I2C - Inter Integrated Circuits bus
by Philips Semiconductors

Dr. Sarwan Singh

Deputy Director

NIELIT Chandigarh

Agenda

- Introduction

- History

- Working

- Connecting multiple devices, LCD using I2C

sarwan@NIELIT

I2C bus

- **I²C** (**Inter-Integrated Circuit**), pronounced *I-squared-C*, is a synchronous, multi-master, multi-slave, packet switched, single-ended, serial computer bus invented in 1982 by Philips Semiconductor (now NXP Semiconductors). It is widely used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication..

- Pronounced "eye-squared-see" or "eye-two-see"

- Two wire serial bus specification

- Invented by Philips in the early 1980s

  - The division is now NXP

 - Was a patented protocol, but patent has now expired

I2C uses

- Originally used by Philips inside television sets
- Now a very common peripheral bus standard
- Intended for use in embedded systems
  - Philips, National, Xicor, Siemens, ... all use
- Also used in PCs
  - RTC, Temperature sensors, Variant is the SMBus (system management bus)
- Since October 10, 2006, no licensing fees are required to implement the I²C protocol. However, fees are required to obtain I²C slave addresses allocated by NXP

- The easy implementations comes with the fact that only two wires are required for communication between up to almost 128 (112) devices when using 7 bits addressing and up to almost 1024 (1008) devices when using 10 bits addressing.


Source:howtomechatronics.com

sarwan@NIELIT
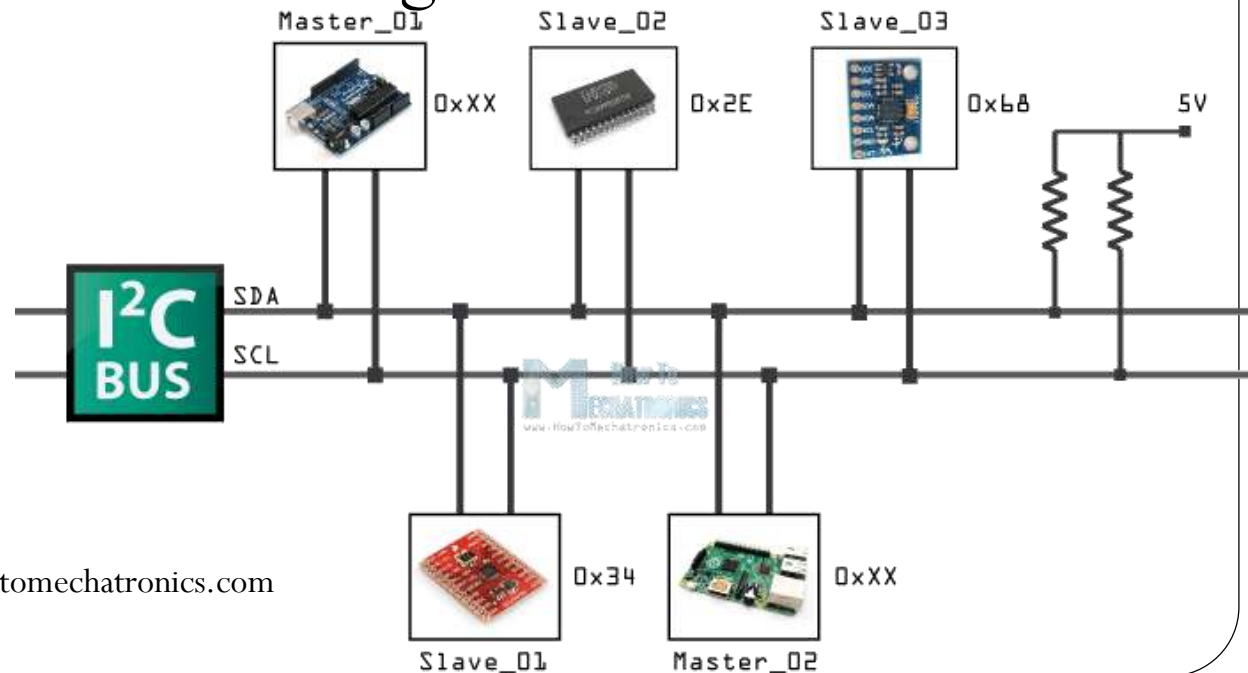
- The two wires, or lines are called Serial Clock (or SCL) and Serial Data (or SDA). The SCL line is the clock signal which synchronize the data transfer between the devices on the I2C bus and it's generated by the master device. The other line is the SDA line which carries the data.

