



Tecnológico de Monterrey

Tecnológico de Monterrey
(Campus Guadalajara)

Ingeniería en Robótica y Sistemas Digitales (IRS)
Diseño de sistemas embebidos avanzados (Gpo. 501)
TE 2004B.501

Reporte Final Smart Heater

Profesor de reto:
Gilberto Ochoa Ruiz

Integrantes:

Alexis Gibrán Acosta Pánuco - A01639818

Eliás Uriel Velázquez Rojas - A01639716

Fernando Cerriteño Magaña - A01702790

Misael Octavio Rodríguez Macías - A01639786

Emiliano Martínez Aguilar - A01352482

Fecha:

28 de Noviembre del 2022

Índice

Resumen	3
Objetivos	3
Especificaciones	4
Requisitos	4
Introducción	5
Marco Teórico	6
Sensores, Materiales, etc.	6
Funcionamiento de los componentes	6
Desarrollo de la práctica (Caso de Prácticas):	7
Diseño de la planta (Pasos)	8
Código del sistema	10
Diseño y simulación:	12
Controlador On/Off	12
Video del funcionamiento de la planta (Velocidad x5)	13
Ecuación de temperatura del sistema	13
Conclusiones Individuales	14
Fernando	14
Alexis Gibran	15
Misael Octavio	15
Elías Uriel	16
Emiliano Martinez	16
Conclusión General	17
Apéndices	17
Apéndice 1 (Material y componentes)	17
Apéndice 2 (Código del sistema)	17
Referencias	18

Resumen

El presente documento consiste en el diseño de un sistema de temperatura inteligente, conformado por una planta del sistema, una PCB y otros mecanismos necesarios para el ajuste de temperatura. El presente sistema utiliza un peltier para la regulación de temperatura, adicionalmente se utiliza una PCB, componentes electrónicos básicos, un sensor de temperatura, entre otros materiales para elaborar el sistema.

También se presentan los contenidos teóricos de los circuitos, la conexión de los componentes y el desarrollo matemático de los cálculos de temperatura. Además, se presenta el desarrollo de prácticas (casos de prueba) de la teoría plasmada para la elaboración de los circuitos y utilizando diferentes softwares para el manejo del controlador y la planta.

Finalmente, se presenta el diseño de control del sistema, al igual que las simulaciones correspondientes, junto con los resultados obtenidos y diferentes análisis realizados en base a estos. Concluyendo con comentarios acerca de las problemáticas que se encontraron, aspectos a mejorar y opiniones en general.

Objetivos

Implementar un sistema inteligente de temperatura, conformado por una planta, sensores y una PCB para su correcto funcionamiento, para que los estudiantes pongan en práctica lo aprendido durante el curso y trabajen en equipo, manteniendo una actitud positiva y trabajando de manera colaborativa para la implementación del sistema deseado.

Desarrollar las subcompetencias establecidas, STE0101 (Modela sistemas embebidos) y STE0102 (Implementa sistemas embebidos) para desarrollar sistemas embebidos cumpliendo con normas de calidad, seguridad y desempeño, STE0202 (Selecciona los periféricos) para desarrollar los componentes de inteligencia que le permiten a un robot ser autónomo en la solución de problemas y SEG0302 (Colaboración) para generar entornos efectivos de colaboración y negociación en contextos multiculturales, con respeto y aprecio por la diversidad de personas, saberes y culturas.

Especificaciones

El sistema consta de los siguientes componentes electrónicos:

- **Celda Peltier Tec112710**
Voltaje: 12 Vcc nominal / 15,4 Vcc máx.
Temperatura de operación: -50°C a 150°C.
Costo: \$200
- **Puente H L298N**
Se usará para cambiar la polaridad en el peltier.
Costo: \$75
- **Sensor de temperatura Ds18b20**
Mide temperatura desde -55°C hasta +125°C, con un rango de error de ±0.5°C.
Rango de Voltaje: -0.5V a +6.0V.
Costo: \$140
- **Placa FRDM - KL25Z**
32-bit ARM Cortex-M0+ core running at 48MHz.
It includes 128KB FLASH, 16KB RAM.
Costo: \$345
- **Potenciómetro**
El potenciómetro se usará para poder determinar la temperatura deseada.
Costo: \$10
- **Display 2x16**
Display de 16 caracteres y 2 filas que se usará para notificarle al usuario sobre la temperatura actual.
Costo: \$100
- **Buzzer**
Buzzer que sonara para notificarle al usuario si el dispositivo detecta una posible falla.
Costo: \$30
- **Sensor de voltaje**
Se encarga de medir el voltaje suministrado y de avisar si llega a ver un sobrevoltaje .
Costo: \$30

Requisitos

Entre los requisitos que se necesitan tener para el funcionamiento correcto del sistema se tiene que:

- El dispositivo debe ser capaz de leer la temperatura dentro de la cabina.

- El dispositivo debe de ser capaz de leer al menos una temperatura mínima de -40°C y una temperatura máxima de 60°C.
- El dispositivo cuenta con un error de lectura máximo de $\pm 3^{\circ}\text{C}$.
- El sensor de temperatura debe de dar retroalimentación al dispositivo.
- La retroalimentación del sensor de temperatura debe de ser usada para realizar una acción que enfríe o caliente a la cabina.
- El dispositivo debe ser capaz de notificar al usuario la temperatura actual.
- El uso de un sensor auditivo o táctil debe ser usado para notificar de una posible falla en el dispositivo.

