



Instituto Politécnico Nacional
Escuela Superior de Computo
"ESCOM"



Unidad de Aprendizaje:
Instrumentación y control

Grupo: 5CV2

PROYECTO
"MEDIDOR DE DISTANCIA"

Integrantes:
Ramírez Juárez Arturo Yamil
Suárez López Diego Hipólito
Zurita Cariño Emmanuel Einar

Maestro:
Cervantes De Anda Ismael

Fecha de entrega:
11/01/2024

Contenido

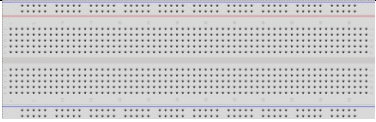






OBJETIVO	3
MATERIAL.....	3
EQUIPO	5
INTRODUCCIÓN TEÓRICA	4
DESARROLLO	7
Armado del CAS.....	7
Creación de la PCB	10
Visualización gráfica	13
CÁLCULOS	19
SIMULACIONES.....	21
CONCLUSIONES	25
BIBLIOGRAFÍA.....	26

OBJETIVO

El objetivo de este proyecto es poder conocer el funcionamiento de un sensor de distancia, empleando elementos básicos como lo es el potenciómetro, en este caso uno lineal.

- Como primera fase del proyecto se llevó a cabo el armado de un circuito de acondicionamiento de señal (CAS) empleando un puente de Wheatstone y el uso de amplificadores operacionales en sus diferentes configuraciones. Se realizaron todas las mediciones y cálculos necesarios para que su acondicionamiento sea correcto.
- Como segunda fase se procedió a conectar un ADC a nuestro circuito, con el fin de obtener de salida en formato de combinación binaria, esto representado en los leds que se conectan.
- Para la tercera fase se realizó el montaje de la placa PCB (Circuito impreso) del circuito, así como el ensamblado de este en la placa. Finalmente se realizaron las modificaciones y pasos pertinentes para que las medidas de distancia puedan ser visualizadas en un entorno gráfico.


MATERIAL








ProtoBoard	
Potenciómetro lineal	
LM324	
Resistencias 10K Ω a $\frac{1}{4}$ W	
ADC0804	
Resistencia de 390 Ω a $\frac{1}{4}$ W	
Led	

Capacitor de 150pF	
Capacitor de 10mF	
Capacitor de 0.1mF	
LM358	
Hojas de papel couche	
Placa de cobre 10x15	
Cutter	

Cloruro Férrico	
Estaño	

EQUIPO

Fuente de <u>alimentación</u> dual	
Multímetro digital	
Generador de funciones	
Osciloscopio de propósito general	

Cables coaxiales con terminal BNC-Caimán	
Cables caimán-caimán	
Cables banana-caimán	
Multímetro digital	
Impresora láser	
Plancha	
Arduino	

Cautín



Datasheet LM324

Low power quad op amps

LM124/224/324/324A/
SA534/LM2902

DESCRIPTION

The LM124/SA534/LM2902 series consists of four independent, high-gain, internally frequency-compensated operational amplifiers designed specifically to operate from a single power supply over a wide range of voltages.

UNIQUE FEATURES

In the linear mode, the input common-mode voltage range includes ground and the output voltage can also swing to ground, even though operated from only a single power supply voltage.

The unity gain crossover frequency and the input bias current are temperature-compensated.

FEATURES

- Internally frequency-compensated for unity gain
- Large DC voltage gain: 100dB
- Wide bandwidth (unity gain): 1MHz (temperature-compensated)
- Wide power supply range Single supply: $3V_{DC}$ to $30V_{DC}$ or dual supplies: $\pm 1.5V_{DC}$ to $\pm 15V_{DC}$
- Very low supply current drain: essentially independent of supply voltage (1mW/op amp at $+5V_{DC}$)
- Low input biasing current: $45nA_{DC}$ (temperature-compensated)
- Low input offset voltage: $2mV_{DC}$ and offset current: $5nA_{DC}$
- Differential input voltage range equal to the power supply voltage
- Large output voltage: $0V_{DC}$ to $V_{CC}-1.5V_{DC}$ swing

PIN CONFIGURATION

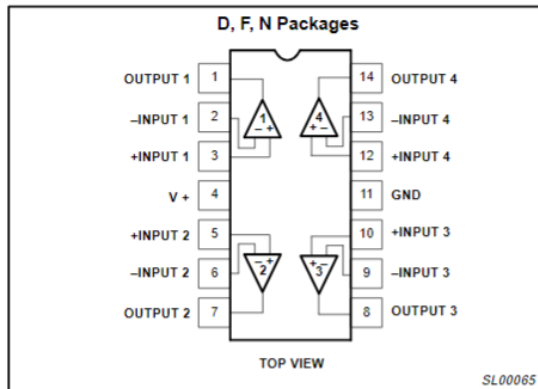


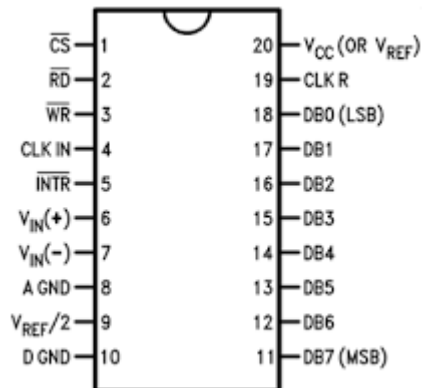
Figure 1. Pin Configuration

¿Cómo funciona el potenciómetro lineal?

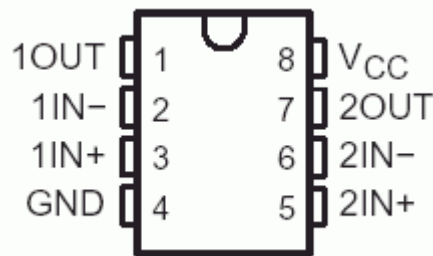
Los potenciómetros lineales son sensores de distancia resistivos que incorporan un cursor arrastrado por un vástago sobre una pista resistiva plástica, que provoca una variación de resistencia en los terminales de salida.



Datasheet ADC0804



Datasheet Im358

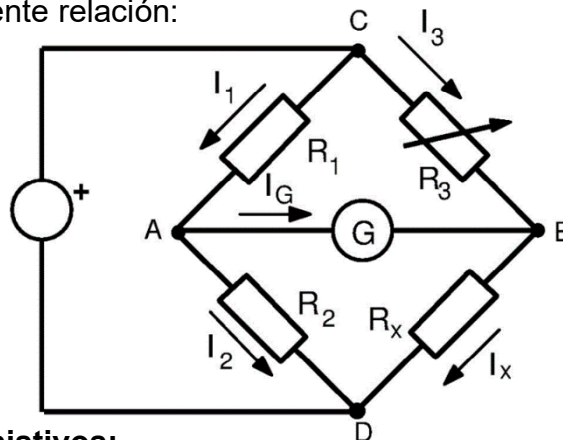


INTRODUCCIÓN TEÓRICA

Puente de Wheatstone

El puente de Wheatstone es un método para medir resistencias bastante exacto, ya que permite conocer el valor de una resistencia sabiendo el valor de las demás resistencias. Como se puede ver en la imagen, R_X es un valor de resistencia desconocido, mientras que R_1 , R_2 y R_3 son resistencias de valor conocido. El puente se alimenta con una fuente de tensión continua y se varía el valor de la resistencia R_3 mediante un mando hasta conseguir que el galvanómetro (que es un amperímetro muy sensible) indique que la corriente I_G tiene un valor nulo. En este caso se puede demostrar que se verifica la siguiente relación:

$$R_X = R_3 \frac{R_2}{R_1}$$



Tipos comunes de sensores resistivos:

Resistencias Dependientes de la Luz (LDR): Su resistencia varía con la cantidad de luz incidente. Son comúnmente utilizados en circuitos de detección de luz y oscurecimiento.

Termistores: Son resistencias que cambian su valor con la temperatura. Hay dos tipos principales:

NTC (Negative Temperature Coefficient): Su resistencia disminuye a medida que aumenta la temperatura.

PTC (Positive Temperature Coefficient): Su resistencia aumenta a medida que aumenta la temperatura.

Sensores de fuerza resistiva (FSR): Cambian su resistencia en función de la presión o fuerza aplicada sobre ellos.

Potenciómetros: Aunque son principalmente utilizados como divisores de voltaje o para controlar la corriente, la resistencia de un potenciómetro varía con la posición de su cursor, y esto puede ser utilizado en aplicaciones de detección de posición o rotación.

Circuito de acondicionamiento de señal

"Circuito de Acondicionamiento de Señal" (CAS) es crucial en sistemas de medición y control para adaptar y optimizar la señal procedente de un sensor antes de que sea digitalizada o procesada.

