Agenda Clase III

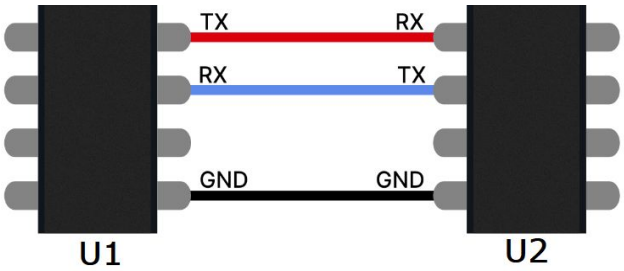
09:30 hs - Enlace de conexión	Consultas clases anteriores y ejercitación.
10:00 hs - Comunicación i2c.	Comunicación i2c. Display y LCD-i2c. Convertidor Lógico de Tensiones (LLC).
10:45 hs - Nuevos conceptos	Repaso y nuevos conceptos. Salidas digitales → PWM → **TIMERS** Entradas digitales → **INTERRUPCIONES, ANTI-REBOTE** Funciones → **ARCHIVOS** → **LIBRERÍAS**
11:30 hs - Break	Receso 15 minutos.
11:45 hs - ¿Cómo seguir?	¿Cómo agregar nuevos módulos a mi proyecto? Ejemplo sensor DHT11. Guía de 10 pasos.
12:30 hs - Cierre de clases	Próximos pasos para finalizar el taller Espacio de Feedback Novedades y cierre.

