



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

**“DISEÑO DE RUTINAS EMPLEANDO SISTEMA ARDUINO Y
MINICOMPUTADORA RASPBERRY PI PARA EL CONTROL
DE IMPRESORAS 3D CON APLICACIÓN PRÁCTICA
UTILIZANDO PROTOTIPO”**

TESINA DE SEMINARIO

Previa la obtención del Título de:

INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN INDUSTRIAL

Presentado por:

Hernán Heriberto Vera Rodríguez

Juan Carlos Sacoto Marquina

GUAYAQUIL – ECUADOR

AÑO 2014

AGRADECIMIENTO

A Dios todo poderoso por permitirme llegar a esta etapa de mi vida, gracias por su infinita misericordia y por darme toda la sabiduría para seguir adelante.

A mis padres y familiares ya que ellos han sido el pilar fundamental de mi vida y todo lo que he logrado es gracias al esfuerzo y sacrificio de ellos.

A mis amigos y compañeros de estudio, gracias por estar siempre ahí con sus consejos en los buenos y malos momentos.

A mi director de tesis Ing. Carlos Valdivieso y a todos mis profesores durante mis años de estudio.

Hernán Heriberto Vera Rodríguez

AGRADECIMIENTO

Infinitamente a Dios por su amor y su misericordia y por darme una salida a cada prueba puesta en mi camino, ya que apartado de él nada puedo hacer.

A mis padres que siempre estuvieron inculcándome el temor a Dios, dándome principios y valores morales que me acompañarán por siempre, gracias por sus esfuerzos y sacrificios sin pedir nada a cambio.

A mis hermanos, que han estado siempre juntos a mí apoyándome y motivándome a ser mejor cada día.

A mi director de tesis Ing. Carlos Valdivieso y a cada uno de mis profesores.

Juan Carlos Sacoto Marquina

DEDICATORIA

A Dios, a mis padres, a mis abuelos y a toda mi familia, también a todos mis amigos y a todos aquellos que siempre han estado apoyándome durante mi carrera universitaria.

Hernán Heriberto Vera Rodríguez

DEDICATORIA

A Dios, quien siempre ha estado siempre dándome nuevas fuerzas para llegar a este tiempo de culminación.

A mis padres por todos los sacrificios hechos para verme realizado como profesional.

A cada uno de los profesores que siempre estuvieron desafiándome a nuevos retos, motivándome para llegar al final de este proceso y a cada uno que hizo posible que llegue con éxito a este punto.

Juan Carlos Sacoto Marquina

TRIBUNAL DE SUSTENTACIÓN



Ing. Carlos Valdivieso A.

PROFESOR DEL SEMINARIO DE GRADUACIÓN



Ing. Hugo Villavicencio

PROFESOR DELEGADO POR LA UNIDAD ACADÉMICA

DECLARACIÓN EXPRESA

“La responsabilidad del contenido de esta Tesina, nos corresponde exclusivamente; y el patrimonio intelectual del mismo a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL”.

(Reglamento de exámenes y títulos profesionales de la ESPOL)



Hernán Heriberto Vera Rodríguez



Juan Carlos Sacoto Marquina

RESUMEN

El presente proyecto de graduación tiene como finalidad el diseño de rutinas para controlar una impresora 3D, utilizando dos tarjetas electrónicas como lo son la Raspberry Pi y el Arduino Mega 2560, la primera tarjeta mencionada es una computadora de bolsillo llamada así debido a su tamaño el cual es similar a una tarjeta de crédito, mientras que el Arduino Mega es una tarjeta la cual contiene un microcontrolador incorporado el cual es programado para realizar distintos tipos de aplicaciones como lo son, el control de los motores de los ejes de la impresora, control del motor extrusor, control de la punta extrusora y el control de los finales de carrera.

La Raspberry Pi hará la función de un ordenador cuyo sistema operativo es Raspbian "Wheezy" que es una distribución de Linux, por medio del software de control Pronterface vamos a controlar la impresora 3D, este programa contiene el software Skeinforge el cual se encarga de generar el código G o Gcode a partir de un archivo.STL de un diseño 3D, este código es enviado desde la Raspberry Pi hacia el Arduino para que este lo interprete por medio de su firmware logrando que los motores realicen el correcto desplazamiento para que la extrusora imprima los diseños 3D capa por capa.

ÍNDICE GENERAL

AGRADECIMIENTO	I
DEDICATORIA	III
TRIBUNAL DE SUSTENTACIÓN	V
DECLARACIÓN EXPRESA	VI
RESUMEN	VII
ÍNDICE GENERAL.....	VIII
ÍNDICE DE FIGURAS.....	XV
INTRODUCCIÓN.....	XX
CAPÍTULO 1	1
DESCRIPCIÓN GENERAL DEL PROYECTO	1
1.1. Antecedentes	1
1.2. Motivación	2
1.3. Objetivos	3
1.4. Justificación del Proyecto.....	4
1.5. Descripción del Proyecto.....	4

1.6. Limitaciones	6
CAPÍTULO 2.....	8
FUNDAMENTO TEÓRICO	8
2.1. Requerimientos para aplicación del proyecto.....	8
2.2. Herramientas de software	9
2.2.1. Firmware Sprinter y Arduino IDE	10
2.2.2. Software de control (Pronterface)	11
2.2.3. Software de control (Repetier Host)	12
2.2.4. Software de generación de Gcode (Skeinforge)	13
2.2.5. Software de generación de Gcode (Slic3r)	15
2.3. Herramientas de hardware	16
2.3.1. Tarjeta Arduino Mega 2560.....	16
2.3.2. Tarjeta Raspberry Pi	17
2.3.2.1. El puerto GPIO de la Raspberry Pi	19
2.3.3. Protoboard	19
2.3.4. Fuente ATX.....	20

