



Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of "Quality Parts,Customers Priority,Honest Operation,and Considerate Service",our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!



Contact us

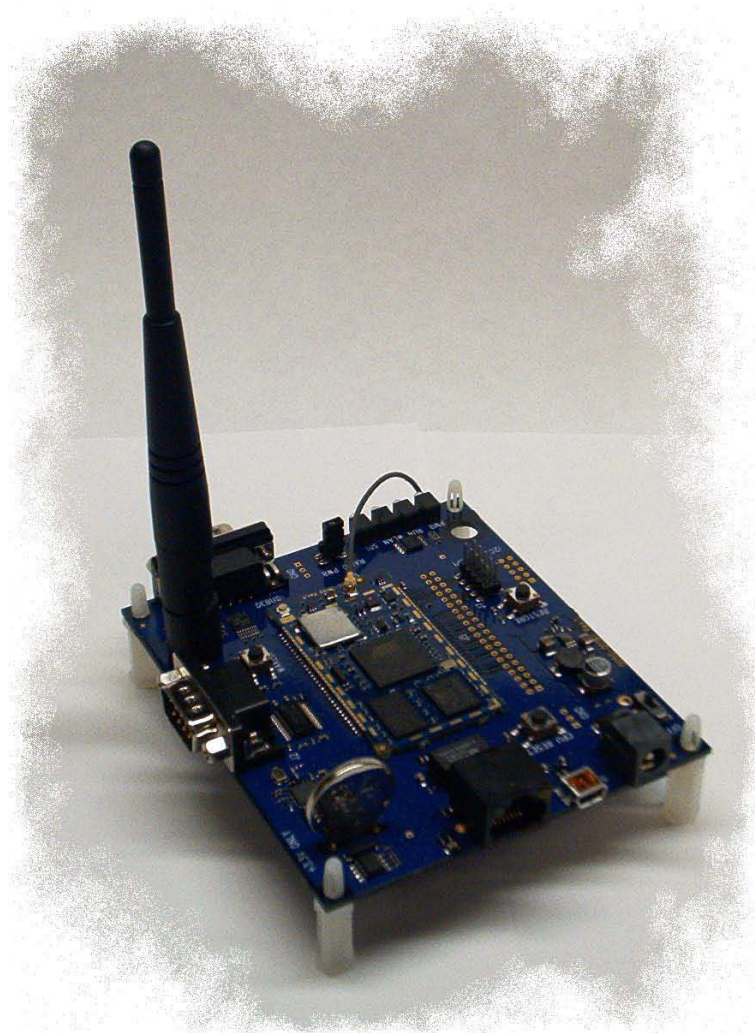
Tel: +86-755-8981 8866 Fax: +86-755-8427 6832

Email & Skype: info@chipsmall.com Web: www.chipsmall.com

Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China



DVK-WLM400/402



Developer Kit Manual



Innovative Technology for a Connected World

Laird Technologies is the world leader in the design and manufacture of customized, performance-critical products for wireless and other advanced electronics applications. Laird Technologies partners with its customers to find solutions for applications in various industries such as:

- Network Equipment
- Telecommunications
- Data Communications
- Automotive Electronics
- Computers
- Aerospace
- Military
- Medical Equipment
- Consumer Electronics

Laird Technologies offers its customers unique product solutions, dedication to research and development, as well as a seamless network of manufacturing and customer support facilities across the globe.

Copyright © 2010 Laird Technologies, Inc. All rights reserved.

The information contained in this manual and the accompanying software programs are copyrighted and all rights are reserved by Laird Technologies, Inc. Laird Technologies, Inc. reserves the right to make periodic modifications of this product without obligation to notify any person or entity of such revision. Copying, duplicating, selling, or otherwise distributing any part of this product or accompanying documentation/software without the prior consent of an authorized representative of Laird Technologies, Inc. is strictly prohibited.

All brands and product names in this publication are registered trademarks or trademarks of their respective holders.

This material is preliminary information furnished by Laird Technologies in this specification is believed to be accurate. Devices sold by Laird Technologies are covered by the warranty and patent indemnification provisions appearing in its Terms of Sale only. Laird Technologies makes no warranty, express, statutory, and implied or by description, regarding the information set forth herein. Laird Technologies reserves the right to change specifications at any time and without notice. Laird Technologies' products are intended for use in normal commercial and industrial applications. Applications requiring unusual environmental requirements such as military, medical life-support or life-sustaining equipment are specifically not recommended without additional testing for such application.

Limited Warranty, Disclaimer, Limitation of Liability For a period of one (1) year from the date of purchase by the OEM customer, Laird Technologies warrants the OEM transceiver against defects in materials and workmanship. Laird Technologies will not honor this warranty (and this warranty will be automatically void) if there has been any (1) tampering, signs of tampering; 2) repair or attempt to repair by anyone other than an Laird Technologies authorized technician. This warranty does not cover and Laird Technologies will not be liable for, any damage or failure caused by misuse, abuse, acts of God, accidents, electrical irregularity, or other causes beyond Laird Technologies' control, or claim by other than the original purchaser. In no event shall Laird Technologies be responsible or liable for any damages arising: From the use of product; From the loss of use, revenue or profit of the product; or As a result of any event, circumstance, action, or abuse beyond the control of Laird Technologies, whether such damages be direct, indirect, consequential, special or otherwise and whether such damages are incurred by the person to whom this warranty extends or third party. If, after inspection, Laird Technologies' determines that there is a defect, Laird Technologies will repair or replace the OEM transceiver at their discretion. If the product is replaced, it may be a new or refurbished product.

Change History

Issue	Change	Author	Date
00a	Document Created	John Talley	Dec-1-2009
00b	Preliminary version	John Talley	April-2-2010
00d	Distributed to Sales Team	John Talley	April-25-2010
1.00	Initial Product Release	John Talley	June-2-2010

Glossary of Terms

Acronym	Term	Description
dboard	dboard.exe	Performance tool which creates a UDP socket on a remote PC to receive packets and measure system throughput. dboard.exe IS ACCESSABLE AFTER THE INSTALLATION BY... Start→All Programs→Wism+→DashBoard Performance Monitor
DVK	Dev Kit	This refers to the Developers Kit PCB which is designed to provide all power supply and interface support hardware for a WISM+
WAPI	WISM+ API	This is the API command language used between a host and WISM+ for configuration and data transfer
WIF	WIF.exe	This stands for Wism+ InterFace program. This is the PC based executable program for configuring and running performance tests for the WISM+/Dev Kit. WIF.EXE IS ACCESSABLE AFTER THE INSTALLATION BY... Start→All Programs→Wism+→Wism+ Config Tool
WISM+		This refers to a WLM400 (for US) or WLM402 (for CE)

