Introduction

This application note describes the ATWINC15x0 host driver Access Point (AP) mode application in detail, and demonstrates basic Wi-Fi connection. The device acts as an access point running in Ethernet mode.

Note: **WINC1500 Native mode** provides an integrated solution, is simplest, and is our recommended mode of operation. Fully supported in ASF and Harmony.

- **Main Features:**
  - Wi-Fi and communication protocols (IP and TLS) are all fully integrated on ATWINC15x0 (FW loaded from WINC flash)
  - Host MCU runs user application and WINC driver (limited resources needed)
  - Royalties, third party code or support is not required
  - Native mode can scale to any MCU core and size from 8-bit to 32-bit

- **Main Use Cases:**
  - Wi-Fi add-on to any MCU
  - Cloud connected devices that support only Wi-Fi connectivity (no ETH)

**WINC1500 Ethernet mode** is used when Ethernet (ETH) is used in parallel to Wi-Fi (i.e., ETH on PIC32MZ) or a specific IP stack is required (for example: IPv6, customer proprietary).

- **Main Features:**
  - Wi-Fi driver and stack runs on the ATWINC15x0 (FW loaded from WINC flash)
  - Host MCU runs TCP/IPv and TLS stack (Bypass ATWINC15x0 communication stacks)
  - Royalties or support contract is required for IP/TLS stack
  - Ethernet mode focuses mainly on 32-bit MCUs

- **Main Use Cases:**
  - Host MCUs with ETH and Wi-Fi interfaces (i.e., PIC32MZ, SAME51, etc)
  - IPv6 support is required
  - Specific TLS stack is required (i.e., TLS1.3)

Features

- ATWINC15x0 host MCU driver architecture
- ATWINC15x0 internal architecture
- Application description with code snippets
- Events handled in the Wi-Fi callback function with appropriate structure for each event
• Steps to execute the demo application using an Xplained Pro (XPro) board and the ATWINC15x0 XPro board
# Table of Contents

Introduction......................................................................................................................1

Features.......................................................................................................................... 1

1. Application Description.............................................................................................. 4

2. Host Driver Architecture............................................................................................ 5

3. ATWINC15x0 System Architecture............................................................................6

4. Wi-Fi Host Driver Initialization................................................................................... 7

5. Wi-Fi Host Driver Event and Callback....................................................................... 9

6. Running the Application ..........................................................................................10

7. Changes from Native Mode to Ethernet Mode ....................................................... 13

8. TCP/IP Networking Stack - lwIP............................................................................. 15

9. References.............................................................................................................. 16

10. Document Revision History..................................................................................... 17

The Microchip Web Site................................................................................................ 18

Customer Change Notification Service........................................................................18

Customer Support......................................................................................................... 18

Microchip Devices Code Protection Feature................................................................. 18

Legal Notice...................................................................................................................19

Trademarks................................................................................................................... 19

Quality Management System Certified by DNV.............................................................20

Worldwide Sales and Service........................................................................................21
1. **Application Description**

This application demonstrates the execution of ATWINC15x0 as a Wi-Fi access point using the SAM G55 XPro board as a host MCU. The device is running in Ethernet (also known as Bypass) mode.

**Note:** The Ethernet mode and the Bypass mode can be used interchangeably in the document.
2. **Host Driver Architecture**

The ATWINC15x0 host driver software is a C library, which provides the host MCU application with the necessary APIs to perform WLAN and socket operations. It also has a third party TCP/IP stack if the ATWINC15x0 is running in Ethernet mode and not using its own TCP/IP stack. The following figure shows the architecture of the ATWINC15x0 host driver software, which runs on the host MCU. For more details on the components of the host driver, refer to *ATWINC1500 Wi-Fi Network Controller Software Design Guide* (DS00002389).

Figure 2-1. Host Architecture
3. **ATWINC15x0 System Architecture**

The following figures show the ATWINC15x0 system architecture. In addition to its built-in Wi-Fi IEEE-802.11 physical layer and RF front end, the ASIC contains an embedded APS3S-Cortus 32-bit CPU to run the ATWINC15x0 firmware. The firmware comprises the Wi-Fi IEEE-802.11 MAC layer and embedded protocol stacks. For more details on the components of the system, refer to *ATWINC1500 Wi-Fi Network Controller Software Design Guide* (DS00002389).

