UÍDE

UNIVERSIDAD INTERNACIONAL DEL ECUADOR
SEDE LOJA

ESCUELA DE INGENIERÍA EN INFORMÁTICA Y MULTIMEDIA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN INFORMÁTICA Y MULTIMEDIA

DESARROLLO DE UN DISPOSITIVO PARA TRADUCIR LENGUAJE CÓDIGO MORSE A EXPRESIÓN SONORA, PARA PERSONAS CON DISCAPACIDAD SENSORIAL

JOSÉ FRANCISCO CALVA MERINO

DIRECTOR:

MGS. LUIS CUENCA

FEBRERO 2017

LOJA – ECUADOR

Yo, José Francisco Calva Merino declaro bajo juramento, que el trabajo aquí descrito es de mi autoría; que no ha sido presentado anteriormente para ningún grado o calificación profesional y que se ha consultado la bibliografía detallada.

Cedo mis derechos de propiedad intelectual a la Universidad Internacional del Ecuador, para que sea publicado y divulgado en internet, según lo establecido en la Ley de Propiedad Intelectual, reglamento y leyes.

José Francisco Calva Merino
C.I. 1104905433

Yo, Ing. Luis Cuenca, certifico que conozco al autor del presente trabajo siendo él responsable exclusivo tanto de su originalidad y autenticidad, como de su contenido.

Ing. Luis Cuenca
DIRECTOR DE TESIS

A Dios, por bendecir mi camino, por concederme salud y fortaleza para alcanzar mis objetivos.

A mis padres: Segundo Calva y Blanca Merino, porque sus sabios consejos me han permitido superar cada etapa de la vida; porque son mi mayor fuerza e inspiración.

A mi hermana Jéssica Calva, por apoyarme en momentos difíciles.

A Katherine, por ser pilar fundamental de mi existencia.

A la Universidad Internacional del Ecuador, a sus Autoridades y a todo su Cuerpo de Docentes, por brindarme los conocimientos y enseñanzas necesarias para forjarme como persona y profesional.

Y un reconocimiento especial al Mgs. Luis Cuenca M., por su generosa y desinteresada colaboración con sus asesorías y consejos.

José Francisco

Resumen

Siguiendo el canon que establece la Metodología de Desarrollo de Hardware libre, se diseñó e implementó el dispositivo para traducir Código Morse a expresión sonora para personas con discapacidad sensorial visual—auditiva.

La elaboración del dispositivo electrónico (físico), se fundamentó en principios de electrónica para el ensamblado de componentes. La elaboración del software para el dispositivo electrónico (lógica), precisó la representación del Código Morse: *dit* como 1 y *dah* como 0, asistido de las técnicas de programación: a) bit de descarte, b) desplazamiento de bits.

El dispositivo ofrece dos salidas de información: a) Para personas con discapacidad visual, se dispone de altavoces que permiten escuchar la digitación del Código Morse y la traducción de texto a audio; b) Para personas con discapacidad auditiva, se dispone de una pantalla LCD que muestra los datos digitados por el pulsador “Morse”.

La realización de este proyecto impulsa una forma alternativa de comunicarse para personas con discapacidad sensorial visual—auditiva, mediante el apoyo de la tecnología.

Palabras Clave: Arduino Uno, Emic2 Text-to-Speech, Código Morse, Metodología de Hardware Libre, discapacidad sensorial, C++, desplazamiento de bits, bit de descarte

Abstract

Following the canon established by the Free Hardware Development Methodology, the device for translating Morse Code to sound expression for people with visual-auditory sensory impairment was designed and implemented.

The elaboration of the electronic device (physical), was based on principles of electronics for the assembly of components. The development of the software for the electronic device (logic) required the representation of the Morse Code: dit as 1 and dah as 0, assisted by programming techniques: a) a dismissing bit, b) bit shifting.

The device offers two outputs: a) For people with visual impairments, speakers are offered to listen to Morse Code fingering, and text to audio translation; B) For people with hearing impairment, an LCD screen is available that shows the data entered by the "Morse" button.

The realization of this project promotes an alternative way of communicating for people with visual-auditory sensory disabilities through the support of technology.

Keywords: Arduino Uno, Emic2 Text-to-Speech, Morse Code, Free Hardware Methodology, sensory impairment, C++, dismissing bit, bit shifting

DESARROLLO DE UN DISPOSITIVO PARA TRADUCIR LENGUAJE CÓDIGO MORSE A EXPRESIÓN SONORA, PARA PERSONAS CON DISCAPACIDAD SENSORIAL.

Índice de Tablas

Índice de Ilustraciones

Índice de Anexos

Introducción

La necesidad que tenemos los seres humanos para comunicarnos y expresarnos con el resto de la sociedad es de vital importancia. A lo largo de la historia, personas con discapacidades en la comunicación, se han visto olvidadas y hasta despreciadas de diferentes maneras.

Las personas con discapacidad sensorial visual—auditiva, requieren de atención específica, es por ello que con el surgimiento de las Tecnologías de Información (TICS), se ha podido generar proyectos de vinculación y de protección que rompen las barreras espaciales de años atrás.

Las TICS son muy útiles, se las puede adaptar para potenciar las capacidades de las personas con necesidades especiales.

La accesibilidad de dispositivos electrónicos y aplicaciones, facilitan la calidad de comunicación de personas con discapacidad sensorial visual—auditiva, mejorando su autonomía y su estima personal.

Con el dispositivo para traducir Lenguaje Código Morse a expresión sonora, se pretende proveer una alternativa diferente de comunicación para personas con discapacidad visual—auditiva.

Capítulo 1. Análisis

1.1. Planteamiento del problema.

Las personas con discapacidad sensorial visual—auditiva, no disponen de alternativas tecnológicas a la hora de comunicarse con otras personas que no se encuentren en su entorno de comunicación. No obstante, al existir una gran variedad de lenguajes dactilológicos (comunicación mediante el uso de los dedos de la mano), la comunicación verbal sigue siendo el principal problema para personas con discapacidad sensorial visual—auditiva.

Considerando el gran avance tecnológico en los mecanismos de comunicación, para el grupo de personas con discapacidad sensorial visual—auditiva, no es fácil ni asequible obtener dispositivos que apoyen su comunicación o que les ayude a procesos del aprendizaje.

En este contexto y en la actualidad, en nuestro país, Ecuador, las personas con discapacidad sensorial visual—auditiva siguen sufriendo algún tipo aislamiento, debido principalmente a la imposibilidad de expresarse ante la sociedad normal, originando que no puedan ejercer su derecho de expresarse.

Existen dispositivos para traducir Código Morse a lenguaje escrito y expresión sonora equivalente, se manejan mediante dispositivos con acceso a Internet, pero no son portables y sin acceso a Internet no funcionan, impidiendo que se pueda disponer de ellos todo el tiempo en cualquier lugar.

La carencia de dispositivos electrónicos portátiles que faciliten la comunicación verbal entre personas con discapacidad sensorial visual—auditiva y la sociedad en general —que desconoce el lenguaje de señas—, representa el problema a solucionarse en el presente trabajo de investigación.

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un dispositivo para traducir lenguaje Código Morse a expresión sonora, para personas con discapacidad sensorial.

1.2.2. Objetivos Específicos

- Analizar y establecer los diferentes requerimientos necesarios para el desarrollo dispositivo.
- Aplicar la Metodología desarrollo de hardware libre para el desarrollo e implementación del proyecto.
- Diseñar el prototipo de hardware del dispositivo.
- Realizar un algoritmo de traducción de Código Morse a Lenguaje Natural.
- Programar el software basado en el algoritmo para la traducción de Código Morse a lenguaje natural y lenguaje sonoro.
- Realizar las pruebas de comportamiento entre el software y el hardware del dispositivo.

1.3. Alcance del Proyecto

Desarrollar un dispositivo electrónico que traduzca Lenguaje Código Morse a expresión sonora, para personas con discapacidad sensorial visual—auditiva.

Con el dispositivo electrónico se pretende mejorar ampliamente la comunicación de personas con discapacidad sensorial visual—auditiva en el entorno familiar y social.

El dispositivo electrónico deberá estar precargado con el software capaz de administrar el hardware, y, aplicar satisfactoriamente el algoritmo de conversión de Código Morse a Lenguaje formal.

El dispositivo electrónico deberá ser portátil, constará de tres botones: 1) Un botón para ingresar las pulsaciones correspondientes al Código Morse, 2) Un botón para borrado de caracteres, y, 3) Un botón que realice la traducción del texto ingresado a lenguaje natural audible.

El dispositivo electrónico, además contará con: 1) Una pantalla LCD 16x2 caracteres con la finalidad de mostrar la información digitada; 2) Un Buzzer (Zumbador) que emitirá los tonos correspondientes al punto (dit) y raya (dah); y, 3) un Altavoz que representará con lenguaje audible, el texto mostrado en la pantalla LCD.

Finalmente se realizarán las pruebas de comportamiento, y se expondrán los resultados, conclusiones, y recomendaciones, obtenidas durante el transcurso de la investigación del presente proyecto.

Capítulo 2. Marco Referencial

La era de las nuevas vanguardias de información y tecnológicas, han servido como instrumento potenciador para el desarrollo y evolución de muchas disciplinas, entre ellas, los avances en el ámbito de comunicación, beneficiando a personas con alguna discapacidad o problemas sensoriales. Las herramientas tecnológicas actuales permiten superar algunos problemas de comunicación que se han presentado en el trascurso de la historia.

2.1. Definiciones

2.1.1. Deficiencia, Discapacidad, Minusvalía

Conceptualizando y delimitando algunos términos, en la actualidad el sistema de clasificación más aceptado es el propuesto por la Organización Mundial de la Salud (OMS) en la Clasificación Internacional de Deficiencias, Discapacidades y Minusvalías (CIDDM); ésta clasificación define tres términos que considera los más importantes dentro de la experiencia de la salud.

- **Deficiencia:** “Toda pérdida o anormalidad de una estructura o función psicológica, fisiológica o anatómica, puede ser temporal o permanente”

- **Discapacidad:** “Toda restricción o ausencia (debida a una deficiencia) de la capacidad para realizar una actividad en la forma o dentro del margen que se considera normal para un ser humano”.

- **Minusvalía:** “Es la situación desventajosa para un individuo determinado, como consecuencia de una deficiencia o discapacidad, que limita o impide el desempeño

de un rol que es normal en su caso en función de su edad, sexo y factores sociales y culturales". (Cáceres Rodríguez, 2004)

2.1.2. Tecnologías de la Información y la Comunicación

Citando a (Annan, 2003), "Las tecnologías de la información y la comunicación no son ninguna panacea ni fórmula mágica, pero pueden mejorar la vida de todos los habitantes del planeta. Se dispone de herramientas para llegar a los Objetivos de Desarrollo del Milenio, de instrumentos que harán avanzar la causa de la libertad y la democracia y de los medios necesarios para propagar los conocimientos y facilitar la comprensión mutua"; entonces, en otras palabras, las TIC abarcan los recursos necesarios para aprovechar las computadoras, las aplicaciones informáticas y las redes necesarias para convertir, almacenar, administrar, transmitir y encontrar la información.

Dentro de las técnicas y dispositivos que conciernen al presente trabajo de investigación, están los dispositivos de entrada, dispositivos de salida, y dispositivos de entrada/salida de la información:

2.1.2.1. Dispositivos de Entrada de la Información

• **Adaptaciones al teclado o Carcasas:** Evitan que el usuario pulse una tecla indeseada, para ello una solución es situar carcasas en el teclado de modo que solo pueda accederse a determinadas teclas.

• **Conmutadores o Pulsadores:** Se adaptan a las necesidades de los usuarios que presentan discapacidades motoras o mentales. Su funcionamiento es

semejante al de un interruptor, activándose únicamente cuando se actúa sobre ello, para ello se utilizan diferentes técnicas: soplo, sonido, presión, etc.

2.1.2.2. Dispositivos de Salida de la Información

- **Pantalla LCD:** (Acrónimo del inglés Liquid Crystal Display) Es una pantalla delgada y plana formada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora.

- **Sintetizador de voz:** Convierte en voz cualquier texto escrito en la pantalla. La voz es creada partiendo de algoritmos de programación. No presenta ningún límite en el vocabulario y frases que se pueden producir.

2.1.2.3. Dispositivos de Entrada/Salida de Información

- **Comunicador Morse:** Dispositivo a desarrollarse que permitirá a las personas con discapacidades visuales—auditivas comunicarse, valiéndose de pulsadores y del alfabeto Morse.

2.1.3. Ventajas de las TIC en personas con Discapacidad

- Las TIC permiten que personas con discapacidad puedan expresarse o comunicarse.

- Las TIC permiten el proceso educativo sin ningún límite de tiempo, y facilita la formación y comunicación de personas con discapacidad.

- Las TIC potencian el proceso de enseñanza—aprendizaje de personas con discapacidad, lo que les ayuda a integrarse a la sociedad.

- La enseñanza a través de las TIC resulta favorable para este colectivo, ya que se adapta a sus necesidades y a su ritmo de aprendizaje sin perjudicar a ningún tipo de persona.

2.1.4. Código Morse

El Código Morse fue desarrollado por Alfred Vail durante la invención del telégrafo eléctrico con Samuel Morse en 1830.

Según (Guenther, 1973) el Código Morse es un esquema de codificación binario rudimentario para el lenguaje, en el que cada carácter está representado por una secuencia única de pulsos y espacios. El alfabeto del Código Morse Internacional se aprecia en el Anexo H.

En el alfabeto Morse se utilizan dos tipos de pulsos DOT (punto), DASH (raya) y tres tipos de espacio: SÍMBOLO, CARÁCTER y PALABRA, Todos estos pulsos se distinguen por sus duraciones de tiempo relativas. La duración del punto es la mínima posible. Una raya tiene una duración aproximada de tres veces la del punto. Los caracteres se definen por una secuencia de pulsos separados por espacios. El espacio de SÍMBOLOS separa los pulsos entre carácter; el espacio de CARACTER separa los caracteres entre palabras, y el espacio de PALABRAS separa las palabras.

En unidades de tiempo, las definiciones aceptadas para los pulsos y espacios son:

Tabla 1: Unidades de tiempo relativas del Código Morse

Recurso	Tiempo
Pulso DOT (Punto)	1
Pulso DASH (Raya)	3
Espacio de SIMBOLOS	1
Espacio de CARACTER	3
Espacio de PALABRAS	5 - 7

Fuente: (Guenther, 1973)
Elaboración: El Autor

El promedio de velocidad de digitación de Código Morse es de 5 caracteres por minuto, medido en términos de palabras por minuto (words per minute wpm).

2.2. Herramientas de Desarrollo

Para la realización de la aplicación se utilizó las siguientes herramientas:

2.2.1. IDE Arduino

El entorno de desarrollo integrado (IDE acrónimo del inglés Integrated Development Enviroment) de Arduino, contiene un editor de texto para escribir código, una consola de texto, y una serie de menús con las funciones comunes. Entre las características imponentes del IDE se encuentran:

Verificar: Comprueba posibles errores existentes en el código y compila el programa.

Subir: Compila el código, y lo sube a la placa Arduino.

Monitor Serial: Muestra datos seriales enviados entre la placa Arduino (USB o serial).

Se prefirió este entorno de desarrollo porque ofrece soporte nativo para la mayoría de placas (Arduino Uno, Mega, Leonardo) y por su simplicidad a la hora de subir las aplicaciones y monitorear las aplicaciones en la placa Arduino.

2.2.1.1. Librería LiquidCrystal_I2C

Esta librería permite controlar, a la placa Arduino, una pantalla LCD i2c basada en el chipset PCF85741 o compatible, que se encuentra presente en la mayor parte los dispositivos disponibles en el mercado.

2.2.2. Fritzing

Es una herramienta de código abierto para crear diseños electrónicos tales como: prototipos de electrónica y placas impresas de circuito (PCB).

Permite:

* Verificar la continuidad de los cables conectados a los componentes (en modo diseño).

* Agregar componentes electrónicos.

Fritzing no es un simulador, pero la versión más reciente, guarda el código de diseño y lo carga directamente al dispositivo Arduino. Cuenta, además, con una versión de escritorio y una versión web.

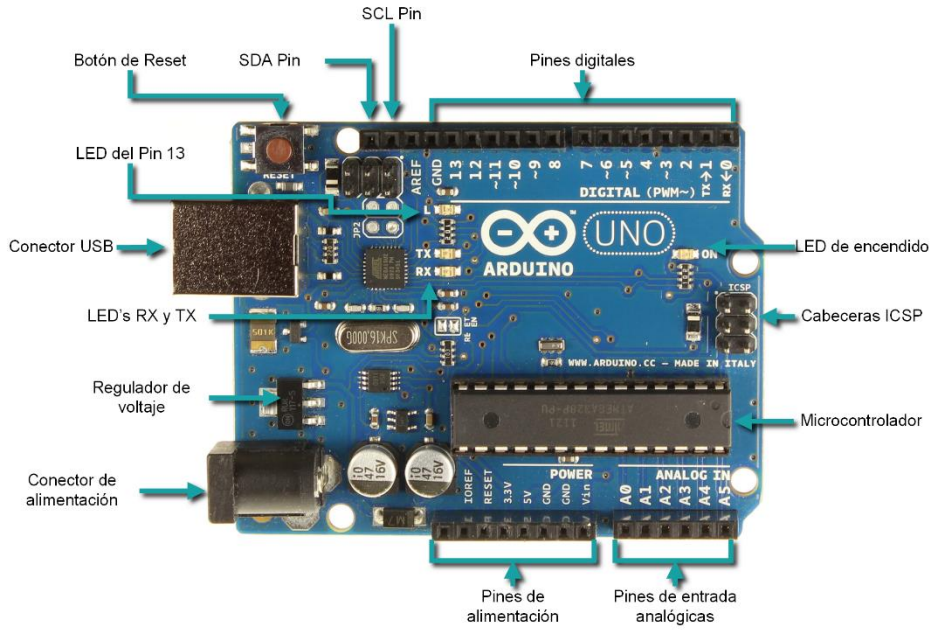
Se prefirió usar Fritzing, por ser una herramienta libre sustentada en principios de Processing y Arduino, lo que permitió generar el prototipo basado en Arduino y crear el esquema de circuitos para su posterior fabricación.

2.3. Hardware necesario para el desarrollo

2.3.1. Placa Arduino Uno

La placa Arduino Uno es un microcontrolador que dispone de 14 pines de entrada/salida (E/S) digital (de los cuales 6 pueden ser usados como salidas PWM), 6 entradas analógicas, un oscilador de cuarzo a 16MHz, una conexión USB, un conector para alimentación, una cabecera ICSP, un botón de Reset, como se desprende de la Ilustración 1.

Ilustración 1: Placa Arduino Uno



Elaboración: El Autor

La placa Arduino Uno dispone 32KB de memoria (0.5KB son usados para el cargador de inicio —bootloader—).

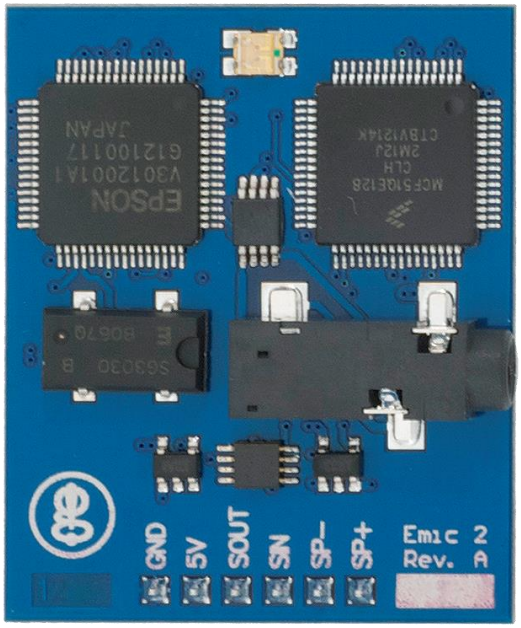
La utilización de la placa Arduino Uno, se prefirió porque su programación se hace mediante el IDE Arduino, lo que facilita cargar nuevo código rápidamente desde la computadora hacia la placa sin la utilización de hardware externo.

