16

Managing Power

A convenient feature of USB is the ability for devices to draw power from the bus. Many devices can be entirely bus powered. But drawing power from the bus carries the responsibility to live within the limits of available power, including entering the low-power Suspend state when required.

This chapter will help you decide whether or not your design can use bus power. And whether your design is bus-powered or self-powered, you'll find out how to ensure that your design follows the USB specification's requirements for managing power.

Powering Options

Inside a typical PC is a power supply with amperes to spare. Many hubs have their own power supplies as well. Many USB peripherals can take advantage of these existing supplies rather than having to provide their own power sources.

The ability to draw power from the same cable that carries data to and from the PC has several advantages. Users no longer need an electrical outlet near the peripheral, and a peripheral can be physically smaller and lighter in weight without an internal power supply. A peripheral without a power supply costs less to manufacture and thus can sell for less. A bus-powered device can save energy because power supplies in PCs use efficient switching regulators rather than the cheap linear regulators in the "wall bugs" that many peripherals provide instead of an internal supply. (Most self-powered hubs use wall bugs, however.)

Before USB, most peripherals used the PC's RS-232 serial and printer ports. Neither of these includes a power-supply line. The ability to use bus power was so compelling that the designers of some peripherals that connect to these ports used schemes that borrow the small amount of current available from unused data or control outputs in the interface. With a super-efficient regulator, you can get a few milliamperes from a serial or parallel port to power a device. Another approach used by some peripherals was to kludge onto the keyboard connector, which has access to the PC's power supply. With USB, you don't have to resort to these tricks.

Voltages

The nominal voltage between the VBUS and GND wires in a USB cable is 5V, but the actual value can be a little more or quite a bit less. A device that uses bus power must be able to handle the variations.

These are the minimum and maximum voltages allowed at a hub's downstream ports:

Hub Type	Minimum Voltage	Maximum Voltage
High Power	4.75	5.25
Low Power	4.4	5.25

To allow for cable and other losses, devices should be able to function with supply voltages a few tenths of a volt less than the minimum available at the

hub's connector. In addition, transient conditions can cause the voltage at a low-power hub's port to drop briefly to as low as 4.07V.

If components in a device need a higher voltage, the device can contain a step-up switching regulator. Most USB controller chips require a +5V or +3.3V supply. Components that use 3.3V are handy because the device can use an inexpensive low-dropout linear regulator to obtain 3.3V from VBUS.

Which Peripherals Can Use Bus Power?

Not every peripheral can take advantage of bus power. Although the bus can provide generous amounts of current in comparison to other interfaces, the current available from a PC's power supply or an external hub is limited. Figure 16-1's chart will help you decide whether a device can use bus power.

Advances in semiconductor technology have reduced the power required by many electronic devices. This is good news for designers of bus-powered devices. Thanks to CMOS processes used in chip manufacturing, lower supply voltages for components, and power-conserving modes in CPUs, you can do a lot with 100 milliamperes.

A device that requires up to 100 milliamperes can be bus powered from any host or hub. A device that requires up to 500 milliamperes can use bus power when attached to a self-powered hub or any host except some battery-powered hosts. No device should draw more than 100 milliamperes until the host has configured the device. And all devices must limit their power consumption when the bus is suspended.

Of course, some devices need to function when they're not attached to a host. A digital camera is an example. These devices need their own supplies. Self power can use batteries or power from a wall socket. To save battery power without requiring users to plug in a supply, a device can be designed to be bus-powered when connected to the bus and self-powered otherwise. Some battery-powered devices can recharge when attached to the bus.

A device in the Suspend state can draw very little current from the bus, so some devices will need their own supplies to enable operating when the bus is suspended.

