AN1565

Configuring and Using the WDT Module of MCP795WXX Real-Time Clock/Calendars

Author: *Alexandru Valeanu*
Microchip Technology Inc.

INTRODUCTION

This application note is designed to take the design engineer through the steps needed to configure and setup the Watchdog Timer (WDT) module of Microchip SPI MCP795WXX Real-Time Clock/Calendar (RTCC) family. Topics covered include:

- Configuring the WDT module
- Using the WDT module
- Overview of the provided drivers/libraries

The information presented in this document is designed serve as an example of possible configurations. The code supplied can be modified to change the device functionality.

This application note should be read in conjunction with the application note AN1365 – *"Recommended Usage of Microchip Serial RTCC Devices"* (DS 00001365) and the device data sheet. The latest documentation can be found at this location on the Microchip website: http://www.microchip.com/rtcc.

WHAT IS THE WDT MODULE?

A WDT is an electronic module that performs a specific task (such as generating an interrupt) after a programmed period of time, if a critical event does not appear before the end of the delay period. WDTs are most commonly used to detect when the software enters an unintended operation and then helps it recover.

If the software cannot clear the WDT in time, it is usually because two parts or programs conflict, a process is stuck, or an important event did not occur in time.

A WDT has a digital counter that counts up to its programmed limit. If the software loop does not clear the WDT at regular intervals of time (shorter than the WDT period), the WDT will launch an interrupt, forcing/resetting the system to a known state.

The Watchdog Timer in the MCP795WXX is a free-running counter that operates from the crystal. This counter is independent of the time and date counters and can be used to generate an interrupt after a programmable period of time.

The counter can be reset at any time using either an SPI bus command or by toggling an external pin (EVHS), if configured. Note that using the EVHS pin in this manner prevents the use of the high-speed event detect feature.

SCHEMATIC DIAGRAM

The schematic illustrated in Figure 1 shows the minimum components required to operate the RTCC.

The schematic also shows the required components for battery backup operation using a lithium coin cell (for other options refer to AN1365). If the V_{BAT} input and battery backup feature are not required, this pin should be tied to V_{SS}.

The SPI RTCC is directly connected to the MSSP1 module of the PIC18 MCU. There is an additional connection between the $\overline{WDO}$ signal of the RTCC and the RB2/INT2 pin of the MCU. The RTCC is selected through the RC2 GPIO of the MCU.