2.3.2. Módulo Emic 2 Text-to-Speech

El módulo Emic 2 Text-to-Speech, como se aprecia en la Ilustración 2, es un sintetizador multilenguaje de voz que convierte texto digital en lenguaje natural audible.

Posee una interfaz basada en comandos que facilita integrarlo en cualquier sistema embebido.

Ilustración 2: Módulo Emic 2 Text-to-Speech



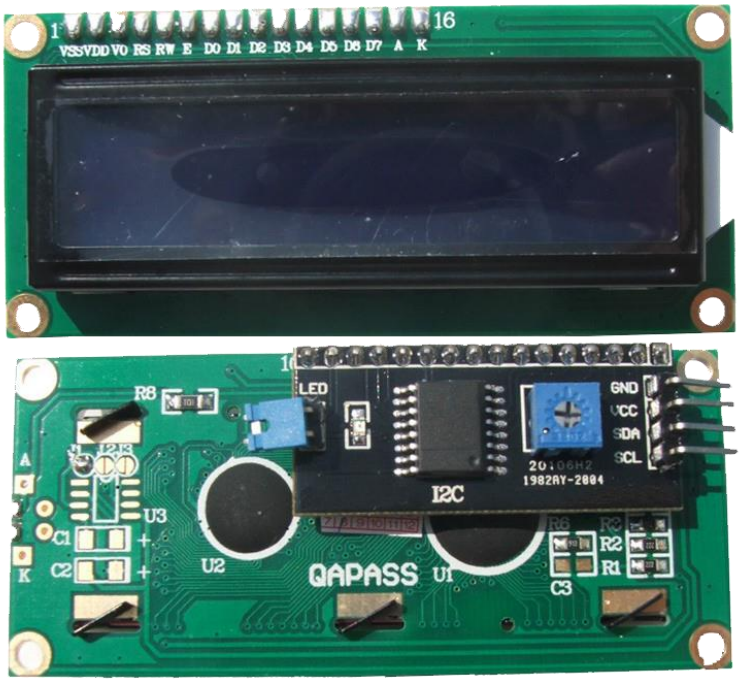
Fuente: (Parallax Inc., 2015)

El módulo Emic 2 Text-to-Speech, se eligió por su alta calidad de habla para el español, por su amplia documentación, y porque usa el motor universalmente reconocido DECTalk Text-to-Speech, lo que garantiza una funcionabilidad y acople excelentes.

2.3.3. Pantalla de Cristal Líquido (LCD)

La pantalla de cristal líquido (LCD acrónimo del inglés Liquid Crystal Display) como se muestra en la Ilustración 3, es una pantalla delgada y plana formada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora.

Ilustración 3: Pantalla de Cristal Líquido (Vista Frontal y Trasera)



Elaboración: El Autor

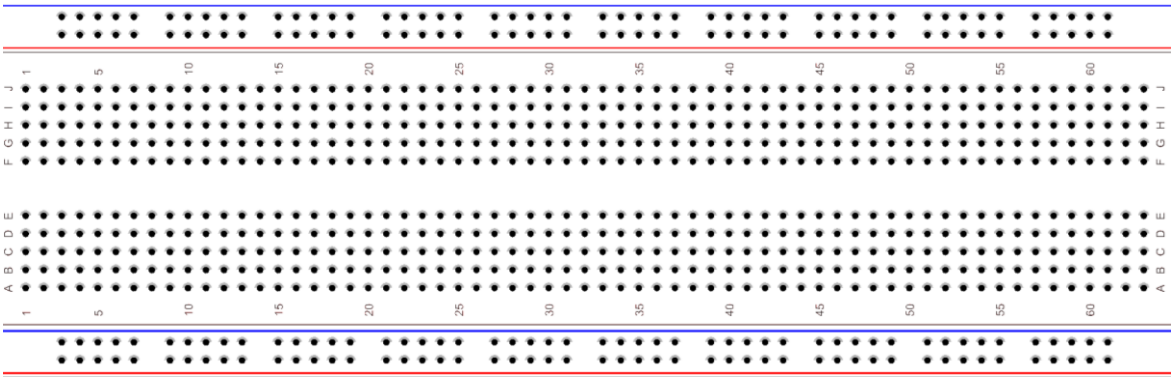
Las pantallas de cristal líquido son una opción de visualización muy sencilla de usar, que integradas a la librería LiquidCrystal_I2C, permiten una fácil visualización de cualquier carácter de texto.

Para el presente proyecto se ha elegido una pantalla de cristal líquido de 16 columnas x2 filas, tamaño que se traduce en 32 caracteres.

2.3.4. Protoboard

La protoboard (placa de ensayo), como se muestra en la Ilustración 4, es un tablero con agujeros preparados en una cuadrícula con las conexiones guías en su interior. La protoboard hace que la conexión entre los dispositivos sea considerablemente simple sin la necesidad de soldadura alguna. Se desmontan con la misma facilidad con la que se ensamblan, de ahí el nombre de placa de ensayo.

Ilustración 4: Protoboard



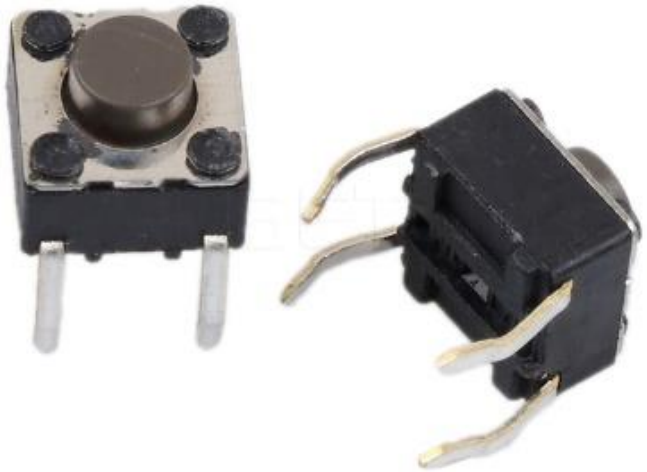
Elaboración: El Autor

El uso de la protoboard en el presente proyecto está restringido a las etapas de diseño y prueba del prototipo, ya que el entregable final cuenta con una placa impresa que la reemplaza.

2.3.5. Pulsadores

Los pulsadores (como se muestra en la Ilustración 5) son interruptores los cuales establecen posiciones de encendido mediante la pulsación de un botón gracias a la presión que se ejerce sobre una lámina conductora interna. Al dejar la pulsación sobre dicho pulsador, un resorte hace recobrar la lámina a su posición inicial, volviendo a la posición de “abierto”.

Ilustración 5: Pulsador



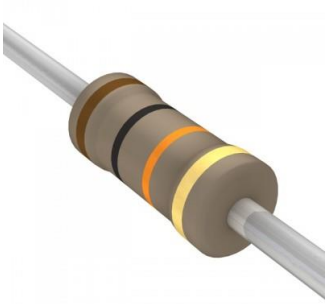
Elaboración: El Autor

En el presente proyecto se hace uso de tres pulsadores para las necesidades que exige el desarrollo de la aplicación.

2.3.6. Resistores

Son componentes electrónicos diseñados para oponerse al paso de una corriente eléctrica produciendo una caída de voltaje entre sus terminales en proporción a la corriente. Los resistores siempre se miden en Ohmios y también puede ser representada por el símbolo Ω.

Ilustración 6: Resistores



Elaboración: El Autor

En el presente proyecto se usan 4 resistores de 220Ω.

2.3.7. Zumbador

Zumbador (del inglés Buzzer), es un transductor electroacústico que produce un sonido o zumbido continuo o intermitente de un mismo tono

Ilustración 7: Zumbador (Buzzer)

Elaboración: El Autor

Se optó por el uso de un Buzzer (Ilustración 7), con el objetivo que emita los sonidos correspondientes a la digitación del Código Morse —tanto para los PUNTOS y RAYAS— mediante el pulsador correspondiente.

2.3.8. Altavoz

Ilustración 8: Altavoz

Elaboración: El Autor

El altavoz, al igual que el Buzzer, es un transductor electroacústico utilizado para la reproducción de sonido. A diferencia que el altavoz, produce una variedad mucho más amplia de sonido.

Para el presente proyecto se utiliza un altavoz de 3.5 pulgadas, como se evidencia en la Ilustración 8, conectado al módulo EMIC2 Text-to-Speech que es el encargado de emitir en forma de audio, el texto ingresado mediante el pulsador a expresión sonora.

Capítulo 3. Metodología y Desarrollo

3.1. Metodología

La metodología puede describirse como a un conjunto de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, usada para estructurar, planear y controlar el proceso de desarrollo de software.

Dentro de las propuestas metodológicas que abarcan las distintas dimensiones del desarrollo de software tememos dos vertientes, por un lado, las *metodologías tradicionale*s que se enfocan en establecer rigurosamente las actividades involucradas, el control de procesos, los artefactos que se deben producir, las herramientas y las notaciones que se emplearán para el desarrollo de software; demostrando ser eficaces e indispensables en determinados proyectos, sin embargo, también presentan graves problemas en otros proyectos.

Por otro lado, la filosofía de las *metodologías ágiles*, se enfoca en otras dimensiones, por ejemplo, el factor humano o el producto de software. Esta filosofía le otorga un mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas, lo que deriva en efectividad cuando se requiere reducir radicalmente los tiempos de desarrollo, pero manteniendo una alta calidad en proyectos con requisitos muy variables.

La revolución de la forma de producir software —provista por las *metodologías ágiles*—, ha generado un amplio debate entre sus seguidores y quienes por escepticismo o convencimiento no las ven como alternativas a las metodologías tradicionales. (Canós, Letelier, & Penadés, 2003)

Las *metodologías ágiles*, establecen soluciones a medida del entorno, abrevian prácticas asegurando la calidad de producto, que diferencia de las *metodologías tradicionales*, son lentas muy controladas y poco flexibles.

Para el presente proyecto, se ha preferido la utilización de las *metodologías ágiles,* expresada como la Metodología de Desarrollo de Hardware Libre, dado que se acopla a la manera de cómo deseamos realizar las cosas, y a las siguientes características:

* Es flexible y se ajusta a las necesidades del desarrollo hardware y de software.

* El equipo de desarrollo puede ser pequeño, medio, grande o comunitario.

* Los requisitos no evidencian un amplio grado de complejidad para su desarrollo.

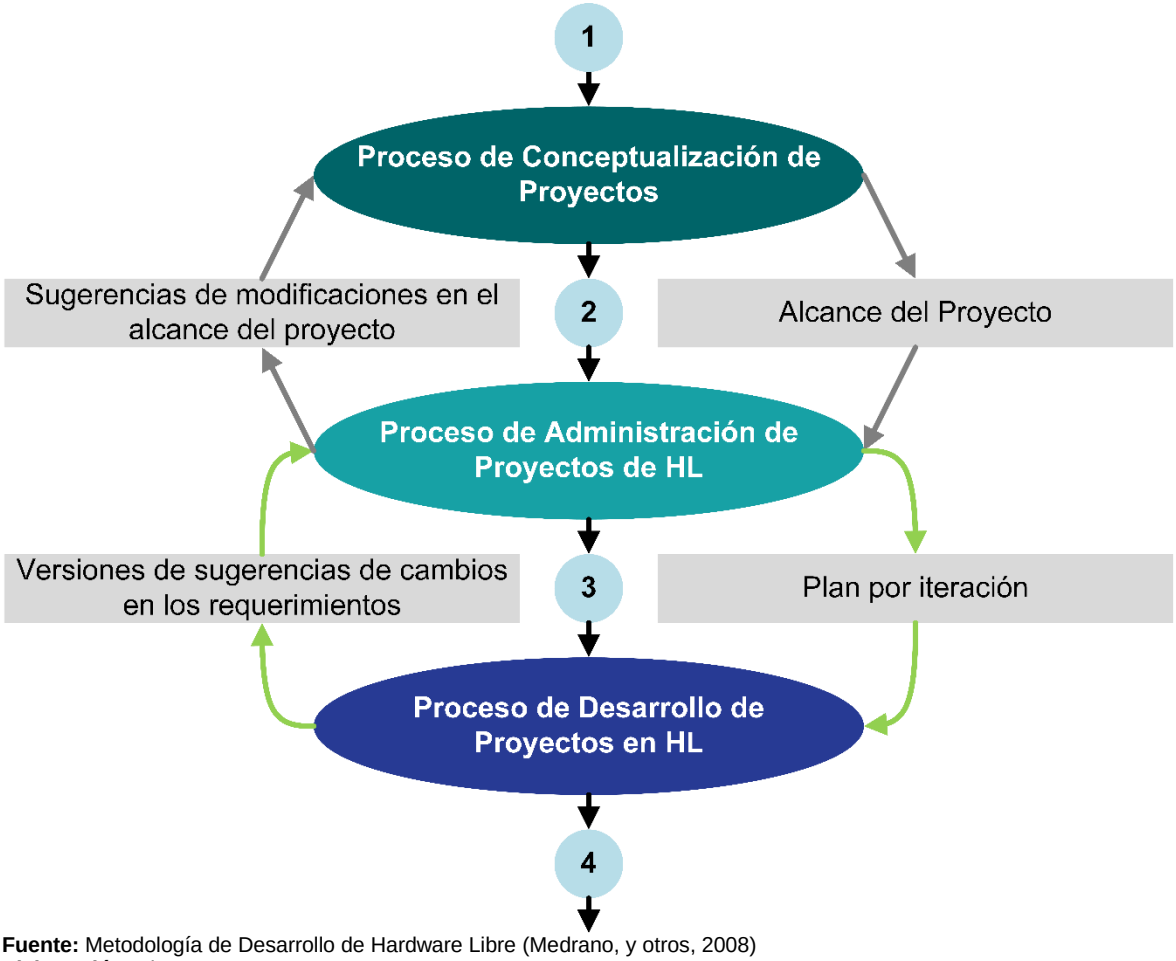
* Facilita un tiempo corto de desarrollo.

La Metodología de Desarrollo de Hardware Libre v1.8, usada en el presente proyecto, fue desarrollada en Venezuela por la Fundación CENDITEL y concedida bajo licencia GFDL 1.2 de la Free Software Foundation, esta licencia es de tipo *copyleft*, lo que significa que los trabajos derivados del documento deben a su vez

ser libres en el mismo sentido. Complementa la Licencia Pública General de GNU, que es una licencia tipo copyleft diseñada para el software libre

Según (Medrano, y otros, 2008), la Metodología de Desarrollo de Hardware Libre posee tres procesos, como se muestra en la Ilustración 9.

Ilustración 9: Plataforma de Desarrollo de Hardware Libre

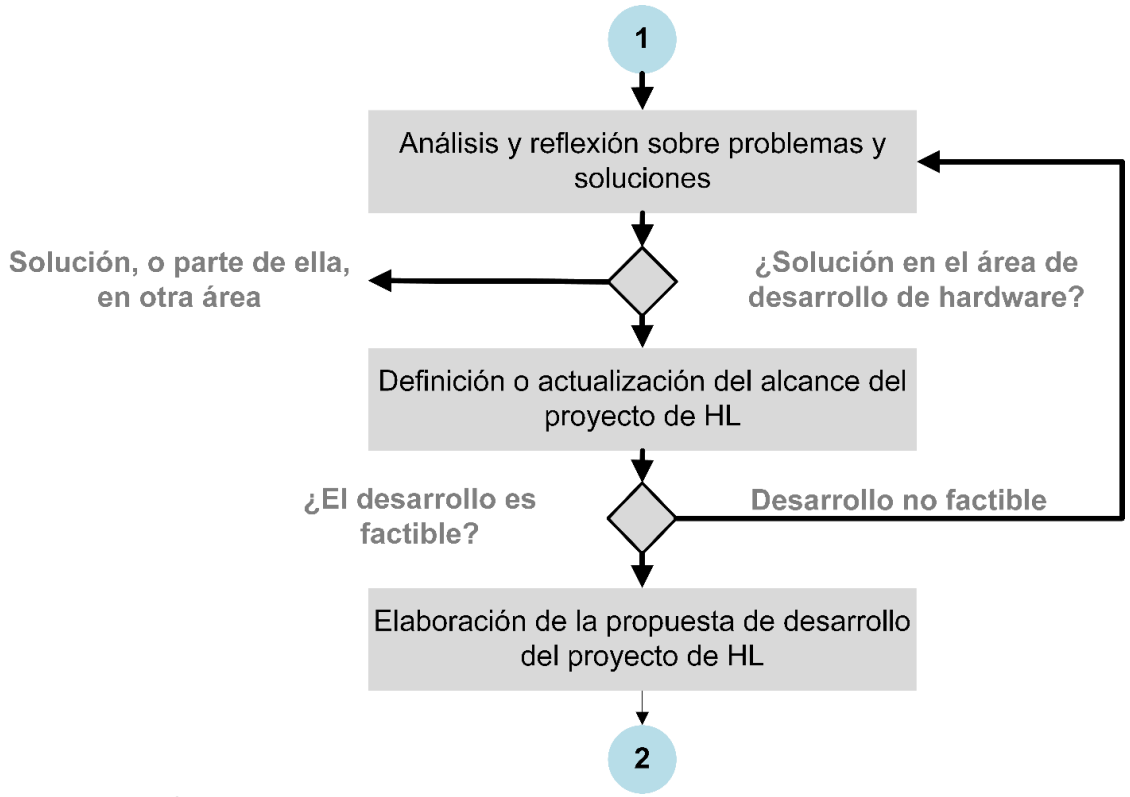


Fuente: Metodología de Desarrollo de Hardware Libre (Medrano, y otros, 2008)
Elaboración: El Autor

3.1.1. Proceso de Conceptualización de Proyectos: Durante este proceso, se estudian necesidades y problemas que puedan demandar de una solución en área de hardware; además, en este proceso, se delimita el alcance que se quiere para el proyecto en desarrollo.

En la Ilustración 10, se muestra la secuencia de ejecución de los pasos necesarios para el Proceso de Conceptualización de Proyectos.

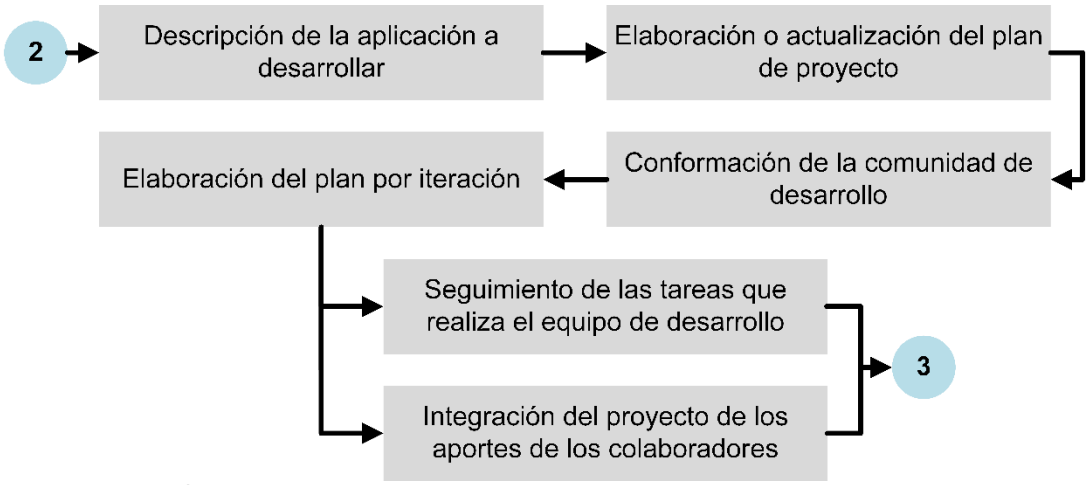
Ilustración 10: Pasos del proceso de conceptualización



Fuente: Metodología de Desarrollo de Hardware Libre (Medrano, y otros, 2008)
Elaboración: El Autor

3.1.2. Proceso de Administración de Proyectos de Hardware Libre: Este proceso encierra las actividades orientadas a la planificación del diseño, regulación y ayuda al orden del proyecto. Estas actividades se orientan hacia la facilitación de lo planteado en el proceso de administración, como se desprende de la Ilustración 11.

