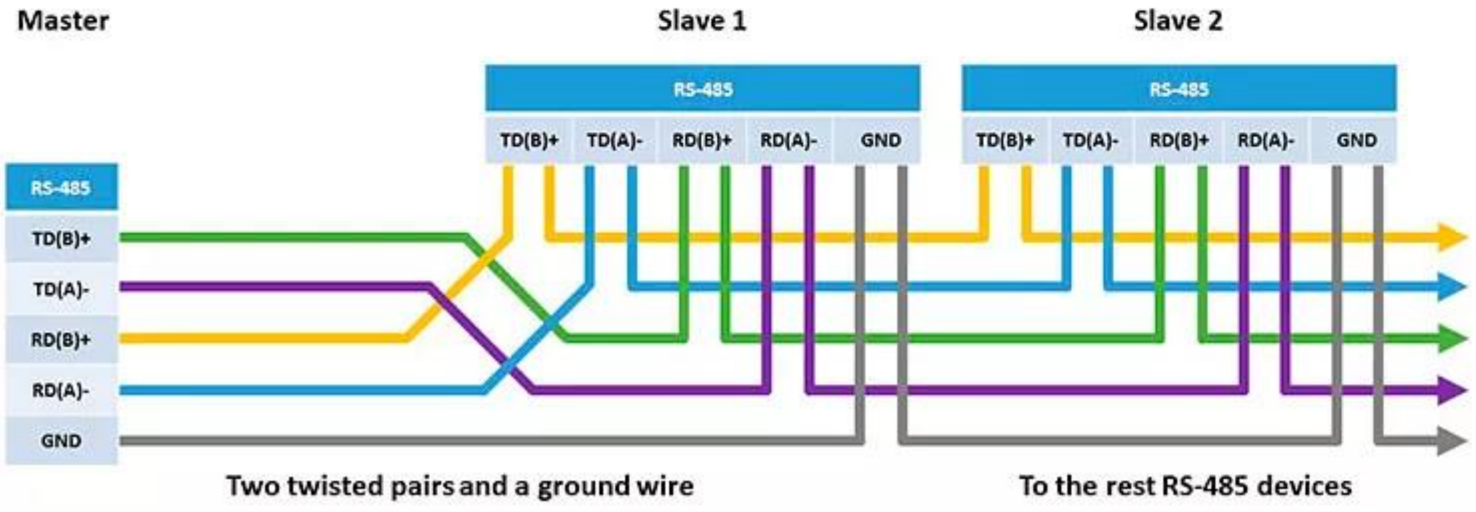
# RS-485



- There are two types of RS-485:
  - RS-485 with 2 contacts, operates in half duplex mode
  - RS-485 with 4 contacts, operates in full duplex mode
- In full duplex mode, you can simultaneously receive and transmit data, and in half-duplex mode either transmit or receive.



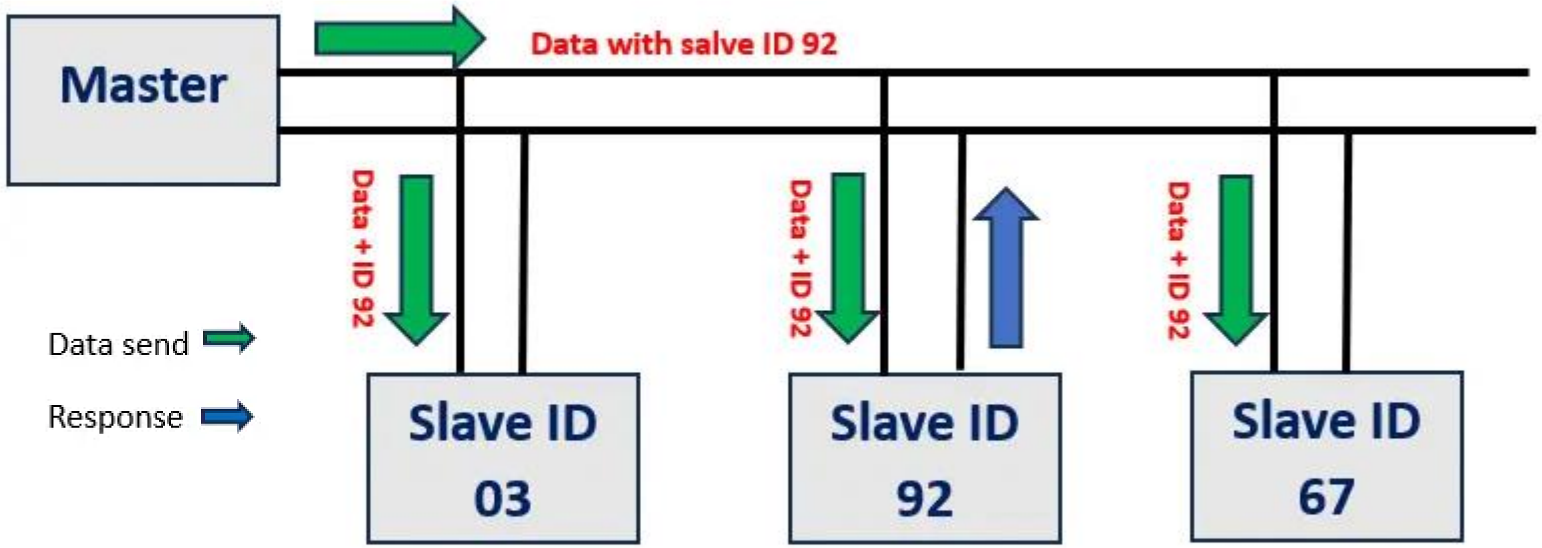

Connect RS-485 devices with 2 contacts.




Connect RS-485 devices with 4 contacts.

# RS-485 Communication

- Using RS485 communication, you can define one master and up to 32 slaves. Each slave has a slave ID. Master sends the data into the common two data lines A (+signal) and B (-signal) with slave addresses. The A and B lines are connected to all the slaves. If the slave ID is matched with the sending slave address, which comes from the master, then this particular slave will receive those data and send a response. The given picture demonstrates the whole process.

