

Wi-Fi Evapotranspiration

Irrigation Controller

by

Ryan Goodman

Senior Project

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

Winter 2010

Abstract

This report discusses the applications of an Irrigation Controller with Wi-Fi connectivity, with focus on cost-effectiveness and usability. It addresses the amount to which traditional irrigation is wasteful, and proposes solutions to solve this problem. Methods for using online weather data resources to increase efficiency of residential landscaping are discussed in depth. Included is the procedural development of an irrigation controller using the Microchip Explorer 16 Development Board and TCP/IP Stack. Also discussed is web hosting and use of the scripting language PHP, and database MySQL. This project found inexpensive means for providing HTTP and FTP services. Results indicate an easy to use, simplistic user experience. The site's pages provide the only control of the Irrigator and is a good tool for creating efficient irrigation schedules, either automatically or manually. Also, the use of Evapotranspiration showed water savings when compared with models of average residential water use, especially in wet seasons. In summary, further development is necessary to make it a marketable product. Ideas for expanding upon it are included.

Table of Contents

Chapters	Page
Abstract.....	<i>i</i>
List of Tables and Figures.....	<i>iii</i>
Acknowledgments.....	<i>iv</i>
I. Introduction.....	1
II. Background	3
III. Requirements.....	9
IV. Design.....	10
V. Procedure.....	23
VI. Testing.....	28
VII. Results and Analysis.....	30
VIII. Conclusions and Recommendations.....	32
IX. Bibliography.....	34
Appendices	
A. Costs.....	37
B. Software Lists.....	38
C Senior Project Analysis.....	39

List of Tables and Figures

Tables	Page
1. Table I: TCP/IP Stack Members Used.....	26

Figures	Page
1. Figure 1: Penman-Monteith Equation.....	3
2. Figure 2: LAN Block Diagram.....	10
3. Figure 3: setup.php Screenshot.....	12
4. Figure 4: edit.php Screenshot.....	13
5. Figure 5: manualMode.php Result.....	14
6. Figure 6: get.php Result.....	15
7. Figure 7: Main State Machine.....	16
8. Figure 8: Irrigator State Machine.....	17
9. Figure 9: FTP State Machine.....	19
10. Figure 10: HTTP State Machine.....	21
11. Figure 11: Explorer 16 Development Board.....	24

Acknowledgments

I would like to thank my father for his insight. I sincerely wish you have more people to chat with about this stuff. I would like to sincerely thank Dr. Harris for his guidance being my adviser. You have given me great opportunity. Thank you, Dr. Burgoa for your insight, understanding, and patience. To my buddy David, thanks for your advice. I would be lost without people like you around me.

I. Introduction

Residents of the United States, on average, use over seven billion gallons of water a day outdoors, primarily on landscaping^[1]. Many people know little about their own use, and how inefficient their watering methods are. In the past, I've stepped around water streaming into the gutter from someone's lawn or felt the mist of a sprinkler blow past me, and I didn't realize at the time how much was being wasted. An average lawn is watered 10,000 gallons per year^[2] and often over half of it never reaches the plants it's intended to^[3].

The way to increase efficiency is to only provide landscapes with what they need, much like how we use the thermostat. That's where Evapotranspiration (ET) is useful. ET is a measurement of the minimum amount of water plants need to be healthy; it is the amount of water plants consume and the amount evaporated off of it^[4]. When irrigators use ET they use less water. For instance, the Escondido School System reduced water use 30% at 18 schools saving 32 million gallons a year^[5].

Homeowners can realize the same reductions in water use as many irrigation professionals do. Most of the systems we already use have the potential to be much more efficient with periodic adjustments. It is just easier to "set it and forget it".^[6] On the other hand, many of us have very little difficulty navigating the internet everyday.

The goal of this project is to build an easy to use and more efficient irrigator system for everyday residential users. The system will connect to the internet through a user's home Wi-Fi router where it is setup and controlled by navigating a website. The default setting will be to automatically download weather data and calculate the time each valve is activated. Using this mode prevents the need for users to make the calculations and adjustments required for maximum efficiency.

Section II presents the Background, with some history behind the equations and services used. Then the Requirements of the project, and its components, follow. Next in section IV, I discuss Design, the implementation and structure for the controller and website. Section V Procedure contains the process of the project's development. Next section VI Testing presents testing methods I used, and is followed by the tests' Results and Analysis in section VII. Section VII Conclusions and Recommendations summarize the final performance and future improvements to the system. The information for references made in this report are in the Bibliography. The Appendices present Costs of the project in A, then a listing of software code follow in B, and in C the Senior Project Analysis is included at the end of this document.

II. Background

i. Evapotranspiration

In 1948, Howard Penman, an English micro-climatologist, made a major break-thru in Irrigation Science. By studying the energy exchange on a water, soil, and grass surfaces, he was able to formulate the Penman equation as a way to model evaporation.^[7] Then in 1965, John Monteith made some improvements on his equation, forming the Penman-Monteith equation for Evapotranspiration.^[8] The major difference is that the new equation is more applicable to vegetative surfaces. Their equation is still regarded as the most accurate estimate of daily Evapotranspiration.^[9]

$$\lambda_v E = \frac{\text{Energy flux rate}}{\Delta + \gamma (1 + g_a/g_s)} \iff ET_o = \frac{\text{Volume flux rate}}{(\Delta + \gamma (1 + g_a/g_s)) \lambda_v}$$

Figure 1: Penman-Monteith Equation^[7]

λ_v = Latent heat of vaporization (J/g)

L_v = Volumetric latent heat of vaporization ($L_v = 2453 \text{ MJ m}^{-3}$)

E = Mass water evapotranspiration rate ($\text{g s}^{-1} \text{ m}^{-2}$)

ET_o = Water volume evapotranspired ($\text{m}^3 \text{ s}^{-1} \text{ m}^{-2}$)

Δ = Rate of change of saturation specific humidity with air temperature. (Pa K^{-1})

R_n = Net irradiance (W m^{-2}), the external source of energy flux

c_p = Specific heat capacity of air ($\text{J kg}^{-1} \text{K}^{-1}$)

ρ_a = dry air density (kg m^{-3})

δe = vapor pressure deficit, or specific humidity (Pa)

g_a = Conductivity of air, atmospheric conductance (m s^{-1})

g_s = Conductivity of stoma, surface conductance (m s^{-1})

γ = Psychrometric constant ($\gamma \approx 66 \text{ Pa K}^{-1}$)

Then in 1999 the ASCE (American Society of Civil Engineers) took it a step further and, in an effort to standardize the method of calculating reference ET, created the Standardized Reference Evapotranspiration Equation. This new version adapted the Penman-Monteith equation to include hourly Evapotranspiration and used both a short and tall reference crops. The equation uses measurements of air temperature, humidity, solar radiation, and wind speed.

ii. CIMIS

The California Irrigation Management Information System (CIMIS) was formed in 1992 for Irrigation Managers to increase water use efficiency in California.

