Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

PICDEM™ USB User's Guide

Note the following details of the code protection feature on PICmicro® MCUs.

- The PICmicro family meets the specifications contained in the Microchip Data Sheet.
- Microchip believes that its family of PICmicro microcontrollers is one of the most secure products of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the PICmicro microcontroller in a manner outside the operating specifications contained in the data sheet. The person doing so may be engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable”.
- Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our product.

If you have any further questions about this matter, please contact the local sales office nearest to you.

DNV Certification, Inc. USA — ANSI-RAB QMS ACCREDITED — DNV MSC The Netherlands Accredited by the RvA — DNV — ISO 9001 / QS-9000 REGISTERED FIRM

Microchip received QS-9000 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona in July 1999. The Company’s quality system processes and procedures are QS-9000 compliant for its PICmicro® 8-bit MCUs, KEELOQ® code hopping devices, Serial EEPROMs and microperipheral products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001 certified.

Table of Contents

Preface

Introduction

This chapter contains general information about this manual and contacting customer support.

Highlights

Topics covered in this chapter:

- About this Guide
- Warranty Registration
- Recommended Reading
- The Microchip Internet Web Site
- Development Systems Customer Notification Service
- Customer Support

About This Guide

Document Layout

This document describes how to use PICDEM™ USB to attach a new peripheral to a PC. The manual layout is as follows:

- **Chapter 1: Getting Started with the PICDEM™ USB** – What PICDEM™ USB is and how it works.
- **Chapter 2: PICDEM™ USB Demonstration Code** – Provides USB demonstration code and information on Gameport™, PS/2® Keyboard/ Mouse, Combination Gameport/PS/2, and LCD Demo.
- **Chapter 3: PICDEM™ USB Hardware** – Contains oscillator support, connector pinout, buttons and jumpers, and power information.
- **Chapter 4: Chapter 9 USB Firmware** – Describes the USB software interface.
- **Chapter 5: Troubleshooting** – Provides solutions to common problems users may experience with PICDEM™ USB. It also includes FAQ on Hardware, PC/Windows® and Macintosh® concerning the PICDEM™ USB.
- **Appendix A: Schematics** – Provides the schematics for the PICDEM™ USB.
- **Appendix B: PS/2 Lookup Tables** – Provides scan and command code tables for easy reference.

- **Index** – Provides a cross-reference listing of terms, features, and sections of this document.
- **PICDEM™ USB Worldwide Sales and Service** – Lists Microchip sales and service locations and telephone numbers worldwide.

Conventions Used in this Guide

This manual uses the following documentation conventions:

Documentation Conventions

Description	Represents	Examples
Code (Courier font):		
Plain characters	Sample code, file names and paths	`#define START c:\autoexec.bat`
Angle brackets: < >	Variables	`<label>, <exp>`
Square brackets [ ]	Optional arguments	`MPASMWIN [main.asm]`
Curly brackets and pipe character: { | }	Choice of mutually exclusive arguments, an OR selection	`errorlevel {0|1}`
Lower case characters in quotes	Type of data	`"filename"`
Ellipses...	Used to imply (but not show) additional text that is not relevant to the example	`list ["list_option.., "list_option"]`
0xnnn	A hexadecimal number where 'n' is a hexadecimal digit	`0xFFFF, 0x007A`
Italic characters	A variable argument; it can be either a type of data (in lower case characters) or a specific example (in uppercase characters)	`char isascii (char, ch);`
Interface (Arial font):		
Underlined, italic text with right arrow	A menu selection from the menu bar	*File > Save*
Bold characters	A window or dialog button to click	**OK**, **Cancel**
Characters in angle brackets < >	A key on the keyboard	<Tab>, <Ctrl-C>
Documents (Arial font):		
Italic characters	Referenced books	*MPLAB® IDE User's Guide*

Updates

All documentation becomes dated, and this user's guide is no exception. Since the MPLAB IDE, PICDEM™ USB and other Microchip tools are constantly evolving to meet customer needs, some MPLAB® dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site at http://www.microchip.com to obtain the latest documentation available.

Warranty Registration

Please complete the enclosed Warranty Registration Card and mail it promptly. Sending in your Warranty Registration Card entitles you to receive new product updates. Interim software releases are available at the Microchip web site.

Recommended Reading

This user's guide describes how to use PICDEM™ USB. The data sheets contain current information on programming the specific microcontroller devices.

`README.USB`

For the latest information on using PICDEM™ USB, read the `README.USB` file (ASCII text file) included with the PICDEM™ USB software. The `README.USB` file contains update information that may not be included in this document.

