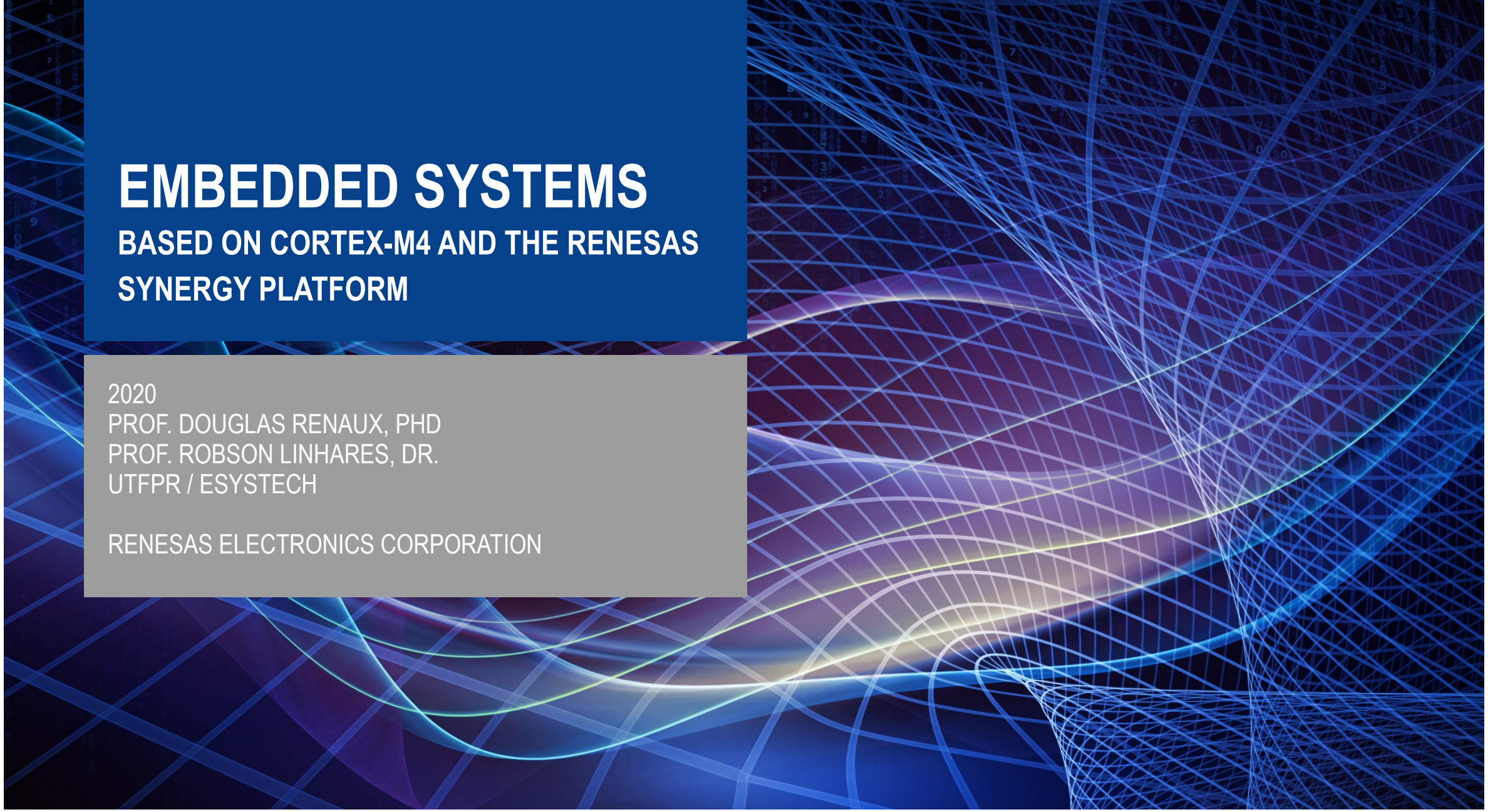


EMBEDDED SYSTEMS

BASED ON CORTEX-M4 AND THE RENESAS SYNERGY PLATFORM

2020
PROF. DOUGLAS RENAUX, PHD
PROF. ROBSON LINHARES, DR.
UTFPR / ESYSTECH

RENESAS ELECTRONICS CORPORATION

8 – SERIAL COMMUNICATIONS

- Serial Communications - Introduction
- UART
  - Concepts, Block Diagram, Registers
- SPI
  - Concepts, Block Diagram, Registers
- I2C
  - Concepts, Block Diagram, Registers

8.1 – INTRODUCTION TO SERIAL COMMUNICATIONS

Concept:

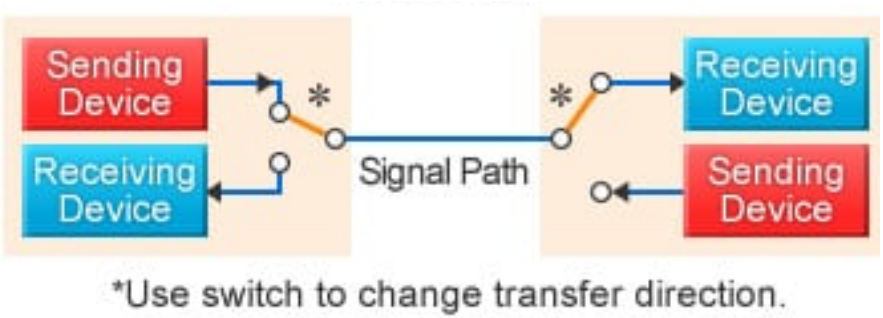
- In serial communications ONE bit is transmitted at a time, from the transmitter device (TX) to the receiver device (RX). As opposed to parallel communications where several bits, e.g. 8 bits or 1 byte, are transmitted concurrently.
- In serial communications a reduced number of wires are required.
- Long distance wired communications typically use serial communication.

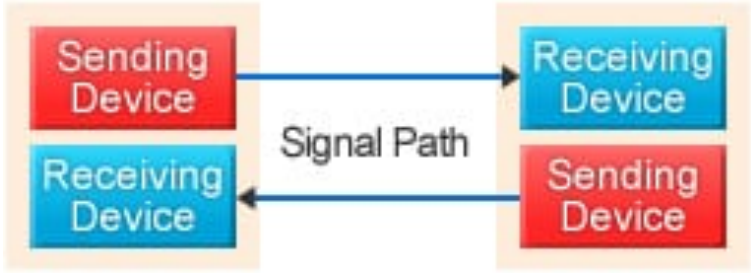
EXAMPLES OF SERIAL COMMUNICATIONS STANDARDS

- UART: Universal Asynchronous Receiver Transmitter
- SPI: Serial Peripheral Interface
- I2C: Inter-Integrated Circuit
- USB: Universal Serial Bus
- Ethernet

DIRECTION OF COMMUNICATION

- **Simplex:** the communication occurs in a single direction – one device transmits and the other receives.
- **Half-Duplex:** the communication occurs in both directions but not simultaneously. Both communicating devices (DevA and DevB) have transmitters and receivers. At a given time, either DevA transmits and DevB receives or vice-versa. A single wire is needed to carry the communication.
- **Full-Duplex:** the communications occurs in both directions and can be simultaneous. Usually two wires are used: one to transmit from DevA to DevB and another to transmit from DevB to DevA.
  rem: a transceiver consists of a transmitter and a receiver.





