AN1212

Using USB Keyboard with an Embedded Host

| Author: | Amardeep Gupta |
| | Microchip Technology Inc. |

INTRODUCTION

Microcontroller applications can easily support USB embedded host functionality with the introduction of Microchip's microcontrollers with the USB OTG peripheral. Traditionally, personal computers have been used as hosts in a USB network. Microchip's USB OTG product line can now replace the personal computer in the system by implementing an embedded host.

Many embedded applications require control inputs from external devices, such as keyboards, mice, joysticks, Point-of-Sale (POS) barcode scanners and magnetic code readers having HID Keyboard Emulation mode. This application note demonstrates how to develop a USB keyboard application that can run on the Explorer 16 demo board with a USB PICtail™ Plus daughter board using a Microchip USB OTG microcontroller as the embedded host.

USB Keyboard Overview

USB is now the preferred method to interface peripherals with computers. The conventional PS2 keyboard has been replaced with a standard USB keyboard to interface with the computer (host). A standard USB keyboard sends an 8-byte input report to the host. Table 1 describes the keyboard input report (8 bytes).

TABLE 1:   USB KEYBOARD INPUT REPORTS

Byte	Description
0	Modifier Keys
1	Reserved
2	Key Code 1
3	Key Code 2
4	Key Code 3
5	Key Code 4
6	Key Code 5
7	Key Code 6

For some keyboards that have a custom implementation that does not match a standard keyboard implementation, a look-up table may be required in order to map the key code returned to the desired usage ID. Keys, such as Ctrl, Shift, Alt and GUI keys make up the 8-bit modifier byte in a standard keyboard report. Byte 1 of this report is a constant. This byte is reserved for the use of the Original Equipment Manufacturer (OEM).

Table 2 describes the keyboard output report (1 byte).

TABLE 2:   USB KEYBOARD OUTPUT REPORT

Bit	Description
0	Num Lock
1	Caps Lock
2	Scroll Lock
3	Compose
4	KANA
5 to 7	Constant

A standard keyboard has three LEDs to display Num Lock, Caps Lock and Scroll Lock status. The LEDs are absolute output items; the state of each LED must be included in the output reports (0 = OFF, 1 = ON). In the keyboard demo application, Num Lock and Caps Lock LEDs have been implemented.

USB Keyboard with an Embedded Host

This application demo illustrates the Human Interface Device (HID) host capability. A low-speed or full-speed USB keyboard can be connected to the host MCU on an Explorer 16 board through a USB PICtail™ Plus daughter board. The demo schedules the HID transfers and interprets the reports received from the keyboard. Key codes are decoded to their respective ASCII values and are displayed on the LCD display on the Explorer 16 demo board.

Since the purpose of this application is to demonstrate the capabilities of the HID host, not all the key codes have been implemented. This demo helps the user to understand the HID host capabilities. Users can incorporate the necessary changes required for their application. All of the alphabetic, numeric and special characters have been implemented, as well as the Esc, Shift, Caps Lock, Arrow and Spacebar keys.

> **Note:** Refer to the "Device Class Definition for Human Interface Devices (HID)" for keyboard implementation details (see the **"References"** section).
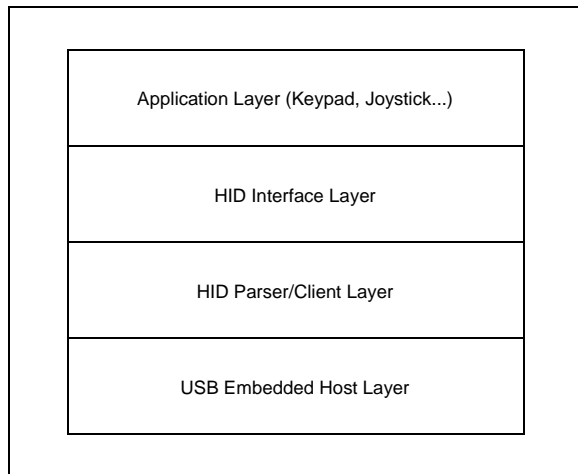
AN1212

Application Architecture

The keyboard application is actually a multi-layer stack (see Figure 1) with different components of Microchip's USB embedded host support package contributing to different layers.

Table 3 lists the source files used in this application and which layer those files implement.

FIGURE 1:   APPLICATION ARCHITECTURE

| Application Layer (Keypad, Joystick...) |
| HID Interface Layer |
| HID Parser/Client Layer |
| USB Embedded Host Layer |

Getting Started with the USB Host Stack

This section describes the various steps involved in getting started with the USB host stack.

