Code Snippets & Circuit Diagrams

pH Code

```
const int PHin = 30;// to get voltage passed the op-amp
const int base = 17;  //
const int acid= 18;  //
       // value read from the pot

void setup() {
    // initialise serial communications at 9600 bps:
    Serial.begin(9600);
    pinMode(acid, OUTPUT);
    pinMode(base, OUTPUT);
    pinMode(PHin,INPUT);
}


int counter_add_acid = 0;
int counter_add_base = 0;
int lag = 0;

//to set up a loop

int getPH()
{
    // constant values
    const int phS = 7;        //ph of standard solution
    const float Es = 1.67;     // electric potential at reference or standard electrode
    const float F = 9.6485309*10000;        // faraday constant
    const float R = 8.314510;       // the universal gas constant
    const float ln10 = 2.30258509;

    // initialise
    float ReadPH = 0.0;
    double Temp = 0.0;
    float ph_with_tp = 0.0; //PH calculated with known temperature
    float new_readPH = 0.0;


    // read the Voltage from the pins:
    ReadPH = analogRead(PHin);//since the op-amp shifted the signal by 512mV, we did the reverse.
    new_readPH = map(ReadPH, 0, 1023, 0, 3350);
    new_readPH = new_readPH / 1000;
    Serial.println("new_readPH");
    Serial.println(new_readPH);


    Temp = 25.0; /* getTemperature(); */      // get the temperature value from tp sensor
    // print the results to the serial monitor:

    Serial.println("Temperature:");
    Serial.println(Temp);

    ph_with_tp = (((Es - new_readPH) * F) / (R * (Temp + 273.15) * ln10)) + phS;

    Serial.println("PHValue:");
    Serial.println(ph_with_tp);
    return ph_with_tp;
}
```

```
void setPH()
{
    int PHinput = 12 /* User input PH needed */;       // initialise the PH that was given by the user
    int PHmax = PHinput + 1;
    int PHmin = PHinput - 1;

    int currentPH = getPH();

    if (currentPH > PHmax && lag <= 10)     // wait 3 seconds to see if the value is continuously greater than we expected
    {
        lag += 1;
    }
    else if (currentPH > PHmax && lag > 10)     //the solution need acid
    {
        if (counter_add_acid < 500)
        {
            counter_add_acid += 50;
        }
        analogWrite(acid, counter_add_acid);
        digitalWrite(base, LOW);
    }
    else if (currentPH < PHmin && lag <= 10)        // wait 3 seconds to see if the value is continuously lower than we expected
    {
        lag += 1;
    }
    else if (currentPH < PHmin && lag > 10)        // the solution need base
    {
        if (counter_add_base < 500)
        {
            counter_add_base += 50;
        }
        analogWrite(base, counter_add_base);
        digitalWrite(acid, LOW);
    }
    else        // we are in acceptable range
    {
        digitalWrite(acid, LOW);
        digitalWrite(base, LOW);
        counter_add_acid = 0;
        counter_add_base = 0;
        lag = 0;
    }
    delay(100);     // wait 100 milliseconds before the next loop
}
```

pH Code Explanation

Code in structural english:

Read a voltage for PH

Get the temperature from temperature sensor

Use a formula to calculate the actual PH

The transfer function of the pH electrode is:

$$pH\,(X) = pH\,(S) + \frac{(E_S - E_X)\,F}{RT\,\ln(10)}$$

where

- pH(X) = pH of unknown solution(X)
- pH(S)= pH of standard solution = 7
- E_S = Electric potential at reference or standard electrode
- E_X = Electric potential at pH-measuring electrode
- F is the Faraday constant = 9.6485309×10^4 C mol^{-1},
- R is the universal gas constant = 8.314510 J K^{-1} mol^{-1}
- T is the temperature in Kelvin

