UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA POLITECNICA SUPERIOR DE GANDIA

Grado en Ing. Sist. de Telecom., Sonido e Imagen

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

ESCOLA POLITÈCNICA
SUPERIOR DE GANDIA

"Soporte robótico para posicionamiento inalámbrico de micrófonos, con motores paso a paso"

TRABAJO FINAL DE GRADO

Autor/a:

Rafa Esplugues Mengual

Tutor/a:

Mª Asunción Pérez Pascual

GANDIA, 2018

RESUMEN

En este proyecto se desarrolla una herramienta para el posicionamiento inalámbrico de la microfonía en distintos entornos de trabajo como por ejemplo estudios de grabación, actuaciones en directo, mediciones acústicas, etc… donde se requiere cierta precisión a la hora de buscar una determinada sonoridad de un instrumento musical, coherencias de fase con las distintas fuentes de captación que se pueden utilizar, o también, por ejemplo, para el posicionamiento del micrófono de medición en ciertos ángulos de captación o distancias concretas, para realizar ciertas mediciones acústica. Esta herramienta que hemos desarrollado cuenta con distintos motores paso a paso para cada uno de sus ejes, los cuales se controlan de forma inalámbrica, concretamente por Bluetooth, mediante una aplicación móvil que hemos implementado.

Palabras clave: Posicionamiento de micrófonos, Arduino, Robot.

ABSTRACT

In this project, a wireless tool for the accurate positioning of microphones under different work environments such as recording studios, live performances, acoustic measurements, etc... has been developed, allowing a high precision in the determination of the sonority of a musical instrument, the adjustment of phase coherence with the different sound input devices or the precise positioning of the measuring microphone: distance, angle, etc.. for acoustic measurements. This tool has several stepper motors, one for each axis, that can be controlled using a especially designed cell phone app via bluetooth wireless technology.

Key words: Positioning of microphones, Arduino, Robot.

INDICE

INDICE DE FIGURAS

INDICE DE PALABRAS CLAVE Y ACRÓNIMOS

INTRODUCCIÓN

Uno de los procesos más arduos que tiene lugar durante la grabación de una pieza en un estudio profesional consiste en el correcto posicionamiento de los micrófonos. Este posicionamiento influye mucho en el resultado final de la grabación, de manera que, si se consigue optimizar la ubicación del micrófono en relación al instrumento y a la sala de grabación, se evitará el tener que realizar un post procesado que muchas veces puede llegar a hacer menos auténtica la experiencia del usuario final.

El posicionamiento de los micrófonos va a influir sobre el timbre o sonido natural deseado dado que éste depende de que las señales captadas al emplear más de una fuente de captación o técnica microfónica tengan una correcta coherencia en fase. Normalmente este proceso se realiza de manera manual, alargando mucho el proceso de grabación y haciéndolo bastante incómodo, inexacto, e irreproducible en la mayoría de los casos que se quiera replicar el montaje para futuras grabaciones.
El mismo problema ocurre en el ámbito de mediciones acústicas, donde muchas veces se pretende evaluar el comportamiento de una fuente o material, posicionando el sonómetro a distintos ángulos o distancias exactas. El realizar de manera manual este proceso resulta muy tedioso e inexacto.

Con la finalidad de automatizar este proceso se ha llevado a cabo el siguiente proyecto, a través del cual no solo se ha realizado la implementación de la estructura mecánica y electrónica del sistema, sino que además se ha implementado el software necesario para controlar todo el sistema de forma inalámbrica para así evitar que el técnico tenga que desplazarse físicamente por la sala de control, manteniendo en todo momento la referencia del sonido sobre el sistema de monitores o las distintas herramientas de control de la señal (Vu-metros, analizadores de fase, RTAS, etc.) .

Para este proyecto se ha utilizado motores paso a paso para controlar los 3 ejes o tipos de movimiento de nuestra estructura (elevación, avance y rotación). Estos motores se caracterizan por su alta precisión de movimiento permitiéndonos realizar movimientos o ajustes muy precisos los cuales se podrán almacenar como "presets" de posiciones en una memoria SD-Card integrada con la finalidad de que el usuario pueda disponer y recurrir a dichos ajustes en todo momento.

La estructura de la memoria de este trabajo final de grado es la siguiente: en primer lugar, presentaremos los objetivos, dividiéndolos en objetivo principal y los objetivos secundarios. En el capítulo 1 se ha realizado una breve vigilancia tecnológica con el objetivo de averiguar qué sistemas de posicionamiento de micrófonos ofrece el mercado actualmente y qué características tienen. En el capítulo 2 se explica todo el proceso que se ha seguido para lograr diseñar y construir una estructura que fuera funcional para nuestra máquina, esta parte ha sido bastante decisiva para determinar la viabilidad del proyecto ya que sin una estructura mecánica adecuada que funcionara, no podríamos seguir con los siguientes puntos. En el siguiente capítulo (capítulo 3), se explica todo lo referente

al hardware utilizado, como por ejemplo se explica la placa de arduino que se ha utilizado junto con la CNC shield y todos los módulos electrónicos que se han añadido como la SD Card, el módulo buletooth, los motores paso a paso, etc. También se muestra todo el conexionado electrónico y la descripción de las principales aplicaciones prácticas para las cuales está pensada esta máquina.
En el siguiente capítulo, una vez ya se ha explicado todo lo referente al tema de hardware necesario para el desarrollo de nuestro prototipo, pasaremos a otro gran bloque muy importante, que es el del desarrollo del software. En este apartado de la memoria, se divide a su vez en dos partes; en primer lugar, el software interno que incorporará la máquina sobre la plataforma arduino, el cual implementará todas las funcionalidades necesarias para que la máquina pueda realizar todos los movimientos y gestión de su memoria interna, y por otro lado, el software que se ha desarrollado para las aplicaciones móviles, donde hemos implementado todo lo referente a las utilidades de control inalámbrico de las distintas funciones de la máquina y la interacción con usuario a la hora de poder crear, utilizar o borrar presets de posiciones. A continuación en el siguiente capítulo se comenta algunas operaciones básicas de mantenimiento, como su correcta lubricación de las partes móviles, y su opción de "calibrado" de los ejes, la cual es muy importante, por ejemplo en casos de fallo de alimentación donde la máquina es posible que pierda su referencia de posición al no poder guardar en su memoria interna, los valores de las variables asociadas a la cuenta del número de pasos de los motores de los distintos ejes.
Por último se establecen las conclusiones y las mejoras que se podrían incluir en el sistema desarrollado, la bibliografía utilizada para reunir todo la información necesaria y el apartado de anexos , donde se complementa la información de distintos puntos como los códigos completos que se han implementado en cada parte, el listado de funciones que hemos tenido que programar, y el desarrollo de todo el presupuesto para la construcción de la máquina.

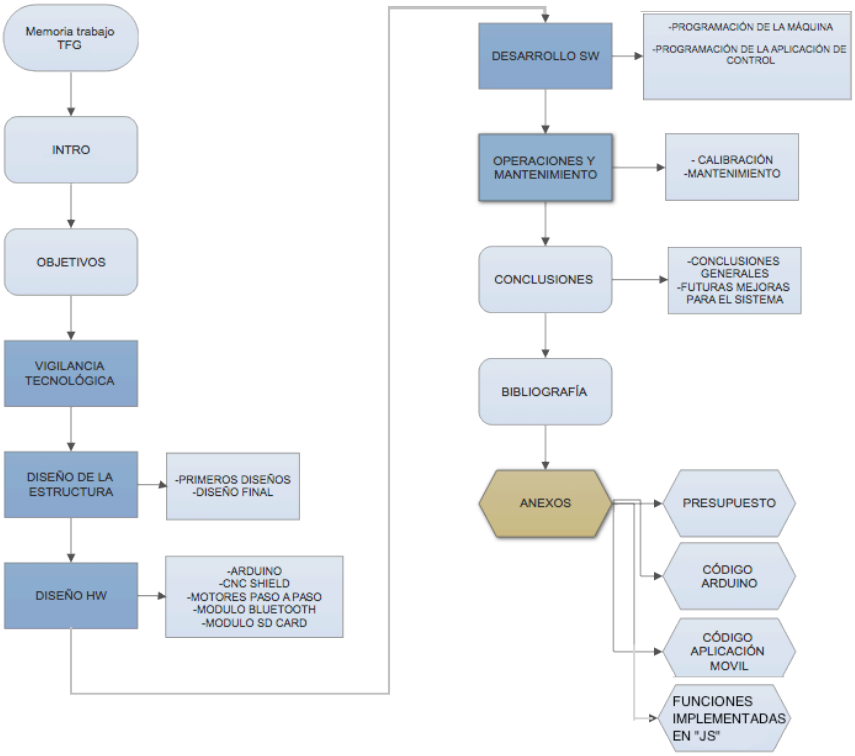


Figura 1 Estructura memoria del trabajo

OBJETIVOS

El objetivo principal del proyecto es la implementación de un sistema de posicionamiento de micrófonos completamente automatizado, controlable en remoto y con capacidad de almacenar diferentes posiciones, para su uso en estudios de grabación musical.

Podemos dividir el objetivo anterior en los siguientes objetivos secundarios:

- Implementación de la estructura que deberá soportar diferentes tipos de micrófonos y dotar de movimiento a los mismos, de manera que se puedan posicionar para realizar diferentes mediciones acústicas.
- Diseño y programación del hardware y del sistema mecánico. Será necesario introducir un microcontrolador que controle el movimiento de los micrófonos a través de varios motores
- Diseño y programación del software que controlará todo el sistema, realizando para ello una aplicación móvil

La estructura física deberá ser lo suficientemente resistente y al mismo tiempo ligera para que se adecúe a las necesidades de movimiento, y sea capaz de soportar de forma estable el peso de los distintos modelos de microfonía que nos podemos encontrar en los distintos entornos de trabajo. Existen diferentes modelos de micrófonos con una amplia variación en peso y tamaño, por ejemplo, un micrófono de condensador consiste en un pequeño diafragma muy habitual en mediciones acústicas, por otro lado existen modelos de gran diafragma con electrónica a válvulas. Por lo tanto, desde un primer momento la solución aportada a nuestro diseño de la estructura ha sido determinante para el funcionamiento y la capacidad de todo el proyecto. Esta estructura tendrá que contemplar todas las partes

móviles y albergar los distintos motores, usillos, rodamientos, electrónica, etc. necesarios.

Otra de las partes importantes que se ha contemplado desde un primer momento es la parte de electrónica y mecánica, la cual deberá ser capaz de impulsar cada uno de los ejes con la precisión y potencia requerida según el tipo de movimiento y peso de cada parte de la estructura. Para ello se ha utilizado la plataforma Arduino Uno junto con el módulo CNC Shield por su comodidad a la hora de realizar todo el conexionado con los distintos elementos sin tener que recurrir a utilizar placas de prototipado, lo cual sería muy tedioso a la hora de realizar todas las conexiones de forma correcta y estable.
Por otra parte, se han utilizado motores paso a paso de los cuales detallaremos su funcionamiento a lo largo de la memoria del proyecto. De estos, cabe destacar su alta precisión y facilidad de control basada en "pasos" o lo que es lo mismo subdivisiones de una vuelta completa de su eje. Esto nos permitirá determinar la posición de cada uno de los motores mediante un contador de pasos que implementaremos en el código, lo cual será fundamental para poder controlar los movimientos de la máquina, programar finales de carrera en el propio código y almacenar configuraciones en la tarjeta SD.

Finalmente se ha programado el hardware para controlar cada uno de los ejes de forma inalámbrica, y a su vez poder almacenar y leer en todo momento las posiciones o configuraciones de los distintos motores lo cual será una parte muy importante en el momento que se reinicie la maquina o si por cualquier motivo fallara la alimentación.

Otra parte importante dentro del apartado de programación será la implementación de la aplicación que utilizaremos para controlar todo el sistema de forma inalámbrica desde nuestro dispositivo móvil o Tablet. Como primer prototipo para este trabajo, se utilizará para crear la aplicación, la plataforma Apache Cordova. Esta primera versión de la aplicación cumplirá todas las funciones necesarias mediante una programación basada en CSS, HTML y Js enfocada a la creación de aplicaciones hibridas para dispositivos móviles.

CAPÍTULO 1. VIGILANCIA TECNOLÓGICA

En este apartado se ha buscado información sobre la existencia de productos similares en el mercado con los que se pueda comparar a nuestro proyecto. A nivel comercia se ha encontrado recientemente una empresa que se dedica a la construcción de este tipo de soportes electrónicos. La empresa en cuestión se llama "Dynamount" https://dynamount.com/products/studio/. Esta ya dispone de productos más avanzados con la incorporación de cámaras de video en tiempo real en su último producto, También dispone de distintos ejes de movimientos según la aplicación, no obstante requieren de un soporte adicional, como un pie de micrófono estándar para su colocación. Es una empresa bastante reciente, la cual empezó su producción en el 2016, y hasta la fecha no había encontrado ninguna aplicación similar, solo algunos proyectos caseros controlados por cable sin la posibilidad de almacenar presets de posiciones ni nada por el estilo. El precio del producto de su catálogo que más se aproxima al de nuestro proyecto ronda los 679$.

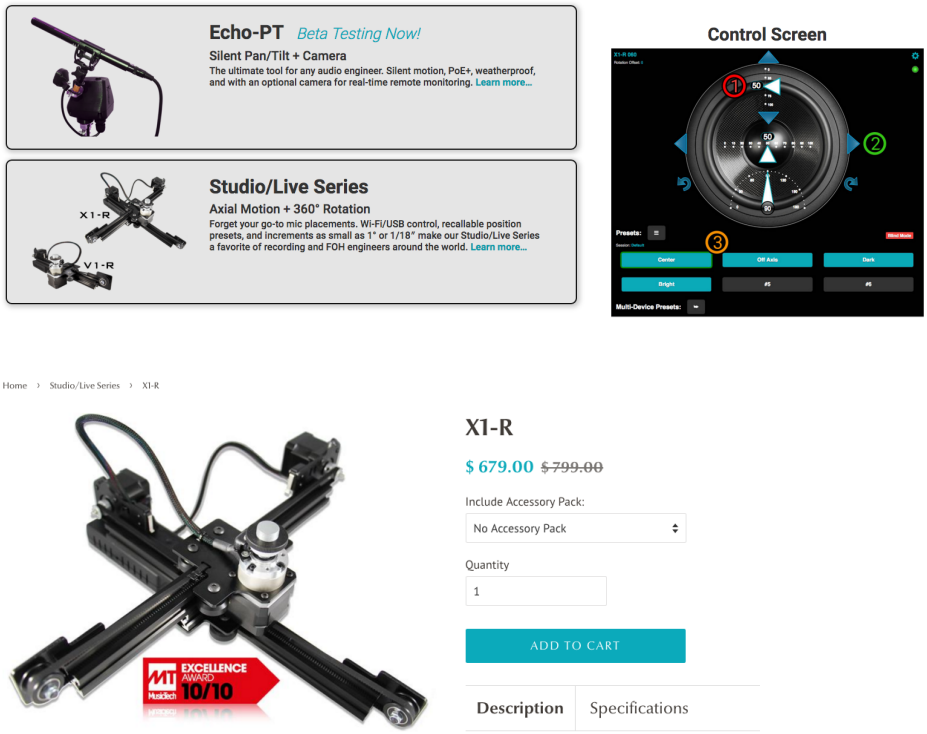


Figura 2 Productos similares que se comercializan

Como posible alternativa a dicha competencia se plantea enfocar dicho proyecto como un proyecto "do it yourself" (DIY) sobre Arduino, con un rediseño de la estructura mediante impresión 3D el cual facilite su confección, para poder ofrecer Kits de montaje para el usuario a fin de reducir costes en la confección y horas de trabajo.