The USB keyboard application is available as part of Microchip's complete USB embedded host support package (see **Appendix A: "Software Discussed in this Application Note"** for more details).

INSTALLING THE USB HOST STACK

To install all the required project files on a host PC:

1. Download the installation file from the Microchip web site:

    http://microchip.com/usb

2. Run the executable installer file.

    By default, the project and stack files will be installed in the directory structure displayed in Figure 2.

FIGURE 2:   DEFAULT DIRECTORY STRUCTURE FOR USB KEYBOARD HOST SUPPORT
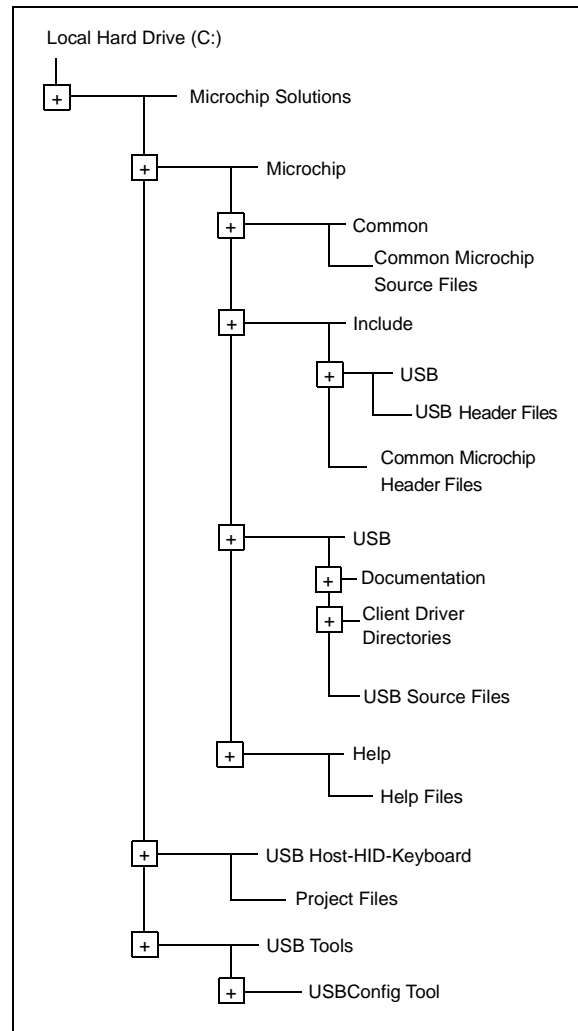
USB EMBEDDED HOST LAYER

The USB embedded host layer provides basic USB embedded host support. The interface to this layer is provided automatically in the HID client driver. For more information about this layer, refer to Microchip's *AN1140, "USB Embedded Host Stack"* and *AN1141, "USB Embedded Host Stack Programmers Guide"*.

It is not necessary to be familiar with this layer's operation or configuration in order to use it with the keyboard application.

HID CLIENT DRIVER/HID PARSER

The next layer provides the client driver for the Human Interface HID (Class) and HID parser. The HID client is needed for interfacing to HID devices, like mice, keyboards, joystick controls, etc. Refer to Microchip Application Note *AN1144,* "*USB Human Interface Device Class on an Embedded Host*" for more information about the HID client driver.

Note:	Memory allocation for descriptor related information is dynamic. Report descriptor data structures consume less than 300 bytes per interface. The application should reserve at least 1 Kbyte of heap while using the HID host stack.

HID INTERFACE LAYER

The interface layer is provided to integrate the user application with the HID client/HID parser. This layer provides interface functions for the application to inter-act with the HID client/HID parser. HID client interface functions can be used to exchange reports with devices attached on the bus. HID parser interface functions can be used to interpret the report descriptor and decode the input report received from the device. Refer to *AN1144,* "*USB Human Interface Device Class on an Embedded Host*" for more information on the HID parser.

Note:	For detailed information about the USB embedded host HID interface APIs, refer to the USB Embedded Host Library documentation provided in the Help directory.

