Microcontroladores

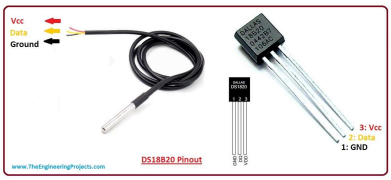
Semana 14

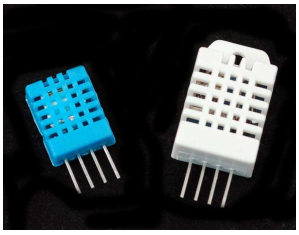
Por Kalun Lau

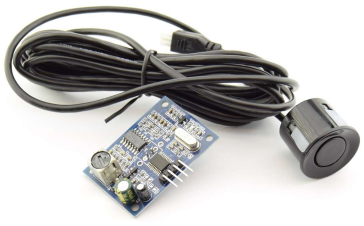
1

Agenda:

- Manejo del DHT11/DHT22
- Manejo del HC-SR04
- Funcionamiento del DS18B20 (1-wire)







2

DHT11/DHT22

- Sensores para medir la humedad y temperature del ambiente.
- Ampliamente usados en proyectos de sistemas embebidos con Arduino.

DHT11

- Ultra low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 20-80% humidity readings with 5% accuracy
- Good for 0-50°C temperature readings ±2°C accuracy
- No more than 1 Hz sampling rate (once every second)
- Body size 15.5mm x 12mm x 5.5mm
- 4 pins with 0.1" spacing

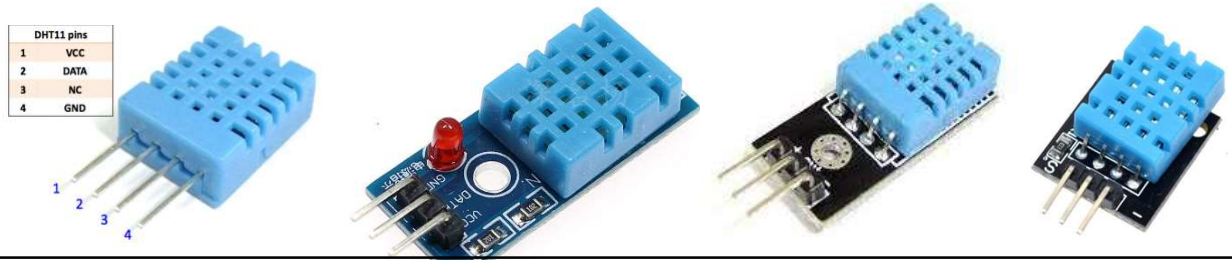
DHT22 / AM2302 **(Wired version)**

- Low cost
- 3 to 5V power and I/O
- 2.5mA max current use during conversion (while requesting data)
- Good for 0-100% humidity readings with 2-5% accuracy
- Good for -40 to 80°C temperature readings ±0.5°C accuracy
- No more than 0.5 Hz sampling rate (once every 2 seconds)
- Body size 15.1mm x 25mm x 7.7mm
- 4 pins with 0.1" spacing

3

El DHT11

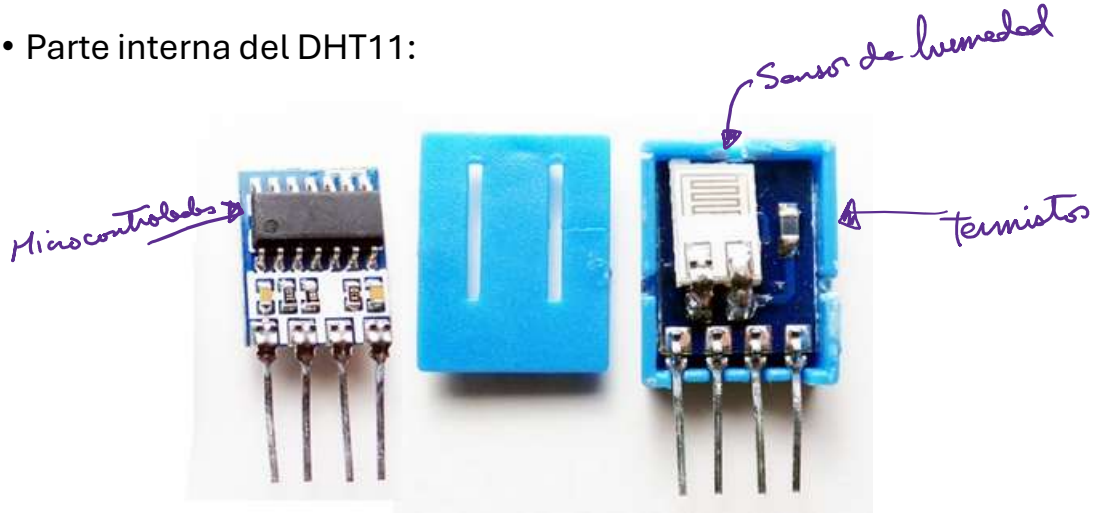
- Aspectos iniciales:
  - Al revisar la hoja técnica del DHT11 podemos ver que el DHT11 tiene un rango de voltaje de operación de 3V a 5.5V por lo que la conexión hacia el Curiosity Nano IC18F57Q43 será de manera directa.
  - Dependiendo del modelo de DHT11 puede que tenga integrado la resistencia de pull-up, sobre todo lo que tienen el sensor montado en una PCB:
  - Hoja técnica: https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf



4

El DHT11

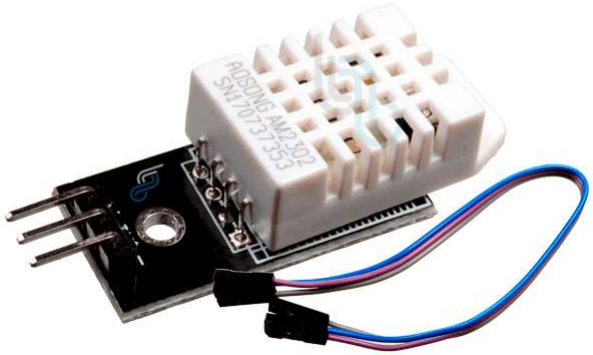
- Parte interna del DHT11:



5

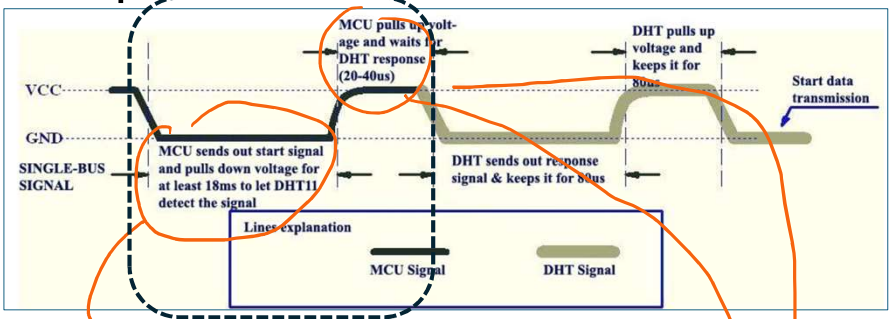
El DHT22

- También se le puede encontrar como AM2302 AOSONG
- De funcionamiento similar al DHT11, pero con mayor resolución en las medidas de temperatura y humedad



6

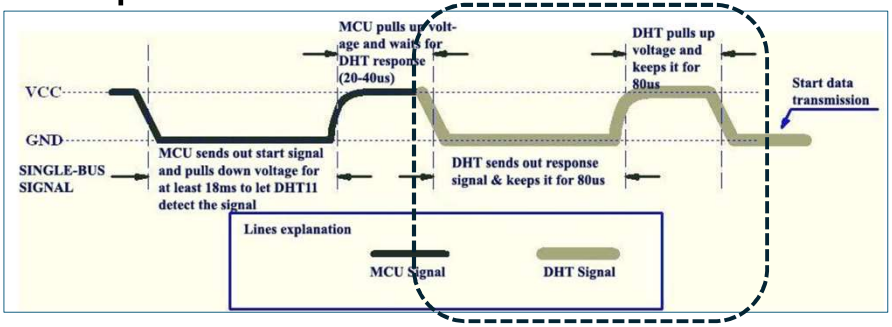
Inicio del proceso de comunicación con el DHT11




```
void DHT11_Start(void){
    TRISBbits.TRISB0 = 0;       //Puerto RB0 como salida
    ANSELBbits.ANSELB0 = 0;     //Para que sea digital
    LATBbits.LATB0 = 0;         //Mandamos cero
    __delay_ms(18);             //Esperamos 18 milisegundo
    LATBbits.LATB0 = 1;         //Mandamos uno
    __delay_us(40);             //Esperamos 40 microsegundos
    TRISBbits.TRISB0 = 1;       //Puerto como entrada
}
```

7

Inicio del proceso de comunicación con el DHT11




```
void DHT11_Check(void){
    while(PORTBbits.RB0);
    __delay_us(80);
    while(!PORTBbits.RB0);
    __delay_us(80);
    while(PORTBbits.RB0);
}
```

while (PORTBbits.RB0 == 1)

while (PORTBbits.RB0 != 1)

8

Proceso de envío de datos del DHT11 al MCU

- En el proceso de envío de datos (son 5 bytes ó 40 bits)

Data consists of decimal and integral parts. A complete data transmission is **40bit**, and the sensor sends **higher data bit** first.
Data format: 8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data + 8bit check sum. If the data transmission is right, the check-sum should be the last 8bit of "8bit integral RH data + 8bit decimal RH data + 8bit integral T data + 8bit decimal T data".

