CONNECTING THE INTERNET OF THINGS

The 4th Age of Wireless™

GG101 MICROCHIP GETTING STARTED GUIDE

FOR ZG2100S-03, ZG2100S-04, ZG2100S-05, ZG2100S-06

GG-101.35
REVISION 5.2

Table of Contents

1. Development with ZeroG Wi-Fi

1.1 Overview

The Wi-Fi® PICtail™/PICtail™+ Daughtercard Board is an 802.11b demonstration board for evaluating the ZeroG Wireless ZG2100 Wi-Fi® controller on a Microchip Technology's processing platform. It is an expansion board compatible with the Explorer 16 and PICDEM™.NET 2 development boards.

1.2 Features

- ZG2100 Wi-Fi controller module fully integrating 802.11b MAC and RF PHY requirements
- Power regulator to enable use on 3.3V or 5V development boards
- PICtail and PICtail+ Daughter Board connection interface

1.3 Hardware

The following items are required for development or evaluation of the Microchip based ZeroG 802.11b solution:

> **WARNING:** ***The boards in this kit are highly sensitive to electrostatic discharge (ESD). Please ground yourself at all times while in contact with the boards.***

1. ZG2100PCB Wi-Fi® PICtail
2. Either of the following Microchip hardware development platforms:
   - Explorer16 (PIC24, dsPIC, or PIC32 depending on personality module)
   - PICDEM.Net2 (PIC18)
   - Microchip ZeroG software driver and integrated Microchip TCP/IP stack (v5.20), available on the disc accompanying the development kit or direct from Microchip's website (see the *Appendix* for more information)
3. Power supply (9v, 300mA)
4. C compiler(s), downloadable from the Microchip website (please see the *Appendix* for download information)
   - MPLAB C compiler for PIC18 v3.34
   - MPLAB C compiler for PIC24/dsPIC v3.20
   - MPLAB C compiler for PIC32 v1.05
5. Microchip MPLAB IDE v8.40 (please see the *Appendix* for download information)
6. 802.11 access point (b, b/g, or b/g/draft n) required for using the development board in infrastructure (BSS) mode.
   - Linksys WRT54G or WRT54G2 is recommended

1.4 Software

This guide has sections on installing the Microchip MPLAB IDE, the Microchip and ZeroG SDK and TCP/IP stack, and the necessary changes to configure the demo software for networks with different characteristics than the default settings. Please see the *Appendix* for direct links to all the Microchip software you may require for your project. The latest Microchip documentation can always be found on their website, and takes precedence over software bundles on the installation CD. Likewise, the latest ZeroG documentation can be found on the ZeroG support website, and is the most accurate.

This guide documents how to configure the wireless network by hard coding the network parameters into the software, then compiling, and storing this information on the Microchip MCU. The software uses C calls to modify the values in variables used to keep the configurations. Customer application code can thus create a user interface that allows scanning for networks and then configuring based on the end users selection. As an example of using the variables, the included demo projects have a number of source files to allow run time configuration of the networks (e.g. select ad hoc or infrastructure, change SSID, change security methods and keys, etc.). These files are located in the ZeroG source directory (located at `C:\Microchip Solutions\Microchip\TCPIP Stack\ZeroG` if installed to the default location) and start with `ZGConsole*.c` This example code creates a very powerful run time command line interface for modifying all attributes of the wireless network. The interface is accessed via a terminal connected to the RS232 port of the Microchip development board. Documentation for this command line interface is covered under the ZGS2101 CLI Usage Document (see the *Appendix* for link to document).

1.5 Common Terms and Definitions

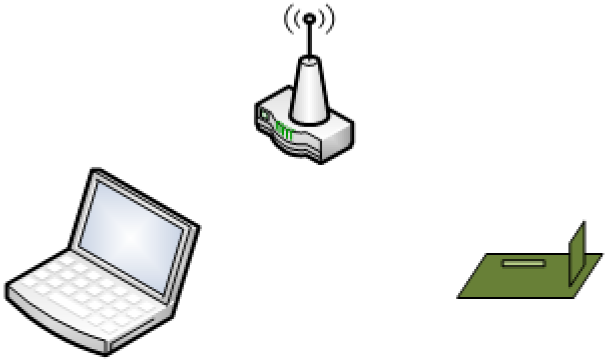
Table 1: Common Terms and Definitions

Term	Definition
STA	A station (local station) is a device on the network, typically referring to a wireless device. This can be a laptop, PC, or the Microchip development board with ZeroG PICtail.
LAN	A local area network (or local network) is just a collection of computers talking to each other. In the simplest form, this consists of two devices talking through a wireless access point or router (see picture below). Local networks can talk to each other, or, with a WAN (internet) connection, they can talk to other networks over the internet.
WAP	Wireless access point (access point, or AP for short) is a device that creates a wireless network that multiple wireless devices can connect to. When connected to a wired network, wired and wireless devices can communicate with each other. Typically, most access points and routers come as one single unit, making the connection between wired, wireless, and internet

	seamless.
Router	A network device that directs and forwards traffic. Commonly, a router and an access point are combined together, so that wired and wireless devices can talk to each other.
DHCP	Dynamic host configuration protocol. This protocol is an application layer protocol that manages the IP network. This reduces the amount of manual intervention required when putting a new device on the network. You'll see DHCP in action mostly when you try to connect to a network, and you are given an IP address by the DHCP server.
OUI	Organizationally unique identifier. This is a 24-bit number (3 bytes) that composes the first three bytes (octets) of the six byte MAC address. The OUI is managed by the IEEE. The OUI that identifies ZeroG Wireless currently is 00:1E:C0. With each OUI, you are guaranteed 2^{24} unique combinations that can be used for device identification.
MAC address	The media access control address is a unique identifier to each device on a network. For a given OUI, there can be 2^{24} uniquely programmed MAC addresses.
BSS	The basic service set is the basic building block of a wireless LAN. This usually consists of an AP, with one or more wireless stations. This is commonly referred to as infrastructure mode.
IBSS	The independent basic service set is a network with no controlling AP. The first device to start the network will broadcast the SSID, and other local stations can join. This is commonly referred to as ad-hoc mode.
SSID	The service set identifier is a name for the wireless network.
WEP	Wired equivalent privacy is an encryption mechanism for wireless networks. Most APs implement two different flavors of WEP, 64-bit WEP (WEP-40) and 128-bit WEP (WEP-104). Several weaknesses have been identified in WEP, resulting in networks that are easily hacked. The use of WEP has been superseded by WPA/WPA2.
WPA	Wi-Fi protected access is the implementation of the IEEE 802.11i specification. It is considered a more secure alternative to WEP.
WAN	A wide area network is a network that connects other types of networks (LANs) together. The most famous WAN is the internet.
MCU	Microcontroller unit. In the case of this demonstration, this will either be a PIC18 (PICDEM.net 2), PIC24 (Explorer 16) or a PIC32 (Explorer 16).

1.6 Local Network Examples

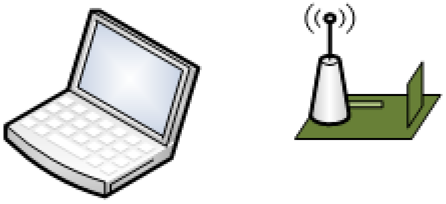
A common example of a local network operating in infrastructure mode is shown in Figure 1. This network shows a laptop computer and the Microchip development board with ZeroG PICtail communicating with each other through a wireless access point and router. This network can gain access to the internet if the router is connected to a WAN.



Legend		
Simplest Local Network (BSS)		
Symbol	Count	Description
	1	Wireless access point
	1	Router
	1	ZeroG development board
	1	Laptop computer

Figure 1: BSS Network

Another example of a common local network is the ad-hoc network, shown in Figure 2. In this example, we assume that the Microchip development board with ZeroG PICtail is the first station to broadcast that it wants to create the network (and it is successfully able to do so). In this case, the laptop will then join the ad-hoc network after the development board has gone through the steps of setting up the ad-hoc network.



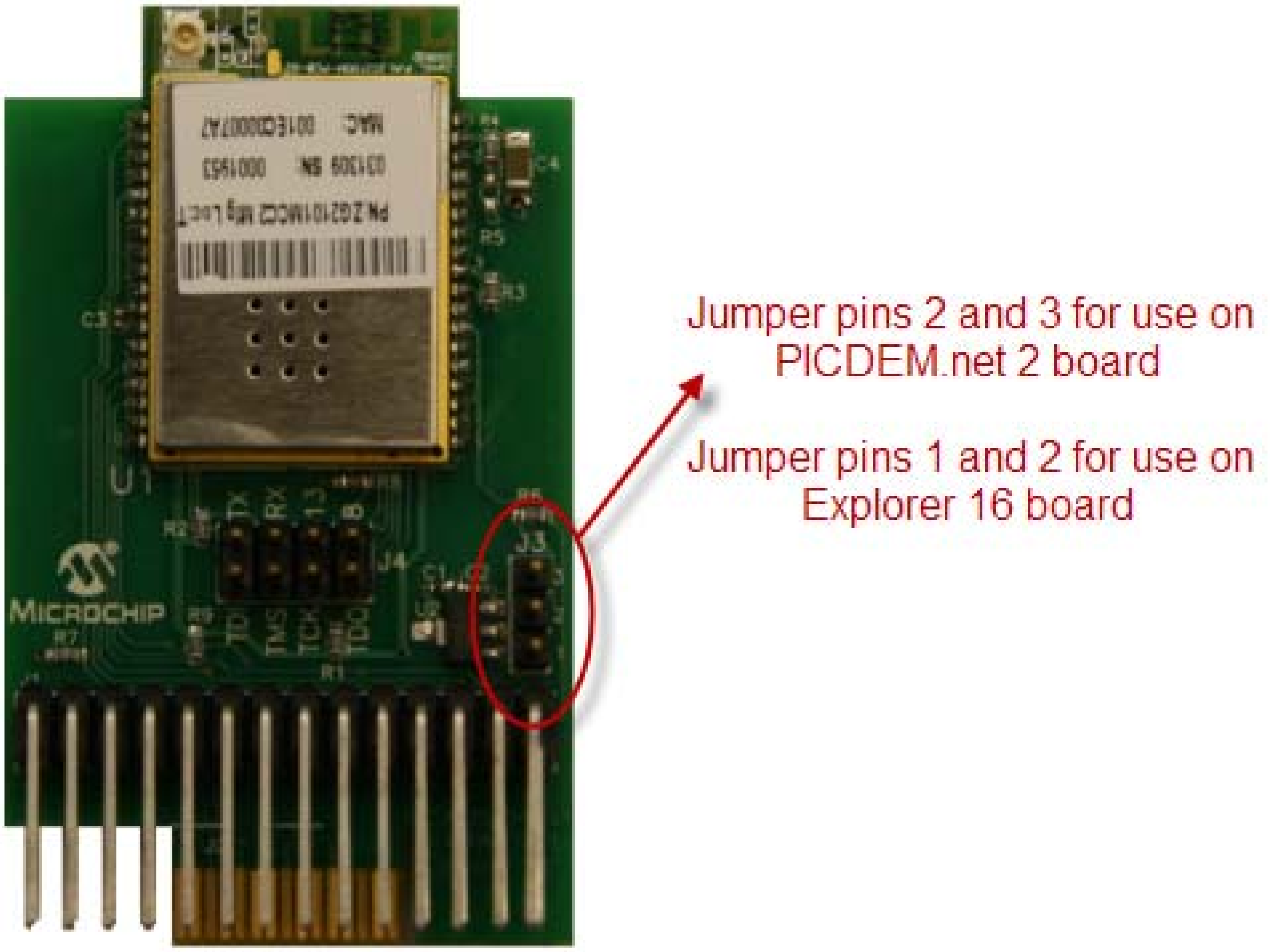
Legend		
Simplest Local Network (IBSS)		
Symbol	Count	Description
	1	ZeroG development board ad hoc
	1	Laptop computer

Figure 2: IBSS Network

2. Hardware Setup and Configuration

2.1 PICtail Setup

Figure 3 shows the ZeroG PICtail that will plug into either the PICtail Plus (Explorer 16) using the card edge connector or the PICtail slot (PICDEM.net 2) using the pin header on the development board. Because the two development boards operate at different voltages, the end user will need to set the jumper JP3 on the ZeroG PICtail correctly for the setup that is being used.



Jumper pins 2 and 3 for use on PICDEM.net 2 board

Jumper pins 1 and 2 for use on Explorer 16 board

Figure 3: ZeroG PICtail

When inserting the PICtail into the development board, always orient the PICtail with the module facing the microcontroller. Also, for the Explorer 16 board, if you are using the dsPIC33FJ256GP710 PIM module, you will need to insert the card edge into the middle slot of the socket. If you are using the PIC24FJ128GA010, you can insert the PICtail into either the upper or lower slot. Please see the section *Hardware Configuration Options* for instructions on how to configure the software depending on which slot is used. Out of the box, the software in the demos is configured to use the upper slot.

2.2 Connecting the Development Board

WARNING: The boards in this kit are highly sensitive to electrostatic discharge (ESD). Please ground yourself at all times while in contact with the boards.

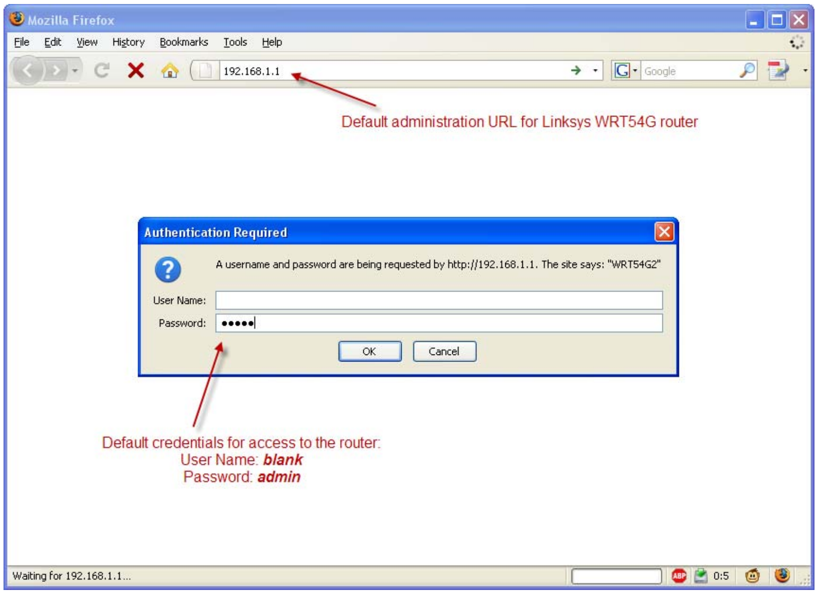
1. Connect the RJ11 cable (grey phone cable) to the RJ11 port on the development board to the ICD.
2. Connect the serial cable to the serial port (UART port) of the development board and to the serial port (COM port) on the PC. (Typically, the default port on the PC is COM 1, but this default number may differ from PC to PC.) This is only required if you want to monitor the debug messages coming from the PICtail. Please see *Serial Monitor Setup* for more information on setting up the serial connection correctly.
3. Connect the USB cable from the ICD to the PC.
4. Power on the router, and connect the PC to the router with an Ethernet cable. If a wireless laptop (or PC) is being used, then associate the computer with the correct SSID of the wireless access point.

2.3 Wireless Access Point Setup

The following instructions show the settings and configuration options for the Linksys WRT54G2 Wireless-G Broadband Router. Note that in this particular situation, the use of the term “access point” and “router” are synonymous, and refer to the combination of these two networking parts into a single unit. While the screenshots in this section are specific to this particular access point, the concepts and items that need to be configured should be identical.

2.3.1 Accessing the Access Point Configuration Pages

The screenshot shown in Figure 4 illustrates the web interface to the access point. These pages can be viewed from any web browser. Enter the address http://192.168.1.1 into the URL address bar. When prompted with the authentication text box, leave the “User Name:” field blank, and use “admin” for the password in the “Password:” field.



192.168.1.1

Default administration URL for Linksys WRT54G router

Authentication Required

A username and password are being requested by http://192.168.1.1. The site says: "WRT54G2"

User Name:

Password: •••••

Default credentials for access to the router:
User Name: ***blank***
Password: ***admin***

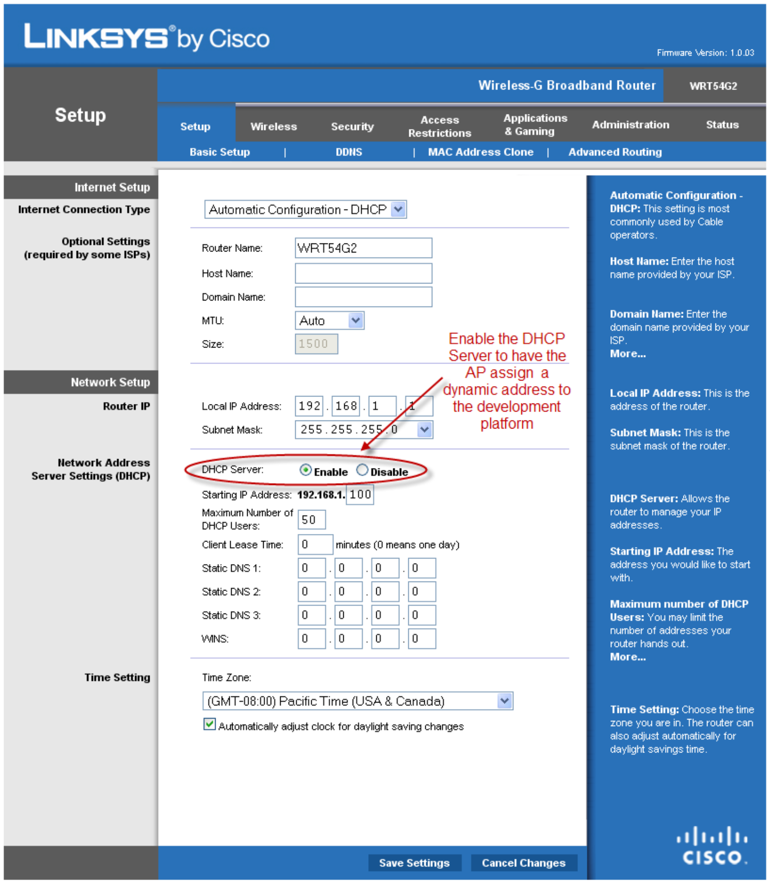
Waiting for 192.168.1.1...

