CY4632 WirelessUSB Software
User's Guide

Cypress Semiconductor
3901 North First Street
San Jose, CA 95134
408-943-2600

Febuary 11, 2004

CY4632 WirelessUSB Software User's Guide
Table of Contents

Table Listings

1. INTRODUCTION

1.1 Scope

The audience for this document is intended to be software developers that desire to understand and modify the software that communicates with the WirelessUSB LS Bridge HID device using the standard HID Class library provided by Microsoft.

1.2 Overview

This document describes the software source code modules used in order to communicate with the WirelessUSB LS Bridge HID device to obtain the current radio parameters for the attached WirelessUSB devices. It will not cover the details of the Microsoft Foundation Class (MFC) Library or the HID Library that contains standard system-supplied routines that user-mode applications use to communicate with USB devices that comply with the USB HID Standard. Please refer to the Microsoft Visual C++ documentation for more on MFC and HID Class concepts, in addition to the Device Class Definition for Human Interface Devices (HID) defined by the USB Implementers Forum, Inc. (*http://www.usb.org/developers/hidpage*).

2. DEFINITIONS

Following are some definitions of acronyms and words found in this document. There may be other meanings to these definitions outside of this document.

Bridge – The Bridge is the receiving radio and USB hardware that connects to the PC and enumerates as a Human Interface Device.

Device – The reference to device in this document means the keyboard or mouse device that is sending radio packets to the bridge.

DVK – A development kit produced by Cypress Semiconductor for showcasing Cypress products with a working development environment.

HID – Stands for Human Interface Device and is a product that allows an individual to interface with a computer. A keyboard and mouse are HID devices.

RDK – A reference design kit produced by Cypress Semiconductor and used by 3rd parties to produce off-the-shelf products. Everything required to take a product to production is included in the kit. This document is part of the CY4632 Mouse/Keyboard RDK.

USB – The acronym for Universal Serial Bus, a well-known serial standard used in the computing world.

WirelessUSB™ – a trademark name for Cypress 2.4 GHz radio products.

3. DEVELOPMENT ENVIRONMENT

The following tools are required to build and develop the Wireless USB Software application.

- Microsoft Visual C++ .NET
- Windows Driver Development Kit (DDK)

A Microsoft Windows based PC is used for tool execution.

The Microsoft Visual C++ .NET solution file can be found at the following location:

.\WirelessUSBSysTray\WirelessUSBTray.sln

4. SOFTWARE CODE MODULES

There are three main modules contained in the WirelessUSB Software:

- USB HID API module – generic class interface to HID Class compliant devices
- System Tray module – generic class to create and control an icon on the system tray
- WirelessUSB System Tray Application module – main system tray application module

Following are descriptions of the software module contents.

4.1 USB HID API module

The USB HID API module defines two classes, CHidDevice and CHidManager. The CHidDevice class is the primary interface to a HID device, while the CHidManager class keeps track of the arrival and removal of HID devices, along with notification to the application of such events. The building blocks for the USB HID API module was derived from the HCLIENT sample code provided in the Windows DDK. This module was designed to provide a generic interface to any HID Class compliant device and is not expected to require any modification, however all source code is provided for reference.

4.1.1 CHidDevice Class Methods

Table 2: CHidDeviceClass Methods

Method	Type	Description
OpenHidDevice()	Public	This method sets appropriate access rights, attempts to open a handle to the HID device, obtains the top collection data, and makes a call to setup input, output, and feature data buffers.
CloseHidDevice()	Public	This method closes the HID device handle, un-registers the

		HID device notification, and frees pre-parsed data and data/report buffers
RegisterHidDevice()	Public	This method registers the HID device handle for event notification
IsOpen()	Public	This method is used to report if a valid handle is open to the HID device.
IsOpenForRead()	Public	This method is used to report if the handle open to the HID device allows for read access
IsOpenForWrite()	Public	This method is used to report if the handle open to the HID device allows for write access
IsOpenOverlapped()	Public	This method is used to report if the handle open to the HID device allows for overlapped I/O
IsOpenExclusive()	Public	This method is used to report if the handle open to the HID device is setup for exclusive access
GetHandle()	Public	This method returns the handle to the HID device
Read()	Public	This method reads an input report from the HID device, performs a validity check, and unpacks the report data
Write()	Public	This method for every report ID, pack a report buffer and write the report data to the HID device
GetFeature()	Public	This method obtains the feature report from each report ID exposed by the HID device
SetFeature()	Public	This method send a feature report for each report ID exposed by the HID device
UnpackReport()	Public	This method scans through the HID report and fills in any data in the structures it can
PackReport()	Public	This method packages the HID

		report based on the data in the structures
GetManufacturerString()	Public	This method obtains the USB manufacturer string from the HID device
GetProductString()	Public	This method obtains the USB product string from the HID device
GetSerialNumberString()	Public	This method obtains the USB serial number string from the HID device
RegGetValue()	Public	This method attempts to get a registry value from the registry key where the device-specific configuration information is stored fro the HID device
RegSetValue()	Public	This method attempts to set a registry value in the registry key where the device-specific configuration information is stored for the HID device
SetupHidDevice()	Protected	This method sets up HID Input, Output and Feature data buffers used to simplify communication with HID devices
ValidateHidDevice()	Protected	This method simply returns TRUE, it is expected that this routine will be overridden by the application where the actual validation will be handled

