Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

DISEÑO Y DESARROLLO DE UN SISTEMA DE ALERTA TEMPRANA SOBRE RIESGOS AMBIENTALES EN COMBATE DE INCENDIOS ESTRUCTURALES

Jorge Augusto Balsells Orellana
Asesorado por el Ing. Byron Odilio Arrivillaga Méndez

Guatemala, abril de 2018

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO Y DESARROLLO DE UN SISTEMA DE ALERTA TEMPRANA SOBRE RIESGOS AMBIENTALES EN COMBATE DE INCENDIOS ESTRUCTURALES

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

JORGE AUGUSTO BALSELLS ORELLANA
ASESORADO POR EL ING. BYRON ODILIO ARRIVILLAGA MÉNDEZ

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN ELECTRÓNICA

GUATEMALA, ABRIL DE 2018

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Pedro Antonio Aguilar Polanco
VOCAL I	Ing. Angel Roberto Sic García
VOCAL II	Ing. Pablo Christian de León Rodríguez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Oscar Humberto Galicia Nuñez
VOCAL V	Br. Carlos Enrique Gómez Donis
SECRETARIA	Inga. Lesbia Magalí Herrera López

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

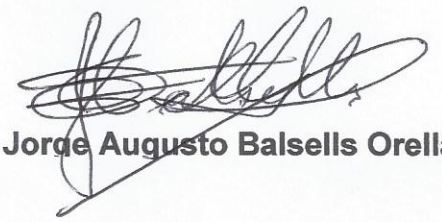
DECANO	Ing. Pedro Antonio Aguilar Polanco
EXAMINADORA	Inga. Ingrid Salomé Rodríguez de Loukota
EXAMINADOR	Ing. José Aníbal Silva de los Ángeles
EXAMINADOR	Ing. Byron Odilio Arrivillaga Méndez
SECRETARIA	Inga. Lesbia Magalí Herrera López

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO Y DESARROLLO DE UN SISTEMA DE ALERTA TEMPRANA SOBRE RIESGOS AMBIENTALES EN COMBATE DE INCENDIOS ESTRUCTURALES

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha 6 de octubre de 2016.



Jorge Augusto Balsells Orellana

Nueva Guatemala de la Asunción, Jueves 11 de Enero del 2018

Ingeniero.
Julio Cesar Solares Peñate.
Coordinador del área de electrónica.
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

Apreciable Ingeniero Solares:

Me permito dar aprobación al trabajo de graduación titulado **"Diseño y desarrollo de un sistema de alerta temprana sobre riesgos ambientales en combate de incendios estructurales"** del señor Jorge Augusto Balsells Orellana, cuyo carnet estudiantil es **"200819335"** por considerar que cumple con todos los requisitos establecidos.

Por tanto, el autor de este trabajo de graduación y yo, como su asesor, nos hacemos responsables por el contenido, conclusiones y desarrollo del mismo.

Sin otro particular, me es grato saludarle.

Atentamente:

Byron Arrivillaga Méndez
Ingeniero Electrónico
Colegiado 5217

f. _____
Ing. Byron Odilio Arrivillaga Mendez.
Colegiado 5,217.
Asesor.

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Guatemala, 25 de enero de 2018

Señor Director
Ing. Otto Fernando Andrino González
Escuela de Ingeniería Mecánica Eléctrica
Facultad de Ingeniería, USAC.

Señor Director:

Por este medio me permito dar aprobación al Trabajo de Graduación titulado: **DISEÑO Y DESARROLLO DE UN SISTEMA DE ALERTA TEMPRANA SOBRE RIESGOS AMBIENTALES EN COMBATE DE INCENDIOS ESTRUCTURALES**, desarrollado por el estudiante **Jorge Augusto Balsells Orellana**, ya que considero que cumple con los requisitos establecidos.

Sin otro particular, aprovecho la oportunidad para saludarlo.

Atentamente,

ID Y ENSEÑAD A TODOS

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

Ing. Julio César Solares Peñate
Coordinador de Electrónica

NIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

REF. EIME 08. 2018.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto bueno del Coordinador de Área, al trabajo de Graduación del estudiante: JORGE AUGUSTO BALSELLS ORELLANA Titulado: DISEÑO Y DESARROLLO DE UN SISTEMA DE ALERTA TEMPRANA SOBRE RIESGOS AMBIENTALES EN COMBATE DE INCENDIOS ESTRUCTURALES procede a la autorización del mismo.

Ing. Otto Fernando Andrino González

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

GUATEMALA, 1 DE MARZO 2018.

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 131 .2018

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **DISEÑO Y DESARROLLO DE UN SISTEMA DE ALERTA TEMPRANA SOBRE RIESGOS AMBIENTALES EN COMBATE DE INCENDIOS ESTRUCTURALES,** presentado por el estudiante universitario: **Jorge Augusto Balsells Orellana,** y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Pedro Antonio Aguilar Polanco
Decano

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERIA

Guatemala, abril de 2018

/gdech

ACTO QUE DEDICO A:

Dios	A pesar de mi falta de compromiso hacia Él y cantidad de preguntas de su entorno, no me ha abandonado en ningún trayecto de mi vida y me ha brindado las respuestas necesarias en el momento preciso.
Mi familia	Alba Nohemí Orellana Cordón de Balsells, Jorge Augusto Balsells Balcárcel, Melba Nohemí Balsells Orellana, Irma Esperanza Balsells Orellana, Ana Silvia Balsells Orellana y Mariela Escobar Paiz, por ser las personas en las que más confío.
Mi mentor	Ing. Byron Odilio Arrivillaga Méndez, por su completo apoyo en la asesoría de este texto, en el trayecto laboral y en el trayecto universitario.
Mis amigos de infancia	Leonel Alberto Vásquez García, Diego Armando Fernández Briones y Pedro Estuardo Paniagua Gómez, porque no es necesario estar siempre cerca para saber que se cuenta con su hermandad.

AGRADECIMIENTOS A:

Mi familia	Jorge Augusto Balsells Balcárcel, Alba Nohemí Orellana Cordón de Balsells, Melba Nohemí, Irma Esperanza y Ana Silvia Balsells Orellana y Mariela Escobar Paiz, por su cariño y apoyo incondicional.
Mi mentor	Ing. Byron Odilio Arrivillaga Méndez, por su apoyo incondicional en la facultad de ingeniería.
Mis amigos de la la carrera	Por haber compartido buenos y malos momentos inolvidables. Especialmente a José Rodrigo de León Velásquez, Haroldo López de los Ríos, Freddy Fernando Chang Chau, Kenie Gary Chuy Azurdia.
Universidad de San Carlos de Guatemala	Alma mater, casa de estudios que me ayudo y me ayudara a desarrollarme en varios ámbitos de mi vida, por ser la fuente de conocimientos fundamentales y enseñanzas de vida.
Bomberos Voluntarios de Guatemala	Por ser una institución de servicio voluntario que me generó la idea a desarrollar sobre este texto.

Fuentes de información abiertas	Porque sin estas fuentes, la transferencia de conocimiento requeriría de un tiempo mucho mayor al actual en un grupo delimitado de personas.
Agradecimientos generales	Oficial de bomberos Carlos Guillermo de León.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES VII
LISTA DE SÍMBOLOS XIII
GLOSARIO XV
RESUMEN XXIII
OBJETIVOS XXV
INTRODUCCIÓN XXVII

1. FUNDAMENTOS TEÓRICOS Y NECESIDADES DE DESARROLLO 1
1.1. Fundamentos teóricos sobre combustión 1
1.2. Elementos para que exista un proceso de combustión 2
1.2.1. ¿Qué es una reacción química? 3
1.2.1.1. Reacción química en cadena 3
1.2.1.2. ¿Qué es pirólisis? 4
1.2.1.3. Comportamiento del combustible según su estado físico inicial 4
1.2.1.4. Energía de activación 5
1.2.2. Tipos de combustión 7
1.2.3. Reacciones de combustión 10
1.2.4. Gases y fluidos 10
1.2.4.1. Constante universal y constante particular de los gases 11
1.2.4.1.1. Mezclas de gases 12
1.2.4.1.2. Propiedades del aire y los gases combustibles 13

1.2.5. Etapas de un incendio 25
1.2.6. Comportamiento del fuego en espacios cerrados ... 30
1.2.7. Fenómenos físico-químicos en incendios 31
1.2.8. Propiedades del agua en incendios 34
1.3. Necesidades y restricciones para el desarrollo del sistema 37
1.3.1. Necesidades del sistema 37
1.3.2. Restricciones del sistema 38
1.3.3. Desarrollo del sistema 40

2. HARDWARE 43
2.1. Dispositivos de hardware utilizados 43
2.1.1. Microprocesador 43
2.1.2. Microcontrolador 45
2.1.2.1. ATMEL AVR MEGA 32U4 45
2.1.3. Sensores 50
2.1.3.1. Sensor de humedad y temperatura DHT22 50
2.1.3.2. Sensor de temperatura DS18B20 52
2.1.3.3. Sensor de temperatura MLX90614 54
2.1.3.4. Sensores de gases MQ 56
2.1.3.4.1. Sensor de gas MQ-2 57
2.1.3.5. Sensor de movimiento y aceleración MPU6050 61
2.1.4. Dispositivos de alerta y visualización 62
2.1.4.1. Dispositivos visuales 62
2.1.4.1.1. Pantalla OLED SH1106 63
2.1.4.1.2. Pantalla OLED SSD1306 64

2.1.4.1.3. Actuadores luminosos .. 65
2.1.4.2. Dispositivos táctiles 66
2.1.4.2.1. Minimotor vibrador 66
2.1.4.2.2. Pulsadores DPST 67
2.1.4.3. Dispositivos audibles 68
2.1.4.3.1. Buzzer 68
2.1.4.3.2. Sirena 68
2.1.4.3.3. LC9801 – LC9806 69
2.1.5. Dispositivos de comunicación 70
2.1.5.1. Módulos bluetooth BC417 (HC-05 – HC-06) 70
2.1.5.2. Módulo wifi IEEE 802,11N 71
2.1.5.3. Módulo USB a UART CP2102 72
2.1.6. Alimentación, carga y protección 74
2.1.6.1. Baterías LiPo 74
2.1.6.1.1. Batería LiPo 903052 75
2.1.6.1.2. Batería LiPo 702035 76
2.1.6.2. Cargador Lineal Li-Ion LTC4056 77
2.2. Análisis de circuitos y conexiones 78
2.2.1. Ingeniería inversa de placas 78
2.2.1.1. Arduino pro micro 78
2.2.1.2. FC-75 81
2.2.1.3. GY-521 83
2.2.1.4. ZS-040 84
2.2.1.5. Placas MQ 85
2.2.2. Diagramas de circuitos y conexiones 87
2.2.2.1. Careta 87
2.2.2.2. *Pass* 93
2.2.3. Diseño de placa de circuito impreso (PCB) 101

2.3. Estructura 106
2.3.1. Materiales utilizados 107
2.3.2. Diseño y desarrollo de la estructura 108

3. SOFTWARE Y FIRMWARE 141
3.1. Conceptos básicos de programación 141
3.2. Paradigmas de programación 144
3.2.1. Paradigma imperativo 145
3.2.2. Paradigma orientado a objetos 146
3.2.3. Paradigma declarativo 146
3.2.4. Paradigma funcional 147
3.2.5. Paradigma lógico 147
3.3. Librerías 147
3.3.1. Librerías de *Firmware* utilizadas 148
3.3.2. Librerías de Software utilizadas 150
3.4. Firmware 153
3.4.1. Protocolos de comunicación 154
3.4.1.1. Protocolo UART/USART 154
3.4.1.2. Protocolo SPI 158
3.4.1.3. Protocolo I^2C 161
3.4.1.4. Protocolo *One Wire* 164
3.4.2. Filtros digitales 166
3.4.2.1. Filtro complementario 167
3.4.2.2. Filtro de media móvil 168
3.4.2.3. Gráficos de filtros 170
3.4.3. Configuración de dispositivos bluetooth 178
3.5. Software 184
3.5.1. Configuraciones necesarias en el sistema Raspbian-Jessie 185

3.5.1.1. Configurar terminal serial de Raspberry Pi Zero en Raspbian-Jessie 186
3.5.1.2. Activar protocolos de comunicación en Raspberry Pi Zero..... 189
3.5.1.3. Configuración de wifi desde una terminal Raspbian-Jessie en Raspberry Pi Zero..... 191
3.5.1.4. Configuración de pines GPIO 192
3.5.2. Base de datos..... 193
3.5.2.1. Conceptos básicos 194
3.5.2.2. Diseño 196
3.6. Algoritmos..... 205
3.7. Diagramas de flujo..... 206
3.7.1. Diagramas de flujo de careta..... 207
3.7.2. Diagramas de flujo de *pass* 215

4. PRESUPUESTO 237

CONCLUSIONES 239
RECOMENDACIONES..... 241
BIBLIOGRAFÍA..... 243
APÉNDICE..... 249

ÍNDICE DE ILUSTRACIONES

FIGURAS

1. Límites del propano 17
2. Rango de inflamabilidad 18
3. Saturación de combustible con oxígeno 19
4. Variación de la densidad del aire respecto a la temperatura 28
5. Fase latente o decrecimiento 29
6. Comportamiento del fuego en espacios cerrados 31
7. Microprocesador 44
8. Raspberry Pi B+ GPIO 44
9. Configuración de pines Microcontrolador MEGA32U4 48
10. Diagrama de bloques Microcontrolador ATMEGA 32U4 49
11. Tarjeta de desarrollo Arduino Pro Micro 50
12. Sensor DHT22 51
13. Empaquetados sensor DS18B20 52
14. Diagrama de bloques interno sensor DS18B20 53
15. Conexión con SMBus sensor MLX90614 55
16. Sensor MLX 90614 55
17. Circuito de sensores de gases 56
18. Estructura sensor MQ-2 58
19. Características e influencias del sensor MQ-2 58
20. Estructura sensor MQ-135 59
21. Dependencia del sensor MQ-135 respecto a la humedad y temperatura 60
22. Características sensitivas del sensor MQ-135 60

23. Sensor MPU-6050 62
24. Pantalla OLED 65
25. Minimotor vibrador 67
26. Modulo bluetooth 71
27. Antena y capacitores para Wireless dentro de Raspberry Pi Zero W ... 72
28. Diagrama de bloques CP2102 73
29. Módulo CP2102 73
30. Batería LiPo 903052 76
31. Batería LiPo 702035 76
32. Circuito de carga de baterías y gráfica de Voltaje de entrada Vs Corriente del mismo circuito 78
33. Regulador de voltaje 79
34. Oscilador 80
35. Arduino Pro Micro 32U4 81
36. Cargador de baterias LiPo 82
37. Acelerometro y giroscopio MPU6050 84
38. Bluetooth HC-05 85
39. Sensores MQ Series 86
40. Circuito de careta 88
41. Regulador de voltaje careta 89
42. Conversor USB a UART CP2 90
43. Módulo MPU6050 91
44. Módulo bluetooth HC-05 92
45. Pantalla OLED SH1106 93
46. Pass completo 95
47. Regulador de voltaje y cargador de *pass* 96
48. Sirena *pass* 96
49. Buzzer 97
50. Módulo de pantalla OLED *pass* 98

51. Módulo HC-05 *pass* 99
52. Sensores *pass* 100
53. PCB completa 102
54. PCB sin componentes 103
55. PCB parte superior 104
56. PCB parte inferior 104
57. PCB con capa de soldaduras y componentes 105
58. PCB unicamente con capa de soldaduras 106
59. Pieza 1, careta frontal 109
60. Pieza 1, careta dorsal 110
61. Pieza 1, careta laterales 111
62. Pieza 1, careta superior y posterior 112
63. Pieza 1, careta vista angular superior 112
64. Pieza 1, careta vista dorsal superior 113
65. Pieza 2, careta frontal 114
66. Pieza 2, careta dorsal 115
67. Pieza 2, careta laterales 116
68. Pieza 2, careta superior y posterior 117
69. Pieza 2, careta inferior izquierda 117
70. Pieza 2, careta dorsal derecha 118
71. Pieza 3, careta frontal 119
72. Pieza 3, careta angular 119
73. Piezas careta ensambladas vista angular superior 120
74. Piezas careta ensambladas vista dorsal inferior 121
75. Pieza 1, *pass* frontal 122
76. Pieza 1, *pass* dorsal 123
77. Pieza 1, *pass* lateral derecha 124
78. Pieza 1, *pass* lateral izquierda 125
79. Pieza 1, *pass* superior 126

80. Pieza 1, *pass* posterior 126
81. Pieza 1, *pass* angular superior derecha 127
82. Pieza 1, *pass* angular dorsal poster 128
83. Pieza 2, *pass* frontal 129
84. Pieza 2, *pass* dorsal 129
85. Pieza 2, *pass* laterales 130
86. Pieza 2, *pass* superior 130
87. Pieza 2, *pass* inferior 131
88. Pieza 2, *pass* angular superior izquierda 132
89. Pieza 2, *pass* angular dorsal inferior 133
90. Pieza 3, *pass* frontal 134
91. Pieza 3, *pass* dorsal 134
92. Pieza 3, *pass* laterales 135
93. Pieza 3, *pass* Superior y posterior 136
94. Pieza 3, *pass* angular superior derecha 136
95. Pieza 3, *pass* angular dorsal inferior 137
96. *Pass* ensamblado vista angular frontal 137
97. *Pass* ensamblado vista frontal inferior 138
98. *Pass* ensamblado vista frontal y lateral 139
99. Paradigmas de programación 145
100. Diagrama de bloques del protocolo UART 155
101. Esquema de la transmisión y recepción asíncrona de un protocolo UART 156
102. Sincronización de distintos modos de SPI 159
103. Diagramas de conexiones correspondientes al protocolo SPI 161
104. Trama de datos en un bus I2C 164
105. Transmisión y recepción de datos del protocolo One Wire 166
106. Datos recibidos directamente del acelerómetro 173
107. Datos recibidos directamente del Giroscopio 174

108. Datos normalizados aplicando un filtro complementario 175
109. Comparación de filtrado de datos 176
110. Comparación de filtros 177
111. Datos de temperatura MPU6050 178
112. Configuración de puerto 179
113. Comandos AT, 1 181
114. Comandos AT, 2 183
115. Debian en Raspberry Pi 186
116. Configuración Raspi-config 187
117. Configuración terminal puerto serial Raspberry Pi Zero 188
118. Puerto Com y Bauds 188
119. Protocolos comunicación Raspberry Pi Zero 190
120. Raspberry pi wireless config 191
121. Configuración de redes wireless 192
122. Base de datos completa 196
123. Usuarios 198
124. Alarma 199
125. Incidente y tipos de incidente 200
126. Acc Gyr y logs Acc Gyr 201
127. Temperatura remota y logs de temperatura remota 202
128. Logs 203
129. Humedad relativa y logs de humedad relativa 204
130. Vapores, gases y logs de vapores y gases 205
131. Diagrama de flujo careta 207
132. Inicio diagrama de flujo careta 209
133. Fin diagrama careta 209
134. Carga batería diagrama careta 210
135. Diagrama de lecturas careta 211
136. Diagrama de transmisiones de datos careta 212

137. Diagrama de perdida de conexión careta 213
138. Diagrama de vistas en pantalla careta 215
139. Diagrama de microcontrolador *pass* 217
140. Diagrama de Raspberry *pass* 218
141. Diagrama principal *pass* 220
142. Diagrama ciclo principal *pass* 221
143. Diagrama sirena *pass* 223
144. Diagrama botón rojo *pass* 224
145. Diagrama botón verde *pass* 225
146. Diagrama transmisiones de datos *pass* 226
147. Diagrama de datos *pass* 227
148. Diagrama de cálculos *pass* 228
149. Diagrama de vibrador y alerta *pass* 229
150. Diagrama de lectura de datos y perdida de conexión *pass* 230
151. Diagrama de filtros *pass* 232
152. Diagrama de cálculo de promedio *pass* 233
153. Diagrama de cálculos *pass* 234
154. Diagrama de alerta *pass* 235

TABLAS

I. Densidad de los gases más comunes en incendios 15
II. Cantidad de gases que tienden a quemarse más rápido 20
III. Rangos de escala y el valor máximo raw 61
IV. Relación entre frecuencia de reloj, tasa de baudios y probabilidad de error 157
V. Presupuesto 238

LISTA DE SÍMBOLOS

Símbolo	Significado
ABS	Acrilonitrilo butadiene estireno
ADC	*Analog to digital converter*
ARM	*Advanced RISC machine*
BDD	Base de datos
CLK	*Clock*
CPHA	*Clock phase*
CPOL	*Clock polarity*
AT	Comandos de configuración
Byte	Conjunto ordenado de 8 bits
DPST	Doble polo simple tiro
GPIO	*General purpose input-output*
Hz	Hertz
IMU	*Inertial medition unit*.
I2C	*Inter integrated circuit*
LIE	Límite inferior de explosividad
LII	Límite inferior de inflamabilidad
LSE	Límite superior de explosividad
LSI	Límite superior de inflamabilidad
LDO	*Low dropout regulator*
MI	Mezcla ideal
OLED	*Organic light emitting diode*
Rx	Pin de recepción de datos serial
Tx	Pin de transmisión de datos serial

PCB	*Printed circuit board*
PLA	*Polylactic acid*
PWM	*Pulse width modulation*
RAM	*Read only memory*
ROM	*Read only memory*
SCBA	*Self-contained breathing apparatus*
SCL	*Signal clock*
SDA	*Serial data*
SPI	*Serial peripheral interface*
SS	*Slave select*
VCC	Suministro positivo de voltaje en transistor de unión bipolar
VDD	Suministro positive de voltaje en transistor de efecto de campo
VEE	Suministro negativo de voltaje en transistor de unión bipolar
VSS	Suministro negativo de voltaje en transistor de efecto de campo
GND	Tierra o retorno en un circuito

GLOSARIO

Actuador Dispositivo capaz de transformar un tipo de energía en otra con la finalidad de generar un efecto sobre un proceso automatizado.

Array Listas de datos ordenados en memoria utilizadas en programación para manipulación de daos múltiples.

Baudio Unidad de medida de velocidad de transmisión de datos expresada en bits por segundo.

Bit Es la unidad mínima de información binaria, ya que puede tener solamente 2 valores, alto y bajo.

Bluetooth Tecnología de redes inalámbricas que permite transmisión múltiple de datos entre diferentes dispositivos en una radiofrecuencia de 2.4 GHz.

Calor Es la cantidad de energía térmica que un cuerpo pierde o gana en contacto con otro. Por lo tanto, es una medida de esta misma.

Char Tipo de variable en programación que ocupa un byte el cual, guarda en memoria un carácter ASCII.

Corriente Es el flujo de electrones que recorre un material conductor a través de un circuito cerrado, medida en culombios por segundo, lo cual equivale a amperios.

Diagrama Es una representación gráfica de las variaciones que obtiene un proceso. Es una representación fácil de analizar de un algoritmo.

EEPROM Electrically erasable programable read only memory.

Entalpia Es la cantidad de energía que contiene una sustancia, representa una medida termodinámica. Se representa con una H mayúscula y se expresa en joules. Su expresión matemática se deriva de la energía del sistema más el producto de la presión por el volumen, por lo tanto, se define que la entalpía depende de la cantidad de materia y de la energía de un sistema. La energía del sistema no se puede medir, por lo tanto, tampoco la entalpía; pero si el cambio de esta, y este cambio representa el calor que entra o sale al sistema desde el ambiente si se mantiene a presión constante.

Fracción de mezcla Fracción de mezcla f define la estequiometría en cualquier punto de un combustor.

Estabilidad Es la capacidad de un módulo o instrumento de mantener su comportamiento referente a la medición

	durante su vida útil y de almacenamiento especificados.
Exactitud	Es la referencia a cuan cerca del valor real se encuentra el valor medido. La exactitud está relacionada con la desviación de una estimación.
Filtro	Capacidad de un proceso de eliminar ruido y mediciones no requeridas de una señal a partir de limitantes específicas.
Float	Tipo de variable en el cual se pueden almacenar números decimales.
Histéresis	Tendencia de los materiales en la cual pueden conservar sus propiedades a lo largo del tiempo.
Humedad relativa	Cantidad de agua en cualquier estado presente en un volumen en relación al aire de ese volumen.
IEEE	Institute of Electrical and Electronics Engineers. Organización profesional a nivel mundial dedicada al avance y servicio tecnológico.
Ingeniería Inversa	Proceso utilizado para descubrir los principios tecnológicos, funciones y operación de un dispositivo con el fin de estudiarlo y utilizarlo.

Integer	Tipo de variable en la cual se pueden almacenar datos numéricos no decimales.
Interrupciones	Es una sucesión de la ejecución de un proceso de programación temporal la cual se lleva a cabo para ejecutar otra porción de código predefinido.
ISO	International Organization for Standardization. Es la mayor organización no gubernamental que crea estándares internacionales.
LiPo	Bateria de polímero de iones de litio.
Llave foránea	Es un campo de una tabla que hace referencia al campo que contiene una llave primaria en otra tabla.
Llave primaria	Es un campo de una tabla que se define como un identificador único de cada registro.
Mezcla pobre	Contiene más combustible del necesario o falta de comburente.
Mezcla rica	Contiene menos combustible del necesario o exceso de comburente.
MISO	Master input slave output.
Mol	Una cierta cantidad de cada elemento de la tabla periódica posee una masa atómica igual a su

	número atómico, dada en gramos. Las moléculas formadas por más de un átomo también poseen masa molecular, pero esta es la suma de todos los números atómicos de los elementos que la componen. La cantidad de moléculas que posee esta masa se denomina una mol.
MOSI	Master output slave input.
NFPA	National Fire Protection Association. Organización encargada de crear normas y mantenerlas para prevenir incendios.
Número de Reynolds	Número adimensional utilizado en mecánica de fluidos para caracterizar el movimiento de un fluido. Se define como la relación entre las fuerzas inerciales y las fuerzas viscosas en un fluido.
Oscilador	Es la base de un reloj en electrónica. Es un sistema capaz de crear perturbaciones o cambios periódicos.
Pitch	Segundo ángulo de Euler utilizado para describir la orientación de un objeto en 3 dimensiones, equivalente al eje Y.
Plano neutro	El plano neutro es la división horizontal entre capas de gases y el aire en un área cerrada, siendo así la división entre gases más densos y gases menos densos. Se llama plano neutro debido a que la

	presión en este punto es idéntica a la presión en el exterior.
Potencia	Es la capacidad medible de realizar una función. En el caso de electrónica, es la multiplicación de la corriente por el voltaje, expresada en watts.
Precisión	Es la referencia a la dispersión del conjunto de valores obtenidos en las mediciones cíclicas de una magnitud especifica. Entre menor sea la dispersión de las mediciones, mayor es la precisión.
Puerto	Espacio destinado para flujo de datos, ya sea por hardware o por software.
Pull down	Resistencia colocada en un pin de datos para mantener el pin en estado bajo cuando el pin está en reposo.
Pull up	Resistencia colocada en un pin de datos para mantener el pin en estado alto cuando el pin está en reposo.
Radicales libres	Son especies químicas capaces de existir independientemente, y tienen electrones desapareados en sus orbitales más lejanos del núcleo. Son altamente reactivos y tienden a compensar el desequilibrio electrónico cediendo o capturando electrones de otras moléculas cercanas.

Repetibilidad	Es la capacidad de un módulo o instrumento de medición de brindar las mismas lecturas de datos en mediciones diferentes realizadas en las mismas condiciones de trabajo.
Resistencia	Es la oposición que representa un material al flujo de corriente eléctrica.
Resolución	Es el incremento más pequeño que tiene un instrumento que permite diferenciar una lectura medida de otra.
Roll	Primer ángulo de Euler utilizado para describir la orientación de un objeto en 3 dimensiones, equivalente al eje X.
Sensor	Dispositivo capaz de captar magnitudes físicas y convertirlas en pulsos eléctricos medibles.
STL	Abreviacion de stereolithography o estereolitografia.
String	Cadena de caracteres o palabras en una secuencia ordenada. Es un tipo de datos inmutable que contiene múltiples caracteres.
UART	Universal asynchronous receiver transmitter.
Voltaje	Es el potencial eléctrico con el que se suministra un dispositivo. Es expresado en voltios.

Yaw	Tercer ángulo de Euler utilizado para describir la orientación de un objeto en 3 dimensiones, equivalente al eje Z.

RESUMEN

En el presente trabajo de graduación se desarrolló el diseño de la primera versión de un elemento de medición, análisis y alerta de riesgos ambientales en incendios estructurales.

El capítulo 1 presenta toda la información básica y resumida de cómo se comporta el fuego pasando por cada una de sus fases desde el inicio; fenómenos, predicciones y variables que llevan a que un evento de baja magnitud se convierta en un evento totalmente fuera de control.

El capítulo 2 describe todos los dispositivos de hardware necesarios microprocesador, microcontroladores, sensores, actuadores, dispositivos de comunicación, alimentación y carga. De igual manera, se muestran los diagramas de conexión, diagramas internos de cada dispositivo, circuito impreso desarrollado y estructura de protección.

El capítulo 3 explica los conceptos, algoritmos y tecnologías utilizadas referente a software y firmware. Se especifica en los conceptos de programación, paradigmas, librerías utilizadas, filtros digitales, configuraciones del sistema, configuración de dispositivos de comunicación, base de datos, algoritmos y diagramas de flujo.

OBJETIVOS

General

Diseñar un sistema autónomo de alerta en incendios estructurales a partir de la medición de variables ambientales dentro del siniestro y desarrollar la explicación de cada una de sus partes para una comprensión fácil y completa.

Específicos

1. Establecer los fundamentos de combate de incendios.

2. Identificar el proceso de diseño de hardware del sistema completo.

3. Identificar el proceso del diseño de software del sistema completo.

4. Presentar el diseño final del sistema completo.

INTRODUCCIÓN

Desde el inicio de los tiempos, el fuego ha sido una base fundamental para el desarrollo de la vida, no se diga para la base del desarrollo de la civilización humana y el desarrollo científico. La ciencia actual se ha fundamentado en gran parte en la observación y experimentación, lo cual es la base del empirismo. El desarrollo de ciencias modernas como la robótica, inteligencia artificial, bioquímica y muchas más no hubieran sido posibles sin un análisis empírico anterior y teorías filosóficas que han llevado a los seres humanos a razonar.

Actualmente, la mayor parte de la documentación científica de combustión hace énfasis en procesos industriales o motores, pero es muy poco lo que se conoce científicamente en cuanto a las características del comportamiento del fuego en diferentes ambientes fuera de control. Cuando se hace referencia a fuera de control es seguro que no se podrá llegar a un valor puntual teóricamente; esto no significa que no se pueda llegar a una tendencia del comportamiento del fuego y anteponerse por medio de análisis estadísticos.

Tradicionalmente, el estudio de la combustión no controlada en incendios no se ha desarrollado a grandes pasos como otros procesos u otras ramas de la ingeniería, motivo por el cual la mayor cantidad de datos o de formulaciones se basan en desarrollos empíricos o experimentales, dando así paso a como se hace y no porqué se hace.

La mayoría de manuales de bomberos en ambientes descontrolados se describe el porqué de las cosas, mas no completamente el como de las cosas;

porque los bomberos se basan en técnicas útiles en el momento de un siniestro, mas no son útiles en el caso del análisis profundo de cómo se produjo un fuego en específico, cómo anticiparse teóricamente, entre otros. Ningún análisis técnico de bomberos puede anteponerse a un evento con bases probabilísticas.

Este texto en su primer capítulo hace referencia de manera resumida a algunos estudios en cuanto a combustión no controlada y explica cómo se lleva a cabo cada una de las partes del fuego desde una fase incipiente hasta una fase final; hace énfasis sobre cada uno de los comportamientos del fuego y sus partes. Los siguientes capítulos se desarrollan cálculos, configuraciones, análisis, construcción y programación de un prototipo electrónico para tomar datos ambientales dentro de un incendio y mostrarlos en tiempo real a la persona que posee el prototipo, para anteponerse a riesgos y prepararse de la manera las técnicas le han capacitado.

El prototipo mencionado tiende a ser similar a lo conocido por los bomberos como un Pass: una alarma si en caso un bombero tiene como una emergencia dentro de un incendio o queda inconsciente. En este caso cuenta con las medidas y el peso aproximados de un Pass, por lo cual el bombero que lo tenga quizás no note la diferencia en cuanto a peso o dimensiones del equipo.

El cambio, que si es radical en equipamiento, es un componente de visualización colocado dentro de una careta de SCBA; muestra todos los datos obtenidos y analizados por el Pass para que la persona que lo tenga colocado este consciente en todo momento de las variables más significativas que intervienen en el lugar.

Algunas de las capacidades del equipo desarrollado son: medición de humedad, temperatura ambiental, temperatura puntual a distancia y gases. De igual manera, puede determinar si una persona está en movimiento para activar una alarma, lo cual hace una función mejorada de un dispositivo Pass normal. Posee una interfaz de usuario fácil de utilizar para observar los datos necesarios de las mediciones; y posee una interfaz para preparar la descarga de datos desde el Pass, revisar conexión, entre otros. Es necesario mencionar que el análisis de dicho sistema actualmente es para incendios cuyas variaciones en temperatura y humedad siguen una trayectoria lineal o exponencial; no es útil en casos donde se generan explosiones o existen cambios radicales sin tener una variación anterior de datos en los sensores.

Este texto comprende solamente la primera versión del dispositivo, la cual es una versión de prueba. Luego, se ha seguido desarrollando dicho dispositivo para minimizar su tamaño y mejorar su utilidad. A pesar de que el dispositivo final puede que tenga muchas diferencias al dispositivo inicial, siempre se basa en los mismos principios del dispositivo de prueba que describe este texto; los cambios más radicales son en cuanto a su construcción y programación, pero sus utilidades siguen siendo las mismas con algunos agregados.

1. FUNDAMENTOS TEÓRICOS Y NECESIDADES DE DESARROLLO

1.1. Fundamentos teóricos sobre combustión

La combustión es uno de los procesos de descomposición química más importantes que han existido en el trayecto del tiempo; seguramente, sin este proceso no se estarían leyendo estas líneas. El fuego es una proyección visual de lo que en realidad es la combustión; base de la civilización actual para el transporte, generar electricidad o simplemente mantener vivos a todos los seres que habitan este planeta. La combustión simplemente es liberación de calor, radiación de luz y la proporción directa que existe entre el desarrollo de la temperatura respecto a la cantidad de gases combustibles y comburentes.

De la misma manera en que la combustión aporta mucho a la vida y al desarrollo, cuando esta se encuentra fuera de control suele ser muy destructiva y en muchos casos, puede llegar a dejar completamente inhabitable en un tiempo determinado cualquier sitio. Estos casos son determinados como incendios o explosiones. Un incendio es una oxidación violenta y descontrolada de partículas en la que interviene un combustible en fase gaseosa que, al mezclarse con un comburente en las proporciones adecuadas, genera luz visible, energía térmica y productos secundarios como gases, humo, etc.

Si se analiza el concepto de combustión con el concepto de incendio, se ve que la combustión es un concepto puntual, mientras que un incendio es el comportamiento de la combustión en un tiempo determinado y sin una variable de control. Para comprender a fondo el proceso de combustión en un incendio,

es necesario tener conocimiento de mecánica de fluidos, termodinámica, transferencia de calor, cinética química, transferencia de masa y comportamientos del fuego en recintos cerrados. Este último se basa en todos los estudios de ingeniería o ciencias puras mencionados anteriormente.

1.2. Elementos para que exista un proceso de combustión

- Material combustible o agente reductor: material que puede ser oxidado. Se pueden presentar en cualquier estado de agregación (solido, líquido o gaseoso).

- Material comburente u oxidante: agente que puede oxidar a un combustible con reacciones electro-químicas.

- Energía de activación: es la energía mínima expresada en calor que debe alcanzar una substancia para iniciar una combustión, para que esta se sostenga así misma independientemente de otras fuentes de energía.

Si se unen los 3 elementos de la combustión, se obtiene el Triángulo del fuego, un modelo geométrico propuesto fundamentalmente para explicar y comprender los mecanismos de extinción del fuego, los cuales serían las mismas formas de destruir el triángulo: quitando uno de sus lados. El triángulo del fuego y sus derivados nos da una idea del concepto físico del fuego, pero no del proceso físico-químico detrás de este. Para comprender el proceso detrás del fuego se debe agregar un elemento más, formando un tetraedro del fuego, el cual comprende una reacción química en cadena.

1.2.1. ¿Qué es una reacción química?

Una reacción química es un proceso de transformar una sustancia con un tipo de propiedades en otra con propiedades diferentes. Todos los compuestos se caracterizan por tener una cierta cantidad de energía que cambia cuando son expuestos a una transformación. Cuando las partículas se combinan o se fragmentan, se produce la liberación o absorción de energía, dejando lugar a reacciones exotérmicas y endotérmicas.

- Reacciones exotérmicas: las reacciones exotérmicas liberan energía térmica hacia la atmósfera que lo rodea, motivo por el cual, aumenta considerablemente la temperatura del sistema completo donde se lleva a cabo la reacción. El calor liberado se debe a que la entalpía de los productos es menor a la entalpía de los reactantes.

- Reacciones endotérmicas: las reacciones endotérmicas absorben energía debido a que la entalpía de los productos es mayor a la entalpía de los reactantes. Estas reacciones no ocurren en condiciones ambientales normales, por lo tanto, no son reacciones espontaneas.

1.2.1.1. Reacción química en cadena

Es la fragmentación de moléculas formando productos intermedios inestables denominados radicales libres. La concentración de los radicales libres es el factor determinante para la velocidad de la llama. En otras palabras, es el proceso o ciclo que interviene para que la combustión se mantenga y se propague de manera independiente a partir de la cantidad de gases que se generan. Esto depende definitivamente de la cantidad del combustible que exista, ya que el combustible es el elemento que tiende a desaparecer en ayuda

de un elemento comburente. Esto es debido a una reacción química llamada pirólisis. Este proceso libera calor, por lo cual es una reacción endotérmica. En este punto, se extiende un proceso de gasificación, sublimación o cualquier otra transformación del combustible que da lugar a que la combustión se mantenga sin intervención externa.

1.2.1.2. ¿Qué es pirólisis?

Es un proceso químico de descomposición térmica de cualquier material (excepto vidrio y metal) en una atmósfera con comburentes deficientes, el cual transforma el material expuesto a una mezcla líquida de hidrocarburos, agua, gases combustibles y residuos secos de carbón.

1.2.1.3. Comportamiento del combustible según su estado físico inicial

A pesar que la combustión siempre es con base en los gases generados por cualquier materia, líquida, sólida o gaseosa, el comportamiento del fuego en combustibles sólidos, líquidos o gaseosos es completamente diferente, ya que el estado inicial de estos determina el proceso que estos llevarán a cabo al entrar en combustión.

- Estado gaseoso: normalmente, no son perceptibles a simple vista. Las partículas tienden a dispersarse en la mayor parte del volumen en el que se ubican. Tienen alta movilidad en el medio y enlaces débiles, por lo tanto, son enlaces de poca energía. Parece lógico que se necesite una fuente de energía, aunque sea baja, para separar dichas moléculas en el caso de gases inflamables; pero en realidad esta energía es tan baja que se supone que con cualquier fuente de ignición sumamente baja activa

este combustible. Los combustibles en estado gaseoso, al tener una menor densidad que los demás estados, la movilidad es mucho mayor, por lo cual, la velocidad a la que puede transcurrir un proceso de combustión es sumamente alta, incluso puede apreciarse en muchos casos como una onda de choque.

- Estado líquido: este caso se parece mucho al estado gaseoso, la diferencia es que los líquidos son más densos y las partículas se encuentran unidas con mayor cohesión. Otra limitante de este estado es que las partículas tienen un límite de dispersión relacionado con base en una tensión superficial del elemento. En el caso de los líquidos ya que sus enlaces son más fuertes, se necesita una mayor energía para activarlo, ya que se necesita la energía suficiente para gasificar un porcentaje de este material para iniciar un proceso de combustión.

- Estado sólido: en este tipo de materiales, se necesita una energía sumamente alta para romper enlaces, ya que las moléculas en este caso si tienen enlaces muy fuertes y de igual manera, se necesita gasificar parte del material para iniciar una combustión. La movilidad de las partículas es prácticamente nula y la densidad de estos elementos normalmente es muy alta, por lo cual, un proceso de combustión es proporcionalmente más lento que los gases y los líquidos.

1.2.1.4. Energía de activación

Hasta este punto, se ha interpretado la manera como se comporta el fuego en relación con un comburente y un combustible, pero falta comprender uno de los lados del triángulo, que es la energía de activación, la cual comprende la energía mínima que necesita un sistema antes de iniciar una combustión. Esta

energía se presenta como energía térmica y puede iniciar una reacción en cadena al entrar en contacto con una mezcla de combustible-comburente. Las maneras en las que puede entrar en contacto una cantidad de calor, con algún material son las siguientes:

- Conducción: es el proceso de transmisión de calor por contacto directo entre dos cuerpos sin intercambio de materia, por el que el calor fluye del cuerpo con mayor temperatura, al cuerpo con menor temperatura. Esto es debido a la agitación térmica de las moléculas cuando no existe un desplazamiento real entre ellas.

- Convección: es el proceso de transmisión de calor que se caracteriza por producirse en fluidos, y este transporta el calor entre zonas con diferentes temperaturas, motivo por el cual, el fluido siempre se mantiene en movimiento y existe intercambio de materia. En este caso si existe desplazamiento real de las moléculas de una sustancia.

- Radiación: es el proceso de transmisión de calor el cual, se genera a distancia, sin contacto, solamente por la intensidad de calor que depende de la temperatura que puede producir un cuerpo y de la longitud de onda considerada. La manera de transmitir el calor en este caso es por medio de ondas electromagnéticas. Estas ondas que viajan a velocidad de la luz y, al entrar en contacto con un cuerpo, producen calor. La radiación térmica tiene las mismas propiedades de cualquier onda electromagnética, las cuales son las siguientes:

    - Reflexión: modificación que se produce en la dirección de una onda.

- Refracción: modificación en la dirección y velocidad de una onda al cambiar el medio en el que se propaga.

- Dispersión: descomposición de una radiación compleja en diferentes radiaciones simples.

- Difracción: fenómeno característico de las ondas que se basa en la desviación de estas al encontrar un obstáculo o rendija.

- Absorción: proceso por el cual la radiación es captada por la materia.

1.2.2. Tipos de combustión

- Combustión completa: es una combustión ideal. En este tipo de combustión no existe ningún producto secundario y por esto mismo, la mezcla entre combustible y comburente se consume por completo ya que se encuentran en proporciones correctas en una atmósfera determinada. Normalmente, en este tipo de combustión, el residuo que queda al final es casi nulo en comparación con la cantidad de materia inicial. De igual manera se alcanza una temperatura óptima, siendo esta la mayor temperatura que es posible alcanzar con un tipo de combustible. Los únicos derivados son dióxido de carbono (CO_2) y agua (H_2O).

- Combustión incompleta: en este tipo de combustión, no están mezclados en proporciones correctas el combustible con el comburente, simplemente existe un déficit de uno de los dos, motivo por el cual existe humo (Combustible), carbono (C), gases como monóxido de carbono (CO), dióxido de carbono (CO_2), hidrógeno(H), entre otros. En este tipo

de combustión, la cantidad de residuos es alta, dejando muchas veces combustible sin quemar, y la temperatura máxima alcanzada es inferior a la que se puede alcanzar en condiciones ideales de una combustión completa.

- Combustión con llamas: la combustión con llamas normalmente se mantiene activa con una reacción en cadena que gasifica los combustibles y estos se queman, siendo una reacción en cadena no inhibida. La difusión y re ignición de la combustión depende mayormente del calor de la llama. La apariencia de la llama en este tipo de combustión puede dar información sobre la eficiencia de este proceso.

Las llamas se pueden clasificar según el movimiento de los fluidos involucrados como llamas laminares o llamas turbulentas. De igual manera en ambos casos, se pueden diferenciar como llamas perfectamente mezcladas (premezcladas) o llamas con reactantes separados antes de la combustión (difusión). En un incendio, se pueden encontrar llamas en un estado intermedio, conocidas como llamas parcialmente premezcladas, o llamas en la que una corriente es turbulenta y la otra laminar.

- Llamas laminares: se caracterizan por tener números de Reynolds bajos aproximadamente inferiores a 2000. Este tipo de llamas se presenta normalmente en quemadores o salidas muy pequeñas y tienen un frente continuo y muy bien definido.

- Llamas turbulentas (llamas rellenas): se caracterizan por tener un numero de Reynolds muy alto, variante entre 2 000 y 10 000. El frente de este tipo de llama oscila a gran velocidad. En algunos casos estas producen sonidos desarticulados y se caracterizan por presentar muchos remolinos.

- Llamas de difusión: en este tipo de llamas, el rendimiento en el proceso de combustión es de aproximadamente 25 % respecto a la llama en el mismo combustible en óptimas condiciones. Teniendo esto en cuenta, se puede determinar que la mayoría de conatos de incendios inician con llamas de difusión, generando gases y obteniendo oxígeno directamente del ambiente. Este tipo de llamas normalmente son las que generan menor cantidad de energía térmica teniendo una menor longitud de onda y de igual manera, una menor velocidad de deflagración. Estas llamas se ven de color amarillo a rojo y no son eficientes. Las llamas de difusión laminares ocurren cuando el comburente y combustible se presentan separados en la combustión y el número de Reynolds del flujo es suficientemente pequeño para no permitir turbulencia.

- Llamas premezcladas: este tipo de llamas ocurren cuando se mezcla el combustible y el comburente antes de iniciar la combustión, aumentando así significativamente la eficiencia de la llama. La cantidad de gases sin quemar se reduce en grandes cantidades. Estas llamas generan una mayor cantidad de energía térmica teniendo una mayor longitud de onda, es por esto que la llama se ve de color azul.

- Combustión sin llamas o latente: es una reacción exotérmica en combustibles sólidos porosos, no existe reacción en cadena, ya que se agotan rápidamente las sustancias volátiles. Este tipo de combustión eleva la temperatura de todo el ambiente cercano a temperaturas muy altas, siendo una combustión como sólido incandescente, con oxígeno en contacto con la superficie del combustible.

1.2.3. Reacciones de combustión

La reacción de un combustible respecto a un oxidante puede dar diferentes resultados dependiendo de la proporción tanto de combustible como de oxidante, y a su vez de la velocidad de propagación del frente de la llama. La velocidad de propagación se define de 3 maneras:

- Oxidación lenta: la liberación de calor por unidad de tiempo es muy baja. La luminosidad de la llama aún no se aprecia. Este es un caso límite de una deflagración y no es de mayor interés en estudios de combustión.

- Deflagración: la liberación de calor por unidad de tiempo es mayor, en este caso si se aprecia luminosidad. La velocidad del frente de la llama es menor a la velocidad del sonido.

- Detonación: es una violenta liberación calor, luminosidad y sonido que sobrepasa la velocidad del sonido y genera una onda expansiva detrás del frente de la llama. Es muy común que una detonación se ejecute a partir de una deflagración.

1.2.4. Gases y fluidos

Los gases son producto de una combustión incompleta, por lo tanto, son un conjunto de combustible y comburente mezclado en algunos casos. Lo interesante a saber en el caso de los gases, aparte de su toxicidad y su comportamiento en relación al desarrollo de la combustión es el comportamiento del volumen y la presión en relación a la combustión.

1.2.4.1. Constante universal y constante particular de los gases

Es una constante física que relaciona varias funciones de termodinámica, relacionando entre si variables de temperatura, energía y masa, siendo la masa en un volumen determinado la densidad del gas.

Los gases se pueden comprimir, por lo tanto, la densidad depende de la presión ejercida en el volumen determinado. De igual manera, al ser el volumen del gas una función de la temperatura, también lo es la densidad. La forma más particular de la ley de gases es la siguiente:

$$P = \frac{\rho \cdot @}{W} \cdot T \qquad \text{[Ec. 1]}$$

- Presión total o de estancamiento $P[Pa]$
- Constante universal de los gases@$= 8314{,}4[m^2/s^2/K]$
- Masa molecular $W[kg/mol]$
- Temperatura total o de estancamiento $T[K]$

La razón entre la constante universal y la masa molecular se designa como el constante particular de un gas.

La cantidad de moléculas que, sumadas proporcionan el total de la masa del gas (según los números atómicos), está propuesta por el número de Avogadro, que se aproxima a 6,023 E^{23} moléculas. Esta cantidad de moléculas está denominada como 1 mol. La masa molecular de un gas seria la multiplicación del número de gramos de un elemento por mol.

Si se analiza la ecuación 1, se puede despejarla de cierto modo que quede de la siguiente manera:

$$\frac{@ \cdot T}{P} = \frac{W}{\rho}$$

El lado izquierdo de la igualdad no está en función del gas, ya que es una relación constante por una razón de temperatura y presión. El lado derecho de la igualdad tiene unidades de volumen por mol, lo cual si está directamente relacionado con el gas. Con esto se deduce que, para una temperatura y una presión determinada, el volumen del gas siempre será el mismo. Este volumen molar con una presión de 101 325 Pa, que equivale a una atmósfera y una temperatura ambiente promedio de 273,15 K es aproximadamente de 0,0223 m^3/mol, lo cual refleja que, en condiciones estándar, una mol de todos los gases ocupa 22,4 litros, podemos entender de igual manera que hablar de gases es hablar de moléculas.

1.2.4.1.1. Mezclas de gases

El aire es una mezcla de gases, en esta mezcla se encuentra oxígeno molecular [O_2], nitrógeno molecular [N_2], dióxido de carbono [CO_2], vapor de agua [H_2O] y varios gases inertes en mucho menores cantidades, como el argón, xenón, entre otros. En incendios, ni el N_2 ni los gases inertes reaccionan en la combustión, se agrupan como un volumen que no se ve afectado, por esto mismo se dice que el aire está formado por 21 % de O_2 y 79 % de N_2, llamando a este mismo como aire técnico simplificado (ATS) y la masa promedio de este aire es de 28,85 Kg/mol.

1.2.4.1.2. Propiedades del aire y los gases combustibles

La mayoría de combustibles se queman en estado gaseoso, y las propiedades más características de estos, ya sea en una reacción controlada o descontrolada son las siguientes:

- Composición

La composición del combustible es fundamental para determinar la reacción de este en el proceso de combustión, de igual manera ayuda a determinar si los residuos de este serán nocivos o no. Ambas cosas son propiamente de la composición del combustible. La forma ideal de indicar la composición de un gas es en porcentaje de cada uno de sus componentes en presión y temperatura ambientales, por ejemplo:

  - Oxígeno: O_2
  - Dióxido de carbono: CO_2
  - Sulfuro de hidrógeno: SH_2
  - Vapor de agua: H_2O

- Poder calorífico

No es más que la cantidad de energía que se desprende en la combustión en relación a la cantidad de masa inicial del combustible.

- Viscosidad

En esta propiedad no influye la cantidad de muestra que se tome para estudiarla; esto se conoce como propiedad intensiva y tiene suma importancia

en combustibles de estado inicial líquido. Los estudios de viscosidad actualmente se realizan de manera experimental.

En cuanto a los gases, la viscosidad está relacionada en una ecuación en función de la variación de la temperatura. Comúnmente se evalúa utilizando la ecuación de Sutherland, que es la siguiente:

$$\mu = ({}^{T}/_{273,15})^{3/2} * ({}^{273,15 + C_s}/_{T + C_s}) * \mu_0$$

- C_s: constante de Sutherland
- μ_0: viscosidad dinámica a $0\,^{o}C$
- μ: viscosidad dinámica [kg/m/s]
- T: temperatura total o de estancamiento [K]

- Densidad

En los combustibles gaseosos se utiliza la densidad absoluta, expresada en Kg/m³, como la relativa en el aire, que es adimensional. El motivo de esto es dado que la referencia es la atmósfera. La densidad relativa se define como:

$$\rho_r = {}^{\rho}/_{\rho_a}$$

- ρ_r: densidad relativa
- ρ_a: densidad absoluta del aire $[Kg/m^3]$
- ρ: densidad absoluta del gas $[Kg/m^3]$

La densidad absoluta del aire, a una atmósfera y 0 ^{o}C es de 1,287 Kg/m³.

Si un volumen de gases está formado por un número infinito de gases, se puede calcular la densidad relativa media del gas mezclado como tal, siendo la sumatoria de la fracción molar correspondiente al gas por la densidad relativa del mismo.

En algunos casos se puede encontrar que la terminología de la densidad es en grados API, no es más que una forma de expresar esta misma. API significa American Petroleum Institute, la medida es en comparativo con el agua a temperaturas iguales y precisas que tan pesado o liviano es en referencia a esta. Normalmente, se utiliza con petróleo y sus derivados. Para poder obtener grados API, la ecuación es la siguiente:

$$G = \frac{141{,}5}{\rho} - 131{,}5$$

La siguiente tabla muestra la información de densidad absoluta, densidad relativa y constante de Sutherland de los gases más comunes en incendios:

Tabla I. **Densidad de los gases más comunes en incendios**

Gas	Cs	Densidad Absoluta Kg/m^3	Densidad Relativa
Aire (ATS)	120	1,293	1
Nitrógeno N_2	111	1,250	0,967
Oxígeno O_2	127	1,429	1,105
Dióxido de carbono CO_2	240	1,973	1,526
Monóxido de carbono CO	118	1,250	0,967
Hidrógeno H_2	72	0,089	0,069
Amoniaco gaseoso NH_3	370	0,767	0,593
Dióxido de azufre SO_2	416	2,894	2,238

Fuente: elaboración propia.

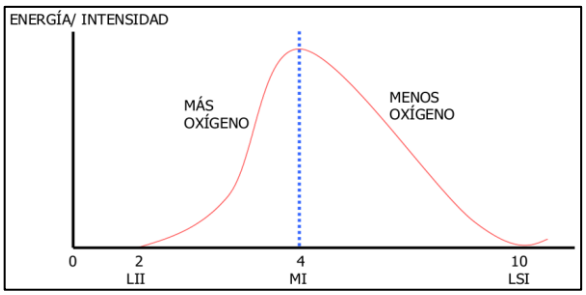
- Límites de inflamabilidad

Las reacciones químicas de combustión idealmente ocurren a cualquier temperatura, incluso en temperaturas muy bajas, la diferencia es la velocidad de propagación. Dado que ningún ambiente es completamente ideal, las reacciones dependen también de la cantidad de aire que exista alrededor, ya que, si la cantidad no es la óptima, la combustión simplemente es muy difícil de obtener.

Los límites de inflamabilidad están sumamente relacionados con la cantidad del combustible y del comburente. El límite superior de inflamabilidad (LSI) es cuando la densidad de gases combustibles es mucho mayor a la densidad de comburente en un mismo volumen, motivo por el cual no existirá combustión. El límite inferior de inflamabilidad (LII) es cuando la densidad de comburente es mucho mayor a la densidad de gases combustibles en un mismo volumen, motivo por el cual tampoco existirá combustión. El rango de inflamabilidad se encuentra dentro del límite inferior de inflamabilidad y el límite superior de inflamabilidad. En este rango se puede definir la mínima temperatura y máxima a la que un combustible puede emitir gases necesarios para que, al mezclarse con un comburente, pueda iniciar una combustión.

El punto de mezcla ideal (MI) es la combinación óptima de combustible y comburente para que esta reacción tenga un 100 % de eficiencia. En este punto, la combustión es total, mientras que en los límites de inflamabilidad la combustión es deficiente. En la figura 1 se observan los límites del propano, siendo el límite inferior de inflamabilidad un 2 % en un volumen, y el límite superior de inflamabilidad un 10 % en el mismo volumen. La proporción óptima para generar la mayor intensidad de llama y energía térmica es un 4 %.

Figura 1. **Límites del propano**

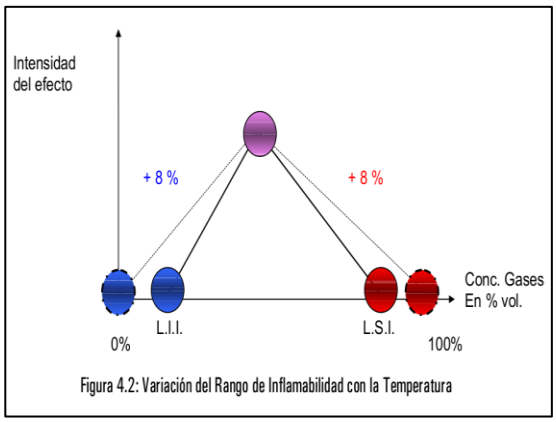


Fuente: BASSET BLESA, José Miguel. *FlashOver, desarrollo y control*. p. 19.

El concepto de límites inferiores de explosividad(LIE), límites superiores de explosividad(LSE) y rango de explosividad es similar, la diferencia es la misma diferencia entre inflamable y explosivo. Un material inflamable es un material susceptible a quemarse con llama a baja velocidad de propagación en relación al volumen del combustible, liberando de esta manera energía a una velocidad menor a la velocidad del sonido y sin una onda expansiva, esto se le conoce como deflagración.

La temperatura puede afectar tanto al combustible como al comburente, por lo cual puede aportar energía para llevar un combustible a un rango de inflamabilidad y puede disminuir el efecto de refrigeración en una atmósfera saturada de gases. Al disminuir el efecto refrigerante del aire y gases en un espacio cerrado, existe variante no solo en función del combustible, sino ahora también en función de la concentración de energía térmica en este espacio. Cuando esto ocurre, pueden variar los límites de inflamabilidad de un combustible hacia los extremos, aumentando su rango de inflamabilidad como se muestra en la ilustración 2.

Figura 2. **Rango de inflamabilidad**



Fuente: BASSET BLESA, José Miguel. *FlashOver, desarrollo y control*. p. 23.

De igual manera, la presión puede afectar los límites de inflamabilidad. Al aumentar la presión en una habitación cerrada, los límites de inflamabilidad se separan (normalmente, el LSI aumenta en una proporción mucho más alta que el LII), aumentando así el rango de inflamabilidad. Si la presión disminuye, los límites se acercan, disminuyendo el rango de inflamabilidad. Por lo cual, se puede decir que el rango de inflamabilidad es directamente proporcional a la presión relativa.

- Densidad de oxígeno y su efecto en la combustión

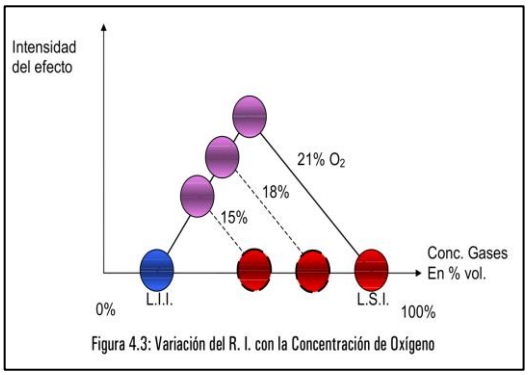
La densidad de oxígeno afecta los límites de inflamabilidad, solo que de una manera completamente diferente a la temperatura. Para que una combustión sea ideal, se necesita una proporción de oxígeno y combustible concreta para que se consuman por completo. Si a esta proporción se le elimina un porcentaje de oxígeno, lógicamente la cantidad de combustible que arderá será menor. Si se aumenta un porcentaje de oxígeno, existirá un volumen

remanente de oxígeno al ser consumido todo el combustible (siempre permaneciendo dentro del rango de inflamabilidad).

En el LII se influye ningún cambio drástico respecto a la densidad de oxígeno, ya que el LII existe cuando la cantidad de oxígeno es mínima. Los cambios significativos existen en el LSI, ya que, si existen cambios en la densidad del oxígeno en este punto, ya sea eliminando o agregando este, la variación en el LSI será lineal.

