Light of the World

Senior Design Project

Joe Altura, Brian Appleton, Sean Baur, James Yurkovich

2012-2013

Table of Contents

1 Introduction

The engineering world has taken a major turn toward designing energy efficient devices and technologies. This global "green" initiative has caused a great shift in engineering design wherein companies are sacrificing costs in order to make more environmental friendly decisions. The United States General Services Administration (GSA), as well as other government agencies, is working to reduce the environmental impact of the federal government.[1]

Due in part to large government tax incentives[2], many large companies are taking part in this green initiative. The automotive industry is shifting to hybrid and plugin hybrid electric vehicles, while large commercial buildings are installing power management systems. One major way of reducing the energy consumption of buildings is to install more energy efficient lighting systems with control systems; many new buildings are being constructed with such lighting systems.

Although many efficient lighting solutions exist, such as those offered by Lutron, these systems have their flaws. The primary issue with such systems is the monumental cost of converting massive buildings from incandescent lighting to more efficient systems, such as those that utilize compact fluorescent lamps (CFLs) or light emitting diodes (LEDs). This is because many of the high-end systems involve a control box wired into the wall. CFLs are a good alternative to incandescent bulbs, as they are also more energy efficient; however, CFLs are difficult to dispose of. In an attempt to address the desire for more energy efficient lighting technology, we have designed and constructed a prototype LED light that is "plug and play" – able to be screwed into an existing socket without any additional installation.

A major part of our design was thus centered on the control for our bulbs. We implemented a user interface that controls individual bulbs from an Android smartphone without having to physically access a wall-mounted portal. With this control comes the idea of "smart lighting," a system that could be programmed to adjust for home automation, stage lighting control, or detailed response to motion sensors. Each of these applications has far reaching implications for reaching the goal of greener design systems. Additionally, the fact that incandescent light bulbs are being phased out provides greater opportunity for a system such as ours to have an impact on the market.

In order to effectively reach the goal of producing a viable LED bulb with good utility, the prototype must meet certain requirements. These requirements were proposed in the high-level design report in the first semester, and we have evaluated our progress toward each requirement. Overall, we met our design specifications very well. We made several key design decisions at the very beginning of the process which contracted our original system requirements. Immediately after the initial system requirements were drawn up, we decided against attempting to make a multi-color light. Our reasoning for this was primarily based on the on our desire to make a functioning consumer-oriented product that would be able to be used in the common household or large commercial building; we decided that a multi-color bulb was not suited for this market.

A multi-color bulb is less desirable from a technical standpoint as well. A multi-color bulb would be achieved by combining the light from four different colored LEDs: white, red, green, and blue. The addition of more bulbs requires additional electronics to provide power and handle the dimming and

[1] FY 2011-2016 Strategic Sustainability Performance Plan, http://www.gsa.gov/graphics/staffoffices/SSPP_11022011.pdf

[2] Energy Policy Act of 2005, http://www.epa.gov/statelocalclimate/documents/pdf/4_20_06_EPACT_Tax_incentives_Prindle.pdf

switching, but it also reduces the light output significantly. For these two reasons, we decided against designing in the color changing ability of our prototype bulb.

The most general system requirements were met. Our prototype bulb was constructed within the budget, was successfully dimmed from a remote, and was powered by screwing the assembled prototype into a normal light socket. By following the exact specifications spelled out by manufacturer of the LED and other parts, we have no reason to expect that our bulb will not last as long as it rated to last.

We identified five subsystems. The first subsystem, the Light Module, was successfully built and all requirements met. We estimate the light output from our prototype to be equivalent to a 110-watt incandescent bulb. After several iterations and tweaking of the microcontroller timing, the dimming of the bulb was smooth and did not result in any flickering.

The Package subsystem also met all of the design requirements, surpassing many of the original expectations due to the 3D printed plastic housing. The housing protected the user from any bare wires, in addition to providing strong housing capable of being handled. The final prototype looks and functions very similar to a normal light bulb, with all the electronics contained inside the plastic housing and the only connection to live and ground made through the socket. We researched heat dissipation and decided upon a sizable heatsink that gave us headroom for heat dissipation.

The Power subsystem also met all of its requirements. All major components were provided the necessary power, all converted from a standard 120 V wall socket. The Control subsystem also met all necessary requirements, although several were not met due to design decisions early on. The TRIAC dimming was accomplished with pulse width modulation from microcontroller signals. It was determined that nonvolatile memory was unnecessary for the successful functioning of our prototype. The bulb also was able to communicate a remote (either Android app or webpage on server).

The Remote subsystem met all of the specified requirements. The bulb was able to communicate effectively and handle inputs from two different remotes without consequence. The server was set up such that an unlimited number of bulbs could be connected and controlled individually. Although the system was initialized with only one bulb, more could be added.

Several Future Enhancements were identified and completed. A custom flyback converter was designed and implemented to great success. The Android app was successfully designed and implemented. The app currently requires that it be connected to the same network as the server and the bulb, but that set up currently works. A better implementation is discussed later in this report.

A more detailed description of the rest of the design requirements is presented in **Table 2.1**.

2 System Requirements

Table 2.1 Prototype bulb requirements.

SYSTEM REQUIREMENTS		
Requirement	Description	Result
General Purpose	Must be able to communicate with the user interface to switch lights off and on as well as have a dimming function	Completed
User Implementation	Must be able to be screwed into normal light socket and operate without any other implementation	Completed
Expected Life of Product	Must be comparable to commercial LED bulbs, i.e. have an expected lifetime of several thousand hours	Completed
Cost	System prototype must be within $500 budget to design and produce	Completed

SUBSYSTEM REQUIREMENTS		
Light Module		
General	Must illuminate reasonable area Must provide consistent brightness over wide range Must dim in a controllable manner Must be able to translate dimming signal from MC	Completed Completed Completed Completed
Package		
Safety	Must adhere to standard electrical codes Must be no danger of electrical fires Must have no exposed line voltages	Completed Completed Completed
Performance	Must distribute light evenly Must dissipate heat effectively	Completed Completed
Mechanical	Must have mechanical strength when turning Must be durable to endure possible falls	Completed Completed
Power		
General	Must be able to provide rated power to each necessary subsystem (microcontroller, light module, and communication) Must be able to transform standard U.S. line power (120 V_{RMS}, 60 Hz) to necessary board ratings (3.3 V_{DC}) Must minimize effects on other external devices on circuit	Completed Completed Completed
Control		
General	Must have internal clock Must be able to perform tasks at specified times	Completed No- design decision
Memory	Must be able to store system state in nonvolatile memory Must be able to load system state from nonvolatile memory Must be able to store presets in nonvolatile memory Must be able to load presets from nonvolatile memory	No–design decision No–design decision No–design decision No–design decision
Input	Must be able to receive input from remote Must be able to change system state based on remote signal in nonvolatile memory	Completed Completed
Output	Must produce a dimming control signal	Completed

Remote		
User Interface	Capable of taking user input from one or multiple sources Able to distinguish between input sources Able to resolve conflicting inputs in predefined manner User can specify one light bulb among many User must be able to select from preset conditions or define manually	Completed Completed Completed Completed
Communication	Able to relay user inputs to microcontroller Must not interfere with other equipment on same circuit within a building	Completed Completed
FUTURE ENHANCEMENT REQUIREMENTS		
Power	Custom flyback converter Power factor optimization Efficiency optimization	Completed Completed Completed
User Interface & Control	Mobile application Adaptive response to patterned input behavior	Completed No–design decision
Ambient State	Must be comparable to commercial LED bulbs, i.e. have an expected lifetime of several thousand hours	Completed
Lighting Features	RGBW LEDs with mixing control Ability to detect ambient light levels and signal changes Ability to detect motion	No–design decision No–design decision No–design decision

3 Detailed Project Description (Final Designs)

3.1 Server

The original requirement of the bulb being able to communicate with an external remote for control of the bulb (Android app/website) made the creation of a server necessary. This involved creating a MYSQL database that would store the values and a server that would handle the 'getting' and 'setting' of these database values. The server was constructed using the free Wamp Server software available online and a corresponding phpMyAdmin database (a service that comes with the Wamp server download). We set up a Wamp server on one of our laptops in order to handle communications between our light module and the user interface. We wrote the server in javascript, php, SQL, and java. In order to facilitate the connection from our server to the Wi-Fi module without interruption from other devices, we set up a Netgear router with its own Wi-Fi network. The laptop running the Wamp server connects to this network, as does any device attempting to pull data from the database (such as an Android phone). The database was set up with values for an ID number for each bulb, a dimming value, and four additional variables in case they were needed. The dimming value ranges from 0 to 100. A JSON string contains the information that is accepted by the server and used to set values in the database (see **Figure 3.1.1**) Finally, a test page was created initially to test the server's functionality. This page was later modified to serve as another client through which the user could adjust the brightness of the bulb.

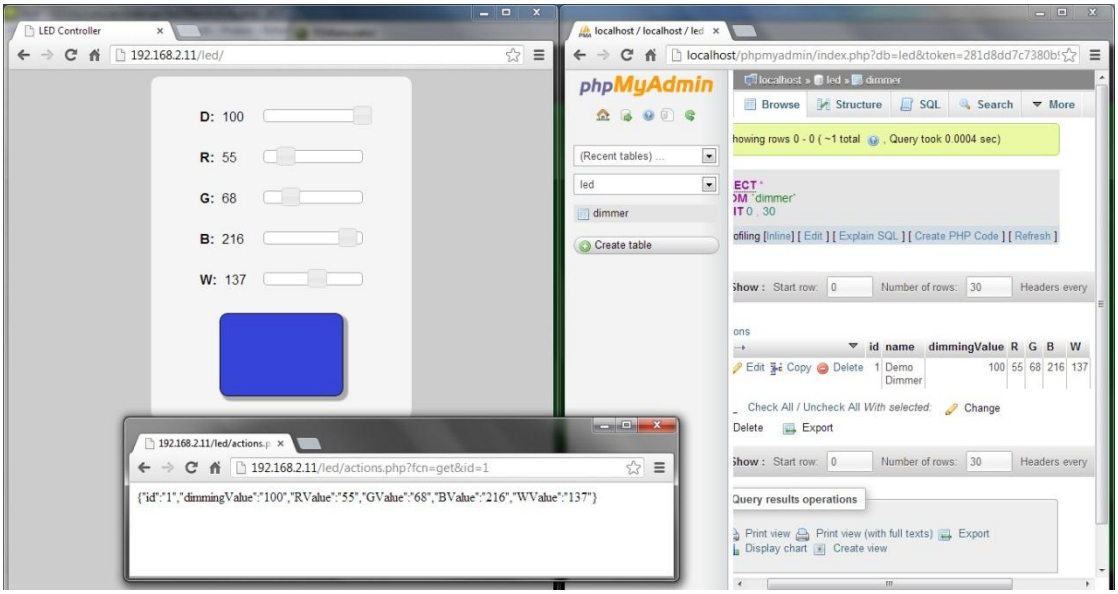

Figure 3.1.1 Screenshot of the test page, JSON communication string, and database. The left pane shows the initial page used to ensure proper functioning of the server's ability to set values in the database. The right pane is a view of the database with a single bulb ("Demo Dimmer") attached. The bottom pane is the JSON string that is used to set values in the server.

The server was configured such that many different clients could connect to the server. In this context, a client is any device used to get or set values in the server. This hierarchy is shown in **Figure 3.1.2**.

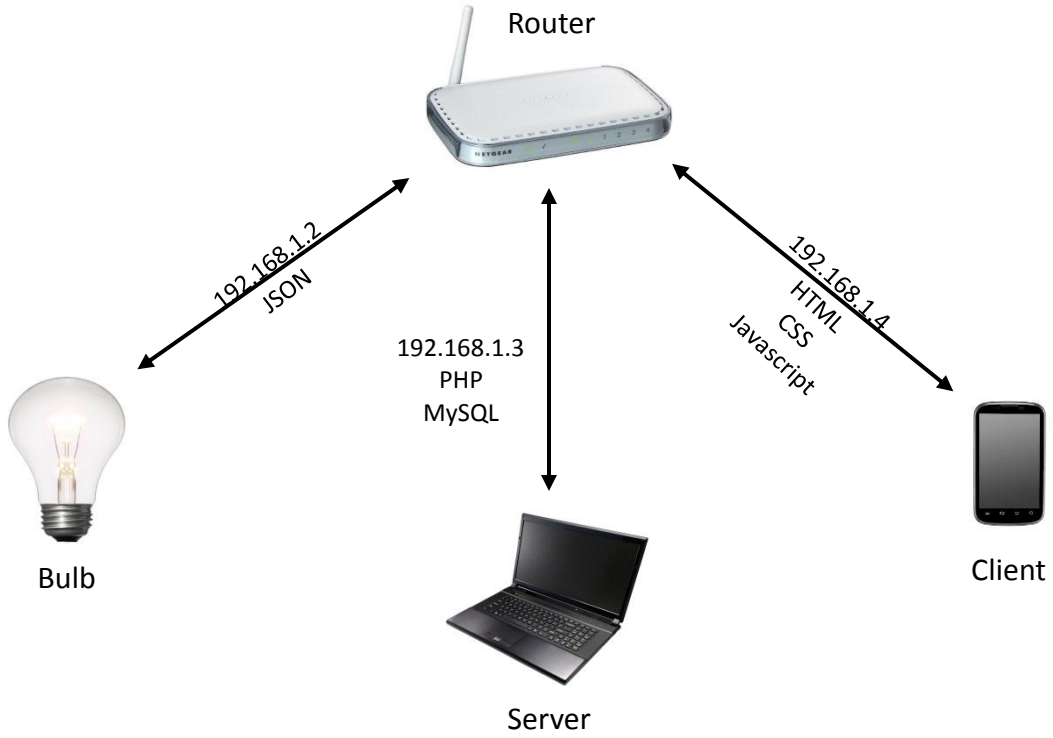

Figure3.1.2 Server hierarchy diagram. The server communicates with the router using PHP and MySQL, which is then transmitted to the bulb in the form of a JSON string. The client (Android phone or laptop) gets or sets values from the server using HTML, CSS, and JavaScript.

In order for the user to most easily interface with our prototype bulb, we developed an Android application (called “LED”) that would enable the user to dim the bulb from their Android smartphone (see **Figure 3.1.3**). The app itself has minimal function, simply a slider to adjust the brightness and a refresh button to pull values from the server.

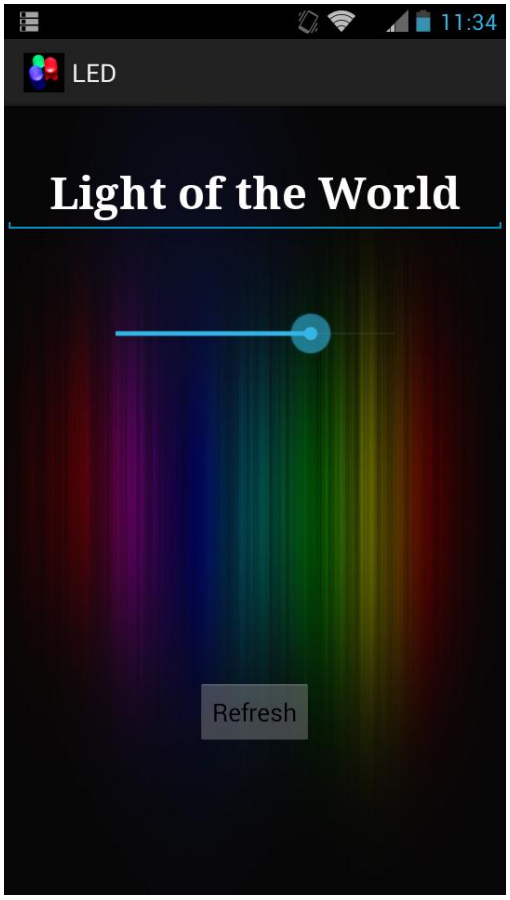

Figure 3.1.3 Android app. The slider is used to adjust the brightness (using a set function), and the refresh button is used to get fresh values from the server in case they were updated from another client since the app was last used.

3.2 Embedded Systems

Embedded systems within the light bulb encompassed two of the subsystems within the design. The major concerns and design challenges within these two subsystems included the ability to interface with other portions of the design.

3.2.1 Communication

Communication between the light bulb and server required a system that could connect the light bulb to the same wireless local area network (WLAN) as the server. From the same network, the microcontroller/transceiver needed to be able to pull values from the server using the Java get function previously explained in the server description. Calling this function then returned the string associated with the server as well as other information regarding date, time, and connection. This string could then be parsed and the dimming value acquired to be used in the dimming portions of the code.

This required a number of protocols and hardware. First, it was necessary to implement some kind of hardware that would be capable of connecting to a Wi-Fi network. This hardware would also need to interface with the microcontroller and power supply. Connecting to the router and network would also require an implementation of some sort of TCP/IP stack that would be capable of handling the software necessary with connecting to a common router and then communicating with a server on a computer.

Given these requirements we decided to look at dedicated Wi-Fi transceivers. Wi-Fi is a common trademark that denotes use of the IEEE 802.11 standards. The transceiver chosen was the Microchip MRF24WB0MA. This module is capable of connecting to a Wi-Fi network while communicating with a microcontroller using a 4-wire SPI interface. Beyond, the four signals associated with SPI, the modules also requires power(3.3V, GND), interrupt, hibernate, and reset signals. The last three of these signals also go to the microcontroller for the purpose of interface. Figure 3.3.1 below contains a block diagram of the Wi-Fi transceiver. Note that the figure also contains JTAG and serial debug signals. These signals are used for testing if the module is functioning properly and are regularly unused during operation.

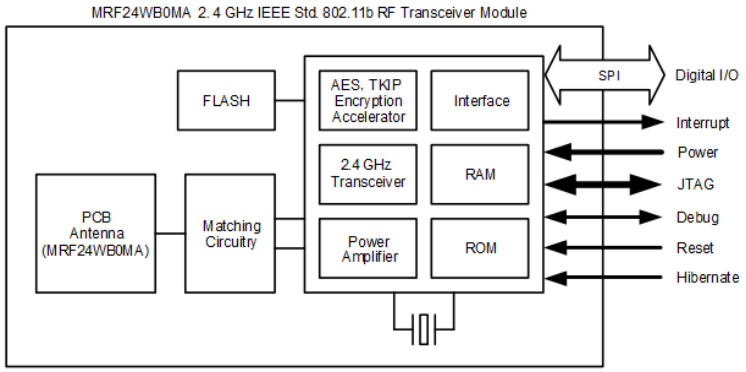

Figure 3.3.1 MRF24WBOMA Block Diagram

This transceiver is supported with Microchip's TCP/IP stack that contains a number of protocols capable of connecting to the router and then requesting data from the server.

A TCP/IP stack is structured in a layer of protocols. Higher-level protocols such as TCP utilize lower level protocols such as IP or MAC. Figure 3.3.2 contains an image illustrating some of the protocols and libraries of functions utilized in the TCP/IP stack.

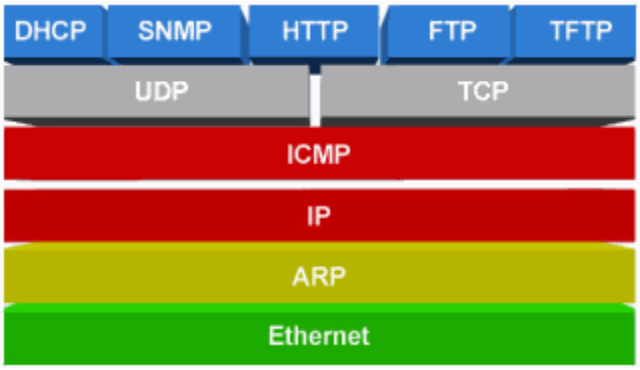

Figure 3.3.2 Microchip Stack Image

This module fulfills all the requirements for communication. It is capable of being interfaced with a microcontroller. It can connect to a network and it can communicate with a server. Furthermore, it is a Microchip product intended for use with Microchip microcontrollers. This module dictates that we need to use a PIC18, PIC24, dsPIC, or PIC32. The development board used initially was a PIC18 series (PIC18F97J60), which made porting over to a similar microcontroller (PIC18LF6722) in final implementation easier.

The SPI interface and other signals necessary for the transceiver to function are dictated in the datasheet for the module and mostly involve the use of decoupling capacitors and pull-up/pull-down resistors. Figure 3.3.3 illustrates the signals run from the transceiver and necessary biasing circuitry.

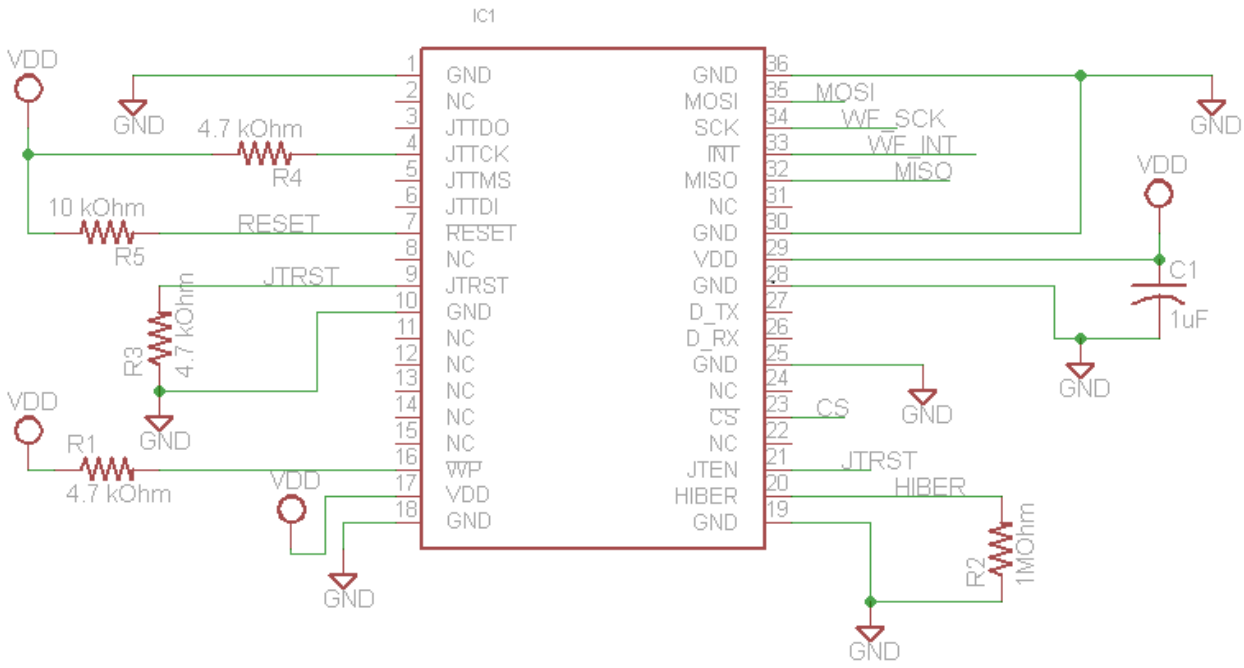

Figure 3.3.3 Schematic of Wi-Fi module

The transceiver communicates as an SPI slave to the microcontroller. This involves the four signals CS (Chip Select), MOSI (Master Out Slave In), MISO (Master In Slave Out), and WF_SCK (Serial Clock). SPI communication occurs as follows. The CS signal is driven low during communication and the master begins the serial clock. If the master is sending data to the slave then on the MOSI line data is written to be read on the rising edge, while on the MISO line dummy data is sent and read on the falling edge. Then the clock is turned off and CS pulled high once the transfer of information is complete. If the slave is sending data to the master then the same thing occurs except there is dummy data on the MOSI line and meaningful data on the MISO line. Beyond that the HIBER (Hibernate), RESET, and WF_INT (Wi-Fi External Interrupt) are also routed to the microcontroller. The HIBER and RESET signals allow the microcontroller to place the transceiver into a hibernation state and reset. Reset occurs on the RESET being pulled low and hibernation occurs on HIBER being pulled high. Meanwhile, the WF_INT signal is placed on an external interrupt pin of the microcontroller. When this is pulled low by the transceiver it indicates that the transceiver has information to send to the microcontroller. Otherwise this line is kept high by a pull-up resistor. It was necessary to bias the WP to prevent a rewriting of the firmware on the transceiver. It was also necessary to bias JTEN, JTTCK, and JTRST to disable JTAG.

The microcontroller also required circuitry largely decoupling capacitors and current limiting resistors. Figure 3.3.4 illustrates the circuitry associated with the PIC18LF6722.

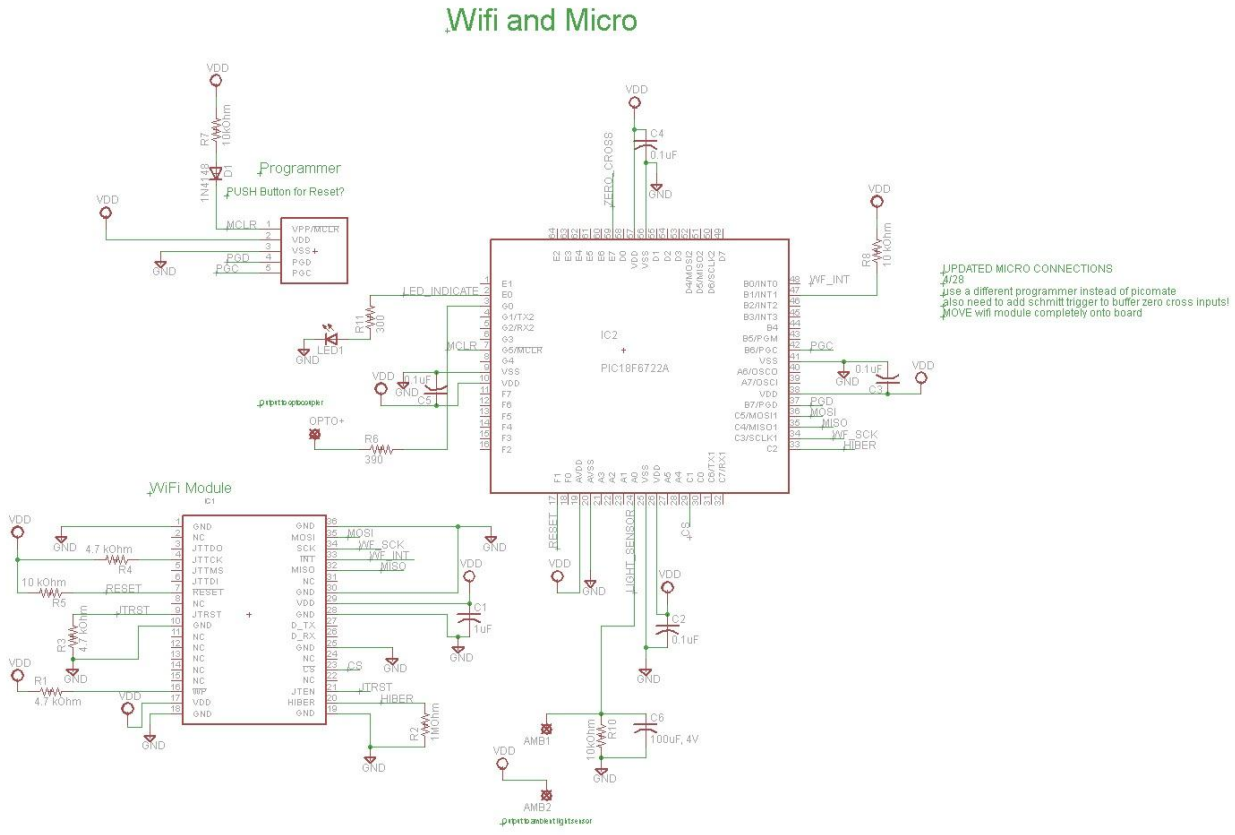

Figure 3.3.4 Schematic of PIC18LF6722

C2, C4, C3, C5 are all decoupling capacitors. R8 is the pull up for the interrupt line mentioned previously. R6 is a current limiting resistor for turning on the optocouple on the flyback board where the TRIAC dimming circuitry was located. Then there are the signals associated with programming the microcontroller including VPP/MCLR, PGD, and PGC. These signals run between the programmer and microcontroller. R7 and D1 are for protection of the 3.3V supply. When programming the microcontroller power supply must be on, meanwhile Vpp can potentially get as high as 14 volts. For this reason, it seemed prudent to add the diode to prevent current from flowing from Vpp to Vdd.

Beyond the hardware associated with communication there is also the software necessary to connect and communicate. The code used is a modified version of Demo App available in the Microchip TCP/IP Stack v5.42.06. This demo app is made for a number of different pieces of hardware including the PICDEM.net 2 board initially used by the group for development. The configuration intended for the PICDEM.net 2 board was edited to be used with our final board. This involved editing the HardwareProfile.h and TCPIPConfig.h files specific to the PICDEM.net 2 board and editing the WF_Config.h, WF_Eint.c, WFAPI.h, MainDemo.h, and MainDemo.c files common to all configurations.

The hardware profile contained a number of #define statements used throughout the TCP/IP stack. For example, it contained definitions for all the I/O pins associated with controlling the transceiver from the microcontroller. It also contained the control registers and buffers associated with SPI. Many of these definitions had to be change to match the hardware in our system as opposed to the original configuration on the development board.

Connection to a router requires definition of the name (SSID) of the network, the security protocol to use, and the kind of network. These items are defined in the WF_Config.h file. For this network, the SSID was JYURKOVICH-VAIO_Network_1, the security was open, and the network type was infrastructure. These are the only changes made to WF_Config.h. This information allows connection to the network as long as DHCP(Dynamic Host Configuration Protocol) is enabled. Enabling DHCP involved defining STACK_USE_DHCP_CLIENT in the TCPIPConfig.h file. Using DHCP, the transceiver is able to acquire an IP address from the router as well as properly configure itself for this network. Also in the TCPIPConfig.h file, the definition GENERIC_TCP_CLIENT_EXAMPLE is important because the generic tcp client example was the one our getServerValue function is based off of.

The only change made to WF_Eint.c is in the initialization function placing the priority of the Wi-Fi external interrupt into the low priority interrupt service routine. The change made to WFAPI.h is also because of a wish to place the external interrupt in the low ISR. The definition for WF_DEBUG needed to be commented out otherwise the linker would throw an error. In the final implementation of the code this line and the removal of debug strings from the Wi-Fi is not a large problem because there is no UART to assert issues from. The changes to MainDemo.h include the removal of unused functions from the header. The most extensive changes are in the MainDemo.c and the created getServerValue.c files which are discussed in greater depth later on.

As discussed earlier, the bulb connects to the router via DHCP and involved the inclusion of a definition and making sure that the Wi-Fi configuration file points towards the correct network. With the light bulb

on the network, it is then necessary to acquire values from the server. Communication with the server involves sending the get function to the proper IP address over the Transmission Control Protocol (TCP). The demo app contains an example of communication over TCP that performed a Google search and printed the results to a serial COM. This example was adapted for our purposes and became the getServerValue function.

This function uses a state machine where different states are evaluated. It is of note that when the function is called in main only one state is evaluated at each function call. In the first state, SM_HOME, the client (light bulb) tries to connect a socket between the client and the server then advances into the second state if a valid socket is assigned. The second state, SM_SOCKET_OBTAINED, checks if the server accepted the connection request. If it does not accept after five seconds then the state machine returns to SM_HOME. If accepted, the following string is placed in the TCP buffer to be sent to the server

GET /led/actions.php?fcn=get&id=1 HTTP/1.0
Host: 192.168.1.2
Connection: close

The microcontroller then advances into the SM_PROCESS_RESPONSE state. It stays in this state until the socket is closed. In this state it is waiting for a response from a server in the form of a string. This string contains the values associated with id=1 on the server and then some other information regarding the state of the connection. Figure 3.3.5 contains an image of this output returned string.

```
{"id":"1","dimmingValue":"72","RValue":"79","GValue":"166","BValue":"72","WValue":"0"}HTTP/1.1 200 OK
Date: Thu, 21 Feb 2013 16:18:54 GMT
Server: Apache/2.2.22 (Win32) PHP/5.4.3
X-Powered-By: PHP/5.4.3
Content-Length: 86
Connection: close
Content-Type: text/html
```

Figure 3.3.5 Returned string from server

It is then necessary to parse the number directly after “dimmingValue”. The microcontroller gets an array out of the receive buffer on the transceiver and then parses it. It does this by looking for the characters ‘n’,’g’,’V’ in that order as this is specific to the string dimmingValue. Then it advances until it finds ‘:’ and ‘””’. Once it finds those characters the next character is the most significant digit of the dimming value. It places this digit and any subsequent character that is between ‘0’ and ‘9’ into an integer array until it reaches the non digit ‘””’ at the end. Then it throws a flag saying it has found the number and leaves the loop involved in parsing. From the integer array and the counter saying how many digits were found in the number, a dimming value is constructed. This value is then passed out of the function by reference and the rest of the receive buffer is discarded. Once the remote node has disconnected the next state SM_DISCONNECT is selected. In this state, the socket is closed and the SM_DONE state is entered, which returns to the SM_HOME state. Figure 3.3.6 shows a flow chart of the states.

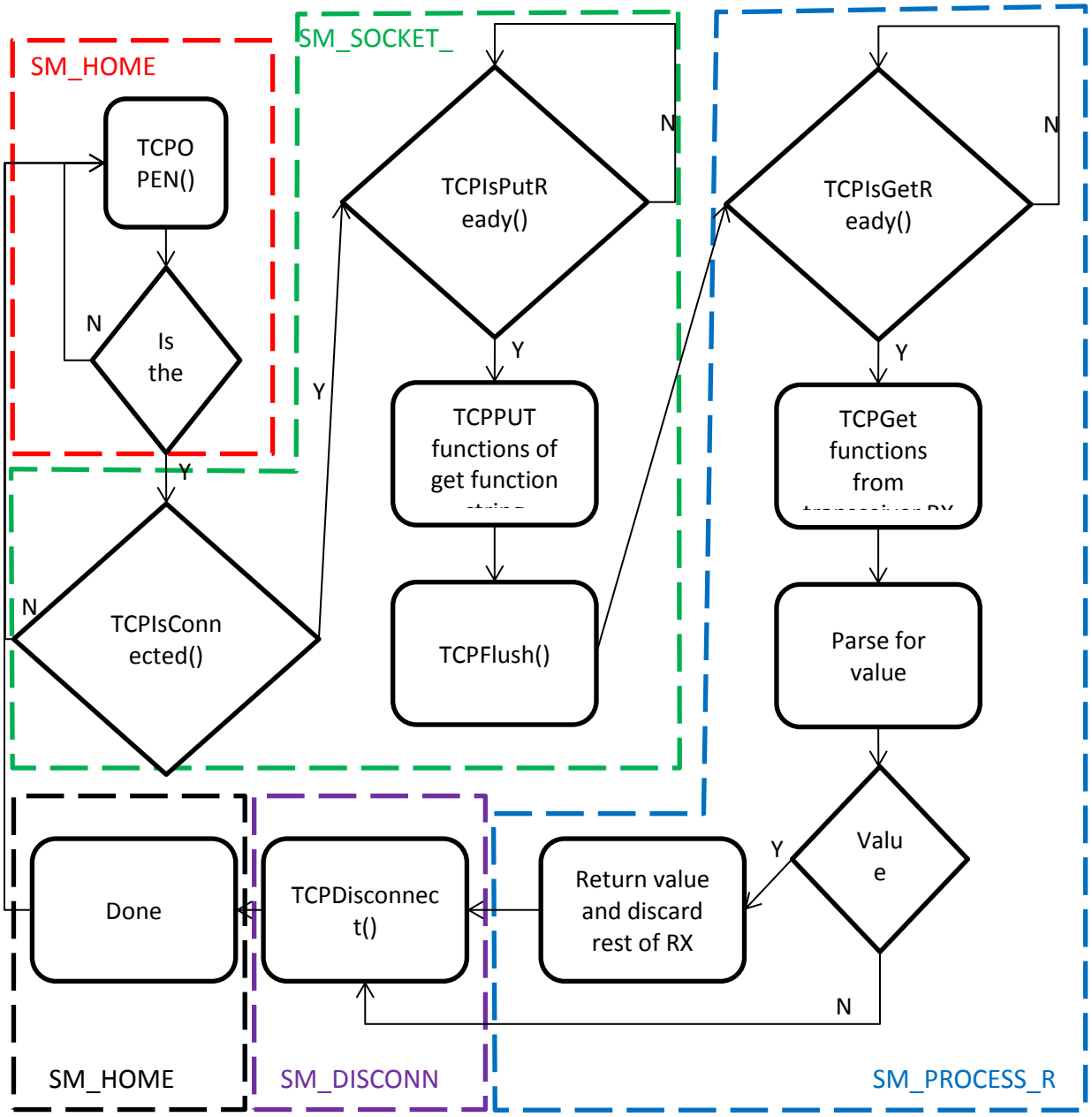

Figure 3.3.6 Flowchart of getServerValue

With these two things in mind the overall main works as follows. First, the board is initialized for a specific clock speed, turning off the A/D converter, enabling interrupts, and the modules necessary for the dimming. Then the timer necessary for Tick.c (TMR0) is initialized. Then the AppConfig structure, which contains the address information such as IP and MAC addresses, is initialized. The stack is initialized next, which resets and begins communication with the transceiver. Then connection to the network is attempted. After trying to connect the code enters an infinite while loop. This loop performs stack tasks, Wi-Fi tasks, and the getServerValue function. Upon exiting the getServerValue function the dimming value passed by reference is evaluated and is mapped to a time delay for the TRIAC pulse using a look up table. This is the extent of communication. Communication also requires two interrupts both

placed in the low priority. One is for the Timer0 counter relevant for certain timing applications of the TCP/IP stack. The other is for the external interrupt from the transceiver. As explained before the transceiver will pull this external interrupt low when it needs to communicate with the microcontroller.

Whether the subsystem was functioning in its final iteration or not was done by checking the devices connected to router. A transceiver would have an IP address, but no name. Beyond that testing of the subsystem was done in conjunction with testing of the dimming and seeing if the pulse for the TRIAC moved with changing the dimming values.

3.2.2 Dimming Control

In creating a pulse to turn on the TRIAC and dim the bulb, there were several requirements necessary. First, it was necessary to detect zero cross's coming from the Schmidt trigger. Then on detecting a zero cross an accurate delay time needed to be counted before setting a pin high on the microcontroller for over 180 microseconds. This delay had to be fairly constant otherwise the bulb would flicker. After over 180 microseconds the pin then needed to be set low before the next zero cross.

In its final iteration, this was implemented with a timer (Timer1) and two capture/compare/PWM (CCP2, CCP3) modules. One of the CCP's (CCP2) was set to capture and the other to compare (CCP3). In capture mode, on an event such as the falling edge or rising edge of a capture pin (E7) the value in a designated timer's counting registers (TMR1H/L) is saved in the CCP's value registers (CCP2H/L). This essentially saves the time of an event. In compare mode, when the CCP's value registers (CCP3H/L) equals the value in the designated timer's counting register, the compare module will drive its output pin (G0) high or low. This depends upon configuration. Using this it is possible to implement the requirements stated above.

Referring back to Figure 3.3.4, the optocouple output in is G0, which runs to a current limiting resistor. Meanwhile E7 is the input from the zero cross detection. The zero cross detection is actually an inverting Schmidt trigger which is powered by Vdd and Gnd. Its input is the 60Hz AC line through a 1.87Mohm current limiting resistor. Figure 3.3.7 contains the optocouple and TRIAC found on the LED supply board. When G0 goes high it allows current to flow through the diode in the optocouple. This then allows the TRIAC to turn on. Meanwhile R10 is a current limiting resistor. R11 and C8 are part of a snubber.

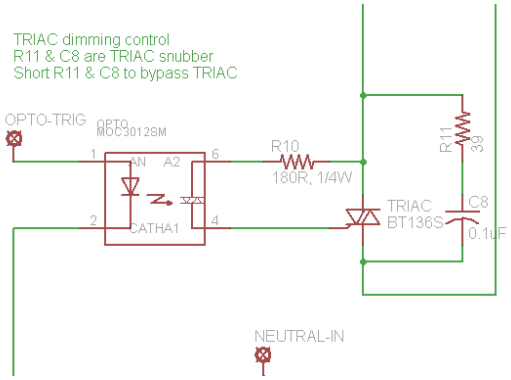

Figure 3.3.7 Optocouple and TRIAC

The overall operation is that timer 1 is initially set to run continuously. The capture module is set to detect a falling edge and the compare module is turned off. On the detection of a falling edge the capture module saves the time and throws an interrupt flag. When this interrupt is addressed, the high priority interrupt service routine (ISR) adds the value saved in the capture module register to the necessary delay time and places it in the compare module's value register. Then it turns on the compare module, clears the interrupt flag, and resets the capture module so that it will now detect on a rising edge instead of a falling edge. The code then exits the ISR and continues to execute the main loop. When the compare module's value register equals the timer 1 counter register it causes the compare output pin to be driven high and throw an interrupt flag. In the ISR, timer 1 is turned off and cleared. Then the compare module is changed so that it will throw the output pin low on a compare and the value placed in the compare value register will cause a pulse of at least 200 microseconds. Then the timer is turned back on and the interrupt flag cleared. When the compare triggers again it drives the pin low, turns off the timer, clears the timer registers, and disables the compare module. The interrupt flag is then cleared and the timer turned back on. This process then repeats on the next zero cross ad-infinitum. There are two special cases in the operation of the pulse generation. If the dimming value corresponds to off then a flag called firingPin is assigned to zero in the main loop. After the firing pin is pulled and the capture module throws an interrupt the compare module will be disabled inside the ISR and the output pin to the optocouple never goes high. The other case is if the dimming value corresponds to full brightness. Then in the ISR on the throwing of a capture interrupt the compare module output pin is forced high in the capture iteration of the ISR. In the same iteration of the ISR the compare is then configured to pull the pin low after a certain period of time and the rest of the steps function in the same fashion.