Introducción

Diseñar sistemas de control tiene principalmente dos problemas o dificultades, la primera es encontrar un modelo matemático de la planta que se ajuste con gran exactitud al comportamiento del sistema real y la segunda es diseñar un controlador estable para el modelo de la planta. Básicamente, se trata de la implementación matemática en base al comportamiento real, por lo que la calidad del sistema de control se basará en la exactitud de la implementación matemática teórica. [2]

Para ello se debe de implementar un sistema que regule la temperatura de una planta la cual debe de medir la temperatura en la que se encuentra la planta en ese instante y sea si este muy baja aumentar la temperatura de dentro o si es muy alta la temperatura actual se tenga que bajar hasta regularse en el rango acordado, y por último dar una retroalimentación sobre los últimos datos recabados por los sensores .

Durante estas diez semanas estuvimos trabajando con el proyecto de sistemas embebidos avanzados, en el cual simulamos una cabina de tractor donde controlamos la temperatura a la que está sometida ya sea para enfriamiento o calentamiento de la cabina. Teniendo como objetivo implementar un sistema de temperatura inteligente con los diferentes componentes que necesitábamos para la realización de este proyecto. Para ello se realizó una investigación acerca de los diferentes componentes que conforman el sistema, su teoría y funcionamiento, además de los problemas que encontramos entre la unión de los componentes.

El principal objetivo de este documento es documentar todo el proceso teórico y práctico necesario para la implementación de un sistema de temperatura inteligente, estableciendo los principios científicos, simulaciones, procedimientos matemáticos y prácticas físicas con circuitos y el sistema real, además de analizar los resultados obtenidos para llegar a una conclusión.

Marco Teórico

- **Sensores, Materiales, etc.**

Debido a las altas temperaturas a las que se puede llegar en las áreas de trabajo, es importante contar con un sistema el cual ayude con la nivelación dentro de la temperatura, ya sea que enfrié si la cabina se encuentra caliente, o que caliente si la cabina se encuentra helada. Es este sistema el cual se piensa replicar con sistemas de control en un ambiente controlado, con el propósito de ver que tan efectivo resultaría este sistema y será alimentado por una placa kl25z y baterías para generar el voltaje diferente utilizándolo como un microcontrolador y generando un código para poder alimentar las celdas, sensores y actuadores que utilizaremos.

Para la planta del sistema se planean usar materiales, los cuales son más fáciles de obtener, más baratos, y más fáciles de usar, debido a que el objetivo es obtener el control del sistema. Para la cabina se utilizará un tupper hermético de 6.5L, el cual cuenta con medidas de 29.5 x 13 x 22.5 centímetros, ya que de esta forma se tiene una cabina transparente la cual nos garantiza un sello hermético para la temperatura dentro del contenedor, para obtener la temperatura dentro de la cabina, se usará un sensor de temperatura LM35 posicionado en la esquina de la “cabina” con la información del sensor, se obtendrá un promedio de la temperatura y se determinará la acción que la celda de peltier hará. Además de un sensor de voltaje que se encargará de medir el voltaje suministrado a la planta y alertará si existe un sobrevoltaje para los demás sensores.

- **Funcionamiento de los componentes**

El sensor de temperatura Lm35 es un sensor analógico el cual nos permite medir temperaturas desde -50 grados Centígrados hasta 150 grados Centígrados, se lee de forma analógica y se implementa la ecuación otorgada por el fabricante para obtener la temperatura.

Para el manejo de la temperatura se usará una celda peltier, la cual estará posicionada en un lado de la “Cabina” esta celda será nuestro actuador, debido a que tiene la capacidad de enfriar de uno de los dos lados o viceversa, dependiendo de la polaridad, con este actuador se nos hará posible enfriar o calentar la cabina para mantenerla a una temperatura de aproximadamente 21°C, “Actuando como una bomba de calor permitiendo de esta forma transmitir el calor de un foco frío a uno caliente”, puesto que básicamente cuenta con dos placas por cada lado con semiconductores dentro de estas para realizar el proceso de refrigeración, los semiconductores suelen ser boro, indio y galio para el tipo P y fosforo, arsenico y antimonio para el tipo N, al pasar corriente entre los semiconductores logra enfriar una placa dependiendo hacia donde circule, todo el calor que pierde esta placa es absorbido por la segunda, un funcionamiento realmente sencillo de entender, de esta forma un lado de la celda se enfriará y el otro se calentará, aunque actualmente existen diferentes sistemas más eficientes para generar calor al nosotros enfocarnos un poco más en el tema de enfriamiento de la cabina al esta simular un tractor que pasa bastante tiempo al sol, ademas de contar con la ventaja de que la celda no es dependiente de la temperatura ambiente, no obstante también al requerir en nuestro caso enfriar la cabina con la que trabajaremos utilizaremos un ventilador para poder permitir un mejor flujo de aire ya sea caliente o frío para poder regular la temperatura de la cabina en el menor tiempo posible. [1]

Finalmente para la inversión de la polaridad en la celda peltier, se pretende utilizar un puente H, utilizando cuatro transistores para controlar la dirección de la corriente que circula por el motor. Para que el motor gire hacia delante o hacia atrás, sólo se pueden encender dos transistores a la vez, de esta forma podremos enfriar la cabina por el interior y sacar el calor por fuera de esta, al nosotros invertir la polaridad podremos hacer que la placa de peltier cambie igual su funcionamiento calentando la parte del interior, y enviándolo hacia afuera. Al usar este dispositivo eléctrico podemos dar poder a grandes elementos eléctricos que requieren de gran corriente al solo alimentarlo con poca corriente.

Desarrollo de la práctica (Caso de Prácticas):

En las áreas de trabajo se pueden llegar a presentar altas temperaturas por eso es importante contar con un sistema que ayude a equilibrarla enfriando si la planta está caliente o al contrario calentando si la planta está fría. El sistema pretende replicar un entorno controlado mediante el uso de sistemas de control.