2.3.5. Kit de la Impresora 3D Prusa Mendel	22
2.3.6. RepRap Arduino Mega Pololu Shield (RAMPS).....	23
2.3.7. Motores paso a paso.....	26
2.3.7.1. Motores paso a paso Bipolares	27
2.3.7.2. Motores paso a paso Unipolares	28
2.3.7.3. Secuencias para controlar motores paso a paso bipolares.....	30
2.3.7.4. Secuencias para controlar motores paso a paso unipolares	31
2.3.7.5. Identificación de los terminales de un motor paso a paso.....	33
2.3.8. Drivers para controlar los motores de la impresora.....	35
2.3.8.1. Conexiones del driver A4988.....	36
2.3.9. Otros dispositivos y periféricos	37
CAPÍTULO 3.....	38
DISEÑO E IMPLEMENTACIÓN DEL PROYECTO.....	38
3.1. Introducción.....	38

3.2. Ejercicio 1: Comunicación entre Raspberry Pi y Arduino vía puerto USB.....	39
3.2.1. Diagrama de bloques del Ejercicio 1.....	40
3.2.2. Diagrama de flujo del Ejercicio 1.....	40
3.2.3. Descripción del algoritmo del Ejercicio 1.....	41
3.2.4. Código fuente del Ejercicio 1.....	41
3.3. Ejercicio 2: Control de leds con PWM.....	42
3.3.1. Diagrama de bloques del Ejercicio 2.....	42
3.3.2. Diagrama de flujo del Ejercicio 2.....	43
3.3.3. Descripción del algoritmo del Ejercicio 2.....	44
3.3.4. Código fuente del Ejercicio 2.....	44
3.4. Ejercicio 3: Control de un motor paso a paso bipolar mediante el uso del driver L293D.....	46
3.4.1. Diagrama de bloques del Ejercicio 3.....	46
3.4.2. Diagrama de flujo del Ejercicio 3.....	47
3.4.3. Descripción del algoritmo del Ejercicio 3.....	48
3.4.4. Código fuente del Ejercicio 3.....	49

3.5. Proyecto final de graduación: Diseño de rutinas empleando Sistema Arduino y minicomputadora Raspberry Pi para el control de impresoras 3D con aplicación práctica utilizando prototipo.....	50
3.5.1. Diagrama de bloques del proyecto final.....	51
3.5.2. Diagrama de flujo del proyecto final.....	52
3.5.3. Descripción del algoritmo del proyecto final.....	53
3.5.4. Código fuente del proyecto final.....	54
CAPÍTULO 4.....	58
SIMULACIÓN Y PRUEBAS DEL PROYECTO	58
4.1. Introducción.....	58
4.2. Simulaciones y pruebas en protoboard	58
4.2.1. Ejercicio 1: Comunicación entre Raspberry Pi y Arduino vía puerto USB.....	59
4.2.1.1. Conexiones del Ejercicio 1.....	60
4.2.1.2. Simulación del Ejercicio 1	61
4.2.1.3. Conclusiones del Ejercicio 1	62
4.2.2. Ejercicio 2 : Control de Leds con PWM.....	63

4.2.2.1. Conexiones del Ejercicio 2.....	63
4.2.2.2. Simulación del Ejercicio 2.....	64
4.2.2.3. Conclusiones del Ejercicio 2.....	65
4.2.3. Ejercicio 3: Control de un motor de paso bipolar mediante el uso del driver L293D	66
4.2.3.1. Conexiones del Ejercicio 3.....	68
4.2.3.2. Simulación del Ejercicio 3.....	69
4.2.3.3. Conclusiones del Ejercicio 3.....	70
4.3. Proyecto final de graduación: Diseño de rutinas empleando Sistema Arduino y minicomputadora Raspberry Pi para el control de impresoras 3D con aplicación práctica utilizando prototipo.....	71
4.3.1. Calibración del firmware Sprinter	72
4.3.2. Conexión de la RAMPS 1.4 con la impresora 3D	73
4.3.2.1. Conexión de los drivers A4988.....	74
4.3.2.2. Conexión de los motores paso a paso.....	75
4.3.2.3. Conexión de los finales de carrera mecánicos	76
4.3.2.4. Conexión de la punta extrusora.....	78

4.3.3. Prueba del firmware Sprinter con Pronterface	79
4.3.4. Como imprimir un diseño 3D.....	81
4.3.5. Primeros objetos impresos.....	82
CONCLUSIONES	85
RECOMENDACIONES.....	87
BIBLIOGRAFÍA	89
ANEXOS	92

ÍNDICE DE FIGURAS

Figura 2.1 Arduino IDE con el firmware Sprinter.....	11
Figura 2.2 Software de control Pronterface.	12
Figura 2.3 Software de control Repetier-Host.....	13
Figura 2.4 Calibración de Skeinforge.....	14
Figura 2.5 Calibración de slic3r.....	15
Figura 2.6 Arduino Mega 2560.....	17
Figura 2.7 Raspberry Pi.....	18
Figura 2.8 Puerto GPIO de la Raspberry Pi.....	19
Figura 2.9 Implementación de un circuito en protoboard.	20
Figura 2.10 Fuente ATX.....	21
Figura 2.11 Puente entre el cable verde y negro.	21
Figura 2.12 Voltajes que proporciona la fuente ATX.....	22
Figura 2.13 Impresora 3D Prusa Mendel armada	23
Figura 2.14 RAMPS 1.4	24
Figura 2.15 Conexión de la RAMPS 1.4.	25