La saturación de combustible con oxígeno carente provocará que el LSI sea más cercano al LII, mientras que la saturación de oxígeno y combustible carente provocara que el LSI sea más lejano del LII.

Figura 3. **Saturación de combustible con oxígeno**



Fuente: BASSET BLESA, José Miguel. *FlashOver, desarrollo y control*. p. 24.

En la tabla II se muestra una cantidad de gases que tienden a quemarse más rápido junto a los límites inferiores y superiores de inflamabilidad:

Tabla II. **Cantidad de gases que tienden a quemarse más rápido**

Gas	LII	LSI
H_2	4,0	75,0
CH_4	5,0	15,0
C_2H_6	3,2	12,45
C_3H_8	2,4	9,5
i-C_4H_{10}	1,8	8,4
n-C_4H_{10}	1,9	8,4
C_5H_{12} (pentano)	1,4	7,8
C_6H_{14} (hexano)	1,25	6,9
C_7H_{16} (heptano)	1,0	6,0
C_2H_4 (etileno)	3,05	28,6
C_3H_6 (propileno)	2,0	11,1
C_4H_6 (butadieno)	2,0	11,5
C_2H_2 (acetileno)	2,5	81,0
C_6H_6 (benceno)	1,4	6,75
CO	12,5	74,2
NH_3	15,5	27,0
SH_2	4,3	45,5

Fuente: elaboración propia.

Si se necesita saber los límites de inflamabilidad de una mezcla gaseosa, de igual manera se debe tomar en cuenta la siguiente ecuación:

$$L_{ii} = \frac{1}{\sum_{i=1}^{n}(x_i/L_i)} L_{si} = \sum_{i=1}^{n} \quad x_i * L_i$$

Donde: x_i es la fracción molar de un gas i. L_i es el límite de inflamabilidad del componente.

- Punto de inflamación o temperatura de ignición

Para que se produzca combustión, la mezcla de reactante - comburente debe partir de una temperatura mínima; esta temperatura mínima tiene el nombre de punto de inflamación o temperatura de ignición. Cuando esta temperatura es alcanzada, la reacción en cadena posiblemente supere este punto de temperatura. Mientras el comburente y los límites de inflamabilidad se mantengan, la reacción se acaba hasta que se agote el reactante.

- Temperaturas características

Las temperaturas características son 4 puntos a los cuales existe un cambio de estado en un material al aplicarle una cantidad de calor. Son los siguientes:

  - Punto de inflamación (*flash point*): temperatura mínima a la que se necesita calentar un material o un producto para que entren en combustión los gases emitidos momentáneamente en presencia de llama bajo condiciones específicas.

  - Punto de combustión (*fire point*): temperatura mínima a la cual prende un material en su máxima capacidad y continúa ardiendo durante un tiempo especificado después de aplicarle una llama normalizada.

  - Punto de Ignición (*ignition point*): temperatura mínima a la cual puede iniciarse una combustión sostenida bajo condiciones específicas de condiciones físicas (presión, temperatura, humedad, entre otros). Este punto crea una combustión sostenida.

- Punto de autoignición (*auto-ignition point*): temperatura a la cual se obtiene la autoignición en un ensayo de fuego. Es una ignición resultante de un aumento de temperatura sin una fuente de ignición separada.

- Fuentes de ignición

La energía de activación, inicialmente en un tiempo 0, es la fuente de ignición que inicia todo el proceso de combustión, convirtiendo este proceso en cualquier tipo de combustión de los mencionados anteriormente. Estas fuentes de ignición son muy importantes en un incendio, ya que esta puede determinar el comportamiento de la magnitud del incendio en función del tiempo. Las fuentes de ignición pueden ser de 3 tipos:

- Fuentes de ignición abiertas: este tipo de ignición permanece activa mientras existe una fuga de gas o una combinación de varios gases. Se da siempre cuando la cantidad de comburente es mayor a la cantidad de combustible, por lo cual se inicia en el límite inferior de inflamabilidad (LII). Esta fuente de ignición si interactúa con la capa de gases generada por una combustión incompleta.

- Fuentes de ignición ocultas: este tipo de ignición permanece dentro de un margen establecido físicamente, por lo cual tiende a demorar el proceso de combustión de la capa de gases generados por una combustión incompleta. No interactúa directamente en esta capa de gases. En presencia de este tipo de fuentes de ignición, la capa de gases se convierte en una capa con mayor volumen, mayor densidad o ambas, y al no tener contacto la llama

directamente con estos, puede generar una combustión proporcionalmente mayor a la que generaría una fuente de ignición abierta.

- Fuentes de ignición intermitentes: este tipo de fuente de ignición se observa eventualmente cuando en una ubicación determinada, se satura de gases inflamables se conserva hasta alcanzar el límite superior de inflamabilidad. Esta fuente se encuentra en función del flujo del gas y de la ignición del mismo y se puede decir que es una fuente cíclica.

- Temperatura de combustión

Es la temperatura de máxima llama o el punto máximo que puede alcanzarse, similar al punto de combustión, la diferencia entre el punto de combustión y la temperatura de combustión es que el punto es el inicio de este proceso.

- Contenido de azufre

El azufre se encuentra en la tabla periódica con el símbolo S y es un material abundante. Algunos usos en la industria son fabricación de pólvora, fósforos, laxantes, insecticidas, jabones, tinturas, plásticos, pigmentos, etc. Es necesario saber el contenido de azufre en los combustibles que estén ardiendo, ya que de esta manera podemos tener una idea de cuánto SO_2 (dióxido de azufre) se tendrá en la atmósfera cercana. El primer motivo por el cual se necesita saber el azufre de los combustibles es porque es un contaminante que puede causar un severo daño a las vías respiratorias, y dependiendo la cantidad de azufre que una persona inhale puede ser mortal (aproximadamente

500 mg/m³). El segundo motivo es porque en una combustión, el SO_2 se oxida lentamente convirtiéndose en SO_3 (trióxido de azufre), y este componente químico es el responsable de las lluvias ácidas.

Una manera de intentar reducir los efectos del SO_3 en la atmósfera es reducir el volumen enviado hacia esta, y la manera ideal de hacerlo en un incendio es controlar el exceso de aire, de tal forma que, en un incendio, si existe la necesidad de exceso de aire, se emplee la menor cantidad posible.

- Humo

El humo en un incendio como ya se ha mencionado anteriormente, es producto de una combustión incompleta, el cual está compuesto de diferentes tipos de gases y material combustible en diferente estado. Detectar el comportamiento y la composición del humo puede ayudar en la mayoría de casos a predecir el comportamiento del fuego.

Los atributos más útiles en la lectura del humo son volumen, velocidad (presión) y densidad, los cuales han sido estudiados y relacionados anteriormente a la temperatura. Adicional a estos atributos está el color del humo, que indica en muchos casos que tipo de combustible se está quemando.

El volumen del humo puede indicar un aproximado de la cantidad de combustible que se está quemando en un lugar y la proporción de oxígeno; sin embargo, solamente obtener información del volumen no es útil. Muchos combustibles domésticos hoy en día, pueden provocar grandes volúmenes de humo y una pequeña llama en proporción al humo o viceversa.

La velocidad del humo se encuentra en función de la presión, lo cual indica como se ha aumentado la misma dentro de una habitación en relación a la temperatura o al volumen. En otras palabras, un cambio en la velocidad del flujo del humo puede representar 2 cosas: la primera puede ser un cambio proporcional de la velocidad del humo respecto a la temperatura del lugar, lo cual puede volver a una atmósfera estable en una completamente inestable y peligrosa; la segunda puede ser un cambio proporcional de la velocidad del humo respecto al volumen, el cual puede representar ya sea una cantidad de combustible muy grande implicada en la combustión, o una combustión en un lugar donde el flujo de salida es menor a la cantidad de humo generado, lo cual de la misma manera puede ser un riesgo muy alto, ya que puede ocasionar un *flash over* (en caso de contar con la cantidad de oxígeno suficiente) o un *back draft*. Para predecir un *flash over* visualizando solamente el humo, se debe observar si el flujo de humo es turbulento o laminar, ya que este cambio depende absolutamente de la velocidad y la presión interna donde se produce la combustión.

1.2.5. Etapas de un incendio

El fuego en una combustión de cualquier tipo si se encuentra en un ambiente ideal o no, tiene fases dependientes del tiempo. Se puede decir que las variables que se tengan en un inicio de un incendio seguramente no serán las mismas que al final del mismo. Así que la extinción de un incendio y la seguridad de la estructura del mismo, no depende solo del triángulo del fuego y sus derivaciones, también, se encuentra en función del tiempo desde que este se inició. La mayoría de incendios en estructuras cerradas sigue un patrón definido, el cual se comporta de manera diferente las siguientes fases:

- Fase incipiente o fase de ignición: en esta fase, el oxígeno en el ambiente cerrado no ha sido reducido en proporciones mayores. Por este motivo, la combustión es prácticamente completa, y la reacción en cadena de este puede llegar a ser exponencial. El calor de la llama en esta fase puede ser de aproximadamente 538 °C, pero al ser un proceso endotérmico, la mayor parte de la energía se consume en la misma combustión y no en la temperatura del ambiente, que tiene variaciones muy pequeñas en relación a la llama.

- Fase de crecimiento: esta fase existe poco después de la fase de ignición. Y en este caso se inicia la formación de gases poco densos sobre la superficie superior del lugar. A medida que esta capa de gases se desarrolla, se inicia un proceso de succión de aire desde los alrededores del fuego hacia el fuego. La velocidad de esta etapa se encuentra en función del volumen de aire que puede existir dentro del lugar y de la posición del punto de ignición del incendio.

El motivo por el que depende de la posición del punto de ignición es por la cantidad de aire frio que puede succionar este mismo. Si el aire proviene de 360° alrededor (el centro de una habitación), esta fase será más lenta que si proviene de 90° alrededor (una esquina de una habitación). Por lo cual, si una fuente de ignición existe cerca de una pared, esto implica que existe menor cantidad de aire refrigerante sobre la llama y, por lo tanto, mayor temperatura en la columna de humo y gases. Esto afecta de manera característica la temperatura de la capa de gases en la parte superior del incendio, y a su vez la velocidad de dispersión sobre el mismo. Si los gases ya no tienen espacio para dispersarse, el volumen y la densidad de esta capa comienza a aumentar. Esta etapa normalmente se encuentra controlada por combustible.

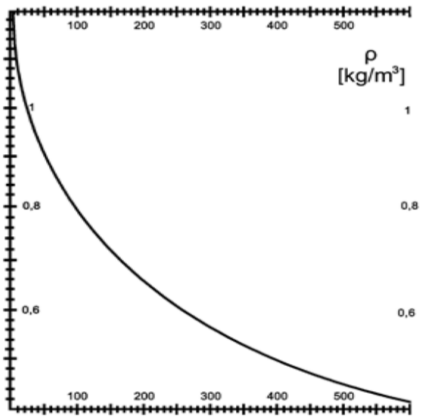
Si fuera el caso que esta etapa es controlada por ventilación, el plano neutro se mantendrá elevado, mientras que la succión de oxígeno generará presión negativa en el lugar como consecuencia de liberación de presión en la zona de presión positiva y, cuando el aire succionado alcance el foco del incendio, aumentaran los gases generados por pirólisis. A su vez, aumentará de nuevo la presión en la habitación y descenderá la densidad del oxígeno.

- Combustión súbita generalizada o *flash over*: esta es la transformación de la fase de crecimiento a la fase de incendio desarrollado. Este cambio es muy rápido, y claramente puede marcar un incendio que puede ser controlado rápidamente a un incendio totalmente fuera de control. Si en este punto existen personas dentro del recinto, corren peligro extremo, aunque se encuentren totalmente equipados y cuenten con equipo de protección personal completo. Esta parte se encuentra descrita debajo en la parte de comportamiento del fuego en recintos cerrados.

- Fase de combustión libre o incendio totalmente desarrollado: en esta fase se consume en proporciones mayores el oxígeno dentro del ambiente y el aire rico en oxigeno es arrastrado hacia las llamas, convirtiendo el incendio controlado por ventilación. La generación de gases en esta fase es alta, ya que la necesidad de oxígeno convierte el proceso en una combustión incompleta. El calor liberado y el volumen total de gases generados dependen del área total de los agujeros que existan ventilando la habitación. Si no existe ventilación, los gases generados en esta fase pueden originar incendios en otros lugares que tengan conexión por tubería de ventilación o cualquier otro medio en el cual se pueda transportar el calor por convección. En esta fase, también los gases más calientes forzan a los gases menos calientes a ir hacia la parte inferior, así facilitando la ignición de todos los materiales en el área

superior del lugar. En este momento, las temperaturas de la parte superior pueden alcanzar aproximadamente los 700 °C y a medida que el fuego crece y tiene acceso a oxígeno el plano neutro desciende.

Figura 4. **Variación de la densidad del aire respecto a la temperatura**

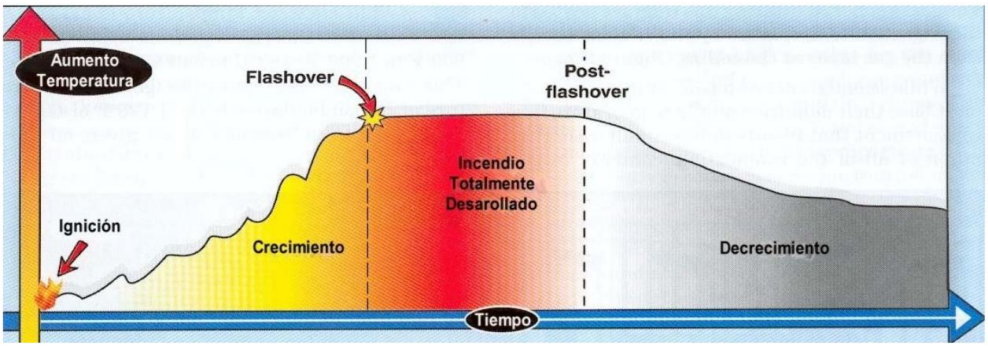


Fuente: ARNALICH, Arturo. *Incendios de interior, ventilación de incendios*. p. 5.

- Fase latente o decrecimiento: en esta fase es muy probable que dejen de existir llamas si el área se encuentra muy cerrada, ya que existe una deficiencia de oxígeno y el incendio se convierte en incendio controlado por ventilación, por lo cual la combustión se reduce a brasas incandescentes o a líquidos en ebullición. Siendo este un proceso exotérmico o una fase de combustión sin llama, la mayor parte de la energía térmica es transferida al ambiente que rodea la combustión alcanzando así una temperatura arriba de 538 °C. El calor intenso tenderá a vaporizar fracciones ligeras de combustibles como hidrógeno o metano, añadiendo estos a la lista de gases, esto incrementa el peligro para bomberos y crea la posibilidad de una explosión por flujo reverso.

Si en esta fase el incendio es controlado por combustible y no por ventilación, el fuego consume el combustible disponible y la energía térmica disminuirá hasta agotar todo tipo de fuente de ignición y terminar el proceso de combustión.

Figura 5. **Fase latente o decrecimiento**



Fuente: BASSET BLESA, José Miguel. *FlashOver, desarrollo y control*. p. 25.

Estas fases, en caso de detectarlas, las maneras son las siguientes:

- Incipiente o inicial: detector iónico
- Combustión libre: detector óptico
- Combustión libre: detector de llamas
- Latente: detector térmico

Al tener muchas posibilidades de cambios de variables dentro de un incendio en función del tiempo, es posible que existan muchos comportamientos del fuego si existe la situación adecuada. Estos comportamientos del fuego no son nada más que magnitudes de variables correctas en tiempos correctos y una atmósfera correcta, en este caso el enfoque es a ambientes cerrados.

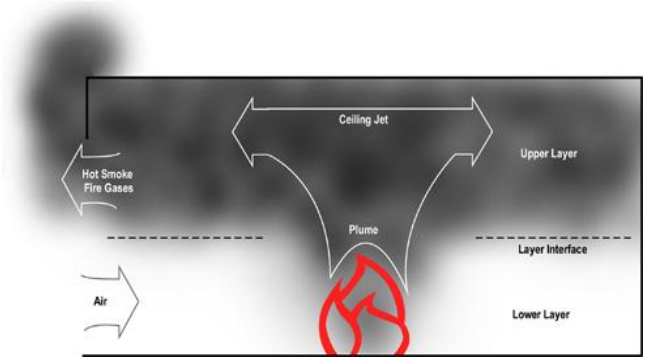
1.2.6. Comportamiento del fuego en espacios cerrados

La diferencia entre incendios en espacios cerrados y espacios abiertos es que, en espacios abiertos, los gases generados en la primera fase del incendio si pueden fluir libremente alejándose de cualquier punto de ignición y, de igual manera, en un incendio abierto el fuego se alimenta del oxígeno que se encuentra en el ambiente, siendo este más frio que el que se encuentra calentado por la llama. Esto ayuda a que los gases generados puedan refrigerarse evitando de esta manera que puedan alcanzar un punto de inflamación. La diferencia con un entorno cerrado es que en este caso no fluyen libremente los gases hacia el exterior, quedando atrapados entre el mismo oxígeno que alimenta el fuego. Esto a su vez lo que genera es que el oxígeno se caliente y elimina todo posible efecto refrigerante mencionado anteriormente.

El cambio de la temperatura promedio de un espacio cerrado puede variar la densidad del aire, volviendo menos denso el aire caliente que el aire frio. La diferencia de densidades entre los gases fríos y calientes genera un flujo vertical del humo hacia las partes más altas, siendo la velocidad de este flujo es directamente proporcional a la temperatura del mismo. El cambio de temperatura de igual manera puede variar la presión en el recinto, siendo la densidad menor la que sufra cambios de presión más rápidos. A la línea que divide los gases calientes de los gases fríos se le denomina plano neutro.

Por encima del plano neutro existe una presión superior a la presión atmosférica, por lo cual el humo y gases calientes intentan salir del lugar. Por debajo del plano neutro existe una presión inferior a la presión atmosférica, lo cual origina una entrada de aire fresco que alimenta la combustión. Cabe mencionar que el plano neutro se encuentra a la misma presión atmosférica (101325 Pa).

Figura 6. **Comportamiento del fuego en espacios cerrados**



Fuente: *Modeling the backdraft*. http://cfbt-us.com/wordpress/?tag=nist&paged=2. Consulta: 11 de octubre de 2017.

En el desarrollo de incendios en espacios cerrados si la cantidad de combustible es más limitada que la cantidad de comburente, el incendio se encuentra controlado por combustible. Si la cantidad de comburente es más limitada que la cantidad de combustible, el incendio se encuentra controlado por ventilación. De igual manera, la mayor parte de las veces los incendios son sofocados con agua, por lo que es necesario tomar en cuenta cómo puede afectar el cambio de la densidad del agua y la presión respecto a la temperatura del lugar.

1.2.7. Fenómenos físico-químicos en incendios

- Combustión súbita generalizada (*flash over*): según la norma ISO 8421-8, un *flash over* es una transición rápida al estado donde todas las superficies de los materiales contenidos en un compartimento se ven involucrados en un incendio. Según la norma NFPA 921, 2014.3.3.83 es la fase transitoria en el desarrollo de un incendio de interior en el que las superficies expuestas a la radiación térmica alcanzan su temperatura de

inflamación de una manera casi simultánea y el incendio se extiende rápidamente por todo el espacio disponible generalizando el incendio en el recinto. Dicho de otra manera, un *flash over* es un fenómeno que se observa solamente en incendios estructurales con ambientes cerrados, en el cual, de manera repentina, todos los elementos combustibles que existan en el lugar comienzan a arder sin necesidad de contacto físico de una llama.

Es el efecto del fuego que más muertes ha provocado en bomberos según las estadísticas de la NFPA. No existe una temperatura asociada exactamente con este comportamiento del fuego, pero suele darse en un rango de 483 °C y 649 °C y el calor liberado por una habitación promedio en el punto exacto en que sucede un *flash over*, puede ser superior a 10 000 Kw. Este rango es pertinente con la temperatura de auto-inflamación del monóxido de carbono (CO), que es uno de los gases más comúnmente vistos en un proceso de pirólisis.

- Explosión de gases por flujo reverso (*back draft/ back draught*): según la norma NFPA 921, 2014.3.3.16 el *back draft* es una deflagración como consecuencia de un aporte de aire repentino a un incendio en un espacio confinado en el que existen productos incompletos de combustión por falta de oxígeno. Dicho de otra manera, un back draft es cuando un lugar en combustión necesita oxígeno, por lo cual la combustión cesa, pero el ambiente sigue a una temperatura suficientemente alta para que, al tener el volumen una apertura mínima, pueda ingresar oxígeno y liberar presión. Lo que sucedería a continuación es que el combustible reinicie de nuevo un proceso de combustión de una manera tan rápida que no sería posible que se enfríen los gases expulsados al liberar la presión del lugar, lo cual generaría no solo una combustión rápida de los gases

internos y expulsados del lugar, también violenta. Un *back draft* solamente es posible en un volumen cerrado.

- Rebosamiento por ebullición (*boil over*): la definición de NFPA 30 dice que un *boil over* es un evento en el incendio de determinados aceites en un recipiente abierto cuando, después de un largo periodo de inflamación, se produce un aumento repentino de la intensidad de fuego asociado con la expulsión del aceite inflamado fuera del recipiente. En otras palabras, un *boil over* es un rebosamiento de un líquido combustible incendiado, cuya densidad es inferior a la densidad del agua y el punto de ebullición es superior al punto de ebullición del agua. Esto significa que el agua hervirá antes que el líquido y no tendrá la capacidad de absorber el calor de él, por lo cual el agua servirá como un expulsor del líquido inflamable aun en combustión llevando a este a ocupar un mayor volumen con la misma cantidad de combustible (aumentando así la proporción de comburente respecto al combustible) dando un efecto de brisa. En estos casos, el tiempo es inversamente proporcional a la temperatura máxima generada por la combustión.

- Rebosamiento superficial (*slop over*): este comportamiento del fuego es similar al *boil over*, solo que con menor rapidez y violencia. Este es común en combate de incendios cuando se aplica agua a una estructura y existe algún líquido viscoso en el lugar.

- Llamas por el techo (*flame over / roll over / dancing angels*): este comportamiento del fuego es común en incendios estructurales y sucede como un activador del *flash over*, disminuyendo el tiempo en el que el *flash over* puede suceder. En este caso las capas de gases acumulados bajo el techo se inflaman de manera que las llamas literalmente se ven

corriendo por el techo, lo cual aumenta significativamente la radiación del lugar. Este fenómeno si puede existir si el ambiente no es lo suficientemente cerrado o no se ha alcanzado una temperatura sumamente alta, por lo cual si puede existir sin necesidad de que posteriormente exista un *flash over*.

- Expansión explosiva del vapor de un líquido en ebullición (*bleve*): b*oiling liquid expanding vapour explosion*. Este comportamiento del fuego existe únicamente en tanques presurizados que almacenan gases licuados cuando son sobrecalentados. En este caso, ya sea por ruptura del tanque o por liberación de la válvula de seguridad, el gas interior sale a una gran velocidad y se incorpora a la combustión exterior sin necesidad de tener contacto directo del fuego en la superficie del tanque. Normalmente, la velocidad de salida es tan alta que no existe llama a una distancia del tanque proporcional a la velocidad. La causa más frecuente de estas explosiones es debido a un incendio externo que implica una manera de transmitir el calor a una parte del tanque.

1.2.8. Propiedades del agua en incendios

El agua tiene capacidad de extinguir el fuego absorbiendo el calor, diluyendo el oxígeno o expandiéndose. Cuando el agua absorbe el calor y llega a su temperatura de ebullición, se convierte en vapor de agua el cual es invisible. El vapor se comienza a observar cuando se enfría, y este se denomina vapor condensado. Cuando el agua se encuentra en su estado gaseoso ocupa un mayor volumen, siendo aproximadamente 1 700 veces más (a 100 °C) que en estado líquido a temperatura ambiente. Esto puede ayudarnos (dependiendo el caso) a presurizar el recinto y obligar a los gases calientes a buscar una salida y acondicionar el lugar, ya que este cambio no es gradual, sino rápido.

El agua a una temperatura mayor es capaz de disolver una menor cantidad de oxígeno, por lo cual una descarga de agua caliente puede significar un combate ineficiente en algunos casos (en otros puede que no se note a simple vista el cambio). Entre más baja sea la temperatura del agua, es capaz de absorber más oxígeno y de bajar la temperatura promedio del lugar al ser esparcida.

La mayor densidad del agua se tiene cuando esta se encuentra a 4 °C. Si la temperatura sube de allí, el agua se dilata. Si la temperatura baja, el agua se contrae. El agua puede bajar drásticamente la temperatura de un incendio si se aplica en las proporciones correctas, sino solamente podría aumentarnos la presión del lugar. En la actualidad, la mayor parte de los chorros de bomberos pueden lanzar agua pulverizada en un volumen amplio, por lo cual se estaría utilizando una cantidad de agua muy baja, pero lo suficientemente alta para sofocar el fuego. Según los ingenieros suecos Krister Gieselsson y Mats Rosander se necesitan aproximadamente 2 millones de gotas por metro cubico de llama para lograr su extinción. Esta teoría funciona partiendo de los 0,3mm de radio de la gota.

- Estequiometría

Estequiometría significa en la medida correcta. La base de esta es que el número de átomos no varía en la combustión. En incendios, es el cálculo cuantitativo basado en teoría atómica entre el combustible y el comburente durante la reacción de combustión. La estequiometría es el factor dominante en este proceso. En una llama premezclada, según la proporción de combustible y comburente, básicamente se puede determinar límites de inflamabilidad, velocidad de la llama, estabilidad de la llama, rango de temperaturas, puntos de ignición y formación de contaminantes en una combustión incompleta.

En un sistema premezclado, la estequiometría no varía dentro de la llama, siendo las proporciones de C, O y H las mismas en todo el campo de esta. En sistemas no premezclados varia en absolutamente todo el campo. La estequiometría es dominante en la combustión ya que de esta depende la forma en que estas diferencias en composición en las llamas no premezcladas se van eliminando.

El concepto de fracción de mezcla ayuda a pensar separadamente sobre procesos independientes de reacciones químicas y mezclado, sirve en la mayor cantidad de casos en sistemas no premezclados.

En casos en los que la temperatura es menor a 1 000K, solo se encuentran especies estables, idealmente para mezclas pobres los productos de reacción serán CO_2, H_2O, O_2 y N_2 en caso de ser combustión completa. Si fuera combustión incompleta podríamos encontrar también CO, H_2 y combustible sin quemar, aunque en el seno de la llama, que es el lugar de alta temperatura podríamos encontrar muchas especies, por ejemplo NO.

Si se tienen mezclas ricas, se pueden encontrar concentraciones más altas de CO, H_2, carbón libre (hollín) y combustible sin quemar. También, en algunos casos se puede encontrar compuestos de azufre y cenizas. En este caso, en la parte de mayor temperatura, que de igual manera sería el seno de la llama, se puede encontrar radicales libres (OH, O, N, H), especies activas(CH_2O) y especies intermedias, como C_2H_2, que es muy importante en la formación de hollín. Al requerirse una temperatura muy superior, muchos de estos elementos solamente se encontrarán en concentraciones apreciables, cerca de la llama.

- Termoquímica

Es una rama de la ciencia que asocia el intercambio de calor de un material a una reacción química (conversión de energía química a energía calorífica o viceversa).

El intercambio de calor es indefinido, ya que consta de más de dos incógnitas en una sola ecuación. En este caso si se necesita tener un valor definido, se debe dejar solamente con 2 incógnitas, volviendo los demás valores constantes. Por este motivo, las mediciones se realizan a presión o volumen constante, obteniendo el resultado solamente por el estado inicial o final del sistema.

A. L. Lavoisier y P.S. Laplace enunciaron una ley que establece que la cantidad de calor que debe suministrarse a un compuesto para descomponerlo en sus partes es igual a la involucrada cuando se forma dicho compuesto a partir de sus elementos. Esto quiere decir que el intercambio de calor de una reacción química en una dirección determinada es igual en magnitud, pero con sentido contrario al calor asociado con la misma reacción.

1.3. Necesidades y restricciones para el desarrollo del sistema

1.3.1. Necesidades del sistema

En bomberos es necesario desarrollar un sistema que pueda alertar de cualquier cambio drástico en algunas variables si se encuentran dentro de una estructura en llamas, como sería el caso de la temperatura, gases y humedad, ya que al tener el equipo de protección personal completo los cambios no son sensibles de una manera sencilla. Si a esto se agrega el estrés al que se

someten los bomberos en el momento que se sofoca un incendio es casi insensible el cambio. Sino se puede determinar de una manera sencilla el cambio que sufre la atmósfera interna a una habitación, los bomberos que se encuentran dentro corren un gran peligro al exponerse a un *flash over* o a un *back draft*.

Al igual que la medición de riesgos respecto a la temperatura y gases, es necesario alertar de diferentes maneras al bombero que tenga el sistema, ya que solamente una alerta visual puede no ser suficiente si el bombero tiene su atención en otro punto. Estas mismas alertas en caso de análisis de concurrencias y estudios posteriores de los datos analizados, podrían utilizarse para otros tipos de desarrollos útiles en bomberos, por lo cual el sistema debe quedar abierto a poder acoplarse fácilmente a desarrollos posteriores.

Las necesidades más significativas del sistema son:

- Medición de temperatura.
- Medición de humedad relativa.
- Medición de gases específicos.
- Visualización de datos en tiempo real.
- Duración superior a una hora (tiempo aproximado para recarga de cilindro con aire).
- Alertas de varios tipos (visual, audible y táctil).
- Datos guardados para análisis.
- Acople con otros desarrollos.

1.3.2. Restricciones del sistema

Según la norma NFPA 1971 que establece los requerimientos de la ropa protectora contra incendios estructurales, el bombero debe tener protección en

todo el cuerpo, sin dejar un solo espacio sin protección, ya sea para protección contra el fuego, agua, cortes, etc. Esto incluye botas, pantalón, tirantes, chaqueta, guantes, escafandra, casco y equipo de protección respiratoria SCBA (*self contained breathing apparatus*). El peso aproximado del equipo de protección en bomberos es el siguiente:

- Casco: 1,72 kg
- Escafandra (Hood): 0,095 kg
- Mascara de SCBA: 1,23 kg
- Radio: 0,55 kg
- Linterna: 0,36 kg
- Chaqueta: 2,38 kg
- Cilindro SCBA: 11,88 kg
- Arnés: 1,18 kg
- Guantes: 0,36 kg
- Pantalón: 1,915 kg
- Botas: 2,76 kg
- Total: 24,43 kg

Tomando en cuenta que un bombero no entra a un incendio solamente con su traje de protección sino también con herramientas de trabajo, se pueden añadir muchas herramientas que debe llevar, por ejemplo cordinos (*paracords*), cintas tubulares (*webbings*), mosquetones (*snaplings*), linterna secundaria, alarma de movimiento (*pass*), llave de acople de mangueras (*res-q-rench*), navaja multiusos para bomberos(*rescue-tool*) y diferentes tipos de guantes (guantes para incendios, guantes para extricación, guantes para manejo de cuerdas). Adicional a esto, puede llevar un pitón y manguera, o una herramienta de mano ya sea un hacha, *halligan tool, kelly tool, denver tool*, entre otros. Todo esto significa un aumento drástico del peso y de igual manera en el rendimiento.

Todas las herramientas mencionadas anteriormente sirven tanto para entradas forzadas, técnicas de equipos de intervención de rescate (RIT), ventilación o extinción de incendios, y todas estas tareas requieren las 2 manos para trabajar de una manera segura, por lo cual no se puede depender del uso de ellas para manejar todo el tiempo una herramienta nueva. De igual manera, no se puede depender de estar visualizando la información en una zona incomoda o que requiera mucha atención, ya que puede ocasionar problemas en cuanto a la seguridad personal al no observar algún tipo de riesgo cercano. Con esto se puede decir que las restricciones más significativas del sistema son las siguientes:

- No interferir con la seguridad personal
- No aumentar significativamente el peso
- Sistema fácil de utilizar y visualizar
- Sistema confiable
- Resistente al agua y a altas temperaturas

1.3.3. Desarrollo del sistema

Observando las necesidades y restricciones del sistema, se puede tomar en cuenta que una alarma de movimiento (*pass*), actualmente se utiliza en bomberos solamente para activar una alarma sonora en caso el bombero que la porta se quede sin moverse, o dado el caso inconsciente. Esta alarma tiene la característica de ser muy grande para lo que en realidad hace. Las dimensiones promedio de un Pass son de 2,5” de ancho, 3,5” de alto y 2” de profundidad, pesando aproximadamente 8 a 9 onzas.

Si se elimina el *pass* del equipo de seguridad de un bombero y se convierte en un *smart-pass* no se afectaría en una gran proporción el peso ni se

agregaría un elemento adicional al equipo, teniendo siempre las funciones de un *pass* convencional (alarma de movimiento) y las funciones necesarias para mediciones ambientales.

Inhalar humo es dañino para la salud, por lo cual es obligatorio para un bombero llevar siempre un sistema de protección respiratoria SCBA (*self contained breathing apparatus*) que contiene aire limpio a presión. Esto indica que un bombero debe colocarse una máscara protectora que abarca todo el rostro tanto para proteger del humo como de la radiación ambiental. Usualmente en un incendio, donde existe una mayor densidad de humo no existe una gran densidad de radiación y viceversa, por lo cual es obligado tener colocada siempre la máscara tanto para proteger de cualquier peligro. De acuerdo a lo anterior, un lugar ideal para visualizar los datos es en la misma máscara en un lugar donde no distraiga, ya que esto elimina el uso de las extremidades para visualizar datos y las deja exclusivamente para operar las herramientas de trabajo.

2. HARDWARE

2.1. Dispositivos de hardware utilizados

2.1.1. Microprocesador

Un microprocesador es un circuito integrado, normalmente el circuito más complejo de una tarjeta electrónica. Es el circuito por el que se realizan todas las operaciones aritméticas y lógicas en modo de instrucciones programadas en lenguaje de bajo nivel.

Microprocesador utilizado: broadcom BCM2835, arquitectura ARM (*advanced RISC machine*), velocidad de 1GHz con un único núcleo.

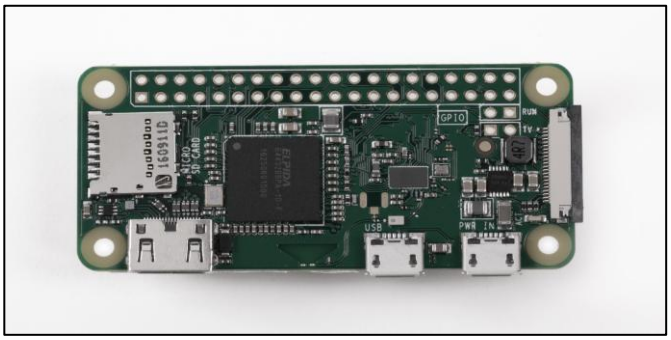
La tarjeta que forma la estructura necesaria para que el microprocesador funcione se denomina Raspberry Pi Zero W. Cuenta con 512 MB de memoria RAM, puerto mini HDMI 1080p/60p, 2 puertos micro-USB (uno de alimentación y uno de datos), puerto para tarjetas micro-SD, 40 pines GPIO, Video composite, conector de cámara CSI, soporte para wifi 802.11n y bluetooth 4.0.

La tarjeta tiene dimensiones de 65,0 mm X 30,0 mm y un costo aproximado de $ 11,00, lo cual la hace completamente accesible al desarrollo del prototipo al ajustarse a las necesidades.

El sistema operativo se carga sobre una memoria micro SD, en este caso cuenta con un sistema libre GNU/Linux denominado Raspbian-Jessie, basado principalmente en Debian-Jessie.

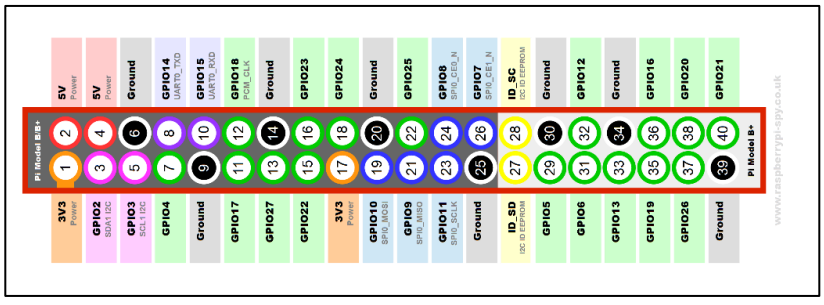
Una de las cosas más esenciales de este sistema es que cuenta con soporte optimizado para cálculos en coma flotante por hardware. Cuenta también con un entorno LXDE como escritorio y contiene ya instaladas varias librerías y herramientas de desarrollo para Python.

Figura 7. **Microprocesador**



Fuente: *Raspberry PI zero W*. https://www.raspberrypi.org/magpi/wp-content/uploads/2017/02/PI-Zero-W-web.jpg. Consulta: 11 de octubre de 2017.

Figura 8. **Raspberry Pi B+ GPIO**



Fuente: *Raspberry Pi B+ GPIO*. https://www.raspberrypi-spy.co.uk/2014/07/raspberry-pi-b-gpio-header-details-and-pinout/. Consulta: 11 de octubre de 2017.

2.1.2. Microcontrolador

Un microcontrolador es un dispositivo de hardware que permite llevar a cabo de manera programada una secuencia lógica de combinaciones para interactuar con otros dispositivos. Los controladores son la base de los sistemas embedidos y están diseñados de tal manera que contengan todos los componentes necesarios para funcionar dentro del mismo chip.

Las principales características que se han tomado en cuenta son las siguientes:

- Cantidad de periféricos a conectar.
- Entradas y salidas digitales de propósito general GPIO (*general purpose* Input/Output).
- Entradas analógicas (ADC).
- Protocolos de comunicación (UART, I^2C, SPI).
- Memoria (EEPROM, flash, entre otros).

El microcontrolador utilizado en el prototipo de implementación es ATMEL AVR MEGA 32U4 (*arduino pro micro*).

2.1.2.1. ATMEL AVR MEGA 32U4

Este microcontrolador es muy simple, pero cuenta con las capacidades necesarias para el desarrollo actual, lo cual de igual manera hace que sea fácil de utilizar en una versión de implementación. Este microcontrolador en una tarjeta denominada arduino pro micro viene con una entrada micro USB directamente para poderlo programar o alimentar, al igual que con un oscilador

de 16 MHz, leds de Tx y Rx y múltiples capacitores de tantalio, resistencias de protección y un LDO (*low-dropout regulator*).

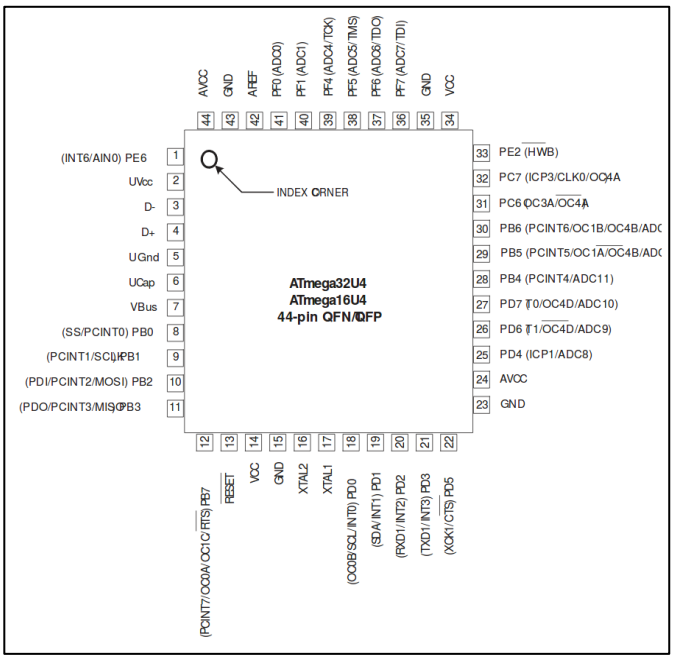
Características principales:

- Microcontrolador de alto rendimiento (*high performance*) de 8 bits
- 135 instrucciones, ciclo de ejecución de reloj individual
- 32X8 registros de propósito general
- Arriba de 16 MIPS en un rendimiento a 16 MHz
- On Chip 2-cycle Mult
- Memoria
    - 32 Kb de memoria flash programable
    - 2,5 Kb de memoria SRAM
    - 1 Kb de memoria EEPROM
    - Escrituras aproximadas: 10 000 Flash/100 000 EEPROM
- Interfaz JTAG (*joint test action group* – IEEE 1149,1)
- USB Rev2,0
- Características de periféricos
    - On-chip PLL, HS timer – operación de 32 hasta 96 MHz.
    - Temporizador / contador de 8 bits con prescaler separado y modo comparador.
    - Dos temporizadores / contadores de 16 bits con prescaler separado y modo *compare-and-capture*.
    - Temporizador / contador de 10 bits con PLL (64 MHz) y modo comparador.
    - 4 canales PWM de 8 bits.

- 4 canales PWM de resolución programable de 2 a 16 bits.
- 6 canales PWM de alta velocidad, con resolución programable de 2 a 11 bits.
- Modulador de comparación de salida.
- 12 canales ADC de 10 bits.
- UART/USART programable con control de flujo por hardware.
- *Interfaz serial master/slave SPI*.
- *Interfaz serial 2-wire byte oriented*.
- Watchdog timer programable *on-chip*.
- Comparador Análogo *on-chip*.
- Sensor de temperatura *on-chip*.
- *Power-on-reset – brown-on-reset programmable*.

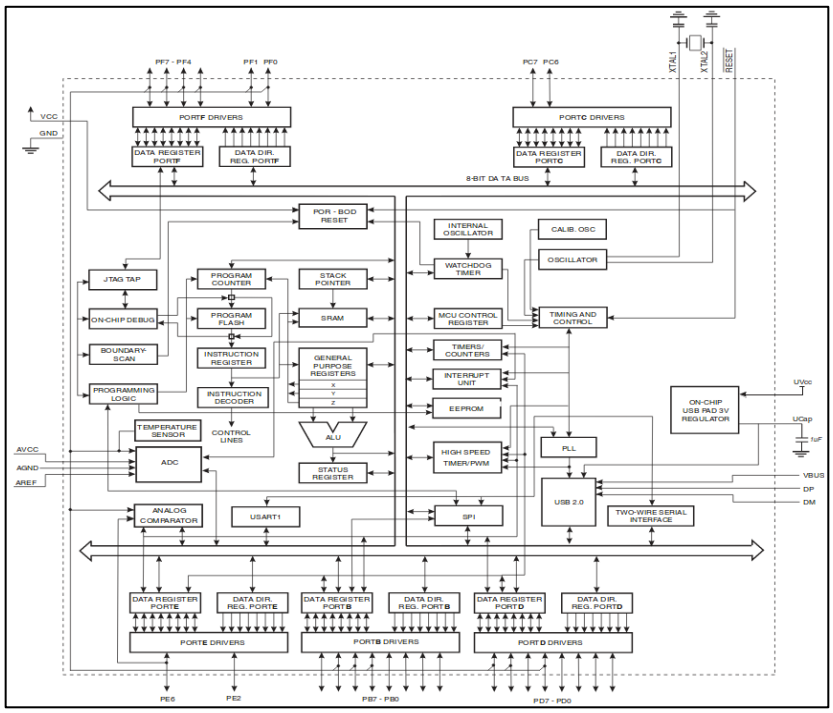
- Voltaje de operación: 2,7 – 3,5V
- Oscilador interno calibrado a 8 MHz
- Temperatura de operación: -40 °C a +85 °C
- Frecuencias máximas: 8 MHz a 2,7 V, 16 MHz a 4,5 V

Figura 9. **Configuración de pines Microcontrolador MEGA32U4**



Fuente: *Configuración de pines Microcontrolador MEGA32U4*. http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf. Consulta: 11 de octubre de 2017.

Figura 10. **Diagrama de bloques Microcontrolador ATMEGA 32U4**

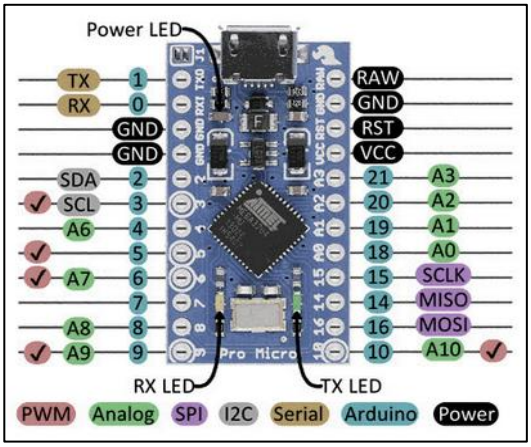


Fuente: *Diagrama de bloques Microcontrolador ATMEGA 32U4*. http://www.atmel.com/ Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf. Consulta: 11 de octubre de 2017.

- Tarjeta arduino pro micro

Esta tarjeta de desarrollo tiene el controlador MEGA32U4 mencionado anteriormente, solamente que no posee todos los pines del microcontrolador como entradas o salidas de la tarjeta. La tarjeta cuenta con 9 canales ADC de 10 bits, 5 salidas PWM programables, 12 entradas-salidas de propósito general, puertos seriales (UART, I^2C, SPI). funciona a 16 MHz – 5 V.

Figura 11. **Tarjeta de desarrollo Arduino Pro Micro**



Fuente: *Tarjeta de desarrollo Arduino Pro Micro*. https://learn.sparkfun.com/tutorials/pro-micro-fio-v3-hookup-guide. Consulta: 11 de octubre de 2017.

2.1.3. Sensores

2.1.3.1. Sensor de humedad y temperatura DHT22

El tiempo de respuesta del sensor es menor de 10 segundos leyendo datos de temperatura, es decir, tarda menos de 10 segundos en mostrar un cambio de temperatura real en el ambiente. En cambio, en cuanto a humedad relativa, el tiempo de respuesta es menor a 5 segundos.

Características:

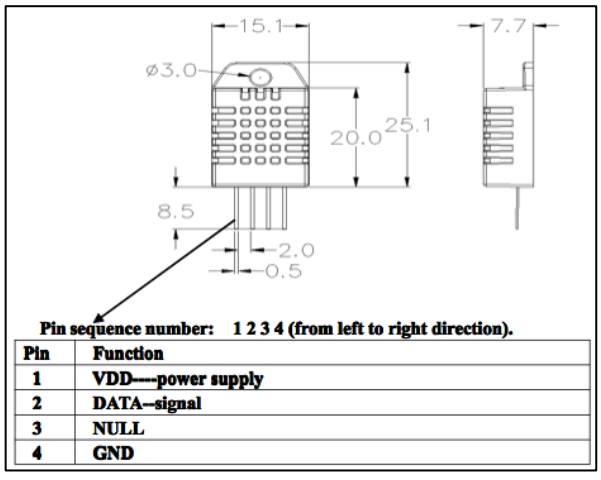
- VCC de 3,3V a 5,0 V
- Señal en bus serial single-bus wire
- Elemento de detección: capacitor de polímero y termistor
- Rango de operación: 0,0 % – 99,9 % HR. ~ -40 °C – 80 °C

- Exactitud: ±2 % HR ~ <±0,5 °C
- Resolución: 0,1 % HR ~ 0,1 °C
- Repetibilidad: ±1 %RH ~ ±0,2 °C
- Histéresis de humedad: ±0,3 % HR
- Estabilidad a largo plazo: ±0,5 % por año
- Periodo de muestreo: 2s
- Dimensiones: 14 mm * 18 mm * 5,5 mm

Este sensor es útil, siempre que no se necesite una medición constante ni milimétrica. Si se desea conectar múltiples sensores DHT22 a un solo dispositivo, cada sensor debe tener su pin de datos, ya que el sensor no tiene capacidad para poder leer señales de entrada.

Este sensor a pesar que utiliza un solo cable para enviar datos no es compatible con el protocolo Dallas one-wire.

Figura 12. **Sensor DHT22**



Pin sequence number: 1 2 3 4 (from left to right direction).

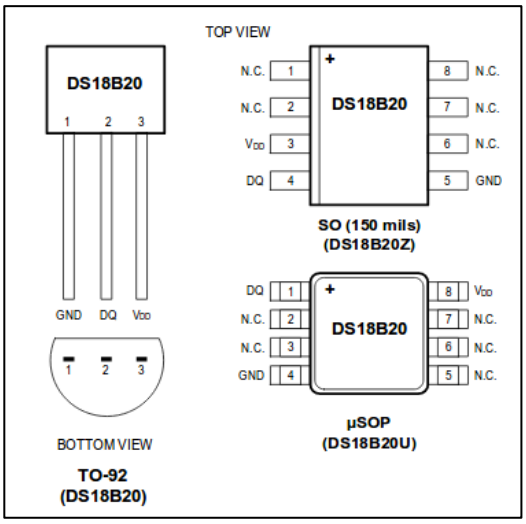
Pin	Function
1	VDD-----power supply
2	DATA--signal
3	NULL
4	GND

Fuente: *Sensor DHT22*. https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf. Consulta: 11 de octubre de 2017.

2.1.3.2. Sensor de temperatura DS18B20

Sensor digital, utiliza un protocolo de comunicación de un solo pin de datos llamado Dallas one-wire. Este protocolo permite enviar y recibir datos por el mismo pin de datos a diferencia de la mayoría de protocolos de comunicación, que necesitan como mínimo 2 cables.

Figura 13. **Empaquetados sensor DS18B20**



Fuente: *Empaquetados sensor DS18B20*. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf. Consulta: 11 de octubre de 2017.

- Aplicaciones principales

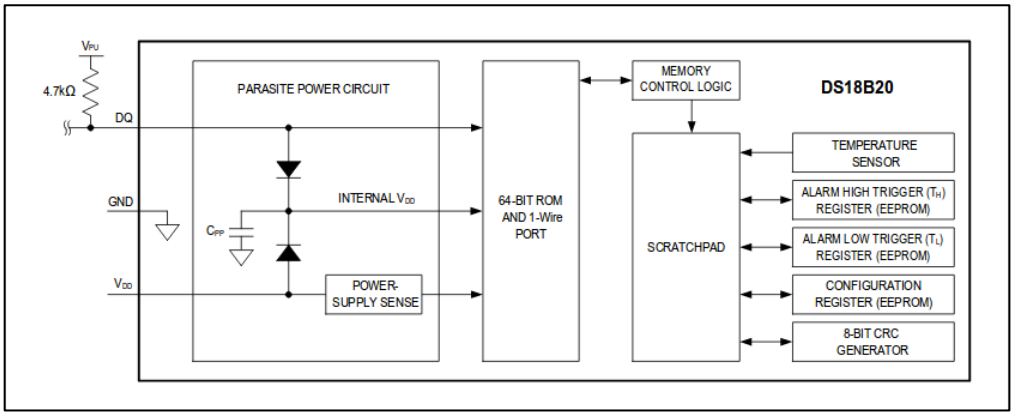
    - Controles termométricos
    - Sistemas industriales
    - Productos de consumo
    - Termómetros digitales
    - Sistemas sensibles a temperatura

- Características

  - VCC de 3,0 V a 5,5 V.
  - Señal de un solo pin Dallas one-wire, 64 bits on-board ROM.
  - Rango de operación: -55 °C a +125 °C ~ -67 °F a +257 °F.
  - Exactitud: ±0,5 °C entre -10 °C y +85 °C.
  - Resolución: ±0,2 °C.
  - Incerteza: ±0,5 °C entre -10 °C y +85 °C ~ ±2 °C entre -55 °C y 125 °C.
  - Resolución: 9 bits a 12 bits de mediciones de temperatura.
  - Función de alarma programable conn base en *trigger-points* NV (*user-definable nonvolatile*).
  - Corriente de *standby*: 750 nA. – 1 000 nA.
  - Corriente en estado activo: 1 mA. – 1,5 mA.

Este sensor no requiere componentes externos para operar.

Figura 14. **Diagrama de bloques interno sensor DS18B20**



Fuente: *Diagrama de bloques interno sensor DS18B20*. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf. Consulta: 11 de octubre de 2017.

2.1.3.3. Sensor de temperatura MLX90614

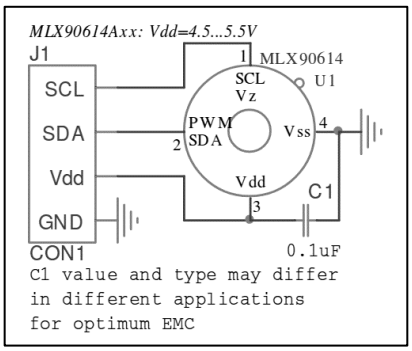
Es un chip de silicio con una micromembrana mecanizada sensible a la radiación infrarroja de un objeto distante. El proceso de medición parte de amplificar y digitalizar una señal proveniente de la membrana y calcular la temperatura del objeto analizado. La temperatura es linealizada y compensada contra las variaciones de temperatura ambiente.

Este sensor incorpora amplificadores avanzados de bajo ruido, un convertidor análogo-digital (A/D) de 17 bits y un procesador digital de señales (DSP) dentro del mismo, lo cual permite un rango de medición de temperaturas de -40 °C a 125 °C y un rango de temperaturas ampliado de -70 °C a 380 °C que mantiene una resolución de 0,02 °C.

Características:

- VCC 3,0 V o 5,0 V
- Calibración de fábrica de:
    - -40 °C a 125 °C de sensor de temperatura
    - -70 °C a 380 °C de temperatura de objeto remoto
- Exactitud: 0,5 °C. Calibración con estándar médico
- Resolución: 0,02 °C
- Interfaz digital compatible SMBus
- Salida PWM programable para lectura contínua
- Modo ahorro de energía incorporado

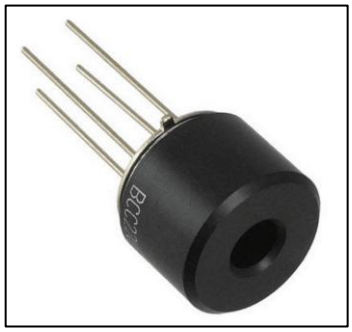
Figura 15. **Conexión con SMBus sensor MLX90614**



Fuente: *Conexión con SMBus sensor MLX90614*. https://www.pololu.com/file/download/MLX90614.pdf?file_id=0J170. Consulta: 11 de octubre de 2017

El sensor se incorpora en una placa de circuito impreso con código GY-906.

Figura 16. **Sensor MLX 90614**



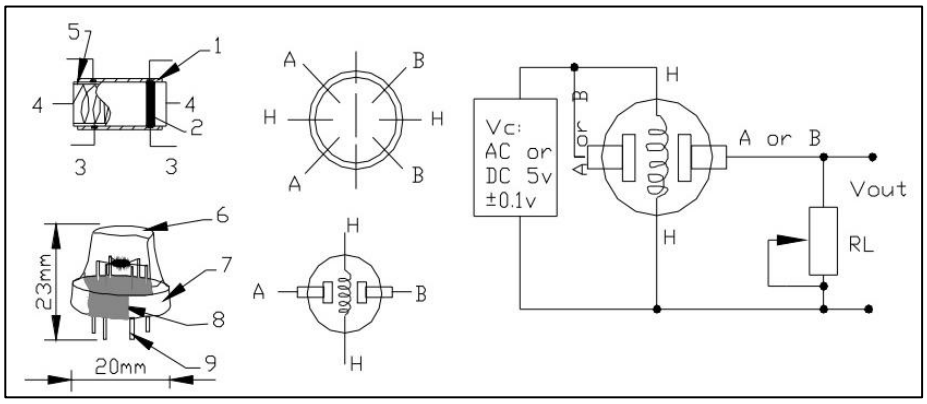
Fuente: *Sensor MLX 90614*. https://www.digikey.com/product-detail/en/melexis-technologies-nv/MLX90614KSF-ACC-000-TU/MLX90614KSF-ACC-000-TU-ND/3641023. Consulta: 4 de noviembre de 2017.

2.1.3.4. Sensores de gases MQ

Los sensores de gases MQ son sensores análogos, fáciles de implementar con un conversor análogo-digital (A/D). Son sensores electroquímicos y varían su resistencia interna cuando son expuestos a cierta gamma de gases. Internamente estos sensores poseen un calentador que sirve solamente para aumentar la temperatura interna y que la reacción de los gases del entorno con la resistencia sea completa.

El calentador necesita entre 2 o 5 voltios, dependiendo el sensor MQ que se utilice. Dicho sensor se comporta como una resistencia y necesita una resistencia de carga (RL) para cerrar el circuito, de esta forma se crea un divisor de tensión el cual es la salida del sensor.

Figura 17. **Circuito de sensores de gases**



Fuente: *Circuito de sensores de gases*. https://playground.arduino.cc/Main/MQGasSensors. Consulta: 4 de noviembre de 2017.

Debido a que la forma ideal de operación de este sensor es con el calentador estable, se debe esperar un tiempo estimado antes de que la salida de medición de igual manera sea estable y cumpla con las características de la hoja de datos.

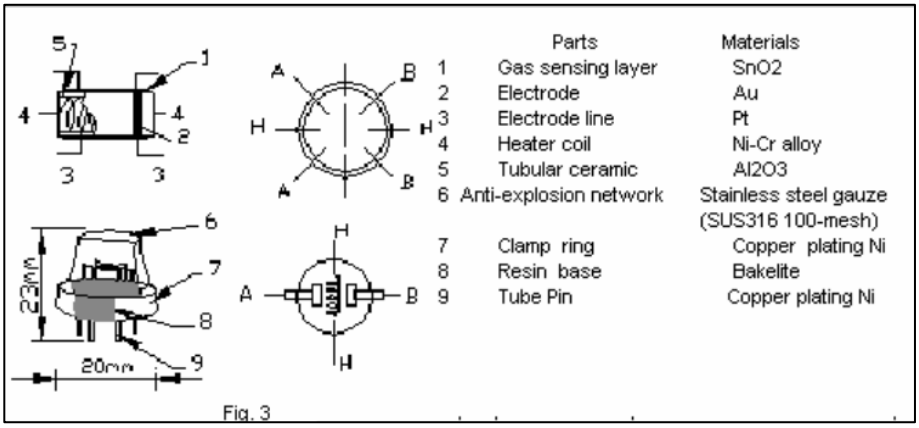
2.1.3.4.1. Sensor de gas MQ-2

Sensor adecuado para detectar GLP, propano, metano, alcohol, hidrógeno y humo, siendo más sensibles el GLP y propano.

Características:

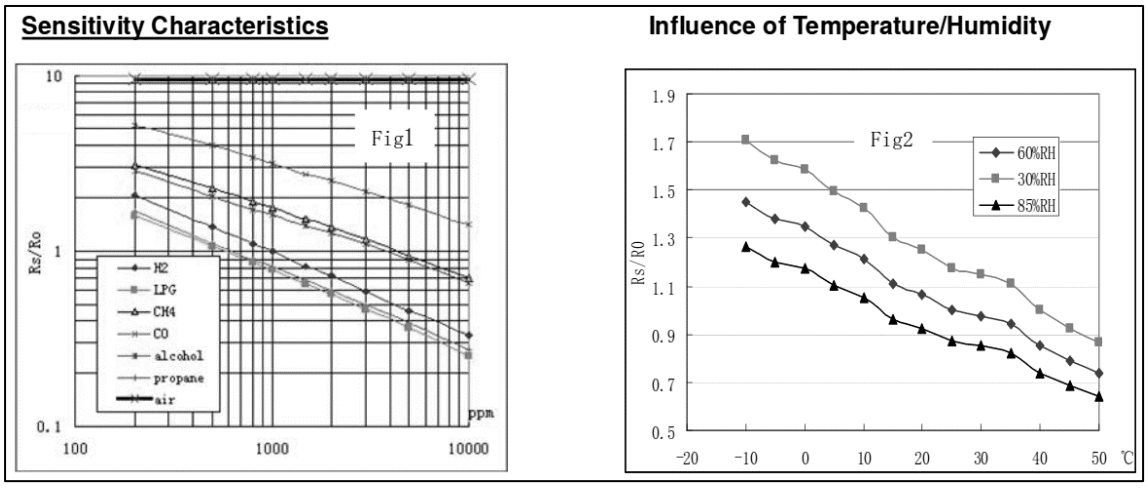
- Sensor semiconductor
- Principales gases: gas combustible y humo
- Concentración: 300 – 10 000 ppm
- Voltaje de circuito (Vc): <24 V
- Voltaje de calentador (Vh): 5,0 V ± 0,2 V AC o DC
- Voltaje de resistencia de carga (RL): ajustable
- Potencia del calentador: <900 mW
- Resistencia de lectura: 2kO hms – 20KO hms (2 000 ppm C_3H_8)

Figura 18. **Estructura sensor MQ-2**



Fuente: *Estructura sensor MQ-2*. https://www.pololu.com/file/0J309/MQ2.pdf. Consulta: 4 de noviembre de 2017.

