PRACTICA 5 : Buses de comunicación II (SPI)

El objetivo de la practica es comprender el funcionamiento del bus spi

Introducción teórica

BUS SPI

El bus SPI (Serial Peripheral Interface) fue desarrollado por Motorola en 1980. Sus ventajas respecto a otros sistemas han hecho que se convierta en un standard de facto en el mundo de la electrónica y automatización.

El bus SPI tiene una arquitectura de tipo maestro-esclavo. El dispositivo maestro (master) puede iniciar la comunicación con uno o varios dispositivos esclavos (slave), y enviar o recibir datos de ellos. Los dispositivos esclavos no pueden iniciar la comunicación, ni intercambiar datos entre ellos directamente.

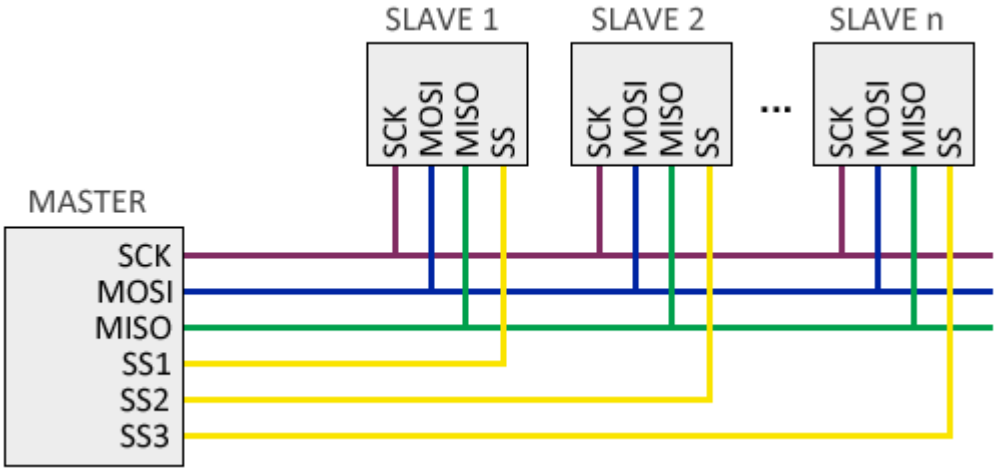
En el bus SPI la comunicación de datos entre maestros y esclavo se realiza en dos líneas independientes, una del maestro a los esclavos, y otra de los esclavos al maestro. Por tanto la comunicación es Full Duplex, es decir, el maestro puede enviar y recibir datos simultáneamente.

Otra característica de SPI es que es bus síncrono. El dispositivo maestro proporciona una señal de reloj, que mantiene a todos los dispositivos sincronizados. Esto reduce la complejidad del sistema frente a los sistemas asíncronos.

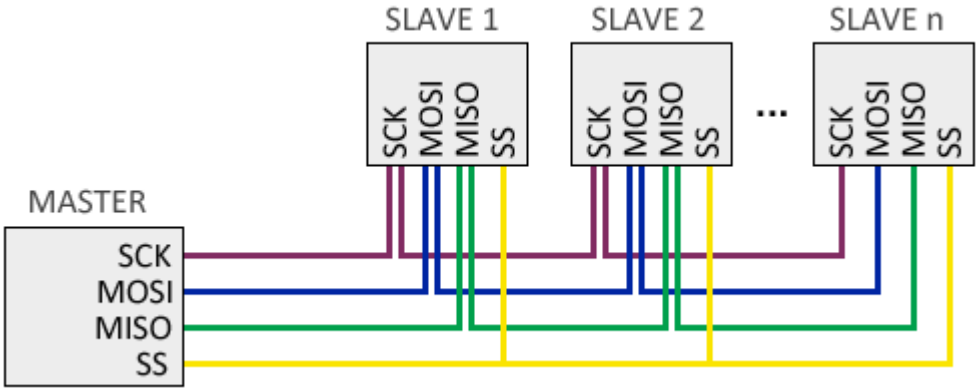
Por tanto, el bus SPI requiere un **mínimo de 3 líneas**.

- MOSI (Master-out, slave-in) para la comunicación del maestro al esclavo.
- MISO (Master-in, slave-out) para comunicación del esclavo al maestro.
- SCK (Clock) señal de reloj enviada por el maestro.

Además, se requiere **una línea adicional SS (Slave Select)** para cada dispositivo esclavo conectado, para seleccionar el dispositivo con el que se va a realizar la comunicación.