MPLAB® IDE User's Guide (DS51025)

Comprehensive guide that describes installation and features of Microchip's MPLAB Integrated Development Environment (IDE), as well as the editor and simulator functions in the MPLAB environment.

MPASM™ User's Guide with MPLINK™ Linker and MPLIB™ Librarian (DS33014)

Describes how to use Microchip Universal PICmicro® Microcontroller Assembler (MPASM™), Linker (MPLINK™), and Librarian (MPLIB™).

Technical Library CD-ROM (DS00161)

This CD-ROM contains comprehensive data sheets for Microchip PICmicro® MCU devices available at the time of print. To obtain this disk, contact the nearest Microchip Sales and Service location (see back page), or download individual data sheet files from the Microchip web site (http://www.microchip.com).

Embedded Control Handbook (DS00711)

This handbook consists of several documents that contain a wealth of information about microcontroller applications. To obtain these documents, contact the nearest Microchip Sales and Service location (see back page).

The application notes described in these manuals are also obtainable from Microchip Sales and Service locations or from the Microchip web site (http://www.microchip.com).

PICmicro™ Mid-Range MCU Family Reference Manual (DS33023)

This manual explains the general details and operation of the MCU family architecture and peripheral modules. It is designed to complement the device data sheets.

Microsoft® Windows® Manuals

This manual assumes that users are familiar with Microsoft Windows operating system. Many excellent references exist for this software program, and should be consulted for general operation of Windows.

USB Complete

This book is a good introduction to the USB interface and how to use it. It was written by Jan Axelson.

PIC16C745/765 Data Book (DS41124)

This book contains everything you ever wanted to know about the PIC16C745/765 and more.

USB IF

The USB IF can be downloaded from www.usb.org, which has all the specifications for the USB interface. It is a valuable tool. Be sure to download the USBCheck PC tools and register on the USB mailing list.

Microsoft DDK

If you are developing PC drivers for Windows, don't forget to get the driver development kit and the professional version of Visual Studio®.

Apple® USB Software Developer Kit (SDK)

If you are going to develop Mac drivers, go to `www.developer.apple.com/SDK/index.html` and download the SDK and register on the USB mailing list.

The Microchip Internet Web Site

Microchip provides online support on the Microchip World Wide Web (WWW) site.

The web site is used by Microchip as a means to make files and information easily available to customers. To view the site, the user must have access to the Internet and a web browser, such as Netscape® Communicator or Microsoft® Internet Explorer®. Files are also available for FTP download from our FTP site.

Connecting to the Microchip Internet Website

The Microchip web site is available by using your favorite Internet browser to attach to:

http://www.microchip.com

The file transfer site is available by using an FTP program/client to connect to:

ftp://ftp.microchip.com

The web site and file transfer site provide a variety of services. Users may download files for the latest Development Tools, Data Sheets, Application Notes, User's Guides, Articles, and Sample Programs. A variety of Microchip specific business information is also available, including listings of Microchip sales offices, distributors and factory representatives. Other data available for consideration is:

- Latest Microchip Press Releases
- Technical Support Section with Frequently Asked Questions
- Design Tips
- Device Errata
- Job Postings
- Microchip Consultant Program Member Listing
- Links to other useful web sites related to Microchip Products
- Conferences for Products, Development Systems, Technical Information and more
- Listing of Seminars and Events

Development Systems Customer Notification Service

Microchip started the customer notification service to help our customers keep current on Microchip products with the least amount of effort. Once you subscribe to one of our list servers, you will receive email notification whenever we change, update, revise or have errata related to that product family or development tool. See the Microchip web page at http://www.microchip.com for other Microchip list servers.

The Development Systems list names are:

- Compilers
- Emulators
- Programmers
- MPLAB IDE
- Otools (other tools)

Once you have determined the names of the lists that you are interested in, you can subscribe by sending a message to:

```
listserv@mail.microchip.com
```

with the following as the body:

```
subscribe <listname> yourname
```

Here is an example:

```
subscribe programmers John Doe
```

To UNSUBSCRIBE from these lists, send a message to:

```
listserv@mail.microchip.com
```

with the following as the body:

```
unsubscribe <listname> yourname
```

Here is an example:

```
unsubscribe programmers John Doe
```

The following sections provide descriptions of the available Development Systems lists.

Compilers

The latest information on Microchip C compilers, Linkers and Assemblers. These include MPLAB® C17, MPLAB® C18, MPLINK™ Object Linker (as well as MPLIB™ Object Librarian), and MPASM™ Assembler.