Changing the delay time only requires that one change the value added to the capture value. This is done in the main when a dimming value is acquired. After a call to getServerValue() the dimming value is checked for certain conditions. If the dimming value is zero then the firing pin is removed (set to 0), the intermediateValue is set to zero, and the current system time from Timer 0 is acquired. If it is not zero then there are two options. One, the previous value found in intermediateValue is zero and the bulb must be thrown above a threshold for a period of time before moving towards its final brightness. Two, the dimming value is being changed from one brightness value to another without turning off the bulb. The reason for a threshold in the first case is due to a deep dimming mode found within the flyback converter IC. If the bulb is turned off then turned on below threshold it is thrown into a deep dimming state. In this state, the lower dimming values all below threshold will all appear to have the same brightness. Also because the IC in the LED supply was intended to be used with a normal dimmer switch (and not something capable of jumping from one delay time to another) it was decided that it was best to increment dimming levels when changing brightness. Changing brightness would occur either when changing from one brightness level to another, or when being turned on. In the latter case, the level was first thrown above threshold, and then moved up or down towards the proper brightness. This means if the server gets a value of 50 when previously having a value of 100 that it steps through the values in between to provide a better dimming characteristic. This incrementing is done by moving the previous value stored in intermediateValue towards the new value found in dimmingValue; incrementing only happens once a zero cross due to a flag in the ISR. The changing intermediateValue is

mapped to a delay time using a look up array of numbers. The number in the array corresponds to the values added to the capture registers (CCPR2H/L) and placed in the compare registers (CCPR3H/L). Since zero does not contain a value in the array, the array has 100 entries. To index the correct value all that is necessary is to subtract one from the intermediate value. This implementation resulted in a pulse that would only change by about 50 microseconds in reference to the zero-cross.

The testing of subsystem was done in conjunction with the communication

3.3 Power Electronics

The realm of power electronics pervades two of our project's critical subsystems. Primary challenges for both subsystems included producing adequate regulation and efficiency in a low-cost, small form factor solution. To develop the most satisfactory solution, it was necessary to reach a broad understanding of the available technologies. The design process is described in detail in section 4.3.

3.3.1 Flyback Converter

The flyback converter is responsible for powering our LED. Its design depended heavily on our choice of LED: a Cree CXA1512 array. This single package incorporates a number of LEDs in series, such that its nominal operating conditions are a forward voltage of 37V at 350mA. This translates to a nominal power output of 13W. Thanks to the high efficacy of this LED, the light output approximates a 100W incandescent bulb.

Power Integrations offers phenomenal design resources for LED power supplies, in addition to a line of PWM controllers with an integrated power MOSFET. These controllers drastically reduce the number of components required to operate a flyback power supply. Furthermore, their PI Expert design software assists in designing the required flyback transformer.

An additional requirement for our flyback supply was dimming compatibility. Many integrated LED controllers, such as those that Power Integrations offers, are compatible with TRIAC dimmers. Due to this wide compatibility, this method was chosen to dim our LED.

Figure 3.3.1, below, shows an example application for the Power Integrations LinkSwitch-PL controller. Following is an overview of the normal operation of the circuit.

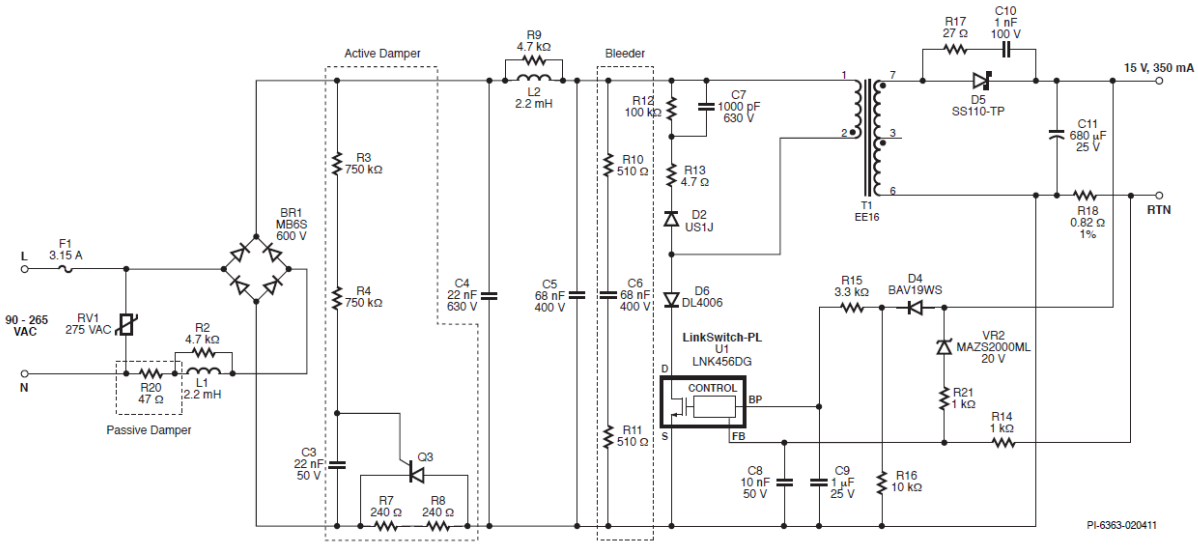

Figure 3.3.1 LinkSwitch-PL example application with active/passive dampers and bleeder incorporated

The input for this particular example is a mains connection at 60Hz. BR1 performs full wave rectification, and with C4, L2, and C5, a slightly smoothed AC waveform is presented to the primary of the transformer. C4, L2, and C5 are arranged as a pi-filter, which serves the purpose of reducing conducted EMI. L1 also works to this end.

The dotted primary terminal is connected to the drain of the LinkSwitch-PL's internal power MOSFET. When this switch conducts, magnetic flux is stored in the core of the transformer. Because of the winding polarity, no current flows on the secondary at this time. Immediately upon the turn-off of this switch, however, the voltage across the secondary terminals reverses and forward-biases D5 (which is typically a Schottky type for higher efficiency). Filtering on the secondary is provided by the bulk capacitor C11, which is connected across the LED load. Output voltage is determined by the turns ratio of the transformer and the duty cycle of the controller. R18 produces a voltage proportional to the LED current; this signal is applied to the controller's feedback pin through a LPF (R14, C8) to achieve constant current through the LED.

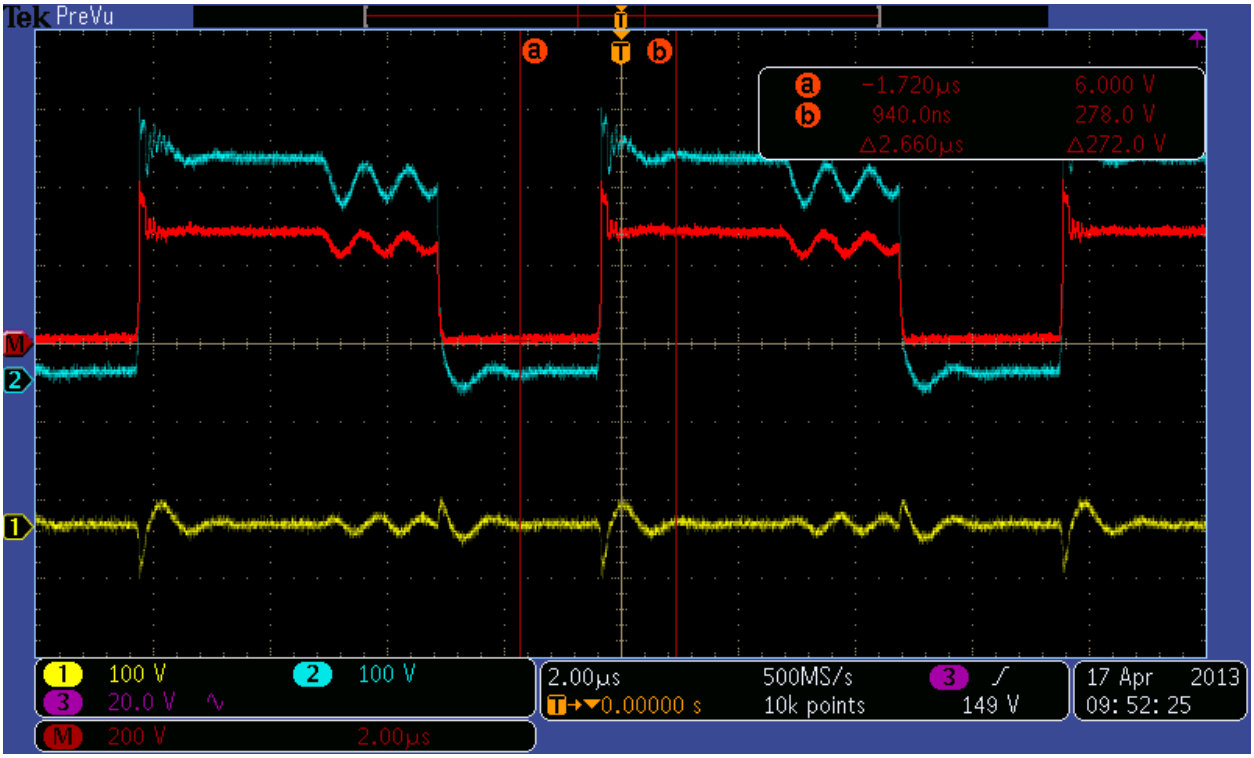

Figure 3.3.2 Voltages on the power MOSFET within the LinkSwitch-PL. Yellow is source, Blue is drain, and red is V_{DS}. The switch operates near 100 kHz at about 60% duty cycle.

The LinkSwitch-PL requires no external regulator. When its internal MOSFET it off, it draws a small amount of current from the drain pin to create a ~6V regulated supply at the bypass (BP) pin. During MOSFET conduction, the voltage at the drain pin is near-zero and the IC is powered completely by C9, a 1uF bypass capacitor. Zener diode VR2, in conjunction with R21, creates overvoltage protection. If the output of the supply exceeds the reverse breakdown voltage of the zener diode, a much larger than normal feedback voltage is applied to the feedback pin and the IC enters an auto-restart protection mode.

Two additional blocks are included to dampen switching transients. The primary side of the transformer contains a small amount of leakage inductance (for our transformers, Power Integrations measured this parameter at <8.0uH). Due to this inductance, a large transient is produced at the output when the MOSFET current abruptly falls to zero. This example design remedies the transient with an RCD-R snubber, but our design instead utilized a 200V unidirectional TVS diode to lower component count. Furthermore, R17 and C10 dampen transients associated with D5.

A number of additions are required to make this example power supply TRIAC dimmable. Firstly, because the controller regulates the output based on a feedback loop, it does not inherently respond well to phase-chopping of the AC input. Smooth dimming is accomplished through the detection of zero crossing at the tail end of every half-cycle. The controller adjusts its feedback threshold (typically 290mV) in proportion to the conduction angle of the input waveform. Section 3.3.2 discusses TRIAC dimming in further detail.

In addition to zero cross detection, it is necessary to ensure that enough primary-side current is drawn in order to meet the holding current requirement of the TRIAC. For incandescent loads, this is never a point of concern due to their high power consumption. However, to deliver 13W to an LED load, the average primary current drawn is only ~10mA. Some TRIACs have a holding current requirement above this level. The boxed circuit sections—the active damper, passive damper, and bleeder—improve performance with TRIAC dimmers. Furthermore, a BP supply bias network is included to feed the BP supply from the output voltage.

Firstly: The active damper increases primary current consumption at the beginning of a half-cycle, where the current drawn would otherwise be the smallest. After a time delay set by R3, R4, and C3, the SCR is triggered and shorts out resistors R7 and R8. At this time, the primary current must be sufficiently large to meet the holding current requirement and thus maintain TRIAC conduction. The bleeder continuously consumes a small amount of current, but its primary function is to collect inrush current and thus ensure that the TRIAC holding current requirement is met immediately upon firing. Finally, the passive damper causes increased current consumption throughout every half-cycle. It is particularly helpful at the tail end of a half cycle, where primary current is once again likely to fall below the TRIAC holding current requirement.

A good way to understand the functionality of the active and passive dampers is to consider the input voltage that is dropped across them. Then, holding constant the power delivered to the primary of the transformer, more current is required as the voltage available to the primary terminals is decreased. Naturally, the addition of these TRIAC compatibility blocks lowers efficiency.

Finally, D4 and R15 help maintain the BP supply voltage on C9 during deep dimming. At very small TRIAC conduction angles, the LinkSwitch-PL IC must operate for long intervals with no current available from its drain pin. R15 is selected such that 1-2mA flows into C9 at the minimum output voltage. C9 may also be increased to ensure BP supply operation during deep dimming.

Our final flyback topology makes minor adjustments to this example application. We omitted the LPF on the feedback pin after finding that adequate LED regulation was achieved without it. Furthermore, by choosing a TRIAC with low holding current, we were able to shorten the timing of the active damper and use less damping resistance.

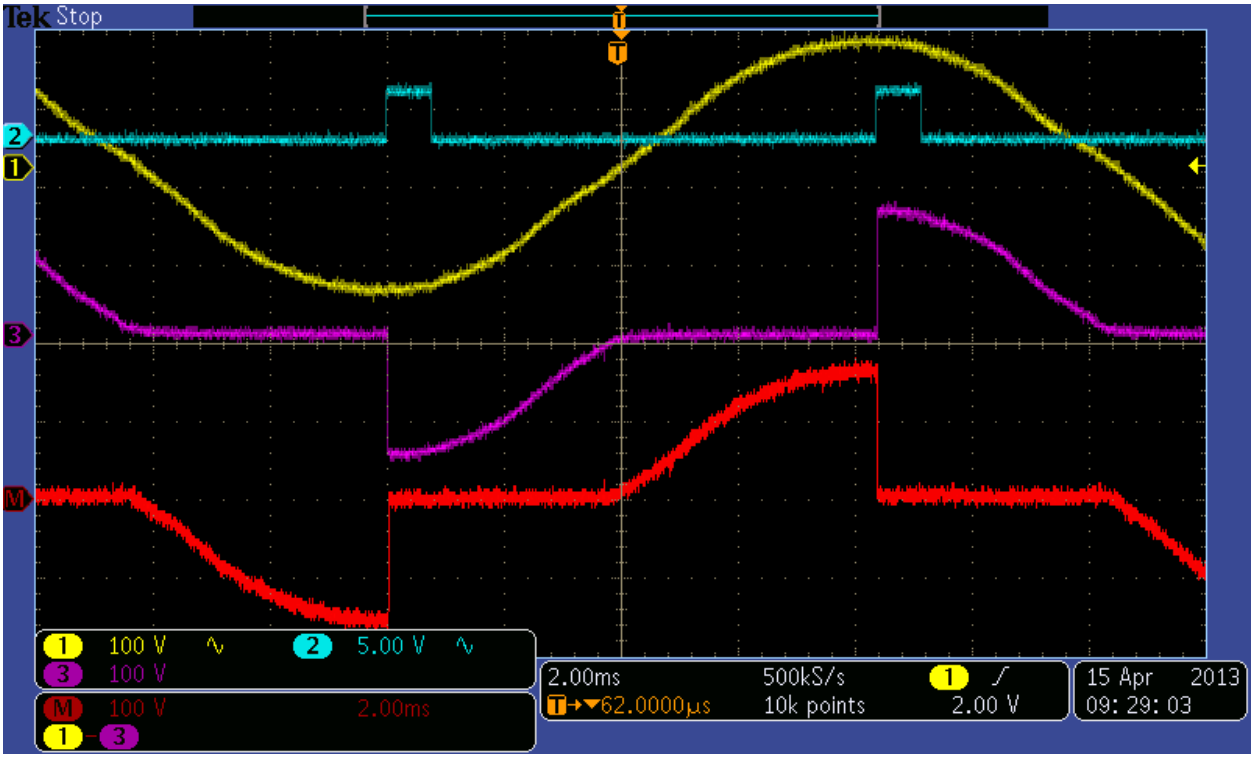

Figure 3.3.3 Scope output showing AC input (yellow), TRIAC firing pulse (blue), voltage at input of flyback power supply (magenta), and a math function calculating the voltage across the TRIAC (red).

3.3.2 TRIAC Dimming

The use of a TRIAC to achieve dimming is an unfortunate relic from the days of incandescent bulbs. For compatibility reasons, many LED bulbs sold today support TRIAC dimming despite its negative implications for efficiency and power factor. Seen as a unit, our bulb is not strictly compatible with an external TRIAC dimmer; internally, however, a TRIAC is used to accomplish leading edge dimming.

As described in section 3.3.1, this LinkSwitch-PL application includes a number of features that enable TRIAC dimming. The LinkSwitch-PL adjusts its feedback threshold in a manner that is proportional to the delay time from zero cross to TRIAC firing. However, both upper and lower thresholds exist which constrain the dimming range. **Figure 3.3.4**, below, illustrates the waveforms associated with leading-edge TRIAC dimming.

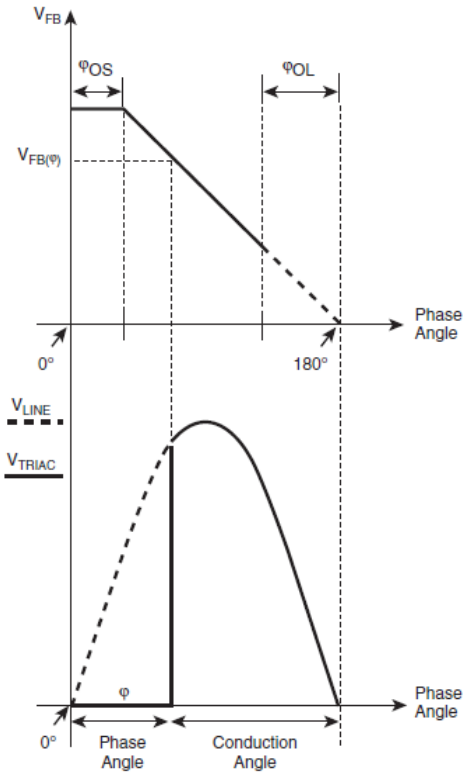

Figure 3.3.4 Modes of LinkSwitch-PL operation when used with a leading edge TRIAC dimmer

Because the LinkSwitch-PL datasheet did not provide explicit information about dimming behavior, it required characterization. A test setup was constructed in which a conventional leading-edge TRIAC dimmer was placed on the mains line and its output was connected to our prototype supply. A DMM probed the LinkSwitch-PL's feedback voltage while an oscilloscope measured the Conduction Delay (the time-equivalent of the Phase Angle in **Figure 3.3.4**).

It can be seen from this graph that the LinkSwitch-PL contains an unfortunate lower limit in dimming ability. At roughly 6.7 ms conduction delay, the feedback voltage is 30mV. Based on our feedback resistor choice of 0.825 Ω, the LED current at this point is approximately 35mA. If the conduction delay is increased any further, the supply exits its mode of normal operation and assumes a state of deep dimming. For smooth dimming, we wished to avoid this state.

The lower dimming limit of this supply was expected, however, due to TRIAC holding current requirements. For an LED current of less than 35mA, a relatively high amount of power must be wasted to ensure that the TRIAC holding current (~15mA on the *primary*) is maintained. We found that the dimming range of the LinkSwitch-PL was perfectly compatible with the dimming range of the

conventional dimmer that we were using for our testing (i.e., the conventional dimmer also possessed a lower dimming limit).

An important point of improvement that our design makes over a conventional dimmer is its ability to dim linearly with respect to perception of brightness. To do this, it was necessary to employ Steven's power law. It is used to establish a relationship between a physical stimulus and the subjective magnitude of the sensation evoked by the stimulus. This relationship takes the form of a simple exponential function:

$$\psi(I) = kI^a$$

where I is the stimulus and $\psi(I)$ is the perceived sensation. The exponent, a, is between 0.33 and 0.50 based on Steven's data. From this, the following equation is derived:

$$y = kx^{1/a} + c$$

This equation is fit such that y is the feedback voltage for a given arbitrary brightness index x which may assume any integer from 1 to 99, inclusive. Constants k and c are used to achieve this, and a is Steven's exponent. If we let x go from 1 to n and impose upper and lower feedback voltage limits $V_{fb,low}$ and $V_{fb,high}$, the following expressions are obtained:

$$k = \frac{V_{fb,high} - V_{fb,low}}{n^{1/a} - 1}$$

$$c = V_{fb,low} - k$$

Using the fit obtained in Figure B7, the feedback voltage equation y is translated to conduction delay times for every x. These delay times are then rounded such that they may be represented in microcontroller timer units, which for our application are 250 ns long. Figure B8, below, shows the culmination of these calculations for $a = 0.5$.

3.3.3 3.3V Supply

To supply power for our onboard electronics (microcontroller, Wi-Fi module, etc.), it was necessary to include a low-voltage supply that featured very tight regulation. Though it would have been possible to piggyback such a supply on the 37V LED supply, a switching converter would be required to achieve reasonable efficiency. Thus, not much would be gained from piggybacking the 37V supply. In addition, the 3.3V supply would not be isolated from the flyback supply and it would have been impossible to implement the 3.3V supply independent of the flyback supply. This line of reasoning led to the decision to convert 115VAC directly to 3.3VDC @ 200mA.

Rather than use a bulky and relatively expensive transformer to reach 3.3V, a switching converter in the buck topology was implemented. This topology requires only a few inexpensive components, thanks to another solution from Power Integrations. Typically, buck topologies in which the input voltage is many times larger than the output voltage require large inductors and capacitors. This requirement is necessitated by the fact that, in a buck topology, only a switch and inductor separate the input and

output. The following equation illustrates this issue in more explicit terms:

$$I_L = \frac{1}{L}\int_0^t V_L\, dT = \frac{V_L t}{L}$$

where I_L and V_L are the inductor current and voltage, respectively, and t is the time that the switch remains on. It is assumed that the inductor current is equal to zero at $T = 0$ and increases until the switch is turned off at time $T = t$. Let V_L remain constant at 120VDC. Unless L is very large, the inductor current reaches the ampere range in a matter of microseconds.

To remedy this phenomenon, the LinkSwitch-TN buck controller IC uses a novel but surprisingly simple control strategy. Rather than implement PWM (as is typical for a buck converter), the LinkSwitch-TN uses simple on-off control and imposes a current ceiling for its integrated power MOSFET (which serves as the main switch of the buck converter). This current ceiling is approximately 400mA. **Figure 3.3.5**, below, illustrates this control method.

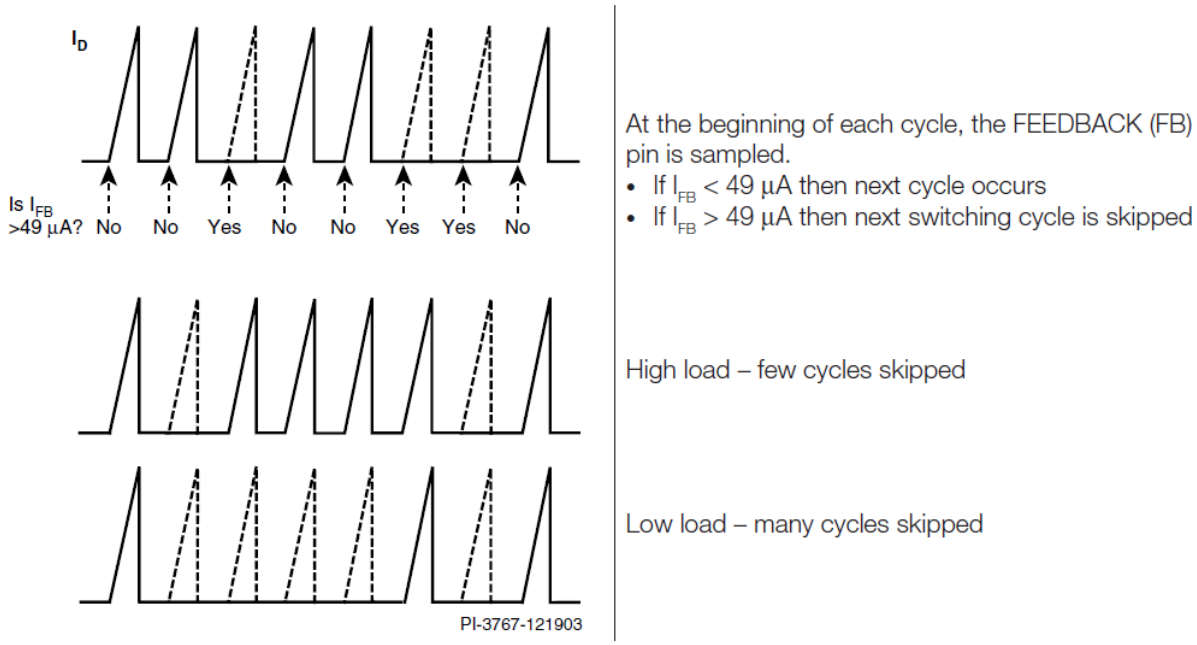

Figure 3.3.5 Illustration of LinkSwitch-TN current control scheme

Because the maximum switch (inductor current) is only 400mA, a smaller filter capacitor on the output may be used. **Figure 3.3.6**, below, shows our completed 3.3V supply schematic. It is followed by an overview of supply operation.

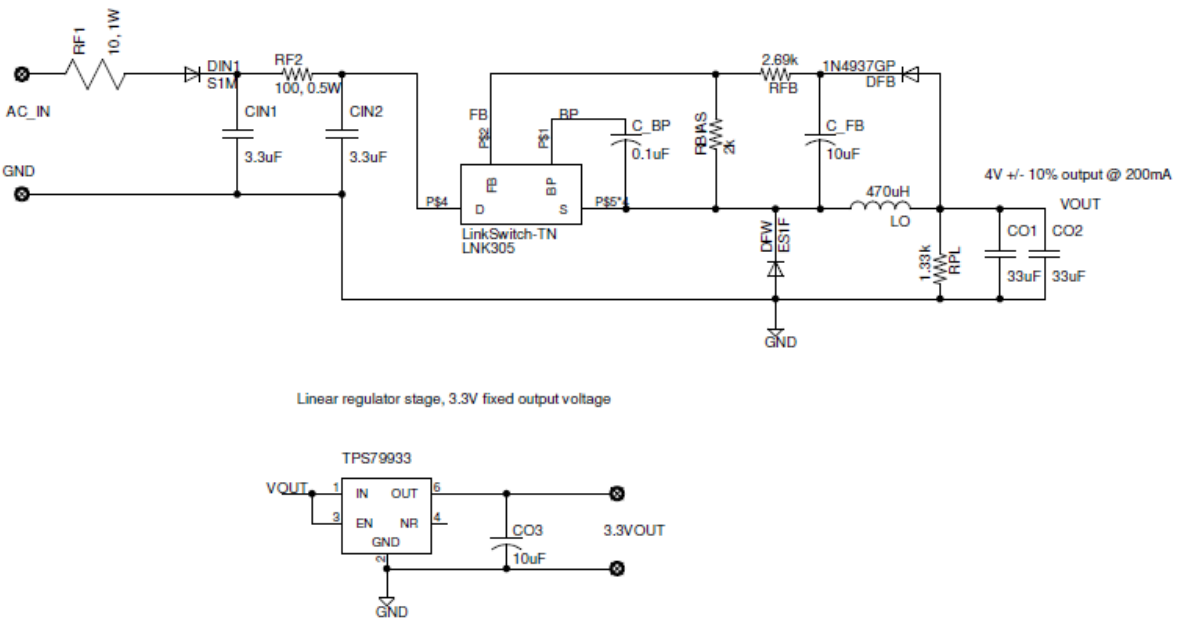

Figure 3.3.6 Final low-voltage supply circuit, with additional linear regulator at output of LinkSwitch-TN circuit

115V_{AC} mains is applied to the AC_IN terminal of this supply. An input network consisting of RF1, RF2, Din1, Cin1, and Cin2 perform half-wave rectification and provide sufficient capacitance to smooth the rectified input to a minimum of ~90VDC at full load. RF_1 and RF_2 are of the fusible type and will fail safely in the case of a fault. Similar to the LinkSwitch-PL chip, the LinkSwitch-TN requires no external supply. It uses an internal regulator to establish ~6V on its BP pin and C_BP. The drain and source terminals of the LinkSwitch-TN's power MOSFET are labeled on its symbol. When this MOSFET conducts, the rectified and filtered input voltage is applied to Lo, the buck converter inductor. Its current increases until the MOSFET reaches its current limit and turns off. Due to the abrupt change in inductor current, its voltage polarity reverses immediately and forward biases Dfw, the freewheeling diode. The freewheeling diode allows the stored inductor energy to be transferred to CO1 and CO2, the output filter capacitors. Using two output filter capacitors achieves lower overall ESR and thus smaller switching transients seen at the output.

Feedback is achieved through the Rbias, Rfb, Cfb, and Dfb components. The LinkSwitch-TN FB pin sinks a nominal current of 49uA at a voltage of 1.65V when the supply is within regulation. Rbias serves the purpose of biasing Dfb and is typically 2kohm. Under these conditions, Rbias is selected to set the output voltage. Figure B6, below, shows this calculation.

As illustrated in **Figure 3.3.6**, whenever the output voltage rises above the nominal value, the voltage on the feedback pin also increases and more than 49uA flows into the FB pin. The IC then skips cycles until this 49uA threshold is no longer exceeded.

3.4 Packaging

In order to present our prototype best, we designed packaging for the electronics and the heatsink in Google SketchUp 8 (see **Figure 3.4.1** and **Figure 3.4.2**). The heat sink had an 11 mm diameter hole in the bottom that we used to attach the packaging to it. This was accomplished by creating a pole approximately 10 mm in diameter that would fit into the hole of the heatsink; it was attached with epoxy. The PCBs sit on four small ledges evenly spaced around the interior of the packaging and attached with hot glue. The main housing itself was designed in two pieces that fit together in only one way.

This prototype packaging was developed in two stages, the first being created on a MakerBot 3D printer (see **Figure 3.4.3**). This enabled us to inspect the design and determine the necessary changes that would enable the final prototype to meet our design specifications. The MakerBot Replicator is an entry-level 3D printer, printing much more cheaply and less accurately than other 3D printers that the College of Engineering has. Additionally, it takes far less time to print, as it operates at a much lower level or precision. With these characteristics, we used this machine to print our housing before taking it to the slower and more expensive machines, to ensure that we could catch any errors. This ended up being a good idea, as there were several small errors in our CAD files that were nearly impossible to catch before printing. We were able to make the necessary adjustments to these files before the final print. The process for printing on the Replicator is fairly straightforward, and we followed the directions posted on the 3D Design Deck website, with some minor changes (http://www.n3d.nd.edu).

The first step was to convert the CAD files from SketchUp format to a stereo lithography (.stl) file, using a SketchUp plugin downloaded from guitar-list.com. This file was opened using the ReplicatorG software that accompanies the 3D printer, and converted to usable instructions (G-code) via an export process within the program. The parameters set during this export can greatly affect the quality and speed of the print, and we were sure to enable supports while printing. Because our packaging design had components that extended away from where the 3D printer would lay down the base layer, additional material needs to be printed under where these extensions occur. This is because the printer extrudes plastic in a liquid, and without these supports those extension pieces would fall and ruin the piece as they were printed.

Our second prototype model (see **Figure 3.4.4**) was created on a Stratasys Fortus 250mc 3D printer. It used a total of 5.81 in^3 of material (4.24 in^3 for the model, the remaining 1.57 in^3 for support) and ran for 7 hours and 5 minutes. The Cree LED array was attached to the top of the heatsink with thermal epoxy. The protocol for the Fortus printer was almost identical to that of the MakerBot, except that it took much longer to print. One additional advantage of using the Fortus printer was the much cleaner removal of support structures. Both 3D printers utilizied these structures, but the Fortus machine was able to do so in a different material than the desired product was being printed in. This is then easily removed by rinsing in a light chemical bath, whereas the MakerBot supports are attached to the final product and must be removed by hand. This is not particularly difficult, but leaves a less refined-looking finished product.

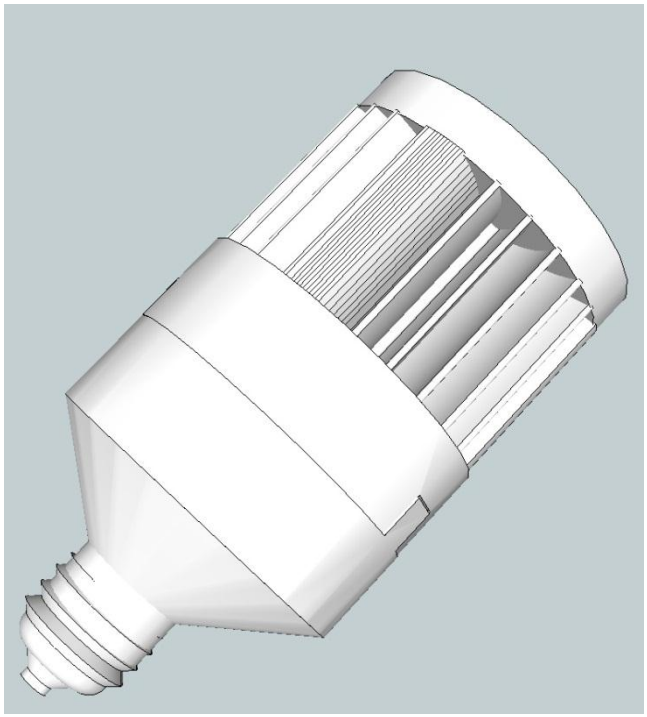

Figure 3.4.1 CAD model

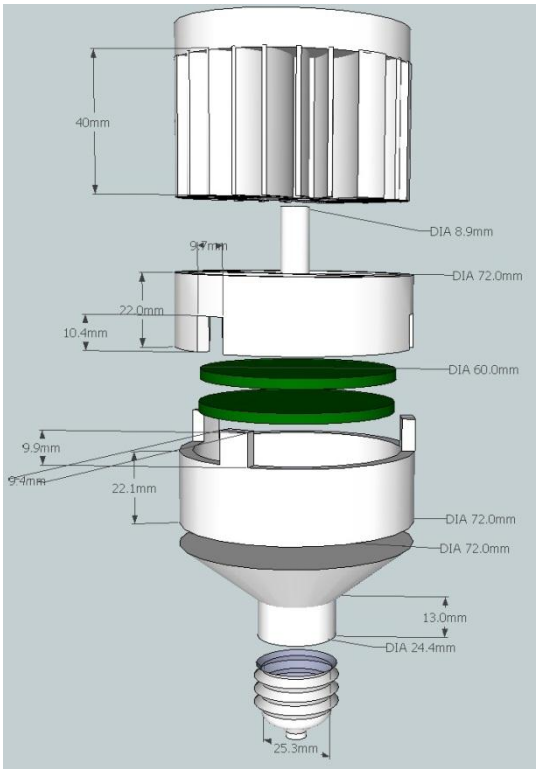

Figure 3.4.2 CAD model with dimensions

Figure 3.4.3 Prototype I

Figure 3.4.4 Prototype II

4 Development (Design Decisions)

4.1 Communication

Using a Wi-Fi transceiver was decided fairly early in the semester. It was narrowed to the Microchip products as well because they would most likely be easiest to interface with Microchip microcontrollers and had example code that would function with both. There was another transceiver that Microchip sold the RN-171 that was another candidate however, this was decided against quickly because it would have been extremely difficult to solder to the board, prohibitively so because of the need for a reflow oven.

4.2 Dimming Control

Initially the dimming control was attempted using two timers and an interrupt on change pin. It was realized when integrating the TCP/IP stack with the interrupt on change this would not work. Interrupt on change occupies the same PORT as the external interrupt used in the transceiver interrupt. It is possible to miss interrupts on change when the PORT is being used for other purposes. This caused the group to try using two external interrupts where one threw a flag when going from low to high and the other threw a flag going from high to low. This worked substantially better than the previous; however, the variation in timing due to other interrupts caused the bulb to flicker which was undesirable. This is why the capture and compare modules were chosen in the final iteration. They severely reduced flickering.

4.3 Power Electronics

4.3.1 Flyback Converter

Our decision to build a flyback converter is not one that we took lightly. We knew that switching power supplies were difficult to design and build, and that it would add significant cost to our project. Careful research is what enabled our success.

From the beginning, our back-up plan was to disassemble a commercial LED bulb and remove its flyback converter for use with our project. This would certainly have been the most economical choice (in terms of both time and money), as designing and building an LED power supply was not an explicit design requirement.

The resources provided by Power Integrations were ultimately what spurred on progress in late March. Their PI Expert software was an enormous resource for our learning and design. It took power supply requirements as input (i.e., our LED load and regulation specifications) and performed some computation to help choose an appropriately sized transformer core. It then suggested a schematic for a flyback power supply, gave transformer build instructions, and even gave a BOM.

Because we were somewhat daunted by the prospect of building our own transformer (transformer cores and construction materials are difficult to find), we contacted the Rapid Transformer Samples staff at Power Integrations. They were eager to help and graciously supplied us with two flyback transformers that had been built to our specifications.

However, our development until this point was not a walk in the park. This decision to use a flyback supply came after weeks of developing a suitable buck converter design in PSpice simulation software. Though we did achieve a working design in PSpice, it was abandoned because of its high component count and susceptibility to implementation issues (this was, after all, a circuit that only worked in *theory*). The beauty of the Power Integrations solution is that they, as a company, have made it their full-time job to maximize reliability, ease of implementation, safety, and cost-effectiveness. With these advantages in mind, the Power Integrations solution was the clear choice.

We constructed a single prototype flyback supply before the final version. This, unfortunately, did not work the first time. It was quickly discovered that Power Integrations sent us an incorrectly wound transformer; it had the primary and secondary sides swapped. This was deduced through a resistance measurement of the windings and the knowledge of their turns ratio and AWG size. In addition, we spent approximately another week trying to increase compatibility with a TRIAC dimmer. To remove uncertainty, we obtained a standard wall dimmer from a hardware store. With some experimentation, we found that our supply required the addition of a passive damper resistor to meet our TRIAC's holding current requirement when firing near the tail end of AC half-cycles. Our prototype is pictured below in Figures 4.3.1 and 4.3.2.

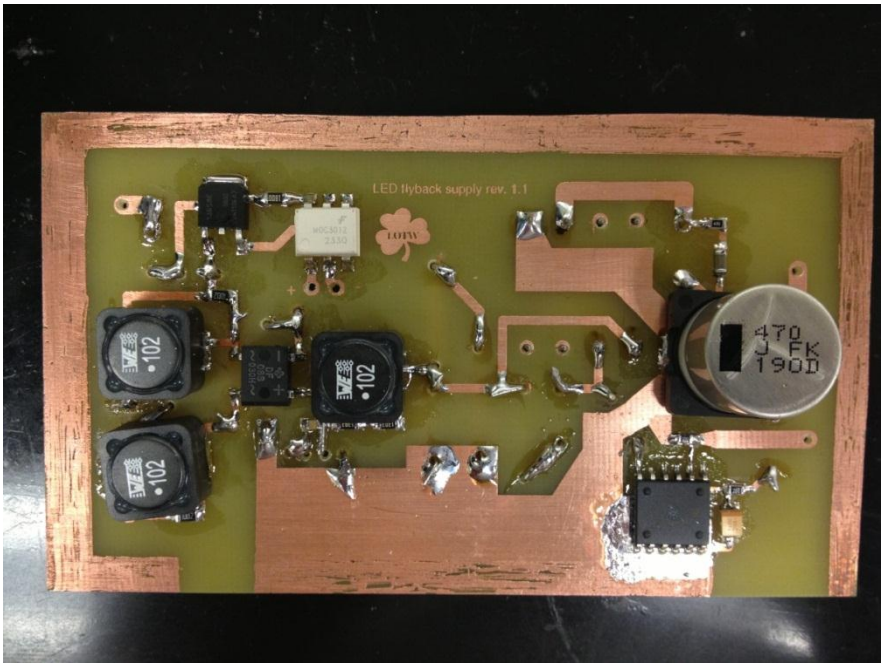

Figure 4.3.1 Revised flyback power supply board prototype (top)

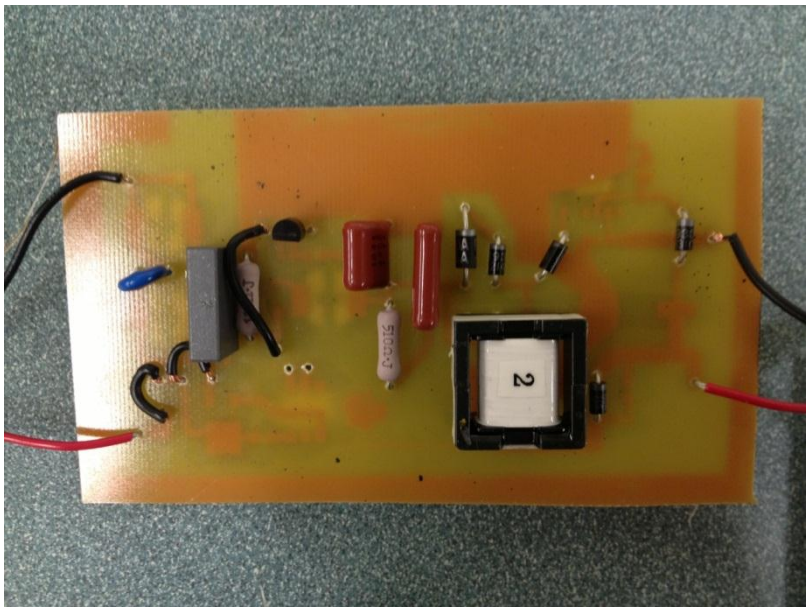

FIGURE 4.3.2 Revised flyback power supply board prototype (bottom)

4.3.2 3.3V Supply

In the first revision of our 3.3V supply in the fall, we employed a (transformer-less) capacitive current limiting circuit. It achieved regulation with the use of a reverse-biased zener diode. While effective, its efficiency at light load was quite poor because any current not consumed by the load was sunk by the zener diode. Furthermore, the current that flows through the limiting capacitor is wasted during each negative half cycle. The original schematic is shown below in **Figure 4.3.3**.