Figura 2.16 Motor paso a paso.	26
Figura 2.17 Relación entre el pulso y los pasos del motor.....	27
Figura 2.18 Motor paso a paso Bipolar.	27
Figura 2.19 Control de un motor paso a paso bipolar mediante el uso del driver L293.....	28
Figura 2.20 Motor paso a paso Unipolar de 5 cables de salida	29
Figura 2.21 Motor paso a paso Unipolar con 6 cables de salida.....	29
Figura 2.22 Control de un motor paso a paso unipolar mediante el uso del driver ULN2803.....	30
Figura 2.23 Secuencia para controlar un motor paso a paso Bipolar.	30
Figura 2.24 Secuencia normal.	31
Figura 2.25 Secuencia del tipo wave drive.	32
Figura 2.26 Secuencia del tipo medio paso	33
Figura 2.27 Identificación de un motor de 4 terminales	34
Figura 2.28 Identificación de un motor de 5 terminales	34
Figura 2.29 Identificación de un motor de 6 terminales	35
Figura 2.30 Driver A4988 de Pololu	35

Figura 2.31 Esquema de conexión del driver A4988.	36
Figura 2.32 Dispositivos y periféricos.	37
Figura 3.1 Diagrama de bloques del Ejercicio 1.....	40
Figura 3.2 Diagrama de flujo del Ejercicio 1.	40
Figura 3.3 Modulación por Ancho de Pulso PWM.....	42
Figura 3.4 Diagrama de bloques del Ejercicio 2.....	42
Figura 3.5 Diagrama de flujo del Ejercicio 2.	43
Figura 3.6 Diagrama de bloques del Ejercicio 3.....	46
Figura 3.7 Diagrama de flujo del Ejercicio 3.	47
Figura 3.8 Diagrama de bloques del proyecto final.....	51
Figura 3.9 Diagrama de flujo del proyecto final.....	53
Figura 4.1 Igualando voltajes.	59
Figura 4.2 Comunicación entre Raspberry Pi y Arduino vía puerto USB.	60
Figura 4.3 Envío de mensajes entre Raspberry Pi y Arduino.....	61
Figura 4.4 Esquema de conexión del Ejercicio 2.	63
Figura 4.5 Simulación del Ejercicio 2.	64

Figura 4.6 Control de un motor bipolar mediante el driver L293D	66
Figura 4.7 Relación de la señal PWM con el voltaje efectivo.....	67
Figura 4.8 Esquema de conexiones del motor con el driver L293D.....	68
Figura 4.9 Simulación del Ejercicio 3.	69
Figura 4.10 Conexión de la RAMPS 1.4 con la impresora 3D	73
Figura 4.11 Ajuste de los drivers A4988.	74
Figura 4.12 Conexión de los motores paso a paso.....	75
Figura 4.13 Final de carrera mecánico	76
Figura 4.14 Conexión de los finales de carrera mecánicos	77
Figura 4.15 Punta extrusora o hot-end	78
Figura 4.16 Probando el firmware con Pronterface.....	79
Figura 4.17 Prueba de los motores del eje X y Y.....	80
Figura 4.18 Prueba de los motores del eje Z y el Extrusor.	80
Figura 4.19 Imprimiendo un diseño 3D.....	82
Figura 4.20 Pruebas fallidas 1.	83
Figura 4.21 Pruebas fallidas 2.	83

Figura 4.21 Objetos impresos 1.....	84
Figura 4.21 Objetos impresos 2.....	84

INTRODUCCIÓN

El presente proyecto tiene como finalidad el diseño de rutinas para el control de una impresora 3D, mediante el uso de una tarjeta Arduino y una tarjeta Raspberry Pi como ordenador la cual será configurada con todos los software necesarios para este proyecto, el cual será desarrollados en los siguientes capítulos.

Capítulo 1: Descripción General del Proyecto, en este capítulo estableceremos los antecedentes, motivación, objetivos justificación, descripción y limitaciones del proyecto.

Capítulo 2: Fundamento Teórico, en este capítulo hablaremos de los requerimientos para la aplicación del proyecto, es decir identificaremos todas las herramientas de hardware y software que necesitamos para el proyecto.

Capítulo 3: Diseño e Implementación del Proyecto, en este capítulo detallaremos las etapas para lograr nuestro objetivo principal, para esto realizaremos ejercicios de prueba fundamentales para el desarrollo del proyecto final.

Capítulo 4: Simulación y Pruebas del Proyecto, en este capítulo daremos a conocer de forma detallada todas las conexiones, simulaciones y conclusiones de cada uno de los ejercicios de prueba así como del proyecto final de graduación.

CAPÍTULO 1

DESCRIPCIÓN GENERAL DEL PROYECTO

1.1. Antecedentes

Las impresoras 3D surgieron de la idea de convertir en objetos reales los diseños 3D realizados en una computadora. En la actualidad se utilizan para la creación de prototipos o prefabricación de piezas en sectores como la arquitectura, el diseño industrial y la medicina en especial para la creación de prótesis médicas.

Hoy en día existen varios tipos de impresora 3D, por un lado están las de compactación, en las que una masa de polvo se compacta por estratos, dentro de esta categoría tenemos las impresoras 3D de laser en la cual el láser transfiere energía al polvo haciendo que se polimerice, después se sumerge la pieza en un líquido que hace que las zonas polimerizadas se solidifiquen y también tenemos las impresoras 3D de tinta que funcionan de manera que inyectan tinta aglomerante al polvo para compactarlo.

Las ventajas que tienen las impresoras 3D de tinta son que al inyectar tinta se pueden mezclar colores y su proceso es mucho más rápido y económico que las impresoras 3D de láser, la desventaja que tienen las impresoras 3D de tinta es que las piezas impresas son más frágiles que las piezas impresas por una impresora 3D de láser.

