Universidad de la República
Facultad de Ingeniería

Bandonberry: simulador electrónico de bandoneón

Tesis presentada a la Facultad de Ingeniería de la Universidad de la República por

José Bentancour, Franco Toscano, Rodrigo Patiño

en cumplimiento parcial de los requerimientos para la obtención del título de Ingeniero Electricista.

Tutores

Mag. Ignacio Irigaray ..... Universidad de la República
Ing. Pablo Zinemanas ..... Universidad de la República

Tribunal

Dr. Ing. Martín Rocamora ..... Universidad de la República
Dr. Ing. Leonardo Stenfield ..... Universidad de la República
Ing. Pedro Arzuaga ..... Universidad de la República

Montevideo
miércoles 16 octubre, 2019

Bandonberry: simulador electrónico de bandoneón, José Bentancour, Franco Toscano, Rodrigo Patiño.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 137 páginas.
Compilada el miércoles 16 octubre, 2019.
http://iie.fing.edu.uy/

Agradecimientos

Al Grupo de Procesamiento de Audio por su ayuda en las etapas iniciales y en particular a nuestros tutores Ignacio Irigaray y Pablo Zinemanas que nos ayudaron y guiaron durante todo el proyecto.

A Martín Pugín por darnos su tiempo y entusiasmo, permitiéndonos tener sugerencias y observaciones desde la perspectiva de un bandoneonista, a lo largo del desarrollo de los prototipos.

A Daniel Villamil por sugerirnos ideas con los problemas estructurales y mecánicos del modelo 3D. En particular nos planteó soluciones para el diseño del mecanismo de los botones, mostrándonos diferentes métodos de anclaje para las PCBs en carcasas de plástico y nos brindó muestras de anclajes herméticos para cables.

Al Instituto de Ingeniera Eléctrica que nos abrió sus puertas y nos permitió utilizar herramientas indispensables para el desarrollo del proyecto, como por ejemplo la prototipadora de PCBs e impresora 3D.

A Matteo Chiancone, por ayudarnos en la gestión y seguimiento de las importaciones de varios insumos y de una impresora 3D.

A nuestras familias y nuestras parejas, por acompañarnos en todo el proceso,

Esta página ha sido intencionalmente dejada en blanco.

Resumen

El proyecto Bandonberry tiene como objetivo crear un simulador electrónico de bandoneón. Pretende ser una alternativa económica a un bandoneón tradicional, de forma de facilitar el acceso al instrumento a los estudiantes. No solo se limita a reproducir su forma y sonido, sino que incorpora tecnologías que ayudan y estimulan el proceso de aprendizaje.

En este documento se describe el diseño de un simulador electrónico de bandoneón. Se detalla todo el proceso de diseño, fabricación, criterios de selección de componentes y consideraciones para trabajos futuros. Por otro lado, se hace un estudio del sistema finalizado, comparando los resultados obtenidos con los requerimientos planteados.

Algunos de los requerimientos están asociados a la reproducción fidedigna del sonido característico del instrumento y la forma en la que es tocado. En particular, es necesario mitigar las latencias inherentes al sistema, impactando en decisiones de diseño en el área de software, como la electrónica y los métodos de detección de señales. Se diseña el sistema para que sea autónomo en términos de alimentación eléctrica, por lo que se contemplan temas asociados a baterías y consumos de energía. Para la simulación de la mecánica y dinámica del instrumento se hace uso de un sensor de presión y un servomotor. Para la fabricación del instrumento se abordan conceptos de modelado e impresión 3D.

Esta página ha sido intencionalmente dejada en blanco.

Glosario

GPIO:	Del inglés *General Purpose Input/Output*, entrada/salida de propósito general.
PCB:	Del inglés *Printed Circuit Board*, placa de circuito impreso.
USB:	Del inglés *Universal Serial Bus*, bus serial universal.
USB OTG:	Del inglés *USB On The Go*, extensión de la norma USB que permite a los dispositivos intercambiar roles.
UNESCO:	Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura.
BMS:	Del inglés *Battery Management System*, sistema de gestión de baterías.
LiPo:	Polímero de Litio.
SBC:	Del inglés *Single Board Computer*, computadora de placa única.
PC:	Del inglés *Personal Computer*, computadora personal.
MIDI:	Del inglés *Musical Instrument Digital Interface*, interfaz digital para instrumentos musicales.
SPI:	Del inglés *Serial Peripheral Interface*, interfaz de periférico serie.
I2C:	Del inglés *Inter-Integrated Circuit*, bus serie de datos.
PWM:	Del inglés *Pulse Width Modulation*, modulación por ancho de pulsos.
SoC:	Del inglés *State of Charge*, estado de carga (asociado a una batería).
PLA:	Poliácido láctico, tipo de plástico utilizado para impresión 3D.
ABS:	Del inglés *Acrylonitrile Butadiene Styrene*, tipo de plástico utilizado para impresión 3D.
MDF:	Del inglés *Medium Density Fibreboard*, compensado de madera.
IDE:	Del inglés *Integrated Development Environment*, entorno de desarrollo integrado.
I2S:	Del inglés *Inter IC Sound*, interfaz digital para interconectar dispositivos de audio.

MIT:	Massachusetts Institute of Technology, Instituto de Tecnología de Massachusetts.
LED:	Del inglés *Light-Emitting Diode*, diodo emisor de luz.
SSH:	Del inglés *Secure SHell*, Protocolo de red para el intercambio de datos de manera segura (cifrado) entre dispositivos.
CPU:	Del inglés *Central Processing Unit*, unidad de procesamiento central.
RAM:	Del inglés *Random Access Memory*, memoria de acceso aleatorio.
IIR:	Del inglés *Infinite Impulse Response*, filtro de respuesta al impulso infinita.
DMA:	Del inglés *Direct memory access*, memoria de acceso directo.
OLED:	Del inglés Organic Light Emitting Diode, diodo emisor de luz orgánico.
PLX-DAQ:	Parallax Data Acquisition tool- Macro para Excel utilizado para adquirir datos de una placa Arduino.

Tabla de contenidos

Tabla de contenidos

Tabla de contenidos

Capítulo 1

Introducción

En este capítulo se explica qué es el bandoneón, su importancia en la región y en la identidad ríoplatense, los objetivos y el alcance del proyecto en términos generales.

1.1. Motivación

El bandoneón pertenece al grupo de instrumentos denominados de lengüeta libre. Los bandoneones se fabricaron de forma industrial hasta la década del 1960. Actualmente, son fabricados de forma artesanal por algunos pocos luthiers. Consecuentemente, el costo de un bandoneón estándar es alto, aproximadamente unos 3.000 USD. Esto hace que sea difícil adquirir el instrumento, especialmente para quienes recién se inician, lo que limita la transmisión de la práctica a las nuevas generaciones.

Como ejemplo, se puede citar el caso de la Escuela Municipal de Música Vicente Ascone de la ciudad de Montevideo, que al momento del inicio del proyecto contaba con un solo bandoneón para 20 estudiantes que participan de la cátedra. Se han recabado testimonios que ilustran esta difícil situación que viven los alumnos, los cuales deben imprimir en papel el mapa de las botoneras para practicar los movimientos.

Lo que motiva al proyecto Bandonberry es facilitar el acceso al instrumento. El objetivo primordial es construir un instrumento de práctica, para darle una opción accesible a los interesados en aprender a tocar el bandoneón. Si bien no se busca generar un sonido de alta calidad, sí deberá ser suficiente como para poder aprender y dominar el instrumento. Para su construcción, se recurre a la electrónica y a la impresión 3D. De esta manera, se pretende lograr costos considerablemente menores respecto a un bandoneón convencional. Por otro lado, se podrán incorporar tecnologías que motiven al estudiante en el proceso de aprendizaje. Por ejemplo, poder seguir una partitura en tiempo real en un PC.

Cabe señalar que el bandoneón, siendo el instrumento solista principal y característico del tango como género musical, es una figura muy importante para nuestra identidad popular. Es así que este proyecto, pretende contribuir a la salvaguardia

Figura 1.1: El bandoneón.

de un símbolo del patrimonio cultural uruguayo, así como fomentar la práctica del bandoneón en su contexto tanguero, con todo lo que eso significa: música, danza y poesía. Se debe tener en cuenta, que al día de hoy el tango como danza y genero musical, trasciende fronteras.

Por último, se señala que el diseño es de carácter libre, es decir, los modelos 3D, los esquemáticos, la documentación y el *software*, serán públicos, buscando contribuir al libre acceso [1]. Se busca liberar todo el conocimiento generado en el proyecto para que, de alguna manera, en la comunidad de personas interesadas, surjan nuevas ideas, mejoras, cambios, que aporten al instrumento así como también motiven el aprendizaje de la ingeniería y la música de forma conjunta. También se espera que con la popularización de las impresoras 3D, la disminución de sus precios y el mejor acceso a la electrónica, se logre que la construcción y tenencia del instrumento sea cada vez más fácil. Por este motivo, se elige la licencia MIT, la cual permite usar, copiar y modificar el software. El hardware se define como libre.

1.2. El Bandoneón

1.2.1. Origen e historia

El bandoneón es un instrumento musical de origen alemán, se le atribuye su invención a Heinrich Band (1821-1860). Se creó con el objetivo de tener un instrumento más económico, portable y que tuviera un sonido similar al de un órgano de iglesia. Dichos objetivos buscaban crear un nuevo instrumento para acompañar las ceremonias cristiano-evangélicas. La paradoja es que habiendo sido creado para alegrar y animar los rituales evangélicos no tuvo mucho éxito, pero sí logró popularizarse y ser el alma de la música en los prostíbulos y bares del arrabal del Río de la Plata [30].

[1] Disponible en el repositorio https://github.com/jebentancour/Bandonberry.

Tanto en Uruguay como en Argentina, fundamentalmente en la región del Río de la Plata, el bandoneón tiene un impacto cultural muy fuerte en la cultura que se genera con las corrientes migratorias europeas del siglo XX. En primera instancia el bandoneón era un instrumento relacionado con lugares de bajo prestigio socio-económico. El tango se populariza en el contexto de burdeles, bares y lugares marginales, teniendo el bandoneón como su máximo protagonista musical [7].

En el lunfardo [2] rioplatense se lo denomina "fuelle" o también escrito como "fueye" a todo el instrumento. Por otro lado la palabra fuelle también quiere decir pulmón o sistema respiratorio en general, por lo que es muy común que en el lunfardo, el bandoneón se use con palabras de afecciones respiratorias. Como dice el famoso tango "Garganta con Arena"[3]: "... con el asma de un viejo bandoneón...", haciendo alusión a que el instrumento al ser viejo no tiene un sonido tan limpio, como el chillido respiratorio de un asmático.

La etimología del nombre "bandoneón", no tiene un origen bien definido. Hay varias teorías planteadas, por un lado se habla de que una de las primeras marcas era "Band Union" y de alguna manera eso se transforma en la palabra "bandoneón". Otra teoría menciona que el supuesto creador del instrumento (o al menos, el primero en comercializarlo), como se mencionó anteriormente, era de apellido "Band" y dado que se basó en el sistema del "akkordion" (acordeón en alemán), definió su nombre como "Band-o-nion" [2]. Esta última teoría sirve de inspiración para el nombre del proyecto, ya que se basa en la invención de "Band" y en la conocida placa programable Raspberry Pi [4], generando el nombre "Bandonberry".

1.2.2. Contexto y tango

El tango es una de las expresiones artísticas más importantes del siglo XX. En el año 2009 fue declarado Patrimonio Cultural Inmaterial de la Humanidad por la UNESCO [26]. Dentro de los instrumentos utilizados en el tango, el bandoneón es quizás el que más define su sonido.

Como se mencionó antes, en el Río de la Plata en el siglo XIX y prinicipios del XX, el tango era uno de los géneros más importantes en el ámbito popular. Como muchos géneros musicales, trajo una danza asociada. En dicho momento y región, era el baile utilizado por las prostitutas para seducir a sus eventuales clientes. Esto llevó a que el rol de la mujer en el tango, estuviera cargado de mucha sensualidad en sus movimientos, llamándose "tango del arrabal". En ese entonces, la alta sociedad de Montevideo rechazaba la idea de ese tipo de danza. Luego comenzó a aparecer el "tango de salón", influenciado por las culturas europeas y norte-americanas. Este último, mantenía ciertos elementos del "tango del arrabal" pero con más

[2] El lunfardo es un conjunto de palabras originadas en el Río de la plata caracterizadas por el choque de culturas que se dio en siglo XIX y XX, muy utilizada en el tango y todo su contexto.

[3] Tango de Cacho Castaña del álbum "Soy un tango" publicado en 1994.

[4] *Raspberry Pi Foundation* es una organización benéfica con sede en el Reino Unido que trabaja para democratizar la informática y la creación digital en todo el mundo (`https://www.raspberrypi.org/`).

"rigidez", buscando una mayor elegancia y una sensualidad menos explícita. Esto, sumado todos los cambios sociales del siglo XX, logró que poco a poco, el tango se comenzara a aceptar con todas sus variaciones por gran parte de la sociedad [7].

Por otro lado, cabe mencionar que la palabra "tango" engloba tres tipos de géneros: tango, milonga y vals. Siendo los tres géneros muy particulares y diferentes, pero todos relacionados por compartir el bandoneón y los mismos instrumentos de orquesta. A su vez, los tres géneros están vinculados por su forma de bailar, ya que comparten movimientos similares. Otra particularidad de las palabras asociadas al tango, es que el vocablo "milonga", hace referencia tanto al género musical como al evento y/o lugar físico donde se bailan los tres géneros antes mencionados. En las milongas (lugares físicos), se escuchan y bailan los tres géneros intercalados a lo largo del evento. El contexto musical del bandoneón no se limita al tango, milonga y vals, sino que también esta presente en otros géneros musicales [2].

Con respecto a la fabricación y mantenimiento de bandoneones, hay una problemática seria, que compromete considerablemente el futuro del instrumento. Encontrar personas que sean idóneas en el oficio de afinador, es excepcional al día de hoy. Son muy pocos los luthiers que hacen ese tipo de trabajos, y a su vez es un oficio que esta en proceso de desaparición, ya que no hay escuela de afinadores. Por otro lado, la fabricación de bandoneones a nivel industrial nunca tuvo lugar en la región del Río de la Plata. Los bandoneones que se consideran de buena calidad son los "doble A" fabricados por Alfred Arnold en Alemania, los cuales ya no se producen más [7].

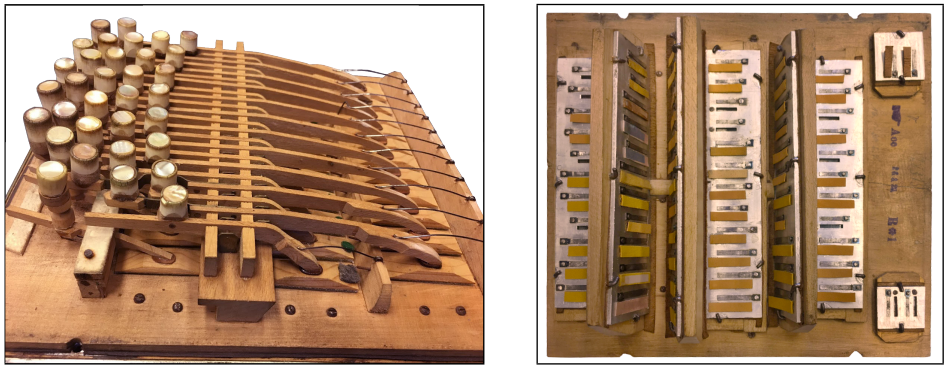
1.2.3. Descripción física

El bandoneón consiste de lengüetas metálicas conectadas por canales a un fuelle hermético. A su vez dichos canales poseen un mecanismo de cerrado y apertura a través de botones. En la Figura 1.2 se observa un bandoneón con las tapas desensambladas, en el centro se encuentra el fuelle y a cada lado los sistemas de botones y válvulas con sus respectivas tapas. En la Figura 1.3 se muestran en mayor detalle el mecanismo de botones (1.3a) y las lengüetas (1.3b) de una de las botoneras.

El bandoneón posee 71 botones en total, donde cada botón tiene asociada dos notas. Se produce una nota cuando el fuelle se comprime (cuando el flujo de aire es saliente) y otra nota cuando el fuelle se expande (cuando el flujo de aire es entrante). Los 71 botones, están divididos en dos partes, llamadas botoneras. Una botonera es controlada por la mano derecha y la otra por la mano izquierda de quien ejecuta el instrumento. De los 71 botones, 38 son para las notas agudas o consideradas para ser "la primera voz", estando en la botonera derecha. Los 33 botones restantes se encuentran en la botonera izquierda y son los graves, que para cumplen la función de acompañamiento. La disposición de los botones en términos espaciales y musicales, fue aumentando su complejidad y variando a lo largo de su evolución, hasta llegar a la disposición actual utilizada en el tango, llamada Rheinische Tonlage 38/33 [18].

El sonido se genera con la coordinación de la fuerza que se ejerce sobre el

Figura 1.2: Bandoneón desarmado.



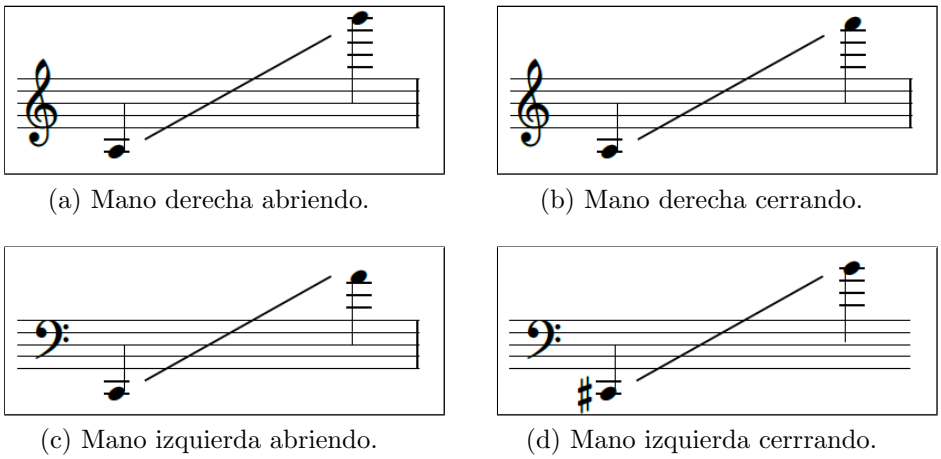
(a) Botones. (b) Lengüetas.

Figura 1.3: Sistema de botones y lengüetas de la mano izquierda.

fuelle y la manipulación de los botones. En términos físicos, el sonido lo produce la vibración de las lengüetas al ser excitadas por el flujo de aire que las atraviesa. El registro del bandoneón, es decir el rango de notas posibles de generar con el instrumento, es el mostrado en la Figura 1.4. La distribución de las notas fue pensada para facilitar el ejercicio de música religiosa evangélicas, como se mencionó anteriormente. Por este motivo se diferenciade las lógicas de distribución de las notas de la gran mayoría de los instrumentos. Sumado con la complejidad de que cada botón tiene dos posibles notas asociadas, dependiendo del movimiento del fuelle, hace que sea un instrumento un aprendizaje difícil.

El bandoneón cuenta con una palanca en la botonera derecha, la cual abre una escotilla, que permite expandir o contraer todo el fuelle rápidamente sin generar sonido. En la Figura 1.5 se observa el sistema de polea que acciona la escotilla de toma de aire.

Existe una variedad de instrumentos similares en su timbre, forma y lógica de funcionamiento. Algunos de ellos son: acordeón diatónico, acordeón cromático de teclas, acordeón cromático de botones, concertina, entre otros.



(a) Mano derecha abriendo. (b) Mano derecha cerrando.

(c) Mano izquierda abriendo. (d) Mano izquierda cerrrando.

Figura 1.4: Pentagrama que muestra el registro del bandoneón.

Figura 1.5: Sistema de polea que acciona la escotilla de liberación aire.

La mecánica del instrumento es compleja, y por ende, el trabajo que requiere su construcción, lo convierte en una artesanía costosa y sofisticada. Por otro lado, la afinación del bandoneón es una tarea sumamente compleja y delicada, ya que lo que da el tono de cada nota depende del tamaño de la lengüeta. Para poder afinarlo se debe limar las lengüetas cuidadosamente hasta encontrar el punto exacto.

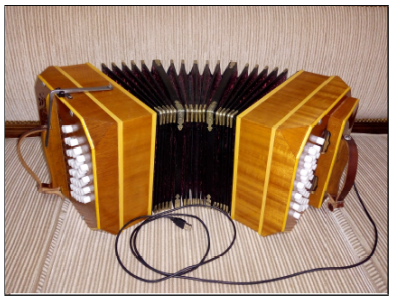
1.3. Antecedentes

Una iniciativa similar, en el sentido de hacer el bandoneón más accesible en términos económicos, es el proyecto Pichuco [21], realizado en Argentina. Busca

utilizar tecnologías de producción modernas para fabricar bandoneones de manera industrializada. Dicho proyecto comenzó en 2009 en la Universidad de Lanus y llegaron a fabricarse prototipos en 2013. Sin embargo, hasta el momento no se tiene información de que se hayan producidos en gran escala (ver Figura 1.6a).

Por otro lado, hay referencias de otro proyecto que se fue desarrollado paralelamente al Bandonberry por el argentino Federico Fraysee. El proyecto se describe en el documento "Bandoneón electrónico" [11]. El planteo es similar al del Bandonberry en su objetivo primordial, pero difiere en otros. Fraysee plantea construir y diseñar un dispositivo que emule al bandoneón con fines didácticos y de bajo costo, buscando que sea más accesible. La diferencia con respecto al proyecto Bandonberry, es que este bandoneón argentino tiene una terminación más definida en su presentación estética y una excelente simulación de la mecánica de los botones. Por otra parte, no posee ni baterías ni parlantes integrados, como si los tiene el Bandonberry, brindándole independencia al momento de su ejecución. La propuesta de Fraysee sólo posibilita que la ejecución se realice conectando el instrumento a una fuente de alimentación y a un sintetizador externo. Ver Figura 1.6b.

(a) Bandoneón Pichuco, extraída de [3]



(b) Bandoneón Electrónico, extraída de [11]

Figura 1.6: Bandoneones similares de proyectos anteriores.

1.4. Objetivo

Se busca diseñar y construir un simulador de un bandoneón utilizando tecnología actuales de bajo costo. El diseño busca mejorar la accesibilidad económica, así como poseer fidelidad en términos de simulación de un bandoneón. Se busca, crear un simulador de bandoneón para estudiantes, sin la pretensión de sustituir el instrumento original.

Se pretende un diseño, que sea reproducible de forma relativamente sencilla, con electrónica básica y con el uso de impresión 3D. Parte del objetivo es que la mayor cantidad de personas tenga acceso al instrumento. Por lo anterior, todos los diseños se publicarán de forma libre, logrando dos cosas importantes:

- Democratizar y dar la oportunidad de que cualquier persona, de cualquier parte del mundo, pueda tener un bandoneón de forma más accesible.

- Permitir y aceptar modificaciones que contribuyan a mejorar el proyecto. Por ejemplo, cambios en los diseños 3D, mejoras en la electrónica, programas alternativos, entre otros.

El actor con más importante para el proyecto es el usuario final, en este caso el estudiante de música que utilizará el Bandonberry. Parte del objetivo es también diseñar el instrumento de la forma más amigable y útil para el usuario.

1.5. Alcance

En términos generales, el alcance del proyecto incluye el diseño y construcción de un prototipo que logre simular la forma, la dinámica y el sonido de un bandoneón. La dinámica, hace referencia a los movimientos posibles de un fuelle convencional, así como la resistencia al cambio de volumen que genera el aire dentro del mismo, en concordancia con el pulsado de los botones. A continuación se enumeran las características esperadas del Bandonberry:

1. Poseer los 71 botones (de aspecto similar y con la misma disposición) del bandoneón que se tuvo como referencia, mostrado en la Figura 1.1.
2. Ser una reproducción fiel del bandoneón, respetando todas las dimensiones y aspecto exterior.
3. Incorporar parlantes para escuchar las notas sintetizadas.
4. Ser portable, contando con baterías incorporadas y sin necesidad de dispositivos adicionales para funcionar.
5. Contar con interfaz USB[5] que permita enviar las notas tocadas a una PC. Debe ser reconocido por la misma como un dispositivo MIDI[6] y enviar los eventos correspondientes al presionar los botones y accionar el fuelle.
6. La documentación debe ser pública y libre, de manera que pueda ser replicado, modificado y mejorado por otras personas.
7. Poseer una latencia menor a los 200 ms entre el momento de oprimir un botón cualquiera y la reproducción del sonido correspondiente.
8. Debe tener un costo total en materiales menor a los 12.000 pesos uruguayos.
9. La estructura física debe soportar el estrés mecánico de al menos 30 minutos de uso normal por día, a lo largo de una semana.

[5]Del inglés *Universal Serial Bus*, bus serial universal.

[6]Del inglés *Musical Instrument Digital Interface*, interfaz digital para instrumentos musicales.

Capítulo 2

Descripción general y arquitectura

En el presente capítulo se presenta el Bandonberry, su forma física, y sus componentes de *hardware* y *software*.

2.1. Descripción general

El Bandonberry tiene la mismas dimensiones y aspecto que un bandoneón estándar, tal como se puede ver en la Figura 2.1 [1]. El material utilizado para construir las tapas, botones y soportes es PLA [2]. El fuelle está hecho de cartulina y *contact* plegados de manera que forman una estructura hermética y flexible. El mismo, tiene el hermetismo suficiente para simular la resistencia hidráulica del fuelle de un bandoneón.

Figura 2.1: El Bandonberry.

[1] Las medidas del bandoneón de referencia se pueden encontrar en `https://github.com/jebentancour/Bandonberry/tree/master/stl/FotosConMedidas`

[2] Poliácido láctico, tipo de plástico utilizado para impresión 3D.

En la Figura 2.2 se presenta un diagrama general del sistema. Más a delante se explicarán en detalle la construcción y funcionamiento de los diferentes bloques.

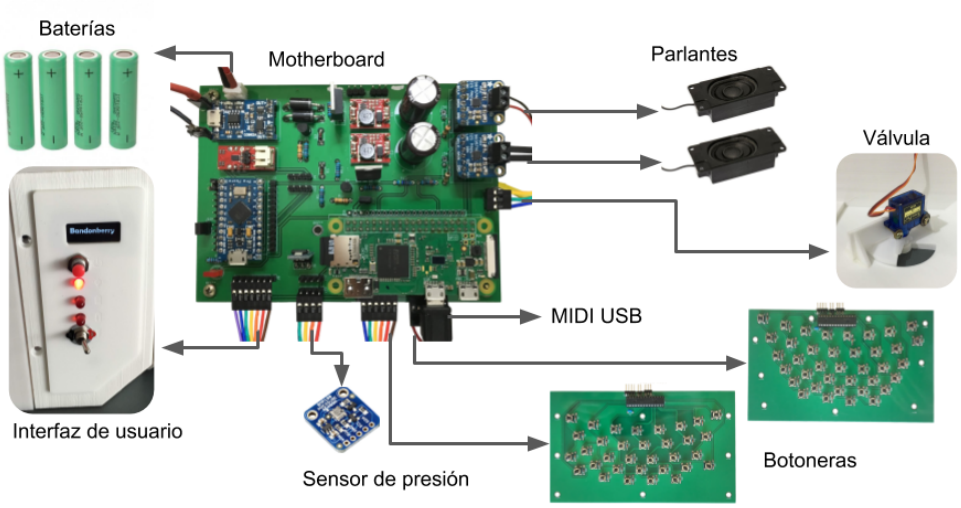


Figura 2.2: Diagrama general del sistema.

Cada nota está asociada a la posición del botón y al estado dinámico del fuelle (cerrándose o abriéndose). Los botones, tienen una parte mecánica a la que se denomina botón, y una parte eléctrica a la que se denomina pulsador (ver Figura 8.4). Al presionarse cada botón, la parte mecánica trasmite la presión al pulsador. Estos últimos, están montados en unas PCBs [3] (una para cada lado, derecha e izquierda) denominadas botoneras. Los pulsadores son de estado binario, por lo que solo se tienen los estados de "presionado" o "no presionado". La información sobre el estado de los botones, se envía a la SBC [4] interconectada a través de la placa *motherboard*. En la SBC, se tiene un sintetizador donde se genera la señal de sonido, la cual es enviada a unas placas amplificadoras, para finalmente llegar a los parlantes. En la Figura 2.3 se tiene la tapa izquierda del Bandonberry abierta, donde se pueden observar algunos de los componentes mencionados. Por otro lado, se puede apreciar la batería, los conectores externos y la interfaz de usuario.

En los conectores externos, se tiene una entrada de alimentación de 5 V para una fuente externa y un conector micro USB con el que se puede conectar a una PC, como se observa en la Figura 2.4.

La dinámica del fuelle se mide con un sensor de presión. En función de dichas mediciones se modula el volumen del sonido, es decir a mayor presión, mayor volumen y viceversa. El fuelle conforma en su interior una cámara cerrada y vacía, donde solo se tiene el sensor de presión adherido a la pared izquierda y los cables que conectan ambas botoneras, ver Figura 2.5.

[3] Del inglés *Printed Circuit Board*, placa de circuito impreso.
[4] Del inglés *Single Board Computer*, computadora de placa única.

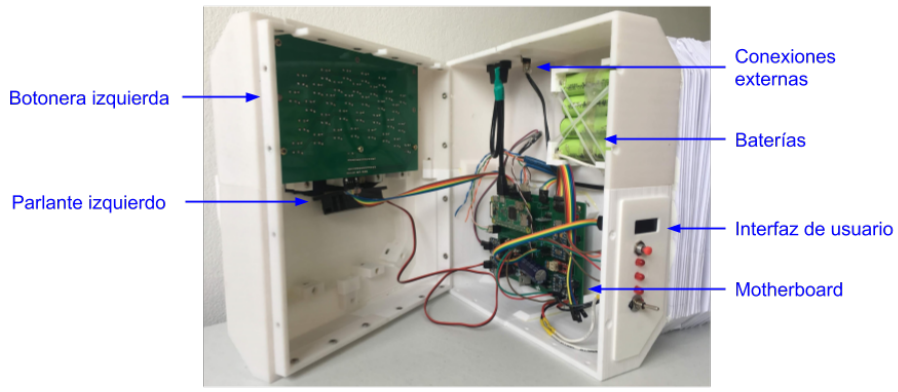


Figura 2.3: Interior de la botonera izquierda.

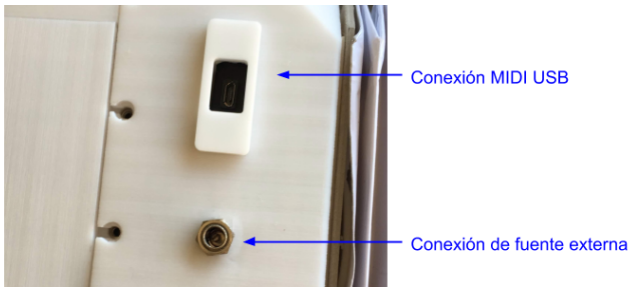


Figura 2.4: Puertos de conexión en la botonera izquierda.

En la Figura 2.6 se muestra el interior de la tapa derecha, donde se encuentran el servomotor, la botonera derecha y uno de los parlantes.

Al presionar uno o más botones en el bandoneón, la resistencia hidráulica del fuelle disminuye, ya que se abren canales donde el aire se escapa. En el Bandonberry, se simula este efecto, con un servomotor que abre y cierra una válvula dependiendo del estado de los botones. Cuando un botón se presiona, el servomotor abre la válvula bajando la resistencia hidráulica, y de manera inversa al soltar el botón. En la Figura 2.7 se muestra la válvula abierta y cerrada, accionada por el servomotor.

En el bandoneón se tiene una palanca que abre una escotilla de forma tal que baja considerablemente la resistencia hidráulica, para poder contraer o expandir el fuelle de forma rápida. En el Bandonberry, eso se logra de forma muy similar, se tiene un escotilla mecánica que se abre y se cierra en función de una palanca. En la Figura 8.1 se puede ver el funcionamiento de la escotilla en sus dos estados. La misma se encuentra en la tapa derecha tal como se puede apreciar en la Figura 2.6.

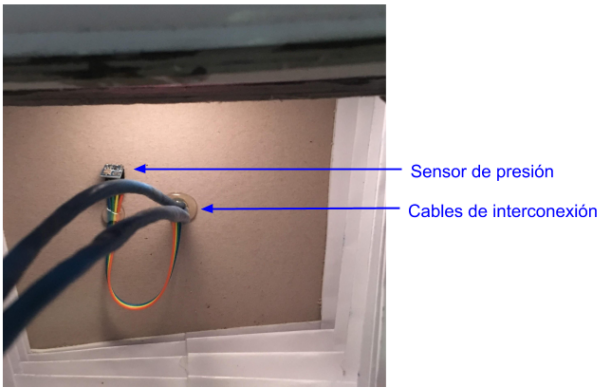


Figura 2.5: Interior del fuelle.

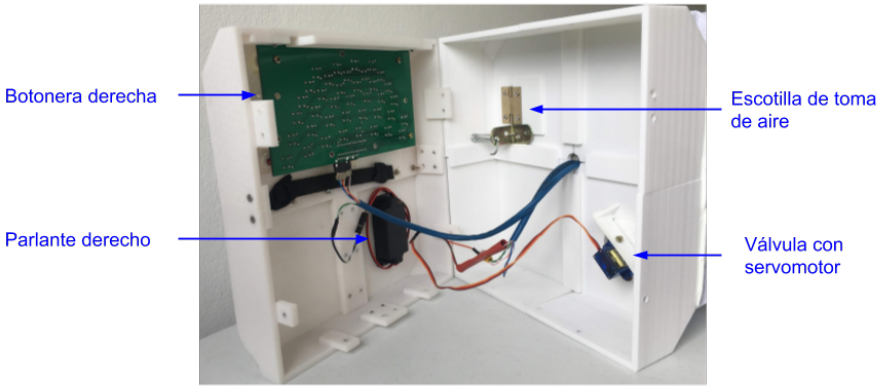
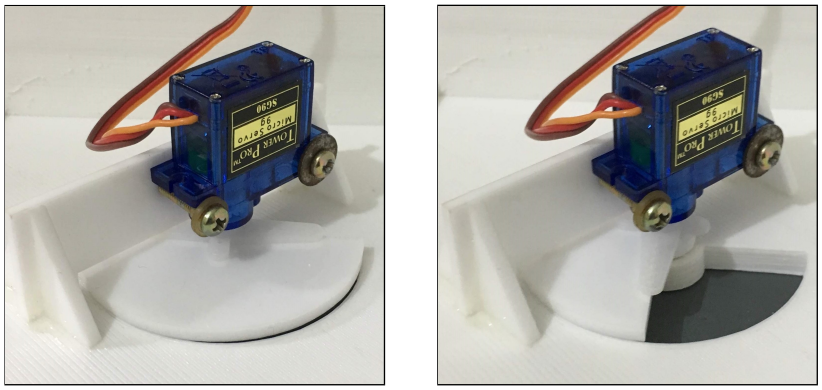


Figura 2.6: Interior de la botonera derecha.

