¿Cuáles son los protocolos serie más usados?, nombre 3 y descríbalos.

Existen varios protocolos de comunicación en serie ampliamente utilizados en la industria electrónica. A continuación, se describen tres de los protocolos de comunicación en serie más comunes:

1. RS-232: es uno de los protocolos de comunicación en serie más antiguos y populares. A pesar de las innovaciones en comunicación serie que han introducido otros protocolos como USB, WiFi y Ethernet, la comunicación RS232 aún es muy usada.Hay varias razones para esa longevidad del protocolo RS232. Una de ellas es que tiene mejor resistencia al ruido de línea que otros protocolos. El protocolo de comunicación RS232 también es mejor para transmitir señales a distancias más largas que las señales generadas por dispositivos I2C o TTL. También es compatible como estándar con muchos ordenadores y fabricantes de periférico de hardware. Se utiliza principalmente para la comunicación entre dispositivos electrónicos, como módems, impresoras, escáneres y dispositivos de red. RS-232 utiliza señales eléctricas para la transmisión de datos y define una serie de pines para la conexión de dispositivos.

Este diagrama muestra la conexión entre un DTE RS232 (ordenador) y un DCE RS232 (módem). En el ejemplo, el DTE envía datos binarios "11011101" al DCE y el DCE transmite la secuencia binaria "11010101" al dispositivo DTE.

RS232 define los estándares eléctricos, los modos operativos, los niveles de voltaje comunes y la cantidad de bits que se transfieren entre un DTE y un DCE. Es el protocolo de transmisión estándar utilizado en líneas telefónicas terrestres.

Estándares de RS232

Las especificaciones eléctricas de la interfaz RS232 se definieron en 1969. Describen los voltajes eléctricos, la velocidad en baudios, los modos operativos, la impedancia de línea y la velocidad de respuesta utilizada por el protocolo.

Niveles de Voltaje RS232

Los voltajes de línea del RS232 varían de -25V a +25V. Definidos como voltajes de señal y voltajes de contro

Signal Voltage Levels	Logical State
−3 to −25	OFF (0)
+3 to +25	ON (1)

Control Signal Voltage Levels (volts)	Logical State
−3 to −25	OFF (0)
+3 to +25	ON (1)

Un voltaje de señal entre -3V y -25V representa un '1' lógico y los voltajes entre +3V a +25V representan un '0' lógico. Las señales de voltaje de control usan la lógica negativa donde el "1" lógico indica -3V a -25V y un "0" lógico indica +3V a +25V. Una lectura de voltaje entre -3V y +3V se considera indeterminada.

Velocidad en Baudios

La velocidad en baudios describe el número de bits binarios que se transmiten por segundo. En el protocolo RS232, se admiten velocidades de transmisión de 110 a 230400. Las velocidades en baudios 1200, 4800, 9600 y 115200 son las más usadas. La velocidad en baudios determina la velocidad a la que se produce la transmisión y debe ser la misma en ambos lados de la comunicación.

Modo de Operación

Los dispositivos RS232 utilizan señalización de un solo extremo (dos hilos) para realizar la transmisión de los datos. En este tipo de cableado, un cable está conectado a tierra mientras que el otro se usa para transmitir un voltaje variable. Pueden ser afectados por el ruido producido por las diferencias en el voltaje de tierra de los circuitos del controlador y el receptor. Una ventaja del método de un solo extremo es que se necesitan menos cables la comunicación.

Impedancia de Línea

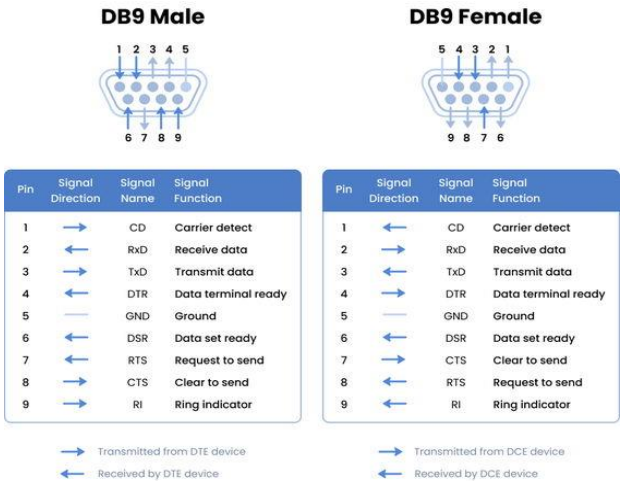
La variación de impedancia entre el controlador RS232 y el receptor está en el rango de 3KΩ a 7KΩ y está destinado a maximizar la transferencia de voltaje entre los dispositivos.