Figure 4: Access Point Login

2.3.2 Main Access Point Configuration Page

After authenticating with the access point, you will be presented with a configuration page similar to

The settings on this page are for configuring the router portion of the system. Most of the default settings that are programmed from the factory are adequate for the demo. By default, the WRT54G router acts as a DHCP server. For these demos, it is best to have the access point be the DHCP server, as is show in Figure 5.



Internet Setup

Internet Connection Type: Automatic Configuration - DHCP

Optional Settings (required by some ISPs)
Router Name: WRT54G2
Host Name:
Domain Name:
MTU: Auto
Size: 1500

Network Setup

Router IP
Local IP Address: 192 . 168 . 1 . 1
Subnet Mask: 255 . 255 . 255 . 0

Network Address Server Settings (DHCP)
DHCP Server: ● Enable ○ Disable
Starting IP Address: 192.168.1. 100
Maximum Number of DHCP Users: 50
Client Lease Time: 0 minutes (0 means one day)
Static DNS 1: 0 . 0 . 0 . 0
Static DNS 2: 0 . 0 . 0 . 0
Static DNS 3: 0 . 0 . 0 . 0
WINS: 0 . 0 . 0 . 0

Time Setting
Time Zone: (GMT-08:00) Pacific Time (USA & Canada)
☑ Automatically adjust clock for daylight saving changes

Enable the DHCP Server to have the AP assign a dynamic address to the development platform

Automatic Configuration - DHCP: This setting is most commonly used by Cable operators.

Host Name: Enter the host name provided by your ISP.

Domain Name: Enter the domain name provided by your ISP.
More...

Local IP Address: This is the address of the router.

Subnet Mask: This is the subnet mask of the router.

DHCP Server: Allows the router to manage your IP addresses.

Starting IP Address: The address you would like to start with.

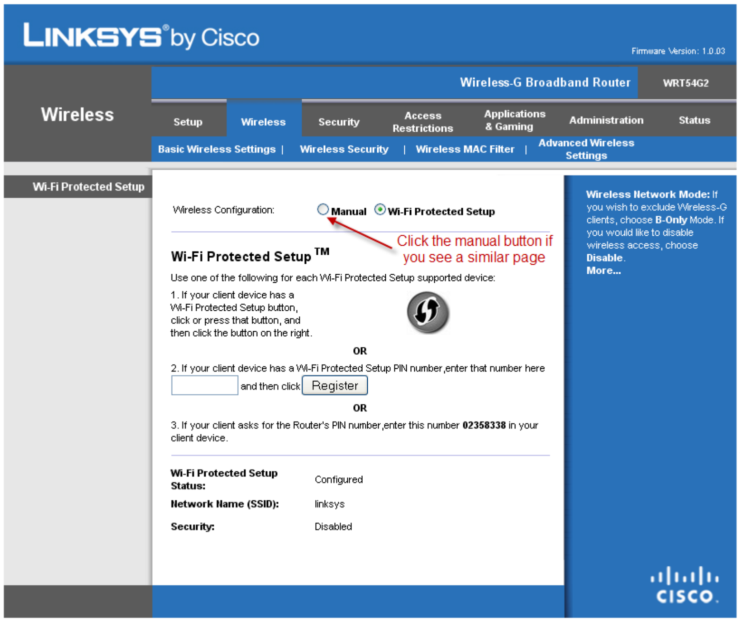
Maximum number of DHCP Users: You may limit the number of addresses your router hands out.
More...

Time Setting: Choose the time zone you are in. The router can also adjust automatically for daylight savings time.

Figure 5: AP Configuration Setup

2.3.3 Setting up the Wireless Access Point

After clicking on the “Wireless” tab at the top of the page, you will see either of the following two configuration screens. If you see the screen show in Figure 6, then you will need to click the “Manual” radio button. At this point, everybody should see the screen shown in Figure 7.



Wireless Configuration: ○ Manual ● Wi-Fi Protected Setup

Click the manual button if you see a similar page

Wi-Fi Protected Setup ™

Use one of the following for each Wi-Fi Protected Setup supported device:

1. If your client device has a Wi-Fi Protected Setup button, click or press that button, and then click the button on the right.

OR

2. If your client device has a Wi-Fi Protected Setup PIN number,enter that number here and then click Register

OR

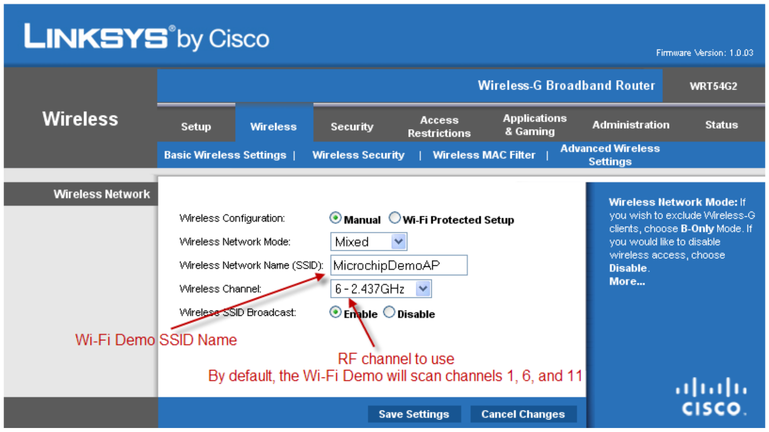
3. If your client asks for the Router's PIN number,enter this number **02358338** in your client device.

Wi-Fi Protected Setup Status: Configured
Network Name (SSID): linksys
Security: Disabled

Wireless Network Mode: If you wish to exclude Wireless-G clients, choose **B-Only** Mode. If you would like to disable wireless access, choose **Disable**.
More...

Figure 6: W-Fi Protected Setup

Figure 7 shows the basic wireless settings needed for the demonstrations to follow.



Wireless Configuration: ● Manual ○ Wi-Fi Protected Setup
Wireless Network Mode: Mixed
Wireless Network Name (SSID): MicrochipDemoAP
Wireless Channel: 6 - 2.437GHz
Wireless SSID Broadcast: ● Enable ○ Disable

Wireless Network Mode: If you wish to exclude Wireless-G clients, choose **B-Only** Mode. If you would like to disable wireless access, choose **Disable**.
More...

Wi-Fi Demo SSID Name

RF channel to use
By default, the Wi-Fi Demo will scan channels 1, 6, and 11

Figure 7: Basic Wireless Setup

Table 2 shows a summary of the options that need to be setup and the expected values:

Table 2: Wireless Settings

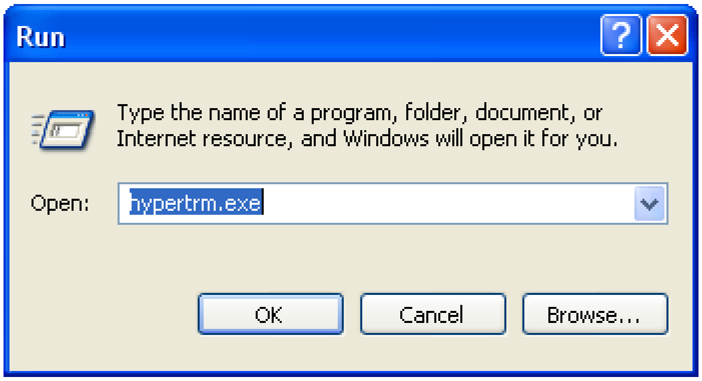
Option	Value/Setting
Wireless network name (SSID)	MicrochipDemoAP (case sensitive)
Wireless channel	Either channel 1, 6, or 11
Wireless network mode	Either mixed mode (b and g service) or B-only

After setting everything up, click the “Save Settings” button. The access point should confirm that the settings have been saved successfully. This is all that is required to setup the access point for the demonstrations. If you are experimenting with wireless security modes, then please see the appropriate section: WEP or WPA/WPA2.

2.4 Serial Monitor Setup

The ZeroG PICtail prints out status and state information to the serial UART port on the Microchip development board. This can be useful for debug purposes. This section will walk you through setting up a terminal session to view this output.

First, start by launching HyperTerminal from either the Run dialog in the Windows Start menu or by launching the HyperTerminal application directly from the Start menu.



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: hypertrm.exe

Figure 8: Starting HyperTerminal from Run Dialog

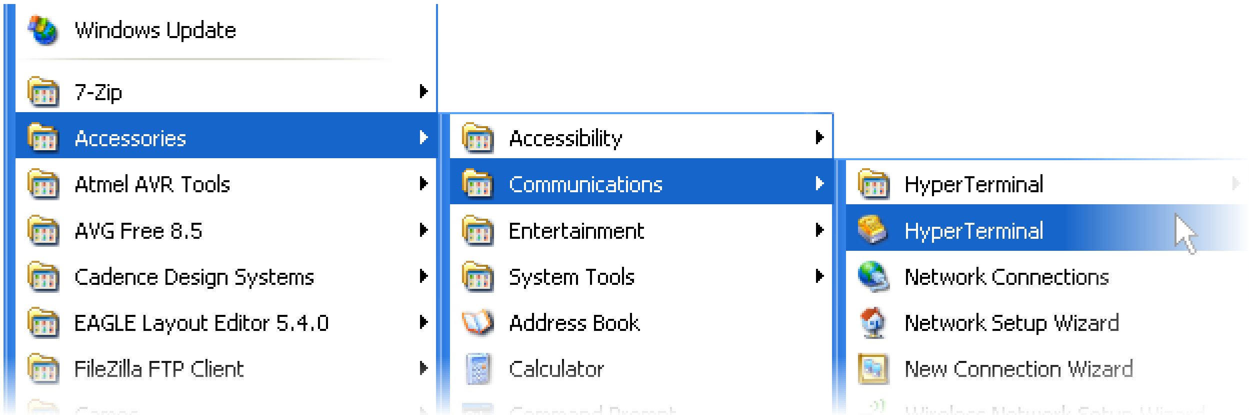
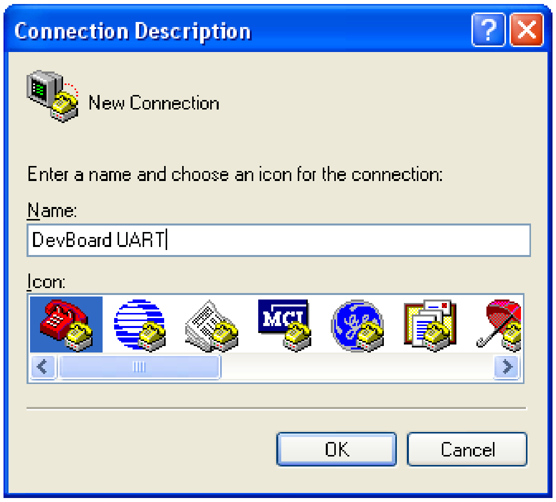


Figure 9: Starting HyperTerminal from Start Menu

Name your connection and press the **OK** button.



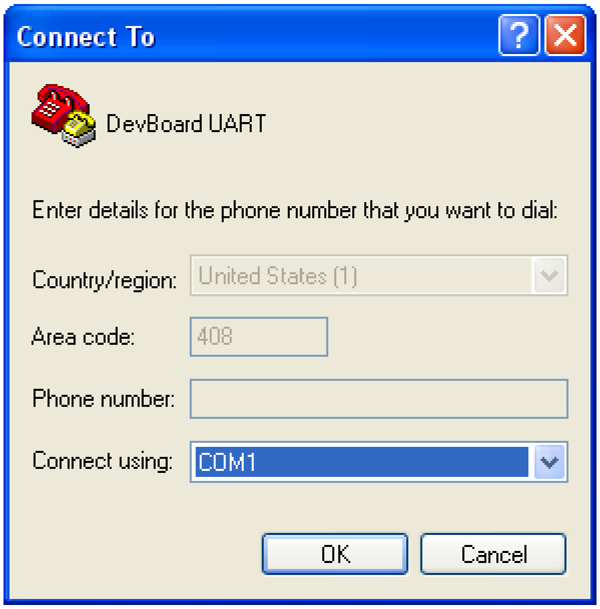
Enter a name and choose an icon for the connection:

Name: DevBoard UART

Icon:

Figure 10: HyperTerminal Name Setup

Select the appropriate COM port as the listening port. Normally, this will be COM1, however check your system to make sure you select the correct port. Press the **OK** button after completion.



DevBoard UART

Enter details for the phone number that you want to dial:

Country/region: United States (1)

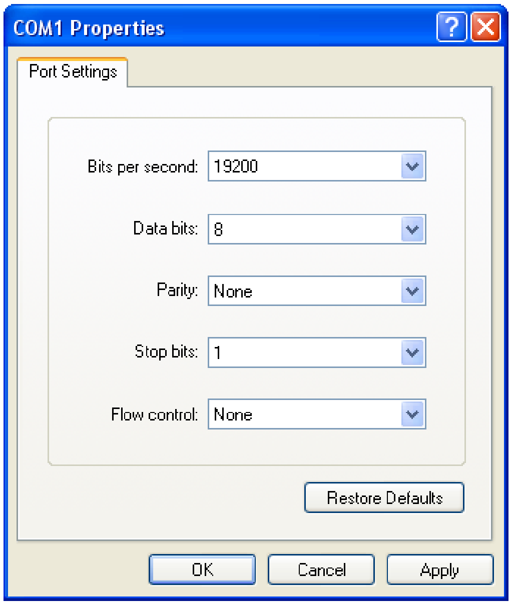
Area code: 408

Phone number:

Connect using: COM1

Figure 11: COM Port Connection

Setup the communication port parameters as shown in Figure 12. Press the **OK** button when finished. The UART monitor is now setup and will print out messages from the ZeroG PICtail once it is running.



Bits per second: 19200

Data bits: 8

Parity: None

Stop bits: 1

Flow control: None

Figure 12: COM Communication Settings

3. Software Setup and Configuration

3.1 Software Items to Install

The list of software items to install includes the following:

- Microchip MPLAB integrated development environment v8.40 or later
- Microchip MPLAB C compiler
- Microchip in-circuit debugger (ICD 2 or ICD 3) driver
- Microchip TCP/IP stack installer (which contains the ZeroG Wi-Fi driver)

Note that the screenshots may show references to older versions of the MPLAB IDE, as well as older compiler and TCP/IP stack versions. These screenshots are for visual cues only, and the latest versions should be installed from the Microchip website.

Also note that Microchip is recommending that new designs use the ICD3 debugger. These instructions detail setup with the ICD2, but the instructions are similar with an ICD3.

3.2 Installing the Microchip MPLAB IDE v8.40 or Later

Please see the *Appendix* for a link to the location of the MPLAB IDE on the Microchip website. After downloading the installer, execute the setup file, and follow the GUI instructions for installing MPLAB IDE on your system. If required, restart the computer after installation is complete.

3.3 Installing the Microchip MPLAB C Compiler

There are different versions of the C compiler for different PIC microprocessors. For users with a PICDEM.net 2 board, you will need the C compiler for PIC18 MCUs. If you have a Explorer 16 board, you will either need the PIC24, dsPIC, or PIC32 C compiler (or all of them). In the event you are doing development on multiple platforms, you'll need to install the appropriate number of compilers.

Microchip offers evaluation copies of their compilers (student versions) that can be downloaded from the Microchip website (links provided in the *Appendix*). The evaluation copies are usable for 60 days, after which, you will still be able to compile code, but will be presented with a warning each time the compiler is invoked and will lose the ability to do compiler optimization on the code.

After downloading the installer, run through the setup to install the compiler on your system.

3.4 Installing the Microchip In-Circuit Debugger Driver

After installing the MPLAB IDE and compilers, the last step to getting the development environment running is to install the ICD (ICD2 or ICD3). If you are installing an ICD2, care must be taken **not** to install the Windows recommended driver. For the ICD3, the Windows recommended driver is acceptable. Microchip provides detailed installation instructions for the ICD2/ICD3 that is installed with MPLAB IDE. Please see the *Appendix* for information on the location of these help files.

3.5 Installing the Microchip TCP/IP Stack with ZeroG Driver

The TCP/IP stack and ZeroG driver are available in two different forms. A CDROM disc with the installer is bundled with the SDK provided by ZeroG Wireless. Optionally, the installer can be downloaded from the Microchip website (link provided in the *Appendix*).

The installer by default will install the stack code, driver, documentation, and demo project files in the directory `C:\Mirochip Solutions`. If this is not the first time the code has been installed, take care to save any custom work in the `C:\Mirochip Solutions` directory to another directory so that it will not be overwritten.

3.6 Installing Interim Code Releases

It may become necessary to install special interim code releases that are not part of officially available code library provided by Microchip and ZeroG. This could be due to various reasons, such as high priority bug fixes, new features that are required by customers, and the like. In this case, you may get a special zipped version of the `C:\Mirochip Solutions` directory.

To install this interim code, follow the following instructions:

1. Save any open files in the `C:\Mirochip Solutions` directory and quit the MPLAB IDE, if necessary.
2. Open the zip file to see which directories are affected.
3. Be sure to save any custom work you may wish to keep in these directories to another location.
4. Unzip or copy the directories in the zip archive to the `C:\Mirochip Solutions` directory.

When you restart the MPLAB IDE, the new code should be usable.

4. Sample Application Demonstrations

The SDK comes equipped with two, out-of-the-box Wi-Fi demonstrations to showcase the ZeroG Wi-Fi module. The first application is a powerful Wi-Fi demonstration that shows a webserver, and allows you to do a lot of application level activity, such as send and process form data, send email, upload files, and so forth. This demo highlights a lot of applications that are supported by the Microchip TCP/IP stack and how they can be easily used with Wi-Fi. The second demonstration is a throughput performance demonstration using a tool called iperf, which is a commonly used networking test tool. iperf will allow you to measure the throughput bandwidth on the Wi-Fi link for both receive and transmit.

This section is broken up into the following logical sections:

- Opening existing projects
- Configurable options during compile time
- Compiling and downloading code images to the development board
- Walkthrough of the Wi-Fi Demo application
- Walkthrough and instructions on running the iperf demonstration.

4.1 Opening Existing Projects

After starting MPLAB IDE, you will need to open an existing project. This guide focuses on the TCPIP WiFi Demo App running on an Explorer 16 development board with a PIC24 PIM module installed. However, the configuration, compile, and downloading of the code image to the PIC MCU is identical for all development boards, MCUs, and demo applications.

To open an existing project, select Open… from the Project menu as shown here.