Sin embargo, esto tiene la desventaja de requerir una línea por cada dispositivo esclavo. En caso de disponer muchos dispositivos esclavos esto puede no ser práctico, por lo que es posible adoptar una conexión en cascada, donde cada esclavo trasmite datos al siguiente.



Por contra, en esta configuración la información debe llegar a todos los esclavos para que la comunicación sea finalizada por lo que, en general, la velocidad de respuesta del bus es menor.

funcionamiento del BUS SPI

El funcionamiento del bus SPI es sencillo.

Por defecto el maestro mantiene en estado HIGH todas las líneas SS. Cuando el maestro quiere establecer comunicación con esclavo pone a LOW la línea SS correspondiente, lo que indica al esclavo que debe iniciar la comunicación.

En cada pulso de la señal de reloj, normalmente en el flanco de subida, el dispositivo maestro envía un bit del esclavo y a la vez que recibe un bit del esclavo seleccionado.

La trama (los datos enviados) no sigue ninguna regla, es decir, podemos enviar cualquier secuencia arbitraria de bits. Esto hace que los dispositivos conectados necesiten tener pre-acordado la longitud y significado de los que van a enviar y recibir.

La electrónica requerida para implementar el bus SPI es sencilla y barata, incluso un único registro de desplazamiento puede ser suficiente. Además, como la señal de reloj es proporcionada por el maestro, los esclavos ni siquiera necesitan disponer de un reloj propi

Ventajas

- Alta velocidad de trasmisión (hasta 80 Mhz en ESP32) y Full Duplex
- Los dispositivos necesarios son sencillos y baratos, lo que hace que esté integrado en muchos dispositivos.
- Puede mandar secuencias de bit de cualquier tamaño, sin dividir y sin interrupciones.

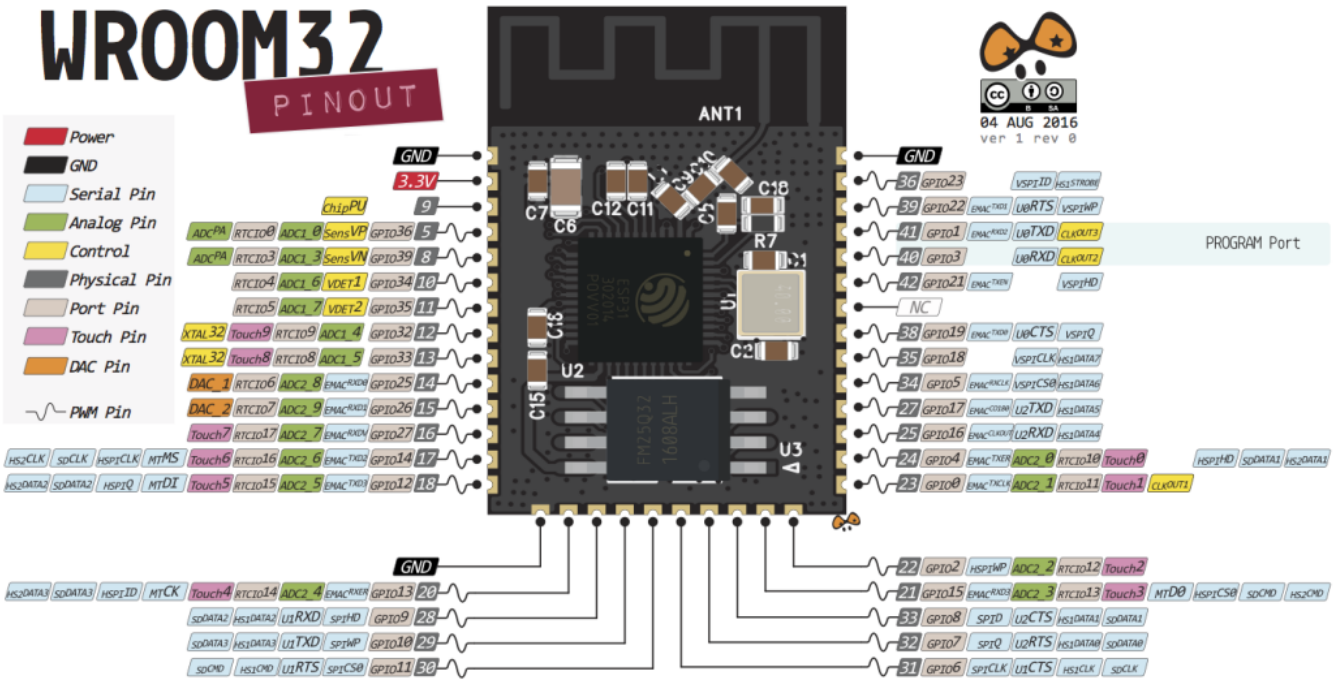
desventajas

- Se requiere 3 cables (SCK, MOSI y MISO) + 1 cable adicional (SS) por cada dispositivo esclavo.