ADC0804

El ADC0804 es un convertidor analógico a digital de 8 bits. Este dispositivo tiene una gran variedad de aplicaciones, ya que convierte las señales de forma analógica a digital. Estas señales digitalizadas se utilizan para el procesamiento de los procesadores digitales. Por ejemplo, encontramos una gran diversidad de sensores que convierten las características físicas del medio en señales analógicas.

El ADC0804 trabaja con un voltaje de entrada analógica de 0V a 5V. Trae una entrada analógica y las convierte a 8 salidas digitales. Además, este convertidor está diseñado para permitir el funcionamiento con el bus de control derivativo NSC800 e INS8080A con latches de salida Tri-State que dirigen directamente el bus de datos.

LM385

El LM35 es un sensor de temperatura. Sin embargo, no es resistivo como los termistores; en cambio, es un sensor de temperatura lineal con una salida de voltaje proporcional a la temperatura Celsius (Centígrada). Tiene varias ventajas sobre los sensores resistivos, como los termistores:

- Salida lineal: El LM35 produce una salida de voltaje que varía linealmente con la temperatura. Típicamente, tiene una sensibilidad de 10 mV/°C, lo que significa que por cada grado Celsius de aumento en la temperatura, la salida aumenta en 10 mV.
- No requiere calibración externa: Dado que su salida es lineal y calibrada en grados Celsius, el LM35 no necesita ningún tipo de calibración externa.

- Rango de operación: Por lo general, el LM35 puede medir temperaturas desde -55°C hasta 150°C , aunque existen versiones que pueden medir en otros rangos.
- Precisión: Dependiendo del modelo específico, el LM35 puede tener una precisión de $\pm 0.5^{\circ}\text{C}$ a temperatura ambiente.
- Bajo consumo de energía: A diferencia de algunos sensores resistivos que requieren corriente constante para operar, el LM35 tiene un consumo de corriente muy bajo, lo que lo hace adecuado para aplicaciones con baterías.

PCB

Sus siglas en inglés corresponden a placa de circuito impreso ("printed circuit board" en inglés), y se define como un circuito cuyos componentes y conductores están contenidos dentro de una estructura mecánica. Las funciones conductoras de la PCB incluyen trazas de cobre, terminales, disipadores de calor o conductores planos. La estructura mecánica se hace con material laminado aislante entre capas de material conductor. A su vez, la estructura general de la placa es chapada y cubierta con una máscara de soldadura no conductora y una pantalla de impresión para la ubicación de leyenda de componentes electrónicos. La placa de circuito impreso está construida por capas que se alternan de cobre conductor con capas de material aislante no conductor. Durante la fabricación de la PCB, se graban las capas de cobre internas dejando trazas de cobre intencionadas para conectar los componentes de circuito. Una vez laminado el material de aislamiento este es grabado a las capas de cobre y así sucesivamente, hasta que la placa de circuito impreso esté completa.

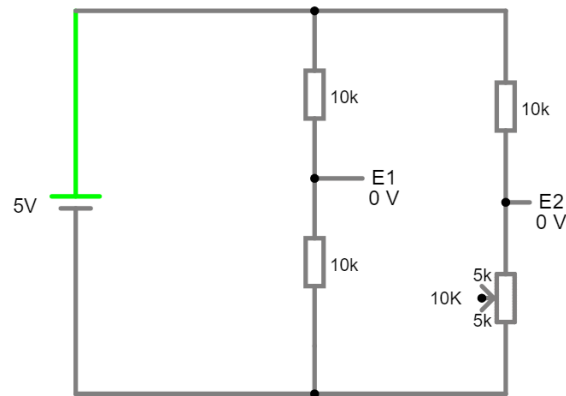
Los componentes se agregan a las capas externas de la placa de circuito impreso o PCB cuando todas las capas se han grabado y laminado juntas. Las partes de montaje superficial se aplican automáticamente con robots y las partes con orificio pasante se colocan manualmente. Una vez hecho esto, todas las partes se sueldan en la placa utilizando técnicas tales como reflujo o soldadura por ola. El montaje final se chapa después de aplicar la máscara de soldadura y la pantalla de impresión de la leyenda.

DESARROLLO

Armado del CAS

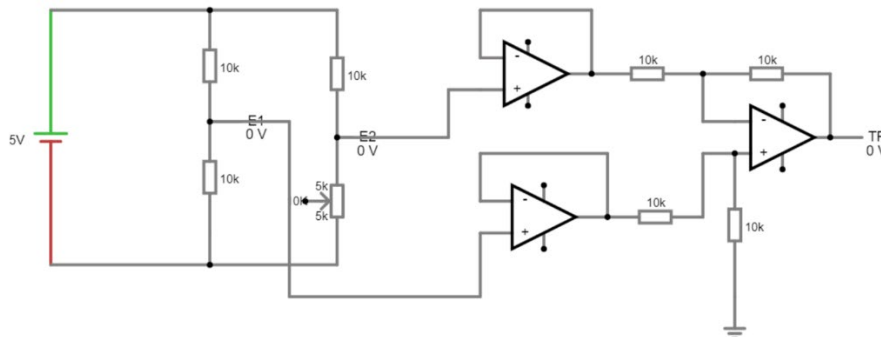
La elaboración del Circuito de Acondicionamiento de Señal (CAS) implicó una serie de pasos a los cuales se le realizaron cálculos previos, ejecutados para establecer correctamente la relación entre la distancia y el voltaje de salida.

Primero, se diseñó un puente de Wheatstone utilizando un potenciómetro lineal y resistencias de 10k ohms como elementos resistivos. Este potenciómetro lineal se seleccionó debido a su capacidad para cambiar su resistencia en función de la posición de su cursor, lo que lo hace ideal para medir la distancia.

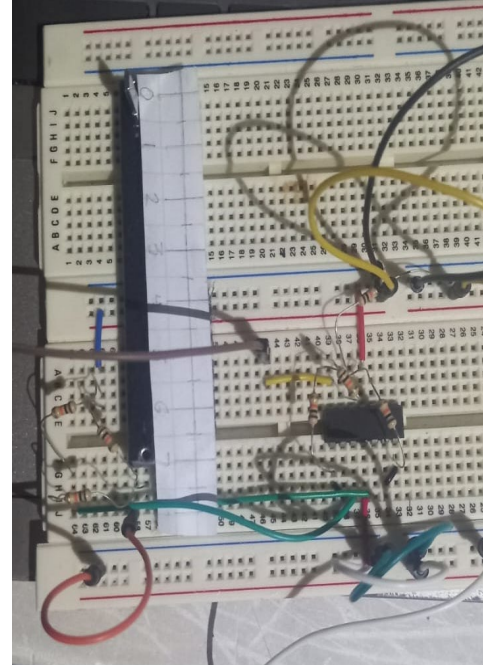
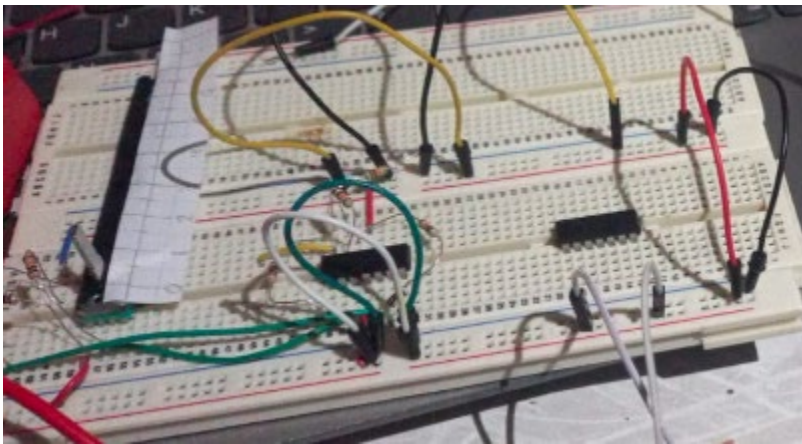


Posteriormente, se realizaron cálculos para determinar la relación entre la distancia que recorre el cursor del potenciómetro y el cambio resultante en el voltaje de salida del puente.

Estos cálculos se basaron en las propiedades eléctricas del potenciómetro y del puente, así como en las características físicas del sistema que se está midiendo. El voltaje de entrada para el puente fue de 5V.