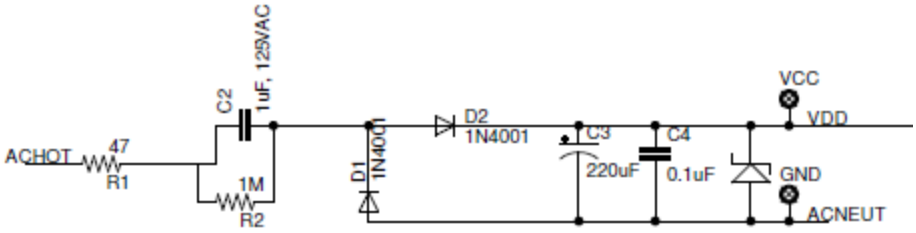

Figure 4.3.3 3.3V supply prototype, featuring capacitive current limiting and zener diode biasing

The greatest advantage of this circuit was its simplicity; it was implemented at extremely low cost and dissipated very low power because its greatest load was the occasional 5mA pulse to fire a TRIAC. However, once the decision was made to add a Wi-Fi module, the current requirements for this supply skyrocketed to 200mA. Though this method of regulation could certainly be adopted to meet that current requirement, it no represented the best design decision.

The search began for a more efficient 3.3V power supply, and we stumbled across the LinkSwitch-TN offering from Power Integrations. The rest of our design evolved from their detailed application notes. Before moving to our final design and layout, we constructed two separate iterations of this supply. They are picture below in **Figure 4.3.4** along with a summary of their development.

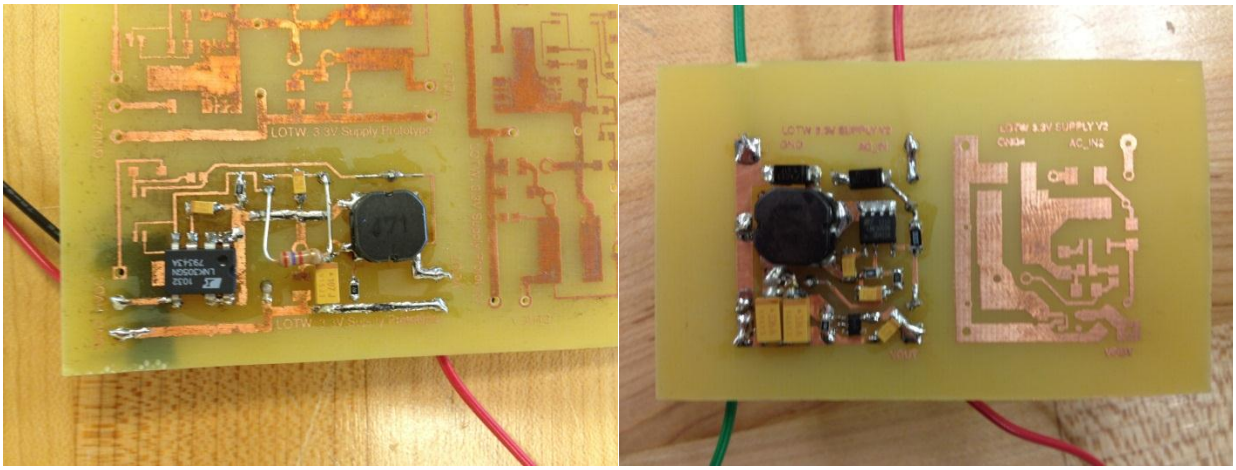

FIGURE 4.3.4 From left: first and second revisions of microcontroller power supply board

The first iteration of the supply appears in the left side of **Figure 4.3.4**. This supply featured a simple and spacious layout. Multiple inductors and capacitors were being considered, and space was provided for each component. We were happy to discover that it worked the first time around, and that these extra component spaces were unnecessary. Our next goal was to greatly reduce the board's footprint, as seen in the next revision of the same supply in the right side of **Figure 4.3.4**. In this iteration, we switched to half-wave rectification to simplify our microcontroller's zero-cross detection circuit. Another large improvement of was the inclusion of a linear regulator. This improved our regulation from ±10% to ±2%. Furthermore, it utilizes two output-filtering capacitors in parallel to reduce switching transients. In our final design, we changed the two high-voltage tantalum capacitors that consume the entire rear of the board to electrolytic types, whose volumetric efficiency is greater with similar performance.

4.4 Packaging

In our setup, the maximum power draw of the XLamp LED is approximately 13W. In their documentation, Cree notes that approximately 75% of this figure will need to be dissipated as heat, or 9.75W. Besides knowing this dissipation, other parameters to selecting a heatsink were the junction operating temperature and the effective thermal resistance between the diode junctions and case. These are 85°C and 2.1°C/W, respectively. The heat transfer away from the device is given by

$$\dot{Q} = \frac{T_J - T_{ambient}}{R_{thermal}} = \frac{85°\text{C} - 25°\text{C}}{2.1\ °\text{C}/W + R_{\text{heatsink}}}$$

To dissipate the 9.75 W produced at 85°C, the thermal resistance of the heatsink must be no higher than 4.1°C/W. We chose the LSB70 heatsink in the 50mm length. The larger, 70mm long heatsink had a

better resistance, of 2.1°C/W, but even with a thermal resistance of 3.2°C/W from the 50mm version, we were able to meet the heatsinking specification with some additional headroom. This allowed us to keep our bulb relatively small, while still being able to dissipate up to 11.3W if necessary.

5 User's Guide

5.1 Installation, Setup, and Troubleshooting

The Android app and webpage were designed to be as easy-to-use as possible. The app itself, once installed on the phone, will automatically connect to the server provided the phone is logged into the same network as the server and bulb. The slider on the app adjusts the brightness of the bulb, while the refresh button will get fresh values from the server. The webpage is accessible from any computer on the same network as the server and bulb from the url: 192.168.1.2/led. The slider on the webpage adjusts the brightness of the bulb, while refreshing the page will get fresh values from the server.

The bulb itself is extremely easy to install. Screwing it into a socket is all it really needs. As a default in case a user does not have a network, the light bulb will go to full brightness. In future iterations, set up would be about as simple as a wireless printer. It would create its own adhoc network that a user could log in to and pass the relevant network information to the bulb to reconfigure itself. This could be done with a simple UI application.

Whether the product is functioning could be ascertained by whether the light bulb is on and capable of dimming using the app or webpage. Those two things would be the best ways of telling if the product is functioning. In terms of troubleshooting, one could check if the light bulb is connected to the router. It is a normal feature in many routers to be capable of pulling up attached devices. A device with an IP address, but no name is most likely a bulb. If a bulb is missing from list of devices it would suggest that the bulb is not connected. Then it might be useful to remove power from the bulb and then place power back on the bulb. This would reset the module. If it did not connect to the router at all then there was some issue with the Wi-Fi module or something it requires such as the 3.3 supply or the microcontroller. It would be ill advised for a user to open a light bulb considering the potential to be exposed to 120 AC or affect circuitry that will eventually be used at such a voltage. Considering it is not an exorbitantly expensive product, such as a motorcycle it might be best for them to not try a DIY fix. However, if the product failed while still under warranty we would like to troubleshoot ourselves what went wrong so as to improve the product. This could be done by potentially providing a new light bulb, assuming proper usage, as long as the broken bulb was returned. This would improve customer satisfaction and provide us excellent feedback in terms of why bulbs were failing prematurely.

6 Conclusions

6.1 Future Improvements

As it is currently set up, the IP addresses for connecting to the server and database are hardcoded into the app and microcontroller. Although this was a problem that we identified, we decided that it was unnecessary to make the needed changes because our demonstration only utilized a single bulb. In the future, we would need to program in a way that the bulb could access a secured network, give itself an ID number (so that multiple bulbs could be used), and create an entry in the database wherein it would store its values. This would most easily be accomplished by setting up an ad hoc network on the WB0MB Wi-Fi module, and communicating with the server in such a way that the user could log into this network using a laptop and pass the bulb the necessary information (such as ID number, name, and network password).

Additionally, a market-ready product would not rely on a free, downloadable server. Although using the Wamp server and corresponding PHPMyAdmin suited our purpose perfectly, it is not practical for our target uses. The correct implementation would require using a server somewhere that is accessible even when not on the same network; this would allow the Android app user to change the brightness of the bulb from any location over a 3G or 4G data network instead of having to be tethered to the same network as the bulb.

Another concern resides in bulbs consuming too much bandwidth in trying to access the database. This may be addressed by creating a bay station that connects directly to the router. The bay station could communicate to the server in a similar fashion to the way the bulbs do presently. Then it can pass information to bulbs via another protocol such as Zigbee. This would eliminate reduced performance of the router as well as configuring multiple bulbs. It would also decrease the cost of individual bulbs, the complexity of the circuitry inside a bulb, simplify the code inside, and possibly cause it to consume less power.

7 Appendices

7.1 Server Code

actions.php

```
<?php

require_once 'Db2PhpEntity.class.php';
require_once 'Db2PhpEntityBase.class.php';
require_once 'Db2PhpEntityModificationTracking.class.php';
require_once 'DFCInterface.class.php';
require_once 'DFCAggregate.class.php';
require_once 'DFC.class.php';
require_once 'DSC.class.php';

require_once 'DimmerModel.class.php';

$db = new PDO('mysql:host=localhost;dbname=led', 'root', '');
$id = $_GET['id'];
$fcn = $_GET['fcn'];


switch ($fcn) {
    case "get":

        $dimmer = new DimmerModel();
        $dimmer->setId($id);
        $dimmers = @$dimmer->findByExample($db, $dimmer);

        if (count($dimmers) > 0) {
            $dimmer = $dimmers[0];
        } else {
            echo 'No dimmer found by ID ' . $id;
        }

        //echo 'Dimmer value was ' . $dimmer->getDimmingValue() . '<br />';

        echo json_encode(array("id"=>$id,
            "dimmingValue"=>$dimmer->getDimmingValue(),
            "RValue"=>$dimmer->getR(),
            "GValue"=>$dimmer->getG(),
            "BValue"=>$dimmer->getB(),
            "WValue"=>$dimmer->getW()));

        break;

    case "set":

        $v = $_GET['val'];
        $v_R = $_GET['val_R'];
        $v_G = $_GET['val_G'];
        $v_B = $_GET['val_B'];
        $v_W = $_GET['val_W'];
        $dimmer = new DimmerModel();
        $dimmer->setId($id);
```

```
        $dimmers = @$dimmer->findByExample($db, $dimmer);

        if (count($dimmers) > 0) {
            $dimmer = $dimmers[0];
        } else {
            echo 'No dimmer found by ID ' . $id;
        }

        $dimmer->setDimmingValue($v);
        $dimmer->setR($v_R);
        $dimmer->setG($v_G);
        $dimmer->setB($v_B);
        $dimmer->setW($v_W);
        @$dimmer->updateToDatabase($db);

        break;

}
?>
```

index.php

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>LED Controller</title>
        <script
src="//ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script>
        <script    src="//ajax.googleapis.com/ajax/libs/jqueryui/1.9.2/jquery-
ui.min.js"></script>
        <link rel="stylesheet" type="text/css" href="css/style.css">
        <link rel="stylesheet" href="james.css" />
    </head>
    <body>

        <div id="content">

            <!-- Dimming slider and value -->
            <div class="packet">
            <div id="slider_dim" class="slide"></div>
            <div id="val" class="num">0</div>
            </div>

            <!-- RGBW sliders and values -->
            <div class="packet">
            <div id="slider_R" class="slide"></div>
            <div id="val_R" class ="num">0</div>
            </div>

            <div class="packet">
            <div id="slider_G" class="slide"></div>
            <div id="val_G" class="num">0</div>
            </div>

            <div class="packet">
            <div id="slider_B" class="slide"></div>
```

```
            <div id="val_B" class="num">0</div>
            </div>

            <div class="packet">
            <div id="slider_W" class="slide"></div>
            <div id="val_W" class="num">0</div>
            </div>

            <!-- Color preview box -->
            <div id="colorBox"></div>

        </div>

        <script>
            // Dimming slider
            $("#slider_dim").slider();
            $("#slider_dim").slider("option","orientation","horizontal");

            // Gets current values from MYSQL
            $.ajax({
                url: "actions.php?fcn=get&id=1",
                type: "GET",
                success : function(d){

$("#slider_dim").slider("option","value",$.parseJSON(d).dimmingValue);

$("#slider_R").slider("option","value",$.parseJSON(d).RValue);

$("#slider_G").slider("option","value",$.parseJSON(d).GValue);

$("#slider_B").slider("option","value",$.parseJSON(d).BValue);

$("#slider_W").slider("option","value",$.parseJSON(d).WValue);
                }
            });

            // Event listener
            $('.slide').on('slidechange', function(e){
                var v = $("#slider_dim").slider("option","value");
                var v_R = $("#slider_R").slider("option","value");
                var v_G = $("#slider_G").slider("option","value");
                var v_B = $("#slider_B").slider("option","value");
                var v_W = $("#slider_W").slider("option","value");

                $('#val').html('<span class="label">D:</span> '+v);
                $('#val_R').html('<span class="label">R:</span>  '+v_R);
                $('#val_G').html('<span class="label">G:</span>  '+v_G);
                $('#val_B').html('<span class="label">B:</span>  '+v_B);
                $('#val_W').html('<span class="label">W:</span>  '+v_W);

                // Dimming previewer
                $('#colorBox').css('opacity',v/100);

                // Color box previewer

$('#colorBox').css('background','rgb('+v_R+','+v_G+','+v_B+')');
```

```
                // Set values for all sliders
                $.ajax({
                    url:
"actions.php?fcn=set&id=1&val="+v+"&val_R="+v_R+"&val_G="+v_G+"&val_B="+v_B+"
&val_W="+v_W,
                    type: "GET",
                    success : function(d){
                        //alert(d);
                    }
                });
            })


            // R slider
            $("#slider_R").slider();
            $("#slider_R").slider("option","orientation","horizontal");
            $("#slider_R").slider("option","max",255);


            // G slider
            $("#slider_G").slider();
            $("#slider_G").slider("option","orientation","horizontal");
            $("#slider_G").slider("option","max",255);


            // B slider
            $("#slider_B").slider();
            $("#slider_B").slider("option","orientation","horizontal");
            $("#slider_B").slider("option","max",255);


            // W slider
            $("#slider_W").slider();
            $("#slider_W").slider("option","orientation","horizontal");
            $("#slider_W").slider("option","max",255);


        </script>

    </body>
</html>
```

james.css

```
/*! jQuery UI - v1.9.2 - 2012-11-23
* http://jqueryui.com
* Includes: jquery.ui.core.css, jquery.ui.accordion.css,
jquery.ui.autocomplete.css, jquery.ui.button.css, jquery.ui.datepicker.css,
jquery.ui.dialog.css, jquery.ui.menu.css, jquery.ui.progressbar.css,
jquery.ui.resizable.css, jquery.ui.selectable.css, jquery.ui.slider.css,
jquery.ui.spinner.css, jquery.ui.tabs.css, jquery.ui.tooltip.css,
jquery.ui.theme.css
* Copyright 2012 jQuery Foundation and other contributors; Licensed MIT */

/* Layout helpers
----------------------------------*/
.ui-helper-hidden { display: none; }
```

```
.ui-helper-hidden-accessible { border: 0; clip: rect(0 0 0 0); height: 1px;
margin: -1px; overflow: hidden; padding: 0; position: absolute; width: 1px; }
.ui-helper-reset { margin: 0; padding: 0; border: 0; outline: 0; line-height:
1.3; text-decoration: none; font-size: 100%; list-style: none; }
.ui-helper-clearfix:before, .ui-helper-clearfix:after { content: ""; display:
table; }
.ui-helper-clearfix:after { clear: both; }
.ui-helper-clearfix { zoom: 1; }
.ui-helper-zfix { width: 100%; height: 100%; top: 0; left: 0; position:
absolute; opacity: 0; filter:Alpha(Opacity=0); }


/* Interaction Cues
----------------------------------*/
.ui-state-disabled { cursor: default !important; }


/* Icons
----------------------------------*/

/* states and images */
.ui-icon { display: block; text-indent: -99999px; overflow: hidden;
background-repeat: no-repeat; }


/* Misc visuals
----------------------------------*/

/* Overlays */
.ui-widget-overlay { position: absolute; top: 0; left: 0; width: 100%;
height: 100%; }

.ui-accordion .ui-accordion-header { display: block; cursor: pointer;
position: relative; margin-top: 2px; padding: .5em .5em .5em .7em; zoom: 1; }
.ui-accordion .ui-accordion-icons { padding-left: 2.2em; }
.ui-accordion .ui-accordion-noicons { padding-left: .7em; }
.ui-accordion .ui-accordion-icons .ui-accordion-icons { padding-left: 2.2em;
}
.ui-accordion .ui-accordion-header .ui-accordion-header-icon { position:
absolute; left: .5em; top: 50%; margin-top: -8px; }
.ui-accordion .ui-accordion-content { padding: 1em 2.2em; border-top: 0;
overflow: auto; zoom: 1; }

.ui-autocomplete {
      position: absolute;
      top: 0;
      left: 0;
      cursor: default;
}

/* workarounds */
* html .ui-autocomplete { width:1px; } /* without this, the menu expands to
100% in IE6 */

.ui-button { display: inline-block; position: relative; padding: 0; margin-
right: .1em; cursor: pointer; text-align: center; zoom: 1; overflow: visible;
} /* the overflow property removes extra width in IE */
```

```
.ui-button, .ui-button:link, .ui-button:visited, .ui-button:hover, .ui-
button:active { text-decoration: none; }
.ui-button-icon-only { width: 2.2em; } /* to make room for the icon, a width
needs to be set here */
button.ui-button-icon-only { width: 2.4em; } /* button elements seem to need
a little more width */
.ui-button-icons-only { width: 3.4em; }
button.ui-button-icons-only { width: 3.7em; }

/*button text element */
.ui-button .ui-button-text { display: block; line-height: 1.4;  }
.ui-button-text-only .ui-button-text { padding: .4em 1em; }
.ui-button-icon-only .ui-button-text, .ui-button-icons-only .ui-button-text {
padding: .4em; text-indent: -9999999px; }
.ui-button-text-icon-primary .ui-button-text, .ui-button-text-icons .ui-
button-text { padding: .4em 1em .4em 2.1em; }
.ui-button-text-icon-secondary .ui-button-text, .ui-button-text-icons .ui-
button-text { padding: .4em 2.1em .4em 1em; }
.ui-button-text-icons .ui-button-text { padding-left: 2.1em; padding-right:
2.1em; }
/* no icon support for input elements, provide padding by default */
input.ui-button { padding: .4em 1em; }

/*button icon element(s) */
.ui-button-icon-only .ui-icon, .ui-button-text-icon-primary .ui-icon, .ui-
button-text-icon-secondary .ui-icon, .ui-button-text-icons .ui-icon, .ui-
button-icons-only .ui-icon { position: absolute; top: 50%; margin-top: -8px;
}
.ui-button-icon-only .ui-icon { left: 50%; margin-left: -8px; }
.ui-button-text-icon-primary .ui-button-icon-primary, .ui-button-text-icons
.ui-button-icon-primary, .ui-button-icons-only .ui-button-icon-primary {
left: .5em; }
.ui-button-text-icon-secondary .ui-button-icon-secondary, .ui-button-text-
icons .ui-button-icon-secondary, .ui-button-icons-only .ui-button-icon-
secondary { right: .5em; }
.ui-button-text-icons .ui-button-icon-secondary, .ui-button-icons-only .ui-
button-icon-secondary { right: .5em; }

/*button sets*/
.ui-buttonset { margin-right: 7px; }
.ui-buttonset .ui-button { margin-left: 0; margin-right: -.3em; }

/* workarounds */
button.ui-button::-moz-focus-inner { border: 0; padding: 0; } /* reset extra
padding in Firefox */

.ui-datepicker { width: 17em; padding: .2em .2em 0; display: none; }
.ui-datepicker .ui-datepicker-header { position:relative; padding:.2em 0; }
.ui-datepicker .ui-datepicker-prev, .ui-datepicker .ui-datepicker-next {
position:absolute; top: 2px; width: 1.8em; height: 1.8em; }
.ui-datepicker .ui-datepicker-prev-hover, .ui-datepicker .ui-datepicker-next-
hover { top: 1px; }
.ui-datepicker .ui-datepicker-prev { left:2px; }
.ui-datepicker .ui-datepicker-next { right:2px; }
.ui-datepicker .ui-datepicker-prev-hover { left:1px; }
.ui-datepicker .ui-datepicker-next-hover { right:1px; }
.ui-datepicker .ui-datepicker-prev span, .ui-datepicker .ui-datepicker-next
```

```
span { display: block; position: absolute; left: 50%; margin-left: -8px; top:
50%; margin-top: -8px;  }
.ui-datepicker .ui-datepicker-title { margin: 0 2.3em; line-height: 1.8em;
text-align: center; }
.ui-datepicker .ui-datepicker-title select { font-size:1em; margin:1px 0; }
.ui-datepicker select.ui-datepicker-month-year {width: 100%;}
.ui-datepicker select.ui-datepicker-month,
.ui-datepicker select.ui-datepicker-year { width: 49%;}
.ui-datepicker table {width: 100%; font-size: .9em; border-collapse:
collapse; margin:0 0 .4em; }
.ui-datepicker th { padding: .7em .3em; text-align: center; font-weight:
bold; border: 0;  }
.ui-datepicker td { border: 0; padding: 1px; }
.ui-datepicker td span, .ui-datepicker td a { display: block; padding: .2em;
text-align: right; text-decoration: none; }
.ui-datepicker .ui-datepicker-buttonpane { background-image: none; margin:
.7em 0 0 0; padding:0 .2em; border-left: 0; border-right: 0; border-bottom:
0; }
.ui-datepicker .ui-datepicker-buttonpane button { float: right; margin: .5em
.2em .4em; cursor: pointer; padding: .2em .6em .3em .6em; width:auto;
overflow:visible; }
.ui-datepicker .ui-datepicker-buttonpane button.ui-datepicker-current {
float:left; }

/* with multiple calendars */
.ui-datepicker.ui-datepicker-multi { width:auto; }
.ui-datepicker-multi .ui-datepicker-group { float:left; }
.ui-datepicker-multi .ui-datepicker-group table { width:95%; margin:0 auto
.4em; }
.ui-datepicker-multi-2 .ui-datepicker-group { width:50%; }
.ui-datepicker-multi-3 .ui-datepicker-group { width:33.3%; }
.ui-datepicker-multi-4 .ui-datepicker-group { width:25%; }
.ui-datepicker-multi .ui-datepicker-group-last .ui-datepicker-header {
border-left-width:0; }
.ui-datepicker-multi .ui-datepicker-group-middle .ui-datepicker-header {
border-left-width:0; }
.ui-datepicker-multi .ui-datepicker-buttonpane { clear:left; }
.ui-datepicker-row-break { clear:both; width:100%; font-size:0em; }

/* RTL support */
.ui-datepicker-rtl { direction: rtl; }
.ui-datepicker-rtl .ui-datepicker-prev { right: 2px; left: auto; }
.ui-datepicker-rtl .ui-datepicker-next { left: 2px; right: auto; }
.ui-datepicker-rtl .ui-datepicker-prev:hover { right: 1px; left: auto; }
.ui-datepicker-rtl .ui-datepicker-next:hover { left: 1px; right: auto; }
.ui-datepicker-rtl .ui-datepicker-buttonpane { clear:right; }
.ui-datepicker-rtl .ui-datepicker-buttonpane button { float: left; }
.ui-datepicker-rtl .ui-datepicker-buttonpane button.ui-datepicker-current {
float:right; }
.ui-datepicker-rtl .ui-datepicker-group { float:right; }
.ui-datepicker-rtl .ui-datepicker-group-last .ui-datepicker-header { border-
right-width:0; border-left-width:1px; }
.ui-datepicker-rtl .ui-datepicker-group-middle .ui-datepicker-header {
border-right-width:0; border-left-width:1px; }

/* IE6 IFRAME FIX (taken from datepicker 1.5.3 */
.ui-datepicker-cover {
```

```
    position: absolute; /*must have*/
    z-index: -1; /*must have*/
    filter: mask(); /*must have*/
    top: -4px; /*must have*/
    left: -4px; /*must have*/
    width: 200px; /*must have*/
    height: 200px; /*must have*/
}
.ui-dialog { position: absolute; top: 0; left: 0; padding: .2em; width:
300px; overflow: hidden; }
.ui-dialog .ui-dialog-titlebar { padding: .4em 1em; position: relative;  }
.ui-dialog .ui-dialog-title { float: left; margin: .1em 16px .1em 0; }
.ui-dialog .ui-dialog-titlebar-close { position: absolute; right: .3em; top:
50%; width: 19px; margin: -10px 0 0 0; padding: 1px; height: 18px; }
.ui-dialog .ui-dialog-titlebar-close span { display: block; margin: 1px; }
.ui-dialog .ui-dialog-titlebar-close:hover, .ui-dialog .ui-dialog-titlebar-
close:focus { padding: 0; }
.ui-dialog .ui-dialog-content { position: relative; border: 0; padding: .5em
1em; background: none; overflow: auto; zoom: 1; }
.ui-dialog .ui-dialog-buttonpane { text-align: left; border-width: 1px 0 0 0;
background-image: none; margin: .5em 0 0 0; padding: .3em 1em .5em .4em; }
.ui-dialog .ui-dialog-buttonpane .ui-dialog-buttonset { float: right; }
.ui-dialog .ui-dialog-buttonpane button { margin: .5em .4em .5em 0; cursor:
pointer; }
.ui-dialog .ui-resizable-se { width: 14px; height: 14px; right: 3px; bottom:
3px; }
.ui-draggable .ui-dialog-titlebar { cursor: move; }

.ui-menu { list-style:none; padding: 2px; margin: 0; display:block; outline:
none; }
.ui-menu .ui-menu { margin-top: -3px; position: absolute; }
.ui-menu .ui-menu-item { margin: 0; padding: 0; zoom: 1; width: 100%; }
.ui-menu .ui-menu-divider { margin: 5px -2px 5px -2px; height: 0; font-size:
0; line-height: 0; border-width: 1px 0 0 0; }
.ui-menu .ui-menu-item a { text-decoration: none; display: block; padding:
2px .4em; line-height: 1.5; zoom: 1; font-weight: normal; }
.ui-menu .ui-menu-item a.ui-state-focus,
.ui-menu .ui-menu-item a.ui-state-active { font-weight: normal; margin: -1px;
}

.ui-menu .ui-state-disabled { font-weight: normal; margin: .4em 0 .2em; line-
height: 1.5; }
.ui-menu .ui-state-disabled a { cursor: default; }

/* icon support */
.ui-menu-icons { position: relative; }
.ui-menu-icons .ui-menu-item a { position: relative; padding-left: 2em; }

/* left-aligned */
.ui-menu .ui-icon { position: absolute; top: .2em; left: .2em; }

/* right-aligned */
.ui-menu .ui-menu-icon { position: static; float: right; }

.ui-progressbar { height:2em; text-align: left; overflow: hidden; }
.ui-progressbar .ui-progressbar-value {margin: -1px; height:100%; }
.ui-resizable { position: relative;}
```

```
.ui-resizable-handle { position: absolute;font-size: 0.1px; display: block; }
.ui-resizable-disabled .ui-resizable-handle, .ui-resizable-autohide .ui-
resizable-handle { display: none; }
.ui-resizable-n { cursor: n-resize; height: 7px; width: 100%; top: -5px;
left: 0; }
.ui-resizable-s { cursor: s-resize; height: 7px; width: 100%; bottom: -5px;
left: 0; }
.ui-resizable-e { cursor: e-resize; width: 7px; right: -5px; top: 0; height:
100%; }
.ui-resizable-w { cursor: w-resize; width: 7px; left: -5px; top: 0; height:
100%; }
.ui-resizable-se { cursor: se-resize; width: 12px; height: 12px; right: 1px;
bottom: 1px; }
.ui-resizable-sw { cursor: sw-resize; width: 9px; height: 9px; left: -5px;
bottom: -5px; }
.ui-resizable-nw { cursor: nw-resize; width: 9px; height: 9px; left: -5px;
top: -5px; }
.ui-resizable-ne { cursor: ne-resize; width: 9px; height: 9px; right: -5px;
top: -5px;}
.ui-selectable-helper { position: absolute; z-index: 100; border:1px dotted
black; }

.ui-slider { position: relative; text-align: left; }
.ui-slider .ui-slider-handle { position: absolute; z-index: 2; width: 1.2em;
height: 1.2em; cursor: default; }
.ui-slider .ui-slider-range { position: absolute; z-index: 1; font-size:
.7em; display: block; border: 0; background-position: 0 0; }

.ui-slider-horizontal { height: .8em; }
.ui-slider-horizontal .ui-slider-handle { top: -.3em; margin-left: -.6em; }
.ui-slider-horizontal .ui-slider-range { top: 0; height: 100%; }
.ui-slider-horizontal .ui-slider-range-min { left: 0; }
.ui-slider-horizontal .ui-slider-range-max { right: 0; }

.ui-slider-vertical { width: .8em; height: 100px; }
.ui-slider-vertical .ui-slider-handle { left: -.3em; margin-left: 0; margin-
bottom: -.6em; }
.ui-slider-vertical .ui-slider-range { left: 0; width: 100%; }
.ui-slider-vertical .ui-slider-range-min { bottom: 0; }
.ui-slider-vertical .ui-slider-range-max { top: 0; }
.ui-spinner { position:relative; display: inline-block; overflow: hidden;
padding: 0; vertical-align: middle; }
.ui-spinner-input { border: none; background: none; padding: 0; margin: .2em
0; vertical-align: middle; margin-left: .4em; margin-right: 22px; }
.ui-spinner-button { width: 16px; height: 50%; font-size: .5em; padding: 0;
margin: 0; text-align: center; position: absolute; cursor: default; display:
block; overflow: hidden; right: 0; }
.ui-spinner a.ui-spinner-button { border-top: none; border-bottom: none;
border-right: none; } /* more specificity required here to overide default
borders */
.ui-spinner .ui-icon { position: absolute; margin-top: -8px; top: 50%; left:
0; } /* vertical centre icon */
.ui-spinner-up { top: 0; }
.ui-spinner-down { bottom: 0; }

/* TR overrides */
.ui-spinner .ui-icon-triangle-1-s {
```

```
	/* need to fix icons sprite */
	background-position:-65px -16px;
}

.ui-tabs { position: relative; padding: .2em; zoom: 1; } /* position:
relative prevents IE scroll bug (element with position: relative inside
container with overflow: auto appear as "fixed") */
.ui-tabs .ui-tabs-nav { margin: 0; padding: .2em .2em 0; }
.ui-tabs .ui-tabs-nav li { list-style: none; float: left; position: relative;
top: 0; margin: 1px .2em 0 0; border-bottom: 0; padding: 0; white-space:
nowrap; }
.ui-tabs .ui-tabs-nav li a { float: left; padding: .5em 1em; text-decoration:
none; }
.ui-tabs .ui-tabs-nav li.ui-tabs-active { margin-bottom: -1px; padding-
bottom: 1px; }
.ui-tabs .ui-tabs-nav li.ui-tabs-active a, .ui-tabs .ui-tabs-nav li.ui-state-
disabled a, .ui-tabs .ui-tabs-nav li.ui-tabs-loading a { cursor: text; }
.ui-tabs .ui-tabs-nav li a, .ui-tabs-collapsible .ui-tabs-nav li.ui-tabs-
active a { cursor: pointer; } /* first selector in group seems obsolete, but
required to overcome bug in Opera applying cursor: text overall if defined
elsewhere... */
.ui-tabs .ui-tabs-panel { display: block; border-width: 0; padding: 1em
1.4em; background: none; }

.ui-tooltip {
	padding: 8px;
	position: absolute;
	z-index: 9999;
	max-width: 300px;
	-webkit-box-shadow: 0 0 5px #aaa;
	box-shadow: 0 0 5px #aaa;
}
/* Fades and background-images don't work well together in IE6, drop the
image */
* html .ui-tooltip {
	background-image: none;
}
body .ui-tooltip { border-width: 2px; }

/* Component containers
----------------------------------*/
.ui-widget { font-family: Verdana,Arial,sans-serif/*{ffDefault}*/; font-size:
1.1em/*{fsDefault}*/; }
.ui-widget .ui-widget { font-size: 1em; }
.ui-widget input, .ui-widget select, .ui-widget textarea, .ui-widget button {
font-family: Verdana,Arial,sans-serif/*{ffDefault}*/; font-size: 1em; }
.ui-widget-content { border: 1px solid #aaaaaa/*{borderColorContent}*/;
background: #ffffff/*{bgColorContent}*/ url(images/ui-
bg_flat_75_ffffff_40x100.png)/*{bgImgUrlContent}*/ 50%/*{bgContentXPos}*/
50%/*{bgContentYPos}*/ repeat-x/*{bgContentRepeat}*/; color:
#222222/*{fcContent}*/; }
.ui-widget-content a { color: #222222/*{fcContent}*/; }
.ui-widget-header { border: 1px solid #aaaaaa/*{borderColorHeader}*/;
background: #cccccc/*{bgColorHeader}*/ url(images/ui-bg_highlight-
soft_75_cccccc_1x100.png)/*{bgImgUrlHeader}*/ 50%/*{bgHeaderXPos}*/
50%/*{bgHeaderYPos}*/ repeat-x/*{bgHeaderRepeat}*/; color:
#222222/*{fcHeader}*/; font-weight: bold; }
```

```
.ui-widget-header a { color: #222222/*{fcHeader}*/; }

/* Interaction states
----------------------------------*/
.ui-state-default, .ui-widget-content .ui-state-default, .ui-widget-header
.ui-state-default { border: 1px solid #d3d3d3/*{borderColorDefault}*/;
background: #e6e6e6/*{bgColorDefault}*/ url(images/ui-
bg_glass_75_e6e6e6_1x400.png)/*{bgImgUrlDefault}*/ 50%/*{bgDefaultXPos}*/
50%/*{bgDefaultYPos}*/ repeat-x/*{bgDefaultRepeat}*/; font-weight:
normal/*{fwDefault}*/; color: #555555/*{fcDefault}*/; }
.ui-state-default a, .ui-state-default a:link, .ui-state-default a:visited {
color: #555555/*{fcDefault}*/; text-decoration: none; }
.ui-state-hover, .ui-widget-content .ui-state-hover, .ui-widget-header .ui-
state-hover, .ui-state-focus, .ui-widget-content .ui-state-focus, .ui-widget-
header .ui-state-focus { border: 1px solid #999999/*{borderColorHover}*/;
background: #dadada/*{bgColorHover}*/ url(images/ui-
bg_glass_75_dadada_1x400.png)/*{bgImgUrlHover}*/ 50%/*{bgHoverXPos}*/
50%/*{bgHoverYPos}*/ repeat-x/*{bgHoverRepeat}*/; font-weight:
normal/*{fwDefault}*/; color: #212121/*{fcHover}*/; }
.ui-state-hover a, .ui-state-hover a:hover, .ui-state-hover a:link, .ui-
state-hover a:visited { color: #212121/*{fcHover}*/; text-decoration: none; }
.ui-state-active, .ui-widget-content .ui-state-active, .ui-widget-header .ui-
state-active { border: 1px solid #aaaaaa/*{borderColorActive}*/; background:
#ffffff/*{bgColorActive}*/ url(images/ui-
bg_glass_65_ffffff_1x400.png)/*{bgImgUrlActive}*/ 50%/*{bgActiveXPos}*/
50%/*{bgActiveYPos}*/ repeat-x/*{bgActiveRepeat}*/; font-weight:
normal/*{fwDefault}*/; color: #212121/*{fcActive}*/; }
.ui-state-active a, .ui-state-active a:link, .ui-state-active a:visited {
color: #212121/*{fcActive}*/; text-decoration: none; }

/* Interaction Cues
----------------------------------*/
.ui-state-highlight, .ui-widget-content .ui-state-highlight, .ui-widget-
header .ui-state-highlight  {border: 1px solid
#fcefa1/*{borderColorHighlight}*/; background: #fbf9ee/*{bgColorHighlight}*/
url(images/ui-bg_glass_55_fbf9ee_1x400.png)/*{bgImgUrlHighlight}*/
50%/*{bgHighlightXPos}*/ 50%/*{bgHighlightYPos}*/ repeat-
x/*{bgHighlightRepeat}*/; color: #363636/*{fcHighlight}*/; }
.ui-state-highlight a, .ui-widget-content .ui-state-highlight a,.ui-widget-
header .ui-state-highlight a { color: #363636/*{fcHighlight}*/; }
.ui-state-error, .ui-widget-content .ui-state-error, .ui-widget-header .ui-
state-error {border: 1px solid #cd0a0a/*{borderColorError}*/; background:
#fef1ec/*{bgColorError}*/ url(images/ui-
bg_glass_95_fef1ec_1x400.png)/*{bgImgUrlError}*/ 50%/*{bgErrorXPos}*/
50%/*{bgErrorYPos}*/ repeat-x/*{bgErrorRepeat}*/; color:
#cd0a0a/*{fcError}*/; }
.ui-state-error a, .ui-widget-content .ui-state-error a, .ui-widget-header
.ui-state-error a { color: #cd0a0a/*{fcError}*/; }
.ui-state-error-text, .ui-widget-content .ui-state-error-text, .ui-widget-
header .ui-state-error-text { color: #cd0a0a/*{fcError}*/; }
.ui-priority-primary, .ui-widget-content .ui-priority-primary, .ui-widget-
header .ui-priority-primary { font-weight: bold; }
.ui-priority-secondary, .ui-widget-content .ui-priority-secondary,  .ui-
widget-header .ui-priority-secondary { opacity: .7; filter:Alpha(Opacity=70);
font-weight: normal; }
.ui-state-disabled, .ui-widget-content .ui-state-disabled, .ui-widget-header
.ui-state-disabled { opacity: .35; filter:Alpha(Opacity=35); background-
```

```
image: none; }
.ui-state-disabled .ui-icon { filter:Alpha(Opacity=35); } /* For IE8 - See
#6059 */

/* Icons
----------------------------------*/

/* states and images */
.ui-icon { width: 16px; height: 16px; background-image: url(images/ui-
icons_222222_256x240.png)/*{iconsContent}*/; }
.ui-widget-content .ui-icon {background-image: url(images/ui-
icons_222222_256x240.png)/*{iconsContent}*/; }
.ui-widget-header .ui-icon {background-image: url(images/ui-
icons_222222_256x240.png)/*{iconsHeader}*/; }
.ui-state-default .ui-icon { background-image: url(images/ui-
icons_888888_256x240.png)/*{iconsDefault}*/; }
.ui-state-hover .ui-icon, .ui-state-focus .ui-icon {background-image:
url(images/ui-icons_454545_256x240.png)/*{iconsHover}*/; }
.ui-state-active .ui-icon {background-image: url(images/ui-
icons_454545_256x240.png)/*{iconsActive}*/; }
.ui-state-highlight .ui-icon {background-image: url(images/ui-
icons_2e83ff_256x240.png)/*{iconsHighlight}*/; }
.ui-state-error .ui-icon, .ui-state-error-text .ui-icon {background-image:
url(images/ui-icons_cd0a0a_256x240.png)/*{iconsError}*/; }

