Cristihan David Meza Poveda
Brayan Camilo Joya Herrera
Laura Andrea Hurtado Acosta
Geronimo Marin Hurtado
Juan Camilo Muñoz Ramírez
Herney Vega
Veronica Garcia Castro

TERMOFORMADORA - PROYECTO ANÁLOGA

Universidad de los Andes
Facultad de Ingeniería

Universidad de los Andes | Vigilada Mineducación
Reconocimiento como Universidad: Decreto 1297 del 30 de mayo de 1964.
Reconocimiento personería jurídica: Resolución 28 del 23 de febrero de 1949 Minjusticia.

Contenido

1. CARACTERIZACIÓN DEL PROBLEMA

ARTHUR C. CLARKE

Cualquier tecnología lo suficientemente avanzada es indistinguible de la magia

1.0.1 Introducción

El termoformado es una técnica empleada en la industria que consiste en calentar láminas de material termoplástico para moldearlas sobre un diseño preestablecido. Este proceso aprovecha las propiedades de los plásticos, que se vuelven maleables al aumentar su temperatura, y es utilizado en múltiples sectores como el empaque, la manufactura y el transporte, gracias a su versatilidad y eficiencia.

El equipo encargado de llevar a cabo este procedimiento es la máquina termoformadora, la cual combina sistemas de calentamiento, succión o presión, y enfriamiento para garantizar el correcto moldeamiento de las piezas a termo formar. Este tipo de maquinaria es clave en la producción industrial moderna, ya que permite obtener productos con un acabado detallado, optimizando simultáneamente los tiempos de fabricación y los costos asociados.

En comparación con otros métodos de transformación de plásticos, el termoformado tiene ventajas económicas, particularmente en producciones de menor escala o especializadas. Sin embargo, el alto costo de las máquinas termoformadoras tradicionales representa una barrera significativa para su implementación en pequeñas empresas.

Desarrollar una máquina termoformadora de bajo costo como un proyecto enfocado en electrónica analógica constituye una propuesta práctica e innovadora. Este no solo permite explorar el diseño y control de circuitos, sino que también ofrece una solución accesible para sectores con recursos limitados o en crecimiento. Incorporar componentes analógicos en el diseño permite un control eficiente de variables críticas del proceso, como la temperatura y la presión, manteniendo bajos costos y garantizando resultados de calidad.

1.0.2 Objetivos

- Diseñar y construir un termoformador que permita moldear láminas de plástico con dimensiones de 25x25 cm, incluyendo subsistemas de calentamiento y vacío que trabajen de forma conjunta, precisa y eficaz para realizar el proceso de termoformado de manera correcta y replicable.

- Evaluar el desempeño térmico del sistema de calentamiento, asegurando una distribución uniforme de la temperatura sobre la lámina de plástico. Este análisis debe incluir la medición de temperaturas y su estabilidad durante el proceso, evitando degradación térmica o zonas con temperaturas no adecuadas, que puedan afectar los resultados del termoformado.

- Incluir en el diseño un sistema de vacío que genere la presión necesaria para que el plástico, tras alcanzar la temperatura correcta en el proceso de calentamiento, se adhiera al molde con precisión. Este análisis debe evaluar la capacidad del sistema para capturar los detalles del molde sin dañar la lámina ni generar rugosidad, asegurando una presión uniforme y adecuada.

- Implementar un sistema de control térmico que regule la temperatura de operación, además de diseñar medidas de seguridad como indicadores visuales y un manual de usuario que facilite la manipulación del prototipo y minimice los riesgos asociados con el uso del termoformador, tanto por el calor como por el sistema de vacío.

- Documentar el proceso, presentando cálculos, diagramas de bloques y esquemáticos, detalles del diseño, listado de componentes, presupuesto, resultados experimentales y conclusiones que permitan replicar y mejorar el sistema en futuras aplicaciones en el campo de la ingeniería.

2. DIAGRAMA DE BLOQUES

A continuación se encuentra el diagrama de bloques, el cual ilustra de manera estructurada las etapas principales del proyecto y las relaciones entre ellas.