Velocidad de Subida

La velocidad a la que responde el controlador RS232 se conoce como velocidad de respuesta. Está determinado por los cambios de voltaje de entrada registrados por el controlador. El protocolo RS232 define una velocidad de respuesta mínima con tiempos de subida y bajada lentos. Esto está diseñado para minimizar la interferencia entre señales adyacentes. La velocidad de respuesta máxima normal permitida es de 30V/µs

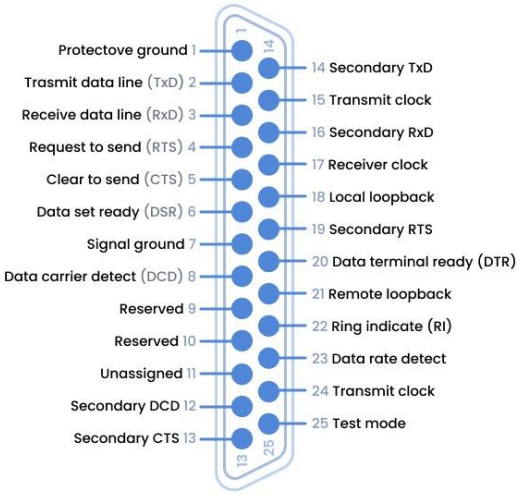
Pines RS232

La comunicación entre el DTE y el DCE mediante el protocolo RS232 emplea conectores DB9 o DB25. Ambos tipos de conectores D-sub tienen terminales macho y hembra. Los conectores DB25 emplean 25 pines y los DB9 usan 9 con cada uno de los pines en un pin del RS232 que tiene una función específica. Los diagramas de pines de DB9 y los de DB25 se pueden ver a continuación.



La interfaz serie RS232 tiene nueve pines y se puede obtener en modelos de tipo macho o hembra.

La interfaz de comunicación serie RS232C es una versión actualizada del RS232. Conserva todas las características del estándar original, pero emplea 25 pines. De los 25 o 9 pines de un conector, solo tres se usan para conectar los dispositivos terminales.



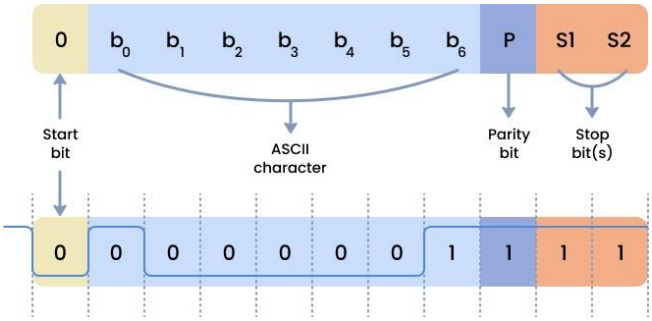
Descripcion Funcional

Además de especificar las características eléctricas, el protocolo RS232 define las funciones de cada señal utilizada en la interfaz. Estos incluyen señales de control y temporización, tierra común y señales de datos. Aquí tiene una tabla de las señales y funciones de los pines RS232.

Signal name	Function
Protective ground	The signal is connected to chassis ground of metallic connector
Common ground	Zero reference voltage level for all the control signals
TxD (Transmit Pin)	To transmit data from DTE to DCE
RxD (Receive Pin)	Sends data from DCE to DTE
DTR (Data terminal Ready)	DTE is ready to accept request
DCD (Data carrier Detect)	DCE accepts a carrier from DTE located at remote location
DSR (Data set ready)	DCE is prepared to send and receive the information
RI (Ring indicator)	Detects the incoming ring tone on the telephone line.
RTS (Request to send)	DTE call for DCE to send the data
RTR (Ready to receive)	DTE is geared up to receive data coming from DCE
CTS (Clear to send)	DCE is in a ready state to accept data coming from DTE

RS232 también proporciona señales secundarias que complementan las señales primarias descritas anteriormente. Estos incluyen señales secundarias TxD, TxD, DTE, RTS y DCD que se pueden utilizar para configurar las conexiones entre equipos DCE y DTE.

Formato de Protocolo

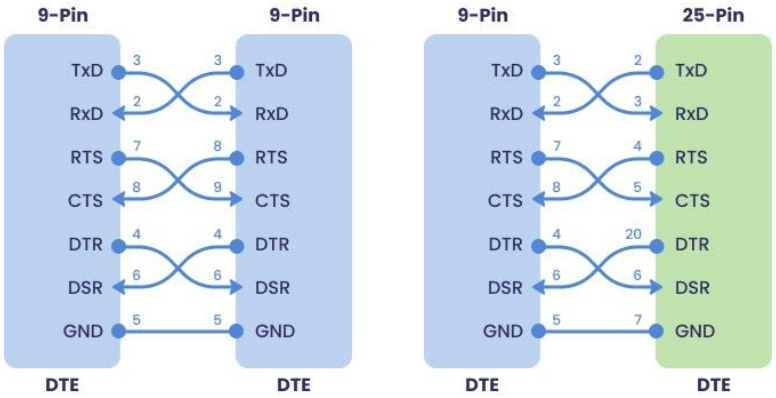


Un mensaje entregado a través del protocolo RS232 comienza enviando un bit de inicio "0". Seguido por siete bits de datos ASCII con un bit de paridad agregado para realizar la verificación. Los bits de paridad determinan la validez del mensaje. La transmisión finaliza con un bit de parada binario "1". Generalmente se envían uno o dos bits de parada.

