# I2C Interfacing on eYFi-Mega Board

e-Yantra Team

Embedded Real-Time Systems (ERTS) Lab
Indian Institute of Technology, Bombay

IIT Bombay
July 6, 2022

# What is I2C?

# What is I2C?

Inter-Integrated Circuit, $I^2C$ or I2C or TWI

# What is I2C?

Inter-Integrated Circuit, $I^2C$ or I2C or TWI

- Serial and synchronous communication Protocol.

# What is I2C?

Inter-Integrated Circuit, $I^2C$ or I2C or TWI

- Serial and synchronous communication Protocol.
- Master-Slave, half duplex protocol.

# What is I2C?

Inter-Integrated Circuit, $I^2C$ or I2C or TWI

- Serial and synchronous communication Protocol.
- Master-Slave, half duplex protocol.
- Can be multi-master.

# What is I2C?

Inter-Integrated Circuit, $I^2C$ or I2C or TWI

- Serial and synchronous communication Protocol.
- Master-Slave, half duplex protocol.
- Can be multi-master.
- Ensures transmission by acknowledgment.

# Serial

# Serial

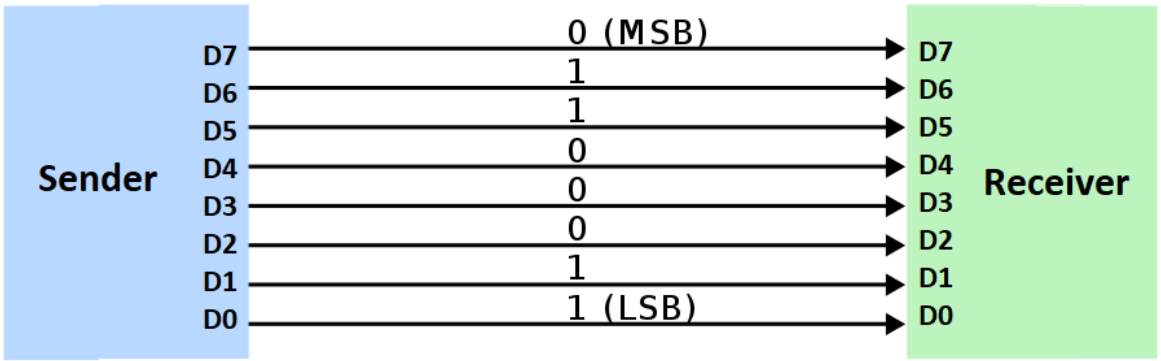



Figure: Parallel Communication

# Serial

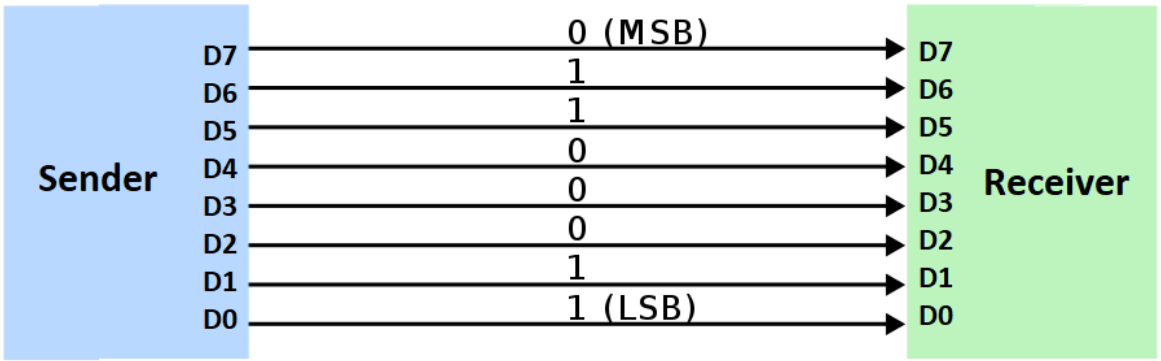



Figure: Parallel Communication

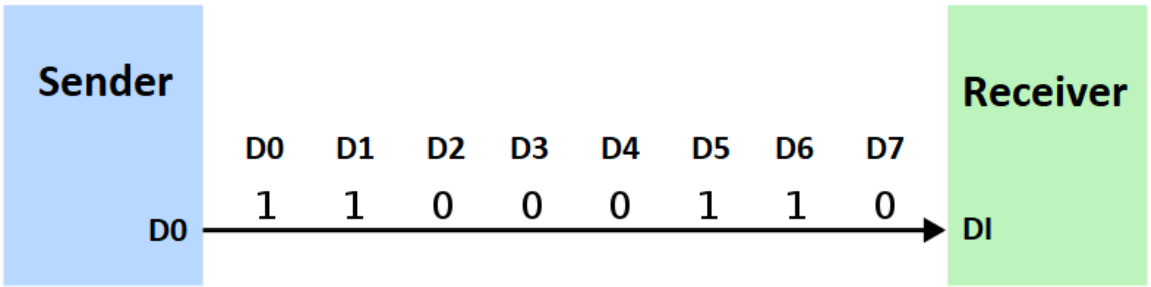



Figure: Serial Communication

# Serial

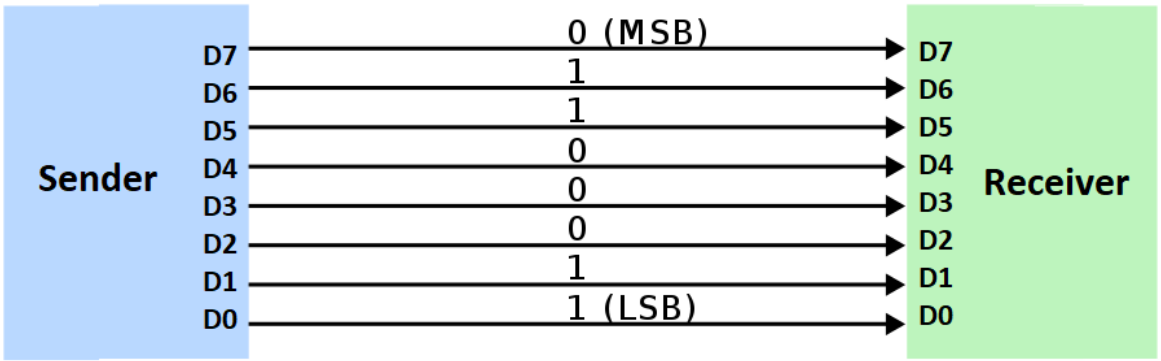



Figure: Parallel Communication

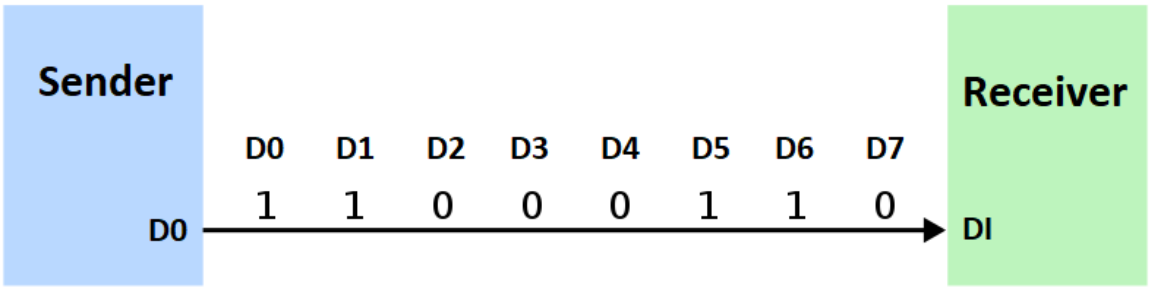



Figure: Serial Communication

One bit is sent at a time.

# Synchronous:

# Synchronous:

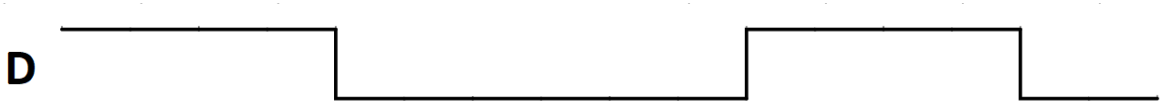



Figure: Data sent on Data Pin

# Synchronous:

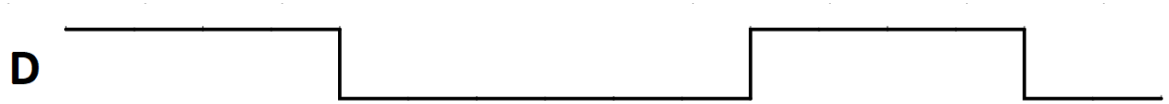



Figure: Data sent on Data Pin

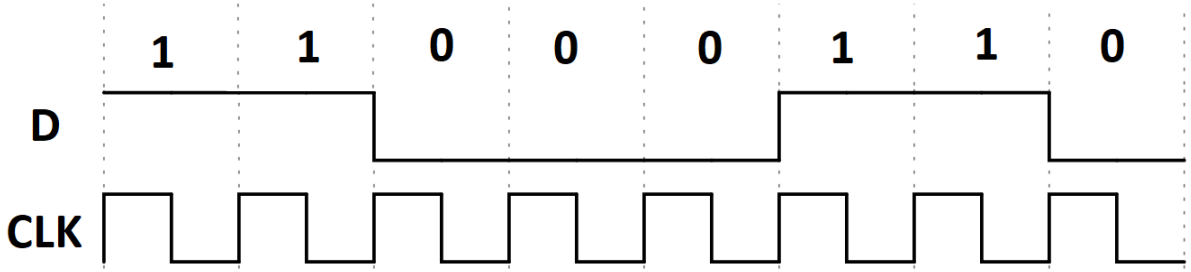



Figure: Data along with Clock

# Synchronous:

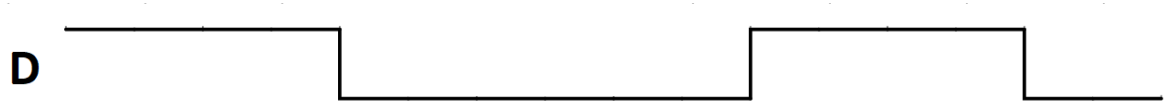



Figure: Data sent on Data Pin

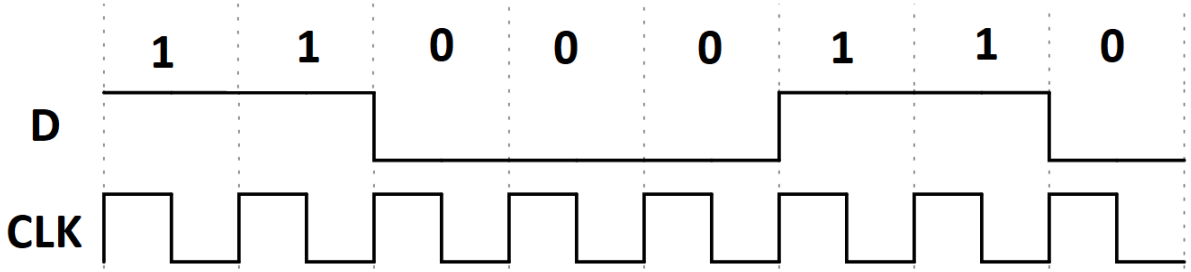



Figure: Data along with Clock

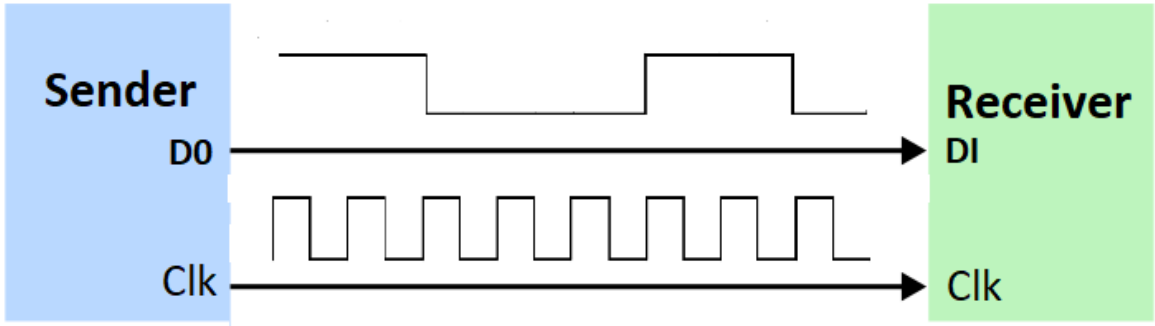



Figure: I2C Communication between Two Devices

# Master-Slave configuration:

# Master-Slave configuration:

Master is responsible for initiating a communication.