CAPÍTULO 2. DISEÑO DE LA ESTRUCTURA MECÁNICA

Introducción

En este apartado hablaremos sobre la confección de la estructura mecánica de nuestro prototipo. Este primer paso, ha sido muy importante y decisivo a la hora de elaborar el proyecto ya que el correcto funcionamiento de todas las partes mecánicas, su robustez, un diseño que se adecúe a las necesidades y, por supuesto, los inconvenientes y dificultades a la hora de construir el primer prototipo con las herramientas que se disponía, han sido un factor clave para determinar la viabilidad de éste. El resto del trabajo referente a la programación, implementación electrónica, etc. dependerá en gran medida de la estructura mecánica desarrollada.

Por otra lado cabe destacar la importancia del coste económico a la hora de realizar por primera vez dicha estructura, ya que como muchos proyectos, el resultado final ha sido una evolución de distintas ideas, diseños y pruebas que aparentemente parecían cumplir sobre el papel, no obstante una vez se ha pasado a la construcción del prototipo han surgido distintos problemas con los que se ha tenido que lidiar a lo largo del proceso hasta llegar a un modelo final con el cual ya se podría trabajar.

Diseño del prototipo

Primeros diseños del prototipo

El diseño de la estructura física ha sido, como ya hemos dicho, la parte que más trabajo, tiempo y cambios ha requerido. Para dicho diseño se ha tenido en cuenta, desde un primer momento, la capacidad y funcionalidad que requería nuestra máquina para que se ajustara a las necesidades del usuario, y a la finalidad de su objetivo principal como soporte microfónico dentro del entorno de las grabaciones musicales o mediciones acústicas.

Concretamente, esta idea partió de la necesidad de "microfonear" amplificadores de instrumentos en un estudio de grabación, donde la caja de altavoces suele estar ubicada en una sala insonorizada y acondicionada acústicamente, que en la mayoría de casos está separada de la sala de control, donde está situado el ingeniero de sonido con todas las herramientas necesarias para poder registrar la señal proveniente de estos instrumentos.
El posicionamiento de los micrófonos es esencial a la hora de captar el tono deseado del instrumento, así como poder ajustar las fases entre los distintos micrófonos que se suelen utilizar al mismo tiempo para dicha captación. Es necesario que se pueda realizar este proceso sin tener que desplazarse de la sala

de control, donde se encuentra el equipo de monitorización, para mover los distintos micrófonos manualmente. En caso contrario se pierde la referencia que proporciona el sistema de monitorización, y el posicionamiento se realiza en base a la propia intuición.

A lo largo del proceso, surgieron distintas ideas de diseños, los cuales tuve que desarrollarlos en primer lugar con un software 3D como **SketchUp**, ya que de esta forma mientras representaba cada una de las partes sobre el propio dibujo ya se podía valorar la viabilidad de este, así como los distintos materiales que necesitaría. El primer diseño de estructura que se realizó era muy similar al de un soporte de micrófono convencional con una plataforma como base, no obstante se desechó debido a su construcción con hierro, la cual resultaba muy pesada para que un solo motor en el eje "Y" pudiera levantar todo el peso incluyendo el del propio micrófono. También era más difícil trabajar con hierro para realizar las distintas soldaduras, cortes y para incorporar todas las partes móviles como los distintos motores, rodamientos, etc.

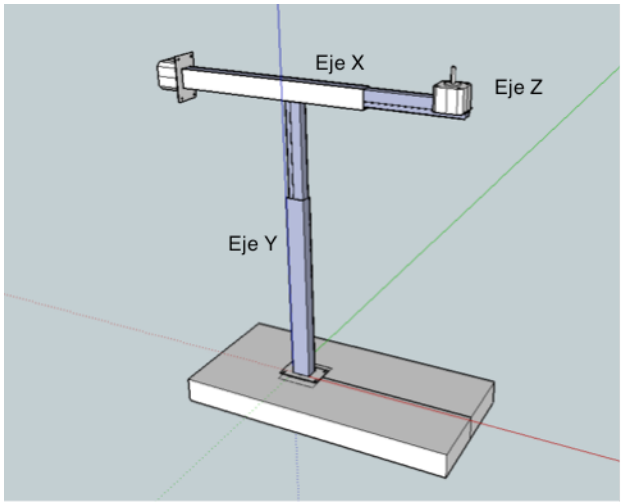


Figura 3 Primer diseño de la estructura mecánica

A continuación, se pensó en un diseño distinto donde su principal modificación fue el material empleado, que en este caso sería aluminio.
El aluminio es un material mucho más ligero que el hierro, con lo cual se consiguió reducir el peso de la estructura considerablemente. También es un material más fácil de cortar y agujerear al ser más blando que el hierro. No obstante aparecieron otro tipo de problemas, por una parte, es un material más caro que el hierro y más problemático a la hora de realizar las uniones de las distintas partes y piezas, ya que es mucho más difícil de soldar que el hierro, se necesitan soldadores más potentes, u otros métodos de soldadura más delicados.
Finalmente, la solución más viable, aunque mucho más laboriosa, ha sido utilizar remaches y tornillería para todas las uniones. Esto ha provocado que se tengan que hacer muchos taladros de distintos diámetros, con la problemática añadida de la precisión a la hora de realizar los distintos agujeros entre las piezas a unir. Además, las distintas tuercas y pliegues de los remaches sobresalen unos milímetros de la superficie de la estructura, por lo que en muchos casos se han tenido que tener en cuenta para que no rozaran ni interfirieran con ninguna parte

móvil. Por otro lado, con la finalidad de cuidar la estética del sistema, se han tenido que ocultar todas estas uniones.

Otro cambio importante en este nuevo diseño ha sido incorporar un sistema de doble columna para el eje "Y" con la finalidad de distribuir y equilibrar un poco más el peso del eje "X" teniendo en cuenta que éste último ha de sujetar el micrófono sobresaliendo y haciendo un contrapeso considerable a toda la estructura cuando esté en su posición más extendida.
Otra decisión importante fue la de intentar suprimir la base o plataforma que sostenía toda la estructura en el primer diseño, tal y como se muestra en la Figura 4. Se planteó esta solución a fin de reducir espacio pudiendo enroscar toda la estructura a un pie de micrófono convencional como si de un pinza de micrófono se tratara. Este nuevo diseño, desde nuestro punto de vista, contempla ideas muy interesantes como el hecho de ser mucho más compacto y versátil, ya que también se puede utilizar de forma vertical o horizontal según la aplicación.

No obstante para este trabajo se ha optado por un diseño menos arriesgado y más sólido, con la finalidad de evitar contratiempos intentando reforzar y en algunos casos sobredimensionar esas partes que nos podrían dar problemas en un futuro, como por ejemplo el hecho de añadir dos motores para el eje "Y" ya que es el que más peso ha de soportar, evitando de esta manera el sobrecalentamiento de los drivers de potencia de los motores o forzar el motor en cuestión cuando se utilice algún tipo de micrófono mucho más pesado.

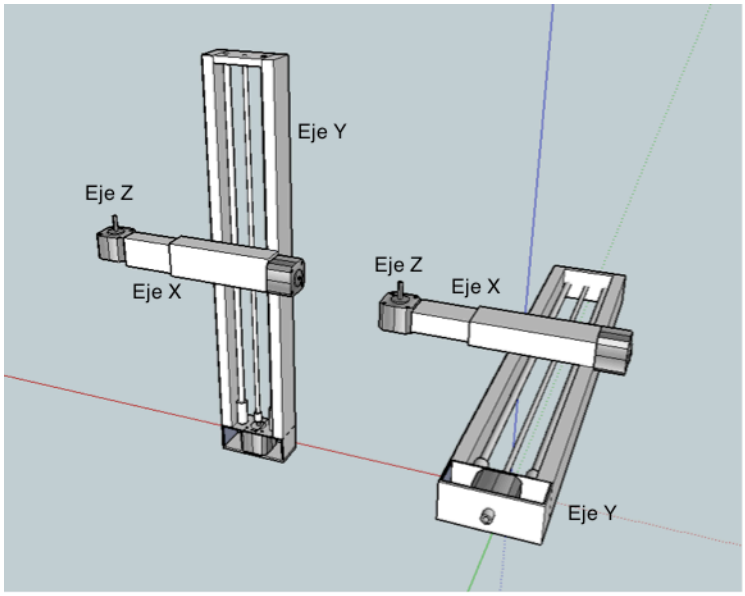


Figura 4 Estructura propuesta suprimiendo la base

Diseño del prototipo final

Finalmente, el resultado final del diseño de la estructura para este prototipo ha sido el resultado y la combinación de las distintas ideas plasmadas en los diseños anteriores intentado reforzar las partes que podrían ser más problemáticas para evitar posibles contratiempos a la hora de realizar este trabajo.

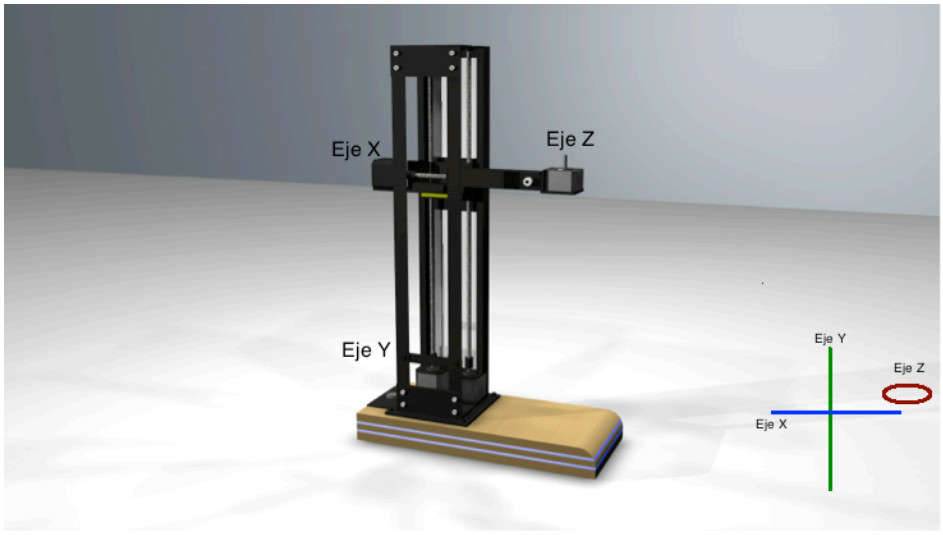


Figura 5 Estructura implementada

En este diseño el principal cambio que hemos realizado como ya hemos comentado anteriormente, es el de añadir un segundo motor para reforzar el eje "Y", de esta forma nos aseguramos disponer de suficiente fuerza para poder levantar sin ningún problema cualquier tipo de micrófono que pueda resultar más pesado.

Otro aporte importante relacionado con lo anterior, es el hecho de utilizar dos husillos en este eje Y, con lo cual nos proporciona una mayor distribución de la carga al intervenir sobre el eje X en dos puntos de sujeción además de los cuatro rodamientos que incorpora el carril deslizante encargado de sujetar el eje "X".

Con este sistema ya tendríamos prácticamente garantizado el correcto funcionamiento para este eje "Y", siendo este el más delicado en este tipo de estructura.

Otros aspectos menos importantes con los que hemos tenido que lidiar han sido:

1. el ruido de carácter estructural que se producía al girar los motores los cuales hacían vibrar la estructura metálica resultando ser un poco molesto sobre todo a nivel "estético", ya que solo se produce al mover el micrófono de sitio, no obstante, una vez está en el sitio deseado, los motores dejan de estar en funcionamiento con lo cual no se produce ningún ruido ni interferencia en la señal de audio captada por el micrófono. A fin de reducir este problema, se ha añadido en aquellas partes más conflictivas y vulnerables, unos separadores de plástico y espuma de alta densidad a modo de "*silentblock*", el cual absorbe gran parte de las vibraciones de los husillos al girar sobre su tuerca deslizante.

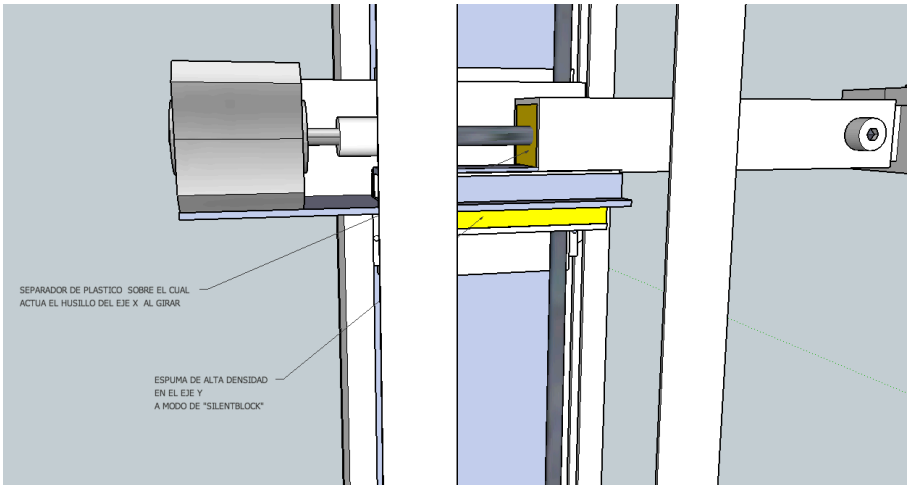


Figura 6 Aislamiento de ruido estructural

Esta estructura es un primer prototipo que podría mejorarse significativamente. Una buena alternativa para la confección de futuras estructuras, sería preparar un nuevo diseño enfocado a impresoras 3D, con esta alternativa sería mucho más fácil realizar todo el montaje el cual ha sido la parte más tediosa y delicada en este proyecto. Otra ventaja de esto, sería el conseguir reducir mucho más el peso al utilizar como material principal el plástico.

CAPÍTULO 3. DISEÑO DEL HARDWARE

Introducción

En este apartado comentaremos los distintos componentes que hemos utilizado para el desarrollo de la parte motriz y todos los dispositivos electrónicos que intervienen en nuestro prototipo para el control y potencia.

Los componentes utilizados han sido:

- Microcontrolador Arduino Uno
- Placa de conexiones "CNC Shield"
- 4 x Drivers 8825
- 4 x Soportes Motor
- 4 x Motores paso a paso
- Modulo Bluetooth DSD TECH HC-05
- Modulo Tarjeta SD
- 3 x Husillos
- 2 x Cojinetes
- Fuente alimentación 12 V

Descripción y Características técnicas

Características técnicas del ARDUINO UNO

Arduino Uno es el microncontrolador que hemos utilizado para implementar el control de nuestro prototipo.
Se trata de una placa con un microcontrolador de la marca Atmel y con toda la circuitería de soporte, que incluye, reguladores de tensión, un puerto USB (En los últimos modelos, aunque el original utilizaba un puerto serie) conectado a un módulo adaptador USB-Serie que permite programar el microcontrolador desde cualquier PC o MAC de manera cómoda y también hacer pruebas de comunicación con el propio chip.
Este arduino dispone de 14 pines que pueden configurarse como entrada o salida y a los que puede conectarse cualquier dispositivo que sea capaz de transmitir o recibir señales digitales de 0 y 5 V. También dispone de entradas y salidas analógicas. Mediante las entradas analógicas podemos obtener datos de sensores en forma de variaciones continuas de un voltaje. Las salidas analógicas suelen utilizarse para enviar señales de control en forma de señales PWM.