TABLE 3: FILES USED FOR KEYBOARD APPLICATION

Layer	File Name	Description
USB Embedded Host Layer	usb_host.c	Provides USB embedded host support for all devices. Does not provide class support.
	usb_host.h	Header file with definitions required for USB embedded hosts. Defines the interface to the USB embedded host driver.
	usb.h, usb_ch9.h, usb_common.h, usb_hal.h, usb_hal_pic24.h	Other USB support header files.
HID Client Driver/Parser	usb_host_hid.c	Provides HID class support to a USB embedded host.
	usb_host_hid.h	Header file with definitions for USB embedded hosts supporting the HID class. Defines the interfaces to the HID client driver.
	usb_host_hid_parser.c	Provides a HID report descriptor parser and data structures.
	usb_host_hid_parser.h	Header file with interface functions to report descriptor parser and definitions supporting the data structures.
HID Interface	usb_host_hid_appl_interface.c	Provides Interface functions to access the HID client/parser.
	usb_host_hid_appl_interface.h	Header file containing interface definitions used to access the HID client/parser.
Application	uart2.c	Provides an interface to UART2 to provide RS-232 input and output to the application.
	uart2.h	Header file for the UART2 functions.
	usb_config.c	Configures the USB stack for the application; generated by the configuration tool.
	usb_config.h	Configures the USB stack for the application; generated by the configuration tool.
	HardwareProfile.h	Contains system level constants for libraries to reference.
	lcd_demo.c	Provides support functions for the LCD driver and display routines.
	lcd_demo.h	Header file containing interface definitions for the LCD.
	keyboard_demo.c	Main application code.

Application Functionality

The USB keyboard application demo is intended to demonstrate the features of the Microchip HID host. The application can interface to a USB keyboard with the Explorer 16 demo board using the USB PICtail Plus daughter board. The user might have to modify the TPL settings depending on the number of interfaces the keyboard supports. The main features of the keyboard demo are:

- Parsed Data Collection Handler
- Receive Input Reports
- Transmit Output Reports
- LCD Display Function

Parsed Data Collection Handler

An HID device must have at least one report descriptor. Report descriptors are composed of pieces of information. Each piece of information is called an item. The HID client driver incorporates an HID parser. The HID class client driver contains a parser used to analyze items found in the report descriptor. The parser extracts information from the descriptor in a linear fashion. The parser collects the state of each item known as it walks through the descriptor and stores it in a structure in the form of a table. Refer to Microchip's *AN1144,* "*USB Human Interface Device (HID) Class on an Embedded Host*" for more information about the HID client driver.

The HID client driver generates an event, EVENT_HID_RPT_DESC_PARSED, after a report descriptor has been parsed without an error. The application must provide a callback function to collect the parsed information stored in the data structures. The configuration tool has a provision to enter the function's name. In this keyboard application demo, the `USB_HID_DataCollectionHandler()` function collects the parsed information and stores it in the structure, HID_DATA_DETAILS.

Table 4 lists the structures provided.

Receive Input Reports

The host application receives an 8-byte input report from the device. Timer3 is used to trigger the send request for the input report. Timer3 triggers the request depending on the polling rate defined by the configuration descriptor. The polling rate for the input report is stored in the structure, USB_HID_DEVICE_RPT_INFO, and accessed using the interface function, `USBHostHID_GetCurrentReportInfo()`.

The interface function, `USBHostHID_ApiGetReport()`, is used to request the input report from the device. The raw input report received from the device is stored in the `Appl_raw_report_buffer` structure. It contains information of all the keys pressed in that duration. The raw report is then passed to the `USBHostHID_ApiImportData()` interface function. Then, the data is extracted and the application buffers are populated.

The application then processes the data based on the key pressed. Two shadow buffers, `Appl_ShadowBuffer1` and `Appl_ShadowBuffer2`, are used to keep track of the keys pressed in the previous two reports. Data from the current buffer is copied into the shadow buffers after processing the current report buffer and the data in the current report buffer and raw report buffers are cleared.

Table 5 lists the input report buffers declared in the application.

Transmit Output Reports

Common USB keyboards have LEDs to display Caps Lock, Num Lock and Scroll Lock status. The LED status is sent by the host application as an output report to the device. The number of LEDs supported by the keyboard and the format of the report are described in the report descriptor. As described in the previous section, the application should collect the report format details after the report descriptor has been parsed. Depending on the key combinations pressed, the application can decide the status of the LED indicator and store it in the output report buffer.

The rate of sending output reports is not fixed; they are aperiodic transfers. The user application can decide to transmit the report whenever a change in the LED status occurs.

Table 6 lists the output report buffer declared in the application.

