INTRODUCTION

This Technical Brief details the translation of a PS/2® keyboard to a USB keyboard using the PIC16C745/765. Although this is a fairly simple application for the PIC16C745/765, it accomplishes the intended purpose of this brief: to provide an in-depth look at the Set_Report request.

The Set_Report Request

The Set_Report request is a HID specific command. It was the only provision in Version 1.0 of the USB specification that could send data from the host to a peripheral. This later changed with Version 1.1 and the introduction of the interrupt OUT transfer. All low-speed devices created before the introduction of Version 1.1 had to use the Set_Report request for host-to-device communication. The limitation of this is that Set_Report communicates over Endpoint 0 (EP0). Endpoint 0 is the control endpoint. In other words, EP0 is the avenue through which the device and host send each other commands, so any outputted data from the host has to share bandwidth with administrative commands sent back and forth between the host and the device. Developers wanted a way to create a dedicated OUT endpoint with one or both of the endpoints available to them, EP1 and EP2. As a result, Version 1.1 of the USB specification included a provision for the interrupt OUT transfer.

There are many instances where the Set_Report request is useful in today’s USB devices. In devices where the maximum amount of bandwidth is needed for device-to-host transactions, both EP1 and EP2 can be set as IN endpoints. If the same device must receive occasional data packets from the host, it is better to use the Set_Report request to send that data from the host to the device than to change either EP1 or EP2 to an OUT endpoint. The keyboard is an example of a peripheral in which the host occasionally sends it data, namely, the status of the keyboard LEDs (Caps Lock, Num Lock and Scroll Lock LEDs are the most common LEDs). This data is sent from the host to the device with the Set_Report request only when the user presses one of the corresponding keys for these LEDs (i.e., the Caps Lock key).

Descriptors

Figure 1 shows the USB keyboard report descriptor. There are two Output items in this report descriptor. These items describe a byte of data that will be sent from the host to the device comprised of five LED status bits and three bits of padding. Because this report descriptor is associated with an IN endpoint (specified by the endpoint descriptor), the host knows all Output items describe data that will be sent to the device via a Set_Report. The remainder of the descriptor describes the format that data will be sent from the keyboard to the host. The output data should be ignored when trying to envision what the device-to-host report format looks like. See Figure 2 for the USB keyboard-to-host data format.
FIGURE 1: USB KEYBOARD REPORT DESCRIPTOR

<table>
<thead>
<tr>
<th>Byte</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x05, 0x01</td>
<td>usage page (generic desktop)</td>
</tr>
<tr>
<td>0x09, 0x06</td>
<td>usage (keyboard)</td>
</tr>
<tr>
<td>0xA1, 0x01</td>
<td>collection (application)</td>
</tr>
<tr>
<td>0x05, 0x07</td>
<td>usage page (key codes)</td>
</tr>
<tr>
<td>0x19, 0xE0</td>
<td>usage minimum (224)</td>
</tr>
<tr>
<td>0x29, 0xE7</td>
<td>usage maximum (231)</td>
</tr>
<tr>
<td>0x15, 0x00</td>
<td>logical minimum (0)</td>
</tr>
<tr>
<td>0x25, 0x01</td>
<td>logical maximum (1)</td>
</tr>
<tr>
<td>0x75, 0x01</td>
<td>report size (1)</td>
</tr>
<tr>
<td>0x95, 0x08</td>
<td>report count (8)</td>
</tr>
<tr>
<td>0x81, 0x02</td>
<td>input (data, variable, absolute)</td>
</tr>
<tr>
<td>0x95, 0x01</td>
<td>report count (1)</td>
</tr>
<tr>
<td>0x75, 0x08</td>
<td>report size (8)</td>
</tr>
<tr>
<td>0x81, 0x01</td>
<td>input (constant)</td>
</tr>
<tr>
<td>0x95, 0x05</td>
<td>report count (5)</td>
</tr>
<tr>
<td>0x75, 0x01</td>
<td>report size (1)</td>
</tr>
<tr>
<td>0x05, 0x08</td>
<td>usage page (page# for LEDs)</td>
</tr>
<tr>
<td>0x19, 0x01</td>
<td>usage minimum (1)</td>
</tr>
<tr>
<td>0x29, 0x05</td>
<td>usage maximum (5)</td>
</tr>
<tr>
<td>0x91, 0x02</td>
<td>output (data, variable, absolute)</td>
</tr>
<tr>
<td>0x95, 0x01</td>
<td>report count (1)</td>
</tr>
<tr>
<td>0x75, 0x03</td>
<td>report size (3)</td>
</tr>
<tr>
<td>0x91, 0x01</td>
<td>output (constant)</td>
</tr>
<tr>
<td>0x95, 0x06</td>
<td>report count (6)</td>
</tr>
<tr>
<td>0x75, 0x08</td>
<td>report size (8)</td>
</tr>
<tr>
<td>0x15, 0x00</td>
<td>logical minimum (0)</td>
</tr>
<tr>
<td>0x25, 0x65</td>
<td>logical maximum (101)</td>
</tr>
<tr>
<td>0x05, 0x07</td>
<td>usage page (key codes)</td>
</tr>
<tr>
<td>0x19, 0x00</td>
<td>usage minimum (0)</td>
</tr>
<tr>
<td>0x29, 0x65</td>
<td>usage maximum (101)</td>
</tr>
<tr>
<td>0x81, 0x00</td>
<td>input (data, array)</td>
</tr>
<tr>
<td>0xC0</td>
<td>end collection</td>
</tr>
</tbody>
</table>