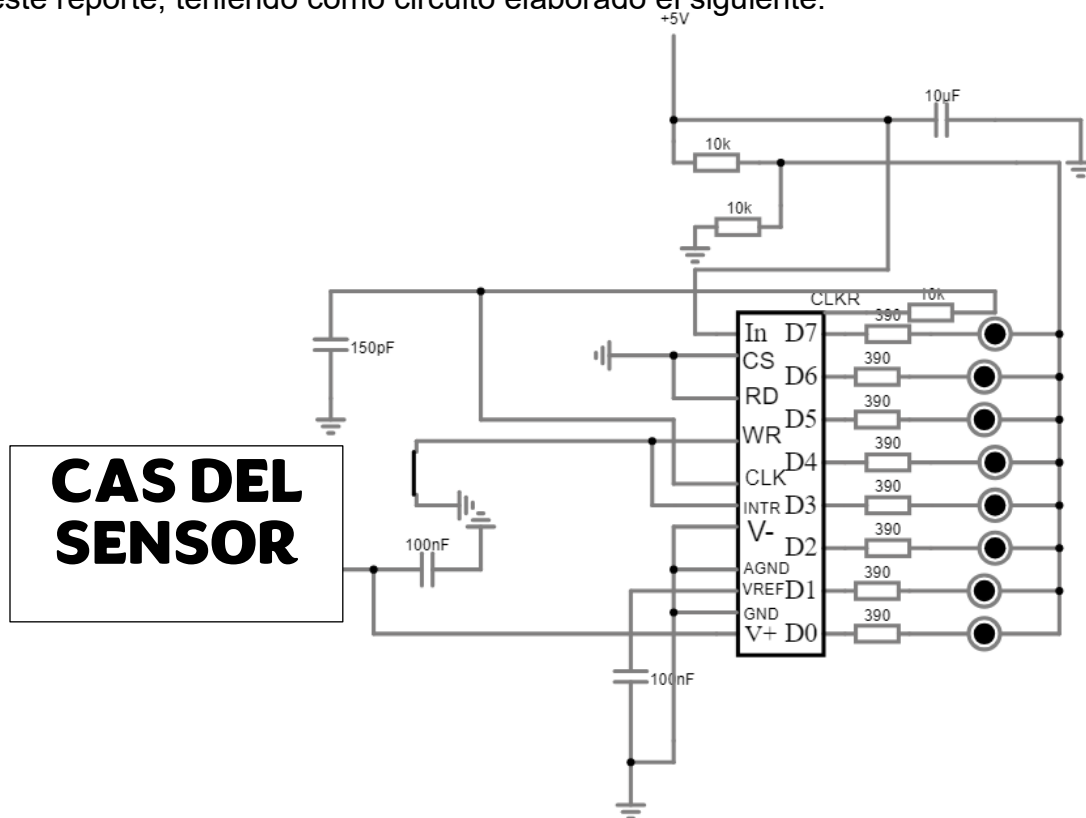
El circuito comienza con el puente de Wheatstone en el que se mide E1 y E2, como se comprobará en los cálculos, al hacer las mediciones E1 mide 2.5V mientras que E2 va de 0V a 2.5V, posteriormente cada voltaje es enviado a un amplificador operacional en configuración de seguidor de voltaje, con el objetivo de que no exista una variación de corriente, ya que necesitamos utilizar los mismos valores de voltaje. Ambas salidas del seguidor de voltaje son enviadas a otro amplificador operacional en configuración de restador, lo que nos permitirá hacer la resta de voltajes. Y finalmente medimos el voltaje de salida.



Para poder saber cuánto avanza la palanca del potenciómetro, se graduó la barra en centímetros.

Una vez que se estableció esta relación, se pudo calibrar el sistema para producir un voltaje de salida que sea proporcional a la distancia medida. Esto permite que el sistema mida con precisión las distancias basándose en los cambios de voltaje.

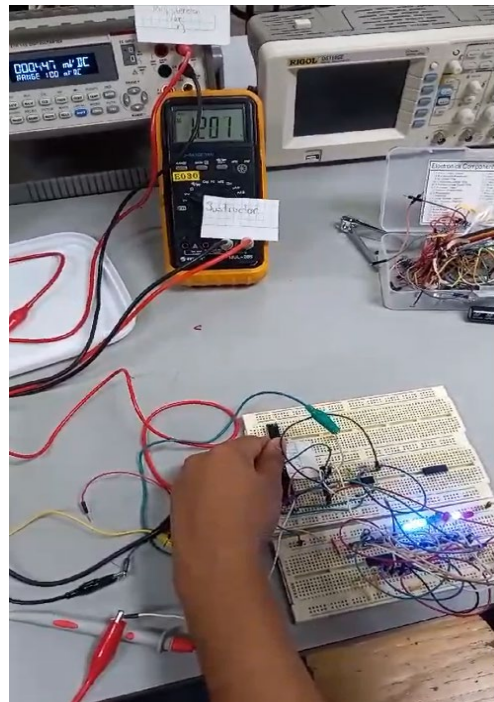
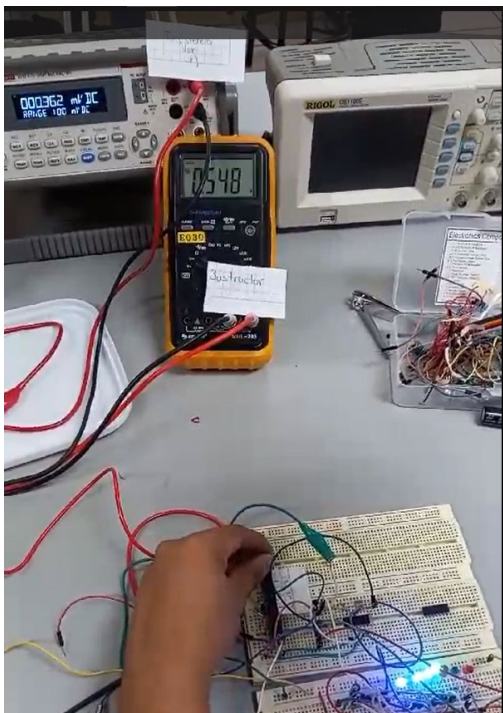
Procedemos ahora a crear un circuito básico con un ADC, con el fin de convertir la señal analógica en una digital, este fue realizado en la práctica anterior a la entrega de este reporte, teniendo como circuito elaborado el siguiente:



Desconectamos el potenciómetro que se encuentra en el diagrama. A continuación, se conectó el circuito de nuestro proyecto a este:

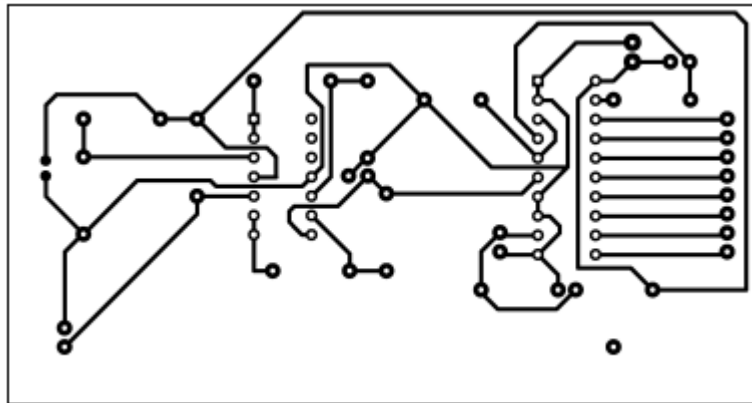
Como se mostrará en la siguiente tabla, variamos el valor del potenciómetro, con el objetivo de tener las medidas correspondientes a cada distancia.
En este caso, se tomaron en cuenta menos medidas debido a un problema con el potenciómetro pasado los 4 cm.

No. Medición	Distancia (cm)	Voltaje de Salida Medido (V)	Combinación Binaria								Comb. Decimal
			B7	B6	B5	B4	B3	B2	B1	B0	
1	0	0.548	0	0	0	1	1	1	0	1	29
2	0.5	0.556	0	0	0	1	1	1	0	0	28
3	1.0	0.549	0	0	0	1	1	1	0	0	28
4	1.5	0.568	0	0	0	1	1	1	1	0	30
5	2.0	0.879	0	0	0	1	1	1	1	1	31
6	2.3	0.973	0	0	1	1	0	1	0	1	53
7	2.5	1.163	0	0	1	1	1	1	1	1	63
8	2.7	1.201	0	1	0	0	0	1	1	1	71
9	2.9	1.580	0	1	0	1	0	0	0	1	81
10	3.00	1.638	0	1	0	1	0	1	0	1	85
11	3.2	1.642	0	1	0	1	0	1	0	1	85
12	3.4	1.705	0	1	0	1	1	0	1	1	91
13	3.5	1.911	0	1	1	0	0	0	1	1	99



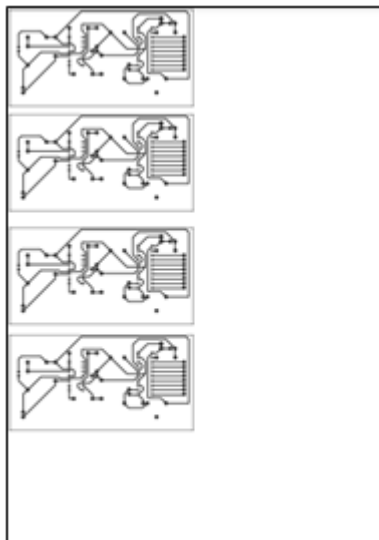
Creación de la PCB

Para el armado del proyecto es necesario imprimir el circuito en una placa de cobre, con el fin de armar la PCB. El circuito diseñado es el siguiente:



Es importante que el circuito sea impreso en la hoja usando una impresora láser, debido a que esta usa tóner y queremos que este sea transferido exactamente como está a la placa de cobre.