Ilustración 11: Actividades del proceso de administración de procesos de desarrollo de hardware libre



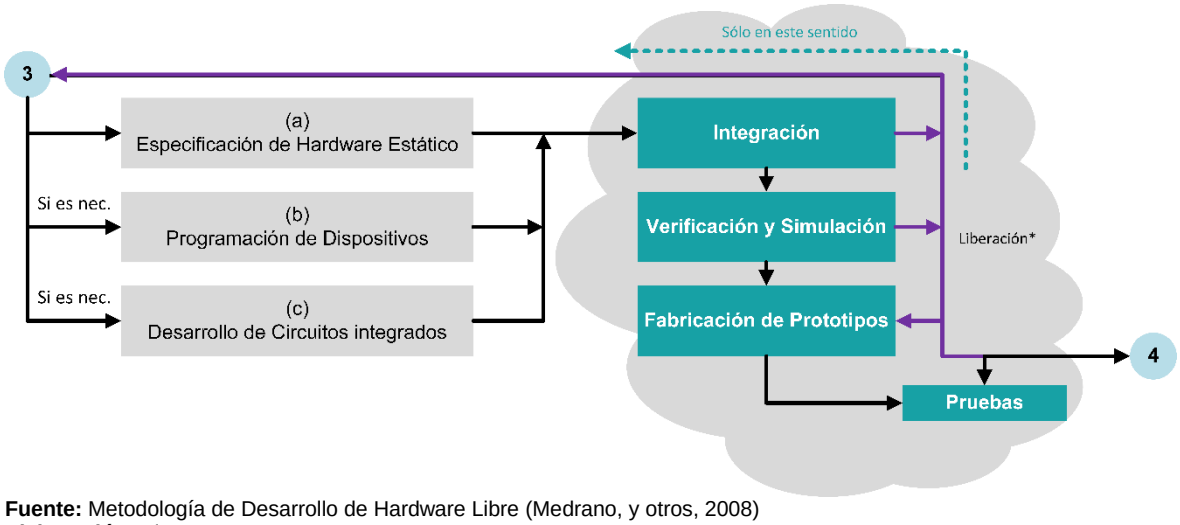
Fuente: Metodología de Desarrollo de Hardware Libre (Medrano, y otros, 2008)
Elaboración: El Autor

3.1.3. Proceso de Desarrollo de Proyectos en Hardware Libre: Tomando como base las descripciones y características elaboradas en los procesos de conceptualización y administración, se especifican los pasos que en principio se deben cumplir —dependiendo la naturaleza del dispositivo— para obtener un diseño completo del hardware.

Como se aprecia en la Ilustración 12, el Proceso de Desarrollo de Proyectos en Hardware Libre, se puede dividir en tres pasos concurrentes: (a) Especificación de hardware estático, (b) Programación de dispositivos y (c) Desarrollo de IC, que son los pasos encargados de generar y depurar los diseños que sean necesarios para implementar las características requeridas; estos pasos pueden activarse o no según los requerimientos del proyecto. Finalizados los pasos (a), (b), y (c), la etapa de integración se encarga de ajustar todos los detalles para obtener un diseño completo del hardware. Esta etapa, permite —si se detecta la necesidad—

reformular el alcance y características del proyecto, posiblemente debido a las incompatibilidades entre diseños o protocolos de comunicación.

Ilustración 12: Proceso de Desarrollo de Proyectos en Hardware Libre



Fuente: Metodología de Desarrollo de Hardware Libre (Medrano, y otros, 2008)
Elaboración: El Autor

La verificación y depuración —ejecutadas mediante simulaciones— del circuito integrado se realiza cuando se obtiene el dispositivo integrado. Los resultados aquí adquiridos, permiten modificaciones en el proceso de integración o en la formulación del alcance y características del proyecto.

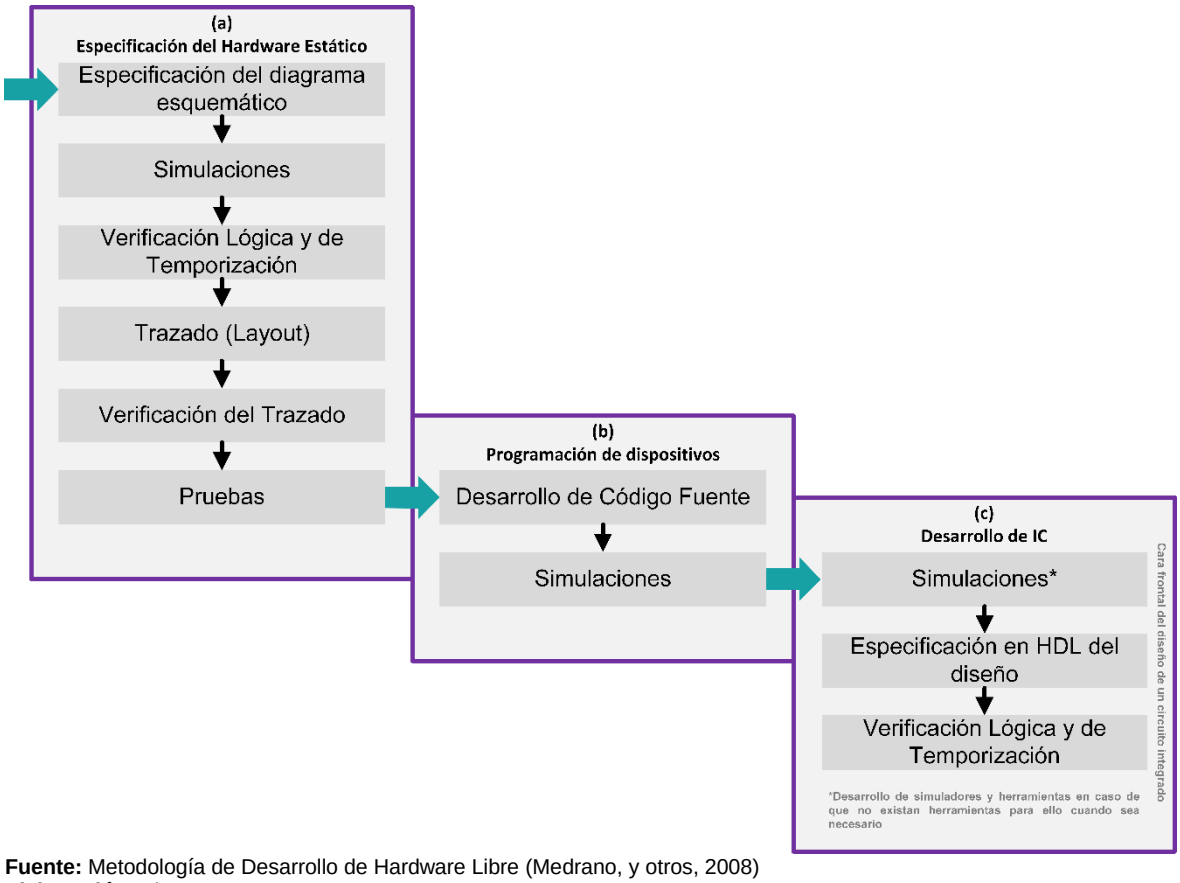
Durante la Fabricación del Prototipo, éste es sometido a diversas pruebas, que pueden revelar la necesidad de modificar dicho prototipo en cualquiera de las etapas anteriores del proyecto, como se muestra en la Ilustración 12 en la línea de color púrpura.

La liberación de los diseños de hardware envuelve a dos formas: *a) La liberación de versiones preliminares o de prueba* (que se pueden obtener en cualquier paso del desarrollo de la Ilustración 12, y, b) *Las versiones estables* que solamente

pueden ser liberadas en cualquiera de los cuatro últimos bloques, a diferencia de las versiones preliminares que pueden ser liberadas en cualquier momento del ciclo de desarrollo.

En la Ilustración 13, se muestran las sub—fases del Proceso de desarrollo de proyectos de Hardware Libre: (a) Especificación del Hardware estático, (b) Programación de dispositivos y (c) Desarrollo de IC

Ilustración 13: Sub—fases del Proceso de Desarrollo de Proyectos de Hardware Libre



Fuente: Metodología de Desarrollo de Hardware Libre (Medrano, y otros, 2008)
Elaboración: El Autor

3.2. Desarrollo aplicando la Metodología de Hardware Libre

El presente proyecto se ejecutó usando la Metodología de Hardware Libre en un plazo de 12 meses, tiempo durante el cual se desarrolló la versión prototipo del dispositivo capaz de traducir Código Morse a expresión sonora; todo el proceso se describe a continuación.

3.2.1. Proceso de Conceptualización de Proyectos

Mediante el Proceso de Conceptualización de Proyectos, se identificaron y analizaron las necesidades y funcionalidades que tiene el dispositivo; además se revisó y actualizó el alcance existente para el del proyecto, y se elaboró la propuesta de desarrollo del proyecto.

En las Tablas 2, 3 y 4 se presentan las sub—fases del Proceso de Conceptualización de Proyectos y su progreso.

Tabla 2: Análisis y reflexión sobre problemas y soluciones

Análisis y reflexión sobre problemas y soluciones					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Identificar problemas y necesidades en el proyecto a desarrollarse.	**Responsables:** El Autor **Participantes:** El Autor	Problemas y requerimientos del dispositivo.	Estimar posibles inconvenientes durante la ejecución del proyecto.	**Técnicas** Observación Investigación Análisis	Propuesta de desarrollo del proyecto
Analizar y reflexionar sobre algunos problemas que se presenten durante el transcurso de desarrollo del proyecto y sus posibles soluciones.	**Responsables:** El Autor **Participantes:** El Autor	Problemas y necesidades identificados	Se estimó la solución más pertinente basándose en la necesidad de mejorar la comunicación para las personas con discapacidad visual-auditiva.	**Técnicas:** Planificación Análisis	

Elaboración: El Autor

Tabla 3: Actualización del alcance de proyecto de Hardware Libre

Actualización del alcance del proyecto de Hardware Libre					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Actualización del alcance del proyecto de investigación: **"Desarrollo de un dispositivo para traducir código Morse a expresión sonora para personas con discapacidad sensorial"**	**Responsables:** El Autor **Participantes:** El Autor	Anteproyecto.	—	**Técnicas** Análisis	Propuesta de desarrollo del proyecto

Elaboración: El Autor

Tabla 4: Elaboración de la propuesta de desarrollo del proyecto

Elaboración de la propuesta de desarrollo del proyecto					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Elaborar la propuesta de desarrollo para el proyecto: **"Desarrollo de un dispositivo para traducir código Morse a expresión sonora para personas con discapacidad sensorial".**	**Responsables:** El Autor **Participantes:** El Autor	Anteproyecto "Alcance del Proyecto"	—	**Técnicas** Análisis Planificación	Propuesta de desarrollo del proyecto

Elaboración: El Autor

3.2.2. Proceso de Administración de Proyectos de Hardware Libre

En el Proceso de Administración de Proyectos de Hardware libre, se coordinó y mantuvo el orden del proyecto, permitiendo realizar la descripción del dispositivo a desarrollar (Tabla 5); se actualizó el Plan del Proyecto (Tabla 6); se estableció la comunidad de desarrollo (Tabla 7); se elaboró el Plan por Iteración (Tabla 8) que generó el seguimiento de las tareas que realizó el equipo de desarrollo (Tabla 9) y finalmente se integraron los aportes del equipo de desarrollo al proyecto. (Tabla 10)

Tabla 5: Descripción del dispositivo a desarrollar

Descripción del dispositivo a desarrollar					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Descripción del dispositivo a desarrollar en función del alcance establecido en para el proyecto	**Responsables:** El Autor **Participantes:** El Autor	Alcance del proyecto.	Proporcionar una visión en términos de requisitos y funcionalidades del dispositivo.	**Técnicas** Análisis Requerimientos	Ficha Técnica (Anexo A)

Elaboración: El Autor

Tabla 6: Actualización del plan del proyecto

Actualización del plan del proyecto					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Especificación y priorización de las funcionalidades del dispositivo, basándose en los procesos de entrada/salida de datos definidos en la Ficha Técnica.	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica (Anexo A)	—	**Técnicas** Análisis Requerimientos	Priorización de funcionalidades (Anexo B)
Levantamiento del Cronograma de Actividades	**Responsables:** El Autor **Participantes:** El Autor	—	—	**Plantillas** Cronograma	Cronograma de actividades (Anexo C)

Elaboración: El Autor

Tabla 7: Conformación de la comunidad de desarrollo

Conformación de la comunidad de desarrollo					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Definir comunidad (equipo) de trabajo-desarrollo	**Responsables:** El Autor **Participantes:** El Autor	—	El autor se encargará de liderar, dirigir y ejecutar todas las tareas planteadas para el desarrollo del presente proyecto.	—	—

Elaboración: El Autor

Tabla 8: Elaboración del Plan por Iteración

Elaboración del Plan por Iteración					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Elaboración del Plan por Iteración: * Iteración 1, * Iteración 2, * Iteración 3, y, * Iteración 4	**Responsables:** El Autor **Participantes:** El Autor	Proceso de desarrollo de Hardware Libre	El Plan de Iteración propone las tareas que se realizarán durante la construcción de las funcionalidades pertinentes a cada iteración indicando la duración de cada tarea en horas.	**Técnicas** Diagramas Otros	Plan por Iteración (Anexo D)

Elaboración: El Autor

Tabla 9: Seguimiento de las tareas que realiza el equipo de desarrollo

Seguimiento de las tareas que realiza el equipo de desarrollo					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Ejecutar el seguimiento de la tareas para el Plan de Iteración	**Responsables:** El Autor **Participantes:** El Autor	Plan de Iteración	Esta actividad realiza el seguimiento de las iteraciones descritas en el Plan de Iteración	**Técnicas:** Comprobación	Seguimiento de tareas (Anexo E)
Validar los resultados obtenidos para cada tarea presente en el Seguimiento de Tareas	**Responsables:** El Autor **Participantes:** El Autor	Seguimiento de Tareas	La validación asegura la calidad del producto obtenido.	—	Validación de resultados por tarea (Anexo F)

Elaboración: El Autor

Tabla 10: Integración del proyecto de los aportes de los colaboradores

Integración del proyecto de los aportes de los colaboradores					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Analizar el código fuente, archivos, y documentos para incluirlos en el proyecto.	**Responsables:** El Autor **Participantes:** El Autor	Código fuente Documentación	Esta actividad integra el código fuente, archivos y documentación al proyecto como tal.	—	—

Elaboración: El Autor

3.2.3. Proceso de Desarrollo de Proyectos en Hardware Libre

El Proceso de Desarrollo de Proyectos en Hardware Libre, contempla 8 sub—fases agrupadas en dos grupos, el primer grupo sigue los lineamientos descritos en el Plan de Iteración y agrupa las fases:(a) Especificación de Hardware estático, (b) Programación de Dispositivos, y (c) Desarrollo de IC descritas en las Tablas 11, 12 y 13 respectivamente. El grupo restante contiene las fases: Integración, Verificación y Simulación, Fabricación del dispositivo, Pruebas, y Liberación detalladas en las Tablas 14, 15, 16, 17 y 18 respectivamente.

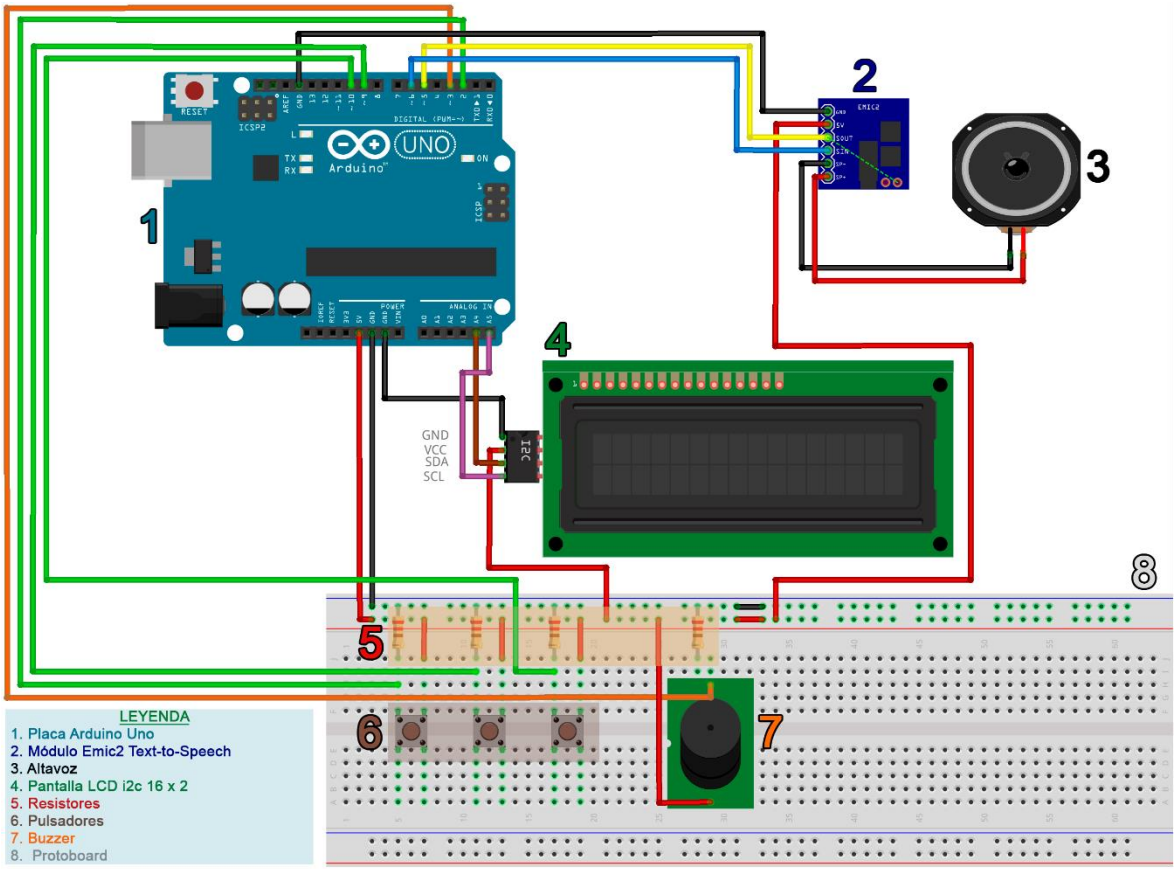
3.2.3.1. Especificación de Hardware Estático

En la Especificación de Hardware Estático (Tabla 11), durante la Especificación del diagrama esquemático —tomando como base la Ficha Técnica—, se determinó que componentes de hardware serían necesarios para el desarrollo del dispositivo: una placa Arduino Uno, una Pantalla LCD i2c de 16x2 caracteres, un Módulo Emic2 Text-to-Speech, un altavoz de 0.5W, tres pulsadores, cuatro resistores de 220 ohm un Buzzer, y un Protoboard.

Utilizando la herramienta libre Fritzing se realizó el Esquema del Circuito como se muestra en la Ilustración 14.