2.2. Arquitectura de Hardware

La arquitectura de *hardware* del sistema se puede apreciar en el diagrama de bloques de la Figura 2.9. A continuación se presenta una breve descripción de los bloques y en los capítulos posteriores se encuentra una descripción detallada de su diseño e implementación.

Para gestionar la alimentación y brindar autonomía se cuenta con un BMS (*Battery Management System*). El mismo se subdivide en cuatro módulos: Baterías, Selector, Microcontrolador e Interfaz de usuario. Las baterías son de polímero de litio (LiPo), formadas por cuatro celdas de 3.6 V nominales en paralelo. El módulo Selector gestiona la entrada y distribución de la energía en el sistema. Posee entrada para una fuente externa de 5 V, que es capaz de cargar las baterías y generar los distintos niveles de tensión necesarios para alimentar el resto del sistema. Además, el Selector cuenta con las protecciones contra sobrecargas, sobredescargas y cortocircuitos asociadas a las baterías.



(a) Cerrado. (b) Abierto.

Figura 2.7: Sistema de válvula con servomotor.

(a) Cerrada. (b) Abierta.

Figura 2.8: Sistema de palanca y escotilla de liberación de aire.

El Microcontrolador, del tipo Arduino [5], es el encargado de gestionar el Selector. Toma decisiones sobre qué partes del sistema deben recibir energía, asegurando la correcta secuencia de encendido y apagado. Esta tarea es clave para asegurar la integridad de la SBC.

Por último, dentro del BMS, se tiene una interfaz de usuario, la cual posee una pantalla donde se brinda información del estado de la batería y de la SBC. Además, en dicha interfaz se tiene un botón que permite el encendido y apagado del Bandonberry.

Para sintetizar el sonido y leer los datos de entrada del fuelle y botoneras se utiliza una SBC del tipo Raspberry Pi. Ella corren los diferentes programas desarrollados junto con un sintetizador. Este último genera las señales necesarias y las envía al bloque de audio de forma digital. El bloque de audio transforma las señales digitales en señales analógicas de potencia, para poder excitar los parlantes. Por otro lado, se encuentra una interfaz USB la cual permite conectar una PC al Bandonberry y establecer una comunicación por mensajes MIDI, lo que permite utilizar aplicaciones que interactúen con el Bandonberry o utilizar un sintetizador

[5]Placas con microcontroladores ATmega de *software* y *hardware* abierto (`https://www.arduino.cc/`).

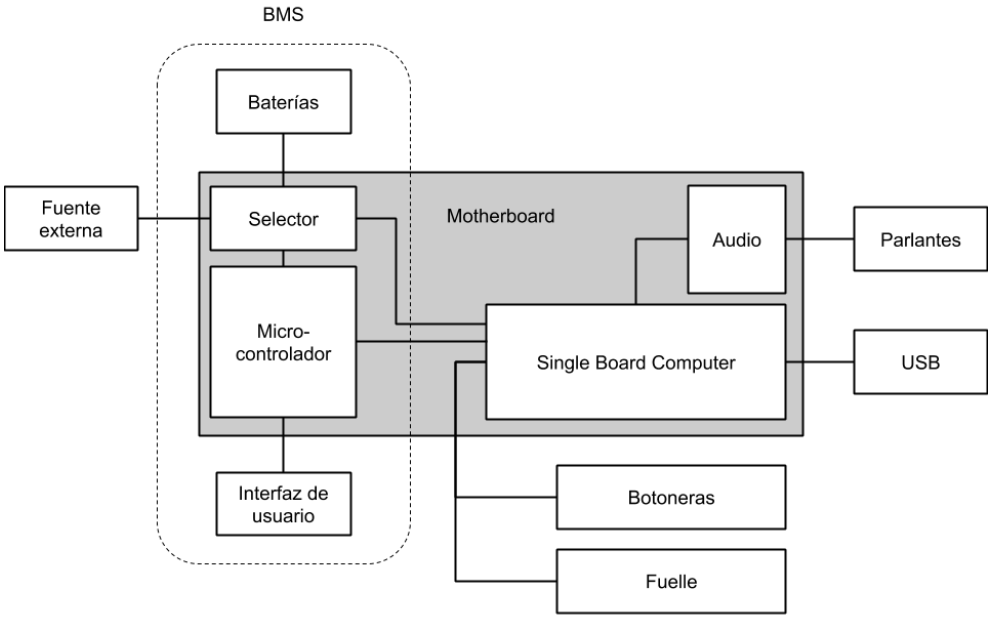


Figura 2.9: Esquema general de *hardware*.

externo (distinto al provisto en la Raspberry).

Las botoneras poseen la misma disposición y cantidad de botones que un bandoneón tradicional. Son capaces de detectar el accionar de distintos botones de manera simultánea. Las mismas están conectadas a la SBC, que asocia cada botón a una nota específica, dependiendo de la dirección del movimiento del fuelle.

El fuelle es de construcción similar a uno convencional, proporcionando una resistencia al movimiento que depende de la cantidad de botones presionados, que se logra con la utilización de la válvula con servomotor. Para la detección del movimiento del fuelle se utiliza un sensor de presión, el cual es capaz de detectar el sentido del movimiento y la fuerza que se ejerce en el mismo.

La Motherboard es una placa de circuito impreso que interconecta todos los otros bloques y contiene todos los circuitos de soporte para el correcto funcionamiento del Bandonberry.

2.3. Arquitectura de Software

La arquitectura de *software* del sistema se muestra en el diagrama de bloques de la Figura 2.10. Todos los programas se desarrollaron en Python, dado que es el entorno natural de la Raspberry Pi y por lo tanto existe gran cantidad de documentación, librerías y soporte para este lenguaje.

A continuación se presenta una breve descripción de cada bloque y en los

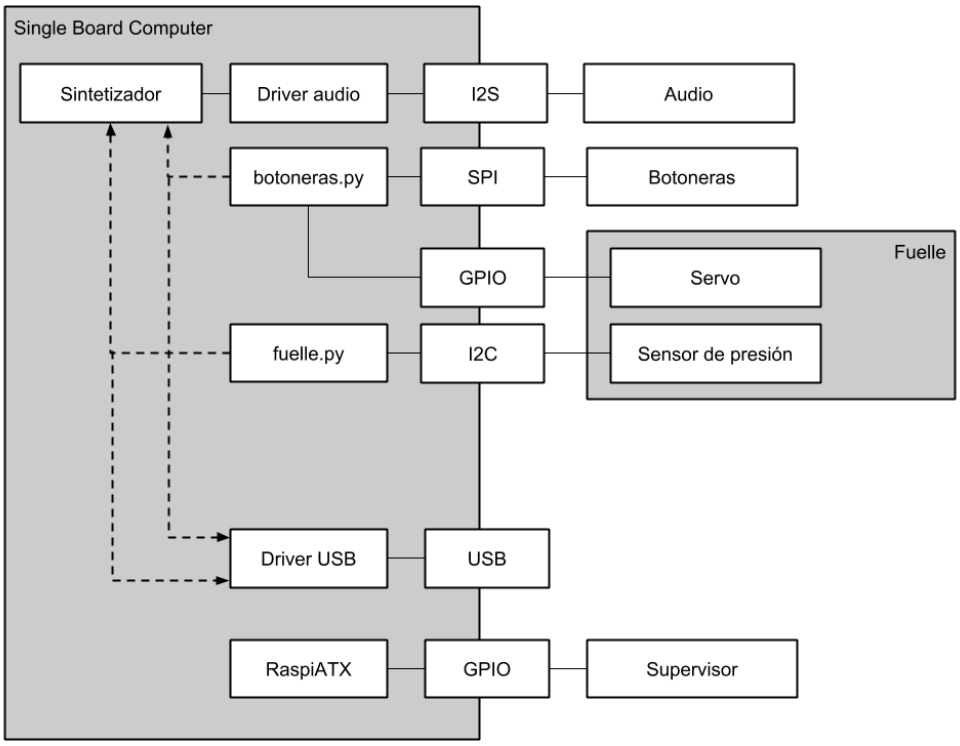


Figura 2.10: Esquema general de *software*.

capítulos posteriores se encuentra una descripción detallada de su funcionamiento.

2.3.1. Sistema operativo

Como sistema operativo se eligió Raspbian[6], en su versión Raspbian Stretch Lite, la cual es ligera y no posee entorno gráfico. Se eligió dicha opción de Raspbian, para liberar la mayor cantidad de recursos de la Raspberry Pi. Se recuerda que es necesario tener latencias de respuesta muy bajas, por lo que los tiempos de procesamiento deben ser lo más cortos posibles. Por otro lado, el sistema operativo trae funciones de comunicación de red por WiFi. Esto permite controlar la Raspberry de forma inalámbrica.

2.3.2. Sintetizador

El Sintetizador es el programa encargado de interpretar mensajes MIDI y convertirlos en el sonido correspondiente, en este caso en particular se eligió FluidSynth[7]. Cabe mencionar que hay otras opciones válidas para usar de sintetizador como por

[6]Distribución de Linux oficial de la Fundación Raspberry Pi, basada en Debian.
[7]Disponible en `http://www.fluidsynth.org/`.

ejemplo: TiMidity++[8], Yoshimi[9] o ZynAddSubFX[10].

Por otro lado, el sintetizador es utilizado en conjunto con *SoundFonts*[11] de bandoneón para crear el sonido de las diferentes notas. Las *SoundFonts* consisten básicamente en muestras de sonido de cada nota posible, de un instrumento determinado. Muchos sintetizadores utilizan dichas *SoundFonts*, las cuales son reproducidas el tiempo que sea necesario para lograr emular al instrumento en cuestión.

Cabe mencionar, que muchas veces son un factor decisivo en la calidad del sonido. Por ejemplo existen *SoundFonts* tales que no incluyen la totalidad de las notas posibles grabadas, sino que se graban algunas y otras se aproximan alterando la frecuencia, de forma sintética. Para el público al cual apunta el Bandonberry, es suficiente una *SoundFonts* estándar de bandoneón[12].

2.3.3. Botoneras

El programa *botoneras.py* es el encargado de leer el estado de todos los botones y enviar al sintetizador las notas MIDI correspondientes a cada botón. Por otro lado, a través de un GPIO (*General Purpose Input Output*) controla el servomotor, de manera de ajustar la resistencia hidráulica del fuelle. Esto depende, al igual que en el bandoneón, de la cantidad de botones presionados.

2.3.4. Fuelle

El programa *fuelle.py* se comunica por I2C con el sensor de presión para obtener las medidas y calcular el presión relativa en el fuelle. Esta información es enviada a través de mensajes MIDI al sintetizador de manera de modular la intensidad de las notas.

2.3.5. RaspiATX

El módulo de software RaspiATX[13] fue creado para utilizarse con un módulo electrónico específico denominado ATXRasp, vendido por la compañía LowPowerLab [14]. Básicamente, maneja las señales involucradas y gestiona el suministro de energía.

En la Raspberry Pi se instala RaspiATX y se crea, con el Microcontrolador del BMS, señales equivalentes a las que hace el módulo electrónico ATXRasp. De esta

[8]Disponible en `http://timidity.sourceforge.net/`.

[9]Disponible en `http://yoshimi.sourceforge.net/`.

[10]Disponible en `http://zynaddsubfx.sourceforge.net/`.

[11]Formato de archivo (.sf2) utilizado para la síntesis de audio a través de muestras de sonido.

[12]Publicadas y disponibles de forma gratuita en `http://joergbleymehl.de/en/bandochords/soundfont/`.

[13]Disponible en el repositorio `https://github.com/LowPowerLab/ATX-Raspi`.

[14]Empresa estadounidense con sede en Canton, Michigan (`https://lowpowerlab.com/`).

manera, se reutiliza el código y se adapta el BMS para que trabaje con él. El objetivo del software es generar señales que permitan ejecutar apagados y encendidos seguros, en el sentido de no producir cortes de energía en momentos inadecuados. Se busca evitar dichos cortes particularmente en momentos de escritura en memoria, ya que podrían causar que la misma se corrompa.

En la Sección 6.6.1 se detalla más en detalle como funcionan las señales y cómo esta definida la comunicación entre el BMS y la Raspberry Pi.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 3

Single Board Computer

Para sintetizar el sonido y leer los datos de entrada del fuelle y botoneras se utiliza una SBC. En el presente capítulo se explica el funcionamiento, conexiones y configuraciones realizadas en la SBC.

3.1. Requerimientos

La SBC debe ser capaz de leer los datos de entrada del sensor del fuelle, leer el estado de las botoneras y sintetizar el sonido del bandoneón. Debe ser modular o ampliable para permitir a futuro agregar nuevas funcionalidades, por ejemplo, herramienta para estimular el aprendizaje del instrumento.

Debe soportar un sistema operativo y software de código abierto. También, es necesario la presencia de un puerto USB OTG [1] que permita la conexión con una PC y ser reconocido como un periférico. Utilizando dicha conexión, se deberá generar una comunicación a través de mensajes MIDI.

3.2. Selección de placa SBC

En la Tabla 3.1 se observa la comparación de algunas prestaciones de cuatro SBCs presentes en el mercado. Los costos son expresados en el mercado de origen y no incluyen envío.

La Raspberry Pi Zero W, cumple con muchas de las características buscadas, a pesar de no tener una salida de audio analógica. Dicha ausencia, se resuelve agregando tarjetas de sonido externas. Por otro lado, la Raspberry Pi Zero W, tiene un precio superior a la Orange Pi Zero. Sin embargo la primera posee mayor documentación disponible y comunidad más activa, con respecto a la segunda, y por lo tanto es la que se selecciona para el proyecto.

[1] USB OTG (On The Go) permite a los dispositivos actuar tanto como el *host* o el *client* en la comunicación USB. En el Bandonberry la SBC es el *client* y la PC el *host*.

	Raspberry Pi 3 B	**Raspberry Pi Zero W**	**BeagleBone Black**	**Orange Pi Zero**
Costo (USD)	35	10	55	9
USB OTG	No	Si	Si	Si
WiFi	Si	Si	No	Si
Bluetooth	Si	Si	No	No
Audio analógico	Si	No	No	Si
Origen	USA	USA	USA	China

Tabla 3.1: Comparación de distintas SBCs.

3.3. Raspberry Pi Zero W

En la Figura 3.1 se observa una placa Raspberry Pi Zero W. Posee dos puertos micro USB, uno es utilizado como entrada de energía (PWR), mientras que el otro es el puerto USB OTG. Posee un puerto mini HDMI para conectar, eventualmente, un monitor. Además, se tiene un conector de cable plano para conectar una cámara. El sistema operativo y las aplicaciones se instalan en la unidad de almacenamiento principal, la cual se coloca en el puerto micro SD.

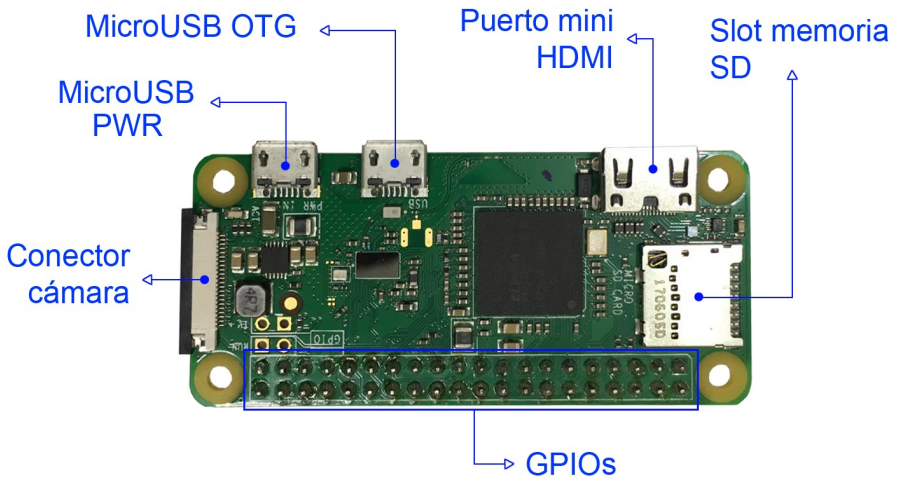


Figura 3.1: Raspberry Pi Zero W.

Una de las características más importantes de las Raspberry Pi Zero W es la presencia de un conector de 40 GPIOs, que permiten la interconexión con otros periféricos, extendiendo así las capacidades de la SBC. En la Tabla 3.2 se encuentra la descripción de los usos de los 40 pines y sus respectivas funciones en el sistema del Bandonberry. Entre dichos pines se encuentra un puerto I2S [2] (ver Anexo C.1) utilizado enviar la información de audio digital a los amplificadores de audio. Un

[2]Del inglés *Inter IC Sound*, interfaz digital para interconectar dispositivos de audio.

puerto SPI[3] (ver Anexo C.2) donde se conectan las botoneras y un puerto I2C (ver Anexo C.3) donde se conecta el sensor de presión. La comunicación con el servomotor y el BMS se realiza a través de pines de propósito general. Un total de 12 GPIOs quedan libres para extender las capacidades del Bandonberry en desarrollos futuros.

Uso	**Función**	**Pin**	**Pin**	**Función**	**Uso**
	3.3 V	1	2	5 V	5 V
PRESIÓN	I2C1 SDA	3	4	5 V	5 V
PRESIÓN	I2C1 SCL	5	6	GND	GND
SERVOMOTOR	GPIO 04	7	8	GPIO 14	
GND	GND	9	10	GPIO 15	
BMS	GPIO 17	11	12	BITCLK	AUDIO
	GPIO 27	13	14	GND	GND
	GPIO 22	15	16	GPIO 23	
	3.3 V	17	18	GPIO 24	
BOTONERAS	SPI0 MOSI	19	20	GND	GND
BOTONERAS	SPI0 MISO	21	22	GPIO 25	
BOTONERAS	SPI0 SCLK	23	24	SPI0 CS0	BOTONERAS
GND	GND	25	26	SPI0 CS1	BOTONERAS
	EPROM	27	28	EPROM	
	GPIO 05	29	30	GND	GND
	GPIO 06	31	32	GPIO 12	BMS
	GPIO 13	33	34	GND	GND
AUDIO	LRCLK	35	36	GPIO 16	BMS
	GPIO 26	37	38	GPIO 20	
GND	GND	39	40	DATAOUT	AUDIO

Tabla 3.2: Uso de pines en Raspberry Pi Zero W.

La alimentación de la Raspberry Pi es de 5 V y se puede alimentar tanto desde el conector micro USB, como desde pines específicos, concebidos para dicha función. Los niveles lógicos de entradas y salidas digitales son todos de 3,3 V.

El sistema operativo utilizado no posee entorno gráfico, por lo tanto la forma de comunicarse es mediante el protocolo SSH [4]. La presencia de WiFi incorporado en la Raspberry Pi Zero W, permite que se puedan hacer cambios en su programación sin la necesidad de una conexión por cable.

[3]Del inglés *Serial Peripheral Interface*, interfaz de periférico serie.

[4]Del inglés *Secure SHell*, Protocolo de red para el intercambio de datos de manera segura (cifrado) entre dispositivos conectados en red.

3.3.1. Rendimiento

La Raspberry Pi Zero W posee una CPU[5] de un solo núcleo con un reloj de 1 GHz y 512 Gb de memoria RAM[6]. Utilizando el comando *top* [7] se puede visualizar el estado y carga del sistema. Los resultados obtenidos en condición de reposo, se presentan en la Tabla 3.3.

Programa	**CPU (%)**	**RAM (%)**
Sintetizador	28,9	2,9
Botoneras	21,3	1,8
Fuelle	21,0	2,0

Tabla 3.3: Estado de carga del sistema en reposo.

Las tareas que más recursos consumen son el sintetizador y los *scripts* de Python, que corresponden a los programas asociados al fuelle y a las botoneras. Los resultados antes mostrados corresponden al sistema en reposo (sin tocar el instrumento) y conectado por WiFi.

Cuando se comienza a tocar, el uso de CPU del sintetizador se aproxima al 40 %, mientras que el consumo de RAM permanece constante. Los otros programas no muestran aumento de demanda de recursos. El estar conectado a una PC no afecta el uso de recursos.

En resumen, se observa de un 70 a 80 % de uso de CPU y 7 % de memoria RAM. Esto muestra que las prestaciones de la SBC son suficiente para la aplicación. En caso de querer agregar más funcionalidades al sistema o mejorar los tiempos de repuesta, la mayor limitante es la CPU.

3.4. Audio

Dado que la Raspberry Pi Zero W no posee salida analógica de audio es necesario agregar hardware adicional para generar las señales. Como hardware de salida de audio analógica se elige el integrado MAX98357 [16]. Este circuito integrado posee entrada digital I2S y su salida es una señal analógica con la potencia necesaria para ser conectada directamente a un parlante.

En el mercado es posible adquirirlo en un módulo, en particular se selecciona uno de la empresa Adafruit [8]. El módulo posee los componentes pasivos adicionales para su correcto funcionamiento, tal como se observa en la Figura 3.2. Los pines LRC, BCLK y DIN son los correspondientes al puerto I2S los cuales se conectan a la Raspberry Pi (ver Tabla 3.2). El pin GAIN se utiliza para controlar la ganancia. En el caso del Bandonberry, dicho pin se deja desconectado y por lo tanto queda definida la ganancia con su valor por defecto de 9 dB.

[5]Del inglés *Central Processing Unit*.

[6]Del inglés *Random Access Memory, RAM*.

[7]El comando *top* permite monitorear los procesos y el uso de recursos del sistema en Linux.

[8]Empresa estadounidense con base en Nueva York (https://www.adafruit.com/).

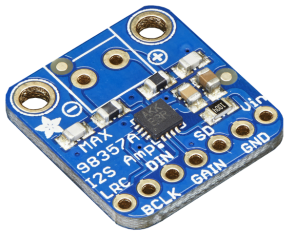


Figura 3.2: Módulo MAX98357.

Estos módulos solo poseen un canal de audio, por lo tanto es necesario utilizar dos dado que los sonidos de la mano derecha salen por la derecha y los de la mano izquierda salen por la izquierda en el bandoneón. En el parlante izquierdo se reproducen los sonidos de la botonera izquierda, mientras que en el parlante derecho se reproducen los sonidos de la botonera derecha. El pin SD es utilizado para seleccionar el canal que reproduce cada módulo: izquierdo, derecho o una combinación de los dos. Una resistencia a Vin en los pines SD indica a cada módulo el canal que debe reproducir:

- Canal izquierdo 150 kΩ
- Canal derecho 1 MΩ

En la Figura 3.3 se muestra el diagrama de interconexión.

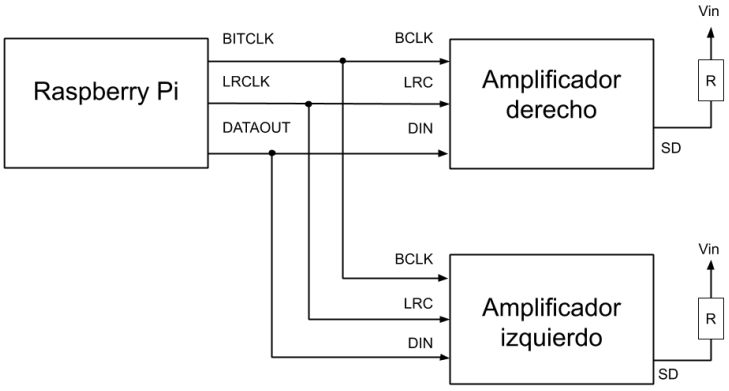


Figura 3.3: Conexión de módulos de audio.

El voltaje de alimentación, según las especificaciones es de 2,7 V a 5,5 V. Para obtener su potencia máxima de 3 W en un parlante de 4 Ω, el módulo se alimenta con 5 V. Sin embargo las líneas de datos son de niveles lógicos de 3,3 V, lo cual permite que se puedan conectar directamente a la Raspberry Pi.

La salida de potencia es de Clase D [23] y consta de una señal PWM[9] de 300 kHz la cual es filtrada por un filtro LC y luego por el parlante conectado a ella.

[9]Del inglés *Pulse Width Modulation*, modulación por ancho de pulsos.

3.5. Comunicación MIDI

MIDI es un estándar de comunicación que permite que instrumentos musicales electrónicos, computadoras y otros dispositivos digitales puedan conectarse y comunicarse entre sí. El estándar define tanto la interfaz física (conectores y niveles de voltaje), como el protocolo de comunicación y tipos de archivo.

En el Bandonberry no se utiliza ninguna interfaz MIDI física (solo la emulada a través del puerto USB). Sin embargo, el protocolo es utilizado a través de *software* como una forma de comunicar los diferentes programas. Se elige MIDI ya que es un estándar en el mundo de la música y es la forma natural de interactuar con el sintetizador.

Un mensaje MIDI está compuesto de dos o tres *bytes*, el primero es llamado STATUS BYTE el cual es acompañado de uno o dos DATA BYTES. Los cuatro *bits* más significativos del STATUS BYTE indican el tipo de mensaje y los cuatro *bits* menos significativos indican el CHANNEL. Es posible utilizar hasta 16 canales en MIDI. Cada canal puede cargarse con diferentes instrumentos, reproducir sus propias notas y tener configuraciones diferentes. La cantidad y significado de los DATA BYTES dependen del tipo de mensaje.

En la Tabla 3.4 se muestra un subconjunto de los mensajes MIDI utilizados en el Bandonberry. La "X" en STATUS BYTE indica el canal al cuál está dirigido el mensaje, puede tomar valores de 0 a F en hexadecimal.

Los primeros dos mensajes, NOTE_ON y NOTE_OFF, se utilizan para indicar el comienzo y fin de la reproducción de una nota, respectivamente. El primer DATA BYTE indica la nota y el segundo la intensidad con qué es reproducida.

Para los mensajes del tipo CONTROL, el primer DATA BYTE indica el tipo de control que se desea modificar y el segundo el valor que se desea asignar a dicho control.

STATUS BYTE		**DATA BYTE 1**		**DATA BYTE 2**	
Valor	**Significado**	**Valor**	**Significado**	**Valor**	**Significado**
0x9X	NOTE_ON	0x00 - 7F	NOTA	0x00 - 7F	VELOCITY
0x8X	NOTE_OFF	0x00 - 7F	NOTE	0x00 - 7F	VELOCITY
0xBX	CONTROL	0x07	VOLUME	0x00 - 7F	VOLUME
0xBX	CONTROL	0x0A	BALANCE	0x00 - 7F	BALANCE

Tabla 3.4: Mensajes MIDI utilizados.

En el Bandonberry, se utilizan cuatro canales para producir las diferentes notas (abriendo o cerrando) como se muestra en la Tabla 3.5. Los canales 0 y 1, correspondientes a la botonera derecha, se configuran con el balance (la intensidad con que se reproduce el sonido en cada parlante) hacia la derecha (DATA BYTE 2 = 0x7F). Los canales 2 y 3, correspondientes a la botonera izquierda, se configuran con el balance hacia la izquierda (DATA BYTE 2 = 0x00).

El programa *botoneras.py* reproduce ambas notas (abriendo y cerrando) para los canales correspondientes a cada botonera. El programa *fuelle.py* se encarga de

Canal	**Uso**
0	Botonera derecha abriendo
1	Botonera derecha cerrando
2	Botonera izquierda abriendo
3	Botonera izquierda cerrando

Tabla 3.5: Distribución de canales MIDI.

modular la amplitud del canal correspondiente dependiendo del movimiento del fuelle. Cuando el fuelle se está abriendo, los canales 0 y 2 son modulados con un volumen proporcional a la presión, mientras los canales 1 y 3 tienen un volumen nulo. La lógica se invierte cuándo el fuelle se está cerrando.

Un único canal no es suficiente para generar el efecto necesario de reproducir notas diferentes en cada uno de los parlantes, el mínimo necesario es dos. La solución adoptada utiliza cuatro para independizar los programas *botoneras.py* y *fuelle.py*. De haber utilizado solo dos canales el programa *botoneras.py* necesita conocer el estado del fuelle (abriendo o cerrando) para identificar cuál de las dos notas debe reproducir. Esto introduce latencias, no deseadas, en la respuesta a los movimientos del fuelle.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Botoneras

En el presente capítulo se explica cómo utilizando los 5 pines correspondientes a la comunicación SPI de la SBC, es posible manejar la entrada de 71 botones con latencias aceptables. Se explica tanto a nivel de *hardware*, como de *software* las consideraciones que se tuvieron a la hora de diseñar las botoneras.

4.1. Requerimientos

Es necesario detectar el accionar de distintos botones de manera simultánea (10 si se considera ambas manos) para que el Bandonberry se comporte como el bandoneón. Se debe tener una latencia entre la detección y la generación de sonido del orden de los 200 ms para que no sea molesto al usuario.

4.2. Restricciones

La Raspberry Pi posee un puerto de 40 pines, solo 26 son GPIOs que pueden ser utilizados (sin considerar los pines necesarios para utilizar los otros módulos). Por lo tanto, no es posible tener para cada botón una entrada directa. Para resolver el problema, las botoneras deben poseer un sistema de multiplexado que permita usar una cantidad reducida de conexiones y a la vez mantenerse dentro de los objetivos de tiempos de respuesta.

4.3. Solución planteada

La botonera con el mayor número de botones es la derecha, con 38 botones, por lo tanto se diseña una solución para esta cantidad y luego se replica en la otra. Una combinación de 6 filas y 8 columnas da la posibilidad de medir 48 puntos, de esta forma se necesitan 26 GPIOs en total. De todas formas sigue siendo un número alto de pines, dado que limita las posibilidades para conectar otras partes del Bandonberry.

Este problema se resuelve con la incorporación de circuitos integrados que permiten agregar GPIOs adicionales y controlarlos mediante una interfaz serie, conocidos como *GPIO expanders*. Se elige esta solución sobre la posibilidad de utilizar registros de desplazamiento. Su uso implica una mayor cantidad de circuitos integrados (teniendo en cuenta integrados de 8 *bits*), mayor cantidad de pines y la necesidad de escribir *software* que maneje las líneas de control.

El Bandonberry utiliza módulos expansores de puertos de Microchip, el MCP23S17 [20], que implementa una interfaz SPI y proporciona 16 GPIOs. Este integrado puede ser alimentado con voltajes entre 1,8 y 5,5 V, pero para asegurar la compatibilidad con los niveles lógicos de la SBC se alimenta con 3,3 V.

Se opta por utilizar una comunicación SPI dado que es más rápida que una I2C. La versión equivalente de este integrado con comunicación I2C (MCP23017) puede llegar a velocidades de reloj de hasta 1,7 MHz, mientras que el seleccionado llega a 10 MHz.

4.4. Conexión y multiplexado

En la Figura 4.1 se observa un ejemplo de conexión de una matriz de cuatro pulsadores. Los pulsadores se conectan en las interconexiones entre filas y columnas con un diodo en serie.

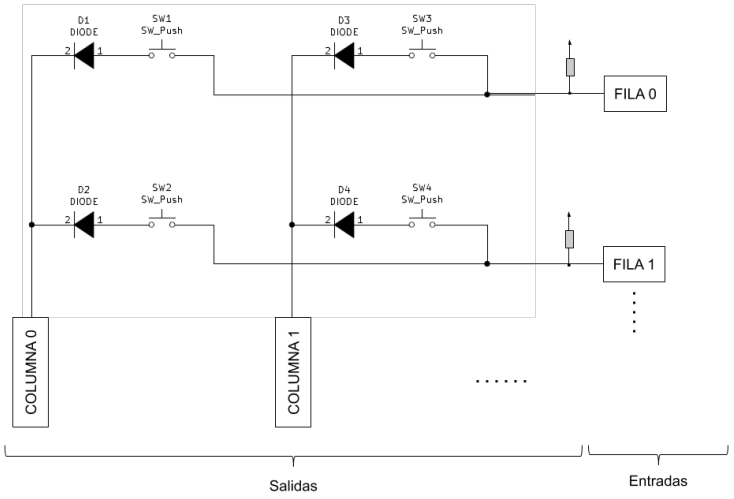


Figura 4.1: Esquema de conexión de la matriz de botones.

Se adquiere el estado de los botones una columna por vez. Se activa una columna, poniendo su salida en nivel LOW y se leen el estado de las filas. Las restantes columnas permanecen en nivel HIGH. Cuando un pulsador en la columna activa (nivel LOW) es presionado, la correspondiente fila es también activada (nivel LOW). El resto de las filas permanecen en estado inactivo (nivel HIGH) debido a resistencias de *pullup*. La presencia de los diodos es necesaria para evitar que se

produzcan cortocircuitos o falsas lecturas cuando más de un pulsador es presionado a la misma vez.

Ambas botoneras cuentan con los mismos criterios de diseño electrónico, cambiando solo la ubicación y cantidad de pulsadores. Los 16 GPIOs del integrado MCP23S17 se organizan en dos puertos (A y B) de 8 *bits* cada uno. En la Tabla 4.1 se muestra la distribución de pines del integrado MCP23S17 con sus respectivas funciones y el uso que se le da en las botoneras. Como se explica en el Apéndice C.2 las señales CLK, MISO y MOSI son compartidas y se conectan a los pines 23, 21 y 19 de la SBC, respectivamente. Por otro lado, la señal CS de la botonera derecha se conecta a CS0 y el de la izquierda a CS1, pines 24 y 26 de la SBC, respectivamente. De esta manera, en *software*, la botonera derecha responde al puerto SPI0 y la izquierda al SPI1.

