AN4767

USB5734 FlexConnect Operation

Authors: *Andrew Rogers and Jo Chen*
Microchip Technology Inc.

1.0 INTRODUCTION

The Microchip USB5734 smart hub allows the FlexConnect-capable USB port 1 to assume the role of a USB host at any time during hub operation. This host role exchange feature is called FlexConnect.

This functionality can be used in two primary ways:

- *Host Swapping*: This functionality can be achieved through a hub where a host and device can agree to swap the host-device relationship. The host becomes a device, and the device becomes a host.
- *Host Sharing*: A USB ecosystem can be shared between multiple hosts. Note that only one host may access the USB tree at a time.

FlexConnect can be enabled through any of the following methods:

- *Register Control:* An embedded SMBus/I^2C controller can configure and initiate FlexConnect through SMBus/I^2C commands during runtime.
- *USB Command:* A USB host can initiate FlexConnect via vendor-specific USB commands to either the hub's internal Hub Feature Controller (HFC) device or Hub Control Endpoint.
- *Direct Pin Control*: **PROG_FUNC6** in configuration 1 or 2 is assigned the role of a FlexConnect control pin.

1.1 Sections

This document includes the following topics:

1.2 References

Consult the following documents for details on the specific parts referred to in this document:

- *USB5734 Data Sheet* (www.microchip.com/DS00001854)
- *AN1903 - Configuration Options for the USB5734, USB5744, and USB5742* (www.microchip.com/en-us/application-notes/an1903)

1.3 Terms and Abbreviations

- **FlexConnect**: The feature allows the hub to reassign the host role to any port dynamically.
- **Flexed state**: This is the state that the hub enters into when FlexConnect has been initiated.
- **Unflexed state**: This is the default state of the hub when port 0 is configured as the host role.

Note: *On USB Compliance and FlexConnect:* It is strongly recommended that the user designs their FlexConnect application such that while the system is not in any Flexed state, the system is capable of passing all USB compliance tests. The FlexConnect feature is a function that operates outside of the boundaries of the USB specifications. It is important to understand that while in a Flexed state, there is no USB specification to draw from, and hence no test procedure for testing while in the Flexed state is defined. Hence, even if a system is designed, which is capable of passing USB compliance tests while in the Flexed state, the results should be interpreted as informative only. Conversely, a system that implements FlexConnect but fails USB compliance tests while in the Flexed state is not inherently designed improperly. It is the system implementer's responsibility to ensure that their FlexConnect design does not operate in an irresponsible manner, which is capable of causing damage or unstable operation when interfaced with standard certified hosts, hubs, and devices that are available for purchase in the market. The user should ensure that their FlexConnect application still operates as is required by the USB specification while not in the Flexed state, and that the Flexed state can only be initiated in a controlled and deliberate manner such as to ensure safe and reliable operation.

2.0 FUNCTIONAL OVERVIEW

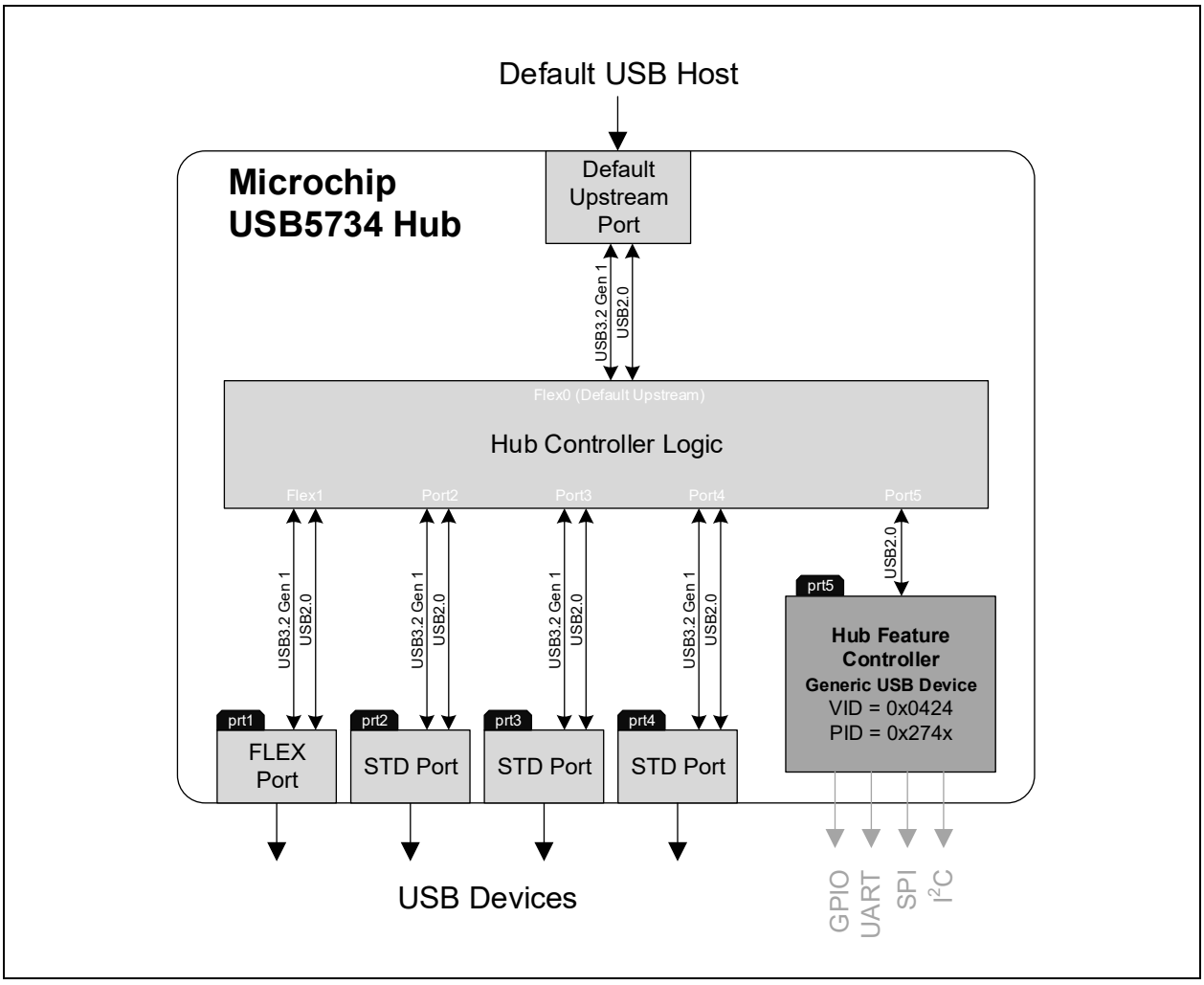
The internal block diagram of Microchip USB5734 device is shown in Figure 1.

The USB5734 device has an internal USB device called the Hub Feature Controller (HFC), which enables the advanced features of the hub.

The HFC device is a standard generic device class with a Product ID (PID) of 0x274x (where “x” depends on the configuration mode selected via the **CFG_STRAP** pin).

The HFC is permanently connected to internal port 5 on the USB5734. This port is configured as non-removable.

FIGURE 1: USB5734 INTERNAL BLOCK DIAGRAM



2.1 FlexConnect Initiation

FlexConnect can be initiated via one of the following three methods. It is strongly recommended to select only one method of control to ensure that the system operates in a deterministic manner. Hybrid methods can be implemented, but special attention should be made to ensure all control mechanisms can stay in sync.

- *SMBus Control* through register settings: This method is recommended if configuration of the hub through SMBus is needed for other purposes.The SMBus and runtime register details are located in Section 4.0, "FlexConnect Control Via SMBus".

Note:	The hub is reset every time FlexConnect is initiated or terminated. If the SMBus configuration channel is used in the system implementation, the hub will wait in the configuration stage until it is configured and instructed to enter the runtime stage via the SMBus Attach command. This can create conflicts if FlexConnect is controlled via Direct Pin Control or USB Command methods along with SMBus configuration, as the controller configuring the hub must be aware of the change in FlexConnect state and ensure to reconfigure the hub in a timely manner.