- The two lines are "open-drain" which means that pull up resistors needs to be attached to them so that the lines are high because the devices on the I2C bus are active low. Commonly used values for the resistors are from 2K for higher speeds at about 400 kbps, to 10K for lower speed at about 100 kbps.

sarwan@NIELIT
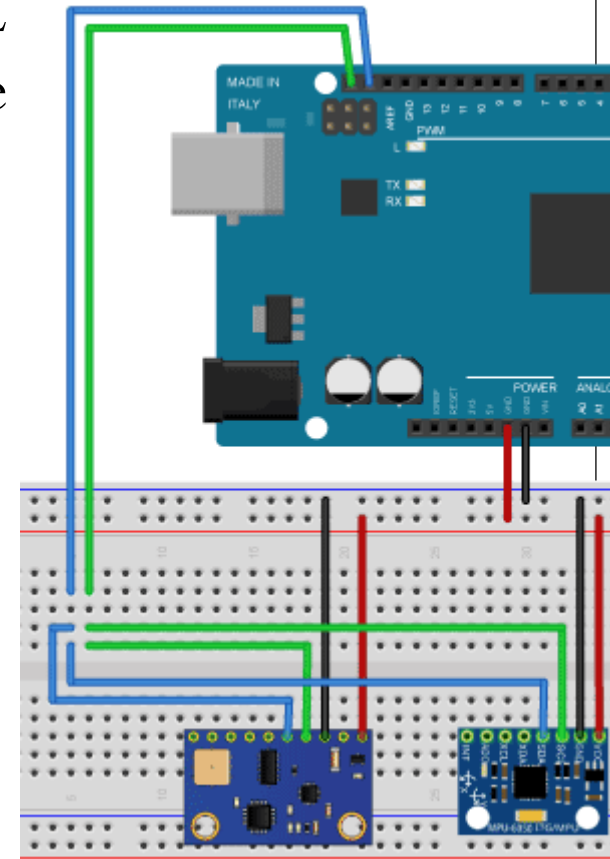
LCD connection using I2C

sarwan@NIELIT
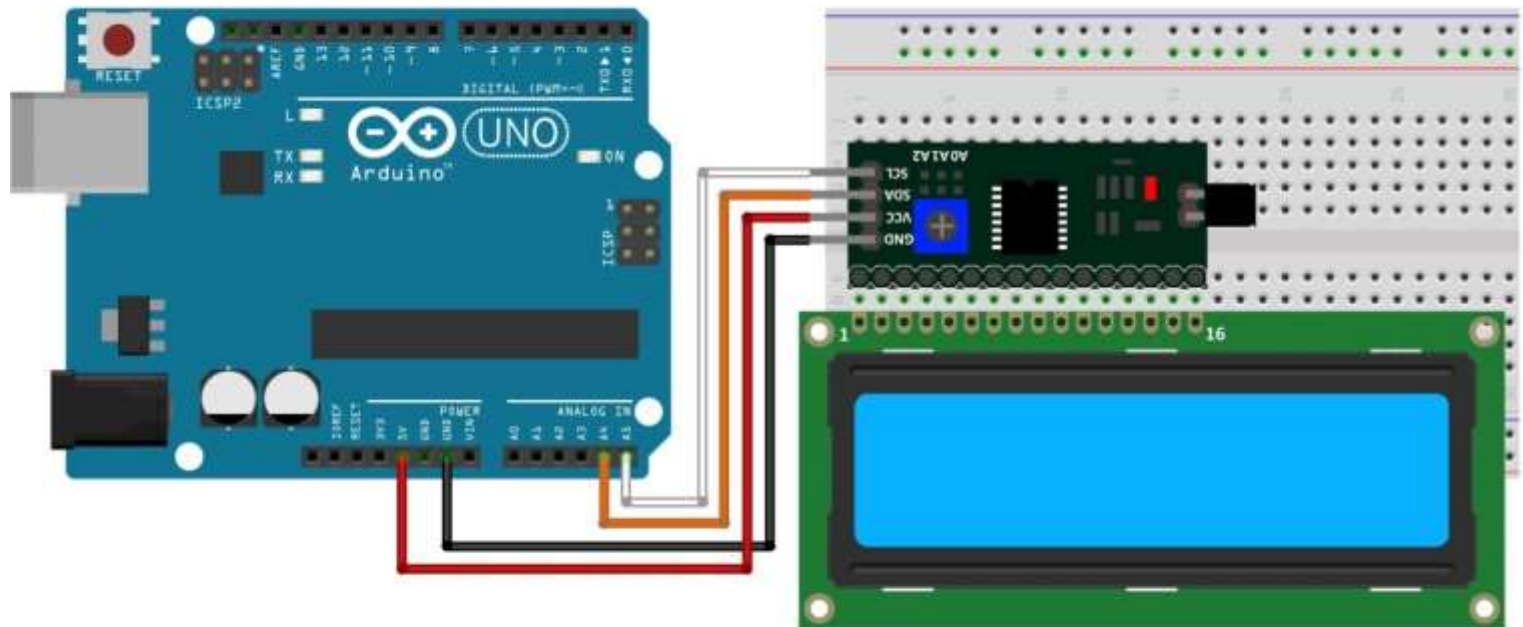
Connecting multiple devices using I2C
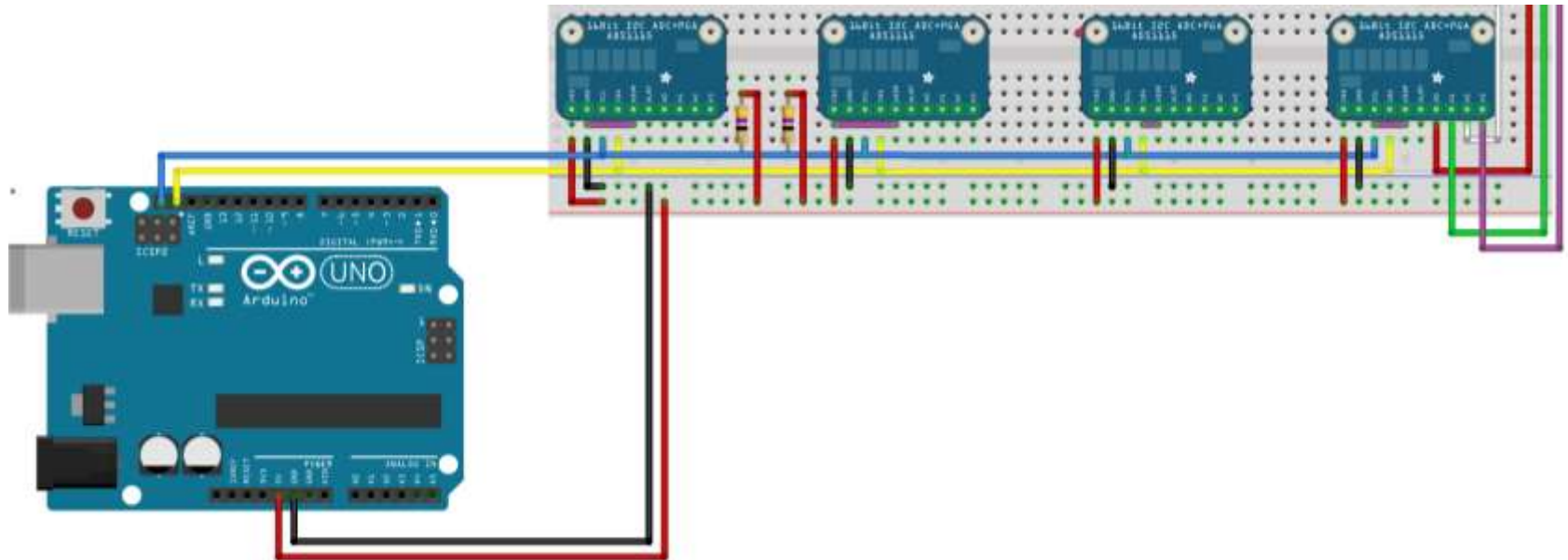
TWI / I²C (I-two-C)