/* positioning */
.ui-icon-carat-1-n { background-position: 0 0; }
.ui-icon-carat-1-ne { background-position: -16px 0; }
.ui-icon-carat-1-e { background-position: -32px 0; }
.ui-icon-carat-1-se { background-position: -48px 0; }
.ui-icon-carat-1-s { background-position: -64px 0; }
.ui-icon-carat-1-sw { background-position: -80px 0; }
.ui-icon-carat-1-w { background-position: -96px 0; }
.ui-icon-carat-1-nw { background-position: -112px 0; }
.ui-icon-carat-2-n-s { background-position: -128px 0; }
.ui-icon-carat-2-e-w { background-position: -144px 0; }
.ui-icon-triangle-1-n { background-position: 0 -16px; }
.ui-icon-triangle-1-ne { background-position: -16px -16px; }
.ui-icon-triangle-1-e { background-position: -32px -16px; }
.ui-icon-triangle-1-se { background-position: -48px -16px; }
.ui-icon-triangle-1-s { background-position: -64px -16px; }
.ui-icon-triangle-1-sw { background-position: -80px -16px; }
.ui-icon-triangle-1-w { background-position: -96px -16px; }
.ui-icon-triangle-1-nw { background-position: -112px -16px; }
.ui-icon-triangle-2-n-s { background-position: -128px -16px; }
.ui-icon-triangle-2-e-w { background-position: -144px -16px; }
.ui-icon-arrow-1-n { background-position: 0 -32px; }
.ui-icon-arrow-1-ne { background-position: -16px -32px; }
.ui-icon-arrow-1-e { background-position: -32px -32px; }
.ui-icon-arrow-1-se { background-position: -48px -32px; }
.ui-icon-arrow-1-s { background-position: -64px -32px; }
.ui-icon-arrow-1-sw { background-position: -80px -32px; }
.ui-icon-arrow-1-w { background-position: -96px -32px; }
.ui-icon-arrow-1-nw { background-position: -112px -32px; }
.ui-icon-arrow-2-n-s { background-position: -128px -32px; }
.ui-icon-arrow-2-ne-sw { background-position: -144px -32px; }
.ui-icon-arrow-2-e-w { background-position: -160px -32px; }
```

```
.ui-icon-arrow-2-se-nw { background-position: -176px -32px; }
.ui-icon-arrowstop-1-n { background-position: -192px -32px; }
.ui-icon-arrowstop-1-e { background-position: -208px -32px; }
.ui-icon-arrowstop-1-s { background-position: -224px -32px; }
.ui-icon-arrowstop-1-w { background-position: -240px -32px; }
.ui-icon-arrowthick-1-n { background-position: 0 -48px; }
.ui-icon-arrowthick-1-ne { background-position: -16px -48px; }
.ui-icon-arrowthick-1-e { background-position: -32px -48px; }
.ui-icon-arrowthick-1-se { background-position: -48px -48px; }
.ui-icon-arrowthick-1-s { background-position: -64px -48px; }
.ui-icon-arrowthick-1-sw { background-position: -80px -48px; }
.ui-icon-arrowthick-1-w { background-position: -96px -48px; }
.ui-icon-arrowthick-1-nw { background-position: -112px -48px; }
.ui-icon-arrowthick-2-n-s { background-position: -128px -48px; }
.ui-icon-arrowthick-2-ne-sw { background-position: -144px -48px; }
.ui-icon-arrowthick-2-e-w { background-position: -160px -48px; }
.ui-icon-arrowthick-2-se-nw { background-position: -176px -48px; }
.ui-icon-arrowthickstop-1-n { background-position: -192px -48px; }
.ui-icon-arrowthickstop-1-e { background-position: -208px -48px; }
.ui-icon-arrowthickstop-1-s { background-position: -224px -48px; }
.ui-icon-arrowthickstop-1-w { background-position: -240px -48px; }
.ui-icon-arrowreturnthick-1-w { background-position: 0 -64px; }
.ui-icon-arrowreturnthick-1-n { background-position: -16px -64px; }
.ui-icon-arrowreturnthick-1-e { background-position: -32px -64px; }
.ui-icon-arrowreturnthick-1-s { background-position: -48px -64px; }
.ui-icon-arrowreturn-1-w { background-position: -64px -64px; }
.ui-icon-arrowreturn-1-n { background-position: -80px -64px; }
.ui-icon-arrowreturn-1-e { background-position: -96px -64px; }
.ui-icon-arrowreturn-1-s { background-position: -112px -64px; }
.ui-icon-arrowrefresh-1-w { background-position: -128px -64px; }
.ui-icon-arrowrefresh-1-n { background-position: -144px -64px; }
.ui-icon-arrowrefresh-1-e { background-position: -160px -64px; }
.ui-icon-arrowrefresh-1-s { background-position: -176px -64px; }
.ui-icon-arrow-4 { background-position: 0 -80px; }
.ui-icon-arrow-4-diag { background-position: -16px -80px; }
.ui-icon-extlink { background-position: -32px -80px; }
.ui-icon-newwin { background-position: -48px -80px; }
.ui-icon-refresh { background-position: -64px -80px; }
.ui-icon-shuffle { background-position: -80px -80px; }
.ui-icon-transfer-e-w { background-position: -96px -80px; }
.ui-icon-transferthick-e-w { background-position: -112px -80px; }
.ui-icon-folder-collapsed { background-position: 0 -96px; }
.ui-icon-folder-open { background-position: -16px -96px; }
.ui-icon-document { background-position: -32px -96px; }
.ui-icon-document-b { background-position: -48px -96px; }
.ui-icon-note { background-position: -64px -96px; }
.ui-icon-mail-closed { background-position: -80px -96px; }
.ui-icon-mail-open { background-position: -96px -96px; }
.ui-icon-suitcase { background-position: -112px -96px; }
.ui-icon-comment { background-position: -128px -96px; }
.ui-icon-person { background-position: -144px -96px; }
.ui-icon-print { background-position: -160px -96px; }
.ui-icon-trash { background-position: -176px -96px; }
.ui-icon-locked { background-position: -192px -96px; }
.ui-icon-unlocked { background-position: -208px -96px; }
.ui-icon-bookmark { background-position: -224px -96px; }
.ui-icon-tag { background-position: -240px -96px; }
```

```
.ui-icon-home { background-position: 0 -112px; }
.ui-icon-flag { background-position: -16px -112px; }
.ui-icon-calendar { background-position: -32px -112px; }
.ui-icon-cart { background-position: -48px -112px; }
.ui-icon-pencil { background-position: -64px -112px; }
.ui-icon-clock { background-position: -80px -112px; }
.ui-icon-disk { background-position: -96px -112px; }
.ui-icon-calculator { background-position: -112px -112px; }
.ui-icon-zoomin { background-position: -128px -112px; }
.ui-icon-zoomout { background-position: -144px -112px; }
.ui-icon-search { background-position: -160px -112px; }
.ui-icon-wrench { background-position: -176px -112px; }
.ui-icon-gear { background-position: -192px -112px; }
.ui-icon-heart { background-position: -208px -112px; }
.ui-icon-star { background-position: -224px -112px; }
.ui-icon-link { background-position: -240px -112px; }
.ui-icon-cancel { background-position: 0 -128px; }
.ui-icon-plus { background-position: -16px -128px; }
.ui-icon-plusthick { background-position: -32px -128px; }
.ui-icon-minus { background-position: -48px -128px; }
.ui-icon-minusthick { background-position: -64px -128px; }
.ui-icon-close { background-position: -80px -128px; }
.ui-icon-closethick { background-position: -96px -128px; }
.ui-icon-key { background-position: -112px -128px; }
.ui-icon-lightbulb { background-position: -128px -128px; }
.ui-icon-scissors { background-position: -144px -128px; }
.ui-icon-clipboard { background-position: -160px -128px; }
.ui-icon-copy { background-position: -176px -128px; }
.ui-icon-contact { background-position: -192px -128px; }
.ui-icon-image { background-position: -208px -128px; }
.ui-icon-video { background-position: -224px -128px; }
.ui-icon-script { background-position: -240px -128px; }
.ui-icon-alert { background-position: 0 -144px; }
.ui-icon-info { background-position: -16px -144px; }
.ui-icon-notice { background-position: -32px -144px; }
.ui-icon-help { background-position: -48px -144px; }
.ui-icon-check { background-position: -64px -144px; }
.ui-icon-bullet { background-position: -80px -144px; }
.ui-icon-radio-on { background-position: -96px -144px; }
.ui-icon-radio-off { background-position: -112px -144px; }
.ui-icon-pin-w { background-position: -128px -144px; }
.ui-icon-pin-s { background-position: -144px -144px; }
.ui-icon-play { background-position: 0 -160px; }
.ui-icon-pause { background-position: -16px -160px; }
.ui-icon-seek-next { background-position: -32px -160px; }
.ui-icon-seek-prev { background-position: -48px -160px; }
.ui-icon-seek-end { background-position: -64px -160px; }
.ui-icon-seek-start { background-position: -80px -160px; }
/* ui-icon-seek-first is deprecated, use ui-icon-seek-start instead */
.ui-icon-seek-first { background-position: -80px -160px; }
.ui-icon-stop { background-position: -96px -160px; }
.ui-icon-eject { background-position: -112px -160px; }
.ui-icon-volume-off { background-position: -128px -160px; }
.ui-icon-volume-on { background-position: -144px -160px; }
.ui-icon-power { background-position: 0 -176px; }
.ui-icon-signal-diag { background-position: -16px -176px; }
.ui-icon-signal { background-position: -32px -176px; }
```

```
.ui-icon-battery-0 { background-position: -48px -176px; }
.ui-icon-battery-1 { background-position: -64px -176px; }
.ui-icon-battery-2 { background-position: -80px -176px; }
.ui-icon-battery-3 { background-position: -96px -176px; }
.ui-icon-circle-plus { background-position: 0 -192px; }
.ui-icon-circle-minus { background-position: -16px -192px; }
.ui-icon-circle-close { background-position: -32px -192px; }
.ui-icon-circle-triangle-e { background-position: -48px -192px; }
.ui-icon-circle-triangle-s { background-position: -64px -192px; }
.ui-icon-circle-triangle-w { background-position: -80px -192px; }
.ui-icon-circle-triangle-n { background-position: -96px -192px; }
.ui-icon-circle-arrow-e { background-position: -112px -192px; }
.ui-icon-circle-arrow-s { background-position: -128px -192px; }
.ui-icon-circle-arrow-w { background-position: -144px -192px; }
.ui-icon-circle-arrow-n { background-position: -160px -192px; }
.ui-icon-circle-zoomin { background-position: -176px -192px; }
.ui-icon-circle-zoomout { background-position: -192px -192px; }
.ui-icon-circle-check { background-position: -208px -192px; }
.ui-icon-circlesmall-plus { background-position: 0 -208px; }
.ui-icon-circlesmall-minus { background-position: -16px -208px; }
.ui-icon-circlesmall-close { background-position: -32px -208px; }
.ui-icon-squaresmall-plus { background-position: -48px -208px; }
.ui-icon-squaresmall-minus { background-position: -64px -208px; }
.ui-icon-squaresmall-close { background-position: -80px -208px; }
.ui-icon-grip-dotted-vertical { background-position: 0 -224px; }
.ui-icon-grip-dotted-horizontal { background-position: -16px -224px; }
.ui-icon-grip-solid-vertical { background-position: -32px -224px; }
.ui-icon-grip-solid-horizontal { background-position: -48px -224px; }
.ui-icon-gripsmall-diagonal-se { background-position: -64px -224px; }
.ui-icon-grip-diagonal-se { background-position: -80px -224px; }


/* Misc visuals
----------------------------------*/

/* Corner radius */
.ui-corner-all, .ui-corner-top, .ui-corner-left, .ui-corner-tl { -moz-border-
radius-topleft: 4px/*{cornerRadius}*/; -webkit-border-top-left-radius:
4px/*{cornerRadius}*/; -khtml-border-top-left-radius: 4px/*{cornerRadius}*/;
border-top-left-radius: 4px/*{cornerRadius}*/; }
.ui-corner-all, .ui-corner-top, .ui-corner-right, .ui-corner-tr { -moz-
border-radius-topright: 4px/*{cornerRadius}*/; -webkit-border-top-right-
radius: 4px/*{cornerRadius}*/; -khtml-border-top-right-radius:
4px/*{cornerRadius}*/; border-top-right-radius: 4px/*{cornerRadius}*/; }
.ui-corner-all, .ui-corner-bottom, .ui-corner-left, .ui-corner-bl { -moz-
border-radius-bottomleft: 4px/*{cornerRadius}*/; -webkit-border-bottom-left-
radius: 4px/*{cornerRadius}*/; -khtml-border-bottom-left-radius:
4px/*{cornerRadius}*/; border-bottom-left-radius: 4px/*{cornerRadius}*/; }
.ui-corner-all, .ui-corner-bottom, .ui-corner-right, .ui-corner-br { -moz-
border-radius-bottomright: 4px/*{cornerRadius}*/; -webkit-border-bottom-
right-radius: 4px/*{cornerRadius}*/; -khtml-border-bottom-right-radius:
4px/*{cornerRadius}*/; border-bottom-right-radius: 4px/*{cornerRadius}*/; }

/* Overlays */
.ui-widget-overlay { background: #aaaaaa/*{bgColorOverlay}*/ url(images/ui-
bg_flat_0_aaaaaa_40x100.png)/*{bgImgUrlOverlay}*/ 50%/*{bgOverlayXPos}*/
50%/*{bgOverlayYPos}*/ repeat-x/*{bgOverlayRepeat}*/; opacity:
```

```
.3;filter:Alpha(Opacity=30)/*{opacityOverlay}*/; }
.ui-widget-shadow { margin: -8px/*{offsetTopShadow}*/ 0 0 -
8px/*{offsetLeftShadow}*/; padding: 8px/*{thicknessShadow}*/; background:
#aaaaaa/*{bgColorShadow}*/ url(images/ui-
bg_flat_0_aaaaaa_40x100.png)/*{bgImgUrlShadow}*/ 50%/*{bgShadowXPos}*/
50%/*{bgShadowYPos}*/ repeat-x/*{bgShadowRepeat}*/; opacity:
.3;filter:Alpha(Opacity=30)/*{opacityShadow}*/; -moz-border-radius:
8px/*{cornerRadiusShadow}*/; -khtml-border-radius:
8px/*{cornerRadiusShadow}*/; -webkit-border-radius:
8px/*{cornerRadiusShadow}*/; border-radius: 8px/*{cornerRadiusShadow}*/; }
```

style.css

```
/*
    Document   : style
    Created on : Dec 29, 2012, 10:41:01 PM
    Author     : jyurkovich
    Description:
        Purpose of the stylesheet follows.
*/

html{
    font-family: Arial;

}

body{
    background: #ccc;

}

#content{
    margin: 0 auto;
    width: 300px;
    height: 400px;
    border-radius: 10px;
    background: whitesmoke;
    padding: 10px;

}

#colorBox{
    margin: 0 auto;
    margin-top: 80px;
    width: 150px;
    height: 100px;
    background: red;
    border: 1px inset black;
    border-radius: 10px;
    box-shadow: 4px 4px 2px 2px darkgray;

}

.label{
    padding-right: 5px;
    font-weight: bold;
```

```
}

.num{
    margin-top: 30px;
    float: left;

}

.slide{
    margin-top: 30px;
    width: 120px;
    float: right;

}

.packet{
    clear: both;
    margin: 0 auto;
    width: 200px;

}
```

7.2 Android Application Code

DimmerState.java

```
package com.seniordesign.led;

public class DimmerState {

      public String id;
      public String dimmingValue;
      public String RValue;
      public String GValue;
      public String BValue;


      public String getRValue() {
            return RValue;
      }
      public void setRValue(String rValue) {
            RValue = rValue;
      }
      public String getGValue() {
            return GValue;
      }
      public void setGValue(String gValue) {
            GValue = gValue;
      }
      public String getBValue() {
            return BValue;
      }
      public void setBValue(String bValue) {
            BValue = bValue;
      }
      public String getId() {
```

```
            return id;
      }
      public void setId(String id) {
            this.id = id;
      }
      public String getDimmingValue() {
            return dimmingValue;
      }
      public void setDimmingValue(String dimmingValue) {
            this.dimmingValue = dimmingValue;
      }

}
```

MainActivity.java

```
package com.seniordesign.led;

import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;

import com.google.gson.Gson;

import android.os.Build;
import android.os.Bundle;
import android.os.StrictMode;
import android.annotation.SuppressLint;
import android.annotation.TargetApi;
import android.app.Activity;
import android.graphics.Color;
import android.util.JsonReader;
import android.view.Menu;
import android.view.View;
import android.widget.EditText;
import android.widget.SeekBar;
import android.widget.SeekBar.OnSeekBarChangeListener;

public class MainActivity extends Activity {


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        //Set the label with the current value of the dimmer
        try {
            getVals();
```

```
            } catch (IOException e) {
                  // TODO Auto-generated catch block
                  e.printStackTrace();
            }

        SeekBar sb1 = (SeekBar) findViewById(R.id.seekBar1);

        OnSeekBarChangeListener sbcl = new OnSeekBarChangeListener() {

                  @Override
                  public void onStopTrackingTouch(SeekBar seekBar) {
                        // TODO Auto-generated method stub

                  }

                  @Override
                  public void onStartTrackingTouch(SeekBar seekBar) {
                        // TODO Auto-generated method stub

                  }

                  @Override
                  public void onProgressChanged(SeekBar seekBar, int
progress, boolean fromUser) {

                        SeekBar sb1 = (SeekBar) findViewById(R.id.seekBar1);


                           int sb_val = sb1.getProgress();

                           Integer temp1 = new Integer(sb_val);

                        try {
                                    sendToServer(temp1.toString());
                              } catch (IOException e) {
                                    // TODO Auto-generated catch block
                                    e.printStackTrace();
                              }

                        //View mlayout = findViewById(R.id.layOut);

                          // mlayout.setBackgroundColor(Color.rgb(sb_valr,
sb_valg, sb_valb));

                  }

            };



        sb1.setOnSeekBarChangeListener(sbcl);

    }

    public void sendToServer(String dVal) throws IOException{

      URL url = new
```

```
URL("http://192.168.1.2/led/actions.php?id=1&fcn=set&val="+dVal);
      URLConnection urlConnection = url.openConnection();
      InputStream in = new
BufferedInputStream(urlConnection.getInputStream());

    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }

    /*
    @TargetApi(Build.VERSION_CODES.GINGERBREAD) @SuppressLint({ "NewApi",
"UseValueOf" }) public void button_sendValue(View view) throws IOException {
      EditText editText = (EditText) findViewById(R.id.editText1);
      String str = editText.getText().toString();

      StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();

      StrictMode.setThreadPolicy(policy);

      SeekBar sb1 = (SeekBar) findViewById(R.id.seekBar1);
         int sb_val = sb1.getProgress();

         Integer temp = new Integer(sb_val);
         editText.setText(temp.toString());

      URL url = new
URL("http://192.168.2.11/led/actions.php?id=1&fcn=set&val="+sb_val+"&");
         URLConnection urlConnection = url.openConnection();
         InputStream in = new
BufferedInputStream(urlConnection.getInputStream());

         getVals();

    }
    */

    @TargetApi(Build.VERSION_CODES.ICE_CREAM_SANDWICH)
@SuppressLint("NewApi") public void button_refresh(View view) throws
IOException {

         getVals();

    }



    //Gets the current value of the dimmer
    @SuppressLint({ "NewApi", "UseValueOf" }) public void getVals() throws
IOException{
```

```
      StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().permitAll().build();

      StrictMode.setThreadPolicy(policy);

      URL url = new URL("http://192.168.1.2/led/actions.php?id=1&fcn=get");
         URLConnection urlConnection = url.openConnection();



         BufferedReader in = new BufferedReader(new InputStreamReader(
                     urlConnection.getInputStream()));
                  //String inputLine;
                  //while ((inputLine = in.readLine()) != null)
                  //System.out.println(inputLine);


                  Gson g = new Gson();
                  DimmerState dimmer = g.fromJson(in.readLine(),
DimmerState.class);

                  Integer i1 = new Integer(dimmer.getDimmingValue());


                  SeekBar sb1 = (SeekBar) findViewById(R.id.seekBar1);
                  sb1.setProgress(i1.intValue());


    }

}
```

activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/layOut"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/android_background"
    tools:context=".MainActivity" >

    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="33dp"
        android:ems="10"
        android:gravity="center_vertical|center"
        android:text="Light of the World"
        android:textColor="@android:color/white"
        android:textSize="32dp"
        android:textStyle="bold"
        android:typeface="serif" />

    <SeekBar
        android:id="@+id/seekBar1"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="147dp"
        android:max="100"
        android:maxWidth="150dp"
        android:minWidth="200dp" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="107dp"
        android:onClick="button_refresh"
        android:text="Refresh" />

</RelativeLayout>
```

7.3 Website Code

index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">

<head>

<title>Light of the World</title>

<meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-
8" />
<meta name="author" content="Erwin Aligam - styleshout.com" />
<meta name="description" content="Site Description Here" />
<meta name="keywords" content="keywords, here" />
<meta name="robots" content="index, follow, noarchive" />
<meta name="googlebot" content="noarchive" />

<link rel="stylesheet" href="images/Colourise.css" type="text/css" />

</head>

<body>

<!-- wrap starts here -->
<div id="wrap">

      <!--header -->
      <div id="header">

            <h1 id="logo-text"><a href="index.html" title="">Light of the
World</a></h1>
```

```
            <p id="intro">
            An Electrical Engineering Senior Design Project at the University
of Notre Dame, 2012-2013
            </p>

            <div  id="nav">
                  <ul>
                        <li id="current"><a href="index.html">Home</a></li>
                        <li><a href="documents.html">Documentation</a></li>
                        <li><a href="code.html">Data</a></li>
                        <li><a href="team.html">Team</a></li>
                  </ul>
            </div>



      <!--header ends-->
      </div>

      <!-- content-wrap starts -->
      <div id="content-wrap">

            <div id="main">

                  <a name="TemplateInfo"></a>
                  <h2>Our Project</h2>

            <p><strong>Light of the World</strong> is an Electrical
Engineering senior design project group at the
                  University of Notre Dame for the 2012-2013 school year. Our
goal is to create a prototype LED
                  lightbulb that will be able to fit into a standard light
socket. This prototype bulb has built-in
                  wireless capability through which the user will be able to
dim the light via an Android app from
                  any location that has internet access.
            </p>

            <p>Our project was motivated by the "green" initiative enacted by
many companies in an attempt to reduce energy consumption.
                  The common 60 watt incandescent light bulb lasts
approximately 1,200 hours, while an equivalent LED (800
                  lumens) uses 6-8 watts and lasts approximately 50,000 hours.
Incandescent lighting is still the predominant light
                  source, however, mainly because it can be expensive to revamp
an entire building’s lighting scheme.
            </p>

            <p>Our goal is to create an efficient, cost-effective prototype
LED lightbulb that could replace the traditional incandescent lightbulb.
                  The design process was motivated primarily by functionality,
but we paid special care to minimize costs
                  whenever possible. Our LED bulb has a wifi module which
communicates to our server and is able to be controlled either through
                  a webpage on our server or by an Android app. The bulb itself
is "plug-and-play;" simply plug it in to any standard light socket,
                  download the app to your Android smartphone, and control the
```

```
bulb(s) from anywhere -- even when you're on vacation! This functionality,
                combined with the tremendous energy savings of LED
technology, makes our product fill a niche in both commercial and consumer
markets.
            </p>

      <!-- content-wrap ends-->
      </div>

            <!-- sidebar starts -->
            <div id="sidebar">

                  <h3>Navigation</h3>
                  <ul class="sidemenu">
                        <li><a href="documents.html">Documentation</a></li>
                        <li><a href="code.html">Data</a></li>
                        <li><a href="team.html">Team</a></li>
                  </ul>


            <!-- sidebar ends -->
            </div>

      <!-- footer starts here -->
      <div id="footer-wrap">

                <div class="col float-right">
                    <p> &copy; copyright 2013 </p>
                </div>


      </div></div>
      <div class="clearer"></div>
      <!-- footer ends here -->

<!-- wrap ends here -->


<!-- Thanks for the design template: http://www.styleshout.com -->
</body>
</html>
```

code.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">

<head>

<title>Light of the World</title>

<meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-
8" />
<meta name="author" content="Erwin Aligam - styleshout.com" />
<meta name="description" content="Site Description Here" />
```

```
<meta name="keywords" content="keywords, here" />
<meta name="robots" content="index, follow, noarchive" />
<meta name="googlebot" content="noarchive" />

<link rel="stylesheet" href="images/Colourise.css" type="text/css" />

</head>

<body>

<!-- wrap starts here -->
<div id="wrap">

      <!--header -->
      <div id="header">

            <h1 id="logo-text"><a href="index.html" title="">Light of the
World</a></h1>
            <p id="intro">
            An Electrical Engineering Senior Design Project at the University
of Notre Dame, 2012-2013
            </p>

            <div  id="nav">
                  <ul>
                              <li><a href="index.html">Home</a></li>
                        <li><a href="documents.html">Documentation</a></li>
                        <li id="current"><a href="code.html">Data</a></li>
                        <li><a href="team.html">Team</a></li>
                  </ul>
            </div>



      <!--header ends-->
      </div>

      <!-- content-wrap starts -->
      <div id="content-wrap">

            <div id="main">

                  <a name="TemplateInfo"></a>
                  <h2>Code</h2>

            <p>The code used in our project:

            <li><strong>Microcontroller code</strong></a>
                </li>
                <li><strong>Android app code</strong></a>
                </li>
                <li><strong>Server and UI website code</strong></a>
                </li>

            </p>

                        <a name="TemplateInfo"></a>
```

```
                <h2>PCB Design and Fabrication</h2>

                    <p>In order to test our circuitry along the design
process, we
                    fabricated our own PCBs using the following protocol:

                    <li><a
href="images/fab_step1.jpg"><strong>Fabrication (step 1):</strong></a>
                        A single-sided PCB pattern is laser printed on
specialized
                    toner transfer paper.</li>
                    <li><a
href="images/fab_step2.jpg"><strong>Fabrication (step 2):</strong></a>
                        This pattern is placed face-down on a bare copper
PCB.
                        Heat and pressure are applied, adhering the
transfer paper to the
                        copper. The PCB is then soaked in deionized water
which dissolves
                        the transfer paper, leaving a toner mask on the
PCB.</li>
                    <li><a
href="images/fab_step3.jpg"><strong>Fabrication (step 3):</strong></a>
                        The board is placed in an etchant, which removes
                    unwanted copper.</li>
                    <li><a
href="images/fab_step4.jpg"><strong>Fabrication (step 4):</strong></a>
                        Acetone is used to remove the toner mask from the
PCB.</li></p>

                    <p>Self-fabricated boards:
                        <li><a
href="images/fab_flyback_002.JPG"><strong>Prototype flyback
converter</strong></a></li>
                        <li><a
href="images/fab_flyback_003.jpeg"><strong>Prototype flyback
converter</strong></a></li>
                        <li><a href="images/3-
3_supply_001.jpeg"><strong>3.3V Supply</strong></a></li>
                    </p>



                    <p>Schematics and diagrams for our project, designed
in Eagle:

            <li><a href="images/subsystems.jpg"><strong>Subsystem
design</strong></a>
            </li>
            <li><a href="images/wifi_b.png"><strong>Wifi
Board</strong></a>
            </li>
            <li><a href="images/wifi_s.png"><strong>Wifi Board
Schematic</strong></a>
            </li>
            <li><a href="images/flyback_b.png"><strong>Flyback
Board</strong></a>
```

```
                </li>
                <li><a href="images/flyback_s.png"><strong>Flyback Board
Schematic</strong></a>
                </li>


            </p>

                        <p>Final boards, produced by the boardhouse:

                <li><a href="images/uncut_001.jpeg"><strong>Boards
(uncut)</strong></a>
                </li>
                <li><a href="images/uncut_002.jpeg"><strong>Boards
(uncut)</strong></a>
                </li>
                <li><a href="images/uncut_003.jpeg"><strong>Boards
(uncut)</strong></a>
                </li>
                <li><a href="images/boards_001.jpeg"><strong>Boards
(cut)</strong></a>
                </li>
                <li><a href="images/flyback_001.jpeg"><strong>Flyback
board</strong></a>
                </li>
                <li><a href="images/flyback_002.jpeg"><strong>Flyback
board</strong></a>
                </li>
                <li><a href="images/flyback_003.jpeg"><strong>Flyback
board</strong></a>
                </li>
                <li><a href="images/wifi_001.jpeg"><strong>Wifi
board</strong></a>
                </li>
                <li><a href="images/wifi_002.jpeg"><strong>Wifi
board</strong></a>
                </li>
                <li><a href="images/wifi_003.jpeg"><strong>Wifi
board</strong></a>
                </li>


            </p>

                        <a name="TemplateInfo"></a>
                  <h2>CAD Drawings</h2>

                        <p>CAD drawings for our prototype packaging, designed
in Google Sketchup 8:


                <li><a href="images/bulb_final_comp.jpg"><strong>Size
Comparison</strong></a>
                </li>
                <li><a href="images/bulb_dimensions.jpg"><strong>LOTW
Prototype Dimensions</strong></a>
                </li>
```

```
                <li><a href="images/bulb_final.jpg"><strong>LOTW
Prototype</strong></a>
                </li>
                <li><a href="images/bulb_final_001.jpg"><strong>LOTW
Prototype Bulb</strong></a>
                </li>
                <li><a href="images/bulb_final_002.jpg"><strong>LOTW
Prototype Bulb</strong></a>
                </li>

            </p>

                        <a name="TemplateInfo"></a>
                  <h2>Packaging Prototypes</h2>

                        <p>The CAD drawings were used to 3D print two
prototypes. Prototype I (yellow) was printed
                            using The Replicator (Makerbot); Prototype II
(black) was printed using a Fortus 250mc (Stratasys).
                            The second prototype fixed several issues in the
design of the first attempt.</p>

                        <p>Makerbot 3D Printed prototype:

                <li><a href="images/makerbot_001.jpeg"><strong>Makerbot 3D
Printer</strong></a>
                </li>
                <li><a href="images/draft_bottom_001.jpeg"><strong>Prototype
I</strong></a>
                </li>
                <li><a href="images/draft_bottom_002.jpeg"><strong>Prototype
I</strong></a>
                </li>
                <li><a href="images/draft_top_001.jpeg"><strong>Prototype
I</strong></a>
                </li>
                <li><a href="images/draft_top_002.jpeg"><strong>Prototype
I</strong></a>
                </li>
                <li><a
href="images/draft_assembled_001.jpeg"><strong>Prototype I</strong></a>

                        </p>

                        <p>Stratasys 3D Printer prototype:
                </li>
                <li><a href="images/fortus.jpeg"><strong>Stratasys 3D
Printer</strong></a>
                </li>
                <li><a href="images/printing_001.jpeg"><strong>Printing
Process</strong></a>
                </li>
                <li><a href="images/printing_002.jpeg"><strong>Printing
Process</strong></a>
                </li>
                <li><a href="images/printing_003.jpeg"><strong>Printing
Process</strong></a>
```

```
                </li>
                <li><a href="images/printing_004.jpeg"><strong>Printing
Process</strong></a>
                </li>
                <li><a href="images/printing_005.jpeg"><strong>Printing
Process</strong></a>
                </li>
                <li><a href="images/printing_006.jpeg"><strong>Printing
Process</strong></a>
                </li>
                <li><a href="images/scaffold_001.jpeg"><strong>Prototype II
(with scaffolding)</strong></a>
                </li>
                <li><a href="images/scaffold_002.jpeg"><strong>Prototype II
(with scaffolding)</strong></a>
                </li>
                <li><a href="images/scaffold_003.jpeg"><strong>Prototype II
(with scaffolding)</strong></a>
                </li>
                <li><a href="images/bottom_001.jpeg"><strong>Prototype
II</strong></a>
                </li>
                <li><a href="images/bottom_002.jpeg"><strong>Prototype
II</strong></a>
                </li>
                <li><a href="images/bottom_003.jpeg"><strong>Prototype
II</strong></a>
                </li>
                <li><a href="images/top_001.jpeg"><strong>Prototype
II</strong></a>
                </li>
                <li><a href="images/top_002.jpeg"><strong>Prototype
II</strong></a>
                </li>
                <li><a
href="images/final_assembled_001.jpeg"><strong>Prototype II</strong></a>
                </li>
                <li><a
href="images/final_assembled_002.jpeg"><strong>Prototype II</strong></a>
                </li>

            </p>

                        <p>Both prototypes:

                <li><a href="images/comp_006.jpeg"><strong>Prototypes I &
II</strong></a>
                </li>
                <li><a href="images/comp_005.jpeg"><strong>Prototypes I &
II</strong></a>
                </li>
                <li><a href="images/comp_004.jpeg"><strong>Prototypes I &
II</strong></a>
                </li>
                <li><a href="images/comp_002.jpeg"><strong>Prototypes I &
II</strong></a>
                </li>
```

```
                <li><a href="images/comp_003.jpeg"><strong>Prototypes I &
II</strong></a>
                </li>

                        </p>

      <!-- content-wrap ends-->
      </div>

            <!-- sidebar starts -->
            <div id="sidebar">

                  <h3>Navigation</h3>
                  <ul class="sidemenu">
                        <li><a href="index.html">Home</a></li>
                        <li><a href="documents.html">Documentation</a></li>
                        <li><a href="team.html">Team</a></li>
                  </ul>
                </div>

      <!-- footer starts here -->
      <div id="footer-wrap">

                <div class="col float-right">
                    <p> &copy; copyright 2013 </p>
                </div>


      </div></div>
      <div class="clearer"></div>
      <!-- footer ends here -->

<!-- wrap ends here -->


<!-- Thanks for the design template: http://www.styleshout.com -->
</body>
</html>
```

documents.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">

<head>

<title>Light of the World</title>

<meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-
8" />
<meta name="author" content="Erwin Aligam - styleshout.com" />
<meta name="description" content="Site Description Here" />
<meta name="keywords" content="keywords, here" />
<meta name="robots" content="index, follow, noarchive" />
<meta name="googlebot" content="noarchive" />
```

```
<link rel="stylesheet" href="images/Colourise.css" type="text/css" />

</head>

<body>

<!-- wrap starts here -->
<div id="wrap">

      <!--header -->
      <div id="header">

            <h1 id="logo-text"><a href="index.html" title="">Light of the
World</a></h1>
            <p id="intro">
            An Electrical Engineering Senior Design Project at the University
of Notre Dame, 2012-2013
            </p>

            <div  id="nav">
                  <ul>
                              <li><a href="index.html">Home</a></li>
                        <li id="current"><a
href="documents.html">Documentation</a></li>
                        <li><a href="code.html">Data</a></li>
                        <li><a href="team.html">Team</a></li>
                  </ul>
            </div>



      <!--header ends-->
      </div>

      <!-- content-wrap starts -->
      <div id="content-wrap">

            <div id="main">

                  <a name="TemplateInfo"></a>
                <h2>Project Requirements</h2>

                        <p>Electrical Engineering Senior Design is a two-
semester capstone course at the University of
                           Notre Dame geared toward giving students hands-on
experience with general electrical engineering
                           skills. Teams of 4-5 students work together to
build a functional prototype which is demonstrated
                           to students and faculty at the end of the school
year. During the first semester of the course, class meets
                           twice a week and focuses on the design process,
forming teams, and choosing a project idea. The second semester focuses on
                           constructing the prototype and demonstrating its
functionality. There are no classes second semester, but
                           teams are expected to hold weekly meetings and
work toward the completion of their prototype. Teams have various deadlines
```

```
along the way which require
                          certain subsystems to be completed. A detailed
final report is due at the end of the semester.

            </p>

                  <a name="TemplateInfo"></a>
                  <h2>Documents</h2>

            <p>The following files document the design process and decision
making for our project:

                <li><a href="images/proposal.pdf"><strong>Proposal
Presentation:</strong></a> the
                    initial project proposal presentation
                </li>
                <li><a href="images/written_proposal.pdf"><strong>Written
Proposal:</strong></a> the
                    initial project proposal
                </li>
                <li><a href="images/HLD.pdf"><strong>High Level
Design:</strong></a> a
                    brief description of the project's goals, projected
costs,
                    and difficulties
                </li>
                <li><strong>Final Project Report:</strong></a> a detailed
overview of the design process,
                required materials and cost, application of prototype, and
future improvements
                </li>

            </p>

                        <a name="TemplateInfo"></a>
                  <h2>Meetings</h2>

                        <p>The agendas and minutes for our weekly group
meetings in the spring semester:

                <li><a href="images/LOTW-minutes-1_14_13.pdf"><strong>Minutes
- 1.14.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-1_21_13.pdf"><strong>Agenda -
1.21.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-1_21_13.pdf"><strong>Minutes
- 1.21.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-1_28_13.pdf"><strong>Agenda -
1.28.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-1_28_13.pdf"><strong>Minutes
- 1.28.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-2_4_13.pdf"><strong>Agenda -
2.4.2013</strong></a>
```

```
                </li>
                <li><a href="images/LOTW-minutes-2_4_13.pdf"><strong>Minutes
- 2.4.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-2_10_13.pdf"><strong>Agenda -
2.10.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-2_18_13.pdf"><strong>Agenda -
2.18.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-2_18_13.pdf"><strong>Minutes
- 2.18.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-2_25_13.pdf"><strong>Agenda -
2.25.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-2_25_13.pdf"><strong>Minutes
- 2.25.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-3_4_13.pdf"><strong>Agenda -
3.4.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-3_25_13.pdf"><strong>Agenda -
3.25.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-4_1_13.pdf"><strong>Agenda -
4.1.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-4_1_13.pdf"><strong>Minutes
- 4.1.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-4-8-13.pdf"><strong>Agenda -
4.8.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-4_8_13.pdf"><strong>Minutes
- 4.8.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-4_15_13.pdf"><strong>Agenda -
4.15.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-4_15_13.pdf"><strong>Minutes
- 4.15.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-4_22_13.pdf"><strong>Agenda -
4.22.2013</strong></a>
                </li>
                <li><a href="images/LOTW-minutes-4_22_13.pdf"><strong>Minutes
- 4.22.2013</strong></a>
                </li>
                <li><a href="images/LOTW-agenda-4_29_13.pdf"><strong>Agenda -
4.29.2013</strong></a>
                </li>
            </p>

      <!-- content-wrap ends-->
      </div>
```

```
            <!-- sidebar starts -->
            <div id="sidebar">

                  <h3>Navigation</h3>
                  <ul class="sidemenu">
                        <li><a href="index.html">Home</a></li>
                        <li><a href="code.html">Data</a></li>
                        <li><a href="team.html">Team</a></li>
                  </ul>
                </div>

      <!-- footer starts here -->
      <div id="footer-wrap">

                <div class="col float-right">
                    <p> &copy; copyright 2013 </p>
                </div>


      </div></div>
      <div class="clearer"></div>
      <!-- footer ends here -->

<!-- wrap ends here -->


<!-- Thanks for the design template: http://www.styleshout.com -->
</body>
</html>
```

team.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">

<head>

<title>Light of the World</title>

<meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-
8" />
<meta name="author" content="Erwin Aligam - styleshout.com" />
<meta name="description" content="Site Description Here" />
<meta name="keywords" content="keywords, here" />
<meta name="robots" content="index, follow, noarchive" />
<meta name="googlebot" content="noarchive" />

<link rel="stylesheet" href="images/Colourise.css" type="text/css" />

</head>

<body>

<!-- wrap starts here -->
<div id="wrap">
```

```
      <!--header -->
      <div id="header">

            <h1 id="logo-text"><a href="index.html" title="">Light of the
World</a></h1>
            <p id="intro">
            An Electrical Engineering Senior Design Project at the University
of Notre Dame, 2012-2013
            </p>

            <div  id="nav">
                  <ul>
                        <li><a href="index.html">Home</a></li>
                        <li><a href="documents.html">Documentation</a></li>
                        <li><a href="code.html">Data</a></li>
                        <li id="current"><a href="team.html">Team</a></li>

                  </ul>
            </div>



      <!--header ends-->
      </div>



      <!-- content-wrap starts -->
      <div id="content-wrap">

                <!-- sidebar starts -->
            <div id="sidebar">

                  <h3>Navigation</h3>
                  <ul class="sidemenu">
                        <li><a href="index.html">Home</a></li>
                        <li><a href="documents.html">Documentation</a></li>
                        <li><a href="code.html">Data</a></li>
                  </ul>
               </div>

            <div id="main">




                  <a name="TemplateInfo"></a>
                  <h2>The Team</a></h2>


                  <h3>Joe Altura</h3>
                  <p><img src="images/joe.jpg" width="100" height="120"
alt="firefox" class="float-left" /></a>
                  Joe is a senior electrical engineer. He will be working for
QualiTECH starting in the fall.
                  </p>
```

```
                        <p>  </p>
                        <p>  </p>

                        <p><strong>Project Focuses:</strong> embedded
systems, wireless communications and protocols
                        </p>
                        <p>  </p>


                        <h3>Brian Appleton</h3>
                    <p><a><img src="images/brian.jpg" width="100" height="120"
alt="firefox" class="float-left" /></a>
                    <p>  </p>
                        <p>  </p>
                        <p>  </p>
                        <p>  </p>


                        <p><strong>Project Focuses:</strong> power
electronics, board fabrication,3D printing, packaging
                        </p>
                        <p>  </p>

                        <h3>Sean Baur</h3>
                    <p><a><img src="images/sean.jpg" width="100" height="120"
class="float-left" /></a>
                    Sean is a senior electrical engineering major, with a
concentration in energy.  He is originally from Colorado,
                              but is working for an engineering firm in Chicago
working on substation design.
                        <p>  </p>
                        <p>  </p>



                        <p><strong>Project Focuses:</strong> power
electronics, thermal packaging, 3D printing
                        </p>
                           <p>  </p>

                        <h3>James Yurkovich</h3>
                    <p><a><img src="images/JamesYurkovich.jpg" width="100"
height="120" class="float-left" /></a>
                    James is a senior electrical engineering major (class of
2013) with a concentration in biosystems. He completed
                        the course requirements for medical school and will
be pursuing a PhD in Bioinformatics and Systems
                        Biology at the University of California, San Diego.
                        </p>

                        <p><strong>Project Focuses:</strong> server
development, website design, wireless communication and protocols, Android
                        app development and design, CAD drawings, packaging,
3D printing
                        </p>
```

```
            <!-- main ends -->
            </div>


      <!-- content-wrap ends-->
      </div>



      <!-- footer starts here -->
      <div id="footer-wrap">

                <div class="col float-right">
                    <p> &copy; copyright 2013 </p>
                </div>

      <!-- footer ends here -->

<!-- wrap ends here -->
</div>

<!-- Thanks for the design template: http://www.styleshout.com -->

</body>
</html>
```

