Diseño de plataforma electrónica de pruebas para la implementación de los módulos GPIO, PWM, UART, I^2C, I^2S y SPI de la Raspberry Pi 3B+

Francisco José Montúfar Gudiel

UVG UNIVERSIDAD DEL VALLE DE GUATEMALA

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseño de plataforma electrónica de pruebas para la implementación de los módulos GPIO, PWM, UART, I^2C, I^2S y SPI de la Raspberry Pi 3B+

Trabajo de graduación presentado por Francisco José Montúfar Gudiel
para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala,

2024

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseño de plataforma electrónica de pruebas para la implementación de los módulos GPIO, PWM, UART, I^2C, I^2S y SPI de la Raspberry Pi 3B+

Trabajo de graduación presentado por Francisco José Montúfar Gudiel
para optar al grado académico de Licenciado en Ingeniería Electrónica

Guatemala,

2024

Vo.Bo.:

(f) _____

MBA. Pablo Daniel Mazariegos de la Cerda

Tribunal Examinador:

(f) _____

Ing. Kurt Kellner

(f) _____

MSc. José Eduardo Morales

(f) _____

MSc. Pedro Iván Castillo

Fecha de aprobación: Guatemala, 6 de enero de 2024.

Prefacio

La elaboración del siguiente proyecto surge del interés en el área de sistemas embebidos, ya que, a raíz de este campo, es posible el desarrollo y optimización de sistemas electrónicos, los cuales son el pilar principal para el funcionamiento de cualquier dispositivo en la actualidad. Para llevar a cabo este proyecto, fue necesario aplicar habilidades y conocimientos de diversas áreas, tales como programación de sistemas embebidos, electrónica analógica, diseño de PCBs y sistemas operativos. La presente tesis no habría sido posible sin la capacitación y aprendizaje otorgado por los catedráticos de la Universidad del Valle de Guatemala.

Quiero agradecer a Dios por haberme dado la oportunidad de estudiar y a mis padres por sus esfuerzos para permitirme pertenecer a la Universidad del Valle de Guatemala y recibir la educación adecuada para formarme como profesional. También un agradecimiento a la familia García Bolaños, por haber brindado el apoyo necesario para poder seguir mis estudios cuando me encontraba en una situación complicada. De igual manera, quiero agradecer a mi asesor de tesis al MBA Pablo Mazariegos de la Cerda por el tiempo dedicado para resolver mis dudas, compartir su conocimiento y dar retroalimentación.

Por último, pero no menos importante, quiero agradecer mucho a toda mi familia por haberme animado a siempre seguir mis metas. También a las amistades formadas durante la carrera, les agradezco por los momentos, las risas, el apoyo y especialmente el cariño. Del mismo modo, agradezco mucho a mi grupo de amigos del colegio que han estado a mi lado de principio a fin de mis estudios, ya que a pesar de los años, siempre se mantuvo el lazo que formamos. Finalmente quiero agradecer a las personas especiales en mi vida que siempre han estado para sostenerme, inspirarme y recordarme que no estoy solo.

Índice

Prefacio III

Lista de figuras XI

Lista de cuadros XII

Resumen XIII

Abstract XIV

1. Introducción 1

2. Antecedentes 2
2.1. Curso Electrónica Digital 3 . . . 2
2.2. Curso Simulación de circuitos y fabricación de PCBs . . . 2
2.3. Ordenador de placa única Raspberry Pi . . . 3
2.4. Implementación de Raspberry Pi en proyectos anteriores . . . 7

3. Justificación 8

4. Objetivos 9
4.1. Objetivo general . . . 9
4.2. Objetivos específicos . . . 9

5. Alcance 10

6. Marco teórico 11
6.1. *Raspberry Pi 3B+* . . . 11
6.2. *Universal Asynchronous Receiver-Transmitter* (UART) . . . 12
6.3. *Serial Peripheral Interface* (SPI) . . . 14
6.4. *Inter-Integrated Circuit* (I^2C) . . . 16
6.5. *Pulse-Width Modulation* . . . 18
6.6. *Inter-IC Sound* (I^2S) . . . 19
6.7. *Printed Circuit Boards* . . . 21

6.8. *Wiring-Pi* . . . 25
6.9. Driver A4988 . . . 26
6.9.1. Operación del dispositivo . . . 27
6.10. Micrófono digital SPH0645LM4H-B . . . 28
6.10.1. Operación del dispositivo . . . 29
6.11. Amplificador de potencia MAX 98357 . . . 30
6.12. Conversor analógico-digital MCP 3002 . . . 31
6.13. Conversor digital-analógico MCP 4921 . . . 31

7. Diseño de la plataforma **33**
7.1. Librerías . . . 33
7.2. Esquemáticos . . . 33
7.2.1. *Layout* . . . 35
7.3. Modelo 3D . . . 38

8. Fabricación de la plataforma **40**
8.1. Archivos generados . . . 40
8.1.1. *Gerbers* . . . 41
8.1.2. *NC Drill* . . . 44
8.2. Placa física . . . 44
8.3. Ensamble de la placa . . . 45

9. Experimentos con la plataforma **48**

10. Documentación **60**

11. Conclusiones **62**

12. Recomendaciones **63**

13. Bibliografía **64**

14. Anexos **66**
14.1. Instalaciónes requeridas . . . 66
14.1.1. Instalación Altium Designer . . . 66
14.1.2. Instalación *Raspberry Pi OS* . . . 79
14.1.3. Instalación *Wiring Pi* . . . 81
14.1.4. Instalación de paquetes para micrófono SPH0645LM4H . . . 82
14.1.5. Instalación de paquetes para amplificador MAX98357 . . . 83
14.2. Símbolos . . . 83
14.3. *Footprints* . . . 88
14.4. Modelos 3D . . . 94
14.5. Esquemáticos . . . 103
14.6. Prototipo . . . 111
14.6.1. *Layout* . . . 111
14.6.2. Modelo 3D . . . 113
14.6.3. *Gerbers* . . . 114
14.6.4. *NC Drill* . . . 115
14.6.5. *PCB Prints* . . . 116

14.6.6. Placa física 117
14.6.7. Ensamble de la placa 118
14.7. Experimentos con el prototipo 120

Lista de figuras

1. Raspberry Pi Zero . . . 3
2. Raspberry Pi Zero W . . . 4
3. Raspberry Pi 1 Model A+ . . . 4
4. Raspberry Pi 1 Model B+ . . . 4
5. Raspberry Pi 3 Model B . . . 5
6. Raspberry Pi 3 Model B+ . . . 5
7. Raspberry Pi 3 Model A+ . . . 5
8. Raspberry Pi 4 Model B . . . 6
9. Raspberry Pi Pico . . . 6
10. Raspberry Pi Compute Module . . . 6
11. AlphaBot 2 . . . 7

12. Esquema de comunicación UART entre dispositivos. . . . 13
13. Demostración de bus de datos y UART . . . 13
14. Paquete UART . . . 13
15. Lectura de datos en SPI . . . 15
16. Comunicación SPI . . . 15
17. Comunicación SPI con un *Master* y un *Slave* . . . 16
18. Comunicación SPI con un *Master* y múltiples *Slaves* . . . 16
19. Esquema de comunicación I^2C entre dispositivos . . . 17
20. Mensaje de I^2C . . . 17
21. Múltiples nodos y múltiples principales en I^2C . . . 18
22. Diferentes ciclos de trabajo de una PWM . . . 19
23. *Inter-IC Sound* . . . 20
24. *Pulse code modulation* . . . 20
25. *Diagrama de bloques de Pulse code modulation* . . . 21
26. Capas de un PCB . . . 22
27. Placa de circuito impreso . . . 22
28. Corriente de sección transversal según estándar IPC-2221 para conductores externos . . . 24
29. Ancho de conductor respecto al sección transversal según estándar IPC-2221 . 24
30. *pinout*de *WiringPi* . . . 26
31. Driver A4988 para motor *Stepper* . . . 27

32. Diagrama de aplicación Driver A4988 para motor *Stepper* 27
33. Diagrama de aplicación Driver A4988 para motor *Stepper* 28
34. Diagrama de aplicación micrófono SPH0645LM4H-B 29
35. Micrófono digital SPH0645LM4H-B 29
36. Diagrama de aplicación para amplificador MAX 98357 30
37. Amplificador MAX 98357 30
38. ADC MCP 3002 31
39. DAC MCP 4921 32

40. Esquemáticos de la plataforma 34
41. *Layout* de la *Top Layer* de la plataforma 35
42. *Layout* de la *Bottom Layer* de la plataforma 35
43. *Layout* de la *Layer* interna 1 de la plataforma 36
44. *Layout* de la *Layer* interna 2 de la plataforma 36
45. *Layout* del *Top Overlay* de la plataforma 37
46. Modelo 3D de la plataforma 38

47. *Gerber* de la *Top Layer* 41
48. *Gerber* de la *Layer* interna 1 41
49. *Gerber* de la *Layer* interna 2 42
50. *Gerber* de la *Bottom Layer* 42
51. *Gerber* de la *Keepout Layer* 43
52. *Gerber* del *Top Overlay* 43
53. Archivo generado NC Drill 44
54. *Top Layer* placa física 44
55. *Bottom Layer* placa física 45
56. Vista superior de la placa física 46
57. Vista ortogonal de placa física conectada a la *Raspberry Pi 3B+* 46
58. Vista superior de placa física conectada a la *Raspberry Pi 3B+* 47
59. Vista frontal de placa física conectada a la *Raspberry Pi 3B+* 47

60. Contador de 4 bits utilizando LEDs y pulsadores 49
61. Contador de 4 bits utilizando LEDs y pulsadores 49
62. Vista de la señal analógica en el osciloscopio 50
63. Gráfica de los valores de la conversión 50
64. Resultado del comando arecord 51
65. Salida generada por micrófono I^2S 51
66. Resultado de comando aplay 52
67. Señal de salida del módulo MAX98357 52
68. Motor *Stepper* 53
69. Motor *Stepper* 54
70. Dato enviado por UART 55
71. Cadena de Arduino enviada por UART y recibida por la *Raspberry Pi* 3B+ . 55
72. Salida del DAC 56
73. PWM generada 57
74. PWM para servomotor 58
75. Datos recibidos del RTC usando I^2C 58

76. Carátula del folleto 60

77. Pinout de la plataforma mostrado en el folleto . . . 61
78. Parte de las instalaciones requeridas mostradas en el folleto . . . 61

79. Instalador Altium Designer . . . 66
80. Instalador Altium Designer . . . 67
81. Instalador Altium Designer . . . 67
82. Instalador Altium Designer . . . 68
83. Instalador Altium Designer . . . 68
84. Instalador Altium Designer . . . 69
85. Instalador Altium Designer . . . 69
86. Instalador Altium Designer . . . 70
87. Instalador Altium Designer . . . 70
88. Instalador Altium Designer . . . 71
89. Instalador Altium Designer . . . 71
90. Instalador Altium Designer . . . 72
91. Instalador Altium Designer . . . 72
92. Instalador Altium Designer . . . 73
93. Instalador Altium Designer . . . 73
94. Instalador Altium Designer . . . 74
95. Instalador Altium Designer . . . 74
96. Instalador Altium Designer . . . 75
97. Instalador Altium Designer . . . 75
98. Altium Designer . . . 76
99. Altium Designer . . . 76
100. Altium Designer . . . 77
101. Altium Designer . . . 77
102. Altium Designer . . . 78
103. Altium Designer . . . 78
104. Altium Designer . . . 79
105. Altium Designer . . . 79
106. *Raspberry Pi Imager* . . . 80
107. *Raspberry Pi Imager* . . . 80
108. Instalación WiringPi . . . 81
109. Reconocimiento del micrófono I^2S . . . 82
110. Símbolo de esquemático de 2N7000 . . . 83
111. Símbolo de esquemático de *Header* hembra 2x20 . . . 84
112. Símbolo de esquemático de *Driver* A4988 . . . 84
113. Símbolo de esquemático de *Buzzer* pasivo . . . 84
114. Símbolo de esquemático de *Pinhead 1x1* . . . 85
115. Símbolo de esquemático de *Pinhead 1x3* . . . 85
116. Símbolo de esquemático de *Pinhead 1x6* . . . 85
117. Símbolo de esquemático de LED . . . 86
118. Símbolo de esquemático de amplificador MAX98357 . . . 86
119. Símbolo de esquemático de MCP 3002 . . . 86
120. Símbolo de esquemático de MCP 4921 . . . 86
121. Símbolo de esquemático de resistencia . . . 87
122. Símbolo de esquemático de potenciómetro . . . 87
123. Símbolo de esquemático de micrófono SPH0645LM4H . . . 87

124. Símbolo de esquemático de pulsador . . . 88
125. *Footprint* de 2N7000 . . . 88
126. *Footprint* de *Header*hembra 2x20 . . . 88
127. *Footprint* de *Driver*A4988 . . . 89
128. *Footprint* de *buzzer* . . . 89
129. *Footprint* de *Pinhead 1x1* . . . 90
130. *Footprint* de *Pinhead 1x3* . . . 90
131. *Footprint* de *Pinhead 1x6* . . . 90
132. *Footprint* de LED . . . 91
133. *Footprint* de amplificador MAX98357 . . . 91
134. *Footprint* de MCP 3002 . . . 91
135. *Footprint* de MCP 4921 . . . 92
136. *Footprint* resistencia THT . . . 92
137. *Footprint* de resistencia SMD . . . 92
138. *Footprint* de potenciómetro . . . 93
139. *Footprint* de micrófono SPH0645LM4H . . . 93
140. *Footprint* de pulsador . . . 93
141. Modelo 3D de 2N7000 . . . 94
142. Modelo 3D de 2N7002 SMD . . . 95
143. Modelo 3D de *Header*macho-hembra 2x20 . . . 95
144. Modelo 3D de *Header*hembra 2x20 . . . 95
145. Modelo 3D de *Driver*A4988 . . . 96
146. Modelo 3D de *buzzer* . . . 96
147. Modelo 3D de *Pinhead 1x1* . . . 97
148. Modelo 3D de *Pinhead 1x3* . . . 97
149. Modelo 3D de *Pinhead 1x6* . . . 98
150. Modelo 3D de de LED . . . 98
151. Modelo 3D de amplificador MAX98357 . . . 98
152. Modelo 3D de de MCP 3002 . . . 99
153. Modelo 3D de de MCP 4921 . . . 99
154. Modelo 3D de resistencia THT . . . 100
155. Modelo 3D de resistencia SMD . . . 100
156. Modelo 3D de potenciómetro . . . 101
157. Modelo 3D de micrófono SPH0645LM4H . . . 101
158. Modelo 3D de pulsador . . . 102
159. Conexiones para el ADC . . . 103
160. Conexiones para los pulsadores . . . 104
161. Conexiones para el *Buzzer* . . . 105
162. Conexiones para el DAC . . . 105
163. Conexiones para I^2C . . . 106
164. Conexiones para I^2S . . . 107
165. Conexiones para PWM . . . 107
166. Conexiones para stepper . . . 108
167. Conexiones del GPIO . . . 108
168. Conexiones para los LEDs . . . 109
169. Conexiones para UART . . . 110
170. *Layout* de la *Top Layer* de la plataforma . . . 111
171. *Layout* de la *Bottom Layer* de la plataforma . . . 112

172. Modelo 3D de la plataforma . . . 113
173. *Gerber* de la *Top Layer* . . . 114
174. *Gerber* de la *Bottom Layer* . . . 114
175. *Gerber* de la *Keepout Layer* . . . 115
176. Archivo generado NC Drill . . . 115
177. *Prints* de *Top Layer* . . . 116
178. *Prints* de *Bottom Layer* . . . 117
179. *Top Layer* placa física . . . 117
180. *Bottom Layer* placa física . . . 118
181. Vista superior de la placa física . . . 118
182. Vista ortogonal de placa física conectada a la *Raspberry Pi 3B+* . . . 119
183. Vista superior de placa física conectada a la *Raspberry Pi 3B+* . . . 119
184. Contador de 4 bits utilizando LEDs y pulsadores . . . 120
185. Contador de 4 bits utilizando LEDs y pulsadores . . . 120
186. Contador de 4 bits utilizando LEDs y pulsadores . . . 121
187. Vista de la señal analógica en el osciloscopio . . . 121
188. Gráfica de los valores de la conversión . . . 121
189. Resultado del comando arecord . . . 122
190. Salida generada por micrófono I^2S . . . 122
191. Segmento de la salida generada por micrófono I^2S . . . 123
192. Señal de entrada del módulo MAX98357 . . . 123
193. Señal de salida del módulo MAX98357 . . . 124
194. Pulso generado para el funcionamiento del módulo A4988 . . . 124
195. Motor *Stepper* . . . 125
196. Motor *Stepper* . . . 126
197. Dato enviado por UART . . . 127

Lista de cuadros

1. Comparación con modelos anteriores a *Raspberry Pi 3B+* 12
2. Comparación con modelos posteriores a *Raspberry Pi 3B+* 12

Resumen

En el proyecto desarrollado, se muestra un trabajo de graduación que tuvo como objetivo principal el diseño e implementación de una plataforma electrónica para validar y experimentar con las diversas funciones de un ordenador de placa única *Raspberry Pi* 3B+. Para poder llevar a cabo el diseño y la fabricación del proyecto, se seleccionaron diversos componentes para ser utilizados con cada uno de los módulos y pines digitales del dispositivo para luego realizar un esquemático y un modelo 3D de la plataforma. Posteriormente, la plataforma fue fabricada y se realizaron diversos experimentos que comprobaron el funcionamiento adecuado de los módulos.

El diseño de los esquemáticos y de la plataforma fue posible con el *software* de diseño de PCBs llamado *Altium Designer*, ya que este ofrece una gran cantidad de herramientas especializadas para el diseño electrónico. También nos brinda una vista previa sobre el aspecto físico de la placa ya materializada, lo que nos da una mejor idea de como sería el producto final. En el diseño de PCBs, es necesario seguir reglas de fabricación y un estándar de diseño que nos orientan a hacer una plataforma electrónica funcional.

Toda la experimentación fue hecha con algoritmos desarrollados en C y en *Python*, en donde se comprueba el funcionamiento de todos los protocolos y módulos utilizados. Esto es una parte esencial para el uso de la plataforma electrónica, ya que los algoritmos son los que accionan los pines de propósito general que irán conectados a la plataforma electrónica.

La plataforma se realizó con el fin de facilitar el uso y comprensión de ciertas funcionalidades del Raspberry Pi, por lo que se realizó una documentación que incluye un instructivo, imágenes que ilustran como conectar la plataforma y códigos de ejemplo para cada módulo.

Abstract

In the following graduation project, the main objective is to design and implement an electronic platform to validate and experiment with the various functions of a Raspberry Pi 3B+. In order to design and fabricate the project, various components were selected to use with each of the modules and digital pins of the device, resulting in the creation of a schematic and a 3D model of the platform. Then, the platform was manufactured, and various experiments were conducted to verify the proper functioning of the modules.

The design of the schematics and the platform was possible using the PCB design software called Altium Designer, because it provides a wide range of specialized tools for electronic design. It also offers a preview of the physical appearence of the board, giving us a better idea of the final result. In PCB design, it is essential to follow fabrication rules and design standards that guide us in creating a functional electronic platform.

All experimentation was made by using algorithms developed in C, where the functionality of all protocols and modules was verified. This is a crucial part of using the electronic platform, because the algorithms control the general-purpose pins connected to it.

The platform was created to simplify the use and understanding of certain Raspberry Pi functionalities, so proper documentation was prepared, including instructions, images illustrating how to connect the platform, and example codes for each module.

CAPÍTULO 1

Introducción

En un mundo con constante desarrollo tecnológico, es necesario contar con herramientas versátiles y fáciles de utilizar para abordar los desafíos de manera eficiente. El ordenador de placa única Raspberry Pi, ha sido una herramienta poderosa y accesible que ha revolucionado la informática de bajo costo y la automatización de proyectos. En este trabajo, se presenta el diseño y fabricación de una plataforma electrónica de tipo HAT (*Hardware Attached on Top*), creado específicamente para facilitar el uso de los pines de propósito general y aprovechar sus funciones principales. Se busca la validación del funcionamiento de los diferentes módulos y protocolos de comunicación por medio del desarrollo de diversos algoritmos y la realización de diversos experimentos.

El diseño y fabricación PCBs o *Printed Circuit Boards* es posible con diversas herramientas de software y equipo especial de fresado. Los PCBs nos permiten crear crear circuitos electrónicos compactos y eficientes que son fundamentales en la electrónica moderna. Estos componentes son la base de cualquier dispositivo, desde el mas simple hasta el más complejo. Para lograr un proceso adecuado de diseño y fabricación, se utilizan diversas herramientas tanto de *software* como de *hardware*.

Por medio del programa Altium designer, se diseñaron todos los símbolos, esquemáticos y prototipos de una plataforma electrónica. El proceso comenzó con una evaluación de componentes para asegurarse de que cumplan con las especificaciones necesarias para interactuar con diversos módulos y protocolos del *Raspberry Pi*. Una vez seleccionados los componentes, se crearon las librerías que contienen el símbolo esquemático, el *Footprint* y el modelo 3D, para luego poder hacer el prototipo inicial estableciendo la ubicación de los componentes y realizando las conexiones necesarias.

La etapa de experimentación consiste en la utilización del prototipo, junto a algoritmos desarrollados y la librería *WiringPi*. Con los algoritmos creados,es posible utilizar los protocolos UART, I^2C, I^2S , SPI y el módulo PWM para poder validar el funcionamiento de la plataforma, ya que todos los componentes fueron seleccionados con el fin de utilizar todo lo mencionado. Cabe recalcar que el diseño de la placa se realizó de forma modular para que los componentes que estén mas propensos a dañarse puedan ser reemplazados fácilmente.

CAPÍTULO 2

Antecedentes

2.1. Curso Electrónica Digital 3

En la Universidad del Valle de Guatemala, en el Departamento de Ingeniería Electrónica, Mecatrónica y Biomédica se imparte un curso llamado Electrónica digital 3, en el cual se cubren aspectos teóricos y prácticos del desarrollo de sistemas embebidos. En esta iniciativa académica se incluyen temas como sistemas operativos, *kernels*, procesos, hilos múltiples, comunicación entre dispositivos, sincronización entre procesos, implementación de dispositivos periféricos y comunicación de red. Principalmente, se utiliza un ordenador de placa única de la línea de *Raspberry Pi* con un sistema operativo basado en Debian llamado *Rasperry Pi OS* (Raspbian). Todas las prácticas o demostraciones de conceptos se basan en *software* desarrollado en el lenguaje C.

Las competencias principales del curso tienen como propósito comprender aspectos básicos de sistemas operativos Linux, utilizando dispositivos*Raspberry Pi*, al igual que la creación de programas en el lenguaje C/C++ para PC y sistemas embebidos. También tienen un enfoque hacia la interconexión entre *Raspberry Pi* y otros dispositivos mediante circuitos y/o controladores externos utilizando diferentes protocolos de comunicación.

La metodología del curso consiste en sesiones de teoría en donde se explican todos los conceptos para luego ponerlos en práctica en sesiones de laboratorio, en donde se trabaja con el ordenador de placa única para realizar diferentes tareas que validan los conceptos introducidos en las sesiones anteriores de teoría. Luego de finalizar todas las prácticas, se realiza un proyecto que engloba todo lo visto en el curso.

2.2. Curso Simulación de circuitos y fabricación de PCBs

En la Universidad del Valle de Guatemala, en el Departamento de Ingeniería Electrónica, Mecatrónica y Biomédica se imparte un curso llamado Simulación de circuitos y fabricación

de PCBs, en el cual se introducen las herramientas de simulación de circuitos y los diferentes análisis que pueden realizarse. Asimismo, se trabaja la diagramación y fabricación de PCBs (*Printed Circuit Boards*) por medio de un software llamado Altium Designer. Este software ofrece un entorno de diseño unificado, que permite tener una visión única de todos los aspectos del proceso de diseño de un PCB.[1] La fabricación de las placas se hace mediante los archivos generados por el *software* y una máquina de fresado de placas PCB. Luego se emplean herramientas y técnicas de soldadura de componentes electrónicos para poder finalizar la fabricación del PCB.

Las competencias principales del curso tienen como propósito el análisis de circuitos electrónicos con software especializado para poder comprender el comportamiento de estos. También tienen un enfoque hacia todo lo que es diseño y manufactura de PCBs por medio de la elaboración de esquemáticos, desarrollo de librerías, modelos 3D, reglas de diseño estándar y generación de archivos necesarios para la fabricación.

La metodología del curso consiste en sesiones de teoría en donde se introducen los conceptos y herramientas de *software* a utilizar. Luego en sesiones de laboratorio, se ponen en práctica los conceptos vistos por medio de diferentes tareas realizadas con las herramientas de *software*. El primer bloque del curso gira en torno a las diferentes simulaciones y pruebas que se le pueden realizar a diversos circuitos. Luego se nos introduce al diseño de PCBs en otra herramienta de *software* y, por último, se realizan prácticas para la fabricación por medio de diferentes ejercicios de soldadura con herramientas especializadas.

2.3. Ordenador de placa única Raspberry Pi

Cuando se habla de un *Raspberry Pi*, se hace referencia a una computadora de bajo costo con una amplia variedad de aplicaciones. Fue desarrollada por la *Raspberry Foundation* con el fin de promover y enseñar las ciencias de la computación. La popularidad del mismo fue debido a la versatilidad y desempeño en diferentes proyectos en diversas áreas. [2]

Hoy en día, ya se han desarrollado diversos modelos:

- Raspberry Pi Zero

Figura 1: Raspberry Pi Zero

- Raspberry Pi Zero W

Figura 2: Raspberry Pi Zero W

- Raspberry Pi 1 Model A+

Figura 3: Raspberry Pi 1 Model A+

- Raspberry Pi 1 Model B+

Figura 4: Raspberry Pi 1 Model B+

- Raspberry Pi 3 Model B

Figura 5: Raspberry Pi 3 Model B

- Raspberry Pi 3 Model B+

Figura 6: Raspberry Pi 3 Model B+

- Raspberry Pi 3 Model A+

Figura 7: Raspberry Pi 3 Model A+

- Raspberry Pi 4 Model B

Figura 8: Raspberry Pi 4 Model B

- Raspberry Pi Pico

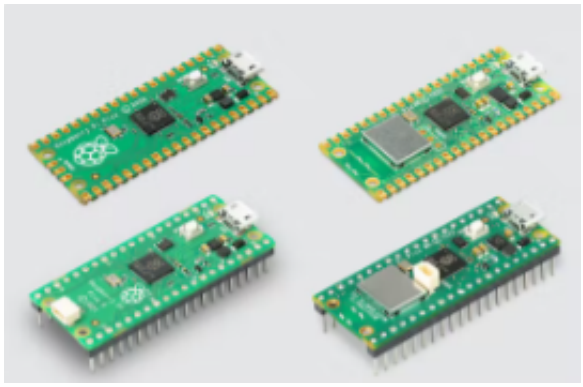

Figura 9: Raspberry Pi Pico

- Raspberry Pi Compute Module

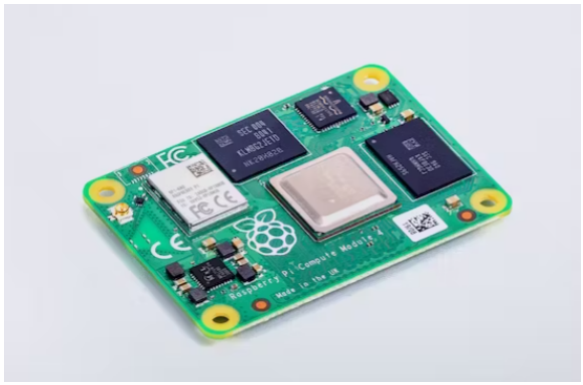

Figura 10: Raspberry Pi Compute Module

2.4. Implementación de Raspberry Pi en proyectos anteriores

En el proyecto llamado *Implementación y Validación del Algoritmo de Robótica de Enjambre Particle Swarm Optimization en Sistemas Físicos* realizado por Alex Daniel Maas de la Universidad del Valle de Guatemala, se utilizó un Raspberry Pi 3B debido a que se requería desarrollar un algoritmo de Optimización del Enjambre de partículas (PSO), en el cual se necesita una gran capacidad de memoria y una frecuencia de operación aceptable debido a que el PSO es un algoritmo que se basa en múltiples iteraciones.[3]

En otro proyecto llamado *Evaluación y validación de plataformas móviles para aplicaciones prácticas de robótica* realizado por Luis Javier Nij de la Universidad del Valle de Guatemala, se utilizó un Raspberry Pi 3B debido a que se requería de programación multihilos, un Módulo de cámara y diversos sensores para un agente robótico denominado *AlphaBot2* [4]

Figura 11: AlphaBot 2

CAPÍTULO 3

Justificación

La *Raspberry Pi* 3B+ es un ordenador de placa única que nos facilita el aprendizaje y comprensión de los diferentes protocolos, módulos y conceptos de sistemas operativos, hilos y multiprocesos, por lo que es necesario desarrollar una herramienta que brinde todos los componentes necesarios para poner en práctica todos lo mencionado. Esta debe tener una gran variedad de componentes, de tal manera que se simplifique o evite la elaboración de los circuitos físicos.

El proyecto tiene dos enfoques principales: crear una herramienta que facilite enseñanza del funcionamiento de un *Raspberry Pi* 3B+ implementando componentes físicos y *software*, de tal manera que pueda ser utilizada en diversos cursos del departamento de Ingeniería Electrónica, Mecatrónica y Biomédica de la Universidad el Valle de Guatemala y evidenciar el funcionamiento adecuado de los componentes por medio de diversos programas de prueba. Para lograr los objetivos, es necesario poner en práctica los conocimientos previos sobre diseño de PCBs, sistemas embebidos, programación y protocolos de comunicación.

La implementación de esta plataforma va a permitir experimentar con las diferentes funciones de un *Raspberry Pi* 3B+ sin necesidad de construir circuitos físicos. Con esta herramienta se facilitará el entendimiento de los pines de propósito general, los protocolos de comunicación y los diversos módulos del ordenador.

CAPÍTULO 4

Objetivos

4.1. Objetivo general

Diseñar una plataforma electrónica para validación y experimentación de las diversas funciones de un de microordenador Raspberry Pi 3B+.

4.2. Objetivos específicos

- Validar el funcionamiento de los diferentes módulos de un Raspberry Pi 3B+ utilizando los pines de propósito general (GPIO).
- Diseñar una placa que permita la interconexión al Raspberry Pi 3B+ y realizar diversos experimentos utilizando los diferentes módulos (PWM) y protocolos de comunicación(UART, SPI, I^2C e I^2S).
- Desarrollar una documentación adecuada para facilitar el entendimiento sobre los diferentes módulos, protocolos y funcionamiento general de la placa.
- Realizar el diseño de la placa de forma modular, lo cual permita reemplazar los componentes defectuosos fácilmente.
- Desarrollar algoritmos que permitan comprobar el funcionamiento adecuado de la plataforma electrónica.

CAPÍTULO 5

Alcance

Con las diferentes consideraciones de uso y las características establecidas, se pudo diseñar y fabricar una plataforma electrónica que tuviera la flexibilidad de trabajar con diversos componentes y que facilitara el uso de los módulos de una *Raspberry Pi*. La plataforma contiene componentes que se acoplan a un protocolo o módulo específico y que tienen la facilidad de ser reemplazadas en caso de algún desperfecto. Se optó por utilizar algoritmos de los lenguajes de programación C y *Python* debido a la diversidad de librerías que tienen y un fácil acceso a documentación.

Como consideraciones para el diseño de la plataforma, se tuvieron que realizar diversos cálculos para establecer el ancho de los *tracks* para interconectar los componentes, lo cual nos lleva a nuestra primer limitante, ya que en la Universidad del Valle de Guatemala se cuenta con una fresadora CNC para la fabricación de PCBs, pero hay que cumplir con requerimientos específicos para la fabricación, entre estos, está el ancho recomendado de *track*, el cual es de 0.508mm. Esto nos limita a la hora de diseñar, ya que se dificulta más el conectar los componentes, ya que el espacio es limitado y los *tracks* pueden llegar a ser muy anchos. También cuenta con otros requerimientos específicos, pero esos no dan mayor problema. Otra limitante es la búsqueda de algunos componentes, ya que no todos son distribuidos dentro del país.

Cuando se habla de las limitantes en el tema de *software*, hablamos de *WiringPi*, ya que es una librería a la cual se le dejó de dar soporte en la página oficial, por lo que fue de gran dificultad descargarla para ser utilizada, se tuvo que buscar en otras fuentes que lo guardaron y que al día de hoy brindan apoyo.

Antes de realizar un proyecto de este tipo, es necesario tomar en cuenta que, debido a las limitantes de fabricación, se debe hacer un diseño no muy complejo, ya que a la hora de soldar componentes, puede llegar a ser un poco complicado. También considerar qué lenguaje de programación usar para la simplificación de tareas.

CAPÍTULO 6

Marco teórico

6.1. *Raspberry Pi 3B+*

La *Raspberry Pi 3B+* es el último producto de la línea de ordenadores de placa única *Raspberry Pi 3*, posteriormente se desarrolló el modelo de la linea *Raspberry Pi 4* . Este cuenta con un procesador de 4 núcleos y 64 bits que funciona a 1.4GHz. La unidad de procesamiento central que utiliza este ordenador de placa única es un BCM2837B0, el cual es un chip de Broadcom utilizado en los modelos de *Raspberry Pi* A+ y B+. A diferencia del BCM 2837, el BCM 2837B0 es un 17 % más rápido, tiene un empaquetado diferente y contiene más disipadores de calor. [5]

El chip está fabricado con base en una arquitectura ARM (Advanced Reduced instruction set computing machine) versión 8, lo cual le permite un consumo bajo de energía, menos generación de calor y mayor velocidad. [6]

En el Cuadro 1 se pueden observar las características principales del *Raspberry Pi 3B+* y las diferencias con modelos anteriores. En el Cuadro 2 se pueden observar las características del *Raspberry Pi 3B+* y las diferencias con los modelos posteriores.

	Pi A+	Pi B+	Pi 2 B	Pi 3 B	Pi 3 B+
Dimensiones	66 x 56 x 14mm	85 x 56 x 17mm	85 x 56 x 17mm	85 x 56 x 17mm	85 x 56 x 17mm
SoC	BCM2835	BCM2835	BCM2836	BCM2837	BCM2837B0
Núcleo del procesador	ARM11	ARM11	ARM Cortex-A7	ARM Cortex-A53	ARM Cortex-A53
Capacidad de procesamiento	700MHz	700MHz	900MHz	1.2GHz	1.4GHz
Memoria	256MB	512MB	1GB	1GB LPDDR2	1GB LPDDR2
Puertos	1x USB 2.0	4x USB 2.0 1x 10/100 Ethernet	4x USB 2.0 1x 10/100 Ethernet	4x USB 2.0 1x 10/100 Ethernet	4x USB 2.0 1x 10/100 Ethernet
GPIO	40	40	40	40	40

Cuadro 1: Comparación con modelos anteriores a *Raspberry Pi 3B+*

	Pi 3B+	Pi 4B
Dimensiones	85 x 56 x 17mm	85 x 56 x 17mm
SoC	BCM2837B0	BCM2711
Núcleo del procesador	ARM Cortex-A53	ARMv8 Cortex-A72
Capacidad de procesamiento	1.4GHz	1.5MGHz
Memoria	1GB LPDDR2	8GB LPDDR4
Puertos	4x USB 2.0 1x 10/100 Ethernet	2x USB 2.0 2x USB 3.0 1x Gigabit Ethernet
GPIO	40	40

Cuadro 2: Comparación con modelos posteriores a *Raspberry Pi 3B+*

Este ordenador de placa única funciona con un sistema operativo basado en *Debian*, llamado *Raspberry Pi OS*, el cual está específicamente optimizado para el hardware de *Raspberry Pi*. El sistema operativo cuenta con más de 35,000 paquetes, que permiten una fácil instalación. [7]

Cuando se habla de *Debian*, se hace referencia a una distribución de Linux, la cual tiene la ventaja de incluir un sistema de gestión de paquetes que otorgan al administrador total control sobre los paquetes instalados. La distribución le da muchas facilidades al usuario cuando se habla de instalación de paquetes y actualización de *software*. [8]

6.2. *Universal Asynchronous Receiver-Transmitter* (UART)

El *Universal Asynchronous Receiver-Transmitter* es uno de los protocolos de comunicación más utilizados por sistemas embebidos, computadoras y microcontroladores para poder

tener comunicación entre dispositivos. Cuando se habla de UART, se hace referencia a una comunicación por *hardware* que utiliza comunicación serial asíncrona con velocidad configurable. Que la comunicación sea asíncrona implica que no hay una señal de reloj que sincronice los bits de salida del dispositivo transmisor con el receptor. [9]

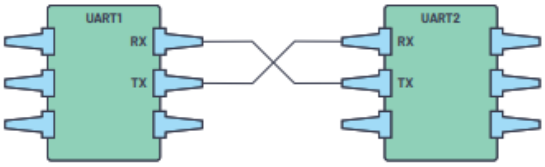

Figura 12: Esquema de comunicación UART entre dispositivos.

El protocolo cuenta con dos señales principales:

- *Transmitter*(Tx)
- *Receiver*(Rx)

El transmisor del UART está conectado a un bus de datos que envía información de forma paralela. A partir de esto, el transmisor enviará de forma serial, bit por bit, al receptor del otro dispositivo. Para que la comunicación sea exitosa, es indispensable establecer el *Baud Rate* o número de unidades de señal por segundo (en este caso bits por segundo) en ambos dispositivos, para que ambos logren sincronizar la transmisión y recepción de datos, ya que al ser asíncrono, no utilizan ninguna señal de reloj.[9]

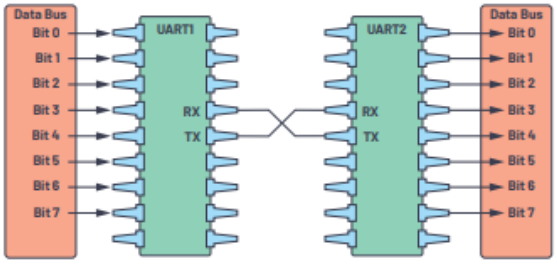

Figura 13: Demostración de bus de datos y UART

Para que la transmisión de datos sea posible, se crean paquetes que contienen toda la información y ciertos indicadores. En la Figura 14 se puede observar la estructura de un paquete UART.[9]

Figura 14: Paquete UART

El primer bit del paquete es el *Start Bit*, el cual indica el inicio de la comunicación. El puerto de transmisión siempre se mantiene encendido cuando no se está transmitiendo

ningún tipo de información, pero una vez el puerto hace una transición de *HIGH* a *LOW* por un ciclo de reloj, el receptor detecta la transición e interpreta que debe empezar a leer los bits (a la frecuencia de *Baud Rate*) en el *Dataframe*, el cual es el siguiente bloque del paquete.

El *Dataframe* contiene la información a transferir. Este puede ser de 5 a 8 bits de largo si se utiliza un bit de paridad. Si no se utiliza un bit de paridad, el *dataframe* puede ser de hasta 9 bits de largo. En la mayoría de los casos de comunicación UART, el transmisor envía primero el bit menos significativo, hasta llegar al más significativo.[9]

El bit de paridad describe si el número de bits que están en *HIGH* es par o impar, con el fin de verificar que la transmisión haya sido sin errores. Si el bit de paridad está en 1, el número de bits en *HIGH* debería ser impar y si el bit de paridad está en 0, el número de bits en *HIGH* debería ser par. Si el bit de paridad no coincide con el número de bits en *HIGH* recibidos, significa que hubo un error en la comunicación.[9]

Los últimos bits del paquete son los *Stop Bits* los cuales detienen la comunicación por un tiempo determinado una vez ya se envió el paquete deseado, brindando un intervalo de tiempo en donde el receptor se prepara para recibir más información sin mezclarla con la anterior.[9]

El protocolo de comunicación UART tiene muchas ventajas a la hora de querer transferir información por solamente dos canales, con la certeza de que puede identificar errores y enviar datos a grandes velocidades de transmisión. [9]

6.3. *Serial Peripheral Interface* (SPI)

El *Serial Peripheral Interface* es un protocolo utilizado para enviar información entre microcontroladores y periféricos tales como: *Shift registers*, sensores y memorias.[10] Se trata de una interfaz serial síncrona que resuelve el problema de los protocolos asíncronos: la transmisión de datos erróneos por falta de sincronización. Al ser un protocolo síncrono, implica que depende de una señal de reloj. [11]

En pocas palabras, el SPI es un bus de datos síncrono que envía datos por diferentes puertos y utiliza una señal de reloj para sincronizarse entre dispositivos. La señal de reloj es una señal cuadrada que le indica al receptor cuando muestrear los bits enviados, esto puede ser en los flancos positivos o en los flancos negativos, depende de los periféricos utilizados. Esto se observa en la Figura 15. [10]

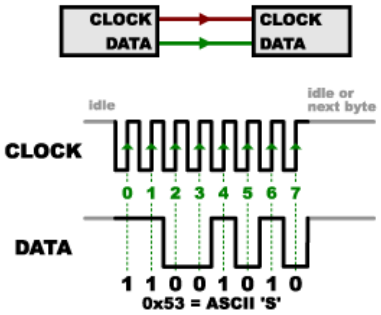

Figura 15: Lectura de datos en SPI

En SPI, la señal de reloj es generada únicamente por un dispositivo, llamado *Master* y el dispositivo que la recibe es denominado como *Slave*. Solo puede haber un *Master*, pero si puede haber más de un *Slave*.