Entradas y salidas del Arduino UNO

Cada uno de los 14 pines digitales se puede usar como entrada o como salida. Funcionan a 5V, cada pin puede suministrar hasta 40 mA.
La intensidad máxima de entrada también es de 40 mA.
Cada uno de los pines digitales dispone de una resistencia de pull-up interna de entre 20KΩ y 50 KΩ que está desconectada, salvo que nosotros indiquemos lo contrario.
Arduino también dispone de 6 pines de entrada analógicos que trasladan las señales a un conversor analógico/digital de 10 bits.

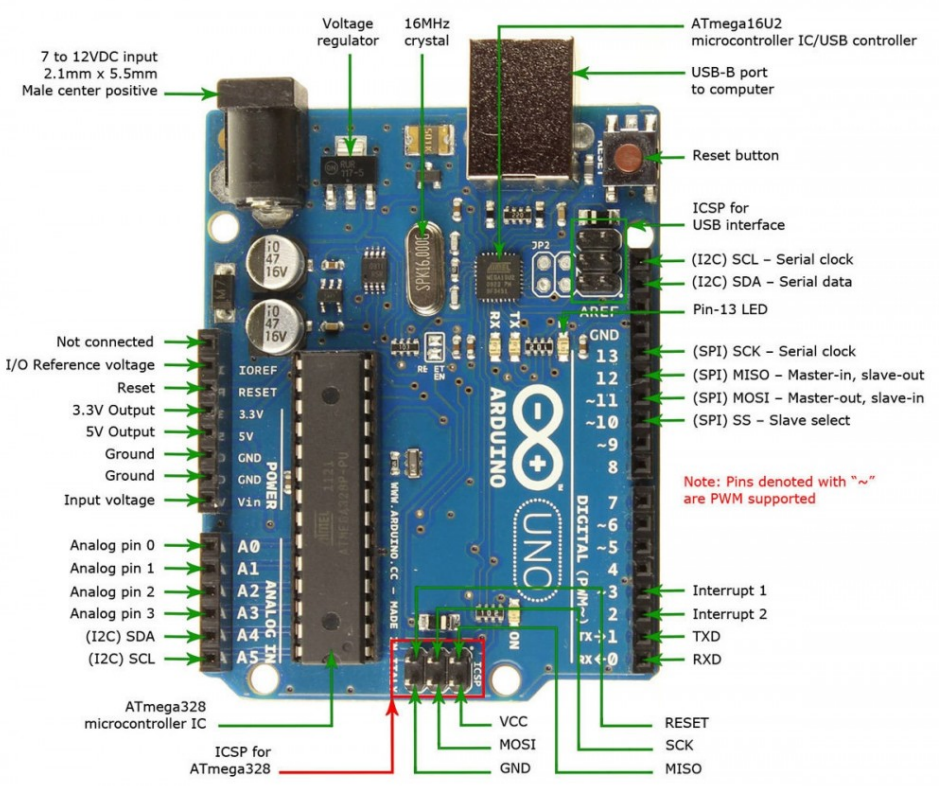


Figura 7 Placa Arduino UNO

Pines especiales de entrada y salida

Existen toda una serie de pines que se utilizan para comunicarse con diferentes periféricos. Caben destacar los siguientes:

- Pines RX y TX: Se usan para transmisiones serie de señales TTL, utilizando el interface de transmisión UART
- Interrupciones externas: Los pines 2 y 3 están configurados para generar una interrupción en el microcontrolador. Las interrupciones pueden dispararse cuando se encuentra un valor bajo en estas entradas y con flancos de subida o bajada de la entrada.
- PWM: Arduino dispone de 6 salidas destinadas a la generación de señales PWM de hasta 8 bits.

- SPI: Los pines 10, 11, 12 y 13 pueden utilizarse para llevar a cabo comunicaciones SPI, que permiten trasladar información full dúplex en un entorno Maestro/Esclavo.
- I 2C: Permite establecer comunicaciones a través de un bus I 2C. El bus I 2C es un producto de Phillips para interconexión de sistemas embebidos. Actualmente se puede encontrar una gran diversidad de dispositivos que utilizan esta interfaz, desde pantallas LCD, memorias EEPROM, sensores...

Alimentación de Arduino

Puede alimentarse directamente a través del propio cable USB o mediante una fuente de alimentación externa, como puede ser un pequeño transformador o, por ejemplo una pila de 9V.

Los límites están entre los 6 y los 12 V. Como única restricción hay que saber que, si la placa se alimenta con menos de 7V, la salida del regulador de tensión a 5V puede dar menos que este voltaje y si sobrepasamos los 12V, probablemente dañaremos la placa.

La alimentación puede conectarse mediante un conector de 2,1mm con el positivo en el centro o directamente a los pines Vin y GND marcados sobre la placa.

Hay que tener en cuenta que podemos medir el voltaje presente en el jack directamente desde el pin Vin. En el caso de que el Arduino esté siendo alimentado mediante el cable USB, ese voltaje no podrá monitorizarse desde aquí.

Resumen de características Técnicas Arduino

Microcontrolador	Atmega328
Voltaje de Operación	5V
Voltaje de entrada (Recomendado)	7 - 12V
Voltaje de entrada (Límite)	6 - 20V
Pines para entrada-salida digital.	14(6 pueden usarse como salida de PWM)
Pines para entrada analógica.	6
Corriente continua por pin IO	40 mA
Corriente continua en el pin 3.3V	50 mA
Memoria Flash	32 KB (0,5 KB ocupados por el bootloader)
SRAM	2 KB
EEPROM	1 KB
Frecuencia de reloj	16 MHZ

Características técnicas de la CNC SHIELD

La CNC Shield es básicamente una placa modular diseñada principalmente para facilitar el montaje de una CNC (control numérico por computadora) muy utilizado actualmente para realizar proyectos de impresoras 3D, grabadoras y cortadoras laser, fresadoras, etc… sobre la plataforma Arduino UNO.

La principal característica de esta placa modular es la facilidad de conexionado a la hora de trabajar con motores paso a paso, ya que esta incorpora todos los zócalos, puertos y pines necesarios para la conexión directa sobre la placa Arduino de todos los elementos que necesitaremos para la construcción de una maquina CNC sin la necesidad de recurrir a placas protoboard, o cableado externo, el cual es muy tedioso y vulnerable a sufrir problemas de malas conexiones.
Esta placa tiene la capacidad de controlar tres ejes (X, Y, Z), no obstante soporta 4 motores paso a paso con sus respectivos drivers de potencia ya que como opción nos da la posibilidad de replicar uno de estos tres ejes a un cuarto llamado "A" mediante una sencilla configuración de unos "jumpers". La finalidad de esta opción es la de poder añadir otro motor para dotar de más fuerza a un determinado eje el cual necesite mover un mayor peso de la estructura mecánica. Esta opción, nos ha sido de gran ayuda para nuestro proyecto , ya que como hemos comentado en el apartado referente al diseño de la estructura, nuestro eje Y era el más problemático a la hora de levantar todo el peso sin sobrecargar el motor o el driver de potencia, y por este motivo se ha utilizado un segundo motor configurado como réplica del eje Y.
Esta placa contempla la posibilidad de añadir pulsadores como finales de carrera para los distintos ejes, no obstante en nuestro proyecto los finales de carrera se han implementado en el propio código calculando y limitando el número de pasos para cada eje, de esta forma hemos podido utilizar estos pines para otras funciones específicas para nuestra aplicación como puede ser el conexionado del módulo bluetooth, o el módulo de la tarjeta SD, los cuales no están pensados para el propósito principal de esta placa a la hora de construir una CNC estándar.

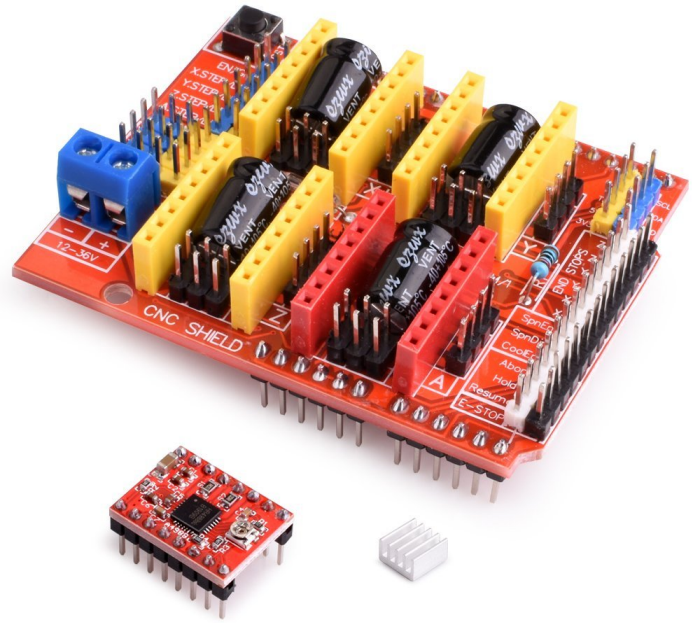


Figura 8 Placa CNC Shield

Diagrama de Pines de entrada y salida de la CNC Shield:

La placa CNC *Shield* tiene 9 partes principales, que se describen a continuación:

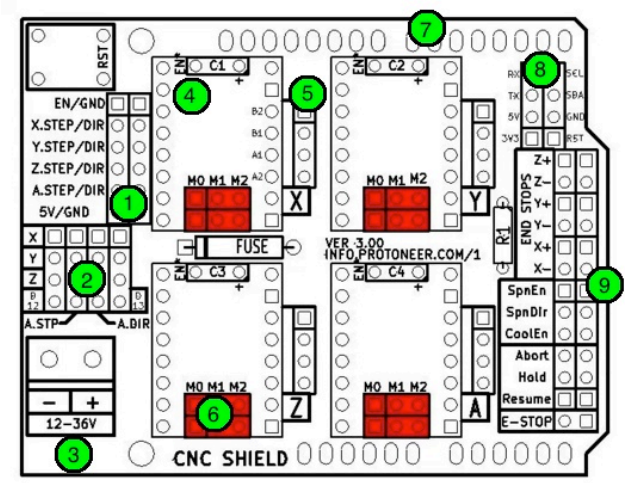


Figura 9 CNC Shield Diagrama de conexiones principales

1. Pines STEP y Direction, de cada motor para conexión externa. Nos permite poder controlar los motores desde el exterior con otro microcontrolador.
2. Pines X, Y, Z para clonarse con la opción duplicado del controlador A. Permite seleccionar mediante dos *jumpers*, que pines controlaran los STEP y DIR del eje a el cual se puede realizar con los ejes X, Y o Z y también con el mismo A(pines D12 yD13). Sin estos *jumpers* el eje A aunque este programado no hará nada.
3. Entrada de voltaje DC, cuyo valor debe ser de 12 a 36V.
4. Zócalos dobles disponibles para insertar el controlador de motor paso a paso (A4988 u otro compatible).
5. Salidas para conectar el motor paso a paso.
6. Pines para ajustar los micro-paso del motor paso a paso (de acuerdo al controlador). Junto a los puntos 4 y 5, los 4 motores independientes.
7. Pines que se comunican con el Arduino.
8. Pines libres que corresponden a A4, A5, D0 y D1 mas fuentes 5 y 3,3V.
9. Pines de salida y entrada para el control de: finales de carrera, control de husillo, control de refrigerante y control de “parada de emergencia”.

Drivers de control para motores Paso a Paso

Los drivers actúan como una válvula que controla la corriente de los motores. Cuando el Arduino manda la señal, el driver permite pasar la corriente desde nuestra fuente de alimentación a la bobina del motor. De esta forma, el driver separa la corriente que alimenta al Arduino (*normalmente de 5V*), de la corriente que circula por las bobinas de los motores (*normalmente de 12V)*.

Por este motivo, podemos alimentar un Arduino con un cable USB de 5V, pero necesitamos además una fuente de alimentación de 12V y mucha potencia para los motores.

Tipos de driver y cómo funcionan

A continuación se comentarán los diferentes tipos de drivers podemos encontrar. Lo primero que hay que saber es que no es lo mismo un motor unipolar que un motor bipolar. Su funcionamiento es distinto y, por lo tanto, el circuito del driver también. En este proyecto se han utilizado motores bipolares, no obstante, haremos una pequeña explicación de ambos modelos para entender las diferencias y el porqué de utilizar unos u otros según la aplicación.

Definición de un motor paso a paso Unipolar

Un motor unipolar (también llamado motor homopolar) es un motor de corriente continua (CC) típicamente con anillos colectores en cada extremo de un rotor cilíndrico e imanes de campo o un devanado de campo de CC que genera un campo magnético en el estator. El rotor típicamente no tiene un devanado, sino simplemente conexiones rectas en dirección axial entre los anillos colectores (por ejemplo, un tubo de cobre que encapsula el rotor o las barras incrustadas en el rotor).
Se denominan así debido a que la corriente que circula por sus bobinas lo hace en un mismo sentido, a diferencia de los bipolares. Se componen de 6 cables externos, dos para cada bobina, y otro para cada par de éstas, aunque también se pueden ver con 5 cables, compartiendo el de alimentación para los 2 pares de bobinas.

**Los motores paso a paso unipolares se componen de 4 bobinas.*

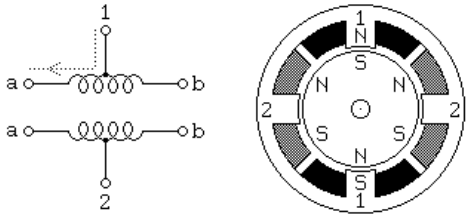


Figura 10 Driver y Motor Unipolar

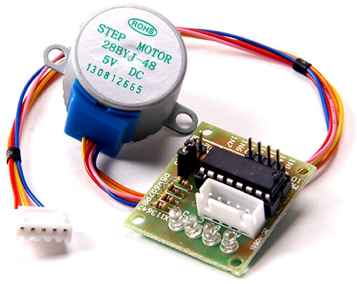


Figura 11 Esquema Motor unipolar

Driver para un motor paso a paso unipolar

Los motores unipolares son los más sencillos de controlar. Como hemos comentado en la propia definición de un motor unipolar, la corriente circula por las bobinas siempre en la misma dirección, y el trabajo del driver se reduce a conocer qué bobina tenemos que activar en cada pulso. El driver más conocido es el ULN2003A. Es un driver muy económico y muy sencillo de usar.
Por dentro, este driver no es más que unos transistores Darlington, que es un tipo de transistor muy potente. Cuando el Arduino envía un pulso, los transistores dejan pasar la corriente a las bobinas, y el motor avanza un paso.

