Inter-Integrated Circuit

I2C is a common serial protocol that is primarily used for short-distance data communication.

It is a two-wire synchronous master-slave protocol that supports communication between more than two devices.

- **Bidirectional**: Both master and slave devices can send and receive data, but only one device can transmit at a time (half-duplex).
- **Multi-Master Support**: I2C supports multiple masters and allows for dynamic bus control and arbitration.

The two wires of I2C are as follows:

1. **Serial Data Line (SDA)**: Used to transfer data between devices on the bus.
2. **Serial Clock Line (SCL)**: Used to synchronize data transfer, providing the clock signal for communication.

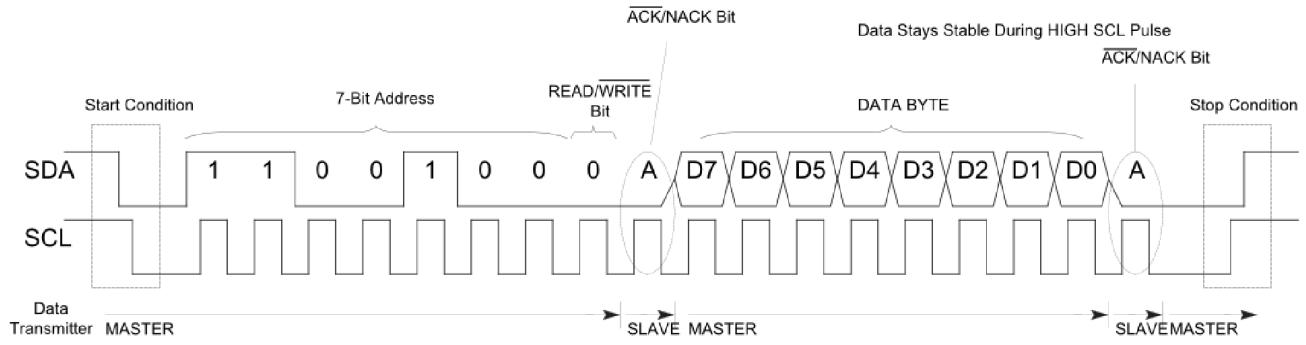
Both of these wires must be connected to a pull-up resistor to maintain a high voltage state when no device is pulling the line low.

- Note that there is only one pull-up resistor per line, not per device, regardless of how many devices are on the bus.

I2C Frames

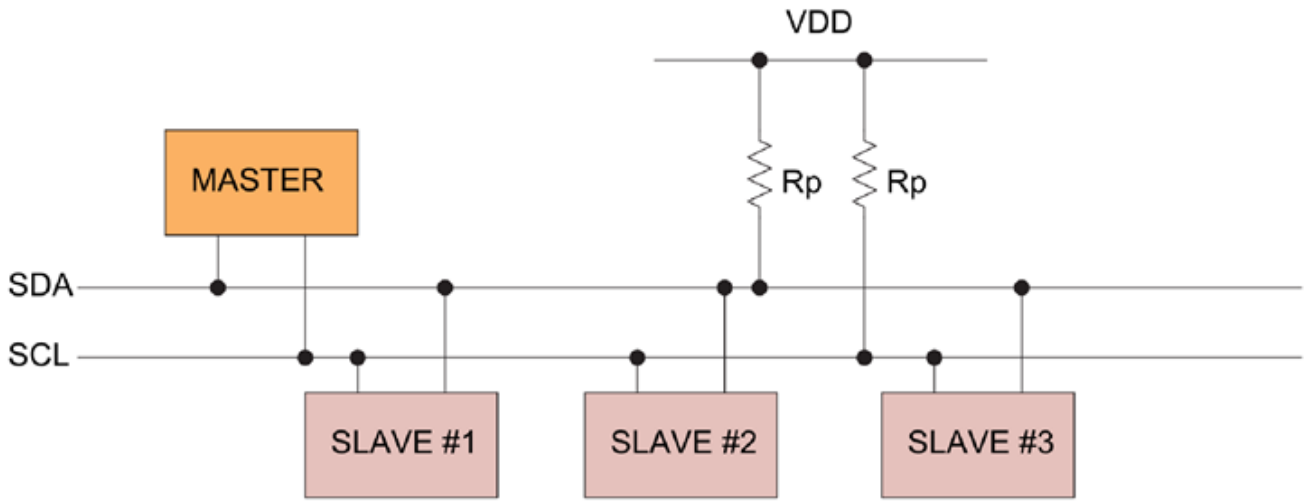
- **Start Condition**: The master initiates communication by pulling the SDA line low while the SCL line remains high. This signals the start of a transmission.