Este protocolo de comunicación utiliza un máximo de cuatro líneas de señal: SCK (*Serial Clock*), SS(*Slave Select*), MOSI(*Master Out Slave In*) y MISO (*Master In Slave Out*). El SCK es la señal de reloj generada por el *Master* y recibida por el *Slave* para poder sincronizar ambos dispositivos. [11]

Cuando se envían datos del *Master* a cualquier *Slave*, se transmiten por el MOSI y si el *Master* requiere de una respuesta por parte del *Slave*, este envía los datos por el MISO, mientras el *Master* sigue generando la señal de reloj. Esta señal de reloj tiene una cantidad de pulsos exacta, ya que el *Master* envía una instrucción específica al *Slave* y este da una respuesta con una cantidad de bits ya definida.

Para que el *Slave* pueda enviar información de regreso, el *Master* activar/desactivar la señal del SS, ya que esta siempre se mantiene en *HIGH*, manteniendo al *Slave* desactivado. Una vez se apague el bit de SS, el *Slave* será activado, leerá una instrucción y enviará información de regreso. En la Figura 16 se puede observar un diagrama de como funciona el SPI. [10]

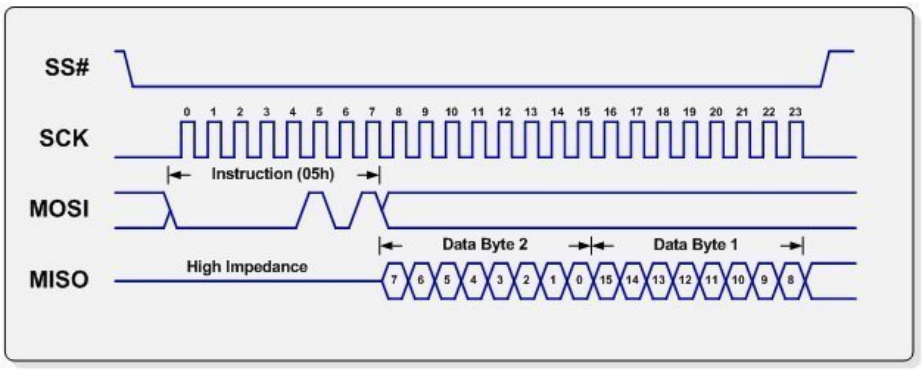

Figura 16: Comunicación SPI

La comunicación SPI puede ser con únicamente un *Slave* o múltiples *Slaves* y un *Master*.

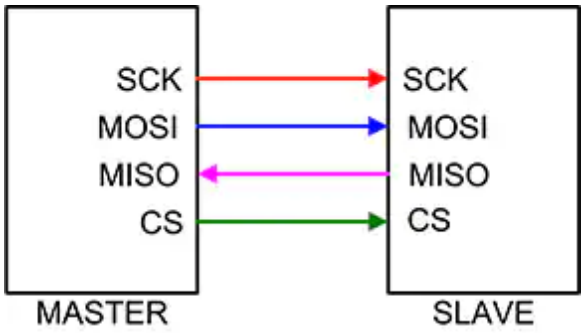

Figura 17: Comunicación SPI con un *Master* y un *Slave*

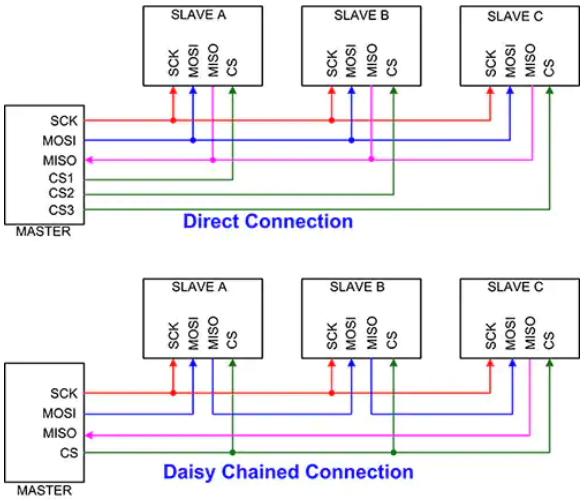

Figura 18: Comunicación SPI con un *Master* y múltiples *Slaves*

Hay dos casos en donde se utilizan múltiples *Slaves*. El primero consiste en activar uno a la vez, por medio del SS, lo que hace un periférico completamente independiente del otro. El otro caso consiste en activar todos los periféricos de manera simultánea y recibiendo la información de todos. Esto se puede observar en la Figura 18. A esa conexión se le llama *Daisy Chain*, el cual tiene la peculiaridad que todos los esclavos son controlados únicamente por un *Slave Select*. El primer esclavo es el único dispositivo que recibe comandos directamente del maestro, el resto de esclavos recibe comandos provenientes de la salida del dispositivo anterior. Una vez, el SS se mantenga en *LOW*, la información enviada desde el maestro se propagará en toda la cadena de esclavos hasta que cada uno de reciba el comando apropiado.[10]

6.4. *Inter-Integrated Circuit* (I^2C)

El I^2C es un protocolo de comunicación síncrona con un bus serial, multimaestro y multiesclavo. Este protocolo se utiliza principalmente para conectar periféricos de bajas velocidades a procesadores y microcontroladores.

El I^2C se podría decir que mezcla al UART y al SPI, ya que utiliza únicamente dos canales de comunicación (SDA y SCL) y es una comunicación síncrona que funciona con *Masters* y con *Slaves*.

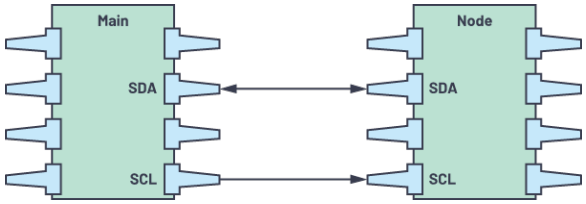

Figura 19: Esquema de comunicación I^2C entre dispositivos

El canal de comunicación SDA (*Serial Data*) es para que el dispositivo principal y un nodo puedan enviar y recibir información. El canal SCL (*Serial Clock*) es el que genera la señal de reloj distribuida a todos los nodos. Ambos canales deben tener una resistencia de *Pull-up* para funcionar, ya que la comunicación se realiza alternando el estado de los canales (*HIGH/LOW*). El estado de los canales cuando no hay comunicación siempre debe ser *HIGH*.[12]

El protocolo transfiere información en *frames*: la condicion inicial, el direccionamiento, bit de lectura/escritura, un bit de reconocimiento/no reconocimiento, la información a enviar, un bit de reconocimiento/no reconocimiento y un bit de condición de parada.

Figura 20: Mensaje de I^2C

La *Start Condition* ocurre al inicio de la transmisión cuando es iniciado por el dispositivo principal. El canal SDA cambia de *HIGH* a *LOW* antes que el canal SCL haga la misma transición.

El direccionamiento contiene una secuencia de 7 o 10 bits, dependiendo de la disponibilidad. El dispositivo principal envía la dirección del nodo con el que quiere establecer comunicación, cada nodo compara la dirección enviada con la dirección propia y si coinciden, el nodo envía un bit ACK o de reconocimiento al dispositivo principal. Si las direcciones no coinciden, el canal SDA permanece en *HIGH* y el nodo permanece inactivo. El direccionamiento también contiene un bit para indicar al nodo si debe recibir o enviar información. [12]

Cada *frame* en un mensaje es seguido de un bit de ACK/NACK (reconocimiento/no reconocimiento) que informa al transmisor si se recibió o no la información.

Una vez el dispositivo principal recibe el ACK del direccionamiento, este lo interpreta como un indicador de que ya se puede enviar la información o la instrucción a los nodos. El *Data frame* tiene un tamaño de 8 bits y siempre es enviado con el bit más significativo de primero. Una vez se envían todos los bits, el dispositivo principal queda a la espera de el bit de ACK/NACK. [12]

Una vez toda la información fue enviada, el dispositivo principal envía una condición de parada al nodo para que pare la transmisión de datos.

Debido a que I^2C utiliza direccionamiento, es posible utilizar múltiples nodos. Cuando se utiliza una dirección de 7 bits, se tienen 128 direcciones únicas disponibles y cuando se utiliza una dirección de 10 bits, se tienen 1024 direcciones únicas. [12]

En el caso de que se quieran utilizar múltiples dispositivos principales y múltiples nodos, los dispositivos leen el estado del SDA, ya que si este se encuentra en *LOW*, significa que otro dispositivo principal está controlando el bus de datos, de lo contrario, si se encuentra en *HIGH*, significa que el bus de datos está libre. [12]

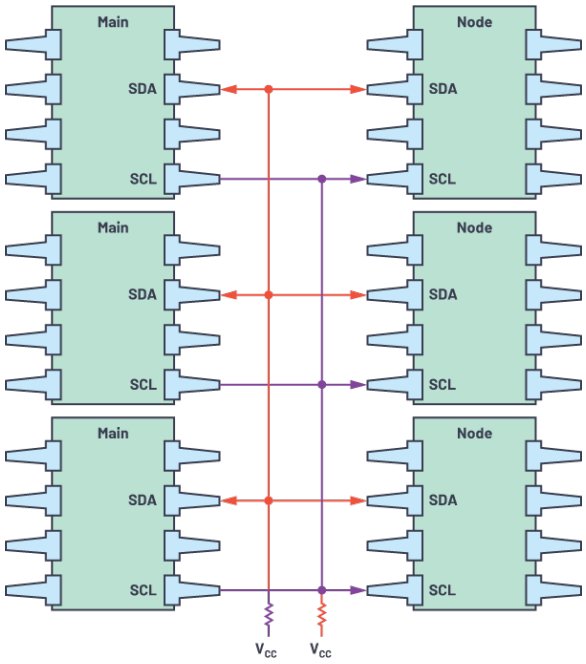

Figura 21: Múltiples nodos y múltiples principales en I^2C

6.5. *Pulse-Width Modulation*

PWM (*Pulse-Width Modulation*) es una técnica de modulación digital utilizada para transmitir una señal periódica, a la cual se le pueda modificar el ciclo de trabajo para controlar la cantidad de energía que se envía a una carga.

El ciclo de trabajo de una señal es la relación entre el tiempo encendido de la señal y su período. Esto se puede observar en la Ecuación 1, en donde Th es el tiempo encendido y Tw el período de la señal. [13]

$$D = 100 * \frac{Th}{Tw} \tag{1}$$

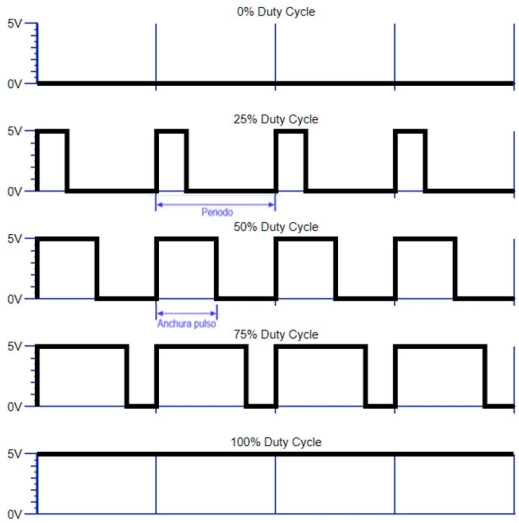

Figura 22: Diferentes ciclos de trabajo de una PWM

Para calcular la frecuencia de la PWM en *Raspberry Pi*, es necesario tomar en cenca la Ecuación 2.

$$\frac{1}{f} = 19.2MHz * \frac{pwmclock}{pwmrange} \tag{2}$$

6.6. *Inter-IC Sound* (I^2S)

El I^2S es un protocolo de comunicación síncrono con un bus serial, multimaestro y multiesclavo. Este protocolo tiene el propósito de facilitar el desarrollo de la electrónica de audio por medio de una interfaz estandarizada para transmitir datos digitales entre dispositivos de conversión analógica a digital o viceversa, filtros digitales y otros circuitos utilizados en sistemas de audio. [14]

El bus serial puede manejar únicamente datos de audio. Este tiene únicamente 3 líneas que consisten en una línea de dos canales multiplexados (SD), un *Word Select*(WS) y un *clock*(SCK).

El transmisor y el receptor tienen la misma señal de reloj para la transmisión de datos, el transmisor debe generar el bit de SCK, una señal de WS y la SD. Eso es en caso de un sistema simple, pero en el caso de un sistema más complejo, hay más de un transmisor y más de un receptor, lo cual dificulta definir al maestro que controla las señales. En este tipo de sistemas, se cuenta con un dispositivo que funciona como el maestro, el cual controla el flujo de datos entre dispositivos. [14]

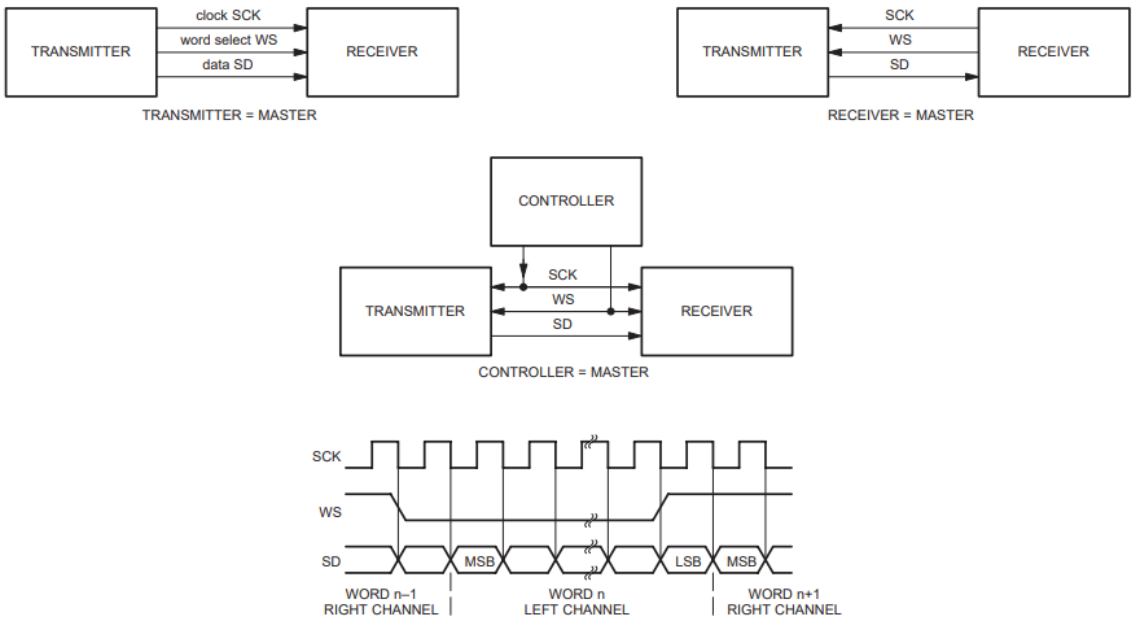

Figura 23: *Inter-IC Sound*

En la línea SD (*Serial Data*) se transmiten los datos con el bit más significativo de primero y en complemento a 2. Se transmite de primero el bit más significativo porque tanto el receptor como el transmisor pueden enviar datos con ancho distinto. Debido a esto, no es necesario que el transmisor sepa cuantos bits debe manejar el receptor y tampoco el receptor debe saber cuantos bits están siendo transmitidos.

En la línea WS (*Word Select*) se indica el canal en donde se está transmitiendo. Si el bit tiene un valor de 0, se está transmitiendo por el canal 1, mientras que si el bit tiene un valor de 1, se está transmitiendo por el canal 2.

Para transmitir los datos analógicos, se deben modular de cierta manera para obtener una representación binaria de la misma. En I^2S se utiliza el PCM (*Pulse Code Modulation.*

PCM (*Pulse-Code Modulation*) es una técnica de modulación digital utilizada para convertir señales analógicas a secuencias binarias. Este se utiliza para producir una serie de números o dígitos en forma binaria. En esta técnica, una secuencia de pulsos representa la señal del mensaje. Esta señal representa tiempo y amplitud de una señal analógica. [15]

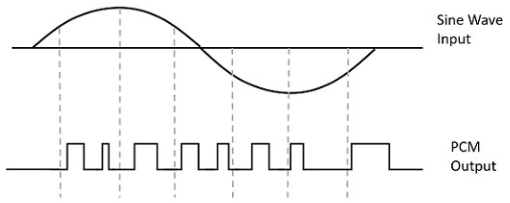

Figura 24: *Pulse code modulation*

El proceso de PCM se realiza a través de los siguientes pasos:

- **Filtro de paso bajo**: El filtro de baso bajo ayuda a remover las componentes de la señal analógica de entrada que son de frecuencias altas. Estas componentes tienen una frecuencia mayor a la frecuencia más alta de la señal de mensaje.
- ***Sampler***: Este permite recopilar los datos muestreados en cualquier momento de la señal de mensaje, de tal manera de que sea posible reformar la señal original. La tasa de muestreo debe ser mayor que el componente de frecuencia mas alta de la señal de mensaje.
- ***Quantizer***: Este ayuda a minimizar el error mediante un proceso llamado cuantificación. Esto permite que se descarten los bits innecesarios y ayuda a comprimir los valores obtenidos.
- ***Encoder***: Se utiliza para digitalizar la señal analógica.
- **Repetidor regenerativo**:Se utiliza para compensar las perdidas de la señal y también para regenerar la misma. Incluso ayuda a amplificar la señal.
- ***Decoder***: El *decoder* ayuda a formar la señal original decodificando la señal muestreada.
- **Filtro de reconstrucción**: Este filtro ayuda a reconstruir la señal analógica. [15]

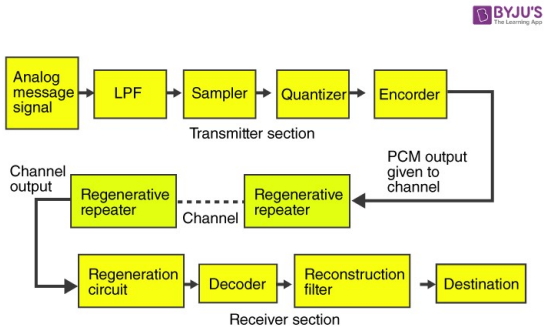

Figura 25: *Diagrama de bloques de Pulse code modulation*

6.7. *Printed Circuit Boards*

Un PCB (*Printed Circuit Board* o placa de circuito impreso) es un circuito cuyos componentes y conductores están dentro de una estructura mecánica. Los conductores o cables constan de trazas de cobre, terminales, disipadores de calor o conductores planos. La estructura mecánica es fabricada con un material aislante entre capas de material conductivo, también es cubierta con una máscara de soldadura no conductora. [16]

Un PCB está compuesto de cuatro materiales principales: *Silkscreen, Soldermask, cobre y substrato (FR4).*

El sustrato o material base, usualmente es fibra de vidrio, lo cual le da grosor y rigidez al circuito. El cobre es un material que se adhiere a ambos lados del substrato en placas de 2 capas. Usualmente los PCB tienen una onza de cobre por pie cuadrado. El *Soldermask* es la capa encima del cobre. Esta le da el color verde a la placa y tiene la función de aislar las trazas de cobre para evitar contactos accidentales con otros metales. Por último se tiene el *Silkscreen*, el cual es un material que se agrega encima del *Soldermask* para poder agregar letras, nombres y símbolos al PCB. [17]

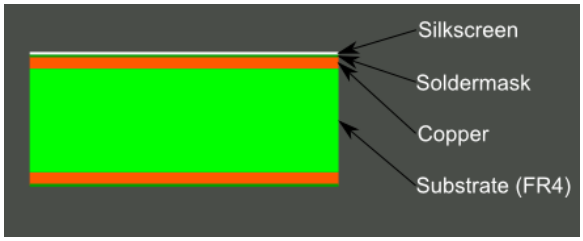

Figura 26: Capas de un PCB

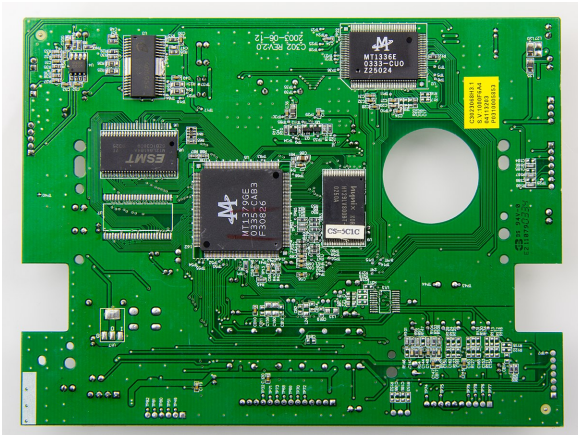

Figura 27: Placa de circuito impreso

En el diseño de PCBs, se cuenta con una herramienta que nos permite calcular la corriente máxima en la traza,la impedancia, etc. Esta se conoce como calculadora de trazas de PCB.[18] Además, siempre se debe seguir el estándar IPC-2221 *Generic Standard on Printed Board Design*, el cual establece todos las carácteristicas genéricas que deberían de tomarse en cuenta a la hora de diseñar un PCB.

La calculadora se basa en expresiones específicas para calcular parámetros.[19]

La expresión para calcular el área se representa en la Ecuación 3:

$$A = (\frac{I}{K * T_{rise}^{b}})^{\frac{1}{c}} \tag{3}$$

Donde:

- A = Área
- b = constante 0.44
- K = constante, 0.024 para capas internas y 0.048 para capas externas
- Trise = Aumento de la temperatura
- c = constante 0.725

La expresión para calcular el ancho de *track* se representa en la Ecuación 4:

$$W = \frac{A}{t * 1.378} \tag{4}$$

Donde:

- t = *thickness* o grosor
- A = Área
- W = ancho o *Width*

Como se mencionó anteriormente, los cálculos se basan en el estándar IPC-2221, en donde mencionan los requerimientos más importantes, entre estos esta la sección transversal y el ancho de pista para un PCB. En la Figura 28 se muestra una gráfica de corriente respecto a la sección transversal de la pista. Esta gráfica se utiliza para determinar la capacidad de corriente que tiene el conductor según su sección transversal a diferentes cambios de temperatura. [20]

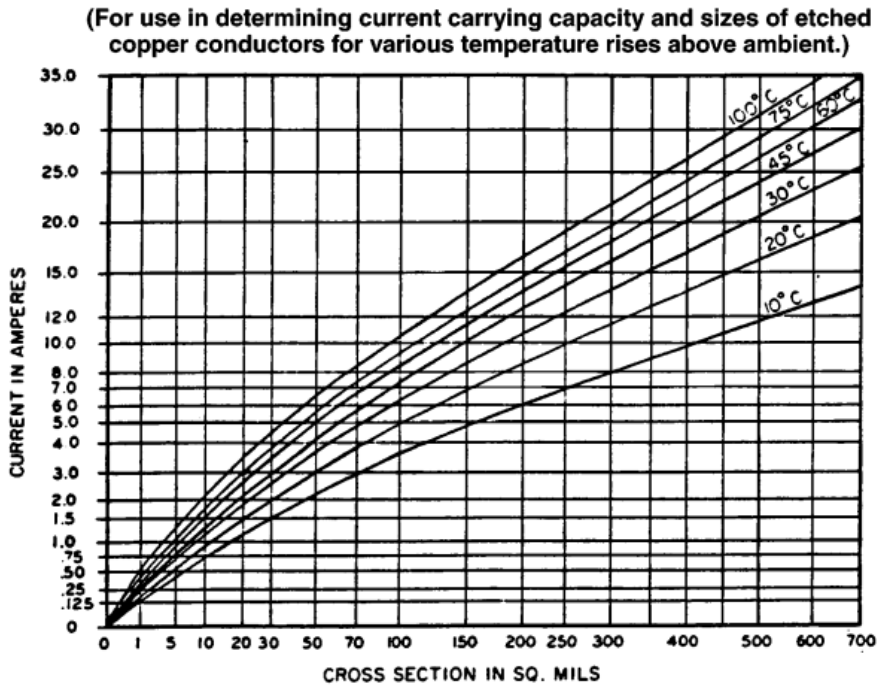

Figura 28: Corriente de sección transversal según estándar IPC-2221 para conductores externos

Una vez se tiene la sección transversal, el estándar IPC-2221 también menciona con respecto al cálculo del ancho de pista según la sección transversal. En la Figura 29 se muestra una gráfica de ancho respecto a sección transversal de la pista, en donde podemos determinar que ancho debe tener la pista según el resultado de la gráfica anterior, donde se estimó la sección transversal.[20]

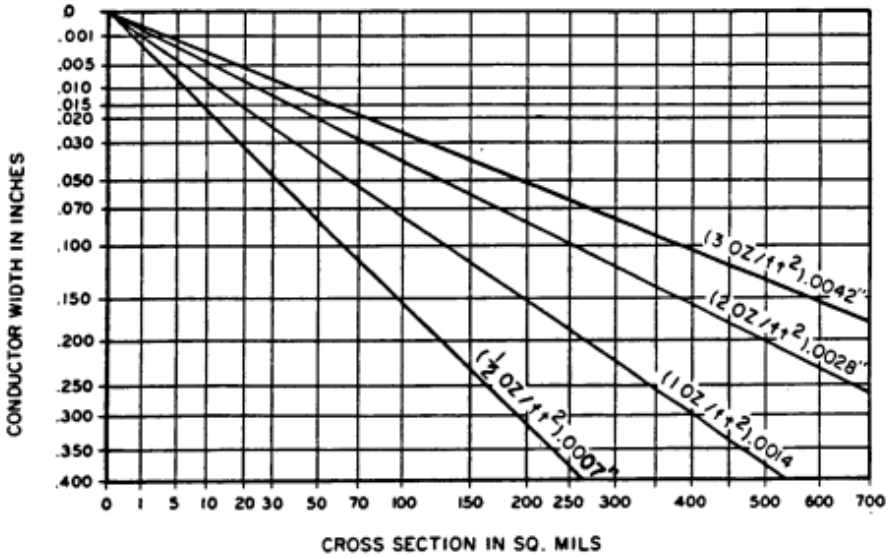

Figura 29: Ancho de conductor respecto al sección transversal según estándar IPC-2221

Estos son algunos de los parámetros que describe el estándar mencionado. Es importante seguir estas regulaciones para fabricar una placa funcional.

6.8. *Wiring-Pi*

WiringPi es una librería de acceso GPIO escrita en C para los SoC (System on a chip) BCM2835, BCM2836 y BCM2837 utilizados en los *Raspberry Pi*.

La librería está diseñada para que le sea familar a las personas que han utilizado el sistema de "*wiring*"de arduino y esta destinado principalmente a programadores experimentados en C/C++. [21]

WiringPi incluye una utilidad de CMD para programar y configurar los pines de propósito general (GPIO). Se puede utilizar esto para encender y apagar pines e incluso para controlarlos desde *scripts* en el *shell* [21].

Este utiliza un *pinout* específico observado en la Figura 30

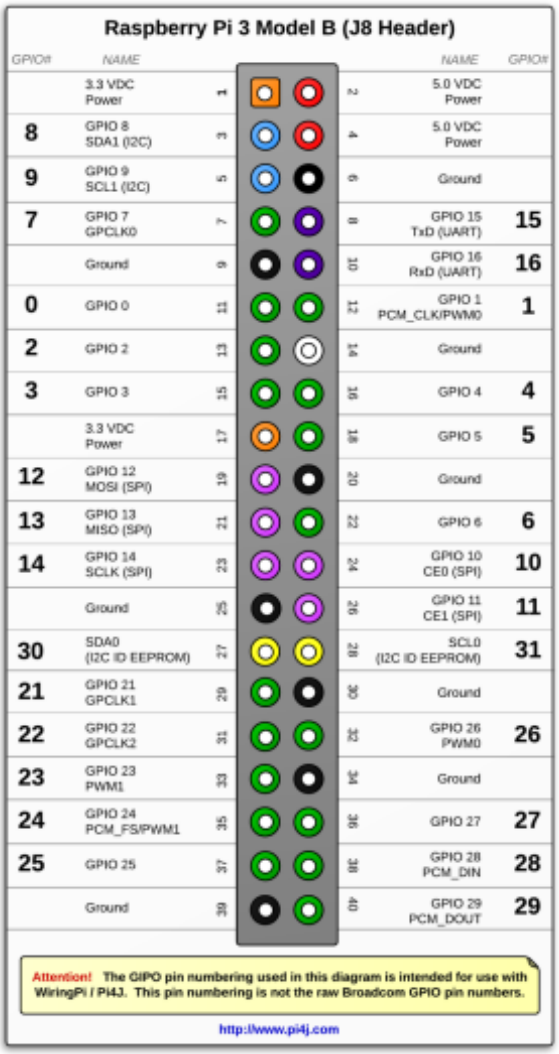

Figura 30: *pinout*de *WiringPi*

6.9. Driver A4988

El A4988 es un controlador de motor *Stepper*, el cual está diseñado para operar motores bipolares en paso completo, medio, cuarto, octavo y decimosexto con una capacidad de accionamiento de salida de hasta 35 V y ±2 A. [22]

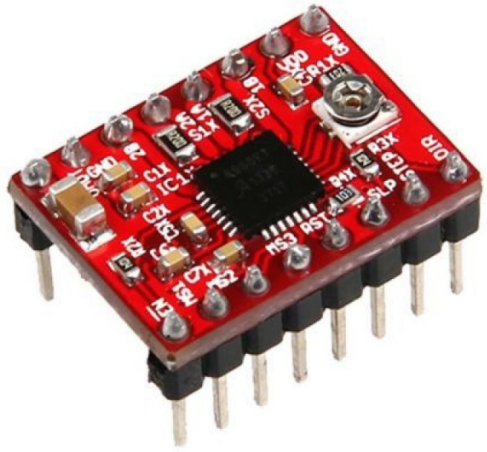

Figura 31: Driver A4988 para motor *Stepper*

6.9.1. Operación del dispositivo

El controlador contiene un traductor que simplifica la operación del mismo. Esto permite mover el motor un micropaso ingresando un pulso en la entrada *STEP* y modificar la dirección facilmente en la entrada *DIR*. Las corrientes de la salida están reguladas con una un circuito de control de PWM con tiempo de inactividad fijo. Como se obeserva en la 5 La corriente de cada salida es establecida por el voltaje de referencia VREF, el DAC controlado por el traductor y el valor de las resistencias RS1 y RS2. [22]

$$Imax = \frac{VREF}{8 * RS} \tag{5}$$

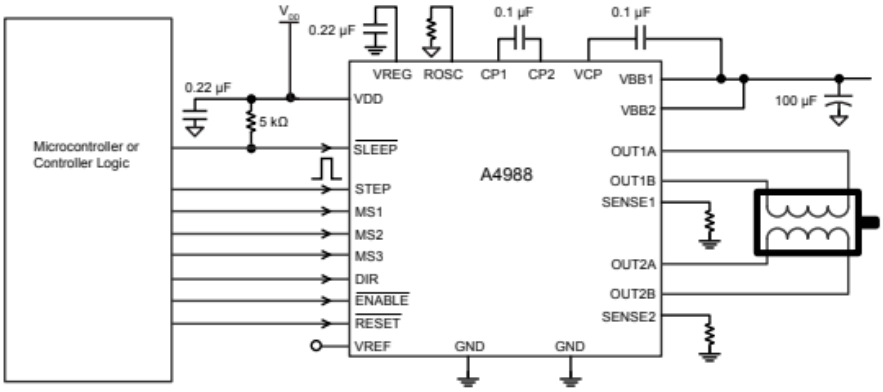

Figura 32: Diagrama de aplicación Driver A4988 para motor *Stepper*

La resolución de los micropasos depende del voltaje en las entradas lógicas MSx como se

muestra en la Figura 33. Cuando se cambia el modo de paso, no se activa hasta el siguiente flanco positivo del pulso de *STEP*.

MS1	MS2	MS3	Microstep Resolution	Excitation Mode
L	L	L	Full Step	2 Phase
H	L	L	Half Step	1-2 Phase
L	H	L	Quarter Step	W1-2 Phase
H	H	L	Eighth Step	2W1-2 Phase
H	H	H	Sixteenth Step	4W1-2 Phase

Figura 33: Diagrama de aplicación Driver A4988 para motor *Stepper*

La dirección del motor depende de la entrada lógica DIR, en donde según el valor (0 o 1) el motor gira a la izquierda o a la derecha y para habilitar el controlador se debe dejar en 0 la entrada del pin *ENABLE*.[22]

6.10. Micrófono digital SPH0645LM4H-B

El SPH0645LM4H-B es un micrófono miniatura de bajo consumo con una salida digital para I^2S. El dispositivo consiste en un sensor acústico de alto rendimiento, un conversor analógico a digital serial y una interfaz para ajustar la señal a un estándar I^2S de 24 bits. La interfaz I^2S permite conectar directamente a procesadores digitales y microcontroladores. [23]

El micrófono consiste de 7 etapas de funcionamiento:

- Bomba de carga(Rojo)
- Transductor MEMS
- Amplificador(Amarillo)
- Conversor Sigma-Delta(Verde)
- Decimador(Naranja)
- Filtro de paso bajo(Turquesa)
- Control Tri-estado(Gris)

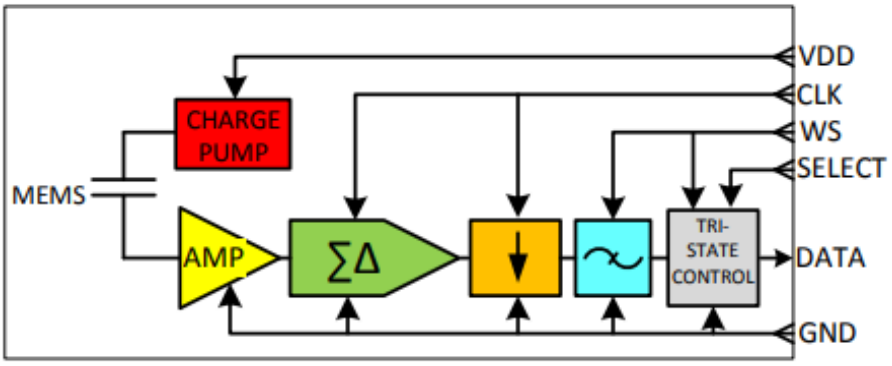

Figura 34: Diagrama de aplicación micrófono SPH0645LM4H-B

6.10.1. Operación del dispositivo

La bomba de carga aumenta VDD a un nivel adecuado para alimentar el transductor MEMS, para que este transforme una entrada SPL a un voltaje. Luego el amplificador aumenta la ganancia de la señal de salida del transductor y proporciona la suficiente potencia para tener una señal estable para el convertidor Sigma-Delta.

El convertidor Sigma-Delta convierte la señal analógica amplificada en una señal PDM (*Pulse Density Modulated*). Después, el decimador convierte la señal PDM a una señal PCM de múltiples bits, para que luego la señal elimine altas frecuencias en el filtro de paso bajo y se mejore el ancho de banda.

Por último, el control tri-estado usa el estado de los pines WS y SELECT para determinar si el pin de DATA esta activo o en tri-estado. [23]

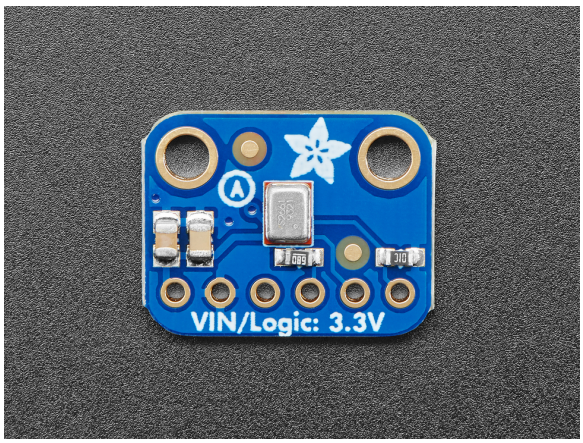

Figura 35: Micrófono digital SPH0645LM4H-B

6.11. Amplificador de potencia MAX 98357

El MAX98357x es un amplificador clase D con entrada digital PCM que provee desempeño de un amplificador clase AB. La interfaz de audio digital es altamente flexible, ya que soporta datos de I^2S y muestras entre 8kHz y 96kHz para todos los formatos admitidos. El módulo puede ser configurado para producir un canal izquierdo o derecho, operando con datos de 16, 24 y 32 bits con ganancias de 3dB, 6dB, 9dB, 12dB y 15dB. [24]

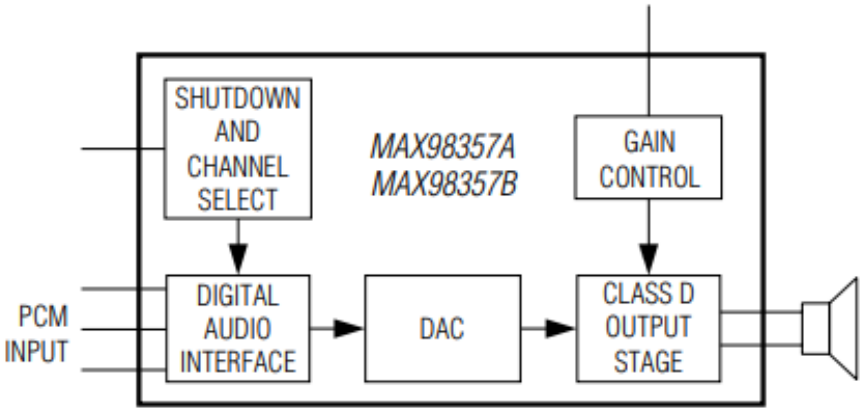

Figura 36: Diagrama de aplicación para amplificador MAX 98357

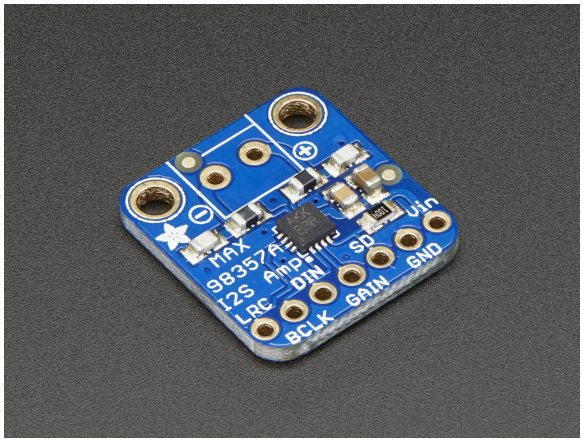

Figura 37: Amplificador MAX 98357

6.12. Conversor analógico-digital MCP 3002

El MCP 3002 es un convertidor analogico-digital con un circuito incorporado de *Sample and hold*. La comunicación con el dispositivo se realiza utilizando una interfaz compatible con el protocolo SPI. Este cuenta con dos canales analógicos, un *Chip-Select*, un *Clock* Serial, una entrada de datos seriales y una salida de datos seriales. [25]

El *Chip-Select* se utiliza para iniciar la comunicación con el dispositivo cuando está en 0 lógico. El 1 lógico se utiliza para poner el dispositivo en modo de espera.

Los pines de datos seriales son principalmente para configurar el dispositivo, iniciar la conversión y mostrar los resultados de la conversión. [25]

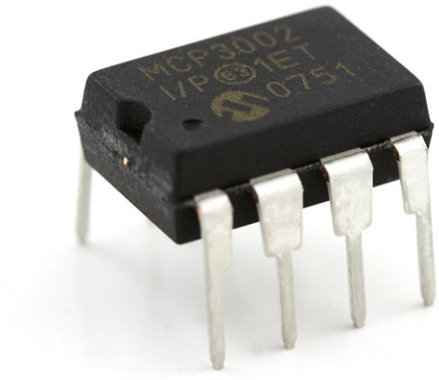

Figura 38: ADC MCP 3002

6.13. Conversor digital-analógico MCP 4921

El MCP 4921 es un convertidor digital-analógico (DAC) que ofrece alta precisión y rendimiento con bajo ruido para aplicaciones industriales en las cuales se requiera la calibración de señales. [26]

El dispositivo está diseñado para utilizarse con el protocolo SPI, es decir que de forma serial, se configura el dispositivo. Este cuenta con un *Chip-Select*, un *Clock* serial, una entrada de datos seriales, voltajes de referencia y las salidas de la conversión.