Figura 19. **Características e influencias del sensor MQ-2**



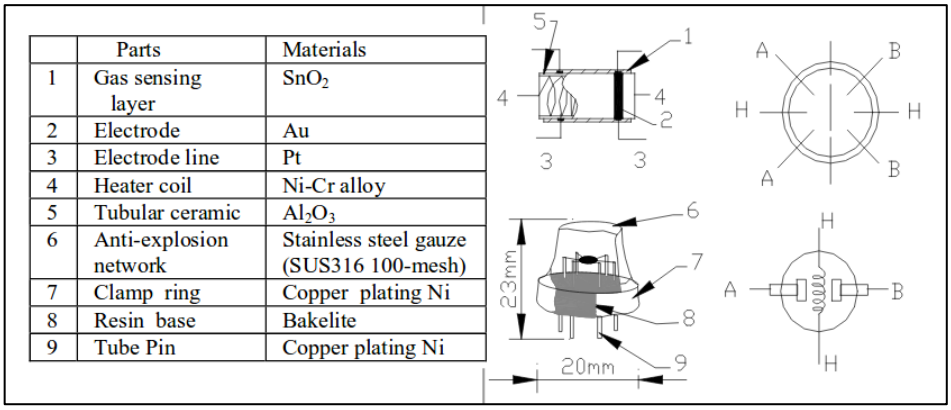
Fuente: *Características e influencias del sensor MQ-2*. https://www.pololu.com/file/0J309/MQ2.pdf. Consulta: 4 de noviembre de 2017.

Sensor adecuado para control de calidad del aire, útiles para detectar NH_3, No_x, alcohol, benceno, humo, CO_2, entre otros. Este sensor es sensible en la misma proporción a los gases mencionados anteriormente, por lo cual se determina si el aire está limpio.

Características:

- Sensor semiconductor
- Principales gases: NH_3, No_x, alcohol, benceno, humo, CO_2
- Concentración: 300 – 10 000 ppm
- Voltaje de circuito (Vc): <24 V
- Voltaje de calentador (Vh): 5,0 V ± 0,1 V AC o DC
- Voltaje de resistencia de carga (RL): ajustable
- Potencia del calentador: <800 mW
- Resistencia de lectura: 30 kOhms – 200 KOhms (100 ppm NH_3)

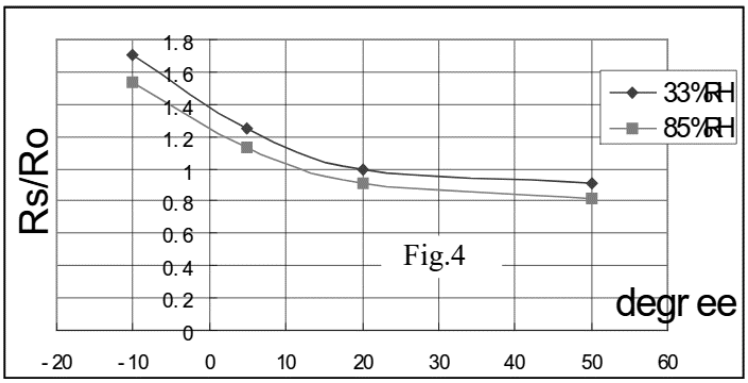
Figura 20. **Estructura sensor MQ-135**



	Parts	Materials
1	Gas sensing layer	SnO_2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al_2O_3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

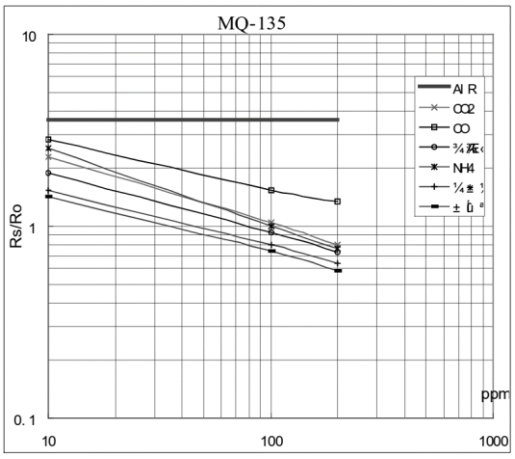
Fuente: *Estructura sensor MQ-135*. https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf. Consulta: 4 de noviembre de 2017.

Figura 21. **Dependencia del sensor MQ-135 respecto a la humedad y temperatura**



Fuente: *Dependencia del sensor MQ-135 respecto a la humedad y temperatura*. https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf. Consulta: 4 de noviembre de 2017.

Figura 22. **Características sensitivas del sensor MQ-135**



Fuente: *Características sensitivas del sensor MQ-135 a una temperatura de 20 °C y una humedad relativa de 65 %. Concentración de O2 a 21 % con RL=20KOhms*. https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf. Consulta: 4 de noviembre de 2047.

2.1.3.5. Sensor de movimiento y aceleración MPU6050

Es un circuito integrado que consta internamente de un acelerómetro un giroscopio y un procesador digital de movimiento (DMP). El acelerómetro es un sensor encargado de medir aceleraciones transformando una magnitud física de aceleración en una magnitud eléctrica. El giroscopio es un sensor que sirve para medir velocidad angular basándose en el mantenimiento del impulso de rotación. El MPU6050 cuenta con una resolución de 16 bits, dividiendo el rango para cada eje x, y z.

Tabla III. **Rangos de escala y el valor máximo raw**

Rango de escala completa giroscopio	Sensibilidad del giroscopio	Rango de escala completa acelerómetro	Sensibilidad del acelerómetro
±250 dps	131	±2g	16 384
±500 dps	65,5	±4g	8 192
±1 000 dps	32,8	±8g	4 096
±2 000 dps	16,4	±16g	2 048

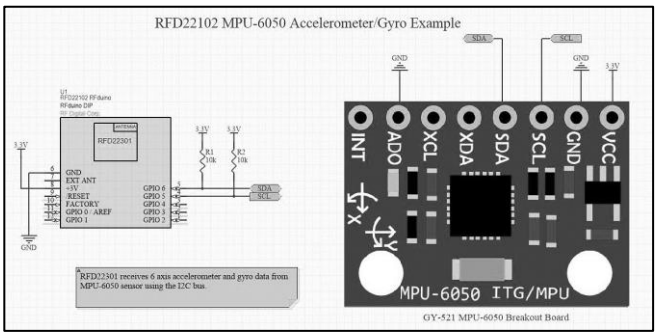
Fuente: elaboración propia.

Características:

- Salida digital de 6 ejes.
- Algoritmos embedidos para calibración.
- Sensor de temperatura digital.
- Entrada digital de video FSYNC.
- Interrupciones programables: *user programmable, free-fall, high-G, zero motion*.

- VCC: 2,37 a 3,46 V.
- Voltaje lógico: 1,8 V ± 5 % o VDD.
- 10 000 g tolerancia de aceleración máxima.
- ADC de 16 bits integrado para muestreo.
- Corriente de operación de acelerómetro: 500 uA.

Figura 23. **Sensor MPU-6050**



Fuente: *Sensor MPU-6050*. http://www.rfduino.com/wp-content/uploads/2014/04/RFduino-MPU-6050-Circuit.jpg. Consulta: 4 de noviembre de 2017.

2.1.4. Dispositivos de alerta y visualización

2.1.4.1. Dispositivos visuales

Ambos equipos visuales son pantallas OLED de un tamaño menor a 1,5”, ambas son necesarias, una para detectar el estado de los dispositivos y la otra para mantener en constante alerta a cambios drásticos. No se muestran todos los datos en una sola pantalla para evitar distracciones innecesarias en todo momento.

Aplicaciones:

- *Smartwatch* o reloj inteligente
- Equipos médicos
- Equipos industriales
- Equipos de audio
- IOT

La diferencia entre ambas pantallas principalmente es el controlador que utilizan, una utiliza controlador SH1106 y la otra SSD1306. El controlador SSD1306 tiene una matriz de memoria RAM de 128*64 bits que se asigna directamente a los 128x64 pixeles de la pantalla, el controlador SH1106 tiene una matriz de RAM de 132*64 bits, por lo que hay 4 bits en cada fila de la matriz que no se utilizan. Por este motivo puede que si se utiliza el mismo código para ambas pantallas no se visualice de la manera correcta.

2.1.4.1.1. Pantalla OLED SH1106

Esta pantalla se encuentra en el dispositivo de la careta, muestra todas las variables y peligros ambientales provenientes de los sensores y puede alertar si se corre algún peligro referente a estas mismas.

Características:

- Interfaz utilizada: SPI
- Controlador: SH1106
- VCC: 3,3 V – 5 V
- Potencia: 0,08 W al 100 %
- Pantalla: OLED de alto contraste monocromática

- Resolución: 128 x 64
- Longitud: 1,3”
- Angulo de visión: >160°
- Temperatura sin fallas: -30 °C - +70 °C

2.1.4.1.2. Pantalla OLED SSD1306

Esta pantalla se encuentra en el dispositivo colocado en el SCBA (*self contained breathing apparatus*), muestra todos los datos de conexión de ambos dispositivos, cadenas de caracteres enviadas de un dispositivo a otro, estado de los dispositivos, entre otros. Esta pantalla sirve directamente para el control de configuraciones y conexiones, la cual puede ser útil en momentos de falla o en el momento que el equipo es colocado antes de ingresar a un incendio para asegurarse que toda conexión está en buen estado. Esta pantalla no muestra datos de emergencia.

Características:

- Interfaz utilizada: I^2C
- Controlador: SSD1306
- VCC: 2,5 V – 5 V
- Potencia: 0,08 W al 100 %
- Pantalla: OLED de alto contraste monocromática
- Resolución: 128 x 64
- Longitud: 0,96”
- Ángulo de visión: > 160°
- Temperatura sin fallas: -30 °C - +70 °C

Figura 24. **Pantalla OLED**

Fuente: *Pantallas OLED*. https://www.xataka.com/moviles/pantallas-oled-en-un-smartphone-lo-que-ganamos-y-perdemos. Consulta: 4 de noviembre de 2017.

2.1.4.1.3. Actuadores luminosos

Los diodos emisores de luz (led) son utilizados en 2 funciones directamente en uno de los dispositivos desarrollados.

El primer uso es solamente como un método de *feedback* a la persona que presiona un botón para asegurarle que el botón fue presionado y una entrada de propósito general tomó correctamente dicha acción.

El segundo uso es cuando se activa la sirena para saber cuál dispositivo está sonando, eso no sirve para emergencias, sino para saber si se activó por error y desactivarlo. Esto sucede muy seguido y muchas veces no se sabe cuál *pass* o SCBA está dando una señal de alarma teniéndolo a unos metros de distancia.

Los leds utilizados son de color rojo y verde, un color para cada pulsador.

Características:

- VCC: 2,4 V
- Intensidad: 1 300 a 1 600 mcd
- Imax: 10 mA
- Ángulo de iluminación: 30° respecto a la horizontal
- Diámetro: 3 mm
- Largo: 5,25 mm

2.1.4.2. Dispositivos táctiles

2.1.4.2.1. Minimotor vibrador

Este motor es utilizado como una manera de alertar por si la persona que utiliza los dispositivos no está pendiente de los demás sistemas de alerta.

Características:

- Tipo: Motor DC
- VCC: 2 – 5,5V
- Imax: 26 mA
- RPM: 9 000 ± 2 000
- Dimensiones: 10 mm de diámetro, 2,7 mm de ancho
- Peso: 0,9 g

Figura 25. **Minimotor vibrador**

Fuente: *Minimotores*. https://www.google.es/search?q=minimotor+vibrador&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiHnZuLzOzZAhULnFkKHeCdDDsQ_AUICigB&biw=1366&bih=662#imgrc=TtOlviAWpRdvbM. Consulta: 4 de noviembre de 2017.

2.1.4.2.2. Pulsadores DPST

Los pulsadores son utilizados para que una persona pueda interactuar directamente con el hardware. Solamente un dispositivo (*pass*) lleva dos pulsadores, el fin de utilizar solo 2 pulsadores y no otro método de interacción es para que el dispositivo sea lo más fácil de utilizar para el usuario. Al igual que los diodos emisores de luz rojo y verde, los pulsadores son del mismo color para diferenciarlos.

- Pulsador verde: sirve para movilizarse dentro de un menú de control.
- Pulsador rojo: sirve para ingresar en la opción del menú seleccionada.
- Pulsador verde y rojo: juntos sirven para activar o desactivar la alarma de emergencia.

Características:

- Tipo: normalmente abierto – *momentary tactile switch*

- Dimensiones: 12 mm * 12 mm * 12 mm
- Diámetro de botón: 11 mm
- Peso: 1,4 g
- Material: plástico

2.1.4.3. Dispositivos audibles

2.1.4.3.1. Buzzer

Este dispositivo denominado buzzer o zumbador, es utilizado como una manera de alertar por si la persona que utiliza los dispositivos no está pendiente de los demás sistemas de alerta.

Características:

- Diámetro: 12 mm
- Alto: 9,5 mm
- Voltaje: 3,5V – 7V DC
- Imax: 25 mA – 32 mA
- Sonido: >85 dB
- Frecuencia resonancia: 2 300 ± 300 Hz
- Rango de temperatura: -20 °C a 45 °C

2.1.4.3.2. Sirena

Este dispositivo es muy importante, ya que actualmente en los bomberos si existe y se conoce como Pass, este sirve para alertar a los bomberos cuando un bombero queda inconsciente dentro de un incendio activándose a los 30 segundos después de que no detecta un movimiento fuerte. La desventaja de

este dispositivo es que se activa muchas veces por error, ya que el bombero que lo lleva puesto lo tiene sobre el pecho o al lado de la cintura, y estos son lugares que muchas veces cuando se está trabajando pueden quedar inmóviles más de 30 segundos. Cuando este se activa, se necesita un movimiento fuerte para silenciarlo normalmente un golpe con la mano y así empiece de nuevo a contar los 30 segundos sin movimiento.

En este caso, el dispositivo de medición va dentro de la careta del SCBA dejando 2 ventajas a favor de este:

- El dispositivo se activará con menor frecuencia por error, ya que la cabeza normalmente tiene un rango de movimiento mayor al pecho o a la cintura de una persona.

- En caso se active por error, bastara con mover la cabeza para desactivar el sonido, no se necesita una extremidad para reiniciar el *pass*.

El motivo por el cual se necesita hacer este dispositivo si ya existe es porque se necesita tener ese dispositivo junto a muchos elementos más de este desarrollo en el mismo espacio que ocupa el diseño actual en bomberos, para que de esta manera se hagan las mismas funciones y muchas más en el mismo volumen y un peso similar al que se tiene en la actualidad; por lo cual, la sirena necesita un circuito oscilador para que esta funcione, y se debe activar en casos de emergencia de manera automática o manual.

2.1.4.3.3. LC9801 – LC9806

Este circuito integrado sirve para diseñar sirenas o alarmas, generando automáticamente un ciclo o secuencia audible en un altavoz.

Características:

- Dispositivo CMOS
- Oscilador RC *on-chip*
- Función de auto apagado
- VCC: -0,3 V a 5 V
- Rango de temperatura soportado: -25 °C a 125 °C
- Temperatura de operación: 0 °C a +70 °C

2.1.5. Dispositivos de comunicación

2.1.5.1. Módulos bluetooth BC417 (HC-05 – HC-06)

Estos módulos se utilizan para comunicarse directamente de la careta al *pass* desarrollado, ya que la distancia entre estos no será mayor de 1m. Así sin importar las condiciones es muy poco probable que la comunicación entre ambos sea inestable.

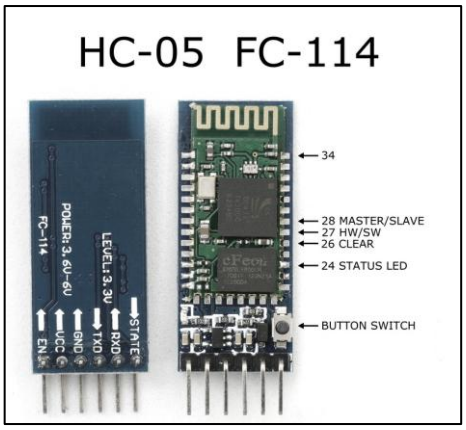
La diferencia entre los módulos HC-05 y HC-06 es principalmente el *firmware* cargado en ellos, el módulo HC-05 puede funcionar como maestro o esclavo, mientras que el HC-06 puede funcionar solamente como esclavo.

Características:

- Compatible con protocolo bluetooth 2,0.
- Vcc: 3,3 VDC a 6 VDC.
- Voltaje de operación: 3,3 VDC.
- Baud Rate: 1 200 bps, 2 400 bps, 4 800 bps, 9 600 bps, 19 200 bps, 38 400 bps, 57 600 bps, 115 200 bps.

- Imax: <40 mA.
- Isleep: <1 mA.
- Protocolo de salida: UART.

Figura 26. **Modulo bluetooth**



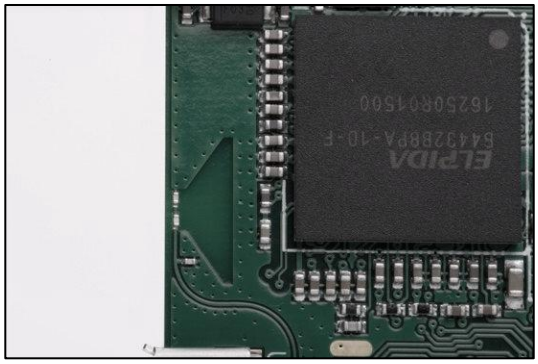
Fuente: *Modulo Bluetooth*. https://wiki.eprolabs.com/index.php?title=File:HC-05_FC-114.jpg. Consulta: 4 de enero de 2018.

2.1.5.2. Módulo wifi IEEE 802,11N

Dado que se está utilizando una Raspberry Pi Zero W, esta trae implementado dentro de la misma tarjeta un módulo wifi y bluetooth; no se utilizó el bluetooth ya que no se podía garantizar una conexión estable.

Las ventajas de utilizar el módulo wifi incluido en la Raspberry Pi Zero W es que no se necesita adaptar más hardware, y en este caso si es útil ya que la conexión a wifi sería solamente en caso de descarga de datos o actualizaciones.

Figura 27. **Antena y capacitores para Wireless dentro de Raspberry Pi Zero W**



Fuente: *Antena y capacitores para Wireless dentro de Raspberry Pi Zero W*. https://www.raspberrypi.org/magpi/pi-zero-w-wireless-antenna-design/. Consulta: 4 de enero de 2018.

2.1.5.3. Módulo USB a UART CP2102

Este módulo sirve para poder entrar a la terminal serial que ofrece el sistema GNU/LINUX Raspian Jessie al configurar en la Raspberry Pi Zero W el puerto Serial como acceso. De igual manera sirve para programar el microcontrolador utilizado por medio de puerto serial de una manera más sencilla.

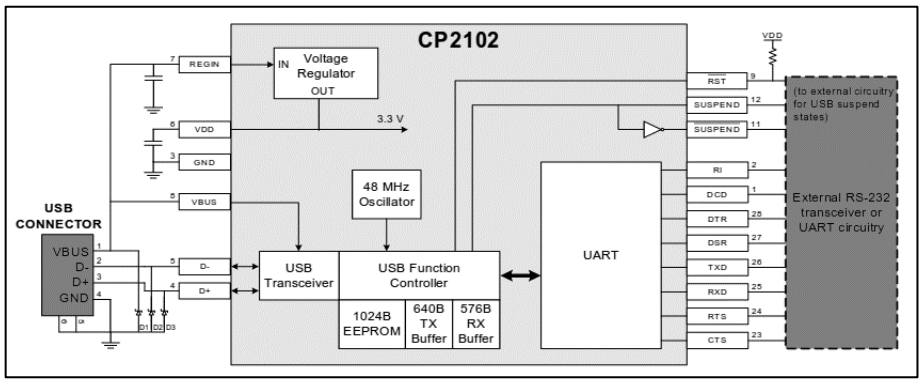
Este módulo sirve para conectar un puerto serial UART a un puerto USB sin necesidad de un MAX232, aunque de igual manera se puede añadir para manejar niveles del protocolo RS232.

Características:

- Transceptor USB integrado, no necesita resistencias externas
- Reloj integrado
- Power *on reset on-chip*
- Regulador de 3,3 V *on-chip*

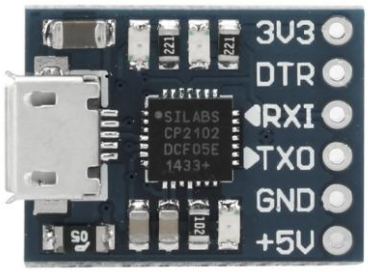
- USB 2,0 (12 Mbps)
- Baud Rate: 300 bps to 1MBits
- 576 bytes en buffer de recepción, 640 bytes en buffer de transmisión
- Vcc UART: 3,0 VDC a 3,6 VDC
- Vcc USB: 4,0 VDC a 5,25 VDC

Figura 28. **Diagrama de bloques CP2102**



Fuente: *Diagrama de bloques CP2102*. https://www.sparkfun.com/datasheets/IC/cp2102.pdf. Consulta: 4 de enero de 2018.

Figura 29. **Módulo CP2102**



Fuente: *Módulo CP2102*. https://www.sparkfun.com/datasheets/IC/cp2102.pdf. Consulta: 4 de enero de 2018.

2.1.6. Alimentación, carga y protección

2.1.6.1. Baterías LiPo

LiPo es la abreviatura de polímero de litio, son baterías recargables utilizadas en aplicaciones donde la demanda de corriente es considerable, requiriendo poco espacio y poco peso, ya que estas tienen una mejor relación entre tamaño y eficiencia.

Aplicaciones:

- Sistemas de radio control
- Sistemas DMR (*digital mobile radio*)
- Teléfonos celulares y dispositivos móviles
- Dispositivos portátiles de medición

Estas baterías sino se les da un uso correcto, pueden incendiarse o explotar, por lo cual requieren un cuidado único y adecuado en los procesos de carga, descarga y almacenamiento.

Las baterías LiPo de igual manera se pueden inflar a causa de la temperatura, la cual puede aumentar con cargas que exijan mucha corriente. Para utilizar estas baterías en el dispositivo, se debe tomar en cuenta la protección de la misma y a la vez, mantenerla a un alcance visual para evitar que la batería se encuentre dañada sin siquiera notarlo.

Las baterías de litio son muy delicadas, un buen uso de la batería puede realizar más de 300 ciclos de carga/descarga, mientras que un mal uso puede

estar muy debajo del 10 % de ese rango. Para manejar las baterías de forma segura es importante seguir ciertos cuidados:

- No dejar desatendidas las baterías mientras se cargan, ni en lugares con materiales combustibles cercanos.

- Utilizar un cargador específico para este tipo de baterías, estas baterías utilizan un cargador balanceador que carga cada celda de manera independiente.

- No cargar las baterías por encima del voltaje indicado.

- Mantenerlas por lo menos con el 40 % de batería para aumentar su vida útil.

- No cargarlas mientras se encuentren muy frías (5 °C o menos), ni muy calientes.

- No usar una batería dañada.

2.1.6.1.1. Batería LiPo 903052

Características:

- Resistencia interna: 0,002 a 0,006 Ohms
- Capacidad: 1 200 mAh
- VCC: 3,7 VDC
- Dimensiones: 52 mm * 30 mm * 9 mm
- Peso: 31 g

Figura 30. **Batería LiPo 903052**

Fuente: *Catálogo de fabricantes.* https://spanish.alibaba.com/g/lipo-battery-903052.html. Consulta: 4 de noviembre de 2017.

2.1.6.1.2. Batería LiPo 702035

Características:

- Resistencia interna: 0,002 a 0,006 Ohms
- Capacidad: 350 mAh
- VCC: 3,7 VDC
- Dimensiones: 40 mm * 20 mm * 7,5 mm
- Peso: 8 g

Figura 31. **Batería LiPo 702035**

Fuente: *Catálogo de fabricantes.* https://spanish.alibaba.com/g/lipo-battery-903052.html. Consulta: 4 de noviembre de 2017.

2.1.6.2. Cargador Lineal Li-Ion LTC4056

Cargador utilizado específicamente para la carga de baterías LiPo y protección de estas mismas al momento de cargarlas.

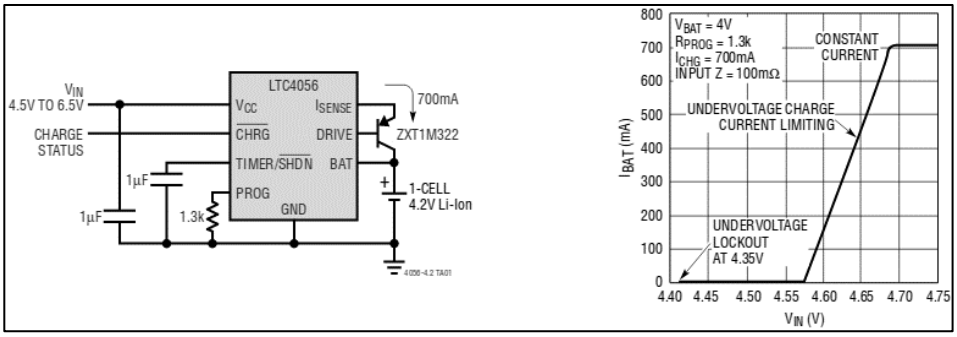
Aplicaciones:

- Teléfonos celulares
- Computadoras portátiles
- Cámaras digitales
- Docks de carga
- Cargadores de bajo costo y tamaño pequeño

Características:

- Cargador TP4056 – LTC4056
- VCC: 4,5 VDC a 5,5 VDC
- Tipo de baterías: LiPo – Li-Ion
- Ajuste automático de corriente de carga
- Voltaje de carga: 4,2 V
- Precisión de carga: 1,5 %
- Exactitud de carga: ±0,6 %
- Imax: 1 000 mA de corriente de carga
- Corriente de carga programable: 200 mA a 700 mA
- Protección: sobre flujos de corriente o temperatura
- Dimensiones: 30 mm * 16 mm
- Peso: 3 g

Figura 32. **Circuito de carga de baterías y gráfica de Voltaje de entrada Vs Corriente del mismo circuito**



Fuente: *Circuito de carga de baterías y gráfica de Voltaje de entrada Vs Corriente del mismo circuito*. http://cds.linear.com/docs/en/datasheet/405642f.pdf. Consulta: 4 de enero de 2018.

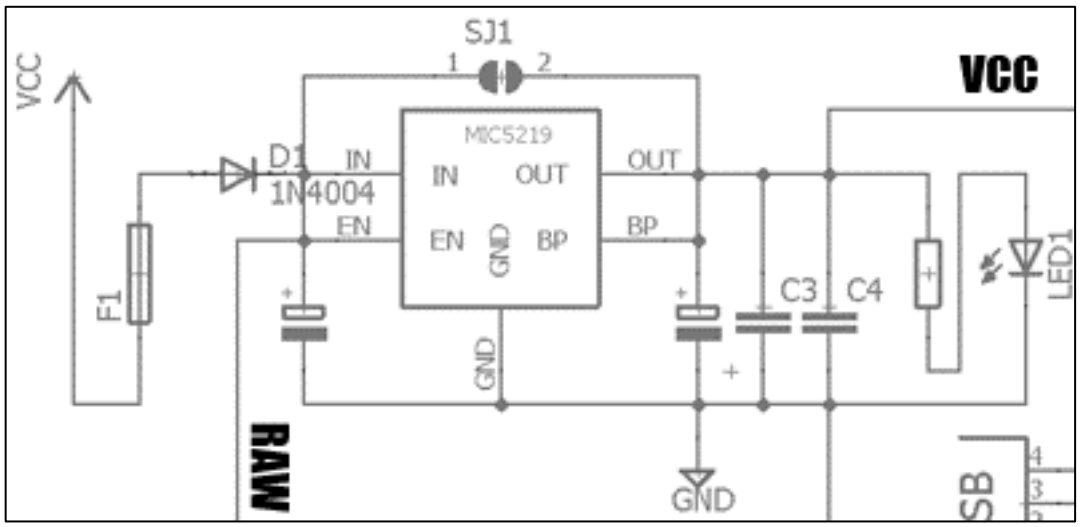
2.2. Análisis de circuitos y conexiones

2.2.1. Ingeniería inversa de placas

2.2.1.1. Arduino pro micro

En la entrada de potencia del arduino pro micro se tiene un fusible de 500 mA para protección del circuito, luego un regulador de voltaje de ultra bajo ruido MIC5219, con capacidad de 500 mA de salida en 3,3 VDC. Dicho regulador cuenta con un condensador electrolítico de tantalio de 10 microfaradios en la entrada para reducir la impedancia de la fuente y consta con 3 condensadores en la salida (uno electrolítico de tantalio de 10 microfaradios y dos cerámicos debajo de 1 microfaradio) del regulador para estabilizar la salida DC y mejorar la respuesta transitoria del mismo.

Figura 33. **Regulador de voltaje**

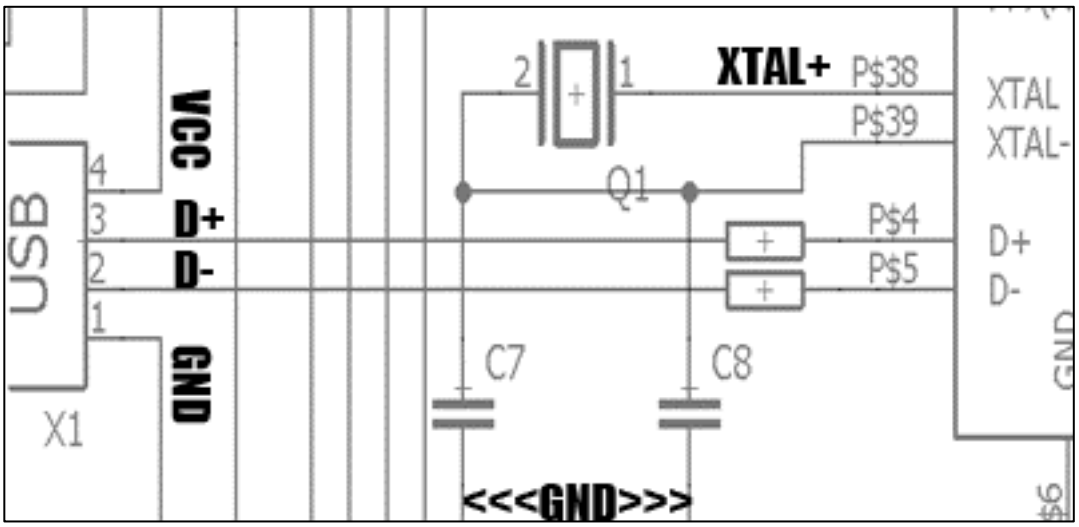


Fuente: elaboración propia empleando Eagle.

Luego, de los condensadores de salida del regulador se tiene una resistencia de 1 KOhms y un led solamente para indicar cuando está conectado a una fuente de voltaje el circuito.

En la parte del microcontrolador se tienen componentes necesarios para el funcionamiento del mismo, entre los cuales destaca una resistencia de 10 KOhms como *reset* (*master clear* en otros casos), condensadores de referencias, resistencias y leds para control de recepción y transmisión de datos en un puerto serial y un circuito oscilador, que consta de un cristal de 16 MHz y 2 condensadores de 22 picofaradios, útiles también para atenuar el ruido eléctrico.

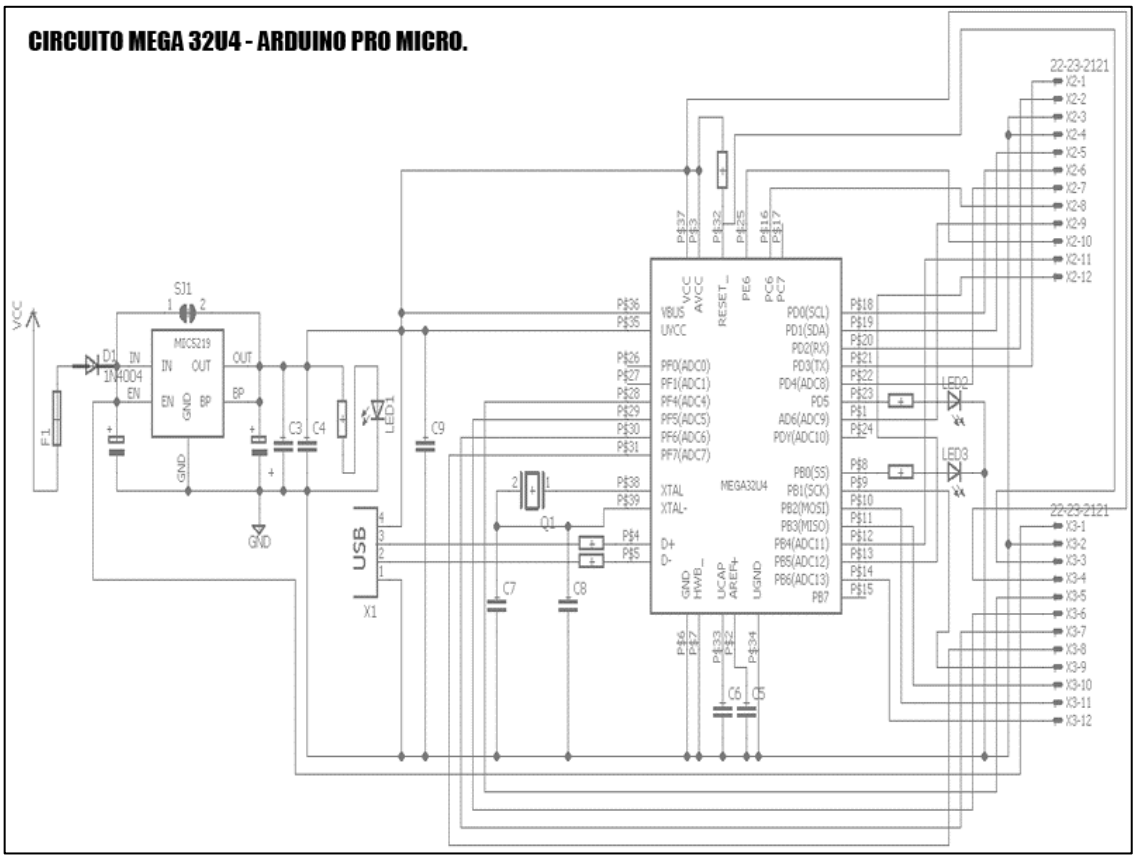
Figura 34. **Oscilador**



Fuente: elaboración propia empleando Eagle.

Si se observa el puerto USB que se comunica directamente a los pines D+ y D- del microcontrolador (Data+ y Data-), solo existen 2 resistencias de 22 Ohms que se oponen al paso de la corriente y atenúan el voltaje para protección de los pines. De igual manera, los pines GPIO se conectan directamente a pines externos de la placa, dejando sin utilizar 5 pines, los cuales son PF0 (ADC0), PF1 (ADC1), PC7, PD7 (ADC10) y PB7.

Figura 35. **Arduino Pro Micro 32U4**



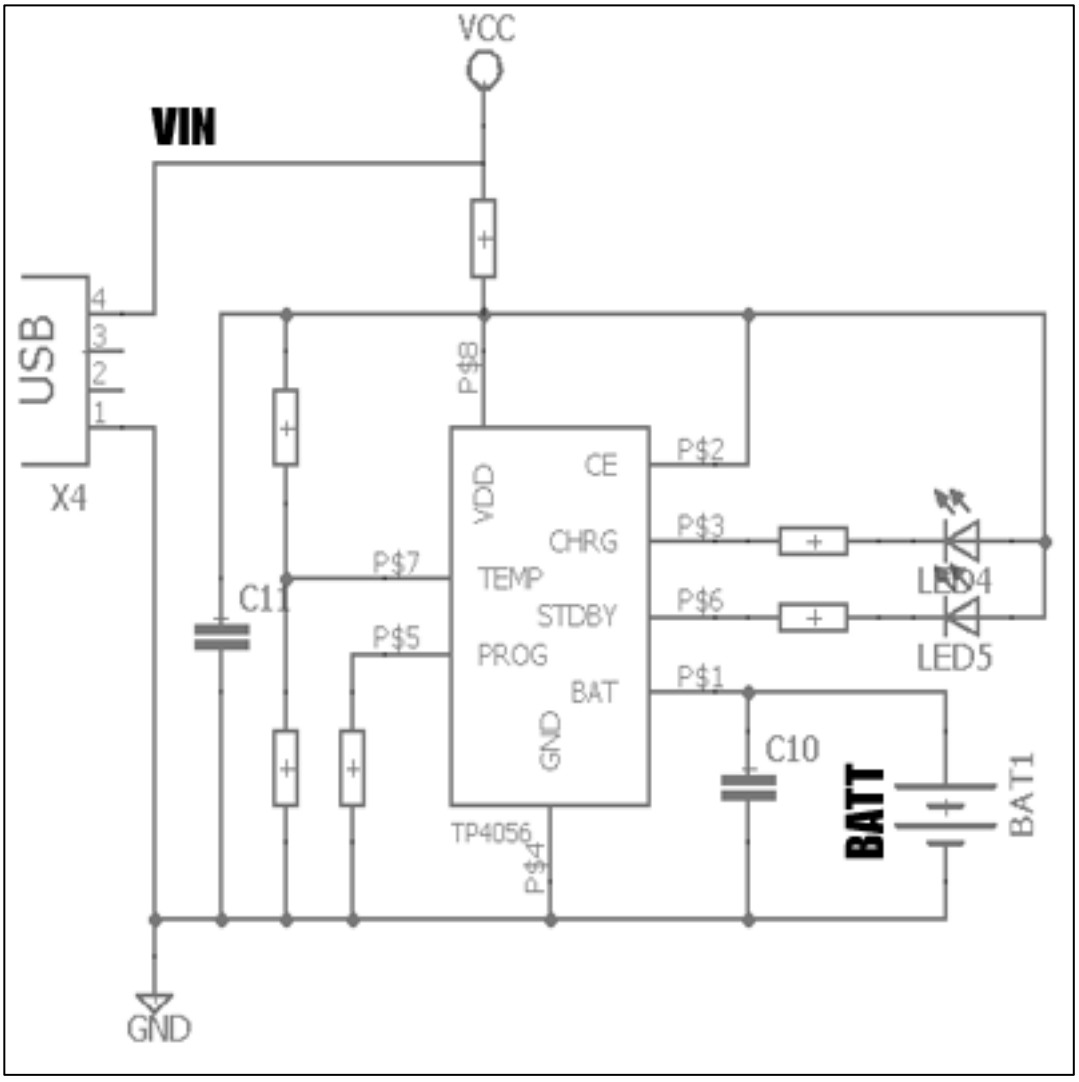
Fuente: elaboración propia empleando Eagle.

2.2.1.2. FC-75

Esta placa consta principalmente de un cargador lineal Li-Ion LTC4056 con conector USB. El cargador es un integrado específico para esto ya que cargar una batería Li-Po es algo muy delicado, si no se le aplica la carga que se requiere o si se le aplica la carga muy rápido, se puede ocasionar un incendio incipiente. El conector USB solamente tiene terminales de VCC y GND, las líneas de datos no se encuentran soldadas a la placa.

El circuito consta básicamente de un monitor de corriente, bloqueo por sobre voltajes, recarga automática y 2 pines que indican el estado de carga.

Figura 36. **Cargador de baterias LiPo**



Fuente: elaboración propia empleando Eagle.

2.2.1.3. GY-521

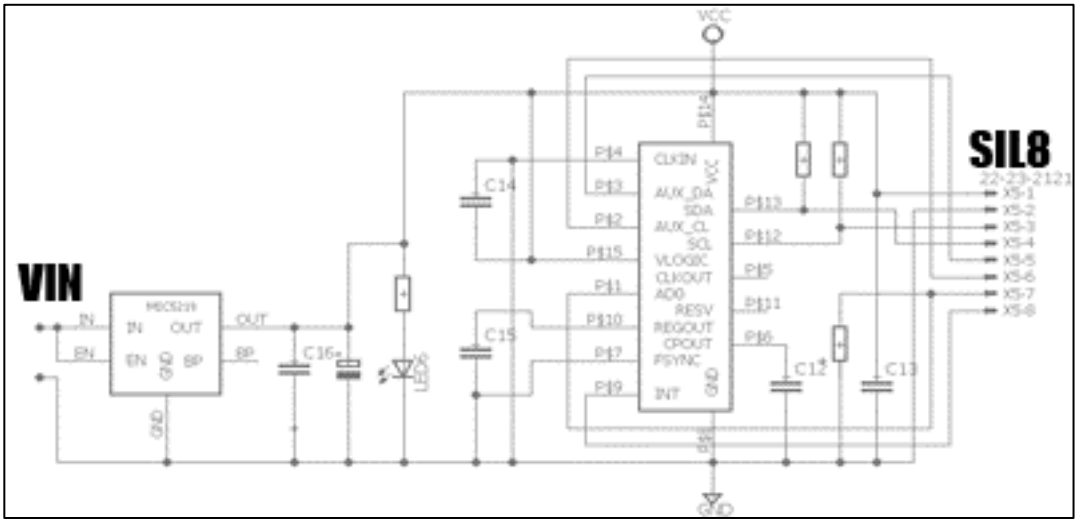
Esta placa consiste principalmente en un acelerómetro y giroscopio MPU6050, el cual es controlado por I^2C por los pines SDA (datos) SCL (reloj). En esta placa estos pines están conectados a 2 resistencias *pull-up*, manteniendo siempre que no están activos los pines en un valor de 3,3 V.

Junto a este puerto de comunicación se tiene también otro puerto auxiliar que aparece en el esquemático como XDA (auxiliar de datos) y XCL (reloj auxiliar).

El pin AD0 define la dirección del protocolo de comunicación a utilizar (0X68, 0X69), para poder tener como máximo 2 sensores en el mismo pin de datos del microcontrolador. El pin INT sirve para programar interrupciones internas del MPU6050 y obtener una salida alta o baja dependiendo de la necesidad que se tenga.

Por último, la entrada de voltaje de esta placa posee un regulador de voltaje lineal con un capacitor de entrada cuya función es reducir la impedancia de entrada y dos capacitores de salida para filtrar la señal de voltaje y estabilizarla; luego se encuentra una resistencia y un led, cuya función es señalizar cuando la placa se encuentre energizada.

Figura 37. **Acelerometro y giroscopio MPU6050**



Fuente: elaboración propia empleando Eagle.

2.2.1.4. ZS-040

Esta placa al igual que las anteriores, cuenta con un rectificador y un regulador de voltaje con sus capacitores correspondientes para reducir la impedancia de entrada y estabilizar el voltaje.

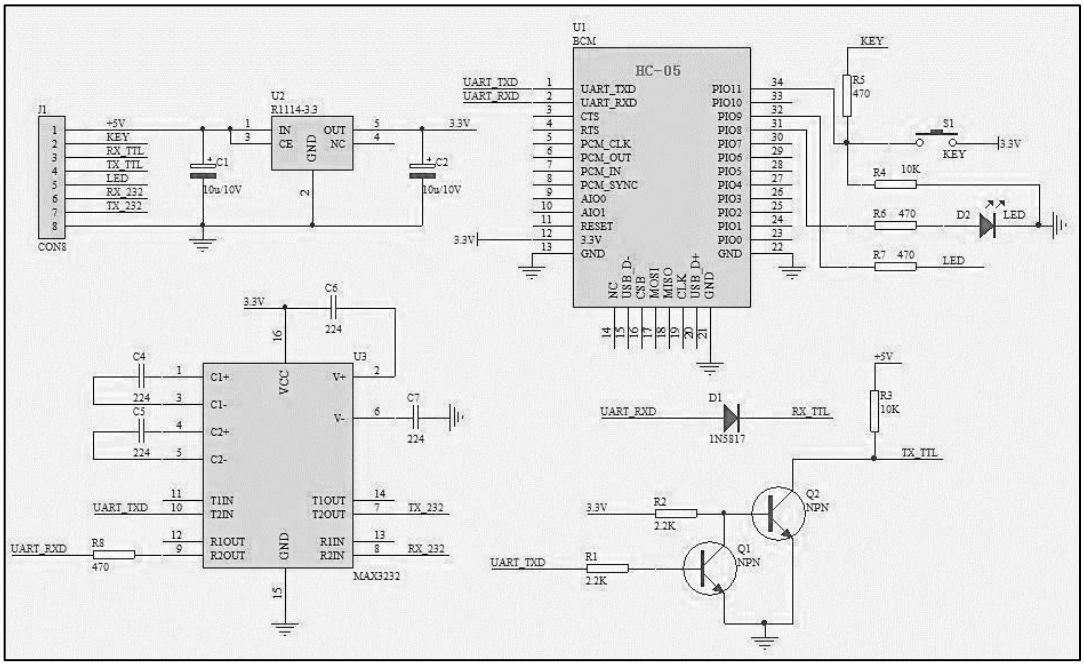
El circuito principal de este módulo es el BC417, que es un chip bluetooth 2.0.

El protocolo de comunicación de este circuito es UART, los cuales se encuentran directamente conectados hacia los pines de la placa.

Entre los componentes que se encuentran en la placa hay una resistencia pull-down conectada a un *push-button*, lo cual sirve para colocar el circuito en

modo configuración. De igual manera, se tiene una resistencia R3 y un led para indicar el estado de conexión del bluetooth.

Figura 38. **Bluetooth HC-05**



Fuente: *Wordpress*. https://rydepier.wordpress.com/2015/05/27/jymcu-bluetooth-hc05-bluetooth-part-2/. Consulta: 4 de enero de 2018.

2.2.1.5. Placas MQ

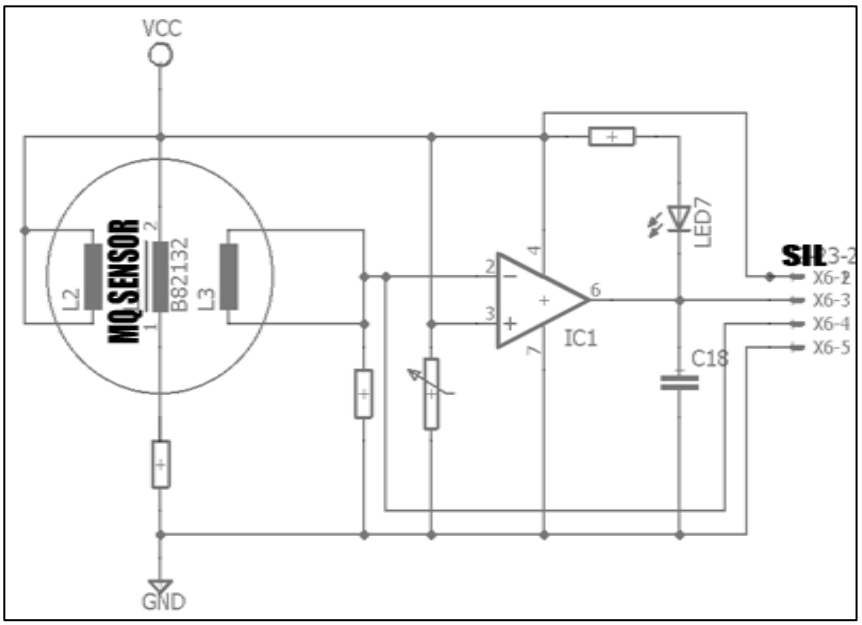
Todas las placas de sensores MQ tienen el mismo hardware requerido, lo único que cambia es el sensor y los valores de algunos de los componentes.

En esta placa se tiene un sensor de gas el cual tiene una bobina interna que se encarga de calentar el gas, la que es conectada a una resistencia externa para no crear un puente directo de VCC a GND.

Luego se tiene una resistencia interna que varía según la densidad de un gas especifico en el ambiente, la cual pasa a formar parte de un divisor de voltaje junto con RL. A partir de la salida de este divisor de voltaje, entramos a un amplificador operacional cuya función es comparar el voltaje de salida del divisor, junto a un voltaje otorgado por el potenciómetro Rp.

En la salida del amplificador operacional tenemos un condensador de desacoplo y un led que se activara cuando sea alta la salida *dout*, la cual solamente indicará una alarma cuando se sobrepase un nivel de comparación ajustable, mientras que para análisis de una señal se necesita la salida analógica presente en el divisor de voltaje anterior al amplificador operacional, llamada *aout*.

Figura 39. **Sensores MQ Series**



Fuente: elaboración propia empleando Eagle.

2.2.2. Diagramas de circuitos y conexiones

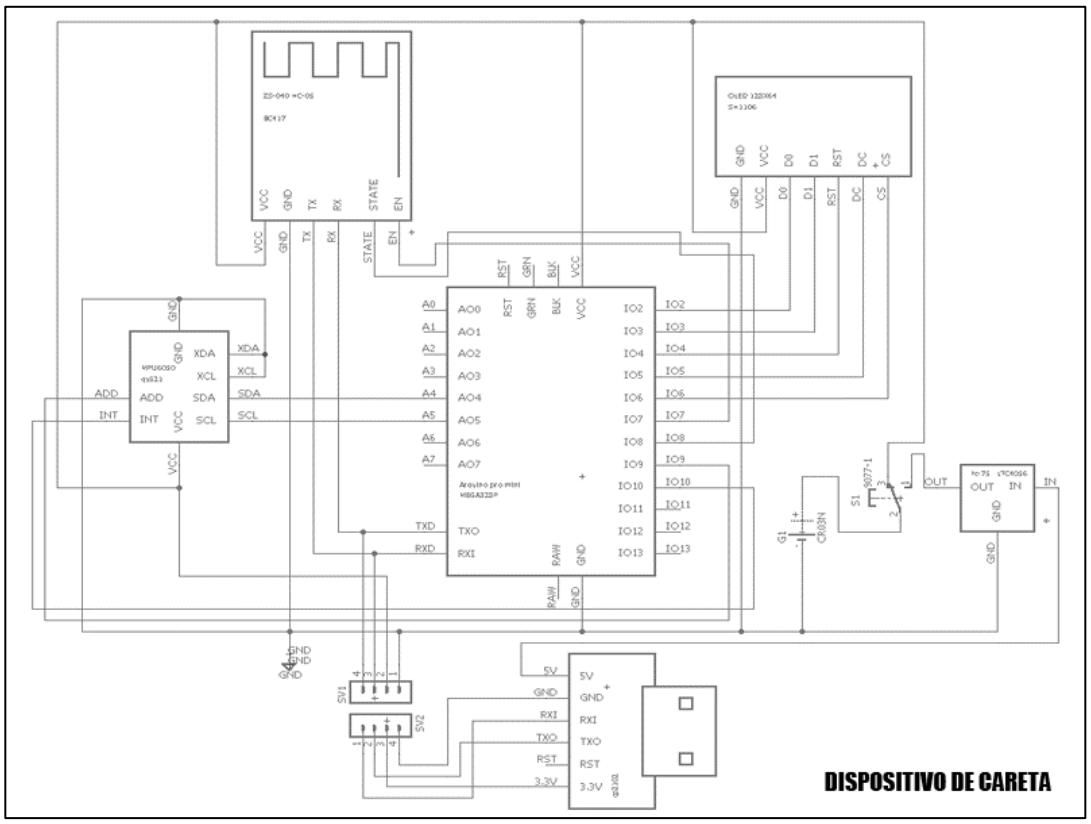
2.2.2.1. Careta

La careta consta de las siguientes partes:

- Bluetooth ZS-040
- USB CP2102
- Sensor GY-521
- Cargador FC-75
- Switch SPDT
- Pantalla OLED
- Arduino pro micro
- Batería 702035

El módulo de la careta está diseñado para colocarse alrededor de un diafragma que se encuentra dentro de la careta del SCBA fabricada por MSA, ya que es el único lugar donde puede quedar inmóvil de una forma fácil en este tipo de caretas.

Figura 40. **Circuito de careta**



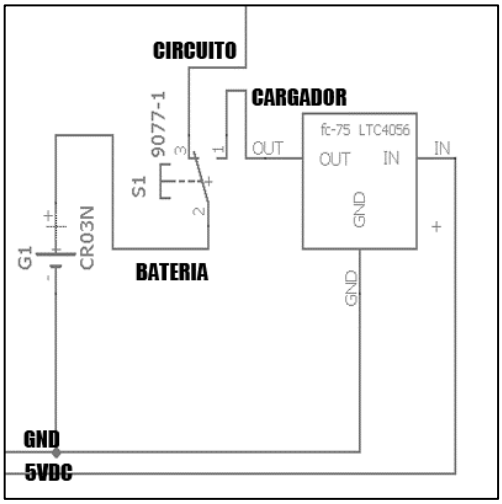
Fuente: elaboración propia empleando Eagle.

Mientras el switch se encuentra en la posición 1, el circuito electrónico se abre y solamente se cierra el cargador de la batería junto a la batería, así se asegura que el cargador funciona solamente cuando el controlador y demás partes se encuentran apagadas y de esta forma se evita dañar la batería.

Mientras el switch se encuentra en la posición 3, funciona todo el circuito normalmente. Es de útil importancia mencionar que para programar el microcontrolador debe estar en la posición de carga el circuito, ya que cuando

está energizado el módulo bluetooth, este interfiere con la programación del controlador.

Figura 41. **Regulador de voltaje careta**

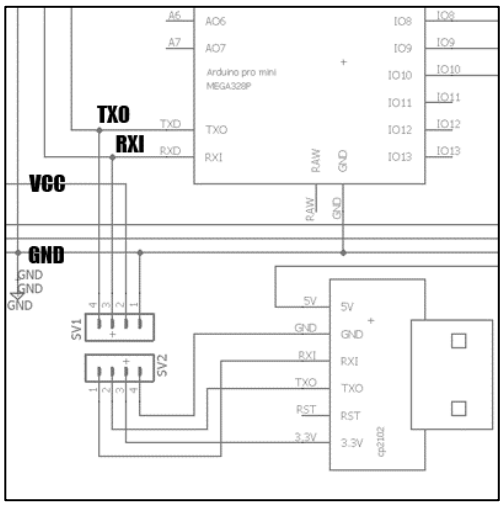


Fuente: elaboración propia empleando Eagle.

La entrada “IN” es una entrada de 5 VDC proveniente de cualquier fuente externa, en el caso de este circuito está conectada al CP2102 para aprovechar esta salida proveniente de un puerto USB de una computadora.

La interfaz de comunicación del dispositivo con una computadora es por un puerto USB, con un conversor de USB a TTL. Esta interfaz puede ser útil para programar el microcontrolador y configurar el módulo bluetooth por medio de un código cargado al microcontrolador.

Figura 42. **Conversor USB a UART CP2**



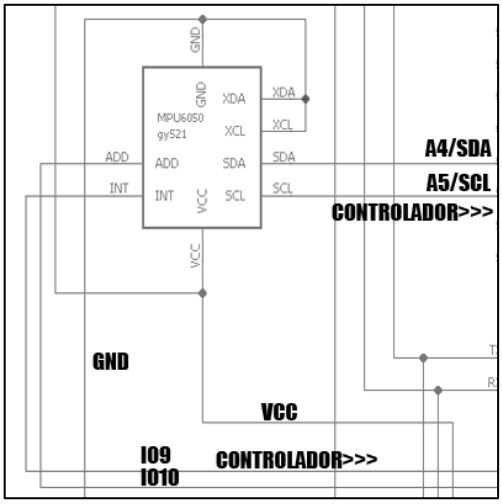
Fuente: elaboración propia empleando Eagle.

En la parte del circuito de control de la careta, se tiene un módulo de comunicación con el *pass*, un módulo de visualización de datos, un sensor de movimiento para tener una versión mejorada de un Pass existente y un controlador. El sensor se comunica con el microcontrolador por medio del protocolo denominado I^2C.

Los pines ADD e INT se conectan a 2 entradas del controlador. Ambas se encuentran en un voltaje de referencia de 0V. El motivo por el que se conectaron directamente al controlador y no a GND es para tener control de toras las partes del dispositivo sin necesidad de desarmar el dispositivo de la careta en caso sea necesario.

Los pines XDA y XDL si están conectados a GND, ya que estas son entradas auxiliares, y no tenemos otro puerto I^2C en el controlador.

Figura 43. **Módulo MPU6050**

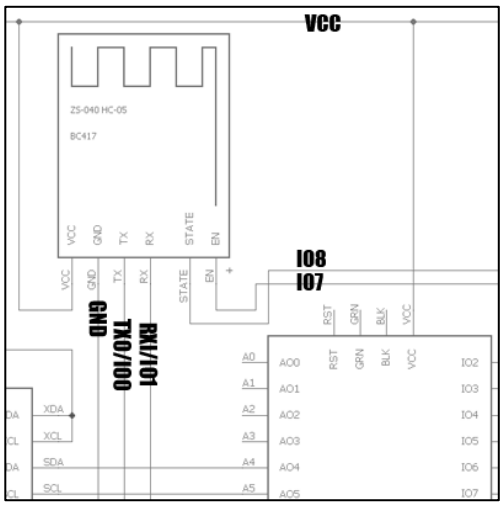


Fuente: elaboración propia empleando Eagle.

En la parte de comunicación con el *pass* se encuentra un módulo bluetooth 2.0. Esta parte es muy necesaria ya que el sensor de movimiento envía datos al *pass* para mantener apagada una alarma de emergencia mientras la careta tenga movimiento. Si la careta no tiene movimiento la alarma se activará automáticamente.

De igual forma es necesaria para enviar los datos provenientes del *pass* hacia la careta, y poder visualizar la magnitud de las variables que se encuentran en el ambiente sin desviar mucho la atención.

Figura 44. **Módulo bluetooth HC-05**

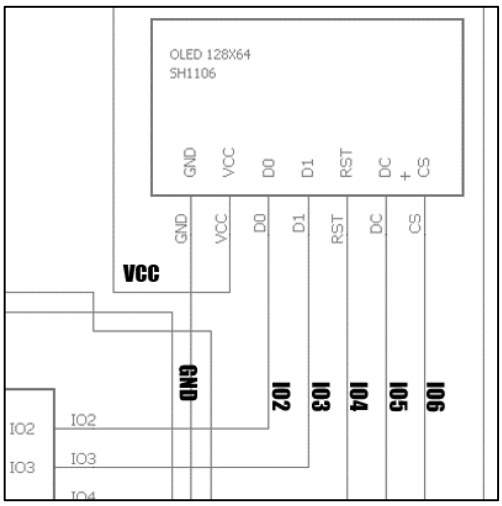


Fuente: elaboración propia empleando Eagle.