Por otro lado tenemos las impresoras 3D de inyección de polímeros, en las cuales el material se añade por capas y cuyo funcionamiento se basa en la inyección de resinas líquidas que son tratadas con luz ultravioleta haciendo que la pieza se solidifique. Su ventaja frente a las impresoras 3D de compactación de polvo es la gran precisión de los acabados de la impresión, en cuanto termina la impresión las piezas están listas para la manipulación, no hay tiempos de espera. [1]

1.2. Motivación

Lo que nos motivó a realizar este proyecto fue la idea de tener nuestra propia impresora 3D en casa para diseñar y construir nuestras propias piezas y este proyecto de graduación nos pareció la oportunidad perfecta para ampliar nuestros conocimientos en los campos de la electrónica y la programación de los microcontroladores.

Otro motivo que nos condujo a realizar este proyecto es su costo, debido a que esta impresora 3D es de Open Source es decir de Código Abierto, nos resulta considerablemente más barata que otras impresoras 3D comercial, ya que todo lo que necesitamos para construirla y hacerla funcionar como: documentos, software, firmware, etc., es libre y lo podemos conseguir y descargar gratuitamente en internet.

Lo que también nos motivó a realizar este proyecto de graduación, fue la idea de que a futuro podamos emprender nuestro propio negocio, basado en el diseño y construcción de objetos en 3D con un bajo costo ya que prácticamente podemos construir cualquier pieza de plástico ya se de nuestra propia impresora así como piezas o partes de otros equipos.

1.3. Objetivos

El objetivo principal de este proyecto de graduación es diseñar rutinas mediante la programación del microcontrolador del Arduino, para el control de la impresora 3D desde entorno de la minicomputadora Raspberry Pi, es decir lograr el control de los motores, extrusora y sensores de nuestra impresora para imprimir un diseño en 3D.

Investigar que necesitamos para poner en funcionamiento cada una de las tarjetas involucradas en este proyecto, tanto de la tarjeta Raspberry Pi como de la tarjeta Arduino Mega 2560 y así diseñar una interface de conexión para poder comunicar los dos dispositivos.

Instalar y configurar en la Raspberry Pi todas las herramientas de software necesarias que vamos a utilizar en este presente proyecto de graduación, pudiendo así tener todos los software para el control de la impresora 3D.

Aplicar en este proyecto nuestros conocimientos de electrónica y la programación de microcontroladores adquiridos en nuestra etapa como estudiante y así aportar al desarrollo tecnológico del país.

1.4. Justificación del Proyecto

Con el crecimiento de la industria ecuatoriana en los últimos años, podemos aprovechar esta tecnología de la impresión 3D, para construir objetos o piezas de equipos no solamente utilizadas por arquitectos, agencias de publicidad y diseñadores industriales sino que ahora también se puede aprovechar esta tecnología en el campo de la medicina como para fabricar prótesis médicas.

1.5. Descripción del Proyecto

Para realizar este proyecto de graduación lo primero que tenemos que hacer es configurar la Raspberry Pi para que el Arduino pueda ser reconocido de tal manera que estos dos dispositivos se comuniquen, para esto debemos identificar todas las herramientas de hardware y software que vamos a necesitar para conectar el Arduino a la Raspberry Pi que es el entorno donde vamos a trabajar en vez de una PC.

Después de configurar la Raspberry Pi procedemos a instalar el software de Arduino y su firmware "Sprinter", por lo que se procede a descargar la versión Linux de Arduino IDE para Raspberry Pi cuyo sistema operativo es Raspbian que es una distribución de Linux, el Arduino IDE es la ventana de trabajo donde vamos a calibrar el firmware que será cargado al controlador de Arduino, para que este lo interprete logrando así controlar los motores y sensores de la impresora 3D.

También procedemos a instalar el software de control "Pronterface" que es la interface grafica que nos permitirá interactuar con la impresora, otro programa que debemos tener es el software de generación del Gcode "Skeinforge", el cual se encarga de generar el "capeado" es decir transforma los diseños 3D es nuestro caso los archivos.STL a un formato llamado código G más conocido como Gcode.

Luego este código G es enviado desde la Raspberry Pi hacia el Arduino para que este lo interprete gracias al firmware que lleva en su interior, este código G contiene toda la información y todas las instrucciones para hacer mover los motores de la forma adecuada, también contiene la información acerca de las temperaturas y velocidades para que el diseño en 3D se imprima capa por capa. [2]

1.6. Limitaciones

Como limitaciones para el desarrollo nuestro proyecto podemos citar el hecho de trabajar con el sistema operativo Linux el cual no estamos muy familiarizados ya que la mayoría de los usuarios estamos acostumbrados a usar el sistema operativo Windows, y debido a que la tarjeta Raspberry Pi está basada en el sistema operativo Linux se nos hace un poco complejo a la hora de configurar e instalar los diferentes software necesarios para realizar nuestro proyecto.

En cuanto a la tarjeta Raspberry Pi podemos decir que es limitada en su disco duro ya que dependemos de la capacidad de una memoria microSD y esto nos puede traer ciertos inconvenientes a la hora de instalar software que sean muy pesados, por esta razón se recomienda usar una memoria microSD con una capacidad mayor a 8GB.

Otro hecho que podemos citar es que la Raspberry Pi solo tiene dos puertos USB disponibles, es decir que al conectar el teclado y el mouse ya habremos ocupado los dos puertos disponibles, por lo que se recomienda usar un HUB USB para tener más puertos disponibles de tal manera que podamos conectar el Arduino o otro dispositivo como un pendrive.

En cuanto al procesador de la Raspberry Pi tenemos que mencionar que es un poco lento y esto nos traerá inconvenientes a la hora de generar el Gcode de un diseño 3D.

Otra limitación es en cuanto al puerto serial que Raspberry Pi le asigna a Arduino para poder comunicarse, al parecer este puerto muchas veces se cae o se deshabilita lo que nos ocasiona que otras tarjetas Arduino no sean reconocidas por Raspberry Pi.

También podemos citar el hecho de que no podemos conectar la impresora directamente al Arduino sino que debemos tener una interface de conexión entre los motores y sensores de la impresora, ya que las rutinas programadas serán interpretadas por el Arduino para que este de las instrucciones precisas para mover los motores y hacer funcionar los sensores para que la extrusora imprima el diseño 3D capa por capa.