El *Chip-Select* se utiliza para iniciar la comunicación con el dispositivo cuando esta en 0 lógico. El 1 lógico se utiliza para poner el dispositivo en modo de espera.[26]

Los pines de datos seriales son principalmente para configurar el dispositivo, iniciar la conversión y mostrar los resultados de la conversión.

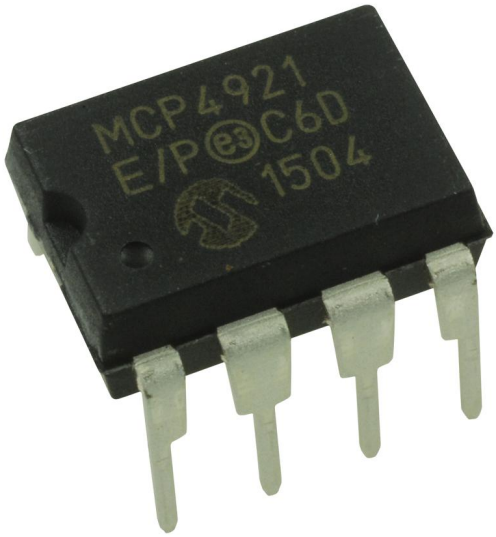

Figura 39: DAC MCP 4921

CAPÍTULO 7

Diseño de la plataforma

7.1. Librerías

Para poder empezar a realizar el diseño, fue necesario representar a los componentes con un símbolo para el esquemático y con un *Footprint* para el PCB.

Todo esto se realizó con la creación de una librería, en donde están todos los componentes a utilizar, su *Footprint* y su modelo 3D. Algunos *Footprints* y Modelos 3D fueron creados y otros fueron extraídos del sitio: https://www.snapeda.com. Para observar los modelos revisar anexos 13.2, 13.3 y 13.4.

7.2. Esquemáticos

Una vez se completó la librería, se crearon los esquemáticos, en donde se realizaban las respectivas conexiones entre componentes para poder definir *nets* y tener un documento que describa como está conectado todo lo de la plataforma. En la Figura 40 se observa el esquemático completo, en donde se utilizaron los símbolos creados anteriormente. Para mayor orden, se segmentó el esquemático para identificar con facilidad para que es cada bloque. En el anexo 13.5 se puede observar más de cerca todos los bloques del esquemático de forma individual.

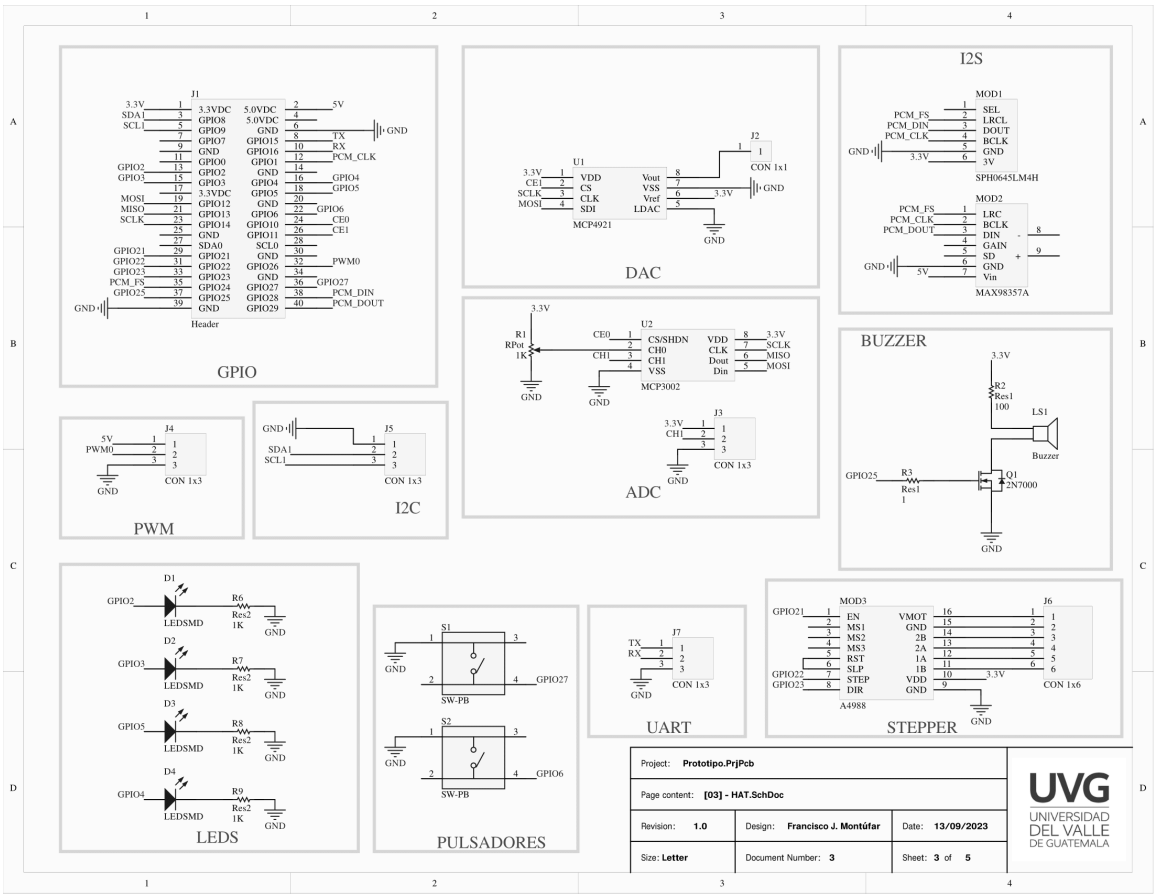

Figura 40: Esquemáticos de la plataforma

7.2.1. *Layout*

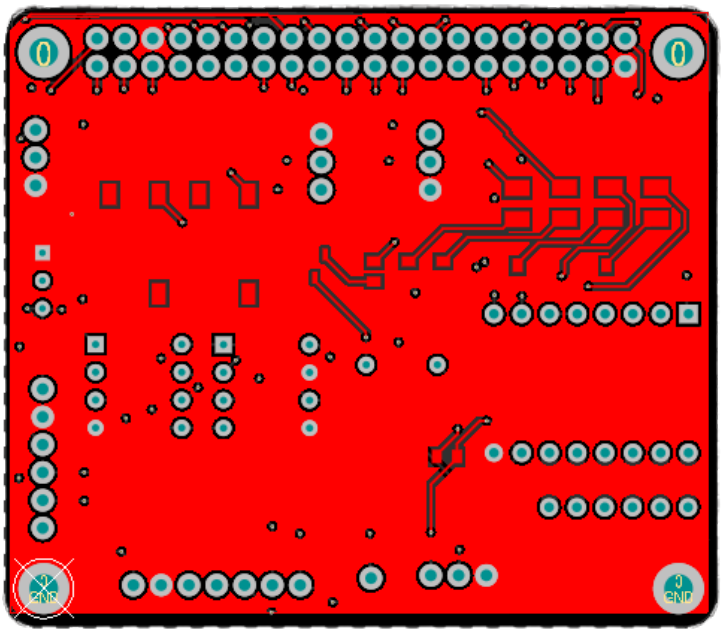

Figura 41: *Layout* de la *Top Layer* de la plataforma

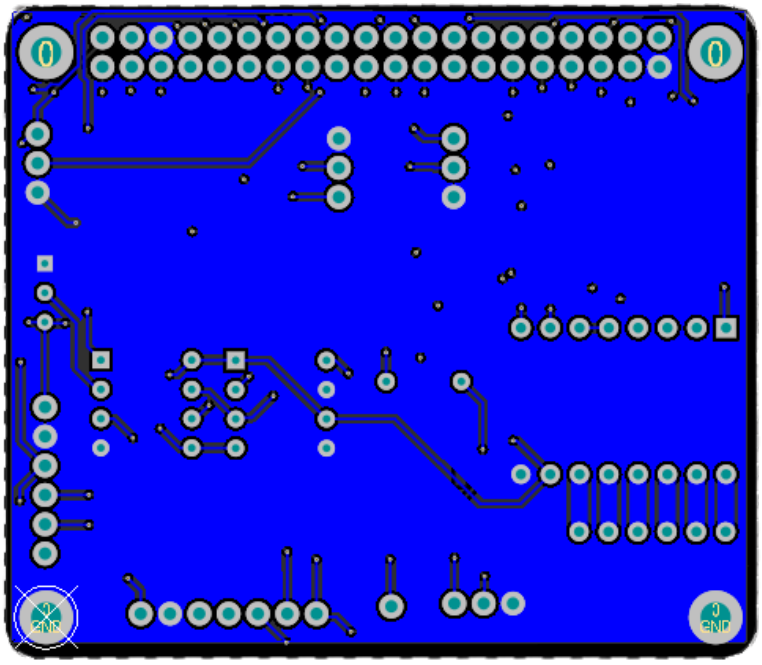

Figura 42: *Layout* de la *Bottom Layer* de la plataforma

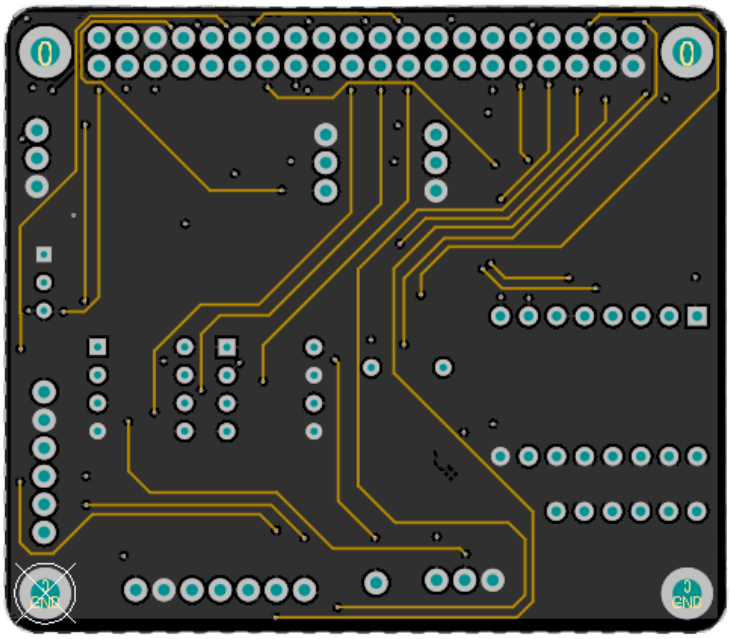

Figura 43: *Layout* de la *Layer* interna 1 de la plataforma

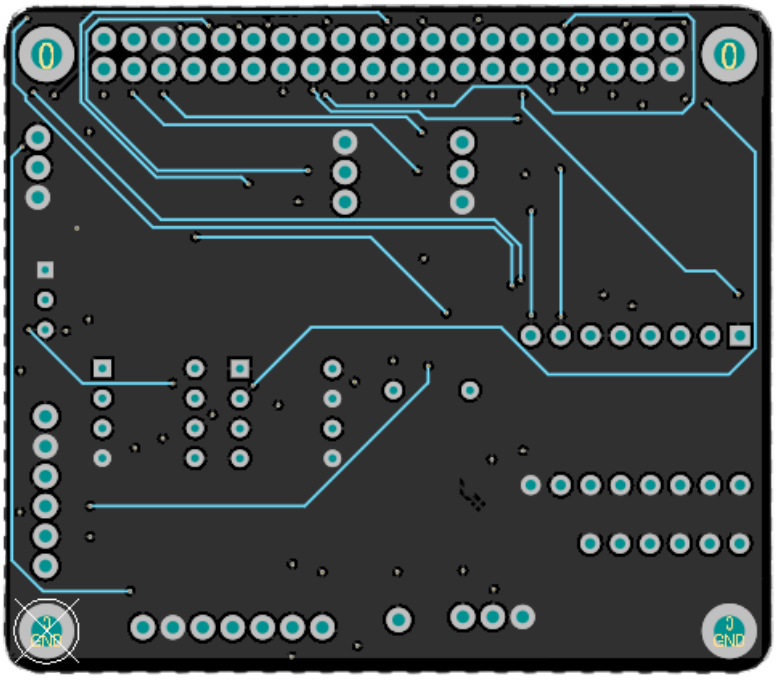

Figura 44: *Layout* de la *Layer* interna 2 de la plataforma

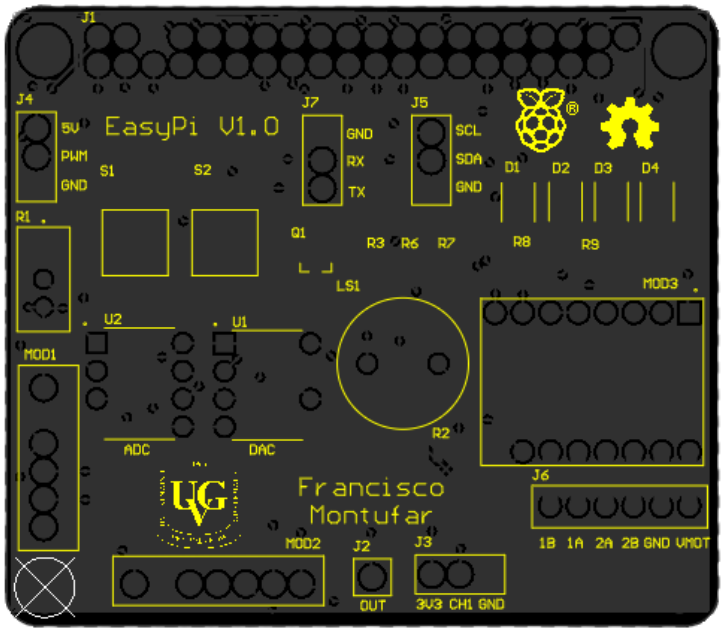

Figura 45: *Layout* del *Top Overlay* de la plataforma

7.3. Modelo 3D

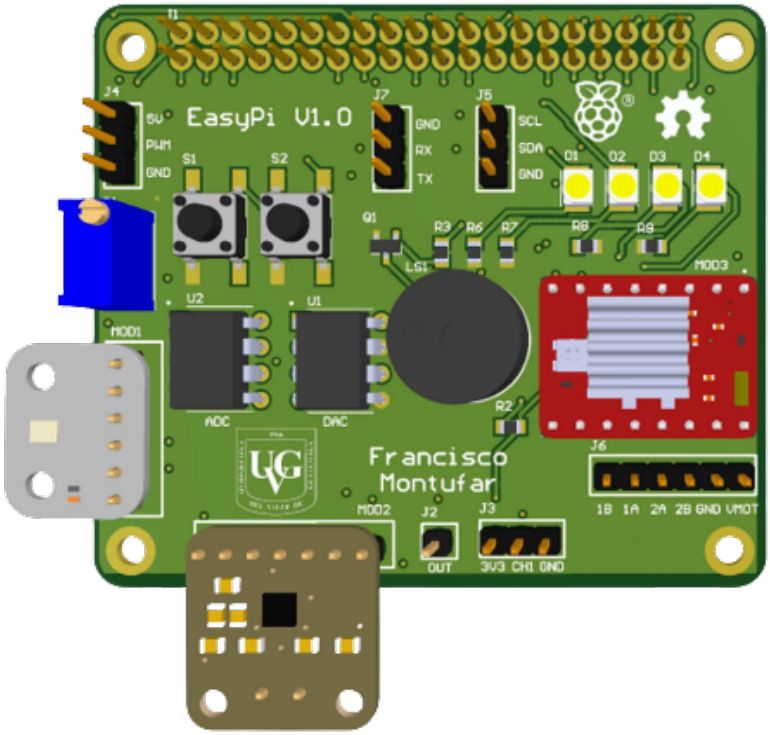

Figura 46: Modelo 3D de la plataforma

Para el ancho de pista de la plataforma, se realizaron los respectivos cálculos basándose en el estándar IPC-2221 con ayuda de una herramienta para realizar este tipo de cálculos.

Debido a que la *Raspberry Pi* entrega un máximo de 16mA por pin, se utilizaron los siguientes parámetros para el cálculo de ancho de pista de las capas externas:

- Corriente máxima: 16mA
- *Temperature Rise*: 10°
- Grosor: 1 oz/ft^2
- Temperatura ambiente: 25°

Con esos valores, se obtiene que el ancho de pista es de 0.04mils, lo cual es muy delgado, por lo que es posible usar mas anchos, en este caso, se utilizó un de 15 mils/0.381mm lo cual es un poco más ancho que lo recomendado para fabricación. Ese ancho se utilizó para todos las interconexiones, excepto la salida del driver A4988, ya que ese componente entrega hasta 2A de corriente, por lo que en esas interconexiones se calculó para un ancho de 31.55 mils o 0.8mm, pero por mejorar la la disipación de calor, se utilizó el doble del ancho (1.6mm).

Para el ancho de pista de las capas internas se utilizaron los siguientes parámetros:

- Corriente máxima: 16mA
- *Temperature Rise*: 10°
- Grosor: 0.5 oz/ft^2
- Temperatura ambiente: 25°

Con esos valores, se obtiene que el ancho de pista es de 0.21mils/0.01mm, lo cual es muy delgado, por lo que es posible usar mas anchos, en este caso, se utilizó un de 15 mils/0.381mm lo cual es un poco más ancho que lo recomendado para fabricación.

Para los otros parámetros de diseño, se tomaron en cuenta las especificaciones de uso del fabricante (JLCPCB) y se editaron en las reglas de diseño. Se editaron parámetros como *clearance*, tamaño de vía, tamaño de agujero, etc.

CAPÍTULO 8

Fabricación de la plataforma

8.1. Archivos generados

Para poder fabricar la plataforma, es necesario generar ciertos archivos para que un equipo especial pueda interpretar esos documentos y poder materializar lo ingresado. Entre estos archivos, tenemos los archivos Gerber, los cuales son un archivo que contiene parámetros de configuración, definiciones de apertura, ubicaciones de las coordenadas XY y códigos de comando para dibujo.[27]

Luego tenemos el NC Drill o *Numeric control drill file* que regula toda la información respecto a la perforación de vías y agujeros. Este documento incluye la ubicación de los agujeros, el tamaño de estos y el tamaño de la herramienta. [28]

Por último tenemos los PCB prints, los cuales fueron generados con el fin de imprimirlos en papel acetato y cubrir la placa para aplicar luz ultravioleta a la máscara de soldadura aplicada.

8.1.1. *Gerbers*

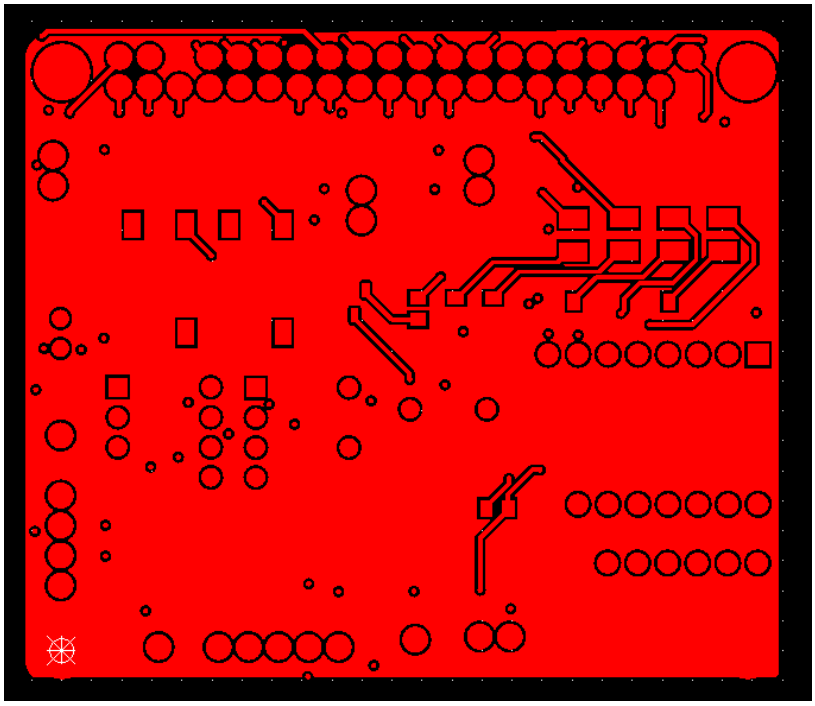

Figura 47: *Gerber* de la *Top Layer*

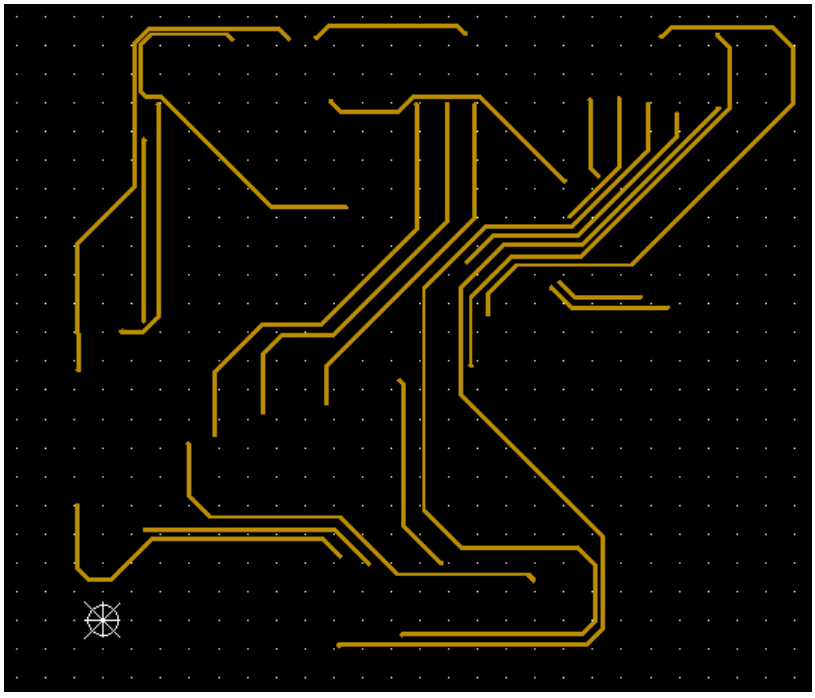

Figura 48: *Gerber* de la *Layer* interna 1

Figura 49: *Gerber* de la *Layer* interna 2

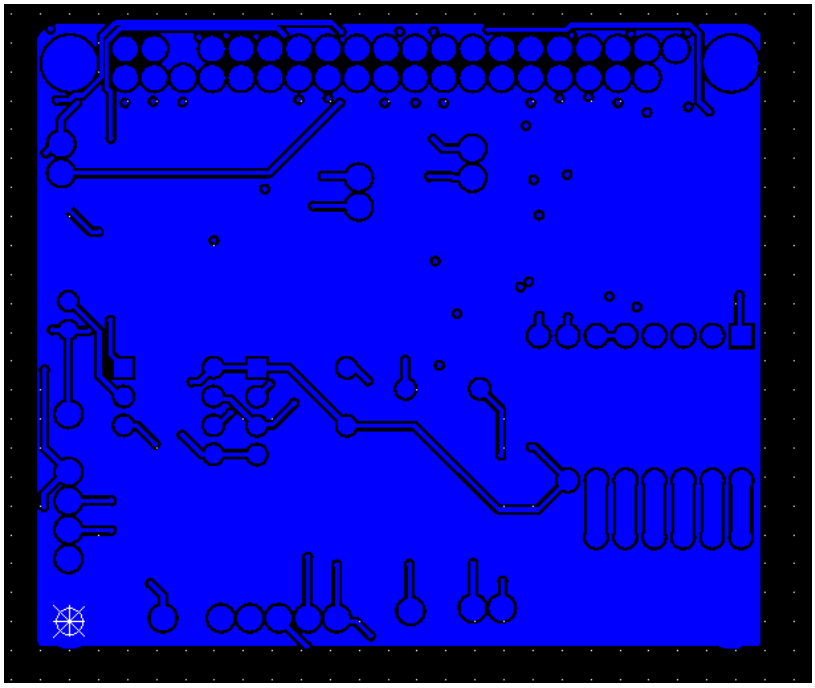

Figura 50: *Gerber* de la *Bottom Layer*

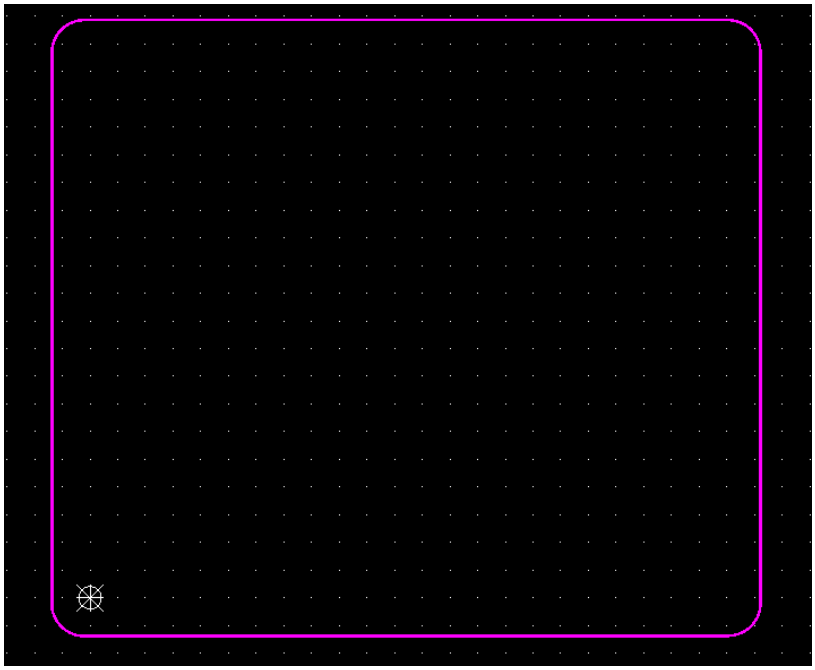

Figura 51: *Gerber* de la *Keepout Layer*

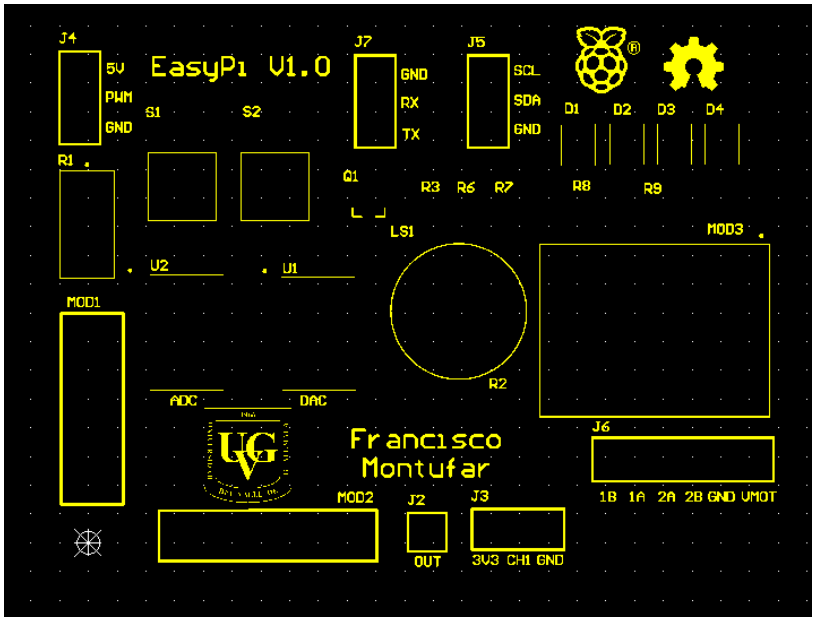

Figura 52: *Gerber* del *Top Overlay*

8.1.2. *NC Drill*

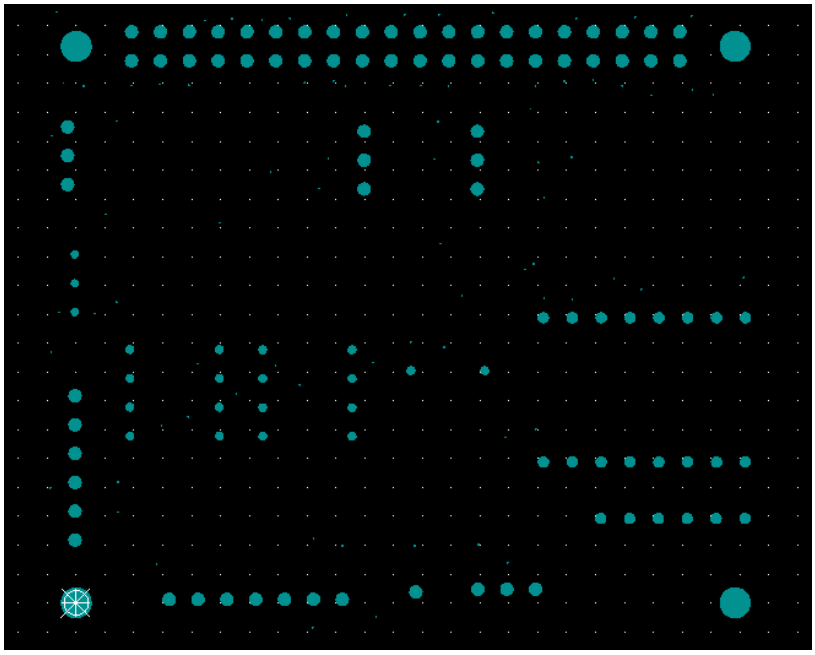

Figura 53: Archivo generado NC Drill

8.2. Placa física

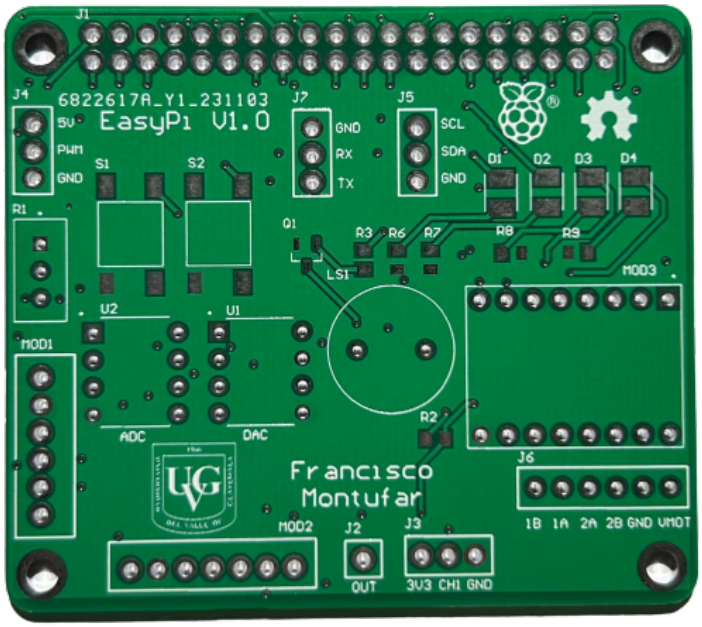

Figura 54: *Top Layer* placa física

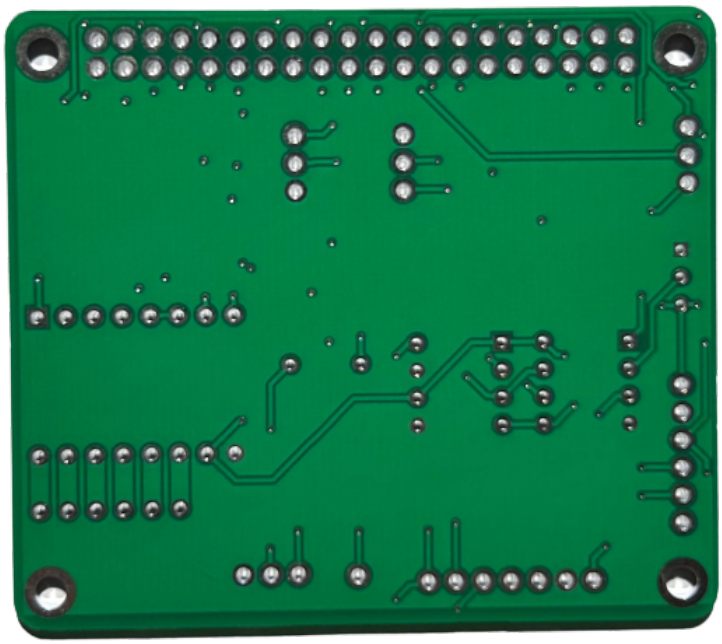

Figura 55: *Bottom Layer* placa física

8.3. Ensamble de la placa

Una vez terminada la soldadura de la plataforma, se debe de ensamblar, ya que al ser modular, la mayoría de componentes son removibles. Es importante señalar que la plataforma cuenta con pines sin algún módulo conectado. Esto es debido a que el diseño fue hecho pensando en la versatilidad de la plataforma, ya que en ciertos módulos o protocolos de comunicación se utiliza una gran variedad de dispositivos, por lo que no se quería limitar a un único artefacto y se dejaron únicamente los pines respectivos para realizar las conexiones acorde al protocolo o módulo utilizado.

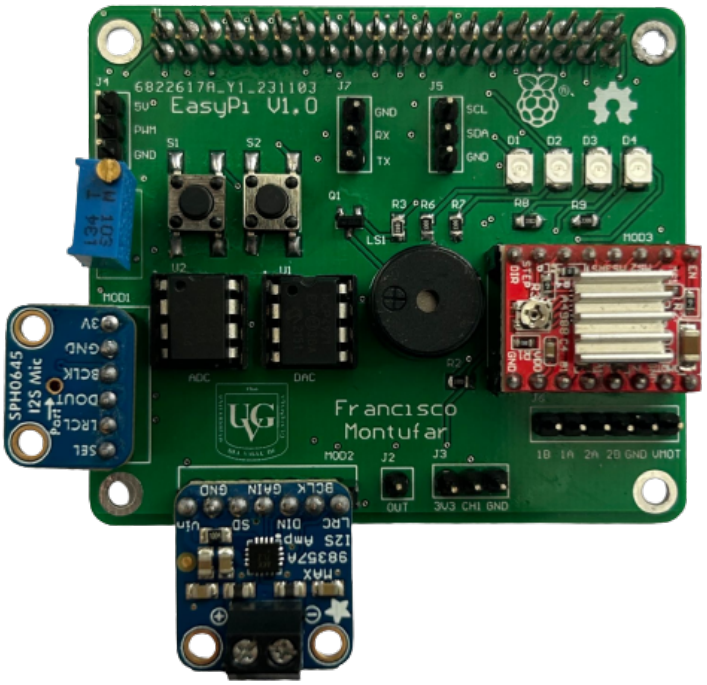

Figura 56: Vista superior de la placa física

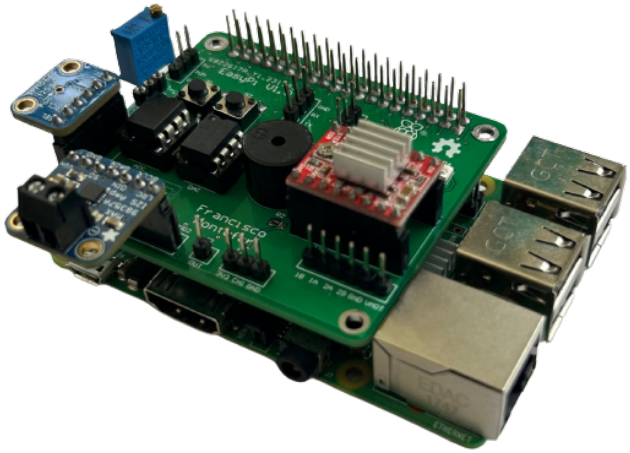

Figura 57: Vista ortogonal de placa física conectada a la *Raspberry Pi 3B+*

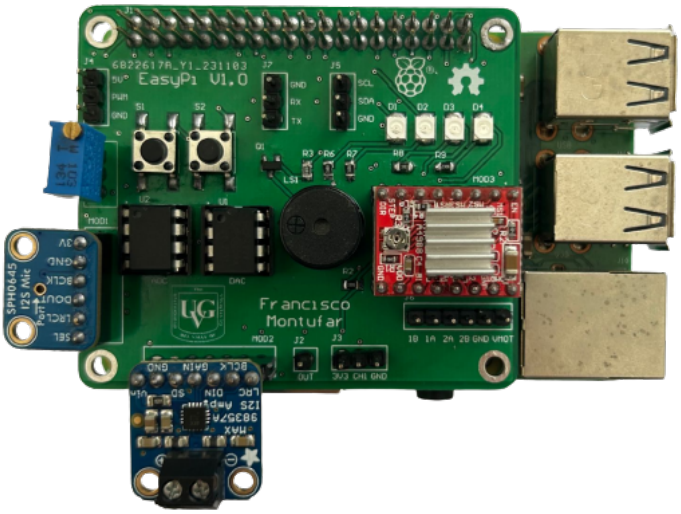

Figura 58: Vista superior de placa física conectada a la *Raspberry Pi 3B+*

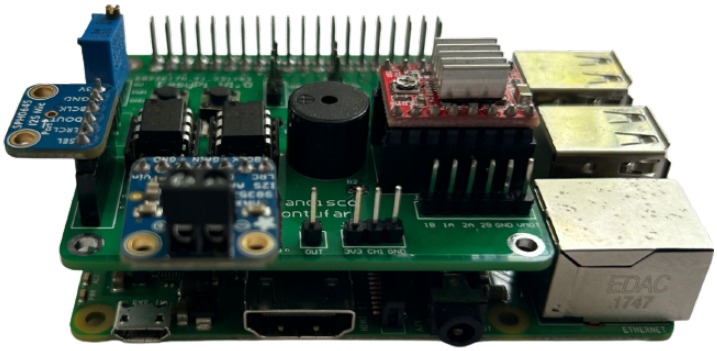

Figura 59: Vista frontal de placa física conectada a la *Raspberry Pi 3B+*

CAPÍTULO 9

Experimentos con la plataforma

Con el ensamblaje completo y una conexión exitosa a la *Raspberry Pi 3B+*, se llevaron a cabo diversos experimentos para validar el funcionamiento de la plataforma y sus componentes mediante el desarrollo de algoritmos creados en C y utilizando la librería WiringPi. Es importante mencionar que dos algoritmos son desarrollados en *Python*, ya que la documentación de los módulos I^2S incluía todas las configuraciones y comandos para utilizar efectiva los módulos.

El primer experimento consistía en validar los componentes más básicos de la plataforma: los LEDs y los pulsadores. Para verificar el correcto funcionamiento de estos, se realizó un algoritmo en donde se inicializa un contador y según el pulsador que se presione, el contador incrementa o decrementa su valor. Según el valor que tenga el contador, se accionan las LEDs, mostrando el valor en binario (1 siendo la LED encendida y 0 siendo la LED apagada). En las figuras 60 y 61 se puede observar el comportamiento de las LEDs según el contador.

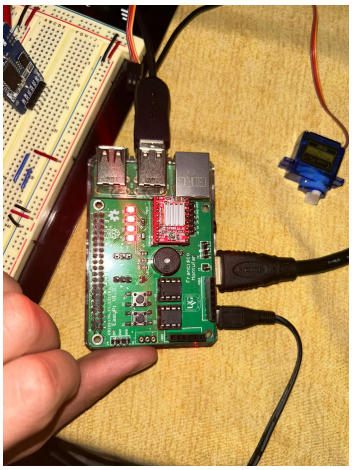

Figura 60: Contador de 4 bits utilizando LEDs y pulsadores

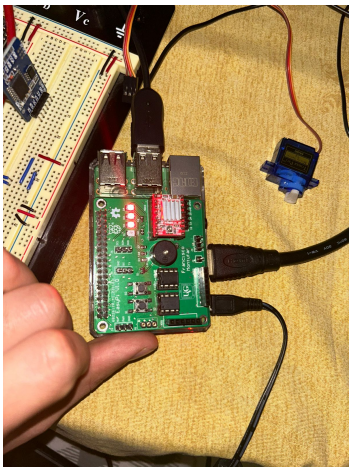

Figura 61: Contador de 4 bits utilizando LEDs y pulsadores

El segundo experimento consistía en utilizar el protocolo SPI para configurar un convertidor analógico a digital (ADC) MCP 3002 para que lea valores de un potenciómetro y que en la consola se despliegue la conversión del valor analógico. Para tener algo más visual en la placa, se realizó un escalamiento de los valores para tener un rango menor, pero que sea equivalente al rango original que es de 0 a 1023, ya que el ADC es de 10 bits. Con ese nuevo rango se agregó la instrucción de que conforme fuera incrementando o decrementando el valor, se fueran encendiendo o apagando las LEDs de forma secuencial. También se generó un archivo en el formato .csv, en donde se escriben todos los valores obtenidos de la conversión. Los valores obtenidos se grafican respecto al tiempo para verificar que la conversión haya sido exitosa (Figuras 62 y 63).