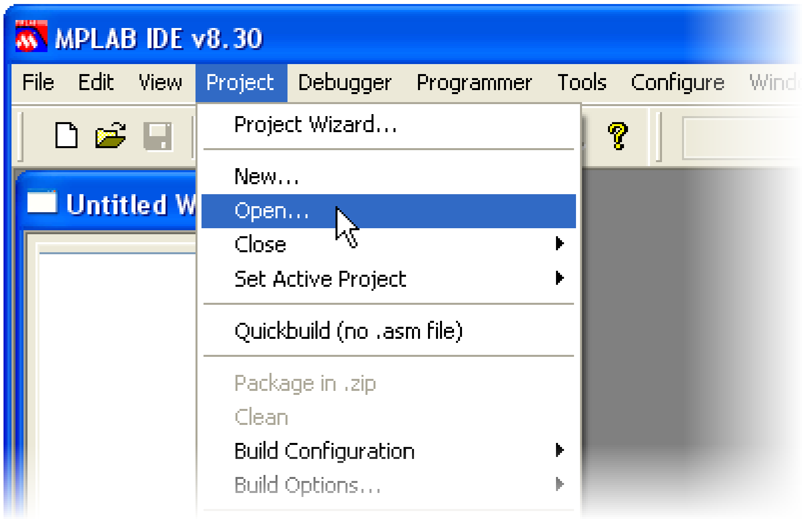
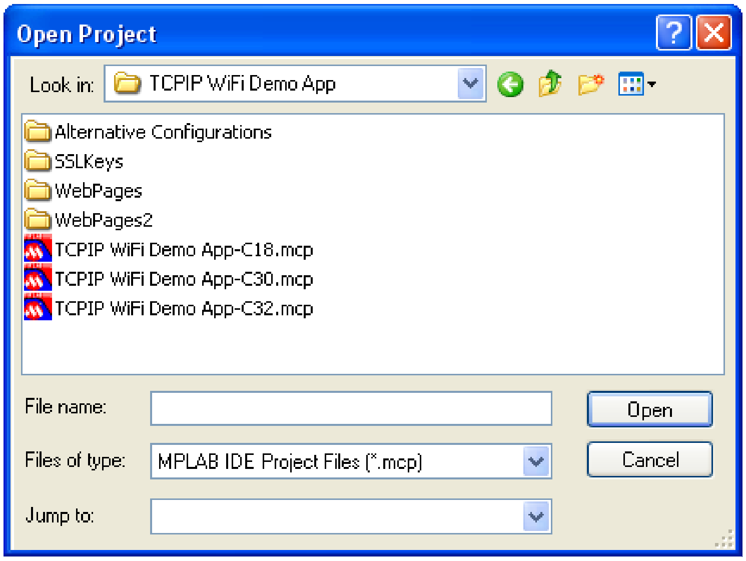


Figure 13: Open an existing MPLAB Project

Browse to `C:\Microchip Solutions\TCPIP WiFi Demo App`. You should see something similar to Figure 14.

Select the appropriate .mcp file for your development board. For the PICDEM.net 2 board with the PIC18 MCU, select TCPIP Wifi Demo App-C18.mcp. For the Explorer 16 board, select TCPIP Wifi Demo App-C30.mcp if you are using the PIC24 MCU or dsPIC. Select TCPIP Wifi Demo App-C32.mcp if you are using a PIC32 MCU. Open the appropriate project file.



Look in: TCPIP WiFi Demo App

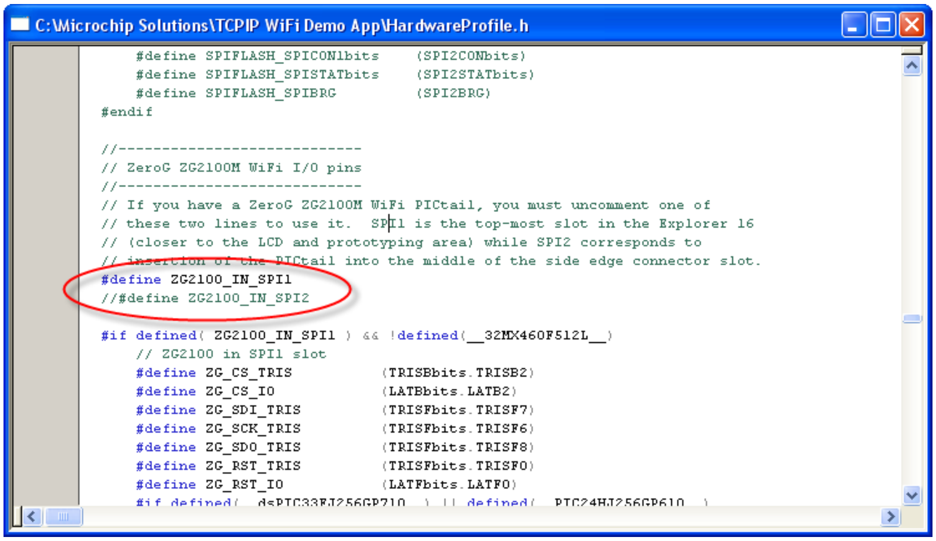
Alternative Configurations
SSLKeys
WebPages
WebPages2
TCPIP WiFi Demo App-C18.mcp
TCPIP WiFi Demo App-C30.mcp
TCPIP WiFi Demo App-C32.mcp

File name:
Files of type: MPLAB IDE Project Files (*.mcp)
Jump to:

Figure 14: Open Project dialog

4.2 Hardware Configuration Options

Depending on the development board used, the `HardwareProfile.h` file must be changed to match the configuration of which slot the PICtail is plugged into. Figure 15 shows the location of the changes needed:


C:\Microchip Solutions\TCPIP WiFi Demo App\HardwareProfile.h

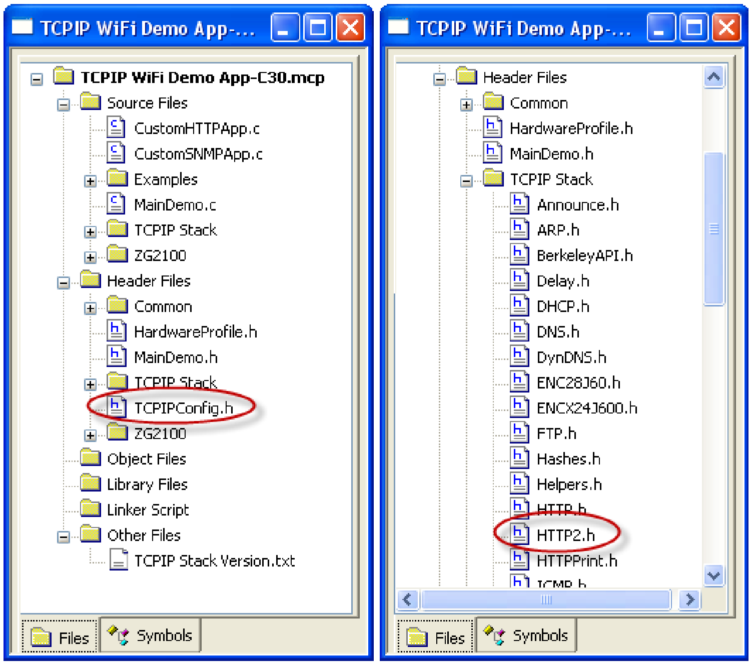
```
    #define SPIFLASH_SPICON1bits     (SPI2CONbits)
    #define SPIFLASH_SPISTATbits     (SPI2STATbits)
    #define SPIFLASH_SPIBRG          (SPI2BRG)
#endif

//----------------------------
// ZeroG ZG2100M WiFi I/O pins
//----------------------------
// If you have a ZeroG ZG2100M WiFi PICtail, you must uncomment one of
// these two lines to use it.  SPI1 is the top-most slot in the Explorer 16
// (closer to the LCD and prototyping area) while SPI2 corresponds to
// insertion of the PICtail into the middle of the side edge connector slot.
#define ZG2100_IN_SPI1
//#define ZG2100_IN_SPI2

#if defined( ZG2100_IN_SPI1 ) && !defined(__32MX460F512L__)
    // ZG2100 in SPI1 slot
    #define ZG_CS_TRIS          (TRISBbits.TRISB2)
    #define ZG_CS_IO            (LATBbits.LATB2)
    #define ZG_SDI_TRIS         (TRISFbits.TRISF7)
    #define ZG_SCK_TRIS         (TRISFbits.TRISF6)
    #define ZG_SDO_TRIS         (TRISFbits.TRISF8)
    #define ZG_RST_TRIS         (TRISFbits.TRISF0)
    #define ZG_RST_IO           (LATFbits.LATF0)
    #if defined( dsPIC33FJ256GP710 ) || defined( PIC24HJ256GP610 )
```

Figure 15: HardwareProfile.h SPI Options

For PICDEM.net 2 development board users, there is only one configuration for the PICtail, and `ZG2100_IN_SPI1` must be defined. For the Explorer 16 board, if you are using the PIC24FJ128GA010, you can use either `ZG2100_IN_SPI1` or `ZG2100_IN_SPI2`, so long as it matches the location that the PICtail is plugged into the card edge connector (SPI1 refers to the upper location, closest to the LCD). If you are using the dsPIC33FJ256GP710, currently, this PIM will only work when `ZG2100_IN_SPI2` is defined and the PICtail is plugged into the middle card edge socket.

4.3 Compile-time Configuration Options

There are two main files that contain most of the compile-time options for the demonstrations, `TCPIPConfig.h` and `HTTP2.h`. Both of these files can be viewed from within MPLAB IDE, by using the file navigator. Note that HTTP2.h is located under Header Files/TCPIP Stack.



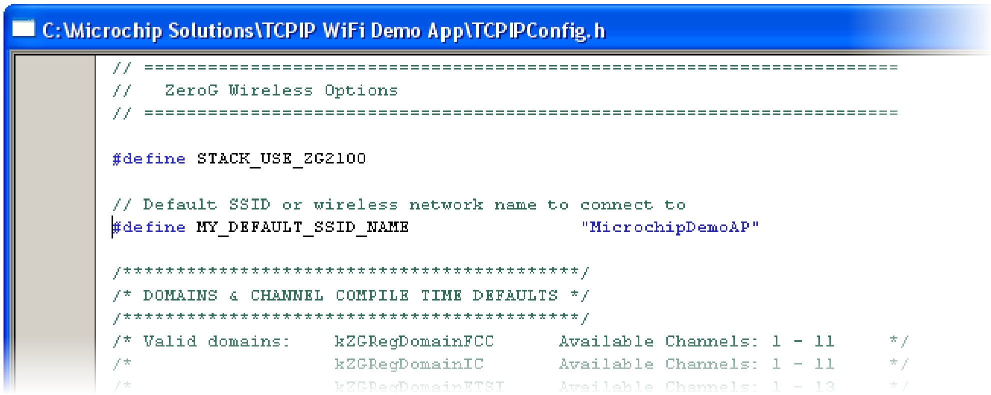
```
TCPIP WiFi Demo App-C30.mcp
    Source Files
        CustomHTTPApp.c
        CustomSNMPApp.c
        Examples
        MainDemo.c
        TCPIP Stack
        ZG2100
    Header Files
        Common
        HardwareProfile.h
        MainDemo.h
        TCPIP Stack
        TCPIPConfig.h
        ZG2100
    Object Files
    Library Files
    Linker Script
    Other Files
        TCPIP Stack Version.txt
```

```
Header Files
    Common
    HardwareProfile.h
    MainDemo.h
    TCPIP Stack
        Announce.h
        ARP.h
        BerkeleyAPI.h
        Delay.h
        DHCP.h
        DNS.h
        DynDNS.h
        ENC28J60.h
        ENCX24J600.h
        FTP.h
        Hashes.h
        Helpers.h
        HTTP.h
        HTTP2.h
        HTTPPrint.h
        ICMP.h
```

Figure 16: Configuration File Locations

4.3.1. SSID

The network name is stored in the variable `MY_DEFAULT_SSID_NAME` in `TCPIPConfig.h`. To change it, modify the defined name.



```
// ========================================================================
//   ZeroG Wireless Options
// ========================================================================

#define STACK_USE_ZG2100

// Default SSID or wireless network name to connect to
#define MY_DEFAULT_SSID_NAME                "MicrochipDemoAP"

/*********************************************/
/* DOMAINS & CHANNEL COMPILE TIME DEFAULTS */
/*********************************************/
/* Valid domains:     kZGRegDomainFCC      Available Channels: 1 - 11      */
/*                    kZGRegDomainIC       Available Channels: 1 - 11      */
/*                    kZGRegDomainETSI     Available Channels: 1 - 13      */
```

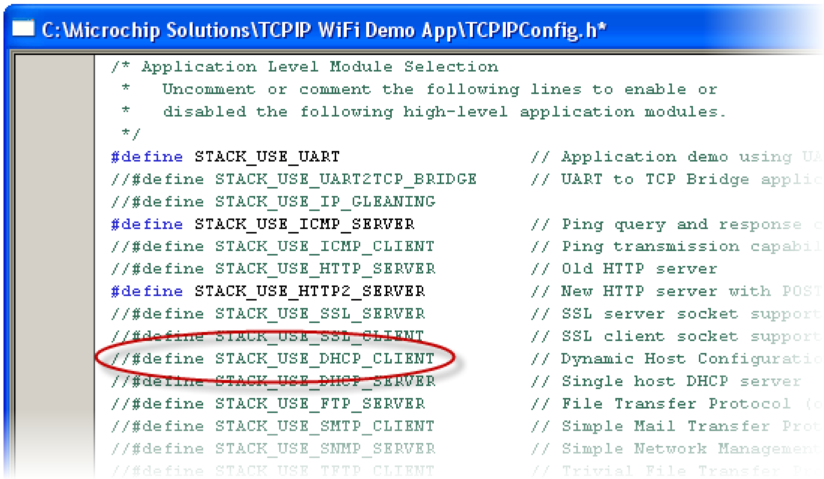
Figure 17: Modifying SSID

4.3.2 Static IP Address

By default, the demonstrations use DHCP, and rely on the DHCP server in the access point to give the development board an IP address on the network. To enable the use of a static IP address, a couple of things must be done.

First, the static IP address that is selected must be on the same subnet as the access point. As an example, by default, the Linksys WRT54G manages IP addresses on the subnet 192.168.1.x (192.168.1.1 is reserved for the access point itself). Additionally IP addresses above 192.168.1.100 are dynamically managed by the DHCP server, so it is best to assign a static IP address in the range of 192.168.1.2 – 192.168.1.99.

Second, two sections of code need to be changed to enable use of the static IP address. The first is to stop the dev kit from trying to request a dynamic address. Comment out the variable `STACK_USE_DHCP_CLIENT`, as shown.

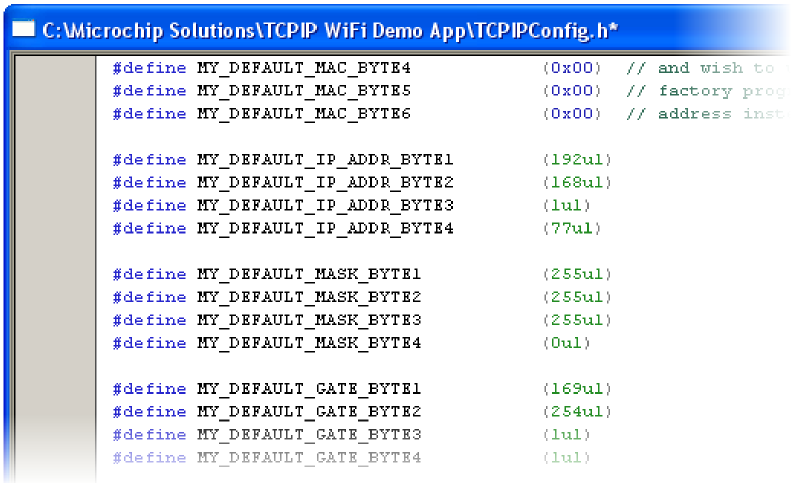


C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIPConfig.h*

```
/* Application Level Module Selection
 *   Uncomment or comment the following lines to enable or
 *   disabled the following high-level application modules.
 */
#define STACK_USE_UART                   // Application demo using UA
//#define STACK_USE_UART2TCP_BRIDGE      // UART to TCP Bridge applic
//#define STACK_USE_IP_GLEANING
#define STACK_USE_ICMP_SERVER            // Ping query and response c
//#define STACK_USE_ICMP_CLIENT          // Ping transmission capabil
//#define STACK_USE_HTTP_SERVER          // Old HTTP server
#define STACK_USE_HTTP2_SERVER           // New HTTP server with POST
//#define STACK_USE_SSL_SERVER           // SSL server socket support
//#define STACK_USE_SSL_CLIENT           // SSL client socket support
//#define STACK_USE_DHCP_CLIENT          // Dynamic Host Configuratio
//#define STACK_USE_DHCP_SERVER          // Single host DHCP server
//#define STACK_USE_FTP_SERVER           // File Transfer Protocol (o
//#define STACK_USE_SMTP_CLIENT          // Simple Mail Transfer Prot
//#define STACK_USE_SNMP_SERVER          // Simple Network Management
//#define STACK_USE_TFTP_CLIENT          // Trivial File Transfer Pro
```

Figure 18: Disable DHCP Client

Also, define the IP address you would like to statically use in `MY_DEFAULT_IP_ADDR_BYTEx` and match the default mask (`MY_DEFAULT_MASK_BYTEx`) to match the router (for the default Linksys, this is 255.255.255.0), both of which are shown in here:



C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIPConfig.h*

```
#define MY_DEFAULT_MAC_BYTE4            (0x00)  // and wish to
#define MY_DEFAULT_MAC_BYTE5            (0x00)  // factory prog
#define MY_DEFAULT_MAC_BYTE6            (0x00)  // address inst

#define MY_DEFAULT_IP_ADDR_BYTE1        (192ul)
#define MY_DEFAULT_IP_ADDR_BYTE2        (168ul)
#define MY_DEFAULT_IP_ADDR_BYTE3        (1ul)
#define MY_DEFAULT_IP_ADDR_BYTE4        (77ul)

#define MY_DEFAULT_MASK_BYTE1           (255ul)
#define MY_DEFAULT_MASK_BYTE2           (255ul)
#define MY_DEFAULT_MASK_BYTE3           (255ul)
#define MY_DEFAULT_MASK_BYTE4           (0ul)

#define MY_DEFAULT_GATE_BYTE1           (169ul)
#define MY_DEFAULT_GATE_BYTE2           (254ul)
#define MY_DEFAULT_GATE_BYTE3           (1ul)
#define MY_DEFAULT_GATE_BYTE4           (1ul)
```

Figure 19: Setting up Static IP Address

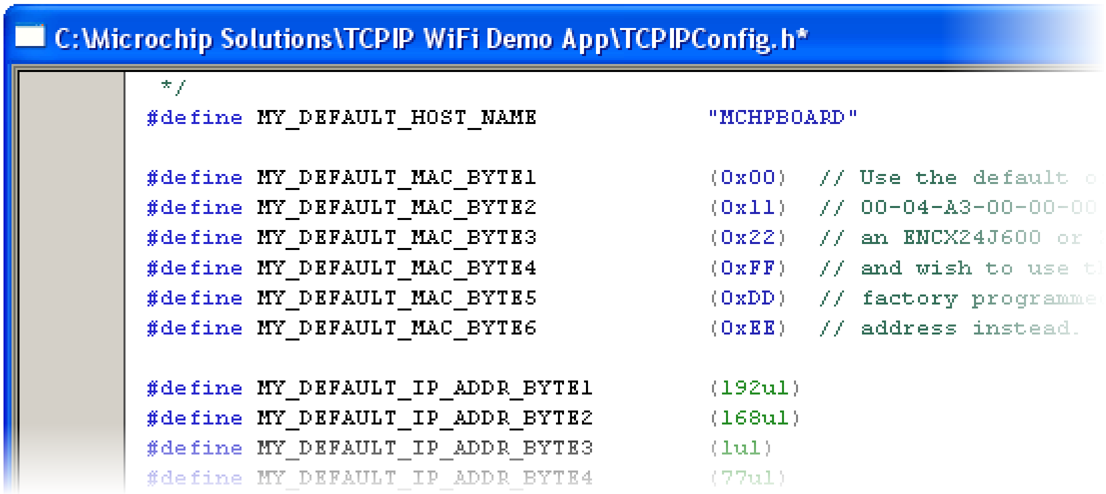
4.3.3 MAC Address

There are three different sources for the MAC address in the system. There is a built-in MAC address on the ZeroG module that is preprogrammed from the factory with the ZeroG OUI. The second source is from the programmed code image. The third source is from a value that is stored in the EEPROM.