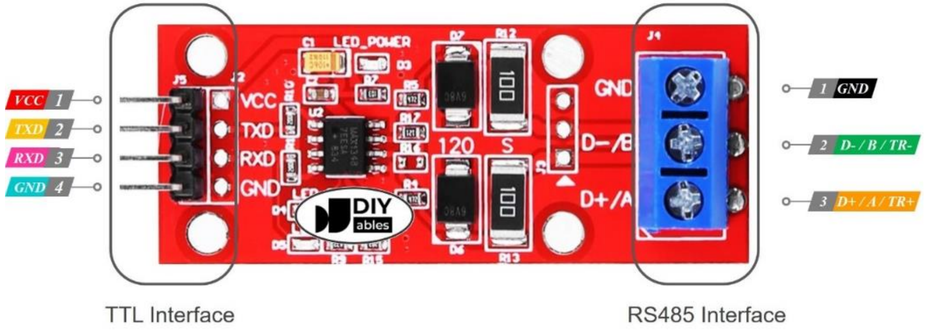


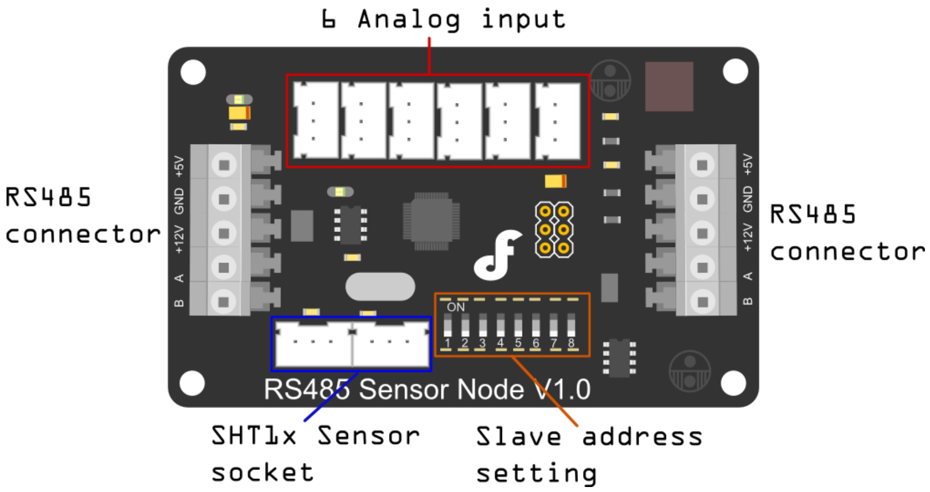

https://embeddedthere.com/rs485-communication-protocol/

# RS485 Hardware



UART – RS485 Convertor



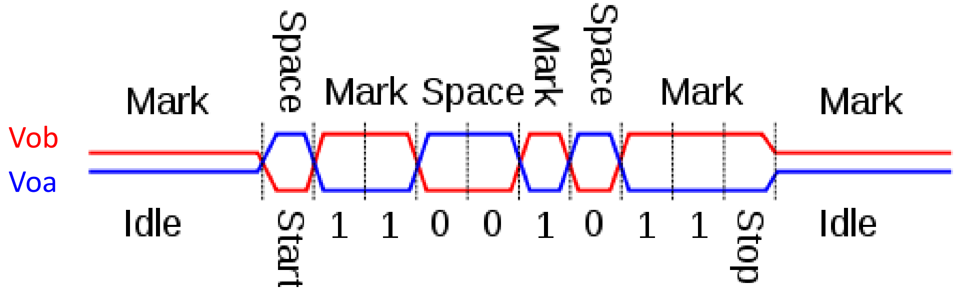




Opto-Isolated RS485 4-Port Hub


# RS-485 Summary



- Physical media: Balanced Interconnecting Cable
- Network topology: Point-to-point, Multi-dropped, Multi-point
- Maximum devices: At least 32 unit loads
- Maximum distance: Not specified
- Mode of operation:
  Differential Receiver levels
  - Binary 1 (OFF) (Voa−Vob < −200 mV)
  - Binary 0 (ON) (Voa−Vob > +200 mV)
- Available signals:
  - A ( '+' or TxD+/RxD+, non-inverting pin)
  - B ( '−' , TxD−/RxD−, inverting pin)
  - C (SC, G, or reference pin)
- Connector types: Not specified

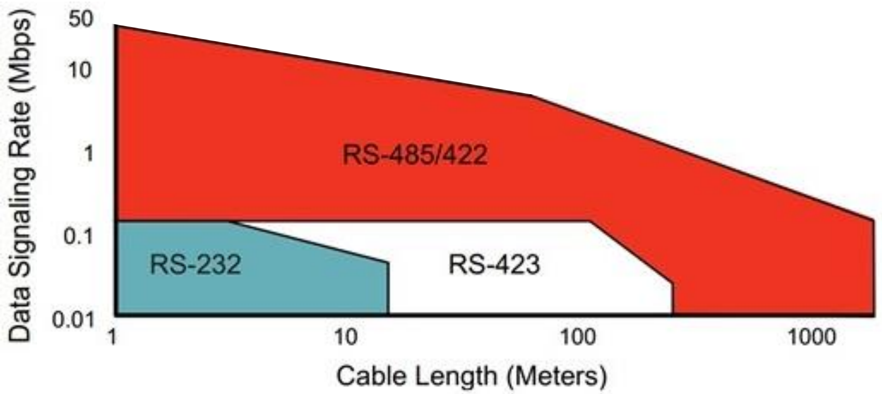



Contents from http://en.wikipedia.org/wiki/RS-485

# Comparison



| SPECIFICATIONS | RS-232 | RS-422 | RS-485 |
|---|---|---|---|
| Line configuration | Single ended | Differential | Differential |
| Type of transfer | Full duplex | Full duplex | Half duplex (2 wire)<br>Full duplex (4 wire) |
| Signals used | $T_X$, $R_X$, RTS, CTS, DTR, DSR, DCD, Ground | $T_X$A, $T_X$B, $R_X$A, $R_X$B, Ground | Data A, Data B, Ground |
| Bus topology | Point-to-point | Point-to-point | Multi-point |
| Maximum connected devices | 1 | 10 (in receive mode) | 32 |
| Maximum length | 15 meters at 9600 bits/s | 1.2 km at 100 Kbits/s | 1.2 km at 100 Kbits/s |
| Maximum data rate | 1 Mbit/s | 10 Mbits/s | 10 Mbits/s |
| Receiver sensitivity | ±3 volts | ±200 millivolts | ±200 millivolts |



