

Detachable and Captive Cables

The USB specification defines cables as being either detachable or captive. From the names, you might think that a detachable cable is one you can remove, while a captive cable is permanently attached to its downstream device. But in fact, a captive cable can be removable as long as its downstream connector is *not* one of the standard USB connector types.

A detachable cable must be full/high speed, with a Series-A plug for the upstream connection and a Series-B or mini-B plug for the downstream connection. A captive cable may be low or full/high speed. The upstream end has a Series-A plug. For the downstream connection, the cable can be permanently attached or removable with a non-standard connector type. The non-standard connector doesn't have to be hot pluggable, but the Series-A connector must be hot pluggable. Requiring low-speed cables to be captive eliminates the possibility of trying to use a low-speed cable in a full- or high-speed segment.

Cable Length

Version 1.0 of the USB specification gave maximum lengths for cable segments. A full-speed segment could be up to 5 meters and a low-speed segment could be up to 3 meters. Version 1.1 dropped the length limits in favor of a discussion of characteristics that limit a cable's ability to meet the interface's timing and voltage requirements. On full- and high-speed cables, the limits are due to signal attenuation, cable propagation delay (the amount of time it takes for a signal to travel from driver to receiver), and the voltage drops on the VBUS and GND wires. On low-speed cables, the length is limited by the rise and fall times of the signals, the capacitive load presented by the segment, and the voltage drops on the VBUS and GND wires.

The original limits of 3 and 5 meters are still good guidelines. A 2.0-compliant 5-meter cable will work at full and high speeds. Compliant cables of these lengths are readily available. Chapter 16 explained how the length limits translate to a maximum distance of 30 meters between a host and its peripheral, assuming the use of five hubs and six 5-meter cable segments.

The USB specification prohibits extension cables, which would extend the length of a segment by adding a second cable in series. An extension cable for the upstream side of a cable would have a Series-A plug on one end and a Series-A receptacle on the other, while an extension cable for the downstream side would have a Series-B plug and receptacle.

Prohibiting extension cables eliminates the temptation to stretch a segment beyond the interface's electrical limits. Extension cables are available, but just because you can buy one doesn't mean that it's a good idea or that the cable will work. Instead, buy a single cable of the length you need and add hubs as needed.

An exception is an active extension cable that contains a hub, a downstream port, and a cable. This type of cable works fine because it contains the required hub. Depending on the attached devices, the hub may need its own power supply. Chapter 20 discusses two cable adapters that are approved for use only with On-The-Go devices.

An option for longer distances is to use a standard USB cable that connects to a device that translates between USB and Ethernet, RS-485, or another interface designed for use over long distances. The remote device would then need to support the long-distance interface, rather than USB.

Another option enables you to place a USB device anywhere in a local Ethernet network. Two products that use this approach are the AnywhereUSB hub from Inside Out Networks, Inc. and the USB Server from Keyspan. The hub/server contains one or more host controllers that communicate with the host PC over an Ethernet connection using the Internet Protocol (IP). The hub/server can attach to any Ethernet port in the PC's local network. The device drivers are on the PC. The PC can use the hub/server to access many devices that use bulk and interrupt transfers, with some increased latency due to the additional protocol layer.

Ensuring Signal Quality

The USB specifications for drivers, receivers, and cable design ensure that virtually all data transfers occur without errors. Requirements that help to

ensure signal quality include the use of balanced lines and shielded cables, twisted pairs required for full/high-speed cables, and slower edge rates required for low-speed drivers.

Sources of Noise

Noise can enter a wire in many ways, including by conductive, common-impedance, magnetic, capacitive, and electromagnetic coupling. If a noise voltage is large enough and is present when the receiver is attempting to detect a transmitted bit, the noise can cause the receiver to misread the received logic level. Very large noise voltages can damage components.

Conductive and common-impedance coupling require ohmic contact between the signal wire and the wire that is the source of the noise. Conductive coupling occurs when a wire brings noise from another source into a circuit. For example, a noisy power-supply line can carry noise into the circuit the supply powers. Common-impedance coupling occurs when two circuits share a wire, such as a ground return.

