I2C Protocol

Version 1.2
By Murtadha AlSaeedi

روابــي القابضة
Rawabi Holding

Revision History

Date	Rev	Details
February 6, 2014	1.2	Release

Table of Contents

1. Definition

I2C (Inter-Integrated Circuit, also known as Two Wire Interface) is a multi-master bus used to connect low-speed peripherals to an embedded system, motherboard or other digital electronic devices. I2C is a synchronous serial protocol uses only two wires to connect multiple masters and multiple slaves. One wire is called the Serial Clock Line (SCL) which is used for clocking, and the other one is called the Serial Data Line (SDA) which is used for exchanging data.

Term	Description
Transmitter	A device that send data through the bus
Receiver	A device that receives data through the bus
Master	A device that initiates and terminates data exchange sessions, and generates clock signals
Slave	A device addressed by a master
Multi-master	More than one master connected to the bus
Arbitration	A procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted
Synchronization	A procedure to synchronize the clock signals of two or more masters

TABLE 1: TERMS DEFINITIONS

2. Mechanism

In the standard configuration of I2C, the SCL and SDA lines are always pulled up (always HIGH) using pull-up resistors, however, the SCL line can instead be driven by a master. Devices on the bus can only pull the SDA line LOW, and can never drive it HIGH. Prior to any data exchange, a master sends a slave address followed by a write/read bit serially. Then the addressed slave starts collecting or transmitting data, depending on the value of the read/write bit, while the other slaves ignore the call. Exchanging data on the SDA line can be at rates up to 100 kbps in the Standard-mode, 400 kbps in the Fast-mode, 1 Mbps in the Fast-mode Plus and 3.4 Mbps in the High-speed mode. A slow slave may need to pull the SCL line LOW while it receives data, forcing the master to wait, then the master cannot proceeds before the slave releases the SCL line.

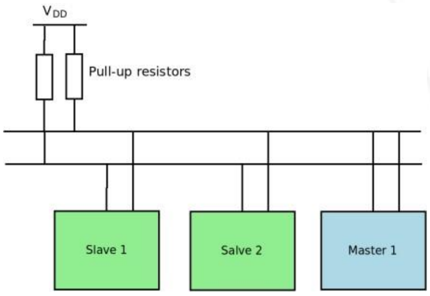


FIGURE 1

3. Addressing

Each device on the I2C bus should store a unique 7-bits address to identify itself on the bus. The lower address bits of some devices can be set to support multiple copies of the same device on the same bus. A device that initiates a data transfer and generates the SCL signal is considered as a master. On the other hand, devices addressed by a master are considered as slaves. It is worth noting that some devices can act as a master or as a slave, and some can only act as a slave. The maximum possible number of devices to connect to the bus is limited by the capacitance of the bus and available addresses.

4. Reserved Addresses

The 7-bit address allows 128 different addresses, however, some of these addresses are actually reserved for special purposes. The reserved addresses reduce the number of allowed addresses to 112 different addresses. If more addresses are needed, a 10 bits address scheme can be used.

Address	Purpose
0000000 0	General Call
0000000 1	Start byte for slow microcontrollers
0000001 X	CBUS addresses, a different bus protocol
0000010 X	Reserved for different bus formats
0000011 X	Reserved for future purposes
00001XX X	High speed master code
11110XX X	10-bit slave addressing
11111XX X	Reserved for future purposes

TABLE 2: RESERVED ADDRESS

5. Starting and Stopping a Data Exchange Session

Prior to any data exchange session, a master issues a start condition by pulling the SDA line from HIGH to LOW while the SCL line is HIGH. The start condition notifies all devices connected to the bus that a transaction is about to happen. All devices connected to the bus listen to the first message after a start condition.

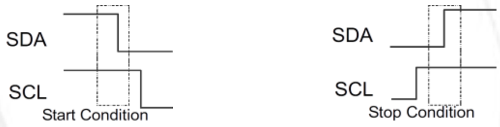


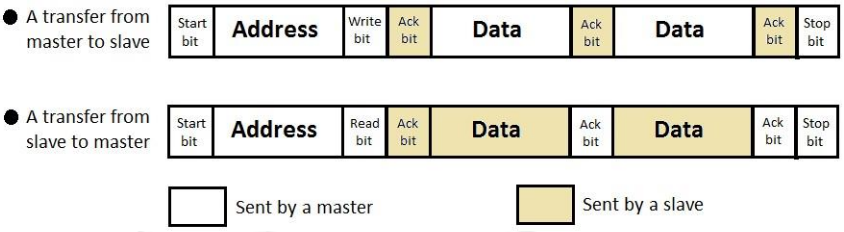
FIGURE 2

At the end of each data exchange session, the master started the session issues a stop condition by releasing the SDA line (transition from LOW to HIGH) while the SCL line is HIGH. The stop condition indicates to the devices on the bus the end of a data exchange session and that the bus is available again.