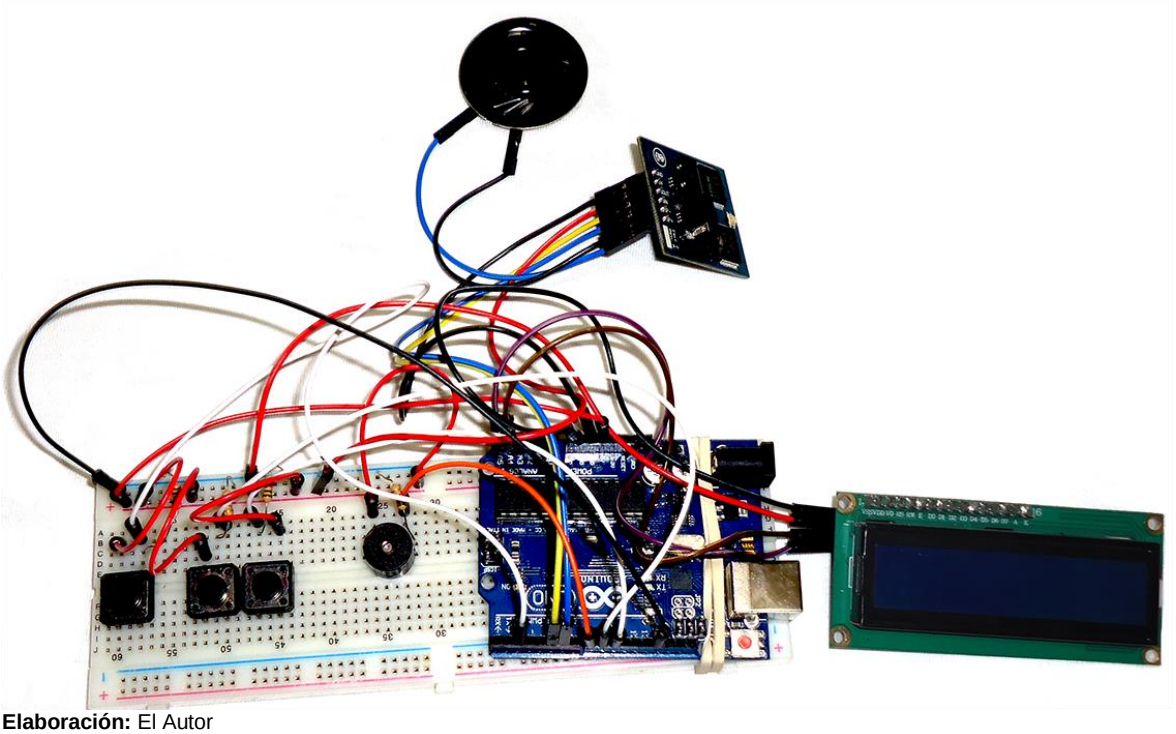
Luego se realizó el cableado físico del Prototipo del circuito como se muestra en la ilustración 15.

Ilustración 14: Esquema del Circuito



Elaboración: El Autor

Ilustración 15: Prototipo del Circuito

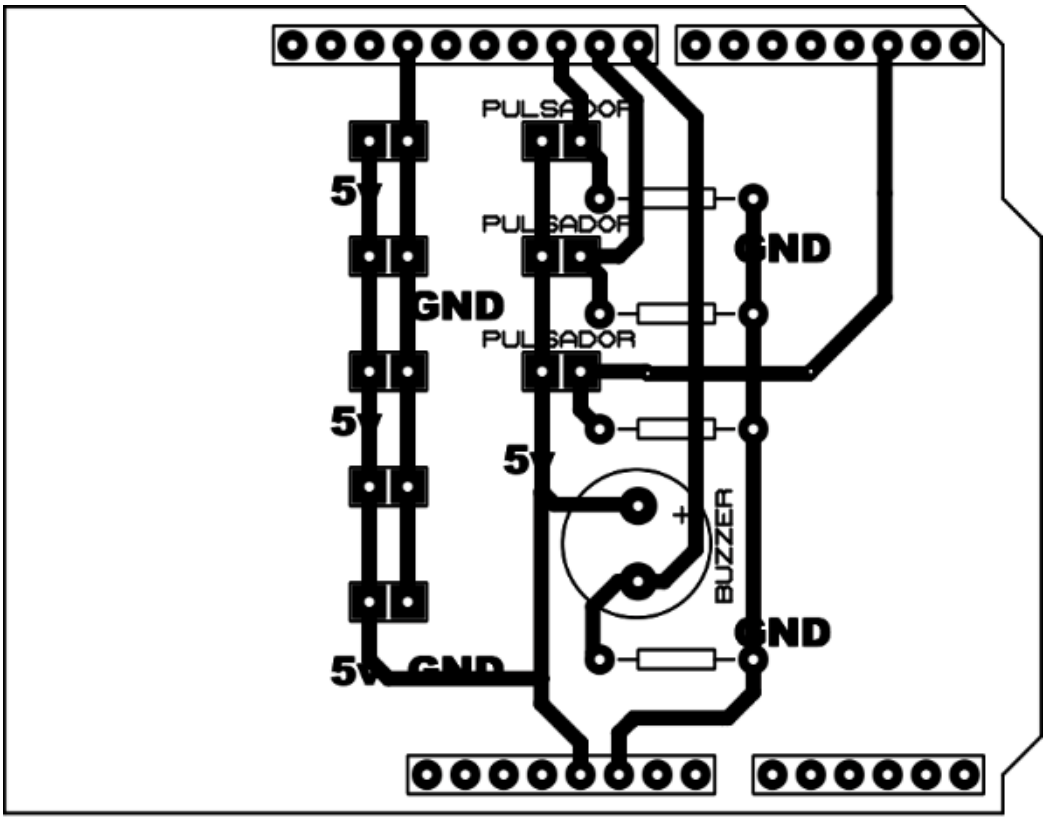


Elaboración: El Autor

La sub—fase Simulación, aplicada al prototipo del dispositivo, facilitó obtener la Validación de resultados por tareas (Anexo F) y la especificación de casos de prueba (Anexo G).

Durante la sub—fase Trazado (layout), ejecutada durante la Iteración 4, utilizando la herramienta libre Inkscape, se realizó el diseño para el PCB, como se muestra en la Ilustración 16.

Ilustración 16: Diseño PCB



Elaboración: El Autor

La sub—fase Verificación del Trazado, ejecutada durante la Iteración 4, permitió comprobar el Circuito Impreso PCB, y se agregaron los resultados a la Validación de resultados por tarea (Anexo F).

La sub—fase Pruebas, se ejecutaron durante todas las iteraciones, y sus resultados permitieron depurar errores que se agregaron tanto a la Validación de resultados por tareas (Anexo F), como a la Especificación de Casos de Prueba (Anexo G).

Tabla 11: Especificación de Hardware Estático (a)

Especificación de hardware Estático (a)					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Especificación del diagrama esquemático	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica	Esta actividad se ejecuta en la Iteración: (✓) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | ( ) Iteración 4	**Técnicas** Diseño Esquemático **Herramientas** Fritzing	Esquema del Circuito (Ilustración 14) Prototipo del circuito (Ilustración 15)
Simulación	**Responsables:** El Autor **Participantes:** El Autor	Prototipo del Dispositivo	Esta actividad se ejecuta en la Iteración: (✓) Iteración 1 | (✓) Iteración 2 (✓) Iteración 3 | (✓) Iteración 4	**Herramientas** Arduino IDE	Validación de resultados por tarea (Anexo F) Especificación de Casos de Prueba (Anexo G)
Verificación Lógica y de Temporización.	**Responsables:** El Autor **Participantes:** El Autor	Circuito Esquemático	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | (✓) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	**Técnica:** Diseño PCB **Herramientas** Arduino IDE	Validación de resultados por tarea (Anexo F) Especificación de Casos de Prueba (Anexo G)

Trazado (layout)	**Responsables:** El Autor **Participantes:** El Autor	Diseño Componente	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	**Técnica:** Autoruteo **Herramientas** Fritzing, Inkscape	Diseño PCB (Ilustración 16)
Verificación de trazado	**Responsables:** El Autor **Participantes:** El Autor	Circuito impreso PCB	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	**Técnica** Comprobación de pistas	Validación de resultados por tarea (Anexo F)
Pruebas	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica Prototipo del Dispositivo Circuito Esquemático Circuito Impreso PCB	Esta actividad se ejecuta en la Iteración: (✓) Iteración 1 | (✓) Iteración 2 (✓) Iteración 3 | (✓) Iteración 4	**Técnica:** Verificación **Herramientas** Arduino IDE	Validación de resultados por tarea (Anexo F) Especificación de Casos de Prueba (Anexo G)

Elaboración: El Autor

3.2.3.2. Programación de Dispositivos

Durante la fase Programación de Dispositivos, descrita en la Tabla 12 —tomando como base la información obtenida en los procesos anteriores—, se desarrolló el Código Fuente (Anexo I) usando el IDE de Arduino y el lenguaje de programación C++.

Se agregaron las librerías: a) LiquidCrystal_I2C.h para el funcionamiento de la pantalla LCD;y, b) SoftwareSerial.h para la comunicación con el módulo EMIC2 Text-to-Speech

La Priorización de Funcionalidades (Anexo B) determinó la prioridad de codificación de los métodos disponibles para el dispositivo.

Esta sub—fase, determinó la funcionalidad para los tres pulsadores, la pantalla LCD, y el módulo EMIC2 Text-to-Speech. Esta sub-fase se ejecutó durante las Iteraciones 2, 3, y 4.

La sub-fase Simulación, aplicada al prototipo del dispositivo, facilitó obtener la Validación de resultados por tareas (Anexo F) y la Especificación de Casos de Prueba (Anexo G).

Tabla 12: Programación de Dispositivos (b)

Programación de dispositivos (b)					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Desarrollar código fuente	**Responsables:** El Autor **Participantes:** El Autor	Algoritmo de funcionamiento	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | (✓) Iteración 2 (✓) Iteración 3 | (✓) Iteración 4	**Técnicas** Programación **Herramientas** Arduino IDE	Código fuente (Anexo I)
Simulación	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica Código Fuente Circuito Esquemático Circuito Impreso PCB	Esta actividad se ejecuta en la Iteración: (✓) Iteración 1 | (✓) Iteración 2 (✓) Iteración 3 | (✓) Iteración 4	**Técnicas** Pruebas de software **Herramientas** Arduino IDE	Validación de resultados por tarea (Anexo F) Especificación de Casos de Prueba (Anexo G)

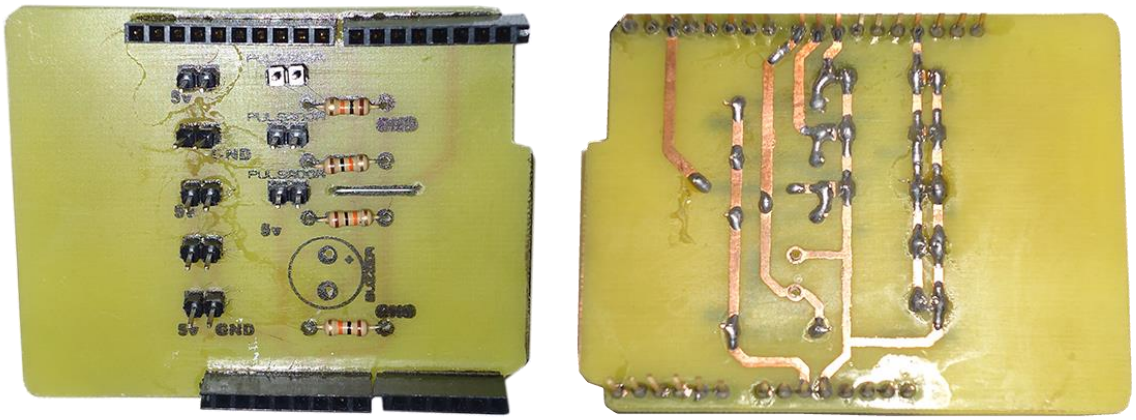
Elaboración: El Autor

3.2.3.3. Desarrollo de Circuitos Integrados

En el Desarrollo de Circuitos Integrados, como se muestra en la Tabla 13, se ejecutó la Impresión del Diseño del PCB expuesto en la Ilustración 16.

El circuito Impreso PCB se presenta en la Ilustración 17.

Ilustración 17: Circuito Impreso PCB



Elaboración: El Autor

Tabla 13: Desarrollo de Circuitos Integrados (c)

Desarrollo de Circuitos Integrados (c)					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Impresión del Diseño del PCB	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica Código Fuente Circuito Impreso PCB	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	**Técnicas** Diseño	Circuito Impreso PCB (Ilustración 17)

Elaboración: El Autor

3.2.3.4. Integración

La Integración, descrita en la Tabla 14, acopla los procesos anteriores: Especificación de Hardware Estático (a), Programación de Dispositivos (b) y Desarrollo de IC (c), es decir, se integra tanto hardware, así como también, se cargar el código fuente al prototipo.

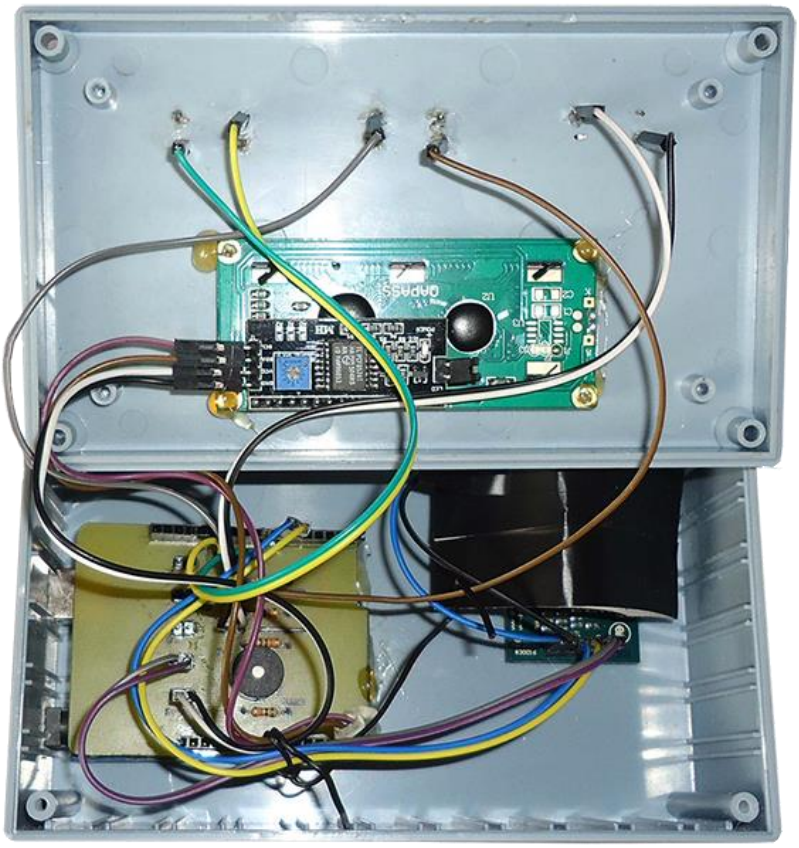
3.2.3.5. Verificación y Simulación

La fase de Verificación y Simulación —como se muestra en la Tabla 15—ejecutada posterior a la fase de integración, faculta la comprobación del cumplimiento de los requisitos y el control de las funcionalidades. Los resultados se anexan a la Validación de resultados por tarea (Anexo F) y a la Especificación de Casos de Prueba (Anexo G).

3.2.3.6. Fabricación del dispositivo

Durante esta fase se realizó la adecuación del chasis y se ensamblaron los componentes electrónicos dentro del mismo, como se especifica en la Tabla 16, y se expone en la ilustración18.

Ilustración 18: Fabricación del dispositivo



Elaboración: El Autor

Ilustración 19: Dispositivo apto para pruebas

Elaboración: El Autor

3.2.3.7. Pruebas

Durante esta fase, se hicieron las pruebas pertinentes y se gestionó la detección de posibles fallas del dispositivo, como se desprende de la Tabla 17.

3.2.3.8. Liberación

La fase de Liberación, permitió:

a) Corregir los errores encontrados en la versión de prueba, y,

b) Realizar la liberación de la versión estable del dispositivo, como se expone en la Tabla 18, y se muestra en la Ilustración 20.

Ilustración 20: Dispositivo Final

Elaboración: El Autor

Tabla 14: Integración

Integración					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Integración de lo módulos desarrollados en: (A) Especificación de hardware estático, (B) Programación de dispositivos, (C) Desarrollo de IC	**Responsables:** El Autor **Participantes:** El Autor	Propuesta de desarrollo del proyecto	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	—	Cargar código fuente al prototipo

Elaboración: El Autor

Tabla 15: Verificación y Simulación

Verificación y Simulación					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Verificación del diseño luego de la integración.	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica Código Fuente Circuito Impreso PCB	Esta actividad se ejecuta en la Iteración: (✓) Iteración 1 | (✓) Iteración 2 (✓) Iteración 3 | (✓) Iteración 4	**Herramientas** Protocolo de pruebas	Validación de resultados por tarea (Anexo F) Especificación de Casos de Prueba (Anexo G)
Simulación del dispositivo.	**Responsables:** El Autor **Participantes:** El Autor	Ficha Técnica Código Fuente Circuito Impreso PCB	Esta actividad se ejecuta en la Iteración: (✓) Iteración 1 | (✓) Iteración 2 (✓) Iteración 3 | (✓) Iteración 4	**Técnicas** Pruebas de software **Herramientas** Arduino IDE	Validación de resultados por tarea (Anexo F) Especificación de Casos de Prueba (Anexo G)

Elaboración: El Autor

Tabla 16: Fabricación del Dispositivo

Fabricación del dispositivo					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Adecuación del chasis para el dispositivo.	**Responsables:** El Autor **Participantes:** El Autor	Circuito Impreso PCB	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	—	Fabricación del dispositivo (Ilustración 18)
Ensamblaje de componentes electrónicos.	**Responsables:** El Autor **Participantes:** El Autor	—	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	—	Dispositivo apto para las pruebas finales (Ilustración 19)

Elaboración: El Autor

Tabla 17: Pruebas

Pruebas					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Ejecutar el protocolo de pruebas	**Responsables:** El Autor **Participantes:** El Autor	Dispositivo apto para las pruebas finales (Ilustración 20)	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	—	Resumen de Pruebas
Detección de posibles fallas o deficiencias del dispositivo.	**Responsables:** El Autor **Participantes:** El Autor	Dispositivo apto para las pruebas finales (Ilustración 20)	Esta actividad se ejecuta en la Iteración: ( ) Iteración 1 | ( ) Iteración 2 ( ) Iteración 3 | (✓) Iteración 4	—	Resumen de Pruebas

Elaboración: El Autor

Tabla 18: Liberación

Liberación					
Actividad	**Responsables Participantes**	**Insumo**	**Observaciones**	**Técnicas Herramientas Plantillas**	**Producto**
Corrección de errores encontrados en las versiones pruebas.	**Responsables:** El Autor **Participantes:** El Autor	—	—	**Técnica** Depuración	—
Liberación de versión estable.	**Responsables:** El Autor **Participantes:** El Autor	—	—	—	Dispositivo Final (Ilustración 20)

Elaboración: El Autor

Capítulo 4. Resumen de Pruebas y Liberación del Dispositivo

4.1. Resumen de pruebas del dispositivo

Para determinar el correcto funcionamiento del dispositivo, se ejecutaron dos tipos de prueba:

a) Pruebas de funcionamiento del dispositivo

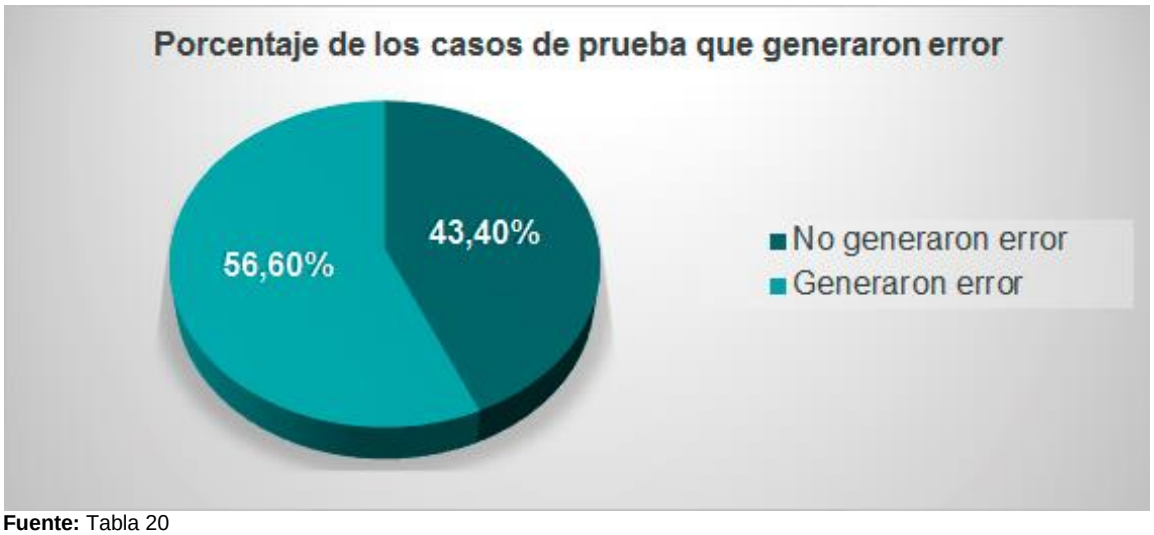
Mediante las pruebas de funcionamiento del dispositivo se comprobaron tanto posibles errores en la conectividad del hardware, como errores que puedan provenir de la ejecución lógica del código fuente de la aplicación. En la Tabla 19, se exponen los casos de prueba que generaron errores.

