The Analog Mixed Signal Company

Application Note

Rev. 1.30 / July 2012

ZWIR451x

Enabling Firmware Over-the-Air Updates

Contents

List of Figures

List of Tables

1 Introduction

In wireless sensor and control systems it is typically hard to impossible to change the software on a device once it has been installed in its application. If software bugs are arising in after the installation often the only way of fixing the problem is a complete replacement of the defect nodes or time and cost expensive disassembly, reprogramming and reassembly of the device.

In ZWIR451x based sensor and control networks this can be circumvented with a firmware over-the-air update mechanism. ZMDI provides this mechanism by means of a easily configurable library which can be linked into the application firmware.

The firmware update is robust against brown-out errors, capable of handling different devices in the same network and can be secured using the IPSec and IKEv2 protocols.

This application note explains some technical background of the firmware update and shows how a device must be configured to enable the over-the-air update functionality.

2 Over-the-Air Update Overview

2.1. Update Strategy

One main challenge of the OTAU is to replace the old user code section with a new one. First the new firmware has to be received and saved to a free area in the flash memory. It is not possible to replace the firmware on the fly because the firmware will be executed while receiving the new firmware image. Furthermore the new firmware image must be stored in the persistent flash memory to save the very limited data ram and to allow deep sleep modes between receiving firmware fragments.

During exchanging the old with the new firmware every external interruption causing a system reset can be fatal for the update process. In case of an external reset or brownout the integrated update function tries to continue the exchange process or recovers the old firmware. In any case the system remains in an executable state after an update.

The system startup code and the integrated update functions are located in a non updateable section in the flash memory to prevent unrecoverable update failures.

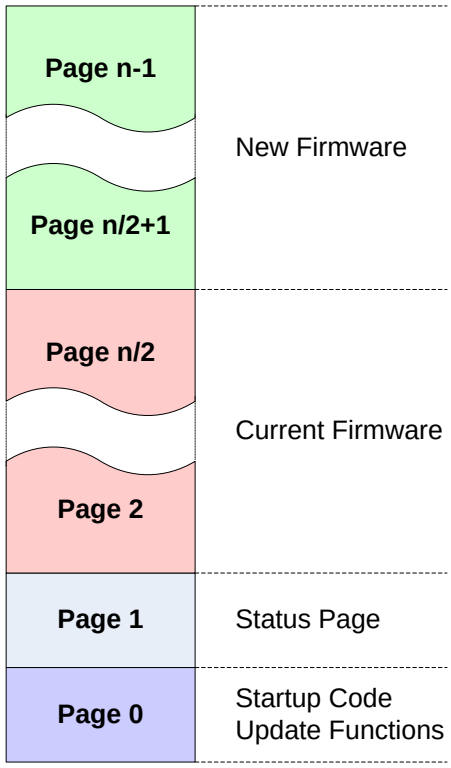
The integrity of every firmware segment is ensured by a 32 Bit CRC checksum.

2.2. Flash Memory Layout

With active OTAU the useable free flash memory will be halved because the new Firmware needs space in the flash memory. Additionally one flash page is required for the startup code and the update functions and another page is needed for storing update relevant data like status and CRC checksum.

The Flash segmentation is shown in Figure 2.1.