En el diagrama anterior, el carácter ASCII "A" se transmite con un flujo binario en serie de "1" y "0". Hay un retraso predeterminado entre la transmisión de cada bit cuando la línea se considera inactiva.

Protocolo de enlace RS232

El proceso de intercambiar señales de información entre un emisor y un receptor se conoce como protocolo de enlace. Un enlace de comunicación se construye entre el transmisor y el receptor mediante estas señales. Hay dos tipos de interconexiones RS232: protocolo de enlace de hardware y software.



Los conectores DB9 y DB25 son utilizados en el protocolo de enlace RS232. Solo los pines TxD (Transmisor) y RxD están cruzados si no se implementa el protocolo de enlace. Los otros pines, como RTS, CTS, DSR y DTR, están conectados en bucle.

Cuando el protocolo de enlace es utilizado, las señales RTS y CTS se cruzan al igual que los pines DTR y DSR.

Ventajas

Las múltiples ventajas del protocolo RS232 lo han convertido en un estándar muy utilizado en la comunicación serie. Entre ellos:

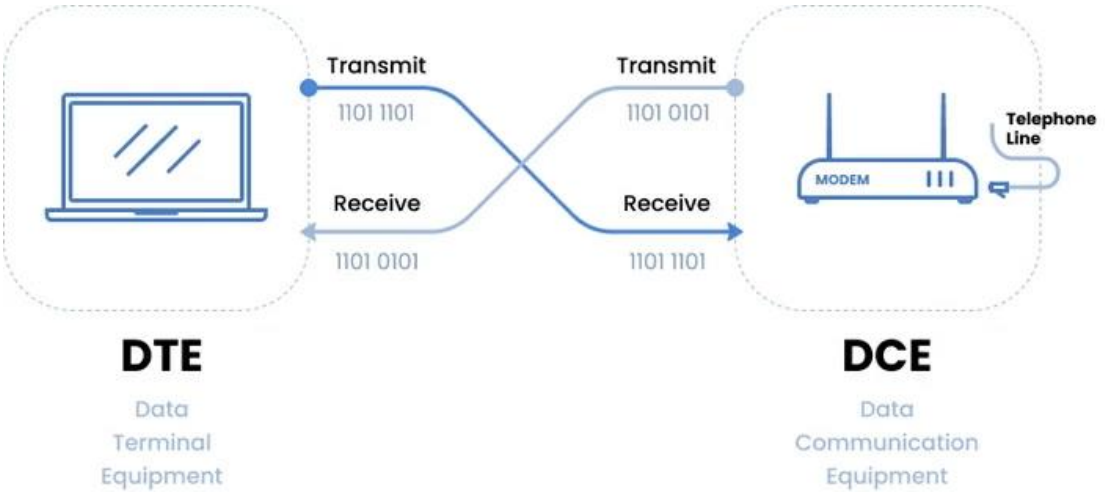
- El diseño simple del protocolo.
- Reducción de la sobrecarga de hardware en comparación con la comunicación en paralelo.
- Compatibilidad con la comunicación DTE y DCE.
- Un protocolo económico para desarrollo.
- Muy recomendable para aplicaciones de corta distancia.

Desventajas

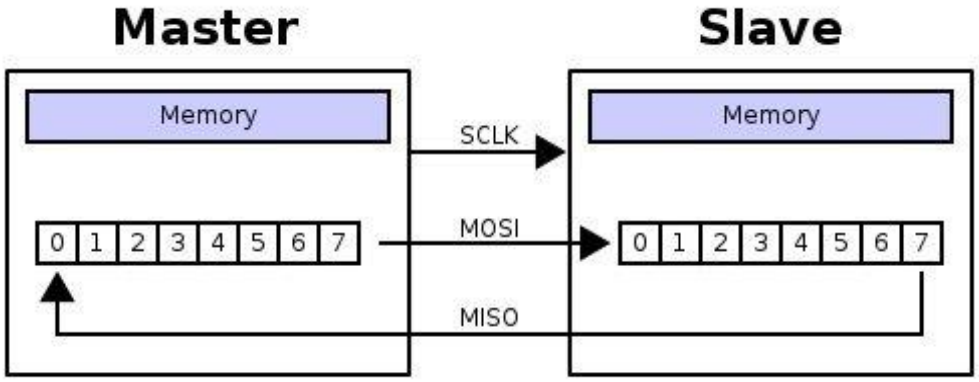
El protocolo RS232 tiene ciertas desventajas y limitaciones. No puede utilizar la comunicación full-duplex con el estándar. Al ser un protocolo de un solo extremo puede cambiar el potencial de tierra. RS232 no se recomienda para la comunicación a larga distancia, ya que los cables más largos producen interferencias que afectan a la transmisión en serie.

Ejemplos de su uso:

- Comunicación entre un ordenador y una impresora.
- Comunicación entre una caja registradora y un lector de código de barras.
- Conexión de un módem para acceder a Internet.