At runtime, a data structure is created in RAM, which stores the valid MAC address (amongst other information) to be used for that session. The code will check to see if there is a valid data structure located in EEPROM. If something exists in EEPROM, then those values will be used, overriding what is programmed inside the chip and/or programmed in the code at compile time.

If no data structure exists in EEPROM, then the value that is stored in `TCPIPConfig.h` will be used. If the value in the source code is 00:04:A3:00:00:00, then this will indicate to the program that the value that has been preprogrammed in the ZeroG module should be used. Otherwise, the value that is placed into the `MY_DEFAULT_MAC_BYTEx` will be used. Additionally, in the case where no valid data structure exists in EEPROM, the new value will be programmed and stored to the EEPROM for future use.

In terms of priority, EEPROM has the highest priority. Therefore, if a value for the MAC address has already been programmed to the EEPROM, then no other value will override it. If you require the value to be changed, then you will have to erase the EEPROM.



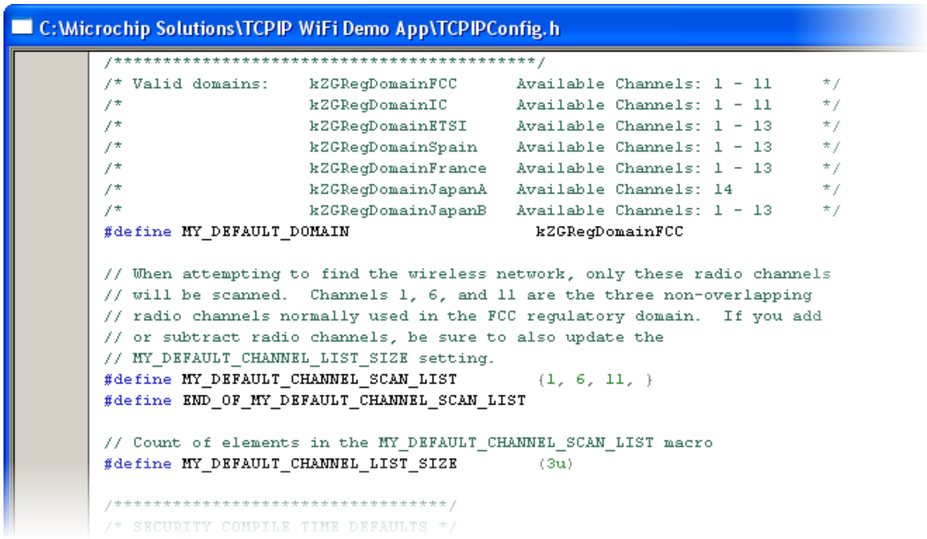
C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIPConfig.h*

```
 */
#define MY_DEFAULT_HOST_NAME           "MCHPBOARD"

#define MY_DEFAULT_MAC_BYTE1           (0x00)  // Use the default o
#define MY_DEFAULT_MAC_BYTE2           (0x11)  // 00-04-A3-00-00-00
#define MY_DEFAULT_MAC_BYTE3           (0x22)  // an ENCX24J600 or
#define MY_DEFAULT_MAC_BYTE4           (0xFF)  // and wish to use t
#define MY_DEFAULT_MAC_BYTE5           (0xDD)  // factory programme
#define MY_DEFAULT_MAC_BYTE6           (0xEE)  // address instead.

#define MY_DEFAULT_IP_ADDR_BYTE1       (192ul)
#define MY_DEFAULT_IP_ADDR_BYTE2       (168ul)
#define MY_DEFAULT_IP_ADDR_BYTE3       (1ul)
#define MY_DEFAULT_IP_ADDR_BYTE4       (77ul)
```

Figure 20: Changing MAC Address

4.3.4 Channel Configuration

The RF channel that is used can also be configured at compile-time. There are three interrelated options that help control which channels data is transmitted on. The first is the regulatory domain, as specified by `MY_DEFAULT_DOMAIN`. Different domains have different channel offerings, so this needs to match the intended country and channel. The second is the channel scan list (`MY_DEFAULT_CHANNEL_SCAN_LIST`). This is an array of channels that will be scanned for RF activity. Note that the more channels that are scanned, the longer time it will take to connect. The variable `MY_DEFAULT_CHANNEL_LIST_SIZE` must match the number of channels that are going to be scanned in `MY_DEFAULT_CHANNEL_SCAN_LIST`. In the following screenshot, you can see that the domain is set to FCC, and there are 3 total channels in the scan list (1,6,11).

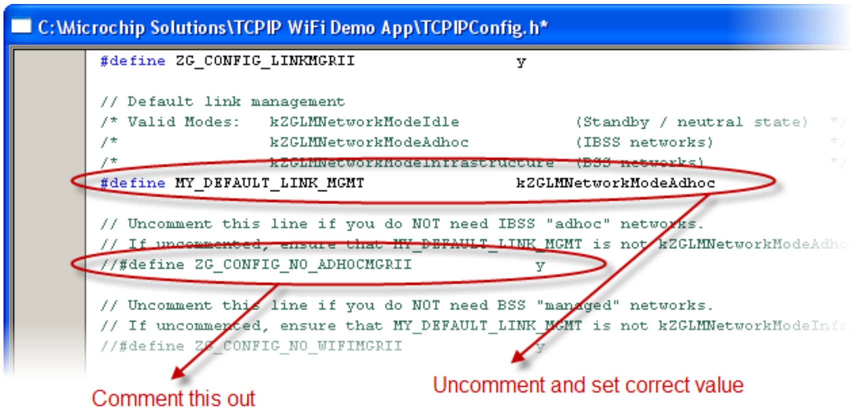


```
/*********************************************/
/* Valid domains:     kZGRegDomainFCC      Available Channels: 1 - 11     */
/*                    kZGRegDomainIC       Available Channels: 1 - 11     */
/*                    kZGRegDomainETSI     Available Channels: 1 - 13     */
/*                    kZGRegDomainSpain    Available Channels: 1 - 13     */
/*                    kZGRegDomainFrance   Available Channels: 1 - 13     */
/*                    kZGRegDomainJapanA   Available Channels: 14         */
/*                    kZGRegDomainJapanB   Available Channels: 1 - 13     */
#define MY_DEFAULT_DOMAIN                    kZGRegDomainFCC

// When attempting to find the wireless network, only these radio channels
// will be scanned.  Channels 1, 6, and 11 are the three non-overlapping
// radio channels normally used in the FCC regulatory domain.  If you add
// or subtract radio channels, be sure to also update the
// MY_DEFAULT_CHANNEL_LIST_SIZE setting.
#define MY_DEFAULT_CHANNEL_SCAN_LIST        {1, 6, 11, }
#define END_OF_MY_DEFAULT_CHANNEL_SCAN_LIST

// Count of elements in the MY_DEFAULT_CHANNEL_SCAN_LIST macro
#define MY_DEFAULT_CHANNEL_LIST_SIZE        (3u)

/************************************/
/* SECURITY COMPILE TIME DEFAULTS */
```

Figure 21: Channel Setup

4.3.5. Ad-hoc Network Configuration

By default, the demonstrations work out-of-the-box in infrastructure (BSS) mode. Changing to ad-hoc (IBSS) can be done at compile time as well. To do so, two code defines have to be setup correctly, as shown.



```
#define ZG_CONFIG_LINKMGRII                     y

// Default link management
/* Valid Modes:   kZGLMNetworkModeIdle             (Standby / neutral state)  */
/*                kZGLMNetworkModeAdhoc            (IBSS networks)            */
/*                kZGLMNetworkModeInfrastructure   (BSS networks)             */
#define MY_DEFAULT_LINK_MGMT                    kZGLMNetworkModeAdhoc

// Uncomment this line if you do NOT need IBSS "adhoc" networks.
// If uncommented, ensure that MY_DEFAULT_LINK_MGMT is not kZGLMNetworkModeAdho
//#define ZG_CONFIG_NO_ADHOCMGRII                 y

// Uncomment this line if you do NOT need BSS "managed" networks.
// If uncommented, ensure that MY_DEFAULT_LINK_MGMT is not kZGLMNetworkModeInfr
//#define ZG_CONFIG_NO_WIFIMGRII                  y
```

Figure 22: Source Code Ad-hoc Settings

4.3.6. Wireless Security

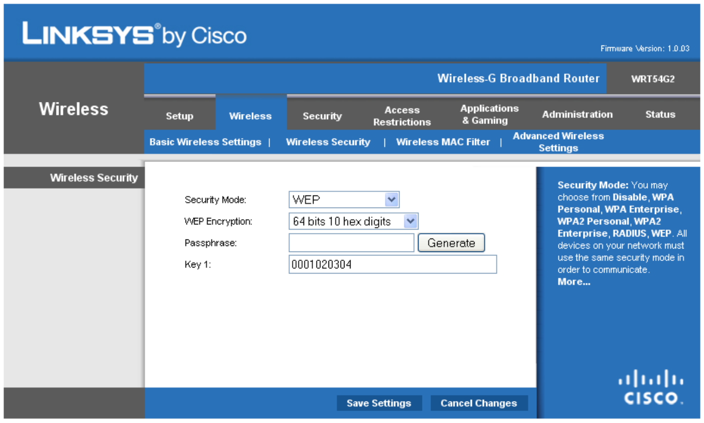
4.3.6.1 WEP

WEP security comes in two different forms, 64-bit WEP which uses a 40-bit key (WEP-40) and 128-bit WEP which uses a 104-bit key (WEP-104). In the most basic form, WEP keys are just hexadecimal values, 5 bytes for WEP-40 and 13 bytes for WEP-104. Some routers, like the Linksys WRT54G, try to increase the random nature of the WEP key by adding an additional layer that will convert an ASCII passphrase into a hexadecimal key. The ZeroG PICtail will require a hexadecimal key, no matter which way it is generated.

To enable WEP security between the development board and the access point, you will need to setup the access point for security, and program the development board with the correct WEP keys.

4.3.6.1.1 Setup Access Point for WEP

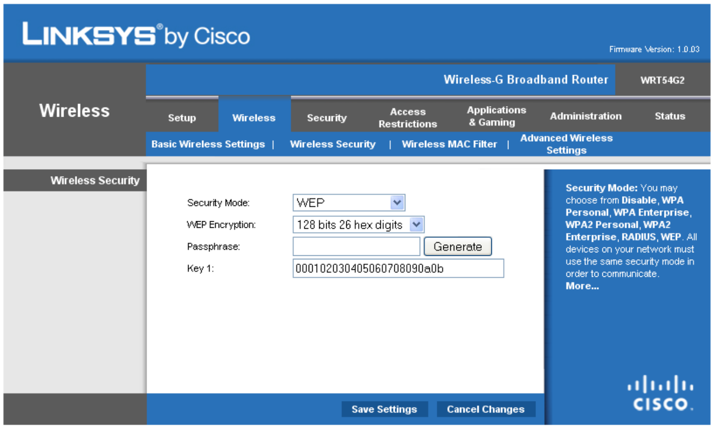
First, login to the router by following the instructions in the *Wireless Access Point Setup* section. You can access the wireless security options by clicking on the Wireless tab, then click on Wireless Security. To match the default code in the hardware, setup the security as shown.



Security Mode: WEP
WEP Encryption: 64 bits 10 hex digits
Passphrase:
Key 1: 0001020304

Security Mode: You may choose from **Disable, WPA Personal, WPA Enterprise, WPA2 Personal, WPA2 Enterprise, RADIUS, WEP**. All devices on your network must use the same security mode in order to communicate. **More...**

Figure 23: 64-bit WEP Security Setup



Security Mode: WEP
WEP Encryption: 128 bits 26 hex digits
Passphrase:
Key 1: 000102030405060708090a0b

Security Mode: You may choose from **Disable, WPA Personal, WPA Enterprise, WPA2 Personal, WPA2 Enterprise, RADIUS, WEP**. All devices on your network must use the same security mode in order to communicate. **More...**

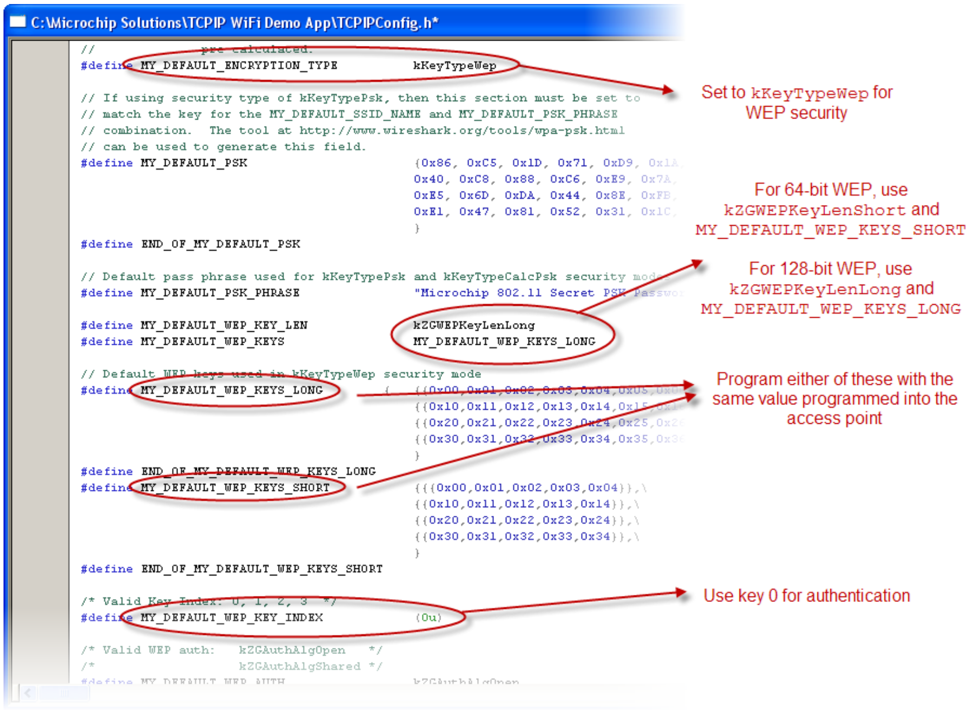
Figure 24: 128-bit WEP Security Setup

4.3.6.1.2 Setup Source Code for WEP

Once the access point is configured for WEP, you will now need to program the development board to use WEP as well. There are 4 different items that need to be configured for WEP:

- Set the encryption type to WEP
- Indicate whether you are using short (WEP-40) or long (WEP-104) keys
- Program the correct key in hexadecimal form
- The key index to use

All four of these items are located in `TCPIPConfig.h`, and should be setup similar to the following diagram:



C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIPConfig.h*

```
//              pre-calculated.
#define MY_DEFAULT_ENCRYPTION_TYPE          kKeyTypeWep

// If using security type of kKeyTypePsk, then this section must be set to
// match the key for the MY_DEFAULT_SSID_NAME and MY_DEFAULT_PSK_PHRASE
// combination.  The tool at http://www.wireshark.org/tools/wpa-psk.html
// can be used to generate this field.
#define MY_DEFAULT_PSK                      {0x86, 0xC5, 0x1D, 0x71, 0xD9, 0x1A,
                                             0x40, 0xC8, 0x88, 0xC6, 0xE9, 0x7A,
                                             0xE5, 0x6D, 0xDA, 0x44, 0x8E, 0xFB,
                                             0xE1, 0x47, 0x81, 0x52, 0x31, 0x1C,
                                             }
#define END_OF_MY_DEFAULT_PSK

// Default pass phrase used for kKeyTypePsk and kKeyTypeCalcPsk security mode
#define MY_DEFAULT_PSK_PHRASE               "Microchip 802.11 Secret PSK Passwo

#define MY_DEFAULT_WEP_KEY_LEN              kZGWEPKeyLenLong
#define MY_DEFAULT_WEP_KEYS                 MY_DEFAULT_WEP_KEYS_LONG

// Default WEP keys used in kKeyTypeWep security mode
#define MY_DEFAULT_WEP_KEYS_LONG        {   {{0x00,0x01,0x02,0x03,0x04,0x05,0x0
                                             {{0x10,0x11,0x12,0x13,0x14,0x15,0x1
                                             {{0x20,0x21,0x22,0x23,0x24,0x25,0x2
                                             {{0x30,0x31,0x32,0x33,0x34,0x35,0x3
                                             }
#define END_OF_MY_DEFAULT_WEP_KEYS_LONG
#define MY_DEFAULT_WEP_KEYS_SHORT            {{{0x00,0x01,0x02,0x03,0x04}},\
                                             {{0x10,0x11,0x12,0x13,0x14}},\
                                             {{0x20,0x21,0x22,0x23,0x24}},\
                                             {{0x30,0x31,0x32,0x33,0x34}},\
                                             }
#define END_OF_MY_DEFAULT_WEP_KEYS_SHORT

/* Valid Key Index: 0, 1, 2, 3  */
#define MY_DEFAULT_WEP_KEY_INDEX            (0u)

/* Valid WEP auth:   kZGAuthAlgOpen   */
/*                   kZGAuthAlgShared */
#define MY_DEFAULT_WEP_AUTH                 kZGAuthAlgOpen
```