Table Of Contents

1	OVERVIEW	1
1.1	PARTS LIST	2
2	HARDWARE	3
2.1	PICTURE/MECHANICAL DRAWING DETAILING HIGH LEVEL COMPONENTS	3
2.2	HOST INTERFACES	4
2.2.1	Host Com Port	4
2.2.2	SPI Header	5
2.2.3	I2C Header	6
2.2.4	10/100 Ethernet Connector	6
2.2.5	USB Device “MINI” Connector	6
2.2.6	Debug Com Port	6
2.3	RF	6
2.3.1	WISM+ U.FL Port#1 & #2	6
2.3.2	Antenna Mounting Holes	6
2.4	POWER/GND	7
2.4.1	DC Power Jack	7
2.4.2	Power Switch	7
2.4.3	Switching Power Supply	7
2.4.4	WISM+ Input Power Jumper	7
2.4.5	Ground Pads	7
2.5	LEDs/INDICATORS	7
2.5.1	Power LED	7
2.5.2	RUN LED	7
2.5.3	WLAN LED	7
2.5.4	SPI	7
2.6	BUTTONS	8
2.6.1	Reset Button	8
2.6.2	Restore Defaults Button	8
2.6.3	WAKEUP Button	8
2.7	CLOCK	8
2.7.1	Real Time Clock	8
2.7.2	Real Time Clock Battery	8
2.8	WISM+	8
2.9	40 PIN HEADER	8
3	USING AT COMMANDS WITH PC SERIAL PORT	9
4	SOFTWARE- WIF	13
4.1	SETTING UP PC PORTS FOR WIF-WISM+ COMMUNICATIONS	13
4.1.1	Configuring PC USB port for WIF.exe	13
4.1.2	Configuring PC Serial port for WIF.exe	14
4.1.3	Configuring PC Ethernet port for WIF.exe	15
4.2	STARTING WIF.EXE	16
4.2.1	Starting WIF.exe over the PC USB port	16
4.2.2	Starting WIF.exe over the PC Serial port	16
4.2.3	Starting WIF.exe over PC Ethernet port	17
4.3	WIF.EXE OPERATION	18
4.4	WIF ACCESS POINT SCAN	18
4.5	CONFIGURING WISM+ ACCESS POINT TABLE	19
4.6	CONFIGURE WISM+ ANTENNA	19
5	PERFORM A WIRELESS PING PC→WISM+	20
6	AT MODE WIRELESS PING WISM+→PC	24

7	USING THE PERFORMANCE TOOL IN AD-HOC MODE.....	25
8	USING THE PERFORMANCE TOOL IN INFRASTRUCTURE MODE.....	27
9	USING WISM+ IN BRIDGE MODE W/ACCESS POINT	29
10	REGULATORY CONSIDERATIONS.....	32
10.1	US AND CANADA.....	32
10.2	EUROPE.....	32
11	DEV KIT DESIGN	33
11.1	PCB STACK-UP	33
11.2	PCB LAYOUT DRAWING.....	34
11.3	SCHEMATICS.....	35

1 OVERVIEW

This Dev Kit consists of a WLM400/WLM402 WISM+ (Wi-Fi Intelligent Serial Module) wireless module mounted to a support board (Dev Kit) which allows you to connect a PC to the WISM+ using your choice of interface. A configuration/performance tool (WIF.exe) is provided so you can set up custom WISM+ configurations, then and measure the performance of the WISM+ wireless module from Laird Technologies.

The download and installation of this package provides you with the [following executables](#) by clicking...

Start→All Programs→Wism+→

- **DashBoard Performance Monitor** (this is actually **dboard.exe** performance tool)
- **Wism+ Config Tool** (this is actually **WIF.exe** configuration tool)

The download and installation of this package provides you with the [following documentation](#) by clicking...

Start→All Programs→Wism+→Documentation→

- **Developer Kit Manual** -this manual that you are reading
- **Release Notes** -important notes about this particular release
- **SDK API Reference** -description of the Software Developer Kit source code
- **WISM+ User Manual** -the User Manual for WISM+ module (WLM400 & WLM402)
- **Wireless API Reference** -description of the low level WISM API (WAPI) commands

The download and installation of this package also provides you the [SDK \(Software Developer Kit\) source code](#). This source code (written in c) represents a working sample of how to communicate with the WISM+.

Use the **“SDK API Reference”** as a guide to understanding the C functions. The directory of source files can be found at the following path...

C:\Program Files\Laird Technologies\Wism+\wapi

There are several choices for which interface to use between the PC and the Dev Kit, but the easiest to use right out of the box, is the USB. This is the suggested way to get started...

- Get familiar with the Dev Kit hardware (see section: [“Hardware”](#)).
- Set up the WISM+ USB driver (see section: [“Configuring PC USB port for WIF.exe”](#)).
- Launch WIF.exe (see section: [“Starting WIF.exe over the PC USB port”](#)).
- Become familiar with the configuration tool (see section: [“WIF.exe Operation”](#)).
- Use WIF.exe’s Performance Tool (UDP 802.11b mode) with the program “dboard.exe” to measure the performance of the WISM+ (see section: [“Using the Performance Tool in Ad-Hoc Mode”](#)).
- Use WIF.exe’s Performance Tool (UDP 802.11b/g mode) in infrastructure mode with the program “dboard.exe” between a laptop with an access point and another laptop with a WISM+ see section: [“Using the Performance Tool in Infrastructure Mode”](#).
- Create an infrastructure mode (802.11b/g mode) network between a laptop with an access point and another laptop with a WISM+. Share a folder on each, and then copy a large file across the wireless network, and measure the performance of the WISM+... see section: [“Using WISM+ in Bridge Mode w/Access Point”](#).

Note: All PC specific references are made to the Windows XP operating system.

