



---

---

## USI Host User Guide

---

---

### Description

---

USI Host library user guide for PRIME and G3 protocols.

DRAFT

---

---

## Table of Contents

---

Description.....	1
1. Configuration Steps.....	3
1.1. Configuration Header: PrjCfg.h.....	3
1.2. User Defined Functions: userFnc.h.....	3
1.3. Initialization.....	3
1.4. Protocol Interfaces and Setting Callbacks.....	3
1.5. USI Host Process.....	7
2. Revision History .....	8
2.1. Rev A – 12/2017.....	8
The Microchip Web Site.....	9
Customer Change Notification Service.....	9
Customer Support.....	9
Microchip Devices Code Protection Feature.....	9
Legal Notice.....	10
Trademarks.....	10
Quality Management System Certified by DNV.....	11
Worldwide Sales and Service.....	12

## 1. Configuration Steps

### 1.1 Configuration Header: PrjCfg.h

This header file is used for the USI Host configuration. The user must include this file with this name. At USI Host repository there is a file named PrjCfg.h with the example below:

```
#define NUM_PORTS 1
#define PORT_0 CONF_PORT(UART_TYPE,3, 115200, 4096, 4096)
#define NUM_PROTOCOLS 5
#define USE_MNGP_PRIME_PORT 0
#define USE_PROTOCOL_SNIF_PRIME_PORT 0
#define USE_PROTOCOL_PRIME_API 0
#define USE_PROTOCOL_ADP_G3_PORT 0
#define USE_PROTOCOL_COORD_G3_PORT 0
```

It is possible to configure up to 4 ports. This is configured with *NUM\_PORTS* and *PORT\_0*, *PORT\_1*, *PORT\_2*, *PORT\_3*. *PORT\_x* is configured with the macro *CONF\_PORT*(type, channel, baudrate, txSize, rxSize), where:

- *Type*: Port type (*UART\_TYPE*, *USART\_TYPE* or *COM\_TYPE*)
- *Channel*: COM port number
- *Baudrate*: Port speed
- *txSize*: Buffer size for transmission
- *rxSize*: Buffer size for reception

At the moment, 3 PRIME protocols (MNGP\_PRIME, PRIME\_API and PRIME\_SNIFFER) and 2 G3 protocols (ADP and COORD) are available. Each protocol can be enabled by using the appropriate macro. The value of the definition sets the port used (0 to 3) for that protocol.

The port configuration can also be changed once the program is running, using the function *addUsi\_ConfigurePort* (uint8\_t logPort, uint8\_t commPort, uint32\_t speed).

### 1.2 User Defined Functions: userFnc.h

The user must implement the following functions:

```
int8_t addUsi_Open(uint8_t port_type, uint8_t port, uint32_t bauds);
uint16_t addUsi_TxMsg(uint8_t port, uint8_t *msg, uint16_t msglen);
int8_t addUsi_RxChar(uint8_t port, uint8_t *c);
```

This functions typically will interface to a serial port to read a character or transmit a message.

### 1.3 Initialization

The user has to call *addUsi\_Init()* at the beginning. If the port has to be configured (it is different than the port configured in configuration header) *addUsi\_ConfigurePort()* should be called before *addUsi\_Init()*.

### 1.4 Protocol Interfaces and Setting Callbacks

The user can set the callbacks which will be called when a confirm or indication is received from the device.

## 1.4.1 PRIME API

PRIME API functions are defined in *prime\_api\_host.h*. This file contains the interface with PRIME API.

There are request functions, e.g.:

```
prime_cl_null_mlme_get_request(uint16_tus_pib_attr)
```

When a confirm or indication message is received, the corresponding callback is called. The callbacks are initialized with the following functions:

```
prime_cl_null_set_callbacks(prime_cl_null_callbacks_t *px_prime_cbs);
prime_cl_432_set_callbacks(prime_cl_432_callbacks_t *px_prime_cbs);
```