Read a acceptable PH from user

Compare the current PH with the acceptable PH

If Higher, add acid

If lower, add base

Else, means we are within the given range, then to nothing

pH Circuit

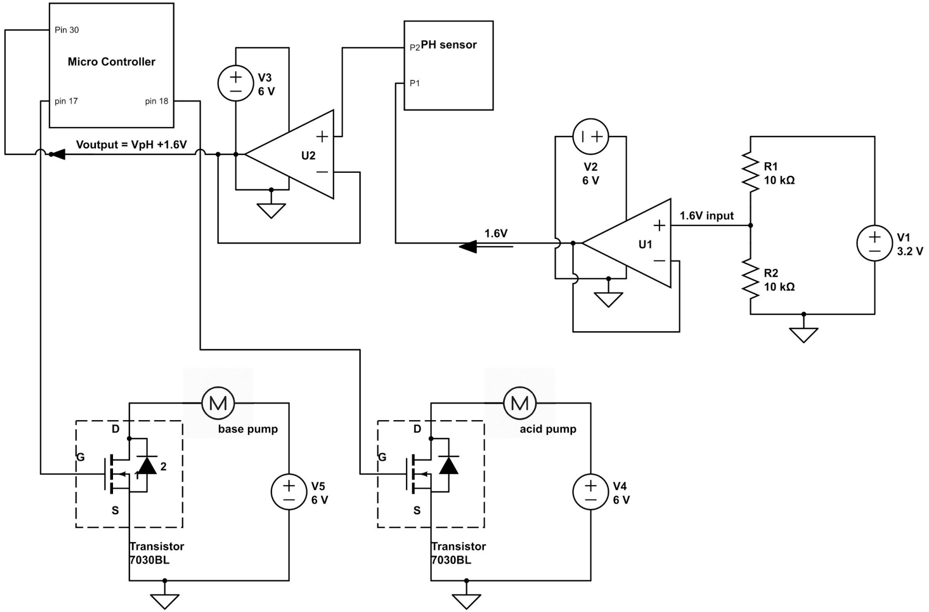


Figure 2.3.3.1 : Completely Integrated pH subsystem

pH Circuit

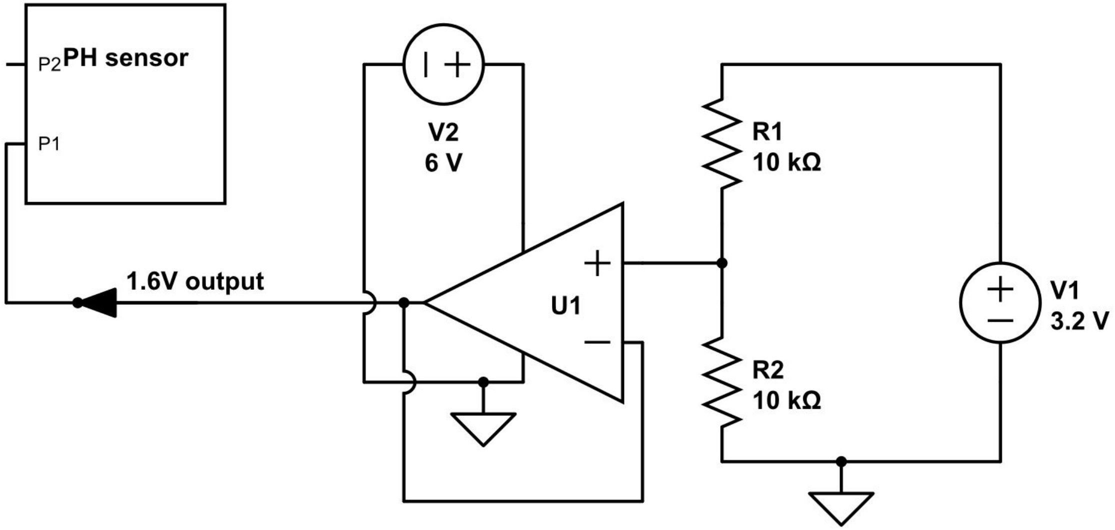


Figure 2.3.1.1 : Op-amp U1 used as a voltage follower to provide 1.6V DC offset to the output of pH probe.