To SUBSCRIBE to this list, send a message to:

```
listserv@mail.microchip.com
```

with the following as the body:

```
subscribe compilers yourname
```

Emulators

The latest information on Microchip In-Circuit Emulators. These include MPLAB® ICE and PICMASTER® Emulator.

To SUBSCRIBE to this list, send a message to:

```
listserv@mail.microchip.com
```

with the following as the body:

```
subscribe emulators yourname
```

Programmers

The latest information on Microchip PICmicro device programmers. These include PRO MATE® II and PICDEM™ USB.

To SUBSCRIBE to this list, send a message to:

`listserv@mail.microchip.com`

with the following as the body:

`subscribe programmers yourname`

MPLAB IDE

The latest information on Microchip MPLAB IDE, the Windows Integrated Development Environment for development systems tools. This list is focused on MPLAB IDE, MPSIM™ Simulator, MPLAB's Project Manager and general editing and debugging features. For specific information on MPLAB compilers, linkers and assemblers, subscribe to the COMPILERS list. For specific information on MPLAB emulators, subscribe to the EMULATORS list. For specific information on MPLAB device programmers, please subscribe to the PROGRAMMERS list.

To SUBSCRIBE to this list, send a message to:

`listserv@mail.microchip.com`

with the following as the body:

`subscribe mplab yourname`

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Corporate Applications Engineer (CAE)
- Hotline

Customers should call their distributor, representative, or field application engineer (FAE) for support. Local sales offices are also available to help customers. See the back cover for a listing of sales offices and locations.

Corporate applications engineers (CAEs) may be contacted at (480) 792-7627.

In addition, there is a Systems Information and Upgrade Line. This line provides system users a listing of the latest versions of all of Microchip's development systems software products. Plus, this line provides information on how customers can receive any currently available upgrade kits.

The Hotline Numbers are:

1-800-755-2345 for U.S. and most of Canada, and

1-480-792-7302 for the rest of the world.

NOTES:

Chapter 1. Getting Started with the PICDEM™ USB

1.1 Introduction

The Universal Serial Bus (USB) has become the most accepted way for any new peripheral to be attached to a PC. The interface is fully supported by all major computer manufacturers and most operating systems. Since this interface has become more accepted, it has moved out of the consumer market. It is starting to find its way into data acquisition and industrial markets. Within these markets, a large number of PICmicro solutions are looking for a migration path to USB. By providing a USB derivative of the classic PIC16C72/74 devices, Microchip encourages you to move your applications into the new world of USB; and as an addition bonus, the examples provide an opportunity to play games.

1.2 Highlights

The topics covered in this chapter are:

- Unpacking your PICDEM™ USB
- Running the Default Demonstration
- Branching out on your own

1.3 Unpacking your PICDEM™ USB Kit

1.3.1 Supplied Items

The items contained in your PICDEM™ USB box are:

- Microchip USB CD-ROM containing USB support documentation
- PICDEM™ USB Circuit Board with a PIC16C765 installed
- CD-ROM containing MPLAB IDE
- 3 ft. USB A-B cable
- Small box containing a windowed PIC16C745 and PIC16C765

1.3.2 Required Items

The items required are:

- PC for running MPLAB IDE
- Visual Basic and/or Visual C++ to modify the PC examples
- PC with USB running Windows® 98 or newer (for the PC examples)
- Macintosh with USB running MacOS X 10.0 or newer (for the Macintosh examples). The HID examples work with MacOS 8.6 or newer.
- Apple® Project Builder to modify the Macintosh code examples

- PICSTART® Plus or PRO MATE® II to program the devices
- UV chip eraser to clean the mistakes
- Copy of Apple's USB DDK so you can use the USB bus monitoring tools on the Macintosh (http://developer.apple.com/hardware/usb/)
- Copy of the USB-IF PC tools (www.usb.org)

1.3.3 Suggested Items

The items suggested are:

- USB protocol analyzer such as CATC
- Membership in the USB-IF, Inc. (www.usb.org)
- MPLAB® ICE 2000

1.4 Running the Default Demonstration

If you have a PC/Macintosh with USB, attach your PICDEM™ USB with the supplied cable. The LED's should quickly blink as the PICDEM™ USB identifies itself. The EP1 ACT light should start to flicker steadily, and the mouse cursor on your computer should start to move in a circle.

Unplug the USB cable and plug a PS/2 mouse into the PS/2 connector. Re-attach the USB cable. You will notice the mouse cursor is no longer moving in a circle, but is responding to the mouse. Unplug the USB cable and plug in a PS/2 keyboard. Re-attach the USB cable. The LED's flicker and the keyboard is functioning as a USB keyboard.