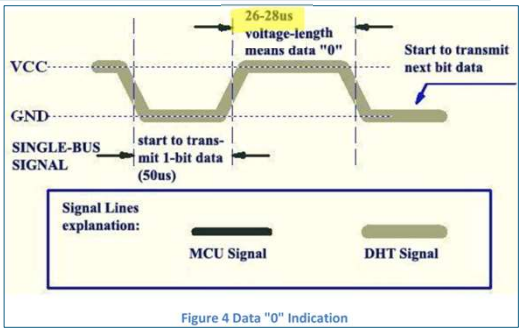


Figure 4 Data "0" Indication

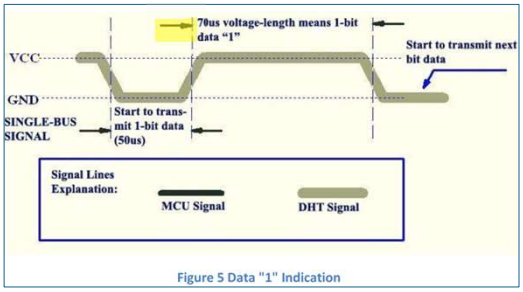


Figure 5 Data "1" Indication

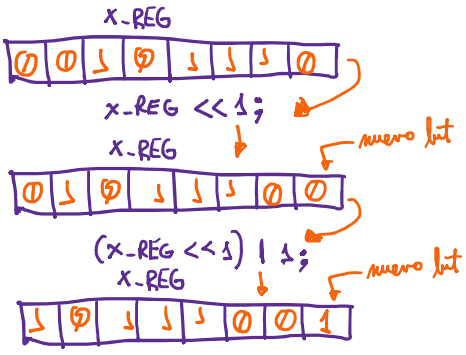
```
unsigned char DHT11_Read(void){
    unsigned char x = 0, data = 0;
    for(x=0;x<8;x++){
        while(!PORTBbits.RB0);
        __delay_us(35);
        if(PORTBbits.RB0){
            data = ((data<<1) | 1);
        }
        else{
            data = (data<<1);
        }
        while(PORTBbits.RB0);
    }
    return data;
}
```

Lee un dato de ocho bits

Haciendo cinco veces!

9

Entendiendo las operaciones de desplazamiento de datos en un registro

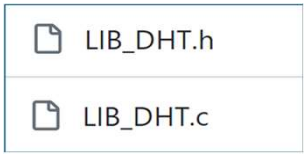


5

10

La nueva librería DHT (LIB_DHT)

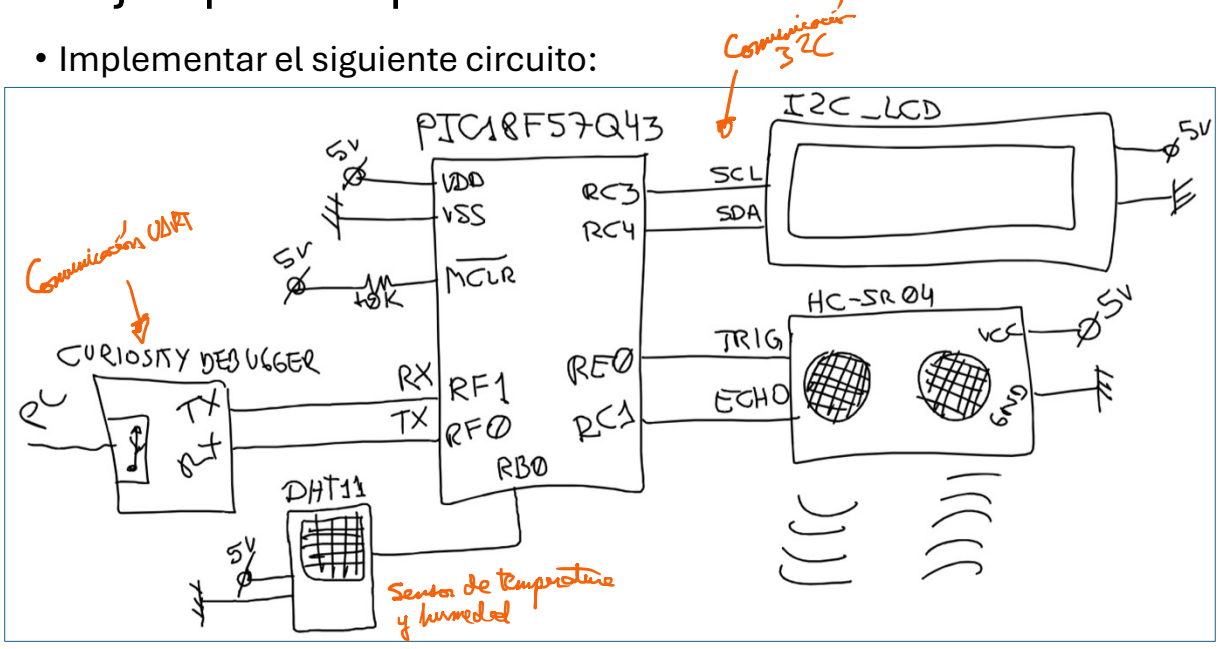
- Referencia: https://github.com/tocache/Microchip-PIC18F57Q43/tree/main/ELEC225%202024-1%20Examples/Prueba_DHT22_I2CLCD.X
- Soporta dispositivos DHT11 y DHT22