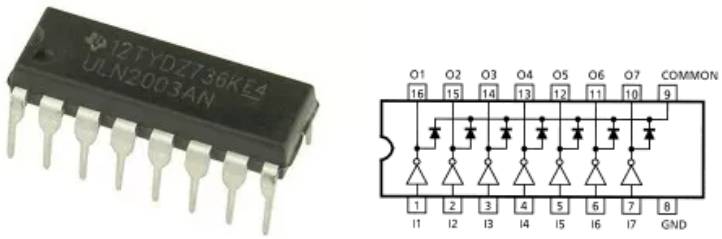


Figura 12 Driver Unipolar ULN2003

Además, este circuito incluye unos diodos de protección, para que la corriente almacenada en la bobina no venga de vuelta, y dañe nuestro circuito.
En realidad no es un driver de motores paso a paso. Se puede utilizar para cualquier proyecto que necesite encender y apagar algo, como para activar relés, encender y apagar LEDs, etc.

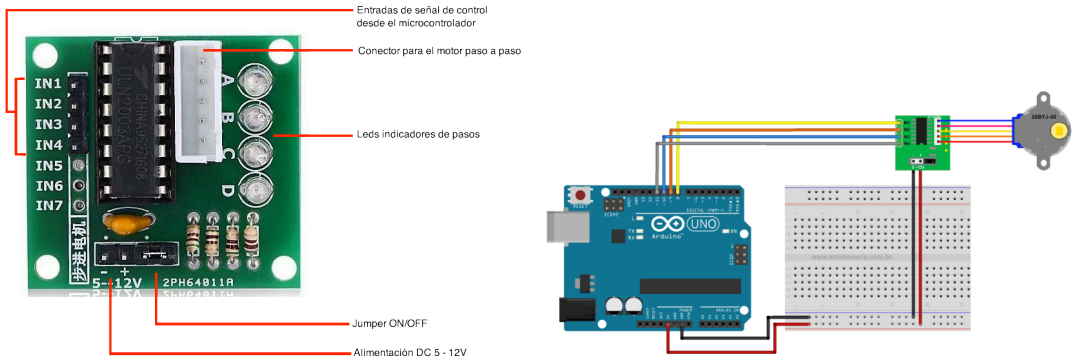


Figura 13 PCB Driver para motor Unipolar y conexiones en Arduino

Definición de un motor paso a paso Bipolar

Los motores bipolares se construyen en forma similar que los unipolares pero los dos devanados se conectan en forma más sencilla, puesto que no posee terminales centrales. El motor en sí es más sencillo, pero el circuito de control se torna más complejo puesto que se necesita revertir la polaridad de cada bobina.

Para hacer girar un motor paso a paso bipolar, se aplican impulsos en secuencia a los devanados, la secuencia de estos impulsos, se aplican externamente con un controlador electrónico que en nuestro caso será a través de Arduino. Los motores bipolares, se pueden hacer avanzar a frecuencias de relativamente altas, lo que les permite girar muy velozmente. Con un controlador apropiado, se les puede hacer arrancar y detenerse en cualquier instante y en una posición determinada lo cual es una gran ventaja y la opción más apropiada para proyectos donde se requiere una precisión y control de giro.

**Los motores paso a paso Bipolares se componen de 2 bobinas.*

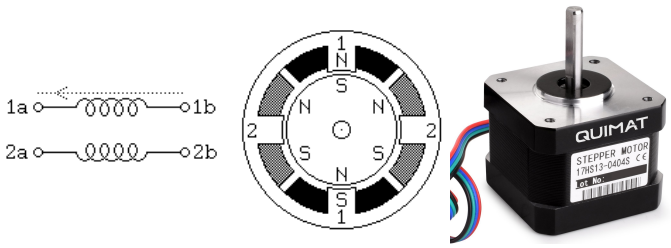


Figura 14 Esquema Motro Bipolar

Estos motores, tienen varios bobinados que, para producir el avance de un paso, deben ser alimentados en una secuencia adecuada. Al invertir el orden de esta secuencia, se logra que el motor gire en sentido opuesto. El torque de detención, hace que un motor paso a paso (bipolares y unipolares) se mantenga firmemente en su posición, estando alimentado aun cuando no esté girando.

Los motores paso a paso bipolares cuentan con cuatro cables los cuales se corresponden a los cuatro terminales, dos de cada devanado, aunque internamente pueden contener varios pares de bobinas, como se aprecia en la imagen anterior. Los motores paso a paso, difieren de los motores de CC, en la relación entre velocidad y torque o "par motor". Su mayor capacidad de torque se produce a baja velocidad.

Los motores bipolares requieren circuitos de control y de potencia más complejos que los unipolares. Estos circuitos de control, se suelen implementar con un circuito integrado que, soluciona dicha complejidad con un solo componente. Cuando se requieren mayores potencias, se deben agregar algunos componentes, como transistores y diodos para las contracorrientes, aunque esto no es del todo necesario en los pequeños motores.

La configuración de los motores bipolares requiere que las bobinas reciban corriente en ambos sentidos y no solamente un encendido-apagado como en los unipolares. Esto hace necesario el uso de un Puente H con una secuencia sobre cada uno de los devanados. En la siguiente figura se muestra una secuencia simple donde se puede observar cómo se va alimentando secuencialmente cada una de la bobinas. De esta forma, según el orden y la frecuencia en la que se vayan activando cada una de las bobinas, se determinará el sentido de giro del motor, la velocidad y mediante otro tipo de secuencia en la que se combine la activación de más de una bobina, se podrá a conseguir mayor precisión de giro.

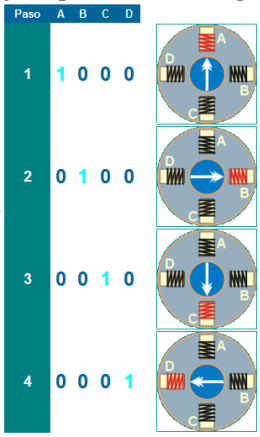


Figura 15 Ejemplo secuencia de pasos

Al avanzar un paso de la secuencia al paso siguiente, hace mover el estator del motor un paso. Si vamos hacia adelante en la secuencia, el motor se mueve hacia adelante un paso y si lo hacemos al revés, el motor se mueve un paso atrás.

Driver para un motor paso a paso bipolar

La inmensa mayoría de drivers se conectan a nuestro Arduino usando 3 cables.
El Arduino envía el pulso (**STEP**). Cada pulso le dice al motor que tiene que avanzar un paso.
Otro pin nos dice la dirección de avance (**DIR**). Si queremos ir en el sentido de las agujas del reloj, o al contrario.
El otro cable es el **GND** común, necesario para conectar las señales con el arduino.
De esta forma, cada vez que el driver recibe un pulso en el pin **STEP**, el circuito comprueba el voltaje del pin **DIR**, y alimenta las bobinas del motor en el orden adecuado.

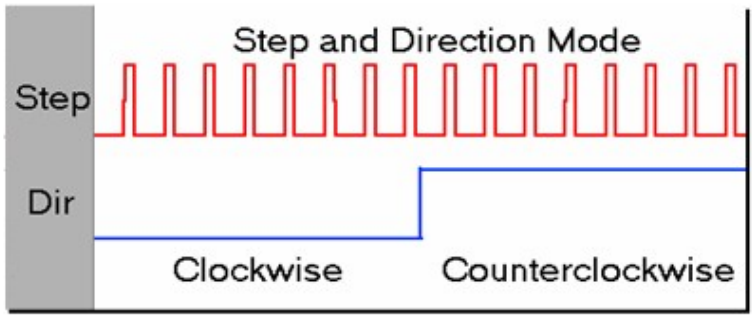


Figura 16 Ejemplo secuencia STEP Y DIR

Para nuestro proyecto utilizando la CNC Shield, existen 2 modelos más comunes de drivers que podríamos utilizar diseñados para acoplar a los zócalos de nuestra placa. Estos drivers son el driver A4988 y el DRV8825.

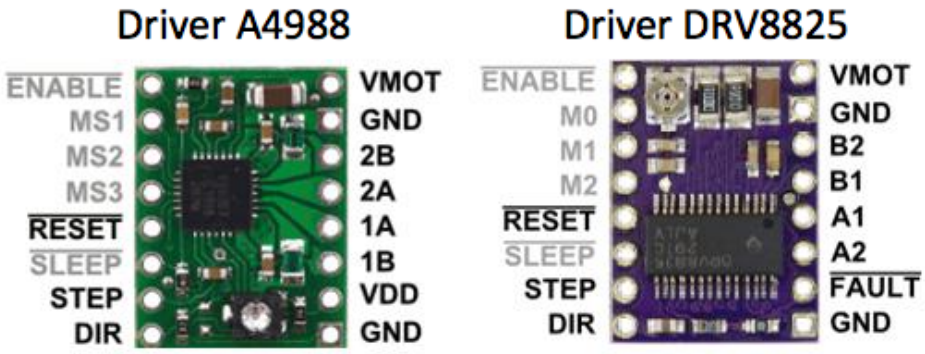


Figura 17 Driver A4988 y DRV8825

Descripción del driver A4988

Este driver tiene la capacidad de controlar un motor paso a paso bipolar, desde 1 paso por pulso hasta 1/16 pasos por pulso, esto se configura colocando jumper en los pines que aparecen en la siguiente figura. La configuración de estos jumpers en los pines nos determinara la resolución de pasos en la que los motores trabajaran. Los pines MS1, MS2 y MS3, por defecto están a GND o 0V por lo que para llevarlo a 5V, se deben colocar jumpers a VDD, las combinaciones se muestran en la tabla.

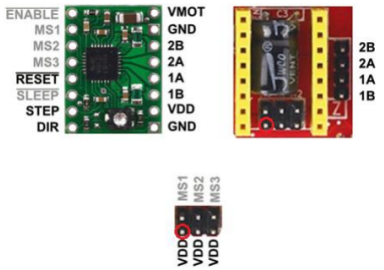


Figura 18 Pines driver A4988

MS1	MS2	MS3	Microsteap Resolution	Excitación Mode
L	L	L	Full Steap	2 Phase
H	L	L	Half Steap 1/2	1-2 Phase
L	H	L	Quarter Steap 1/4	W1-2 Phase
H	H	L	Eighter Steap 1/8	2W1-2 Phase
H	H	H	Sixttenth Steap 1/16	4W1-2 Phase

H= High o alto, significa jumper colocado
L= Low o bajo, significa jumper no colocado

Descripción del driver DRV8825

A diferencia del driver anterior este, tienen la capacidad de controlar un motor PaP bipolar, desde 1 paso por pulso hasta 1/32 pasos por pulso, es decir podemos obtener mayor precisión de giro. Este es el driver que hemos utilizado para nuestro proyecto, aunque cualquiera de los dos nos servía ya que no necesitamos tanta resolución, por eso aunque se comente la opción de configurar dichos jumpers, en nuestro proyecto, los drivers están configurados en modo "FullSteap", es decir, sin hacer uso de los jumpers de la CNC Shield.

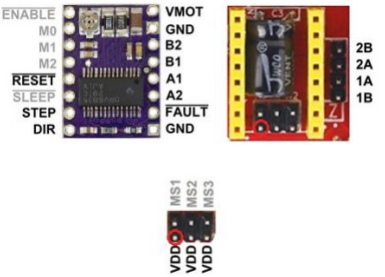


Figura 19 Pines driver DRV8825

MS1	MS2	MS3	Microsteap Resolution	Excitación Mode
L	L	L	Full Steap	2 Phase
L	L	H	Steap 1/2	1-2 Phase
L	H	L	Steap 1/4	W1-2 Phase
L	H	H	Steap 1/8	2W1-2 Phase
H	L	L	Steap 1/16	4W1-2 Phase
H	L	H	Steap 1/32	8W1-2 Phase
H	H	L	Steap 1/32	8W1-2 Phase
H	H	H	Steap 1/32	8W1-2 Phase

H= High o alto, significa jumper colocado
L= Low o bajo, significa jumper no colocado

Características motores paso a paso utilizados

Para este proyecto hemos utilizado 4 motores PaP bipolares, de los cuales 3 de ellos corresponden exactamente al mismo modelo de la marca "*Kuman*" y el cuarto motor utilizado de la marca "*Ueetek*" para el eje "Z", es decir , el eje que se encargaría de rotar el micrófono, es simplemente un modelo similar, pero con un tamaño más reducido, más económico y un poco menos potente ya que no

necesitamos cargar con ningún peso de la propia estructura. A continuación, se mostraran las características técnicas de dichos motores.

Especificaciones motor Kuman Nema 17

KUMAN Nema 17 Motor PaP	
Ángulo del paso	1.8 grados
Par de retención	0.59Nm(84 oz.in) 6kg/cm
Corriente nominal/fase	2.0 A
Voltaje	12V
Longitud del cuerpo	46 mm
Tamaño del marco	41 x 41 mm
Longitud del eje	22 mm
Peso	380 g

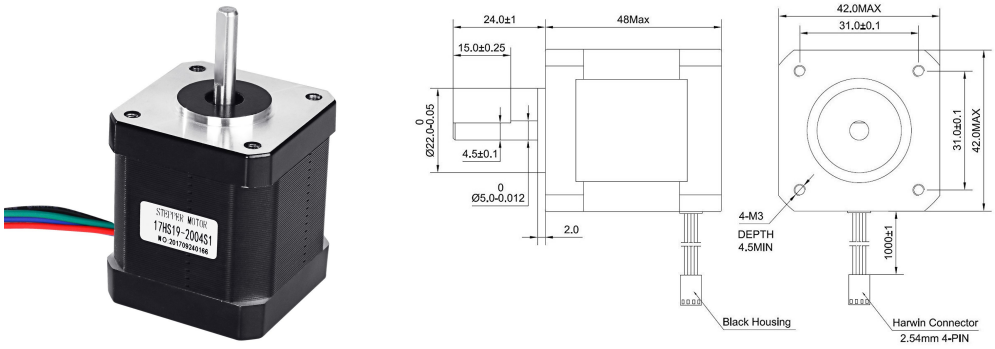


Figura 20 Motor Kuman Nema 17

Especificaciones motor Ueetek Nema 17

UEETEK Nema 17 Motor PaP	
Ángulo del paso	1.8 grados
Par de retención	0.26Nm(36.8 oz.in) **2.65**kg/cm
Corriente nominal/fase	0.4 A
Voltaje	12V
Longitud del cuerpo	34 mm
Tamaño del marco	42 x 42 mm
Longitud del eje	20 mm
Peso	230 g

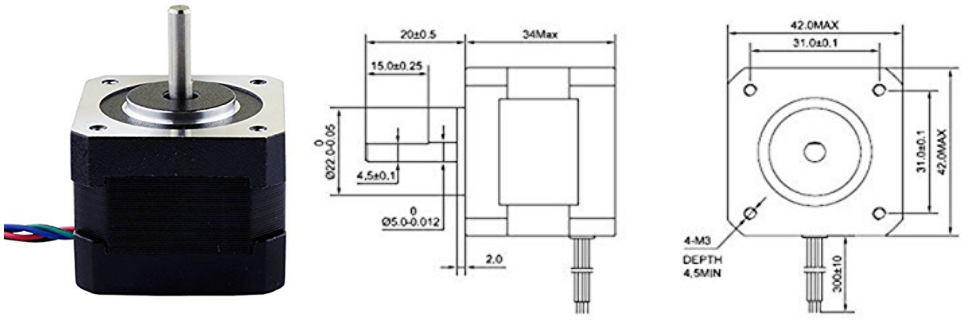


Figura 21 Motor UEETEK NEMA 17

Características técnicas modulo Bluetooth DSD TECH HC-05

Para este proyecto hemos utilizado para la comunicación inalámbrica entre la máquina y nuestra aplicación, el modulo bluetooth DSD Tech HC-05. El módulo bluetooth HC-05 es el que ofrece una mejor relación de precio y características, ya que es un módulo Maestro-Esclavo, esto quiere decir que además de recibir conexiones desde una PC o tablet, también es capaz de generar conexiones hacia otros dispositivos bluetooth. Esto nos permite por ejemplo, conectar dos módulos de bluetooth y formar una conexión punto a punto para transmitir datos entre dos microcontroladores o dispositivos. El módulo HC-05 El HC-05 tiene un modo de comandos AT que debe activarse mediante un estado alto en el PIN marcado como "Key" mientras se enciende (o se resetea) el módulo. Una vez que estamos en el modo de comandos AT, podemos configurar el módulo bluetooth y cambiar parámetros como el nombre del dispositivo, password, modo maestro/esclavo, etc. A continuación se muestra el esquema de conexiones y pines del módulo con nuestra placa Arduino Uno, y el "Sketch" que hemos utilizado para configurar nuestro módulo.

