Universidad Francisco de Vitoria

Programación con Arduino II

Práctica Final

Diego Viñals Lage

Contenido

Introducción

En esta practica tenemos que conectar 4 placas Arduino Uno en Tinkercad, en la que la placa Máster lea un valor mediante un potenciómetro. Este valor, se traslada a las otras tres placas Slave, en la que se realizara una serie de acciones.

En la primera placa Slave, se pasará el numero dado del potenciómetro a binario, y mediante el registro de desplazamiento 74HC595, se encenderán 8 Leds acorde con el numero en binario, por ejemplo, si el numero es 0, estarán todos los Leds apagados (0 en binario es 0). Si el número es 255, se encenderán todos los Leds (255 en binario es 11111111).

En la segunda placa Slave se tiene que conectar un display de 16 segmentos, en el que se imprimirá el numero del potenciómetro en decimal, hexadecimal, octal y binario.

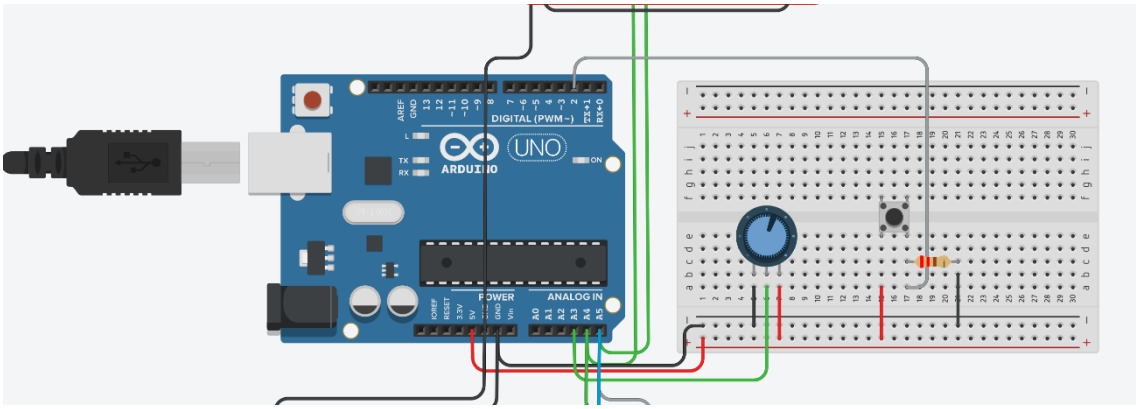
En la tercera placa Slave, se conecta un motor stepper, en el que las revoluciones por minuto vienen dadas por el valor del potenciómetro en la placa Máster.

Placa Máster

En la placa Máster, como ya he dicho antes, conectamos un potenciómetro. Este potenciómetro lo he conectado al pin A3 (analógico). Este pin lo que hago con el código es leerlo, y mediante el protocolo I2c visto en clase, mandamos el valor a las otras tres placas Slave.

A parte del potenciómetro tenemos un botón que cuando se presiona manda el valor del potenciómetro. Este valor solo se manda y solo se actualiza cuando el botón se pulsa, por lo que el usuario decide cuando cambiar de valor.

El valor del potenciómetro se lee con la función AnalogRead(pin), y lo mandamos con la función Wire.write().



Código:

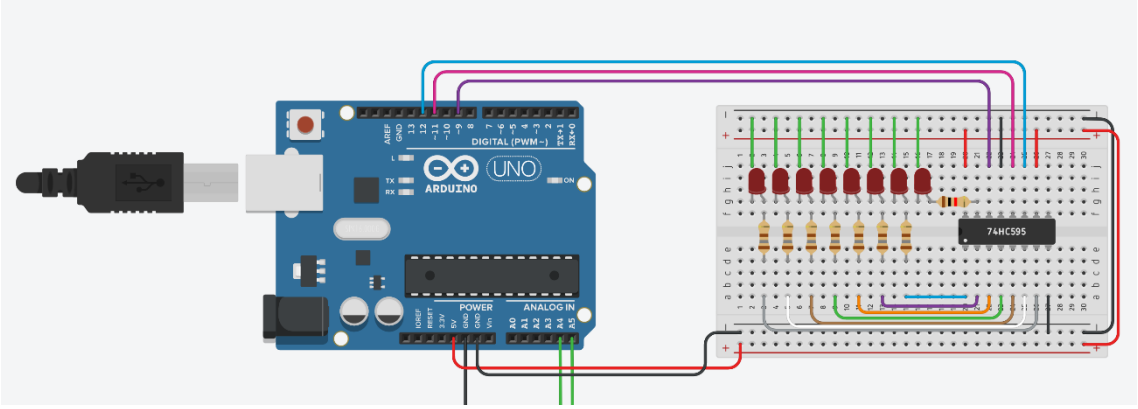
```
#include <Wire.h>

/* MASTER */

const int pulsador = 2;
byte device = 0; // Variable que almacenará la dirección del dispositivo
byte data = 0;

void setup() {
  pinMode(pulsador, INPUT); // Activamos el pulsador como entrada.
  pinMode(A3, INPUT);                    // Activamos el pin A3 como entrada, que es el pin del potenciometro.
  Serial.begin(9600);              // Para poder ver el monitor serie lo que haga falta.
  Serial.println ("Escaneando dispositivos I2C: ");
  byte count = 0;
  Wire.begin();                          // Empezamos la transmision de datos a las otras placas Slave.
  for(byte i = 1; i < 120; i++) {
    Wire.beginTransmission(i);

    if (Wire.endTransmission () == 0) {
      Serial.print("Encontrado dispositivo en la direccion: ");
      Serial.print(i, DEC);
      Serial.print(" (0x");
      if(i<16) Serial.print("0");
      Serial.print(i, HEX);
      Serial.println(")");
      count++;
      delay(1);
```

```
    device = i;
   }

  }

  Serial.println("Comienza la transmision, selecciona el monitor serie del otro dispositivo.");
}

void loop() {

  // boton nos dice si el boton esta pulsado o no. Low es no pulsado, HIGH es pulsado.
  int boton = digitalRead(pulsador);
  // Si el boton se pulsa, se ejecuta lo de dentro del if.
  if(boton == HIGH){
                Wire.beginTransmission(device); // Transmitimos al dispositivo encontrado en el setup
                // Mapeamos el valor del potenciometro, que se encuentra entre 0
    // y 1023, y nosotros queremos un valor entre 0 y 255.
    int valor = map(analogRead(A3), 0, 1023, 0, 255);
    // Escribimos el valor del numero
    Wire.write("Valor de data: "); // Enviamos 15 bytes (caracteres)
    Wire.write(valor); // Enviamos un byte (entero)
                // Finalizamos la transmision para que solo se envie el dato cuando el
    // boton esta pulsado.
    Wire.endTransmission();
  }

}
```

Placa Slave_1

En la placa Slave_1 tenemos 8 Leds conectados a la placa de Arduino mediante un registro de desplazamiento 74HC595. Esta placa recibe el dato enviado por la placa máster y lo pasa a binario. Mediante un for loop lee cada bit de este numero, y si es 1, enciende el led de ese valor y si es 0, lo apaga. Así conseguimos que en los leds se lea el numero en binario. Esta parte ha sido sencilla de hacer ya que es mas sencillo que la practica de Arduino I del año pasado.

Mediante el protocolo Ic2, recibe el dato del máster, si solo se ejecuta cuando recibe el dato, por lo que los leds se mantendrán encendidos o apagados hasta que se presione el botón y se cambie de numero.



Código:

```
/* SLAVE_1 */

#include <Wire.h>
// Variables que almacenan los pines necesario para el registro
// de desplazamiento 74HC595
const int data_pin = 9;
const int latch_pin = 11;
const int clock_pin = 12;

void setup() {
  Wire.begin(40); // Configuramos el dispositivo con la dirección 40
  Wire.onReceive(receiveEvent); // Evento que se ejecutará cuando reciba datos
  Serial.begin(9600);                    // Inicializamos el Monitor serie para lo que haga falta.
  pinMode(data_pin, OUTPUT);   // Iniciamos los pines del registro en modo salida, para poder escribit los datos
  pinMode(clock_pin, OUTPUT);
  pinMode(latch_pin, OUTPUT);

}

void loop() {
  delay(100);
}

// Función que se ejecutará cuando cualquier dato se reciba del maestro
void receiveEvent(int howMany) {
  int x = 0;
  //Serial.print("How Many: ");
  //Serial.println(howMany);
  while(Wire.available() > 1) { // Leemos todos los datos excepto el último
                        // (que es el número entero que incrementamos en el master)
```

```
  char c = Wire.read(); // Leemos los bytes que son caracteres
  //Serial.print(c);
 }
 x = Wire.read(); // Leemos el byte restante como entero
 // Imprimos el valor en el monitor serie, por si acaso.
 Serial.print("Numero del Master: ");
 Serial.println(x);
 byte binario = x;       // Almacenamos como byte el numero del potenciometro de la placa master.
 digitalWrite(latch_pin,0);
 //bitClear(binario , 0);
 for(int i = 0 ; i <= 7 ; i++){
  // a es el bit numero i del byte.
  byte a = bitRead(binario, i);
  // si es uno, activamos el led numero i como encendido
  if(a == 1){
    shiftOut(data_pin, clock_pin, MSBFIRST, bitSet(binario, i));

  }else {
    // si es 0, apagamos el led numero i.
    shiftOut(data_pin, clock_pin, MSBFIRST, bitClear(binario , i));
  }
 }

 digitalWrite(latch_pin,1);
}
```