4.1.2 CHidManager Class Methods

Table 3: CHidManagerClass Methods

Method	Type	Description
Create()	Public	This method creates an invisible window and uses the returned window handle to register for HID device notification events, then it

		creates a list of existing HID devices that will be maintained by the HID manager
IsHidDevicePresent()	Public	This method attempts to open a handle to the HID device to determine if it is present (or not) and returns the result
RefreshHidDevices()	Public	This method validates that all HID devices in the list are still present, removes those from the list that are currently not present, scans the list of all existing HID devices present, and then attempts to add the existing HID devices to the list
GetDeviceCount()	Public	This method
GetFirstHidDevice()	Public	This method returns a pointer to the first HID device in the list
GetNextHidDevice()	Public	This method returns a pointer to the next HID device in the list
GetCurrentHidDevice()	Public	This method returns a pointer to the current HID device in the list
GetHidDeviceWithPath()	Public	This method scans the current list of HID devices and returns a pointer to the HID device that matches the device path provided
GetHidDevice WithHandle()	Public	This method scans the current list of HID devices and returns a pointer to the HID device that matches the device handle provided
HidDeviceAlreadyExists()	Public	This method determines if the HID device already exists in the list
AddHidDevice()	Public	This method checks if the provided HID device already exists, and if not, adds the new HID device to the end of the list, increments the HID device

		counter, and call the HID callback function to indicate a new HID device was added
RemoveHidDevice()	Public	This method closes outstanding handle to the HID device, call HID callback function to indicate HID device is being removed, removes the HID device from the list, and deletes the HID device
RemoveAllHidDevices()	Public	This method scans though all HID devices in the list and removes them
CreateUniqueDeviceID()	Public	This method attempts to create and maintain a unique ID for the associated HID device
FreeUniqueDeviceID()	Public	This method frees the specified unique ID
NewHidDevice()	Protected	This method allocates memory for a new HID device structure
DeleteHidDevice()	Protected	This method deletes previously allocated memory for an existing HID device structure
RegisterHidNotification()	Protected	This method registers for notification of events for all HID devices and calls HID callback function to indicate registration was completed
HidDeviceArrival()	Protected	This method makes sure the HID device does not already exist in the list, and then creates a new HID device device, opens a handle to the device, adds the new HID device to the list, and registers event notification for this new HID device
HidDeviceQuery Removal()	Protected	This method readies the HID device for removal by making sure the handle is closed
HidDeviceRemoval()	Protected	This method removes the HID device

4.2 System Tray Module

The System Tray module defines the CCySysTray class which provides the interface to the system tray for the application. This module is not expected to require any modification, however all source code is provided for reference.

4.2.1 CCySysTray Class Methods

Table 4: CCySysTray Methods

Method	Type	Description
Create()	Public	This method creates an invisible window and sets up the system tray icon (if needed)
SetIcon()	Public	This method sets (or replaces) the icon displayed on the system tray
RemoveIcon()	Public	This method removes the icon from the system tray
SetToolTip()	Public	This method sets the tool tip to be displayed on the system tray
SetMenuItem()	Public	This method sets the default menu item executed when the icon is double-clicked on the system tray
IsHidden()	Public	This method is used to determine if the system tray icon is hidden
ShowBalloonTip()	Public	This method displays a balloon style tip message (only supported on W2K or higher)
OnTrayNotification()	Public	This method processes events that occur to the icon in the system tray
OnTaskbarCreated()	Protected	This method is called when the system tray is being restarted (like if explorer crashes)
WindowProc()	Protected	This method overrides the default WindowProc to call OnTrayNotification for messages targeting the system

		tray icon or OnTaskbarCreated if the system tray is being restarted

4.3 WirelessUSB System Tray Application Module

The WirelessUSB System Tray module is the main system tray application. This module places the icon on the system tray bar, manages the HID devices, displays popup messages, and controls the WirelessUSB Status Property Sheet. Additionally, via command-line parameters, this module can enable and disable the system tray application from running at startup.

4.3.1 CWirelessUSBTrayApp Class Methods

The CWirelessUSBTrayApp class performs application initialization and removal, in addition it parses command-line parameters used to enable or disable the system tray application from being run at startup.

Table 5: CWirelessUSBTrayApp Methods

Method	Type	Description
InitInstance()	Public	This method performs basic initialization and checks for any command-line parameters: if command-line parameters are found, it takes action the appropriate action and ends the application; if no command-line parameters are found then it checks to make sure the application is not currently running and, if not, then proceeds to run the system tray application
ExitInstance()	Public	This method performs some standard cleanup before the application ends
RegisterAutoLoader()	Protected	This method registers the application (itself) to always be run at startup and optionally launches itself as well
UnregisterAudoLoader()	Protected	This method un-registers the

		application (itself) to prevent running at startup and optionally ends itself from running
AutoLoadExe()	Protected	This method launches the specified EXE application

4.3.2 CMainFrame Class Methods

The CMainFrame class is the Visual C++ generated file that is a derived frame-window class for the system tray application's main frame window. This class has been modified to also perform the timer based polling of the WirelessUSB LS Bridge HID device to obtain the radio parameters and display any appropriate popup messages. Additionally, this class also processes the command message to create the WirelessUSB Status Property Sheet.

