

# TRABAJO FIN DE GRADO

UNIVERSIDAD DE ALMERIA

ESCUELA SUPERIOR DE INGENIERÍA

Control de un horno de bajo coste para  
soldadura de PCB por reflujo



**Curso 2023/2024**

**Modalidad TFG:** Trabajo Técnico

**Alumno/a:**

Jordano Patricio Almeida Gavilanes

**Director/es:**

José Luis Blanco Claraco  
José Luis Guzmán Sánchez

UNIVERSIDAD DE ALMERÍA  
ESCUELA SUPERIOR DE INGENIERÍA

TRABAJO FIN DE GRADO  
INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA



**Control de un horno de bajo coste para soldadura  
de PCB por reflujo**

Curso 2023/2024

Autor:

Jordano Patricio Almeida Gavilanes

Directores:

José Luis Blanco Claraco

José Luis Guzmán Sánchez



## ÍNDICE GENERAL

Agradecimientos .....	VII
Acrónimos.....	IX
Índice de figuras.....	XI
Índice de tablas .....	XV
Resumen .....	XVII
Abstract .....	XIX
<b>1    Introducción .....</b>	<b>1</b>
1.1 <i>Motivación del proyecto</i> .....	2
1.2 <i>Objetivos</i> .....	2
1.3 <i>Resumen de resultados</i> .....	3
1.4 <i>Uso de las competencias adquiridas</i> .....	5
1.5 <i>Estructura de la memoria</i> .....	5
1.6 <i>Fases de realización y cronograma</i> .....	6
<b>2    Materiales y métodos .....</b>	<b>9</b>
2.1 <i>Tecnología SMT</i> .....	9
2.1.1    Proceso de impresión de soldadura en pasta .....	9
2.1.2    Proceso de <i>pick and place</i> .....	10
2.1.3    Proceso de reflujo de soldadura .....	11
2.2 <i>Dispositivos electrónicos discretos</i> .....	13
2.2.1    MOSFET.....	13
2.2.2    Resistencias.....	14
2.2.3    Condensadores .....	15
2.2.4    Diodos .....	15
2.2.5    Referencia de tensión .....	16
2.2.6    Amplificador de instrumentación .....	18
2.2.7    Optoacoplador.....	18
2.3 <i>Fuentes de alimentación resistivas</i> .....	20
2.4 <i>Placa de desarrollo</i> .....	21
2.5 <i>Métodos y estrategias de control</i> .....	22
2.5.1    Control PID.....	22
2.5.2    Ponderación del punto de consigna .....	24
2.5.3    Esquema <i>antiwindup</i> .....	25
2.5.4    Modo Manual y Automático .....	27
2.5.5    Método analítico cancelación polo cero.....	28

2.5.6	Método analítico $\lambda$ .....	28
2.6	<i>Herramientas informáticas</i> .....	29
2.6.1	Matlab.....	29
2.6.2	Simulink .....	30
2.6.3	Guide.....	30
2.6.4	<i>System Identification Toolbox</i> .....	30
2.6.5	Arduino IDE .....	30
<b>3</b>	<b>Desarrollo del sistema .....</b>	<b>33</b>
3.1	<i>Modificación del horno</i> .....	33
3.2	<i>Diseño de la electrónica</i> .....	37
3.2.1	Especificaciones de las placas .....	37
3.2.2	Cálculos teóricos del acondicionador de señal .....	37
3.2.3	Cálculos teóricos de la placa de potencia .....	47
3.2.4	Simulaciones de las placas .....	58
3.2.5	Diseño de los PCB.....	61
3.3	<i>Montaje y conexiones de los subsistemas</i> .....	70
3.4	<i>Programación de la interfaz gráfica de Matlab</i> .....	74
3.5	<i>Programación del Arduino</i> .....	81
3.6	<i>Modelado del horno</i> .....	89
3.6.1	Obtención de los modelos .....	89
3.6.2	Verificación de los modelos .....	95
3.7	<i>Diseño de la estrategia de control</i> .....	99
3.7.1	Diseño teórico .....	99
3.7.2	Simulación de la estrategia de control.....	101
<b>4</b>	<b>Resultados y discusión .....</b>	<b>105</b>
4.1	<i>Sistema del horno</i> .....	105
4.1.1	Prueba de funcionamiento de las placas diseñadas .....	105
4.1.2	Prueba de funcionamiento de los periféricos .....	106
4.2	<i>Interfaz gráfica</i> .....	107
4.2.1	Visualización y recogida de datos .....	107
4.3	<i>Sistema de control</i> .....	109
4.3.1	Prueba de control manual a automático .....	109
4.3.2	Prueba de seguimiento de referencias .....	110
<b>5</b>	<b>Conclusiones y trabajos futuros .....</b>	<b>111</b>
5.1	<i>Conclusiones</i> .....	111
5.2	<i>Trabajos futuros</i> .....	112
<b>6</b>	<b>Bibliografía .....</b>	<b>115</b>
<b>7</b>	<b>Anexos.....</b>	<b>119</b>
7.1	<i>Código Arduino del proyecto</i> .....	119
7.2	<i>Código Matlab de la interfaz gráfica</i> .....	128

7.3	<i>Esquemas electrónicos.....</i>	133
-----	-----------------------------------	-----



## Agradecimientos

A mi querida familia, gracias por ser mi roca, mi refugio y mi mayor fuente de inspiración. Vuestra constante dedicación y sacrificio han sido los cimientos sobre los cuales he construido mis sueños y ambiciones. Cada sonrisa, cada abrazo y cada palabra de aliento han iluminado mi camino y me han dado la fuerza para superar cualquier obstáculo.

A mis amigos, ustedes son mi segunda familia, siempre presentes en cada aventura, en cada risa y en cada desafío. Vuestra amistad ha enriquecido mi vida de innumerables maneras, recordándome constantemente el valor de la camaradería y el compañerismo. Gracias por compartir conmigo los buenos momentos y por estar a mi lado en los momentos difíciles.

A mis estimados profesores, les debo un profundo agradecimiento por su dedicación y compromiso con mi educación. Vuestra pasión por enseñar ha encendido en mí una llama de curiosidad y conocimiento que nunca se extinguirá. Cada lección aprendida no solo ha enriquecido mi intelecto, sino que también ha moldeado mi carácter y mi perspectiva del mundo.

Y finalmente, pero no menos importante, a mi querido gato Tommy, mi compañero peludo y confidente. Tu amor incondicional y tu presencia reconfortante han sido un faro de luz en los días oscuros y una fuente inagotable de alegría en los días soleados. Gracias por tus ronroneos reconfortantes, tus travesuras juguetonas y por simplemente ser tú mismo.



## Acrónimos

<b>PCB</b>	Printed Circuit Board
<b>THT</b>	Through Hole Technology
<b>SMD</b>	Surface Mount Device
<b>SMT</b>	Surface Mount Technology
<b>RSS</b>	Ramp Soak Spike
<b>RTS</b>	Ramp To Spike
<b>JIT</b>	Just In Time
<b>TAL</b>	Time Above Liquids
<b>MOSFET</b>	Metal-Oxide-Semiconductor Field-Effect-Transistor
<b>RMS</b>	Root Mean Square
<b>CMRR</b>	Common Mode Rejection Ratio
<b>DDC</b>	Control Digital Directo
<b>OLED</b>	Organic Light Emitting Diode
<b>USB</b>	Universal Serie Bus
<b>PWM</b>	Pulse Width Modulation
<b>CI</b>	Circuito Integrado
<b>PID</b>	Proporcional Integral Derivativo



## Índice de figuras

<i>Figura 1.1. Esquema general del sistema.</i> .....	3
<i>Figura 1.2. Resultado de validación del modelo nominal utilizado en el diseño del controlador.</i> .....	4
<i>Figura 1.3. Resultado de la transferencia sin saltos.</i> .....	4
<i>Figura 1.4. Resultados obtenidos del horno al seguir referencias de temperatura.</i> .....	5
<i>Figura 2.1. Proceso de impresión de pasta de soldadura (Fuente:[8]).</i> .....	10
<i>Figura 2.2. Máquina pick and place modelo NeoDen10 de NeoDen® (Fuente:[10]).</i> .....	11
<i>Figura 2.3. Perfil de temperatura RSS (Fuente:[12]).</i> .....	12
<i>Figura 2.4. Perfil de temperatura RTS (Fuente:[12]).</i> .....	13
<i>Figura 2.5. Características de salida de un MOSFET tipo enriquecimiento (Fuente:[13]).</i> .....	14
<i>Figura 2.6. Resistencias de diversos valores (Fuente:[14]).</i> .....	15
<i>Figura 2.7. Clasificación general de los condensadores (Fuente:[16]).</i> .....	15
<i>Figura 2.8. Diferentes tipos de diodos (Fuente:[18]).</i> .....	16
<i>Figura 2.9. Referencia de tensión TL1431 (Fuente:[20]).</i> .....	17
<i>Figura 2.10. Amplificador operacional LT1490 (Fuente:[21]).</i> .....	18
<i>Figura 2.11. Esquema del amplificador de instrumentación INA128 (Fuente:[23]).</i> .....	18
<i>Figura 2.12. Características del dispositivo CNY65 (Fuente:[25]).</i> .....	19
<i>Figura 2.13. Ratio de transferencia de corriente (Fuente:[25]).</i> .....	19
<i>Figura 2.14. Optoacoplador en colector común y en emisor común (Fuente:[26]).</i> .....	20
<i>Figura 2.15. Fuente de alimentación resistiva (Fuente:[27]).</i> .....	20
<i>Figura 2.16. Placa de desarrollo modelo mega 2560 de Arduino (Fuente:[28]).</i> .....	22
<i>Figura 2.17. Respuestas a cambios de consigna en el punto de referencia (Fuente:[29]).</i> .....	25
<i>Figura 2.18. Ilustración de la saturación del integrador (Fuente:[29]).</i> .....	25
<i>Figura 2.19. Controlador PID con mecanismo de protección antiwindup (Fuente:[29]).</i> .....	27
<i>Figura 2.20. PID con implementación paralela que conmuta entre el control manual y automático (Fuente:[29]).</i> .....	27
<i>Figura 2.21. IDE de Arduino (Fuente:[34]).</i> .....	31
<i>Figura 3.1. Esquema visual del desarrollo del sistema (Repetida de la figura 1.1 por conveniencia).</i> ...	33
<i>Figura 3.2. Horno eléctrico modelo HO980 de la marca Orbegozo® (Fuente:[35]).</i> .....	33
<i>Figura 3.3. Horno sin modificaciones.</i> .....	34
<i>Figura 3.4. Cámara termográfica modelo HTi80P de Huepar®.</i> .....	34
<i>Figura 3.5. Vista termográfica lateral del horno.</i> .....	35
<i>Figura 3.6. Vista termográfica del interior del horno.</i> .....	35
<i>Figura 3.7. Vista lateral del Sensor PT100 colocado.</i> .....	35
<i>Figura 3.8. Vista interior del Sensor PT100 colocado.</i> .....	36
<i>Figura 3.9. Conexiones eléctricas del horno.</i> .....	36
<i>Figura 3.10. Esquema del circuito para el acondicionador</i> .....	38
<i>Figura 3.11. Ecuación de la PT100 linealizada y los errores cometidos.</i> .....	39
<i>Figura 3.12. Circuito básico del TL1431 (Fuente:[20]).</i> .....	40
<i>Figura 3.13. Circuito utilizado para proporcionar la alimentación a la PT100.</i> .....	41
<i>Figura 3.14. Circuito equivalente para los cálculos de <math>R_{Lim}</math>.</i> .....	41
<i>Figura 3.15. Etapa inversora para obtener 45.9 mV en el puente.</i> .....	42
<i>Figura 3.16. Circuito final para alimentar la PT100.</i> .....	43
<i>Figura 3.17. Referencia de tensión que va al INA128.</i> .....	45

<i>Figura 3.18. Circuito final de la referencia de tensión que va al INA128.</i> .....	45
<i>Figura 3.19. Circuito del filtro antialiasing.</i> .....	47
<i>Figura 3.20. Circuito de potencia utilizado.</i> .....	47
<i>Figura 3.21. Recorrido de la corriente en el semiciclo positivo.</i> .....	48
<i>Figura 3.22. Recorrido de la corriente en el semiciclo negativo.</i> .....	48
<i>Figura 3.23. Características principales del dispositivo PB3006 (Fuente:[41]).</i> .....	48
<i>Figura 3.24. Características principales del dispositivo STF13N60M2 (Fuente:[42]).</i> .....	49
<i>Figura 3.25. Características de encendido y apagado (Fuente:[42]).</i> .....	49
<i>Figura 3.26. Fuente de alimentación resistiva empleada.</i> .....	50
<i>Figura 3.27. Caída de tensión de la fuente de alimentación resistiva para una carga de <math>1k\Omega</math>.</i> .....	51
<i>Figura 3.28. Factor de rizado para distintos <math>\tau</math> (Fuente:[43]).</i> .....	51
<i>Figura 3.29. Rise time off y factor de rizado con 0 voltios en el PWM.</i> .....	52
<i>Figura 3.30. Rise time on y factor de rizado con 5 voltios en el PWM.</i> .....	53
<i>Figura 3.31. Características eléctricas del CNY65 (Fuente:[25]).</i> .....	54
<i>Figura 3.32. Curva <math>I_F</math> vs <math>V_F</math> y curva <math>I_C</math> vs <math>V_{CE}</math> del CNY65 (Fuente:[25]).</i> .....	55
<i>Figura 3.33. Temperatura máxima en la unión del STF13N60M2 (Fuente:[42]).</i> .....	55
<i>Figura 3.34. Circuito equivalente para calcular la <math>I_D</math> que atraviesa el MOSFET.</i> .....	56
<i>Figura 3.35. Curva <math>I_D</math> vs <math>V_{DS}</math> del STF13N60M2 (Fuente:[42]).</i> .....	56
<i>Figura 3.36. Resistencia térmica unión-ambiente del STF13N60M2 (Fuente:[42]).</i> .....	57
<i>Figura 3.37. Equivalente térmico unión-ambiente del STF13N60M2.</i> .....	57
<i>Figura 3.38. Circuito térmico al incluir un dissipador y una lámina de mica (Fuente:[44]).</i> .....	58
<i>Figura 3.39. Dissipador de referencia utilizado (Fuente:[45]).</i> .....	58
<i>Figura 3.40. Salida del puente para una variación de <math>-5^\circ\text{C}</math>.</i> .....	59
<i>Figura 3.41. Salida del puente para una variación de <math>277^\circ\text{C}</math>.</i> .....	59
<i>Figura 3.42. Simulación del circuito de potencia con 0V en PWM.</i> .....	60
<i>Figura 3.43. Simulación del circuito de potencia con 5V en PWM.</i> .....	60
<i>Figura 3.44. Esquema electrónico del acondicionador de señal.</i> .....	61
<i>Figura 3.45. Footprint creado de forma manual para el condensador del filtro antialiasing.</i> .....	62
<i>Figura 3.46. Ejemplo de mala conexión de componentes en Multisim (Fuente:[46]).</i> .....	62
<i>Figura 3.47. Componente creado de forma manual para el INA128.</i> .....	62
<i>Figura 3.48. Recomendación del fabricante Texas Instruments® para las pistas en el INA128 (Fuente:[47]).</i> .....	63
<i>Figura 3.49. Disposición de los componentes en la PCB del acondicionador de señal.</i> .....	63
<i>Figura 3.50. Capa top del acondicionador de señal.</i> .....	64
<i>Figura 3.51. Capa bottom del acondicionador de señal.</i> .....	64
<i>Figura 3.52. Capa de serigrafía del acondicionador de señal.</i> .....	65
<i>Figura 3.53. Capa de visualización 3D del acondicionador de señal.</i> .....	65
<i>Figura 3.54. Footprint creado de forma manual para el optoacoplador.</i> .....	66
<i>Figura 3.55. Esquema electrónico de la placa de potencia.</i> .....	66
<i>Figura 3.56. Disposición de los componentes en la PCB de la placa de potencia.</i> .....	67
<i>Figura 3.57. Capa top del acondicionador de la placa de potencia.</i> .....	68
<i>Figura 3.58. Capa bottom de la placa de potencia.</i> .....	68
<i>Figura 3.59. Capa de serigrafía de la placa de potencia.</i> .....	69
<i>Figura 3.60. Capa de visualización 3D de la placa de potencia.</i> .....	69

<i>Figura 3.61. Placa del acondicionador de señal.</i> .....	70
<i>Figura 3.62. Placa de la electrónica de potencia.</i> .....	70
<i>Figura 3.63. Conexión de la red eléctrica a la placa de potencia y de esta al horno.</i> .....	71
<i>Figura 3.64. Conexión del acondicionador de señal al Arduino y a la fuente de alimentación.</i> .....	71
<i>Figura 3.65. Conexión de los leds de aviso de temperatura del horno al Arduino.</i> .....	72
<i>Figura 3.66. Conexiones eléctricas entre el Arduino y la pantalla OLED.</i> .....	72
<i>Figura 3.67. Conexiones eléctricas entre el ventilador, el Arduino y la alimentación.</i> .....	73
<i>Figura 3.68. Vista frontal y posterior de la caja.</i> .....	73
<i>Figura 3.69. Vista en planta de la caja con los sistemas conectados.</i> .....	74
<i>Figura 3.70. Código para la creación de la figura.</i> .....	74
<i>Figura 3.71. Código para la creación de los axes.</i> .....	75
<i>Figura 3.72. Código para la creación de botones, cuadros de texto y cajas de texto.</i> .....	76
<i>Figura 3.73. Código de las funciones asociados a los botones.</i> .....	78
<i>Figura 3.74. Código de las funciones asociadas a los botones y comienzo para graficar los axes.</i> .....	79
<i>Figura 3.75. Código de para actualizar los axes de la interfaz.</i> .....	80
<i>Figura 3.76. Código para declaración de variables e inclusión de librerías (parte 1).</i> .....	81
<i>Figura 3.77. Código para declaración de variables e inclusión de librerías (parte 2).</i> .....	82
<i>Figura 3.78. Código del Setup.</i> .....	83
<i>Figura 3.79. Código del loop.</i> .....	84
<i>Figura 3.80. Código de la función reset, ventilación y iluminación.</i> .....	85
<i>Figura 3.81. Código de la función control manual.</i> .....	86
<i>Figura 3.82. Código del procedimiento control automático.</i> .....	87
<i>Figura 3.83. Código del procedimiento pantalla y enviar datos.</i> .....	87
<i>Figura 3.84. Código del procedimiento del controlador PID.</i> .....	88
<i>Figura 3.85. Código para actualizar la pantalla OLED.</i> .....	89
<i>Figura 3.86. Escalones realizados para obtener las funciones de transferencia.</i> .....	90
<i>Figura 3.87. Código para recortar los datos del modelado del horno (parte1).</i> .....	91
<i>Figura 3.88. Código para recortar los datos del modelado del horno (parte2).</i> .....	92
<i>Figura 3.89. Código para recortar los datos del modelado del horno (parte3).</i> .....	93
<i>Figura 3.90. Ventana para cargar datos en system identification.</i> .....	94
<i>Figura 3.91. Ventana de process models en system identification.</i> .....	94
<i>Figura 3.92. Ensayo de verificación (Repetida de la figura 1.2 por conveniencia).</i> .....	95
<i>Figura 3.93. Verificación del modelo obtenido en el primer escalón ascendente (Repetida de la figura 1.2 por conveniencia).</i> .....	96
<i>Figura 3.94. Verificación del modelo obtenido en el segundo escalón ascendente</i> .....	96
<i>Figura 3.95. Verificación del modelo obtenido en el tercer escalón ascendente.</i> .....	97
<i>Figura 3.96. Verificación del modelo obtenido en el primer escalón descendente.</i> .....	97
<i>Figura 3.97. Verificación del modelo obtenido en el segundo escalón descendente.</i> .....	98
<i>Figura 3.98. Verificación del modelo obtenido en el tercer escalón descendente.</i> .....	98
<i>Figura 3.99. Esquema de Simulink para la simulación del controlador obtenido teóricamente.</i> .....	101
<i>Figura 3.100. Respuesta del sistema en lazo cerrado ante escalón unitario.</i> .....	101
<i>Figura 3.101. Respuesta del sistema en lazo cerrado ante un escalón de 150 de amplitud.</i> .....	102
<i>Figura 3.102. Esquema de Simulink para la simulación de seguimiento de referencias.</i> .....	102
<i>Figura 3.103. Respuesta del sistema en lazo cerrado ante seguimiento de referencias.</i> .....	103

<i>Figura 4.1. Señal alterna troceada por una señal PWM medida con osciloscopio.</i> .....	105
<i>Figura 4.2. Prueba de funcionamiento del PWM en una bombilla.</i> .....	106
<i>Figura 4.3. Prueba de funcionamiento del acondicionador de señal.</i> .....	106
<i>Figura 4.4. Prueba de funcionamiento de los Leds y la pantalla OLED.</i> .....	107
<i>Figura 4.5. Interfaz gráfica de Matlab para el Arduino.</i> .....	108
<i>Figura 4.6. Ventanas para guardar los datos.</i> .....	108
<i>Figura 4.7. Cambio de modo manual a automático (Repetida de la figura 1.3 por conveniencia).</i> ....	109
<i>Figura 4.8. Prueba de seguimiento de referencias (Repetida de la figura 1.4 por conveniencia).</i> .....	110

## Índice de tablas

<i>Tabla 1.1. Cronograma del proyecto</i> .....	7
<i>Tabla 3.1. Correspondencia entre Vs y Vo.</i> .....	44
<i>Tabla 3.2. Modelos obtenidos.</i> .....	95



## Resumen

En la actualidad, la tecnología de montaje superficial para el desarrollo de circuitos impresos es la comúnmente empleada por los diseñadores, puesto que permite una reducción considerable del tamaño de los circuitos electrónicos, así como una disminución de los costes de producción, dejando la tecnología de orificio pasante para el desarrollo de prototipos, ya que permiten una rápida implementación y facilidad a la hora de soldar los componentes electrónicos. El empleo y testeo de prototipos es un paso previo a la comercialización de un producto final o también para afianzar los conocimientos en diseño de circuitos electrónicos a nivel educativo.

En este trabajo de fin de grado, se ha modificado un horno de bajo coste para que siga perfiles de temperatura que se emplean en la soldadura por reflujo, con el objetivo de poder realizar prototipos de montaje superficial a nivel educativo. Para ello se ha utilizado un sensor de temperatura PT100, se ha diseñado la electrónica para acondicionar la señal del sensor, la electrónica de potencia para controlar la resistencia del horno, además del modelado del sistema y el diseño de una estrategia de control. Los componentes utilizados, en la mayor parte posible, han sido reciclados de otros dispositivos averiados promoviendo la reutilización de componentes electrónicos.

**Palabras clave:** Soldadura por reflujo, control PID, Arduino.



## Abstract

Currently, surface mount technology is commonly employed by designers for the development of printed circuit boards, as it allows for a significant reduction in the size of electronic circuits, as well as a decrease in production costs. Through-hole technology is left for prototype development, as it allows for quick implementation and ease of soldering electronic components. The use and testing of prototypes are a preliminary step in the commercialization of a final product or to solidify knowledge in electronic circuit design at an educational level.

In this bachelor's thesis, a low-cost oven has been modified to follow temperature profiles used in reflow soldering, aiming to prototype surface mount technology at an educational level. To achieve this, a PT100 temperature sensor has been used, and electronics have been designed to condition the sensor signal, as well as power electronics to control the oven's resistance, in addition to system modelling and the design of a control strategy. Components have been mostly sourced from recycled parts of other malfunctioning devices, promoting the reuse of electronic components.

**Keywords:** Reflow soldering, PID control, Arduino.



## 1 Introducción

Los primeros circuitos impresos datan de 1900 debido a la necesidad de contar con equipos eléctricos de reducido tamaño con el fin de reemplazar las complejas y extensas conexiones de los dispositivos electrónicos y eléctricos. Después de la segunda guerra mundial Paul Eisler contribuyó al desarrollo de la tecnología de circuito impreso al emplear un material aislante revestido de cobre como material base, en el cual, se dibuja el patrón de circuito impreso de manera que el cobre descubierto se elimina por ataque químico. También se propuso una nueva generación de placas en donde ambos lados del material base contienen conductores a través de orificios, esta tecnología se denomina de orificio pasante [1].

Los problemas más comunes con la tecnología orificio pasante, donde se utiliza la soldadura manual a nivel educativo e incluso profesional, son las de articulación perturbada, junta fría, junta sobrecalentada y humectación insuficiente [2], que pueden dar lugar a un mal funcionamiento del circuito. La ventaja de esta tecnología es que se pueden crear prototipos de una manera rápida.

En la década de 1960 la tecnología de montaje superficial comenzó a reemplazar a la tecnología THT (*Through Hole Technology*) debido a la necesidad de automatizar los procesos de creación de PCBs (*Printed Circuit Board*), con el fin de realizar una producción masiva para la electrónica de consumo, desarrollándose nuevas técnicas de soldadura como, por ejemplo, la soldadura por reflujo [3].

Este tipo de soldadura es el método más utilizado para soldar componentes SMD (*Surface Mount Device*) a nivel profesional, consta de cinco fases o zonas, cada una con un perfil térmico, que son la evaporación, activación, precalentado, reflujo y enfriado, proporcionando resultados más precisos y fiables que la soldadura manual [4].

El perfil térmico es una consideración crítica en el proceso de soldadura por reflujo. En las máquinas profesionales la soldadura por reflujo se perfila colocando varios termopares en distintas zonas y en diferentes componentes de la placa que se está procesando, fijando estos con una soldadura o epoxi. Esto permitirá obtener una comprensión más completa del perfil térmico [5].

Los perfiles pueden medirse en estado cargado o sin carga, la medición que proporcionan los termopares se realiza a través de cables especiales, es decir, resistentes a altas temperaturas o utilizando un dispositivo de registro que puede pasar a través del horno de reflujo mediante el uso de una cubierta. El perfil del horno medido se muestra visualmente, como un gráfico, pero los datos sin procesar frente al tiempo también se pueden procesar por Excel u otros programas estadísticos [5].

En cuanto al control del perfil de temperatura, los hornos profesionales ajustan la velocidad de la cinta transportadora y algunos tienen la capacidad de controlar la tasa de convección, es decir, la velocidad a la que los gases calientes llegan al producto. Esta velocidad se ajusta mediante la adecuación de la presión utilizando un ventilador. El control de la velocidad del ventilador, por tanto, es un elemento clave. Además, algunos hornos emplean herramientas software para generar perfiles automatizados facilitando el desarrollo de creación de perfiles específicos, no obstante, al emplear dichas herramientas se debe tener en cuenta la tasa de convección [5].

## 1.1 Motivación del proyecto

Los circuitos electrónicos, como toda la tecnología en general, ha evolucionado a lo largo del tiempo, pasando de los primeros circuitos impresos que eran de un tamaño considerable a los que nos encontramos actualmente en el mercado de dispositivos electrónicos.

En la actualidad, los circuitos electrónicos se usan para numerosas aplicaciones de diferentes ámbitos de la ciencia e ingeniería. La evolución de la tecnología de circuitos impresos se ha puesto de manifiesto sobre todo en el ámbito empresarial, donde los profesionales del sector diseñan sus circuitos con la tecnología de montaje superficial, empleándolos incluso cuando se realizan prototipos antes de comercializar el producto final.

A nivel educativo los conocimientos de electrónica se adquieren de manera teórica y se utilizan simuladores para contrastar los resultados obtenidos. Luego, se realizan montajes en *protoboards* para verificar que el circuito funciona como se debería esperar, empleando la instrumentación que suele existir en los laboratorios de electrónica como osciloscopios, fuentes de alimentación, polímetros, etc.

La mayoría de las escuelas de ingeniería dan un paso más y hacen que sus estudiantes se inicien en la fabricación de prototipos de forma manual, diseñando circuitos de una sola capa, empleando una placa de cobre con una resina fotosensible, en donde, se imprimirá el circuito diseñado mediante una insoladora, después mediante ataque químico se eliminará el cobre que no esté impregnado de la resina dejando al descubierto el circuito final, luego se taladrarán los pads de los componentes y se soldarán.

En asignaturas específicas como, por ejemplo, diseño de sistemas electrónicos, que se imparte en la Universidad de Almería, se da el primer contacto con la fabricación de prototipos a nivel profesional, realizándose una práctica de un sistema embebido basado en microcontrolador y otra placa con aplicación en radiofrecuencia. Estas placas se diseñan y se envían los archivos *gerber* generados a una empresa especializada para que las fabrique y los alumnos suelden los componentes, no obstante, en muchas ocasiones algunos componentes electrónicos no se encuentran disponibles en tecnología de orificio pasante, sino de montaje superficial, estos componentes son difíciles de soldar con un cautín, sobre todo si su tamaño es pequeño, generando una mala soldadura y posible rotura por choque térmico del componente.

Con este trabajo fin de grado, se pretende conseguir que un horno de bajo coste permita seguir perfiles de temperatura empleados en la soldadura por reflujo, dando la posibilidad, si se logran los objetivos, que los alumnos se introduzcan y diseñen prototipos con componentes de montaje superficial utilizando el horno y experimentando con él.

## 1.2 Objetivos

El objetivo principal de este proyecto es la modificación de un horno de bajo coste para seguir perfiles de temperatura que se utilizan en la soldadura por reflujo. Para ello, se añadirá un sensor de temperatura PT100 que se encargará de medir la temperatura dentro del horno, se diseñará la electrónica de instrumentación para acondicionar la señal proveniente del sensor, la electrónica de potencia que se encargará de activar y desactivar la resistencia del horno, se modelará el sistema del horno mediante su función de transferencia, utilizando *System Identification Toolbox* de Matlab, y se diseñará una estrategia de control para controlar la temperatura, empleando un microcontrolador.

Con el fin de llevar a cabo este trabajo se deben alcanzar los siguientes subobjetivos:

- Funcionamiento del sensor de temperatura dentro del horno.
- Correcto funcionamiento del acondicionador para el sensor y la electrónica de potencia.
- Obtención de la dinámica del sistema a controlar.
- Selección de la estrategia de control.
- Programación en el microcontrolador de la estrategia de control elegida.
- Verificación del correcto funcionamiento del sistema.
- Análisis de los resultados obtenidos.

### 1.3 Resumen de resultados

En este proyecto se ha desarrollado la modificación de un horno de bajo coste para soldadura por reflujo. Primero se modificó el horno para incluir el sensor de temperatura PT100 y se movió la resistencia inferior a la parte superior.

Se diseñó la electrónica de potencia para controlar la resistencia del horno, el acondicionador de señal para medir la temperatura proveniente del sensor, se incluyó periféricos como un ventilador, pantalla OLED (*Organic Light Emitting Diode*) y leds para visualizar información del sistema. Se diseñó una interfaz gráfica en Matlab para la recogida de datos y modelar el sistema, enviar referencias de temperatura. Se programó el Arduino para controlar todos los subsistemas y realizar el algoritmo de control.

En la figura 1.1 se muestra el resultado del sistema una vez conectado todos los subsistemas diseñados, aquí se puede observar cómo se conecta la placa de potencia al horno, el sensor de temperatura al acondicionador de señal. El Arduino conectado a las placas, periféricos y al ordenador mediante USB, también se puede observar la inclusión de una fuente de alimentación proveniente de un ordenador reciclado para alimentar las placas, por último, se aprecia la interfaz gráfica en la pantalla del portátil para la recogida de datos y envío de referencias.

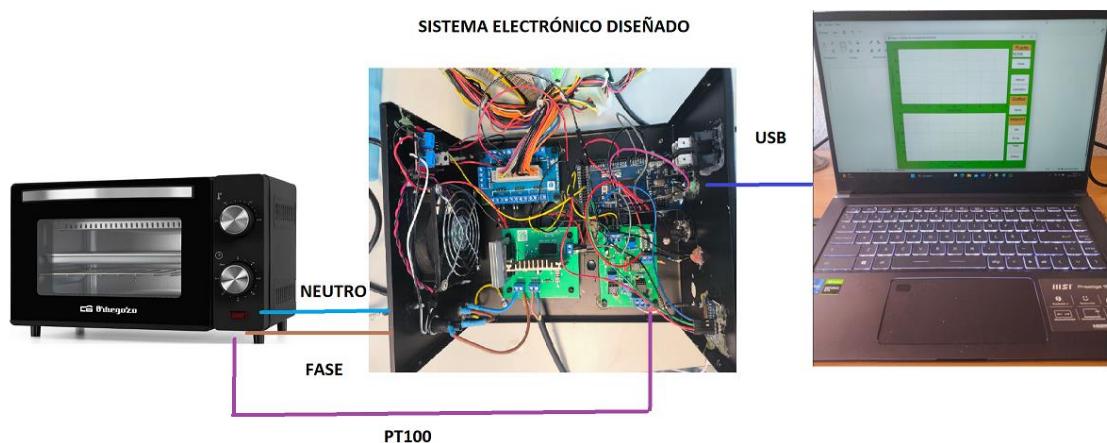


Figura 1.1. Esquema general del sistema.

Una vez comprobado que todos los subsistemas funcionan correctamente, que el Arduino realiza y ejecuta correctamente la programación de los procedimientos, funciones y el algoritmo de control, y que la interfaz gráfica recoge datos y envía correctamente al Arduino se procedió a modelar el sistema.

En la figura 1.2 se muestra el resultado de validación del modelo que se utilizó como modelo nominal para el diseño del controlador. Se puede apreciar que tiene una similitud del 92.56%, por tanto, el modelo se da como válido. Se diseñó un controlador conservador con el modelo nominal más desfavorable. Se utilizó un controlador PID y mediante cancelación polo-cero se obtuvieron los parámetros del controlador de forma teórica, luego se simuló la estrategia de control en *Simulink* y, a continuación, se probó en el sistema real.

En la figura 1.3 se muestra el resultado de la transferencia sin saltos entre el modo manual y el modo automático. Primero se parte en el modo manual dando un escalón del 20 % de PWM y se espera a que el sistema alcance el régimen permanente. Luego se cambia al modo automático en el segundo 3452 se puede apreciar que la señal de control se mantiene constante, después sigue las referencias establecidas.

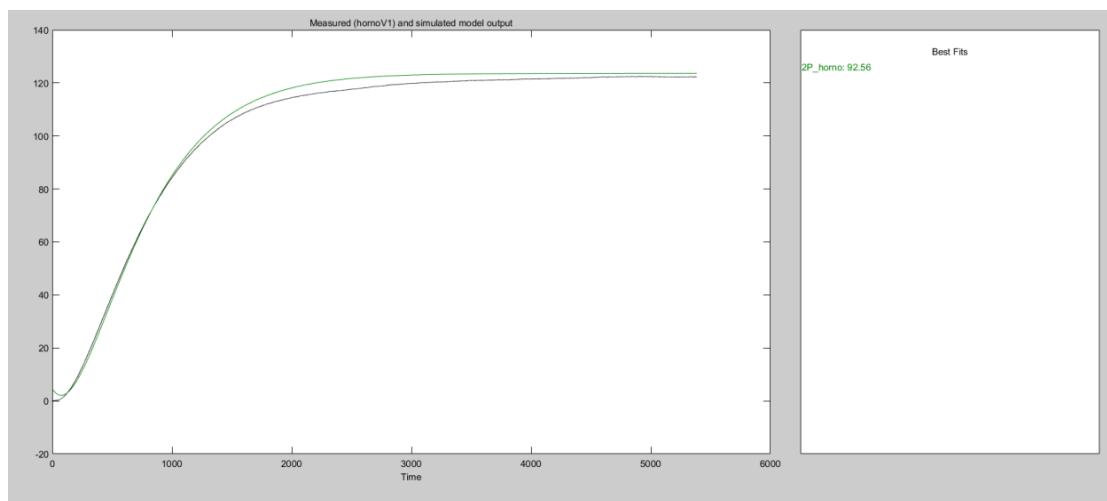


Figura 1.2. Resultado de validación del modelo nominal utilizado en el diseño del controlador.

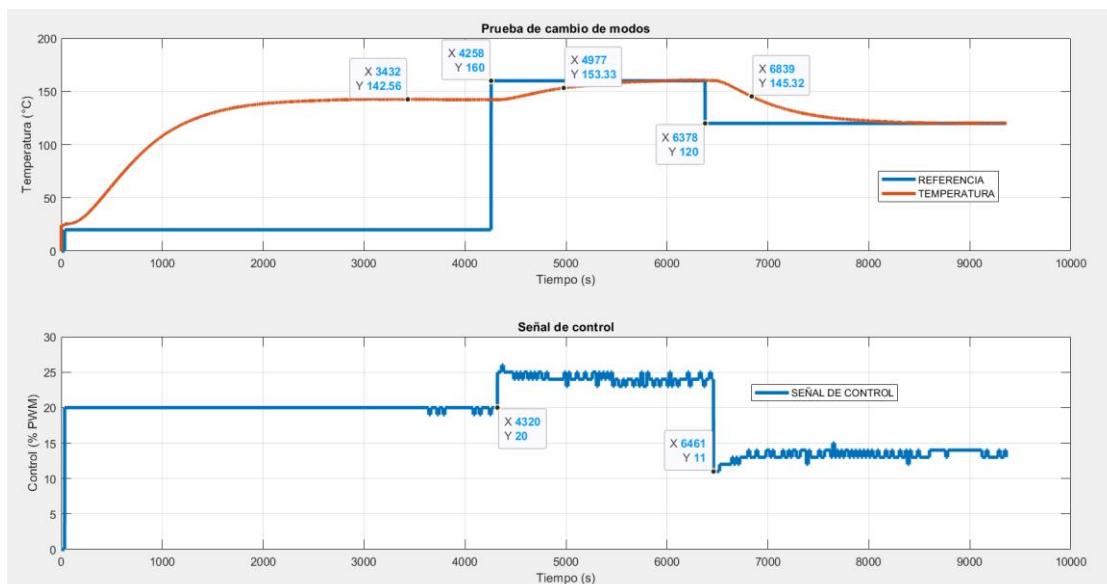


Figura 1.3. Resultado de la transferencia sin saltos.

En la figura 1.4 se muestra el resultado del seguimiento de referencias de temperatura. Como se puede apreciar el sistema consigue seguir las referencias, pero como es un sistema fuertemente no lineal, las especificaciones teóricas no consiguen cumplirse del todo en el sistema real teniendo unas constantes de tiempo más lentas que en el diseño teórico.

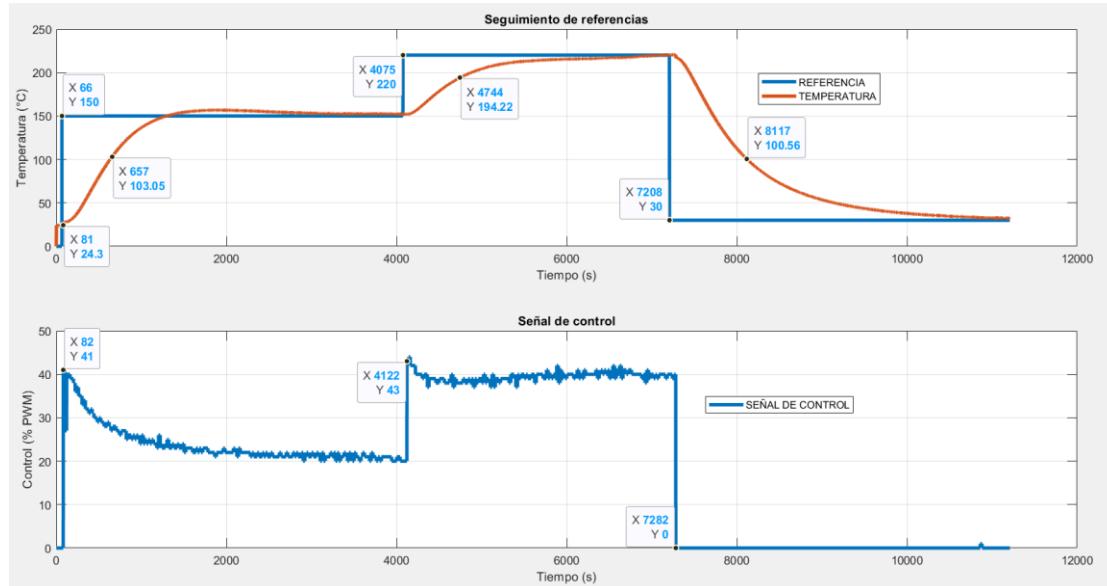


Figura 1.4. Resultados obtenidos del horno al seguir referencias de temperatura.

## 1.4 Uso de las competencias adquiridas

En la elaboración de este trabajo fin de grado se ha empleado una serie de competencias específicas adquiridas en el grado de Ingeniera Electrónica Industrial. Durante el diseño de la electrónica se han desarrollado las competencias CRI5, CTEE2, CTEE3, CTEE4, CTEE5, CTEE6. En el modelado del horno se utilizó la competencia CTEE7. Para el diseño de la estrategia de control se usó la competencia CTEE11. También se aplicó la competencia CB3 para programar el Arduino. Además, se ha puesto en práctica la competencia CT4 para resolver los problemas que han surgido en la elaboración de este trabajo y resolverlos de forma creativa, empleando el razonamiento crítico, de esta forma, se valoró el impacto social y medioambiental en las soluciones técnicas empleando la competencia CT7.

Por otra parte, las competencias básicas utilizadas en este proyecto fueron: la capacidad de emitir juicios (CB3), la capacidad de comunicar y aptitud social (CB4), la habilidad para el aprendizaje (CB5). Con referencia a las competencias transversales se han aplicado: conocimientos básicos de la profesión, la habilidad en el uso de las TIC, la capacidad de resolver problemas, comunicación oral y escrita en la propia lengua, capacidad de crítica y autocrítica, compromiso ético, competencia social y ciudadanía global.

## 1.5 Estructura de la memoria

El trabajo fin de grado está organizado en siete apartados, estructurados de la siguiente manera:

- 1) Introducción:** aquí se introduce el proyecto, se explica la motivación para su realización, se plantean los objetivos principales, se hace un resumen de los resultados obtenidos, así como, los tiempos y fases de desarrollo para realizar este trabajo.

- 2) **Materiales y métodos:** en este apartado se describe la tecnología de reflujo, los componentes electrónicos empleados, el microcontrolador Arduino, el método de modelado empleado en el horno, así como los métodos y estrategias de control empleados.
- 3) **Desarrollo del sistema:** dentro de este apartado se explica de forma detallada los pasos realizados durante el diseño de la electrónica de forma teórica, las simulaciones realizadas para verificar su correcto funcionamiento, las modificaciones realizadas en el horno, la colocación de la sonda, el montaje de la electrónica en una caja reciclada, el modelado experimental y las técnicas de control utilizadas.
- 4) **Resultados y discusión:** se muestran las placas y los subsistemas funcionando, la interfaz gráfica programada, y pruebas en el sistema real para los cambios de modo de manual a automático y el seguimiento de referencias de temperatura.
- 5) **Conclusiones y trabajos futuros:** aquí se presentan las conclusiones que se ha llegado tras la realización del trabajo y se proponen posibles trabajos futuros que podrían mejorar o continuar este proyecto.
- 6) **Bibliografía:** en este apartado se encuentran las referencias de los textos o sitios web consultados en este proyecto.
- 7) **Anexos:** en esta sección se adjuntan los códigos realizados en Matlab, Arduino y los esquemas electrónicos del proyecto.

## 1.6 Fases de realización y cronograma

Este proyecto consta de las siguientes fases:

- **Revisión bibliográfica:** donde se ha consultado bibliografía relacionada con el tema a tratar en este proyecto. Sobre todo, se han consultado libros de texto, páginas de internet, los centros de ayuda de Matlab, Multisim y Arduino, así como tutoriales en YouTube. Esta tarea se llevó a cabo entre los meses de septiembre y diciembre de 2023.
- **Estudio de colocación de la sonda y modificación del horno:** una vez consultada la bibliografía existente, se ha analizado el sistema con una cámara termográfica y se ha decidido donde se colocaría la sonda PT100, así como las modificaciones oportunas del horno, es decir, se cambió la resistencia inferior y se colocó en la parte superior. Durante los meses de noviembre y diciembre de 2023 se realizó esta tarea.
- **Diseño de la electrónica:** en esta fase, se han realizado los cálculos teóricos, las simulaciones necesarias previas, la selección de componentes, el pedido a un proveedor y el diseño del PCB. Esta tarea se realizó durante los meses de enero y febrero de 2024.
- **Montaje de los componentes y comprobación de su funcionamiento:** aquí se ha realizado la soldadura de los componentes electrónicos en la PCB que se diseñó, la calibración del acondicionador de señal, el testeo de los voltajes y corrientes con el multímetro verificando los cálculos teóricos obtenidos. Se incluyó los componentes para controlar el ventilador que enfriará la electrónica de potencia y una pantalla OLED para visualizar los datos más relevantes como temperatura, porcentaje de control, y si está apagado o encendido. Una vez comprobado que la electrónica funciona correctamente se conectó al horno para realizar las pruebas reales. Esta tarea se realizó durante el mes de febrero de 2024.

- **Diseño de la interfaz gráfica en Matlab:** para obtener los datos de una forma interactiva al usuario, se procedió al diseño de una interfaz gráfica en Matlab, se realizó la programación en el Arduino para el envío y recepción de los datos de tiempo, temperatura y señal de control, así como, la recepción y envío en Matlab, visualizándose en tiempo real los datos obtenidos para su posterior procesado con la potencia que ofrece el entorno de Matlab. Esta tarea se ha desarrollado entre los meses de febrero y marzo de 2024.
- **Modelado experimental del sistema:** una vez que se ha montado el sistema y se ha comprobado la correcta obtención de los datos, se ha obtenido el modelo experimental mediante *System Identification Toolbox* de Matlab. Esta tarea se desarrolló en el mes de abril de 2024.
- **Programación de la estrategia de control:** llevado a cabo el modelado del sistema, se simuló la estrategia de control en *Simulink* y se realizó la programación en el microcontrolador de Arduino para posteriormente implementarlo en el sistema real y ver el comportamiento del controlador al seguir referencias. Esta tarea se elaboró en el mes de mayo de 2024.
- **Redacción de la memoria:** en esta última fase, se ha redactado la memoria del proyecto. La redacción de la memoria se ha ido llevando a cabo en paralelo con otras fases. Esta tarea se ha llevado a cabo entre los meses de marzo y junio de 2024.

En la tabla 1.1 se muestra el cronograma con cada una de las fases realizadas en este proyecto. En ella se muestran las horas invertidas en cada una de estas tareas:

Fases	2023				2024				Horas		
	S	O	N	D	E	F	M	A	M	J	
Revisión bibliográfica											70
Estudio de colocación de la sonda y modificación del horno											20
Diseño de la electronica											70
Montaje de los componentes y comprobación de su funcionamiento											15
Diseño de la interfaz gráfica en Matlab											20
Modelado experimental del sistema											20
Programación de la estrategia de control											30
Redacción de la memoria											80
	Total								325		

Tabla 1.1.Cronograma del proyecto

Por motivos laborales y personales se dispuso de poco tiempo para ir al laboratorio alargándose el trabajo final de grado.



## 2 Materiales y métodos

### 2.1 Tecnología SMT

En el campo de la ingeniería electrónica, la tecnología SMT (*Surface Mount Technology*) asegura la fabricación eficiente de dispositivos electrónicos y ha remodelado la industria de componentes electrónicos. La mayoría de los componentes como resistencias, capacitadores, microprocesadores, circuitos integrados, diodos, transistores, etc., se encuentran en una versión SMD [6].

La soldadura en pasta es la técnica que se emplea y garantiza conexiones eléctricas entre los componentes y la PCB. La selección de esta pasta es clave, factores como la composición, el tamaño de la partícula y las propiedades térmicas influyen en la calidad de la soldadura. Utilizando técnicas de impresión la pasta se coloca en los pads de la PCB, siendo un factor clave la precisión y la cantidad colocada en cada pad [6].