# Master-Slave configuration:

Master is responsible for initiating a communication.
Clock should be generated by Master only.

# Master-Slave configuration:

Master is responsible for initiating a communication.
Clock should be generated by Master only.

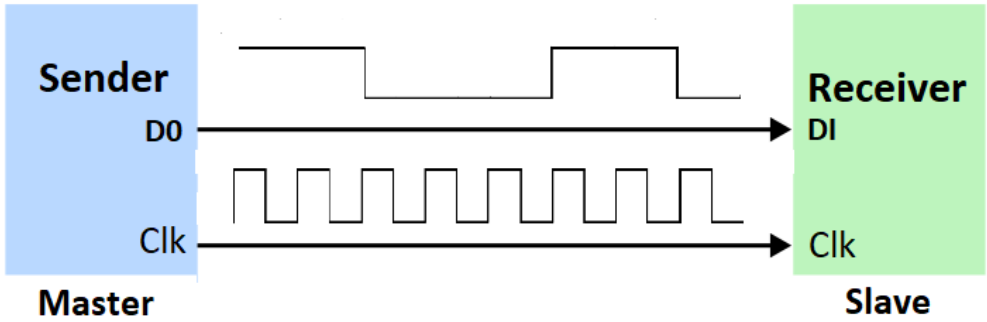



Figure: Master Transmitter Slave Receiver

# Master-Slave configuration:

Master is responsible for initiating a communication.
Clock should be generated by Master only.

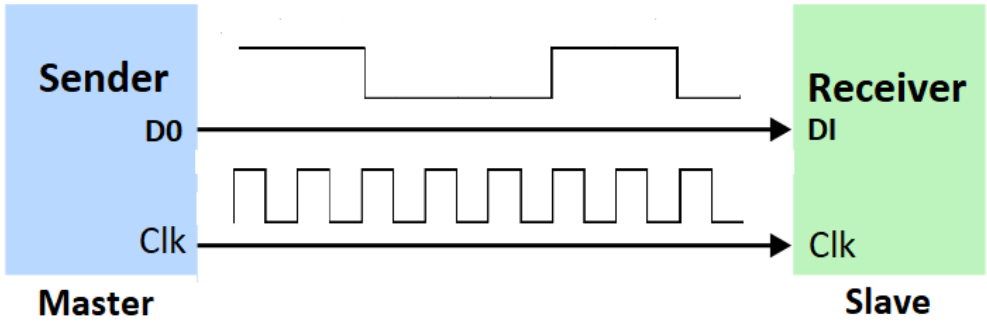



Figure: Master Transmitter Slave Receiver

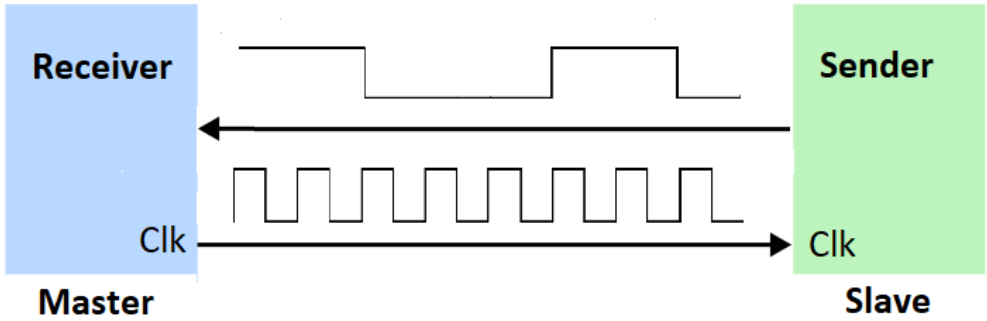



Figure: Master Receiver Slave Transmitter

# Master-Slave configuration:

Master is responsible for initiating a communication.
Clock should be generated by Master only.

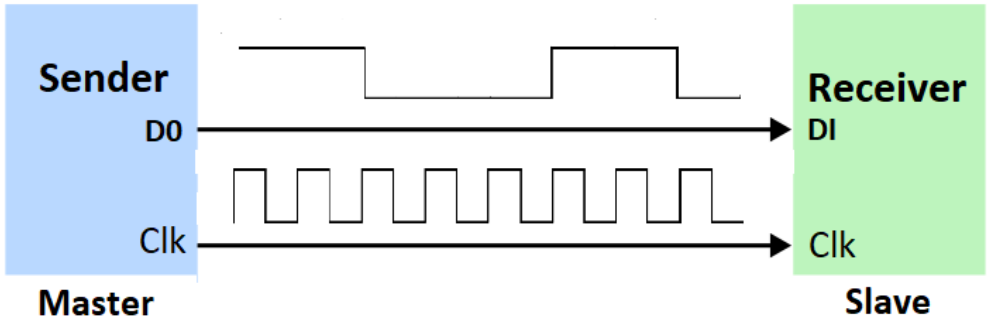



Figure: Master Transmitter Slave Receiver

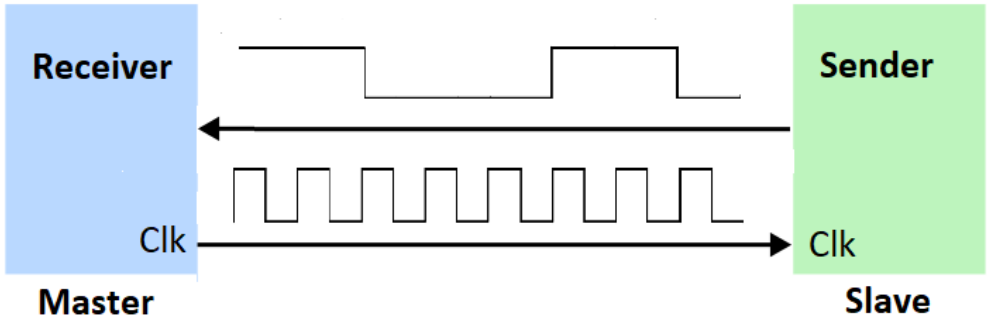



Figure: Master Receiver Slave Transmitter

I2C is a Half-Duplex communication.

# Multi-master and multi-slave

# Multi-master and multi-slave

We can connect upto 128 devices on I2C bus.

# Multi-master and multi-slave

We can connect upto 128 devices on I2C bus.

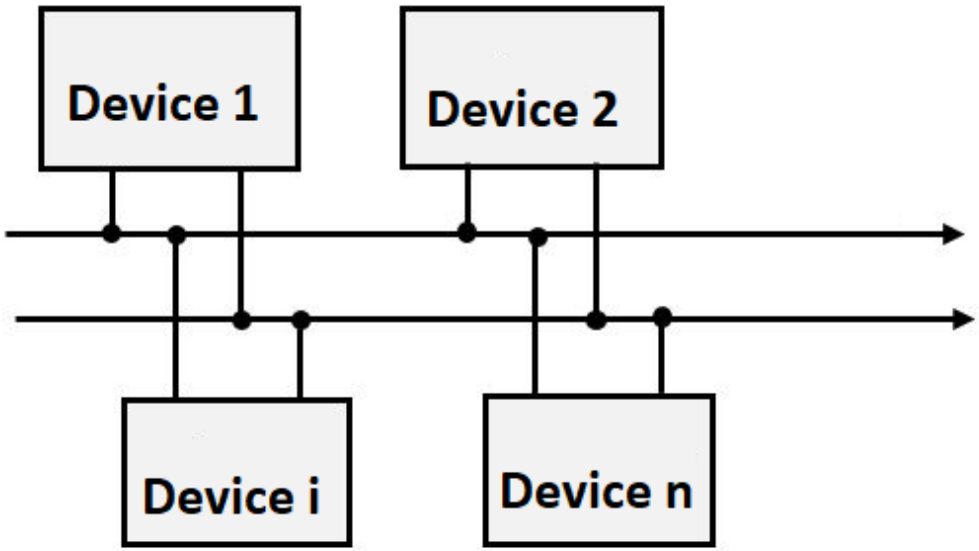



Figure: I2C Communication

# Multi-master and multi-slave

We can connect upto 128 devices on I2C bus.

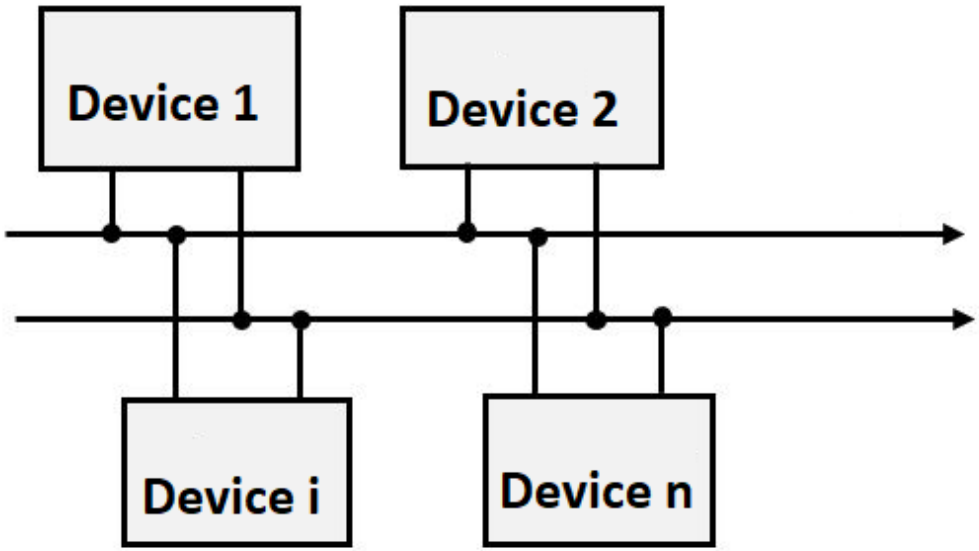



Figure: I2C Communication

Hence n can be maximum 128

# Multi-master and multi-slave

We can connect upto 128 devices on I2C bus.

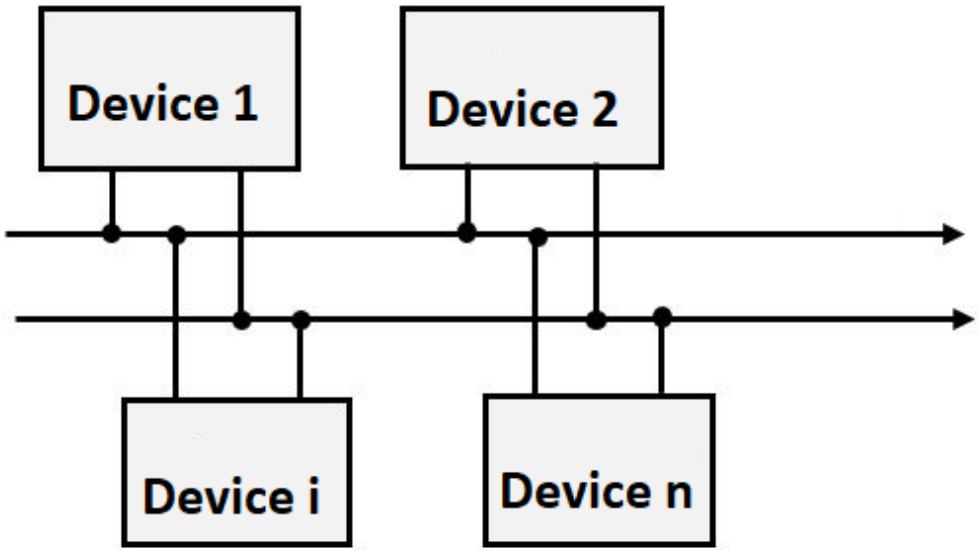



Figure: I2C Communication

Hence n can be maximum 128
At a time only one device will act as a master

# Multi-master and multi-slave

We can connect upto 128 devices on I2C bus.