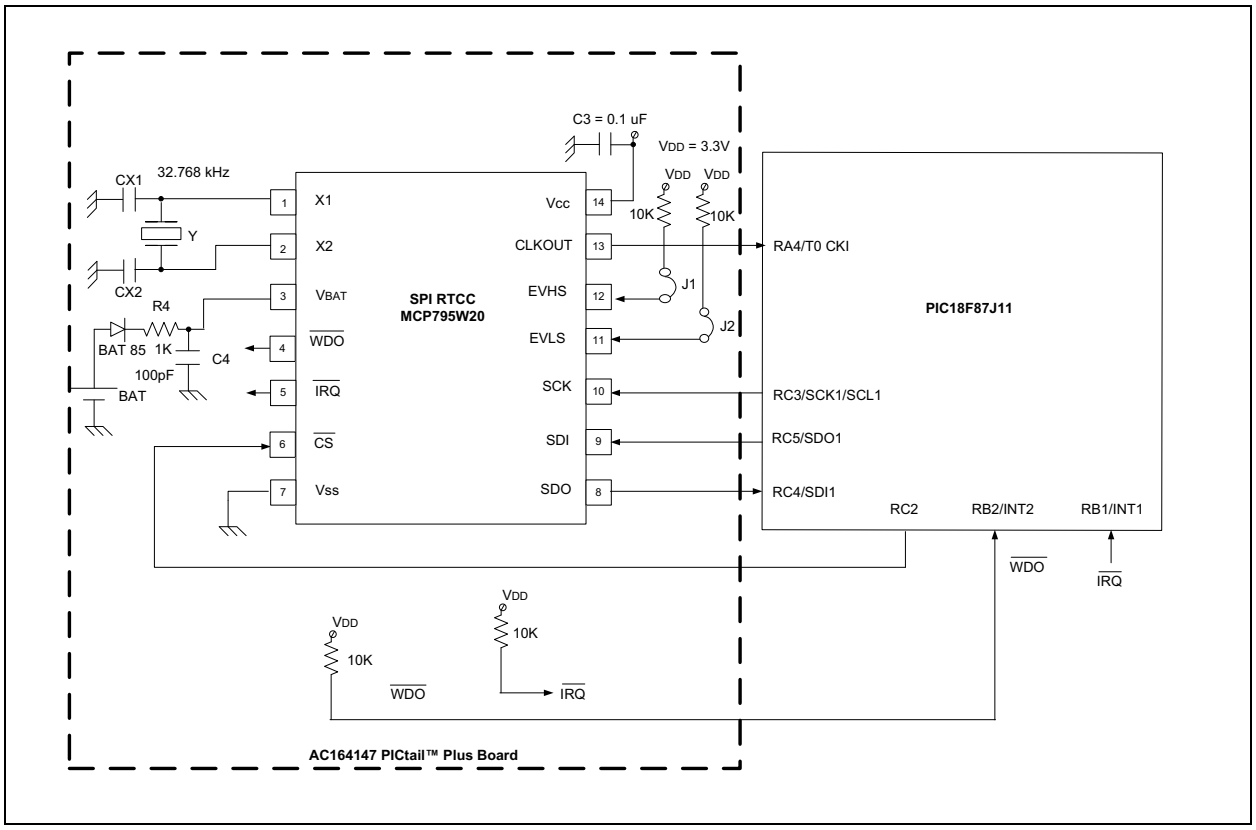
In the provided code, the following signal is defined:

```
#define NCS_SPI_RTCC PORTCbits.RC2
// CS for the SPI RTCC.
```

The RTCC offers interrupts at a programmed time interval on the $\overline{WDO}$ pin, connected to the INT2 pin (external hardware interrupt 2) of the MCU. The related interrupt functions are used and described in the main function.

The easiest way to start the project is by using the related SPI RTCC PICtail™ Plus Board (AC164147).

FIGURE 1: SCHEMATIC



FIRMWARE OVERVIEW

The host MCU will communicate with the RTCC using the SPI protocol, based on the MSSP1 serial module of the PIC18 MCU.

The code presented with this application note is designed to compile with the XC8 (V 1.34) compiler and MPLAB® X IDE (V3.55) for the following hardware:

- Explorer 18 Evaluation Board (DM183032)
- PIC18F87J11 PIM Module (MA180020)
- RTCC SPI PICtail Module (AC164147)

The code is presented in C and is portable with minimal effort to other PIC© MCU devices. This code is designed to be a starting point for application development and is based on the drivers/libraries presented in this text and the related code.

APPLICATION DESCRIPTION

The code presents the basics of the WDT's configuration and usage. After initializing the RTCC and the WDT, the code displays the value of the WDT register in binary on LEDs. The code then waits for the time-out period to expire, at which point all of the LEDs are turned on to indicate that the WDT has triggered.

CONFIGURATION OF THE WDT MODULE

TABLE 1: WATCHDOG REGISTER

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
WDTEN	WDTIF	WDTDLYEN	WDTPWS	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’

bit 7 **WDTEN:** Watchdog Timer Enable bit
This is a read/write bit that is set by the user and can be cleared by the user of the hardware. This bit is set to enable the WDT function and cleared to disable the function. This bit is cleared by the hardware when the VCC supply is not present, it is not set again when VCC is present.

bit 6 **WDTIF:** Watchdog Timer Interrupt Flag bit
This is a read/write bit that is set in hardware when the WDT times out and the WD pin is asserted. This bit must be cleared in software to restart the WDT.