Table 6: CMainFrame Methods

Method	Type	Description
OnCreate()	Public	This method is called when a new window is created for this frame; it sets up the HID Notification callback, device status property sheet, initializes the HID manager, creates the system tray icon, sets up the menu and tool tips, if any HID devices are present then displays the icon on the system tray, and makes a call to start the timer
HIDNotification()	Public	This method processes notifications of when a HID device is added or removed from the list; it will add or remove property pages to the wireless status page and add or remove the icon from the system tray when the first or last HID devices is added or removed
OnStartTimer()	Public	This method starts the timer based on the hard-coded poll

		timer (currently set at once every 5 seconds)
OnStopTimer()	Public	This method stops the timer
OnTimer()	Public	This method is the timer routine that is called when the timer expires, it loops through all the HID devices in the list and updates their status values and then restarts the timer; also it will occasionally request an update in the battery level, currently set at once every hour
OnDestroy()	Public	This method is called when the frame window is destroyed; it stops the timer, removes the property sheet (if displayed), and removes the icon from the system tray
OnAppWireless USBStatus()	Public	This method displays the wireless status page, if it is not already displayed

4.3.3 CWirelessUSBStatusPropertyPage Class Methods

The CWirelessUSBStatusPropertyPage class is the Visual C++ generated file that implements the WirelessUSB Device Status Property Page, a unique property page is created for each WirelessUSB device enumerated.

Table 7: CWirelessUSBStatusPropertyPage Methods

Method	Type	Description
OnInitDialog()	Public	This method initializes the wireless status page, reads the current value of the Disable Warning Message checkbox from the registry, and makes a call to start the timer
OnDestroy()	Public	This method removes the wireless status page and stops the timer
OnStartTimer()	Public	This method starts the timer for the wireless status page

		based on the hard-coded poll timer (currently set at once ever 500ms)
OnStopTimer()	Public	This method stops the timer for the wireless status page
OnTimer()	Public	This method updates the HID device values displayed on the status page and then restarts the timer; also it will occasionally request an update in the battery level, currently set at once every 5 seconds while the status page is displayed
OnBnClickedWireless USBDisableWarning Message()	Public	This method is called when the Disable Warning Messages checkbox is changed; base on the checkbox value it will either disable or enable battery and signal strength warning messages for specific the HID device, the updated value is stored in the then stored in the device-specific configuration information for the HID device

4.3.4 CWirelessUSBStatusPropertySheet Class Methods

The CWirelessUSBStatusPropertySheet class is the Visual C++ generated file that implements the WirelessUSB Status Property Sheet, which generates a unique WirelessUSB Device Status Property Page for each WirelessUSB device enumerated.

Table 8: CWirelessUSBStatusPropertySheet Methods

Method	Type	Description
OnInitDialog()	Public	This method initializes the wireless status property sheet and adds a property page for each HID device in the list
OnBnClickedClose()	Public	This method ends the dialog box if the user selects the Close button

4.3.5 CHidTrayDevice Class Methods

The CHidTrayDevice class is derived from the CHidDevice class and is the class used to interface with WirelessUSB devices.

Table 9: CHidTrayDevice Methods

Method	Type	Description
RequestNewUsageValues()	Public	This method sets up and issues a Set Feature request to the HID device, which now simply requests the wireless device to provide an update of it's battery level the next time it communicates with the USB Bridge
UpdateUsageValues()	Public	This method retrieves the latest usage values from the USB Bridge, which includes wireless channel, wireless pn code, last reported battery level, and signal strength
UpdateDeviceInfo()	Public	This method makes a call to update the HID device usage values and will display a warning message (if enabled)
GetUsageIDValue()	Public	This method extracts the value of the provided Usage ID from the feature data
VerifyHidDevice()	Public	This method is called to verify that the HID device is one that should be added to the list; right now this is done by making sure the usage page reported is WIRELESSUSB_USAGEPAGE and the usage reported is either WIRELESSUSB_USAGE_KEYBOARD or WIRELESSUSB_USAGE_MOUSE

4.3.6 CHidTrayManager Class Methods

The CHidTrayManager class is derived from the CHidManager class and is used to manage WirelessUSB devices.

Table 10: CHidTrayManager Methods

Method	Type	Description
NewHidDevice()	Protected	This method creates a new HID device, initializes it, and adds it to the list of existing HID devices
DeleteHidDevice()	Protected	This method removes the HID device from the list and deletes the HID device

5. REFERENCES

WirelessUSB Theory of Operation

WirelessUSB LS 2-Way HID Systems

WirelessUSB LS Getting Started Guide

CY4632 Protocol Library

Device Class Definition for Human Interface Devices (HID) (*http://www.usb.org/developers/hidpage*)

CYWUSB6934 WirelessUSB LS Datasheet (38-16007)