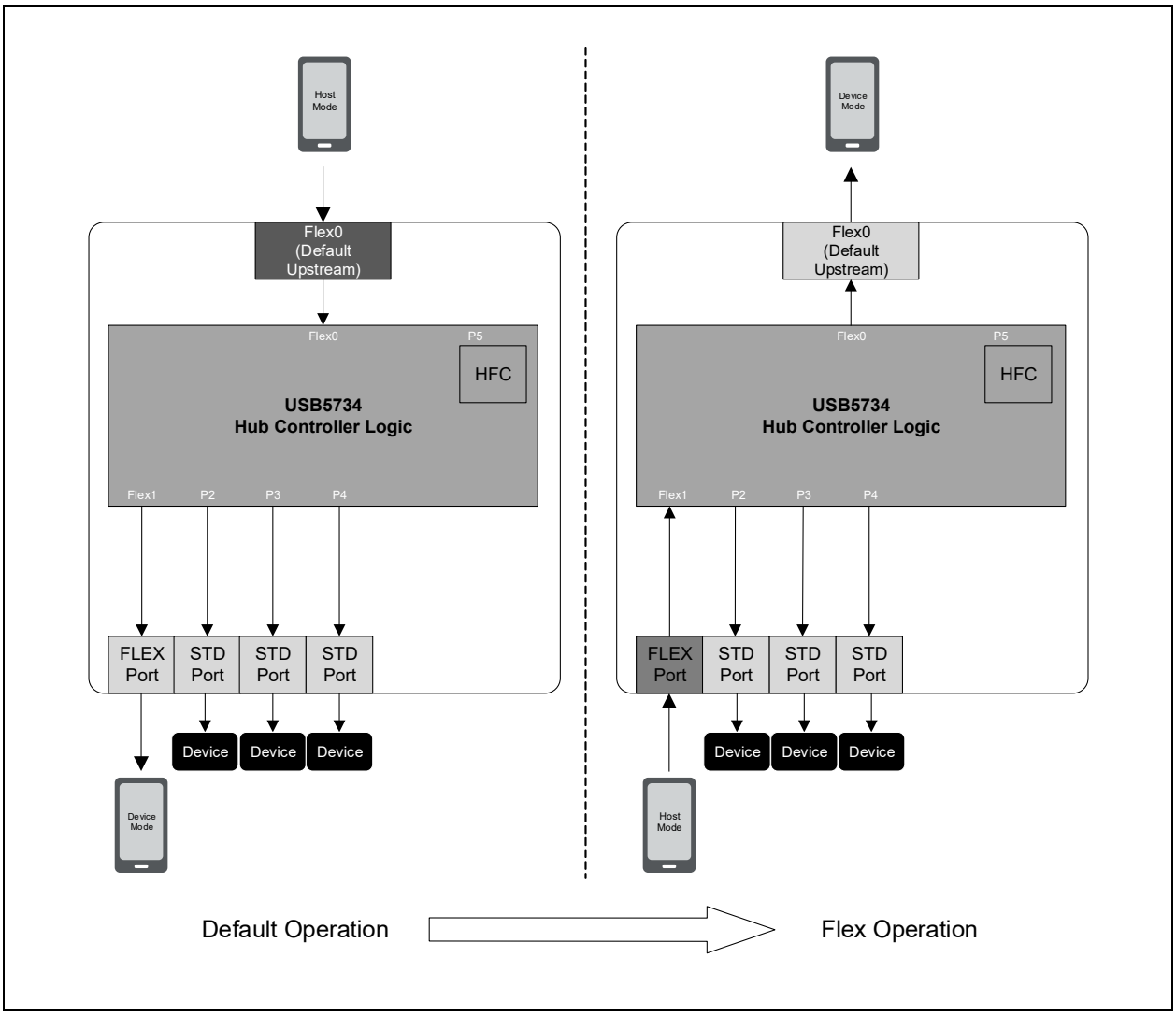
- *USB Command* to the internal HFC device or the hub control endpoints if using vendor-specific messaging (VSM) commands: Details of this USB command are shown in Section 5.0, "FlexConnect USB Command Details", and an example USB protocol trace capture of the command is detailed in Section 5.1.1, "FlexConnect USB Command Example".
- *Direct Pin Control* by digital control of the **FLEXCMD** pin (**PROG_FUNC6** in configuration 1 or 2): This pin has an internal pull-down to ensure that the hub stays in the default state if the pin control method is not used. Details of pin control are shown in Section 6.0, "FlexConnect Pin Control".

When FlexConnect is initiated, the current host port will change to a device port, and the designated port will change into a host port. This can be used for Host Swapping-type or Host Sharing-type applications.

2.2 Host Swapping

In a Host Swapping application, the role of host is exchanged between two dual-role capable devices. This host mode switching can be initiated through ID pin control, protocol handshake, or some other proprietary method. An example is shown in Figure 2.

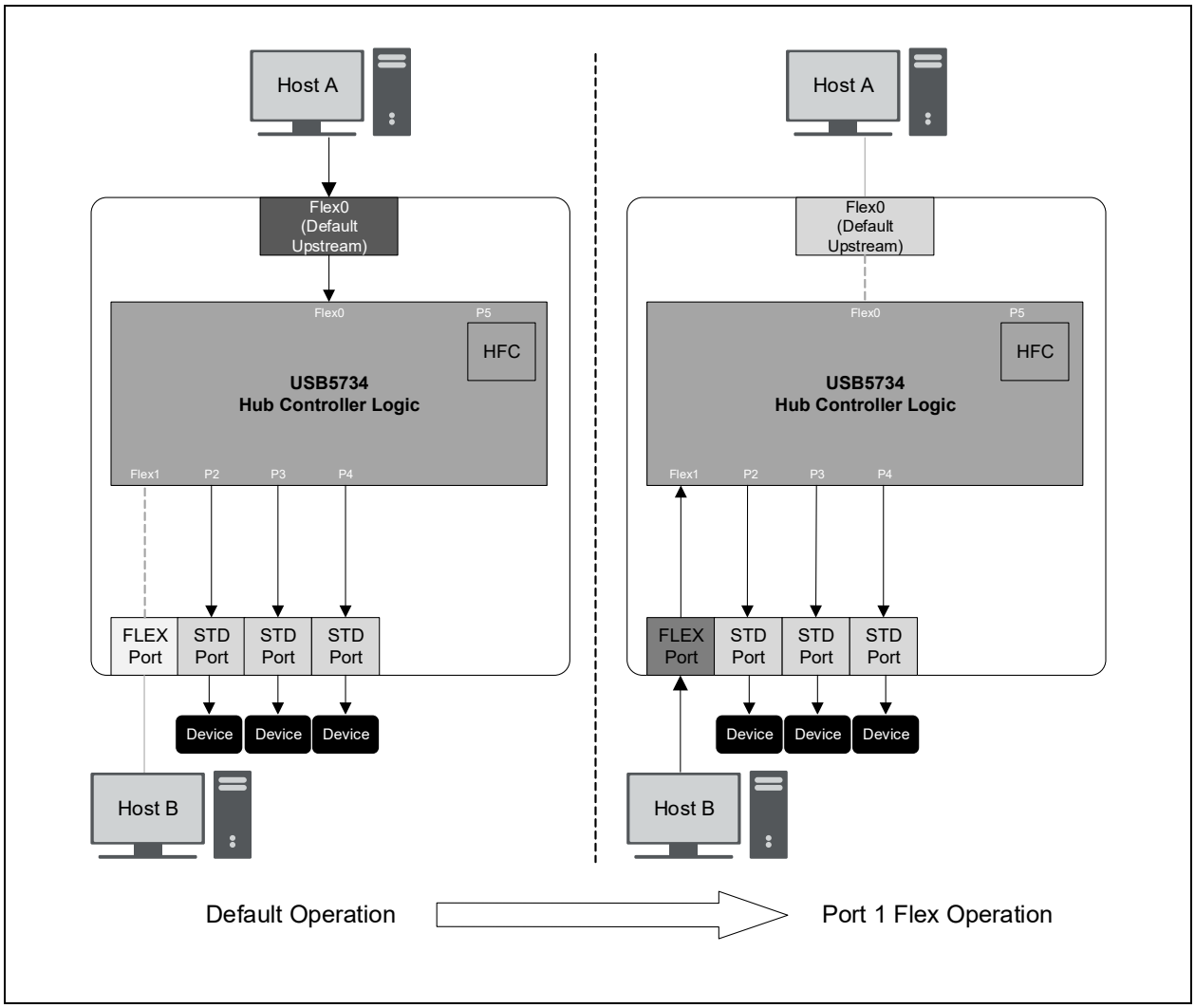
FIGURE 2: FLEXCONNECT HOST SWAPPING EXAMPLE



2.3 Host Sharing

In a Host Sharing application, multiple hosts are connected to the hub, and only one of these hosts has access to the USB device tree at a time. All other hosts are effectively disconnected from the USB tree when they do not control the host port. An example of Host Sharing is shown in Figure 3.