- **Diseño de la planta (Pasos)**

El primer paso que hicimos pensar en un diseño que nos pueda llegar a funcionar para que la temperatura de la planta pueda ser controlado con componentes junto al a tarjeta k125z, entonces pensamos en el peltier como el componente que nos ayudaría a poder calentar o enfriar este trabajaría en conjunto del sensor de temperatura, en base a la temperatura controlarlo con el peltier el sensor de temperatura lo controlaremos con programación con la tarjeta K125Z y con un puente H la polaridad del peltier.

El siguiente paso fue diseñar un bosquejo conforme al sistema que queríamos realizar, como el siguiente, con los componentes, la tarjeta K125Z, el sensor de temperatura, el peltier y un ventilador, junto las conexiones.

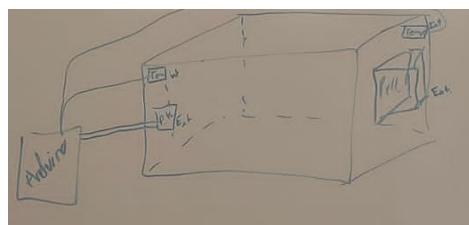


Figura 1. Planeación inicia con arduino de la planta

Después de eso realizamos el circuito en tinkerCAD para intentar simular, lo que teníamos planeado (simulamos que el monitor serial, como una pantalla y el peltier como un sensor, el motor reductor simula un ventilador para enfriar el peltier y que tenga un mejor funcionamiento).

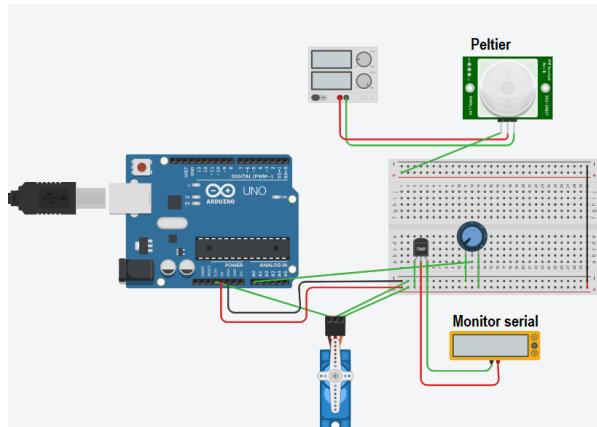


Figura 2. Diseño del circuito

de la planta

Posteriormente se consiguieron los materiales indicados para la realización de la planta, el siguiente paso fue preparar la caja para la implementación de los componentes, hicimos orificios en la parte frontal y al costado para después cubrir con cinta anti aislante en las posibles fugas.



Figura 3. Primer modelo de la planta

Luego de hacer los respectivos orificios en la caja, metemos el peltier en su hoyo correspondiente, pegando el ventilador a la parte trasera del peltier para una mayor ventilación de la temperatura a su vez de insertar el potenciómetro en su hoyo del costado.



Figura 4. Colocación del peltier en la planta

Después se diseñó el cableado de la PCB para poder conectar todo de forma mas eficiente, conectamos el Puente H al peltier y a la vez se instaló el sensor de temperatura en el interior para una mejor toma de está y se conectan los respectivos pines al LCD.

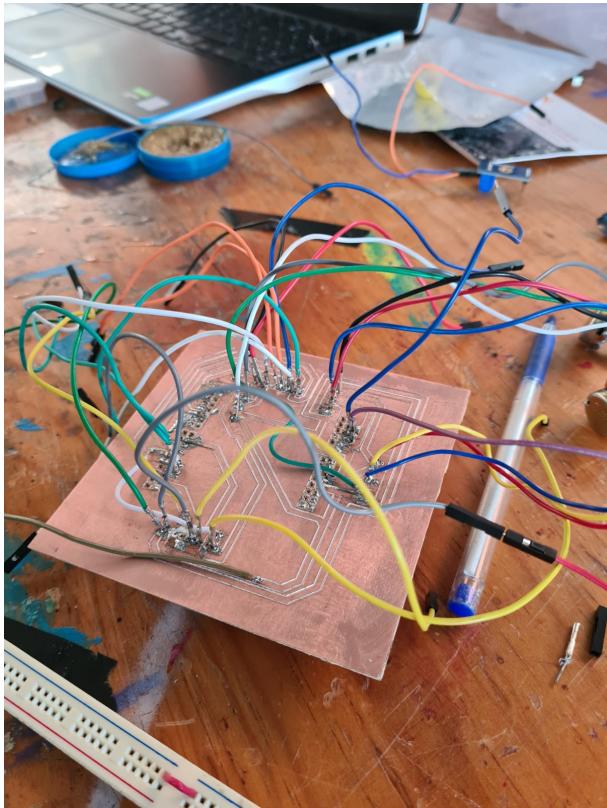


Figura 5. Conexiones de la planta

Al finalizar con la preparación de las plantas con los componentes físicamente, usamos el software Mbed para el funcionamiento del sistema creamos un programa en mbed para así poder controlar la KI25Z, que a su vez alimenta a diversos dispositivos, tales como los sensores de temperatura, potenciómetro, sensores de voltaje Puente H, LCD, etc....