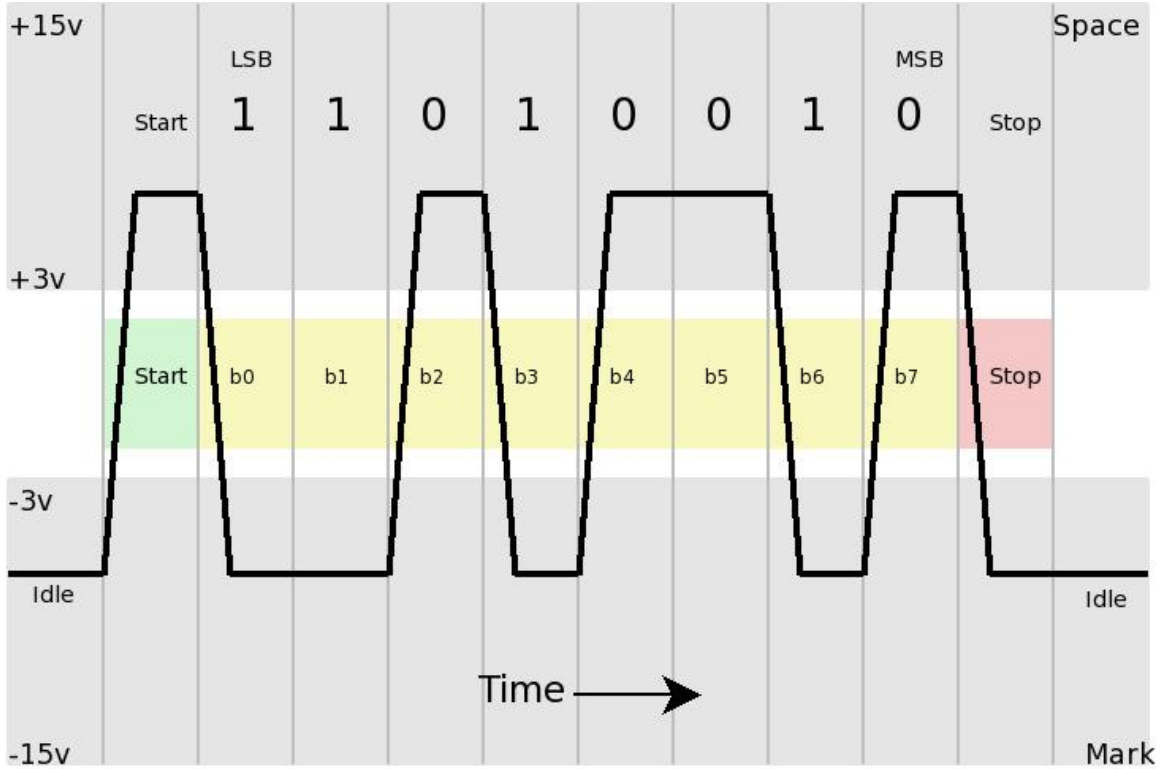
source: Renesas

SYNCHRONOUS VS ASYNCHRONOUS COMMUNICATIONS

- **Synchronous:** there is clock signal that identifies the time instances when a data bit is valid. Thus, the receiver uses the clock signal to recover the transmitted information. SPI and I2C are examples of synchronous communication protocols.
- **Asynchronous:** there is no common clock signal, thus, the two communicating devices must previously agree on a mechanism to identify each bit in the data stream. Therefore, the synchronization information must be embedded in the data signal. Quite often there are transitions in the data signal to identify the time slot of each bit in the data stream. RS-232 is an example of asynchronous communication.

BIT RATE VS BAUD RATE

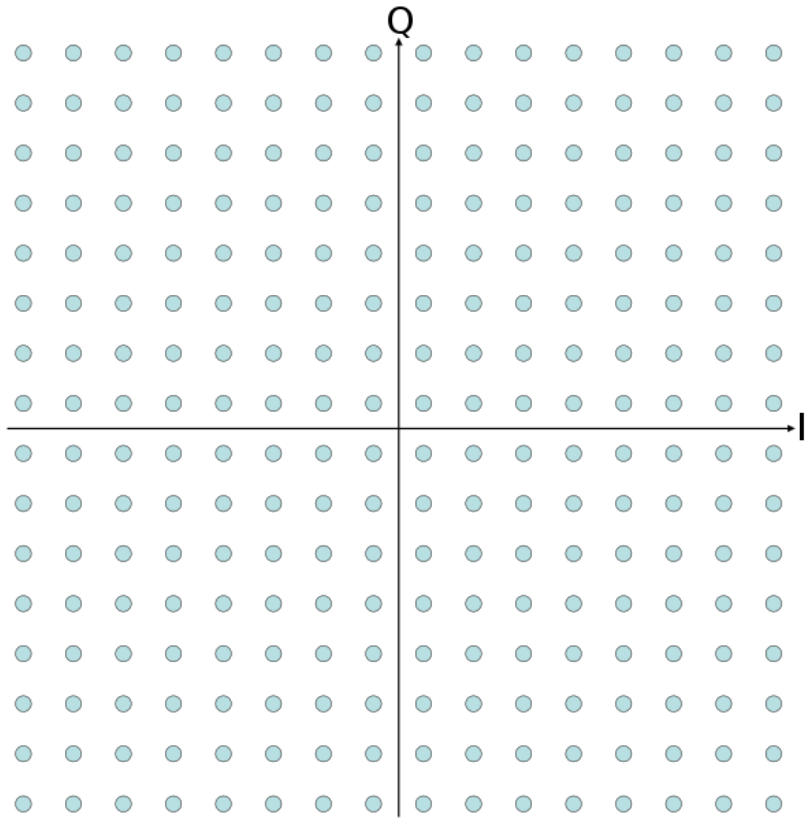
- **Symbols:** when transmitting signals over a wire, each possible combination of amplitude, phase and frequency is called a symbol. Simple schemes use only two symbols: for instance 0V and 3.3V to represent 0 and 1, or the coding used by RS-232 where amplitudes from -3 to -15 represent a 1 while +3 to +15 represent a 0.



source: commons.wikimedia.org (CC)

BIT RATE VS BAUD RATE

- **Symbols(cont):** a much larger set of symbols is also possible, for instance, 256QAM, one of the many encodings used for high speed ethernet, has 256 possible symbols, hence, each symbol encodes 8 bits.
- **Bit rate:** is the number of bits that are transmitted per time unit. Expressed in bits/s.
- **Baud rate:** is the number of symbols that are transmitted per time unit. Expressed in baud/s.
- If a symbol encodes a single bit, such as the case for RS-232, then the baud rate and the bit rate are the same.
  Yet, for 256QAM, the bit rate is 8 times higher than the baud rate.



source: commons.wikimedia.org (CC)

8.2 – UART

A UART is a peripheral present in most MCUs. Typically it receives data from the processor over the bus, hence, in parallel format, and handles the serialization and frame formatting.

A UART (Universal Asynchronous Receiver Transmitter) is capable of transmitting asynchronous data frames to another device as well as receiving. Both devices must be configured for the same speed and frame format.

Typical speeds used for asynchronous serial communication are: 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 bits per second. Higher speeds may also be used as long as agreed among transmitter and receiver.

Typically, the bit encoding defined by the RS-232 standard is used.