En cuanto la impresora muchas veces el objeto a imprimir se despega del plato de impresión, esto es debido muchas veces a la parte mecánica y a la nivelación de la impresora, por lo que debemos calibrar bien los ejes para obtener un buen acabado en el objeto a imprimir.

CAPÍTULO 2

FUNDAMENTO TEÓRICO

2.1. Requerimientos para aplicación del proyecto

Los requerimientos para desarrollar este proyecto se van a dividir en dos partes, por un lado tenemos las herramientas de software que son todos los programas que debemos instalar en la Raspberry Pi, entre estos están el sistema operativo Raspbian que es el software de inicio de Raspberry Pi, el Arduino IDE y su firmware, el software de control Pronterface y el que genera el Gcode que es Skeinforge.

Por otro lado tenemos las herramientas de hardware que son todos los componentes físicos que conforman este proyecto como lo son las tarjetas Raspberry Pi, Arduino y RAMPS 1.4, drivers pololu A4988 y otros elementos como teclado, mouse, HUB USB, cable HDMI, monitor, cargador de alimentación miniUSB, fuente de alimentación de 12Vdc y todos los componentes de la impresora como los motores y sensores.

2.2. Herramientas de software

Para poder acceder al entorno de Raspberry Pi usaremos el Sistema Operativo Raspbian cuyo sistema está basado en Linux, este sistema operativo puede ser descargado de la página web de Raspberry Pi en el siguiente enlace [4], existen otros sistemas operativos compatibles para esta tarjeta pero en nuestro caso usaremos el Raspbian.

Para poder acceder a internet y conectarnos a una red desde la minicomputadora vamos a necesitar configurar una dirección IP a nuestra Raspberry Pi, esto podemos hacerlo configurando manualmente una dirección IP estática.

Para instalar un programa en nuestra minicomputadora se recomienda siempre actualizar la base local del repositorio de Raspberry Pi antes de instalar cualquier programa, las instrucciones para instalar y configurar la Raspberry Pi las podemos encontrar en el Anexo A.

Para iniciar la programación en la plataforma de Arduino previamente vamos a necesitar dos software instalados en la Raspberry Pi, estos son el firmware Sprinter y el Arduino IDE que es la ventana donde se procederá a realizar la programación de los ejercicios de prueba y las rutinas para el control de la impresora 3D.

Otros software que vamos a necesitar en este proyecto es Pronterface que es el software de control de la impresora pero también se puede usar el Repetier Host, también vamos a necesitar el Skeinforge o Slic3r, estos software se encargan de generar el código G.

2.2.1. Firmware Sprinter y Arduino IDE

Para instalar el firmware en el microcontrolador, previamente tenemos que proceder a descargar el software de Arduino para Linux en el siguiente enlace [6], por lo general debemos descargar la última versión de Arduino IDE y así poder iniciar con la programación del microcontrolador.

Una vez que tengamos en nuestra Raspberry Pi el IDE de Arduino correctamente configurado procedemos a descargar el firmware del siguiente enlace [7], realizado esto ya podemos comenzar a programar en el Arduino, la ventana de trabajo donde se programaran las rutinas para el control de la impresora 3D se muestra en la figura 2.1, la instalación la podemos encontrar en el Anexo B. [5](Página 70)

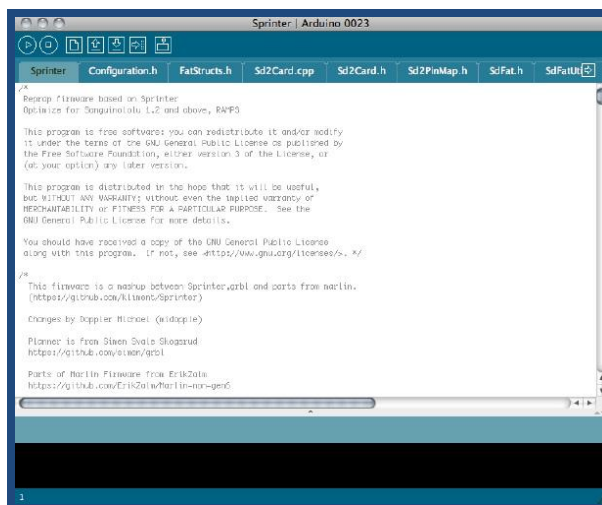


Figura 2.1 Arduino IDE con el firmware Sprinter. [5](Página 71)

2.2.2. Software de control (Pronterface)

Una vez que se instaló el firmware debemos instalar el software de control de la impresora, existen muchos software como el Pronterface, ReplicatorG, Repetier Host y otros.

En este proyecto usaremos el software de control “Pronterface” que es un programa que nos permite cargar un diseño en 3D en formato.STL para generar el Gcode del diseño que será interpretado por el firmware de Arduino. Este programa tiene un entorno gráfico que nos permite manipular los motores que hacen mover los ejes de la impresora, calentar la base y la punta del extrusor, monitorear la temperatura de ambos elementos, cambiar la velocidad de transmisión de datos, comenzar impresiones, pausarlas, reiniciarlas, etc.

La instalación de este programa la podemos encontrar en el Anexo C, una vez instalado el software de control abrimos el archivo.py, y al ejecutarlo nos muestra una ventana como la figura 2.2, [8] (Página 122).