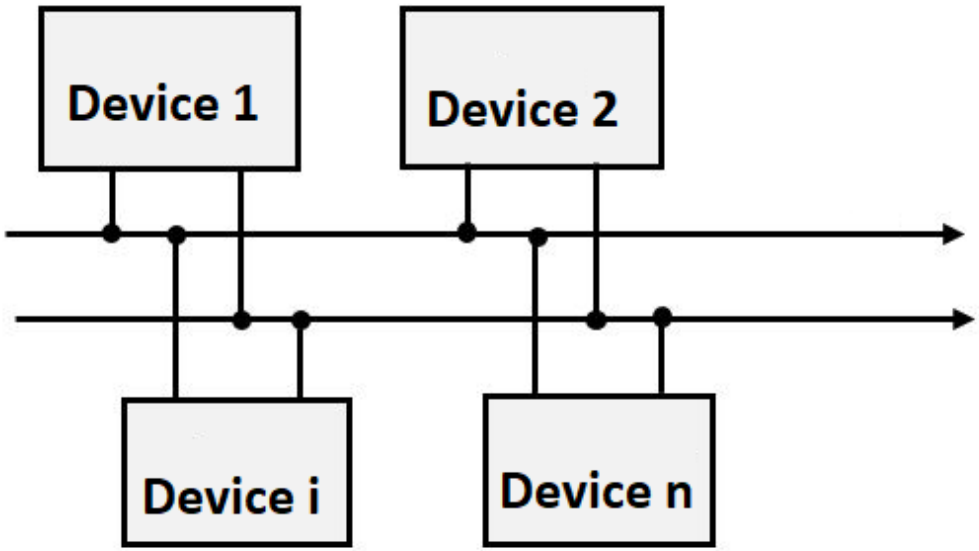
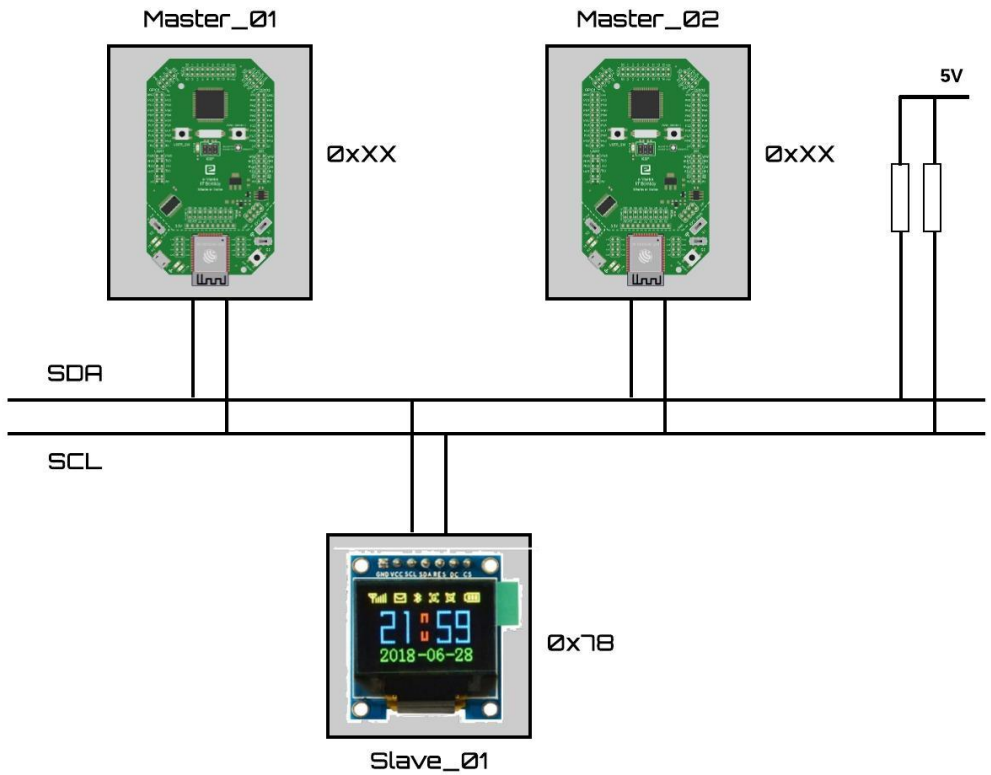



Figure: I2C Communication

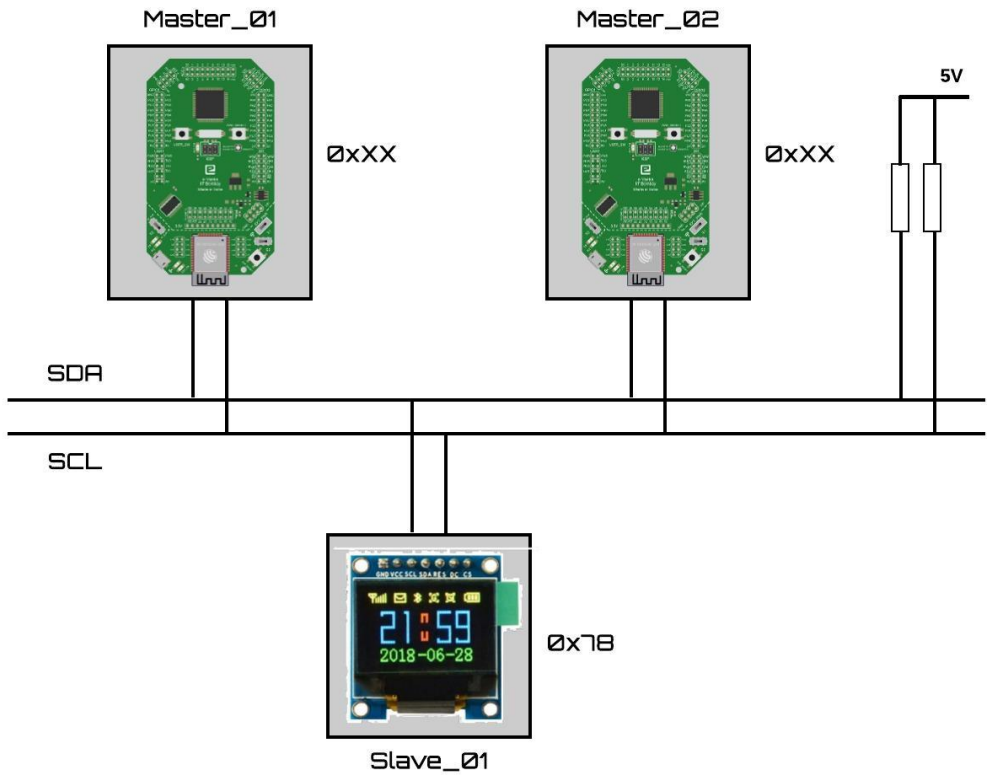
Hence n can be maximum 128
At a time only one device will act as a master
Each device in I2C is addressed by its unique address

# Connection Diagram:

# Connection Diagram:



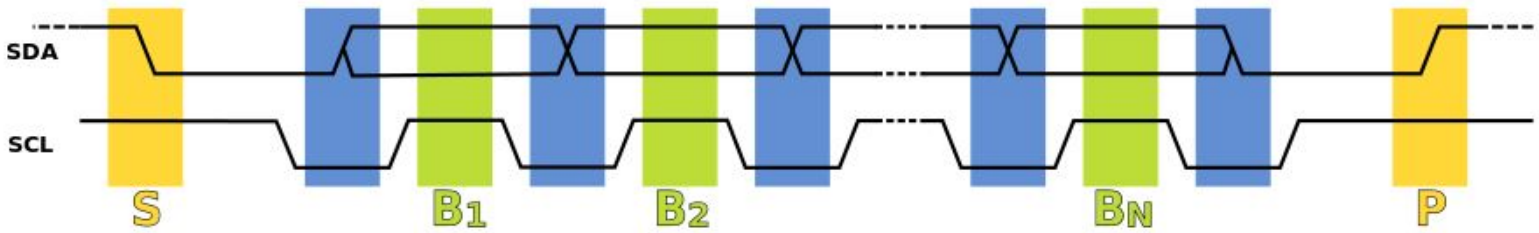
# Connection Diagram:



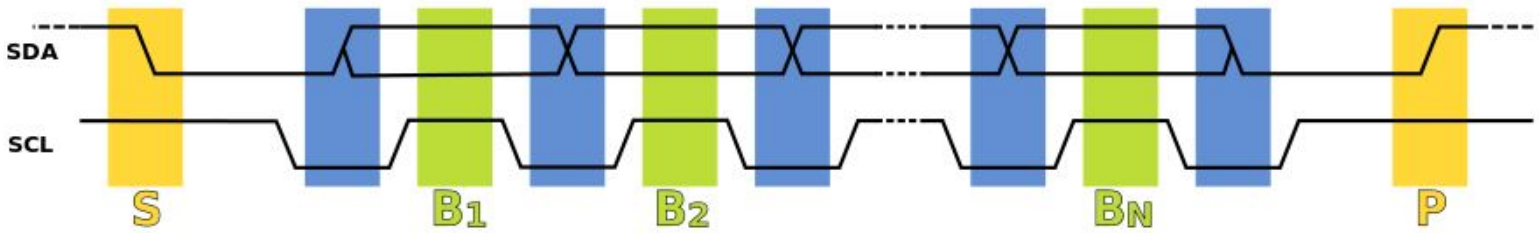

- Pins required for I2C:
  1. SDA: Serial Data Line - To send and receive information.
  2. SCL: Serial Clock Line - To synchronize the communication.

