Plant Watering and Humidifier System

Guillem Cucurull and Jonathan Trieloff

BOSTON UNIVERSITY

Abstract

The goal is to utilize an arduino and sensors to:

- Maintain proper soil moisture
- Inform user of the state of the system, such as moisture level, humidity, etc.

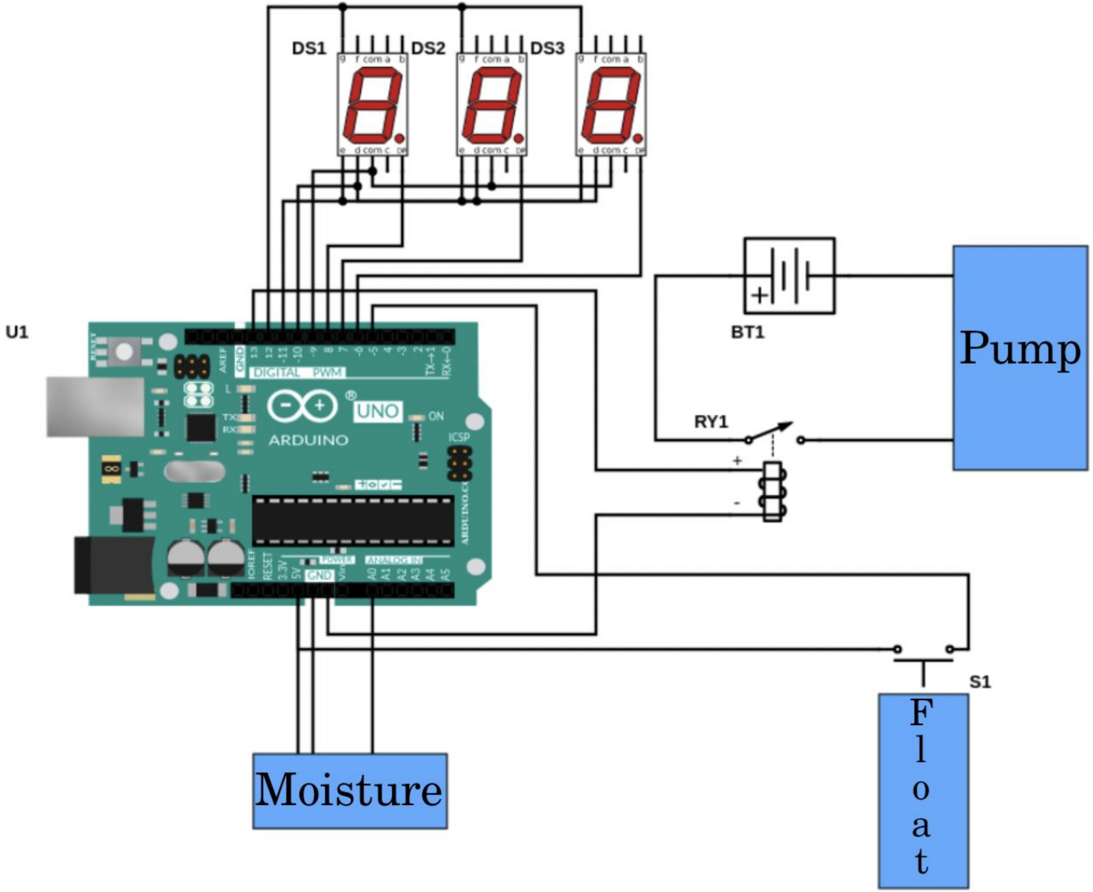


Figure 1: Schematic Diagram

Parts

Collected from eLab:

Arduino UNO

Breadboard

Soil moisture sensor (tested!)