- In TWI the serial data transmission is done in asynchronous mode.

- This protocol uses only two wires for communicating between two or more ICs.

- The two bidirectional open drain lines named SDA (Serial Data) and SCL (Serial Clock) with pull up resistors are used for data transfer between devices.

- One of the two devices, which controls the whole process, is known as Master
- the other which responds to the queries of master is known as Slave device.
- The ACK (acknowledgement) signal is sent/received from both the sides after every transfer and hence reduces the error.
- SCL is the clock line bus used for synchronization and is controlled by the master.
- SDA is known as the data transfer bus.
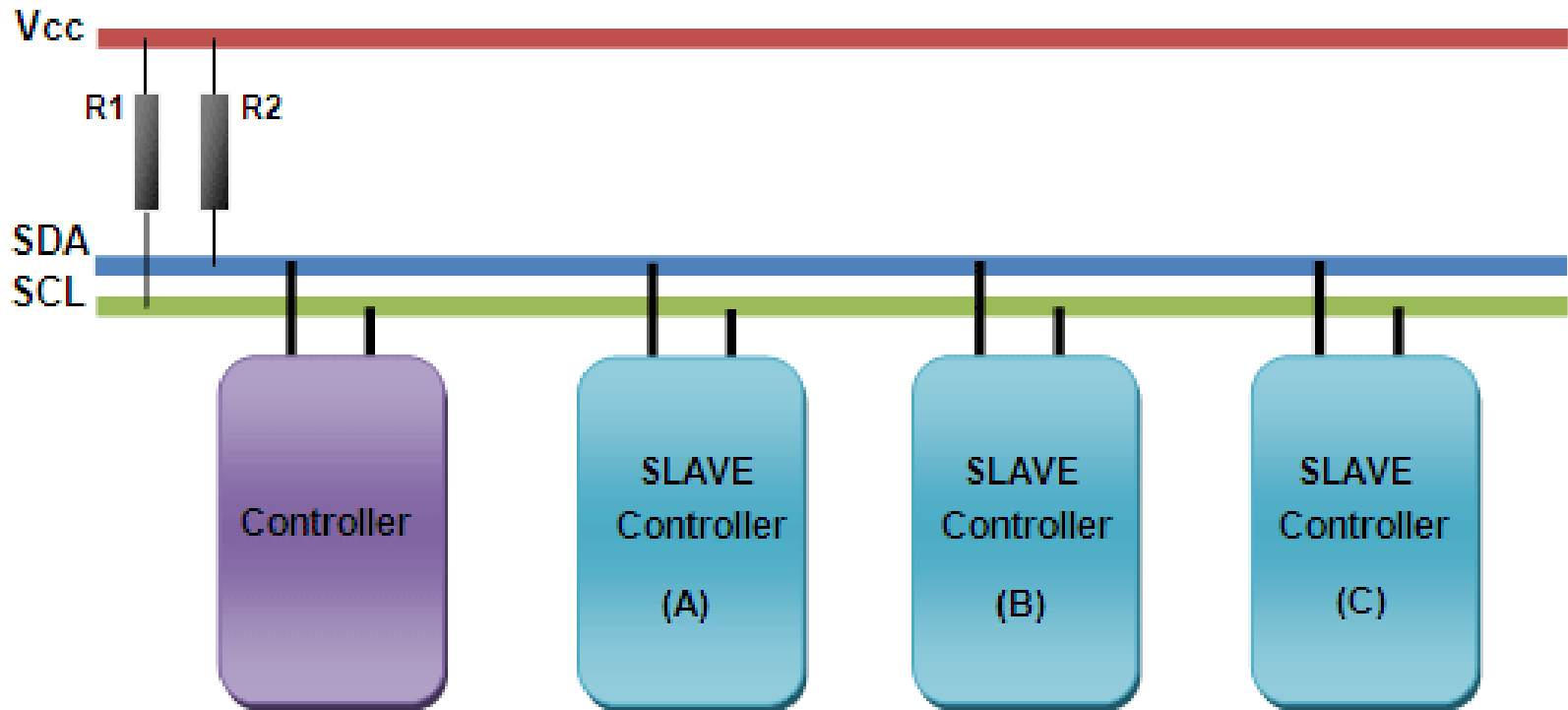
- I^2C/TWI is a half duplex serial transmission and hence the data flow can be in a direction at a time.