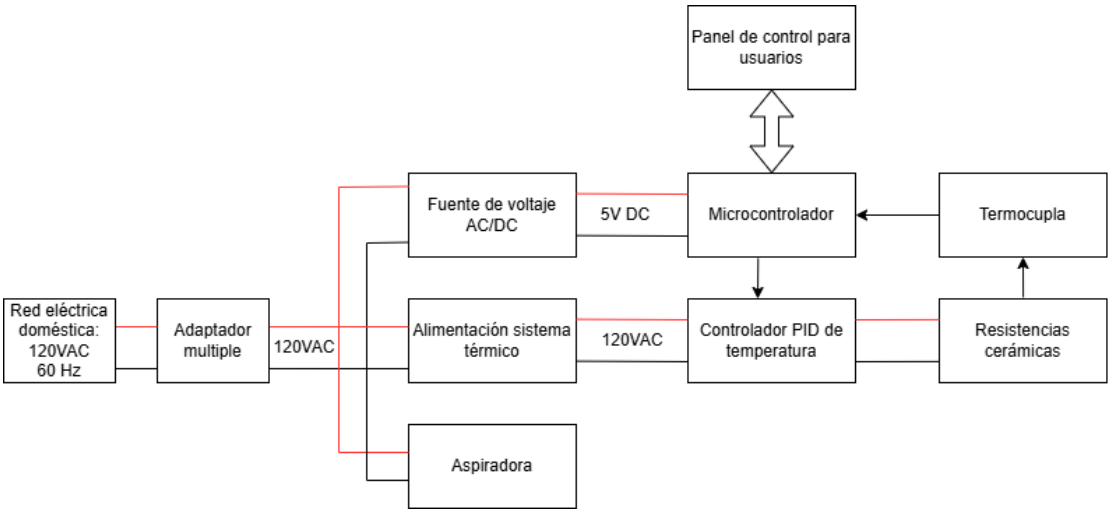


Figure 2.1: Diagrama de bloques

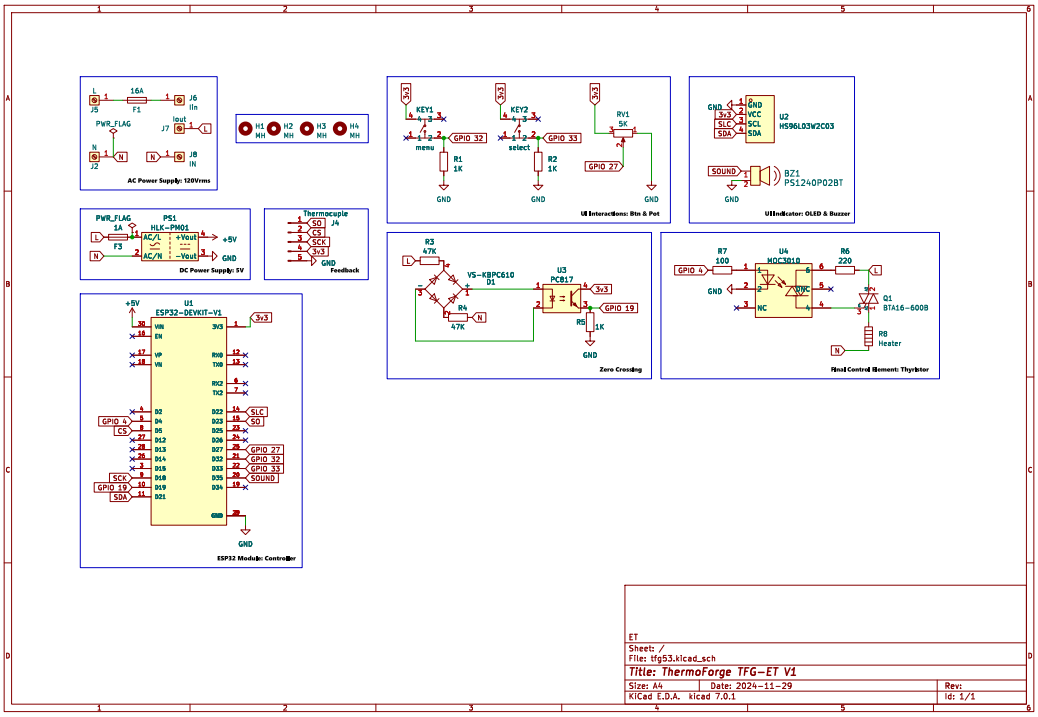
3. PLANOS ELÉCTRICOS

JANE GOODALL

No se puede pasar un solo día sin tener un impacto en el mundo que nos rodea. Lo que hacemos marca la diferencia, y tenemos que decidir qué tipo de diferencia queremos hacer.

A continuación se presentan los planos electricos,

3.0.1 ESQUEMÁTICOS



AC Power Supply: 120Vrms

Este elemento comprende la fuente principal de energía para el dispositivo. Se cuenta con dos terminales correspondientes a la entrada AC/120V del suministro de energía del hogar *(J5, J2)*, además de tres terminales para el interruptor de encendido del sistema.

DC Power Supply: 5Vrms

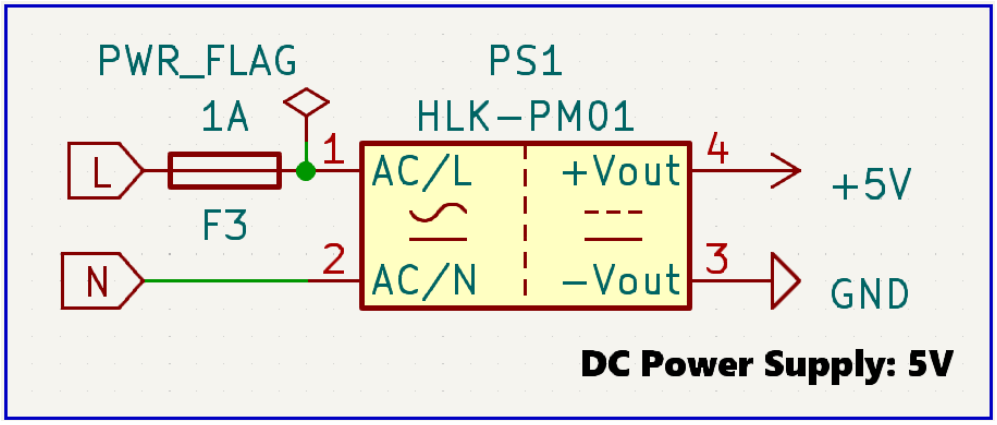


Figure 3.1: DC Power Supply: V_{RMS}

Se cuenta con la fuente HLK-PM01 encargada de convertir la alimentación principal de 120V a 5V, con el propósito de alimentar adecuadamente el microcontrolador elegido (ESP32 DEV KIT V1).

Zero Crossing

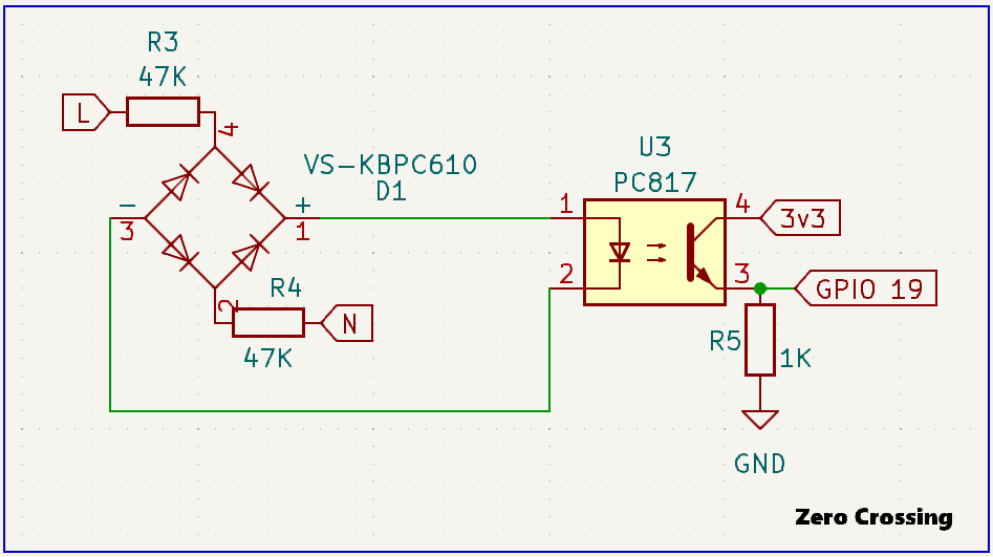


Figure 3.2: Zero Crossing

Este elemento permite obtener una lectura del instante en que la señas AC de alimentación tiene un cruce por cero. Se realiza con el propósito de controlar los tiempos en los pulsos del elemento de control que se presenta más adelante.

Para la correcta lectura se cuenta con el puente rectificador **KBPC610**, el cual se conecta a según la convención tradicional, siendo la entrada la señal AC de 120V, y en la salida, una onda

con el ciclo negativo rectificado a positivo.

