3. One WIRE

El protocolo One-Wire establece el estándar para un tipo de comunicación asincrónica que solo utiliza un pin de datos. Se usa para comunicar microcontroladores con sensores pequeños o memorias. Tiene soporte para poder conectar múltiples dispositivos al mismo bus de datos en donde existe un único maestro, el microcontrolador, y múltiples esclavos, los sensores.

Debido a que no tiene bus dedicado para clock, el dispositivo que tome control del bus tiene que generar pulsos de una determinada duración para poder determinar si es un uno o un cero el bit que se está transmitiendo.

Al tener un único bus, este es bidireccional, calificando como una comunicación half-duplex. El maestro inicia la comunicación controlando el bus de datos, una vez iniciada, el esclavo envía de forma serial la cantidad de bits necesarios de datos.

Dependiendo del sensor usado, el tamaño de la transmisión varía. Normalmente tienen además bytes adicionales que se usan para verificar que no se hayan corrompido los bits en la comunicación. La técnica empleada suele ser checksum o CRC.

2.1 CONEXIONES

Al tener un solo pin de datos, el único pin que debe conectarse al microcontrolador es el de datos. Este bus requiere de una resistencia de pull-up para mantener el estado alto cuando el bus no se utiliza.

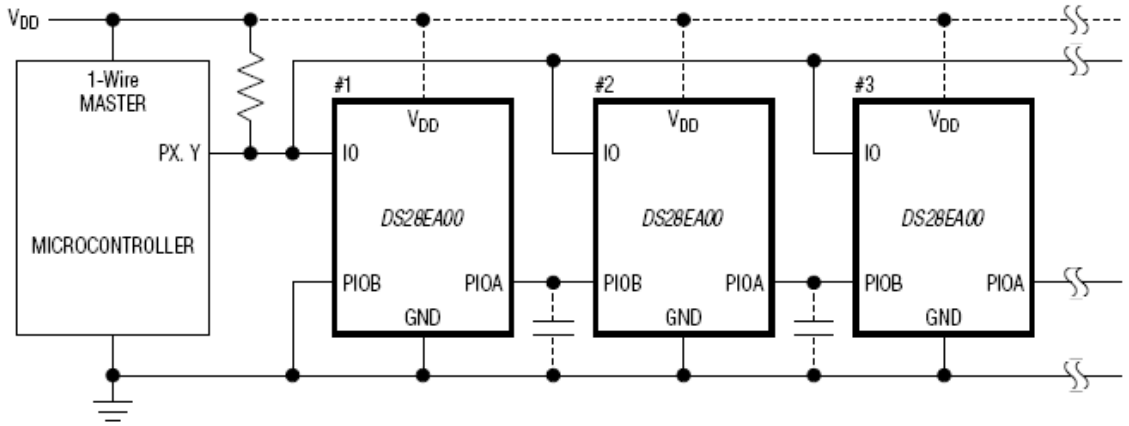

Figura 1: Conexión de protocolo One-Wire

Algunos sensores o dispositivos de muy bajo consumo admiten la posibilidad de alimentarse desde la misma línea de datos (alimentación parásita). Esto es posible debido a un capacitor interno que se carga cada vez que la línea de datos queda en estado alto. Para que este tipo de conexión funcione apropiadamente, la transmisión de datos no debe tener secuencias muy largas de ceros. Este tipo de conexión permite que solamente se requieran dos cables para conectar cada dispositivo: la línea de datos/alimentación y la masa para referencia.

Como alternativa, siempre se pueden conectar de la manera habitual con tres cables, agregando a la conexión anterior su alimentación aparte de la línea de datos.

2.2 TRANSMISIÓN DE DATOS

Como se mencionó anteriormente, solo una línea de datos es necesaria para que este protocolo funcione. Las limitaciones de este son más bien respecto a los tiempos que deben respetarse para cada operación. Como referencia, se usarán datos extraídos de una nota de aplicación de Analog Devices para entender el protocolo, pero cada dispositivo con el que se vaya a comunicar debe ser estudiado previamente para asegurar los tiempos apropiados.