1.1 Parts List

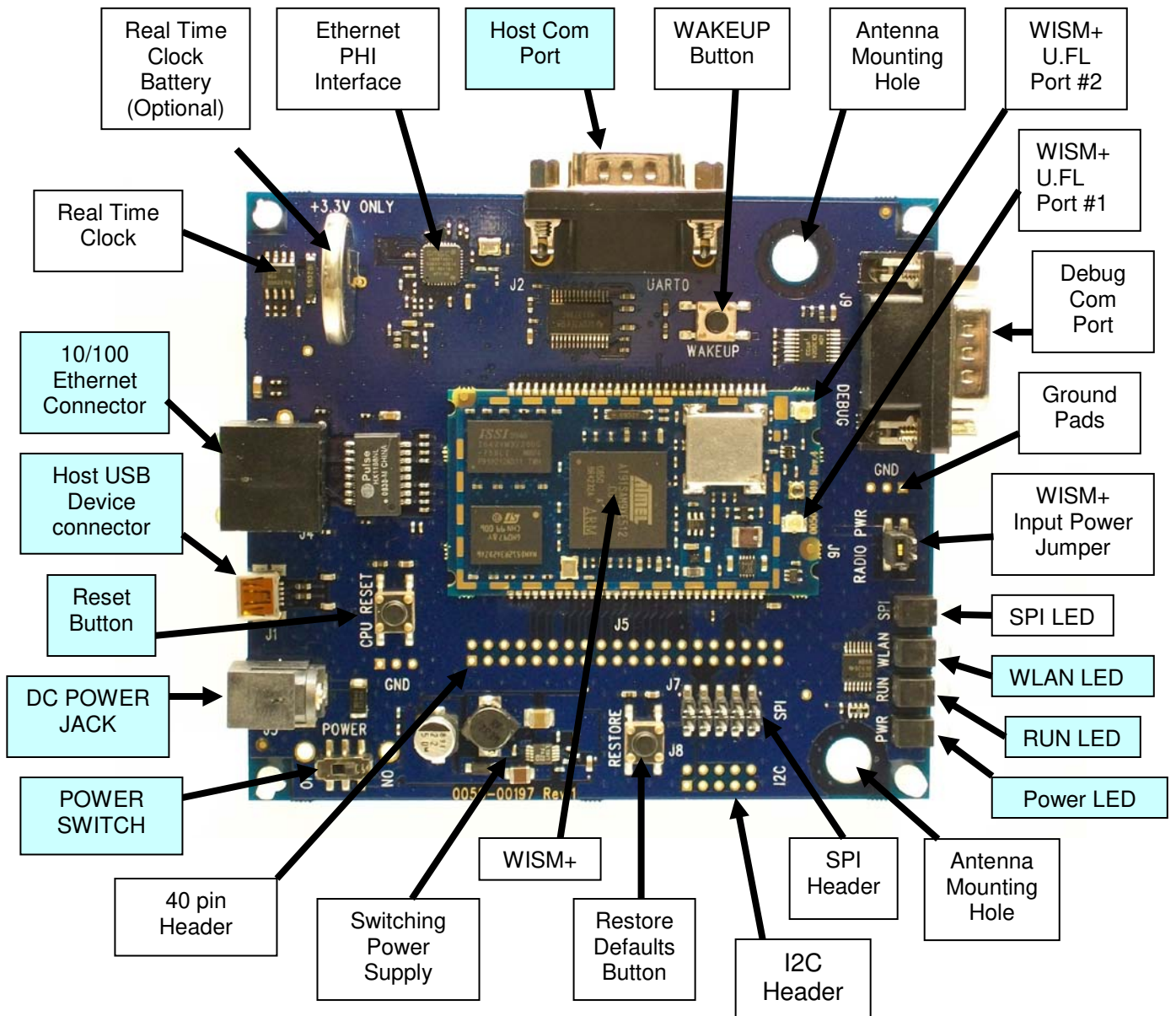
The WISM+ Dev Kit includes

- Dev Kit PCB with a WLM400 (FCC/IC) or a WLM402 (CE) WISM+ mounted
- Two Antennae installed on the Dev Kit and connected to the WISM+ antenna ports
- 120VAC/7.5VDC wall brick power adapter
- USB cable "type A to mini" for WISM+ Dev Kit connection to connect to a PC
- RS-232 serial cable (female-female) for WISM+ COM port connection to a PC serial port

Note: an Ethernet cable is not provided.

2 Hardware

2.1 Picture/Mechanical Drawing detailing high level components



Full schematics and assembly drawing are provided (see section: [“Dev Kit Design”](#)).

← Blocks in this colour are essential things to become familiar with.

2.2 Host Interfaces

2.2.1 Host Com Port

This is a DCE COM port which operates at RS-232 levels. It is a male DB9 connector which is meant to be connected to a PC serial port via the serial cable (female to female) provided with the Dev Kit.

DB9 PIN#	DB9 PIN NAME	I/O	WISM+ PIN #	WISM+ PIN NAME	DESCRIPTION
1	DB9_DCD	O	59	<u>CONNECTED</u>	The WISM+ <u>CONNECTED</u> pin is translated to RS-232 levels (and drives pin 1 of the DB9) and indicates to the PC that we have joined a network. When “connected” WISM+ drives pin 59 pin low and is level translated to a positive RS-232 level at the DB9.
2	DB9_TXD	O	6	UART_TXD	This pin is transmit data from the WISM+ back to the PC driven through a RS-232 level translator.
3	DB9_RXD	I	9	UART_RXD	This is receive data from the PC to the WISM+. An RS-232 receiver converts RS-232 levels at the connector to CMOS levels for WISM+.
4	DB9_DTR	I	12	<u>UART_DTR</u>	This is an output from the PC and is converted to CMOS levels to drive the WISM+ <u>UART_DTR</u> input. CURRENTLY THIS PIN HAS NO FUNCTION.
5	GROUND	-	-	-	This is tied to Dev Kit GROUND.
6	DB9_DSR	O	8	<u>DSR_READY</u>	When the WISM+ cannot take any more data (all buffers are full) it will assert <u>DSR_READY</u> pin low—which is translated to a high on DB9 pin 6. - This pin becomes DTR at the PC-
7	DB9_RTS	I	10	<u>UART_RTS</u>	This is converted to CMOS levels to drive the WISM+ <u>UART_RTS</u> input. The host can hold off the WISM+ from transmitting serial data when asserted- ONLY IF FLOW CONTROL IS ENABLED ON WISM+.
8	DB9_CTS	O	7	<u>UART_CTS</u>	WISM+ <u>UART_CTS</u> pin drives an RS-232 level translator to signal the host that it is ready to accept serial data. WISM+ drives this pin low when it can accept serial input from the host (creating a positive RS-232 level voltage to the PC). If WISM+ cannot accept serial input, it drives the pin high (creating a negative RS-232 voltage).
9	DB9_RI	O	-	-	This pin should be ignored at the host.

The host com port can be used for the following things...

- HyperTerminal can be used on a PC to send AT commands to the WISM+ and display responses.
- WIF.exe can be configured to operate over host com port after an “at+quit”.
- The entire WAPI command interface can be implemented over the host com port after an “at+quit”.

2.2.2 SPI Header

This is the host communications SPI interface to the WISM+. This is a dual-row 10 pin male header which mates with the Ardvark USB to SPI module http://www.totalphase.com/products/ardvark_i2cspi/. If you purchase this device, and load the drivers, WIF.exe can be configured to operate over the SPI port. The following table indicates the pin out of the SPI header...

Pin 1 is the lower left pin. The bottom row of pins are numbered 1,3,5...etc (with even pin numbers on top).

SPI Header PIN#	SPI Header PIN NAME	I/O	WISM+ PIN #	WISM+ PIN NAME	DESCRIPTION
1	$\overline{\text{SPI_REQ}}$	O	19	$\overline{\text{SPI_REQ}}$	This is an output from the WISM+ which is asserted (low) when the WISM+ has data to be transferred back to the host.
2	GND				Ground pin connected to the ground plane of the Dev Kit PCB.
3	RESERVED				This pin does go to a WISM+ port pin, but has no function.
4	N/C				This pin is not connected
5	SPI_MISO	O	28	SPI_MISO	This is the MISO (Master In Slave Out) data line for the SPI interface. This should be connected to the MISO of a SPI Master device.
6	N/C				This pin is not connected
7	SPI_CLK	I	30	SPI_CLK	This is the clock input for the SPI interface. This is an input to the WISM+ because the WISM+ is always a SPI SLAVE device, intended to be connected to the clock output of a SPI Master device.
8	SPI_MOSI	I	29	SPI_MOSI	This is the MOSI (Master Out Slave In) data line for the SPI interface. This should be connected to the MOSI of a SPI Master device.
9	$\overline{\text{SPI_SS}}$	I	31	$\overline{\text{SPI_SS}}$	This is the SPI Slave Select input to the WISM+. This is driven low by a SPI Master device to frame the data transfer.
10	GND				Ground pin connected to the ground plane of the Dev Kit PCB.

2.2.3 I2C Header

This is the WISM+ I2C two-wire interface. It is also connected to the real time clock on the Dev Kit board. These pins are also compatible with the Ardvark USB to SPI/I2C device, but currently, the only function for the I2C bus is the interface to the Real-Time-Clock populated on the Dev Kit PCB. In the future, this interface might be set up so the WISM+ can interface to other defined I2C sensors.

I2C Header PIN#	I2C Header PIN NAME	I/O	WISM+ PIN #	WISM+ PIN NAME	DESCRIPTION
1	I2C_CLK	I/O	23	I2C_CLK	This is the I2C clock line.
3	I2C_DAT	I/O	22	I2C_DAT	This is the I2C data line.
2,10	GND				These pins are tied to the ground plane of the Dev Kit PCB
4,5,6,7,8,9	N/C				These pins are not connected.

2.2.4 10/100 Ethernet Connector

This is a standard RJ-45 Ethernet connector and is another host interface to the WISM+. A standard Ethernet cable can be connected between a PC and the Dev Kit and WIF.exe can be configured to operate over the PC's Ethernet port.