Para visualizar los datos de una manera en la que no quite mucho la atención al trabajo, se muestra en una pantalla OLED de 1,2” de longitud, la cual se encuentra en la parte superior del circuito, pero en la careta se visualiza en la parte posterior aproximadamente a 30 grados de inclinación. Para observar la pantalla si es necesario mover la vista hacia ese lugar, es muy poco visible cuando se ve indirectamente, pero está lo suficientemente accesible para visualizar una alarma si es necesario mostrarla.

La pantalla se comunica con el controlador por medio del protocolo SPI. Esta necesita 5 pines para la interfaz, un pin de GND y un pin de VCC. Se utilizó una interfaz SPI para no saturar el canal I^2C que ya tiene el sensor.

Figura 45. **Pantalla OLED SH1106**



Fuente: elaboración propia empleando Eagle.

2.2.2.2. *Pass*

El *pass* consta de las siguientes partes:

- Sensor DHT11
- Sensor DS18B20
- Sensor GY-906
- Sensor MQ-2
- Sensor MQ-135
- Motor vibrador
- Sirena completa
- Bluetooth ZS-040
- USB CP2102
- Cargador FC-75
- Switch SPDT

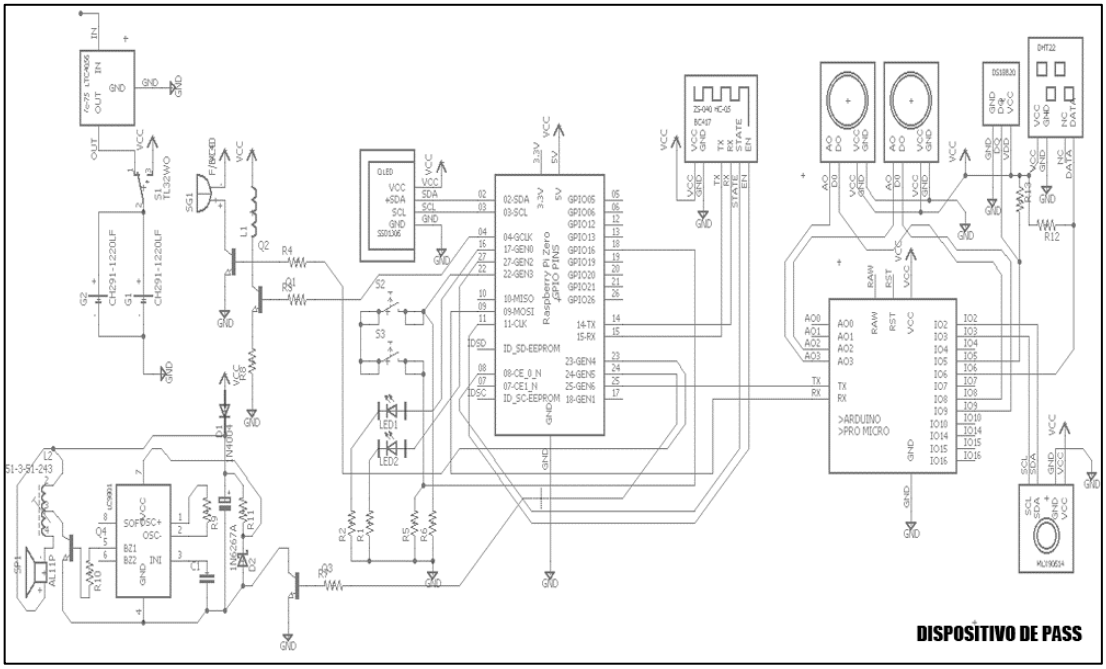
- Pantalla OLED
- Arduino pro micro
- Raspberry pi zero
- Batería 903052

El módulo *pass* está diseñado para colocarse en el arnés del equipo SCBA, el tamaño y el peso de este dispositivo es similar al *pass* ya existente, solo que con muchas más funciones que solo una alarma de emergencia.

El Pass es el dispositivo principal, ya que es el que mide la densidad en partes por millón de los gases correspondientes, temperatura puntual, temperatura remota y humedad. Aparte de esto analiza los datos y los guarda en una base de datos para estudios posteriores.

De igual manera el *pass* muestra estados de conexión con la careta y con un router al tenerlo disponible en caso necesitemos descargar los datos de la base de datos, muestra el uso del procesador y memoria, y puede tomar fotografías en caso sea necesario.

Figura 46. **Pass completo**

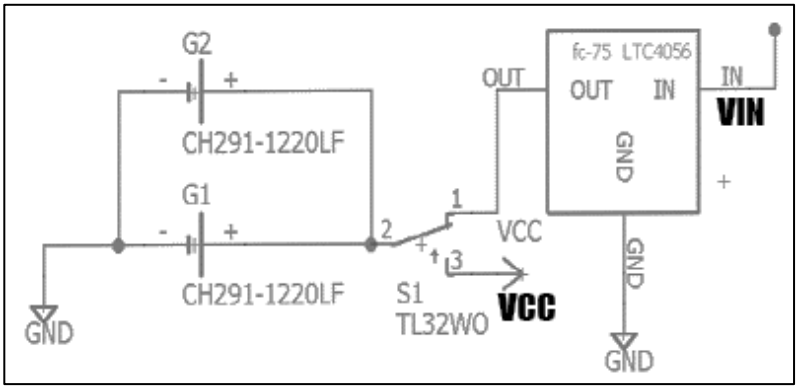


Fuente: elaboración propia empleando Eagle.

Al igual que el diagrama de la careta, mientras el switch se encuentra en la posición 1, el circuito electrónico se abre y solamente se cierra el cargador de ambas baterías, ya que en este circuito se cuenta con 2 baterías en paralelo para aumentar el tiempo de autonomía del circuito. Mientras el switch se encuentra en la posición 3, funciona todo el circuito normalmente.

La entrada VIN del cargador hace referencia a una entrada de 5 V a 5,5 V que sirve para cargar las baterías. La referencia es en común a todo el circuito del *pass*. Este cargador es el mismo que el de la careta, solo que la configuración de las baterías y la cantidad de mAh de ambas baterías es sumamente mayor, por lo cual el tiempo de carga de la careta es aproximadamente un 20 % relativo al tiempo de carga del *pass*

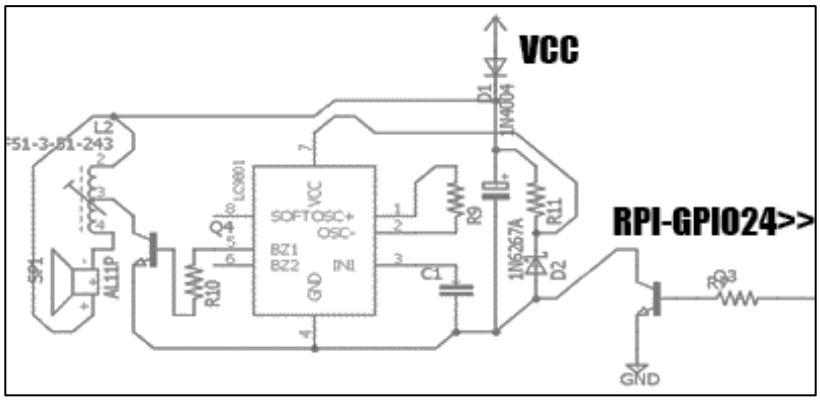
Figura 47. **Regulador de voltaje y cargador de *pass***



Fuente: elaboración propia empleando Eagle.

En cuanto al *pass* directamente, para que este funcione como tal se sincroniza la careta con este para obtener los datos del acelerómetro y giroscopio. Del lado del *pass* se tiene una sirena que emite una alerta sonora activada por un pin en alto en la Raspberry pi zero y desactivada teniendo ese mismo pin en bajo.

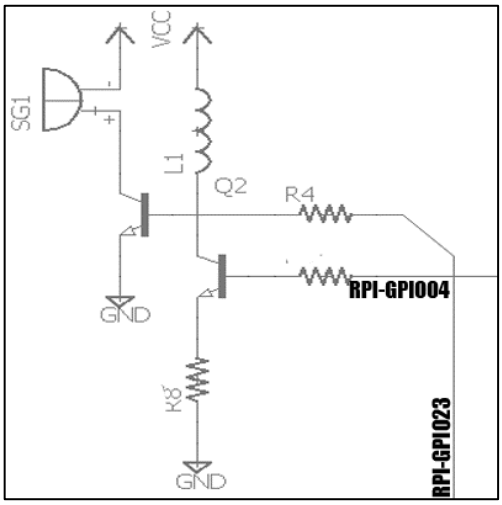
Figura 48. **Sirena *pass***



Fuente: elaboración propia empleando Eagle.

De igual manera, cuenta con 2 tipos de alertas más, una alerta sonora no tan fuerte y un motor vibrador para dar avisos personales como no tocar, ver pantalla, precaución, entre otros. Estas alertas son accionadas por una salida en alto de la Raspberry pi zero.

Figura 49. **Buzzer**



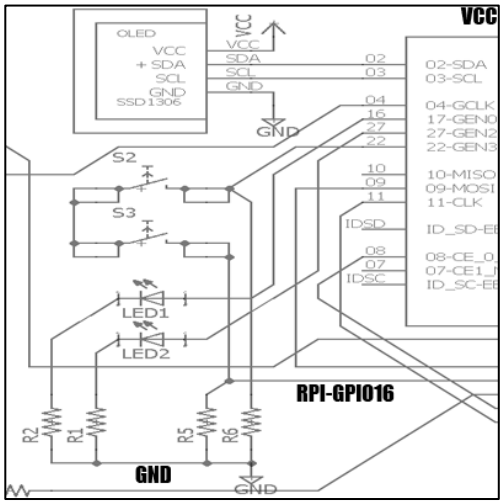
Fuente: elaboración propia empleando Eagle.

En la interfaz para poder interactuar con el prototipo se tienen dos pulsadores y 2 leds al igual que en un *pass* existente. De igual manera se tiene una pantalla OLED para mostrar datos específicos del sistema desplegados en un menú.

En un *pass* común, al presionar los 2 botones una vez se activa la alarma y al presionar los 2 botones juntos de nuevo se desactiva. En este *pass* desarrollado, se debe presionar primero un botón y, mientras se tiene presionado se presiona el otro para activar la alarma, mientras que para desactivarla es el mismo proceso, pero al revés. Esto es simplemente porque

los pulsadores son mucho más sensibles que los botones de un *pass* y de esta forma se disminuye la probabilidad que se accione por algún golpe o algún contacto por accidente en alguna superficie.

Figura 50. **Módulo de pantalla OLED *pass***



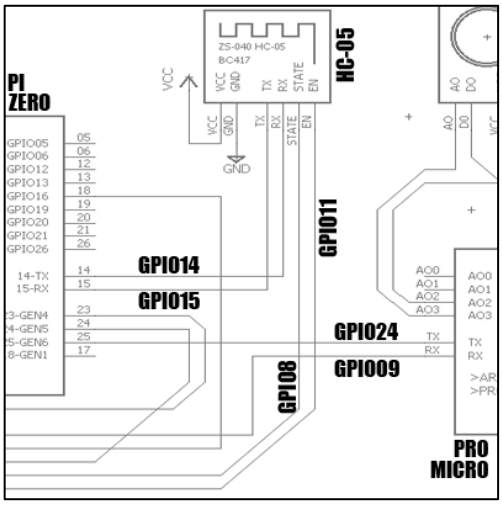
Fuente: elaboración propia empleando Eagle.

La comunicación del *pass* es necesaria tanto para la careta como para el mismo controlador que tiene conectados todos los sensores. Ambos protocolos de comunicación son UART, que son métodos de comunicación serial. Ya que la Raspberry pi zero trae solamente un módulo de comunicación UART definido por hardware; se tomaron 2 pines de la Raspberry pi zero: uno como entrada Rx y el otro como salida Tx, para de esta manera configurar un protocolo UART definido por software. Los pines definidos por software son GPIO25 como Rx y GPIO09 como Tx.

La conexión de la careta hacia el *pass* está dada por un módulo bluetooth HC-05 maestro del lado del *pass*, utilizando el módulo de comunicación UART

definido por hardware, mientras que la conexión de la Raspberry Pi Zero hacia el microcontrolador será directamente por UART, pero en este caso definido por software. El motivo de hacerlo de esta manera es que, si en caso llega a fallar la comunicación por software, quede siempre disponible la alarma.

Figura 51. **Módulo HC-05 *pass***

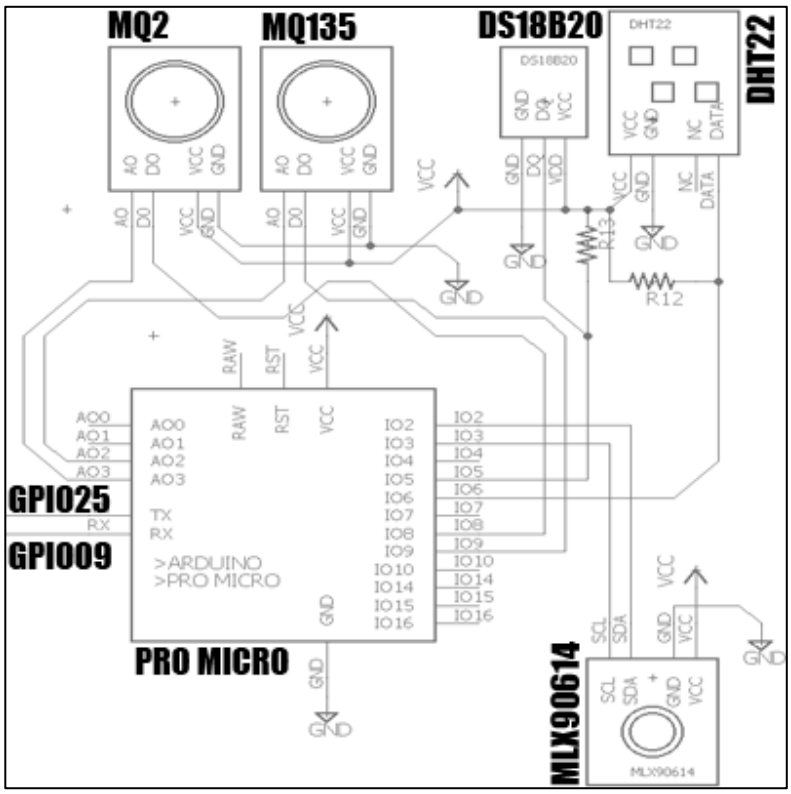


Fuente: elaboración propia empleando Eagle.

Se utilizó el protocolo serial UART ya que es muy sencillo de utilizar y lo suficientemente rápido para transmitir datos sin necesidad de configurar un dispositivo como maestro y otro como esclavo. La configuración maestro-esclavo es solamente para la funcionalidad de los módulos Bluetooth.

En el *pass*, se dividió el muestreo de datos del área de análisis, los datos son muestreados y evaluados con un microcontrolador, que solamente envía datos certeros al microprocesador. Esto es para no saturar la Raspberry Pi Zero y que esta solamente sirva para análisis de datos, conexión y almacenamiento.

Figura 52. **Sensores *pass***



Fuente: elaboración propia empleando Eagle.

El microcontrolador toma muestras de gases en el ambiente para comprobar la pureza del aire, temperatura ambiental por medio de 3 sensores diferentes, temperatura de un objeto a distancia y humedad relativa. Estos son los únicos datos ambientales muestreados, ya que hasta el momento son los únicos necesarios. Es posible que para un mejor análisis sea necesario un sensor de presión relativa, sensor de oxígeno y sensor de sulfuro de hidrogeno.

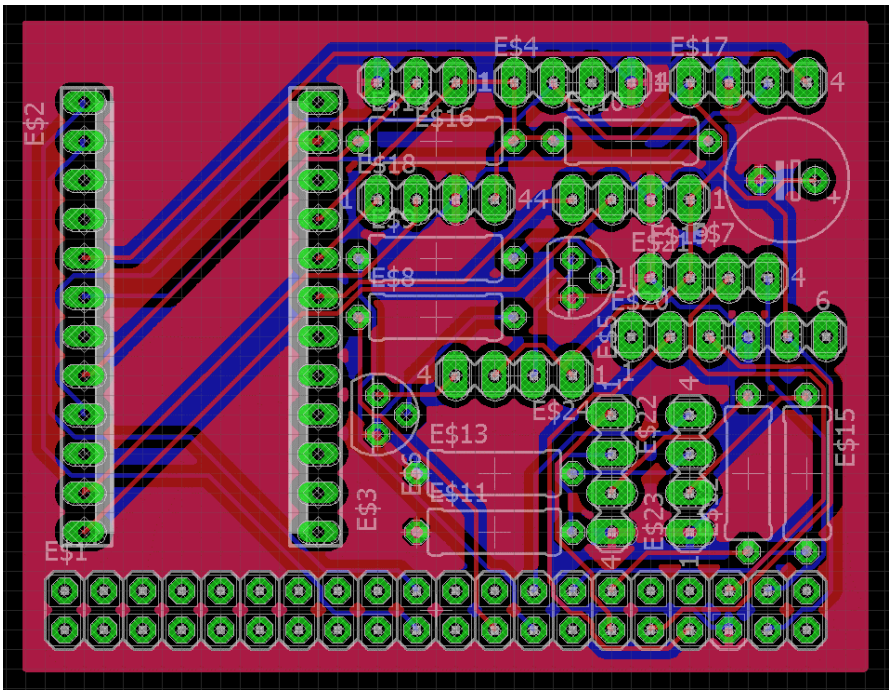
2.2.3. Diseño de placa de circuito impreso (PCB)

La placa de circuito impreso de este módulo de prueba, contiene únicamente 2 transistores con su resistencia de base correspondiente para activar un buzzer y una sirena, 2 resistencias *pull down* para pulsadores, 2 resistencias *pull up* para sensores, 2 resistencias para controlar la corriente en los leds y múltiples conectores para pantalla, módulo bluetooth, sensores, controlador y headers para Raspberry Pi Zero. Esta placa es de doble cara, ya que, con la cantidad de pistas relacionada al tamaño de esta misma, no podía completarse la cantidad de rutas en una sola cara.

Ambas caras de la tarjeta están diseñadas con masa común, esto se refiere a que todas las referencias del circuito van hacia la tarjeta sin quemar en ambos lados de la PCB. Esto sirve para reducir la cantidad de pistas en el circuito y para reducir el ruido eléctrico en el circuito, ya que estaría funcionando de una manera similar a una jaula de Faraday la tarjeta completa.

La tarjeta de prueba ha sido desarrollada con componentes normales, sin componentes superficiales. En el caso de desarrollar este prototipo por completo, la placa debe ir con componentes superficiales y sin ningún modulo, ya que todos los circuitos anteriores en este mismo capítulo irían montados sobre la misma placa. Aquí se muestra la PCB con sus 2 caras, drills, serigrafia, nombres, valores y componentes distribuidos.

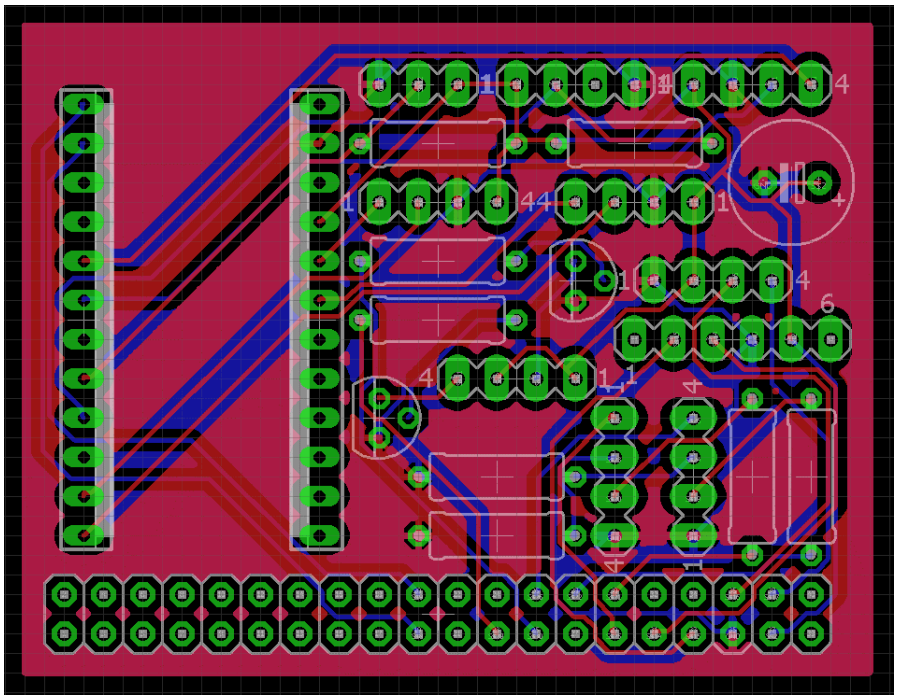
Figura 53. **PCB completa**



Fuente: elaboración propia, empleando Eagle.

En esta vista, se muestra la tarjeta como quedaría luego de desarrollarla, solamente con las 2 caras, drills y serigrafía de cómo van colocados los componentes.

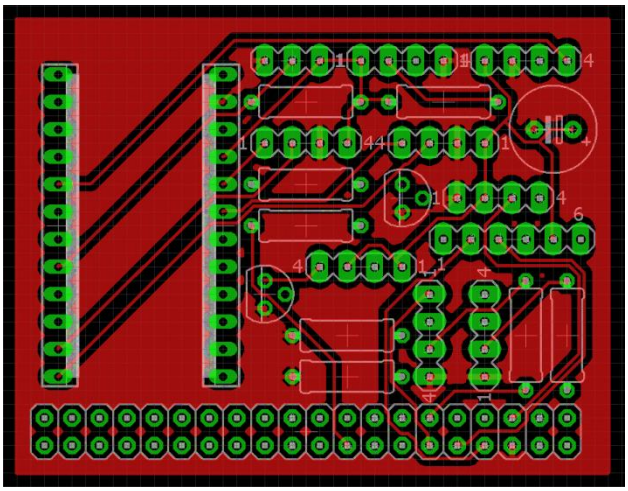
Figura 54. **PCB sin componentes**



Fuente: elaboración propia, empleando Eagle.

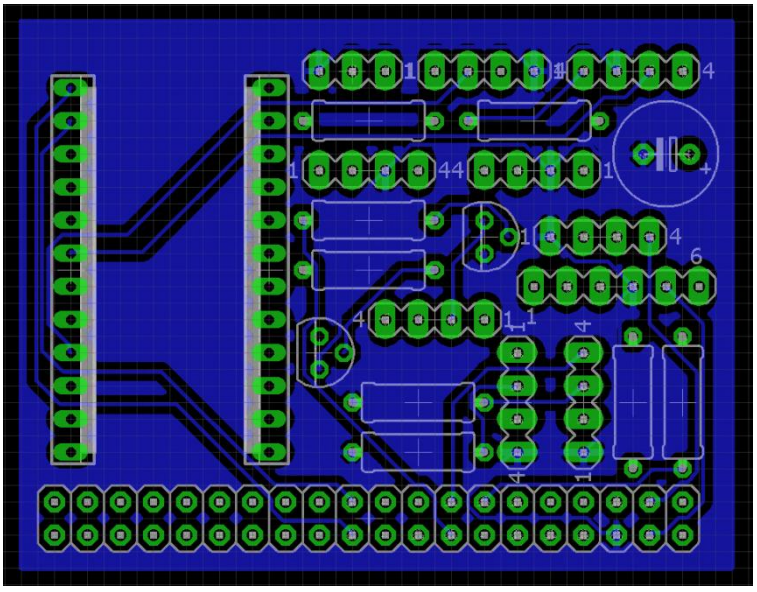
Las próximas 2 vistas de la PCB, son cada una de las caras de la misma, obteniendo la imagen con pistas azules como la parte posterior de la tarjeta, y la imagen con pistas rojas como la parte superior de la tarjeta. Todos los componentes van colocados sobre la cara superior y van soldados sobre la cara posterior. En el caso de realizar la placa a mano, es necesario unir ambas caras en los lugares donde existen pistas de ambos lados. Si las placas se mandan a hacer, estas vienen con tubos metálicos bañados en estaño para que solamente sean ensamblados por uno de los dos lados.

Figura 55. **PCB parte superior**



Fuente: elaboración propia, empleando Eagle.

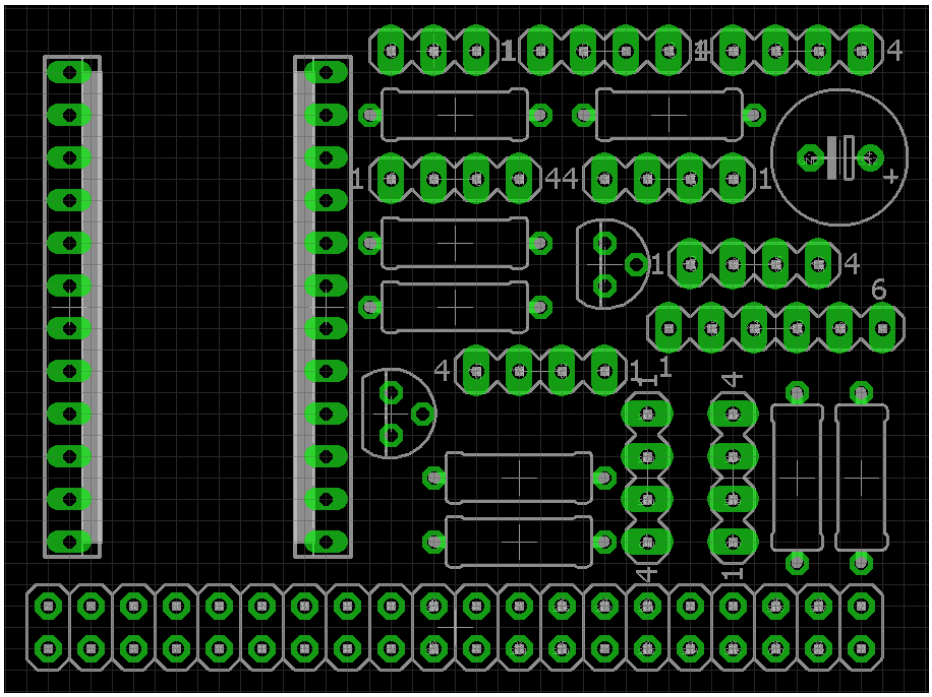
Figura 56. **PCB parte inferior**



Fuente: elaboración propia, empleando Eagle.

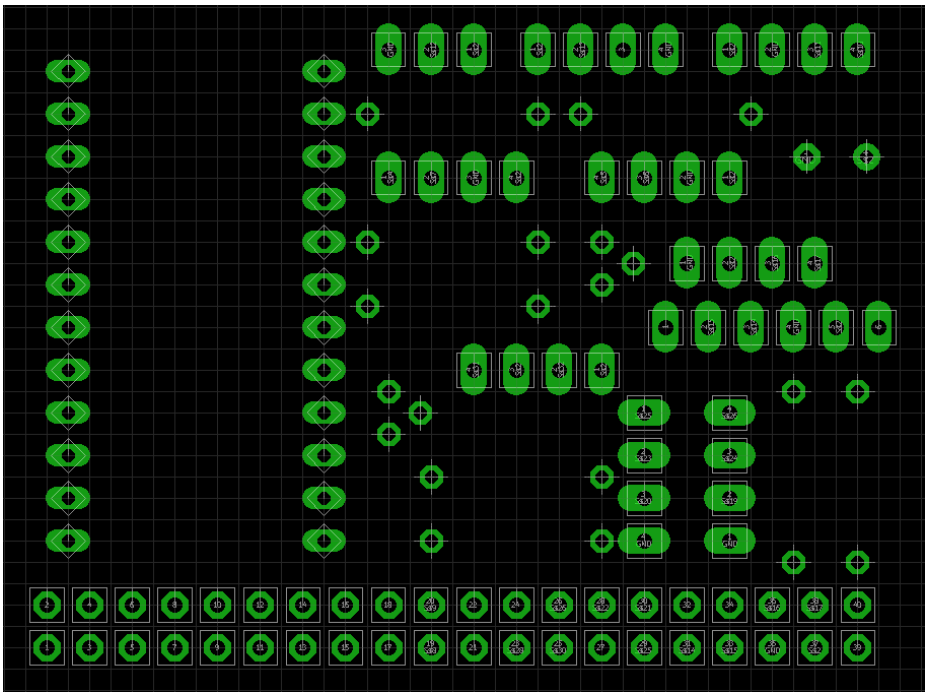
Las últimas 2 vistas de la PCB, son para mostrar los componentes, su ubicación y los lugares que necesitan agujeros en la PCB; de igual manera muestra en verde los únicos lugares que deben llevar estaño, ya que son los únicos lugares con cierta barrera de protección entre el cobre de la placa y el área para soldar. Si el estaño sale fuera de esa área es muy probable que se unifiquen 2 o más pistas.

Figura 57. **PCB con capa de soldaduras y componentes**



Fuente: elaboración propia, empleando Eagle.

Figura 58. **PCB unicamente con capa de soldaduras**



Fuente: elaboración propia, empleando Eagle.

2.3. Estructura

El objetivo de desarrollar una estructura es proporcionar una solución de ensamblaje y protección y estética a los componentes electrónicos.

El diseño de la estructura no debe ser insignificante, debe ser óptimo cuando se necesita solventar una carencia en algún desarrollo de cualquier tipo, específicamente cuando necesitamos proteger y encapsular algún mecanismo de peligros externos.

La elección del material, funcionalidad, geometría y servicio postdesarrollo de la estructura definirán el éxito de la misma, ya que, al no ser útil alguno de

estos puede que se necesite regresar de nuevo a una etapa completa de diseño.

Dado que al desarrollar una estructura se deben conocer todas las funciones y condiciones de trabajo en donde se someterá la pieza, debe ser ligera para no cargar mucho peso extra en un bombero, sólida para soportar cualquier golpe o caída, resistente a temperaturas medias-altas y resistente a la humedad.

La estructura de igual manera debe otorgar una fácil manipulación de los componentes electrónicos que tendrá internamente con la menor cantidad de pulsadores posible.

El último de los factores que influye en el diseño de la pieza es su costo, ya que el sistema es una solución para los diferentes cuerpos de bomberos existentes, de altos o escasos recursos.

2.3.1. Materiales utilizados

- PLA (ácido poliláctico): material utilizado en su mayoría de veces para hacer envases biodegradables. Es un termo plástico obtenido a partir de almidón de maíz, caña de azúcar, yuca o mandioca. Este termo plástico es ampliamente utilizado en impresión 3D bajo un proceso llamado FDM.

- ABS (acrilonitrilo butadieno estireno): material utilizado en piezas que requieren alta resistencia al impacto y en protecciones industriales. Este plástico es muy resistente a altas temperaturas y aporta una gran rigidez y tenacidad a temperaturas tanto altas como bajas.

- Tornillos galvanizados de acero 4,2: tornillos milimétricos utilizados para unir partes separadas en la estructura.

- Silicón rojo: silicón utilizado para construir empaques resistentes a temperaturas cercanas a 315 °C.

2.3.2. Diseño y desarrollo de la estructura

- Módulo 1

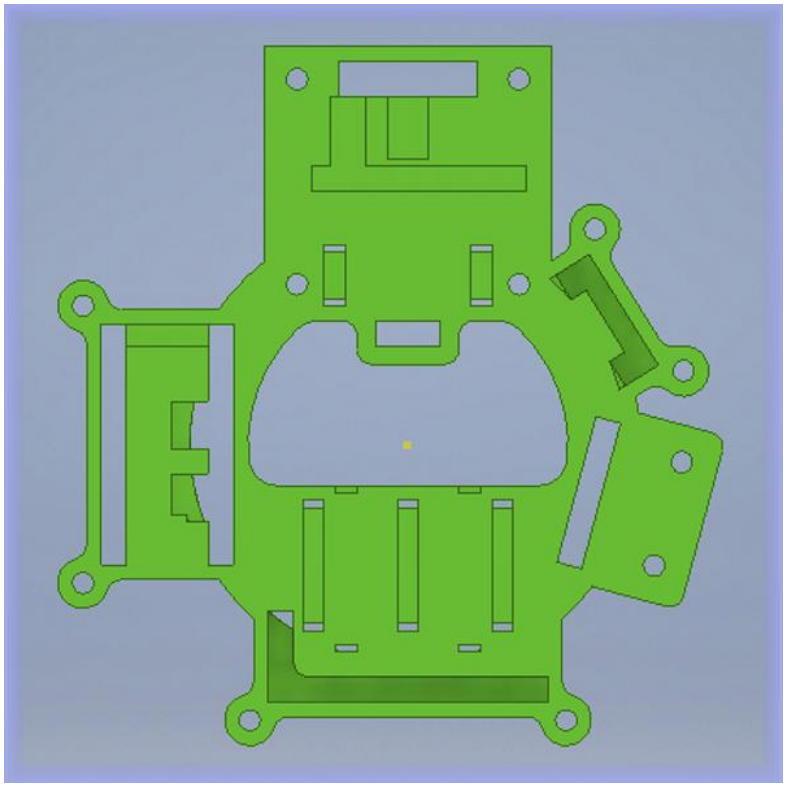
El módulo 1 hace referencia al diseño completo del dispositivo colocado en la careta de protección respiratoria de bomberos, el cual muestra los datos analizados por el módulo 2 y envía muestras de datos del giroscopio y acelerómetro hacia el módulo 2.

La pieza 1 de este módulo es la pieza principal, en donde se acoplan los sensores, microcontrolador, batería, cargador de batería, pantalla OLED y módulo de comunicación bluetooth. En la parte frontal de esta pieza se ensamblan los dispositivos, mientras que en la parte trasera de la pieza se instalan todos los cables necesarios para su correcto funcionamiento.

Toda ranura, agujero, hendidura o forma desarrollada en esta estructura tiene un motivo, ya sea desde un correcto acople con los dispositivos electrónicos existentes en cada tarjeta electrónica para evitar presionar una tarjeta en un punto específico al atornillar, para facilitar el acceso del usuario a la visualización, pines de programación, acceso a carga de la batería o para colocar de una manera óptima los dispositivos más concurrentes a cambios.

La siguiente imagen muestra la pieza 1 vista desde el frente. Esta vista muestra la base de la estructura donde se colocarán los dispositivos. De igual manera presenta bordes y agujeros para acoplar otra pieza a esta misma.

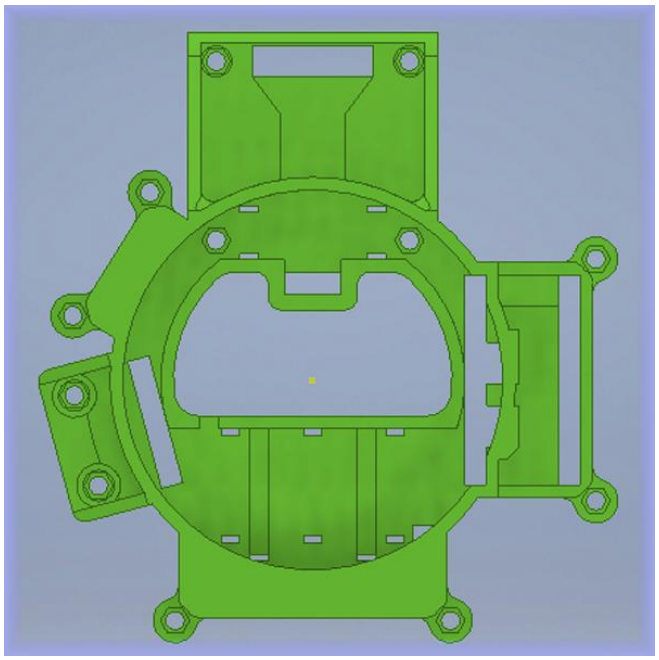
Figura 59. **Pieza 1, careta frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

En la siguiente imagen se muestra la parte posterior de la pieza 1, la cual tiene como fin acoplar la pieza a la careta del equipo de respiración SCBA marca MSA. El acople va directamente sobre el perímetro del diafragma de la careta únicamente a presión, para colocar y quitar fácilmente.

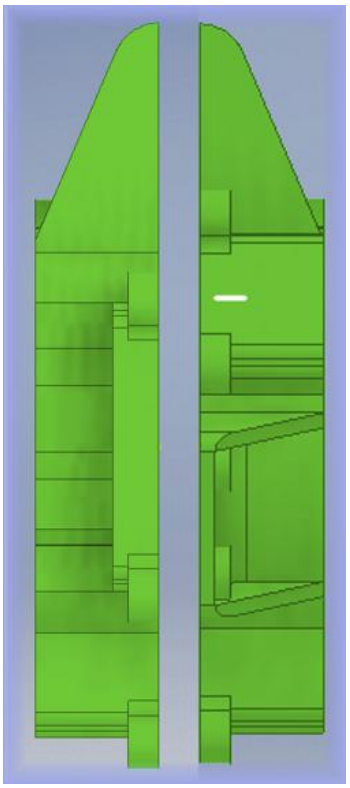
Figura 60. **Pieza 1, careta dorsal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra los laterales de la pieza 1. Aquí se puede visualizar la relación del grosor de la pieza 1 en cuanto a las demás partes. De igual manera se tiene visibilidad del ángulo de los soportes de la pantalla, el cual tiene la misma magnitud que el ángulo del visor de la careta.

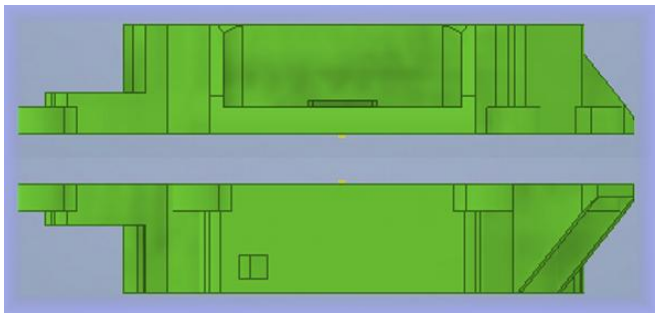
Figura 61. **Pieza 1, careta laterales**



Fuente: elaboración propia, empleando Autodesk Inventor.

En la siguiente imagen, se muestra la parte superior y la parte posterior de la pieza 1. Se puede visualizar un agujero para pasar fácilmente los cables procedentes de la pantalla y un agujero para visualizar el led de conexión del módulo bluetooth.

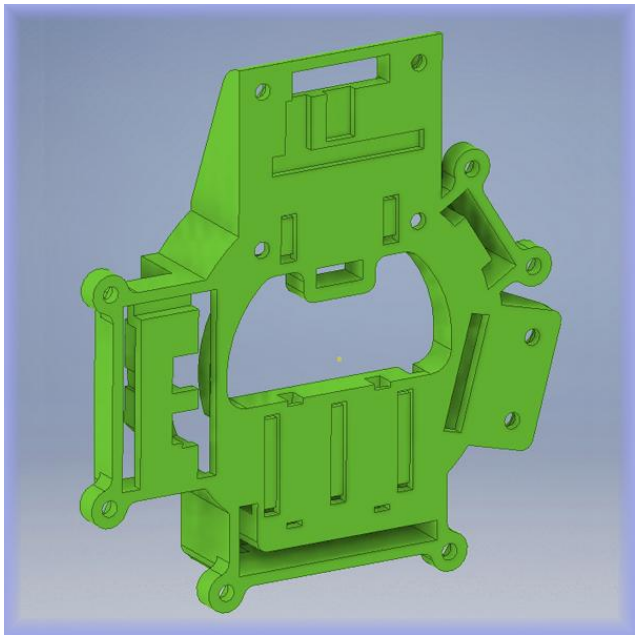
Figura 62. **Pieza 1, careta superior y posterior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista angular de la pieza 1 vista desde el lado izquierdo de la parte superior.

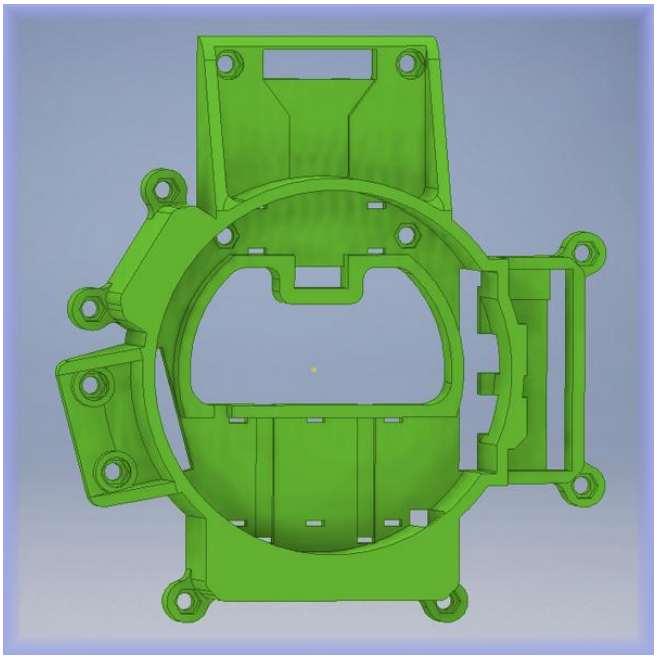
Figura 63. **Pieza 1, careta vista angular superior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista angular de la pieza 1, vista dese la parte lateral derecha posterior.

Figura 64. **Pieza 1, careta vista dorsal superior**



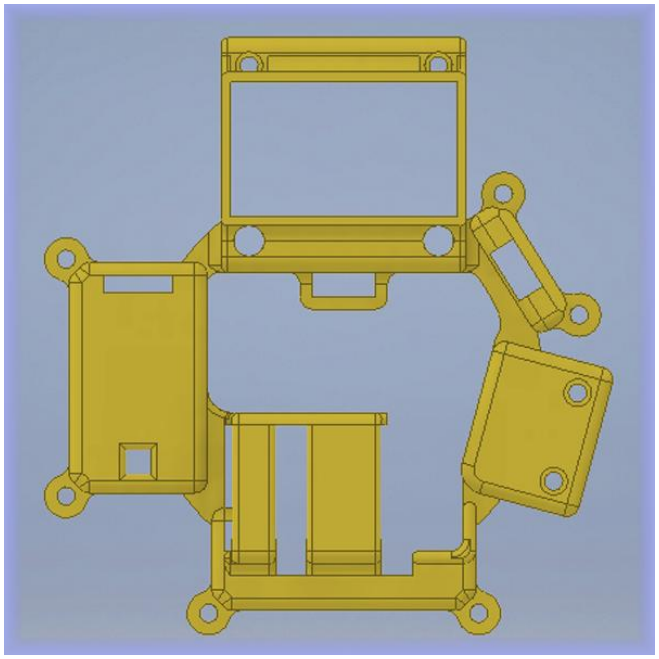
Fuente: elaboración propia, empleando Autodesk Inventor.

La pieza 2 de este módulo hace referencia a la parte que protege todos los componentes electrónicos del exterior, esto es en cuanto a golpes y a temperaturas medias – altas. Esta pieza es muy importante, ya que es la pieza superior de todo el ensamblaje y toda la electrónica se concentra entre esta pieza y la anterior. Para proteger la electrónica de la humedad, se debe aplicar una capa de empaques de silicón negro a los agujeros que exponen al circuito.

La siguiente imagen muestra la pieza 2 vista desde el frente, esta es la parte de esta pieza que estará expuesta en el interior de la careta. Esta pieza

es fundamental, ya que protege de golpes y caídas todos los componentes colocados sobre la pieza 1.

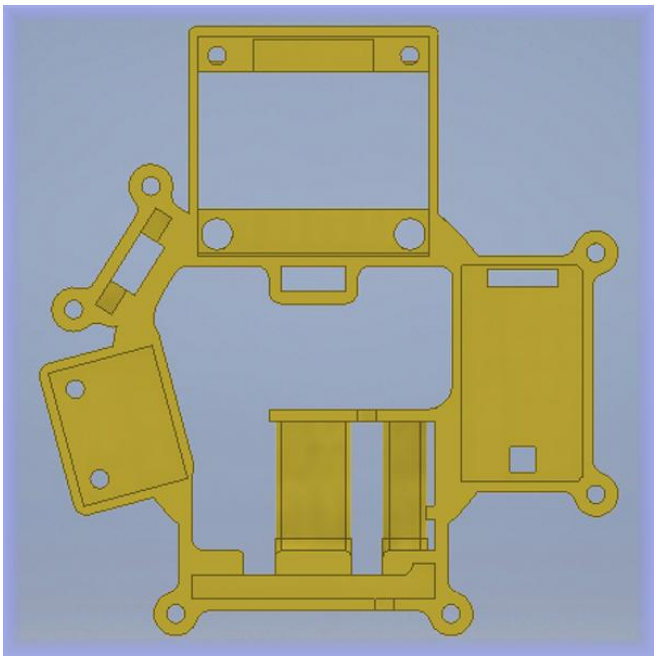
Figura 65. **Pieza 2, careta frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

En la siguiente imagen se tiene la pieza 2 vista desde la parte trasera. Esta parte es la que esta interna junto al circuito. En esta vista se aprecian algunos agujeros de acople y otros agujeros que son para comunicación, visualización o simplemente para atravesar cables de un punto a otro.

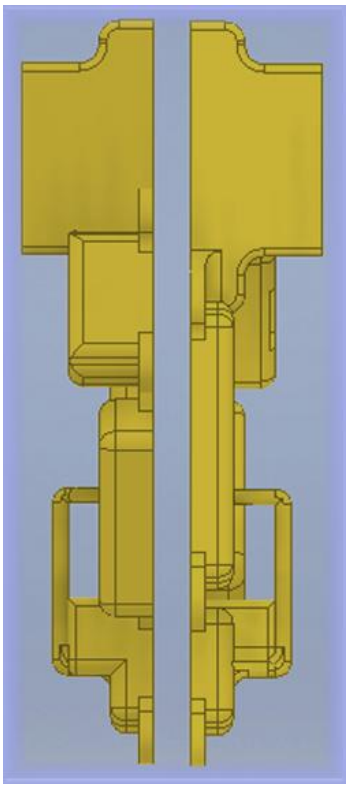
Figura 66. **Pieza 2, careta dorsal**



Fuente: elaboración propia, empleando Autodesk Inventor.

En la siguiente imagen se tiene la pieza 2 vista desde sus laterales. Se puede observar la altura de la protección de la pantalla, esta no solo es para proteger, sino para poder colocar un lente de Fresnel a la distancia correcta de la pantalla para ampliar proporcionalmente la imagen. De igual manera se pueden apreciar algunos agujeros en la parte que protege a la batería; esto es por si en algún momento la batería tiene problemas para que se pueda visualizar fácilmente antes de colocarse la careta, así evitar accidentes referentes a la batería por desconocer la condición de la batería.

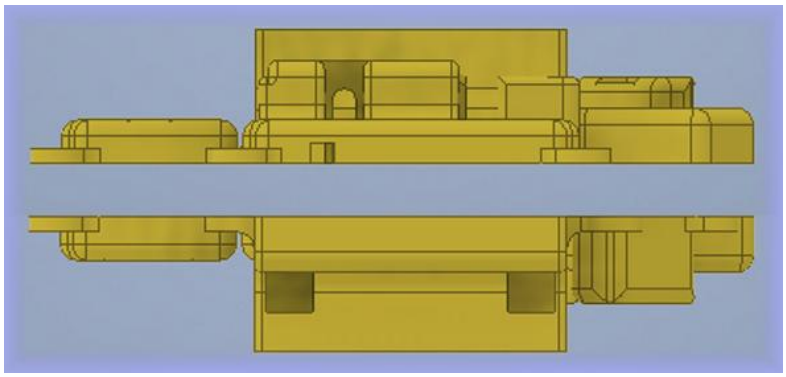
Figura 67. **Pieza 2, careta laterales**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la parte superior y posterior de la pieza 2. La pieza hace contacto directamente con la pieza 1, por lo cual al acoplarse se recomienda sellar la pieza con silicón de alta resistencia.

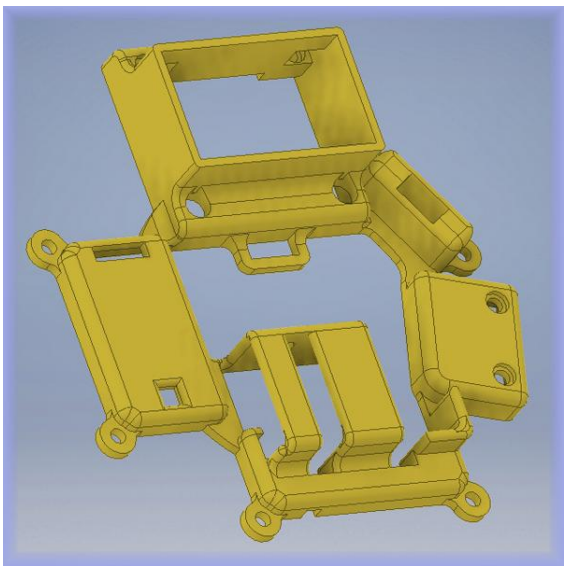
Figura 68. **Pieza 2, careta superior y posterior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista angular frontal superior izquierda de la pieza 2.

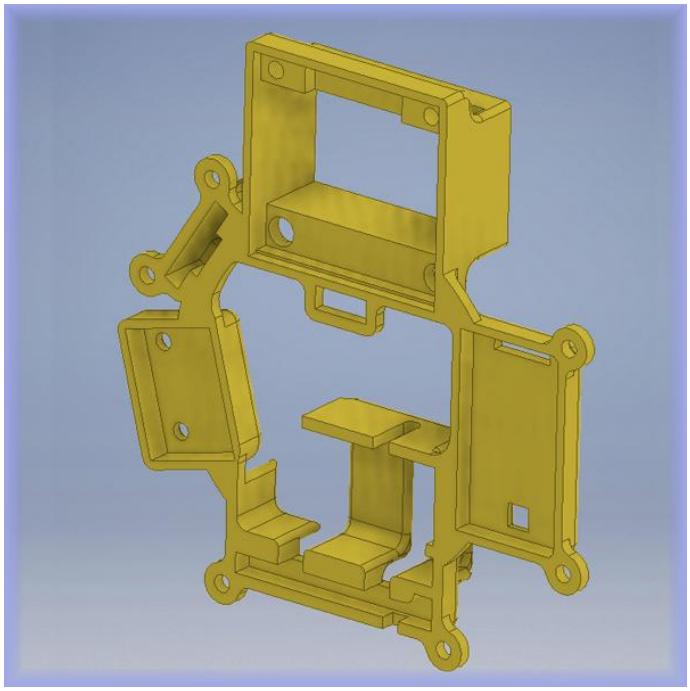
Figura 69. **Pieza 2, careta inferior izquierda**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista posterior izquierda de la pieza 2.

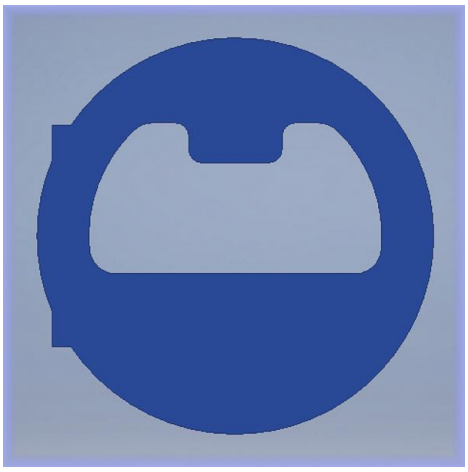
Figura 70. **Pieza 2, careta dorsal derecha**



Fuente: elaboración propia, empleando Autodesk Inventor.

La pieza 3 de este módulo es únicamente protección a los cables necesarios para conectar la alimentación de las tarjetas de circuitos y los cables de comunicación. Estos van dentro del canal circular de la pieza 1. Esta pieza es necesaria para evitar presionar los cables y evitar fracturas internas dentro de los mismos, por lo cual esta pieza quizás sea la de menor importancia de todas, pero no es irrelevante.

Figura 71. **Pieza 3, careta frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista superior izquierda de la pieza 3.

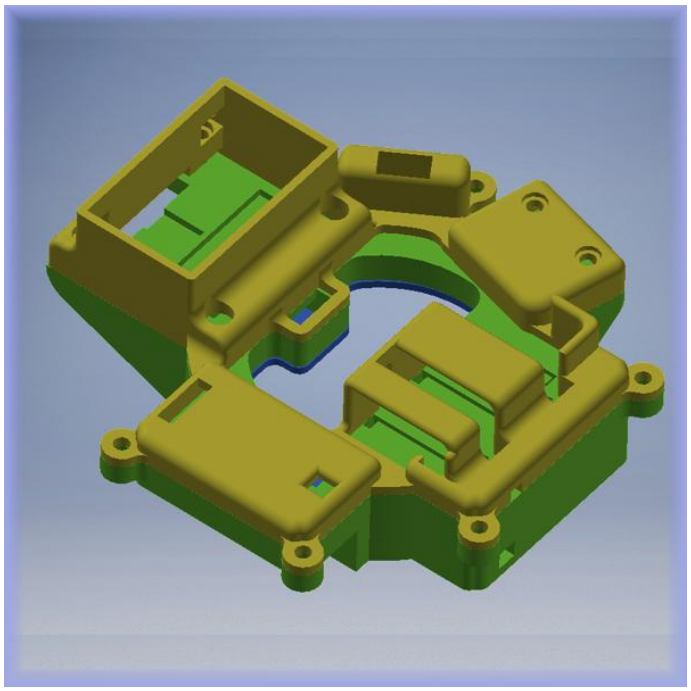
Figura 72. **Pieza 3, careta angular**

Fuente: elaboración propia, empleando Autodesk Inventor.

En las siguientes ilustraciones, se observan las 3 piezas del módulo 1 acopladas tal y como deben de ubicarse. La pieza 3 visualizada en azul se acopla únicamente a presión a la pieza 1, mientras que la pieza 2 se atornilla a la pieza 1. El motivo de esto es el espacio en el que se ubican, ya que la pieza 3 se encuentra colocada por encima de un diafragma que sirve para poder escuchar en el exterior de la careta lo que habla la persona que la lleva puesta, y este espacio está libre de temperaturas muy altas, chorros directos de agua y golpes contusos.

La siguiente imagen muestra una vista frontal izquierda del acople de todas las piezas del módulo de careta.

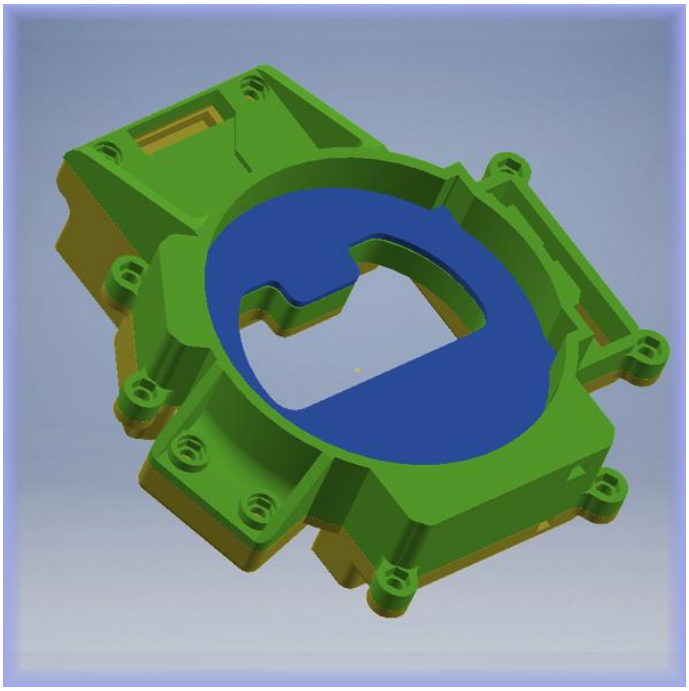
Figura 73. **Piezas careta ensambladas vista angular superior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista posterior derecha del acople de todas las piezas del módulo de careta.

Figura 74. **Piezas careta ensambladas vista dorsal inferior**



Fuente: elaboración propia, empleando Autodesk Inventor.

- Módulo 2

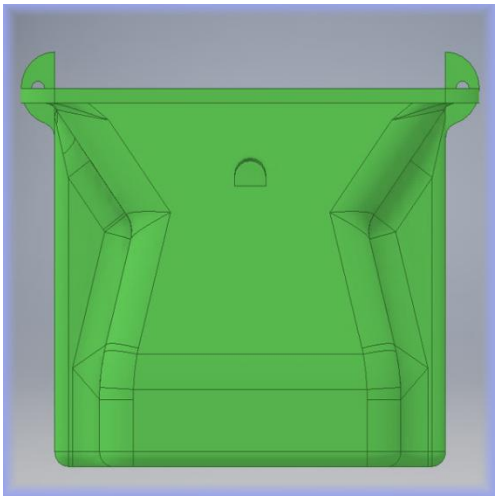
El módulo 2 de este dispositivo completo hace referencia a la parte que va externa a la careta. Esta parte se conoce como *pass* y cumple con las funciones del mismo junto a la toma de muestras de datos ambientales. Esta estructura va colocada sobre la chaqueta de bomberos o sobre un sostenedor del equipo SCBA, quedando siempre al frente del bombero.

Es importante que este módulo siempre se encuentre al frente y no en la parte de atrás, ya que si se coloca en la parte de atrás puede que los datos medidos sean erróneos.

La pieza 1 de este módulo hace referencia a la parte donde se ubican los sensores más importantes y el *pass*. En este módulo se ubica el circuito de sirena activado por un pulso en alto, la bocina de la sirena de emergencia, sensor de humedad y temperatura DHT22, sensor de temperatura DS18B20, buzzer y motor vibrador para alertas de información personal.

En la siguiente imagen se visualiza la pieza 1 desde el frente. En el frente contiene únicamente un agujero para el sensor DS18B20, ya que es el más importante en cuanto a mediciones y no puede interferir ningún objeto entre el sensor y la radiación del fuego para que las mediciones sean completamente certeras.

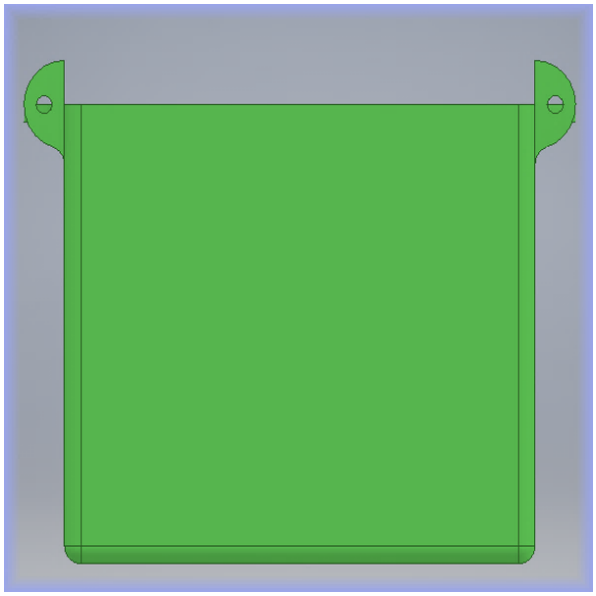
Figura 75. **Pieza 1, *pass* frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la pieza 1 desde la parte trasera. Esta parte es la que hace contacto con el tórax del bombero sobre la chaqueta, motivo por el cual está absolutamente vacía. Dentro de la parte posterior de esta pieza existe un espacio para ingresar una tarjeta RFID, la cual puede servir para usos múltiples en bomberos, desde tomar asistencia al iniciar un turno, saber quiénes integran ciertos grupos en las unidades y saber quién está adentro o afuera de un incendio.