This program uses a network of 120 automated weather stations across California to record real-time data and calculate hourly and daily averages.^[10] Their data records include values for solar radiation, soil temperature, air temperature, vapor pressure, wind speed, relative humidity, and precipitation. They offer charts on hourly and daily measurements free-of-charge in various formats. Also in these reports is a calculated hourly and daily average of reference ET, using the Penman-Monteith equation.

iii. Irrigation Scheduling

An effective method for efficient Irrigation Management is the use of irrigation scheduling. The difficulty often lies in irrigation system owners' lack of irrigation schedule use.

“For maximum water efficiency, irrigation system controllers should be reprogrammed at least monthly to respond to changing rainfall and temperature conditions. In reality, few homeowners or landscape managers make such adjustments either because they don't know how or because they simply neglect to operate the controller properly. ... The most common-and most inefficient- practice with an irrigation system controller is to 'set it and forget it' once a season.”^[6]

Irrigation schedules aren't necessarily difficult to make, but can be intimidating to inexperienced people. Mostly its estimating factors based on a landscape's composition.

Traditionally ET has been a tool of Agriculture Professionals. ET for an individual crop is the product of the reference ET (ET_0) and the Crop Coefficient (K_C). K_C is the relation of a crop's Evapotranspiration and the reference crop's Evapotranspiration, and it is dependent on the crop type, stage of growth, and reference crop used.^[11] This doesn't apply directly to a landscape, because, unlike a field of corn, different plant species share a water source and the habitat isn't always a large flat open area. So, we introduce a new coefficient, K_L , the landscape coefficient. It works just the same as K_C , but it is not as simple as looking up it's value in the back of a book.

The Species Coefficient, K_S , represents the ET of the plant in relation to the ET of the reference crop. Individual plants can require a different amount of water compared to another of the same species in a different region. Seasonal differences are not significant to landscapes, and are considered in some mid-season agricultural crops. So, K_S can be found in references, but is not necessarily the same in every one. For instance, CIMIS provides some references in [A Guide to Estimating Irrigation Water Needs of Landscape Plantings in California](#).^[12] In Part 2, they provide charts with estimates of K_S split by region and species. The charts have four values represented by letters corresponding to K_S ranges: “High (H): 70 - 90% ET_0 ; Moderate (M): 40 - 60% ET_0 ; Low (L): 10 - 30% ET_0 ; and Very Low (VL): <10% ET_0 ”^[12] So, in San Luis Obispo, the California Lilac is VL, $K_S < 10\%$, and bamboo is H, $70\% < K_S < 90\%$. If

you have an area with mixed K_S it is best to use the highest value, but be careful not to damage the other plants with over-watering.

K_D is the density coefficient, and it represents the amount of canopy cover or the ground that is shaded. It ranges from 0.5 to 1.3, sparse to dense.^[12] What is also important to consider is vertical density. A grass lawn may seem very dense, but in K_D terms it is not. An example of a very dense landscape is one with vertical density also, like low shrubs, a few bushes, and a tree.

K_{mc} is the micro-climate factor, and it ranges from 0.5-1.4.^[12] An average K_{mc} is a landscape similar to the reference crop conditions, like an open area without extreme wind or heat. A high K_{mc} comes from extra heat from heat-absorbing surfaces, like asphalt or concrete, or higher than average winds. Low K_{mc} often comes from being in the shade most of the day or having high walls shielding the wind.

K_L is the product of these three factors. So, just like crop efficiency $ET_L = K_L * ET_0$, where ET_L is the ET for the landscape. The amount of natural precipitation (NP) and artificial precipitation is equal to the ET_L in the ideal landscape.

Irrigation Efficiency (IE) is important too, and it represents the amount of water that reaches the targeted plants. The rest is runoff or blown away as mist. Typically this ranges from 50-70%, but very well-planned professional landscaping can reach up to 90%. The total water applied (TWA) is the ratio of ET_L and IE.^[11]

$$TWA = ET_L / IE$$

Precipitation Rate (PR) is the amount of in/hr watered in a valve's zone. It can be measured using a rain gauge or a non-tapered can or cup and a ruler. It can also be calculated by multiplying the GPM from the sprinkler's specifications with 96.25, the conversion factor of (in*min*sq. ft)/(hr*gal) , and dividing by the area covered by watering.^[13] Finally, the time to water with this valve is the ratio of TWA and PR.

$$T_{\text{hours}} = \text{TWA} / \text{PR}$$

Everything together makes the equation:

$$T_{\text{hours}} = ((K_D * K_S * K_{mc} * ET_0) - NP) / (\text{PR} * \text{IE})$$

K_S = Species Factor (unitless)

K_D = Density Factor (unitless)

K_{mc} = Micro-climate Factor (unitless)

ET_0 = Reference Evapotranspiration (in.)

NP = Natural Precipitation

PR = Precipitation Rate (in./hr)

IE = Irrigation Efficiency (unitless)

III. Requirements

i. Website

The website will act as the irrigation system's user interface. Users will connect using an internet browser and access a control server via HTTP protocols. It will allow the user to create new accounts and with an account they will be able to control an irrigator. Control will consist of editing irrigator parameters and generating irrigation schedules. There will be an automatic and manual mode of operation. In the automatic mode, the server will poll weather data and generate activation times for each of the irrigator's valves. In the manual mode, the user can assign activation times for the irrigator valves.

ii. Hardware

The hardware will simulate the operation of an irrigator, by activating an array of LEDs representing activating irrigation valves. For demonstration purposes, the irrigator will continually update the activation times for each of its LEDs. Updating occurs by accessing the server, and downloading instructions. It will communicate with the server using a wireless IEEE 802.11b standard TCP connection to a wireless access point. The wireless access point will be a common household router.

IV. Design

i. Network

The network, as seen in Figure 2, is a simple local area network (LAN) between the PC, development board, and router. The PC hosts the website and instructions at the IP address 192.168.1.101. It connects wirelessly to the router, and accepts incoming requests on HTTP and FTP ports, 80 and 21. The development board is also connected wirelessly to the router; its IP address is 192.168.1.100. The router's IP address is 192.168.1.1 and it is the gateway to connections outside the LAN. All of the wireless connections are IEEE 802.11b standard protocol.