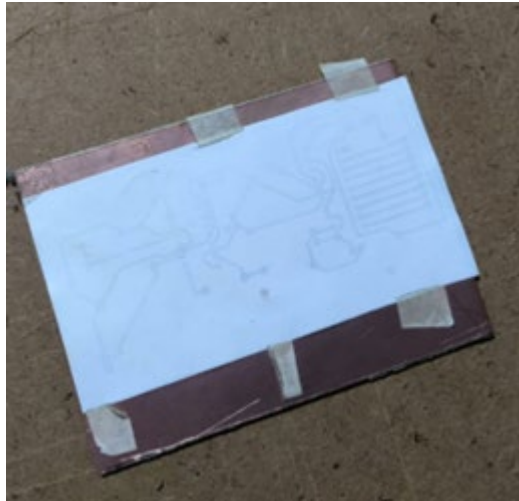
Se imprimieron varias copias del circuito en la hoja con el fin de tener respaldo en caso de cometer algún error.



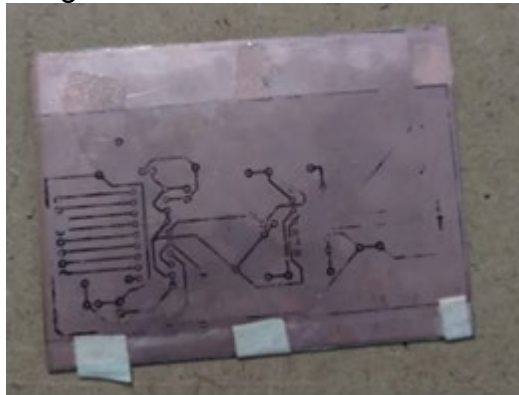
Procedemos a recortar nuestro circuito de la hoja, lo fijamos usando masking y usando un plumón marcamos las dimensiones de este en la placa de cobre con el fin de cortarla.

Se le da un poco de margen para que la placa y el circuito no queden justos de tamaño en caso de cometer algún error.

Ahora procedemos a usar el cutter para cortar la placa con mucho cuidado y pasando mas de una vez en los cortes para facilitar la separación.



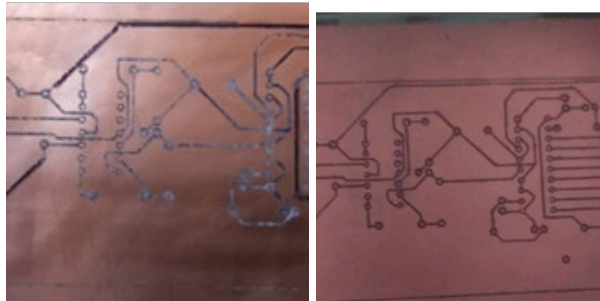
Procedemos a usar una plancha, previamente calentada a su temperatura máxima, para adherir el tóner de la hoja a la placa durante 20 a 25 segundos, se tuvo que intentar más de una vez debido a que el tóner parecía no adherirse a la placa correctamente a pesar de seguir las indicaciones de no mover la placa o la plancha.



Es importante remarcar que se usó una plancha lisa y no a base de vapor debido al poco espacio liso que esta puede proporcionar. Continuamos sujetando la placa de cobre usando un trapo y procedimos a frotar dicho trapo con la placa y el circuito, con el fin de ayudar a que el tóner se termine de transferir a la placa. Ahora para remover el papel couche del circuito de la placa se llenó un recipiente, lo suficientemente grande como para que esta quepa, con agua y frotamos con nuestro dedo la hoja hacia las esquinas, si llegáramos a desprender el papel sin cuidado, entonces la hoja se llevaría todo el tóner.



Una vez terminado este proceso, se revisa la placa y si esta tiene marcada correctamente las líneas y puntos del circuito, si no es así procedemos a usar un marcador para remarcar dichas líneas.

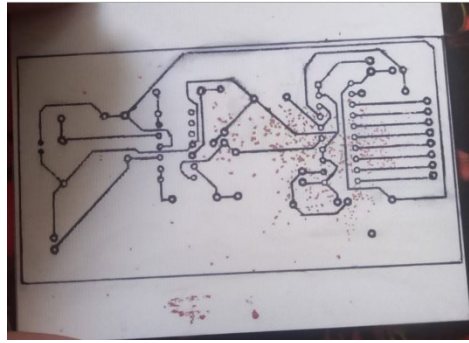


En caso de cometer errores, o que la placa no tenga al menos el 80% del circuito transferido, es necesario borrar la placa e intentarlo de nuevo mediante el uso de un flux para limpiar o un solvente para tinta, esto con el fin de facilitar el trabajo de remarcar y que el circuito quede lo más exacto posible.

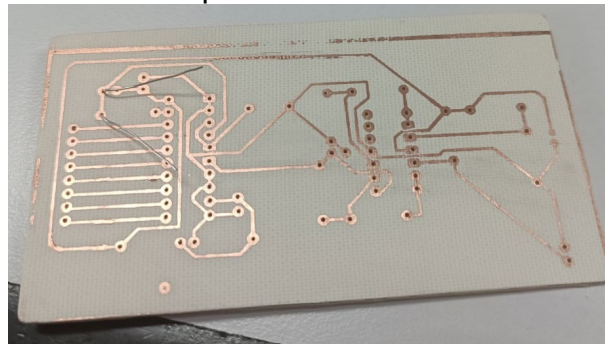
Procedemos ahora a eliminar la parte de cobre que no nos interesa, dejando solamente las líneas de tóner con cobre, esto mediante el uso del mismo recipiente y el uso de Cloruro Férrico, volviendo a meter la placa en el recipiente y el cloruro y agitándolos para ayudar a que el proceso sea más rápido.



Una vez terminado el proceso retiramos nuestro circuito impreso y lo lavamos con agua, luego regresamos el cloruro férrico al recipiente para volver a ser usado en el futuro.



Continuamos limpiando el t  n de las l  neas del circuito usando el flux o solvente. De esta manera nuestro circuito impreso quedar  a finalizado y lo   nico que faltar  a ser  a perforar la placa donde corresponde usando brocas de 1/16 y 1/32 de pulgada.



El resultado final es la placa ya con los agujeros para comenzar el ensamblado de los componentes y posteriormente el soldado de estas.

Visualizaci  n gr  fica

Como parte fundamental y final para este proyecto, fue necesario desarrollar un c  digo que, junto a una conexi  n a Arduino, fuera capaz de recibir y manejar los datos proporcionados por el circuito e interpretarlos para hacer una representaci  n gr  fica en la computadora.

Este proyecto consiste principalmente de dos c  digos desarrollados en Python que realizan dicho trabajo, tenemos primero el c  digo "proyecto2p.py":

```

1  import pygame
2  import player
3  import serial
4  import threading
5
6  pygame.init()
7  pygame.font.init()
8
9  COLOR_TEXTO = (28, 40, 51)
10 ANCHO_VENTANA = 640
11 ALTO_VENTANA = 480
12 VRESOL = 0.0196078
13 R = 9570
14 fuente = pygame.font.SysFont(None, 36)
15
16 screen = pygame.display.set_mode((ANCHO_VENTANA, ALTO_VENTANA))
17 pygame.display.set_caption("Proyecto Instru")
18 clock = pygame.time.Clock()
19
20 PUERTO_ARDUINO = 'COM5'
21 BAUD_RATE = 9600
22 arduino = serial.Serial(PUERTO_ARDUINO, BAUD_RATE)
23 player = player.Kate((ANCHO_VENTANA/2, ALTO_VENTANA - 140)) # Ajusta según la clase 'Kate'
24 datos = 0.0
25 is_run = False # Corregido a 'is_run'
26 game_over = False
27
28 def leer_datos_arduino():
29     global datos, is_run
30
31     while is_run:
32         try:
33             if arduino.in_waiting > 0:
34                 linea = arduino.readline().decode('utf-8').strip()
35                 try:
36                     datoss = float(linea)
37                     # Cálculo con los datos recibidos
38                     ele2 = VRESOL * datoss
39                     e2 = 2.5 - ele2
40                     dr = ((e2 * (2 * R)) - (5 * R)) / (e2 - 5)
41                     rsen = (e2 * ((2 * R) - dr)) / 5
42                     datos = (rsen - 1.0001) / 1460.9160
43                 except ValueError:
44                     print("Error al convertir datos:", linea)
45             except serial.SerialException:
46                 print("Error en la comunicación serial")
47                 break