Tabla 19: Porcentaje de los casos de prueba que generaron error

Porcentaje de los casos de prueba que generaron error		
Casos de Prueba	**Errores**	**%**
No generaron error	23	43,40%
Generaron error	30	56,60%
Total	**53**	**100,00%**

Fuente: Anexo G
Elaboración: El Autor

Ilustración 21: Porcentaje de los casos de prueba que generaron error



Fuente: Tabla 20
Elaboración: El Autor

Como deviene de la Ilustración 21, el 56,60% de los casos de prueba generaron error debido a que Arduino no dispone de métodos que permitan hacer un conteo exacto del tiempo durante el cual un pulsador se mantiene presionado; así como también las excepciones sin controlar.

El código fuente se rectificó, reparando todos los casos de prueba que generaron error.

En la Tabla 20, se despliegan los métodos que presentaron errores en la ejecución de los casos de prueba del dispositivo.

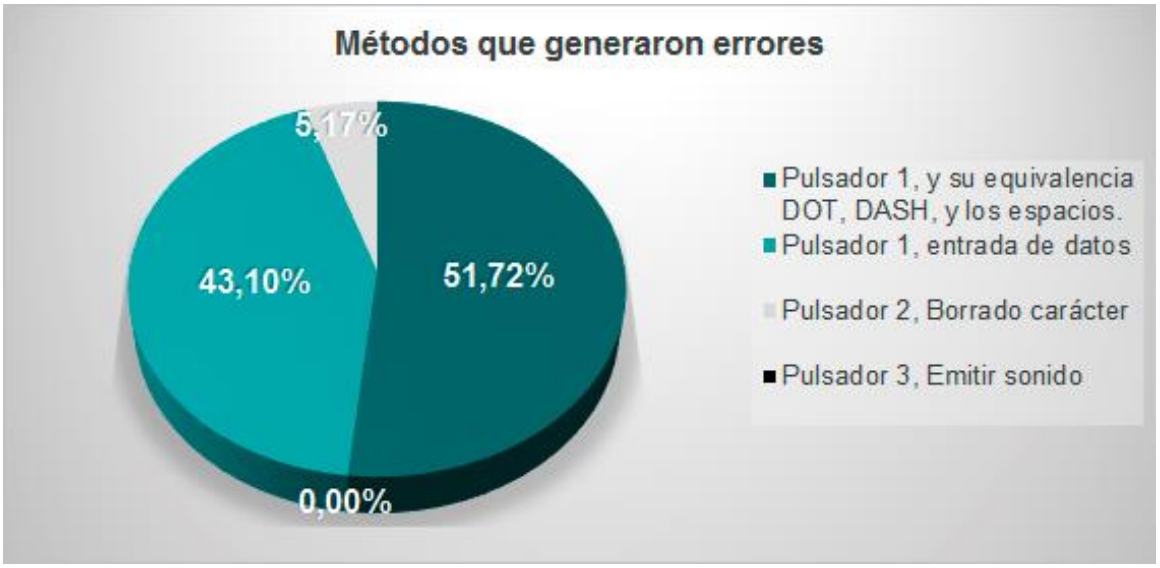
Tabla 20: Métodos que generaron errores

Métodos que generaron errores		
Funcionalidad	**Errores**	**%**
Pulsador 1, y su equivalencia DOT, DASH, y los espacios.	30	51,72%
Pulsador 1, entrada de datos	25	43,10%
Pulsador 2, Borrado carácter	3	5,17%
Pulsador 3, Emitir sonido	0	0,00%
Total	**58**	**100,00%**

Fuente: Anexo G
Elaboración: El Autor

En la Ilustración 22, se muestra los métodos que generaron errores durante la ejecución de los casos de prueba.

Ilustración 22: Métodos que generaron errores



Fuente: Tabla 20
Elaboración: El Autor

El elevado grado de métodos que generaron errores, presentados en la Ilustración 22, se debieron al alto grado de complejidad al desarrollar el código

fuente para interpretar las pulsaciones generadas por el usuario y convertirlas a texto.

Estas pruebas permitieron localizar los errores en las funcionalidades y realizar las acciones correctivas para el correcto funcionamiento del dispositivo.

b) Pruebas de Integridad de Datos

La integridad de datos se comprobó utilizando la herramienta “Monitor Serial” de Arduino, que permite visualizar a manera de depuración las pulsaciones, caracteres y espacios digitados por el usuario mediante el Pulsador1. Además de visualizar los datos, también se pueden comprobar mediante la audición del texto presionando el Pulsador3.

Estas pruebas permitieron comprobar la correcta interpretación que la aplicación realiza para el Código Morse.

4.2. Liberación

La liberación del dispositivo se realizó el día viernes, 03 de marzo del 2017, en su versión estable y final 1.0.

Capítulo 5. Arquitectura Funcional

5.1. Arquitectura funcional

Para determinar la arquitectura funcional de la aplicación, es conveniente recurrir— en primera instancia— al algoritmo que permitirá descifrar el Código Morse y emitir en formato de audio la expresión correspondiente al Código Morse ingresado.

El algoritmo contiene de 10 métodos:

1) Método loop,

2) Método pulsadorON,

3) Método pulsadorOFF,

4) Método shiftBits,

5) Método printCaracter,

6) Método printEspacio,

7) Método printPuntuacion,

8) Método speak,

9) Método borrar_caracter, y,

10) Método beep.

El método principal es el método **loop**; este método se ejecuta indefinidamente mientras el dispositivo esté alimentado con energía eléctrica. Desde este método se hace el llamado a los métodos que se requieren para la funcionalidad del dispositivo.

Los métodos **pulsadorON** y **pulsadorOFF,** corresponden al pulsador 1 (Morse).

El método **pulsadorON**, se ejecuta siempre que el pulsador 1 (Morse) esté presionado, y dispone de las siguientes variables:

a) tPresionado: Almacena el número de veces que el método loop se repite mientras el pulsador esté presionado, con la finalidad de determinar el valor de un dit y un dah.

b) Espera: Variable que permitirá, al método pulsadorOFF, reconocer cuando el usuario termina de escribir un carácter.

c) caracterListo, Variable booleana que determina cuándo el carácter que le usuario está digitando está listo

d) ditODah, Variable booleana usada para determinar cuál de los dos valores (dit o dah) digitó el usuario.

El método **pulsadorOFF,** se ejecuta siempre y cuando el pulsador 1 (Morse) no esté presionado. Mediante las acciones realizadas en este método, se establece:

a) Si el usuario está escribiendo una nueva palabra o no. Si se determina que es una nueva palabra, se ejecuta el método **printEspacio,** que agregará un espacio tanto en la visualización de la pantalla LCD, como en la cadena FraseEMIC.

b) Se hace la llamada al método **shiftBits**.

c) Se determina si usuario ha terminado de digitar la combinación respectiva de Código Morse para el carácter deseado, de ser así, se ejecuta el método **printCaracter,** que es el encargado de presentar en pantalla el carácter correspondiente a la combinación de Código Morse digitada, además almacena dicho carácter en la variable FraseEMIC**.**

Desde el método **printCaracter** —si se considera oportuno— se ejecutará el método alternativo **printPuntuacion** que se usa para presentar/almacenar caracteres de puntuación especiales '**:**', '**,**', '**!**', '**-**', '@', '**.**', '**?**'.

La lógica de programación del algoritmo obedece a la ejecución del método **shiftBits,** método en el cual se determina, en base al valor de tPresionado, si el valor ingresado es un dit o un dah. Si el valor es un dit, se considera como 1. SI el valor ingresado es un dah, se considera como 0.

Tomando esto como referencia, se procede a hacer el desplazamiento de bits con la instrucción

Num = Num << 1

Cada vez que el método **shiftBits** se ejecuta, se desplaza un bit hacia la izquierda en la variable Num —con el propósito de producir las combinaciones expuestas en la Tabla 21—, que luego el método **printCaracter** usa para localizar el carácter deseado en el arreglo Caracteres[Num].

El método **borrar_caracter,** efectúa las acciones necesarias para que, cada vez que presionemos el pulsador 2 (Retroceso), borre el último carácter de la cadena FraseEmic.

El método **speak,** realiza las acciones necesarias para convertir el texto mostrado en la pantalla LCD a lenguaje audible usando el módulo Emic2 Text-to-Speech. Si el usuario aún no ha escrito nada, se emite el mensaje audible: “Aún no ha escrito nada, Digite algo para traducirlo para Usted. Por favor”.

El método **beep,** implementa las acciones necesarias para encender y apagar el Buzzer, en otras palabras, gracias a este método, el Buzzer emite/cancela una señal acústica cada vez que presionamos/soltamos el pulsador uno.

Tabla 21: Representación Binaria/Decimal para el arreglo Caracteres[ ]

Representación Binaria/Decimal para el arreglo Caracteres[ ]					
Valor Binario	Valor Decimal	Repr. Carácter	Valor Binario	Valor Decimal	Repr. Carácter
00000000	0	#	00100000	32	0
00000001	1	#	00100001	33	9
00000010	2	T	00100010	34	#
00000011	3	E	00100011	35	8
00000100	4	M	00100100	36	#
00000101	5	N	00100101	37	#
00000110	6	A	00100110	38	#
00000111	7	I	00100111	39	7
00001000	8	O	00101000	40	#
00001001	9	G	00101001	41	#
00001010	10	K	00101010	42	#
00001011	11	D	00101011	43	#
00001100	12	W	00101100	44	#
00001101	13	R	00101101	45	#
00001110	14	U	00101110	46	#
00001111	15	S	00101111	47	6
00010000	16	#	00110000	48	1
00010001	17	#	00110001	49	#
00010010	18	Q	00110010	50	#
00010011	19	Z	00110011	51	#
00010100	20	Y	00110100	52	#
00010101	21	C	00110101	53	#
00010110	22	X	00110110	54	#
00010111	23	B	00110111	55	#
00011000	24	J	00111000	56	2
00011001	25	P	00111001	57	#
00011010	26	#	00111010	58	#
00011011	27	L	00111011	59	#
00011100	28	#	00111100	60	3
00011101	29	F	00111101	61	#
00011110	30	V	00111110	62	4
00011111	31	H	00111111	63	5

Elaboración: El Autor

5.2. Algoritmo de la aplicación

Definición de Constantes

LCDMAXCOL = 16
LCDMAXFIL = 2

Definición de Variables

entero pulsador_morse = 0
entero pulsador_retroceso = 0
entero pulsador_hablar = 0

entero dit = 100
entero tiempopromedioDah = 150
entero tRebote = 2
entero tPresionado = 0
entero Num

booleano ditODah = verdadero
booleano caracterListo = verdadero

entero largo EsperaCompleta = 10000
entero largo Espera = EsperaCompleta
entero largo nuevaPalabra = 0

caracteres Caracteres[] =
"##TEMNAIOGKDWRUS##QZYCXBJP#L#FVH09#8###7#######61#######2###3#45"

Cadena FraseEMIC

MÉTODO loop

1. inicio
2. Leer pulsador_morse, pulsador_retroceso, pulsador_hablar
3. Si (pulsador_morse = verdadero) Entonces
    3.1. Ejecutar método pulsadorON
4. Si (pulsador_morse = false) Entonces
    4.1. Ejecutar método pulsadorOFF
5. Si (pulsador_retroceso = verdadero) Entonces
    5.1. Ejecutar método borrar_caracter(1000)
6. Si (pulsador_hablar) Entonces
    6.1. Ejecutar método speak (2000);
7. Fin

MÉTODO pulsadorON

1. Inicio
2. Ejecutar método beep (verdadero)
3. Espera = EsperaCompleta
4. tPresionado = tPresionado + 1
5. SI (Num = 0) Entonces
        5.1. Num = 1
    Fin Si
6. caracterListo = falso
7. ditODah = falso;
8. Pausa (tRebote)
9. Fin

MÉTODO pulsadorOFF

1. Inicio
2. Ejecutar método beep (falso)
3. Si (nuevaPalabra > 0) Entonces
    3.1. nuevaPalabra = nuevaPalabra – 1
   Fin Si
4. Si (nuevaPalabra == 1) Entonces
    4.1. Ejecutar método printEspacio
   Fin Si
5. Si (!ditODah) Entonces
    5.1. Ejecutar método shiftBits
   Fin Si
6. Si (!caracterListo) Entonces
    6.1. Espera = Espera - 1
    6.2. tPresionado = 0
    6.3. Si (Espera == 0) Entonces
        6.3.1. Espera = EsperaCompleta
        6.3.2. Ejecutar método printCaracter
        6.3.3. caracterListo = verdadero
        63.4. Num = 0
      Fin Si
   Fin Si
6. Fin

MÉTODO shiftBits

1. Inicio
2. ditODah = verdadero
3. Si (tPresionado < dit/3) Entonces
    3.1. Retornar
   Fin Si
4. Si (tPresionado < dit) Entonces
    4.1. Num = Num << 1
    4.2. Num = Num + 1
   Caso Contrario
    4.3. Num = Num << 1
   Fin Si
5. Fin

MÉTODO printCaracter

1. Inicio
2. EsperaCompleta = tiempopromedioDah * 100
3. nuevaPalabra = EsperaCompleta * 5
4. Si (Num > 63) Entonces
    4.1. Ejecutar método printPuntuacion
    4.2. Retornar
   Fin Si
5. Presentar Caracteres[Num]
6. FraseEMIC = FraseEMIC + Caracteres[Num];
7. Fin

MÉTODO printEspacio

1. Inicio
2. Presentar ‘ ‘
3. FraseEMIC = FraseEMIC + " "
4. Fin

MÉTODO printPuntuacion

1. Inicio
2. byte marcaPuntuacion = '#'
3. En caso de (Num)
   - Caso 71: marcaPuntuacion = ':'
   - Caso 76: marcaPuntuacion = ','
   - Caso 84: marcaPuntuacion = '!'
   - Caso 94: marcaPuntuacion = '-'
   - Caso 101: marcaPuntuacion = '@'
   - Caso 106: marcaPuntuacion = '.'
   - Caso 115: marcaPuntuacion = '?'

   Fin En Caso
4. Presentar marcaPuntuacion
5. FraseEMIC = FraseEMIC + marcaPuntuacion;
6. Fin

MÉTODO speak(entero espera)

1. Inicio
2. Cadena ErrMsg =“Aún no ha escrito nada. Digite algo para traducirlo para Usted. Por favor.”
3. Esperar inicialización de Emic2
4. Si (FraseEmic.tamaño > 0) Entonces
   - 4.1. Emitir mensaje audible (texto)
   - 4.2. FraseEMIC = ""

   Caso Contrario
   - 4.3. Emitir mensaje audible (ErrMsg)

   Fin Si
5. Pausa (espera)
6. Fin

MÉTODO borrar_caracter (entero espera)

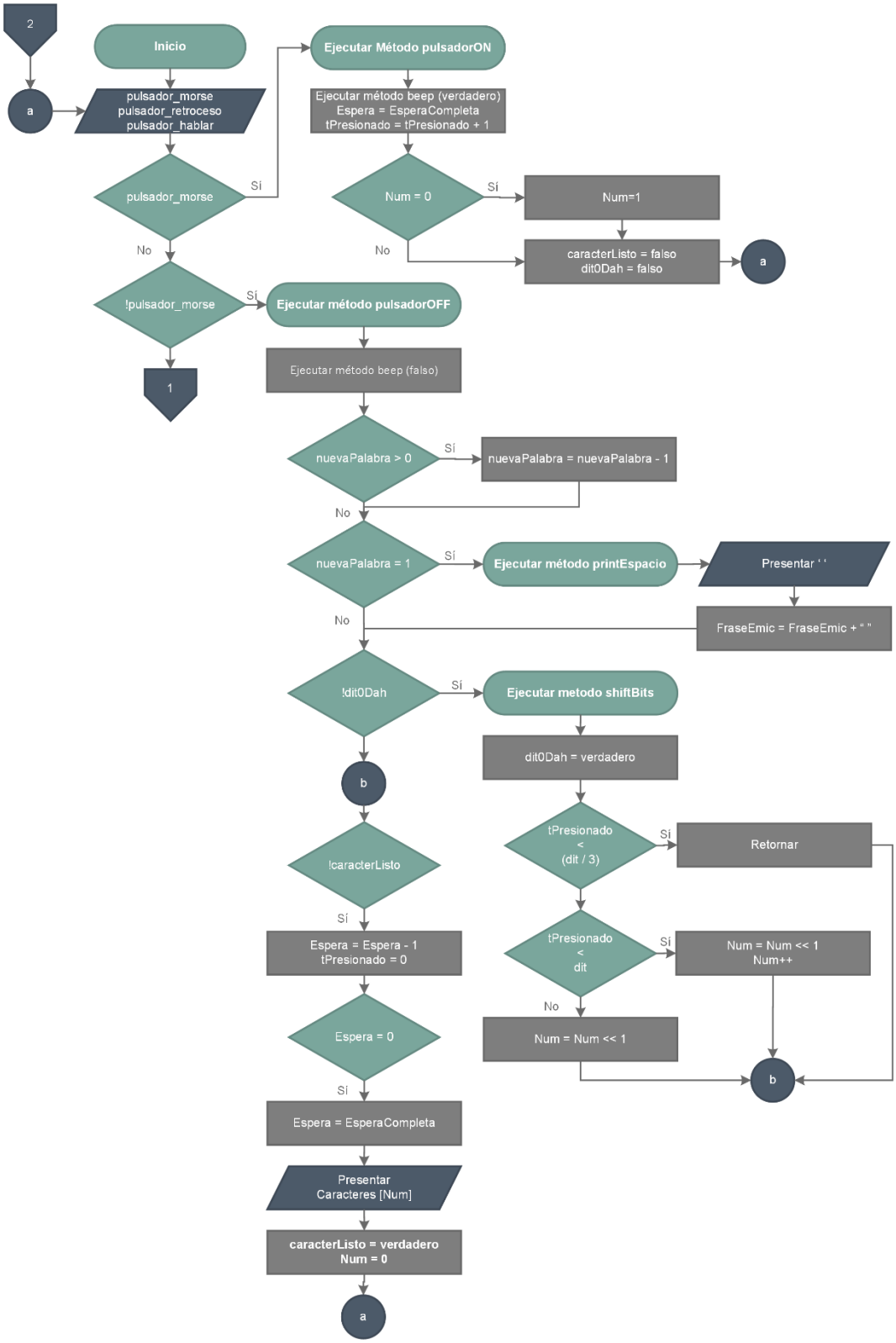
1. Inicio
2. entero tamCadena = (FraseEMIC.tamaño) - 1
4. Si (tamCadena >= 0) Entonces
    4.1. FraseEMIC.eliminar_caracter (tamCadena)
5. Pausa (espera)
6. Fin