> **Note:** For detailed information about the USB embedded host HID interface APIs, refer to the USB Embedded Host Library documentation provided in the Help directory.

LCD Display Function

Upon power-up, the following banner displays on the LCD display:

```
                Device Detached
```

Upon connecting a keyboard, if the device is enumerated without any errors, the following banner displays on the LCD display:

```
            Explorer16 Board
            USB HIDHost Demo
```

The keyboard is now ready for use. When the first key press is received, the application clears the banner and starts displaying keys as they are pressed.

The input report received from the device consists of the usage IDs of the keys pressed within that period. All the alphabetic and special characters are converted to their respective ASCII values. The characters are displayed on the LCD on the Explorer 16 board.

Table 7 displays the action taken on the key codes received from the device.

TABLE 4: HID_DATA_DETAILS OBJECTS

Object	Description
Appl_ModifierKeysDetails	This structure stores the key details for the modifier keys (Ctrl, Shift, Alt and GUI).
Appl_NormalKeysDetails	This structure stores the input key details for all the other keys.
Appl_LED_Indicator	This structure stores the LED indicator output report details.

TABLE 5: INPUT REPORT BUFFER

Buffer	Description
Appl_raw_report_buffer	This buffer stores the report details and a pointer to the raw input report data received from the device.
Appl_BufferNormalKeys	This buffer stores the pressed instance data for the rest of the keys on the keyboard.
Appl_BufferModifierKeys	This buffer stores the pressed instance data for the modifier keys (Ctrl, Shift, Alt and GUI) for the current report received from the device.

TABLE 6: TRANSMIT OUTPUT REPORT BUFFER

Buffer	Description
Appl_led_report_buffer	This buffer stores the indicator status for the Caps Lock, Num Lock and Scroll Lock LEDs.

TABLE 7:   ACTION ON KEY CODES RECEIVED FROM THE DEVICE

Key	Pressed	Action
Characters	ON	All the letters and special characters are displayed on the LCD.
Shift	ON	If the Caps Lock is on, letters are displayed in lower case; otherwise, they are displayed in upper case. Special characters are displayed instead of numbers.
	OFF	If the Caps Lock is off, letters are displayed in upper case; otherwise, they are displayed in lower case. Numbers are displayed.
Esc	ON	LCD display is cleared.
Space	ON	Advances the cursor on the LCD by one space.
Back Space	ON	Character on the left of the cursor is deleted.
Caps Lock	ON	Letters are displayed in upper case. Output report with Caps Lock LED status high is sent.
	OFF	Letters are displayed in lower case. Output report with Caps Lock LED status low is sent.
Num Lock	ON	Output report with Num Lock LED status high is sent. Enables numeric keys on the keypad.
	OFF	Output report with Caps Lock LED status low is sent. Enables Arrow keys on the keypad.
Arrow Keys	ON	Moves the cursor on the LCD display.

THE USB CONFIGURATION TOOL

The USB keyboard application has already been configured by using the graphic USB configuration tool, `USBConfig.exe`, located in the installation directory, `\USB Tools\USBConfig Tool`. The configuration files (`usb_config.c` and `usb_config.h`) were generated and stored in the `.\USB Host - HID - Keyboard` project directory. The following configuration options have been selected:

1. On the Main tab (Figure 3):
   - Device Type: USB Embedded Host
   - Ping-Pong Mode: All Endpoints
2. On the Host tab (Figure 4):
   - Transfer Types: Interrupt only, with 50 NAKs allowed (control transfers are also enabled with dialog text appearing in grey)
   - Attach Debounce Interval: 250 ms
   - Name of Application Event Handler: `USB_ApplicationEventHandler`

> **Note 1:** The Attach Debounce Interval is increased from the USB specification of 100 ms to allow for slower devices.
>
> **2:** To conserve program and data memory, transfer events are not used.
>
> **3:** The number of allowed NAKs depends on the Idle rate of the device attached.

3. At the TPL tab (Figure 5):
   - TPL Entry 1
     - Support via Class ID is selected
     - Client Driver: HID
     - Class: HID (0x03)
     - SubClass: Boot (0x01)
     - Protocol: Keyboard (0x01)
     - Initial Configuration: 0
     - Initialization Flags: 0
     - HNP: Not Allowed

- TPL Entry 2
  - Support via Class ID is selected
  - Client Driver: HID
  - Class: HID (0x03)
  - SubClass: None (0x00)
  - Protocol: None (0x00)
  - Initial Configuration: 0
  - Initialization Flags: 0
  - HNP: Not Allowed

> **Note:** This demo application supports only standard keyboard interfaces. If the user keyboard supports multimedia functions, then the TPL entries need to be changed to support new interfaces.