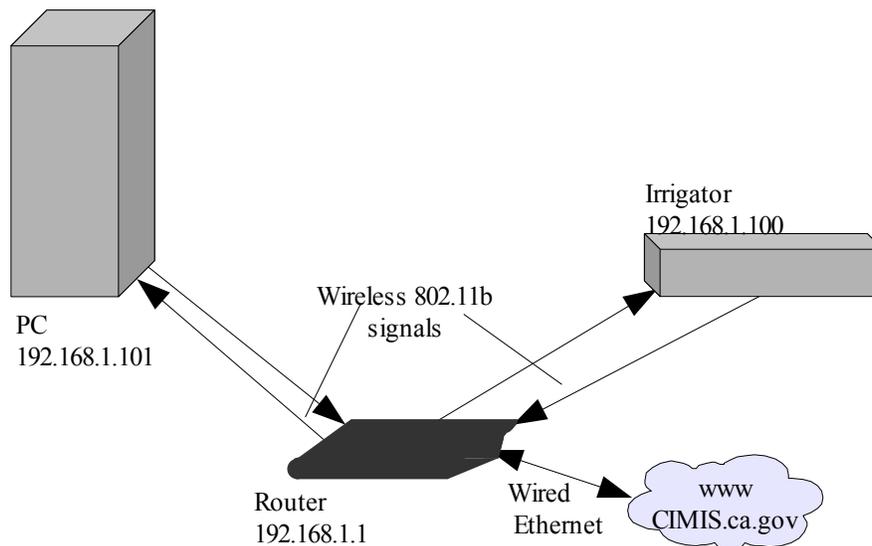


Figure 2: LAN Block Diagram

ii. Website

The website consists of 5 PHP pages, each hosted on the PC with the WAMP Server. The first page is for creating the database tables, one for each individual Irrigator. Next, is the edit page where the irrigator's values can be set. There are two modes of generating an irrigation schedule, manual and automatic, each with its own PHP page. Finally, to help maintain the controller, a PHP page for clearing the stored history is included.

The create controller page starts with a table full of forms, each assigned to a valve's parameters. Figure 2 shows what the interface looks like. There are 5 parameters each for 8 valves, making a total of 40 forms. It functions simply as an interface for creating the MySQL database easily, while avoiding creating a table that already exists. If one already exists it edits the entries of the previous table. The parameters are K_S , K_D , K_{mc} , IE, and PR.

[Home](#)

Irrigator ID

Valve 1	Valve 2	Valve 3	Valve 4
Ks: <input type="text" value="1"/>	<input type="text" value="1"/>	Ks: <input type="text" value="1"/>	Ks: <input type="text" value="1"/>
Kd: <input type="text" value="1"/>	<input type="text" value="1"/>	Kd: <input type="text" value="1"/>	Kd: <input type="text" value="1"/>
Kmc: <input type="text" value="1"/>			
IE: <input type="text" value="0.8"/>			
Pr: <input type="text" value="0"/>			
Valve 5	Valve 6	Valve 7	Valve 8
Ks: <input type="text" value="1"/>			
Kd: <input type="text" value="1"/>			
Kmc: <input type="text" value="1"/>			
IE: <input type="text" value="0.8"/>			
Pr: <input type="text" value="0"/>			

Figure 3: setup.php Screenshot

It starts with the HTML code for the forms, display shown in Figure 3, then begins the PHP script code. First, the link to the mySQL is established, and a query is sent to switch to the correct database, named etscript. Next, it queries using CREATE TABLE IF NOT EXISTS command. So, if the database already contains the table it skips writing it. The next query is REPLACE, which writes over existing rows, if at that index, or creates a new row if the index doesn't exist. The index for this table is valveNo, the valve number 1-8. The REPLACE query is run 8 times one for each valve. The data it writes in comes from the forms submitted via a POST action. Other columns are left alone at their default values. It's final action is to generate and

display the resulting values read back from the database.

Irrigator Number 0001

valveNo	speciesFactor	densityFactor	microClimateFactor	landFactor	irrigationEfficiency	precipitationRate
1	1.00	1.00	1.00	1.00	0.80	0.2000
2	1.00	1.00	1.00	1.00	0.80	0.3000
3	1.00	1.00	1.00	1.00	0.80	0.4000
4	1.00	1.00	1.00	1.00	0.80	0.5000
5	1.00	1.00	1.00	1.00	0.80	0.1000
6	1.00	1.00	1.00	1.00	0.80	0.1500
7	1.00	1.00	1.00	1.00	0.80	0.2500
8	1.00	1.00	1.00	1.00	0.80	0.3500

Irrigator ID 1 ▾

Valve 1	Valve 2	Valve 3	Valve 4
Ks: 1.00	Ks: 1.00	Ks: 1.00	Ks: 1.00
Kd: 1.00	Kd: 1.00	Kd: 1.00	Kd: 1.00
Kmc: 1.00	Kmc: 1.00	Kmc: 1.00	Kmc: 1.00
IE: 0.80	IE: 0.80	IE: 0.80	IE: 0.80
Pr: 0.2000	Pr: 0.3000	Pr: 0.4000	Pr: 0.5000
Valve 5	Valve 6	Valve 7	Valve 8
Ks: 1.00	Ks: 1.00	Ks: 1.00	Ks: 1.00
Kd: 1.00	Kd: 1.00	Kd: 1.00	Kd: 1.00
Kmc: 1.00	Kmc: 1.00	Kmc: 1.00	Kmc: 1.00
IE: 0.80	IE: 0.80	IE: 0.80	IE: 0.80
Pr: 0.1000	Pr: 0.1500	Pr: 0.2500	Pr: 0.3500

Figure 4: edit.php Screenshot

Editing the controller really does the same thing as creating one, because of the way that MySQL database queries handle creating a data table that already exists. The differences are that the default values in the forms are the current values in the database for the Edit Controller page, and reasonable values for the New Controller page. For each valve of a controller, there are Ks, Kd, Kmc, IE, and PR. Ks is the

species factor, Kd is the density factor, and Kmc is the micro-climate factor for the valve. These are multiplied together and stored as the landscape factor. IE is the irrigation efficiency and PR is the precipitation rate. Both of these are also stored in the database, for each valve. So, a total of 40 inputs are given when submitted, but some are combined and only 24 values are stored. After being stored the contents of the irrigator's database table are read and displayed on a new page, see Figure 4.