Clase III - Módulo A

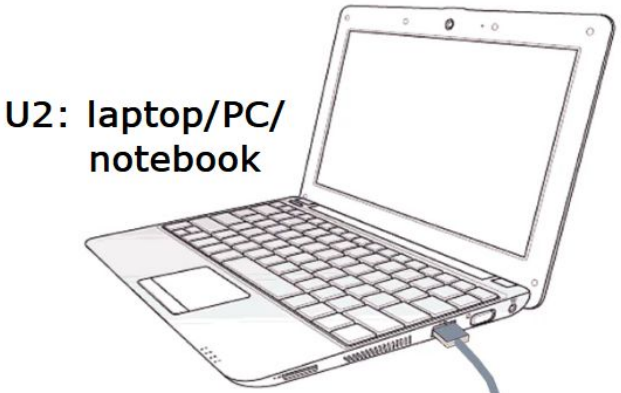
Comunicación i2c. Nuevos conceptos

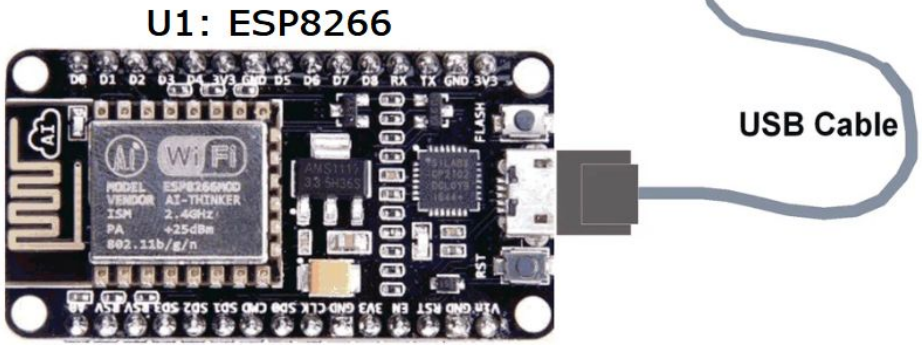
UART

- UNIVERSAL
- ASÍNCRONA
- RECEPCIÓN
- TRANSMISIÓN

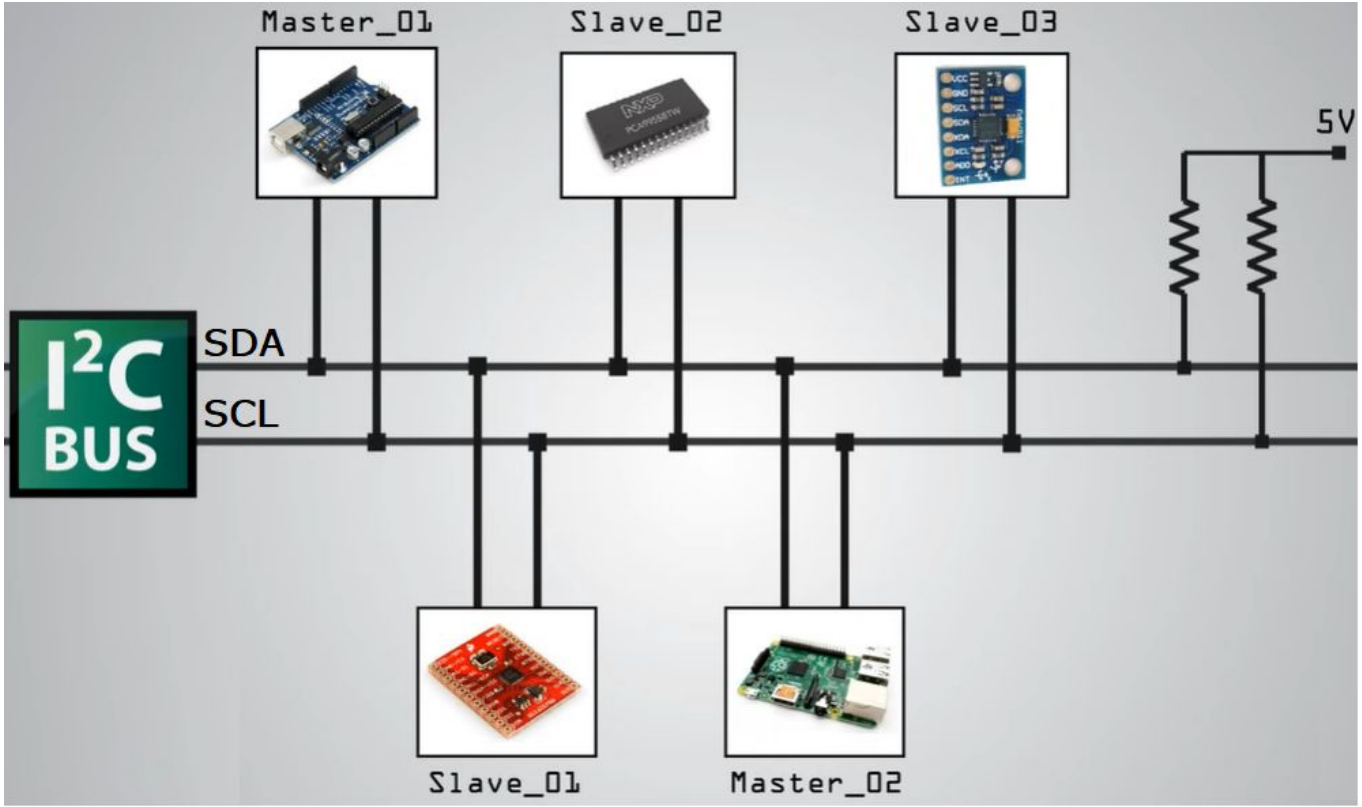


REPASO

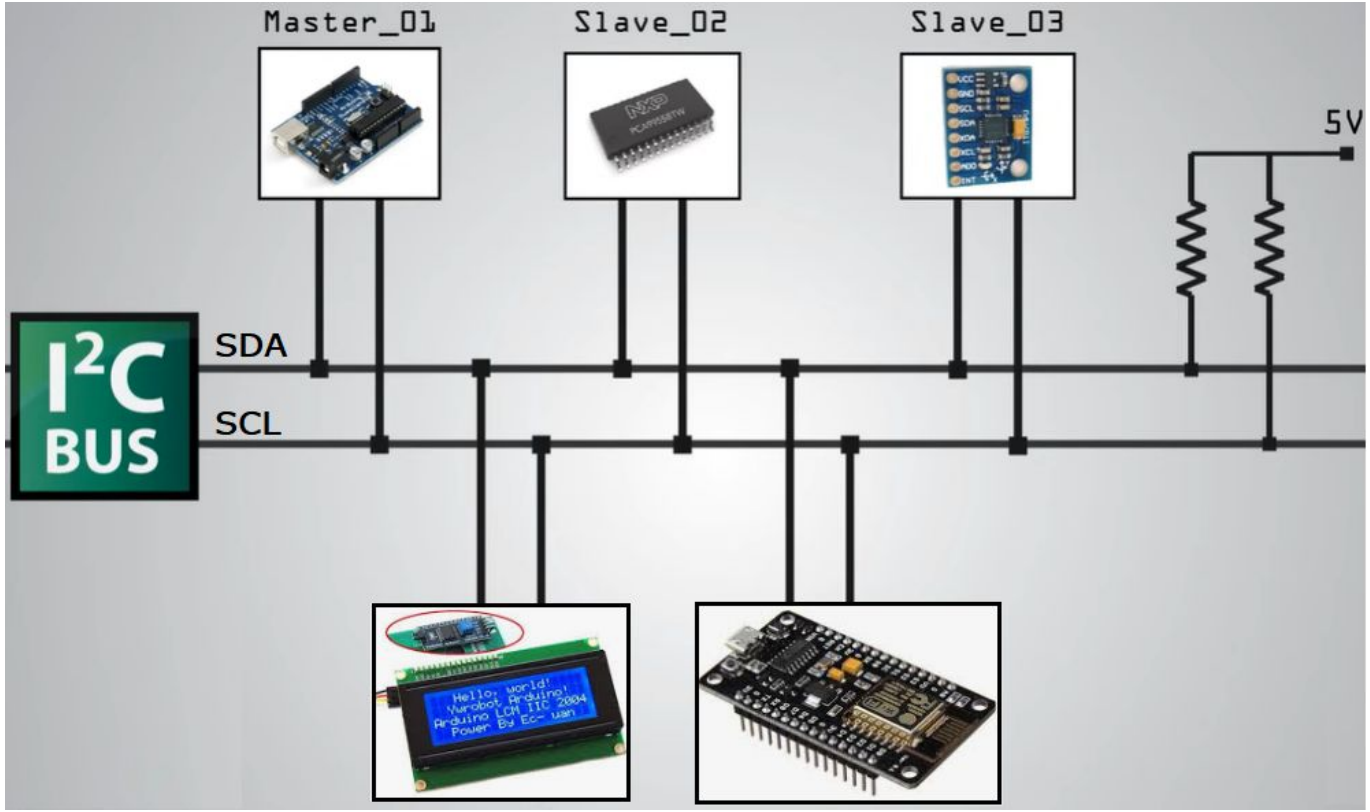




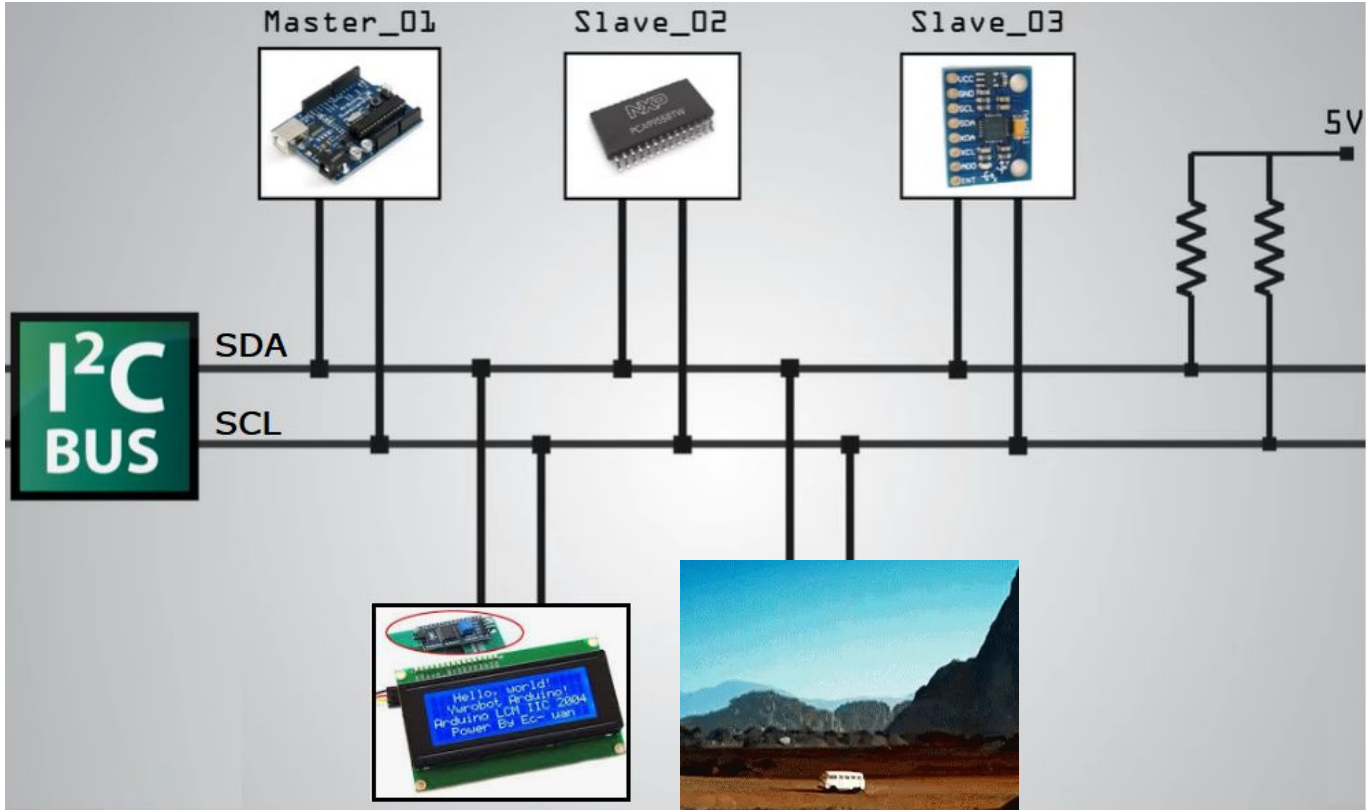
I2C


Master_01
Slave_02
Slave_03
5V
I²C BUS
SDA
SCL
Slave_01
Master_02

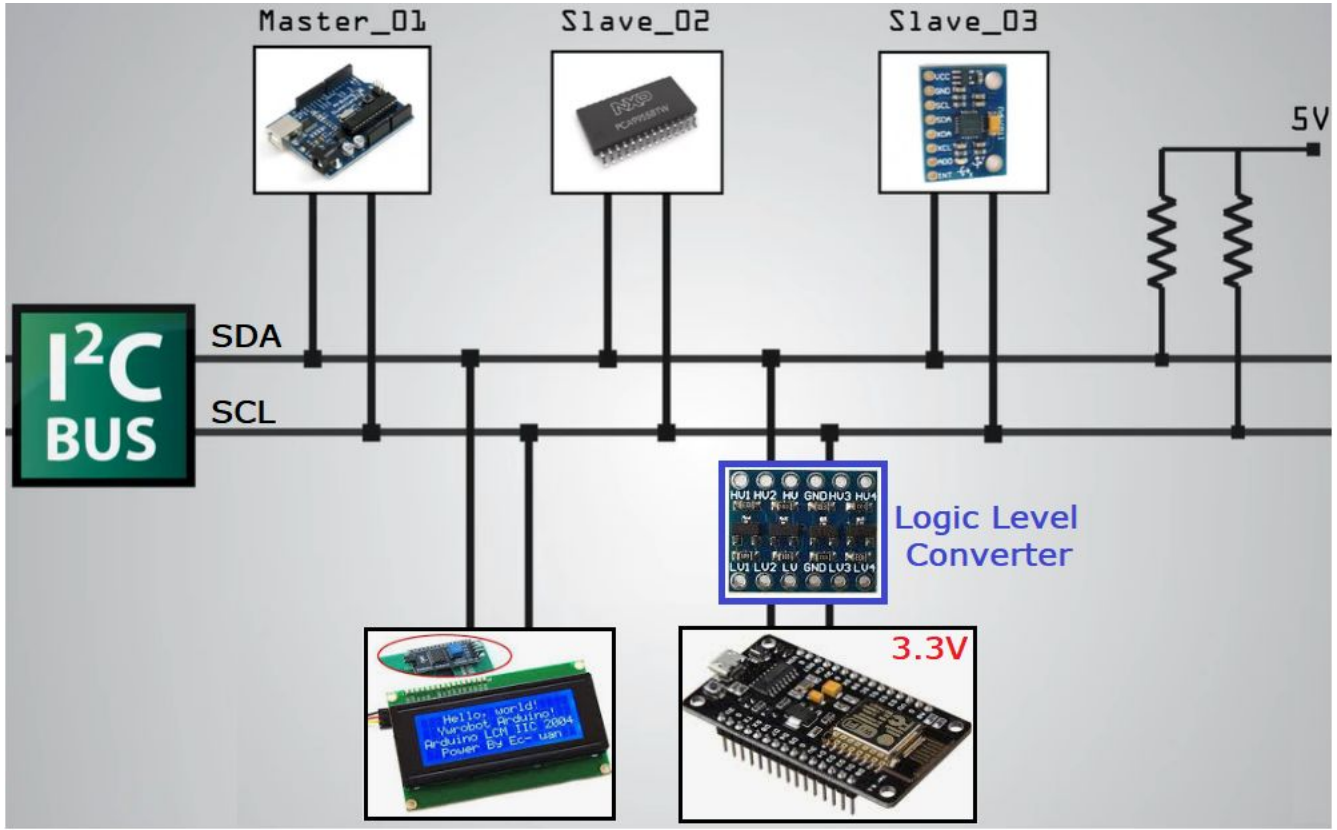
I2C


Master_01
Slave_02
Slave_03
5V
I²C BUS
SDA
SCL
Hello, world!
Wwrobot Arduino!
Arduino LCM IIC 2004
Power By Ec-yan

I2C


Master_01