Figure 2.1 Flash Memory Segmentation with Active OTAU

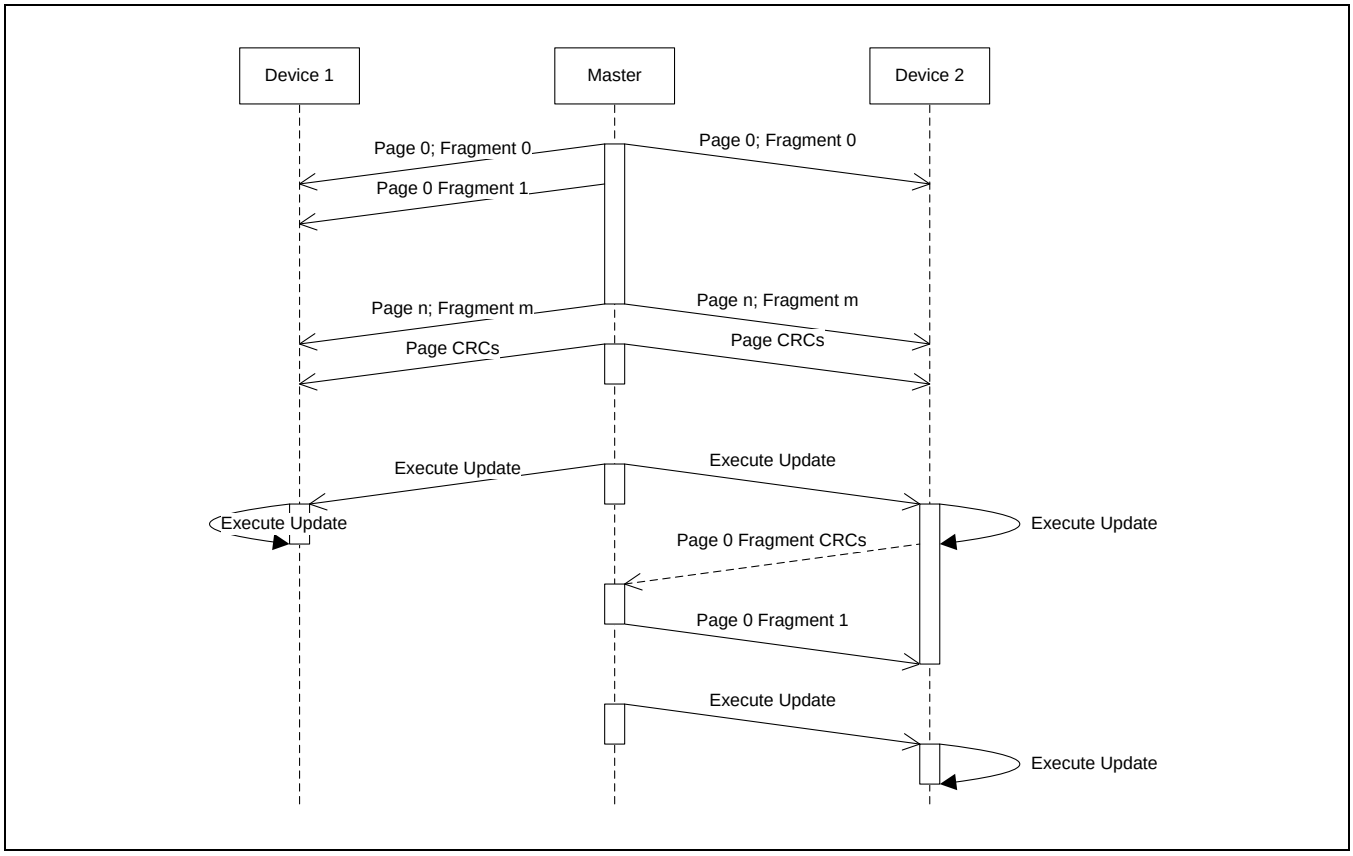


2.3. Firmware Distribution

The distribution of a new firmware takes place over the air, thus over UDP-IP. Therefore a special Update master sends the new firmware fragmented in broadcast, multicast or unicast packets to one or several devices. To ensure that every packet was received special packets, containing the CRC page checksums, are transmitted after the firmware. To execute the update the master sends execute update packets. Receiving this packet all addressed nodes check their received firmware image and start the update process. If one page containing an error, i.e. the calculated CRC over a page doesn't equal the transmitted CRC, the device calculate the CRC of every fragment of each page and send these checksums to the master. With the help of those checksums the master retransmits the missing fragments.

The communication between update master and devices is visualized in Figure 2.2.

Figure 2.2 Communication Sequence Between Update Master and Devices



2.4. Update Packet Format

There are four different packet types used for the communication between update server and devices.