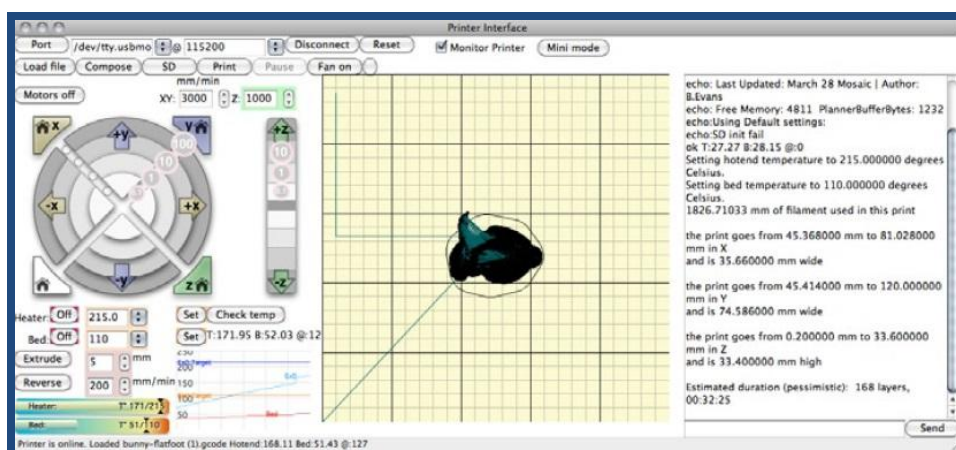


Figura 2.2 Software de control Pronterface. [5](Página 43)

2.2.3. Software de control (Repetier Host)

Repetier Host es uno de los nuevos software de control para impresoras 3D, las instrucciones para su instalación la podemos encontrar en el Anexo D, este programa abre los archivos del diseño 3D y llama a otro programa llamado Slic3r el cual genera el código G, dicho código contiene toda la información sobre la impresión, como las temperaturas, los tipos de relleno, velocidades, cantidad de perímetros, numero y altura de cada capa del objeto, diámetro del filamento, etc.

Repetier-Host tiene la capacidad de modificar el archivo de código G y ver los cambios en tiempo real desde el panel de vista previa, lo cual nos permite modificar, centrar, rotar y ajustar el objeto 3D antes de imprimir como podemos apreciar en la figura 2.3. [5](Página 45), [9], [10]

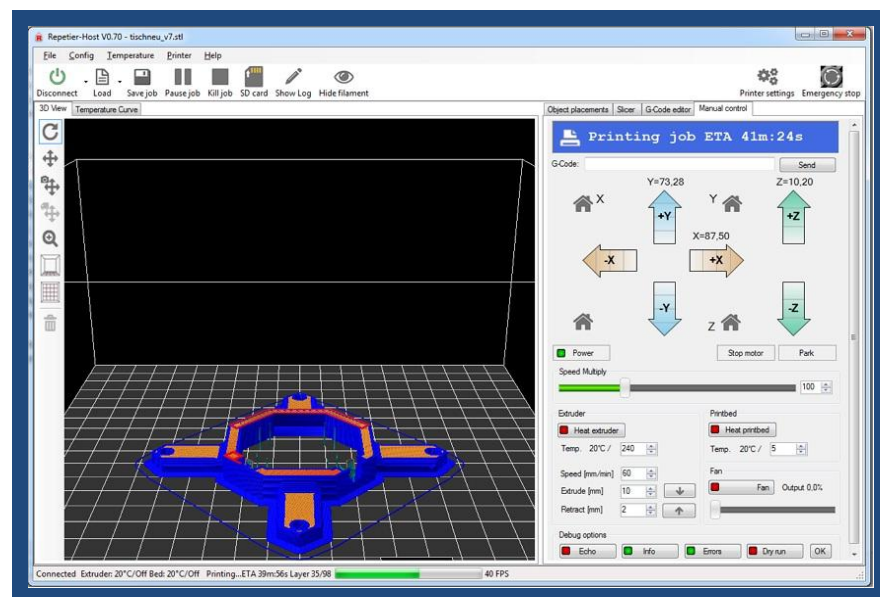


Figura 2.3 Software de control Repetier-Host. [11].

2.2.4. Software de generación de Gcode (Skeinforge)

Para generar el código G tenemos algunos software entre los más usados están el Skeinforge y el Slic3r, en nuestro proyecto usaremos el Skeinforge, este software utilizado para generar el Gcode no requiere de un proceso de instalación ya que viene incluido al instalar el software de control Pronterface.

Una vez hecho esto, será desde el software de control desde donde vamos a abrir Skeinforge para configurar sus parámetros. Para abrir Skeinforge debemos hacer clic en la pestaña “Settings/SlicingSettings” de la barra de herramientas por lo que veremos una ventana como la de la figura 2.4, [8](Página 124).