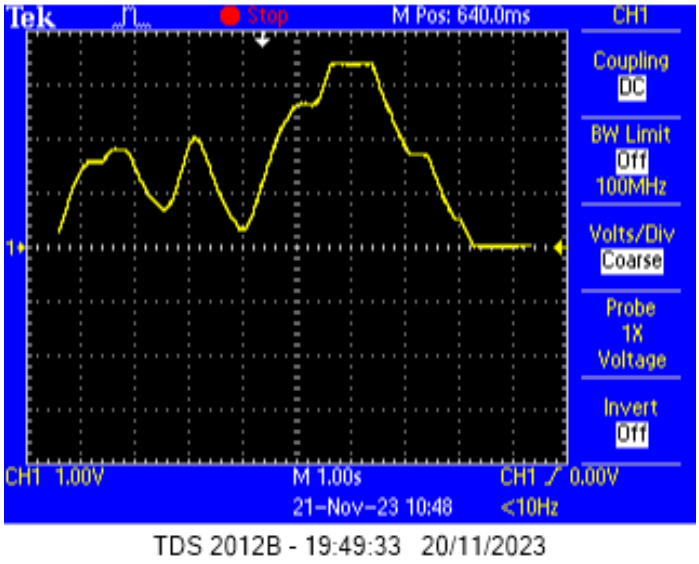

Figura 62: Vista de la señal analógica en el osciloscopio

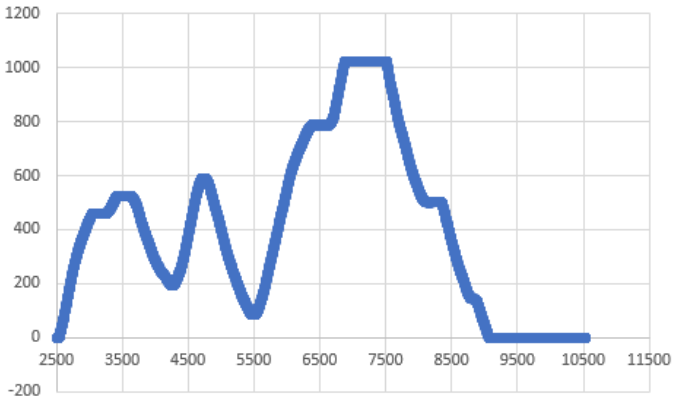

Figura 63: Gráfica de los valores de la conversión

El tercer experimento consiste en validar el funcionamiento de los módulos de I^2S, los cuales consistían en un módulo de micrófono y otro de amplificador. Para accionar las funciones del módulo, fue necesario utilizar la documentación del fabricante, la cual incluía comandos para utilizar dichos módulos. Primero se utilizó el comando *arecord -l* el cual despliega todos los dispositivos de entrada de audio conectados a la *Raspberry Pi 3B+*(en este caso, el micrófono I^2S).

```
pi@raspberrypi:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: sndrpii2scard [snd_rpi_i2s_card], device 0: simple-card_codec_link snd-s
oc-dummy-dai-0 [simple-card_codec_link snd-soc-dummy-dai-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

Figura 64: Resultado del comando arecord

Para empezar el proceso de grabado con el micrófono se debe de utilizar el siguiente comando:

```
arecord -D plughw:0 -c1 -r 48000 -f S32_LE -t wav -V mono -v file.wav
```

El cual genera un archivo .WAV con el resultado una vez se hayan generado sonidos.

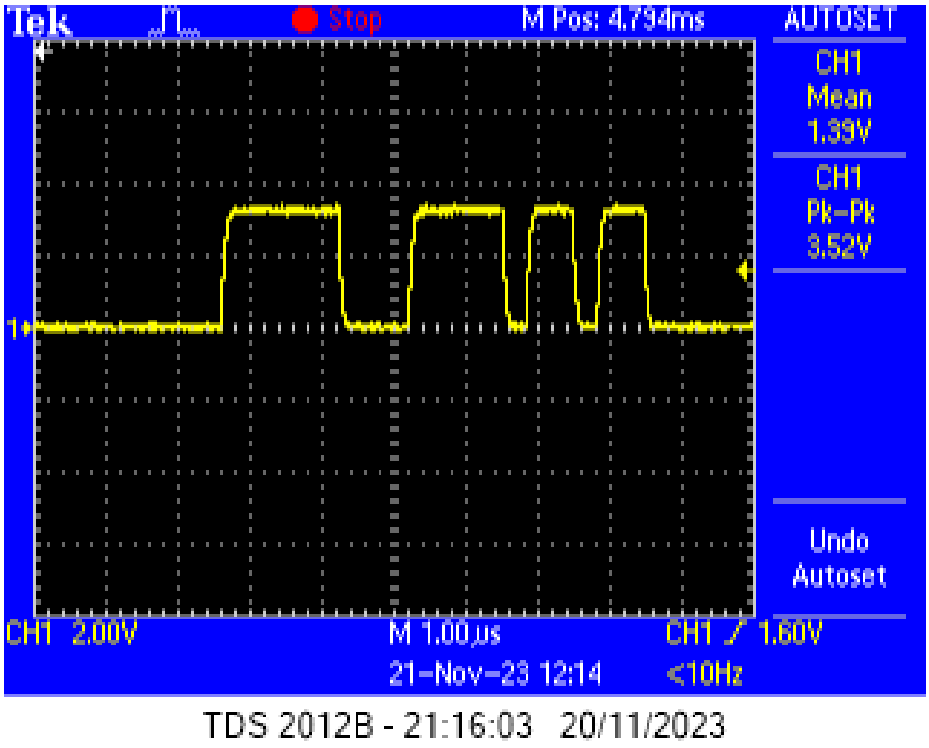

Figura 65: Salida generada por micrófono I^2S

En la Figura 65 se puede observar la señal generada por el micrófono. La señal es transmitida en formato PCM, por eso es que tiene forma de señal cuadrada, ya que son los datos analógicos modulados para poder ser transmitidos sin riesgo de pérdidas.

Después, con el comando *aplay -l* se verifican los dispositivos de salida de audio conectados a la *Raspberry Pi 3B+* (en este caso el módulo de amplificador).

```
pi@raspberrypi:~ $ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: sndrpii2scard [snd_rpi_i2s_card], device 0: simple-card_codec_link snd-s
oc-dummy-dai-0 [simple-card_codec_link snd-soc-dummy-dai-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

Figura 66: Resultado de comando aplay

Una vez se confirmó que el módulo fue reconocido, se procedió a ejecutar el archivo .WAV generado anteriormente y según se esperaba, la bocina conectada al módulo estaba reproduciendo el sonido del archivo.

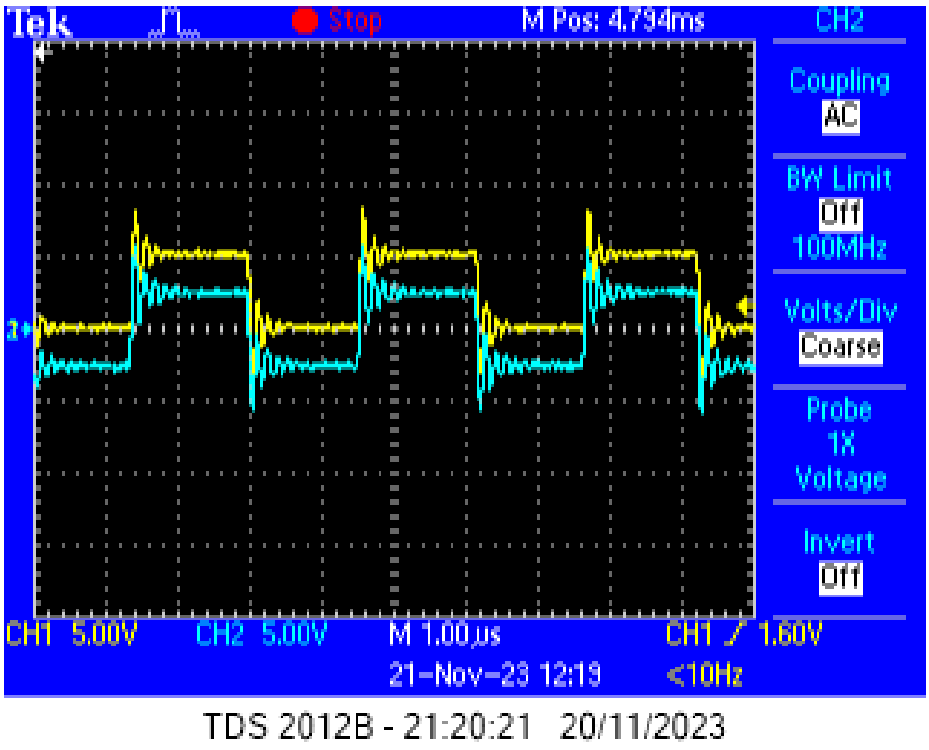

Figura 67: Señal de salida del módulo MAX98357

Así como el micrófono genera una señal en formato PCM, el módulo MAX98357 recibe una señal en el mismo formato. La señal generada por el módulo que va hacia una bocina se puede observar en la Figura 67 y se puede apreciar que la señal fue demodulada y procesada.

El cuarto experimento tenía como objetivo utilizar accionar un motor *Stepper* con el módulo A4988. Para lograrlo, se desarrolló un algoritmo en C en donde por medio de los pines de *STEP* y DIR, en los cuales se controla el movimiento y la dirección del motor. El resultado fue el esperado ya que el motor giraba en el sentido indicado por el pin DIR y debido a la resolución de los micropasos, se observa que el motor se mueve con un paso completo o *Full Step*.

En las figuras 68 y 69 se observa el motor *Stepper* en diferentes posiciones luego de haberlo accionado.

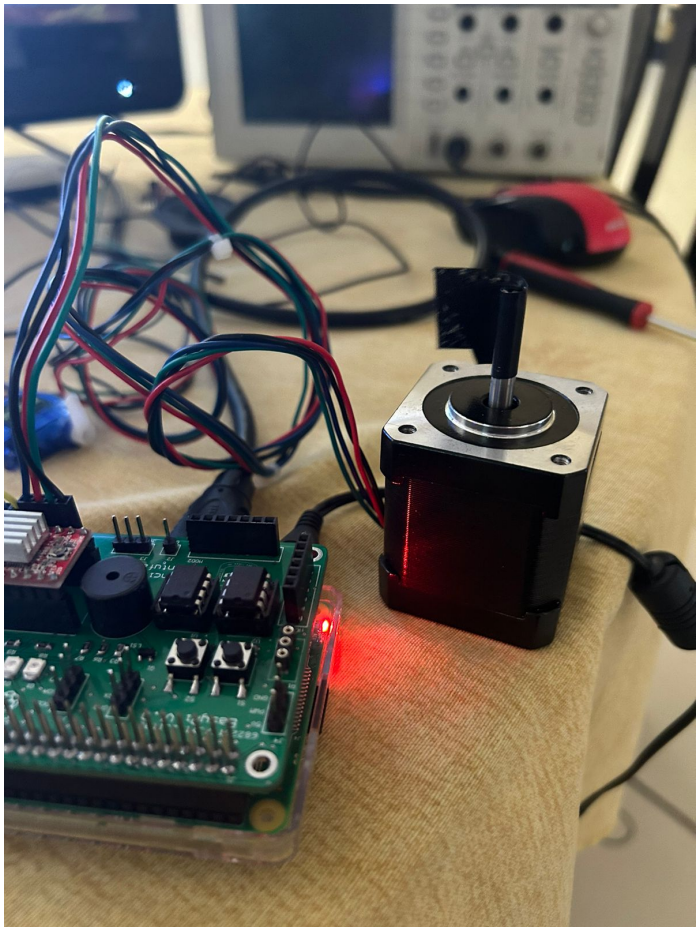

Figura 68: Motor *Stepper*

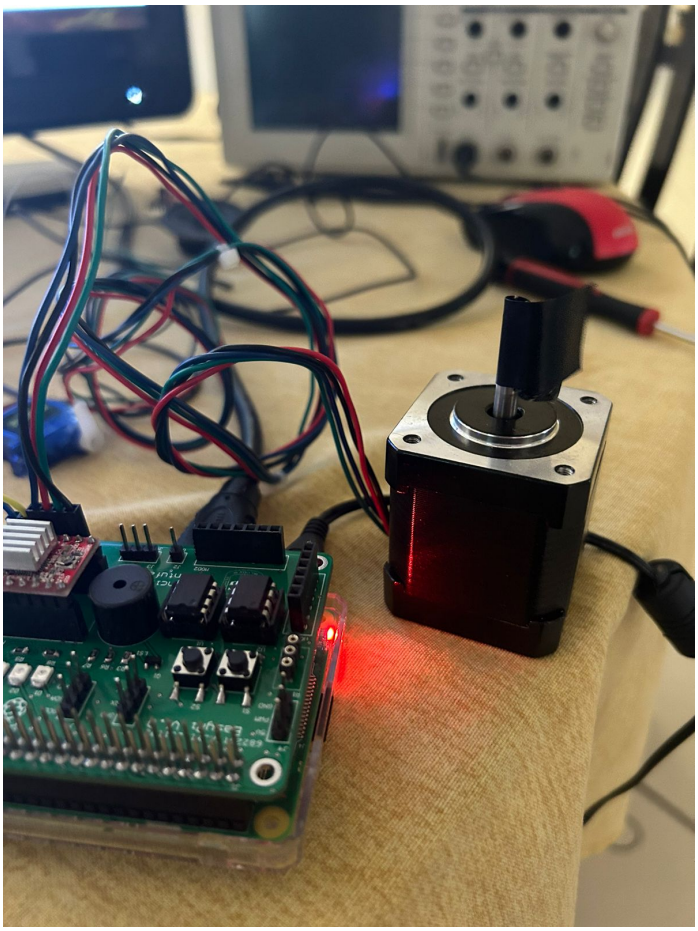

Figura 69: Motor *Stepper*

El quinto experimento buscaba la comunicación entre dispositivos, ya sean microcontroladores u otros artefactos utilizando el protocolo UART. Se desarrolló un algoritmo en donde la *Raspberry Pi 3B+* enviaba el caracter 'h' a un Arduino Nano. Por medio de un osciloscopio, se pudo observar la secuencia de bits enviados por UART. En esa señal se aprecia el byte completo y la representación binaria del caracter, en este caso se aprecia un byte con el valor de 01101000, pero empezando por el bit menos significativo, ya que UART envía los bits desde el menos significativo hasta el más significativo.

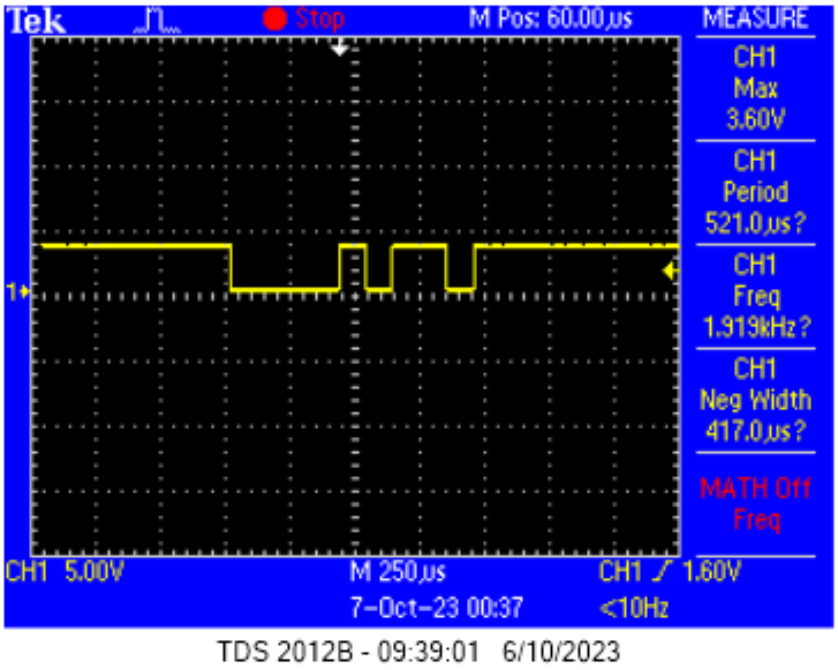

Figura 70: Dato enviado por UART

Luego, se realizó otra prueba en donde la *Raspberry Pi* 3B+ se comunicaba con un Arduino Nano por medio de UART. La *Raspberry Pi* 3B+ envía una cadena de datos y cuando el Arduino recibe los datos, le envía una cadena de regreso, comprobando que la comunicación fue exitosa. Esto se observa en la figura .

```
pi@raspberrypi:~/TES/example_codes_EzPI $ ./uart
Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!

Esto envió el arduino: Arduino responde: Hola desde Arduino!
```

Figura 71: Cadena de Arduino enviada por UART y recibida por la *Raspberry Pi* 3B+

El sexto experimento consistía en utilizar el protocolo SPI para configurar un convertidor digital a analógico (DAC) MCP 4921 para que lea valores digitales generados por el ordenador y que se genere una señal analógica equivalente. Para poder llevar esto a cabo, se desarrolló un código e C donde se configura el DAC y se envía un contador, el cual luego de

llegar a su valor máximo, se reinicia. Esto forma una señal diente de sierra, la cual se puede observar en la Figura 72.

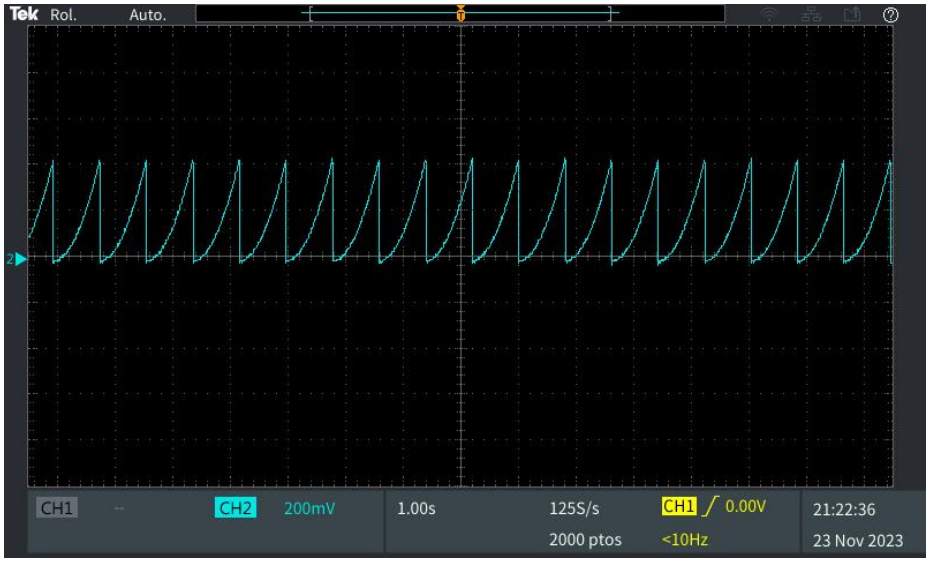

Figura 72: Salida del DAC

El séptimo experimento consistía en generar una señal PWM utilizando el módulo de la *Raspberry Pi* 3B+. Para generar la PWM, se tuvo que habilitar el módulo y modificar parámetros como el reloj, el modo de operación y el ciclo de trabajo. En la Figura 73 se puede observar una PWM generada con un ciclo de trabajo del 50 %, lo que implica que el tiempo de encendido y de apagado es el mismo.

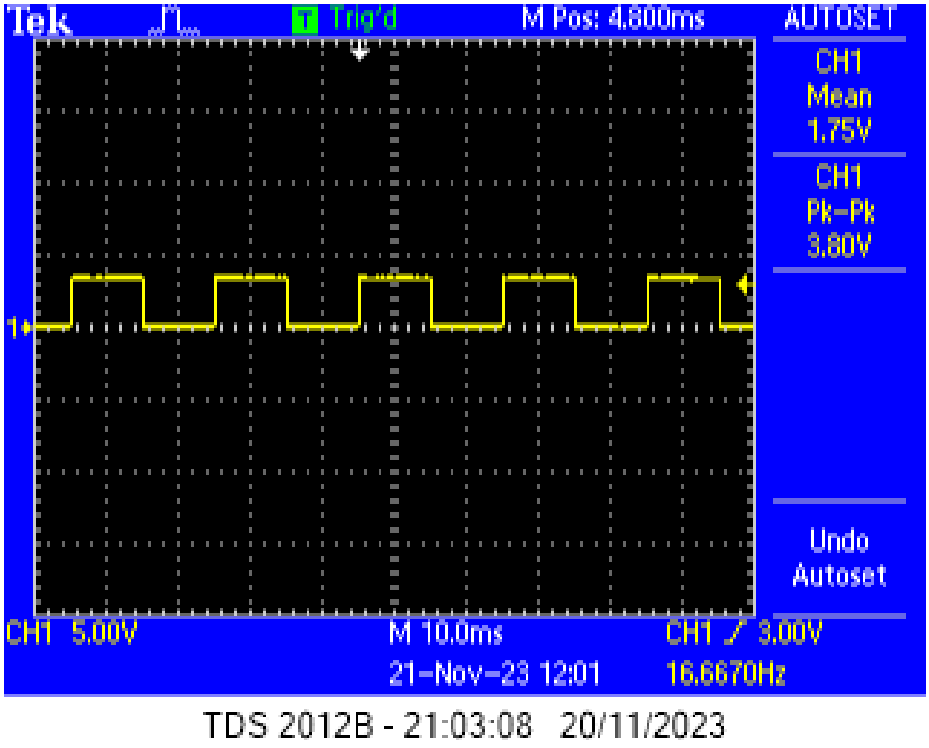

Figura 73: PWM generada

Una vez generada la PWM, se modificó para que fuera capaz de controlar un servo motor, por lo que se le cofiguró un período de 20ms (ecuación 2) y se fue modificando el ciclo de trabajo para que el motor se accionara. En la Figura 74 se observa la señal resultante con el ciclo de trabajo modificado.

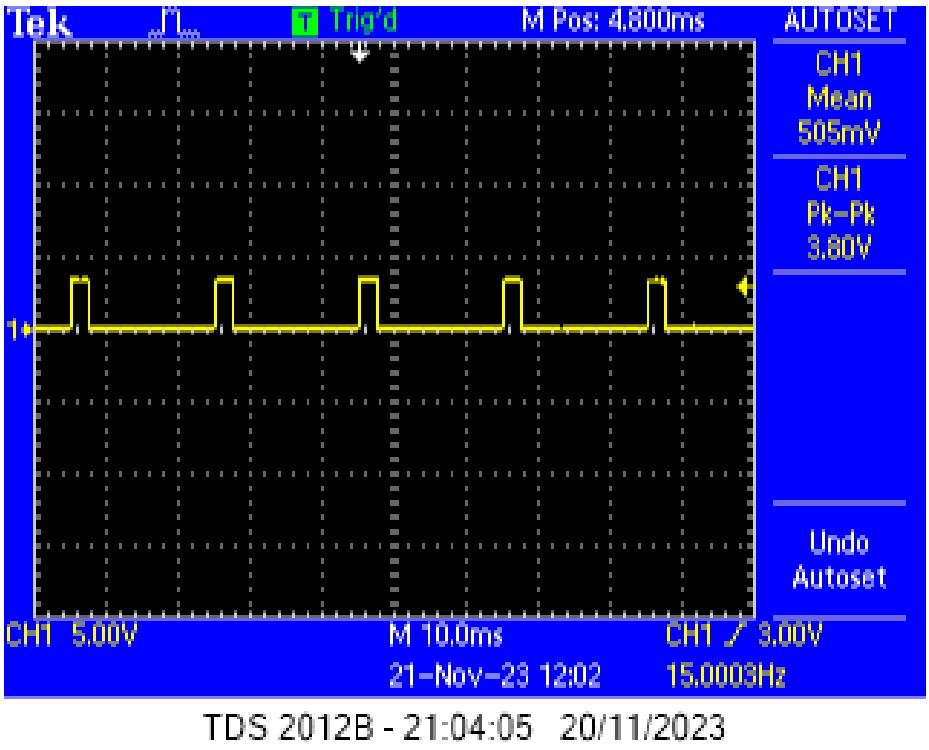

Figura 74: PWM para servomotor

El octavo experimento consistía en utilizar I^2C para configurar un RTC DS3231, el cual es un reloj en tiempo real. Para lograrlo, se desarrolló un código en C donde por medio de I^2C se configuran los segundos, minutos, horas, día, mes y año, para que luego la *Raspberry Pi* lea los datos y los despliegue. Los resultados se pueden observar en la Figura 75.

```
pi@raspberrypi:~/TES/example_codes_EzPI $ ./i2c
Hora: 13:19:00
Fecha: 26/11/23
Hora: 13:19:01
Fecha: 26/11/23
Hora: 13:19:02
Fecha: 26/11/23
Hora: 13:19:03
Fecha: 26/11/23
Hora: 13:19:04
Fecha: 26/11/23
Hora: 13:19:05
Fecha: 26/11/23
Hora: 13:19:06
Fecha: 26/11/23
```

Figura 75: Datos recibidos del RTC usando I^2C

Por último se desarrolló un código en C donde se genera una señal cuadrada por medio

de el encendido y apagado de un pin en un tiempo establecido, es decir, se generó una señal cuadrada a una frecuencia específica. Esto fue con el fin de emitir un sonido con el buzzer pasivo, ya que este al no contar con componentes electrónicos internos, es necesario utilizar una señal cuadrada para manipularlo. El experimento funcionó acorde a lo esperado ya que entre más alta la frecuencia, más agudo era el sonido y mientras más baja la frecuencia, más grave era el sonido.

CAPÍTULO 10

Documentación

Una vez se finalizó con el ensamblaje, la experimentación y la validación, se procedió a desarrollar un folleto que incluye información general sobre como utilizar la plataforma, precauciones de uso, el *pinout*, descripciones generales sobre los componentes, descripción general de los módulos y protocolos, instalaciones requeridas y los códigos de ejemplo desarrollados para poder utilizar los módulos y protocolos mencionados.

Figura 76: Carátula del folleto

Para desarrollar el folleto, se utilizó una plataforma en línea llamada *Canva*, en donde se utilizaron diversas herramientas de diseño grafico para darle estructura y un diseño estético al folleto. El folleto junto con los códigos de ejemplo se pueden encontrar en el siguiente enlace: `https://github.com/mon19379/example_codes_EzPI`.

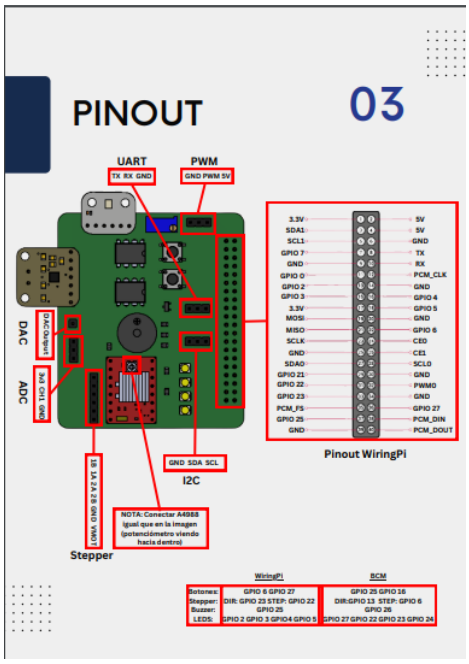

Figura 77: Pinout de la plataforma mostrado en el folleto

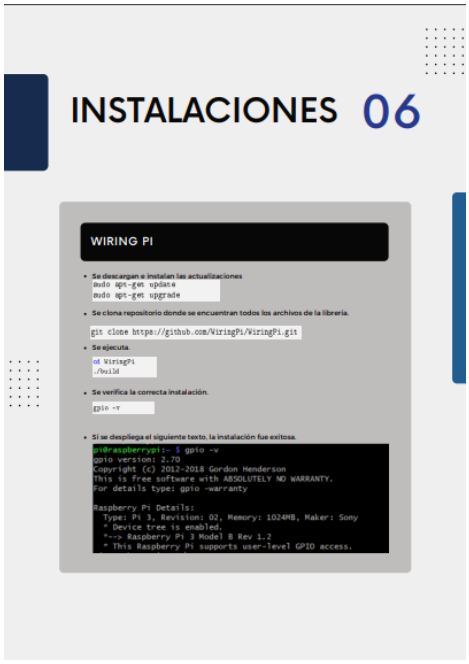

Figura 78: Parte de las instalaciones requeridas mostradas en el folleto

CAPÍTULO 11

Conclusiones

- Se validó el funcionamiento de los diferentes módulos de un *Raspberry Pi 3B+* con el uso de los pines de propósito general y la plataforma electrónica fabricada.
- Se fabricó de manera exitosa una plataforma electrónica funcional y de forma modular para poder intercambiar componentes con mayor facilidad.
- Se diseñó una placa que facilite la interconexión con la *Raspberry Pi 3B+* y que permite utilizar de forma correcta los diferentes módulos y protocolos.
- Se utilizó el módulo I^2S MAX 98357 con la versión 5.10.17-v7+ del sistema operativo *Raspberry Pi OS* o versiones anteriores.
- Se desarrolló una documentación adecuada que facilita el entendimiento de la utilización de la plataforma electrónica.

CAPÍTULO 12

Recomendaciones

- Seleccionar componentes que tengan una amplia documentación para facilitar su implementación y procurar que sean de fácil acceso en la región, ya que se dificulta el traslado y podrían aumentar los costos a la hora de adquirir estos dispositivos.
- Antes de soldar componentes en la placa aplicar una *Solder mask* para facilitar el proceso y prevenir problemas de corto circuito a la hora de soldar. Se recomienda siempre tener los *PCB Prints* en papel acetato para evitar que la *Solder mask* cubra los pads de la placa.
- Utilizar *Python* en lugar de C para facilitar el desarrollo de algoritmos ya que este cuenta con más documentación, ejemplos y librerías.
- Desarrollar una interfaz gráfica que facilite la interacción con los componentes de la plataforma y proporcione una experiencia más amigable sin necesidad de tener que manipular directamente la plataforma.
- Utilizar la versión Linux version 5.10.17-v7+ de *Raspberry Pi OS* para utilizar ambos módulos I^2S sin problemas, ya que al utilizar versiones más recientes, la configuración cambia y puede llegar a dar problemas de funcionamiento.
- Integrar el *overlay* 'i2s-mmap' en versiones posteriores a Linux version 5.10.17-v7+ de *Raspberry Pi OS* para evitar problemas con I^2S.
- Realizar conexiones a los pines del driver A4988 (MS1, MS2 y MS3) en donde se pueda modificar el *microstep* a utilizar. Se recomienda hacer esa modificación directamente en el diseño del PCB. Otra alternativa podría ser la adición de *jumpers* o alguna conexión fisica a la plataforma ya fabricada.

CAPÍTULO 13

Bibliografía

[1] A. Limited, *Altium Designer*, https://www.altium.com/es/altium-designer/, Accessed : 22-04-2023, 2023.

[2] R. P. Foundation, *About Us*, https://www.raspberrypi.com/about/, Accessed : 22-04-2023, 2023.

[3] A. D. M. Esquivel, *Implementación y validación del Algoritmo de Robótica de Enjambre Particle Swam optimization*, Guatemala,GT, ene. de 2022.

[4] L. J. Nij, *Evaulación y validación de plataformas móviles para aplicaciones prácticas de robótica*, Guatemala,GT, ene. de 2022.

[5] R. P. Foundation, *Raspberry Pi Documentation, Procesors*, https://www.raspberrypi.com/documentation/computers/processors.html, Accessed : 21-05-2023, 2023.

[6] R. Hat, *What is an ARM processor?* https://www.redhat.com/en/topics/linux/what-is-arm-processor, Accessed : 21-05-2023, 2022.

[7] R. P. Foundation, *Raspberry Pi OS*, https://www.raspberrypi.com/documentation/computers/os.html, Accessed : 21-05-2023, 2023.

[8] I. Software in the Public Interest, *¿Qué es Debian GNU/Linux?* https://www.debian.org/releases/jessie/armel/ch01s02.html.es, Accessed : 21-05-2023, 2023.

[9] E. Peña y M. Grace Legaspi, "UART: A Hardware Communication protocol, understanding Universal Asynchronous Receiver/Transmitter," Analog Devices, Wilmington, MA, inf. téc. Vol 53, dic. de 2020.

[10] M. Grusin, *Serial Peripheral Interface (SPI)*, https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all, Accessed : 21-05-2023, 2023.

[11] A. Pini, *Por qué y cómo usar la interfaz periférica serial para simplificar las conexiones entre distintos dispositivos*, https://www.digikey.com/es/articles/why-how-to-use-serial-peripheral-interface-simplify-connections-between-multiple-devices, Accessed : 21-05-2023, 2019.

[12] E. Peña y M. Grace Legaspi, "I^2C Communication Protocol: Understanding I^2C Primer, PMBus, and SMBus," Analog Devices, Wilmington, MA, inf. téc. Vol 55, nov. de 2021.

[13] Solectroshop, *¿Qué es PWM y cómo usarlo?* https://solectroshop.com/es/blog/que-es-pwm-y-como-usarlo--n38, Accessed : 21-05-2023, 2020.

[14] P. Semiconductors, *I^2S bus specification*, https://www.sparkfun.com/datasheets/BreakoutBoards/I2SBUS.pdf, Accessed : 29-06-2023, 1996.

[15] BYJU'S, *Pulse code modulation*, https://byjus.com/physics/pulse-code-modulation/, Accessed : 29-05-2023, 2023.

[16] Z. Peterson, *¿Qué es un PCB o Placa de Circuito Impreso?* https://resources.altium.com/es/p/what-is-a-pcb, Accessed : 21-05-2023, 2020.

[17] Sparkfun, *PCB Basics*, https://learn.sparkfun.com/tutorials/pcb-basics, Accessed : 21-05-2023, 2023.

[18] M. Wang, *Calculadora de trazas de PCB: todo lo que necesitas saber en 2022*, https://www.pcbmay.com/es/pcb-trace-calculator, Accessed : 29-05-2023, 2022.

[19] DigiKey, *PCB Trace Width Calculator*, https://www.digikey.com/en/resources/conversion-calculators/conversion-calculator-pcb-trace-width, Accessed : 10-09-2023, 2023.

[20] IPC, *Generic Standard on printed board design*, https://tinymicros.com/mediawiki/images/1/15/IPC-2221.pdf, Accessed : 10-09-2023, 1998.

[21] Drogon, *WiringPi*, http://wiringpi.com/contact/, Accessed : 29-05-2023, 2023.

[22] Allegro, *DMOS Microstepping Driver with Translator And Overcurrent Protection*, https://www.pololu.com/file/0J450/a4988_DMOS_microstepping_driver_with_translator.pdf, Accessed : 10-09-2023, 2014.

[23] Knowlses, *I2S Output Digital Microphone*, https://cdn-shop.adafruit.com/product-files/3421/i2S+Datasheet.PDF, Accessed : 10-09-2023, 2015.

[24] MaximIntegrated, *PCM Input Class D Audio Power Amplifiers*, https://cdn-shop.adafruit.com/product-files/3006/MAX98357A-MAX98357B.pdf, Accessed : 10-09-2023, 2016.

[25] Microchip, *2.7V Dual Channel 10-Bit A/D Converter with SPI Serial interface*, https://ww1.microchip.com/downloads/aemDocuments/documents/APID/ProductDocuments/DataSheets/21294E.pdf, Accessed : 10-09-2023, 2011.

[26] Microchip, *12-Bit DAC with SPI interface*, https://ww1.microchip.com/downloads/en/DeviceDoc/21897B.pdf, Accessed : 10-09-2023, 2007.

[27] Altium, *¿Qué son los archivos Gerber y cómo se utilizan en el proceso de fabricación de placas de circuito impreso?* https://resources.altium.com/es/p/what-gerber-file-pcb-fabrication-process, Accessed : 10-09-2023, 2017.

[28] RaymondPCB, *What is NC Drill File?* https://www.raypcb.com/nc-drill-file/, Accessed : 10-09-2023, 2023.

CAPÍTULO 14

Anexos

14.1. Instalaciónes requeridas

14.1.1. Instalación Altium Designer

1. Descargar instalador desde la página oficial: https://www.altium.com/es/altium-designer/
2. Ejecutar el instalador y presionar *NEXT*.