Finalmente la salida rectificada es conectada a un opto-acoplador, el cual cuenta con una salida en configuración de pull-down, sin embargo, se inicia con el interruptor normalmente cerrado, teniendo una salida de 3.3V. De esta manera cada que la señal rectificada tenga un cruce por el cero, el interruptor estará abierto, teniendo una lectura de 0V.

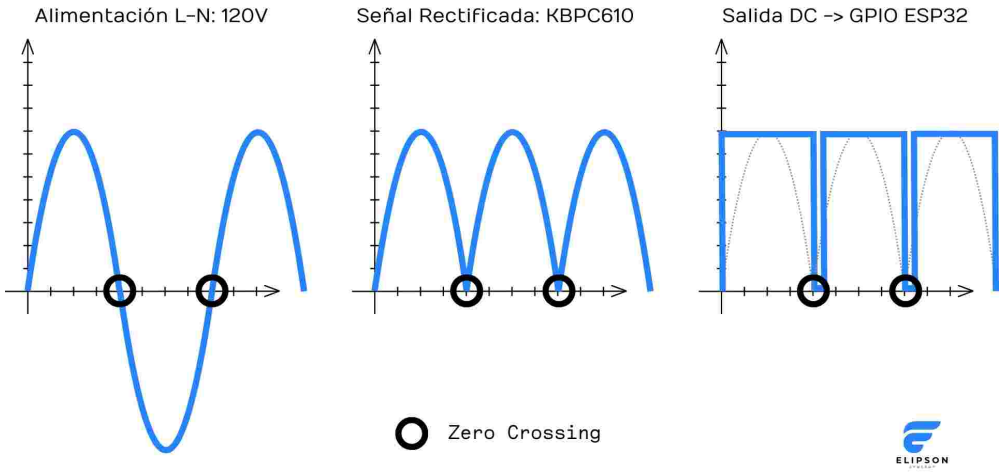


Figure 3.3: Funcionamiento Zero Crossing

Final Control Element: TRIAC

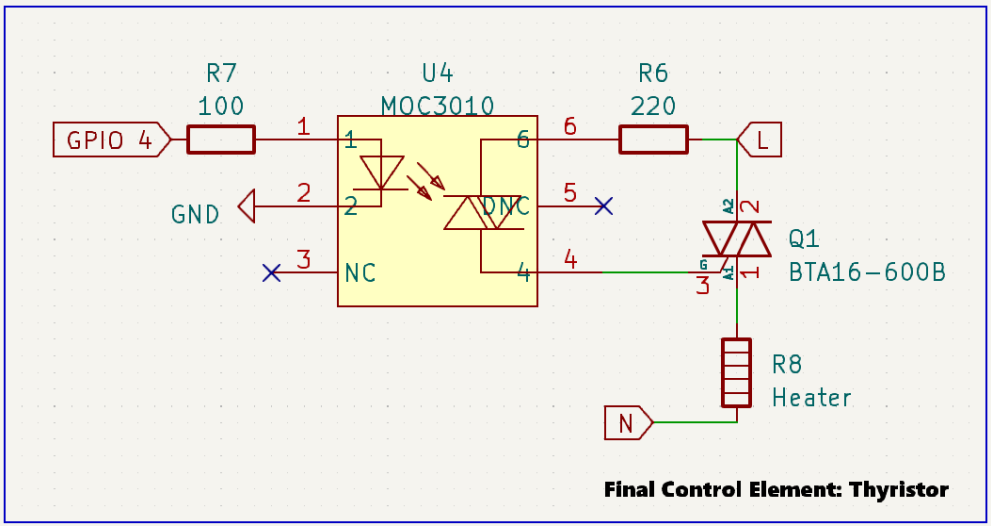


Figure 3.4: Final Control Element: TRIAC

El elemento final de control comprende un TRIAC, el cual se encarga de regular la potencia entregada al calentador. El TRIAC logra regular la potencia recortando la onda de voltaje en base a un pulso el cual es controlado por un delay o tiempo muerto proporcionado por el microcontrolador y el control PID. Para lograrlo se cuenta con el opto-acoplador tipo TRIAC MOC3010, el cual tiene a la entrada el pulso del microcontrolador, y su salida conectada al TRIAC principal BTA16-600B.

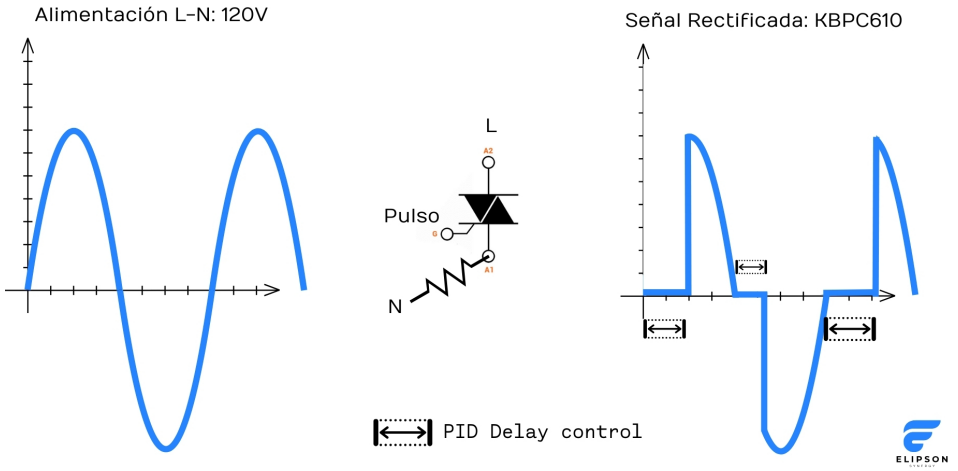


Figure 3.5: Regulación de la temperatura

FeedBack

En el bloque de retroalimentación para el control de la temperatura se cuenta con una termocupla tipo K. Este sensor se acompaña del módulo MAX6675, el cual amplifica la señal del sensor y compensa el efecto de punta fría, logrando así una medición precisa y exacta.