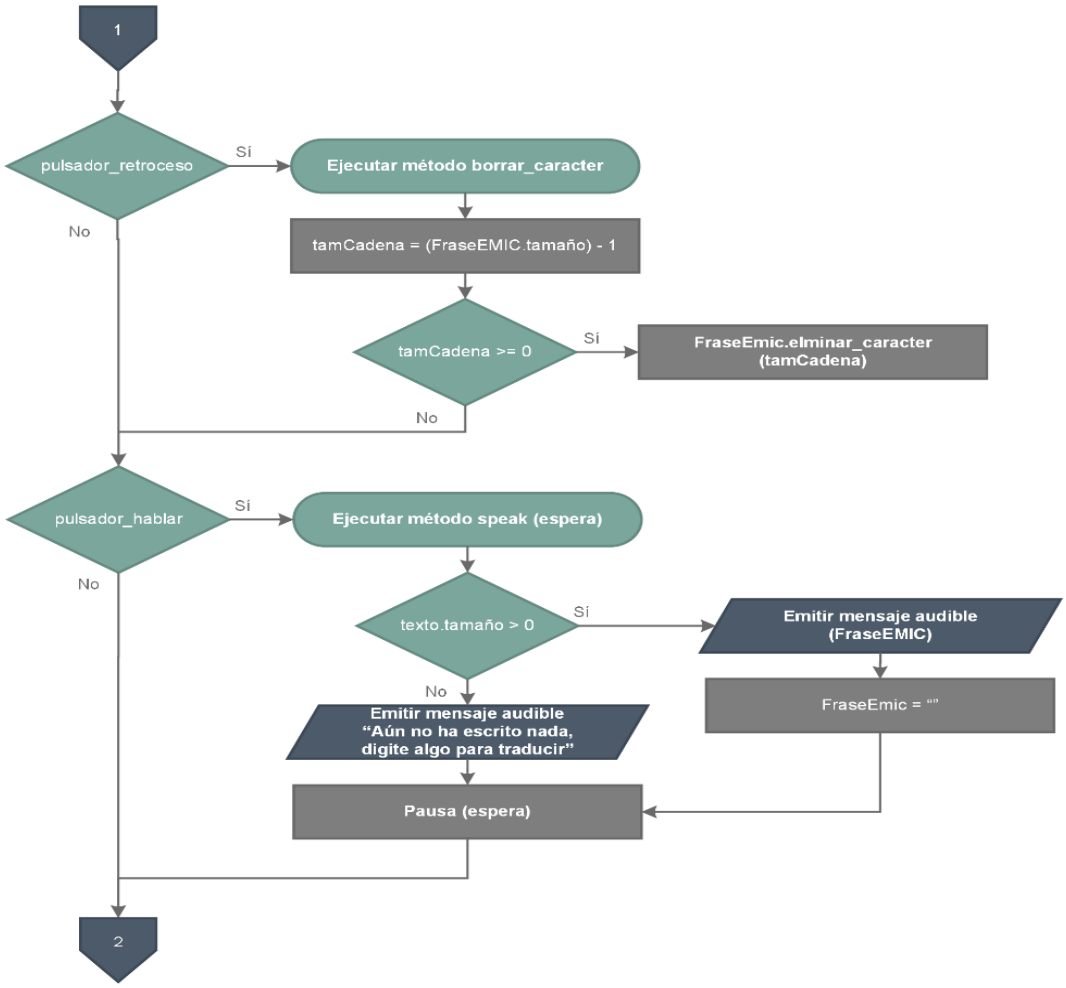
MÉTODO beep(booleano estado)

1. Inicio
2. Si (estado = verdadero) Entonces
        2.1. Emitir señal beep en Buzzer
    Caso Contrario
        2.2. Cancelar señal audible en Buzzer
    Fin Si
3. Fin

5.3. Diagrama de flujo de la aplicación

Ilustración 23: Diagrama de flujo de la aplicación





Elaboración: El Autor

Conclusiones

* Representar los puntos (Dit) y rayas (Dah) de forma binaria, 1’s y 0’s respectivamente, proporcionó una técnica que simplificó la conversión del carácter digitado del Lenguaje Código Morse a carácter del lenguaje textual.

* La técnica desplazamiento de bits (Bit Shifting), aportó la fórmula para que cada pulsación —ya sea punto (dit) o rayas (dah)— sea almacenada correctamente para su posterior interpretación al carácter correspondiente del Código Morse.

* La técnica Bit de descarte, solucionó el problema que representaba almacenar combinaciones del Código Morse que comenzaren con ceros. Por ejemplo: La letra C, que en Código Morse es representada por **—.—.** , la aplicación la representaría como 0101. El bit de descarte (**1**0101), entonces, se agregó un bit al inicio de la combinación sin alterar la posterior interpretación del Código Morse.

* La implementación del algoritmo permitió la simulación y comprobación de resultados experimentales para evaluar parámetros que se utilizan en la aplicación.

* El diseño y fabricación del dispositivo para traducir lenguaje Código Morse a expresión sonora para personas con discapacidad sensorial, se logró mediante el trabajo conjunto de la placa Arduino Uno con el módulo Emic2 Text-to-Speech, facultando la traducción del Código Morse a expresión sonora gracias a la aplicación desarrollada en el IDE Arduino.

* Las pruebas de funcionamiento del dispositivo determinaron que los casos de prueba que generaron error fueron de un 56,60%. Todos estos casos fueron corregidos satisfactoriamente para el funcionamiento óptimo del dispositivo.

* El entorno IDE de Arduino, es una herramienta de programación de manejo intuitivo, con una interfaz amigable y fácil de entender.

* La utilización del módulo Emic2 Text-to-Speech, permitió convertir el texto obtenido a partir del Código Morse a lenguaje sonoro sin restricciones o límite de palabras.

* La presentación visual del texto obtenido del Código Morse, se realizó gracias a la incorporación de la pantalla LCD (16 columnas por 2 filas, 32 caracteres) y su módulo integrado I2C.

* La Metodología de Desarrollo de Hardware Libre, goza de una amplia adaptabilidad y flexibilidad; es sencilla de aplicar, pues no requiere de gran cantidad de especificaciones y permiten reconocer requisitos para dispositivos de manera ágil.

Recomendaciones

* En el ámbito del desarrollo software, es recomendable trabajar con el lenguaje nativo de Arduino, C++ por: a) su familiaridad; b) genera programas compactos; c) el manejo de memoria es eficiente: y, d) portabilidad.

* En el ámbito de hardware, se recomienda usar hardware Arduino por: a) Disponer de una licencia libre, b) Tener una amplia documentación y c) Gozar de una gran comunidad de usuarios, lista a brindar soporte cuando sea necesario.

* Es recomendable usar el bus I2C, ya que, gracias a éste, no es necesaria la presencia de controladores de adaptación de tensión externos (integración de dispositivos).

* Es altamente recomendable el uso del módulo Emic2 Text-to-Speech, porque posee una amplia documentación, es de fácil implementación, y conversiones a lenguaje sonoro pueden realizarse en varios lenguajes y tonalidades..

* La herramienta libre Fritzing, es recomendable porque representa un excelente apoyo para la creación de esquemas de circuitos impresos y para el diseño electrónico de prototipos.

* Si fuese necesario realizar modificaciones y compilar nuevamente el código fuente, se recomienda buscar los controladores de la placa Arduino, por número de revisión.

* Es recomendable el uso del dispositivo electrónico en centros de capacitación para personas con discapacidad sensorial visual—auditiva.

* Es recomendable utilizar el tipo de datos más pequeño que se pueda para almacenar datos, para optimizar la memoria de la Arduino Uno, debido a que cuenta con tamaño limitado de almacenamiento.

Mejoras Potenciales para el Dispositivo

La realización del presente proyecto se ha desarrollado como un prototipo. Entre las mejoras potenciales que el dispositivo puede ostentar, se lista las siguientes:

* La estética puede mejorarse imprimiendo el chasis en 3D, reduciendo el tamaño y maximizando presentación del dispositivo.

* El volumen del altavoz, puede aumentarse utilizando un altavoz más potente con un amplificador electrónico.

* Alimentar de corriente al dispositivo electrónico, utilizando baterías recargables, eliminando la utilización del cable conectado al dispositivo.

* Es posible también, actualizar la pantalla LCD por una pantalla táctil, lo que permitiría suprimir los pulsadores y realizar el proceso de entrada de Código Morse desde la pantalla.

Bibliografía

Annan, K. (2003). *Discurso inaugural de la primera fase de la WSIS.* Ginebra.

Cáceres Rodríguez, C. (2004). Sobre el concepto de discapacidad. *Revista Electrónica de Audiología, Vol. 2*, 1-3.

Canós, J. H., Letelier, P., & Penadés, M. C. (12 de Noviembre de 2003). Metodologías ágiles en el Desarrollo del Software. Alicante, España.

D´Addario, M. (2015). *Manual de Electrónica.* Comunidad Europea.

ECDA. (14 de enero de 2014). *ECDA*. Obtenido de ECDA: http://elcajondeardu.blogspot.com/2014/01/tutorial-haciendo-sonidos-con-ardu.html

Guenther, J. A. (Diciembre de 1973). Machine Recognition Of Hand-Sent Morse Code Using PDP-12 Computer. USA.

Herrador, R. (2009). *Guia de Usuario de Arduino.* Córdoba, España.

Laster, C. (1984). *Guia del Radioaficonado Principiante.* Barcelona: S.A. Marcombo.

Medrano, A., Araujo, A., Soto, C., Dias, D., Moreno, G., & Colina, H. (01 de Junio de 2008). Metodología de Desarrollo en Hardware Libre. Mérida, Venezuela.

Parallax Inc. (15 de Octubre de 2015). Emic 2 Text-to-Speech Module (#30016).

Petzold, C. (2009). *Code: The Hidden Language of Computer Hardware and Software.* Microsoft Press books.

Torrente Artero, Ó. (2013). *Arduino: Curso práctico de formación.* Alfaomega Grupo Editor, S.A.

Anexos

Anexo A: Ficha Técnica

Especificación de requisitos hardware y software para el dispositivo para traducir lenguaje Código Morse a Expresión Sonora.

Introducción:

Desarrollo de un dispositivo para traducir lenguaje Código Morse a expresión sonora para personas con discapacidad sensorial.

1. Propósito

Asistir a las personas con discapacidad sensorial con un dispositivo electrónico alternativo de comunicación, que les permita afianzar una conversación con resto de la sociedad desconoce el lenguaje de comunicación propio de las personas con discapacidades sensoriales.

1.2. Alcance

Desarrollo de un dispositivo para traducir Código Morse a expresión sonora para personas con discapacidad sensorial visual—auditiva, que permita la comunicación —de este grupo de personas— en el entorno familiar y social.

2. Requerimientos de hardware y software.

2.1. Resumen de requerimientos de hardware.

Está sección entrega un resumen de todos los requerimientos de hardware.

• RH1 Ingresar datos mediante los pulsadores

• RH2 Visualizar los caracteres en la pantalla LCD.

• RH3 Escuchar los datos digitados mediante lenguaje audible

2.2. Resumen de requerimientos de software.

Está sección entrega un resumen de todos los requerimientos de software.

• RS1 Mantener el software disponible en todo momento.

• RS2 Traducir los caracteres ingresados a expresión sonora.

3. Detalle de requerimientos de hardware y software

3.1. Identificación de requisitos funcionales

• El dispositivo será capaz de interpretar el Código Morse mediante un pulsador.

• El dispositivo será capaz de traducir Código Morse ingresado a expresión sonora.

Entradas

• Pulsador 1: para caracteres usando el Código Morse.

• Pulsador 2: para borrar caracteres desde la posición actual hacia atrás.

• Pulsador 3: para emitir como expresión audible de los caracteres ingresados.

Salidas

• Se desplegarán los caracteres ingresados en pantalla.

• Se emitirá el sonido correspondiente mediante la salida de audio.

3.2. Especificaciones eléctricas y funcionales

• Implementación de una placa Arduino, y pulsadores para ingresar caracteres en el dispositivo.

• Implementación de módulo Emic 2 Text to Speech.

3.3. Requerimientos mecánicos

Tanto los componentes utilizados en la placa Arduino, como los pulsadores con sus respectivas resistencias, se incluyeron en la placa del circuito impreso de acuerdo a las necesidades del dispositivo

3.3.1. Requerimientos del desarrollo de lenguaje del software.

• Lenguaje Arduino

• Software Arduino 1.6.12,

• Librería LiquidCrystal_I2C.h

• Sistema operativo Windows 10

3.3.2. Requerimientos del circuito Impreso

• Esquema del diseño impreso en PCB.

3.3.3. Ubicación de componentes

• Ubicación de pulsadores en la placa.

• La pantalla en la parte superior, junto a los pulsadores.

• Arduino y la placa impresa juntos por debajo de la pantalla.

• El módulo Emic 2 a un lado de las placas arduino y circuito impreso.

3.4. Restricciones Específicas

• Conocer el lenguaje Código Morse.

3.5. Requisitos Específicos

La interfaz del producto consta de una pantalla LCD de 16x2, mediante la cual se visualizarán los caracteres digitados. Tres pulsadores se situarán en la parte inferior de la pantalla LCD

• Pulsador 1: Permitirá digitar Código Morse

• Pulsador 2: Borra un carácter desde la posición actual hacia atrás

• Pulsador 3: Traducirá el texto digitado a expresión audible.

Anexo B: Priorización de Funcionalidades

Tabla 22: Priorización de Funcionalidades

Priorización de Funcionalidades		
Nº	**Prioridad**	**Funcionalidad**
1	Alta	El Pulsador1 y su codificación correspondiente, deberán obtener la equivalencia a DOT, DASH y los Espacios con la finalidad de interpretar correctamente el mensaje que ingresa el usuario.
2	Alta	Los caracteres correspondientes al Código Morse ingresado mediante el Pulsador1, deberán mostrarse en la Pantalla LCD. (Y también en el monitor serial provisto en Arduino IDE por motivos de depuracion)
3	Alta	El Pulsador3 y su codificación correspondiente, deberán emitir la expresión sonora precedente del texto ingresado mediante el Pulsador1, y en caso de no haber escrito nada, deberá informar mediante un mensaje audible.
4	Media	El Pulsador2 y su codificación correspondiente, deberán borrar un carácter —desde la posición actual— hacia la izquierda tras cada pulsación.

Fuente: Anexo A
Elaboración: El Autor

Anexo C: Cronograma de Actividades

Tabla 23: Cronograma de Actividades

DESARROLLO DE UN DISPOSITIVO PARA TRADUCIR LENGUAJE CÓDIGO MORSE A EXPRESIÓN SONORA, PARA PERSONAS CON DISCAPACIDAD SENSORIAL.		
Actividad Actual	**Duración**	**Fecha**
1. Proceso de Conceptualización	**15 días**	**jueves, 10 de marzo de 2016**
Análisis y reflexión sobre problemas y soluciones	5 días	jueves, 17 de marzo de 2016
Definición o Actualización del alcance del proyecto de Hardware Libre	5 días	jueves, 24 de marzo de 2016
Elaboración de la propuesta de desarrollo del proyecto de Hardware Libre	5 días	jueves, 31 de marzo de 2016
2. Proceso de Administración de Desarrollo de Hardware Libre	**30 días**	**jueves, 31 de marzo de 2016**
Descripción de la aplicación a desarrollar	5 días	jueves, 07 de abril de 2016

Elaboración o actualización del plan de proyecto	5 días	jueves, 14 de abril de 2016
Conformación de la comunidad de desarrollo	5 días	jueves, 21 de abril de 2016
Elaboración del Plan por Iteración	5 días	jueves, 28 de abril de 2016
Seguimiento de las tareas que realiza el equipo de desarrollo	5 días	jueves, 05 de mayo de 2016
Integración del proyecto de los aportes de los colaboradores	5 días	jueves, 12 de mayo de 2016
3. Proceso de Desarrollo de Proyectos en Hardware Libre	**211 días**	**jueves, 12 de mayo de 2016**
Especificación del hardware estático	18 días	martes, 07 de junio de 2016
Programación de dispositivos	120 días	martes, 22 de noviembre de 2016
Desarrollo de Circuitos Integrados (IC)	10 días	martes, 06 de diciembre de 2016

Integración	18 días	viernes, 30 de diciembre de 2016
Verificación y Simulación	20 días	viernes, 27 de enero de 2017
Fabricación de Prototipos	4 días	jueves, 02 de febrero de 2017
Pruebas	20 días	jueves, 02 de marzo de 2017
Liberación	1 día	viernes, 03 de marzo de 2017

Elaboración: El Autor

Anexo D: Plan por Iteración

Tabla 24: Planificación para la primera iteración y su estado inicial

Planificación para la primera iteración y su estado inicial			
Tarea	**Tipo**	**Estado**	**Responsable**
Diseño de la arquitectura del circuito	Diseño	Terminado	José Calva
Conexión de la placa Arduino Uno al protoboard	Circuito	Terminado	José Calva
Conexión del Pulsador 1 y Resistencia 1 al protoboard	Circuito	Terminado	José Calva
Conexión del Pulsador 2 y Resistencia 2 al protoboard	Circuito	Terminado	José Calva
Conexión del Pulsador 3 y Resistencia 3 al protoboard	Circuito	Terminado	José Calva
Conexión del Buzzer y Resistencia 4 al protoboard	Circuito	Terminado	José Calva
Conexión y acoplamiento de la pantalla LCD a la placa Arduino	Circuito	Terminado	José Calva
Conexión del módulo de Emic 2 a la placa Arduino y al protoboard	Circuito	Terminado	José Calva
Conexión y soldadura del altavoz al módulo de Emic 2	Circuito	Terminado	José Calva
Pruebas del dispositivo	Control	Terminado	José Calva

Elaboración: El Autor

Tabla 25: Planificación para la segunda iteración y su estado inicial

Planificación para la segunda iteración y su estado inicial			
Tarea	**Tipo**	**Estado**	**Responsable**
Elaboración del Algoritmo	Diseño	Terminado	José Calva
Implementación Botón 1: Ingreso de datos	Codificación.	Terminado	José Calva
Implementación Botón 2: Borrado de caracteres	Codificación.	Terminado	José Calva
Implementación Botón 3: Emisión de señal auditiva	Codificación.	Terminado	José Calva
Implementación Pantalla LCD (Salida de datos)	Codificación.	Terminado	José Calva
Pruebas del dispositivo	Control	Terminado	José Calva

Elaboración: El Autor

Tabla 26: Planificación para la tercera iteración y su estado inicial

Planificación para la tercera iteración y su estado inicial			
Tarea	**Tipo**	**Estado**	**Responsable**
Conexión del módulo Emic 2	Circuito	Terminado	José Calva
Implementación del módulo Emic 2 Text-to-speech	Codificación	Terminado	José Calva
Nivelar volumen de altavoz	Codificación	Terminado	José Calva
Pruebas del dispositivo	Control	Terminado	José Calva

Elaboración: El Autor

Tabla 27: Planificación para la cuarta iteración y su estado inicial

Planificación para la cuarta iteración y su estado inicial			
Tarea	**Tipo**	**Estado**	**Responsable**
Diseño de circuitos de placa impresa	Diseño	Terminado	José Calva
Preparación del chasis	Ensamblaje	Terminado	José Calva
Ensamblaje del dispositivo	Ensamblaje	Terminado	José Calva
Pruebas del dispositivo	Control	Terminado	José Calva