Uso	**Función**	**Pin**	**Pin**	**Función**	**Uso**
COLUMNA 0	B0	1	28	A7	FILA 7
COLUMNA 1	B1	2	27	A6	FILA 6
COLUMNA 2	B2	3	26	A5	FILA 5
COLUMNA 3	B3	4	25	A4	FILA 4
COLUMNA 4	B4	5	24	A3	FILA 3
COLUMNA 5	B5	6	23	A2	FILA 2
	B6	7	22	A1	FILA 1
	B7	8	21	A0	FILA 0
3.3 V	VDD	9	20	INTA	
GND	VSS	10	19	INTB	
CS0 o CS1	nCS	11	18	nRESET	3.3 V
SPI0 SCLK	SCK	12	17	A2	GND
SPI0 MOSI	SI	13	16	A1	GND
SPI0 MISO	SO	14	15	A0	GND

Tabla 4.1: *Pinout* de los *GPIO expanders*.

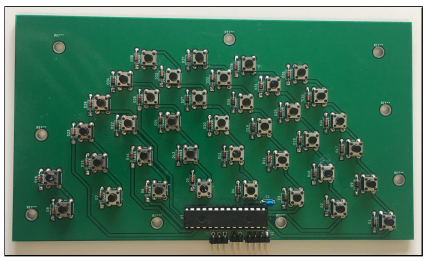
El puerto B se encuentra conectado a las columnas y el puerto A a las filas. El puerto A se configura como entrada con *pullups* en todos sus pines y el puerto B se configura como salida, donde la columna activa se encuentra en nivel LOW y el resto de los pines permanecen en HIGH.

Los pulsadores elegidos son genéricos de 5x5 mm, de bajo costo y sencillos de conseguir. Estos tienen una excursión mecánica corta y solo dos estados (cerrado o abierto). Sin embargo, los botones del bandoneón tienen mayor excursión mecánica (1 cm aproximadamente) y permiten alterar el sonido de forma diferente a lo largo de la excursión.

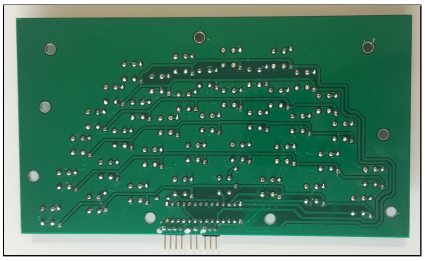
Para dar una sensación similar a la del bandoneón, se diseñan botones con un sistema de resorte que permite tener un recorrido mayor. Lograr alterar el sonido en forma gradual según el recorrido requiere agregar una alta complejidad al diseño. Dado que este recurso no es algo utilizado por principiantes, se decide no considerarlo como requerimiento en el desarrollo del Bandonberry.

4.5. PCBs

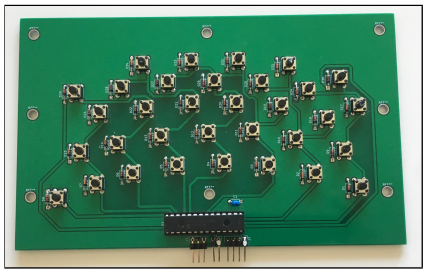
Las PCBs ubican a los interruptores en la misma posición del bandoneón y los interconecta con los *expanders* en forma de matriz. Para el diseño de las botoneras se utilizó el software KiCAD [1]. Los diseños se enviaron a una empresa china de fabricación de PCBs llamada JLCPCB [2] , obteniéndose las PCBs de la Figura 4.2.



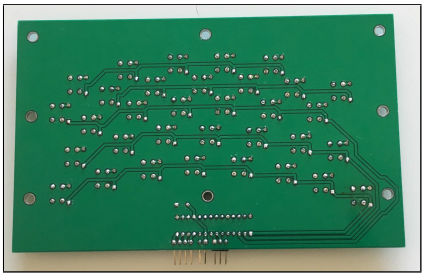
(a) Capa superior derecha.



(b) Capa inferior derecha.



(c) Capa superior izquierda.



(d) Capa inferior izquierda.

Figura 4.2: PCBs de botoneras.

Como estrategia de diseño de la disposición espacial de las pistas en la PCB, se priorizaron las que tienen dirección horizontal en la capa inferior y las verticales en la capa superior. En la parte inferior se encuentran las conexiones para la alimentación y comunicación de la placa, facilitando el acceso en el ensamble final del Bandonberry.

4.6. Software

Para el programa *botoneras.py*, se crea la librería BDN_MCP23S17 la cual abstrae la comunicación SPI y el manejo de los registros del integrado MCP23S17. Esta se basa en otra librería, RPi-MCP23S17 [3], la cuál presentó problemas a la hora de ser utilizada. Por esa razón se toma como base y se modifica para cumplir las necesidades del Bandonberry.

[1] Software para diseño de PCBs, disponible en `http://kicad-pcb.org/` de licencia libre.

[2] Para más información `https://jlcpcb.com/`.

[3] Disponible en https://github.com/petrockblog/RPi-MCP23S17.

En primer lugar se establece conexión con el puerto MIDI del sintetizador y el puerto MIDI emulado en la interfaz USB. Todos los mensajes son enviados por igual a ambos puertos. La presencia o no de una PC conectada por USB no afecta el comportamiento del Bandonberry en ninguno de sus aspectos.

Para el proceso de lectura de las señales de las botoneras se definen dentro del programa botoneras.py, dos arreglos de tres dimensiones que contienen el mapeo de notas de cada botón para cada mano. La primera coordenada define la columna, la segunda la fila y la tercera la dirección (abriendo o cerrando). En el Cuadro 4.1 se muestra el pseudocódigo de lectura de botoneras.

Cada fila de cada botonera es leída NUM_OF_READS veces con un intervalo de DEBOUNCE_DELAY ms entre sucesivas lecturas. Se le aplica un *OR* lógico entre ellas. De esta manera el estado LOW es detectado solo si las NUM_OF_READS lecturas son todas LOW, eliminando los efectos de rebote de la señal cuando el botón cambia de estado.

```
# Para cada columna en COLUMS:
    # Activo el pin del puerto B correspondiente en cada
      botonera
    # NUM_OF_READS veces:
        # Leo el estado de los puertos A en cada botonera
        # Realizo OR logico con las lecturas anteriores
        # Espero DEBOUNCE_DELAY milisegundos
    # Busco cambios en las ROWS filas
```

Cuadro 4.1: Pseudocódigo de lectura de botoneras.

Los valores utilizados son: seis columnas, con dos lecturas de fila y un *delay* entre lecturas de 1 ms. Si se considera el tiempo de 15 μs observados en la escritura y lectura de registros de los expansores de puertos, se estima que el *loop* completo de lectura de botoneras es de:

$$COL \times ((2 \times SPI) + NUM\_OF\_READS \times ((2 \times SPI) + DEBOUNCE)) \quad (4.1)$$

$$6 \times ((2 \times 15\ \mu s) + 2 \times ((2 \times 15\ \mu s) + 1\ ms)) = 12,54\ ms \quad (4.2)$$

Sumado al *delay* entre que se envía el mensaje MIDI y el sonido es reproducido, el total se encuentra cercano a los 100 ms, lo cual es un valor aceptable con respecto al objetivo de 200 ms. En la Sección 7.6 se realiza un estudio del *delay* final con todos los programas corriendo en simultáneo.

4.7. Latencias asociadas a los expansores de puertos

Dado que los expansores de puertos utilizan un protocolo de comunicación serial se introduce un *delay* para poder enviar y recibir los datos entre la SBC y sus puertos.

4.7.1. Comunicación serial

Con el fin de determinar si los integrados seleccionados son adecuados para la tarea, se diseñan programas de prueba que permiten medir los tiempos de ejecución. Se utilizan GPIOs conectados a un osciloscopio para poder visualizar y medir los *delays*.

En primer lugar se mide el tiempo necesario para pasar de valor lógico LOW a HIGH lógico en un GPIO propio de la SBC para considerarlo en otras medidas. Luego se mide lo que demora enviar el mensaje MIDI al sintetizador. El tiempo que lleva cambiar el estado de un GPIO propio de la SBC es de 10 μs y el tiempo que lleva enviar un mensaje MIDI es 0,5 ms (incluyendo el tiempo en cambiar el estado del GPIO).

```
# GPIO en SBC a LOW
# GPIO en SBC a HIGH
# GPIO en SBC a LOW

# GPIO en SBC a HIGH
# Envio dato MIDI a Sintetizador
# GPIO en SBC a LOW
```

Cuadro 4.2: Pseudocódigo de medida.

Una escritura o lectura de los registros del integrado MCP23S17 necesita de la transmisión de 3 *bytes*:

```
DEVICE OPCODE + REGISTER ADDRESS + REGISTER VALUE
```

Las velocidades de reloj en el puerto SPI de las Raspberry Pi se encuentran preestablecidas, siendo 7,8 MHz la frecuencia máxima compatible con el expansor de puertos. A la frecuencia del bus SPI seleccionada, la duración de la tarea teóricamente es:

$$3\ B \times 8\ bit/B \times 7{,}8\ Mb/s = 3{,}1\ \mu s \tag{4.3}$$

Para comprobar esto se mide el tiempo en que la señal CS [4] del puerto SPI permanece activa, observándose tiempos entre 5 y 15 μs. Las variaciones en estos tiempos, se deben a la presencia de un procesador de un solo núcleo, el cual debe atender a todos los programas que corren de manera secuencial.

Se debe tener en cuenta que todos los tiempos medidos son afectados por la cantidad de aplicaciones que corren en el sistema operativo. De todas maneras, estas medidas, dan una idea de órdenes y permiten identificar los lugares que contribuyen con mayores tiempos de respuesta.

4.7.2. Salida de audio

Se diseña un programa que permite medir el tiempo entre que se oprime un botón y el sonido es reproducido en el parlante. Lee constantemente uno de los

[4] Señal *Chip Select*.

GPIOs del MCP23S17 configurado como entrada y cuando detecta un flanco de bajada (el botón es presionado) activa un GPIO y envía un mensaje MIDI al sintetizador. En la Figura 4.3 se muestra un diagrama de bloques y los puntos de conexión del osciloscopio.

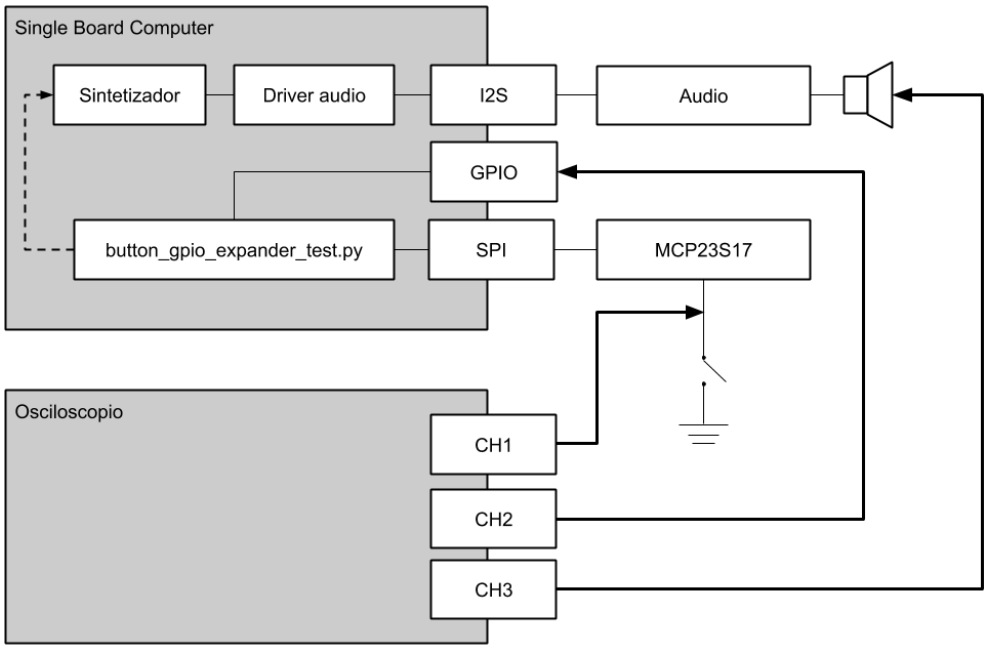


Figura 4.3: Conexión para medición de *delay* de la botonera a la salida de audio.

Este experimento permite identificar cual es la mayor contribución de tiempo al *delay* entre que se envía el mensaje al sintetizador y se reproduce la nota en el parlante. La Figura 4.4 muestra un *delay* entre que se presiona el botón (CH1) y el sonido sale por el parlante (CH3) de 200 ms. El tiempo que se demora en reconocer que el botón es presionado y el mensaje MIDI se envía, es despreciable (CH2) con respecto al otro.

Según la "Wiki de Fluidsynth" [29] la latencia se define en la Ecuación 4.4, donde $RATE$ es la frecuencia de muestreo, NUM es la cantidad de *buffers* de audio y $SIZE$ es el tamaño de cada uno.

$$Latencia = \frac{NUM \times SIZE}{RATE} \tag{4.4}$$

En la Tabla 4.2 se muestran los valores por defecto y los valores utilizados en el Bandonberry. Los valores se configuraron para minimizar la latencia, manteniendo la frecuencia de muestreo, de manera de no perder calidad de sonido. Con los parámetros por defecto se tiene una latencia esperada de 22 ms mientras que con la configuración seleccionada se obtiene un valor esperado 2,7 ms.

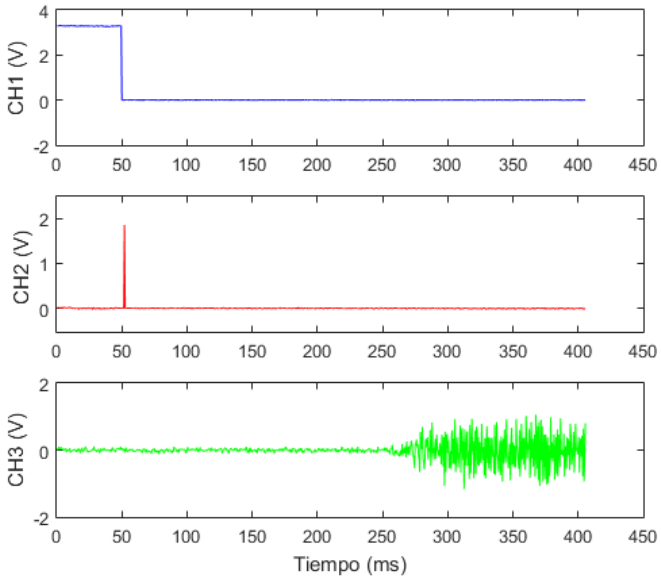


Figura 4.4: Resultados de medición del *delay* de la botonera a salida de audio.

Parámetro	**Rango**	**Por defecto**	**Bandonberry**
RATE (kHz)	8 - 96	44,1	44,1
NUM (buffers)	2 - 64	16	2
SIZE (muestras)	64 - 8192	64	64

Tabla 4.2: Parámetros de configuración de Fluidsynth

En la práctica, se mide con la configuración por defecto 200 ms y con la nueva configuración 80 ms, disminuyendo en 120 ms la latencia. Lo observado muestra que hay otras contribuciones de tiempo que escapan a la ecuación planteada. Como posibles fuentes de retardo se tiene el rendimiento del *driver* de sonido y el conversor analógico digital de la salida de audio. De todas maneras, la nueva latencia disminuye a menos de 100 ms, por lo que se considera un resultado satisfactorio.

Capítulo 5

Fuelle

El fuelle es uno de los componentes fundamentales del bandoneón, es el encargado de mover el aire que pasa por las lengüetas para producir el sonido. En este capítulo se describe el proceso de construcción, la técnica de medición de presión y los mecanismos de control de flujo de aire que posee el fuelle del Bandonberry.

5.1. Requerimientos

Uno de los objetivos importantes del proyecto consiste en imitar la física y dinámica del bandoneón. Esto implica que debe ser capaz de simular la resistencia característica del fuelle sobre los brazos de quien utiliza el instrumento. Al igual que en el bandoneón, la intensidad sonora de la nota reproducida debe depender de la presión ejercida sobre el fuelle. La resistencia que el fuelle presente al intérprete debe variar en función de la cantidad de botones presionados y debe ser muy baja cuando se acciona la palanca de toma de aire. Por este motivo, el fuelle del Bandonberry debe medir la presión ejercida y controlar el flujo de aire que entra y sale de él.

5.2. Solución planteada

Para poder identificar y medir si el fuelle se está comprimiendo o expandiendo, y la intensidad con la cual lo hace, se consideran dos opciones:

1. Medir con un sensor la presión interna del fuelle y compararla con la presión atmosférica.

2. Medir la posición relativa de una botonera respecto a su opuesta.

El Bandonberry utiliza la primera opción dado que, por su naturaleza, brinda una relación directa a los requerimientos planteados. Como contraparte este tipo de medida requiere de un fuelle con pocas fugas de aire, ya que las diferencias de presión deben ser apreciables por el sensor que se utiliza.

Para controlar la resistencia que el fuelle ejerce al movimiento se consideran en la etapa de investigación dos opciones para resolver este problema:

1. Utilizar una electroválvula controlada por corriente para permitir o impedir el paso del aire, desde o hacia el interior del fuelle.

2. Con un servomotor y un dispositivo mecánico acorde, construir una válvula capaz de controlar el paso del aire en función del giro del eje del servo.

La primera opción presenta un inconveniente con respecto a lo disponible en plaza, el mínimo voltaje de las electroválvulas es de 12 V. Esto difiere con los voltajes ya presentes en el sistema de 3,3 V y 5 V.

Se realizaron pruebas con una electroválvula que indicaron que si bien es capaz de controlar los flujos de aire, están diseñada originalmente para fluidos y no para gases. Esta última utiliza un solenoide con un tamaño considerable y un consumo importante en referencia al resto de los componentes utilizados en el Bandonberry. Por otro lado, no hay posibilidad de modificar el tamaño del tubo y por ende controlar el flujo de aire. Por último, la válvula solo permite dos estados: abierto o cerrado. Por lo que para simular distintos niveles de resistencia se deben utilizar múltiples electroválvulas, aumentando aún más el nivel de consumo. Todos estos aspectos descartan esta opción.

La segunda opción es la implementada en el Bandonberry. La válvula con servomotor es capaz de brindar la funcionalidad buscada y sus requerimientos de alimentación son acordes a las capacidades del sistema.

5.3. Sensor de presión

El sensor seleccionado es el BMP180 [8] de BOSCH. Se trata de un sensor digital de presión barométrica con comunicación I2C. Puede medir presiones desde los 300 a los 1100 hPa y es calibrado en fábrica. Se puede alimentar con tensiones de 1,8 a 3,6 V. Para asegurar la compatibilidad con los niveles lógicos de la SBC se alimenta con 3,3 V. Este sensor se obtiene en forma de módulo con los componentes pasivos adicionales para su correcto funcionamiento, como se observa en la Figura 5.1.

Para asegurar que el sensor seleccionado cumple con su función se estudian dos aspectos importantes:

1. La sensibilidad, puesto que si el sensor no logra medir con precisión variaciones pequeñas de presión, no se puede utilizar para identificar compresiones o expansiones del fuelle.

2. El tiempo de respuesta, ya que resulta necesario que las variaciones de presión sean medidas y registradas dentro de los intervalos aceptables.

Para poder analizar los aspectos anteriores, se construye un primer modelo de fuelle de dimensiones menores al definitivo y se introduce el sensor de presión

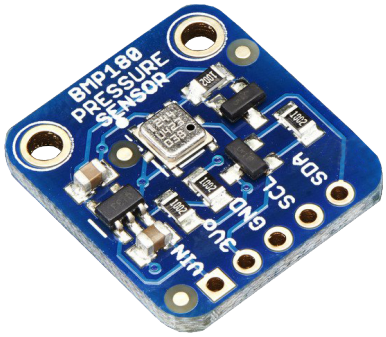
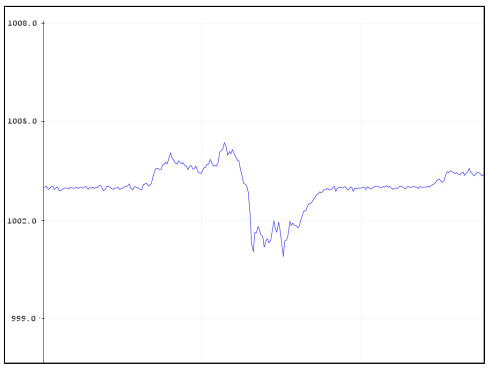
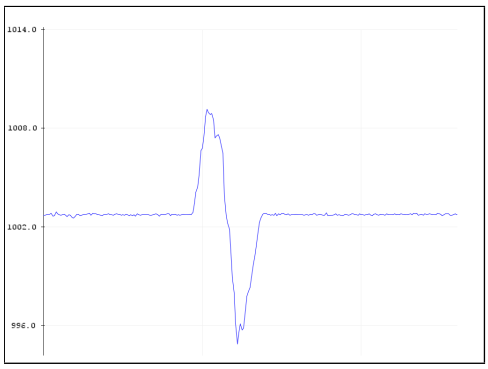


Figura 5.1: Módulo BMP180.

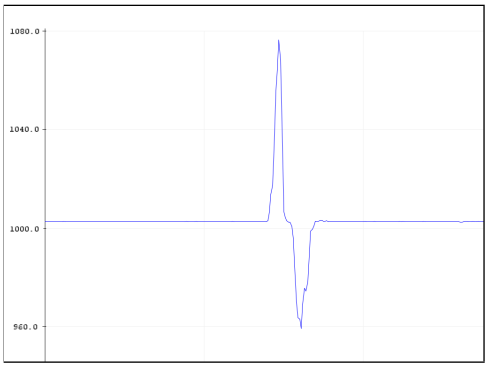
dentro. Los datos del sensor son obtenidos a través de un Arduino UNO y enviados a una PC por el puerto serial. Haciendo uso de la herramienta Serial Plotter del IDE[1] de Arduino, se observan gráficamente los valores que el Arduino envía en intervalos de 1 ms. Los resultados se pueden observar en la Figura 5.2.



(a) Suave



(b) Medio



(c) Fuerte

Figura 5.2: Variación de la presión en el tiempo en el interior del fuelle de prueba.

Se observa que los valores oscilan en torno al valor de presión del ambiente

[1]Del inglés *Integrated Development Environment*, entorno de desarrollo integrado.

(1003 hPa aproximadamente). Para los movimientos más suaves (Figura 5.2a) se observa una variación de entre 1 y 2 hPa. Si se tiene en cuenta que el error relativo del sensor a temperatura ambiente es de ± 0,12 hPa y que la resolución de los datos de salida es de 0,01 hPa (según la hoja de datos del fabricante), podemos concluir que el sensor es capaz de medir y distinguir las variaciones pequeñas.

Para el caso de movimientos fuertes y bruscos (Figura 5.2c), se puede observar que la variación máxima medida no alcanza los 80 hPa. Esto permite asegurar también que el sensor no sale de su rango operativo (recordando que el máximo del rango de operación es 1100 hPa).

5.3.1. Estudio de latencia asociada al sensor de presión

El sensor BMP180 además de medir presión, posee un sensor de temperatura, el cual es utilizado para compensar las medidas. La SBC debe obtener los datos de temperatura y presión para poder compensar esta última. El fabricante sugiere que se realice el flujo mostrado en la Figura 5.3.

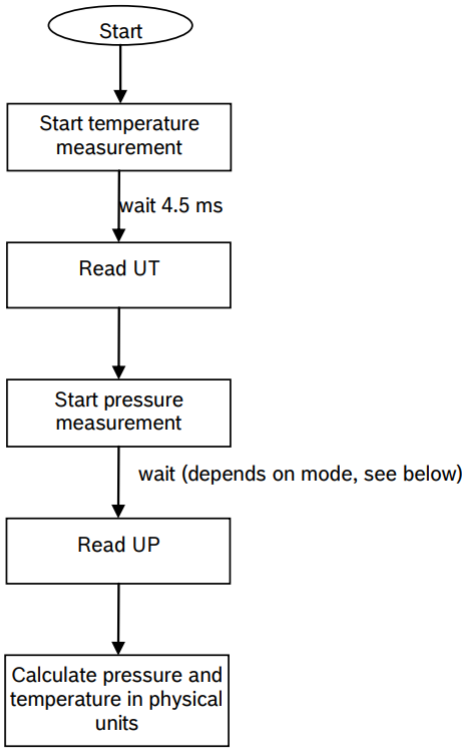


Figura 5.3: Flujo de medición para el sensor BMP180, extraída de [8].

La medición de presión presenta distintos tiempos de respuesta en la adquisición según el modo de funcionamiento elegido. Los distintos tiempos se muestran en la Tabla 5.1.

Modo	**Tiempo típico (ms)**	**Tiempo máximo (ms)**	**Cantidad interna de muestras**
Ultra low power	3	4,5	1
Standard	5	7,5	2
High resulution	9	13,5	4
Ultra high resolution	17	25,5	8

Tabla 5.1: Tiempos de conversión de presión del BMP180 para distintos modos de funcionamiento, extraído de [8].

Las medidas de temperatura no son necesarias hacerlas en cada ciclo. El fabricante recomienda en su documentación realizar una medida de temperatura por lo menos cada 1 segundo a efectos de que las medidas de presión sean correctas.

Los comandos para iniciar las lecturas y los datos deben ser transmitidos por el bus I2C. La señal tiene una velocidad de transmisión de datos con un valor por defecto de 100 kbps. En el inicio de cada transacción la Raspberry Pi debe enviar la dirección (de 8 *bits*) del sensor para indicarle que se quiere comunicar con el mismo. La cantidad de *bits* de datos que deben ser enviados para cada caso son:

- *Start temperature measurement*: 8 *bits*
- *Read UT*: 16 *bits* (palabra única)
- *Start pressure measurement*: 8 *bits*
- *Read UP*: 3x8 *bits* (3 palabras independientes)

Si se suma la cantidad de *bits* necesarios totales se puede estimar la duración de la comunicación I2C. Entonces siguiendo el esquema 5.3 y agregando 8 bits para agregar el dato de la dirección I2C por cada 8 bits enviados por el módulo se tiene:

$$[(8+8)+(8+16)+(8+8)+3\times(8+8)]\ bits/100\ kbps = 1040\ \mu s \qquad (5.1)$$

Se puede hacer un cálculo estimado del tiempo máximo total que se requiere en la medición, realizando la lectura de temperatura en cada ciclo, como la sumatoria de los tiempos involucrados desde el primer *byte* enviado por el sensor:

- 4,5 ms máximo para la medición de temperatura.
- 7,5 ms máximo para la presión (utilizando el modo estándar).
- 1,040 ms de la comunicación I2C.

Esto da un total de 13,04 ms como peor caso, lo cual representa un valor aceptable para la percepción humana. Se debe tener en cuenta que estos tiempos consideran únicamente la actividad del sensor y no de la rutina que lo utiliza, la cual agrega sus propios tiempos de ejecución.

5.3.2. Software

El programa *fuelle.py* es el encargado de leer los datos del sensor de presión y determinar el estado del fuelle. El sentido del movimiento es utilizado para determinar qué canales MIDI deben ser escuchados, mientras que la magnitud controla el nivel de volumen de dichos canales. Para la comunicación con el sensor se utiliza la librería Adafruit_BMP[2].

La diferencia con la presión atmosférica es la que permite detectar el sentido del movimiento y la presión ejercida. Para ello, es necesario determinar en primer lugar cual es la presión atmosférica. Se realizan 10 medidas y se hace el promedio entre ellas para determinar el valor de referencia de la presión atmosférica (haciendo el supuesto que el bandonberry esté en reposo durante la inicialización).

El *loop* principal del programa, mostrado en el Cuadro 5.1, toma nuevas muestras de la presión (variable *presion*) y calcula la diferencia con el valor de la atmosférica (variable *media*). Luego de obtener el valor absoluto de la diferencia (variable *value*) se escala el resultado para que se mapee en los valores de volumen esperados por mensajes MIDI (de 0 a 127). El valor de sensibilidad (variable *sense*) se obtuvo experimentalmente luego de observar los niveles de presión máximos alcanzados dentro del fuelle.

```
sense = 150
presion = sensor.read_pressure()
value = abs(presion - media)
value = (value * 127) / sense
if value > 127:
    value = 127
```

Cuadro 5.1: Código de lectura de valores de presión.

El valor de la magnitud del movimiento es asignada al canal correspondiente dependiendo de la dirección como se muestra en la Tabla 5.2.

Canal	***presion* >*media***	***presion* <*media***
Botonera derecha abriendo	*0*	*value*
Botonera derecha cerrando	*value*	*0*
Botonera izquierda abriendo	*0*	*value*
Botonera izquierda cerrando	*value*	*0*

Tabla 5.2: Asignación de volúmenes en canales MIDI según la dirección del movimiento.

El valor de la presión atmosférica es ajustado en cada iteración como una media móvil. Esta operación se muestra en el Cuadro 5.2.

```
media[n-1].alfa + presion.(1-alfa)
media = media * 0.999 + presion * 0.001
```

Cuadro 5.2: Código del cálculo de la presión de referencia.

[2]Disponible en `https://github.com/adafruit/Adafruit_Python_BMP`.

Esto permite, por una lado, ajustar de forma gradual los posibles cambios de presión del ambiente donde se está reproduciendo el Bandonberry. Por otro, hace tender el valor de *value* a cero cuando se deja de tocar, impidiendo que por algún motivo el Bandonberry se mantenga con una nota activa estando estático.

Se puede ver que la ecuación del Cuadro 5.2 es en esencia un filtro pasabajos. Para mostrar este comportamiento, se supone una entrada sinusoidal a la cual se le varía la frecuencia de excitación. En la Figura 5.4 se puede ver la variación de la ganancia en función de la frecuencia de oscilación, donde claramente se ve disminuida para altas frecuencias.

La caída de -3 dB está en aproximadamente 0,1 Hz. Por lo tanto, se puede decir que los movimientos con períodos menores a 10 segundos no afectarán el valor de la media de referencia.

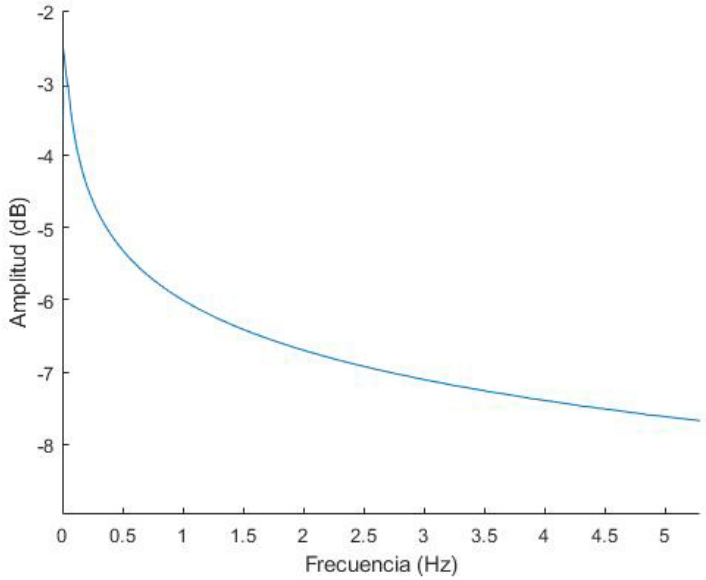


Figura 5.4: Amplitud en función de la frecuencia.

5.4. Control de flujo de aire

En un bandoneón, a mayor cantidad de botones oprimidos, mayor cantidad de válvulas abiertas y menor resistencia hidráulica. Algo similar ocurre con la palanca que abre una escotilla grande en relación a las válvulas individuales. Es necesario entonces diseñar un mecanismo capaz de emular todas estas situaciones, diferenciando tres estados:

- Ni palanca ni botón accionado.
- Al menos un botón presionado.

- Palanca presionada.

En el Bandonberry esto se logra con la combinación de una válvula con servomotor y una escotilla de liberación de aire accionada manualmente. Se plantea la solución con la combinación de palanca mecánica y servomotor, porque si solo se usar una servomotor, éste último debería tener más potencia para poder mover una escotilla grande que simule el efecto de la palanca.

5.4.1. Escotilla de liberación

Otra componente del control del flujo es la palanca que libera aire del fuelle. Si bien parece lógico resolver esto también con el servomotor, resulta que el flujo de aire al activar la palanca es mucho mayor que el producido por un botón, lo que da lugar a un disco giratorio de mayor tamaño y un ángulo mayor a barrer por el mismo. Esto provoca que el servomotor SG90 no sea suficiente para mover la estructura propuesta en la Figura 5.7 y se deba recurrir a uno de mayor costo, tamaño y consumo eléctrico.

Dado que estos últimos aspectos no son deseados y que el comportamiento de la palanca es simplemente una conmutación entre los estados abierto y cerrado, se decide resolver la palanca en forma mecánica, similar al del bandoneón, pero con piezas modeladas e impresas en 3D.

En la Figura 5.5 se puede ver el modelo realizado para la palanca basado en la del instrumento original.

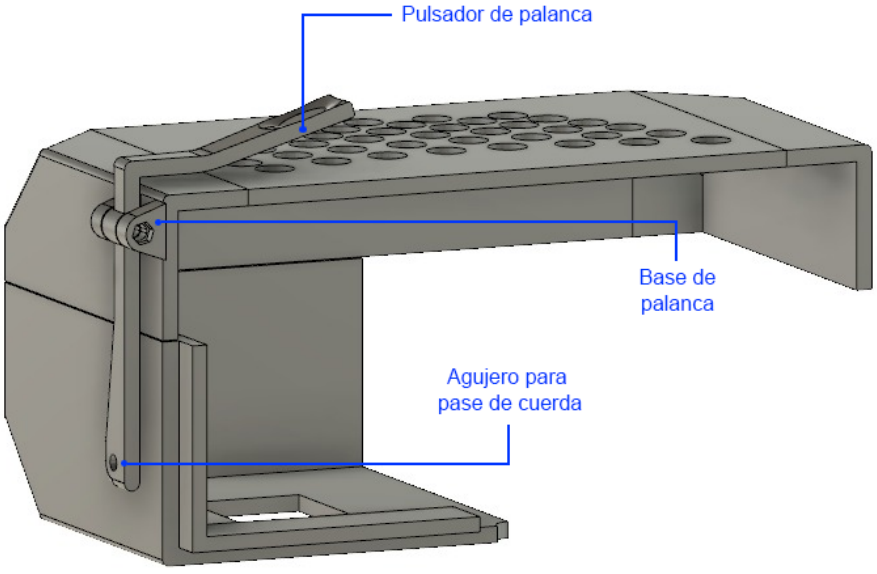


Figura 5.5: Modelo 3D de la palanca y escotilla.