pH Circuit

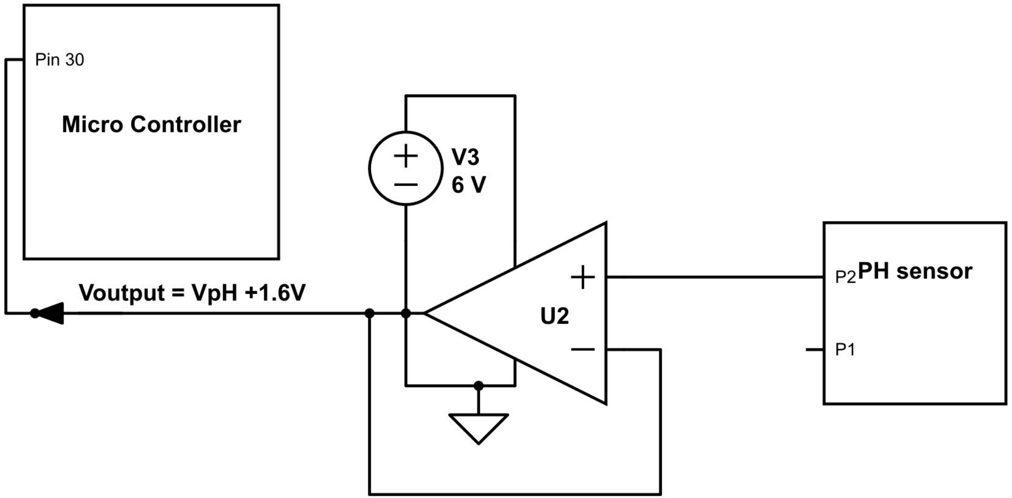
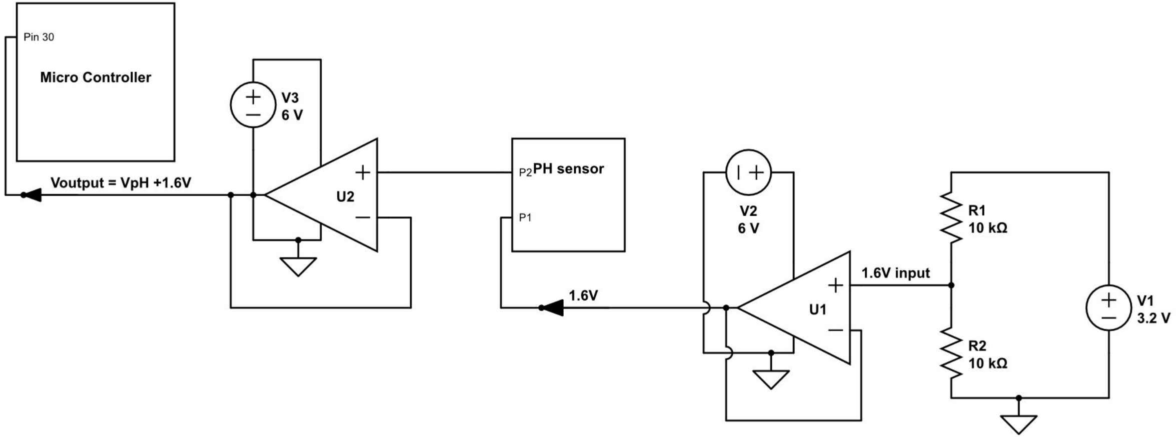


Figure 2.3.1.2 : Op-amp U2 used as a voltage follower to reduce the high impedance of the pH probe.

pH Circuit

Figure 2.3.1.3 : Integrated two op-amps U1 and U2 to provide readable voltage for the microcontroller