7.4 Microcontroller Code

Due to the size of the Microchip TCP/IP stack, we have only included the portions that have been added or edited. This includes MainDemo.h, MainDemo.c, getServerValue.c, setServerValue.c, the hardware profile header file originally intended for the PICDEM2.net board that was edited, the TCPIP configuration header file originally intended for the PICDEM2.net board that was edited, the Wi-Fi configuration header file, WFAPI.h, and WF_EInt.c. Otherwise all other files are the same as the Microchip TCP/IP stack libraries version 5.42.06

7.4.1 Main Header, Main c file and Functions used in main

MainDemo.h

```
#ifndef _MAINDEMO_H
#define _MAINDEMO_H

#define BAUD_RATE       (19200)           // bps

#if !defined(THIS_IS_STACK_APPLICATION)
      extern BYTE AN0String[8];
#endif

void DoUARTConfig(void);

      #define SaveAppConfig(a)


void getServerValue(BYTE *);
```

```
BYTE setServerValue(BYTE);



// Define a header structure for validating the AppConfig data structure in
EEPROM/Flash
typedef struct
{
      unsigned short wConfigurationLength;      // Number of bytes saved in
EEPROM/Flash (sizeof(APP_CONFIG))
      unsigned short wOriginalChecksum;         // Checksum of the original
AppConfig defaults as loaded from ROM (to detect when to wipe the
EEPROM/Flash record of AppConfig due to a stack change, such as when
switching from Ethernet to Wi-Fi)
      unsigned short wCurrentChecksum;          // Checksum of the current
EEPROM/Flash data.  This protects against using corrupt values if power
failure occurs while writing them and helps detect coding errors in which
some other task writes to the EEPROM in the AppConfig area.
} NVM_VALIDATION_STRUCT;


// An actual function defined in MainDemo.c for displaying the current IP
// address on the UART and/or LCD.
void DisplayIPValue(IP_ADDR IPVal);


#endif // _MAINDEMO_H
```

MainDemo.c

```
#define THIS_IS_STACK_APPLICATION
#define LIGHT_BULB
//#define LIGHT_SENSOR


// Include all headers for any enabled TCPIP Stack functions
#include "TCPIP Stack/TCPIP.h"

// Include functions specific to this stack application
#include "MainDemo.h"

// Used for Wi-Fi assertions
#define WF_MODULE_NUMBER   WF_MODULE_MAIN_DEMO

// Declare AppConfig structure and some other supporting stack variables
APP_CONFIG AppConfig;
static unsigned short wOriginalAppConfigChecksum;    // Checksum of the ROM
defaults for AppConfig
BYTE AN0String[8];

 volatile BYTE firingPin=0b1000;
 volatile BYTE zeroCross=0;

// Private helper functions.
// These may or may not be present in all applications.
static void InitAppConfig(void);
static void InitializeBoard(void);
```

```
#if defined(LIGHT_SENSOR)
WORD doAD(void);
#endif
#if defined(WF_CS_TRIS)
    void WF_Connect(void);
    #if !defined(MRF24WG)
    extern BOOL gRFModuleVer1209orLater;
    #endif
    #if defined(DERIVE_KEY_FROM_PASSPHRASE_IN_HOST)
    tPassphraseReady g_WpsPassphrase;
    #endif    /* defined(DERIVE_KEY_FROM_PASSPHRASE_IN_HOST) */
#endif

//
// PIC18 Interrupt Service Routines
//
// NOTE: Several PICs, including the PIC18F4620 revision A3 have a RETFIE
FAST/MOVFF bug
// The interruptlow keyword is used to work around the bug when using C18

    #pragma interruptlow LowISR
    void LowISR(void)
    {
            TickUpdate();
            WFEintISR();
    }

    #pragma interruptlow HighISR
    void HighISR(void)
    {

#if defined(LIGHT_BULB)
        if(PIR2bits.CCP2IF ){
            if( CCPR1H!=0 || CCPR1L!=1){
                CCPR3H=CCPR2H+CCPR1H;
                CCPR3L=CCPR2L+CCPR1L;
                if(CCPR3L<CCPR2L && CCPR3L<CCPR1L)//Catch overflow on lower 8
bits
                    CCPR3H=CCPR3H+1;
                CCP3CONbits.CCP3M=0b1000&firingPin;
            }
            else{
                CCPR3H=CCPR2H+0x09;
                CCPR3L=CCPR2L+0x20;
                if(CCPR3L<CCPR2L && CCPR3L<CCPR1L)//Catch overflow on lower 8
bits
                    CCPR3H=CCPR3H+1;
                CCP3CONbits.CCP3M=0b1001;
            }
            CCP2CONbits.CCP2M0 ^=1;
            CCPR2L=0;
            CCPR2H=0;
            zeroCross=1;
            PIR2bits.CCP2IF=0;
        }
        else if(PIR3bits.CCP3IF){
            T1CONbits.TMR1ON=0;
```

```
            TMR1L=0;
            TMR1H=0;
            if(CCP3CONbits.CCP3M==0b1000){
            CCPR3H=0x03;
            CCPR3L=0x20;
            CCP3CONbits.CCP3M0=1;
            }
            else
                CCP3CONbits.CCP3M=0b0000;
            T1CONbits.TMR1ON=1;
            PIR3bits.CCP3IF=0;//Clear interrupt flag
        }
#endif

    }

    #pragma code lowVector=0x18
    void LowVector(void){_asm goto LowISR _endasm}
    #pragma code highVector=0x8
    void HighVector(void){_asm goto HighISR _endasm}
    #pragma code // Return to default code section

#if defined(WF_CS_TRIS)
// Global variables
UINT8 ConnectionProfileID;
#endif

//
// Main application entry point.
//

void main(void)
{
#if defined(LIGHT_BULB)
    WORD
LookUpCCP[100]={26869,26862,26851,26835,26815,26790,26761,26727,26689,26646,2
6599,26547,26491,26430,26365,26295,26221,26143,26059,25972,25879,25783,25681,
25576,25466,25351,25232,25108,24980,24847,24710,24568,24422,24271,24116,23956
,23792,23623,23450,23273,23090,22904,22713,22517,22317,22112,21903,21689,2147
1,21248,21021,20790,20553,20313,20068,19818,19564,19305,19042,18774,18502,182
25,17944,17659,17369,17074,16775,16471,16163,15850,15533,15212,14885,14555,14
220,13880,13536,13187,12834,12477,12114,11748,11377,11001,10621,10236,9847,94
54,9056,8653,8246,7834,7418,6998,6573,6143,5709,5270,4827,1};
#endif
#if defined(LIGHT_SENSOR)
    WORD LookUpAD[10]={641,609,581, 555, 531, 509, 488, 469, 452, 436};

    WORD adVal;
    unsigned int j;
#endif
    static DWORD t = 0;
    static DWORD dwLastIP = 0;
    BYTE dimmingValue = 100;//Maybe change to 0 and initially remove firing
pin
    BYTE intermediateValue=dimmingValue;
    BYTE sent=0;
#if defined(STACK_USE_UART)
```

```
    char print[15];
#endif
#if defined(LIGHT_BULB)
    CCPR1H =(LookUpCCP[(dimmingValue-1)]&0xFF00)>>8;
    CCPR1L =(LookUpCCP[(dimmingValue-1)]&0x00FF);
#endif
    // Initialize application specific hardware
    InitializeBoard();
#if defined(LIGHT_SENSOR)
    //First A to D
    adVal=doAD();
    for(j=0;j<10;j++){
        if(adVal>LookUpAD[j])
            break;
    }
    dimmingValue=j*10;
#endif

    // Initialize stack-related hardware components that may be
    // required by the UART configuration routines
    TickInit();
    // Initialize Stack and application related NV variables into AppConfig.
    InitAppConfig();
    // Initiates board setup process if button is depressed
    // on startup

    // Initialize core stack layers (MAC, ARP, TCP, UDP) and
    // application modules (HTTP, SNMP, etc.)
    //ClrWdt();
    StackInit();

    #if defined(WF_CS_TRIS)
    #if defined(DERIVE_KEY_FROM_PASSPHRASE_IN_HOST)
        g_WpsPassphrase.valid = FALSE;
    #endif    /* defined(DERIVE_KEY_FROM_PASSPHRASE_IN_HOST) */
    //ClrWdt();
    WF_Connect();
    #endif
    // Now that all items are initialized, begin the co-operative
    // multitasking loop.  This infinite loop will continuously
    // execute all stack-related tasks, as well as your own
    // application's functions.  Custom functions should be added
    // at the end of this loop.
    // Note that this is a "co-operative mult-tasking" mechanism
    // where every task performs its tasks (whether all in one shot
    // or part of it) and returns so that other tasks can do their
    // job.
    // If a task needs very long time to do its job, it must be broken
    // down into smaller pieces so that other tasks can have CPU time.
    while(1)
    {
        //ClrWdt();
        // This task performs normal stack task including checking
        // for incoming packet, type of packet and calling
        // appropriate stack entity to process it.
            StackTask();
```

```
        #if defined(WF_CS_TRIS)
        #if !defined(MRF24WG)
        if (gRFModuleVer1209orLater)
        #endif
            WiFiTask();
        #endif
        //ClrWdt();
        #if defined(LIGHT_BULB) &&
defined(STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE)
            getServerValue(&dimmingValue);
            if(dimmingValue==0){
                 firingPin=0;
                 if((intermediateValue - dimmingValue)!=0){
                        intermediateValue=dimmingValue;
                        #if defined(STACK_USE_UART)
                            sprintf(print,"\r\nValue %i",dimmingValue);
                            putsUART(print);
                        #endif
                  }
                 t=TickGet();
            }
            else{
                if(intermediateValue == 0){
                    firingPin=0b1000;
                    intermediateValue=50;
                    CCPR1H =(LookUpCCP[(intermediateValue-1)]&0xFF00)>>8;
                    CCPR1L =(LookUpCCP[(intermediateValue-1)]&0x00FF);
                    intermediateValue=0;

                    if((TickGet()-t)>=TICK_SECOND/2ul){
                        intermediateValue=50;
                        if(dimmingValue>intermediateValue)
                            intermediateValue++;
                        else if(dimmingValue<intermediateValue)
                            intermediateValue--;
                        else
                            intermediateValue=dimmingValue;
                        t=TickGet();
                    }
                }
                else if((dimmingValue-intermediateValue)!=0){
                    firingPin=0b1000;
                    if(zeroCross){
                        if(dimmingValue>intermediateValue)
                            intermediateValue++;
                        else if(dimmingValue>intermediateValue)
                            intermediateValue--;
                        else
                            intermediateValue=dimmingValue;
                        CCPR1H =(LookUpCCP[(intermediateValue-1)]&0xFF00)>>8;
                        CCPR1L =(LookUpCCP[(intermediateValue-1)]&0x00FF);
                        zeroCross=0;
                    }
                }
```

```
            }
        #endif
        #if defined(LIGHT_SENSOR) &&
defined(STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE)
            //if((TickGet()-t)>= 5ul*TICK_SECOND ){
                //Do A to D
                adVal=doAD();
                for(j=0;j<10;j++){
                    if(adVal>LookUpAD[j])
                        break;
                }
                dimmingValue=j*10;
                //t=TickGet();
                sent=0;
            //}
            //if(!sent){
                sent=setServerValue(dimmingValue);
                //intermediateValue=dimmingValue;
            //}
#endif
        // If the local IP address has changed (ex: due to DHCP lease change)
        // write the new IP address to the LCD display, UART, and Announce
        // service
        if(dwLastIP != AppConfig.MyIPAddr.Val)
        {
            dwLastIP = AppConfig.MyIPAddr.Val;

            #if defined(STACK_USE_UART)
                putrsUART((ROM char*)"\r\nNew IP Address: ");
            #endif

            //DisplayIPValue(AppConfig.MyIPAddr);

            #if defined(STACK_USE_UART)
                putrsUART((ROM char*)"\r\n");
            #endif


        }
    }
}


#if defined(WF_CS_TRIS)

/*****************************************************************************
*
 * FUNCTION: WF_Connect
 *
 * RETURNS:  None
 *
 * PARAMS:   None
 *
 *  NOTES:   Connects to an 802.11 network.  Customize this function as
needed
 *           for your application.
```

```
*****************************************************************************
/
void WF_Connect(void)
{
    UINT8 channelList[] = MY_DEFAULT_CHANNEL_LIST;

    /* create a Connection Profile */
    WF_CPCreate(&ConnectionProfileID);

    WF_SetRegionalDomain(MY_DEFAULT_DOMAIN);

    WF_CPSetSsid(ConnectionProfileID,
                 AppConfig.MySSID,
                 AppConfig.SsidLength);

    WF_CPSetNetworkType(ConnectionProfileID, MY_DEFAULT_NETWORK_TYPE);

    WF_CASetScanType(MY_DEFAULT_SCAN_TYPE);


    WF_CASetChannelList(channelList, sizeof(channelList));

    // The Retry Count parameter tells the WiFi Connection manager how many
attempts to make when trying
    // to connect to an existing network.  In the Infrastructure case, the
default is to retry forever so that
    // if the AP is turned off or out of range, the radio will continue to
attempt a connection until the
    // AP is eventually back on or in range.  In the Adhoc case, the default
is to retry 3 times since the
    // purpose of attempting to establish a network in the Adhoc case is only
to verify that one does not
    // initially exist.  If the retry count was set to WF_RETRY_FOREVER in
the AdHoc mode, an AdHoc network
    // would never be established.
    WF_CASetListRetryCount(MY_DEFAULT_LIST_RETRY_COUNT);

    WF_CASetEventNotificationAction(MY_DEFAULT_EVENT_NOTIFICATION_LIST);

    WF_CASetBeaconTimeout(MY_DEFAULT_BEACON_TIMEOUT);

    #if !defined(MRF24WG)
        if (gRFModuleVer1209orLater)
    #else
        {
            // If WEP security is used, set WEP Key Type.  The default WEP
Key Type is Shared Key.
            if (AppConfig.SecurityMode == WF_SECURITY_WEP_40 ||
AppConfig.SecurityMode == WF_SECURITY_WEP_104)
            {
                WF_CPSetWepKeyType(ConnectionProfileID,
MY_DEFAULT_WIFI_SECURITY_WEP_KEYTYPE);
            }
        }
    #endif

    #if defined(MRF24WG)
```

```
        // Error check items specific to WPS Push Button mode
        #if (MY_DEFAULT_WIFI_SECURITY_MODE==WF_SECURITY_WPS_PUSH_BUTTON)
            #if !defined(WF_P2P)
                WF_ASSERT(strlen(AppConfig.MySSID) == 0);  // SSID must be
empty when using WPS
                WF_ASSERT(sizeof(channelList)==11);        // must scan all
channels for WPS
            #endif

             #if (MY_DEFAULT_NETWORK_TYPE == WF_P2P)
                WF_ASSERT(strcmp((char *)AppConfig.MySSID, "DIRECT-") == 0);
                WF_ASSERT(sizeof(channelList) == 3);
                WF_ASSERT(channelList[0] == 1);
                WF_ASSERT(channelList[1] == 6);
                WF_ASSERT(channelList[2] == 11);
            #endif
        #endif

    #endif /* MRF24WG */

    #if defined(DERIVE_KEY_FROM_PASSPHRASE_IN_HOST)
        if (AppConfig.SecurityMode == WF_SECURITY_WPA_WITH_PASS_PHRASE
            || AppConfig.SecurityMode == WF_SECURITY_WPA2_WITH_PASS_PHRASE
            || AppConfig.SecurityMode ==
WF_SECURITY_WPA_AUTO_WITH_PASS_PHRASE) {
            WF_ConvPassphrase2Key(AppConfig.SecurityKeyLength,
AppConfig.SecurityKey,
                AppConfig.SsidLength, AppConfig.MySSID);
            AppConfig.SecurityMode--;
            AppConfig.SecurityKeyLength = 32;
        }
    #if defined (MRF24WG)
        else if (AppConfig.SecurityMode == WF_SECURITY_WPS_PUSH_BUTTON
                    || AppConfig.SecurityMode == WF_SECURITY_WPS_PIN) {
            WF_YieldPassphrase2Host();
        }
    #endif    /* defined (MRF24WG) */
    #endif    /* defined(DERIVE_KEY_FROM_PASSPHRASE_IN_HOST) */

    WF_CPSetSecurity(ConnectionProfileID,
                     AppConfig.SecurityMode,
                     AppConfig.WepKeyIndex,   /* only used if WEP enabled */
                     AppConfig.SecurityKey,
                     AppConfig.SecurityKeyLength);

    #if MY_DEFAULT_PS_POLL == WF_ENABLED
        WF_PsPollEnable(TRUE);
    #if !defined(MRF24WG)
        if (gRFModuleVer1209orLater)
            WFEnableDeferredPowerSave();
    #endif    /* !defined(MRF24WG) */
    #else     /* MY_DEFAULT_PS_POLL != WF_ENABLED */
        WF_PsPollDisable();
    #endif    /* MY_DEFAULT_PS_POLL == WF_ENABLED */


    WF_CMConnect(ConnectionProfileID);
```

```
}
#endif /* WF_CS_TRIS */

// Writes an IP address to the LCD display and the UART as available
void DisplayIPValue(IP_ADDR IPVal)
{

    BYTE IPDigit[4];
    BYTE i;

    for(i = 0; i < sizeof(IP_ADDR); i++)
    {
        uitoa((WORD)IPVal.v[i], IPDigit);

        #if defined(STACK_USE_UART)
            putsUART((char *) IPDigit);
        #endif

            if(i == sizeof(IP_ADDR)-1)
                break;

        #if defined(STACK_USE_UART)
            while(BusyUART());
            WriteUART('.');
        #endif
    }
}


/*****************************************************************************
  Function:
    static void InitializeBoard(void)

  Description:
    This routine initializes the hardware.  It is a generic initialization
    routine for many of the Microchip development boards, using definitions
    in HardwareProfile.h to determine specific initialization.

  Precondition:
    None

  Parameters:
    None - None

  Returns:
    None

  Remarks:
    None

*****************************************************************************/
static void InitializeBoard(void)
{
  TRISE=0xFF;
  TRISB=0xFF;
  TRISAbits.RA0=1;
  OSCCONbits.OSTS = 0;//Probably read only so doesn't actually do anything
```

```
  OSCCONbits.IRCF=0b110;//Choose 4Mhz clock
  OSCCONbits.SCS=0b00;//Choose primary oscillator as clock
  OSCTUNEbits.INTSRC=1;//Use internal oscillator for 32kHz clock
  OSCTUNEbits.PLLEN=1;//Enable PLL
  OSCTUNEbits.TUN=0b00000;//For tuning RC
  while (!(OSCCONbits.IOFS)) { continue; } //Wait for stable


    ADCON0bits.ADON=0;//Turn off AD converter
    ADCON0bits.CHS=0b0000;//Set to channel 0
#if defined(LIGHT_BULB)
    ADCON1bits.PCFG=0b1111;//Make all analog pins digital
#else
    TRISAbits.RA0=1;
    ADCON1bits.VCFG=0b00;
    ADCON1bits.PCFG=0b1110;//Make all analog pins digital except AN0

    ADCON2bits.ADFM=1;
    ADCON2bits.ACQT=0b011;
    ADCON2bits.ADCS=0b010; // Conversion clock source Fosc/32
    ADCON0bits.ADON=1;//Turn on AD converter
#endif

    // Disable internal PORTB pull-ups
    INTCON2bits.RBPU = 1;

    // Configure USART
    #if defined(STACK_USE_UART)
    TRISGbits.RG2=1;
    TRISGbits.RG1=0;
    TXSTA2 = 0x20;
    RCSTA2 = 0x90;
    TXSTA2bits.BRGH=1;
    SPBRG2=51;
    SPBRGH2=0;
    #else
    TXSTA2=0x00;
    RCSTA2=0x00;
    #endif

    // Enable Interrupts
    RCONbits.IPEN = 1;        // Enable interrupt priorities
    INTCONbits.GIEH = 1;
    INTCONbits.GIEL = 1;

    //Set up chip select
#if defined(WF_CS_TRIS)
    WF_CS_IO = 1;
    WF_CS_TRIS = 0;
#endif


#if defined(LIGHT_BULB)

    T3CONbits.T3CCP1=0;//Set up CCP modules to Timer 1
    T3CONbits.T3CCP2=0;
```

```
    PIR3bits.CCP3IF=0;

    TRISGbits.RG0=0;
    PORTGbits.RG0=0;
    CCP3CONbits.CCP3M=0b1000;//Set up CCP3 to go from low to high
    CCPR3H=0x00;
    CCPR3L=0x00;
    IPR3bits.CCP3IP=1;
    PIE3bits.CCP3IE=1;

    PIR2bits.CCP2IF=0;
    TRISEbits.RE7=1;
    CCP2CONbits.CCP2M=0b0100;//Set up CCP3 to capture every falling edge

    IPR2bits.CCP2IP=1;
    PIE2bits.CCP2IE=1;
    CCPR2H=0;
    CCPR2L=0;

    //Set up TMR1 CCP Timer
    //IPR1bits.TMR1IP =1;
    PIR1bits.TMR1IF = 0;
    PIE1bits.TMR1IE=0;
    T1CONbits.TMR1ON=0;
    T1CONbits.RD16=0;
    T1CONbits.T1RUN=0;
    T1CONbits.T1OSCEN=0;
    T1CONbits.TMR1CS=0;
    T1CONbits.T1CKPS=0b00;
    TMR1L=0;
    TMR1H=0;
    T1CONbits.TMR1ON=1;

#endif
}

/*****************************************************************
 * Function:        void InitAppConfig(void)
 *
 * PreCondition:    MPFSInit() is already called.
 *
 * Input:           None
 *
 * Output:          Write/Read non-volatile config variables.
 *
 * Side Effects:    None
 *
 * Overview:        None
 *
 * Note:            None
 *****************************************************************/
// MAC Address Serialization using a MPLAB PM3 Programmer and
// Serialized Quick Turn Programming (SQTP).
// The advantage of using SQTP for programming the MAC Address is it
// allows you to auto-increment the MAC address without recompiling
// the code for each unit.  To use SQTP, the MAC address must be fixed
// at a specific location in program memory.  Uncomment these two pragmas
```

```
// that locate the MAC address at 0x1FFF0.  Syntax below is for MPLAB C
// Compiler for PIC18 MCUs. Syntax will vary for other compilers.
//#pragma romdata MACROM=0x1FFF0
static ROM BYTE SerializedMACAddress[6] = {MY_DEFAULT_MAC_BYTE1,
MY_DEFAULT_MAC_BYTE2, MY_DEFAULT_MAC_BYTE3, MY_DEFAULT_MAC_BYTE4,
MY_DEFAULT_MAC_BYTE5, MY_DEFAULT_MAC_BYTE6};
//#pragma romdata

static void InitAppConfig(void)
{
    while(1)
    {
        // Start out zeroing all AppConfig bytes to ensure all fields are
        // deterministic for checksum generation
        memset((void*)&AppConfig, 0x00, sizeof(AppConfig));

        AppConfig.Flags.bIsDHCPEnabled = TRUE;
        AppConfig.Flags.bInConfigMode = TRUE;
        memcpypgm2ram((void*)&AppConfig.MyMACAddr, (ROM
void*)SerializedMACAddress, sizeof(AppConfig.MyMACAddr));
//        {
//            _prog_addressT MACAddressAddress;
//            MACAddressAddress.next = 0x157F8;
//            _memcpy_p2d24((char*)&AppConfig.MyMACAddr, MACAddressAddress,
sizeof(AppConfig.MyMACAddr));
//        }
        AppConfig.MyIPAddr.Val = MY_DEFAULT_IP_ADDR_BYTE1 |
MY_DEFAULT_IP_ADDR_BYTE2<<8ul | MY_DEFAULT_IP_ADDR_BYTE3<<16ul |
MY_DEFAULT_IP_ADDR_BYTE4<<24ul;
        AppConfig.DefaultIPAddr.Val = AppConfig.MyIPAddr.Val;
        AppConfig.MyMask.Val = MY_DEFAULT_MASK_BYTE1 |
MY_DEFAULT_MASK_BYTE2<<8ul | MY_DEFAULT_MASK_BYTE3<<16ul |
MY_DEFAULT_MASK_BYTE4<<24ul;
        AppConfig.DefaultMask.Val = AppConfig.MyMask.Val;
        AppConfig.MyGateway.Val = MY_DEFAULT_GATE_BYTE1 |
MY_DEFAULT_GATE_BYTE2<<8ul | MY_DEFAULT_GATE_BYTE3<<16ul |
MY_DEFAULT_GATE_BYTE4<<24ul;
        AppConfig.PrimaryDNSServer.Val = MY_DEFAULT_PRIMARY_DNS_BYTE1 |
MY_DEFAULT_PRIMARY_DNS_BYTE2<<8ul  | MY_DEFAULT_PRIMARY_DNS_BYTE3<<16ul  |
MY_DEFAULT_PRIMARY_DNS_BYTE4<<24ul;
        AppConfig.SecondaryDNSServer.Val = MY_DEFAULT_SECONDARY_DNS_BYTE1 |
MY_DEFAULT_SECONDARY_DNS_BYTE2<<8ul  | MY_DEFAULT_SECONDARY_DNS_BYTE3<<16ul
| MY_DEFAULT_SECONDARY_DNS_BYTE4<<24ul;



        // Load the default NetBIOS Host Name
        memcpypgm2ram(AppConfig.NetBIOSName, (ROM void*)MY_DEFAULT_HOST_NAME,
16);
        FormatNetBIOSName(AppConfig.NetBIOSName);

        #if defined(WF_CS_TRIS)
            // Load the default SSID Name
            WF_ASSERT(sizeof(MY_DEFAULT_SSID_NAME) <=
sizeof(AppConfig.MySSID));
            memcpypgm2ram(AppConfig.MySSID, (ROM void*)MY_DEFAULT_SSID_NAME,
sizeof(MY_DEFAULT_SSID_NAME));
```

```
            AppConfig.SsidLength = sizeof(MY_DEFAULT_SSID_NAME) - 1;

            AppConfig.SecurityMode = MY_DEFAULT_WIFI_SECURITY_MODE;

            #if (MY_DEFAULT_WIFI_SECURITY_MODE == WF_SECURITY_OPEN)
                memset(AppConfig.SecurityKey, 0x00,
sizeof(AppConfig.SecurityKey));
                AppConfig.SecurityKeyLength = 0;

            #elif MY_DEFAULT_WIFI_SECURITY_MODE == WF_SECURITY_WEP_40
                AppConfig.WepKeyIndex  = MY_DEFAULT_WEP_KEY_INDEX;
                memcpypgm2ram(AppConfig.SecurityKey, (ROM
void*)MY_DEFAULT_WEP_KEYS_40, sizeof(MY_DEFAULT_WEP_KEYS_40) - 1);
                AppConfig.SecurityKeyLength = sizeof(MY_DEFAULT_WEP_KEYS_40)
- 1;

            #elif MY_DEFAULT_WIFI_SECURITY_MODE == WF_SECURITY_WEP_104
                AppConfig.WepKeyIndex  = MY_DEFAULT_WEP_KEY_INDEX;
                memcpypgm2ram(AppConfig.SecurityKey, (ROM
void*)MY_DEFAULT_WEP_KEYS_104, sizeof(MY_DEFAULT_WEP_KEYS_104) - 1);
                AppConfig.SecurityKeyLength = sizeof(MY_DEFAULT_WEP_KEYS_104)
- 1;

            #elif (MY_DEFAULT_WIFI_SECURITY_MODE == WF_SECURITY_WPA_WITH_KEY)
|| \
                  (MY_DEFAULT_WIFI_SECURITY_MODE ==
WF_SECURITY_WPA2_WITH_KEY)      || \
                  (MY_DEFAULT_WIFI_SECURITY_MODE ==
WF_SECURITY_WPA_AUTO_WITH_KEY)
                memcpypgm2ram(AppConfig.SecurityKey, (ROM
void*)MY_DEFAULT_PSK, sizeof(MY_DEFAULT_PSK) - 1);
                AppConfig.SecurityKeyLength = sizeof(MY_DEFAULT_PSK) - 1;

            #elif (MY_DEFAULT_WIFI_SECURITY_MODE ==
WF_SECURITY_WPA_WITH_PASS_PHRASE)     || \
                  (MY_DEFAULT_WIFI_SECURITY_MODE ==
WF_SECURITY_WPA2_WITH_PASS_PHRASE)    || \
                  (MY_DEFAULT_WIFI_SECURITY_MODE ==
WF_SECURITY_WPA_AUTO_WITH_PASS_PHRASE)
                memcpypgm2ram(AppConfig.SecurityKey, (ROM
void*)MY_DEFAULT_PSK_PHRASE, sizeof(MY_DEFAULT_PSK_PHRASE) - 1);
                AppConfig.SecurityKeyLength = sizeof(MY_DEFAULT_PSK_PHRASE) -
1;
            #elif (MY_DEFAULT_WIFI_SECURITY_MODE ==
WF_SECURITY_WPS_PUSH_BUTTON)
                memset(AppConfig.SecurityKey, 0x00,
sizeof(AppConfig.SecurityKey));
                AppConfig.SecurityKeyLength = 0;
            #elif (MY_DEFAULT_WIFI_SECURITY_MODE == WF_SECURITY_WPS_PIN)
                memcpypgm2ram(AppConfig.SecurityKey, (ROM
void*)MY_DEFAULT_WPS_PIN, sizeof(MY_DEFAULT_WPS_PIN) - 1);
                AppConfig.SecurityKeyLength = sizeof(MY_DEFAULT_WPS_PIN) - 1;
            #elif (MY_DEFAULT_WIFI_SECURITY_MODE == WF_SECURITY_EAP)
                memset(AppConfig.SecurityKey, 0x00,
sizeof(AppConfig.SecurityKey));
                AppConfig.SecurityKeyLength = 0;
            #else
```

```
                #error "No security defined"
            #endif /* MY_DEFAULT_WIFI_SECURITY_MODE */

        #endif

        // Compute the checksum of the AppConfig defaults as loaded from ROM
        wOriginalAppConfigChecksum = CalcIPChecksum((BYTE*)&AppConfig,
sizeof(AppConfig));

        break;
    }
}
#if defined(LIGHT_SENSOR)
WORD doAD(void){
    WORD value=0;
    ADCON0bits.GO = 1;
    //Wait until A/D conversion is done
    while(ADCON0bits.GO);
    value=ADRESH;
    value=value<<8ul;
    value=value | ((ADRESL&0xFC));
    return value;
}
#endif
```

getServerValue.c

```
/*********************************************************************
 *
 *  Generic TCP Client Example Application
 *  Module for Microchip TCP/IP Stack
 *   -Implements an example HTTP client and should be used as a basis
 *      for creating new TCP client applications
 *     -Reference: None.  Hopefully AN833 in the future.
 *
 ********************************************************************
 * FileName:        GenericTCPClient.c
 * Dependencies:    TCP, DNS, ARP, Tick
 * Processor:       PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32
 * Compiler:        Microchip C32 v1.05 or higher
 *                           Microchip C30 v3.12 or higher
 *                           Microchip C18 v3.30 or higher
 *                           HI-TECH PICC-18 PRO 9.63PL2 or higher
 * Company:         Microchip Technology, Inc.
 *
 * Software License Agreement
 *
 * Copyright (C) 2002-2009 Microchip Technology Inc.  All rights
 * reserved.
 *
 * Microchip licenses to you the right to use, modify, copy, and
 * distribute:
 * (i)  the Software when embedded on a Microchip microcontroller or
 *      digital signal controller product ("Device") which is
 *      integrated into Licensee's product; or
 * (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
 *          ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device
 *          used in conjunction with a Microchip ethernet controller for
```

```
 *          the sole purpose of interfacing with the ethernet controller.
 *
 * You should refer to the license agreement accompanying this
 * Software for additional information regarding your rights and
 * obligations.
 *
 * THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT
 * WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT
 * LIMITATION, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A
 * PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL
 * MICROCHIP BE LIABLE FOR ANY INCIDENTAL, SPECIAL, INDIRECT OR
 * CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF
 * PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS
 * BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE
 * THEREOF), ANY CLAIMS FOR INDEMNITY OR CONTRIBUTION, OR OTHER
 * SIMILAR COSTS, WHETHER ASSERTED ON THE BASIS OF CONTRACT, TORT
 * (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR OTHERWISE.
 *
 *
 * Author               Date    Comment
 *~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 * Howard Schlunder     8/01/06     Original
 ********************************************************************/
#define __GENERICTCPCLIENT_C

#include "TCPIPConfig.h"

#if defined(STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE)

#include "TCPIP Stack/TCPIP.h"


// Defines the server to be accessed for this application
#ifdef WIFI_NET_TEST
static BYTE ServerName[] =    "www" WIFI_NET_TEST_DOMAIN;
#else
static BYTE ServerName[] =    "192.168.1.2";
#endif

// Defines the port to be accessed for this application
#if defined(STACK_USE_SSL_CLIENT)
    static WORD ServerPort = HTTPS_PORT;
      // Note that if HTTPS is used, the ServerName and URL
      // must change to an SSL enabled server.
#else
    static WORD ServerPort = HTTP_PORT;
#endif

// Defines the URL to be requested by this HTTP client
static ROM BYTE RemoteURL[] = "/led/actions.php?fcn=get&id=1";


/*****************************************************************************
*
  Function:
      void GenericTCPClient(void)
```

```
  Summary:
      Implements a simple HTTP client (over TCP).

  Description:
      This function implements a simple HTTP client, which operates over TCP.
      The function is called periodically by the stack, and waits for BUTTON1
      to be pressed.  When the button is pressed, the application opens a TCP
      connection to an Internet search engine, performs a search for the word
      "Microchip" on "microchip.com", and prints the resulting HTML page to
      the UART.

      This example can be used as a model for many TCP and HTTP client
      applications.

  Precondition:
      TCP is initialized.

  Parameters:
      None

  Returns:
      None

***************************************************************************/
void getServerValue(BYTE *dim)
{
        BYTE                            num[3]={0,0,0};
        BYTE                            found;
        BYTE                            value;
        BYTE                            end;
        BYTE                            places;
        char                            print[15];
        enum DEC_PARSE_STATES{
            CHECK_0=0,
            CHECK_1,
            CHECK_2,
            FAST_FORWARD_1,
            FAST_FORWARD_2,
            PARSE
        }PARSE_STATES=CHECK_0;
      BYTE                      i;
        WORD                            j;
        BYTE                            k;
      //WORD                          w;
        WORD                            length;
      BYTE                      vBuffer[125];
        const int BUFFSIZE=125;
      static DWORD              Timer;
      static TCP_SOCKET MySocket = INVALID_SOCKET;
      static enum _GenericTCPExampleState
      {
            SM_HOME = 0,
            SM_SOCKET_OBTAINED,
            #if defined(STACK_USE_SSL_CLIENT)
      SM_START_SSL,
      #endif
            SM_PROCESS_RESPONSE,
```

```
            SM_DISCONNECT,
            SM_DONE
      } GenericTCPExampleState = SM_DONE;

      switch(GenericTCPExampleState)
      {
            case SM_HOME:
                  // Connect a socket to the remote TCP server
                  //MySocket = TCPOpen((DWORD)(PTR_BASE)&ServerName[0],
TCP_OPEN_RAM_HOST, ServerPort, TCP_PURPOSE_GENERIC_TCP_CLIENT);
                        MySocket = TCPOpen((DWORD)&ServerName[0],
TCP_OPEN_RAM_HOST, ServerPort, TCP_PURPOSE_GENERIC_TCP_CLIENT);

                        //putrsUART((ROM char*)"\r\nConnecting ... \r\n ");
                  // Abort operation if no TCP socket of type
TCP_PURPOSE_GENERIC_TCP_CLIENT is available
                  // If this ever happens, you need to go add one to
TCPIPConfig.h
                  if(MySocket == INVALID_SOCKET)
                        break;


                  GenericTCPExampleState++;
                  Timer = TickGet();

                  break;

            case SM_SOCKET_OBTAINED:
                  // Wait for the remote server to accept our connection
request
                  if(!TCPIsConnected(MySocket))
                  {
                        // Time out if too much time is spent in this state
                        if(TickGet()-Timer > 5*TICK_SECOND)
                        {
                              // Close the socket so it can be used by other
modules
                              TCPDisconnect(MySocket);
                              MySocket = INVALID_SOCKET;
                              GenericTCPExampleState--;
                        }
                        break;
                  }
                        //putrsUART((ROM char*)"\r\nConnected\r\n ");
                  Timer = TickGet();


    #if defined (STACK_USE_SSL_CLIENT)
            if(!TCPStartSSLClient(MySocket,(void *)"thishost"))
                break;
                  GenericTCPExampleState++;
                  break;

        case SM_START_SSL:
            if (TCPSSLIsHandshaking(MySocket))
            {
                        if(TickGet()-Timer > 10*TICK_SECOND)
```

```
                        {
                              // Close the socket so it can be used by other
modules
                              TCPDisconnect(MySocket);
                              MySocket = INVALID_SOCKET;
                              GenericTCPExampleState=SM_HOME;
                        }
                break;
            }
    #endif

                  // Make certain the socket can be written to
                  if(TCPIsPutReady(MySocket) < 125u)
                        break;

                  // Place the application protocol data into the transmit
buffer.  For this example, we are connected to an HTTP server, so we'll send
an HTTP GET request.
                  TCPPutROMString(MySocket, (ROM BYTE*)"GET ");
                  TCPPutROMString(MySocket, RemoteURL);
                  TCPPutROMString(MySocket, (ROM BYTE*)" HTTP/1.0\r\nHost:
");
                  TCPPutString(MySocket, ServerName);
                  TCPPutROMString(MySocket, (ROM BYTE*)"\r\nConnection:
close\r\n\r\n");

 /* sprintf((char *)buff,"GET /led/actions.php?fcn=get&id=1 HTTP/1.0\r\nHost:
%s \r\n", ServerName);
                        putsUART((char*)buff);
                        TCPPutROMString(MySocket, (ROM BYTE*)buff);
                        TCPPutROMString(MySocket, (ROM BYTE*)"Connection:
close\r\n\r\n");*/
                  // Send the packet
                  TCPFlush(MySocket);
                  GenericTCPExampleState++;
                  break;

            case SM_PROCESS_RESPONSE:
                  // Check to see if the remote node has disconnected from us
or sent us any application data
                  if(!TCPIsConnected(MySocket))
                  {
                        GenericTCPExampleState = SM_DISCONNECT;
                        // Do not break;  We might still have data in the TCP
RX FIFO waiting for us
                  }

                  // Get count of RX bytes waiting

                        length =TCPIsGetReady(MySocket);
                        found=0;
                        end=0;
                        places=0;
                        i=BUFFSIZE-1;
                    while(length){//Look through Data received for

                        if(found)
```

```
            break;
        length -= TCPGetArray(MySocket, vBuffer, i);

        for(j=0;j<BUFFSIZE;j++){
            if(found)
              break;
            switch (PARSE_STATES){
        case CHECK_0:
                    if(vBuffer[j]=='n')
                    PARSE_STATES++;
              else
                    PARSE_STATES=CHECK_0;
              break;
        case CHECK_1:
              if(vBuffer[j]=='g')
                    PARSE_STATES++;
              else
                    PARSE_STATES=CHECK_0;
              break;
        case CHECK_2:
              if(vBuffer[j]=='V')
                    PARSE_STATES++;
              else
                    PARSE_STATES=CHECK_0;
              break;
        case FAST_FORWARD_1:
              if(vBuffer[j]==':')
                    PARSE_STATES++;
              break;
        case FAST_FORWARD_2:
              if(vBuffer[j]=='"')
                    PARSE_STATES++;
              break;
        case PARSE:
              num[places]=vBuffer[j]-'0';

              if(num[places]<=9 && num[places] >= 0)
                    places++;
              else
                    found=1;
              break;
            }
        }
    }
    if(found){
        switch (places){
            case 1:
        value=num[0];
        break;
            case 2:
        value=num[0]*10+num[1];
        break;
            case 3:
        value=num[0]*100+num[1]+num[2];
        break;
        }
        *dim=value;
```

```
                        TCPDiscard(MySocket);
                      }
                    // Obtian and print the server reply
                        //                w = TCPIsGetReady(MySocket);
/*                  i = sizeof(vBuffer)-1;
                    vBuffer[i] = '\0';

                    while(w)
                    {
                        if(w < i)
                        {
                            i = w;
                            vBuffer[i] = '\0';
                        }

                        w -= TCPGetArray(MySocket, vBuffer, i);


                            #if defined(STACK_USE_UART)
                            //if(flag)
                        putsUART((char*)vBuffer);
                        #endif
                        #if defined(USE_LCD)
                        memcpypgm2ram((void*)&LCDText[16], (ROM void
*)vBuffer, 10);
                        LCDUpdate();
                        #endif

                        // putsUART is a blocking call which will slow down
the rest of the stack
                        // if we shovel the whole TCP RX FIFO into the serial
port all at once.
                        // Therefore, let's break out after only one chunk
most of the time.  The
                        // only exception is when the remote node disconncets
from us and we need to
                        // use up all the data before changing states.
                        if(GenericTCPExampleState == SM_PROCESS_RESPONSE)
                            break;
                    }
*/
                    break;

                case SM_DISCONNECT:
                    // Close the socket so it can be used by other modules
                    // For this application, we wish to stay connected, but
this state will still get entered if the remote server decides to disconnect
                    TCPDisconnect(MySocket);
                    MySocket = INVALID_SOCKET;
                        //Timer=TickGet();
                    GenericTCPExampleState = SM_DONE;
                    break;

                case SM_DONE:
                    // Do nothing unless the user pushes BUTTON1 and wants to
restart the whole connection/download process
//                  if(BUTTON1_IO == 0u)
```

```
//                      LATFbits.LATF3=1;
                         //if((TickGet()-Timer)>=30ul)
                          GenericTCPExampleState = SM_HOME;
                    break;
      }
}

#endif      //#if defined(STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE)
```