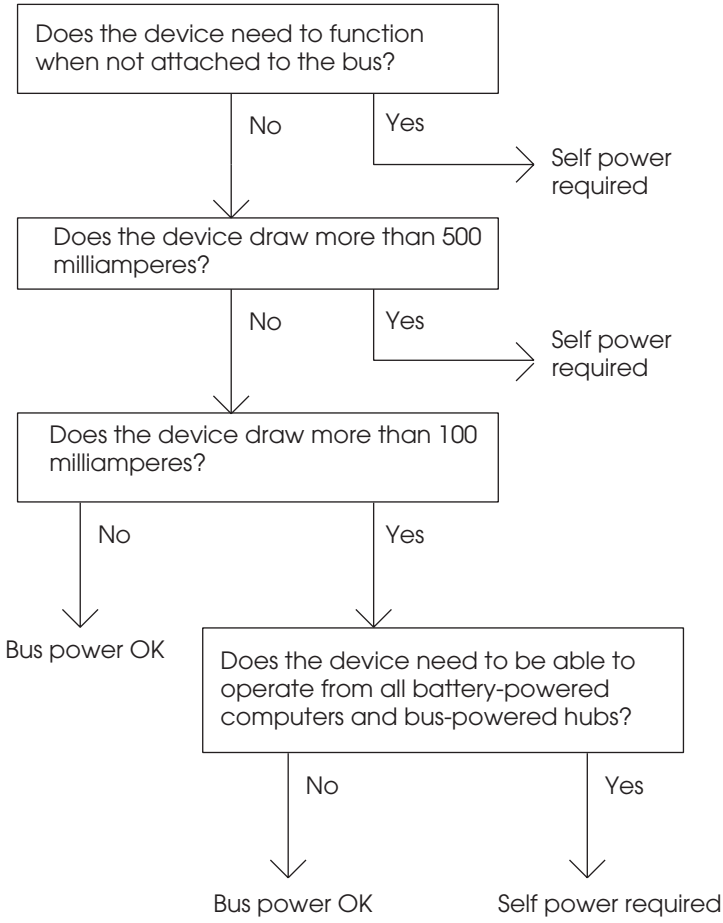

Figure 16-1: Not every device can use bus power alone. A device that uses bus power must also meet the USB specification's limits for Suspend current.

Power Needs

The USB specification defines a low-power device as one that draws up to 100 milliamperes from the bus, and a high-power device as one that draws up to 500 milliamperes from the bus. A self-powered device can draw up to 100 milliamperes from the bus and as much power as is available from the device's supply.

A high-power device must be able to enumerate at low power. On power-up, any device can draw up to 100 milliamperes from the bus until the device is configured during enumeration. After retrieving a configuration descriptor, the host examines the amount of current requested in bMaxPower, and if the current is available, the host sends a Set_Configuration request specifying the configuration. The device can then draw up to bMaxPower from the bus. In reality, hosts and hubs are likely to allocate either 100 or 500 milliamperes to a device rather than the precise amount requested in bMaxPower.

A self-powered device may also draw up to 100 milliamperes from the bus at any time that the bus isn't suspended. This capability enables the device's USB interface to function when the device's power supply is off so the host can detect and enumerate the device. Otherwise, if a device's pull-up is bus-powered but the rest of the interface is self-powered, the host will detect the device but won't be able to communicate with it.

These limits are absolute maximums, not averages. And remember that the bus's power-supply voltage can be as high as 5.25V, which may result in greater current consumption.

A device must never provide upstream power. Even the pull-up resistor must remain unpowered until VBUS is present. So self-powered devices must have a connection to VBUS to detect its presence even if the device never uses bus power.

Informing the Host

During enumeration, the host learns whether the device is self powered or bus powered and the maximum current the device will draw from the bus. As Chapter 4 explained, each device's configuration descriptor holds a bMaxPower value that specifies the maximum bus current the device requires. All hubs have over-current protection that prevents excessive current from flowing to a device.