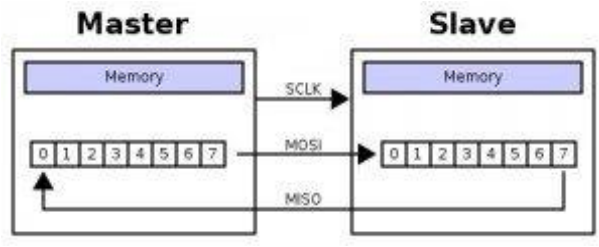
2. SPI (Interfaz de Periféricos Seriales): es un protocolo de comunicación en serie utilizado para la comunicación entre microcontroladores y dispositivos periféricos, como sensores y pantallas. SPI es una forma rápida de transmitir datos y se utiliza comúnmente en aplicaciones que requieren una alta velocidad de transmisión de datos.



Estructura general del protocolo SPI.

SPI es un acrónimo para referirse al protocolo de comunicación serial Serial Peripherical Interface. Este protocolo nace casi a principios de 1980 cuando Motorola lo comienza a introducir y desarrollar en el primer microcontrolador derivado de la misma arquitectura del microcontrolador 680000. SPI se ha convertido es uno de los más populares protocolos para trabajar con comunicación serial debido a su velocidad de transmisión, simplicidad, funcionamiento y también gracias a que muchos dispositivos en el mercado como pantallas LCD, sensores, microcontroladores pueden trabajar con el.

El SPI es un protocolo síncrono que trabaja en modo full duplex para recibir y transmitir información, permitiendo que dos dispositivos pueden comunicarse entre sí al mismo tiempo utilizando canales diferentes o líneas diferentes en el mismo cable. Al ser un protocolo síncrono el sistema cuenta con una línea adicional a la de datos encarga de llevar el proceso de sincronismo. Veamos cómo funciona:

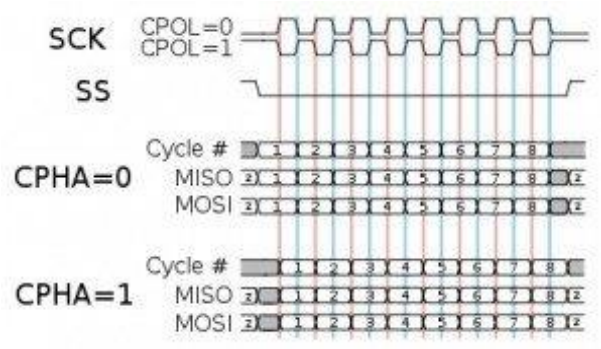


Estructura general del protocolo SPI.

Dentro de este protocolo se define un maestro que será aquel dispositivo encargado de transmitir información a sus esclavos. Los esclavos serán aquellos dispositivos que se encarguen de recibir y enviar información al maestro. El maestro también puede recibir información de sus esclavos, cabe destacar. Para que este proceso se haga realidad es necesario la existencia de dos registros de desplazamiento, uno para el maestro y uno para el esclavo respectivamente. Los registros de desplazamiento se encargan de almacenar los bits de manera paralela para realizar una conversión paralela a serial para la transmisión de información.

Existen cuatro líneas lógicas encargadas de realizar todo el proceso:

- **MOSI (Master Out Slave In):**. Línea utilizada para llevar los bits que provienen del maestro hacia el esclavo.
- **MISO (Master In Slave Out):**. Línea utilizada para llevar los bits que provienen del esclavo hacia el maestro.
- **CLK (Clock):**. Línea proviniente del maestro encarga de enviar la señal de reloj para sincronizar los dispositivos.
- **SS (Slave Select):**. Línea encargada de seleccionar y a su vez, habilitar un esclavo.
- Se presenta una imagen donde se tienen todas estas líneas con sus respectivos registros de desplazamiento y su dirección de flujo:



Distintos modos en SPI.

Existen cuatro modos en el cual se puede enviar información dependiendo de dos párametros basados en la señal de reloj. El primer de ellos es la polaridad y el segundo es la fase. Al tener dos parámetros donde cada uno puede tomar dos estados se tendrá entonces cuatro modos distintos de poder llevar a cabo el proceso de transmisión y envío de información.

- Modo 0: CPOL = 0 y CPHA = 0. Modo en el cual el estado del reloj permanece en estado lógico bajo y la información se envía en cada transición de bajo a alto, es decir alto activo.
- Modo 1: CPOL = 0 y CPHA = 1. Modo en el cual el estado del reloj permanece en estado lógico bajo y la información se envía en cada transición de alto a bajo, es decir bajo activo.
- Modo 2: CPOL = 1 y CPHA = 0. Modo en el cual el estado del reloj permanece en estado lógico alto y la información se envía en cada transición de bajo a alto, es decir alto activo.
- Modo 3: CPOL = 1 y CPHA = 1. Modo en el cual el estado del reloj permanece en estado lógico alto y la información se envía en cada transición de alto a bajo, es decir bajo activo.