- **Código del sistema**

```
#include "mbed.h"
#include <MKL25Z4.h>
#include <string>
#include <iostream>

// main() runs in its own thread in the OS
#define RS 0x04 /* PTA2 Pin */
#define RW 0x10 /* PTA4 Pin */
#define EN 0x20 /* PTA5 Pin */

AnalogIn pot(PTB1); /* Potentiometer middle pin connected to P0_11, other two ends
connected to GND and 3.3V */
AnalogIn TempSensor(PTB0);
AnalogIn Fz(PTB2);
AnalogIn swch(PTB3);
DigitalOut led(LED1); /* LED blinks with a delay based on the analog input read */
```

```

DigitalOut IN1(PTE31); //Pines del puente H
DigitalOut IN2(PTA17);

/*Functions*/
void delayMs(int n);

void LCD_command(unsigned char command);
void LCD_data(unsigned char data);
void LCD_init(void);

void LCD_set(float n);
void LCD_displayWish(float n, int ms);
void LCD_displayTemp(float n, int ms);

void giroStop();
void giroIzq();
void giroDer();

void holdOn();

void vRead();

int sensorValue;
float value; //Variable para FZ0340
int over = 0;

/*Main function*/
int main(void){
    LCD_init();
    int chk = 0;
    float ain, temporal, wished_temp; // Variables to store the analog input
    float f; // Variables for temperature
    ain = pot.read(); /* Read analog value (output will be any value between 0 and 1 */
    while (1) {
        vRead();
        if(over == 1){
            while(over){
                LCD_command(1); /* clear display */
                LCD_command(0x80); /* set cursor at first line */
                delayMs(500);
                giroStop();
                LCD_data('O');
                LCD_data('v');
                LCD_data('e');

                LCD_data('r');
                LCD_data('l');
                LCD_data('O');
                LCD_data('a');
            }
        }
    }
}

```

```

LCD_data('d');
delayMs(1000);
vRead();

}

}

float statusON = swch.read();
if(statusON < 0.5){
    holdOn();
}
else{
    f = (5.0 * (TempSensor.read() * 690) * 100.0)/1024.0; //Read Temperature
    if (chk == 0){
        LCD_displayTemp(f,2500);
        chk++;
    }
    temporal = ain*100; //Variable temporal para la lectura de la temperatura
deseada
    ain = pot.read(); /* Read analog value (output will be any value between 0 and 1
*/
    wished_temp = (ain*100);
    if((int)temporal != (int)(ain*100)){
        LCD_displayWish(wished_temp, 1000);
    }
    if(f < wished_temp - 2 && over == 0){
        vRead();
        if (over == 1) {

            break;
        }
        giroIzq();
        while(f < wished_temp - 2 && over == 0){
            vRead();
            if (over == 1) {
                break;
            }
            while(swch.read()<0.5){
                holdOn();
            }
            LCD_command(1); /* clear display */
            LCD_command(0x80); /* set cursor at first line */
            delayMs(500);
            LCD_data(' ');
            LCD_data(' ');
            LCD_data(' ');
            LCD_data(' ');
            LCD_data('H');
            LCD_data('e');
            LCD_data('a');
            LCD_data('t');

```



```

LCD_data('e');
delayMs(1000);
}
else if(f > wished_temp + 2 && over == 0){
    vRead();
    if (over == 1) {
        break;
    }
    while(swch.read()<0.5){
        holdOn();
    }
    giroDer();
    while(f > wished_temp + 2 && over == 0){
        vRead();
        if (over == 1) {
            break;
        }
        while(swch.read()<0.5){
            holdOn();
        }
        LCD_command(1); /* clear display */
        LCD_command(0x80); /* set cursor at first line */
        delayMs(500);
        LCD_data(' ');
        LCD_data(' ');
        LCD_data(' ');
        LCD_data(' ');
        LCD_data('C');
        LCD_data('o');
        LCD_data('o');
        LCD_data('l');
        LCD_data('i');
        LCD_data('n');
        LCD_data('g');
        delayMs(1000);
        vRead();
        if (over == 1) {
            break;
        }
        while(swch.read()<0.5){
            holdOn();
        }
        f = (5.0 * (TempSensor.read() * 690) * 100.0)/1024.0; //Read Temperature
        LCD_displayTemp(f, 1000);
        vRead();
        if (over == 1) {
            break;
        }
        while(swch.read()<0.5){
            holdOn();
        }
    }
}

```

```

        }
        temporal = ain*100; //Variable temporal para la lectura de la temperatura
deseada
        ain = pot.read(); /* Read analog value (output will be any value between 0
and 1 */
        wished_temp = (ain*100);
        LCD_displayWish(wished_temp, 1000);
        vRead();
        if (over == 1) {
            break;
        }
        while(swch.read()<0.5){
            holdOn();
        }
    }
    giroStop();
    LCD_command(1); /* clear display */
    LCD_command(0x80); /* set cursor at first line */
    delayMs(500);
    LCD_data(' ');
    LCD_data('D');
    LCD_data('o');
    LCD_data('n');
    LCD_data('e');
    delayMs(1000);
}
}

void holdOn(){
    giroStop();
    LCD_command(1); /* clear display */
    LCD_command(0x80); /* set cursor at first line */
    delayMs(500);
    LCD_data('O');
    LCD_data('n');
    LCD_data(' ');
    LCD_data('h');
    LCD_data('o');
    LCD_data('l');
    LCD_data('d');
    wait_us(500000);
}