FIGURE 3: FLEXCONNECT HOST SHARING EXAMPLE



3.0 IMPORTANT NOTE ON FLEXCONNECT AND USB3 LINKS

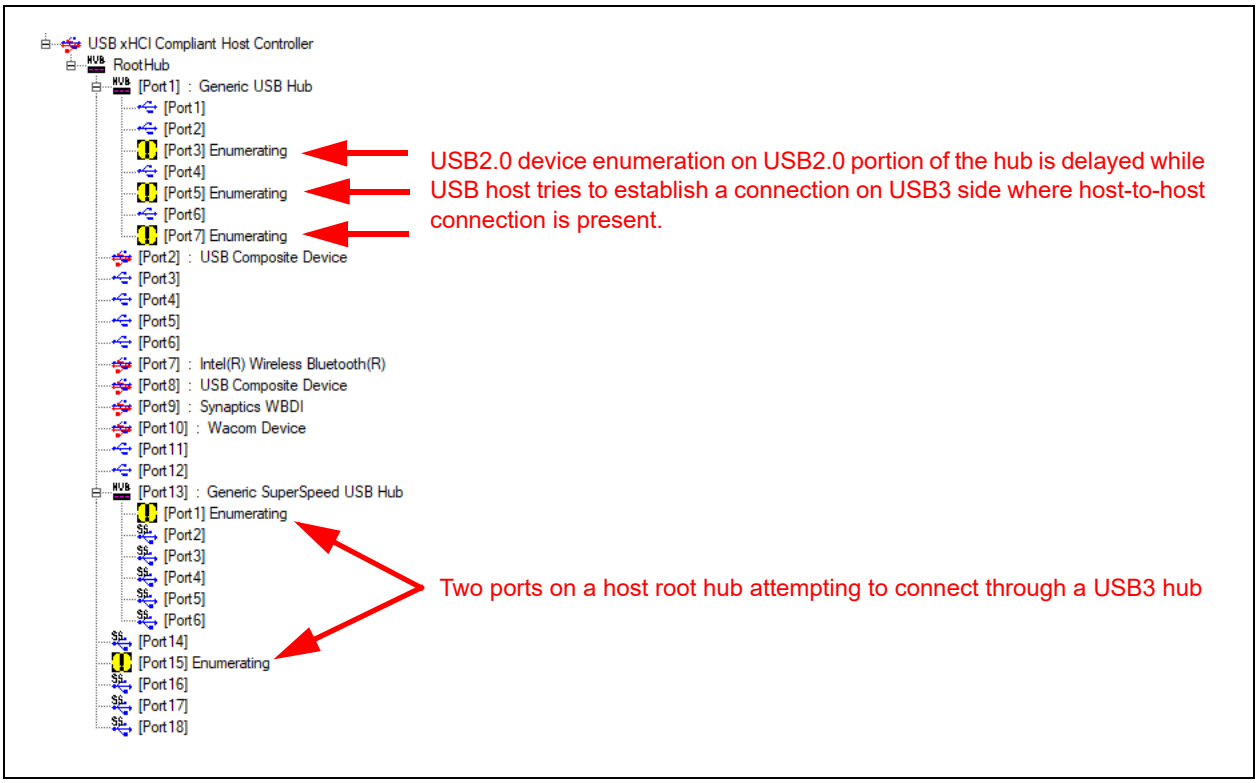
FlexConnect allows a USB3 hub to swap host roles between two dual-role capable devices or to share a hub and its devices. This creates a potential conflict for USB3 links when a system design has corner cases that allow two USB3 hosts to be connected together through a hub for any duration of time.

USB3 links are symmetrical, and the Link layer operates autonomously. The Link layer is not aware of the data direction; its job is to detect a Link partner, train the Link, and enter the U0 state where it waits for commands from the Protocol layer. When this occurs, the operating system USB driver is alerted that a link has been established and will attempt to communicate with the device (that is, a host). Since hosts will never respond to device requests, this may occupy host system resources and bandwidth for an extended amount of time while the host driver attempts to establish communication.

If other devices are connected to the hub, the enumeration of those devices may be significantly delayed while the host attempts to establish the invalid connection to the secondary host. The host may eventually time out and stop attempting to establish the connection. In many common operating systems, this may take 10 seconds or more.

In Windows® systems, a pop-up message may be received indicating a device malfunction. Using a USB tree viewing application, the problem may be visually observed as shown in Figure 4.

FIGURE 4: USB TREE VIEW OF TWO HOST PORTS CONNECTED THROUGH A HUB



To help prevent these kinds of issues, the following are recommended:

- *For Host Sharing applications:* When designing a Host Sharing application, always disable port 1 via One-Time Programmable (OTP) or SMBus configuration to prevent connection issues with the inactive host.
- *For Host Swapping applications:* It is recommended to always follow this sequence when changing the FlexConnect state:
  a) Place the hub into RESET or into Low-Power mode by setting **VBUS_DET** low.
  b) Make necessary mode changes to the system settings (see Note):
     - *Basic Type-C Ports:* Change state from DFP mode to UFP mode or vice versa.
     - *USB Power Delivery Ports:* Initiate data role swap and confirm acceptance of data role swap request.
     - *Embedded On-The-Go (OTG) Hosts/Devices:* Change the mode of operation from Host mode to Device mode, or vice versa.
  c) Ensure sufficient delay/debounce time is given for above system setting changes to take place and become stable.
  d) Reboot or reconfigure the hub with FlexConnect state change in place.

Note: Legacy Type-A/Type-B connectors are not recommended in Host Swapping applications.

4.0 FLEXCONNECT CONTROL VIA SMBUS

The FlexConnect feature can be controlled and configured via SMBus control. This method is useful for systems with an embedded microcontroller that can control the state of the FlexConnect feature via SMBus. FlexConnect can only be controlled via I^2C/SMBus target during the configuration stage. If the hub will be flexed after establishing an active connection with a host, the MCU/SoC controlling the I^2C/SMBus interface must first reset the hub, then make the necessary register modifications while the hub is in the configuration stage.

To initiate FlexConnect of the hub, set bit 0 of the CONNECT_CFG register. To terminate FlexConnect and revert to the default state, clear bit 0 of CONNECT_CFG.

TABLE 1: USB2 HUB FLEXCONNECT FEATURE CONTROL REGISTER

CONNECT_CFG (318Eh)			Connect Configuration Register
Bit	**Name**	**R/W**	**Description**
7:1	Reserved	R	Reserved (Always ‘0b’)
0	FLEXCONNECT	R/W	FlexConnect Control When asserted, the USB5734 changes its hub connections so that the physical port 1 becomes an upstream port, and the physical port 0 transitions from an upstream port to a downstream port. 0: Port 0 = upstream, port 1 = downstream 1: Port 0 = downstream, port 1 = upstream

5.0 FLEXCONNECT USB COMMAND DETAILS

There are two options for issuing the FlexConnect command:

- USB Command SET_ROLE_SWITCH to HFC
- VSM USB Command to Hub Control Endpoint

5.1 USB Command SET_ROLE_SWITCH to HFC

Generally, this method is most suitable for systems that wish to control FlexConnect via the Application layer.

The USB command is a NO DATA Control Transfer that must be issued to the Control Endpoint 0 (EP0) of the internal HFC device. On USB5734, the HFC is the internal device located on port 5. The SETUP command format is shown in Table 2.

TABLE 2: FLEXCONNECT CONNECT COMMAND SETUP PACKET

Setup Parameter	Value	Description
bmRequestType	0x41	Host-to-device, vendor class, targeted to interface
bRequest	0x90	SET_ROLE_SWITCH
wValue	0xYYYY	Bits detailed in Table 3
wIndex	0x0000	Reserved
wLength	00	No data

TABLE 3: WVALUE DETAIL OF FLEXCONNECT COMMAND SETUP PACKET

Bit	Name	Description
15	CMD_SELECT	Always ‘1b’
14:7	Reserved	Reserved (Always ‘0b’)
6	FLEX_STATE	1: Initiate FlexConnect to make Physical port 1 UFP/Physical port 0 DFP 0: Revert to the Unflexed state
5:0	Reserved	Reserved (Always ‘0b’)

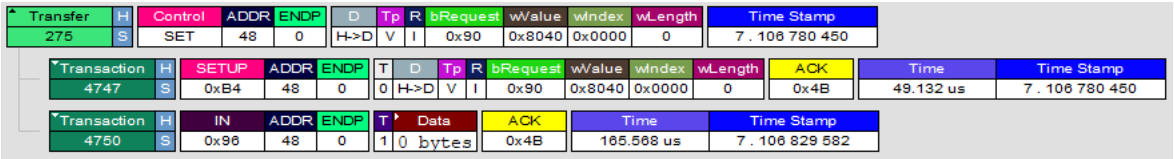
5.1.1 FLEXCONNECT USB COMMAND EXAMPLE

An example of a FlexConnect initialization command is shown in Table 4. This command is sent to EP0 of the HFC.

TABLE 4: FLEXCONNECT SETUP COMMAND EXAMPLE

Setup Parameter	Value	Note
bmRequestType	0x41	Host-to-device, vendor class, targeted to interface
bRequest	0x90	SET_ROLE_SWITCH
wValue	0x8040	Bit 15: 1b Bits 14:7: 00000000b (Reserved) Bit 6: 1b Bits 5:0: 000000b (Reserved)
wIndex	0x0000	Reserved
wLength	00	No data

FIGURE 5-1: FLEXCONNECT COMMAND SETUP TRANSACTION EXAMPLE



5.2 VSM USB Command to Hub Control Endpoint

Generally, this method is most suitable for systems that wish to control FlexConnect via the Kernel layer as many operating systems do not grant control of the hub Control Endpoints through the Application layer.