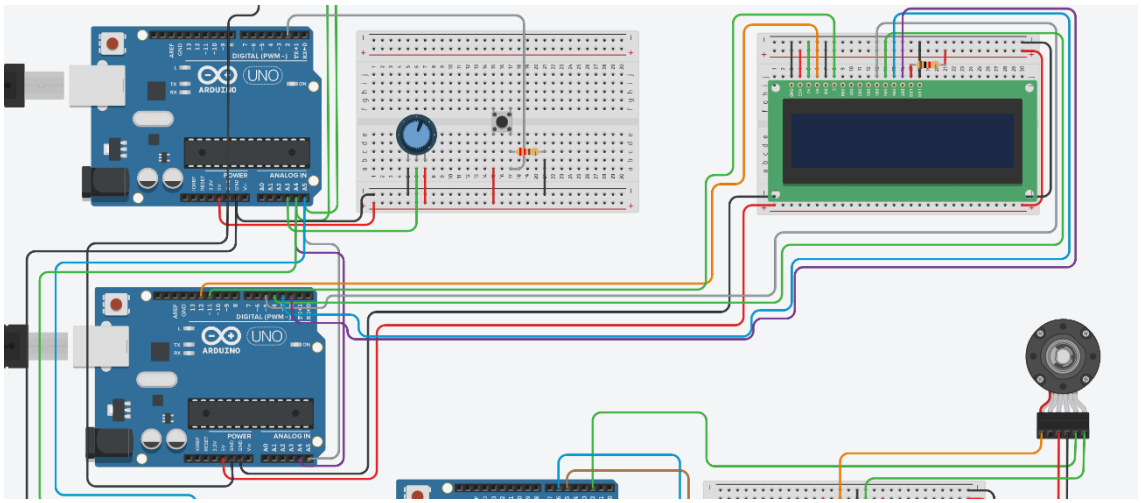
Placa Slave_2

En la placa Slave_2 tenemos conectado una pantalla LCD en la cual, mediante la librería Liquid Crystal vamos escribiendo información.

Lo primero que hacemos es leer el dato que nos llega de la placa máster. Este dato lo escribimos en decimal, tal cual nos llega. Luego ponemos un delay para que se pueda leer, y lo escribimos en Hexadecimal. En Arduino es tan fácil como poner el numero(x), y luego HEX, esto lo convierte automáticamente a Hexadecimal. Lcd.print(x , HEX); te imprime en pantalla el valor de x en Hexadecimal.

Ponemos otro delay de 1 segundo, y borramos la pantalla con lcd.Clear(), ponemos el cursor al principio y escribimos en Octal. En vez de HEX pongo OCT y me escribe el numero en Octal, por lo que no tendríamos que hacer el cambio nosotros mismos.

Añadimos otro delay de 1 segundo, y esta vez escribimos el numero en binario. Borramos pantalla con la misma función de antes, y escribimos binario con el código BIN. Escribe el numero en binario, pero los ceros a la izquierda no los escribe, por lo que si el numero dado es 1, por ejemplo, en pantalla se vería 1 y no 00000001.



Código:

```
/* SLAVE_2 */

#include <Wire.h>
#include <LiquidCrystal.h>
// Creamos un nuevo objeto lcd con los pines conectados.
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
  Wire.begin(40); // Configuramos el dispositivo con la dirección 40
  Wire.onReceive(receiveEvent); // Evento que se ejecutará cuando reciba datos
  Serial.begin(9600);                  // Activamos el monitor serie.
```