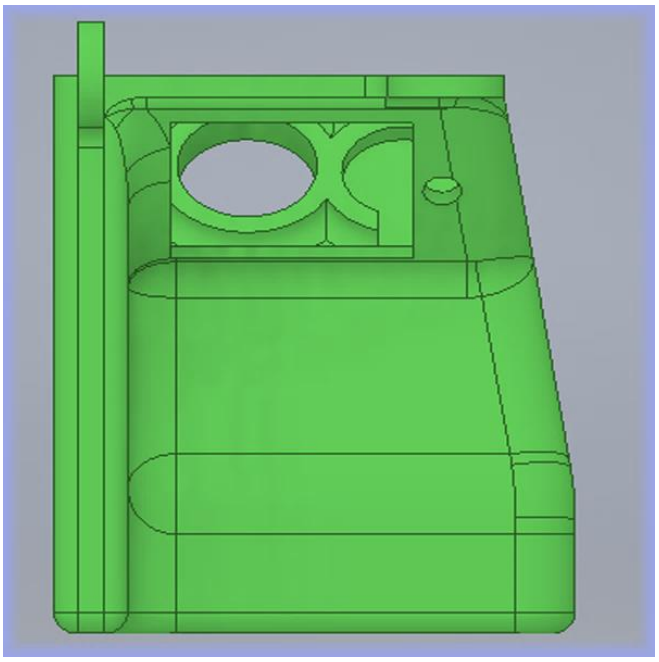
Figura 76. **Pieza 1, *pass* dorsal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la pieza 1 desde el lateral izquierdo. En esta parte se acopla el sensor de humedad y temperatura DHT22, el cual debe medir la radiación y humedad de manera indirecta, motivo por el cual se encuentra en ese sitio.

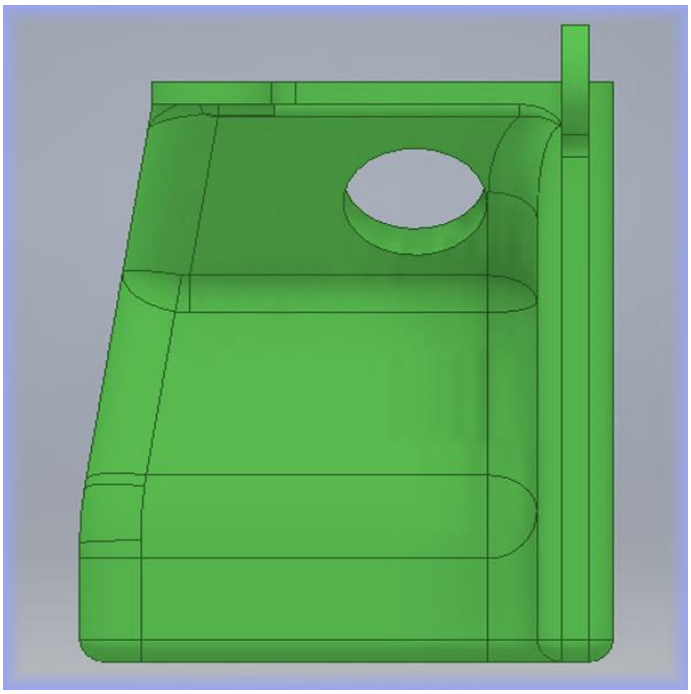
Figura 77. **Pieza 1, *pass* lateral derecha**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la pieza 1 desde el lateral derecho. En este sitio son ubicados los componentes de alerta personal (buzzer y motor vibrador). El motivo de colocarlos allí es simplemente para ahorrar espacio ya que ningún otro modulo podía colocarse en este punto por sus dimensiones.

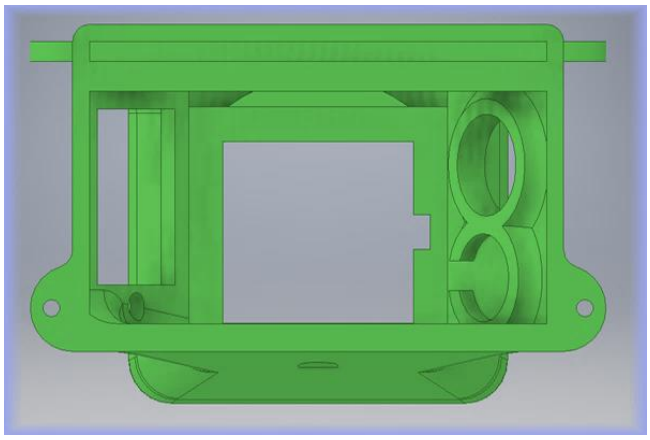
Figura 78. **Pieza 1, *pass* lateral izquierda**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la pieza 1 desde la parte superior. Esta parte va acoplada a otras 2 partes, por lo cual al acoplarla necesita sellarse con silicón de alta resistencia. Desde esta parte superior son ingresados todos los módulos y sensores a excepción de la bocina. La bocina debe ingresarse por abajo y los cables hacia el circuito correspondiente deben soldarse desde adentro antes de ser colocada.

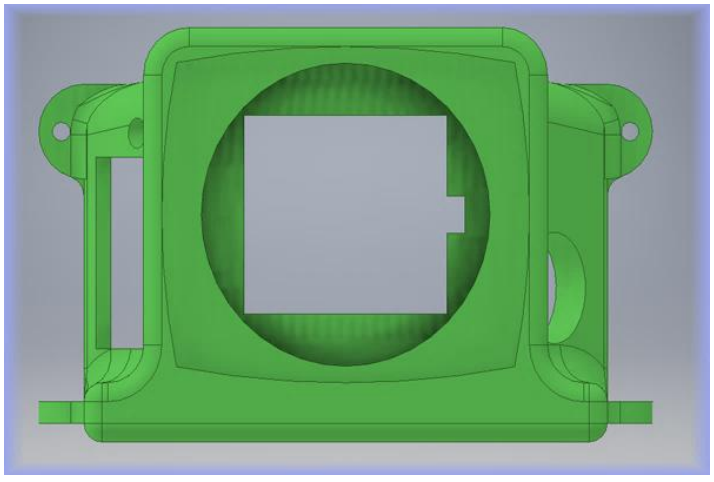
Figura 79. **Pieza 1, *pass* superior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la pieza 1 desde la parte posterior. En esta parte se coloca la bocina de la sirena.

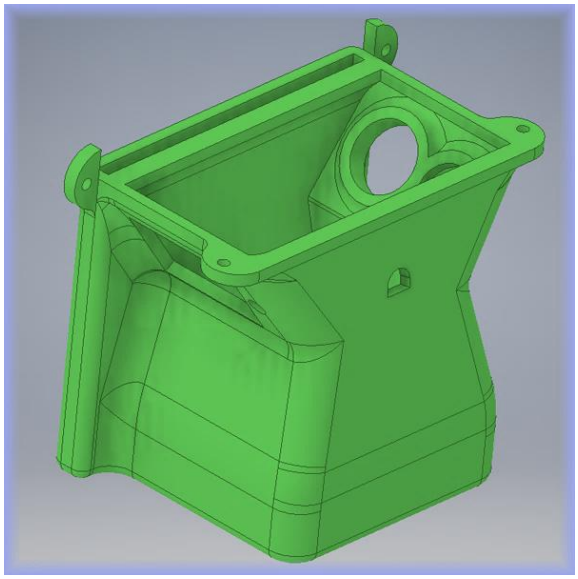
Figura 80. **Pieza 1, *pass* posterior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista frontal superior izquierda de la pieza 1.

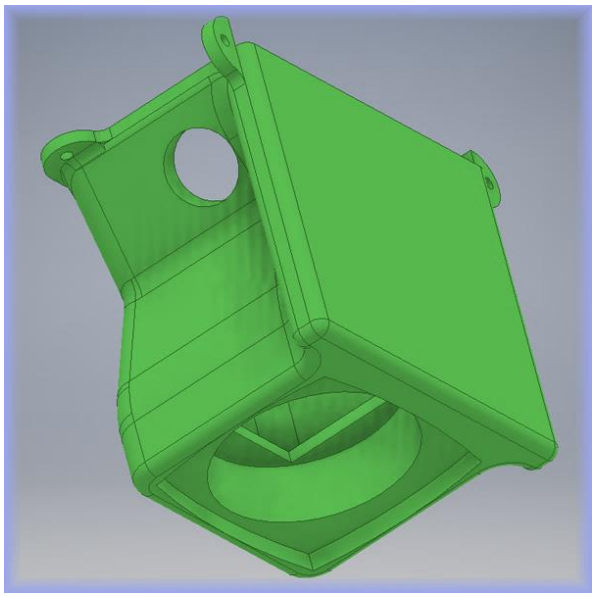
Figura 81. **Pieza 1, *pass* angular superior derecha**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista posterior derecha de la pieza 1.

Figura 82. **Pieza 1, *pass* angular dorsal poster**

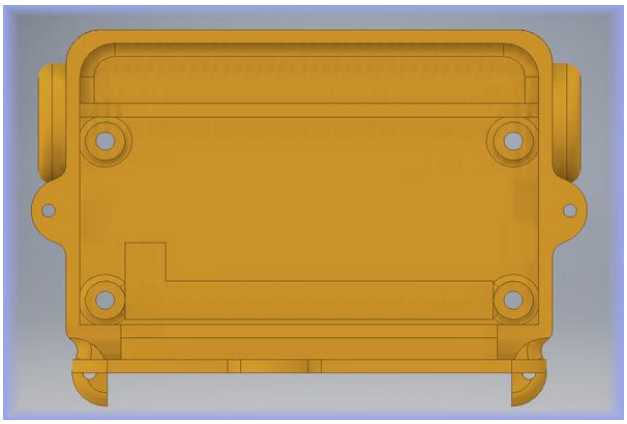


Fuente: elaboración propia, empleando Autodesk Inventor.

La pieza 2 de este módulo de *pass*, es utilizada para cerrar la estructura donde se coloca la tarjeta RFID. Además, en esta pieza se ensambla la Raspberry Pi Zero con el shield correspondiente al desarrollo de este *pass*. Dentro de la estructura se coloca frente al shield el módulo bluetooth y las baterías, sin tener estos un lugar específico. El motivo de no tener un lugar específico es porque esta estructura es una versión de implementación. Cuando esta versión funcione en su totalidad se terminará de desarrollar la versión de producción completa, la cual tendrá dimensiones mucho menores.

La siguiente imagen muestra la parte frontal de la pieza 2, dejando expuesto el lugar específico donde se ensambla la Raspberry Pi Zero.

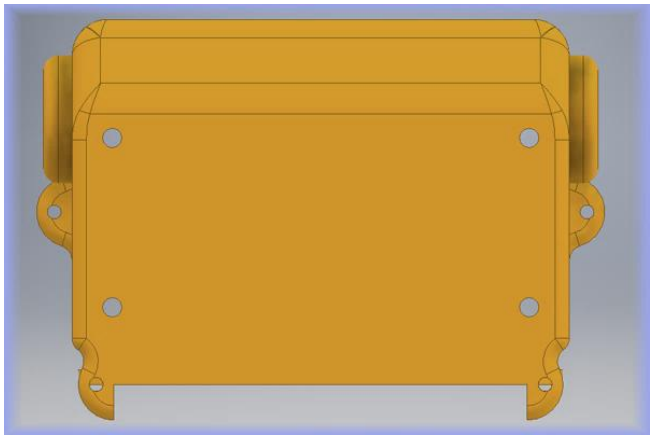
Figura 83. **Pieza 2, *pass* frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la parte trasera de la pieza 2. Esta parte contiene la parte superior de la tarjeta RFID y los agujeros para atornillar la Raspberry Pi Zero.

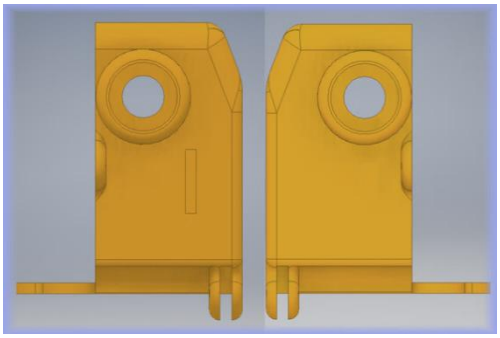
Figura 84. **Pieza 2, *pass* dorsal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra los laterales de la pieza 2. En estos laterales se encuentran ubicados los botones para interactuar directamente con el *pass*.

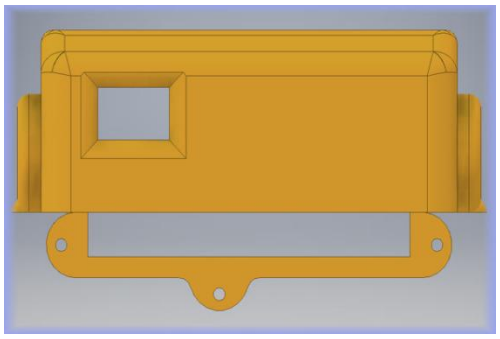
Figura 85. **Pieza 2, *pass* laterales**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la parte superior de la pieza 2. En esta parte se muestra un agujero que será útil solamente en las primeras pruebas del módulo completo. Este agujero es para descargar los datos manualmente o para alimentar la raspberry desde una batería externa.

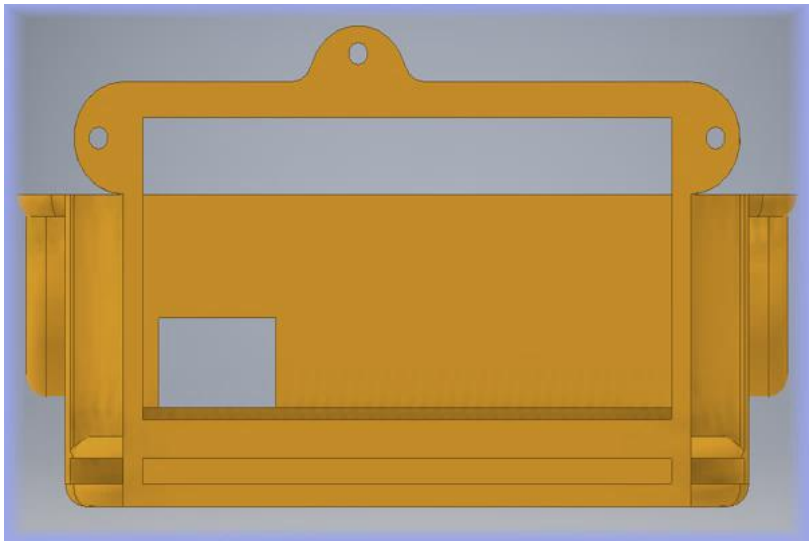
Figura 86. **Pieza 2, *pass* superior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la pieza 2 desde una vista posterior. Esta parte se ensambla directamente con la pieza 1, por lo cual el perímetro de esta pieza debe ser sellado con silicón de alta resistencia.

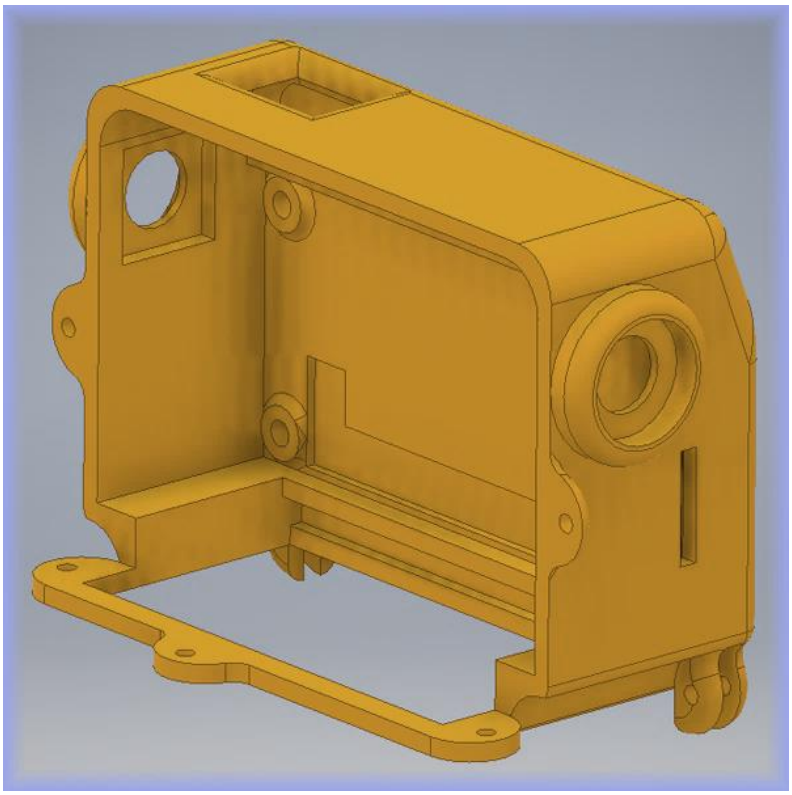
Figura 87. **Pieza 2, *pass* inferior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista angular de la pieza 2 desde la parte frontal superior derecha.

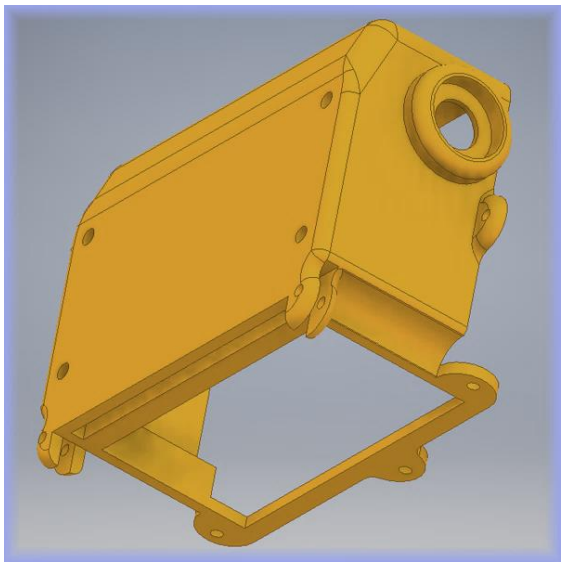
Figura 88. **Pieza 2, *pass* angular superior izquierda**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista posterior izquierda de la pieza 2.

Figura 89. **Pieza 2, *pass* angular dorsal inferior**



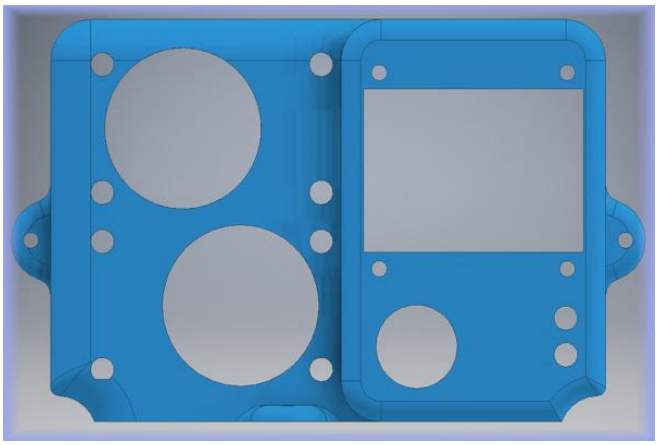
Fuente: elaboración propia, empleando Autodesk Inventor.

La pieza 3 del *pass* hace referencia a la parte donde se colocan los sensores de gases, el sensor de temperatura a distancia y las partes de visualización. Esta pieza complementa a la pieza 2 de este mismo módulo.

El motivo por el cual se hicieron separadas la pieza 2 de la pieza 3 es para tener un espacio abierto para cablear todos los dispositivos dentro del *pass*. Esta parte es fácil de colocar y quitar, solamente es por medio de 3 tornillos de fácil acceso, de igual manera es necesario sellar con silicón de alta resistencia el perímetro que hace contacto con la pieza 2.

La siguiente imagen muestra la pieza 3 desde una vista frontal. Esta parte es la que va expuesta al ambiente. En esta pieza se exponen los agujeros para ensamblar 2 leds de 3mm, 2 sensores de gases MQ, un sensor de temperatura a distancia y una pantalla OLED de 0,96 pulgadas.

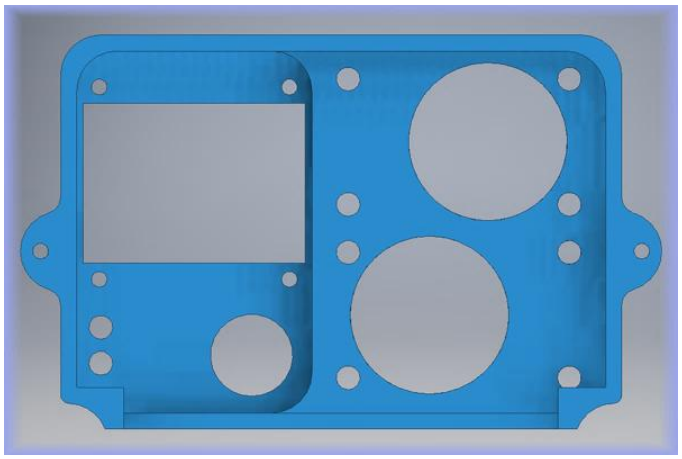
Figura 90. **Pieza 3, *pass* frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la vista trasera de la pieza 3. Esta parte es la que queda interna junto a los circuitos. En esta parte va pegado el módulo bluetooth detrás de la pantalla oled.

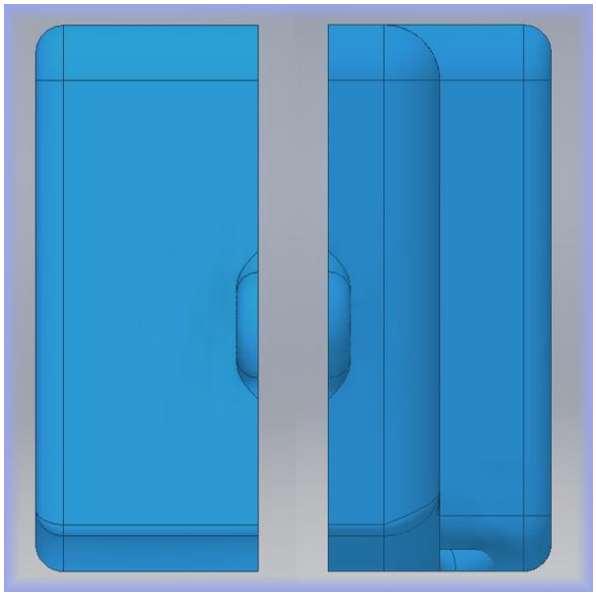
Figura 91. **Pieza 3, *pass* dorsal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra los laterales de la pieza 3. Los laterales de esta pieza solamente contienen las partes para ensamblar a la pieza 2.

Figura 92. **Pieza 3, *pass* laterales**

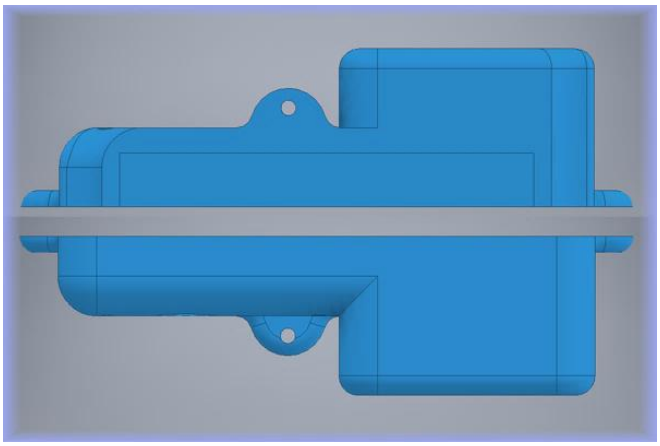


Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra la parte posterior y la parte superior de la pieza 3. Esta parte de igual manera tiene solamente los ensamblajes hacia la pieza 2.

Los motivos por los que el sensor de temperatura a distancia, la pantalla y los leds están sobresaltados a comparación de los sensores de gases es para que puedan estar al mismo perfil con los sensores ya colocados y, además, para que tenga un espacio seguro de instalación el módulo bluetooth detrás de la pantalla oled.

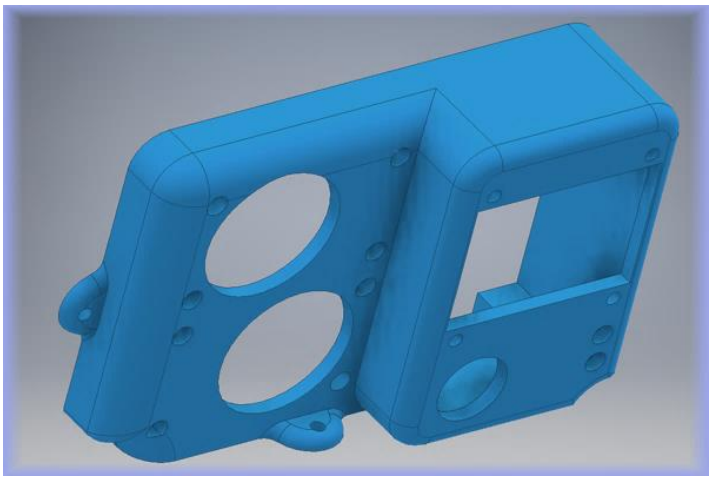
Figura 93. **Pieza 3, *pass* Superior y posterior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista frontal superior izquierda de la pieza 3.

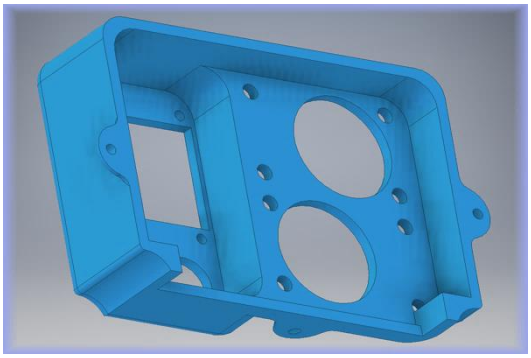
Figura 94. **Pieza 3, *pass* angular superior derecha**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista posterior derecha de la pieza 3.

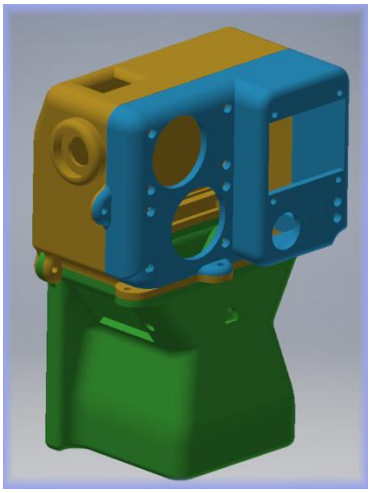
Figura 95. **Pieza 3, *pass* angular dorsal inferior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista superior izquierda del acople de todas las piezas del módulo de *pass*.

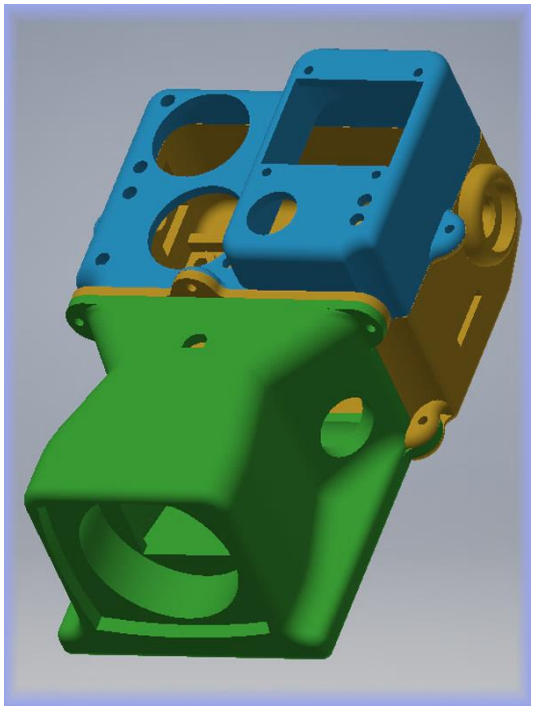
Figura 96. ***Pass* ensamblado vista angular frontal**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista inferior derecha del acople de todas las piezas del módulo de *pass*.

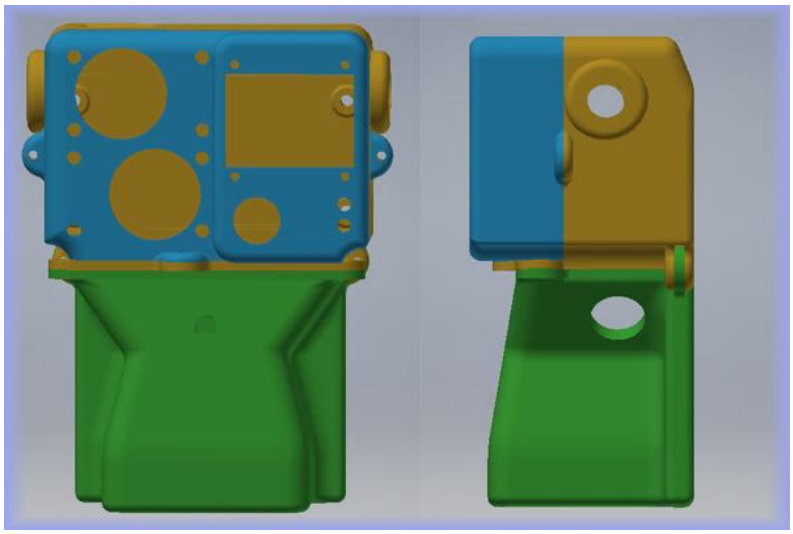
Figura 97. ***Pass* ensamblado vista frontal inferior**



Fuente: elaboración propia, empleando Autodesk Inventor.

La siguiente imagen muestra una vista frontal y una lateral del acople de todas las piezas del módulo *pass*.

Figura 98. ***Pass* ensamblado vista frontal y lateral**



Fuente: elaboración propia, empleando Autodesk Inventor.

3. SOFTWARE Y FIRMWARE

3.1. Conceptos básicos de programación

La programación en cuanto a desarrollo hoy en día, es uno de los procesos más importantes de un proyecto. Actualmente, la mayoría de desarrollos de hardware llevan a la par un desarrollo de software muy completo. La programación es un instrumento que sirve para plasmar las ideas de un desarrollador en un entorno informático para ejecutar tareas automatizadas desde procesos sumamente simples, hasta inteligencia artificial, teoría de grafos, etc.

De los conceptos de programación, se utilizan en este desarrollo la programación modular y estructurada. Es necesario mencionar que en futuras versiones será necesario tener un concepto de programación orientada a objetos, ya que las variaciones en cuanto a eficiencia de programación pueden llegar a ser mucho mayores.

La programación se define como un proceso de desarrollar, depurar y actualizar programas de computadoras mediante sentencias lógicas y matemáticas escritas en un lenguaje fácil de comprender para una persona, para luego ser traducido a lenguaje de máquina.

Los conceptos más básicos en común con todos los lenguajes de programación son los siguientes:

- Identificadores: son símbolos que sirven para nombrar elementos dentro de un programa y de la misma manera hacen referencia a los mismos elementos. Todos los lenguajes de programación cuentan con restricciones para utilizar sus identificadores.

- Palabras reservadas: Son términos que forman parte de un lenguaje de programación en específico, ya que las palabras reservadas de un lenguaje de programación difieren muchas veces de las de otro lenguaje, mientras que los símbolos si son comúnmente iguales o parecidos. Las palabras reservadas tienen significados gramaticales dentro del lenguaje, por lo que no pueden utilizarse como identificadores.

- Variables: Una variable es un espacio reservado normalmente en memoria RAM durante la ejecución de un programa. Una variable contiene valores o magnitudes susceptibles a alteraciones en el transcurso del tiempo de ejecución de un programa. Las variables tienen la característica de poseer una longitud definida, siendo esta proporcional a la cantidad de memoria ocupada durante la ejecución del programa. La longitud de las variables puede ser de 2 tipos:

    - Fija: posee un tamaño delimitado al declararla inicialmente dentro del programa.

    - Variable**:** el tamaño definido inicialmente varía dependiendo la cantidad de información almacenada en esta.

- Constantes: una constante es un espacio reservado normalmente en memoria RAM al igual que una variable, solamente que el valor de una

constante no se altera en ningún momento durante el tiempo de ejecución de un programa.

- Tipos de datos: todas las variables y constantes poseen un tipo de datos delimitado al momento de programar. Estos no son más que atributos que indican la clase de datos a manejar y a operarlos de diferente manera, teniendo de esta manera restricciones en los datos guardados dentro de cada tipo. Los tipos más comunes son:

    - Enteros (*integer*)
    - Flotantes (*float*)
    - Caracteres (*char*)
    - Cadenas de caracteres (*string*)
    - Arreglos de datos (*array*)

- Operadores de datos: son elementos del programa que se aplican a una o varias variables o constantes, por medio de los cuales se construyen expresiones o instrucciones. Los tipos más comunes son:

    - Aritméticos
    - Lógicos
    - Relacionales
    - Asignación
    - Dirección

- Estructuras de control: permiten al programador modificar el flujo de ejecución de un programa, haciendo que este ejecute repetidas veces un bloque de código o simplemente cambiar el puntero a otra dirección. Las estructuras de control son de 2 tipos:

- Selectivas: ejecutan instrucciones dependiendo de algunas condiciones.
- Iterativas: ejecutan instrucciones repetidamente mientras se cumpla una condición.

- Comentarios: los comentarios son anotaciones útiles para el programador. Estos tienen una sintaxis inicial y/o final para determinar el inicio y el final del comentario, ya que estos son transparentes para la ejecución del programa.

3.2. Paradigmas de programación

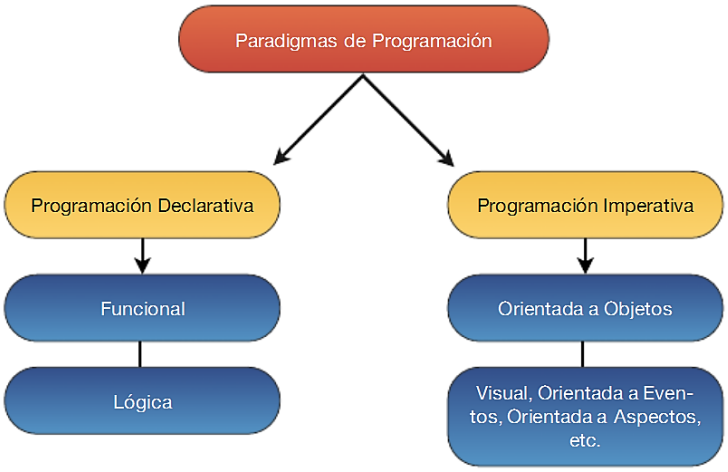
La programación es algo mucho más complejo que solo definir instrucciones para que un ordenador las ejecute, en realidad aparte de complejo es algo muy diverso, ya que existen muchos lenguajes de programación y a su vez, estos siguen en aumento. Cada lenguaje tiene sus propias características, estructura y definiciones. A esta diversidad mezclada con la manera de solucionar problemas en código se le conoce como paradigmas de programación.

Los paradigmas de programación representan un enfoque específico en cuanto a búsqueda de soluciones, son filosofías sugeridas por comunidades de programadores o desarrolladores para adoptar propuestas de resolución a problemas claramente definidos.

Los paradigmas de programación están delimitados temporalmente en cuanto a aceptación y utilidad, ya que surgen nuevos paradigmas en un momento indeterminado que ofrecen nuevas, mejores y optimas soluciones que sustituyen un paradigma completamente o parcialmente. Si un programador

desarrolla un código de programación basado en un paradigma específico poco cuestionable, es muy probable que en cierto tiempo ese paradigma se vuelva obsoleto, por lo cual, todo software debe de someterse a procesos de actualización.

Figura 99. **Paradigmas de programación**



Fuente: *Paradigmas de programación*. https://programacionluzedu.files.wordpress.com/2012/06/sin-tc3adtulo.png. Consulta: 4 de enero de 2018.

3.2.1. Paradigma imperativo

Este paradigma se basa específicamente en ordenar o ejecutar. Simplemente es ordenarle a una maquina lo que queremos que esta realice paso a paso. Describe un problema en términos de instrucciones de código para llegar a un resultado. En el caso de este tipo de paradigmas, al modificar los valores dentro de un programa se modifica el estado del mismo. Estos utilizan procedimientos o funciones impuras como mecanismos de control, los cuales normalmente generan efectos secundarios dentro de un programa, de igual manera los paradigmas imperativos se basan en funciones cíclicas, como

ciclos FOR, WHILE, LOOP, etc. Aquí existe una gran dificultad para tener procesos de programación paralelos o paralelización, debido a que la memoria es compartida entre muchos procesos más.

3.2.2. Paradigma orientado a objetos

Este paradigma es lo más parecido a nuestro entorno real como humanos, ya que nosotros percibimos todo en el mundo real como uno o varios conjuntos de objetos o cosas, y cada uno de estos objetos tiene sus propios atributos y métodos de uso. De esta misma manera se clasifican en código todas las partes de un programa.

3.2.3. Paradigma declarativo

Este paradigma describe un problema en términos de proposiciones y afirmaciones, sin ser sumamente especifico en los pasos necesarios para resolver el problema. En este caso, se evita que cambie el estado de un programa al prevenir los cambios en los valores internos del mismo. Los paradigmas declarativos utilizan funciones puramente matemáticas, las cuales evitan cualquier tipo de efecto secundario dentro de un programa. En estos no existen estructuras cíclicas como en los paradigmas imperativos, en estos casos la única forma de iterar es por medio de funciones recursivas de alto nivel, como MAP, REDUCE, FILTER, entre otros. Este tipo de paradigma esta optimizado para concurrencias y ejecuciones paralelas debido a que la memoria no se comparte entre otros procesos.

3.2.4. Paradigma funcional

Este paradigma se basa utilizando funciones puras, trasladando directamente conceptos matemáticos hacia la programación. Este concepto es utilizado comúnmente en programas donde se requieran análisis complejos de diseño, hojas de cálculo, etc.

3.2.5. Paradigma lógico

Este paradigma utiliza a menudo la lógica matemática, la cual es considerada la manera más sencilla para el intelecto humano de solucionar problemas. De esta manera se pueden expresar desde problemas sencillos hasta problemas muy complejos y resolverlos mediante aplicaciones de hipótesis, teoremas, reglas, etc.

3.3. Librerías

Una librería es un conjunto de código de programación que sirve para realizar uno o varios procesos específicos. Las librerías normalmente tienen un método de entrada de datos, un método de salida de datos y una interfaz de configuración. Normalmente, son utilizadas para simplificar la programación de un proceso especifico, ya que ha habido otras personas que han trabajado en ese proceso anteriormente.

Las librerías no son un control automatizado de código, sino simplemente una simplificación de código delimitado por objetos, funciones u otros. Son muy útiles en los casos en los que se necesita interactuar con un software o hardware no desarrollado por nosotros y poder dedicar el tiempo exclusivamente a la lógica del desarrollo deseado y no al funcionamiento

especifico de los mismos módulos. Las librerías pueden vincularse con un programa en diferentes puntos de ejecución según el vínculo.

3.3.1. Librerías de *Firmware* utilizadas

Las librerías de *firmware* han sido utilizadas directamente en el entorno de programación de Arduino IDE, que es un lenguaje C simplificado. Estas específicamente son para utilizar los sensores y la pantalla para visualizar datos en la careta. Aquí se muestran todas las librerías utilizadas tanto en la parte del firmware de la careta como la parte del *firmware* del pass:

- Adafruit_MLX90614.h: esta librería ha sido diseñada específicamente para el sensor de temperatura MLX90614, el cual es un sensor de temperatura ambiental y temperatura a distancia con calibración inclusive para uso médico. Esta librería no necesita la importación de otra librería para funcionar.

- DallasTemperature.h: esta librería ha sido diseñada específicamente para el sensor Dallas Sensor 18B20, el cual mide temperatura en una escala moderada de hasta 120 grados centígrados. Esta librería necesita la importación de la librería OneWire.h para poder funcionar, ya que esta otra contiene en código el protocolo de comunicación 1Wire, con el que funciona dicho hardware.

- Adafruit_SH1106.h: librería de hardware especifica que contiene los registros e instrucciones para interactuar con el hardware del display con controlador SH1106. Esta librería debe ir emparejada con la librería Adafruit_GFX.h para poder desarrollar una programación más sencilla,

ya que esta segunda librería contiene código para realizar gráficos básicos.

- Adafruit_Sensor.h: esta librería estandariza los módulos de sensores desarrollados por Adafruit, ya que relaciona cada uno de los sensores con toda la información y protocolos de comunicación necesarios para cada uno como objetos de esta misma librería. Esta librería es necesario importarla en cuanto utilicemos sensores y librerías de sensores desarrolladas por Adafruit.

- Adafruit_GFX.h: es la librería grafica utilizada principalmente para todas las pantallas comercializadas por Adafruit. Proporciona un conjunto común de gráficos sencillos como líneas, círculos, puntos, etc. Esta debe emparejarse con una librería específica del hardware utilizado, la cual contenga todas las instrucciones y registros para manejar un nivel inferior, que sería hardware. Esta librería contiene algunos recursos que son muy útiles al momento de graficar en una pantalla, por ejemplo, Image2Code, que es una herramienta de GUI de Java para convertir un archivo BMP en código matricial, así de esta manera poder mostrar una imagen monocromática por medio de una función drawBitmap.

- OneWire.h: esta librería contiene el código necesario para poder comunicar un sensor con protocolo 1Wire a un pin GPIO especifico. Este protocolo es desarrollado directamente por software, por lo cual no se requiere ningún pin con ningún protocolo definido por hardware para que este funcione.

- DHT_U.h: esta librería requiere el código necesario para poder utilizar sensores DHT, los cuales sirven para medir temperatura y humedad

relativa en el ambiente. Estos sensores no utilizan un pin definido por hardware para poder funcionar, y de igual manera que el protocolo OneWire, solamente necesita una línea de datos. A pesar de las semejanzas con el protocolo OneWire, ambos protocolos son totalmente diferentes y no se adapta uno al otro.

- Wire.h: esta librería permite una comunicación directa con dispositivos I2C/TWI. Esta librería requiere los pines definidos por hardware para el protocolo I2C para poder funcionar. Estos pines son los denominados SDA (*signal data*) y SCL (*signal clock*). En las placas Arduino se encuentran cerca del pin AREF.

- DHT.h: esta librería contiene el protocolo de comunicación directamente del hardware con el dispositivo de control. Esta librería debe emparejarse junto a la librería DHT_U.h para poder utilizar el sensor de una manera fácil y óptima.

- SPI.h: esta librería contiene el código necesario para poder comunicar un dispositivo SPI con un controlador. El protocolo de comunicación SPI (*serial peripheral interface*) si necesita directamente de 3 a 4 pines definidos por hardware para poder funcionar, los cuales son MISO, MOSI, SCK y SS.

3.3.2. Librerías de Software utilizadas

Las librerías de software han sido utilizadas en un lenguaje de programación llamado Python. Este lenguaje sirve en este desarrollo para realizar toda la lógica, análisis y almacenamiento de los datos obtenidos por los sensores. En este caso las librerías corren en el mismo lenguaje Python y se

instalan de manera manual e independiente cada una de las que no vienen instaladas en el sistema Linux.

- Adafruit_SSD1306: esta librería contiene todo el código necesario para poder utilizar una pantalla con controlador SSD1306. Esta librería si necesita protocolos definidos por hardware ya sea SPI o I2C para funcionar.

- Subprocess: este es el primero de los 3 módulos del sistema destacados que Python provee en una librería estándar. Esta librería permite trabajar directamente con órdenes del sistema operativo.

- Sys: este es el segundo de los 3 módulos del sistema destacados que Python provee en una librería estándar. Esta librería provee funcionalidades y variables estrictamente relacionados con el intérprete de Python o interprete de comandos de programación.

- Os: este es el tercero de los 3 módulos del sistema destacados que Python provee en una librería estándar. Este módulo permite acceder a funciones que dependen del sistema operativo utilizado, específicamente funciones que devuelven información sobre el entorno que se está trabajando, manipular datos, directorios, entre otros.

- MySQLdb: este más que una librería es un conector de base de datos que provee Python database API. Sirve simplemente para comunicar Python con una base de datos MariaDB o MySQL, y utilizar una sintaxis de selecciones o de ingresos tal y como se usa en MySQL.

- Threading: esta librería sirve para generar hilos de ejecución en Python. Los hilos de ejecución de un proceso comparten el mismo espacio que el hilo principal del programa. Esta librería agiliza el proceso de un programa, ya que ejecutar un proceso de varios hilos suele requerir menos recursos en cuanto a memoria que ejecutar el código equivalente sin hilos simplemente en procesos separados.

- Pigpio: esta librería interactúa directamente con el hardware de la raspberry, dejando en simples configuraciones y líneas de código el estado de los pines GPIO (*general purpose input/output*), protocolos de comunicación, interrupciones, entre otros.

- Difflib: esta librería proporciona clases y funciones para comparar secuencias, cadenas o tuplas y obtener una salida con todas las diferencias encontradas en las mismas. Difflib proporciona diferentes formatos de salida y resulta muy útil para comparar contenidos.

- Serial: esta librería contiene todo el código necesario para comunicarse con otro dispositivo por medio de un protocolo serial UART/USART. Dicha librería cuenta con todas las configuraciones necesarias y necesita los pines Rx y Tx definidos por hardware para poder funcionar.

- Time: esta librería contiene el código para utilizar la fecha y hora en todas sus permutaciones y formatos posibles en caso necesitemos ambas o parte de estas en el programa.

3.4. Firmware

El IDE de programación para el *firmware* es conocido como Arduino IDE, este es utilizado con un lenguaje basado en C++ que se puede denominar como C simplificado.

El *firmware* en ambas partes (careta y pass) fue desarrollado en Arduino, ya que este desarrollo es un prototipo para implementación de nuevas tecnologías en bomberos; este desarrollo inicial es simplemente para determinar cuál será la primera versión en cuanto a las necesidades vistas a partir de pruebas de campo. Las siguientes versiones se recomienda hacerlas con microcontroladores PIC en el IDE principal de desarrollo de estos mismos, llamado MPLAB, cuyos compiladores son XC8 y XC16.

Arduino es muy útil en cuanto a facilidad de programación en versiones de prueba, versiones que no se requieran con demasiada estabilidad o cuestiones que no necesiten una frecuencia muy alta. En este caso es muy útil por 2 motivos principales: primero por tiempo de desarrollo de prueba, ya que el tiempo de programación en Arduino es mucho menor al tiempo de programación que lleva un microcontrolador PIC, y segundo la facilidad de uso, ya que se evita estar colocando múltiples dispositivos solamente para programar o hacer que funcione el microcontrolador.

Otros aspectos tomados como modificaciones en instrucciones del *firmware* de dispositivos es la configuración de dispositivos por comandos AT, por ejemplo, los dispositivos de comunicación HC-05.

3.4.1. Protocolos de comunicación

Los protocolos de comunicación son sistemas que cuentan con las restricciones y reglas necesarias para poder trasladar información de un punto a otro. En electrónica y programación, estos protocolos son sumamente útiles, ya que sin ellos no podríamos interactuar de un circuito a otro, interactuar con sensores o simplemente no se tendrían maneras universales de transferencia de datos, por lo cual desarrollar hardware sería mucho más difícil de lo que es hoy en día. Los protocolos de comunicación utilizan formatos muy bien definidos para enviar y recibir mensajes. Los protocolos de comunicación utilizados en este desarrollo son seriales, y son los siguientes:

3.4.1.1. Protocolo UART/USART

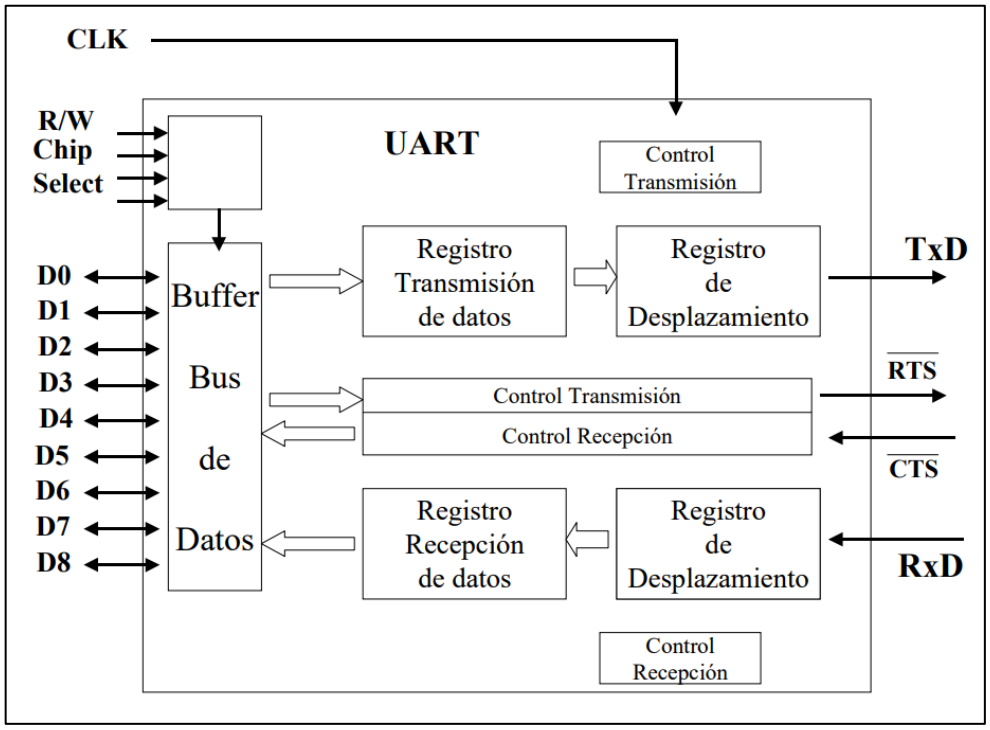
UART son las siglas en inglés de universal asynchronous receiver transmitter, lo cual significa transmisor-receptor asíncrono universal. Este protocolo requiere de un chip embebido en el hardware para ser más eficiente, el cual en la actualidad el más utilizado es el chip UART16550, que soporta velocidades de hasta 921,6 Kbps.

La diferencia entre UART universal asynchronous receiver transmitter y USART universal synchronous asynchronous receiver transmitter es que USART puede trabajar en modo síncrono o asíncrono, y debido a esto puede configurarse en él una velocidad más alta (más baudios). UART es un protocolo que, en principio basa su envío y recepción de datos a partir de 2 buses, los cuales son Tx (*transmitter*) y Rx (*receiver*). Y su fin es convertir datos de serie a paralelos cuando se trata de recepción y de convertir datos paralelos a serie cuando se trata de transmisión.

En un diagrama general del protocolo UART, se tienen registros de datos tanto de recepción como de transmisión con sus correspondientes registros de desplazamiento, se tienen registros de control de transmisión y recepción de señales de sincronización para iniciar la transmisión o recepción (RTS, CTS).

A pesar de que la transmisión realmente es síncrona, normalmente se le conoce como asíncrona para hacer referencia a que existe flexibilidad para la transmisión de las palabras de datos. La siguiente imagen muestra un diagrama de bloques del protocolo UART.

Figura 100. **Diagrama de bloques del protocolo UART**

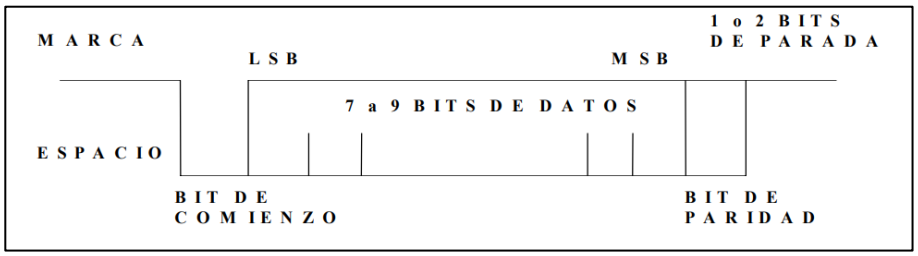


Fuente: *Protocolo UART*. http://www.el.uma.es/marin/Practica4_UART.pdf. Consulta: 4 de enero de 2018.

La sincronización en la transmisión de datos se realiza colocando un bit de inicio, denominado *Start Bit*, seguido de la trama de bits de datos la cual, normalmente es entre 5 y 9 bits iniciando por el bit menos significativo (LSB). Este proceso de envío se termina enviando un bit de parada, denominado *Stop Bit*.

Los protocolos UART que trabajan con 8 bits normalmente añaden un bit de paridad, el cual se pone en alto cuando la sumatoria de bits altos es par o viceversa. De esta manera un bit de paridad puede funcionar como un mecanismo de bajo procesamiento que funciona para detectar errores. Este bit es útil en buses de transmisión de datos que tengan una probabilidad de error muy baja. La siguiente imagen muestra un esquema de la transmisión y recepción asíncrona de un protocolo UART de ejemplo.

Figura 101. **Esquema de la transmisión y recepción asíncrona de un protocolo UART**



Fuente: *Protocolo UART*. http://www.el.uma.es/marin/Practica4_UART.pdf. Consulta: 4 de enero de 2018.

La tasa de transmisión de datos en un protocolo UART se mide en baudios (bauds), lo cual no es más que el número de bits que se transmiten en un segundo (bps). Los bits por segundo se pueden obtener a partir de cualquier valor, no existe restricción teórica que no permita esto, la restricción de

configuración de bps viene directamente a partir de tasas estandarizadas: 110, 150, 300, 600, 900, 1 200, 2 400, 4 800, 9 600, 14 400, 19 200, 28 800, 38 400, 57 600, 76 800, 115 200, 230 400, 250 000, etc. Estas tasas de bps se van incrementando con el tiempo mientras los dispositivos se van haciendo cada vez más rápidos. En un UART, la tasa de baudios se define a partir de la relación entre baudios, lo que es igual a la frecuencia de reloj(fi) interno y externo(CLKi) y n es un valor entre 0X00 y 0XFF ecuación mostrada en la siguiente imagen.

$$baudios = \frac{f_i}{16 * (n+1)}$$

La siguiente imagen muestra la relación entre frecuencia de reloj, tasa de baudios y probabilidad de error.

Tabla IV. **Relación entre frecuencia de reloj, tasa de baudios y probabilidad de error**

Baud Rate (bps)	f_{osc} = 16.0000MHz				f_{osc} = 18.4320MHz				f_{osc} = 20.0000MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	416	-0.1%	832	0.0%	479	0.0%	959	0.0%	520	0.0%	1041	0.0%
4800	207	0.2%	416	-0.1%	239	0.0%	479	0.0%	259	0.2%	520	0.0%
9600	103	0.2%	207	0.2%	119	0.0%	239	0.0%	129	0.2%	259	0.2%
14.4K	68	0.6%	138	-0.1%	79	0.0%	159	0.0%	86	-0.2%	173	-0.2%
19.2K	51	0.2%	103	0.2%	59	0.0%	119	0.0%	64	0.2%	129	0.2%
28.8K	34	-0.8%	68	0.6%	39	0.0%	79	0.0%	42	0.9%	86	-0.2%
38.4K	25	0.2%	51	0.2%	29	0.0%	59	0.0%	32	-1.4%	64	0.2%
57.6K	16	2.1%	34	-0.8%	19	0.0%	39	0.0%	21	-1.4%	42	0.9%
76.8K	12	0.2%	25	0.2%	14	0.0%	29	0.0%	15	1.7%	32	-1.4%
115.2K	8	-3.5%	16	2.1%	9	0.0%	19	0.0%	10	-1.4%	21	-1.4%
230.4K	3	8.5%	8	-3.5%	4	0.0%	9	0.0%	4	8.5%	10	-1.4%
250K	3	0.0%	7	0.0%	4	-7.8%	8	2.4%	4	0.0%	9	0.0%
0.5M	1	0.0%	3	0.0%	–	–	4	-7.8%	–	–	4	0.0%
1M	0	0.0%	1	0.0%	–	–	–	–	–	–	–	–
Max.(1)	1Mbps		2Mbps		1.152Mbps		2.304Mbps		1.25Mbps		2.5Mbps	

Fuente: *Módulo*. https://arduino.stackexchange.com/questions/18132/setting-non-standard-baud-rate-250k-for-the-hc-05-bluetooth-module. Consulta: 4 de enero de 2018.

3.4.1.2. Protocolo SPI

SPI significa serial peripherical interface, este protocolo es uno de los más populares para trabajar en comunicación serial de hardware a hardware debido a su velocidad de transmisión y lo simple que es para configurarse. Gracias a estas ventajas, muchos dispositivos de hardware tienen un control SPI incluido para comunicación.

SPI es un protocolo que, a diferencia del UART, este es síncrono. De igual manera este protocolo trabaja en full dúplex; es decir, es capaz de mantener una comunicación bidireccional enviando y recibiendo datos de manera simultánea por multiplexación de frecuencias en un mismo bus de transmisión o por dos líneas separadas. Al ser un protocolo síncrono, se cuenta con un bus adicional con pulsos de reloj.

En este protocolo se debe definir un dispositivo como maestro y uno o varios dispositivos como esclavos. El dispositivo maestro es el que envía información a los demás dispositivos dando órdenes de la información que el maestro requiere. Los dispositivos esclavos se encargan de recibir las órdenes del maestro y enviarle a este la información que requiere. Para este proceso se requieren dos registros de desplazamiento, uno para el maestro y uno para el esclavo utilizado respectivamente, ya que estos se encargan de almacenar los bits de manera paralela para realizar una conversión paralela-serial para transmitir los datos.

En SPI, existen 4 buses lógicos que se encargan de todo el proceso realizado por el protocolo, estas líneas son las siguientes:

- MOSI (*Master output slave input*), esta línea del bus sirve para enviar los datos convertidos en bits desde el dispositivo maestro hacia los dispositivos esclavos.

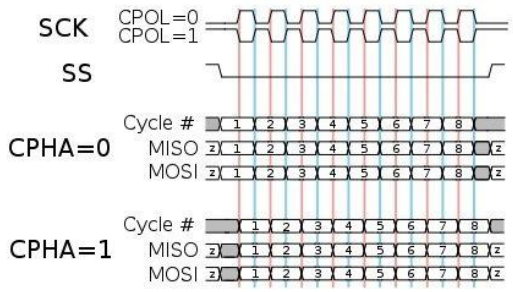
- MISO (*master input slave output*), esta línea del bus sirve para enviar los datos convertidos en bits desde los dispositivos esclavos hacia el dispositivo maestro.

- CLK (*clock*), esta línea del bus sirve para enviar una señal de reloj desde el dispositivo maestro hacia los esclavos. Esta línea sirve para sincronizar todos los dispositivos.

- SS (*slave select*), esta línea se encarga de seleccionar un dispositivo a la vez y habilitarlo, mientras mantiene deshabilitados los demás dispositivos.

La siguiente imagen muestra una representación de sincronización de distintos modos de SPI.

Figura 102. **Sincronización de distintos modos de SPI**



Fuente: *Como funciona el protocolo*. http://panamahitek.com/como-funciona-el-protocolo-spi/. Consulta: 4 de enero de 2018.

En el protocolo SPI, existen 4 modos de transmisión de información, los cuales son a partir de las combinaciones posibles entre polaridad y fase. Los modos son los siguientes:

- Modo 0: CPOL=0, CPHA=0. en este modo, el estado del reloj permanece en estado lógico bajo y la información se envía en la transición de bajo a alto. A esto se le conoce como alto activo.

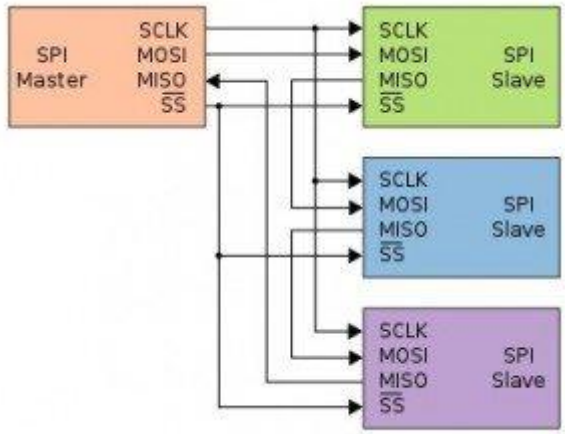
- Modo 1: CPOL=0, CPHA=1. en este modo, el estado del reloj permanece en estado lógico bajo y la información se envía en cada transición de alto a bajo. A esto se le conoce como bajo activo.