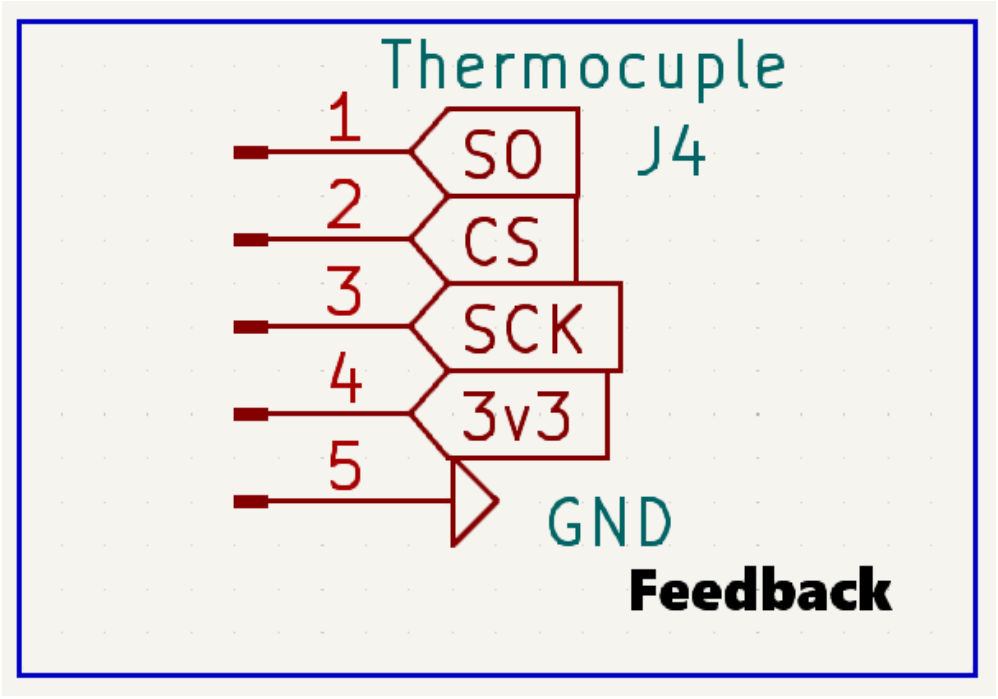


Figure 3.6: TermoCupla: FeedBack

ESP32 Module: Controller

El microcontrolador utilizado en el dispositivos es la ESP32 DEV KIT V1, en el esquemático general se encuentran especificados los pines de conexión de cada uno de los elementos.

UI / Interfaz

Para la manipulación del dispositivo y sus opciones, se cuenta con una interfaz que consta de dos botones, en configuración pull-down, un potenciómetro, un pantalla OLED y un buzzer indicador, encargado de validar al usuario cada una de sus interacciones, con los elementos y el menú.

3.0.2 PCBs

La PCB se encuentra diseñada con un ruteado de pistas en dos capas. Existen dos planos de cobre, siendo de la capa superior, el correspondiente al plano a GND, y el de la capa inferior corresponde al plano de +3.3V. Se cuenta con protecciones, como fusibles, en caso de picos de corriente que pudieran causar daños a la placa. La totalidad de los componentes del diseño se encuentran dispuestos para montura Through-Hole.

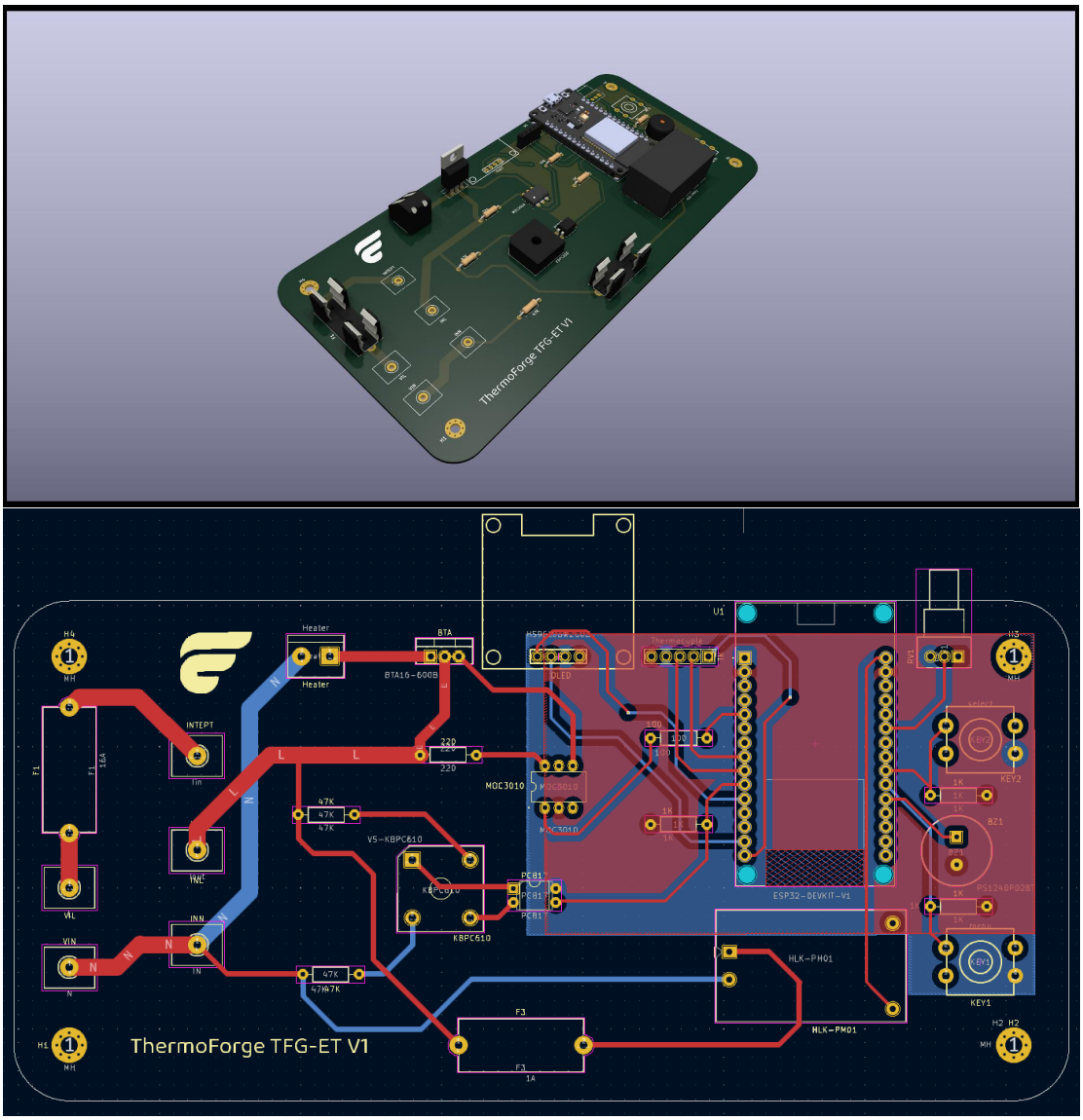


Figure 3.7: Vista 3D y rutas de la PCB

4. PLANOS ESTRUCTURALES.

JANE GOODALL

No se puede pasar un solo día sin tener un impacto en el mundo que nos rodea. Lo que hacemos marca la diferencia, y tenemos que decidir qué tipo de diferencia queremos hacer.