2.2.1 OPERACIONES

Las cuatro operaciones básicas del protocolo son: escribir un uno, escribir un cero, leer y reset. Cada una de ellas tiene un tiempo estricto que debe respetarse.

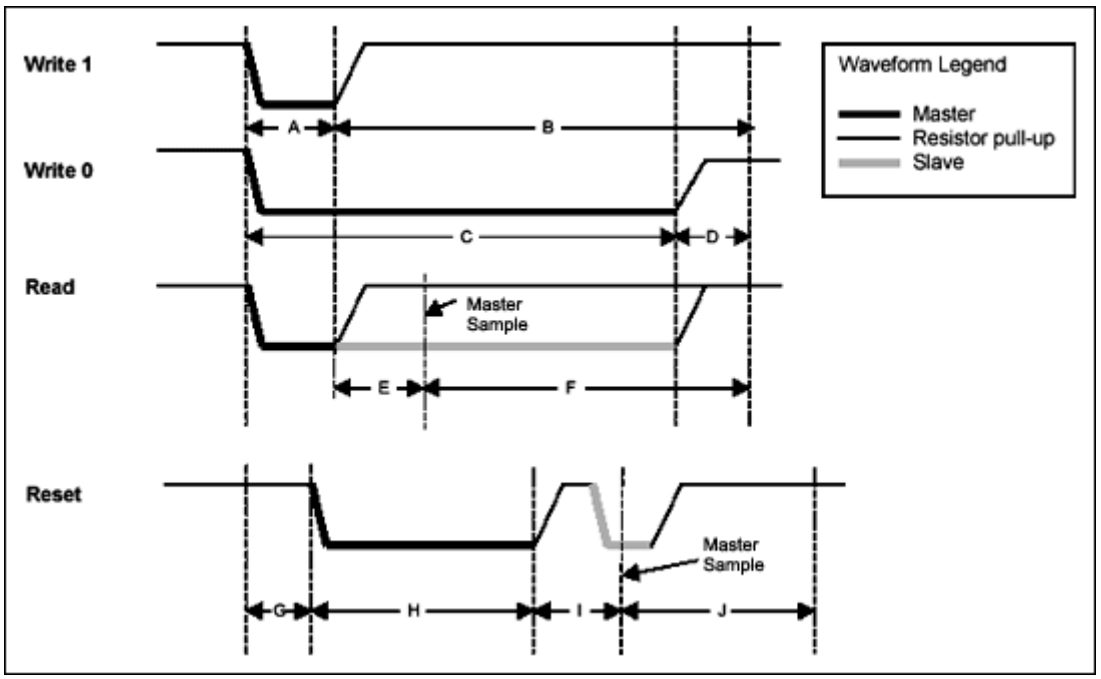

Figura 2: Operaciones de protocolo One-Wire

En la figura de arriba se muestran los distintos tipos de operaciones que soporta el protocolo. Nótese la diferencia en las líneas cuando el responsable del estado del bus es el maestro, la resistencia de pull-up o el dispositivo esclavo.

Cada una de estas operaciones tiene tiempos asociados de control del bus y de delay que se adjuntan en la tabla a continuación. Una tabla más completa se encuentra en la nota de aplicación.

Parámetro	Tiempo recomendado (μs)
A	6
B	64
C	60
D	10

E	9
F	55
G	0
H	480
I	70
J	410

Tabla 1: Tiempos de One-Wire

2.2.2 CPU O PIO

Debido a que no hay clock en este protocolo, el microcontrolador encargado de comunicarse con los dispositivos debe estar muy atento al bus de datos para respetar los tiempos que establece el estándar. Debido a que los tiempos requeridos son muy cortos (del orden de los microsegundos), si el procesador se encarga personalmente de los tiempos del bus, no le deja mucho tiempo para que pueda atender otras tareas del sistema en el que se encuentre. Debido a eso, los sistemas más grandes requieren mucho cuidado cuando se usen este tipo de dispositivos.

Por otro lado, existen microcontroladores con máquinas de estados programables que pueden controlar o leer un GPIO de manera independiente, aliviando al procesador de esa tarea. Estos módulos son llamados PIO (Programmable Input Output). Un microcontrolador con este periférico tendría la capacidad de manejar un bus de One-Wire sin cargar al procesador, permitiendo que este se dedique a todas las otras tareas que deba realizar.

