

San Francisco State University

School of Engineering



# Plant Monitoring System

FINAL PROJECT

Xochitl Menjivar | ENGR 478-01 | December 20, 2020

## Table of Contents

Introduction .....	2
Project Goals.....	2
System Requirements .....	3
Critical Aspects .....	4
Design and Implementation.....	5
Hardware architecture .....	5
System Architecture .....	6
Hardware Implementation .....	7
Microcontroller .....	7
Soil Moisture sensor .....	8
ICStation Resistive Soil Moisture Sensor .....	8
Temperature Sensor .....	9
MCP9700A-E/TO with Linear Active Thermistor IC.....	9
Light Sensor.....	14
Software Algorithms and Flow Charts .....	15
Main Function .....	15
GPIO Interrupt .....	16
ADC Interrupts .....	16
Module Implementation.....	17
Experiment & Results.....	19
Future Improvements .....	21
Datasheets .....	22

## Introduction

This project is inspired by a hobby I started taking up over the summer: gardening. I had bought a small pepper plant from Home Depot and planted it outside in my yard. It took a while (a few months), but eventually the plant started growing peppers.

When it came time to think about what projects we wanted to do for this class, I decided that I wanted to make something practical that could be useful in my day-to-day life.

Sometimes it's hard to tell for instance if the soil is moist enough without sticking your fingers in the soil to test or if the plant is consistently getting the right amount of sun while also staying in the right temperature. So, because of this I decided to build a small plant monitoring system that would help me meet the plant's needs better and thus, become a better gardener.



## PROJECT GOALS

The goal of this project is to create a small-scale plant monitoring system that will measure values such as the temperature, soil moisture, and the amount of sun the plant is getting.

The project will also meet the requirements for this class by implementing a real-time system, using software interrupts, and interfacing external sensors to the microcontroller.

## System Requirements

The plant monitoring system will take a few measurements:

- Temperature
  - Units: °C

The temperature of the surrounding environment will be taken in real-time and displayed to Tera Term in degrees Celsius.

- Soil moisture
  - Status




The moisture of the soil will be measured in real-time and the measurements taken will be displayed to Tera Term. The measurement will be given as a status indicating how wet or dry the soil is.

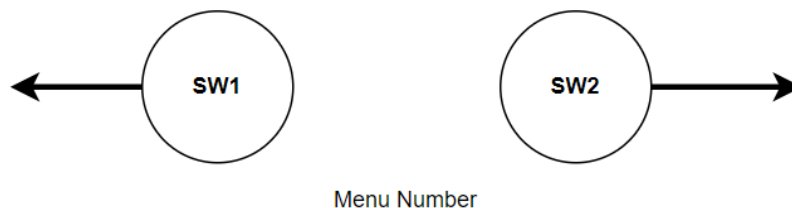
- Sunlight
  - Status

The amount of sunlight the plant is getting will be measured and real-time and displayed on the Tera Term. The measurement will be given as a status indicating how much sunlight the sensor is exposed to.

### User Interface

The system will also make use of the on-board buttons and LEDs on the TM4C by letting the user switch between measurements. When SW1 is pressed, the options rotate from right to left. When SW2 is pressed, the options rotate from left to right.

Default Prompt	Soil Moisture	Temperature	Sunlight
LEDs off	Green 	 Blue	 Red



## Critical Aspects

### **Reliability of the sensors**

The reliability of each sensor should be considered when building this project since some sensors will be better quality than others. Since this is a project for a class, I try to keep costs down when considering what components to buy and implement into my system. The sensors may work well enough for my project but since they are relatively inexpensive, they may also not be completely accurate.

### **Correct interpretation of the data**

The interpretation of the data is also very important for this project. Although I am not trying to create an extremely accurate system, I at least want the data to be approximately correct. The temperature data requires the most accuracy out of all the components used for this project and so I need to apply a couple of equations to try and get the correct reading of the temperature.

### **Software design**

The software design is also crucial for the program to work correctly otherwise the MCU could get overwhelmed. Good OOP design could be used so that there are several methods with their own special functions that execute only when they need to be. Methods with several different functions will become very bloated very quickly.

