Department of Electrical Engineering
Indian Institute of Technology Bombay

Temperature Control Using Current Reversing Drive on Peltier Element - Cooltier

Group: TUE-KT-4-2

Harsh Vinayak Borkar - 190070027
Vineet Deepak Gala - 190070072
Vipin Rajeshkumar Ochiramani - 190070073

Faculty Mentor: Prof. Kushal R. Tuckley

August 19, 2022

Contents

1 Aim of the Project

The aim of this project is to plan and build a cooler using a current controlling drive on a Peltier element, and control the temperature of the object to be cooled.

2 Design Approach

2.1 Power Calculations

We started with the aim of cooling a standard can with some liquid (about 300 ml) to about 5 °C from a initial room temperature of 25 °C, in a reasonably small time (<15 minutes).

We assumed that the initial object temperature is same as room temperature, i.e., 25 °C, and as soon as we supply current to the Peltier-element, the hotter side of the element reaches 35 °C and the cold side (in contact with the object-to-be-cooled) is at 25 °C (at $t = 0^+$ s).

Let

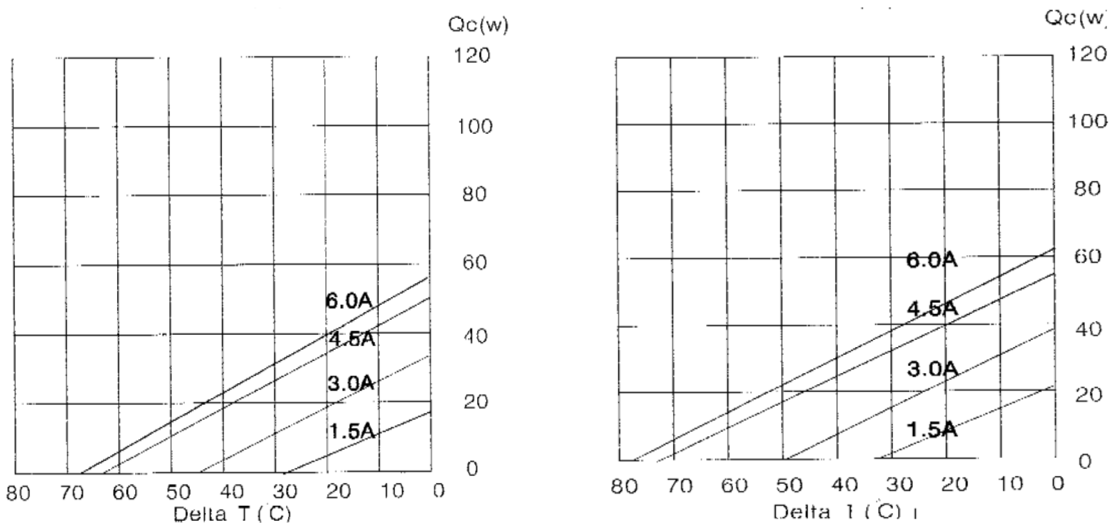
- Θ denote the difference between the hot side and the cold side of Peltier-element (in °C
- Θ_w denote the temperature of object being cooled
- $\Delta\Theta_w$ denote the fall in Θ_w per second

$$\Theta = T_{\text{hot\_side}} - T_{\text{cold\_side}}$$
$$\Theta = 10\ ^\circ\text{C (at } t = 0 \text{ s)}$$
$$\Theta = 10\ ^\circ\text{C} + \int_0^t \Delta\Theta_w \, dt$$

(Assuming object is in thermal equilibrium with the cold side of Peltier-element at all time)

Specific heat of water is 4.2 $\text{Jg}^{-1}\text{K}^{-1}$
Density of water is 1 g/ml
Assuming average volume of can being cooled is V ml,

$$\Delta\Theta_w = \frac{Q_w}{V \times 4.2}$$
$$\frac{d(\Theta)}{dt} = \Delta\Theta_w = \frac{Q_w}{V \times 4.2}$$



(a) $T = 25^oC$ (b) $T = 50^oC$

Figure 2.1: Q_c vs ΔT

Following the TEC-12706 datasheet (fig. 2.1), the Q_c vs Θ graph is linear with slope α and y-intercept β:

$$Q_c = \alpha(\Theta) + \beta$$
$$Q_w = K \times Q_c$$

where $K \in (0, 1)$ is the efficiency of the Peltier element cooling action

$$\frac{dy}{dt} = \frac{K \times (\alpha y + \beta)}{V \times 4.2}$$
$$\frac{dy}{dt} = K' \times (\alpha y + \beta)$$
$$y = \lambda e^{K'\alpha t} - \frac{\beta}{\alpha}$$

where $y = \Theta$, $K' = \frac{K}{V \times 4.2}$ and λ will be decided by the boundary conditions

From TEC1-12706 datasheet, for the hotter side being at 35 °C (308 K) and a 4.5 A current being passed through Peltier, we get

$$\alpha = -0.8$$
$$\beta = 50$$

Assuming $K = 5/6$ and using initial conditions, we get:

$$\Theta = y = -52.5e^{-t/2079} + 62.5$$
$$t = 2079 \times \ln(\frac{52.5}{62.5 - \Theta}) \text{ s}$$

Total time required to get $\Theta = 35$ K , i.e., cooler side is at 0 °C (273 K), is 1344 s $\approx$ 22.4 mins. Since the time required is very high, we decided to take N Peltier elements and pass 4.5 A current through all of them.