Figure 3-1. ATWINC15x0 Native Mode and Ethernet Mode
4. **Wi-Fi Host Driver Initialization**

This chapter describes operation of the Wi-Fi functions via execution of a set of synchronous initialization sequences.

1. Initialize the ATWINC15x0 XPro board, which includes clock, hardware, events, and external hardware (if any).
   
   
   ```c
   /* Initialize the board. */
   sysclk_init();
   board_init();
   ```

2. Configure the UART interface console to debug the output. The debug log level value can be set using the `M2M_LOG_LEVEL` macro in `nm_debug.h` file.
   
   ```c
   /* Initialize the UART console. */
   configure_console();
   puts(STRING_HEADER)
   ```

3. Initialize the network stack using the following function.
   
   ```c
   /* Initialize the network stack. */
   net_init();
   ```

4. To initialize the BSP (Board Support Package) driver, the ATWINC15x0 initialization sequence must be followed. Follow the Chip Enable and Reset sequence of the ATWINC15x0.
   
   ```c
   /* Initialize the BSP. */
   nm_bsp_init();
   ```

5. Initialize the Wi-Fi host driver starting with the API `m2m_wifi_init()` and `tstrWifiInitParam` structure, completed with appropriate information. The Wi-Fi initialization sequence configures the SPI communication interface and external interrupt, with respect to the host MCU and the ATWINC15x0 pin connections. Apart from this, the Wi-Fi host application layer callback function `wifi_cb()` is registered with the initialization sequence.

6. After the successful initialization of the communication interface between the ATWINC15x0 and the host MCU, it is able to read the ATWINC15x0 chip-id indicating that the ATWINC15x0 is ready to accept the Wi-Fi initialization and configuration commands of the WLAN module. In this process, the ATWINC15x0 WLAN module is reset by the host MCU and waits for the module firmware to start confirmation.

7. The driver initialization needs to ensure that the user configures the Ethernet mode in the ATWINC15x0. Since the extra parameters are initialized for the `m2m_wifi_init()` function, the ATWINC15x0 bypasses its own networking stack.

   ```c
   void winc_fill_callback_info(tstrEthInitParam *info)
   {
   info->pfAppEthCb = winc_netif_rx_callback;
   info->au8EthRcvBuf = rx_buf;
   info->u8EthernetEnable = 1; //For bypassing the TCP/IP Stack of WINC
   }
   ```

This code ensures that the bit `rHAVE_ETHERNET_MODE_BIT` (0x80) is set in the register `rNMI_GP_REG_1` (0x14A0), when the driver is initialized via `m2m_wifi_init()`.

8. The `pfAppEthCbin` and the `winc_fill_callback_info()` are initialized to the Ethernet callback function which is a handle to the networking stack for passing the Ethernet packet received from the ATWINC15x0.

   ```c
   static void winc_netif_rx_callback (uint8 u8MsgType, void * pvMsg, void * pvCtrlBuf)
   {
   ```
switch(u8MsgType)
{
    case M2M_WIFI_RESP_ETHERNET_RX_PACKET:
    //.....insert code as per app requirements break;
    default:
        break;
}

9. The purpose of the Wi-Fi application callback function is to indicate the events such as connect and disconnect status, and to obtain the IP address from the Dynamic Host Configuration Protocol (DHCP) server with respect to the DHCP request from the networking stack running on the host.

    /* Initialize Wi-Fi parameters structure. */
    /* Initialize Wi-Fi driver with data and status callbacks. */
    param.pfAppWifiCb = wifi_cb;
    winc_fill_callback_info(param.strEthInitParam);
    ret = m2m_wifi_init(&param);
    if (M2M_SUCCESS != ret)
    {
        printf("main: m2m_wifi_init call error!(%d)\r\n", ret);
        while (1) {}
    }

10. For transmission, the following API can be used for sending the TCP/IP packets provided by the TCP/IP stack.

    m2m_wifi_send_ethernet_pkt(tx_buf, num_bytes_transmitted);
5. Wi-Fi Host Driver Event and Callback