2.2.5 USB Device "MINI" Connector

This is a standard USB mini connector which can be used to connect the Dev Kit to a PC over the host USB interface. A USB cable is provided to connect to a PC. By default, WIF.exe will use the USB port for WISM+ communications. This is the recommended interface to use for the Dev Kit right out of the box.

On the Dev Kit PCB, the USB connector is wired to the WISM+ USB device pins: WISM+ pin 4 (USB_DDM) and WISM+ pin 5 (USB_DDP).

2.2.6 Debug Com Port

This is an RS-232 DCE serial port. This port outputs internal debug information which can be used by Laird Technologies to attempt to debug WISM+ operations. There is no hardware flow control. DB9 Pin 2 is transmit data to the PC, and DB9 pin 3 is receive data from the PC.

2.3 RF

2.3.1 WISM+ U.FL Port#1 & #2

These are the two WISM+ RF ports. They utilize a miniature U.FL type non-standard RF connector per FCC/IC rules. The WISM+ Dev Kit comes with two antennae connected. This gives you the ability to test antenna diversity, or to configure a single antenna.

NOTE: See section: ["Configure WISM+ Antenna"](#) for the details on the procedure to change the WISM+ antenna configuration.

2.3.2 Antenna Mounting Holes

These holes are used to mount antennae. Both mounting holes can have antennae installed and connected to WISM+ RF ports #1 & #2 U.FL connectors. The Dev Kit is shipped stock with both antennae and is configured out of the box to use antenna diversity.

2.4 POWER/GND

2.4.1 DC Power Jack

The DC power jack accepts a 2mm plug from the AC/DC wall-brick power supply adapter. It is recommended that you use the wall-brick power supply which is shipped with the Developer Kit, although, the input voltage range is 5 to 25 Volts (plus on the center).

2.4.2 Power Switch

The power switch when in the ON position will connect the DC power jack to the input of the switching power supply, providing power to the entire development kit and the WISM+.

2.4.3 Switching Power Supply

The input to this switching power supply is provided from the DC power jack and is protected for reverse polarity with a series diode. The output of the switching power supply provides power (3.3V) to the entire development board circuitry as well as the WISM+.

2.4.4 WISM+ Input Power Jumper

This is a 4 position, dual row header. The two lower pins are connected to the WISM+ VCC input pins, and the two upper pins are connected to the output of the 3.3V Dev Kit switching power supply. For normal operation, only one jumper need be installed (shorting an upper pin to a lower pin).

To make WISM+ current measurements, remove the jumper and connect an ammeter between an upper and a lower pin.

To insert an external DC power input, connect a power supply ground to one of the ground pads, and the positive side of the power supply to one of the lower input power pins.

2.4.5 Ground Pads

These three thru-hole vias are made available simply to allow the user to have access to a system ground.

2.5 LEDs/INDICATORS

2.5.1 Power LED

This LED indicates DC power is being supplied to the WISM+ (DC power connection is made to the Dev Kit board and the ON/OFF switch is in the ON position).

2.5.2 RUN LED

This LED is the WISM+ heartbeat. It will blink fast at boot time, pause, and then blink at a 1Hz rate once the WISM+ processor system has booted. This LED is driven by the WISM+ "HEARTBEAT" pin (WISM+ pin# 57).

2.5.3 WLAN LED

This LED will be lit when the WISM+ is wirelessly connected to another device (or joined a network). This LED is driven by the WISM+ "CONNECTED" pin (WISM+ pin#59)

2.5.4 SPI

This LED will be lit whenever the SPI port is being used as the host interface, and communications are active over the SPI port.

2.6 BUTTONS

2.6.1 Reset Button

This provides a hardware reset to the WISM+. The WISM+ will re-boot after this has been pressed. This button drives WISM+ $\overline{\text{RESET_IN}}$ (pin# 21) to ground when it is pressed.

2.6.2 Restore Defaults Button

The Restore Defaults Button will restore the WISM+ with the saved environment which was set up at Laird production/test time. The WISM+ must perform a boot while the Restore Defaults Button is pressed and remain pressed until the heartbeat LED begins flashing in order for this function to work. This button is connected to the WISM+ $\overline{\text{RESTORE}}$ pin (pin# 56). $\overline{\text{RESTORE}}$ is pulled LOW when the button is pressed.

2.6.3 WAKEUP Button

This button is used to wake the WISM+ from a deep sleep. A WAPI API command is used over the host interface to enter deep sleep. This button is connected to the WISM+ pin WAKEUP (pin# 63). When the button is pressed the WISM+ WAKEUP pin is pulled to 3.3V which initiates a wake from deep sleep.

2.7 CLOCK

2.7.1 Real Time Clock

This is a Texas Instruments BQ32000 real time clock chip connected to the WISM+ I2C bus. WISM+ API commands can be executed to both set and read the current time and date. This is only required in certain situations when using WPA Enterprise security.

2.7.2 Real Time Clock Battery

This is the backup battery (not installed) for the real time clock on the Dev Kit board. The Real Time Clock chip will be powered when the Dev Kit is powered up -even if the battery is not installed.

This battery is a standard lithium 3.0V 220mAH tabbed coin cell. If you want battery backup for the Clock, we recommend part# CR2032RV-MFR (made by Renata) available at Mouser.

2.8 WISM+

Depending on which development kit which has been purchased, this will be either:

- WLM400 –for North America carrying FCC and IC approvals
- WLM402 –for Europe with CE regulatory compliance

2.9 40 Pin Header

The 40 pin header allows access to certain WISM+ pins. The pin out of the header (J5) is defined in the Dev Kit board schematics- See Section: [“Schematics”](#)

3 Using AT Commands with PC Serial Port

A PC serial port can be used to execute a set of simple AT commands using HyperTerminal. The following section shows you how to set up a HyperTerminal session to communicate with the WISM+ using its host COM port.

Refer to the “**AT Command Reference**” for descriptions of the AT command set.

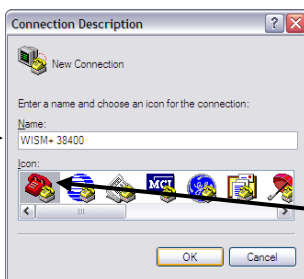
The WISM+ module (WLM400/402) implements a serial port at 3.3V levels. The WISM+ Dev Kit has industry standard RS-232 level translators and provides a male DB-9 connector so you can connect to a PC serial port using the female-female serial cable provided with the Dev Kit. The WISM+ dev kit is viewed as a DCE (Data Communications Equipment). If you use a standard serial cable (male to female), you will need a gender changer (not a Null-Modem adapter) at the WISM+ Dev Kit end.

If your PC does not have a serial port, you’ll need a USB to Serial converter. Most USB to Serial converters will require a driver to be loaded. You must first follow the instructions which came with your USB to Serial converter and install the driver as they recommend.

Once your serial port is in place and you know the COM port number (COM6, COM1 etc.) you can perform the following steps to set up HyperTerminal to communicate from your PC to the WISM+ Dev Kit hardware over the serial port...

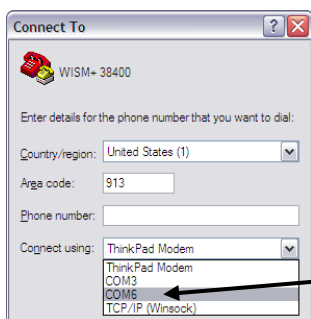
- Launch HyperTerminal by clicking...
- Start Button→All Programs→Accessories→Communications→HperTerminal and you will see the following...

We have entered a name for this HyperTerminal Session (“WISM+ 38400”).