Regardless of what PS/2 device is attached (or not attached), the EP2 ACT light will be flashing. This is due to the PICDEM™ USB constantly streaming gameport data to the host, regardless of whether a gaming device is plugged in or not. You can test the gameport by using the gamepad specified in Section 2.1: Gameport - USB Translator.

This example takes advantage of the existing human interface device (HID) code in the host's operating system. For a more complex example showing host driver code, you will have to consult the LCD Demo example.

1.5 Branching Out on Your Own

1.5.1 Developing New USB Applications

The following steps are recommended to develop most new USB applications.

- Describe the application.
- Create the descriptors.
- Debug the report descriptor with dummy data.
- Develop the rest of the application.

By following these steps, the hardest part of the development can be completed right away on known good hardware (the PICDEM™ USB). After the application is communicating to the PC correctly, the application specific hardware and software can be developed.

1.5.1.1 Describing the Application

When you start developing your application, make sure that your data requirements fit the USB specification. A low speed device is limited to 2 channels of communication (end points), with each channel limited to 800 bytes per second. The most common mistake is to assume that the entire 1.5 Mbs is available for your application.

1.5.1.2 Creating the Descriptors

The most difficult part of any USB application is determining what the device descriptors should be. Every USB device communicates its requirements to the host through a process called enumeration. During enumeration, the device descriptors are transferred to the host and the host assigns a unique address to the device. The descriptors are described in detail in Chapter 9 of the USB 2.0 specification. Bundled with the USB tools CD, a descriptor tool is provided to assist you in creating your own descriptors.

1.5.1.3 Debugging the Report Descriptor

The report descriptor allows HID devices to communicate to the host. The report descriptor communicates the exact packet format of your data. This is where the PC determines how large your packets will be. Report descriptors range from the very specific (a multi-function joystick) to the very generic (a specialized communications device for your application). Tools are available to assist you in creating your report descriptors. The descriptor tool, which is bundled in your kit, will have report descriptor capabilities with a future revision. After the report descriptor is written make sure that it is working with the PC by using PC analysis tools. These are available from the USB-IF web site for the PC, and Apple Computer for Macintosh machines. Use simple counters and other dummy data to test report descriptor traffic. After the communications link is working, it will be much easier to develop the rest of the application.

1.5.1.4 Developing the Rest of the Application

The next step is to add your specific hardware and software. Carefully study the schematic in this guide and use the same circuitry for the USB connections. The circuitry will never change for this device. When your hardware is built, you can be confident that your communications code will work because it was developed on the development system with the same communications circuitry.

NOTES:

Chapter 2. USB Demonstration Code

A variety of examples have been provided on the CD-ROM to speed you towards a successful start with USB. These examples range from the very basic gameport translator, to a very specialized LCD display. The gameport and PS/2 examples are excellent tools to use, in order to become more familiar with report descriptors and basic USB communications. These basic examples are extremely useful because the OS vendor has already written generic Human Interface Device (HID) drivers. The LCD display example does not have a generic PC driver. In this example, PC software has been provided to demonstrate how you can develop your own PC applications to interface to your device. Example code is provided for PC and Macintosh.

2.1 Gameport - USB Translator

2.1.1 Introduction

The gameport to USB translator is a simple example that reads a PC gameport and reports the information over USB. Since peripherals for gameports come in all shapes and sizes, it is important to identify exactly what peripheral is used, in this example -- the Dexxa® 8-button gamepad. Other gamepads can be used, however, some change to the firmware functions may be required. The example code will enumerate as a gamepad with 2 axis and 6 buttons. PORTA on the PICmicro MCU is used to read the analog voltages from resistors in the direction pad (D-pad) and two of the buttons, while PORTD reads the switches for the four remaining buttons. Refer to Figure 2.1.