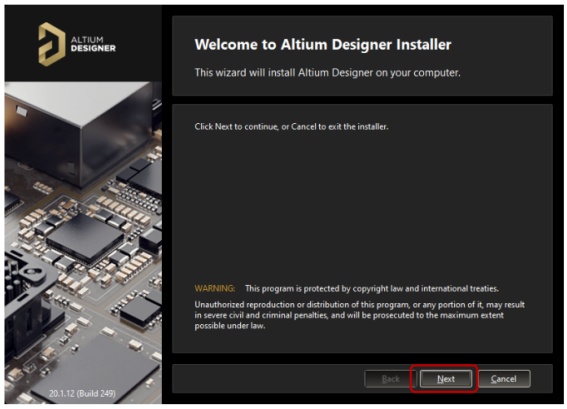

Figura 79: Instalador Altium Designer

3. Aceptar licencias y y presionar *NEXT*.

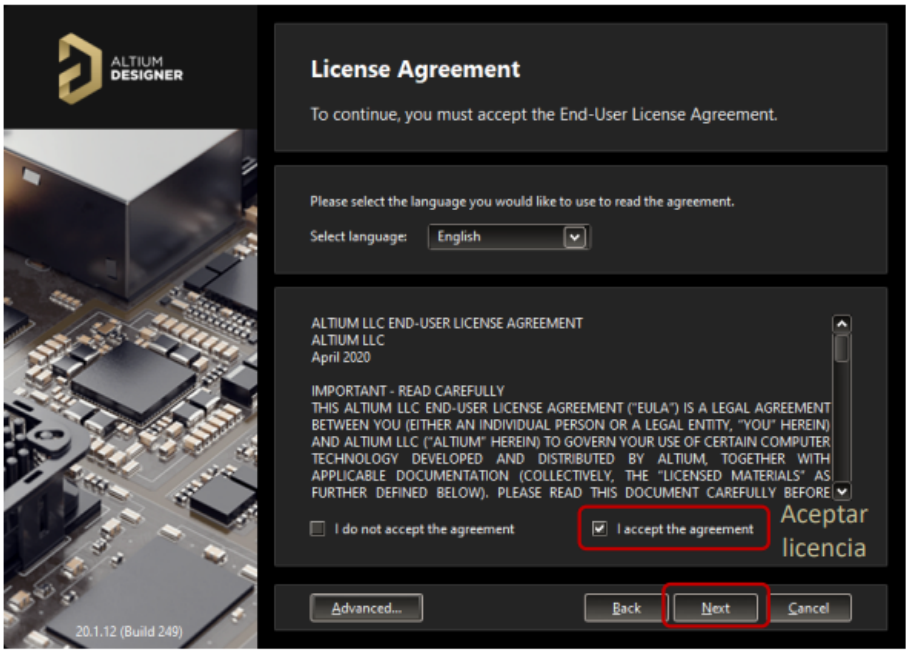

Figura 80: Instalador Altium Designer

4. Abrir *Platform extensions*.

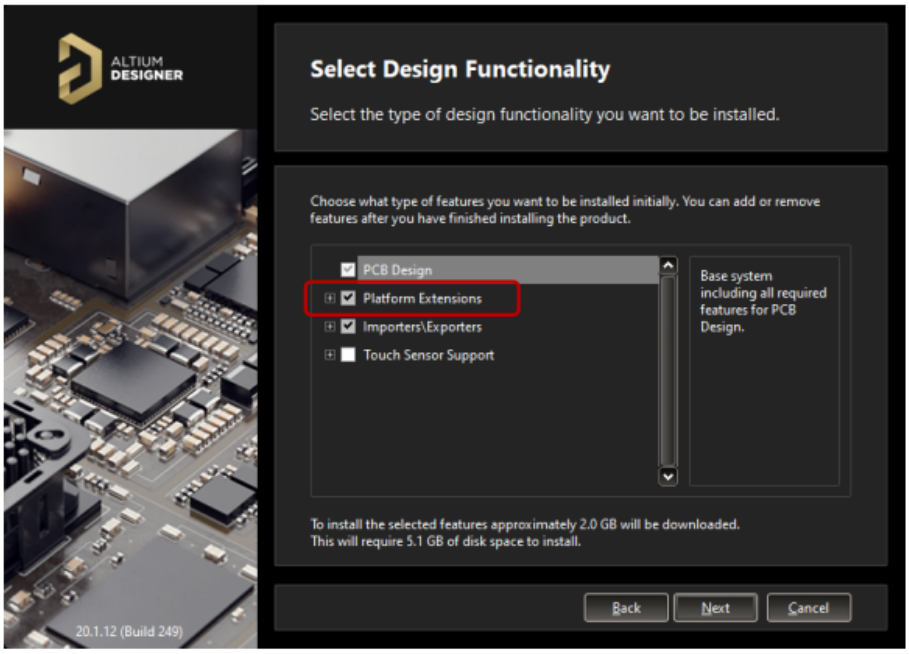

Figura 81: Instalador Altium Designer

5. Seleccionar *Mixed Simulation* y presionar *NEXT*.

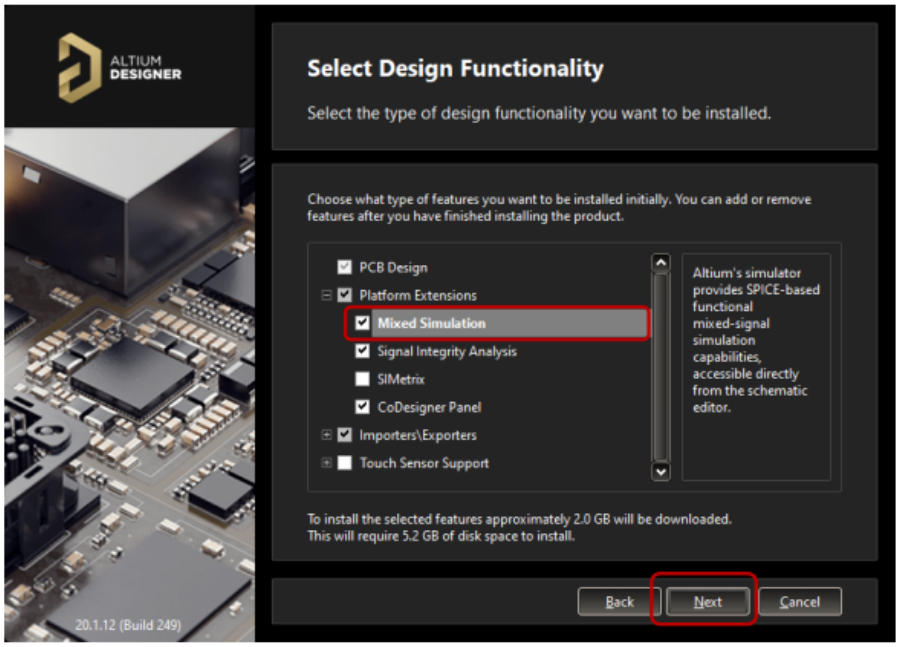

Figura 82: Instalador Altium Designer

6. Verificar directorios de instalación y presionar *NEXT*.

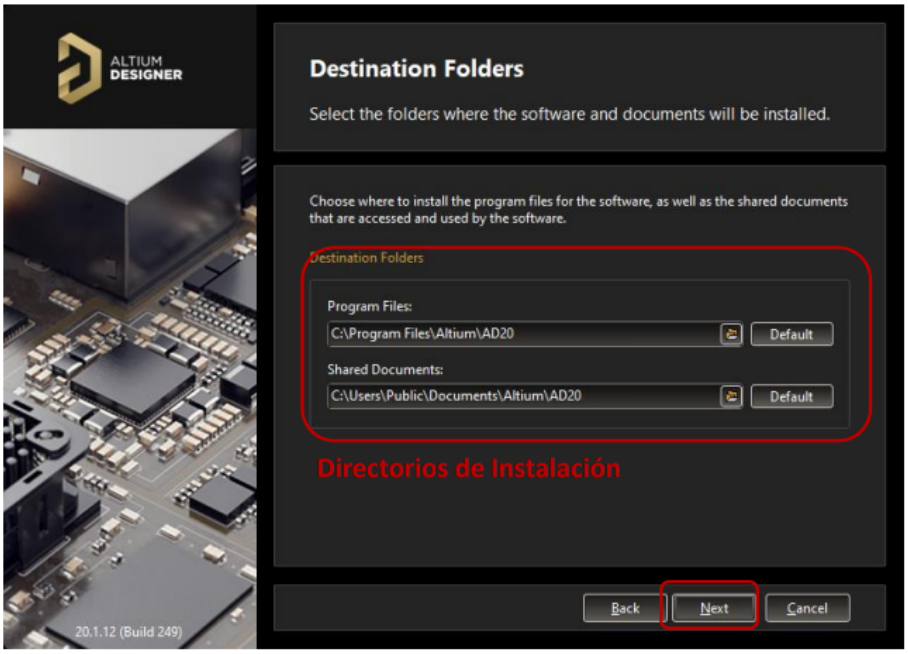

Figura 83: Instalador Altium Designer

7. Seleccionar cualquiera de los cuadros y presionar *NEXT*.

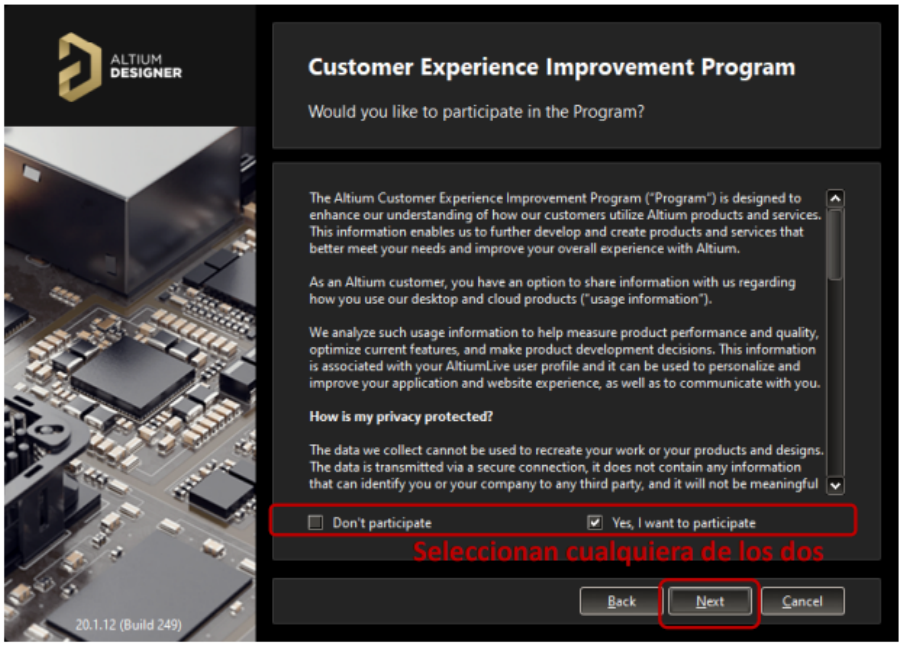

Figura 84: Instalador Altium Designer

8. Presionar *NEXT*.

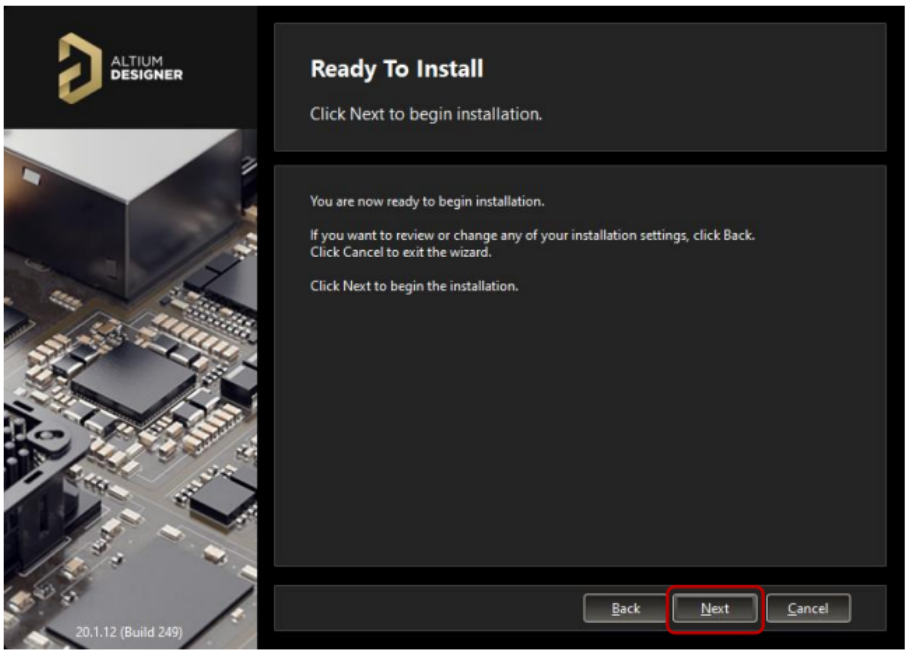

Figura 85: Instalador Altium Designer

9. Presionar *FINISH*.

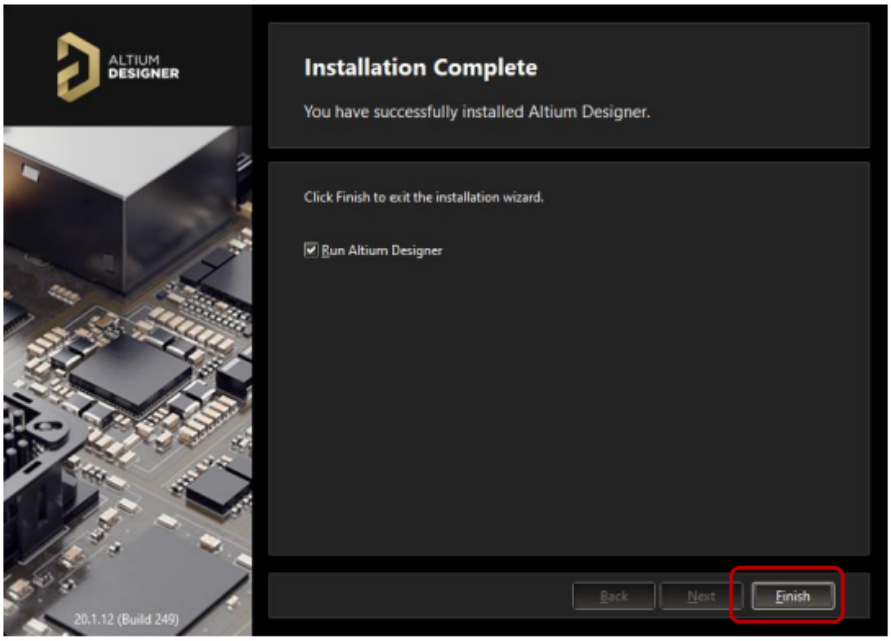

Figura 86: Instalador Altium Designer

10. Permitir acceso.

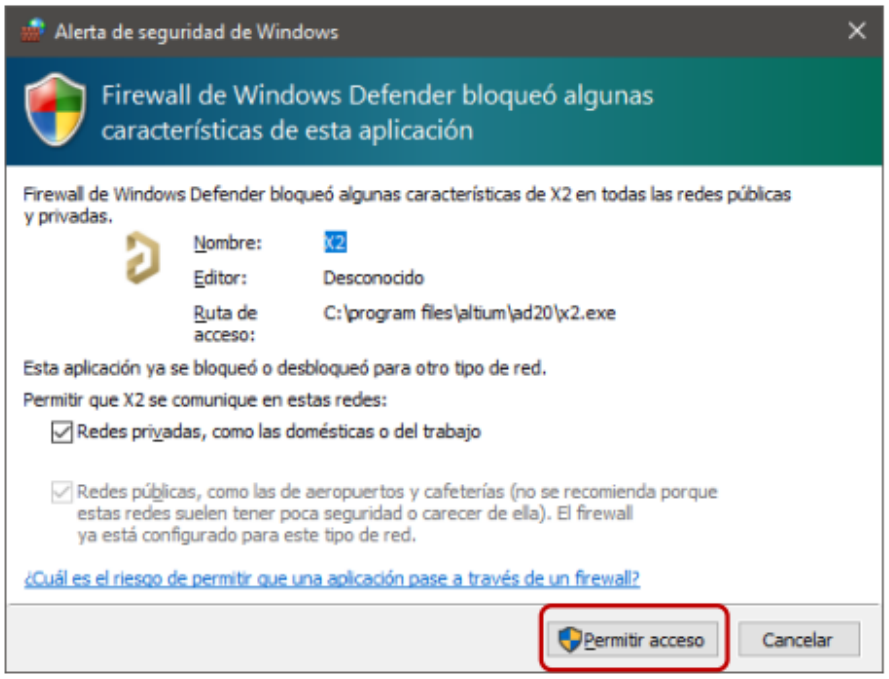

Figura 87: Instalador Altium Designer

11. Seleccionar *Sign In*.

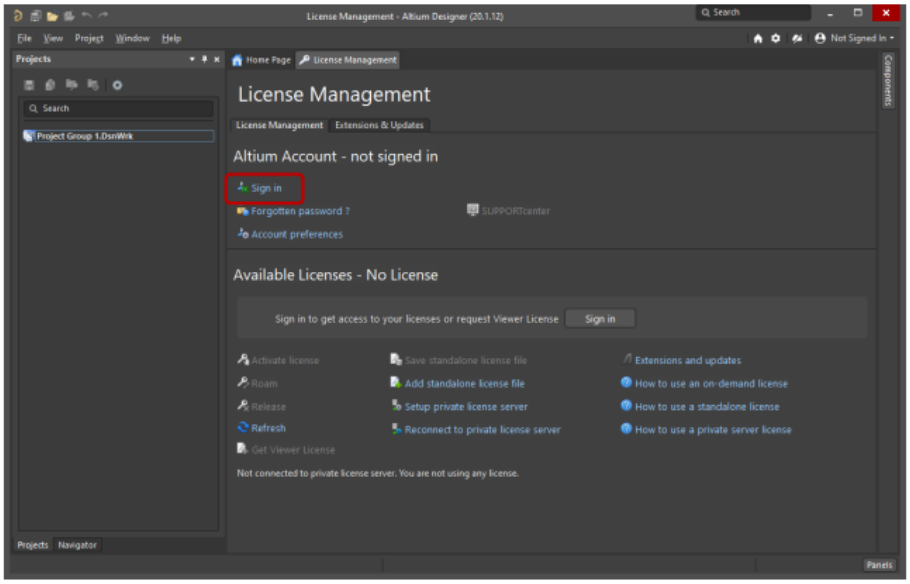

Figura 88: Instalador Altium Designer

12. Llenar campos con usuario y contraseña. Luego habilitar inicio automático.

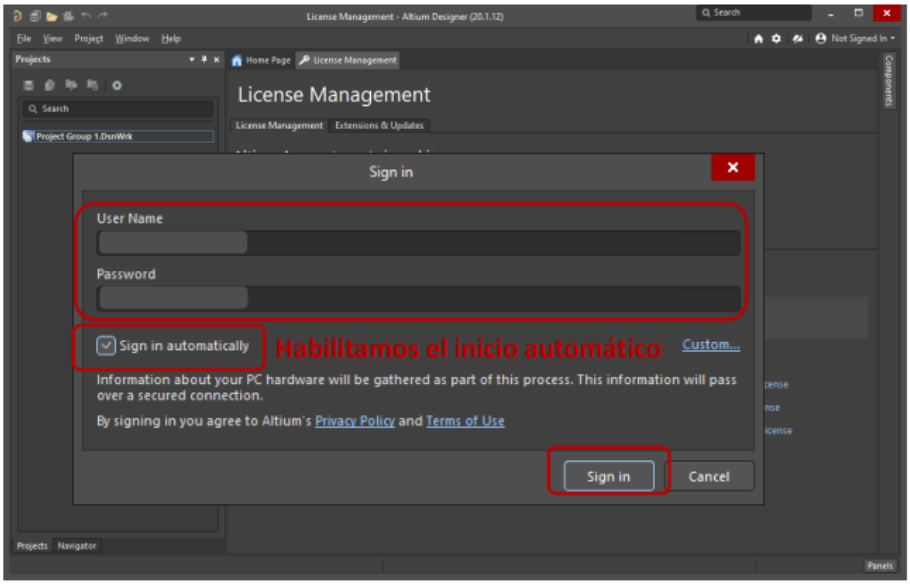

Figura 89: Instalador Altium Designer

13. Si se despliega una ventana para updates, presionar NO.

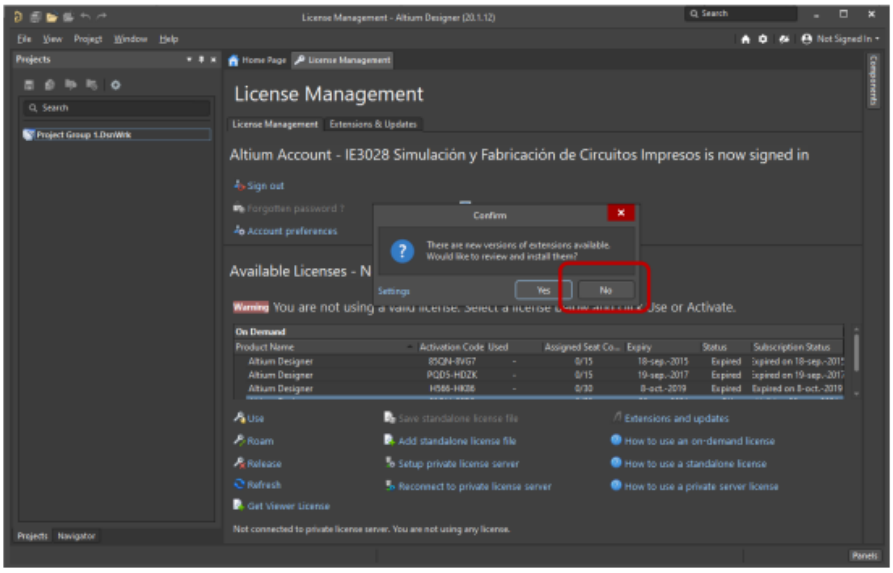

Figura 90: Instalador Altium Designer

14. Seleccionar licencia y presionar *USE*.

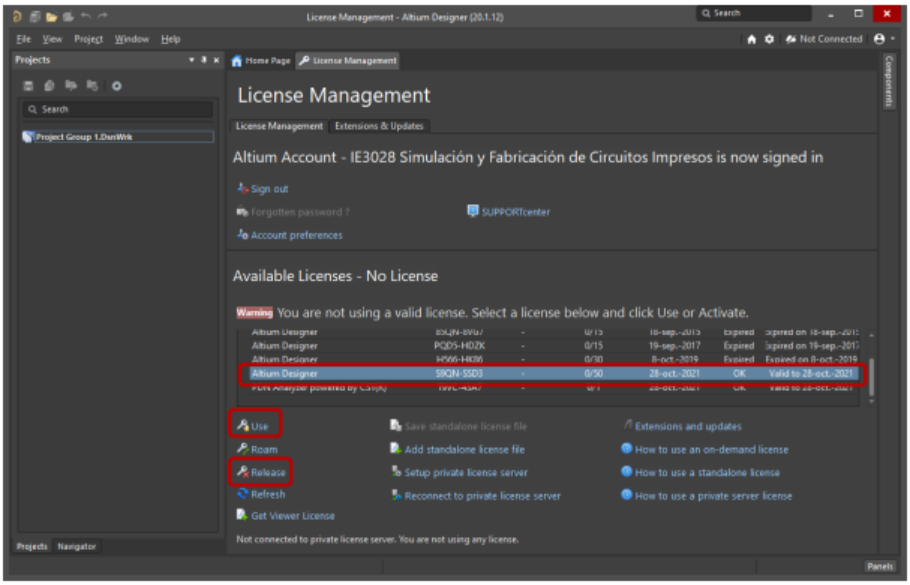

Figura 91: Instalador Altium Designer

15. Para últimas actualizaciones presionar *Extensions and updates*.

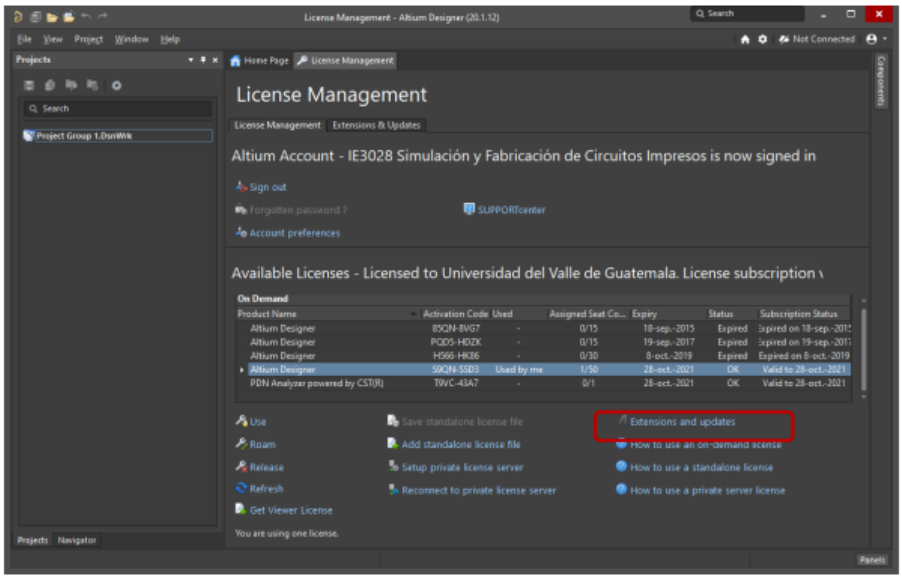

Figura 92: Instalador Altium Designer

16. Presionar botón de engrane.

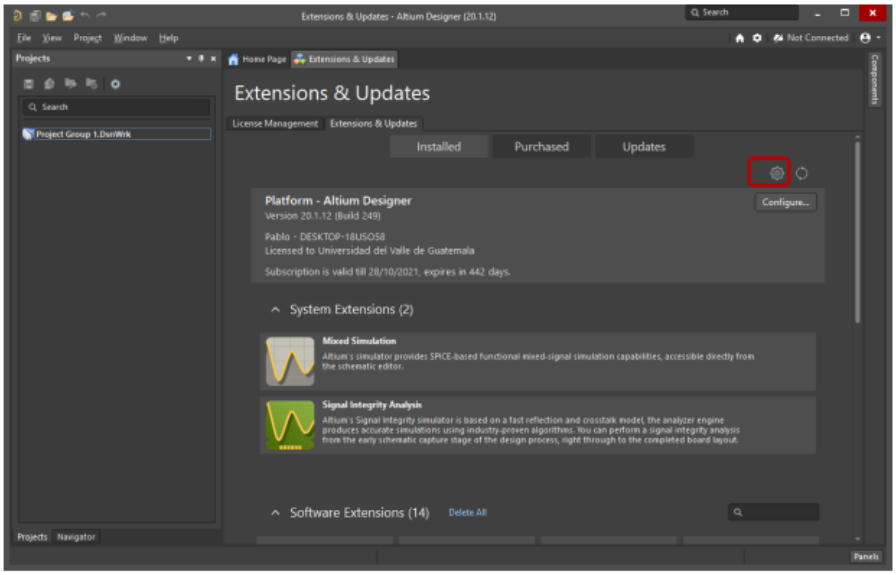

Figura 93: Instalador Altium Designer

17. En preferencias, seleccionar *Global installation service*, luego presionar *Apply* y *OK*.

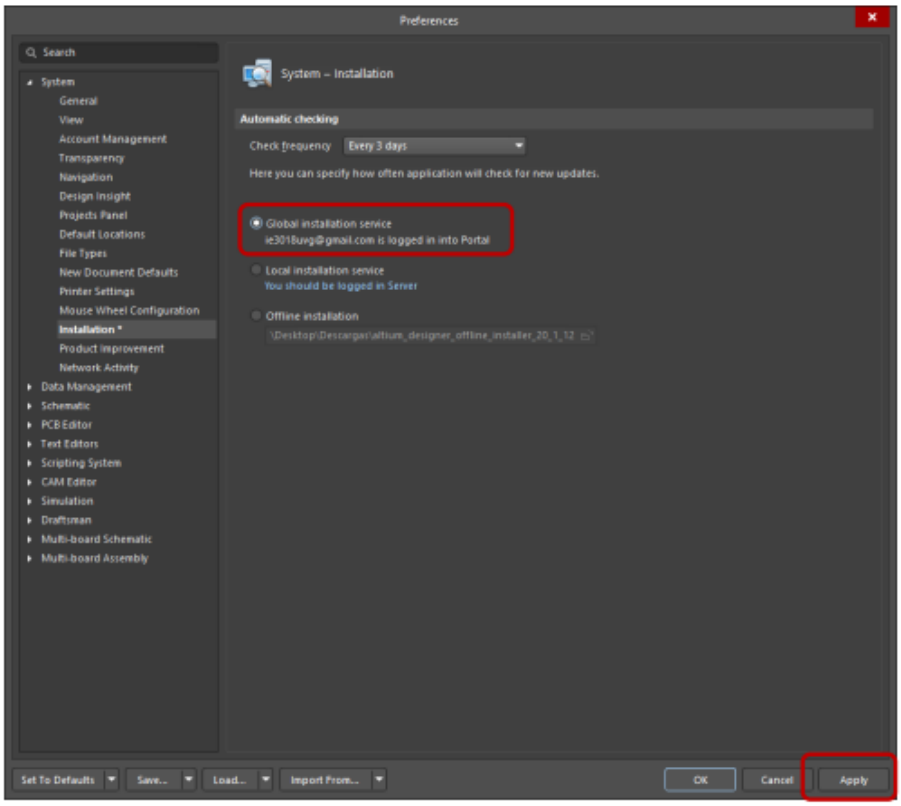

Figura 94: Instalador Altium Designer

18. Seleccionar pestaña de *UPDATES* y seleccionar el que se despliegue.

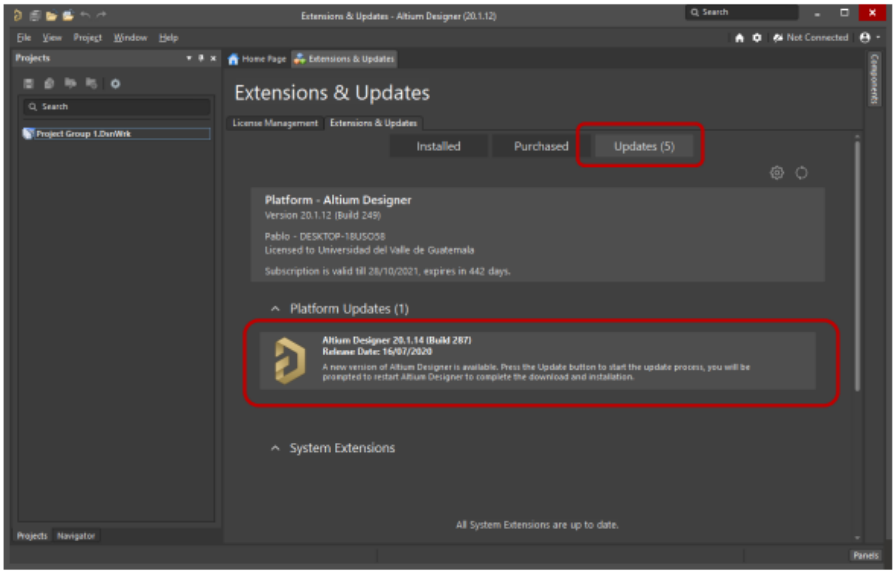

Figura 95: Instalador Altium Designer

19. Seleccionar *UPDATE*.

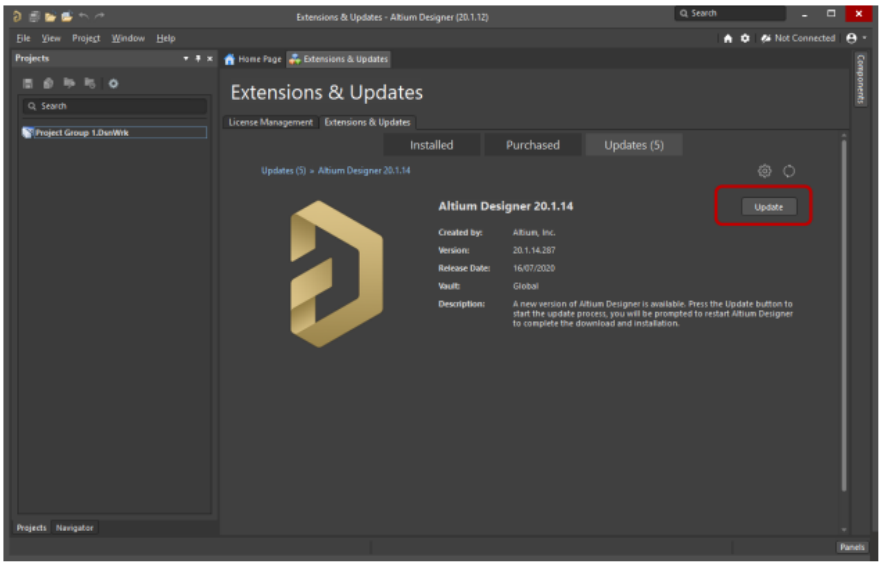

Figura 96: Instalador Altium Designer

20. Seleccionar *YES*

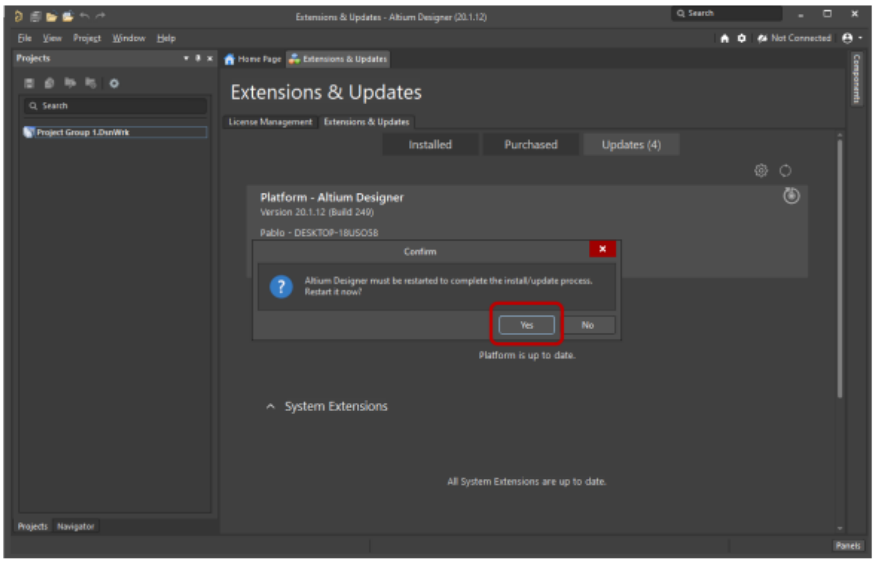

Figura 97: Instalador Altium Designer

21. Descargar librerías actualizadas en: https://designcontent.live.altium.com/#UnifiedComponents

22. Descomprimir la carpeta y copiar en el directorio: C:\Users\Public\Documents\Altium\AD20\Library

23. En Altium, dirigirse a barra de herramientas y seleccionar donde dice Componentes.

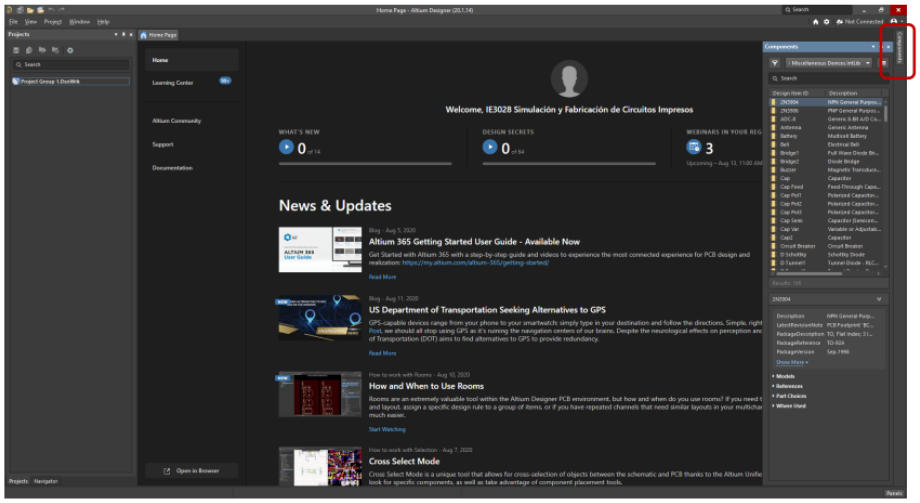

Figura 98: Altium Designer

24. Seleccionar siguiente ícono y presionar *File-Based Libraries Preferences*.

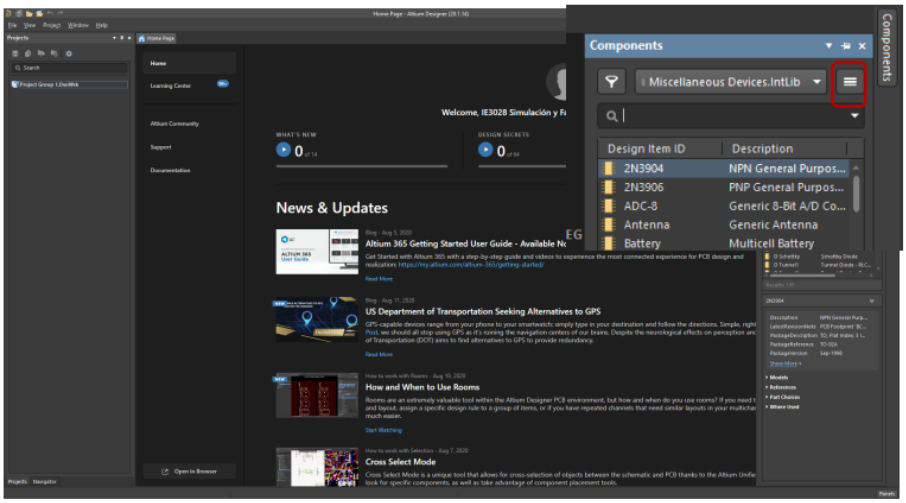

Figura 99: Altium Designer

25. Presionar pestaña llamada *Search Path*

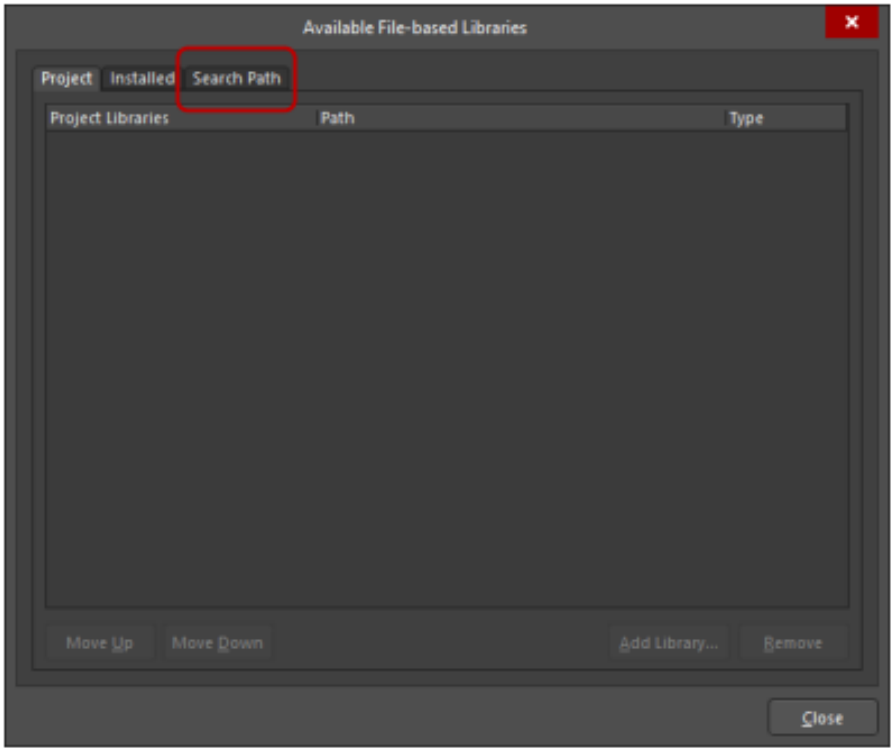

Figura 100: Altium Designer

26. Presionar ícono de *Paths*.

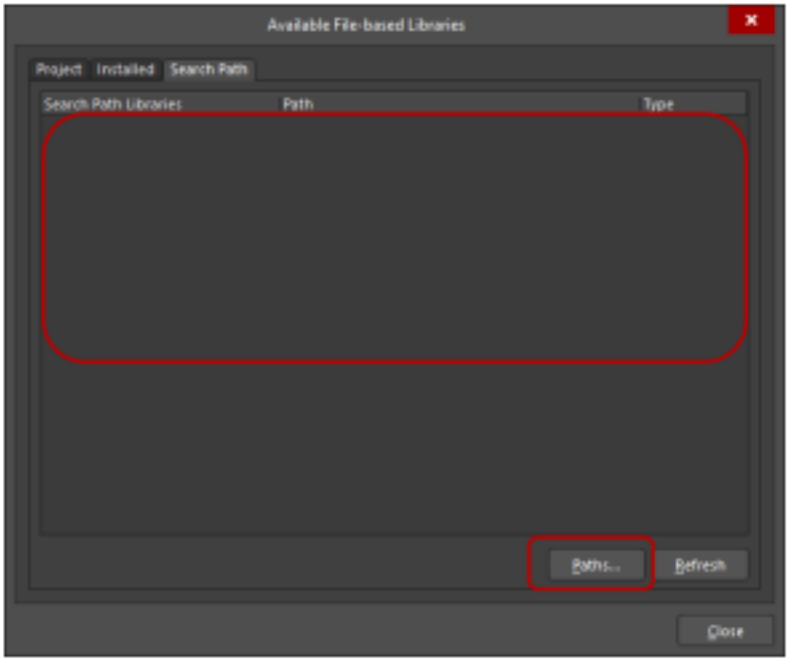

Figura 101: Altium Designer

27. Seleccionar ícono de *Add*.

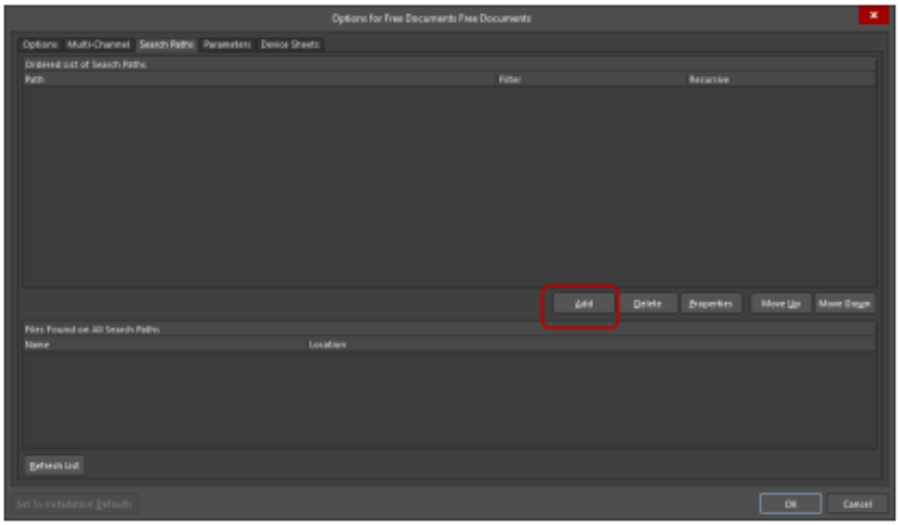

Figura 102: Altium Designer

28. Pegar ruta del directorio donde están las librerías, presionar *Refresh List* y *OK*.

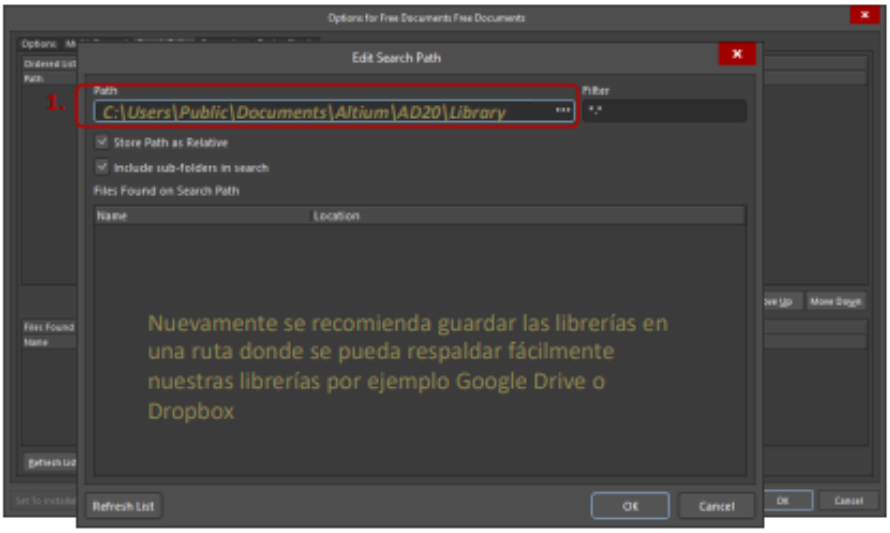

Figura 103: Altium Designer

29. Presionar *Refresh List*, luego *OK*.

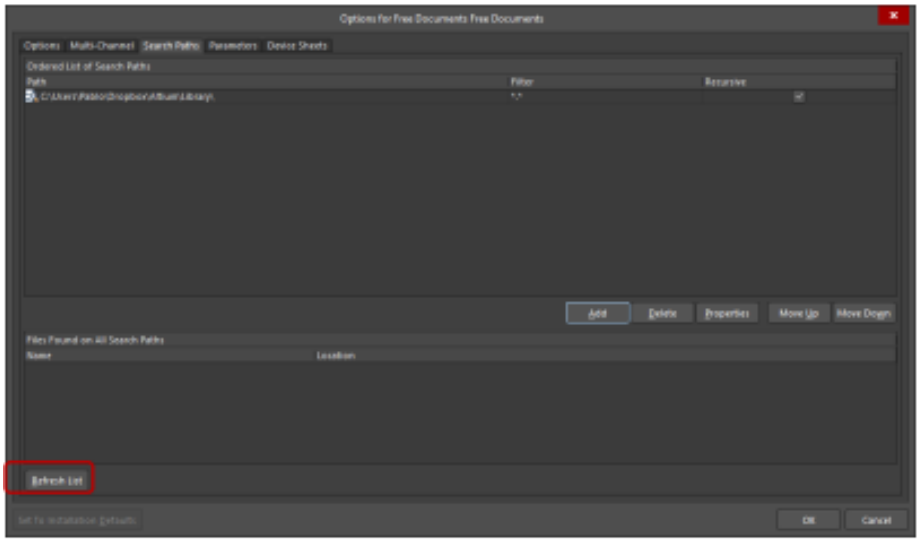

Figura 104: Altium Designer

30. Presionar *Refresh*, luego *Close*.

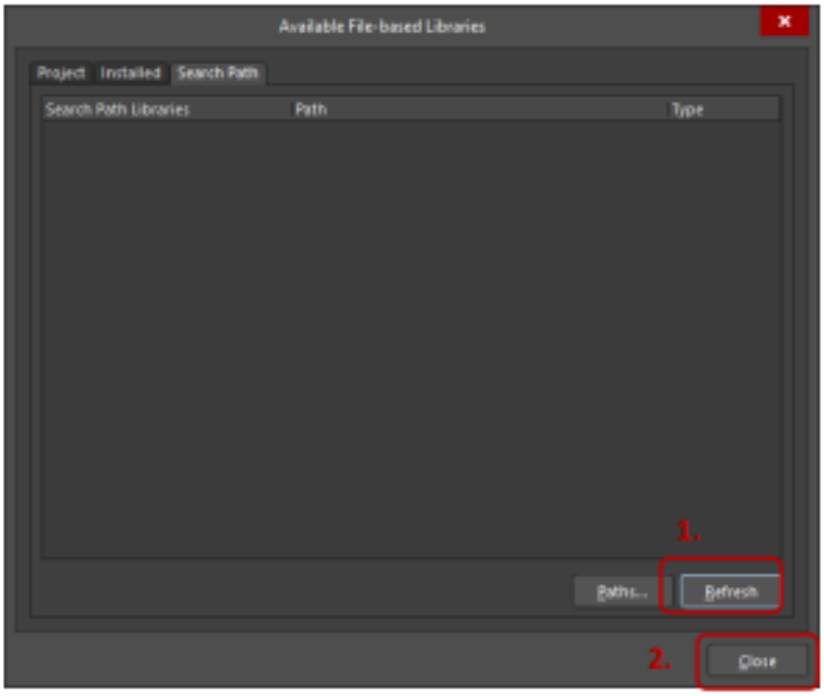

Figura 105: Altium Designer

14.1.2. Instalación *Raspberry Pi OS*

1. Instalar *Imager* desde el sitio oficial: https://www.raspberrypi.com/software/
2. Insertar tarjeta SD en el computador.
3. Abrir el *Imager*