If you connect a high-power device to a low-power hub, Windows displays a message informing you that the hub doesn't have enough power available and offering assistance. If the bus has a low-power device connected to a

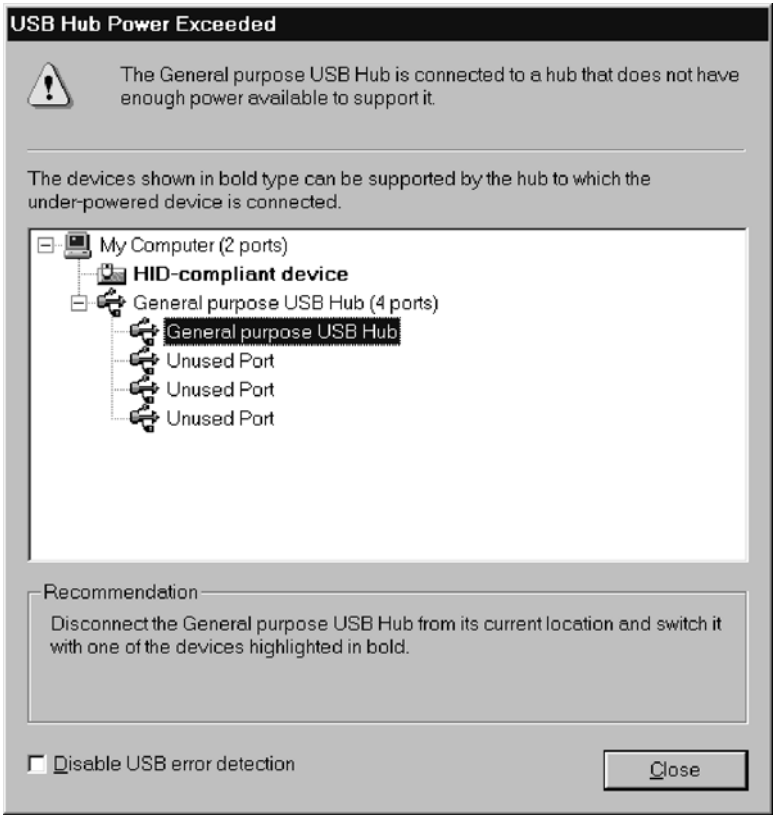

Figure 16-2: Windows warns users when they connect a high-power device to a low-power hub, and helps them find an alternate connection.

high-power port, Windows will recommend swapping the device with the high-power device (Figure 16-2).

A device can support both bus-powered and self-powered options, using self power when available and bus power (possibly with limited abilities) otherwise. When the power source changes, the host must re-enumerate the device. To enable forcing a re-enumeration, power to the device's bus pull-up resistor may be switched off briefly, then back on, to simulate a disconnect and re-connect. If the device doesn't have this ability, users will need to remove the device from the bus before attaching or removing the power supply. The device reports its use of bus or self power in response to a Get_Status (Device) request from the host.

Hub Power

Powering options for hubs are similar to the options for other devices, but hubs have some special considerations. A hub must also control power to its devices and monitor power consumption, taking action when the devices are using too much current and presenting a safety hazard.

Power Sources

Like other devices, all hubs except the root hub are self-powered or bus-powered. The root hub gets its power from the host.

If the host uses AC power from a wall socket or another external source, the root hub must be high power and capable of supplying 500 milliamperes to each port on the hub. If the host is battery-powered, the hub may supply either 500 or 100 milliamperes to each port on the hub. A hub that supplies 100 milliamperes is a low-power hub.

All of a bus-powered hub's downstream devices must be low power. The hub can draw no more than 500 milliamperes and the hub itself will use some of this, leaving less than 500 milliamperes for all attached devices combined.

Don't connect two bus-powered hubs in series. The upstream hub can guarantee no more that 100 milliamperes to each downstream port, and that doesn't leave enough current to power a second hub that also has one or more downstream ports, each requiring 100 milliamperes. An exception is a bus-powered compound device, which consists of a hub and one or more downstream, non-removable devices. In this case, the hub's configuration descriptor can report the maximum power required by the hub's electronics plus its non-removable device(s). The configuration descriptors for the non-removable device(s) report that the devices are self-powered with bMaxPower = 0. The hub descriptor indicates whether a hub's ports are removable.

Like other high-power, bus-powered devices, a bus-powered hub can draw up to 100 milliamperes until it's configured and up to 500 milliamperes after being configured. During configuration, the hub must manage the

available current so its devices and the hub combined don't exceed the allowed current.