The other types of noise coupling result from interactions between the electric and magnetic fields of the wires themselves and signals that couple into the wires from outside sources, including other wires in the interface. Capacitive and inductive coupling can cause crosstalk, where signals on one wire enter another wire. Capacitive coupling, also called electric coupling, occurs when two wires carry charges at different potentials, resulting in an electric field between the wires. The strength of the field and the resulting capacitive coupling varies with the distance between the wires. Inductive, or magnetic, coupling occurs because current in a wire causes the wire to emanate a magnetic field. When the magnetic fields of two wires overlap, the energy in each wire's field induces a current in the other wire. When wires are greater than $1/6$ wavelength apart, the capacitive and inductive coupling is considered together as electromagnetic coupling. An example of electromagnetic coupling is when a wire acts as a receiving antenna for radio waves.

Balanced Lines

One way that USB eliminates noise is with the balanced lines that carry the bus's differential signals. Balanced lines are electrically quiet. Noise that couples into the interface is likely to couple equally into both signal wires. At a differential receiver, which detects only the difference between the two wires' voltages, any noise that is common to both wires cancels out.

In contrast, in the unbalanced, single-ended lines used by RS-232 and other interfaces, the receiver detects the difference between a signal wire and a ground line shared by other circuits. The ground line is likely to be carrying noise from a number of sources, and the receiver sees this noise when it detects the difference between the signal voltage and ground.

Twisted Pairs

In a full/high-speed USB cable, the two signal wires must form a twisted pair. Twisted pairs are recommended, but not required, for low-speed cables. A twisted pair is two insulated conductors that spiral around each other with a twist every few inches (Figure 19-9). The twisting reduces noise in two ways: by reducing the amount of noise in the wires and by canceling whatever noise does enter the wires. Twisting is most effective at eliminating low-frequency, magnetically coupled signals such as 60-Hz power-line noise.

Twisting reduces noise by minimizing the area between the conductors. The magnetic field that emanates from a circuit is proportional to the area between the conductors. Twisting the conductors around each other reduces the total area between them. The tighter the twists, the smaller the area. Reducing the area shrinks the magnetic field emanating from the wires and thus reduces the amount of noise coupling into the field.

A twisted pair tends to cancel any noise that enters the wires because the conductors swap physical positions with each twist. Any noise that magnetically couples into the wires reverses polarity with each twist. The result is that the noise present in one twist is cancelled by a nearly equal, opposite

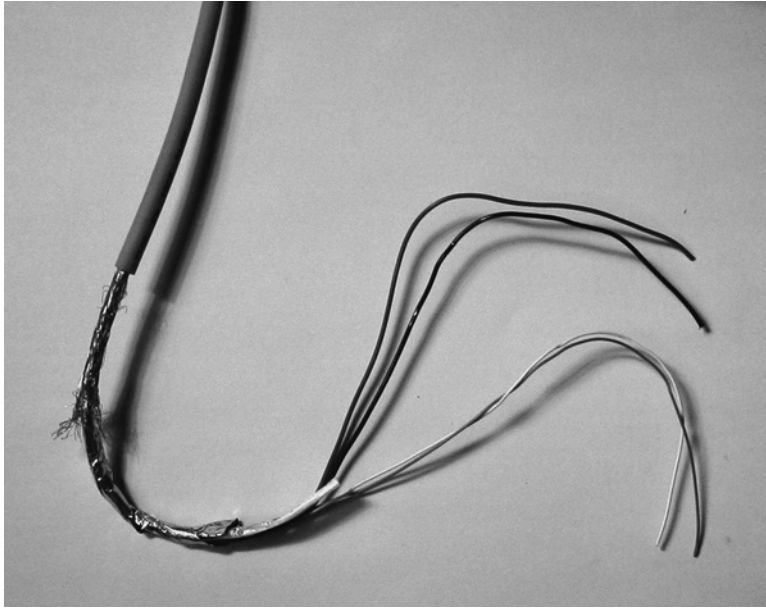


Figure 19-9: A full/high-speed USB cable contains a twisted pair for data, VBUS and GND wires, and aluminum metallized polyester and braided copper shields.

noise signal in the next twist. Of course, the twists aren't perfectly uniform, so the canceling isn't perfect, but noise is much reduced.