Choose the usage page "keyboard" is on
Device is a keyboard
This collection comprises all the data words
Choose the key code usage page
Choose key codes 224 to 231 which are modifier keys
(left and right alt, shift, ctrl and win)
Each of these eight key codes will report ranging in value from zero to one
Assign each of these keys a 1-bit report
Report eight times
The defined byte above is an IN transaction
Report eight bits one time
Input the byte just described as a constant
Report five bits one time
Choose LED usage page
Define five LEDs
The defined bits above are an OUT transaction
Three bit padding for the OUT transaction
Report six bytes
The byte values can range from 0 to 101
Change usage page to key codes
Select key code range of 0 to 101
Input the above six bytes
End application collection

FIGURE 2: USB KEYBOARD DATA FORMAT

<table>
<thead>
<tr>
<th>Byte 7</th>
<th>Byte 6</th>
<th>Byte 5</th>
<th>Byte 4</th>
<th>Byte 3</th>
<th>Byte 2</th>
<th>Byte 1</th>
<th>Byte 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Code 6</td>
<td>Key Code 5</td>
<td>Key Code 4</td>
<td>Key Code 3</td>
<td>Key Code 2</td>
<td>Key Code 1</td>
<td>Reserved</td>
<td>Modifier Keys</td>
</tr>
</tbody>
</table>

MSB

Byte 0
IMPLEMENTATION

Hardware

The PS/2 port is a 6-pin DIN. Only four pins are used:
• Ground
• Power
• Clock
• Data

The power and ground pins are tied directly to VDD and VSS. The clock and data pins are connected to RC0 and RC1, respectively, via current limiting resistors. The clock is driven by the PS/2 keyboard. Figure 3 shows the complete system.

Software

PS/2 DATA FORMAT

In order to better understand how PS/2 keyboard data is translated to USB, it is necessary to touch upon the PS/2 data format. Data is sent via PS/2 one byte at a time regardless of direction, host-to-device or vice-versa. The data has the following form:
• Start bit (always low)
• Data byte (Least Significant bit to Most Significant bit)
• Parity bit (high for an even number of high bits in the data byte and low for an odd number)
• 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 line are held high by the device. See Figure 4 and Figure 5 for device-to-host and host-to-device communication, respectively.

FIGURE 3: PS/2® TO USB KEYBOARD TRANSLATOR HARDWARE DIAGRAM

Note 1: C1 and C2 values selected according to crystal load capacitance.
FIGURE 4: DEVICE-TO-HOST COMMUNICATION (DATA BIT READ ON FALLING EDGE OF CLOCK)

FIGURE 5: HOST-TO-DEVICE COMMUNICATION (DATA BIT READ ON FALLING EDGE OF CLOCK)

PS/2 KEYBOARD REPORT FORMAT
The PS/2 keyboard data report format is summarized for every key in Appendix A. 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 milliseconds, the make code will be sent repeatedly until the user releases the key. At this point, the break code is sent.