4. On the HID tab (see Figure 6):
   a. The HID Client used in Host mode is to be selected.
   b. Input reports can be of varied lengths. While configuring, the host application needs to inform the client of the maximum data field size in bits. As illustrated in Example 1, for this input report, the user must enter 14 as the "Maximum Data Field Size".
   c. Under "Application Interface", "Default Interface" is selected. The "Parsed Data Collection Handler" is provided by the application.

> **Note:** If the host application is aware of the input/output report format, then the parsed data collection handler can be ignored and the field can be disabled by deselecting the check box.

EXAMPLE 1: DATA FIELDS FOR 4-BYTE INPUT REPORT
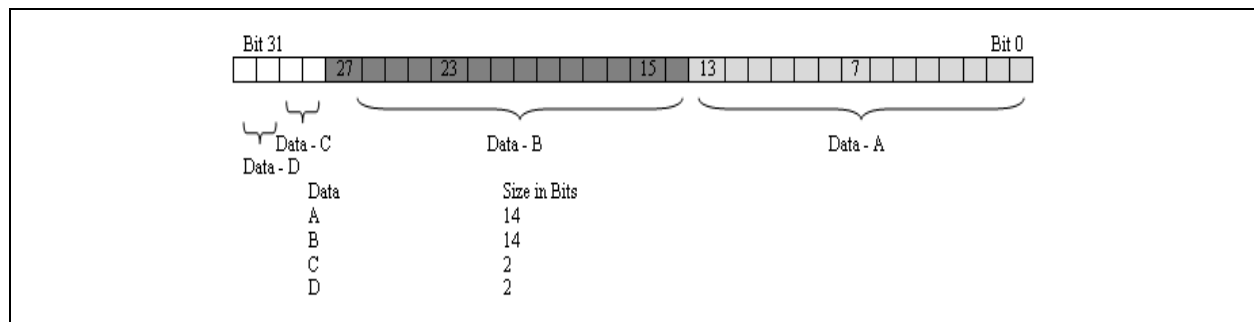
AN1212