Set to `kKeyTypeWep` for WEP security

For 64-bit WEP, use `kZGWEPKeyLenShort` and `MY_DEFAULT_WEP_KEYS_SHORT`

For 128-bit WEP, use `kZGWEPKeyLenLong` and `MY_DEFAULT_WEP_KEYS_LONG`

Program either of these with the same value programmed into the access point

Use key 0 for authentication

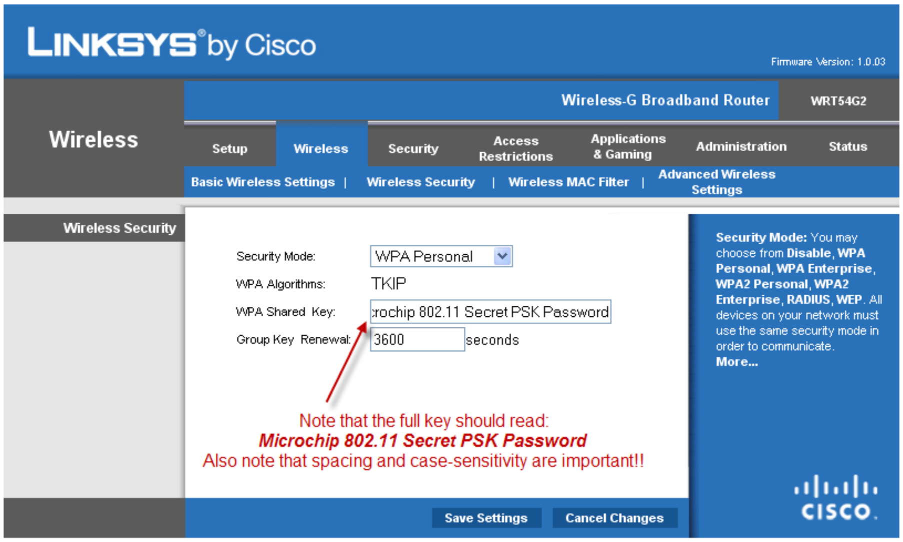
Figure 25: Source Code WEP Setup

4.3.6.2 WPA/WPA2

WPA and WPA2 are security modes that implement the 802.11i specification. They are more secure than WEP encrypted networks. It utilizes the SSID and user passphrase to generate the PSK. The ZeroG PICtail has two different ways of calculating the PSK for use in WPA/WPA2. The end user can program in the 32-byte hexadecimal key into the source code, or the ZeroG PICtail can calculate the 32-byte hexadecimal key internally. Note that if the ZeroG PICtail has to calculate the key, this will add an additional 30-45 seconds to the initial connection time as the chip calculates the value.

4.3.6.2.1 Setup Access Point for WPA/WPA2

First, login to the router by following the instructions in the *Wireless Access Point Setup* section. You can access the wireless security options by clicking on the Wireless tab, then click on Wireless Security. To match the default code in the hardware, setup the security as shown in the following three figures.



Security Mode: WPA Personal

WPA Algorithms: TKIP

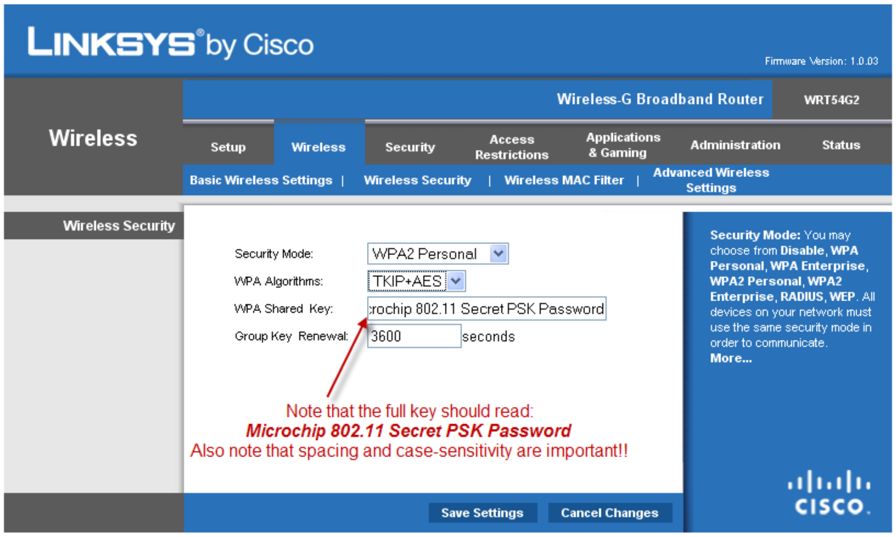
WPA Shared Key: :rochip 802.11 Secret PSK Password

Group Key Renewal: 3600 seconds

Note that the full key should read:
Microchip 802.11 Secret PSK Password
Also note that spacing and case-sensitivity are important!!

Security Mode: You may choose from **Disable**, **WPA Personal**, **WPA Enterprise**, **WPA2 Personal**, **WPA2 Enterprise**, **RADIUS**, **WEP**. All devices on your network must use the same security mode in order to communicate. **More...**

Figure 26: WPA Security Setup



Security Mode: WPA2 Personal

WPA Algorithms: TKIP+AES

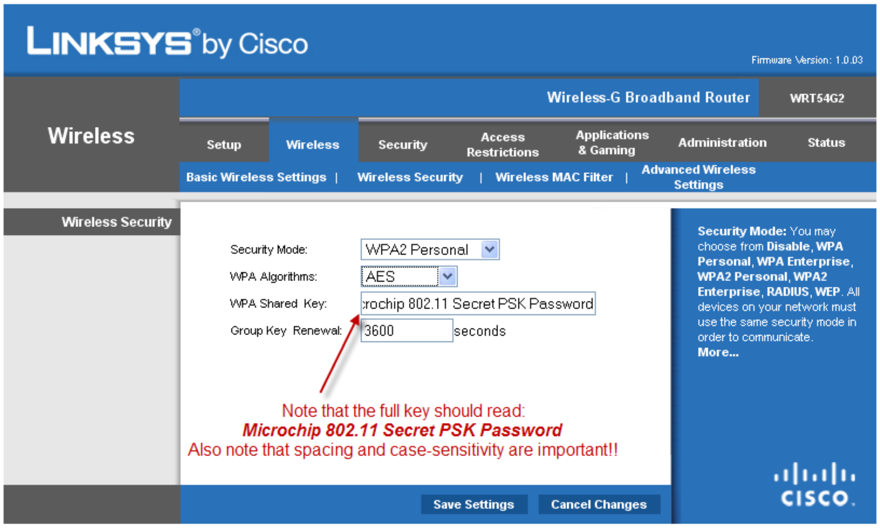
WPA Shared Key: :rochip 802.11 Secret PSK Password

Group Key Renewal: 3600 seconds

Note that the full key should read:
Microchip 802.11 Secret PSK Password
Also note that spacing and case-sensitivity are important!!

Security Mode: You may choose from **Disable**, **WPA Personal**, **WPA Enterprise**, **WPA2 Personal**, **WPA2 Enterprise**, **RADIUS**, **WEP**. All devices on your network must use the same security mode in order to communicate. **More...**

Figure 27: WPA2 TKIP/AES Security Setup



Security Mode: WPA2 Personal

WPA Algorithms: AES

WPA Shared Key: :rochip 802.11 Secret PSK Password

Group Key Renewal: 3600 seconds

Security Mode: You may choose from **Disable**, **WPA Personal**, **WPA Enterprise**, **WPA2 Personal**, **WPA2 Enterprise**, **RADIUS**, **WEP**. All devices on your network must use the same security mode in order to communicate. **More...**

Note that the full key should read:
Microchip 802.11 Secret PSK Password
Also note that spacing and case-sensitivity are important!!

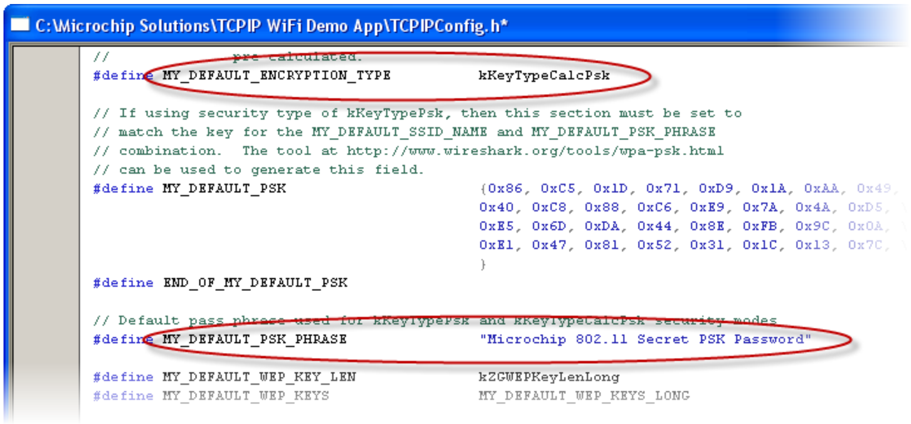
Figure 28: WPA2 AES Security Setup

4.3.6.2.2 Setup Source Code for WPA/WPA2

The ZeroG PICtail has two different modes of running WPA/WPA2 security. The most straight-forward approach is to supply the passphrase, and the PICtail will calculate the PSK based on the SSID and supplied passphrase. Due to the computationally intensive nature of this operation, this will take approximately 30 to 45 seconds to complete. Another approach is to provide the 32-byte PSK, and directly plug this value into the source code.   How to setup and use both approaches is described below.

4.3.6.2.3 On-the-fly PSK Calculation

There are only two pieces of information that are needed to enable the ZeroG PICtail to calculate the PSK and use that for the encryption process. Remember that the passphrase is case sensitive and that spacing does matter.

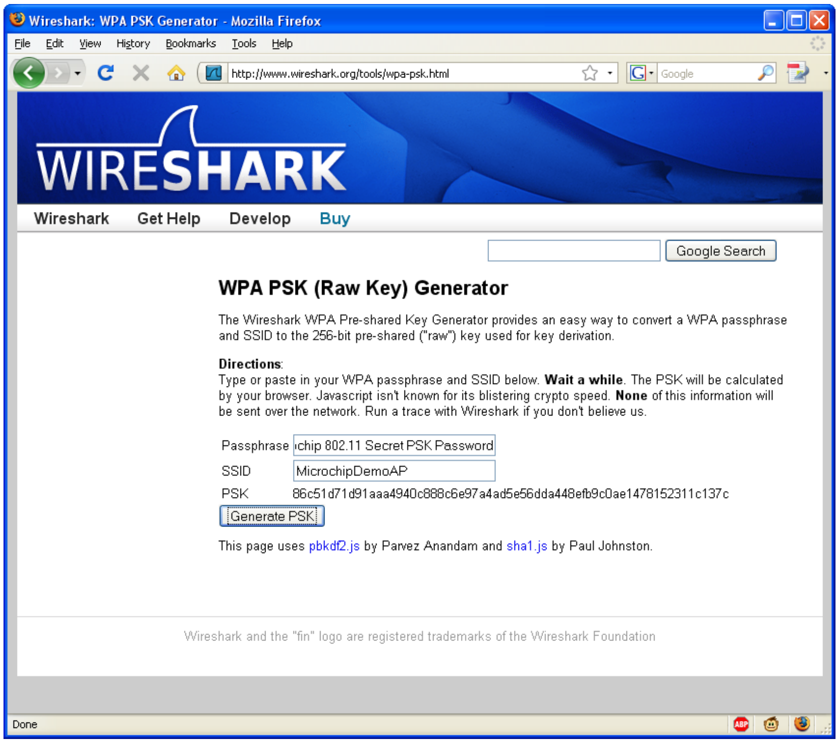

C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIPConfig.h*

```
//                   pre-calculated.
#define MY_DEFAULT_ENCRYPTION_TYPE          kKeyTypeCalcPsk

// If using security type of kKeyTypePsk, then this section must be set to
// match the key for the MY_DEFAULT_SSID_NAME and MY_DEFAULT_PSK_PHRASE
// combination.  The tool at http://www.wireshark.org/tools/wpa-psk.html
// can be used to generate this field.
#define MY_DEFAULT_PSK                      {0x86, 0xC5, 0x1D, 0x71, 0xD9, 0x1A, 0xAA, 0x49,
                                             0x40, 0xC8, 0x88, 0xC6, 0xE9, 0x7A, 0x4A, 0xD5,
                                             0xE5, 0x6D, 0xDA, 0x44, 0x8E, 0xFB, 0x9C, 0x0A,
                                             0xE1, 0x47, 0x81, 0x52, 0x31, 0x1C, 0x13, 0x7C,
                                             }

#define END_OF_MY_DEFAULT_PSK

// Default pass phrase used for kKeyTypePsk and kKeyTypeCalcPsk security modes
#define MY_DEFAULT_PSK_PHRASE               "Microchip 802.11 Secret PSK Password"

#define MY_DEFAULT_WEP_KEY_LEN              kZGWEPKeyLenLong
#define MY_DEFAULT_WEP_KEYS                 MY_DEFAULT_WEP_KEYS_LONG
```

Figure 29: Source Code Calculate PSK Setup

4.3.6.2.4 Pre-generated PSK

You also have the option to pre-generate the PSK and use the 32-byte PSK directly in the source code. One handy tool to generate the PSK can be found online at the Wireshark Foundation (link to website is supplied in the Appendix). The Wireshark website can generate the expected 32-byte PSK key with the SSID name and the passphrase. You can then use these values in the variable `MY_DEFAULT_PSK` in `TCPIPConfig.h`.


Wireshark: WPA PSK Generator - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.wireshark.org/tools/wpa-psk.html
Google
WIRESHARK
Wireshark
Get Help
Develop
Buy
Google Search

WPA PSK (Raw Key) Generator

The Wireshark WPA Pre-shared Key Generator provides an easy way to convert a WPA passphrase and SSID to the 256-bit pre-shared ("raw") key used for key derivation.

Directions:
Type or paste in your WPA passphrase and SSID below. **Wait a while**. The PSK will be calculated by your browser. Javascript isn't known for its blistering crypto speed. **None** of this information will be sent over the network. Run a trace with Wireshark if you don't believe us.

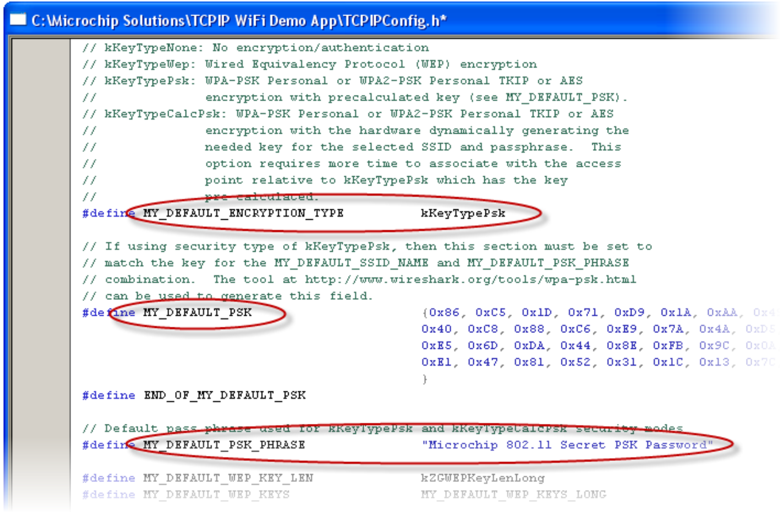
Passphrase: chip 802.11 Secret PSK Password
SSID: MicrochipDemoAP
PSK: 86c51d71d91aaa4940c888c6e97a4ad5e56dda448efb9c0ae1478152311c137c

This page uses pbkdf2.js by Parvez Anandam and sha1.js by Paul Johnston.

Wireshark and the "fin" logo are registered trademarks of the Wireshark Foundation

Figure 30: WPA PSK Generation

You'll notice that for the default setup, the generated PSK value from the Wireshark website is identical to the value in `MY_DEFAULT_PSK`. Setup the source code for using the pre-generated PSK. Note that this method is only useful for engineering purposes only. In a real application, this method is not usable, and the on-the-fly calculation is the more generally accepted method for PSK calculation.