Like other self-powered devices, a self-powered hub may also draw up to 100 milliamperes from the bus so the hub interface can continue to function when the hub's power supply is off. If the hub's power is from an external source, such as AC power from a wall socket, the hub is high power and must be capable of supplying 500 milliamperes to each port on the hub. If the hub uses battery power, the hub may supply 100 or 500 milliamperes to each port on the hub.

Over-current Protection

As a safety precaution, hubs must be able to detect an over-current condition, which occurs when the current used by the total of all devices attached to the hub exceeds a preset value. When the port circuits on a hub detect an over-current condition, they limit the current at the port and the hub informs the host of the problem. Windows warns the user when a device exceeds the current limit of its hub port (Figure 16-3).

The USB 2.0 specification says only that the current that triggers the over-current actions must be less than 5 amperes. To allow for transient currents, the over-current value should be greater than the total of the maximum allowed currents for the devices. In the worst case, seven high-power, bus-powered downstream devices can legally draw up to 3.5 amperes. So a supply for a self-powered hub with up to seven downstream ports would provide much less than 5 amperes at all times unless something goes very wrong.

The USB specification allows a device to draw larger inrush currents when it attaches to the bus. This current is typically provided by the stored energy in a capacitor that is downstream from the over-current protection circuits so the protection circuits don't see the inrush current. If the inrush current is too large, the device will fail the USB-IF's compliance tests.

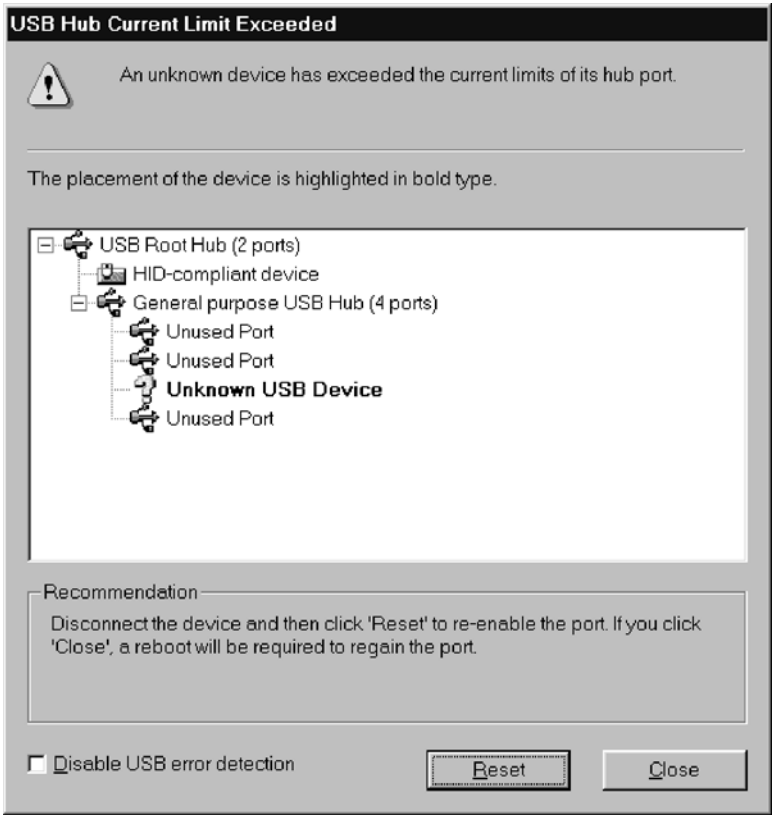

Figure 16-3: When a device exceeds the current limit of its hub's port, Windows warns the user and offers assistance.

Power Switching

A bus-powered hub must support power switching that can provide and cut off power to downstream ports in response to control requests. A single switch may control all ports, or the ports may switch individually. A self-powered hub must support switching to the Powered Off state and may also support power switching via control transfers.

Saving Power

The Suspend state reduces a device's use of bus power when the host has no reason to communicate with the device. A device must enter the Suspend state when there has been no activity on the bus for 3 milliseconds.