FIGURE 3: USB CONFIGURATION TOOL, MAIN TAB
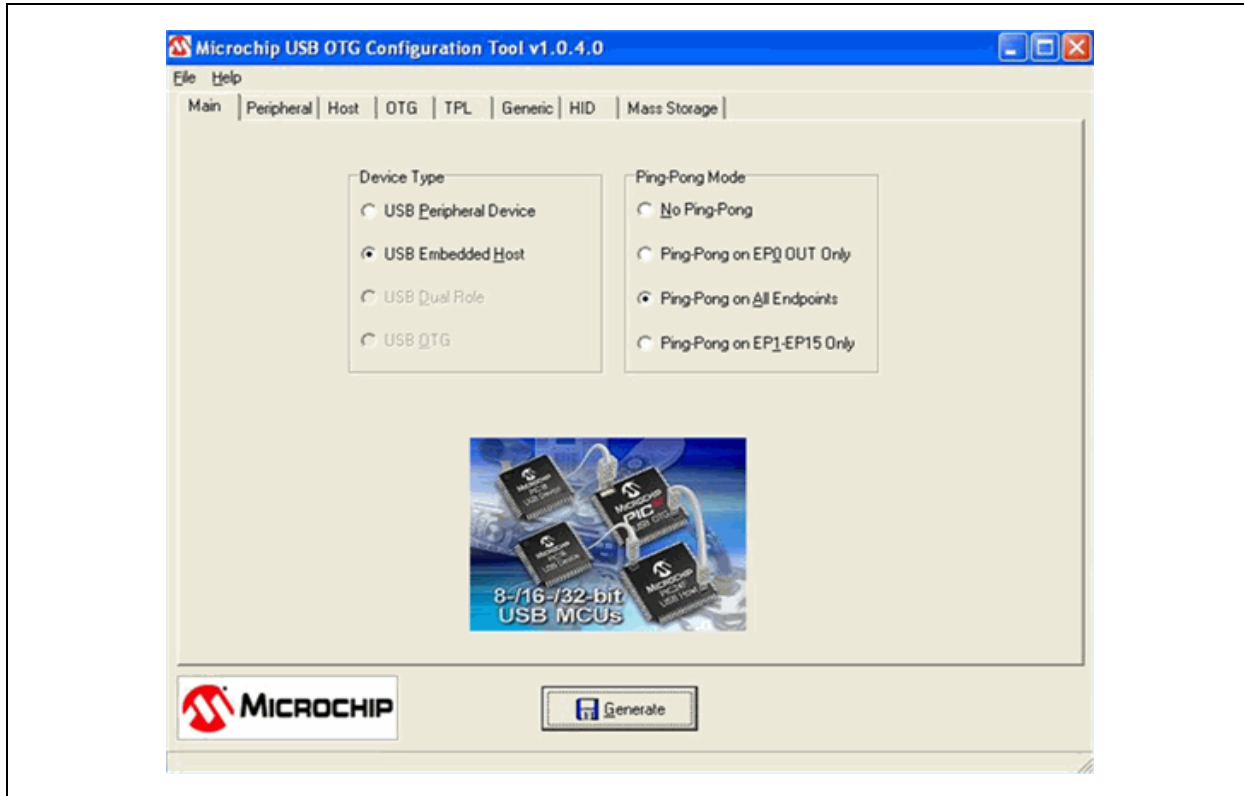
FIGURE 4: USB CONFIGURATION TOOL, HOST TAB
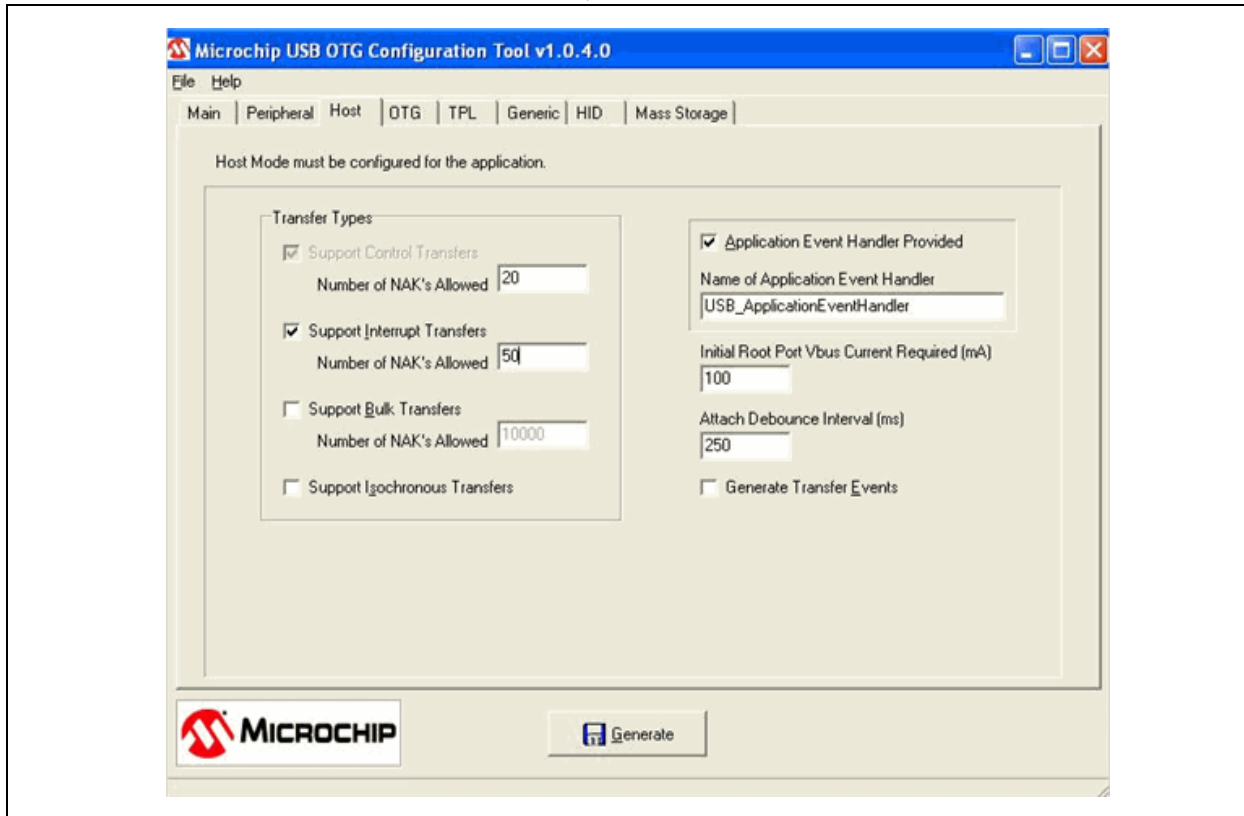