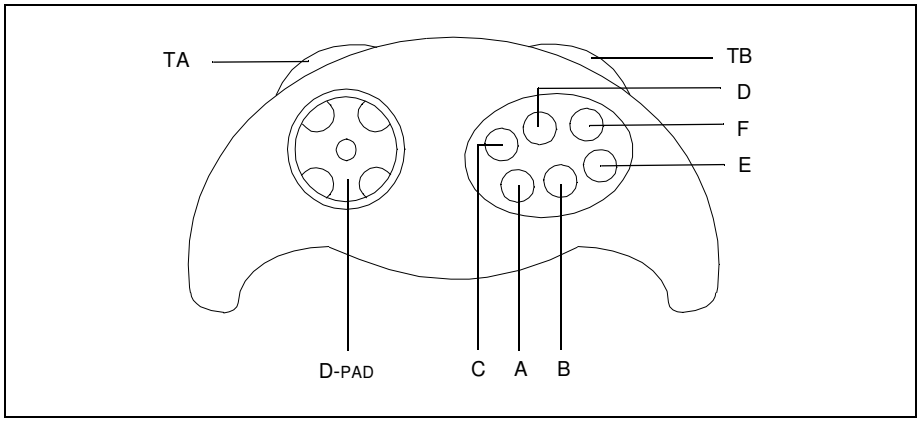

Figure 2.1: Dexxa® Gamepad

Note: Although the Dexxa® gamepad has 8 buttons, the code enumerates as a 6-button gamepad. This is due to the gamepad having two "special" buttons, that rapidly fire two of the other button outputs.

2.1.2 About the PC Gameport

The PC gameport was designed to support 2 joysticks. Each joystick was intended to have 2 axis and 2 buttons. The gameport was later extended to support a MIDI serial interface. The gameport hardware supplied with the PICDEM™ USB does not support the MIDI interface, so those pins have been disconnected. Two joysticks are supported, but to simplify the example, only one joystick is used.

2.1.3 Hardware Implementation

The hardware is supplied on the PICDEM™ USB circuit board. The board is wired with PORTA<3:0> and PORTD<3:0>, connected to the DB-15 connector of the gameport. PORTD<3:0> are the digital inputs for buttons A-D. PORTA<0> and PORTA<1> are the X-Y inputs for the D-pad on the gamepad. PORTA<2> and PORTA<3> are the inputs for buttons E and F (see Appendix A for the gamepad to PICDEM™ USB schematic). All of the analog pins have a series resistor, a resistor, and capacitor tied to ground. The reason for this comes from the way analog pins were originally read by the PC. The PC would clear a capacitor tied to ground at its end and then time how long it took the capacitor to charge up. The capacitor would charge at a rate proportional to the resistance of a pot, varied by one axis of the joystick, for instance. This is legacy technology and is not needed when using a PICmicro MCU with an analog-to-digital converter. As a result, the before mentioned circuitry was put into place in order to obtain an analog output from the D-pad of the gamepad and buttons E and F. Another point of confusion develops from viewing the analog output of the gamepad with this circuitry in place. The

voltage level is not symmetrical around 2.5 volts. You would expect the center position of the D-pad's x-axis, for instance, to output 2.5 volts, the left position to output 0 volts and the right position to output 5 volts. Ideally it would, but the circuitry in the gamepad is very simple; besides, this symmetry is not needed. All that is needed is a lower voltage input into the PICmicro MCU when the pad is pressed one direction from center, and a higher voltage input when the pad is pressed in the other direction from center. The circuitry found on the PICDEM™ USB board amplifies the differences in voltages between these three positions and therefore, gives three very distinct analog-to-digital readings.

Note 1: Buttons E and F are connected to analog pins because the gameport has four analog pins and four digital pins. Buttons A-D use up all the digital pins so the analog pins are the only inputs left over. Many joysticks have a multiplexer on button numbers greater than four, in order to achieve the same result using just the digital pins.

2: Be very careful when using the LCD interface and the gameport together because they share PORTD.

2.1.4 Gamepad Firmware

The firmware has four functions. Each function either returns one piece of information concerning the gameport status, or initiates the gameport registers. Refer to Table 2.1.

Table 2.1: Gameport Firmware

Function	Description
ReadXAxis	Returns a digital value for the x-axis in W
ReadYAxis	Returns a digital value for the y-axis in W
ReadButtons	Returns the state of the six buttons in W (bits 0-5 correspond to buttons A-F)
InitGameport	Initialize the system to use the Gameport

2.1.5 Gamepad Report Descriptor

The report descriptor used in this example is for a gamepad with 6 buttons. The minimum and maximum report values for the axis are set to 0 - 255. This is to be used with Windows. Some versions don't seem to handle a range of -127 to 127. The -127 value is translated to 255 and causes extreme movement to the right. The range selected works correctly on Windows and on Macintosh. Interestingly, the Macintosh performs a simple filter on the center of the range. It will filter changes of 1 count to prevent cursor jitter when the stick is centered. This caused problems with a minimum and maximum range of 0 - 2. When the range was extended to 0 - 4, it worked much better. In this example, the analog input from the D-pad will be converted to digital, filtered, and then sent to the PC via USB. Since this digital value can range from