Slave_02
Slave_03
5V
I²C BUS
SDA
SCL
Hello, world!
Ywrobot Arduino
Arduino LCM IIC 2004
Power By Ec-yan

I2C


Master_01
Slave_02
Slave_03
5V
I²C BUS
SDA
SCL
Logic Level Converter
3.3V

Ejemplo Display LCD-i2c 16x02

```
#include <LiquidCrystal_I2C.h>
#define I2C_ADDR    0x27  //Dirección I2C del dispositivo
#define LCD_COLUMNS 16    //Número de Columnas Display
#define LCD_ROWS    2     //Número de Filas Display
LiquidCrystal_I2C miDisplay(I2C_ADDR, LCD_COLUMNS, LCD_ROWS);
void setup() {
  delay(400);
  miDisplay.begin(LCD_COLUMNS, LCD_ROWS); //Configura Display
  miDisplay.init();       //Inicia Display
  miDisplay.clear();      //Limpia Pantalla
  miDisplay.backlight();  // Enciende la luz de fondo (backlight)
  miDisplay.setCursor(0, 0);
  miDisplay.print("   Display OK   ");
  delay(3000);
}
void loop() {
  miDisplay.setCursor(0, 0);  miDisplay.print("     FILA 1   ");
  delay(1500);                miDisplay.clear();
  miDisplay.setCursor(0, 1);  miDisplay.print("     FILA 2   ");
  delay(1500);                miDisplay.clear();
}
```

¡Nuevos conceptos!

- Salidas Digitales → **TIMERS**
- Entradas Digitales → **INTERRUPCIONES**
  ANTI-REBOTE
- Funciones → **ARCHIVOS, LIBRERÍAS**

TIMERS con TICKER

```
#include <Ticker.h>  //Ticker Library
Ticker t1, t2, t3;

#define LED_1 D4
#define LED_2 D8

void TIMER_1(){
  digitalWrite(LED_1, !digitalRead(LED_1));
}

void TIMER_2(){
  digitalWrite(LED_2, !digitalRead(LED_2));
}

void TIMER_3(){
  Serial.println("TIMER 3");
}
```

```
void setup() {
  delay(400);
  Serial.begin(115200);
  Serial.println("\n\nTimers con TICKER");
  pinMode(LED_1,OUTPUT);
  pinMode(LED_2,OUTPUT);
  t1.attach(0.5, TIMER_1);
  t2.attach(  1, TIMER_2);
  t3.attach(  2, TIMER_3);
}

void loop() {
}
```