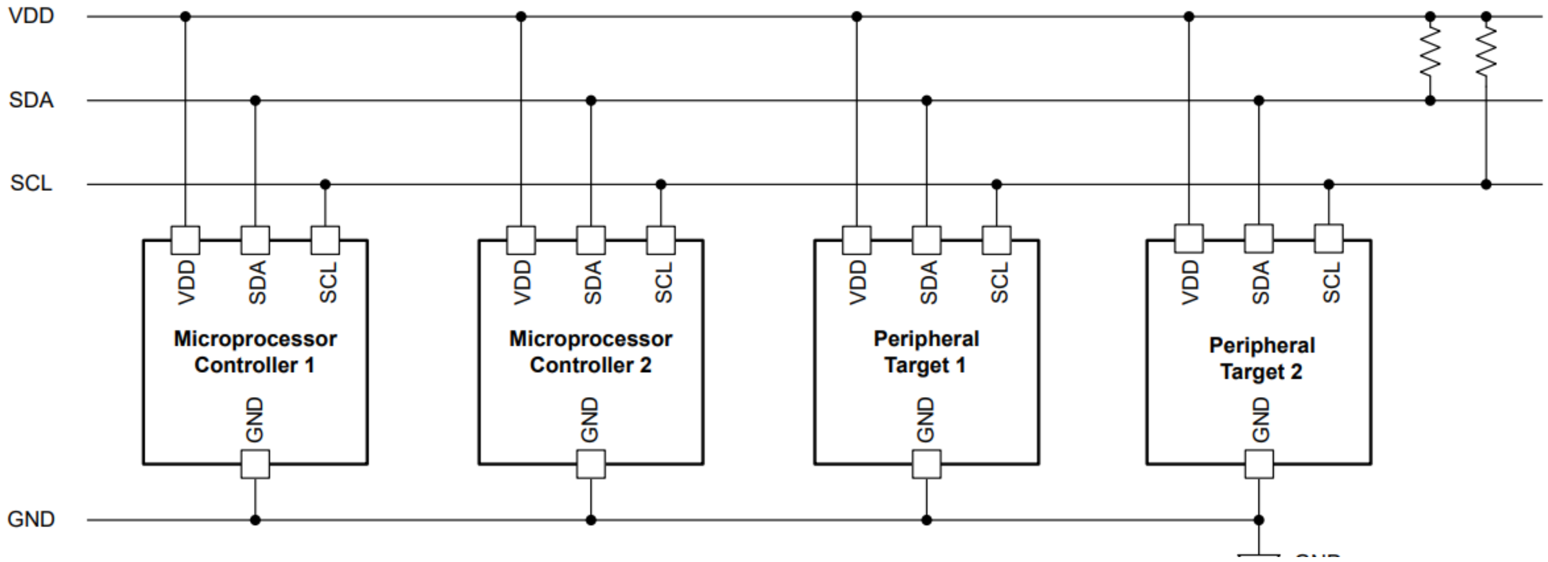
# Inter-Integrated Circuit ($I^2C$)

Images from http://en.wikipedia.org/wiki/I%C2%B2C

# I²C (Inter-Integrated Circuit)



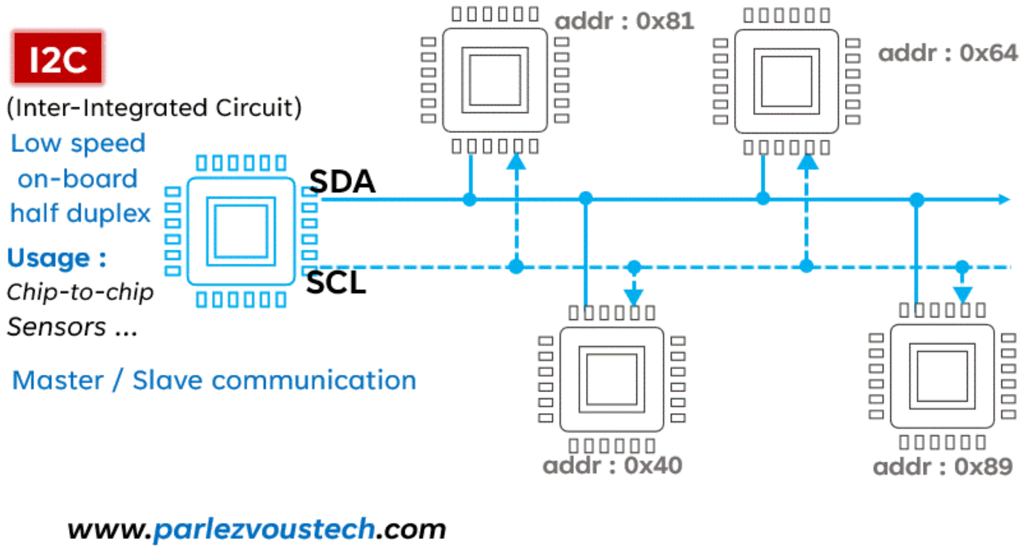
- **I²C** (**Inter-Integrated Circuit**), pronounced *I-squared-C*, is a multi-master, multi-slave, serial computer bus invented by Philips Semiconductor (today known as NXP Semiconductors) designed for attaching low-speed peripherals to computer motherboards and embedded systems.
- It is a **synchronous** and bidirectional serial communication protocol in half-duplex mode.
- Communications are always made at the initiative of a master towards one or more slaves, without direct communication between masters or slaves.
- The protocol uses two lines: SDA (Serial Data Line) for data and SCL (Serial Clock Line) for the clock.



# I²C (Inter-Integrated Circuit)

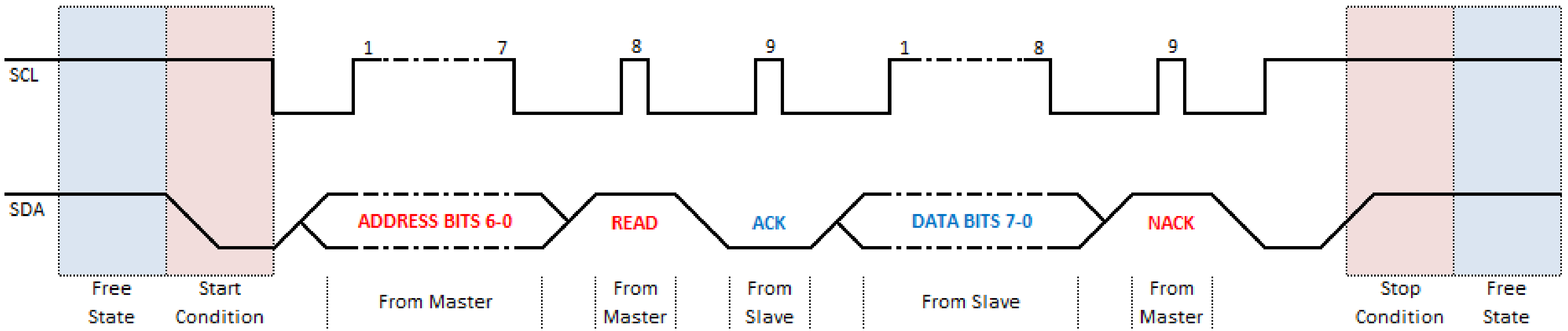
- Typical voltages used are +5 V or +3.3 V, although systems with other voltages are permitted.
- The I²C reference design:
  - A 7-bit or a 10-bit (depending on the device used) address space.
  - There are also other features, such as 16-bit addressing.
  - common I²C bus speeds:

| I²C Mode | Maximum Bit Rate |
|---|---|
| Standard-mode | 100kbps |
| Fast-mode | 400kbps |
| Fast-mode Plus | 1Mbps |
| High-speed mode | 3.4Mbps |
| Ultra-Fast mode | 5Mbps |