All the Wi-Fi host driver events are handled in the `m2m_wifi_handle_events()` by running in the infinite loop. This function internally uses the HIF (Host Hardware Interface) layer API to monitor the ATWINC15x0 external interrupt. This external interrupt is registered using the following host MCU external interrupt configuration.

```c
while (1) {
    /* Handle pending events from network controller. */
    while (m2m_wifi_handle_events(NULL) != M2M_SUCCESS)
        {} // infinite loop
}
```

When the ATWINC15x0 external interrupt occurs, the host interface ISR layer reads the ATWINC15x0 control register to identify the type of event which triggered the external interrupt by the ATWINC15x0 and reads the data related to the event from the ATWINC15x0 buffer. After reading the complete data for the event, the corresponding callback function is triggered. The host MCU Wi-Fi driver handles two types of event categories, including `M2M_REQ_GRP_WIFI` and `M2M_REQ_GRP_OTA`.

```c
m2m_wifi_cb() /*handles all the Wi-Fi configuration and connection events.*/
```
6. **Running the Application**

This example demonstrates the execution of ATWINC15x0 as a Wi-Fi Access Point using the SAM G55 XPro board as a host MCU.

The demo uses the following hardware:
- SAM G55 Xplained Pro
- ATWINC15x0 on EXT1 header
- IEEE 802.11 b/g/n supported Wi-Fi Station

Perform the following steps to run the application:

1. Open Atmel Studio 7 and select **File > New > Example Projects**.
2. In the New Example Project from ASF or Extensions window:
   1. Enter “Bypass” keyword in the search area.
   2. Select the WINC1500_AP_MODE_BYPASS_EXAMPLE project for SAM G55 and open the project.
   3. An AP mode Bypass Application example is shown in the following image.

![Figure 6-1. AP Mode Bypass Example](image)

3. AP mode demo Wi-Fi credentials such as SSID and Security type are defined in the `main.h` file. The demo application comes up as Access Point, and waits for connection from a Wi-Fi Station. A third party Wi-Fi Station scans for the AP and establishes a Wi-Fi connection with right credentials. This demo is explained based on the authentication type set by the user on security method (for example: OPEN). To set an authentication type, the `MAIN_WLAN_AUTH` macro value needs to be configured as mentioned in the code snippet below. Various authentications such as,
M2M_WIFI_SEC_WPA_PSK, etc are supported by the demo application. The AP’s SSID can be filled for MAIN_WLAN_SSID macro.

3.1. Configure the AP credentials in the Wi-Fi Station.
3.2. This application demo is based on the M2M_WIFI_SEC_OPEN method. Do configure the security settings based on the AP settings used.

```c
/** Wi-Fi Settings */
#define MAIN_WLAN_SSID                "WINC1500_BYPASS" /* < AP SSID */
#define MAIN_WLAN_AUTH                M2M_WIFI_SEC_OPEN /* < Security manner */
```

4. Open the serial port terminal application with the following COM port configuration
   - Baudrate – 115200
   - Data – 8 bit
   - Parity – None
   - Stop – 1 bit
   - Flow control – None

Figure 6-2. SAM G55 Xplained Pro Board Setup

5. Compile and download the image into the SAM G55 XPro board.
6. Run the application to display success or error messages in the serial port terminal.
Figure 6-3. Application Output on Serial Terminal

--- WINC1500 AP in Bypass Mode example ---
--- SAMCS5_XPLAINED_PRO ---
--- Compiled: Feb 8 2018 17:49:42 ---

(APP)<INFO>Chip ID 1500c0
(APP)<INFO>DriverVerInfo: 0x13301354
(APP)<INFO>Firmware ver: 19.5.4 Sunrev 15567
(APP)<INFO>Firmware Build Oct 4 2017 Time 14:59:09
(APP)<INFO>Firmware Min driver ver: 19.3.0
(APP)<INFO>Driver version: 19.5.4
(APP)<INFO>Driver built at Feb 8 2018 17:49:01
WiFi IP is 192.168.5.1
AP node started. You can connect to WINC1500_BYPASS.
Station Connected
7. Changes from Native Mode to Ethernet Mode