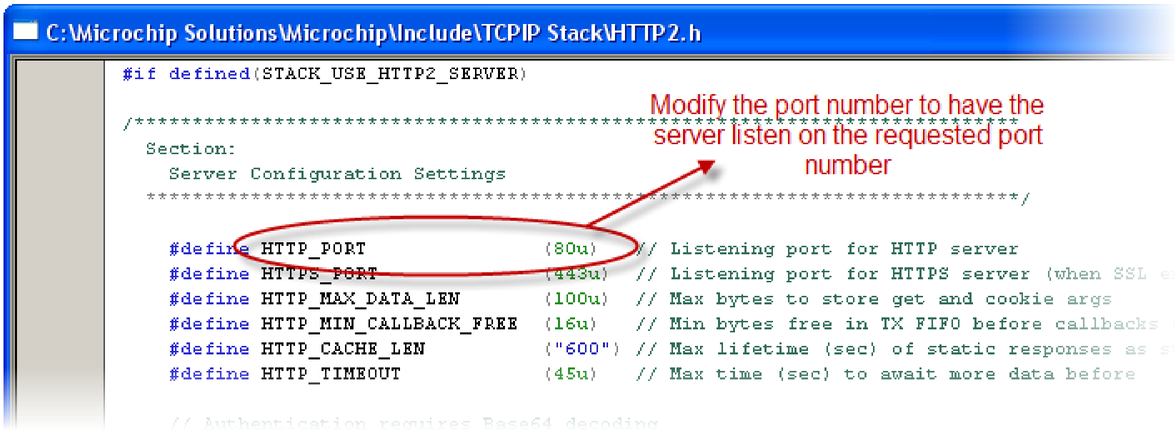


```
// kKeyTypeNone: No encryption/authentication
// kKeyTypeWep: Wired Equivalency Protocol (WEP) encryption
// kKeyTypePsk: WPA-PSK Personal or WPA2-PSK Personal TKIP or AES
//              encryption with precalculated key (see MY_DEFAULT_PSK).
// kKeyTypeCalcPsk: WPA-PSK Personal or WPA2-PSK Personal TKIP or AES
//              encryption with the hardware dynamically generating the
//              needed key for the selected SSID and passphrase.  This
//              option requires more time to associate with the access
//              point relative to kKeyTypePsk which has the key
//              pre-calculated.
#define MY_DEFAULT_ENCRYPTION_TYPE          kKeyTypePsk

// If using security type of kKeyTypePsk, then this section must be set to
// match the key for the MY_DEFAULT_SSID_NAME and MY_DEFAULT_PSK_PHRASE
// combination.  The tool at http://www.wireshark.org/tools/wpa-psk.html
// can be used to generate this field.
#define MY_DEFAULT_PSK                      {0x86, 0xC5, 0x1D, 0x71, 0xD9, 0x1A, 0xAA, 0x4
                                             0x40, 0xC8, 0x88, 0xC6, 0xE9, 0x7A, 0x4A, 0xD5
                                             0xE5, 0x6D, 0xDA, 0x44, 0x8E, 0xFB, 0x9C, 0x0A
                                             0xE1, 0x47, 0x81, 0x52, 0x31, 0x1C, 0x13, 0x7C
                                             }
#define END_OF_MY_DEFAULT_PSK

// Default pass phrase used for kKeyTypePsk and kKeyTypeCalcPsk security modes
#define MY_DEFAULT_PSK_PHRASE               "Microchip 802.11 Secret PSK Password"

#define MY_DEFAULT_WEP_KEY_LEN              kZGWEPKeyLenLong
#define MY_DEFAULT_WEP_KEYS                 MY_DEFAULT_WEP_KEYS_LONG
```

Figure 31: Source Code Pre-generated PSK Setup

4.3.7 Changing Listening Port

By default, most World Wide Web HTTP traffic uses port 80 for communication (e.g. http://google.com and http://google.com:80 would by synonymous). It is possible to change the code to listen for traffic on a different port by modifying the `HTTP_PORT` define in `HTTP2.h` as shown:


C:\Microchip Solutions\Microchip\Include\TCPIP Stack\HTTP2.h

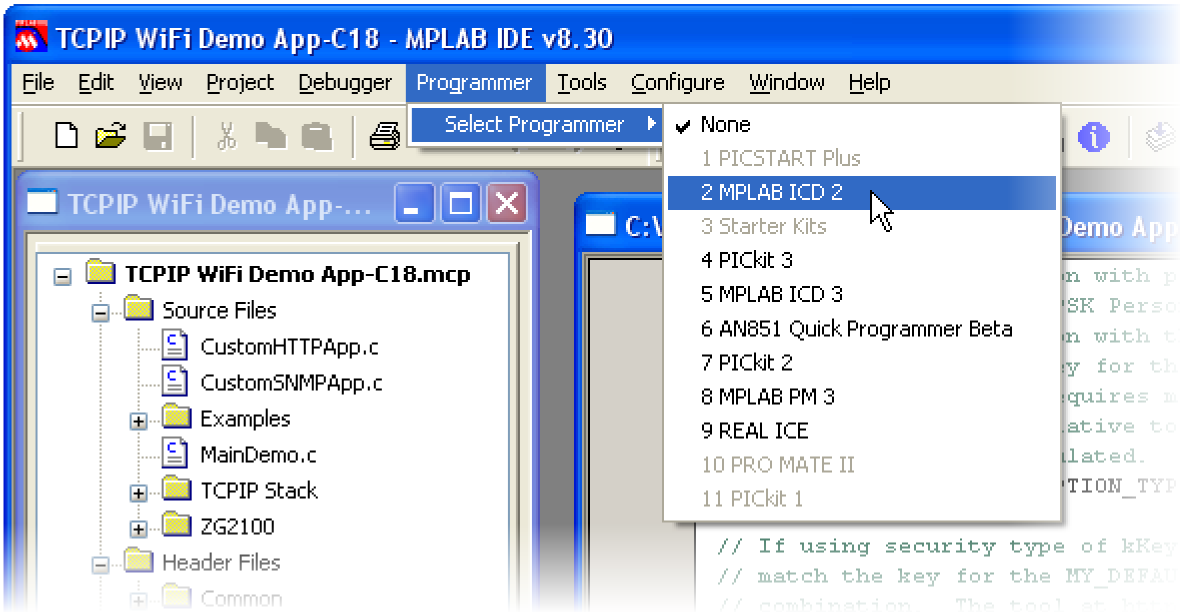
```
#if defined(STACK_USE_HTTP2_SERVER)

/****************************************************************************
  Section:
    Server Configuration Settings
  ***************************************************************************/

    #define HTTP_PORT               (80u)   // Listening port for HTTP server
    #define HTTPS_PORT              (443u)  // Listening port for HTTPS server (when SSL e
    #define HTTP_MAX_DATA_LEN       (100u)  // Max bytes to store get and cookie args
    #define HTTP_MIN_CALLBACK_FREE  (16u)   // Min bytes free in TX FIFO before callbacks
    #define HTTP_CACHE_LEN          ("600") // Max lifetime (sec) of static responses as s
    #define HTTP_TIMEOUT            (45u)   // Max time (sec) to await more data before

    // Authentication requires Base64 decoding
```

Modify the port number to have the server listen on the requested port number

Figure 32: Changing Listening Port

4.4 Compiling and Downloading Images

Start by selecting the appropriate ICD you have in your setup (either ICD2 or ICD3). The tool can be selected under the Programmer menu.



```
// If using security type of kKey
// match the key for the MY_DEFAU
```

Figure 33: Selecting ICD

Next, compile all the code by issuing a "Build All" command under the Project menu.

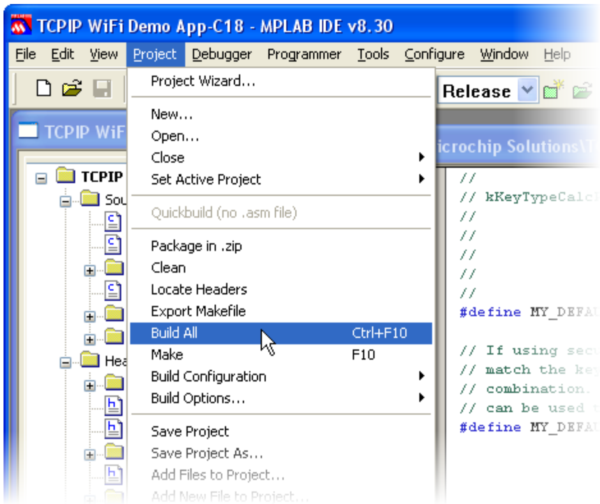
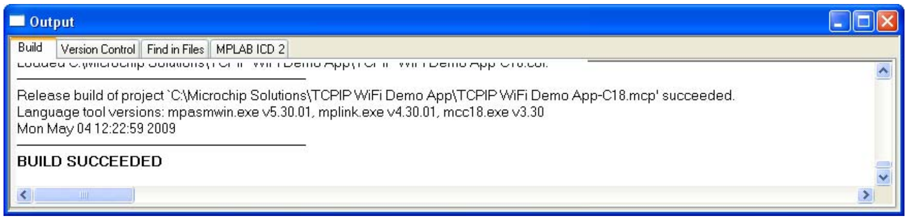


Figure 34: Build All

By monitoring the Output window in MPLAB, you will be able to see any compiler errors or warnings, and when everything has compiled and build correctly, you should see something similar to the following output:



```
Loaded C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIP WiFi Demo App-C18.cof.
----------------------------------------------------------------------
Release build of project `C:\Microchip Solutions\TCPIP WiFi Demo App\TCPIP WiFi Demo App-C18.mcp' succeeded.
Language tool versions: mpasmwin.exe v5.30.01, mplink.exe v4.30.01, mcc18.exe v3.30
Mon May 04 12:22:59 2009
----------------------------------------------------------------------
BUILD SUCCEEDED
```

Figure 35: Build Succeeded

After the build is successful, the next step is to program the PIC microcontroller with the build image. To do so, select Program from the Programmer menu.

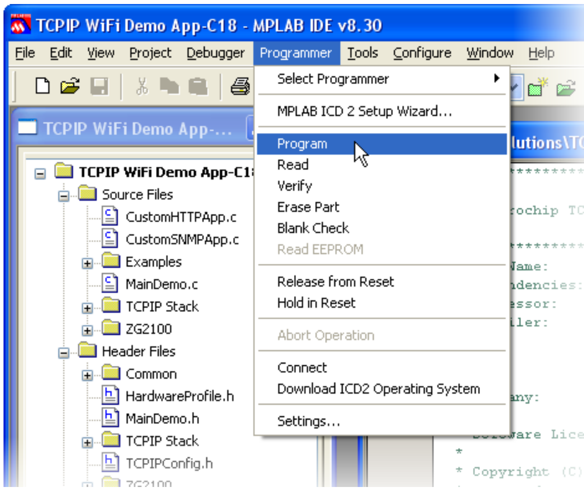
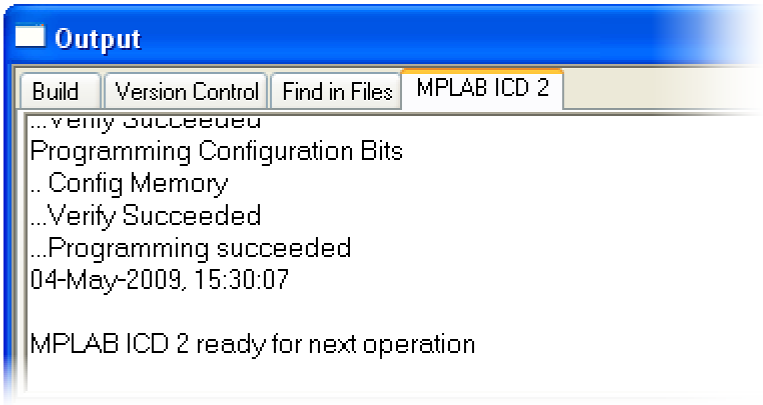


Figure 36: Programming MCU

Programming takes a good while longer using an ICD2 versus an ICD3. Once programming is completed, you should see the following:



```
...Verify Succeeded
Programming Configuration Bits
.. Config Memory
...Verify Succeeded
...Programming succeeded
04-May-2009, 15:30:07

MPLAB ICD 2 ready for next operation
```

Figure 37: Successful Programming

For the ICD2, as the last step, the PIC microcontroller needs to be taken out of reset. To do so, select Release From Reset from the Programmer menu, as shown here:

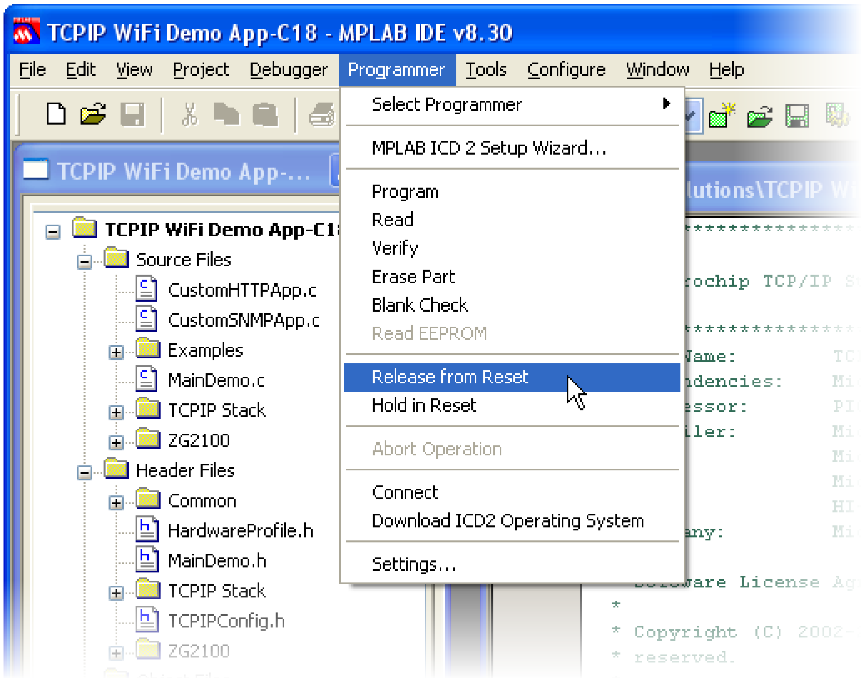


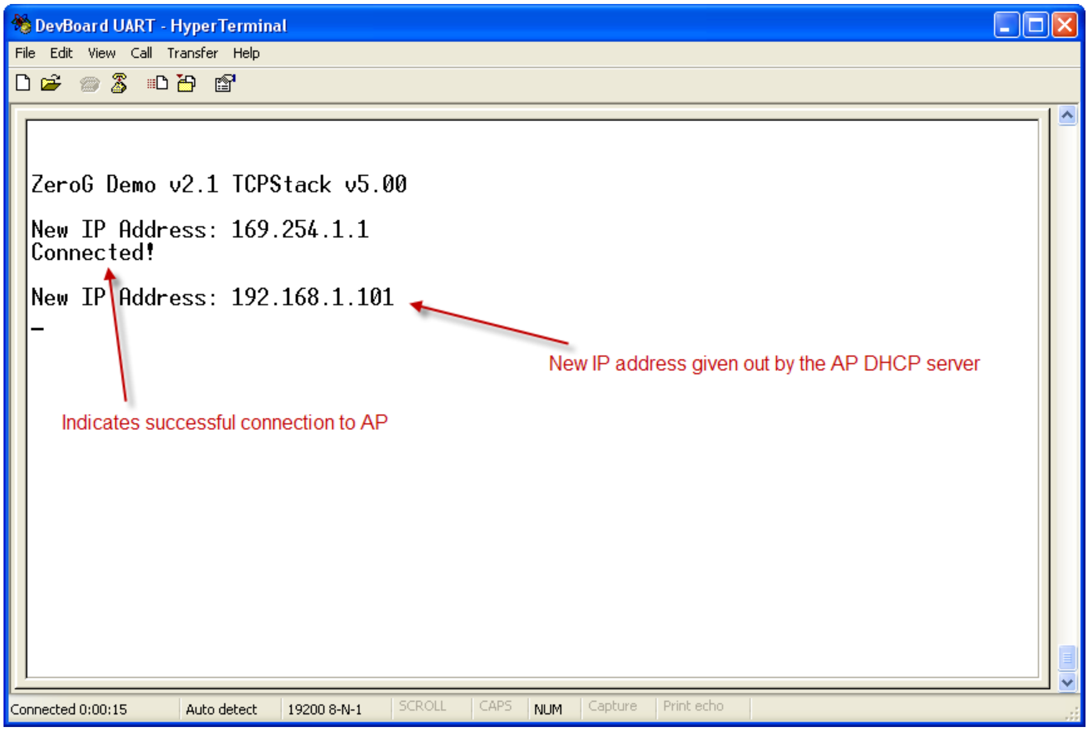
Figure 38: Release from Reset

4.5 Running the TCPIP Wi-Fi Demo

If this is not the first time any TCP/IP demo has been run, you should be aware that the code is crafted in such a way to use any previously stored values in the EEPROM to help speed up load times. Because of this, you may see situations where you have made changes in the code that are not reflected in the running of the demo (e.g. you changed the SSID name, but don't see it being used). In this case, you will need to erase the EEPROM. Instructions can be found in the section *Erasing EEPROM.* Otherwise, if this is the first time the demo has been run, you can continue on to the following steps.

Now that all the hardware and software has been setup, it's now time to see the sample Wi-Fi web server demonstration in action! If this is the first time the demonstration has been run, the web server code will need to be downloaded into the EEPROM of the development board. To do so, you will need to use your browser to navigate to a special page that will allow you to upload the image file.