Figura 106: *Raspberry Pi Imager*

4. Seleccionar sistema operativo.

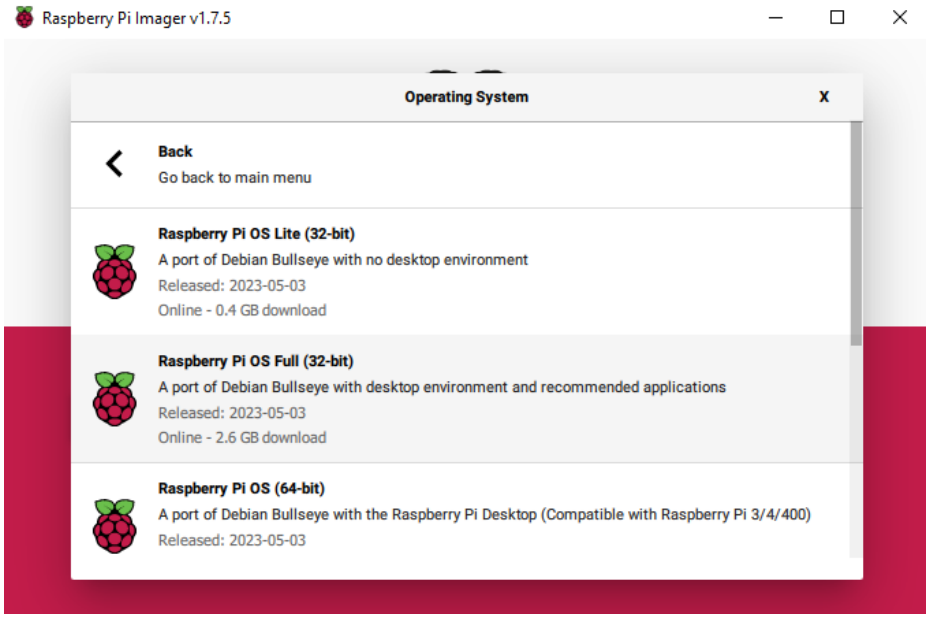

Figura 107: *Raspberry Pi Imager*

5. Seleccionar tarjeta SD
6. Presionar *Write y esperar.*
7. Insertar SD en *Raspberry Pi*

La *Raspberry Pi* 3B+ tiene compatibilidad con la versión 6 y las versiones posteriores del *Raspberry Pi OS*. En caso de que se desee utilizar versiones anteriores del sistema operativo en el modelo 3B+, es necesario hacer una actualización del kernel y el *firmware* en modelos previos (3B por ejemplo) con los siguientes comandos:

```
sudo apt update
sudo apt full-upgrade
```

14.1.3. Instalación *Wiring Pi*

- Se descargan e instalan actualizaciones.

```
sudo apt-get update
sudo apt-get upgrade
```

- Se clona repositorio donde se encuentran todos los archivos de la librería.

```
git clone https://github.com/WiringPi/WiringPi.git
```

- Se ejecuta.

```
cd WiringPi
./build
```

- Se verifica la correcta instalación.

```
gpio -v
```

- Si se despliega el siguiente texto, significa que la instalación fue exitosa.

```
pi@raspberrypi:~ $ gpio -v
gpio version: 2.70
Copyright (c) 2012-2018 Gordon Henderson
This is free software with ABSOLUTELY NO WARRANTY.
For details type: gpio -warranty

Raspberry Pi Details:
  Type: Pi 3, Revision: 02, Memory: 1024MB, Maker: Sony
  * Device tree is enabled.
  *--> Raspberry Pi 3 Model B Rev 1.2
  * This Raspberry Pi supports user-level GPIO access.
```

Figura 108: Instalación WiringPi

14.1.4. Instalación de paquetes para micrófono SPH0645LM4H

- Se descargan e instalan actualizaciones.

```
sudo apt-get update
sudo apt-get upgrade
```

- Se descarga e instala Python3 (en caso de no tenerlo instalado).

```
sudo apt install python3-pip
```

- Instalar y actualizar paquetes de Python para la compatibilidad con los paquetes de Adafruit.

```
sudo pip3 install --upgrade adafruit-python-shell
```

- Se descargan los *scripts* de Adafruit.

```
wget https://raw.githubusercontent.com/adafruit
/Raspberry-Pi-Installer-Scripts/master/i2smic.py
```

- Se ejecuta el *script* para instalar todo lo necesario para utilizar el micrófono.

```
sudo python3 i2smic.py
```

- Verificar funcionamiento con el siguiente comando. Si se despliega algo como en la imagen, la *Raspberry Pi* reconoce el dispositivo.

```
arecord -l
```

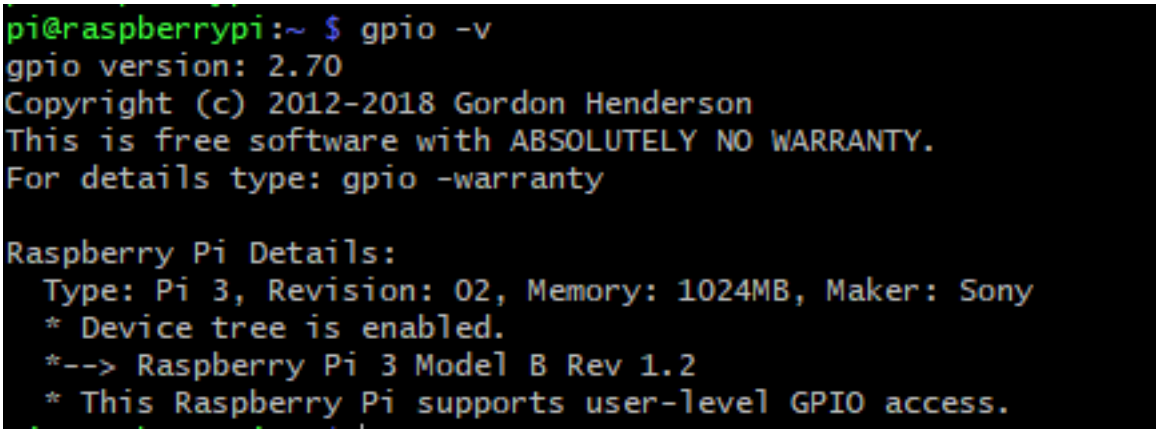

Figura 109: Reconocimiento del micrófono I^2S

- Utilizar siguiente comando para empezar a grabar.

```
arecord -D plughw:0 -c1 -r 48000
-f S32_LE -t wav -V mono -v file.wav
```

14.1.5. Instalación de paquetes para amplificador MAX98357

- Ejecutar el siguiente comando y seleccionar 'Si' en todas las opciones.

```
curl -sS https://raw.githubusercontent.com
/adafruit/Raspberry-Pi-Installer-Scripts
/master/i2samp.sh | bash
```