La palanca tiene la función de tirar de un alambre, que mediante un sistema de bisagra y resorte levantan una tapa que libera la salida o entrada de aire en la base del fuelle. Si no se presiona la palanca, la tapa se mantiene contra la base,

obstruyendo la abertura gracias a un resorte. El resultado es una palanca que imita a la original, sin agregar complejidad al proyecto, teniendo una respuesta inmediata sobre el cambio del flujo y sin aportar consumo eléctrico.

5.4.2. Válvula con servomotor

Un servomotor es un motor eléctrico controlado digitalmente que posee la capacidad de girar y mantenerse en una posición estable. Además posee un sistema de engranajes que permite controlar la velocidad de giro y torque que puede realizar.

Para la construcción de una válvula de las características requeridas se utiliza uno pequeño, ya que con un mínimo torque es posible accionar un sistema que permite pasar el flujo de aire necesario. Se busca que el servo sea del menor tamaño posible a efectos de optimizar el uso de las baterías. Se utiliza un microservo Tower Pro SG90 [22] como se muestra en la Figura 5.6.

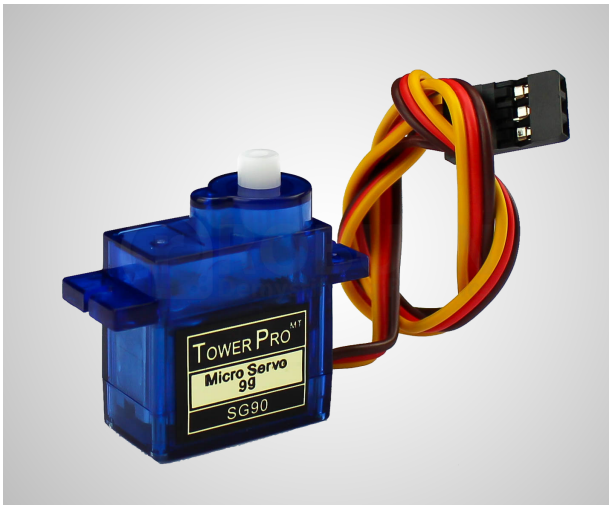


Figura 5.6: Micro servomotor Tower Pro SG90.

Se construye una sección de disco de centro en el eje del servo que puede girar respecto a la base del fuelle. En este último se recorta también una sección de disco de menor área, menor diámetro y concéntrico con el otro disco, de forma que al girar el disco solidario al servo este regule la apertura. El mecanismo se muestra en la Figura 5.7.

Variando el ángulo θ es posible regular la abertura en la base del fuelle y por tanto controlar el flujo de aire. Gracias a que la variable θ toma valores casi continuos es posible establecer y poder representar los estados que se necesite. Además, la variación del flujo de aire y de la variable θ resulta lineal, simplificando la tarea de emular múltiples botones presionados.

Respecto al control del servomotor, el mismo posee 3 cables de conexión: alimentación de 5 V, tierra y línea de control. Por esta última se envía una señal PWM de 50 Hz, donde el ciclo de trabajo indica el ángulo en el que se debe posicionar. En otras palabras, la señal tiene un período de 20 ms, donde el tiempo en

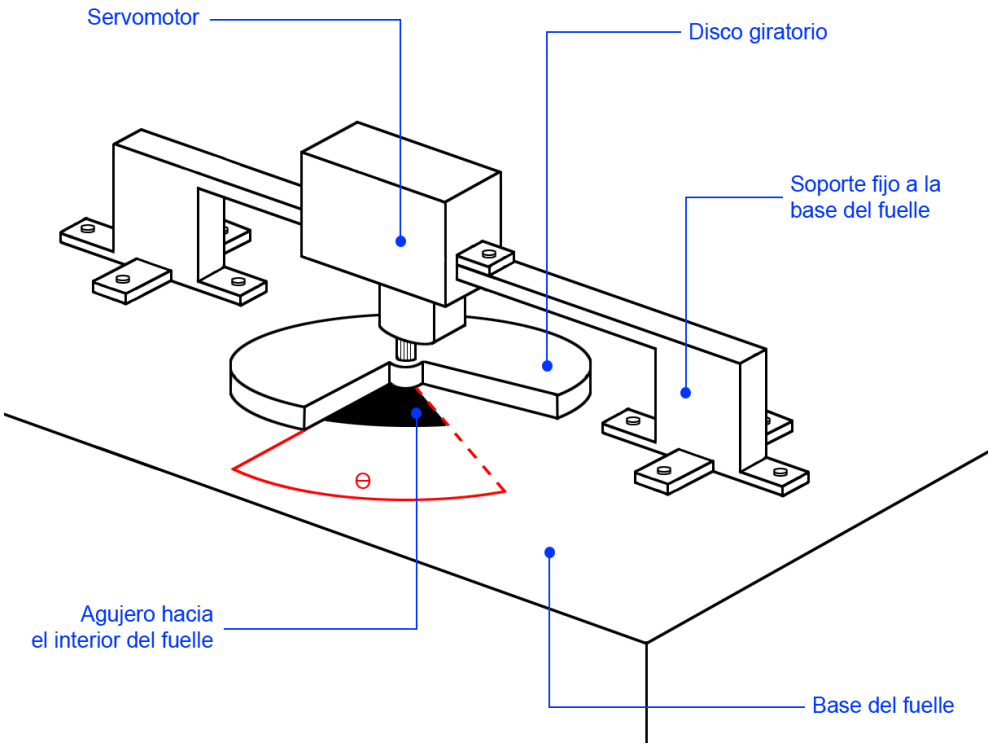


Figura 5.7: Esquemático representativo del mecanismo utilizado para el control del flujo de aire con un servomotor.

alto tiene una relación con el ángulo de giro (-90º para un pulso de 1ms, 0º para 1.5 ms, 90º para 2 ms y variando linealmente) y este es actualizado por el servo en cada período.

Desde la Raspberry Pi, existen muchas formas de generar las señales PWM. Una de ellas es escribir una rutina, en Python, que establezca el valor "1" y "0" lógico en un pin de salida. Se pueden regular las duraciones de cada valor utilizando el comando *sleep* [3]. Pero esta solución introduce dos problemas: primero, la precisión en la generación de la onda cuadrada no es buena, ya que el comando *sleep* tiene una resolución de 1 ms, lo cual no es suficiente teniendo en cuenta que el pulso esperado varía entre 1 y 2 ms. Segundo, durante el comando *sleep* el sistema operativo dedica recursos de CPU a otras tareas, por lo que no se puede asegurar con certeza la duración.

Por lo tanto, se decide utilizar la librería *RPIO.PWM* [12] que genera una señal PWM utilizando un canal DMA[4] presente en la Raspberry Pi. De esta forma se evita utilizar recursos del procesador, y se logra precisión de hasta 1 μs.

Se señala que lo ideal sería que el servomotor fuera capaz de moverse en función de la cantidad de botones presionados, pero se observó que con el sistema implementado no se logró generar una diferencia que fuera apreciable, abriendo

[3]Este comando suspende le ejecución del programa por la cantidad de tiempo definido
[4]Del inglés *Direct memory access*, memoria de acceso directo.

de forma parcial la válvula por. Por lo que se definió dos estados: o totalmente cerrado o totalmente abierto.

Dado que el programa *botoneras.py* es el que conoce el estado de los botones, se agregó el control del servomotor en el mismo. En este programa se lleva la cuenta de la cantidad de notas que están siendo reproducidas y se llama a una función que se encarga de llevar el servo a la posición correspondiente. Cuando una o más notas están siendo reproducidas, la válvula se abre completamente. Por el contrario, cuando ninguna nota debe ser reproducida, la válvula es cerrada.

Se debe tener en cuenta que el servomotor altera la relación entre movimiento y cambio de presión, debido a que demora en pasar de un estado a otro (de un cierto θ_1 a un θ_2). Este tiempo depende de la velocidad de giro, que tiende a ser la máxima siempre, teniendo un valor de 600 grados por segundo. Dicha velocidad es independiente de los grados a recorrer con la salvedad de considerar la inercia relativa al movimiento anterior. Sin embargo, el tiempo no solo depende de la velocidad de giro, sino también del ángulo a recorrer el cual no siempre es constante. Esta última dependencia genera un *delay* variable entre que se comienza el movimiento y la presión se estabiliza.

La excursión completa del servomotor lleva un tiempo de 300 ms. Este fenómeno no afecta el retardo de tiempo entre que se oprime el botón y se tiene la salida de audio, pero si afecta levemente la percepción de la modulación del volumen con respecto a la fuerza ejercida.

5.4.3. Bases

Para finalizar esta sección, los componentes faltantes para obtener un objeto hermético son las bases del fuelle, que forman parte de las carcasas. Es decir, son diseñados en conjunto con las mismas y son impresas en PLA. Ambas bases poseen en el centro un agujero con un pasacables para permitir la comunicación de cada una de las botoneras impidiendo la fuga de aire.

Además, la base derecha (ver Figura 5.8), se diseña con dos agujeros adicionales: uno para la tapa que se acciona con la palanca que libera aire; y el otro con forma de sección de disco según vimos en detalle en la sección del servomotor.

5.5. Construcción del fuelle

El método de construcción es simple y garantiza la hermeticidad necesaria. Con técnicas similares a como se realizan los origamis con papel plegado, basado en el modo de construcción de los fuelles de James Eckert [5], se crea un nuevo diseño. Mediante dobleces en una cartulina plastificada, se obtiene un fuelle similar al del bandoneón (que se realiza con cartones y tela pegados de forma especial).

El plastificado de la cartulina consiste en adicionar dos capas de papel *contact*, una de cada lado. Luego la cartulina es doblada para formar los pliegues y pegada

[5]El plano de pliegues corresponden al diseño “Pneumatic Power System 1” obtenido de `https://www.jeplans.com`.

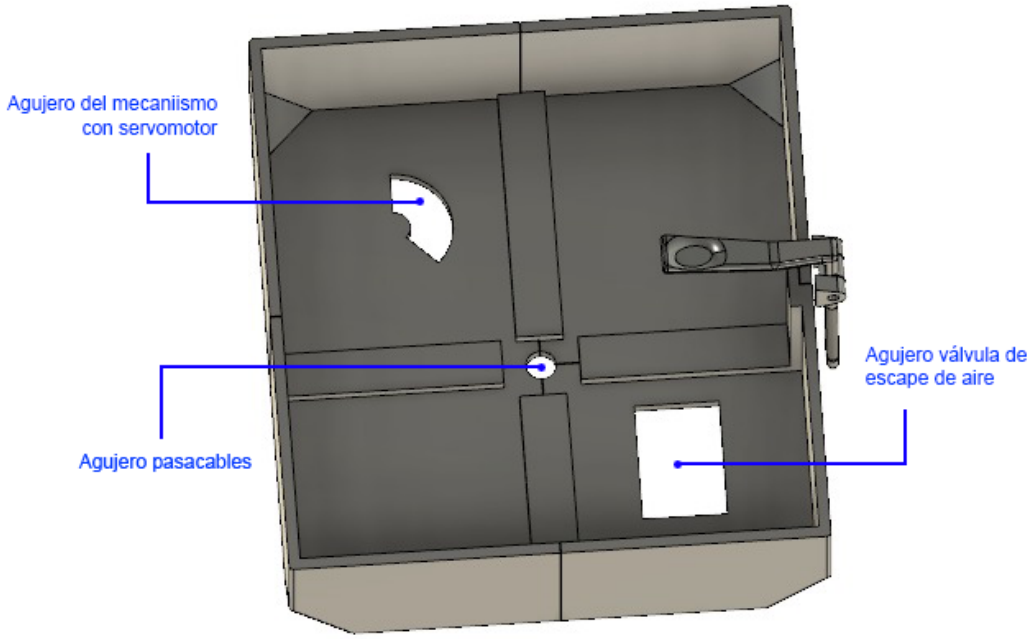
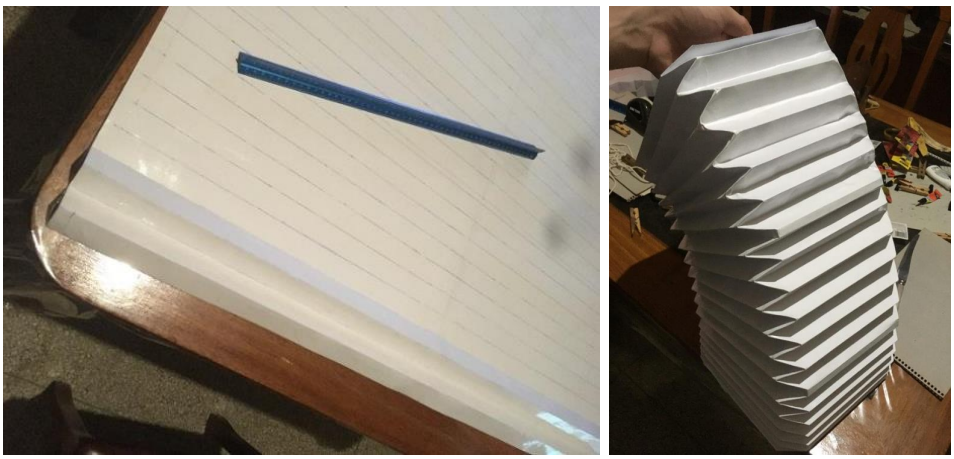


Figura 5.8: Modelo 3D de base derecha.

sobre si misma con adhesivo instantáneo a base de cianocrilato. A continuación se detallan características físicas específicas del fuelle fabricado:

1. Se construye a dimensiones reales.
2. Se incluyen marcos para reforzar el fuelle en las separaciones de los 3 cuerpos, en forma similar al original, mejorando la estructura de la pieza.
3. Se realiza en color blanco, para disimular el efecto del pegamento sobre el *contact* y el del desgaste de la cartulina sobre los bordes. Además logra una estética acorde con las carcasas del Bandonberry que fueron impresas con PLA blanco.



(a) Armado de los pliegues. (b) Fuelle extendido

Figura 5.9: Fuelle del Bandonberry.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 6

Battery Management System

En este capítulo se describe en profundidad el BMS. Se expone todo lo relacionado con el diseño e implementación de la alimentación del sistema, carga y gestión de la batería se interfaz de usuario.

6.1. Requerimientos

El BMS debe gestionar la alimentación de todo el sistema eléctrico, permitir al usuario encender y apagar el Bandonberry y medir el estado de las baterías. Este bloque interactúa con una fuente externa de donde extrae la energía para operar y recargar el sistema. Debe interactuar con el usuario a través de un *display* y un botón de encendido y apagado.

Se propone, que las baterías permitan como mínimo entre una y dos horas de autonomía de energía, en condiciones de consumo normales. Primero se hace un planteo teórico haciendo estimaciones y luego en el Capítulo 7 se explica las condiciones y mediciones para verificar el cumplimiento de este objetivo.

También, debe comunicarse con la SBC para conocer su estado y determinar el momento adecuado para habilitar o deshabilitar su alimentación. Si la alimentación de la SBC es interrumpida de forma abrupta, se corre el riesgo de corromper el estado de la memoria y dañar el funcionamiento del *software*.

Por último, el BMS, debe generar los niveles de voltaje necesarios para todos los otros módulos del sistema. Los módulos SBC, Audio y Servomotor se alimentan con 5 V, mientras que las botoneras y el sensor de presión con 3,3 V.

6.2. Arquitectura planteada

Esquemático general y presentación de los bloques

Para poder describir y exponer el sistema en su totalidad se divide el BMS en 4 bloques (ver Figura 6.1): Selector, Baterías, Microcontrolador e Interfaz de usuario. A su vez dentro del bloque del Selector, se subdivide en 3 módulos: Cargador y protecciones, Medidor de carga y Reguladores de voltaje.

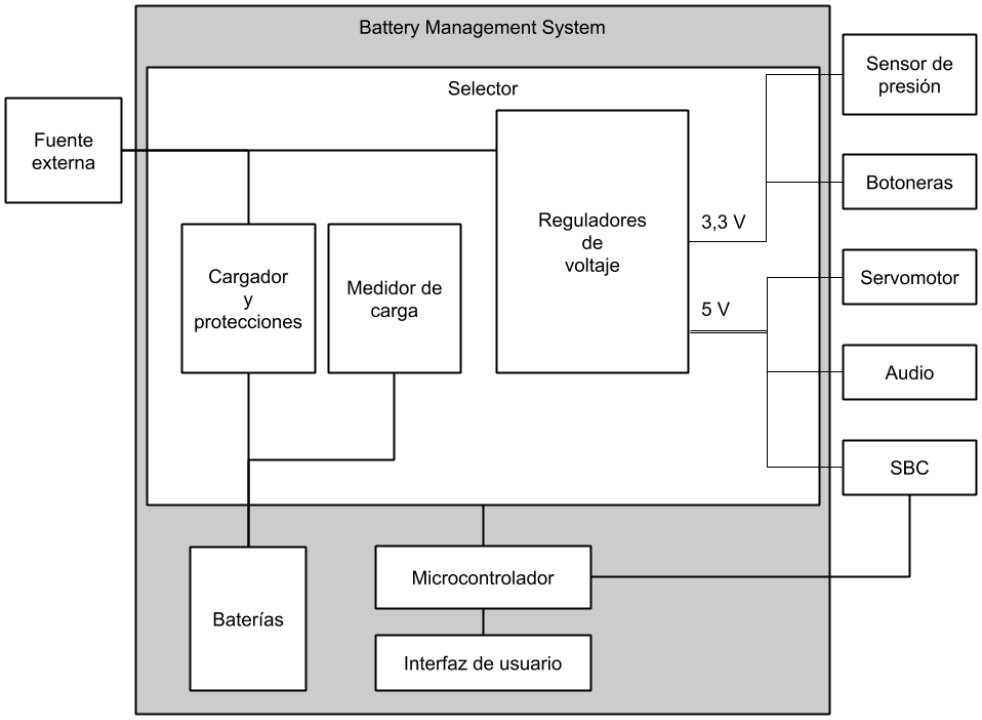


Figura 6.1: Diagrama de bloques de arquitectura general del BMS.

El módulo Selector gestiona la entrada y distribución de la energía en el sistema. Posee entrada para una fuente externa de 5 V, es capaz de cargar las baterías y generar los distintos niveles de voltaje necesarios para alimentar el resto del sistema. Además, el Selector cuenta con las protecciones contra sobrecargas, sobredescargas y cortocircuitos asociadas a la batería.

La batería está formada por cuatro celdas de polímero de litio (LiPo) de 3,6 V nominales.

El Microcontrolador, un Arduino Pro Micro, es el encargado de gestionar el Selector. Toma decisiones sobre qué partes del sistema deben recibir energía, asegurando la correcta secuencia de encendido y apagado. Por último, se tiene una Interfaz de usuario, la cual posee un *display* donde se brinda información del estado de la batería y de la SBC.

Funcionamiento general

Partiendo del sistema apagado, el usuario oprime el botón de encendido-apagado en la Interfaz de usuario. El microcontrolador es el primer bloque en ser energizado. Este evalúa el estado del sistema y de encontrarse en condiciones adecuadas, envía la orden al modulo Selector de alimentar el resto del sistema.

Una vez alimentado el sistema, el microcontrolador despliega en la pantalla de la Interfaz de usuario el proceso de encendido. Cuando la inicialización del sistema operativo se ha finalizado, se despliega en la pantalla el estado de la batería.

Habiéndose llegado a esta instancia, el sistema esta listo para utilizarse de forma normal.

Para apagar, se oprime nuevamente el botón de encendido-apagado y el Microcontrolador envía la señal de apagado a la SBC. Una vez que indique que es seguro des-energizar, se corta la entrada de alimentación a todo el sistema.

El Selector da preferencia a la fuente externa (estando conectada) en lugar de la batería. El proceso de carga, es independiente del estado (encendido o apagado) del Bandonberry.

6.3. Interfaz de usuario

En la interfaz de usuario, ver Figura 6.2, cuenta con un *display* que brinda información del estado del sistema, de la batería y eventuales errores críticos. También se tiene un pulsador de encendido-apagado y un *switch* de corte de alimentación de las baterías. Su función es desconectar la batería del resto de forma manual, por motivos de seguridad y testeo. También se tienen LEDs [1] que indican si la batería se está cargando, si finalizó la carga y si el Bandonberry está encendido.

Figura 6.2: Interfaz de usuario.

El LED de encendido indica la presencia de voltaje en la línea de 3,3 V, esto permite visualizar que el sistema está energizado, independientemente de que el *display* esté mostrando, o no, información.

Se tiene un interruptor que corta la alimentación de las baterías para poder desacoplarlas en caso de tener la necesidad de cortar la alimentación del sistema de forma completa.

El *display* es de tecnología OLED[2] de 128x32 *píxeles*, con conexión I2C [4] (Ver Figura 6.3.). Este tipo de *display* es un dispositivo fácil de encontrar en el mercado, tiene un tamaño adecuado para su función, es económico, y tiene a

[1] Del inglés *Light-Emitting Diode*, diodo emisor de luz.
[2] Del inglés Organic Light Emitting Diode, diodo emisor de luz orgánico

disposición librerías de software libres para Arduino. Las librerías utilizadas son todas de código abierto, brindadas por su fabricante, Adafruit [3].

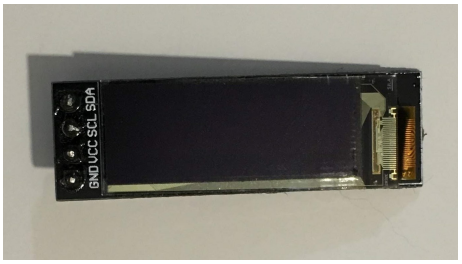


Figura 6.3: *Display* OLED de la interfaz de usuario.

6.4. Batería

Las celdas de la batería del Bandonberry se eligieron con la química de litio-ion, ya que, son accesibles, confiables y tienen una gran variedad de ofertas diferentes en el mercado, tanto en marcas como en presentaciones. Las baterías de litio-ión generalmente vienen en celdas, con un voltaje nominal entre 3,6 y 3,7 V. Sin embargo, la capacidad de almacenamiento (medida en mWh) varía según la calidad y el fabricante.

Se define el factor C como la corriente máxima que se puede drenar de una batería completamente cargada en una hora [25]. El fabricante generalmente da los datos de la *potencia.hora* nominal y voltaje nominal definiéndose la capacidad nominal:

$$C = \frac{potencia.hora_N}{V_N} = I_N.hora \tag{6.1}$$

Para estimar los tiempos de descarga se utilizan las curvas características de descarga, proporcionadas por el fabricante. En dichas curvas, se utiliza el factor C para diferenciar distintas condiciones de ensayo, ya que a diferente demanda de corriente se tiene una curva diferente.

En la etiqueta de las celdas utilizadas en el Bandonberry no se tiene información sobre el fabricante. Se toma como referencia información de una celda del mismo tamaño, capacidad y química del modelo LIR18650. La curva de descarga de dicha batería se muestra en la Figura 6.4 y otros datos de interés en la Tabla 6.1.

Se aprecia que el voltaje tiende a variar muy lentamente cuando las demandas de corriente son pequeñas. Al descargarse más de un 80 %, el voltaje decae rápidamente. La curva termina en el voltaje denominado "voltaje de corte", en inglés *"cut off voltage"*. Cabe señalar que no se recomienda tener una carga menor al 20 % de la carga total, ya que la batería se encuentra próxima al voltaje de *cut off*.

[3]Las librerías se pueden encontrar en los repositorios `https://github.com/adafruit/Adafruit-GFX-Library` y `https://github.com/adafruit/Adafruit_SSD1306`

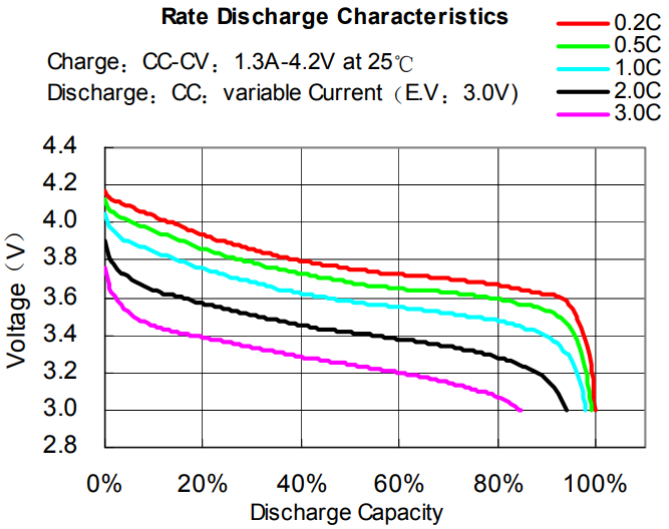


Figura 6.4: Curvas características de descarga de la batería LIR16650 2.600 mAh, extraído de [10].

Especificación	**Valor (A)**
Corriente de carga estándar	0.52
Corriente rápida de carga	1.30
Corriente de descarga estándar	0.52
Corriente de descarga rápida	1.30
Máx. pulso de corriente de descarga	2.60

Tabla 6.1: Valores de la batería LIR18650 2.600 mAh, extraído de [10].

El módulo de carga, mostrado en la Sección 6.5.1, posee las protecciones necesarias de sobrecarga y sobredescarga, las cuales aseguran el correcto uso de las baterías.

6.4.1. Consumo y capacidad de las baterías

En primer lugar se estima el consumo esperado del sistema para calcular la capacidad de las baterías. Para el cálculo, se asumen las siguientes hipótesis:

1. Los módulos de los reguladores de voltaje, necesarios para crear las líneas de 3,3 V y 5 V posee una eficiencia del 90 %.

2. Los módulos de sonido tienen un consumo de 3 W cada uno estando a máxima potencia, por lo que se tiene 6 W en total [16].

3. Se consideran los consumos estimados de los módulos restantes obtenidos de sus respectivas hojas de datos, tal como se muestra en la Tabla 6.2.

4. Se considera que los componentes del sistema que no se encuentran en la

Tabla 6.2, tienen un consumo despreciable, ya que son resistencias y transistores en configuraciones de llaves que están abiertas o cerradas.

Dispositivo	**Corriente (mA)**	**Potencia(mW)**
SBC	140	700
Audio	1200	6000
Servomotor	550	2750
Microcontrolador	15	75
Interfaz de usuario	20	100
Total	**1925**	**9625**

Tabla 6.2: Consumos de corriente y potencia considerando un voltaje de 5 V.

Tomando la potencia total de la Tabla 6.2 y considerando la hipótesis 1, se aplica el factor de eficiencia de 90 % para llegar a la potencia efectiva demandada:

$$P_D = 9,625\ W \tag{6.2}$$

$$P_{EffD} = 10,588\ W \tag{6.3}$$

Por otro lado, considerando 3,6 V como el voltaje promedio de la batería, se calcula la corriente máxima efectiva demandada promedio:

$$I_{MaxEffD}@3,6\ V = \frac{10,588\ W}{3,6\ V} = 2,94\ A \tag{6.4}$$

En el caso de las baterías utilizadas para el Bandonberry se tienen celdas con las siguientes características nominales en sus etiquetas:

$$3,6\ V \qquad 9,0\ Wh \tag{6.5}$$

Considerando un voltaje constante nominal, es decir que no se está en las cercanías de los extremos de la curva de la Figura 6.4, se tiene que:

$$C = \frac{9\ Wh}{3{,}6\ V} = 2500\ mAh \tag{6.6}$$

En las etiquetas de los fabricante generalmente se muestra la potencia por unidad de tiempo máxima teórica total. Por este motivo se considera un 30 % menos del dato de etiqueta, para proteger la batería, por los motivos antes explicados.

Por lo tanto se considera una capacidad, por celda, de:

$$C_{celda} = 1750\ mAh \tag{6.7}$$

Si se conectan celdas en paralelo las capacidades se suman ya que las corrientes se suman. Para que el Bandonberry sea capaz de funcionar por al menos 2 horas es necesario una capacidad C_{Total} tal que:

$$C_{Total} > I_{MaxEffD} \times t = 2{,}94\ A \times 2\ h = 5880\ mAh \tag{6.8}$$

Por lo tanto se precisa una cantidad N de celdas tal que:

$$N > \frac{5880\ mAh}{1750\ mAh} = 3{,}36 \Rightarrow N = 4 \tag{6.9}$$

Entonces, para satisfacer dicha carga, se utilizan cuatro celdas conectadas en paralelo. Por lo que se tiene una batería resultante con una capacidad de carga C = 7000 mAh.

6.5. Selector

Se llama Selector a toda la electrónica que tiene la función de controlar las líneas de entrada de alimentación del sistema. El esquemático del circuito se muestra en la Figura 6.5.

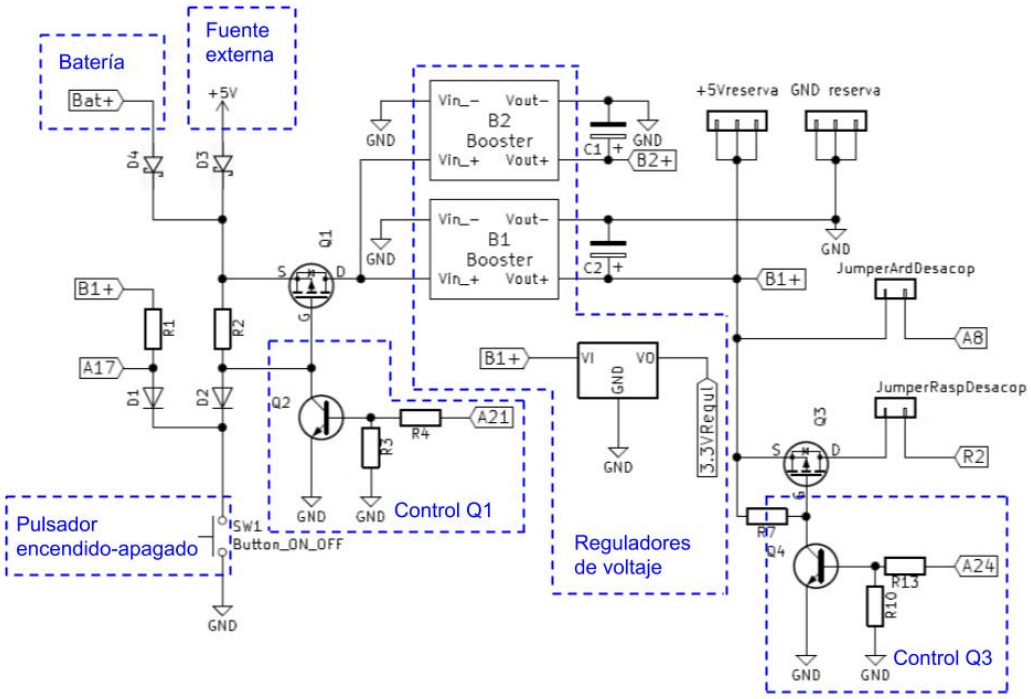


Figura 6.5: Esquemático del módulo Selector y Reguladores de voltaje.

Los diodos D3 y D4 son de tipo schottky tiene la función de seleccionar la fuente de alimentación. En el caso de tener la fuente conectada, las baterías tienen un voltaje nominal de 3,6 V y el suministro de alimentación es una fuente externa de 5 V, por lo que se tiene que por dicha diferencia de potencial el diodo D3

conducirá y el D4 no. Si la fuente no está conectada, el diodo D4 conducirá y el diodo D3 no. Se va a considerar que la caída de voltaje del diodo es de 0.5V con un pasaje de 3A [17]

Una vez activado el diodo correspondiente, la corriente pasa al transistor Q1 (NDP6020P). Dicho transistor maneja la entrada de energía de todo el sistema. Si el pulsador *Button ON_OFF* está abierto y el transistor Q2 (BC337) no conduce, los nodos V_{G1} y V_{S1} de Q1 están al mismo potencial debido a la presencia de la resistencia R2. Esto implica que el sistema se encuentra sin alimentación.

Al oprimir el pulsador, la corriente fluye por el diodo D2 y se genera en él una caída de 0,7 V. Se tiene entonces que $V_{S1} = 4,5V$ con fuente externa, o V_{S1} = 3,1 V con baterías, y V_{G1} = 0,7 V. De esta manera se tiene que el menor V_{GS1} = -2,4 V.

Según la hoja de datos del fabricante [24], para que el transistor Q1 se encuentre en zona de conducción debe cumplir:

$$2V < V_{GS1} < 4V \tag{6.10}$$

Por lo tanto el transistor Q1 cuenta con la polarización necesaria para alimentar el sistema. La elección del transistor Q1 se detalla en el Anexo A.

Suponiendo que el transistor Q1 es equivalente a una llave cerrada, el bloque de Reguladores de voltaje comienza a operar. El microcontrolador, el servomotor, el módulo de audio, el sensor de presión y las botoneras reciben el voltaje necesario para funcionar.

El microcontrolador tiene un inicio muy rápido, en relación a la velocidad en la que una persona oprime el botón, por lo que su reacción del mismo se percibe de forma instantánea. De esta manera el microcontrolador, como primer comando en su programación, lleva a 5 V el nodo A21. Se polariza el transistor Q2. El mismo, entra en saturación (V_{CE2} = 0,3 V) y de esta manera el transistor Q1 puede mantener su polarización. El sistema queda alimentado hasta que el microcontrolador baje el voltaje de A21 y despolarice el transistor Q2 (suponiendo que el botón de encendido no esta oprimido).

Para dar la orden de apagado al sistema, se oprime nuevamente el pulsador *Button ON_OFF*. Con el sistema encendido, el nodo V_{B1+} = 5 V y suponiendo que *Button ON_OFF* no se encuentra presionado, la resistencia R1 lleva el nodo A17 a 5 V (nivel lógico HIGH). Con el *Button ON_OFF* presionado, fluye corriente por el diodo D1 y se genera en él una caída de 0,7 V (nivel lógico LOW). El pin A17 es una entrada lógica del microcontrolador y de esta manera se puede detectar si se oprime el botón de apagado estando encendido el equipo.