setServerValue.c

```
/*********************************************************************
 *
 *  Generic TCP Client Example Application
 *  Module for Microchip TCP/IP Stack
 *   -Implements an example HTTP client and should be used as a basis
 *      for creating new TCP client applications
 *     -Reference: None.  Hopefully AN833 in the future.
 *
 *********************************************************************
 * FileName:        GenericTCPClient.c
 * Dependencies:    TCP, DNS, ARP, Tick
 * Processor:       PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32
 * Compiler:        Microchip C32 v1.05 or higher
 *                           Microchip C30 v3.12 or higher
 *                           Microchip C18 v3.30 or higher
 *                           HI-TECH PICC-18 PRO 9.63PL2 or higher
 * Company:         Microchip Technology, Inc.
 *
 * Software License Agreement
 *
 * Copyright (C) 2002-2009 Microchip Technology Inc.  All rights
 * reserved.
 *
 * Microchip licenses to you the right to use, modify, copy, and
 * distribute:
 * (i)  the Software when embedded on a Microchip microcontroller or
 *      digital signal controller product ("Device") which is
 *      integrated into Licensee's product; or
 * (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
 *          ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device
 *          used in conjunction with a Microchip ethernet controller for
 *          the sole purpose of interfacing with the ethernet controller.
 *
 * You should refer to the license agreement accompanying this
 * Software for additional information regarding your rights and
 * obligations.
 *
 * THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT
 * WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT
 * LIMITATION, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A
 * PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL
 * MICROCHIP BE LIABLE FOR ANY INCIDENTAL, SPECIAL, INDIRECT OR
 * CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF
 * PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS
 * BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE
 * THEREOF), ANY CLAIMS FOR INDEMNITY OR CONTRIBUTION, OR OTHER
 * SIMILAR COSTS, WHETHER ASSERTED ON THE BASIS OF CONTRACT, TORT
```

```
 * (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR OTHERWISE.
 *
 *
 * Author               Date    Comment
 *~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 * Howard Schlunder     8/01/06     Original
 ********************************************************************/
#define __GENERICTCPCLIENT_C

#include "TCPIPConfig.h"

#if defined(STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE)

#include "TCPIP Stack/TCPIP.h"


// Defines the server to be accessed for this application
#ifdef WIFI_NET_TEST
static BYTE ServerName[] =    "www" WIFI_NET_TEST_DOMAIN;
#else
static BYTE ServerName[] =    "192.168.1.2";
#endif

// Defines the port to be accessed for this application
#if defined(STACK_USE_SSL_CLIENT)
    static WORD ServerPort = HTTPS_PORT;
      // Note that if HTTPS is used, the ServerName and URL
      // must change to an SSL enabled server.
#else
    static WORD ServerPort = HTTP_PORT;
#endif

// Defines the URL to be requested by this HTTP client
//static ROM BYTE RemoteURL[] = "/led/actions.php?id=1&fcn=set&val=25";


/*****************************************************************************
*
  Function:
      void GenericTCPClient(void)

  Summary:
      Implements a simple HTTP client (over TCP).

  Description:
      This function implements a simple HTTP client, which operates over TCP.
      The function is called periodically by the stack, and waits for BUTTON1
      to be pressed.  When the button is pressed, the application opens a TCP
      connection to an Internet search engine, performs a search for the word
      "Microchip" on "microchip.com", and prints the resulting HTML page to
      the UART.

      This example can be used as a model for many TCP and HTTP client
      applications.

  Precondition:
      TCP is initialized.
```

```
  Parameters:
      None

  Returns:
      None

*******************************************************************************/
BYTE setServerValue(BYTE dim)
{
        char                            print[15];
        BYTE                            flag=0;
      BYTE                    i;
        WORD                            j;
        BYTE                            k;
      WORD                    w;
        WORD                            length;
      BYTE                    vBuffer[200];
        const int BUFFSIZE=110;
      static DWORD            Timer;
      static TCP_SOCKET MySocket = INVALID_SOCKET;
      static enum _GenericTCPExampleState
      {
            SM_HOME = 0,
            SM_SOCKET_OBTAINED,
            #if defined(STACK_USE_SSL_CLIENT)
      SM_START_SSL,
      #endif
            SM_PROCESS_RESPONSE,
            SM_DISCONNECT,
            SM_DONE
      } GenericTCPExampleState = SM_DONE;

      switch(GenericTCPExampleState)
      {
            case SM_HOME:
                  // Connect a socket to the remote TCP server
                  //MySocket = TCPOpen((DWORD)(PTR_BASE)&ServerName[0],
TCP_OPEN_RAM_HOST, ServerPort, TCP_PURPOSE_GENERIC_TCP_CLIENT);
                        MySocket = TCPOpen((DWORD)&ServerName[0],
TCP_OPEN_RAM_HOST, ServerPort, TCP_PURPOSE_GENERIC_TCP_CLIENT);

                        //putrsUART((ROM char*)"\r\nConnecting ... \r\n ");
                  // Abort operation if no TCP socket of type
TCP_PURPOSE_GENERIC_TCP_CLIENT is available
                  // If this ever happens, you need to go add one to
TCPIPConfig.h
                  if(MySocket == INVALID_SOCKET)
                        break;


                  GenericTCPExampleState++;
                  Timer = TickGet();
                  break;

            case SM_SOCKET_OBTAINED:
                  // Wait for the remote server to accept our connection
```

```
request
                  if(!TCPIsConnected(MySocket))
                  {
                        // Time out if too much time is spent in this state
                        if(TickGet()-Timer > 5*TICK_SECOND)
                        {
                              // Close the socket so it can be used by other
modules
                              TCPDisconnect(MySocket);
                              MySocket = INVALID_SOCKET;
                              GenericTCPExampleState--;
                        }
                        break;
                  }
                        //putrsUART((ROM char*)"\r\nConnected\r\n ");
                  Timer = TickGet();


    #if defined (STACK_USE_SSL_CLIENT)
            if(!TCPStartSSLClient(MySocket,(void *)"thishost"))
                break;
                  GenericTCPExampleState++;
                  break;

        case SM_START_SSL:
            if (TCPSSLIsHandshaking(MySocket))
            {
                        if(TickGet()-Timer > 10*TICK_SECOND)
                        {
                              // Close the socket so it can be used by other
modules
                              TCPDisconnect(MySocket);
                              MySocket = INVALID_SOCKET;
                              GenericTCPExampleState=SM_HOME;
                        }
                break;
            }
    #endif

                  // Make certain the socket can be written to
                  if(TCPIsPutReady(MySocket) < 125u)
                        break;

                  // Place the application protocol data into the transmit
buffer.  For this example, we are connected to an HTTP server, so we'll send
an HTTP GET request.
                  TCPPutROMString(MySocket, (ROM BYTE*)"GET ");
                        sprintf((char
*)vBuffer,"/led/actions.php?fcn=set&id=1&val=%i",dim);
                  TCPPutString(MySocket, vBuffer);
                  TCPPutROMString(MySocket, (ROM BYTE*)" HTTP/1.0\r\nHost:
");
                  TCPPutString(MySocket, ServerName);
                  TCPPutROMString(MySocket, (ROM BYTE*)"\r\nConnection:
close\r\n\r\n");

                  TCPFlush(MySocket);
```

```
                GenericTCPExampleState++;
                break;

            case SM_PROCESS_RESPONSE:
                // Check to see if the remote node has disconnected from us
or sent us any application data
                // If application data is available, write it to the UART
                if(!TCPIsConnected(MySocket))
                {
                      GenericTCPExampleState = SM_DISCONNECT;
                      // Do not break;  We might still have data in the TCP
RX FIFO waiting for us
                }

                // Get count of RX bytes waiting
                w = TCPIsGetReady(MySocket);
                //i = sizeof(vBuffer)-1;
                /*vBuffer[i] = '\0';
                while(w)
                {
                      if(w < i)
                      {
                            i = w;
                            vBuffer[i] = '\0';
                      }

                      w -= TCPGetArray(MySocket, vBuffer, i);


                        #if defined(STACK_USE_UART)
                        //if(flag)
                      putsUART((char*)vBuffer);
                      #endif
                      #if defined(USE_LCD)
                      memcpypgm2ram((void*)&LCDText[16], (ROM void
*)vBuffer, 10);
                      LCDUpdate();
                      #endif

                      // putsUART is a blocking call which will slow down
the rest of the stack
                      // if we shovel the whole TCP RX FIFO into the serial
port all at once.
                      // Therefore, let's break out after only one chunk
most of the time.  The
                      // only exception is when the remote node disconncets
from us and we need to
                      // use up all the data before changing states.
                      if(GenericTCPExampleState == SM_PROCESS_RESPONSE)
                            break;
                }*/
                      flag=1;
                break;

            case SM_DISCONNECT:
                // Close the socket so it can be used by other modules
                // For this application, we wish to stay connected, but
```

```
this state will still get entered if the remote server decides to disconnect
                        //TCPDiscard(MySocket);
                    TCPDisconnect(MySocket);
                    MySocket = INVALID_SOCKET;
                    GenericTCPExampleState = SM_DONE;
                    break;

            case SM_DONE:
                    // Do nothing unless the user pushes BUTTON1 and wants to
restart the whole connection/download process
//                  if(BUTTON1_IO == 0u)
                        GenericTCPExampleState = SM_HOME;
                    break;
      }
        return flag;
}

#endif      //#if defined(STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE)
```

7.4.2 Configuration Files

HWP PICDN2_MRF24W.h

```
/*********************************************************************
 *
 *    Hardware specific definitions for:
 *    - PIC18F6722
 *    - MRF24W Wi-Fi PICtail
 *
 *********************************************************************
 * FileName:        HardwareProfile.h
 * Dependencies:    Compiler.h
 * Processor:       PIC18
 * Compiler:        Microchip C18 v3.36 or higher
 * Company:         Microchip Technology, Inc.
 *
 * Software License Agreement
 *
 * Copyright (C) 2002-2010 Microchip Technology Inc.  All rights
 * reserved.
 *
 * Microchip licenses to you the right to use, modify, copy, and
 * distribute:
 * (i)  the Software when embedded on a Microchip microcontroller or
 *      digital signal controller product ("Device") which is
 *      integrated into Licensee's product; or
 * (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
 *          ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device
 *          used in conjunction with a Microchip ethernet controller for
 *          the sole purpose of interfacing with the ethernet controller.
 *
 * You should refer to the license agreement accompanying this
 * Software for additional information regarding your rights and
 * obligations.
 *
 * THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT
 * WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT
 * LIMITATION, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A
```

```
 * PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL
 * MICROCHIP BE LIABLE FOR ANY INCIDENTAL, SPECIAL, INDIRECT OR
 * CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF
 * PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS
 * BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE
 * THEREOF), ANY CLAIMS FOR INDEMNITY OR CONTRIBUTION, OR OTHER
 * SIMILAR COSTS, WHETHER ASSERTED ON THE BASIS OF CONTRACT, TORT
 * (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR OTHERWISE.
 *
 *
 * Author               Date        Comment
 *~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 * Howard Schlunder           09/16/2010  Regenerated for specific boards
 ********************************************************************/
#ifndef HARDWARE_PROFILE_H
#define HARDWARE_PROFILE_H

#include "Compiler.h"

// Define a macro describing this hardware set up (used in other files)
#define MINE

// Set configuration fuses (but only in MainDemo.c where
THIS_IS_STACK_APPLICATION is defined)
#if defined(THIS_IS_STACK_APPLICATION)


/* 20 MHz crystal

 */
// CONFIG1H

#pragma config OSC = INTIO67         // Oscillator Selection bits (HS
oscillator)
#pragma config FCMEN = OFF      // Fail-Safe Clock Monitor Enable bit (Fail-
Safe Clock Monitor disabled)
#pragma config IESO = OFF       // Internal/External Oscillator Switchover
bit (Two-Speed Start-up disabled)


// CONFIG2H
#pragma config WDT = OFF       // Watchdog Timer (WDT disabled (control is
placed on the SWDTEN bit))
#pragma config WDTPS = 32768    // Watchdog Timer Postscale Select bits
(1:32768)

// CONFIG3H
#pragma config CCP2MX = PORTE   // CCP2 MUX bit (ECCP2 input/output is
multiplexed with RC1)
#pragma config MCLRE = OFF       // MCLR Pin Enable bit (MCLR pin enabled;
RG5 input pin disabled)

#pragma config LVP = OFF
#endif


// Clock frequency values
```

```
// These directly influence timed events using the Tick module.  They also
are used for UART and SPI baud rate generation.
#define GetSystemClock()            (16000000ul)                  // Hz
#define GetInstructionClock() (GetSystemClock()/4)    // Normally
GetSystemClock()/4 for PIC18, GetSystemClock()/2 for PIC24/dsPIC, and
GetSystemClock()/1 for PIC32.  Might need changing if using Doze modes.
#define GetPeripheralClock()  (GetSystemClock()/4)    // Normally
GetSystemClock()/4 for PIC18, GetSystemClock()/2 for PIC24/dsPIC, and
GetSystemClock()/1 for PIC32.  Divisor may be different if using a PIC32
since it's configurable.


// Hardware I/O pin mappings

// MRF24W Wi-Fi PICtail I/O pins
#define WF_CS_TRIS                  (TRISCbits.TRISC1)
#define WF_SDI_TRIS                 (TRISCbits.TRISC4)
#define WF_SCK_TRIS                 (TRISCbits.TRISC3)
#define WF_SDO_TRIS                 (TRISCbits.TRISC5)
#define WF_RESET_TRIS                   (TRISFbits.TRISF1)
#define WF_RESET_IO                 (LATFbits.LATF1)
#define WF_INT_TRIS                 (TRISBbits.TRISB1)
#define WF_INT_IO             (PORTBbits.RB1)
#define WF_CS_IO              (LATCbits.LATC1)
#define WF_HIBERNATE_TRIS               (TRISCbits.TRISC2)
#define     WF_HIBERNATE_IO                 (PORTCbits.RC2)
#define WF_INT_EDGE                 (INTCON2bits.INTEDG1)
#define WF_INT_IE             (INTCON3bits.INT1IE)
#define WF_INT_IF             (INTCON3bits.INT1IF)
#define WF_INT_IP                       (INTCON3bits.INT1IP)
#define WF_SPI_IF             (PIR1bits.SSPIF)
#define WF_SSPBUF             (SSP1BUF)
#define WF_SPISTAT                  (SSP1STAT)
#define WF_SPISTATbits                  (SSP1STATbits)
#define WF_SPICON1                  (SSP1CON1)
#define WF_SPICON1bits                  (SSP1CON1bits)
#define WF_SPICON2                  (SSP1CON2)
#define WF_SPI_IE             (PIE1bits.SSPIE)
#define WF_SPI_IP             (IPR1bits.SSPIP)
// UART mapping functions for consistent API names across 8-bit and 16 or
// 32 bit compilers.  For simplicity, everything will use "UART" instead
// of USART/EUSART/etc.

#define BusyUART()                  BusyUSART()
#define CloseUART()                 CloseUSART()
#define ConfigIntUART(a)      ConfigIntUSART(a)
#define DataRdyUART()         DataRdyUSART()
#define OpenUART(a,b,c)       OpenUSART(a,b,c)
#define ReadUART()                  ReadUSART()
#define WriteUART(a)          WriteUSART(a)
#define getsUART(a,b,c)       getsUSART(b,a)
#define putsUART(a)                 putsUSART(a)
#define getcUART()                  ReadUSART()
#define putcUART(a)                 WriteUSART(a)
#define putrsUART(a)          putrsUSART((far rom char*)a)

#endif // #ifndef HARDWARE_PROFILE_H
```

TCPIP MRF24W.h

```
/*********************************************************************
 *
 *    Microchip TCP/IP Stack Demo Application Configuration Header
 *
 *********************************************************************
 * FileName:        TCPIPConfig.h
 * Dependencies:    Microchip TCP/IP Stack
 * Processor:       PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32
 * Compiler:        Microchip C32 v1.10 or higher
 *                           Microchip C30 v3.12 or higher
 *                           Microchip C18 v3.34 or higher
 *                           HI-TECH PICC-18 PRO 9.63PL2 or higher
 * Company:         Microchip Technology, Inc.
 *
 * Software License Agreement
 *
 * Copyright (C) 2002-2010 Microchip Technology Inc.  All rights
 * reserved.
 *
 * Microchip licenses to you the right to use, modify, copy, and
 * distribute:
 * (i)  the Software when embedded on a Microchip microcontroller or
 *      digital signal controller product ("Device") which is
 *      integrated into Licensee's product; or
 * (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
 *          ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device
 *          used in conjunction with a Microchip ethernet controller for
 *          the sole purpose of interfacing with the ethernet controller.
 *
 * You should refer to the license agreement accompanying this
 * Software for additional information regarding your rights and
 * obligations.
 *
 * THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT
 * WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT
 * LIMITATION, ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A
 * PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL
 * MICROCHIP BE LIABLE FOR ANY INCIDENTAL, SPECIAL, INDIRECT OR
 * CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST OF
 * PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS
 * BY THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE
 * THEREOF), ANY CLAIMS FOR INDEMNITY OR CONTRIBUTION, OR OTHER
 * SIMILAR COSTS, WHETHER ASSERTED ON THE BASIS OF CONTRACT, TORT
 * (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR OTHERWISE.
 *
 *
 * V5.36   ---- STACK_USE_MPFS support has been removed
 ********************************************************************/
#ifndef __TCPIPCONFIG_H
#define __TCPIPCONFIG_H

#include "GenericTypeDefs.h"
#include "Compiler.h"
#define GENERATED_BY_TCPIPCONFIG "Version 1.0.3383.23374"
```

```
#if defined(__C32__)
//#define WIFI_NET_TEST
/*
* This is only for Wi-Fi internal test.
* Only "Demo App" can run this test.
*/
#define WIFI_NET_TEST_DOMAIN  ".wpdsw.com"
#endif

// =============================================================================
//   Application Options
// =============================================================================

/* Application Level Module Selection
 *   Uncomment or comment the following lines to enable or
 *   disabled the following high-level application modules.
 *
 * If certain compilations are enabled (eg
STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE),
 * check whether the files (eg GenericTCPClient.c) are located in folder (eg
\TCPIP\WiFi EZConfig).
 * You may need to copy such files from the Demo App or WiFi Console folder.
 *
 */
//#define STACK_USE_UART                                // Application demo
using UART for IP address display and stack configuration
//#define STACK_USE_UART2TCP_BRIDGE       // UART to TCP Bridge application
example
//#define STACK_USE_IP_GLEANING
#define STACK_USE_ICMP_SERVER             // Ping query and response
capability
//#define STACK_USE_ICMP_CLIENT                 // Ping transmission
capability
//#define STACK_USE_HTTP2_SERVER                // New HTTP server with POST,
Cookies, Authentication, etc.
//#define STACK_USE_SSL_SERVER                  // SSL server socket support
(Requires SW300052)
//#define STACK_USE_SSL_CLIENT                  // SSL client socket support
(Requires SW300052)
//#define STACK_USE_AUTO_IP               // Dynamic link-layer IP address
automatic configuration protocol
#define STACK_USE_DHCP_CLIENT             // Dynamic Host Configuration
Protocol client for obtaining IP address and other parameters
//#define STACK_USE_DHCP_SERVER                 // Single host DHCP server
//#define STACK_USE_FTP_SERVER                  // File Transfer Protocol
(old)
//#define STACK_USE_SMTP_CLIENT                 // Simple Mail Transfer
Protocol for sending email
//#define STACK_USE_SNMP_SERVER                 // Simple Network Management
Protocol v2C Community Agent
//#define STACK_USE_SNMPV3_SERVER               // Simple Network Management
Protocol v3 Agent
//#define STACK_USE_TFTP_CLIENT                 // Trivial File Transfer
Protocol client
#define STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE    // HTTP Client example in
GenericTCPClient.c
```

```
//#define STACK_USE_GENERIC_TCP_SERVER_EXAMPLE  // ToUpper server example in
GenericTCPServer.c
//#define STACK_USE_TELNET_SERVER               // Telnet server
//*#define STACK_USE_ANNOUNCE                   // Microchip Embedded
Ethernet Device Discoverer server/client
//*#define STACK_USE_DNS                              // Domain Name Service
Client for resolving hostname strings to IP addresses
//#define STACK_USE_DNS_SERVER                  // Domain Name Service Server
for redirection to the local device
//*#define STACK_USE_NBNS                             // NetBIOS Name Service
Server for repsonding to NBNS hostname broadcast queries
//*#define STACK_USE_REBOOT_SERVER               // Module for resetting this
PIC remotely.  Primarily useful for a Bootloader.
//#define STACK_USE_SNTP_CLIENT                 // Simple Network Time
Protocol for obtaining current date/time from Internet
//#define STACK_USE_UDP_PERFORMANCE_TEST  // Module for testing UDP TX
performance characteristics.  NOTE: Enabling this will cause a huge amount of
UDP broadcast packets to flood your network on the discard port.  Use care
when enabling this on production networks, especially with VPNs (could tunnel
broadcast traffic across a limited bandwidth connection).
//#define STACK_USE_TCP_PERFORMANCE_TEST  // Module for testing TCP TX
performance characteristics
//#define STACK_USE_DYNAMICDNS_CLIENT           // Dynamic DNS client updater
module
//#define STACK_USE_BERKELEY_API                // Berekely Sockets APIs are
available
//#define STACK_USE_ZEROCONF_LINK_LOCAL   // Zeroconf IPv4 Link-Local
Addressing
//#define STACK_USE_ZEROCONF_MDNS_SD            // Zeroconf mDNS and mDNS
service discovery


// =============================================================================
//   Data Storage Options
// =============================================================================

/* MPFS Configuration
 *   MPFS is automatically included when required for other
 *   applications.  If your custom application requires it
 *   otherwise, uncomment the appropriate selection.
 */
//#define STACK_USE_MPFS2

/* MPFS Storage Location
 *   If html pages are stored in internal program memory,
 *   comment both MPFS_USE_EEPROM and MPFS_USE_SPI_FLASH, then
 *   include an MPFS image (.c or .s file) in the project.
 *   If html pages are stored in external memory, uncomment the
 *   appropriate definition.
 *
 *   Supported serial flash parts include the SST25VFxxxB series.
 */
//#define MPFS_USE_EEPROM
//#define MPFS_USE_SPI_FLASH

/* EEPROM Addressing Selection
 *   If using the 1Mbit EEPROM, uncomment this line
```

```
 */
//#define USE_EEPROM_25LC1024

/* EEPROM Reserved Area
 *   Number of EEPROM bytes to be reserved before MPFS storage starts.
 *   These bytes host application configurations such as IP Address,
 *   MAC Address, and any other required variables.
 *
 *   For MPFS Classic, this setting must match the Reserved setting
 *     on the Advanced Settings page of the MPFS2 Utility.
 */
#define MPFS_RESERVE_BLOCK                      (205ul)

/* MPFS File Handles
 *   Maximum number of simultaneously open MPFS2 files.
 *   For MPFS Classic, this has no effect.
 */
#define MAX_MPFS_HANDLES                        (7ul)


// ============================================================================
//   Network Addressing Options
// ============================================================================

/* Default Network Configuration
 *   These settings are only used if data is not found in EEPROM.
 *   To clear EEPROM, hold BUTTON0, reset the board, and continue
 *   holding until the LEDs flash.  Release, and reset again.
 */
#define MY_DEFAULT_HOST_NAME              "MCHPBOARD"

#define MY_DEFAULT_MAC_BYTE1            (0x00)  // Use the default of 00-04-
A3-00-00-00
#define MY_DEFAULT_MAC_BYTE2            (0x04)  // if using an ENCX24J600,
MRF24WB0M, or
#define MY_DEFAULT_MAC_BYTE3            (0xA3)  // PIC32MX6XX/7XX internal
Ethernet
#define MY_DEFAULT_MAC_BYTE4            (0x00)  // controller and wish to use
the
#define MY_DEFAULT_MAC_BYTE5            (0x00)  // internal factory
programmed MAC
#define MY_DEFAULT_MAC_BYTE6            (0x00)  // address instead.

#define MY_DEFAULT_IP_ADDR_BYTE1        (169ul)
#define MY_DEFAULT_IP_ADDR_BYTE2        (254ul)
#define MY_DEFAULT_IP_ADDR_BYTE3        (1ul)
#define MY_DEFAULT_IP_ADDR_BYTE4        (1ul)

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

```
#define MY_DEFAULT_PRIMARY_DNS_BYTE1      (169ul)
#define MY_DEFAULT_PRIMARY_DNS_BYTE2      (254ul)
#define MY_DEFAULT_PRIMARY_DNS_BYTE3      (1ul)
#define MY_DEFAULT_PRIMARY_DNS_BYTE4      (1ul)

#define MY_DEFAULT_SECONDARY_DNS_BYTE1    (0ul)
#define MY_DEFAULT_SECONDARY_DNS_BYTE2    (0ul)
#define MY_DEFAULT_SECONDARY_DNS_BYTE3    (0ul)
#define MY_DEFAULT_SECONDARY_DNS_BYTE4    (0ul)

// ========================================================================
//   PIC32MX7XX/6XX MAC Layer Options
//   If not using a PIC32MX7XX/6XX device, ignore this section.
// ========================================================================
#define     ETH_CFG_LINK                  0           // set to 1 if you need
to config the link to specific following parameters
                                                      // otherwise the
default connection will be attempted
                                                      // depending on
the selected PHY
      #define     ETH_CFG_AUTO            1           // use auto negotiation
      #define     ETH_CFG_10              1           // use/advertise 10
Mbps capability
      #define     ETH_CFG_100             1           // use/advertise 100
Mbps capability
      #define     ETH_CFG_HDUPLEX         1           // use/advertise half
duplex capability
      #define     ETH_CFG_FDUPLEX         1           // use/advertise full
duplex capability
      #define     ETH_CFG_AUTO_MDIX 1           // use/advertise auto MDIX
capability
      #define     ETH_CFG_SWAP_MDIX 1           // use swapped MDIX. else
normal MDIX

#define EMAC_TX_DESCRIPTORS         2           // number of the TX
descriptors to be created
#define EMAC_RX_DESCRIPTORS         8           // number of the RX
descriptors and RX buffers to be created

#define     EMAC_RX_BUFF_SIZE       1536  // size of a RX buffer. should be
multiple of 16
                                                      // this is the
size of all receive buffers processed by the ETHC
                                                      // The size
should be enough to accomodate any network received packet
                                                      // If the packets
are larger, they will have to take multiple RX buffers
                                                      // The current
implementation does not handle this situation right now and the packet is
discarded.


// ========================================================================
//   Transport Layer Options
// ========================================================================
```

```
/* Transport Layer Configuration
 *   The following low level modules are automatically enabled
 *   based on module selections above.  If your custom module
 *   requires them otherwise, enable them here.
 */
//#define STACK_USE_TCP
//#define STACK_USE_UDP

/* Client Mode Configuration
 *   Uncomment following line if this stack will be used in CLIENT
 *   mode.  In CLIENT mode, some functions specific to client operation
 *   are enabled.
 */
#define STACK_CLIENT_MODE

/* TCP Socket Memory Allocation
 *   TCP needs memory to buffer incoming and outgoing data.  The
 *   amount and medium of storage can be allocated on a per-socket
 *   basis using the example below as a guide.
 */
      // Allocate how much total RAM (in bytes) you want to allocate
      // for use by your TCP TCBs, RX FIFOs, and TX FIFOs.
      #define TCP_ETH_RAM_SIZE                              (8192ul)
      #define TCP_PIC_RAM_SIZE                              (0ul)
      #define TCP_SPI_RAM_SIZE                              (0ul)
      #define TCP_SPI_RAM_BASE_ADDRESS                (0x00)

      // Define names of socket types
      #define TCP_SOCKET_TYPES
            #define TCP_PURPOSE_GENERIC_TCP_CLIENT 0
            #define TCP_PURPOSE_GENERIC_TCP_SERVER 1
            #define TCP_PURPOSE_TELNET 2
            #define TCP_PURPOSE_FTP_COMMAND 3
            #define TCP_PURPOSE_FTP_DATA 4
            #define TCP_PURPOSE_TCP_PERFORMANCE_TX 5
            #define TCP_PURPOSE_TCP_PERFORMANCE_RX 6
            #define TCP_PURPOSE_UART_2_TCP_BRIDGE 7
            #define TCP_PURPOSE_HTTP_SERVER 8
            #define TCP_PURPOSE_DEFAULT 9
            #define TCP_PURPOSE_BERKELEY_SERVER 10
            #define TCP_PURPOSE_BERKELEY_CLIENT 11
      #define END_OF_TCP_SOCKET_TYPES

      #if defined(__TCP_C)
            // Define what types of sockets are needed, how many of
            // each to include, where their TCB, TX FIFO, and RX FIFO
            // should be stored, and how big the RX and TX FIFOs should
            // be.  Making this initializer bigger or smaller defines
            // how many total TCP sockets are available.
            //
            // Each socket requires up to 56 bytes of PIC RAM and
            // 48+(TX FIFO size)+(RX FIFO size) bytes of TCP_*_RAM each.
            //
            // Note: The RX FIFO must be at least 1 byte in order to
            // receive SYN and FIN messages required by TCP.  The TX
            // FIFO can be zero if desired.
            #define TCP_CONFIGURATION
```

```
            ROM struct
            {
                  BYTE vSocketPurpose;
                  BYTE vMemoryMedium;
                  WORD wTXBufferSize;
                  WORD wRXBufferSize;
            } TCPSocketInitializer[] =
            {
#ifdef STACK_USE_GENERIC_TCP_CLIENT_EXAMPLE
                  {TCP_PURPOSE_GENERIC_TCP_CLIENT, TCP_ETH_RAM, 125, 100},
#endif
#ifdef STACK_USE_GENERIC_TCP_SERVER_EXAMPLE
                  {TCP_PURPOSE_GENERIC_TCP_SERVER, TCP_ETH_RAM, 20, 20},
#endif
                  //{TCP_PURPOSE_TELNET, TCP_ETH_RAM, 200, 150},
                  //{TCP_PURPOSE_TELNET, TCP_ETH_RAM, 200, 150},
                  //{TCP_PURPOSE_TELNET, TCP_ETH_RAM, 200, 150},
                  //{TCP_PURPOSE_FTP_COMMAND, TCP_ETH_RAM, 100, 40},
                  //{TCP_PURPOSE_FTP_DATA, TCP_ETH_RAM, 0, 128},
                  {TCP_PURPOSE_TCP_PERFORMANCE_TX, TCP_ETH_RAM, 200, 1},
                  //{TCP_PURPOSE_TCP_PERFORMANCE_RX, TCP_ETH_RAM, 40, 1500},
#ifdef STACK_USE_UART2TCP_BRIDGE
                  {TCP_PURPOSE_UART_2_TCP_BRIDGE, TCP_ETH_RAM, 256, 256},
#endif
                  {TCP_PURPOSE_HTTP_SERVER, TCP_ETH_RAM, 1000, 1000},
                  {TCP_PURPOSE_HTTP_SERVER, TCP_ETH_RAM, 1000, 1000},
                  //{TCP_PURPOSE_DEFAULT, TCP_ETH_RAM, 1000, 1000},
                  //{TCP_PURPOSE_BERKELEY_SERVER, TCP_ETH_RAM, 25, 20},
                  //{TCP_PURPOSE_BERKELEY_SERVER, TCP_ETH_RAM, 25, 20},
                  //{TCP_PURPOSE_BERKELEY_SERVER, TCP_ETH_RAM, 25, 20},
                  //{TCP_PURPOSE_BERKELEY_CLIENT, TCP_ETH_RAM, 125, 100},
            };
            #define END_OF_TCP_CONFIGURATION
      #endif

/* UDP Socket Configuration
 *   Define the maximum number of available UDP Sockets, and whether
 *   or not to include a checksum on packets being transmitted.
 */
#define MAX_UDP_SOCKETS     (8u)
//#define UDP_USE_TX_CHECKSUM       // This slows UDP TX performance by
nearly 50%, except when using the ENCX24J600 or PIC32MX6XX/7XX, which have a
super fast DMA and incurs virtually no speed pentalty.


/* Berkeley API Sockets Configuration
 *   Note that each Berkeley socket internally uses one TCP or UDP socket
 *   defined by MAX_UDP_SOCKETS and the TCPSocketInitializer[] array.
 *   Therefore, this number MUST be less than or equal to MAX_UDP_SOCKETS +
the
 *   number of TCP sockets defined by the TCPSocketInitializer[] array
 *   (i.e. sizeof(TCPSocketInitializer)/sizeof(TCPSocketInitializer[0])).
 *   This define has no effect if STACK_USE_BERKELEY_API is not defined and
 *   Berkeley Sockets are disabled.  Set this value as low as your
application
 *   requires to avoid waisting RAM.
 */
```

```
#define BSD_SOCKET_COUNT (5u)


// =====================================================================
//   Application-Specific Options
// =====================================================================

// -- HTTP2 Server options ---------------------------------------------

      // Maximum numbers of simultaneous HTTP connections allowed.
      // Each connection consumes 2 bytes of RAM and a TCP socket
      #define MAX_HTTP_CONNECTIONS  (2u)

      // Optional setting to use PIC RAM instead of Ethernet/Wi-Fi RAM for
      // storing HTTP Connection Context variables (HTTP_CONN structure for
each
      // HTTP connection).  Undefining this macro results in the Ethernet/Wi-
Fi
      // RAM being used (minimum PIC RAM usage, lower performance).  Defining
      // this macro results in PIC RAM getting used (higher performance, but
uses
      // PIC RAM).  This option should not be enabled on PIC18 devices.  The
      // performance increase of having this option defined is only apparent
when
      // the HTTP server is servicing multiple connections simultaneously.
      //#define HTTP_SAVE_CONTEXT_IN_PIC_RAM

      // Indicate what file to serve when no specific one is requested
      #define HTTP_DEFAULT_FILE           "index.htm"
      #define HTTPS_DEFAULT_FILE          "index.htm"
      #define HTTP_DEFAULT_LEN            (10u)       // For buffer overrun
protection.
                                                                        //
Set to longest length of above two strings.

      // Configure MPFS over HTTP updating
      // Comment this line to disable updating via HTTP
      #define HTTP_MPFS_UPLOAD            "mpfsupload"
      //#define HTTP_MPFS_UPLOAD_REQUIRES_AUTH  // Require password for MPFS
uploads
            // Certain firewall and router combinations cause the MPFS2
Utility to fail
            // when uploading.  If this happens, comment out this definition.

      // Define which HTTP modules to use
      // If not using a specific module, comment it to save resources
      #define HTTP_USE_POST                           // Enable POST support
      #define HTTP_USE_COOKIES                        // Enable cookie
support
      #define HTTP_USE_AUTHENTICATION                 // Enable basic
authentication support

      //#define HTTP_NO_AUTH_WITHOUT_SSL        // Uncomment to require SSL
before requesting a password

    // Define the listening port for the HTTP server
      #define HTTP_PORT               (80u)
```

```
    // Define the listening port for the HTTPS server (if
STACK_USE_SSL_SERVER is enabled)
      #define HTTPS_PORT                        (443u)

    // Define the maximum data length for reading cookie and GET/POST
arguments (bytes)
      #define HTTP_MAX_DATA_LEN           (100u)

    // Define the minimum number of bytes free in the TX FIFO before
executing callbacks
      #define HTTP_MIN_CALLBACK_FREE      (16u)

      #define STACK_USE_HTTP_APP_RECONFIG       // Use the AppConfig web page
in the Demo App (~2.5kb ROM, ~0b RAM)
      #define STACK_USE_HTTP_MD5_DEMO                 // Use the MD5 Demo web
page (~5kb ROM, ~160b RAM)
      #define STACK_USE_AUTOUPDATE_HTTPSERVER  //Using http to upload Patch
to wifi Module
      //#define STACK_USE_HTTP_EMAIL_DEMO       // Use the e-mail demo web
page

// -- SSL Options ----------------------------------------------------------

      #define MAX_SSL_CONNECTIONS         (2ul) // Maximum connections via
SSL
      #define MAX_SSL_SESSIONS            (2ul) // Max # of cached SSL
sessions
      #define MAX_SSL_BUFFERS             (4ul) // Max # of SSL buffers (2
per socket)
      #define MAX_SSL_HASHES              (5ul) // Max # of SSL hashes  (2
per, plus 1 to avoid deadlock)

      // Bits in SSL RSA key.  This parameter is used for SSL sever
      // connections only.  The only valid value is 512 bits (768 and 1024
      // bits do not work at this time).  Note, however, that SSL client
      // operations do currently work up to 1024 bit RSA key length.
      #define SSL_RSA_KEY_SIZE            (512ul)


// -- Telnet Options -------------------------------------------------------

      // Number of simultaneously allowed Telnet sessions.  Note that you
      // must have an equal number of TCP_PURPOSE_TELNET type TCP sockets
      // declared in the TCPSocketInitializer[] array above for multiple
      // connections to work.  If fewer sockets are available than this
      // definition, then the the lesser of the two quantities will be the
      // actual limit.
      #define MAX_TELNET_CONNECTIONS      (1u)

      // Default local listening port for the Telnet server.  Port 23 is the
      // protocol default.
      #define TELNET_PORT                       23

      // Default local listening port for the Telnet server when SSL secured.
      // Port 992 is the telnets protocol default.
      #define TELNETS_PORT                992
```

```
      // Force all connecting clients to be SSL secured and connected via
      // TELNETS_PORT.  Connections on port TELNET_PORT will be ignored.  If
      // STACK_USE_SSL_SERVER is undefined, this entire setting is ignored
      // (server will accept unsecured connections on TELNET_PORT and won't
even
      // listen on TELNETS_PORT).
      //#define TELNET_REJECT_UNSECURED

      // Default username and password required to login to the Telnet
server.
      #define TELNET_USERNAME             "admin"
      #define TELNET_PASSWORD             "microchip"


// -- SNMP Options ------------------------------------------------------

      // Comment following line if SNMP TRAP support is needed
      //#define SNMP_TRAP_DISABLED

      //#define SNMP_STACK_USE_V2_TRAP
      #if defined(STACK_USE_SNMPV3_SERVER)
            #define SNMP_V1_V2_TRAP_WITH_SNMPV3
      #endif

      // This is the maximum length for community string.
      // Application must ensure that this length is observed.
      // SNMP module adds one byte extra after SNMP_COMMUNITY_MAX_LEN
      // for adding '\0' NULL character.
      #define SNMP_COMMUNITY_MAX_LEN      (8u)
#ifdef WIFI_NET_TEST
      #define SNMP_MAX_COMMUNITY_SUPPORT  1u
#else
      #define SNMP_MAX_COMMUNITY_SUPPORT  (3u)
#endif
      #define NOTIFY_COMMUNITY_LEN        (SNMP_COMMUNITY_MAX_LEN)

      // Default SNMPv2C community names.  These can be overridden at run
time if
      // alternate strings are present in external EEPROM or Flash (actual
      // strings are stored in AppConfig.readCommunity[] and
      // AppConfig.writeCommunity[] arrays).  These strings are case
sensitive.
      // An empty string means disabled (not matchable).
      // For application security, these default community names should not
be
      // used, but should all be disabled to force the end user to select
unique
      // community names.  These defaults are provided only to make it easier
to
      // start development.  Specifying more strings than
      // SNMP_MAX_COMMUNITY_SUPPORT will result in the later strings being
      // ignored (but still wasting program memory).  Specifying fewer
strings is
      // legal, as long as at least one is present.  A string larger than
      // SNMP_COMMUNITY_MAX_LEN bytes will be ignored.
      #define SNMP_READ_COMMUNITIES       {"public", "read", ""}
```

```
      #define END_OF_SNMP_READ_COMMUNITIES
      #define SNMP_WRITE_COMMUNITIES      {"private", "write", "public"}
      #define END_OF_SNMP_WRITE_COMMUNITIES
#endif
```

WF_Config.h

```
/*****************************************************************************
**

 MRF24W Driver Customization
 Module for Microchip TCP/IP Stack
  -Provides access to MRF24W WiFi controller
  -Reference: MRF24W Data sheet, IEEE 802.11 Standard

*****************************************************************************
**
 FileName:        WF_Config.h
 Dependencies:    TCP/IP Stack header files
 Processor:       PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32
 Compiler:        Microchip C32 v1.10b or higher
                        Microchip C30 v3.22 or higher
                        Microchip C18 v3.34 or higher
 Company:         Microchip Technology, Inc.