A continuación se presentan los planos Estructurales

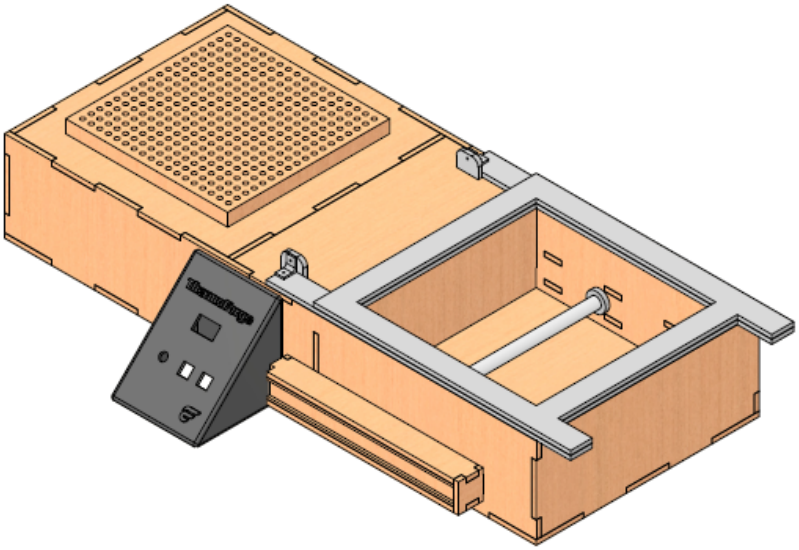


Figure 4.1: Estructura termoformadora

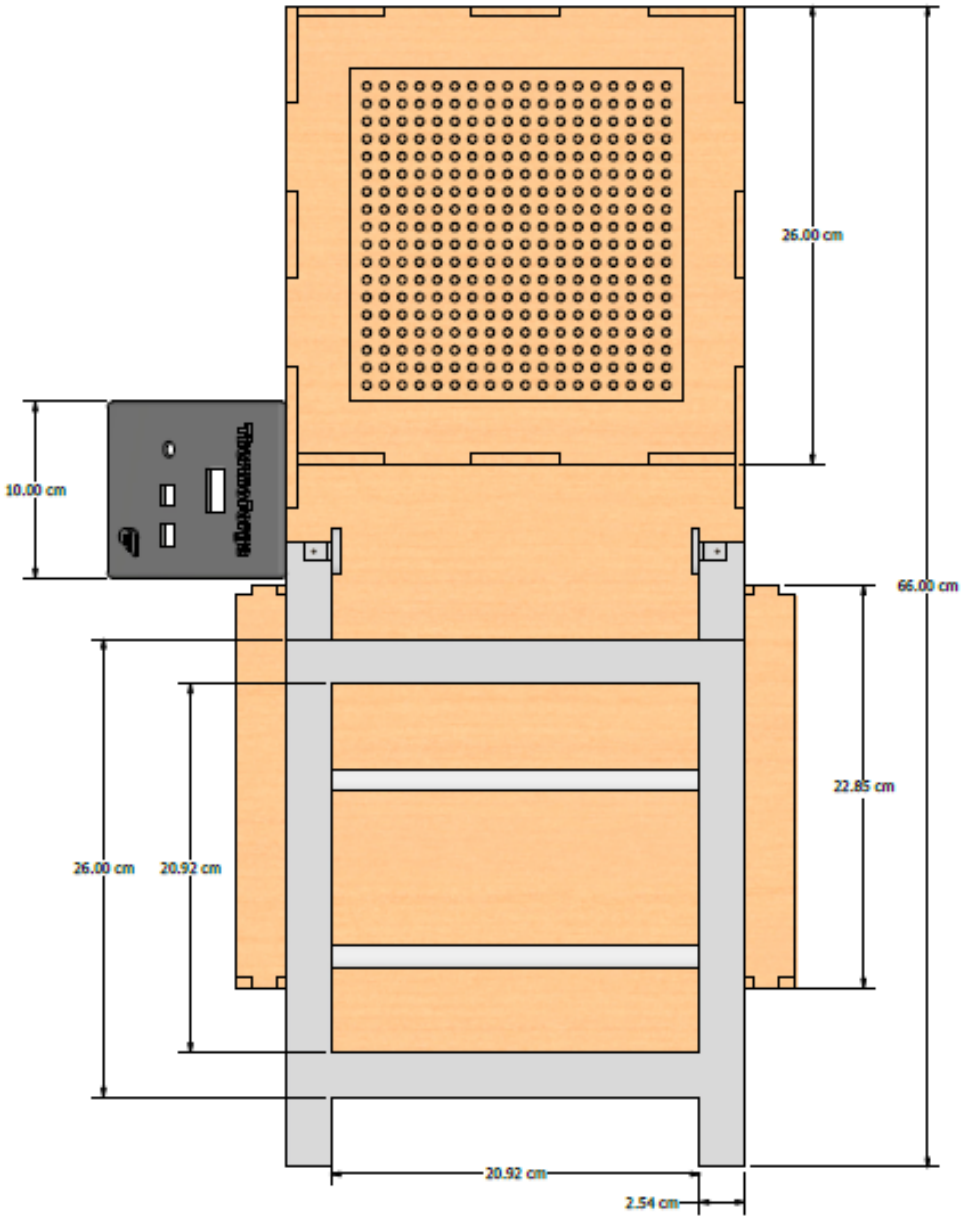


Figure 4.2: Plano superior

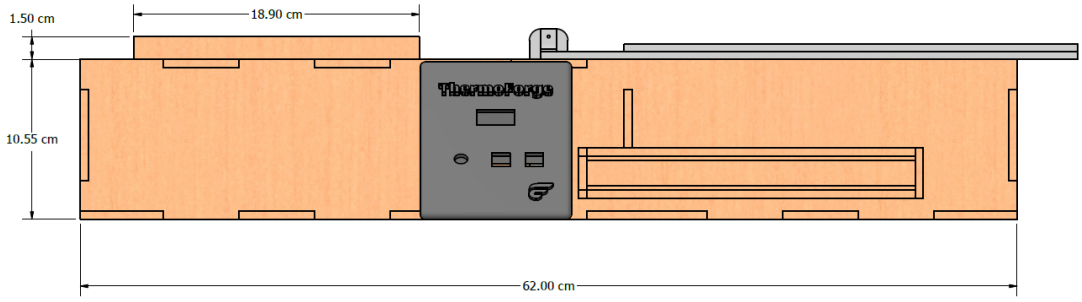


Figure 4.3: Plano frontal

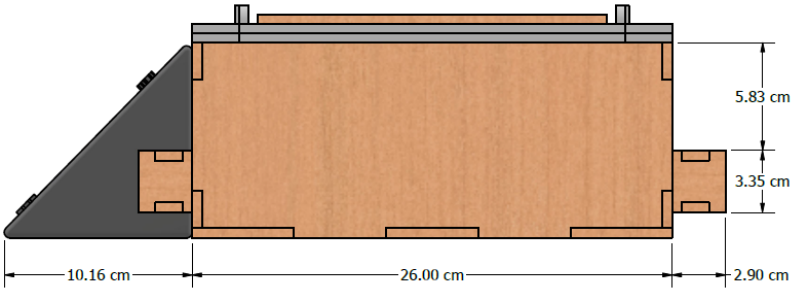


Figure 4.4: Plano lateral

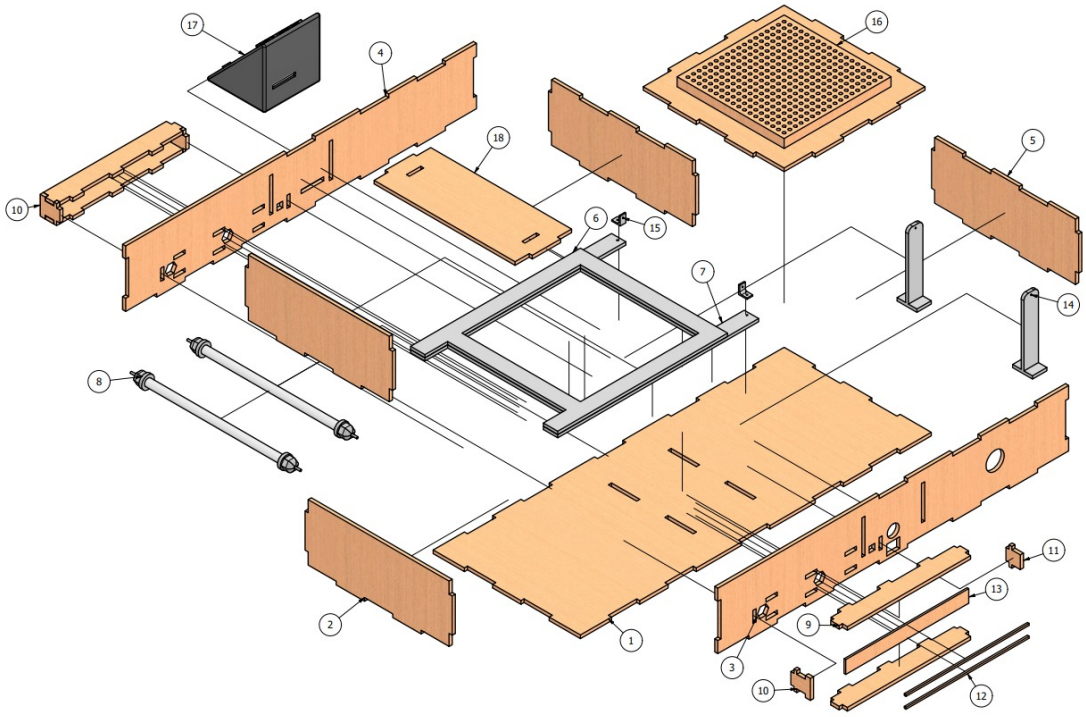


Figure 4.5: Despiece estructura

ID	Cantidad	Nombre	Material
1	1	Base	Madera (MDF 5.5mm)
2	2	Lateral 1	Madera (MDF 5.5mm)
3	1	Lateral 2	Madera (MDF 5.5mm)
4	1	Lateral 3	Madera (MDF 5.5mm)
5	2	Lateral 4	Madera (MDF 5.5mm)
18	1	Cubierta Circuitería	Madera (MDF 5.5mm)
6	1	Soporte Metálico 1	Acero
7	1	Soporte Metálico 1	Acero
8	2	Resistencia Tubular	Cerámica
9	4	Recubrimiento 1	Madera (MDF 5.5mm)
10	2	Recubrimiento 2	Madera (MDF 5.5mm)
11	2	Recubrimiento 4	Madera (MDF 5.5mm)
12	4	Soporte Madera	Madera (MDF 3mm)
13	2	Cubierta Recubrimiento	Madera (MDF 3mm)
14	2	Soporte Metálico 3	Acero
15	2	Ángulo	Acero
16	1	Cubierta para Aspirado	Madera (MDF 5.5mm)
17	1	Interfaz Usuario	PLA