```

```

/* initialize and configures the LCD on the FRDM board */
void LCD_init(void){
    SIM->SCGC5 |= 0x1000; /* enable clock to Port D */

    PORTD->PCR[0] = 0x100; /* make PTD0 pin as GPIO */
    PORTD->PCR[1] = 0x100; /* make PTD1 pin as GPIO */
    PORTD->PCR[2] = 0x100; /* make PTD2 pin as GPIO */
    PORTD->PCR[3] = 0x100; /* make PTD3 pin as GPIO */
    PORTD->PCR[4] = 0x100; /* make PTD4 pin as GPIO */
    PORTD->PCR[5] = 0x100; /* make PTD5 pin as GPIO */
    PORTD->PCR[6] = 0x100; /* make PTD6 pin as GPIO */
    PORTD->PCR[7] = 0x100; /* make PTD7 pin as GPIO */

    PTD->PDDR = 0xFF; /* make PTD7-0 as output pins */

    SIM->SCGC5 |= 0x0200; /* enable clock to Port A */
    PORTA->PCR[2] = 0x100; /* make PTA2 pin as GPIO */
    PORTA->PCR[4] = 0x100; /* make PTA4 pin as GPIO */
    PORTA->PCR[5] = 0x100; /* make PTA5 pin as GPIO */
    PTA->PDDR |= 0x34; /* make PTA5, 4, 2 as out pins*/
    delayMs(30);

    /* initialization sequence */
    LCD_command(0x38);
    delayMs(1);
    LCD_command(0x01);/* Clear screen */

    LCD_command(0x38);/* Set 8-bit data, 2-line, 5x7 font */
    LCD_command(0x06);/* Move cursor right */
    LCD_command(0x01);/* Clear screen, move cursor to home */
    LCD_command(0x0F);/* Turn on display, cursor blinking */
}

/* Sends command to the LCD*/
void LCD_command(unsigned char command){
    PTA->PCOR = RS | RW; /* RS = 0, R/W = 0 */
    PTD->PDOR = command;
    PTA->PSOR = EN; /* pulse E */

    delayMs(0);
    PTA->PCOR = EN;

    if (command < 4)
        delayMs(4); /* command 1 and 2 needs up to 1.64ms */
    else
        delayMs(1); /* all others 40 us */
}

```

```

/* Sends data to the LCD*/
void LCD_data(unsigned char data){

    PTA->PSOR = RS; /* RS = 1, R/W = 0 */

    PTA->PCOR = RW;

    PTD->PDOR = data;

    PTA->PSOR = EN; /* pulse E */

    delayMs(0);
    PTA->PCOR = EN;
    delayMs(1);

}

/* Delay n milliseconds */
/* The CPU core clock is set to MCGFLLCLK at /* 41.94 MHz in SystemInit(). */
void delayMs(int n) {
    int i;
    SysTick->LOAD = 41940 - 1;
    SysTick->CTRL = 0x5; /* Enable the timer and choose sysclk as the clock source */

    for(i = 0; i < n; i++) {
        while((SysTick->CTRL & 0x10000) == 0)
            /* wait until the COUTN flag is set */
            {}
    }
    SysTick->CTRL = 0;
    /* Stop the timer (Enable = 0) */
}

void LCD_set(float n){
    string floatString = to_string(n);
    char temp[floatString.length() + 1];
    strcpy(temp, floatString.c_str());
    for (int i = 0; i<floatString.length(); i++){
        LCD_data((unsigned char)temp[i]);
    }
    delayMs(1000);
}

void LCD_displayTemp(float n, int ms){
    LCD_command(1); /* clear display */
    LCD_command(0x80); /* set cursor at first line */
    delayMs(500);
    LCD_data('C');
    LCD_data('u');
    LCD_data('r');
    LCD_data('r');
}

```

```

LCD_data('e');
LCD_data('n');
LCD_data('t');
LCD_data(' ');
LCD_data('T');
LCD_data('e');
LCD_data('m');
LCD_data('p');
LCD_data('.');
LCD_data(' ');
LCD_command(0xC0);
LCD_data(' ');
LCD_data(' ');
LCD_set(n);
delayMs(ms);
}

void LCD_displayWish(float n, int ms){
    LCD_command(1); /* clear display */
    LCD_command(0x80); /* set cursor at first line */
    delayMs(500);
    LCD_data('W');
    LCD_data('i');
    LCD_data('s');
    LCD_data('h');
    LCD_data('e');
    LCD_data('d');
    LCD_data(' ');
    LCD_data('T');
    LCD_data('e');
    LCD_data('m');
    LCD_data('p');
    LCD_data('.');
    LCD_data(' ');
    LCD_command(0xC0);
    LCD_data(' ');
    LCD_data(' ');
    LCD_set(n);
    delayMs(ms);
}

void giroIzq(){ //Calienta Adentro de la planta
    IN1 = 0;
    IN2 = 1;
    return;
}

void giroDer(){ //Enfria Adentro de la planta
    IN1 = 1;
    IN2 = 0;
}