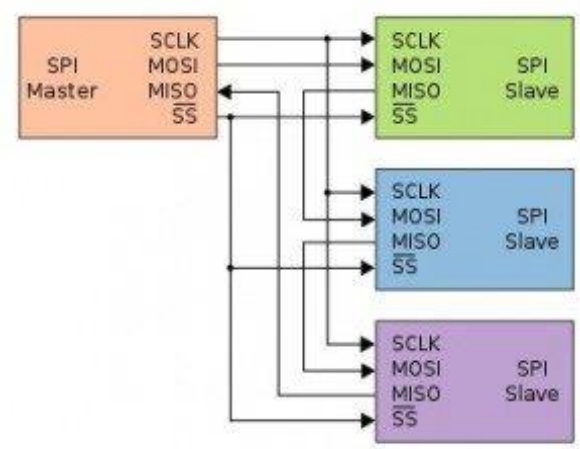
Esta puede puede brindarle una idea más intuitiva de los modos explicados anteriormente.

La configuración de modos es independiente para cada esclavo con esto quiere decir que cada esclavo puede tener una configuración de CPOL y CPHA distinta a la de otros esclavos, inclusive una frecuencia de trabajo distinta y entonces para esto, el maestro deberá adaptarse a la configuración de cada esclavo. Por esta razón es recomendable que el sistema trate de trabajar con los mismo parámetros de ser posible porque sino, será un dolor de cabeza.

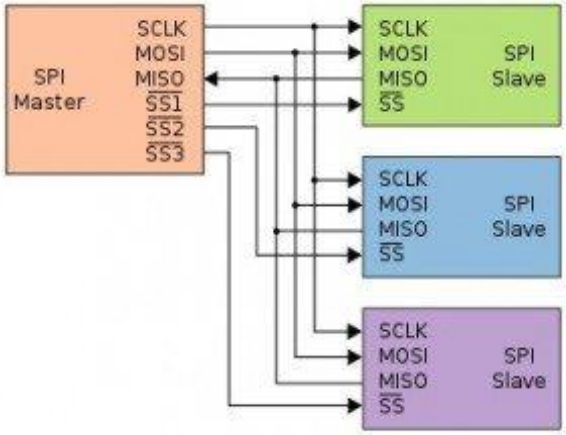
En este protocolo se define únicamente un maestro y varios esclavos. La manera en la cual estos dispositivos se conectan pueden ser de dos tipos: encadenado o paralelo. El de tipo encadenado las entrada del mosi de cada esclavo va conectada con el mosi del master para el primer caso o de su esclavo anterior para el resto. Además, se utiliza un único de selección de esclavo proveniente del maestro en forma paralela hacia cada esclavo.Por otro lado, en el tipo paralelo se utiliza un único mosi proviente del maestro en forma paralela hacia cada

esclavo. Además, se adiciona una línea de selección de esclavo proveniente del maestro por cada esclavo que exista en el sistema.

Para tener un mejor entendimiento de estas conexiones, se presenta una imagen ilustrativa:



Tipo encadenado.



Tipo paralelo.

La transmisión de información puede darse de muchas maneras dependiendo del fabricante, en muchos casos la línea SS habilita un esclavo cuando ésta se pone en estado lógico cero pero eso puede cambiar. La transimisión de bits se puede dar comenzando con el LSB o con MSB dependiendo también del fabricante, es muchos casos se comienza por el bit más significativo. Un bit es transmitido cada ciclo de reloj.

¿Ventajas? Existe una serie de ventajas que ofrece este protocolo, entre ellas está la velocidad de transmisión ya que es configurable a través de software y dependerá también de los dipositivos utilizados en el sistema. Con respecto a otros protocolos seriales que trabajan a modo half duplex, el SPI tiene velocidades de transmisión mucho mayores debido a que éste trabaja en modo full duplex. Otros parámetros configurables a través de software son la frecuencia del reloj, la configuración de fase (CPHA) y polaridad (CPOL). Si solo existe un esclavo, puede colocarse la línea SS fija si el esclavo lo permite. No se limitan a

trabajar con palabras de ocho bits. Es ampliamente utilizado cuando se necesita comunicar con equipos a distancias cortas.

Ejemplos de su uso:

- Interfaz de comunicación entre un microcontrolador y una pantalla OLED.
- Interfaz de comunicación entre un microcontrolador y un sensor de temperatura.
- Interfaz de comunicación entre un microcontrolador y un módulo de memoria Flash.

3. I2C (Inter-Integrated Circuit): es otro protocolo de comunicación en serie utilizado en sistemas electrónicos. I2C es una forma de comunicación maestro/esclavo y se utiliza para la comunicación entre microcontroladores, sensores, memoria EEPROM y otros dispositivos. I2C utiliza solo dos líneas de señal para la transmisión de datos, lo que lo hace ideal para aplicaciones con limitaciones de espacio o donde se requiere una comunicación de baja velocidad.