11

Ejemplo de aplicación 2024-1

- Implementar el siguiente circuito:



12

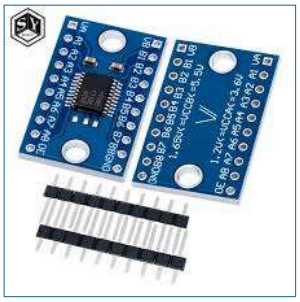
Uso de las librerías I2C_LCD y LIB_DHT

```
#include "I2C_LCD.h"
#include "LIB_DHT.h"

unsigned int temperatura=0;
unsigned int humedad=0;

void configuro(void){
    OSCCON1 = 0x60;
    OSCFRQ = 0x06;
    OSCEN = 0x40;
}
void main(void) {
    configuro();
    I2C_LCD_INIT();
    while(1){
        temperatura = DHT_GetTemp(DHT22);
        __delay_ms(500);
        humedad = DHT_GetHumid(DHT22);
        __delay_ms(500);
        I2C_POS_CURSOR(1,0);
        I2C_ESCRIBE_MENSAJE2("Temp: ");
        I2C_LCD_ESCRIBE_VAR_INT(temperatura, 3, 1);
        I2C_ENVIA_LCD_DATA(0xDF);
        I2C_ENVIA_LCD_DATA('C');
        I2C_POS_CURSOR(2,0);
        I2C_ESCRIBE_MENSAJE2("Humid: ");
        I2C_LCD_ESCRIBE_VAR_INT(humedad, 3, 1);
        I2C_ESCRIBE_MENSAJE2("% RH");
    }
}
```

13

El sensor HC-SR04

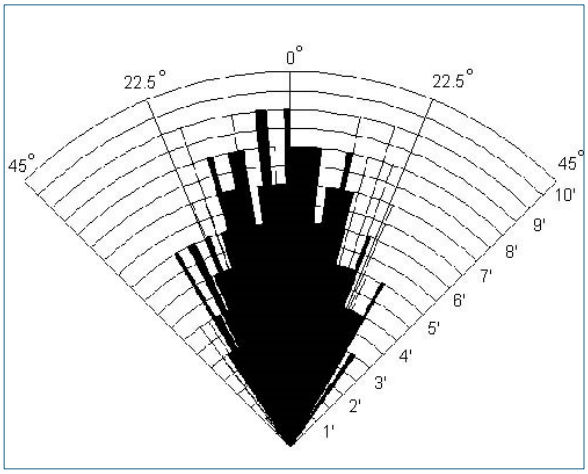
- El sensor HC-SR04 es un módulo que permite medir la distancia entre éste y un objeto entre un rango de 3cm y 300cm.
- Trabaja con alimentación de 5V por lo que se necesitará de un conversor de niveles lógicos 3.3V-5V, se recomienda emplear el TXS0108 en caso de que el microcontrolador trabaje en LVTTL
- Los transductores trabajan en 40KHz



14

El sensor HC-SR04

- Tener en cuenta que la emisión ultrasónica no es muy directiva, a mayor distancia que recorre el sonido emitivo mayor será la dispersión.