Después, se produce la etapa de reflujo controlado, aquí la pasta se funde y se adhiere a los componentes y los pads del PCB. La temperatura y el tiempo tienen que ser controladas de forma precisa para evitar daños térmicos y conseguir uniones confiables [6]. A continuación, se procederá a profundizar cada una de las etapas más importantes en esta tecnología.

#### 2.1.1 Proceso de impresión de soldadura en pasta

Es un proceso crítico que implica depositar con precisión pasta de soldadura. Para ello se utilizan impresoras de serigrafía específicas para SMT, estas máquinas están equipadas con una pantalla de impresión y un mecanismo de regleta que distribuye uniformemente la pasta sobre los pads a través de una plantilla.

Los parámetros de impresión que se ajustan para obtener resultados óptimos son:

- **La velocidad de impresión:** controla la velocidad a la cual la máquina se mueve a lo largo de la PCB. Una velocidad adecuada asegura una distribución uniforme de la pasta y evita problemas como la obstrucción de la plantilla.
- **Presión de la regleta:** ajusta la presión con la que la regleta aplica la pasta sobre la plantilla, una presión bien equilibrada garantiza una capa uniforme sobre las almohadillas.
- **Ángulo de la regleta:** este ángulo afecta la cantidad de pasta depositada, un correcto ángulo evita el arrastre excesivo de pasta.
- **Volumen de pasta:** controla la cantidad de pasta que se deposita sobre la plantilla.
- **Altura de separación:** determina la distancia entre la plantilla y la PCB, una altura adecuada asegura un depósito uniforme sin aplastamiento excesivo de la pasta.

La pasta de soldadura es una mezcla de partículas de aleación de soldadura suspendida en un aglutinante. Existen varios tipos de soldadura en pasta según el tamaño y la cantidad de esferas en la pasta. Las pastas tipo 3, 4 y 5 son las más comunes, por ejemplo, las de tipo 3 tienen las partículas más grandes, entre 24 µm y 45 µm, y se utilizan para componentes grandes y pads de mayor tamaño, las de tipo 4 se utilizan en una gama amplia de aplicaciones y tienen un tamaño de partícula entre 20 µm y 38 µm. Por otro lado, la pasta de tipo 5 contienen partículas pequeñas entre 15 µm y 25 µm, se utilizan para pads de tamaño reducido [7].

Para elevar la calidad y la eficiencia en este proceso se recurre a herramientas estadísticas que permite una compresión profunda de como los diferentes parámetros interactúan y su impacto en los resultados. Tambien se han introducido avances en los sistemas de visión y sistemas de control automatizados para la alimentación automática de la soldadura en pasta sobre la plantilla, ajustes en el soporte inferior, control de cantidad de soldadura y sistemas de limpieza mejorados [7].

En la figura 2.1 se puede apreciar cómo se aplica la pasta de soldadura mediante la regleta sobre la plantilla.

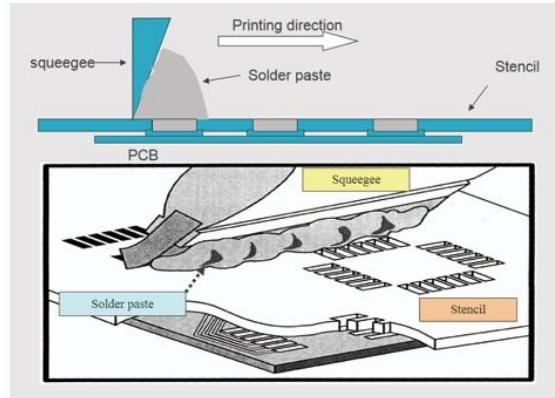


Figura 2.1. Proceso de impresión de pasta de soldadura (Fuente:[8]).

### 2.1.2 Proceso de *pick and place*

En este proceso se caracteriza por la habilidad de recoger componentes y ubicarlos con exactitud en la PCB. Los parámetros críticos que influyen en la calidad y precisión son [9]:

- **Control de la fuerza de colocación:** estas máquinas permiten ajustar la fuerza empleada al colocar cada componente para evitar daños a componentes delicados y garantizar una conexión en los pads del PCB.
- **Sistemas de sujeción por vacío:** estos aseguran que los componentes permanezcan en su lugar durante el proceso de recolección y colocación. Un control preciso del vacío previene de desplazamientos no deseados y mantener la precisión.
- **Sistemas de visión avanzados:** estos permiten la inspección en tiempo real de componentes y PCB, detectando desalineaciones y proceder a su corrección, defectos y ausencias de componentes.

Los cabezales de montaje son los responsables de recoger y colocar componentes existen dos tipos, los de tipo rotativo que poseen un diseño giratorio y permiten recoger componentes desde distintos ángulos son ideales para aplicaciones de alta velocidad y eficiencia. Por otra parte, están los cabezales lineales que se mueven en línea recta y son adecuados cuando se necesita una colocación muy precisa [9].

La alimentación de componentes a la máquina se logra a través de dispositivos llamados alimentadores (del inglés feeders), los métodos de alimentación más comunes son [9]:

- **Tape Feeder:** en este método, los componentes están dispuestos en cintas, cada componente se coloca en una posición específica en la cinta, permitiendo una alimentación secuencial, los anchos de la cinta son de 8, 12, 16, 24 y 32 mm.

- **Tray Feeder:** utiliza bandejas que contiene los componentes en ubicaciones designadas, es adecuado para componentes más grandes o los que no pueden alimentarse mediante cinta, ya sea por tamaño o complejidad, es muy utilizado para componentes sensibles.
- **Stick Feeder:** alimenta componentes en forma de tubos, se utiliza para componentes pequeños y delicados.
- **Bowl Feeder:** utiliza un alimentador tipo tazón vibratorio para suministrar los componentes es ideal para componentes no convencionales que soportan manejo agresivo.

El proceso de *pick and place* ha evolucionado hacia un ecosistema interconectado, al principio estas máquinas operaban como sistemas *off-line*, donde se cargaban manualmente y la comunicación con otros dispositivos era limitada. Sin embargo, con los avances en la industria 4.0 y el internet de las cosas, estas máquinas forman parte de un sistema de producción interconectado. Esta conectividad en tiempo real ha permitido una optimización de la producción y la implementación de estrategias JIT (*Just In Time*), donde la producción se ajusta de manera dinámica a las demandas reales [9]. En la figura 2.2 se puede apreciar una máquina *pick and place* comercial del fabricante NeoDen®.



Figura 2.2. Máquina *pick and place* modelo NeoDen10 de NeoDen® (Fuente:[10]).

### 2.1.3 Proceso de reflujo de soldadura

En el proceso de reflujo la transferencia de calor se erige como la protagonista, esta puede ser llevada a cabo a través de tres mecanismos fundamentales [11]:

- **Radiación:** ondas electromagnéticas transmiten energía, proporcionando el calor necesario para fundir la soldadura.
- **Conducción:** el calor fluye directamente entre los componentes, la soldadura y la PCB, garantizando una distribución uniforme.
- **Convección:** flujos de aire caliente circulan y difunden calor eficazmente, acelerando el proceso de fusión.

El mecanismo de convección es el más utilizado en el proceso de reflujo, cuando el aire caliente asciende desde el calentador en el horno de reflujo, entra en contacto con la soldadura trasfiriendo su energía térmica. Este aire, ahora más liviano, se eleva, permitiendo que el aire frío y denso ocupe su lugar. Este ciclo perpetuo crea una corriente ascendente de calor, distribuyendo uniformemente la temperatura sobre la PCB y los componentes [11].

Para mantener un equilibrio preciso de convección, se ajustan parámetros como la velocidad del flujo de aire, la temperatura y la presión. Sensores de alta precisión miden estos factores en tiempo real, permitiendo un control meticuloso y adaptación constante. El perfil de temperatura asegura que el flujo de aire y la temperatura permanezcan dentro de los límites necesarios para una fusión exitosa [11].

Los dos perfiles mayormente utilizados en SMT son RSS (*Ramp Soak Spike*) y RTS (*Ramp To Spike*) ambos perfiles tienen un objetivo en común, alcanzar temperaturas precisas en momentos específicos durante el proceso de soldadura [11].

El perfil RSS se caracteriza por las siguientes etapas un ejemplo de este perfil se puede apreciar en la figura 2.3 [11]:

- **Ramp (Rampa):** la temperatura aumenta gradualmente hasta alcanzar el punto de precalentamiento (por ejemplo 150°C).
- **Soak (Estabilización):** una vez que se alcanza la temperatura de precalentamiento, se mantiene constante durante un periodo de tiempo específico. Esto permite que los componentes y la PCB alcancen una temperatura uniforme.
- **Spike (Pico):** en esta etapa crítica, la temperatura se eleva rápidamente a la temperatura de fusión (por ejemplo, 220°C) para lograr una soldadura adecuada. Luego, se enfria rápidamente para evitar daños térmicos.

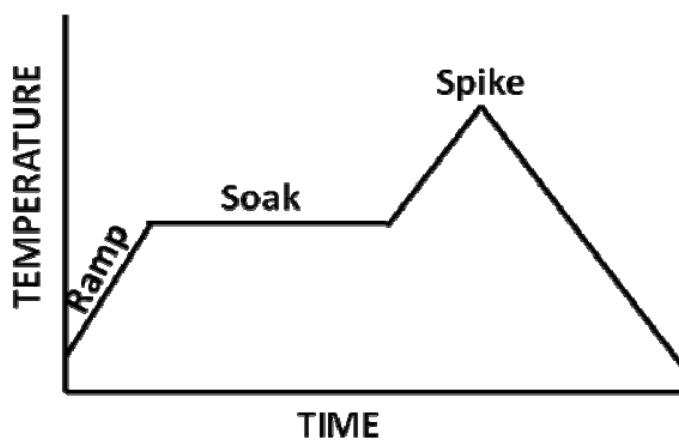


Figura 2.3. Perfil de temperatura RSS (Fuente:[12]).

El perfil RTS se caracteriza por las siguientes etapas [11]:

- **Ramp (Rampa):** similar al perfil RSS, lleva la temperatura al punto de precalentamiento.
- **Spike (Pico):** en lugar de una etapa de estabilización, el perfil RST continúa con una elevación rápida de temperatura hasta el punto de fusión. La estabilización ocurre en este aumento rápido.

Un ejemplo de este perfil se puede apreciar en la figura 2.4.

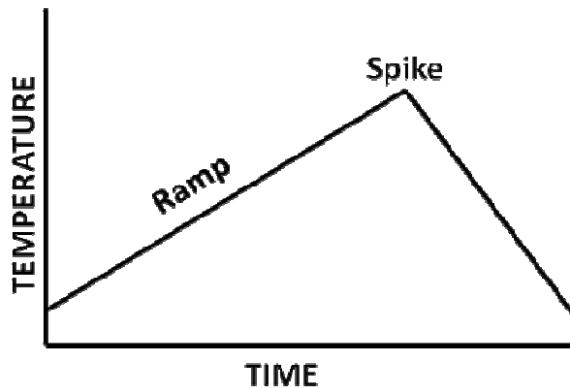


Figura 2.4. Perfil de temperatura RTS (Fuente:[12]).

Los parámetros que se utilizan en los perfiles son [11]:

- **TAL (Time Above Liquids)**: es el tiempo que la temperatura se mantiene por encima del punto de fusión del material de soldadura. Se mide en segundos.
- **Tiempo de permanencia**: es el tiempo durante el cual la temperatura se mantiene constante en la etapa de estabilización. Se mide en segundos.
- **Velocidad de rampa**: determina la velocidad a la que la temperatura aumenta o disminuye en cada etapa del perfil. Una velocidad de rampa controlada es esencial para evitar daños térmicos. Se mide en °C/segundos.
- **Temperatura de precalentamiento**: es la temperatura a la cual la placa y los componentes se calientan gradualmente antes de la etapa de fusión. Un precalentamiento adecuado reduce el estrés térmico y mejora la calidad de la soldadura. Se mide en °C.
- **Temperatura de pico**: es la temperatura máxima alcanzada durante la etapa de fusión. La temperatura de pico es crítica para lograr una soldadura completa y adecuada. Se mide en °C.

El estándar IPC-7530 aborda la medición y el control de perfiles de temperatura durante los procesos de reflujo y soldadura por ola, proporciona directrices para garantizar que los perfiles de temperatura sean adecuados para diferentes tipos de componentes y placas. Mientras que el estándar IPC-7801 detalla los requisitos y métodos para controlar y verificar el rendimiento de los hornos de reflujo incluyendo pautas sobre la calibración de sensores de temperatura, la uniformidad del horno y la validación del proceso [11].

## 2.2 Dispositivos electrónicos discretos

### 2.2.1 MOSFET

Un MOSFET (*Metal-Oxide-Semiconductor Field-Effect-Transistor*) de potencia es un dispositivo controlado por voltaje que requiere solo una pequeña corriente de entrada del orden de nano amperes, presentan una alta velocidad y bajas pérdidas de conmutación, y los tiempos de conmutación son de nanosegundos.

Existen dos tipos de MOSFET, de agotamiento y de enriquecimiento, tienen tres terminales que son la compuerta, drenaje y fuente. Para esta aplicación se ha utilizado un MOSFET de enriquecimiento, estos suelen usarse como dispositivos de conmutación en la electrónica de potencia [13].

### Características de salida

Un MOSFET de enriquecimiento de canal N tiene tres regiones de operación. La primera región es la de corte en donde  $V_{GS} < V_T$ , aquí el dispositivo no deja pasar la corriente por el drenador. La segunda región es la de saturación donde  $V_{DS} \geq V_{GS} - V_T$ , la corriente de drenaje permanece casi constante para cualquier incremento de  $V_{DS}$ , esta zona se utiliza para ampliación de voltaje, la saturación tiene el significado opuesto a los transistores bipolares. La tercera región es la lineal u óhmica donde  $V_{DS} \leq V_{GS} - V_T$ , en la figura 2.5 se puede observar estas tres zonas de funcionamiento [13].

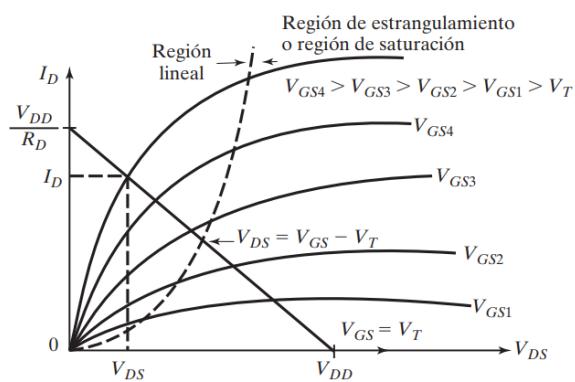


Figura 2.5. Características de salida de un MOSFET tipo enriquecimiento (Fuente:[13]).

### 2.2.2 Resistencias

Los resistencias son los componentes electrónicos más utilizados en circuitos y dispositivos electrónicos. El propósito principal de una resistencia es limitar el flujo de corriente eléctrica y mantener valores específicos de voltaje en un circuito electrónico. Los resistencias también pueden utilizarse para proporcionar un voltaje específico a un dispositivo activo, como un transistor [14].

Una resistencia es simétrica o toma la forma de un dipolo lineal. Simétrica significa que su funcionamiento no depende de la dirección en la que está conectado, ya que la polaridad puede invertirse, produciendo el mismo efecto en el circuito en el que se inserta. Lineal significa que sigue la ley de Ohm [14].

Algunos tipos de resistencias son:

- **Resistencias de carbón:** son económicos y utilizados en aplicaciones generales.
- **Resistencias de película metálica:** tienen mayor precisión y estabilidad.
- **Resistencias de alambre:** utilizados en alta potencia y tienen gran capacidad de disipación de calor.

En este proyecto se han empleado de película metálica, como se muestra en la figura 2.6, puesto que para el diseño del puente de Wheatstone se necesita mucha precisión de las resistencias. Las aplicaciones donde se utilizan las resistencias en este proyecto son para limitar corriente, divisores de tensión y estabilización de señales.



Figura 2.6. Resistencias de diversos valores (Fuente:[14]).

### 2.2.3 Condensadores

Un condensador es un componente electrónico que se utiliza para almacenar energía (carga eléctrica) en un campo eléctrico interno. Está formado por dos placas metálicas con un material dieléctrico (un material que no conduce la electricidad) entre las placas. Los condensadores se utilizan para crear osciladores, temporizadores, aumentar la potencia, etc [15].

Existen varios tipos de condensadores en la figura 2.7 se muestra una clasificación general de los condensadores, cabe destacar que la selección de los condensadores es importante en el diseño de los circuitos electrónicos, en este proyecto se ha utilizado condensadores cerámicos de clase 3 para el desacoplamiento de los circuitos integrados, también condensadores electrolíticos para la fuente de alimentación y uno de mica para realizar el filtro *antialiasing*.

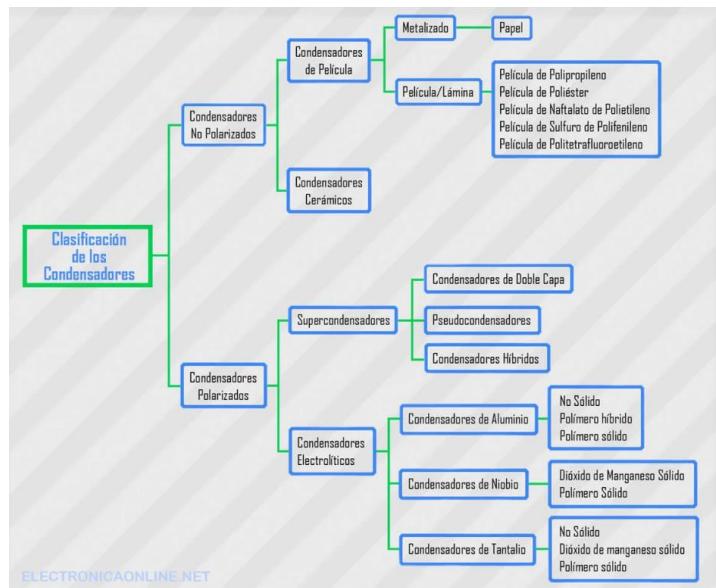


Figura 2.7. Clasificación general de los condensadores (Fuente:[16]).

### 2.2.4 Diodos

Los diodos son dispositivos electrónicos de dos terminales que funcionan como un interruptor unidireccional, es decir, permiten que la corriente fluya solo en una dirección. Estos diodos se fabrican con materiales semiconductores como silicio, germanio y arseniuro de galio.

Los dos terminales del diodo se conocen como ánodo y cátodo. Según la diferencia de potencial entre estos dos terminales, el funcionamiento del diodo se puede clasificar de dos formas [17]:

- Si el ánodo tiene un potencial más alto que el cátodo, entonces se dice que el diodo está en polarización directa y permite que fluya la corriente.
- Si el cátodo tiene un potencial más alto que el ánodo, entonces se dice que el diodo está en polarización inversa y no permite que fluya la corriente.

Todos los tipos de diodos tienen diferentes requisitos de voltaje. Para los diodos de silicio, el voltaje directo es de 0.7V y para los diodos de germanio, es de 0.3V. Por lo general, en los diodos de silicio, la banda oscura en un extremo del diodo indica el terminal del cátodo y el otro terminal es el ánodo.

Existen diversos tipos de diodos como se aprecia en la figura 2.8, los que se utilizaron para este proyecto fueron de tipo Zener, concretamente el modelo BZX85C10-T50R cuya tensión Zener es de 10 voltios que nos servirá para activar la puerta del MOSFET y para estabilizar la tensión de salida de la fuente resistiva.

También se empleó diodos rectificadores, el modelo utilizado es el 1N4007, se usó como un diodo *flyback* dispuesto en paralelo al ventilador ya que es una carga de tipo inductiva, otra aplicación donde se empleó este diodo fue para dirigir la corriente en la fuente resistiva.

Para el puente rectificador se empleó el modelo PB3006 que viene en un encapsulado para facilitar el diseño del PCB.

DIODO	SÍMBOLOGÍA	PICTÓRICO
DIODO RECTIFICADOR		
DIODO LED		
DIODO ZENER		
DIODO PIN		
DIODO TÚNEL		
DIODO SCHOTTKY		
DIODO VARISTOR		
DIODO VARICAP		
FOTODIODO		

Figura 2.8. Diferentes tipos de diodos (Fuente:[18]).

## 2.2.5 Referencia de tensión

Una referencia de tensión proporciona una tensión de continua estable a corto y largo plazo que se utiliza como referencia estándar de otros muchos circuitos, como reguladores de tensión, convertidores A/D, D/A, tensión/frecuencia y frecuencia/tensión, multímetros, sensores, amplificadores logarítmicos, y otros muchos circuitos de instrumentación que tienen como finalidad medir magnitudes físicas de sistemas reales [19].

Los principales requerimientos de una referencia de tensión son la precisión y la estabilidad. La precisión define las diferencias de su salida con referencia al valor nominal, se suele medir como una cota del error absoluto o con el tanto por ciento de error relativo. La estabilidad define la influencia que sobre el valor de salida tienen los cambios de parámetros del entorno, temperatura, tensión de alimentación, carga, etc [19].

Se suele medir en variación absoluta o relativa de la tensión de salida por unidad de variación de la magnitud externa cuya influencia se describe. Para evitar errores debidos a autocalentamiento o interferencias externas intensas, las referencias de voltajes se diseñan con una baja capacidad de proporcionar intensidad de salida (habitualmente en el rango de algunos mA) [19].

En este proyecto se ha utilizado la referencia de tensión programable TL1431, como se muestra en la figura 2.9, para utilizarlo como tensión de referencia al pin  $V_{REF}$  del INA128 y además para el puente de Wheatstone.



Figura 2.9. Referencia de tensión TL1431 (Fuente:[20]).

## Amplificador operacional

Un amplificador operacional, es un circuito integrado que mediante componentes externos permite crear una gran variedad de circuitos. Entre los circuitos construidos mediante amplificadores operacionales y componentes externos se encuentran:

- Inversores.
- No inversores.
- Sumadores.
- Diferenciadores.
- Comparadores.
- Convertidores A/D y D/A.
- Filtros activos.
- Amplificadores de muestreo y retención.

En este proyecto se ha utilizado el amplificador operacional LT1490, como se muestra en la figura 2.10, que es del tipo *rail to rail*, es decir, la salida aprovecha todo el rango de alimentación y no satura antes, este amplificador es útil cuando se alimenta el amplificador con bajas tensiones de alimentación.

No obstante, aquí se utilizó por las características que tiene como baja entrada de offset, un valor alto de CMRR (*Common Mode Rejection Ratio*) y baja deriva térmica para diseñar el acondicionador de señal.

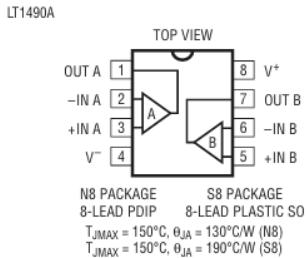


Figura 2.10. Amplificador operacional LT1490 (Fuente:[21]).

## 2.2.6 Amplificador de instrumentación

Un amplificador de instrumentación está hecho a partir de amplificadores operacionales. Está diseñado para tener una alta impedancia de entrada y un alto CMRR que es una medida del rechazo que ofrece la configuración a la entrada de voltaje común. Se puede construir a base de componentes discretos o se puede encontrar encapsulado (por ejemplo, el INA114). Su utilización es común en aparatos que trabajan con señales muy débiles, tales como equipos médicos (por ejemplo, el electrocardiógrafo), para minimizar el error de medida [22].

Para diseñar el acondicionador de señal se empleó el amplificador de instrumentación INA128, cuyo esquema electrónico se muestra en la figura 2.11.

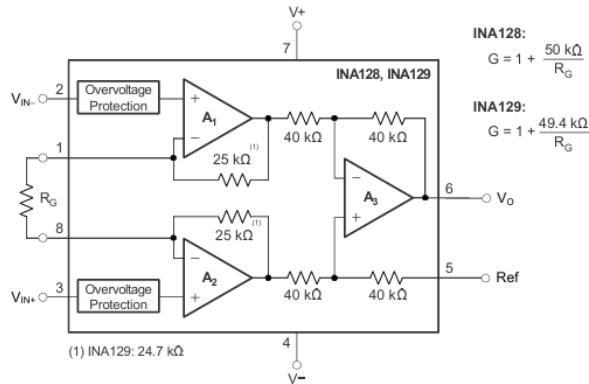


Figura 2.11. Esquema del amplificador de instrumentación INA128 (Fuente:[23]).

## 2.2.7 Optoacoplador

Los dispositivos optoacopladores se utilizan para aislar eléctricamente a dispositivos muy sensibles, como es el caso de un microcontrolador, y para separar dos elementos de circuitos que operan con voltajes muy distintos, evitando daños en la parte que trabaja a un voltaje menor y también como protección ante voltajes inversos o sobrecargas de tensión [24].

En este caso, se aísla galvánicamente la electrónica de control de la electrónica de potencia, permitiendo un control seguro de la activación y desactivación del MOSFET, utilizando el dispositivo CNY65 que permite una frecuencia de conmutación alta y una tensión de aislamiento elevada de 1450 V. Las características principales se muestran en la figura 2.12, donde la corriente máxima que soporta el dispositivo en su entrada es de 75 mA, la tensión máxima entre colector y emisor es de 32 Voltios y la corriente máxima de colector que puede dar es de 50 mA.

<b>ABSOLUTE MAXIMUM RATINGS</b> ( $T_{amb} = 25^{\circ}\text{C}$ , unless otherwise specified)				
PARAMETER	TEST CONDITION	SYMBOL	VALUE	UNIT
<b>INPUT</b>				
Reverse voltage		$V_R$	5	V
Forward current		$I_F$	75	mA
Forward surge current	$t_p \leq 10 \mu\text{s}$	$I_{FSM}$	1.5	A
Power dissipation		$P_{diss}$	120	mW
Junction temperature		$T_j$	100	$^{\circ}\text{C}$
<b>OUTPUT</b>				
Collector emitter voltage		$V_{CEO}$	32	V
Emitter collector voltage		$V_{ECO}$	7	V
Collector current		$I_C$	50	mA
Collector peak current	$t_p/T = 0.5, t_p \leq 10 \text{ ms}$	$I_{CM}$	100	mA
Power dissipation		$P_{diss}$	130	mW
Junction temperature		$T_j$	100	$^{\circ}\text{C}$
<b>COUPLER</b>				
Total power dissipation		$P_{tot}$	250	mW
Ambient temperature range		$T_{amb}$	-55 to +85	$^{\circ}\text{C}$
Storage temperature range		$T_{stg}$	-55 to +100	$^{\circ}\text{C}$
Soldering temperature	2 mm from case, $\leq 10 \text{ s}$	$T_{sld}$	260	$^{\circ}\text{C}$

Figura 2.12. Características del dispositivo CNY65 (Fuente:[25]).

### Razón de transferencia de corriente (CTR)

La razón de transferencia de corriente de un optoacoplador es la proporción del valor de la corriente de salida a la corriente de entrada, es un parámetro equivalente al  $h_{FE}$  de un transistor, el CTR es dependiente de la corriente directa  $I_F$  que fluye a través de la entrada, le afectan los cambios en la temperatura ambiente y varía conforme pasa el tiempo [26]:

En la figura 2.13, se puede apreciar que para una corriente  $I_F$  de 4 mA el CTR toma un valor del 80%, cabe destacar que la curva de la gráfica es para una  $V_{CE}$  de 5 voltios, en el diseño se ha utilizado una corriente de 4 mA y una tensión  $V_{CE}$  de 10 voltios.

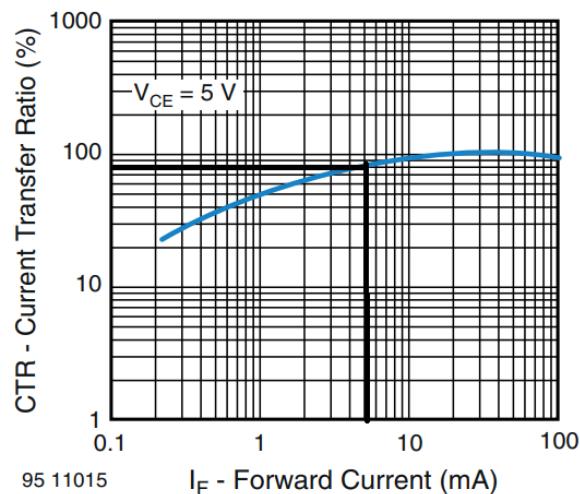


Figura 2.13. Ratio de transferencia de corriente (Fuente:[25]).

### Modos de funcionamiento de un optoacoplador

Básicamente hay dos modos de funcionamiento del fototransistor, modo lineal (no saturado) y modo lógico (saturado). En modo lógico, la señal de salida es lógica alta cuando toma el valor  $V_{CC}$  (tensión de alimentación en el colector) o lógica baja cuando toma 0 Voltios. En modo lineal el voltaje de salida se puede poner como una fracción de  $V_{CC}$  [26].

También se puede establecer si la salida esta invertida o no, con respecto a la entrada, colocando una resistencia de *pull up* o *pull down* respectivamente, en la figura 2.14 se muestra las configuraciones de salida no invertida a la izquierda e invertida a la derecha.

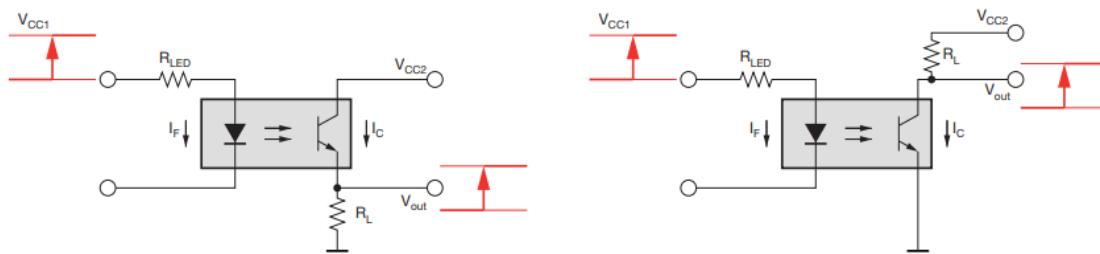


Figura 2.14. Optoacoplador en colector común y en emisor común (Fuente:[26]).

### 2.3 Fuentes de alimentación resistivas

Hay muchas maneras de convertir un voltaje AC en voltaje DC. Tradicionalmente se ha hecho con un transformador y un circuito rectificador o una fuente conmutada (*switching power supply*), pero para alimentar un microcontrolador o circuito que requiere poca corriente, estas soluciones son demasiadas caras y ocupan mucho espacio. Esto es de gran importancia en el mercado actual donde el tamaño de los componentes y el costo, resultan de gran importancia en el diseño. Las fuentes sin transformador resistivas y capacitivas permiten una alternativa de bajo coste a las fuentes de alimentación mencionadas anteriormente, no obstante, tienen una gran desventaja, y es la poca eficiencia que presentan junto a que no están aisladas de la red eléctrica, por ello, se debe tener mucho cuidado al manejar estos circuitos y aislarlos, por ejemplo, con una carcasa plástica [27].

En la Figura 2.15 se muestra una fuente de alimentación resistiva básica sin transformador. En lugar de usar reactancia para limitar la corriente, esta fuente de alimentación simplemente usa una resistencia. Al igual que con la fuente de alimentación capacitiva,  $V_{OUT}$  permanecerá estable siempre que la corriente de salida  $I_{OUT}$  sea menor o igual que la corriente de entrada  $I_{IN}$  [27].

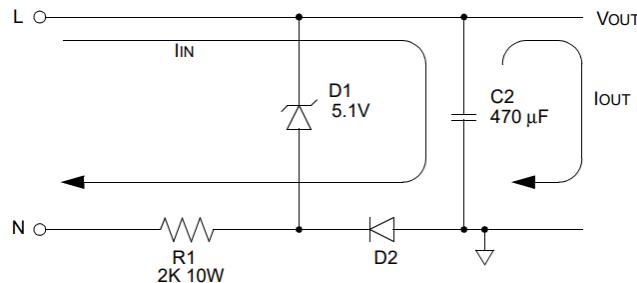


Figura 2.15. Fuente de alimentación resistiva (Fuente:[27]).

La ecuación  $I_{IN}$  es dada por [27]:

$$I_{IN} = \frac{V_{HFRMS}}{R_1} \geq I_{OUT} \quad (2.1)$$

Donde  $V_{HFRMS}$  es el voltaje RMS de media onda de una señal alterna sinusoidal, esta tensión se calcula utilizando la siguiente ecuación [27]:

$$V_{HFRMS} = \frac{V_{PEAK} - V_Z}{2} = \frac{\sqrt{2} * V_{RMS} - V_Z}{2} \quad (2.2)$$

Donde  $V_{PEAK}$  es la tensión de pico de la señal eléctrica y  $V_z$  es el voltaje del diodo Zener, sustituyendo en la ecuación que nos da  $I_{IN}$  queda la siguiente expresión [27]:

$$I_{IN} = \frac{\sqrt{2} * V_{RMS} - V_Z}{2 * R_1} \quad (2.3)$$

## 2.4 Placa de desarrollo

Las placas de desarrollo son dispositivos que se utilizan para el desarrollo y prototipado de sistemas electrónicos y proyectos de ingeniería. Estas placas cuentan con componentes básicos como microcontroladores, sensores, actuadores y otros elementos que permiten la implementación de circuitos y sistemas complejos. Además, las placas de desarrollo suelen contar con un conjunto de pines y conectores que facilitan la conexión con otros dispositivos y periféricos.

### Arduino Mega 2560

Es una placa de desarrollo basada en un microcontrolador ATMEL de 8 bits. Los microcontroladores son circuitos integrados en los que se pueden grabar instrucciones, las cuales se escriben con el lenguaje de programación C++ en el entorno Arduino IDE o Visual Studio Code. Estas instrucciones permiten crear programas que interactúan con los circuitos de la placa [28].

El microcontrolador de Arduino posee lo que se llama una interfaz de entrada, que es una conexión en la que podemos conectar en la placa diferentes tipos de periféricos. La información de estos periféricos que conectes se trasladará al microcontrolador, que se encargará de procesar los datos que le lleguen a través de ellos [28].

El tipo de periféricos que puedes utilizar para enviar datos al microcontrolador depende en gran medida del uso que se esté pensando dar. Pueden ser cámaras para obtener imágenes, teclados para introducir datos, o diferentes tipos de sensores [28].

También cuenta con una interfaz de salida, que es la que se encarga de llevar la información que se ha procesado en el Arduino a otros periféricos. Estos periféricos pueden ser pantallas o altavoces en los que reproducir los datos procesados, pero también pueden ser otras placas o controladores [28].

En el proyecto se ha utilizado el modelo mega2560, como se muestra en la figura 2.16, que tiene las siguientes características:

- Microcontrolador: ATmega2560
- Voltaje Operativo: 5V
- Arquitectura: AVR-RISC
- Tensión de Entrada: 7-12V
- Voltaje de Entrada(límites): 6-20V
- Pines digitales de Entrada/Salida: 54 (de los cuales 14 proveen salida PWM)
- Pines análogos de entrada: 16
- Corriente DC por cada Pin Entrada/Salida: 40 mA
- Corriente DC entregada en el Pin 3.3V: 50 mA
- Memoria Flash: 256 KB (8KB usados por el *bootloader*)
- SRAM: 8KB
- EEPROM: 4KB
- Frecuencia de reloj: 16 MHz



Figura 2.16. Placa de desarrollo modelo mega 2560 de Arduino (Fuente:[28]).

## 2.5 Métodos y estrategias de control

En esta sección se van a comentar las estrategias de control que se han utilizado en el proyecto para controlar el horno.

### 2.5.1 Control PID

El controlador PID es el algoritmo de control más común. La mayoría de los lazos de realimentación se controlan mediante este algoritmo u otro con pequeñas variaciones. Se implementa de muchas formas diferentes como un controlador único o como parte de un paquete DDC (Control Digital Directo) [29].

El algoritmo PID se puede describir como:

$$u(t) = K \left( e(t) + \frac{1}{T_i} \int_0^t e(\tau) d\tau + T_d \frac{de(t)}{dt} \right) \quad (2.4)$$

Donde  $u(t)$  es la señal de control y  $e(t)$  la señal del error de control, que es la diferencia entre la referencia y la salida del sistema. La señal de control es así una suma de tres términos la acción proporcional, la acción integral y la acción derivativa. Los parámetros del controlador son la ganancia proporcional  $K$ , el tiempo integral  $T_i$ , y el tiempo derivativo  $T_d$  [29].

El algoritmo PID se puede representar, en transformada de Laplace, por la función de transferencia:

$$C(s) = K \left( 1 + \frac{1}{sT_i} + sT_d \right) \quad (2.5)$$

A esta forma se le denomina estándar o no interactuante o algoritmo ISA. La acción proporcional, integral, y derivativa son no interactuantes en el dominio del tiempo. Este algoritmo admite ceros complejos, lo cual es útil cuando se controlan sistemas con modos oscilatorios [29].

Otra forma de representar el controlador es la denominada forma serie o forma clásica que tiene la siguiente función de transferencia:

$$C'(s) = K' \left( 1 + \frac{1}{sT_i'} \right) (1 + sT_d') \quad (2.6)$$

La forma serie se obtiene cuando se implementa un controlador como un dispositivo analógico, por la sencillez de realizarlo con dos amplificadores a diferencia del no interactivo que requería tres. Esta representación es atractiva en el dominio de la frecuencia porque los ceros corresponden a los valores inversos de los tiempos derivativo e integral. Todos los ceros del controlador son reales. Se caracteriza porque cualquier modificación de las constantes temporales  $T_i$  y  $T_d$  afecta a las tres acciones. Es también de gran utilidad para el uso de métodos analíticos del tipo de asignación de polos, ya que permite afrontar algunos diseños donde el esquema no interactivo presenta limitaciones [29].

El algoritmo paralelo tiene la siguiente función de transferencia:

$$C''(s) = K'' + \frac{K_i''}{s} + K_d''s \quad (2.7)$$

Es el único algoritmo que permite modificar las tres acciones por separado, es de gran utilidad en el diseño frecuencial y para diseños analíticos ya que permite estudiar las aportaciones que realiza cada una de las acciones del controlador, pero tiene limitaciones a la hora de una interpretación física sobre el efecto de los parámetros [29].

Todos los algoritmos son equivalentes entre sí, pero con algunas restricciones. Para pasar de interactivo a no interactivo:

$$K = K' \left( \frac{T_i' + T_d'}{T_i'} \right) \quad (2.8)$$

$$Ti = T_i' + T_d' \quad si \; T_i > 4T_d \quad (2.9)$$

$$T_d = \frac{T_i' T_d'}{T_i' + T_d'} \quad (2.10)$$

Para pasar de no interactivo a interactivo:

$$K' = \frac{K}{2} \left( 1 + \sqrt{1 - \frac{4T_d}{T_i}} \right) \quad (2.11)$$

$$T_i' = \frac{T_i}{2} \left( 1 + \sqrt{1 - \frac{4T_d}{T_i}} \right) \quad (2.12)$$

$$T_d' = \frac{T_i}{2} \left( 1 - \sqrt{1 - \frac{4T_d}{T_i}} \right) \quad (2.13)$$

Para pasar de paralelo a no interactivo:

$$K'' = K ; \quad K_i'' = \frac{K}{T_i} ; \quad K_d'' = KT_d \quad (2.14)$$

## 2.5.2 Ponderación del punto de consigna

Cuando se está trabajando con controladores tipo PID, una forma eficiente de poder desacoplar los problemas de seguimiento y regulación se conoce como *setpoint weighting*. La idea consiste en tratar la referencia (*setpoint*) y la salida del proceso de forma separada. Esto se puede conseguir expresando un controlador PID de la siguiente forma:

$$U_c(s) = K \left( E_p(s) + \frac{1}{sT_i} E(s) + sT_d E_d(s) \right) \quad (2.15)$$

Donde:

$$E_p(s) = bR(s) - Y(s) \quad (2.16)$$

$$E(s) = R(s) - Y(s) \quad (2.17)$$

$$E_d(s) = cR(s) - Y(s) \quad (2.18)$$

Los parámetros  $b$  y  $c$  son conocidos como pesos de referencia, el peso  $c$  que afecta a la parte de la derivada se escoge generalmente igual a 0 para evitar grandes saltos en la señal de control frente a cambios en escalón. El peso  $b$  toma valores entre 0 y 1, permitiendo reducir las sobreoscilaciones frente a cambios en la referencia [29].

Para este proyecto se utilizó  $b=0$  para evitar que la señal de control cuando se produce un cambio en la referencia pague un pico. En la figura 2.17 se puede apreciar el efecto para distintos valores de  $b$ .

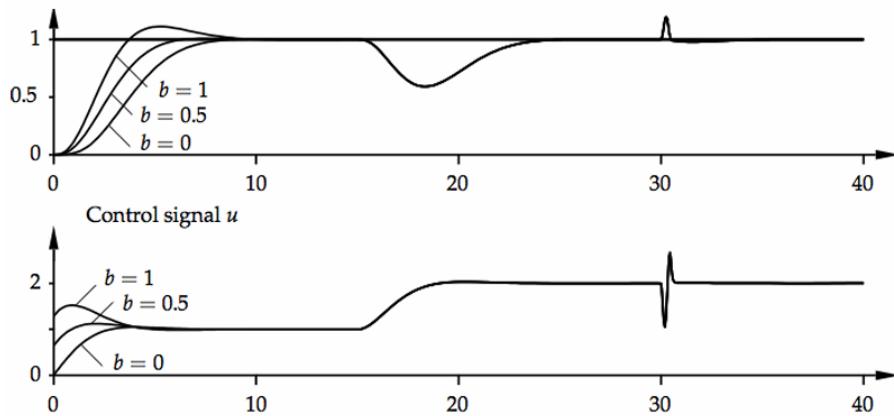


Figura 2.17. Respuestas a cambios de consigna en el punto de referencia (Fuente:[29]).

### 2.5.3 Esquema antiwindup

Todos los actuadores tienen limitaciones. Para un sistema de control con un amplio rango de condiciones operativas puede suceder que la variable de control alcance los límites del actuador. Cuando esto sucede, el lazo de realimentación se rompe y el sistema opera como un sistema en lazo abierto porque el actuador permanecerá en su límite independientemente de la salida del proceso. Si se utiliza un controlador con acción integral, el error puede continuar siendo integrado si el algoritmo no se diseña adecuadamente. Esto significa que el término integral puede hacerse muy grande como se aprecia en la figura 2.18 [29].

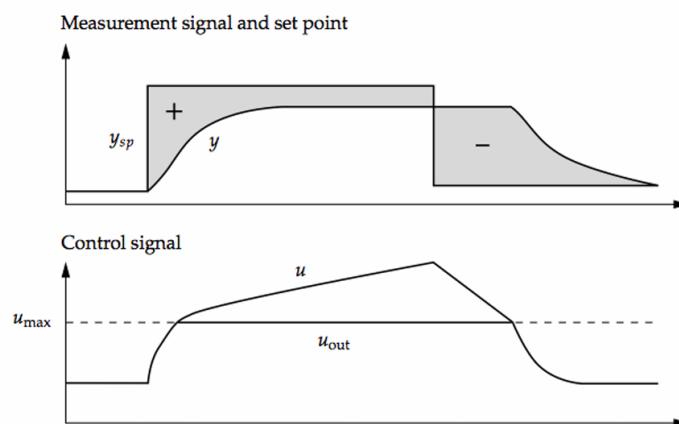


Figura 2.18. Ilustración de la saturación del integrador (Fuente:[29]).

La saturación del integrador o *windup* puede ocurrir en conexión con grandes cambios en el punto de consigna o puede ser causado por grandes perturbaciones. También puede ocurrir cuando se usan selectores de forma que algunos controladores están moviendo un actuador. En el control en cascada, el *windup* puede ocurrir en el controlador primario cuando el controlador se comuta a modo manual [29].

Existen varios métodos para evitar el *windup* como la limitación del punto de consigna, algoritmos incrementales, y el esquema de recálculo y seguimiento (*back calculation*). En la figura 2.19 se muestra el diagrama de bloques de un controlador PID con protección *antiwindup* basado en el recálculo. El sistema tiene un camino de realimentación extra que se genera midiendo la salida real del actuador, o la salida de un modelo del actuador con saturación y formando una señal de error ( $e_s$ ) como la diferencia entre la salida del controlador  $v$  y la salida del actuador  $u$ . La señal  $e_s$  se alimenta a la entrada del integrador a través de la ganancia  $1/T_t$ . La señal es cero cuando no hay saturación, cuando el actuador se satura la señal  $e_s$  es diferente de cero. Se rompe el camino de realimentación normal alrededor del proceso porque la entrada al mismo permanece constante. Hay, sin embargo, un camino de realimentación alrededor del integrador. A causa de esto, la salida del integrador se nivela hacia un valor tal que la entrada del integrador se hace cero [29].

La entrada del integrador es:

$$\frac{1}{T_t} e_s + \frac{K}{T_i} e \quad (2.19)$$

La señal  $e_s$  toma el valor:

$$e_s = -\frac{KT_t}{T_i} e \quad (2.20)$$

En estado estacionario. Como  $e_s=u-v$ , se deduce que:

$$v = u_{lim} + \frac{KT_t}{T_i} e \quad (2.21)$$

Donde  $u_{lim}$  es el valor de saturación de la variable de control. Como las señales  $e$  y  $u_{lim}$  tienen el mismo signo, se sigue que  $v$  es siempre mayor que  $u_{lim}$  en magnitud. Esto previene al integrador del *windup*. La velocidad a la cual se reinicia la salida del controlador está gobernada por la ganancia de realimentación  $1/T_t$  que determina la rapidez con que se reinicia la integral una regla heurística es utilizar la ecuación (2.22).

$$T_t = \sqrt{T_i T_a} \quad (2.22)$$

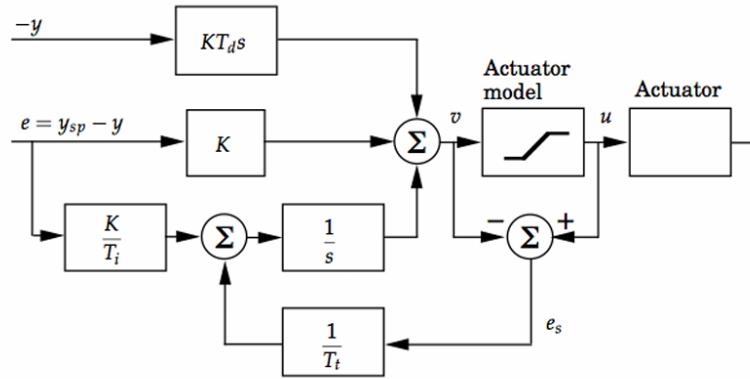


Figura 2.19. Controlador PID con mecanismo de protección antiwindup (Fuente:[29]).

## 2.5.4 Modo Manual y Automático

La gran mayoría de los controladores se pueden arrancar en dos modos, manual y automático. En modo manual, la salida del controlador se manipula directamente por el operador, que usualmente utiliza botones para aumentar o disminuir la salida del controlador. Cuando existen cambios de modos y parámetros, es esencial evitar los transitorios en las comutaciones [29].

**Transferencia sin salto entre modo manual y automático.**

Para controladores con implementación paralela, el integrador del controlador PID se puede utilizar para sumar los cambios en modo manual. Cuando el controlador opera en modo manual como se muestra en la figura 2.20, la realimentación de la salida  $v$  del controlador PID sigue a la salida  $u$ . Con un seguimiento eficiente, la señal  $v$  estará cerca de  $u$  todo el tiempo. Hay un mecanismo de seguimiento similar que asegura que el integrador en el circuito de control manual sigue a la salida del controlador [29].