- **Address**: The 7-bit (or sometimes 10-bit) address of the target device is sent on the bus, identifying the intended recipient.
  - The general call address (0×00) allows the master to address all devices on the bus simultaneously.
- **Read/Write Bit**: A single bit is sent after the address to indicate the operation. `1` means a read operation, and `0` means a write operation.
- **ACK**: The slave device pulls the SDA line low for one bit to acknowledge it has received the address. If the line remains high, it means no device responded to the address (NACK).
- **Data Frame(s)**: Data is sent in 8-bit frames (fixed size). Multiple frames can be sent sequentially, separated by one clock pulse, to allow for acknowledgment.
  - Often in write operations, the first data frame contains the address of a register on the receiving device, and subsequent frames contain the data to be written to that register.
- **ACK(s)**: After each data frame, the receiving device (slave or master) pulls the SDA line low to acknowledge successful reception. If no ACK is sent, communication stops.
- **Stop Condition**: The master ends the communication by pulling the SDA line high while the SCL line is high, signaling the release of the bus.



Open-Drain Design

The pull-up resistors on the SDA and SCL lines are essential for maintaining the high state of the bus when no device is actively pulling the lines low.



The speed at which data can be transmitted is limited by the resistance of these resistors because they control how quickly the line returns to a high state after being pulled low.

- Higher resistance makes it easier to pull the line high but slows down the rise time, limiting the maximum data rate.
- Lower resistance allows faster rise times, enabling higher speeds, but increases power consumption and can strain devices.

Generally, resistances of **10 kΩ**, **4.7 kΩ**, and **2.2 kΩ** are used, depending on the speed of the bus and the total capacitance of the lines.

I2C has a maximum bus capacitance of **400 pF**, which limits the length of the wires and the number of connected devices.

- Exceeding this can cause communication errors.

The speed modes supported by I2C are as follows:

- **Standard Mode**: Up to 100 kHz.
- **Fast Mode**: Up to 400 kHz.
- **Fast Mode Plus**: Up to 1 MHz.
- **High-Speed Mode (Hs-mode)**: Up to 3.4 MHz.

Advanced Features

Clock stretching is a mechanism where a slave device holds the clock line (SCL) low to pause communication.

- This allows the slave extra time to process data or prepare for the next operation. The master must wait until the slave releases SCL, allowing it to go high, before continuing the transmission.

Repeated Start: I2C allows for a repeated start condition, enabling the master to retain control of the bus and start a new operation without releasing the bus (no stop condition in between).

Bus Arbitration: I2C allows for multiple masters on the same bus, with mechanisms in place to avoid overlapping transmissions.

- Since devices can only pull the SDA line low from its idle state (which is high), they rely on monitoring the bus to detect conflicts when transmitting simultaneously.
- When one device sends a `1` (high) but detects a `0` (low) from another device, it recognizes that it has lost arbitration.
- The losing master stops transmitting and waits for the bus to become idle before retrying.
- The device with the lowest binary value during arbitration wins, meaning I2C lacks a priority system like CAN.

Multi-master synchronization: Multiple masters synchronize their clocks by releasing the SCL line when another master pulls it low,

ensuring a unified clock signal on the bus.