bit 5 **WDTDLYEN:** Watchdog Timer Delay bit
This is a read/write bit and is set to enable a 64-second delay before the WDT starts to count. If this bit is set and the WDTIF bit is cleared then there will be a 64 second delay before the WDT starts to count. This bit should be set before the WDTEN bit is set.

bit 4 **WDTPWS:** Watchdog Timer Reset Pulse Width bit
This is a read/write bit that is used to select the pulse width on the WD pin when the WDT times out.
0 = 122 µs Pulse
1 = 125 ms Pulse

bit 3-0 **WDTPS<3:0>:** Watchdog Timer Configuration bits
These are read/write bits that are used to set the WDT time-out period as below (all times are based off the uncalibrated crystal reference). Bit 3 should be cleared and is reserved for future use:
000 = 977 µs
001 = 15.6 ms
010 = 62.5 ms
011 = 125 ms
100 = 1s
101 = 16s
110 = 32s
111 = 64s

To program the register, several masks were used, as shown in Example 1:

EXAMPLE 1:

```
#define WDTEN          0x80            // WDT enable bit
#define WDTDLYEN       0x20            // enable WDT delay
#define WDTIF          0x40            // WDT interrupt flag
#define WDTPWS         0x10            // WDT pulse-width: 125 ms
                                       // (0 value -> 122 µs)
                                       // WDT time-out periods
#define WDT_TOP_00     0x00            // 977 µs
#define WDT_TOP_01     0x01            // 15.625 ms
#define WDT_TOP_02     0x02            // 62.5 ms
#define WDT_TOP_03     0x03            // 125 ms
#define WDT_TOP_04     0x04            // 01 sec
#define WDT_TOP_05     0x05            // 16 sec
#define WDT_TOP_06     0x06            // 32 sec
#define WDT_TOP_07     0x07            // 64 sec
#define WDTCON         0x0a            // the address of the WDT register
```

A representation of how to initialize the WDT register is shown in Example 2:

EXAMPLE 2:

```
spi_rtcc_wr(WDTCON, WDTEN + WDTDLYEN + WDTPWS + WDT_TOP_00)
```

Table 2 presents all possible combinations of period and pulse width values, depending on the bits: WDTDLYEN, WDTPWS, WDTPS<3:0> bits, included in the Watchdog register.

TABLE 2:

WDTDLYEN	WDTPWS	WDTPS<3:0>	Total Period	Pulse Width
0	0	0-7	977 µsec, 15.625 msec, …….. , 64 sec	122 µsec
0	1	4-7(1)	1 sec, 16 sec, 32 sec, 64 sec	125 msec
1	0	0-7	64 sec + [977 µsec, 15.625 msec,.., 64 sec]	122 µsec
1	1	0-7	64 sec + [977 µsec, 15.625 msec,.., 64 sec]	125 msec

Note 1: The user must ensure that the WDT period is longer than the pulse width. For example, if the pulse width is 125 msec and the delay is '0', the WDT time-out period must be longer than one second.

RE-ENABLING THE WDT

To allow repetitive interrupts (coming from the WDT), the related WDT interrupt flag must be cleared after every action. Two methods could be used:

- Write the initial configuration value again
  `spi_rtcc_wr(WDTCON, wdt_mask);`

`// including WDTEN=1, WDTIF = 0;`

- Clear only the interrupt flag
  `rtcc_buff=spi_rtcc_rd(WDTCON);`

`spi_rtcc_wr(WDTCON,(rtcc_buf&~WDTIF));`

CLEARING THE WDT

When the state of the WDT counter is unknown, or whenever a WDT time out and subsequent interrupt are not desired, the WDT counter must be cleared. This is performed using the MCP795WXX's CLRWDT command. The related `#define directive` is shown in Example 3, while Example 4 shows the function in the provided code that executes the CLRWDT command.