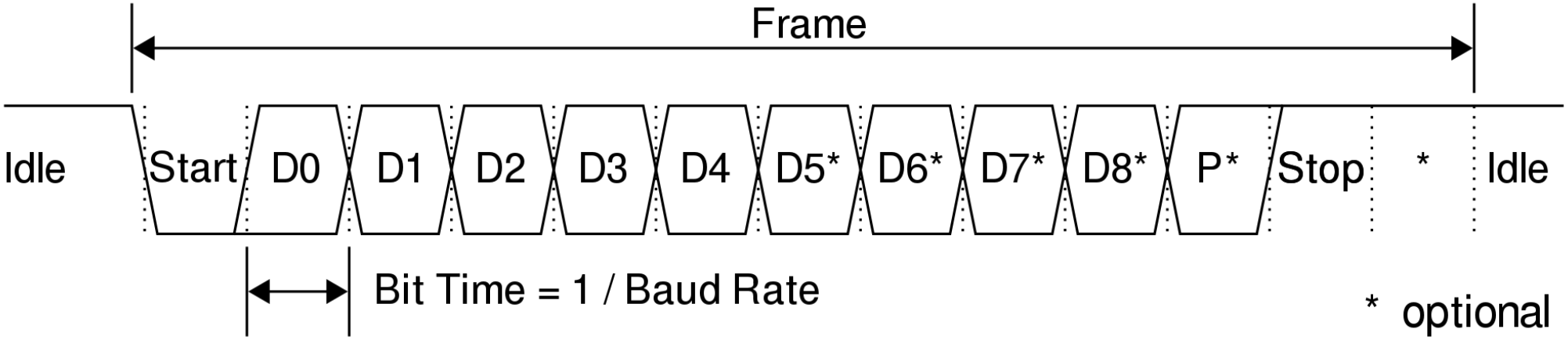
RS-232 defines separate lines for transmission and reception, thus, it is **full duplex** communication.

UART

Asynchronous frame format – consists of a start bit, data bits, an optional parity bit and stop bits.

The frame format is configurable:

- number of data bits: 5 to 8,
- number of stop bits: 1, 1.5 or 2,
- parity: none, odd, even.



source: commons.wikimedia.org (CC)

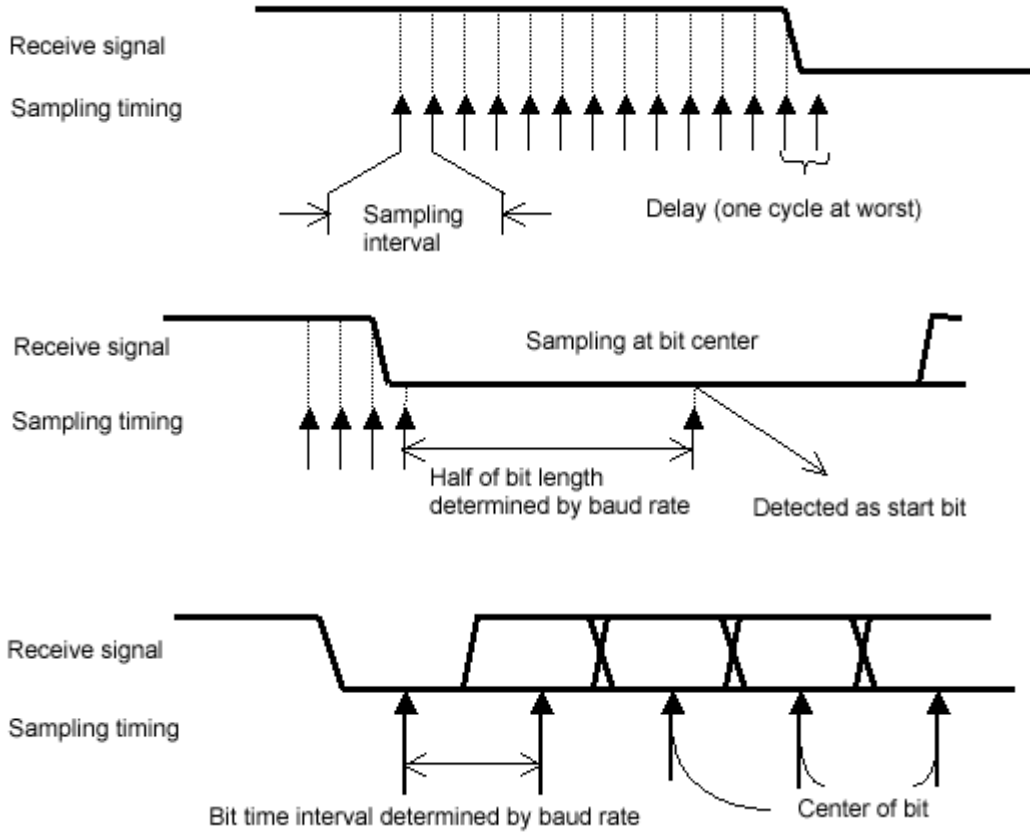
UART

How does the receiver synchronize with the transmitter?

1. Both the transmitter and the receiver are configured in the same way: speed and frame format.
2. Maximum clock skew allowed between the two sides is typically lower than 2%.
3. Receiver samples at a higher rate (e.g. 16 times de baud rate) for the start-bit transition. A delay of half-bit period determines the mid-bit position. From then on, sample every one-bit time.

UART

How does the receiver synchronize with the transmitter?