Elaboración: El Autor

Anexo E: Seguimiento de Tareas

Tabla 28: Seguimiento Primera Iteración

Seguimiento Primera Iteración						
Fecha	**Tareas Pendientes**	**Horas Restantes**	**Responsable**	**Actividad Anterior**	**Actividad Actual**	**Requerimientos Extras**
jueves, 10 de marzo de 2016	10	52	José Calva	—	Planificación de la iteración	—
viernes, 11 de marzo de 2016	10	48	José Calva	Planificación de la iteración	Diseño de la arquitectura del circuito	Ninguno
lunes, 14 de marzo de 2016	9	44	José Calva	Diseño de la arquitectura del circuito	Conexión de la placa Arduino Uno al protoboard	Ninguno
martes, 15 de marzo de 2016	8	40	José Calva	Conexión de la placa Arduino Uno al protoboard	Conexión del Pulsador 1 y Resistencia 1 al protoboard	Ninguno
miércoles, 16 de marzo de 2016	7	36	José Calva	Conexión del Pulsador 1 y Resistencia 1 al protoboard	Conexión del Pulsador 2 y Resistencia 2 al protoboard	Ninguno
jueves, 17 de marzo de 2016	6	32	José Calva	Conexión del Pulsador 2 y Resistencia 2 al protoboard	Conexión del Pulsador 3 y Resistencia 3 al protoboard	Ninguno

viernes, 18 de marzo de 2016	5	28	José Calva	Conexión del Pulsador 3 y Resistencia 3 al protoboard	Conexión del Buzzer y Resistencia 4 al protoboard	Ninguno
lunes, 21 de marzo de 2016	4	24	José Calva	Conexión del Buzzer y Resistencia 4 al protoboard	Conexión y acoplamiento de la pantalla LCD a la placa Arduino	Ninguno
martes, 22 de marzo de 2016	3	20	José Calva	Conexión y acoplamiento de la pantalla LCD a la placa Arduino	Conexión del módulo de Emic 2 a la placa Arduino y al protoboard	Ninguno
miércoles, 23 de marzo de 2016	2	16	José Calva	Conexión del módulo de Emic 2 a la placa Arduino y al protoboard	Conexión del módulo de Emic 2 a la placa Arduino y al protoboard	Ninguno
jueves, 24 de marzo de 2016	2	12	José Calva	Conexión del módulo de Emic 2 a la placa Arduino y al protoboard	Conexión y soldadura del altavoz al módulo de Emic 2	Ninguno
viernes, 25 de marzo de 2016	1	8	José Calva	Conexión y soldadura del altavoz al módulo de Emic 2	Pruebas del dispositivo	Ninguno
lunes, 28 de marzo de 2016	1	4	José Calva	Pruebas del dispositivo	Pruebas del dispositivo	Ninguno
miércoles, 30 de marzo de 2016	**0**	**0**	**José Calva**	**Pruebas del dispositivo**	**Iteración Terminada**	**—**

Elaboración: El Autor

Tabla 29: Seguimiento Segunda Iteración

Seguimiento Segunda Iteración						
Fecha	**Tareas Pendientes**	**Horas Restantes**	**Responsable**	**Actividad Anterior**	**Actividad Actual**	**Requerimientos Extras**
miércoles, 30 de marzo de 2016	6	500	José Calva	—	Planificación de la iteración	—
miércoles, 30 de marzo de 2016	6	500	José Calva	Planificación de la iteración	Elaboración del Algoritmo	Ninguno
miércoles, 04 de mayo de 2016	5	400	José Calva	Elaboración del Algoritmo	Implementación Botón 1: Ingreso de datos	Ninguno
viernes, 20 de mayo de 2016	4	350	José Calva	Programar botón uno, de ingreso de datos	Implementación Botón 2: Borrado de caracteres	Ninguno
viernes, 24 de junio de 2016	3	250	José Calva	Implementación Botón 2: Borrado de caracteres	Implementación Botón 3: Emisión de señal auditiva	Ninguno
viernes, 29 de julio de 2016	2	150	José Calva	Implementación Botón 3: Emisión de señal auditiva	Implementación Pantalla LCD (Salida de datos)	Ninguno
miércoles, 17 de agosto de 2016	1	50	José Calva	Programar pantalla, para salida de datos.	Pruebas del dispositivo	Ninguno
lunes, 05 de septiembre de 2016	**0**	**0**	**José Calva**	**Pruebas del dispositivo**	**Iteración Terminada**	**—**

Elaboración: El Autor

Tabla 30: Seguimiento Tercera Iteración

Seguimiento Tercera Iteración						
Fecha	**Tareas Pendientes**	**Horas Restantes**	**Responsable**	**Actividad Anterior**	**Actividad Actual**	**Requerimientos Extras**
lunes, 05 de septiembre de 2016	4	350	José Calva	—	Planificación de la iteración	—
lunes, 05 de septiembre de 2016	4	350	José Calva	Planificación de la iteración	Conexión del módulo Emic 2	Ninguno
miércoles, 21 de septiembre de 2016	3	300	José Calva	Conexión del módulo Emic 2	Implementación del módulo Emic 2 Text-to-speech	Ninguno
lunes, 14 de noviembre de 2016	2	150	José Calva	Implementación del módulo Emic 2 Text-to-speech	Nivelar volumen de altavoz	Ninguno
miércoles, 30 de noviembre de 2016	1	100	José Calva	Nivelar volumen de altavoz	Pruebas del dispositivo	Ninguno
miércoles, 04 de enero de 2017	**0**	**0**	**José Calva**	**Pruebas del dispositivo**	**IteraciónTerminada**	**—**

Elaboración: El Autor

Tabla 31: Seguimiento Cuarta Iteración

Seguimiento Cuarta Iteración						
Fecha	**Tareas Pendientes**	**Horas Restantes**	**Responsable**	**Actividad Anterior**	**Actividad Actual**	**Requerimientos Extras**
miércoles, 04 de enero de 2017	4	120	José Calva	—	Planificación de la iteración	.....
miércoles, 04 de enero de 2017	4	120	José Calva	Planificación de la iteración	Diseño de circuitos de placa impresa	Ninguno
miércoles, 18 de enero de 2017	3	90	José Calva	Diseño de circuitos de placa impresa	Preparación del chasis	Ninguno
miércoles, 01 de febrero de 2017	2	60	José Calva	Preparación del chasis	Ensamblaje del dispositivo	Ninguno
miércoles, 15 de febrero de 2017	1	30	José Calva	Ensamblaje del dispositivo	Pruebas del dispositivo	Ninguno
viernes, 03 de marzo de 2017	**0**	**0**	**José Calva**	**Pruebas del dispositivo**	**Iteración Terminada**	**—**

Elaboración: El Autor

Anexo F: Validación de Resultados por Tarea

Tabla 32: Validación de resultados por tarea — Iteración 1

Validación de resultados por tarea: Iteración 1				
Tarea	**Responsable**	**Duración Horas**	**Resultado**	**Finalizado**
Diseño de la arquitectura del circuito	José Calva	8	Positivo	Finalizado
Conexión de la placa Arduino Uno al protoboard	José Calva	4	Positivo	Finalizado
Conexión del Pulsador 1 y Resistencia 1 al protoboard	José Calva	4	Positivo	Finalizado
Conexión del Pulsador 2 y Resistencia 2 al protoboard	José Calva	4	Positivo	Finalizado
Conexión del Pulsador 3 y Resistencia 3 al protoboard	José Calva	4	Positivo	Finalizado
Conexión del Buzzer y Resistencia 4 al protoboard	José Calva	4	Positivo	Finalizado
Conexión y acoplamiento de la pantalla LCD a la placa Arduino	José Calva	4	Positivo	Finalizado
Conexión del módulo de Emic 2 a la placa Arduino y al protoboard	José Calva	4	Positivo	Finalizado
Conexión y soldadura del altavoz al módulo de Emic 2	José Calva	8	Positivo	Finalizado
Pruebas del dispositivo	José Calva	8	Positivo	Finalizado
Tareas Cumplidas	**José Calva**	**52**	**—**	**—**

Elaboración: El Autor

Tabla 33: Validación de resultados por tarea — Iteración 2

Validación de resultados por tarea: Iteración 2				
Tarea	**Responsable**	**Duración Horas**	**Resultado**	**Finalizado**
Elaboración del Algoritmo	José Calva	100	Positivo	Finalizado.
Implementación Botón 1: Ingreso de datos	José Calva	100	Positivo	Finalizado
Implementación Botón 2: Borrado de caracteres	José Calva	50	Positivo	Finalizado
Implementación Botón 3: Emisión de señal auditiva	José Calva	100	Positivo	Finalizado
Implementación Pantalla LCD (Salida de datos)	José Calva	100	Positivo	Finalizado
Pruebas del dispositivo	José Calva	50	Positivo	Finalizado
Tareas Cumplidas	**José Calva.**	**500**	**—**	**—**

Elaboración: El Autor

Tabla 34: Validación de resultados por tarea — Iteración 3

Validación de resultados por tarea: Iteración 3				
Tarea	**Responsable**	**Duración horas**	**Resultado**	**Finalizado**
Conexión del módulo Emic 2	José Calva	50	Positivo	Finalizado
Implementación del módulo Emic 2 Text-to-speech	José Calva	150	Positivo	Finalizado
Nivelar volumen de altavoz	José Calva	50	Positivo	Finalizado
Pruebas del dispositivo	José Calva	100	Positivo	Finalizado
Tareas Cumplidas	**José Calva**	**350**	**—**	**—**

Elaboración: El Autor

Tabla 35: Validación de resultados por tarea — Iteración 4

Validación de resultados por tarea: Iteración 4				
Tarea	**Responsable**	**Duración horas**	**Resultado**	**Finalizado**
Diseño de circuitos de placa impresa	José Calva	30	Positivo	Finalizado
Preparación del chasis	José Calva	30	Positivo	Finalizado
Ensamblaje del dispositivo	José Calva	30	Positivo	Finalizado
Pruebas del dispositivo	José Calva	30	Positivo	Finalizado
Tareas Cumplidas	**José Calva**	**120**	**—**	**—**

Elaboración: El Autor

Anexo G: Especificación de Casos de Prueba

Tabla 36: Especificación de casos de prueba

Especificación de casos de prueba								
N°	Módulo-Dispositivo	Escenario	Caso de Prueba	Resultado Esperado	Ejecución Satisfactoria	Impacto	Severidad	Estado
1	Circuito	El dispositivo debe encenderse	Al conectar el dispositivo mediante alimentación, el LED de encendido de la placa Arduino debe iluminarse	El indicador LED de encendido debe iluminarse	Si	Alto	Alta	Finalizado
2	Código Fuente	Verificación lógica y de temporización.	Validación de tiempos de pulsación para el Pulsador 1	Al presionar el pulsador 1 durante 100ms debe considerarse como un DOT. Al presionar el pulsador 1 durante 150ms debe considerarse como un DASH.	Si	Alto	Alta	Finalizado
3	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-**).	Ingresar la combinación morse correspondiente a: (**T**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (T).	Si	Bajo	Media	Finalizado

4	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (.).	Ingresar la combinación morse correspondiente a: (**E**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (E).	Si	Bajo	Media	Finalizado
5	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**--**).	Ingresar la combinación morse correspondiente a: (**M**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (M).	Si	Bajo	Media	Finalizado
6	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-.**)	Ingresar la combinación morse correspondiente a: (**N**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (N).	Si	Bajo	Media	Finalizado
7	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.-**).	Ingresar la combinación morse correspondiente a: (**A**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (A).	Si	Bajo	Media	Finalizado
8	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**..**).	Ingresar la combinación morse correspondiente a: (**I**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse,(I).	Si	Bajo	Media	Finalizado

9	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**---**).	Ingresar la combinación morse correspondiente a: (**O**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (O).	Si	Bajo	Media	Finalizado
10	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**--.**).	Ingresar la combinación morse correspondiente a: (**G**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (G).	Si	Bajo	Media	Finalizado
11	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-.-**).	Ingresar la combinación morse correspondiente a: (**K**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (K).	Si	Bajo	Media	Finalizado
12	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-..**).	Ingresar la combinación morse correspondiente a: (**D**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (D).	Si	Bajo	Media	Finalizado
13	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.--**).	Ingresar la combinación morse correspondiente a: (**W**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (W).	Si	Bajo	Media	Finalizado

14	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.-.**).	Ingresar la combinación morse correspondiente a: (**R**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (R).	Si	Bajo	Media	Finalizado
15	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**..-**).	Ingresar la combinación morse correspondiente a: (**U**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (U).	Si	Bajo	Media	Finalizado
16	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**...**) .	Ingresar la combinación morse correspondiente a: (**S**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (S).	Si	Bajo	Media	Finalizado
17	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**--.-**).	Ingresar la combinación morse correspondiente a: (**Q**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (Q).	Si	Bajo	Media	Finalizado
18	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**--..**).	Ingresar la combinación morse correspondiente a: (**Z**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (Z).	Si	Bajo	Media	Finalizado

19	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-.--**).	Ingresar la combinación morse correspondiente a: (**Y**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (Y).	Si	Bajo	Media	Finalizado
20	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-.-.**).	Ingresar la combinación morse correspondiente a: (**C**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (C).	Si	Bajo	Media	Finalizado
21	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-..-**).	Ingresar la combinación morse correspondiente a: (**X**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (X).	Si	Bajo	Media	Finalizado
22	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-...**).	Ingresar la combinación morse correspondiente a: (**B**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (B).	Si	Bajo	Media	Finalizado
23	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.---**).	Ingresar la combinación morse correspondiente a: (**J**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (J).	Si	Bajo	Media	Finalizado

24	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.--.**).	Ingresar la combinación morse correspondiente a: (**P**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (P).	Si	Bajo	Media	Finalizado
25	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.-..**).	Ingresar la combinación morse correspondiente a: (**L**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (L).	Si	Bajo	Media	Finalizado
26	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**..-.**).	Ingresar la combinación morse correspondiente a: (**F**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (F).	Si	Bajo	Media	Finalizado
27	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**...-**).	Ingresar la combinación morse correspondiente a: (**V**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (V).	Si	Bajo	Media	Finalizado
28	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**....**).	Ingresar la combinación morse correspondiente a: (**H**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (H).	Si	Bajo	Media	Finalizado

29	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-----**).	Ingresar la combinación morse correspondiente a: (**0**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (0).	Si	Bajo	Media	Finalizado
30	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**----.**).	Ingresar la combinación morse correspondiente a: (**9**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (9).	Si	Bajo	Media	Finalizado
31	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**---..**).	Ingresar la combinación morse correspondiente a: (**8**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (8).	Si	Bajo	Media	Finalizado
32	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**--...**).	Ingresar la combinación morse correspondiente a: (**7**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (7).	Si	Bajo	Media	Finalizado
33	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-....**).	Ingresar la combinación morse correspondiente a: (**6**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (6).	Si	Bajo	Media	Finalizado

34	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.----**).	Ingresar la combinación morse correspondiente a: (**1**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (1).	Si	Bajo	Media	Finalizado
35	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**..---**).	Ingresar la combinación morse correspondiente a: (**2**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (2).	Si	Bajo	Media	Finalizado
36	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**...--**).	Ingresar la combinación morse correspondiente a: (**3**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (3).	Si	Bajo	Media	Finalizado
37	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**....-**).	Ingresar la combinación morse correspondiente a: (**4**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (4).	Si	Bajo	Media	Finalizado
38	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.....**).	Ingresar la combinación morse correspondiente a: (**5**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (5).	Si	Bajo	Media	Finalizado

39	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**---...**).	Ingresar la combinación morse correspondiente a: (**:**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (:).	Si	Bajo	Media	Finalizado
40	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**--..--**).	Ingresar la combinación morse correspondiente a: (**,**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (,).	Si	Bajo	Media	Finalizado
41	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**..--..**).	Ingresar la combinación morse correspondiente a: (**!**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (!).	Si	Bajo	Media	Finalizado
42	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-.-.-.-**).	Ingresar la combinación morse correspondiente a: (**-**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (-).	Si	Bajo	Media	Finalizado
43	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**-.-----.**).	Ingresar la combinación morse correspondiente a: (**@**)	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (@)	Si	Bajo	Media	Finalizado

44	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**.-.-.-**).	Ingresar la combinación morse correspondiente a: (**.**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse, (.).	Si	Bajo	Media	Finalizado
45	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar la combinación correspondiente a: (**..--..**).	Ingresar la combinación morse correspondiente a: (**?**).	Se debe visualizar en pantalla el carácter correspondiente a cada combinación de morse (?).	Si	Bajo	Media	Finalizado
46	Pulsador 1	Al presionar el pulsador 1, el Buzzer debe emitir un sonido, y la pantalla debe mostrar un carácter (**#**).	Ingresar la combinación no especificada en morse.	Se debe visualizar en pantalla el carácter correspondiente a (#). Indicando que el carácter no existe	Si	Bajo	Media	Finalizado
47	Pulsador 2	Al presionar el pulsador 2, se borrará un carácter hacia atrás, y se mostrará en pantalla LCD.	Borrar un carácter.	Borrar un carácter hacia atrás.	Si	Bajo	Media	Finalizado
48	Pulsador 3	Al presionar el pulsador 3, interpretará los caracteres introducidos a sonido.	Emitir sonido.	El altavoz emitirá en audio la frase escrita.	Si	Bajo	Alto	Finalizado

49	Pulsador 3	Al presionar el pulsador 3, el altavoz deberá emitir un mensaje de sonido que dirá: "aún no ha escrito nada. Digite algo para traducirlo para usted, por favor"	Emitir mensaje predeterminado.	El altavoz emitirá en audio la frase predeterminada.	Si	Bajo	Bajo	Finalizado
50	Pulsador 1	al dejar de presionar el pulsador 1, se dará un espacio de carácter.	Dar un espacio de carácter	Espacio de carácter	si	Alto	Alta	Finalizado
51	Pantalla LCD	Mostrar datos en pantalla en lenguaje natural.	Imprimir los datos ingresados por el usuario, mediante el Lenguaje Código Morse.	Mostrar los datos ingresados por el usuario.	si	Alto	Alta	Finalizado
52	Emitir sonido	Emitir sonido de los datos que se encuentran en pantalla.	Emitir sonido, de los datos en pantalla	Emitir sonido en español.	si	Alto	Alta	Finalizado
53	Circuito	Verificación del PCB	Comprobación de Pistas	Pistas no defectuosas	si	Alto	Alta	Finalizado

Elaboración: El Autor

Anexo H: Tabla Internacional de Código Morse

Tabla 37: Tabla Internacional de Código Morse

Código Morse			
Codificación Morse	Carácter	Codificación Morse	Carácter
.-	A	...-	V
-...	B	.--	W
-.-.	C	-..-	X
-..	D	-.--	Y
.	E	--..	Z
..-.	F	-----	0
--.	G	.----	1
....	H	..---	2
..	I	...--	3
.---	J	....-	4
-.-	K	.....	5
.-..	L	-....	6
--	M	--...	7
-.	N	---..	8
---	O	----.	9
.--.	P	.-.-.-	.
--.-	Q	--..--	,
.-.	R	..--..	?
...	S	..--.	!
-	T	---...	:
..-	U		

Fuente:
Elaboración: El Autor

Anexo I: Código Fuente

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>

#define MORSEKEY        2
#define BACKSPACEKEY    9
#define SPEAKKEY        10
#define BUZZER          8
#define LCDMAXCOL       16
#define LCDMAXFIL       2
#define rxPin           5
#define txPin           6

/************************
 *  VARIABLES GLOBALES  *
 ************************/

LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
SoftwareSerial emicSerial =  SoftwareSerial(rxPin, txPin);

int pulsador_morse=0;
int pulsador_retroceso=0;
int pulsador_hablar=0;

int dit=100;
int tiempopromedioDah=150;
int tRebote=2;
int tPresionado=0;
int Num=0;

int x = 0, y = 0, pagina=0;

boolean ditODah=true;
boolean caracterListo=true;

long EsperaCompleta=10000;
long Espera=EsperaCompleta;
long nuevaPalabra=0;

char Caracteres[]
="##TEMNAIOGKDWRUS##QZYCXBJP#L#FVH09#8###7#######61#######2###3#45";

String FraseEMIC;

/****************************
 *  DEFINICIÓN DE MÉTODOS   *
 ****************************/

void speak(int espera);

void pulsadorON();
void pulsadorOFF();

void beep(bool state);
```

```
void printCaracter();
void printEspacio();
void printPuntuacion();
void shiftBits();

void borrar_caracter(int espera);

void updatelcd();
void updatelcd(char symbol);

void mostrarpagina(int nropagina);

