AN1064

IR Remote Control Transmitter

Author: *Tom Perme*
John McFadden
Microchip Technology Inc.

INTRODUCTION

This application note illustrates the use of the PIC10F206 to implement a two-button infrared remote controller. The PIC10F2XX family of microcontrollers is currently the smallest in the world, and their compact sizes and low cost make them preferable for small applications such as this one.

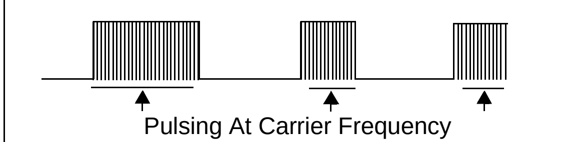
Two example protocols are shown. The first is Philips® RC5, and the second is Sony™ SIRC. These two protocols were chosen because they are fairly common and their formats are well documented on professional and hobbyists' web sites. They also demonstrate two differing schemes for formatting the transmission.

THEORY OF OPERATION

Infrared transmitter and receiver pairs have been used for many years in television, stereo and home theater remote controls. The infrared spectrum provides less ambient noise than other spectrums of light, like visible light, and this makes infrared ideal for inexpensive reliable communication.

To create an IR control link, one needs a transmitter, a receiver, and a protocol for how to communicate between them. The transmitter sends pulses through an infrared LED. These pulses modulate a carrier frequency in a pattern defined by the protocol (Figure 1). This modulation improves the SNR at the receiver. The IR light then travels through the air and is detected at the receiver by a photo-diode. The photo-diode is often contained in a complete module which demodulates the modulated signal for a given carrier frequency. This module outputs a logic level signal, but it contains no timing information. Recovering the timing and determining if a bit is a "`1`" or a "`0`" are the remaining steps to receive a signal. However, this application note will focus on the transmitter only.

FIGURE 1: MODULATED SIGNAL



Note:	Some common protocols are Philips® RC5, NEC®, Sony™ SIRC, and Matsushita®.

PHILIPS RC5 PROTOCOL

The RC5 protocol is a common IR transmission protocol. It uses a 14-bit transmission consisting of 2 start bits, a toggle bit, 5 address bits, and 6 data bits. This 14-bit transmission forms a packet and is repeated every 114 ms while the button initiating the transmission is held down (Figure 2). Once released, a final transmission is sent indicating a button's release.

FIGURE 2: RC5 CONTINUOUS TRANSMISSION

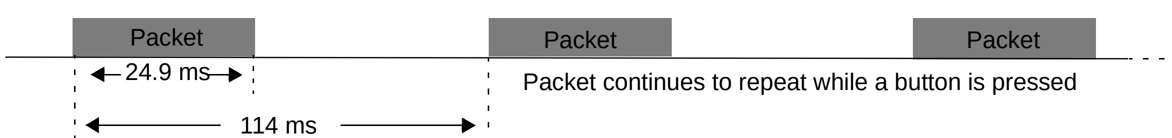
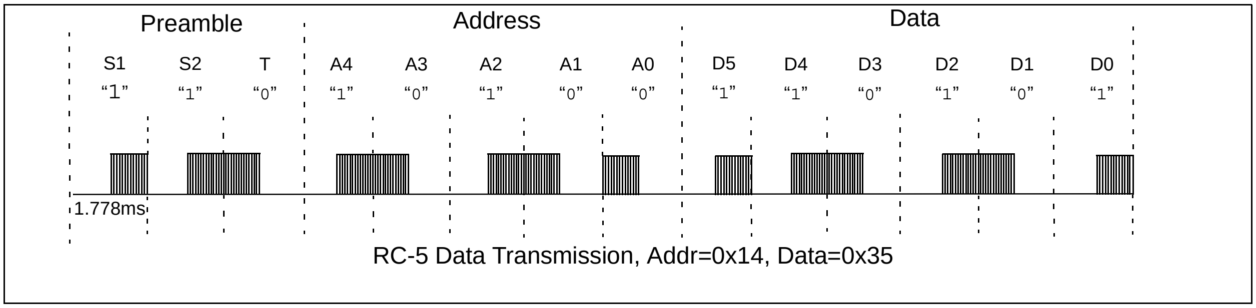