INTERRUPT ROUTINE
The translator firmware is entirely interrupt-driven (with the exception of sending the data via USB to the host). An interrupt is generated when the PS/2 Start bit is received, at which time the firmware will begin its receive routine. In addition to this interrupt, every 683 µs a timer overflow interrupts the main program and implements one state of the keyboard state machine. This state machine handles sending bytes to (and translating bytes received from) the PS/2 device automatically. All of this is done in the background while the main program runs in the foreground. The only operation that the main program implements is sending keyboard data to the PC via USB.

TRANSLATION TO USB
Incoming PS/2 keyboard data can be one to eight bytes in length, depending on the button that is pressed. No pattern or mathematical expression can be used to convert incoming PS/2 data to a USB key code. USB key codes are one byte in length. There is one USB key code for every key on the keyboard. A lengthy look-up table, found in file table_kb.asm, translates from PS/2 to USB key codes. The USB key codes are listed in Section 9 of the USB HID Usage Tables (see References).
MAKING THE KEYBOARD LEDS WORK

The state of the LEDs on a USB keyboard is passed from the host to the device via the Set_Report request. The Set_Report request transfers the data specified by the Output items in the report descriptor listed in Figure 1. The data consists of one byte, in which the first five bits represent the LED status. This data is sent from the host to the PICmicro® MCU only when the host receives a key code that modifies the state of a LED, such as the CAPS LOCK key code.

The microcontroller firmware services the Set_Report request. Set_Report requests are serviced in the HidSetReport routine. HidSetReport copies the Set_Report data from the EP0 OUT buffer to the EP1 OUT buffer. This is how the default USB support firmware treats a Set_Report request -- it makes a Set_Report look like an interrupt EP1 OUT transfer. A developer using the support firmware needs only to decide how to service the interrupt EP1 OUT transfer. For the keyboard example, this trick was not utilized. The keyboard state machine simply reads the EP0 OUT buffer to find out the LED states. Special Function Register (SFR) BD0OAL contains the address of this buffer. BD0OBC, also a Special Function Register, specifies the number of bytes received. The LED status report comprises one byte so only the first byte of the buffer needs to be read in order to determine the state of the keyboard LEDs. Once the status is known, this information is relayed to the keyboard via PS/2.

CONCLUSION

The Set_Report request is useful in low-speed USB applications where the maximum amount of bandwidth available is required for IN transactions, yet it is necessary to occasionally support OUT transactions. In addition to providing an example of the Set_Report request, the firmware included with this technical brief provides ready-made routines for communicating via PS/2.

MEMORY USAGE

In the PIC16C765, the following memory was used

Data Memory: 50 Bytes
Program Memory: 2.1K Bytes

REFERENCES

1. USB Specification, Version 1.1: Chapter 9 (located at www.usb.org)
2. Device Class Definition for Human Interface Devices (located at www.usb.org)
3. HID Usage Tables (located at www.usb.org)
5. USB Complete, Second Edition, Jan Axelson; Lakeview Research, 2001 (www.lvr.com)
6. PS/2® Mouse/Keyboard Protocol, Adam Chapweske
8. TB054: An Introduction to USB Descriptors with a Game Port to USB Game Pad Translator (DS91054)
9. TB055: PS/2® to USB Mouse Translator (DS91055)
10. TB057: USB Combination Devices Demonstrated by a Combination Mouse and Game Pad device (DS91057)
11. TB058: Demonstrating the Soft Detach Function with a PS/2® to USB Translator Example (DS91058)
## APPENDIX A: PS/2® KEYCODES