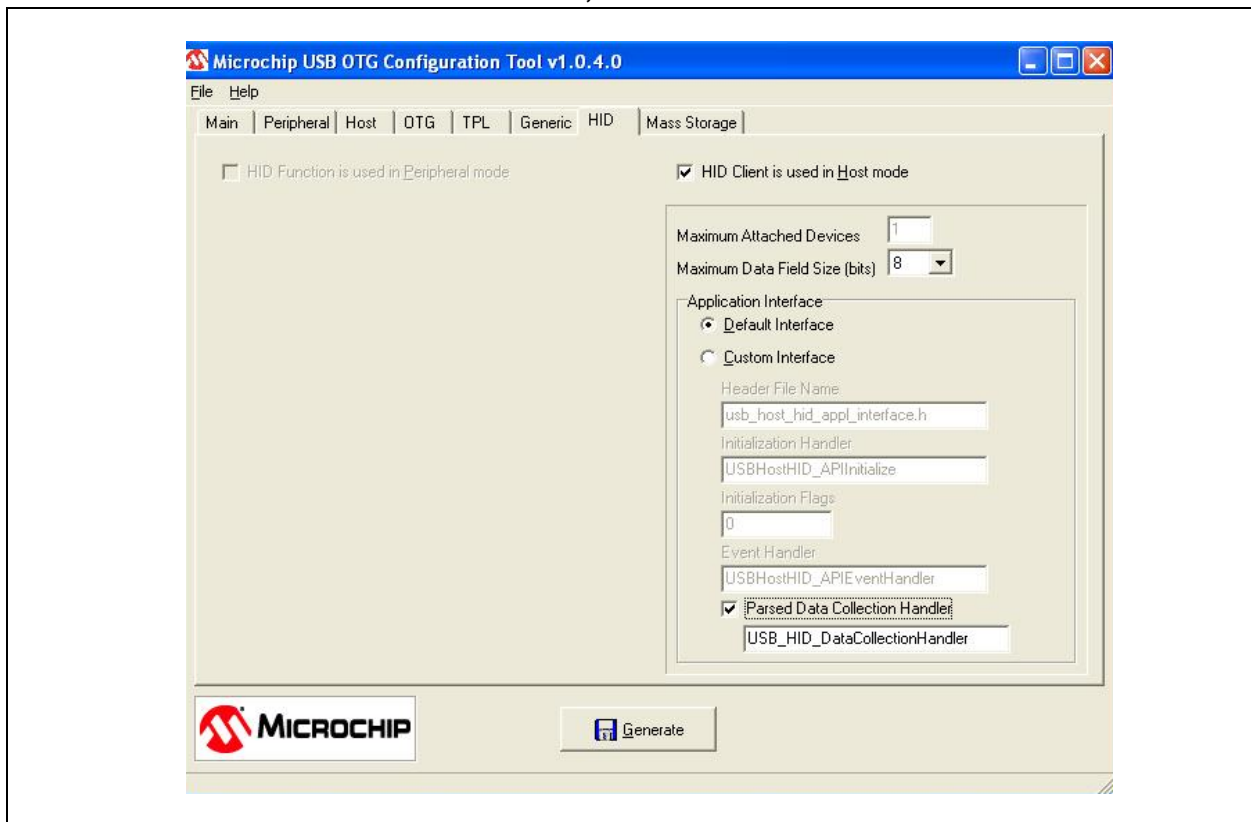**FIGURE 5:** USB CONFIGURATION TOOL, TPL TAB
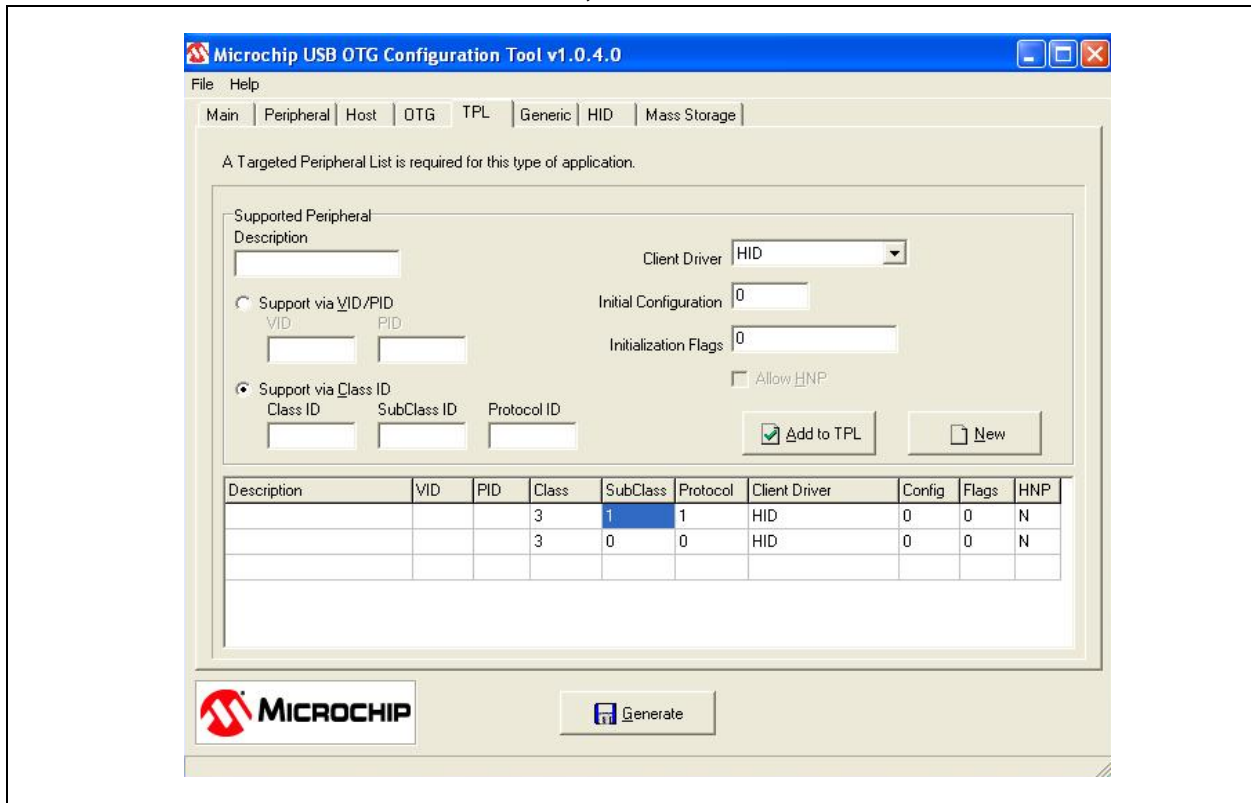
FIGURE 6: USB CONFIGURATION TOOL, HID TAB