- Modo 2: CPOL=1, CPHA=0. en este modo, el estado del reloj permanece en estado lógico alto y la información se envía en cada transición de bajo a alto o en alto activo.

- Modo 3: CPOL=1, CPHA=1. en este modo, el estado del reloj permanece en estado lógico alto y la información se envía en cada transición de alto a bajo o bajo activo.

La siguiente imagen muestra los diagramas de conexiones correspondientes al protocolo SPI.

Figura 103. **Diagramas de conexiones correspondientes al protocolo SPI**



Fuente: *Como funciona el protocolo*. http://panamahitek.com/como-funciona-el-protocolo-spi/. Consulta: 4 de enero de 2018.

3.4.1.3. Protocolo I^2C

I2C significa *inter integrated circuit*, es un protocolo de comunicación serial desarrollado para poder comunicar varios dispositivos al mismo tiempo. Resumidamente, el protocolo I2C integra lo mejor del protocolo UART con lo mejor del protocolo SPI, lo cual significa que se puede tener un dispositivo maestro controlando muchos dispositivos esclavos con un límite muy superior a la cantidad de dispositivos que puede controlar SPI, y de igual manera, simplificando la cantidad de líneas de transmisión de datos. Este protocolo es muy útil cuando se utilizan varios microcontroladores para almacenar un registro de datos hacia una sola memoria.

Este protocolo se caracteriza por ser un protocolo síncrono, motivo por el cual deberá tener una línea de reloj aparte de la línea de datos. Dado que se busca como reducir la cantidad de líneas en el bus, la línea de datos es

semibidireccional, ya que puede transmitir datos desde el maestro al esclavo o viceversa, pero no simultáneamente. Esto se cree que puede ser una limitación, pero en realidad este bus se utiliza en aplicaciones donde la bidireccionalidad no es estrictamente necesaria.

Este protocolo de igual manera debe admitir topologías multipunto, esto se refiere a que se pueden conectar al bus múltiples dispositivos pudiendo actuar como emisores de datos, pero no todos emitiendo a la vez o pudiendo actuar todos los dispositivos como emisores y receptores en distintos momentos.

Otra gran característica de este protocolo es la admisión de tecnologías multimaestro, lo cual significa que un dispositivo puede actuar tanto como maestro o como esclavo; o, dicho de otra forma, los maestros pueden actuar como transmisores o receptores al igual que los esclavos. En este caso la única diferencia entre un maestro y un esclavo es la capacidad de gobernar el bus, mas no existen otras diferencias. En caso se intente manejar con más de un maestro el bus, este establece un mecanismo de arbitraje.

Existen 2 buses lógicos que se encargan de todo el proceso realizado por el protocolo, estas líneas son las siguientes:

- SDA (*serial data*): esta línea se encarga de comunicar y transmitir los datos entre maestro y esclavo.

- SCL (*serial clock*): esta línea se encarga de transmitir la señal de reloj desde el maestro hacia los esclavos. En el maestro esta señal es una salida, mientras que en los esclavos esta señal es una entrada.

Los estados de este protocolo son 5, los cuales son los siguientes:

- Libre: estado en el que lógicamente los buses SDA y SCL se encuentran en alto sin realizar ningún tipo de actividad.

- Inicio: este estado se produce cuando un maestro inicia una transacción, esta transacción es un cambio directo en el bus SDA de alto a bajo, mientras SCL sigue en estado alto sin alterarse. A partir de este cambio, se considera que el bus se encuentra ocupado y ningún otro maestro intentara generar la condición de inicio.

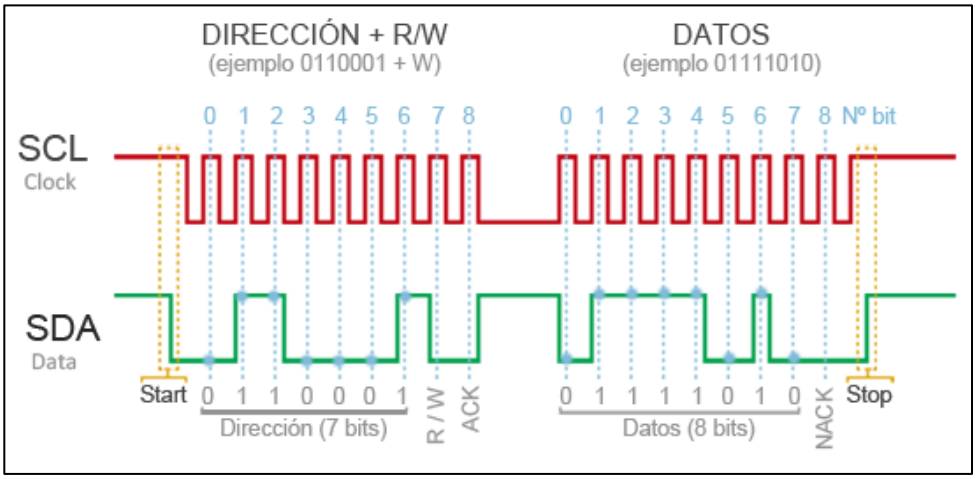
- Cambio: este estado se produce cuando la línea SDA puede cambiar de estado lógico mientras la línea SCL se mantiene en estado lógico bajo. Este es el único momento en el que el dispositivo seleccionado a transmitir, podrá colocar la información bit por bit en el bus para la transferencia.

- Dato: este estado se produce cuando, dado el estado de inicio, se define con estado lógico alto la señal de sincronía SCL. En este punto se considera que todo dato emitido tiene validez y no se puede cambiar. El único momento en el que es posible el cambio es en el estado de cambio.

- Parada: este estado final se produce cuando, estando SCL en estado lógico alto, se produce un cambio de estado lógico bajo a alto en SDA. Esto en realidad es una violación al estado de dato, y es precisamente por esto que se utiliza como fin de transmisión en el bus.

La siguiente imagen muestra un ejemplo de una trama de datos en un bus I2C.

Figura 104. **Trama de datos en un bus I2C**



Fuente: *Arduino-I2C*. https://www.luisllamas.es/arduino-i2c/. Consulta: 4 de enero de 2018.

3.4.1.4. Protocolo *One Wire*

One Wire es un protocolo de comunicación serie basado en un bus que tiene un maestro y varios esclavos en una sola línea de datos sin contar la tierra común necesitada por todos los protocolos para establecer una misma referencia. Una gran desventaja de este protocolo es la baja velocidad de transmisión de datos.

En este tipo de comunicación, cada dispositivo en el bus tiene un número de identificación único e inalterable de 64 bits, y la comunicación se produce en espacios de tiempo de 60 microsegundos. Aquí el maestro inicia y controla toda

comunicación en el bus y los esclavos se sincronizan con el reloj del maestro a través de la misma línea de datos.

En el caso de One Wire, la línea de datos/alimentación requiere una resistencia Pull-Up conectada a la alimentación del dispositivo.

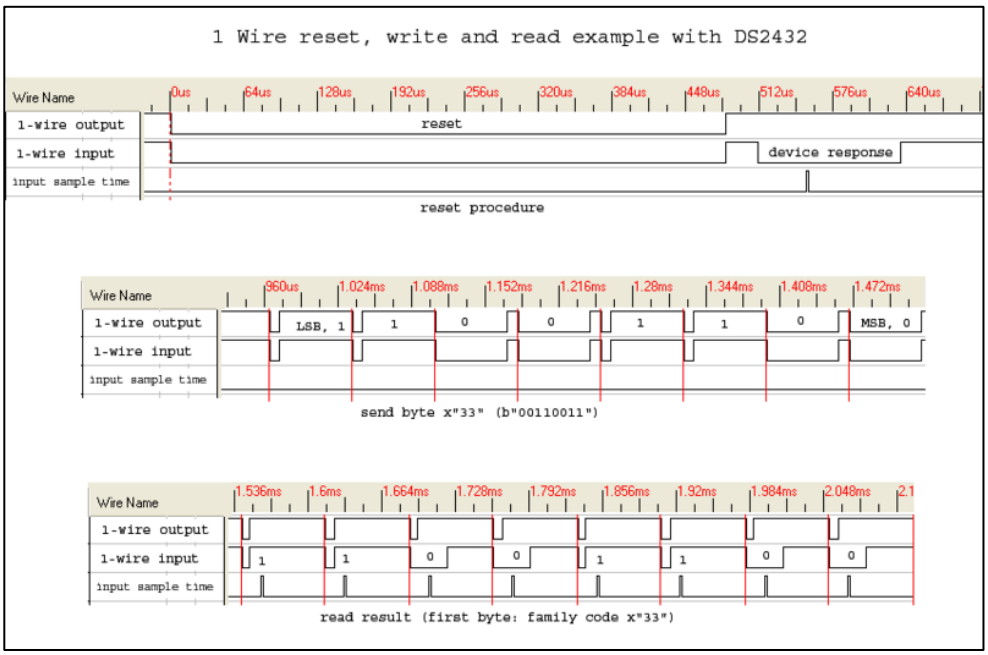
Este protocolo consta de 2 partes:

- Reinicio del bus: en este punto, se mantiene la señal en el bus de datos a 0 voltios durante 480 microsegundos para reiniciar todos los dispositivos conectados al bus. De igual manera, los dispositivos reiniciados indican de nuevo su conexión manteniendo la señal de datos a 0 voltios durante 60 microsegundos.

- Envío y recepción de datos: para transmitir un bit en estado alto, se lleva a un estado lógico bajo el bus de datos durante 1 a 15 microsegundos, mientras que, para transmitir un bit en estado bajo, se lleva a 0 voltios el bus de datos durante 60 microsegundos.

La siguiente imagen muestra un ejemplo de transmisión y recepción de datos del protocolo One Wire.

Figura 105. **Transmisión y recepción de datos del protocolo One Wire**



Fuente: *Transmisión y recepción de datos del protocolo One Wire*. https://en.wikipedia.org/wiki/1-Wire. Consulta: 4 de enero de 2018.

3.4.2. Filtros digitales

Un filtro digital es un bloque de código, ya sea sobre un hardware a nivel de firmware embebido o como parte de un desarrollo de software que opera directamente sobre señales discretas y cuantizadas. Estos dependen de las variaciones de las señales de entrada en función del tiempo (frecuencia) y amplitud. Varían la salida respecto a la entrada en función de amplitud, frecuencia o fase.

Los filtros son comúnmente utilizados en sensores para eliminar ruido, drift u otros errores de una señal leída, dejando como salida una función más

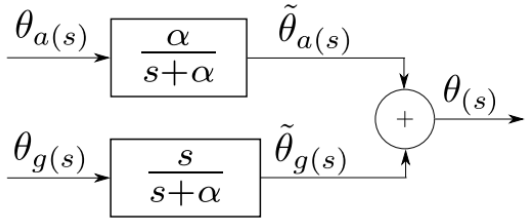
estable y más adecuada al uso que se necesita. Los filtros digitales utilizados son los siguientes:

3.4.2.1. Filtro complementario

El filtro complementario es la unión de dos filtros independientes, un filtro pasa altos (*high pass filter*), y un filtro pasa bajos (*low pass filter*). En el caso del MPU6050 que es en el cual se utilizó, se aplica el filtro pasa altos al giroscopio y el filtro pasa bajos al acelerómetro. El primero de estos deja pasar únicamente los valores arriba del límite propuesto por el filtro, mientras que el segundo deja pasar los valores debajo del límite. La ecuación resultante al combinar ambos procesos de los filtros es la de la siguiente imagen.

$$Ángulo = 0{,}98 * \left(Ángulo + AnguloGyro \ \cdot \ \Delta t\right) + 0{,}02 \ \cdot AnguloAccel$$

Donde AnguloGyro es el ángulo del giroscopio calculado previamente al cálculo, y AnguloAccel es el cálculo del ángulo obtenido por el acelerómetro previamente. El diferencial de tiempo hace relación al tiempo que ha pasado a partir de la última vez que se calculó el filtro (en segundos). Los valores de 0.98 y 0.02 en resumen son valores proporcionales al ángulo del giroscopio y del acelerómetro para tener un mayor énfasis en algunos resultados. Estos valores se pueden variar, solamente se deben dejar constantes en las mediciones y ambos valores deben sumar 1. La siguiente imagen muestra el diagrama de bloques referente al filtro complementario.



3.4.2.2. Filtro de media móvil

Estadísticamente, la media es un valor intermedio de un conjunto de valores finitos denominado promedio. La media móvil por lo tanto es un cálculo que sirve para analizar un conjunto de datos para crear series de valores promedio. La media móvil es una sucesión de números de los cuales, cada valor es un promedio de un rango de números dependientes. Una serie de media móvil puede ser calculada para cualquier serie de números temporal.

Un filtro de media móvil se utiliza normalmente para estabilizar señales sin tendencia ni estacionalidad. Sirve para suavizar fluctuaciones de plazos cortos, resaltando de esta manera las tendencias o ciclos de plazos largos.

Tipos de media móvil

- Media móvil simple (*moving average*):
    - Media móvil previo: es simplemente la media aritmética de n datos anteriores. Mientras más grande sea n, mayor será la influencia de datos antiguos. Por otro lado, mientras más pequeño sea n, más influencia tendrán los datos recientes.
    - Media móvil central: no utiliza solamente datos anteriores, sino también datos posteriores al que se quiere analizar.
- Media móvil ponderado (*weighted moving average*): es una media multiplicada por algunos factores proporcionales que darán determinada importancia y determinada visibilidad a ciertos valores necesarios. Esta desarrolla y mejora las aplicaciones de la media móvil simple. Este sirve

para adaptar rápidamente el valor predicho de las fluctuaciones en datos recientes.

- Media móvil exponencial: es una media móvil ponderada exponencialmente. La ponderación de los datos más antiguos decrece exponencialmente pero nunca llega a un valor cero.

Si se aplican los conceptos anteriores a un filtro digital, se obtienen lecturas más fiables y limpias a un plazo proporcionalmente distante según los criterios aplicados. En la mayor cantidad de veces que se utilicen IMUs (*inertial medition unit*), se necesita filtrar las señales ya que estos sensores son muy inestables en presencia de los diferentes tipos de ruido.

En el caso de filtro digital de media móvil es muy simple de aplicar, ya que se utiliza una media móvil simple, basándose simplemente en promediar todos los datos obtenidos desde el inicio de la medición requerida hasta el punto en un tiempo t. Dadas las ventajas de este filtro en cuanto a eficiencia y sencillez, sus desventajas principales son relacionadas con la debilidad del uso de la media como estimador de una tendencia. La sencillez de este filtro proviene de la siguiente ecuación.

$$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k]$$

- Donde Y es la variable dependiente, en este caso la media obtenida, M es el valor total de datos muestreados dentro del filtro, k es la variable que relaciona la sumatoria de cada uno de los valores desde cero hasta la posición M-1, n es el valor muestreado en un tiempo t, y x es la variable independiente o valor muestreado. La ecuación dado que realiza

operaciones relacionadas a base de valores pasados genera un retraso, y este retraso solamente es dependiente de M.

3.4.2.3. Gráficos de filtros

Normalmente, los filtros digitales son considerados para la mayoría de sensores. En este caso, han sido aplicados solamente para el acelerómetro, giroscopio, sensor de temperatura del MPU6050 y sensores de gases, ya que los demás sensores se muestran lo suficientemente estables en cuanto a variaciones y sus mediciones son muy lentas en comparación a los demás, lo cual, dependiendo el tipo de filtro, puede retrasar mucho una lectura necesitada en un momento preciso. En estos casos lo que se ha hecho es sacar un promedio de los últimos datos (similar al filtro de media móvil) pero siempre mostrando de igual manera el dato real medido.

Los gráficos mostrados a continuación son referentes a los datos de *roll*, *pitch* y *yaw* provenientes del acelerómetro y giroscopio en el IMU MPU6050. *roll*, *pitch* y *yaw* hacen referencia a los ejes X, Y, Z, respectivamente.

En cuanto al MPU6050, las mediciones brindadas están llenas de impurezas (ruido, drift, etc.) y si es necesario filtrarlas. Para filtrar estas señales, es necesario antes normalizar las mediciones a medidas estándar entendibles para una persona. En el caso del acelerómetro y giroscopio, es convertir el rango de valores recibidos a grados decimales. Este cálculo se hace por medio de una tangente inversa en el caso del acelerómetro, y una división sobre magnitud de datos en el caso del giroscopio.

$$AnguloY = atan\left(\frac{x}{\sqrt{y^2 + z^2}}\right)$$
$$AnguloX = atan\left(\frac{y}{\sqrt{x^2 + z^2}}\right)$$

Luego de esta conversión de datos, ya se puede aplicar cualquier filtro. En este caso se aplicó un filtro de media móvil y un filtro complementario para hacer comparativas. Las diferencias observadas entre ambos filtros son muchas, pero principalmente se mencionan algunas.

- El filtro de media móvil es útil para valores únicos que no dependen de otras variables. Dígase el caso de la temperatura, presión, entre otros. En este filtro, la señal se ve retrasada proporcionalmente al tamaño de la ventana de datos que se operan, de igual manera la amplitud de algunas señales se ve afectada dependiendo la proporción con la que se multiplique el filtro.

- El filtro complementario es útil para valores que dependan de más de una variable para poder estabilizar este valor. Este es el caso del MPU6050, ya que un ángulo se define mejor obteniendo en conjunto los datos del acelerómetro y del giroscopio. En el caso del filtro complementario, que es la unión de un filtro pasa altos y un filtro pasa bajos, el filtro pasa altos es referente al giroscopio y el filtro pasa altos es referente al acelerómetro.

- El filtro de media móvil tarda un lapso proporcional a la ventana para estabilizarse por completo, ya que los datos procesados desde una lectura 0 hasta n, donde n es la longitud de la ventana, deben promediarse entre su posición, ya que si se promedian entre la longitud de la ventana obtendremos un gráfico que tiende al infinito.

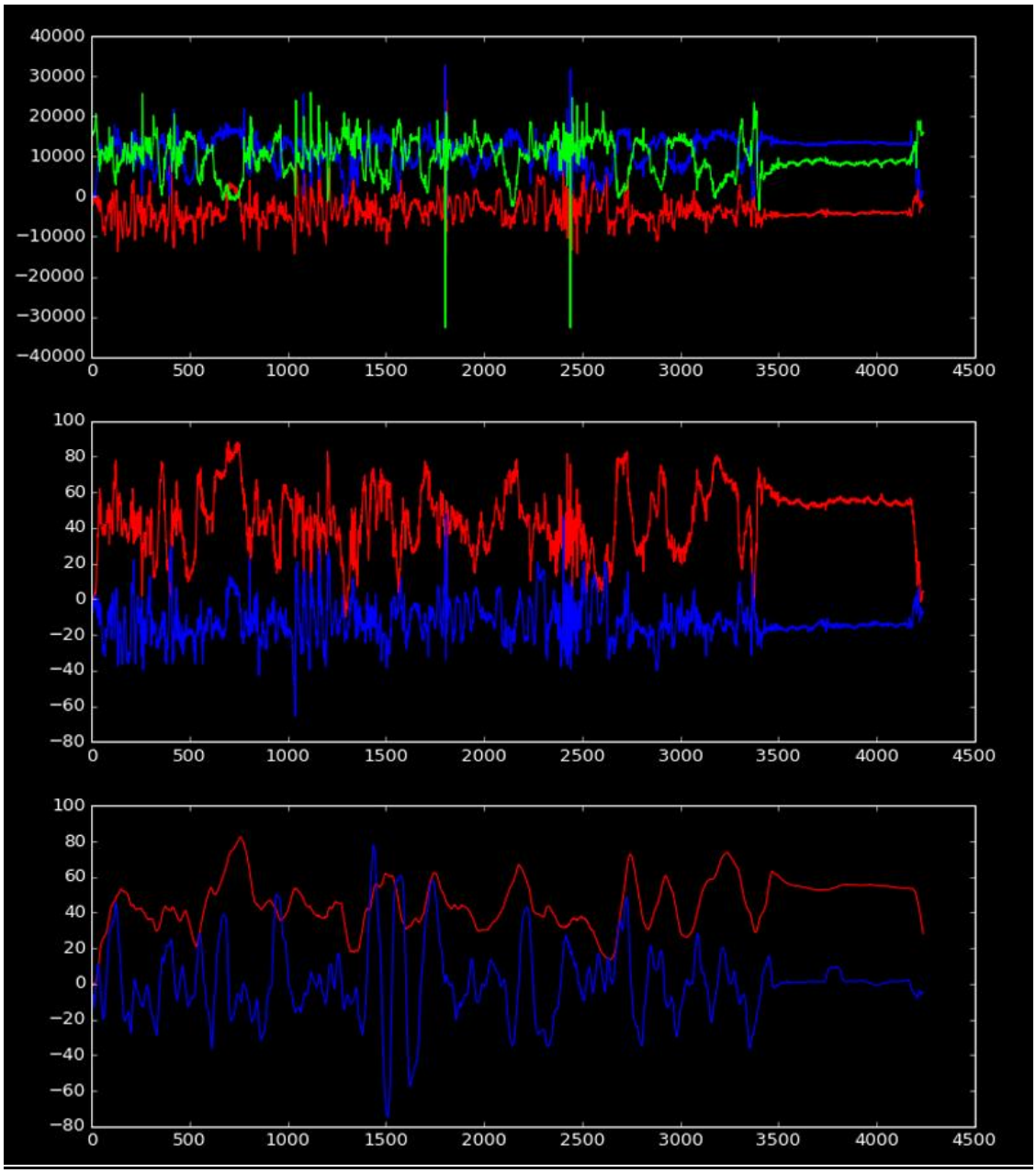
- Es posible para una calibración más precisa, aplicar un filtro de media móvil con una ventana muy baja a los datos del acelerómetro y giroscopio, para luego aplicarle un filtro complementario a ambos, así de

esta manera obtener una señal más limpia, pero un poco atrasada en el tiempo.

A continuación, se muestran las mediciones del MPU6050 completas medidas en un lapso de 3 minutos. Estas mediciones son reales del sensor ya colocado en una careta de bomberos y son el tipo de gráficos esperados en un ambiente de alto estrés y movimiento. *yaw* solamente se muestra entre los datos obtenidos del sensor, ya que no es útil en el resto del proceso. En caso de necesitar el eje Z más adelante, que es el que se nombra como *yaw*, se actualizaría al MPU9250, ya que para este eje es necesario un magnetómetro.

En la siguiente imagen se muestran los datos recibidos directamente del acelerómetro. La primera gráfica muestra *roll* (rojo), *pitch* (azul) y *yaw* (verde) en las magnitudes y escalas provenientes del sensor. La segunda fila muestra los datos ya normalizados de *roll* y *pitch*, en un rango de -90 a 90 grados cada eje. La tercera fila muestra los datos ya filtrados de *roll* y *pitch*. En este caso, los datos filtrados son por medio de un filtro de media móvil dejando independientes ambas mediciones.

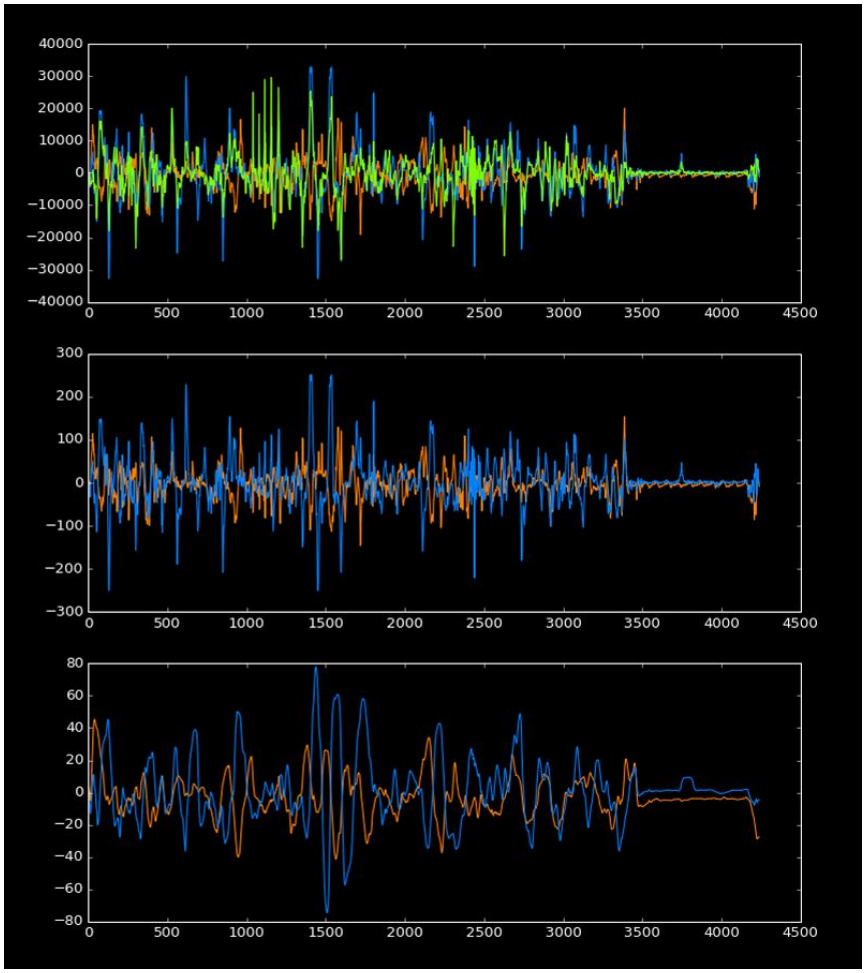
Figura 106. **Datos recibidos directamente del acelerómetro**



Fuente: elaboración propia.

En la siguiente imagen se muestran los datos recibidos directamente del giroscopio. La primera grafica muestra *roll* (naranja), pitch (celeste) y yaw (verde) en las magnitudes y escalas provenientes del sensor. La segunda fila muestra los datos ya normalizados de *roll* y *pitch*, en un rango de -90 a 90 grados cada eje. La tercera fila muestra los datos ya filtrados de *roll* y *pitch*. En este caso, los datos filtrados son por medio de un filtro de media móvil dejando independientes ambas mediciones.

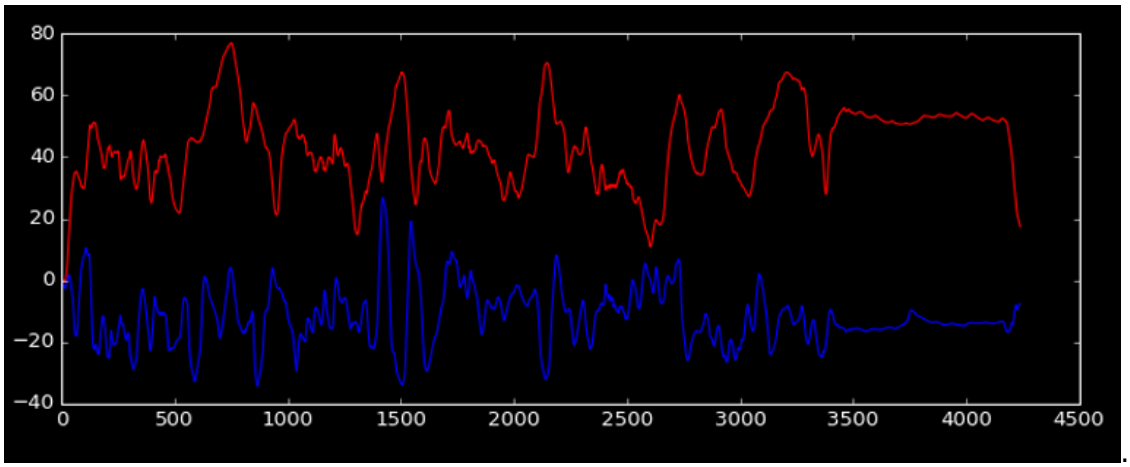
Figura 107. **Datos recibidos directamente del Giroscopio**



Fuente: elaboración propia.

La siguiente imagen muestra los datos normalizados aplicando un filtro complementario. En este caso se muestran solo 2 resultados, uno de *roll* y uno de *pitch*, ya que el filtro unifica los resultados de ambos sensores como un solo filtro pasa banda. *roll* se muestra en color rojo en la parte superior del grafico mientras *pitch* se muestra de color azul en la parte inferior del gráfico.

Figura 108. **Datos normalizados aplicando un filtro complementario**

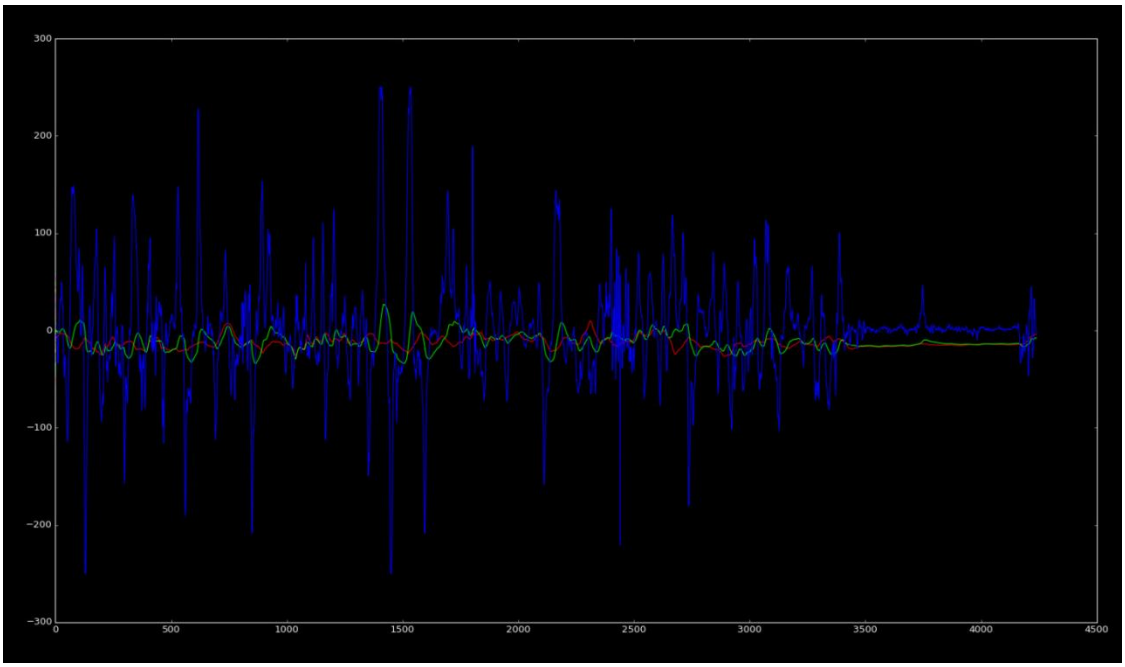


Fuente: elaboración propia.

En el caso del filtro complementario, se esperan variaciones mucho mayores en casos específicos, ya que como se ha mencionado anteriormente, no solamente es una manera de estabilizar la señal.

En la siguiente imagen, se muestra la señal de *roll* del acelerómetro en color rojo, la señal de *roll* del giroscopio en color azul y la señal filtrada en color verde. En ese gráfico se aprecia claramente la acción de ambas señales en proporciones diferentes establecidas en la ecuación.

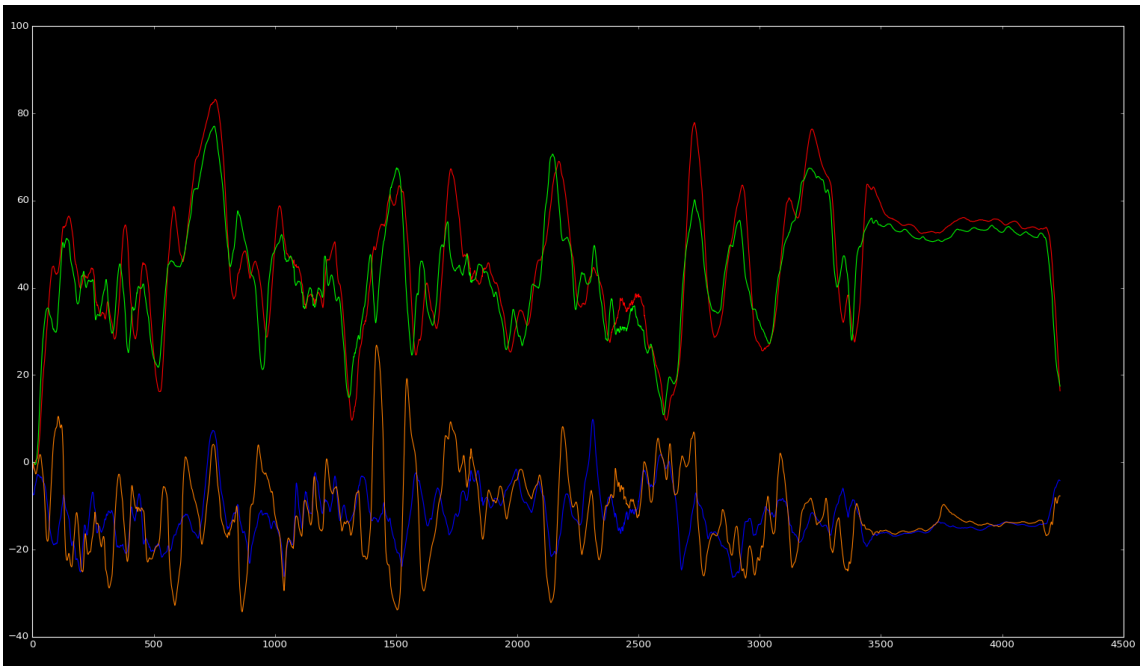
Figura 109. **Comparación de filtrado de datos**



Fuente: elaboración propia.

La siguiente imagen solamente muestra una comparativa entre los filtros de media móvil solamente con los datos del acelerómetro y los filtros complementarios con los datos del acelerómetro y giroscopio. En el caso de *roll* son las gráficas superiores, siendo el filtro de media móvil el gráfico rojo y el filtro complementario el gráfico verde. En el caso de *pitch* son las gráficas posteriores, siendo el gráfico del filtro de media móvil el de color azul y el filtro complementario el gráfico de color naranja. Claramente se observa como el cambio de velocidad angular del sensor afecta en la estabilidad del sensor. En este caso, si es necesario determinar esta velocidad angular, ya que esto puede determinar algún golpe en la careta o algún evento no pronosticado de otra manera.

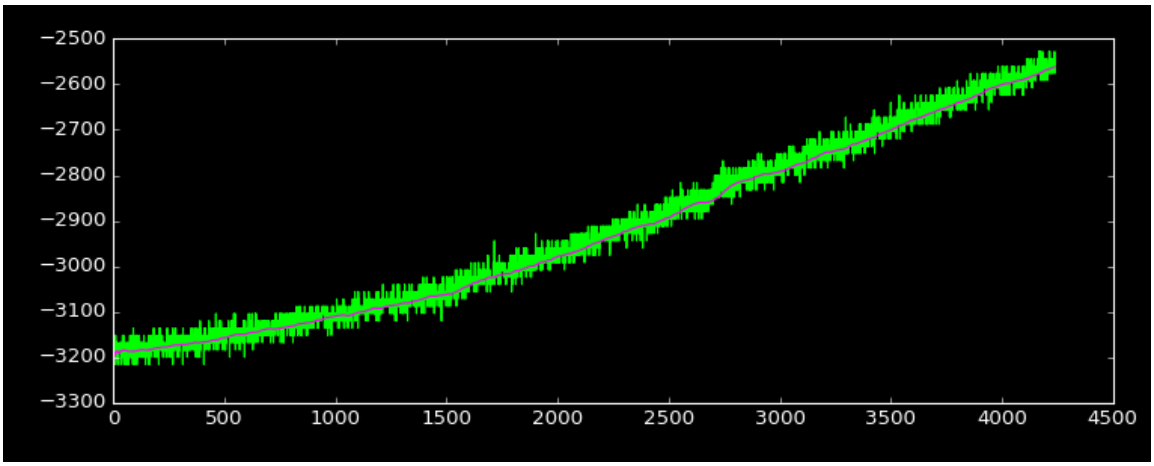
Figura 110. **Comparación de filtros**



Fuente: elaboración propia.

Por último, se tiene una imagen del sensor de temperatura del MPU6050, el cual es altamente inestable y rápido, lo cual da acceso a aplicarle un filtro de media móvil con una ventana de 100 valores sin tener mucho retraso. La siguiente imagen muestra el comportamiento de esta señal.

Figura 111. **Datos de temperatura MPU6050**



Fuente: elaboración propia.

3.4.3. Configuración de dispositivos bluetooth

Los dispositivos bluetooth HC-05 se pueden modificar como maestros o esclavos con comandos AT; la diferencia entre un dispositivo configurado como maestro a un dispositivo configurado como esclavo es que un maestro puede conectarse a un esclavo sin importar cuál sea, mientras que un esclavo a un maestro no. En realidad, quien inicia el proceso de emparejamiento es el maestro.

Los pasos para emparejar 2 dispositivos bluetooth y para hacer que estos dispositivos se busquen entre si al encenderse sin aplicar código de nuevo es el siguiente:

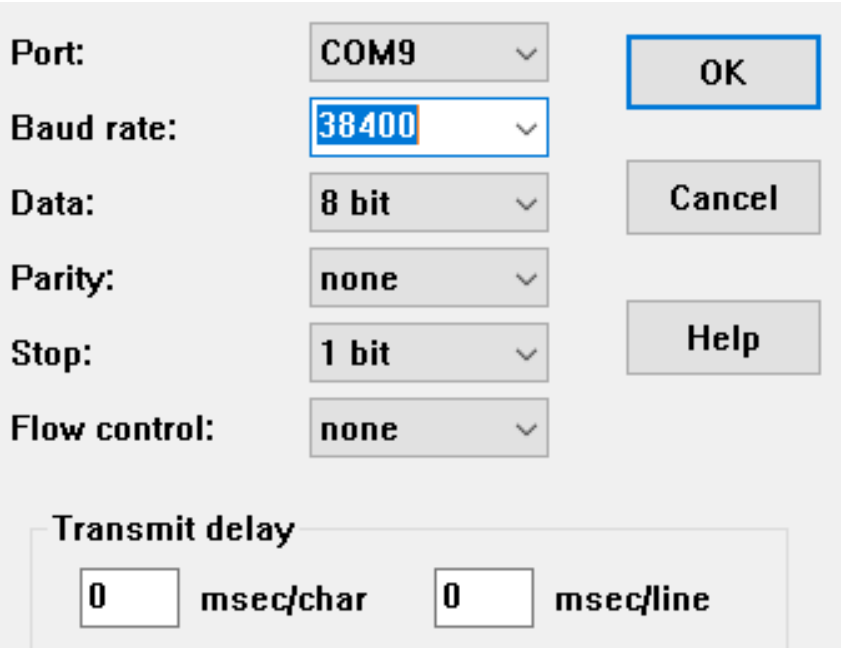
- Se enciende un módulo bluetooth con el pulsador presionado que se dirige al pin PIO11, o pin 34 del encapsulado. Al hacer esto, la luz intermitente cambiará su frecuencia.

- Se conecta el dispositivo bluetooth a un módulo TTL a USB y se asegura que detecte el dispositivo como un puerto COM(Windows) o un puerto TTY(Linux); normalmente, se utilizan tarjetas con el circuito integrado CP2102.

- Se conecta los 3.3VDC del módulo USC CP2102 a VCC del módulo bluetooth HC-5, GND del CP2102 a GND del HC-05, RX del CP2102 al TX del HC-05 y TX del CP2102 al RX del HC-05.

- Se abre una terminal serial (putty, como ejemplo) y se abre el puerto que hace referencia al dispositivo. El puerto se configura con un *baud rate* de 38 400 baudios.

Figura 112. **Configuración de puerto**



Fuente: elaboración propia.

- Se escribe el comando AT y como respuesta, en la terminal debe aparecer un “OK”.

- Si se llega a este paso es porque las conexiones están bien realizadas y se tiene comunicación en modo configuración con el dispositivo bluetooth. Inicialmente se configura el esclavo escribiendo los comandos de la manera siguiente:

    - “AT+ORGL”,ç: con este comando se configura el módulo con las configuraciones de fábrica.

    - “AT+ROLE=0”: este comando configura como esclavo el módulo bluetooth.

    - “AT+POLAR=1,0”: este comando es para configurar la polaridad del pin de estado, que es el pin 32.

    - “AT+ADDR”: con este comando se obtiene la dirección física del dispositivo esclavo. Esta servirá más adelante para ingresarla en el dispositivo maestro. En este caso devuelve la siguiente dirección: '98d3:31:b398d5'.

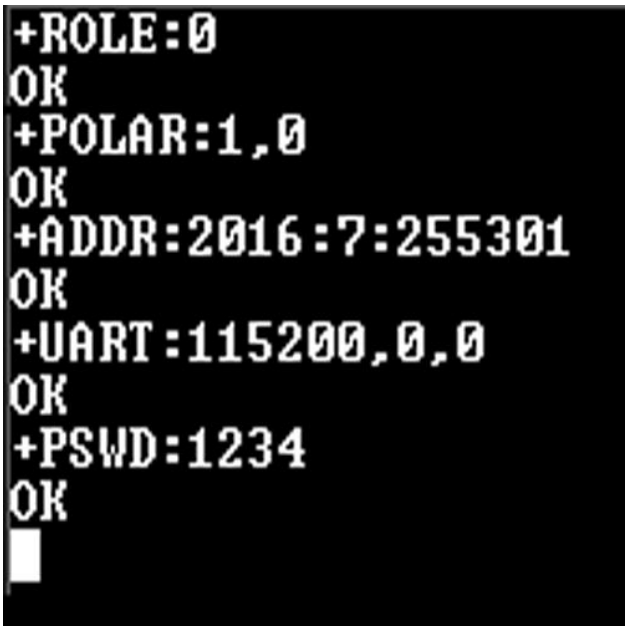
    - “AT+UART=115200,0,0”: Con este comando configuramos los baudios del dispositivo a 115200.

    - “AT+PSWD=1234”: con este comando configuramos la clave para emparejar el dispositivo como '1234'.

    - “AT+INIT”: con este comando se reinicia el módulo.

La siguiente imagen muestra cómo debe quedar el módulo esclavo luego de ser configurado. ¿Para visualizar los datos se escribe 'AT+ROLE?', 'AT+POLAR?', '¿AT+ADDR', 'AT+UART?' y 'AT+PSWD?'. El signo de interrogación sirve para obtener la respuesta de lo que como está configurado lo que se está solicitando.

Figura 113. **Comandos AT, 1**



Fuente: elaboración propia.

- Luego de configurar el módulo esclavo, se desconecta y se conecta el segundo módulo. Al conectarlo se repite desde el primer paso hasta el quinto paso, que es donde escribimos el comando 'AT esperando una respuesta 'OK'. Si esto sucede, se configura el dispositivo maestro con los comandos de la siguiente manera:

- “AT+ORGL”: con este comando reconfiguramos el módulo con las configuraciones de fábrica.

- “AT+ROLE=1”: este comando configura como maestro el módulo bluetooth.

- “AT+POLAR=1,0”: este comando es para configurar la polaridad del pin de estado, que es el pin 32.

- “AT+CMODE=0”: con este comando configuramos que el dispositivo maestro se conecte solamente con el dispositivo esclavo que se selecciona.

- “AT+UART=115200,0,0”: con este comando configuramos los baudios del dispositivo a 115200. La velocidad en el dispositivo esclavo y en este dispositivo debe ser exactamente la misma, de lo contrario no se comunicará o simplemente se recibirán caracteres que no son útiles ni información certera.

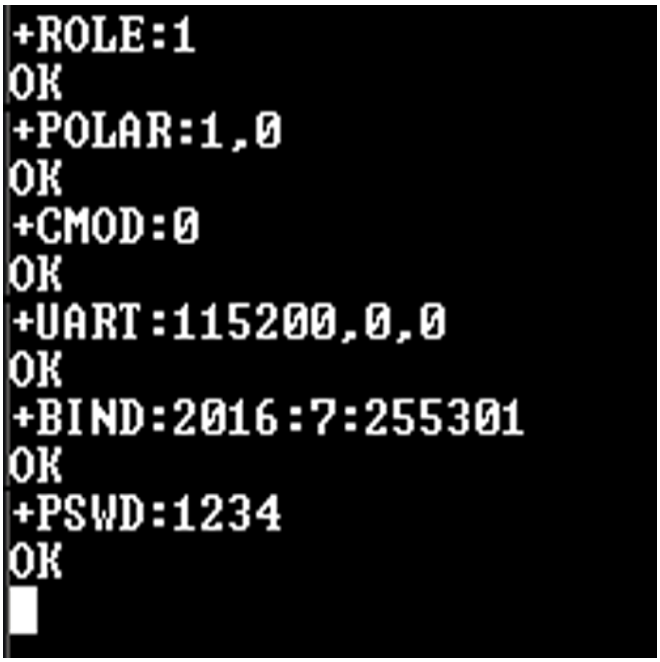
- “AT+BIND=98d3,31,b398d5”: con este comando se establece que el dispositivo a buscar es el de la dirección mencionada. La dirección se ingresa separada por comas, no por puntos como se muestra en el dispositivo esclavo.

- “AT+PSWD”: con este comando configuramos la clave para emparejar el dispositivo como '1234'.

- “AT+INIT”: con este comando reiniciamos el módulo.

La siguiente imagen muestra cómo debe quedar configurado el módulo maestro luego de ser configurado. Para visualizar los datos escribimos 'AT+ROLE?', 'AT+POLAR?', 'AT+CMODE?', 'AT+UART?', 'AT+BIND?', 'AT+PSWD?'. El signo de interrogación sirve para obtener la respuesta de lo que como está configurado lo que se está solicitando.

Figura 114. **Comandos AT, 2**



Fuente: elaboración propia.

- Al llegar hasta este punto, se desconecta ambos módulos y se conecta de manera independiente a 2 puertos USB por medio de módulos CP2102, o 2 microcontroladores configurados para leer un puerto serial. Si se logra recibir en un módulo los caracteres transmitidos por el otro modulo, la configuración está completa. En caso de no tener envio y recepción de datos en este punto, revisar el *baud rate*, bits de paridad y

bits de configuración; luego, revisar que la dirección del dispositivo esclavo este correcta en el dispositivo maestro.

3.5. Software

A veces separar los conceptos de *firmware* y de software tienden a ser confusos, aunque en realidad las diferencias entre software y *firmware* son muchas y bastante grandes. Una de las diferencias principales entre *firmware* y software es que el firmware es un bloque de registros e instrucciones programadas para fines específicos almacenados sobre una memoria no volátil, la cual puede ser EEPROM, Flash, ROM, etc. Estos registros e instrucciones programadas contienen la información y lógica necesaria para interactuar directamente sobre el hardware requerido o comunicarse con otros dispositivos de hardware. El software por otro lado, son lenguajes que la mayor parte de las veces son lenguajes de alto nivel que pueden ser compilados, sintetizados o interpretados por otro lenguaje hasta llegar a lenguaje de máquina. En el caso del software no interactúa directamente con el hardware hasta llegar al lenguaje de máquina.

El software es necesario para la programación en la Raspberry pi zero. La tarjeta se basa en un microprocesador Broadcom BCM2835 con un sistema GNU/Linux corriendo en esta.

Como software de desarrollo principal se utiliza un lenguaje de programación llamado Python. Python es un lenguaje de programación interpretado el cual, por su simpleza en cuanto a sintaxis de programación es un lenguaje muy simple de utilizar y muy fácil de aprender. Es un lenguaje de código abierto compatible con la licencia publica general GNU a partir de la versión 2.1.1.

Python es un lenguaje de programación multiparadigma, lo cual significa que, en lugar de forzar a un programador a un estilo único del lenguaje, el programador puede adaptar el lenguaje a su estilo de programación, ya que permite una programación funcional, estructurada, imperativa y orientada a objetos. Python utiliza un tipado dinámico y conteo de referencias para administrar la memoria.

Por la facilidad de uso de Python, de igual manera existen muchas librerías y mucha documentación sobre su uso, lo cual para el desarrollo de un prototipo es muy útil no solo por su simpleza e interactividad, sino también por su estabilidad. En lo personal, el lenguaje Python me parece mucho más estable y más fácil de utilizar en un sistema GNU/LINUX que en otro sistema.

Otro software utilizado es MySQL y sirve para gestionar bases de datos de manera relacional. Es desarrollado bajo licencia dual GPL/licencia comercial por Oracle y es conocido como el sistema open source de bases de datos relacionales más popular a nivel mundial.

Llevar a cabo un sistema de control y almacenamiento de datos en este caso sirve para poder analizar y estudiar los datos de las variables ambientales dentro de un incendio. De igual manera sirve para llevar el control y los registros del tiempo de trabajo en un incendio y sacar estadísticas de los puntos más afectados en cuanto a las variables muestreadas en incendios a futuro.

3.5.1. Configuraciones necesarias en el sistema Raspbian-Jessie

Las configuraciones del sistema en una Raspberry Pi Zero sirven desde el acceso por *Secure Shell* (SSH), hasta la configuración de los pines GPIO con

los protocolos correspondientes asignados a cada pin. En algunos casos específicos no se puede utilizar el código realizado en la Raspberry sin antes configurarla. En este caso, se toma como punto inicial desde que la Raspberry Pi Zero W ya se encuentra con el sistema operativo instalado. Las configuraciones correspondientes a la Raspberry Pi Zero son las siguientes:

3.5.1.1. Configurar terminal serial de Raspberry Pi Zero en Raspbian-Jessie

La terminal de un sistema operativo Linux es el entorno de comandos por el cual se puede interactuar con absolutamente todas las partes, modificaciones, configuraciones y documentos del sistema. Esta terminal es necesaria para utilizar el sistema operativo.

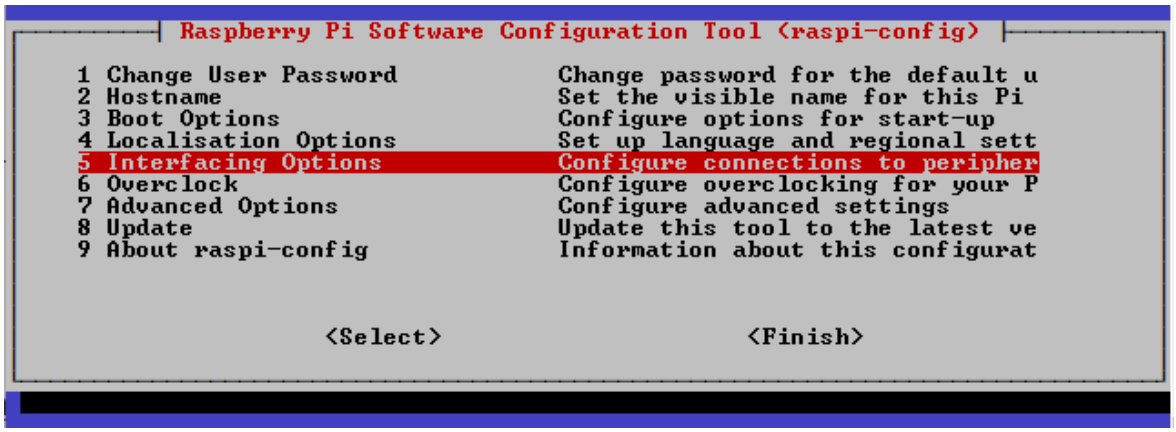
En el caso de la Raspberry Pi Zero, al no traer puerto Ethernet es necesario configurar el ingreso a la terminal de manera remota por medio de un puerto serial. El motivo de esto es para poder tener acceso desde un computador hacia la Raspberry Pi y no tener en la Raspberry pi zero un monitor y teclado, lo cual aparte de ineficiente sería muy tedioso y necesitaríamos siempre estos para poder utilizarla.

Figura 115. **Debian en Raspberry Pi**

Fuente: elaboración propia.

Para activar esta, abrimos una terminal en el sistema (ya sea utilizando un monitor y teclado por una única vez en la Raspberry Pi Zero o configurando el ingreso por SSH al sistema directamente, lo cual se describe más adelante). Luego de abrir la terminal, se ingresa como usuario *root* escribiendo el comando 'sudo su' tal y como muestra la imagen anterior.

Figura 116. **Configuración Raspi-config**



Fuente: elaboración propia.

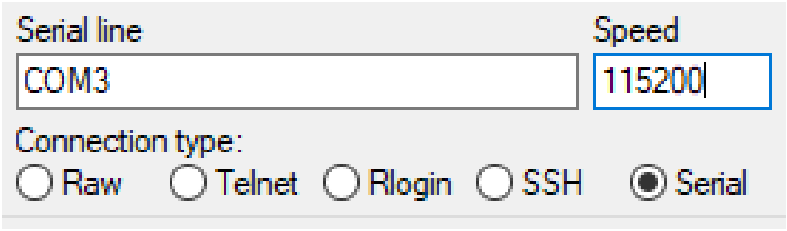
Luego ya con la terminal abierta como *root*, se escribe el comando 'raspi-config' y se abrirá una pantalla de configuraciones la cual tiene un menú de 9 opciones, la opción número 5 es 'interfacing options' y en esta se encuentran las activaciones de todos los protocolos de comunicación por hardware de la Raspberry Pi. La imagen anterior muestra la visualización esperada.

Figura 117. **Configuración terminal puerto serial Raspberry Pi Zero**

Fuente: elaboración propia.

Se ingresa en la opción 'P6 Serial' y aparecerá un mensaje que dice 'Would you like a login Shell to be accesible over serial?' y 2 opciones, una donde dice '<YES>' y otra donde dice '<NO>'. Se selecciona la opción '<YES>'. Luego de esto ya se tiene configurado el puerto serial para ingresar a la terminal Serial de Linux.

Figura 118. **Puerto Com y Bauds**



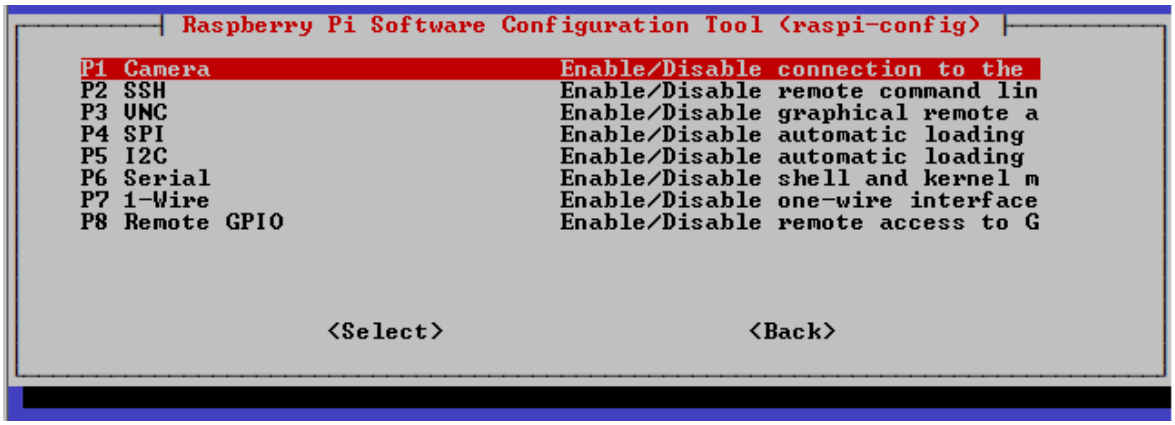
Fuente: elaboración propia.

Para ingresar a la terminal por medio del puerto serial, se necesita un conversor de USB a UART; en este caso se utilizara un CP2102, el cual se encuentra específicamente entre los dispositivos de Hardware del capítulo anterior. Se conecta el pin de VCC del conversor al pin de VCC de la Raspberry Pi Zero, el pin de GND del conversor al pin de GND de la raspberry pi zero, el pin de Tx del conversor al pin de Rx de la Raspberry Pi Zero y el pin de Tx del conversor al pin de Rx de la Raspberry Pi Zero. Al hacer bien esto, lo siguiente que queda es conectarse al dispositivo. Para conectarse al dispositivo se necesita saber a qué puerto COM (Windows) o TTY (Linux) hace referencia el hardware conectado y abrir ese puerto a 115 200 baudios; luego de esto aparecerá la pantalla de inicio de la terminal de Jessie Raspbian para configurar lo que necesitemos.

3.5.1.2. Activar protocolos de comunicación en Raspberry Pi Zero

Para activar los protocolos de comunicación, se siguen los mismos pasos de 'Configurar terminal serial de Raspberry Pi Zero en Raspbian-Jessie' mencionados en el inciso anterior, hasta la elección de 'Interfacing Options' en el menú de 'raspi-config'. Luego se encontrará un menú donde se encuentran todos los protocolos de comunicación definidos por hardware en la raspberry, los cuales son útiles para comunicar un dispositivo con otro de cualquier tipo que maneje el mismo protocolo utilizado. La siguiente imagen muestra las opciones de comunicación con otro hardware que se tiene por medio de la Raspberry Pi Zero.

Figura 119. **Protocolos comunicación Raspberry Pi Zero**



Fuente: elaboración propia.

Luego de ingresar a este menú, se selecciona cada uno de los protocolos a utilizar y aparecerá un mensaje de habilitación del protocolo, al cual se responde presionando un '<YES>'. Los protocolos necesarios para este desarrollo son los protocolos SSH, SPI, I2C, Serial, 1-Wire y remote GPIO.

Es necesario mencionar que aparte de estos protocolos de comunicación, la Raspberry Pi Zero W cuenta con un puerto USB, un puerto HDMI, una interfaz wireless y bluetooth embebida en la tarjeta, pero estos no son utilizados solamente para hardware de desarrollo, sino para múltiples usos. También, es necesario mencionar que, en la configuración de la cámara dentro de las configuraciones de la Raspberry Pi Zero, hace referencia al puerto especifico que trae esta dentro de la tarjeta.

Por motivos de seguridad, el sistema operativo trae por defecto el ingreso por SSH desactivado. Este acceso es necesario para poder configurar, programar o hacer cualquier cambio necesario en la Raspberry Pi remotamente

sin necesidad de tener un monitor y un teclado específicamente para esta, de esta manera se puede programar en un entorno más cómodo y eficiente desde cualquier lugar.

3.5.1.3. Configuración de wifi desde una terminal Raspbian-Jessie en Raspberry Pi Zero

La necesidad de poder configurar el wifi desde la terminal es dada la posibilidad que no se tenga acceso a un teclado y un monitor directamente para la Raspberry Pi Zero, ya que el desarrollo de este prototipo es buscando la menor cantidad de recursos posibles. Si en caso se tienen, esta cuenta con un puerto mcro USC y un mini HDMI, lo cual genera muchos problemas en cuanto a utilizar adaptadores; además, si se sabe manejar por medio de una terminal toda la información, es mucho más rápida y eficiente toda configuración.

Para configurar el protocolo wifi desde la terminal, se accede a esta ya sea por un puerto serial, por SSH o físicamente, se ingresa como Root escribiendo el comando 'sudo su'. Luego de esto se abre en un editor de texto el archivo llamado 'wpa_supplicant.conf', que se encuentra en el directorio '/etc/wpa_supplicant', esto se hace con el comando 'nano /etc/wpa_supplicant_wpa_supplicant.conf' como muestra la siguiente imagen.

Figura 120. **Raspberry pi wireless config**

Fuente: elaboración propia.