 Software License Agreement

 Copyright (C) 2002-2010 Microchip Technology Inc.  All rights reserved.

 Microchip licenses to you the right to use, modify, copy, and distribute:
 (i)  the Software when embedded on a Microchip microcontroller or digital
      signal controller product ("Device") which is integrated into
      Licensee's product; or
 (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
      ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device used in
        conjunction with a Microchip ethernet controller for the sole purpose
        of interfacing with the ethernet controller.

 You should refer to the license agreement accompanying this Software for
 additional information regarding your rights and obligations.

 THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
 KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY
 OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND
 NON-INFRINGEMENT. IN NO EVENT SHALL MICROCHIP BE LIABLE FOR ANY INCIDENTAL,
 SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST
 OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS BY
 THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), ANY CLAIMS
 FOR INDEMNITY OR CONTRIBUTION, OR OTHER SIMILAR COSTS, WHETHER ASSERTED ON
 THE BASIS OF CONTRACT, TORT (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR
 OTHERWISE.


 Author                         Date        Comment
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~
 KH                  27 Jan 2010 Created for MRF24W
*****************************************************************************
```

```
*/


#ifndef __WF_CONFIG_H_
#define __WF_CONFIG_H_

/*
******************************************************************************
****************************
*                                          DEFINES
******************************************************************************
****************************
*/

#define WF_TCPIP_DEMO



/******************************************************************************
*/
/******************************************************************************
*/
/*                              WIFI SECURITY COMPILE-TIME DEFAULTS
*/
/******************************************************************************
*/
/******************************************************************************
*/
// Security modes available on WiFi network:
//   WF_SECURITY_OPEN                      : No security
//   WF_SECURITY_WEP_40                    : WEP Encryption using 40 bit keys
//   WF_SECURITY_WEP_104                   : WEP Encryption using 104 bit
keys
//   WF_SECURITY_WPA_WITH_KEY              : WPA-PSK Personal where binary
key is given to MRF24W
//   WF_SECURITY_WPA_WITH_PASS_PHRASE      : WPA-PSK Personal where
passphrase is given to MRF24W and it calculates the binary key
//   WF_SECURITY_WPA2_WITH_KEY             : WPA2-PSK Personal where binary
key is given to MRF24W
//   WF_SECURITY_WPA2_WITH_PASS_PHRASE     : WPA2-PSK Personal where
passphrase is given to MRF24W and it calculates the binary key
//   WF_SECURITY_WPA_AUTO_WITH_KEY         : WPA-PSK Personal or WPA2-PSK
Personal where binary key is given and MRF24W will
//                                           connect at highest level AP
supports (WPA or WPA2)
//   WF_SECURITY_WPA_AUTO_WITH_PASS_PHRASE : WPA-PSK Personal or WPA2-PSK
Personal where passphrase is given to MRF24W and it
//                                           calculates the binary key and
connects at highest level AP supports (WPA or WPA2)
//   WF_SECURITY_WPS_PUSH_BUTTON             : WPS push button method
//   WF_SECURITY_WPS_PIN                          : WPS PIN method

#define CFG_WF_INFRASTRUCTURE 1
#define CFG_WF_ADHOC          2
#define CFG_WF_P2P            3

//#define MY_DEFAULT_NETWORK_TYPE             CFG_WF_INFRASTRUCTURE  /*
```

```
CFG_WF_INFRASTRUCTURE, CFG_WF_ADHOC, CFG_WF_P2P */
#define MY_DEFAULT_NETWORK_TYPE CFG_WF_INFRASTRUCTURE
#define MY_DEFAULT_DOMAIN                   WF_DOMAIN_FCC
#define MY_DEFAULT_LIST_RETRY_COUNT         WF_RETRY_FOREVER            /*
Number of times to try to connect to the SSID when using Infrastructure
network type */

#if MY_DEFAULT_NETWORK_TYPE == CFG_WF_ADHOC
#undef MY_DEFAULT_LIST_RETRY_COUNT
#define MY_DEFAULT_LIST_RETRY_COUNT         3
#endif

#if MY_DEFAULT_NETWORK_TYPE == CFG_WF_INFRASTRUCTURE ||
MY_DEFAULT_NETWORK_TYPE == CFG_WF_ADHOC
    #define MY_DEFAULT_WIFI_SECURITY_MODE       WF_SECURITY_OPEN
    #define MY_DEFAULT_SCAN_TYPE                WF_ACTIVE_SCAN
/* WF_ACTIVE_SCAN or WF_PASSIVE_SCAN */
    #define MY_DEFAULT_BEACON_TIMEOUT           (40)
/* Number of beacon periods          */
    #define MY_DEFAULT_SSID_NAME                "JYURKOVICH-VAIO_Network_1"
/* if WF_SECURITY_WPS_PUSH_BUTTON must be "" (empty string) */
    #define MY_DEFAULT_CHANNEL_LIST         {1,2,3,4,5,6,7,8,9,10,11}   /*
Default channel list for FCC */

    /* Select Infrastructure Power Save Mode */
    #define MY_DEFAULT_PS_POLL                      WF_DISABLED /* PS is
not supported in Adhoc */
    #if !defined(MRF24WG)
    /* #define WF_AGGRESSIVE_PS */  /* WARNING !!! : This only can work with
1209 module FW version or later.
                                    * If you use the earlier version such as
1207 or 1205, then you should not define this.
                                    * Defining this will lead ASSERT problem
with old module FW.
                                    */
    #endif

/* Warning !!! Please note that :
* RF Module FW has a built-in connection manager, and it is enabled by
default.
* So if you want to run your own connection manager in host stack application
side,
* then you should disable the module connection manager to avoid some
possible conflict
* between the two.  Especially these two APIs can be affected if you do not
disable it.
* A) UINT16 WF_CMDisconnect(void)
* B) UINT16 WF_Scan(UINT8 CpId)
* If some conflict occurs then these APIs will return failure.
* Furthermore if you use old MRF24WB FW version, older than 120C, then
* it can cause fatal issue in module FW side such as FW crash.
* So for simplicity, if you want to run your own connection manager actively,
* we strongly recommend to disable the module connection manager, and this
* #define is make that thing possible. Just un-comment it to do so !
*/
//#define DISABLE_MODULE_FW_CONNECT_MANAGER_IN_INFRASTRUCTURE
```

```
/*-------------------------------------------------------*/
/* else if starting this demo in P2P(Wi-Fi Direct) mode */
/*-------------------------------------------------------*/
#elif MY_DEFAULT_NETWORK_TYPE == CFG_WF_P2P
#if defined (MRF24WG)
      /*
      * Wi-Fi Direct has been validated with Samsung Galaxy Tab 2 7.0 (
Android 4.0.3, Ice cream Sandwitch)
      * a flag-ship Android device Galaxy-Nexus and Galaxy S III(Android
4.04). We can connect to a GO as a GC.
      * This demo runs a HTTP server, so you can connect to our device via a
web browser in your Android device.
      */
    #define MY_DEFAULT_WIFI_SECURITY_MODE
      WF_SECURITY_WPS_PUSH_BUTTON
    #define MY_DEFAULT_SCAN_TYPE                              WF_ACTIVE_SCAN
    #define MY_DEFAULT_SSID_NAME                              "DIRECT-"    /*
Fixed SSID. Do not change */
    #define MY_DEFAULT_CHANNEL_LIST                    {1, 6, 11}    /* Social
channels. Do not change */
    #define MY_DEFAULT_BEACON_TIMEOUT                          (40)   /* Number
of beacon periods */
    #define MY_DEFAULT_PS_POLL                                WF_DISABLED
#else /* !defined (MRF24WG) */
    #error "MRF24WB does not support Wi-Fi Direct (P2P)"
#endif      /* defined (MRF24WG) */
#endif      /* MY_DEFAULT_NETWORK_TYPE == CFG_WF_INFRASTRUCTURE */

#if defined(__C32__)
/* This option allows host to convert the passphrase to the key by itself
instead of relying on RF module FW.
* Even if you do not use this option, RF module FW will still take care of
this key deriviation.
* However it will take much more time such as 32 seconds for MRF24WB or 25
seconds for MRF24WG.
* Also note that the reason PIC18/24 are not allowed to use this option is
just to save memory space on it.
* So if you have enough memory on PIC18/24, then you can also use this option
with adding WF_pbkdf2.c
* in your projects.
*/
#define DERIVE_KEY_FROM_PASSPHRASE_IN_HOST
#endif

#if defined (MRF24WG)
/* The module HW has 2 hardware multicast filters. If that is not enough on
your application,
* then you can choose this option to extend it to max 16.  As the macro name
indicates this forces
* the module FW to use software to run the filters instead of hardware.
Downside of this option
* is the performance can degrade when there are so many multicast packets on
air becasue the
* filtering is done by SW
*/
//#define ENABLE_SOFTWARE_MULTICAST_FILTER
#endif
```

```
// For WPS Push-Button demo, press the button of AP (Registrar) first before
running this demo.
// Input this pin number on AP (Registrar), and activate Registrar first
before connection attempt
// Also note that this 8 digit is not randomly generated. Last digit is the
checksum of first 7 digits.
// The checksum must be correct, otherwise MRF24WG module wil reject the pin
code
#define MY_DEFAULT_WPS_PIN                          "12390212"    /* only used
when security is WF_SECURITY_WPS_PIN */

#define MY_DEFAULT_WIFI_SECURITY_WEP_KEYTYPE  WF_SECURITY_WEP_OPENKEY /*
WF_SECURITY_WEP_OPENKEY (default) or        */

                                /*  WF_SECURITY_WEP_SHAREDKEY.
*/

//------------------------------------------------------------------------
--------
// Default WEP keys used in WF_SECURITY_WEP_40  and WF_SECURITY_WEP_104
security mode
//------------------------------------------------------------------------
--------
#define MY_DEFAULT_WEP_PHRASE           "WEP Phrase"

// string 4 40-bit WEP keys -- corresponding to passphraseof "WEP Phrase"
#define MY_DEFAULT_WEP_KEYS_40 "\
\x5a\xfb\x6c\x8e\x77\
\xc1\x04\x49\xfd\x4e\
\x43\x18\x2b\x33\x88\
\xb0\x73\x69\xf4\x78"
// Do not indent above string as it will inject spaces

// string containing 4 104-bit WEP keys -- corresponding to passphraseof "WEP
Phrase"
#define MY_DEFAULT_WEP_KEYS_104 "\
\x90\xe9\x67\x80\xc7\x39\x40\x9d\xa5\x00\x34\xfc\xaa\
\x77\x4a\x69\x45\xa4\x3d\x66\x63\xfe\x5b\x1d\xb9\xfd\
\x82\x29\x87\x4c\x9b\xdc\x6d\xdf\x87\xd1\xcf\x17\x41\
\xcc\xd7\x62\xde\x92\xad\xba\x3b\x62\x2f\x7f\xbe\xfb"
// Do not indent above string as it will inject spaces

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


// Default pass phrase used for WF_SECURITY_WPA_WITH_PASS_PHRASE and
// WF_SECURITY_WPA2_WITH_PASS_PHRASE security modes
#define MY_DEFAULT_PSK_PHRASE               "Microchip 802.11 Secret PSK
Password"

// If using security mode of WF_SECURITY_WPA_WITH_KEY or
WF_SECURITY_WPA2_WITH_KEY, then this section
// must be set to  match the key for MY_DEFAULT_SSID_NAME and
MY_DEFAULT_PSK_PHRASE
// combination.  The values below are derived from the SSID "MicrochipDemoAP"
```

```
and the pass phrase
// "Microchip 802.11 Secret PSK Password".
// The tool at http://www.wireshark.org/tools/wpa-psk.html can be used to
generate this field.
#define MY_DEFAULT_PSK "\
\x86\xC5\x1D\x71\xD9\x1A\xAA\x49\
\x40\xC8\x88\xC6\xE9\x7A\x4A\xD5\
\xE5\x6D\xDA\x44\x8E\xFB\x9C\x0A\
\xE1\x47\x81\x52\x31\x1C\x13\x7C"
// Do not indent above string as it will inject spaces

/*** Selecting Event Notification Type ***/
#define MY_DEFAULT_EVENT_NOTIFICATION_LIST
(WF_NOTIFY_CONNECTION_ATTEMPT_SUCCESSFUL  |         \

WF_NOTIFY_CONNECTION_ATTEMPT_FAILED      |         \

WF_NOTIFY_CONNECTION_TEMPORARILY_LOST    |         \

WF_NOTIFY_CONNECTION_PERMANENTLY_LOST    |         \

WF_NOTIFY_CONNECTION_REESTABLISHED)


#endif /* __WF_CONFIG_H_ */
```

7.4.3 Edited Portions of the TCP/IP Stack

WFAPI.h

```
/*****************************************************************************
**

 MRF24W Driver API Interface
 Module for Microchip TCP/IP Stack
  -Provides access to MRF24W WiFi controller (MRF24WB0MA/B, MRF24WG0MA/B)
  -Reference: MRF24W Data sheet, IEEE 802.11 Standard

*****************************************************************************
**
 FileName:      WFApi.h
 Dependencies:  TCP/IP Stack header files
 Processor:     PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32
 Compiler:      Microchip C32 v1.10b or higher
                Microchip C30 v3.22 or higher
                Microchip C18 v3.34 or higher
 Company:       Microchip Technology, Inc.

 Software License Agreement

 Copyright (C) 2002-2010 Microchip Technology Inc.  All rights reserved.

 Microchip licenses to you the right to use, modify, copy, and distribute:
 (i)  the Software when embedded on a Microchip microcontroller or digital
      signal controller product ("Device") which is integrated into
      Licensee's product; or
 (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
```

```
      ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device used in
      conjunction with a Microchip ethernet controller for the sole purpose
      of interfacing with the ethernet controller.

 You should refer to the license agreement accompanying this Software for
 additional information regarding your rights and obligations.

 THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
 KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY
 OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND
 NON-INFRINGEMENT. IN NO EVENT SHALL MICROCHIP BE LIABLE FOR ANY INCIDENTAL,
 SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST
 OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS BY
 THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), ANY CLAIMS
 FOR INDEMNITY OR CONTRIBUTION, OR OTHER SIMILAR COSTS, WHETHER ASSERTED ON
 THE BASIS OF CONTRACT, TORT (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR
 OTHERWISE.


 Author             Date        Comment
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~
 KH                 27 Jan 2010 Created for MRF24W
********************************************************************************
*/

#ifndef __WF_API_H_
#define __WF_API_H_

#include "GenericTypeDefs.h"

/*= WF_ASSERT MACRO
============================================================*/
/* Customize how the WiFi driver assert macro (WF_ASSERT) should operate.
*/
/*  To DISABLE the WF_ASSERT macro: Comment out '#define WF_DEBUG'
*/
/*  To ENABLE the WF_ASSERT macro:  Unoomment out '#define WF_DEBUG'
*/
/*============================================================================
=========*/
//#define WF_DEBUG
#if !defined(__18CXX)
    #define DISPLAY_FILENAME  /* will diplay file names instead of module
numbers on an assert */
#endif

/*----------------------------------------------------------------------------
-*/
/* This block of defines allows for code and data reduction by removing
*/
/* WiFi driver code and or data that is not needed by the application.
*/
/* Comment out those function blocks that are not needed.
*/
/*----------------------------------------------------------------------------
-*/
```

```
#define WF_USE_SCAN_FUNCTIONS
#define WF_USE_TX_POWER_CONTROL_FUNCTIONS
#define WF_USE_POWER_SAVE_FUNCTIONS
#define WF_USE_MULTICAST_FUNCTIONS
#define WF_USE_INDIVIDUAL_SET_GETS
#define WF_USE_GROUP_SET_GETS
//#define USE_GRATUITOUS_ARP


/*
******************************************************************************
**************************
*                                          DEFINES
******************************************************************************
**************************
*/
/*-----------------------------*/
/* WiFi Driver Version Number  */
/*-----------------------------*/
#define WF_HOST_DRIVER_VERSION_NUMBER    "2.7.0"

/* API defines */
#define WF_MAX_NUM_CHANNELS             (14)
#define WF_MAX_NUM_CONNECTION_PROFILES  (8)
#define WF_CP_LIST_LENGTH               (8)
#define WF_MAX_SSID_LENGTH              (32)
#define WF_BSSID_LENGTH                 (6)
#define WF_RETRY_FOREVER                (255)
#define WF_CHANNEL_LIST_LENGTH          (14)
#define WF_MAX_SECURITY_KEY_LENGTH      (64)

#define WF_MIN_NUM_CPID                 (1)
#define WF_MAX_NUM_CPID                 (8)

#define WF_NO_CPID_ACTIVE               (0)

#define WF_RTS_THRESHOLD_MAX            (2347) /* maximum RTS threshold size
in bytes */

#define WF_MAX_NUM_RATES                (8)

/* Key size defines */
#define WF_MIN_WPA_PASS_PHRASE_LENGTH         (8)
#define WF_MIN_WPA_PASS_PHRASE_LENGTH         (8)
#define WF_MAX_WPA_PASS_PHRASE_LENGTH        (64)
#define WF_MAX_WPA2_PASS_PHRASE_LENGTH       (64)
#define WF_WPA_KEY_LENGTH                    (32)
#define WF_WPA2_KEY_LENGTH                   (32)

/*-------------------------------------------------------------------------
---*/
/* These are error codes returned in the result field of a management
response. */
/*-------------------------------------------------------------------------
---*/
#define WF_SUCCESS                                          ((UINT16)1)
#define WF_ERROR_INVALID_SUBTYPE                            ((UINT16)2)
```

```
#define WF_ERROR_OPERATION_CANCELLED                                ((UINT16)3)
#define WF_ERROR_FRAME_END_OF_LINE_OCCURRED                         ((UINT16)4)
#define WF_ERROR_FRAME_RETRY_LIMIT_EXCEEDED                         ((UINT16)5)
#define WF_ERROR_EXPECTED_BSS_VALUE_NOT_IN_FRAME                    ((UINT16)6)
#define WF_ERROR_FRAME_SIZE_EXCEEDS_BUFFER_SIZE                     ((UINT16)7)
#define WF_ERROR_FRAME_ENCRYPT_FAILED                               ((UINT16)8)
#define WF_ERROR_INVALID_PARAM                                      ((UINT16)9)
#define WF_ERROR_AUTH_REQ_ISSUED_WHILE_IN_AUTH_STATE                ((UINT16)10)
#define WF_ERROR_ASSOC_REQ_ISSUED_WHILE_IN_ASSOC_STATE              ((UINT16)11)
#define WF_ERROR_INSUFFICIENT_RESOURCES                             ((UINT16)12)
#define WF_ERROR_TIMEOUT_OCCURRED                                   ((UINT16)13)
#define WF_ERROR_BAD_EXCHANGE_ENCOUNTERED_IN_FRAME_RECEPTION        ((UINT16)14)
#define WF_ERROR_AUTH_REQUEST_REFUSED                               ((UINT16)15)
#define WF_ERROR_ASSOCIATION_REQUEST_REFUSED                        ((UINT16)16)
#define WF_ERROR_PRIOR_MGMT_REQUEST_IN_PROGRESS                     ((UINT16)17)
#define WF_ERROR_NOT_IN_JOINED_STATE                                ((UINT16)18)
#define WF_ERROR_NOT_IN_ASSOCIATED_STATE                            ((UINT16)19)
#define WF_ERROR_NOT_IN_AUTHENTICATED_STATE                         ((UINT16)20)
#define WF_ERROR_SUPPLICANT_FAILED                                  ((UINT16)21)
#define WF_ERROR_UNSUPPORTED_FEATURE                                ((UINT16)22)
#define WF_ERROR_REQUEST_OUT_OF_SYNC                                ((UINT16)23)
#define WF_ERROR_CP_INVALID_ELEMENT_TYPE                            ((UINT16)24)
#define WF_ERROR_CP_INVALID_PROFILE_ID                              ((UINT16)25)
#define WF_ERROR_CP_INVALID_DATA_LENGTH                             ((UINT16)26)
#define WF_ERROR_CP_INVALID_SSID_LENGTH                             ((UINT16)27)
#define WF_ERROR_CP_INVALID_SECURITY_TYPE                           ((UINT16)28)
#define WF_ERROR_CP_INVALID_SECURITY_KEY_LENGTH                     ((UINT16)29)
#define WF_ERROR_CP_INVALID_WEP_KEY_ID                              ((UINT16)30)
#define WF_ERROR_CP_INVALID_NETWORK_TYPE                            ((UINT16)31)
#define WF_ERROR_CP_INVALID_ADHOC_MODE                              ((UINT16)32)
#define WF_ERROR_CP_INVALID_SCAN_TYPE                               ((UINT16)33)
#define WF_ERROR_CP_INVALID_CP_LIST                                 ((UINT16)34)
#define WF_ERROR_CP_INVALID_CHANNEL_LIST_LENGTH                     ((UINT16)35)
#define WF_ERROR_NOT_CONNECTED                                      ((UINT16)36)
#define WF_ERROR_ALREADY_CONNECTING                                 ((UINT16)37)
#define WF_ERROR_DISCONNECT_FAILED                                  ((UINT16)38)
#define WF_ERROR_NO_STORED_BSS_DESCRIPTOR                           ((UINT16)39)
#define WF_ERROR_INVALID_MAX_POWER                                  ((UINT16)40)
#define WF_ERROR_CONNECTION_TERMINATED                              ((UINT16)41)
#define WF_ERROR_HOST_SCAN_NOT_ALLOWED                              ((UINT16)42)
#define WF_ERROR_INVALID_WPS_PIN                                    ((UINT16)44)


/*------------------------------------------------------------------------*/
/* Used for eventNotificationField bit mask in tWFCAElements structure */
/*------------------------------------------------------------------------*/
#define WF_NOTIFY_CONNECTION_ATTEMPT_SUCCESSFUL     ((UINT8)(0x01))
#define WF_NOTIFY_CONNECTION_ATTEMPT_FAILED         ((UINT8)(0x02))
#define WF_NOTIFY_CONNECTION_TEMPORARILY_LOST       ((UINT8)(0x04))
#define WF_NOTIFY_CONNECTION_PERMANENTLY_LOST       ((UINT8)(0x08))
#define WF_NOTIFY_CONNECTION_REESTABLISHED          ((UINT8)(0x10))
#define WF_NOTIFY_ALL_EVENTS                        ((UINT8)(0x1f))

/*------------------------------------------------------------------------*/
/* Used for Tx mode selection */
/*------------------------------------------------------------------------*/
#define WF_TXMODE_G_RATES       0
```

```
#define WF_TXMODE_B_RATES        1
#define WF_TXMODE_LEGACY_RATES   2

/*------------------------------------------------------------------------
---------------*/
/* Multicast Filter ID's
*/
/* Infrastructure can use 2,3,4,5 and AdHoc can only use 4,5.  Use 4,5 which
works for both */
/*------------------------------------------------------------------------
---------------*/
#define WF_MULTICAST_FILTER_1       (4)
#define WF_MULTICAST_FILTER_2       (5)
#if defined(MRF24WG)
    #define WF_MULTICAST_FILTER_3       (6)
    #define WF_MULTICAST_FILTER_4       (7)
    #define WF_MULTICAST_FILTER_5       (8)
    #define WF_MULTICAST_FILTER_6       (9)
    #define WF_MULTICAST_FILTER_7       (10)
    #define WF_MULTICAST_FILTER_8       (11)
    #define WF_MULTICAST_FILTER_9       (12)
    #define WF_MULTICAST_FILTER_10      (13)
    #define WF_MULTICAST_FILTER_11      (14)
    #define WF_MULTICAST_FILTER_12      (15)
    #define WF_MULTICAST_FILTER_13      (16)
    #define WF_MULTICAST_FILTER_14      (17)
    #define WF_MULTICAST_FILTER_15      (18)
    #define WF_MULTICAST_FILTER_16      (19)

    #define WF_MULTICAST_DISABLE_ALL    (0)
    #define WF_MULTICAST_ENABLE_ALL     (1)
    #define WF_MULTICAST_USE_FILTERS    (2)
#endif /* MRF24WG */

#define WF_MASK_DEAUTH_REASONCODE       ((UINT8)0x80)
#define WF_MASK_DISASSOC_REASONCODE     ((UINT8)0x40)

#define WF_SCAN_ALL ((UINT8)(0xff))
#define WF_HWRSSIVAL_MAX    200    /* hw RSSI reference value to be used to
derive real RSSI */
/*
******************************************************************************
**************************
*                                            DATA TYPES
******************************************************************************
**************************
*/

/*------------------------------------------------------------------------
-*/
/* Events that can be invoked in WF_ProcessEvent().  Note that the
*/
/* connection events are optional, all other events the app must be notified.
*/
/*------------------------------------------------------------------------
-*/
```

```
#define WF_EVENT_CONNECTION_SUCCESSFUL           (1)   /* Connection attempt
to network successful            */
#define WF_EVENT_CONNECTION_FAILED               (2)   /* Connection attempt
failed                           */


#define WF_EVENT_CONNECTION_TEMPORARILY_LOST     (3)   /* Connection lost;
MRF24W attempting to reconnect     */
#define WF_EVENT_CONNECTION_PERMANENTLY_LOST     (4)   /* Connection lost;
MRF24W no longer trying to connect */
#define WF_EVENT_CONNECTION_REESTABLISHED        (5)

#define WF_EVENT_FLASH_UPDATE_SUCCESSFUL         (6)   /* Update to FLASH
successful                        */
#define WF_EVENT_FLASH_UPDATE_FAILED             (7)   /* Update to FLASH
failed                            */



#define WF_EVENT_KEY_CALCULATION_REQUEST        (8)   /* Key calculation is
required                       */

#define WF_EVENT_SCAN_RESULTS_READY              (9)   /* scan results are
ready                             */
#define WF_EVENT_IE_RESULTS_READY                (10)  /* IE data ready
*/


#define WF_EVENT_RX_PACKET_RECEIVED              (11)   /* Rx data packet has
been received by MRF24W          */
#define WF_EVENT_INVALID_WPS_PIN                 (12)   /* Invalid WPS pin
was entered                           */


typedef struct WFMacStatsStruct
{
    /**
      Number of frames received with the Protected Frame subfield of the
Frame
      Control field set to zero and the value of dot11ExcludeUnencrypted
causes
      that frame to be discarded.
      */
    UINT32 MibWEPExcludeCtr;
    UINT32 MibTxBytesCtr; // Total number of Tx bytes that have been
transmitted

    /**
      Number of frames successfully transmitted that had the multicast bit
set
      in the destination MAC address.
      */
    UINT32 MibTxMulticastCtr;
    /**
      Number of Tx frames that failed due to the number of transmits
exceeding
      the retry count.
```

```
      */
    UINT32 MibTxFailedCtr;
    UINT32 MibTxRtryCtr;        // Number of times a transmitted frame needed
to be retried
    UINT32 MibTxMultRtryCtr;    // Number of times a frame was successfully
transmitted after more than one retransmission.
    UINT32 MibTxSuccessCtr;     // Number of Tx frames successfully
transmitted.
    UINT32 MibRxDupCtr;         // Number of frames received where the
Sequence Control field indicates a duplicate.
    UINT32 MibRxCtsSuccCtr;     // Number of CTS frames received in response
to an RTS frame.
    UINT32 MibRxCtsFailCtr;     // Number of times an RTS frame was not
received in response to a CTS frame.
    UINT32 MibRxAckFailCtr;     // Number of times an Ack was not received in
response to a Tx frame.
    UINT32 MibRxBytesCtr;       // Total number of Rx bytes received.
    UINT32 MibRxFragCtr;        // Number of successful received frames
(management or data)
    UINT32 MibRxMultCtr;        // Number of frames received with the
multicast bit set in the destination MAC address.
    UINT32 MibRxFCSErrCtr;      // Number of frames received with an invalid
Frame Checksum (FCS).

    /**
      Number of frames received where the Protected Frame subfield of the
Frame Control Field is set to
      one and the WEPOn value for the key mapped to the transmitter's MAC
address indicates the frame
      should not have been encrypted.
      */
    UINT32 MibRxWEPUndecryptCtr;
    UINT32 MibRxFragAgedCtr; // Number of times that fragments "aged out",
or were not received in the allowable time.
    UINT32 MibRxMICFailureCtr; // Number of MIC failures that have occurred.
} tWFMacStats;


/*-------------------------------------------------------*/
/* Security Type defines                                 */
/* Used in WF_CPSet/GetSecurityType WF_CPSet/GetElements */
/*-------------------------------------------------------*/
#define WF_SECURITY_OPEN                        (0)
#define WF_SECURITY_WEP_40                      (1)
#define WF_SECURITY_WEP_104                     (2)
#define WF_SECURITY_WPA_WITH_KEY                (3)
#define WF_SECURITY_WPA_WITH_PASS_PHRASE        (4)
#define WF_SECURITY_WPA2_WITH_KEY               (5)
#define WF_SECURITY_WPA2_WITH_PASS_PHRASE       (6)
#define WF_SECURITY_WPA_AUTO_WITH_KEY           (7)
#define WF_SECURITY_WPA_AUTO_WITH_PASS_PHRASE   (8)
#define WF_SECURITY_WPS_PUSH_BUTTON             (9)
#define WF_SECURITY_WPS_PIN                     (10)
#define WF_SECURITY_EAP                         (11)    /* currently not
supported */
```

```
/* Wep key types */
#define WF_SECURITY_WEP_SHAREDKEY  (0)
#define WF_SECURITY_WEP_OPENKEY    (1)


/*-------------------------------------------------------------------------*/
/* Network Type defines                                                    */
/* Used in WF_CPSet/GetNetworkType, WF_CPSetElements, WF_CPGetElements */
/*-------------------------------------------------------------------------*/
#define WF_INFRASTRUCTURE 1
#define WF_ADHOC          2
#define WF_P2P            3
#define WF_SOFT_AP        4

/*---------------------------------------------------------*/
/* Ad Hoc behavior defines                                 */
/* Used in WF_CPSet/GetAdhocBehavor, WF_CPSet/GetElements */
/*---------------------------------------------------------*/
#define WF_ADHOC_CONNECT_THEN_START  (0)
#define WF_ADHOC_CONNECT_ONLY        (1)
#define WF_ADHOC_START_ONLY          (2)

/*-----------------------------------------------------*/
/* Scan type defines                                   */
/* Used in WF_CASet/GetScanType, WF_CASet/GetElements */
/*-----------------------------------------------------*/
#define WF_ACTIVE_SCAN   (1)
#define WF_PASSIVE_SCAN  (2)


/*-------------------------------------------------------------------------------
--------------*/
/* Beacon Timeout and Deauth defines
*/
/* Used in WF_CASet/GetBeaconTimeoutAction, WF_CASet/GetDeauthAction,
WF_CASet/GetElements */
/*-------------------------------------------------------------------------------
--------------*/
#define WF_DO_NOT_ATTEMPT_TO_RECONNECT  (0)
#define WF_ATTEMPT_TO_RECONNECT         (1)


#define WF_DISABLED  (0)
#define WF_ENABLED   (1)

/* eventInfo defines for WF_ProcessEvent(), case WF_EVENT_CONNECTION_FAILED
*/
/* Also value for index 3 of WF_CONNECTION_FAILED_EVENT_SUBTYPE */
#define WF_JOIN_FAILURE                         (2)
#define WF_AUTHENTICATION_FAILURE               (3)
#define WF_ASSOCIATION_FAILURE                  (4)
#define WF_WEP_HANDSHAKE_FAILURE                (5)
#define WF_PSK_CALCULATION_FAILURE              (6)
#define WF_PSK_HANDSHAKE_FAILURE                (7)
#define WF_ADHOC_JOIN_FAILURE                   (8)
#define WF_SECURITY_MISMATCH_FAILURE            (9)
```

```
#define WF_NO_SUITABLE_AP_FOUND_FAILURE         (10)
#define WF_RETRY_FOREVER_NOT_SUPPORTED_FAILURE  (11)
#define WF_LINK_LOST                            (12)
#define WF_TKIP_MIC_FAILURE                     (13)
#define WF_RSN_MIXED_MODE_NOT_SUPPORTED         (14)
#define WF_RECV_DEAUTH                          (15)
#define WF_RECV_DISASSOC                        (16)
#define WF_WPS_FAILURE                          (17)
#define WF_P2P_FAILURE                          (18)

/* Reason Codes */
#define WF_UNSPECIFIED                          (1)
#define WF_REASON_PREV_AUTH_NOT_VALID           (2)
#define WF_DEAUTH_LEAVING                       (3)
#define WF_DISASSOC_DUE_TO_INACTIVITY           (4)
#define WF_DISASSOC_AP_BUSY                     (5)
#define WF_CLASS2_FRAME_FROM_NONAUTH_STA        (6)
#define WF_CLASS3_FRAME_FROM_NONASSOC_STA       (7)
#define WF_DISASSOC_STA_HAS_LEFT                (8)
#define WF_STA_REQ_ASSOC_WITHOUT_AUTH           (9)
#define WF_INVALID_IE                           (13)
#define WF_MIC_FAILURE                          (14)
#define WF_4WAY_HANDSHAKE_TIMEOUT               (15)
#define WF_GROUP_KEY_HANDSHAKE_TIMEOUT          (16)
#define WF_IE_DIFFERENT                         (17)
#define WF_INVALID_GROUP_CIPHER                 (18)
#define WF_INVALID_PAIRWISE_CIPHER              (19)
#define WF_INVALID_AKMP                         (20)
#define WF_UNSUPP_RSN_VERSION                   (21)
#define WF_INVALID_RSN_IE_CAP                   (22)
#define WF_IEEE8021X_FAILED                     (23)
#define WF_CIPHER_SUITE_REJECTED                (24)


/* eventInfo defines for WF_ProcessEvent(), case
WF_EVENT_CONNECTION_TEMPORARILY_LOST */
#define WF_BEACON_TIMEOUT        (1)
#define WF_DEAUTH_RECEIVED       (2)
#define WF_DISASSOCIATE_RECEIVED (3)

/* Status Codes */
#define WF_UNSPECIFIED_FAILURE          (1)
#define WF_CAPS_UNSUPPORTED             (10)
#define WF_REASSOC_NO_ASSOC             (11)
#define WF_ASSOC_DENIED_UNSPEC          (12)
#define WF_NOT_SUPPORTED_AUTH_ALG       (13)
#define WF_UNKNOWN_AUTH_TRANSACTION     (14)
#define WF_CHALLENGE_FAIL               (15)
#define WF_AUTH_TIMEOUT                 (16)
#define WF_AP_UNABLE_TO_HANDLE_NEW_STA  (17)
#define WF_ASSOC_DENIED_RATES           (18)
#define WF_ASSOC_DENIED_NOSHORTPREAMBLE (19)
#define WF_ASSOC_DENIED_NOPBCC          (20)
#define WF_ASSOC_DENIED_NOAGILITY       (21)
#define WF_ASSOC_DENIED_NOSHORTTIME     (25)
#define WF_ASSOC_DENIED_NODSSSOFDM      (26)
#define WF_S_INVALID_IE                 (40)
```

```
#define WF_S_INVALID_GROUPCIPHER        (41)
#define WF_S_INVALID_PAIRWISE_CIPHER    (42)
#define WF_S_INVALID_AKMP               (43)
#define WF_UNSUPPORTED_RSN_VERSION      (44)
#define WF_S_INVALID_RSN_IE_CAP         (45)
#define WF_S_CIPHER_SUITE_REJECTED      (46)
#define WF_TIMEOUT                      (47)



#define MRF24WB0M_DEVICE  (1)
#define MRF24WG0M_DEVICE  (2)

#if defined(MRF24WG)
    /* Domain Codes */
    #define WF_DOMAIN_FCC      (0)          /* Available Channels: 1 - 11 */
    #define WF_DOMAIN_ETSI     (2)          /* Available Channels: 1 - 13 */
    #define WF_DOMAIN_JAPAN    (7)          /* Available Channels: 1 - 14 */
    #define WF_DOMAIN_OTHER    (7)          /* Available Channels: 1 - 14 */
#else
    /* Domain Codes */
    #define WF_DOMAIN_FCC      (0)          /* Available Channels: 1  - 11
*/
    #define WF_DOMAIN_IC       (1)          /* Available Channels: 1  - 11
*/
    #define WF_DOMAIN_ETSI     (2)          /* Available Channels: 1  - 13
*/
    #define WF_DOMAIN_SPAIN    (3)          /* Available Channels: 10 - 11
*/
    #define WF_DOMAIN_FRANCE   (4)          /* Available Channels: 10 - 13
*/
    #define WF_DOMAIN_JAPAN_A  (5)          /* Available Channels: 14
*/
    #define WF_DOMAIN_JAPAN_B  (6)          /* Available Channels: 1  - 13
*/
#endif

/* Power save states */
#define WF_PS_HIBERNATE              (1)
#define WF_PS_PS_POLL_DTIM_ENABLED   (2)
#define WF_PS_PS_POLL_DTIM_DISABLED  (3)
#define WF_PS_OFF                    (4)

/* Hibernate states */
#define WF_HB_NO_SLEEP          (0)
#define WF_HB_ENTER_SLEEP       (1)
#define WF_HB_WAIT_WAKEUP       (2)

/* Pin Level */
#define WF_LOW   (0)
#define WF_HIGH  (1)

/*-----------------------------------------------------------------------*/
/* defines used for the p_currentCpID value in WF_CMGetConnectionState() */
/*-----------------------------------------------------------------------*/
#define WF_CURRENT_CPID_NONE    (0)
#define WF_CURRENT_CPID_LIST    (0xff)
```

```
/* Connection States */
#define WF_CSTATE_NOT_CONNECTED                 (1)
#define WF_CSTATE_CONNECTION_IN_PROGRESS        (2)
#define WF_CSTATE_CONNECTED_INFRASTRUCTURE      (3)
#define WF_CSTATE_CONNECTED_ADHOC               (4)
#define WF_CSTATE_RECONNECTION_IN_PROGRESS      (5)
#define WF_CSTATE_CONNECTION_PERMANENTLY_LOST   (6)

/* eventInfo define for WF_ProcessEvent() when no additional info is supplied
*/
#define WF_NO_ADDITIONAL_INFO       ((UINT16)0xffff)

#define ENABLE_WPS_PRINTS    1 //(1 << 0)
#define ENABLE_P2P_PRINTS    (1 << 1)

/*------------------------------*/
/* Connection Profile Elements */
/*------------------------------*/

// Connection profile elements structure
typedef struct WFCPElementsStruct
{
    /**
      SSID, which must be less than or equal to 32 characters.  Set to all
0s
      if not being used.  If ssidLength is 0 this field is ignored.  If SSID
is
      not defined then the MRF24W, when using this profile to connect, will
      scan all channels within its regional domain.

      Default: SSID not used.
      */
    UINT8  ssid[WF_MAX_SSID_LENGTH];
    /**
      Basic Service Set Identifier, always 6 bytes.  This is the 48-bit MAC
of
      the SSID.  It is an optional field that can be used to specify a
specific
      SSID if more than one AP exists with the same SSID.  This field can
also
      be used in lieu of the SSID.

      Set each byte to 0xFF if BSSID is not going to be used.
      Default: BSSID not used (all FFs)
      */
    UINT8  bssid[WF_BSSID_LENGTH];
    /**
      Number of ASCII bytes in ssid.  Set to 0 is SSID is not going to be
used.

      Default: 0
      */
    UINT8  ssidLength;
    /**
      Designates the desired security level for the connection.  Choices are:
```

```
      <table>
        WF_SECURITY_OPEN                          No security encryption used.
        WF_SECURITY_WEP_40                        Use WEP security.
                                                  WEP key, using four 5-byte
keys will be provided in securityKey.
                                                   Note that only open
authentication is supported for WEP.
        WF_SECURITY_WEP_104                       Use WEP security.
                                                   WEP key, using four 13-byte
keys will be provided in securityKey.
                                                   Note that only open
authentication is supported for WEP.
        WF_SECURITY_WPA_WITH_KEY                  Use WPA security.
                                                   Binary PSK (Pre-shared Key)
key will be provided in securityKey.
        WF_SECURITY_WPA_WITH_PASS_PHRASE          Use WPA security.
                                                   ASCII WPA passphrase will be
provided in securityKey and,
                                                   after a call to
WF_CMConnect(), the MRF24W will calculate
                                                   the PSK key (which can take
up to 30 seconds).
        WF_SECURITY_WPA2_WITH_KEY                 Use WPA-2 security.
                                                   Binary WPA-2 key will be
provided in securityKey.
        WF_SECURITY_WPA2_WITH_PASSPHRASE          Use WPA-2 security.
                                                   ASCII WPA-2 passphrase will
be provided in securityKey and,
                                                   after a call to
WF_CMConnect(), the MRF24W will calculate
                                                   the PSK key (which can take
up to 30 seconds).
        WF_SECURITY_WPA_AUTO_WITH_KEY             Same as
WF_SECURITY_WPA_WITH_KEY or WF_SECURITY_WPA2_WITH_KEY
                                                   except connection manager
will connect to the AP using highest
                                                   level security the AP
supports (WPA or WPA2).
        WF_SECURITY_WPA_AUTO_WITH_PASSPHRASE      Same as
WF_SECURITY_WPA_WITH_PASS_PHRASE or

WF_SECURITY_WPA2_WITH_PASS_PHRASE except connection manager
                                                   will connect to the AP using
highest level security the AP
                                                   supports (WPA or WPA2).
      </table>
      Default: WF_SECURITY_OPEN
      */
    UINT8  securityType;
    /**
      Set to NULL if securityType is WF_SECURITY_OPEN.  If securityKeyLength
is 0
      this field is ignored.
      <table>
        WEP Keys        If using WEP this field must contain 4 keys.  Each
key must be
                         either 5 bytes in length (if securityType is
```

```
WF_SECURITY_WEP_40)
                        or 13 bytes in length  (if securityType is
WF_SECURITY_WEP_104).
                        The keys must be contiguous within this field.  For
example, if
                        using 5 bytes keys the first key starts at index 0,
the second
                        key at index 5, the third key at index 10, and the
last key at
                        index 15.  Unused keys should be set to all 0s.
        WPA/WPA2 Keys   If using WPA or WPA2 you can provide the actual
binary key or
                        ASCII passphrase used to generate a key.  [64 byte
array covers
                        all cases of keys or passphrases].  If using this
field for a
                        security passphrase the MRF24W will need to
calculate the
                        binary key after the call to WF_CMConnect() this
can add about
                        30 seconds to the connection time.
      </table>
      Default: No security key defined
      */
    UINT8  securityKey[WF_MAX_SECURITY_KEY_LENGTH];
    /**
      Number of bytes used in the securityKey.  Set to 0 if securityType is
WF_SECURITY_OPEN.
      <table>
        WEP Keys        If securityType is WF_SECURITY_WEP_40 or
WF_SECURITY_WEP_104
                        then this field is the length of the four WEP keys.