Once the development board has established connection to the access point (or, it has created the network in ad-hoc mode), the LCD panel should display the IP address that is being used. Alternatively, the UART output should display something similar that will give you the same information (sample screenshots for both infrastructure and ad-hoc are shown below):



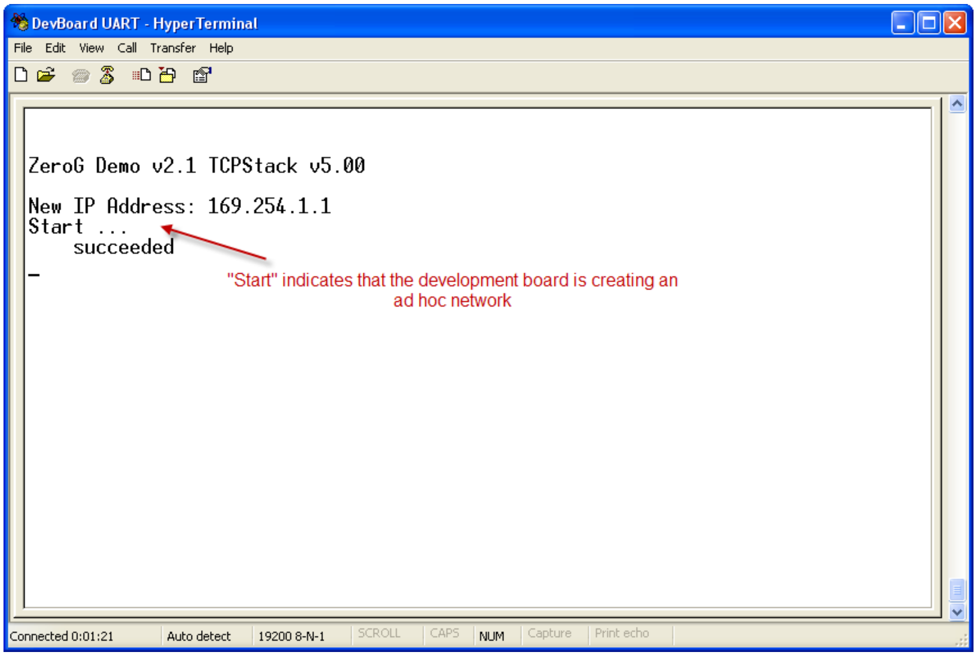
DevBoard UART - HyperTerminal
File Edit View Call Transfer Help

```
ZeroG Demo v2.1 TCPStack v5.00

New IP Address: 169.254.1.1
Connected!

New IP Address: 192.168.1.101
_
```

New IP address given out by the AP DHCP server

Indicates successful connection to AP

Figure 39: UART Infrastructure Output


DevBoard UART - HyperTerminal
File Edit View Call Transfer Help

```
ZeroG Demo v2.1 TCPStack v5.00

New IP Address: 169.254.1.1
Start ...
    succeeded
_
```

"Start" indicates that the development board is creating an ad hoc network

Figure 40: UART Ad-hoc Output

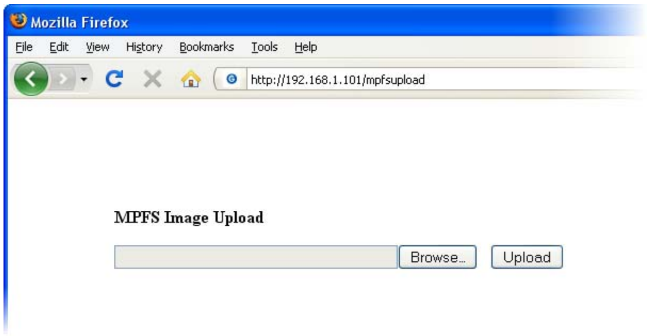
Next, we will need to go to the upload page of the development board. This will take the form of

http://xxx.xxx.xxx.xxx/mpfsupload

Note that if you changed the listening port number (as described in the section *Changing Listening Port),* you will need to provide the new port number as well:

http://xxx.xxx.xxx.xxx:12345/mpfsupload (this assumes port 12345 as an example)

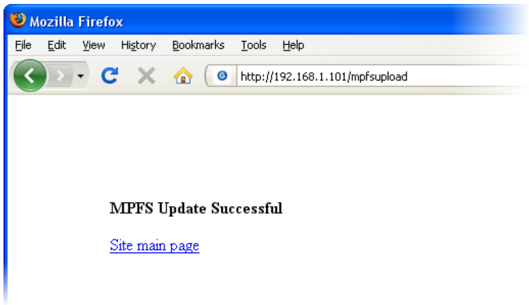
Using *Figure 41* as an example (and assuming the listening port was not changed), press the Browse button and upload the file `MPFSImg2.bin`. This file is located in the root directory of the TCPIP WiFi Demo App directory. If you installed all the software to the default location, then the directory would be `C:\Microchip Solutions\TCPIP WiFi Demo App\`. Open this file, and press the Upload button.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://192.168.1.101/mpfsupload

MPFS Image Upload

Figure 41: Web Server Image Upload

After the upload is finished, you should see this page. Click the Site main page link to be taken to the main page of the web server.

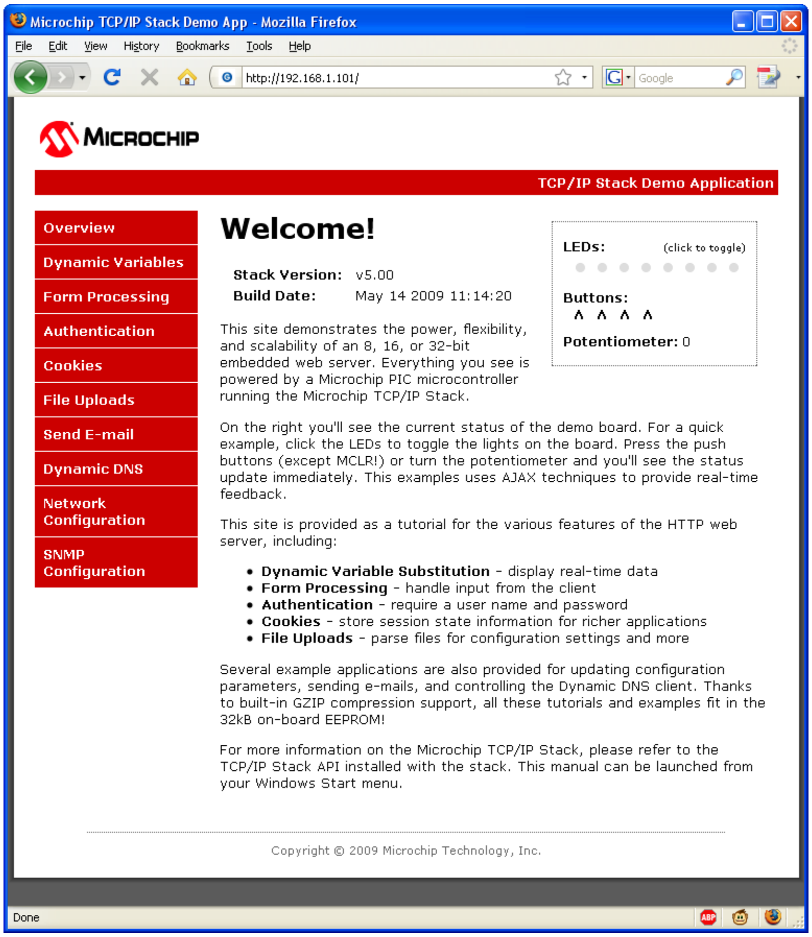

Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://192.168.1.101/mpfsupload

MPFS Update Successful

Site main page

Figure 42: Web Server Image Upload Successful

The TCP/IP Wi-Fi demo application is a great starting point for understanding the features and applications that are available with the Microchip TCP/IP stack, and how they can all be used on wireless medium. From the main page, you can interact with the development board hardware to toggle LEDs, push buttons, and change potentiometer values. Other pages of the demo allow you to send email, upload files, and change network configuration values.



Welcome!

Stack Version: v5.00
Build Date: May 14 2009 11:14:20

This site demonstrates the power, flexibility, and scalability of an 8, 16, or 32-bit embedded web server. Everything you see is powered by a Microchip PIC microcontroller running the Microchip TCP/IP Stack.

LEDs: (click to toggle)
● ● ● ● ● ● ● ●

Buttons:
∧ ∧ ∧ ∧

Potentiometer: 0

On the right you'll see the current status of the demo board. For a quick example, click the LEDs to toggle the lights on the board. Press the push buttons (except MCLR!) or turn the potentiometer and you'll see the status update immediately. This examples uses AJAX techniques to provide real-time feedback.

This site is provided as a tutorial for the various features of the HTTP web server, including:

- **Dynamic Variable Substitution** - display real-time data
- **Form Processing** - handle input from the client
- **Authentication** - require a user name and password
- **Cookies** - store session state information for richer applications
- **File Uploads** - parse files for configuration settings and more

Several example applications are also provided for updating configuration parameters, sending e-mails, and controlling the Dynamic DNS client. Thanks to built-in GZIP compression support, all these tutorials and examples fit in the 32kB on-board EEPROM!

For more information on the Microchip TCP/IP Stack, please refer to the TCP/IP Stack API installed with the stack. This manual can be launched from your Windows Start menu.

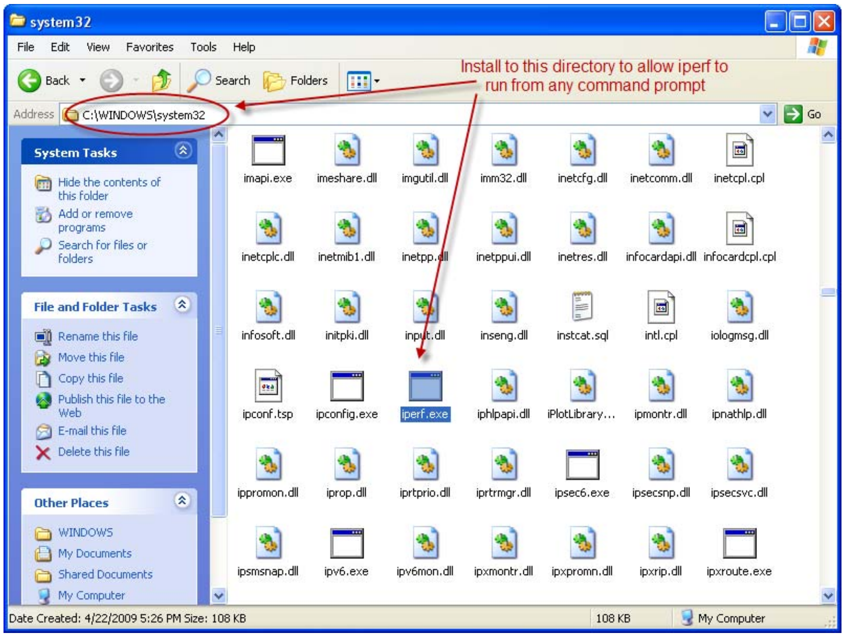
Copyright © 2009 Microchip Technology, Inc.

Figure 43: TCP/IP WiFi Demo Application

4.6 Running the WiFi Iperf Demo

iperf is a commonly used networking test tool that allows you to test throughput for network performance measurements. iperf operates with a server and a client. The client will send data to the server at a specified rate, and bandwidth will be calculated from the server side (even though both the client and server will product throughput numbers, the numbers that are in the server window are the most accurate). iperf requires that the serial UART port be connected to a PC (you will be issuing commands through the HyperTerminal session). To run the iperf demonstration, you will need to compile in the correct project. See the section *Opening Existing Projects,* but instead use the projects that are located in: `C:\Microchip Solutions\WiFi Iperf App\`.

After compiling and downloading the code (see the section *Compiling and Downloading Images*), you will need to get and install the iperf application on your PC. See the *Appendix* for a link on where to get iperf from.

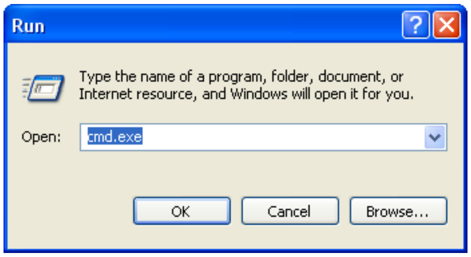


Install to this directory to allow iperf to run from any command prompt

Figure 44: Installing iperf.exe

For ease of use, download and install the iperf.exe binary file in `C:\Windows\system32` to make it easier to use from any command prompt in the future.

After installing iperf, you will need to open a command prompt on your PC. You can either do this by running cmd.exe from the Run dialog window, or by selecting it from your Programs menu.



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: cmd.exe

Figure 45: Command Prompt

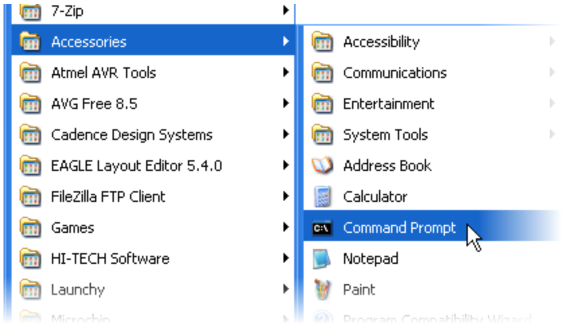
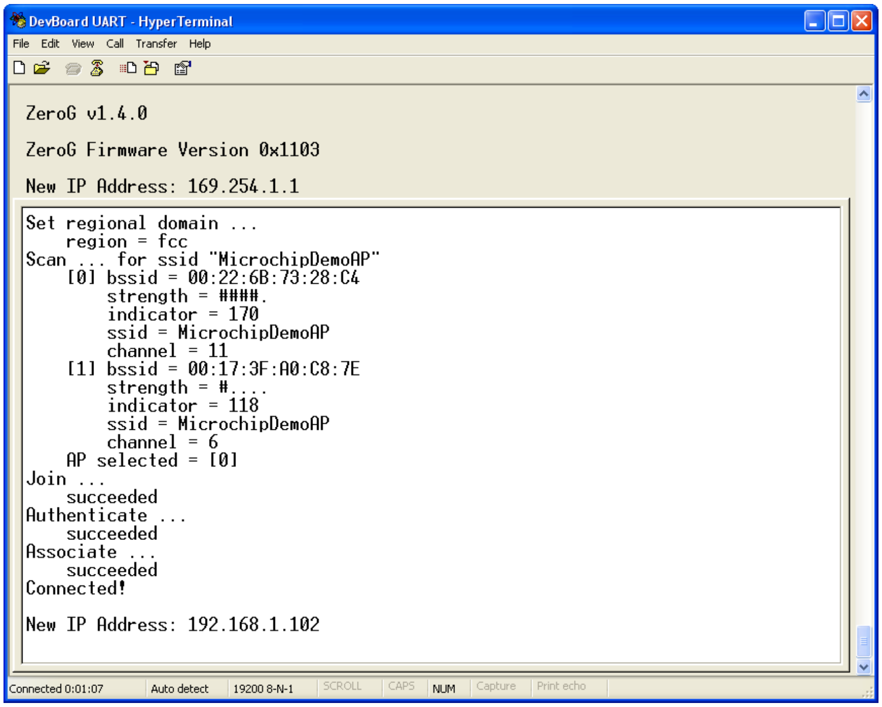


Figure 46: Prompt from Programs Menu

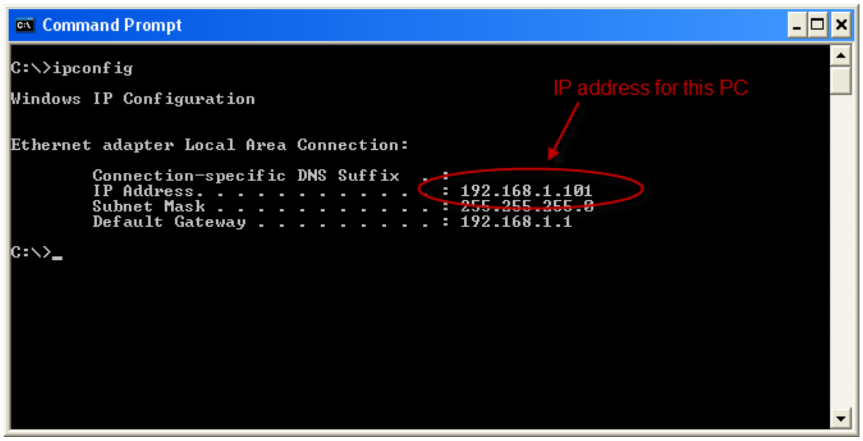
In order to run iperf, you will need to know the IP address of both the development board and the PC (server and client). The IP address of the development board can be seen on the LCD display or on the UART serial output, as shown below:



```
ZeroG v1.4.0

ZeroG Firmware Version 0x1103

New IP Address: 169.254.1.1
Set regional domain ...
    region = fcc
Scan ... for ssid "MicrochipDemoAP"
    [0] bssid = 00:22:6B:73:28:C4
        strength = ####.
        indicator = 170
        ssid = MicrochipDemoAP
        channel = 11
    [1] bssid = 00:17:3F:A0:C8:7E
        strength = #....
        indicator = 118
        ssid = MicrochipDemoAP
        channel = 6
    AP selected = [0]
Join ...
    succeeded
Authenticate ...
    succeeded
Associate ...
    succeeded
Connected!

New IP Address: 192.168.1.102
```

Figure 47: iperf Serial Output

To get the IP address of the PC, an easy way to do so is to run the command `ipconfig` from the command prompt:

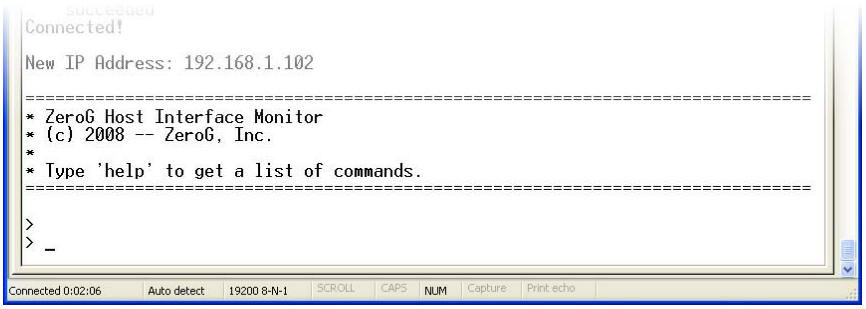


Command Prompt

```
C:\>ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.1.101
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1

C:\>_
```

Figure 48: Using ipconfig to get PC IP Address

In the HyperTerminal window, hit the Enter key. Once you see the following, the system is ready to run iperf:



```
   succeeded
Connected!

New IP Address: 192.168.1.102

================================================================================
* ZeroG Host Interface Monitor
* (c) 2008 -- ZeroG, Inc.
*
* Type 'help' to get a list of commands.
================================================================================

>
> _
```

Figure 49: HyperTerminal Window Ready

To test the ZeroG PICtail transmit performance, we need to setup the PC (command prompt) to be the server, and setup the PICtail (HyperTerminal session) to be the client. Below is a typical server and client command, along with what each options means:

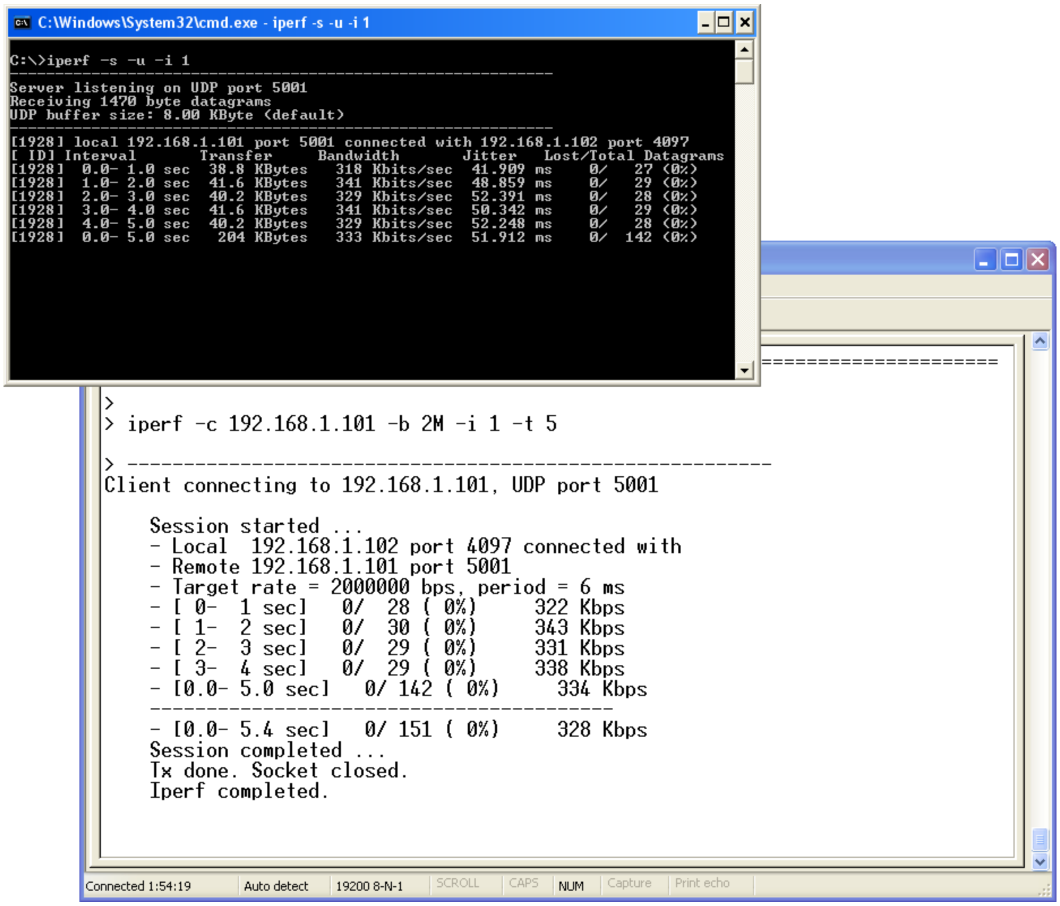
`iperf –s –u –i <seconds>`

`-s`	Indicates this is the server
`-u`	Sends UDP datagrams
`-i`	Indicates how often the screen will update with status

`iperf –c <ip_addr> -b <bw> -i <seconds> -t <seconds>`

`-c <ip_addr>`	Indicates that this is the client and the `ip_addr` is the server to talk to
`-b <bw>`	Specifies the amount of data to try and pass through as bandwidth
`-I <seconds>`	Indicates how often the screen with update with status
`-t <seconds>`	Indicates how long to run the test for

Below is a sample screen capture of what a typical iperf run will look like for testing transmit performance. Note that you must start the server connection first!