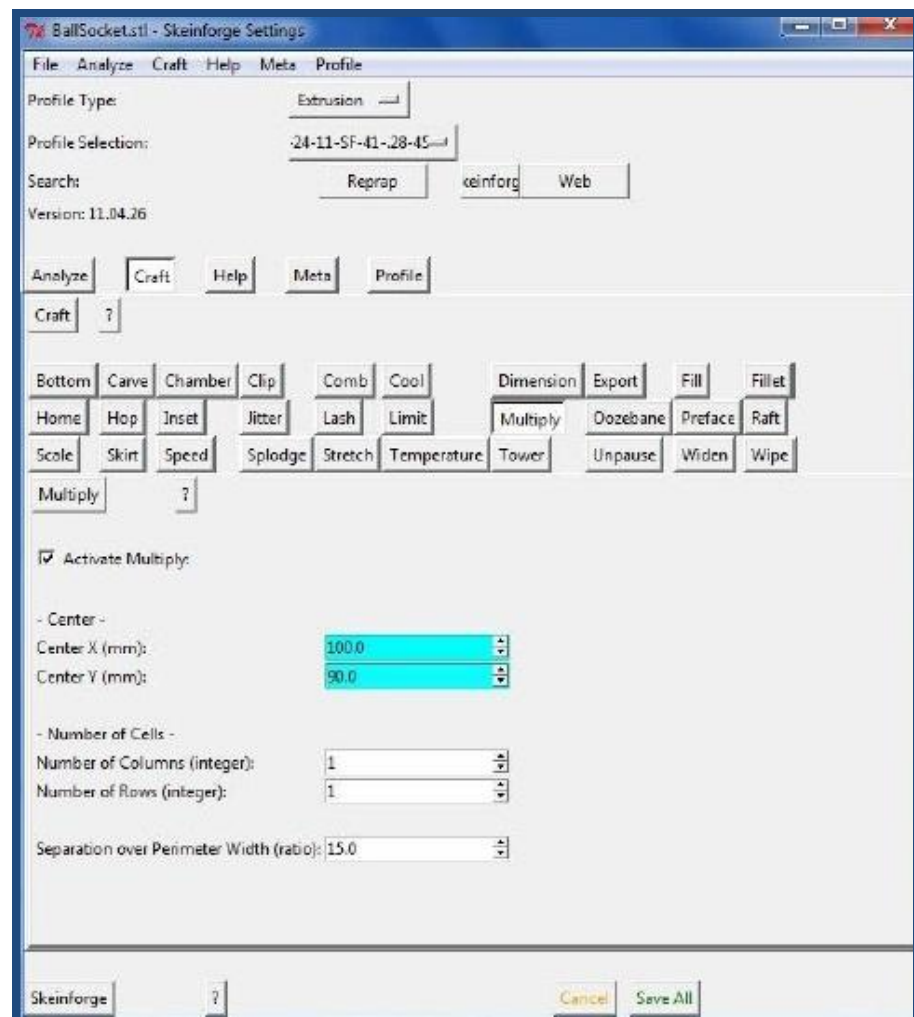


Figura 2.4 Calibración de Skeinforge. [8](Página 125)

2.2.5. Software de generación de Gcode (Slic3r)

El programa Slic3r es otro software de generación de Gcode, este programa toma el diseño 3D de formato.STL y genera el Gcode de dicho diseño 3D, este código contiene todas las instrucciones para controlar la impresora, es decir el movimiento de los motores, temperaturas, velocidades, etc.

A continuación en la figura 2.5 se aprecia la ventana de Slic3r, las instrucciones para su instalación y configuración las podemos de los parámetros la podemos encontrar en el Anexo E. [9]

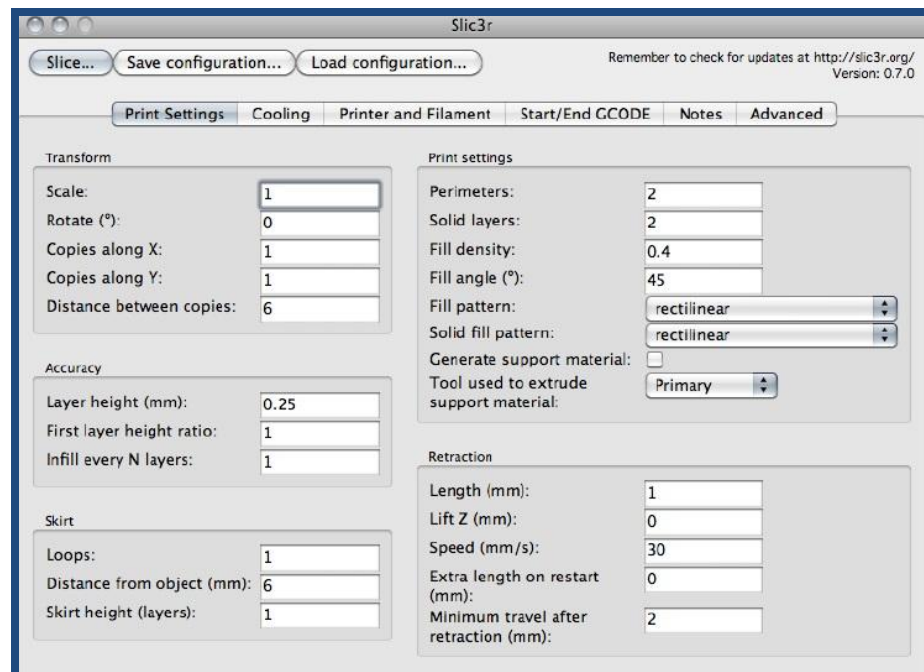


Figura 2.5 Calibración de Slic3r [5](Página 40).

2.3. Herramientas de hardware

La realización del proyecto se realizará en etapa de prototipo de tal manera que no necesariamente tendremos que implementar en circuito impreso, los elementos utilizados para la implementación del prototipo son los siguientes: tarjeta Arduino Mega 2560, tarjeta Raspberry Pi, protoboard, fuente ATX, kit de la impresora 3D Prusa Mendel, kit de la RAMPS 1.4, motores paso a paso, drivers para controlar los motores, otros dispositivos y periféricos.

2.3.1. Tarjeta Arduino Mega 2560

El Arduino Mega 2560 es una tarjeta de software libre que tiene incorporado un microcontrolador ATmega 2560, el cual puede ser programado mediante el lenguaje de programación de alto nivel Processing, tiene 54 entradas/salidas digitales de las cuales 14 proporcionan salida PWM y 16 entradas digitales usadas para distintas aplicaciones, tiene un cristal oscilador de 16MHz, conexión USB, conector para la alimentación, conector ICSP y botón de reset.

