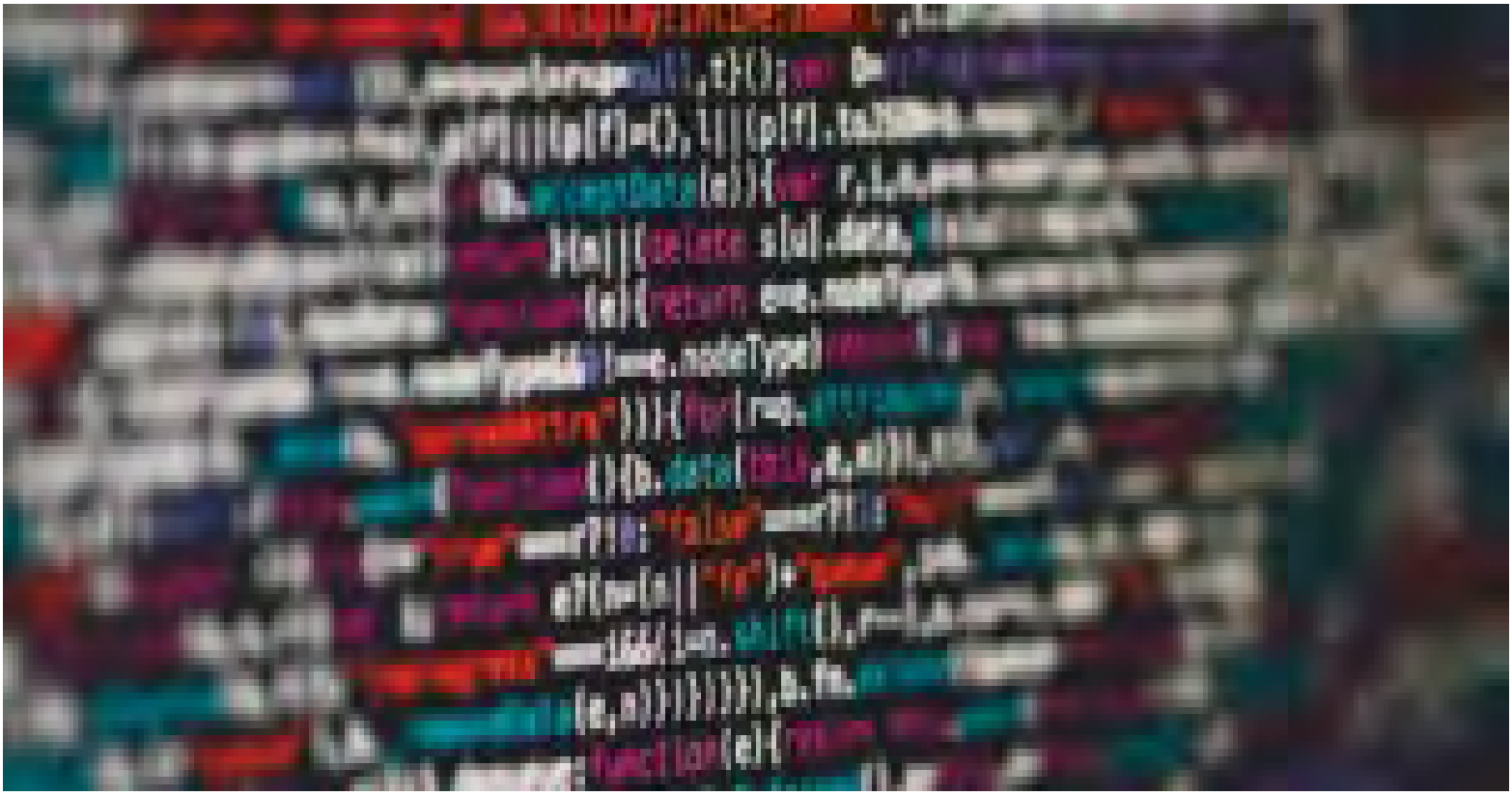
TOTAL PHASE

Understanding the Difference Between UART vs SPI

In this piece, we will explain what UART and SPI are, explain the differences between UART vs SPI, discuss some common applications that use UART and SPI.



What is UART?

UART stands for Universal Asynchronous Receiver Transmitter. At a high-level, a UART is simply a microchip that enables communications between a computing device (3D printer's mother board) and stepper motor drivers.

To understand what a UART does in more detail, it is useful to understand serial communication and parallel communication. We will provide a crash course on those two topics here.