Shielding

Metal shielding prevents noise from entering or emanating from a cable. Shielding is most effective at blocking noise due to capacitive, electromagnetic, and high-frequency magnetic coupling. The USB 2.0 specification requires both low-speed and full/high-speed cables to be shielded, though the requirements differ.

In a full/high-speed cable, an aluminum metallized polyester shield surrounds the four conductors. Around this shield is an outer shield of braided, tinned copper wire. Between the shields and contacting both is a copper drain wire. The outside layer is a polyvinyl chloride jacket. The shield terminates at the connector plug.

A low-speed cable has the same requirements except that the braided outer shield is recommended but not required. The 1.x specification required no shielding for low-speed cables on the premise that the slower rise and fall times made shielding unnecessary. The shielding requirement was added in USB 2.0 not because the USB interface is noisy in itself, but because the cables are likely to attach to computers that are noisy internally. Shielding helps to keep the cable from radiating this noise and helps the cable pass FCC tests. The downside is that 2.0-compliant low-speed cables are more expensive to make and physically less flexible.

Edge Rates

Low speed's slower data rate enables the drivers to use slower edge rates that reduce both the reflected voltages seen by receivers and the noise that emanates from the cable.

When a digital output switches, a mismatch between the line's characteristic impedance and the load presented by the receiver can cause reflected voltages that affect the voltage at the receiver. If the reflections are large enough and last long enough, the receiver may misread a transmitted bit.

In low-speed cables, the slower edge rate ensures that any reflections have died out by the time the output has finished switching. The slow edge rate also means that the signals contain less high-frequency energy and thus the noise emanated by the cables is less.

Isolated Interfaces

Galvanic isolation can be useful in preventing electrical noise and power surges from coupling into a circuit. Circuits that are galvanically isolated from each other have no ohmic connection. Typical methods of isolation include using a transformer that transfers power by magnetic coupling and optoisolators that transfer digital signals by optical coupling.

USB devices should require no additional protection in offices, classrooms, and similar environments. For industrial environments or anywhere that devices require additional protection, USB's timing requirements and use of a single pair of wires for both directions make it difficult to completely iso-

late a USB device from its host. It is feasible, however, to isolate the circuits that a device controller connects to. For example, in a motor controller with a USB interface, the motor and control circuits can be isolated from the USB controller and bus.

Another option is an isolated hub available from B & B Electronics. The hub has four low- and full-speed downstream ports with 2500 VAC of optical isolation between the upstream port and the downstream ports.

Wireless Links

For the same reasons that isolated USB interfaces are difficult to implement, replacing a USB cable with a wireless connection isn't a simple task. USB transactions involve communicating in both directions with tight timing requirements. For example, when a host sends a token and data packet in the Data stage of an interrupt OUT transaction, the device must respond quickly with ACK or another code in the handshake packet.

But the idea of a wireless connection for USB devices is so appealing that several technologies that incorporate USB in wireless devices are available and under development. In most implementations, the wireless links use conventional wired devices that serve as wireless bridges, or adapters. The bridge or adapter uses USB to communicate with the host and a wireless link to communicate with the peripheral. The peripheral contains a wireless bridge to convert between the wireless interface and the peripheral's circuits.

Cypress WirelessUSB

Cypress Semiconductor offers the WirelessUSB technology as a solution for low-speed devices, including HIDs, without cables. The obvious market is wireless keyboards, mice, and game controllers. With a wireless range of up to 50 meters, the technology might also find uses in building and home automation and industrial control. The wireless interface uses radio-frequency (RF) transmissions at 2.4 Gigahertz in the unlicensed Industrial, Scientific, and Medical (ISM) band.