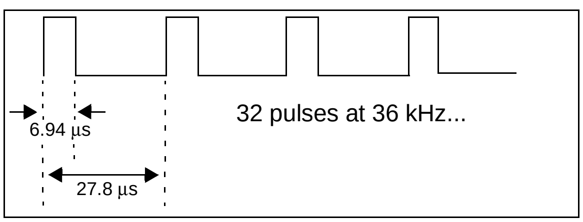
FIGURE 3: RC5 TRANSMISSION PACKET EXAMPLE



The first two bits, S1 and S2 are Start bits. Each bit is always a one. The third bit, T, is a toggle bit. The toggle bit is used to indicate if a button is pushed down and held down, and when the button is released, the bit is toggled back to a one for a final packet transmission. The five bits for *Address* comprise the device address to receive the data. For example, this device address specifies if the television or the stereo should react to the message. The last 6 bits, *Data*, comprise the command sent to the addressed device, and they dictate how the device reacts.

The formatting for ones and zeros is Manchester encoding. As seen in Figure 3, a "1" transitions from low-to-high during the bit period. A "0" does the opposite; it transitions from high-to-low. Also note the bit period is 1.778 ms. This means, during a high period the signal will pulse at the carrier frequency for half of that time, or 889 µs. The pulsing carrier signal should have a duty cycle of ¼ at a frequency of 36 kHz (Figure 4).

FIGURE 4: CARRIER TIMING



The duty cycle of the carrier frequency does not need to be exact. A 25% duty cycle reduces power consumption, but transmitting at 50% duty cycle would boost power output. The carrier frequency is critical for the demodulation process, and an accuracy of +/- 1% for the carrier frequency should be sufficient. The problem with straying too far from the carrier frequency occurs because there is a band-pass filter centered at the carrier frequency in the receiver's demodulation module. It acts to reduce noise by attenuating frequencies outside of the BPF, such as ambient light and other sources of interference. For a packaged demodulator, its data sheet will specify in some form, the range of frequencies it accepts, such as a -3dB point for the BPF about the center frequency.

An important aspect for improving efficiency is matching the wavelength of light a transmitter LED sends to the receiver module. If an IR LED emits light at 890 nm and the demodulator works best at 950 nm, then compared to an LED sending 950 nm wavelength light, the 890 nm LED will use more power transmitting to achieve the same received signal power. In short, the value in matching the carrier frequency, duty cycle and wavelength to the receiver's optimal specifications are increased efficiency and increased distance of reliable transmission.

For the provided RC5 software, the carrier frequency is actually 35.7 kHz due to rounding the values in Figure 4 from 6.94 µs and 27.8 µs to 7 µs and 28 µs. This results in a difference from ideal of -0.83%, which is an acceptable amount.

SONY 12-BIT PROTOCOL

Another protocol is Sony SIRC, which is used in Sony devices. There are 12-bit, 15-bit and 20-bit versions of the Sony code, but the 12-bit was chosen because it is common and simple to implement.

The Sony protocol also modulates a carrier, but the key distinguishing feature is the representation of the ones and zeros. The Sony protocol uses a pulse-width modulation (PWM) scheme. In this case, it means that a one has a longer bit period than a zero. Shown below in Figure 5, a logical "1" will modulate the carrier for 1.2 ms, twice as long as the zero which is modulated for 600 µs on-time.