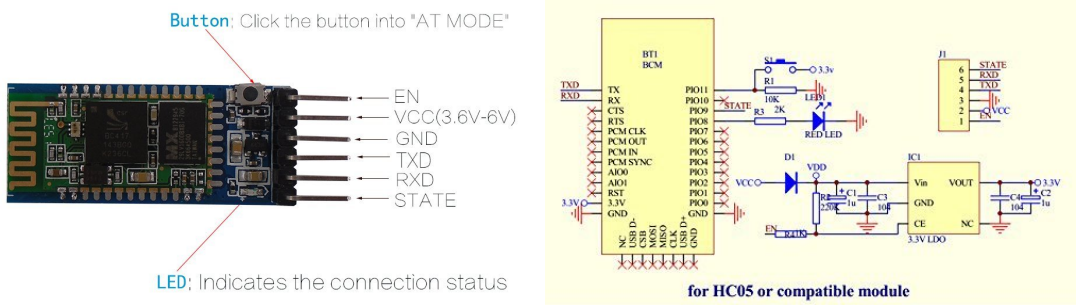


Figura 22 Modulo Bluetooth HC 05

Esquema conexiones modulo Bluetooth

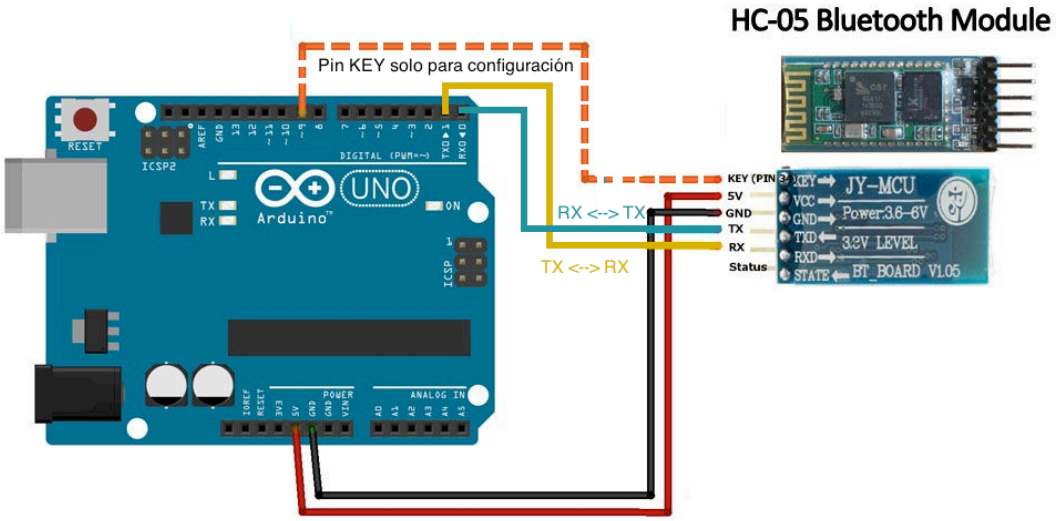


Figura 23 Conexiones modulo Bluetooth

Características Hardware del módulo buletooth

- Compatible con Arduino
- Sensibilidad Típica: -80dBm.
- Hasta +4 dBm de potencia de transmisión RF.
- Fully Qualified Bluetooth V2.0 +modulción EDR 3Mbps.
- Funcionamiento de bajo consumo.
- PIO control.
- Interfaz UART con velocidad de modulación en baudios programable.
- Antena PCB Integrada.

Características del Software del módulo buletooth

- Velocidad en baudios (Modo comandos AT): 38400, Bits de datos: 8, Bit de parada: 1, Paridad: Sin paridad.
- Tasa de velocidad de modulación en baudios soportadas: 9600, 19200, 38400, 57600, 115200, 230400, 460800.
- Auto-conexión del dispositivo con la última configuración por defecto.
- Permiso conectar el dispositivo emparejado de forma predeterminada.
- Por defecto PINCODE: "1234".
- Reconexión automática en 30 min cuando se desconecta como consecuencia de pérdida de conexión por salirse del rango de alcance.

"Sketch" para la configuración del modulo Bluetooth HC-05

```
const int LED = 13;
const int BTPWR = 12;

char nombreBT[7] = "MicBot"; //nombre que hemos puesto para nuestra maquina
char velocidad ='4';//9600 velocidad en baudios
char pin [5]= "0000";// nuestro password

void setup(){
       pinMode(LED, OUTPUT);
       pinMode(BTPWR, OUTPUT);
       digitalWrite(LED, LOW);
       digitalWrite(BTPWR, HIGH);
       Serial.begin(9600);
       Serial.print("AT");
       delay(1000);
       Serial.print("AT+NAME");
       Serial.print(nombreBT);
       delay(1000);
       Serial.print("AT+BAUD");
       Serial.print(velocidad);
       delay(1000);
       Serial.print("AT+PIN");
       Serial.print(pin);
       delay(1000);
       digitalWrite(LED, HIGH);
}
void loop(){
}
```

Características técnicas del módulo para tarjetas SD

Las memorias SD son las más usadas por dispositivos portátiles, por su gran capacidad y su reducido tamaño, debido a su gran demanda son fáciles de conseguir en diferentes capacidades y precios. Estas características nos dan una buena alternativa de almacenamiento para usarlo en Arduino, sobre todo cuando necesitamos guarda gran cantidad de información. En nuestro caso, para nuestro proyecto, la finalidad de introducir una memoria SD externa, no es por necesidad de almacenar gran cantidad de información, sino por la necesidad de almacenar los valores de ciertas variables correspondientes a las posiciones de los motores en una memoria no volátil o con una vida útil como podría ser la memoria *EPROM* del propio microcontrolador.
Estas memorias vienen en tres tamaños, SD estándar, Mini SD y Micro SD, siendo este último el tamaño más común, funcionalmente son iguales, pudiéndose usar adaptadores para utilizarlos en sockets de diferente tamaño.
Con respecto al formato podemos encontrar 4 tipos, las tarjetas SD o SDSC (Standard Capacity), SDHC (High Capacity), SDXC (Extended Capacity) y las SDIO (Input/Output), permitiéndonos Arduino trabajar con los dos primeros tipos.
La comunicación de la memoria es por SPI pero trabajan con 3.3V, para utilizarlo con Arduino necesitamos módulos externos que aparte de tener el socket traen los componentes necesarios para adaptar los voltajes a TTL y poder conectarlo de forma fácil a nuestro Arduino.

Esquema conexiones modulo tarjeta SD

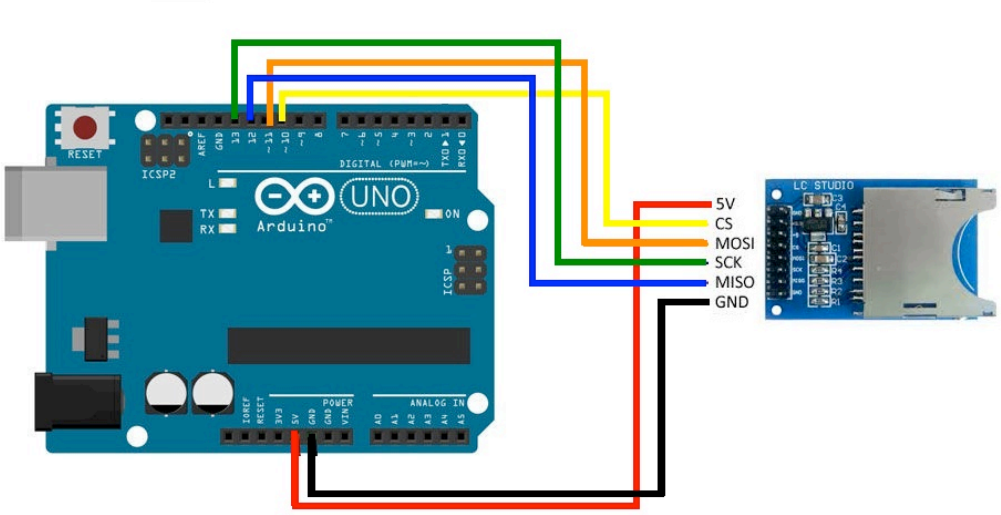


Figura 24 Conexiones modulo SD Card

Características Hardware modulo SD Card

En nuestro caso hemos utilizado los pines 10, 11, 12 ,13 para la comunicación con nuestra memoria SD. Cabe destacar que hemos tenido que cambiar la asignación del pin correspondiente al *"CS"* (Chip select), ya que por defecto al incorporar la librería "SD.h" genérica para poder trabajar con la SD , dicho pin está asignado al pin 4 de nuestro Arduino, no obstante este pin, en nuestro caso ya se está utilizando para controlar el *"Step"* de uno de los motores.

Los pines más destacables en un módulo SD son los siguientes:

CS	Slave select o chip select. Para habilitar a cada dispositivo esclavo. En general es activo bajo.
SCK	Serial clock. Para los pulsos generados por el maestro.
MISO	Master input salve output. Por este pin se envían los datos del esclavo al maestro.
MOSI	Master output slave input. Por este pin se envían los datos del maestro al esclavo.

Características Software modulo SD Card

En este apartado se explicará la forma de comunicarnos con nuestra memoria SD, en nuestro proyecto de Arduino mediante el uso de una librería la cual contempla una series de métodos o funciones con la que se hace muy intuitivo trabajar.
La librería ya viene junto con el *IDE* de arduino por lo que no necesitamos instalar ni descargar nada.
Para poder usar la librería en nuestro Sketch es necesario incluir a la librería SD al inicio del código:

```
#include <SD.h>
```

Funciones de la librería SD:

A continuación explicamos las funciones principales de la librería SD, el cual es un resumen de la referencia proporcionada en la página oficial de Arduino: https://www.arduino.cc/en/Reference/SD

SD.begin(cspin)
Inicializa la biblioteca SD y la tarjeta, como parámetro se le indica el pin CS al que está conectado el modulo, si no se especifica *cspin* , se usa el valor por defecto del CS por hardware. Los demás pines deben estar conectados al SPI por hardware del Arduino.

SD.exists(filename)
Comprueba si existe el archivo especificado, *filename* es el nombre del archivo y/o directorio en la tarjeta SD si este existe la función nos retorna un true, de lo contrario retorna false.

SD.mkdir(directory)
Crea el directorio especificado, si los subdirectorios no existen, también se crearán. Por ejemplo: SD.mkdir("Arduino/proyecto1/archivos), crea la carpeta "archivos" y si las carpetas Arduino y proyecto1 no existen, entonces también se crearan. La función retorna true si la creación del directorio fue exitosa de lo contrario nos retorna un false.

SD.remove(filename)
Elimina el archivo (filename) de la tarjeta SD, se debe de incluir el directorio. Solo elimina el archivo más no el directorio. Devuelve true se logra eliminar el archivo de lo contrario nos retorna un false.

SD.rmdir(dirname)
Eliminar el directorio (dirname) de la tarjeta SD. El directorio debe estar vacío. Devuelve TRUE si la eliminación del directorio tuvo éxito o FALSE en caso contrario.

SD.open(filepath, mode)
Abre el archivo especificado y se debe de incluir el directorio si el archivo está en carpetas. Si el archivo no existe, se creará un archivo con el nombre especificado, pero no será posible crear el directorio si este no existe. Se puede abrir un archivo como solo lectura (si mode es FILE_READ) o como lectura y escritura (si mode es FILE_WRITE), el modo por defecto en caso no se especifique es FILE_READ
Ésta función nos retorna un objeto tipo FILE, el cual es necesario declararlo antes como una variable. Por ejemplo:

```
File myFile;    myFile = SD.open("archivo.txt", FILE_WRITE);
```

Funciones de la clase File:

file.available()
Compruebe si hay bytes disponibles para leer en el archivo y retorna el número de bytes disponibles
file.read()
Lee un byte de la variable File (archivo abierto anteriormente con SD.open()).
file.write(data)
Escribe un byte en el archivo, el archivo debe estar abierto en modo lectura y escritura. Usando file.write(buf, len) se puede escribir un array de byte (buf) pero se debe especificar el tamaño (len).
file.print(data)
Esta función tiene las mismas características que un Serial.print(); data puede ser una variable o texto, el cual será enviado como caracteres. Si queremos agregar al final un salto o nueva línea se usa file.println(data).
file.size()
Retorna el tamaño en bytes del archivo.
file.position()
Retorna la posición actual en donde se leerá o escribirá el siguiente byte.
file.seek(pos)
Nos ubicamos en una posición específica en el archivo. *Pos* debe ser un número entre 0 y el tamaño en bytes del archivo
file.close()
Cerramos el archivo, y recién en este momento los datos se guardan en la SD, pudiendo extraer de forma segura nuestra SD.

Otros materiales utilizados

En este apartado se comentarán el resto de materiales utilizados en la parte referente a la mecánica de la máquina. Se hablará de los "husillos" utilizados y de la relación entre el diámetro y los pasos necesarios que debería realizar un motor para poder conseguir desplazar la estructura 1 milímetro, también se comentará la fuente de alimentación utilizada y los cojinetes y rodamientos que hemos utilizado para las partes móviles.

Husillos y tuercas

Para este proyecto hemos utilizado 3 varillas de tornillo como husillos, las cuales están unidas a los distintos motores mediante unos acoplamientos flexibles diseñados para adaptar el diámetro del eje del motor con el de los husillos. El funcionamiento es muy sencillo y muy utilizado en muchas maquinas que utilicen sistemas lineales donde se necesite desplazar partes estructurales sobre algún tipo de carriles, como por ejemplo en impresoras, fresadoras "CNC", etc. En girar el motor, la varilla que está unida a este, gira al mismo tiempo y de esta forma la tuerca que está unida a la parte móvil de la estructura se va desplazando en una dirección u otra sobre el tornillo según el sentido de giro de los motores, arrastrando consigo la parte estructural que se pretende mover.