# I2C Reading (Slave toward Master) :

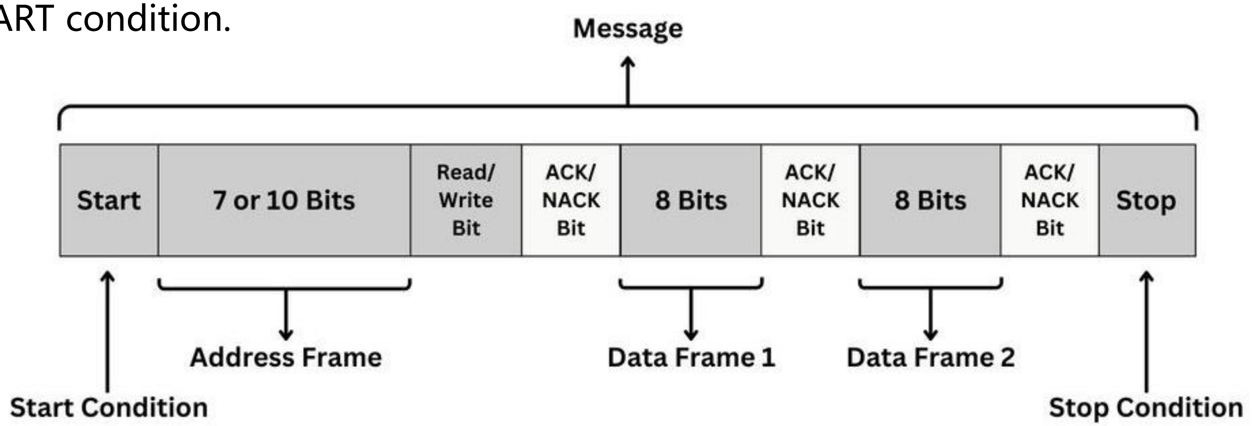
- **START condition** : Reading begins with a START condition initiated by the master. We achieve this by lowering the data line (SDA) low while keeping the clock line (SCL) high. This distinct signal tells all devices on the bus that the master is about to begin a new transmission.
- **Slave Address and Read Bit** : The master then sends the address of the targeted slave on the bus. This address is followed by a control bit, set to 1 to indicate a read operation. Each bit is transmitted sequentially, with a change on the rising edge of each clock pulse.
- **Acknowledgment (ACK/NACK)** : After receiving its address, the slave responds with an acknowledgment (ACK) bit if ready to communicate, pulling the SDA line low for one clock cycle. A non-acknowledgment (NACK) would be signaled by leaving the SDA line high.
- **Receipt of Data** : The slave then begins to send the data to the master, byte by byte.
- **Acquittal by the Master** : After receiving each byte, the master sends an acknowledgment (ACK) bit, pulling the SDA line low, to signal the slave to continue sending data. If the master does not wish to receive more data, it sends a non-acknowledgement (NACK) after the last byte received.
- **STOP condition** : The communication ends with a STOP condition. The master generates this signal by switching the SDA line from low to high while the SCL line is high. This change indicates the end of the reading session and frees the bus for other communications



# Writing (Master toward Slave) :



- **START condition** : Just like in the reading process, writing begins with a START condition initiated by the master. This signal is generated by pulling the data line (SDA) low, while the clock line (SCL) is held high, signaling the start of a transmission.
- **Slave Address and Write Bit** : The master then transmits the address of the recipient slave, followed by a control bit set to 0 to indicate a write operation.
- **Acknowledgment (ACK/NACK)** : The slave, after receiving and recognizing its address, sends an acknowledgment bit (ACK) by pulling the SDA line low for one clock cycle. A non-acknowledgment (NACK) would be indicated by an SDA line held high.
- **Sending Data by the Master** : After receiving the ACK from the slave, the master starts sending the data, byte by byte.
- **Acknowledgment by the Slave** : At the end of each byte transmitted, the slave acknowledges reception by sending an acknowledgment bit (ACK), thus confirming successful reception of the byte. If there is a problem, a NACK can be sent.
- **STOP/RESTART condition** : The operation ends with a STOP condition, generated by the master by passing the SDA line from low to high while SCL is high, thus indicating the end of transmission and release of the bus. If the master wishes to continue with another write or read operation, it can generate a RESTART condition.



# Strengths, weaknesses and use cases



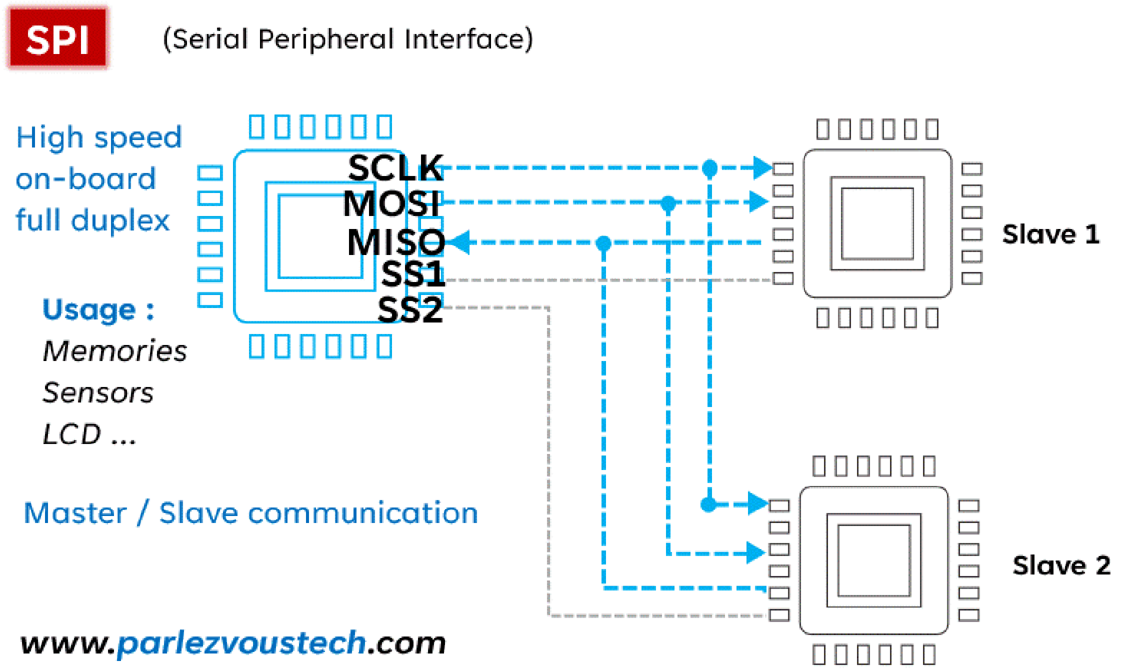
- ***Strengths*** :
  - Low pin consumption, using only two wires.
  - Allows communication with multiple slaves.
  - Relatively simple setup and use.
- ***Weaknesses*** :
  - Limited speed compared to other protocols like SPI.
  - Sensitivity to interference over long distances.
  - Increasing complexity with increasing number of slaves.
- ***Use cases*** : Concerning its applications, I2C shines in contexts requiring simple and economical communications in terms of connectivity. It particularly excels in integration with small sensors, LCD screens, and RTC (Real Time Clock) modules. Additionally, I2C is useful in temperature control devices, battery management systems, and LED controllers for its efficiency in compact circuits. However, in projects requiring rapid or long-distance data transfers, it would be better to opt for other protocols."