Assuming the volume of liquid inside can is V ml ($V \times 10^{-6}$ m^3), density of liquid is D g/ml ($D \times 10^3$ kg/m^3), and specific heat of liquid is S Jg^{-1}K^{-1}($S \times 10^3$ Jkg^{-1}K^{-1}),

$$\frac{dy}{dt} = \frac{N \times K \times (\alpha y + \beta)}{V \times D \times S}$$
$$K' = \frac{K}{V \times D \times S}$$
$$\frac{dy}{dt} = N \times K' \times (\alpha y + \beta)$$
$$y = \lambda e^{NK'\alpha t} - \frac{\beta}{\alpha}$$

Now, for a 4.5 A current passed through each Peltier, we know

$$\alpha = -0.8$$
$$\beta = 50$$
$$\Theta = y = -52.5e^{-NK'4t/5} + 62.5$$
$$t = \frac{5}{4 \times N \times K'} \times \ln(\frac{52.5}{62.5 - \Theta})$$
$$t = \frac{5 \times V \times D \times S}{4 \times N \times K} \times \ln(\frac{52.5}{62.5 - \Theta})$$

Total time required to get $\Theta = 35$ K i.e cooler side is at 0 °C (273 K) is

$$t = \frac{5 \times V \times D \times S}{4 \times N \times K} \times \ln(\frac{52.5}{27.5}) \text{ seconds}$$
$$t = \frac{5 \times 300 \times 1 \times 4.2}{4 \times 2 \times \frac{5}{6}} \times \ln(\frac{52.5}{27.5}) = 611 \text{ s} \approx 10 \text{ min}$$

10 minutes is reasonably low. Hence we go ahead with using 2 Peltier elements for our design.

2.2 Controlling Current Flow Direction

For controlling the flow of current through the Peltier module, an H-Bridge circuit was used. For handling the large flow of currents, power MOSFETs were used. To control the MOSFETs, L293D motor driver was used. An H-Bridge consists of 2 NMOS and 2 PMOS, where maximum 1 NMOS and 1 PMOS are active at a given time. The direction of current was decided by which MOSFETs are on or off, dictated by their gate voltage which was supplied by the L293D motor driver. Motor driver received input from the microcontroller about which pair of MOSFETs to turn on and when

2.3 Achieving Target Temperature

Since the object only needed to be cooled to the target temperature and then this needed to be controlled, establishing a closed loop was important. For this, we planned to use a temperature sensing system which measures the real time object temperature and processes it to suitably supply current to the Peltier for cooling the object. The temperature sensor used was LM35, which was fed to an ADC and was read and processed by a microcontroller. We used a basic ON/OFF control mechanism based on comparator logic, which will be described ahead. The microcontroller was used to control the motor driver which drove the MOSFETs of the H-Bridge.

2.4 Block Diagram

Fig. 2.2 shows the block diagram for all components involved in this project.