Select the “Telephone” icon and in the “Name” field,

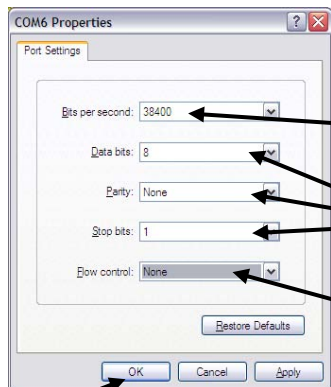
Click “OK” and you will see the following screen...



Select the COM port connected your to the WISM+ and you will see this



Click “OK” and the following COM port properties screen will appear...

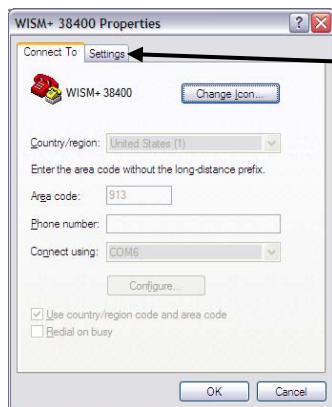


Select 38400 for the baud rate (WISM+ defaults to 38400 baud on its COM port)

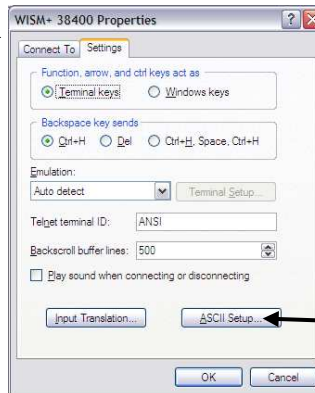
Select 8 bits, No Parity, and 1 Stop Bit

Select NO Hardware Flow Control because WISM+ is set by default to NOT use Hardware Flow Control

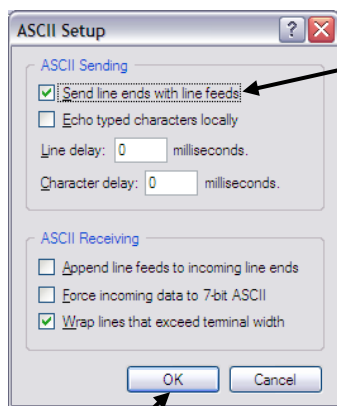
- Click "OK"
- At the top (HyperTerminal menu) click...**File→Properties** and the following screen will appear...



Click on the "Settings" Tab and you will see this



Click the "ASCII Setup" tab and you will see the ASCII Setup screen

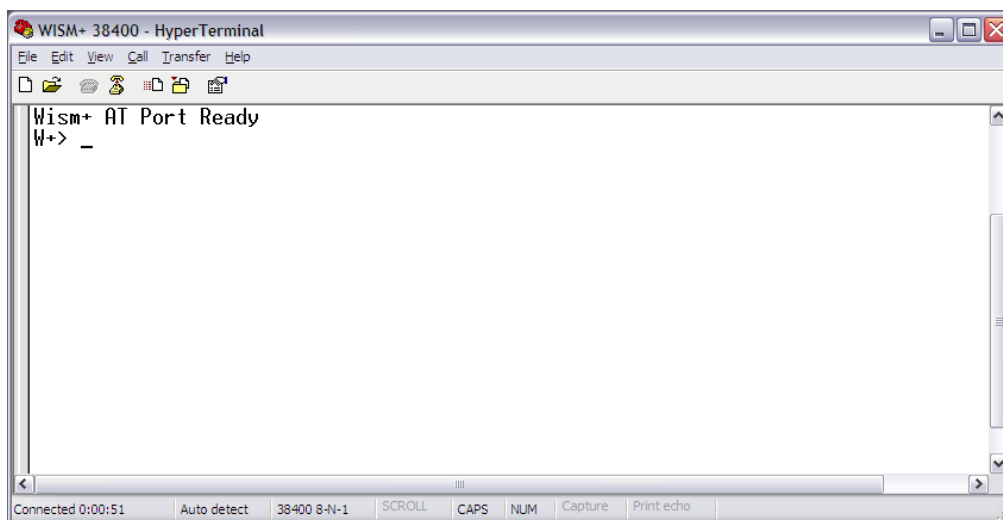


Select the "Send line ends with line feeds" checkbox. This will ensure that when you press the "ENTER" key while typing in HyperTerminal, that a Line End character is sent with the string to the WISM+.

Click the "OK" button and your COM port configuration is complete.

Now the PC serial port is configured to communicate with the WISM+ host com port. Now follow the steps below...

- Ensure that the WISM+ Dev Kit power switch is in the “OFF” position.
- Plug in the AC/DC power adapter to 120AC power outlet
- Plug the AC/DC power adapter’s DC power plug into the Dev Kit DC power jack
- Make sure that the serial cable supplied with the Dev Kit is connected between the PC (or PC USB to Serial adapter).
- In HyperTerminal, make sure the “hook” is picked up (HyperTerminal is connected)
- Switch the WISM+ Dev Kit power switch to the “ON” position
- Wait 6 or 8 seconds for the WISM+ to boot
- You will see the “Heartbeat” LED” (labelled “RUN” on the WISM+ Dev Kit) begin to flash at a 1Hz rate
- You should then see the following on the HyperTerminal



The HyperTerminal screen shot above shows the notification from WISM+ (“WISM+ AT Port Ready”) that it has successfully booted. Below the boot message, you can see the “W+>” prompt sent by WISM+ indicating that it is ready for you to type an “at+xxxx” command. By simply pressing the “ENTER” key, another “W+>” prompt should be displayed on the next line showing that the COM port connection is working in both directions.

If you **do not** see a new “W+>” prompt each time you press “ENTER”, you could have one of the following...

- You may not have selected the proper COM port in HyperTerminal
- Your USB to serial adapter may not be performing properly
- You may not have set up HyperTerminal COM port properties properly... 38400, 8bits, no parity, 1 stop bit, NO hardware flow control
- You are not using the supplied serial cable between the Dev Kit and the PC serial port
- You may not have selected “Send line ends with line feed” in the HyperTerminal ASCII setup

4 Software- WIF

WIF.exe has been installed by running the setup file from the download and is launched by clicking...

Start→All Programs→Wism+→Wism+ Config Tool

WIF.exe actually exists in the following folder... *c:\Program Files\Laird Technologies\Wism+*

The WIF.exe tool is used for the following:

- basic WISM+ configuration (over USB, Serial, Ethernet, or SPI)
- performance tool to measure WISM+ to Wireless Access Point/Router data through-put

By default, WIF.exe uses the USB interface to a Windows PC.

NOTE: WIF.exe can be made to communicate over any of the WISM+ host interfaces,

There are some specific things which need to be done depending upon which host interface of the WISM+ Dev Kit which you are going to use. USB is the default, but Ethernet, RS-232, and SPI will also work.

The required preparation for using each of these interfaces will be described in the following sections.

4.1 Setting up PC ports for WIF-WISM+ communications

4.1.1 Configuring PC USB port for WIF.exe

To use the PC USB port to communicate with the WISM+ Dev Kit, a USB driver must be installed. This driver exists on your PC after running SETUP.EXE. By default, WIF.exe will look for an active USB connection from the PC to the WISM+.

Before starting the WIF.exe software, perform the following steps...

- Set the Dev Kit power switch to the OFF position
- Plug the AC/DC wall brick power adapter into an AC power outlet, and the other end into the Dev Kit's DC power jack.
- Attach the USB cable between the Dev Kit USB jack and your PC.
- Switch the Dev Kit power switch to the "ON" position
- Wait for the WISM+ to boot- the "Heartbeat LED" (labelled "RUN" on the WISM+ Dev Kit) will begin to flash at a 1Hz rate and then you will hear the USB device register with the PC.
- Your PC will notify you that new hardware has been found
- When Window's "New Hardware Wizard" pops up, instruct it to look for the appropriate driver **on your PC**. The wizard will find the USB driver and install it.
- Once the driver installation is complete, you are ready to proceed.
- You are now ready to start Wif.exe...