No obstante en proyectos donde se precise un control exacto de los pasos necesarios que debe realizar el motor para poder asociar la distancia que debe recorrer dicha parte móvil a una equivalencia de distancia con unidades métricas como por ejemplo en el caso de una impresora 3D donde se pretende plasmar un diseño con unas medidas exactas de una pieza determinada la cual ha sido representada mediante un software tipo "Autocad", es necesario poder traducir y sincronizar esas unidades métricas de nuestro diseño digital, a número de pasos que deben realizar en todo momento los motores para poder reproducir las medidas exactas de la pieza que estamos imprimiendo. Para ello se recurre a una serie de cálculos con los cuales podemos obtener dicha equivalencia.

Material	Husillo de acero inoxidable Tuerca de cobre
Diámetro husillo	8mm
Longitud	2 x 600mm , 1x300mm
Espaciamiento de tornillo	2mm

Figura 25 Husillo y Tuerca

En un husillo podemos distinguir las siguientes características:

- **Número de entradas** (**Z)**, o filetes de rosca característica, que es el número de hélices que se enroscan en paralelo sobre el núcleo del tornillo. Generalmente es 1, 2 o 3.

- **El paso de rosca** (**P)**: es la distancia entre dos filetes consecutivos de una misma hélice. Habitualmente se mide en milímetros (mm). El paso de rosca es igual a la longitud que avanza el husillo en cada vuelta.

- **El avance (A)**: que es la distancia que avanza la tuerca al girar el husillo una vuelta completa:

$$\boldsymbol{A = Z \cdot P}$$

- **La longitud desplazada**, (**L)**: que es la distancia que se desplaza la tuerca cuando el husillo gira **n** vueltas completas.

$$\boldsymbol{L = A \cdot n = Z \cdot P \cdot n}$$

Finalmente para poder asociar dichas características de un husillo al número de pasos de nuestros motores como hemos comentado será necesario realizar una serie de cálculos como los del siguiente ejemplo:

Nuestro motor tiene una ángulo de paso de 1.8º, es decir cada vez que se le ordene al motor girar un paso, el eje de nuestro motor girará 1.8º, por lo tanto si sabemos que una vuelta completa son 360º, podemos deducir cuantos pasos necesitaríamos para lograr que nuestro motor de una vuelta completa.

$$\mathbf{360^{\circ}/1.8^{\circ} = 200\ pasos/vuelta}$$

Ahora bien, si de las características de nuestro husillo obtenemos que, con una vuelta completa, la tuerca avanza 2mm:

$$\mathbf{200/2 = 100\ pasos/mm}$$

Además, aquí nos puede variar la resolución de los drivers utilizados, y según se configuren los "jumpers" para determinar el modo de trabajo de los motores (fullsteap, halfsteap, etc…). De lo cual podemos deducir que la expresión adecuada para el cálculo de pasos es la siguiente:

$$\boldsymbol{Pulsos\ necesarios = \frac{pasos\ por\ vuelta\ motor}{paso\ husillo} * resolución\ del\ driver}$$

$$\boldsymbol{\frac{Distancia\ ordenada}{Distancia\ recorrida} * Pulsos\ idicados = Pulsos\ que\ debemos\ indicar}$$

Cojinetes y rodamientos

En este apartado solo se enseñarán los distintos rodamientos y cojinetes utilizados para las distintas partes móviles, aparentemente no parece una parte importante, no obstante ha sido bastante decisivo a la hora de desarrollar la estructura ya que gran parte de la complejidad de este proyecto y de su viabilidad después de realizar todo el trabajo de diseño estructural, y su coste económico asociado en la compra de las distintas piezas, recaía en que la maquina pudiera realizar los distintos movimientos de forma precisa una vez montada sin ningún problema de enganches o otros factores que pudieran sobrecargar los motores y posiblemente llegar al punto de tener que replantear o rehacer ciertas partes de la estructura con lo cual nos hubiera retrasado todo el proyecto.

Rodamientos carril Eje Y

Los rodamientos más importantes con los que hemos tenido que ser muy precisos a la hora de construir la pieza con las herramientas que se disponía, son los que utilizan el carril principal que se desliza sobre el "eje Y", el cual aparte de cargar con todo el peso de la estructura, también es el que tiene mayor capacidad de movimiento y está sometido a la fuerza de los dos motores que controlan dicho eje.

Los rodamientos utilizados, concretamente 4, son los mismos que se utilizan en las ventanas o puertas correderas de aluminio y son fáciles de encontrar en cualquier centro de bricolaje . Este carril se desplaza por unos railes de aluminio que se encuentran en ambos lados en los laterales de la estructura.

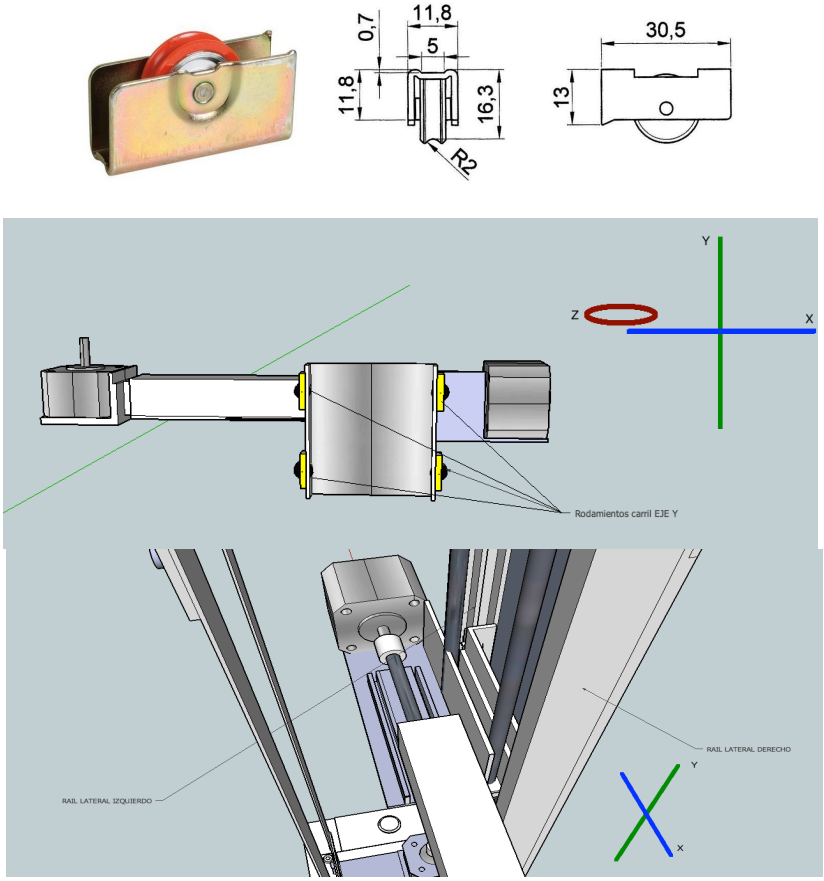


Figura 26 Rodamientos Eje Y

Rodamientos Eje X

Para este eje, se ha utilizado un sistema lineal de rodamientos que está pensado para ser utilizado en las cajoneras, para facilitar el arrastre y roce de los cajones sobre sus guías. Estos rodamientos son muy fáciles de instalar, su mecanismo está formado por dos railes de metal que se deslizan uno sobre el otro al mismo tiempo que una pequeñas bolas de metal hacen de rodamiento facilitando el movimiento y el roce entre dichos railes.

Material	Metal
Alto (cm)	28 cm
Ancho (cm)	3 cm
Peso soportado	15kg
Fondo	0.5 cm

Figura 27 Guía de metal para cajones

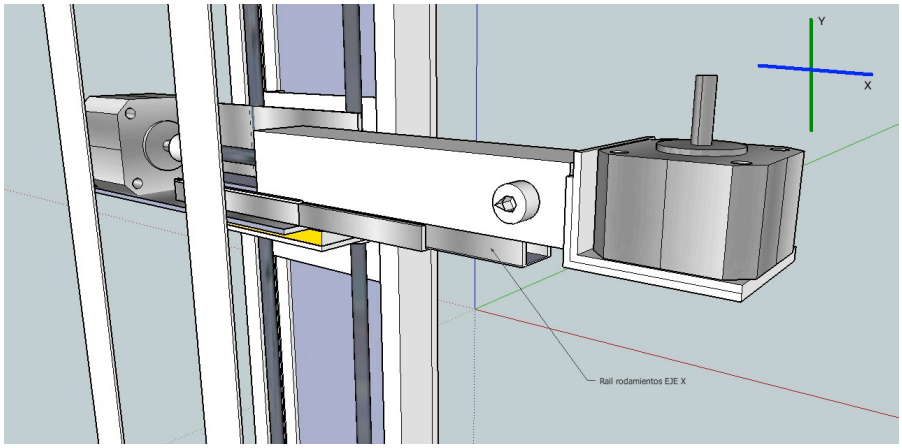


Figura 28 Rodamientos para el Eje X

Cojinetes para los husillos

Para facilitar el giro de los dos husillos del Eje Y se ha utilizado dos cojinetes especiales para este tipo de varillas. Básicamente se trata de dos rodamientos circulares a bolas, no obstante, están preparados para poder unirse a los husillos con un pequeño tornillo prisionero. Para instalar estos dos cojinetes se requiere cierta precisión ya que en nuestro caso los dos husillos han de estar completamente paralelos y equidistantes entre ellos a lo largo de todo el recorrido, ya que las tuercas están unidas al carril principal del eje Y, por lo tanto han de subir toda la estructura al mismo tiempo. Si hubiera algún problema con la instalación de dichos cojinetes, los husillos quedarían torcidos y en subir la estructura habría puntos donde los motores ejercerían mucha fuerza pudiendo llegar a romper cualquier pieza o quemar los drivers.

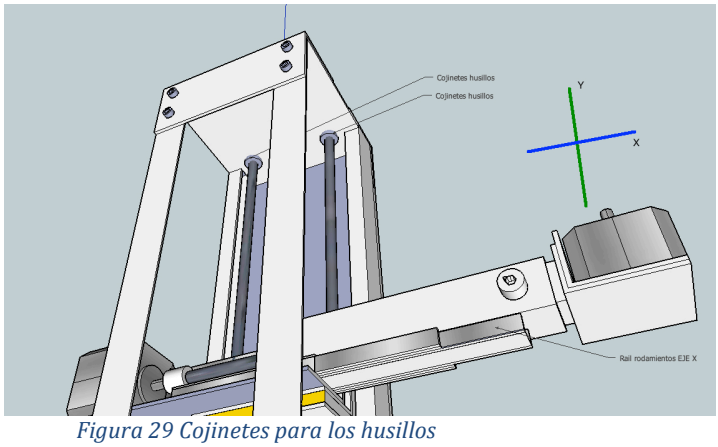


Figura 29 Cojinetes para los husillos

Diagrama general de conexiones

En la siguiente figura se muestra todo el conexionado de los distintos elementos que conforman la parte electrónica de nuestro prototipo así como la configuración de pines utilizados en la CNC Shield para la conexión de los módulos, para la SD Card y Bluetooth. Dado que estas últimas conexiones no se contemplan como conexiones dedicadas para dicho objetivo en el "layout" original de la CNC Shield, ya que el propósito de esta placa es para utilizarla en maquinas CNC o impresoras 3D que no requieren dichos módulos, hemos tenido que utilizar ciertos pines, que aunque están serigrafiados con otros nombres, están conectados directamente con los pines de entrada y salidas digitales de la placa de Arduino. También se indicará en este esquema la configuración de los *"jumpers"* utilizados para clonar el Eje Y para así poder utilizar dos motores con las mismas ordenes en dicho eje.

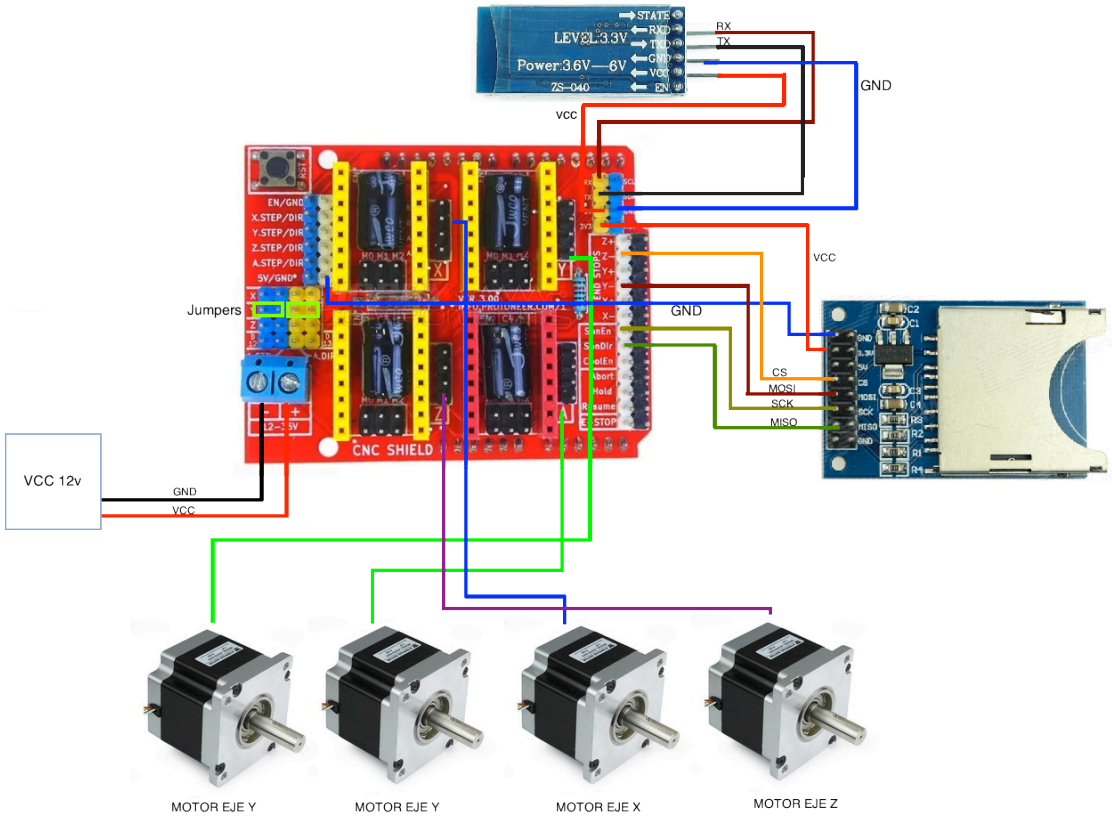


Figura 30 Esquema general de conexiones

Presupuesto

Como se puede observar en el ANEXO I, el presupuesto calculado para el desarrollo de este prototipo ha sido de 2.745€ aproximadamente teniendo en cuenta los distintos materiales utilizados y un cálculo aproximado de las horas de trabajo y otros gastos asociados.