```

```

49 def iniciar():
50     global is_run
51     is_run = True
52     threading.Thread(target=leer_datos_arduino).start()
53     animation()
54 def detener():
55     global isrun, arduino
56     isrun = False
57     arduino.close()
58
59 def draw_full_gradient(surface, color1, color2, rect):
60
61     height = rect.height
62     top_color = pygame.Color(*color1)
63     bottom_color = pygame.Color(*color2)
64     step = 1.0 / height
65
66     for y in range(height):
67         color = top_color.lerp(bottom_color, step * y)
68         pygame.draw.line(surface, color, (rect.left, rect.top + y), (rect.right, rect.top + y))
69
70 def dibujar_calle(surface, rect):
71
72     ancho_calle = rect.width
73     alto_calle = rect.height // 4
74     y_calle = rect.height - alto_calle
75
76     # carretera
77     pygame.draw.rect(surface, (50, 50, 50), (rect.left, y_calle, ancho_calle, alto_calle))
78
79     # líneas de la carretera
80     ancho_linea = 5
81     espacio_linea = 20
82     for x in range(0, ancho_calle, ancho_linea * 2 + espacio_linea):
83         pygame.draw.rect(surface, (255, 255, 255), (x, y_calle + alto_calle // 2 - ancho_linea // 2, ancho_linea, ancho_linea))
84
85 def animation():
86     global game_over, player, datos, ANCHO_VENTANA, ALTO_VENTANA
87     while not game_over:
88         for event in pygame.event.get():
89             if event.type == pygame.QUIT:
90                 game_over = True
91
92         #fondo
93         draw_full_gradient(screen, (255, 204, 51), (255, 102, 102), screen.get_rect())
94         dibujar_calle(screen, screen.get_rect())
95         # movimiento personaje
96         player.mover_segun_voltaje(datos, ANCHO_VENTANA)
97
98         # Dibujar el texto
99         texto = fuente.render(f'Distancia recorrida: {datos:.2f}', True, COLOR_TEXTO)
100        screen.blit(texto, (10, 10))
101
102
103        screen.blit(player.image, player.rect)
104        pygame.display.flip()
105        clock.tick(20)
106
107    detener()
108    pygame.quit()
109
110    iniciar()
111

```

Este código consiste en lo siguiente:

1. Configuración inicial:

- Se importan las bibliotecas necesarias como Pygame, threading, serial y el módulo **player** que parece ser un archivo Python externo (llamado **player.py**).
- Se inicializa Pygame y se configuran algunas constantes y parámetros de la ventana, como el tamaño y los colores.

2. Conexión con Arduino:

- Se establece la conexión con un dispositivo Arduino a través de la comunicación serial. El puerto y la velocidad de transmisión se especifican en las variables **PUERTO_ARDUINO** y **BAUD_RATE**, respectivamente.
3. **Configuración del jugador:**
 - Se crea una instancia de la clase **player.Kate** (definida en el archivo **player.py**) que representa al personaje principal del juego. La posición inicial se establece en la parte inferior central de la ventana.
 4. **Hilo para leer datos del Arduino:**
 - Se inicia un hilo (**leer_datos_arduino**) que se encarga de leer datos del Arduino en segundo plano. Los datos se utilizan para calcular variables relacionadas con el juego.
 5. **Funciones para iniciar y detener el juego:**
 - Se definen las funciones **iniciar()** y **detener()** para comenzar y detener el hilo que lee los datos del Arduino, respectivamente.
 6. **Dibujo del entorno del juego:**
 - Se definen funciones para dibujar el fondo del juego y la "calle" donde se moverá el personaje.
 7. **Bucle principal del juego:**
 - Se inicia un bucle principal (**animation()**) que se ejecuta mientras la variable **game_over** sea falsa.
 - En cada iteración del bucle, se actualiza la pantalla con el fondo, la calle, el movimiento del jugador según los datos del Arduino y la visualización de la distancia recorrida.
 - El bucle maneja eventos de Pygame, como el cierre de la ventana.
 8. **Cierre del juego:**
 - Cuando el bucle principal termina (por ejemplo, debido al cierre de la ventana), se llama a la función **detener()** para cerrar adecuadamente la comunicación serial y terminar el hilo que lee los datos del Arduino.

El código crea un juego simple con un personaje controlado por datos provenientes de un Arduino. La distancia recorrida por el personaje se muestra en la ventana del juego. El código utiliza hilos para permitir la lectura continua de datos desde el Arduino mientras el juego se ejecuta.

Cómo código complementario "player.py" tenemos:

```

1  import pygame
2
3  class Kate(pygame.sprite.Sprite):
4      def __init__(self, position):
5          self.sheet = pygame.image.load('kate.png')
6          self.sheet.set_clip(pygame.Rect(0, 0, 52, 76))
7          self.image = self.sheet.subsurface(self.sheet.get_clip())
8          self.rect = self.image.get_rect()
9          self.rect.topleft = position
10         self.frame = 0
11         self.left_states = { 0: (0, 76, 52, 76), 1: (52, 76, 52, 76), 2: (156, 76, 52, 76) }
12         self.right_states = { 0: (0, 152, 52, 76), 1: (52, 152, 52, 76), 2: (156, 152, 52, 76) }
13         self.up_states = { 0: (0, 228, 52, 76), 1: (52, 228, 52, 76), 2: (156, 228, 52, 76) }
14         self.down_states = { 0: (0, 0, 52, 76), 1: (52, 0, 52, 76), 2: (156, 0, 52, 76) }
15
16     def get_frame(self, frame_set):
17         self.frame += 1
18         if self.frame > (len(frame_set) - 1):
19             self.frame = 0
20         return frame_set[self.frame]
21
22     def clip(self, clipped_rect):
23         if type(clipped_rect) is dict:
24             self.sheet.set_clip(pygame.Rect(self.get_frame(clipped_rect)))
25         else:
26             self.sheet.set_clip(pygame.Rect(clipped_rect))
27         return clipped_rect
28
29     def update(self, direction):
30         if direction == 'left':
31             self.clip(self.left_states)
32             self.rect.x -= 5
33         if direction == 'right':
34             self.clip(self.right_states)
35             self.rect.x += 5
36         if direction == 'up':
37             self.clip(self.up_states)
38             self.rect.y -= 5
39         if direction == 'down':
40             self.clip(self.down_states)
41             self.rect.y += 5
42
43         if direction == 'stand_left':
44             self.clip(self.left_states[0])
45         if direction == 'stand_right':
46             self.clip(self.right_states[0])
47         if direction == 'stand_up':
48             self.clip(self.up_states[0])
49         if direction == 'stand_down':
50             self.clip(self.down_states[0])
51
52         self.image = self.sheet.subsurface(self.sheet.get_clip())
53
54     def mover_segun_voltaje(self, voltaje, ancho_ventana):
55         velocidad = 5
56         if voltaje < 2.5: # Mover a la izquierda
57             if self.rect.left > 0: # Verificar límite izquierdo
58                 self.clip(self.left_states)
59                 self.rect.x -= velocidad
60             else: # Mover a la derecha
61                 if self.rect.right < ancho_ventana: # Verificar límite derecho
62                     self.clip(self.right_states)
63                     self.rect.x += velocidad
64
65         self.image = self.sheet.subsurface(self.sheet.get_clip())

```

Este código define una clase llamada **Kate** que representa a un personaje en un juego utilizando la biblioteca Pygame. A continuación, se proporciona una descripción de las principales funcionalidades de la clase:

1. Inicialización:

- La clase tiene un método **__init__** que se llama al crear una instancia de la clase. En este método, se carga una imagen de un archivo llamado 'kate.png' que contiene sprites del personaje. Se configuran diferentes estados del personaje para la animación, como moverse hacia la izquierda, derecha, arriba y abajo.

2. Métodos de animación:

- La clase tiene métodos para gestionar la animación del personaje (**get_frame**, **clip**, **update**). Estos métodos manipulan el sprite del personaje y permiten cambiar la apariencia del personaje en función de la dirección del movimiento.