- Solo es adecuado a corta distancias (unos 30cm)s
- No se dispone de ningún mecanismo de control, es decir, no podemos saber si el mensaje ha sido recibido y menos si ha sido recibido correctamente.
- La longitud de los mensajes enviados y recibidos tiene que ser conocida por ambos dispositivos.

BUS SPI en ESP32

ESP32 tiene 3 buses spi ; uno de ellos se utiliza para la carga del programa y no es utilizable y tre de los restantes son utilizables



Flash SPI integrado en el ESP-WROOM-32 GPIO 6 a GPIO 11 están expuestos en algunas placas de desarrollo ESP32. Sin embargo, estos pines están conectados al flash SPI integrado en el chip ESP-WROOM-32 y no se recomiendan para otros usos. Por lo tanto, no utilice estos pines en sus proyectos:

GPIO 6 (SCK/CLK) GPIO 7 (SDO/SD0) GPIO 8 (SDI/SD1) GPIO 9 (SHD/SD2) GPIO 10 (SWP/SD3) GPIO 11 (CSC/CMD)

SPI la conexion por defecto es la que sigue :

SPI	MOSI	MISO	CLK	CS
VSPI	GPIO 23	GPIO 19	GPIO 18	GPIO 5
HSPI	GPIO 13	GPIO 12	GPIO 14	GPIO 15

Configuración de SPI para habilitar la comunicación SPI

Si queremos iniciar una comunicación, primero tenemos que habilitar SPI con el siguiente código Arduino SPI.beginTransaction (SPISettings (8000000, MSBFIRST, SPI_MODE0)); Verá que al comenzar una comunicación SPI hay un total de 3 configuraciones que se pueden configurar si queremos controlar la configuración manualmente. Esta configuración de SPI no se elimina cuando la comunicación SPI está deshabilitada con SPI.endTransaction () . En su lugar, puede cambiar la configuración de SPI sobrescribiendo la configuración a través de la función SPISettings () . Siempre existe la opción de permitir que el microcontrolador establezca la configuración de SPI por defecto

Ajuste de frecuencia de reloj para SPI

La primera configuración es la frecuencia de reloj SPI, que se establece en 8 Mbits / s en los ejemplos.(por defecto en ESP32)

Configuración de cambio de datos para SPI

El segundo ajuste es el desplazamiento de datos que define qué bit se transfiere primero. Hay 2 opciones:

- Bit más significativo (MSB) → MSBFIRST: El bit 8 es el primer bit que se transfiere a través de SPI
- Último bit significativo (LSB) → LSBFIRST: El bit 1 es el primer bit que se transfiere a través de SPI

Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	0	1	0	1	1
MSB	-	-	-	-	-	-	LSB

La mayoría de los chips SPI utilizan el primer orden de datos MSB.