Note: The VSM commands can be prevented from being used within the system by setting the VSM_DISABLE bit in the HUB_CFG1 register.

The USB command is a control transfer with a two-byte data packet that must be issued to Control Endpoint (EP0) of the USB2 component of the USB hub.

5.2.1 VSM SET FEATURE

The following VSM command is used to set the FlexConnect feature and flex the port 1 as an upstream port.

TABLE 5: VSM SET FEATURE PACKET

Setup Packet	Value	Description
bmRequestType	0x40	Host-to-device, vendor class, targeted to device
bRequest	0x02	WRITE_MEMORY
wValue	0x8040	FlexConnect instructions as defined in Table 3
wIndex	0x0000	Reserved
wLength	2	Two-byte data bytes to write

TABLE 6: VSM SET FEATURE DATA PACKET

Offset	Value	Description
0	0x01	VSM sub-command—SET_FEATURE subcommand opcode
1	0x04	This sets the internal FlexConnect bit and forces the hub through a reset cycle. wValue is defined in FlexConnect wValue table (Table 3).

5.2.2 VSM CLEAR FEATURE

The following VSM command is used to clear the FlexConnect feature and unflex the hub back to port 0.

TABLE 7: VSM CLEAR FEATURE PACKET

Setup Packet	Value	Description
bmRequestType	0x40	Vendor-specific command with host-to-device data transfer
bRequest	0x02	WRITE_MEMORY
wValue	0x8000	FlexConnect instructions as defined in Table 3
wIndex	0x0000	Reserved
wLength	2	Two-byte data bytes to write

TABLE 8: VSM CLEAR FEATURE DATA PACKET

Offset	Value	Description
0	0x02	VSM sub-command—CLEAR_FEATURE sub-command opcode
1	0x04	This clears the internal FlexConnect bit and forces the hub through a reset cycle and reverts the USB host role back to port 0.

6.0 FLEXCONNECT PIN CONTROL

The FlexConnect feature can be initiated via direct pin control of **FLEXCMD** for systems that have an embedded MCU/SoC, which controls the state of FlexConnect.

The **FLEXCMD** pin operates as follows:

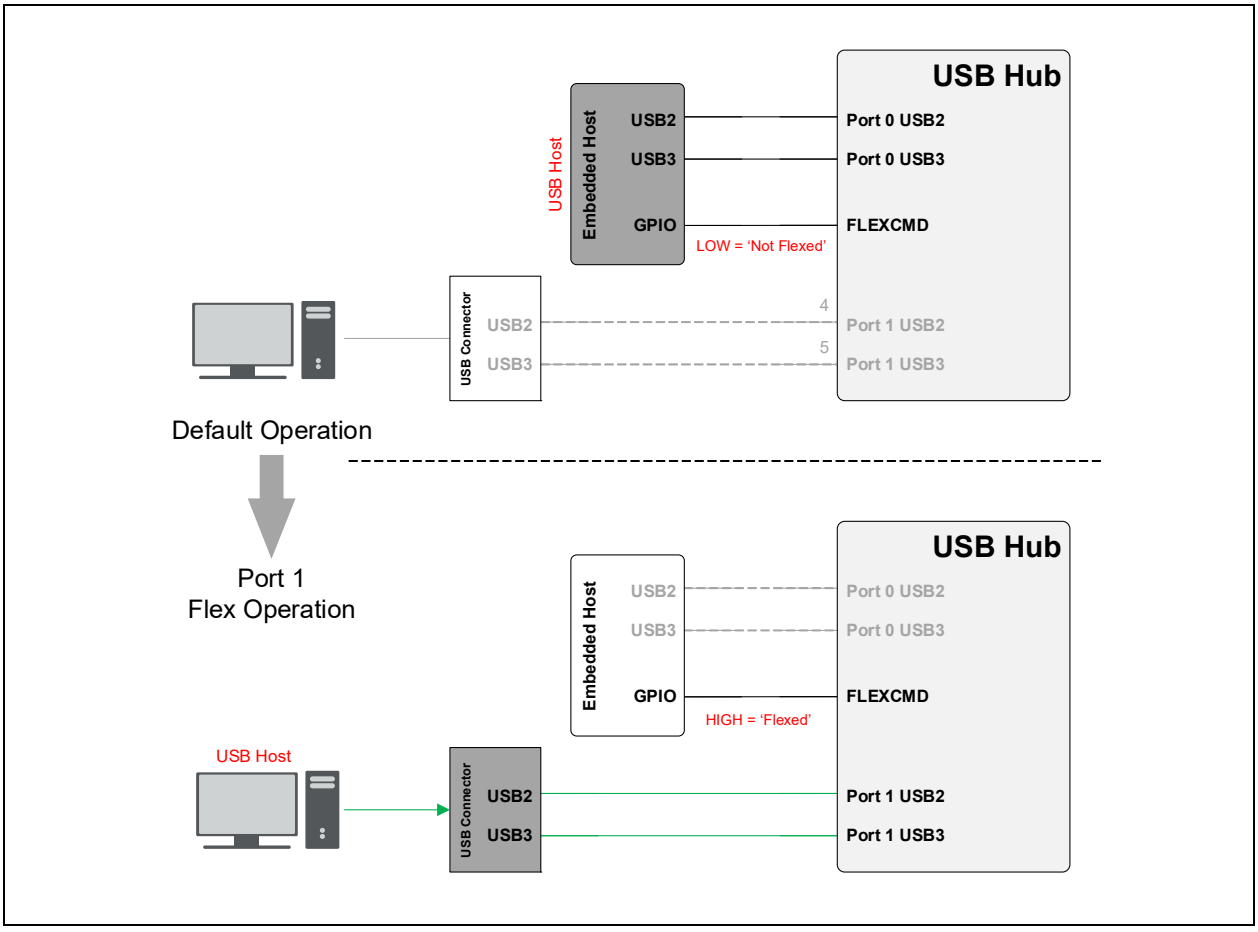
- When a 0 to 1 state transition is detected, the hub resets and reboots into the Flexed state.
- When a 1 to 0 state transition is detected, the hub resets and reboots into the Unflexed state.
- An internal 50 kΩ pull-down is always attached to the pin.
- A input debounce prevents noise/glitches from inadvertently triggering FlexConnect.

Note: The **FLEXCMD** pin is only available in CFG_STRAP Configuration mode 1 or 2.

When FlexConnect is initiated, the hub is reset and rebooted. If the SMBus configuration channel is being used, the hub will wait indefinitely to be reconfigured by the SMBus controller before it enters runtime. Thus, when using the Pin Control method for FlexConnect, hub configuration through pin strap options and/or OTP memory is recommended.

An example of how an embedded USB host may use the direct pin control method is shown in Figure 5.

FIGURE 5: FLEXCONNECT PIN CONTROL EXAMPLE



Note: The SMBus and USB Command methods both have specific conditions where the hub will automatically unflex and return to the default port 0 host state. When using the direct pin control method, the hub will always obey the state of the pins and will never attempt to automatically unflex.

7.0 HUB VBUS DETECTION

VBUS is a signal used by the hub to determine whether a USB host is connected. When VBUS presence is sensed by the hub, the hub will signal to that host that it is present and communication may begin.

When VBUS is not present, the hub enters a low-power state to conserve energy. If a VBUS high-to-low transition is sensed during an active connection, all communication will cease immediately and the hub will disconnect from the host. If this occurs while the hub is in the Flexed state, the hub will automatically unflex and revert the host role to port 0, unless the hub is in the FlexConnect Direct Pin Control mode.