/***************************
 *   METODO SETUP           *
 ***************************/

void setup() {
  pinMode(MORSEKEY, INPUT);
  pinMode(BUZZER,OUTPUT);
  digitalWrite(BUZZER,HIGH);
  Serial.begin(9600);

  pinMode(rxPin, INPUT);
  pinMode(txPin, OUTPUT);
  emicSerial.begin(9600);

  lcd.begin(LCDMAXCOL, LCDMAXFIL);
  lcd.backlight();
  lcd.print ("Trad. Cod. Morse");
  lcd.setCursor(0, 1);
  lcd.print ("Jose F. Calva");
  delay(1000);
  lcd.clear();

  emicSerial.print('\n');
  while (emicSerial.read() != ':');
  delay(10);
  emicSerial.flush();
  emicSerial.print("V18\n");
  emicSerial.print("N0\n");
  emicSerial.print("L2\n");
}

void loop() {
   pulsador_morse=digitalRead(MORSEKEY);
   pulsador_retroceso=digitalRead(BACKSPACEKEY);
   pulsador_hablar=digitalRead(SPEAKKEY);

   if (pulsador_retroceso){
        borrar_caracter(1000);
   }

   if (pulsador_hablar){
      speak (2000);
   }
```

```
   if (pulsador_morse) pulsadorON();
   if (!pulsador_morse) pulsadorOFF();
}

void speak(int espera){

    String ErrMsg = "A\xFAn no ha escrito nada. Digite algo para traducirlo
para usted, por favor.";

    while (emicSerial.read() != ':');

    if (FraseEMIC.length()>0){
        emicSerial.print('S');
        emicSerial.print(FraseEMIC);
        FraseEMIC = "";
    }else{
        emicSerial.print('S');
        emicSerial.print(ErrMsg);
    }

    emicSerial.print("\n");

    delay(espera);
    lcd.clear();
    x=0; y=0; nuevaPalabra=0;
    lcd.setCursor(x,y);
}

void beep(bool state){
  if (!state){
    digitalWrite(BUZZER,HIGH);
  }else{
    digitalWrite(BUZZER,LOW);
  }
}

void pulsadorON(){
  beep(true);
  Espera=EsperaCompleta;
  tPresionado++;

  //Bit Descartable
  if (Num==0) Num=1;

  caracterListo=false;
  ditODah=false;
  delay(tRebote);
}

void pulsadorOFF(){
  beep(false);
  if (nuevaPalabra>0) nuevaPalabra--;
  if (nuevaPalabra==1) printEspacio();

  if (!ditODah) shiftBits();
```

```
  if (!caracterListo) {
      Espera--;
      if (Espera==0) {
        Espera=EsperaCompleta;
        printCaracter();
        caracterListo=true;
        Num=0;
      }
      tPresionado=0;
   }
}

void printCaracter(){
  EsperaCompleta=tiempopromedioDah*100;
  nuevaPalabra=EsperaCompleta*5;
  if (Num>63) {
    printPuntuacion();
    return;
  }
  updatelcd();
  lcd.print(Caracteres[Num]);
  Serial.print(Caracteres[Num]);
  FraseEMIC = FraseEMIC + Caracteres[Num];
}

void printEspacio(){
  if ( x >= LCDMAXCOL) {
    updatelcd();
    Serial.println();
    return;
  }
  lcd.print(' ');
  Serial.print(' ');
  FraseEMIC = FraseEMIC +  " ";
  updatelcd();
}

void printPuntuacion(){
  byte marcaPuntuacion='#';
  switch (Num){
    case 71:  marcaPuntuacion=':'; break;
    case 76:  marcaPuntuacion=','; break;
    case 84:  marcaPuntuacion='!'; break;
    case 94:  marcaPuntuacion='-'; break;
    case 101: marcaPuntuacion='@'; break;
    case 106: marcaPuntuacion='.'; break;
    case 115: marcaPuntuacion='?'; break;
  }
  updatelcd();
  lcd.print(marcaPuntuacion);
  Serial.print(marcaPuntuacion);
  FraseEMIC = FraseEMIC +  marcaPuntuacion;
}
```

```
void shiftBits(){
  ditODah=true;
  if (tPresionado<dit/3) return;
  if (tPresionado<dit) {
    Num = Num << 1;
    Num++;
  }
  else {
    Num = Num << 1;
  }
}

void borrar_caracter(int espera){
     int tamCadena = FraseEMIC.length()-1;
     updatelcd(' ');
     if (tamCadena >= 0){
        FraseEMIC.remove(tamCadena);
     }
     delay(espera);
}

void mostrarpagina(int nropagina){
    lcd.clear();
    if (nropagina == 0) {
        int inicio  = 0;
        int fin     = LCDMAXCOL-1;
        x=0; y=0;
        lcd.setCursor(x,y);
        for (int a=inicio;a<=fin;a++) {
          lcd.print(FraseEMIC[a]);
          x++;
        }
        x=0; y++;

        lcd.setCursor(x,y);
        inicio  = LCDMAXCOL;
        fin     = (LCDMAXCOL*(LCDMAXFIL))-1;

        for (int a=inicio;a<=fin;a++) {
            lcd.print(FraseEMIC[a]);
            x++;
        }
    }
    else{
        //pagina 1,2,3,4 .... n

        int inicio  = (LCDMAXCOL*(pagina*2));
        int fin     =  inicio+(LCDMAXCOL-1);
        x=0; y=0;
        lcd.setCursor(x,y);
        for (int a=inicio;a<=fin;a++) {
          Serial.print(FraseEMIC[a]);
          lcd.print(FraseEMIC[a]);
          x++;
        }
```

```
        x=0;y++;
        lcd.setCursor(x,y);
        inicio  = (LCDMAXCOL*(pagina+(pagina+1)));
        fin     = (LCDMAXCOL * ((pagina+1)*2))-1;
        for (int a=inicio;a<=fin;a++) {
          Serial.print(FraseEMIC[a]);
          lcd.print(FraseEMIC[a]);
          x++;
        }
    }

}

void updatelcd(){
      lcd.setCursor(x, y);
      x++;
      if (y > LCDMAXFIL-1) {
        lcd.clear();
        x = 0;
        y = 0;
        pagina++;
      }
      if (x >= LCDMAXCOL) {
        x = 0;
        y++;
      }
}

void updatelcd(char symbol){
    if ((x == 0) && (y==1)){
        x=LCDMAXCOL;
        y=0;
    }
    if ((x==0) && (y==0) && (pagina>0)) {
        pagina--;
        mostrarpagina(pagina);
        return;
    }

    lcd.setCursor(x-1,y);
    lcd.print(symbol);

    if (x>0) x--;
}
```

Anexo J: Presupuesto

Tabla 38: Recursos Técnicos

Recursos Técnicos			
Recurso	**Cantidad**	**Precio Unitario**	**Total**
Laptop i5 propiedad el autor	1	$ 800,00	$ 800,00
Internet (tiempo horas al dia)	4	$ 0,60	$ 300,00
Licencias Open Source	1	$ 0,00	$ 0,00
TOTAL			**$ 1.100,00**

Elaboración: El Autor

Tabla 39: Recursos Humanos

Recursos Humanos				
Recurso	**Cantidad**	**Meses**	**Costo* Día**	**Total**
Tutor de tesis	-	12	-	$ 1.750,00
Analista	80h	1	$ 0,00	$ 0,00
Programador	200h	5		$ 0,00
Diseñador	50h	1	$ 0,00	$ 0,00
TOTAL				**$ 1.750,00**

Elaboración: El Autor

Tabla 40: Materiales del dispositivo

Materiales del dipositivo			
Recurso	**Cantidad**	**Precio Unitario**	**Total**
Arduino Uno	1	$ 35,00	$ 35,00
Protoboard	1	$ 5,00	$ 5,00
Módulo Speech	1	$ 130,00	$ 130,00
Pantalla LCD	1	$ 16,00	$ 16,00
Cables	20	$ 0,10	$ 2,00
Pulsadores	3	$ 0,20	$ 0,60
Placa circuito	1	$ 15,00	$ 15,00
Buzzer	1	$ 0,50	$ 0,50
Altavoz	1	$ 1,00	$ 1,00
Resistencias	4	$ 0,10	$ 0,40
Baterias	2	$ 3,50	$ 7,00
Chasis	1	$ 10,00	$ 10,00
TOTAL			**$ 222,50**

Elaboración: El Autor

Tabla 41: Presupuesto Total

Presupuesto Total	
Recurso	**Total**
Material Impresión	$ 239,59
Recursos Ténicos	$ 1.100,00
Recuros Humanos	$ 1.750,00
Materiales del dispositivo	$ 222,50
TOTAL	**$ 3.312,09**

Elaboración: El Autor

Anexo K: Glosario de Términos

Tabla 42: Glosario de Términos

Glosario de Términos	
Términos	**Descripción**
Lenguaje Natural	Lenguaje de comunicación del ser humano.
Lenguaje Dactilológico	Alfabeto Manual o Alfabeto de señas
OMS	Organización Mundial de la Salud
CIDDM	Clasificación Internacional de Deficiencias, Discapacidades y Minusvalías
TIC	Tecnologías de la Información y la Comunicación
LCD	(Liquid Crystal Display), Pantalla de Cristal líquido
IDE	(Integrated Developer Enviroment), Entorno de Desarrollo Integrado
WPM	(word per minute), Palabras por minuto
PPM	Pulsaciones por minuto
MHDL	Metodología Hardware Libre
PCB	Printed Circuit Board
CAD	(Computer-Aided Design), Diseños asistidos por computadora
PWM	(Pulse width modulation PWM), modulacion por ancho de pulso
ICSP	(In-Circuit Serial Programming) Programación serial en circuito
GFDL	(GNU Free Documentation License) Licencia de documentación libre de GNU
COPYLEFT	Permite la libre distribución de copias y versiones modificadas, exigiendo que los mismos derechos sean preservados en las versiones modificadas.

Elaboración: El Autor

Anexo L: Manual de Usuario

MANUAL DE USUARIO

El dispositivo para traducir lenguaje código morse a expresión sonora, para personas con discapacidad sensorial, basa sus secuencias para que el usuario realice todas sus tareas.

Encendido

- El usuario debe conectar el dispositivo a una fuente de energía, ya sea mediante un cable USB o conectarlo a una batería de alimentación superior a los 5 voltios.

- El dispositivo estará listo tras presentar el mensaje de bienvenida.

Apagado

- Basta con desconectar el dispositivo de la fuente de alimentación para que el dispositivo se apague.

Pulsador 1 (Morse)

- El pulsador 1 (Morse), obedece al tiempo durante el cual el pulsador está presionado. Por ejemplo: el dit o punto, está representado por una pulsación de 150 milisegundos; y, el dash o raya, está representada por una pulsación de alrededor de 450 milisegundos.

- El usuario puede referirse a las combinaciones descritas en la Tabla 43, correspondientes al Alfabeto Morse.

Ilustración 24: Pulsador 1 Morse

Elaboración: El Autor

- El espacio entre palabras, ocurre cuando no se persona el pulsador 1 (Morse) durante un aproximado de 1 segundo.

Tabla 43: Alfabeto Morse

A	.—	O	———	0	—————
B	—...	P	—..—	1	.————
C	—.—.	Q	——.—	2	..———
D	—..	R	.—.	3	...——
E	.	S	...	4	....—
F	..—.	T	—	5	.....
G	——.	U	..—	6	—....
H	....	V	...—	7	——...
I	..	W	.——	8	———..
J	.———	X	—..—	9	————.
K	—.—	Y	—.——	.	.—.—.—
L	.—..	Z	——..	,	——..——
M	——	?	..——..	:	———...
N	—.	!	..——.		

Elaboración: El Autor

Pulsador 2 (Retroceso)

- El Pulsador 2 (Retroceso), borra un carácter desde la posición actual hacia atrás, según sea presionado.

Ilustración 25: Pulsador 2 Retroceso

Elaboración: El Autor

Pulsador 3 (Hablar)

- El Pulsador 3 (Hablar), es el encargado de traducir lo escrito por el usuario a lenguaje audible, basta con presionarlo para escuchar lo escrito.

Ilustración 26: Pulsador 3 Hablar

Elaboración: El Autor

Anexo M: Capacitación a Usuarios

En este apartado se representa dos formas de aprendizaje del Código Morse, el aprendizaje por memoria para personas que padezcan de discapacidad visual, y el aprendizaje por sonido para personas con discapacidad auditiva.

De esta manera se aborda tanto la discapacidad visual-auditiva, permitiendo capacitar a este grupo de personas.

Aprendizaje por Memoria

El aprendizaje por memoria del Código Morse, requiere únicamente sesiones de práctica y ejercicio de periodos cortos de tiempo para alcanzar la memorización completa de todas las combinaciones de puntos (dit) y rayas (dah).

Por otro lado, también existen métodos que ayudan al aprendizaje, como es la Regla Nemotécnica y el Aprendizaje por Sonido.

Regla Nemotécnica

Este método consiste en asignar a cada letra una palabra clave determinada que comienza con la letra que se quiere recordar; luego, se sustituye cada vocal de la palabra clave por un punto o una raya, como se muestra en la Tabla 44, según las reglas descritas a continuación:

* La letra inicial de la palabra clave, es la letra correspondiente en Código Morse.

* El número de vocales que contiene la palabra clave, indica la longitud del Código Morse correspondiente; es decir: Si hay tres vocales, habrán 3 signos, sean rayas o puntos respectivamente.

* Si la vocal es una “O” se sustituye por una raya (-).

* Si se trata de cualquier otra vocal se coloca un punto (·).

Tabla 44: Regla Nemotécnica

A	asno	.–
B	bonaparte	–...
C	coca-cola	–.–.
D	docena	–..
E	el	.
F	faraona	..–.
G	góndola	––.
H	himalaya	....
I	isla	..
J	jabonoso	.–––
K	kolico	–.–
L	limonada	.–..
M	motor	––
N	nota	–.
O	otoño	–––
P	pisotones	.––.
Q	qocorico	––.–
R	ramona	.–.
S	sardina	...
T	tos	–
U	unico	..–
V	ventilador	...–
W	wigotón	.––
X	xochimilco	–..–
Y	yo-te-soplo	–.––
Z	zozobrada	––..

Elaboración: El Autor

Aprendizaje por sonido

Este aprendizaje se basa en la representación por sonidos, para las personas con discapacidad visual

En este método, como se expone en la Tabla 45, cada letra y cada número deben conocerse por su sonido mejor que por su aspecto. El código es un sistema de comunicación sonora similar a la palabra hablada. La letra A, por ejemplo, es un sonido corto y uno largo, cuya combinación suena dit-dah, y ha de considerarse y conocerse así, más no como "punto-raya". Esta regla se aplica para el resto del Código Morse.

Tabla 45: Aprendizaje por sonido

Letra	Sonido
A	dit-DAH
B	DAH-dit-dit-dit
C	DAH-dit-DAH-dit
D	DAH-dit-dit
E	dit
F	dit-dit-DAH-dit
G	DAH-DAH-dit
H	dit-dit-dit-dit
I	dit-dit
J	dit-DAH-DAH-DAH
K	DAH-dit-DAH
L	dit-DAH-dit-dit
M	DAH-DAH
N	DAH-dit
O	DAH-DAH-DAH
P	dit-DAH-DAH-dit
Q	DAH-DAH-dit-DAH
R	dit-DAH-dit
S	dit-dit-dit
T	DAH

U	dit-dit-DAH
V	dit-dit-dit-DAH
W	dit-DAH-DAH
X	DAH-dit-dit-DAH
Y	DAH-dit-DAH-DAH
Z	DAH-DAH-dit-dit
0	DAH-DAH-DAH-DAH-DAH
1	dit-DAH-DAH-DAH-DAH
2	dit-dit-DAH-DAH-DAH
3	dit-dit-dit-DAH-DAH
4	dit-dit-dit-dit-DAH
5	dit-dit-dit-dit-dit
6	DAH-dit-dit-dit-dit
7	DAH-DAH-dit-dit-dit
8	DAH-DAH-DAH-dit-dit
9	DAH-DAH-DAH-DAH-dit

Elaboración: El Autor

Anexo N: Manual de Mantenimiento

Este apartado permite al usuario o al personal técnico, realizar el mantenimiento del dispositivo, resolver posibles fallos, y brindar posibles soluciones para un correcto funcionamiento del dispositivo.

Ensamblado del Dispositivo

El usuario puede referirse a la Ilustración 14: Esquema del Dispositivo, para determinar el ensamblado del dispositivo.

Problemas Comunes

Comprobar lo siguiente de manera física y visual:

- Comprobar la alimentación sea a mayor a 5v y menor a 12v.
- Verificar que la pantalla LED enciende.
- Comprobar que los pulsadores estén en buen estado.

Resolución de fallos

Remplazo de Pulsadores

Los pulsadores son propensos a desgastarse por el uso. Es por ello que cuando se detecte que no funcionen correctamente se los sustituya de la siguiente manera:

* Con un destornillador de cruz, remover los tornillos y abrir la tapa del chasis.

* Con un cautín caliente, remover el estaño de los cables que están soldados al pulsador defectuoso.

* Cambiar el pulsador defectuoso por el nuevo, teniendo en cuenta la polaridad de los pines del pulsador.

* Con el cautín caliente, se sueldan los cables en el nuevo pulsador.

* Colocar la tapa, y atornillar.

Remplazo de Buzzer

En caso detectar una avería en el Buzzer, se procede a realizar lo siguiente.

* Con un destornillador de cruz, remover los tornillos y abrir la tapa del chasis.

* En la Placa Impresa del Circuito PCB, procedemos a retirar con la ayuda de un cautín caliente, la soldadura que une el Buzzer al PCB.

* Se remplaza el Buzzer, verificando la polaridad del mismo, y con la ayuda del cautín se coloca la nueva soldadura.

* Colocar la tapa, y atornillar.

Remplazo de Altavoz

En caso detectar una avería en el Altavoz, el procedimiento para remplazarlo es el siguiente:

* Con un destornillador de cruz, remover los tornillos y abrir la tapa del chasis.

* Retirar manualmente los cables que van desde la placa Emic 2 hacia el altavoz.

* En el nuevo altavoz, con ayuda de un cautín caliente, soldar los cables correspondientes tomando en cuenta su respectiva polaridad. Conectar el otro extremo de a la placa Emic2 siguiendo el esquema de la Ilustración 14.

* Colocar la tapa, y atornillar.

Remplazo de Pantalla LCD

Para reemplazar la pantalla LCD de 16x2 caracteres, se deberá cambiarla con una compatible que disponga del módulo i2c integrado y a continuación, se realiza lo siguiente:

* Con un destornillador de cruz, remover los tornillos y abrir la tapa del chasis.

* En la placa PCB, se desconecta el cable de alimentación, y luego se retiran los cables de la placa arduino.

* Conectar los cables según el esquema mostrado en la Ilustración 14 a la placa arduino, y conectar el cable de alimentación a la placa PCB.

* Colocar la tapa, y atornillar.

Remplazo del módulo Emic2 Text-to-Speech.

Para remplazar el módulo Emic2 Text-to-Speech, se realizan los siguientes pasos:

* Con un destornillador de cruz, remover los tornillos y abrir la tapa del chasis.

* Desconectar los cables del PCB, y los cables que van hacia el altavoz.

* Cambiar el módulo, remplazarlo; posteriormente conectar los cables en su respectivo lugar, tomando como referencia el diagrama mostrado en la Ilustración 14.

* Colocar la tapa, y atornillar.

Advertencias

* Para realizar mantenimiento, se desconecta el dispositivo de cualquier fuente de alimentación.

* Se utiliza únicamente paños suaves que no desprendan pelusa para la limpieza externa.

* Se debe mantener el dispositivo alejado de cualquier líquido para evitar daños.

* En el mantenimiento, no se debe emplear ningún tipo de aerosoles, disolventes ni sustancias liquidas.

* Evitar rayar la pantalla.