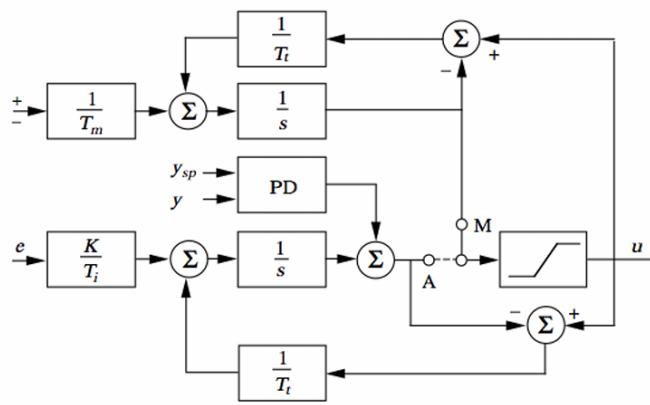


Figura 2.20. PID con implementación paralela que comuta entre el control manual y automático (Fuente:[29]).

En este proyecto se ha implementado la transferencia sin saltos para pasar de modo manual a automático.

### 2.5.5 Método analítico cancelación polo cero

El método requiere un modelo completo del proceso, por ejemplo, para una planta de un polo:

$$P(s) = \frac{k}{(\tau s + 1)} \quad (2.23)$$

La función del lazo para un controlador PI:

$$L(s) = C(s)P(s) = \frac{K(T_i s + 1)}{T_i s} \frac{k}{(\tau s + 1)} \quad (2.24)$$

Cancelando el polo de la planta con el cero del controlador es decir  $T_i = \tau$ :

$$L(s) = \frac{Kk}{s T_i} \quad (2.25)$$

La función de transferencia del lazo cerrado es:

$$T(s) = \frac{L(s)}{1 + L(s)} = \frac{1}{\left(\frac{T_i}{Kk}s + 1\right)} \quad (2.26)$$

Donde la constante de tiempo del lazo cerrado es:

$$\tau_{lc} = \frac{T_i}{Kk} \quad (2.27)$$

Impuesto el tiempo del lazo cerrado se despeja el valor de la ganancia proporcional del controlador:

$$K = \frac{T_i}{\tau_{lc} k} \quad (2.28)$$

### 2.5.6 Método analítico $\lambda$

El método  $\lambda$  es la variante o ampliación del método de cancelación de polos para sistemas con retardo:

$$P(s) = \frac{k}{(\tau s + 1)} e^{-t_r s} \quad (2.29)$$

Utilizando la aproximación en desarrollo de la serie de Taylor para el retardo:

$$e^{-t_r s} \approx (1 - t_r s) \quad (2.30)$$

Se queda la planta con la siguiente función de transferencia:

$$P(s) = \frac{k(1 - t_r s)}{(\tau s + 1)} \quad (2.31)$$

La función del lazo para un controlador PI:

$$L(s) = C(s)P(s) = \frac{K(T_i s + 1)}{T_i s} \frac{k(1 - t_r s)}{(\tau s + 1)} \quad (2.32)$$

Cancelando el polo de la planta con el cero del controlador es decir  $T_i = \tau$ :

$$L(s) = \frac{Kk(1 - t_r s)}{\tau s} \quad (2.33)$$

La función de transferencia del lazo cerrado es:

$$T(s) = \frac{L(s)}{1 + L(s)} = \frac{(1 - t_r s)}{\frac{\tau - Kkt_r}{Kk}s + 1} \quad (2.34)$$

Donde a la constante de bucle cerrado se le denomina  $\lambda$ :

$$\lambda = \frac{\tau - Kkt_r}{Kk} \quad (2.35)$$

Despejando la ganancia proporcional se llega a la siguiente expresión:

$$K = \frac{\tau}{k(t_r + \lambda)} \quad (2.36)$$

## 2.6 Herramientas informáticas

### 2.6.1 Matlab

Matlab es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio. Está disponible para las plataformas Unix, Windows, macOS, y GNU/Linux [30].

Entre sus prestaciones básicas se hallan la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware. El paquete Matlab dispone de dos herramientas adicionales que expanden sus prestaciones, *Simulink* (plataforma de simulación multidominio) y *Guide* (editor de interfaces de usuario - GUI). Además, se pueden ampliar las capacidades de Matlab con las cajas de herramientas (*Toolboxes*); y las de *Simulink* con los paquetes de bloques (*Blocksets*) [30].

En este proyecto se utilizó para la programación de scripts necesarios para la obtención del modelo experimental, así como para programar la interfaz gráfica del horno.

## 2.6.2 Simulink

*Simulink* es un entorno de programación visual que funciona sobre el entorno de programación Matlab. Es un entorno de programación de más alto nivel de abstracción que el lenguaje interpretado por Matlab (archivos con extensión .m). *Simulink* genera archivos con extensión .mdl (de “model”) [31].

Es una herramienta de simulación de modelos o sistemas, con cierto grado de abstracción de los fenómenos físicos involucrados en los mismos. En este proyecto se utilizó *Simulink* para verificar el controlador diseñado que cumple con las especificaciones antes de implementarlo en el sistema real.

## 2.6.3 Guide

Las GUI (también conocidas como interfaces gráficas de usuario o interfaces de usuario) permiten un control sencillo (con uso de ratón) de las aplicaciones de software, lo cual elimina la necesidad de aprender un lenguaje y escribir comandos a fin de ejecutar una aplicación [32].

Las apps de Matlab son programas autónomos de Matlab con un frontal gráfico de usuario GUI que automatizan una tarea o un cálculo. Por lo general, la GUI incluye controles tales como menús, barras de herramientas, botones y controles deslizantes. Muchos productos de Matlab, como *Curve Fitting Toolbox*, *Signal Processing Toolbox* y *Control System Toolbox*, incluyen *apps* con interfaces de usuario personalizadas. También es posible crear *apps* personalizadas propias, incluidas las interfaces de usuario correspondientes, para que otras personas las utilicen [32].

En este proyecto se creó una interfaz gráfica para visualizar los datos en tiempo real de la temperatura del horno, enviar referencias, y guardar los datos recibidos de forma gráfica o con vectores de datos.

## 2.6.4 System Identification Toolbox

*System Identification Toolbox* permite estimar la dinámica de sistemas no lineales utilizando modelos Hammerstein-Wiener y ARX no lineales con técnicas de *Machine Learning* tales como proceso gaussiano, máquina de vectores de soporte, etc [33].

Para llevar a cabo la identificación de los modelos dinámicos necesarios para diseñar los controladores se ha empleado la herramienta *System Identification Toolbox*, que se encuentra dentro de Matlab.

Los pasos necesarios para la identificación de los modelos para este proyecto son los siguientes:

- Aplicar una señal de excitación a la entrada de tipo escalón.
- Recoger los datos de las señales de entrada y salida del sistema.
- Utilizar la herramienta *System Identification Toolbox* y obtener los modelos.
- Verificar los modelos realizando un segundo ensayo.

## 2.6.5 Arduino IDE

El entorno de desarrollo integrado de Arduino, o software Arduino (IDE), se conecta a las placas Arduino para cargar programas y comunicarse con ellas. Los programas escritos con el software Arduino (IDE) se denominan bocetos [34].

El editor contiene cuatro áreas principales, como se observa en la figura 2.21 [34]:

1. Una barra de herramientas con botones para funciones comunes y una serie de menús. Los botones de la barra de herramientas permiten verificar y cargar programas, crear, abrir y guardar bocetos, y abrir el monitor de serie.
2. El área de mensajes, da retroalimentación mientras se guarda y exporta, también muestra errores.
3. El editor de texto para escribir el código.
4. La consola de texto muestra la salida de texto del software Arduino (IDE), incluidos los mensajes de error completos y otra información.

La esquina inferior derecha de la ventana muestra la placa configurada y el puerto serie. Para programar el microcontrolador Arduino se hizo en este entorno de desarrollo, cabe destacar que existe otros entornos de desarrollo como Visual Studio Code pero se decidió utilizarlo por simplicidad.

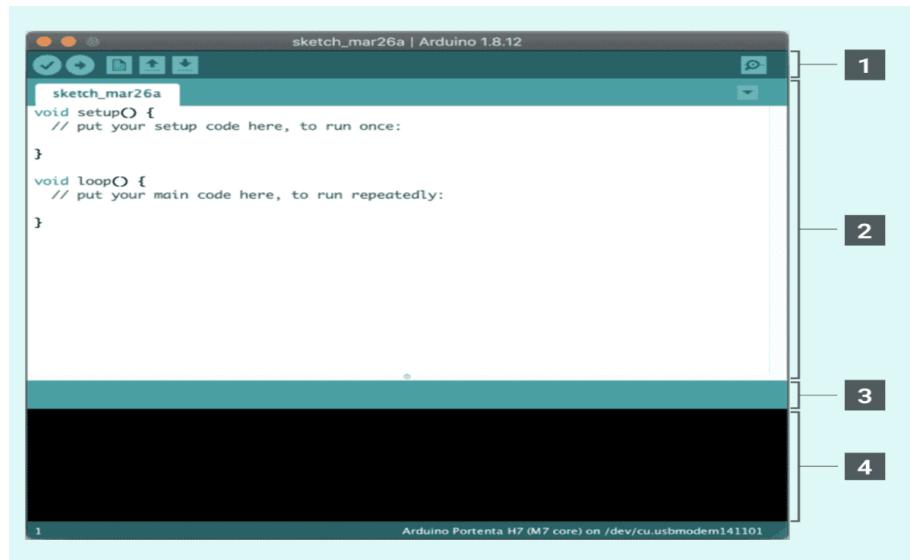


Figura 2.21. IDE de Arduino (Fuente:[34]).



### 3 Desarrollo del sistema

En la figura 3.1 se muestra el sistema finalizado y los bloques desarrollados en este trabajo, a continuación, en los siguientes apartados se explicará cada uno de estos bloques.

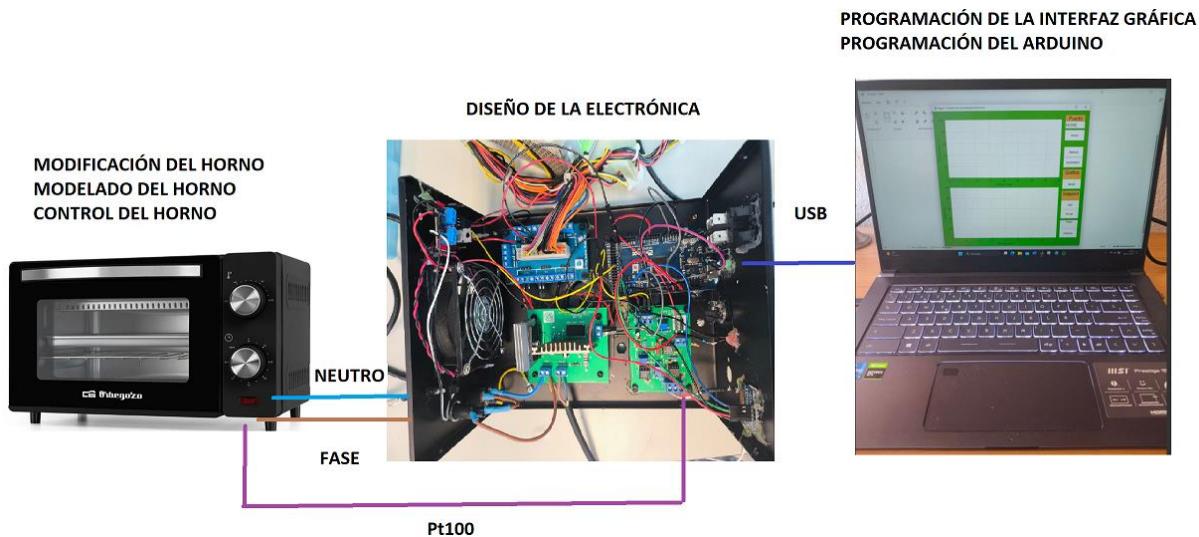


Figura 3.1. Esquema visual del desarrollo del sistema (Repetida de la figura 1.1 por conveniencia).

#### 3.1 Modificación del horno

En este apartado se va a describir las modificaciones que se realizó al horno para la colocación del sensor de temperatura y el cambio de posición de la resistencia eléctrica inferior a la parte superior del horno, así como, la realización de las conexiones eléctricas.

Se ha partido de un horno eléctrico convencional de bajo coste de la marca Orbegozo®, como el que se muestra en la figura 3.2, tiene como características técnicas una potencia de 650W dicha potencia se divide entre las dos resistencias superior e inferior, un regulador de temperatura de hasta 230°C, capacidad de 10 litros, un indicador luminoso y conexión eléctrica de 230 voltios a 50 hercios.



Figura 3.2. Horno eléctrico modelo HO980 de la marca Orbegozo® (Fuente:[35]).

En figura 3.3 se muestra el horno desmontado, en donde se ha quitado la carcasa para ver las conexiones eléctricas y los elementos de control, que en este caso es un termostato que simplemente activa las resistencias eléctricas, si pasa por debajo de una temperatura umbral inferior y las desactiva cuando se supera el umbral superior una vez seleccionada la temperatura de referencia que se ha asignado con el mando exterior.



Figura 3.3. Horno sin modificaciones.

### Colocación del sensor de temperatura

Para colocar el sensor de temperatura PT100 se procedió a hacer un pequeño estudio de como el calor se distribuía dentro del horno para ello se utilizó una cámara termográfica de la marca Huepar® modelo HTi80P. En la figura 3.4 se puede ver el modelo de la cámara termográfica

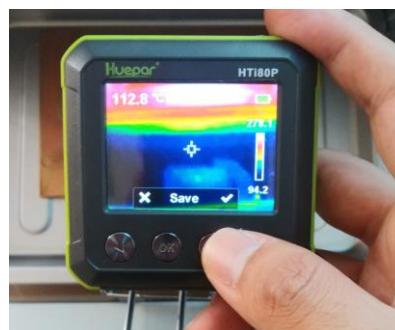


Figura 3.4. Cámara termográfica modelo HTi80P de Huepar®.

A continuación, se configuró la cámara termográfica con una emisividad de 0.9 del acero inoxidable y se puso la temperatura a 230 °C con el mando giratorio, en la figura 3.5 se muestra una vista lateral del horno, aquí se puede apreciar como las resistencias están a 250 °C y la parte superior junto con la parte trasera se encuentra a mayor temperatura que las partes laterales.

En la figura 3.6 se puede apreciar una vista del interior del horno, aquí se puede observar que el calor se concentra en torno a las resistencias que son las fuentes del calor y los laterales próximos a las resistencias.

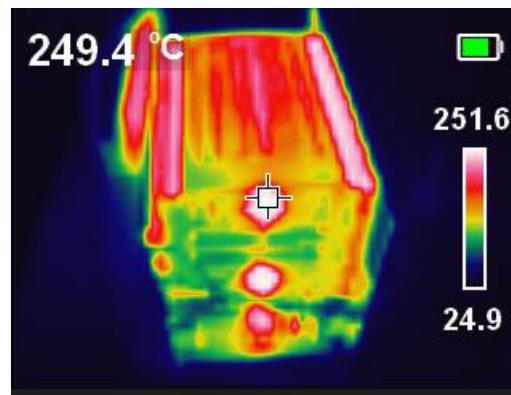


Figura 3.5. Vista termográfica lateral del horno.

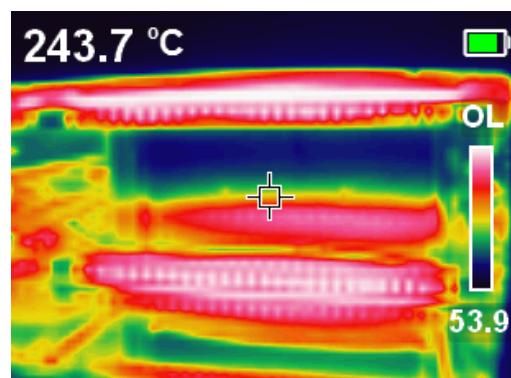


Figura 3.6. Vista termográfica del interior del horno.

Por tanto, se decidió que la sonda de temperatura se colocase en la mitad de las fuentes de calor, y un poco al costado como se aprecia en la figura 3.7, con el fin de medir la temperatura del aire circundante y no irradie directamente la fuente de calor al sensor, también se tuvo en cuenta el espacio para colocar las PCBs en la bandeja que venía con el horno como se puede observar en la figura 3.8.



Figura 3.7. Vista lateral del Sensor PT100 colocado.



Figura 3.8. Vista interior del Sensor PT100 colocado.

### Colocación de la resistencia inferior a la parte superior

También se cambió la resistencia inferior a la parte superior como se puede apreciar en la figura 3.9, con la finalidad que todo el calor sea en una sola dirección, aunque en los equipos profesionales de reflujo el calor inicia de la parte inferior para aprovechar el proceso de convección, por motivos de seguridad y comodidad se eligió la parte superior.

Se desconectó el termostato, que ya no va a ser el elemento de control del sistema, a continuación, se hicieron las conexiones eléctricas, las resistencias están conectadas en serie y se soldó un extremo a la fase (cable marrón) y el otro extremo al neutro (cable azul). Además, se colocó termo retráctil para asegurar el aislamiento eléctrico en los puntos de soldadura para evitar contactos eléctricos y se colocó la toma de tierra en la estructura metálica, como se aprecia en la figura 3.9, por si existe una derivación eléctrica y pueda saltar el diferencial de la instalación eléctrica.

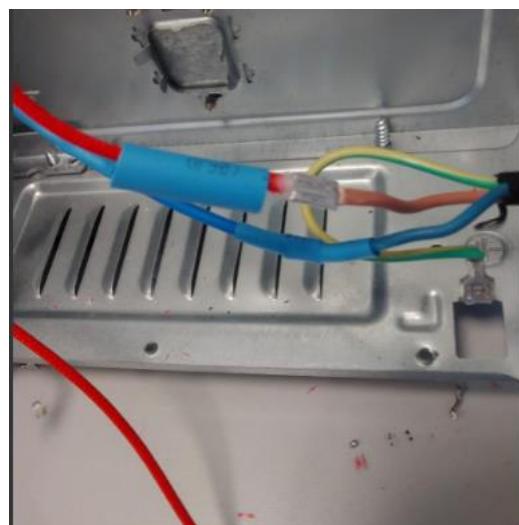


Figura 3.9. Conexiones eléctricas del horno.

## 3.2 Diseño de la electrónica

### 3.2.1 Especificaciones de las placas

En este apartado se va a describir las especificaciones impuestas en el diseño del acondicionador de señal y en la placa de potencia.

#### Especificaciones del acondicionador de señal

Se ha partido de las siguientes especificaciones para el acondicionador de señal:

- Rango de temperatura: 0-282°C
- Corriente máxima por la PT100 de 0.3mA (en el centro del rango de medida)
- Rango de salida del acondicionador entre 0 V (0°C) y 2.56 V (282°C)
- Amplificador de instrumentación INA128
- Referencia de tensión TL1431
- Puente linealizado a tensión constante

#### Especificaciones de la placa de potencia

Se ha partido de las siguientes especificaciones para la placa de potencia:

- Controlar cargas resistivas de 2.3 amperios en corriente alterna.
- Circuito de control aislado mediante optoacoplador.
- Utilización de un MOSFET de potencia como conmutador.
- Alimentación del MOSFET mediante una fuente resistiva.

### 3.2.2 Cálculos teóricos del acondicionador de señal

Se va a recurrir al circuito de la figura 3.10 para acondicionar la señal puesto que presenta un comportamiento lineal la salida del acondicionador con respecto a la variación de la temperatura. Se linealizará la PT100 puesto que el rango de medición de temperatura (0°C -282°C) sobrepasa el rango que mide la PT100 del laboratorio (0°C -250°C), de esta forma obtenemos una ecuación experimental teniendo en cuenta el nuevo rango que se utilizará.

A continuación, se calculará la tensión de referencia en el puente para obtener los 0.3 miliamperios que se estableció en las especificaciones, después se calculará la función de transferencia del acondicionador, la referencia de tensión necesaria para la recta de transferencia del acondicionador (ordenada en el origen de la recta de transferencia), el cálculo para establecer la ganancia del INA128 (pendiente de la recta de transferencia) y finalmente se calculará el filtro *antialiasing*.

#### Linealización de la PT100

El comportamiento de un sensor basado en una variación de resistencia se puede expresar como  $R=R_0 \cdot f(x)$ , con  $f(0)=1$ , para el caso en que la relación sea lineal se obtiene  $R=R_0(1+x)$  [36].

La tensión  $V_s$  es directamente proporcional a las variaciones de una de las resistencias, en este caso a las variaciones de la PT100, se utiliza la configuración del puente de Wheatstone de la figura 3.10 ya que permite la relación lineal antes mencionada.

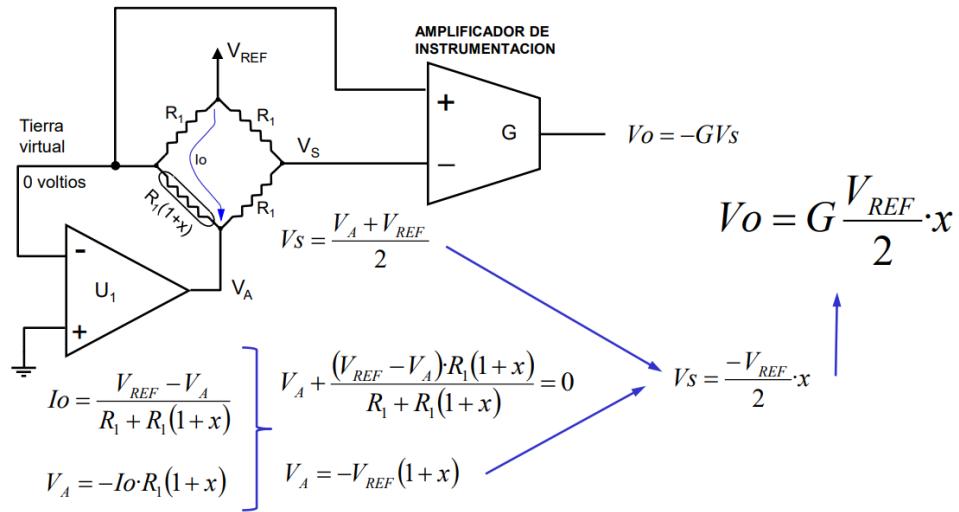


Figura 3.10. Esquema del circuito para el acondicionador

El valor de la resistencia en el puente según la normativa DIN IEC751 para temperaturas entre 0 y 650 grados centígrados viene determinado según la siguiente la ecuación:

$$R_{PT100}(T) = R_0(1 + AT + BT^2) \quad (3.1)$$

Donde  $R_0$  toma el valor de  $100 \Omega$ , los coeficientes  $A$  y  $B$  tienen los valores nominales de  $3.90802 \cdot 10^{-3}$  y  $-5.802 \cdot 10^{-7}$ , respectivamente.

Como se puede apreciar es una relación cuadrática, para temperaturas menores a 100 grados centígrados, el error no es muy crítico para esta aplicación y se optó por aproximar con una relación lineal para las temperaturas entre 100 y 282 grados centígrados, con este método se trató de minimizar el error con respecto a la ecuación que rige la PT100 en dicho rango.

Se utilizó el programa Matlab para realizar dicho cometido, como se puede observar en la figura 3.11, los errores mayores se producen en los extremos, es decir, en 100 y 282 grados centígrados, en donde la variación de temperatura en ohmios es de -0.32 ohmios que equivale a 1 grado centígrado. También se puede observar que existen dos puntos en donde el error es nulo, para las temperaturas de 138.5 y 243.7 grados centígrados. Otro punto importante para analizar el error es en el valor de 260 grados centígrados que es la máxima temperatura que alcanzará el horno antes de la etapa de enfriamiento, aquí se observa un error de 0.11 ohmios.

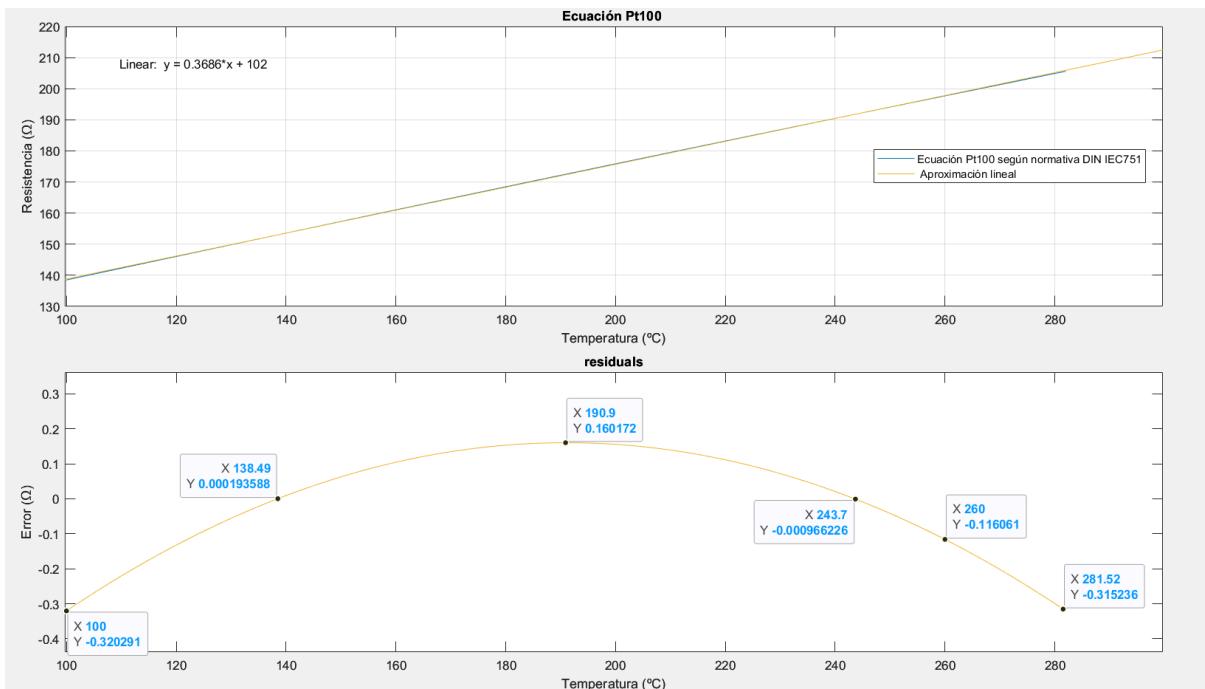


Figura 3.11. Ecuación de la PT100 linealizada y los errores cometidos.

por tanto, la recta que mejor se aproxima es:

$$y = 0.3686 \frac{\Omega}{\text{°C}} x + 102\Omega \quad (3.2)$$

Donde las variables  $y$  e  $x$ , son la temperatura y la resistencia de la PT100 respectivamente.

Haciendo un desarrollo algebraico a la aproximación lineal de la PT100 e igualándola a la ecuación obtenida anteriormente:

$$R(T) = R_1(1 + x) \text{ con } x = \alpha\Delta T \quad (3.3)$$

$$R(T) = R_1 + R_1\beta\Delta T = 102\Omega + 0.3686 \frac{\Omega}{\text{°C}} \Delta T \quad (3.4)$$

$$R(T) = 102[\Omega] + 0.3686 \frac{\Omega}{\text{°C}} \Delta T = 102\Omega \left(1 + \frac{0.3686}{102} \frac{1}{\text{°C}} \Delta T\right) \quad (3.5)$$

$$R(T) = R_1(1 + x) = R_1(1 + \alpha\Delta T) = 102\Omega \left(1 + \frac{0.3686}{102} \frac{1}{\text{°C}} \Delta T\right) \quad (3.6)$$

$$R(T) = 102\Omega \left(1 + 0.0036137 \frac{1}{\text{°C}} \Delta T\right) \quad (3.7)$$

Se obtiene que el valor de la resistencia en el puente es de 102 ohmios que se puede encontrar de forma comercial con una tolerancia del 0.1%, el valor de la constante  $\alpha$  es de  $0.0036137 \text{ } ^\circ\text{C}^{-1}$ .

### Referencia de tensión en el puente

Para calcular el valor de tensión que tendría la referencia de tensión del puente, hay que tener en cuenta que, por la PT100 el fabricante recomienda una corriente máxima de 1mA [37] para evitar el autocalentamiento y afecte a las medidas que proporciona el sensor, luego haciendo un análisis nodal a la resistencia de la rama superior izquierda, podemos despejar el valor que tomaría la referencia de tensión.

$$I_{maxPt100} = \frac{V_{REF} - 0V}{R_1} \quad (3.8)$$

$$V_{REF} = I_{maxPt100} * R_1 = 0.45mA * 102 \Omega = 0.0459V = 45.9mV \quad (3.9)$$

El dispositivo TL1431 se va a utilizar para alimentar el puente de Wheatstone y como tensión de referencia en el INA128. El fabricante recomienda una corriente  $I_K$  mínima de 1mA y máxima de 100 mA para que el dispositivo funcione correctamente [20], se ha elegido 15 mA como valor de diseño.

La configuración de la figura 3.12 es para una tensión de entrada positiva y da una salida  $V_{KA}$  de 2.5V, para obtener el control de la tensión de salida se utiliza la configuración de la figura 3.13. En la que la tensión de entrada es de valor negativo de -12 V para obtener una tensión de salida  $V_{KA}$  negativa de - 2.5 V, con la finalidad de implementar una configuración inversora en la siguiente etapa y tener el control de la ganancia en tensión mediante un potenciómetro en el lazo de realimentación.

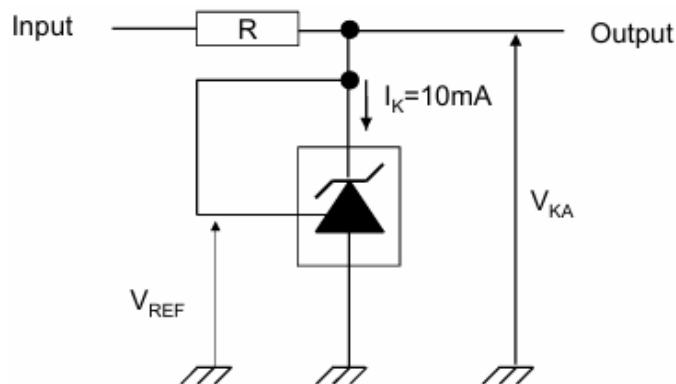


Figura 3.12. Circuito básico del TL1431 (Fuente:[20]).

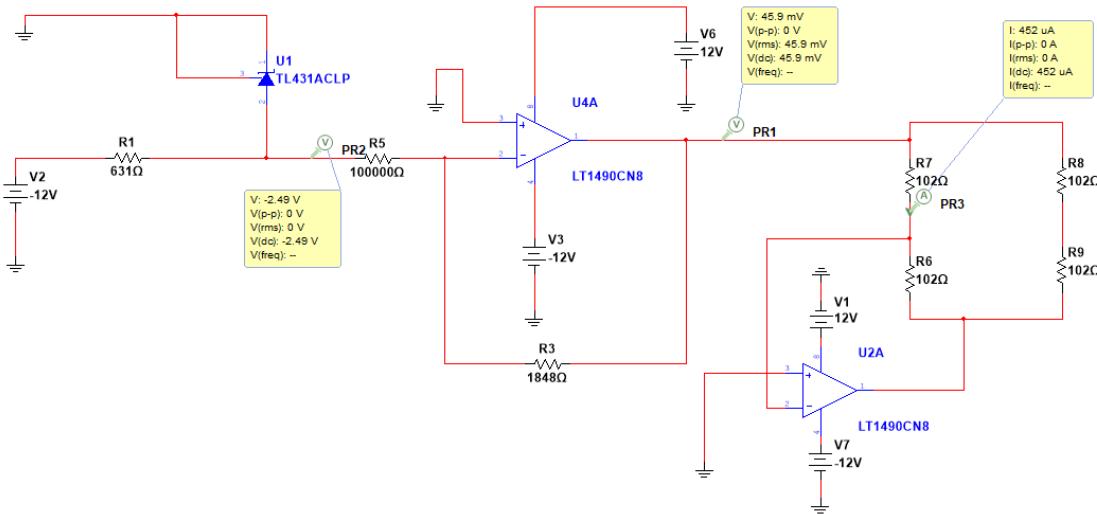
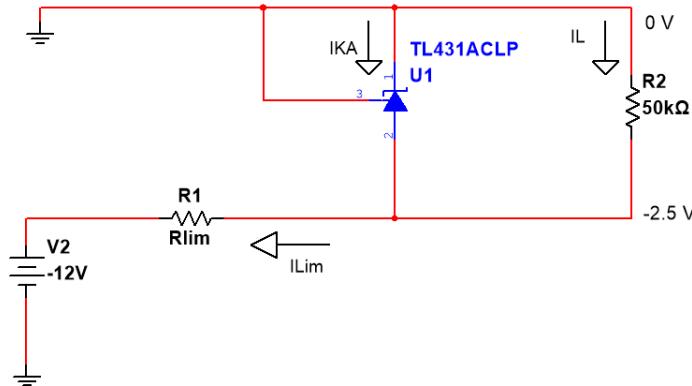


Figura 3.13. Circuito utilizado para proporcionar la alimentación a la PT100.

El TL1431 a su salida tiene como carga el paralelo de las resistencias de  $100\text{ k}\Omega$  cuyo valor de la resistencia equivalente es de  $50\text{k}\Omega$ , por tanto, el circuito equivalente para los cálculos de la resistencia limitadora del dispositivo se muestra en la figura 3.14.


 Figura 3.14. Circuito equivalente para los cálculos de  $R_{Lim}$ .

Se calcula primero la corriente  $I_L$  que pasa por la carga, después,  $I_{Lim}$  es la suma de  $I_{KA}$  y  $I_L$ , el valor de  $I_{KA}$  se ha fijado en 15 mA como valor de diseño mencionado anteriormente. Finalmente utilizando la ley de ohm se calcula la resistencia  $R_{Lim}$ . La resistencia comercial que se ha puesto es de  $680\text{ }\Omega$  con tolerancia del 1%.

$$I_L = \frac{0V - (-2.5V)}{50\text{ k}\Omega} = 50\mu A \Rightarrow I_{Lim} = I_{KA} + I_L = 15mA + 0.05mA \quad (3.10)$$

$$R_{lim} = \frac{-2.5V - (-12 V)}{I_{Lim}} = \frac{12V - 2.5V}{15.05 mA} = 631\Omega \quad (3.11)$$

Para obtener la tensión de 45.9 mV en el puente se utiliza una etapa inversora como se muestra en la figura 3.15. Los valores de las resistencias se calculan de la siguiente manera: se fija la resistencia R1 con 100kΩ y utilizando la ecuación de salida de un amplificador inversor se despeja R2. También se puede apreciar en la figura 3.15 la corriente que se obtiene por la PT100 de 0.45 mA impuestos en el diseño.

$$V_o = -\frac{R2}{R1} * Vi \Rightarrow R2 = -\frac{V_o * R1}{Vi} = -\frac{0.0459 V * 100000 \Omega}{-2.5 V} = 1848 \Omega \quad (3.12)$$

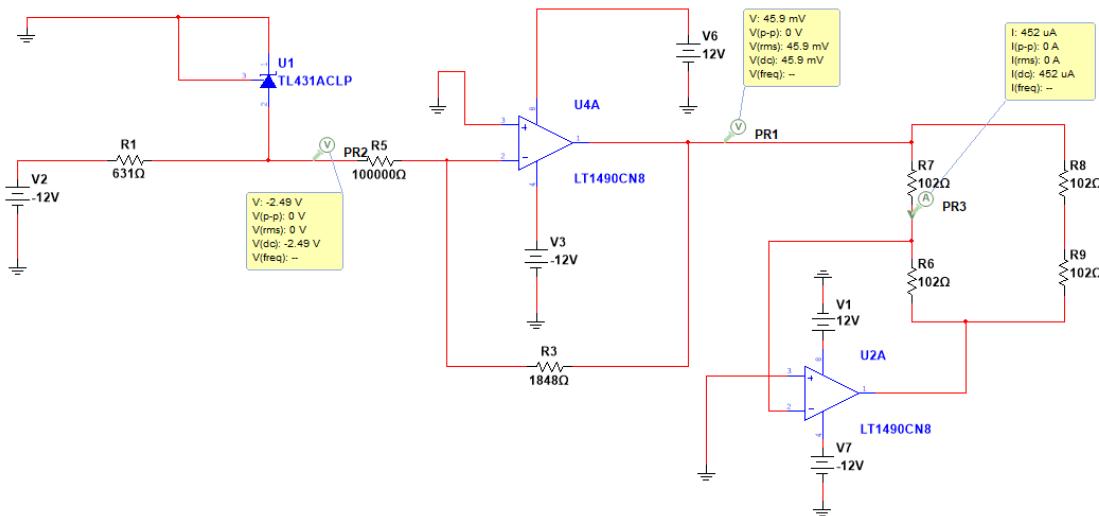


Figura 3.15. Etapa inversora para obtener 45.9 mV en el puente.

No obstante, se necesita un potenciómetro para poder ajustar la tensión de referencia del puente puesto que las tolerancias de los componentes siempre producirán un error que se puede corregir con un ajuste. Para ello se decidió tomar dos valores de salida del amplificador operacional, uno mínimo y otro máximo.

$$\begin{aligned} V_{omin} &= 0.030 V \Rightarrow V_{omin} = -V_{in} \frac{R_{min}}{100k\Omega} \\ V_{omax} &= 0.060 V \Rightarrow V_{omax} = -V_{in} \frac{R_{max}}{100k\Omega} \end{aligned} \quad (3.13)$$

$$\begin{aligned} R_{min} &= -V_{omin} \frac{100k\Omega}{V_{in}} = -0.030 V \left( \frac{100k\Omega}{-2.5 V} \right) = 1200 \Omega \\ R_{max} &= -V_{omax} \frac{100k\Omega}{V_{in}} = -0.060 V \left( \frac{100k\Omega}{-2.5 V} \right) = 2400 \Omega \end{aligned} \quad (3.14)$$

Se ha colocado una resistencia fija de 1200 Ω con un 1% de tolerancia y un potenciómetro de 1kΩ de 25 vueltas para obtener un ajuste fino. En la figura 3.16 se muestra el resultado final.

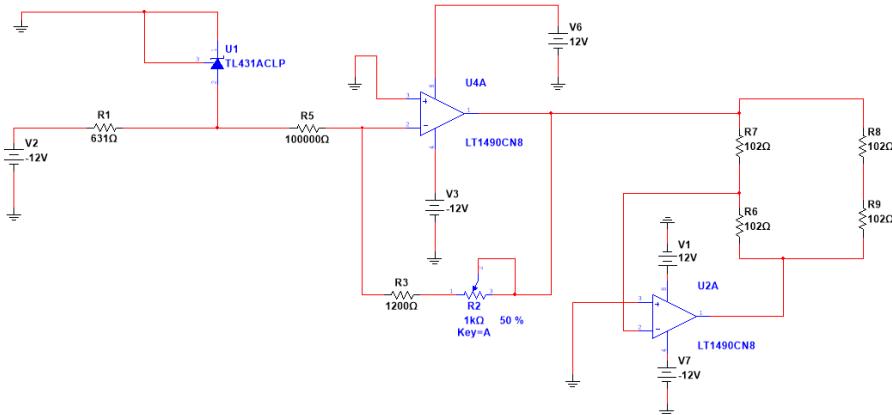


Figura 3.16. Circuito final para alimentar la PT100.

### Cálculo de la función de transferencia del acondicionador

#### Temperatura de equilibrio en el puente

La temperatura en la cual está equilibrado el puente es de aproximadamente 5 grados. Se ha elegido este valor puesto que hemos tomado como referencia la ecuación linealizada de la PT100 del apartado linealización de la PT100, no obstante, normalmente se elige el valor intermedio del rango de medida para asegurar que el error se distribuye en los extremos y en el punto de equilibrio el error es cero.

Según la tabla de la PT100 [38] una temperatura de 5 grados está asociada a un valor de 101.9526 ohms, se ha colocado una resistencia de 102 ohms con una tolerancia del 0.1% en el puente ya que es el valor que más se aproxima de forma comercial. A continuación, se calcula las variaciones de temperatura con respecto a la de equilibrio.

$$T_{equi} = 5^{\circ}\text{C} \quad (3.15)$$

$$\Delta T = T_{max} - T_{equi} = 282^{\circ}\text{C} - 5^{\circ}\text{C} = 277^{\circ}\text{C} \quad (3.16)$$

$$\Delta T = T_{min} - T_{equi} = 0^{\circ}\text{C} - 5^{\circ}\text{C} = -5^{\circ}\text{C} \quad (3.17)$$

#### Salida del puente de medida $V_s$

Una vez calculado las variaciones se sustituye y se obtiene el valor de  $V_s$  teórico:

$$V_s = -\frac{V_{ref}}{2}x ; \text{ con } x = \alpha\Delta T = 0.0036137 \frac{1}{^{\circ}\text{C}}\Delta T \quad (3.18)$$

$$\begin{cases} V_s = -\frac{V_{ref}}{2}x = -\frac{0.046V}{2} \left[ 0.0036137 \frac{1}{^\circ C} (-5^\circ C) \right] = 0.000415 V \\ V_s = -\frac{V_{ref}}{2}x = -\frac{0.046V}{2} \left[ 0.0036137 \frac{1}{^\circ C} (277^\circ C) \right] = -0.023023 V \end{cases} \quad (3.19)$$

### Función de transferencia del acondicionador

Las tensiones de entrada del convertidor analógico digital por defecto en el microcontrolador son de 0 voltios a 5 voltios, pero dichos 5 voltios están generados por un regulador de tensión, el cual tiene una tolerancia peor que una referencia de tensión. Arduino proporciona una tensión de 2.56 voltios generados por una referencia de tensión la cual es mucho más estable con respecto a un regulador, por dicho motivo se ha preferido utilizar los 2.56 voltios en vez de 5 voltios.

A la temperatura de 282°C le corresponde una tensión de salida ( $V_o$ ) de 2.56 voltios y una tensión  $V_s$  del puente de -0.023023 voltios y para 0°C le corresponde una tensión de salida ( $V_o$ ) de 0 voltios y una tensión  $V_s$  del puente de 0.000415 voltios (ver tabla 3.1), con estos dos puntos calculamos la recta de transferencia del acondicionador, es decir, su pendiente y la ordenada en el origen mediante un sistema de ecuaciones.

$V_s$	$V_o$
-0.023023 V	2.56 V
0.000415 V	0 V

Tabla 3.1. Correspondencia entre  $V_s$  y  $V_o$ .

El sistema de ecuaciones formado por las dos rectas es el siguiente:

$$\begin{cases} 2.56 V = -0.023023 V * m + n \\ 0 V = 0.000415 V * m + n \end{cases} \quad (3.20)$$

Multiplicamos la segunda ecuación por -1:

$$\begin{cases} 2.56 V = -0.023023 V * m + n \\ 0 V = -0.000415 V * m - n \end{cases} \quad (3.21)$$

Sumamos ambas ecuaciones:

$$2.56 V = -0.02344 V * m \Rightarrow m = \frac{2.56 V}{-0.02344 V} = -109.21 \quad (3.22)$$

De la segunda ecuación se despeja  $n$ :

$$n = (-0.000415 V) * (-109.21) = 0.04532 V \quad (3.23)$$

Quedando la siguiente recta de transferencia del acondicionador:

$$V_o = -109.21 V_s + 0.04532 V \quad (3.24)$$

### Referencia de tensión para el INA128

La tensión de 45.32 mV, necesaria para obtener la función de transferencia calculada anteriormente y que va a la entrada REF del INA128, se obtiene utilizando una etapa inversora como se muestra en la figura 3.17. Los valores de las resistencias se calculan de la siguiente manera: se fija la resistencia R1 con 100kΩ y utilizando la ecuación de salida de un amplificador inversor se despeja R2.

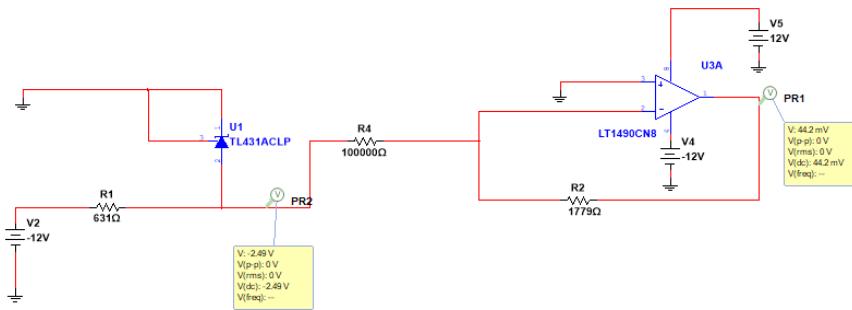


Figura 3.17. Referencia de tensión que va al INA128

No obstante, se necesita un potenciómetro para poder ajustar la tensión de referencia del puente puesto que las tolerancias de los componentes siempre producirán un error que se puede corregir con un ajuste. Para ello se decidió tomar dos valores de salida del amplificador operacional, uno mínimo y otro máximo.

$$V_{omin} = 0.020 \text{ V} \Rightarrow V_{omin} = -V_{in} \left( \frac{R_{min}}{100k\Omega} \right)$$

$$V_{omax} = 0.070 \text{ V} \Rightarrow V_{omax} = -V_{in} \left( \frac{R_{max}}{100k\Omega} \right) \quad (3.25)$$

$$R_{min} = -V_{omin} \left( \frac{100k\Omega}{V_{in}} \right) = -0.020 \text{ V} \left( \frac{100k\Omega}{-2.5 \text{ V}} \right) = 800 \Omega$$

$$R_{max} = -V_{omax} \left( \frac{100k\Omega}{V_{in}} \right) = -0.070 \text{ V} \left( \frac{100k\Omega}{-2.5 \text{ V}} \right) = 2800 \Omega \quad (3.26)$$

Se ha colocado una resistencia fija de 820 Ω con 1% de tolerancia y un potenciómetro de 2kΩ de 25 vueltas para obtener un ajuste fino. En la figura 3.18 se muestra el resultado final.

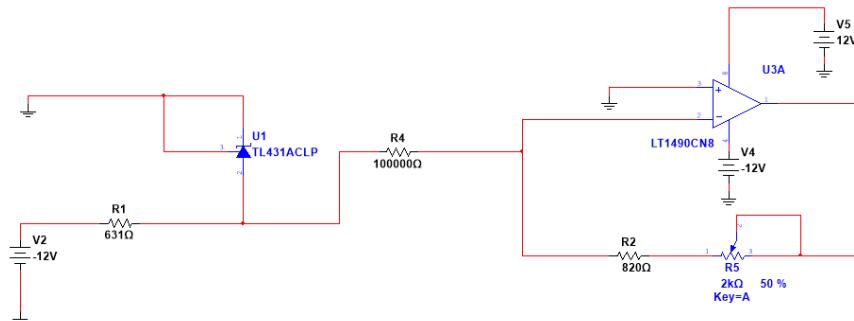


Figura 3.18. Circuito final de la referencia de tensión que va al INA128.

### Cálculo de la ganancia del INA128

La ganancia del INA128 se calcula con la siguiente fórmula que nos dice el fabricante [23]:

$$G = 1 + \frac{50k\Omega}{R_G} \quad (3.27)$$

Despejamos  $R_G$  de la ecuación sustituyendo la ganancia de 106.65 necesaria en el acondicionador:

$$109.21 = 1 + \frac{50k\Omega}{R_G} \Rightarrow R_G = \frac{50k\Omega}{(109.21 - 1)} = 462 \Omega \quad (3.28)$$

Se necesita un ajuste para controlar la ganancia, se dio un valor máximo y mínimo de la ganancia, es decir, 90 y 120 respectivamente. Se calcularon las resistencias asociadas y se fijó la resistencia de valor mínimo en serie con un potenciómetro, los cálculos son los siguientes:

$$\begin{aligned} R_{Gmin} &= \frac{50k\Omega}{(90 - 1)} = 561.8 \Omega \\ R_{Gmax} &= \frac{50k\Omega}{(120 - 1)} = 420.16 \Omega \end{aligned} \quad (3.29)$$

Se colocó una resistencia fija de 420 Ω y 1% de tolerancia en serie con un potenciómetro de 200 Ω de 25 vueltas.