There are two options for VBUS detection:

- *Via VBUS_DET pin*: This is a basic digital 3.3V input. This pin is used when the port is configured as a Type-B port and only a simple digital high or digital low is required for sensing VBUS.
- *Manual override via SMBus during runtime*: The VBUS detection functions can be overridden by manually controlling PIO32 and PIO24 registers (that is, set PIO as output and drive high or low) via SMBus during runtime after the USB Attach command is issued. Both PIO32 and PIO24 or internal-only signals have no connection outside of the hub device. To manually override the VBUS detection, perform the following register writes:
  - VBUS_DET_PASS_THRU [3C40h] = 00h
  - GPIO_24_31_DIR [0830h] = 01h to set internal PIO24 as output
  - GPIO_32_39_DIR [0933h] = 01h to set internal PIO32 as output
  - GPIO_24_31_OUT [0834h] = 01h to set internal PIO24 as drive high
  - GPIO_32_39_OUT [0937h] = 01h to set internal PIO32 as drive high

TABLE 9: VBUS_DET_PASS_THRU CONTROL REGISTER

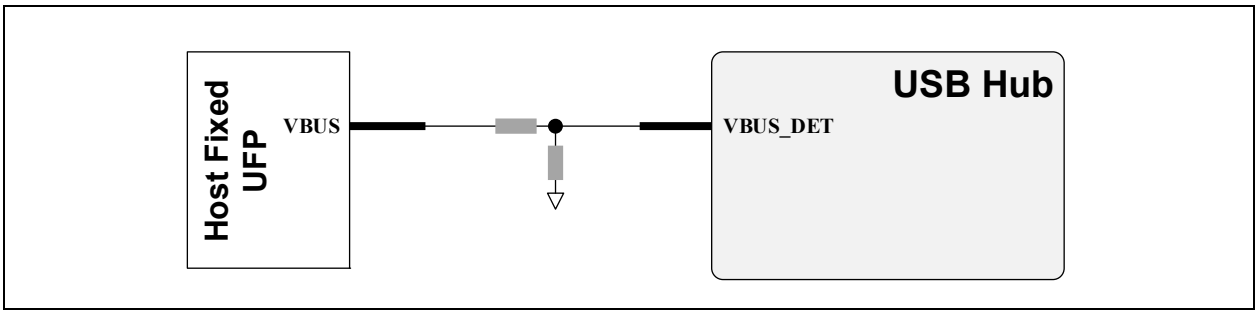
VBUS_DET_PASS_THRU (3C40h)			VBUS_DET Pass-Through Register Default = 03h
Bit	**Name**	**R/W**	**Description**
7:2	Reserved	R	Always 0
1	USB3_PASS_THRU	R/W	0: VBUS detection to the USB3 hub comes from the internal PIO24 (this option is only through SMBus during runtime). 1: VBUS detection to the USB3 hub comes from the **VBUS_DET** pin.
0	USB2_PASS_THRU	R/W	0: VBUS detection to the USB2 hub comes from the internal PIO32 (this option is only through SMBus during runtime). 1: VBUS detection to the USB2 hub comes from the **VBUS_DET** pin.

7.1 No Power Role Swap VBUS_DET Implementation

If power roles do not change when initiating FlexConnect, it may be appropriate to simply connect **VBUS_DET** directly to the default USB host's **VBUS** pin through a resistor divider. This is the standard **VBUS_DET** implementation, as shown in Figure 6. This is a popular implementation in the automotive application space as the head unit will always be connected to the hub.

Note: Ensure that the default USB host will not toggle its VBUS supply at any time when FlexConnect is initiated. Otherwise, the hub will be reset and revert to its default state.

FIGURE 6: NO POWER ROLE SWAP VBUS_DET CONFIGURATION



7.2 VBUS_DET Connected Directly to a Fixed 3.3V Supply

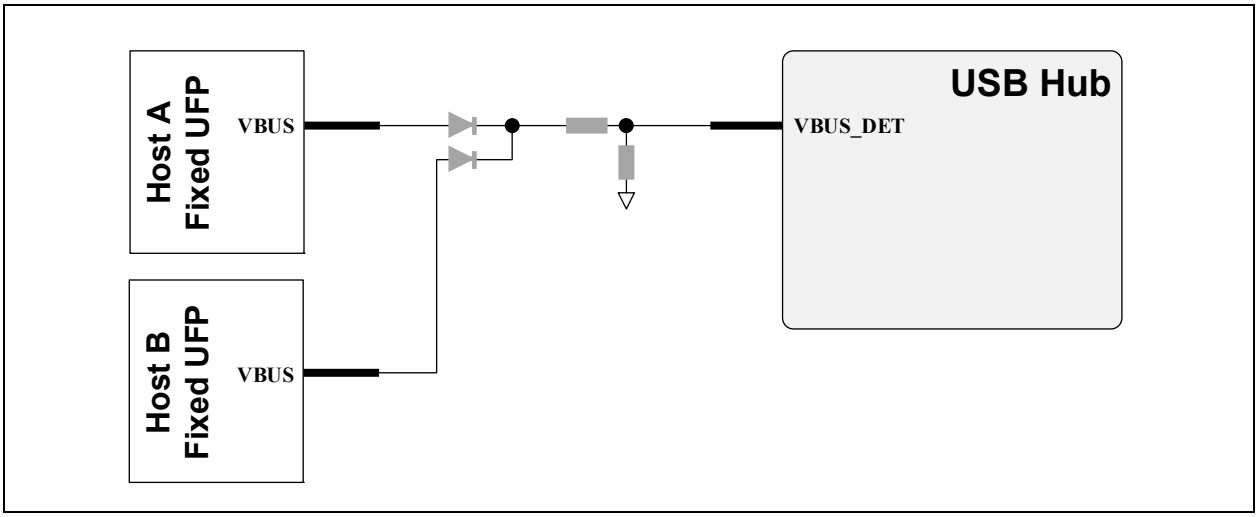
If low-power states are not required when no USB host is present, it is acceptable to connect VBUS_DET to a fixed 3.3V supply on the PCB. This method is only recommended for designs where the hub is connected to an embedded host IC.

Note: In some instances, a USB host may attempt to force a hard reset on a device by toggling VBUS. In this instance, the hub will not reset.

7.3 'OR' All Host VBUS Connections to VBUS_DET

In Host Sharing type systems, 'OR'ing all host VBUS connections is an appropriate solution as shown in Figure 7. This can be used when the "one host at a time" implementation is used. The drawback with this type of solution is that there is no way to distinguish which host port is actually supplying VBUS.

FIGURE 7: DIODE-OR VBUS_DET SIGNALS



Note: The resistor divider values will change depending on the characteristics of the diodes used.

8.0 ADDITIONAL HARDWARE AND DESIGN CONSIDERATIONS

8.1 PRT_CTL1

The **PRT_CTL1** pin operates as shown in Table 10.

TABLE 10: PRT_CTL1 CONTROL REGISTER

Preconfiguration Settings	Default State (Unflexed State)	Flexed State
Default register settings: • USB3_PRT_CFG_SEL1 [3C00h] = 83h • OCS_SEL1 [3C20h] = 01h	When port 1 is enabled, this pin is set as an input and is pulled-high internally, which allows the power to the port and monitoring for an active-low overcurrent signal assertion from an external current monitor. This pin will change to an output and be driven low when an overcurrent event is detected (that is, the pin is detected as low while in Input mode) or the port is disabled by the host controller (via USB Command).	
Optional alternate settings: (port power disabled) • USB3_PRT_CFG_SEL1 [3C00h] = 80h • OCS_SEL1 [3C20h] = 00h	The **PRT_CTL1** pin operates as GPIO17. GPIO_16_20_PD [082Dh]: Bit 1 0: Internal 50 kΩ pull-down disabled 1: Internal 50 kΩ pull-down enabled (recommended to set only if GPIO_16_20_DIR is set as Input mode) GPIO_16_20_DIR [0831h]: Bit 1 0: Input 1: Output GPIO_16_20_OUT [0835h]: Bit 1 0: Output Low (valid only if GPIO_16_20_DIR is set as Output mode) 1: Output High (valid only if GPIO_16_20_DIR is set as Output mode) GPIO_16_20_IN [083Dh]: Bit 1 0: Input Low (valid only if GPIO_16_20_DIR is set as Input mode) 1: Input High (valid only if GPIO_16_20_DIR is set as Input mode) GPIO_16_20_PU [083Dh]: Bit 1 0: Internal 50 kΩ pull-up to 3.3V disabled 1: Internal 50 kΩ pull-up to 3.3V enabled (recommended to set only if GPIO_16_20_DIR is set as Input mode)	The **PRT_CTL1** pin operates in a fixed, drive-low state.

Note: See the *AN1903* application note for more details and full register bit field definitions.

8.2 FlexConnect State Indicator Output

In Configuration 2, **PROG_FUNC3** and **PROG_FUNC7** reflect the current state of **FLEXCMD**. These outputs can be connected to external LEDs or an embedded microcontroller to communicate the FlexConnect state.

9.0 FLEXCONNECT APPLICATIONS AND COMMON USE-CASE EXAMPLES

The FlexConnect feature can be implemented with a very large number of permutations. The successful implementation of a FlexConnect-based design requires deep knowledge of USB specification requirements, or otherwise direct leveraging of an existing design implementing a known FlexConnect use case. (The automotive smartphone integration implementation is an example of a known use case with reference designs available.)

Each permutation has special hardware considerations that must be properly handled. As such, there is no 'one size, fits all' hardware design. A cross-section of all possible permutations is offered in examples within this section. The components of the given examples allows any user to achieve their desired results, although mixing and matching sections from multiple examples may be necessary to assemble the final solution. To give an idea of the number of possible permutations, see Table 11.

The following terms are used to define the overall architecture and are defined as:

- *Host Sharing*: Two hosts or more host ports. When any given host port is the active host port, all other host ports are effectively inactive (that is, not sourcing power and not operating as a device DFP port). This method is similar to a KVM switch-type functionality.
- *Host Role Swapping*: Two or dual-role host and device ports. When any given host port is the active host port, all other host-capable ports operate as device ports (that is, sourcing power and allowing a device to connect to the port).
- *Automotive Smartphone Integration*: The automotive smartphone integration use case is a known use-case, but limited. In this architecture, FlexConnect may be initiated via any control method, but both ports involved in FlexConnect make no changes to their power role.