- 5V and GRD

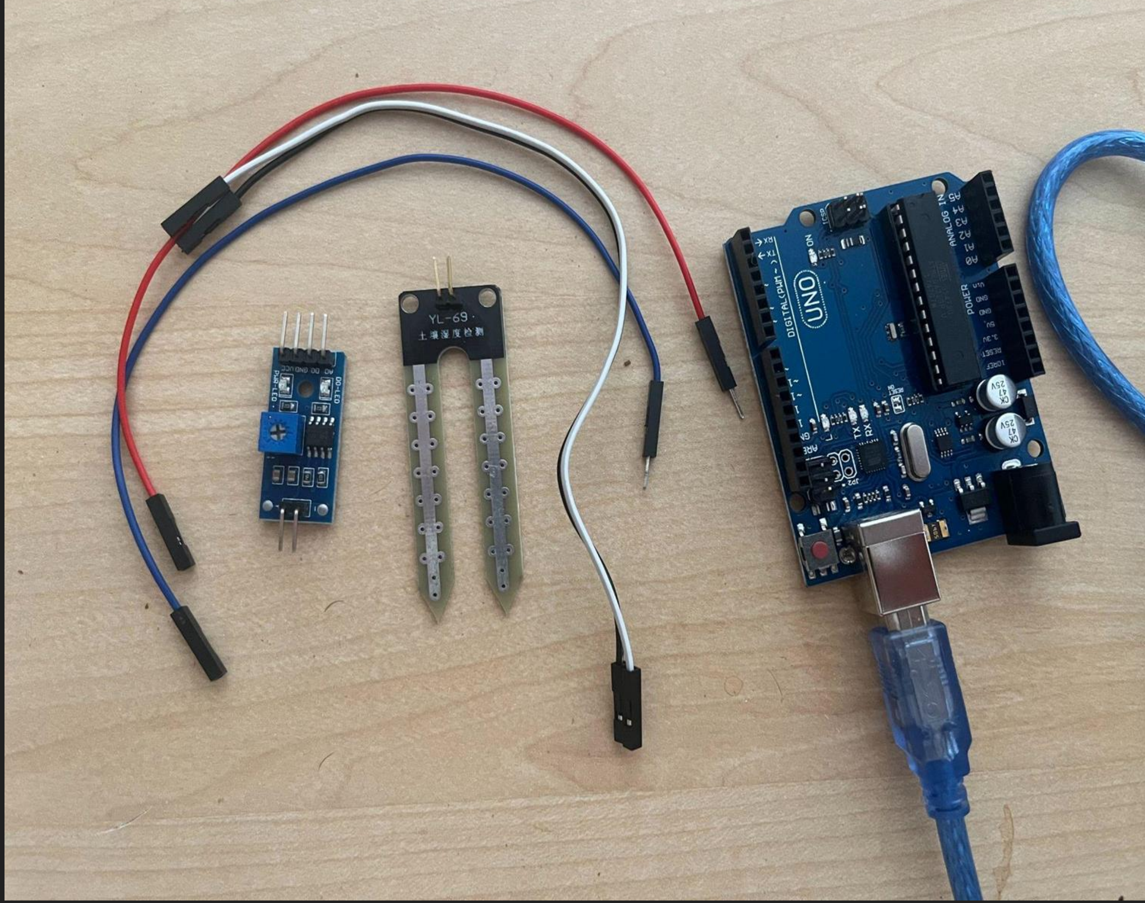
Relay module

- Outputs: 12V and GRD
- Pins: (-) to GRD, (+) to 5V, switch
- 10 Amps