# I2C Communication Steps:

# I2C Communication Steps:

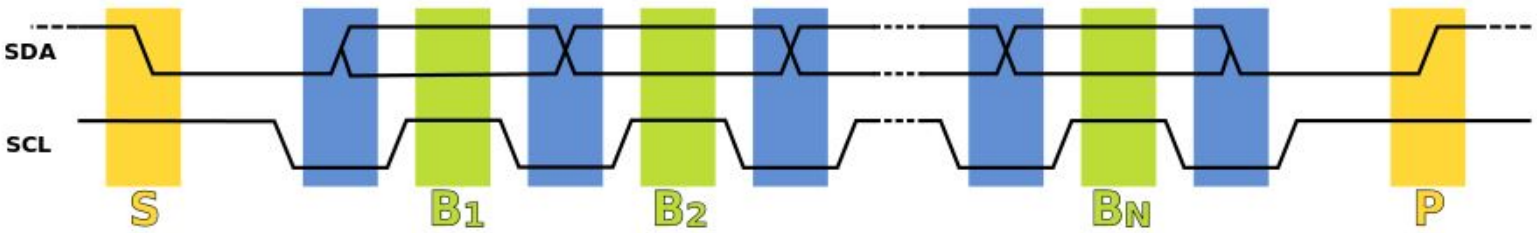


# I2C Communication Steps:



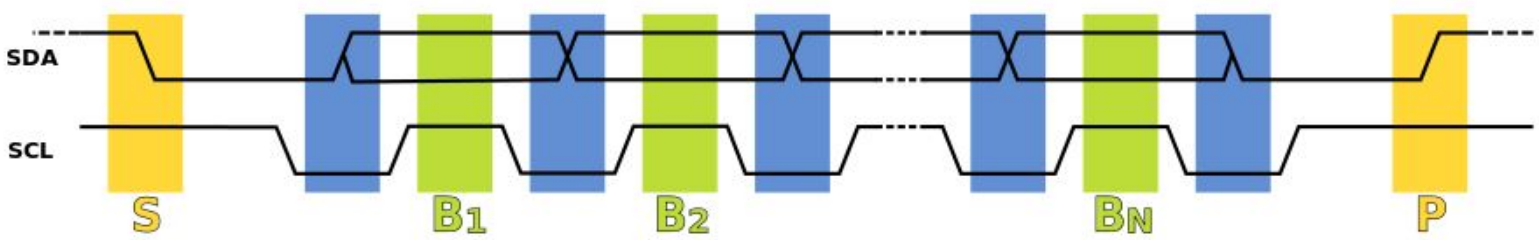

1. Initiate data transfer with a start bit (S) - SDA being pulled low while SCL stays high.

# I2C Communication Steps:



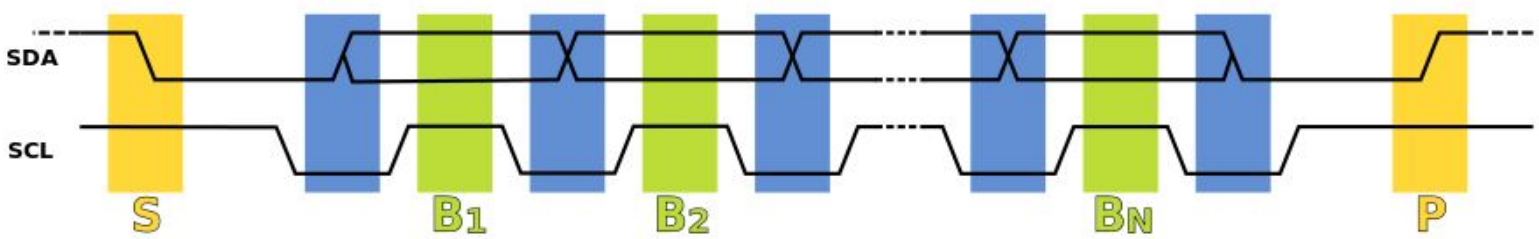

1. Initiate data transfer with a start bit (S) - SDA being pulled low while SCL stays high.
2. Send data - SCL is pulled low, and SDA sets the first data bit level.

# I2C Communication Steps:



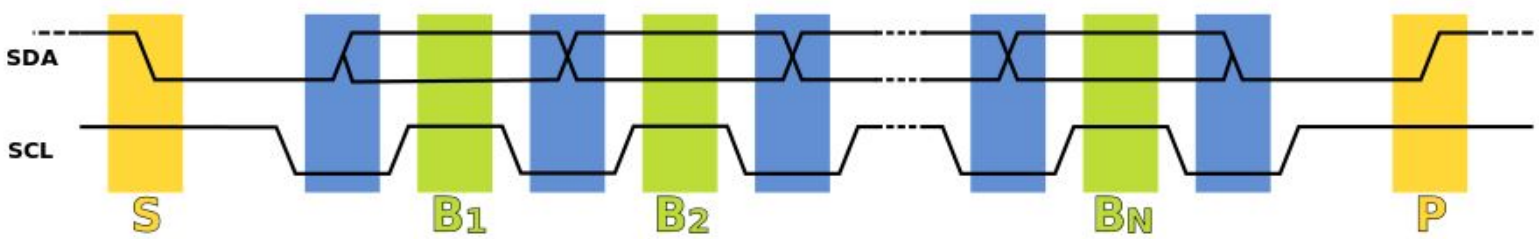

1. Initiate data transfer with a start bit (S) - SDA being pulled low while SCL stays high.
2. Send data - SCL is pulled low, and SDA sets the first data bit level.
3. Receive data - SCL rises for the first bit (B1). For a bit to be valid, SDA must not change between a rising edge of SCL and the subsequent falling edge.

# I2C Communication Steps:



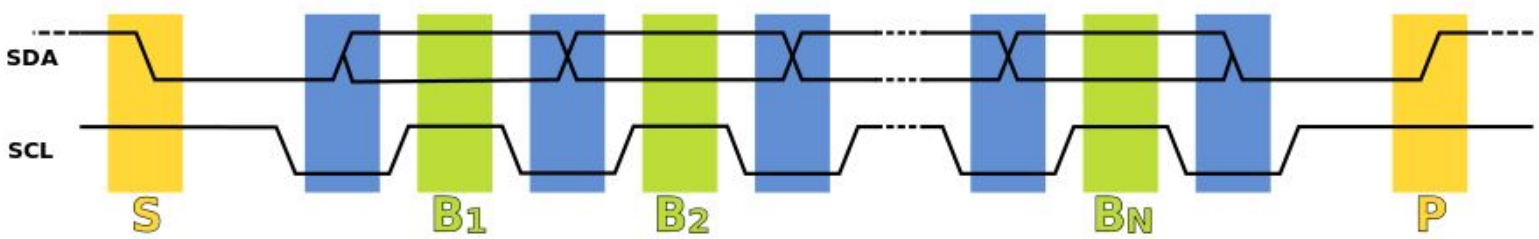

1. Initiate data transfer with a start bit (S) - SDA being pulled low while SCL stays high.
2. Send data - SCL is pulled low, and SDA sets the first data bit level.
3. Receive data - SCL rises for the first bit (B1). For a bit to be valid, SDA must not change between a rising edge of SCL and the subsequent falling edge.
4. Process repeats, SDA transitioning while SCL is low, and the data being read while SCL is high.

# I2C Communication Steps:




1. Initiate data transfer with a start bit (S) - SDA being pulled low while SCL stays high.
2. Send data - SCL is pulled low, and SDA sets the first data bit level.
3. Receive data - SCL rises for the first bit (B1). For a bit to be valid, SDA must not change between a rising edge of SCL and the subsequent falling edge.
4. Process repeats, SDA transitioning while SCL is low, and the data being read while SCL is high.
5. Stop data transfer - SDA is pulled low in preparation for the stop bit.

# I2C Communication Steps:




1. Initiate data transfer with a start bit (S) - SDA being pulled low while SCL stays high.
2. Send data - SCL is pulled low, and SDA sets the first data bit level.
3. Receive data - SCL rises for the first bit (B1). For a bit to be valid, SDA must not change between a rising edge of SCL and the subsequent falling edge.
4. Process repeats, SDA transitioning while SCL is low, and the data being read while SCL is high.
5. Stop data transfer - SDA is pulled low in preparation for the stop bit.
6. A stop bit (P) is signaled when SCL rises, followed by SDA rising.

# I2C Protocol

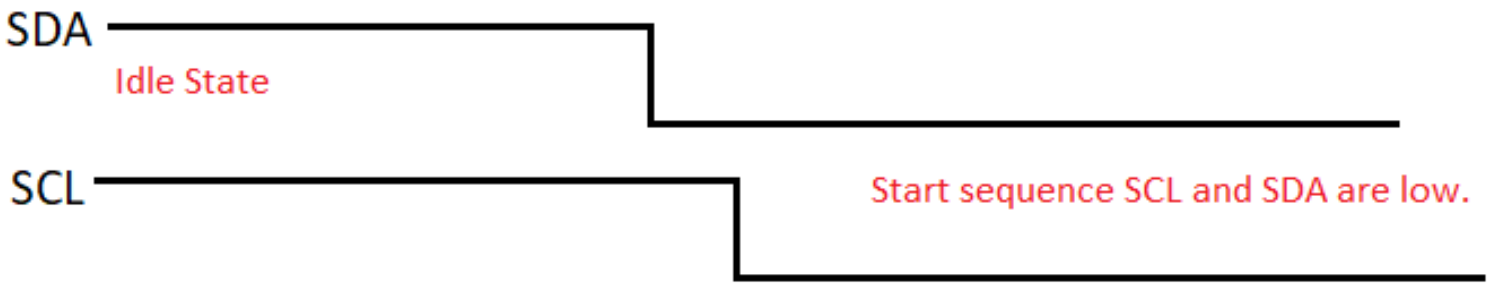
# I2C Protocol

Start + Slave Addressing + Ack + Data transfer + Ack + Stop


# I2C Protocol

Start + Slave Addressing + Ack + Data transfer + Ack + Stop

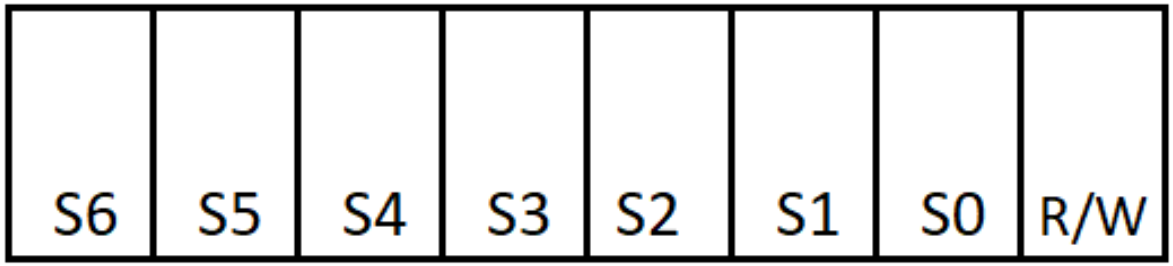

- Start condition marks the start of the protocol.
- SCL line is pulled down by lowering the voltage.




# I2C Protocol

Start + Slave Addressing + Ack + Data transfer + Ack + Stop


- Slaves are selected by sending 7 or 10 bit along data line.




- R/W bit decides the read or write operation.

# I2C Protocol

Start + Slave Addressing + Ack + Data transfer + Ack + Stop

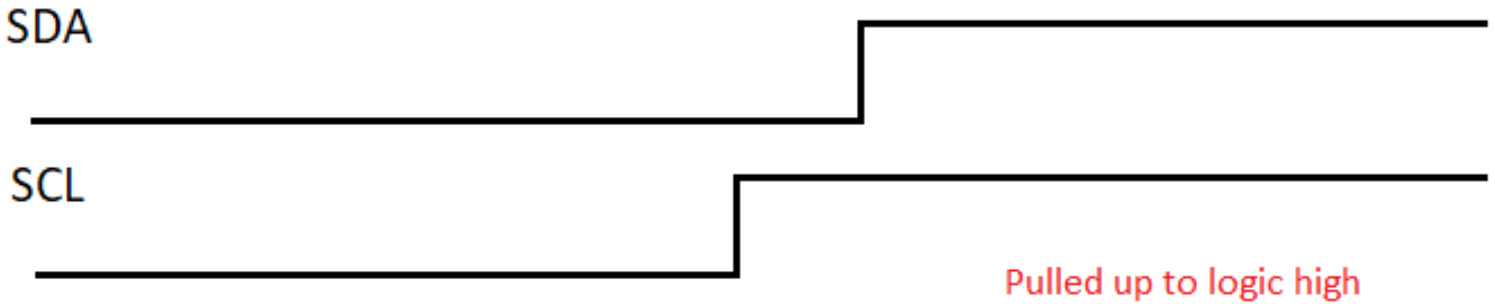

- Data is transferred (Read or Write) b/w master and selected.


- The direction of data transfer is determined by R/W bit.

# I2C Protocol

Start + Slave Addressing + Ack + Data transfer + Ack + Stop


- Stop condition marks the End of the protocol.




- SCL and SDA lines are released.