pH Circuit

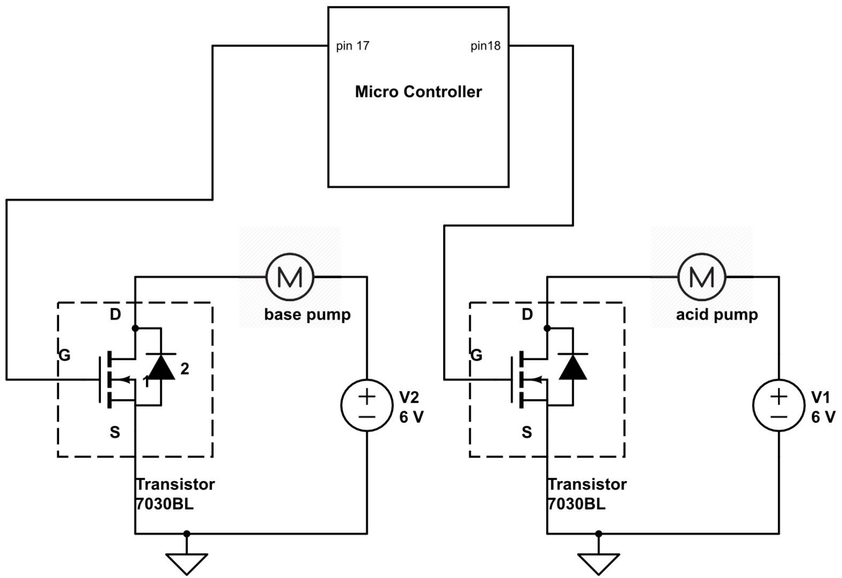


Figure 2.3.2.1 : Transistors controlled by the microcontroller that are used to turn on/off the solution pumps with external power supply.

Stirring Code Explanation°

FUNCTION:
(what is in function)

setup:
Set Serial Baudrate to 9600
Set Motor Pin to OUTPUT
Set Photointerrupter Pin to INPUT

setRPM:
Check if RPM is between 500 to 1500
Set the desired value to what the user inputted

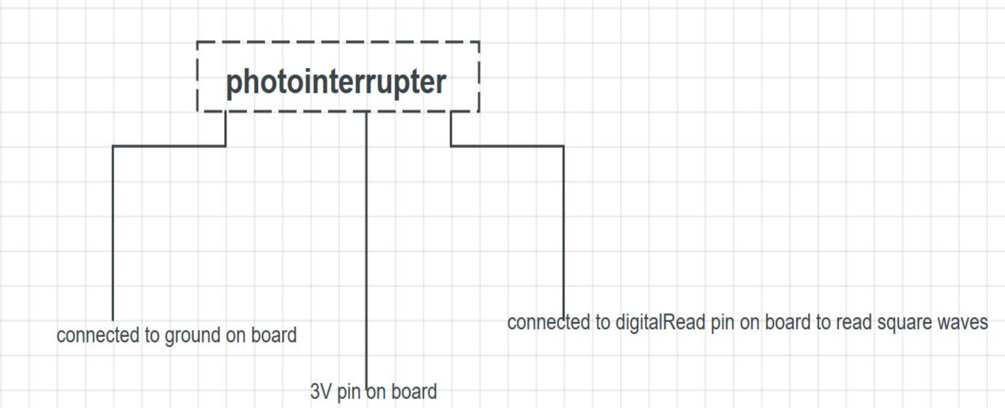
getRPM:
Return the averageRPM

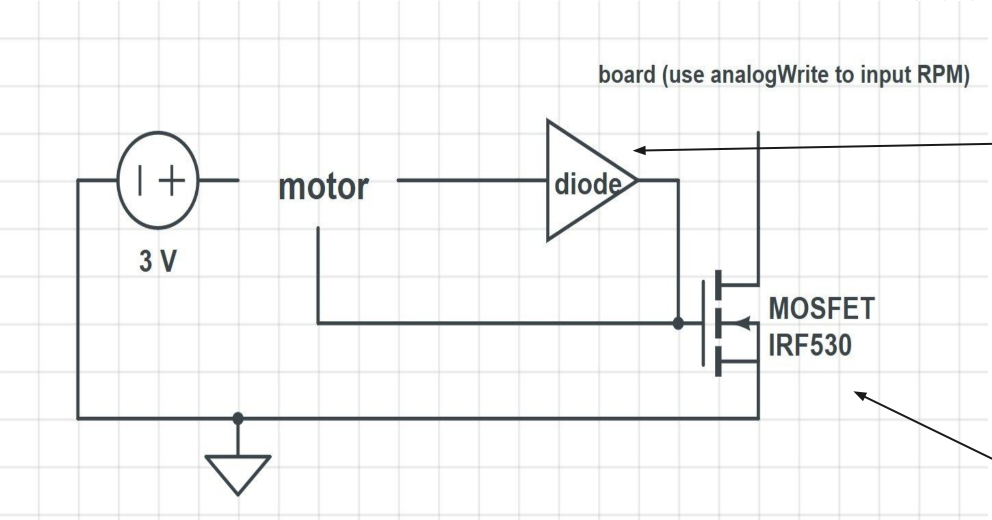
getAverage:
Calculate the average of the last 50 RPM values
Return value and store to averageRPM

loop:
Work out period by working out the time between two peaks
Divide period by two since motor has two blades
Calculate frequency from period
Calculate the RPM by multiplying frequency by 60
if rpm from Photointerrupter > desired value then reduce voltage
if rpm from Photointerrupter < desired value then increase voltage
Pass voltage via analogWrite to Motor Pin