EXAMPLE 3:

```
#define  SPI_RTCC_CLRWDT   0x44       // SPI CLEAR WDT  COMMAND
```

EXAMPLE 4:

```
void spi_rtcc_clw(void){              // clear WD timer/counter
spi_rtcc_start()               ;      // START SPI COMMUNICATION
spi_wrbyte(SPI_RTCC_CLRWDT)    ;      // launch the CLEAR WDT command
spi_rtcc_stop()                ;  }   // stop SPI COMMUNICATION
```

CONCLUSION

Following the steps in this application note, along with the included MPLAB X IDE project, will help setup the basic configuration and usage of the WDT module available on the Microchip MCP795WXX SPI RTCC. By using available off-the-shelf development tools, any hardware issues will be mitigated, allowing the engineer to concentrate on the firmware development.

The code is presented in C (XC8-V1.34 compiler on MPLAB X IDE-V3.55) and can easily be ported to other PIC device platforms.

APPENDIX A: MAIN FUNCTION OF THE WDT TEST/USAGE CODE

```
wdt_mask = WDTPWS+WDT_TOP_05       ; // mask to write in the WDT register
                                     // PLS = 125 msec ; period = 16 sec
                                    // you may write any other combination presented
                                     // in the text of the application note
wdt_mask |= WDTEN                  ; // enable the WDT
LATD = wdt_mask                    ; // display the control byte on LEDs
num_per = 0x00                     ; // init the counter of WDT periods
ini_spi_rtcc()                     ; // init the SPI RTCC
spi_rtcc_wr(WDTCON,wdt_mask)       ; // set the WDT register, enabling the module
ini_wdt_int2()                     ; // init   INTR2 on PIC18 (for the WDT)
start_wdt_int2()                   ; // enable INTR2 on PIC18 (for the WDT)
while(num_per==0) {;}                // waiting the first interrupt from WDT
stop_wdt_int2()                    ; // stop interrupts from WDT / RTCC
spi_rtcc_wr(WDTCON,0x00)           ; // clear WDT register, disable WDT.
LATD = 0xff                        ; // message for the end of the WDT period
while(1) { ; }                       // loop for ever
```

APPENDIX B: DRIVERS LIBRARY

Delay Drivers (`delay_drivers.h`)

Not accessed in the present project. Used by the LCD drivers or as general purpose timing functions.

Represent a good starting point for further development on PIC18-based projects, as they use the timers of the MCU (see AN1355, AN1364, AN1413 for the usage of these drivers).

LCD Drivers (`lcd_drivers.h`)

Not accessed in the present project. Represent a good starting point on Explorer18-based projects (see AN1355, AN1364, AN1413 for the usage of these drivers).

SPI Drivers (`spi_drivers.h`)

Represent the low-level communication between the MSSP1 module of the PIC18 and the SPI RTCC.

The related functions will be detailed in the next paragraph, as called functions.

SPI RTCC Drivers (`spi_rtcc_drivers.h`)

Represent the medium-level communication between the MSSP1 module of the PIC18 and the SPI RTCC.

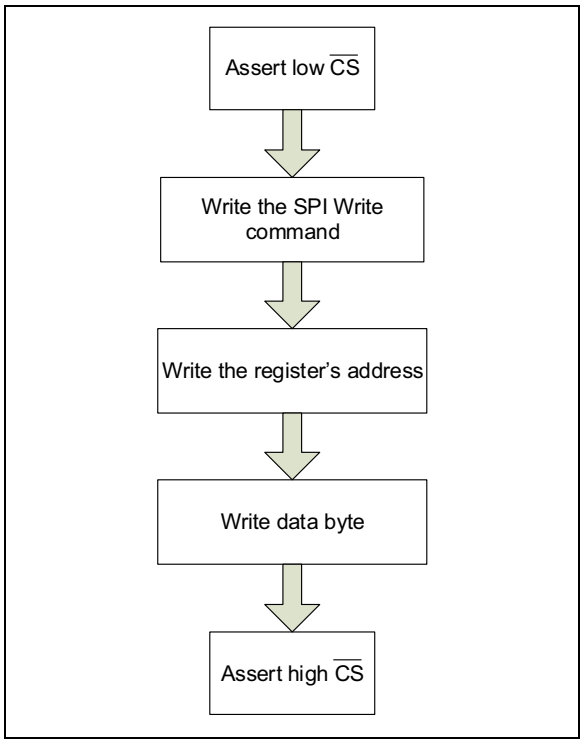
The related functions call the SPI drivers, as described below. Moreover, the library defines all necessary constants, as register addresses and masks.