- 30 V DC

Water pump

- 12 V

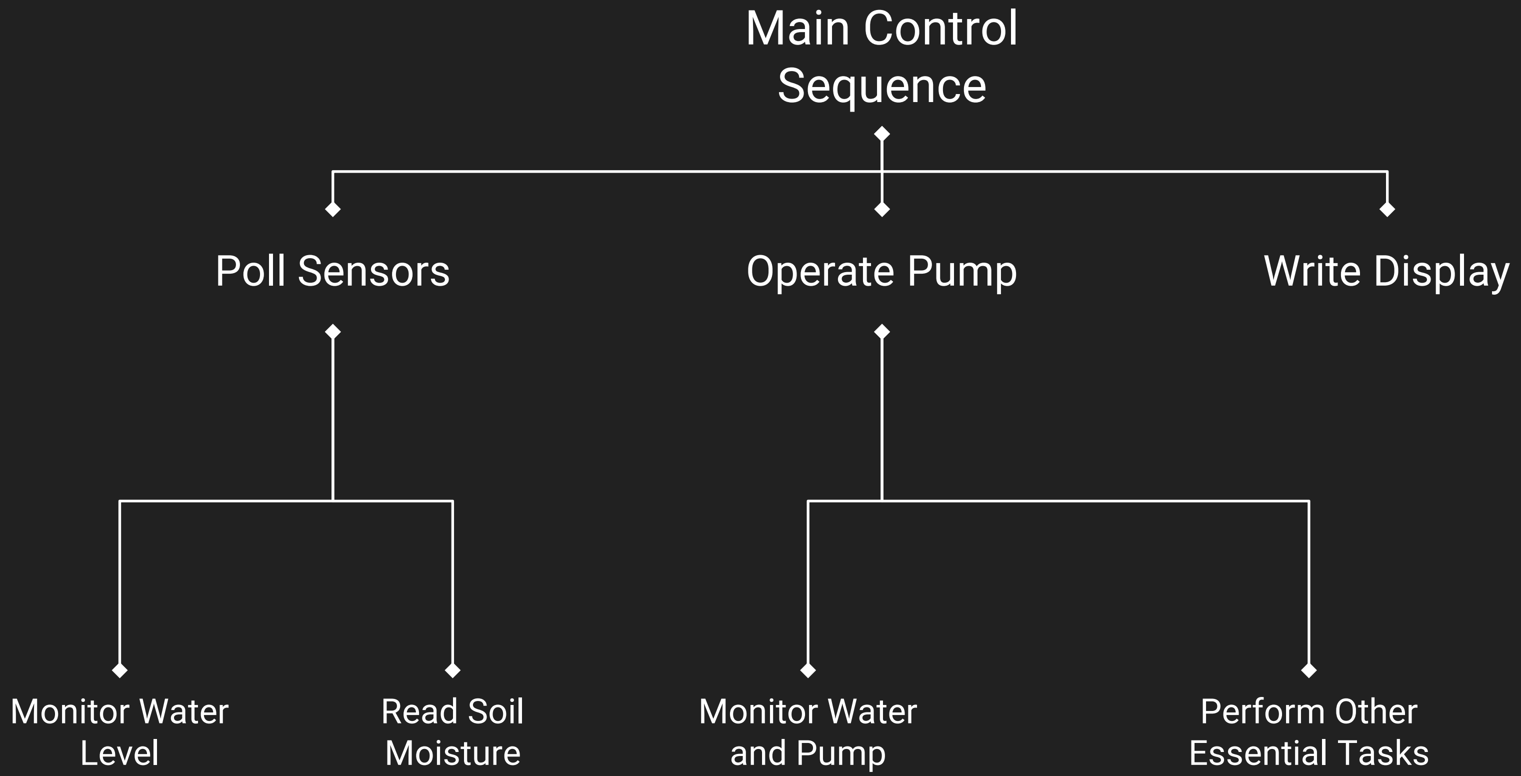
Plastic water bottle



Yet to be collected but present in eLab:

Power supply

Plastic pipe



Main Control Sequence

Poll Sensors

Operate Pump

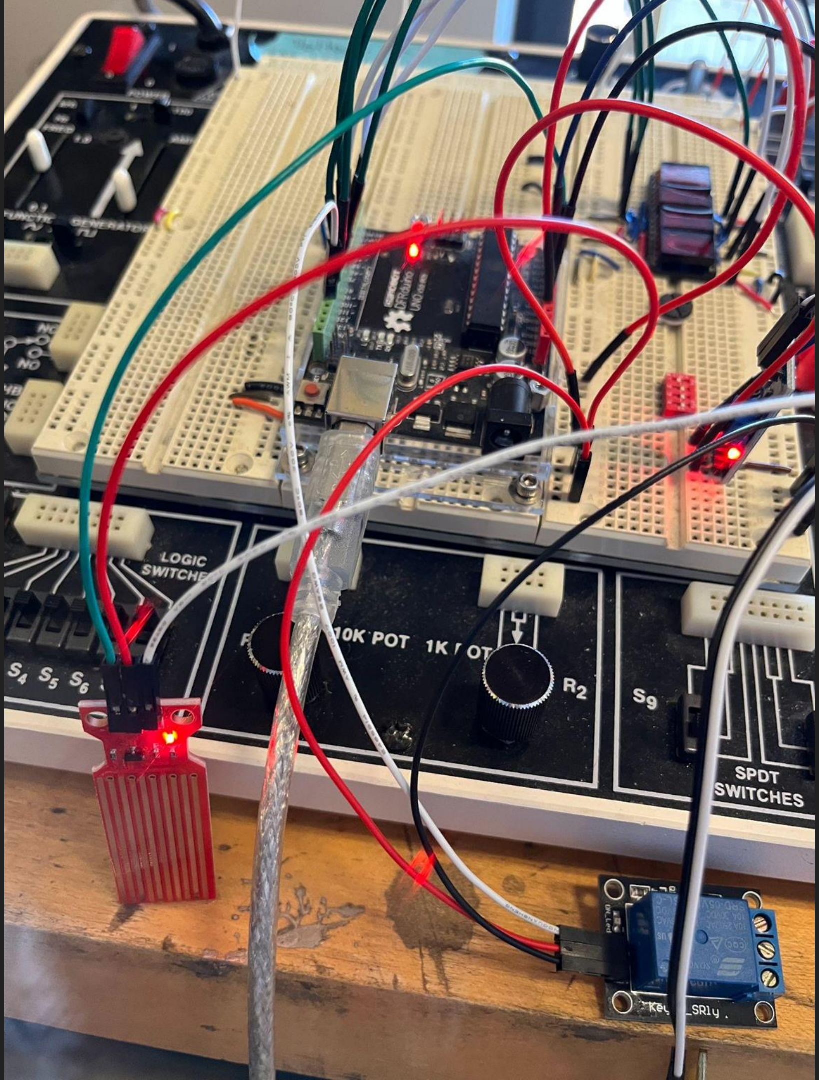
Write Display

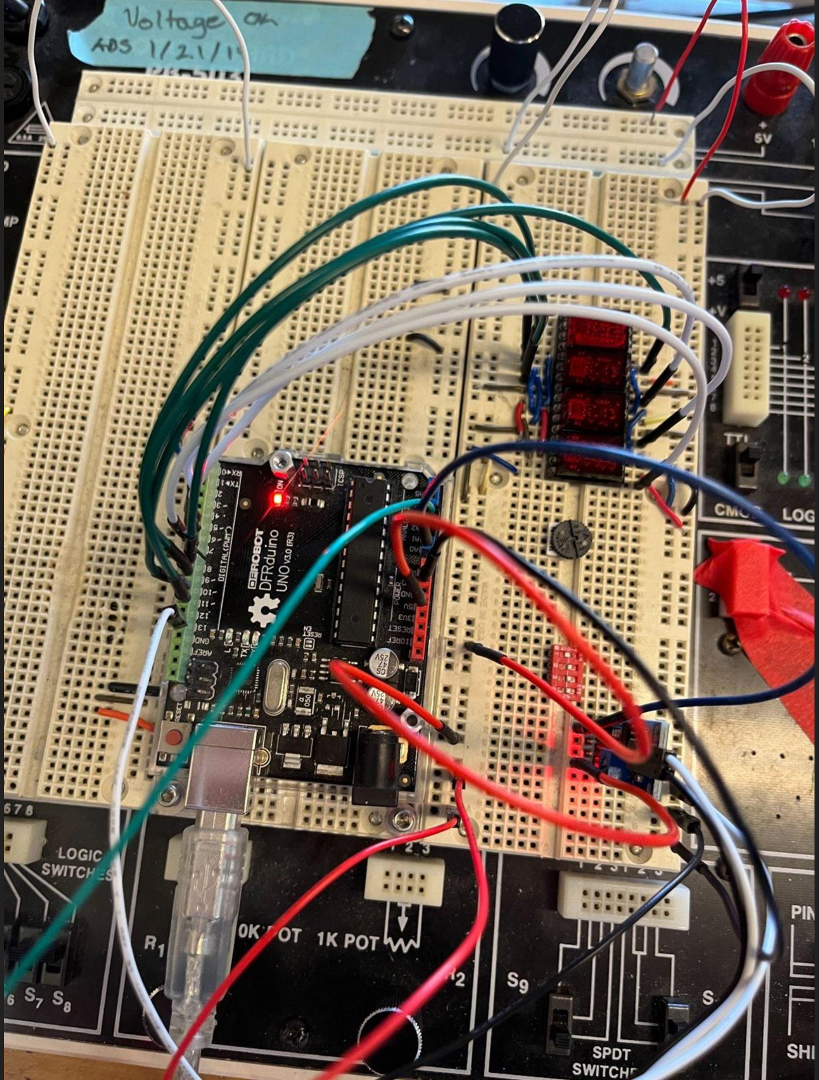
Monitor Water Level

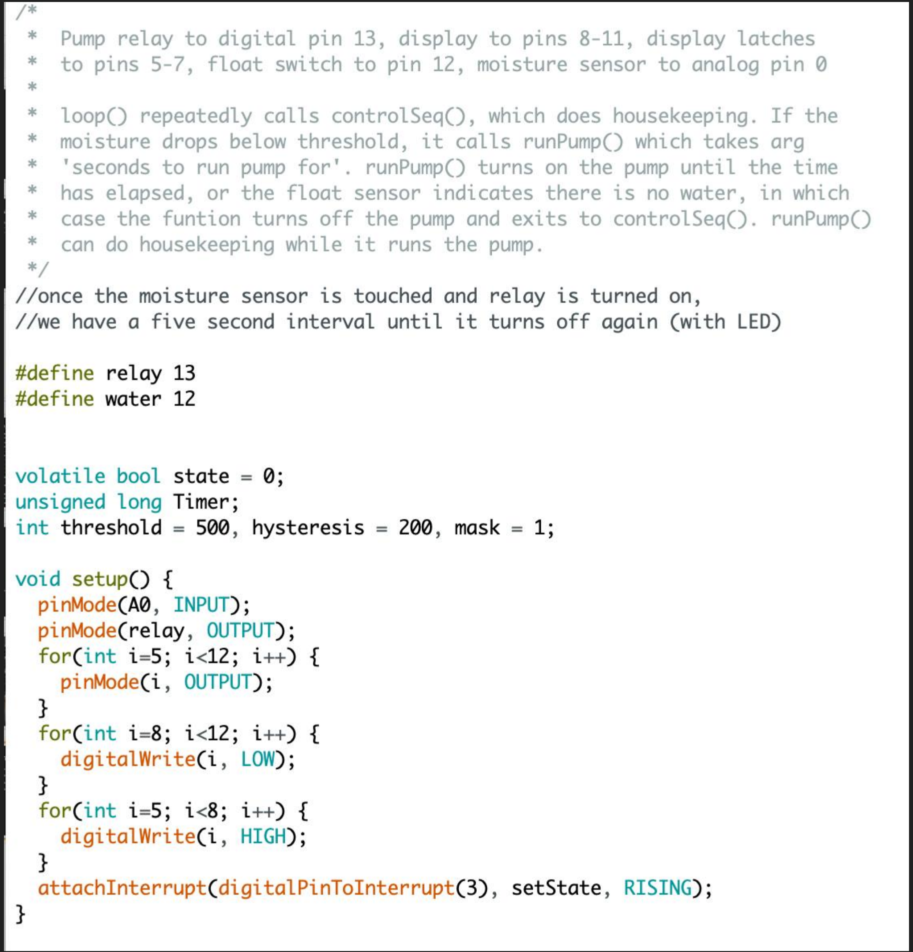
Read Soil Moisture

Monitor Water and Pump

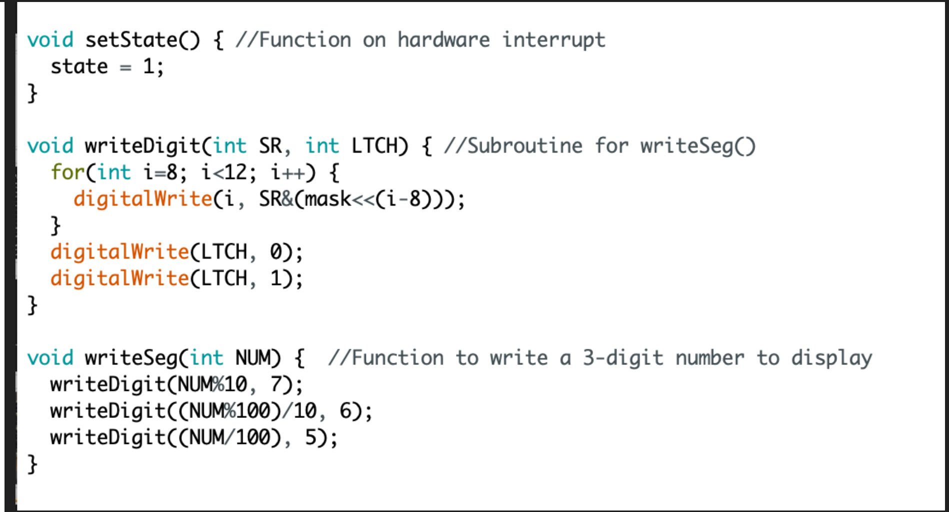
Perform Other Essential Tasks


LOGIC SWITCHES
S4 S5 S6
10K POT
1K POT
R2
S9
SPDT SWITCHES
DFRduino UNO
Keyes_SRly

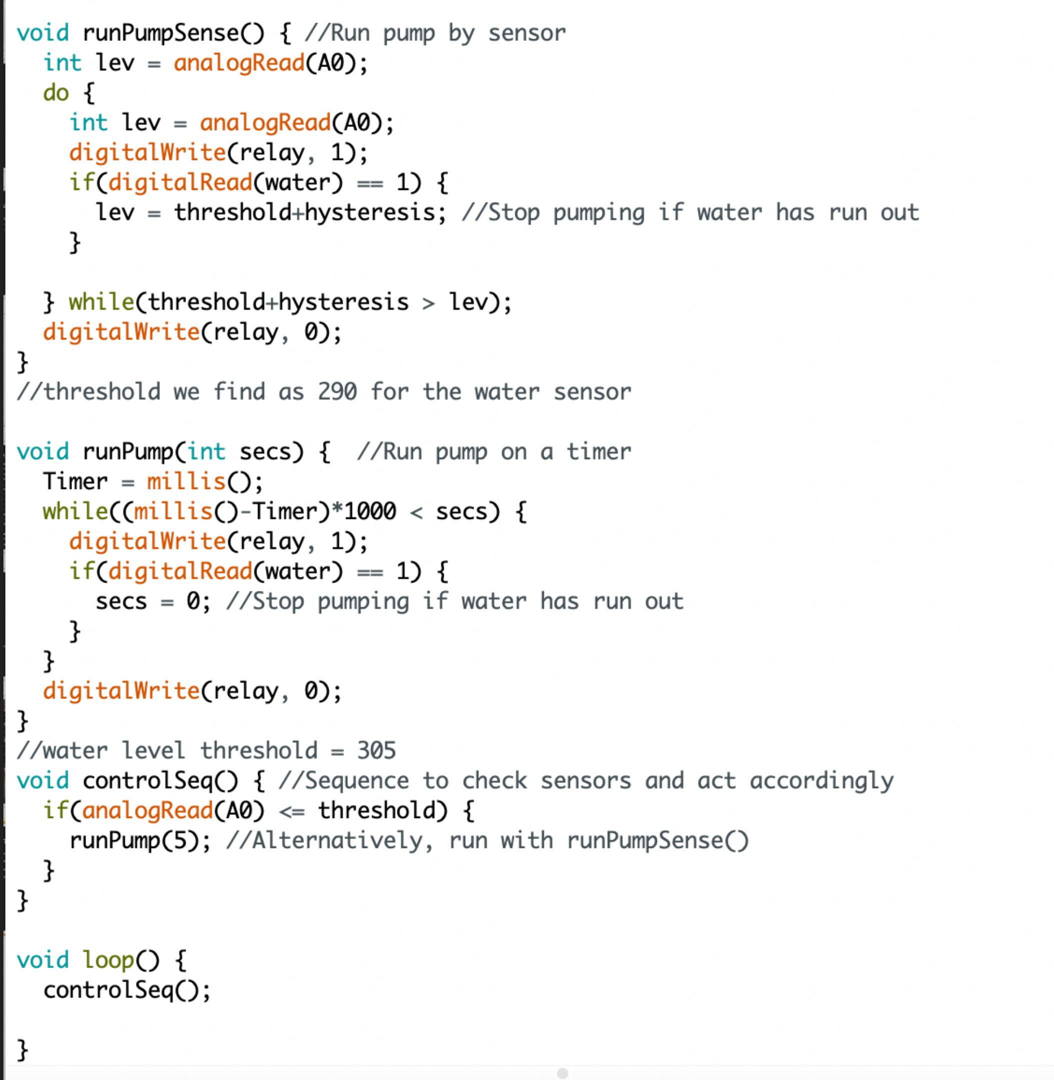

Voltage ok
ADS 1/21/1
5V
TTL
CMOS
LOGIC
DFRduino UNO V3.0 [R3]
LOGIC SWITCHES
R1
10K POT
1K POT
R2
S9
SPDT SWITCHES
S6 S7 S8



```
/*
 *  Pump relay to digital pin 13, display to pins 8-11, display latches
 *  to pins 5-7, float switch to pin 12, moisture sensor to analog pin 0
 *
 *  loop() repeatedly calls controlSeq(), which does housekeeping. If the