FIGURE 5: SONY™ BIT REPRESENTATIONS

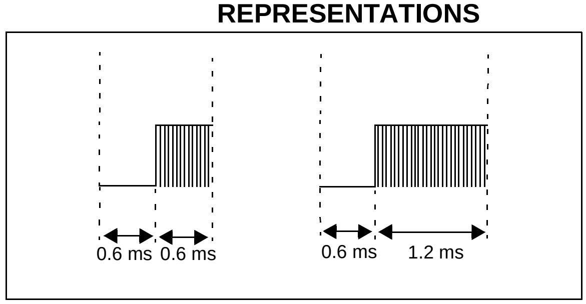


FIGURE 6: SONY™ CONTINUOUS TRANSMISSION

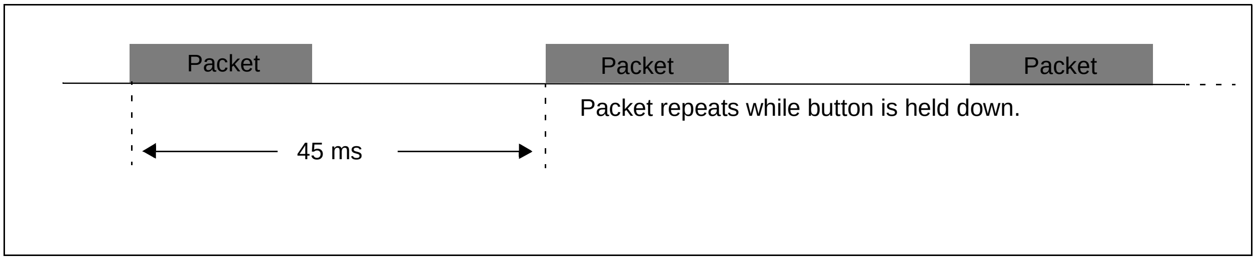


FIGURE 7: SONY™ TRANSMISSION PACKET EXAMPLE

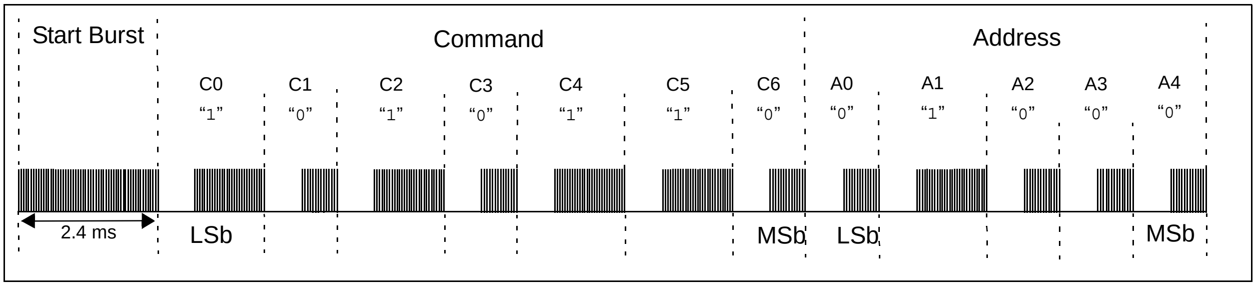
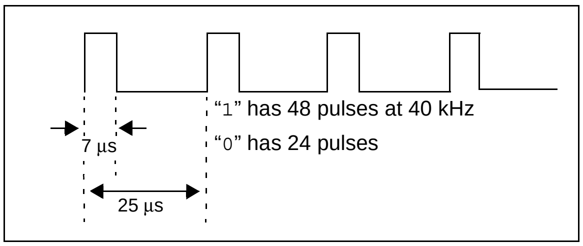


FIGURE 8: SONY™ CARRIER TIMING



The carrier frequency is 40 kHz, which equates to 24 pulses during a zero's high time (600 µs) and 48 pulses during a one's high time (1.2 ms). The duty cycle should be about 25%.

The formatting of the packet is also different. A Sony packet repeats after 45 ms while a button is held down, and it contains a "start burst" of 2.4 ms followed by a 7-bit command and a 5-bit address sent LSB first. The start burst distinguishes the start of a packet, and it also allows for a receiver to adjust its gain for varying received signal levels. The repeating packet is shown in Figure 7.

IMPLEMENTATION

All discussion so far has been to specify protocols to be implemented. These protocols are timing sensitive, and so all the proper rates and tolerances should be heeded. Demodulators can handle reasonable timing errors, but they are tuned to ignore gross errors. Because imperfections are allowable, this can make writing the software for a PIC® microcontroller easier.

For the provided method to create transmission timings, it is difficult to obtain a carrier frequency of exactly 36 kHz for RC5 (T=27.8 µs). Instead it is much easier to make the frequency 35.7 kHz (T=28 µs) while still maintaining reliable performance. For SIRC, because the period of 40 kHz is 25 µs, an integral value, it is easy to represent exactly 40 kHz using twenty-five 1 µs instruction cycles.

Note:	A reasonable and attainable level of accuracy for timing is 1%. Check with the datasheet of the IR demodulator to confirm what it is capable of.

Figure 9 shows a common method to drive an LED. Because infrared LEDs typically require current on the order of 50-100 mA, driving the LED directly from the output of the microcontroller's pin is insufficient. Using a transistor as a switch to allow current to flow through the LED allows larger currents.