Irrigator ID

Valve 1:

Valve 2:

Valve 3:

Valve 4:

Valve 5:

Valve 6:

Valve 7:

Valve 8:

Valve 1	Valve 2	Valve 3	Valve 4	Valve 5	Valve 6	Valve 7	Valve 8
45	2300	210	1222	650	400	0	0

Figure 5: manualMode.php Result

The Manual Mode page, Figure 5, accepts 8 inputs and they are the desired amount of time for each valve to be activated in seconds. This is similar to setting the times on non-ET irrigation controllers. The inputs are stored in the database, then read back and displayed in a table. Each value is also given the appropriate number of

leading zeros and written in order to isXXXX.html. XXXX is representing the controller number 0001 in the case of the irrigator.

Irrigator ID ET0 sum: 0.98 inches Precipitation sum: 0.16 inches

Valve 1	Valve 2	Valve 3	Valve 4	Valve 5	Valve 6	Valve 7	Valve 8
1926	1285.2	964.8	770.4	3852	2566.8	1540.8	1101.6

Figure 6: get.php Result

The Automatic Mode page, get.php, accepts no inputs. It logs into the CIMIS website and downloads the last seven days of hourly sensor readings for the station number of the irrigator and extracts the precipitation amounts and the reference ETs for each hour and sums them into two totals. The total inches in ETo is the summed ETo subtracted by the total precipitation and by the sum of the amount watered the last 6 days. This value is the water required for that day in inches. It is then divided by the precipitation rate to get the time to water in hours, which is multiplied by 3600 to get the time in seconds, see Figure 6. Each value in seconds is given the appropriate number of leading zeros and written in order to isXXXX.html.

iii. Hardware

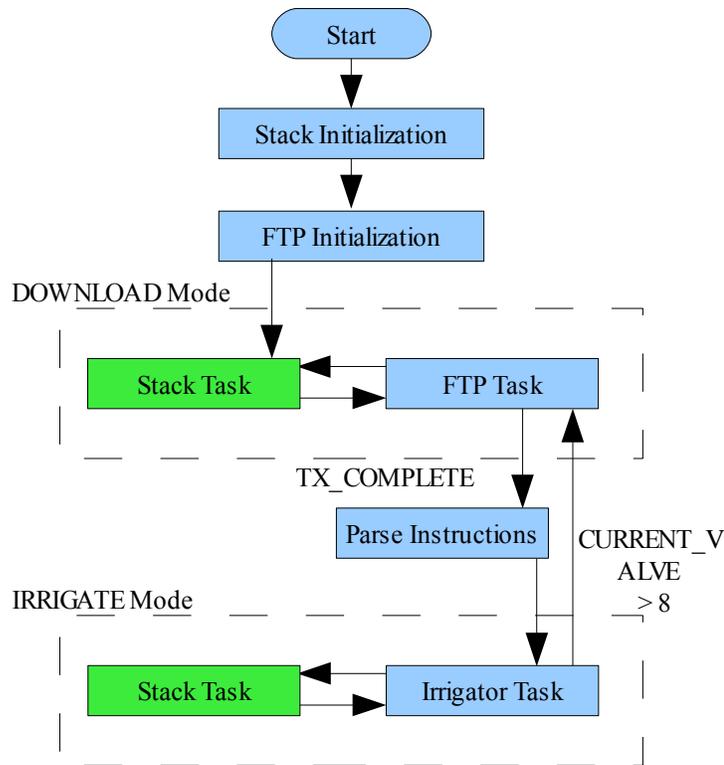


Figure 7: Main State Machine

The Main state machine is presented in Figure 7. The program runs through initializations, then into an infinite loop where the three different tasks perform one after another. This is cooperative multitasking, and it is to allow different parts of the program to perform simultaneously without taking complete control of the MCU. The program must be written so that tasks are performed in small quick portions with returns to the main loop in between. The Stack Task performs internal tasks for the TCP/IP. The Stack Application Task performs the program's network tasks, mostly

transferring and receiving the commands and data. The Irrigator Task manipulate the LED outputs.

The Irrigator Task, see Figure 8, uses a state machine with one state for each of eight valves. It uses three variables: `START_TIME` holds the time that the current valve was activated; it only is updated when the irrigator switches activates a new valve. `CURRENT_VALVE` holds the value of the state machine and corresponds to

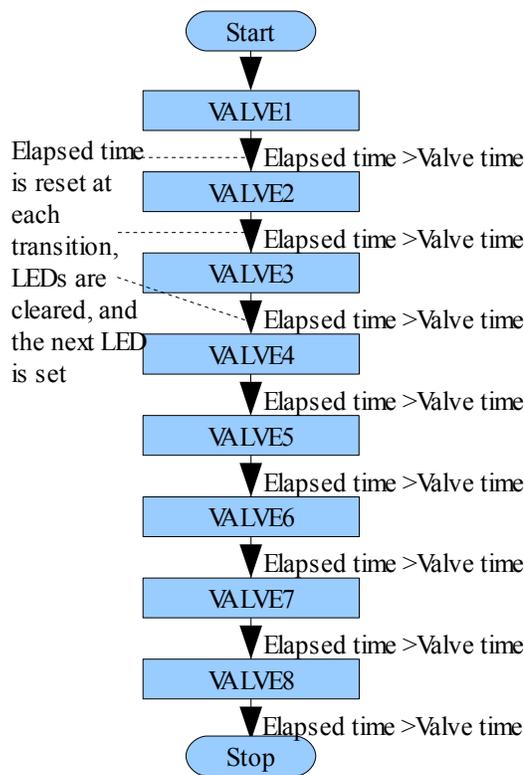


Figure 8: Irrigator State Machine

the valve currently in use. In between states it is incremented to the next state, or valve. `IRR_TIMER` stores the maximum amount of time the current valve will be active. In the beginning of each Irrigator Task call, the program compares the time elapsed since the current valve was activated with the maximum time and breaks if

time elapsed is still the lesser. When the maximum is reached, first , the LED is cleared, the next valve in-line is activated, and START_TIME is set to the current time. Then, via a switch statement, the active valve sets IRR_TIMER to the maximum time it is to elapse before the next valve is activated and turns on its corresponding LED. This process repeats itself until the final valve has finished, then the first is reactivated, starting all over.