The user can configure the ATWINC15x0 behaviour from the host application. The following information (basic snippet of the code) is provided for better understanding of using or configuring the ATWINC15x0 module in Ethernet mode:

1. Compile the host driver with `ETH_MODE`. The user can define the compile time switch in `nm_common.h`.

2. Perform the Network stack initialization before calling the `m2m_wifi_init()`. Set the additional parameters for `m2m_wifi_init()`:

   ```c
   tstrWifiInitParam param;
   param.pfAppWifiCb = wifi_cb;
   param.strEthInitParam.au8ethRcvBuf = eth_buffer;
   param.strEthInitParam.u16ethRcvBufSize = sizeof(eth_buffer);
   param.strEthInitParam.u8EthernetEnable = 1;
   param.strEthInitParam.pfAppEthCb = eth_cb;
   ret = m2m_wifi_init(&param);
   ```

   The above code ensures that the bit `rHAVE_ETHERNET_MODE_BIT (0x80)` is set in the register `rNMI_GP_REG_1 (0x14A0)`. Also, the `eth_buffer` is the buffer given by the host from which the packet is received ATWINC15x0.

3. The following Ethernet callback function can be implemented as per the user requirements:

   ```c
   static void eth_cb(uint8 u8MsgType, void * pvMsg, void * pvCtrlBuf) {
      switch(u8MsgType) {
      case M2M_WIFI_RESP_ETHERNET_RX_PACKET:
        //.....insert code as per app requirements
        break;
      default:
        break;
      }
   }
   ```

Table 7-1. Ethernet Callback

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>u8MsgType</td>
<td>Uint8</td>
<td>Message Type - <code>M2M_WIFI_RESP_ETHERNET_RX_PACKET</code>. This is an event occurring when the Ethernet packet is received from ATWINC15x0.</td>
</tr>
<tr>
<td>pvMsg</td>
<td>Void *</td>
<td>Pointer to the Rx buffer. In case of an unfragmented packet, the packet received is in the 802.3 Ethernet format.</td>
</tr>
<tr>
<td>pvCtrlBuf</td>
<td>Void *</td>
<td>If the packet received is bigger than the Rx buffer, the packet can be fragmented and sent in multiple such messages to the host. The structure <code>tstrM2mIpCtrlBuf</code> is used for sending this information. <code>ul6DataSize</code> has the number of bytes in the current buffer, and <code>ul6RemainingDataSize</code> has the remaining bytes of the packet received which is sent to the host in subsequent calls.</td>
</tr>
</tbody>
</table>
4. For transmission, the user can use the following API to send the TCP/IP packets to the ATWINC15x0.

```c
m2m_wifi_send_ethernet_pkt(tx_buf, num_bytes_transmitted)
```

Table 7-2. Ethernet Packet

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_buf</td>
<td>Void *</td>
<td>Valid pointer to the buffer having the data packet in the 802.3 Ethernet format</td>
</tr>
<tr>
<td>num_bytes_transmitted</td>
<td>Uint16</td>
<td>Size of the buffer</td>
</tr>
</tbody>
</table>

5. For reception, when the device receives any Ethernet packets it notifies the application through the callback eth_cb() which is called in case of M2M_WIFI_RESPEtherETNET_RX_PACKET in wifi_cb().
8. **TCP/ IP Networking Stack - lwIP**

The TCP/ IP stack used in this example is a third party Networking stack lwIP. The lwIP is a small independent implementation of the TCP/IP protocol suite, initially developed by Adam Dunkels. The focus of the lwIP TCP/IP implementation is to reduce resource usage while still having a full scale TCP.

The main features include:

- **Protocols** – IP, ICMP, UDP, TCP, IGMP, ARP, PPPoS, and PPPoE
- **Clients** – DHCP client, DNS client, AutoIP/APIPA (Zeroconf), and SNMP agent (private MIB support)
- **APIs** – specialized APIs to enhance performance and optional Berkeley-alike socket API
- **Extended features** – IP forwarding over multiple network interfaces, TCP congestion control, RTT estimation, and fast recovery/fast re-transmit
- **Add on applications** – HTTP server, SNTP client, SMTP client, ping, and NetBIOS nameserver
9. References
The following documents can be used for further study:

- ATWINC1500 WI-Fi Network Controller Software Design Guide
- ATWINC15x0 in Ethernet Mode
10. **Document Revision History**

Rev A - 03/2018

<table>
<thead>
<tr>
<th>Section</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document</td>
<td>Initial Release</td>
</tr>
</tbody>
</table>
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.