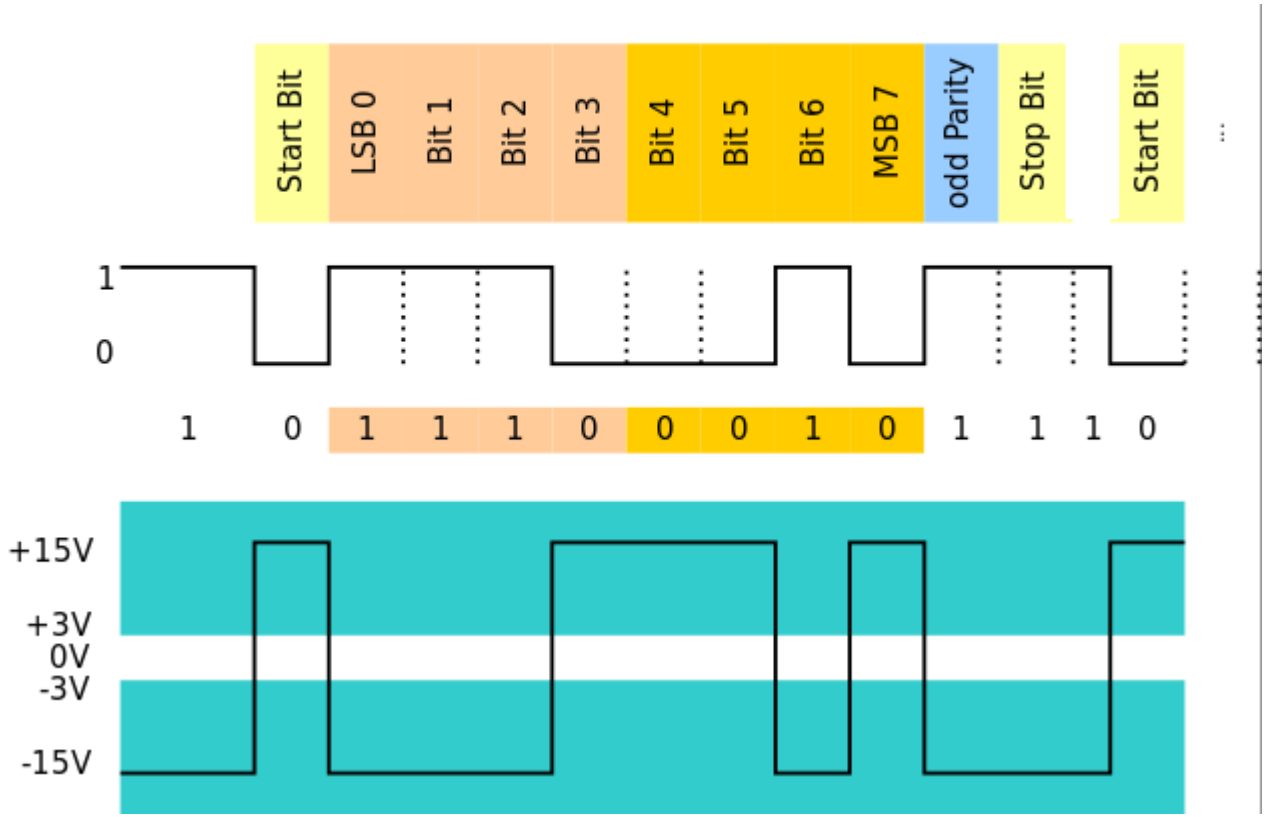


source: Renesas

UART

Example of the transmission of the character G (ASCII code 0x47) over an asynchronous line. Note that the LSb is transmitted first.
Upper figure shows UART levels while lower figure shows RS-232 levels.

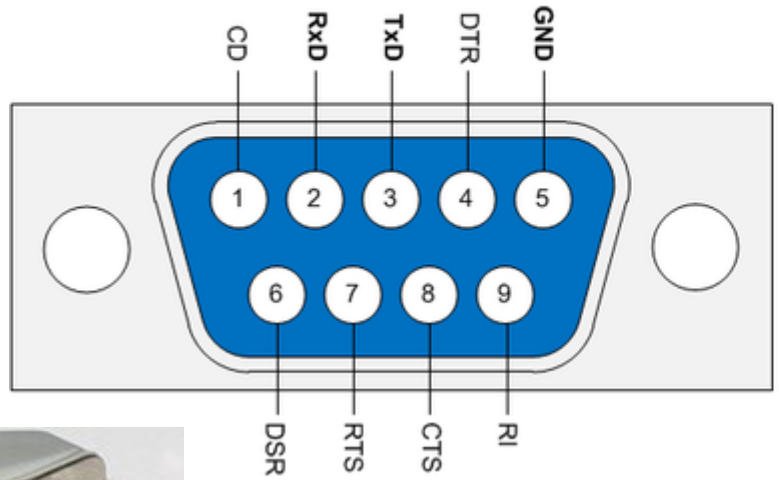
Configuration: 8O1.5 (eight bits, odd parity, 1.5 stop bits).
For a 9600 bps, the duration of each bit is 104.16 us.



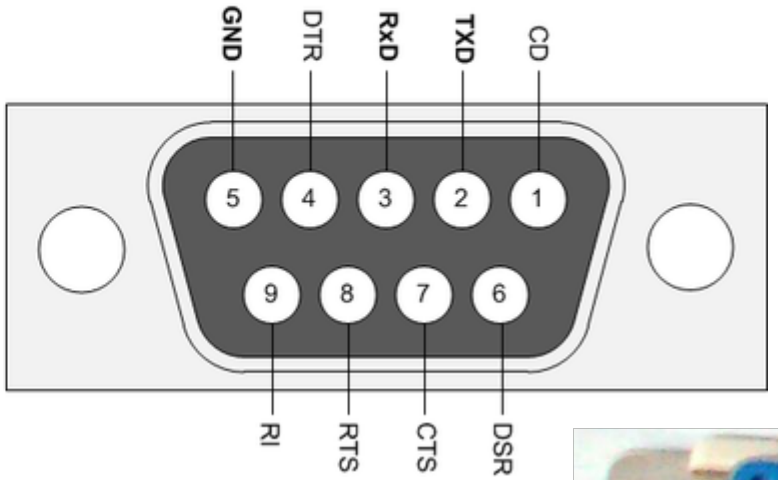
source: commons.wikimedia.org (CC)

UART

A frequently used connector for RS-232 is the DB-9. Shown here are the signals on each pin. TxD carries the transmitted data and RxD carries incoming data to the receiver.



male



female

source: commons.wikimedia.org (CC)

8.3 – SPI: SERIAL PERIPHERAL INTERFACE

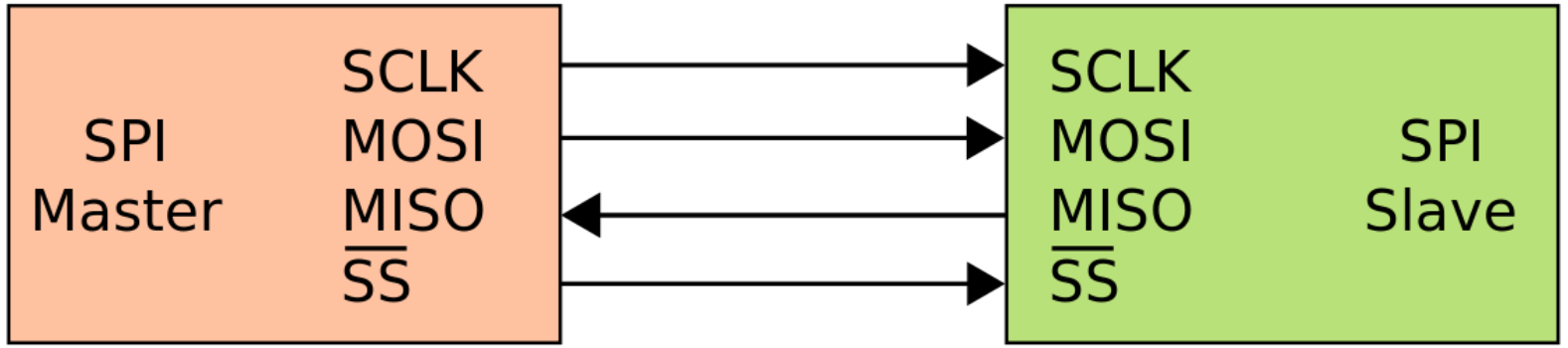
SPI = Serial Peripheral Interface

- Is a **synchronous** serial communication intended to be used to connect an MCU to external memory devices and peripherals such as: Flash EEPROM, ADC, DAC, temperature sensor, digital potentiometer, Real-Time Clock, …
- The **topology** is based in master and slave devices. There can be a single master, but multiple slaves are allowed.
- It uses 4 wires: data from master to slave, data from slave to master, clock and slave select. It is **full duplex** communication.
- There are many possible configurations; it has been reported that not all SPI devices are compatible, i.e. have a common configuration.