3. Método update:

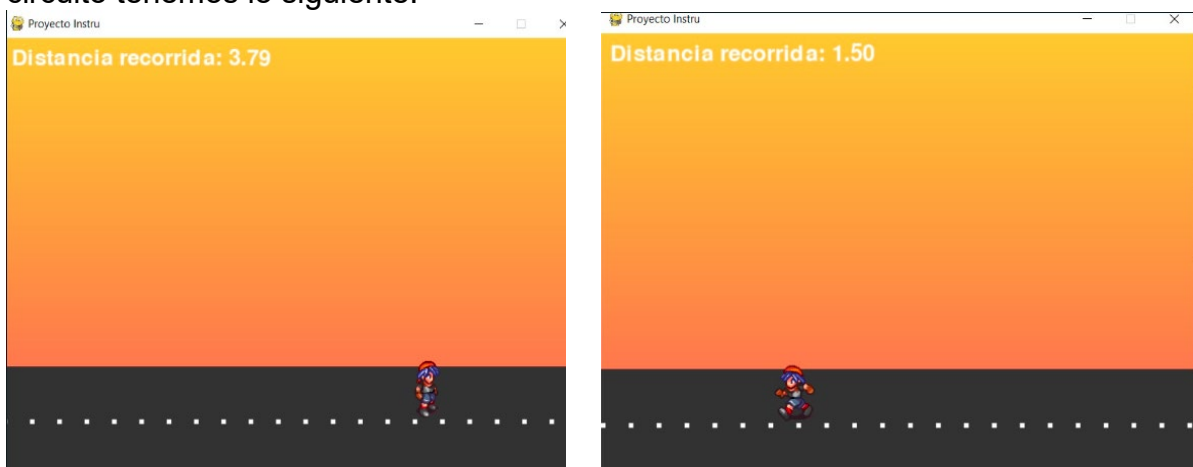
- El método **update** toma una dirección como argumento y ajusta la apariencia del personaje en consecuencia. Además, actualiza la posición del personaje en función de la dirección proporcionada.

4. Método mover_segun_voltaje:

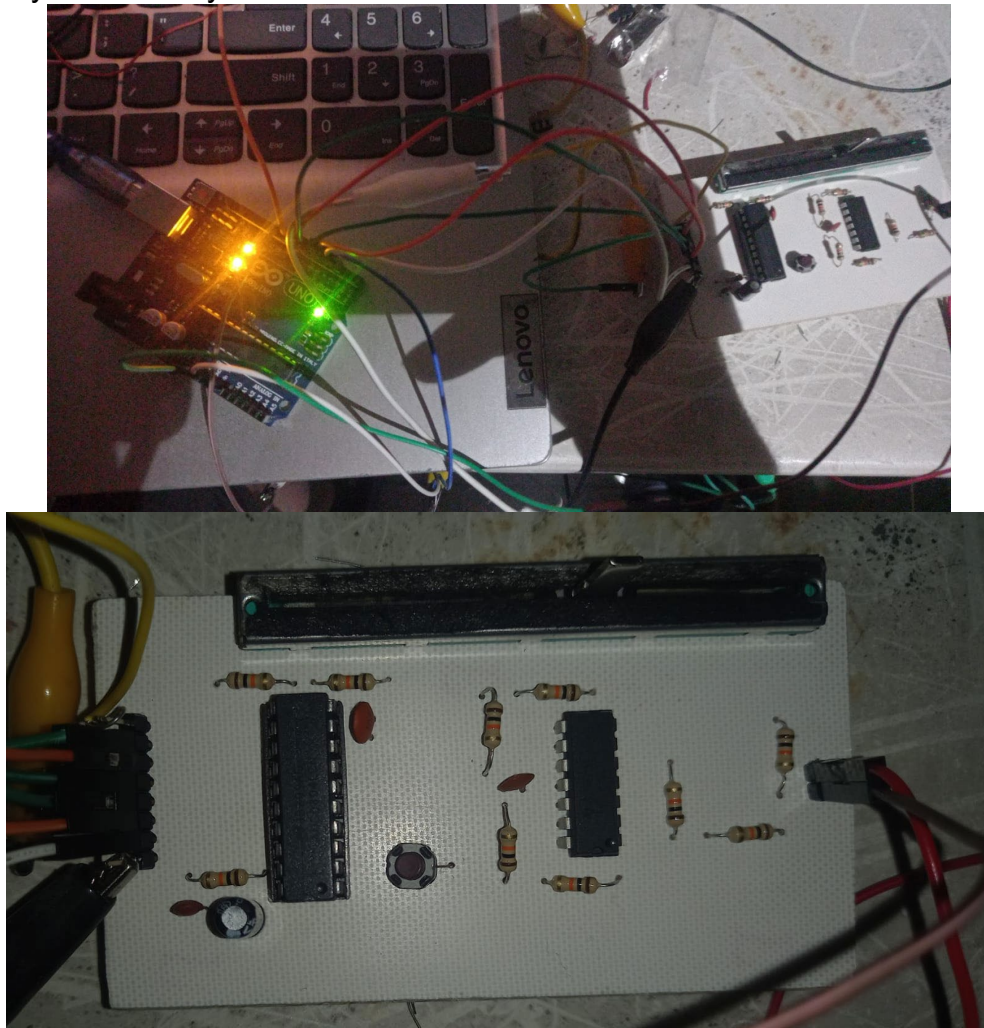
- Este método toma un valor de voltaje y el ancho de la ventana como argumentos. Dependiendo del valor del voltaje, el método decide si mover el personaje hacia la izquierda o hacia la derecha. La velocidad de movimiento se establece en 5, y se verifica que el personaje no se salga de los límites de la ventana.

La clase encapsula la lógica para gestionar la apariencia y el movimiento de un personaje en un juego 2D. Se utiliza principalmente para animar y desplazar el personaje en respuesta a eventos específicos, como el voltaje proveniente de un dispositivo externo, según se indica en el método **mover_segun_voltaje**. La clase es parte integral de un juego más grande que utiliza la biblioteca Pygame para la creación de interfaces gráficas y la animación.

Finalmente como representación visual para observar las mediciones realizadas por el circuito tenemos lo siguiente:



El circuito ya montado y funcional:



CÁLCULOS

Diseñando el CAS tenemos:

$$\begin{aligned}
 &0\text{cm a } 6.55\text{ cm} \\
 &1\ \Omega \text{ a } 9.57\text{k}\Omega \\
 m &= (y_2 - y_1)/(x_2 - x_1) = (9.57\text{k}\Omega - 1\Omega)/(6.55\text{cm} - 0\text{cm}) = 1460.9160 \\
 y &= mD + b \rightarrow R_{sen} = mD + b \\
 E1 &= \frac{E}{2} = \frac{5v}{2} = 2.5v \\
 R_{sen} &= R \pm \Delta R
 \end{aligned}$$

Para p1:

$$R_{sen} = (1460.9160)(0\text{cm}) + b$$

$$\begin{aligned}
b &= (1\Omega) - 0 = 1 \\
I &= \frac{E}{R_{sen}} = \frac{5V}{1\Omega} = 5A \\
E2 &= \frac{E(R - \Delta R)}{2R - \Delta R} = \frac{5V(9.57k\Omega - (9.57k\Omega - 1\Omega))}{2(9.57k\Omega) - (9.57k\Omega - 1\Omega)} = 522.411mV \\
E1 - E2 &= 2.5V - 522.411mV = 2.4994V
\end{aligned}$$

Para p2:

$$\begin{aligned}
R_{sen} &= (1460.9160)(6.55cm) + b \\
b &= (9.57k\Omega) - (1460.9160 * 6.55) = 1.0002 \rightarrow 1 \\
I &= \frac{E}{R_{sen}} = \frac{5V}{9.57k\Omega} = 5.22466mA \\
E2 &= \frac{E(R - \Delta R)}{2R - \Delta R} = \frac{5V(9.57k\Omega - (9.57k\Omega - 9.57k\Omega))}{2(9.57k\Omega) - (9.57k\Omega - 9.57k\Omega)} = 2.5V \\
E1 - E2 &= 2.5V - 2.5V = 0V \\
R_{sen} &= (1460.9160)(D) + 1.0001
\end{aligned}$$