I2C es un puerto y protocolo de comunicación serial, define la trama de datos y las conexiones físicas para transferir bits entre 2 dispositivos digitales. El puerto incluye dos cables de comunicación, SDA y SCL. Además el protocolo permite conectar hasta 127 dispositivos esclavos con esas dos líneas, con hasta velocidades de 100, 400 y 1000 kbits/s. También es conocido como IIC ó TWI – Two Wire Interface.
El protocolo I2C es uno de los más utilizados para comunicarse con sensores digitales, ya que a diferencia del puerto Serial, su arquitectura permite tener una confirmación de los datos recibidos, dentro de la misma trama, entre otras ventajas.
La conexión de tantos dispositivos al mismo bus, es una de las principales ventajas. Además si comparamos a I2C con otro protocolo serial, como Serial TTL, este incluye más bits en su trama de comunicación que permite enviar mensajes más completos y detallados.
Los mensajes que se envían mediante un puerto I2C, incluye además del byte de información, una dirección tanto del registro como del sensor. Para la información que se envía siempre existe una confirmación de recepción por parte del dispositivo. Por esta razón es bueno diferenciar a los distintos elementos involucrados en este tipo de comunicación .**I2C – Esquema de comunicación y elementos**

4. Siempre que hablamos de una comunicación oral, se entiende que es entre dos o más personas. Como consecuencia podemos también indicar que en una comunicación digital existen distintos dispositivos o elementos. En el caso de **I2C** se diferencian dos elementos básicos, un¿ **MAESTRO** y un **ESCLAVO**. La **Figura-1**, muestra una conexión típica de tres dispositivos, el bus consiste de dos líneas llamadas, **S**erial **DATA** – **SDA** y **S**erial **CL**ock – **SCL**. Es decir, Datos Seriales y Reloj Serial. En particular al bus se le conectan dos resistencias en arreglo **pull-up**, de entre 2.2K y 10K.

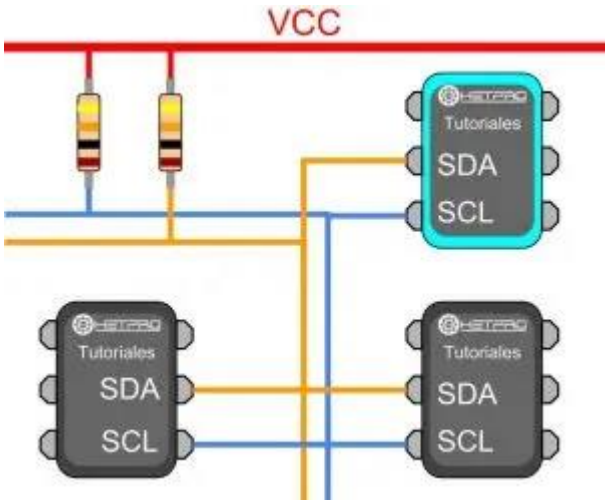


Figura-1. Conexión de tres dispositivos a un bus de comunicación I2C.

El **MAESTRO** I2C se encarga de controlar al cable de reloj, por sus siglas en inglés llamada SCL – Serial CLock. Además el MAESTRO se encarga de iniciar y parar la comunicación. La información binaria serial se envía sólo por la línea o cable de datos seriales, en inglés se llama SDA – Serial DAta. Dos Maestros no pueden hacer uso de un mismo puerto I2C. Puede funcionar de dos maneras, como maestro-transmisor o maestro-receptor. Sus funciones principales son:

Iniciar la comunicación – S
Enviar 7 bits de dirección – ADDR
Generar 1 bit de Lectura ó Escritura – R/W
Enviar 8 bits de dirección de memoria
Transmitir 8 bits de datos –
Confirmar la recepción de datos – ACK – ACKnowledged
Generar confirmación de No-recepción, NACK – No-ACKnowledged
Finalizar la comunicación

El **ESCLAVO** I2C, generalmente suele ser un sensor. Este elemento suministra de la información de interés al MAESTRO. Puede actuar de dos formas: esclavo-transmisor ó esclavo-receptor. Un dispositivo I2C esclavo, no puede generar a la señal SCL. Sus funciones principales son:

Enviar información en paquetes de 8 bits.
Enviar confirmaciones de recepción, llamadas ACK

I2C – Bits de la trama del puerto

El protocolo de comunicación I2C se refiere al conjunto de bits que son necesarios para enviar uno o varios bytes de información. En lo particular, para este protocolo existen los siguientes bits importantes:

- Inicio ó Start – S
- Parada – P
- Confirmación – ACK
- NoConfirmación – NACK

- Lectura-/Escritura – L/W
- 7 bits para la dirección del dispositivo esclavo/maestro
- 8 bits de dirección (para algunos sensores pueden ser 16 bits)
- 8 bits de datos

El conjunto de estos bits y su orden va formando distintas tramas de comunicación. Existen distintos modos de comunicación dependiendo del arreglo de estos bits. Tanto el maestro como el esclavo pueden o no generar los bits anteriores, según los modos de comunicación. El puerto I2C esta disponible si las dos líneas, SDA y SCL están en un nivel lógico alto.