Now go to section: "[Starting WIF.exe](#)"

4.1.2 Configuring PC Serial port for WIF.exe

By default at boot time, the WISM+ host COM port only interprets AT commands. You can use WIF.exe configuration/performance tool over the WISM+ host port, but you must first send an “**at+quit**” command. After this command is sent to the WISM+, there will be no response from WISM+, but the WISM+ host COM port will now be in WAPI mode and is compatible with the WISM+ API commands used by WIF.exe.

To use WIF.exe over the WISM+ host COM port, perform the following steps:

- Follow the steps in section: “[Using AT Commands with PC Serial Port](#)” in order to set up your PC’s serial port for communications.
- Issue an “**at+quit**” AT command using HyperTerminal (and there will be no response).
- Now quit the HyperTerminal session (or hang up the hook) which will release the serial port for WIF.exe.
- Now go to section: “[Starting WIF.exe over the PC Serial port](#)”

4.1.3 Configuring PC Ethernet port for WIF.exe

WIF.exe can be configured to operate over the PC Ethernet port. There are a couple of things which must be done first, in order to allow proper Ethernet operation...

- WinPcap must be installed on your PC.
- Any PC service which traps out of the ordinary network traffic must be halted.

WinPcap

WinPcap is a networking package which **MUST** be installed on your PC in order for WIF.exe to communicate to the WISM+ over Ethernet. **You would have been warned at install time if WinPcap didn't exist on your PC.** This package is used by programs like "WireShark" to monitor network traffic (just one example). You must install this (if you don't have it) before attempting to use WIF.exe over the Ethernet port. It allows WIF.exe to capture and transmit network packets bypassing the windows protocol stack.

To install WinPcap on your PC, go to "<http://www.winpcap.org>". Click on the option on the left of the screen called "Get WinPcap". Under the "WinPcap 4.1.1 download" section, click on the link...

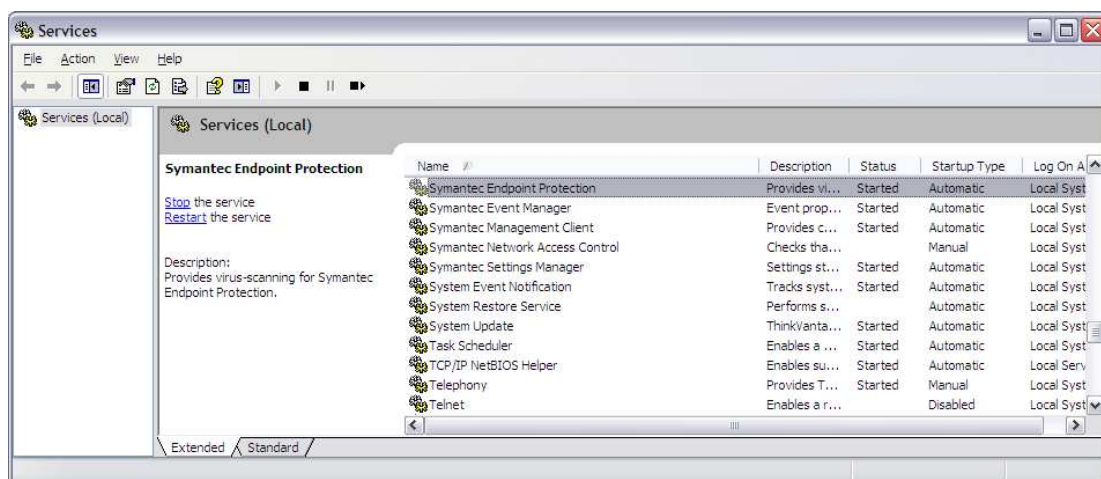
"WinPcap auto-installer (driver +DLLs)"

Download the installer and launch it on your PC. Follow the installation instructions so that it will automatically start up at boot time.

Stopping services which interfere with network traffic

At Laird, we use Symantec virus protection and network traffic monitoring. Part of its job is to look for potentially harmful network traffic which might be hacking into your computer. The WIF to WISM+ communications look potentially harmful to these types of services, and the services will interfere and capture the network traffic thus keeping WIF.exe from communicating with WISM+ over the Ethernet interface.

Click...Start→Control Panel and double-click on the "Administrative Tools" icon. Next, double-click on the "Services" icon. Once the "Services" window is open, scroll down to the virus protection services. The following screen shot is an example of the Symantec services which are running (if you use Symantec virus protection). Over to the right, under the "Status" column, you can see the services which have been started. Right click on the service, and click "Stop" on all the Symantec services which indicate "Started". Your PC will be vulnerable to attack, but you will only be connected to the WISM+ Dev Kit. When you are finished using WIF.exe, turn the services back on to connect back to your network, or simply re-boot.



4.2 Starting WIF.exe

Since WIF.exe uses the USB as the default port, an active USB connection is checked for at launch time, otherwise a port selection screen will pop up. See the following sections for specific instructions for using USB, serial, Ethernet & SPI.

4.2.1 Starting WIF.exe over the PC USB port

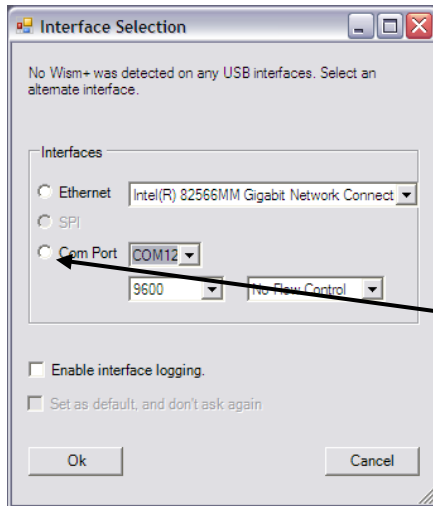
With the USB driver successfully installed, (see section: "[Configuring PC USB port for WIF.exe](#)") the USB cable connected between the WISM+ Dev Kit and the PC, WISM+ Dev Kit powered up, and the USB interface recognized by the PC, launch WIF by clicking...

Start→All Programs→Wism+→Wism+ Config Tool

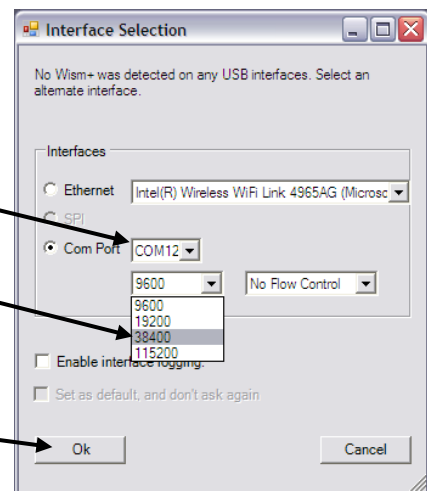
If everything is working properly, you will see the WIF front screen populated with the current state saved in WISM+ constant memory. Now go to section: "[WIF.exe Operation](#)".

4.2.2 Starting WIF.exe over the PC Serial port

- Since WIF.exe is looking for the USB connection upon launch, MAKE SURE THE USB CABLE IS DISCONNECTED.
- Make sure that you have exited AT command mode for the WISM+ host COM port as described in section: "[Configuring PC Serial port for WIF.exe](#)" (issue the "at+quit" command).
- Now, launch WIF.exe by clicking...**Start→All Programs→Wism+→Wism+ Config Tool**
- The following interface selection screen will appear...