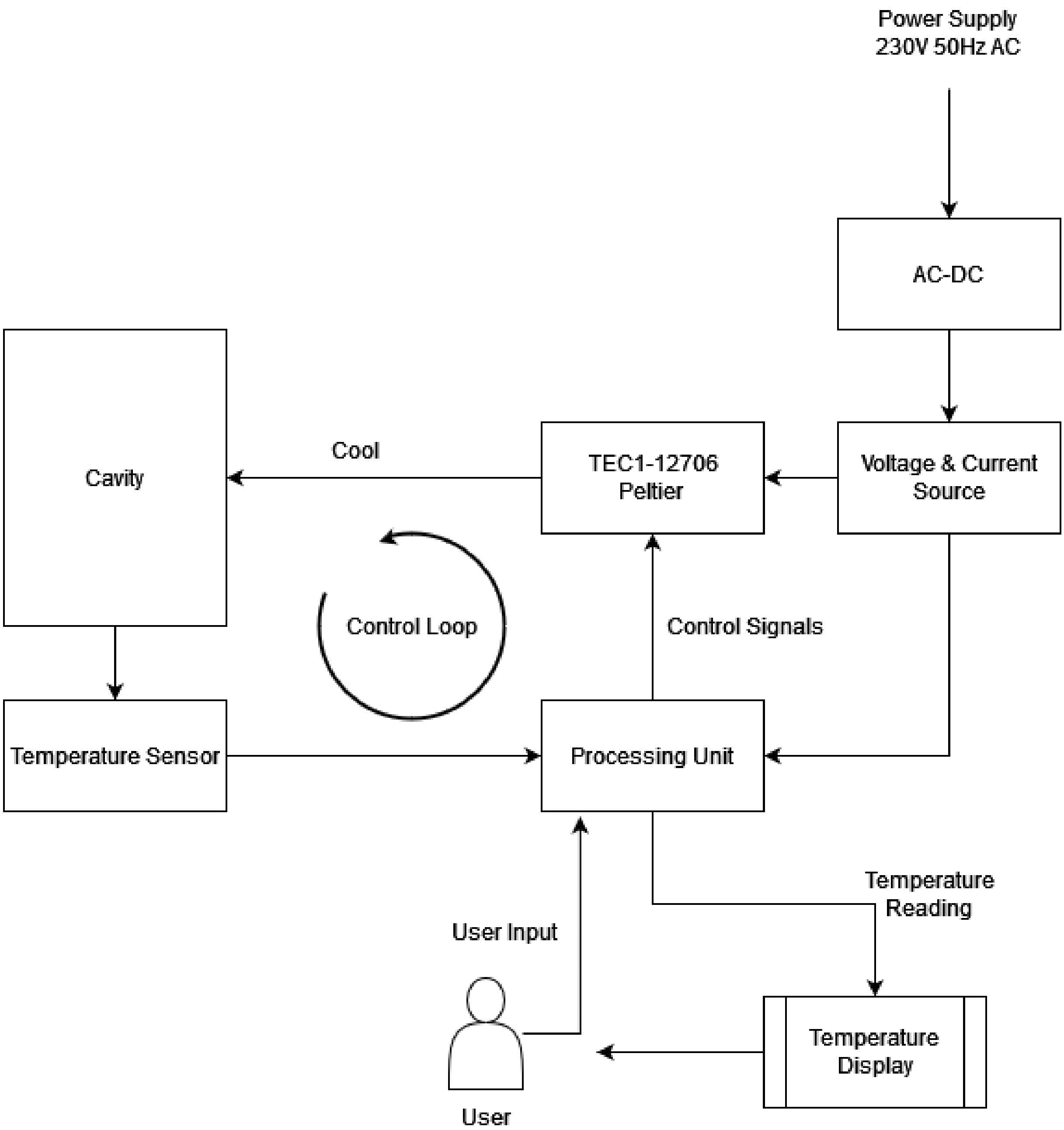


Figure 2.2: Block Diagram

3 Subsystem Breakdown

3.1 Temperature Measurement

We used LM35 Temperature Sensor for this purpose. The most relevant specifications for this component are:

1. Operating Temperature range: -55 °C to 150 °C range.
2. 0.5 °C accuracy guarantee-able (at 25 °C)
3. Power Supply Range: 4 to 30 volts.
4. Linear at 10.0 mV/°C scale factor.
5. Non-linearity only 0.25 °C typical.

Then we interfaced an LCD with our microcontroller to display the temperature measured by the LM35 temperature sensor. The connections for the same are in fig. 3.1.

3.2 Current Driver

The primary component of this was the H-Bridge (fig. 3.2), which was used to drive current through the Peltier element. Based on calculations done earlier, we decided to use 2 Peltier elements each passing 4.5 A through it, taking approximately 10 minutes to cool the object described. Although 1 H-Bridge can drive 2 Peltier elements, it would require over 9 A current carrying capacity. This was impractical to realise in a PCB hence we broke our circuit into 2 H-Bridges, 1 for each Peltier element.

Each H-Bridge consists of 2 NMOS transistors (IRFZ44N) and 2 PMOS transistors (IRF9540N). When the H-Bridge conducts current, only 1 NMOS and PMOS is in saturation. Of the MOSFETs available in lab, we chose IRFZ44N and IRF9540N because:

1. They worked at high current (>10 A)
2. Have a low On-Resistance
3. Low threshold voltage
4. Higher operating temperature range

The gates of MOSFETs were controlled by a motor driver. A motor driver was used here because it is easy to operate and supports low current applications (like gate voltage control). The motor driver we chose was L293D as it satisfied our criteria and was available in lab. The input to the motor driver was from the microcontroller.