### Cálculo del filtro a la salida del acondicionador

Se debe limitar en banda la señal que proviene del sensor de temperatura para filtrar ruidos producidos por la red eléctrica y los de alta frecuencia, también para establecer la frecuencia de muestreo que debe ser de al menos 2 veces la frecuencia máxima o el ancho de banda de una señal para evitar *aliasing* [39]. Según la hoja de características del microcontrolador Atmega2560, la entrada del convertidor analógico/digital esta optimizada para señales con una impedancia de salida de 10 kΩ o menos [40] y recomiendan utilizar fuentes de baja impedancia con señales que varían lentamente, como es en este caso.

El filtro que se ha diseñado es un filtro pasivo de primer orden y en su salida se ha colocado un seguidor de tensión para que tenga una impedancia de salida baja como recomiendan en el *datasheet* del microcontrolador.

Para realizar los cálculos teóricos, se establece la frecuencia de corte del filtro, que en este caso es de 10 Hz, y se fija el valor del condensador, que tiene que ser de mica ya que presenta una tolerancia menor que otro tipo de condensadores.

Con estos valores se calcula el valor que debe tener la resistencia:

$$R = \frac{1}{F_c * 2\pi * C} = \frac{1}{10Hz * 2\pi * 100nF} = 159155 \Omega \quad (3.30)$$

En la figura 3.19 se muestra cómo se queda la etapa de filtrado, se ha colocado una resistencia normalizada de  $160\text{ k}\Omega$  de 1% de tolerancia, la entrada de la etapa de filtrado sería la salida del INA128, y la salida la entrada del convertidor analógico-digital del Arduino.

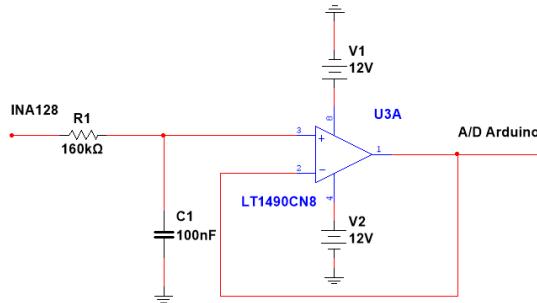


Figura 3.19. Circuito del filtro antialiasing.

### 3.2.3 Cálculos teóricos de la placa de potencia

Para controlar la resistencia eléctrica de  $100\text{ }\Omega$  del horno, se utilizó el esquema electrónico de la figura 3.20, en el cuál, se empleó un transistor MOSFET como conmutador, un optoacoplador para aislar la parte de control de la de potencia. También aprovechando la alimentación de la red eléctrica se hizo una fuente de alimentación básica, para activar la puerta del MOSFET, finalmente se hizo el cálculo del dissipador para el MOSFET. A continuación, se detalla cada uno de estos subcircuitos y sus cálculos teóricos.

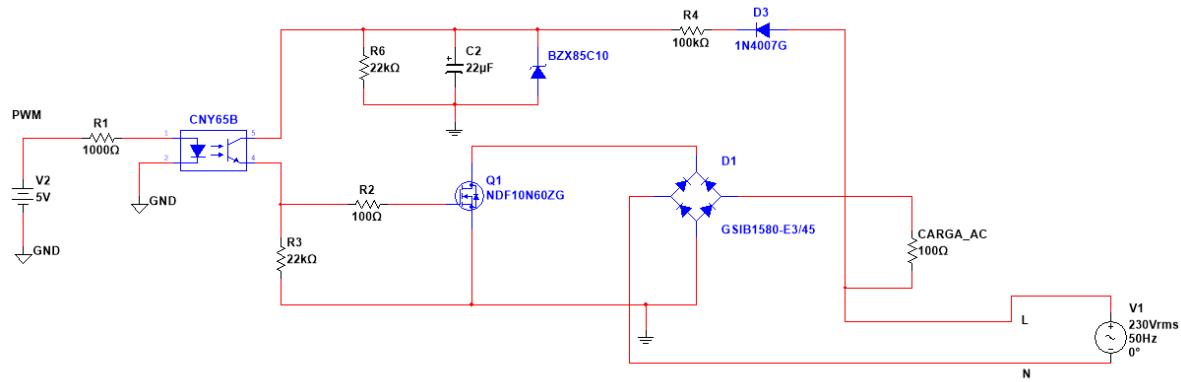


Figura 3.20. Circuito de potencia utilizado.

### Control de cargas resistivas en AC

Se ha utilizado un MOSFET de potencia como dispositivo de control para activar y desactivar la resistencia del horno, puesto que presenta tiempos de conmutación mucho más altos que por ejemplo un triac, no obstante, estos dispositivos se utilizan para activar y desactivar cargas de corriente continua. Por tanto, se tuvo que utilizar un puente de diodos, de esta forma, cuando la señal de corriente alterna este en su semiciclo positivo y el MOSFET se comporte como un interruptor cerrado, la corriente hace el recorrido que se muestra en la figura 3.21.

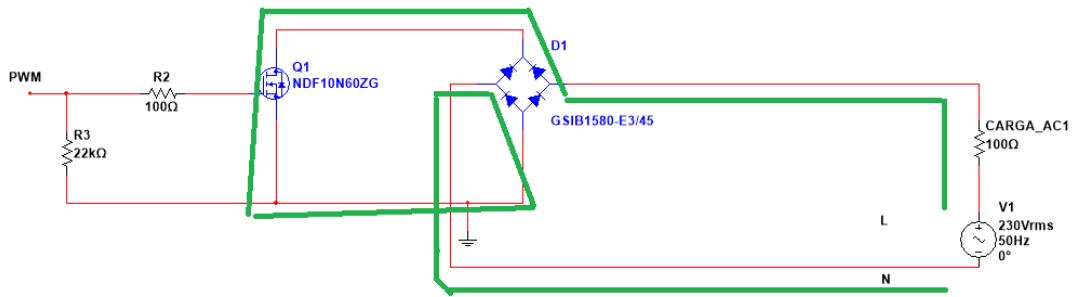


Figura 3.21. Recorrido de la corriente en el semiciclo positivo.

Por el contrario, cuando se produce el semiciclo negativo de la señal eléctrica, con el MOSFET funcionando como interruptor cerrado, la corriente realiza el recorrido de la figura 3.22, por tanto, en ambos semiciclos la corriente pasará por la carga.

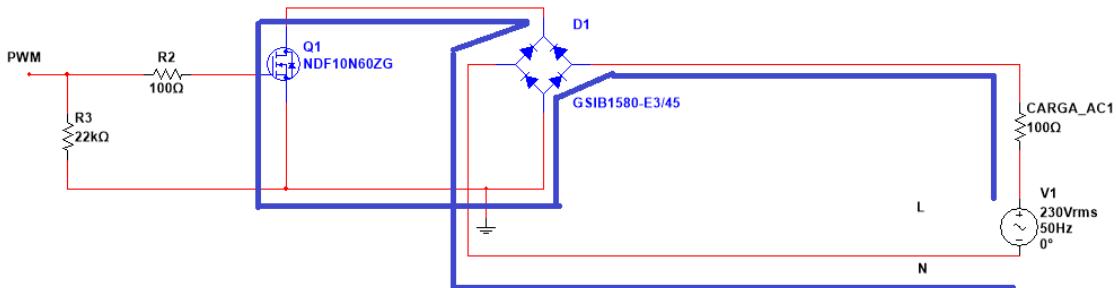


Figura 3.22. Recorrido de la corriente en el semiciclo negativo.

El puente de diodos tiene que soportar una corriente de 3 amperios que atravesará la carga y la tensión de alimentación de la red eléctrica de 325 voltios, se utilizó el modelo PB3006 del fabricante Vishay®. Las características principales que presenta este dispositivo son una tensión y corriente máxima de 600 voltios y 30 amperios respectivamente, como se muestra en la figura 3.23.

PRIMARY CHARACTERISTICS	
Package	PB
$I_{F(AV)}$	30 A
$V_{RRM}$	600 V, 800 V, 1000 V
$I_{FSM}$	240 A
$I_R$	10 $\mu$ A
$V_F$ at $I_F = 15$ A	0.97 V
$T_J$ max.	150 °C
Circuit configuration	In-line

Figura 3.23. Características principales del dispositivo PB3006 (Fuente:[41]).

El MOSFET utilizado es el modelo STF13N60M2 que presenta una tensión de 650 voltios, una resistencia máxima entre drenador y surtidor de 0.38 ohms en la zona lineal del transistor y una corriente de drenador de 11 amperios como se muestra en la figura 3.24.

## Features

Order codes	$V_{DS} @ T_{Jmax}$	$R_{DS(on)} \text{ max}$	$I_D$
STF13N60M2	650 V	0.38 Ω	11 A
STFI13N60M2			

- Extremely low gate charge
- Lower  $R_{DS(on)}$  x area vs previous generation
- Low gate input resistance
- 100% avalanche tested
- Zener-protected

## Applications

- Switching applications

Figura 3.24. Características principales del dispositivo STF13N60M2 (Fuente:[42]).

Según el *datasheet* del MOSFET STF13N60M2 la tensión  $V_{GS(th)}$  de activación es de 4 voltios, pero da una corriente de drenador de 250 microamperios, la figura 3.25 muestra las diferentes tensiones de  $V_{GS}$  y la corriente de drenador  $I_D$ , se ha elegido una  $V_{GS}$  de 10 voltios para activar la puerta del MOSFET, ya que, para esta tensión la resistencia  $R_{DS(on)}$  entre el drenador y la fuente toma el valor más bajo, además la corriente de drenador es de 5.5 amperios.

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
$V_{(BR)DSS}$	Drain-source breakdown voltage	$I_D = 1 \text{ mA}, V_{GS} = 0$	600			V
$I_{DSS}$	Zero gate voltage drain current ( $V_{GS} = 0$ )	$V_{DS} = 600 \text{ V}$ $V_{DS} = 600 \text{ V}, T_C=125^\circ \text{C}$			1 100	$\mu\text{A}$ $\mu\text{A}$
$I_{GSS}$	Gate-body leakage current ( $V_{DS} = 0$ )	$V_{GS} = \pm 25 \text{ V}$			$\pm 10$	$\mu\text{A}$
$V_{GS(th)}$	Gate threshold voltage	$V_{DS} = V_{GS}, I_D = 250 \mu\text{A}$	2	3	4	V
$R_{DS(on)}$	Static drain-source on-resistance	$V_{GS} = 10 \text{ V}, I_D = 5.5 \text{ A}$		0.35	0.38	Ω

Figura 3.25. Características de encendido y apagado (Fuente:[42]).

Para descargar la capacidad parásita  $C_{gs}$  intrínseca al MOSFET entre la puerta y el surtidor cuando la tensión  $V_{GS}$  sea 0 voltios se ha colocado una resistencia de  $22\text{K}\Omega$ , también se ha colocado una pequeña resistencia de 100 ohms entre el emisor del optoacoplador y la puerta del transistor para limitar la corriente por la puerta.

## Fuente para alimentar el MOSFET

Se ha optado por realizar una fuente resistiva. La salida  $V_{OUT}$  permanecerá estable mientras la corriente de salida  $I_{OUT}$  sea menor o igual que la corriente de entrada  $I_{IN}$  [27].

El esquema que se utilizó para los cálculos es el que se muestra en la figura 3.26.

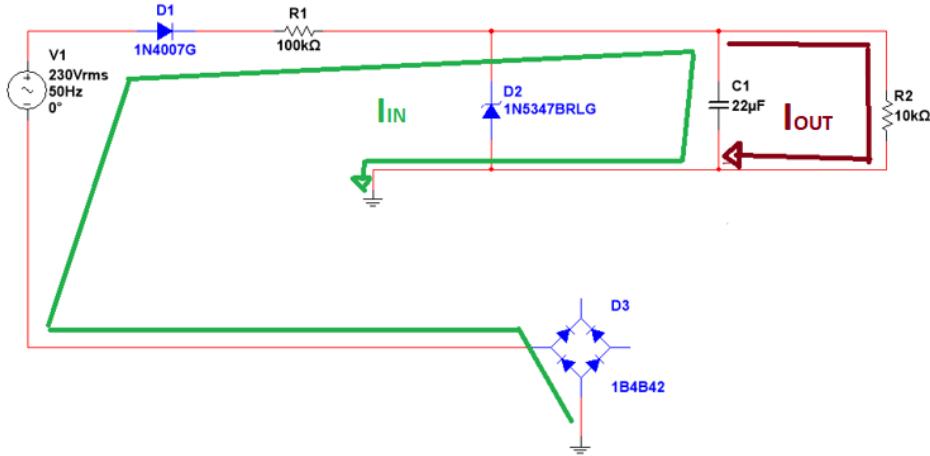


Figura 3.26. Fuente de alimentación resistiva empleada.

Para proporcionar la tensión  $V_{gs}$  necesaria en la puerta del MOSFET, que se encuentra entre 7 y 10 voltios según el *datasheet* del dispositivo STF13N60M2. Se aprovechó la tensión de la red eléctrica, y se hizo pasar por un diodo para rectificar la señal eléctrica y obtener una señal continua pulsante. El diodo que se utilizó es el 1N4007G, este tiene que soportar la tensión de red que son de unos 325 V, la tensión que soporta es de 1000 V, por tanto, cumple los requerimientos.

#### Valor mínimo de $I_{IN}$

Asumiendo un valor mínimo  $V_{RMS}$  de 225V, un valor máximo  $V_Z$  de 10.6 Voltios del Zener BZX85C10, que tiene una tolerancia del 5%, y de 101000 ohmios de R1, con una tolerancia del 1%, se obtiene:

$$I_{INmin} = \frac{\sqrt{2} * 225 V - 10.6 V}{2 * 101000 \Omega} = 1.52 mA \quad (3.31)$$

Se ha colocado una resistencia de valor alto, ya que, no se necesita mucha corriente en la entrada del optoacoplador, además de no disipar mucha potencia a través de la resistencia.

#### Valor máximo de $I_{IN}$

Asumiendo un valor mínimo  $V_{RMS}$  de 235 V, un valor mínimo  $V_Z$  de 9.4 Voltios del Zener BZX85C10, que tiene una tolerancia del 5%, y de 99000 ohmios de R1, con una tolerancia del 1%, se obtiene:

$$I_{INmax} = \frac{\sqrt{2} * 235 V - 10.6 V}{2 * 99000 \Omega} = 1.62 mA \quad (3.32)$$

#### Tensión de salida

La tensión de salida  $V_{OUT}$  se calcula de la siguiente expresión [27]:

$$V_{OUT} = V_Z - V_D \quad (3.33)$$

Donde  $V_Z$  es la tensión Zener y  $V_D$  es la caída de tensión del diodo polarizado directamente.

Teniendo en cuenta las tolerancias de los componentes habrá una tensión máxima y mínima:

$$V_{OUTMAX} = 10.6V - 1.1V = 9.5 V$$

$$V_{OUTMIN} = 9.4 - 1.1 = 8.3 V \quad (3.34)$$

Como se mencionó anteriormente la tensión necesaria para la puerta del MOSFET está comprendida entre 7 y 10 voltios, cumpliéndose los requerimientos. Cabe también mencionar que dependerá del valor de la carga, si se coloca una carga demasiada baja, la corriente de salida  $I_{OUT}$  que demandará el circuito será mayor que la corriente que puede proporcionar la fuente  $I_{IN}$ , haciendo que se baje la tensión de salida para compensar la corriente demandada por la carga. En la figura 3.27 se ha colocado una resistencia de valor de 1 kΩ para observar dicha caída de tensión, como se puede apreciar, la tensión  $V_{OUT}$  cae a 1 voltio siendo incapaz de activar la puerta del MOSFET cuando se produce un valor alto en el PWM. Por tanto, se ha elegido un valor de 22 kΩ.

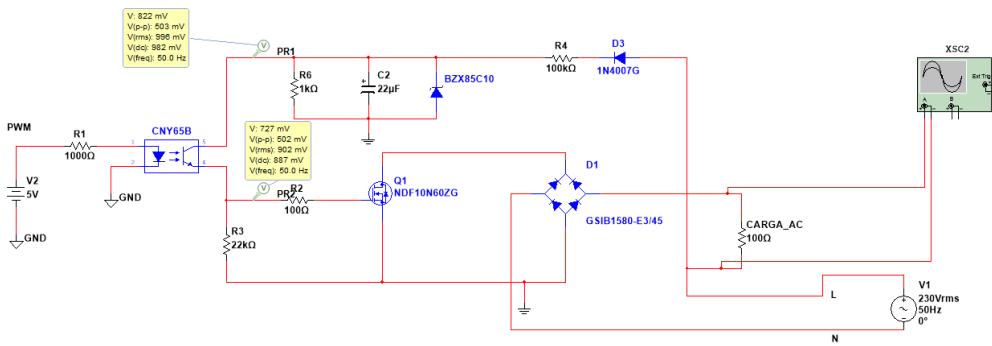


Figura 3.27. Caída de tensión de la fuente de alimentación resistiva para una carga de 1kΩ.

#### Factor de rizado de la fuente de alimentación y tiempo de subida de Vout

En la figura 3.28 se muestra el factor de rizado para diferentes valores de  $\tau$ , mientras más alto sea, menor rizado habrá. El valor de la constante de tiempo es:

$$\tau = R_L * C \quad (3.35)$$

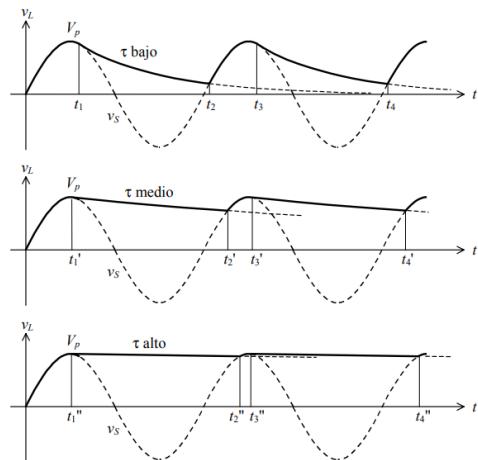


Figura 3.28. Factor de rizado para distintos  $\tau$  (Fuente:[43]).

Como se ha fijado el valor del condensador a  $22 \mu F$ , el rizado junto con el tiempo de estabilización de  $V_{OUT}$ , dependerá del valor de la carga  $R_L$ .

En la figura 3.29 se muestra el factor de rizado y el tiempo de estabilización de  $V_{OUT}$  cuando en el PWM tenemos un valor de 0 Voltios, aquí la carga tiene el valor de  $22 k\Omega$ , como se puede apreciar el rizado es de 0.217 Voltios y el tiempo de subida es de 270 milisegundos.

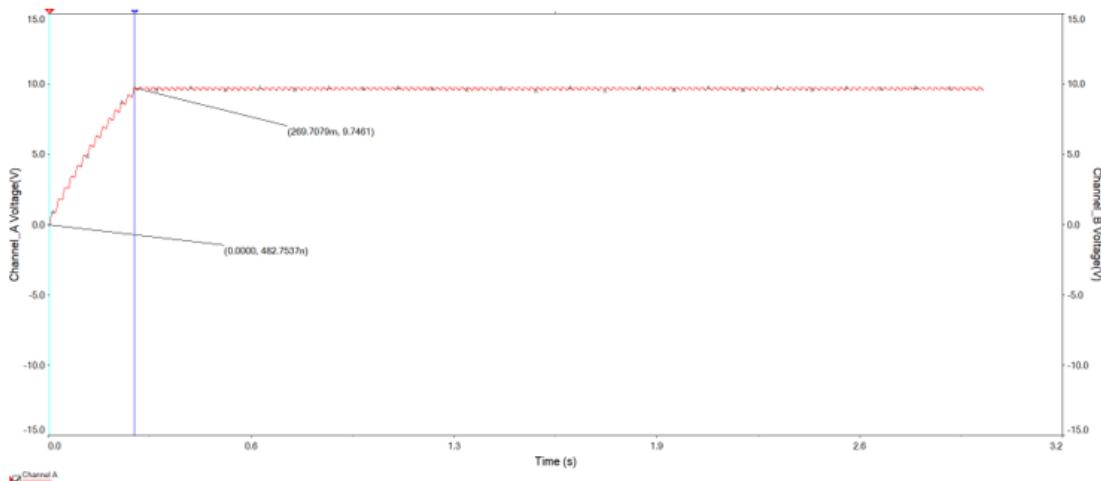
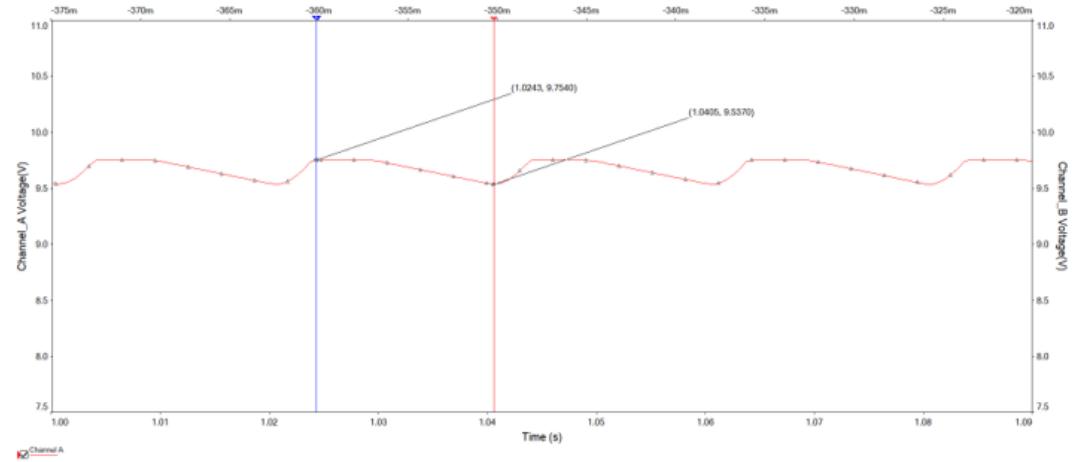


Figura 3.29. Rise time off y factor de rizado con 0 voltios en el PWM.

En la figura 3.30 se muestra el factor de rizado y el tiempo de estabilización de  $V_{OUT}$  cuando en el PWM tenemos un valor de 5 Voltios, aquí la carga es el paralelo de la resistencia R6 de  $22 k\Omega$  y R3 que también es de  $22 k\Omega$ . Puesto que al estar abierto el transistor, cambia la resistencia de carga y toma el valor de  $11 k\Omega$ . El rizado es de 0.4389 Voltios y el tiempo de subida es de 482 milisegundos.

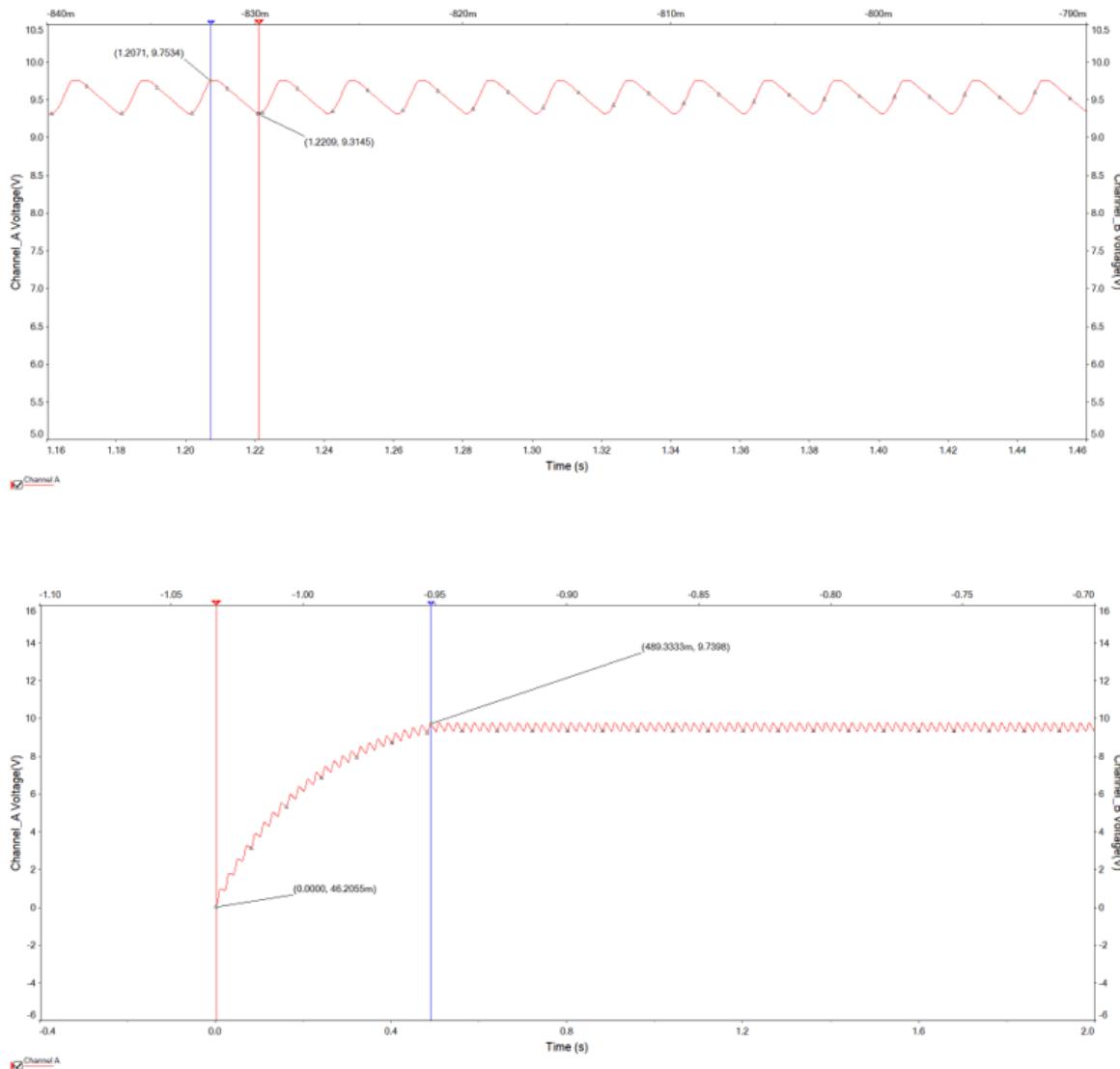


Figura 3.30. Rise time on y factor de rizado con 5 voltios en el PWM.

### Dimensionamiento de la resistencia R1 y el Zener.

En circuitos de potencia la elección de los componentes es una consideración crítica, como regla general los dispositivos deben dimensionarse al doble de la potencia máxima calculada en cada uno de ellos. Los valores RMS de corriente y tensión se utilizarán para los cálculos.

Para dimensionar la resistencia R1 de la figura 37 se calculó la potencia que disipa, asumiendo una resistencia con una tolerancia del 1% y como se mencionó anteriormente se utilizará la corriente  $I_{INmax}$ :

$$P_{R1} = I_{INmax}^2 * R_1 = \frac{V_{RMSmax}^2}{R_1} = \frac{(235 V)^2}{100k\Omega * 0.99} = 0.56 W \quad (3.36)$$

Se utilizará una resistencia de  $100\text{ k}\Omega$  de 1% de tolerancia y que disipa 2 watos, para dimensionar el diodo Zener se tuvo en cuenta que la máxima corriente se producirá cuando no exista una carga, la corriente que atraviesa el diodo Zener será aproximadamente igual a la corriente que atraviesa  $R_1$ , y teniendo en cuenta las tolerancias de los componentes  $R_1$  y del Zener se obtiene:

$$P_{zener} = V_z * I_{INmax} = V_z * \frac{V_{RMSmax}}{R_1} = 10.6\text{ V} * \frac{235\text{ V}}{100\text{ k}\Omega * 0.99} = 0.025\text{ W} \quad (3.37)$$

Se ha colocado un diodo Zener de 1 watio de potencia, en el dimensionamiento del diodo D1 se tuvo en cuenta que la corriente que lo atraviesa es la misma que  $R_1$ , la caída de tensión que se produce en el diodo es de 1.1 Voltios.

$$P_{D1} = V_F * I_{Inmax} = V_F * \frac{V_{RMSmax}}{R_1} = 1.1\text{ V} * \frac{235\text{ V}}{100\text{ k}\Omega * 0.99} = 0.0026\text{ W} \quad (3.38)$$

se puso un diodo de 1/8 de watio, el condensador C1 tiene que soportar la tensión Zener, un condensador electrolítico de 35 voltios es suficiente.

### Dimensionamiento del optoacoplador

En el diseño se ha optado por la configuración no inversora, y el modo de funcionamiento lógico para cuando por la entrada del optoacoplador llegue una tensión de 5 voltios del PWM sature el fototransistor y cortocircuite haciendo que la tensión de salida  $V_{OUT}$  (tensión en el Emisor) sea igual a la tensión de alimentación  $V_{CC}$  (tensión en el colector) menos la tensión  $V_{CESAT}$  (caída de tensión entre colector y emisor en saturación) que es de aproximadamente 0.3 voltios según el *datasheet* del CNY65 como se puede apreciar en la figura 3.31. Por el contrario, cuando le llega una tensión de 0 voltios, el fototransistor entra en corte y la tensión de salida  $V_{OUT}$  toma el valor de cero.

ELECTRICAL CHARACTERISTICS ( $T_{amb} = 25\text{ }^{\circ}\text{C}$ , unless otherwise specified)						
PARAMETER	TEST CONDITION	SYMBOL	MIN.	TYP.	MAX.	UNIT
input						
Forward voltage	$I_F = 50\text{ mA}$	$V_F$	-	1.25	1.6	V
Junction capacitance	$V_R = 0$ , $f = 1\text{ MHz}$	$C_j$	-	50		pF
output						
Collector emitter voltage	$I_C = 1\text{ mA}$	$V_{CEO}$	32	-	-	V
Emitter collector voltage	$I_E = 100\text{ }\mu\text{A}$	$V_{ECO}$	7	-	-	V
Collector emitter leakage current	$V_{CE} = 20\text{ V}$ , $I_F = 0\text{ A}$	$I_{CEO}$	-	-	200	nA
coupler						
Collector emitter saturation voltage	$I_F = 10\text{ mA}$ , $I_C = 1\text{ mA}$	$V_{CESat}$	-	-	0.3	V
Cut-off frequency	$V_{CE} = 5\text{ V}$ , $I_F = 10\text{ mA}$ , $R_L = 100\text{ }\Omega$	$f_c$	-	110	-	kHz
Coupling capacitance	$f = 1\text{ MHz}$	$C_k$	-	0.3	-	pF

Figura 3.31. Características eléctricas del CNY65 (Fuente:[25]).

### Corriente de entrada $I_F$

El fabricante Vishay® recomienda una corriente directa baja para aumentar el tiempo de vida del dispositivo, se ha limitado la corriente directa a 4 mA que circula por el diodo led, las fórmulas para calcular la resistencia limitadora y la potencia que disipa son las siguientes:

$$R_{LED} = \frac{V_{in} - V_F}{I_{LED}} = \frac{5V - 1.1V}{4mA} = 975\Omega \quad (3.39)$$

$$P_{R_{LED}} = R_{LED} * I_{LED}^2 = 975\Omega * 4mA^2 = 0.0156W \quad (3.40)$$

Donde  $V_{in}$  es la tensión de alimentación a la entrada al optoacoplador,  $V_F$  la caída de tensión directa del diodo. En la figura 3.32 se muestra la caída de tensión directa del diodo y la corriente por el colector para una corriente directa de 4 mA y una tensión  $V_{CE}$  de 10 voltios, que son de 1.1 voltios y 4 mA respectivamente. Se ha colocado una resistencia de 1000 ohmios con una tolerancia del 1% y de un cuarto de watio.

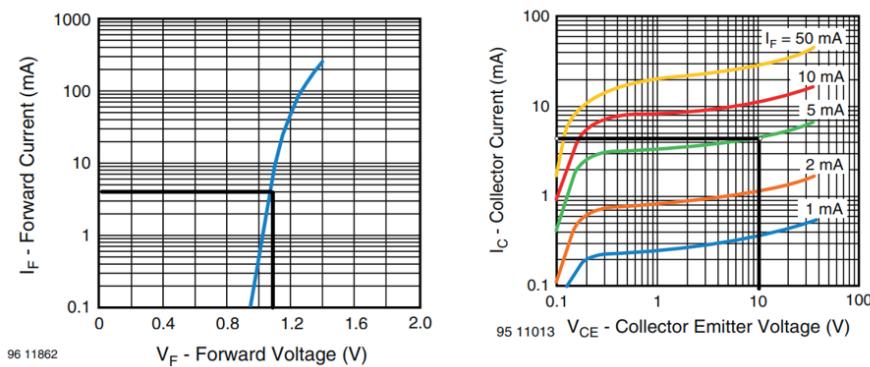


Figura 3.32. Curva  $I_F$  vs  $V_F$  y curva  $I_C$  vs  $V_{CE}$  del CNY65 (Fuente:[25]).

### Dimensionamiento del disipador para el MOSFET

En la figura 3.33 se muestra la temperatura en la unión del transistor  $T_j$  máxima que puede soportar antes de romperse el semiconductor.

Symbol	Parameter	Value	Unit
$V_{GS}$	Gate-source voltage	$\pm 25$	V
$I_D$	Drain current (continuous) at $T_C = 25^\circ C$	11 <sup>(1)</sup>	A
$I_D$	Drain current (continuous) at $T_C = 100^\circ C$	7 <sup>(1)</sup>	A
$I_{DM}$ <sup>(2)</sup>	Drain current (pulsed)	44 <sup>(1)</sup>	A
$P_{TOT}$	Total dissipation at $T_C = 25^\circ C$	25	W
$dv/dt$ <sup>(3)</sup>	Peak diode recovery voltage slope	15	V/ns
$dv/dt$ <sup>(4)</sup>	MOSFET $dv/dt$ ruggedness	50	V/ns
$V_{ISO}$	Insulation withstand voltage (RMS) from all three leads to external heat sink ( $t = 1 s$ ; $T_C = 25^\circ C$ )	2500	V
$T_{stg}$	Storage temperature	- 55 to 150	$^\circ C$
$T_j$	Max. operating junction temperature		

Figura 3.33.Temperatura máxima en la unión del STF13N60M2 (Fuente:[42]).

Para evitar que llegue a esa temperatura se va a emplear un coeficiente de seguridad, y con dicha temperatura de unión se realizarán los cálculos:

$$T_{jmax} = k * T_{jmax}(\text{datasheet}) \quad (3.41)$$

Se escoge un valor de 0.5 para  $k$  por tanto la temperatura de unión para los cálculos del disipador es:

$$T_{jmax} = 0.5 * 150^{\circ}\text{C} = 75^{\circ}\text{C} \quad (3.42)$$

Se calcula la potencia que va a disipar el MOSFET para ello se calcula la corriente que pasa por el dispositivo utilizando el esquema de la figura 3.34.

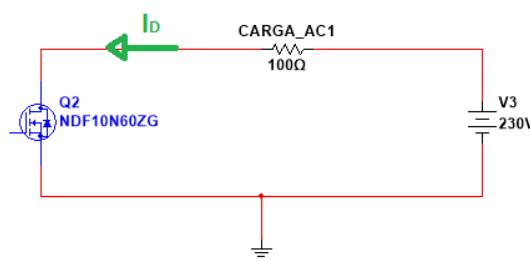


Figura 3.34. Circuito equivalente para calcular la  $I_D$  que atraviesa el MOSFET.

Cuando el MOSFET conduce se comporta como un circuito cerrado, por tanto, la corriente que lo atraviesa es:

$$I_D = \frac{V_{RMS}}{R_{Horno}} = \frac{230V}{100\Omega} = 2.3 A \quad (3.43)$$

Con esta corriente se va al *datasheet* del dispositivo y se busca el valor de la tensión  $V_{DS}$  como se puede apreciar en la figura 3.25 es de 1 Voltio.

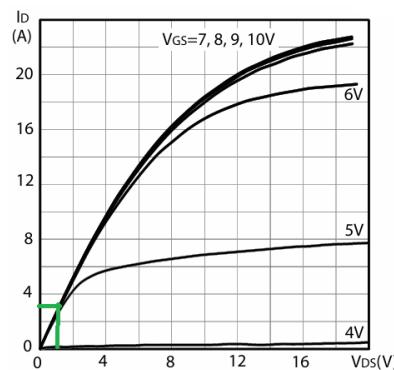


Figura 3.35. Curva  $I_D$  vs  $V_{DS}$  del STF13N60M2 (Fuente:[42]).

La potencia disipada por conducción del MOSFET es:

$$P_{MOSFET} = V_{DS} * I_D = 1V * 2.3A = 2.3W \quad (3.44)$$

En la figura 3.36 el valor de la resistencia térmica entre la unión y el ambiente es de 62.5 °C/W realizando el equivalente térmico como se aprecia en la figura 3.37, se procede a calcular la temperatura de la unión.

Symbol	Parameter	Value	Unit
R <sub>thj-case</sub>	Thermal resistance junction-case max	5	°C/W
R <sub>thj-amb</sub>	Thermal resistance junction-ambient max	62.5	°C/W

Figura 3.36. Resistencia térmica unión-ambiente del STF13N60M2 (Fuente:[42]).

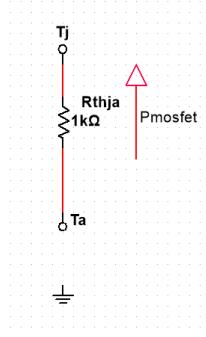


Figura 3.37. Equivalente térmico unión-ambiente del STF13N60M2

La temperatura en la unión, asumiendo una temperatura ambiente de 30 grados que puede llegar a tener el laboratorio en verano sin aire acondicionado, se calcula de la siguiente forma:

$$T_j = P_{MOSFET} * R_{Thj-amb} + T_{amb} = 2.3W * 62.5 \frac{^{\circ}C}{W} + 30^{\circ}C = 173^{\circ}C \quad (3.45)$$

Es decir, sin disipador el MOSFET sobrepasaría la temperatura máxima en la unión sin aplicarle el factor de seguridad, por tanto, se debe colocar un disipador.

En la figura 3.38 se muestra el circuito térmico cuando se añade un disipador y una lámina de mica entre el transistor y el disipador. El término  $R_{Thcd}$  depende de la lámina de mica, la pasta térmica que se aplique y el par de apriete de la fijación, como no se tiene información del par de apriete y no se ha colocado una lámina de mica los términos que restan para calcular  $R_{da}$  no se han tenido en cuenta, por tanto, la resistencia térmica entre el disipador y el ambiente toma el valor siguiente:

$$R_{da} = \frac{T_{jmax} - T_{amb}}{P_{MOSFET}} = \frac{75^{\circ}C - 30^{\circ}C}{2.3 W} = 19.56 \frac{^{\circ}C}{W} \quad (3.46)$$

Si colocamos un disipador de forma vertical hay que dividir este valor por el coeficiente 1.1 como nos lo recomienda el fabricante DISIPA® entonces el valor  $R_{da}$  es:

$$R_{da} = \frac{19.56 \frac{\text{°C}}{\text{W}}}{1.1} = 17.78 \frac{\text{°C}}{\text{W}} \quad (3.47)$$

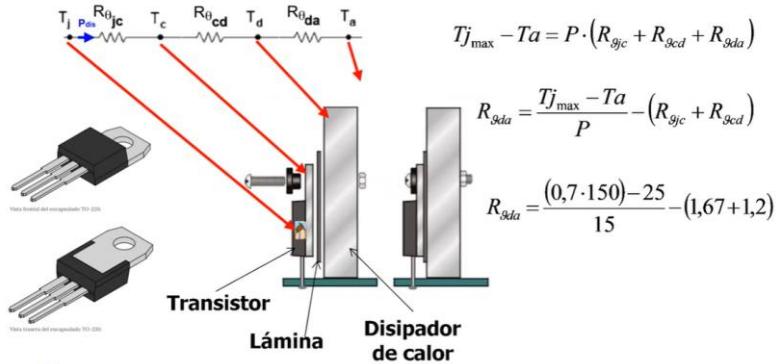


Figura 3.38. Circuito térmico al incluir un disipador y una lámina de mica (Fuente:[44]).

Se procede a elegir un disipador que tenga un menor valor que  $17.78 \text{ °C/W}$  de resistencia térmica, como es un valor muy alto y se reutilizó un disipador de otro dispositivo de forma cuadrada y 5 centímetros de lado, se buscó en el catálogo de DISIPA® un disipador parecido para ver el valor de la resistencia térmica, como se puede apreciar en la figura 3.39 el valor es de  $2.4 \text{ °C/W}$  que nos sirve, además para asegurarnos que el MOSFET no se caliente se incluyó un ventilador que también se reutilizó de otro dispositivo que iba a ser reciclado.

Un factor para tener en cuenta es que no se calcularon las pérdidas por conmutación del MOSFET, solo las de conducción, ya que la frecuencia de conmutación es baja y las pérdidas por conmutación no predominan en aplicaciones a frecuencia de la red eléctrica sino las de conducción.

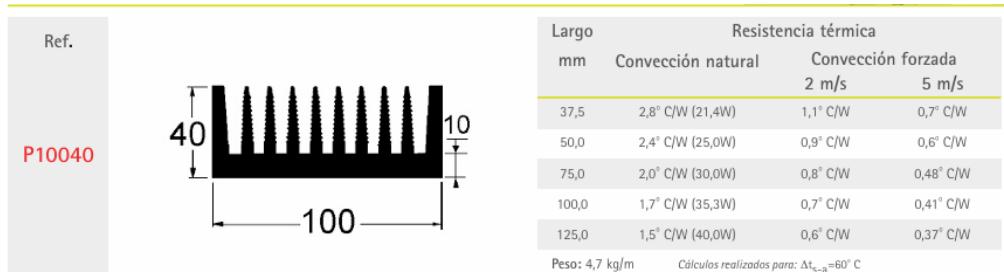


Figura 3.39. Disipador de referencia utilizado (Fuente:[45]).

### 3.2.4 Simulaciones de las placas

En este apartado se van a comentar los circuitos que se simularon en Multisim para verificar los cálculos teóricos obtenidos.

#### Simulación del acondicionador de señal

En la figura 3.40 se puede observar la simulación para una variación de temperatura de  $-5^\circ \text{C}$  (extremo inferior de la recta de transferencia del acondicionador, es decir, un valor de 100 ohmios). El valor que toma  $V_s$  es de 0.000452 voltios.

En la figura 3.41 se observa la simulación para una variación de temperatura de 277 °C (extremo superior de la recta de transferencia del acondicionador, es decir, un valor de 205.95 ohmios). El valor que toma  $V_S$  es de -0.0234.

En ambas figuras se puede apreciar que la corriente que atraviesa la PT100 es de 0.452 miliamperios está por encima del calculado teóricamente. Esta diferencia se debe porque los cálculos se realizaron para una resistencia en la PT100 de 102 ohmios.

La tensión de referencia en el puente es 0.0459 voltios y coincide con el valor teórico calculado, no obstante, la tensión de referencia para el INA128 toma el valor 0.0441 voltios en la simulación y se obtuvo un valor teórico de 0.04532 voltios, esto puede ser debido a algunos parámetros internos del simulador que no se hayan tenido en cuenta, de todas formas, se añadieron potenciómetros para calibrar dichas tensiones en el diseño del PCB.

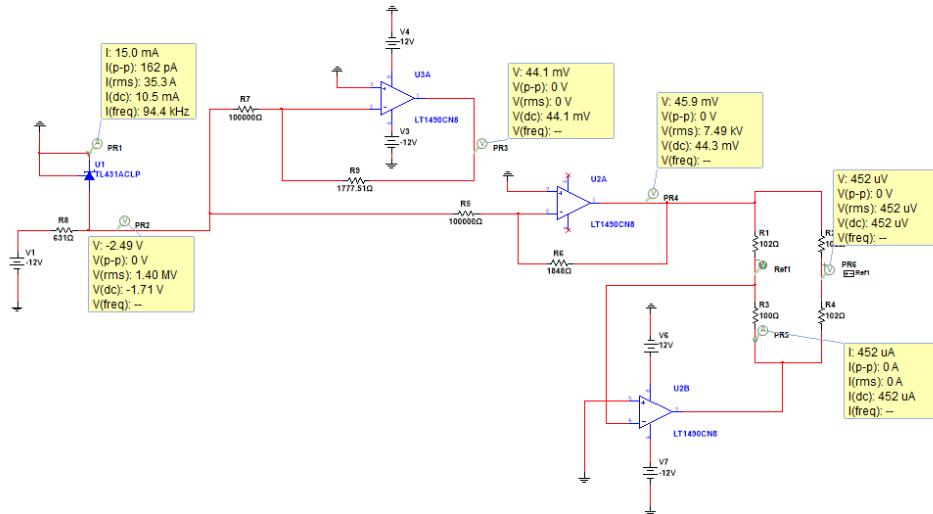


Figura 3.40. Salida del puente para una variación de -5°C.

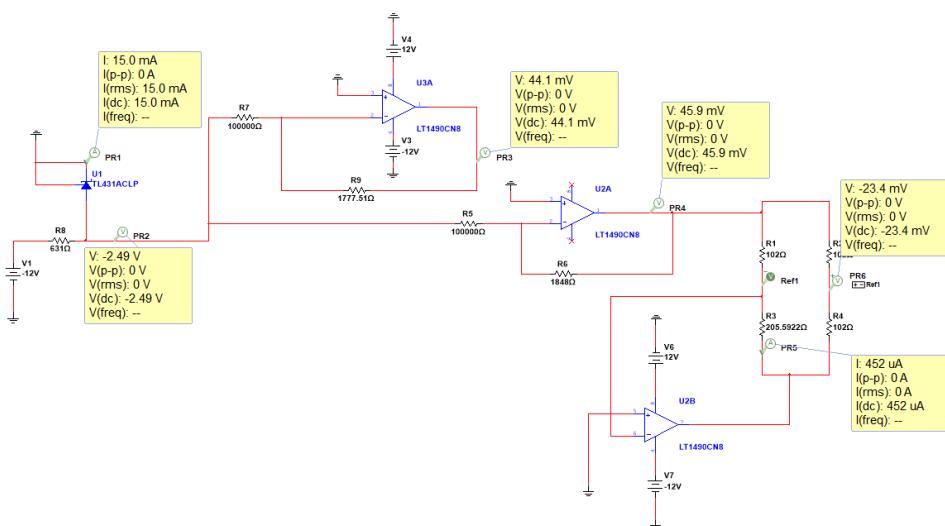


Figura 3.41. Salida del puente para una variación de 277°C.

### Simulación de la placa de potencia

En la figura 3.42 se observa la simulación del circuito de potencia cuando tenemos a la entrada del PWM una tensión de 0 voltios. Para este caso la tensión en la puerta del MOSFET es de 2.52 milivoltios y se encuentra funcionando como un circuito cerrado, por tanto, por la carga (horno) no pasará corriente, es decir, no habrá tensión como se puede mirar en el recuadro derecho (tensión diferencial en la carga).

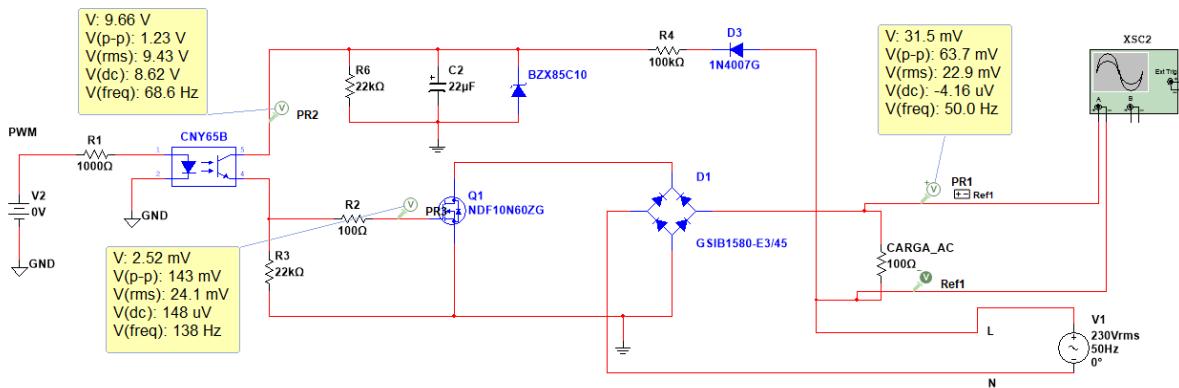


Figura 3.42. Simulación del circuito de potencia con 0V en PWM.