Entonces tenemos:

$$\therefore 2.4994V \text{ a } 0V$$

Si

$$0V \rightarrow 2.4994V$$

$$A_v = \frac{V_{sal}}{E1 - E2} = \frac{2.4994V + 0V}{2.4994V - 0V} = 1$$

$$Si R_f = R_c \text{ y } R_a = R_b \therefore R_f = 10k\Omega = R_c$$

$$A_v = \frac{10k\Omega}{R_a} \rightarrow \frac{10k\Omega}{R_a} = 1 \therefore R_a = R_b = 10k\Omega$$

$$V_{cas} = A_v * (E1 - E2)$$

Dado que, al momento de hacer las mediciones con el ADC conectado, los valores sólo cambian en un rango de 0cm-3.5cm, las mediciones se hicieron bajo este rango de salida

$$V_{resolucion} = \frac{V_{rango}}{2^N - 1}$$

$$V_{resolucion} = \frac{5V - 0V}{2^8 - 1} = 19.6078mV$$

$$Comb = \frac{V_{cas} - V_{ref}(-)}{V_{resolución}} = 27$$

$$Comb_{0cm} = \frac{0.548V - 0V}{19.6078mV} = 27$$

$$Comb_{0.5cm} = \frac{0.556V - 0V}{19.6078mV} = 28$$

$$Comb_{1cm} = \frac{0.549V - 0V}{19.6078mV} = 27$$

$$Comb_{1.5cm} = \frac{0.568V - 0V}{19.6078mV} = 28$$

$$Comb_{2cm} = \frac{0.879V - 0V}{19.6078mV} = 44$$

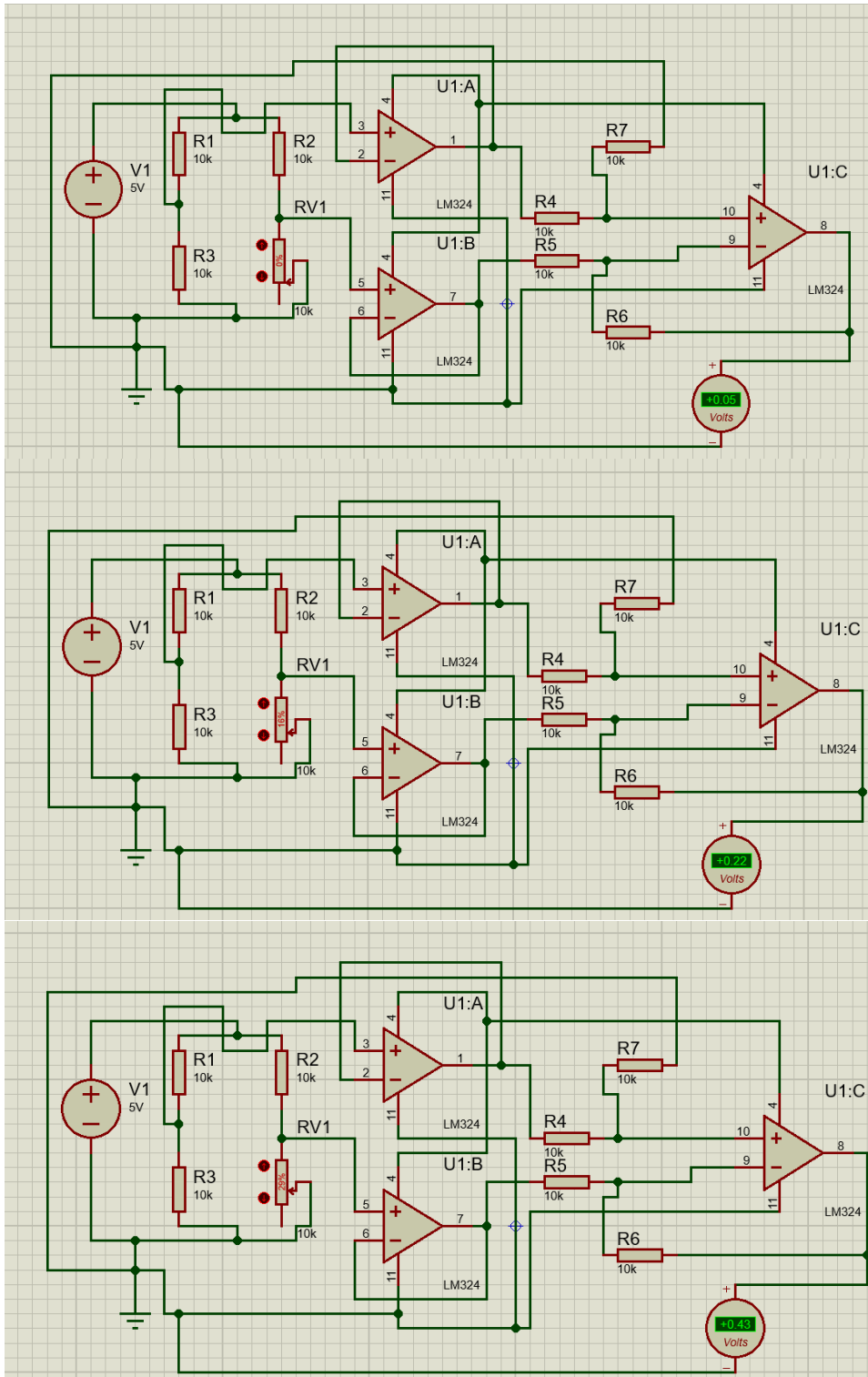
$$Comb_{2.5cm} = \frac{1.163V - 0V}{19.6078mV} = 59$$

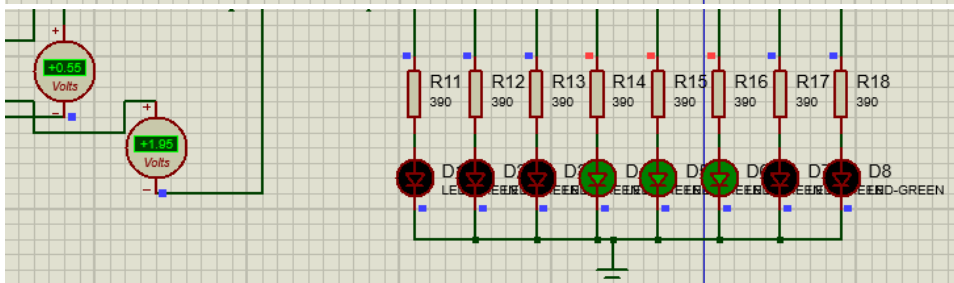
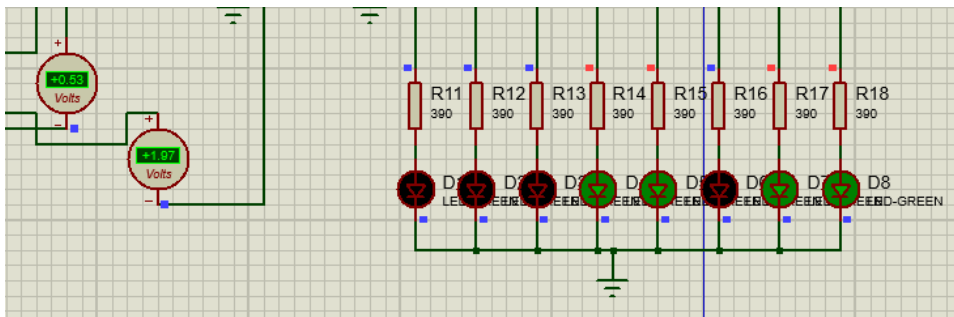
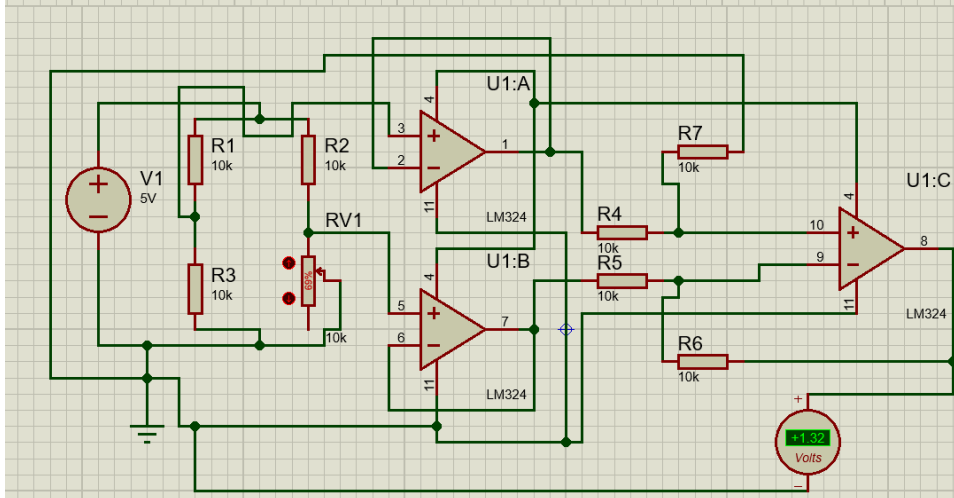
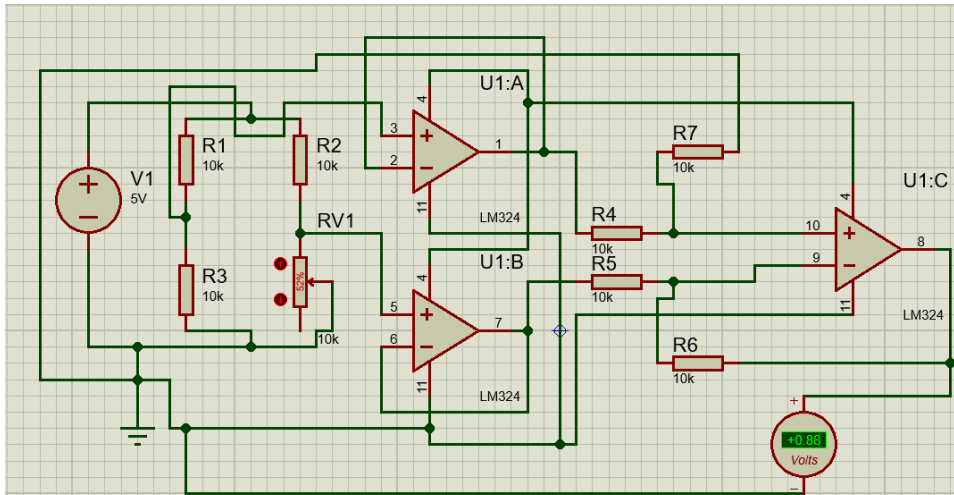
$$Comb_{3cm} = \frac{1.638V - 0V}{19.6078mV} = 83$$