I2C – modos de comunicación

Los modos de comunicación en I2C se refieren a las distintas tramas que pueden formarse en el bus. Estas tramas o modos dependen de por ejemplo, si queremos leer al sensor esclavo, o si lo queremos configurar. Existen principalmente dos modos de comunicación:

- Maestro-Transmisor y Esclavo-Receptor. Este modo se usa cuando se desea configurar un registro del esclavo I2C.

- Maestro-Receptor Y Esclavo-Transmisor. Se usa cuando queremos leer información del sensor I2C.

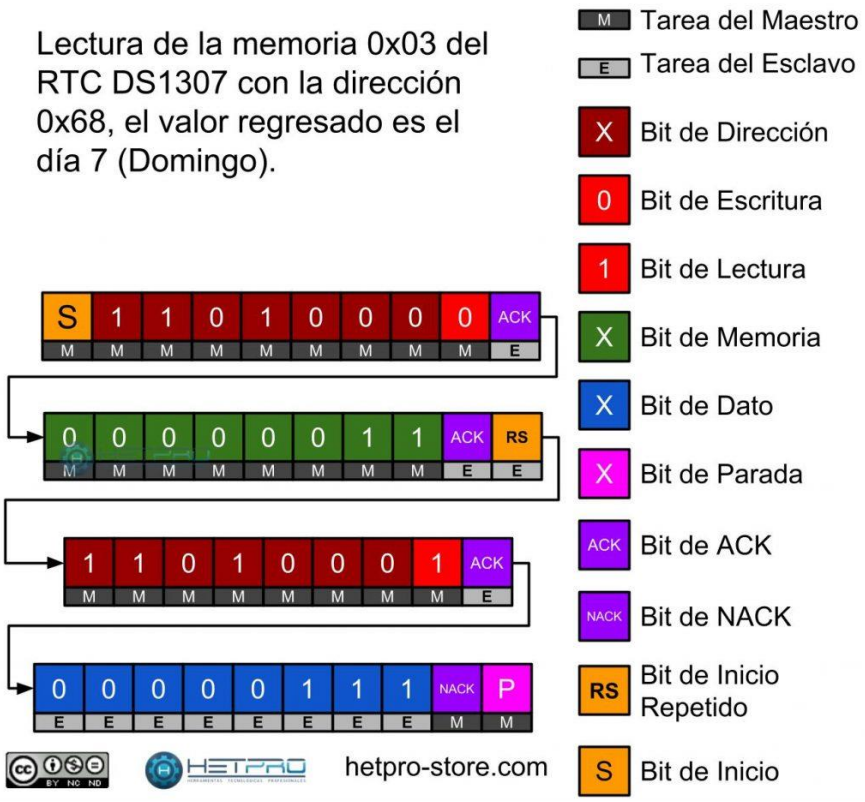
Ejemplo-1 configuración de 1-registro

Ejemplo-1. Para diseñar registrador de datos, es necesario en algunas circunstancias usar un reloj de tiempo Real, RTC. Un **RTC** común es el **DS1307** que tiene un puerto para poder leer y configurar el tiempo. La Figura-2, muestra la trama de datos necesaria para configurar la hora para que sean las 08 hrs. Para poder encontrar el sensor se requiere conocer su dirección. Para determinar la dirección se tienen 7 bits (2^7 = 128), de aquí que se puedan comunicar con dos líneas hasta 127 sensores, la dirección 0 es una llamada general. Después se requiere un bit de lectura o escritura. Entonces, este bit acompaña a los 7-bits de dirección. Si el bit de lectura/escritura vale = 0, significa que se escribirá al esclavo. Si este bit vale = 1, significa entonces que se leerá. Cuando el Esclavo I2C recibe su dirección, es cuando esté, responde con una confirmación (ACK). El esclavo queda en espera de 8 bits de la memoria que se quiere escribir y responde con un ACK. Posteriormente el esclavo espera a que el maestro le envié los 8-bits de datos correspondientes a la configuración del registro anterior y responde con un ACK. Finalmente el Maestro-I2C responde con un bit de fin de comunicación.

Figura-2. Trama de bits que se colocan en el puerto I2C para configurar el registro 0x02 del RTC-DS1307, con el valor de 0x08.

Ejemplo-2. Lectura del día RTC-DS1307 I2C

La Figura-3, muestra la trama que se requiere escribir en el puerto para leer un byte del registro cuya dirección es 0x03. Este registro guarda el día del RTC-DS1307 cuya dirección para el puerto es 0x68. Para poder leer el byte, primero es necesario indicar al dispositivo esclavo que se quiere LEER la dirección 0x03. Una vez que el esclavo responde con el ACK, el Maestro entonces enviará un bit de Inicio-repetido seguido de la dirección pero con bit de Lectura. Después, el esclavo responde con el byte que contiene la memoria 0x03 y seguirá respondiendo con la información de la siguiente memoria (0x04) hasta que el MAESTRO dé por terminado la comunicación, esta vez con un NACK antes del bit de Paro.