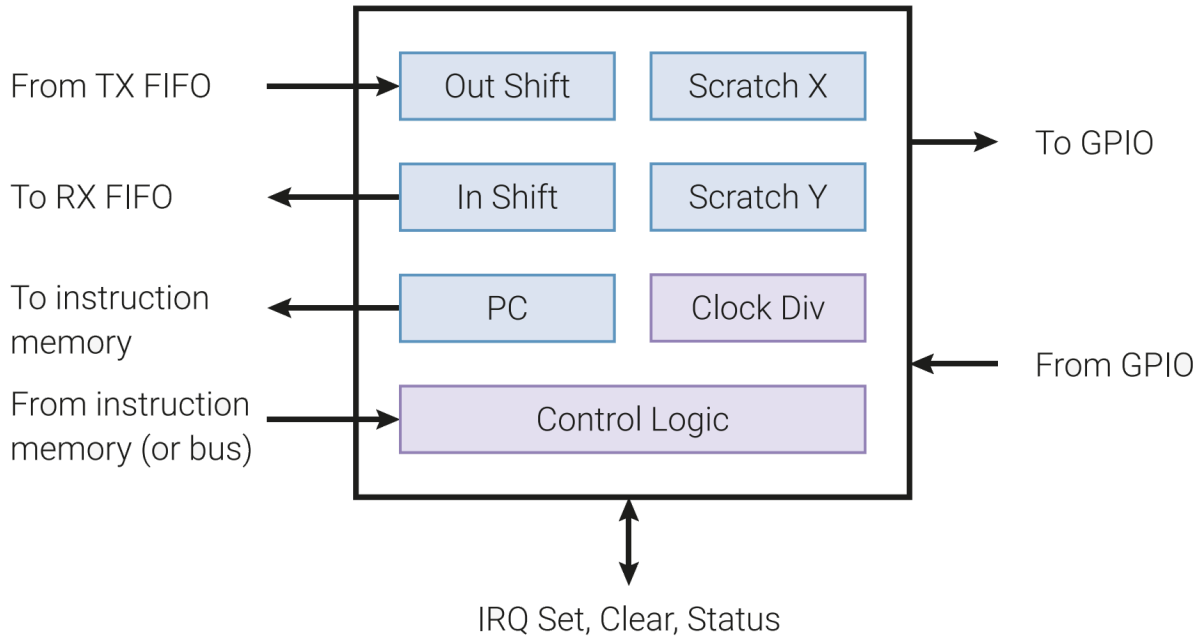

Figura 3: Máquina de estados del RP2040

Arriba se ve una máquina de estados del RP2040, microcontrolador que encontramos en la Raspberry Pi Pico. Cada una de estas tiene registros independientes para utilizar y un registro de desplazamiento de entrada y salida que permiten enviar o leer datos desde o a un FIFO del microcontrolador. De esta forma, la máquina de estado puede ser programada con una serie de instrucciones, ejecutarlas de manera independiente y leer o enviar los datos al procesador cuando estén listos.

Como desventaja, estas máquinas de estado suelen ser programadas en assembler, por lo que requieren cierto conocimiento profundo del hardware y el lenguaje. Su uso en este curso se limita a la implementación directa mediante alguna biblioteca. La programación personalizada de estos PIO queda fuera del alcance de este curso.

2.3 GLOSARIO

A continuación hay una lista de términos usados con sus respectivas definiciones para aclarar cualquier posible confusión.

- **Half-duplex**: comunicación donde un dispositivo puede solo enviar o recibir datos. Tiene línea de datos compartida para ambas operaciones.
- **CRC**: chequeo de redundancia cíclica.
- **PIO**: entrada/salida programable.

2.4 REFERENCIAS ADICIONALES

A continuación se deja material adicional para el que desee profundizar más en el tema que se describió.

- What is the 1-Wire protocol?
- 1-Wire Communication Through Software
- Overview of 1-Wire Technology and Its Use