Where the argument is a pointer to a struct with the callback functions:

```
typedef struct {
    prime_cl_null_establish_ind_cb_t prime_cl_null_establish_ind_cb;
    prime_cl_null_establish_cfm_cb_t prime_cl_null_establish_cfm_cb;
    prime_cl_null_release_ind_cb_t prime_cl_null_release_ind_cb;
    prime_cl_null_release_cfm_cb_t prime_cl_null_release_cfm_cb;
    prime_cl_null_join_ind_cb_t prime_cl_null_join_ind_cb;
    prime_cl_null_join_cfm_cb_t prime_cl_null_join_cfm_cb;
    prime_cl_null_leave_ind_cb_t prime_cl_null_leave_ind_cb;
    prime_cl_null_leave_cfm_cb_t prime_cl_null_leave_cfm_cb;
    prime_cl_null_data_ind_cb_t prime_cl_null_data_ind_cb;
    prime_cl_null_data_cfm_cb_t prime_cl_null_data_cfm_cb;
    prime_cl_null_plme_reset_cfm_cb_t prime_cl_null_plme_reset_cfm_cb;
    prime_cl_null_plme_sleep_cfm_cb_t prime_cl_null_plme_sleep_cfm_cb;
    prime_cl_null_plme_resume_cfm_cb_t prime_cl_null_plme_resume_cfm_cb;
    prime_cl_null_plme_testmode_cfm_cb_t
        prime_cl_null_plme_testmode_cfm_cb;
    prime_cl_null_plme_get_cfm_cb_t prime_cl_null_plme_get_cfm_cb;
    prime_cl_null_plme_set_cfm_cb_t prime_cl_null_plme_set_cfm_cb;
    prime_cl_null_mlme_register_ind_cb_t
        prime_cl_null_mlme_register_ind_cb;
    prime_cl_null_mlme_register_cfm_cb_t
        prime_cl_null_mlme_register_cfm_cb;
    prime_cl_null_mlme_unregister_ind_cb_t
        prime_cl_null_mlme_unregister_ind_cb;
    prime_cl_null_mlme_unregister_cfm_cb_t
        prime_cl_null_mlme_unregister_cfm_cb;
    prime_cl_null_mlme_promote_ind_cb_t prime_cl_null_mlme_promote_ind_cb;
    prime_cl_null_mlme_promote_cfm_cb_t prime_cl_null_mlme_promote_cfm_cb;
    prime_cl_null_mlme_demote_ind_cb_t prime_cl_null_mlme_demote_ind_cb;
    prime_cl_null_mlme_demote_cfm_cb_t prime_cl_null_mlme_demote_cfm_cb;
    prime_cl_null_mlme_reset_cfm_cb_t prime_cl_null_mlme_reset_cfm_cb;
    prime_cl_null_mlme_get_cfm_cb_t prime_cl_null_mlme_get_cfm_cb;
    prime_cl_null_mlme_list_get_cfm_cb_t
        prime_cl_null_mlme_list_get_cfm_cb;
    prime_cl_null_mlme_set_cfm_cb_t prime_cl_null_mlme_set_cfm_cb;
} prime_cl_null_callbacks_t;
```

```
typedef struct {
    prime_cl_432_establish_cfm_cb_t prime_cl_432_establish_cfm_cb;
    prime_cl_432_release_cfm_cb_t prime_cl_432_release_cfm_cb;
    prime_cl_432_dl_data_ind_cb_t prime_cl_432_dl_data_ind_cb;
    prime_cl_432_dl_data_cfm_cb_t prime_cl_432_dl_data_cfm_cb;
} prime_cl_432_callbacks_t;
```

## 1.4.2 Base Management

This protocol features are only present on Base nodes, such as: Firmware Update Protocol, Network Events, Prime Profile (for remote PIBs access), Zero Cross and Whitelist Management. Protocol interface and callbacks interfaces are defined in *prime\_api\_host.h*.