SPI

Single-slave connection:

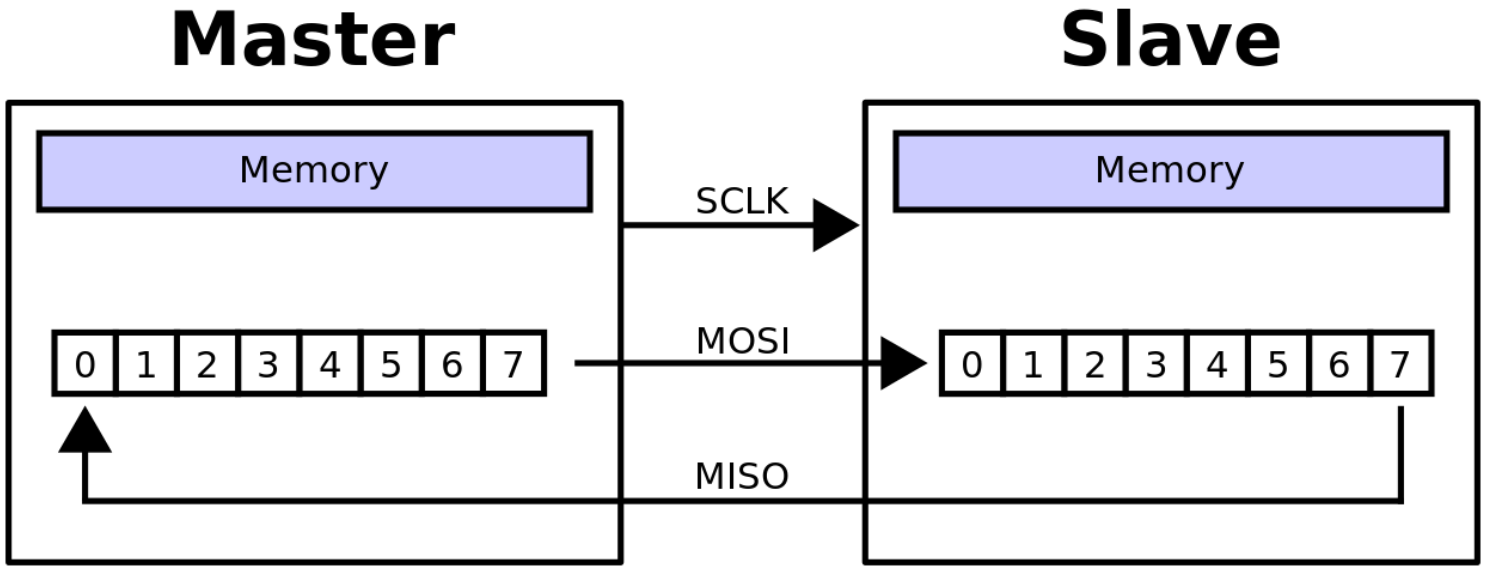
- SCLK: Serial Clock. Generated by the master;
- MOSI: Master Out Slave In;
- MISO: Master In Slave Out;
- $\overline{SS}$: Slave Select (active low).

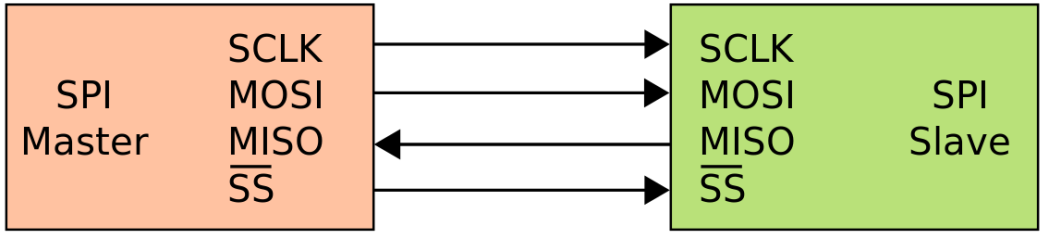


source: commons.wikimedia.org (CC)

SPI – OPERATION

- The Master selects a Slave by activating the /SS line.
- On every clock cycle, one bit is transferred from the master to the slave and another bit is transferred from the slave to the master.
  Not every transfer is significant.
- Typically the transfers occur in multiples of 8 bits.



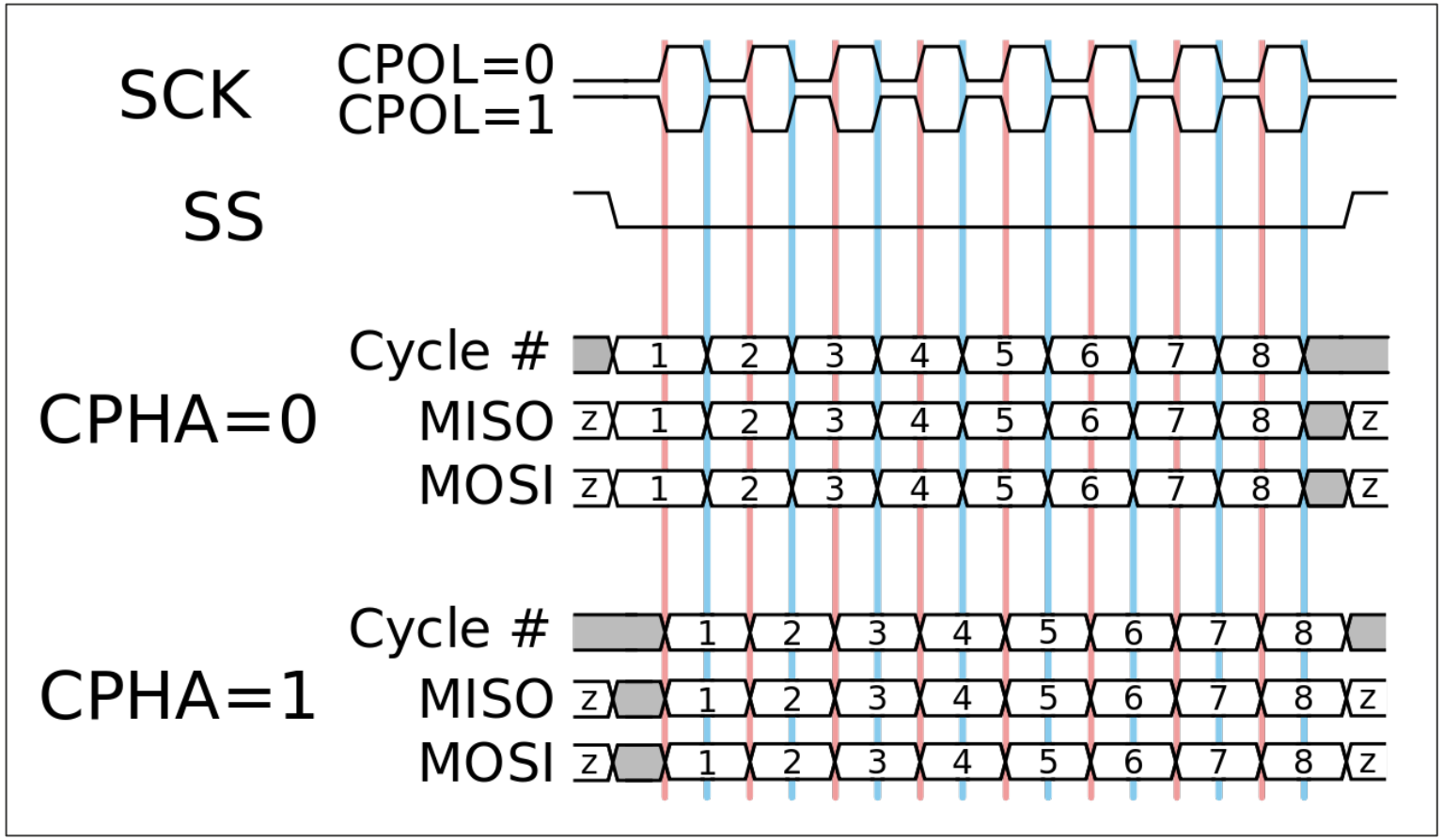


source: commons.wikimedia.org (CC)