# Design and Implementation

## HARDWARE ARCHITECTURE

### Hardware Architecture

#### External Sensors:

161 CDS Photoresistor

MCP9700A

Soil Moisture Sensor

#### External Controls:

SW1

SW2

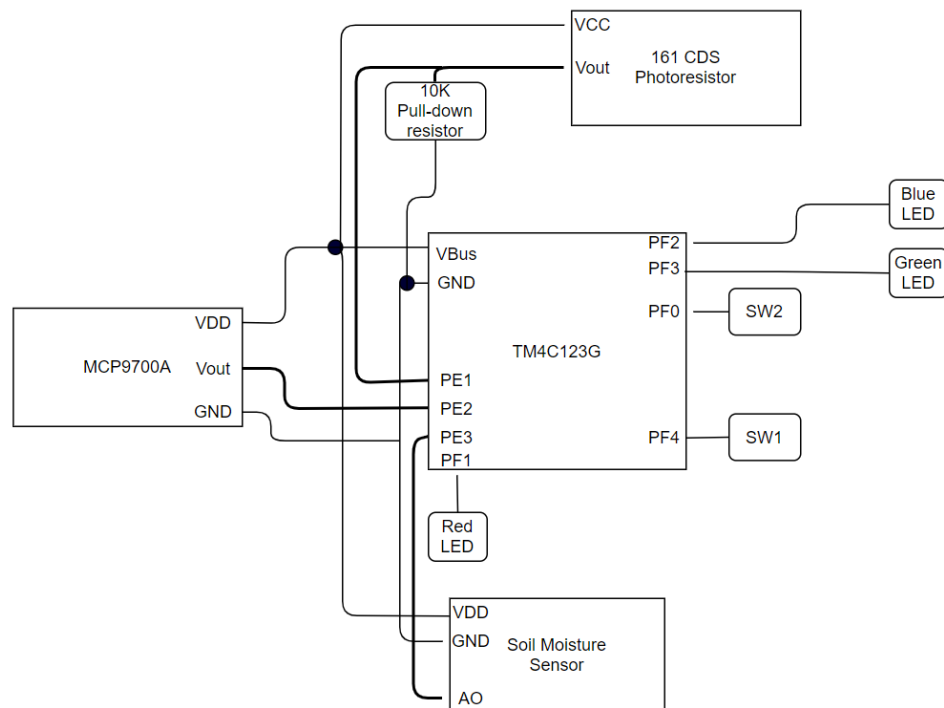
#### Other Components:

10k Resistor

Red LED

Blue LED

Green LED



## SYSTEM ARCHITECTURE

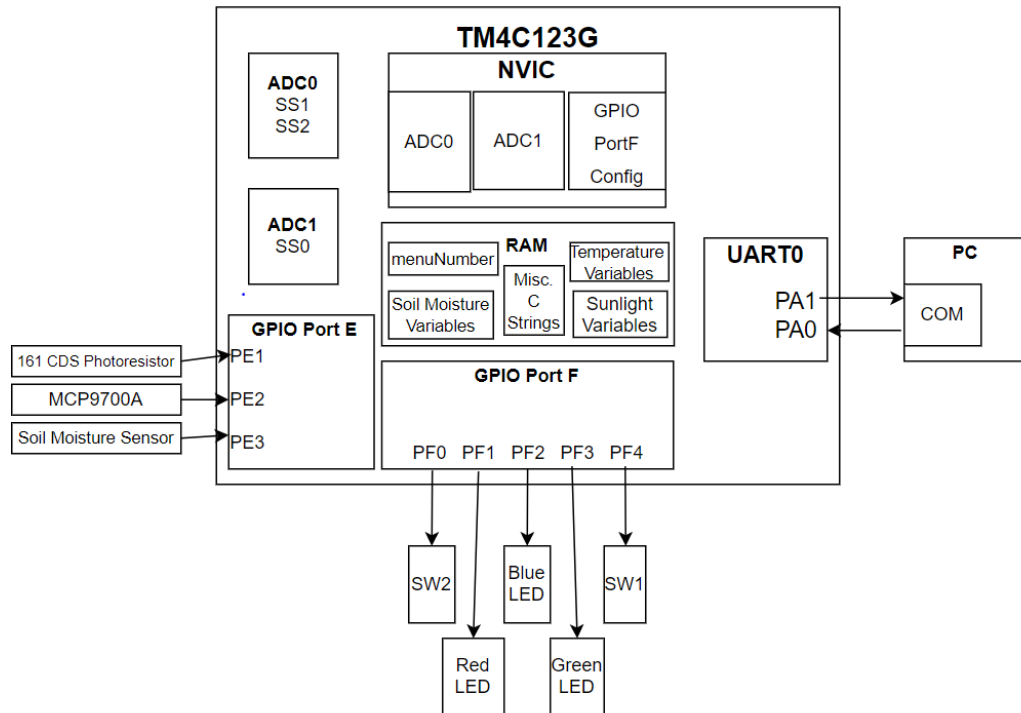
### System Architecture

GPIO: Port E, Port F

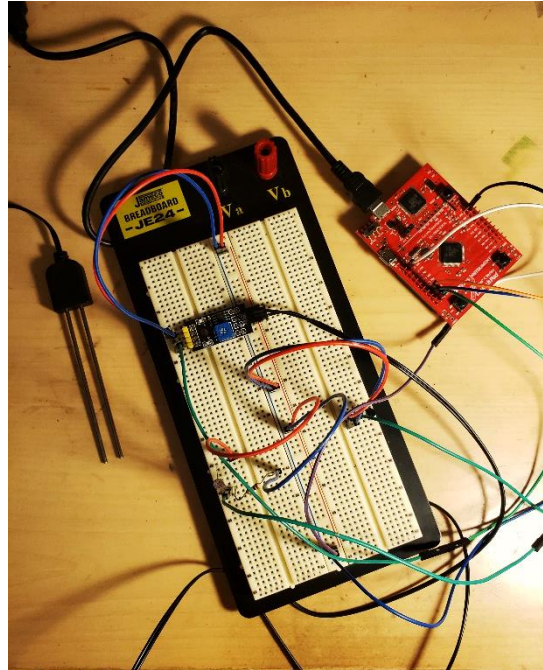
UART0

ADC0: SS1, SS2

ADC1: SS0

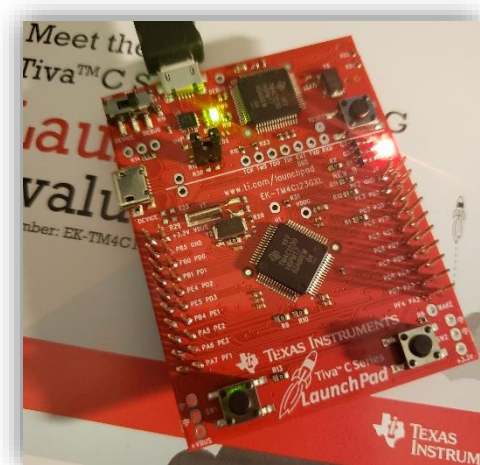


## Hardware Implementation



### MICROCONTROLLER

TI Tiva C Series TM4C123G LaunchPad Evaluation Kit (EK-TM4C123GXL)



The TM4C microcontroller is the required component for this course and will be used to connect all of the external sensors as well as process the data received from them.



## SOIL MOISTURE SENSOR

### ICStation Resistive Soil Moisture Sensor



#### Parameters

Working Voltage	DC 3.3-12V
Working Current	<20mA (Output Current: <30mA)
Board Size	36x15x6mm/1.42X0.59X0.23 inch (L*W*H)
Sensor Probe Length	8.8cm/3.46 inch
Sensor Pin Pitch	5 mm
Sensor Cable Length	1.2 meters /3.94ft
Sensor Connecting Port	-XH2.54 2P
DO Port	digital value output (DO output current is about 10mA)
AO Port	analog value output

The soil moisture sensor comes with a probe that uses analog resistance to detect the amount of moisture present in the soil. The moister the soil is, the less resistance there will be in the sensor.

The sensor comes with an analog output pin that enables me to connect it to an ADC pin on the MCU. The ADC converts the voltage reading into a least significant bit (LSB) value which I can then use to determine the status of the moisture anytime the program takes a reading.

## TEMPERATURE SENSOR

### MCP9700A-E/TO with Linear Active Thermistor IC



The MCP9700A is an analog temperature sensor that measures the surrounding ambient temperature and converts this value into a voltage. It is very straightforward to use and can be directly connected to an ADC pin on the microcontroller.

#### *Specifications*

- Wide Temperature Measurement Range:
  - $-40^{\circ}\text{C}$  to  $+125^{\circ}\text{C}$  (Extended Temperature)
  - $-40^{\circ}\text{C}$  to  $+150^{\circ}\text{C}$  (High Temperature)**(MCP9700, SOT-23-3 and SC70-5 only)**
- Accuracy:
  - $\pm 2^{\circ}\text{C}$  (max.),  $0^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$  **(MCP9700A/9701A)**
  - $\pm 4^{\circ}\text{C}$  (max.),  $0^{\circ}\text{C}$  to  $+70^{\circ}\text{C}$  **(MCP9700/9701)**
- Optimized for Analog-to-Digital Converters (ADCs):
  - $10.0\text{ mV}/^{\circ}\text{C}$  (typical) **(MCP9700/9700A)**
  - $19.5\text{ mV}/^{\circ}\text{C}$  (typical) **(MCP9701/9701A)**
- Wide Operating Voltage Range:
  - $V_{\text{DD}} = 2.3\text{V}$  to  $5.5\text{V}$  **(MCP9700/9700A)**
  - $V_{\text{DD}} = 3.1\text{V}$  to  $5.5\text{V}$  **(MCP9701/9701A)**

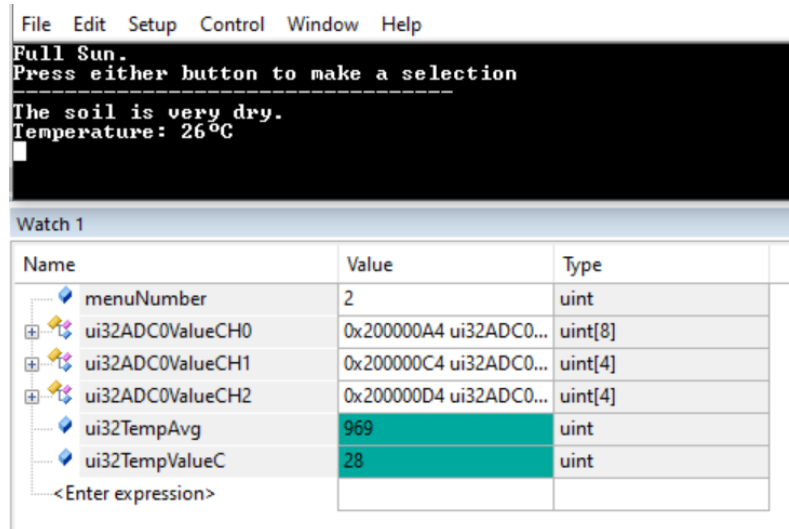
When the voltage is read, the ADC of the microcontroller will convert this value into a least significant bit (LSB) digital value (ranging from 0-4095). For the purpose of this project, I will need to convert the value of the LSB to an actual temperature reading.