A WirelessUSB system consists of a WirelessUSB bridge and one or more WirelessUSB devices (Figure 19-10). The bridge translates between USB and the wireless protocol and medium. The WirelessUSB device carries out the device's function (mouse, keyboard, game controller) and communicates with the bridge.

The bridge contains a USB-capable microcontroller and a WirelessUSB transceiver chip and antenna. The WirelessUSB device contains a Cypress PSoC or another microcontroller and a WirelessUSB transmitter or transceiver chip and antenna. A device with a transceiver is 2-way: the device can communicate in both directions. A device with just a transmitter is 1-way: the device can send data to the host but can't receive data or status information. In both the bridge and device, the transmitter and transceiver chips use the SPI synchronous serial interface to communicate with their microcontrollers.

In a 2-way system, when a device has data to send to the host, the device's microcontroller writes the data to the transceiver chip, which encodes the data and transmits it through the air to the bridge's transceiver. On receiving the data, the bridge returns an acknowledgement to the device, decodes the data, and sends the data to the host in conventional USB interrupt or control transfers. If the device doesn't receive an acknowledgement from the bridge, the device resends the data.

When the host has data to send to the device, the host writes the data to the bridge's USB controller, which ACKs the data (if not busy) and passes the data to the bridge's transceiver. The transceiver encodes the data and sends it over the air to the WirelessUSB device. The device returns an acknowledgement to the bridge. On receiving a NAK or no reply, the bridge retries the transmission.

In a 1-way system, a device sends data to the host in much the same way as in a 2-way system, except that the device receives no acknowledgements from the host. To help ensure that the bridge and host receive all transmitted data, the device sends its data multiple times. Sequence numbers enable the bridge to identify previously received data.