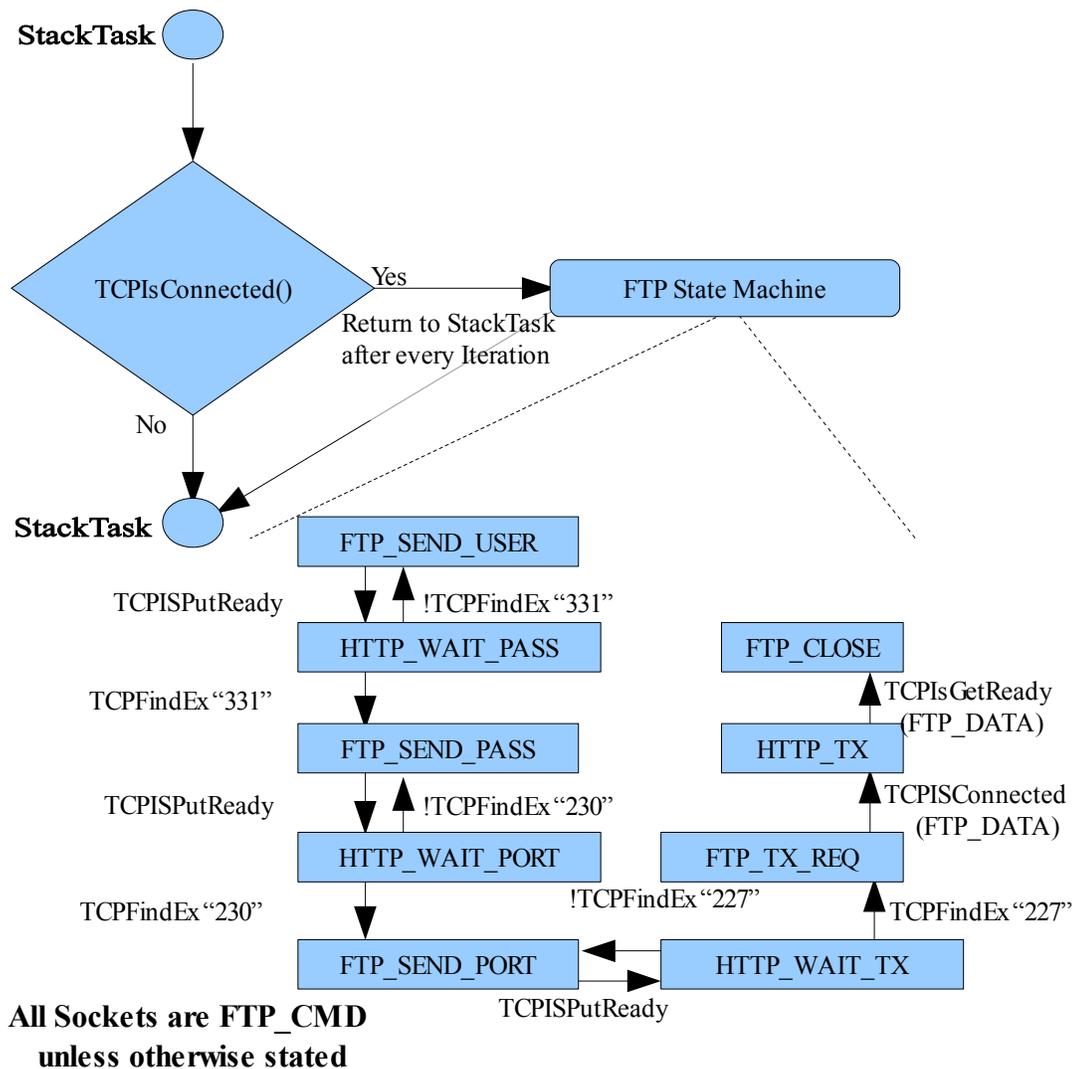


Figure 9: FTP State Machine

In order to retrieve the timer values from the web server, I first used an FTP Client, see Figure 9. The goal was to download and parse a .txt file generated and hosted by the web server. In order to avoid flooding the server with excess commands, a small delay was added before the FTP Client was called. It takes a little less than one second after the command connection was opened in the FTP Client

Initialization. The FTP Client implements a state machine, much like the Irrigator Task, but each state corresponds to a stage in the FTP session. At the start of the FTP Client Task whether or not the command is connected; if it currently isn't, the current state is saved temporarily, the state set to `FTP_NOT_CONNECTED`, and then changed back to the previous state before ending the current iteration. The initial state is `FTP_WAIT_USER` where it stays until it receives the welcome message preceded by "220" and moves to the next state. `FTP_SEND_USER` sends the command "USER *username*" and steps into the next state automatically. There, in the `FTP_WAIT_PASS` state, it goes to `FTP_SEND_PASS` if it received "331" response, or back to `FTP_SEND_USER` if it hasn't. In `FTP_SEND_PASS`, "PASS *password*" is sent and the state becomes `FTP_WAIT_PORT`. This next state works the same as the send password one, except "PASV" is sent in `FTP_SEND_PORT`, "230 login Successful" is the response, and `FTP_WAIT_TX` is after. In `FTP_WAIT_TX`, it waits for "227" response then opens an FTP data connection and sets the state to `FTP_TX_REQ`. The full response includes information about IP and port numbers to connect to, but I keep them fixed so that calculating them isn't necessary. `FTP_TX_REQ` sends the "GET *filename*" command and moves to `FTP_TX` where the data is received from the RX buffer of the data connection and stored. `FTP_CLOSE` comes after `FTP_TX`, and it closes both the data and command connections.

When I used the FTP Client it often disconnected immediately after logging in

or receiving the “PASV” command. In the middle of its development I had the idea of using an HTTP connection instead of an FTP one, see Figure 10. This is more efficient because it only uses a single connection to the server, and it requires less time, because of not requiring a long back-and-forth handshake process.