- Reiniciar el sistema.

```
sudo reboot
```

- Verificar que la *Raspberry Pi* reconoce el módulo.

```
aplay -l
```

- Verificar funcionamiento.

```
speaker-test -c2
```

14.2. Símbolos

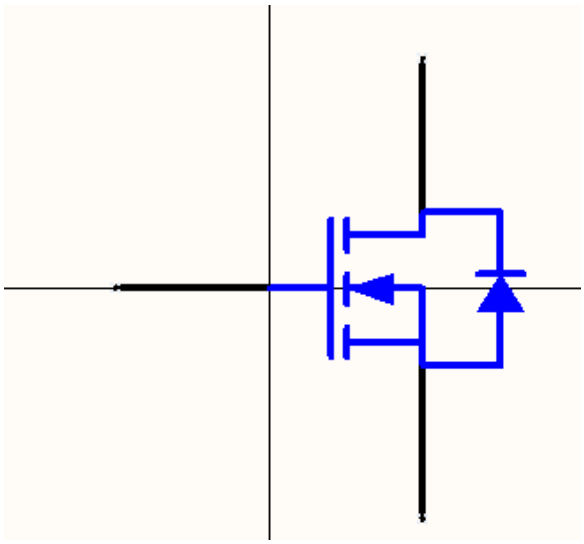

Figura 110: Símbolo de esquemático de 2N7000

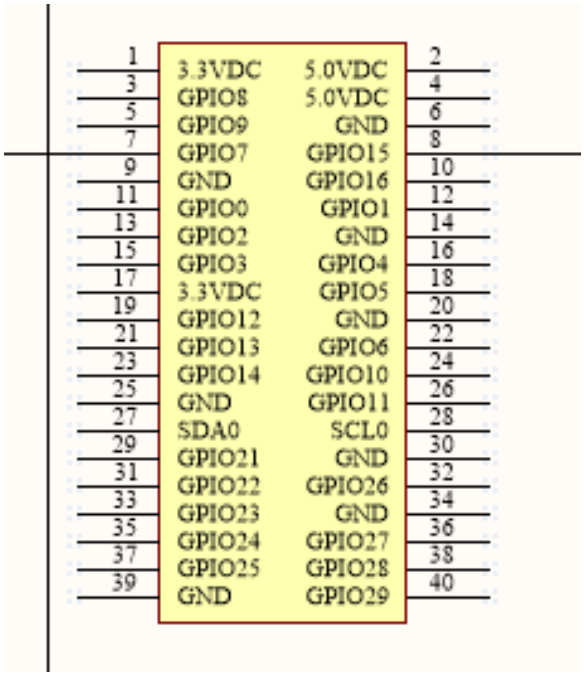

Figura 111: Símbolo de esquemático de *Header*hembra 2x20

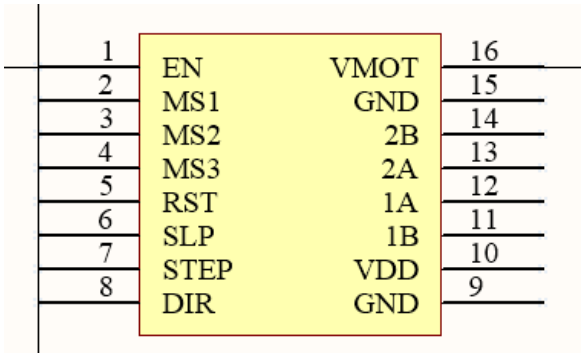

Figura 112: Símbolo de esquemático de *Driver* A4988

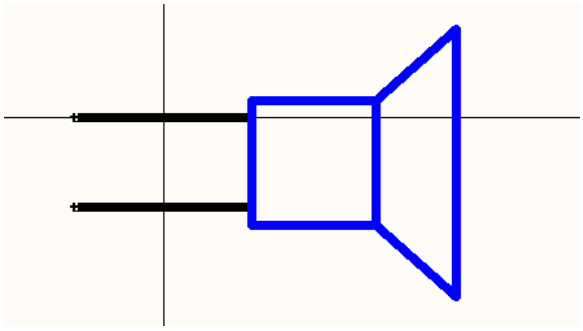

Figura 113: Símbolo de esquemático de *Buzzer* pasivo

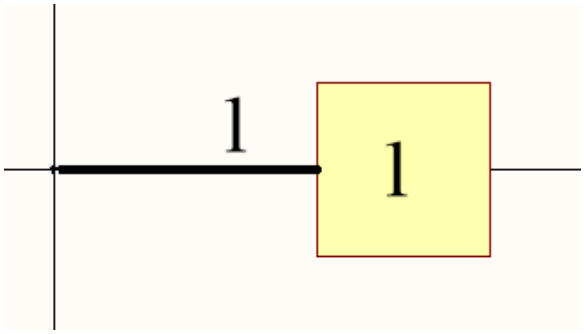

Figura 114: Símbolo de esquemático de *Pinhead 1x1*

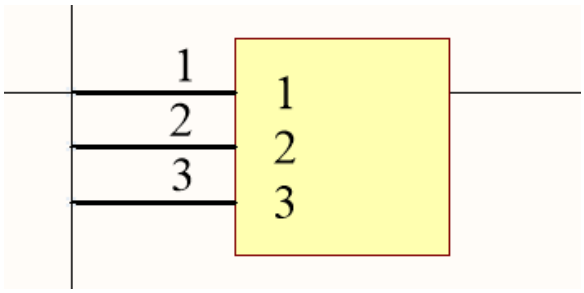

Figura 115: Símbolo de esquemático de *Pinhead 1x3*

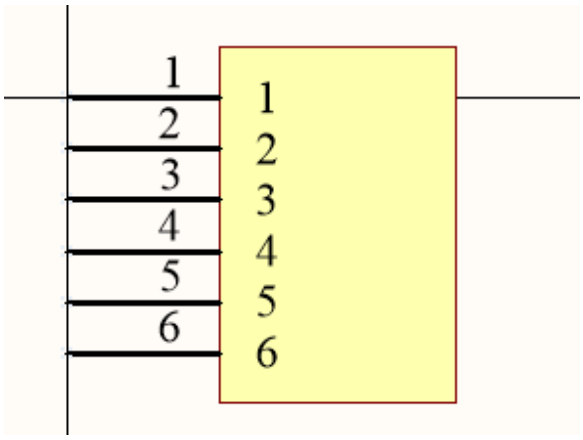

Figura 116: Símbolo de esquemático de *Pinhead 1x6*

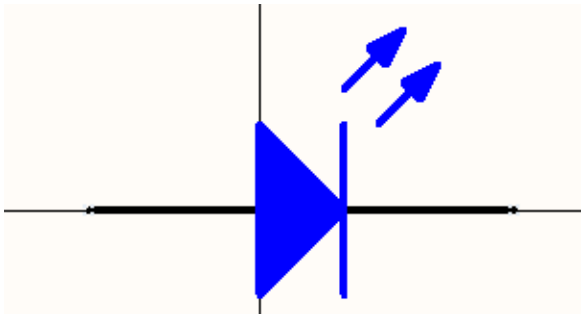

Figura 117: Símbolo de esquemático de LED

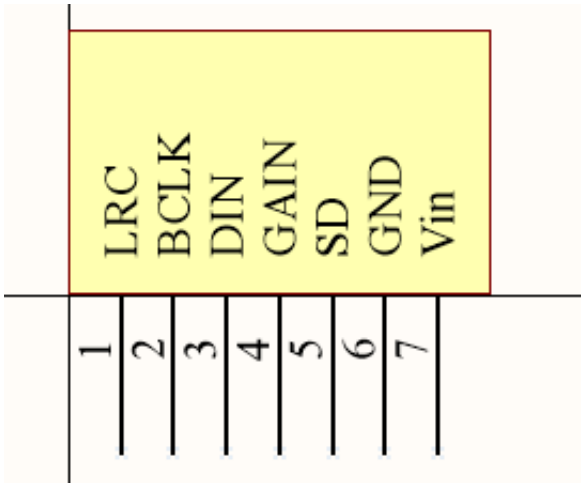

Figura 118: Símbolo de esquemático de amplificador MAX98357

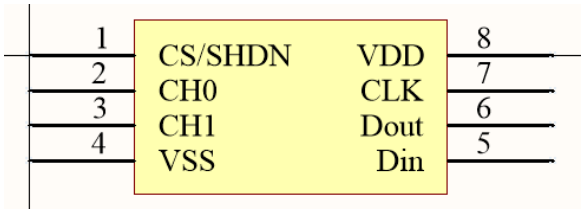

Figura 119: Símbolo de esquemático de MCP 3002

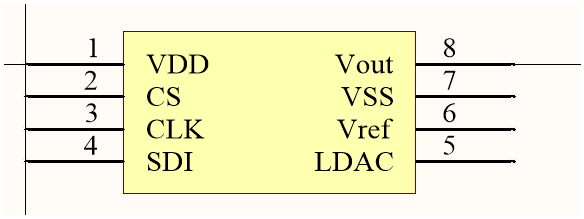

Figura 120: Símbolo de esquemático de MCP 4921

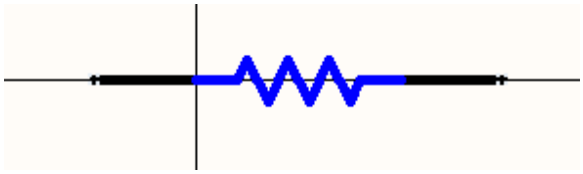

Figura 121: Símbolo de esquemático de resistencia

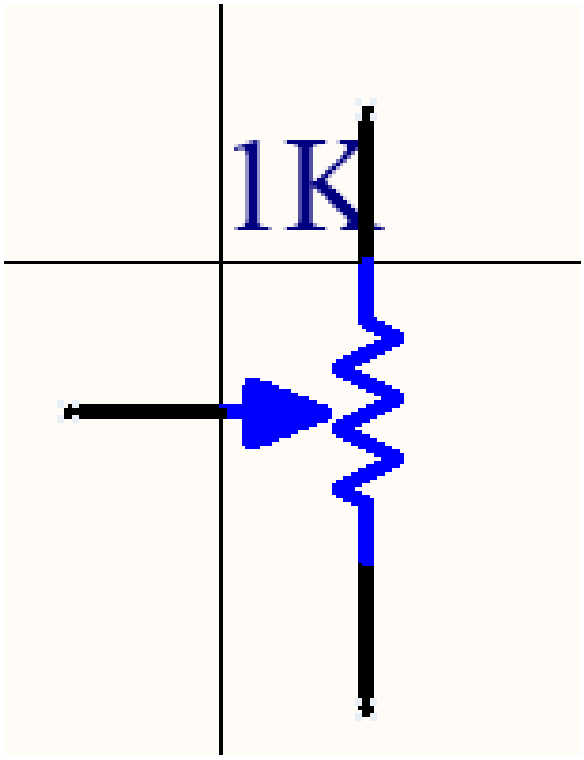

Figura 122: Símbolo de esquemático de potenciómetro

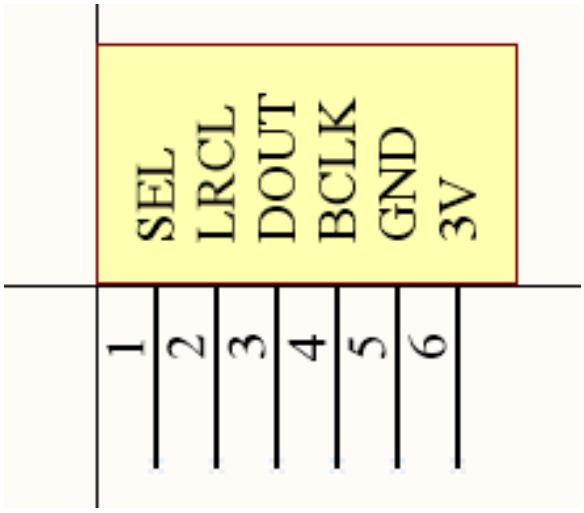

Figura 123: Símbolo de esquemático de micrófono SPH0645LM4H

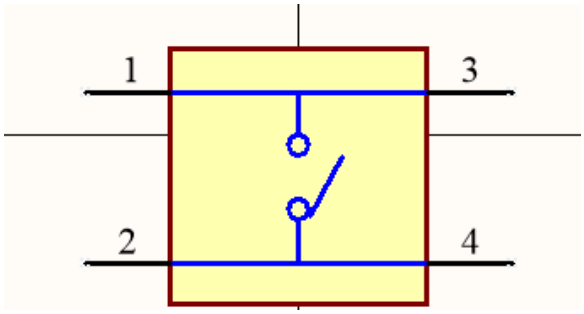

Figura 124: Símbolo de esquemático de pulsador

14.3. *Footprints*

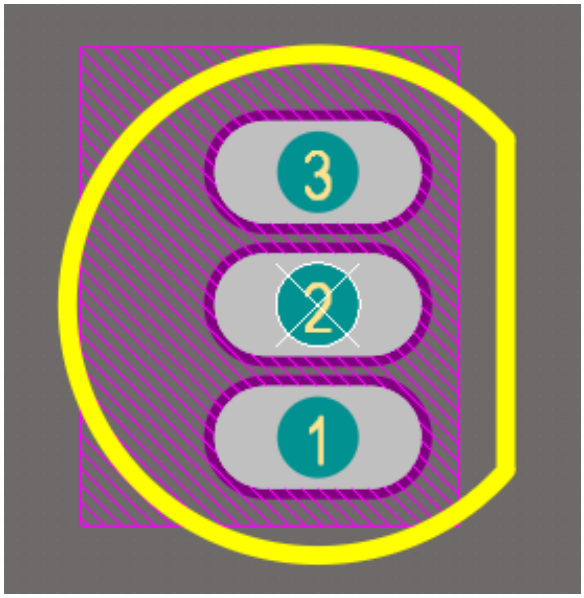

Figura 125: *Footprint* de 2N7000

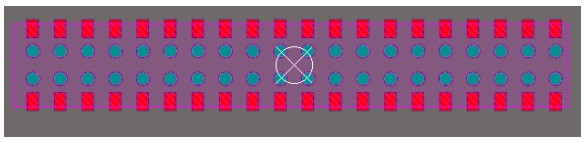

Figura 126: *Footprint* de *Header*hembra 2x20

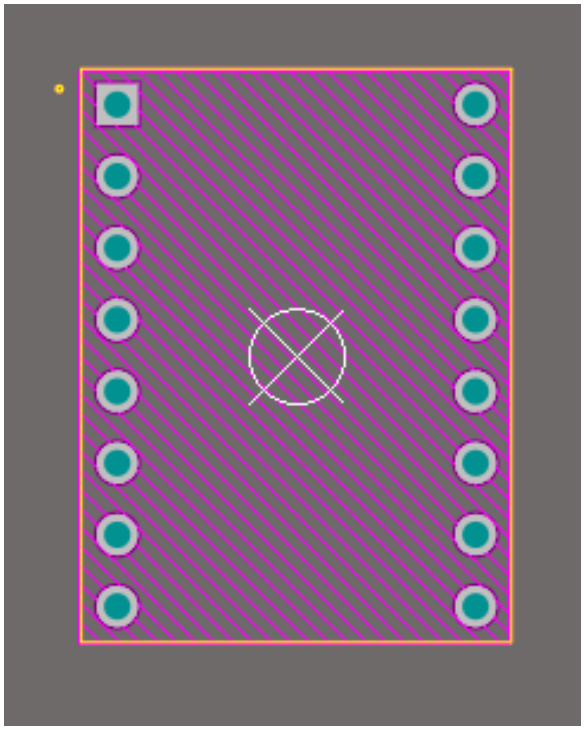

Figura 127: *Footprint* de *Driver* A4988

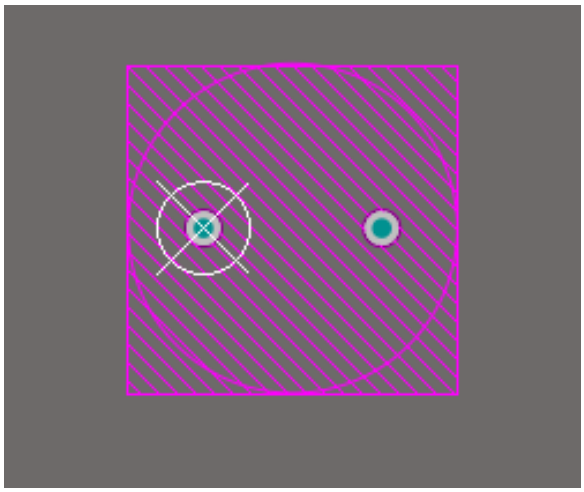

Figura 128: *Footprint* de *buzzer*

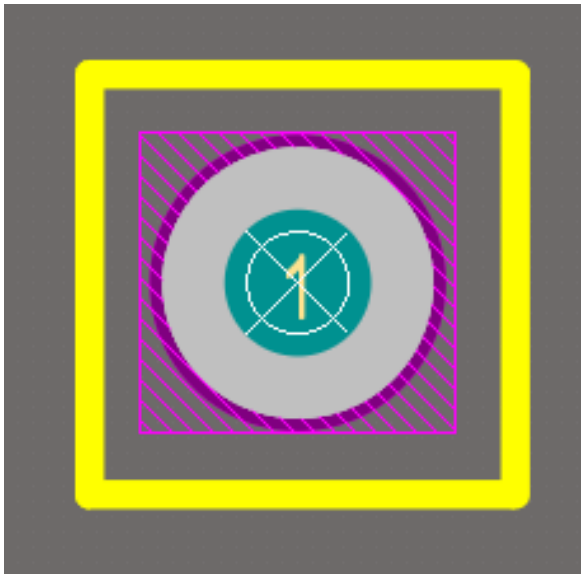

Figura 129: *Footprint* de *Pinhead 1x1*

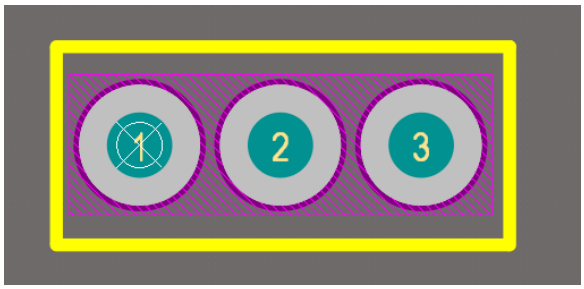

Figura 130: *Footprint* de *Pinhead 1x3*

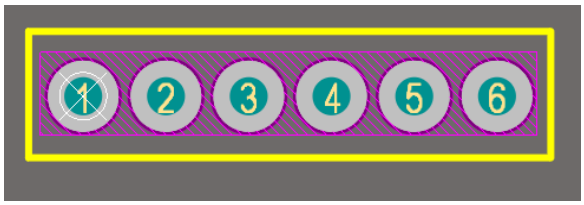

Figura 131: *Footprint* de *Pinhead 1x6*

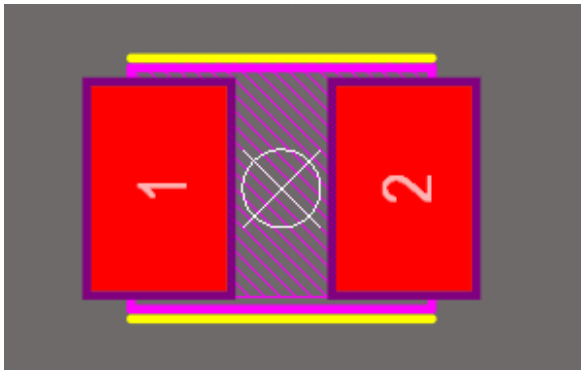

Figura 132: *Footprint* de LED

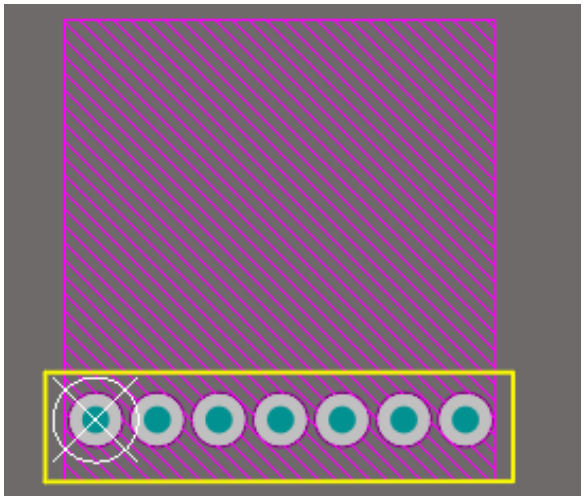

Figura 133: *Footprint* de amplificador MAX98357

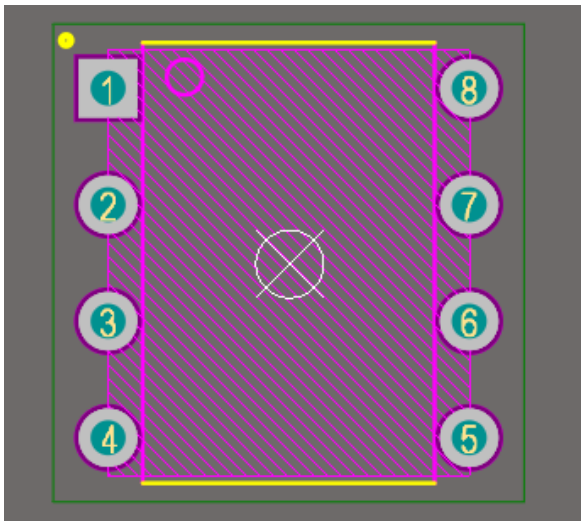

Figura 134: *Footprint* de MCP 3002

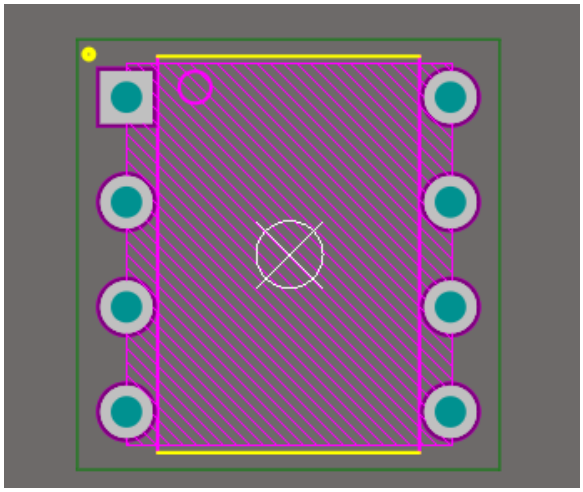

Figura 135: *Footprint* de MCP 4921

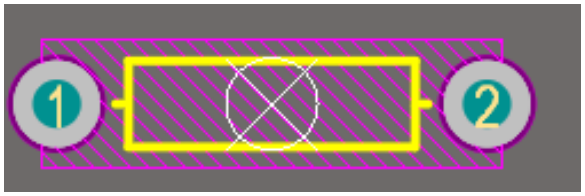

Figura 136: *Footprint* resistencia THT

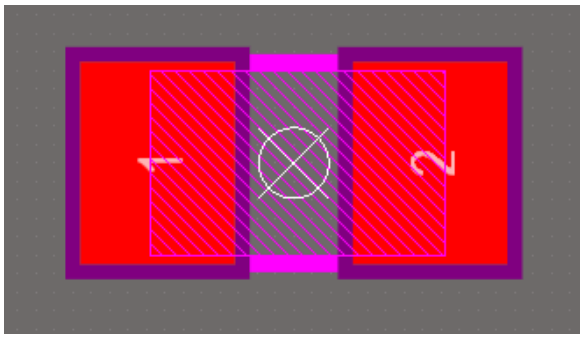

Figura 137: *Footprint* de resistencia SMD

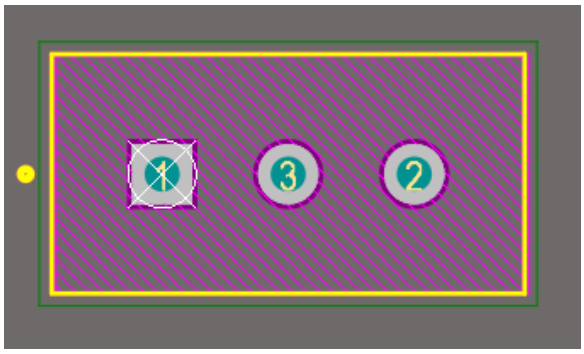

Figura 138: *Footprint* de potenciómetro

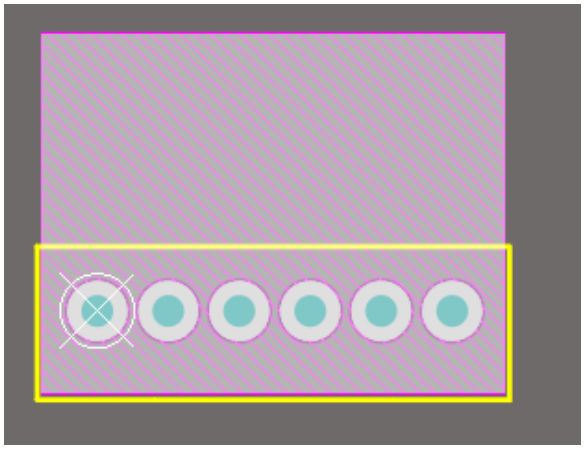

Figura 139: *Footprint* de micrófono SPH0645LM4H

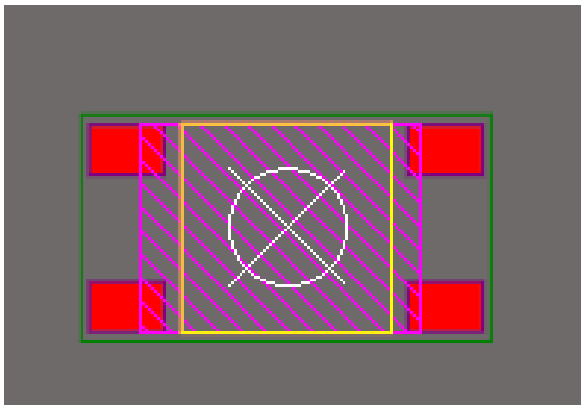

Figura 140: *Footprint* de pulsador

14.4. Modelos 3D

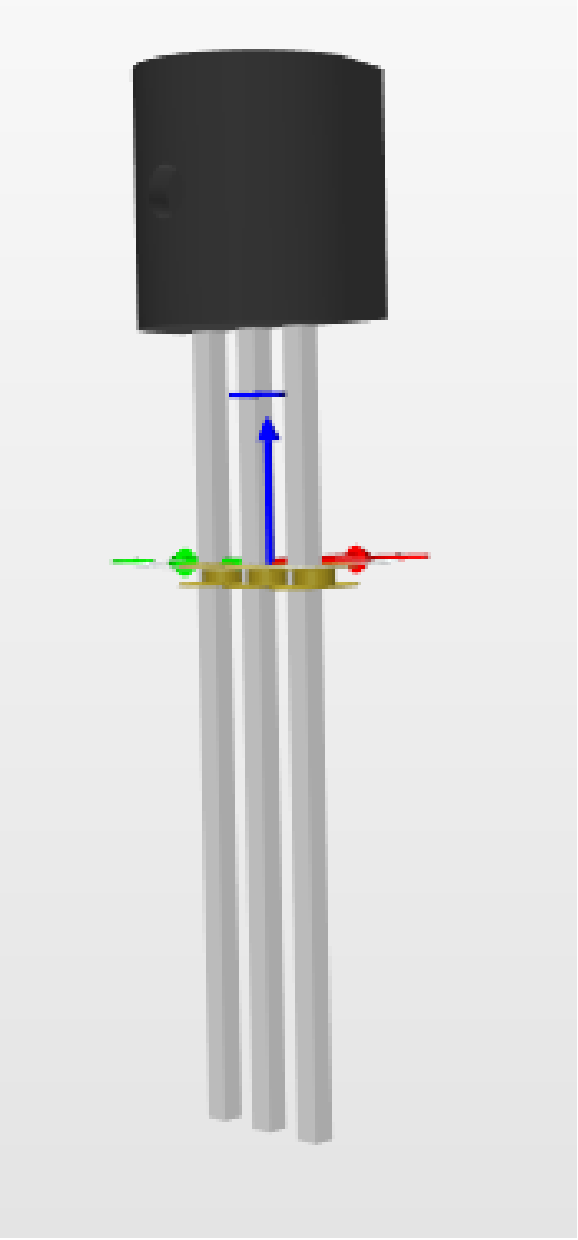

Figura 141: Modelo 3D de 2N7000

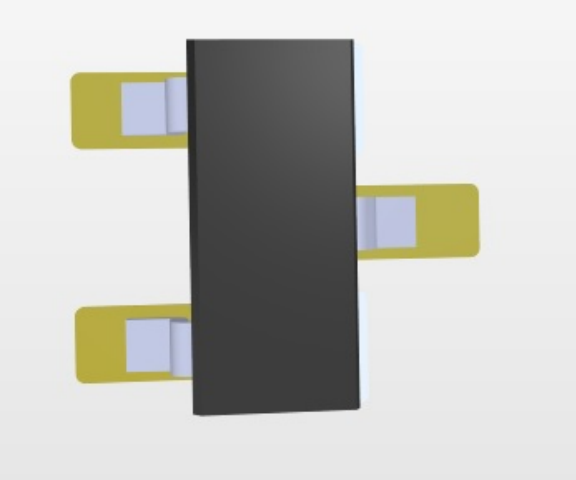

Figura 142: Modelo 3D de 2N7002 SMD

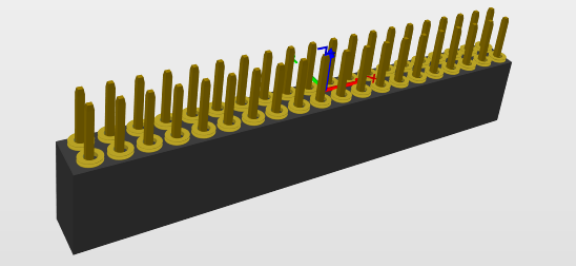

Figura 143: Modelo 3D de *Header* macho-hembra 2x20

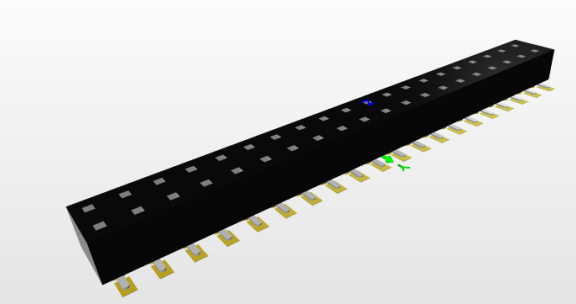

Figura 144: Modelo 3D de *Header* hembra 2x20

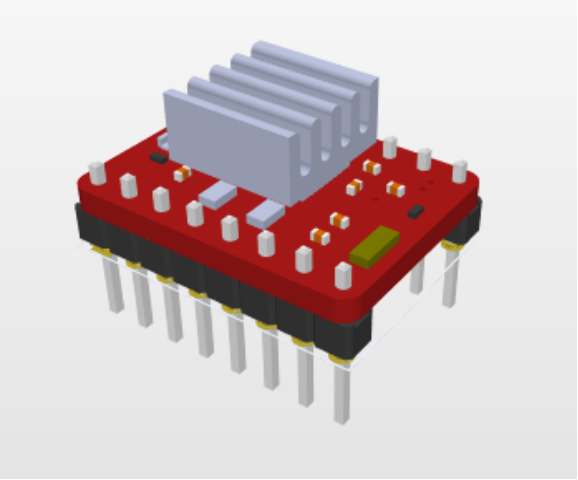

Figura 145: Modelo 3D de *Driver* A4988

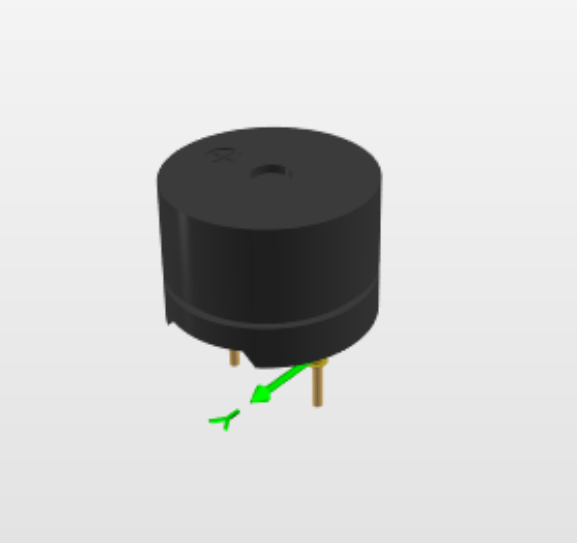

Figura 146: Modelo 3D de *buzzer*

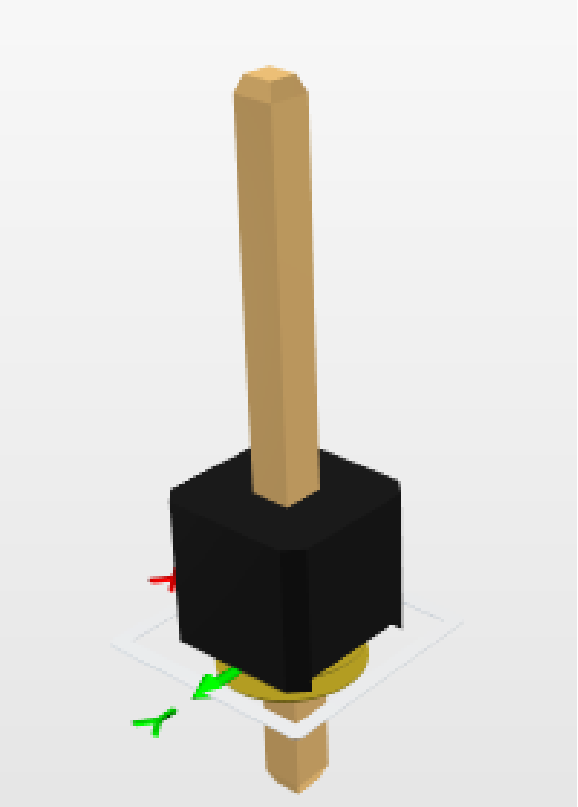

Figura 147: Modelo 3D de *Pinhead 1x1*

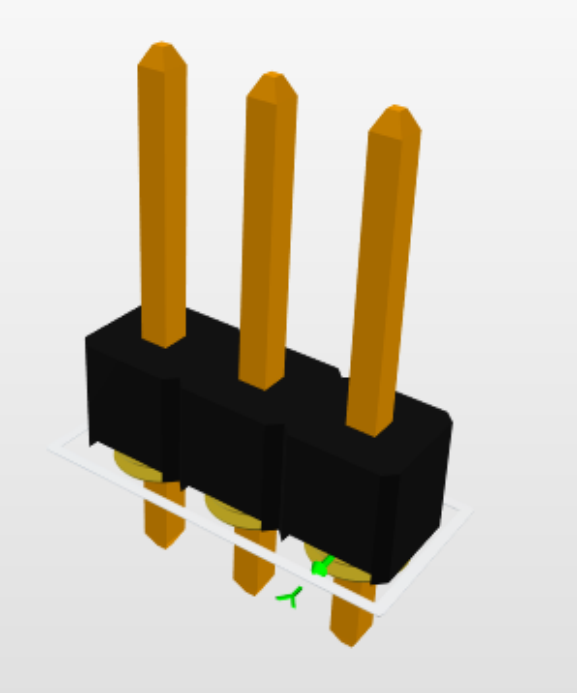

Figura 148: Modelo 3D de *Pinhead 1x3*

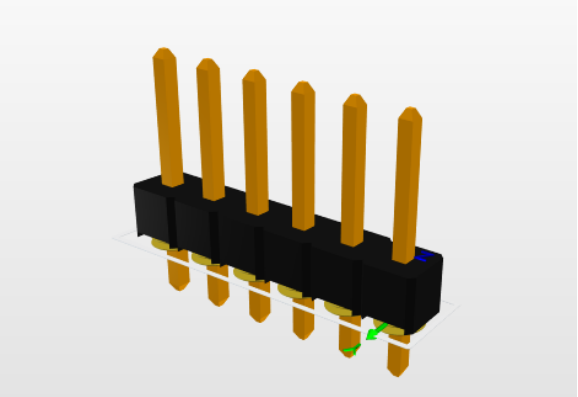

Figura 149: Modelo 3D de *Pinhead 1x6*

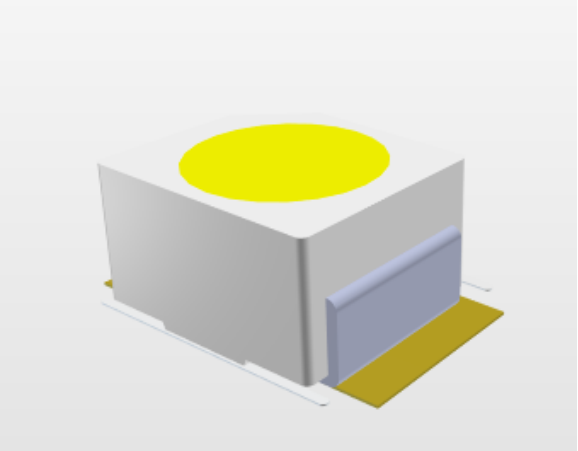

Figura 150: Modelo 3D de de LED

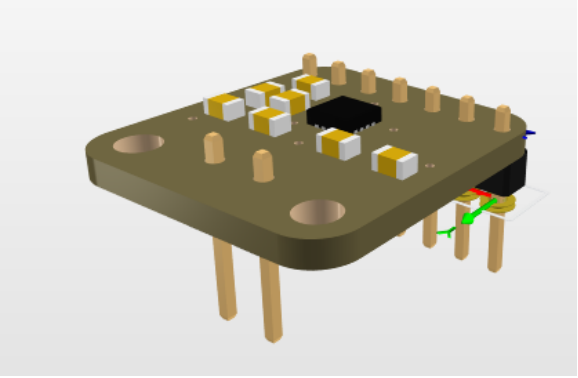

Figura 151: Modelo 3D de amplificador MAX98357

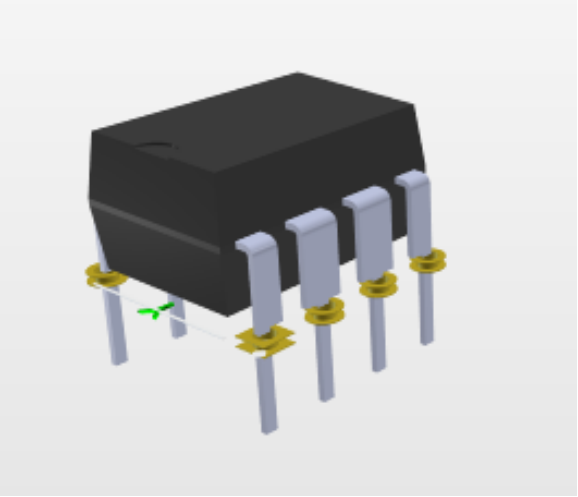

Figura 152: Modelo 3D de de MCP 3002

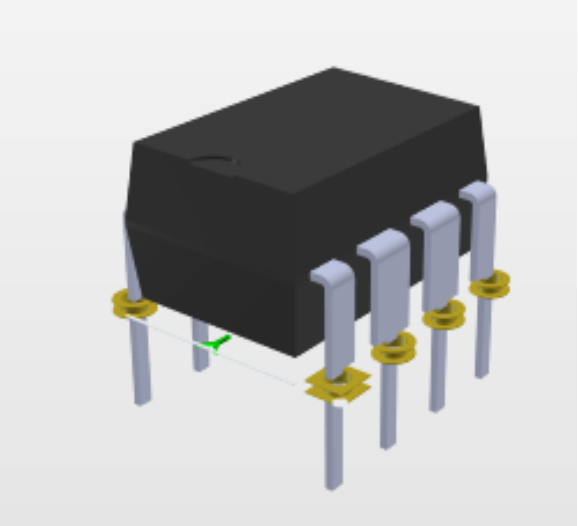

Figura 153: Modelo 3D de de MCP 4921

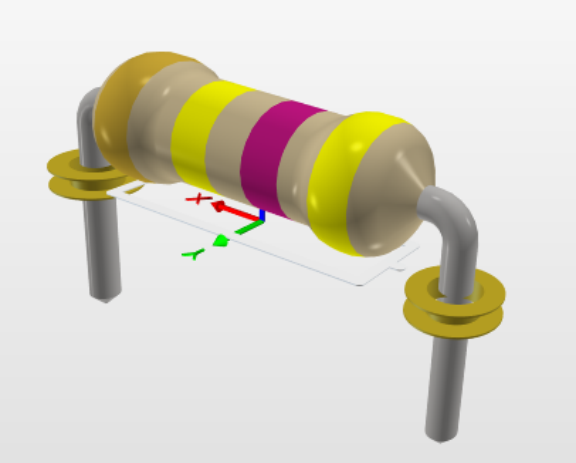

Figura 154: Modelo 3D de resistencia THT

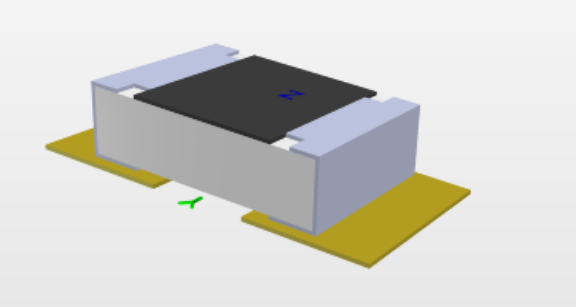

Figura 155: Modelo 3D de resistencia SMD

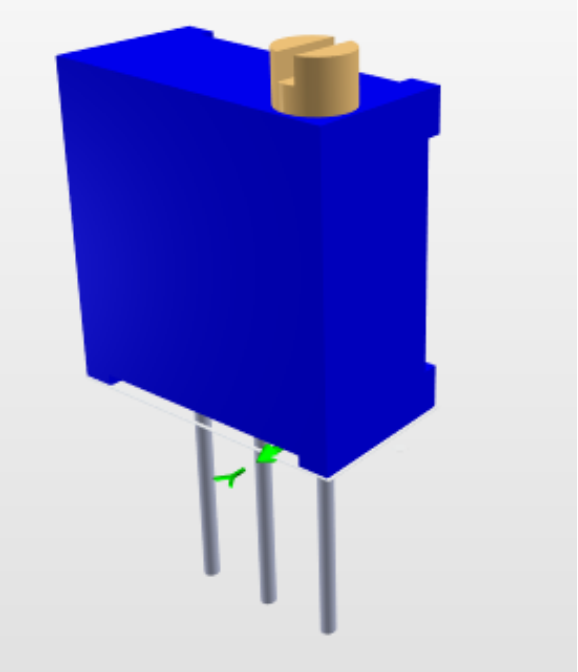