# Serial Peripheral Interface (SPI)

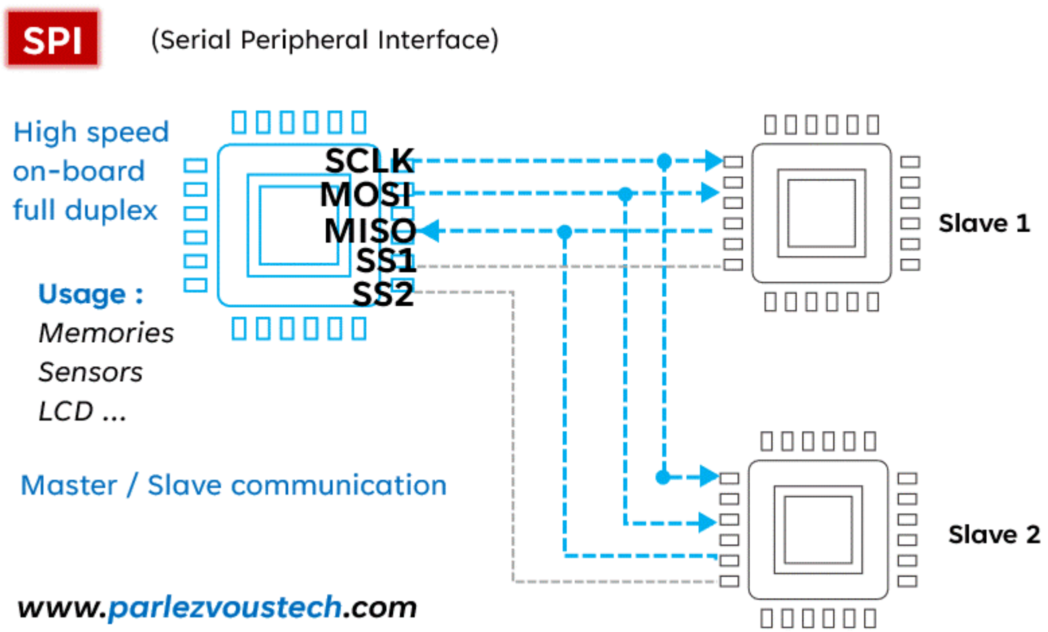
# SPI (Serial Peripheral Interface)

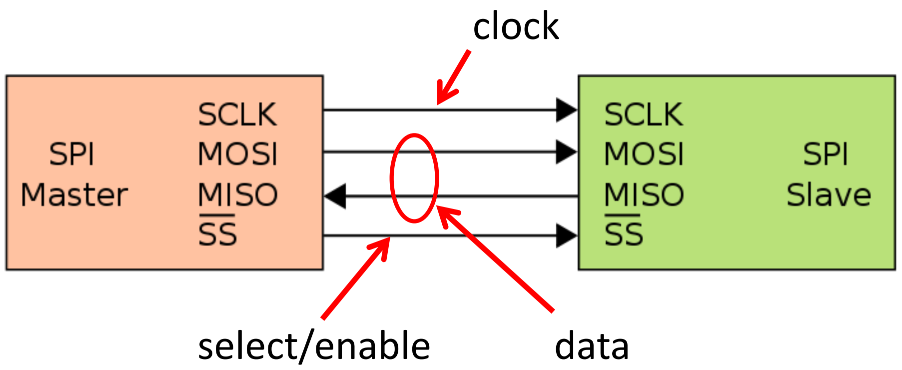


- **SPI (Serial Peripheral Interface)** was developed by Motorola in the mid-1980s, with the earliest SPI microcontrollers modeled after the Motorola 68000 microprocessor, which enjoyed widespread use in early Macintosh computers, arcade games like the Atari ST, and laser printers.
- It stands out for its **high speed**, making it a preferred choice for fast communications.
- The **Serial Peripheral Interface** (**SPI**) is a synchronous serial communication interface specification used for short distance communication, primarily in embedded systems.



- The SPI bus specifies **four logic signals**:
  - **SCLK** : Serial Clock (output from master).
  - **MOSI** : Master Output, Slave Input (output from master).
  - **MISO** : Master Input, Slave Output (output from slave).
  - **SS** : Slave Select (active low, output from master).
- SPI devices communicate in **full duplex mode** using a master-slave architecture with a single master.
- The master device originates the frame for reading and writing. Multiple slave devices are supported through selection with individual slave select (SS) lines.
- Allowing full-duplex communication (send and simultaneous reception).