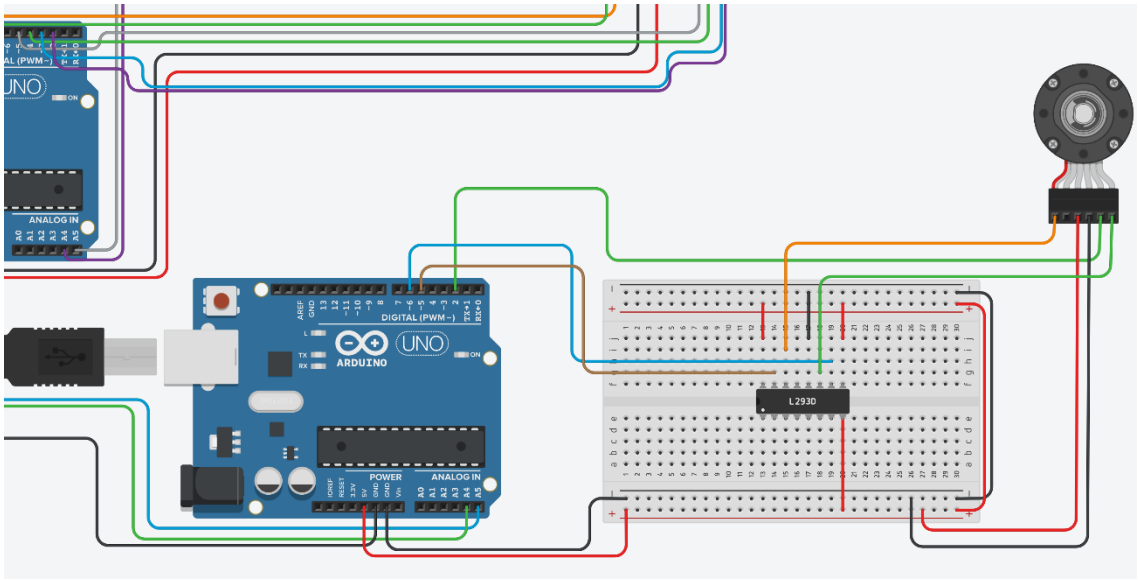
```
  lcd.begin(16, 2);                         // Encendemos la pantalla Led.

}

void loop() {
  delay(100);
}

// Función que se ejecutará cuando cualquier dato se reciba del maestro
void receiveEvent(int howMany) {

  while(Wire.available() > 1) { // Leemos todos los datos excepto el último
                        // (que es el número entero que incrementamos en el master)
    char c = Wire.read(); // Leemos los bytes que son caracteres

  }
  int x = Wire.read(); // Leemos el byte restante como entero
  Serial.print("Numero del Master: ");
  Serial.println(x);
  // Borramos lo que haya en pantalla.
  lcd.clear();
  lcd.setCursor(0, 0);              // Ponemos el cursor en la fila 0 columna 0
  lcd.print("Decimal:");            // Imprimos en el led "Decimal"
  lcd.setCursor(0,1);               // Pasamos a la siguiente fila
  lcd.print(x);                                 // Como el numero lo recibimos en decimal
                                                                  // No hace falta hacer nada, lo imprimos tal cual
  delay(10000);                                 // Delay para poder leer bien el numero

  lcd.clear();                                  // Borramos pantalla
  lcd.setCursor(0, 0);
  lcd.print("Hexadecimal:");// Escribimos Hexadecimal
  lcd.setCursor(0,1);     // Ponemos el cursor en la siguiente fila
  lcd.print(x, HEX);                // Mediante HEX, imprimimos el numero en Hexadecimal
  delay(10000);                                 // Delay

  lcd.clear();                                  // Borramos pantalla
  lcd.setCursor(0, 0);
  lcd.print("Octal:");              // Imprimimos "Octal"
  lcd.setCursor(0,1);     // Pasamos a la siguiente fila
  lcd.print(x, OCT);                // Mediante "OCT" imprimimos el numero en Octal
  delay(10000);                                 // Delay

  lcd.clear();                                  // Borramos pantalla
  lcd.setCursor(0, 0);
  lcd.print("Binario:"); // Escribimos "Binario"
  lcd.setCursor(0,1);     // Pasamos a la siguiente fila
  lcd.print(x, BIN);                // Mediante BIN imprimimos el numero en binario
  delay(10000);                                 // Delay


}
```

Placa Slave_3

En la placa Slave_2 tenemos conectado un chip L293D y un motor stepper, de corriente continua. Lo que hace esta placa Slave es coger el dato de la placa Master y se lo pasa al chip y pone el motor con las revoluciones dadas por la placa Master.

El motor esta conectado al pin 2 de la placa Slave en forma de INPUT_PULLUP y el chip de OUTPUT, para que así poder decirle al chip a que velocidad o revoluciones tiene que ir el motor.

Esta placa tiene poco código ya que solo es decirle a qué velocidad tiene que ir el motor con un simple analogWrite(pin, x) siendo x el valor dado por la placa Master.



Código:

```
/*Slave_3*/
#include <Wire.h>

void setup() {

  pinMode(5, OUTPUT);
  pinMode(6,OUTPUT);
  pinMode(2,INPUT_PULLUP);

  Wire.begin(40); // Configuramos el dispositivo con la dirección 40
  Wire.onReceive(receiveEvent); // Evento que se ejecutará cuando reciba datos
  Serial.begin(9600);

}

void loop() {
  delay(100);
}

// Función que se ejecutará cuando cualquier dato se reciba del maestro
void receiveEvent(int howMany) {

  while(Wire.available() > 1) { // Leemos todos los datos excepto el último
                        // (que es el número entero que incrementamos en el master)
```

```
  char c = Wire.read(); // Leemos los bytes que son caracteres
 }
 int x = Wire.read(); // Leemos el byte restante como entero
 Serial.print("Numero del Master: ");
 Serial.println(x);
 analogWrite(5,x);
}
```

Diseño del circuito