                        Range is
                        20 if securityType is WF_SECURITY_WEP_40 (four 5-
byte keys),
                        52 if securityType is WF_SECURITY_WEP_104 (four 13-
byte keys)
        WPA/WPA2 Keys   If securityType is one of the WPA or WPA2 choices
then this
                        field is the number of bytes in the binary key or
the
                        passphrase, whichever is being used.

      </table>
      Default: 0
      */
    UINT8  securityKeyLength;
    /**
      This field is only used if securityType is WF_SECURITY_WEP.  This field
      designates which of the four WEP keys defined in securityKey to use
when
      connecting to a WiFi network.  The range is 0 thru 3, with the default
      being 0.
      */
    UINT8  wepDefaultKeyId;
```

```
    /**
      WF_INFRASTRUCTURE  or WF_ADHOC

      Default: WF_INFRASTRUCTURE
      */
    UINT8  networkType;
    /**
      Only applicable if networkType is WF_ADHOC.  Configures Adhoc behavior.
Choices are:
      <table>
        WF_ADHOC_CONNECT_THEN_START     Attempt to connect to existing
network.
                                         If that fails, then start a network.
        WF_ADHOC_CONNECT_ONLY           Connect only to an existing network.
                                         Do not start a network.
        WF_ADHOC_START_ONLY             Only start a network.
      </table>
      Default: WF_ADHOC_CONNECT_THEN_START
      */
    UINT8  adHocBehavior;
    /**
    1 - enable hidden ssid in adhoc mode
    */
    UINT8 hiddenSSID;
    /**
    0- shared key, 1 - open key
    */
    UINT8 wepKeyType;
} tWFCPElements;

/*------------------------------*/
/* Connection Algorithm Elements */
/*------------------------------*/
typedef struct WFCAElementsStruct
{
    /**
      This parameter is only used when PS Poll mode is enabled.  See
      WF_PsPollEnable().  Number of 100ms intervals between instances when
the
      MRF24W wakes up to received buffered messages from the network.  Range
      is from 1 (100ms) to 6553.5 sec (~109 min).

      Note that the 802.11 standard defines the listen interval in terms of
      Beacon Periods, which are typically 100ms.  If the MRF24W is
communicating
      to a network with a network that has Beacon Periods that is not 100ms
it
      will round up (or down) as needed to match the actual Beacon Period as
      closely as possible.

      Important Note: If the listenInterval is modified while connected to a
      network the MRF24W will automatically reconnect to the network with the
      new Beacon Period value.  This may cause a temporary loss of data
packets.
      */
    UINT16  listenInterval;
    /**
```

```
      WF_ACTIVE_SCAN (Probe Requests sent out) or WF_PASSIVE_SCAN (listen
only)

      Default: WF_ACTIVE_SCAN
      */
    UINT8   scanType;
    /**
      Specifies RSSI restrictions when connecting.  This field is only used
if:
      1.  The Connection Profile has not defined a SSID or BSSID, or
      2.  An SSID is defined in the Connection Profile and multiple AP's are
discovered with the same SSID.

      <table>
        0       Connect to the first network found
        1-254   Only connect to a network if the RSSI is greater than or
equal to the specified value.
        255     Connect to the highest RSSI found (default)
      </table>
      */
    UINT8   rssi;
    /**
      <b>Note: Connection Profile lists are not yet supported.  This array
should be set to all FF's.</b>
      */
    UINT8   connectionProfileList[WF_CP_LIST_LENGTH];
    /**
      This field is used to specify the number of retries for the single
      connection profile before taking the connection lost action.

      Range 1 to 254 or WF_RETRY_FOREVER (255)

      Default is 3
      */
    UINT8   listRetryCount;
    /**
      There are several connection-related events that can occur.  The Host
has
      the option to be notified (or not) when some of these events occur.
This
      field controls event notification for connection-related events.
      <table>
        Bit 7   Bit 6   Bit 5   Bit 4   Bit 3   Bit 2   Bit 1   Bit 0
        -----   -----   -----   -----   -----   -----   -----   -----
        Not     Not     Not     Event   Event   Event   Event   Event
         used    used    used    E       D       C       B       A
      </table>
      The defines for each bit are shown below.
      <table>
        Event Code  #define
        ----------  -------
        A           WF_NOTIFY_CONNECTION_ATTEMPT_SUCCESSFUL
        B           WF_NOTIFY_CONNECTION_ATTEMPT_FAILED
        C           WF_NOTIFY_CONNECTION_TEMPORARILY_LOST
        D           WF_NOTIFY_CONNECTION_PERMANENTLY_LOST
        E           WF_NOTIFY_CONNECTION_REESTABLISHED
```

```
      </table>
      If a bit is set, then the host will be notified if the associated event
      occurs.  If the bit is not set then the host will not be notified if
the
      associated event occurs.   A #define, WF_NOTIFY_ALL_EVENTS, exists as a
      shortcut to allow notification for all events.

      Note that if an event is not in the above bit mask the application will
      always be notified of the event via the call to WF_ProcessEvent().

      Default: WF_NOTIFY_ALL_EVENTS
      */
    UINT8   eventNotificationAction;
    /**
      Specifies the action the Connection Manager should take if a Connection
is
      lost due to a Beacon Timeout.
      If this field is set to WF_ATTEMPT_TO_RECONNECT then the number of
attempts
      is limited to the value in listRetryCount.

      Choices are:
      WF_ATTEMPT_TO_RECONNECT or WF_DO_NOT_ATTEMPT_TO_RECONNECT

      Default: WF_ATTEMPT_TO_RECONNECT
      */
    UINT8   beaconTimeoutAction;
    /**
      Designates what action the Connection Manager should take if it
receives a
      Deauthentication message from the AP.

      If this field is set to WF_ATTEMPT_TO_RECONNECT then the number of
attempts
      is limited to the value in listRetryCount.

      Choices are:
      WF_ATTEMPT_TO_RECONNECT or WF_DO_NOT_ATTEMPT_TO_RECONNECT

      Default: WF_ATTEMPT_TO_RECONNECT
      */
    UINT8   deauthAction;
    /**
      List of one or more channels that the MRF24W should utilize when
      connecting or scanning.  If numChannelsInList is set to 0 then this
      parameter should be set to NULL.

      Default: All valid channels for the regional domain of the MRF24W (set
      at manufacturing).
    */
    UINT8   channelList[WF_CHANNEL_LIST_LENGTH];
    /**
      Number of channels in channelList.  If set to 0 then the MRF24W will
      populate the list with all valid channels for the regional domain.

      Default: The number of valid channels for the regional domain of the
      MRF24W (set at manufacturing).
```

```
      */
    UINT8   numChannelsInList;
    /**
      Specifies the number of beacons that can be missed before the action
      described in beaconTimeoutAction is taken.

      <table>
        0       * Not monitor the beacon timeout condition
                 * Will not indicate this condition to Host
        1-255   Beacons missed before disconnect event occurs and
beaconTimeoutAction
                 occurs.  If enabled, host will receive an event message
indicating
                 connection temporarily or permanently lost, and if retrying,
a
                 connection successful event.
      </table>
      Default: 0 (no monitoring or notification of beacon timeout)
      */
    UINT8   beaconTimeout;
    /**
      The number of times to scan a channel while attempting to find a
particular
      access point.

      Default: 1
      */
    UINT8   scanCount;
    UINT8   pad1;
    /**
      The minimum time (in milliseconds) the connection manager will wait for
a
      probe response after sending a probe request.  If no probe responses
are
      received in minChannelTime then the connection manager will go on to
the
      next channel, if any are left to scan, or quit.

      Default: 200ms
      */
    UINT16  minChannelTime;
    /**
      If a probe response is received within minChannelTime then the
connection
      manager will continue to collect any additional probe responses up to
      maxChannelTime before going to the next channel in the channelList.
Units
      are in milliseconds.

      Default: 400ms
      */
    UINT16  maxChannelTime;
    /**
      The number of microseconds to delay before transmitting a probe request
      following the channel change event.

      Default: 20us
```

```
    */
    UINT16  probeDelay;

#if defined (MRF24WG)
    /**
    Default : 4
    */
    UINT16 dtimInterval;

    /**
    Default : 100 (ms)
    */
    UINT16 beaconPrd;
#endif
} tWFCAElements;


/*--------------------------*/
/* used in WF_GetDeviceInfo */
/*--------------------------*/
typedef struct tWFDeviceInfoStruct
{
    UINT8  deviceType;    /* MRF24WB0M_DEVICE_TYPE  */
    UINT8  romVersion;    /* ROM version number     */
    UINT8  patchVersion;  /* Patch version number   */
} tWFDeviceInfo;

#if defined (MRF24WG)
/*--------------------------*/
/* used in WF_CMGetConnectContext */
/*--------------------------*/
typedef struct tWFConnectContextStruct
{
    UINT8  channel;    /* channel number of current connection */
    UINT8  bssid[6];    /* bssid of connected AP    */
} tWFConnectContext;
#endif

/*--------------*/
/* Scan Results */
/*--------------*/
typedef struct
{
    UINT8      bssid[WF_BSSID_LENGTH]; // Network BSSID value
    UINT8      ssid[WF_MAX_SSID_LENGTH]; // Network SSID value

    /**
      Access point configuration
      <table>
        Bit 7       Bit 6       Bit 5       Bit 4       Bit 3       Bit 2
Bit 1       Bit 0
        -----       -----       -----       -----       -----       -----
-----       -----
        WPA2        WPA         Preamble    Privacy     Reserved    Reserved
Reserved    IE
      </table>
```

```
      <table>
      IE        1 if AP broadcasting one or more Information Elements, else 0
      Privacy   0 : AP is open (no security)
                1: AP using security,  if neither WPA and WPA2 set then
security is WEP.
      Preamble  0: AP transmitting with short preamble
                1: AP transmitting with long preamble
      WPA       Only valid if Privacy is 1.
                0: AP does not support WPA
                1: AP supports WPA
      WPA2      Only valid if Privacy is 1.
                0: AP does not support WPA2
                1: AP supports WPA2
      </table>
      */
    UINT8      apConfig;
    UINT8      reserved;
    UINT16     beaconPeriod; // Network beacon interval
    UINT16     atimWindow; // Only valid if bssType = WF_INFRASTRUCTURE

    /**
      List of Network basic rates.  Each rate has the following format:

      Bit 7
      * 0 • rate is not part of the basic rates set
      * 1 • rate is part of the basic rates set

      Bits 6:0
      Multiple of 500kbps giving the supported rate.  For example, a value of
2
      (2 * 500kbps) indicates that 1mbps is a supported rate.  A value of 4
in
      this field indicates a 2mbps rate (4 * 500kbps).
      */
    UINT8      basicRateSet[WF_MAX_NUM_RATES];
    UINT8      rssi; // Signal strength of received frame beacon or probe
response
    UINT8      numRates; // Number of valid rates in basicRates
    UINT8      DtimPeriod; // Part of TIM element
    UINT8      bssType; // WF_INFRASTRUCTURE or WF_ADHOC
    UINT8      channel; // Channel number
    UINT8      ssidLen; // Number of valid characters in ssid

} tWFScanResult;

typedef struct WFHibernate {
    UINT8 state;
    UINT8 wakeup_notice;
} tWFHibernate;

#if defined(WF_CM_DEBUG)
typedef struct
{
    UINT8     byte[12*4];  // Currently, CM has 12 states; 4-byte for each
state info entry.
} tWFCMInfoFSMStats;
```

```
#endif

#if defined (MRF24WG)
    #if defined(WF_USE_MULTICAST_FUNCTIONS)
    typedef struct WFMulticastConfigStruct
    {
        UINT8 filterId;
        UINT8 action;
        UINT8 macBytes[6];
        UINT8 macBitMask;

    } tWFMultiCastConfig;
    #endif

typedef struct
{
    UINT8 ssid[32];
    UINT8 netKey[64];
    UINT16 authType;
    UINT16 encType;
    UINT8 netIdx;
    UINT8 ssidLen;
    UINT8 keyIdx;
    UINT8 keyLen;
    UINT8 bssid[6];
} tWFWpsCred;
#endif /* MRF24WG */

/*-------------------------*/
/* if asserts are enabled */
/*-------------------------*/
#if defined(WF_DEBUG)
    /*-------------------------------------------------------------*/
    /* Module numbers that will be used in the WF_ASSERT macro */
    /*-------------------------------------------------------------*/
    typedef enum
    {
        WF_MODULE_MAIN_DEMO                 = 0,   /* MainDemo.c
*/
        WF_MODULE_WF_CONFIG                 = 1,   /* WF_Config.c
*/
        WF_MODULE_WF_EINT                   = 2,   /* WF_Eint.c
*/
        WF_MODULE_WF_SPI                    = 3,   /* WF_Spi.c
*/
        WF_MODULE_WF_MAC                    = 4,   /* WFMac.c
*/
        WF_MODULE_WF_PARAM_MSG              = 5,   /* WFParamMsg.c
*/
        WF_MODULE_WF_CONNECTION_PROFILE     = 6,   /* WFConnectionProfile.c
*/
        WF_MODULE_WF_CONNECTION_ALGORITHM   = 7,   /*
WFConnectionAlgorithm.c  */
        WF_MODULE_WF_CONNECTION_MANAGER     = 8,   /* WFConnectionManager.c
*/
        WF_MODULE_WF_DRIVER_COM             = 9,   /* WFDriverCom.c
*/
```

```
        WF_MODULE_WF_INIT                       = 10,  /* WFInit.c
*/
        WF_MODULE_WF_DRIVER_RAW                 = 11,  /* WFDriverRaw.c
*/
        WF_MODULE_WF_MGMT_MSG                   = 12,  /* WFMgmtMsg.c
*/
        WF_MODULE_MGMT_MSG_TEST                 = 13,  /* WFMgmtMsgTest.c
*/
        WF_MODULE_WF_TX_POWER                   = 14,  /* WFTxPower.c
*/
        WF_MODULE_WF_POWER_SAVE                 = 15,  /* WFPowerSave.c
*/
        WF_MODULE_EVENT_HANDLER                 = 16,  /* WFEventHandler.c
*/
        WF_MODULE_WF_SCAN                       = 17,  /* WFScan.c
*/
        WF_MODULE_DEBUG_STRINGS                 = 18,  /* WFDebugStrings.c
*/
        WF_MODULE_IPERF_APP                     = 19, /* IperfApp.c
*/
        WF_MODULE_HOST_BRIDGE                   = 20, /* WFHostBridge.c
*/
        WF_MODULE_WF_IPERF_CLIENT               = 21, /* WF_iperfClient.c
*/
        WF_MODULE_WF_IPERF_SERVER               = 22, /* WF_iperfServer.c
*/
        WF_MODULE_WF_IPERF_COMMON               = 23  /* WF_iperfCommon.c
*/

    } tWFModuleNumber;


    void WF_AssertionFailed(UINT8 moduleNumber, UINT16 lineNumber);

    #define WF_ASSERT(expr)                                                 \
       do {                                                                 \
           if (!(expr))                                                     \
              {                                                             \
                   WF_AssertionFailed(WF_MODULE_NUMBER, __LINE__);          \
              }                                                             \
       } while (0)
/*--------------------------*/
/* else asserts are disabled */
/*--------------------------*/
#else
    #define WF_ASSERT(expr) do {} while (0)
#endif  /* WF_DEBUG */


/*
******************************************************************************
**************************
*                                          FUNCTION PROTOTYPES
******************************************************************************
**************************
*/
```

```
/*--------------------------*/
/* Initialization Functions */
/*--------------------------*/
void WF_Init(void);

/*-------------------*/
/* Version functions */
/*-------------------*/
void WF_GetDeviceInfo(tWFDeviceInfo *p_deviceInfo);

/*-------------------*/
/* Retrieve connection context  */
/*-------------------*/
#if defined(MRF24WG)
void WF_CMGetConnectContext(tWFConnectContext *p_ctx);
#endif

/*---------------------------*/
/* WF Driver process function */
/*---------------------------*/
void WFProcess(void);


/*----------------------------------------*/
/* WF Driver External Interrupt function */
/* Must be called when:                   */
/*  1) External interrupt is enabled AND */
/*  2) EXINT line is asserted (low)      */
/*----------------------------------------*/
void WFEintISR(void);

/*-----------------------*/
/* MAC Address Functions */
/*-----------------------*/
void WF_SetMacAddress(UINT8 *p_mac);
void WF_GetMacAddress(UINT8 *p_mac);


/*-----------------------------*/
/* Connection Profile Functions */
/*-----------------------------*/
void WF_CPCreate(UINT8 *p_CpId);
void WF_CPDelete(UINT8 CpId);
void WF_CPGetIds(UINT8 *cpIdList);

#if defined(WF_USE_GROUP_SET_GETS)
void WF_CPSetElements(UINT8 CpId, tWFCPElements *p_elements);
void WF_CASetElementsN(const tWFCAElements *p_elements);
void WF_CPGetElements(UINT8 CpId, tWFCPElements *p_elements);
#endif

#if defined(WF_USE_INDIVIDUAL_SET_GETS)
    void WF_CPSetSsid(UINT8 CpId, UINT8 *p_ssid,  UINT8 ssidLength);
    void WF_CPSetSsidType(UINT8 CpId, UINT8 hidden);
    void WF_CPGetSsidType(UINT8 CpId, UINT8 *hidden);
    void WF_CPGetSsid(UINT8 CpId, UINT8 *p_ssid, UINT8 *p_ssidLength);
    void WF_CPSetBssid(UINT8 CpId, UINT8 *p_bssid);
```

```
    void WF_CPGetBssid(UINT8 CpId, UINT8 *p_bssid);
    void WF_CPSetSecurity(UINT8 CpId,
                          UINT8 securityType,
                          UINT8 wepKeyIndex,
                          UINT8 *p_securityKey,
                          UINT8 securityKeyLength);
    void WF_CPGetSecurity(UINT8 CpId,
                          UINT8 *p_securityType,
                          UINT8 *p_wepKeyIndex,
                          UINT8 *p_securityKey,
                          UINT8 *p_securityKeyLength);
    void WF_CPSetDefaultWepKeyIndex(UINT8 CpId, UINT8 defaultWepKeyIndex);
    void WF_CPGetDefaultWepKeyIndex(UINT8 CpId, UINT8 *p_defaultWepKeyIndex);
    void WF_CPSetNetworkType(UINT8 CpId, UINT8 networkType);
    void WF_CPGetNetworkType(UINT8 CpId, UINT8 *p_networkType);
    void WF_CPSetWepKeyType(UINT8 CpId, UINT8 wepKeyType);
    void WF_CPGetWepKeyType(UINT8 CpId, UINT8 *p_wepKeyType);
#if defined (MRF24WG)
    void WF_CPGetWPSCredentials(UINT8 CpId, tWFWpsCred *p_cred);
#endif
    void WF_CPSetAdHocBehavior(UINT8 CpId, UINT8 adHocBehavior);
    void WF_CPGetAdHocBehavior(UINT8 CpId, UINT8 *p_adHocBehavior);
#endif /* WF_USE_INDIVIDUAL_SET_GETS */

/*--------------------------------*/
/* Connection Algorithm Functions */
/*--------------------------------*/
#if defined(WF_USE_GROUP_SET_GETS)
void WF_CASetElements(tWFCAElements *p_elements);
void WF_CAGetElements(tWFCAElements *p_elements);
#endif

#if defined(WF_USE_INDIVIDUAL_SET_GETS)
    void WF_CASetScanType(UINT8 scanType);
    void WF_CAGetScanType(UINT8 *p_scanType);
    void WF_CASetRssi(UINT8 rssi);
    void WF_CAGetRssi(UINT8 *p_rssi);
    void WF_CASetConnectionProfileList(UINT8 cpList[WF_CP_LIST_LENGTH]);
    void WF_CAGetConnectionProfileList(UINT8 cpList[WF_CP_LIST_LENGTH]);
    void WF_CASetListRetryCount(UINT8 listRetryCount);
    void WF_CAGetListRetryCount(UINT8 *p_listRetryCount);
    void WF_CASetEventNotificationAction(UINT8 eventNotificationAction);
    void WF_CAGetEventNotificationAction(UINT8 *p_eventNotificationAction);
    void WF_CASetBeaconTimeoutAction(UINT8 beaconTimeoutAction);
    void WF_CAGetBeaconTimeoutAction(UINT8 *p_beaconTimeoutAction);
    void WF_CASetDeauthAction(UINT8 deauthAction);
    void WF_CAGetDeauthAction(UINT8 *p_deauthAction);
    void WF_CASetChannelList(UINT8 *p_channelList, UINT8 numChannels);
    void WF_CAGetChannelList(UINT8 *p_channelList, UINT8 *p_numChannels);
    void WF_CASetListenInterval(UINT16 listenInterval);
    void WF_CAGetListenInterval(UINT16 *p_listenInterval);
    #if defined(MRF24WG)
        void WF_CASetDtimInterval(UINT16 dtimInterval);
        void WF_CAGetDtimInterval(UINT16 *p_dtimInterval);
    #endif
    void WF_CASetBeaconTimeout(UINT8 beaconTimeout);
    void WF_CAGetBeaconTimeout(UINT8 *p_beaconTimeout);
```

```
    void WF_CASetScanCount(UINT8 scanCount);
    void WF_CAGetScanCount(UINT8 *p_scanCount);
    void WF_CASetMinChannelTime(UINT16 minChannelTime);
    void WF_CAGetMinChannelTime(UINT16 *p_minChannelTime);
    void WF_CASetMaxChannelTime(UINT16 minChannelTime);
    void WF_CAGetMaxChannelTime(UINT16 *p_minChannelTime);
    void WF_CASetProbeDelay(UINT16 probeDelay);
    void WF_CAGetProbeDelay(UINT16 *p_probeDelay);
    void WF_CASetBeaconPeriod(UINT16 beaconPeriod);
    void WF_CAGetBeaconPeriod(UINT16 *beaconPeriod);
#endif /* WF_USE_INDIVIDUAL_SET_GETS */

/*--------------------------------*/
/* Connection Manager Functions   */
/*--------------------------------*/
void WF_CMConnect(UINT8 CpId);
UINT16 WF_CMDisconnect(void);
void WF_CMGetConnectionState(UINT8 *p_state, UINT8 *p_currentCpId);
void WF_CMCheckConnectionState(UINT8 *p_state, UINT8 *p_currentCpId);

#if defined(MRF24WG)
    void WF_SetTxMode(UINT8 mode);
    void WF_GetTxMode(UINT8 *mode);
    void WFEnableDebugPrint(UINT8 option);
#endif
    void WF_SetLinkDownThreshold(UINT8 threshold);

/*----------------------------*/
/* Tx Power Control Functions */
/*----------------------------*/
#if defined(WF_USE_TX_POWER_CONTROL_FUNCTIONS)
    #if defined (MRF24WG)
        void WF_TxPowerSetMax(INT8 maxTxPower);
        void WF_TxPowerGetMax(INT8 *p_maxTxPower);
    #else /* !defined (MRF24WG) */
        void WF_TxPowerSetMinMax(INT8 minTxPower, INT8 maxTxPower);
        void WF_TxPowerGetMinMax(INT8 *p_minTxPower, INT8 *p_maxTxPower);
        void WF_FixTxRateWithMaxPower(BOOL oneMegaBps);
    #endif /* defined (MRF24WG) */
    void WF_TxPowerGetFactoryMax(INT8 *p_factoryMaxTxPower);
#endif

void WiFiTask(void);

/*----------------------------*/
/* Power Management Functions */
/*----------------------------*/
#if defined(WF_USE_POWER_SAVE_FUNCTIONS)
    void WF_PsPollDisable(void);
    void WF_PsPollEnable(BOOL rxDtim);
    void WF_GetPowerSaveState(UINT8 *p_powerSaveState);
    void WF_HibernateEnable(void);
#endif

/*-------------------------*/
/* RTS Threshold Functions */
/*-------------------------*/
```

```
void WF_SetRtsThreshold(UINT16 rtsThreshold);
void WF_GetRtsThreshold(UINT16 *p_rtsThreshold);

/*-------------------------*/
/* WPS supporting Functions */
/*-------------------------*/
void WF_YieldPassphrase2Host(void);
void WF_SetPSK(UINT8 *psk);

/*---------------------------*/
/* Regional Domain Functions */
/*---------------------------*/
void WF_SetRegionalDomain(UINT8 regionalDomain);     /* see tWFRegDomain
enumerated types */
void WF_GetRegionalDomain(UINT8 *p_regionalDomain);  /* see tWFRegDomain
enumerated types */

/*---------------------*/
/* Multicast Functions */
/*---------------------*/
#if defined(WF_USE_MULTICAST_FUNCTIONS)
    void WF_SetMultiCastFilter(UINT8 multicastFilterId, UINT8
multicastAddress[6]);
    void WF_GetMultiCastFilter(UINT8 multicastFilterId, UINT8
multicastAddress[6]);
    #if defined(MRF24WG)
        void WF_MulticastSetConfig(tWFMultiCastConfig *p_config);
        void WF_EnableSWMultiCastFilter(void);
    #endif

#endif /* WF_USE_MULTICAST_FUNCTIONS */

/* MAC Stats */
void WF_GetMacStats(tWFMacStats *p_macStats);

/*----------------*/
/* Scan Functions */
/*----------------*/
#if defined(WF_USE_SCAN_FUNCTIONS)
UINT16 WF_Scan(UINT8 CpId);
void WF_ScanGetResult(UINT8         listIndex,
                      tWFScanResult *p_scanResult);
#endif /* WF_SCAN_FUNCTIONS */

/*------------------------------*/
/* External Interrupt Functions */
/*------------------------------*/
void WF_EintInit(void);
void WF_EintEnable(void);
void WF_EintDisable(void);
BOOL WF_EintIsDisabled(void);
void WFEintHandler(void);
/* WF_EintIsPending - used by the WF Driver to test for whether */
/* external interrupts are pending.  The pending case is not something */
/* that should normally happen.  It says we have the interrupt line */
/* asserted but the WF_EINT_IF bit is not set, thus, no interrupt generated
*/
```

```
BOOL WF_EintIsPending(void);

/*---------------*/
/* SPI Functions */
/*---------------*/
void     WF_SpiInit(void);
void     WF_SpiEnableChipSelect(void);
void     WF_SpiDisableChipSelect(void);
void     WFSpiTxRx(UINT8   *p_txBuf,
                   UINT16  txLen,
                   UINT8   *p_rxBuf,
                   UINT16  rxLen);

#if defined (__18CXX)
void WFSpiTxRx_Rom(ROM UINT8 *p_txBuf,
                   UINT16    txLen,
                   UINT8     *p_rxBuf,
                   UINT16    rxLen);
#endif /* __18CXX */

/*--------------------------*/
/* Event Handling Functions */
/*--------------------------*/
void WF_ProcessEvent(UINT8 event, UINT16 eventInfo, UINT8 *extraInfo);

#if defined(MRF24WG)
    void WF_DisplayModuleAssertInfo(void);
#endif

#if defined(WF_CM_DEBUG)
/*--------------------------*/
/* CM Info Functions        */
/*--------------------------*/
void WF_CMInfoGetFSMStats(tWFCMInfoFSMStats *p_info);
#endif

extern void WF_DisableModuleConnectionManager(void);

#define SHA1_MAC_LEN 20
extern void pbkdf2_sha1(const char *passphrase, const char *ssid, UINT16
ssid_len,
                        UINT16 iterations, UINT8 *buf, UINT16 buflen);

extern void WF_ConvPassphrase2Key(UINT8 key_len, UINT8 *key, UINT8 ssid_len,
UINT8 *ssid);

#ifdef WIFI_NET_TEST
extern void wifi_net_test_print(char *str, UINT32 param);
#endif

#endif /* __WF_API_H_ */
```

WF_Eint.c

```
/*****************************************************************************
**
```

```
 MRF24W Driver Exernal Interrupt
 Module for Microchip TCP/IP Stack
  -Provides access to MRF24W WiFi controller
  -Reference: MRF24W Data sheet, IEEE 802.11 Standard

*********************************************************************
**
 FileName:      WF_Eint.c
 Dependencies:  TCP/IP Stack header files
 Processor:     PIC18, PIC24F, PIC24H, dsPIC30F, dsPIC33F, PIC32
 Compiler:      Microchip C32 v1.10b or higher
                Microchip C30 v3.22 or higher
                Microchip C18 v3.34 or higher
 Company:       Microchip Technology, Inc.

 Software License Agreement

 Copyright (C) 2002-2010 Microchip Technology Inc.  All rights reserved.

 Microchip licenses to you the right to use, modify, copy, and distribute:
 (i)  the Software when embedded on a Microchip microcontroller or digital
      signal controller product ("Device") which is integrated into
      Licensee's product; or
 (ii) ONLY the Software driver source files ENC28J60.c, ENC28J60.h,
      ENCX24J600.c and ENCX24J600.h ported to a non-Microchip device used in
      conjunction with a Microchip ethernet controller for the sole purpose
      of interfacing with the ethernet controller.

 You should refer to the license agreement accompanying this Software for
 additional information regarding your rights and obligations.

 THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
 KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION, ANY WARRANTY
 OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND
 NON-INFRINGEMENT. IN NO EVENT SHALL MICROCHIP BE LIABLE FOR ANY INCIDENTAL,
 SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, LOST PROFITS OR LOST DATA, COST
 OF PROCUREMENT OF SUBSTITUTE GOODS, TECHNOLOGY OR SERVICES, ANY CLAIMS BY
 THIRD PARTIES (INCLUDING BUT NOT LIMITED TO ANY DEFENSE THEREOF), ANY CLAIMS
 FOR INDEMNITY OR CONTRIBUTION, OR OTHER SIMILAR COSTS, WHETHER ASSERTED ON
 THE BASIS OF CONTRACT, TORT (INCLUDING NEGLIGENCE), BREACH OF WARRANTY, OR
 OTHERWISE.


 Author                Date        Comment
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~
 Michael Palladino    10/13/07    Original
 KO                   31 Oct 2008 Port to PIC24F and PIC32 for TCP/IP stack
v4.52
 KH                   19 Jun 2009 Modified MACMemCopyAsync to support TCB to
TCB copy
 KH                   27 Jan 2010 Updated for MRF24W
*********************************************************************
*/


/*
```

```
*******************************************************************************
*****************************
*                                                 INCLUDES
*******************************************************************************
*****************************
*/

#include "TCPIP Stack/WFMac.h"

#if defined(WF_CS_TRIS)

/* used for assertions */
#if defined(WF_DEBUG)
    #define WF_MODULE_NUMBER  WF_MODULE_WF_EINT
#endif


/*******************************************************************************
*
 * FUNCTION:WF_EintIsDisabled
 *
 * RETURNS: TRUE if MRF24W External Interrupt is disabled, else returns FALSE
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to determine if the MRF24W External
Interrupt
 *          is disabled.

*******************************************************************************
/
BOOL WF_EintIsDisabled(void)
{
    return(WF_INT_IE == 0);  /* works for PIC18, PIC24, and PIC32 */
}

BOOL WF_EintIsPending(void)
{
    return(((WF_INT_IO == 0) && (WF_INT_IF == 0)));  /* works for PIC18,
PIC24, and PIC32 */
}

/*=============================================================================
=========================================*/
/*=============================================================================
=========================================*/
/*                                    PIC18 Interrupt Routines
*/
/*=============================================================================
=========================================*/
/*=============================================================================
=========================================*/
#if defined( __18CXX )
/*******************************************************************************
*
 * PIC18 INTERRUPT SERVICE ROUTINE (Called from LowISR() in MainDemo.c)
```

```
*******************************************************************************
/
void WFEintISR(void)
{
    // if EINT enabled
    if ( WF_INT_IE == 1 )
    {
        // if EINT event occurred
        if ( WF_INT_IF == 1 )
        {
            // clear EINT
            WF_INT_IF = 0;
            WF_INT_IE = 0;          // disable external interrupt

            // invoke handler
            WFEintHandler();
       }
    }
}

/*******************************************************************************
*
 * FUNCTION:WF_EintEnable (Specific to PIC18)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to enable the MRF24W External Interrupt.

*******************************************************************************
/
void WF_EintEnable(void)
{
    // if interrupt line is low, then we may have missed a falling edge
    // while the interrupt was disabled.
    if ( WF_INT_IO == 0 )
    {
        // if the interrupt pin is active
        // then the MRF24W has another event that needs to be serviced.
        // This means that the MRF24W will never generate another falling
edge
        // required to trigger the interrupt... So, we must force an
interrupt.
        // force the EINT
        WF_INT_IF = 1;
    }

    /* enable the external interrupt */
    WF_INT_IE = 1;
}

/*******************************************************************************
*
 * FUNCTION:WF_EintDisable (Specific to PIC18)
 *
 * RETURNS: None
```

```
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to disable the MRF24W External Interrupt.

*****************************************************************************
/
void WF_EintDisable(void)
{
    /* disable the external interrupt */
    WF_INT_IE = 0;
}

/*****************************************************************************
*
 * FUNCTION:WF_EintInit (Specific to PIC18)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to initialize the MRF24W External
Interrupt.

*****************************************************************************
/
void WF_EintInit(void)
{
    /* disable the external interrupt */
    WF_INT_IE = 0;
    WF_INT_IP = 0;

    /* configure IO pin as input and External Interrupt pin*/
    /* assume port b pullups were enabled before entry */
    WF_INT_TRIS = 1;
    WF_INT_EDGE = 0;   /* falling edge triggered */

    /* clear and enable the interrupt */
    WF_INT_IF = 0;
    WF_INT_IE = 1;
}

/*============================================================================
=========================================*/
/*============================================================================
=========================================*/
/*                                  PIC24 Interrupt Routines
*/
/*============================================================================
=========================================*/
/*============================================================================
=========================================*/
#elif defined( __C30__ )

/*****************************************************************************
*
 * PIC24 INTERRUPT SERVICE ROUTINE
```

```
*****************************************************************************
/
#if defined(MRF24W_IN_SPI2 )
    void __attribute__((interrupt, auto_psv)) _INT3Interrupt(void)
#else
    void __attribute__((interrupt, auto_psv)) _INT1Interrupt(void)
#endif
{
    // clear EINT
    if (WF_INT_IF && WF_INT_IE)
    {
        WF_INT_IF = 0;
        WF_INT_IE = 0;         /* disable external interrupt */
        // invoke handler
        WFEintHandler();
    }
}

/*****************************************************************************
*
 * FUNCTION: WF_EintEnable (Specific to PIC24)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to enable the MRF24W External Interrupt.

*****************************************************************************
/
void WF_EintEnable(void)
{
    // if interrupt line is low, then we may have missed a falling edge
    // while the interrupt was disabled.
    if ( WF_INT_IO == 0 )
    {
        // if the interrupt pin is active
        // then the MRF24W has another event that needs to be serviced.
        // This means that the MRF24W will never generate another falling
edge
        // required to trigger the interrupt... So, we must force an
interrupt.
        // force the EINT
        WF_INT_IF = 1;
    }

    /* enable the external interrupt */
    WF_INT_IE = 1;
}


/*****************************************************************************
*
 * FUNCTION: WF_EintDisable (Specific to PIC24)
 *
 * RETURNS: None
```

```
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to disable the MRF24W External Interrupt.

*****************************************************************************
/
void WF_EintDisable(void)
{
    /* disable the external interrupt */
    WF_INT_IE = 0;
}

/*****************************************************************************
*
 * FUNCTION: WF_EintInit (Specific to PIC24)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to initialize the MRF24W External
Interrupt.

*****************************************************************************
/
void WF_EintInit(void)
{
    /* disable the external interrupt */
    WF_INT_IE = 0;

    /* configure IO pin as input and External Interrupt pin*/
    /* set the I/O high since we do not have pull-ups */
    WF_INT_IO   = 1;
    WF_INT_TRIS = 1;
    WF_INT_EDGE = 1;   /* falling edge triggered */

    /* clear and enable the interrupt */
    WF_INT_IF = 0;
    WF_INT_IE = 1;
}

/*=============================================================================
==========================================*/
/*=============================================================================
==========================================*/
/*                                    PIC32 Interrupt Routines
*/
/*=============================================================================
==========================================*/
/*=============================================================================
==========================================*/
#elif defined( __PIC32MX__ )

/*****************************************************************************
*
 * PIC32 INTERRUPT SERVICE ROUTINE
```

```
*****************************************************************************
/
#if defined( MRF24W_IN_SPI2 )
void __attribute((interrupt(ipl3), vector(_EXTERNAL_3_VECTOR), nomips16))
_WFInterrupt(void)
#else
void __attribute((interrupt(ipl3), vector(_EXTERNAL_1_VECTOR), nomips16))
_WFInterrupt(void)
#endif
{
    // clear EINT
    if (WF_INT_IF && WF_INT_IE)
    {
        WF_INT_IF_CLEAR = WF_INT_BIT;
        WF_INT_IE_CLEAR = WF_INT_BIT;         /* disable external interrupt
*/

        /* invoke handler */
        WFEintHandler();
    }
}

/*****************************************************************************
*
 * FUNCTION:WF_EintEnable (Specific to PIC32)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to enable the MRF24W External Interrupt.

*****************************************************************************
/
void WF_EintEnable(void)
{
    // if interrupt line is low, then we may have missed a falling edge
    // while the interrupt was disabled.
    if ( WF_INT_IO == 0 )
    {
        // if the interrupt pin is active
        // then the MRF24W has another event that needs to be serviced.
        // This means that the MRF24W will never generate another falling
edge
        // required to trigger the interrupt... So, we must force an
interrupt.
        // force the EINT
        WF_INT_IF_SET = WF_INT_BIT;
    }

    /* enable the external interrupt */
    WF_INT_IE_SET = WF_INT_BIT;
}


/*****************************************************************************
```

```
*
 * FUNCTION:WF_EintDisable (Specific to PIC32)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to disable the MRF24W External Interrupt.

*****************************************************************************
/
void WF_EintDisable(void)
{
    /* disable the external interrupt */
    WF_INT_IE_CLEAR = WF_INT_BIT;
}

/*****************************************************************************
*
 * FUNCTION:WF_EintInit (Specific to PIC32)
 *
 * RETURNS: None
 *
 * PARAMS:  None
 *
 * NOTES:   Called by WiFi Driver to initialize the MRF24W External
Interrupt.

*****************************************************************************
/
void WF_EintInit(void)
{
    /* disable the external interrupt */
    WF_INT_IE_CLEAR = WF_INT_BIT;

    /* configure IO pin as input and External Interrupt pin*/
    /* set the I/O high since we do not have pull-ups */
    WF_INT_IO   = 1;
    WF_INT_TRIS = 1;
    WF_INT_EDGE = 0;   /* falling edge triggered */

    /* clear and enable the interrupt */
    WF_INT_IF_CLEAR    = WF_INT_BIT;
    WF_INT_IPCCLR      = WF_INT_IPC_MASK;
    WF_INT_IPCSET      = WF_INT_IPC_VALUE;
    WF_INT_IE_SET      = WF_INT_BIT;
}
#endif

#else
    // dummy func to keep compiler happy when module has no executeable code
    void MCHP_Eint_EmptyFunc(void)
    {
    }

#endif /* WF_CS_TRIS */
```

8 Acknowledgements

The Light of the World would like to thank several people for their help with our project. First, we would like to thank Dr. Michael Schafer, our course instructor, for all of his helpful input and design advice throughout our both semesters. Second, we like to thank BJ Yurkovich for his donation of 7 WB0MB wifi chips and his advice throughout the project. We would also like to thank Power Integrations for their assistance with our transformers and their outstanding customer service. We would also like to thank Natalie Geddes for her help in operating the Fortus 3D printer. Finally, we would like to thank all of our professors in the Electrical Engineering department and people in the College of Engineering at the University of Notre Dame who helped make all of our work possible.