Click on "Com Port" and then pull down the COM port selection box to select you PCs COM port. The default settings for WISM+ is 38400 with No Flow Control



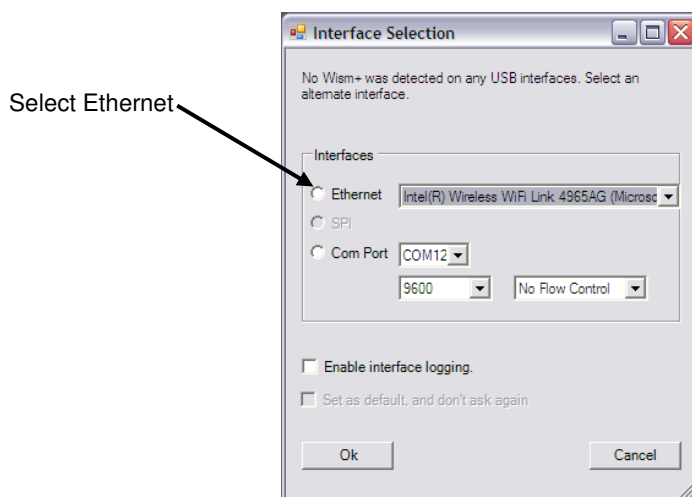
After setting up the COM port parameters click "OK".

If everything is working properly, you will see the WIF front screen populated with the current state saved in WISM+ constant memory.

Now go to section: "[WIF.exe Operation](#)".

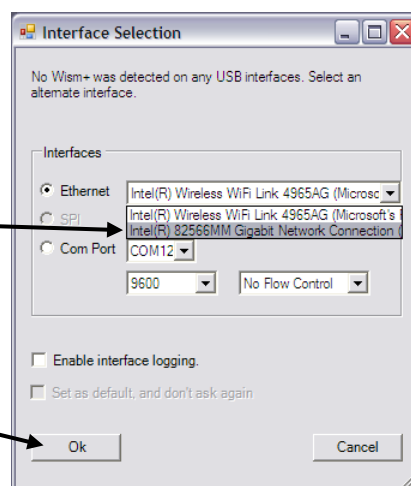
4.2.3 Starting WIF.exe over PC Ethernet port

- Since WIF.exe is looking for the USB connection upon launch, MAKE SURE THE USB CABLE IS DISCONNECTED.
- Connect a network cable between the WISM+ Dev Kit Ethernet port and the PC's Ethernet port.
- Make sure you have followed the steps in section: "[Configuring PC Ethernet port for WIF.exe](#)".
- Power up the WISM+ Dev Kit (or press the reset button) and wait for the "Run Led" to begin flashing at a 1Hz rate.
- Now launch Wif.exe by clicking...**Start→All Programs→Wism+→Wism+ Config Tool.** You will then see WIF.exe's interface selection screen as shown below...



Select the Ethernet adapter associated your PC's wired network connection,

Click OK.



If everything is working properly, you will see the WIF front screen populated with the current state saved in WISM+ constant memory.

Now go to section: "[WIF.exe Operation](#)".

4.3 WIF.exe Operation

This is the front page presented by WIF.exe. It gives you access to three top level menus...

- File
- Tools
- Help

The front page also gives you information on...

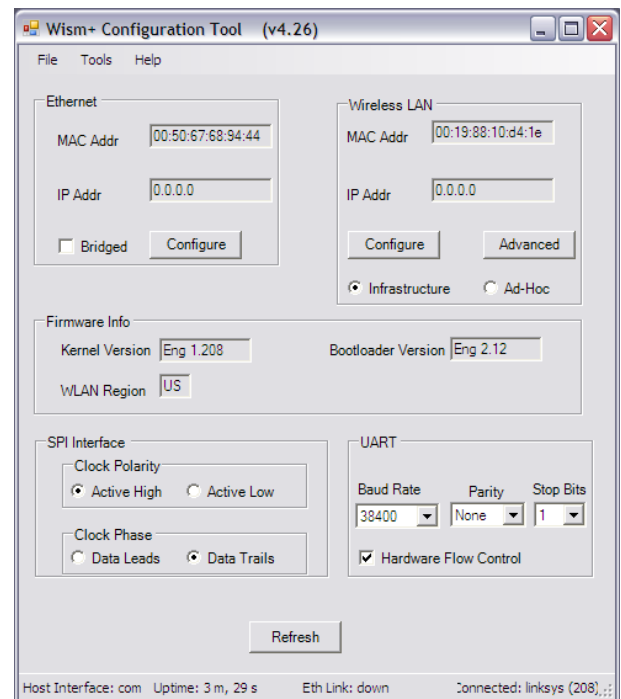
- Wired Ethernet connection MAC & IP addresses
- 802.11 wireless MAC and IP addresses
- Firmware & Bootloader Versions.
- Region Code (US/CE)
- SPI configuration parameters
- UART configuration parameters

You may click the "Refresh" button at the bottom at any time to read the WISM+ again.

This front page also allows you to select or de-select Bridged mode.

It also allows selection of Infrastructure/Ad-Hoc mode.

You can set the SPI bus and UART parameters.



NOTE:

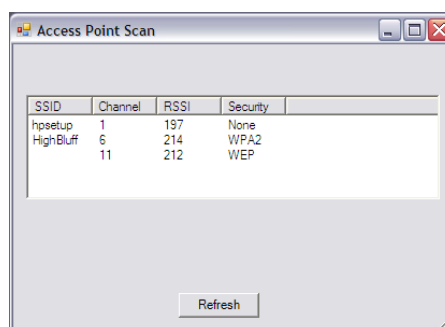
This tool reads and writes the state of the configured parameters stored in the WISM+. Any time you modify parameters, they are written to the WISM+ upon exit of the WIF.exe program. You must then reset the WISM+ for them to take affect, and re-start WIF.exe to view the changes by the configuration tool.

4.4 WIF Access Point Scan

From the WIF front page, pull down the "Tools" menu option and select "Access Point Scan". WISM+ will then go looking for all access points within range and present you with a window like what is shown below...

You will see...

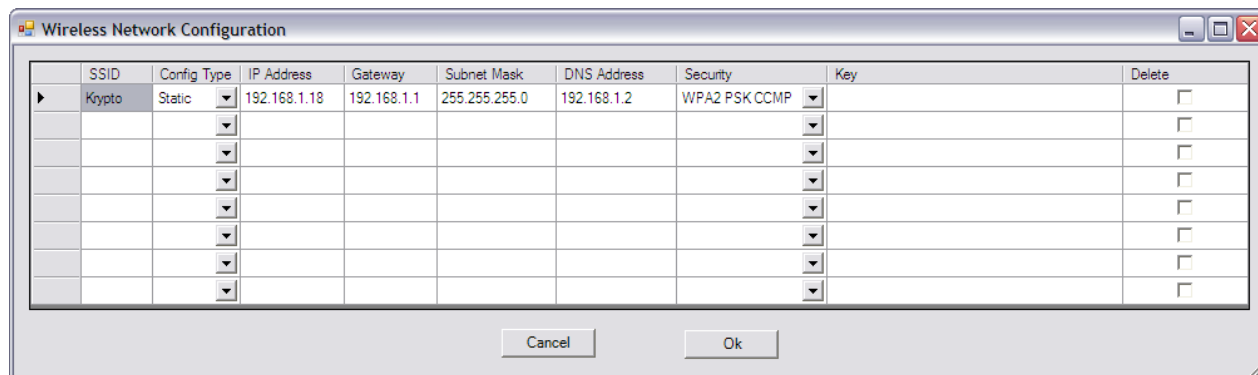
- All access points and Ad-Hoc connections
- Their SSID's
- Their channel
- Associated Signal level (RSSI)
- What type of security



4.5 Configuring WISM+ Access Point Table

From the WIF front page, in the “Wireless LAN” section (upper right), press the “Configure” button and WIF.exe will display the contents of WISM+’s Access Point Table as shown below...