The USB specification limits the current that a suspended device can draw to a few milliamperes for high-power devices with remote wakeup enabled, and to much less than this amount for other devices. A device that needs to function even when the host has ceased communicating may need to be self-powered. However, many peripheral controllers can shut down and consume very little power while still being able to detect activity requiring attention on an I/O pin and wake up the host as needed.

Global and Selective Suspends

Most suspends are global, where the host stops communicating with the entire bus. When a PC detects no activity for a period of time, the PC enters a low-power state and stops sending Start-of-Frame packets on the bus. When a full-or high-speed device detects that no Start-of-Frame packet has arrived for 3 milliseconds, the device enters the Suspend state. Low-speed devices do the same when they haven't received a low-speed keep-alive signal for 3 milliseconds. A device must be in the Suspend state within 10 milliseconds of no bus activity.

A host may also request a selective suspend of an individual port. The host sends a Set_Port_Feature request to the a hub with the Index field set to a port number and the wValue field set to Port_Suspend. (See Chapter 15.) This request instructs the hub to stop sending any traffic, including Start-of-Frames or low-speed keep-alives, to the named port.

Current Limits for Suspended Devices

For all devices except high-power devices whose remote-wakup feature has been enabled, the USB 2.0 specification says that the device can draw no more than 500 *micro*amperes from the bus when in the Suspend state. This is very little current, and it includes the current through the device's bus

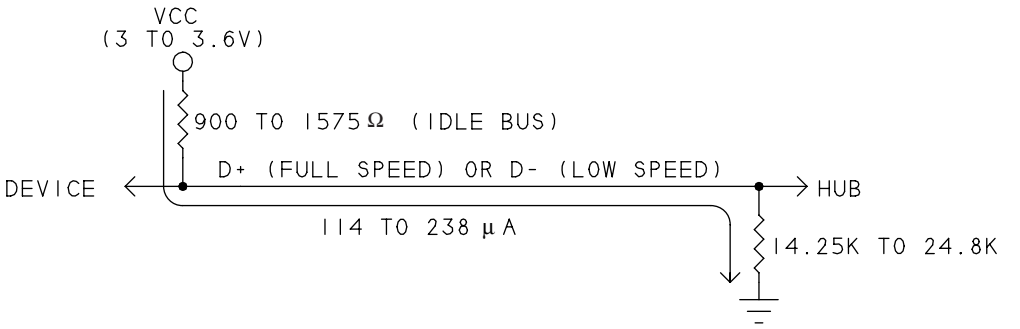

Figure 16-4: The allowed bus current in the Suspend state includes the current through the device's pull-up.

pull-up resistor. As Figure 16-4 shows, the pull-up current flows from the device's pull-up supply, which must be between 3.0 and 3.6V, through the device's pullup and the hub's pull-down, to ground. In the worst case, with a pull-up voltage of 3.6V and resistors that are the minimum allowed values, the pull-up current is 238 microamperes, leaving just 262 microamperes for everything else.

High-speed devices, which don't use pull-ups in normal communications, must switch to full speed and use a pull-up when entering the Suspend state. So high-speed devices have the same restriction on available current.

In compliance testing, however, the USB-IF has granted automatic waivers to low-power devices that consume up to 2.5 milliamperes in the Suspend state.

A high-power device that supports remote wakeup and whose remote-wakeup feature has been enabled by the host can draw up to 2.5 milliamperes from the bus when suspended, including current through the pull-up resistor. Every device connects as low power, so to comply with the USB specification, a device must meet the 500-microampere limit if the host suspends the device before configuring it as high power with remote wakeup enabled (assuming no waiver has been granted).

The limits are averages over intervals of up to 1 second, so brief currents can be greater. For example, a flashing LED that draws 20 milliamperes for one tenth of each second draws an average of 2 milliamperes per second.

A device should begin to enter the Suspend state after its bus segment has been in the Idle state for 3 milliseconds. The device must be in the Suspend state after its bus segment has been in the Idle state for 10 milliseconds.