When a confirm or indication message is received, the corresponding callback is called. The callbacks are initialized with the following functions:

```
void bmng_set_callbacks(prime_bmng_callbacks_t *px_fup_cbs);
```

Where the argument is a pointer to a struct with the grouping all callback functions:

```
typedef struct {
    bmng_fup_ack_ind_cb_t          fup_ack_ind_cb;
    bmng_fup_error_ind_cb_t       fup_error_ind_cb;
    bmng_fup_version_ind_cb_t     fup_version_ind_cb;
    bmng_fup_status_ind_cb_t      fup_status_ind_cb;
    bmng_fup_kill_ind_cb_t        fup_kill_ind_cb;
    bmng_network_event_ind_cb_t    network_event_ind_cb;
    bmng_pprof_ack_ind_cb_t       pprof_ack_ind_cb;
    bmng_pprof_get_response_cb_t  pprof_get_response_cb;
    bmng_pprof_get_enhanced_response_cb_t pprof_get_enhanced_response_cb;
    bmng_pprof_zerocross_response_cb_t pprof_zerocross_response_cb;
    bmng_pprof_zc_diff_response_cb_t pprof_zc_diff_response_cb;
    bmng_whitelist_ack_cb_t       whitelist_ack_cb;
} prime_bmng_callbacks_t;
```

Each callback must have the appropriate parameters. The definition of the callback format is in *prime\_api\_defs\_host.h*. If a callback is set to NULL, the corresponding confirm or indication message won't be received by any callback function.

### 1.4.3 PRIME Management Protocol

Management Protocol (MNGP) is the standard protocol defined by PRIME standard. The interface is defined in *mngLayerHost.h*. In order to send a message, there three steps:

- Create a message: *mngLay\_NewMsg(uint8\_tcmd);*
- Add one or more queries (of the same kind):
  - Get PIB: *mngLay\_AddGetPibQuery(uint16\_t pib, uint8\_t index);*
  - Set PIB: *mngLay\_AddSetPib(uint16\_t pib, uint16\_t length, uint8\_t\* msg);*
  - Reset Statistics: *mngLay\_AddResetStats(uint16\_t pib, uint8\_t index);*
  - FW Upgrade Message: *mngLay\_AddFUMsg(uint16\_t length, uint8\_t\* msg);*
  - Bridge Message: *mngLay\_BridgeMsg(uint16\_t length, uint8\_t\* msg);*
- Send request message: *mngLay\_SendMsg();*

The command in *mngLay\_NewMsg* should be one of the following:

```
/// Protocol ID to serialize (USI)
#define MNGP_PRIME                0x00
#define MNGP_PRIME_GETQRY        0x00
#define MNGP_PRIME_GETRSP        0x01
#define MNGP_PRIME_SET           0x02
#define MNGP_PRIME_RESET         0x03
#define MNGP_PRIME_REBOOT        0x04
#define MNGP_PRIME_FU            0x05
```

The query should be coherent with the protocol ID selected in *mngLay\_NewMsg*. An example for sending a *MNGP\_PRIME\_GETQRY* message is shown below:

```
uint16_t pib_attrib;
mngLay_NewMsg(MNGP_PRIME_GETQRY);
mngLay_AddGetPibQuery(pib_attrib, 0);
mngLay_SendMsg();
```

There is only one callback, which is set by the following function:

```
void mngp_set_rsp_cb(void (*sap_handler)(uint8_t* ptrMsg, uint16_t len));
```

The only MNGP command which is sent by the device is MNGP\_PRIME\_GET\_RESP. This callback will retrieve a buffer with the whole MNGP message (USI header and CRC is removed). The user must know the protocol and process the message.

### 1.4.4 PRIME Sniffer

The interface is quite simple and is defined in *ifacePrimeSniffer.h*. Only two functions are available.

One function to set the callback where sniffer messages will be sent:

```
void prime_sniffer_set_cb(void (*sap_handler)(uint8_t* msg, uint16_t len));
```

One function to set the channel:

```
void prime_sniffer_set_channel(uint8_t uc_channel);
```

### 1.4.5 G3 ADP API

G3 ADP API functions are defined in *AdpApi.h*. This file contains the interface with G3 ADP API, which is the entry point to the G3-PLC stack. This file is a copy of the embedded: the G3 USI host interface is the same one the embedded stack provides.