En la figura 3.43 se observa la simulación del circuito de potencia cuando tenemos a la entrada del PWM una tensión de 5 voltios. Para este caso la tensión en la puerta del MOSFET es de 9.34 voltios y se encuentra funcionando como un circuito abierto, por tanto, por la carga (horno) pasará corriente, es decir, habrá tensión como se puede mirar en el recuadro derecho (tensión diferencial en la carga).

También se puede observar en ambas figuras como la fuente resistiva da una tensión de 9.66V y de 9.45V, estos cambios de tensión se deben a que la carga (de la fuente resistiva) cambiará cuando se active el optoacoplador, en este caso la carga será el paralelo de R3 y R6, y cuando este desactivado será solo R6.

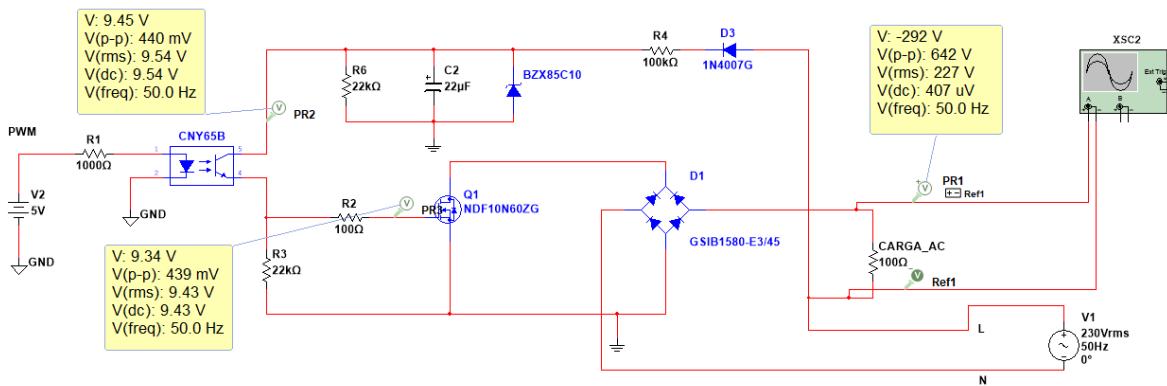


Figura 3.43. Simulación del circuito de potencia con 5V en PWM.

### 3.2.5 Diseño de los PCB

#### Esquema electrónico del acondicionador de señal

El esquema electrónico para realizar el PCB difiere del de simulación, por ejemplo, las entradas y salidas de la señal se sustituirán por conectores, así como, la alimentación del circuito. En la parte de alimentación se añadirán condensadores de desacoplo para altas y bajas frecuencias, también se añadirán condensadores de desacoplo en la alimentación de los circuitos integrados, además se incluyeron puntos de prueba para comprobar tensiones en nodos críticos, para implementar el esquema electrónico se ha utilizado el programa Multisim la versión 14.3 de la compañía National Instruments®. En la figura 3.44 se muestra el esquema del circuito electrónico para el acondicionador de señal (en los anexos se visualiza mejor el esquema), como se puede apreciar los componentes tienen color azul o verde esto significa que dentro del componente este tiene una huella asignada, es decir, la forma física del componente que se utilizará para el diseño del PCB.

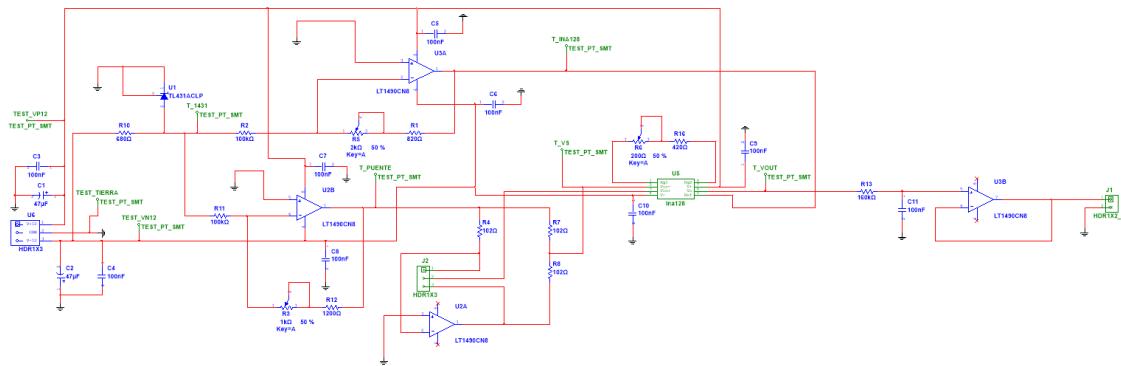


Figura 3.44. Esquema electrónico del acondicionador de señal.

Los puntos de prueba que se han colocado han sido para medir la tensión que nos da el regulador TI1431, la tensión de salida de los amplificadores inversores que nos dará la tensión de referencia en el puente y del INA128, otros puntos de prueba que se colocaron fueron para medir la tensión Vs del puente de medida y la tensión Vout que nos da el INA128. Estos puntos de prueba se han incluido para realizar mediciones de una forma segura ante descargas electrostáticas y comprobar que se dan las tensiones teóricas calculadas, y si no es el caso, ajustar los potenciómetros de precisión hasta obtener los resultados esperados.

Uno de los aspectos importantes a la hora de realizar el esquema electrónico para el acondicionador de señal han sido la selección de las huellas, consultando las hojas de características de cada dispositivo electrónico y comparando las medidas con las huellas ofrecidas en las librerías de Multisim. En el caso de que no existiese una huella se creó de forma manual siguiendo la guía ofrecida en el manual de diseño de circuitos impresos [46], este ha sido el caso del condensador para el filtro donde se tuvo que realizar su huella, en la figura 3.45 se muestra las huellas de estos dos componentes realizados de forma manual. El resto de los componentes se pudieron encontrar en las librerías.

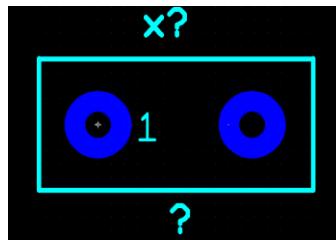


Figura 3.45. Footprint creado de forma manual para el condensador del filtro antialiasing.

Otro aspecto importante que considerar a la hora de realizar el esquema es que los componentes estén bien conectados al nodo que corresponde, por ejemplo, en la figura 3.46 se muestra un ejemplo de una conexión mal realizada, esto puede pasar desapercibido y ser un error grave, puesto que, algunos programas de diseño de circuitos electrónicos pasan desapercibido esta mala conexión, no obstante, existe una herramienta en Multisim que permite comprobar que ninguno de los componentes se haya quedado sin conectar, esta herramienta se denomina *electrical rules check*.

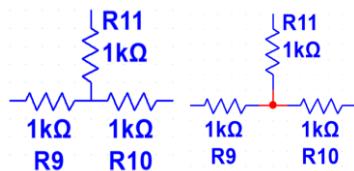


Figura 3.46. Ejemplo de mala conexión de componentes en Multisim (Fuente:[46]).

Para finalizar con el esquema del acondicionador de señal se tuvo también que crear un componente para el INA128, ya que, no se encontraba en las librerías de Multisim, como se mencionó anteriormente, siguiendo el manual [46] se creó un componente y se asignó las conexiones eléctricas de la misma forma que el dispositivo real, en la figura 3.47 se muestra el resultado.

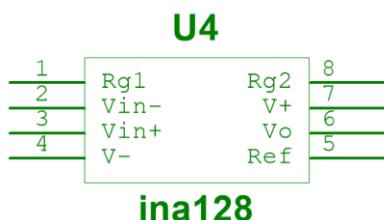


Figura 3.47. Componente creado de forma manual para el INA128.

### Diseño del circuito impreso para el acondicionador de señal

#### Colocación de los componentes

En la figura 3.49 se ve cómo se han colocado las huellas de los componentes, destacando la zona remarcada en color rojo en donde se sitúa el conector para el sensor de temperatura PT100, junto con el INA128 (U5) y las resistencias del puente, se han colocado todos estos componentes lo más próximos posibles para que la señal tenga un recorrido lo más corto posible, otro factor a tener en cuenta que nos recomienda el fabricante Texas Instruments® en sus notas de aplicación [47], es que la resistencia que determina la ganancia tiene que estar lo más cerca posible del CI (Circuito Integrado) como se muestra en la figura 3.48.

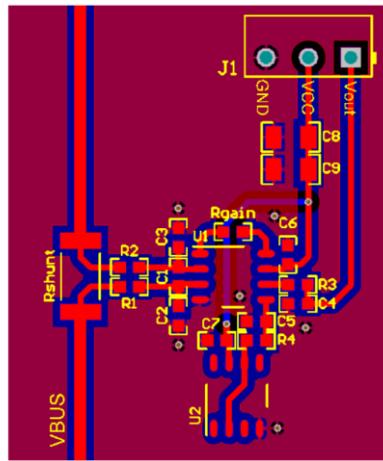


Figura 3.48. Recomendación del fabricante Texas Instruments® para las pistas en el INA128 (Fuente:[47]).

Por lo demás se han seguido las reglas de diseño aconsejadas en el manual [46], por ejemplo, los conectores tienen que estar en los bordes de la placa, los condensadores de desacoplo de los CI tienen que estar lo más cerca posible de los pinos de alimentación, la serigrafía debe tener un máximo de dos direcciones diferentes, los orificios de los taladros para la sujeción de la placa tienen que estar en las esquinas, los condensadores de desacoplo de la fuente de alimentación tienen que estar lo más cerca posible y por último, los componentes tienen que estar lo más ordenados posibles abarcando la menor superficie en el PCB.

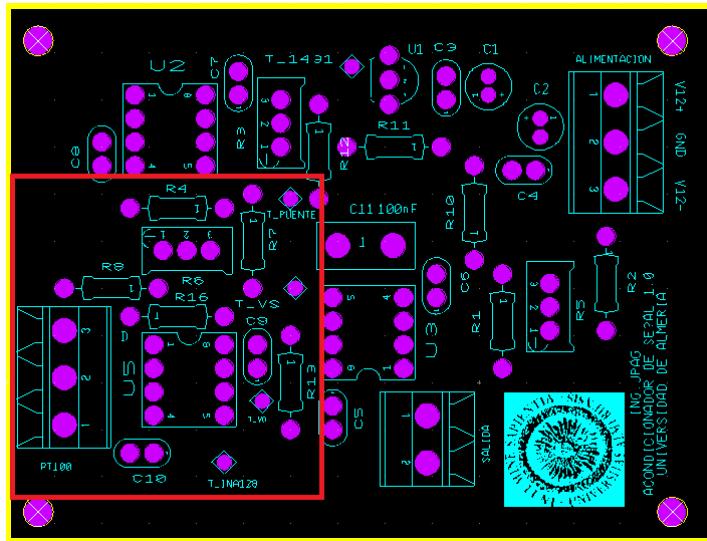


Figura 3.49. Disposición de los componentes en la PCB del acondicionador de señal.

### Capas top y bottom

En la figura 3.50 se observa la capa top, en esta se ha colocado las pistas de alimentación positiva y negativa, aunque sean largas, el posible ruido que se puede acoplar se suprimirá con los condensadores de desacoplo que se colocarán justo antes de los pinos de alimentación de los CI, el resto del espacio se ha cubierto por el plano de tierra, es decir, el nodo cero que se toma como referencia en los programas de diseño de circuitos de electrónicos.

En la figura 3.51 se muestra la capa bottom, en esta se han hecho las conexiones de señal intentando que las pistas sean lo más cortas posibles, el resto del espacio también se ha colocado como un plano de tierra.

Como características generales se han utilizado pads redondos con diámetros de 102 mils (2.6 mm) para la alimentación, 63 mils (1.6 mm) y 78 mils (2 mm) para los demás componentes, los diámetros de los taladros que se han utilizado son de 39 mils (1 mm) para la alimentación y 31 mils (0.8 mm) para los demás componentes, el espacio entre las pistas (*clearance*) son de 20 mils (0.5 mm) y 9 mils (0.22 mm). Para finalizar los taladros para la sujeción de la placa ha sido de 3 mm de diámetro.

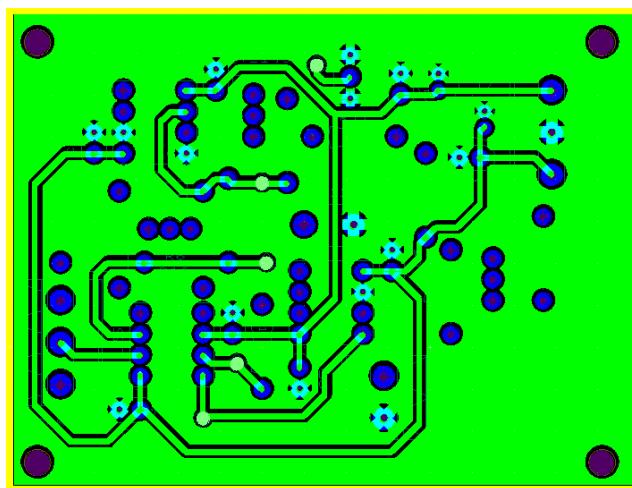


Figura 3.50. Capa top del acondicionador de señal.

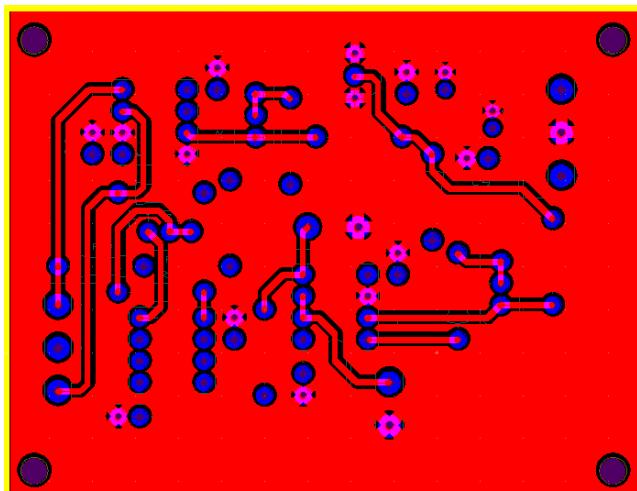


Figura 3.51. Capa bottom del acondicionador de señal.

### Capa de serigrafía y visualización 3D

En la figura 3.52 se visualiza la capa de serigrafía, se ha incluido el símbolo de la universidad de almeria, así como, el nombre de la placa y las iniciales del autor, también se han incluido algunas ayudas textuales en los conectores para la conexión de los cables de alimentación procedentes de una fuente de alimentación externa.

A continuación, en la figura 3.53 se muestra la visualización de la PCB en tres dimensiones, esto nos permite tener una idea del resultado final de la placa cuando se realice el prototipado. Para ello se enviarán los archivos gerber de la placa a una empresa especializada en la fabricación de PCBs.

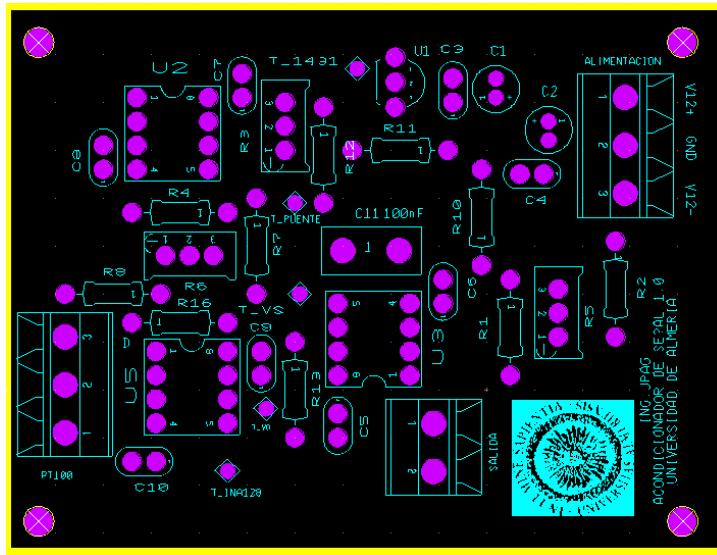


Figura 3.52. Capa de serigrafía del acondicionador de señal.

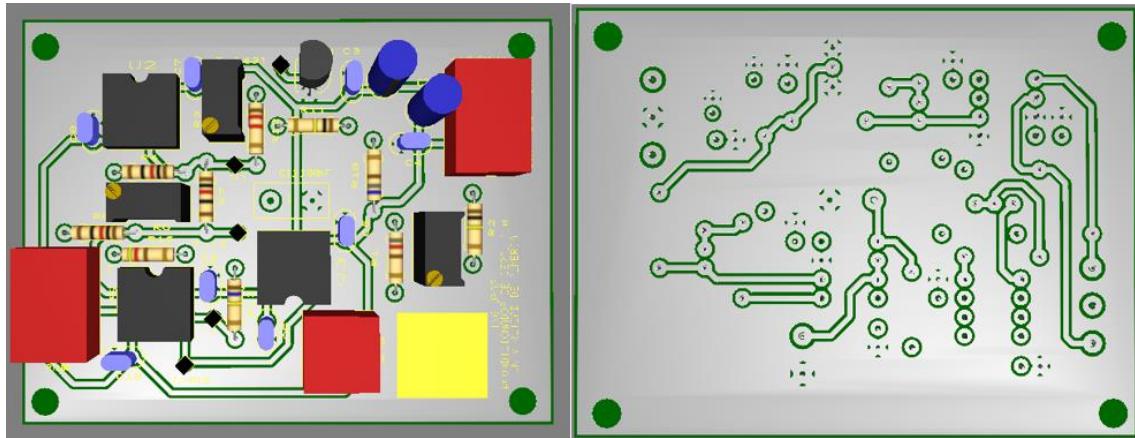


Figura 3.53. Capa de visualización 3D del acondicionador de señal.

### Esquema electrónico de la placa de potencia

El segundo esquema electrónico se realizó para la placa de potencia, que controlará la resistencia del horno, al igual que en el anterior diseño se sustituyeron las entradas y salidas por conectores y se añadieron puntos de prueba. En la figura 3.55 se muestra el esquema final.

Se colocaron dos puntos de prueba, uno para comprobar la tensión que proporciona la fuente y alimenta el colector del fototransistor del optoacoplador, el otro punto se colocó para medir la tensión de activación de la compuerta del MOSFET, al igual que en el otro esquema, se comprobó cuidadosamente que las huellas de todos los componentes tengan las medidas correctas, concordancia en las conexiones eléctricas y que existan en las librerías de Multisim, el componente que se tuvo que crear su huella fue del optoacoplador, en la figura 3.54 se muestra el resultado.

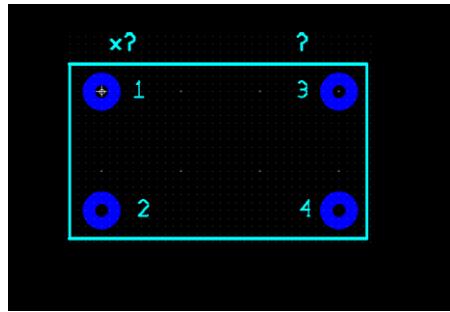


Figura 3.54. Footprint creado de forma manual para el optoacoplador.

Una vez comprobado que no existen falsas conexiones entre los componentes utilizando la herramienta electrical rules check (ERC), se exportaron los netlist de los esquemas electrónicos para realizar el diseño de las PCBs con el programa Ultiboard de la misma compañía National Instruments® y que está especializado en el diseño de circuitos impresos.

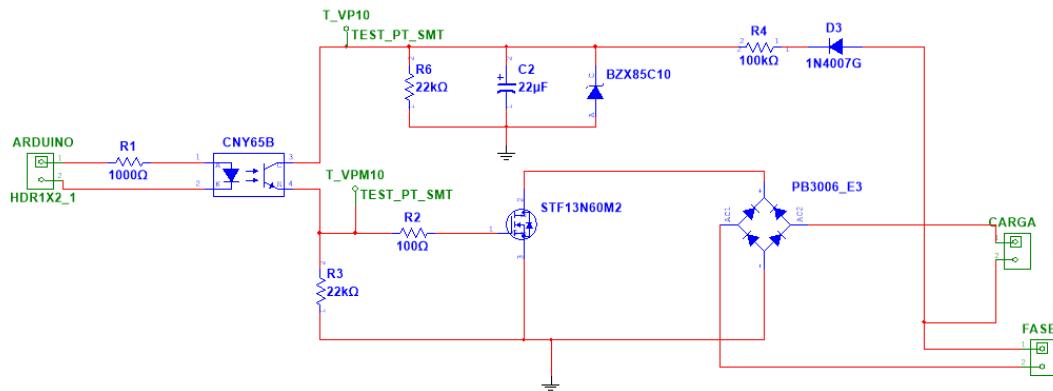


Figura 3.55. Esquema eléctrico de la placa de potencia.

### Diseño del circuito impreso de la placa de potencia

#### Colocación de los componentes

En la figura 3.56 se observa cómo se han colocado las huellas de los componentes, destacando la zona remarcada en color rojo en donde se sitúa el puente de diodos que se ha colocado al borde de la placa para colocar un disipador.

Los conectores se han colocado en los bordes para facilitar el conexionado con la red eléctrica y la resistencia del horno, así como, el conector que se conectará al Arduino y enviará la señal PWM de control para conmutar el transistor MOSFET, también se ha tenido en cuenta la colocación de los componentes que manejan elevadas corrientes, de modo que, estén lo más cercanos posibles entre ellos para que las pistas sean lo más cortas posibles.

La resistencia R4 se ha colocado alejada para que la potencia que va a disipar no caliente los componentes sensibles a su alrededor, y que se produzca un enfriamiento por convección natural lo más eficiente posible.

El optoacoplador se ha colocado lo más cerca posible de la clema que envía la señal PWM y evitar posibles ruidos que se puedan acoplar a la señal, por lo demás se ha seguido las reglas básicas de ocupar el menos espacio posible y una colocación ordenada de los componentes.

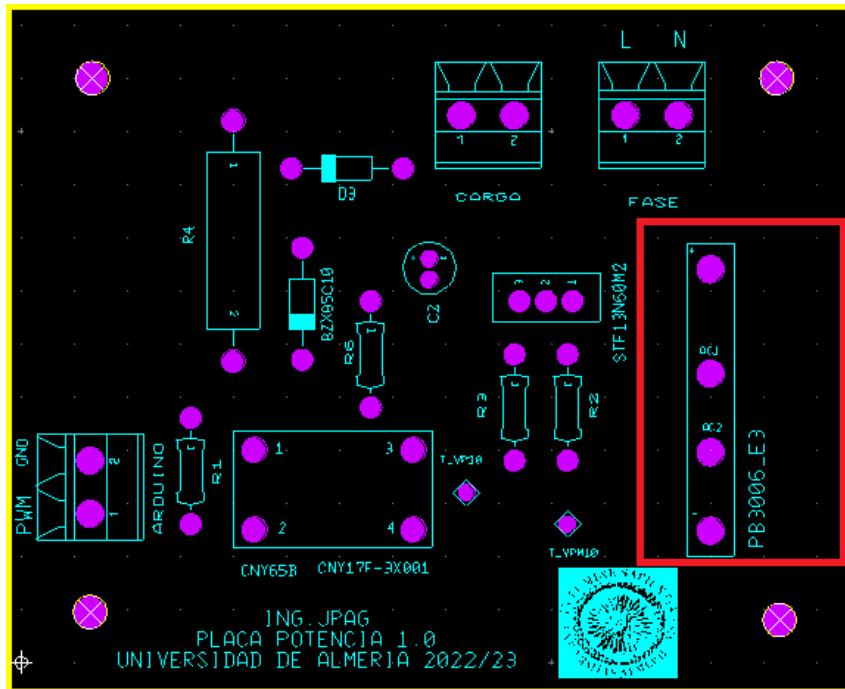


Figura 3.56. Disposición de los componentes en la PCB de la placa de potencia.

#### Capas top y bottom

En la figura 3.55 se muestra la capa top, en esta se ha colocado el plano de tierra y la pista que va de la salida en alterna AC2 del puente rectificador a la carga, también se encuentran los pads de prueba para comprobar la tensión suministrada por la fuente resistiva que se diseñó y la tensión en la puerta del MOSFET.

En la figura 3.56 se muestra la capa bottom, se han hecho las conexiones de señal intentando que las pistas sean lo más cortas posibles, el resto del espacio también se ha colocado como un plano de tierra.

Como características generales se han utilizado pads redondos con diámetros de 102 mils (2.6 mm) para la alimentación, 63 mils (1.6 mm) y 78 mils (2 mm) para los demás componentes, los diámetros de los taladros que se han utilizado son de 39 mils (1 mm) para los conectores, 51 mils (1.3 mm) para el puente rectificador y 31 mils (0.8 mm) para los demás componentes, el espacio entre las pistas (*clearance*) son de 20 mils (0.5 mm) y 9 mils (0.22 mm). Para finalizar los taladros para la sujeción de la placa ha sido de 3 mm de diámetro.

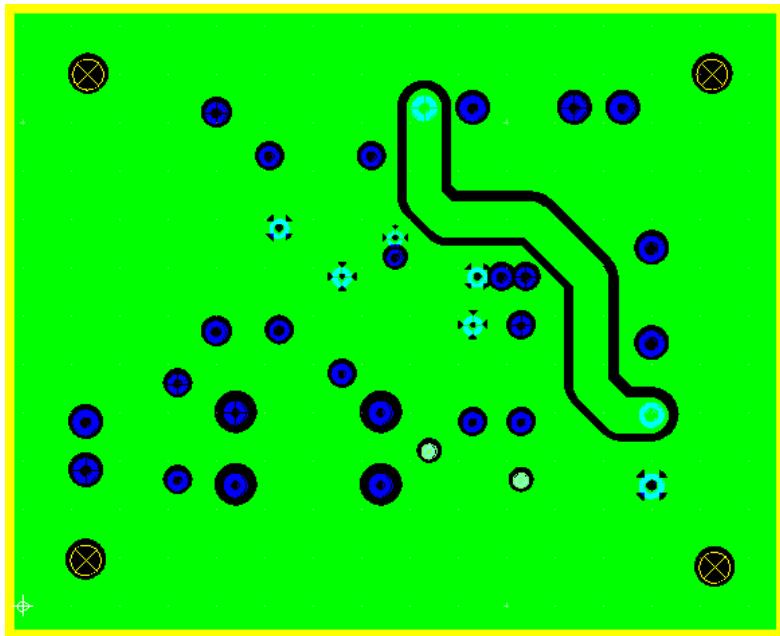


Figura 3.57. Capa top del acondicionador de la placa de potencia.

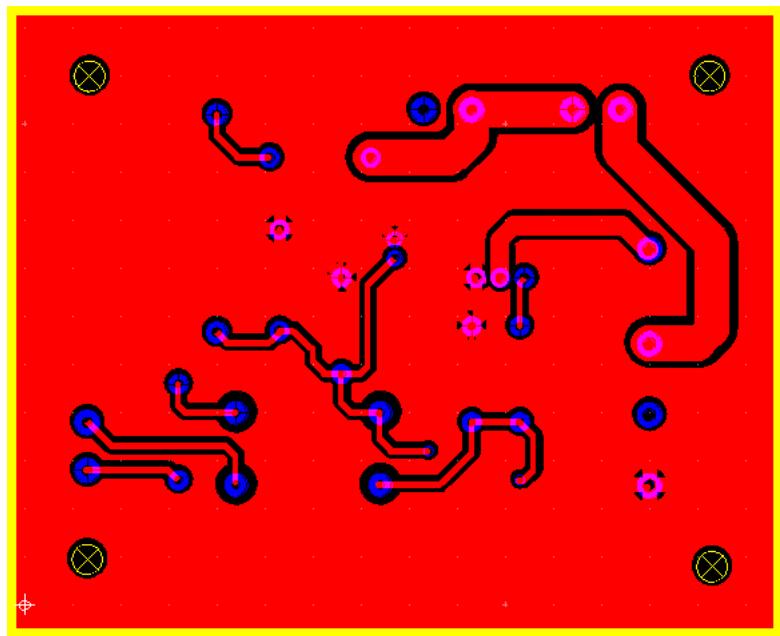


Figura 3.58. Capa bottom de la placa de potencia.

### Capa de serigrafía y visualización 3D

En la figura 3.59 se muestra la capa de serigrafía, se ha incluido el símbolo de la universidad de Almeria, así como, el nombre de la placa y las iniciales del autor, además se han incluido algunas ayudas textuales en los conectores para la conexión de los cables de alimentación procedentes de red eléctrica, para la carga y también para la conexión con el Arduino ya que si se conectase erróneamente podría producirse daños a los componentes.

Por otra parte, en la figura 3.60 se muestra la visualización de la PCB en tres dimensiones, esto nos permite tener una idea del resultado final de la placa cuando se realice el prototipado. Para ello se enviarán los archivos *gerber* de la placa a una empresa especializada en la fabricación de PCBs.

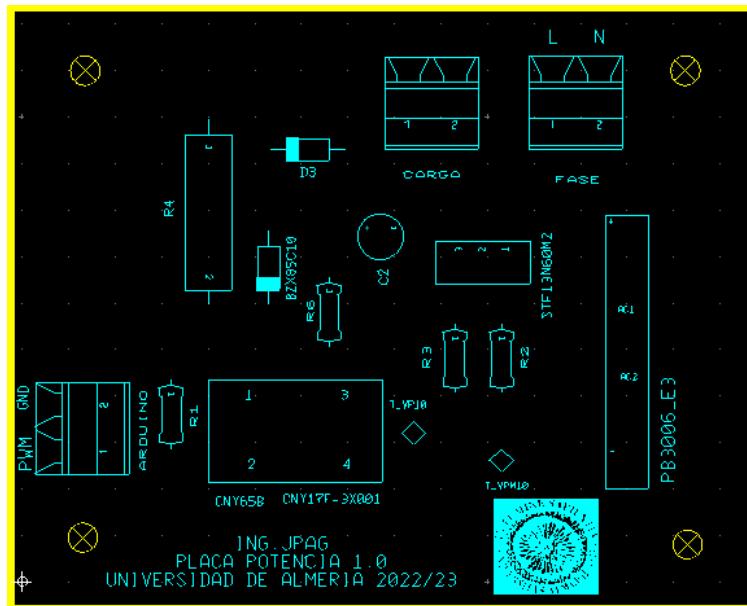


Figura 3.59. Capa de serigrafía de la placa de potencia.

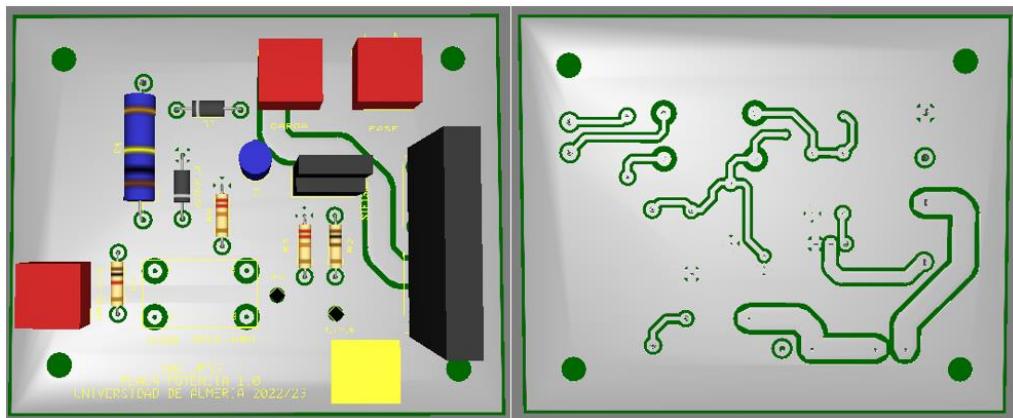


Figura 3.60. Capa de visualización 3D de la placa de potencia.

### Placa del acondicionador de señal

En la figura 3.61 se indica la placa del acondicionador de señal con sus componentes soldados y la función de transferencia calibrada mediante una speudo-PT100 en donde se colocó los valores máximos y mínimos de la recta de transferencia, es decir, para 0V se colocó  $101.9526 \Omega$  ( $5^\circ\text{C}$ ) en la clema donde va la PT100 y se ajustaron las resistencias multivuelta hasta conseguir 0 voltios en el multímetro, a continuación se hizo lo mismo con el valor del otro extremo  $205.5922 \Omega$  ( $282^\circ\text{C}$ ) hasta obtener 2.56 voltios en el multímetro.



Figura 3.61. Placa del acondicionador de señal.

#### Placa de la electrónica de potencia

En la figura 3.62 se observa la placa con sus componentes soldados y añadidos los disipadores de calor para el MOSFET y el puente de diodos.



Figura 3.62. Placa de la electrónica de potencia.

### 3.3 Montaje y conexiones de los subsistemas

En este apartado se va a explicar las conexiones eléctricas realizadas de las placas y los periféricos con en el Arduino.

#### Conexiones eléctricas a la red eléctrica

En la figura 3.63 se visualiza las conexiones eléctricas del horno a la placa de potencia, primero el cable de alimentación de la red eléctrica se conecta a una regleta que tiene un interruptor para dejar pasar la corriente o no en caso de emergencia y poder desconectarlo, a continuación, se conecta a la toma de alimentación que pasa por un fusible en caso de sobrecorriente se fundiría el fusible y cortaría la tensión de alimentación a la placa de potencia, luego la fase (cable marrón) y el neutro (cable azul) se conectan a la clema de entrada de la placa de potencia, la clema de salida se conecta al horno respetando la fase y el neutro marcado en la serigrafía de la placa de potencia.

Además, se conectó el cable de tierra a la caja metálica reciclada de una pistola de calor, en caso de que hubiese una derivación en la parte metálica y pueda saltar el diferencial.

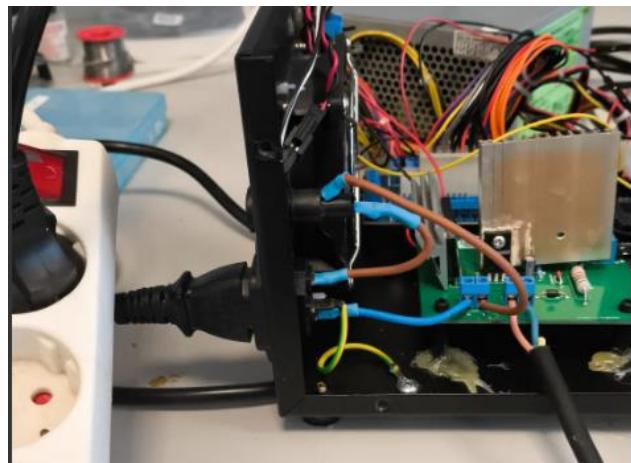


Figura 3.63. Conexión de la red eléctrica a la placa de potencia y de esta al horno.

#### Conexiones eléctricas entre la fuente de alimentación, acondicionador de señal, PT100 y Arduino.

En la figura 3.64, se muestra las conexiones de la PT100 hacia la clema de la PT100 en el acondicionador de señal, también se muestra las conexiones de la alimentación de la placa del acondicionador, el cable rojo es la polaridad positiva de 12 voltios, el cable amarillo es la polaridad negativa de 12 voltios, y el cable negro es la conexión de tierra común entre la placa de alimentación y la del acondicionador de señal. La salida del acondicionador de señal (cable rojo) va a la entrada analógica A4 del Arduino y el cable negro es la conexión de tierra común entre la placa Arduino y la placa del acondicionador de señal.

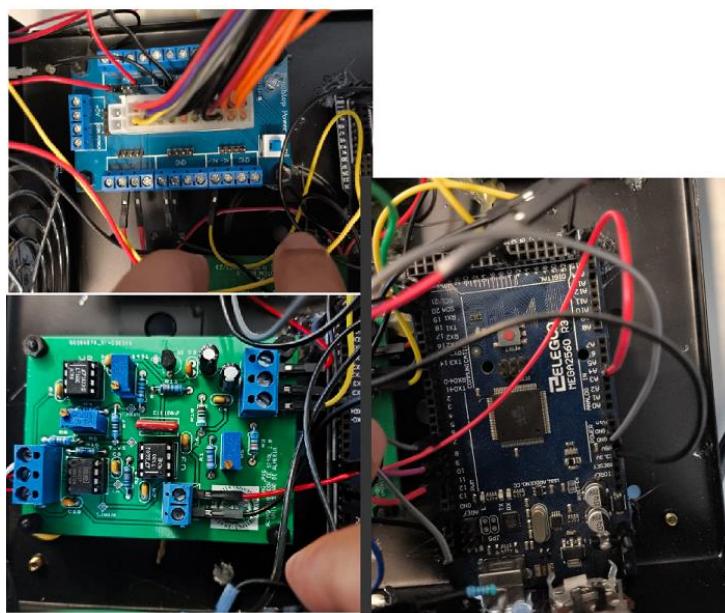


Figura 3.64. Conexión del acondicionador de señal al Arduino y a la fuente de alimentación.

### Conexiones eléctricas del Arduino a los led

En la figura 3.65 se aprecia las conexiones eléctricas de los leds indicadores, el led rojo indica que la temperatura es superior a 30 grados centígrados, las conexiones eléctricas con el Arduino son dos, el cable negro para la conexión a tierra y el cable rojo para la alimentación (pin digital 10 del Arduino). El led verde se enciende cuando la temperatura es inferior a 30 grados centígrados, las conexiones con el Arduino son dos, cable gris conexión a tierra del Arduino y cable violeta para alimentación al LED (pin digital 8 del Arduino).

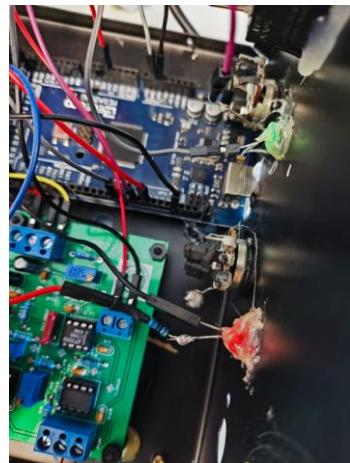


Figura 3.65. Conexión de los leds de aviso de temperatura del horno al Arduino.

### Conexiones eléctricas del Arduino a la pantalla OLED

Para visualizar información resumida del horno sin necesidad de visualizar la interfaz gráfica se incluyó una pantalla OLED, esta pantalla mostrará la temperatura, la señal de control, y si se encuentra encendido el horno o no. En la figura 3.66 se aprecia las conexiones eléctricas de la pantalla con el Arduino y la fuente de alimentación, el cable rojo y negro se conectan a la placa de la fuente de alimentación, el rojo se conecta a 5 voltios de la fuente de alimentación y el cable negro a la tierra común en la fuente, por otra parte, el cable azul se conecta al pin SDA y el cable verde al pin SCL del Arduino.



Figura 3.66. Conexiones eléctricas entre el Arduino y la pantalla OLED.

### Conexiones eléctricas del ventilador al Arduino y a la alimentación

En la figura 3.67 se muestra las conexiones del módulo que controla al ventilador los cables superiores son los que alimentan al ventilador (carga), estos se conectan a la placa de alimentación, el cable rojo a la tensión de 12 voltios y el cable negro a la tierra de la fuente de alimentación. Los cables inferiores son para alimentar al circuito del MOSFET, el cable rojo va a la fuente de alimentación a la tensión de 5 voltios y el cable negro a tierra de la fuente. El cable amarillo va al pin 40 del Arduino.

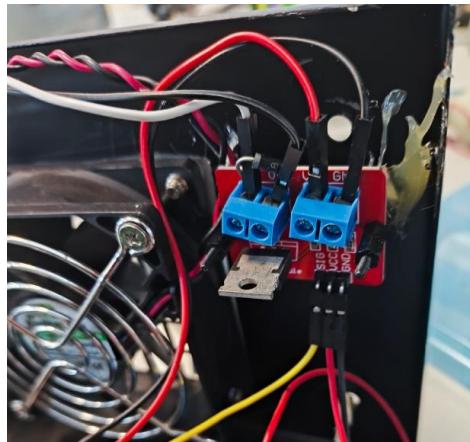


Figura 3.67. Conexiones eléctricas entre el ventilador, el Arduino y la alimentación.

### Montaje en la caja reciclada

Todos los subsistemas se introdujeron en una caja reciclada, como se aprecia en la figura 3.69, proveniente de una pistola de calor que ya no funcionaba. Se colocaron las placas del acondicionador, la de potencia, el Arduino, el módulo que controla el ventilador, y la placa de alimentación. Además, se hicieron algunos agujeros para colocar el ventilador, la pantalla OLED y la salida USB del Arduino como se aprecia en la figura 3.68.



Figura 3.68. Vista frontal y posterior de la caja.

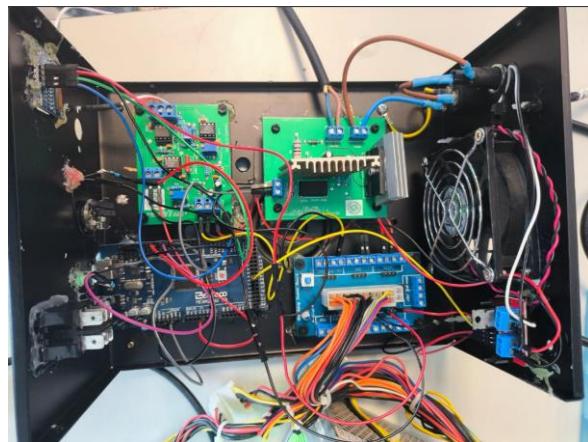


Figura 3.69. Vista en planta de la caja con los sistemas conectados.

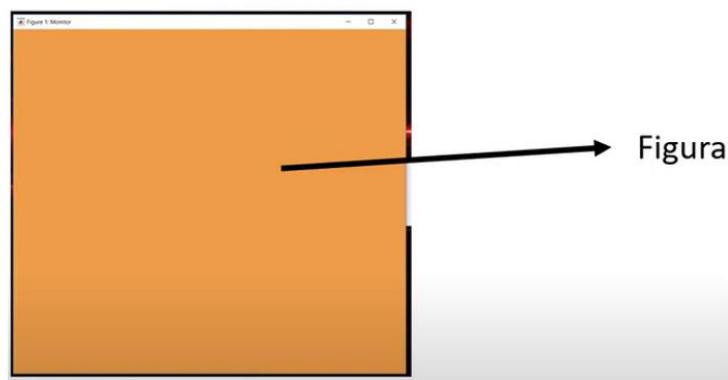
### 3.4 Programación de la interfaz gráfica de Matlab

En este apartado se va a describir las partes del script que se realizó para la interfaz gráfica, las funciones que se crearon para establecer la comunicación entre el Matlab y Arduino, para guardar los datos, y las funciones asociadas a los botones y poder visualizar los datos enviados por el Arduino a Matlab.

#### Creación de la figura

Primero tenemos que crear una figura, es decir, la ventana de la interfaz gráfica. En la figura 3.70 se muestra el código para su creación, y como se mostraría al usuario. Tenemos que darle un nombre en este caso se ha nombrado *fig (1)* (puede haber varias figuras, por eso se coloca un uno entre paréntesis, si es solamente una no hace falta).

A continuación, se utiliza el objeto *figure* y se configuran los atributos. Primero va el atributo y luego la modificación, por ejemplo, para asignarle un nombre a la figura se coloca primero '*name*' seguido del valor del atributo. En esta figura se han utilizado los atributos, *name* para cambiar el nombre de la figura, *position* que establece la posición de la figura en la pantalla y su tamaño, y el atributo *color* para cambiar el color de la figura expresado en valores RGB (Red, Green, Blue), luego se ha centrado la figura con el objeto *movegui*.



```
fig(1)=figure('name','Interfaz de la temperatura del horno','menubar','none','position',[200 200 800 700],'color',[0 0.6 0])
movegui(fig(1),'center');
```

Figura 3.70. Código para la creación de la figura.

### Creación de los axes

Se han creado dos *axes*, el primer *axe* mostrará los datos de temperatura y el segundo la señal de control. Como se puede observar en la figura 3.71 se ha utilizado el objeto *axes* y como necesitamos dos se han nombrado como *axe (1)* y *axe (2)*.

Después estos *axes* se tienen que asociar a la figura creada anteriormente, utilizando el atributo *parent* y asignándolo a *fig (1)*. Los demás atributos son unidades, que se ha colocado en pixeles, la posición de los *axes*, los límites de los ejes y finalmente la rejilla.

Luego con la función *set* se modifican los nombres de los ejes y con la función *line* se establece las líneas que van a aparecer en los *axes*. Se estableció dos líneas en el *axe 1*, una para la temperatura y otra para las referencias, y en el segundo *axe* una línea para la señal de control.

```

axe(1)=axes('parent',fig(1),'units','pixels','position',[60 380 600 280],'xlim',[0 40],'ylim',[0 100],'xgrid','on','ygrid','on')
axe(2)=axes('parent',fig(1),'units','pixels','position',[60 50 600 280],'xlim',[0 40],'ylim',[0 100],'xgrid','on','ygrid','on')

set(get(axe(1),'XLabel'),'String','Tiempo (Seg)')
set(get(axe(1),'YLabel'),'String','Temperatura (°C)')
set(get(axe(2),'XLabel'),'String','Tiempo (Seg)')
set(get(axe(2),'YLabel'),'String','Control (%)')

lin(1)=line('parent',axe(1),'xdata',[],'ydata',[],'Color','r','LineWidth',2.5);
lin(2)=line('parent',axe(1),'xdata',[],'ydata',[],'Color','k','LineWidth',2);
lin(3)=line('parent',axe(2),'xdata',[],'ydata',[],'Color','r','LineWidth',2.5);

```

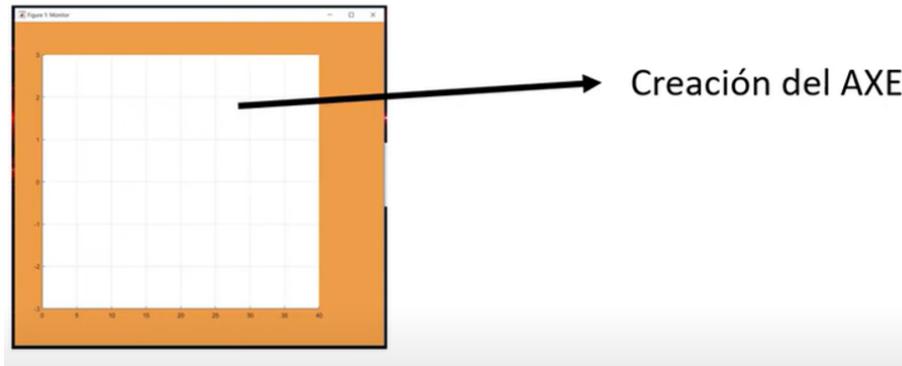


Figura 3.71. Código para la creación de los axes.