<table>
<thead>
<tr>
<th>Key</th>
<th>Make</th>
<th>Break</th>
<th>Key</th>
<th>Make</th>
<th>Break</th>
<th>Key</th>
<th>Make</th>
<th>Break</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1C</td>
<td>F0,1C</td>
<td>9</td>
<td>46</td>
<td>F0,46</td>
<td></td>
<td>L</td>
<td>54</td>
</tr>
<tr>
<td>B</td>
<td>32</td>
<td>F0,32</td>
<td>7</td>
<td>0E</td>
<td>F0,0E</td>
<td></td>
<td>INSERT</td>
<td>E0,70</td>
</tr>
<tr>
<td>C</td>
<td>21</td>
<td>F0,21</td>
<td>5</td>
<td>4E</td>
<td>F0,4E</td>
<td></td>
<td>HOME</td>
<td>E0,6C</td>
</tr>
<tr>
<td>D</td>
<td>23</td>
<td>F0,23</td>
<td>5</td>
<td>55</td>
<td>F0,55</td>
<td></td>
<td>PG UP</td>
<td>E0,7D</td>
</tr>
<tr>
<td>E</td>
<td>24</td>
<td>F0,24</td>
<td>\</td>
<td>5D</td>
<td>F0,5D</td>
<td></td>
<td>DELETE</td>
<td>E0,71</td>
</tr>
<tr>
<td>F</td>
<td>2B</td>
<td>F0,2B</td>
<td>BKSP</td>
<td>66</td>
<td>F0,6</td>
<td></td>
<td>END</td>
<td>E0,69</td>
</tr>
<tr>
<td>G</td>
<td>34</td>
<td>F0,34</td>
<td>SPACE</td>
<td>29</td>
<td>F0,29</td>
<td></td>
<td>PG DN</td>
<td>E0,7A</td>
</tr>
<tr>
<td>H</td>
<td>33</td>
<td>F0,33</td>
<td>TAB</td>
<td>0D</td>
<td>F0,0D</td>
<td></td>
<td>U ARROW</td>
<td>E0,75</td>
</tr>
<tr>
<td>I</td>
<td>43</td>
<td>F0,43</td>
<td>CAPS</td>
<td>58</td>
<td>F0,58</td>
<td></td>
<td>L ARROW</td>
<td>E0,6B</td>
</tr>
<tr>
<td>J</td>
<td>3B</td>
<td>F0,3B</td>
<td>L SHFT</td>
<td>12</td>
<td>F0,12</td>
<td></td>
<td>D ARROW</td>
<td>E0,72</td>
</tr>
<tr>
<td>K</td>
<td>42</td>
<td>F0,42</td>
<td>L CTRL</td>
<td>14</td>
<td>F0,14</td>
<td></td>
<td>R ARROW</td>
<td>E0,74</td>
</tr>
<tr>
<td>L</td>
<td>4B</td>
<td>F0,4B</td>
<td>L WIN</td>
<td>E0,1F</td>
<td>E0,F0,1F</td>
<td></td>
<td>NUM</td>
<td>77</td>
</tr>
<tr>
<td>M</td>
<td>3A</td>
<td>F0,3A</td>
<td>L ALT</td>
<td>11</td>
<td>F0,11</td>
<td></td>
<td>KP /</td>
<td>E0,4A</td>
</tr>
<tr>
<td>N</td>
<td>31</td>
<td>F0,31</td>
<td>R SHFT</td>
<td>59</td>
<td>F0,59</td>
<td></td>
<td>KP *</td>
<td>7C</td>
</tr>
<tr>
<td>O</td>
<td>44</td>
<td>F0,44</td>
<td>R CTRL</td>
<td>E0,14</td>
<td>E0,F0,14</td>
<td></td>
<td>KP -</td>
<td>7B</td>
</tr>
<tr>
<td>P</td>
<td>4D</td>
<td>F0,4D</td>
<td>R WIN</td>
<td>E0,27</td>
<td>E0,F0,27</td>
<td></td>
<td>KP +</td>
<td>79</td>
</tr>
<tr>
<td>Q</td>
<td>15</td>
<td>F0,15</td>
<td>R ALT</td>
<td>E0,11</td>
<td>E0,F0,11</td>
<td></td>
<td>KP EN</td>
<td>E0,5A</td>
