Universidad Iberoamericana Puebla

Departamento de Ciencias e Ingenierías

Laboratorio de Redes Digitales

Alvaro Zaid Gallardo Hernández

Patricia Mercedes Cortés Garcia

Rosa Elena Mar Larios

Date: 19th February 2023 Maestro

Word count: 2,999 Huber Giron Nieto

Practica 3: "Protocolo de Comunicación Asíncrono"

Contents

1 Resumen

Los ejercicios proporcionados consisten en dos partes que se comunican entre sí utilizando un protocolo de comunicación simple. La primera parte consiste en un programa que convierte el valor analógico de un potenciómetro en un número binario de 8 bits y lo envía a través de un pin digital en formato serie. La segunda parte es un programa que recibe el número binario de 8 bits y lo convierte en un número decimal.

2 Introducción

Un protocolo de comunicación asíncrono es un tipo de protocolo de comunicación en el que los datos se transmiten sin un reloj centralizado para sincronizar la comunicación entre los dispositivos. En lugar de utilizar un reloj, los dispositivos utilizan señales de control para coordinar la transmisión de datos.

En un protocolo de comunicación asíncrono, los datos se transmiten en paquetes o tramas, que contienen tanto los datos como la información de control necesaria para la transmisión. Los dispositivos pueden enviar y recibir datos en cualquier momento, sin esperar a que el

otro dispositivo esté listo. Esto hace que los protocolos de comunicación asíncronos sean más flexibles y adaptables a diferentes situaciones de comunicación.

3 Objetivos

3.1 General

Conocer, identificar y comprobar el funcionamiento al desarrollar un protocolo de comunicación asíncrono.

3.2 Específicos

1. Utilizando 1 Arduino, diseñar un sistema que permita convertir el valor analógico de un potenciómetro en un arreglo de variables que representen el numero en binario de 8 bits, y los deposite en un arreglo [ ], Finalmente elaborar un código de programación que permita enviar los bits del arreglo en orden y en serie siguiendo las siguientes características:
2. Iniciar el envió de los datos siempre con una señal HIGH (En total se enviarán 9 datos, el primer bit en HIGH y posteriormente los 8 bits de información).
3. Enviar los datos con una duración del pulso predeterminada, ejemplo 1000 ms
4. Enviar siempre la misma cantidad de bits.
5. Utilizando un segundo Arduino, diseñar una función que permita adquirir el valor en bits de la señal de comunicación del Arduino anterior, la procese y la convierte en un dato decimal que finalmente enviara al monitor serial.

4 Materiales

- 2 Arduinos
- Protoboard
- Botones
- LED's
- Potenciómetro

5 Procedimiento

Primer Arduino

```
int pot=0;

void setup()
{
  pinMode(8, OUTPUT);
  Serial.begin(9600);
  delay(2000);
  pot=analogRead(A5);
  pot=map(pot,0,1023,0,255);
  Serial.println(pot);

  digitalWrite(8, HIGH);
  delay(1000);
  for(int i=0; i<8; i++)
  {
    digitalWrite(8,pot%2);
    delay(1000);
    Serial.print(pot%2);
    pot=pot/2;
  }
  digitalWrite(8, LOW);
  delay(1000);

  Serial.println("");

}

void loop()
{}
```

Este código es la implementación del envío de datos desde el primer Arduino al segundo. El código utiliza el mismo circuito que se ha utilizado para la lectura del valor analógico del potenciómetro. En lugar de mostrar el valor binario del potenciómetro en los pines del Arduino, este valor se envía al segundo Arduino a través de un cable USB.

En el código, el valor del potenciómetro se lee y se convierte en un valor de 8 bits (0 a 255) utilizando la función "map()". Luego, cada bit de este valor se envía uno por uno al segundo Arduino utilizando el pin digital 8 como la línea de datos. El bit se envía utilizando la función "digitalWrite()" y se espera un segundo después de cada bit enviado

utilizando la función "delay()".

La comunicación entre los dos Arduinos se realiza a través del puerto serie utilizando la función "Serial.begin()" y "Serial.println()".

Segundo Arduino

```
int bit;
int suma=0;
int potencia=1;

void setup()
{
  pinMode(8, INPUT);
  Serial.begin(9600);
}

void loop()
{
  if(digitalRead(8)==HIGH)
  {
    delay(1050);

    for(int i=8; i>0; i--)
        {
      bit=digitalRead(8);
      Serial.println(bit);
      suma=suma +(bit*potencia);
      potencia=potencia*2;
      delay(1000);
        }
    Serial.println(suma);
  }
  suma=0;
  potencia=1;
}
```

Este código es un ejemplo de comunicación serial entre dos placas Arduino, donde una envía el valor de un potenciómetro convertido a un número binario de 8 bits y la otra placa recibe y decodifica la información para obtener el valor del potenciómetro.

El primer código "ENVIO potenciometro" se encarga de leer el valor analógico de un

potenciómetro y convertirlo en un número binario de 8 bits, que envía a través del pin 8 de la placa. El código utiliza la función digitalWrite() para enviar cada bit, esperando un segundo entre cada bit para asegurarse de que se reciba correctamente.