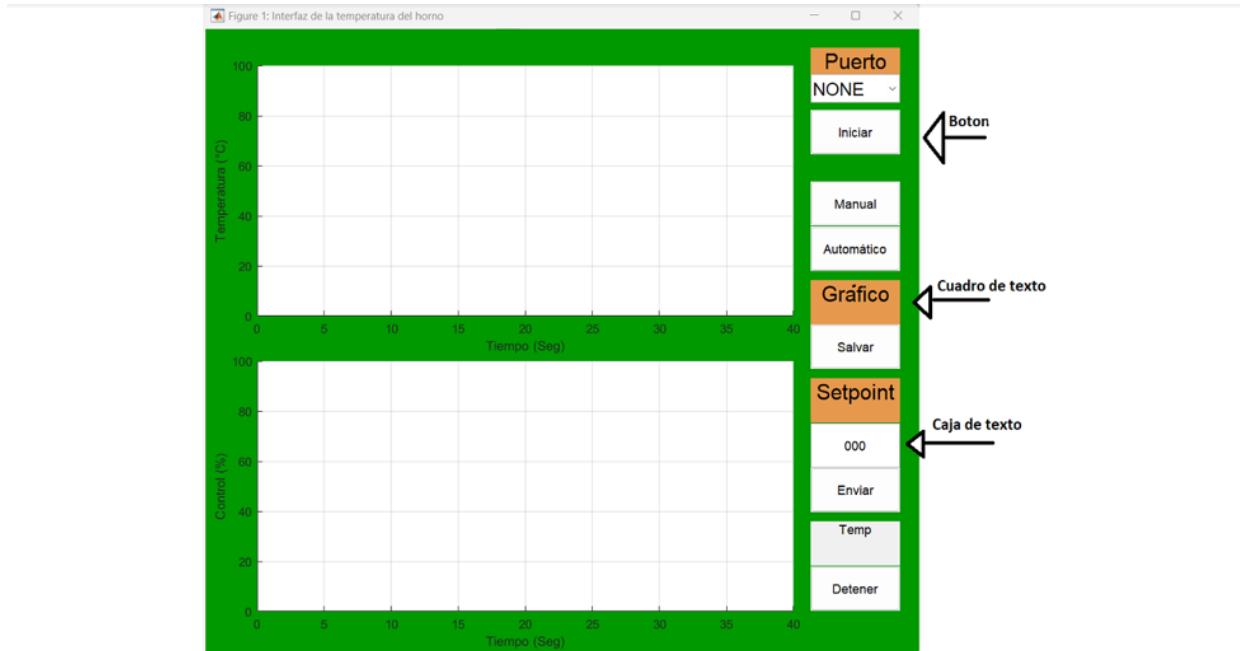
### Creación de botones, cuadros de textos y cajas de texto

El código y la visualización de los botones, cuadros de texto y cajas de texto se puede apreciar en la figura 3.72. Para crear estos elementos, se ha utilizado el objeto *uicontrol*.

Se han creado seis botones (iniciar, manual, automático, salvar, enviar y detener). Los atributos para los botones se configuran de la siguiente manera, en el primer argumento (*parent*) se asigna a *fig (1)*, es decir, donde va estar contenido el botón (que es la figura que hemos creado al principio), luego en *style* se configura como *pushbutton*, en *string* el nombre del botón, en *position* la posición donde se colocara el botón en la interfaz, luego con *callback* asignaremos la función que se va a ejecutar cuando se pulse el botón y con *fontsize* configuraremos el tamaño de la fuente en el botón.

Se creó tres cuadros de texto (Puerto, Gráfico y *Setpoint*), los atributos se configuran igual que los botones, pero aquí no se llama a una función, por eso no está el atributo, en cambio, sí se añade el atributo *BackgroundColor* que establece el color de fondo del cuadro de texto.

Finalmente se crean dos cajas de texto, una para visualizar la temperatura del horno y la otra para enviar referencias desde la interfaz de Matlab. En el atributo *style* para la caja de texto que mostrará la temperatura del horno se configura como *text* (puesto que los datos de temperatura se mostrarán como cadena de caracteres). En cambio, en la caja de texto para las referencias el atributo *style* se configura como *edit* ya que se pueden enviar diferentes referencias.



```

Texto(1)=uicontrol('parent',fig(1),'style','text','string','Puerto','position',[680 630 100 50],'BackgroundColor',[0.9 0.6 0.3],'fontsize',18);
Texto(2)=uicontrol('parent',fig(1),'style','text','string','Setpoint','position',[680 260 100 50],'BackgroundColor',[0.9 0.6 0.3],'fontsize',18);
Texto(3)=uicontrol('parent',fig(1),'style','text','string','Gráfico','position',[680 370 100 50],'BackgroundColor',[0.9 0.6 0.3],'fontsize',18);

bot(1)=uicontrol('parent',fig(1),'style','pushbutton','string','Detener','position',[680 50 100 50],'callback',@stop,'fontsize',11)
bot(2)=uicontrol('parent',fig(1),'style','pushbutton','string','Enviar','position',[680 160 100 50],'callback',@enviar,'fontsize',11)
bot(3)=uicontrol('parent',fig(1),'style','pushbutton','string','Salvar','position',[680 320 100 50],'callback',@salvar,'fontsize',11)
bot(4)=uicontrol('parent',fig(1),'style','pushbutton','string','Iniciar','position',[680 560 100 50],'callback',@iniciar,'fontsize',11)
bot(5)=uicontrol('parent',fig(1),'style','pushbutton','string','Manual','position',[680 480 100 50],'callback',@manual,'fontsize',11)
bot(6)=uicontrol('parent',fig(1),'style','pushbutton','string','Automático','position',[680 430 100 50],'callback',@automatico,'fontsize',11)
txbx(1)=uicontrol('parent',fig(1),'style','tex','string','Temp','position',[680 100 100 50],'fontsize',11)
txbx(2)=uicontrol('parent',fig(1),'style','edit','string','000','position',[680 210 100 50],'fontsize',11)

```

Figura 3.72. Código para la creación de botones, cuadros de texto y cajas de texto.

### Creación de las funciones asociadas a los botones

En la figura 3.73 y la figura 3.74 se muestra el código para las funciones asociadas a cada botón, a continuación, se explicará qué hace cada una.

El botón iniciar está asociado a la iniciación de la toma de datos de temperatura cuando previamente se ha seleccionado el puerto serie de comunicación entre el Matlab y Arduino (aqui la variable *start* toma el valor true en esta parte del código), sino se pulsa el botón iniciar no se cambia de estado a la variable llamada *ready* y se queda en un bucle el programa.

El botón detener llama a la función *stop* (parar), primero envía la trama de datos “I300F” para que reinicie el sistema del horno (desactive el transistor *MOSFET* y reinicie todas las variables) y luego cierra la conexión serie entre el Matlab y Arduino.

El botón manual indicará al sistema (Arduino) que se está en modo manual, enviando la trama de datos “I301F”. En este modo se actúa directamente sobre la señal PWM del MOSFET (actuador) para poder darle escalones al horno en la etapa de modelado o llevar al sistema a un punto de operación concreto. Hay que tener en cuenta que antes de pulsar el botón ya se ha establecido una comunicación serie y se ha pulsado el botón iniciar.

El botón automático indicará al sistema (Arduino) que se está en modo automático, enviando la trama de datos “I301F”. En este modo se activa el control PID en el Arduino que controlará al MOSFET (actuador) para que siga referencias enviadas o rechace perturbaciones. Hay que tener en cuenta que antes de pulsar el botón ya se ha establecido una comunicación serie y se ha pulsado el botón iniciar.

Con el botón enviar se envía por el puerto serial la referencia (que previamente se ha escrito en la caja de texto) como una trama de datos “Ideg1F”, el dato asociado a deg1 se obtiene con la función *get* que recoge el dato de la caja de texto.

En la caja de texto de tipo desplegable se asocia la función denominada puertas, esta asigna a la variable puerta el número de puerto que se ha detectado para crear el objeto Arduino que permitirá establecer la conexión serie.

Por último, el botón salvar activa la función graficar (previamente se debe darle a detener el sistema y parar la recogida de datos), esta función crea una gráfica en Matlab y los vectores de datos para su análisis, por ejemplo, para obtener los modelos con *System Identification Toolbox*.

### Código para actualizar los datos en los axes

En la figura 3.75 se observa el resto del código, este actualiza los datos en los *axes* de la interfaz recibidos del Arduino, se crea un bucle *while* y mientras no se pulse el botón detener o se haya alcanzado el número de iteraciones predefinidas en el código al principio del bucle, no se saldrá del bucle.

Para cada iteración, primero se lee por el puerto serie los datos que se envían desde el Arduino, estos datos llegan como un vector de cadena de caracteres, a continuación, se separan los datos y se guardan en variables, luego se actualizan los vectores dinámicos (tiempo, salida, escalón y control) con estos nuevos valores recibidos, después se espera un segundo para la siguiente iteración (los datos se actualizan cada segundo), si fuese necesario se amplían los límites de las gráficas y se actualiza el contador de iteraciones.

```
%% Funcion iniciar
function varargout=iniciar(hObject, eventdata)
    ready=false;
end
%% Funcion Pare
function varargout=stop(hObject, eventdata)
    parar=true;
    write(arduino, "I"+"300"+"F", 'char' );
    fclose(arduino);
    delete(arduino);
    clear arduino;

%% Funcion enviar
function varargout=enviar(hObject, eventdata)
    deg1=get(txbx(2), 'string' );
    if auto==true
        deg=[ "I"+deg1+"F"];
        write(arduino,deg, 'char' );
    end
    if manu==true
        deg=[ "I"+deg1+"F"];
        write(arduino,deg, 'char' );
    end

    degm=[ "I"+deg1+"F"];
    write(arduino,degm, 'char' );

end
%% Funcion manual
function varargout=manual(hObject, eventdata)
    man="301";
    write(arduino, "I"+man+"F", 'char' );
    manu=true;
    auto=false;

end
%% Funcion automático
function varargout=automatico(hObject, eventdata)
    aut="302";
    write(arduino, "I"+aut+"F", 'char' );
    auto=true;
    manu=false;

end
%% Funcion Puerto serie
function varargout=puertas(hObject, eventdata)

    puerta=popup.String{popup.Value};
end
```

Figura 3.73. Código de las funciones asociados a los botones.

```

%% Funcion Salvar
function varargout=salvar(hObject,eventdata)
    %Renombra variables
    rs=escalon;
    us=control;
    ys=salida;
    ts=tiempo;

    %Grafica datos
    figure
    subplot(2,1,1);
    plot(ts,rs,ts,ys,'linewidth',3),grid
    title('Laboratorio de Temperatura')
    xlabel('Tiempo (s)')
    ylabel('Temperatura (C)')

    subplot(2,1,2);
    plot(ts,us,'linewidth',3),grid
    xlabel('Tiempo (s)')
    ylabel('Control (%)')

    T=[ts;rs;ys;us];
    filter = {'*.txt'; '*.xls'};
    [file,path,indx] = uiputfile(filter);
    fileID = fopen(strcat(path,file),'w');
    fprintf(fileID, '%12s %12s %12s %12s\n', 't', 'r', 'y', 'u');
    fprintf(fileID, '%12.2f %12.2f %12.2f %12.2f\n', T);
    fclose(fileID);
    file(end-2:end)='mat';
    save(strcat(path,file), 'ts', 'rs', 'ys', 'us')

end
%Vectores dinámicos
tiempo=[0];
salida=[0];
escalon=[0];
control=[0];
deg1="0";
dt=1;
limx=[0 40];
limy=[0 100];
set(axe(1),'xlim',limx,'ylim',limy);
while(ready)
    pause(1);
end
if start
    % Configura el Puerto Serial

    arduino=serialport(puerta,9600,"StopBits",1,"DataBits",8);
    fopen(arduino);
    % Aquí se usa flushinput() para limpiar el buffer
    flushinput(arduino);
    %% Grafic
    k=5; nit =200000;

```

Figura 3.74. Código de las funciones asociadas a los botones y comienzo para graficar los axes.

```

while(~parar)
    % Lectura de Datos por Puerto Serial
    % Leo datos del arduino, el tiempo, la temperatura y la señal de control
    data=readline(arduino);
    % Separo los datos y los convierto en un vector de caracteres
    Cadena = strsplit(data);
    % El primer dato de cadena lo transformo de cadena a un numero (double) y
    % dividido entre mil
    time=((str2double(Cadena(1)))/1000);
    %El segundo dato de cadena lo transformo de cadena a un numero (double) y
    % dividido entre cien
    temp=(str2double(Cadena(2))/100;
    tempC=string(temp);
    ctr=str2double(Cadena(3));
    %cargo el valor de temperatura en la caja de texto
    set(txbx(1),'string',tempC);
    %Actualiza las variables del grafico
    tiempo=[tiempo time];
    salida=[salida temp];
    control=[control ctr];
    escalon=[escalon str2double(deg1)];
    %Cargo los datos en los axes
    set(lin(1),'xdata',tiempo,'ydata',salida);
    set(lin(2),'xdata',tiempo,'ydata',escalon);
    set(lin(3),'xdata',tiempo,'ydata',control);
    %Se espera un segundo para volver a recoger los datos
    pause(dt);
    % actualizo grafica cuando llega a su limite en tiempo real
    if tiempo(end)>=limx
        limx=[0 limx(2)+40];
        set(axe(1),'xlim',limx) ;
        set(axe(2),'xlim',limx);
    end
    %actualizo grafica cuando llega a su limite en tiempo real
    if salida(end)>=limy
        limy=[0 limy(2)+10];
        set(axe(1),'ylim',limy);
    end
    %actualizo grafica cuando llega a su limite en tiempo real
    if escalon(end)>=limy
        limy=[0 escalon(end)+10];
        set(axe(1),'ylim',limy);
    end
    %Actualizo el contador
    k=k+1;
    if(k==nIt)
        parar=true;
    end

    end
    parar=false;
end
end

```

Figura 3.75. Código de para actualizar los axes de la interfaz.

### 3.5 Programación del Arduino

En la figura 3.76 y la figura 3.77 se muestra el código donde se incluyen las librerías que se han utilizado, se define los pines del Arduino para el sensor, actuador, los leds, el ventilador, se crea el objeto para la pantalla OLED y se definen las constantes para el envío de datos cada segundo, y el tiempo de muestreo cada 20 segundos. También se definen todas las variables necesarias para la programación del código restante, por ejemplo, las variables booleanas para el cambio de modos (manual o automático), para el cálculo del PID, etc.

```
/*===== DEFINICIONES COMPONENTES, ACTUADOR,LEDS,  
LIBRERIAS =====*/  
// Librerias utilizadas  
#include "Horno.h"  
#include <SPI.h>  
#include <Wire.h>  
#include <Adafruit_GFX.h>  
#include <Adafruit_SSD1306.h>  
// Sensor Pt100  
#define PT100 A4  
// Actuador MOSFET  
#define MOSFET 6  
// Indicadores luminosos LEDS  
#define COLD 8  
#define HOT 10  
// Ventilador  
#define FAN 40  
// Definir constantes de la pantalla  
// Ancho pantalla OLED  
#define ANCHO_PANTALLA 128  
// Alto pantalla OLED  
#define ALTO_PANTALLA 64  
// Creación del objeto display  
Adafruit_SSD1306 display(ANCHO_PANTALLA, ALTO_PANTALLA, &Wire,  
-1);  
// Definicion de constantes  
const unsigned long INTERVALO_DATOS = 1000UL;  
const unsigned long INTERVALO_MUESTREO = 20000UL;  
unsigned long actual, actual2;  
unsigned long evento_datos = 1000UL;  
unsigned long evento_muestreo = 0UL;  
unsigned long ultimoCambioModo = 0.0;  
const float parar = 300.0;  
const float auxM = 301.0;  
const float auxA = 302.0;  
bool bandeA = false;  
bool bandeM = false;  
bool bandeR = false;  
bool automatico = false;  
bool cambioModo = false;
```

Figura 3.76. Código para declaración de variables e inclusión de librerías (parte 1).

```
/*===== VARIABLES GLOBALES =====*/
//Temperatura del sensor
float T1;
//Actuador
float H1 = 0.0;
// Referencia
float r = 0.0;
float auxmodo;
//Parámetros del PID teóricos interactivos
float kc, ti, td;
//Parámetros del PID teóricos esquema paralelo
float kp, ki, kd;
//Variables para el controlador
float proporcional, derivativo, aux_u, Tt;
float lastintegral = 0.0;
float integral;
float es = 0.0;
float salida, lastsalida, dsalida;
//Señal de control
float u_control, u_max, u_min;
//Señal del error
float error;
float lasterror;
//Variables auxiliares
String comparar;
//Variable para definir el modo de control Manual o
Automatico
bool controlA = false;
bool controlM = false;
bool Control = false;
/*===== MODELADO DEL SISTEMA =====*/
float K = 6.1826, tau1 = 452.62, tau2 = 334.93, tr = 10;
float lmd = 394;
//Periodo de Muestreo
int Ts = 20;
```

Figura 3.77. Código para declaración de variables e inclusión de librerías (parte 2).

En la figura 3.78, en el procedimiento *Setup* se inicializa los pines digitales del Arduino como salidas, para los leds que indicarán si el horno está por encima o debajo de 30°C, y el ventilador. A continuación, se inicializa dichos pines con un valor bajo, es decir, un cero digital (empezarán apagados).

Luego se indicará al Arduino que se utilizará la referencia de tensión interna de 2.56 voltios, después se configura el puerto serial, se establece un *timeout* para las comunicaciones de 100 milisegundos, se apaga el actuador, se inicializa la pantalla OLED, se cargan los valores del controlador no interactivo (calculados en Matlab para evitar hacer los cálculos dentro del Arduino) y se pasan a paralelo, se establece el tiempo de tracking para el *antiwindup* y finalmente se establecen los límites del actuador de 0 a 60 (ya que para obtener los modelos solo se dio un máximo de 60 por ciento de PWM, si se daba un escalón superior saturaba el sensor).

```
/*===== FUNCION SETUP =====*/
void setup() {
    pinMode(hot, OUTPUT); //Led "Caliente" como salida
    pinMode(cold, OUTPUT);
    pinMode(fan, OUTPUT);
    digitalWrite(hot, LOW);
    digitalWrite(cold, LOW);
    digitalWrite(fan, LOW);
    analogReference(INTERNAL2V56); //Referencia analógica PIN AREF
(2.56V)
    //Configuración del puerto serial
    Serial.begin(9600);
    Serial.setTimeout(100);
    analogWrite(mosfet, 0);
    //Inicialización de la pantalla oled
    Pantalla();
    /*===== PARAMETROS DEL CONTROLADOR NO INTERACTIVO
=====
    kc = 0.3153;
    ti = 787.55;
    td = 192.4907;

    //Pasar los parámetros del controlador no interactivo a paralelo
    kp = kc;
    ki = (kc) / (ti);
    kd = kc * (td);
    //Tiempo de tracking para el back calculation
    Tt = 389.3533;
    //Definición de los valores máximos del actuador (%PWM)
    u_max = 60.0;
    u_min = 0.0;
}
```

Figura 3.78. Código del Setup.

En la figura 3.79 se observa el procedimiento *loop*, aquí se asigna a la variable (*actual*) el valor de la función *millis* que servirá para establecer el tiempo transcurrido desde que se energizo el Arduino y que servirá como tiempo de referencia para enviar datos y ejecutar el control PID.

A continuación, se declaran dos variables de tipo *string* para la recogida de datos provenientes del puerto serie, luego se ejecutan los procedimientos iluminación y ventilación (se explicarán más adelante). Luego comprobará si llega un dato en el puerto serie (*cambioModo* al principio es falso). Si llegan datos por el puerto serie, es decir, una trama tipo “IdatoF” se guardará en la variable *dato*, se sustrae el dato (quitando la I y la F de la trama) y se guarda en *degC*, luego se cambia el tipo de dato a entero y se almacena en la variable *auxmodo*, después se compara *auxmodo* con las constantes *parar*, *auxA* y *auxM*.

Si es igual a *parar* se reinicia el sistema, si es igual a *auxA*, se pone en modo automático y se inicializa a contar el tiempo en la variable *ultimoCambioModo* para el control PID, también cambia la variable *cambioModo* a verdadero (para que no vuelva a utilizarse el puerto serie al principio del *loop* puesto que ya estamos dentro de un modo). Si es igual a *auxM* se entra en modo manual y se cambia a verdadero la variable *cambioModo*.

Luego, si se ha elegido el modo automático se ejecutará el procedimiento control automático (tiene como finalidad recoger datos para la referencia o parar el sistema por el puerto serie), después comprueba si ha pasado el tiempo de muestreo y ejecuta *PID\_Back\_calculation* (control PID).

Si se ha elegido el modo manual, se activa una bandera que indicará al puerto serie que reciba referencias para el control manual, sino se actualiza la referencia con el valor de la salida (temperatura), se asigna a *lastintegral* el valor de H1 (actuador), *lastsalida* con el valor de la referencia. El último condicional hace que se envie los datos a la interfaz gráfica cada un segundo y se actualice la pantalla OLED.

```
/*===== FUNCION PRINCIPAL =====*/
void loop() {
    actual = millis();
    String dato, degC;
    Iluminacion();
    Ventilacion();

    if (!cambioModo && Serial.available()) {
        //leemos el dato enviado
        String dato = Serial.readString();
        int ini = dato.indexOf('I') + 1;
        int fin = dato.indexOf('F', ini);
        if (ini > 0 && fin > ini) {
            String degC = dato.substring(ini, fin);
            auxmodo = degC.toFloat();

            if (auxmodo == parar) {
                Reset();
            }

            if (auxmodo == auxA) {
                controlA = true;
                cambioModo = true;
                ultimoCambioModo = actual;
                Control = true;
            }

            if (auxmodo == auxM) {
                controlM = true;
                cambioModo = true;
                Control = true;
            }
        }
    }

    if (controlA) {
        ControlAutomatico();
        if (actual - ultimoCambioModo >= evento_muestreo) {
            evento_muestreo = INTERVALO_MUESTREO;
            ultimoCambioModo = actual;
            PID_Back_calculation();
        }
    } else {
        ControlManual();
    }

    if (actual >= evento_datos) {
        EnviarDatos();
        evento_datos += INTERVALO_DATOS;
        Apantalla();
    }
}
```

Figura 3.79. Código del loop.

En la Figura 3.80 se muestran los procedimientos *reset*, iluminación y ventilación. El procedimiento *reset* reinicia todas las variables a cero y las variables booleanas a falso. El procedimiento iluminación activa el led rojo si es mayor que 30 °C y el led verde si es menor. Por último, el procedimiento ventilación enciende el ventilador si el actuador tiene un PMW mayor que 5.

```
void Reset(void) {
    H1 = 0.0;
    analogWrite(mosfet, 0);
    bandeA = false;
    bandeM = false;
    Control = false;
    controlA = false;
    controlM = false;
    automatico = false;
    cambioModo = false;
    proporcional = 0.0;
    derivativo = 0.0;
    lastintegral = 0.0;
    es = 0.0;
    salida = 0.0;
    lastsalida = 0.0;
    integral = 0.0;
    error = 0.0;
    auxmodo = 0.0;
    r = 0.0;
    aux_u = 0.0;
    u_control = 0.0;
    ultimoCambioModo = 0.0;
    evento_muestreo = 0.0;
}

void Ventilacion(void) {

    //Indicación Luminica
    if (H1 > 5.0) {
        //HORNO CALIENTE
        digitalWrite(fan, HIGH);
    } else {
        //HORNO FRIO
        digitalWrite(fan, LOW);
    }
}

void Iluminacion(void) {

    //Indicación Luminica
    if (T1 > 30.0) {
        //HORNO CALIENTE
        digitalWrite(hot, HIGH);
        digitalWrite(cold, LOW);
    } else {
        //HORNO FRIO
        digitalWrite(cold, HIGH);
        digitalWrite(hot, LOW);
    }
}
```

Figura 3.80. Código de la función *reset*, ventilación y iluminación.

En la figura 3.81 se muestra el código del procedimiento manual, como se puede observar es parecido al código en el *loop* del puerto serie (no se volverá a explicar de nuevo). El puerto serie se activará si previamente se ha elegido el modo manual y *controlM* es verdadero (se vuelve verdadero en la selección de modo manual). Una vez recibido por el puerto serie el valor del escalón (valor de PWM desde la interfaz gráfica), se actualiza la variable *lastingetral* con este valor (para cuando se realice el cambio a automático en el régimen permanente se produzca la transferencia sin saltos), luego actualiza las variables referencia y *lastsalida* con el valor de la salida (temperatura).

```
/*== FUNCION DEL CONTROL MANUAL ==*/
void ControlManual(void) {
    static String datoManual, Maux;
    static float HManual;

    if (controlM && Serial.available()) {
        //leemos el dato enviado
        String datoManual = Serial.readString();
        int ini = datoManual.indexOf('I') + 1;
        int fin = datoManual.indexOf('F', ini);
        if (ini > 0 && fin > ini) {
            String Maux = datoManual.substring(ini, fin);
            HManual = Maux.toFloat();

            if (HManual == parar) {
                Reset();
            }
            if (HManual == auxA) {
                controlM = false;
                controlA = true;
                ultimoCambioModo = actual;
            }
        }

        if ((HManual != parar) && (HManual != auxA)) {
            H1 = HManual;
            bandeM = true;
        }
    }
}

if (H1 > 60.0) {
    H1 = 60.0;
}
if (H1 < 0.0) {
    H1 = 0.0;
}
lastintegral = H1;
es = 0.0;
r = TempRead(pt100);
lastsalida = r;
analogWrite(mosfet, map(H1, 0, 100, 0, 255));
}
```

Figura 3.81. Código de la función control manual.

En la Figura 3.82 se observa el procedimiento *controlAutomatico* este tiene la función de recoger las referencias o parar el sistema desde de la interfaz de Matlab cuando está en modo automático.

```
// /*== FUNCION DEL CONTROL AUTOMATICO ==*/
void ControlAutomatico(void) {
    static String datoAuto, Aaux;
    static float HAuto;

    if (controlA && Serial.available()) {
        //leemos el dato enviado
        String datoAuto = Serial.readString();
        int ini = datoAuto.indexOf('I') + 1;
        int fin = datoAuto.indexOf('F', ini);
        if (ini > 0 && fin > ini) {
            String Aaux = datoAuto.substring(ini, fin);
            HAuto = Aaux.toFloat();

            if (HAuto == parar) {
                Reset();
            }
            if (HAuto != parar) {
                r = HAuto;
                bandeA = true;
            }
        }
    }
}
```

Figura 3.82. Código del procedimiento control automático.

En la Figura 3.83 se muestran los procedimientos *Pantalla* y *EnviarDatos*. El primero inicializa la pantalla OLED y el segundo envía los datos de temperatura, señal de control y tiempo a la interfaz gráfica de Matlab.

```
void Pantalla(void) {

    // Iniciar pantalla OLED en la dirección 0x3C
    if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
#ifdef __DEBUG__
        Serial.println("No se encuentra la pantalla OLED");
#endif
        while (true)
            ;
    }
    // Clear the buffer.
    display.clearDisplay();
}

void EnviarDatos(void) {
    //Usar la interfaz de Matlab
    T1 = TempRead(pt100);
    char str[50];
    sprintf(str, "%lu %d %d\n", millis(), int(T1 * 100),
int(H1));
    Serial.print(str);
}
```

Figura 3.83. Código del procedimiento pantalla y enviar datos.

A continuación, en la figura 3.84, se observa el código del controlador PID, primero se lee la salida del sensor, luego se calcula el error y la variación de la salida. Después se calcula la acción proporcional, la integral (aquí se introdujo el *antiwindup* utilizando el esquema *back calculation*) y la derivativa.

Después se le asigna a la variable *u\_control* el valor de la suma de cada componente del PID y se guarda el *u\_control* en una variable auxiliar *aux\_u*. Luego se comprueba si la señal de control (*u\_control*) está dentro de los límites del actuador, después se asigna a la variable *es* (error de saturación) la diferencia entre *u\_control* y *aux\_u*. Por último, se asigna a *H1* el valor de *u\_control* y se envía al MOSFET (actuador), finalmente se actualizan las variables *lasterror*, *lastsalida*, y *lastintegral*.

En la figura 3.85 se muestra el código para cargar los datos en la pantalla de temperatura, señal de control y si se encuentra encendido o apagado el horno.

```
/*Calculo del controlador utilizando aproximaciones euler
hacia atrás para la integral junto con back calculation
para el antiwindup,el derivativo utiliza la aproximación
hacia atrás*/
void PID_Back_calculation(void) {

    //leer salida actual
    salida = TempRead(pt100);
    //Cálculo de la señal del error actual
    error = r - salida;
    //Cálculo de la señal de salida actual
    dsalida = salida - lastsalida;
    //Cálculo de la Acción proporcional
    proporcional = kp * error;
    // Cálculo del termino integral
    integral = lastintegral + Ts * (ki * error + (es/(Tt)));
    // Cálculo del termino derivativo
    derivativo = -(kd * dsalida) / Ts;
    //Cálculo de la señal de control actual
    u_control = proporcional + integral + derivativo;
    aux_u = u_control;
    if (u_control > u_max) {
        u_control = u_max;
    }
    if (u_control < u_min) {
        u_control = u_min;
    }
    es = u_control - aux_u;
    H1 = u_control;
    //Aplica la acción de control en el PWM
    analogWrite(mosfet, map(H1, 0, 100, 0, 255));

    //Actualización de las señales de error y salida
    lasterror = error;
    lastsalida = salida;
    lastintegral = integral;
}
```

Figura 3.84. Código del procedimiento del controlador PID.

```
void Apantalla(void) {
    static const unsigned char PROGMEM datos_imagen[72] = {
        0xff, 0xff, 0xbf, 0xff, 0xec, 0x9f, 0xff, 0xc4,
        0xa7, 0xff, 0x84, 0xb3, 0xff, 0x24, 0xa9, 0x8e, 0xd4,
        0xa4, 0x01, 0x94, 0xa3, 0xff, 0x14, 0xaf, 0xff, 0xd4,
        0xbff, 0xff, 0xf4, 0xbff, 0xff, 0xf4, 0xbff, 0xff, 0xf4,
        0xb8, 0xfc, 0xf4, 0xbd, 0xfc, 0xf4, 0xbff, 0xff, 0xf4,
        0xbff, 0x8f, 0xf4, 0xa0, 0x84, 0x34, 0x9d, 0xcf, 0xec,
        0xd6, 0x5b, 0xec, 0xc9, 0xce, 0xdc, 0xe6, 0x73, 0x9c,
        0xfe, 0x3c, 0xfc, 0x70, 0xfc, 0xff, 0x03, 0xfc };
    static String temp, control;
    temp = String(T1, 2);
    control = String(H1, 2);
    // Tamaño del texto
    display.setTextSize(2);
    // Color del texto
    display.setTextColor(SSD1306_WHITE);
    // Posición del texto
    display.setCursor(0, 0);
    //Dibujar el mapa de bits en la pantalla
    display.drawBitmap(106, 40, datos_imagen, 22, 24, SSD1306_WHITE);
    display.display();
    // Escribir texto
    display.print(temp);
    display.setCursor(70, 0);
    display.setTextSize(1);
    display.write(248);
    display.setTextSize(2);
    display.println("C");
    display.print(control);
    display.setCursor(70, 16);
    display.write(37);
    display.print("PWM");
    // Enviar a pantalla
    display.display();
    if (Control) {
        display.setCursor(0, 48);
        display.println("ON");
        display.display();
    } else {
        display.setCursor(0, 48);
        display.println("OFF");
        display.display();
    }
    display.clearDisplay();
}
```

Figura 3.85. Código para actualizar la pantalla OLED.

## 3.6 Modelado del horno

### 3.6.1 Obtención de los modelos

Una vez realizado el código para el Arduino y la interfaz gráfica de Matlab, y verificado que funcionan correctamente, se procedió a modelar el sistema del horno, en la figura 3.86 se muestran los escalones realizados y las respuestas del sistema a dichos escalones.

Se dieron tres escalones de subida, del 20% de PWM cada uno, (no se dio un escalón superior ya que saturaría el sensor), y a continuación, tres escalones de bajada.

Se puede observar que el sistema se puede modelar con funciones de transferencia de primer orden o de segundo orden sobreamortiguado. También se observa el comportamiento no lineal del sistema, ya que no se obtiene para cada escalón la misma función de trasferencia, sino que al inicio la ganancia es grande (en los sistemas térmicos ocurre esto), luego va disminuyendo la ganancia, además las constantes de tiempo son distintas para cada escalón.

Otra cosa para tener en cuenta es que el sistema es bastante lento, necesitando cerca de una hora para llegar al régimen permanente en cada escalón. También en los ensayos, como se refleja en la gráfica, una vez dado el escalón se tiene que esperar a que la interfaz gráfica dicho escalón (va con un poco de retardo), sino se produce como una desconexión (si se envían varias veces el escalón, en este caso no afecto al ensayo solo afecto a la visualización).

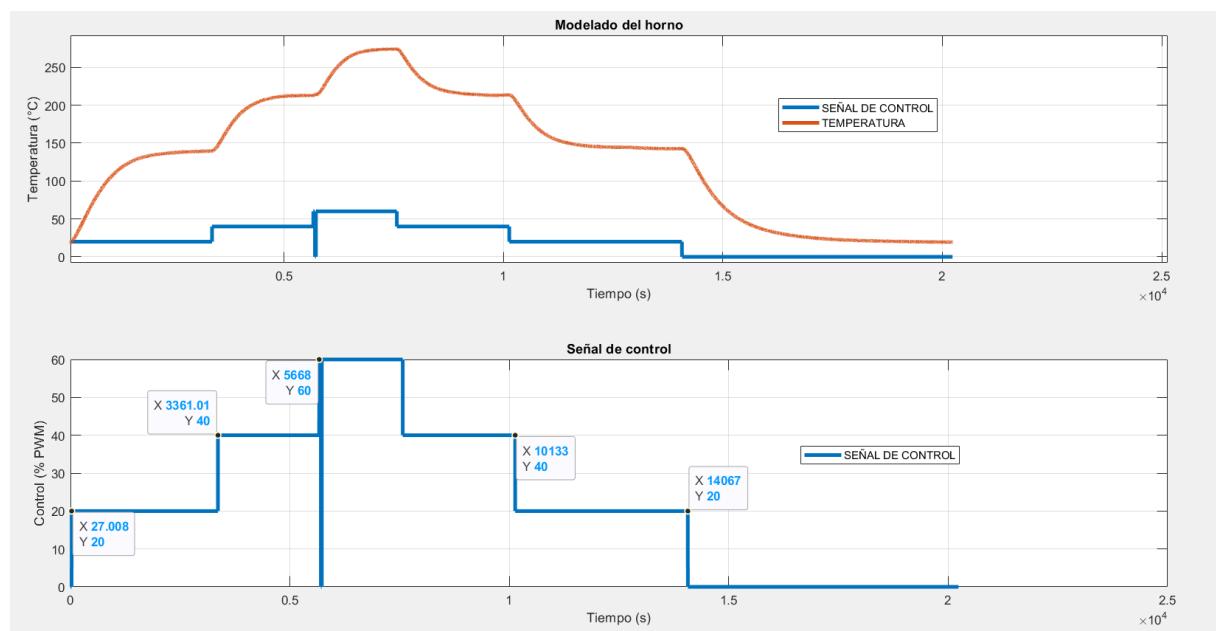


Figura 3.86. Escalones realizados para obtener las funciones de transferencia.

A continuación, una vez guardados los datos, se procede a subdividirlos con el código que se muestra en las figuras 3.87, 3.88 y 3.89.

En el código primero se cargan los datos del ensayo, luego se gráfica todo el ensayo, a continuación, se selecciona el valor del escalón que queremos subdividir, por ejemplo, para el primer escalón ascendente se daría el valor de 20 a la variable *b*, 40 para el segundo escalón ascendente, 60 para el tercer escalón ascendente, para los escalones descendentes se dan los valores -40, -20 y 0 respectivamente en orden de descenso.

El script se encarga de recortar los vectores de datos, los traslada al origen y los guarda en *ut* (señal de control trasladada y recortada), *yt* (señal de salida trasladada y recortada) y *tt* (vector de tiempo recortado y trasladado).

```
%Programa de Identificación
load('datos_modelado_horno.mat')
plot(ts,us,ts,ys,'linewidth',3),grid
title('Datos de la identificación')
xlabel('Tiempo (s)')
ylabel('Temperatura (C)')
% Escalon donde se hizo la IDENTIFICACION 20 40 60 -40 -20 0
% los valores negativos son para los escalones descendentes
b=20;
switch (b)
    case 20
        %Busca el momento exacto donde se insertó el escalón
        i=1;
        while(us(i)<b)
            i=i+1;
        end
        x1=i;
        while(us(i)==b)
            i=i+1;
        end
        x2=i-1;
        %Recortar datos hasta el origen
        ur=us(x1:x2)';
        yr=ys(x1:x2)';
        tr=ts(x1:x2)';
        %transladar los datos al origen
        ut=ur-us(1); %us(1) tiene valor 0 aqui
        yt=yr-yr(1);
        tt=tr-tr(1);
        %Graficar escalon Trasladado
        figure(2)
        plot(tt,ut,tt,yt,'linewidth',3),grid
        title('Datos recortados y transladados')
        xlabel('Tiempo (s)')
        ylabel('Temperatura (C)')
    case 40
        %Busca el momento exacto donde se insertó el escalón
        i=1;
        while(us(i)<b)
            i=i+1;
        end
        x1=i;
        while(us(i)==b)
            i=i+1;
        end
        x2=i-1;
        %Recortar datos hasta el origen
        ur=us(x1:x2)';
        yr=ys(x1:x2)';
        tr=ts(x1:x2)';
        %transladar los datos
        ut=ur-20; % se resta 20 porque se ha dado un escalon de 20
        % no de 40
        yt=yr-yr(1);
        tt=tr-tr(1);
        %Graficar escalon Trasladado
        figure(2)
        plot(tt,ut,tt,yt,'linewidth',3),grid
        title('Datos recortados y transladados')
        xlabel('Tiempo (s)')
        ylabel('Temperatura (C)')
```

Figura 3.87. Código para recortar los datos del modelado del horno (parte1).

```
case 60
    %Busca el momento exacto donde se insertó el escalón
    i=1;
    while(us(i)<b)
        i=i+1;
    end
    x1=i;
    while(us(i)==b)
        i=i+1;
    end
    x2=i-1;
    %Recortar datos hasta el origen
    ur=us(x1:x2)';
    yr=ys(x1:x2)';
    tr=ts(x1:x2)';
    %transladar los datos
    ut=ur-40;
    yt=yr-yr(1);
    tt=tr-tr(1);
    %Graficar escalon Traslado
    figure(2)
    plot(tt,ut,tt,yt,'linewidth',3),grid
    title('Datos recortados y transladados')
    xlabel('Tiempo (s)')
    ylabel('Temperatura (C)')
case -40
    %Busca el momento exacto donde se insertó el escalón
    % iniciamos los datos en u(13628)=60, cambiar si
    es necesario
    i=13628;
    e=40;
    while(us(i)>e)
        i=i+1;
    end
    x1=i;
    while(us(i)==e)
        i=i+1;
    end
    x2=i-1;
    %Recortar datos hasta el origen
    ur=us(x1:x2)';
    yr=ys(x1:x2)';
    tr=ts(x1:x2)';
    %transladar los datos
    ut=ur-60;
    yt=yr-yr(1);
    tt=tr-tr(1);
    %Graficar escalon Traslado
    figure(2)
    plot(tt,ut,tt,yt,'linewidth',3),grid
    title('Datos recortados y transladados')
    xlabel('Tiempo (s)')
    ylabel('Temperatura (C)')
```

Figura 3.88. Código para recortar los datos del modelado del horno (parte2).

```
case -20
%Busca el momento exacto donde se insertó el escalón
% iniciamos los datos en u(17849)=40, cambiar si
    es necesario
i=17849;
e=20;
while(us(i)>e)
    i=i+1;
end
x1=i;
while(us(i)==e)
    i=i+1;
end
x2=i-1;
%Recortar datos hasta el origen
ur=us(x1:x2)';
yr=ys(x1:x2)';
tr=ts(x1:x2)';
%transladar los datos
ut=ur-40;
yt=yr-yr(1);
tt=tr-tr(1);
%Graficar escalon Trasladado
figure(2)
plot(tt,ut,tt,yt,'linewidth',3),grid
title('Datos recortados y transladados')
xlabel('Tiempo (s)')
ylabel('Temperatura (C)')
case 0
%Busca el momento exacto donde se insertó el escalón
% iniciamos los datos en u(17849)=40,datos
verificacion cambiar si es necesario
i=22755;
e=0;
while(us(i)>e)
    i=i+1;
end
x1=i;
x2=length(us);
%Recortar datos hasta el origen
ur=us(x1:x2)';
yr=ys(x1:x2)';
tr=ts(x1:x2)';
%transladar los datos
ut=ur-20;
yt=yr-yr(1);
tt=tr-tr(1);
%Graficar escalon Trasladado
figure(2)
plot(tt,ut,tt,yt,'linewidth',3),grid
title('Datos recortados y transladados')
xlabel('Tiempo (s)')
ylabel('Temperatura (C)')
end
```

Figura 3.89. Código para recortar los datos del modelado del horno (parte3).

Con los datos recortados  $ut$ ,  $yt$  y  $tt$  (por ejemplo, del primer escalón), se abre la herramienta *System Identification Toolbox*, a continuación, se abre una ventana como la de la figura 3.90 y se exportan los datos seleccionando la opción *time domain signals*. Luego en input introducimos el vector  $ut$  y en output  $yt$ , le asignamos un nombre, se empieza en cero y se da un tiempo de muestreo de 1 (se ha graficado cada segundo en la interfaz gráfica de Matlab). Después estos datos se mueven al recuadro *working data* y *validation data* para estimar el modelo.

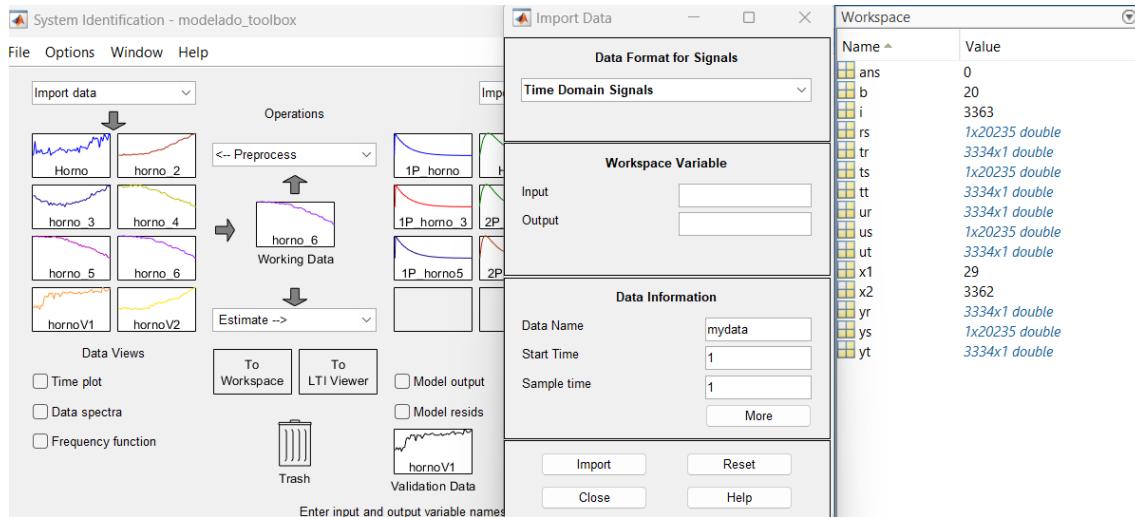


Figura 3.90. Ventana para cargar datos en system identification.

A continuación, en el desplegable *estimate* seleccionamos la opción *process models* y se nos abrirá la ventana de la figura 3.91, aquí indicamos a la herramienta cuantos polos vamos a incluir en el modelo, si son reales o complejos, y si se incluye tiempo de retardo, finalmente se pulsa a *estimate* y aparecerá a la derecha el modelo estimado.

Se modelaron las funciones de transferencia con dos polos y con tiempo de retardo (un polo da una estimación del 87% mientras que con dos polos se obtiene un 97%). Este proceso se repite para los 5 escalones restantes, en la tabla 3.2 se muestra los modelos obtenidos.

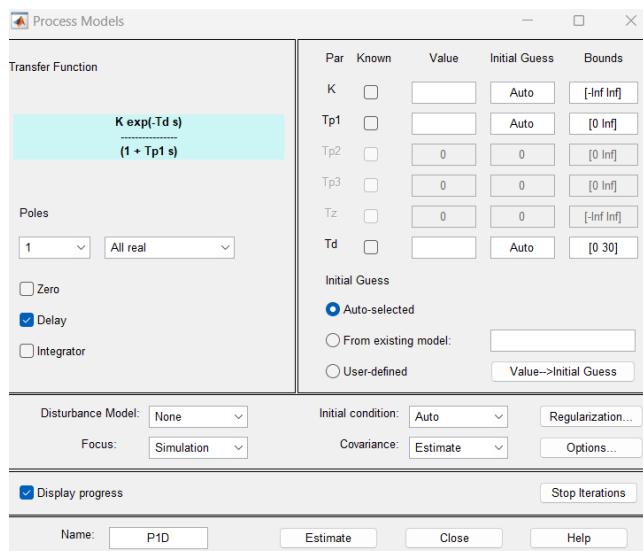


Figura 3.91. Ventana de process models en system identification.

	Rango de temperatura	Modelo obtenido
Escalón de 0 a 20	16°C-140°C	$P(s) = \frac{6.1826 e^{-10s}}{(452.62s + 1)(334.93s + 1)}$
Escalón de 20 a 40	140°C-213°C	$P(s) = \frac{3.69}{(375.21s + 1)(216s + 1)}$
Escalón de 40 a 60	213°C-274°C	$P(s) = \frac{3}{(254.23s + 1)(244.24s + 1)}$
Escalón de 60 a 40	274°C-213°C	$P(s) = \frac{3}{(423.62s + 1)(119.92s + 1)}$
Escalón de 40 a 20	213°C-144°C	$P(s) = \frac{3.5188 e^{-22.61s}}{(520.8s + 1)(111.73s + 1)}$
Escalón de 20 a 0	144°C-19°C	$P(s) = \frac{6.1385 e^{-30s}}{(849.71s + 1)(90.97s + 1)}$

Tabla 3.2. Modelos obtenidos.

### 3.6.2 Verificación de los modelos

Una vez obtenidos los modelos, para verificarlos se volvió a hacer otro ensayo con tres escalones de subida y tres de bajada, en la figura 3.92 se muestra el resultado obtenido.

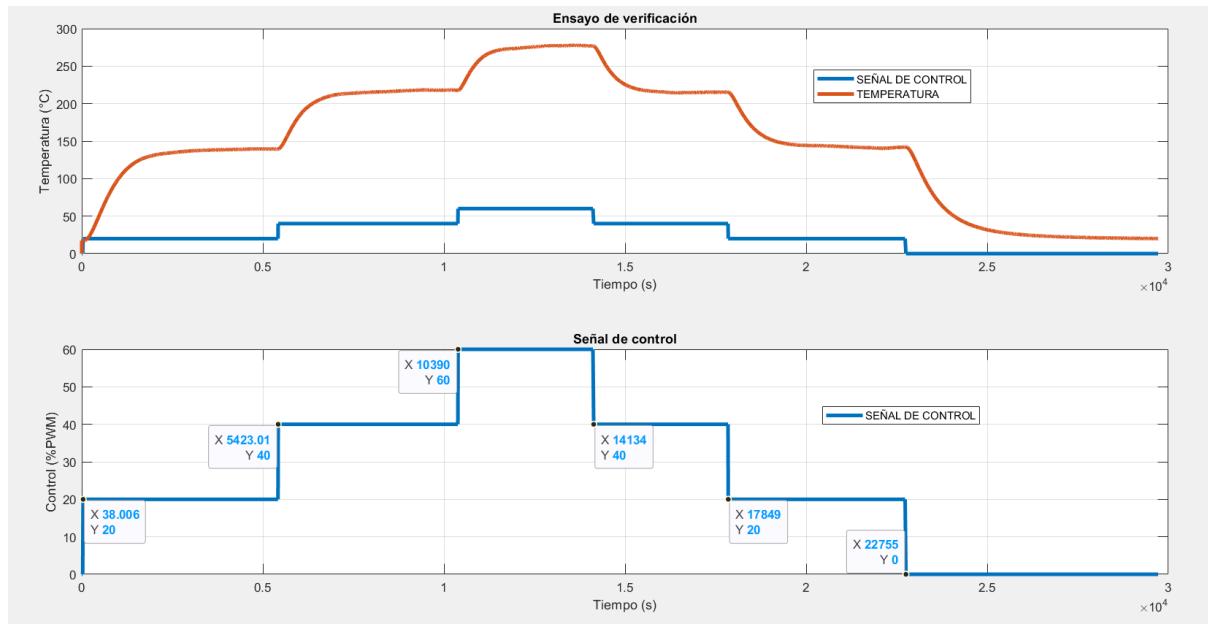


Figura 3.92. Ensayo de verificación (Repetida de la figura 1.2 por conveniencia).