Aplicaciones practicas

Este proyecto está enfocado como una herramienta de trabajo dentro del sector de los estudios de grabación. El problema se plantea cuando se requiere microfonear cierto instrumento, como podría ser amplificadores de guitarras, bajos, etc…, los cuales por norma general se suelen ubicar en cabinas de grabación acondicionas acústicamente para dicho fin. Normalmente la sala de control en un estudio de grabación suele estar en una habitación aparte por lo que muchas veces resulta muy tedioso tener que desplazarse a donde se encentra el instrumento para ubicar la micronfonía en la posición adecuada para obtener los resultados de calidad y sonido que se buscan para el proyecto en el que se está trabajando. El hecho de tener que desplazarse, conlleva perder la referencia de los monitores de audio, con lo cual los ajustes de los distintos micrófonos muchas veces son muy poco precisos y poco eficaces. Con este soporte de micrófono inalámbrico, dicho trabajo se puede realizar a distancia sin la necesidad de tener que desplazarse de la sala de control, logrando de esta manera un ajuste mucho más fino, versátil y a la vez productivo ya que el propio cliente / músico puede participar en el ajuste para lograr el sonido deseado.

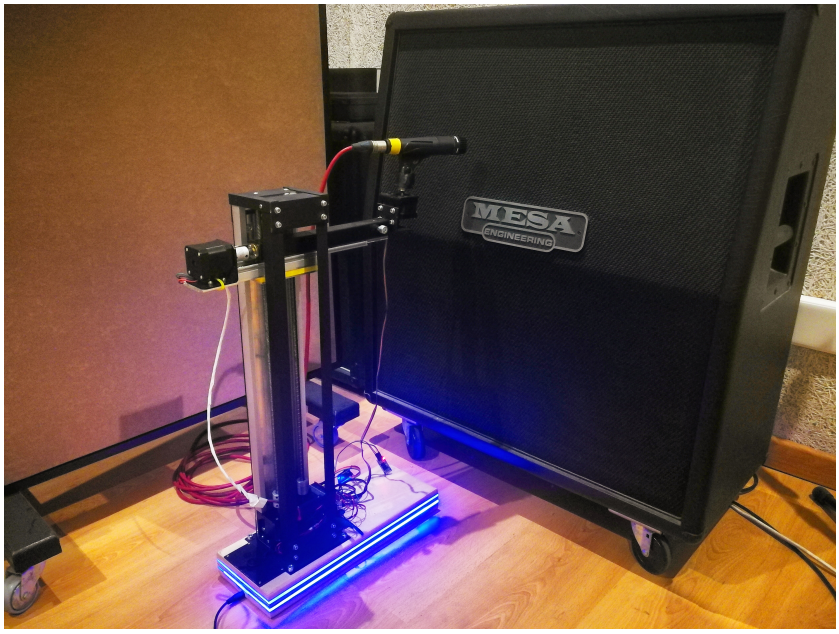


Figura 31 MicBot aplicación grabación de amplificadores

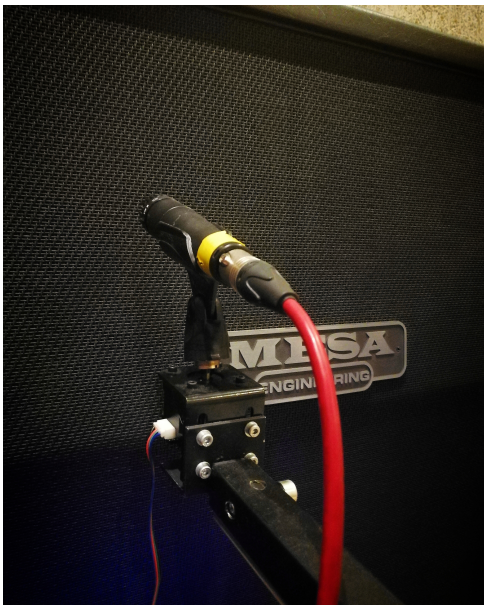


Figura 32 Mic Bot Eje Z

Figura 33 Mic Bot grabación de instrumentos acusticos

CAPÍTULO 4. DESARROLLO DEL SOFTWARE

Programación de la máquina

En este apartado se analizará el código implementado sobre la parte de la máquina el cual se ha desarrollado sobre la plataforma Arduino. En primer lugar, cabe decir que es una herramienta muy cómoda para realizar una gran variedad de proyectos DIY, sobre todo por la gran cantidad de información, ejemplos, librerías, etc… que se pueden descargar de su propia web. Gran parte del código utilizado se basa en el conjunto de ejemplos simples proporcionados en la web de arduino, o en el gran numero tutoriales que podemos encontrar en internet, lo cual hace muy intuitivo el hecho de reutilizar muchas partes de código de otras aplicaciones para adaptarlo a los fines de nuestro proyecto. También es de gran utilidad en gran número de librerías proporcionadas que mediante el uso de sus funciones se hace mucho más fácil el desarrollo de nuestro programa.

Librerías Utilizadas

Librería SPI.

Serial Peripheral Interface (SPI) es un protocolo de datos en serie síncrono utilizado por los microcontroladores para comunicarse rápidamente con uno o más dispositivos periféricos en distancias cortas. También se puede usar para la comunicación entre dos microcontroladores.

Funciones librería SPI:

- SPISettings
- begin ()
- end ()
- beginTransaction ()
- endTransaction ()
- setBitOrder ()
- setClockDivider ()
- setDataMode ()
- transfer ()
- usingInterrupt ()
- Due Extended SPI usage

Librería SD.

La librería SD permite leer y escribir en tarjetas SD, por ejemplo, en Arduino Ethernet Shield. Está construido en sdfatlib por William Greiman. La biblioteca es compatible con los sistemas de archivos FAT16 y FAT32 en tarjetas SD estándar y tarjetas SDHC. Utiliza nombres cortos de 8.3 para archivos. Los nombres de archivo pasados a las funciones de la biblioteca SD pueden incluir rutas separadas por barras diagonales, /,

por ejemplo, "directorio / nombrearchivo.txt". Debido a que el directorio de trabajo siempre es la raíz de la tarjeta SD, un nombre hace referencia al mismo archivo, ya sea que incluya una barra diagonal (por ejemplo, "/ archivo.txt" es equivalente a "archivo.txt"). A partir de la versión 1.0, la biblioteca admite la apertura de múltiples archivos.

Funciones librería SD:

- begin()
- exists()
- mkdir()
- open()
- remove()
- rmdir()
- name()
- available()
- close()
- flush()
- peek()
- position()
- print()
- println()
- seek()
- size()
- read()
- write()
- isDirectory()
- openNextFile()
- rewindDirectory()

Librería ArduinoJson.h

JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos. Es fácil para las máquinas analizar y generar. Se basa en un subconjunto del lenguaje de programación JavaScript, estándar ECMA-262 3ª edición, diciembre de 1999 . JSON es un formato de texto que es completamente independiente del lenguaje, pero utiliza convenciones que son familiares para los programadores de la familia C de idiomas, incluidos C, C ++, C #, Java, JavaScript, Perl, Python y muchos otros. Estas propiedades hacen de JSON un lenguaje ideal de intercambio de datos.
Esta librería se ha utilizado para el almacenamiento de los valores de posición de los distintos motores en la SD card. Con Json hemos podido parsear la información guardada y poder obtener el valor de las variables de posición que configuran la maquina o almacenan un preset de posición .

Funciones implementadas

FUNCIONES DE MOVIMIENTO DE MOTORES

- moveStepper(int dir, int _position)

Se utiliza para el movimiento manual de los distintos ejes.

- moveStepperToPreset(int pos_x, int pos_y, int pos_z)
  Con esta función los motores se desplazaran automáticamente a la posición guardada en un determinado preset..

FUNCIONES DE TRATAMIENTO DE LOS ARCHIVOS

- saveConfiguration(const char *fileName)
  Esta función se utiliza para almacenar en la SD Card la posición de cada uno de los ejes cada vez que se realiza un desplazamiento en el archivo CONFIG.TXT .
- loadConfiguration(const char *filename)
  Se utiliza para cargar la información guardada en el archivo CONFIG.TXT cada vez que se reinicie la máquina, para poder cargar el valor a las variables internas.

FUNCIONES DE MANEJO DE LOS PRESETS

- savePreset(const char *_name)
  Esta función se utiliza cada vez que el usuario quiere almacenar un nuevo preset en la memoria de la SD Card.
- getPresets()
  Se utiliza para mostrar la lista de presets almacenados en la memoria.
- loadPreset(const char *_name)
  Esta función se utiliza para cargar un preset por el nombre, almacenado en la memoria de la SD Card .
- removePresetByName(String _name)
  Esta función se utiliza para eliminar un preset por el nombre, almacenado en la memoria de la SD Card.

**El código completo implementado se puede observar en el ANEXO II.*

Programación de la Aplicación de control

Para implementar dicha aplicación de control se utilizó en un principio la herramienta de Google "App Inventor" , la cual es una opción muy intuitiva y fácil para muchas aplicaciones donde no se requiera una gran complejidad. No obstante, para implementar la parte relacionada con almacenar presets de posiciones, donde el usuario pueda crear sus propias configuraciones, con un nombre personalizado, al mismo tiempo de poder cargar o eliminar un preset, todo estas acciones aumentaron la complejidad que se requeriría para el desarrollo de nuestra aplicación , con lo cual tuve que recurrir a otros métodos de creación de aplicaciones para dispositivos móviles como Apache Cordova o PhoneGap. Con dicha herramienta podemos implementar con un mismo código una aplicación hibrida que sea compatible en sistemas Android, Windows, iOS y OSX , con lo cual es una gran ventaja.

Apache Cordova

Apache Cordova es un marco de desarrollo móvil de código abierto. Permite utilizar las tecnologías estándar web como HTML5, CSS3 y JavaScript para

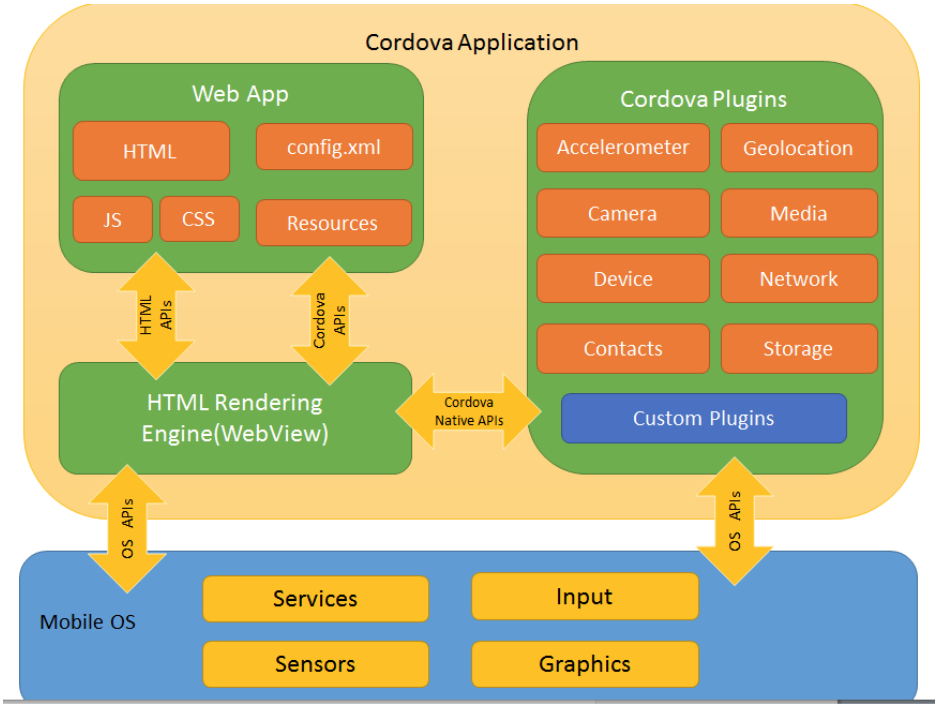
desarrollo multiplataforma, evitando el lenguaje de desarrollo nativo cada plataformas móviles. Las aplicaciones se ejecutan dentro de envolturas para cada plataforma y dependen de enlaces estándares API para acceder a los sensores de cada dispositivo, datos y estado de la red.
Apache Cordova se surgió en octubre de 2012 como un proyecto de nivel superior dentro de la Apache Software Foundation (ASF). Cordova siempre permanecerá libre y de código abierto bajo la licencia Apache, versión 2.0.

CSS : (siglas en inglés de Cascading Style Sheets), en español "Hojas de estilo en cascada", es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML o XHTML; el lenguaje puede ser aplicado a cualquier documento XML, También permite aplicar estilos no visuales, como las hojas de estilo auditivas.
Junto con HTML y JavaScript, CSS es una tecnología usada por muchos sitios web para crear páginas visualmente atractivas, interfaces de usuario para aplicaciones web, y GUIs para muchas aplicaciones móviles (como Firefox OS).

HTML: sigla en inglés de HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros.

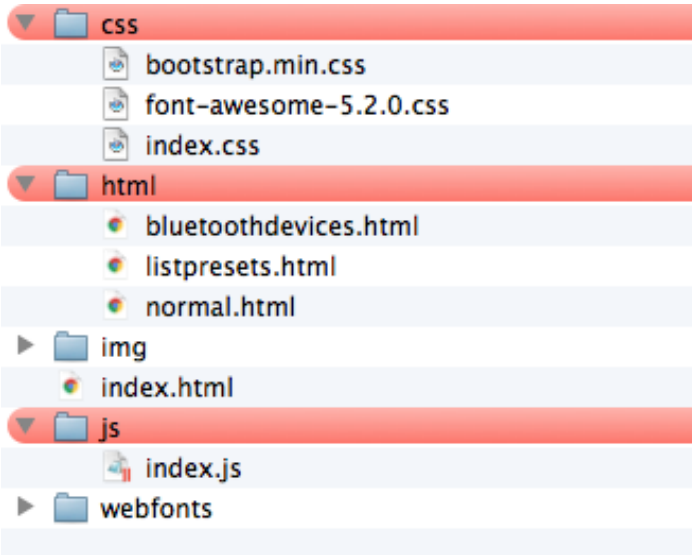
JavaScript (JS): es un robusto lenguaje de programación que puede ser aplicado a un documento HTML y usado para crear interactividad dinámica en los sitios web. Fue inventado por Brendan Eich, co-fundador del proyecto Mozilla, Mozilla Foundation y la Corporación Mozilla .
Con JavaScript se pueden implementar muchas cosas como carruseles, galerías de imágenes, diseños fluctuantes, y respuestas a las pulsaciones de botones. También es capaz de crear juegos, animaciones 2D y gráficos 3D, aplicaciones integradas basadas en bases de datos, etc.

Arquitectura de las aplicaciones Cordova



Diseño de la Aplicación

En la siguiente imagen se puede observar los distintos ficheros que se utilizan para la creación de nuestra aplicación.



Carpeta CSS

En primer lugar dentro de la carpeta CSS, la cual contiene archivos cuyo código está relacionado con el aspecto visual y estético basado en Bootstrap.
Bootstrap es un framework desarrollado y liberado por Twitter que tiene como objetivo facilitar el diseño web. Permite crear de forma sencilla webs de diseño adaptable, es decir, que se ajusten a cualquier dispositivo y tamaño de pantalla y

siempre se vean igual de bien. Es Open Source o código abierto, por lo que lo podemos usar de forma gratuita y sin restricciones.

En el fichero "Index.css" se definirán los parámetros como tamaño, color, alineación etc… de los distintos componentes que conformaran nuestra aplicación , como los botones, pantallas, etc.

***Ver Codigo en ANEXO III*

Carpeta HTML

Esta carpeta contiene las distintas "ventanas" o "template" de las que constará nuestra aplicación. Como se puede observar hemos creado tres ventanas o paginas las cuales contendrán una serie de elementos "Css" necesarios para desempeñar el interfaz visual de control con el que interactuara el usuario para realizar las funciones que necesitamos para cada acción, cuya logia se desarrollará en el fichero de Java Script que veremos a continuación.