Luego de esto, se presiona 'Enter' y se ingresa al archivo de configuración. En este archivo de configuración se ingresan los datos de la red a la que se conectará la Raspberry Pi Zero ingresando los datos de SSID, PSK y Key Management. Esto se hace por medio de una variable denominada 'network' tal y como se muestra en la imagen posterior.

Figura 121. **Configuración de redes wireless**

Fuente: elaboración propia.

Luego de esto, se presiona 'CTRL + O' seguido de 'Enter' para guardar, y luego “CTRL+X” para salir. Se reinicia la Raspberry Pi Zero con el comando 'shutdown -r now' y al ingresar de nuevo al sistema de nuevo, se debería de poder conectar a la red establecida si se tiene alcance a esta misma.

3.5.1.4. Configuración de pines GPIO

Los pines GPIO son pines tanto de propósito general como de comunicación, pero las configuraciones de comunicación solamente requieren activarlas en el sistema tal y como se realizó anteriormente. Para utilizar pines de propósito general se puede configurarlos paso a paso con código de bajo nivel dentro del programa que se esté realizando, o evitarse eso instalando una librería que solucione exactamente lo mismo solo que de una manera más

simple. La librería utilizada se llama Pigpio y se menciona dentro de las librerías de software; aquí se describirán cómo instalarla y utilizarla. Para instalarla abrimos de nuevo una terminal de Raspbian-Jessie y se ingresa como *root* escribiendo el comando 'sudo su'. Luego se escribe el comando de instalación 'apt-get install pigpio' y se siguen los pasos para finalizar. Al terminar, se necesita

Luego de esto, al iniciar a programar en python e inicializar un objeto de la librería, es posible que no cree el objeto que se requiere, por lo cual se necesita correr antes una utilidad llamada 'pigpiod', esta lo que hace es iniciar la librería pigpio como un denomio. Una vez iniciada, la librería pigpio se ejecuta en segundo plano y acepta comandos de las interfaces y *sockets* solicitados.

La utilidad pigpiod requiere privilegios de Root para iniciar la librería, pero a partir de ese punto cualquier tipo de usuario puede interactuar con la librería y utilizar sus configuraciones. Para utilizarla simplemente se escribe en una terminal *root* el comando 'pigpiod'.

3.5.2. Base de datos

Las bases de datos, también son conocidas como bancos de datos, sirven para almacenar información de una manera sistemática. Normalmente, las bases de datos son utilizadas en informática de una manera digital y son gestionadas por medio de sistemas diseñados específicamente para almacenar datos de una manera ordenada.

3.5.2.1. Conceptos básicos

Actualmente, existen 2 tipos de bases de datos, las cuales se diferencian como bases de datos relacionales y no relacionales. Las bases de datos relacionales se conocen de otra manera como SQL y las no relacionales como“NoSQL. Una base de datos SQL (*structured query language*) cumple con un modelo relacional, esto significa que se pueden establecer conexiones entre la información que posee la base de datos, y a través de las relaciones optimizar la memoria utilizada. Una base de datos NoSQL (N*ot only structured query language*) es un modelo que utiliza solventar las limitaciones de una base relacional, esta se utiliza normalmente en entornos de almacenamiento masivo de datos. La ventaja principal de este tipo de base de datos es la velocidad con la que se realizan las consultas.

En cualquier base de datos existen cuatro conceptos fundamentales, los cuales son los siguientes:

- Tablas: son las estructuras internas de una base de datos donde se almacenan los datos de manera ordenada. En una base de datos relacional, las tablas se relacionan por medio de llaves primarias y llaves foráneas, esto es útil para no duplicar las llaves en algunos casos y evitar tener problemas por datos duplicados donde no se necesitan. Las tablas posiblemente son el elemento más importante dentro de una base de datos, y se compone de otros elementos internos, los cuales son los siguientes:

    - Columnas/atributos: útiles para almacenar información de múltiples tipos, por ejemplo, números, cadenas, fechas, enumeraciones, entre otros.

- Filas/registros/tuplas: almacenan secciones de información individuales de un conjunto de columnas especifico.

- Índices: son necesarios para encontrar las filas buscadas.

- Controles: son necesarios para confirmar si los datos ingresados son válidos o no, dependiendo el tipo aplicado a cada columna.

- Consultas: son código con el cual podemos acceder a los datos que se almacenan en las tablas de la base de datos. Las consultas de igual manera son útiles para ordenar, filtrar, depurar, entre otros.

- Informes: son la manera más eficiente de mostrar los datos consultados.

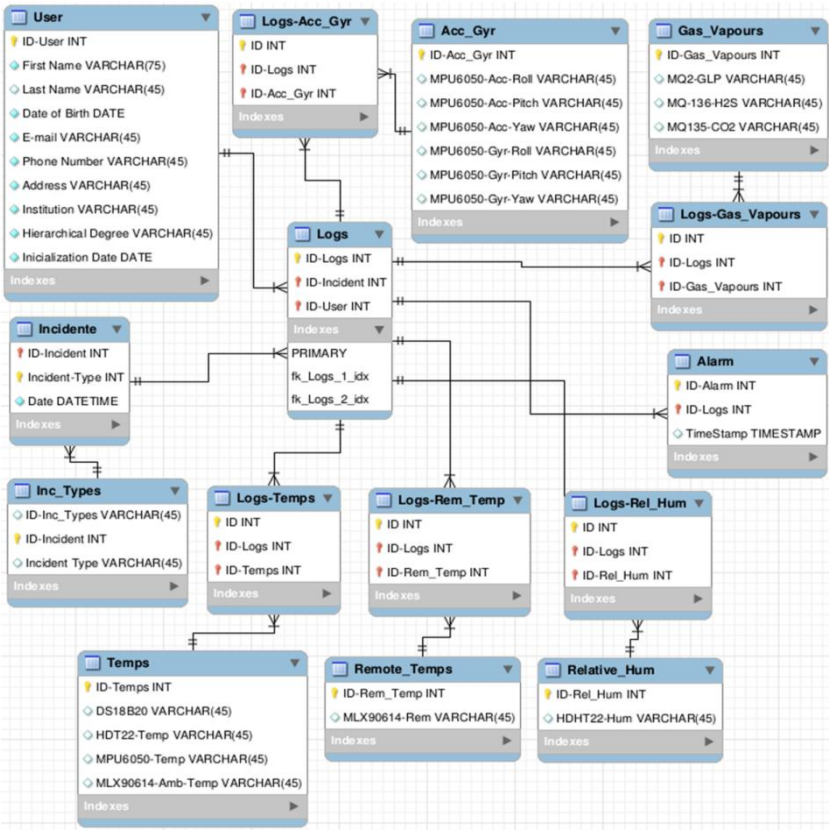
- Formularios: son herramientas útiles para la introducción de datos hacia la base de datos.

En este desarrollo se utilizó un sistema de gestión de bases de datos relacional, desarrollado bajo licencia GPL/Comercial por Oracle Corporation denominada MySQL. Este sistema de gestión se considera como la base de datos Open Source más utilizada en el mundo principalmente para desarrollo web. El motivo principal de utilizar esta base de datos es la cantidad de memoria que esta utiliza, ya que es relacional. El segundo motivo es la popularidad de esta base de datos y la cantidad de información disponible, ya que cualquier persona que desee analizar los datos de manera independiente a lo que el proyecto corresponde, puede hacerlo con suma facilidad. En este caso no se necesita una velocidad demandante en la lectura de datos, pero si necesitamos optimizar el espacio en la mayor escala posible.

3.5.2.2. Diseño

El diseño de la base de datos se hizo de tal manera que, las búsquedas en la base de datos no fueran ineficiente, de la misma manera, que el código sea lo menos repetitivo posible. Uno de los puntos donde el código puede volverse sumamente repetitivo es la tabla 'Acc_Gyr' y la tabla 'Temps' por la cantidad de permutaciones que se pueden tener con los datos obtenidos, pero estas se diseñaron de esta manera ya que son las que más utilidad tendrán. El diseño completo de la base de datos se muestra en la siguiente imagen.

Figura 122. **Base de datos completa**



Fuente: elaboración propia.

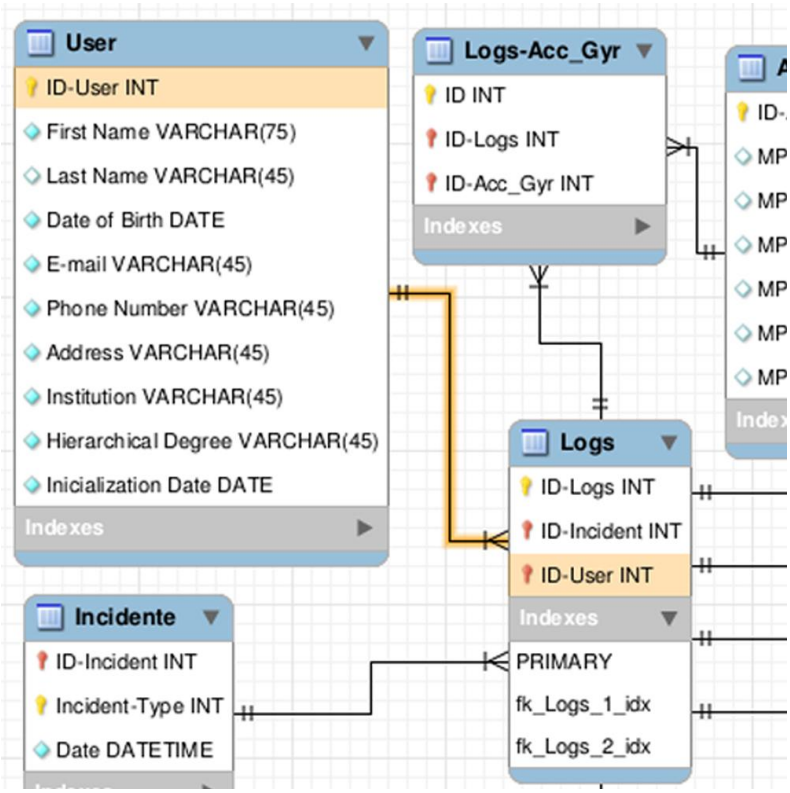
La base de datos guarda información de usuario que opera el dispositivo, tipo de incidente y fecha de este, información de sensores e información de las activaciones de la alarma. Todos los datos se encuentran relacionados entre llaves primarias para optimizar las búsquedas y la memoria utilizada.

La tabla 'Users' guarda toda la información necesaria del usuario, tomando en cuenta también la institución a la que pertenece y el grado jerárquico. La tabla 'Logs' es la que relaciona todos los datos.

El usuario se relaciona a la tabla 'Logs' por medio de 'User', ya que puede haber múltiples usuarios aportando su trabajo dentro de un incidente, así de esta forma sabremos de quien son los datos a guardar. La relación es una relación 'One to Many', ya que solamente un usuario puede aportar datos desde un dispositivo, pero puede aportar muchos datos, tal y como se muestra en la siguiente imagen.

El caso en el que en realidad sea necesaria la tabla 'User' será cuando la base de datos se suba a un servidor y los dispositivos se conecten a este. Mientras la base de datos se utilice en 'Localhost', la tabla usuarios no es sumamente necesaria. Esta tabla se creó ya que este dispositivo está sujeto a cambios.

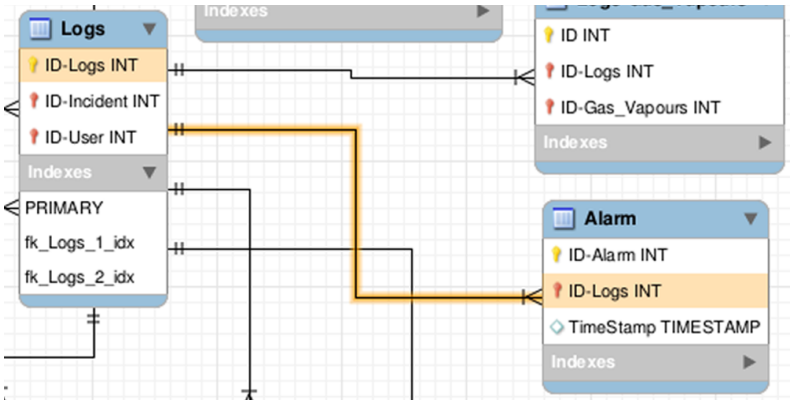
Figura 123. **Usuarios**



Fuente: elaboración propia.

De igual manera se encuentra relacionada la tabla 'Alarm', ya que un 'Log' puede tener múltiples alarmas, pero las alarmas pueden tener un solo 'Log'. La relación se visualiza en la siguiente imagen.

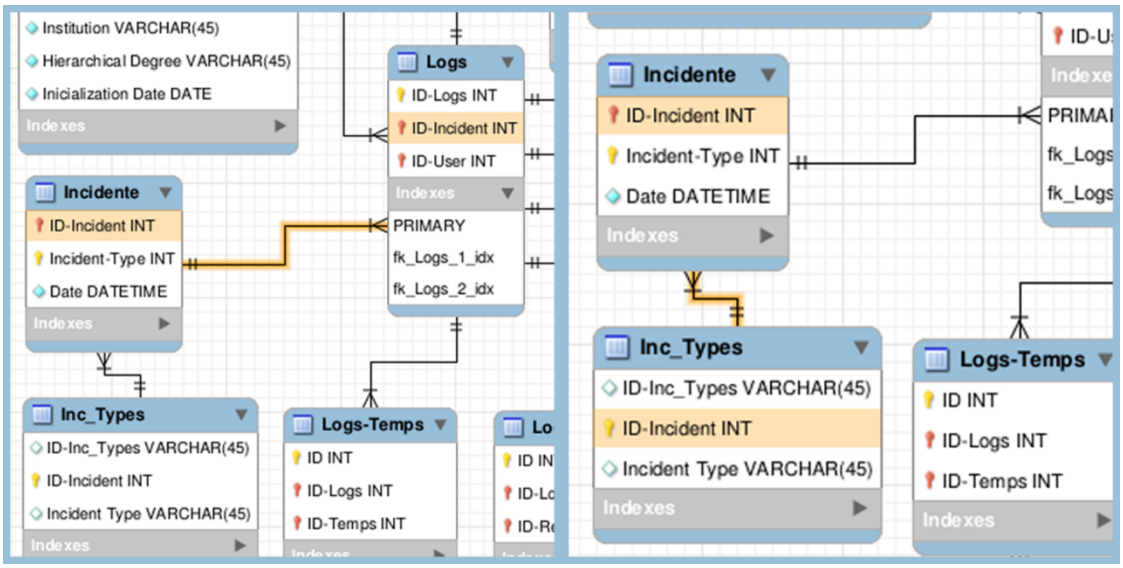
Figura 124. **Alarma**



Fuente: elaboración propia.

La tabla 'Incident' hace referencia a el incidente específico que se está midiendo, sin embargo, puede que se tengan varios ingresos de incidentes en uno solo, esto sería en caso se reinicie el dispositivo en algún momento. Esto puede ser cuestión de mejora en implementaciones futuras. Esta tabla se relaciona con 'Incident Types', la cual contiene la información de los tipos de incidentes en los que puede ser útil este dispositivo, dígase incendios estructurales, incendios vehiculares, lugares con riesgo a incendio, fugas de gas, etc. Estas relaciones se muestran en la siguiente imagen.

Figura 125. **Incidente y tipos de incidente**



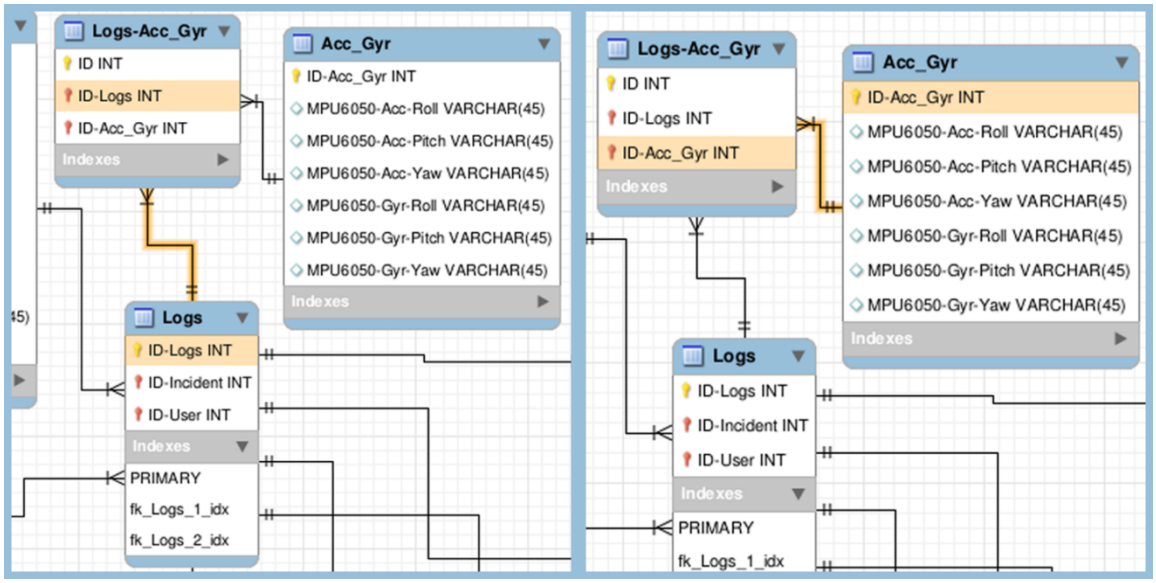
Fuente: elaboración propia.

Las siguientes tablas son tablas de relaciones e información puramente de los datos muestreados por los sensores. Estos datos son relacionados todos directamente con la tabla 'Logs', ya que 'Logs' es la que relaciona toda la información en esta base de datos. La relación se hace por medio de otra tabla intermedia que contiene los identificadores de 'Logs' y los identificadores de la tabla de datos correspondiente por ser de tipo 'Many to Many'.

En cuanto a la información de los sensores, estos pueden provenir de la careta o del pass directamente (suponiendo una base de datos local), esto a nivel de bases de datos es invisible, pero es útil mencionarlo dado el caso en que los datos de la base de datos aparezcan como 'NULL', puede ser por una comunicación inestable. Los datos provenientes de la careta son los de la tabla 'Acc_Gyr', y estos sirven para saber si la persona que tiene el dispositivo se encuentra en movimiento. Estos datos son guardados para poder analizar a

futuro la magnitud de los movimientos y poder desarrollar un dispositivo más eficiente La siguiente imagen muestra las relaciones de esta tabla.

Figura 126. **Acc Gyr y logs Acc Gyr**



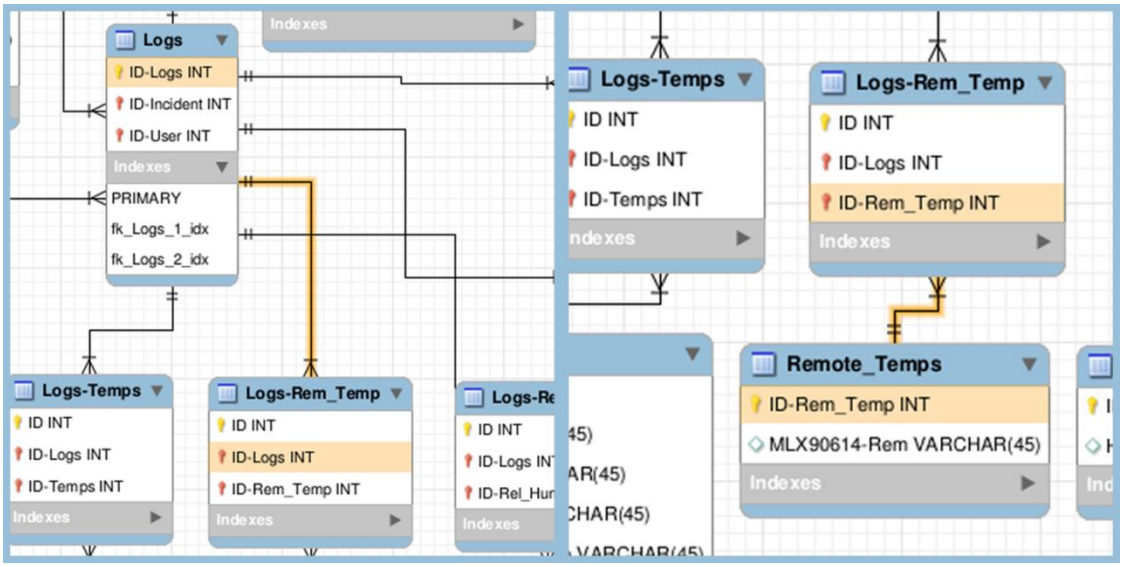
Fuente: elaboración propia.

Todas las tablas siguientes son de datos del pass (o locales trabajando como *localhost*). El único dato remoto es la temperatura proveniente del MPU6050 que se encuentra en la careta.

La tabla 'Temps' contiene temperaturas ambientales medidas desde diferentes sensores. El motivo de tener múltiples mediciones es por 2 motivos. El primero para poder tener mediciones de temperatura no solo directas de la radiación del fuego, sino también indirectas a este. El segundo motivo es para aprovechar todos los datos posibles brindados por lo sensores, ya que hay varios sensores que pueden aportar otras mediciones aparte de las principales, tal y como sucede con el MPU6050. Estos datos pueden servir para tener

redundancia y saber que en realidad los sensores se encuentran en buen estado, o también para mostrar mediciones aproximadas en caso se dañe el sensor principal. Las relaciones de la tabla 'Temps' se muestran en la siguiente imagen.

Figura 127. **Temperatura remota y logs de temperatura remota**

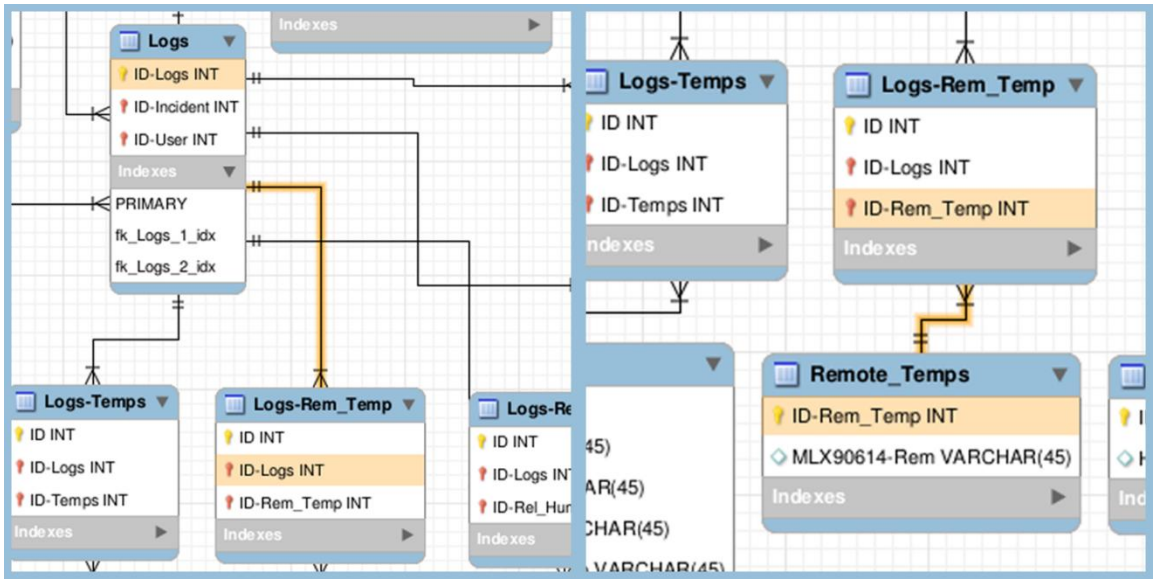


Fuente: elaboración propia.

La tabla 'Remote_Temps' contiene específicamente las mediciones remotas de temperatura del sensor MLX90614, estas mediciones son útiles para saber si algún objeto o cilindro con combustible se encuentra a temperaturas muy altas, o determinar por medio de la temperatura de una puerta si se corre el riesgo a tener un Backdraft al abrirla. Estas temperaturas se almacenan para saber la utilidad de este sensor. De igual manera para saber si los rangos de temperatura de este sensor son lo suficientemente amplios en la práctica o si es necesario cambiarlo por otro, ya que este se utiliza para hacer

una equivalencia entre precio y utilidad. La siguiente imagen muestra las relaciones de esta tabla.

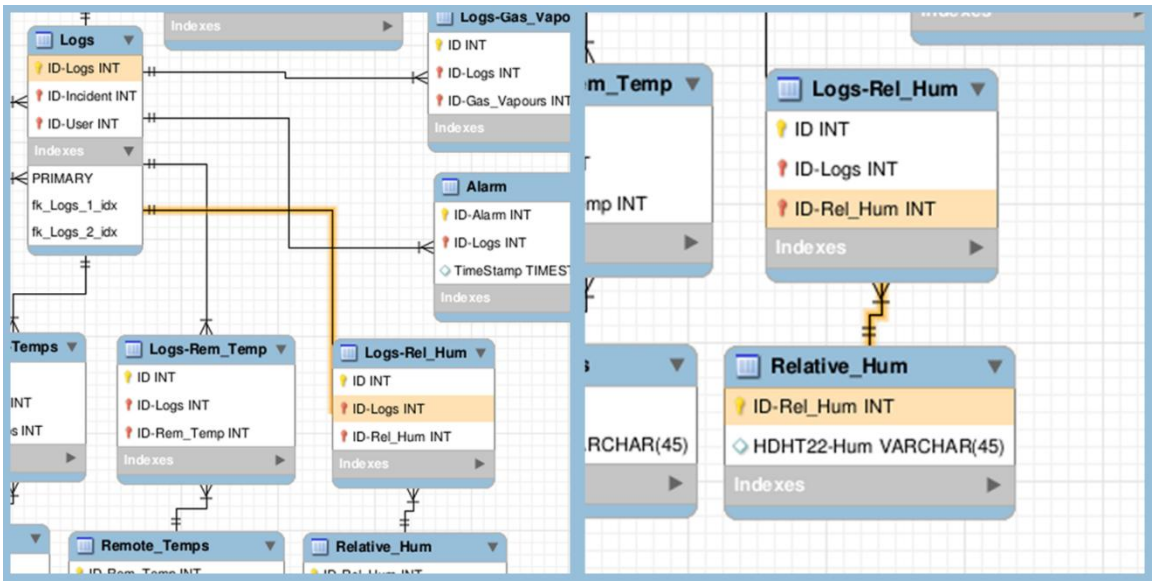
Figura 128. **Logs**



Fuente: elaboración propia.

La tabla 'Relative_Hum' guarda los datos específicamente de la humedad relativa del sensor DHT11; estos datos son útiles para saber qué tan rápido puede evolucionar un incendio y saber si se necesita humedecer más el ambiente para evitar un comportamiento inesperado del fuego. Los datos se almacenan para hacer comparativas del aumento de la temperatura respecto a la humedad relativa del ambiente en futuras ocasiones. La siguiente imagen muestra las relaciones de esta tabla.

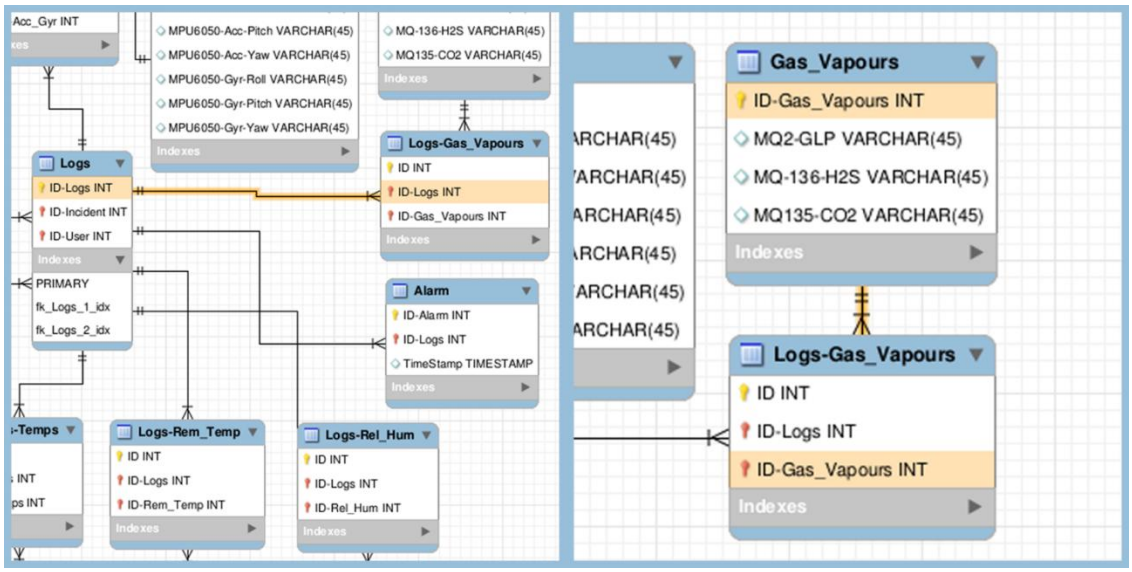
Figura 129. **Humedad relativa y logs de humedad relativa**



Fuente: elaboración propia.

La tabla 'Gas_Vapours' guarda la información de gases en partes por millón ya procesada. Esta es útil para poder sacar estadísticas de proporciones de gases en incendios específicos, de igual manera para agregar más sensores de gases a la lista y estudiar el comportamiento de estos. La siguiente imagen muestra las relaciones de esta tabla.

Figura 130. **Vapores, gases y logs de vapores y gases**



Fuente: elaboración propia.

3.6. Algoritmos

En programación y en matemática, los algoritmos son un conjunto de pasos y reglas a seguir que sirven para llevar a cabo una actividad. En este proceso no quedan ningún tipo de elementos pendientes o vacíos sin tomar en cuenta quien realizara esa actividad. En términos de programación los algoritmos son las secuencias de pasos lógicos a realizar. Existen algoritmos a realizar en cada una de las etapas de un diagrama de flujo, por ende, los diagramas de flujo son la representación gráfica de un algoritmo.

Los algoritmos son empleados para resolver todo tipo de problemas, no solo en programación ni en matemática, sino en cualquier problema de la vida real. Las instrucciones a seguir en algún trabajo son un claro ejemplo de ello.

Unos de los algoritmos más utilizados dentro de la programación son los siguientes:

- Buffer circular: también conocido como buffer cíclico, o de anillo. Es una estructura de datos que utiliza un único buffer en la que se ingresan o eliminan elementos. Estos arreglos son de tamaño finito e internamente se observan como que cada uno de sus extremos estuviera conectado. Un buffer cíclico trabaja con 2 índices para acceder a cada uno de los elementos dentro del arreglo, y ambos índices tienen avance cíclico e incremental. Esto significa que se incrementan de unidad en unidad y, luego de llegar al último elemento del arreglo, inician desde el primer elemento.

- Pendientes: en el caso del cálculo de pendientes matemáticas, es utilizado un buffer cíclico de igual manera para obtener la pendiente no solo de los 2 últimos datos, sino en realidad de poder calibrar la estimación de la pendiente dependiendo de la velocidad de entrada de datos al pass.

3.7. Diagramas de flujo

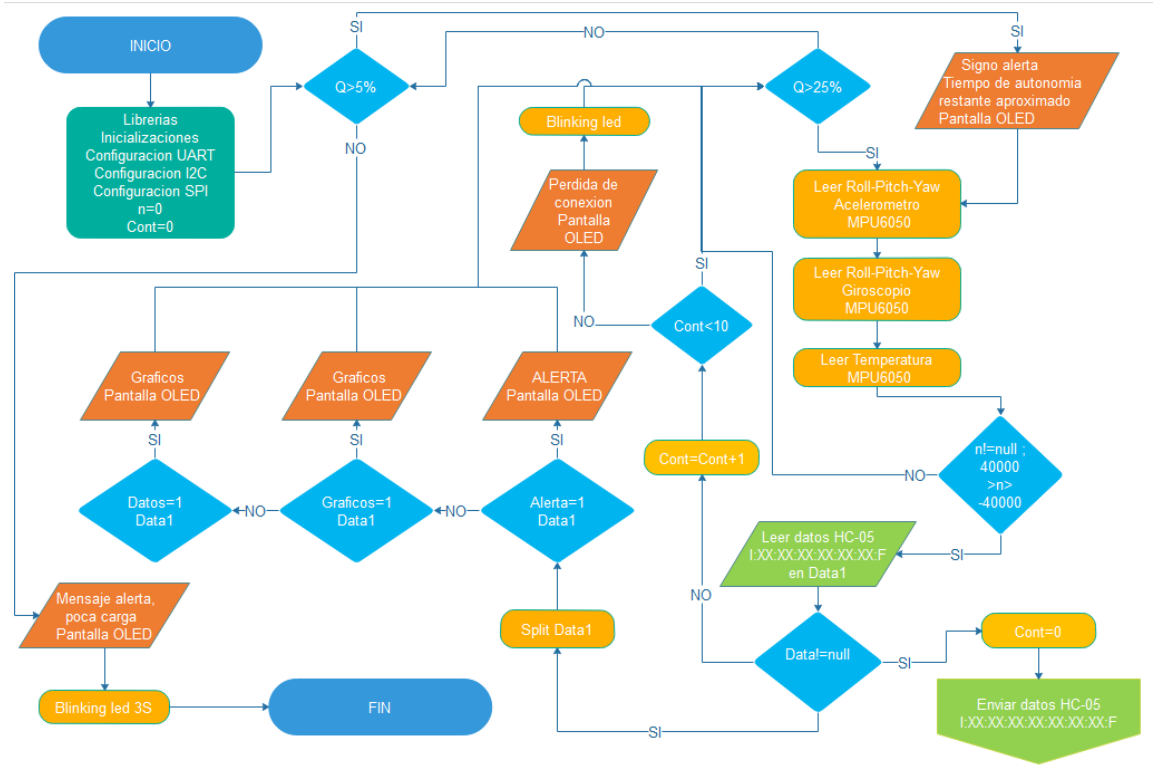
Los diagramas de flujo son útiles en programación de *firmware* o *software* ya que son una representación gráfica del proceso de operación del sistema y algoritmos desarrollados. Un diagrama de flujo muestra desde la actividad principal de un *firmware* o un *software*, hasta los procesos que se llevan a cabo cuando el algoritmo no funciona o se sale de control.

Los diagramas de flujo pueden ser una herramienta útil para encontrar errores de diseño antes de empezar a programar, de esta manera se puede ahorrar tiempo de desarrollo.

3.7.1. Diagramas de flujo de careta

Los diagramas de flujo de *firmware* de la careta son la manera de mostrar gráficamente los procesos ordenados a realizar por el controlador y demás partes del hardware que se encuentran en este módulo. La siguiente imagen muestra el diagrama de flujo de la careta completo.

Figura 131. **Diagrama de flujo careta**

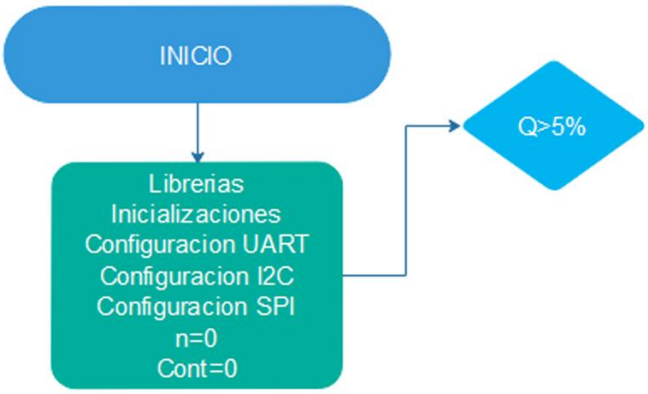


Fuente: elaboración propia.

Si se analiza por partes el diagrama de flujo, es un diagrama muy simple, el cual solamente se encarga de crear visualizaciones en pantalla, leer datos provenientes de un MPU6050, y enviar o recibir información a través de un dispositivo bluetooth HC05.

La primera parte del diagrama corresponde al bloque de 'inicio', el cual debe incluirse en todo diagrama de flujo para saber desde que lugar partir. Luego de este bloque se encuentra un bloque de configuraciones, el cual incluye librerías utilizadas parala pantalla OLED, inicializaciones de protocolos de comunicación e inicialización de variables de programa. Entre las variables a inicializar, hay 3 variables que constituyen parte del proceso del diagrama de bloques. Una es un contador que determina cuantas veces se ha hecho un proceso, otra es un arreglo que guarda los valores del MPU6050 (esta prevista como entero para fines prácticos), y la tercera es una variable de datos, para enviar o recibir datos seriales del HC05. Luego de inicializar las variables, protocolos y librerías, ingresamos a una condicional, que es la única condicional capaz de detener todo el proceso de muestreo y visualización de datos, y es la carga de la batería. Por motivos de seguridad, si la carga de la batería esta debajo del 5 %, detendrá todo el proceso para que el usuario no siga utilizando el dispositivo. La variable Q es la que se encargara en este diagrama de monitorear el porcentaje de batería. La siguiente imagen muestra el diagrama de este proceso.

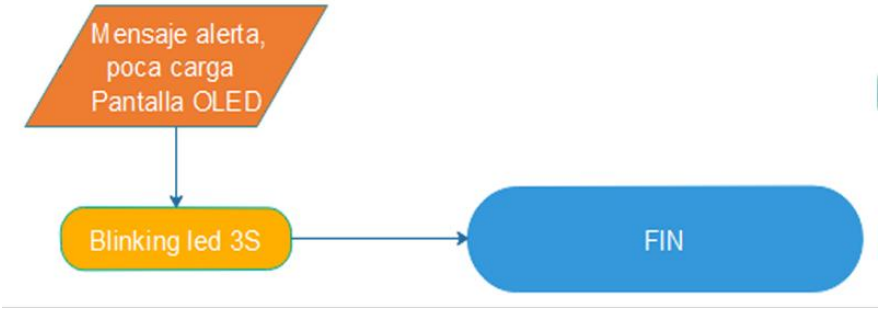
Figura 132. **Inicio diagrama de flujo careta**



Fuente: elaboración propia.

Luego de esto, si la carga esta debajo del 5 %, se muestra un mensaje de alerta en pantalla y muestra un led intermitente por 3 segundos, después de esto no se hace nada más y el controlador entra en un estado de reposo para finalizar todo proceso. La siguiente imagen muestra el diagrama de finalización que lleva este proceso.

Figura 133. **Fin diagrama careta**

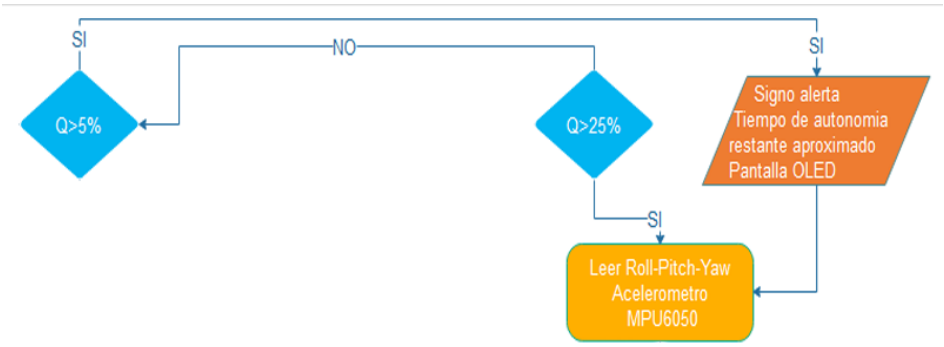


Fuente: elaboración propia.

Si en caso la carga Q de la batería es superior al 5 %, esto lleva a otra área del diagrama de bloques que es en realidad donde se cierra un bucle infinito hasta que la carga sea superior al 5 %. Este bucle infinito incluye la lectura de datos del sensor MPU6050, la visualización del tipo de vista correspondiente, la lectura de trama de datos seriales desde el HC-05 y un bucle de perdida de conexión con el pass.

Si la batería es superior al 5 % de carga y se está inicializando el programa, el diagrama de flujo pasa directamente a mostrar el tiempo de autonomía, luego a la lectura de datos del sensor y seguir un flujo normal a partir de allí. Luego del primer ciclo de ejecución, el diagrama pasa directamente a medir si la batería esta por encina del 25 %, si la batería es superior al 25 % sigue un proceso de lectura de datos del sensor, pero si es menor al 25 %, regresa al condicional del 5 %. Esto se hace para cerrar un bucle infinito donde, si la batería es mayor al 25 %, no sea relevante la carga, mientras que, si la batería es menor al 25 %, se mantenga un mensaje de alerta en pantalla sobre la carga, y si es menor al 5 % que se finalice todo proceso. La siguiente imagen muestra el diagrama correspondiente.

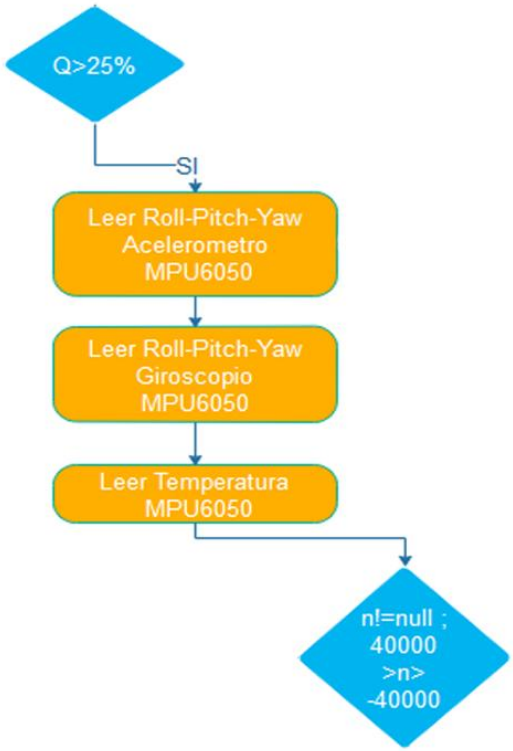
Figura 134. **Carga batería diagrama careta**



Fuente: elaboración propia.

El primer bloque de ejecución de código es de la lectura de datos del MPU6050, obteniendo primero *roll, pitch* y *yaw* del acelerómetro. Seguido de esto se obtiene *roll, pitch* y *yaw* del giroscopio y por último la temperatura. La primera vez que el código se ejecuta, pasa directamente del condicional de la batería que debe ser mayor al 5 %. Luego de las lecturas, se analiza la lectura de datos para saber si en realidad las lecturas son válidas o no, dígase si las lecturas se encuentran dentro del rango de datos correspondiente al sensor MPU6050, o si las lecturas no son nulas. Si ambas partes son ciertas se sigue el trayecto normal del diagrama de flujo, sino se repite el proceso de lectura. La siguiente imagen muestra el diagrama correspondiente.

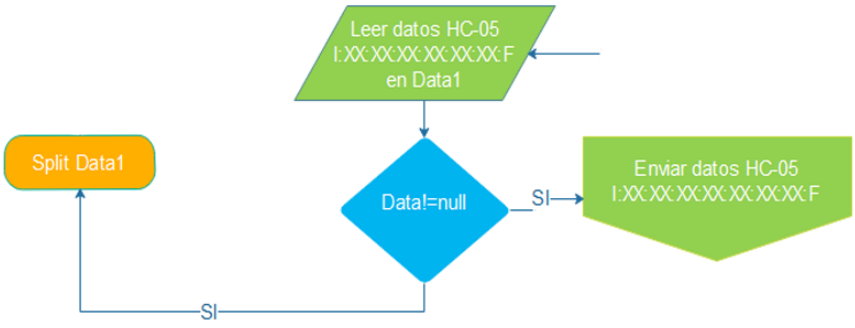
Figura 135. **Diagrama de lecturas careta**



Fuente: elaboración propia.

Si la lectura de datos del sensor es correcta y no se requiere un retorno a tomar datos de nuevo, se pasa a leer la trama de datos proveniente del bluetooth HC-05. En este caso se toma la lectura de la trama de datos completa en un solo *string*, en un formato donde la letra inicial es I y la letra final es F. Dentro de estas 2 letras se reciben todos los valores de todos los sensores separados por dos puntos (:). La lectura de la trama de datos se guarda en una variable tipo *string* llamada Data1 y si la lectura es correcta, se envía una trama de datos tipo *string* con la misma sintaxis anterior al bluetooth HC-05, para que este la envíe al Pass y, al mismo tiempo, se separan las mediciones en posiciones independientes de un solo Array para procesarlas. Si la lectura es incorrecta, de nuevo regresa al ciclo de medir si la carga de la batería es mayor al 25 % y repetir todo el flujo desde allí. La siguiente imagen muestra el diagrama de flujo correspondiente.

Figura 136. **Diagrama de transmisiones de datos careta**

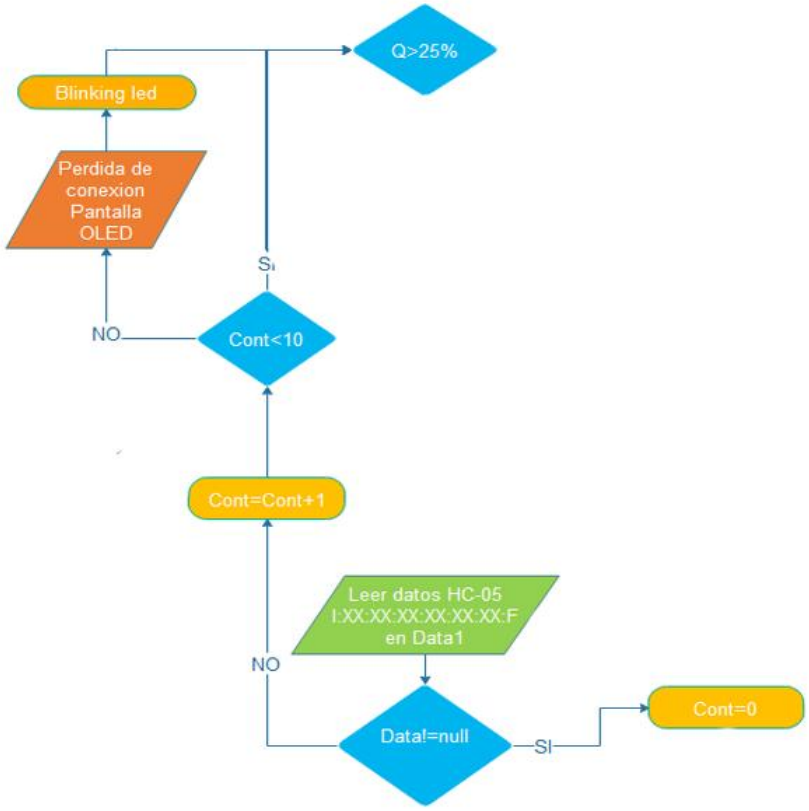


Fuente: elaboración propia.

Si en algún momento se pierde la conexión del *pass* con la careta, la trama de datos recibida será indeterminada, pueden ser caracteres extraños o puede ser nula, esto es depende del motivo de la perdida de conexión. Para estos casos, la variable Data1 no podrá determinar el inicio y el fin del *string*

obtenido, por lo cual siempre retornará al condicional de la batería mayor al 25 %. Para determinar una pérdida de conexión, en este retorno existe una variable auto incrementada denominada contador que solamente sube su valor en 1 si se ingresa a este retorno. Si la variable llega a un valor de 10, significa que se ha pasado 10 veces seguidas por este retorno, lo cual indica la pérdida de conexión, y, en este caso, ingresa a otra condición, la cual muestra una alerta de perdida de conexión en pantalla y un *blinking led*. Esta variable retoma su valor en cero cuando los datos obtenidos son diferentes de null. La siguiente imagen muestra el diagrama de flujo correspondiente.

Figura 137. **Diagrama de perdida de conexión careta**



Fuente: elaboración propia.

Por último, se tiene la parte del diagrama completo que corresponde a la visualización en la pantalla OLED. La visualización se divide en 3 partes:

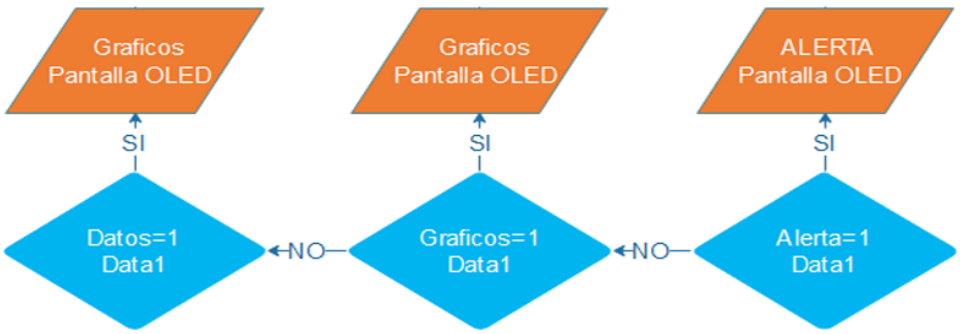
- Alerta: esta visualización se activa cuando la bandera de alerta enviada desde el Pass se encuentra en estado alto. Al activarse esta bandera, lo único que mostrará en pantalla será un mensaje de 'peligro' para que la persona que porta el dispositivo evacue el lugar.

- Gráficos: esta visualización se activa cuando la bandera de gráficos enviada desde el Pass se encuentra en estado alto. Al activarse esta bandera, se observarán las variaciones de las variables principales respecto a un tiempo aproximado de 1 minuto en pantalla, obteniendo de igual manera visualizaciones de las pendientes generadas por el cambio de valores puntuales.

- Datos: esta visualización se activa cuando la bandera de datos enviada desde el Pass se encuentra en estado alto. Al activarse esta bandera, solamente se tendrán datos puntuales de los últimos valores obtenidos en las mediciones de las variables principales.

La visualización de alerta puede activarse en cualquier momento, incluso cuando datos o grafico sesten activas. Cuando la visualización de gráficos se activa, se desactiva automáticamente la visualización de datos y viceversa. La siguiente imagen muestra el diagrama de flujo correspondiente.

Figura 138. **Diagrama de vistas en pantalla careta**



Fuente: elaboración propia.

3.7.2. Diagramas de flujo de *pass*

Los diagramas de flujo del *pass* constan de 2 partes: la primera parte consta de un diagrama básico desarrollado sobre el firmware de un controlador que funciona en cierto modo como un asistente de la Raspberry Pi, dejándole a este las tareas de lectura de datos de sensores para no saturar el flujo del programa principal. La segunda parte consta de un diagrama de flujo sobre el software de la Raspberry Pi, el cual se encarga de ejecutar todas las tareas más complicadas que requieren de un proceso más complicado del que pueden desarrollar los microcontroladores en un tiempo sumamente corto.

El diagrama de flujo del *firmware* del *pass* es sumamente sencillo, consta solamente de un bloque de inicio, seguido por un bloque de inicialización de librerías, protocolos de comunicación y variables necesarias para leer datos de todos los sensores. Luego de estos 2 bloques iniciales sigue la lectura de datos en el siguiente orden:

- Lectura de los datos del sensor MQ2 en un puerto ADC del microcontrolador.

- Lectura de los datos del sensor MQ135 en un segundo ADC del microcontrolador.

- Lectura de la temperatura ambiental del sensor DS18B20 en un puerto 1Wire definido por *firmware*.

- Lectura de la temperatura ambiental del sensor DHT22 en un puerto definido por *firmware*.

- Lectura de la humedad relativa del sensor DHT22 en un puerto definido por *firmware*.

- Lectura de la temperatura ambiental del sensor MLX90614 en un puerto I2C.

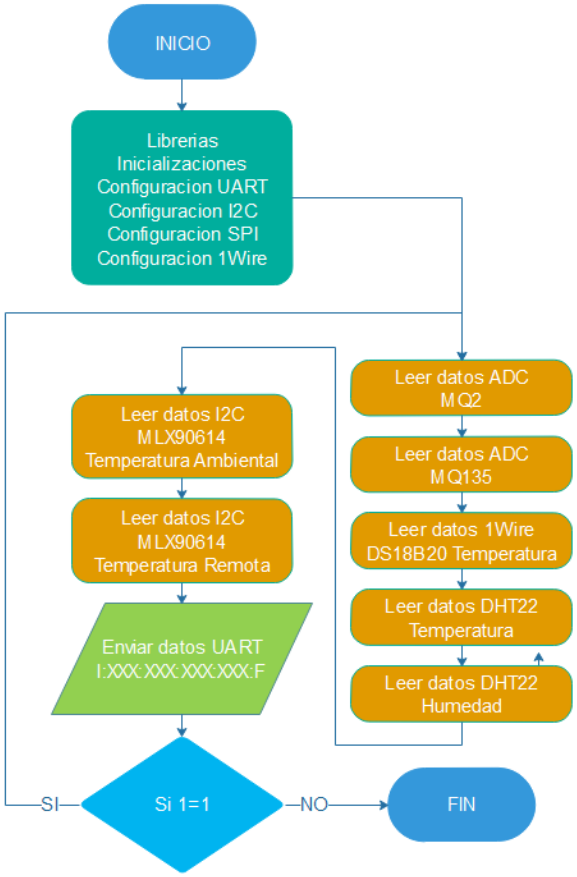
- Lectura de la temperatura remota del sensor MLX90614 en un puerto I2C.

Al culminar todas las lecturas, el microcontrolador envía por un puerto UART todos los datos en una misma nomenclatura de un solo *string*; inicia con la letra I finaliza con la letra F y separa todos los datos ordenados de las variables por medio de dos puntos. Al finalizar este proceso de envío de datos, el flujo sigue hasta una sentencia que siempre será verdadera, esto es para dejar el *firmware* siempre en un mismo bucle infinito. Este firmware no importa si no tiene control alguno, ya que se requiere que siempre se encuentre obteniendo lecturas de los sensores, aunque no esté funcionando nada más.

Es necesario mencionar que el cable del bus UART colocado del Rx del microcontrolador hacia el Tx de la Raspberry Pi Zero es únicamente para el caso en que se desee modificar el firmware más adelante, ya que se puede crear un tipo de *multithreading* para un control más completo y convertir el dispositivo de medición simple en un dispositivo *plug and play*.

La siguiente imagen muestra el diagrama de flujo correspondiente al microcontrolador que toma los datos de los sensores colocado en el *pass*.

Figura 139. **Diagrama de microcontrolador *pass***



Fuente: elaboración propia.

El diagrama de flujo del software del *pass* es el diagrama más largo, por lo tanto, el que tiene una mayor estructura en cuanto a programación. Este diagrama en realidad se encuentra muy resumido, ya que de colocar toda la estructura de la programación en bloques posiblemente sería difícil y largo de comprender. En la siguiente imagen se muestra el diagrama de bloques completo referente al software.

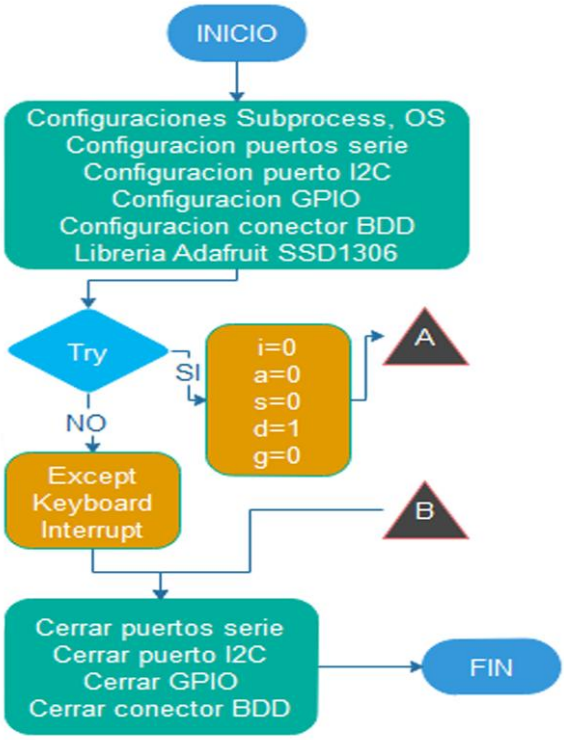
Figura 140. **Diagrama de Raspberry *pass***

Fuente: elaboración propia.

El diagrama se divide en 4 partes: la primera parte se encarga de inicializar todas las configuraciones y librerías y las siguientes 3 partes se ejecutan en multihilos (*multithreading*). En resumen, el primero de los hilos se encarga del menú e interacción con este mismo, siendo el hilo donde se muestra la mayor parte de las visualizaciones y las funciones de los pulsadores. El segundo hilo se encarga de obtener los datos de los sensores enviado desde el microcontrolador que está en el mismo *pass*. Estos datos se filtran, se procesan y se envían a la careta con otros valores necesarios para las visualizaciones. El tercer hilo se encarga de la lectura de datos provenientes de la careta, de filtrar esos datos y de encender o apagar la alarma de movimiento en caso de ser necesario a partir de esos datos.

En las inicializaciones, se incluye la librería de la pantalla OLED SSD1306, la cual sirve para mostrar los datos de configuración y conexión en el mismo pass. Se incluyen también Subprocess y OS, que sirven para poder hacer modificaciones o ejecuciones en una terminal desde el entorno de Python, se incluyen las librerías serial, I2C, PiGPIO y sus configuraciones correspondientes a cada una, estas sirven para protocolos de comunicación y para utilizar los pines GPIO de la Raspberry Pi Zero W. Por último, se incluye MySQL, el cual junto a sus configuraciones correspondientes sirve para conectar la base de datos de MySQL con el programa de Python. La siguiente imagen muestra los bloques correspondientes al inicio del programa, configuraciones, excepciones y final del programa. Los bloques A y B son nodos para unir otras partes del programa a estos mismos puntos.

Figura 141. **Diagrama principal *pass***



Fuente: elaboración propia.

A partir de este punto, en el bloque A se enlazan todos los ciclos de todos los hilos. Y en el bloque B se enlazan todos los retornos en caso que el ciclo no se ejecute por cualquier razón.

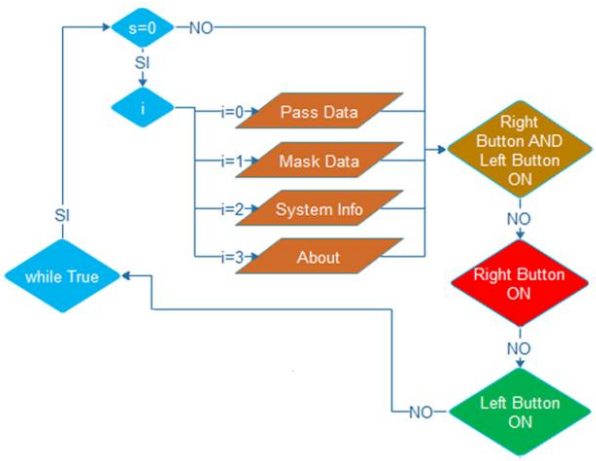
- Hilo 1: este hilo corresponde al bloque de código que ejecuta las visualizaciones de menú y la selección de opciones por medio de los pulsadores. De igual manera contiene la activación de la sirena manualmente y el cambio de tipo de información enviado a la careta. La primera parte de este ciclo consta de una visualización inicial que se ejecuta cuando S es igual a cero, S significa *screens* y es cero solamente

cuando no existe una opción seleccionada. Luego de esto existen 4 opciones en el menú, las cuales realizan las siguientes funciones:

- Pass Data: muestra toda la información proveniente de la careta ya separada en variables independientes.
- Mask Data: muestra toda la información a enviar a la careta aun no concatenada en un solo *string*.
- System Info: muestra los procesos, la memoria RAM y memoria flash utilizadas, la dirección IP en caso de tener conexión a wireless y el estado de conexión con la careta.
- About: muestra los datos de desarrollador, versión de actualización de pass, versión de actualización de careta y versión de actualización de base de datos.

La siguiente imagen muestra el ciclo entre la pantalla inicial y el estado falso de las 3 condiciones de los pulsadores.