Resuming Communications

When in the Suspend state, two actions can cause a device to enter the Resume state and restart communications. Any activity on the bus will cause the device to enter the Resume state. And if the device's remote wakeup feature is enabled by the host, the device itself may request a resume at any time.

To resume, the host places the bus in the Resume state (the K state, defined in Chapter 18) for at least 20 milliseconds. The host follows the Resume with a low-speed End-of-Packet signal. (Some hosts incorrectly send the End-of-Packet after just a few hundred microseconds.) The host then resumes sending Start-of-Frame packets and any other communications requested by the device driver.

A device causes a Resume by driving the upstream bus segment in the Resume state for between 1 and 15 milliseconds. The device then places its drivers in a high-impedance state to enable receiving traffic from the upstream hub. A device may initiate a Resume any time after the bus has been suspended for at least 5 milliseconds. The host-controller software must allow all devices at least 10 milliseconds to recover from a Resume.

Depending on a device's USB controller, monitoring the bus to determine whether to enter the Suspend state may require firmware support. The resume signaling is normally handled by the device's serial interface engine and requires no firmware support.

When a device uses bus power, the USB controller may need to control power to external circuits, removing power on entering the Suspend state and restoring power on resuming. A power switch with soft-start capability can prevent problems by limiting current surges when the switch turns on. Micrel Inc. has several power-distribution switches suitable for use with USB devices. Each switch contains one or more high-side MOSFET switches with soft-start capability.

Power Management under Windows

Recent PCs manage power according to the Advanced Configuration and Power Interface Specification (ACPI). The specification, first released in 1997, was developed by Hewlett-Packard, Intel, Microsoft, Phoenix Technologies, and Toshiba. Revision 3.0 was released in 2004. A system that implements ACPI power management enables the operating system to conserve power by shutting down components, including suspending the USB bus, when the computer is idle.

To implement ACPI, a PC must contain an ACPI controller. An ACPI BIOS provides tables that describe the power-management capabilities of system hardware and routines that the operating system can execute.

PCs support three low-power, or sleeping, states:

In the S1 state, the display is off and drives are powered down. USB buses are suspended, but VBUS remains powered.

In the S3 state, the PCI bus's main power supply is off and memory is not accessed, but system memory continues to be refreshed. Devices that can wake the system receive power from the PCI bus's auxiliary supply (Vaux). In older systems, USB's VBUS is not powered in the S3 state. In newer systems, VBUS is powered by Vaux.

In the S4 state, the system context is saved to disk and the system, including the USB bus, is powered off.

In Windows XP, you can view and change a system's power-management options in Control Panel > Power Options. In the Power Schemes tab (Figure 16-5), you can specify when the system goes into standby and hibernation. Hibernation is the S4 state. Standby is either S1 or S3. On a system that has no USB devices that can wake the system, the standby state is S3. On a system that has a USB keyboard, mouse, or another USB device that can wake the system, the standby state is S1 due to problems in using S3 with some (misbehaving) hardware. The problems include loss of VBUS in

Figure 16-5: The Power Options Properties in Windows' Control Panel enable users to specify power-saving schemes that determine when USB devices must enter the Suspend state.

the S3 state, false device removal and arrival notifications on resuming, resetting of devices during suspend and resume, and failure to resume fully.

To enable or disable remote wakeup capability for a specific device that supports remote wakeup, in Windows' Device Manager, select the device, right-click, select Properties > Power Management, and check or uncheck *Allow this device to bring the computer out of standby.*

On some early Intel host controllers, a suspended root port didn't respond correctly to a remote wakeup. In addition, using remote wake-up requires

work-arounds under Windows 98 Gold, Windows 98 SE, and Windows Me. With these operating systems, a device may wake up properly, but the device's driver isn't made aware of the wakeup so communications can't resume. A white paper from Intel titled *Understanding WDM Power Management* by Kosta Koeman (available from the USB-IF's Web site) details the problem and solutions. In short, a device using these operating systems shouldn't place itself in the Suspend state unless the host requests it, and the device driver requires extra code to ensure that the wake-up completes successfully. Windows 2000 and later don't have this problem.