6. Exchanging Data (Standard-mode)

For any data exchange on the bus, the following steps should be followed:

1. A master issues a start condition
2. The master sends a slave address followed by a read/write bit (1 for reading, 0 for writing), and waits for an acknowledgement bit from the slave.
3. The addressed slave sends an acknowledgement bit (0) to indicate that it is available on the bus.
4. The master transmits a byte of data and waits for an acknowledgement from the addressed slave in case the read/write bit is 0; the addressed slave transmits a byte of data and waits for acknowledgment from the master in case the read/write bit is 1.
5. The receiver sends an acknowledgement bit (0).
6. Steps 4 and 5 are repeated to send more than one byte; otherwise the master terminates this session by issuing a stop condition.



● A transfer from master to slave	Start bit	Address	Write bit	▨ Ack bit	Data	▨ Ack bit	Data	▨ Ack bit	Stop bit
● A transfer from slave to master	Start bit	Address	Read bit	▨ Ack bit	▨ Data	Ack bit	▨ Data	Ack bit	Stop bit

□ Sent by a master ▨ Sent by a slave

FIGURE 3

7. Arbitration

In multi-master configurations, to determine which of the masters on the bus can claim it, a process called arbitration is used. Each master should monitor the bus to avoid collisions and act accordingly. A master on a multi-master bus should be able to do the following.

1. Bus busy detection: a master must be able to detect an ongoing data exchange session and not interrupt it. This is done by watching the start and stop conditions.
2. Arbitration logic following: If two masters issue a start condition at the same time, the wining master owns the bus and the other immediately discontinuous its session. Each master watches the SDA line and checks if the signal on it matches the signal it is generating. If the signal on the SDA line is LOW when it should be HIGH, then this master has lost arbitration.

Note that not all master devices support a bus with a multi-master configuration.

8. Synchronization

Master devices must synchronize their clocks for successful arbitration; to generate a synchronized clock signal on the SCL line. A device with the longest LOW period on the SCL line determines the length of the LOW period of the synchronized clock, while a device with the shortest HIGH period on the SCL line determines the length of the HIGH period of the synchronized clock.

9. Bus Capacitance

The capacitance of the bus increases as the number of devices connected to the bus increases. The maximum allowable capacitance of the I2C bus is 400 pF. This limitation is due to the fact that the bus needs some time to charge/discharge (represent a HIGH or LOW).

10. Data Rate and Pull-up Resistors Relation

In Standard-mode of I2C, the value of pull-up resistors is affected by the supplied voltage, bus capacitance and data rate used on the bus. The minimum value for pull-up resistor can be determined from the following graphs with respect to the supplied voltage or to bus capacitance, where Rs is a protection resistor between the port of a device and the bus.

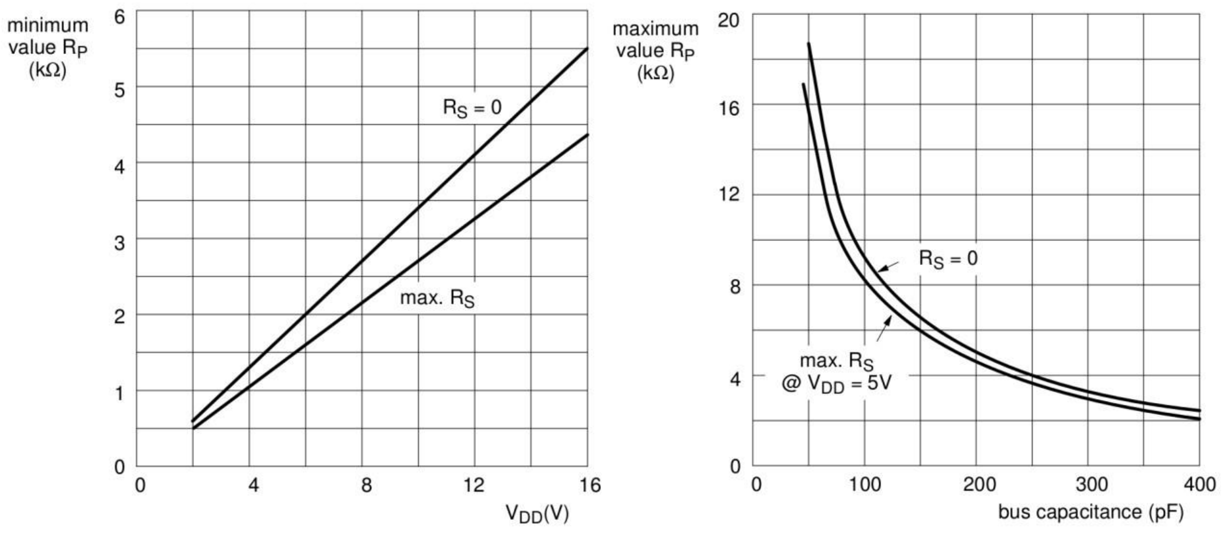


FIGURE 4

Note that the modes higher than standard-mode will need to charge and discharge the bus at a higher rate, which require smaller values for pull-up resistors to allow more current to flow through them.