Modos de transmisión para SPI

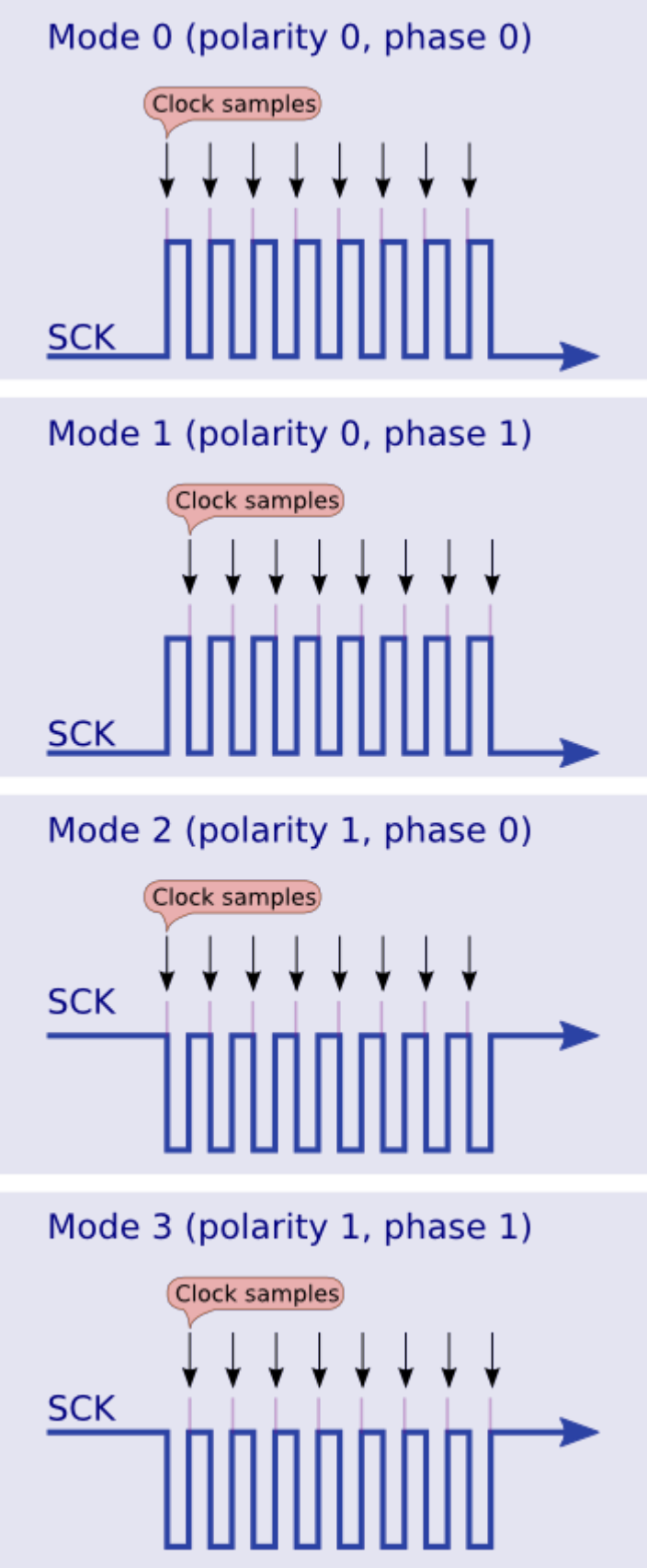
Hay en total 4 modos de transmisión diferentes dependiendo de la combinación de 2 ajustes de transmisión:

- Fase de reloj (CPHA)
- CPHA = 1: Muestras del flanco ascendente del pulso de reloj
- CPHA = 0: Muestras del flanco descendente del pulso de reloj
- Polaridad del reloj (CPOL)

- Reloj inactivo cuando está alto (CPOL = 1): Cada ciclo consta de un pulso de 0. El borde anterior es un borde descendente y el borde posterior es un borde ascendente.
- Reloj inactivo cuando está bajo (CPOL = 0): Cada ciclo consta de un pulso de 1. El borde anterior es un borde ascendente y el borde posterior es un borde descendente.

Modo	Polaridad del reloj (CPOL)	Fase de reloj (CPHA)	Borde de salida	Captura de datos
SPI_MODE0	0	0	Descendente	Creciente
SPI_MODE1	0	1	Creciente	Descendente
SPI_MODE2	1	0	Creciente	Descendente
SPI_MODE3	1	1	Descendente	Creciente

Para la mayoría de los dispositivos, SPI_MODE0 es el modo de transmisión predeterminado. Las siguientes imágenes muestran los cuatro modos diferentes. Ves la línea SCK y cuando el reloj muestra.

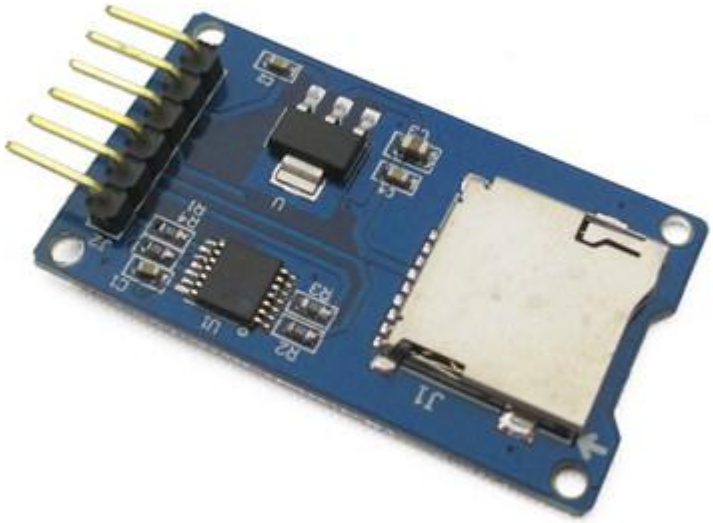


EJERCICIOS PRACTICOS

Existen muchos dispositivos que tienen instalado el SPI para su control ; si bien en estas practicas vamos a poner enfasis en aquellos dispositivos que a traves de comentarios de los alumnos son de facil obtencion .