Se vuelven a dividir los datos (para cada escalón) como en la etapa de modelado, y estos datos serán los que se colocarán en el cuadro de *validation data* de *System Identification Toolbox*. En la figura 3.93 se muestra la verificación para el primer escalón, en donde se compara el modelo obtenido en el primer ensayo y se solapan los datos del ensayo de verificación, se puede apreciar que tiene una similitud del 92.56%, por tanto, el modelo se da como válido.

Se procede con el segundo escalón, en la figura 3.94 se observa el resultado de verificación obtenido, aquí se obtiene una similitud del 84.92%, no obstante, se puede apreciar que es debido a discrepancias en el régimen permanente, la parte del transitorio son muy similares, se da por bueno el modelo.

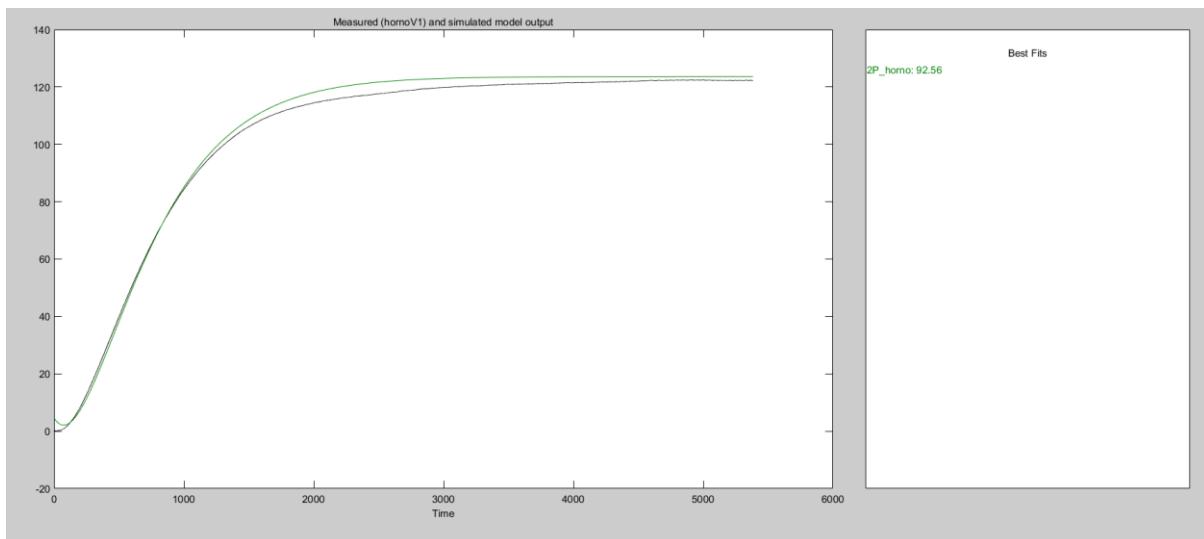


Figura 3.93. Verificación del modelo obtenido en el primer escalón ascendente (Repetida de la figura 1.2 por conveniencia).

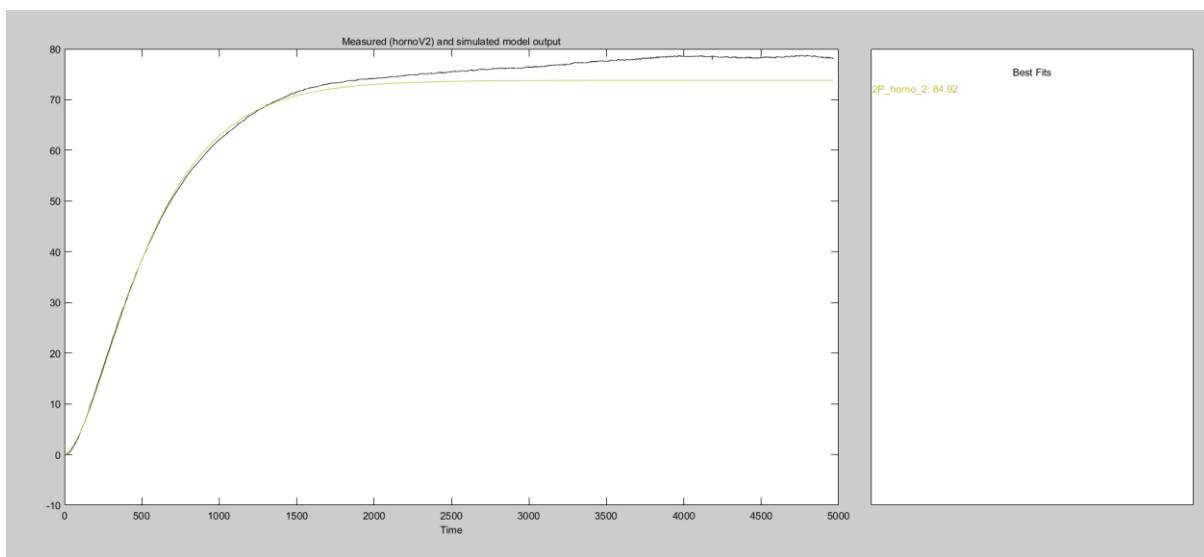


Figura 3.94. Verificación del modelo obtenido en el segundo escalón ascendente

En la figura 3.95 se muestra el resultado de verificación obtenido en el tercer escalón ascendente, aquí se obtiene una similitud del 87.42%, se puede apreciar que es debido a discrepancias al final del transitorio, la parte del régimen permanente son muy similares, se da por bueno el modelo.

Para el primer escalón descendente, como se puede apreciar en la figura 3.96, se obtuvo una similitud del 94.74%, la parte del transitorio y del régimen permanente tienen mucha similitud, se da por bueno el modelo.

En el segundo escalón descendente se obtuvo una similitud del 85.49%, como se puede apreciar en la figura 3.97, las discrepancias se producen en el régimen permanente, se da como válido el modelo.

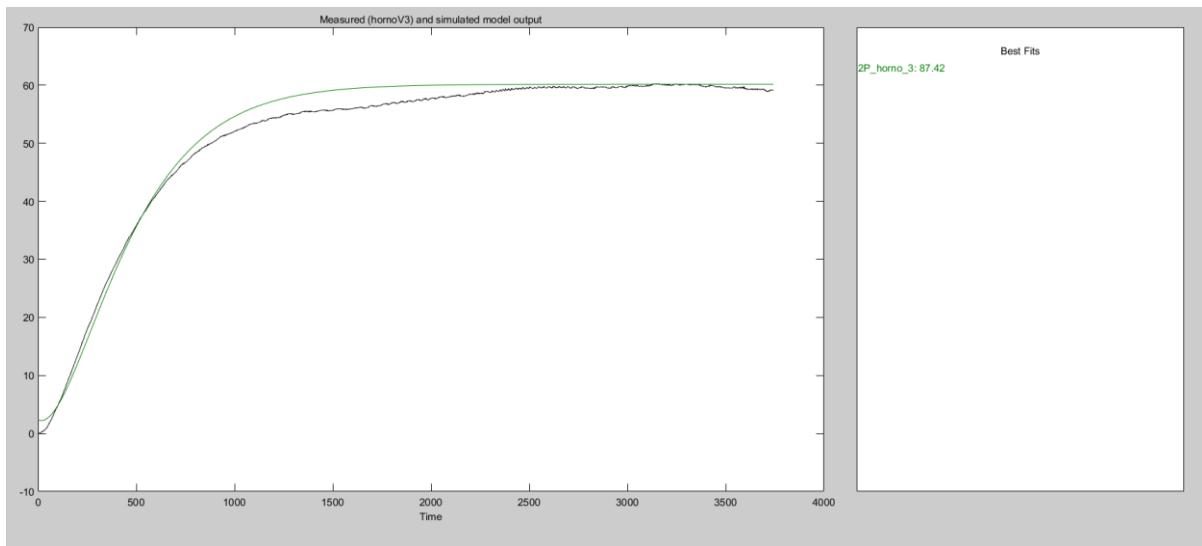


Figura 3.95. Verificación del modelo obtenido en el tercer escalón ascendente.

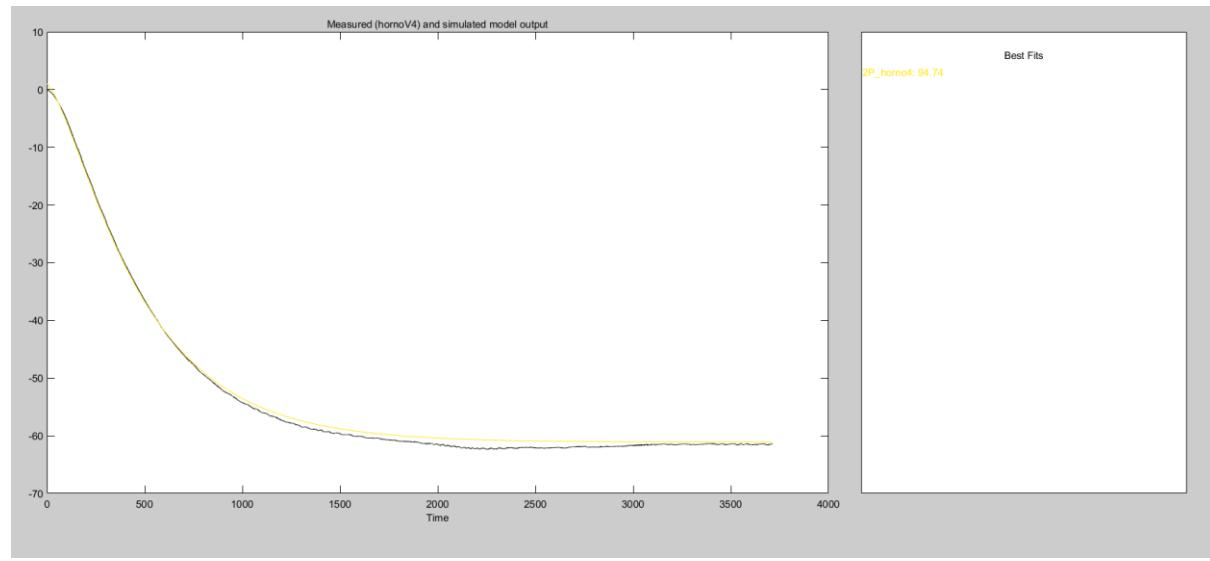
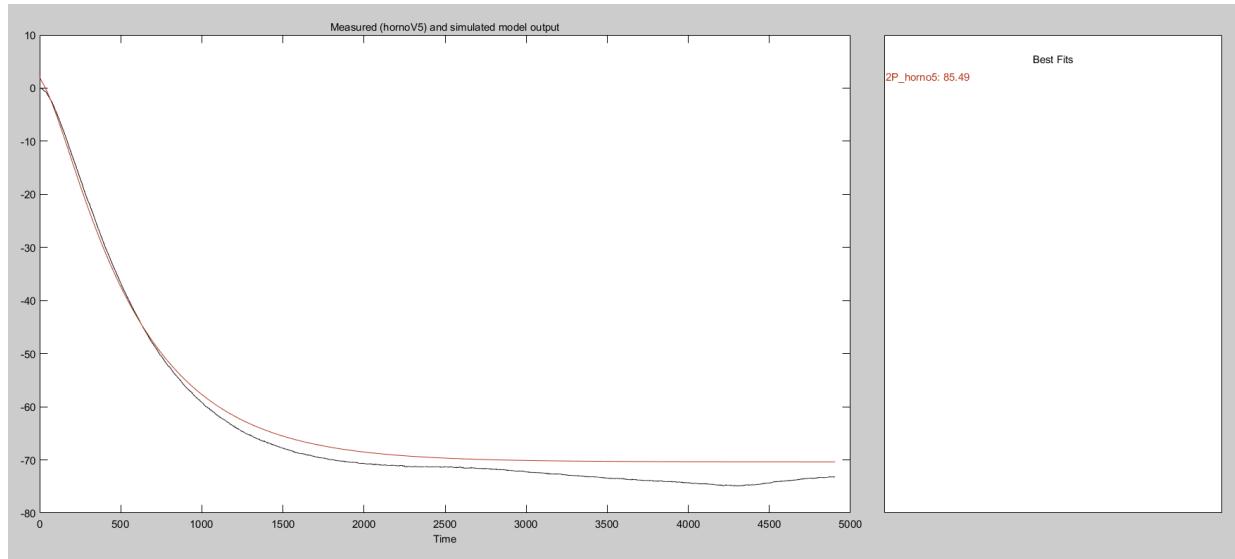


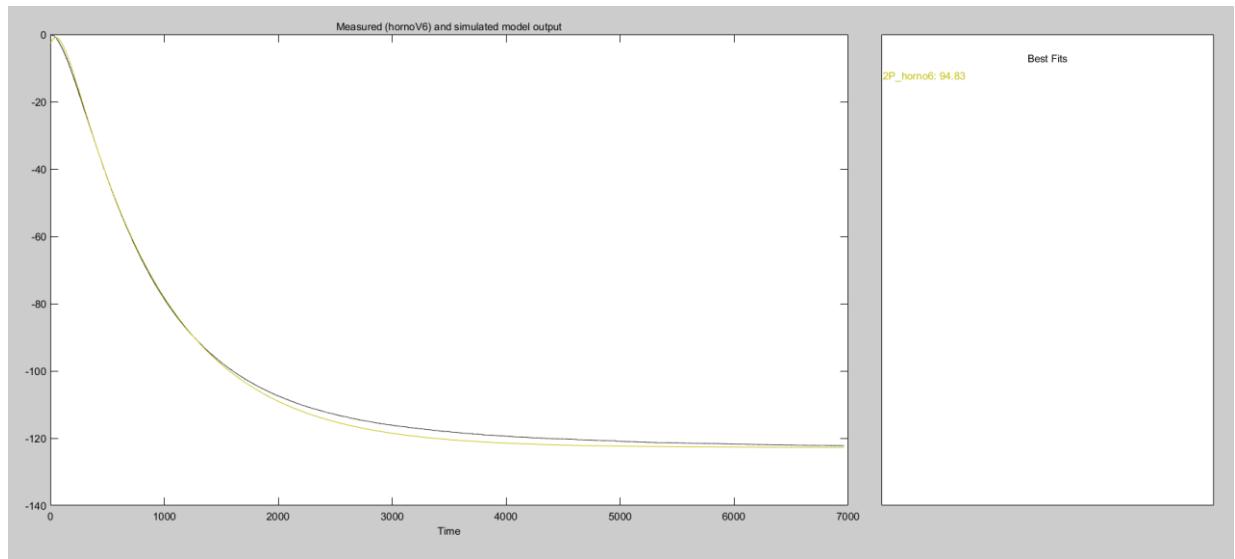
Figura 3.96. Verificación del modelo obtenido en el primer escalón descendente.

Finalmente, en la figura 3.98 se obtiene una similitud del 94.83% para el tercer escalón descendente, se da como válido el modelo.

Como se puede apreciar algunos modelos tienen más similitud que otros, no obstante, los posibles errores en la etapa de modelado serán mitigados con el diseño de un controlador PID (es una de las ventajas de los sistemas de control en lazo cerrado).



*Figura 3.97. Verificación del modelo obtenido en el segundo escalón descendente.*



*Figura 3.98. Verificación del modelo obtenido en el tercer escalón descendente.*

### 3.7 Diseño de la estrategia de control

#### 3.7.1 Diseño teórico

Una vez que se han obtenido los modelos y se han verificado, se va a diseñar la estrategia de control, se va a realizar un controlador conservador, es decir, se va a escoger el sistema más desfavorable (el que tenga la ganancia estática y constante de tiempo más grande). Por tanto, el modelo que se va a escoger será el del primer tramo ya que presenta la ganancia estática más grande. El modelo del tercer escalón descendente también tiene una ganancia grande, pero como se tiene forma de actuar sobre el enfriamiento solamente desconectar por complejo la alimentación a la carga esta será la manera más rápida posible de enfriar el horno.

Dada la función de transferencia de la planta (primer escalón) se realiza el diseño del controlador utilizando un PID en forma interactiva y utilizando el método landa:

$$P(s) = \frac{6.1826}{(452.62s + 1)(334.93s + 1)} e^{-10s} \quad (3.48)$$

Aproximando el retardo mediante el desarrollo de Taylor:

$$G(s) = \frac{6.1826(1 - 10s)}{(452.62s + 1)(334.93s + 1)} \quad (3.49)$$

Utilizando un controlador PID en forma interactiva:

$$L(s) = G(s)C(s) = \frac{6.1826(1 - 10s)}{(452.62s + 1)(334.93s + 1)} \frac{K(T_i s + 1)(T_d s + 1)}{T_i s} \quad (3.50)$$

Cancelamos el polo y el cero haciendo  $T_i=452.62$  y  $T_d=334.93$ :

$$L(s) = \frac{6.1826K(1 - 10s)}{452.62s} \quad (3.51)$$

La función en lazo cerrado es:

$$G(s)_{BC} = \frac{L(s)}{1 + L(s)} = \frac{\frac{6.1826K(1 - 10s)}{452.62s}}{1 + \frac{6.1826Kc(1 - 10s)}{452.62s}} \quad (3.52)$$

$$G(s)_{BC} = \frac{\frac{6.1826Kc(1 - 10s)}{452.62s}}{\frac{452.62s + 6.1826K - 6.1826K10s}{452.62s}} \quad (3.53)$$

$$G(s)_{BC} = \frac{6.1826K(1 - 10s)}{(452.62 - 6.1826K10)s + 6.1826K} \quad (3.54)$$

$$G(s)_{BC} = \frac{\frac{6.1826K(1 - 10s)}{6.1826K}}{\left(\frac{452.62 - 6.1826 * 10K}{6.1826K}\right)s + 1} \quad (3.55)$$

Utilizando una constante de tiempo de bucle cerrado como la media de las constantes de tiempo de los polos de la función de lazo abierto:

$$\lambda = \tau_{BC} = \frac{452.62 + 334.93}{2} = 394 \text{ seg} \quad (3.56)$$

$$\lambda = \left(\frac{452.62 - 6.1826 * 10K}{6.1826K}\right) \rightarrow 394 = \left(\frac{452.62 - 6.1826 * 10K}{6.1826K}\right) \quad (3.57)$$

Despejando  $K$  se obtiene:

$$K = \frac{T_i}{6.1826(\lambda + tr)} = \frac{452.62}{6.1826(394 + 10)} = 0.1812 \quad (3.58)$$

Se calcula los valores asociados para un controlador no interactivo pues en la programación del controlador digital se utiliza un controlador en su forma paralela:

$$K = K' \left( \frac{T_i' + Td'}{T_i'} \right) = 0.1812 \left( \frac{452.62 + 334.93}{452.62} \right) = 0.3153 \quad (3.59)$$

$$T_i = T_i' + Td' = 452.62 + 334.92 = 787.55 \quad (3.60)$$

$$T_d = \left( \frac{T_i' T d'}{T_i' + T d'} \right) = \left( \frac{452.62 * 334.92}{787.55 + 334.92} \right) = 192.49 \quad (3.61)$$

A continuación, se pasa estos parámetros a la forma paralela:

$$K'' = K = 0.3153 \quad (3.62)$$

$$K_i'' = \frac{K}{T_i} = \frac{0.3153}{787.55} = 0.0004 \quad (3.63)$$

$$K_d'' = K T_d = 0.3153 * 192.49 = 60.69 \quad (3.64)$$

### 3.7.2 Simulación de la estrategia de control

En la figura 3.99 se muestra el esquema de *Simulink* que se ha utilizado para simular el sistema en lazo cerrado con el controlador diseñado en el apartado anterior, se ha colocado la planta y el retardo de 10 segundos, a continuación, se ha modelado el actuador colocando los límites de 0 a 60 por ciento de PWM, y se ha añadido el bloque PID que se ha configurado en su forma paralela y se han introducido los valores obtenidos en el diseño teórico.

Se dieron dos escalones para ver cómo responde el controlador, el primer escalón se dio con una amplitud de 1 y el segundo con una amplitud de 150. En la figura 3.100 se muestra el resultado de la simulación para un escalón de amplitud 1, como se puede apreciar cumple con las especificaciones impuestas en el diseño teórico de 395 segundos de constante de tiempo y respuesta de primer orden.

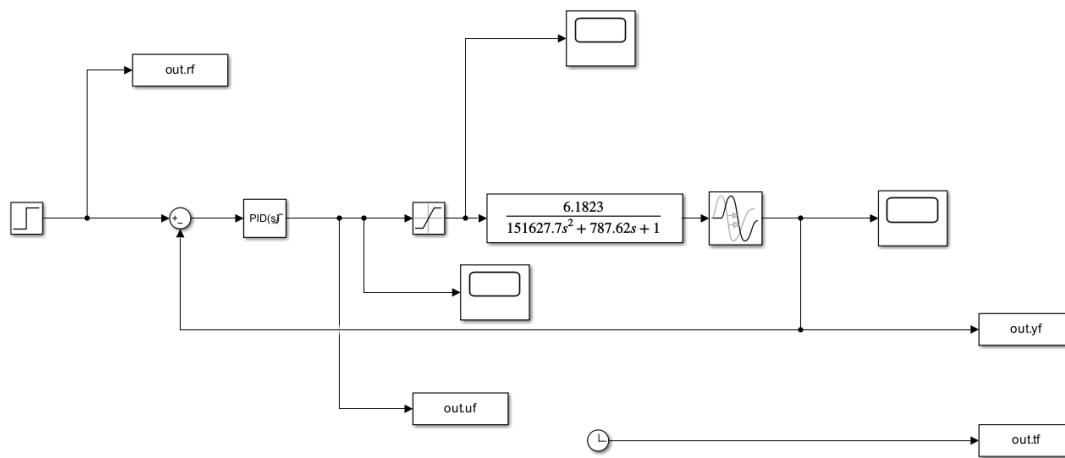


Figura 3.99. Esquema de *Simulink* para la simulación del controlador obtenido teóricamente.

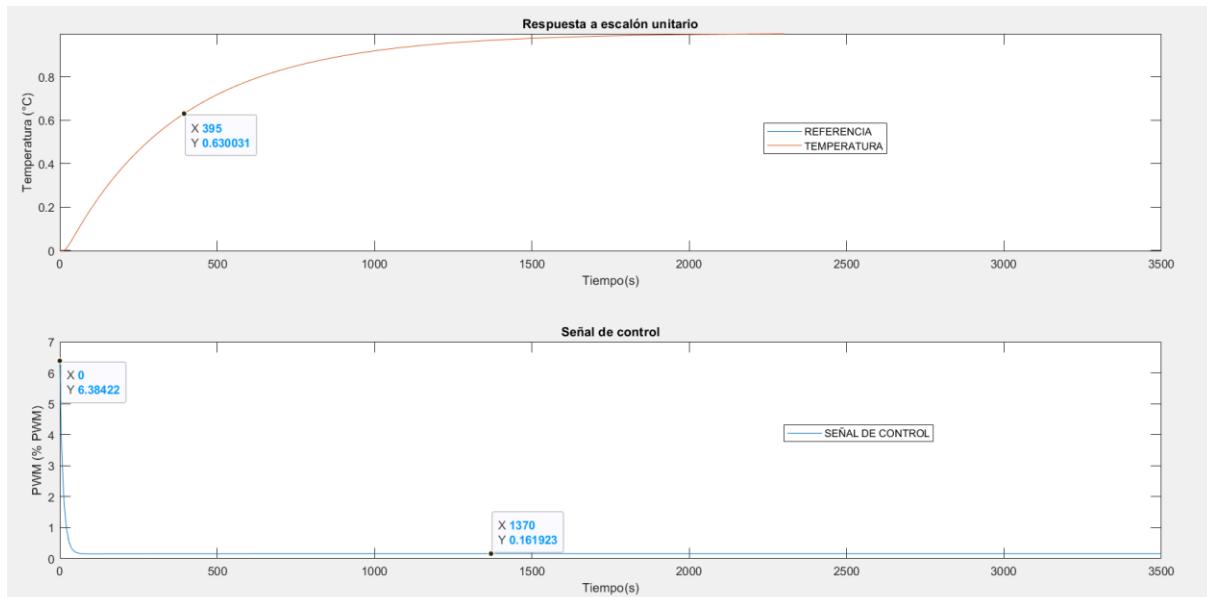


Figura 3.100. Respuesta del sistema en lazo cerrado ante escalón unitario.

En la figura 3.101 se muestra el resultado de la simulación para un escalón de 150 de amplitud, como se puede apreciar no cumple con las especificaciones impuestas en el diseño teórico, esto es debido a que el sistema es no lineal y como se ha utilizado un modelo fijo para un punto de operación, cuando se mueve a otro punto de operación la aproximación del modelo no es perfecta y por tanto esto lleva al no cumplimiento exacto de las especificaciones. Aunque no se cumple la especificación de la constante de tiempo esta no se aleja demasiado y toma un valor de 553 segundos, además la respuesta presenta una sobreoscilación. Por otra parte, se puede apreciar la señal de control es la típica respuesta de cancelación polo cero, tambien se observa la saturación de la señal del actuador del 60% de PWM.

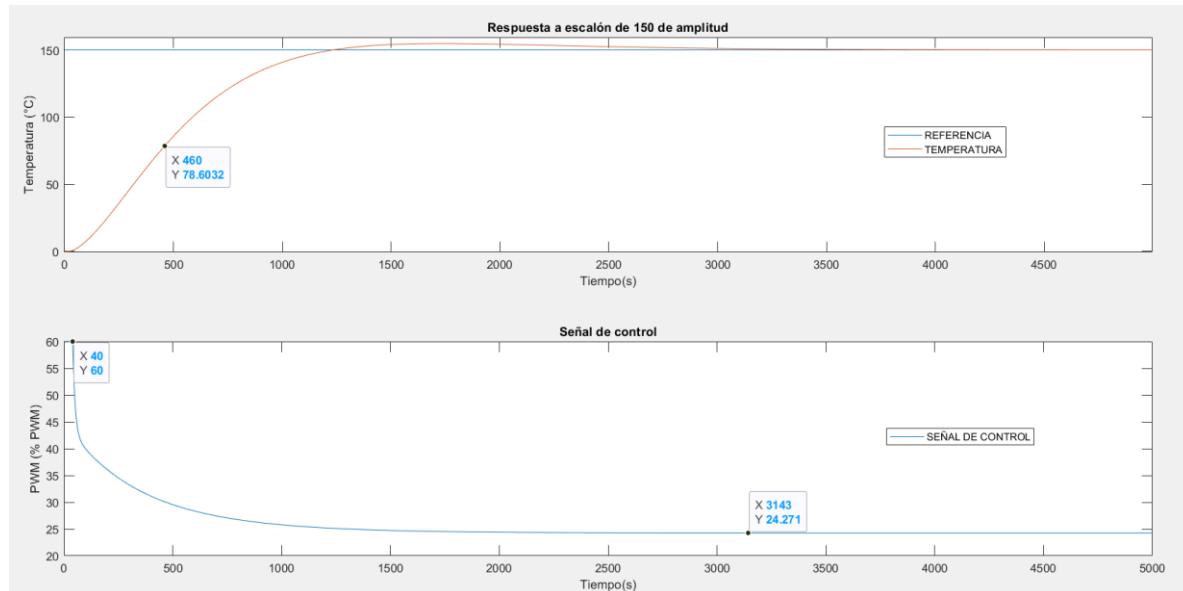


Figura 3.101. Respuesta del sistema en lazo cerrado ante un escalón de 150 de amplitud.

En la figura 3.102, se muestra el esquema de *Simulink* que se utilizó para hacer la simulación del seguimiento de referencias. Este esquema es diferente del utilizado para la simulación teórica, en este esquema se implementó el control PID por bloques (esquema paralelo) y se añadió el *antiwindup* (*back calculation*), además se discretizó la planta para ver el efecto del tiempo de muestreo en la aproximación de la planta. En el esquema se añadieron los valores del controlador calculados teóricamente y se añadió el tiempo de tracking de 389 segundos (la raíz cuadrada de la multiplicación  $T_i$  por  $T_d$ ).

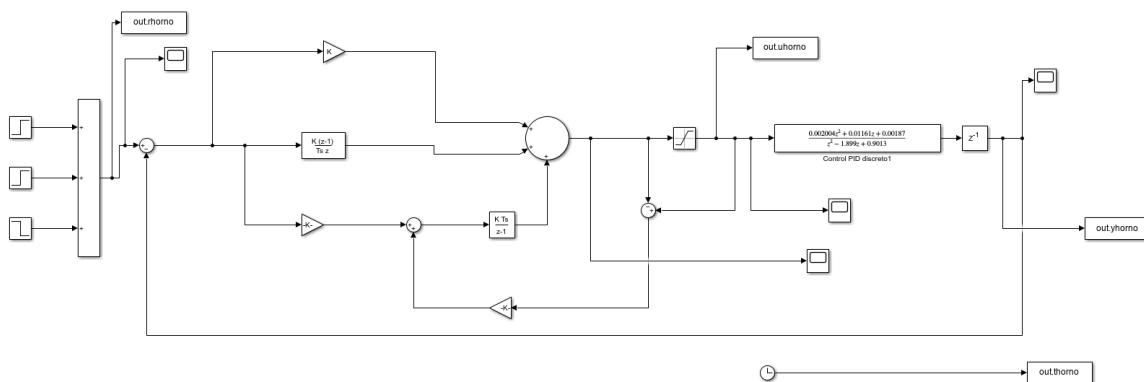


Figura 3.102. Esquema de Simulink para la simulación de seguimiento de referencias.

En la figura 3.103 se muestra los resultados de la simulación para el seguimiento de referencias (utilizando el modelo obtenido en el primer escalón y el controlador diseñado en el apartado teórico).

Se ha dado tres escalones, en el primer escalón de 150 de amplitud (temperatura de precalentamiento en el perfil RSS) se observa que sigue la referencia con una respuesta sobreamortiguada, pero no cumple la especificación de constante de tiempo de 395 segundos impuesta en el diseño teórico, esta toma el valor de 839 segundos para el 63% de la salida (94°C).

Luego se ha dado un escalón de 70 °C, es decir se pasa de 150 °C a 220 °C (temperatura pico en el perfil RSS), aquí se puede ver que sigue la referencia con una dinámica sobreamortiguada, el valor que toma la constante de tiempo es de 819 segundos al llegar al 63% de la salida (194°C).

Finalmente, se introduce un escalón descendente de amplitud 200 (escalón de enfriamiento, en donde se obtiene una respuesta sobreamortiguada y una constante de tiempo de 845.

Con respecto a las señales de control, vemos que la señal satura en el 60% de PWM impuesto como límite en el actuador y sigue la típica respuesta de cancelación polo cero, subiendo al principio y luego bajando.

Se puede apreciar que el modelo y controlador utilizado sigue las referencias con una respuesta sobreamortiguada y una constante de tiempo de 840 segundos, esto es debido a que no se ha dado escalones unitarios sino de mayor amplitud. No obstante, se ha sido bastante conservador en las especificaciones impuestas en el control, ya que si se utiliza un controlador más agresivo la respuesta sobreoscilaría demasiado en el sistema real y se tendría que esperar a que se enfriase el sistema de forma natural (ya que la forma más rápida es desconectar por completo el MOSFET).

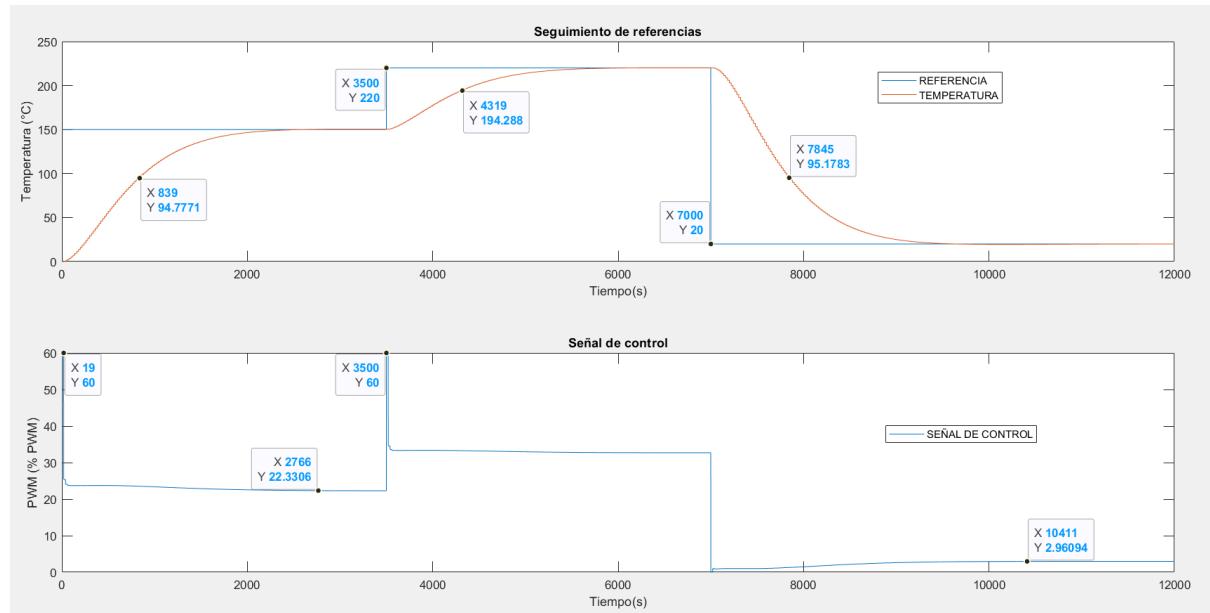


Figura 3.103. Respuesta del sistema en lazo cerrado ante seguimiento de referencias.



## 4 Resultados y discusión

En este apartado se discutirán los resultados obtenidos para este trabajo final de grado, primero se discutirán los resultados de los subsistemas diseñados y el correcto funcionamiento de estos, luego se mostrará el resultado final de la interfaz gráfica para la adquisición de los datos de temperatura, modelar el sistema y enviar referencias. Finalmente se probará el controlador diseñado teóricamente en el sistema real y verificar la transferencia sin saltos entre manual y automático.

### 4.1 Sistema del horno

Una vez diseñado teóricamente las placas de la electrónica de potencia y el acondicionador de señal, se enviaron los archivos *gerber* para que se fabricaran los PCB, se soldaron todos los componentes electrónicos, se comprobaron con el osciloscopio y el multímetro que funcionaban correctamente y daban las tensiones esperadas una vez ajustadas las placas. Luego se realizó todas las interconexiones de los subsistemas al Arduino y se comprobó que la programación realizada en el Arduino funcionaba correctamente.

#### 4.1.1 Prueba de funcionamiento de las placas diseñadas

Se comprobó primero que la placa de potencia funciona y trocea la señal alterna, como se muestra en la figura 4.1, dando más o menos potencia dependiendo de la cantidad de PWM que se aplique a la carga.

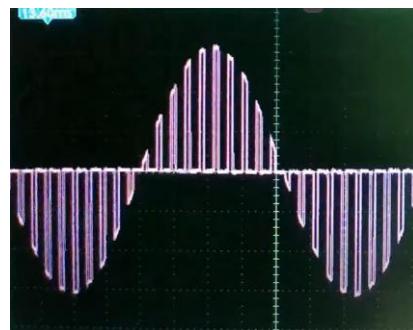


Figura 4.1. Señal alterna troceada por una señal PWM medida con osciloscopio.

Para visualizar este efecto en un sistema real se comprobó primero en un sistema equivalente, es decir, con una carga resistiva, en la figura 4.2 se muestra que se utilizó una bombilla para ver el efecto de la variación de PWM en la carga. Se puede ingresar al siguiente enlace de *YouTube* para observar dicho efecto: [www.youtube.com/shorts/2KBYDYxNHQ4](https://www.youtube.com/shorts/2KBYDYxNHQ4).

Una vez comprobado que la placa de potencia funciona correctamente, se realizó lo mismo con el acondicionador de señal (una vez ajustada la recta de transferencia del acondicionador de señal), en la figura 4.3 se muestra la tensión de salida de 232 milivoltios que equivale a 25.69 °C (temperatura en el laboratorio).

Una cosa para tener en cuenta en el acondicionador de señal es que no se calibro en base a un sistema de referencia profesional, solo se utilizó una resistencia variable que simulaba las variaciones de la PT100 y se comprobó las tensiones de salida obtenidas que se ajustaban a los cálculos teóricos en los valores extremos. Los posibles errores en la medición de la temperatura no se han podido cuantificar de una forma profesional.

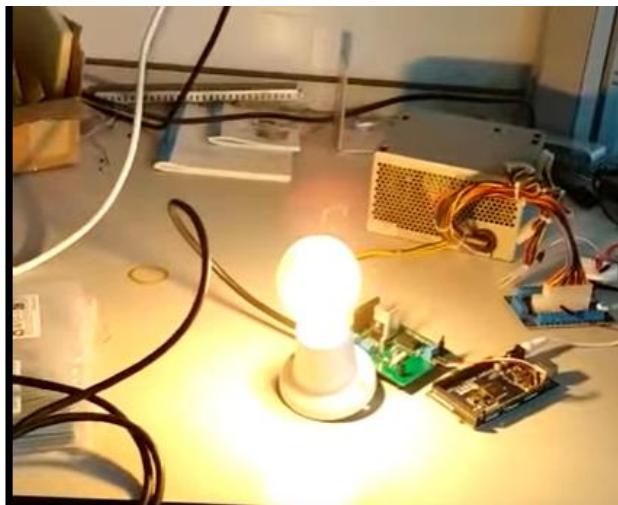


Figura 4.2. Prueba de funcionamiento del PWM en una bombilla.

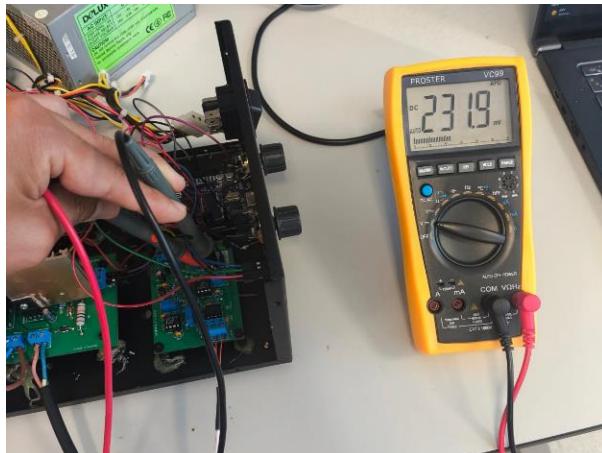


Figura 4.3. Prueba de funcionamiento del acondicionador de señal.

#### 4.1.2 Prueba de funcionamiento de los periféricos

En la figura 4.4 se muestra el funcionamiento de la pantalla OLED que mostrará el valor de la temperatura, el porcentaje de PWM, si esta encendido o apagado el horno, y los diodos LED que indicarán si se ha superado el umbral de temperatura de 30 °C.

Como no se puede hacer una foto del ventilador funcionando se procedió a realizar un video para mostrar que para un porcentaje de PWM superior al 5%, el ventilador se activa, tambien se puede comprobar en el video cómo funciona el sistema mediante la interfaz gráfica de Matlab enviando escalones (en modo manual) al horno. El video se puede observar en la plataforma YouTube con el siguiente enlace: <https://youtu.be/cOOut1HzYnI>.

De esto modo se comprueba que los periféricos, las placas diseñadas, y el horno responde a los comandos enviados por la interfaz gráfica y la programación del Arduino funciona correctamente.



Figura 4.4. Prueba de funcionamiento de los Leds y la pantalla OLED.

## 4.2 Interfaz gráfica

### 4.2.1 Visualización y recogida de datos

Para visualizar los datos obtenidos de la temperatura procedentes del sensor de temperatura, poder ingresar escalones para modelar el sistema (modo manual) o que siga referencias que envíemos al sistema (modo automático). Se programó una interfaz gráfica, el resultado se aprecia en la figura 4.5.

Se va a describir cómo funciona y qué hace cada botón, una vez ejecutado el script que se adjunta en el anexo de la memoria, aparece la interfaz gráfica. A continuación, una vez conectado el Arduino al ordenador por el puerto USB (sin tener abierto el IDE de Arduino) se conecta al puerto por el cual se va a realizar la comunicación, se abre el desplegable y sale el puerto disponible automáticamente, luego se da al botón iniciar para crear la conexión serial entre el Matlab y el Arduino.

Una vez que se ha pulsado iniciar empezaran a leerse los datos provenientes del sensor. A continuación, si se quiere modelar el sistema y darle escalones al horno se pulsa el botón manual, luego se escribe el valor de PWM que se desea introducir, (está limitado de 0 a 60) y se pulsa enviar.

Si se desea iniciar el sistema en modo automático y empezar a seguir referencias, primero se debe pulsar a automático, luego se introduce las referencias en el cuadro superior a enviar (caja de texto).

Si por el contrario se desea empezar en manual y cambiar a automático, primero se inicia el sistema en manual, se da el escalón y luego se debe de esperar a que el sistema llegue al régimen permanente y, a continuación, cambiar a modo automático.



Figura 4.5. Interfaz gráfica de Matlab para el Arduino

Una vez realizado el ensayo y se desea guardar los datos obtenidos, primero se da al botón detener, para finalizar el ensayo y cerrar la comunicación serial, a continuación, se pulsa al botón salvar y se abren dos ventanas emergentes, como se aprecia en la figura 4.6. La primera ventana será la gráfica en formato de Matlab que se puede guardar dando al icono de guardar, detrás de esta ventana se encuentra la ventana para guardar los datos en formato txt.

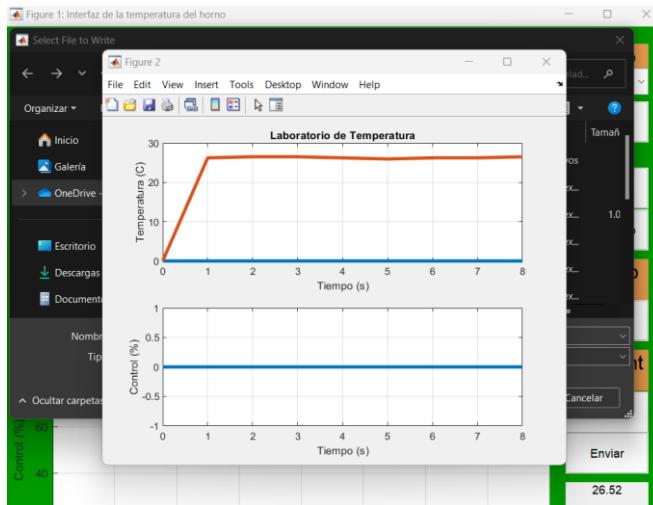


Figura 4.6. Ventanas para guardar los datos.

Para cerrar la interfaz, un vez guardado los datos, se damos al botón cerrar, si se quiere realizar otro ensayo se tienen que repetir los pasos comentados anteriormente.

Un aspecto para tener en cuenta en la interfaz es que cada vez que se abre la comunicación serial se debe cerrar correctamente dando al botón detener sino puede que no se cierre la comunicación y se tenga que forzar la parada pulsando stop en la ventana de ejecución en Matlab. Otro aspecto a tener en cuenta de la interfaz es para ensayos largos, aquí la interfaz gráfica los datos con retardo, por ejemplo, se da un escalón al sistema este lo reconoce instantáneamente, pero lo gráfica en la interfaz segundos después, esto seguramente se deba a los tiempos de ejecución que se estén propagando en el script y en el buffer serial estén datos acumulados no procesados por Matlab.

## 4.3 Sistema de control

En este apartado se va a mostrar el resultado del controlador diseñado teóricamente en el sistema real y se va a comprobar la transferencia sin saltos cuando se parte de modo manual y luego se cambia a automático.

### 4.3.1 Prueba de control manual a automático

En la figura 4.7 se muestra el resultado de la transferencia sin saltos entre el modo manual y automático, se parte del modo manual (este modo se utilizó para modelar el sistema dándole escalones), luego se da un escalón del 20% de PWM, a continuación, se esperó a que se estabilice el sistema, es decir, que llegue al régimen permanente ( $142.56^{\circ}\text{C}$ ), después se commuta al modo automático, como se puede observar en la señal de control no se produce saltos bruscos, se mantiene el valor de la señal puesto que al realizar el cambio se le indica al integrador del controlador el valor inicial que debe tener, y la referencia está tomando el valor de la salida en el instante del cambio el error es cero y se mantendrá si no se cambia la referencia en modo automático.

Una vez comprobado que se mantiene estable la señal de control se dio una referencia de  $160^{\circ}\text{C}$ , aquí se produce lo comentado anteriormente en el anterior apartado, como se lleva mucho tiempo de recogida de datos la interfaz los gráfica con un retardo, como se puede apreciar el escalón lo gráfica en el momento exacto cuando le damos a enviar en el segundo 4258 pero la señal de control tarda en graficarse (en la pantalla OLED sí que se refleja el cambio en la señal de control al darle el escalón, transcurrido el tiempo de muestreo, el problema proviene de la interfaz).

Teniendo esto en cuenta se tomó como referencia para calcular la constante de tiempo del sistema la señal de control que varía en el segundo 4320, de esta forma la constante de tiempo es de 657 segundos transcurrido el 63% de la salida ( $153^{\circ}\text{C}$ ), aunque diverge de los cálculos teóricos, estamos ante un sistema fuertemente no lineal, por tanto, no se llegará a cumplir los requerimientos totalmente, no obstante, si se cumple la dinámica impuesta en el diseño teórico, la respuesta es sobreamortiguada. Por otra parte, en el escalón de bajada la constante de tiempo es de 378 segundos y dinámica de sobreamortiguada, cumpliéndose con la especificación.

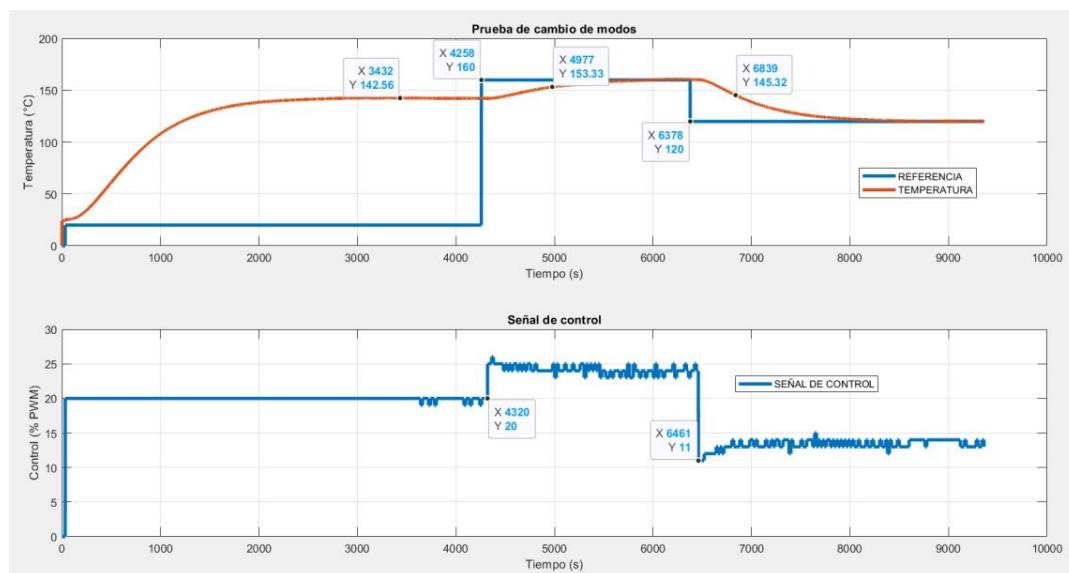


Figura 4.7. Cambio de modo manual a automático (Repetida de la figura 1.3 por conveniencia).