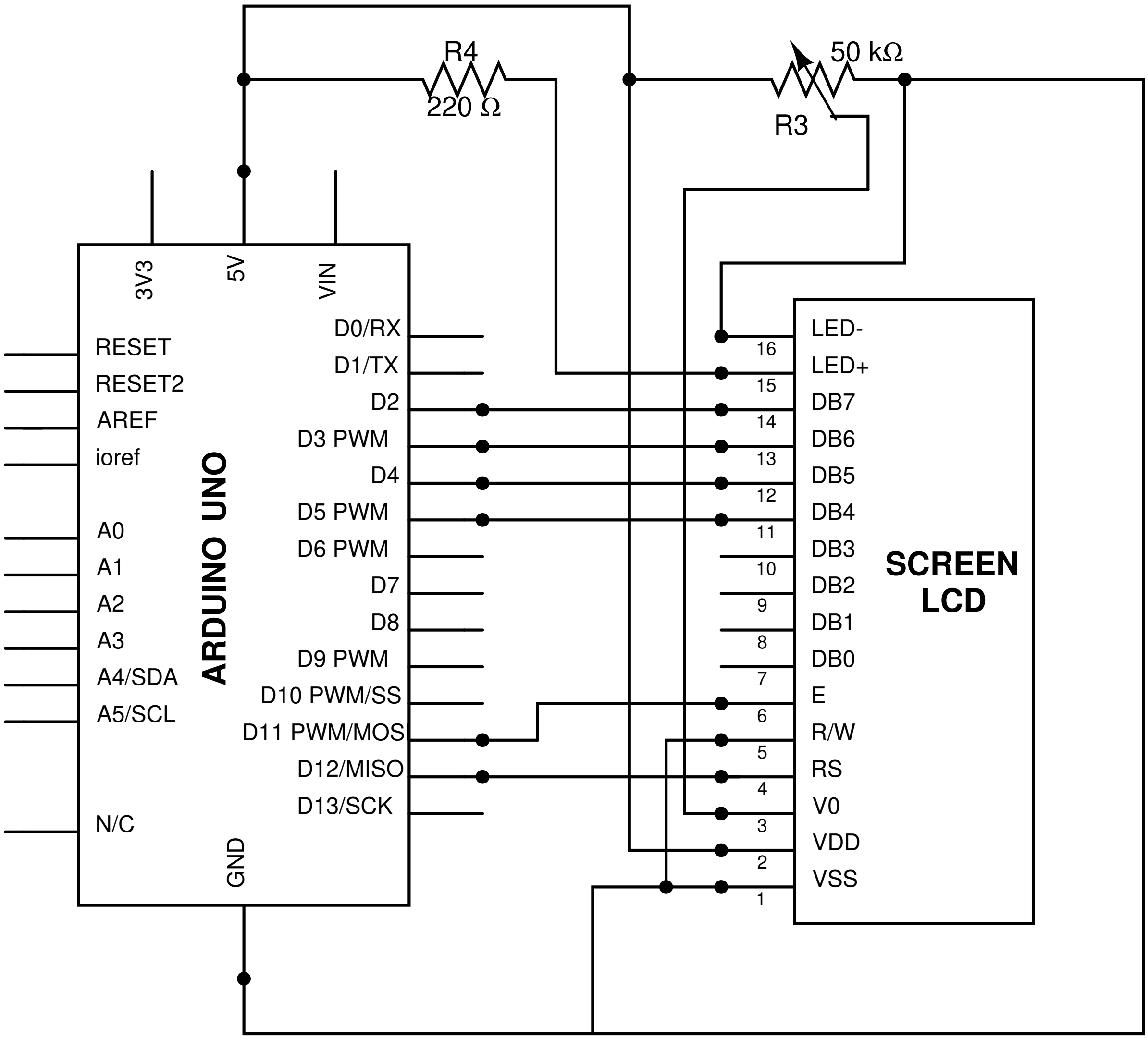


Figure 3.1: LCD Interfacing

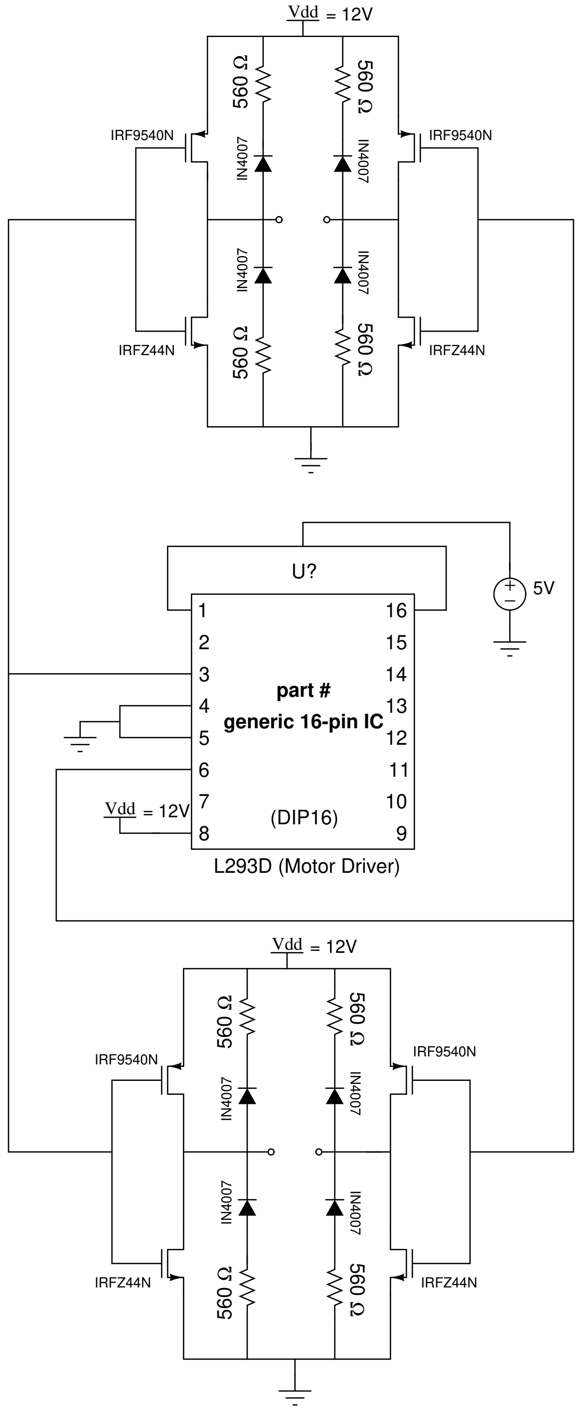


Figure 3.2: H-Bridge Circuit

3.3 Mechanical

This includes the outer covering and other structures to support the system mechanically. The mechanical framework is responsible for:

1. Providing insulation to the inner objects (to be cooled)
2. Making the cavity airtight
3. Have all the necessary provisions for wires to pass in and out of the cavity (to drive Peltier, temperature sensor, etc.), and have provisions for mounting heat sinks
4. Provide a mechanism to cool the hot side of Peltier element and prevent the heat from reaching the cooler side
5. Provide an opening to insert and remove the object(s) to be cooled

We chose to use 3 mm acrylic sheets for this as they are very good insulators of heat, can be cut and stuck to form an enclosure. We used laser cutter to cut the sides of the box as per our required dimensions, along with the cuts required for other purposes (Peltier, heat sink, wires etc). The sheets were stuck using an epoxy glue and were made airtight further by using "Mseal" at the edges.

For heat sinks, we chose processor fans. These have fans which dissipate heat by convection and are powered by the 12 V common supply. They could not make direct contact with the Peltier plates and hence we made "our own heat sink" using a block of mild steel. This was cut and polished by us in the required dimensions and was de-rusted to ensure heat conduction. Thermal paste was used to ensure full thermal contact between surfaces. Fig. 3.3 shows the cross section of the side view of the entire set-up.

3.4 User Input

For taking the user input, a simple potentiometer circuit was used. The state of the potentiometer is measured by the microntroller by reading the analog voltage at the center terminal of the potentiometer. This measured voltage is mapped to temperature in the microcontroller and the subsequent processing is carried out with reference to this target temperature. Target temperature is calculated as

$$T_{\text{target}} = (T_{\text{max}} - T_{\text{min}})\frac{V}{V_{\text{ref}}} + T_{\text{min}}$$

T_{min} and T_{max} are the minimum and maximum target temperatures we want to set, V_{ref} is the reference voltage across the potentiometer and V is the measured voltage.