Utilizaremos una lectura de SD

hardware : https://www.amazon.es/gp/product/B06XHJTGGC/ref=ppx_yo_dt_b_asin_title_o08_s00?ie=UTF8&psc=1



Utilizaremos una lectura RFID

hardware : https://www.amazon.es/BUYGOO-Keychain-Module-Reader-Arduino/dp/B07D9C82W8/ref=sr_1_7?adgrpid=64168290956&dchild=1&gclid=Cj0KCQjw9_mDBhCGARIsAN3PaFM9-kfIgN51lwJEzMFffK_oXlGh0EMOYz9he8RiqdjbXX8q_oYJGW0aAobkEALw_wcB&hvadid=320777693966&hvdev=c&hvlocphy=1005433&hvnetw=g&hvqmt=e&hvrand=2288607395559069836&hvtargid=kwd-302776364356&hydadcr=11857_1752977&keywords=rc522+rfid+module&qid=1618925460&sr=8-7

Ejercicio Practico 1 LECTURA/ESCRITURA DE MEMORIA SD

```
#include <SPI.h>
#include <SD.h>

File myFile;

void setup()
{
  Serial.begin(9600);
  Serial.print("Iniciando SD ...");
  if (!SD.begin(4)) {
    Serial.println("No se pudo inicializar");
    return;
  }
  Serial.println("inicializacion exitosa");
```

```
  myFile = SD.open("archivo.txt");//abrimos  el archivo
  if (myFile) {
    Serial.println("archivo.txt:");
    while (myFile.available()) {
        Serial.write(myFile.read());
    }
    myFile.close(); //cerramos el archivo
  } else {
    Serial.println("Error al abrir el archivo");
  }
}

void loop()
{

}
```

1. Descibir la salida por el puerto serie
2. Explicar el funcionamiento

Ejercicio Practico 2 LECTURA DE ETIQUETA RFID

referencia: https://naylampmechatronics.com/blog/22_tutorial-modulo-lector-rfid-rc522.html

```
#include <SPI.h>
#include <MFRC522.h>

#define RST_PIN 9    //Pin 9 para el reset del RC522
#define SS_PIN  10   //Pin 10 para el SS (SDA) del RC522
MFRC522 mfrc522(SS_PIN, RST_PIN); //Creamos el objeto para el RC522

void setup() {
    Serial.begin(9600); //Iniciamos la comunicación  serial
    SPI.begin();        //Iniciamos el Bus SPI
    mfrc522.PCD_Init(); // Iniciamos  el MFRC522
    Serial.println("Lectura del UID");
}

void loop() {
    // Revisamos si hay nuevas tarjetas  presentes
    if ( mfrc522.PICC_IsNewCardPresent())
        {
        //Seleccionamos una tarjeta
            if ( mfrc522.PICC_ReadCardSerial())
            {
                  // Enviamos serialemente su UID
                  Serial.print("Card UID:");
                  for (byte i = 0; i < mfrc522.uid.size; i++) {
                          Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
                          Serial.print(mfrc522.uid.uidByte[i], HEX);
                  }
                  Serial.println();
                  // Terminamos la lectura de la tarjeta  actual
                  mfrc522.PICC_HaltA();
            }
    }
```

```
}
```

1. Descibir la salida por el puerto serie
2. Explicar el funcionamiento

Ejercicio Practico 3

A realizar como ejercicio en casa

Para la realizacion de este ejercicio pueden utilizar cualquier elemento spi que tengan disponible .

Necesario presentar en el informe y subirlo al github

1. fotos del montaje
2. salidas de depuracion (print...)
3. codigo generado
4. explicación del codigo

Ejercicio de subida de nota (muy valorado)

- Parte 1.- Realizar utilizando el sd y el lector rfid escribiendo en un fichero.log la hora y codigo de cada lectura (describir como se resuelve el hardware para utilizar un spi para dos perifericos) .
- Parte 2.- Generar una pagina web donde se pueda ver la lectura del lector rfid