AN1212

CONCLUSION

Standard input devices, like keyboards, mice and joysticks, intended to be used with a PC, can be easily interfaced with an embedded system.

Using Microchip's USB embedded host capability with the HID client driver, embedded applications can connect user interface devices to an embedded system.

REFERENCES

For more information on components of the Microchip USB embedded host support package, the following documents are available at the Microchip web site (www.microchip.com/usb):

- Microchip's *AN1140,* "*USB Embedded Host Stack*" (DS01140)
- Microchip's *AN1141,* "*USB Embedded Host Stack Programmer's Guide*" (DS01141)
- Microchip's *AN1144,* "*USB Human Interface Device Class on an Embedded Host*"

For more information on USB and embedded host functionality, in general:

- USB Implementers Forum, *"Universal Serial Bus Revision 2.0 Specification"*, http://www.usb.org/developers/docs/
- USB Implementers Forum, "*Device Class Definition for Human Interface Devices (HID)*", http://www.usb.org/developers/docs/
- USB Implementers Forum, *"Requirements and Recommendations for USB Products with Embedded Hosts and/or Multiple Receptacles"*, http://www.usb.org/developers/docs/

APPENDIX A: SOFTWARE DISCUSSED IN THIS APPLICATION NOTE

The USB keyboard application is available for download as part of Microchip's USB Embedded Host Library. This software library contains the source code and support files required for all layers of the application. Interested users may download the USB Embedded Host Library from the Microchip corporate web site, at www.microchip.com/usb.

NOTES:

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

Trademarks

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
=== ISO/TS 16949:2002 ===

MICROCHIP

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://support.microchip.com
Web Address:
www.microchip.com

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

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
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

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

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-572-9526
Fax: 886-3-572-6459

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

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

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

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

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

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

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

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

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/02/08