To start, I first need to convert the LSB to the ADC output voltage reading. To calculate this value, I use Equation 1 for PTC thermistors.

### Equation 1 ADC voltage reading

$$ADC \text{ Voltage} = \left( \frac{ADC \text{ VREF}}{2^{ADC \text{ Resolution}}} \right) \times \text{Measured ADC LSB value}$$

The average voltage reference for a PTC thermistor is 3.3V with the ADC resolution being 4096 bits. To get the measured LSB value, I run the program I created for the project and get the average value received from the ADC sampling (ui32TempAvg = 969 bits).



Once I have all the necessary values, I can solve the equation to find the ADC voltage.

$$ADC \text{ Voltage} = \left( \frac{ADC \text{ VRef}}{2^{ADC \text{ Resolution}}} \right) \times \text{Measured ADC LSB value}$$

$$ADC \text{ Voltage} = ?$$

$$ADC \text{ VRef} = 3.3V$$

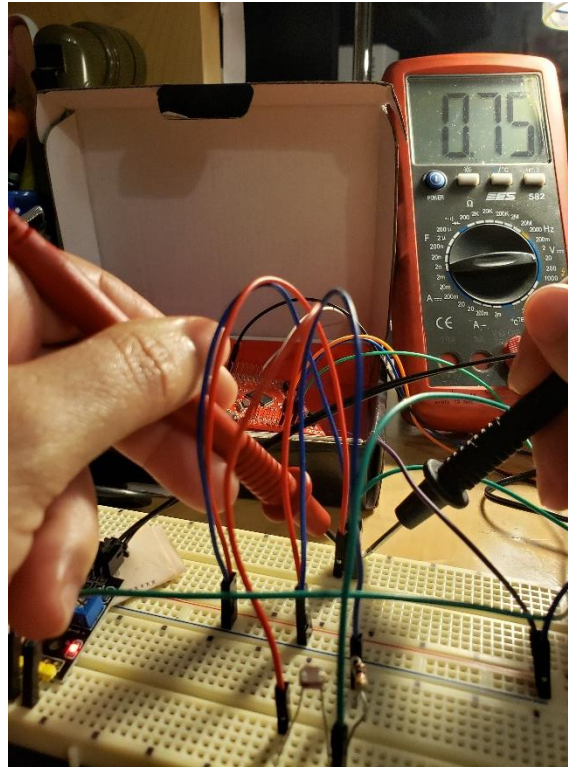
$$2^{ADC \text{ Resolution}} = 4096 \text{ bits}$$

$$\text{Measured ADC LSB value} = 969 \text{ bits}$$

$$ADC \text{ Voltage} = \left( \frac{3.3V}{4096 \text{ bits}} \right) \times 969 \text{ bits}$$

$$ADC \text{ Voltage} = 0.78 V$$

I end up getting ADC Voltage = 0.78V. I cross-check this calculation by doing a voltage reading on the temperature sensor while it is being powered by the circuit.



*1 Voltage reading to verify calculation*

The voltage reading seems to verify the correctness of the above calculation, so I move on to solve for the ambient temperature measured by the sensor.

**EQUATION 4-1: SENSOR TRANSFER FUNCTION**

$$V_{OUT} = T_C \times T_A + V_{0^{\circ}C}$$

Where:

$T_A$  = Ambient Temperature

$V_{OUT}$  = Sensor Output Voltage

$V_{0^{\circ}C}$  = Sensor Output Voltage at 0°C  
(see [DC Electrical Characteristics](#) table)

$T_C$  = Temperature Coefficient  
(see [DC Electrical Characteristics](#) table)

The sensor transfer function is provided by the datasheet that comes with this temperature sensor. I use the calculated value of the ADC voltage for the variable  $V_{OUT}$  and

refer to the DC Electrical Characteristics table also provided in the datasheet to get the constants.