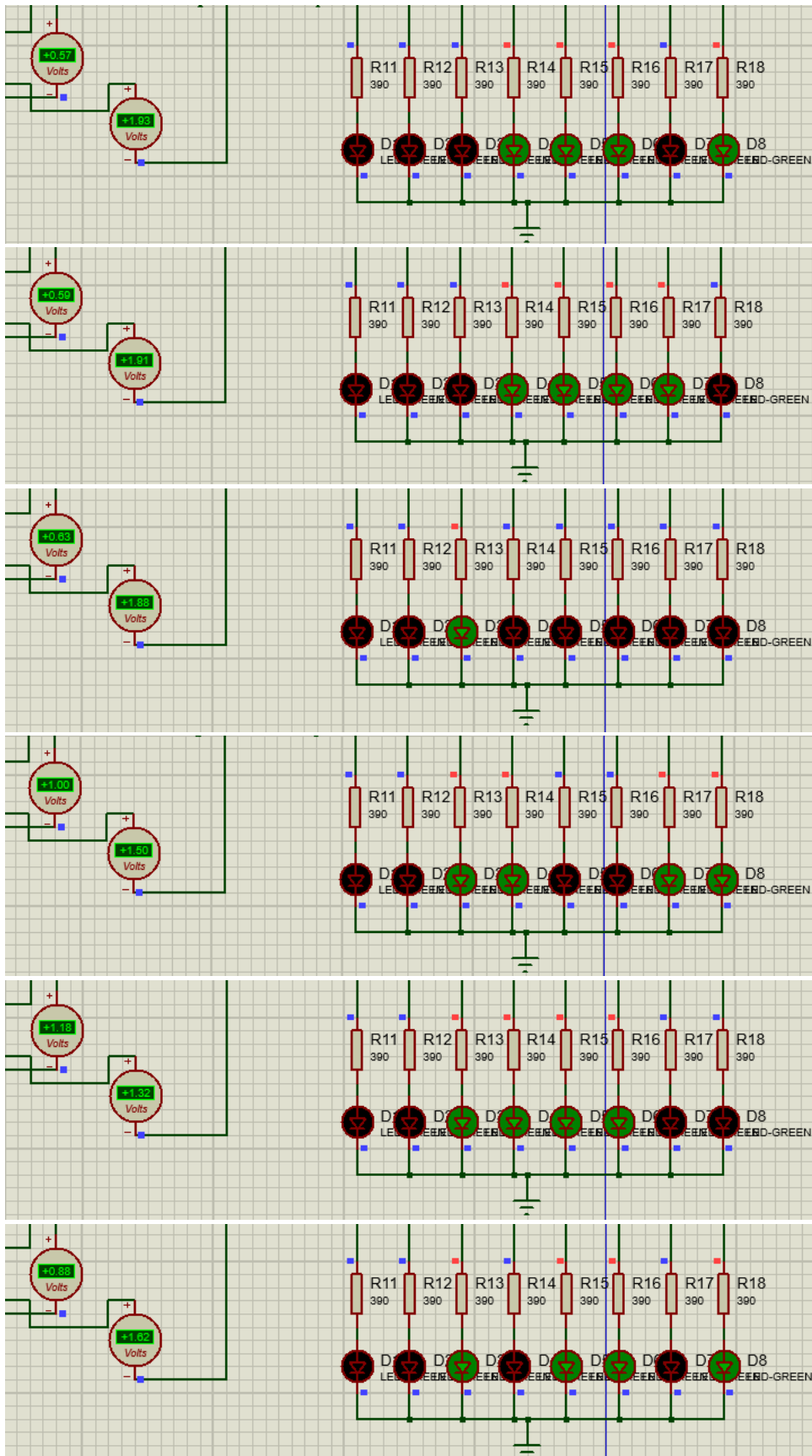
$$Comb_{3cm} = \frac{1.911V - 0V}{19.6078mV} = 97$$

SIMULACIONES

Se muestra a continuación las simulaciones realizadas para comprobar los resultados medidos y calculados de estas dos fases del proyecto.







CONCLUSIONES

La primera fase de este proyecto, que implicó la construcción de un Circuito de Acondicionamiento de Señal (CAS) utilizando un puente de Wheatstone y un potenciómetro lineal, se ha logrado consumir satisfactoriamente. Hemos logrado establecer una relación entre la distancia medida y el voltaje de salida, lo cual es fundamental para el funcionamiento del medidor de distancia.

El uso del puente de Wheatstone y el potenciómetro lineal ha demostrado ser una elección acertada. Estos componentes nos han permitido convertir cambios físicos (distancia) en cambios eléctricos (resistencia), y luego medir estos cambios de manera precisa.

Los cálculos realizados para establecer la relación entre la distancia y el voltaje de salida fueron cruciales para el éxito de esta fase. Estos cálculos nos permitieron ajustar con precisión nuestro CAS para que proporcionara una lectura precisa del voltaje de salida para cada distancia medida.

A pesar de los desafíos que presentó esta fase, hemos aprendido mucho sobre la instrumentación y el control. Hemos adquirido una comprensión más profunda de cómo funcionan los puentes de Wheatstone y los potenciómetros lineales, y cómo se pueden utilizar para medir con precisión las distancias.

Mirando hacia adelante, estamos emocionados por las próximas fases de este proyecto. Si la primera fase es una indicación, estamos seguros de que podremos superar cualquier desafío que se nos presente y llevar a cabo un proyecto exitoso.

La segunda fase de este proyecto, que consistió en agregar un Convertidor Analógico a Digital (ADC) al circuito de la primera fase, también se ha completado con éxito.

El ADC ha permitido una mayor precisión y eficiencia en la conversión de las señales analógicas provenientes del Circuito de Acondicionamiento de Señal (CAS) a señales digitales que pueden ser procesadas y analizadas con mayor facilidad.

Este avance ha mejorado la funcionalidad de nuestro medidor de distancia, permitiéndonos obtener lecturas más precisas y confiables.

Los desafíos encontrados en esta fase nos han proporcionado una valiosa oportunidad para profundizar nuestro entendimiento de los ADCs y cómo se integran en los sistemas de medición.

Como tercera fase tuvimos que ver la creación de la PCB para el ensamblado y soldadura de los componentes del circuito, esto con el fin de posteriormente hacer que la visualización de los datos obtenidos por el circuito pueda observarse en un entorno gráfico. Esta parte fue fundamental para la finalización de nuestro proyecto, por lo que obligatoriamente tenía que hacerse y esto a su vez nos llevó a cometer muchos errores en la creación de la PCB, sin embargo se lograron solventar teniendo como resultado un proyecto funcional y con los requisitos establecidos al principio del semestre cubiertos.

BIBLIOGRAFÍA

- Doebelin, E. O., & Manik, D. N. (2007). Measurement Systems: Application and Design. McGraw Hill.
- Fraden, J. (2010). Handbook of Modern Sensors: Physics, Designs, and Applications. Springer.
- Tocci, R. J., & Widmer, N. S. (2001). Digital Systems: Principles and Applications. Prentice Hall.
- Hoja de datos de ADC0804-N, información de producto y soporte (n.d.). Recuperado de <https://www.ti.com/product/es-mx/ADC0804-N>
- ADC0801/ADC0802/ADC0803/ADC0804/ADC0805 8-Bit P Compatible A/D Converters (n.d.). Recuperado de <http://web.mit.edu/6.115/www/document/adc0801.pdf>
- ADC0804 Conversor Analógico a Digital (n.d.). Recuperado de <https://tdelectronica.com/producto/circuitos-integrados/compuertas-logicas/adc0804-conversor-analogico-a-digital/>
- Electronics Hub. (2021). Applications of PCBs. Electronics Hub. Disponible en: <https://www.electronicshub.org/applications-of-pcbs/>
- Mouser Electronics. (2021). What is a PCB? Mouser Electronics. Disponible en: <https://www.mouser.com/applications/pcb-what-is-a-printed-circuit-board/>