EXAMPLE 5: WRITING A BYTE TO THE SPI RTCC

```
void spi_rtcc_wr (unsigned char rtcc_reg, unsigned char time_var){
                                          // SPI write to the SPI RTCC
spi_rtcc_start()                        ;   // start SPI comm with the SPI RTCC
spi_wrbyte(SPI_RTCC_WRITE)              ;   // send the SPI WRITE command
spi_wrbyte(rtcc_reg)                    ;   // send the register's address
spi_wrbyte(time_var)                    ;   // send the SPI data byte
spi_rtcc_stop()                         ; } // stop SPI comm
```

The firmware for writes to the RTCC follows in Example 6.

EXAMPLE 6:

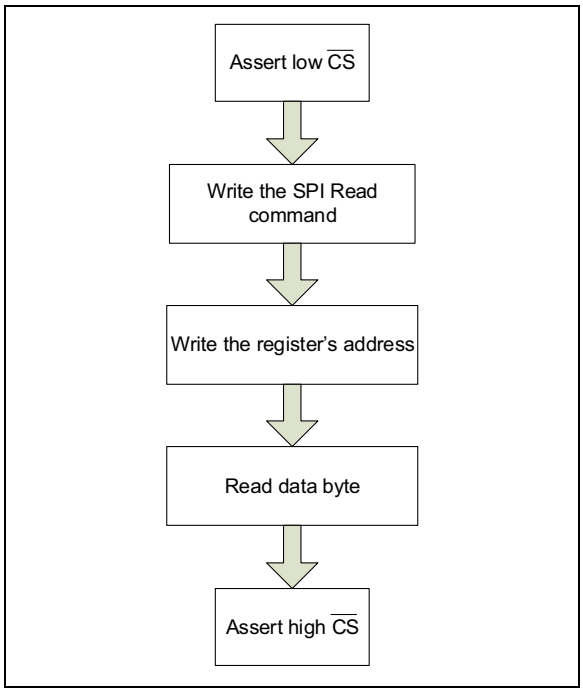


EXAMPLE 7: READING A BYTE FROM THE SPI RTCC

```
unsigned char spi_rtcc_rd(unsigned char rtcc_reg){
                                              // SPI read from the SPI RTCC
spi_rtcc_start()                            ;  // start the SPI comm with the SPI RTCC
spi_wrbyte(SPI_RTCC_READ)                   ;  // send the SPI READ command
spi_wrbyte(rtcc_reg)                        ;  // send the register's address
rtcc_buf = spi_rdbyte()                     ;  // read the result and store it
spi_rtcc_stop()                             ;  // stop the SPI comm with the SPI RTCC
return rtcc_buf                             ;} // return the read result
```

The firmware for reads from the RTCC is shown in Example 8.

EXAMPLE 8:



APPENDIX C: REVISION HISTORY

Revision A (09/2013)

Initial release of this document.

Revision B (10/2017)

Updated register and bit names.

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.

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, 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.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Trademarks

ISBN: 978-1-5224-2209-9

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

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

Austin, TX
Tel: 512-257-3370

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
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

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

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

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

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

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

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

China - Shanghai
Tel: 86-21-3326-8000
Fax: 86-21-3326-8021

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

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

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

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

ASIA/PACIFIC

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

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

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

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

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

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

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

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

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

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

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

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600
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

Finland - Espoo
Tel: 358-9-4520-820

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

France - Saint Cloud
Tel: 33-1-30-60-70-00

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

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

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

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

Italy - Padova
Tel: 39-049-7625286

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

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

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

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820