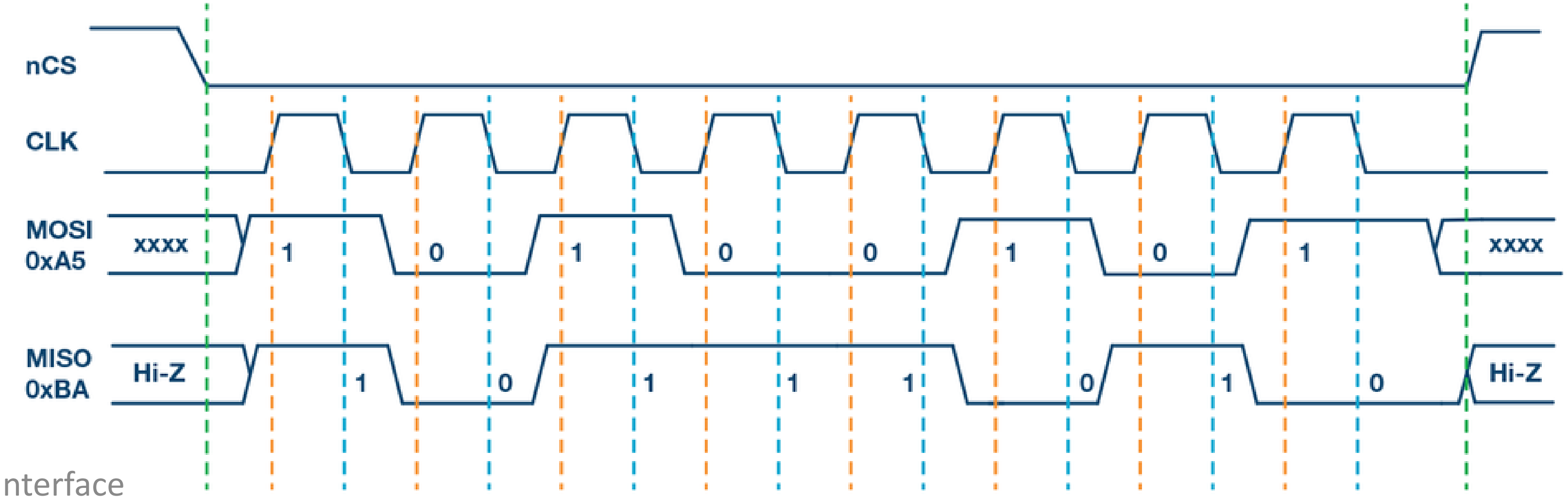



Contents from http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus

# Stages of SPI transmission



- **Clock Signal Generation** : The master starts the communication by generating a clock signal which synchronizes the data exchange.
- **Slave Selection** : The master activates the desired slave by lowering the SS line to a low voltage level.
- **Data Exchange** : The master begins by sending data to the slave through the MOSI line, bit by bit, often starting with the most significant bit. At the same time, the slave can also send data to the master through the MISO line, usually starting with the least significant bit.
- **Simultaneous Communication** : Unlike a simple command-and-response model, SPI allows simultaneous two-way communication. As long as the SS line remains enabled, the master and slave can continue to exchange data at the same time, allowing efficient and fast data transmission in both directions.

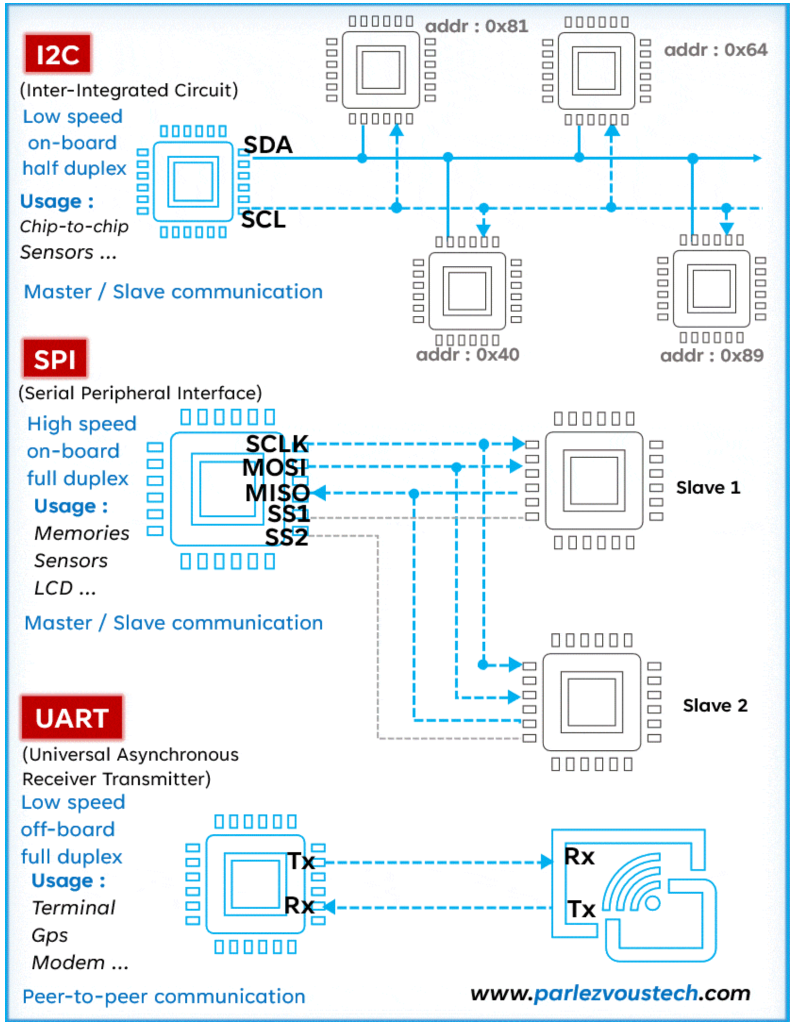


# Advantages, disadvantages and applications

- ***Benefits*** **:**
  - Fast and efficient data transfers.
  - Full-duplex communication for simultaneous sending and receiving.
  - Simplicity of design and implementation.
- ***Drawbacks*** **:**
  - Using multiple pins, which can be a problem in space limited designs.
  - Less efficient for managing multiple slaves compared to I2C.
  - Susceptible to interference at high speeds or over long distances.
- ***Applications*** **:** SPI is ideal for situations requiring fast and reliable data transfers, such as TFT displays, SD memory cards, and wireless communication modules. However, its effectiveness decreases in complex systems with many slaves.

# Comparison

| Characteristic | I2C | SPI | UART |
|---|---|---|---|
| **Speed** | Average (up to a few Mbit/s) | High (several Mbit/s to a few tens of Mbit/s) | Moderate (lower than that of SPI) |
| **Complexity** | Moderate (2 wires, management of several slaves) | Moderate to high (4-wire, full-duplex) | Moderate (2 wires, full-duplex) |
| **Use** | Ideal for short-distance communications with multiple devices | Perfect for fast data transfers | Suitable for simple and long distance serial communications |
| **Duplex** | Half-duplex (two-way communication but not simultaneous) | Full-duplex (simultaneous two-way communication) | Full-duplex (simultaneous two-way communication) |



# Tips for choosing the right protocol for your project

- **Communication speed** : SPI for speed, UART for flexibility, and I2C for less speed-demanding configurations.
- **Circuit design** : I2C for efficient space management with multiple devices, SPI for performance in larger designs, and UART for simplicity and versatility.
- **Distance and communication environment** : UART is robust over long distances, while I2C is better suited to short distances.
- **Duplex Requirements** : SPI and UART provide full-duplex capabilities, while I2C is limited to half-duplex.

# Controller Area Network (CAN Bus)




Contents from http://en.wikipedia.org/wiki/CAN_bus

# Controller Area Network introduction



- A **controller area network** (**CAN bus**) is a vehicle bus standard designed to allow microcontrollers and devices to communicate with each other in applications without a host computer.
- It is a message-based protocol, designed originally for automotive applications, but is also used in many other contexts.
- The CAN standard ISO 11898 which was later restructured into two parts;

  ISO 11898-1, which covers the data link layer, and

  ISO 11898-2, which covers the CAN physical layer for high-speed CAN.
- ISO 11898-3 was released later and covers the CAN physical layer for low-speed, fault-tolerant CAN.