 *  moisture drops below threshold, it calls runPump() which takes arg
 *  'seconds to run pump for'. runPump() turns on the pump until the time
 *  has elapsed, or the float sensor indicates there is no water, in which
 *  case the funtion turns off the pump and exits to controlSeq(). runPump()
 *  can do housekeeping while it runs the pump.
 */
//once the moisture sensor is touched and relay is turned on,
//we have a five second interval until it turns off again (with LED)

#define relay 13
#define water 12


volatile bool state = 0;
unsigned long Timer;
int threshold = 500, hysteresis = 200, mask = 1;

void setup() {
  pinMode(A0, INPUT);
  pinMode(relay, OUTPUT);
  for(int i=5; i<12; i++) {
    pinMode(i, OUTPUT);
  }
  for(int i=8; i<12; i++) {
    digitalWrite(i, LOW);
  }
  for(int i=5; i<8; i++) {
    digitalWrite(i, HIGH);
  }
  attachInterrupt(digitalPinToInterrupt(3), setState, RISING);
}
```



```
void setState() { //Function on hardware interrupt
  state = 1;
}

void writeDigit(int SR, int LTCH) { //Subroutine for writeSeg()
  for(int i=8; i<12; i++) {
    digitalWrite(i, SR&(mask<<(i-8)));
  }
  digitalWrite(LTCH, 0);
  digitalWrite(LTCH, 1);
}

void writeSeg(int NUM) {  //Function to write a 3-digit number to display
  writeDigit(NUM%10, 7);
  writeDigit((NUM%100)/10, 6);
  writeDigit((NUM/100), 5);
}
```



```
void runPumpSense() { //Run pump by sensor
  int lev = analogRead(A0);
  do {
    int lev = analogRead(A0);
    digitalWrite(relay, 1);
    if(digitalRead(water) == 1) {
      lev = threshold+hysteresis; //Stop pumping if water has run out
    }

  } while(threshold+hysteresis > lev);
  digitalWrite(relay, 0);
}
//threshold we find as 290 for the water sensor

void runPump(int secs) {  //Run pump on a timer
  Timer = millis();
  while((millis()-Timer)*1000 < secs) {
    digitalWrite(relay, 1);
    if(digitalRead(water) == 1) {
      secs = 0; //Stop pumping if water has run out
    }
  }
  digitalWrite(relay, 0);
}
//water level threshold = 305
void controlSeq() { //Sequence to check sensors and act accordingly
  if(analogRead(A0) <= threshold) {
    runPump(5); //Alternatively, run with runPumpSense()
  }
}

void loop() {
  controlSeq();

}
```

Timeline

Done writing backbone working code

Week 2: Work with sensing system. Write control functions for sensors.

Week 3: Wire up display. Testing with plants to figure out thresholds

Week 4: Assemble hardware. Finish writing code.