Cuando se oprime el botón de apagado, el microcontrolador espera la señal de la Raspberry, que indica que esta preparada para apagarse y baja el voltaje del pin A21, apagando el sistema.

El bloque compuesto por los componentes de Control Q3, tiene el mismo funcionamiento que los de Control Q1. El objetivo del bloque es permitir controlar, a través del nodo A24, la alimentación de la SBC. El microcontrolador verifica el correcto estado del sistema antes de energizar la SBC.

Los *jumpers* de desacople, brindan la posibilidad de aislar ciertas partes del circuito para realizar eventuales modificaciones, verificaciones o mediciones eléctricas.

6.5.1. Cargador y protecciones

Para estas funcionalidades se utiliza el módulo mostrado en la Figura 6.6, las conexiones específicas están en Guithub. Se elige este módulo por su sencillez, su accesibilidad en el mercado y porque tiene la particularidad de tener las protecciones incluidas. Posee dos circuitos integrados, el TP4056 [9] responsable del proceso de carga y el DW01A [13] encargado de las protecciones. En el módulo se encuentra también un transistor MOSFET doble (8205A [14]) que permite la desconexión automática de la batería en caso de encontrarse fuera de los parámetros normales de funcionamiento.

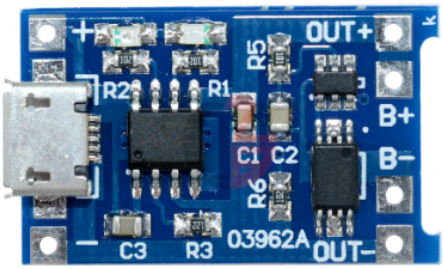


Figura 6.6: Imagen del cargador de 1 A para baterías de litio.

El integrado TP4056 es capaz de cargar baterías de litio-ion con corrientes de hasta 1 A. La curva de carga consta de una fase de corriente constante, seguida de una fase de voltaje constante. Esta curva se muestra en la Figura 6.7.

Si se considera el tiempo de 2 horas para cargar una batería de 1.000 mAh, y se supone una relación lineal entre tiempo y carga, se tiene que para cargar 4 celdas de 2.500 mAh, se precisan en el orden de 20 horas de carga. Para bajar el tiempo de carga se plantea una solución diferente mostrada en el Anexo E. Por otro lado, podemos decir que el cargador tiene una demanda de potencia máxima de 4,25 V @ 1 A = 4,25 W aproximadamente (sin considerar factores de eficiencia del cargador). Entonces, la fuente externa debe tener una potencia mayor o igual a la potencia máxima que consuma el cargador más la potencia máxima del sistema. Es decir:

$$P_{fuente} > P_{MaxCargador} + P_{MaxSistema} \approx 15\ W \tag{6.11}$$

Con respecto a las protecciones eléctricas, el integrado DW01A ofrece protección contra:

- Sobrecarga $V_{Bat} > 4,3$ V
- Sobredescarga $V_{Bat} < 2,4$ V

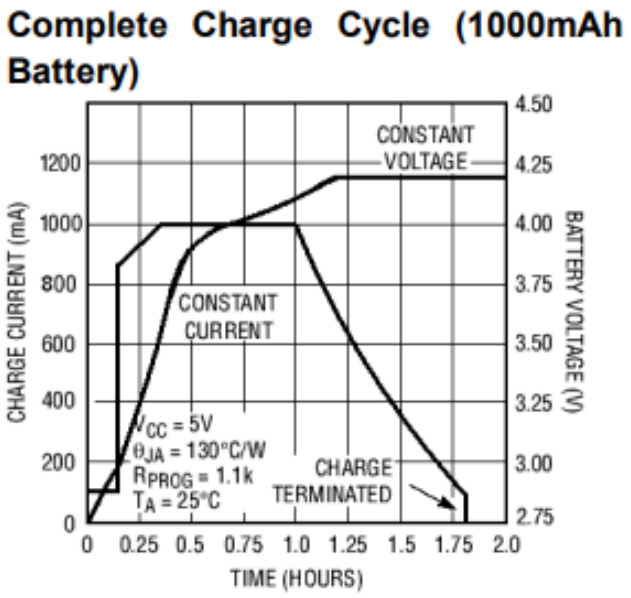


Figura 6.7: Curvas de carga del integrado TP4056, extraído de [9].

- Sobrecorrientes

El circuito asociado al integrado DW01A se muestra en la Figura 6.8. El circuito es capaz de detectar la corriente que circula por la batería a través de la caída de tensión en los MOSFET M1 y M2. De la hoja de datos del MOSFET 8205A, se tiene una R_{DSon} con un valor de 30 mΩ @ V_{GS} = 2,5 V y 21 mΩ @ V_{GS} = 4,5 V. Según la hoja de datos del DW01A, una caída de V_{CS} = 150 mV activa la protección de sobrecorriente, por lo tanto se puede saber la corriente a la que el sistema desconecta la batería en función de las R_{DSon} de los transistores.

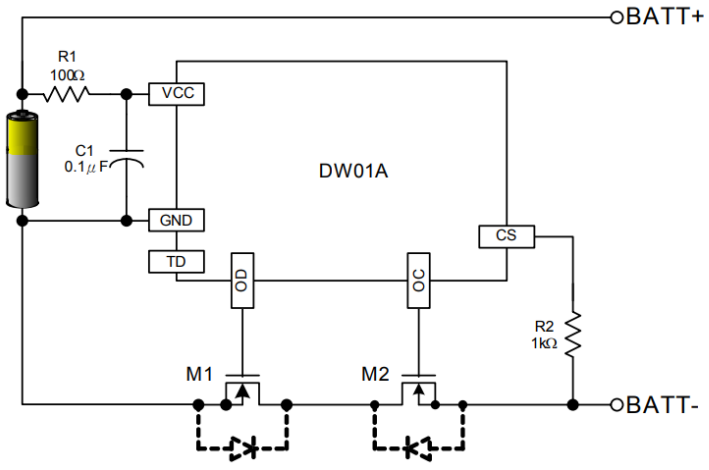


Figura 6.8: Circuito de protección de batería, extraído de [13].

$$I = \frac{150\ mV}{2 \times 30\ m\Omega} = 2,5\ A@V_{GS} = 2,5\ V \tag{6.12}$$

$$I = \frac{150\ mV}{2 \times 21\ m\Omega} = 3,6\ A@V_{GS} = 4,5\ V \tag{6.13}$$

El voltaje V_{GS} depende del estado de carga de la batería. Si se interpolan los resultados considerando el voltaje nominal de 3,6 V, la corriente máxima que deja pasar el módulo es de 3,1 A.

6.5.2. Reguladores de alimentación

Regulador de 5 V

El objetivo de este módulo es generar 5 V a partir de una entrada de voltaje variable. En el caso de estar alimentándose desde la batería, el voltaje de entrada al sistema puede variar desde 4,2 V a 2,5 V. En el caso de estar conectado a la fuente externa, el voltaje a la entrada del sistema es 5 V por lo que no es necesario hacer ningún tipo de conversión.

Un módulo capaz de generar un voltaje mayor al de su entrada en corriente continua, es conocido como *booster* o *step-up DC-DC converter*. De las opciones disponibles en plaza, se selecciona un *booster* capaz de entregar los 5 V y 2 A requeridos. En la Figura 6.9 se observa el *booster* utilizado en el Bandonberry.

Figura 6.9: Imagen del *booster*.

En la Tabla 6.3 se muestra el voltaje y corriente de entrada y salida del *booster*, así como su eficiencia. Según el fabricante [6], el *booster* tiene una corriente máxima de salida de 2 A, por lo que entrega una potencia máxima de 10 W.

En la tabla, el fabricante varía el voltaje y corriente de entrada para tener una eficiencia aproximadamente constante. Pero en los medidas de 3,6 V, 3,5 V y 3,4 V se tiene una corriente de entrada constante de 2,2 A y la eficiencia disminuye en relación al voltaje de entrada.

El sistema tiene una demanda máxima estimada de 9,625 W tal como se mostró en la Sección 6.4. Como dicho valor es muy próximo a la capacidad máxima del *booster*, se utilizan dos que alimentan dos líneas de poder independientes.

Ambos *boosters* comparten la referencia a masa de todo el sistema, pero alimentan a dos grupos distintos de módulos. A uno de ellos se conectan los dispositivos con mayor consumo, y por ende, los más ruidosos desde el punto de viste eléctrico;

Input		**Output**		**Eficien.(%)**
Voltaje (V)	Corriente (A)	Voltaje (V)	Corriente (A)	
5.0	2.50	5.00	1.98	79.2
4.9	2.59	5.00	2.00	78.8
4.8	2.62	5.00	1.99	79.1
4.5	2.53	5.00	1.83	80.4
4.3	2.60	4.98	1.73	77.1
4.2	2.66	4.95	1.73	76.7
3.7	2.37	4.93	1.36	76.5
3.6	2.20	4.98	1.23	77.3
3.5	2.20	4.97	1.14	73.6
3.4	2.20	4.93	1.12	73.8
3.3	2.20	4.90	1.00	67.5
3.2	1.86	5.00	0.92	77.3
3.1	2.00	4.96	0.92	77.3
3.0	2.00	4.96	0.96	79.4
2.9	2.00	4.93	0.81	68.9
2.8	1.60	5.00	0.75	83.7
2.7	1.96	4.92	0.78	72.5
2.6	1.67	4.99	0.63	72.4

Tabla 6.3: Tabla de datos de eficiencia de los *boosters*, extraído de [6].

estos son el servomotor y el audio. Por otro lado, el otro *booster* alimenta a los componentes más sensibles y de menor consumo; estos son la SBC, el microprocesador, la interfaz de usuario y el sector de 3,3 V.

En la Sección 6.4 no se consideró la caída de voltaje en los diodos Schottky y se estimó además una eficiencia del 90 %, pero observando en la Tabla 6.3 y considerando que el voltaje efctivo nominal de la batería es de 3,1 V, se tiene una eficiencia de 77 %. Entonces, la nueva potencia efectiva demandada a la batería es:

$$P_{EffD} = 12,5\ W \tag{6.14}$$

Para poder extraer dicha potencia de la batería debe suministrar:

$$I_{Bat} = \frac{12,5\ W}{3,1\ V} = 4,03\ A \tag{6.15}$$

Dicha corriente se encuentra por encima de lo contemplado por las protecciones, por lo que será necesario cambiar las protecciones o limitar el consumo de alguno de los componentes, como por ejemplo acotar la potencia de la salida de audio para que la corriente máxima baje. La configuración final se define en la Sección 7.3.

Filtrado de transitorios

Se conectó un capacitor a la entrada del *booster* que alimenta el servomotor, para filtrar los picos de corriente demandados. Se consideró como peor caso la demanda del servomotor, ya que es el componente con los picos de corriente más grandes en amplitud. Se observó caídas de tensiones de casi 1 V en las demandas de corriente del servomotor. Para evitar dicho problema y Considerando que la duración del pico de corriente del servomotor es del orden de los 20 ms (se midió el ancho del pulso de la demanda de corriente del motor integrado en el sistema), tenemos una frecuencia fundamental de 50 Hz. La menor resistencia vista por el *booster* es 5 V / 2 A = 2,5 Ω. Si consideramos que en el momento de la demanda máxima la impedancia vista hacia el motor es considerablemente más pequeña que el resto del sistema (que está conectado en paralelo con el booster), se puede aproximar que la impedancia vista por la salida del booster es aproximadamente la impedancia del motor. Se busca que el polo del filtro se encuentre al menos antes de los 50 Hz, entonces:

$$C > \frac{1}{2\pi \cdot 2,5\ \Omega \cdot 50\ Hz} = 1273\ \mu F \tag{6.16}$$

Lo ideal sería que el capacitor esté una década por debajo de los 50 Hz, pero utilizando 5 Hz la ecuación genera capacitores del orden de las decenas de mili Faradios. Esto compromete la propia demanda de corriente del capacitor. Es decir, si el capacitor es muy grande, demanda mucha corriente para poder cargarse, por lo que puede tener un resultado totalmente opuesto al buscado. Por lo que se fueron probando capacitores hasta lograr el más cercano a los 1.273 μF, donde no se genera inestabilidad en el encendido del sistema. Finalmente, el capacitor elegido es de 1.000 μF.

A su vez se agregan capacitores a la salida de los ambos *boosters*, para mejorar la estabilidad de la línea de alimentación. Dichos capacitores se agregan con el mismo criterio antes mencionado. Finalmente se utilizaron capacitores de 470 μF.

Regulador de 3,3 V

Para alimentar el sensor de presión y las botoneras es necesario contar con 3,3 V. Para eliminar el problema de la variabilidad de voltaje de entrada generado por la descarga de la batería, el regulador se conecta a las líneas de 5 V generadas por los *boosters*.

Debido a la baja carga y poca caída de tensión necesarias, se utiliza un regulador lineal. En concreto, el integrado AMS1117V [5], mostrado en la Figura 6.10. Puede entregar hasta 1A de corriente que para las cargas mencionadas es más que suficiente.

Figura 6.10: Regulador de voltaje de 3,3 V.

6.5.3. Medidor de baterías

Medir el *sate of cahrge* (SoC [4], de una batería es considerablemente complejo. Las curvas de la batería son sensibles a la temperatura, a la demanda de corriente que genera la carga, a los ciclos de carga y descarga que haya tenido la batería, y también al pasaje del tiempo. No es posible medir la cantidad de carga que hay en una batería de forma exacta, sin extraer dicha carga de la misma. Por lo tanto el SoC se calcula en función de una medida y una estimación a partir de ensayos previos. Todas estas variables hacen que el algoritmo y mediciones necesarias para poder estimar el SoC no sea sencillo.

La primera aproximación que se puede realizar es medir el voltaje en bornes de la batería. Si se parte de un estado no conocido y un voltaje, como por ejemplo 3,75 V, y teniendo una batería cuyas curvas son como la de la Figura 6.4, no es posible definir en qué parte del intervalo del SoC se encuentra el sistema. Por lo tanto para poder realmente estimar el SoC es necesario tener una memoria del estado inicial.

Se distinguen entonces dos formas de realizar esta tarea:

1. Utilizar circuitos integrados diseñados específicamente para esta tarea.
2. Medir el voltaje en bornes de la batería, con una entrada analógica del microcontrolador. Comparar dicho valor contra la curva característica de la batería partiendo de una condición inicial.

La opción 1 es la utilizada en el Bandonberry dado que es considerablemente más sencilla en su implementación y debido a la popularidad de las baterías de lítio-ion este tipo de solución es fácil de conseguir en plaza. Por otro lado, la opción

[4]Del inglés *State of Charge*), estado de carga (asociado a una batería).

2 requieren de mayor tiempo de investigación y desarrollo no estando contemplados en el alcance.

El integrado utilizado es el MAX17043 [15]. En la Figura 6.11 se observa el módulo llamado *Fuel Gauge* por la empresa que lo comercializa, Sparkfun [5].

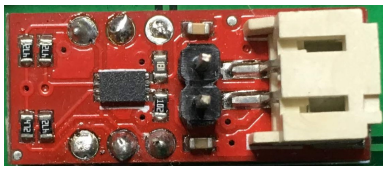


Figura 6.11: Módulo MAX17043.

En la Figura 6.12 se muestra un esquema de bloques del interior del integrado MAX17043. No se cuenta con la información para saber con exactitud como funciona, ya que el fabricante no tiene su código abierto. De todas maneras en la hoja de datos se tiene una explicación en términos generales del funcionamiento. Básicamente el sistema es un medidor de voltaje, el cual va tomando muestras y a medida que pasa el tiempo va generando un histórico, que se compara con un modelo esperado de batería (controlado con una máquina de estados).

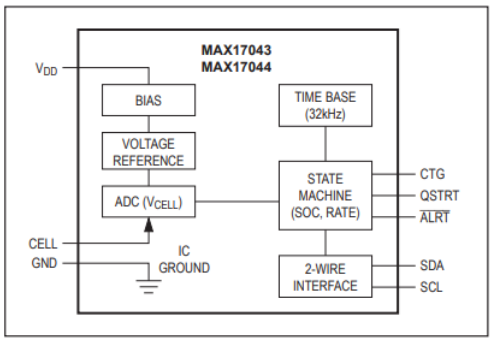


Figura 6.12: Diagrama interno del integrado MAX17043, extraído de [15].

Según el fabricante, a medida que pasa el tiempo el algoritmo converge, por lo que la primera medición puede generar un resultado del estado de carga poco preciso, pero a medida que se van tomando muestras a los largo del tiempo, converge a la medida correcta. Puede llegar a ser necesario realizar una carga y descarga completa de la batería para poder converger de forma correcta.

Finalmente debemos mencionar que el integrado se comunica a través de I2C con el microcontrolador. Tiene la posibilidad de manejar ciertas señales de alarma configurables. No son utilizadas en el Bandonberry, por lo que solo se utilizarán las señales del protocolo I2C. Para toda la comunicación y manejo del módulo existen librerías de Arduino de código abierto[6].

[5] Para más información consultar `https://www.sparkfun.com/`.

[6] Las librerías utilizadas se encuentran en Guithub `https://github.com/lucadentella/ArduinoLib_MAX17043`.

6.6. Microcontrolador

Las funciones del microcontrolador son: medición de la batería, lectura del estado del botón de encendido, actualización de la información desplegada en el *display* y supervisión del encendido y apagado de la SBC. Se utiliza el Arduino Pro Micro, debido a su alta popularidad, accesibilidad, bajo costo y alta eficiencia en lo que respecta al manejo de sistemas de control.

Un microcontrolador, separado de la SBC, permite que las funciones básicas de control puedan ser ejecutadas con rapidez. La SBC requiere de aproximadamente un minuto para terminar su proceso de inicialización, mientras que el Arduino tiene un inicio instantáneo para la percepción humana. Por otra parte se liberan recursos de la SBC, que permiten bajar los valores de latencia, lo cual es fundamental para tener una buena experiencia del instrumento.

Para prescindir del microcontrolador, se debería recrear las funciones realizadas con componentes electrónicos discretos, como por ejemplo compuertas lógicas y amplificadores operacionales. Esto agregaría complejidad y cantidad de componentes de forma considerable. Probablemente los costos de todos los componentes necesarios, no logren ser mucho menores al del Arduino.

Figura 6.13: Imagen de Arduino Pro Micro.

6.6.1. Conexionado y funciones

En la Tabla 6.4 se muestran las conexiones del microcontrolador [7].

Tanto el *display* como el *Fuel Gauge* comparten el bus I2C del microcontrolador, las señales SDA y SCL. Las direcciones asociadas a los dispositivos I2C son 0x3C para el *display* y 0x36 para el *Fuel Gauge*.

El resto de las señales son utilizadas para la comunicación de los estados de la SBC y controlar transistores que funcionan como llaves de alimentación. A continuación se explica brevemente cada una de ellas:

- **A17 - ShutDownBot_IN:** Señal que mide el estado del botón de encendido-apagado (activa por nivel bajo).

[7] No confundir el nombre de la columna pin que se asocia A1 como pin 1 del componente Arduino, con la referencia de la columna función A1 que se asocia a pin analógico 1.

Uso	Función	Pin	Pin	Función	Uso
	TX/D0	A1	A2	RAW	
	RX/D1	A3	A4	GND	
	GND	A5	A6	RESET	
	GND	A7	A8	Vcc(5V)	
SDA_FuelGauge_Display	SDA/D2	A9	A10	A3	
SCL_FuelGauge_Display	SCL/D3	A11	A12	A2	
	A6/D4	A13	A14	A1	
	D5	A15	A16	A0	
ShutDownBot_IN	A7/D6	A17	A18	D15/SCLK	
RaspState_IN	D7	A19	A20	D14/MISO	
PowerOnOff_OUT	A8/D8	A21	A22	D16/MOSI	ROn_OUT
RaspOff_OUT	A9/D9	A23	A24	D10/A10	

Tabla 6.4: Conexión de los pines del microcontrolador.

- **A21 - PowerOnOff_OUT**: Enciende el transistor principal de potencia (Q1 de la Figura 6.5), alimentando todo el sistema. Nivel alto equivale a encender todo el sistema. Nivel bajo, corta la corriente de todo el sistema.

- **A19 - RaspState_IN**: Señal de estado de la SBC. Cuando la señal está en nivel alto indica que la SBC está operando y no es seguro desconectar la alimentación. Si la señal está en nivel bajo la SBC se puede des-energizar de forma segura.

- **A23 - RaspOff_OUT**: Señal que indica a la SBC que comience el proceso de apagado.

- **A22 - ROn_OUT**: Señal para manejar la alimentación de la SBC. Un nivel alto habilita la alimentación de la SBC y un nivel bajo desconecta la energía.

6.6.2. Programación

Por defecto, el Arduino se encuentra en bucle consultando al *Fuel Gauge* el estado de la batería y desplegándolo en el *display*. A su vez, está controlando que el voltaje de la batería no sea inferior a cierto umbral, el cual se define en la Sección 7.4.3. Si el voltaje de la batería es inferior, el Arduino enviará la señal de apagado a la Raspberry. Luego de apagado, no se permite encender el equipo hasta que la batería no alcance un estado mínimo de carga. La razón de este diseño es evitar un eventual descuido de parte del usuario, desconectando la fuente de energía antes de que el sistema pueda mantenerse encendido de forma independiente.

Por otro lado, se menciona que las señales RaspState_IN y RaspOff_OUT interactúan con el software RaspATX instalado en la Raspberry.

RaspATX

Dicho software tiene como entrada la señal RaspOff_OUT que da una orden a la Raspberry dependiendo del tiempo en el que se mantenga en alto:

- Menor o igual 0,5 segundos, es el comando de reset.
- Entre 2 y 8 segundos, es el comando de apagado normal.
- Mayor o igual a 8 segundos es un apagado forzado.

En el Bandonberry solo se le da uso a la segunda opción.

Por último, la otra señal involucrada con el software es RaspState_IN. Dicha señal como ya se mencionó en la descripción de señales, permite al microcontrolador conocer si es seguro o no des-energizar el sistema de forma definitiva.

Manejo de errores básicos

En la programación se hace un manejo de 3 errores básicos:

- **Error1:** La señal RaspState_IN demoró demasiado en activarse por lo que la Raspberry no está pudiendo inicializar correctamente.
- **Error2:** La señal RaspState_IN demoró demasiado en desactivarse, luego de haber enviado el comando de apagado. Por lo que la Raspberry no esta pudiendo entrar en el modo de "apagado seguro".
- **Error3:** El sistema no es capaz de apagarse. Una vez desactivada la señal de PowerOnOff_OUT y habiendo pasado 3 segundos el sistema despliega este error. Esto implica que o bien la señal no se está pudiendo enviar correctamente, o bien algo relacionado con el transistor Q1 está fallando. En caso de entrar en este error se cuenta con el *switch* que desacopla las baterías del sistema, por lo que siempre se lo puede des-energizar de forma manual.

Una vez generado cualquier error se despliega en el *display* "Fatal Err." y el número asociado a cada uno. Luego de unos segundos el sistema se apaga automáticamente siempre que sea posible.

6.7. Motherboard

La *motherboard* es la PCB que interconecta todos los componentes mencionados del BMS, la SBC y tiene pines de salida para el servomotor, sensor de presión, interfaz de usuario, botoneras y parlantes.

Para el diseño de la *motherboard* se utilizó KiCAD y el servicio provisto por la empresa JLCPCB, tal como se realizó con las PCBs de las botoneras. De esta manera, se obtuvo la PCB de la Figura 6.14.

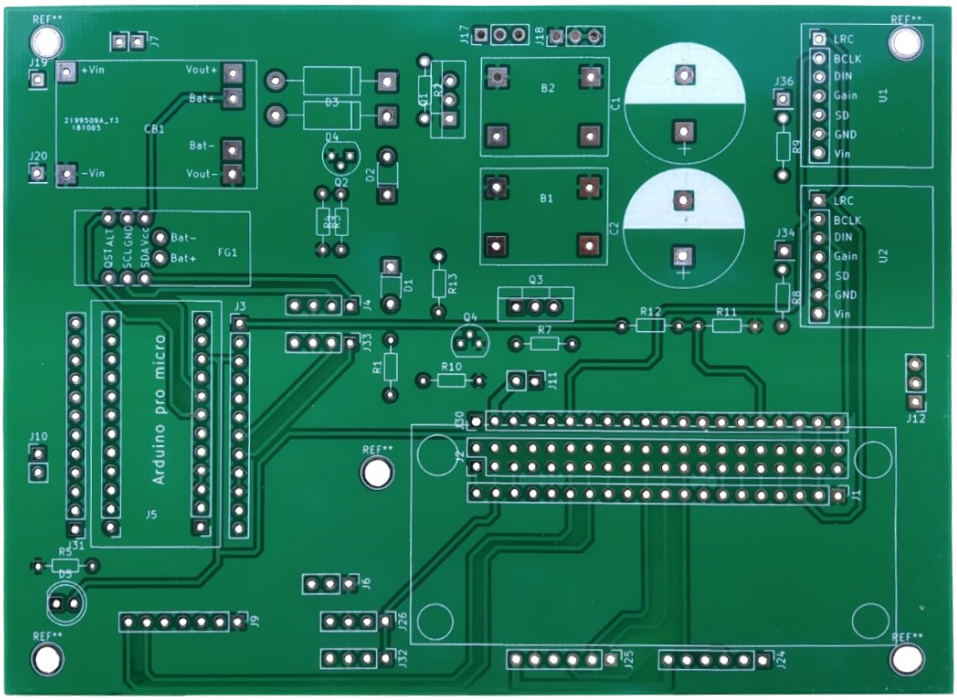


Figura 6.14: Imagen de la PCB de la *motherboard*.

6.7.1. Cálculo de ancho de pistas

Dado que en esta placa se deben manejar líneas de alimentación, es necesario prestar atención al ancho de las pistas. Las principales variables que entran en juego son: el ancho de la pista, la profundidad de la capa de cobre, la temperatura externa a la cual estará sometido el circuito y la corriente máxima que atravesará las pistas en cuestión. Se eligió el espesor de 1 onza, que equivale a 35 μm de cobre. Luego utilizando una aplicación de calculo de anchos de pista [8] se hicieron los siguientes supuestos:

- Corriente máxima de 2 A
- Temperatura máxima 35 ^{o}C.
- Se considera que es una capa externa.
- Largo máximo de la pista del orden de 10 cm.

Con dichos supuestos la calculadora genera las siguientes estimaciones asociadas en las condiciones anteriores:

- Ancho de pista necesario: 0,781 mm

[8] Disponible en `http://circuitcalculator.com/wordpress/2006/01/31/pcb-trace-width-calculator/`.

- Resistencia: 0,0646 Ω
- Caída de voltaje: 0,129 V
- Pérdida de potencia: 0,258 W

Finalmente tomando un margen se optó por hacer las pistas de 1 mm.

Capítulo 7

Estudio eléctrico del sistema

En el presente capítulo se analizan las prestaciones eléctricas del sistema, con el objetivo de definir la autonomía que brindan la batería y los tiempos de respuesta del sistema. Para ello, es necesario hacer un relevamiento de datos. A continuación se realiza dicho relevamiento y se describen los experimentos realizados que pueden ser útiles para la caracterización de futuras fabricaciones del Bandonberry.

7.1. Estudio de batería y consumos

7.1.1. Definición de condiciones de prueba

Para realizar los ensayos se definen dos condiciones de funcionamiento:

1. **En reposo:** Se tiene el sistema encendido y no se genera sonido ni se mueve el servomotor.

2. **En uso:** se genera un sonido constante (dos notas) por ambos canales de audio y al mismo tiempo se mantiene en movimiento el servomotor de forma periódica.

Se aclara que en ambos casos el Bandonberry se encuentra conectado a una red WiFi (que se utiliza para ser configurado) y sin ningún dispositivo conectado al puerto USB. Para simular el estado de uso, se detiene la ejecución de los programas *botoneras.py* y *fuelle.py* para dar lugar a un nuevo programa que hace lo descripto en el Cuadro 7.1.

```
# Reproduce dos notas por tiempo indefinido
# Repite:
    # Servo abre escotilla
    # Esperar 500 ms
    # Apagar PWM del servomotor
    # Esperar 500 ms
    # Servo cierra escotilla
    # Esperar 500 ms
    # Apagar PWM del servomotor
    # Esperar 500 ms
```

Cuadro 7.1: Programa de simulación de uso.

La demanda de corriente del servomotor y de la SBC no pueden ser alterados por software, sin embargo, se puede modificar la intensidad con la que se reproduce el sonido. El *byte* correspondiente al atributo *velocity* en el mensaje MIDI del evento *note on*, indica la intensidad con la que la nota se debe reproducir. De esta manera, se diferencian dos configuraciones en la simulación de uso:

1. Volumen máximo (*velocity* = 127).
2. Volumen medio (*velocity* = 64).

7.2. Ensayos con fuente externa

La fuente externa puede entregar una corriente máxima de 3 A. Dicha corriente se divide entre la necesaria para cargar la batería y la que consume el resto del sistema. En primer lugar se mide la corriente que consume el Bandonberry estando apagado y conectado a la fuente externa, para determinar la corriente de carga de la batería en el momento de realizar el ensayo. Se midió una corriente de carga de 160 mA. Dicho valor fue descontado en los siguientes cálculos de corriente para independizarse del estado de carga de la batería.

En la Figura 7.1 se muestra el sistema utilizado para obtener los datos. Se tiene una resistencia de medida conectada en serie con la línea negativa de la fuente externa, permitiendo que la fuente y el osciloscopio compartan la referencia a tierra. El osciloscopio mide la corriente y el voltaje entregados por la fuente. El valor de la resistencia utilizada es $R_{sense} = 0{,}25 \pm 0{,}04\ \Omega$.

En primer lugar se determinó que el consumo del Bandonberry en estado de reposo es prácticamente constante, teniendo una corriente promedio de aproximadamente 576 mA, sin considerar la corriente de carga de batería. En las Figuras 7.2 y 7.3 se tienen las medidas realizadas con el sistema simulando el uso y configuraciones de volumen medio y máximo, antes mencionadas.

Se observa que en ningún momento la fuente externa tuvo una disminución de voltaje apreciable. Por otro lado, se pueden observar los picos de corriente, correspondientes al movimiento del servomotor. En la Tabla 7.1 se encuentran los datos obtenidos de forma sistematizada.

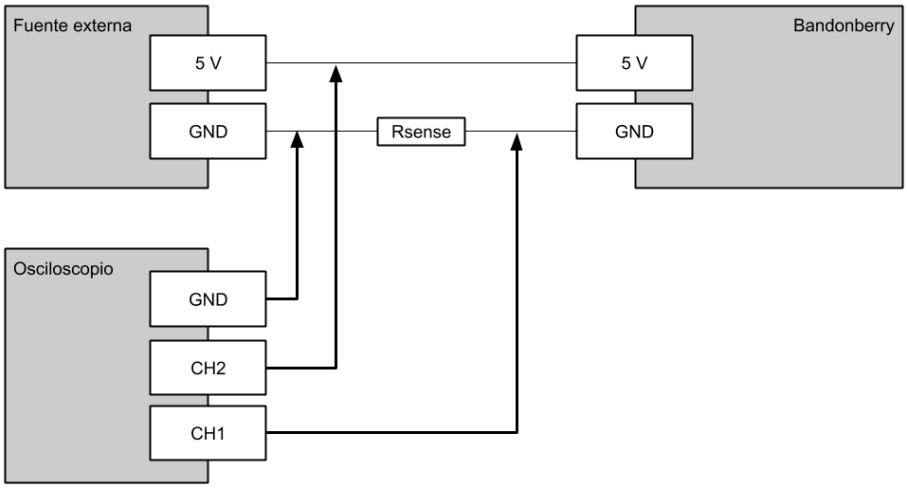


Figura 7.1: Esquema de conexión para realizar la medida con fuente externa.

Figura 7.2: Sistema con volumen máximo (*velocity* = 127, CH1 = V_{Rsense} y CH2 = V_{ext}).

Si se comparan estos resultados con lo estimado en la Sección 6.4.1, podemos concluir que la corriente promedio demandada por el sistema en condiciones de máximo volumen es superior a la esperada. Los picos medidos superan dicha estimación por 336 mA. Dicha diferencia se puede explicar, en parte por la estimación considerada en condiciones de voltaje de entrada de 5 V en la Tabla 6.2. Se asume que los *boosters*, el módulo selector y el regulador de voltaje de 3,3 V, tienen un consumo de energía despreciable. Más adelante, se explica que en particular los *boosters* consumen una corriente considerable a pesar de estar en condiciones

Figura 7.3: Sistema con volumen medio (*velocity* = 64, CH1 = V_{Rsense} y CH2 = V_{ext}).

Estado	**Corriente promedio (mA)**	**Corriente máxima (mA)**
En reposo	576	576
En uso con volumen medio	672	1.728
En uso con volumen máximo	1.056	2.336

Tabla 7.1: Valores de consumo para los distintos modos de funcionamiento.

favorables de trabajo.

De los datos obtenidos se puede concluir que:

1. El sistema funciona correctamente con una fuente de 5 V y 3 A.

2. El sistema consume en uso y con volumen máximo 1,20 A de corriente promedio, con picos de hasta 2,3 A.

3. Disminuyendo el volumen a la mitad, se logra disminuir considerablemente el consumo, en el orden de 36 %.

4. Dejando el sistema con el volumen medio se tiene una corriente promedio de 0,67 A, con picos de 1,73 A. Esto libera capacidad de potencia a la fuente externa permitiendo la demanda de 1 A necesario para la corriente máxima de carga de batería.

7.3. Ensayo de *boosters*

Al momento de probar el Bandonberry, alimentado solo con la batería se observó que el sistema fallaba bajo condiciones de máximo volumen, en el momento en el que se mueve del servomotor. Se supuso en primera instancia que la causa de error podían ser los *boosters* ya que teóricamente, lo único que cambió con respecto a las pruebas de la sección anterior, fue el voltaje de entrada.

Se procedió a estudiar el comportamiento de los *boosters* para entradas de voltaje variable y cargas resistivas constantes en su salida. Se mide la corriente y el voltaje de entrada y el voltaje de salida. La corriente entregada por el *booster* queda determinada por el voltaje de salida y el valor de la carga conectada, aplicando la Ley de Ohm. Los resultados de los experimentos se muestran en la Figura 7.4.