Figura 142. **Diagrama ciclo principal *pass***



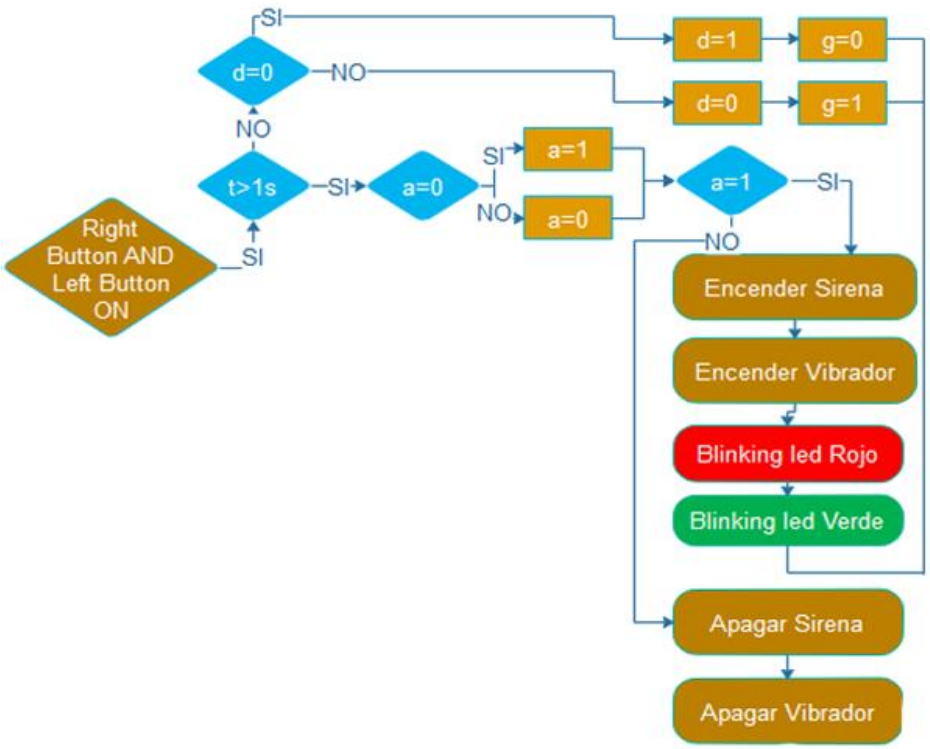
Fuente: elaboración propia.

Por otro lado, analizando los estados en alto de las 3 condiciones, se tiene que la primera condición es cumplir que las variables de ambos pulsadores estén en estado alto, esto se da en el caso de que presione ambos pulsadores al mismo tiempo. Es por este motivo que se concluyó en dejar ambos pulsadores en ambos extremos, ya que en caso de operar el dispositivo se debe poder operar con una sola mano.

Al presionar ambos pulsadores se entra a una segunda condición, la cual mide el tiempo que ha pasado pulsados ambos sin soltarse. Esto sirve para hacer 2 operaciones diferentes con la misma acción, solamente cambiando el tiempo de pulsación.

Si ambos pulsadores se mantienen pulsados por más de un segundo, se inicia una tercera condición que lo único que realiza es cambiar el estado de una variable 'a' y luego, depende el estado de 'a', se enciende o se apaga la alarma. Estas condiciones seguidas son para activar o desactivar la alarma de la misma manera sin variar nada físicamente. Luego de terminar este flujo, se retorna al ciclo infinito para seguir desde el inicio los pasos correspondientes. La siguiente imagen muestra el diagrama de flujo correspondiente a este proceso.

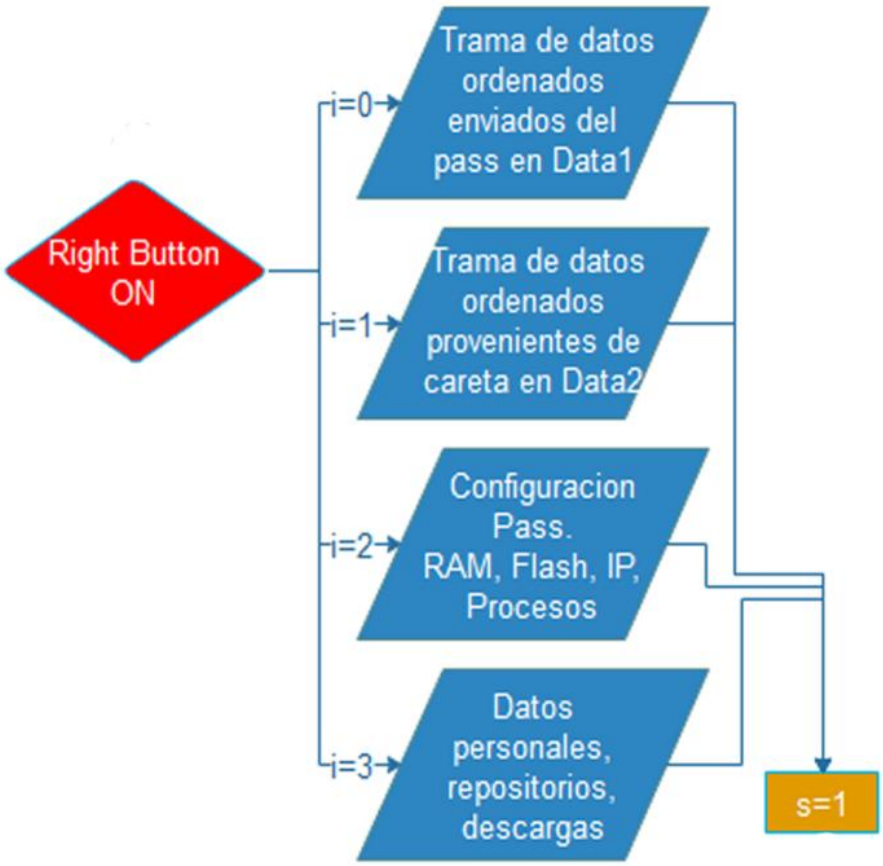
Figura 143. **Diagrama sirena *pass***



Fuente: elaboración propia.

La segunda condición es cuando solamente el pulsador derecho se encuentra presionado. Esto sirve para ingresar a cualquiera de las opciones del menú principal. La manera de mostrar los datos es tomando en cuenta el valor de la variable 'i', que significa índice (es la misma variable que sirve para seleccionar la vista del menú inicial), borrando las vistas en pantalla y mostrando la que corresponde al valor del índice seleccionado. Luego de seleccionar una vista, la variable 'S' cambia de estado bajo a estado alto, por lo cual, al retornar al ciclo, no volverá a cambiar la vista al menú inicial hasta que se decida regresar a este. La siguiente imagen muestra el diagrama de flujo correspondiente a este proceso.

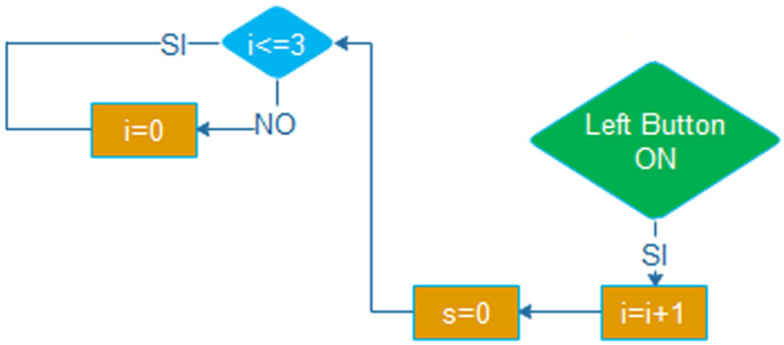
Figura 144. **Diagrama botón rojo *pass***



Fuente: elaboración propia.

La tercera y última condición referente al estado de los pulsadores es para el pulsador izquierdo, el cual lo único que realiza es subir el valor del índice en uno por cada vez que este es presionado y así desplazarnos por el menú principal, y a su vez retornar el estado de la variable 'S' a estado lógico bajo para poder visualizar el menú. Si en caso el índice sobrepasa la cantidad de opciones del menú, el índice retorna a ser cero, por lo cual para navegar en el meno se requiere solamente de un botón. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

Figura 145. **Diagrama botón verde *pass***



Fuente: elaboración propia.

- Hilo 2: El segundo hilo corresponde al bloque de código que sirve para leer los datos del microcontrolador del pass, analizarlos, guardarlos y alertar en caso sea necesario.

La primera parte de este hilo corresponde a la comunicación. Antes de entrar al ciclo se declaran las constantes de temperatura ambiental, humedad y gases que tienen los límites que no se deben sobrepasar. En el caso de la humedad esto no tiene efectos sobre la vida humana, pero si es necesario saberlo para predecir un comportamiento a futuro cercano sobre el fuego. Luego de esto se entra directamente al ciclo del proceso empezando por tomar los datos provenientes del UART definido por software y guardando la trama de datos en 'Data1'. Luego de esto se llega a una condición que, en caso de ser falsa, significa una pérdida de conexión. En este caso, se muestra en la pantalla la perdida de conexión en el área de 'System Info' y se manda una señal audible y sensible al tacto, activando un Buzzer y un vibrador a la vez. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

Figura 146. **Diagrama transmisiones de datos *pass***

Fuente: elaboración propia.

Luego de este punto, al ingresar en el valor verdadero de la segunda condición, se establece que la conexión entre microcontrolador y raspberry pi es afirmativa. Al establecer esta parte se llega a una segunda condición, la cual simplemente desactiva el Buzzer y el vibrador en caso estén funcionando, esto es para que, cuando se recupere la conexión luego de haberse perdido se silencie de nuevo el dispositivo. Luego ingresamos a un bloque que se encarga de guardar los datos netos en la base de datos. Es necesario mencionar que los datos guardados en la base de datos no son procesados, son las lecturas tal y como se toman en los sensores. Esto se hace para tener acceso a la mayor cantidad de datos posibles, tomando también en cuenta los datos erróneos y el ruido como parte de estos para análisis posteriores, ya que para eliminar estos solamente se volverán a aplicar los algoritmos que siguen en el proceso.

Luego de guardar los datos, se aplica un filtro de media móvil a cada uno de los datos provenientes de los sensores, estos datos se guardan en un arreglo llamado 'DatosFiltrados'. Es necesario mencionar que cada uno de los sensores tiene una ventana de datos diferente. Para el caso del sensor

DS18B20, la ventana es de 2 datos, ya que este no necesita un filtro muy amplio; mientras que, para los sensores de gases, la ventana es de 15 datos.

Teniendo los datos ya filtrados se pasa a la siguiente parte del código, la cual procesa los valores obtenidos de los sensores únicamente. Este proceso es simplemente guardar en unas variables definidas las mediciones del proceso anterior e incluirlas en una función con las mediciones del proceso actual. Esto sirve para calcular la pendiente que generan los datos. La pendiente servirá para saber luego el porcentaje en el tiempo respecto al máximo con el que los valores están cambiando. La siguiente imagen muestra el diagrama de flujo correspondiente a este proceso.

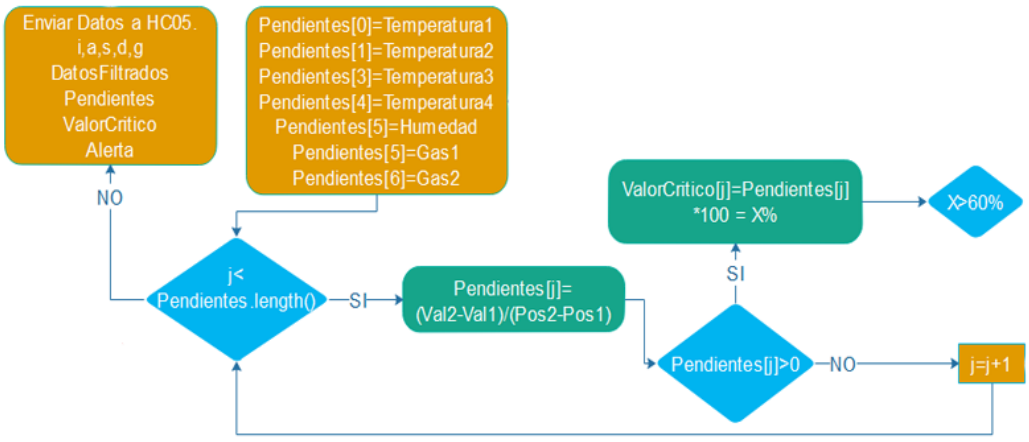
Figura 147. **Diagrama de datos *pass***

Fuente: elaboración propia.

En este punto del proceso, se declara un arreglo con variables definidas en cada una de sus posiciones, y estas variables son las temperaturas, la humedad y gases. Esto es por simplicidad a la hora de transmitir los datos. Luego de esto se entra a otra condición la cual se finaliza cuando se ha pasado por cada una de las posiciones del Array definido anteriormente. Mientras la

condición es verdadera, se calcula y se almacena la pendiente de cada una de las variables para luego entrar a otra condición, esta condición lo único que hace es analizar si la pendiente es positiva o negativa para determinar posibles peligros ambientales. Si la pendiente es negativa en las variables es algo que beneficia en un siniestro, descartando de la humedad (la humedad sirve solamente para fines prácticos de comportamiento estimado), motivo por el cual en pendientes negativas no se analiza nada y solamente se aumenta el contador para pasar a la siguiente posición del arreglo. Si la pendiente es positiva, se obtiene el porcentaje del aumento de la variable medida para tener una mayor facilidad al observar los valores. En este caso se toma como 100 % una pendiente con valor de 1. La siguiente imagen muestra el diagrama de flujo correspondiente a este proceso.

Figura 148. **Diagrama de cálculos *pass***

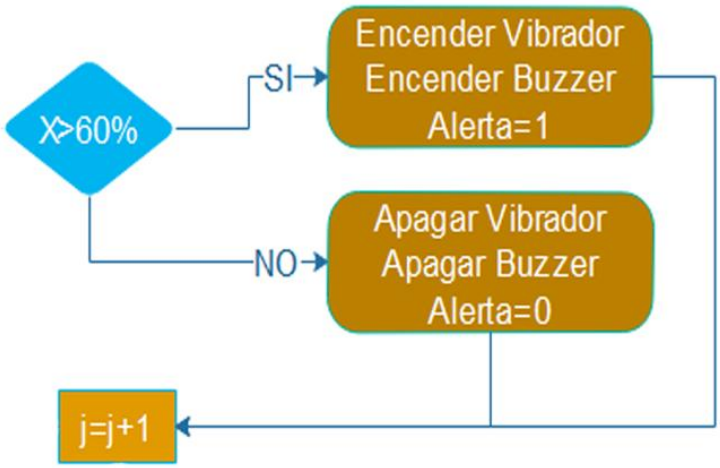


Fuente: elaboración propia.

Por último, se tiene la condición de la alarma automática, la cual funciona tal y como funciona la pérdida de conexión, ya que el caso de una alarma es similar y es solamente para la persona que tiene el dispositivo. El valor de la

condición de alarma es variable y esta se debe ir mejorando mientras se van obteniendo más datos, el diagrama de flujo de la alarma se enciende si la pendiente es mayor al 60 %. En caso contrario la alarma se apaga y simplemente se regresa a aumentar el contador. Es necesario mencionar que el código sigue funcionando mientras la alarma puede estar en estado alto. La siguiente imagen muestra el diagrama de flujo correspondiente a este proceso.

Figura 149. **Diagrama de vibrador y alerta *pass***



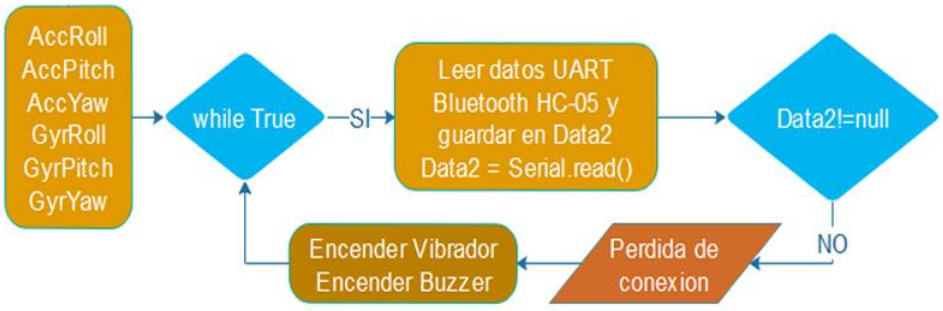
Fuente: elaboración propia.

- Hilo 3: el tercer y último hilo de este diagrama de flujo corresponde al código que hace la función original del pass, el cual a partir de datos del movimiento de una persona pretende activar una alarma luego de un tiempo de movimientos bajos, entendiendo como que la persona que porta el dispositivo esta inconsciente. Este dispositivo normalmente se desactiva generando un movimiento brusco en este. En el caso de este dispositivo funciona de la misma manera, pero el código del *firmware*

está abierto para modificaciones futuras en caso esto requiera un mejor funcionamiento.

Al igual que el segundo hilo la primera parte de este hilo corresponde a la comunicación. Antes de entrar al ciclo se declaran las variables que guardaran los resultados del *string* obtenido desde la careta. Luego de esto se entra directamente al ciclo del proceso empezando por tomar los datos provenientes del UART definido por hardware y guardando la trama de datos en 'Data2'. Luego, se llega a una condición que, en caso de ser falsa, significa una pérdida de conexión. En este caso, se muestra en la pantalla la pérdida de conexión en el área de 'System Info' y se manda una señal audible y sensible al tacto, activando un Buzzer y un vibrador a la vez. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

Figura 150. **Diagrama de lectura de datos y perdida de conexión *pass***



Fuente: elaboración propia.

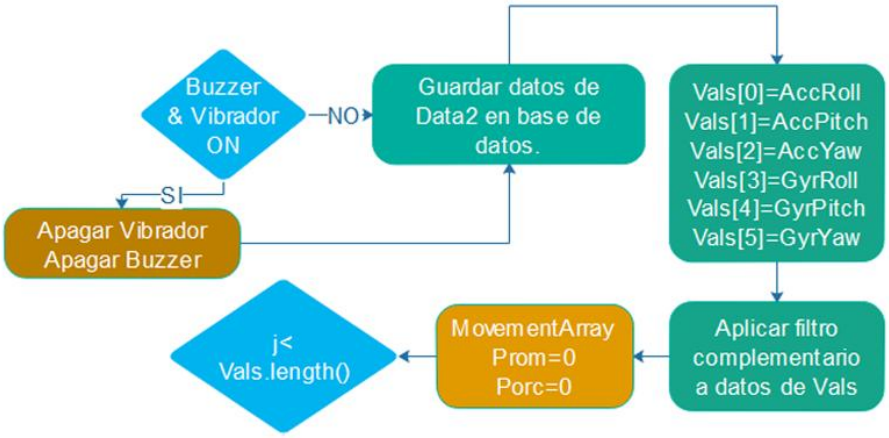
Luego de este punto, al ingresar en el valor verdadero de la segunda condición, se establece que la conexión entre careta y pass. Al establecer esta parte se llega a una segunda condición, la cual simplemente desactiva el Buzzer y el vibrador en caso estén funcionando, esto es para que, cuando se

recupere la conexión luego de haberse perdido se silencie de nuevo el dispositivo. Luego se inggresa a un bloque que se encarga de guardar los datos netos en la base de datos. Es necesario mencionar que, al igual que en el segundo hilo, los datos guardados en la base de datos no son procesados, son las lecturas tal y como se toman en los sensores. Esto se hace para poder tener acceso a la mayor cantidad de datos posibles, tomando también en cuenta los datos erróneos y el ruido como parte de estos para análisis posteriores, ya que para eliminar estos solamente se aplican los algoritmos que siguen en el proceso.

En este punto después de guardar los datos, creamos un arreglo llamado 'Vals', el cual su única función es guardar de manera ordenada los valores de las variables del acelerómetro y el giroscopio para poder procesarlos de una manera más sencilla. Al guardar todos los valores dentro del arreglo, se aplica un filtro complementario a cada una de las 6 posiciones dentro del arreglo. Es necesario que cada una de las 6 posiciones del arreglo 'Vals', contiene otro arreglo dinámico con las mediciones próximas anteriores de las variables.

Luego de aplicar un filtro complementario a los valores dentro del arreglo, se declara otro arreglo denominado 'MovementArray', que servirá para guardar las diferencias entre el promedio y el valor real de la variable en un índice definido. Al terminar esto, se entra a un ciclo que depende de una condición explicada más adelante. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

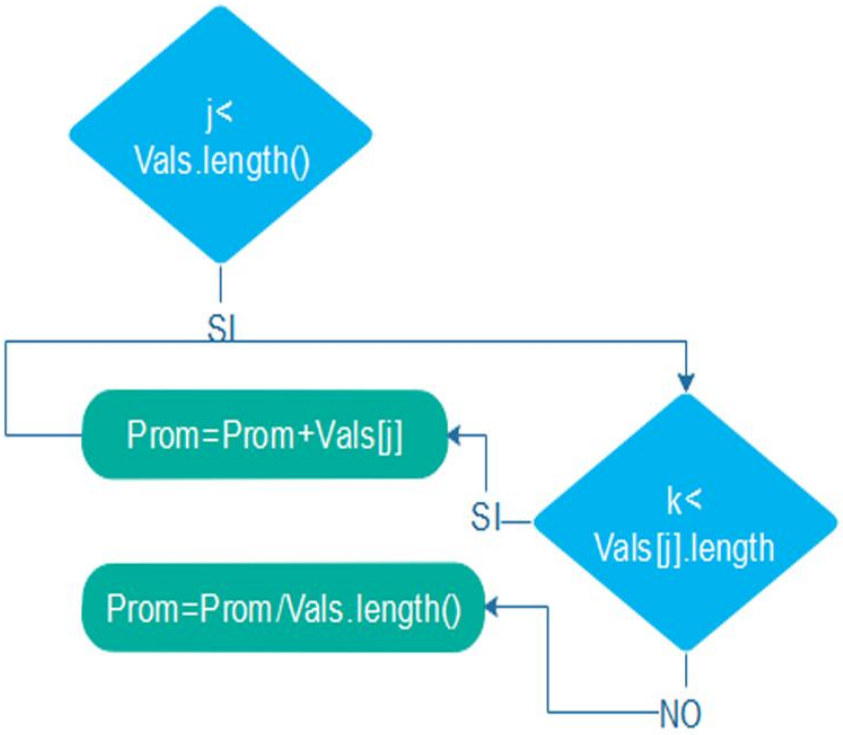
Figura 151. **Diagrama de filtros *pass***



Fuente: elaboración propia.

La condición en el punto actual recorre todo el arreglo 'Vals' con un índice 'j' el cual, determina la posición de cada uno de los arreglos internos. Para ingresar a cada una de las posiciones de estos arreglos internos se requiere otra condición con un índice 'k' que recorre cada posición. A esto se le llama arreglo multidimensional y se requieren 2 índices para ubicar un valor. Al pasar por cada una de las posiciones de los arreglos, se suman todos sus valores y se guardan en otro arreglo de 6 posiciones denominado 'Prom'. Este arreglo lo único que realiza hasta este momento es guardar la sumatoria de los datos de cada arreglo interno dentro del arreglo 'Vals' para que, al terminar de pasar por cada uno de los arreglos internos, se divida dentro de la cantidad total de datos de cada uno de estos arreglos y obtener el promedio. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

Figura 152. **Diagrama de cálculo de promedio *pass***



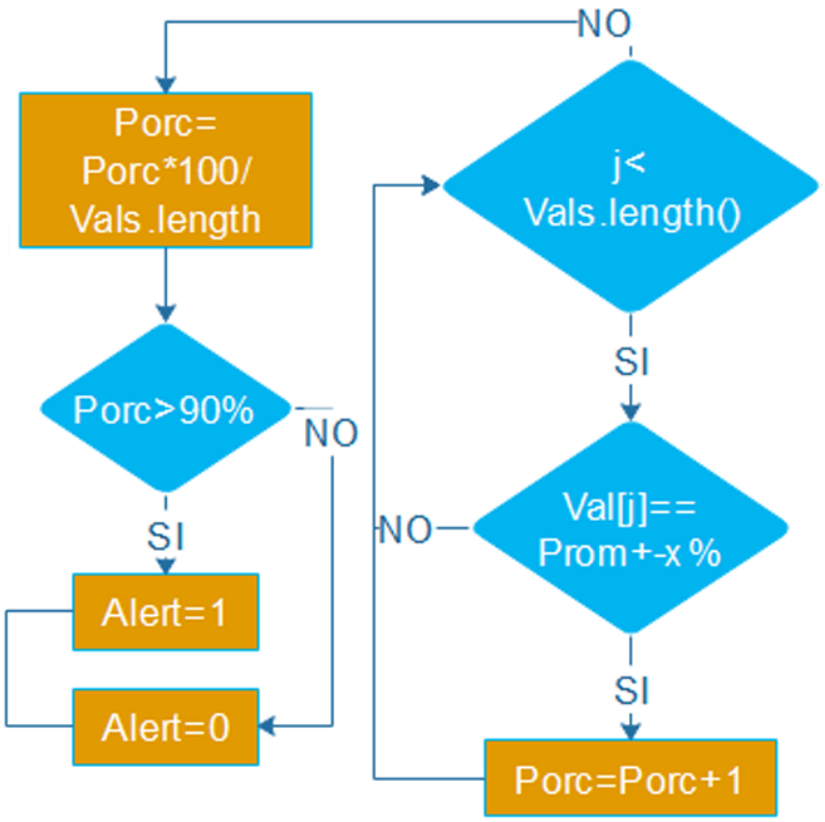
Fuente: elaboración propia.

Ya teniendo un arreglo con promedios, se ingresa a otro ciclo que, de nuevo, pasa por cada una de las posiciones del arreglo 'Vals', solo que en este caso entra a una segunda condición que suma el valor de un porcentaje de error al valor de cada posición del arreglo.

Al terminar este ciclo, se obtiene el porcentaje de datos de cada uno de los 6 arreglos dentro de 'Vars' que cumple con estar dentro del rango delimitado. Si el 90 % de los datos se mantienen dentro de un margen de error, la variable 'Alert' cambia de estado bajo a estado alto, mientras que, si se

mantiene debajo del 90 %, permanece en bajo. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

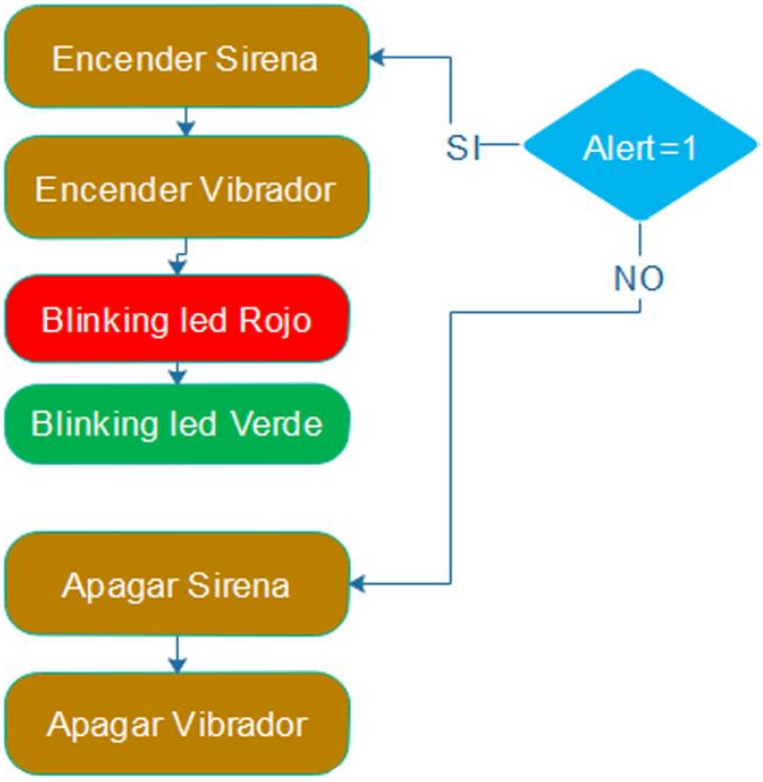
Figura 153. **Diagrama de cálculos *pass***



Fuente: elaboración propia.

Para finalizar se llega a la última condición que, su única función, es activar la alarma y 'Blinking Leds' si la variable 'Alarm' se encuentra en alto, y apagar la alarma si la variable se encuentra en estado bajo. La siguiente imagen muestra el diagrama de bloques correspondiente a este proceso.

Figura 154. **Diagrama de alerta *pass***



Fuente: elaboración propia.

4. PRESUPUESTO

El presupuesto previsto para este desarrollo y mostrado a continuación es referente a los materiales utilizados para crear un prototipo, no se incluyen precios de impresión 3D ni precios de ningún servicio que se necesite. Los precios son basados en el mercado de productos electrónicos de Guatemala en diciembre de 2018.

Tabla V. **Presupuesto**

Dispositivo	Cantidad	Precio Unitario	Precio Total
Arduino Pro Micro Atmega 32U4	2	Q100.00	Q200.00
Baterias LiPo 3.7V 1200mAh 903052	2	Q65.00	Q130.00
Baterias LiPo 3.7V 350mAh 702035	1	Q50.00	Q50.00
Buzzer 5V	1	Q5.00	Q5.00
Cargador lineal Li-ION LTC4056	2	Q45.00	Q90.00
Condensador electrolitico 5v 10uF	1	Q1.00	Q1.00
Disipador de temperatura de cobre	1	Q5.00	Q5.00
Dongle Wifi 802.11n	1	Q80.00	Q80.00
Leds 3mm	2	Q1.00	Q2.00
Memoria Micro SD Clase 10 16GB	1	Q105.00	Q105.00
Mini motor vibrador	1	Q10.00	Q10.00
Mini Switch 3 posiciones	2	Q2.50	Q5.00
Modulo de Sensor MLX90614	1	Q110.00	Q110.00
Modulo de Sensor MPU6050	1	Q45.00	Q45.00
Modulo de Sensor MQ135	1	Q75.00	Q75.00
Modulo de Sensor MQ2	1	Q75.00	Q75.00
Modulo USB a UART CP2102	1	Q55.00	Q55.00
Modulos Bluetooth HC-05	2	Q60.00	Q120.00
Pantalla OLED SH1106 1.12"	1	Q100.00	Q100.00
Pantalla OLED SSD1306 0.96"	1	Q85.00	Q85.00
Pulsadores NA DPST	2	Q2.00	Q4.00
Raspberry Pi Zero W	1	Q135.00	Q135.00
Resistencias 10K Ohms	2	Q1.00	Q2.00
Resistencias 1K Ohms	3	Q1.00	Q3.00
Resistencias 220 Ohms	4	Q1.00	Q4.00
Resistencias 22K Ohms	2	Q1.00	Q2.00
Sensor DHT22	1	Q68.00	Q68.00
Sensor DS18B20	1	Q55.00	Q55.00
Sirena 5-12V con circuito controlador	1	Q140.00	Q140.00
Transistores NPN BC548	3	Q1.00	Q3.00
Cable, metros aproximados	10	Q1.00	Q10.00
Estano, metros aproximados	2	Q0.50	Q1.00
PLA 1.7mm, metros aproximados	40	Q2.50	Q100.00
Placa de cobre-fibra 50mmx80mm	1	Q10.00	Q10.00
Tornillos milimetricos 3mm	20	Q5.00	Q100.00
Total			Q1,985.00

Fuente: elaboración propia.

CONCLUSIONES

1. Para cualquier desarrollo actual referente al área tecnológica actual, es necesario dominar los fundamentos de hardware y de software en general, ya que no existe desarrollo en la actualidad en el que se desarrolle hardware sin firmware o software.

2. Para un buen desarrollo, se implementa una versión inicial de un dispositivo, pero en la mayoría de casos estos dispositivos requieren actualizaciones por el mismo proceso de mejora continua que este ámbito conlleva. Para evitar desechar los dispositivos en caso de actualizaciones, muchas veces se le agregan dispositivos que en algún momento no se utilizan, pero se sabe que pueden llegar a ser útiles en actualizaciones futuras.

3. El comportamiento de la humedad relativa respecto a la temperatura ambiental dentro de un ambiente cerrado no siempre tiende a ser igual. Normalmente, en un incendio, la humedad relativa es inversamente proporcional a la elevación de la temperatura, pero existen casos donde se tienen vapores generados por la combustión que elevan la humedad relativa ambiental. Lo mencionado es en casos en los que aún no se ha iniciado el combate de incendios, ya que del mismo modo se eleva la humedad relativa cuando se lanza un chorro directo a una estructura con una temperatura elevada y se genera vapor de agua, por lo cual esta relación no siempre es igual de estable y se analizan por separado.

4. Para utilizar los datos del acelerómetro y la careta, es necesario filtrar las señales digitales obtenidas antes de normalizar los datos a valores legibles, ya que de no filtrar estos datos antes, se obtienen lecturas no deseadas.

5. El filtro complementario es más útil para utilizarlo en las señales del acelerómetro y el giroscopio, ya que, de utilizar un filtro complementario en este caso, necesitaríamos un proceso mayor para obtener la posición real respecto a ambos sensores. En el caso del filtro complementario, tenemos como limitantes un filtro pasa altos delimitado por el acelerómetro, y un filtro pasa bajos delimitado por el giroscopio para obtener una sola posición en los ejes X y Y.

6. Determinar gráficamente la pendiente de las señales obtenidas ayuda a visualizar una tendencia simple de los datos en un futuro sumamente cercano. En caso de requerir predicciones mas estables, se necesitan conceptos de *machine learning*, los cuales pueden ser agregados en actualizaciones futuras luego de tener mediciones estables en ambientes reales.

RECOMENDACIONES

1. Proponer a las autoridades competentes de la Escuela de Ingeniería Mecánica Eléctrica crear un departamento de apoyo profesional y económico encargado del desarrollo de proyectos profesionales y patentes, con el único objetivo de mejorar el desarrollo de la investigación en la facultad de ingeniería.

2. Revisar el repositorio relacionado con el desarrollo del proyecto para descargar el código original o actualizaciones, junto a documentos o códigos útiles para mejorar el dispositivo o interpretar el análisis de los datos.

3. Utilizar una computadora para el análisis de los datos proporcionados por el sistema y no enfocarse solamente en los datos vistos en la pantalla del dispositivo, ya que un buen análisis de estos datos puede abrir temas de investigación.

4. Intentar en el menor grado posible golpear el dispositivo, mojarlo o exponerlo a una llama directa, ya que cualquier golpe, ingreso de agua al dispositivo o temperatura sumamente alta pueden dañar el dispositivo en general y ocasionarle daños irreversibles.

5. Mantener a una distancia menor a 5 metros la careta y el *pass* siempre, y encender primero la careta seguida del pass, ya que la careta porta el módulo esclavo que será buscado por el maestro y, por ende, el maestro lo porta el pass.

6. Visualizar el estado físico de las baterías antes de utilizar los dispositivos. Si en caso las baterías se encuentran en mal estado, cambiarlas inmediatamente antes de cualquier uso en ambientes reales. Las baterías se dejaron expuestas con un acceso fácil para intercambiarlas rápido en caso se encuentren dañadas.

BIBLIOGRAFÍA

1. *1Wire*. [En línea] <http://web.fdi.ucm.es/posgrado/conferencias/MariaEmiliaCambronero2017-slides.pdf> [Consulta: 7 de septiembre de 2017].

2. *Algoritmo*. [En línea]. <https://es.wikipedia.org/wiki/Algoritmo>. [Consulta: 11 de octubre de 2017].

3. *Arduino*. [En línea] <https://www.adafruit.com/> [Consulta: 7 de septiembre de 2017].

4. *Arduino Pro Micro*. [En línea]. <https://learn.sparkfun.com/tutorials/pro-micro--fio-v3-hookup-guide>. [Consulta: 7 de septiembre de 2017].

5. *AtmelAVR Mega 32U4*. [En línea]. <http://www.atmel.com/Images/Atm el-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf>. [Consulta: 11 de octubre de 2017].

6. *Bases de datos*. [En línea] <http://www.ite.educacion.es/formacion/materiales/93/cd/m1_1/componentes_esenciales_de_una_base_de_datos.html> [Consulta: 7 de septiembre de 2017].

7. *Bateria de litio.* [En línea] <https://es.wikipedia.org/wiki/Bater%C3%ADa_de_pol%C3%ADmero_de_litio> [Consulta: 7 de septiembre de 2017].

8. *Baterías LiPo*. [En línea] <https://www.dynamoelectronics.com/descargas/Baterias_Lipo.pdf> [Consulta: 7 de septiembre de 2017].

9. *Bluetooth HC-06*. [En línea] <https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC06/resources/hc06.pdf> [Consulta: 7 de septiembre de 2017].

10. BRIZUELA, Eduardo A.; DOPAZO, Cesar; ELASKAR, Sergio; FUENTES, Andres; HAUKE, Guillermo; TAMAGNO, José P.; TREVINO, Cesar. *Combustion teoría, aplicaciones e introducción al cálculo*. Argentina: Valletta Ediciones, 2016. 712 p.

11. *Cargador LTC4056*. [En línea] <http://cds.linear.com/docs/en/datasheet/405642f.pdf> [Consulta: 7 de septiembre de 2017].

12. *Comportamiento y fases del fuego*. [En línea]. <http://www.academiadebomberos.org.ar/wpcontent/uploads/2012/07/Cap%202_Fuego_CEMI.pdf>. [Consulta: 4 de noviembre de 2017].

13. *Concepto fuego.* [En línea]. <http://www.fio.unicen.edu.ar/usuario/segumar/Laura/material/Qu%EDmica20del%20Fuego.pdf>. [Consulta: 6 de septiembre de 2017].

14. *CP2102*. [En línea] <https://www.sparkfun.com/datasheets/IC/cp2102.pdf> [Consulta: 7 de septiembre de 2017].

15. *Curso, lucha contra incendios*. [En línea]. <http://emprego.ceei.xunta.gal/export/sites/default/Biblioteca/Documentos/Contidos_Estandar/

prevencion_riscos_laborais/materiais_subvencionados/2015/ASIME_manual_curso_lucha_contra_incendios.pdf>. [Consulta: 4 de enero de 2018].

16. *Desarrollos de incendios en recintos cerrados*. [En línea]. <http://www.olerdola.org/documentos/basset2.pdf>. [Consulta: 11 de octubre de 2017].

17. *DHT22*. [En línea] <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf> [Consulta: 7 de septiembre de 2017].

18. *DS18B20*. [En línea] <https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf> [Consulta: 7 de septiembre de 2017].

19. *Filtro complementario*. [En línea] <http://www2.udec.cl/~fabianinostroza/filtro_comp.pdf> [Consulta: 7 de septiembre de 2017].

20. *Filtro de media móvil*. [En línea] <https://es.wikipedia.org/wiki/Media_movil> [Consulta: 7 de septiembre de 2017].

21. *Firmware y Software*. [En línea] <http://www.alegsa.com.ar/Diccionario/C/2991.php> [Consulta: 7 de septiembre de 2017].

22. HAMMACK DECOSTE, Bill; FARADAY'S, Michael. *The chemical History of a candle –& Don*. Illinois: Urbana, 2011. 200 p.

23. *I2C*. [En línea] <http://www.uco.es/~el1mofer/Docs/IntPerif/Bus%20I2C.pdf> [Consulta: 7 de septiembre de 2017].

24. *LC9801*. [En línea] <http://akizukidenshi.com/download/ds/sctech/LC9806.pdf> [Consulta: 7 de septiembre de 2017].

25. *Librerias de Software utilizadas*. [En línea] <http://librosweb.es/libro/python/capitulo10/modulosdesistema.html> [Consulta: 7 de septiembre de 2017].

26. *Límites de inflamabilidad o explosividad*. [En línea]. <http://cfbt-be.com/images/artikelen/artikel_34_ES.pdf>. [Consulta: 4 de noviembre de 2017].

27. *Manual combate de incendios*. [En línea]. <http://emprego.ceei.xunta.gal/export/sites/default/Biblioteca/Documentos/Contidos_Estandar/prevencion_riscos_laborais/materiais_subvencionados/2015/ASIME_manual_curso_lucha_contra_incendios.pdf>. [Consulta: 4 de noviembre de 2017].

28. *Microprocesador*. [En línea]. <https://www.raspberrypi.org/products/raspberry-pi-zero-w/>. [Consulta: 4 de enero de 2018].

29. *MLX90614*. [En línea] <https://www.pololu.com/file/download/MLX90614.pdf?file_id=0J170> [Consulta: 7 de septiembre de 2017].

30. *MPU6050*. [En línea] <https://store.invensense.com/datasheets/invensense/MPU-6050_DataSheet_V3%204.pdf> [Consulta: 7 de septiembre de 2017].

31. *MQ-135*. [En línea] <https://www.olimex.com/Products/Components/Sensors/SNS-MQ135/resources/SNS-MQ135.pdf> [Consulta: 7 de septiembre de 2017].

32. *MQ-2.* [En línea] <https://www.pololu.com/file/0J309/MQ2.pdf> [Consulta: 7 de septiembre de 2017].

33. *MySQL*. [En línea] <https://es.wikipedia.org/wiki/MySQL> [Consulta: 7 de septiembre de 2017].

34. *Paradigmas de programación*. [En línea] <https://www.genbetadev.com/metodologias-de-programacion/programacion-imperativa-vs-declarativa-i> [Consulta: 7 de septiembre de 2017].

35. *Python*. [En línea] <https://es.wikipedia.org/wiki/Python> [Consulta: 7 de septiembre de 2017].

36. *Raspbian-Jessie*. [En línea]. <https://www.raspberrypi.org/blog/ raspbian-jessie-is-here/>. [Consulta: 4 de noviembre de 2017].

37. *Referencia arduino*. [En línea] <https://www.arduino.cc/en/Reference> [Consulta: 7 de septiembre de 2017].

38. *Sensores MQ*. [En línea] <http://www.naylampmechatronics.com/blog/42_Tutorial-sensores-de-gas-MQ2-MQ3-MQ7-y-MQ13.html> [Consulta: 7 de septiembre de 2017].

39. *SH1106*. [En línea] <https://www.elecrow.com/download/SH1106%20datasheet.pdf> [Consulta: 7 de septiembre de 2017].

40. *SPI*. [En línea] <http://panamahitek.com/como-funciona-el-protocolo-spi/> [Consulta: 7 de septiembre de 2017].

41. *SSD1306*. [En línea] <https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf> [Consulta: 7 de septiembre de 2017].

42. *Tabla de base de datos*. [En línea] <https://basededatosunounivia.wordpress.com/2015/03/13/cuales-son-las-partes-de-una-tabla-de-base-de-datos/> [Consulta: 7 de septiembre de 2017].

43. *UART*. [En línea] <http://www.ti.com/lit/ug/sprugp1/sprugp1.pdf> [Consulta: 7 de septiembre de 2017].

44. *Wireless Raspberry Pi Zero W*. [En línea] <https://www.raspberrypi.org/magpi/pi-zero-w-wireless-antenna-design/> [Consulta: 7 de septiembre de 2017].

APÉNDICE

Apéndice 1. **Filtros digitales**

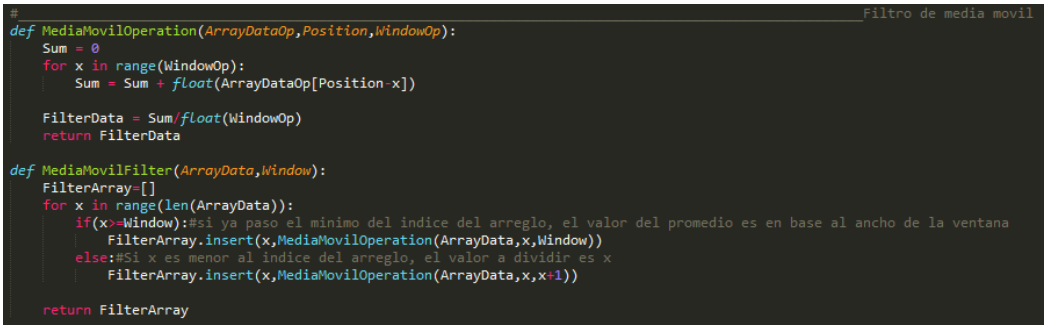
A continuación, se encuentra el código realizado para el filtro de media móvil y filtro complementario, los cuales se utilizaron para filtrar los datos obtenidos por los sensores.

Para el filtro de media móvil, se separó en 2 funciones para hacerlo más eficiente y fácil de entender. La primera función solamente contiene una condición para el divisor del filtro, siendo el divisor igual al tamaño de la ventana del filtro en caso el muestreo se encuentre en un índice igual o superior a la ventana. Si el índice es menor a la ventana de datos, el divisor es el mismo índice. Esto afecta la estabilidad del filtrado de los datos exponencialmente desde cero hasta el valor antes de que se iguale el índice con la ventana.

La segunda función es la que realmente realiza la función de filtrado. Es necesario mencionar que, los únicos 2 parámetros de entrada que solicita la función del filtro es un arreglo de datos puros, y el ancho de la ventana de datos. Lo que devuelve el filtro es un arreglo filtrado de la misma dimensión que el arreglo ingresado. La siguiente imagen muestra el extracto de código desarrollado para este filtro.

Continuación del apéndice 1.

Figura A 1



```
#                                                                                    Filtro de media movil
def MediaMovilOperation(ArrayDataOp,Position,WindowOp):
    Sum = 0
    for x in range(WindowOp):
        Sum = Sum + float(ArrayDataOp[Position-x])

    FilterData = Sum/float(WindowOp)
    return FilterData

def MediaMovilFilter(ArrayData,Window):
    FilterArray=[]
    for x in range(len(ArrayData)):
        if(x>=Window):#si ya paso el minimo del indice del arreglo, el valor del promedio es en base al ancho de la ventana
            FilterArray.insert(x,MediaMovilOperation(ArrayData,x,Window))
        else:#Si x es menor al indice del arreglo, el valor a dividir es x
            FilterArray.insert(x,MediaMovilOperation(ArrayData,x,x+1))

    return FilterArray
```

En cuanto al filtro complementario, se ejecuta en una función totalmente independiente al filtro de media móvil, solo que esta función, a pesar de ser muy simple, necesita un feedback de la variable que se encuentra guardando el retorno del filtro. La siguiente imagen muestra la función principal del filtro complementario.

Figura A 2

```
#                                                                                    Filtro Complementario
def ComplementaryFilter(Ang,Gy,Acc):
    Angle = 0.98*(Ang+Gy*0.010) + 0.02*Acc

    return Angle
```

El retorno del filtro complementario se encuentra en la ejecución principal del programa. La siguiente imagen muestra la variable retornándose dentro de la función del filtro.

Continuación del apéndice 1.

Figura A 3

```
for x in range(len(IDAcc)):
    Angle = ComplementaryFilter(Angle,GyrRDeg[x],AccRDeg[x]) #Lazo cerrado de la variable Angle
    AngleR.insert(x,Angle)
Angle=0
```

Fuente: elaboración propia.

Apéndice 2. **Código de cálculo de pendientes**

Las pendientes en este desarrollo han servido para obtener un porcentaje y un nivel sobre el cual se está elevando una variable dentro de un ambiente y tomar decisiones a partir de esto. Por ejemplo, si la pendiente es positiva y muy elevada en el caso de la temperatura, significa que el incendio está progresando de una manera fuera de control.

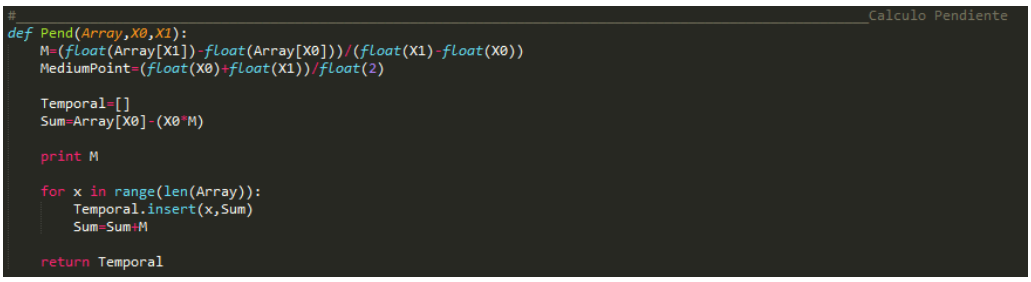
La función que calcula la pendiente solicita únicamente el arreglo con todos los datos de la función a graficar, X0 y X1, donde X0 es el punto inicial donde se medirá la pendiente, mientras que X1 es el segundo punto. La pendiente no siempre se mide entre 2 puntos consecutivos, ya que algunas variables tienen variaciones grandes en lapsos de tiempo muy pequeños o inclusive algunas variables requieren un tiempo más largo que otras para poder determinar si en realidad son un peligro o no.

La primera parte del código dentro de la función calcula la pendiente entre los puntos Y dentro del arreglo y los puntos X. luego de esto se calcula un punto medio, esto, para fines prácticos si se necesita, calcula el punto al medio de X0 y X1 para mostrar el lapso de tiempo que se está midiendo.

Continuación del apéndice 2.

Luego se declara un arreglo temporal que guardara todos los puntos de la función a graficar, seguido de una variable Sum, la cual contiene la posición en el eje Y inicial de la pendiente. Este punto inicial está dado por la posición del arreglo en el primer punto X, restándole todos los valores de pendiente anteriores a este. La siguiente imagen muestra el bloque de código correspondiente a el desarrollo mencionado.

Figura B 1



```
#_____Calculo Pendiente
def Pend(Array,X0,X1):
    M=(float(Array[X1])-float(Array[X0]))/(float(X1)-float(X0))
    MediumPoint=(float(X0)+float(X1))/float(2)

    Temporal=[]
    Sum=Array[X0]-(X0*M)

    print M

    for x in range(len(Array)):
        Temporal.insert(x,Sum)
        Sum=Sum+M

    return Temporal
```

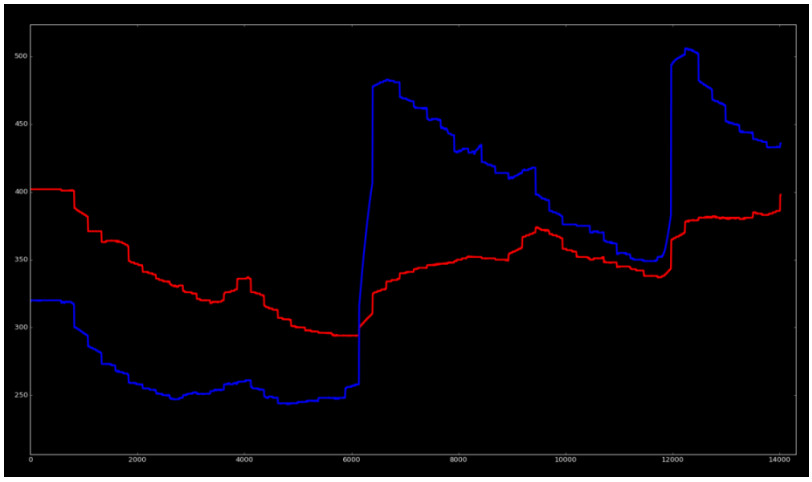
Fuente: elaboración propia.

Apéndice 3. **Gráficos de gases**

Estas mediciones de gases fueron realizadas en un ambiente cerrado simulando una fuga real de CO2 seguida de una fuga de alcohol. En estas mediciones se obtuvieron dos picos a partir del sensor MQ-135> uno a partir de la fuga de CO2 y otro a partir de la fuga de alcohol, el pico fue más alto en la medición de CO2. El sensor MQ-2 muestra variaciones que dependen de la densidad del humo en el sitio medido. La siguiente imagen muestra el comportamiento de estos datos; el gráfico azul es el de un sensor MQ-135 y el gráfico rojo de un sensor MQ-2.

Continuación del apéndice 3.

Figura C 1



Fuente: elaboración propia.

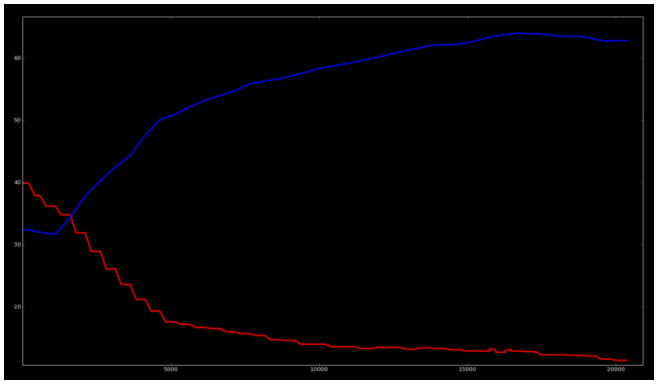
Apéndice 4. **Humedad vs temperatura**

La humedad y la temperatura son las variables de las cuales dependen la mayoría de las alarmas, ya que los gases, a pesar de ser nocivos, no afectan a un bombero que lleva todo su equipo de protección colocado.

Estas mediciones se realizaron de igual manera que la de los gases, en un ambiente cerrado simulando un incendio dentro de una habitación estando en la parte posterior de la habitación, en donde la temperatura es más baja. Mientras la temperatura aumenta, la humedad disminuye y el fuego se propaga más rápido. Mientras más rápido se disminuye la humedad, más fácil se propaga el fuego y, por ende, más rápido llega a un punto máximo la temperatura. En este ejemplo, el punto máximo no es tan alto (70 grados centígrados aprox.) dado que el ambiente cerrado era en una escala pequeña, donde la temperatura es

proporcional al combustible y al volumen donde se propaga el fuego. La siguiente imagen muestra el gráfico de comparación, donde el gráfico azul es la temperatura y el gráfico rojo es la humedad

Figura D 1



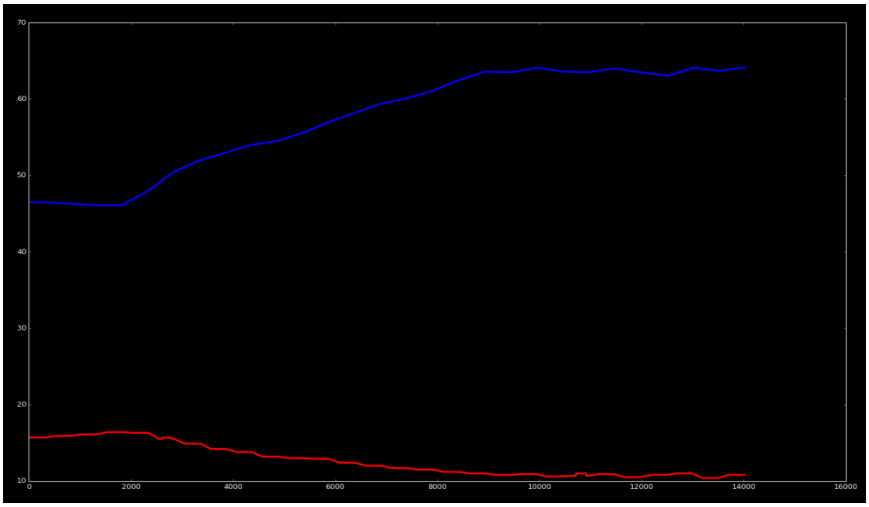
Fuente: elaboración propia.

Apéndice 5. **Pendientes**

Los gráficos de pendientes son útiles para estimaciones y alarmas de evacuación de un sitio. En la siguiente imagen se muestra un gráfico de humedad y temperatura en el cual, la temperatura y la humedad se encuentran más estables solamente para fines prácticos. La temperatura es mostrada en un gráfico azul mientras que la humedad se muestra en un gráfico rojo.

Continuación del apéndice 5.

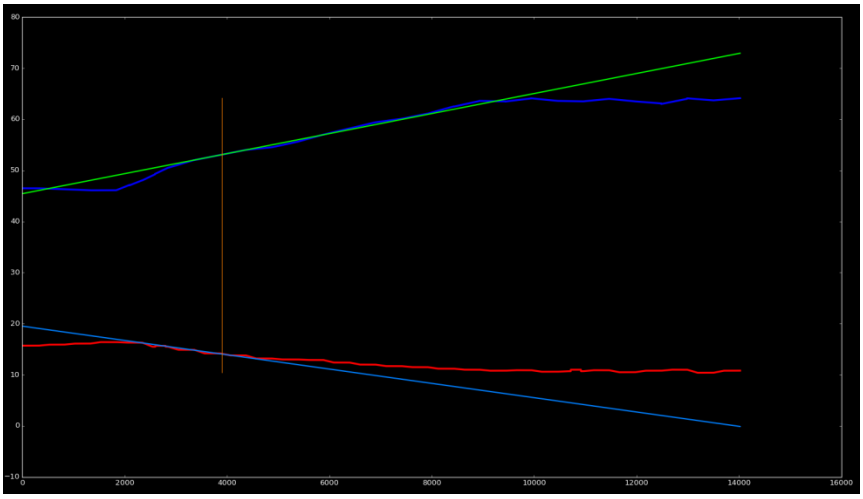
Figura E 1



En relación a la imagen anterior, se muestran los gráficos de las pendientes y una línea de tiempo. La siguiente imagen muestra en color verde la pendiente de la temperatura y en azul claro la pendiente de la humedad. Ambas pendientes fueron medidas en el lapso de tiempo mostrado con la línea vertical de color naranja, con una ventana de 20 datos cerca de los 40 segundos hacia cada lado de la línea de tiempo.

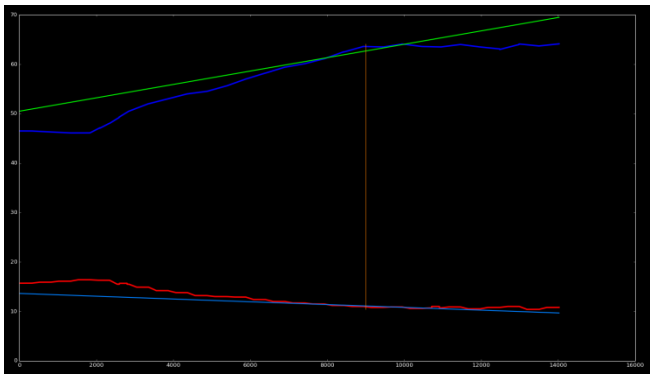
Continuación del apéndice 5.

Figura E 2



La siguiente imagen, muestra las pendientes de las variables con una ventana de 20 datos hacia cerca de los 90 segundos hacia cada lado de la línea de tiempo.

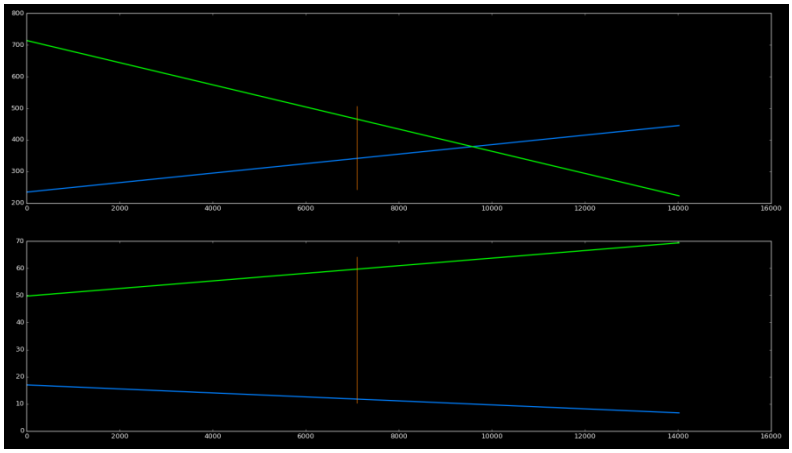
Figura E 3



Continuación del apéndice 5.

La siguiente imagen muestra solamente el comportamiento de las pendientes en un punto cercano a los 70 segundos.

Figura E 4



Fuente: elaboración propia.