Table 2.1 ZWIR_OTAU_Data_t

Byte	0	1	2	3
0	Type = 1		Length	
4	Vendor ID			
8	Product ID		Firmware Version	
C	Packet CRC32			
10	Page		Fragment	
14	Fragment Data 0	Fragment Data 1	Fragment Data 2	…

Table 2.2 ZWIR_OTAU_CRC_t

Byte	0	1	2	3
0	Type = 2		Length	
4	Vendor ID			
8	Product ID		Firmware Version	
C	Packet CRC32			
10	Start CRC		Reserved	
14	CRC 0	CRC 1	CRC 2	…

Table 2.3 ZWIR_OTAU_ExecuteUpdate_t

Byte	0	1	2	3
0	Type = 3		Length	
4	Vendor ID			
8	Product ID		Firmware Version	
C	Packet CRC32			
10	Number of Pages		Execute in [Seconds]	

Table 2.4 ZWIR_OTAU_ErrorCRC_t

Byte	0	1	2	3
0	Type = 129		Length	
4	Vendor ID			
8	Product ID		Firmware Version	
C	Packet CRC32			
10	Page		Reserved	
14	CRC Fragment 0	CRC Fragment 1	CRC Fragment 2	…

2.5. Version Management

Each firmware which should be able to be updated over-the-air must appropriately implement a set of constants specifying the vendor, the product and the firmware version. This is accomplished by setting the constants **ZWIR_vendorID**, **ZWIR_productID**, **ZWIR_firmwareMajorVersion** and **ZWIR_firmwareMinorVersion** in the firmware. Additionally, **ZWIR_firmwareVersionExtension** may be set. However, this is no requirement for the over-the-air update functionality.

The **ZWIR_vendorID** and **ZWIR_productID** constants are used by the over-the-air update daemon to uniquely identify the product. Both values must mach the values encoded in the update packet. **ZWIR_vendorID** is a 32 bit constant carrying a unique ID with a checksum. This 32 bit ID must be requested from ZMDI. One vendor ID is assigned to each company. For testing purposes the vendor ID ***0x0000e966*** is reserved. This ID MUST NOT be used in production code! The over-the-air update daemon only works with vendor IDs with valid checksum. If a wrong vendor ID is configured, the over-the-air update daemon will report a **ZWIR_eInvalidVID** to the **ZWIR_Errror** function and refuse to work. As a result, no updates can be received.

The procduct ID can be chosen freely for each product. It has to be made sure that each unique firmware gets an unique product ID. The responsibility for that is at the application developer. If different product use the same product ID the over-the-air update would update both products with the same firmware.

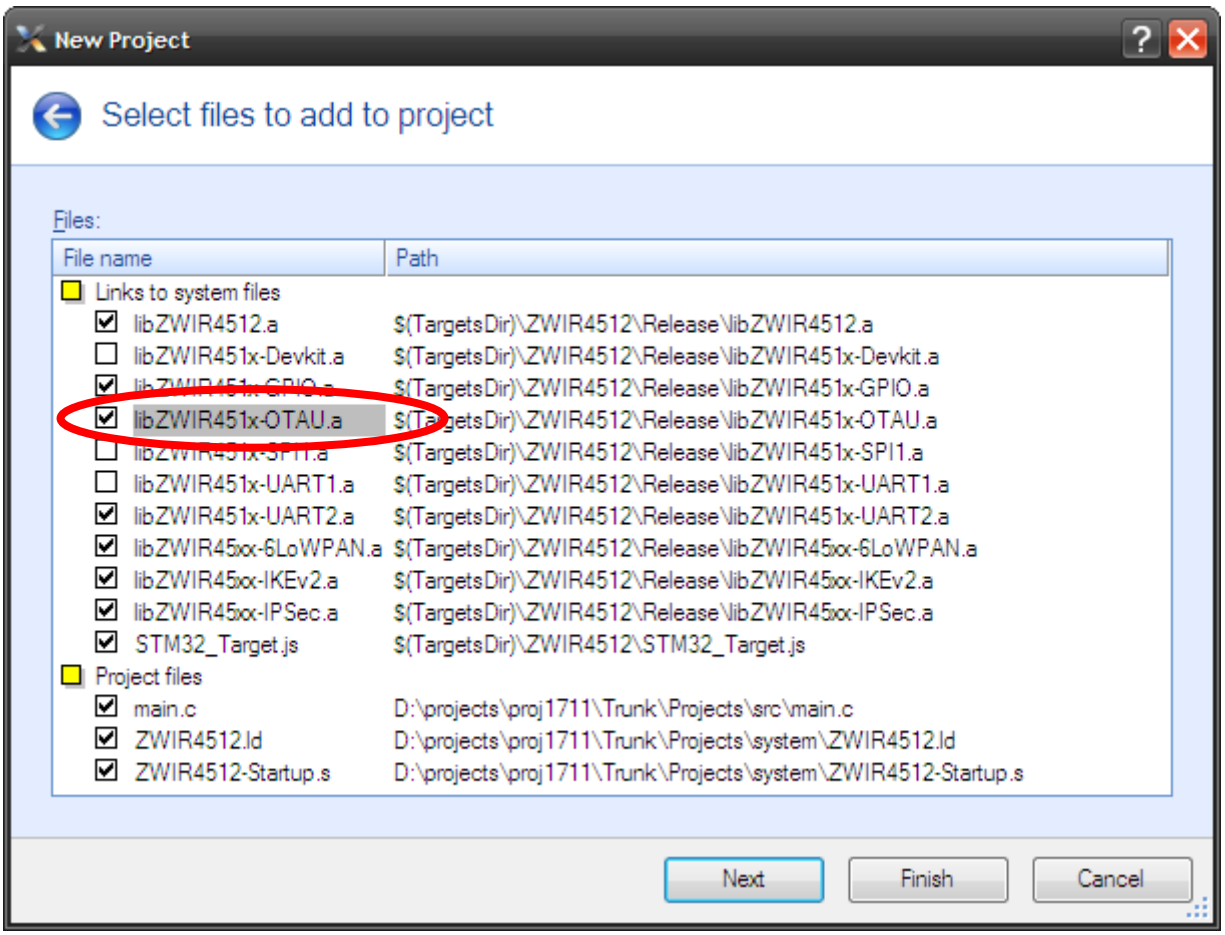
The firmware version information encoded in **ZWIR_firmwareMajorVersion** and **ZWIR_firmwareMinorVersion** is used to check if incoming update packets are newer than the exisiting firmware. Only newer firmware versions are accepted by the OTAU daemon.