- The data transfer rate depends on crystal frequency of slave controller. The rate of data transfer refers to clock frequency on SCL bus which must be 1/16th of slave frequency.

Basic Characteristics

- two-wired bus : originally to interact within small num. of devices (radio/TV tuning, …)

- speeds:
  - 100 kbps (standard mode)
  - 400 kbps (fast mode)
  - 3.4 Mbps (high-speed mode)

- data transfers: serial, 8-bit oriented, bi-directional

- master/slave relationships with multi-master option (arbitration)

- master can operate as transmitter or receiver

- addressing: 7bit or 10bit unique addresses

- device count limit: max. capacitance 400 pF

RTC – Real Time Clock

sarwan@NIELIT

Initializing RTC

```
#include <Wire.h>
#include "RTClib.h"

RTC_DS1307 rtc;

char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};

void setup () {
   Serial.begin(9600);
 if (! rtc.begin()) {
   Serial.println("Couldn't find RTC");
   while (1);
 }
}
```

sarwan@NIELIT

```
void loop () {
    DateTime now = rtc.now();
    Serial.print(now.year(), DEC);
    Serial.print('/');
    Serial.print(now.month(), DEC);
    Serial.print('/');
    Serial.print(now.day(), DEC);
    Serial.print(" (");
    Serial.print(daysOfTheWeek[now.dayOfTheWeek()]);
    Serial.print(") ");
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    Serial.print(':');
    Serial.print(now.second(), DEC);
    Serial.println(); delay(3000);
}
```

sarwan@NIELIT

EEPROM

sarwan@NIELIT

Writing 1 at 0th address location

```
#include <EEPROM.h>

void setup() {
        // write  ( address, value)
 EEPROM.write(0, 1);
}


void loop() {  }
```

Read status from EEPROM on startup

```
#include <EEPROM.h>
int address = 0;
byte value;
void setup() {
  Serial.begin(9600);
  pinMode(13,OUTPUT);  pinMode(2,INPUT);
  value = EEPROM.read(0);
  Serial.print("value in EEPROM  : ");
  Serial.println (value) ;
  if (value ==1)// switch on LED else off
    digitalWrite(13,HIGH);
  else
    digitalWrite(13,LOW);
}
```

sarwan@NIELIT

Write status of button in EEPROM

```
void loop() {
    if (digitalRead(2) ==HIGH) {
        if (value==1) value=0;
        else value=1;
        while (digitalRead(2) ==HIGH) ;
        EEPROM.write(0, value);
        Serial.print(" value Updated in EEPROM  : ");
        Serial.println (value) ;
    }
    if (value ==1)// switch on LED else off
        digitalWrite(13,HIGH);
    else
        digitalWrite(13,LOW);
}
```

sarwan@NIELIT

Happy Coding

Journey begins from here…….

sarwan@NIELIT