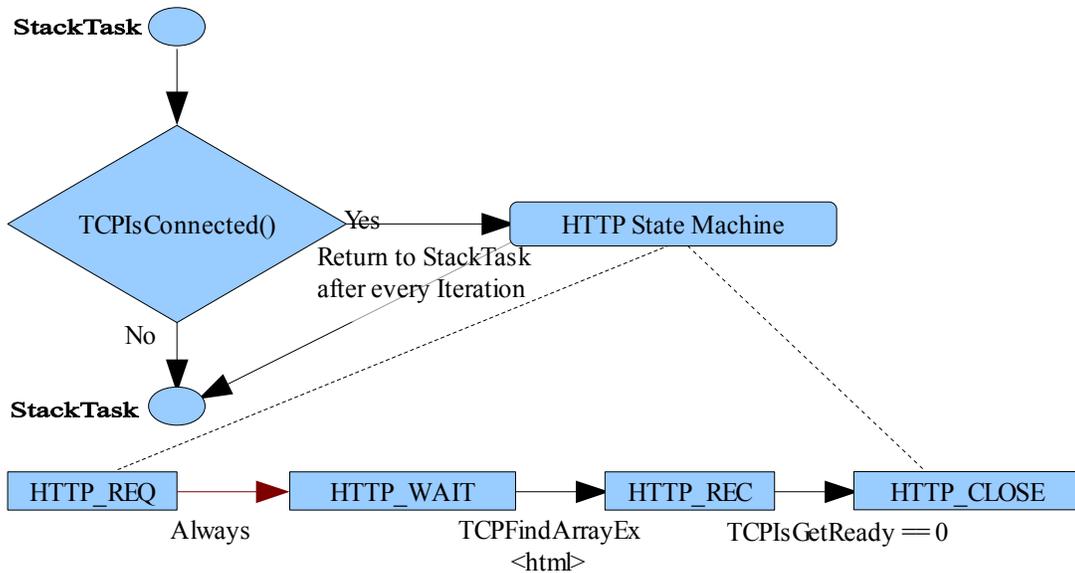


Figure 10: HTTP State Machine

The HTTP Client uses port 80 and downloads a HTML file directly from the server instead of getting a text file through an FTP server program. It moves through 4 states, and like the FTP Client includes a HTTP_NOT_CONNECTED state to allow it to bypass the other states when the TCP connection isn't currently connected. The next and initial state is HTTP_REQ where it sends a “GET /isXXXX.html” to the host IP. It then waits in HTTP_WAIT state until it receives data containing “<html>”. Then it goes to HTTP_REC where it discards everything before the HTML tag and

stores the rest. Lastly, it moves to HTTP_CLOSE where it closes the connection.

V. Procedure

My first task was selecting the hardware I was going to use for the irrigator. I chose to focus on the Wi-Fi module first, because it would be the least common component of the hardware. There are many different modules to pick from, so I decided on some features I wanted to help narrow the search down. In order to connect to a wireless router, the module would have to be 802.11b or 802.11g certified. Next, I wanted to use a module that was relatively inexpensive at low volume and was scalable in purchasing. The idea was to have realistic model for the hypothetical end product. Finally, in order to save time, it should have drivers available; it would require more equipment to develop and be a very time consuming endeavor to write my own.

With this criteria in mind, I narrowed the choices down to two products. The first was Broadcom's BCM6362 integrated access device. It was a versatile chip with many capabilities unfortunately beyond the needs of this project. In addition Broadcom gave the impression that they were only interested in big cell phone manufacturers or other major developers. Then I came across ZeroG wireless. At the time there was a lot of buzz about Microchip buying them and releasing a new Wi-Fi module for PIC micro-controllers. The module was the ZG2100 and it was pitched as

the low-cost, low-power, and low-requirement Wi-Fi module. Previously, I had worked on PIC micro-controllers and I understood their model of Embedded System products. It supported 8, 16, and 32-bit MCUs and did not require an OS to operate. The low cost of the components to be built around the ZG2100 make it the best choice for this application, see Figure 11.

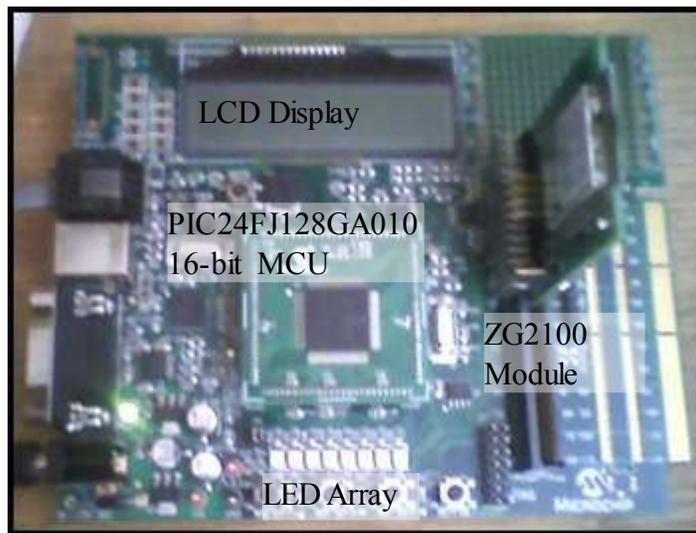


Figure 11: Explorer 16 Development Board

For the website, I sought a cost-effective portable solution. The audience is broad in scope, so portability is crucial. Keeping costs down also works along with reaching a broad audience. My experience with PHP told me that it would make a good choice, because as a scripting language many browsers on a variety of operating systems support it. It is more than capable of handling the data management and tasks required server-side. Once more, it is straight-forward and easy to use as a source of

HTML content.

In order to host the website, I found a free open-source product, WAMP server. (WAMP project, <http://www.wampserver.com/en/index.php>,)WAMP is a common server host program whose name is an acronym for Windows Apache MySQL PHP. Apache is the HTTPd host, and serves both static and dynamic web pages. MySQL is a popular database program, and PHP is a complement of HTML that enables dynamic web pages. FileZilla Server is another open source program free to download and easy to use. (FileZilla Project, <http://filezilla-project.org/>) So, the use of WAMP and FileZilla cover the requirements of the project, lowers cost, and runs on my windows operating system.

My next task was programming the irrigator. The ZG2100 in addition to its own drivers uses the Microchip TCP/IP stack which is included with the irrigator files. The methods from the stack I used (Table I) were located in the files TCP.c, StackTask.c, and Tick.c. TCPIPConfig.h holds information about the connection, such as security type, security keys, Stack Members used, and IP addresses.

Table I: TCP/IP Stack Members Used

Method Name	Method Functionality
StackInit	Initiates the TCP/IP Stack
TickInit	Initiates the synchronous counter
TCPOpen	Opens a connection to a remote site
TCP isConnected	Checks if the connection is working that instant
TCPClose	Closes a connection
TCPIsGetReady	Returns the number of characters in the RX buffer
TCPGet	Receives a character from the RX buffer
TCPGetArray	Receives a string from the RX buffer
TCPPeek	Shows characters in the RX buffer without manipulating them
TCPDiscard	Clears the RX buffer
TCPFindArrayEx	Searches the RX buffer and returns its index
TCPIsGetReady	Checks that the TX buffer is ready to receive data
TCPPutArray	Places a string in the TX buffer for transmission
TCPFlush	Transmits and clears the TX buffer
TickGet	Returns the contents of an synchronous counter

The website starts with an account login page, which is only a place holder and gateway page. It accepts any input as a successful login. The home page shows links forwarding to the control pages: New Controller, Edit Controller, Generate Instructions (Automatic Mode), and Manual Mode.

I then programmed these, working on them one at a time. The PHP scripts require SQL queries for storage, so it was at this stage that I setup WAMP and created the database etscript to hold data. As I went I developed a few functions that worked well to start with on the next. Once I had each of the pages started, development became parallel in nature. In other words, one edit to a page correlated to improvements in the other pages.