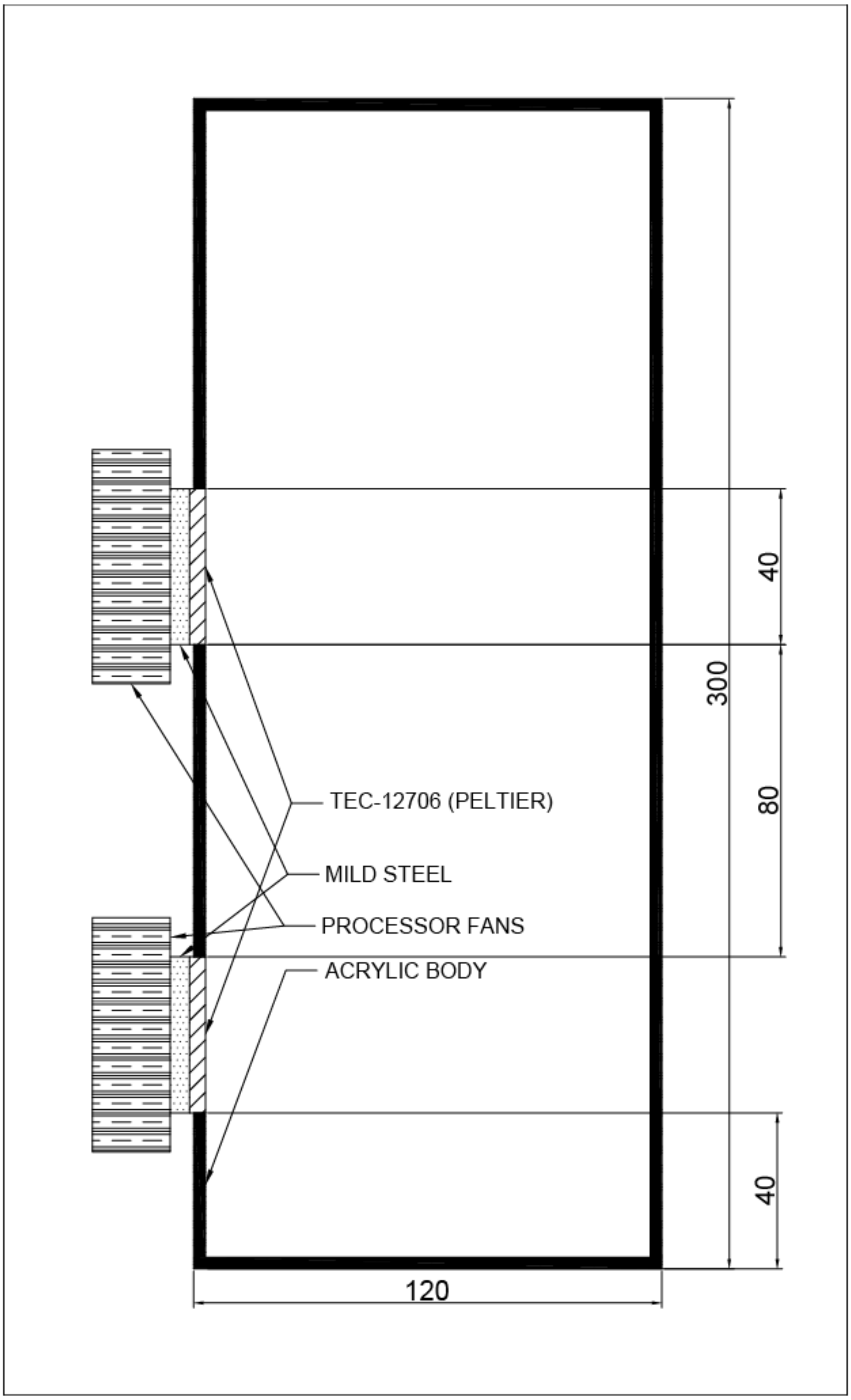


Figure 3.3: Side View Cross Section

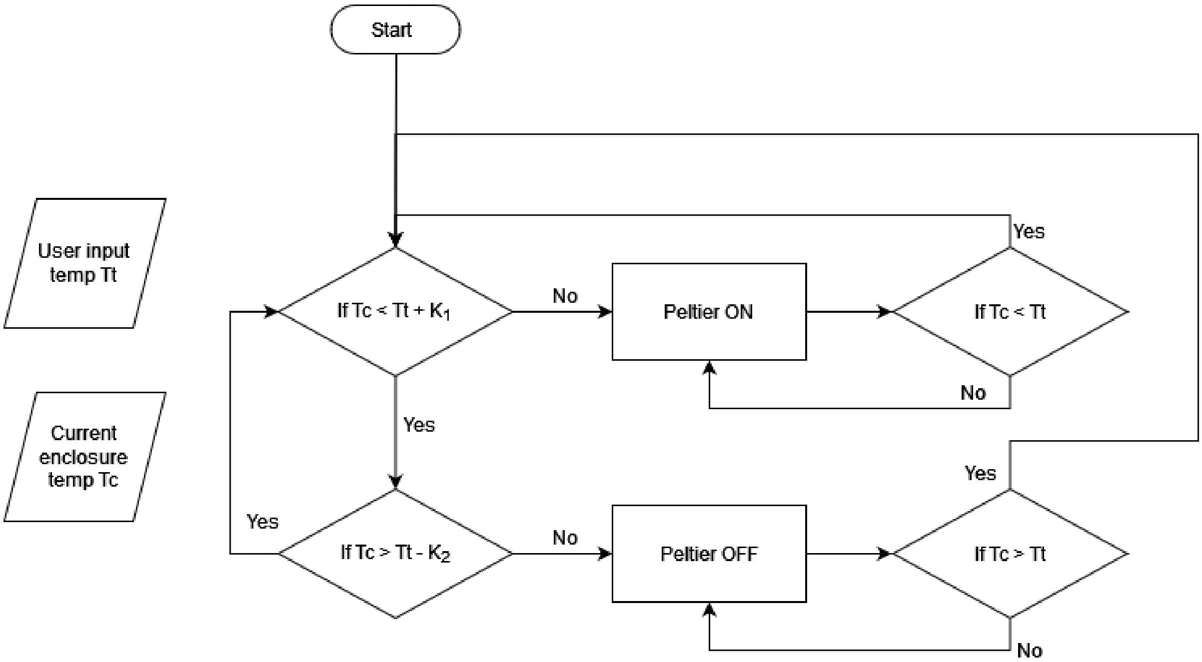


Figure 3.4: Flowchart

3.5 Controller

We used the Arduino UNO microcontroller. It is supplied by the common 12 V supply which also supplies the H-Bridge. It provides 10 bits of resolution in its ADC, which is compatible with the aforementioned LM35 temperature sensor, and thus eliminates the need of an interfacing logic circuit. It also provides 5 V digital signals to the L293D motor driver, controlling the working of H-Bridge and hence the current flow direction through the Peltier element. It also interfaces with the LCD display to display the temperature.