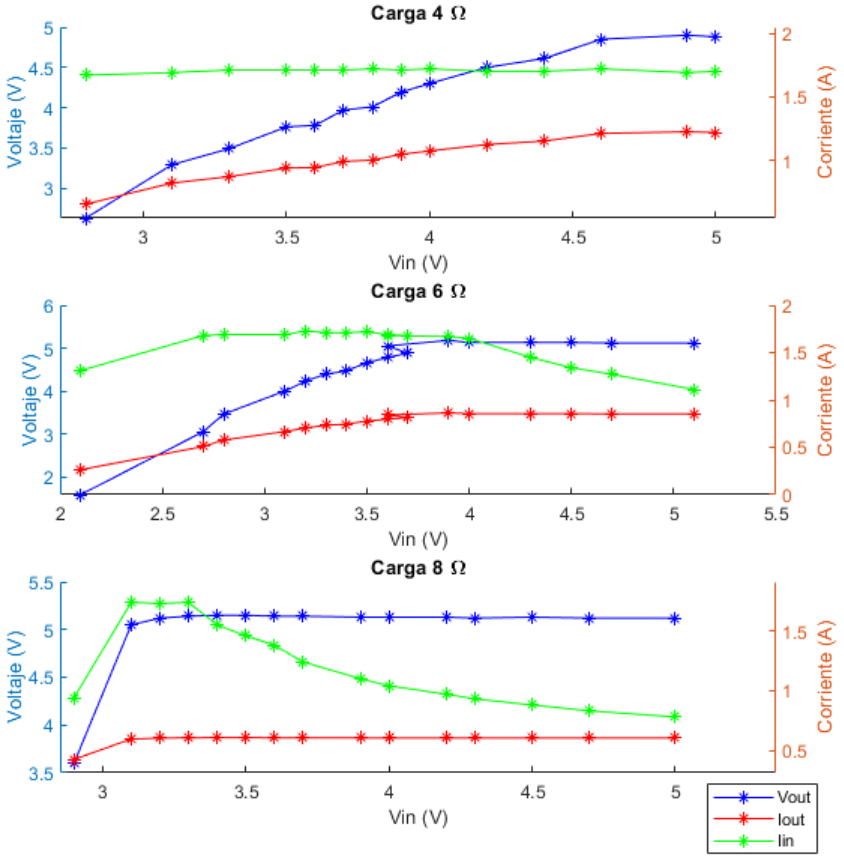


Figura 7.4: Ensayo eléctrico del *booster*.

De dicha figura se deduce que hay un compromiso entre el voltaje de entrada y el voltaje de salida, dependiendo de la carga. En el caso de la carga de 4 Ω, el *booster* no es capaz de sostener su voltaje de salida. La corriente de entrada se mantiene constante en 1,71 A en lugar de aumentar para compensar el menor voltaje de entrada. Del ensayo con carga 6 Ω se deduce que si se quiere tener una

salida de 5 V teniendo una entrada de 3,6 V, la carga tiene que ser menor. Para poder aprovechar mejor la energía de la batería a lo largo de toda su descarga (hasta alcanzar 3 V en los bornes de la misma), se debería tener una impedancia menor o igual a 8 Ω por cada *booster*.

En los datos relevados en el ensayo con carga de 4 Ω hay una diferencia de 480 mA entre I_{in} e I_{out}, teniendo 5 V de entrada, por lo que se tienen en el orden de 2,6 W de energía desperdiciada.

Considerando que el audio y el servomotor conectados en paralelo, tienen una impedancia mínima del orden de 2 a 5 Ω, hace que el sistema se vuelva inestable cuando el voltaje de entrada es menor a 5 V.

Si se comparan los datos de la Figura 7.4 y la Tabla 6.3, se puede apreciar que hay incoherencias entre lo establecido por el fabricante y lo medido. Por ejemplo con una entrada de 3,6 V el fabricante obtiene 4,98 V de salida y en el ensayo con carga de 4Ω se tiene 3,78 V.

Se concluye entonces que para mejorar las prestaciones del sistema se tiene que cambiar algo en los módulos de los *boosters*. Se plantean tres posibles soluciones:

1. Conectar 4 *boosters*, uno para el servomotor, uno por cada módulo de audio y un último para el resto del circuito.

2. Conectar 3 *boosters*, uno para el servomotor, otro para los dos módulos de audio pero acotando la potencia disminuyendo el volumen y un tercero para el resto del circuito.

3. Construir y diseñar un único *booster* a medida que cumpla con las condiciones necesarias.

7.4. Ensayos con batería

Se pretenden realizar mediciones eléctricas que permitan relevar la curva característica de la batería y su respuesta al consumo del sistema. En la condición de reposo, se tiene una carga que no varía mucho en el tiempo, ya que ni el servomotor, ni la generación de audio, están presentes.

En la condición de uso, se puede obtener información sobre la dinámica del sistema, ya que la impedancia va a tener variaciones grandes y bruscas a medida que se active el servomotor y se genere sonido. Para asegurar que el Bandonberry opere de forma correcta en la simulación de uso, se configuró el volumen en el nivel medio.

7.4.1. Situaciones a considerar en el ensayo

Punto de voltaje crítico:

A medida que la batería se vaya descargando, su voltaje comenzará a decaer. Con la caída del voltaje de alimentación, los *boosters* comenzarán a perder eficiencia y también caerá su corriente de salida. Llegará un punto en el que la Raspberry

Pi (siendo el elemento crítico y más sensible a los cambios de voltaje) se apagará. A dicho punto se lo denomina punto de voltaje crítico, o punto crítico.

En el experimento realizado, el punto crítico se manifiesta en el momento en el que la curva de descarga aún no define el codo de caída abrupta, que se espera teniendo como referencia la Figura 6.4. Por este motivo, para poder proteger a la Raspberry Pi de reinicios periódicos por falta de energía, se la desconecta y se coloca una resistencia equivalente ($R_{eq} = 43\ \Omega$). Dicha resistencia, se selecciona con un valor tal que el consumo de corriente del sistema se mantenga igual al promedio medido hasta el momento. De esta manera, se puede continuar midiendo la curva de descarga de la batería.

A fines prácticos del proyecto, lo que realmente interesa es conocer el punto de voltaje crítico donde el sistema se apaga. También, es de interés conocer cuanta energía queda inutilizada en la batería debido al compromiso que genera el *booster*.

Variación de impedancia en el tiempo: Otra consideración a tener en cuenta, es el variación de la carga que se mide. La carga de todo el sistema es muy sensible a los cambios de actividad. En reposo, el consumo es relativamente bajo y constante, pero en el momento que se acciona el servomotor y se emite sonido, es notoriamente más alto y cambiante a lo largo del tiempo.

7.4.2. Diseño del experimento

Dado que se deben realizar muchas mediciones a lo largo de tiempos prolongados, se automatiza el proceso de medida utilizando un Arduino UNO. Se realiza el conexionado de la Figura 7.5. Además, se programa un *script*, que esté midiendo los voltajes en los bornes de la resistencia R_{sense} y paralelamente consultando las medidas del *Fuel Gauge*, para luego poder comparar. El valor de la resistencia utilizada es, de $R_{sense} = 0{,}25 \pm 0{,}04\ \Omega$. Los datos relevados por el Arduino se trasmiten por su puerto serial a una PC, utilizando el sistema PLX-DAQ [1] en Microsoft Excel.

Con los datos relevados y conociendo el valor de R_{sense}, se puede calcular la corriente consumida por todo el sistema. Para obtener los datos de las medidas del *Fuel Gauge* se programa al Arduino UNO, como un esclavo más del sistema I2C. De esta manera, el Arduino Pro Micro (el del BMS), consulta al *Fuel Gauge* y seguidamente envía dichos datos al Arduino UNO [2].

Cabe mencionar, que el sistema PLX-DAQ no acepta tiempos de muestreo menores a una muestra por segundo. Esto pasa a ser una limitante a la hora de medir los cambios rápidos en los consumos de corriente. En particular los picos de demanda que se dan por los movimientos del servomotor.

En la condición de reposo, los datos fueron relevados a razón de una muestra por minuto, ya que la variación de impedancia del sistema es muy pequeña. En la

[1] Parallax Data Acquisition tool - Macro para Excel, utilizado en la adquisición de datos de una placa Arduino. Para más información referirse a `https://www.parallax.com/downloads/plx-daq`.

[2] Para estabalecer la programación maestro - esclavo entre Arduinos se consultó el tutorial `https://www.arduino.cc/en/Tutorial/MasterWriter`.

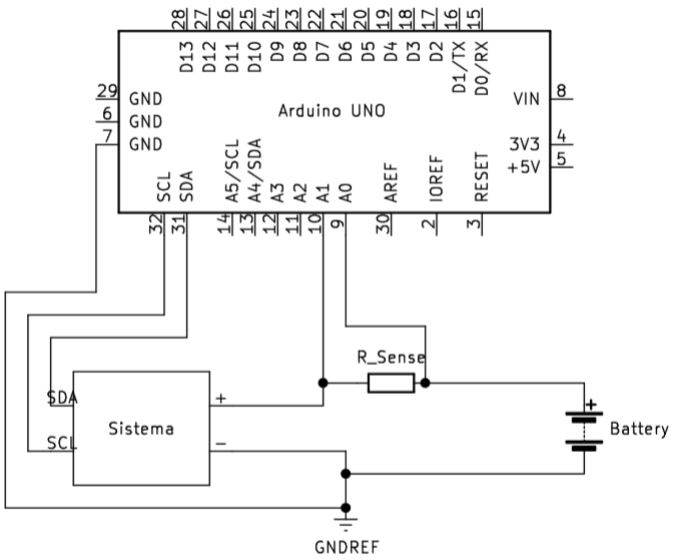


Figura 7.5: Esquema de conexión para ensayo de medida

condición de uso, el tiempo de muestreo se configuró a una muestra por segundo, para poder registrar los cambios en la mejor forma posible.

7.4.3. Datos obtenidos

En la Figura 7.6 se puede apreciar el voltaje de la batería a lo largo del tiempo, teniendo al Bandonberry en reposo. El primer tramo hasta el punto crítico corresponde a la medida realizada con la Raspberry Pi activa. El punto crítico es donde esta se apaga por primera vez. Del punto crítico en adelante, la medida es realizada con la R_{eq} en lugar de la Raspberry Pi.

De la Figura 7.6 se puede decir que, partiendo de la batería cargada, se tiene como máximo tiempo de encendido 1100 minutos (18 horas aproximadamente). Por otro lado se mide el punto crítico, resultando de 3,43 V. Se observa que, en estas condiciones, la descarga de la batería tiene un comportamiento casi lineal hasta los 3,2 V.

Desde el inicio del experimento hasta el punto crítico, se extrajo una carga C = 8.637 mAh, lo que representa una carga de 2.159 mAh por celda. Si se consideran los 2500 mAh nominales por celda como efectivos, se estaría utilizando el 86 % de la capacidad nominal.

Luego se a ejecutó el programa de simulación de uso. En dichas condiciones se obtuvieron los datos representados en la Figura 7.7. Se superponen los datos obtenidos con el sistema en reposo a modo de referencia.

Se puede observar una región del espacio delimitada por las rectas de “aproximación solo audio” y “aproximación audio-motor”. La recta “aproximación audio-motor” representan los puntos en los cuales el sistema tiene su menor consumo, definida en el momento en el que el servomotor y los módulos de audio están en

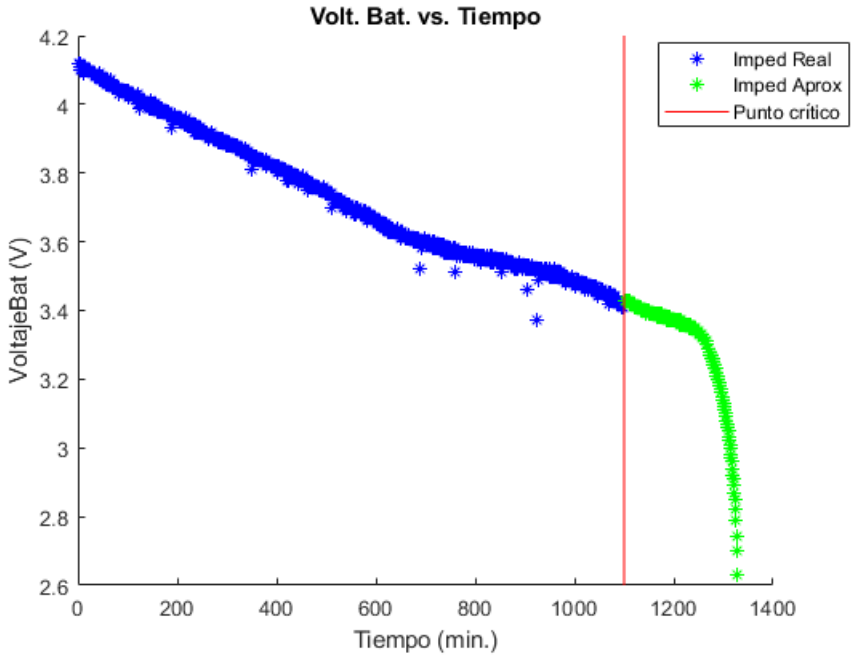


Figura 7.6: Voltaje de la batería en función del tiempo con el sistema en reposo.

su máxima actividad. La recta "aproximación solo audio" representa los puntos donde el servomotor no esta consumiendo pero si lo hacen los módulos de audio.

Como primera observación, la función que asocia el voltaje con el tiempo, es sumamente compleja de definir, ya que depende del consumo instantáneo del sistema. Por otro lado se aprecia que la impedancia del sistema tiene una dinámica muy particular, y que por algún motivo la batería tiene una oscilación en el voltaje en sus bornes de un valor aproximadamente 0.1 V. Dicha oscilación en el rango útil de 3.4 V a 4.1 V corresponde a 14,29 % del intervalo, por lo que es una variación a considerar. Depende fuertemente de la comparación entre la resistencia interna de la batería y la impedancia mínima del sistema. A menor resistencia interna de la batería, la oscilación disminuirá. Se entiende que solucionando el problema del *booster*, la oscilación del voltaje probablemente disminuya, ya que la demanda del sistema va a ser notoriamente menor.

Por último, se puede afirmar que se tiene una autonomía de más de tres horas (sin tener el volumen del sonido a la máxima capacidad).

7.5. Mediciones del Fuel Gauge

Se estudia el módulo de *Fuel Gauge*, con el objetivo de verificar si su funcionamiento es correcto y si se adapta al sistema. En la Figura 7.8 se tiene la medida realizada de la batería en condición de reposo, por parte del Arduino y del Fuel Gauge.

Se observa una diferencia constante entre ambas medidas. Los datos del Fuel

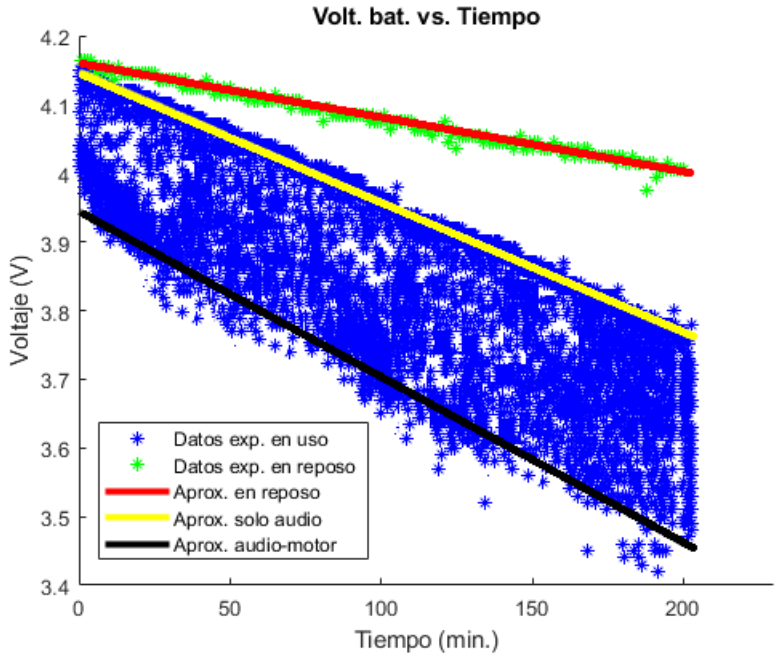


Figura 7.7: Gráfica obtenida por el sistema de medición utilizando la R_{sense} en estado de uso, superpuesto con los datos obtenidos en estado de reposo.

Gauge tienen la corrección de la caída de voltaje inducida por R_{sense}, por lo que el único factor que puede estar generando el *offset* puede ser un diferencia entre las referencias de cada medición. Concretamente, se tiene una diferencia promedio, entre ambas medidas, de $0,09$ V. A pesar de la existencia del *offset*, se puede decir que el Fuel Guage tiene una medida coherente con respecto a la del Arduino. En términos prácticos, al ser una diferencia constante, no influye ya que los umbrales críticos se pueden definir relativos a esa medida.

7.6. Latencias

En esta sección se estudia el desempeño del sistema con respecto a sus tiempos de respuesta. Para ello se programaron instrucciones que cambian el estado de ciertos GPIOs para poder visualizar en un osciloscopio los tiempos de ejecución de los programas. Los siguientes resultados corresponden al sistema corriendo todos los programas (Fluidsynth, *botoneras.py* y *fuelle.py*).

7.6.1. Botoneras y fuelle

En ambos programas se agregaron instrucciones que cambian el estado de un GPIO cada vez que el *loop* principal vuelve a comenzar. Para el programa botoneras.py, se utiliza el GPIO 17 y se conecta al CH2 del osciloscopio. Para el programa fuelle.py, se utiliza el GPIO 27 y se conecta al CH1 del osciloscopio. Los

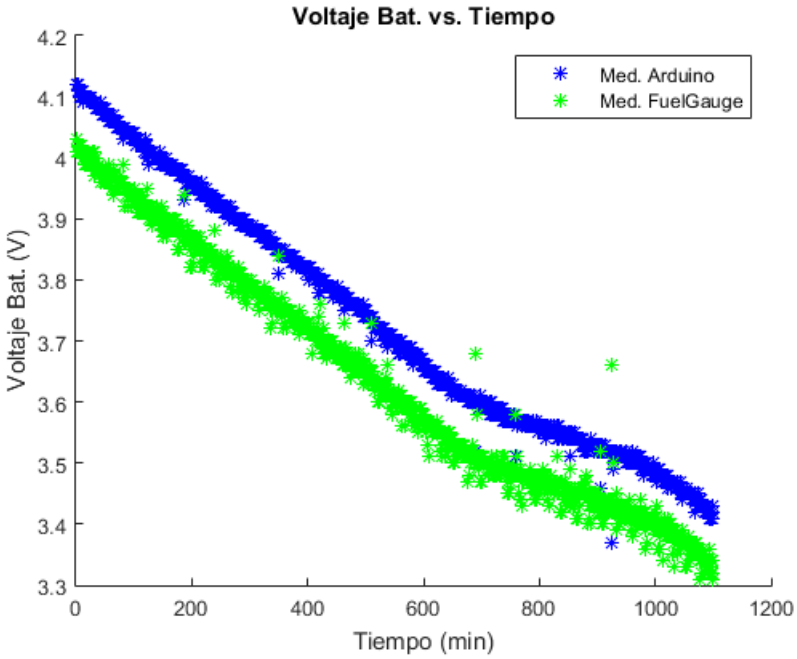


Figura 7.8: Comparación entre las medidas del Arduino y el Fuel Gauge.

resultados se observan en la Figura 7.9.

Se observa que los programas no están sincronizados y que ambos se ejecutan en simultáneo de forma independiente. Para el caso de *botoneras.py* se tiene un tiempo de ejecución de 35 ms y para *fuelle.py* 24 ms.

Botoneras

La duración estimada para el *loop* principal en la Sección 4.6 es de 12 ms, menor a la mitad de lo observado. Una mirada en mayor detalle, utilizando la misma técnica de usar el GPIO como bandera de estado, permite identificar los factores que inciden en la ejecución del programa.

En primer lugar, el tiempo que se tarda en enviar los mensajes por SPI es mayor al estimado. Se observan tiempos de 180 a 800 μs, mucho mayores a los 5 y 15 μs medidos en la Sección 4.7.1. Esto se debe a que la nueva medida es ejecutada con el programa *fuelle.py* corriendo y el programa *botoneras.py* debe alternar la comunicación entre los puertos SPI de cada botonera.

En segundo lugar, la función utilizada para introducir el *delay* entre lecturas sucesivas, en algunas ocasiones, genera un *delay* mayor al especificado. En la documentación de Python [27] se especifica que el verdadero tiempo de *delay* puede diferir. En el caso de tardar más, se debe a que el sistema operativo se encuentra ejecutando otras tareas.

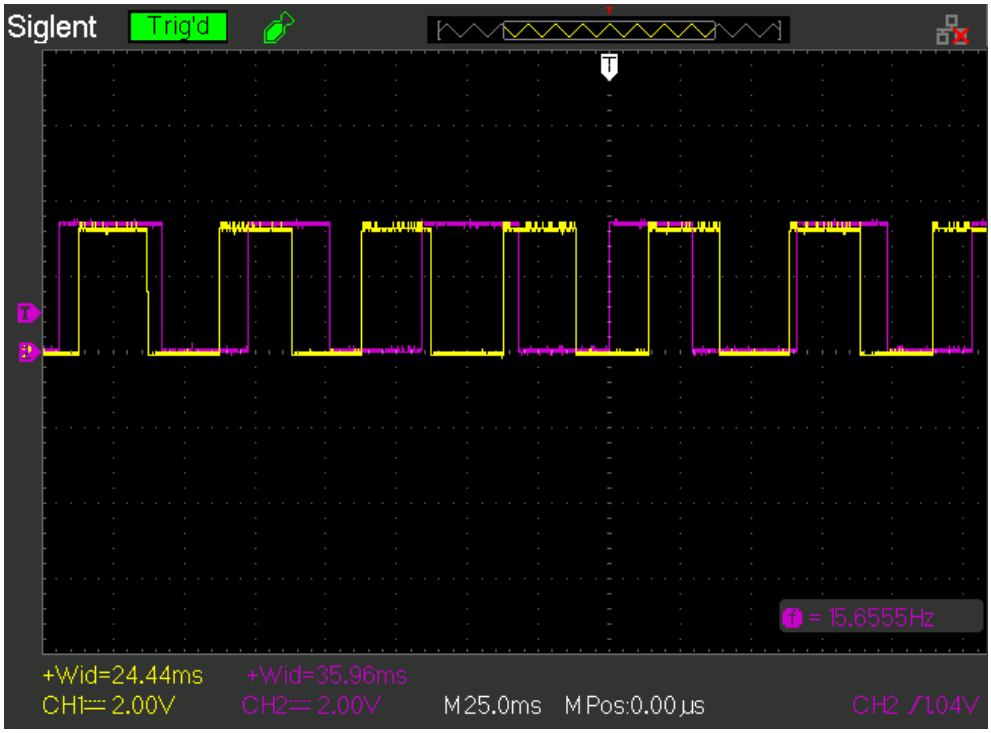


Figura 7.9: Tiempos de ejecución de los programas *botoneras.py* (CH2) y *fuelle.py* (CH1).

Fuelle

La duración estimada para el *loop* principal en la Sección 5.3.1 es de 12,88 ms, aproximadamente la mitad de lo observado. Para medir el proceso de lectura de datos del sensor, se conecta el CH2 del osciloscopio a la línea de datos (SDA) en el puerto I2C.

En la Figura 7.10 se puede observar que la transferencia de datos en el bus I2C no se hace de forma continua, por lo que el tiempo de transferencia es mayor al estimado. Los mensajes por I2C se ven como líneas verticales, siendo estas en realidad ondas cuadradas con un período suficientemente alto en relación a la escala de tiempo.

Por otro lado, los *delays* utilizados para dar lugar a las conversiones de temperatura y presión, sufren del mismo problema que en las botoneras (ver Cuadro 7.2). En general, los *delays* generan mayores retardos a los especificados. Luego de obtener el dato de presión, el programa demora unos milisegundos adicionales para enviar los mensajes MIDI, los cuales a diferencia de las botoneras, deben ser enviados en cada iteración.

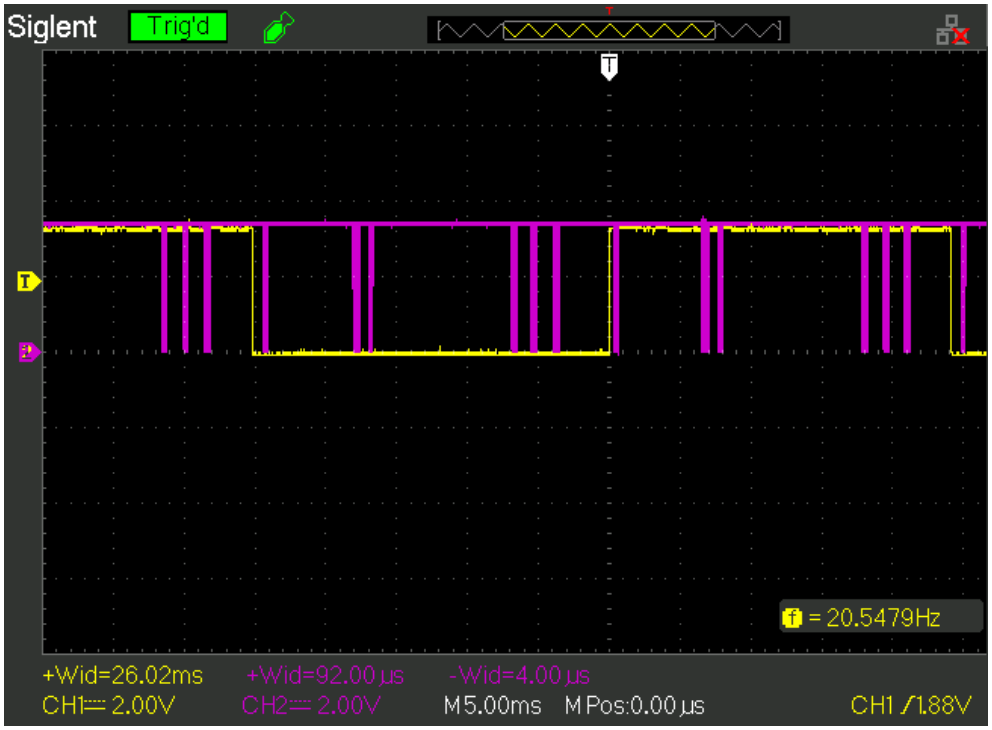


Figura 7.10: Ejecución del programa *fuelle.py* (CH1) y actividad en el bus I2C (CH2).

.

```
# Iniciar medida de temperatura
# Tiempo de espera de 5 ms
# Lectura de datos de temperatura
# Iniciar de medida de presion
# Tiempo de espera de 8ms
# Lectura de datos presion
# Calculo de presion y temperatura en unidades fisicas
```

Cuadro 7.2: Pseudo código general del programa fuelle.

Tomando el período de muestreo como $T_s = 25$ ms, según lo que se puede ver en la Figura 7.9, determinado por la ejecución del programa *fuelle.py*, nos queda que la frecuencia de muestreo real es $f_s = 40$ Hz. Realizando el mismo cálculo hecho en la Sección 5.3.2, la frecuencia del polo de la transferencia asociada al filtro pasabajos es:

$$f_{-3dB} = \frac{0,999.f_s}{2\pi} \approx 6,36\ Hz \tag{7.1}$$

7.6.2. Fluidsynth

Para medir el tiempo de respuesta del sintetizador, se agregaron instrucciones en el programa *botoneras.py* para que el estado del GPIO 17 (CH2) cambie en el

momento que un mensaje MIDI se envía. Se modificó el programa *fuelle.py* para que simule un fuelle contrayéndose, pero sin dejar de tomar muestras de presión. El resultado se muestra en la Figura 7.11. A partir de los 50 ms se puede observar como la amplitud de la señal de audio (CH1) comienza a crecer. Se considera este tiempo como el valor del *delay* del sintetizador.

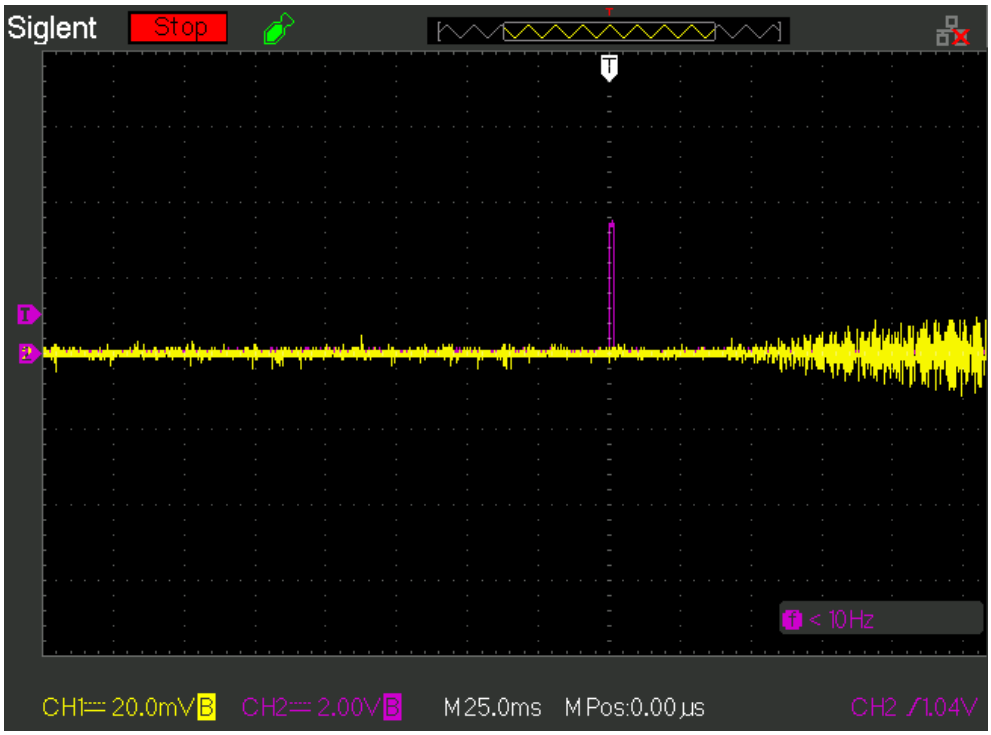


Figura 7.11: Envío de mensaje MIDI (CH2) y salida de audio (CH1).

7.6.3. Total

Si suponemos el peor caso y sumamos todos los *delays* de todos los programas, se llega a un valor total de:

$$35\ ms + 24\ ms + 50\ ms = 109\ ms \tag{7.2}$$

Este valor se encuentra dentro del objetivo planteado y brinda un margen para posibles variaciones en la ejecución.

Capítulo 8

Carcasa y modelado 3D

En el presente capítulo se explica los criterios considerados para la elección del material que compone toda la carcasa y estructura del Bandonberry. Además, se muestran las consideraciones en el diseño y algunas observaciones relativas a la fabricación y construcción. Para el lector que no este familiarizado con la impresión 3D se recomienda que lea el Apéndice D, antes de comenzar este capítulo.

8.1. Elección del material de fabricación

Considerando que el objetivo es construir un instrumento de práctica, se procura seguir con la mayor fidelidad la forma y dimensiones de un bandoneón. Se busca la forma más económica y sencilla de realizar toda la carcasa. Se evalúaron los siguientes materiales y procesos:

- Madera y/o MDF [1] con sistema de corte láser.
- Acrílico con sistema de corte láser.
- Impresión 3D con PLA [2] (Poliácido Láctico) o ABS [3].

Entre las opciones presentadas no hay un material o proceso que realmente destaque de manera determinante frente a los otros. Para el proyecto se eligió la impresión 3D por los siguientes motivos:

- La Facultad de Ingeniería posee dos impresoras 3D en el Instituto De Ingeniería Eléctrica.
- Se cuenta con una impresora 3D para el uso exclusivo del proyecto.

[1] Del inglés *Medium Density Fibreboard*, compensado de madera.

[2] Es un plástico similar al PET, que se produce a partir de almidón de maíz, mandioca o caña de azúcar. Es biodegradable, y se degrada en contacto con agua u óxido de carbono.

[3] Del inglés *Acrylonitrile Butadiene Styrene*, tipo de plástico utilizado para impresión 3D.

- La cantidad de opciones de distintos materiales y colores resulta muy atractiva.

Como material se eligió el PLA frente el ABS por ser menos contaminante, ya que el ABS deriva del petróleo [1], el PLA es biodegradable [19] y más fácil de conseguir en el mercado local. Dado que el modelo 3D es libre, puede ser adaptado para ser construido con otros materiales.

8.2. Modelado 3D

8.2.1. Software utilizado

Para el modelado 3D se utiliza el programa Fusion 360 de la compañía Autodesk[4]. Dicho software cuenta con licencias gratuitas para estudiantes y profesores de instituciones educativas. Fusion 360 es capaz de generar archivos *.stl* utilizados para representar objetos 3D. Son ininteligibles para las impresoras 3D en este formato. Esto lleva a la necesidad de un segundo programa que para cambiar el formato .stl a Gcode[5].

Este último es el que puede interpretar la impresora 3D. Para poder generar los archivos Gcode se utiliza el programa Repetier [6]. Dicho programa es gratuito y muy utilizado para este tipo de aplicaciones.

8.2.2. Diseño de la carcasa

Para el diseño de la carcasa, se midió el bandoneón y se plasmó en modelos 3D [7]. Se muestran en la Figura 8.1.

Las piezas se encuentran divididas en varias partes, ya que las impresoras estándar, y en particular la utilizada para este proyecto, tienen una capacidad volumétrica de $21 \times 21 \times 30$ cm. Las diferentes piezas se unen a través de tuercas y tornillos, esto permite acceso al interior de las caja sin necesidad de destruirlas. En la Figura 8.2 se pueden observar los diferentes encastres diseñados para ensamblar las carcasas. Los encastres A y C son "autoportantes" de tuerca. La idea de dichos encastres es poner una tuerca en el orificio y luego colocar pegamento, tipo masilla epoxi, para fijarla a la pieza. De esta manera, se logra una "tuerca embutida", facilitando la tarea de armado.