</tr>
<tr>
<td>R</td>
<td>2D</td>
<td>F0,2D</td>
<td>APPS</td>
<td>E0,2F</td>
<td>E0,F0,2F</td>
<td></td>
<td>KP .</td>
<td>71</td>
</tr>
<tr>
<td>S</td>
<td>1B</td>
<td>F0,1B</td>
<td>ENTER</td>
<td>5A</td>
<td>F0,5A</td>
<td></td>
<td>KP 0</td>
<td>70</td>
</tr>
<tr>
<td>T</td>
<td>2C</td>
<td>F0,2C</td>
<td>ESC</td>
<td>76</td>
<td>F0,76</td>
<td></td>
<td>KP 1</td>
<td>69</td>
</tr>
<tr>
<td>U</td>
<td>3C</td>
<td>F0,3C</td>
<td>F1</td>
<td>5</td>
<td>F0,05</td>
<td></td>
<td>KP 2</td>
<td>72</td>
</tr>
<tr>
<td>V</td>
<td>2A</td>
<td>F0,2A</td>
<td>F2</td>
<td>6</td>
<td>F0,06</td>
<td></td>
<td>KP 3</td>
<td>7A</td>
</tr>
<tr>
<td>W</td>
<td>1D</td>
<td>F0,1D</td>
<td>F3</td>
<td>4</td>
<td>F0,04</td>
<td></td>
<td>KP 4</td>
<td>6B</td>
</tr>
<tr>
<td>X</td>
<td>22</td>
<td>F0,22</td>
<td>F4</td>
<td>0C</td>
<td>F0,0C</td>
<td></td>
<td>KP 5</td>
<td>73</td>
</tr>
<tr>
<td>Y</td>
<td>35</td>
<td>F0,35</td>
<td>F5</td>
<td>3</td>
<td>F0,03</td>
<td></td>
<td>KP 6</td>
<td>74</td>
</tr>
<tr>
<td>Z</td>
<td>1A</td>
<td>F0,1A</td>
<td>F6</td>
<td>0B</td>
<td>F0,0B</td>
<td></td>
<td>KP 7</td>
<td>6C</td>
</tr>
<tr>
<td>0</td>
<td>45</td>
<td>F0,45</td>
<td>F7</td>
<td>83</td>
<td>F0,83</td>
<td></td>
<td>KP 8</td>
<td>75</td>
</tr>
<tr>
<td>1</td>
<td>16</td>
<td>F0,16</td>
<td>F8</td>
<td>0A</td>
<td>F0,0A</td>
<td></td>
<td>KP 9</td>
<td>7D</td>
</tr>
<tr>
<td>2</td>
<td>1E</td>
<td>F0,1E</td>
<td>F9</td>
<td>1</td>
<td>F0,01</td>
<td></td>
<td>KP 1</td>
<td>5B</td>
</tr>
<tr>
<td>3</td>
<td>26</td>
<td>F0,26</td>
<td>F10</td>
<td>9</td>
<td>F0,09</td>
<td></td>
<td>4C</td>
<td>F0,4C</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
<td>F0,25</td>
<td>F11</td>
<td>78</td>
<td>F0,78</td>
<td></td>
<td>52</td>
<td>F0,52</td>
</tr>
<tr>
<td>5</td>
<td>2E</td>
<td>F0,2E</td>
<td>F12</td>
<td>7</td>
<td>F0,07</td>
<td></td>
<td>41</td>
<td>F0,41</td>
</tr>
<tr>
<td>6</td>
<td>36</td>
<td>F0,36</td>
<td>PAUSE</td>
<td>E1,14,77</td>
<td>NONE</td>
<td></td>
<td>49</td>
<td>F0,49</td>
</tr>
<tr>
<td>7</td>
<td>3D</td>
<td>F0,3D</td>
<td>E1,F0,14</td>
<td>F0,77</td>
<td>PRNT</td>
<td>E0,12</td>
<td>E0,F0,7C</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3E</td>
<td>F0,3E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCROLL</td>
<td>7E</td>
<td>F0,7E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX B: SOURCE CODE