Figura 156: Modelo 3D de potenciómetro

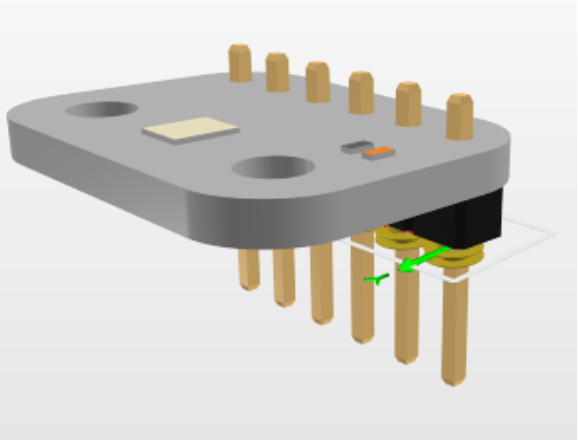

Figura 157: Modelo 3D de micrófono SPH0645LM4H

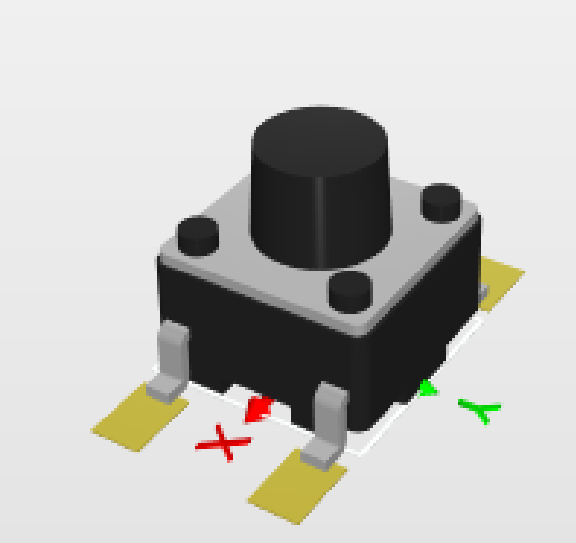

Figura 158: Modelo 3D de pulsador

14.5. Esquemáticos

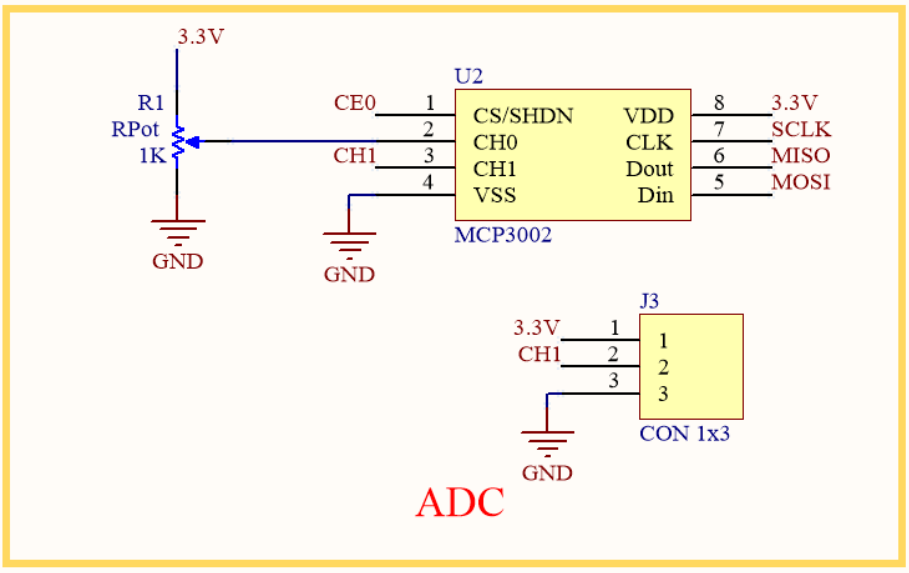

Figura 159: Conexiones para el ADC

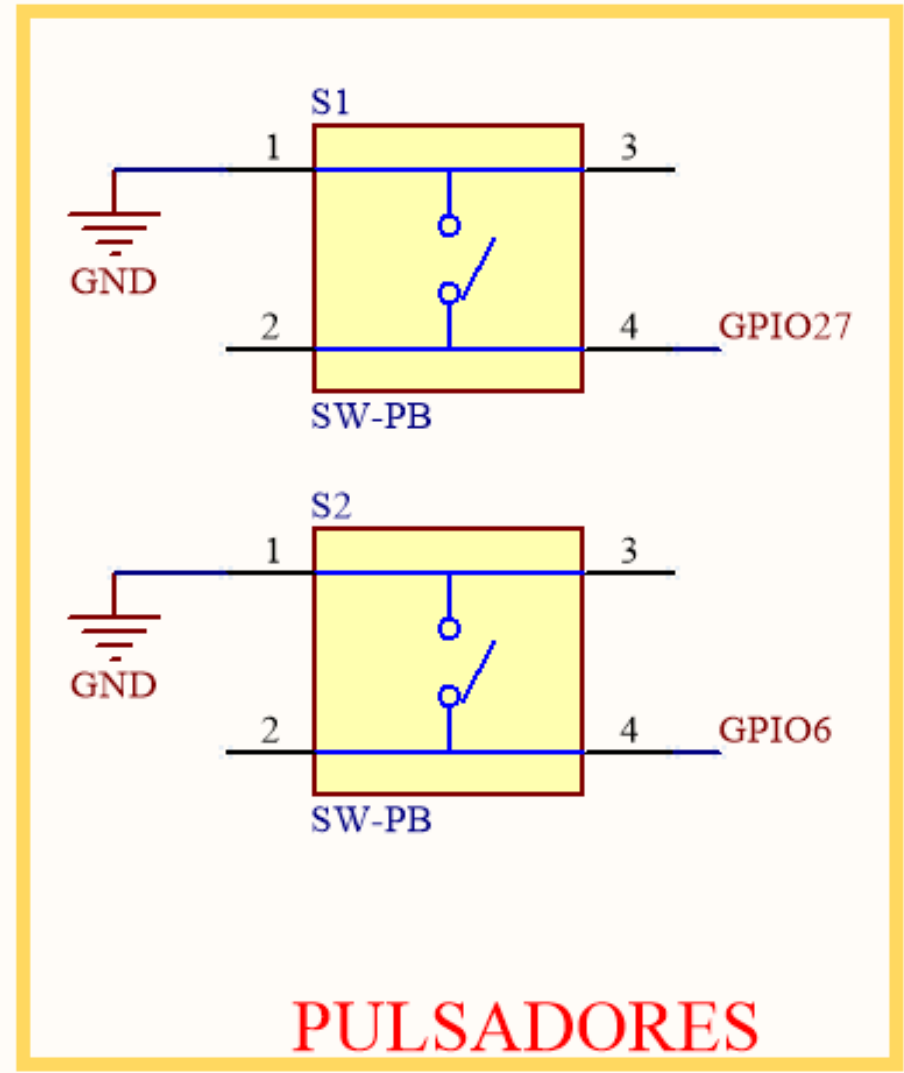

Figura 160: Conexiones para los pulsadores

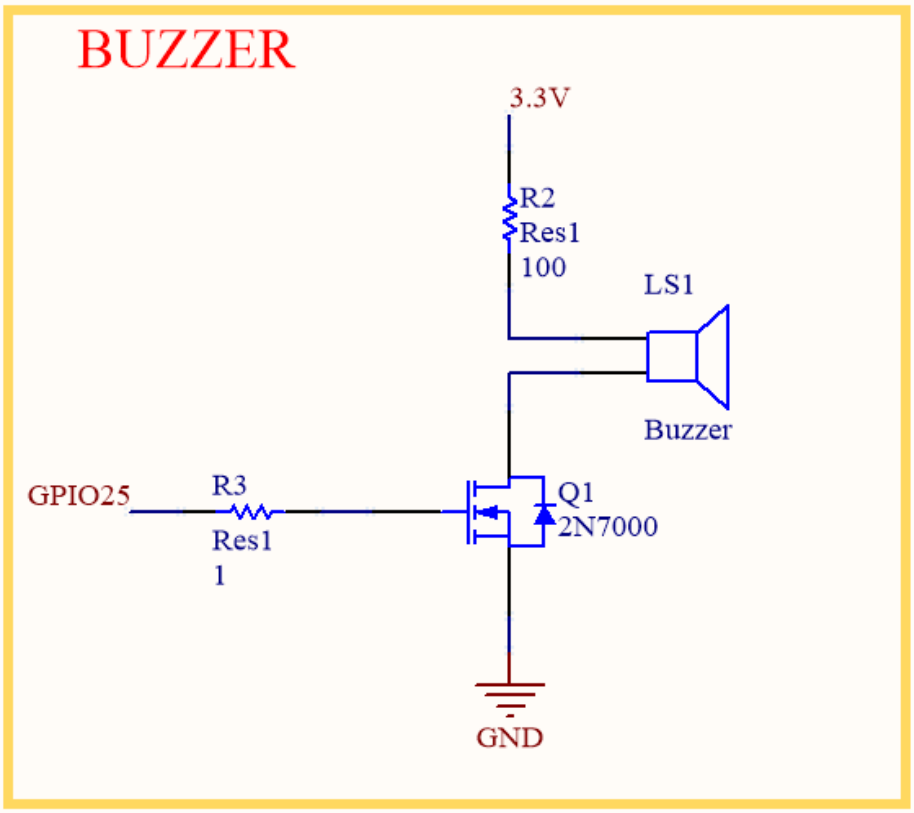

Figura 161: Conexiones para el *Buzzer*

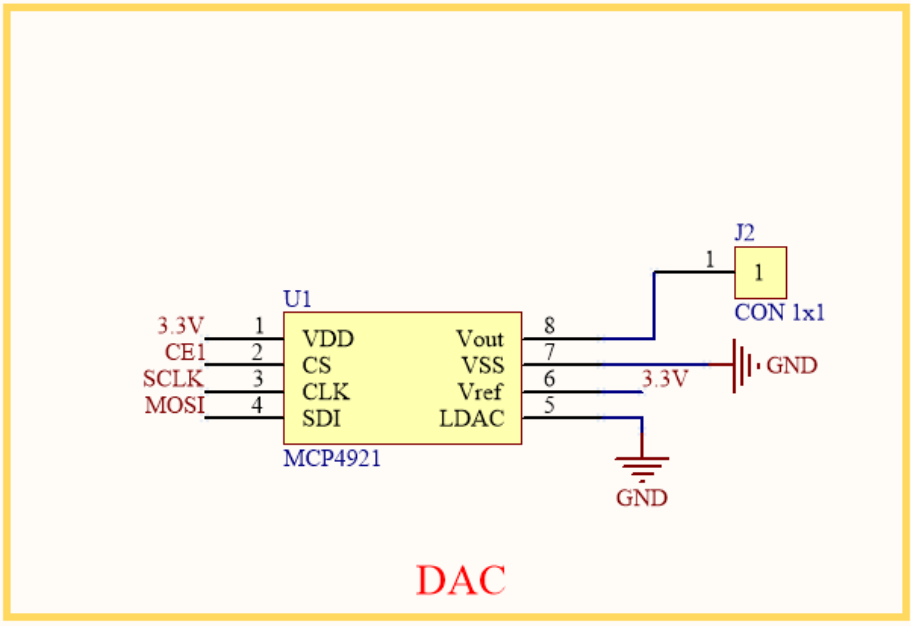

Figura 162: Conexiones para el DAC

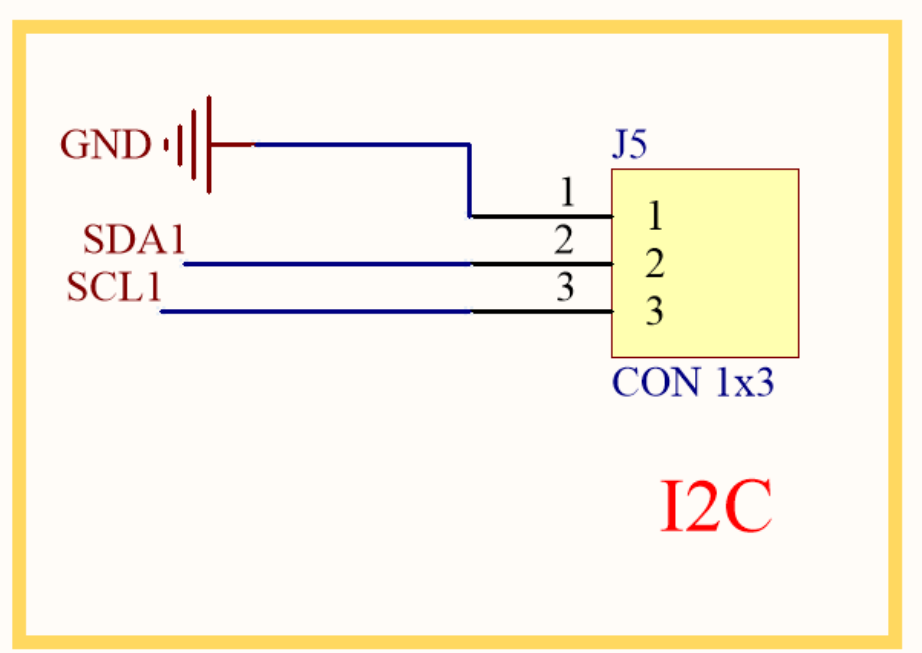

Figura 163: Conexiones para I^2C

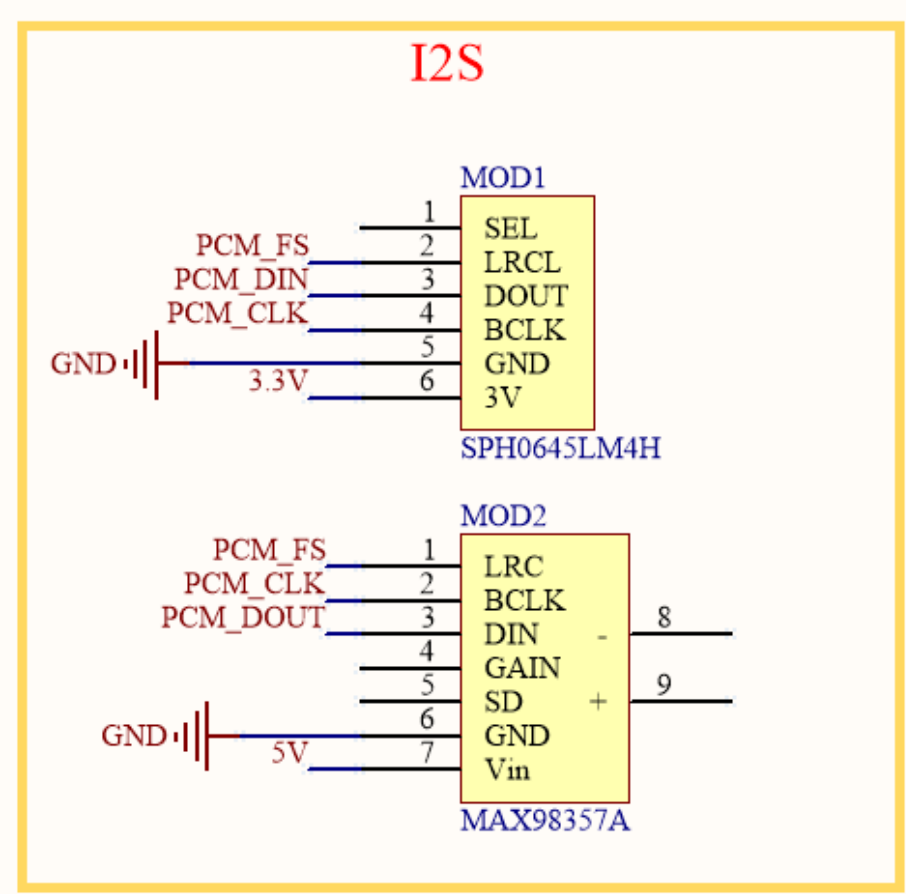

Figura 164: Conexiones para I^2S

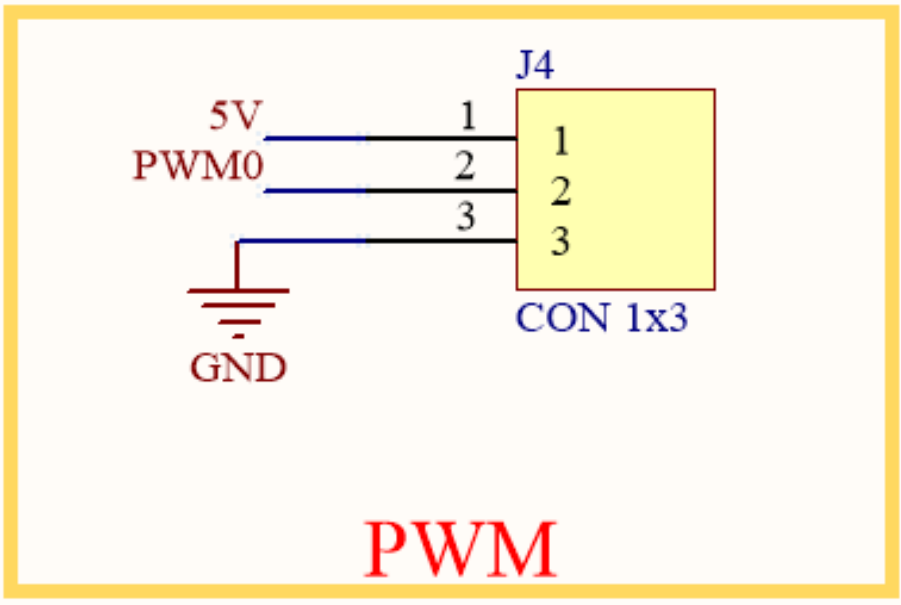

Figura 165: Conexiones para PWM

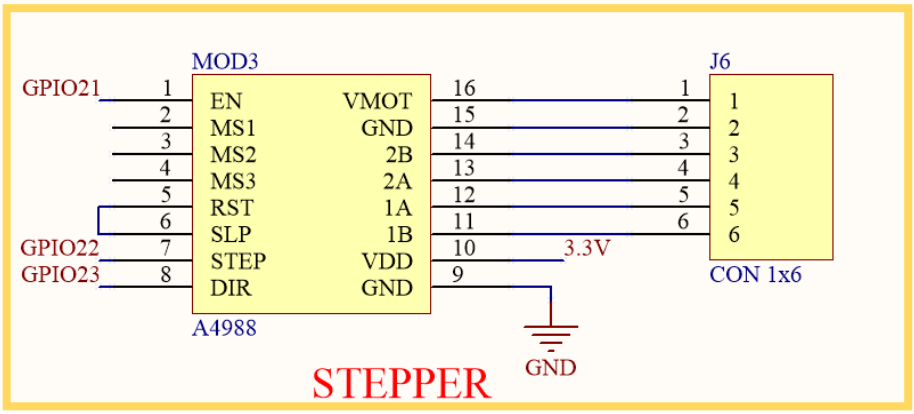

Figura 166: Conexiones para stepper

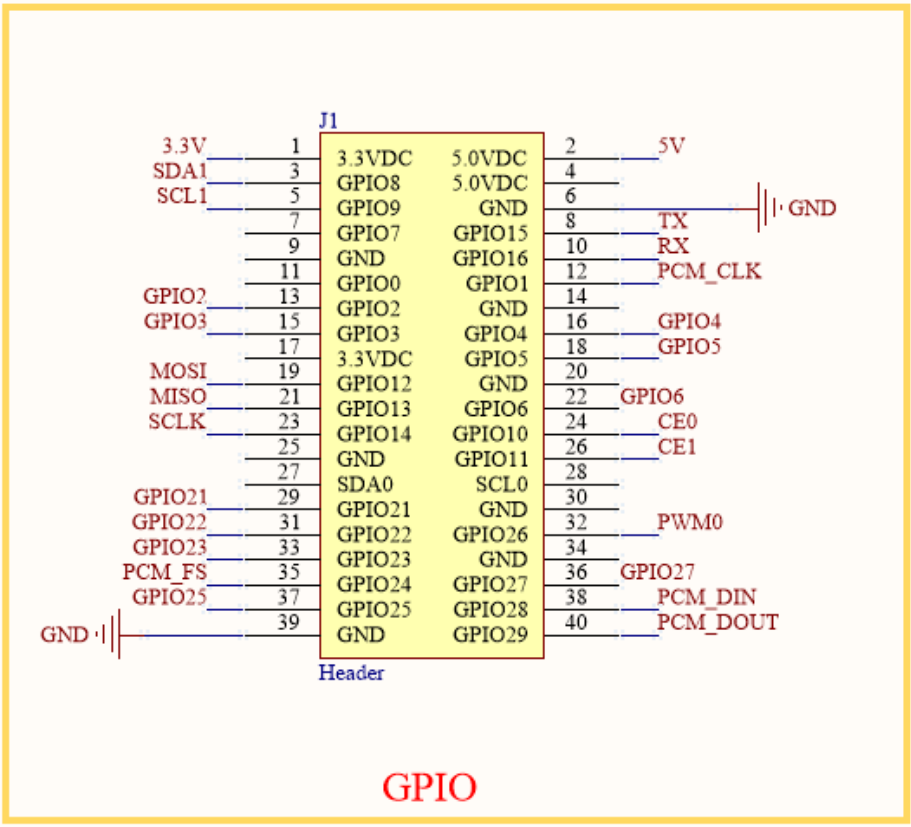

Figura 167: Conexiones del GPIO

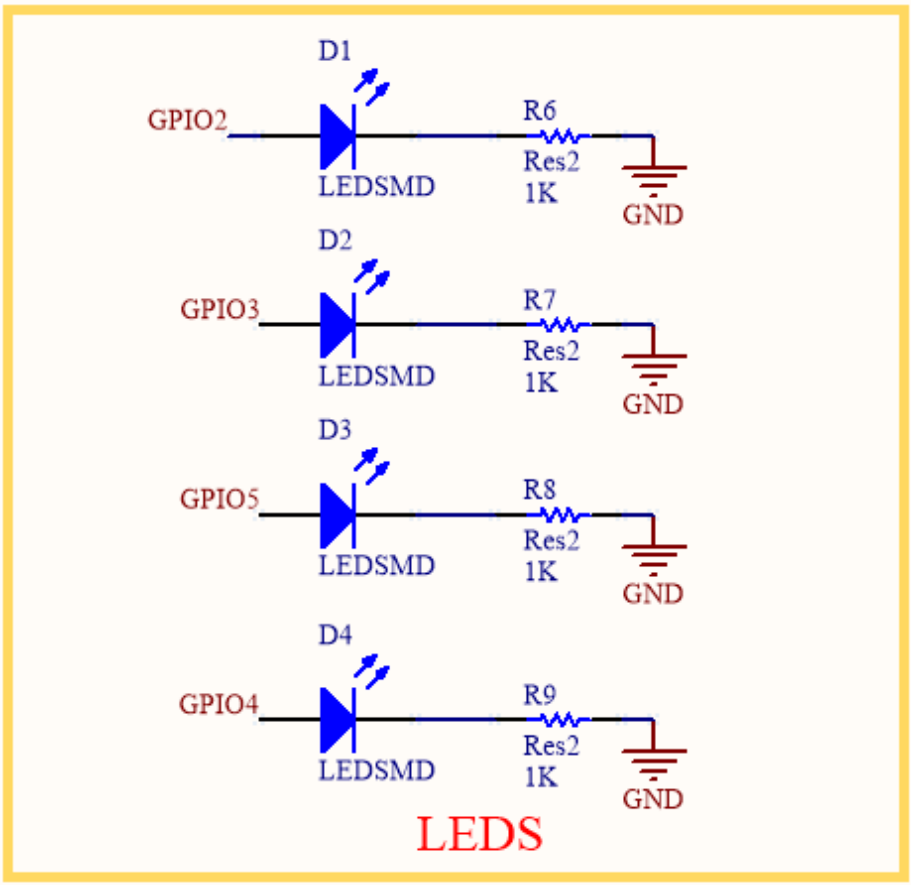

Figura 168: Conexiones para los LEDs

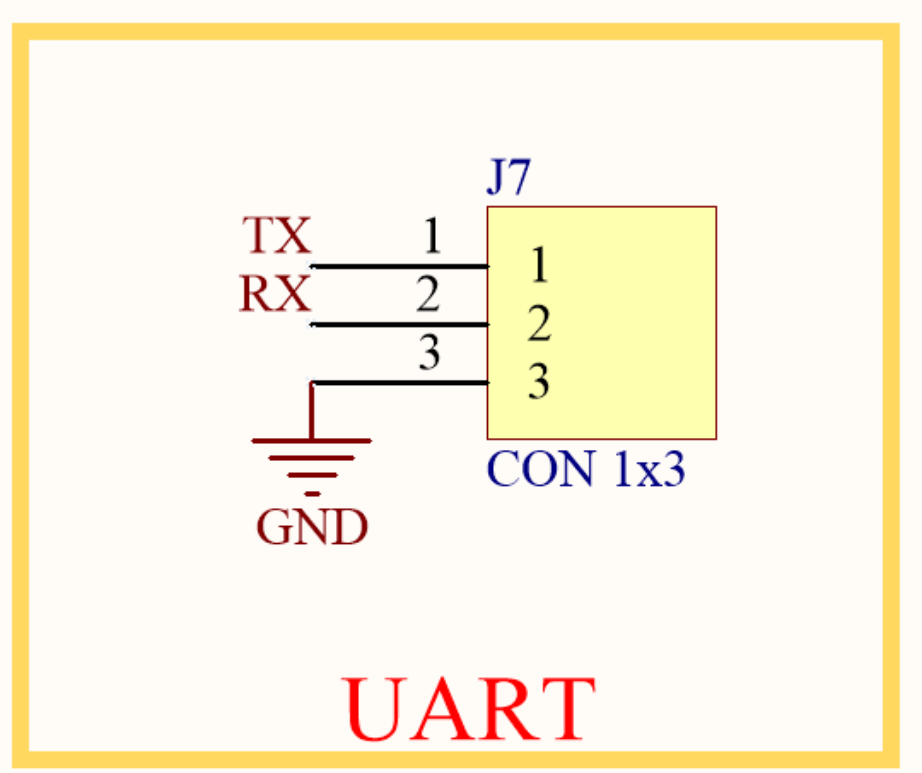

Figura 169: Conexiones para UART

14.6. Prototipo

14.6.1. *Layout*

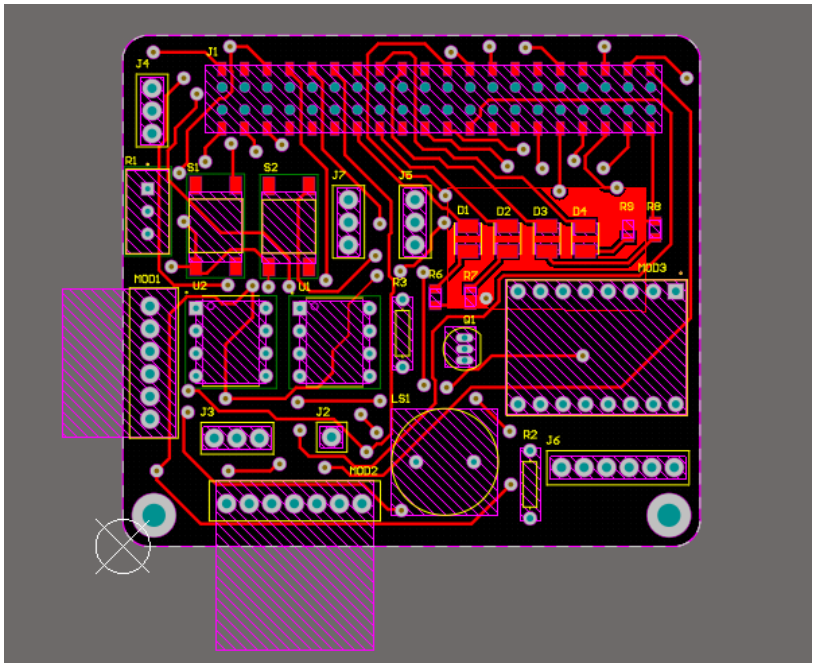

Figura 170: *Layout* de la *Top Layer* de la plataforma

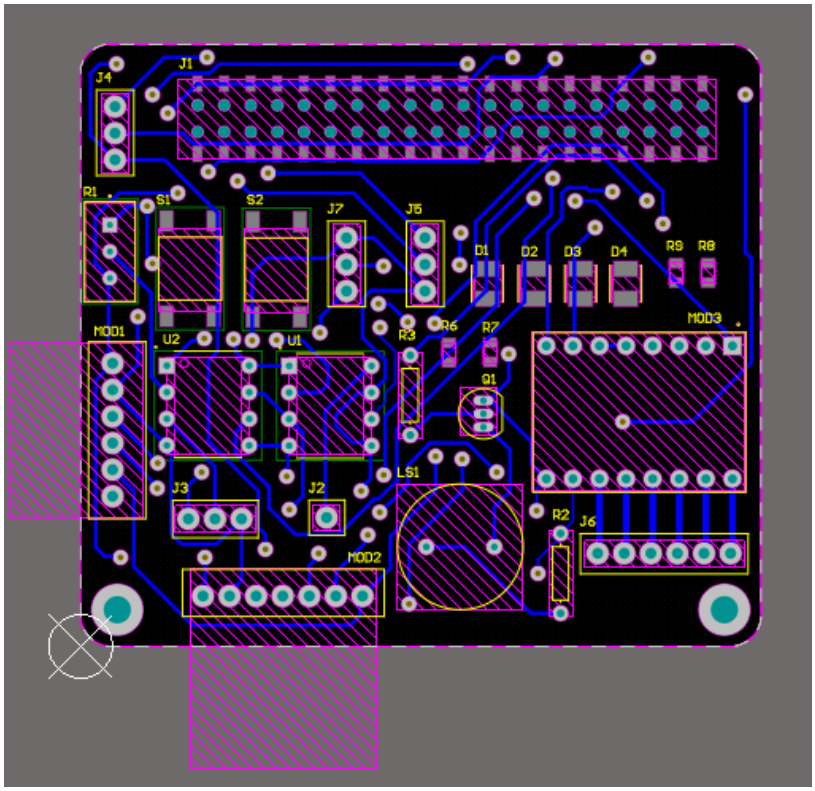

Figura 171: *Layout* de la *Bottom Layer* de la plataforma

14.6.2. Modelo 3D

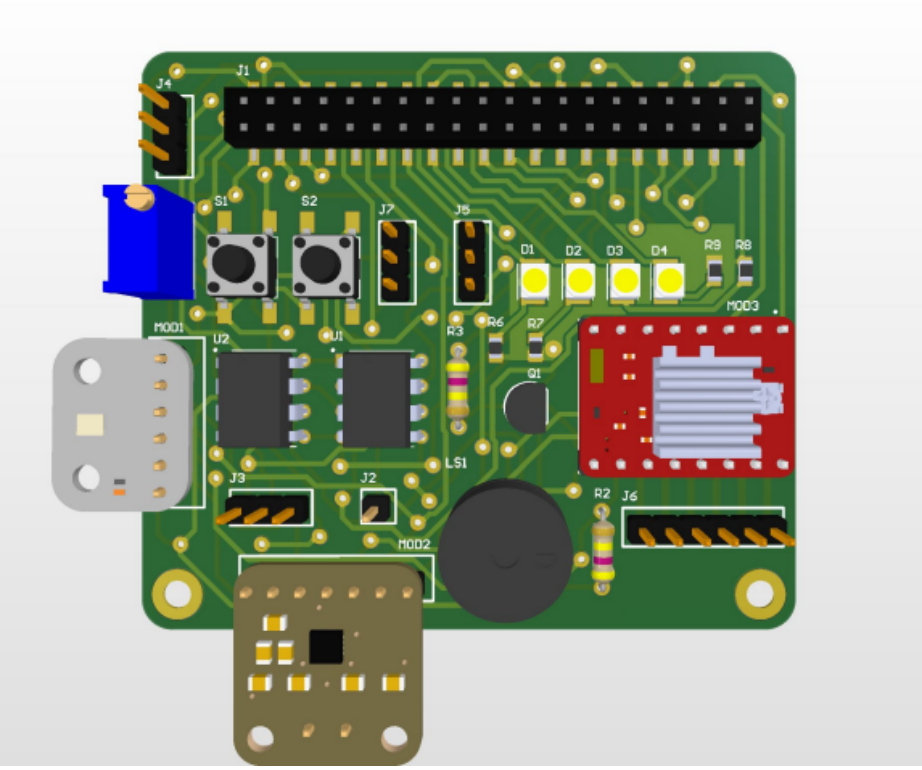

Figura 172: Modelo 3D de la plataforma

14.6.3. *Gerbers*

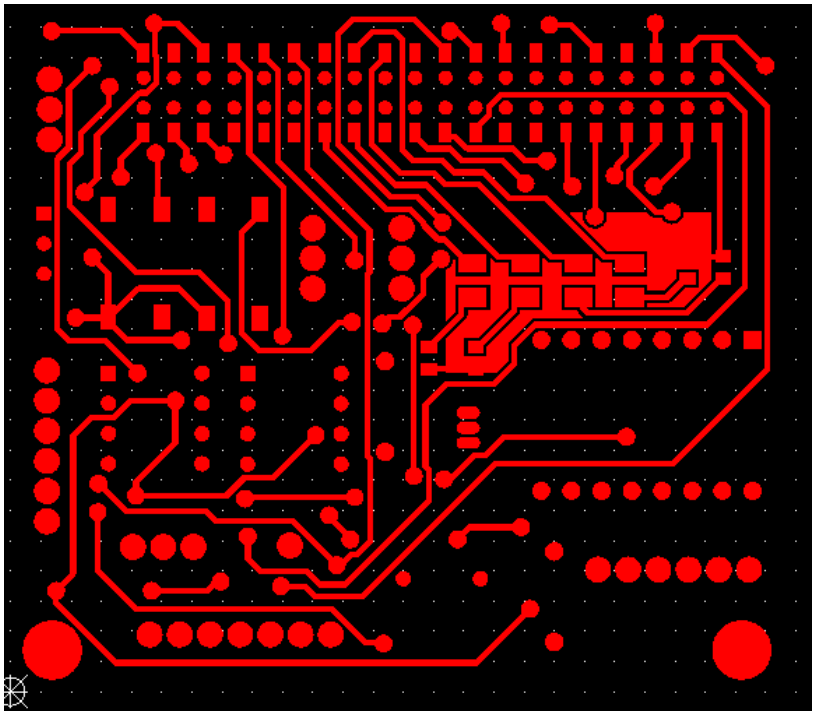

Figura 173: *Gerber* de la *Top Layer*

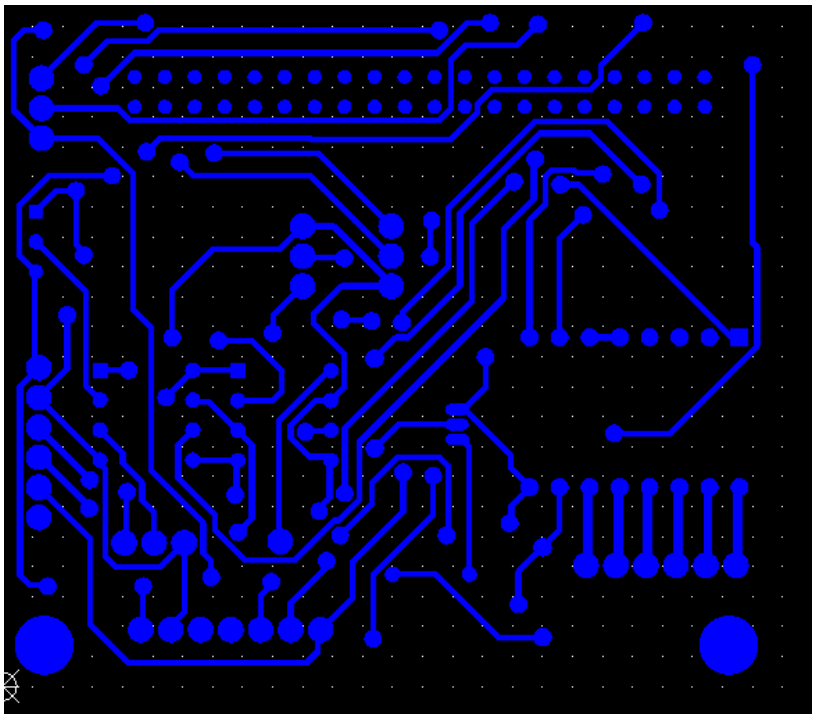

Figura 174: *Gerber* de la *Bottom Layer*

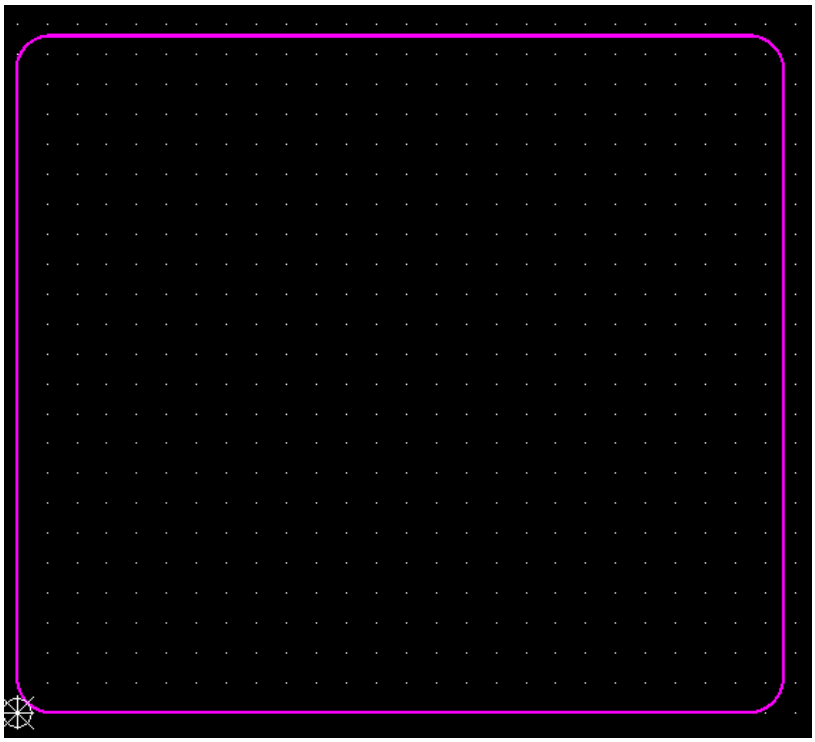

Figura 175: *Gerber* de la *Keepout Layer*

14.6.4. *NC Drill*

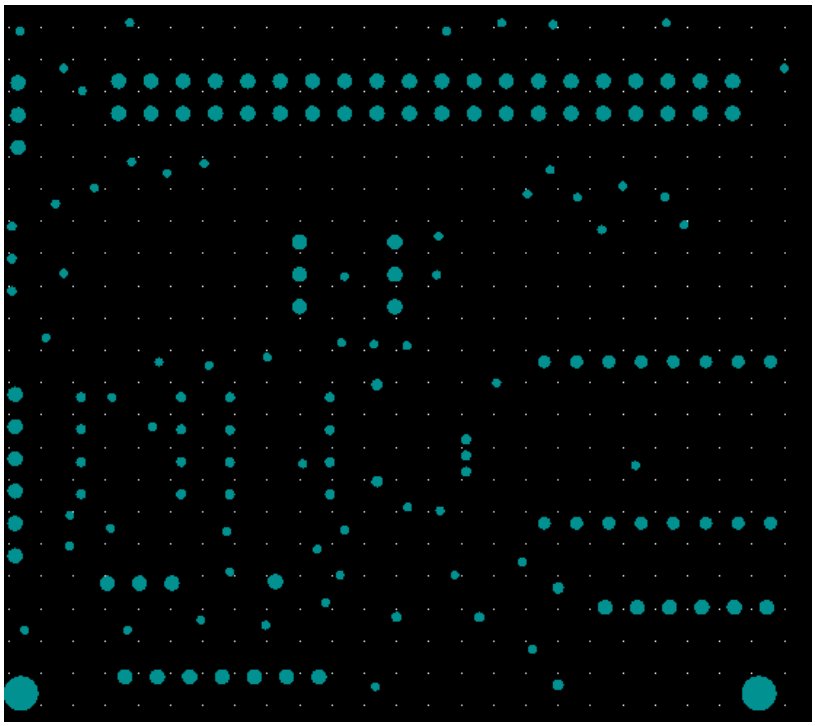

Figura 176: Archivo generado NC Drill

14.6.5. *PCB Prints*

Los PCB prints fueron generados con el fin de imprimirlos en papel acetato y cubrir la placa para aplicar luz ultravioleta a la máscara de soldadura aplicada.

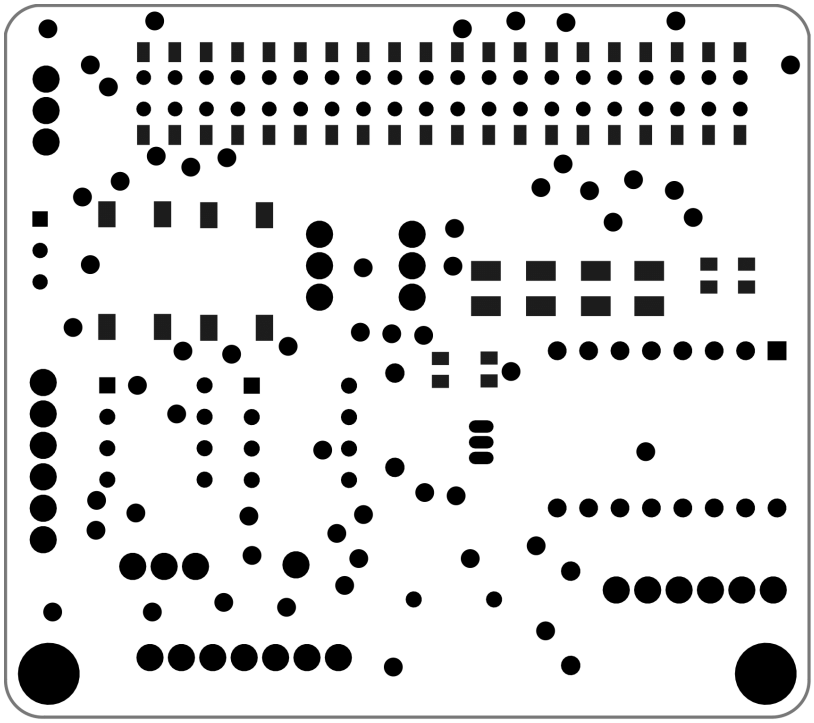

Figura 177: *Prints* de *Top Layer*

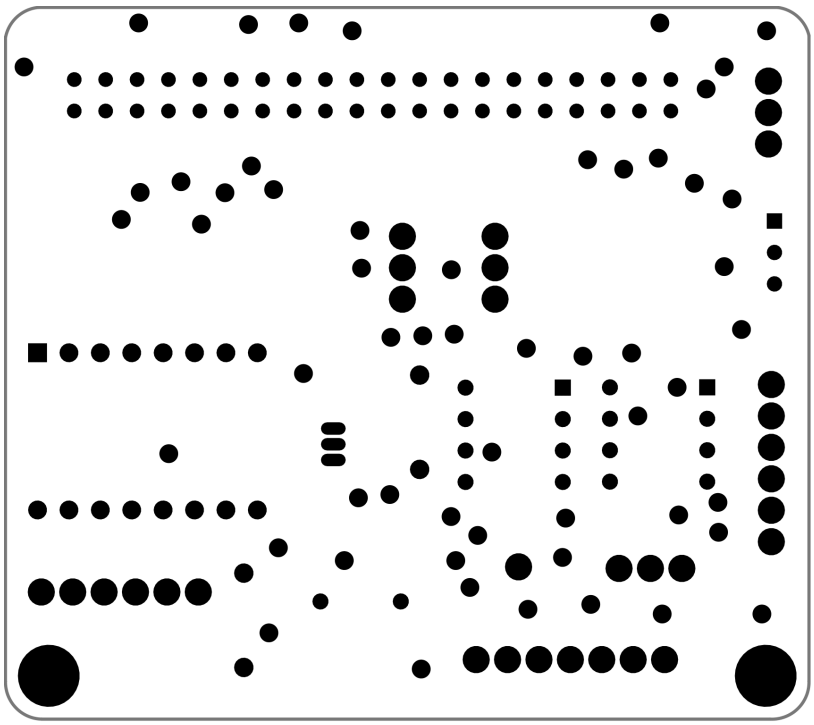

Figura 178: *Prints* de *Bottom Layer*

14.6.6. Placa física

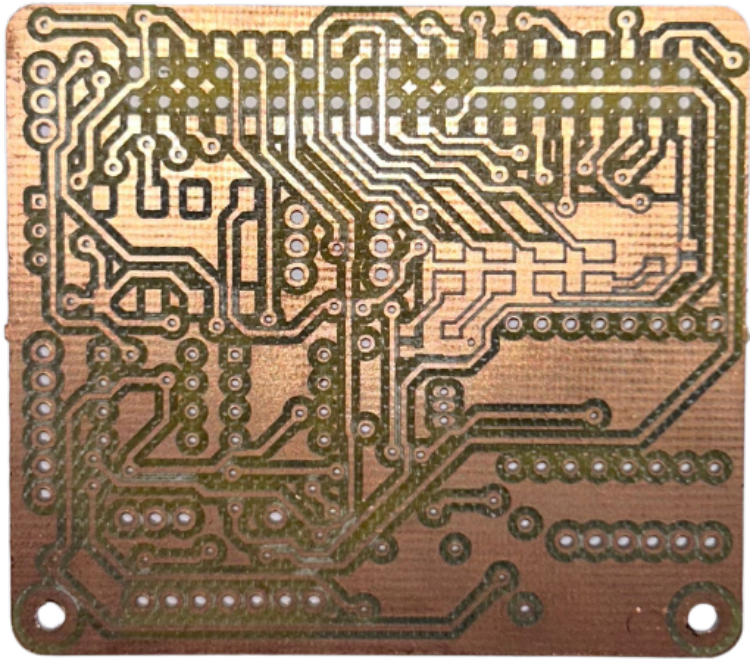

Figura 179: *Top Layer* placa física

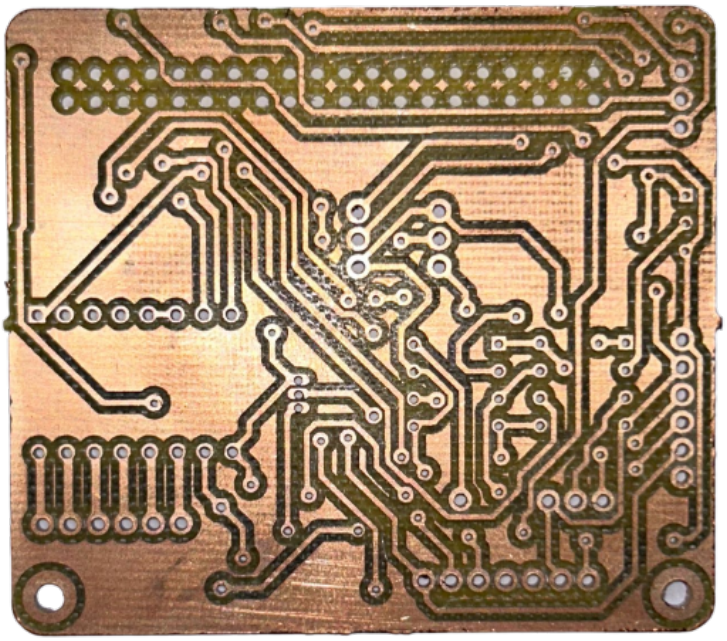

Figura 180: *Bottom Layer* placa física

14.6.7. Ensamble de la placa

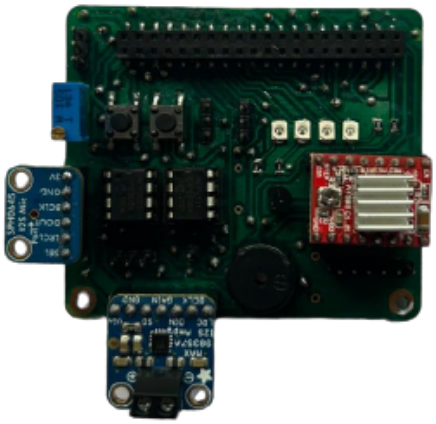

Figura 181: Vista superior de la placa física

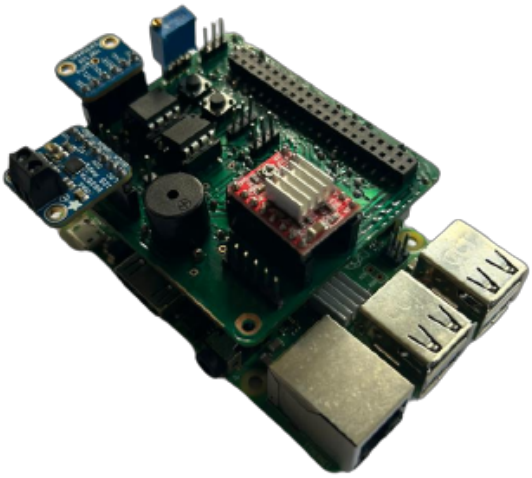

Figura 182: Vista ortogonal de placa física conectada a la *Raspberry Pi 3B+*

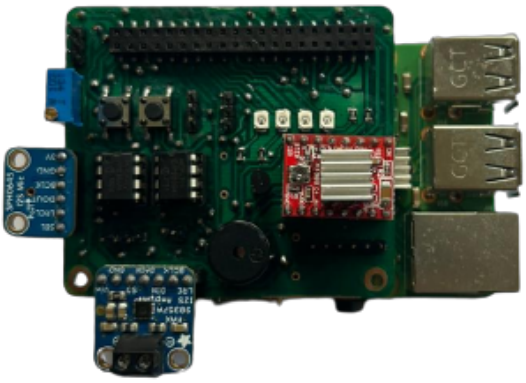

Figura 183: Vista superior de placa física conectada a la *Raspberry Pi 3B+*

14.7. Experimentos con el prototipo

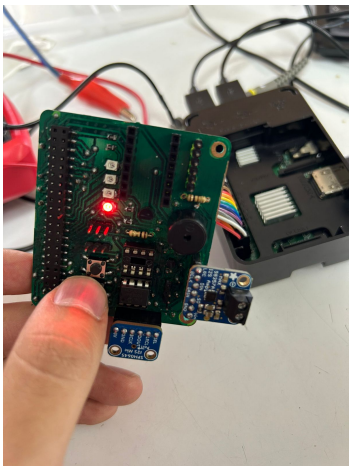

Figura 184: Contador de 4 bits utilizando LEDs y pulsadores

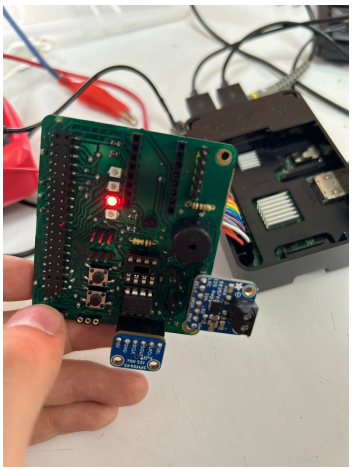

Figura 185: Contador de 4 bits utilizando LEDs y pulsadores

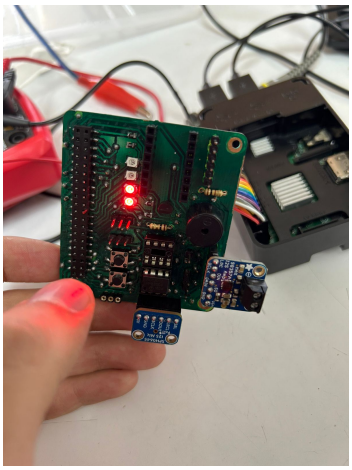

Figura 186: Contador de 4 bits utilizando LEDs y pulsadores

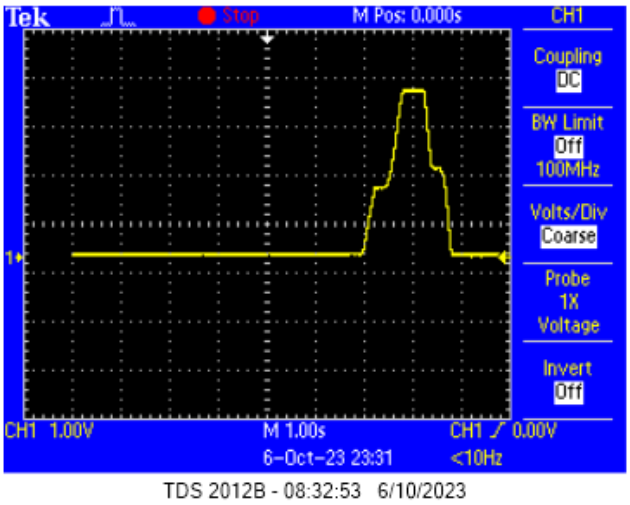

Figura 187: Vista de la señal analógica en el osciloscopio

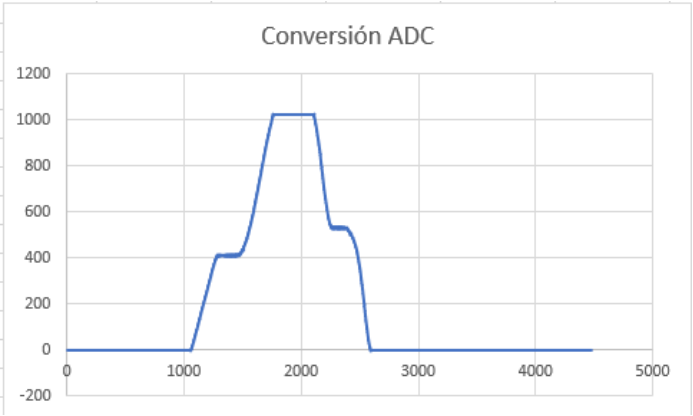

Figura 188: Gráfica de los valores de la conversión

```
pi@raspberrypi:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: sndrpii2scard [snd_rpi_i2s_card], device 0: simple-card_codec_link snd-s
oc-dummy-dai-0 [simple-card_codec_link snd-soc-dummy-dai-0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

Figura 189: Resultado del comando arecord

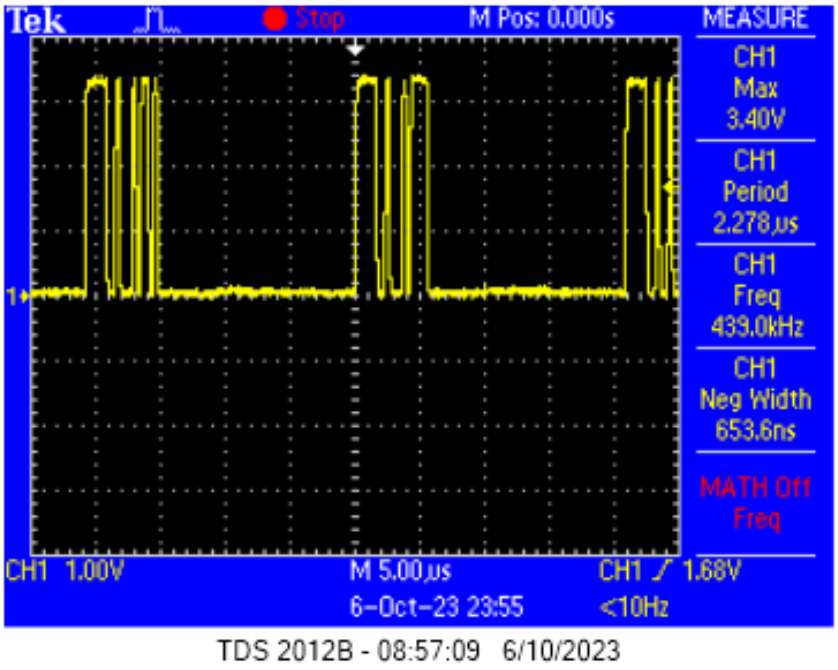

Figura 190: Salida generada por micrófono I^2S

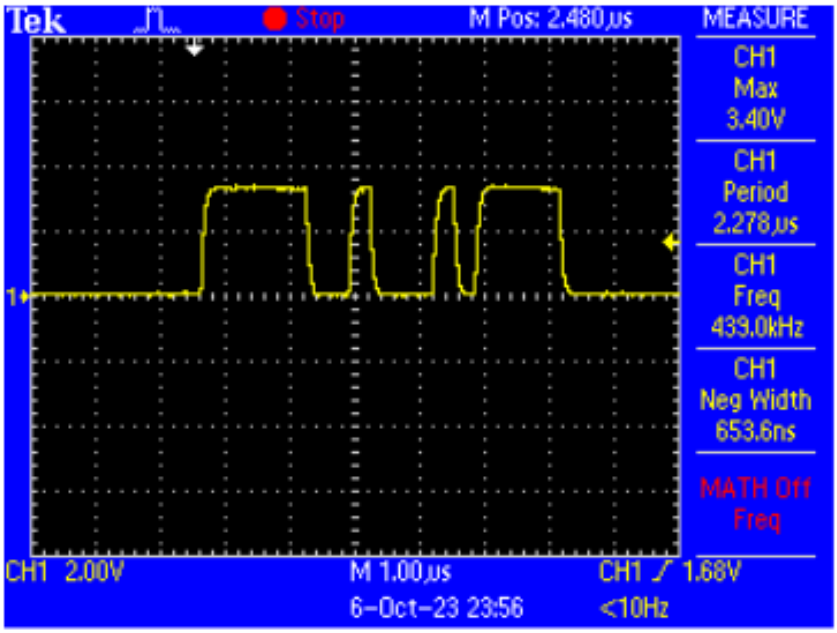

Figura 191: Segmento de la salida generada por micrófono I^2S

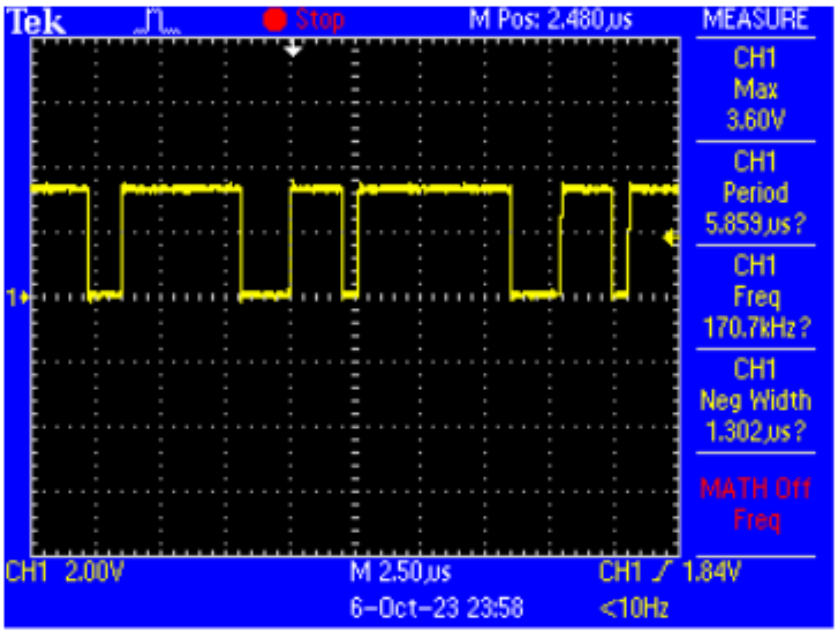

Figura 192: Señal de entrada del módulo MAX98357

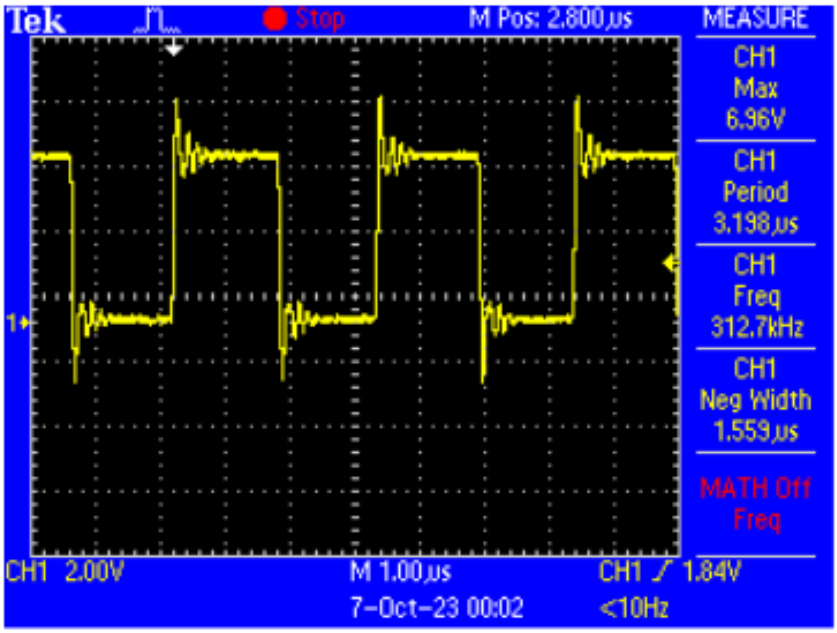

Figura 193: Señal de salida del módulo MAX98357

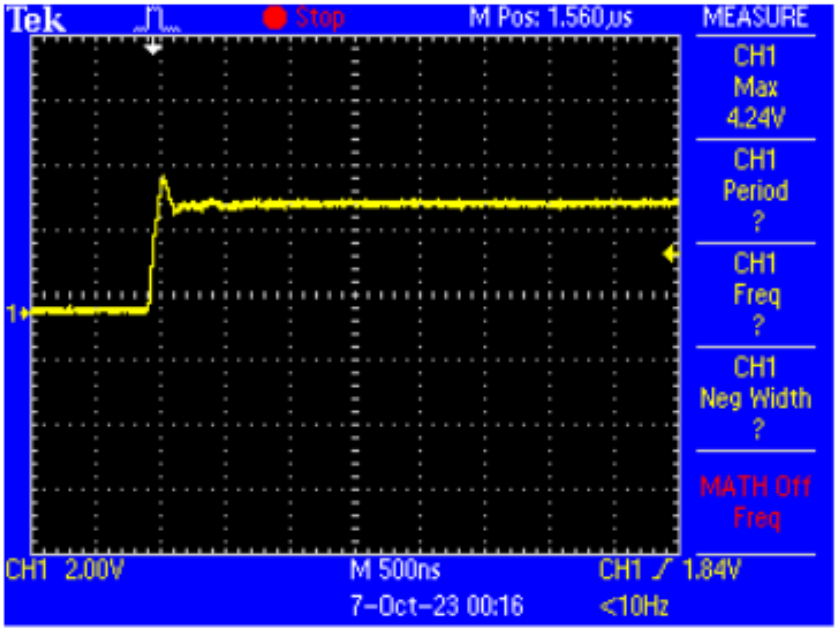

Figura 194: Pulso generado para el funcionamiento del módulo A4988

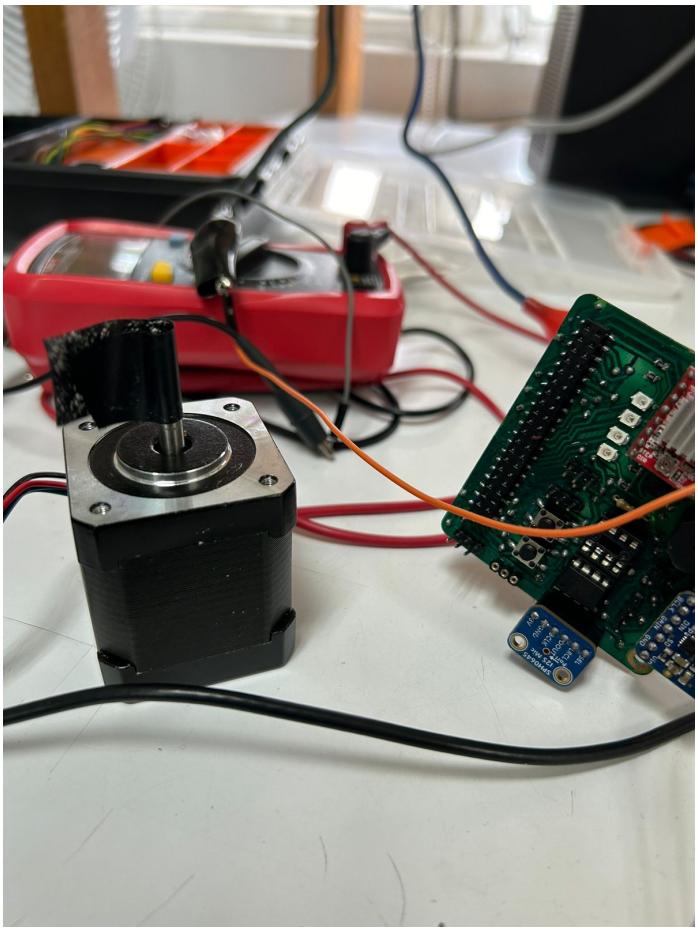

Figura 195: Motor *Stepper*

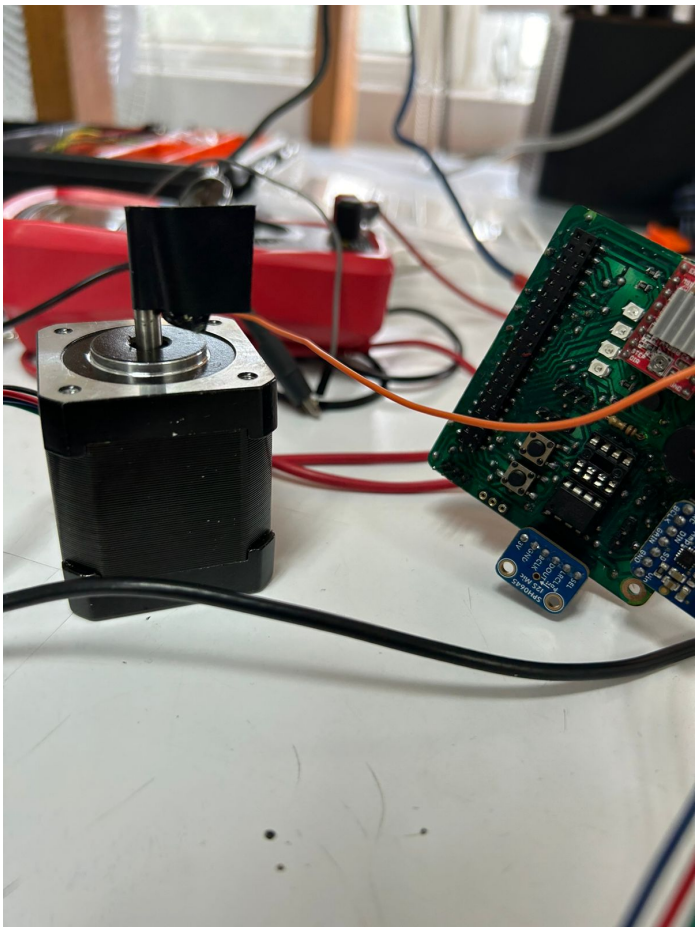

Figura 196: Motor *Stepper*

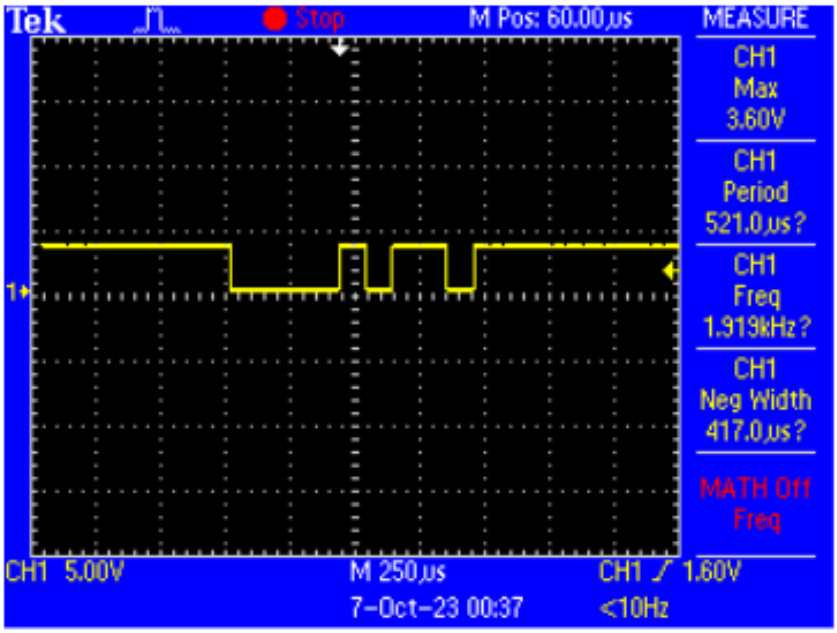

Figura 197: Dato enviado por UART