TABLE 11: COMMON DESIGN USE-CASES FOR A TWO-PORT FLEX IMPLEMENTATION

#	Flex Architecture	Control Method	Port 0 Type	Port 1 Type	Example/Note
1	Host Sharing	USB	Type-B	Type-B	Example 1
2	Host Swapping	Pin Control	Type-A	Embedded OTG (Host and Device) Controller	Example 2

9.1 Example 1: USB Controlled 'KVM-Type' Application with Two Type-B Connectors

EXAMPLE 1: USB CONTROLLED 'KVM-TYPE' APPLICATION WITH TWO TYPE-B CONNECTORS

Architecture: Host Sharing

Control: USB Control

P0 Port Type/Role: Type-B

P1 Port Type/Role: Type-B

Example Application: KVM Switch Use-Case

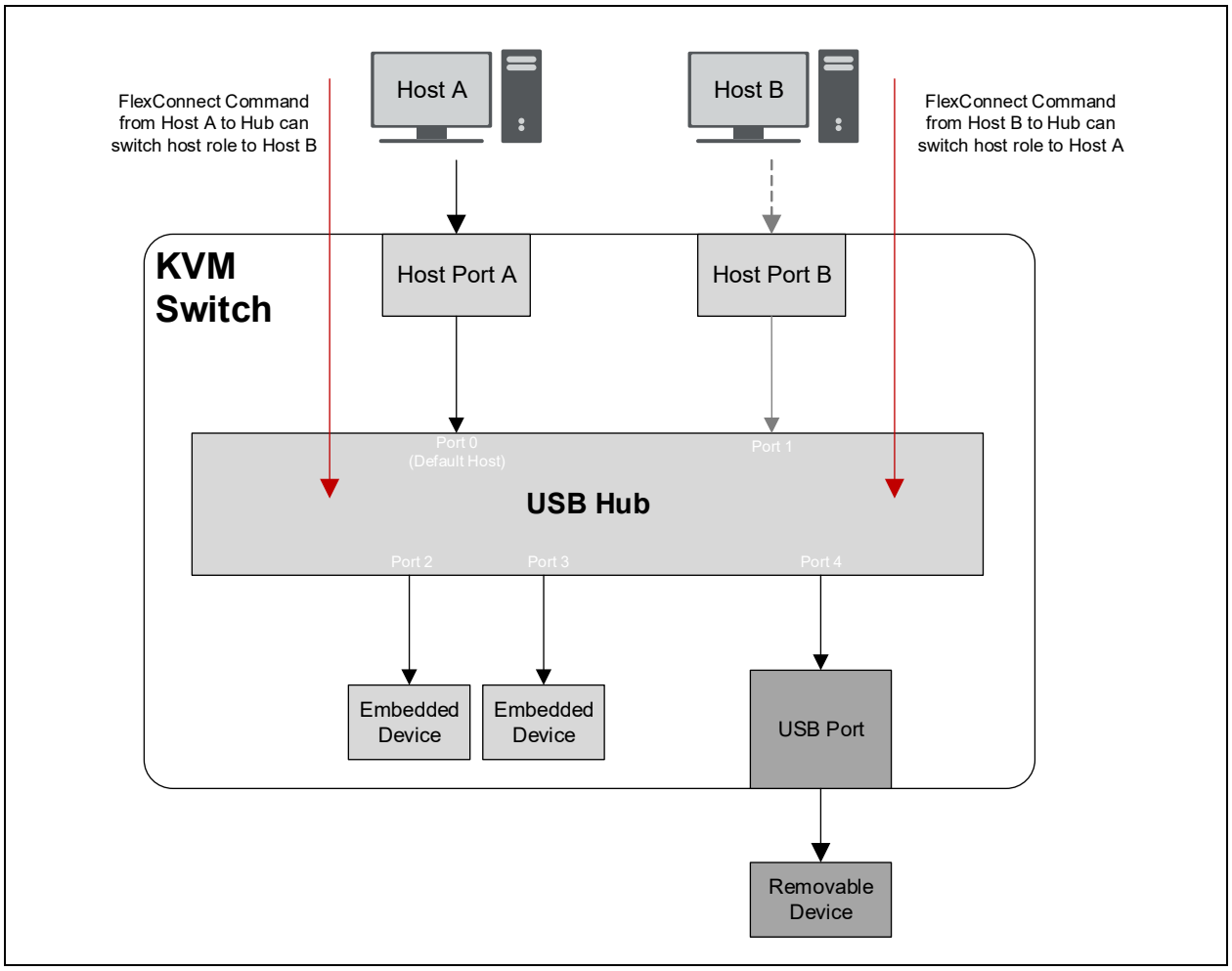
In this example, two hosts can share a common set of USB devices by switching between hosts, and the two equivalent hosts are referred to as 'Host A' and 'Host B'.

Switching from Host A to Host B requires a vendor-specific USB command originating from Host A to the hub.

Switching from Host B to Host A requires a vendor-specific USB command originating from Host B to the hub.

Every time FlexConnect is initiated in the hub, all downstream devices are reset and their addresses are cleared. Full enumeration and configuration must occur for the hub and all devices for each switchover. It is not possible to initiate FlexConnect without clearing the device addresses and full re-enumeration.

FIGURE 8: EXAMPLE 1 SYSTEM BLOCK DIAGRAM



9.1.1 EXAMPLE 1 KEY HARDWARE AND SOFTWARE DESIGN CONSIDERATIONS

- *Upstream Connectors*: Either USB3 Type-B or USB3 Micro-B can be used. Note that the USB3 Micro-B's ID pin cannot be used to control FlexConnect in this example as the method for controlling FlexConnect is via USB Command.
- *VBUS Detection*:
  - Diode-OR (Recommended): The recommended implementation is to diode-OR the two **VBUS** pins from both Type-B upstream connectors. This is recommended because it requires no additional configuration to achieve. Be sure to modify the resistor divider network used to account for the selected diode voltage drop. With this implementation, both Host A and Host B will need to turn off VBUS in order for the hub to lose the VBUS connection and soft-reset/enter low-power state.
  - Force VBUS_DET high in OTP settings: It is possible to force the VBUS Detection function as always asserted within hub register settings, and the **VBUS_DET** pin can be left unused. This requires some initial OTP programming of the hub, which cannot be achieved via USB as **VBUS_DET** is required to be sensed as high for the initial USB connection to take place. Hence, OTP programming through I^2C is the viable option if using this method.
  - Fixed connection to 3.3V: **VBUS_DET** can be pulled directly to a system 3.3V power rail. This is not recommended as it eliminates the host's ability to force a hard reset to the hub during an error recovery scenario.
  - Connect VBUS_DET to only one Type-B connector: If it can be assured that one of the hosts will always supply VBUS during operation, **VBUS_DET** may be connected to only one of the Type-B connectors.
- *Software*: Both Host A and Host B will need to have software that is capable of issuing the FlexConnect command

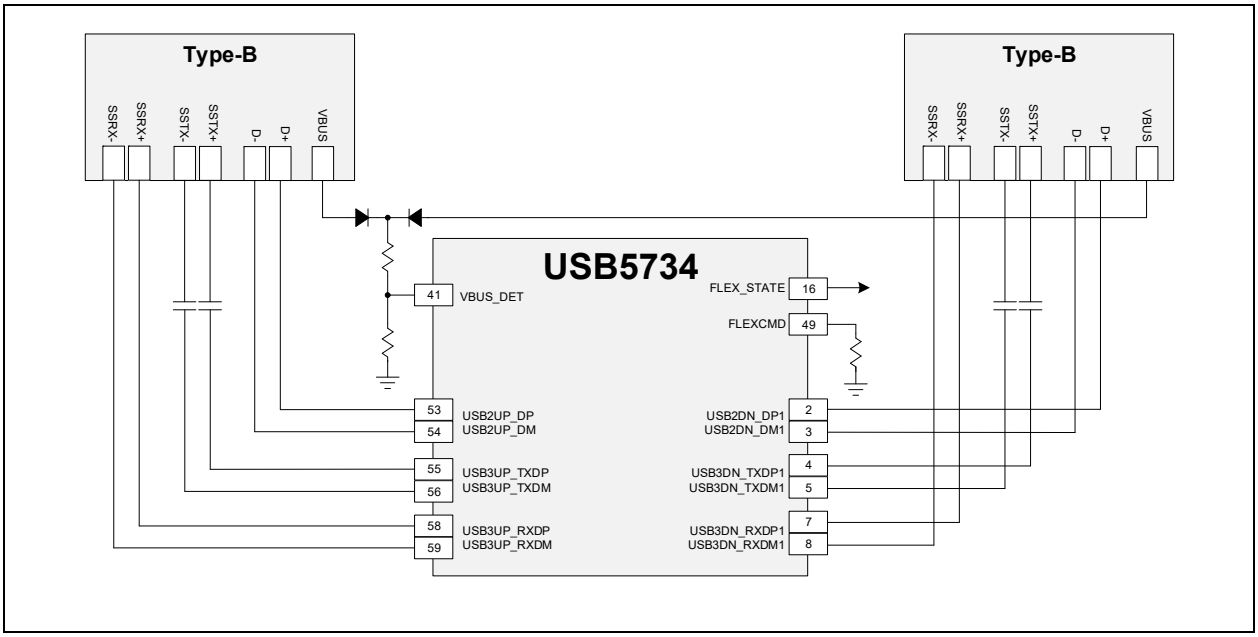
to the hub. Additionally, there should be some interconnection between the two hosts to coordinate the switching and to prevent switching during an important data transfer. There are two options for FlexConnect commands:

- Command issued to internal HFC device: This method uses the generic USB device class (referred to WinUSB in Windows®). This is generally only accessible from the Application layer in many major operating systems. This command is defined in Section 5.1, USB Command SET_ROLE_SWITCH to HFC.
- Command issued to the hub Control Endpoint: This method uses the hub drivers to issue a VSM command for controlling FlexConnect. This is generally only accessible from the Kernel layer in many major operating systems. This command is defined in Section 5.2, VSM USB Command to Hub Control Endpoint.

9.1.2 EXAMPLE 1 SIMPLIFIED SYSTEM SCHEMATICS

Figure 9 shows a simplified schematic of Example 1. For any of the hardware design considerations discussed above with multiple options, the recommended option is selected.

FIGURE 9: EXAMPLE 1 SIMPLIFIED SCHEMATIC DIAGRAM



9.1.3 EXAMPLE 1 FLEXCONNECT INITIATION AND TERMINATION

To initiate the FlexConnect sequence to switch host role from port 0 (Host A) to port 1 (Host B), the following USB command must be issued to the USB hub. In this example, the command format for FlexConnect command issued to the HFC is used.

Note: The user decides the value of ENUM_TIMEOUT. This value configures a timer that is set by the hub after flexing. If the host on port 1 (Host B) does not enumerate the hub before the timer expires, the hub will revert the FlexConnect state back to port 0 (Host A).

TABLE 12: EXAMPLE 1 FLEXCONNECT COMMAND TO HFC FOR PORT 0 TO PORT 1 SWITCH

Setup Parameter	Value	Description
bmRequestType	0x41	Device-to-host, vendor class, targeted to interface
bRequest	0x90	SET_ROLE_SWITCH
wValue	0x8040	[15] CMD_SELECT: 1b (always 1b for FlexConnect) [14:7] Reserved: 0000_0000b [6] FLEX_STATE: 1b (Flexed) [5:0] Reserved: 00_0000b
wIndex	0x0000	Reserved
wLength	00	No data

After the USB host role has latched to port 1 (Host B) following successful enumeration of the hub by Host B, the USB host role may be reverted back to port 0 (Host A) in one of two ways:

- By physically disconnecting/shutting off Host B
- By issuing another USB FlexConnect command

The USB FlexConnect command for unflexing is formatted as in Table 13:

TABLE 13: EXAMPLE 1 FLEXCONNECT COMMAND TO HFC FOR PORT 1 TO PORT 0 SWITCH

Setup Parameter	Value	Description
bmRequestType	0x41	Device-to-host, vendor class, targeted to interface
bRequest	0x90	SET_ROLE_SWITCH
wValue	0x8000	[15] CMD_SELECT: 1b (always 1b for FlexConnect) [14:7] Reserved: 0000_0000b [6] FLEX_STATE: 0b (Unflexed) [5:0] Reserved: 00_0000b
wIndex	0x0000	Reserved
wLength	00	No data

9.2 Example 2: Embedded Host Requiring Programming as Device During Manufacturing

EXAMPLE 2: EMBEDDED HOST REQUIRING PROGRAMMING AS DEVICE DURING MANUFACTURING

Architecture: Host Swapping

Control: FLEXCMD Pin Control

P0 Port Type/Role: Basic Type-A

P1 Port Type/Role: Embedded

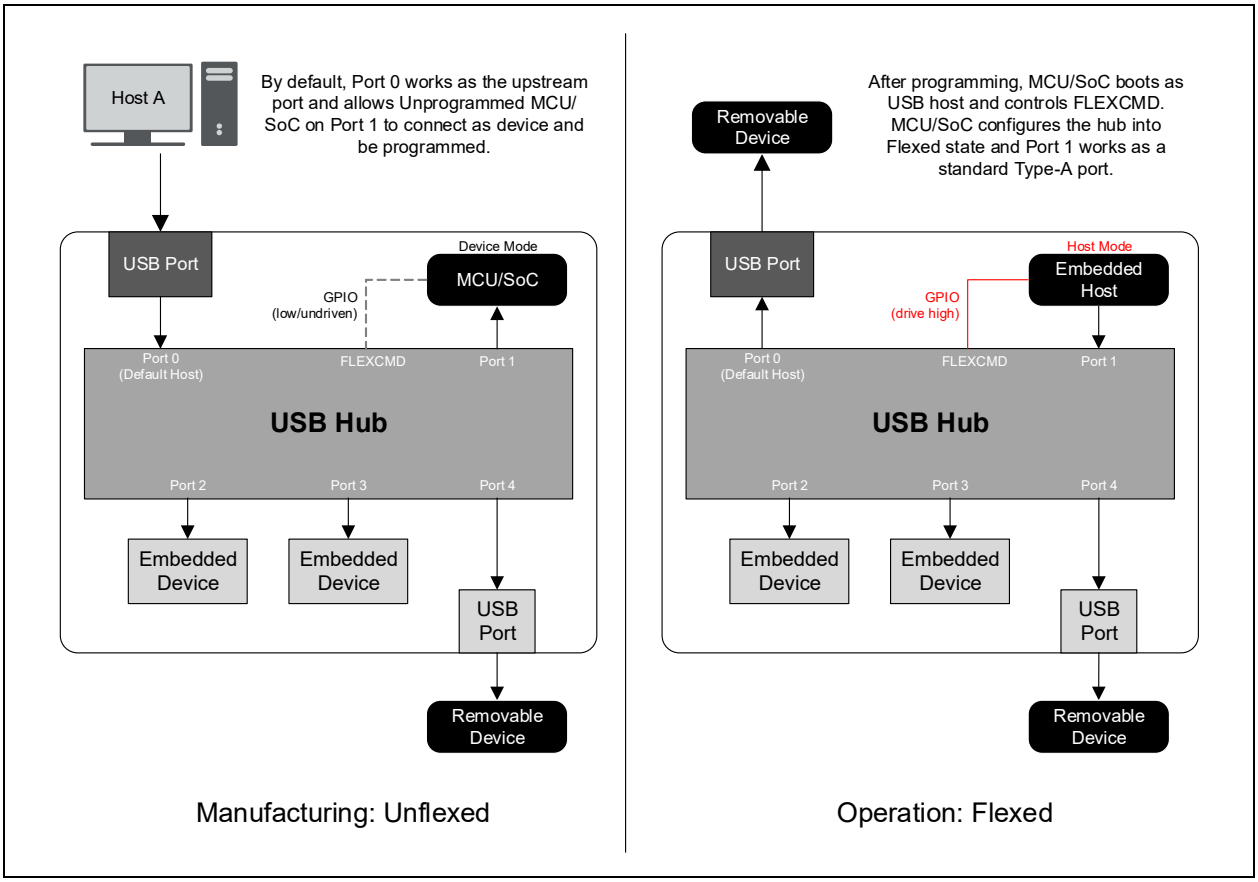
Example Application: Embedded host that requires programming as device during manufacturing

In this example, an embedded host processor is populated on a PCB but is not programmed by default. The processor can connect in USB Device mode and be programmed for the final application.

The system is designed such that the USB can operate with its default settings to allow an external USB host to connect to port 0 and access the MCU/SoC.

After the MCU/SoC is programmed, the system is reset and reinitialized by the MCU/SoC. The MCU/SoC boots and sets the **FLEXCMD** pin high to force the hub to enter the Flexed state.