Table 4.1: Piezas estructura termoformadora

5. LISTADO DE COMPONENTES.

Franklin Roosevelt

El único límite para nuestra realización de mañana serán nuestras dudas de hoy

La siguiente tabla muestra los principales elementos necesarios para la construcción del prototipo, organizados por categoría.

Estructura	**Electrónica**
MDF (100 cm x 80 cm)	Fuente 120V a 5V
Lamina de Aluminio (50 cm x 50 cm)	Puente rectificador KBPC610
Platina metálica	Octoacoplador MOC3010
Bisagras	Octoacoplador PC817
Tornillería	Triac BTA16 de 16A/600V
Resistencias tubulares	SwNeon 3P
Pegamento	Pulsador 4 pines con control C
	Potenciómetro
	Perilla plástica
	Resistencia 100k 1/4 W
	Resistencia 47k 1/4 W
	Conector microUSB SMD
	Módulo termocupla max 6675
	Pantalla OLED
	PCB

Table 5.1: Componentes para la Termoformadora

6. PRESUPUESTO.

JANE GOODALL

No se puede pasar un solo día sin tener un impacto en el mundo que nos rodea. Lo que hacemos marca la diferencia, y tenemos que decidir qué tipo de diferencia queremos hacer.

A continuación se presenta el presupuesto para la realización del proyecto.