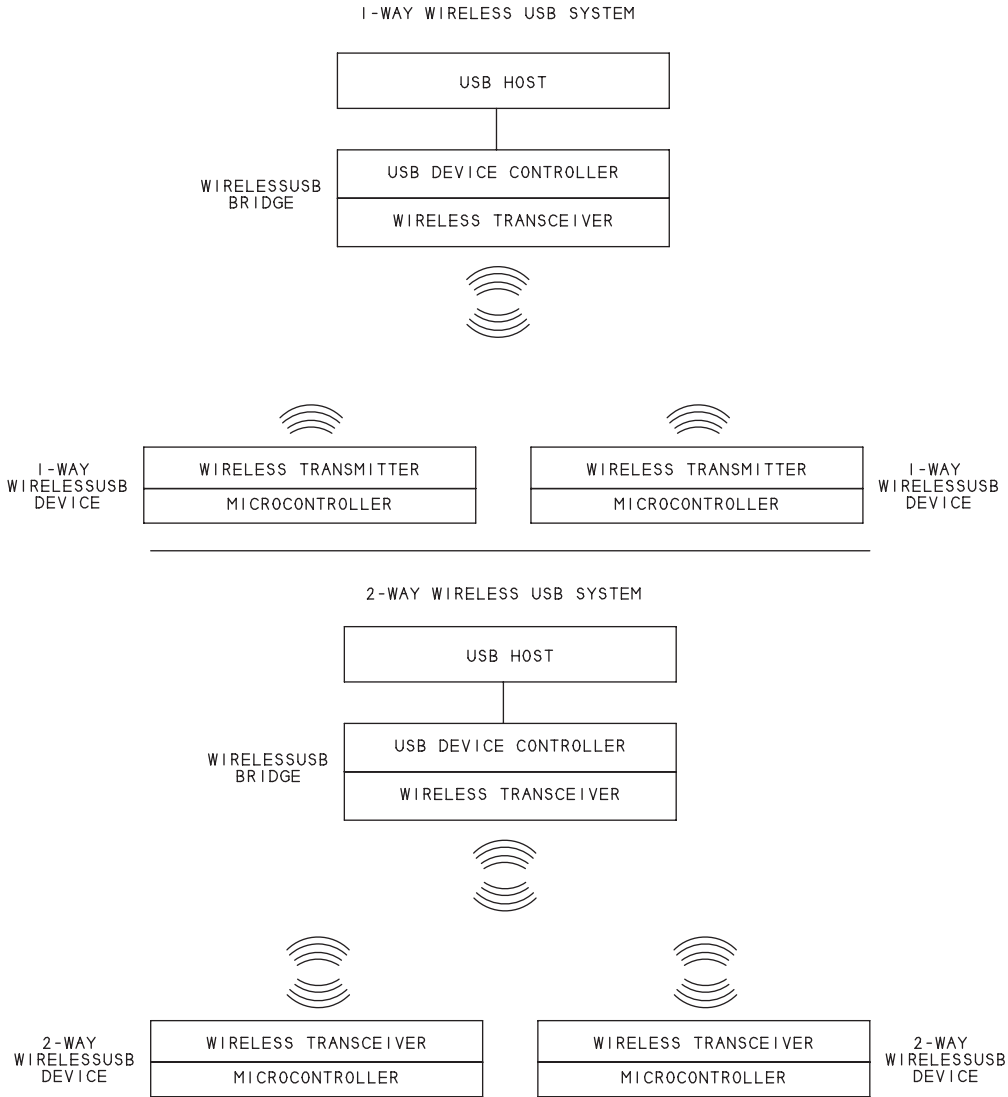


Figure 19-10: WirelessUSB provides a way to design low-speed devices that use a wireless interface.

With both systems, the host thinks it's communicating with an ordinary HID and has no knowledge of the wireless link.

A WirelessUSB link can have a data throughput of up to 62.5 kilobits/sec., but low-speed traffic is of course limited to the USB bandwidth available for low-speed control and interrupt transfers. A device and its bridge must use the same frequency/code pair. A single WirelessUSB bridge can use multiple frequency/code pairs to communicate with multiple devices.

For devices with human interfaces, communications between the wired and wireless interfaces must be fast enough so users don't perceive delays in response to keypresses, mouse movements, and similar actions. For faster performance, the microcontroller can use burst reads to read multiple registers in the WirelessUSB chip in sequence.

The Wireless USB Initiative

The mission of the Wireless USB Promoter Group is to specify a Wireless USB (WUSB) extension that can transmit at 480 Megabits/sec. over a distance of 3 meters (and at lower speeds over longer distances). Note that Wireless USB (WUSB) and Cypress' WirelessUSB have similar names but are different and unrelated technologies!

In Wireless USB, a conventional USB host can have a wired connection to a USB device that functions as a host wire adapter (HWA). The HWA can communicate with native WUSB devices and with device wire adapters (DWAs). A native WUSB device is a peripheral with Wireless USB support built in. A DWA connects to a conventional wired USB device and enables the wired device to communicate over the wireless link. Data on the wireless link is encrypted.

The members of the Wireless USB Promoter Group are Agere Systems, Hewlett Packard, Intel, Microsoft Corporation, NEC, Philips Semiconductors and Samsung Electronics. The specification is due for release in 2005.

Other Options

Other ways to use USB in wireless devices include various wireless bridges and a wireless networking option.

ZigBee is an inexpensive, low-power, RF interface suitable for building and industrial automation and other applications that transmit at up to 250 kilobits/sec. and over distances of up to 500 meters. DLP Design's DLP-RF1 USB/RF OEM Transceiver Module provides a way to monitor and control a Zigbee interface from a USB port. The module's USB controller is FTDI Chip's FT245BM. One or more DLP-RF2 RF OEM Transceiver Modules can communicate with the DLP-RF1.

The IrDA Bridge class described in Chapter 7 defines a way for a USB device to use bulk transfers to communicate over an infrared link.

Another option is a vendor-specific wireless bridge that uses infrared, RF, or other wireless modules designed for use in robotics and other low- to moderate-speed applications. The bridge functions as a wired USB device and supports a wireless interface. A remote device carries out the peripheral's function and also supports the wireless interface. Firmware in the bridge passes received wireless data to the host and passes received USB data to the device.

If you want to use an existing USB device wirelessly, you may be able to use the AnywhereUSB or Keyspan hub/server described earlier in this chapter with a wireless network interface between the host PC and the hub/server.