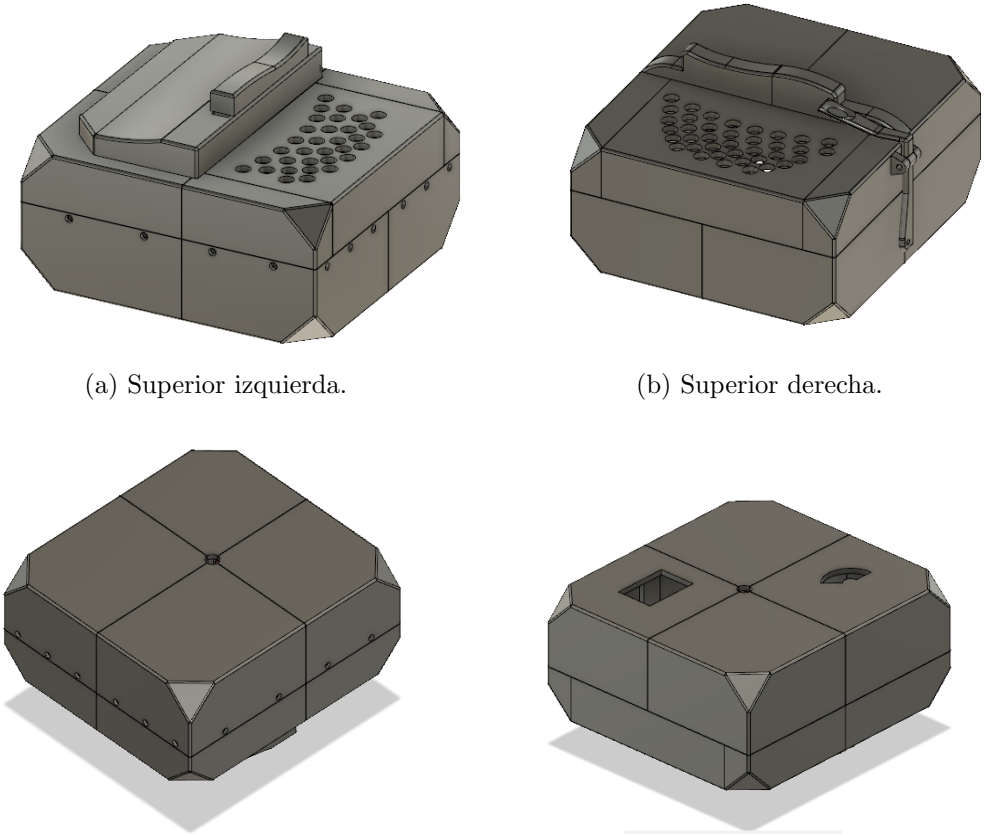
El hecho de ocultar los tornillos, aumenta la complejidad en el proceso de impresión y de diseño, ya que se debe considerar el ensamblado de las piezas. Es decir,

[4]Para descargar el software el requisito es tener una casilla de correo con dominio *.edu*, ver `https://www.autodesk.com/products/fusion-360/students-teachers-educators`.

[5]Código utilizado por la mayoría de las impresoras 3D para el control de motores y temperatura.

[6]Disponible en `https://www.repetier.com/`

[7]Todas las medidas realizadas se encuentra `https://github.com/jebentancour/Bandonberry`



(a) Superior izquierda.

(b) Superior derecha.

(c) Inferior izquierda.

(d) Inferior derecha.

Figura 8.1: Vistas de los modelos 3D de las carcasas.

se debe tener en cuenta los espacios necesarios para introducir las manos, destornilladores, tornillos, entre otros. En la Figura 8.3 se puede observar la ubicación de los encastres, y cómo aumenta la complejidad de la pieza.

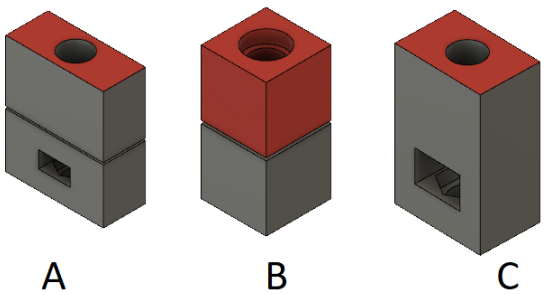


Figura 8.2: Diseños de distintos tipos de encastres.

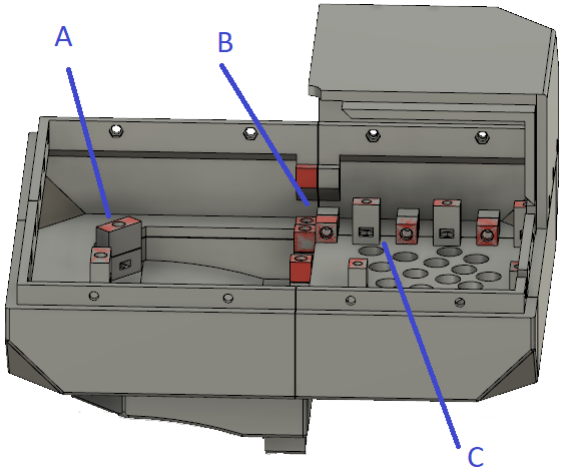


Figura 8.3: Integración de los encastres en la tapa izquierda.

8.2.3. Diseño de los botones

Los botones del bandoneón tienen una excursión que acompaña el movimiento de una válvula. Por lo tanto la apertura de dicha válvula, altera el pasaje de aire, acompañando la presión en todo el recorrido. En el caso del Bandonberry se consideró un modelo binario: presionado o no. El modelo del mecánico del botón, contempla una excursión en la que al final del recorrido acciona el pulsador eléctrico. De esta manera, se tiene una mejor simulación de la mecánica sin aumentar la complejidad de la electrónica. Como contraparte tiene un aumento en el valor de las latencias, por lo que se deberían hacer los estudios correspondientes para poder evaluar el impacto final en el sistema. El diseño de los botones se muestra la Figura 8.4.

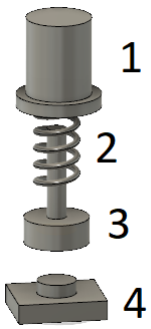


Figura 8.4: Sistema mecánico del botón.

De arriaba hacia abajo se tiene: botón (1), resorte (2), contrabotón (3) y pulsador (4). El contrabotón está adherida a la parte móvil del pulsador eléctrico y genera una guía para el movimiento del resorte y el botón. En el bandoneón todos

los botones tienen cierta diferencia de altura, tal como se muestra en la Figura 8.5. Los más alejados del centro de la tapa, tienen mayor altura que los más cercanos

Figura 8.5: Diferencia de altura entre los botones.

Para simular esta característica, se crearon botones con tres alturas diferentes distribuyéndose en el Bandonberry de forma similar al bandoneón. Dichos botones se pueden apreciar en la Figura 8.6.

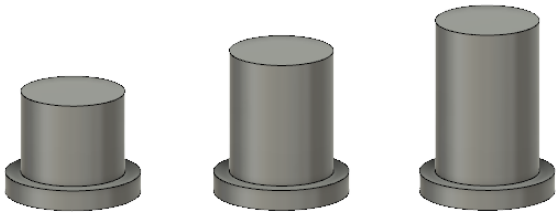


Figura 8.6: Variación de altura de los botones.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 9

Conclusiones, resultados y trabajo a futuro

En este capítulo se exponen las conclusiones del proyecto, se explican cuáles fueron los requerimientos alcanzados, cuáles no y por qué. Además, se proponen trabajos a futuro para mejorar el Bandonberry, tanto en sus prestaciones funcionales como en su potencial didáctico.

9.1. Conclusiones generales

Como primera conclusión, se logró diseñar e implementar un simulador de bandoneón satisfactoriamente. Es capaz de generar el sonido característico del instrumento, imitar su mecánica y forma, y comunicarse con una PC por medio de mensajes MIDI. Esto resulta una opción interesante para los estudiantes de bandoneón.

En segundo lugar, se considera algo positivo, que el sistema esté diseñado en forma modular y con componentes accesibles en el mercado. Esto permite una adaptación rápida y sencilla frente a cambios de disponibilidad de componentes, siendo posible sustituirlos sin tener que modificar el diseño original. Además de esta forma, es posible mediante la modificación de módulos específicos, que el usuario realice mejoras según su conveniencia.

En lo que respecta a lo académico, para llevar adelante el proyecto, se integraron conocimientos de distintas asignaturas de la carrera de Ingeniería Eléctrica. Se utilizaron conocimientos de electrónica analógica y digital, análisis de comportamiento de baterías, estudio de latencias, programación en Python y C++, diseño de PCBs, entre otros. Por otro lado, se incursionó en el campo de la Ingeniería Mecánica, en el diseño e implementación de la carcasa, el fuelle y los botones.

Por último, el desafío de realizar un trabajo en equipo, fue también una fuente de aprendizaje por sí misma. Cada integrante aportó sus propios conocimientos y habilidades de los cuales el resto se nutrió. Durante el transcurso del proyecto se generaron problemas a los que el equipo supo responder y adaptarse de forma exitosa.

9.2. Evaluación y presentación de resultados

Audio

El sistema cuenta con dos parlantes, uno en cada botonera, los cuales funcionan en modo estéreo, simulando la disposición espacial de las botoneras, tal como se había previsto. La calidad y volumen del sonido del Bandonberry se consideran aceptables para espacios reducidos. De requerir mayor amplificación, se puede utilizar la interfaz MIDI para sintetizar el sonido y amplificarlo. Cabe destacar, que el hecho de utilizar un sintetizador permite alterar las características del sonido (afinación, timbre, amplitud, entre otros).

Latencias

Se logró una latencia del orden de los 100 ms, estando por debajo de los 200 ms planteados. Dicho resultado es de gran relevancia, ya que no genera grandes inconvenientes al usuario.

Diseño mecánico y carcasa

Fue posible reproducir la forma del bandoneón en su totalidad respetando las medidas del instrumento utilizado como modelo. Se contemplaron los 71 botones y la palanca de liberación de aire. Con respecto a la fabricación, se espera que a medida que la tecnología avance, la impresión 3D sea más sencilla y de menor costo. A su vez, se espera que las impresoras de mayor tamaño sean más accesibles, evitando así seccionar en tantas partes el diseño de la carcasa, ahorrando trabajo y posibles problemas en el ensamble.

Fuelle

Se logró diseñar y construir un fuelle que tiene la misma cantidad de cuerpos y pliegues que el fuelle del bandoneón. A su vez se propone una forma sencilla y económica de fabricarlo basado en la técnica de origamis. Resultó ser lo suficientemente hermético para lograr medir las variaciones de presión interna para modular el volumen en las notas.

Batería y autonomía

El Bandonberry posee una autonomía del orden de 3 horas en condiciones de uso, siendo capaz de funcionar a lo largo de una lección de música estándar. Con respecto a los tiempos de carga, se considera que no se logró un buen resultado, pero aún así se plantearon opciones para mejorarlo. Se destaca también que el Bandonberry no requiere de dispositivos adicionales para poder funcionar.

Interfaz USB

Se logró que el Bandonberry cuente con una interfaz USB, la cual es reconocida por una computadora como dispositivo MIDI. Esta función le brinda muchas

posibilidades, en lo que refiere a herramientas de software. Se pueden implementar sintetizadores más sofisticados, programas con fines didácticos, sistemas de amplificación, entre otros.

Costos de materiales y fabricación

El costo en materiales resultó del orden de $ 6.800 (pesos uruguayos) . Dicho número cumple satisfactoriamente las expectativas de los requerimientos, que especificaban un valor máximo de $ 12.000. De todas maneras, se aclara que el resultado no considera costos de mano de obra, importación e impuestos. Para más información ver el Apéndice B.

Liberación de la documentación

Toda la documentación y conocimiento generado en este proyecto, se encuentra disponible de forma pública. Una gran parte se encuentra en el presente documento, y el resto en un repositorio de Github [1]. Queda abierta la posibilidad a que otras personas puedan desarrollar mejoras y abrir un universo de posibilidades.

9.3. Desafíos y aprendizajes

A lo largo de la carrera se adquirieron conocimientos técnicos y una sólida capacidad analítica físico-matemática. Con estas capacidades, fue posible resolver gran parte de los problemas y generar nuevos conocimientos. A continuación se exponen brevemente algunos desafíos resueltos:

Raspberry - Python

Se investigaron los sistemas operativos disponibles para Raspberry Pi, programación en Python, protocolo de comunicación como el SSH y MIDI. En particular se tuvieron que analizar con detenimiento las librerías asociadas al control del servomotor, sensor de presión y *GPIO expanders*. Esto permitió entender como se refleja en la programación las especificaciones señaladas por el fabricante en las hojas de datos.

Arduino - C++

Se incursionó en el lenguaje de programación C++ , se tuvieron que investigar las librerías del Fuel Gauge y del *display*. Por otro lado, se experimentó el manejo de errores, buscando la manera de definirlos y mitigarlos.

Botoneras - Algoritmo de barrido de matriz

La botonera representó un gran desafío, ya que implica la lectura de 71 botones con un número mucho menor de entradas. Se debió estudiar la hoja de datos de los

[1] Accesible en `https://github.com/jebentancour/Bandonberry`.

GPIO *expanders* y manejarlos desde la Raspberry para su correcto funcionamiento. Por otro lado, el diseño del circuito fue muy enriquecedor, resultando ser una forma muy sencilla e ingeniosa de resolver un problema frecuente en el diseño de sistemas digitales.

Diseño e implementación de PCB

Se debieron adquirir conocimientos de software de diseño de circuitos. Fue necesario aprender a utilizar la prototipadora disponible en la Facultad de Ingeniería, con la cual se hicieron las primeras placas de prueba antes de enviar el diseño final para su fabricación en el exterior.

Estudio de latencias

La medición de las latencias fue un desafío ya que algunos eventos son difíciles de medir sin alterar su valor al hacerlo. La utilización de banderas de fin de programa fue una estrategia útil, ya que los tiempos de cómputo de los programas eran considerablemente mayores al tiempo de cambio de estado de los GPIOs. Otra complejidad que forma parte del problema, es que la asignación de recursos es controlada por el sistema operativo, impidiendo que los tiempo de ejecución sean constantes a la hora de medir.

Módulo BMS - Estimación y consumos

Para el diseño del BMS, se necesitó investigar los conceptos sobre baterías, convertidores de voltaje, cargadores y afines. Se debieron diseñar ensayos para corroborar las estimaciones y re-evaluar algunas decisiones, por ejemplo, la elección de los transistores de potencia y los *boosters*. En algunas situaciones la teoría puede aplicarse como una guía estimativa, pero a veces los sistemas adquieren una complejidad tal que resulta más eficiente construirlos y testearlos. De dichas pruebas es posible sacar mayor cantidad de conclusiones y con mayor precisión, en comparación a generar modelos predictivos que son más complejos y refinados.

Planificación de tiempos de importación

El alcance inicial, previo a la elaboración de este documento, fue más ambicioso y proponía varios elementos que potenciarían la capacidad didáctica del instrumento. En particular, no se pudo incorporar *displays*, cuya función era indicar la ubicación de las notas en el teclado. Atrasos en el proyecto, junto a una inadecuada estimación de los tiempos requeridos para la importación, contribuyeron a este inconveniente.

Concepto de reutilización de hardware

Se buscó reutilizar módulos ya desarrollados con funcionalidades específicas. De esta manera, se pudieron enfocar los recursos del proyecto en el producto final y no en componentes específicos. También se comprendió el valor de la documentación,

los componentes o proyectos bien documentados son aquellos que pueden usarse de forma más eficiente, siendo más accesibles a la hora de entenderlos y reutilizarlos.

Documentación

Fue muy enriquecedora la experiencia de documentar un proyecto, donde se debieron expresar los requerimientos y los ensayos de forma específica, la forma de exponer los datos, el orden de la información, entre otros. Por otro lado, se considera importante para el ejercicio profesional, tener la experiencia de una redacción científica de esta magnitud.

Impresión 3D

Respecto al diseño, se considera útil haber incorporado nociones del manejo de software de diseño 3D, ya que muchos proyectos de Ingeniería requieren de dicha tecnología para la fabricación de carcasas, piezas especiales y similares.

Trabajo en equipo

Se aprendió a sobrellevar las diferencias, a defender los argumentos de forma adecuada, a tomar decisiones en conjunto, a entender en qué temas y en qué momentos vale la pena entablar discusiones o no. También, comprometerse con el trabajo y responsabilizarse por las tareas que le corresponden a cada uno.

9.4. Trabajo a futuro

El proyecto da lugar a cambios de implementación, de ciertos módulos que pueden mejorar el rendimiento y/o experiencia del usuario. A continuación proponemos algunos cambios que no se realizaron por falta de tiempo o por estar fuera del alcance del proyecto:

- Bajar el tiempo de carga de la batería, cambiando el módulo del cargador.
- Mejorar los *boosters*, ya sea agregando más unidades en paralelo o diseñando uno a medida para el sistema.
- Implementar un software con fines didácticos que pueda interactuar a través de la comunicación MIDI.
- Reducir el sonido producido por el movimiento del servomotor, aisalándolo acústicamente o buscando otro más silencioso.
- Reducir el sonido de conmutación de los pulsadores en las botoneras, cambiando los mismos por botones de silicona o membrana.
- Investigar la fabricación de la carcasa con madera o acrílico y cortadora láser, evaluándolo en términos calidad y costo.

- Explotar la característica de conectividad inalámbrica presente en la Raspberry Pi Zero W. Ya sea a través de WiFi con aplicaciones web o conectividad Bluetooth para transmitir información MIDI o de audio.
- Modificar la etapa de salida de audio, incorporando un puerto para auriculares.
- Agregar un control maestro de volumen de sonido.
- Incorporar una librería de diferente *Sound Fonts* con una interfaz acorde para seleccionarlas.

Apéndice A

Selección de transistor de potencia

En el siguiente apartado se describe el proceso de selección y dimensionamiento del transistor de alimentación utilizado en el Bandonberry.

El transistor de alimentación corresponde a Q1 en el Selector del BMS. En la Figura A.1 se vuelve a presentar dicho circuito. El componente seleccionado en primera instancia fue el transistor IRF9630 [28], el cual, con su corriente nominal de 6.5 A, parecía adecuado para la tarea.

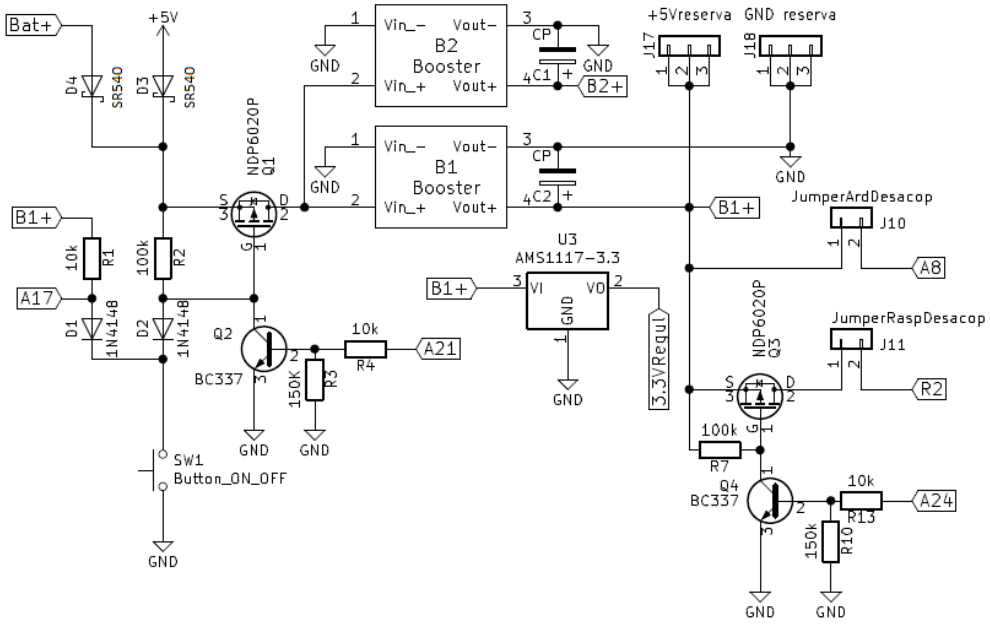


Figura A.1: Esquemático del módulo Selector.

Se comenzaron a tener problemas en la primera implementación del sistema. El mismo no lograba mantenerse estable, se reiniciaba y muchas veces no lograba terminar el proceso de inicialización. En particular, la falla se generaba con demandas de picos de corriente (movimientos del servo, inicialización del sistema operativo de la SBC o inicialización de la salida de audio). Este comportamiento

llevó a investigar la línea de alimentación.

Se concluyó que o bien, no era suficiente la capacidad de la fuente, o el transistor Q1 no estaba en una correcta polarización, impidiendo el pasaje de la corriente necesaria. Para definir el problema, se planteó medir la diferencia entre el nodo Q_{S1} y Q_{D1}. En el nodo Q_{S1} se tiene el voltaje de la fuente (y una pequeña caída de voltaje de los diodos de entrada). En el nodo Q_{D1} queda evaluada la caída de potencial en el transistor, considerando la diferencia con Q_{S1}. De esta manera se puede discriminar si el problema proviene de la fuente o del transistor.

Se midieron los voltajes en los puntos Q_{G1}, Q_{S1} y Q_{D1}. Las medidas, presentadas en la Tabla A.1, fueron realizadas con un multímetro sin tener las baterías conectadas y con una fuente de 5 V como alimentación.

Q1	Solo encendido (V)	Máxima potencia de sonido (V)	Movimiento del servomotor (V)
V_{G1}	0,0	0,0	0,0
V_{S1}	4,4	4,5	4,5
V_{D1}	4,7	4,0	2,0

Tabla A.1: Polarización del transistor IRF9630.

El sistema funciona correctamente con el sonido en su máxima potencia y el servomotor desconectado. Se aprecia cierta diferencia de voltaje en V_{D1}. Al conectar el servomotor al sistema, este último colapsa. Al tener 2 V en V_{D1} cuando se conecta el servomotor, los *boosters* no son capaces de generar los 5 V necesarios para el sistema. Dado que es un efecto transitorio se conectó un osciloscopio, obteniéndose los resultados de la Figura A.2.

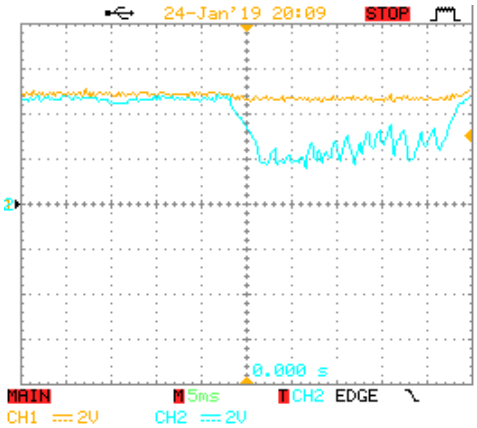


Figura A.2: Medición de la polarización del transistor IR9630 (CH1 = V_{S1} y CH2 = V_{D1}).

Con los mismos se deduce que la fuente no presenta ningún problema ya que V_{S1} permanece constante. Por otro lado, la curva V_{D1} cae abruptamente al mover el servomotor. Esto lleva a concluir que el transistor esta teniendo una caída de potencial de 3 V entre *drain* y *source*, por lo que podemos interpretar que:

- Cuando el servomotor demanda corriente, la impedancia vista hacia los *boosters* cae considerablemente, haciendo que la impedancia del transistor sea comparable y/o mayor a la del sistema. Esto produce que la caída de potencial sea mayor en el transistor y menor en el *booster*, haciendo caer la eficiencia del mismo.

- El voltaje V_{GS1} no es suficiente para ese tipo de transistor. Es decir no logra una polarización que permita dejar pasar la corriente necesaria.

Frente a este problema se plantearon dos soluciones. O bien se eleva el voltaje de alimentación, o bien se buscaba un transistor que pudiera tener una menor impedancia a menor voltaje en el terminal *source*. Se optó por el cambio del transistor ya que era la solución más simple.

La curva de corriente en función de V_{GS} del transistor IRF9630 se muestra en la Figura A.3. Se observa que no hay curvas para voltajes menores a 4,5 V. Se recuerda que la batería genera un voltaje de 3,6 V, por lo que no se puede estimar con exactitud con la información del fabricante. De todas maneras, para continuar con el análisis, se considera el mejor caso. Es decir, tener como entrada la fuente de 5 V y según la Sección 6.4, una corriente demandada de aproximadamente 3 A.

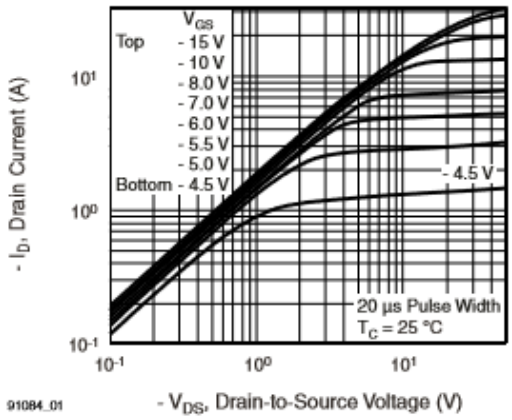


Fig. 1 - Typical Output Characteristics, T_C = 25 °C

Figura A.3: Curva I_D vs. V_{DS} del transistor IRF9630, extraída de [28].

Según la Figura A.3 tenemos que el voltaje V_{DS} a partir de los 3 V se acerca a los 2 A. El problema radica, en que la curva tiende a volverse una constante, por lo que para permitir dejar pasar más corriente es necesaria una gran diferencia de potencial. Esto, fundamenta el problema descripto al principio. Por lo que para solucionarlo se precisa un transistor que tenga una curva I_D vs. V_{DS}, tal que, al menos logre alcanzar una I_D aproximadamente de 3 A con un el voltaje V_{DS} menor posible.

Se investigó y se seleccionó el transistor NDP6020P. En la Figura A.4 se observa que teniendo un $V_{GS} = 3{,}5$ V, es suficiente una diferencia $V_{DS} = 0{,}5V$ para tener una $I_D = 10$ A.

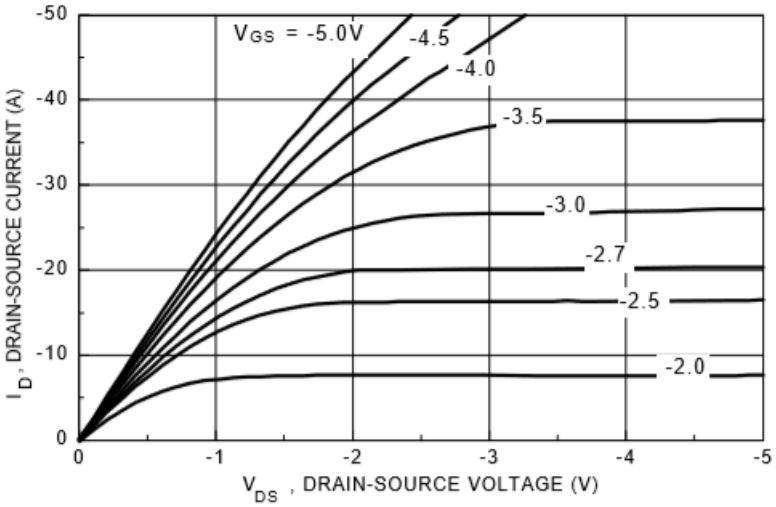


Figura A.4: Curva I_D vs. V_{DS} del transistor NDP6020P, extraída de [24].

Se probó el nuevo transistor en condiciones iguales al transistor anterior, es decir sin batería y con una fuente de 5 V. El resultado se muestra en la Figura A.5.

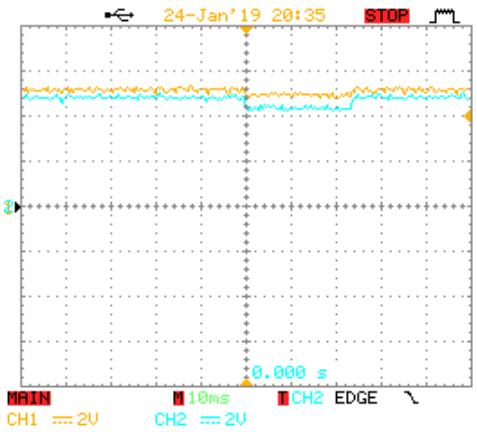


Figura A.5: Medición de la polarización del transistor NDP6020P (CH1 = V_{S1} y CH2 = V_{D1}).

En el gráfico del transistor NDP6020P la diferencia de potencial de V_{DS} es del orden de 0,5 V. Con esa diferencia de potencial los *boosters*, aún pueden trabajar sin demandar mucha cantidad de corriente. El problema quedó resuelto para una fuente de alimentación de 5 V. Para ver el desempeño si se tiene alimentación desde las baterías de 3,6 V se hizo el mismo ensayo. El resultado se muestra en la Figura A.6.

Como primera observación del gráfico de la Figura A.6, se tiene que el voltaje de la batería es aproximadamente 3,6 V. Cada caída de potencial corresponde a un movimiento del servomotor. Como se puede observar, nuevamente se pierde

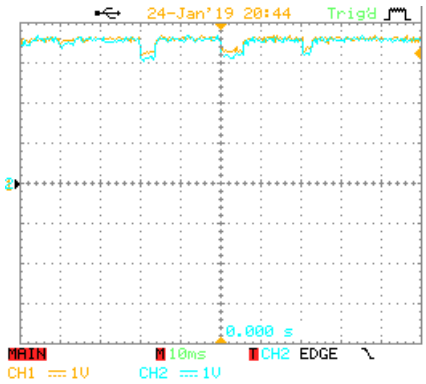


Figura A.6: Medición de comportamiento del transistor NDP6020P alimentando al sistema con las baterías (CH1 = V_{S1} y CH2 = V_{D1}).

0,5 V por lo que los *boosters* terminan recibiendo como voltaje de entrada 3,1 V. Se considera que estos resultados, son adecuados para permitir que el sistema funcione forma correcta.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice B

Análisis de costos

En este apartado se analizan en detalle los costos de fabricación del Bandonberry así como los criterios para la elección de los componentes.

B.1. Criterios de elección general

El criterio de compra consistió en comparar los precios entre los que se ofrecían en el exterior y los del mercado local. En el caso de ser más mucho más económicos en el exterior, se importaba, de lo contrario se resolvía la compra en el mercado local. Por otro lado, hubo casos donde se tuvo que comprar componentes en el mercado local, debido a que los tiempos de envío representaban retrasos en la planificación del proyecto.

B.2. Componentes y materiales

En la Tabla B.1 se detallan los costos de los componentes y materiales utilizados para la construcción del Bandonberry. Se aclara que no se incluyen los costos de importación y de carga impositiva.

La mayoría de los componentes fueron adquiridos en el mercado exterior debido a sus menores costos. La excepciones incluyen el filamento para impresión 3D, el servomotor y los tornillos, los cuales fueron adquiridos en el mercado local.

Componente / Material	Cant.	Unitario (U$)	Subtotal ($)
Raspberry Pi Zero W	1	280,60	280,60
Conversor I2S DAC 3W	2	166,96	333,92
Expansor I/O MCP23S17-E/SP I2C SPI	2	27,49	54,98
Placa de carga Micro USB 5V 1A 18650 Lithium Battery	5	10,20	51,00
Leonardo Pro Micro ATmega32U4	1	169,71	169,71
Conversor de voltaje DC-DC 2V-5V to 5V Boost Power	2	33,94	67,88
Parlante Black 3070 4 Ohm 3W	2	34,56	69,12
Switch pulsador 6x6x6 mm	71	0,51	36,21
Diodo 1N4148	71	0,40	28,40
Diodo SR540	2	10	20
Zócalos de 28 Pines Narrow DIP IC	2	2,96	5,92
Tornillo con tuerca	50	2,20	110,00
Sensor de presión BMP180	1	552,00	552,00
PCB Derecha	1	705,21	705,21
PCB Izquierda	1	527,01	527,01
Pantalla OLED 0.91” 128x32 IIC I2C White	1	88,22	88,22
40 Pines angulares 2.54mm Single Row	1	2,78	2,78
Cable extensor montable Micro USB Male to Female	1	109.79	109,79
Servomotor Tower Pro SG90	1	150,00	150,00
Conectores de plástico 2.54mm JST XH 2 3 4Pin macho y hembra	60	1,94	116,40
Cables jumper macho-macho 20cm 2.54MM 1P-1P	20	2,05	41,00
PCB mother	1	92,80	92,80
Transistor MOSFET NDP6020P-ND	2	106,20	212,40
Sparfun LiPo Fuel Gauge	1	387,44	387,44
1Kg de filamento PLA blanco 1.75mm Reach	1	895,00	895,00
1Kg de filamento PLA blanco 1.75mm Wanhao	2	840,00	1.680,00
		Total($):	**6.767,79**

Tabla B.1: Cantidad y costos de los insumos utilizados en la construcción del Bandonberry.

B.3. Horas de impresión 3D

Para tener una referencia de algunas cifras relativas a la impresión se muestran algunos datos en la Tabla B.2 y B.3 .

	Tiempo (h)	**Peso (g)**
Carcasa derecha	117	1.021
Carcasa izquierda	138	898
Total	**255**	**1.919**

Tabla B.2: Tiempo y material por carcasa

Costo ($U)	**Energía**	**Filamento**	**Subtotal**
Carcasa derecha	117	911	1.028
Carcasa izquierda	134	802	904
		Total	**1.932**

Tabla B.3: Costos de impresión 3D.

La impresora consume 200W y el costo de energía es 5 $U el KWh. Por lo tanto se gasta $ 1 por hora de impresión.

B.4. Horas de mano de obra generales

El tiempo de mano de obra también se debe considerar como un costo. En la Tabla B.4 se muestran las actividades más complejas que son necesarias para el armado del Bandonberry, partiendo de la base que ya se tienen todos los materiales y *software* disponibles.

Pieza	**Concepto**	**Tiempo (h)**
PCBs	Soldar componentes	3 - 4
Carcasa	Atornillar, pegar piezas, cablear y fijar electrónica	3 - 4
Fuelle	Medir, cortar y plegar	20 - 25
	Total	26 - 33

Tabla B.4: Mano de obra estimada general.

B.5. Costos totales

Los costos totales de fabricación del Bandonberry se resumen en la Tabla B.5. Se aclara que el tiempo de mano de obra solo incluye lo explicitado en la sección anterior. No incluye los tiempos de programación, de importación, etc.