Due to the length of the source code for the PS/2 to USB Keyboard Translator example, the source code is available separately. The complete source code is available as a single WinZip archive file, tb056sc.zip, which may be downloaded from the Microchip corporate web site at:

www.microchip.com
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families 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 Microchip products in a manner outside the operating specifications contained in Microchip’s Data Sheets. Most likely, the person doing so is 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 products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KeelLoq, MPLAB, PIC, PICmicro, PICSTART, PRO MATE, PowerSmart and rPic are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AmpLab, FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL, SmartShunt and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Application Maestro, dsPICDEM, dsPICDEM.net, dsPICWorks, ECAN, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICkit, PICDEM, PICDEM.net, Pictail, PowerCal, PowerInfo, PowerMate, PowerTool, rLAB, Select Mode, SmartSensor, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2004, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.

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

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company’s quality system processes and procedures are for its PICmicro® 8-bit MCUs, KeelLoq® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip’s quality system for the design and manufacture of development systems is ISO 9001:2000 certified.
**AMERICAS**

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

**Atlanta**  
3780 Mansell Road, Suite 130  
Alpharetta, GA 30022  
Tel: 770-640-0034  
Fax: 770-640-0307

**Boston**  
2 Lan Drive, Suite 120  
Westford, MA 01886  
Tel: 978-692-3848  
Fax: 978-692-3821

**Chicago**  
333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071  
Fax: 630-285-0075

**Dallas**  
4570 Westgrove Drive, Suite 160  
Addison, TX 75001  
Tel: 972-818-7423  
Fax: 972-818-2924

**Detroit**  
Tri-Atria Office Building  
32255 Northwestern Highway, Suite 190  
Farmington Hills, MI 48334  
Tel: 248-538-2250  
Fax: 248-538-2260

**Kokomo**  
2767 S. Albright Road  
Kokomo, IN 46902  
Tel: 765-864-8360  
Fax: 765-864-8387

**Los Angeles**  
18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 949-263-1888  
Fax: 949-263-1388

**San Jose**  
1300 Terra Bella Avenue  
Mountain View, CA 94043  
Tel: 650-215-1444  
Fax: 650-961-0286

**Toronto**  
6285 Northam Drive, Suite 108  
Mississauga, Ontario L4V 1X5, Canada  
Tel: 905-673-0699  
Fax: 905-673-6509

**ASIA/PACIFIC**

**China - Beijing**  
Unit 706B  
Wan Tai Bei Hai Bldg., No. 6 Chaoyangmen Bei Str.  
Beijing, 100027, China  
Tel: 86-10-85282100  
Fax: 86-10-85282104

**China - Chengdu**  
Rm. 2401-2402, 24th Floor, Ming Xing Financial Tower No. 88 TIDU Street  
Chengdu 610016, China  
Tel: 86-28-88768200  
Fax: 86-28-88766599

**China - Fuzhou**  
Unit 28F, World Trade Plaza No. 71 Wusi Road  
Fuzhou 350001, China  
Tel: 86-591-7503506  
Fax: 86-591-7503521

**China - Hong Kong SAR**  
Unit 901-6, Tower 2, Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2401-1200  
Fax: 852-2401-3431

**China - Shanghai**  
Room 701B, Bldg. B  
Far East International Plaza No. 317 Xian Xia Road  
Shanghai, 200021, China  
Tel: 86-21-6275-5700  
Fax: 86-21-6275-5560

**China - Shenzhen**  
Rm. 1812, 18/F, Building A, United Plaza No. 5022 Binhe Road, Futian District  
Shenzhen 518033, China  
Tel: 86-755-82901380  
Fax: 86-755-8295-1393

**China - Taiwan**  
Room 28F , World Trade Plaza  
No. 71 Wusi Road  
Fuzhou 350001, China  
Tel: 86-591-7503506  
Fax: 86-591-7503521

**Korea**  
168-1, Youngho Bldg, 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea 135-882  
Tel: 82-2-554-7200  
Fax: 82-2-558-5932  
or 82-2-558-5934

**Singapore**  
200 Middle Road  
#07-02 Prime Centre  
Singapore, 189980  
Tel: 65-6334-8870  
Fax: 65-6334-8850

**Taiwan**  
Kaohsiung Branch  
3F - 1 No. 8  
Min Chuan 2nd Road  
Kaohsiung 806, Taiwan  
Tel: 886-7-536-4818  
Fax: 886-7-536-4803

**EUROPE**

**Austria**  
Durisolstrasse 2  
A-4600 Wels  
Austria  
Tel: 43-7242-2244-399  
Fax: 43-7242-2244-399

**Denmark**  
Regus Business Centre  
Lautrup høj 1-3  
Ballere DK-2750 Denmark  
Tel: 45-4420-9895 Fax: 45-4420-9910

**France**  
Parc d’Activite du Moulin de Massy  
43 Rue du Saule Trape  
Batiment A - Ier Etage  
91300 Massy, France  
Tel: 33-1-69-53-63-20  
Fax: 33-1-69-30-90-79

**Germany**  
Steinheilstrasse 10  
D-65737 Tarnsberg, Germany  
Tel: 49-99-627-144-0  
Fax: 49-99-627-144-44

**Italy**  
Via Quasimodo, 12  
20025 Legnano (MI)  
Mila, Italy  
Tel: 39-0331-742611  
Fax: 39-0331-46789

**Netherlands**  
P. A. De Biesbosch 14  
NL-5152 SC Drunen, Netherlands  
Tel: 31-1-69-53-63-20  
Fax: 31-1-69-30-90-79

**United Kingdom**  
505 Eskdale Road  
Winkers Triangle  
Wokingham  
Berkshire, England RG41 5TU  
Tel: 44-118-921-5869  
Fax: 44-118-921-5820

02/17/04