<b>Electrical Specifications:</b> Unless otherwise indicated: <b>MCP9700/9700A:</b> $V_{DD} = 2.3V$ to $5.5V$ , $GND = \text{Ground}$ , $T_A = -40^{\circ}C$ to $+125^{\circ}C$ and No load <b>MCP9701/9701A:</b> $V_{DD} = 3.1V$ to $5.5V$ , $GND = \text{Ground}$ , $T_A = -10^{\circ}C$ to $+125^{\circ}C$ and No load						
Parameter	Sym.	Min.	Typ.	Max.	Unit	Conditions
<b>Power Supply</b>						
Operating Voltage Range	$V_{DD}$ $V_{DD}$	2.3 3.1	—	5.5 5.5	V V	MCP9700/9700A MCP9701/9701A
Operating Current	$I_{DD}$	—	6	12	$\mu A$	
	$I_{DD}$	—	—	15	$\mu A$	$T_A = 150^{\circ}C$ (Note 1)
Line Regulation	$\Delta^{\circ}C/\Delta V_{DD}$	—	0.1	—	$^{\circ}C/V$	
<b>Sensor Accuracy (Notes 2, 3)</b>						
$T_A = +25^{\circ}C$	$T_{ACY}$	—	$\pm 1$	—	$^{\circ}C$	
$T_A = 0^{\circ}C$ to $+70^{\circ}C$	$T_{ACY}$	-2.0	$\pm 1$	+2.0	$^{\circ}C$	MCP9700A/9701A
$T_A = -40^{\circ}C$ to $+125^{\circ}C$	$T_{ACY}$	-2.0	$\pm 1$	+4.0	$^{\circ}C$	MCP9700A
$T_A = -10^{\circ}C$ to $+125^{\circ}C$	$T_{ACY}$	-2.0	$\pm 1$	+4.0	$^{\circ}C$	MCP9701A
$T_A = 0^{\circ}C$ to $+70^{\circ}C$	$T_{ACY}$	-4.0	$\pm 2$	+4.0	$^{\circ}C$	MCP9700/9701
$T_A = -40^{\circ}C$ to $+125^{\circ}C$	$T_{ACY}$	-4.0	$\pm 2$	+6.0	$^{\circ}C$	MCP9700
$T_A = -10^{\circ}C$ to $+125^{\circ}C$	$T_{ACY}$	-4.0	$\pm 2$	+6.0	$^{\circ}C$	MCP9701
$T_A = -40^{\circ}C$ to $+150^{\circ}C$	$T_{ACY}$	-4.0	$\pm 2$	+6.0	$^{\circ}C$	High Temperature (Note 1)
<b>Sensor Output</b>						
Output Voltage, $T_A = 0^{\circ}C$	$V_{0^{\circ}C}$	—	500	—	mV	MCP9700/9700A
Output Voltage, $T_A = 0^{\circ}C$	$V_{0^{\circ}C}$	—	400	—	mV	MCP9701/9701A
Temperature Coefficient	$T_C$	—	10.0	—	mV/ $^{\circ}C$	MCP9700/9700A
	$T_C$	—	19.5	—	mV/ $^{\circ}C$	MCP9701/9701A
Output Nonlinearity	$V_{ONL}$	—	$\pm 0.5$	—	$^{\circ}C$	$T_A = 0^{\circ}C$ to $+70^{\circ}C$ (Note 3)

With all the necessary values, I can now solve for the ambient temperature ( $T_A$ ).

$$V_{out} = T_C \times T_A + V_{0^{\circ}C}$$

$$V_{out} = 0.78V$$

$$T_C = 10.0 \text{ mV}/^{\circ}C$$

$$T_A = ?$$

$$V_{0^{\circ}C} = 500mV$$

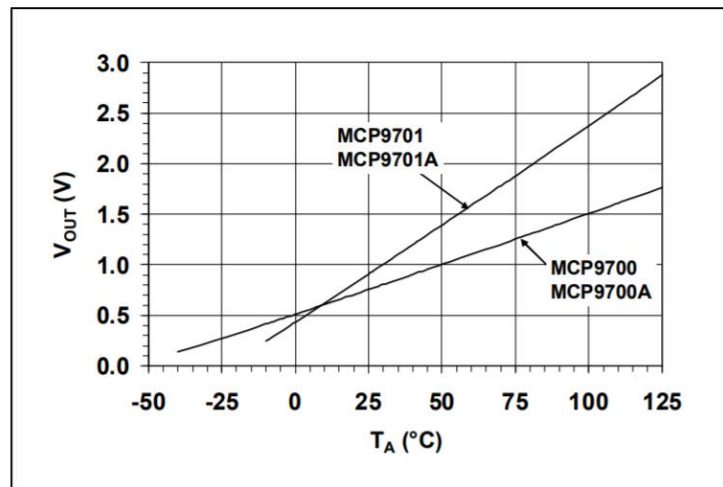
$$0.78V = 10.0 \frac{mV}{^{\circ}C} \times T_A + 500mV$$

$$T_A = 28.07^{\circ}C$$

The value calculated for  $T_A$  came out to be about 28 degrees Celsius.

I can cross-check the value of  $V_{Out}$  and  $T_A$  by using the graph below. This is also provided in the sensor's datasheet and shows the linear relationship between the variables. This graph lets you visually see what the temperature will be depending on the output voltage of the ADC.

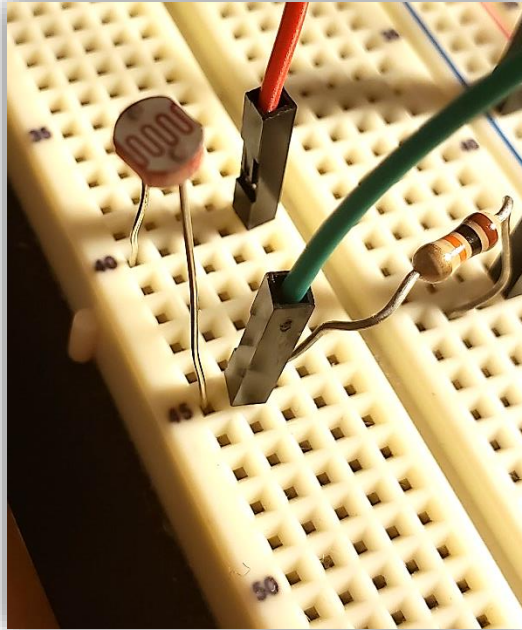
By looking at this graph, I can see that the values I calculated are proportional to each other. I can now integrate this process into code for the program to run.