# I2C Protocol

# I2C Protocol

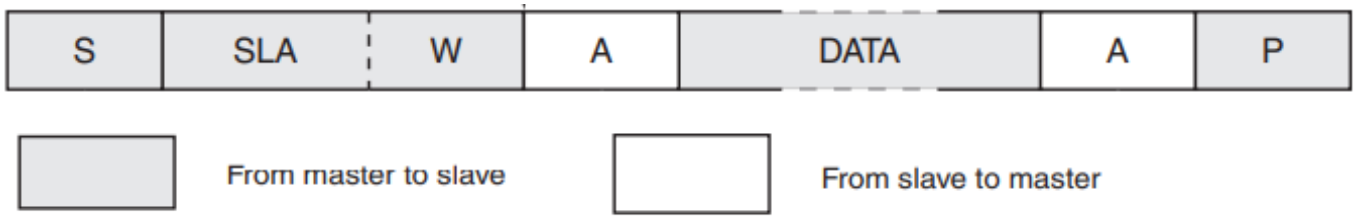



Figure: Master Transmitter

# I2C Protocol

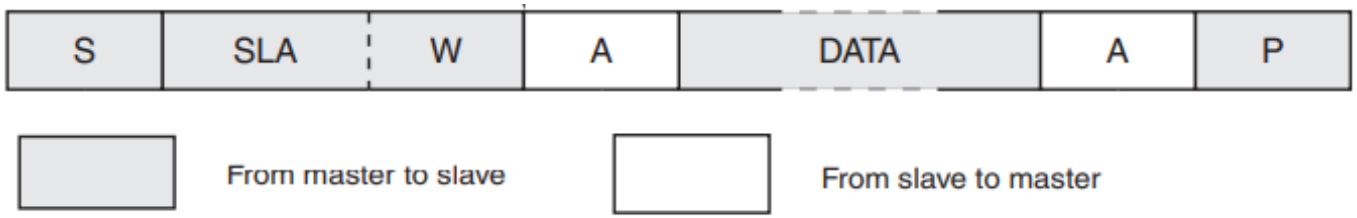



Figure: Master Transmitter

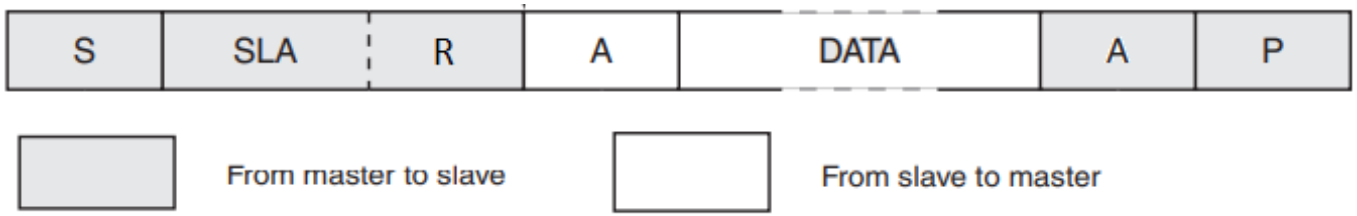



Figure: Master Receiver

# I2C Protocol

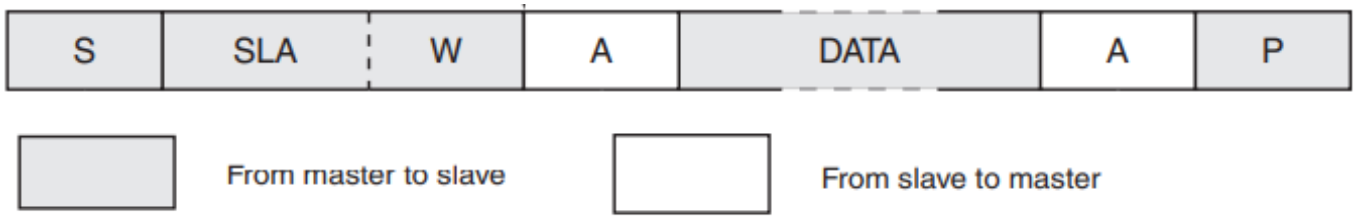



Figure: Master Transmitter

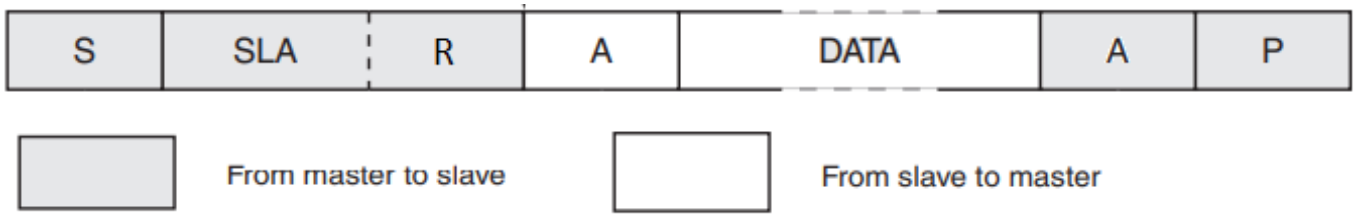



Figure: Master Receiver

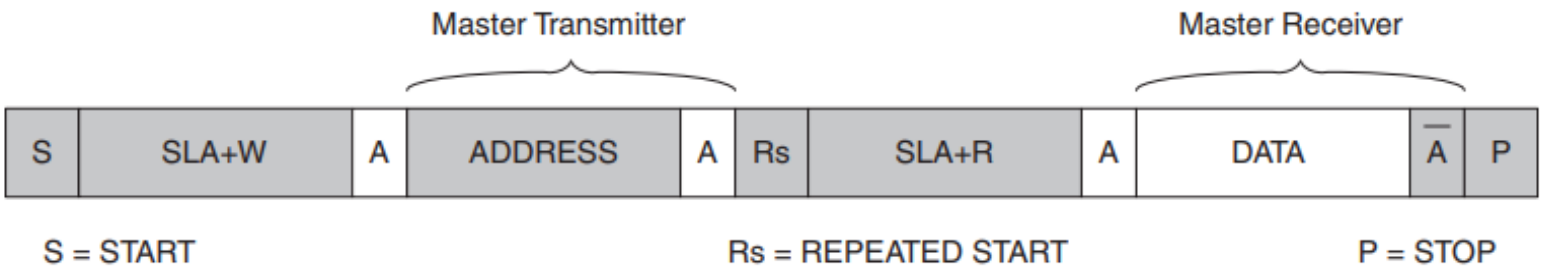
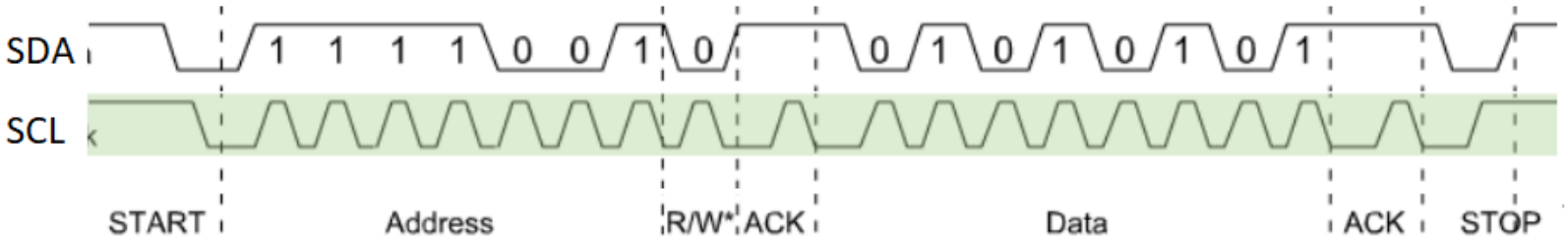



Figure: Master Transmitter and Receiver

# I2C Protocol

# I2C Protocol

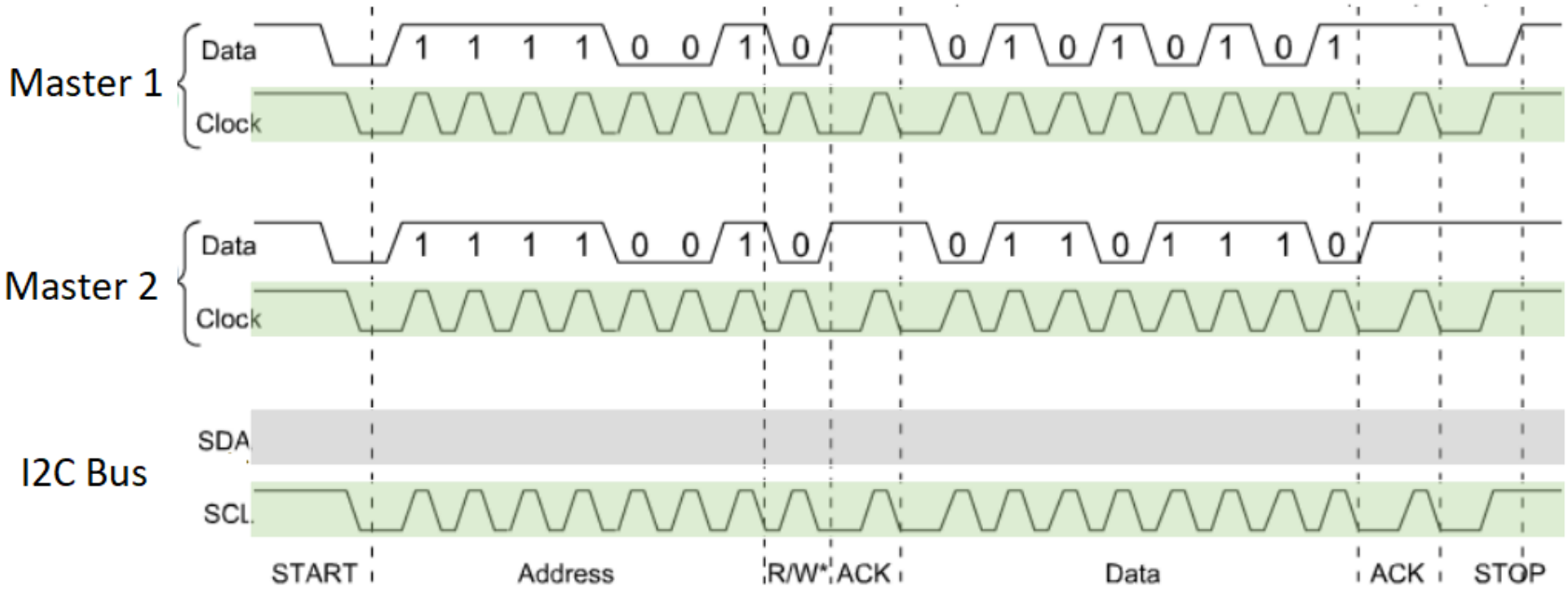



Features of I2C:

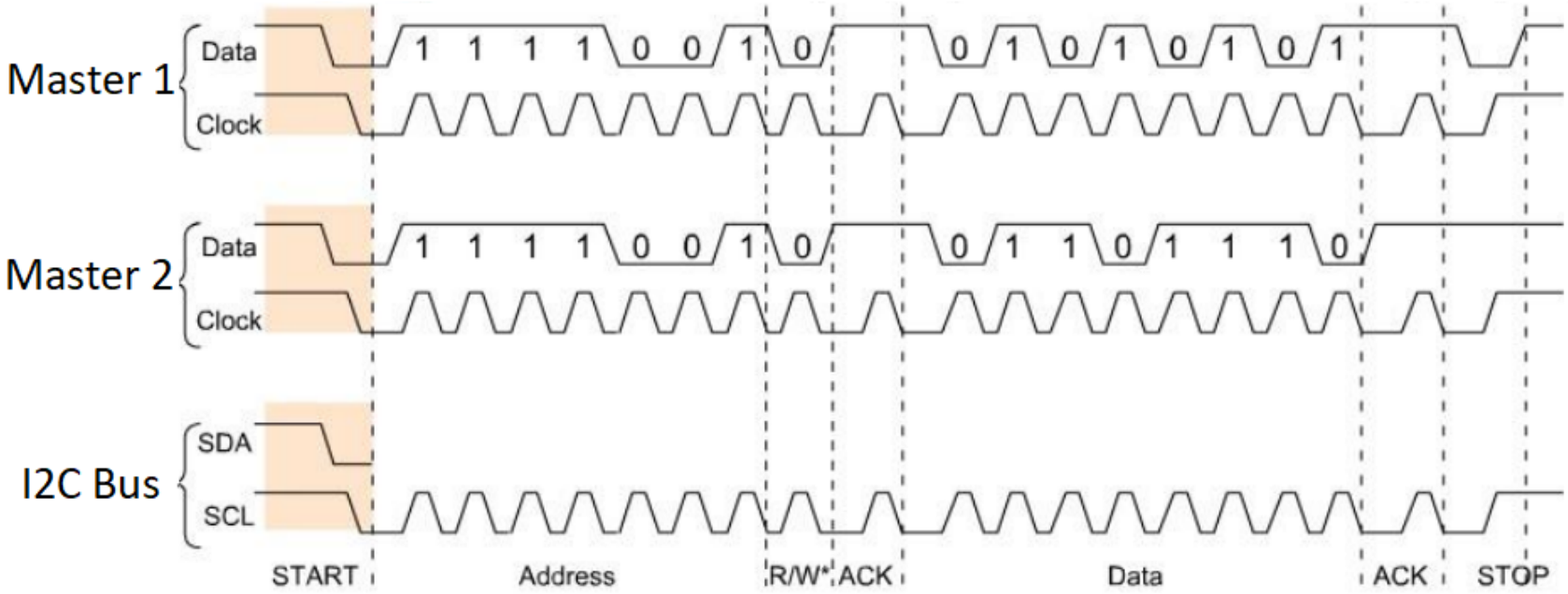
- Bus Arbitration: When multiple devices initiates a communication
- Clock Stretching: When slave wants to take control of the clock