There are request functions, e.g.:

```
void AdpGetRequest(uint32_t u32AttributeId, uint16_t u16AttributeIndex);
```

When a confirm or an indication message is received, the corresponding callback is called. The callbacks are initialized when the ADP layer is initialized:

```
void AdpInitialize(struct TAdpNotifications *pNotifications, enum TAdpBand band);
```

Where:

- The first argument is a pointer to a struct with the callback functions:

```
struct TAdpNotifications {
    AdpDataConfirm fnctAdpDataConfirm;
    AdpDataIndication fnctAdpDataIndication;
    AdpDiscoveryConfirm fnctAdpDiscoveryConfirm;
    AdpDiscoveryIndication fnctAdpDiscoveryIndication;
    AdpNetworkStartConfirm fnctAdpNetworkStartConfirm;
    AdpNetworkJoinConfirm fnctAdpNetworkJoinConfirm;
    AdpNetworkLeaveIndication fnctAdpNetworkLeaveIndication;
    AdpNetworkLeaveConfirm fnctAdpNetworkLeaveConfirm;
    AdpResetConfirm fnctAdpResetConfirm;
    AdpSetConfirm fnctAdpSetConfirm;
    AdpMacSetConfirm fnctAdpMacSetConfirm;
    AdpGetConfirm fnctAdpGetConfirm;
    AdpMacGetConfirm fnctAdpMacGetConfirm;
    AdpLbpConfirm fnctAdpLbpConfirm;
    AdpLbpIndication fnctAdpLbpIndication;
    AdpRouteDiscoveryConfirm fnctAdpRouteDiscoveryConfirm;
    AdpPathDiscoveryConfirm fnctAdpPathDiscoveryConfirm;
    AdpNetworkStatusIndication fnctAdpNetworkStatusIndication;
    AdpBufferIndication fnctAdpBufferIndication;
    AdpPREQIndication fnctAdpPREQIndication;
    AdpUpdNonVolatileDataIndication
        fnctAdpUpdNonVolatileDataIndication;
    AdpRouteNotFoundIndication fnctAdpRouteNotFoundIndication;
};
```

Each callback must have the appropriate parameters. The definition of the callback format is in the same file (*AdpApi.h*).

If a callback is set to NULL, the corresponding confirm or indication message won't be received by any callback function.

- The second argument is the band, whose type (*TAdpBand*) is defined in *AdpApiTypes.h* (*ADP\_BAND\_CENELEC\_A*, *ADP\_BAND\_CENELEC\_B*, *ADP\_BAND\_FCC*, *ADP\_BAND\_ARIB*)

### 1.4.6 G3 COORD API

G3 COORD API functions are defined in *bs\_api.h*. This file contains the interface with G3 bootstrap API, which is the entry point to the G3-PLC bootstrap process used by the coordinator. This file is a copy of the embedded: the G3 USI host interface is the same one the embedded stack provides.

There are request functions, e.g.:

```
void bs_lbp_get_param(uint32_t ul_attribute_id, uint16_t us_attribute_idx,
struct t_bs_lbp_get_param_confirm *p_get_confirm);
```

When a confirm or an indication message is received, the corresponding callback is called.

Confirm callbacks are set in the corresponding request. For example, in the former *bs\_lbp\_get\_param* request function, the last parameter is a pointer confirm callback function.

Indication callbacks are set using functions defined in the same file (*bs\_api.h*):

```
void bs_lbp_leave_ind_set_cb(pf_app_leave_ind_cb_t pf_handler);
void bs_lbp_join_ind_set_cb(pf_app_join_ind_cb_t pf_handler);
```

Each callback must have the appropriate parameters. The definition of the callback format is in the same file (*bs\_api.h*).

If a callback is set to NULL, the corresponding confirm or indication message won't be received by any callback function.

## 1.5 USI Host Process

The function *addUsi\_Process()* must be called periodically. This function will call user function to retrieve data from the serial port and once a full HDLC encoded message is received, it will process it, calling the corresponding callbacks if needed

## 2. Revision History

### 2.1 Rev A – 12/2017

Document	Initial release PRIME&G3.
----------	---------------------------

DRAFT



---

## The Microchip Web Site

---

Microchip provides online support via our web site at <http://www.microchip.com/>. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## Customer Change Notification Service

---

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at <http://www.microchip.com/>. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

## Customer Support

---

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: <http://www.microchip.com/support>

## Microchip Devices Code Protection Feature

---

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.

## Legal Notice

---

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Trademarks

---

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA 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.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

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

---

© 2017, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN:

## **Quality Management System Certified by DNV**

---

### **ISO/TS 16949**

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<sup>®</sup> MCUs and dsPIC<sup>®</sup> DSCs, KEELOQ<sup>®</sup> 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.