COSTOS - TERMOFORMADORA							
ESTRUCTURA			ELECTRÓNICA			ENSAMBLE	
ITEM	**MEDIDAS**	**COSTO**	**ITEM**	**CANTIDAD**	**COSTO**	**ITEM**	**COSTOS**
MDF	100 cm x 80 cm	$40.000,00	Fuente 120V a 5V	1	$27.500,00	Soldadura marco	$15.000,00
Lamina de Aluminio	50 cm x 50 cm	$15.000,00	Puente rectificador KBPC610	2	$2.689,00	Corte Láser	$25.000,00
Platina metálica	3 m	$20.000,00	Octoacoplador MOC3010	2	$2.353,00	Pegamento	$16.000,00
Bisagras	9 cm	$5.000,00	Octoacoplador PC817	2	$672,00		
Tornillería	M3	$5.000,00	Triac BTA16 de 16A/600V	2	$3.697,00		
Resistencias tubulares	Largo: 30 cm, Diametro: 1,250 cm	$86.000,00	SwNeon 3P	1	$1.261,00		
			Pulsador 4 pines con control C	4	$3.025,00		
			Potenciómetro	2	$1.681,00		
			Perilla plástica	2	$672,00		
			Resistencia 100k 1/4 W	4	$168,00		
			Resistencia 47k 1/4 W	4	$168,00		
			Conector microUSB SMD	2	$840,00		
			Módulo termocupla max 6675	1	$17.059,00		
			Pantalla OLED	1	$18.400,00		
			PCB	1	$50.000,00		
						SUBTOTAL	$357.017.00
						IVA 19%	$67.833.23
						TOTAL	$424.850.23

7. ANEXOS.

7.0.1 Manual

Se adjunta documento.

7.0.2 Código

El desarrollo del código para la termoformadora se basó en el paradigma de la programación orientada a objetos (POO), lo que permitió estructurar el software en clases y objetos que representan las principales funcionalidades del sistema, facilitando la modularidad y el mantenimiento del programa. Para el control preciso de la temperatura del horno, se implementó un controlador PID, que regula la potencia suministrada al elemento calefactor mediante un triac, en sincronización con un circuito de detección de cruce por cero, asegurando un control eficiente y estable del calor generado. A continuación, se detalla la estructura del código, explicando los principales bloques funcionales y su integración para lograr el desempeño requerido por el sistema.

Para facilitar el acceso y la replicación del proyecto, el código de la termoformadora ha sido alojado en un repositorio de GitHub. Este repositorio incluye toda la documentación necesaria, el esquema del circuito, y los archivos de código organizados, permitiendo a los usuarios explorar, descargar y adaptar el proyecto según sus necesidades. Accede aquí: Para acceder al código, visita nuestro repositorio en ThermoForge - Elipson.

```
#include "Btn.h"
Btn::Btn(int pinNumber, int btntype) : pin(pinNumber), type(
    btntype) {
  if (type == 1) { pinMode(pin, INPUT_PULLUP); }
  else if (type == 0) { pinMode(pin, INPUT); }
}
float Btn::value() { return digitalRead(pin); }
```

Archivo 7.1: Clase Btn para gestión de botones

```
#ifndef Pot_H
#define Pot_H
```

```
#include <Arduino.h>
struct resultados { int lectura; int option; };
class Pot {
  private: int pin; int maxOpciones; static const int
     NUM_MUESTRAS = 10;
  int leerPromediado() { long suma = 0; for (int i = 0; i <
     NUM_MUESTRAS; i++) { suma += analogRead(pin); delay(5); }
     return suma / NUM_MUESTRAS; }
  public: Pot(int pinNumber, int maxOpciones);
  resultados leerOpcion() { int lecturaPromediada =
     leerPromediado(); int option = map(lecturaPromediada, 0,
     1023, 0, maxOpciones - 1); return {lecturaPromediada, option
     }; }
};
#endif
```

Archivo 7.2: Clase Pot para lectura del potenciómetro

```
#include "Screen.h"
Screen::Screen(Adafruit_SSD1306 &oled): oled(oled) {}
void Screen::iniciar() {
    if (!oled.begin(SSD1306_PAGEADDR, 0x3C)) {
        Serial.println(F("No se pudo inicializar la pantalla!"));
        for (;;);
    }
    oled.clearDisplay();
    oled.display();
}
void Screen::mostrarImagen(const uint8_t* Imagen){
    oled.clearDisplay();
    oled.drawBitmap(0, 0, Imagen, SCREEN_WIDTH, SCREEN_HEIGHT,
       WHITE);
    oled.display();
}
void Screen::mostrarTitulo(String titulo, String titulo2 = "",
   int s1=1, int s2=1) {
    oled.clearDisplay();
    oled.setTextSize(s1);
    oled.setTextColor(SSD1306_WHITE);
    oled.setCursor(10, 10);
    oled.println(titulo);
    oled.setTextSize(s2);
    oled.setCursor(10, 30);
    oled.println(titulo2);
    oled.display();
}
void Screen::mostrarOpciones(String temp, String material) {
    oled.clearDisplay();
    oled.setTextSize(2);
    oled.setTextColor(SSD1306_WHITE);
    oled.setCursor(30, 15);
    oled.println(material);
    oled.setCursor(30, 35);
    oled.println(temp);
    oled.drawCircle(78, 38, 3, SSD1306_WHITE);
    oled.setCursor(75,35);
```

```
37      oled.print("␣C");
38      oled.display();
39  }
40  void Screen::progressBar(int valor, int maxValor, String title) {
41      oled.clearDisplay();
42      oled.setTextSize(1);
43      oled.setTextColor(SSD1306_WHITE);
44      oled.setCursor((SCREEN_WIDTH/2)-30, 5);
45      oled.println(title);
46      int barraAnchoMax = SCREEN_WIDTH - 20;
47      int barraAltura = 10;
48      int barraX = 10;
49      int barraY = (SCREEN_HEIGHT / 2) - 10;
50      int barraAncho = map(valor, 0, maxValor, 0, barraAnchoMax);
51      oled.drawRect(barraX, barraY, barraAnchoMax, barraAltura,
            SSD1306_WHITE);
52      oled.fillRect(barraX, barraY, barraAncho, barraAltura,
            SSD1306_WHITE);
53      oled.setCursor((SCREEN_WIDTH/2 - 10), SCREEN_HEIGHT - 20);
54      oled.setTextSize(2);
55      oled.println(valor);
56      oled.display();
57  }
```