Materiales	**Impresión**	**Mano de obra**
6768 $	14 días	26-33 h

Tabla B.5: Resumen de costos.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice C

Descripción general de los estándares eléctricos utilizados

En este apartado se describen brevemente los protocolos de comunicación utilizados por la SBC para comunicarse con algunos de sus periféricos.

C.1. I2S

El estándar I2S, o I^2S, de la sigla en ingles *Integrated Interchip Sound*, se utiliza fundamentalmente para la trasmisión serial de datos de audio entre dos dispositivos.

Dicho estándar consiste en tres señales digitales:

- SCK - *Serial Clock* - Señal de reloj.
- WS - *Word Select* - Indica el canal de audio ("0" izquierdo, "1" derecho), cuando se tiene audio estéreo.
- SD - *Serial Data* - Línea de transmisión de datos.

La conexión se muestra en la Figura C.1.

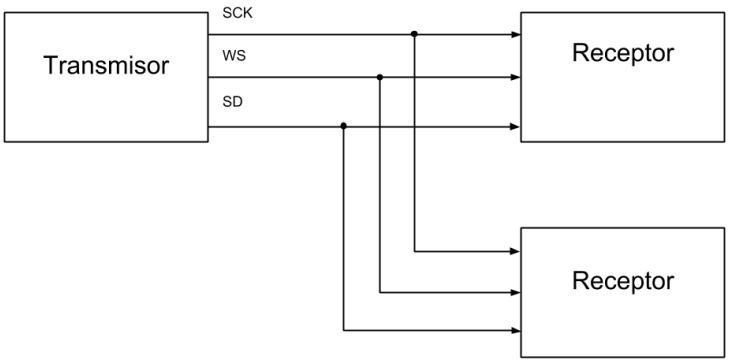


Figura C.1: Conexión I2S.

C.2. SPI

El estándar SPI, de la sigla en inglés *Serial Peripheral Interface*, se utiliza para la comunicación de datos entre circuitos integrados. A diferencia de la comunicación I2C, esta utiliza una mayor cantidad de líneas, pero ofrece mayores velocidades de comunicación.

Dicho estándar consiste en cuatro señales digitales:

- SCLK - *Serial Clock* - Señal de reloj.
- MOSI - *Master Output Slave Input* - Línea de transmisión de datos del maestro al esclavo.
- MISO - *Master Input Slave Output* - Línea de transmisión de datos del esclavo al maestro.
- CS - *Chip Select* - Señal que utiliza el maestro para seleccionar al esclavo.

La conexión SPI consiste de un dispositivo maestro y uno o varios esclavos, tal como se muestra en la Figura C.2. Para que el maestro logre seleccionar el esclavo a ser consultado, la línea CS correspondiente es activada.

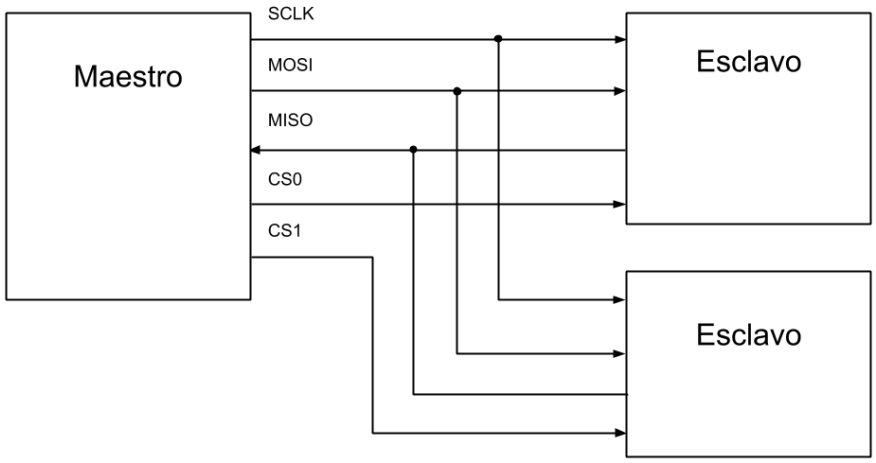


Figura C.2: Conexión SPI.

C.3. I2C

El estándar I2C, o I^2C, de la sigla en inglés *Inter-Integrated Circuit*, se utiliza para la comunicación de datos entre circuitos integrados. Se caracteriza por solo tener dos señales, lo que simplifica mucho su funcionamiento a nivel del cableado.

Dicho estándar consiste en dos señales digitales:

- CLK - *Serial Clock* - Señal de reloj.
- SDA - *Serial Data* - Línea de transmisión de datos.

La conexión I2C consiste de un dispositivo maestro y uno o varios esclavos, tal como se muestra en la Figura C.3. Para que el maestro logre seleccionar el esclavo a ser consultado, el protocolo exige que se defina una dirección única para cada esclavo. De esta manera, el maestro envía en el bus la dirección asociada al esclavo deseado. Este último se activa y responde a dicha consulta, mientras que los demás esclavos se mantienen pasivos para no ocupar el bus de datos.

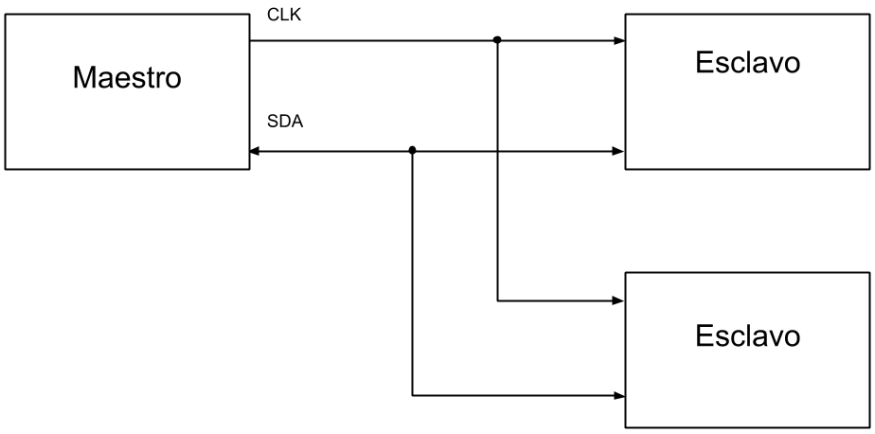


Figura C.3: Conexión I2C.

Apéndice D

Conceptos básicos de impresión 3D

En este apéndice se explican de formas generales algunas características de las impresoras 3D de filamento, así como también ciertos factores y consideraciones que influenciaron fuertemente en la fabricación del Bandonberry.

D.1. Impresora 3D y terminología general

Hay distinto tipos de impresoras 3D. Existen dos grandes grupos, las que funcionan por fundición de filamento plástico y las que funcionan por solidificación de resina por disparo de luz. Para el proyecto se utilizó la tecnología de fundición de filamento plástico, por lo que todo lo que sigue a continuación es relativo a dicha tecnología.

El sistema general de las impresoras 3D consiste en una punta metálica, denominada extrusor, con un canal en su centro, el cual se mantiene a altas temperaturas (superior a 180°C). Por el extrusor pasa un filamento de material plástico, el cual se funde y sale en estado casi líquido. En la salida del extrusor hay un ventilador de enfriamiento, que por convección enfría el plástico solidificándolo rápidamente. El plástico se va depositando sobre una superficie especial, denominada cama, la cual a su vez se calienta a una temperatura aproximadamente de 50°C. Esto genera una mayor adherencia de la primer capa de plástico a la superficie de la cama.

El filamento se vende en bobinas o carretes. Dicho filamento es movido por un motor paso a paso el cual va empujando el filamento por el extrusor dependiendo de la demanda de material y la velocidad de impresión.

La impresora mueve el extrusor con respecto a la cama, depositando capas de material plástico. Superponiendo capa a capa, se logran generar los volúmenes. La altura de las capas son en general menores al milímetro. Todos los movimiento son realizados por varios motores paso a paso, que funcionan de forma coordinada, para poder recorrer todos los puntos necesarios del espacio X, Y, Z. Todo el control de los motores, temperaturas, tiempos, ventiladores, etc., se realizan con una placa electrónica.

Algunas impresoras mueven el extrusor en las coordenadas X, Y y Z. Otras

mueven el extrusor solo en la coordenada Z y la cama en las coordenadas X e Y. También existen impresoras que tienen más de un extrusor para imprimir con colores o materiales diferentes a la vez. En el caso de la fabricación del Bandonberry se utilizó una impresora del estilo Prusa I3. Las impresora 3D, de bajo costo y sencillas como las que se tenían a disposición, aún no tienen la autonomía ni robustez como para que se pueda confiar que solo oprimiendo un botón y esperando algunas horas se obtenga la pieza deseada. Es muy importante estar controlando el proceso en general. Las primeras dos capas de impresión son críticas para todo el desarrollo del proceso, por lo que es de vital importancia estar presente.

D.2. Diseñar para impresoras 3D

La impresión 3D consiste en capas superpuestas de PLA, por lo que es necesario que debajo de cada punto de una capa exista una capa previa que logre sostenerla. Este detalle condiciona fuertemente la forma de la pieza y la orientación relativa frente a la impresora, es decir cual cara de la pieza será la base de apoyo.

Para dar un ejemplo, se supone el caso de la pieza que se muestra en la Figura D.1. Tiene como objetivo ser atravesada por un tornillo que enrosca en una tuera que va dentro del hueco rectangular.

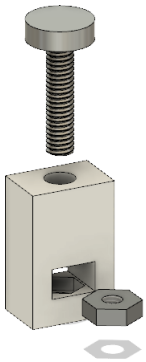


Figura D.1: Pieza utilizada como soporte de unión utilizando un tornillo y una tuerca.

Para imprimir cada pieza de forma correcta, se debe elegir de forma inteligente la orientación relativa a la impresora. Dicha orientación define fuertemente el tiempo de impresión y la cantidad de material que se consume para construir el soporte.

En la Figura D.2 se tiene una pre-visualización de una impresión utilizando el software Repetier definiendo distintas orientaciones relativas la cama. En dicha figura se aprecia como la estructura del soporte varía en cada configuración.

La opción B es la menos compleja ya que todo el eje, en el que la pieza modelada es hueca, se mantiene hueco, a excepción del espacio donde va la tuerca. En la opción A y C todo el eje queda relleno con material de soporte. Una vez finalizada la pieza, en las opciones A y B, es difícil de remover el soporte dado el espacio reducido y la gran adherencia del soporte a la pieza por la cercanía de las paredes.

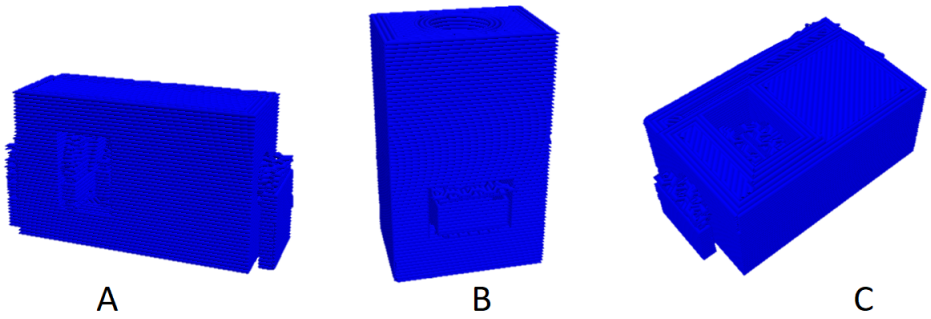


Figura D.2: Pre-visualización de impresión de pieza.

La pieza de la Figura D.1 debería formar parte de otra para generar las uniones, de forma tal que toda la estructura se pueda armar y desarmar utilizando tornillos y tuercas. En el ejemplo antes mencionado la orientación relativa a la cama era muy simple de definir. El problema radica cuando las piezas adquieren complejidad como se muestra en la Figura D.3.

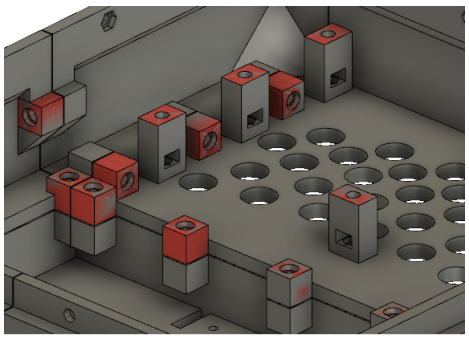


Figura D.3: Interior de caja de resonancia izquierda.

En algunos casos, como el de la pieza del ejemplo de la Figura D.1 de forma independiente y luego se unen con pegamento para plástico a las piezas más grandes.

Para solucionar este problema existen impresoras más complejas que tienen un material independiente para los soportes, el cual es soluble en agua. Una vez finalizada la pieza se sumerge en agua y luego de un tiempo el soporte queda disuelto y la pieza principal queda sin ningún daño.

D.3. Tipos de filamentos

Aunque todos los filamentos sean del mismo material PLA, es normal que si se cambia la marca del producto hayan varios parámetros que tengan que ser reajustados. Estos parámetros son la temperatura de fusión, velocidad del ventilador de enfriamiento, temperatura de la cama, entre otros.

En una primera instancia se utilizó el PLA comprado en la distribuidora Fabrix, y marca REACH RoHS. Luego se agotó el producto en el mercado local y se utilizó el PLA + (También distribuido por Fabrix) el cual tenía serios problemas de adherencia a la cama de la impresora. Finalmente se terminó utilizando un filamento, comprado en Robotec de marca Wanhao premium quality, con el que se obtuvieron buenos resultados. Todos los filamentos mencionados fueron de 1,75 mm de diámetro.

D.4. Tolerancias y ruido de impresión

Todas las impresoras tienen cierto "ruido" al imprimir. Es decir, que la pieza difícilmente quede exactamente como uno se diseña. El ruido se genera por una gran cantidad de factores, desde la temperatura del extrusor, vibraciones, la velocidad del ventilador de enfriamiento, mala calibración de las referencias de coordenadas (en particular la Z), entre otros.

Dicho ruido influye en las tolerancias que se tienen que considerar para el diseño, en particular para la situación de encastre de piezas. En función de la experiencia, se definió una distancia de 0,4 mm entre pieza y pieza como tolerancia.

D.4.1. Configuraciones básicas estructurales

Los criterios relativos a las densidades de material y espesores de las paredes se basan en la experiencia generada por el ensayo y error. Dichos parámetros se definieron con los siguientes valores y se puede apreciar su correspondencia con la impresión.

- Densidad: 20 %
- Ancho de paredes: 0,5 mm
- Estructura interna de construcción: Hexagonal (Panal de abejas)
- Cantidad de capas superiores: 3
- Cantidad de capas inferiores: 3

Apéndice E

Opción alternativa de cargador

Con respecto al tiempo de carga, se considera que se puede mejorar notoriamente. Se investigó y se encontró un módulo similar, mostrado en la Figura E.1, pero con dos diferencias. La primera es que en lugar de tener un solo integrado TP45056, tiene 4. Esto le permite cargar la batería con 3 A máximo de corriente. La segunda diferencia, es que no posee las protecciones de sobrecarga y sobre corriente.

De todas formas, las protecciones se pueden implementar de forma independiente. La protección de sobrecarga y sobredescarga se puede implementar con el microcontrolador. El uso de un fusible es suficiente para tener la protección de sobrecorriente y asegura de la mejor manera la integridad física de todo el sistema.

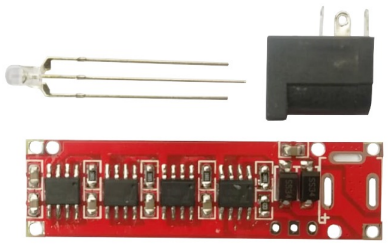


Figura E.1: Cargador de 3 A con 4 integrados TP45056.

El módulo de la Figura E.1 no se llegó a probar, pero dada dicha corriente de carga (3 A) se estima que el tiempo de carga total pase de 20 a 6 horas. Como contraparte la fuente externa deberá ser capaz de entregar 5 A.

Esta página ha sido intencionalmente dejada en blanco.

Referencias

[1] Acrylonitrile butadiene styrene. Recuperado de https://en.wikipedia.org/wiki/Acrylonitrile_butadiene_styrene. Accedido 04/02/19.

[2] Bandoneón. En Wikipedia. Recuperado de https://es.wikipedia.org/wiki/Bandone%C3%B3n. Accedido 28/02/2019.

[3] El bandoneón Pichuco de la UNLa ganó el premio INNOVAR 2012. *Cancioneros.com*, Diciembre 2012.

[4] Adafruit. OLED Display. Recuperado de https://cdn-learn.adafruit.com/downloads/pdf/monochrome-oled-breakouts.pdf. Accedido 30/05/2018.

[5] Advanced Monoloithic Systems. AMS1117 1A low dropout voltage regulator. Recuperado de http://www.advanced-monolithic.com/pdf/ds1117.pdf. Accedido 18/04/2019.

[6] Distribuidor: airsoftparkstore. DC-DC Step Up Módulo de Fuente de alimentación 2V-5V a 5V 2A salida fija Booster. Recuperado de https://www.ebay.com/itm/10x-DC-DC-Step-Up-Power-Supply-Module-2V-5V-to-5V-2A-Fixed-Output-Booster/293023228777?epid=16005045186&hash=item44398ba369:g:3~4AAOSwKA5b12yT. Accedido 18/04/2018.

[7] Martin Borteiro. Relevamiento de materiales históricos de tango. *Patrimonio vivo de Urugauay, relevamiento de tango*, pages 9,67, Abril 2015. Editorial: Ministerio de Educación y Cultura.

[8] BOSCH. BMP180 - Digital pressure sensor. Recuperado de https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BMP180-DS000.pdf. Accedido 21/06/2018.

[9] NanJing Top Power ASIC Corp. TP4056 1A Standalone Linear Li-lon Battery Charger with Thermal Regulation in SOP-8. Recuperado de https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf. Accedido 11/03/2019.

[10] EEMB. Lithium-Ion battery data sheet. Recuperado de https://www.ineltro.ch/media/downloads/SAAItem/45/45958/36e3e7f3-2049-4adb-a2a7-79c654d92915.pdf. Accedido 11/06/2018.

[11] Federico Fraysse. Bandoneón Electrónico. Recuperado de https://drive.google.com/file/d/1dbgtEX5pBRxQnG1RiUmb0zrx0t_82Y01/view?usp=drivesdk. Accedido 25/3/2019.

[12] Chris Hager. RPIO.PWM, PWM via DMA for the Raspberry Pi. Recuperado de https://pythonhosted.org/RPIO/pwm_py.html. Accedido 2018-10-4.

[13] H&Msemi. DW01A. Recuperado de http://hmsemi.com/downfile/DW01A.PDF. Accedido 11/03/2019.

[14] Hottech Semiconductor. 8205A. Recuperado de https://www.maritex.com.pl/product/attachment/91261/8205A.pdf. Accedido 11/03/2019.

[15] Maxim Integrated. 1-Cell/2-Cell Fuel Gauge with ModelGauge and Low-Battery Alert. Recuperado de https://datasheets.maximintegrated.com/en/ds/MAX17043-MAX17044.pdf. Accedido 16/04/2019.

[16] Maxim Integrated. MAX98357 - Tiny, Low-Cost, PCM Class D Amplifier with Class AB Performance. Recuperado de https://datasheets.maximintegrated.com/en/ds/MAX98357A-MAX98357B.pdf. Accedido 07/04/2019.

[17] Io. SR520 THRU SR5100. Recuperado de http://pdf.datasheetcatalog.com/datasheet/bytes/SR530.pdf. Accedido 07/04/2018.

[18] Mercedes Krapovickas. Organografía del bandoneón y prácticas musicales: Lógica dispositiva de los teclados del bandoneón Rheinische Tonlage 38/33 y la escritura ideográfica. *Latin American Music Review*, 33(2):157–185, 2012.

[19] L Martin, O; Avérous. Polylactic acid: plasticization and properties of biodegradable multiphase systems. 2001.

[20] Microchip. MCP23S17 - 16-Bit SPI I/O Expander with Serial Interface. Recuperado de https://www.microchip.com/wwwproducts/en/MCP23S17. Accedido 07/04/2019.

[21] Alejandro Palladino. Bandoneón Pichuco: el descendiente de Aníbal Troilo. *Diario Contexto*, Octubre 2015.

[22] Tower Pro. Servomotor SG90. Recuperado de http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf. Accedido: 10/10/2018.

[23] Angelo Nagari Rémy Cellier, Gaël Pillonnet and Nacer Abouchi. An Review of Fully Digital Audio Class D Amplifiers topologies. Agosto 2009.

[24] ON Semiconductor. NDP6020P/NDB6020P P-Channel Logic Level Enhancement Mode Field Effect Transistor. Recuperado de https://www.onsemi.com/pub/Collateral/NDP6020P-D.PDF. Accedido 24/01/2019.

[25] MIT Electric Vehicle Team. A guide to understanding battery specifications. Recuperado de http://web.mit.edu/evt/summary_battery_specifications.pdf. Accedido 04/05/2018.

[26] UNESCO. Decision of the Intergovernmental Committee (4.COM 13.01). 2009. Recuperado de https://ich.unesco.org/en/decisions/4.COM/13.01.

[27] Python Software Fundation (US). Time access and conversions. Recuperado de https://docs.python.org/3.0/library/time.html#time.sleep. Accedido 12/02/2019.

[28] Vishay Siliconix. Power MOSFET IRF9630 SiHF9630. Recuperado de https://www.vishay.com/docs/91084/sihf9630.pdf. Accedido 11/06/2018.

[29] Marcus Weseloh. Fluidsynth-Lowlatency. Recuperado de https://github.com/FluidSynth/fluidsynth/wiki/LowLatency. Accedido 28/04/2018.

[30] Wikipedia. Bandoneón. Recuperado de https://es.wikipedia.org/wiki/Bandone%C3%B3n. Accedido 02/03/2019.

Esta página ha sido intencionalmente dejada en blanco.

Índice de tablas

Esta página ha sido intencionalmente dejada en blanco.

Índice de figuras

Esta es la última página.
Compilado el miércoles 16 octubre, 2019.
http://iie.fing.edu.uy/

Bandonberry: simulador electrónico de bandoneón

José Bentancour
Egresado de la UDELAR
Montevideo
josebentancour@gmail.com

Franco Toscano
Egresado de la UDELAR
Montevideo
francotoscano92@gmail.com

Rodrigo Patiño
Estudiante de la UDELAR
Montevideo
rodmax77@gmail.com

Ignacio Irigaray
Docente del IIE
Montevideo
irigaray@fing.edu.uy

Pablo Zinemanas
Docente del IIE
Montevideo
pzinemanas@fing.edu.uy

***Abstract*— Bandonberry tiene como objetivo crear un simulador electrónico para el aprendizaje del bandoneón. Pretende ser una alternativa económica en comparación con un bandoneón estándar, de forma de popularizar el acceso al instrumento por parte de los estudiantes. No solo se limita a reproducir su forma y sonido, sino que permite incorporar tecnologías que podrían llegar ayudar y estimular el proceso de aprendizaje.**

Keywords*— *bandoneon, raspberry, bandonberry, tango

I. Introducción

El tango es una de las expresiones artísticas más importantes del siglo XX, en el año 2009 fue declarado Patrimonio Cultural Inmaterial de la Humanidad por la UNESCO [1]. Dentro de los instrumentos utilizados en el tango, el bandoneón es quizás el que más define su sonido. Se fabricaron bandoneones de forma industrial hasta la década del 60 y actualmente son fabricados de forma artesanal por algunos pocos lutieres. Esto conlleva que el costo del instrumento sea alto (en el orden de 3000 USD) y por lo tanto inaccesible para la gran mayoría de los estudiantes de música. Muchas veces los estudiantes no tienen acceso al mismo, o lo tienen solo parcialmente en horarios limitados de práctica establecidos por el instituto de enseñanza. Como ejemplo, podemos citar el caso de la Escuela Municipal de Música Vicente Ascone (EMVA) de la ciudad de Montevideo, que cuenta con más estudiantes que bandoneones (2018).

Con respecto al bandoneón en sí, es un instrumento con un comienzo en el aprendizaje complejo, ya que la disposición de las notas en la botonera es muy particular. A su vez, se agrega la complejidad que la distribución de las notas cambia según la dirección del movimiento del fuelle. Esto hace que sea fundamental para el estudiante tener el instrumento a disposición para adquirir la práctica necesaria.

En el presente proyecto se busca crear un simulador del bandoneón que tenga un menor costo, con el afán de mitigar la barrera económica, democratizar y popularizar el instrumento.

Existen dos iniciativas con objetivos similares a este proyecto. Una de ellas es el proyecto “Pichuco”, realizado en Argentina en la Universidad de Lanus, que busca utilizar tecnologías de producción modernas, para fabricar bandoneones de manera industrializada a gran escala, teniendo la particularidad de un bajo costo. El proyecto comenzó en 2009 y llegaron a fabricarse prototipos en 2013, pero no se lograron producir a grandes cantidades. Por otro lado, hay otro proyecto Argentino realizado de forma paralela al Bandonberry por Federico Fraysee, llamado “Bandoneón electrónico”. El mismo es similar en objetivos al Pichuco y Bandonberry pero concebido desde un enfoque más mecánico.

II. Descripción del prototipo

En la Figura 1 se puede ver el diagrama de bloques del prototipo. El principal bloque de procesamiento es la Single Board Computer (SBC) que controla la comunicación USB, la síntesis de sonido y la salida de audio. La interfaz USB permite enviar las notas tocadas a un PC, emulando ser un teclado MIDI. Esto permite usar el dispositivo junto con programas de aprendizaje musical (por ejemplo, PianoBooster). Además, el SBC es el encargado de reconocer qué botones están apretados y medir las señales de los sensores. El sistema es portable y autónomo, por lo que puede ser alimentando con una fuente externa o un banco de baterías. Para ello, se diseñó un sistema de control de batería (BMS).

A. Single Board Computer (SBC)

Para sintetizar el sonido y leer los datos de entrada del fuelle y botoneras se utiliza una SBC, en particular una Raspberry Pi Zero W. Al conectarla mediante el puerto USB a una computadora se presenta como un dispositivo MIDI y envía la información de las notas con su respectiva dinámica dada por la presión del fuelle.

B. Battery Manager System (BMS)

El BMS es un sistema de carga y protección de las baterías de polímero de litio (LiPo) que alimentan al prototipo. Está en desarrollo una interfaz de control (User Interface) con display y botones, donde se brinda información del estado de la batería permitiendo el encendido y apagado del Bandonberry. Este sistema utiliza un microcontrolador del tipo Arduino.

C. Botoneras

Como se puede observar en la Figura 1 las botoneras poseen una gran cantidad de botones (71 en total) y es necesario detectar el accionar de distintos botones de manera simultánea. Dado que la Raspberry Pi Zero posee solo 26 entradas digitales se utiliza un módulo expansor de puertos MCP23S17 [5] que implementa una interfaz serie-paralelo (SPI) para lograr la cantidad de entradas extra necesarias.

D. Fuelle

Para la detección del movimiento del fuelle se utiliza un sensor de presión BMP180 [6], el cual se comunica de forma serial mediante I2C. La diferencia con la presión atmosférica nos permite detectar el sentido del movimiento y la presión ejercida. El sentido del movimiento es utilizado para configurar la disposición de las notas en las botoneras, mientras que la presión controla la dinámica del sonido. Se espera que el fuelle brinde diferentes niveles de resistencia dependiendo de la cantidad de botones presionados. Para lograr controlar el flujo de aire se diseñaron válvulas en base a servomotores que también son controlados por la SBC.

E. Salida de audio

La etapa de salida de audio toma la información de entrada en forma digital y provee dos salidas analógicas de 3 W de potencia para alimentar a los parlantes. Se utilizan dos circuitos integrados MAX98357A [7], los cuales se conectan por I2S a la SBC.

F. Software

Para generar el sonido de bandoneón utilizamos el sintetizador Fluidsynth [8] en conjunto con una biblioteca de sonidos de uso libre creado por Jorg Bleymehl [9]. La comunicación entre los diferentes m´odulos fue realizado en el lenguaje Python (por ejemplo para leer el teclado y estado del fuelle y enviar la nota correspondiente al sintetizador) utilizando para ello mensajes MIDI.

G. Diseño Físico

Las tapas y los teclados de Bandonberry fueron dice ˜nados en base a un bandoneón marca ELA de 1929 y fabricados con impresora 3D (ver Figura 3). El fuelle fue fabricado en base a cartón plegado. Utilizando el origami como inspiración se modificó la forma clásica del fuelle por una que permita un proceso constructivo más simple, como se puede observar en la Figura 2.

H. Conclusiones y trabajo a futuro

El avance del proyecto hasta el momento muestra que es posible construir un bandoneón que permite simular la dinámica del fuelle y la botonera. El diseño del circuito, software y PCBs se encuentran totalmente desarrollados para ambas botoneras. Se cuenta con la construcción total de la tapa derecha y el fuelle y se piensa completar la botonera izquierda en los próximos meses de forma de tener un prototipo completamente funcional.

Hace faltan pruebas por bandoneonistas profesionales, que ya están previstas en la planificación del proyecto, para obtener información que nos permita mejorar el prototipo. Además, se espera que los estudiantes de la EMVA se apropien de la iniciativa de forma de obtener una realimentación sobre su utilidad como herramienta pedagógica.

El proyecto es de software y hardware libre, lo que implica que todos los archivos de diseño, modelos 3D, circuitos y código producidos serán publicados. Esto permite que el proyecto se pueda replicar (e incluso mejorar) por otras personas que quieran beneficiarse de la herramienta que se está generando.

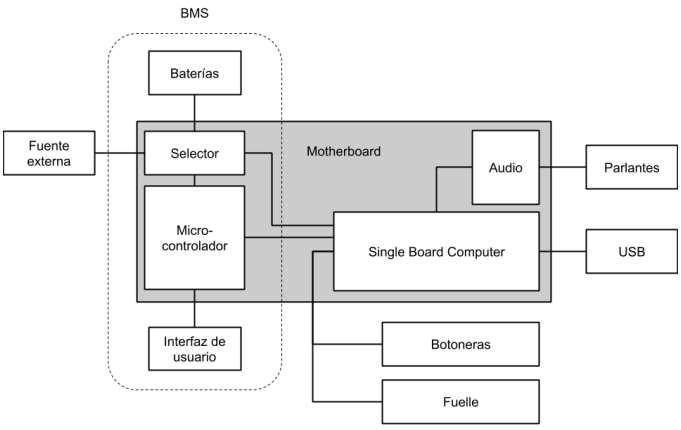


Figura 1

Bandonberry

Simulador electrónico de bandoneón

Franco Toscano, Rodrigo Patiño,
José Bentancour.

IIE

Ingeniería deMuestra
2019 ★ Vení a conocer el futuro

Motivación

Los bandoneones se fabricaron de forma industrial hasta la década del 1960. Actualmente, son fabricados de forma artesanal por algunos pocos luthiers. Consecuentemente, el costo de un bandoneón estándar es alto, aproximadamente unos 3.000 USD. Esto hace que sea difícil adquirir el instrumento, especialmente para quienes recién se inician, lo que limita la transmisión de la práctica a las nuevas generaciones.

Lo que motiva al proyecto Bandonberry es facilitar el acceso al instrumento. El objetivo primordial es construir un instrumento de práctica, para darle una opción accesible a los interesados en aprender a tocar el bandoneón.

Fuelle

Para la detección del movimiento del fuelle se utiliza un sensor de presión BMP180, el cual se comunica de forma serial mediante I2C.
La diferencia con la presión atmosférica nos permite detectar el sentido del movimiento y la presión ejercida.
El sentido del movimiento es utilizado para configurar la disposición de las notas en las botoneras, mientras que la presión controla la dinámica del sonido

Botoneras

Derecha: treinta y ocho botones conectados en una matriz de 8x6.
Izquierda: treinta y tres botones conectados en una matriz de 8x5.
Palanca mecánica para la toma de aire.
Sistema de resortes para aumentar la excursión de los botones.

Raspberry Pi Zero W

Para generar el sonido de bandoneón utilizamos el sintetizador Fluidsynth en conjunto con una biblioteca de sonidos de uso libre creado por Jörg Bleymehl.
La comunicación entre los diferentes módulos se realiza por MIDI. El software fue desarrollado en Python.

GPIOs

Dado que la Raspberry Pi Zero posee solo 26 entradas digitales se utiliza un módulo expansor de puertos MCP23S17 que implementa una SPI para lograr la cantidad de entradas extra necesarias.
Hay uno en cada botonera y proporcionan 32 GPIOs extra en total.

Servomotor

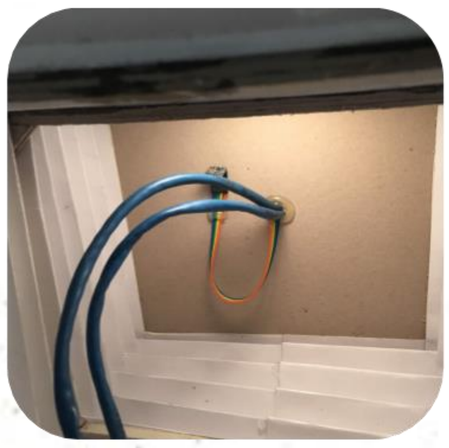
Para lograr controlar el flujo de aire se cuenta con una válvula en base a servomotores.
El fuelle brinda diferentes niveles de resistencia dependiendo de la cantidad de botones presionados.

MIDI USB

Al conectarse a una computadora se presenta como un dispositivo MIDI y envía la información de las notas con su respectiva dinámica dada por la presión del fuelle.

Diseño físico

Las tapas y los teclados de Bandonberry fueron diseñados en base a un bandoneón marca ELA de 1929 y fabricados con impresora 3D.
El fuelle fue fabricado en base a cartón plegado. Utilizando el origami como inspiración se modificó la forma clásica del fuelle por una que permita un proceso constructivo más simple.



El proyecto es de **software y hardware libre**, lo que implica que todos los archivos de diseño, modelos 3D, circuitos y código producidos están publicados **https://github.com/jebentancour/Bandonberry**. Esto permite que el proyecto se pueda replicar (e incluso mejorar) por otras personas que quieran basarse en el trabajo ya creado.

Fundación Julio Ricaldoni
INGENIERÍA EN EL URUGUAY
FACULTAD DE INGENIERIA
UNIVERSIDAD DE LA REPÚBLICA URUGUAY