```

```

    return;
}

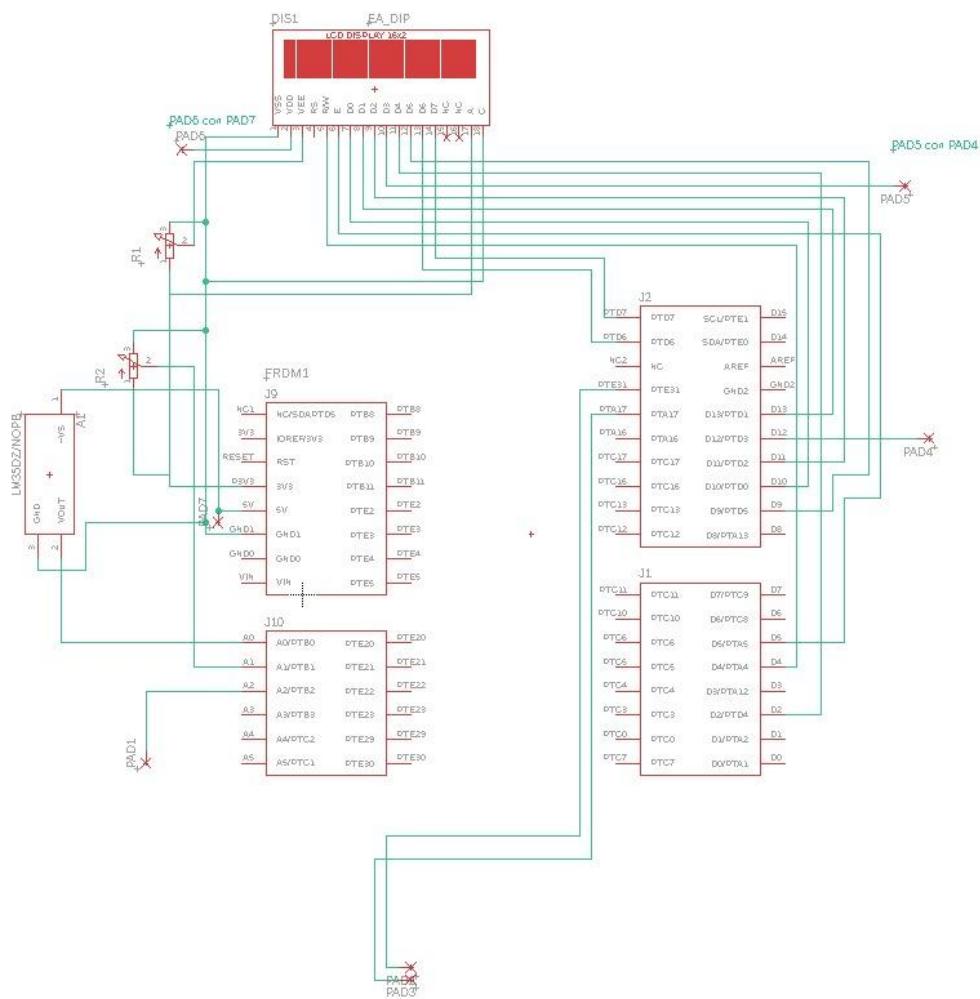
void giroStop(){
    IN1 = 0;
    IN2 = 0;
    return;
}

void vRead(){
    over = 0;
    if((int)(Fz.read()*100) == 67){
        value = 11;
    }
    else{
        value = (11*Fz.read())/0.677;
    }/if((int)(Fz.read()*100);
    if(value>12.50){
        giroStop();
        over = 1;
    }
    return;
}

```

Diseño y simulación:

Para darnos una mejor idea sobre las conexiones iniciales con las que contamos realizamos un diseño esquemático que representa los sensores y la tarjeta con la que controlamos los demás dispositivos



Para lograr un control de voltaje para de esta forma evitar algún sobrevoltaje o ver que se le esté suministrando el voltaje correspondiente se utilizó el sensor de voltaje Fz0430 para de esta forma nos está mandando señales sobre el voltaje suministrado a través del tiempo se plantea que al momento de no recibir el voltaje correspondiente alerte al usuario así como al alertar sobre un voltaje lograr un apagado ante los elementos afectados.

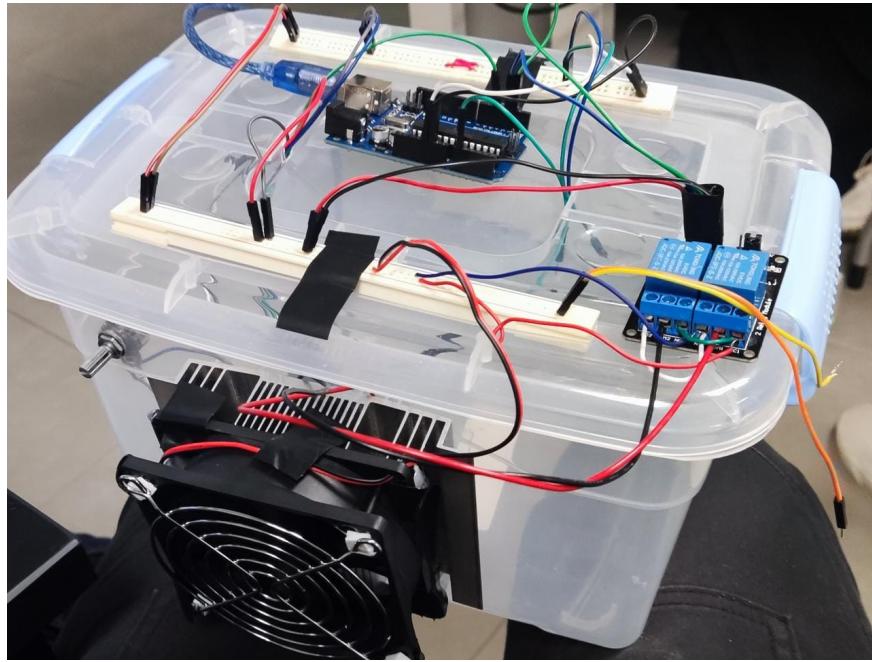


Figura 6. Planta del sistema de enfriamiento

Al poner en prueba el sistema, se observó un comportamiento en particular, el sistema es lento, esto puede ser afectado debido a fugas alrededor del contenedor las cuales impiden que el sistema cuente con un sello hermético.

- **Videos del funcionamiento de la planta**

<https://drive.google.com/drive/folders/1bGm3bQG6YEhvthxSCxJeFBjf1m3uh4f3?usp=sharing>

- **Ecuación de temperatura del sistema**

Debido a que en nuestro caso la planta simula un sistema cerrado, se puede utilizar la fórmula de Balance de Energía en un Sistema Cerrado, tomando en cuenta esto, en un sistema cerrado los términos de generación y consumo del balance general se por lo cual: [3]

$$[Entrada - Salida]_{a \text{ través de las fronteras}} = [Acumulación]_{\text{dentro del sistema}}$$

$$Energía neta transferida_{a \text{ través de las fronteras}} = Energía final - Energía inicial$$

Y dentro de las definiciones de Balance de Energía se define así:

$$Energía neta transferida = Q - W$$

$$\text{Energía inicial} = U_i + Ec_i + Ep_i \quad \text{Energía final} = U_f + Ec_f + Ep_f$$

$$Q - W = (U_f - U_i) + (Ec_f - Ec_i) + (Ep_f - Ep_i)$$

$$Q - W = \Delta U + \Delta Ec + \Delta Ep$$

- $\Delta Ec = 0$, es cero ya que no hay aceleración entonces la energía cinética es igual a 0.
- $\Delta Ep = 0$, el sistema no se cae o se eleva por lo que no hay energía potencial.
- $W = 0$, no hay corriente eléctricas ni radiaciones.