3 Using Over-the-Air Updates

3.1. Enabling Over-the-Air Updateability of Firmware

To enable the over the air update daemon on a ZWIR451x device the library *libZWIR451x-OTAU.a* must be added during the creation of a new project. Alternatively, the library may be added to an existing project. During the device startup the function **`ZWIR_OTAU_Register`** must be called once. Its prototype is found in the header file *ZWIR451x-OTAU.h.*

Figure 3.1 Library Selection Dialog during Project Creation in Rowley CrossStudio



```
bool
  ZWIR_OTAU_Register ( unsigned short  localPort )
```

This function registers the over-the-air update daemon at the operating system and configures the UDP port which is used for reception of updates. It should be called once at system startup, typically from **ZWIR_AppInitNetwork**. Calling it from **ZWIR_AppInitHardware** has no effect. During the registration the daemon check the validity of the **ZWIR_vendorID** constant. If the verification fails, **ZWIR_Error** is called with the error code **ZWIR_eInvalidVID**.

The example below shows a firmware program with product ID 0x1000, version number 1.2, using the demonstration vendor ID 0x96ee. The example program may be updated over-the-air using port 1357:

```
01  #include "ZWIR45xx-6LoWPAN.h"
02  #include "ZWIR45xx-OTAU.h"
03
04  uint32_t const  ZWIR_vendorID = 0x000096ee;
05  uint16_t const  ZWIR_productID = 0x1000;
06  uint8_t  const  ZWIR_firmwareMajorVersion = 1;
07  uint8_t  const  ZWIR_minorFirmwareVersion = 2;
08
09  // Perform network intialization
10  void ZWIR_AppInitNetwork ( void ) {
11    //register OTAU daemon at local port 1357
12    ZWIR_OTAU_Register ( 1357 );
13  }
```

3.2. Securing the Update using IPSec and IKEv2

ZMDI strongly recommends securing the over-the-air update connection. Otherwise any malicious object may be able to send a firmware update to unsecured devices and therefore might be able to destroy the device or use it on its own without giving you the chance of getting control back. IPSec provides sufficient protection against such attacks and it is easily configured in the firmware.

The only thing that has to be done is defining the appropriate security policy for all UDP communication on the port configured in **ZWIR_OTAU_Register** with the device to be secured. Please refer to the application note “Using IPSec and IKEv2 in ZWIR451x Networks” for detailed information and configuration examples.