15

El sensor HC-SR04

- Cálculo de la distancia entre el sensor y el obstáculo:

Velocidad del sonido: 343 m/s (a nivel del mar y a 20°C)

↳ 0.0343 cm/µs ← convertido según resolución del Timer1 y las distancias a medir (de 2cm a 4000cm)

4 metros en condiciones ideales

$$\text{Distancia} = \frac{(\text{velocidad del sonido}) \times \text{duración}}{2}$$

lo pasamos a cociente

$$\text{Distancia} = \frac{(0.0343)\ \text{cm/µs} \times \text{duración en microsegundos}}{2 \leftarrow \text{ida y vuelta}} = 0.01715 \times \text{duración} \Rightarrow \frac{\text{duración}}{58.31}$$

16

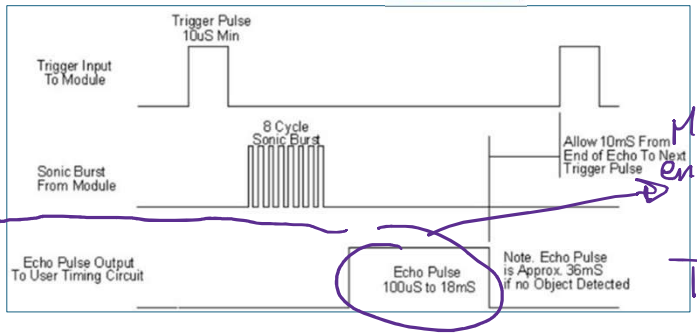
El sensor HC-SR04

- Posee dos puertos:
  - Trigger: El microcontrolador host le envía al puerto "trigger" del HC-SR04 un pulso activo en alto de 10us para que este último envíe ocho pulsos de sonido de 40KHz.
  - Echo: Luego del envío de los ocho pulsos de sonido de 40KHz el HC-SR04 enviará un pulso al microcontrolador host con determinado ancho, este ancho representará el tiempo del eco y mediante un cálculo matemático se obtendrá la distancia entre el módulo y el objeto.

```
T1CLK = 0x01;       //Timer1 con fuente de reloj FOSC/4
T1CON = 0x32;       //presc 1:8, rd16=1, async, timer1 OFF = 1us x cta
```

base de tiempo

```
LATEbits.LATE0 = 1;
__delay_us(10);
LATEbits.LATE0 = 0;
while(PORTCbits.RC0 == 0);
T1CONbits.ON = 1;
while(PORTCbits.RC0 == 1);
T1CONbits.ON = 0;
```



Medir el tiempo en este pulso

Timer1

17

```
#include <xc.h>
#include "cabecera.h"
#include "I2C_LCD.h"
#define _XTAL_FREQ 32000000UL

void configuro(void){
    OSCCON1 = 0x60;
    OSCFRQ = 0x06;
    OSCEN = 0x40;
    //Configuracion de E/S para los HCSR04
    TRISE = 0xFE;       //RE0 como salida
    ANSELE = 0xFE;      //RE0 como digital
    TRISCbits.TRISC1 = 1;   //RC1 como entrada
    ANSELCbits.ANSELC1 = 0;   //RC1 como digital
    T1CLK = 0x01;       //Timer1 con fuente de reloj FOSC/4
    T1CON = 0x32;       //presc 1:8, rd16=1, async, timer1 OFF = 1us x cta
}
```

```
unsigned int Read_HCSR04(void){
    TMR1H = 0;
    TMR1L = 0;
    LATEbits.LATE0 = 1;
    __delay_us(10);
    LATEbits.LATE0 = 0;
    while(PORTCbits.RC1 == 0);
    T1CONbits.ON = 1;
    while(PORTCbits.RC1 == 1);
    T1CONbits.ON = 0;
    return (TMR1);
}
```

```
unsigned int calculo(unsigned int valor){
    float tiempo;
    unsigned int distancia;
    tiempo= (float)valor;
    tiempo= tiempo / 5.8;
    distancia = (int)tiempo + 1;
    return distancia;
}
```

El sensor HC-SR04

- Código propuesto para leer la distancia usando el HC-SR04

```
void main(void) {
    configuro();
    //Inicializacion del I2C LCD
    I2C_LCD_INIT();
    I2C_POS_CURSOR(1,0);
    I2C_ESCRIBE_MENSAJE2("Medidor HC-SR04");
    while(1){
        I2C_POS_CURSOR(2,0);
        I2C_ESCRIBE_MENSAJE2("Dist: ");
        I2C_LCD_ESCRIBE_VAR_INT(calculo(Read_HCSR04()),3,1);
        I2C_ESCRIBE_MENSAJE2(" cm");
        __delay_ms(100);
    }
}
```

18

Fin de la sesión

19