**FIGURE 2-16:** *Output Voltage vs. Ambient Temperature.*

## LIGHT SENSOR

Photocell (161 CDS PHOTORESISTOR)

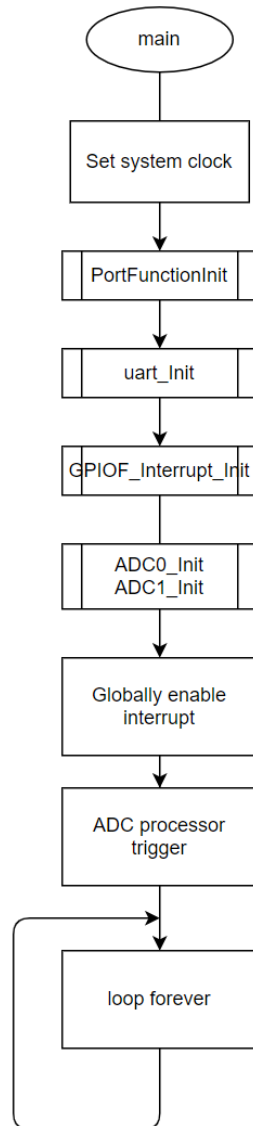


This photoresistor is a very simple device that takes the input voltage and gives back a voltage reading that depends on how much resistance it has. When the sensor is exposed to light, the resistance goes down to about 5-10k $\Omega$  while the resistance can go up to 200k $\Omega$  when it is completely dark.

One side of the sensor is connected to  $V_{CC}$  and the other is connected to a 10k $\Omega$  pull-down resistor which is then connected to ground. The output pin is also connected to an ADC pin on the microcontroller.

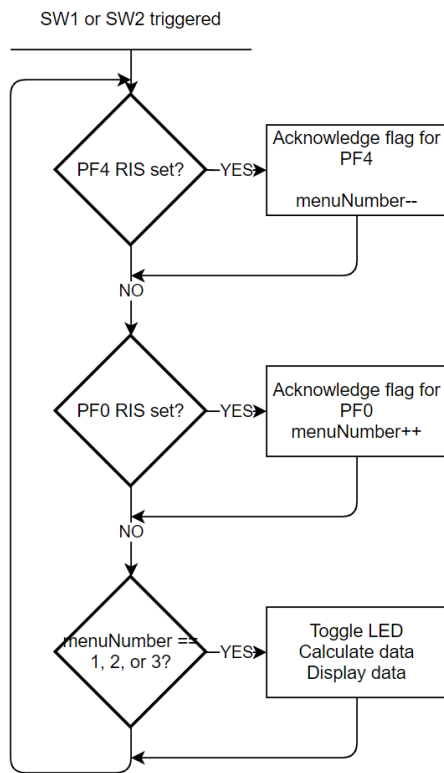
# Software Algorithms and Flow Charts

## Main Function

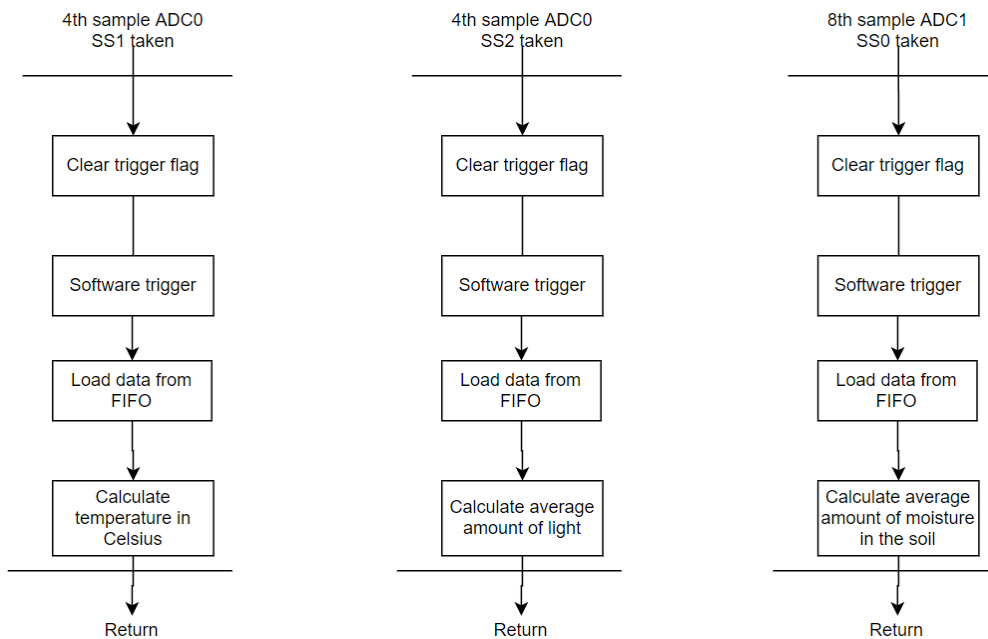




## GPIO Interrupt



## ADC Interrupts



## Module Implementation

Below is the implementation of each module used in this project.