After I was satisfied with the web pages as they were, I was able to begin testing. By this time, I also had the irrigator capable of downloading the text files. There were problems with the format of the text outputs, and I found it necessary to make sure that each valve time was the same length of characters. After evaluating edits, coming up with more fixes, implementing them, and then making more edits several times. I had a system I felt satisfactory to demonstrate.

VI. Testing

i. Website

Testing of the website will be done in two parts. The first is to test its accuracy and range of accepted values. For the Automatic Mode, a few csv files from varying months and locations the CIMIS site archives will be downloaded and saved in the server's www folder. The page will be temporarily redirected to this local version, and a browser directed to this page. Results will be checked for accuracy against hand-calculated expected values and for the correct number of leading zeros. Then within the csv files a couple values will be manually changed and results will be checked for the proper difference.

The second part is website usability. The test plan is to have a few people use the web interface to create, edit, and generate irrigation schedules for a new controller from an example landscape, or case study. They will report what appears difficult and what appears easy to use. They will be questioned about their experience in detail and what were their most/least favorite parts.

ii. Irrigator

Using the Manual Page, values of 0,1,2,... seconds will be assigned starting at valve 2 and submitted. The time each LED is on will be measured for accuracy using

a half second LED toggle on LED0.

VII. Results and Analysis

Results from the Automatic Mode when I tested the week of March 11, 2010 were zero seconds for water times, except for only the most extreme of examples. There had been enough precipitation locally to exceed ET_0 for the period, as indicated by the output of the Automatic Mode. The extreme examples output around five minutes, close to calculated values. The means I used to produce the extreme conditions were passing high PR and low IE as parameters. I believe this caused the usual error amounts to seem bigger. Since it has become dryer, more reasonable results have occurred. Using a simple model with an ET very close to reference, I found that the automatic mode is not bug free. One of the problems was results would be watering for the totals that entire week due to having blank values for the previous six days. Another major bug left in the irrigator at the time of this writing was an error in the Irrigator's state machine. It was skipping the first valve after returning from downloading instructions. I believe this was due to not setting any valve number at transition, and instead relying on default state machine values.

Unfortunately, due to lack of time, I was unable to test the system with users; but I imagine they would be frustrated with hand-calculating water times for an irrigation schedule. As ugly as the website is, it's easier that the alternative for

inexpensive water efficiency.

Overall the website performed as expected. The website was not tracking how many times schedules were generated, so it was difficult to track how well it could use past watering times in its comparison to required water amounts. Two outcomes repeated themselves the most. When the automatic mode was used twice, the second value was always zero seconds to water. This is actually expected, because the first values become stored as the previous day and, since ET from CIMIS didn't change, the website calculated that the required amount of water was already given within the last week. The second most common outcome was very large values calculated for watering. Due to not having any stored history of watering, the site would tell the irrigator to water the whole week's worth in that set of timer values.

VIII. Conclusions and Recommendations

At the conclusion of this project, the irrigator showed good functionality. Aside from a few bugs still in the code, I feel it resulted in a good example of what systems like this are capable of. With further development, I see this system being a great, competitive product. Existing irrigation controllers are much more expensive and harder for the customer to use. Often installation requires a professional; this irrigator will help the user learn how to manage their landscapes and encourage them to look at irrigation differently.

The ET Irrigator has a lot of potential for expansion. For example, using remote data from nearby stations might not be a good representation of a user's micro-climate. Sure, the parameter K_{mc} is an estimate of the difference between their landscape and the weather station. By carefully watching for water stress and noting the effects of editing parameters, a good estimate of the discrepancies between micro-climates can be made. Having multiple systems sharing the results of user adjustment allows us to make accurate models of the nearby micro-climates. A great addition would be a soil moisture sensor, then differences will be better quantified and even more accurate models of residential micro-climates will be available.

The website can be even friendlier to users, such as scroll bars to input

parameters instead of text forms. Maybe allow users to choose species factors from a list or gallery of plants, instead of looking it up in a chart. One major improvement would be the ability to change the security settings and SSID for the irrigator. At the moment, these are compile time settings; not very helpful for a user who wishes to make adjustments to their home network. Another useful feature would be to add the ability to make long-term irrigation schedules. This can be done by a user, but even more useful would an automatically generated schedule using archival data from previous years.

Bibliography

1. *WaterSense and Landscape Water Use: What's Next?* (2005). EPA WaterSense.
Retrieved March 14,2010, from
http://www.epa.gov/WaterSense/pubs/whats_next.html
2. Fenyvesi, Charles (1996, October 20) His Whole World is Grass. Usnews.com.
Retrieved March 14, 2010, from
http://www.usnews.com/usnews/culture/articles/961028/archive_034845_2.htm
3. Landscape Water Use Efficiency. (2007). Alliance for Water Efficiency. Retrieved
March 14, 2010, from
[http://www.allianceforwaterefficiency.org/uploadedFiles/Resource_Center/Library/
non_residential/EBMUD/EBMUD_WaterSmart_Guide_Landscape_Water-
Use_Efficiency.pdf](http://www.allianceforwaterefficiency.org/uploadedFiles/Resource_Center/Library/non_residential/EBMUD/EBMUD_WaterSmart_Guide_Landscape_Water-Use_Efficiency.pdf)
4. Richard G. Allen, Ivan A. Walter, Ronald Elliott, Terry Howell, Daniel Itenfisu,
Marvin Jensen. (2005). *The ASCE Standardized Reference Evapotranspiration
Equation*. American Society of Civil Engineers.
5. Frame, Kent. (2008, Fall). *CIMIS Enhancement*. Water Conservation News.
Retrieved March 14,2010, from

- http://www.water.ca.gov/pubs/conservation/water_conservation_news/water_conservation_news_fall_2008/wcn-summer-08.pdf
6. Vickers, Amy. (2001). *Water Use and Conservation*. Amherst, MA: WaterFlow Press.
 7. *Penman Equation*. (2010). Wikipedia. Retrieved March 14, 2010, from http://en.m.wikipedia.org/wiki/Penman_equation
 8. *Penman-Monteith*. (2010). Wikipedia. Retrieved March 14, 2010, from <http://en.m.wikipedia.org/wiki/Penman-Monteith>
 9. Allen, R.G.; Pereira, L.S.; Raes, D.; Smith, M. (1998). *Crop Evapotranspiration—Guidelines for Computing Crop Water Requirements*. Food and Agriculture Organization of the United Nations. Retrieved March 14,2010, from <http://www.fao.org/docrep/X0490E/x0490e00.HTM>.
 10. California Irrigation Management Information System. (2009). Retrieved March 15, 2010, from <http://wwwcimis.water.ca.gov/cimis/welcome.jsp>
 11. Hanson, Blaine, Schwankl, Larry, Fulton, Allan. (1999). *Scheduling Irrigations: When and How Much to Apply*. Davis,CA: University of California Irrigation Program
 12. Dyer, A. S. (2000). A Guide to Estimating Irrigation Water Needs of Landscape Plantings in California. Retrieved March 15, 2010, from http://www.water.ca.gov/pubs/conservation/a_guide_to_estimating_irrigation_water_needs_of_landscape_plantings_in_calif

ornia__wucols/wucols00.pdf

13. Schwankl, Lawrence, Shaw, David, Harivandi, M. Ali, Snyder, Richard L.

Evaluating Turfgrass Sprinkler Irrigation Systems. Division of Agriculture and Natural Resources. Leaflet 21503.