0 - 255, the descriptor calls for a range of 0 - 255. The descriptor, shown below in HEX form, is in the gamepad descriptor file (usb_ch9.asm).

```
0x05, 0x01          USAGE_PAGE (Generic Desktop)
0x09, 0x05          USAGE (Game Pad)
0xA1, 0x01          COLLECTION (Application)
0x09, 0x01             USAGE (Pointer)
0xA1, 0x00             COLLECTION (Physical)
0x09, 0x30                USAGE (X)
0x09, 0x31                USAGE (Y)
0x15, 0x00                LOGICAL_MINIMUM (0)
0x26, 0xFF, 0x00          LOGICAL_MAXIMUM (255)
0x75, 0x08                REPORT_SIZE (8)
0x95, 0x02                REPORT_COUNT (2)
0x81, 0x02                INPUT (Data,Var,Abs)
0xC0                   END_COLLECTION
0x05, 0x09             USAGE_PAGE (Button)
0x19, 0x01             USAGE_MINIMUM (Button 1)
0x29, 0x06             USAGE_MAXIMUM (Button 6)
0x15, 0x00             LOGICAL_MINIMUM (0)
0x25, 0x01             LOGICAL_MAXIMUM (1)
0x75, 0x01             REPORT_SIZE (1)
0x95, 0x06             REPORT_COUNT (6)
0x81, 0x02             INPUT (Data,Var,Abs)
0x95, 0x02             REPORT_COUNT (2)
0x81, 0x03             INPUT (Constant,Var,Abs)
0xC0                END_COLLECTION
```

This report descriptor describes the packet format for the USB data. The data is filled from Least Significant Byte, Least Significant bit through to the Most Significant Byte, Most Significant bit. The first field found will be the first bit/byte. In the report descriptor above, the first data is 8 bits (the REPORT_SIZE is 8) and it is the X axis (the first USAGE of the physical collection is X). So the first byte on the bus will be the X axis value. The second byte will be the Y axis. The third byte will be button A in bit 0, followed by button B in bit 1, and so on. Because every USB transaction must be in whole number bytes, the data is padded by one constant report, 2-bits long.

2.1.6 Gameport Translation

Translating the bits from the physical hardware to the USB buffer is very simple. Because we set the logical minimum and maximum to be 0 to 255, it exactly matches the scaling of the analog-to-digital converter. So first, we convert the X and Y axis and store the values in the first two buffer locations. Secondly, we read the six buttons and store the values in the third buffer location in bits 0-5. Lastly, we inform the serial interface engine that data is available and wait for the host PC to come pick it up.

2.1.7 Vendor/Product Identification

Besides the report information, the descriptors also contain manufacturing and product identification codes. Microchip has a registered Vendor ID with the USB IF forum, which identifies Microchip's Vendor ID as 0x04D8. You are allowed to use this ID for your own testing, but you may not ship any products with this code without written permission from Microchip. Microchip has defined product ID's for each demonstration code included in this kit. As additional demonstration devices are released, Microchip will ensure that no duplicate ID's are used.

2.2 PS/2 Keyboard/Mouse - USB Translator

2.2.1 Introduction

This is the demonstration firmware that is programmed into the PIC16C765 and installed on the PICDEM™ USB demonstration board. The PS/2 connector was added so PS/2 mice and keyboards could be translated to USB. Again, this is a straightforward application intended to provide practice with device descriptors. The PS/2 interface is a synchronous serial interface with different data protocols for keyboards and mice. Device descriptors in the PICmicro microcontroller allows the unit to report itself as either a keyboard or a mouse. When a mouse is attached to a PS/2 port, it identifies itself as a mouse. When the PICmicro MCU receives this identification, it will perform a soft detach from the USB bus and re-attach as a mouse. When the PICmicro MCU identifies the PS/2 device as a keyboard, it will perform a soft detach and re-attach as a keyboard. Using soft detach may be a useful feature in your application, so you can practice using it with this example.

Note: It may be necessary to attach 110 kOhm pull-down resistors on RC0 and RC1, in order for auto-detect to work without pressing $\overline{\text{MCLR}}$ between plugging and unplugging PS/2 devices.

2.2.2 About the PS/2 Port

IBM® originally developed the PS/2 port for use on its PS/2 family of computers. This port is a synchronous serial port clocked by the PS/2 device (keyboard/mouse). Generally, PC's have two PS/2 ports labeled as keyboard or mouse. The PICDEM™ USB only has one PS/2 port. Since the hardware is the same, either the keyboard or mouse can be used, simply by interpreting the data correctly.

2.2.3 Hardware Implementation

The PS/2 port is a 6-pin DIN which only uses 4 pins. The pins are used for power, ground, clock, and data. Power and ground pins are directly tied to V_{DD} and V_{SS}. If power management is desired, the power pins must be driven via switches from other I/O pins.The clock pin is connected to RC0, while the data pin is connected to RC1. The PS/2 device clocks the host even when it is receiving data. The data pin is used to send and receive data from the keyboard.