# Bus Arbitration

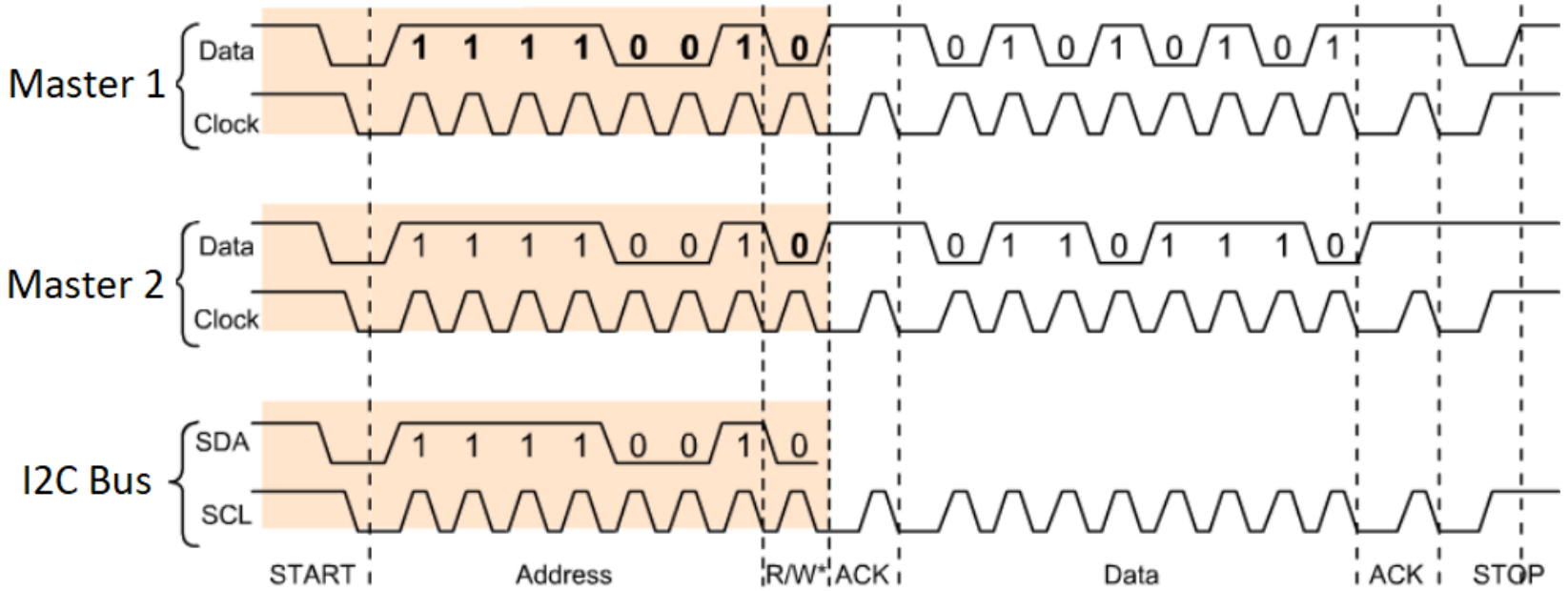
# Bus Arbitration



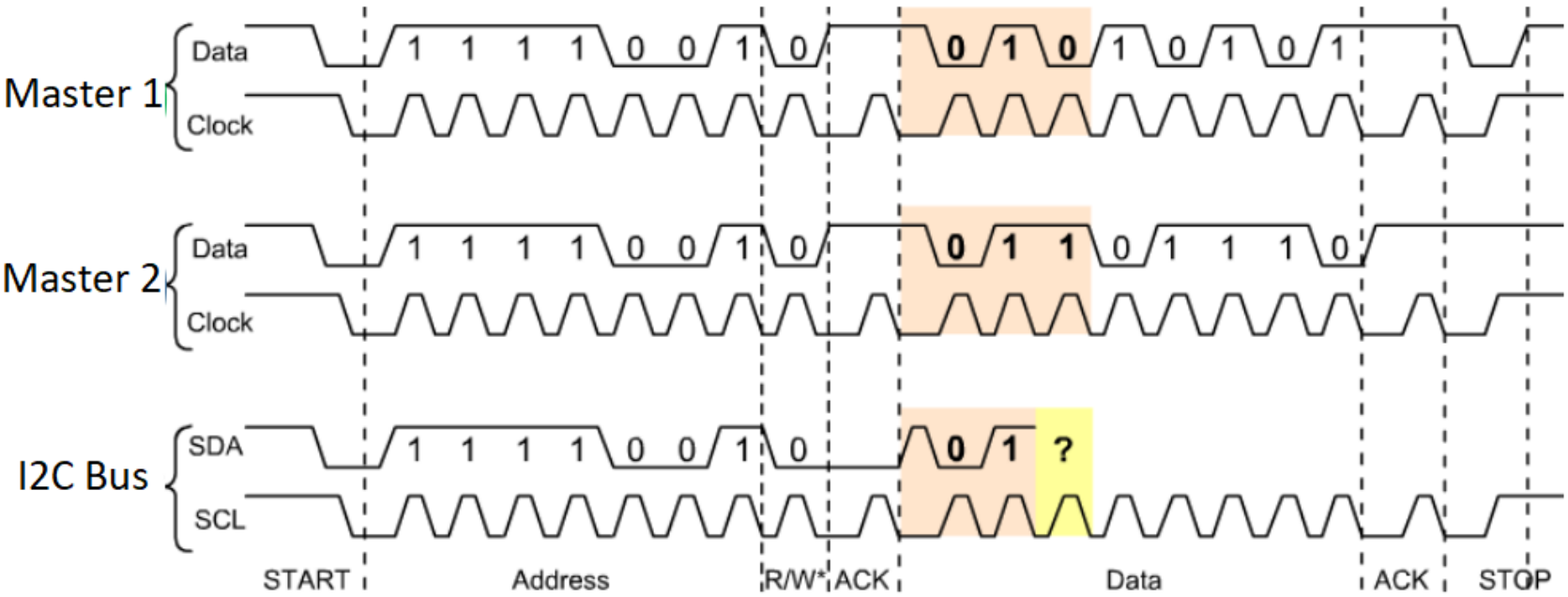
# Bus Arbitration



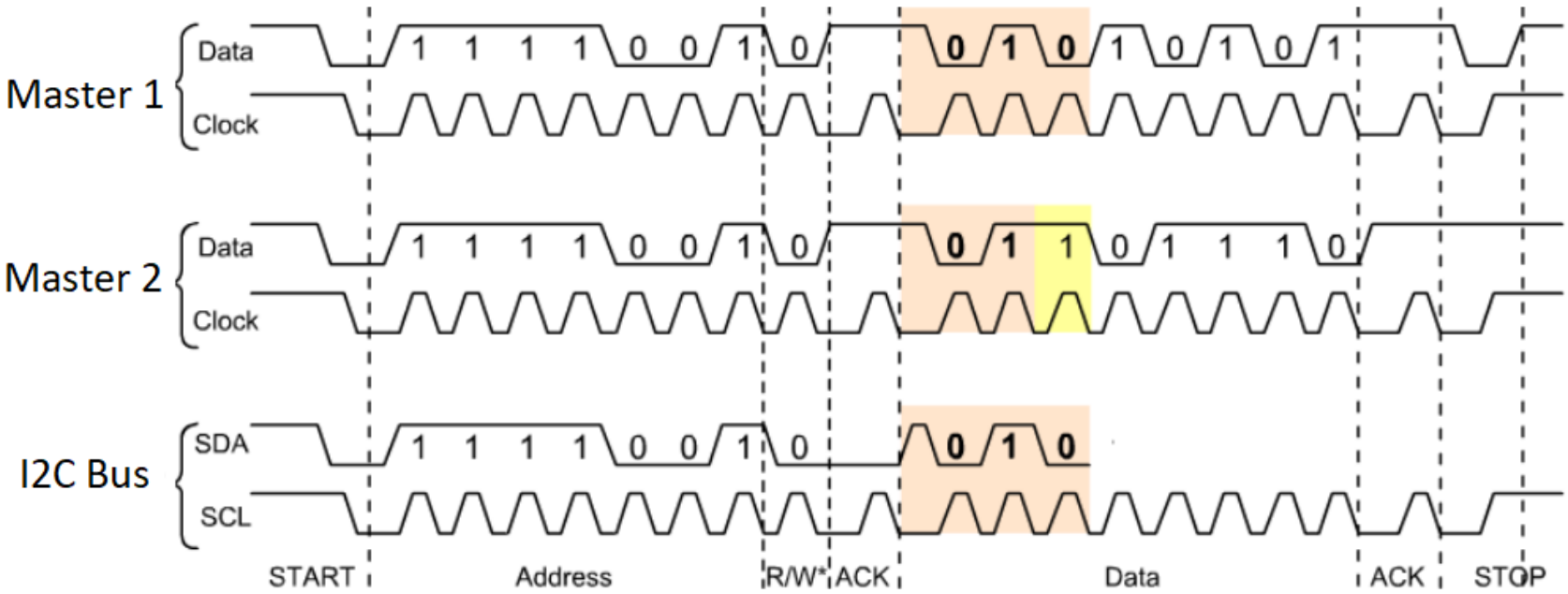
# Bus Arbitration



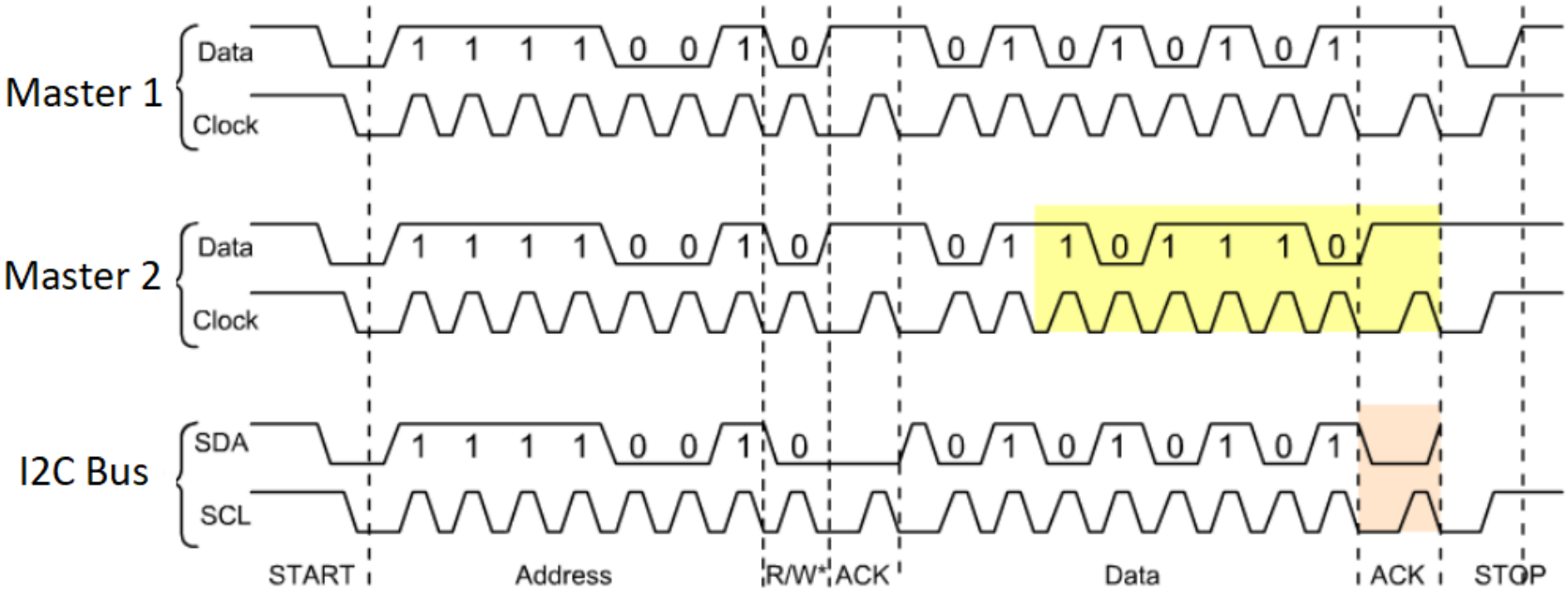
# Bus Arbitration



# Bus Arbitration



# Bus Arbitration



# Bus Arbitration

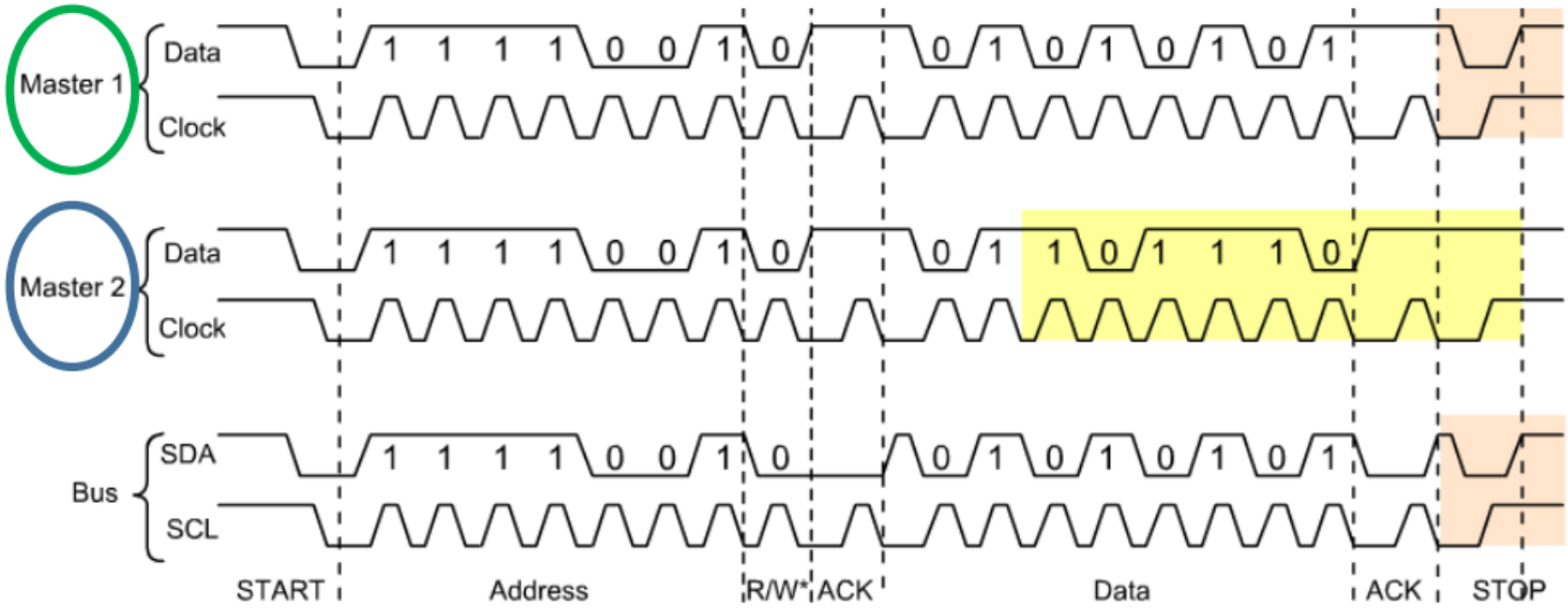
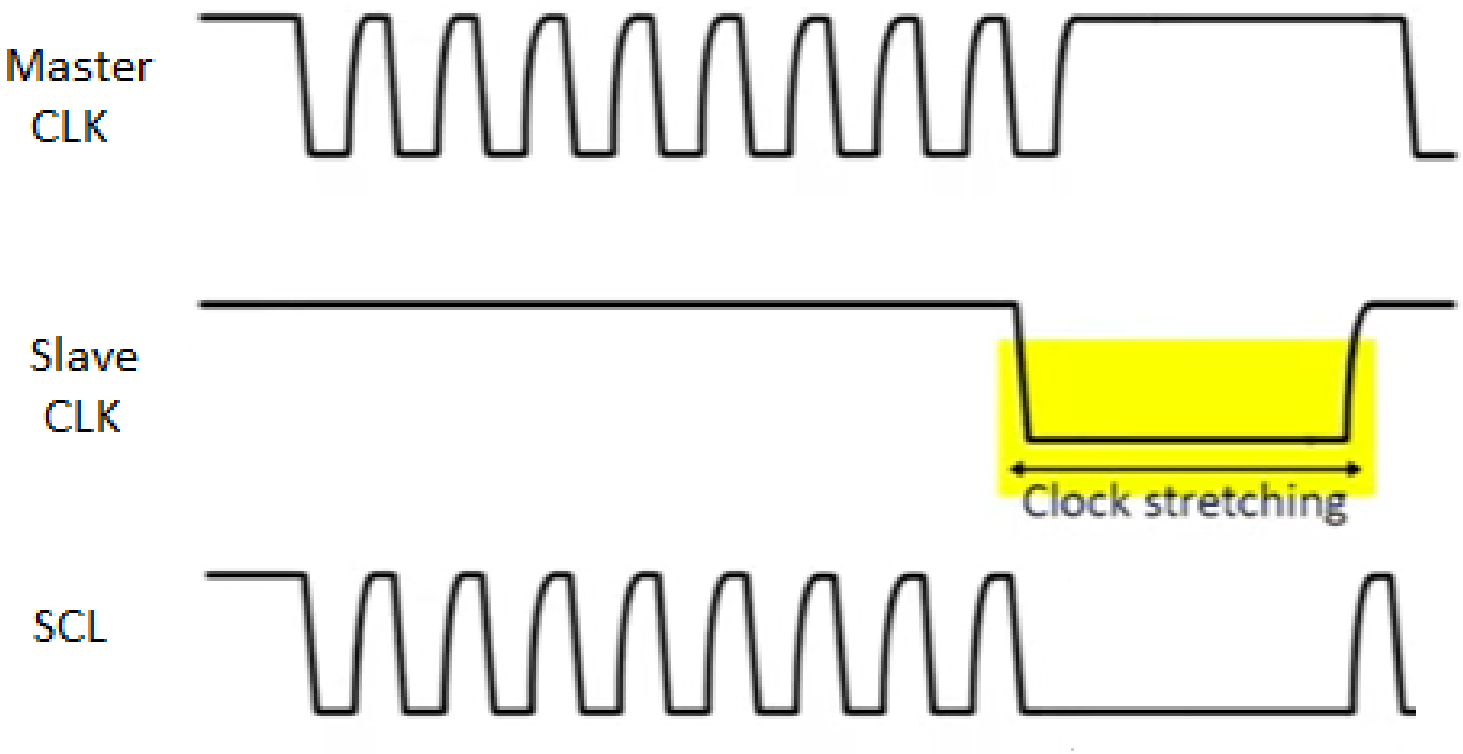