Trama de lectura I2C – DS1307

Figura-3. Trama de datos para el protocolo I2C para leer el registro 0x03 del DS1307. El RTC retorna el valor 0x07 que corresponde al día 7, es decir, Domingo.

Velocidad del puerto I2C

La velocidad del puerto I2C se refiere al tiempo que le toma al puerto transferir un bit de información. Entonces, este valor se mide en bits/segundo. Típicamente vamos a encontrar la referencia en ciclos/segundo o Herz. Existen tres velocidades estándar, 100Khz, 400Khz y 1Mhz, es decir, 100kbits/s, 400kbits/s y 1000kbits/s. Por ejemplo, para la trama de la Figura-2, ese paquete de datos tiene 29 bits, por lo que a una velocidad de 100kbits/s le tomaría al puerto 0.29 ms enviar la información.
El dato digital ó lógica que leerá cada uno de los dispositivos, corresponde el voltaje en los flancos de subida de la señal de reloj – SCL.

Condiciones eléctricas

Cada uno de los bits antes descritos (**Inicio-S**, **ACK**, **NACK**, Parada, **RS**) significan condiciones eléctricas en el bus I2C. Las condiciones de voltaje son las siguientes:

Inicio. La condición de inicio se genera cuando el bus esta disponible, cuando mientras la línea de SCL esta en alto (1), existe un flanco de bajada (un cambio de estado lógico de alto a bajo), en la línea de SDA). Este bit sólo lo puede generar el MAESTRO.

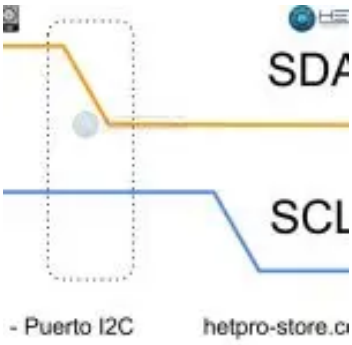


Figura-4. Condición de INICIO-I2C. Estando el bus disponible, hay un flanco de bajada en la línea SDA.

Paro. La señal o bit de PARO se genera cuando hay un flanco de subida en la línea de datos, mientras que la línea de reloj se encuentra en alto. Está condición sólo es posible generarla desde el MAESTRO.

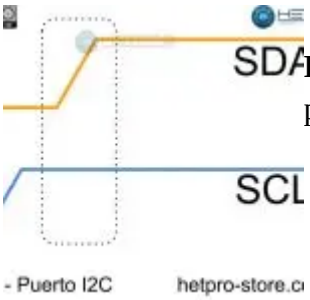


Figura-5. Condición de paro en el puerto I2C.

ACK. Confirmación de recepción. Está condición se crea cuando estando la señal SCL en alto, SDA est en bajo. Está señal la puede generar tanto el MAESTRO como el ESCLAVO.

Figura-6. Condición o bit de ACK. Este bit se usa para confirmar la recepción.

NACK. Este bit es usado en el esquema de comunicación donde se leen varios bytes de un ESCLAVO en una sola transmisión. Entonces, el bit NACK se usa cuando ya no se quieren recibir más bytes. La condición sólo la puede generar el MAESTRO.

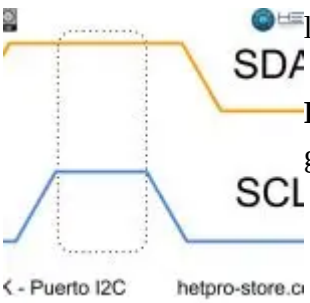


Figura-7. Condición NACK, sólo es generada por el MAESTRO-I2C.

Resumén

El puerto y protocolo I2C es un tema de mucha importancia para los ingenieros que desarrollan sistemas embebidos. Además de poder conectar hasta 127 sensores, el protocolo I2C permite manejar una comunicación segura entre dos dispositivos digitales. Como desventaja podemos mencionar el esquema de comunicación. Debido a que el esquema usado es UN-Maestro MUCHOS-Esclavos, esto puede generar una latencia en el envío y recepción de datos de cada uno de los esclavos. Esto sucede al que sólo podemos usar el bus I2C para comunicar un esclavo a la vez. Sin embargo, el puerto I2C es uno de los más utilizados en aplicaciones embebidas. Podemos encontrar sensores I2C en la mayoría de los dispositivos electrónicos digitales, desde televisores, celulares o laptops. En otro tutorial mencionaré ejemplos del uso de I2C con distintos lenguajes de programación, como Arduino, Mbed, ensamblador y C/C++. Otro puerto similar, es el puerto serial.

Ejemplos de su uso:

- Comunicación entre un microcontrolador y un sensor de presión.
- Interfaz de comunicación entre un microcontrolador y un sensor de luz.
- Comunicación entre un microcontrolador y un circuito integrado de memoria EEPROM.

En resumen, RS-232, SPI e I2C son tres de los protocolos de comunicación en serie más utilizados en la industria electrónica. Cada protocolo tiene sus propias ventajas y se utiliza en diferentes tipos de aplicaciones.