Serial communication in simple terms is sending data a single bit at a time over a given communications bus. While RS-232 (note that UART is commonly used for RS-232) and RS-485 are the two protocols we most commonly associate with the term “serial communication”, many other modern technologies such as USB (Universal Serial Bus), SATA (Serial ATA), and Firewire/IEEE 1394 are also serial protocols. One of the advantages of serial communication is that data can be transmitted at higher frequencies, therefore increasing the amount of data that can be sent despite the fact that less data is sent at once when compared to parallel communication.

Parallel communication, on the other hand, is sending multiple bits of data at the same time over a given communications bus. For example, a 32-bit parallel communications medium such as conventional PCI (Peripheral Component Interconnect) would send 32-bits of data per clock cycle. While one would think that this would give parallel communication significant speed advantages over serial, in practice serial is faster because it can accommodate more clock cycles per second (e.g. GHz speeds).

So, what exactly does this have to do with UART anyway? Simple, UART chips are used to convert inbound serial data to parallel data for the system to read and conversely converts outbound parallel data to serial before sending it on to other systems. Worded differently, a UART enables a system to act as a DTE (Data Terminal Equipment) Device.

Before we move onto SPI, let's quickly cover a few other important details about UART communication:

- UART uses 2 wires: 1 for transmit (or Tx) and one for receive (or Rx)
- UART communication is asynchronous, meaning that it is not synchronized using a clock
- UART has a max communication distance of 15 meters
- UART leverages shift registers to convert serial communication to parallel communication
- UART is commonly used as a "serial port" on computers or in microcontrollers
- UART supports full-duplex communication

What is SPI?

SPI (Serial Peripheral Interface) is a serial communication protocol originally developed by Motorola that enables communication between nearly any electronic device that supports clocked serial streams. SPI uses a master-slave method for communication that enables high-speed data streaming.

As opposed to just using two wires, SPI must use at least 4 wires. As there can be multiple slave devices in an SPI implementation, the actual amount of wires or traces used will be n+3 where n = the number of slave devices in use.

A few additional key details on SPI before we move on:

- SPI is synchronous
- SPI is a simple protocol with little overhead
- SPI supports full-duplex communication
- SPI communication does not have a means for acknowledgment or flow control
- SPI does not use much board space

For more on SPI, check out our knowledge base article on SPI history, SPI theory of operation, and other details on SPI.

Differences between UART vs SPI

While UART and SPI both support full-duplex communication, both use "serial" communication in some way, and they are both only suitable for short-distance use cases, there are not too many similarities beyond that. So, what are the differences between UART vs SPI? The answer is there are many; we will cover a number of them here.

Hardware vs protocol

One of the biggest differences is that UART is a type of hardware while SPI is a protocol. When you are dealing with the nuances of getting things to work in an embedded system, this can be easy to overlook. However, UART is an actual piece of hardware (a microchip) while SPI is a protocol or specification for communication.

Number of pins

UARTs use just two pins while SPI devices need at least 4. When designing a system this means that if pins/traces are scarce, SPI may not be an attractive solution.

Number of devices that can be connected

A corollary to the number of pins, UARTs and SPI devices support a different number of devices. UART, leveraging only Tx and Rx for outbound communication, is effectively limited to 1 to 1 communication. SPI, on the other hand, can leverage its master/slave paradigm to enable one too many communications.

Communications speed

SPI is significantly faster than UART. In some cases, an SPI solution can be three times faster than a UART solution.

Price

In any engineering endeavor, the cost of a given solution is a big driver of choice. Generally speaking, SPI is cheaper than UART.

Asynchronous vs Synchronous

As you can tell from the descriptions of the two earlier, another major difference when comparing UART vs SPI is UART communication is asynchronous while SPI is synchronous.

Size

Generally speaking, SPI devices take up relatively less space than UART chips. This means use cases where there is limited board space may be better served by SPI.

Technology	Transfer type	Number of wires/pins	Data Rate @Distance	Cost	Scalability	Use Cases
UART	Asynchronous	2	20 kbps @ 15m	Moderately expensive	Poor (point-to-point)	Diagnostic displays
SPI	Synchronous	4+	250 Mbps @ .1m	Relatively inexpensive	Moderate	High speed chip-to-chip links

UART vs SPI: What should I use?

There is no one-size-fits-all answer to the “UART vs SPI: What should I use?” question. However, with the information we have given you here, you are now equipped to make an informed decision about which solution is best for a given application you may be working on.

Generally speaking, SPI is preferred in applications that require higher speed communication between chips or applications that require communications between multiple devices. On the other hand, UART may be better suited for applications that have to travel slightly more distance such as diagnostic displays or other applications that require RS-232 support.