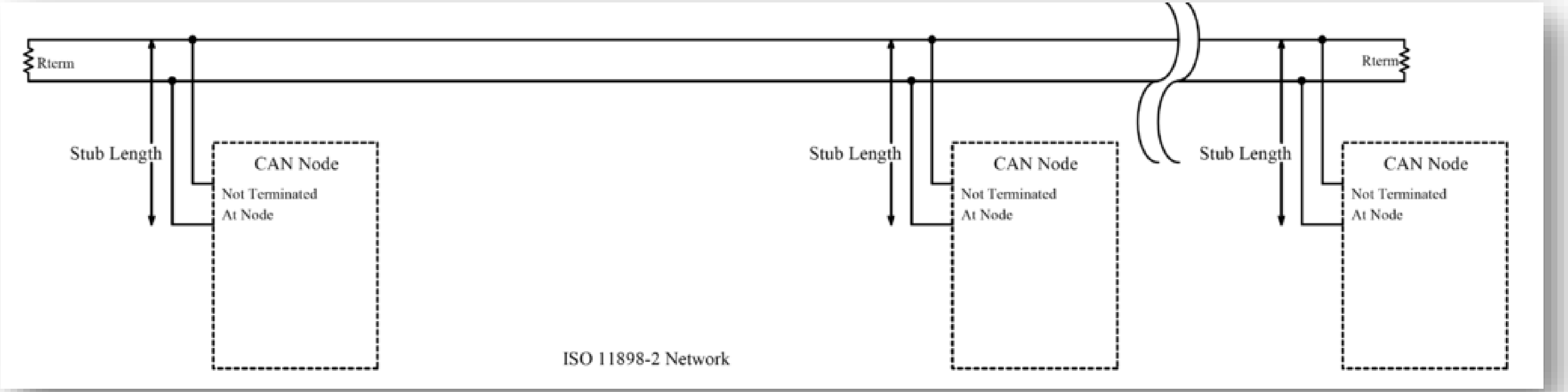


Contents from http://en.wikipedia.org/wiki/CAN_bus

# CAN architecture



- CAN is a multi-master serial bus standard for connecting Electronic Control Units (ECUs) also known as nodes. Two or more nodes are required on the CAN network to communicate.
- The complexity of the node can range from a simple I/O device up to an embedded computer with a CAN interface and sophisticated software. The node may also be a gateway allowing a standard computer to communicate over a USB or Ethernet port to the devices on a CAN network.
- All nodes are connected to each other through a two wire bus. The wires are 120-Ω nominal twisted pair.



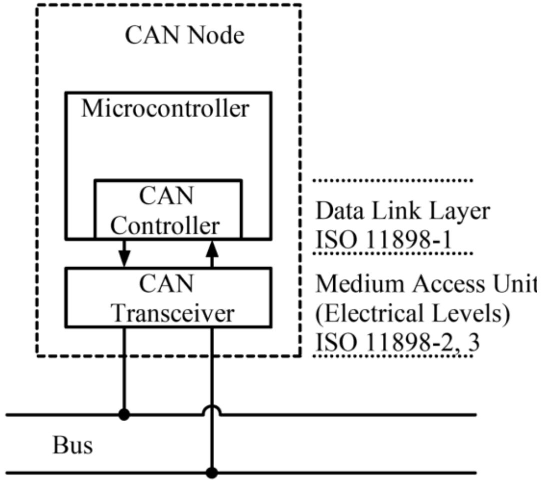



Contents from http://en.wikipedia.org/wiki/CAN_bus

# A CAN node



Each node requires a:




- Central processing unit, microprocessor, or host processor:
  - The host processor decides what the received messages mean and what messages it wants to transmit.
  - Sensors, actuators and control devices can be connected to the host processor.
- CAN controller; often an integral part of the microcontroller:
  - **Receiving**: the CAN controller stores the received serial bits from the bus until an entire message is available, which can then be fetched by the host processor (usually by the CAN controller triggering an interrupt).
  - **Sending**: the host processor sends the transmit message(s) to a CAN controller, which transmits the bits serially onto the bus when the bus is free.
- Transceiver: Defined by ISO 11898-2/3 Medium Access Unit (MAU) standards
  - **Receiving**: it converts the data stream from CANbus levels to levels that the CAN controller uses. It usually has protective circuitry to protect the CAN controller.
  - **Transmitting**: it converts the data stream from the CAN controller to CANbus levels.



Contents from http://en.wikipedia.org/wiki/CAN_bus

# Universal Serial Bus (USB) 2.0

# USB



- **Universal Serial Bus** (**USB**) is an industry standard developed in the mid-1990s that defines the cables, connectors and communications protocols used in a bus for connection, communication, and power supply between computers and electronic devices.
- USB was designed to standardize the connection of computer peripherals (including keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters) to personal computers, both to communicate and to supply electric power.
- USB has effectively replaced a variety of earlier interfaces, such as serial and parallel ports, as well as separate power chargers for portable devices.



Contents from http://en.wikipedia.org/wiki/USB

# USB

- In general, there are three basic kinds or sizes related to the USB connectors and types of established connection:
  - the older "standard" size, in its USB 1.1/2.0 and USB 3.0 variants (for example, on USB flash drives);
  - the "mini" size (primarily for the B connector end, such as on many cameras);  and
  - the "micro" size, in its USB 1.1/2.0 and USB 3.0 variants (for example, on most modern cellphones).
- Unlike other data cables (Ethernet, HDMI etc.), each end of a USB cable uses a different *kind* of connector; an A-type or a B-type. This kind of design was chosen to prevent electrical overloads and damaged equipment, as only the A-type socket provides power. There are cables with A-type connectors on both ends,
  but they should be used carefully.