Stirring Code:
https://docs.google.com/document/d/1diVjHVRaAx0RNde1lLMotF-EvJvRV7G8D4ne33curMs/edit?usp=sharing

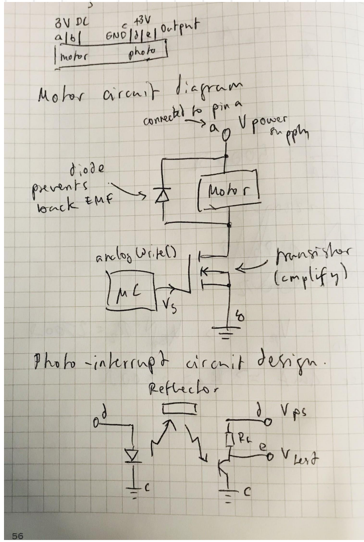
Stirring Circuit





To prevent reverse current which can damage the motor

EDIT: Transistor is only used for switching, not amplifying current



Heating Code (Steinhart-Hart Equation)

```
static const uint8_t RESISTOR = 31;
static const uint8_t HEATER = 11;

double V_total = 3.3;
double R_resistor = 9955;
double SH_A = 1.278808880E-3;
double SH_B = 2.171660200E-4;
double SH_C = 0.9603609520E-7;

/**
 * Gets the thermistor temperature in Celsius.
 */
double heating_get_temperature() {
  double V_thermistor = V_total - analogRead(RESISTOR) * V_total / 1023.0;
  double R_thermistor = (R_resistor * V_thermistor) / (V_total - V_thermistor);

  double ln_R = log(R_thermistor);
  double T = 1/(SH_A + SH_B * ln_R + SH_C * ln_R * ln_R * ln_R) - 273.15;

  return T;
}
```

$$V_{\text{thermistor}} = V_{\text{total}} - V_{\text{resistor}}$$

$$R = R_{\text{thermistor}} = \frac{R_{\text{resistor}} \cdot V_{\text{thermistor}}}{V_{\text{total}} - V_{\text{thermistor}}}$$

$$T = \frac{1}{A + B \ln R + C(\ln R)^3}$$

where A, B and C are coefficients calculated from the thermistor data sheet and R_resistor is the actual resistance of our resistor measured with an Ohmmeter.

Heating Code

```
void setup() {
  Serial.begin(9600);
  pinMode(HEATER, OUTPUT);
}

static const int heater_max_pwm = 255;
static const double heater_start_taper = 1.7;
static const double heater_end_taper = 0.2;

void loop() {

  double target_temperature = ui_get_target_temperature();
  double temperature = heating_get_temperature();

  int heater_pwm = 0;

  if (temperature < target_temperature - heater_end_taper) {

    heater_pwm = temperature < target_temperature - heater_start_taper ?
      heater_max_pwm : map(temperature * 100, (target_temperature - heater_start_taper) * 100, (target_temperature - heater_end_taper) * 100,
      heater_max_pwm, 0);

  }

  analogWrite(HEATER, heater_pwm);

  delay(500);

}
```

Heating Code Explanation

Main loop explanation:

Get the target temperature and current temperature.

The heater_pwm value is:

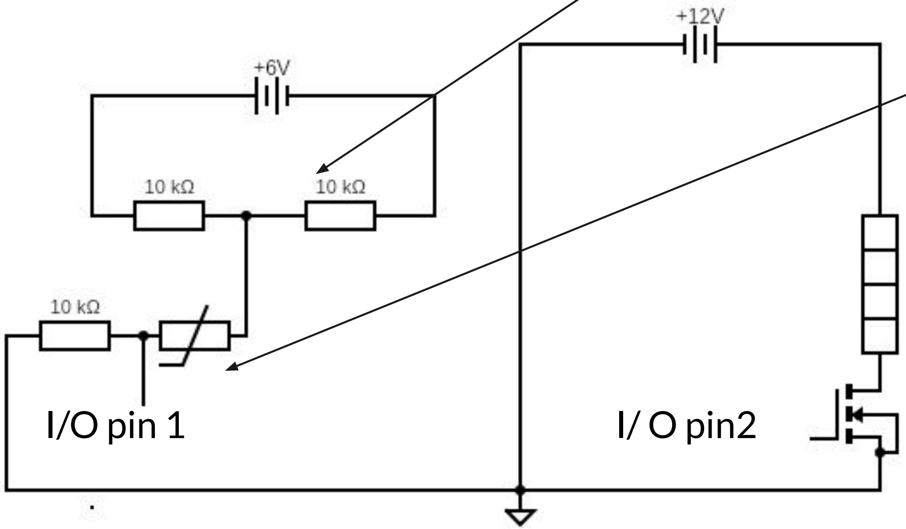
- 255 if temperature < target_temperature - heater_start_taper
- Tapered between 255 and 0 if heater_start_taper <= temperature <= heater_end_taper
- 0 if temperature < target_temperature - heater_end_taper

Analogue write the heater_pwm value to the transistor.

Heating Circuit

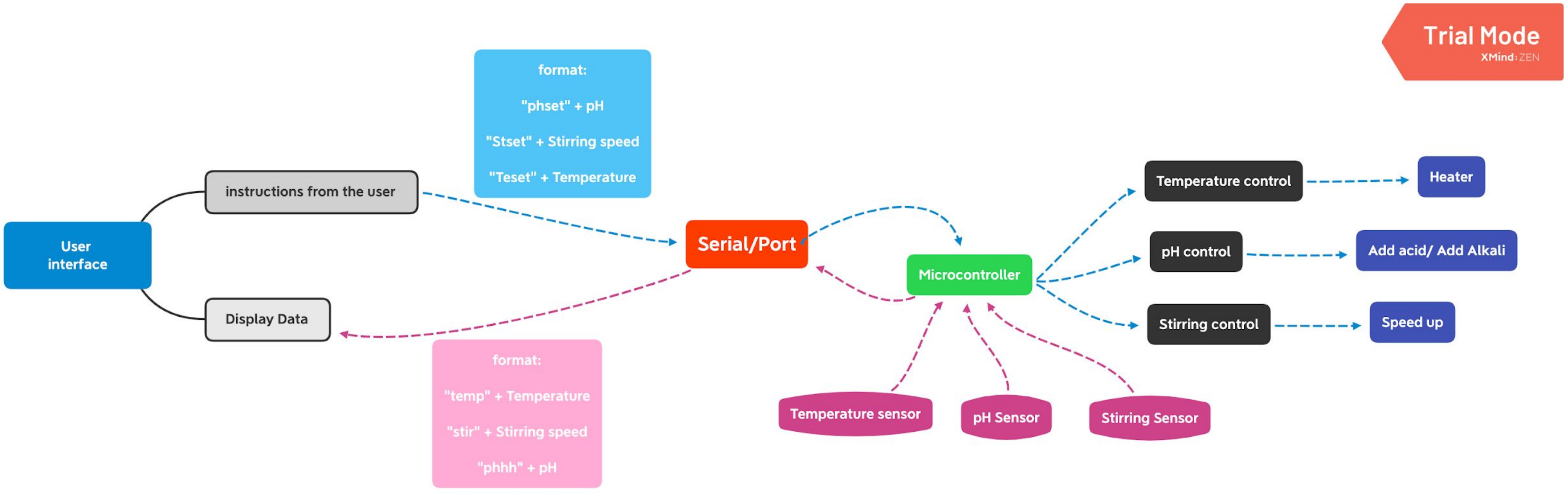
Potential Divider to turn 6 Volts into 3 Volts as we only want to pass 3 volts across the thermistor.

Potential Divider circuit to measure voltage across resistor instead of thermistor as the thermistor would otherwise use all the voltage and always read 3V. Voltage used to work out temperature using Steinhart-hart equation.



Heater

Transistor - Circuit Switch, if HIGH (switch closes) heater is on if LOW (switch opens) heater is off.



User interface explanation