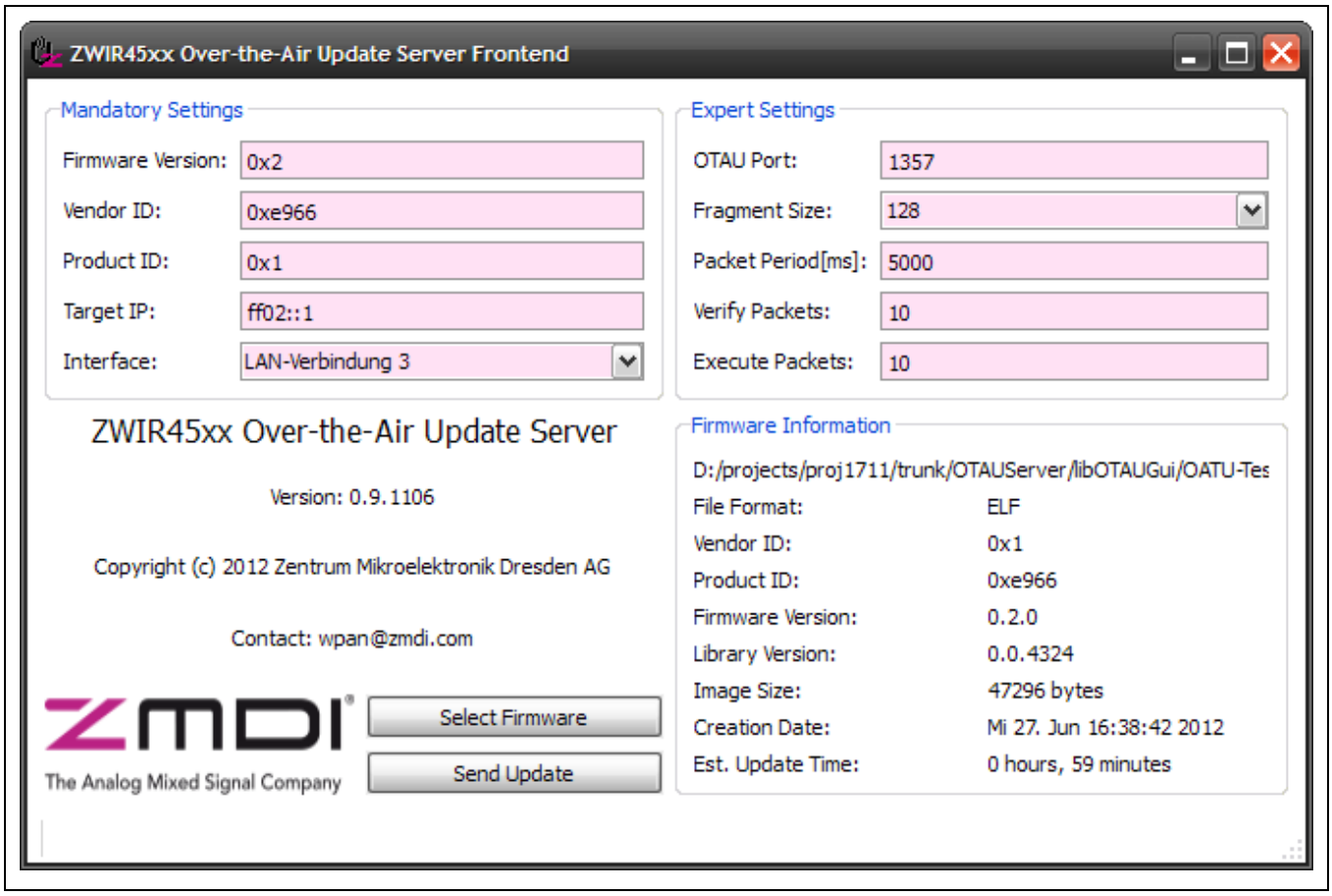
3.3. Distributing Firmware Updates Using the OTAU-Server

ZMDI provides a software tool called OTAU-Server, running on Windows and Linux PCs, which is used to update the firmware on ZWIR45xx based devices. This tool distributes a binary firmware image to one or multiple destination nodes. In order to generate a binary firmware image with Rowley CrossStudio, open the **Properties** dialog of your firmware project and set **Additional Output Format** (under **Linker Options**) to **bin**.

Before starting the update process, basic settings need to be adjusted in the OTAU-sever GUI.

First select the binary firmware file and set the firmware version, Vendor ID, Product ID, the over-the-air update UDP port as well as the IP address of the destination devices. It is recommended to use multicast addresses to update all devices in a subnet concurrently.