SPI – CONFIGURATION

- CPOL:
  clock polarity selection
- CPHA:
  clock phase selection

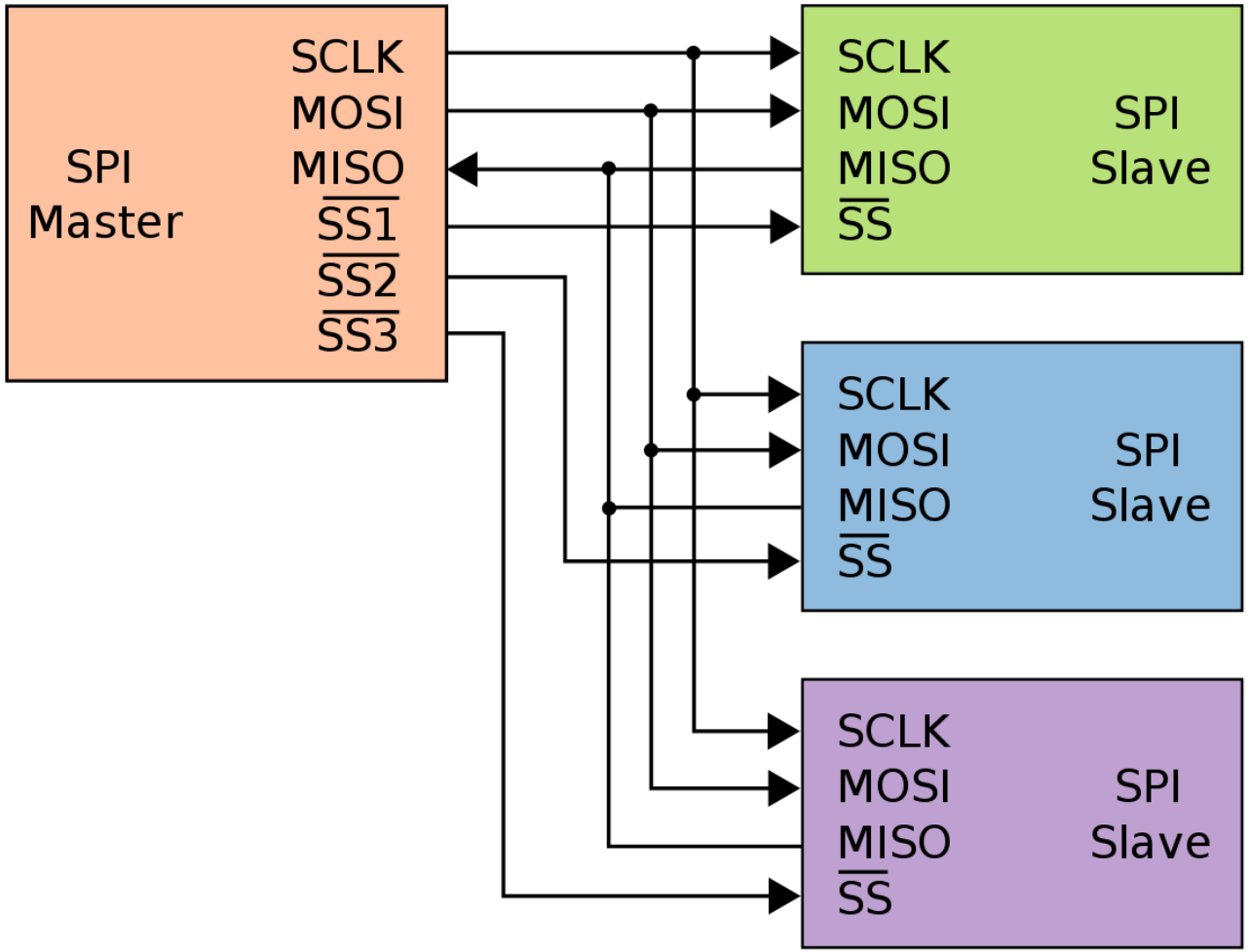
Observe that bits change on one clock edge and they are sampled on the other.



source: commons.wikimedia.org (CC)

SPI – MULTI-SLAVE

- MISO lines must be tri-state to be interconnected. They must go to high-impedance when the /SS line is not active.
- The Master selects one of the slaves to exchange data with it. Hence, separate Slave Select lines are required.



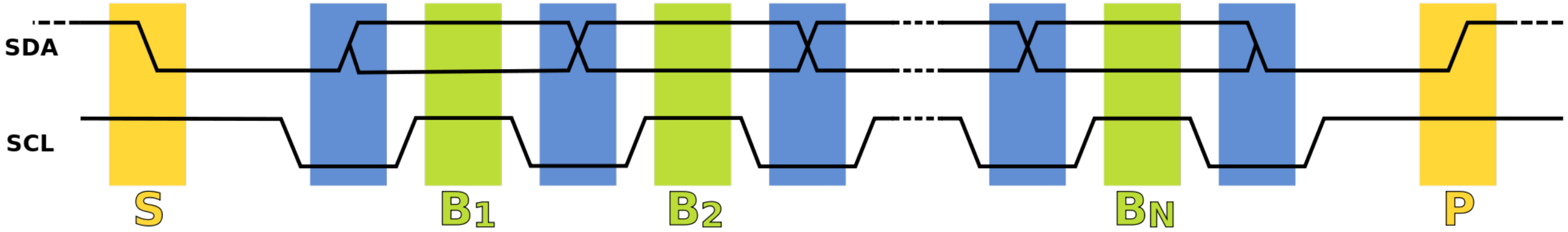
source: commons.wikimedia.org (CC)

8.4 – I2C: INTER-INTEGRATED CIRCUIT

- Synchronous half-duplex communications among multiple devices on a bus.
- Developed by a division of Philips in the 80's. This division is now NXP.
- Needs only two wires: data and clock.
- Bus drivers are open-drain with pull-up resistors, allows for multiple transmitters connected to the same line.
- Multiple masters are allowed on a bus. A master is the one that initiates a data transfer. Addressing modes use 7-bit and 10-bit addressing.
- Data rates: 100kbps, 400 kbps, 1 Mbps, 3.4 Mbps. (version 4 added a 5 Mbps data rate using push-pull drivers on a unidirectional bus).
- Recommended reading: UM10204 – I2C-bus specification and user manual. Rev 6 – 4-April-2014. NXP.

I2C – OPERATION

- The SCL line (Serial Clock) is driven by the bus master. When SCL is high the data line (SDA) is stable and can be read. When SCL is low then the data line can change.
- The exception of this rule is the signaling of the start-bit (negative edge of SDA while SCL is high) and stop-bit (positive edge of SDA while SCL is high).



source: commons.wikimedia.org (CC)