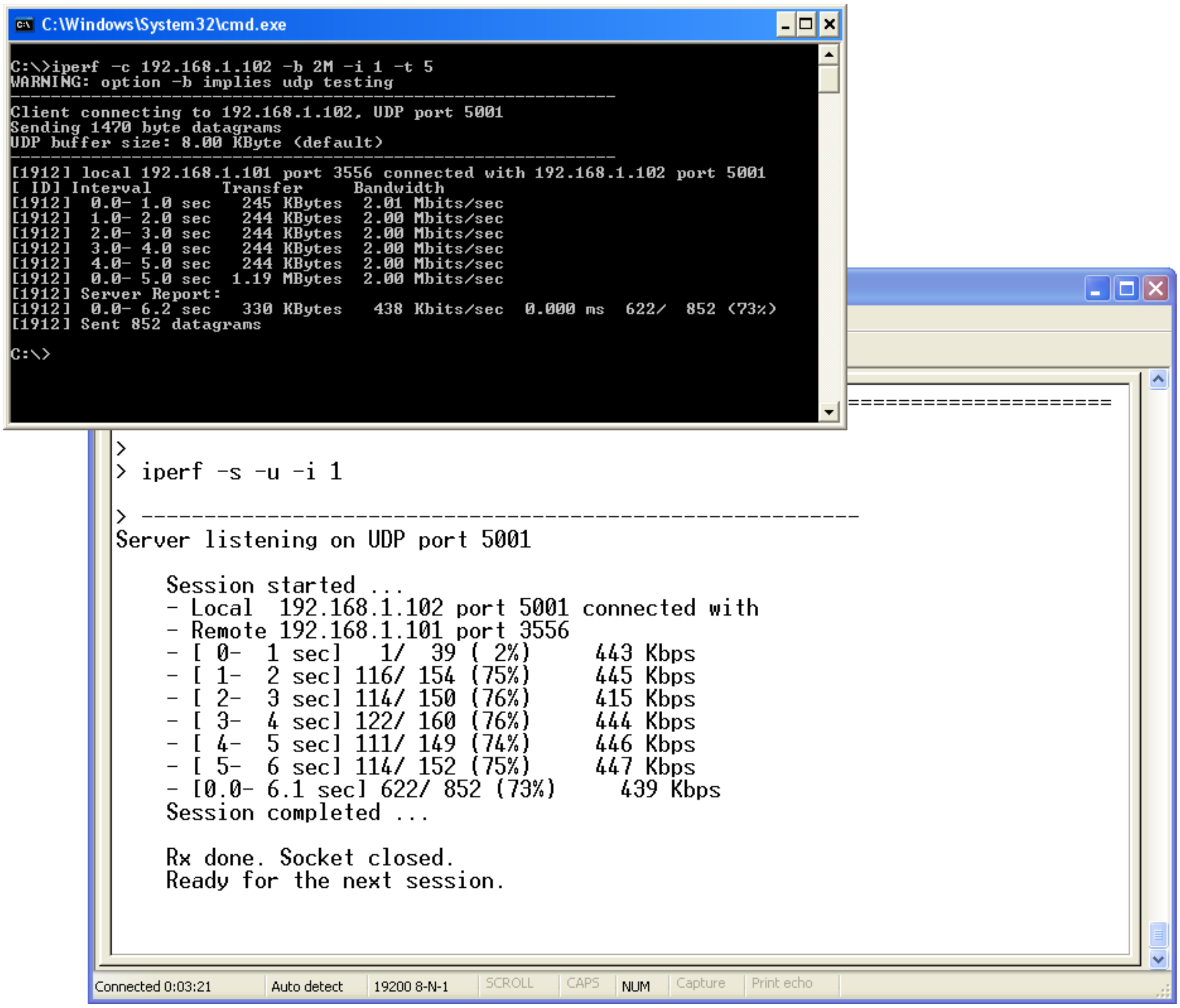


```
C:\>iperf -s -u -i 1
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
[1928] local 192.168.1.101 port 5001 connected with 192.168.1.102 port 4097
[ ID] Interval       Transfer     Bandwidth       Jitter   Lost/Total Datagrams
[1928]  0.0- 1.0 sec  38.8 KBytes   318 Kbits/sec  41.909 ms    0/   27 (0%)
[1928]  1.0- 2.0 sec  41.6 KBytes   341 Kbits/sec  48.859 ms    0/   29 (0%)
[1928]  2.0- 3.0 sec  40.2 KBytes   329 Kbits/sec  52.391 ms    0/   28 (0%)
[1928]  3.0- 4.0 sec  41.6 KBytes   341 Kbits/sec  50.342 ms    0/   29 (0%)
[1928]  4.0- 5.0 sec  40.2 KBytes   329 Kbits/sec  52.248 ms    0/   28 (0%)
[1928]  0.0- 5.0 sec   204 KBytes   333 Kbits/sec  51.912 ms    0/  142 (0%)
```

```
======================
>
> iperf -c 192.168.1.101 -b 2M -i 1 -t 5

> ---------------------------------------------------------
Client connecting to 192.168.1.101, UDP port 5001

    Session started ...
    - Local  192.168.1.102 port 4097 connected with
    - Remote 192.168.1.101 port 5001
    - Target rate = 2000000 bps, period = 6 ms
    - [ 0-  1 sec]   0/  28 ( 0%)     322 Kbps
    - [ 1-  2 sec]   0/  30 ( 0%)     343 Kbps
    - [ 2-  3 sec]   0/  29 ( 0%)     331 Kbps
    - [ 3-  4 sec]   0/  29 ( 0%)     338 Kbps
    - [0.0- 5.0 sec]   0/ 142 ( 0%)     334 Kbps
    ----------------------------------------
    - [0.0- 5.4 sec]   0/ 151 ( 0%)     328 Kbps
    Session completed ...
    Tx done. Socket closed.
    Iperf completed.
```

Figure 50: iperf Transmit Performance

Below is a screen capture of the development board receiver performance. Note that the server and client have reversed rolls (server is now the HyperTerminal session and client is the PC command prompt window). Also note that the IP address has changed to match the server's address.



```
C:\>iperf -c 192.168.1.102 -b 2M -i 1 -t 5
WARNING: option -b implies udp testing
------------------------------------------------------------
Client connecting to 192.168.1.102, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size: 8.00 KByte (default)
------------------------------------------------------------
[1912] local 192.168.1.101 port 3556 connected with 192.168.1.102 port 5001
[ ID] Interval       Transfer     Bandwidth
[1912]  0.0- 1.0 sec   245 KBytes  2.01 Mbits/sec
[1912]  1.0- 2.0 sec   244 KBytes  2.00 Mbits/sec
[1912]  2.0- 3.0 sec   244 KBytes  2.00 Mbits/sec
[1912]  3.0- 4.0 sec   244 KBytes  2.00 Mbits/sec
[1912]  4.0- 5.0 sec   244 KBytes  2.00 Mbits/sec
[1912]  0.0- 5.0 sec  1.19 MBytes  2.00 Mbits/sec
[1912] Server Report:
[1912]  0.0- 6.2 sec   330 KBytes   438 Kbits/sec  0.000 ms  622/  852 (73%)
[1912] Sent 852 datagrams

C:\>
```

```
======================
>
> iperf -s -u -i 1

> --------------------------------------------------------
Server listening on UDP port 5001

    Session started ...
    - Local  192.168.1.102 port 5001 connected with
    - Remote 192.168.1.101 port 3556
    - [ 0-  1 sec]   1/  39 ( 2%)     443 Kbps
    - [ 1-  2 sec] 116/ 154 (75%)     445 Kbps
    - [ 2-  3 sec] 114/ 150 (76%)     415 Kbps
    - [ 3-  4 sec] 122/ 160 (76%)     444 Kbps
    - [ 4-  5 sec] 111/ 149 (74%)     446 Kbps
    - [ 5-  6 sec] 114/ 152 (75%)     447 Kbps
    - [0.0- 6.1 sec] 622/ 852 (73%)     439 Kbps
    Session completed ...

    Rx done. Socket closed.
    Ready for the next session.
```

Figure 51: iperf Receive Performance

5. Microchip Development Board Specifics

5.1 PICDEM.NET 2 Usage

If you have a PICDEM.NET 2 board, please follow the instructions here. If you are using an Explorer16 board, please skip this section. Please note the J1 connector on the PICtail.

Table 3: PICDEM.net 2 PICtail Pin Description

Function	I/O	Pin	Description
CSN	I	J1-24/RC2	SPI Chip Select (asserted low)
SCK	I	J1-12/RC3	SPI Clock
SDO	O	J1-10/RC4	SPI Data Out from ZG2100M
SDI	I	J1-8/RC5	SPI Data In to ZG2100M
INT_NX	O	J1-27	Interrupt signal from ZG2100M (asserted low)
RST_N	I	J1-25/RB1	Master reset (asserted low)
CE_N	I	J1-23/RB2	ZG2100M disable (asserted low)
VDD	I	J1-26	5V power input

5.2 Explorer 16 Usage

If you have an Explorer16 board, please follow the instructions here. If you are using a PICDEM.NET 2 board, please skip this section. Please note the male connector J2 on the PICtail is female connector J5 on the Explorer 16.

Table 4: Explorer 16 PICtail Pin Description

Function	I/O	Pin	Description
CSN	I	J2-1/RB2	SPI Chip Select (asserted low)
SCK	I	J2-3/RF6/SCK1	SPI Clock
SDO	O	J2-5/RF7/SDI1_E	SPI Data Out from ZG2100M
SDI	I	J2-7/RF8/SDO1_E	SPI Data In to ZG2100M
INT_NX	O	J2-18/RE8/INT1	Interrupt signal from ZG2100M (asserted low)
RST_N	I	J2-28/RF0	Master reset (asserted low)
CE_N	I	J2-30/RF1	ZG2100M disable (asserted low)
VDD	I	J2-21 & J2-22	3.3V power input

5.3 Erasing EEPROM

While doing debug, if you notice that settings in code (especially related to SSID name, MAC address and the like) are not taking effect, then you may need to erase the EEPROM. The value in EEPROM takes precedence over values that are defined in source code (i.e. `TCPIPConfig.h`). To erase the EEPROM, perform the following:

1. Make sure the development board is programmed and not in debug mode.
2. Disconnect the MPLAB® ICD2 or MPLAB REAL ICE™ from the board.
3. Press and hold BUTTON0 (RD13/S4 on Explorer 16 and RB3/S5 on PICDEM.net™ 2).
4. Press and release the MCLR button.
5. Continue holding BUTTON0 until several LEDs flash indicating the EEPROM has been cleared. This takes about 4 seconds. Alternatively, if you have UART connected to the development board, you should see the following output:

   BUTTON0 held for more than 4 seconds. Default settings restored.
6. Release BUTTON0.
7. Press and release the MCLR button again to reset the software.

A. Appendix A

A.1 Microchip Hardware

Microchip hardware, including the ZeroG PICtail can be purchases from Microchip Direct.

ICD2 first time installation

ICD3 first time installation

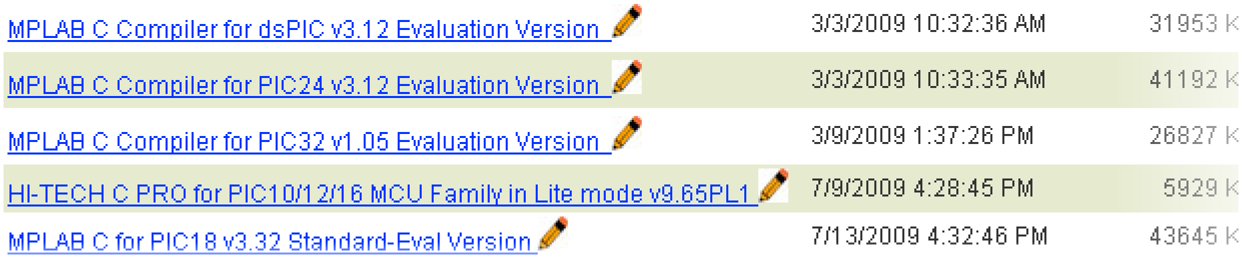
A.2 Microchip Software

TCP/IP stack (link is at the bottom of the page under the “Downloads” heading

MPLAB IDE v8.40

A.2.1 Evaluation Compiler Versions

Microchip provides evaluation versions of their compilers for PIC18, PIC24, dsPIC, and PIC32. These evaluation versions are fully-functional for the first 60 days. After that, the compilers will be unable to generate optimized code. Free compilers can be found at the following link on the Microchip website (free signup required). Click the link at the top to be taken to the free section, which should look something similar to this:



MPLAB C Compiler for dsPIC v3.12 Evaluation Version	3/3/2009 10:32:36 AM	31953 K
MPLAB C Compiler for PIC24 v3.12 Evaluation Version	3/3/2009 10:33:35 AM	41192 K
MPLAB C Compiler for PIC32 v1.05 Evaluation Version	3/9/2009 1:37:26 PM	26827 K
HI-TECH C PRO for PIC10/12/16 MCU Family in Lite mode v9.65PL1	7/9/2009 4:28:45 PM	5929 K
MPLAB C for PIC18 v3.32 Standard-Eval Version	7/13/2009 4:32:46 PM	43645 K

Figure 52: Evaluation Microchip Compilers

If you are interested in purchasing full licenses for the compilers, please see the following links (you can click the “Buy It Now” link at the top of the page:
PIC18 v3.34
PIC24 v3.20
dsPIC v3.20
PIC32 v1.05

A.3 ZeroG Wireless

The ZeroG support website, contains all the latest information, application notes, errata, module datasheets and other useful information (free signup required).

CLI Documentation – Login to the ZeroG support website, and you can find the latest version of this document under Sales, Software and Documentation > System Dev Kits > Microchip SDKs.

A.4 Tools

iperf

Wireshark PSK calculator

WEP Key calculator

WEP Key ASCII calculator

Note: The Microchip name, logo, PICtail, and PICDEM are registered trademarks of Microchip Technology Incorporated.

B. Appendix B

B.1 Federal Communication Commission Interference Statement

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

FCC Caution: To assure continued compliance, (example - use only shielded interface cables when connecting to computer or peripheral devices). Any changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate this equipment.

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

B.2 FCC Radiation Exposure Statement

This equipment complies with FCC radiation exposure limits set forth for an uncontrolled environment. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.

This device is intended only for OEM integrators under the following conditions:

1) The antenna must be installed such that 20 cm is maintained between the antenna and users, and
2) The transmitter module may not be co-located with any other transmitter or antenna.

As long as 2 conditions above are met, further transmitter test will not be required. However, the OEM integrator is still responsible for testing their end-product for any additional compliance requirements required with this module installed (for example, digital device emissions, PC peripheral requirements, etc.).

IMPORTANT NOTE: In the event that these conditions cannot be met (for example certain laptop configurations or co-location with another transmitter), then the FCC authorization is no longer considered valid and the FCC ID cannot be used on the final product. In these circumstances, the OEM integrator will be responsible for re-evaluating the end product (including the transmitter) and obtaining a separate FCC authorization.

C. Appendix C

C.1 End Product Labeling

This transmitter module is authorized only for use in device where the antenna may be installed such that 20 cm may be maintained between the antenna and users (for example access points, routers, wireless ADSL modems, and similar equipment). The final end product must be labeled in a visible area with the following: "**Contains FCCID: W7OZG2100-ZG2101**".

C.2 Manual Information That Must be Included

The user's manual for end users must include the following in-formation in a prominent location.

IMPORTANT NOTE: To comply with FCC RF exposure compliance requirements, the antenna used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

Revision History

Document ID	GG101.35	
Title	*GG101 MICROCHIP GETTING STARTED GUIDE*	
Revision History	1.01	Initial Revision
	2.02	Revised for General release code.
	2.03	Updated MAC address default settings. Added Erasing EEPROM section on Microchip development boards.
	3.0	Match defines to latest v5.00 of Microchip TCP/IP stack
	3.5	Updated to clearly illustrate running both sample applications with the Microchip TCP/IP v5.00 stack
	5.1	- Updated to match v5.10 of Microchip TCP/IP stack. - Fixed broken links to external sites and software. - Moved revision number up to match Microchip's stack version to cause less confusion.
	5.2	- Updated to match v5.20 of Microchip TCP/IP stack. - Fixed formatting - Updated references to match latest latest MPLAB(8.40), and compilers for PIC18/PIC24/PIC32 (3.34, 3.20, 1.05)