El segundo código "RECIBE POTENCIOMETRO" se encarga de recibir la información enviada por la primera placa a través del pin 8. El código utiliza la función digitalRead() para leer cada bit, esperando un segundo entre cada bit para asegurarse de que se reciba correctamente. Luego, el código decodifica los bits para obtener el valor original del potenciómetro en formato decimal.

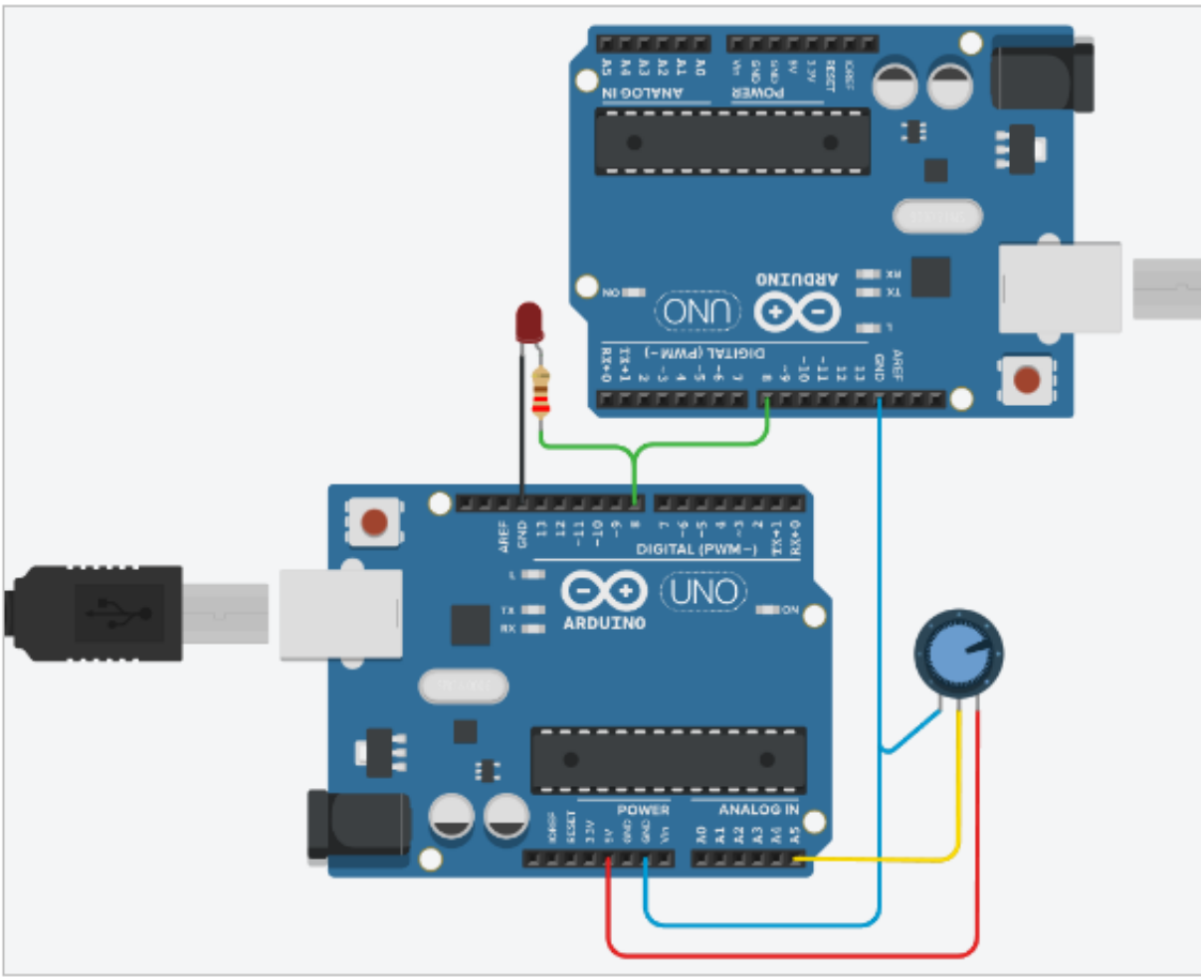

Figure 1: Esquema

Enlace: https://youtube.com/shorts/XG4w-t3ap-o?feature=share

6 Conslusión

Se ha logrado establecer una comunicación efectiva entre dos Arduinos, lo que puede tener aplicaciones prácticas en la construcción de sistemas distribuidos y en el control de dispositivos en tiempo real. Además, se ha desarrollado la habilidad para la programación de Arduinos y la comprensión del funcionamiento del protocolo de comunicación serial.

7 Referencias

Bertoleti, P. (2019). Proyectos com ESP32 y LoRa. Editora NCB.

Babiuch, M., Foltýnek, P., Smutný, P. (2019, May). Using the ESP32 microcontroller for data processing. In 2019 20th International Carpathian Control Conference (ICCC) (pp. 1-6). IEEE.