Archivo 7.3: Clase Screen para controlar la pantalla OLED

El código define tres objetos principales: *Screen*, *Btn* y *Pot*. El objeto Screen maneja la pantalla OLED, con funciones para inicializarla , mostrar imágenes mapeadas con bits, mostrar títulos y textos en diferentes tamaños, mostrar opciones con texto y gráficos, y mostrar barras de progreso. El objeto *Btn* se encarga de leer el estado de un botón, inicializandolo en un modo de entrada, y proporcionando una función (value) para obtener el valor digital del botón. Finalmente, el objeto *Pot* maneja un potenciómetro, lee el valor de este mediante un promedio de múltiples lecturas (*leerPromediado*), y mapea el valor a un rango específico, devolviendo la lectura y la opción mapeada a través de la función *leerOpcion*.

```
1   #ifndef PID_H
2   #define PID_H
3   #include <Arduino.h>
4
5   /**
6    * @brief Clase que implementa un controlador PID.
7    */
8   class Pid {
9       private:
10      int _numSensors;
11      float setPoint;
12      float maxSat;
13      float minSat;
14      float position;
15      float proportional ;
16      float integral;
17      float derivative;
18      int proportional_past;
19      float control_output;
20      float last_value;
```

```
21      float rateTime;
22
23      public:
24      float Kp;
25      float Ki;
26      float Kd;
27
28      /**
29       * @brief Constructor de la clase Pid.
30       * @param cKp Valor del coeficiente proportional .
31       * @param cKi Valor del coeficiente integral.
32       * @param cKd Valor del coeficiente derivative.
33       * @param cSetPoint Valor del punto de consigna.
34       * @param Vreference Valor de referencia.
35       * @param numSensors N mero de sensores.
36       */
37      Pid(float cminSat, float cmaxSat, int numSensors, float
            crateTime);
38
39      /**
40       * @brief M todo para realizar el seguimiento de la
             posici n.
41       * @param position Posici n actual.
42       * @return Valor de salida del controlador.
43       */
44      float traking(int position, int setPoint);
45      /**
46       * @brief M todo para calcular el error.
47       * @param sensors_values Valores de los sensores.
48       * @return Valor del error calculado.
49       */
50      int calculateError(int* sensors_values);
51      float* tuning(float Kp, float T, float L, int method);
52  };
53  #endif
```

Archivo 7.4: Código de la clase PID

Este código implementa un controlador PID (Proporcional, Integral, Derivativo) que se utiliza para ajustar y mantener la temperatura del horno cerca de un valor deseado, llamado setpoint. El controlador PID se ajusta a la señal de error, que es la diferencia entre el valor actual de la variable y el valor deseado.

```
1  #include <Arduino.h>
2  #include <pins.h>
3  #include <max6675.h>
4  #include "imagenes.h"
5  #include "Screen.h"
6  #include "Pot.h"
7  #include "Btn.h"
8  #include "Pid.h"
9
10 MAX6675 thermocouple(thermoCLK, thermoCS, thermoDO);
11 Screen screen(display);
12 Btn enter(selectBtn, 1); Btn menu(menuBtn, 1);
```

```
Pot potenciometroMenu(spinBtn, cant+2); Pot potenciometroTemp(spinBtn
    , 401);

const char *materiales[] = {"BYPACK", "PE", "PP", "ABS", "PVC", "PET"
    , "PS", "PC"};
int temperaturas[] = {160, 140, 170, 185, 155, 240, 105, 175};
int cant = sizeof(materiales) / sizeof(materiales[0]);

float max_firing_pulse = (1/60)*10^3;

int minTem = 1000; int maxTem= 2000;
int _numsensors = 1; float rateTime = 1;
Pid pid(minTem, maxTem, _numsensors, rateTime);

Btn zeroCross(zeroCrossing, 0);

bool zero_cross_detected() {
  return !zeroCross.value();
}

void control(float maximum_firing_delay, float setPoint) {
  float temp = thermocouple.readCelsius();
  float PID_value = pid.traking(temp, setPoint);
  int zero_cross = zero_cross_detected();

  if (zero_cross)  {
    delayMicroseconds(maximum_firing_delay - PID_value);
    digitalWrite(triac, HIGH);
    delayMicroseconds(100);
    digitalWrite(triac, LOW);
    zero_cross = false;
  }
}

int opcionActual = 999;
int tempActual = 0;
int tempRead;
bool seleccionMaterial = true;
resultados pot;
resultados pot2;
String text;

void setup() {
  Serial.begin(9600);
  screen.iniciar();
  screen.mostrarImagen(HOME);
  delay(5000);
  pinMode(triac, OUTPUT);
}

void loop() {
  while (enter.value() && opcionActual == 999) {
    pot = potenciometroMenu.leerOpcion();
    if (pot.option < cant){
        screen.mostrarOpciones(String(temperaturas[pot.option]),
            String(materiales[pot.option]));
```

```
66      }
67      else {
68          screen.mostrarOpciones("␣T␣", "Elegir");
69      }
70    }
71    opcionActual = pot.option;
72    if (opcionActual >= cant && tempActual != 0){
73      delay(100);
74      while (enter.value() && tempActual == 0) {
75        resultados pot2 = potenciometroTemp.leerOpcion();
76        screen.progressBar(pot2.option, 400, "Temperatura");
77        tempRead = pot2.option;
78      }
79      tempActual = tempRead;
80    }
81    else{
82      tempActual = temperaturas[pot.option];
83    }
84    screen.mostrarTitulo("Termoformando␣a␣" + String(tempActual), "",
        1, 2);
85    control(max_firing_pulse, tempActual);
86  }
```

Por último, tenemos el código principal, este código implementa un sistema de control para una termoformadora, utilizando un sensor de temperatura MAX6675, un controlador PID para regular la temperatura, y un triac para controlar el calentamiento del elemento. El código también incluye una pantalla OLED para mostrar las opciones de material y temperatura seleccionadas mediante un potenciómetro, y botones para navegar entre las opciones.

1. Inicialización de sensores y dispositivos: Se inicializan los objetos para la pantalla (Screen), el termopar MAX6675 para medir la temperatura, y varios botones y potenciómetros para la selección del material y la temperatura de operación.
2. Selección de material y temperatura: Los usuarios pueden seleccionar el material y la temperatura de operación a través de un potenciómetro. El programa muestra las opciones en una pantalla OLED y ajusta la temperatura objetivo según el material seleccionado.
3. Control PID de temperatura: El controlador PID ajusta la temperatura del sistema, tomando como entrada la lectura del termopar MAX6675 y comparándola con la temperatura objetivo (setPoint). El valor de salida del PID se utiliza para ajustar la señal de control del triac, regulando así el calentamiento.
4. Manejo del cruce por cero: El sistema utiliza un detector de cruce por cero (una técnica común para controlar el encendido y apagado de triacs) para sincronizar la activación del triac con la señal de la corriente alterna (AC). Esto garantiza que el triac se active en el momento adecuado durante el ciclo de corriente.
5. Interacción del usuario: El código permite la interacción mediante botones para elegir las opciones de material y temperatura. Si se selecciona una opción especial (por ejemplo, "T" para elegir temperatura), el sistema permite que el usuario ajuste la temperatura con un segundo potenciómetro.