### 4.3.2 Prueba de seguimiento de referencias

En la figura 4.8 se muestra el resultado del sistema del horno siguiendo una referencia RSS (sin incluir la rampa) mediante escalones, es decir, se dio una referencia de 150°C que es la temperatura de precalentamiento (*soak*), luego una referencia de 220°C que es la temperatura pico (*spike*) y finalmente el escalón de enfriamiento hasta 30°C.

Tomando como referencia la señal de control para calcular la constante de tiempo, se obtiene para el primer escalón una constante de tiempo de 575 segundos y una respuesta con una sobreoscilación. Para el segundo escalón se obtuvo una constante de tiempo de 622 segundos y una respuesta sobreamortiguada. En el último escalón se obtuvo una constante de tiempo de 835 segundos y dinámica de primer orden.

Con respecto a los valores obtenidos en la simulación, las constantes de tiempo son inferiores en los escalones de subida y no difiere mucho en el escalón de bajada. Como se dijo anteriormente el sistema es no lineal y se puede apreciar que las especificaciones teóricas no se han cumplido debido a que se utilizó un modelo fijo para un punto de operación y al cambiar a otro punto de operación la aproximación del modelo no es perfecta y esto lleva al no cumplimiento exacto de las especificaciones.

En cuanto a la señal de control se puede observar que tiene la forma esperada de una cancelación polo cero, primero sube y luego va bajando con el tiempo, en el escalón de enfriamiento se desconecta por completo la señal de control ya que es la forma más rápida de enfriar el sistema.

Como se puede apreciar el controlador implementado consigue seguir las referencias, pero no con la suficiente rapidez que se necesita para la soldadura por reflujo, si se introduce un controlador más agresivo las sobreoscilaciones serían mayores y el tiempo para llegar al régimen permanente aumentaría. En cuanto al enfriamiento ese sería la máxima respuesta, se podría abrir la puerta del horno para conseguir un enfriamiento más rápido, pero no se tendría control sobre esta acción.

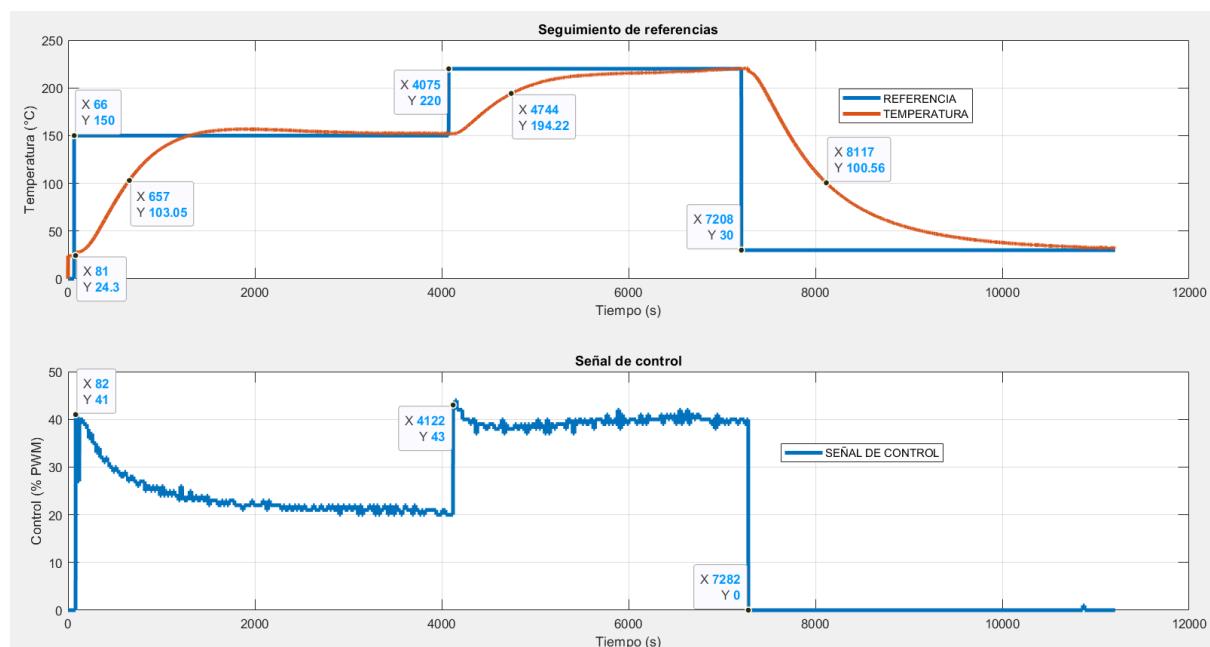


Figura 4.8. Prueba de seguimiento de referencias (Repetida de la figura 1.4 por conveniencia).

## 5 Conclusiones y trabajos futuros

Este apartado se exponen las conclusiones a las que se ha llegado con la realización de este proyecto. También, se indican una serie de mejoras que se podrían llevar a cabo en trabajos futuros.

### 5.1 Conclusiones

El objetivo de este proyecto era realizar el control de un horno de bajo coste para soldadura por reflujo. Para ello se modificó el horno, primero se hizo un estudio con una cámara termográfica para colocar la sonda de temperatura PT100 en el centro del horno y obtener unas medidas fiables de temperatura, luego se colocó la resistencia inferior a la parte superior para que todo el calor salga de una sola dirección, se quitó el termostato que venía como elemento de control y se hizo las conexiones eléctricas del horno modificado.

A continuación, se diseñó la electrónica necesaria para obtener los datos de temperatura provenientes del sensor PT100, se realizaron los cálculos teóricos, la simulación del circuito, el diseño del PCB y su implementación física, luego una vez soldados los componentes se ajustó la recta de transferencia del acondicionador de señal mediante una resistencia variable.

También se diseñó la electrónica de potencia para controlar la resistencia del horno, se empleó un MOSFET de potencia para obtener tiempos de conmutación más rápidos que si se hubiese utilizado por ejemplo un relé, se realizaron los cálculos teóricos, la simulación del circuito, el diseño del PCB y se comprobó la funcionalidad del circuito primero con una bombilla para después implementarlo al horno.

Una vez comprobadas ambas placas, se montó todo dentro de una caja metálica reciclada de un aparato estropeado, se utilizó una fuente de alimentación proveniente de un ordenador reciclado para alimentar las placas, se añadieron periféricos como un ventilador reciclado para enfriar los disipadores de la placa de potencia, una pantalla OLED para visualizar la información en tiempo real del sistema y leds para indicar que se ha superado un umbral de temperatura para no tocar el horno caliente.

Se utilizó una placa Arduino para controlar los subsistemas, obtener los datos del sensor y comunicarse por el puerto serial con Matlab en donde se realizó una interfaz gráfica para la recogida de datos y modelar el sistema para posteriormente diseñar un controlador, se programó la interfaz gráfica en Matlab, se programó el algoritmo de control en el Arduino y los procedimientos para controlar los subsistemas, luego se comprobó el correcto funcionamiento de la interfaz y el código escrito en el microcontrolador una vez realizado las conexiones de las placas y periféricos.

Después se modeló el sistema aplicando escalones al horno, se hizo tres escalones de subida y tres de bajada, se guardaron los datos y utilizando la herramienta *System Identification Toolbox* de Matlab se obtuvieron las funciones de transferencia para cada escalón, luego se repitió un nuevo ensayo para validar dichos modelos. Una vez validados los modelos se diseñó un controlador robusto con el modelo más desfavorable, se simuló en *Simulink* y se comprobó que para escalones unitarios cumple las especificaciones, pero para escalones de mayor amplitud la constante de tiempo es superior a la especificada en el diseño teórico. Una vez realizada la simulación se probó el controlador en el sistema real, introduciendo los valores teóricos obtenidos del controlador en el algoritmo de control en el Arduino, se dio unos escalones de temperatura con valores típicos de la soldadura de reflujo y se analizó los resultados.

En la primera parte de la modificación del horno se cumplió los objetivos, no obstante, se podría haber incluido un mecanismo de apertura de la puerta para tener control en el enfriamiento. En el diseño de la electronica, la placa de potencia conseguía regular la potencia a través de la señal de PWM este subobjetivo también se da por cumplido. En cuanto a la placa del acondicionador de señal se consiguió obtener la recta de transferencia mediante una resistencia variable pero no se pudo obtener una calibración precisa de la placa y realizar una cuantificación de los errores en la medición de temperatura al no disponer de un sistema profesional de medida y obtener una referencia de temperatura fiable.

Los periféricos que se incluyeron hicieron su función específica y funcionaron correctamente según su programación, aunque no era un subobjetivo incluirlos se optó por ello, ya que proporcionaba información visual extra, enfriar la placa de potencia y se consiguió aprender a utilizar y programar dichos dispositivos.

En cuanto a la interfaz gráfica, se observó que para ensayos de larga duración los datos en la señal de control se graficaban después de introducirlos en el sistema, aunque por la pantalla OLED si se refleja la acción de control transcurrido el tiempo de muestreo. Este problema no se consiguió solucionar, pero se tuvo en cuenta a la hora de calcular las constantes de tiempo tomando como inicio de tiempo la señal de control.

La programación realizada en el Arduino funciona correctamente ejecutando el algoritmo de control cada tiempo de muestreo, enviando los datos a la interfaz gráfica cada segundo, controlando la placa de potencia, realizando la lectura de la temperatura proveniente del sensor, la recepción de los comandos desde la interfaz gráfica y ejecutándolos correctamente.

En cuanto al seguimiento de referencias, el sistema de control diseñado sigue las referencias, pero las especificaciones impuestas en el diseño teórico no se cumplen. Esto es debido a que el sistema es no lineal y se ha utilizado un modelo fijo para un punto de operación, al cambiar de punto de operación la aproximación del modelo no es perfecta y por tanto esto lleva al no cumplimiento exacto de las especificaciones. No obstante, se puede apreciar que el sistema tiene una dinámica muy lenta y para sistemas de soldadura de reflujo esto no es admisible, en donde las especificaciones de temperatura y sobre todo el tiempo de exposición a dichas temperaturas son críticas. Por tanto, este sistema no sería apto para la aplicación de la soldadura por reflujo, sin embargo, este sistema resulta interesante como laboratorio de pruebas con fines educativos para diseñar estrategias de control no lineales con dinámicas muy lentas.

## 5.2 Trabajos futuros

A continuación, se comentan los posibles trabajos que se podrían realizar como continuación de este proyecto y además se proponen posibles mejoras que se podrían llevar a cabo en trabajos futuros.

- Mejora de la programación de la interfaz gráfica. Como ya se ha comentado, no sé consiguió solucionar el problema a la hora de graficar datos en la interfaz con tiempos de ensayo largos.
- Calibrar el acondicionador de señal con un sistema profesional.
- Inclusión de un mecanismo para abrir la puerta del horno mediante un motor eléctrico y poder controlar el enfriamiento según el ángulo de apertura de la puerta y de un módulo wifi para la transferencia de datos en red y poder realizar ensayos remotamente.

- Realización de técnicas de control no lineal como la planificación de ganancias para mejorar el control en los distintos rangos de temperatura.



## 6 Bibliografía

- [1] «Evolución histórica de los PCB – Publys». Accedido: 22 de abril de 2024. [En línea]. Disponible en: <https://publys.cl/2019/03/28/evolucion-historica-de-los-pcb/>
- [2] «Soldadura por reflujo en PCB - Tecnología MOKO». Accedido: 21 de abril de 2024. [En línea]. Disponible en: <https://www.mokotechnology.com/es/reflow-soldering-on-pcb/>
- [3] «Tecnología de montaje superficial - Wikipedia, la enciclopedia libre». Accedido: 22 de abril de 2024. [En línea]. Disponible en: [https://es.wikipedia.org/wiki/Tecnolog%C3%ADa\\_de\\_montaje\\_superficial](https://es.wikipedia.org/wiki/Tecnolog%C3%ADa_de_montaje_superficial)
- [4] «Reflow soldering - Wikipedia». Accedido: 22 de abril de 2024. [En línea]. Disponible en: [https://en.wikipedia.org/wiki/Reflow\\_soldering](https://en.wikipedia.org/wiki/Reflow_soldering)
- [5] «Reflow Oven Temperature Profiler - BTU International». Accedido: 21 de abril de 2024. [En línea]. Disponible en: <https://www.btu.com/reflow-ovens/solder-reflow-oven-profiling/>
- [6] «Fundamentos Esenciales de la Tecnología de Montaje en Superficie (SMT): Un Análisis Introductorio | LinkedIn». Accedido: 21 de abril de 2024. [En línea]. Disponible en: <https://www.linkedin.com/pulse/fundamentos-esenciales-de-la-tecnolog%C3%ADa-de-montaje-en-smt-jorge-quijano/?trackingId=KclO%2BAGmSbOLZ%2BGoPt5nlw%3D%3D>
- [7] «Soldadura en Pasta en SMT: De los Componentes a los Parámetros, una Exploración Integral | LinkedIn». Accedido: 22 de abril de 2024. [En línea]. Disponible en: <https://www.linkedin.com/pulse/soldadura-en-pasta-smt-de-los-componentes-par%C3%A1metros-una-quijano/?trackingId=fWL3aciPRwaxub6nyg8Zqw%3D%3D>
- [8] «FP2636 (Frameless Version) User Instructions Top ring Height adjustment handle», Accedido: 8 de mayo de 2024. [En línea]. Disponible en: <https://www.neodenpnp.com/Content/upload/2019436761/PDF/User-manual-FP2636-stencil-printer.pdf>
- [9] «El Proceso de Pick and Place en SMT: Claves Fundamentales | LinkedIn». Accedido: 4 de mayo de 2024. [En línea]. Disponible en: <https://www.linkedin.com/pulse/el-proceso-de-pick-place-en-smt-claves-fundamentales-jorge-quijano/?trackingId=k4KzrszcSHqhVZlhEPLglw%3D%3D>
- [10] «NeoDen10 High Speed Pick and Place Machine User Manual», Accedido: 6 de mayo de 2024. [En línea]. Disponible en: <https://www.neodenpnp.com/Content/upload/2019436761/PDF/User%20Manual-NeoDen10%20PNP%20machine.pdf>
- [11] «Introducción al Reflujo de Soldadura en SMT: Conceptos y Primeras Impresiones | LinkedIn». Accedido: 6 de mayo de 2024. [En línea]. Disponible en: <https://www.linkedin.com/pulse/introducci%C3%B3n-al-reflujo-de-soldadura-en-smt-conceptos-jorge-quijano/?trackingId=aJuFS1iGTMux0Cx1QVN5Pg%3D%3D>
- [12] «SMT-Reflow Profile Difference between RTS & RSS Reflow Profile Make a Best Touch to the top Yes Believe», Accedido: 6 de mayo de 2024. [En línea]. Disponible en: <https://ybtechsolution.com/wp-content/uploads/2019/01/Reflow-Profile-RTS-RSS.pdf>

- [13] M. H. Rashid, *Electrónica de potencia [Recurso electrónico]* / Muhammad H. Rashid., 4ed. Pearson, 2015.
- [14] «¿Qué es un Resistor Eléctrico? - Electrónica Online». Accedido: 17 de mayo de 2024. [En línea]. Disponible en: <https://electronicaonline.net/componentes-electronicos/resistor/que-es-un-resistor/>
- [15] «Condensador Eléctrico: ¿Qué es y Cómo Funciona? - Electrónica Online». Accedido: 17 de mayo de 2024. [En línea]. Disponible en: <https://electronicaonline.net/componentes-electronicos/condensador/que-es-un-condensador/>
- [16] «Tipos de Condensadores: Clasificación y sus Aplicaciones». Accedido: 18 de mayo de 2024. [En línea]. Disponible en: [https://electronicaonline.net/componentes-electronicos/condensador/tipos-de-condensadores/#%C2%BFQue\\_es\\_un\\_Condensador?](https://electronicaonline.net/componentes-electronicos/condensador/tipos-de-condensadores/#%C2%BFQue_es_un_Condensador?)
- [17] «Tipos de Diodos y sus Aplicaciones». Accedido: 18 de mayo de 2024. [En línea]. Disponible en: <https://electronicaonline.net/componentes-electronicos/diodo/tipos-de-diodos/>
- [18] «TEORÍA DE LOS DIODOS». Accedido: 18 de mayo de 2024. [En línea]. Disponible en: [https://www.tecnologiapedagogia.net/2009/10/cartilla-todo-sobre-los-diodos\\_11.html](https://www.tecnologiapedagogia.net/2009/10/cartilla-todo-sobre-los-diodos_11.html)
- [19] J. María y D. Moyano, «Tema VI: Referencias de tensión y reguladores de tensión», Accedido: 18 de mayo de 2024. [En línea]. Disponible en: [https://www.ctr.unican.es/asignaturas/instrumentacion\\_5\\_it/iec\\_6.pdf](https://www.ctr.unican.es/asignaturas/instrumentacion_5_it/iec_6.pdf)
- [20] «STmicroelectronics, "TL1431 Datasheet"», 2017. Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://n9.cl/b2gv9>
- [21] L. Technology Corporation, «Linear Technology, "LT1490A/LT1491A Datasheet"», Accedido: 19 de mayo de 2024. [En línea]. Disponible en: <http://www.linear.com/leadfree/>
- [22] «Amplificador de instrumentación - Wikipedia, la enciclopedia libre». Accedido: 19 de mayo de 2024. [En línea]. Disponible en: [https://es.wikipedia.org/wiki/Amplificador\\_de\\_instrumentaci%C3%B3n](https://es.wikipedia.org/wiki/Amplificador_de_instrumentaci%C3%B3n)
- [23] «Texas instruments, "INA128 Datasheet"», 2022, Accedido: 8 de mayo de 2024. [En línea]. Disponible en: <https://n9.cl/9gonp>
- [24] «Optoacoplador - Wikipedia, la enciclopedia libre». Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://es.wikipedia.org/wiki/Optoacoplador>
- [25] «Vishay, " CNY65 Optocoupler Datasheet"», Accedido: 16 de mayo de 2024. [En línea]. Disponible en: [www.vishay.com/doc?91000](http://www.vishay.com/doc?91000)
- [26] M. Appel y A. M. Kruck, «Optocouplers Guidelines for Reading an Optocoupler Datasheet», Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://www.vishay.com/docs/84256/useoptocouplerdatasheet.pdf>
- [27] «Microchip, "AN954 CAPACITIVE TRANSFORMERLESS POWER SUPPLY"», Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://ww1.microchip.com/downloads/en/appnotes/00954a.pdf>

- [28] «Qué es Arduino, cómo funciona y qué puedes hacer con uno». Accedido: 19 de mayo de 2024. [En línea]. Disponible en: <https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno>
- [29] K. J. Aström, T. Hägglund, S. Dormido, y J. L. Guzmán Sánchez, *Control PID avanzado / Karl J. Aström, Tore Hägglund ; traducción y revisión técnica, Sebastián Dormido Bencomo, José Luis Guzmán Sánchez*. Madrid: Pearson-Prentice Hall, 2009.
- [30] «MATLAB - Wikipedia, la enciclopedia libre». Accedido: 20 de mayo de 2024. [En línea]. Disponible en: <https://es.wikipedia.org/wiki/MATLAB>
- [31] «Simulink - Wikipedia, la enciclopedia libre». Accedido: 20 de mayo de 2024. [En línea]. Disponible en: <https://es.wikipedia.org/wiki/Simulink>
- [32] «GUI de MATLAB - MATLAB & Simulink». Accedido: 20 de mayo de 2024. [En línea]. Disponible en: <https://es.mathworks.com/discovery/matlab-gui.html>
- [33] «System Identification Toolbox - MATLAB». Accedido: 19 de mayo de 2024. [En línea]. Disponible en: <https://es.mathworks.com/products/sysid.html>
- [34] «Using the Arduino Software (IDE) | Arduino Documentation». Accedido: 20 de mayo de 2024. [En línea]. Disponible en: <https://docs.arduino.cc/learn/startng-guide/the-arduino-software-ide/>
- [35] «Hornos - Orbegozo Electrodomésticos». Accedido: 7 de mayo de 2024. [En línea]. Disponible en: <https://orbegozo.com/cat-productos/cocina/hornos/>
- [36] R. Pallás Areny, *Sensores y acondicionadores de señal / Ramón Pallás Areny*, 4<sup>a</sup> ed. corr. Barcelona: Marcombo, 2003.
- [37] «Platinum Resistance Thermometer (PRT) Selection Guide», Accedido: 8 de mayo de 2024. [En línea]. Disponible en: <https://docs.rs-online.com/512c/0900766b815e76a5.pdf>
- [38] «PT100 TEMPERATURE / RESISTANCE TABLE». Accedido: 16 de mayo de 2024. [En línea]. Disponible en: [https://ietlabs.com/pdf/Datasheets/Pt100Temperature\\_Resistance4digits.pdf](https://ietlabs.com/pdf/Datasheets/Pt100Temperature_Resistance4digits.pdf)
- [39] J. G. Proakis, J. G. Proakis, y D. G. Manolakis, *Tratamiento digital de señales [Recurso electrónico] / John G. Proakis, Dimitris G. Manolakis.*, 4<sup>a</sup> ed.. Madrid: Pearson Educación, 2007.
- [40] «Microchip, "ATmega640-1280-1281-2560-2561-Datasheet"», Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://n9.cl/7lihh>
- [41] «Vishay, "PB3006 Bridge Rectifier Datasheet"», Accedido: 15 de mayo de 2024. [En línea]. Disponible en: [www.vishay.com/doc?91000](http://www.vishay.com/doc?91000)
- [42] «STMicroelectronics, STF13N60M2 Datasheet"», 2014, Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://n9.cl/uag01w>
- [43] F. Miyara y U. Nacional De Rosario, «ELECTRÓNICA III», Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <http://www.fceia.unr.edu.ar/enica3>

- [44] «Como evitar quemar los transistores. Calculo y eleccion disipador calor (Clase 54) - YouTube». Accedido: 21 de mayo de 2024. [En línea]. Disponible en: <https://www.youtube.com/watch?v=x8nKy71afas&t=604s>
- [45] «Disipa, “Catálogo de disipadores”,Disipa, Barcelona,2019». Accedido: 16 de mayo de 2024. [En línea]. Disponible en: <https://n9.cl/osc2o>
- [46] M. Fernández Ros, *Manual de diseño de circuitos impresos con Circuit Design Suite v09 de National Instruments® [Recurso electrónico]* / Manuel Fernández Ros... [et al.]. en Textos Docentes; n.º 55. Almería: Editorial Universidad de Almería, 2020.
- [47] «How to layout a PCB for an instrumentation amplifier - Precision Hub - Archives - TI E2E support forums». Accedido: 19 de mayo de 2024. [En línea]. Disponible en: [https://e2e.ti.com/blogs\\_/archives/b/precisionhub/posts/how-to-layout-a-pcb-for-an-instrumentation-amplifier](https://e2e.ti.com/blogs_/archives/b/precisionhub/posts/how-to-layout-a-pcb-for-an-instrumentation-amplifier)

## 7 Anexos

### 7.1 Código Arduino del proyecto

El código está disponible en Github en el siguiente enlace: <https://github.com/Jordan-Meow/TFG.git>

#### Código Main

```
/*===== DEFINICIONES COMPONENTES, ACTUADOR,LEDS, LIBRERIAS =====*/
// Librerias utilizadas
#include "Horno.h"
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
// Sensor Pt100
#define PT100 A4
// Actuador MOSFET
#define MOSFET 6
// Indicadores luminosos LEDs
#define COLD 8
#define HOT 10
// Ventilador
#define FAN 40
// Definir constantes de la pantalla
// Ancho pantalla OLED
#define ANCHO_PANTALLA 128
// Alto pantalla OLED
#define ALTO_PANTALLA 64
// Creación del objeto display
Adafruit_SSD1306 display(ANCHO_PANTALLA, ALTO_PANTALLA, &Wire, -1);
// Definicion de constantes
const unsigned long INTERVALO_DATOS = 1000UL;
const unsigned long INTERVALO_MUESTREO = 20000UL;
unsigned long actual, actual2;
unsigned long evento_datos = 1000UL;
unsigned long evento_muestreo = 0UL;
unsigned long ultimoCambioModo = 0.0;
const float parar = 300.0;
const float auxM = 301.0;
const float auxA = 302.0;
bool bandeA = false;
bool bandeM = false;
bool bandeR = false;
bool automatico = false;
bool cambioModo = false;
/*===== VARIABLES GLOBALES =====*/
//Temperatura del sensor
```

```

float T1;
//Actuador
float H1 = 0.0;
// Referencia
float r = 0.0;

float auxmodo;

//Parámetros del PID teóricos interactivos
float kc, ti, td;
//Parámetros del PID teóricos esquema paralelo
float kp, ki, kd;
//Variables para el controlador
float proporcional, derivativo, aux_u, Tt;
float lastintegral = 0.0;
float integral;
float es = 0.0;
float salida, lastsalida, dsalida;
//Señal de control
float u_control, u_max, u_min;
//Señal del error
float error;
float lasterror;
//Variables auxiliares
String comparar;
//Variable para definir el modo de control Manual o Automatico
bool controlA = false;
bool controlM = false;
bool Control = false;
/*===== MODELADO DEL SISTEMA =====*/
//tramo 1 de temperatura ambiente a 142 grados
float K = 6.1826, tau1 = 452.62, tau2 = 334.93, tr = 10;
float lmd = 394;
//Periodo de Muestreo
int Ts = 20;

/*===== FUNCION SETUP =====*/
void setup() {
  pinMode(HOT, OUTPUT); //Led "Caliente" como salida
  pinMode(COLD, OUTPUT);
  pinMode(FAN, OUTPUT);
  digitalWrite(HOT, LOW);
  digitalWrite(COLD, LOW);
  digitalWrite(FAN, LOW);
  analogReference(INTERNAL2V56); //Referencia analógica PIN AREF (2.56V)
  //Configuración del puerto serial
  Serial.begin(9600);
}

```

```

Serial.setTimeout(100);
analogWrite(mosfet, 0);
//Inicializacion de la pantalla oled
Pantalla();
/*===== PARAMETROS DEL CONTROLADOR NO INTERACTIVO =====*/
kc = 0.3153;
ti = 787.55;
td = 192.4907;

//Pasar los parámetros del controlador no interactivo a paralelo
kp = kc;
ki = (kc) / (ti);
kd = kc * (td);
//Tiempo de tracking para el back calculation
Tt = 389.3533;
//Definición de los valores máximos del actuador (%PWM)
u_max = 60.0;
u_min = 0.0;
}
/*===== CONTROLADOR PID DIGITAL UTILIZANDO APROXIMACIONES POR BLOQUES
=====*/
/*===== FUNCION PRINCIPAL =====*/
void loop() {
T1 = TempRead(PT100);
actual = millis();
String dato, degC;
Iluminacion();
Ventilacion();

if (!cambioModo && Serial.available()) {
//leemos el dato enviado
String dato = Serial.readString();
int ini = dato.indexOf('I') + 1;
int fin = dato.indexOf('F', ini);
if (ini > 0 && fin > ini) {
String degC = dato.substring(ini, fin);
auxmodo = degC.toFloat();

if (auxmodo == parar) {
Reset();
}
}

if (auxmodo == auxA) {
controlA = true;
cambioModo = true;
ultimoCambioModo = actual;
}
}
}

```

```

    Control = true;
}

if (auxmodo == auxM) {
    controlM = true;
    cambioModo = true;
    Control = true;
}
}

if (controlA) {
    ControlAutomatico();
    if (actual - ultimoCambioModo >= evento_muestreo) {
        evento_muestreo = INTERVALO_MUESTREO;
        ultimoCambioModo = actual;
        PID_Back_calculation();
    }
}

} else {
    ControlManual();
}

if (actual >= evento_datos) {
    EnviarDatos();
    evento_datos += INTERVALO_DATOS;
    Apantalla();
}
}

/*===== PROCEDIMIENTOS =====*/
void Reset(void) {
    H1 = 0.0;
    analogWrite(mosfet, 0);
    bandeA = false;
    bandeM = false;
    Control = false;
    controlA = false;
    controlM = false;
    automatico = false;
    cambioModo = false;
    proporcional = 0.0;
    derivativo = 0.0;
    lastintegral = 0.0;
    es = 0.0;
    salida = 0.0;
}

```

```

lastsalida = 0.0;
integral = 0.0;
error = 0.0;
auxmodo = 0.0;
r = 0.0;
aux_u = 0.0;
u_control = 0.0;
ultimoCambioModo = 0.0;
evento_muestreo = 0.0;

}

/*===== PANTALLA =====*/
void Pantalla(void) {

    // Iniciar pantalla OLED en la dirección 0x3C
    if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
#define __DEBUG__
        Serial.println("No se encuentra la pantalla OLED");
#endif
        while (true)
            ;
    }
    // Clear the buffer.
    display.clearDisplay();
}

void Apantalla(void) {
    static const unsigned char PROGMEM datos_imagen[72] = {
        0xff, 0xff, 0xfc, 0xbff, 0xff, 0xec, 0x9f, 0xff, 0xc4,
        0xa7, 0xff, 0x84, 0xb3, 0xff, 0x24, 0xa9, 0x8e, 0xd4,
        0xa4, 0x01, 0x94, 0xa3, 0xff, 0x14, 0xaf, 0xff, 0xd4,
        0xbff, 0xff, 0xf4, 0xbff, 0xff, 0xf4, 0xbff, 0xff, 0xf4,
        0xb8, 0xfc, 0xf4, 0xbd, 0xfc, 0xf4, 0xbff, 0xff, 0xf4,
        0xbff, 0x8f, 0xf4, 0xa0, 0x84, 0x34, 0x9d, 0xcf, 0xec,
        0xd6, 0x5b, 0xec, 0xc9, 0xce, 0xdc, 0xe6, 0x73, 0x9c,
        0xf3, 0xfe, 0x3c, 0xfc, 0x70, 0xfc, 0xff, 0x03, 0xfc
    };

    static String temp, control;
    temp = String(T1, 2);
    control = String(H1, 2);
    // Tamaño del texto
    display.setTextSize(2);
    // Color del texto
    display.setTextColor(SSD1306_WHITE);
    // Posición del texto
}

```

```
display.setCursor(0, 0);
//Pintar bitmap en la pantalla
display.drawBitmap(106, 40, datos_imagen, 22, 24, SSD1306_WHITE);
display.display();
// Escribir texto
//display.print("Temp");
display.print(temp);
display.setCursor(70, 0);
display.setTextSize(1);
display.write(248);
display.setTextSize(2);
display.println("C");
display.print(control);
display.setCursor(70, 16);
display.write(37);
display.print("PWM");
// Enviar a pantalla
display.display();

if (Control) {
    display.setCursor(0, 48);
    display.println("ON");
    display.display();
} else {
    display.setCursor(0, 48);
    display.println("OFF");
    display.display();
}

display.clearDisplay();
}

void Ventilacion(void) {

//Indicación Luminica
if (H1 > 5.0) {
    //HORNO CALIENTE
    digitalWrite(fan, HIGH);
} else {
    //HORNO FRIO
    digitalWrite(fan, LOW);
}
}

void Iluminacion(void) {

//Indicación Luminica
if (T1 > 30.0) {
```

```

//HORNO CALIENTE
digitalWrite(HOT, HIGH);
digitalWrite(COLD, LOW);
} else {
//HORNO FRIO
digitalWrite(COLD, HIGH);
digitalWrite(HOT, LOW);
}
}

/*===== FUNCION DEL CONTROL MANUAL =====*/
void ControlManual(void) {
static String datoManual, Maux;
static float HManual;

if (controlM && Serial.available()) {
//leemos el dato enviado
String datoManual = Serial.readString();
int ini = datoManual.indexOf('I') + 1;
int fin = datoManual.indexOf('F', ini);
if (ini > 0 && fin > ini) {
String Maux = datoManual.substring(ini, fin);
HManual = Maux.toFloat();

if (HManual == parar) {
Reset();
}
if (HManual == auxA) {
controlM = false;
controlA = true;
ultimoCambioModo = actual;
}
if ((HManual != parar) && (HManual != auxA)) {
H1 = HManual;
bandeM = true;
}
}
}

if (H1 > 60.0) {
H1 = 60.0;
}
if (H1 < 0.0) {
H1 = 0.0;
}
lastintegral = H1;
es = 0.0;
}

```

```

r = TempRead(pt100);
lastsalida = r;
analogWrite(MOSFET, map(H1, 0, 100, 0, 255));
}

// /*===== FUNCION DEL CONTROL AUTOMATICO
=====
void ControlAutomatico(void) {
    static String datoAuto, Aaux;
    static float HAuto;

    if (controlA && Serial.available()) {
        //leemos el dato enviado
        String datoAuto = Serial.readString();
        int ini = datoAuto.indexOf('I') + 1;
        int fin = datoAuto.indexOf('F', ini);
        if (ini > 0 && fin > ini) {
            String Aaux = datoAuto.substring(ini, fin);
            HAuto = Aaux.toFloat();

            if (HAuto == parar) {
                Reset();
            }
            if (HAuto != parar) {
                r = HAuto;
                bandeA = true;
            }
        }
    }
}

void PID_Back_calculation(void) {

    /*Calculo del controlador utilizando aproximaciones euler hacia atrás
    para la integral junto con back calculation para el antiwindup,
    el derivativo utiliza la aproximación hacia atrás*/

    //leer salida actual
    salida = TempRead(pt100);

    //Cálculo de la señal del error actual
    error = r - salida;
    //Cálculo de la señal de salida actual
    dsalida = salida - lastsalida;

    //Cálculo de la Acción proporcional
    proporcional = kp * error;
    // Cálculo del termino integral
    integral = lastintegral + Ts * (ki * error + (es / (Tt)));
}

```

```

// Cálculo del termino derivativo
derivativo = -(kd * dsalida) / Ts;
//derivativo = (kd * (error-lasterror)) / Ts;
//Cálculo de la señal de control actual
u_control = proporcional + integral + derivativo;
aux_u = u_control;
if (u_control > u_max) {
    u_control = u_max;
}
if (u_control < u_min) {
    u_control = u_min;
}
es = u_control - aux_u;
H1 = u_control;
//Aplica la acción de control en el PWM
analogWrite(MOSFET, map(H1, 0, 100, 0, 255)); //Max=100, Min=0

//Actualización de las señales de error y salida
lasterror = error;
lastsalida = salida;
lastintegral = integral;
}

/*=====
 ===== ENVIO DE DATOS POR EL PUERTO SERIE
 =====*/
void EnviarDatos(void) {
    //Usar la interfaz de Matlab
    char str[50];
    sprintf(str, "%lu %d %d\n", millis(), int(T1 * 100), int(H1));
    Serial.print(str);
}

}

```

Código cpp

```

#include "Horno.h"

float TempRead(int sen)
{
    float S1,aux;
    int i;
    //Filtro de promedio móvil en la lectura ADC
    //aux=0;
    // for(i=0;i<10;i++){
    //    //cli();

```

```
// aux = aux + (((2.0*(float(analogRead(sen))*2.56/1023.0-0.04532))/(109.21*0.0459*0.0036137))+5.0); //pt100
// sei();
// delay(1);
// }
//S1 = aux/10;
S1 =((2.0*(float(analogRead(sen))*2.56/1023.0-0.04532))/(109.21*0.0459*0.0036137))+5.0;
//pt100
return(S1);
}
```

Cabecera h

```
#include<Arduino.h>
// Encabezados de la librería horno

float TempRead(int sen);

void UpdatePast(float v[],int kT);

float PID(float u[], float e[], float q0, float q1, float q2);
```

## 7.2 Código Matlab de la interfaz gráfica

```
%% Monitoreo de señales en tiempo Real
function varargout=interfaz_modelado(varargin)
parar=false;
ready=true;
auto=false;
manu=false;
fclose('all')
global tiempo salida escalon control

fig(1)=figure('name','Interfaz de la temperatura del horno','menubar','none','position',[200 200 800 700],'color',[0 0.6 0])
movegui(fig(1),'center');
axe(1)=axes('parent',fig(1),'units','pixels','position',[60 380 600 280],'xlim',[0 40],'ylim',[0 100],'xgrid','on','ygrid','on')
axe(2)=axes('parent',fig(1),'units','pixels','position',[60 50 600 280],'xlim',[0 40],'ylim',[0 100],'xgrid','on','ygrid','on')

set(get(axe(1),'XLabel'),'String','Tiempo (Seg)')
set(get(axe(1),'YLabel'),'String','Temperatura (°C)')
set(get(axe(2),'XLabel'),'String','Tiempo (Seg)')
set(get(axe(2),'YLabel'),'String','Control (%)')
```

```

lin(1)=line('parent',axe(1),'xdata',[],'ydata',[],'Color','r','LineWidth',2.5);
lin(2)=line('parent',axe(1),'xdata',[],'ydata',[],'Color','k','LineWidth',2);
lin(3)=line('parent',axe(2),'xdata',[],'ydata',[],'Color','r','LineWidth',2.5);

Texto(1)=uicontrol('parent',fig(1),'style','text','string','Puerto','position',[680 630 100
50],'BackgroundColor',[0.9 0.6 0.3],'fontsize',18);
Texto(2)=uicontrol('parent',fig(1),'style','text','string','Setpoint','position',[680 260 100
50],'BackgroundColor',[0.9 0.6 0.3],'fontsize',18);
Texto(3)=uicontrol('parent',fig(1),'style','text','string','Gráfico','position',[680 370 100
50],'BackgroundColor',[0.9 0.6 0.3],'fontsize',18);

bot(1)=uicontrol('parent',fig(1),'style','pushbutton','string','Detener','position',[680 50 100
50],'callback',@stop,'fontsize',11)
bot(2)=uicontrol('parent',fig(1),'style','pushbutton','string','Enviar','position',[680 160 100
50],'callback',@enviar,'fontsize',11)
bot(3)=uicontrol('parent',fig(1),'style','pushbutton','string','Salvar','position',[680 320 100
50],'callback',@salvar,'fontsize',11)
bot(4)=uicontrol('parent',fig(1),'style','pushbutton','string','Iniciar','position',[680 560 100
50],'callback',@iniciar,'fontsize',11)
bot(5)=uicontrol('parent',fig(1),'style','pushbutton','string','Manual','position',[680 480 100
50],'callback',@manual,'fontsize',11)
bot(6)=uicontrol('parent',fig(1),'style','pushbutton','string','Automático','position',[680 430 100
50],'callback',@automatico,'fontsize',11)
txbx(1)=uicontrol('parent',fig(1),'style','tex','string','Temp','position',[680 100 100 50],'fontsize',11)
txbx(2)=uicontrol('parent',fig(1),'style','edit','string','000','position',[680 210 100 50],'fontsize',11)

start=0;
ports = serialportlist;
if isempty(ports)
    ports ='NONE';
   clc
    disp('No se ha encontrado CONEXIÓN con el Dispositivo de Control');
else
    start=1;
end
puerta =ports(1);
popup = uicontrol('parent',fig(1),'Style', 'popup','String', ports,'Position', [680 600 100
50],'fontsize',15,'Callback', @puertas);

%% Funcion iniciar
function varargout=iniciar(hObject,eventdata)
    ready=false;
end
%% Funcion Pare
function varargout=stop(hObject,eventdata)
    parar=true;
    write(arduino,"I"+"300"+"F",'char');
    fclose(arduino);
    delete(arduino);
    clear arduino;

```

```

end

%% Funcion enviar
function varargout=enviar(hObject,evndata)
deg1=get(txbx(2),'string');
if auto==true
deg=["I"+deg1+"F"];
write(arduino,deg,'char');
end
if manu==true
deg=["I"+deg1+"F"];
write(arduino,deg,'char');
end

degm=["I"+deg1+"F"];
write(arduino,degm,'char');

end
%% Funcion manual
function varargout=manual(hObject,evndata)
man="301";
write(arduino,"I"+man+"F",'char');
manu=true;
auto=false;

end

%% Funcion automático
function varargout=automatico(hObject,evndata)
aut="302";
write(arduino,"I"+aut+"F",'char');
auto=true;
manu=false;

end

%% Funcion Puerto serie
function varargout=puertas(hObject,evndata)

puerta=popup.String{popup.Value};
end
%% Funcion Salvar
function varargout=salvar(hObject,evndata)
%Renombra variables
rs=escalon;
us=control;
ys=salida;
ts=tiempo;

%Grafica datos

```

```

figure
subplot(2,1,1);
plot(ts,rs,ts,ys,'linewidth',3),grid

title('Laboratorio de Temperatura')
xlabel('Tiempo (s)')
ylabel('Temperatura (C)')

subplot(2,1,2);
plot(ts,us,'linewidth',3),grid
xlabel('Tiempo (s)')
ylabel('Control (%)')
T=[ts;rs;ys;us];
filter = {'*.txt','*.xls'};
[file,path,indx] = uiputfile(filter);
fileID = fopen(strcat(path,file),'w');
fprintf(fileID,'%12s %12s %12s %12s\n','t','r','y','u');
fprintf(fileID,'%12.2f %12.2f %12.2f %12.2f\n',T);
fclose(fileID);
file(end-2:end)='mat';
save(strcat(path,file),'ts','rs','ys','us')
end
%% funcion Graficar
% function varargout=grafique(hObject,eventdata)
tiempo=[0];
salida=[0];
escalon=[0];
control=[0];
deg1="0";

dt=1;
limx=[0 40];
limy=[0 100];
set(axe(1),'xlim',limx,'ylim',limy);
while(ready)
    pause(1);
end
if start
    %% Configura el Puerto Serial

arduino=serialport(puerta,9600,"StopBits",1,"DataBits",8);
fopen(arduino);
% Aquí se usa flushinput() para limpiar el buffer
flushinput(arduino);
%% Grafico
k=5; nit =200000;
while(~parar)
    %% Lectura de Datos por Puerto Serial
    % Leo datos del arduino, el tiempo, la temperatura y la señal de control
    data=readline(arduino);
    % Separo los datos y los convierto en un vector de caracteres

```

```

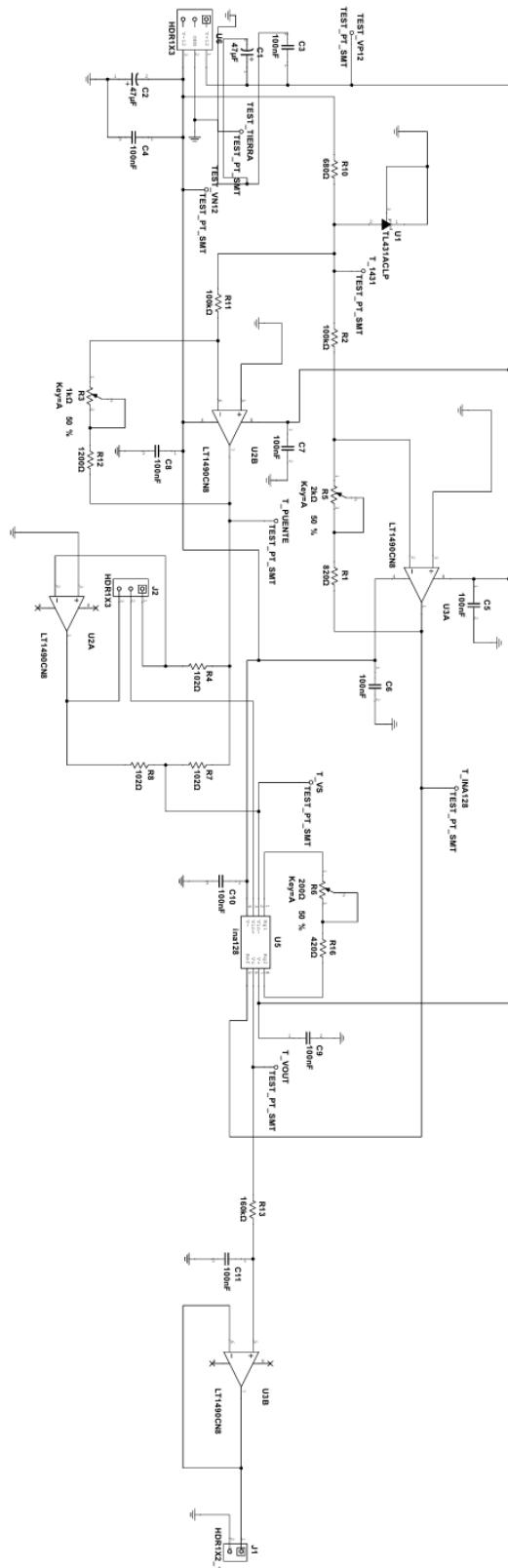
Cadena = strsplit(data);
% El primer dato de cadena lo transformo de cadena a un numero (double) y divido entre mil
time=((str2double(Cadena(1)))/1000);
%El segundo dato de cadena lo transformo de cadena a un numero (double) y divido entre mil
temp=(str2double(Cadena(2))/100;
tempC=string(temp);
ctr=str2double(Cadena(3));
%cargo el valor de temperatura en la caja de texto
set(txbx(1),'string',tempC);
%Actualiza las variables del grafico
tiempo=[tiempo time];
salida=[salida temp];
control=[control ctr];
escalon=[escalon str2double(deg1)];
%Cargo los datos en los axes
set(lin(1),'xdata',tiempo,'ydata',salida);
set(lin(2),'xdata',tiempo,'ydata',escalon);
set(lin(3),'xdata',tiempo,'ydata',control);
%Se espera un segundo para volver a recoger los datos
pause(dt);
% actualizo grafica cuando llega a su limite en tiempo real
if tiempo(end)>=limx
    limx=[0 limx(2)+40];
    set(axe(1),'xlim',limx) ;
    set(axe(2),'xlim',limx);
end
%actualizo grafica cuando llega a su limite en tiempo real
if salida(end)>=limy
    limy=[0 limy(2)+10];
    set(axe(1),'ylim',limy);
end
%actualizo grafica cuando llega a su limite en tiempo real
if escalon(end)>=limy
    limy=[0 escalon(end)+10];
    set(axe(1),'ylim',limy);
end
%Actualizo el contador
k=k+1;
if(k==nit)
    parar=true;
end

end
parar=false;
end
end

```

### 7.3 Esquemas electrónicos

#### Esquema del acondicionador de señal







En la actualidad, la tecnología de montaje superficial para el desarrollo de circuitos impresos es la comúnmente empleada por los diseñadores, puesto que permite una reducción considerable del tamaño de los circuitos electrónicos, así como una disminución de los costes de producción, dejando la tecnología de orificio pasante para el desarrollo de prototipos, ya que permiten una rápida implementación y facilidad a la hora de soldar los componentes electrónicos. El empleo y testeo de prototipos es un paso previo a la comercialización de un producto final o también para afianzar los conocimientos en diseño de circuitos electrónicos a nivel educativo.

En este trabajo de fin de grado, se ha modificado un horno de bajo coste para que siga perfiles de temperatura que se emplean en la soldadura por reflujo, con el objetivo de poder realizar prototipos de montaje superficial a nivel educativo. Para ello se ha utilizado un sensor de temperatura PT100, se ha diseñado la electrónica para acondicionar la señal del sensor, la electrónica de potencia para controlar la resistencia del horno, además del modelado del sistema y el diseño de una estrategia de control. Los componentes utilizados, en la mayor parte posible, han sido reciclados de otros dispositivos averiados promoviendo la reutilización de componentes electrónicos.

Currently, surface mount technology is commonly employed by designers for the development of printed circuit boards, as it allows for a significant reduction in the size of electronic circuits, as well as a decrease in production costs. Through-hole technology is left for prototype development, as it allows for quick implementation and ease of soldering electronic components. The use and testing of prototypes are a preliminary step in the commercialization of a final product or to solidify knowledge in electronic design at an educational level.

In this bachelor's thesis, a low-cost oven has been modified to follow temperature profiles used in reflow soldering, aiming to prototype surface mount technology at an educational level. To achieve this, a PT100 temperature sensor has been used, and electronics have been designed to condition the sensor signal, as well as power electronics to control the oven's resistance, in addition to system modelling and the design of a control strategy. Components have been mostly sourced from recycled parts of other malfunctioning devices, promoting the reuse of electronic components.