INTERRUPCIONES

```
#define LED_1    D4
#define LED_2    D8
#define SW_1     D0
#define SW_2     D3

//Pulsador 2 trabaja por INTERRUPCION.
//Detección INMEDIATA. Interrumpe el ciclo principal
void ICACHE_RAM_ATTR rutinaP2() {
  digitalWrite(LED_2,HIGH);
  //Mientras no suelte pulsador 2, espera.
  while(digitalRead(SW_2)==LOW);
  digitalWrite(LED_2,LOW);
}
```

```
void setup() {
  pinMode(LED_1,  OUTPUT);
  pinMode(LED_2,  OUTPUT);
  attachInterrupt(SW_2, rutinaP2, FALLING);
}

void loop() {
  //SW_1 hace CONSULTA. Puede perder eventos
  //O tardar hasta 3 segundos en detectar cambio
  if(digitalRead(SW_1)==HIGH)
      digitalWrite(LED_1,HIGH);
  else
      digitalWrite(LED_1,LOW);
  delay(3000);
}
```

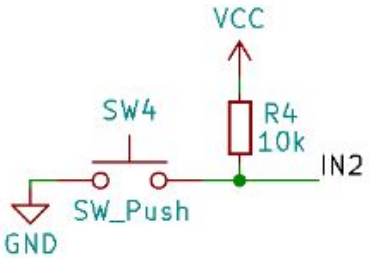
ANTI-REBOTE

```
int contador = 0;
void ICACHE_RAM_ATTR rutinaP2() {
  contador++;
}

void setup() {
  delay(400);
  Serial.begin(115200);
  Serial.println("\n\nPrueba anti Rebote");
  attachInterrupt(SW_2, rutinaP2, FALLING);
}

void loop() {
  Serial.println(contador);
  delay(500);
}
```

comportamiento errático



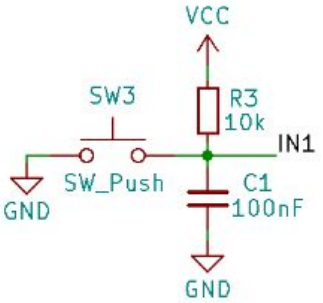
ANTI-REBOTE

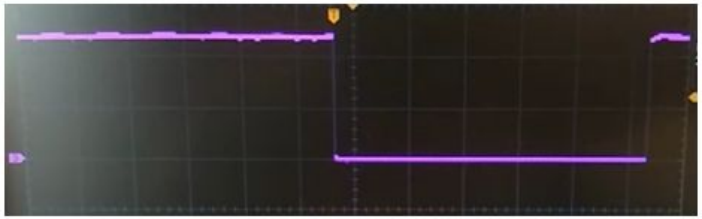
```
int contador = 0;
void ICACHE_RAM ATTR rutinaP2() {
  delay(10); //doble verificación
  if(digitalRead(SW_2)==LOW){
    contador++;
  }
  while(digitalRead(SW_2)==LOW);
  delay(50);
}

void setup() {
  delay(400);
  Serial.begin(115200);
  Serial.println("\n\nPrueba anti Rebote");
  attachInterrupt(SW_2, rutinaP2, FALLING);
}

void loop() {
  Serial.println(contador);
  delay(500);
}
```

comportamiento mejorado
SOFTWARE + HARDWARE





ARCHIVOS, LIBRERÍAS

```ino
#include "miSondaTemperatura.h"

void setup() {
  delay(400);
  Serial.begin(115200);
  configuraSondaTemperatura();
}

void loop() {
  float temperatura = leeSondaTemperatura();
  Serial.print("Temperatura = ");
  Serial.print(temperatura);
  Serial.println("  C");
  delay(2000); // Espera 2 segundos
}
```

miSondaTemperatura.h

```h
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS D6
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
void configuraSondaTemperatura(){
  Serial.println("Iniciando DS18B20...");
  sensors.begin();
}
float leeSondaTemperatura(){
  sensors.requestTemperatures();
  float temperatureCelsius = sensors.getTempCByIndex(0);
  if (temperatureCelsius == -127.00){
    Serial.println("Error al leer la temperatura.");
  }
  return temperatureCelsius;
}
```