Access Point Table



SSID	Config Type	IP Address	Gateway	Subnet Mask	DNS Address	Security	Key	Delete
Krypto	Static	192.168.1.18	192.168.1.1	255.255.255.0	192.168.1.2	WPA2 PSK CCMP		<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>
								<input type="checkbox"/>

This page will allow you to describe multiple access points which WISM+ could connect to. SSID, static IP or DHCP, Gateway, Subnet Mask, DSN Address, Security type and Security key parameters can be configured.

To delete an entry, click the “Delete” box (on the right) and press OK.

Note: Only one Ad-Hoc connection can be described and MUST be configured as the FIRST entry in the table.

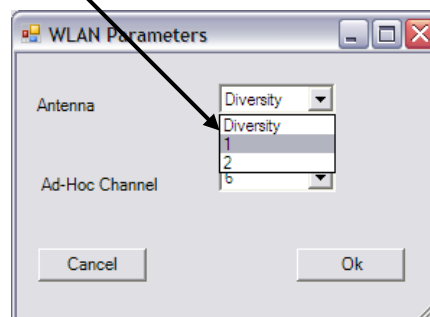
By clicking OK, and then exiting WIF.exe, the parameters will be saved to WISM+’s access point table in Non-Volatile memory.

4.6 Configure WISM+ Antenna

From the WIF.exe front page, in the Wireless LAN section (upper right), click the “Advanced” button. You will see the following “WLAN Parameters” window (on the left)...



By pulling down the Antenna selection box, you can select U.FL port #1, U.FL port #2 or the diversity option in which two antennae MUST be used.



Once a selection has been made, click OK and exit WIF. Re-start the WISM+ and the selection is permanent.

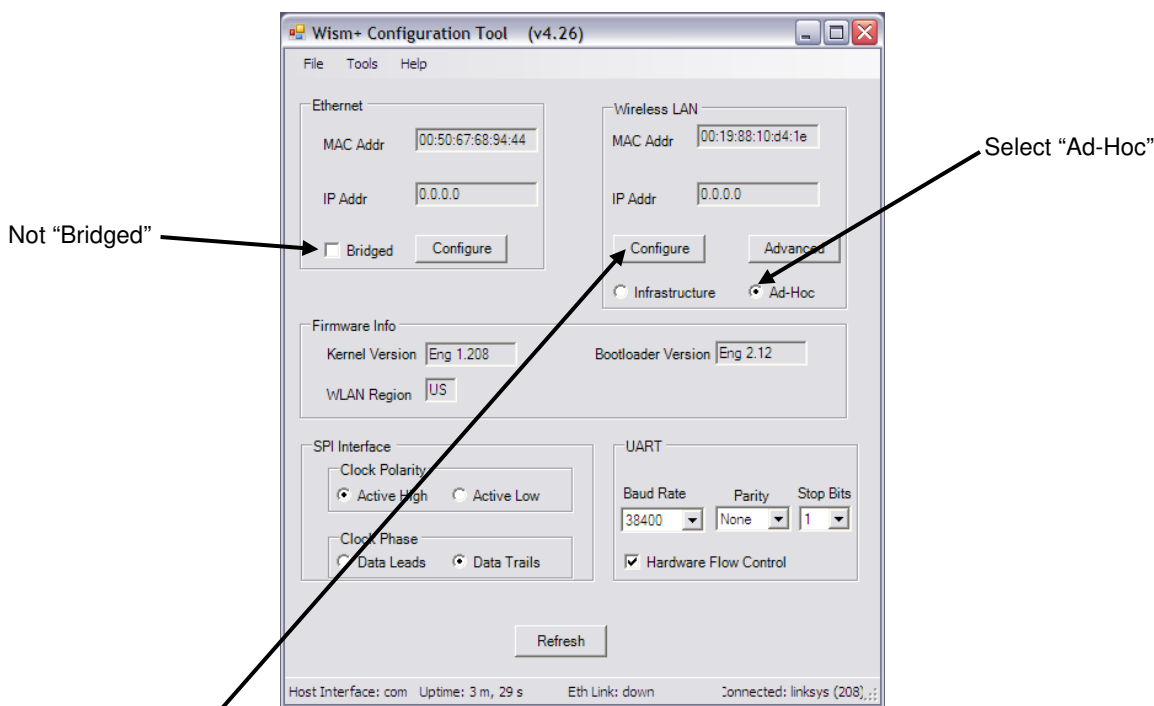
NOTE: When diversity is selected, the Marvell 8686 antenna diversity algorithm will select which antenna is used for both transmitting and receiving. It constantly evaluates signal to noise ratio and packet error rates and will make decisions to switch antennae. By default, antenna #1 is configured to be used for transmit and receive.

5 Perform a Wireless Ping PC→WISM+

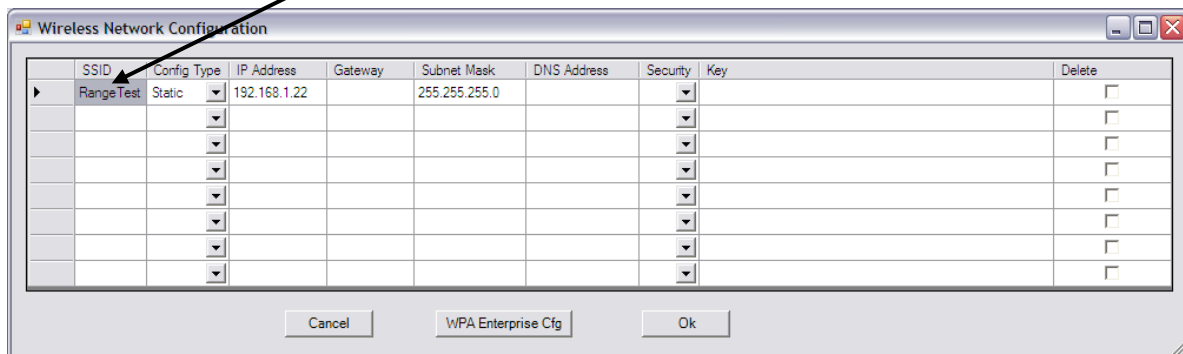
The following section will walk you through the procedure of how to create an Ad-Hoc connection between a WISM+ Dev Kit and a wireless enabled PC in order to perform a wireless network Ping.

The WISM+ can operate either in infrastructure mode (communication with a Wireless Access Point or Router) or in Ad-Hoc mode (with another WISM+, wireless PC in Ad-Hoc mode or any other 802.11b device capable of creating an Ad-Hoc Connection). **An Ad-Hoc mode connection is always 802.11b.**

Launch WIF.exe using the USB connection, and then select the following...



Now click "Configure", and define the following Ad-Hoc static network connection as shown below...



The first access point table entry is the only entry in the table which can be used for an Ad-Hoc connection, and we have set it up with an SSID of "Range Test" with a "Static" IP address of 192.168.1.22 & subnet mask 255.255.255.0. Now click "OK" and then close WIF. Now, reset the WISM+ in order for the changes to be in effect.