Entonces nos queda la ecuación del sistema como : $Q = \Delta U$ (El calor es igual al trabajo intercambiado por el sistema en su entorno).

Construcción de la PCB

Para un funcionamiento más profesional, se diseñó una PCB en el software EAGLE, de esta forma se pudo generar un diseño en el cual se conecta todo de una forma más eficiente

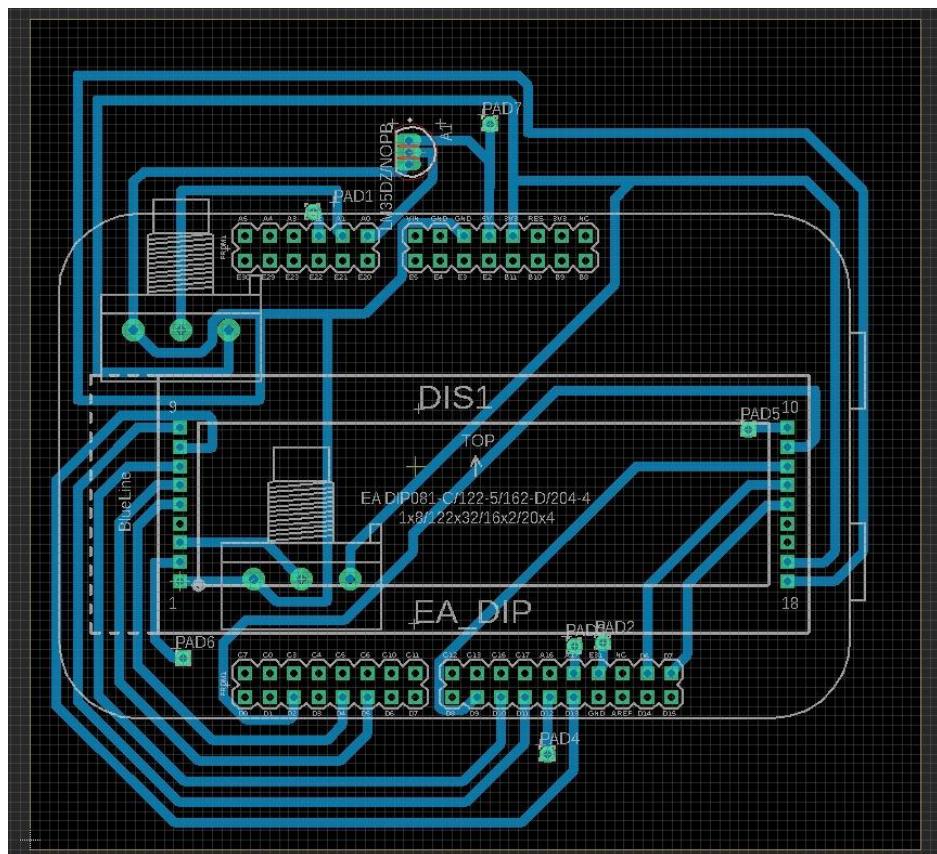


Figura 7. Diseño de la PCB en EAGLE

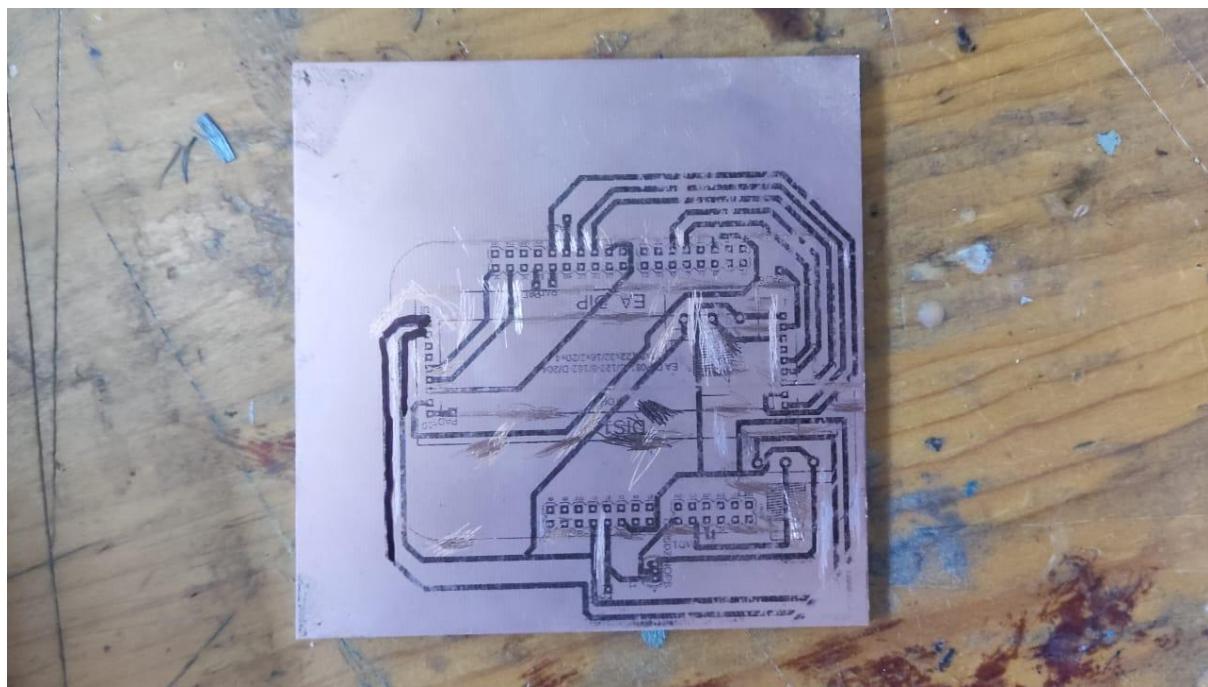


Figura 8. Intento fallido de PCB

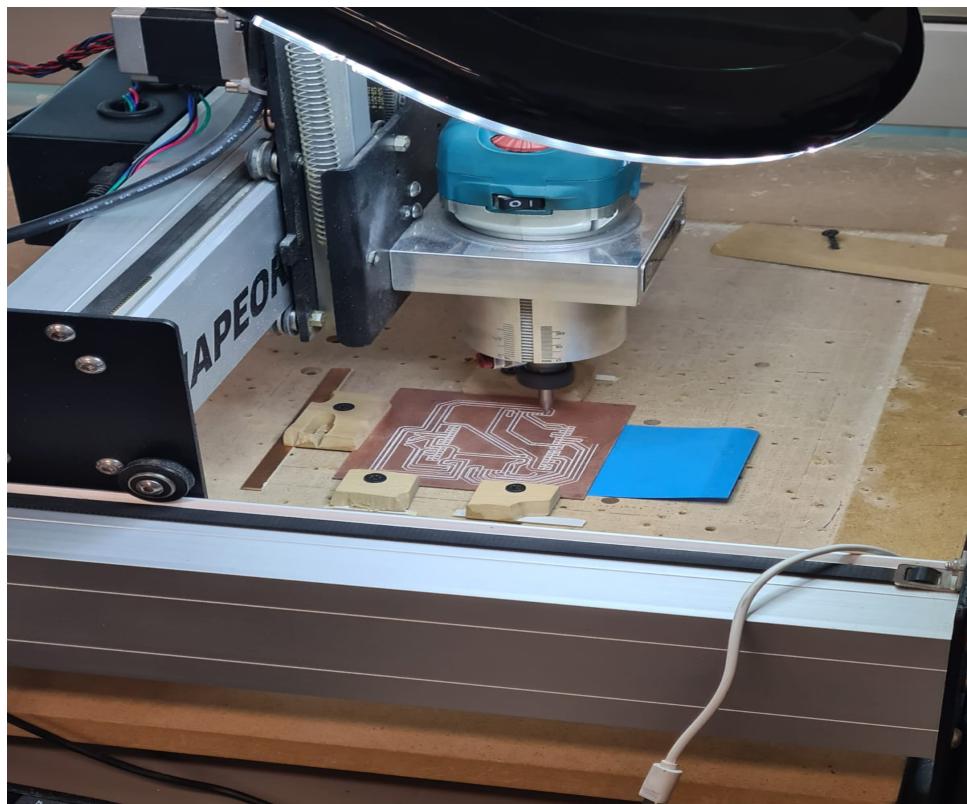


Figura 9. Impresión de la PCB

Conclusión

Durante todo el semestre estuvimos trabajando con diferentes temas para ayudarnos a realizar el proyecto mostrado durante todo el reporte, cada tema nos fue llevando a explorar más sobre los sistemas embebidos haciendo que de esta forma tuviéramos los recursos necesarios para completar este proyecto, estuvimos realizando diferentes actividades como lo fueron el procesamiento de señales, para de esta forma lograr reunir datos a partir de la planta generada, y aunque no se utiliza mucho en el reto la parte de programación en paralelo hizo que pudiéramos tener nuevas ideas para lograr resolver diferentes problemas que se fueron presentando conforme avanzábamos en la planta, el trabajar con sistemas embebidos requiere un gran esfuerzo puesto que el uso de kl25z como microprocesador tiene sus problemas y ventajas por lo que debíamos de estar revisando constantemente libros y diferentes guías para poder lograr conectarlo con los demás elementos de la planta.