Moreover, it is responsible for all the data handling. Based on measured temperature and the target temperature it reads, it provides signal to L293D motor driver. The logic is described in fig. 3.4. Tc and Tt are measured temperature and target temperature respectively. K_1 is the upper acceptance margin and K_2 is the lower acceptance margin. In other words, $Tc \in (Tt - K_2, Tt + K_1)$ is desired and we cool until we reach the lower bound. Here, we assume that turning off the Peltier element causes the object's temperature to rise. We stop cooling when we reach the lower bound and start cooling once we are above the upper bound.

3.6 Power Supply

The power-supply module used was an SMPS, which takes 230 V AC mains as input and outputs 12 Volts regulated DC voltage voltage. The power-supply modules is capable of delivering currents up to 10A. It acts as the Vdd for the H-Bridge. The supply for processor fans is given directly from the 12 Volts output. Also, Arduino is driven by the same source. For the L293D motor driver and LCD display, the required 5 Volts input voltage was provided by Arduino. The Peltier modules required a current of 4.5 A each — this current was provided by the power-supply module.

4 Test Results

4.1 H-Bridge testing

The H-Bridge circuit was laid out and the gates were controlled by the L293D motor driver. L293D motor driver had an input Vss of 12 V which would be supplied to the gates when it receives 'HIGH' input for the same. L293D input was drawn from the microcontroller which supplied 5 V corresponding to 'HIGH' signal and 0 V for a low sigal. The microcontroller was programmed to switch modes every 5 seconds and output voltage at H-Bridge nodes was observed

Open-circuit testing

For testing in open circuit configuration, the output nodes of H-Bridge circuit were kept open and the voltage difference at output nodes was measured. The circuit switched correctly as per our expectations. It gave +12 V and -12 V in the 2 operating modes and gave 0 V output for same gate voltage across all MOSFETs, as seen in fig. 4.1. In the fig., yellow line shows to gate voltage of 2 MOSFETs (1 NMOS and 1 PMOS) and blue line shows the gate voltage of other 2 MOSFETs. The red line shows the output voltage difference seen at the H-Bridge terminals. When both, the yellow and blue lines, are at HIGH or LOW level, 0 voltage is seen at the H-Bridge output. In other cases, H-Bridge output can be seen at +12 V or -12 V depending on the inputs, implying the correct working of H-Bridge.

Low Load Circuit

This time, a large resistance value was used as a load to operate the H-Bridge at small currents (upto 0.5 A). This also saw results in lines with expectations, same as what we saw without any load.

Figure 4.1: H-Bridge Testing

High Current Circuit

The H-bridge circuit and the L293D circuit was soldered on a general-purpose board and the conducting tracks were drawn to be wide enough to support 4,5 A. This circuit was used to carry-out high current testing. To prevent the MOSFET's from burning, heat sinks were attached. First, we used power resistors as load (current upto 2 A) with high power rating. We got satisfactoory results with them. Then finally we used a Peltier element and got perfect results for 4.5 A current.

4.2 LCD testing

All connections were made as in fig. 3.1 between the mircontroller and LCD. The potentiometer was adjusted to set the proper contrast for display text on the LCD. The text displayed on the LCD was compared to that on Serial Monitor of Arduino, enabling the verification of LCD's operation.

4.3 LM35 testing

For testing LM35 sensor, the readings obtained from the LM35 sensor were compared with the readings of a standard temperature measuring device (available in lab, used as reference). We only observed a biasing error in our measurements and they were otherwise varying by same amounts as the reference when measured in different temperature environments. From the obtained results, the bias error was successfully removed.

4.4 Cumulative testing

All modules were combined here and the entire setup was tested, including the control loop and the mechanical setup. We observed that MOSFETs heat instantly despite a heat sink. Due to lack of time, we couldn't test it on a 300 ml liquid container. We performed cooling on a metal block instead which cools rather faster, but uses the exact same principle as cooling any other object. The temperature of the block dropped from 25 °C to 15 °C in about 3 minutes, as seen on the LCD display.

5 Bill of materials

Table 1: Bill of Materials

Sr.No	Quantity	Components	Price (in INR)	Availability in Lab
1	1	Temperature sensor (LM35)	-	✓
2	2	Processor Fans	500/-	×
3	1	Power Supply	1000/-	×
4	2	Peltier Modules (TEC1-12706)	-	✓
5	-	Diodes and MOSFETs	-	✓
6	3	Acrylic sheet	500/-	×
7	1	Micro-controller (Arduino)	-	✓
8	1	LCD Display	-	✓

6 References

1. HB Electronic, "Thermoelectric Cooler", TEC1-12706 datasheet.
2. International Rectifier, "Power MOSFET", IRF9540N datasheet.
3. International Rectifier, "Power MOSFET", IRFZ44N datasheet.