Appendix A: Costs

The components of this project consisted of the PC, Router, Development Board, ZG2100 module, WAMP, and FileZilla Server. The PC I owned before beginning the project and it cost approximately \$400, at the time. The WAMP and FileZilla Server were free online, download-able off their websites. The rest came packaged as the PRO ZG2100 Explorer 16 SDK and cost a total of \$595. In total, the cost of developing this project was \$995

Appendix B: Software Listing

The software is provided on a CD with the submittal of this report. A readme file that describes the files include:

Main.c: the primary file of the irrigator software. It contains the main loop and manages the irrigation functions

FTPClient.c: provides the instructions to act as a client connecting to a remote FTP server

Main.h: the header file for Main.c

FTPClient.h: the header file for FTPClient.c

setup.php: provides the forms to submit parameters for a new controller. Calls generateTables.php at submittal

generateTables.php: processes the forms of setup.php and stores them in a database.

edit.php: edits controller components within the database

manualMode.php: assigns user values for time to water in seconds and generates the instructions in a form of a text file

get.php: automatically downloads data from the CIMIS site and generates instructions in a text file

Appendix C: ANALYSIS OF SENIOR PROJECT DESIGN

Please provide the following information regarding your Senior Project and submit to your advisor along with your final report. Attach additional sheets, for your response to the questions below.

Project Title: Wi-Fi Evapotranspiration Irrigation Controller
--

Student's Name: Ryan Goodman	Student's Signature:
-------------------------------------	-----------------------------

Advisor's Name:	Advisor's Initials:	Date:
------------------------	----------------------------	--------------

Summary of Functional Requirements

The project polls weather data from the CIMIS website and calculates how much time to activate an array of 8 LEDs, representative of irrigator valves. The hardware connects to the website through a Wi-Fi connection and downloads these times and controls the LEDs accordingly.

• Primary Constraints

Describe significant challenges or difficulties associated with your project or implementation. For example, what were limiting factors, or other issues that impacted your approach? What made your project difficult? What

parameters or specifications limited your options or directed your approach?

My only major difficulty was getting reliable FTP connections between hardware and the server. It wasn't a major impact, it only forced a different approach to the problem.

The difficulty was the lack of experience with wireless connections from an embedded system perspective.

- **Economic**

- Original estimated cost of component parts (as of the start of your project).

\$450

- Actual final cost of component parts (at the end of your project)

\$595

- *Attach a final bill of materials for all components.*

Description	Unit price	Qty	Amount
PRO SDK w Explorer 16			
Item #: ZG2100S-06	\$595.00 USD	1	\$595.00 USD
PRO SDK options: PRO SDK 06			

- Additional equipment costs (any equipment needed for development?)

\$400 for a PC to host website

- Original estimated development time (as of the start of your project)

130 hrs

- Actual development time (at the end of your project)

175 hrs

- **If manufactured on a commercial basis:**

- Estimated number of devices to be sold per year

500

- Estimated manufacturing cost for each device

\$80

- Estimated purchase price for each device

\$125

- Estimated profit per year

\$22,500

- Estimated cost for user to operate device, per unit time (specify time interval)

\$20/year

- **Environmental**

- Describe any environmental impact associated with manufacturing or use.

The use of this product will reduce irrigation run off in residential landscaping as much as 100%, saving on water consumption. Industrial manufacturing processes can be harmful in the form of waste materials and excessive use of resources, such as water or electricity.

- **Manufacturability**

- Describe any issues or challenges associated with manufacturing.

The product was designed with manufacturing in mind, it only uses simple PIC micro-controllers and the ZG2100 wireless module. The module itself comes prepackaged with its own drivers and PICs are easily programmed. I don't see any difficulties with the chip packages in terms of manufacturing. There is still the need to layout designs for a PCB board.

- **Sustainability**

- Describe any issues or challenges associated with maintaining the completed device, or system.

It is essential that the device needs to connect to a wireless access point, if one is not available, it is not very usable.

- Describe how the project impacts the sustainable use of resources.

The amount of minerals used in its manufacture is outweighed by the water-efficiency benefits from using the device.

- Describe any upgrades that would improve the design of the project.

A keypad to manually give it timer values in the absence of an internet connection could be added, as well as the ability to program extended irrigation schedules, automatic generation of additional days scheduling through forecasting weather patterns, and remote reprogramming capabilities.

- Describe any issues or challenges associated with upgrading the design.

There is not very much difficulty with most upgrades because the controller is a slave and receives its instructions from a remote server.

- **Ethical**

- Describe ethical implications relating to the design, manufacture, use, or misuse of the project.

There can be serious problems in countries where minerals are extracted to make the product, as well as the possibility of labor issues with its assembly. The project was designed with the security settings for connecting to the Wireless Access Point assigned at compile time, so there can be problems with a homeowner's network being broken into because of having to use standard keys for the products functionality.

- **Health and Safety**

- Describe any health and safety concerns associated with design, manufacture or use of the project.

The product uses radio signals to communicate, which is a form of radiation; however it satisfies FCC standards.

- **Social and Political**

- Describe any social and political concerns associated with design, manufacture or use.

The social and political movement is on a trend of adopting, promoting, and encouraging technologies like this. On the other hand, the spreading of technologies like this is polarizing our society. It's unfortunate, but people can get the notion that looking down on others or even attacking them is justified by the growing popularity

of green technologies. Someone becoming pretentious about how eco-friendly they are is a symptom of the use of fear and exaggerated urgency associated with environmental disasters. More and more, individuals are taking it upon themselves to single-handedly save the world; but, until cooperation becomes a priority, they are just as likely to discourage others from getting them on-board.

• **Development**

• Describe any new tools or techniques, used for either development or analysis that you learned independently during the course of your project.

- MPLAB IDE for PIC MCUs
- TCP/IP stack API
- MySQL databases
- Setting up and maintaining a web server
- File Transfer Protocol
- Hyper-text Transfer Protocol
- TCP/IP layers
- Wireless internet connectivity standards