Figure 3.2 Screenshot of ZMDI's OTAU-Server for Update Distribution



The dialog settings have the following meaning:

Firmware Version	Specifies the firmware version to be distributed. This version number must be higher than the version installed on the device. Otherwise, the device will not accept update packets from the server. The version number doesn't necessarily be aligned with the version information included in the firmware binary (for instance to switch back to an older firmware version). However, it is recommended to be aligned. The **`ZWIR_firmwareVersionExtension`** component of the firmware version information is not considered by the OTAU daemon. Only the major and minor version are of interest.
Vendor ID	This field must be set to your unique Vendor ID. The default value of 0x0000e966 is only for experimental use and must not be used in productive designs. A Vendor ID can be obtained from ZMDI.
Product ID	This field must be set to the product ID of the product to be updated. Each firmware version must have a unique Product ID. Otherwise different products would be updated with the same firmware version.
Target IP	This field controls to which devices the update is sent. The IP must be a valid IPv6 address. The target address may be an unicast or multicast address.
Interface	For link-local addresses, this field determines the interface to be used for communication.
OTAU Port	This field must be set to the UDP port number configured in the firmware using **`ZWIR_OTAU_Register`**.
Fragment Size	This box allows controlling the size of firmware fragments to be transmitted. The optimum value depends on the network size. The larger the network the smaller the fragment size should be chosen. Furthermore, if the target address is a multicast address, a relatively small fragment size should be chosen.
Packet Period	This value controls the time interval the server application sends packet at. This optimum value depends on the communication frequency of the normal application which is still running on the devices. Furthermore, the value must be chosen large enough to not violate the duty-cycle requirements which may be in place in the operation area. The period configured in this field is directly used as interval for the transmission of update fragments. For the transmission of Verify and Execute Packets the 10 and 20-fold value is used, respectively.
Verify Packets	These packets instruct the receiver to verify if it has correctly received the update. If missing fragments are identified during the verification the server is informed about these fragments. The OTAU Server will not switch to the execute phase before it has sent out the number entered in this field, without getting a response.
Execute Packets	The update is activated by so-called "Execute Packets". This field controls the number of packets sent to activate the update. The Over-the-Air Update firmware tries to enable the update on all devices in the network simultaneously.

4 Related Documents

Document	File Name
ZWIR451x Programming Guide	ZWIR451x_ProgGuide_revX.xy.pdf

Visit ZMDI's website www.zmdi.com or contact your nearest sales office for the latest version of these documents.

5 Glossary

Term	Description
CRC	Cyclic Redundancy Check
OTAU	Over the air update
PID	Product ID
UDP	User datagram protocol
VID	Vendor ID

6 Document Revision History

Revision	Date	Description
1.00	March 15, 2011	Initial version
1.10	November 01, 2011	Update OTAU-Server
1.20	January 23, 2012	Improved description of version handling More detailed explanation of OTAU-Server parameters Update of imagery Minor edits Update of contact information
1.30	July 18, 2012	Update of OTAU Server Minor edits

Sales and Further Information www.zmdi.com wpan@zmdi.com

Zentrum Mikroelektronik Dresden AG Grenzstrasse 28 01109 Dresden Germany Phone +49.351.8822.7476 Fax +49.351.8822.87476	**ZMD America, Inc.** 1525 McCarthy Blvd., #212 Milpitas, CA 95035-7453 USA Phone +855-ASK-ZMDI (+855.275.9634)	**Zentrum Mikroelektronik Dresden AG, Japan Office** 2nd Floor, Shinbashi Tokyu Bldg. 4-21-3, Shinbashi, Minato-ku Tokyo, 105-0004 Japan Phone +81.3.6895.7410 Fax +81.3.6895.7301	**ZMD FAR EAST, Ltd.** 3F, No. 51, Sec. 2, Keelung Road 11052 Taipei Taiwan Phone +886.2.2377.8189 Fax +886.2.2377.8199	**Zentrum Mikroelektronik Dresden AG, Korean Office** POSCO Centre Building West Tower, 11th Floor 892 Daechi, 4-Dong, Kangnam-Gu Seoul, 135-777 Korea Phone +82.2.559.0660 Fax +82.2.559.0700