2.2.4 Data Format

The data is sent via PS/2 one byte at a time, regardless of direction, host-to-device, or vice versa. The data is as follows:

- First comes a START bit (always low), followed by a
- data byte (Least Significant bit to Most Significant bit), then by a
- parity bit (high for an even number of high bits in the data byte and low for an odd number), then by a
- STOP bit (always high)

In the case of host-to-device communication, the STOP bit is immediately followed by an ACK bit (low), which is sent by the device to the host. The bits are read on the falling edge of the clock for device-to-host communication and on the rising edge for host-to-device communication. In the IDLE state, the clock and data lines are held high by the device. See Figure 2.2 and Figure 2.3 for device-to-host and host-to-device communication, respectively.

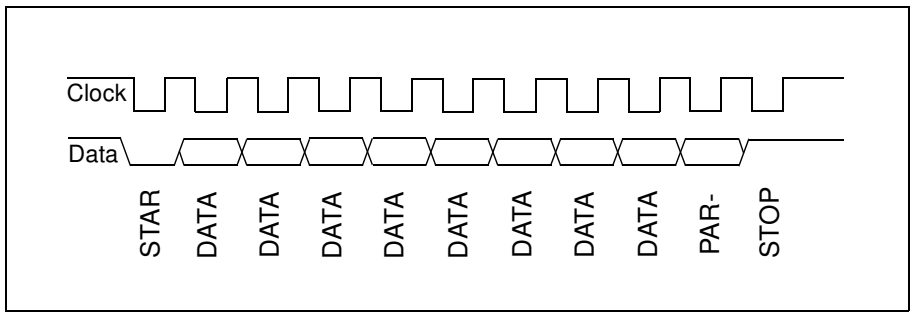

Figure 2.2: Device-to-Host Communication (Data bit Read on Falling Edge of Clock)

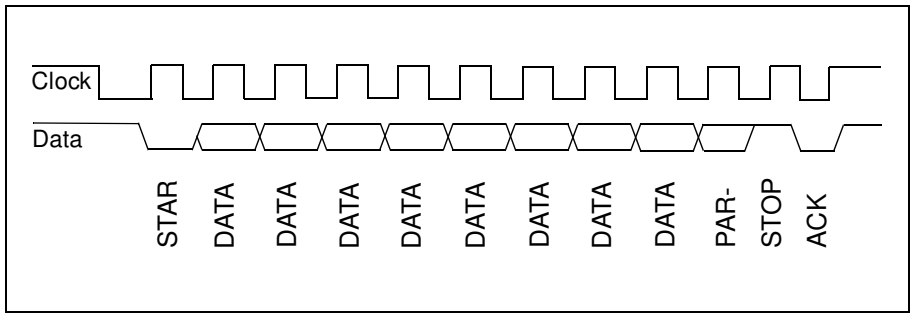

Figure 2.3: Host-to-Device Communication (Data bit Read on Rising Edge of Clock)

2.2.5 Keyboard

The PS/2 keyboard data report format is summarized for every key in Appendix B. Make codes are the byte or bytes that the PS/2 keyboard sends to the host when a certain key is pressed. Break codes are the bytes that the PS/2 keyboard sends when the user releases a key. If the user does not release a specific key for several hundreds of a millisecond, the make code will be sent repeatedly until the user releases the key. At this point, the break code is sent. The Translation to USB, Section 2.2.11, details how the firmware converts PS/2 keycodes to USB keycodes.

Note: The PS/2 keycodes shown in Appendix B do not apply to all PS/2 keyboards. Several code sets have been used through the years. However, this code set is the most common.

2.2.6 Mouse

Table 2.2 details a typical PS/2 mouse data format.

Table 2.2: PS/2 Mouse Data Report Format

Byte	Bit	Description
3	7 6-1 0	MSB of Y Data Y Data LSB of Y Data
2	7 6-1 0	MSB of X Data X Data LSB of X Data
1	7 6 5 4 3 2 1 0	Y Data Overflow, 1 = overflow X Data Overflow, 1 = overflow Y Data Sign, 1 = negative X Data Sign, 1 = negative Reserved Reserved Right Button Status, 1 = pressed Left Button Status, 1 = pressed

2.2.7 Hardware Implementation

A PS/2 port is a 6-pin DIN, but only four pins are used (see Appendix A for pinout.) The pins are power, ground, clock and data. The clock pin is connected to RC0, while the data pin is connected to RC1. A PS/2 device clocks the host even when it is receiving data. By attaching the data pin to RC1, a START bit will interrupt the PICmicro MCU through a Capture/Compare/PWM (CCP) event. Refer to the PIC16C7XX data sheet for details on CCP. Power and ground are directly tied to V_{DD} and V_{SS}. If power management were desired, the power pins would be driven via switches from other I/O pins.