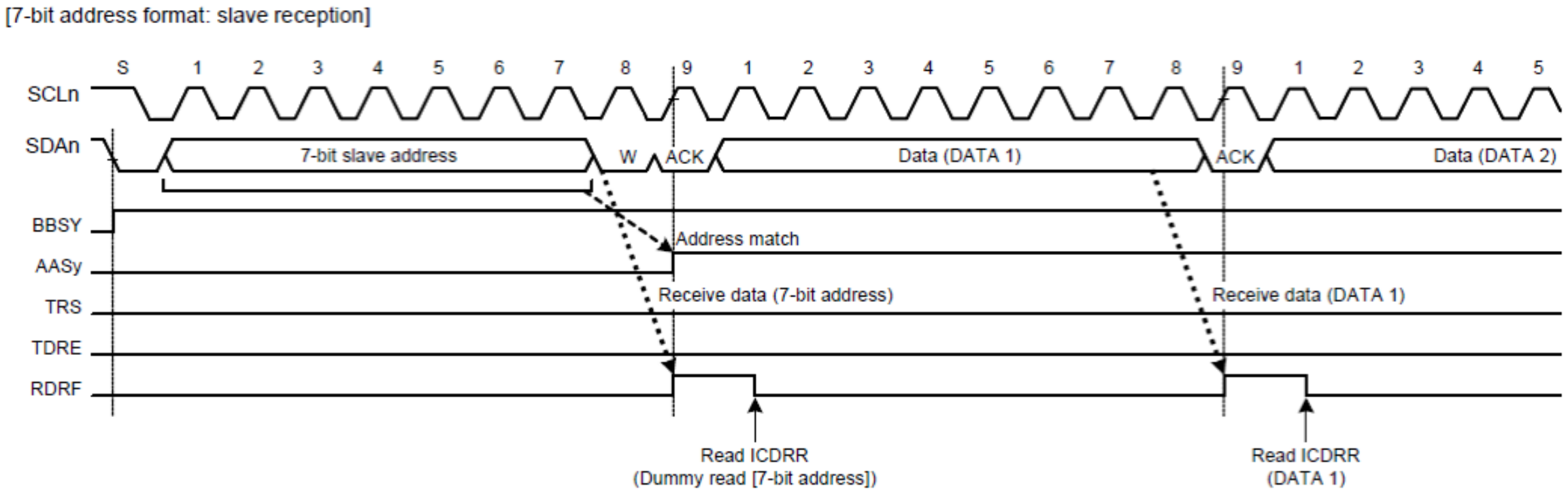
I2C – OPERATION (7-BIT ADDRESSING MODE, MASTER WRITE)

1. Master: sends the start bit.
2. Master: sends 7-bit slave address.
3. Master: sends the direction bit (0 = write).
4. Addressed slave: sends an ACK bit (0 = acknowledge).
5. Master: sends 8-bit data.
6. Addressed slave: sends ACK bit.
7. repeat steps 5 and 6 while there is data to transmit.
8. Master: sends stop bit.

I2C – OPERATION (7-BIT ADDRESSING MODE, MASTER WRITE)

- **I2C peripheral of Renesas S7G2 MCU**

  Timing diagram of a master sending data to a slave:



source: Renesas S7 Series Microcontrollers User's Manual

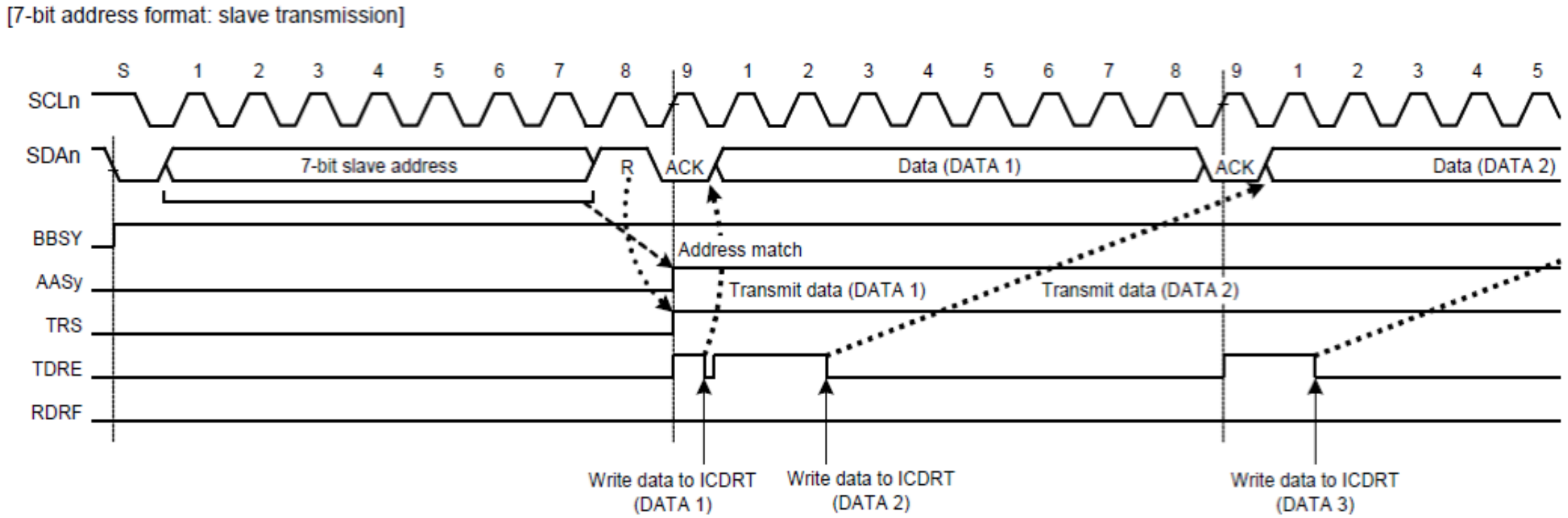
I2C – OPERATION (7-BIT ADDRESSING MODE, MASTER READ)

1. Master: sends the start bit.
2. Master: sends 7-bit slave address.
3. Master: sends the direction bit (1 = read).
4. Addressed slave: sends an ACK bit (0 = acknowledge).
5. Addressed slave: sends 8-bit data.
6. Master: sends ACK bit.
7. repeat steps 5 and 6 while there is data to transmit.
8. Master: sends stop bit.

I2C – OPERATION (7-BIT ADDRESSING MODE, MASTER READ)