### *Port Function Initialization*

- Enable Peripheral Clocks
  - MAP\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC0); //ADC0
  - MAP\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_ADC1); //ADC1
  - MAP\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOE); //GPIO Port E
  - MAP\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF); //GPIO Port F
- Enable Port E pins for input (used for sensors)
  - MAP\_GPIOPinTypeADC(GPIO\_PORTE\_BASE, GPIO\_PIN\_1); //PE1 (for ADC Ain2)
  - MAP\_GPIOPinTypeADC(GPIO\_PORTE\_BASE, GPIO\_PIN\_2); //PE2 (for ADC Ain1)
  - MAP\_GPIOPinTypeADC(GPIO\_PORTE\_BASE, GPIO\_PIN\_3); //PE3 (for ADC Aino)
- Enable Port F pins for input (SW1 & SW2)
  - MAP\_GPIOPinTypeGPIOInput(GPIO\_PORTF\_BASE, GPIO\_PIN\_0); //PF0
  - MAP\_GPIOPinTypeGPIOInput(GPIO\_PORTF\_BASE, GPIO\_PIN\_4); //PF4
- Enable Port F pins for output (Red, Green, Blue LEDs)
  - MAP\_GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_1); //PF1
  - MAP\_GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_2); //PF2
  - MAP\_GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_3); //PF3

### *GPIO Port F Interrupt Initialization*

- Enable Interrupt 30 in NVIC
  - IntEnable(INT\_GPIOF);
- Set priority to 0
  - IntPrioritySet(INT\_GPIOF, 0x00);
- PF0 and PF4 are edge-sensitive
  - GPIO\_PORTF\_IS\_R &= ~0x11;
- PF0 and PF4 not both edges trigger
  - GPIO\_PORTF\_IBE\_R &= ~0x11;
- PF0 and PF4 falling edge event
  - GPIO\_PORTF\_IEV\_R &= ~0x11;

### *UART Initialization*

- Enable Peripheral Clocks
  - SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART0);
  - SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOA);

- Set Baud rate
  - `UARTConfigSetExpClk(UARTo_BASE, SysCtlClockGet(), 115200, (UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE | UART_CONFIG_PAR_NONE));`

### *ADCo Initialization*

- Enable Peripheral clock
  - `SysCtlPeripheralEnable(SYSCTL_PERIPH_ADCo);`
- Configure to use ADCo, SS1, processor-trigger, priority 1
  - `ADCSequenceConfigure(ADCo_BASE, 1, ADC_TRIGGER_PROCESSOR, 0);`
- Configure to use ADCo, SS2, processor-trigger, priority 2
  - `ADCSequenceConfigure(ADCo_BASE, 2, ADC_TRIGGER_PROCESSOR, 1);`
- Configure ADCo SS1 interrupt priority as 2
  - `IntPrioritySet(INT_ADCoSS1, 0x02);`
- Enable interrupt 31 in NVIC (ADCo SS1)
  - `IntEnable(INT_ADCoSS1);`
- Configure ADCo SS2 interrupt priority as 2
  - `IntPrioritySet(INT_ADCoSS2, 0x02);`
- Enable interrupt 32 in NVIC (ADCo SS2)
  - `IntEnable(INT_ADCoSS2);`

### *ADC1 initialization*

- Enable Peripheral clock
  - `SysCtlPeripheralEnable(SYSCTL_PERIPH_ADC1);`
- Configure to use ADC1, SSo, processor-trigger, priority 0
  - `ADCSequenceConfigure(ADC1_BASE, 0, ADC_TRIGGER_PROCESSOR, 0);`
- Configure ADC1 SSo interrupt priority as 0
  - `IntPrioritySet(INT_ADC1SSo, 0x00);`
- Enable interrupt 64 in NVIC (ADC1 SSo)
  - `IntEnable(INT_ADC1SSo);`

### *Main Function*

- Configure the system clock to be 40MHz
  - `SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_OSC_MAIN|SYSCTL_XTAL_16MHZ);`
- Globally enable interrupt
  - `IntMasterEnable();`
- Enable ADC processor triggers
  - `ADCProcessorTrigger(ADC1_BASE, 0);`
  - `ADCProcessorTrigger(ADCo_BASE, 1);`
  - `ADCProcessorTrigger(ADCo_BASE, 2);`

## Experiment & Results

Experimentation mostly just involved testing the system on actual plants or soil samples. The first experiment I did was test the system on a group of soil samples. I first placed gardening soil in three different cups. The first cup has no water in it and is completely dry. The second cup has some water, but it is not completely muddy. The third cup has a lot of water and can be considered muddy.