2.2.8 PS/2 Firmware

The PS/2 firmware is entirely interrupt driven. As mentioned before, an interrupt is generated when the START bit is received, at which time the firmware will begin its receive routine. In addition to this interrupt, every 168 ms, a timer overflow interrupts the normal program flow and implements one state of the mouse/keyboard/cursor demonstration state machine. This state machine handles sending bytes to and translating bytes received from the PS/2 device, automatically. These two interrupts essentially handle everything, except for transferring the bytes via USB to the PC. In addition, it does all of this work in the background while a developer's code runs in the foreground. The only operation that the developer's program must implement is sending keyboard or mouse data to the PC via USB. The developer needs only to be concerned with the TYPE and eight BUFFER registers. BUFFER registers 0 to 7 are the registers where translated PS/2 device data gets placed. TYPE contains the following status bits:

Table 2.3: PS/2 State Machine Status Report

TYPE bit	Name	Description
0	CONNECTED	1 = device connected
1	MOUSE	1 = device connected is a mouse
2	KEYBOARD	1 = device connected is a keyboard
3	DATA READY	1 = data is ready; must be cleared by user

2.2.9 Report Descriptor

The report descriptors used in the example code for both the keyboard and mouse were copied directly out of the *HID Usage Tables*. The *HID Usage Tables* document is published by the USB Implementers Forum (www.usb.org). Many other useful HID report descriptor examples can be found in this document. The keyboard and mouse report descriptors are not sent out at the same time that the PICmicro MCU is enumerated by the host. Rather, the PICmicro MCU will only send the report descriptor that corresponds to the device it has detected as being attached at that time.

2.2.9.1 Keyboard Descriptor

```
0x05, 0x01     usage page (generic desktop)
0x09, 0x06     usage (keyboard)
0xA1, 0x01     collection (application)
0x05, 0x07        usage page (key codes)
0x19, 0xE0        usage minimum (224)
0x29, 0xE7        usage maximum (231)
0x15, 0x00        logical minimum (0)
0x25, 0x01        logical maximum (1)
0x75, 0x01        report size (1)
0x95, 0x08        report count (8)
0x81, 0x02        input (data, variable, absolute)
```

```
0x95, 0x01        report count (1)
0x75, 0x08        report size (8)
0x81, 0x01        input (constant)
0x95, 0x05        report count (5)
0x75, 0x01        report size (1)
0x05, 0x08        usage page (page# for Led)
0x19, 0x01        usage minimum (1)
0x29, 0x05        usage maximum (5)
0x91, 0x02        output (data, variable, absolute)
0x95, 0x01        report count (1)
0x75, 0x03        report size (3)
0x91, 0x01        output (constant)
0x95, 0x06        report count (6)
0x75, 0x08        report size (8)
0x15, 0x00        logical minimum (0)
0x25, 0x65        logical maximum (101)
0x05, 0x07        usage page (key codes)
0x19, 0x00        usage minimum (0)
0x29, 0x65        usage maximum (101)
0x81, 0x00        input (data, array)
0xC0           end collection
```

2.2.9.2 Mouse Descriptor

```
0x05, 0x01     usage page (generic desktop)
0x09, 0x02     usage (mouse)
0xA1, 0x01     collection (application)
0x09, 0x01        usage (pointer)
0xA1, 0x00        collection (linked)
0x05, 0x09           usage page (buttons)
0x19, 0x01           usage minimum (1)
0x29, 0x03           usage maximum (3)
0x15, 0x00           logical minimum (0)
0x25, 0x01           logical maximum (1)
0x95, 0x03           report count (3)
0x75, 0x01           report size (1)
0x81, 0x02           input (3 button bits)
0x95, 0x01           report count (1)
0x75, 0x05           report size (5)
0x81, 0x01           input (constant 5 bit padding)
0x05, 0x01           usage page (generic desktop)
0x09, 0x30           usage (X)
0x09, 0x31           usage (Y)
0x15, 0x81           logical minimum (-127)
0x25, 0x7F           logical maximum (127)
0x75, 0x08           report size (8)
0x95, 0x03           report count (2)
0x81, 0x06           input (2 position bytes X & Y)
0xC0              end collection
0xC0           end collection
```