DRAFT

## Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
<b>Corporate Office</b> 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: <a href="http://www.microchip.com/support">http://www.microchip.com/support</a> Web Address: <a href="http://www.microchip.com">www.microchip.com</a>	<b>Australia - Sydney</b> Tel: 61-2-9868-6733 <b>China - Beijing</b> Tel: 86-10-8569-7000 <b>China - Chengdu</b> Tel: 86-28-8665-5511 <b>China - Chongqing</b> Tel: 86-23-8980-9588 <b>China - Dongguan</b> Tel: 86-769-8702-9880 <b>China - Guangzhou</b> Tel: 86-20-8755-8029 <b>China - Hangzhou</b> Tel: 86-571-8792-8115 <b>China - Hong Kong SAR</b> Tel: 852-2943-5100 <b>China - Nanjing</b> Tel: 86-25-8473-2460 <b>China - Qingdao</b> Tel: 86-532-8502-7355 <b>China - Shanghai</b> Tel: 86-21-3326-8000 <b>China - Shenyang</b> Tel: 86-24-2334-2829 <b>China - Shenzhen</b> Tel: 86-755-8864-2200 <b>China - Suzhou</b> Tel: 86-186-6233-1526 <b>China - Wuhan</b> Tel: 86-27-5980-5300 <b>China - Xian</b> Tel: 86-29-8833-7252 <b>China - Xiamen</b> Tel: 86-592-2388138 <b>China - Zhuhai</b> Tel: 86-756-3210040	<b>India - Bangalore</b> Tel: 91-80-3090-4444 <b>India - New Delhi</b> Tel: 91-11-4160-8631 <b>India - Pune</b> Tel: 91-20-4121-0141 <b>Japan - Osaka</b> Tel: 81-6-6152-7160 <b>Japan - Tokyo</b> Tel: 81-3-6880-3770 <b>Korea - Daegu</b> Tel: 82-53-744-4301 <b>Korea - Seoul</b> Tel: 82-2-554-7200 <b>Malaysia - Kuala Lumpur</b> Tel: 60-3-7651-7906 <b>Malaysia - Penang</b> Tel: 60-4-227-8870 <b>Philippines - Manila</b> Tel: 63-2-634-9065 <b>Singapore</b> Tel: 65-6334-8870 <b>Taiwan - Hsin Chu</b> Tel: 886-3-577-8366 <b>Taiwan - Kaohsiung</b> Tel: 886-7-213-7830 <b>Taiwan - Taipei</b> Tel: 886-2-2508-8600 <b>Thailand - Bangkok</b> Tel: 66-2-694-1351 <b>Vietnam - Ho Chi Minh</b> Tel: 84-28-5448-2100	<b>Austria - Wels</b> Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 <b>Denmark - Copenhagen</b> Tel: 45-4450-2828 Fax: 45-4485-2829 <b>Finland - Espoo</b> Tel: 358-9-4520-820 <b>France - Paris</b> Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 <b>Germany - Garching</b> Tel: 49-8931-9700 <b>Germany - Haan</b> Tel: 49-2129-3766400 <b>Germany - Heilbronn</b> Tel: 49-7131-67-3636 <b>Germany - Karlsruhe</b> Tel: 49-721-625370 <b>Germany - Munich</b> Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 <b>Germany - Rosenheim</b> Tel: 49-8031-354-560 <b>Israel - Ra'anana</b> Tel: 972-9-744-7705 <b>Italy - Milan</b> Tel: 39-0331-742611 Fax: 39-0331-466781 <b>Italy - Padova</b> Tel: 39-049-7625286 <b>Netherlands - Drunen</b> Tel: 31-416-690399 Fax: 31-416-690340 <b>Norway - Trondheim</b> Tel: 47-7289-7561 <b>Poland - Warsaw</b> Tel: 48-22-3325737 <b>Romania - Bucharest</b> Tel: 40-21-407-87-50 <b>Spain - Madrid</b> Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 <b>Sweden - Gothenberg</b> Tel: 46-31-704-60-40 <b>Sweden - Stockholm</b> Tel: 46-8-5090-4654 <b>UK - Wokingham</b> Tel: 44-118-921-5800 Fax: 44-118-921-5820