Consideramos que logramos desarrollar todas las subcompetencias con las que cuenta este bloque, pero siendo unas más notorias que otras u otras que aun no nos sentimos que hayamos podido lograr al 100% estas subcompetencias, no obstante los conocimientos adquiridos durante este bloque fueron de gran ayuda, logrando darnos aprendizajes para continuar con los estudios en esta área de sistemas embebidos.

Apéndices

- **Apéndice 1 (Material y componentes)**

Para el desarrollo de nuestro proyecto utilizamos una fuente de poder del laboratorio del EIAD y cables para poder hacer las conexiones, de la parte de la planta usamos la placa kl25z, protoboard, jumpers macho-macho y macho-hembra, un puente h, un potenciómetro, sensor de temperatura, disipadores, para el calentamiento y enfriamiento de la planta usamos un peltier, LCD, switch y por último la caja.

- **Apéndice 2 (Código del sistema)**

<https://github.com/Shedew/Sistemas-embebidos/blob/main/main.c>

Referencias

1. Betancor C. Cerezo J. & Vega A. (2006, 19 junio). *DISEÑO DE UN SISTEMA DE CONTROL DE TEMPERATURA*. IUMA. Recuperado 17 de agosto de 2022, de <http://e-spacio.uned.es/fez/eserv/taee:congreso-2006-1116/S3F04.pdf>
2. Jarabo, F. & García F. (s.f). *Balances de Energía*. Campus Virtual. Recuperado el 09 de Septiembre de 2022, de https://campusvirtual.ull.es/ocw/pluginfile.php/1535/mod_folder/content/0/Esquema_T04.pdf?forcedownload=1