Las pantallas que hemos creado son las siguientes :

Pantalla principal (Normal)

En esta imagen se muestra el diseño de la pantalla principal donde podemos observar los distintos botones de control de los motores en la parte superior, también se puede observar el menú para las opciones de gestión de los presets.

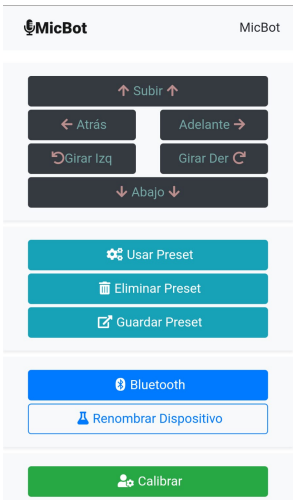


Figura 34 Pantalla principal APP

Pantalla de vinculación con el modulo Bluetooth (bluetoothdevice)

En esta pantalla se muestra distintos dispositivos Bluetooth, concretamente el nombre de "MicBot" es la máquina con la que debemos establecer la conexión. Para realizar y gestionar estas funciones se utilizado un plugin (bluetoothSerial.list) para facilitar el desarrollo de todo lo referente a la

sincronización del dispositivo bluetooth de la maquina con el de nuestro dispositivo móvil y a la transferencia vía serie de los distintos datos que se transmitirán en cada una de las utilidades de la aplicación.

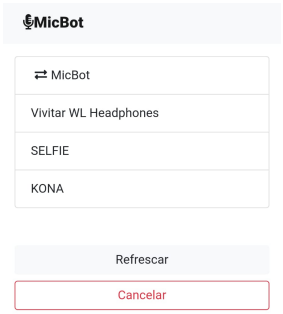


Figura 35 Pantalla Bluetooth

Pantalla creación de presets (listpresets)

Para crear un nuevo preset cabe destacar que utiliza la notación 8.3, es decir un máximo de 8 caracteres que podrá introducir como nombre el propio usuario , más otros 3 caracteres que determinan la extensión , en este caso serán archivos TXT. Cada preset se guardará en la SD Card como un nuevo archivo TXT con las posiciones de los distintos ejes y con el nombre que ha introducido el usuario en esta pantalla de creación.

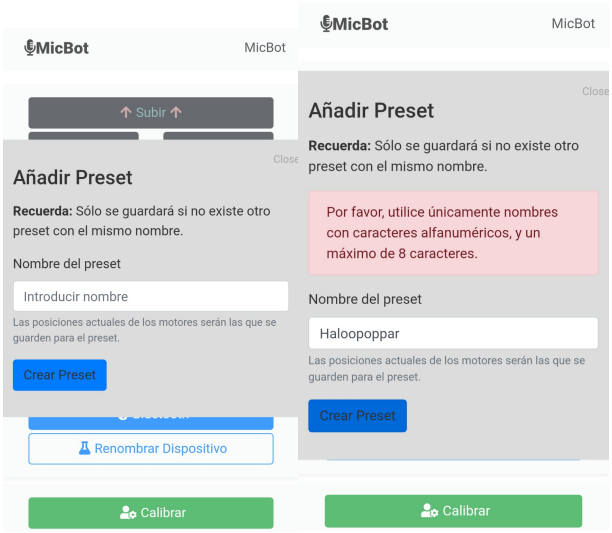


Figura 36 Pantalla creación de presets

Carpeta Js (Java Script)

En esta carpeta es donde podemos encontrar el fichero que contiene el código más directo con las funciones de nuestra aplicación. Es donde se implementa las

distintas funciones y la lógica necesaria para interactuar con las funcionalidades de nuestra máquina.

***En el ANEXO IV, se puede observar el código implementado en JS.*

***En el ANEXO V, se puede observar las distintas funciones implementadas donde se explica su utilidad en cada una de ellas.*

CAPÍTULO 5. OPERACIÓN DE MANTENIMIENTO DEL SISTEMA

En este apartado se comentarán los distintos aspectos y consejos a tener en cuenta a la hora de empezar a utilizar la máquina. En primer lugar cabe destacar los dos modos de funcionamiento con los que se puede trabar, el modo “manual” o modo “automático” basado en la utilización de presets de posiciones de los motores, ya que este último modo de funcionamiento está condicionado a utilizar la máquina en un sitio fijo, sobre una misma fuente ya que una vez se cambie de sitio o cambie la fuente de sonido a registrar, los presets previamente almacenados pueden no coincidir con la posición deseada, en resumen ; dicho modo de funcionamiento está enfocado en entornos de trabajo controlados donde la máquina está en un punto fijo sobre una misma fuente sonora como una misma pantalla de altavoces, mismo instrumento, etc, o durante una misma sesión de grabación donde el micrófono y la máquina estarán utilizándose en un punto fijo a lo largo de dicha sesión.

Por otro lado el modo manual, como su propio nombre indica, se tendrá que ubicar el micrófono de forma manual con los controles de posición de la aplicación por lo que no es necesario situar la maquina en un punto fijo, ya que supuestamente dicha ubicación podría cambiar en futuras ocasiones.

Calibración

El modo calibración se utiliza para ajustar la posición “0” de los motores en los distintos ejes. En un principio la maquina tiene guardada dicho límite inferior y superior como finales de carrera en su memoria SD Card, en un archivo llamado “CONFIG.TXT” , no obstante es posible que en situaciones donde la máquina este en movimiento ocurra un fallo de alimentación donde la maquina no haya podido almacenar la nueva posición a la que se estaba desplazando , con lo cual es posible que donde se hayan quedado los distintos motores no corresponda con la ubicación exacta respecto a los valores almacenados en el archivo “CONFIG.TXT”, por lo que ocasionará un desajuste de los límites de desplazamiento en los distintos ejes.

Para entrar en modo “Calibración” hay que pulsar la opción en la pantalla de la aplicación y situar los distintos ejes en la posición “0”.

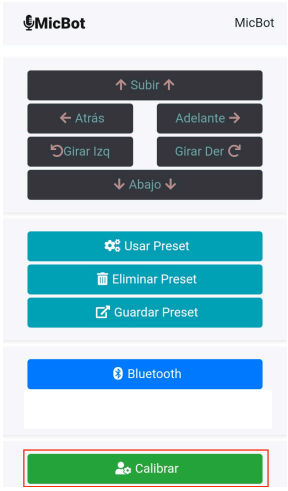


Figura 37 Calibrar

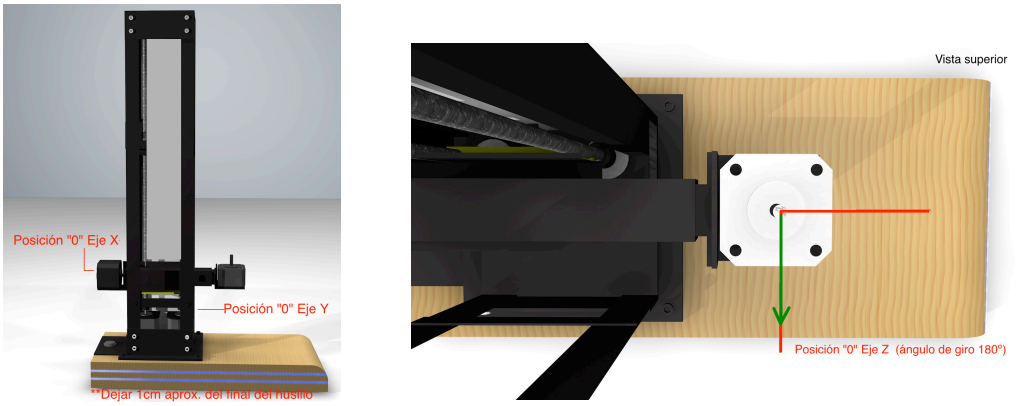


Figura 38 Posiciones "0" de los distintos Ejes

Una vez se ha situado los distintos ejes en su posición “0” se volverá a pulsar el botón de calibrar , el cual cambia de color de rojo a verde, lo cual nos indica que dichas posiciones serán ahora nuestro límite inferior el cual a la hora de utilizar la máquina , ya no se podrá sobrepasar dicho límite cuando utilicemos los controles de movimiento. Es importante ajustar bien este limite inferior , ya que va asociado al límite superior del recorrido, es decir, si calibramos el límite inferior a la mitad del recorrido de nuestro eje Y por ejemplo , la maquina no funcionara correctamente y podríamos romper la maquina al subir dicho eje, más de lo que el recorrido físico nos permite , ya que a nivel de software , el eje superior se ubicaría en este caso fuera de la longitud física de la máquina.

Mantenimiento

El mantenimiento aconsejable hasta la fecha de este primer prototipo, se basa simplemente en la correcta lubricación de los distintos husillos y cojinetes con algún tipo de aceite industrial, con la finalidad de mejorar el correcto funcionamiento, reducir la fricción y ruidos causados por el desplazamiento o vibración de las distintas partes móviles. Se aconseja no cubrir las partes electrónicas, concretamente los drivers de los motores para mejorar la ventilación y disipación del calor.

- Se aconseja apagar la maquina antes de extraer la memoria SD Card, ya que podría ocasionar fallos de configuración.
- Se aconseja no superar el peso de 2kg en la fuente de captación utilizada.
- Se aconseja no trabajar en las posiciones de los extremos de cada eje, con el fin de evitar los puntos en los que se ejerce mayor fuerza sobre la estructura. Es preferible situar físicamente la maquina en una posición adecuada donde se tenga margen de movimiento.
- Se aconseja apagar la maquina después de su uso, o en largas sesiones de trabajo donde tenga que estar en un punto fijo y no se requiera realizar más movimientos con el fin de evitar el consumo innecesario.
- No exponer la maquina en condiciones de humedad o lluvia, hay muchas partes metálicas y electrónicas que podrían deteriorarse.

CONCLUSIÓN

En conclusión, este proyecto ha sido una experiencia bastante próxima al proceso de emprender una idea de desarrollo de un producto donde desde un primer momento se han tenido que abordar muchísimas dificultades e inconvenientes a la hora de crear este primer prototipo. Por otra parte, el hecho de ser un trabajo propuesto por mí mismo me ha servido mucho para aprender muchas cosas de una forma mucho más amena lo cual ha motivado mi interés desde un primer momento para aprender todas las herramientas necesarias para crear este proyecto. No obstante el hecho de crear un diseño de una maquina desde cero, asumiendo el coste de todos sus materiales, ha implicado un mayor tiempo de trabajo previo a la construcción de la primera estructura, ya que de esta dependía el resto del proyecto, ya que no tendría mucho sentido desde mi punto de vista y mis objetivos para este proyecto implementar una aplicación en la que no hubiera una maquina física donde plasmar los resultados funcionales, en el caso de que por motivos de diseño o económicos no hubiera logrado construirla.
En resumen, ha sido una buena experiencia y a la vez muy satisfactoria al ver los resultados finales de todo el trabajo.
Por otra parte, el mundo de la tecnología avanza muy rápidamente, esta idea de trabajo, es algo que llevaba muchos años intentando realizar, concretamente después de haber realizado en su momento, un ciclo de grado superior de electrónica, el cual ya empecé a plantearme llevarlo a cabo en un futuro ya que en su momento no había nada por el estilo y me parecía un proyecto interesante para mi trabajo como Técnico de sonido. Actualmente hay muchísima información al respecto en internet con muchísimos tutoriales, ejemplos, proyectos que comparten otras personas, etc; y todo sobre plataformas de desarrollo tipo arduino que hacen muchísimo más fácil y económico el crear tu propio proyecto electrónico para una infinidad de aplicaciones.

Para llevar a cabo este proyecto, he tenido que utilizar y aprender distintas herramientas y entornos de trabajo, de los cuales algunos referentes al diseño 3D ya los conocía por haberlos utilizado anteriormente por mi cuenta en otras ocasiones, no obstante para todo lo referente a la programación del código para este proyecto, he tenido que buscar información por internet en distintos foros, tutoriales, preguntar y recurrir a distintos compañeros que tenía más experiencia en cierto tipo de programación para resolver dudas, como es en el caso de aplicaciones móviles, etc… Por otra parte, también he tenido que comprar y aprender a utilizar cierto tipo de maquinaria y herramientas de trabajo, tipo sierras de disco metálicas, fresadoras, taladros, soldadores de arco, etc. En resumen, a continuación, describiré una lista de las cosas que he tenido que utilizar y en muchos casos aprender desde cero:

-Utilización de ciertas herramientas de construcción para la estructura.
-Software de diseño gráfico tipo Sketchup o Autocad.
-Software de diseño 3D tipo “Cinema 4D”.
- Programación en un entorno Arduino.
- Electrónica de control de motores.
- Programación de aplicaciones móviles.

Futuras ideas para seguir con el desarrollo

Como idea general para seguir en un futuro con el desarrollo de este proyecto, sobretodo, después de ver que ya se comercializa un producto similar, me parece una buena opción enfocar y rediseñar esta máquina como un Kit de montaje basado en un diseño de impresión de la estructura 3D para facilitar la construcción y seguramente el coste de esta. Sería muy interesante lograr hacer un diseño más compacto y atractivo que cualquiera pueda montar en su casa como un proyecto DIY, ya que es una idea que cada vez está más de moda y mucha gente opta por dicha opción por tema económico y por la satisfacción de montar por uno mismo su propio material técnico.

En este prototipo nos hemos dado cuenta de las inconvenientes y cosas que se deberían mejorar para un fututo; por ejemplo, no sería necesario utilizar dos motores para el Eje Y, ya que un motor de los utilizado sería suficiente para arrastrar el peso de un micrófono convencional. Por otro lado, se podría mejorar el ruido estructural, el cual simplemente rehaciendo el diseño actual y cambiando el material de aluminio, a plástico utilizando un diseño para imprimir en 3D, con lo cual esto creo que ya solucionaría en gran medida este problema.

Referente a la parte del software, es algo que se irá analizando y mejorando a largo plazo analizando el comportamiento de la maquina una vez este expuesta a condiciones reales de trabajo y se pueda evaluar mejor el comportamiento de esta para añadir mejoras o cambios.

BIBLIOGRAFÍA

https://www.diarioelectronicohoy.com/blog/motores-bipolares

https://rydepier.wordpress.com/2016/03/31/uln2003-stepper-motor-driver-with-28byj-48-motor/

http://server-die.alc.upv.es/asignaturas/LSED/2002-03/MotoresPasoaPaso/tipos.htm

https://www.staticboards.es/blog/drv8825-vs-a4988/

https://www.diarioelectronicohoy.com/blog/motores-bipolares

https://elmassolicitadopayan.wordpress.com/2014/04/11/motor-paso-a-paso/

http://www.sigmaelectronica.net/manuals/HOJA%20REFERENCIA%20TARJETA%20HC-05%20ARD.pdf

https://en.wikipedia.org/wiki/Unipolar_motor&prev=search

https://es.wikipedia.org/wiki/Husillo

http://www3.gobiernodecanarias.org/medusa/ecoblog/ralvgon/files/2013/05/Características-Arduino.pdf
https://cordova.apache.org/docs/en/8.x/guide/overview/index.html

https://github.com/