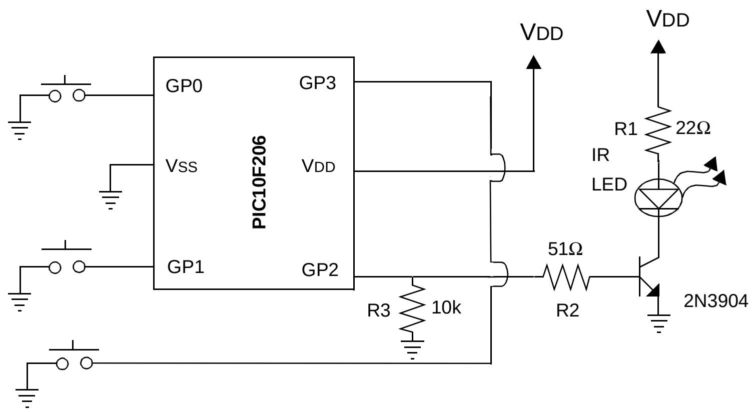
Creating the Waveforms

To create the waveforms on PIC microcontroller baseline or mid-range parts, it is easy to count the instruction cycles during which a signal is set high or low, and then after a certain period change the output on the pin from high-to-low, or vice versa.

All parts with an internal oscillator can use their internal 4 MHz clock for the oscillator. Since the oscillator is 4 MHz, the instruction cycle frequency is 1 MHz, which has a 1 µs period. This makes it easy to create the periods of time desired by the transmission protocols because they both have wave characteristics near whole intervals of microseconds (e.g., 7 µs and 28 µs for RC5's carrier, 7 µs and 25 µs for SIRC).

Note:	Every PIC assembly instruction takes 1 instruction cycle to execute except branch type instructions, which take 2 instruction cycles.

FIGURE 9: SCHEMATICS – BJT DRIVEN LED



When a button press is detected, the program loads the appropriate address and data to be sent, and it then calls a subroutine, either SendSONY or SendRC5. In either case, each send routine calls its own version of subroutines SendOne or SendZero. These subroutines are responsible for creating only one bit of a waveform, such as seen in Figure 5.

For clarity, only the RC5 will be discussed hereon in this section. However, the Sony protocol was implemented in an analogous way, just with its own bit times, data format, and carrier frequency. Any custom format may also be done similarly with its own specifications.

Creating a Bit's Waveform

For RC5, each bit has a period where it is modulated and a period where it is null. The modulated portion comes first for a "`0`", and then is followed by zero output, implemented via the code in Figure 10 followed by Figure 11. For a "`1`" there is zero output transitioning to the modulated output, Figure 11 followed by Figure 10.

Creating the modulated portion is the most critical task when counting clock cycles. The carrier needs 32 pulses at 36 kHz as per Figure 4. So we will loop 32 times overall, and each time we will set the `OUTPUT_LED` to turn on for 7 µs (*`bsf OUTPUT_LED`*), and then off for 21 µs (*`bcf OUTPUT_LED`*). The sum totals 28 µs. The first two instructions are overhead, which should be accounted for in whatever precedes the carrier loop.

FIGURE 10: MODULATING CARRIER CODE

```
     MOVLW          d'32'
     MOVWF          Delay_Count2
CarrierLoopOne:
     BSF            OUTPUT_LED
     GOTO           $+1
     GOTO           $+1
     GOTO           $+1
     BCF            OUTPUT_LED
     MOVLW          d'5'
     MOVWF          Delay_Count
     DECFSZ         Delay_Count, F
     GOTO           $-1
     NOP
     DECFSZ         Delay_Count2, F
     GOTO           CarrierLoopOne
```

To create the null time for the Manchester encoding, ensure that the output is low, and then loop for 889 µs. This requires two loops when running at a 1 MHz instruction cycle, and then to be precise, 2 more instruction cycles on the end (*`GOTO` $+1 = 2 `NOP`'s)* as seen in Figure 11.

FIGURE 11: NULL-TIME CODE

```
   BCF         OUTPUT_LED

   MOVLW       0xFF
   MOVWF       Delay_Count
   DECFSZ      Delay_Count, F
   GOTO        $-1
   MOVLW       D'39'
   MOVWF       Delay_Count
   DECFSZ      Delay_Count, F
   GOTO        $-1
   GOTO        $+1
```

Note: `GOTO` $+1 takes two instruction cycles to get to the next instruction. This is equivalent to 2 `NOP` instructions.