- **I2C peripheral of Renesas S7G2 MCU**

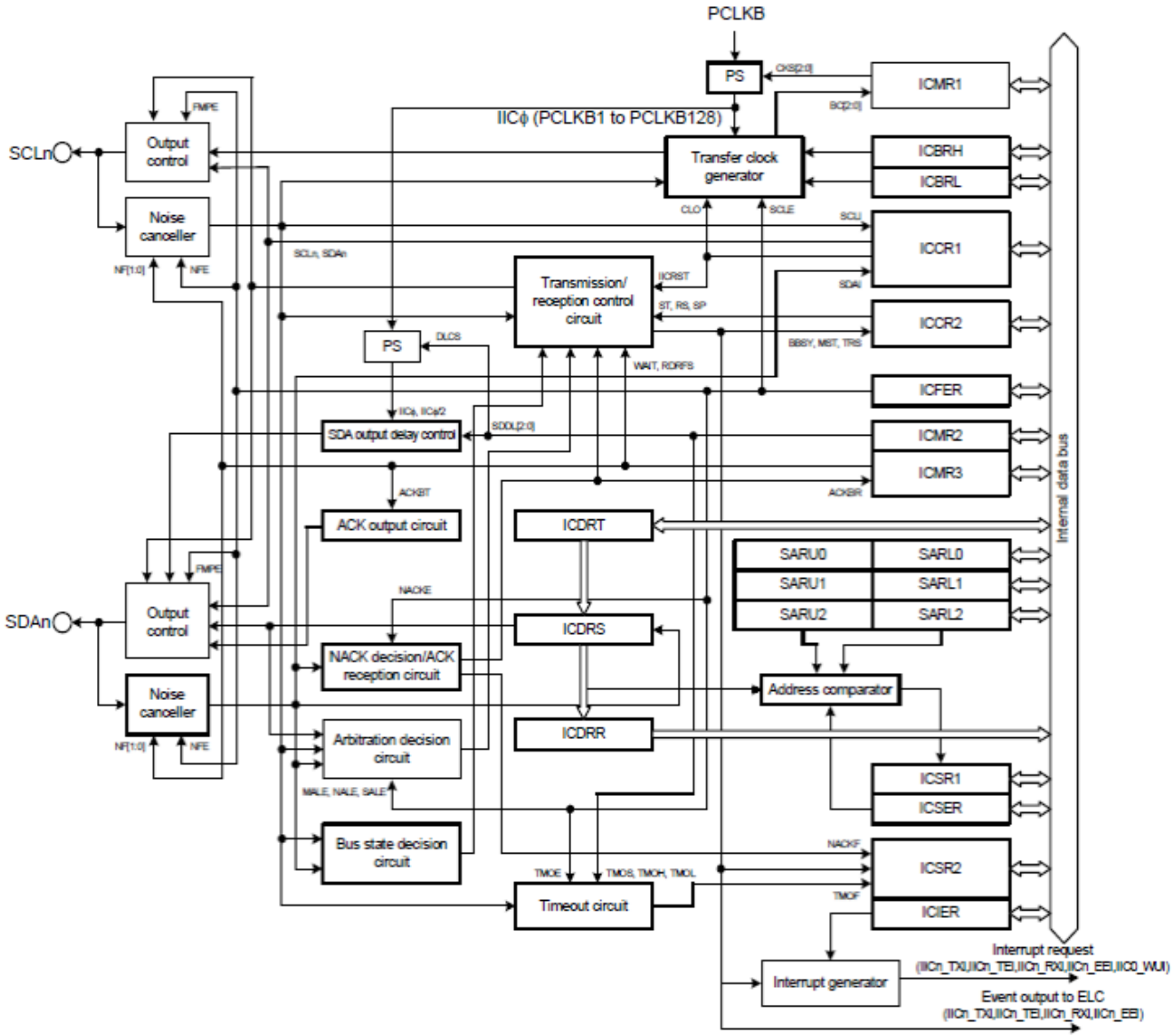
  Timing diagram of a slave sending data to a master:



source: Renesas S7 Series Microcontrollers User's Manual

I2C CASE STUDY – S7G2 IIC

Register	Name
ICCRx	I2C Control Register 1,2
ICMRx	I2C Mode Register 1,2,3
ICFER	I2C Function Enable Register
ICSER	I2C Status Enable Register
ICIER	I2C Interrupt Enable Register
ICSRx	I2C Status Register 1,2
ICWUR	I2C Wakeup Unit Register
SARLx	Slave Address Register L 0,1,2
SARUx	Slave Address Register U 0,1,2
ICBRL	I2C Bit Rate Low-Level Register
ICBRH	I2C Bit Rate High-Level Register
ICDRT	I2C Transmit Data Register
ICDRR	I2C Receive Data Register
ICDRS	I2C Shift Register



source: Renesas S7 Series Microcontrollers User's Manual

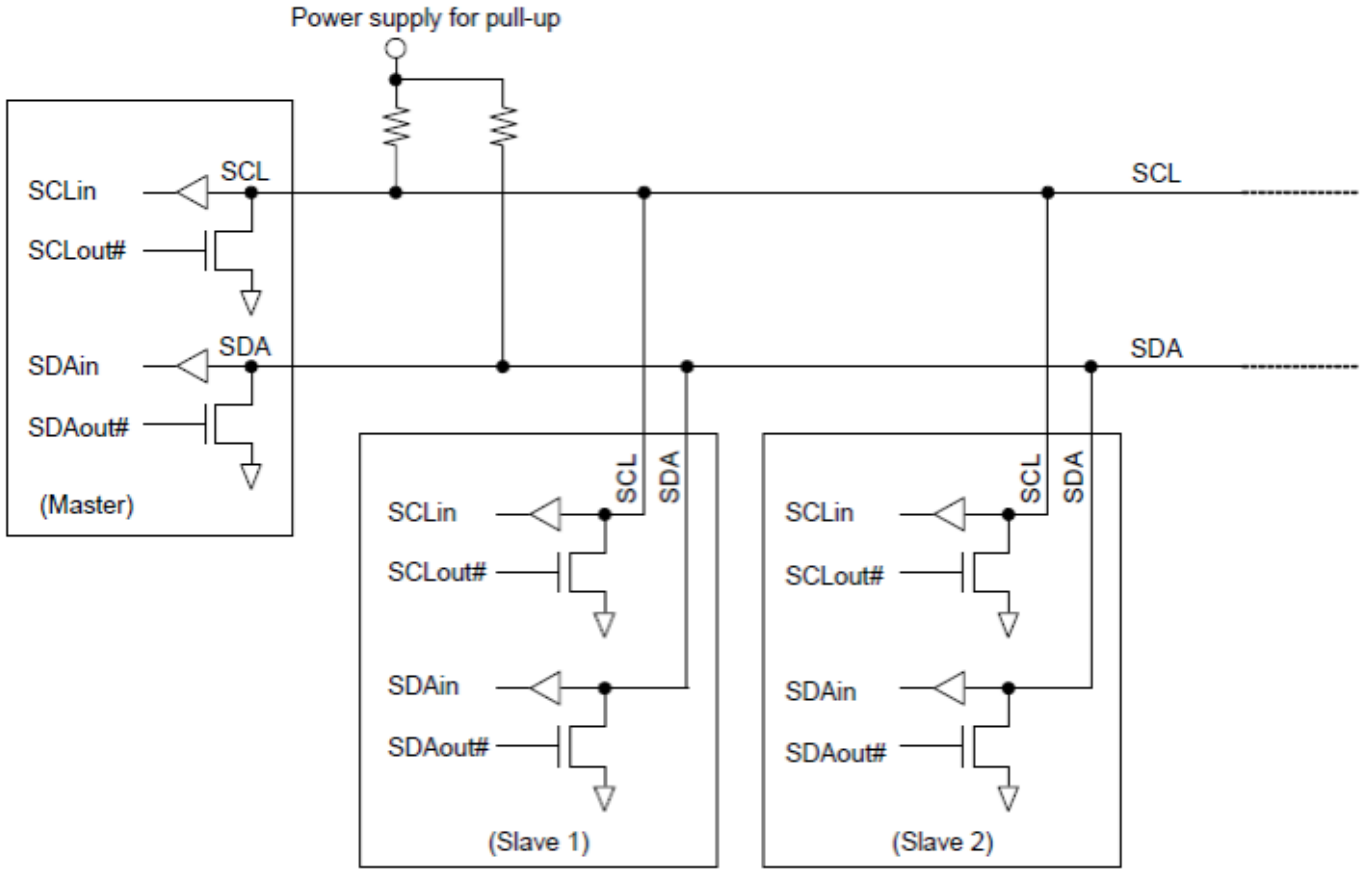
I2C CASE STUDY – S7G2 IIC

- Characteristics of the I2C Bus Interface of the Renesas S7G2 MCU:
- May operate as master or as slave of an I2C bus with data rates up to 1 Mbps.
- Up to 3 different slave addresses may be configured, 7-bit or 10-bit.
- Digital noise filters for SCL and SDA signals.
- Four interrupt sources: Receive data full, Transmit data empty, Transmit end, Error (NACK, timeout, …).

I2C CASE STUDY – S7G2 IIC

Connection of multiple devices on the I2C bus.

Notice open-drain outputs and pull-up resistors forming a wired AND.



source: Renesas S7 Series Microcontrollers User's Manual

Renesas.com