image source: rophoenixmakerevolution

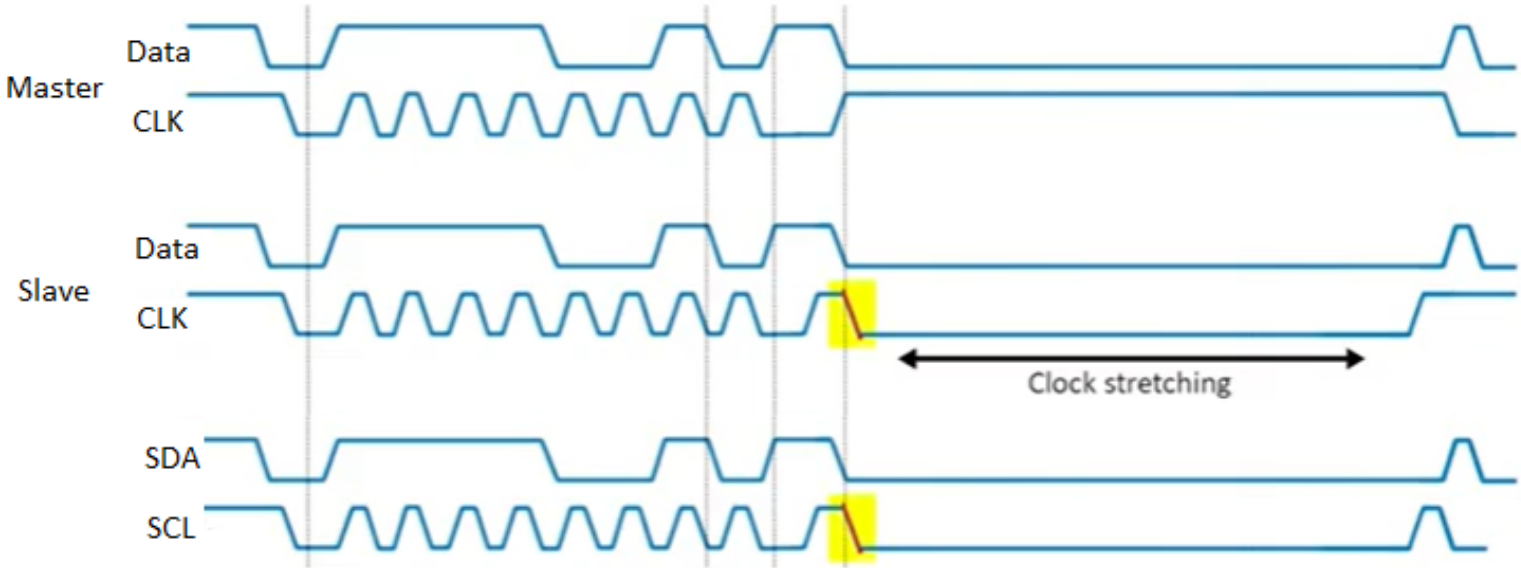
# Clock Stretching

# Clock Stretching



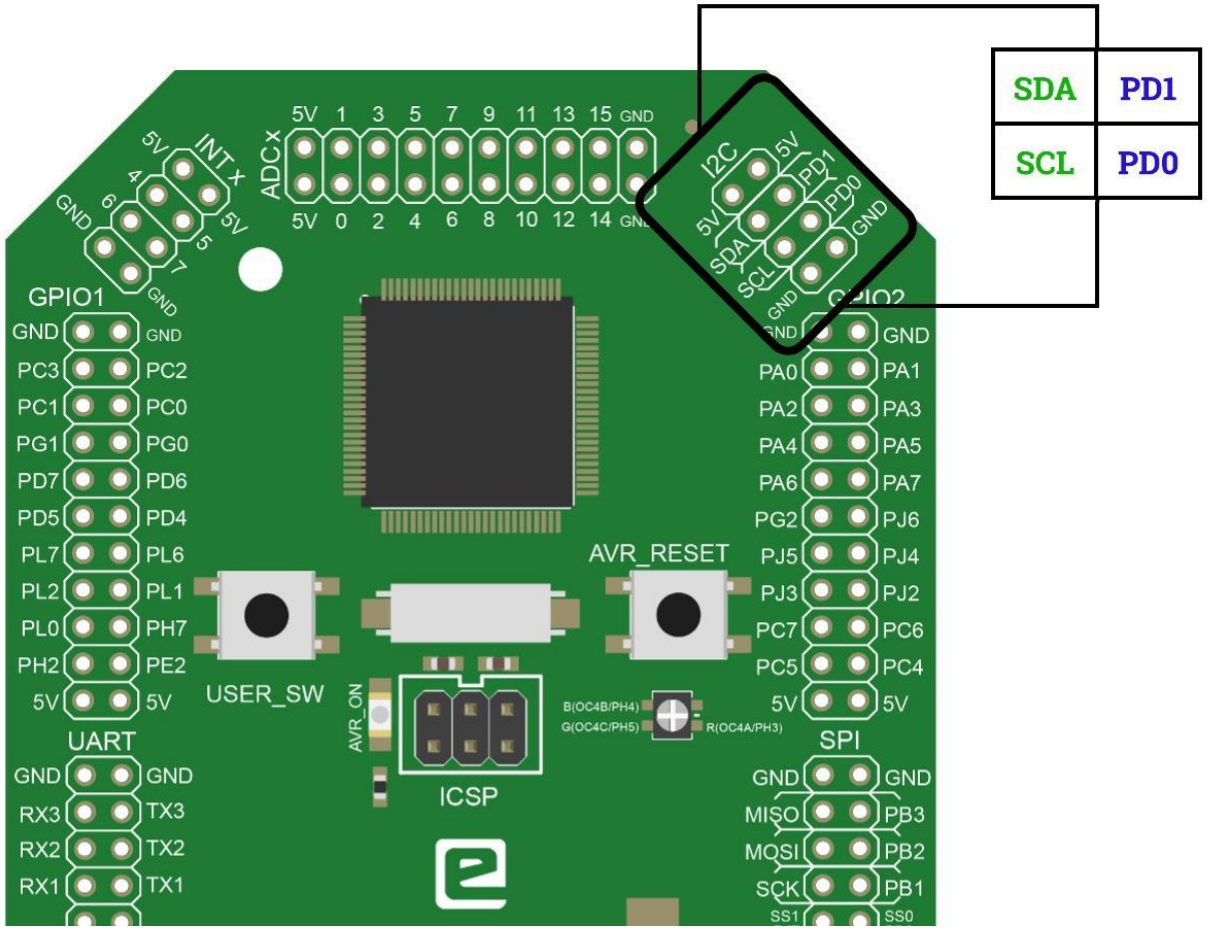
# Clock Stretching

# Clock Stretching



# I2C Header on eYFi-Mega Board

# I2C Header on eYFi-Mega Board



# OLED Display

# OLED Display

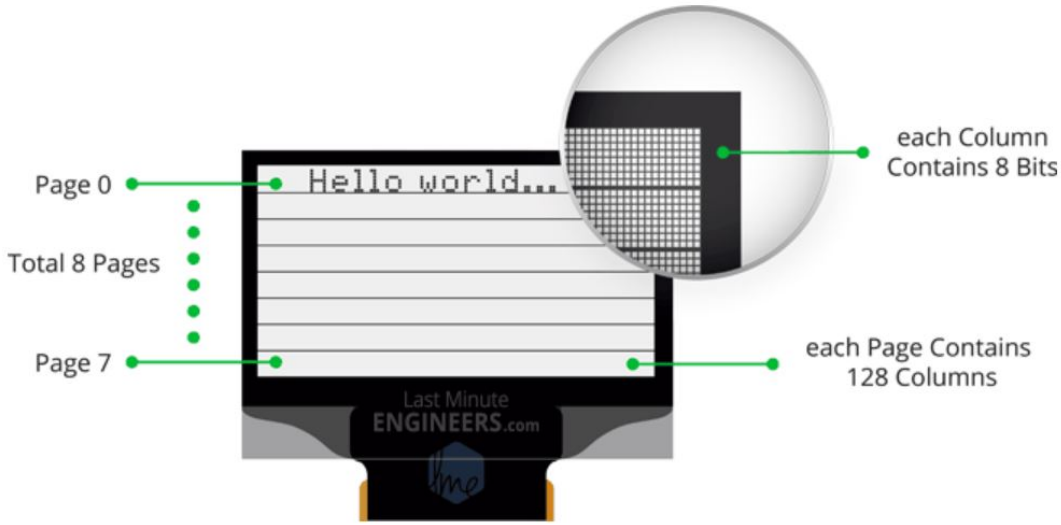
- OLED (Organic Light Emitting Diodes) is a flat light emitting technology.
- Used to display Text, Images and moving pictures.
- Communicates with microcontroller using SPI or I2C.
- Comes in different sizes and colors, for example: 128x64, 128x32, with white OLEDs, Blue OLEDs and Dual Color OLEDs.



# OLED Memory Map

# OLED Memory Map

**SH1106 OLED:**




- Display divided into 8 Pages.
- Each Page - 128 Columns.
- Each column - 8 bit data.

Each bit represents particular OLED pixel on the screen which can be turned ON or OFF programmatically.
Graphic Display Data RAM (GDDRAM) = **1KB**.

# OLED Working

# OLED Working

1. OLED internally consists of a controller

# OLED Working

1. OLED internally consists of a controller
2. Contains Data Register and command register

# OLED Working

1. OLED internally consists of a controller
2. Contains Data Register and command register
3. Useful commands are: clear display, set cursor position, initialise display, set inverting mode

# OLED Working

1. OLED internally consists of a controller
2. Contains Data Register and command register
3. Useful commands are: clear display, set cursor position, initialise display, set inverting mode
4. Master Transmitter mode is used

# OLED Working

1. OLED internally consists of a controller
2. Contains Data Register and command register
3. Useful commands are: clear display, set cursor position, initialise display, set inverting mode
4. Master Transmitter mode is used

# OLED Working

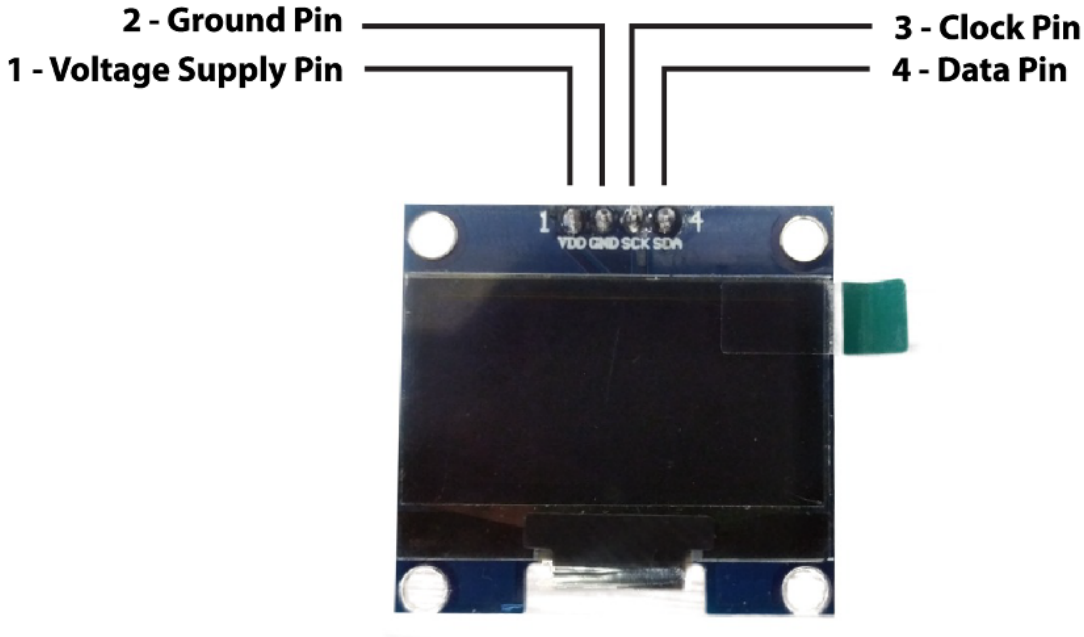
1. OLED internally consists of a controller
2. Contains Data Register and command register
3. Useful commands are: clear display, set cursor position, initialise display, set inverting mode
4. Master Transmitter mode is used

Start + Slave Addressing + Ack + Register Addressing + Ack + Data transfer + Ack + Stop

# OLED Interfacing

# OLED Interfacing

## SH1106 OLED:



# Library functions available for OLED

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();

 // Set the font size. Supports font size from 1 to 8
display.setTextSize(n);
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();

 // Set the font size. Supports font size from 1 to 8
display.setTextSize(n);

 // Set start coordinates for writing text
display.setCursor(x, y);
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();

 // Set the font size. Supports font size from 1 to 8
display.setTextSize(n);

 // Set start coordinates for writing text
display.setCursor(x, y);

 // Set the text color with background
display.setTextColor(SH110X_WHITE); \
display.setTextColor(SH110X_BLACK, SH110X_WHITE);
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();

 // Set the font size. Supports font size from 1 to 8
display.setTextSize(n);

 // Set start coordinates for writing text
display.setCursor(x, y);

 // Set the text color with background
display.setTextColor(SH110X_WHITE); \
display.setTextColor(SH110X_BLACK, SH110X_WHITE);

 // Print a string or number on oled
display.print("String");\
display.print(number);
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();

 // Set the font size. Supports font size from 1 to 8
display.setTextSize(n);

 // Set start coordinates for writing text
display.setCursor(x, y);

 // Set the text color with background
display.setTextColor(SH110X_WHITE); \
display.setTextColor(SH110X_BLACK, SH110X_WHITE);

 // Print a string or number on oled
display.print("String");\
display.print(number);

 // call this method for changes to make effect
display.display();
```

# Library functions available for OLED

```
 // Initialise the display. Address is 0x3C
display.begin(i2c_Address, true);

 // All pixels are off
display.clearDisplay();

 // Set the font size. Supports font size from 1 to 8
display.setTextSize(n);

 // Set start coordinates for writing text
display.setCursor(x, y);

 // Set the text color with background
display.setTextColor(SH110X_WHITE); \
display.setTextColor(SH110X_BLACK, SH110X_WHITE);

 // Print a string or number on oled
display.print("String");\
display.print(number);

 // call this method for changes to make effect
display.display();
```

# Thank You!

Post your queries on: helpdesk@e-yantra.org