Before I start, I check the general temperature inside the house (65°F, 18.3°C). Then I start by running the program and inserting the soil sensor probe into the first cup that has no water in it.

```
The soil is very dry.  
Temperature: 10°C  
Mostly Sunny.  
Press either button to make a selection  
=====
```

Although the soil sensor knows the soil is completely dry, the temperature read from the temperature sensor is different from the temperature measured by the house thermometer.

I then repeat the process but in the second soil sample where there is some water in the cup.

```
-----  
The soil is somewhat wet.  
Temperature: 5°C  
Mostly Sunny.  
Press either button to make a selection  
-----
```

The soil sensor now picks up the moisture present in this sample. The temperature however still seems off and is reading a temperature colder than it actually is.

The soil sensor is then placed in the third sample with the most water inside.

```
-----  
The soil is somewhat wet.  
Temperature: 6°C  
Mostly Sunny.  
Press either button to make a selection  
-----
```

Unfortunately, the sensor isn't reading that there is noticeably more water in this sample and the temperature is still giving low readings for the ambient temperature.

Another trial run I did was outdoors in the yard where my hot peppers are planted. I repeat the process as before with the temperature being around 54.5°F or 12.5°C.

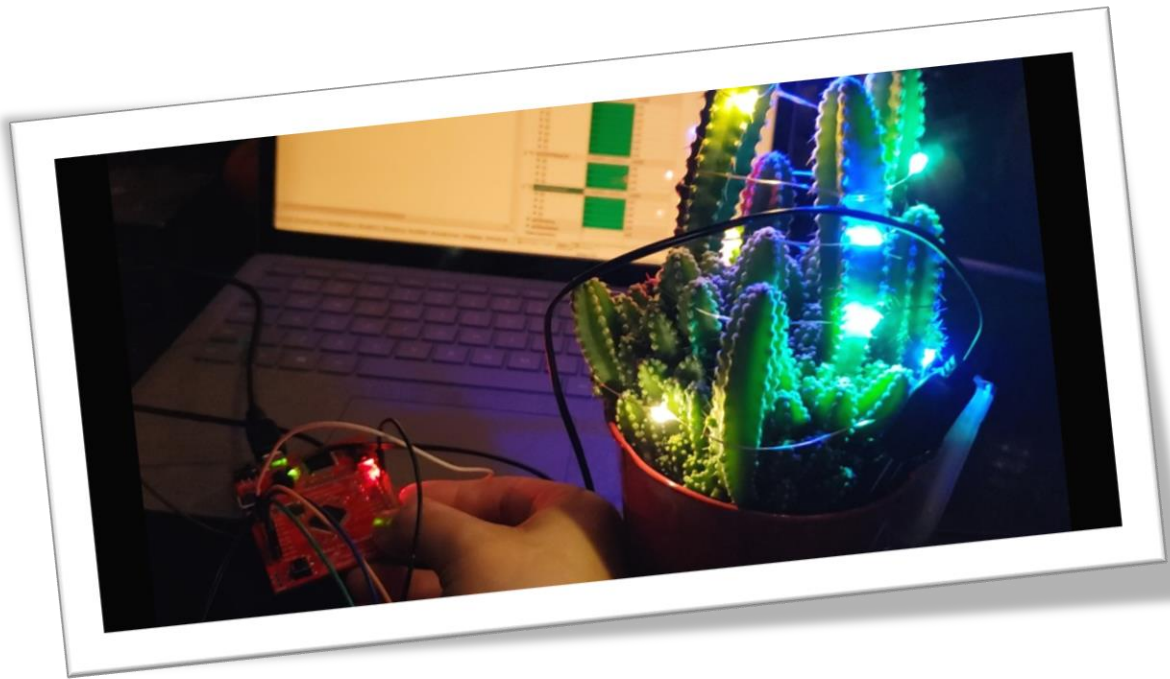


The results I got from this test is shown in Tera Term.

```
COM3 - Tera Term VT  
File Edit Setup Control Window Help  
The soil is somewhat dry.  
Temperature: 6°C  
Mostly Sunny.  
Press either button to make a selection  
-----
```

## Future Improvements

- LCD Screen
  - An LCD Screen could be added to display the data from the sensors so that I wouldn't have to rely on terminal on the PC.
- Humidity Sensor
  - A humidity sensor could be added to the system to detect the moisture levels of the air surrounding the plant.
- Improve data interpretation
  - Configure the code so that the program outputs more accurate data.



## Datasheets

TM4C123GH6PM MCU

[https://www.ti.com/lit/ds/symlink/tm4c123gh6pm.pdf?ts=1608210555841&ref\\_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTM4C123GH6PM](https://www.ti.com/lit/ds/symlink/tm4c123gh6pm.pdf?ts=1608210555841&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTM4C123GH6PM)

ICStation Resistive Soil Moisture Sensor with Corrosion Resistant Probe

<https://images-na.ssl-images-amazon.com/images/I/81SaGn76xgL.pdf>

MCP9700A-E/TO Temperature Sensor

<http://www.farnell.com/datasheets/2243473.pdf>

161 CDS Photoresistor

<http://www.farnell.com/datasheets/1815595.pdf>