Sending the Data Packet

Given routines for sending a one or zero, the last step is to use these routines to send a device address and data corresponding to a button press. In the example programs, there will be two bytes for this info.

FIGURE 12: BYTE VARIABLES

```
DataByte      ;Data or Command
AddrByte      ;Device Address
```

On detection of a button press, the programmer defines what each button does, and what device it should control. The illustration in Figure 3 shows AddrByte=0x14 and DataByte=0x35. The SendRC5 subroutine expects these values to be valid each time it is called; so ensure that they are loaded before each call. The SendRC5 routine then shifts out the MSb of the five bit address one at a time into the carry bit, C, of the STATUS register. Then, by checking the carry bit value, it sends the appropriate value; it sends a "`1`" if C=`1`, and a "`0`" if C=`0`. Code for sending a single bit is shown below.

FIGURE 13: SENDING A BIT

```
RLF        AddrByte, F   ;Note 1
BTFSS      STATUS, C
CALL       SendZero
BTFSC      STATUS, C     ;Note 2
CALL       SendOne
```

> **Note 1:** RC5 uses `RLF` because it sends bits MSb first. Sony uses `RRF` to send bits LSb first.
>
> **2:** Testing the carry bit after a subroutine assumes the commands in the subroutine do not modify the carry flag.

After the preamble, address, and data are sent, the packet must repeat every 114 ms. So, to make the SendRC5 routine easy-to-use, it delays the remaining time after the packet time in order to total 114 ms. Then an immediate recall of SendRC5 (with proper input values), sends the packet again.

Requirements for Software

The software provided was written for PIC microcontroller baseline parts, and is easily portable to any other PIC microcontroller baseline or mid-range part. It should require very little work to migrate to another part. The include file for the specific device must be changed and the I/O pins must be configured properly. There are only two requirements to use the code as is. First, the oscillator must run at 4 MHz for the timing to be correct, and second, an interrupt may not be serviced during a transmission.

1. FOSC = 4 MHz
2. No interrupts during transmission

If the code is going to be heavily augmented, take care that the overhead time between each bit is small enough that the high-pulse time meets protocol specifications.

The implementation of the code can be changed as well. The clock cycles may be recounted for a different speed oscillator, the OSCCAL register may be adjusted to tune the oscillator to specific needs, or a timer with interrupt can be used to create pulses for ones and zeros on mid-range parts. In the last case, interrupts then become 'OK' if properly handled. The attached code is intended to be maximally portable and have accurate timing with its 4 MHz oscillator requirement, and thus counting cycles was chosen as an available means for all processors that can run at 4 MHz.

CONCLUSIONS

An infrared transmitter is simple in concept and easy to implement. By modifying the provided code, the bit timings and shapes may be changed to suit any format.

Also, the example shown only uses a few button inputs. Having more button inputs allows an increased number of commands to be sent to a device.

Using a microcontroller to create the transmitted waveform instead of an ASIC allows for customizability and additional functionality. A microcontroller may also be programmed to transmit several formats while using a single hardware configuration.

MEMORY USAGE

The memory usage for the two programs provided to transmit Sony™ SIRC and Philips® RC5 is shown below. The Sony™ protocol takes more instructions to send the packet properly, thereby increasing the total number words required.

Philips® RC5

ir_tx_RC5.asm 189 words

Sony™ SIRC

ir_tx_SONY.asm 197 words

GLOSSARY OF TERMS

Acronym	Description
IR	Infrared
SNR	Signal to Noise Ratio
LED	Light Emitting Diode
MSb	Most Significant bit
LSb	Least Significant bit

NOTES:

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.

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

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona, Gresham, Oregon and Mountain View, California. The Company's quality system processes and procedures are for its PIC® 8-bit MCUs, KEELOQ® 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.

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
Habour 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 - Fuzhou
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

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

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 - Shunde
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

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

China - Xian
Tel: 86-29-8833-7250
Fax: 86-29-8833-7256

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 - Gumi
Tel: 82-54-473-4301
Fax: 82-54-473-4302

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

Malaysia - Penang
Tel: 60-4-646-8870
Fax: 60-4-646-5086

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

12/08/06