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, Heldo, JukeBlox, KeeLoq, KeeLoq logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTorch, 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.


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.
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® MCUs and dsPIC® DSCs, KEELQ® 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

<table>
<thead>
<tr>
<th>AMERICAS</th>
<th>ASIA/PACIFIC</th>
<th>ASIA/PACIFIC</th>
<th>EUROPE</th>
</tr>
</thead>
</table>
| 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  
Brazil  
Ribeirão Preto  
Tel: 011-0000-0000  
Fax: 011-0000-0000  
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  
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  
Australia - Sydney  
Tel: 61-2-9868-6733  
China - Beijing  
Tel: 86-10-8569-7600  
China - Chengdu  
Tel: 86-28-8665-5511  
China - Chongqing  
Tel: 86-23-8980-9588  
China - Dongguan  
Tel: 86-769-8702-9880  
China - Guangzhou  
Tel: 86-20-8755-8029  
China - Hangzhou  
Tel: 86-571-8792-8115  
China - Hong Kong SAR  
Tel: 852-2943-5100  
China - Nanjing  
Tel: 86-25-8473-2460  
China - Qingdao  
Tel: 86-653-8502-7355  
China - Shanghai  
Tel: 86-21-3326-8000  
China - Shenyang  
Tel: 86-24-2334-2829  
China - Shenzhen  
Tel: 86-755-2983-7252  
China - Xiamen  
Tel: 86-592-2388138  
China - Zhuhai  
Tel: 86-756-3210040  
India - Bangalore  
Tel: 91-80-3090-4444  
India - New Delhi  
Tel: 91-11-4160-8631  
India - Pune  
Tel: 91-20-4121-0141  
Japan - Osaka  
Tel: 81-6-6152-7160  
Japan - Tokyo  
Tel: 81-3-6880-3770  
Korea - Daegu  
Tel: 82-53-744-4301  
Korea - Seoul  
Tel: 82-2-554-7200  
Malaysia - Kuala Lumpur  
Tel: 60-3-7651-7906  
Malaysia - Penang  
Tel: 60-4-227-8870  
Philippines - Manila  
Tel: 63-2-634-9065  
Singapore  
Tel: 65-6334-8870  
Taiwan - Hsin Chu  
Tel: 886-3-577-8366  
Taiwan - Kaohsiung  
Tel: 886-7-213-7830  
Taiwan - Taipei  
Tel: 886-2-2508-8600  
Thailand - Bangkok  
Tel: 66-2-694-1351  
Vietnam - Ho Chi Minh  
Tel: 84-29-5448-2100  
Austria - Wels  
Tel: 43-7242-2244-39  
Denmark - Copenhagen  
Tel: 45-4450-2828  
Finland - Espoo  
Tel: 358-9-4520-820  
France - Paris  
Tel: 33-1-69-53-63-20  
Germany - Garching  
Tel: 49-8931-9700  
Germany - Haan  
Tel: 49-2129-376400  
Germany - Heilbronn  
Tel: 49-711-67-3636  
Germany - Karlsruhe  
Tel: 49-721-625370  
Germany - Munich  
Tel: 49-89-627-144-0  
Germany - Rosenheim  
Tel: 49-89-627-144-44  
Israel - Ra’anana  
Tel: 972-3-744-7705  
Italy - Milan  
Tel: 39-0331-742611  
Fax: 39-0331-466871  
Italy - Padova  
Tel: 39-049-7625286  
Netherlands - Drunen  
Tel: 31-416-690399  
Fax: 31-416-690340  
Norway - Trondheim  
Tel: 77-7289-7561  
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: 46-214708-60-40  
Sweden - Stockholm  
Tel: 46-8-5090-4654  
UK - Wokingham  
Tel: 44-118-921-5800  
Fax: 44-118-921-5820

© 2018 Microchip Technology Inc.  
Application Note  
DS00002671A-page 21