7 Appendix

7.1 PCB layout

The PCB layouts of our circuit is shown in the Figure below

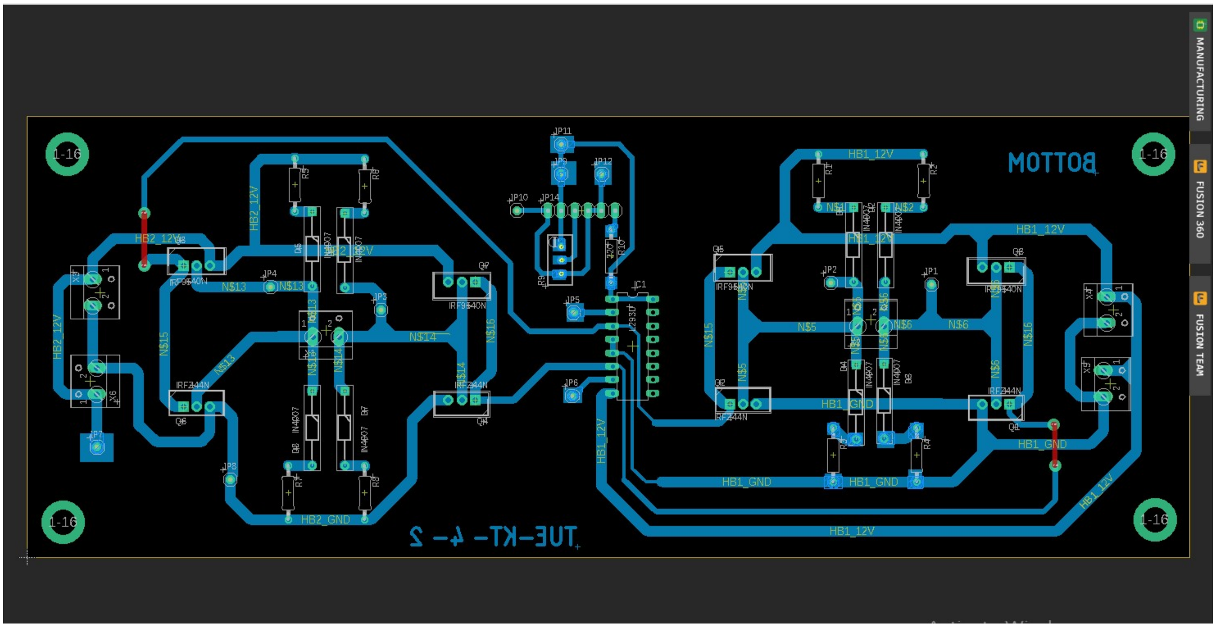


Figure 7.1: H-Bridge PCB layout

7.2 Arduino Code

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
const int LM_35 = A1;
const int Potentiometer = A3;
const int Hbridge_High = 9;
const int Hbridge_Low = 8;
int input_temperature_value = 0;
int input_voltage_value = 0;
float temperature_obtained = 0;
float temperature_desired = 0;
float temperature_margin = 2.0;
int Tmax = 25;
```

```
int Tmin = 5;
int Vref = 5;
bool state = true;

void setup() {
   Serial.begin(9600);
   lcd.begin(16, 2);
   pinMode(Hbridge_High, OUTPUT);
   pinMode(Hbridge_Low, OUTPUT);
}

void loop() {
   input_temperature_value = analogRead(LM_35);
   temperature_obtained = (5.0 * input_temperature_value * 100.0) / 1024;

   input_voltage_value =  analogRead(Potentiometer);
   temperature_desired = (1.0*(Tmax-Tmin)*input_voltage_value)/Vref + Tmin;


   Serial.print("Temperature is : " );
   Serial.println(temperature_obtained);
   lcd.setCursor(0,0);
   lcd.print("Temp = ");
   lcd.print(temperature_obtained);

   if(temperature_obtained>temperature_desired+temperature_margin)
    {
      digitalWrite(Hbridge_High,HIGH);
      digitalWrite(Hbridge_Low,LOW);
      state=true;
    }
    else if(temperature_obtained>temperature_desired-temperature_margin&&state)
    {
      digitalWrite(Hbridge_High,HIGH);
      digitalWrite(Hbridge_Low,LOW);
    }
```

```
    else if(temperature_obtained>temperature_desired-temperature_margin&&!state)
    {
      digitalWrite(Hbridge_High, LOW);
      digitalWrite(Hbridge_Low,LOW);
    }
    else
    {
      digitalWrite(Hbridge_High, LOW);
      digitalWrite(Hbridge_Low,LOW);
      state=false;
    }
    delay(1000);
}
```