FIGURE 10: EXAMPLE 2 SYSTEM BLOCK DIAGRAM



9.2.1 EXAMPLE 2 KEY HARDWARE AND SOFTWARE DESIGN CONSIDERATIONS

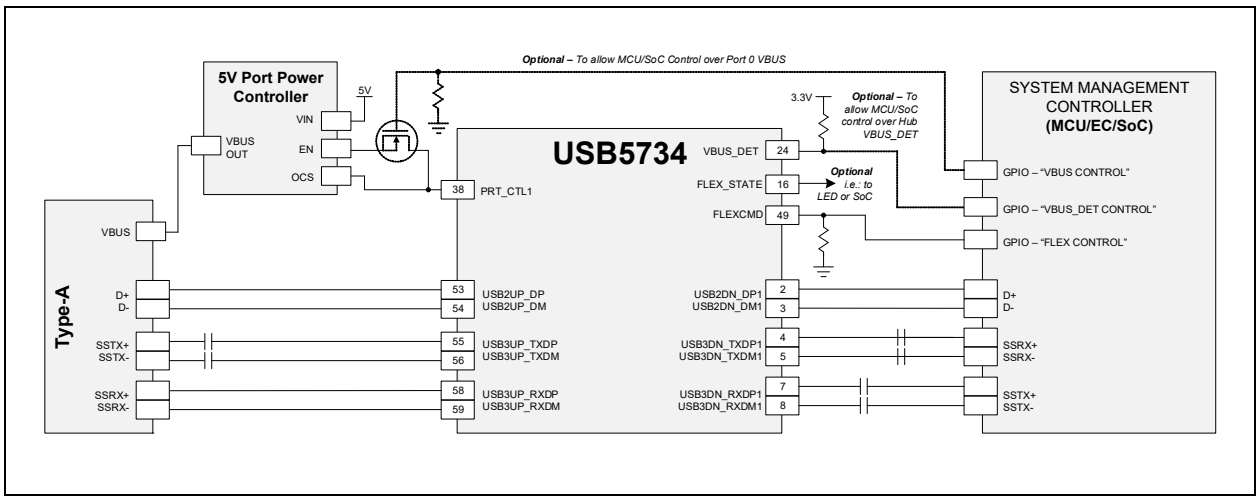
- *Port 0 Connector*: In this example, a USB Type-A connector is connected to port 0 of the hub. For the manufacturing environment, the host will need to connect to the hub and MCU via a special Type-A to Type-A USB cable. A Type-A to Type-A cable is not a permissible cable type that is defined by any USB specification, but this is not a concern since this mode of operation is reserved for the manufacturing environment only.
- *VBUS Detection*: In this example, port 0 only operates as the host port during manufacturing. All other times, the MCU/SoC is an embedded host. Since this is a fixed, permanent connection, the recommended handling is to tie **VBUS_DET** to 3.3V through a weak pull-up resistor, and to optionally connect a spare MCU/SoC GPIO directly to the hub **VBUS_DET** pin to allow the MCU/SoC to pull the signal low in order to force the hub into a low-power state as needed.
- *Port 0 Type-A Port Power Control*: After the MCU/SoC is programmed and becomes the USB host, physical port 0 operates as logical port 1 of the hub and devices may attach to the port and be enumerated by the MCU/SoC. To prevent VBUS back-drive during the manufacturing stage, it is recommended to implement a hardware design that blocks VBUS from being sourced on the Type-A port until the MCU/SoC is programmed and asserts a GPIO to enable VBUS to the downstream port.

Note: To save on I/O count, this signal may be sourced from the same GPIO that controls the **FLEXCMD** pin if there is no use case to forcibly disable VBUS on physical port 0/logical port 1 while the hub is in the Flexed state.

9.2.2 EXAMPLE 2 SIMPLIFIED SYSTEM SCHEMATICS

Figure 11 shows a simplified schematic of Example 2. For any of the hardware design considerations discussed above with multiple options, the recommended option is selected.

FIGURE 11: EXAMPLE 2 SIMPLIFIED SCHEMATIC DIAGRAM



9.2.3 EXAMPLE 2 FLEXCONNECT INITIATION AND TERMINATION

To initiate FlexConnect, the **FLEXCMD** pin must be driven from low to high.

To terminate FlexConnect, the **FLEXCMD** pin must be driven from high to low.

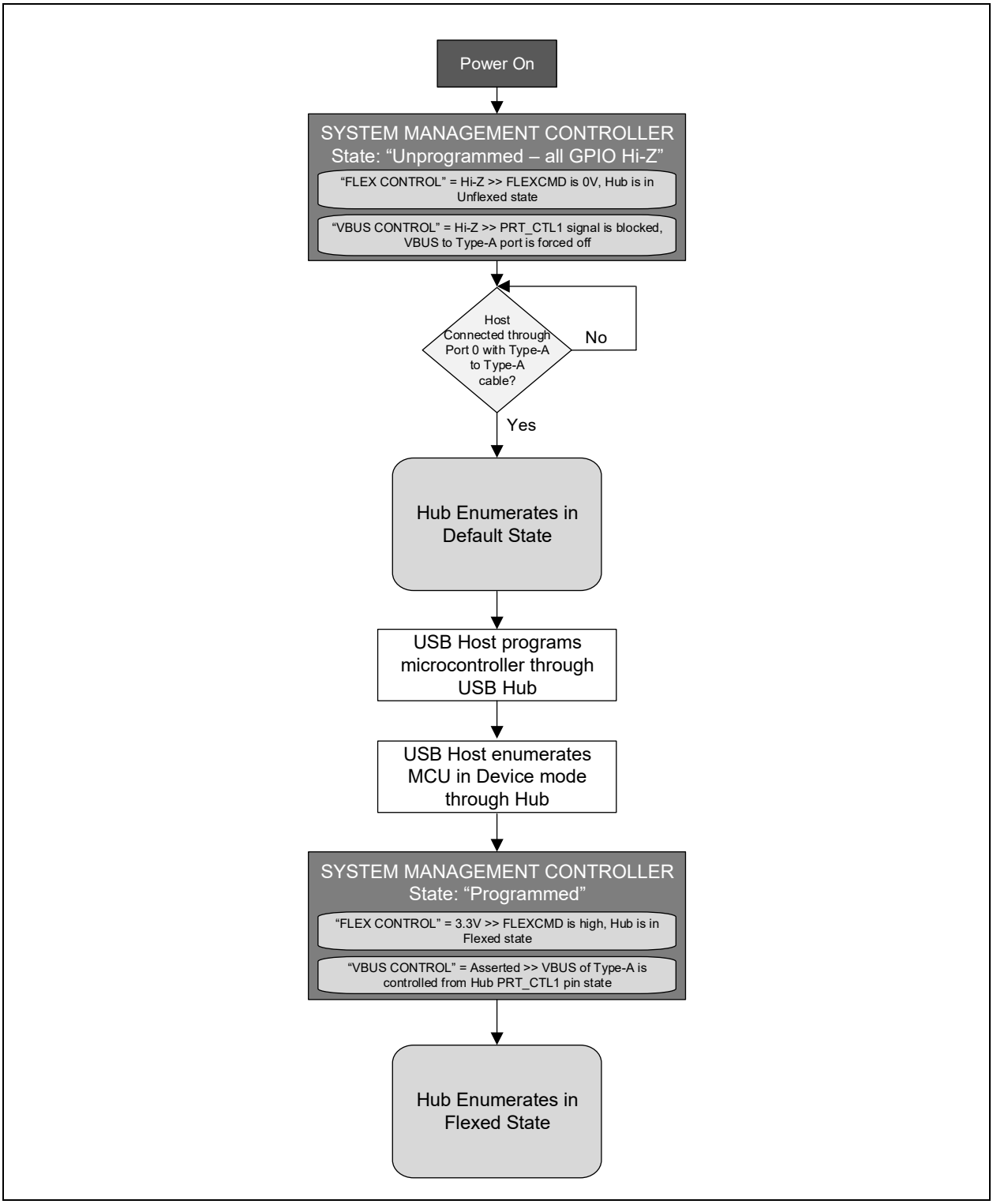
Some additional considerations to note:

- **VBUS_DET** must be detected as high for the hub to connect to a USB host. This holds true regardless of the Flex state of the hub.
- If the SMBus configuration channel of the hub is being used within the system, it is not recommended to control FlexConnect via the **FLEXCMD** pin. Instead, the SMBus Control method is recommended. The reason is that the hub experiences a chip reset when initiating or terminating FlexConnect. If the SMBus configuration channel is present (that is, the hub is configured in a mode that has the SMBus configuration channel enabled, and pull-up resistors are detected), the hub will wait in the configuration stage every time the hub is reset. The configuration stage is only exited when the special USB Attach SMBus command is received. This means that the hub will wait in the configuration stage for the special USB Attach command every time FlexConnect is initiated or terminated from the **FLEXCMD** pin state change.

9.2.4 EXAMPLE 2 BEHAVIORAL DIAGRAM

System behavior for this example may be as shown in Figure 11:

FIGURE 12: EXAMPLE 2 BEHAVIOR DIAGRAM



APPENDIX A: APPLICATION NOTE REVISION HISTORY

Revision Level & Date	Section/Figure/Entry	Correction
DS00004767A (10-11-22)	Initial release	

NOTES:

THE MICROCHIP WEB SITE

Microchip provides online support via our WWW site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://microchip.com/support

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable" Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

ISBN: 978-1-6683-1385-5

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4485-5910
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-72400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7288-4388

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820