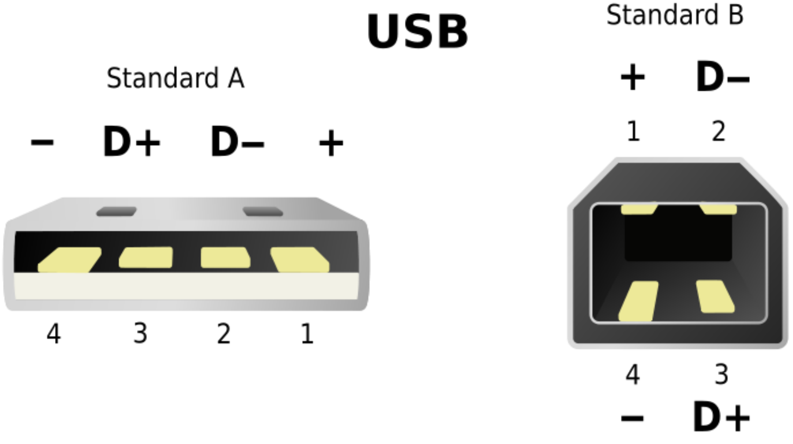


Contents from http://en.wikipedia.org/wiki/USB

# USB

- The A-type and B-type plugs, and the corresponding receptacles are on the computer or electronic device. In common practice, the A-type connector is usually the full size, and the B-type side can vary as needed.
- The mini and micro sizes also allow for a reversible AB-type receptacle, which can accept either an A-type or a B-type plug. This scheme, known as "**USB On-The-Go**", allows one receptacle to perform its double duty in space-constrained applications.

| Mode | Data rate | Introduced in |
|---|---|---|
| Low Speed | 1.5 Mbit/s | USB 1.0 |
| Full Speed | 12 Mbit/s | USB 1.0 |
| High Speed | 480 Mbit/s | USB 2.0 |
| SuperSpeed | 5 Gbit/s | USB 3.0 |
| SuperSpeed+ | 10 Gbit/s | USB 3.1 |





Contents from http://en.wikipedia.org/wiki/USB

# USB On-The-Go



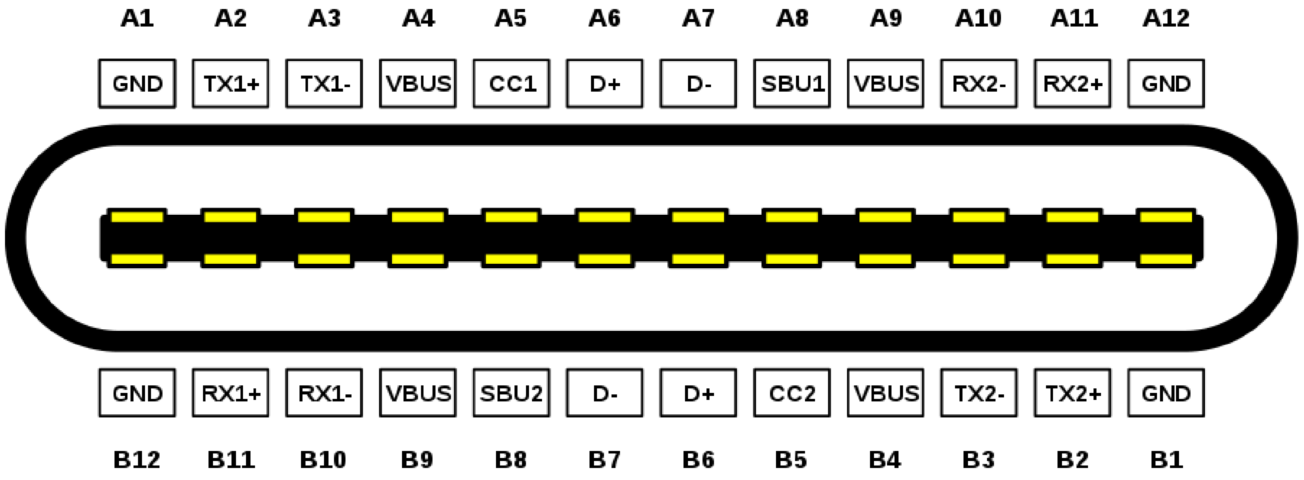
- **USB On-The-Go**, often abbreviated to **USB OTG** or just **OTG**, is a specification first used in late 2001, that allows USB devices such as digital audio players or mobile phones to act as a host, allowing other USB devices like a USB flash drive, digital camera, mouse, or keyboard to be attached to them.
- Use of USB OTG allows these devices to switch back and forth between the roles of host and client devices. For instance, a mobile phone may read from removable media as the host device, but present itself as a USB Mass Storage Device when connected to a host computer.
- In other words, USB On-The-Go introduces the concept that a device can perform both the master and slave roles – whenever two USB devices are connected and one of them is a USB On-The-Go device, they establish a communications link. Whichever device controls that link is called the master or host, while the other is called the slave or peripheral.
- **STM32F103 supports USB On-The-Go Full speed.**

Contents from http://en.wikipedia.org/wiki/USB_On-The-Go

# USB Type-C



- A 24-pin fully reversible-plug USB connector for data and energy transport finalized in August 2014.
- Does not necessarily support USB 3.1 or USB Power Delivery.



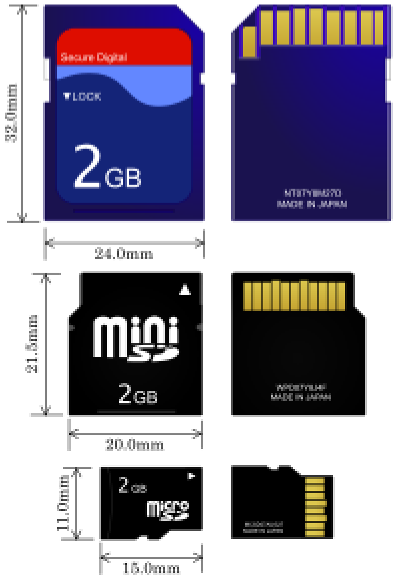




https://en.wikipedia.org/wiki/USB-C and https://www.datapro.net/

# Secure Digital Input Output (SDIO) Interface

- **Secure Digital** (**SD**) is a nonvolatile memory card used extensively in portable devices, such as mobile phones, digital cameras, GPS navigation devices, handheld consoles, and tablet computers.
- The Secure Digital format includes four card families available in three different form factors.
- The four families are the original Standard-Capacity (SDSC), the High-Capacity (SDHC), the eXtended-Capacity (SDXC), and the **SDIO, which combines input/output functions with data storage.**
- The three form factors are the original (standard) size, the mini size, and the micro size. Electrically passive adapters allow a smaller card to fit and function in a device built for a larger card.





SDIO

# SDIO in STM32F103



- The SD/SDIO MMC card host interface (SDIO) provides an interface between the AHB peripheral bus and MultiMediaCards (MMCs), SD memory cards, SDIO cards and CE-ATA devices.
- The SDIO features include the following:
  - Full compliance with *MultiMediaCard System Specification Version 4.2. Card support* for three different databus modes: 1-bit (default), 4-bit and 8-bit
  - Full compatibility with previous versions of MultiMediaCards (forward compatibility)
  - Full compliance with *SD Memory Card Specifications Version 2.0*
  - Full compliance with *SD I/O Card Specification Version 2.0: card support for two* different databus modes: 1-bit (default) and 4-bit
  - Full support of the CE-ATA features (full compliance with *CE-ATA digital protocol Rev1.1)*
  - Data transfer up to 48 MHz for the 8 bit mode
  - Data and command output enable signals to control external bidirectional drivers.

Contents from RM0008

# End