Contiene todo lo necesario para hacer funcionar el microcontrolador, simplemente al conectarlo al ordenador con el cable USB (en nuestro caso al Raspberry Pi). El Arduino Mega es compatible con la mayoría de shields diseñados para el Arduino Duemilanove o Diecimila que son otras versiones de este tipo de tarjetas. [12]

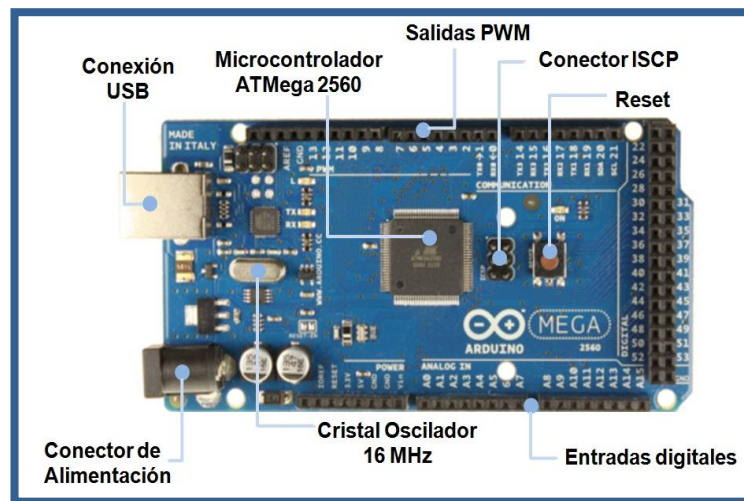


Figura 2.6 Arduino Mega 2560. [12]

2.3.2. Tarjeta Raspberry Pi

La Raspberry Pi es una minicomputadora del tamaño de una tarjeta de crédito como la que vemos en la figura 2.7, en la cual podemos conectar a un televisor, un teclado y un ratón para que funcione como una pequeña PC completamente funcional, que puede ser usado para muchas cosas que tu PC de escritorio realiza, como documentos de procesador de textos, hojas de cálculo, presentaciones, visualizar PDFs, navegar en la internet y ejecutar juegos con alta definición HD.



Figura 2.7 Raspberry Pi [13].

Existen dos modelos de estas tarjetas, el Raspberry Pi modelo A que fue el primer modelo el cual poseían 256MB de RAM y el Raspberry Pi modelo B que es el modelo actual, que cuenta ya con 512MB de RAM, no tiene disco duro por lo que la opción básica es una tarjeta SD de 4GB, pero puede funcionar normalmente hasta con una de 32GB.

Para ponerla en funcionamiento lo único que necesitamos es un cargador mini USB y un cable de red. Eso puede ser lo más básico, activando el acceso ssh puede acceder remotamente a la consola desde otro equipo. Si lo que quieres es montar un ordenador completo o un centro multimedia necesitas añadir a lo citado anteriormente un teclado y un ratón y conectarlo a una pantalla por HDMI (también existen adaptadores para VGA). [13]

2.3.2.1. El puerto GPIO de la Raspberry Pi

La Raspberry Pi posee una serie de pines de propósito general llamado Puerto GPIO, los cuales pueden ser utilizados como entradas o salidas digitales entre los cuales se incluye un pequeño puerto serial de comunicación, es decir por medio de este puerto la Raspberry Pi puede recibir o enviar datos desde y hacia otros dispositivos como en el caso del Arduino. [14]

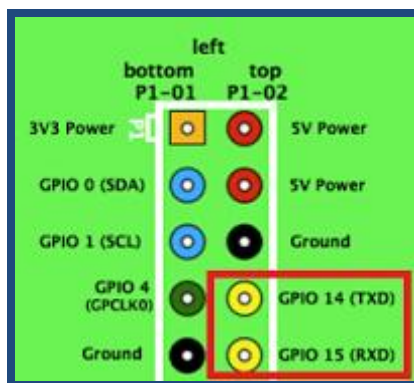


Figura 2.8 Puerto GPIO de la Raspberry Pi. [14]

2.3.3. Protoboard

El Protoboard es un tablero con orificios que tiene conexiones internas, en la cual se pueden insertar componentes electrónicos tales como: resistencias, diodos, transistores, leds, integrados y cables para armar circuitos experimentales llamados prototipos, en la figura 2.9 se muestra un circuito implementado en protoboard.

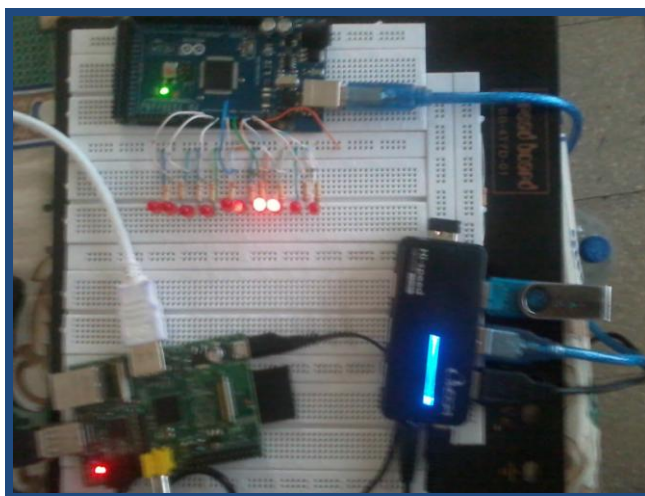


Figura 2.9 Implementación de un circuito en protoboard.

2.3.4. Fuente ATX

Una fuente ATX es un dispositivo interno del CPU de una computadora como el que se muestra en la figura 2.10, esta fuente convierte la corriente alterna en corriente continua y proporciona voltajes DC de 3.3V, 5V, -5V, 12V y -12V.

Esta fuente es idónea para este tipo de proyectos por su potencia y amperaje por lo general mayor a 4A, en nuestro proyecto usaremos 12V suficiente para alimentar los motores de la impresora.



Figura 2.10 Fuente ATX. [2]

Para poner la fuente ATX en funcionamiento, lo único que tenemos que hacer es un puente entre el cable verde y cualquiera de los cables negros como lo muestra la figura 2.11, de este modo si la conectamos a una toma de 110Vac el ventilador de la fuente deberá de encenderse y con un multímetro podremos verificar los voltajes que nos proporciona la fuente ATX.

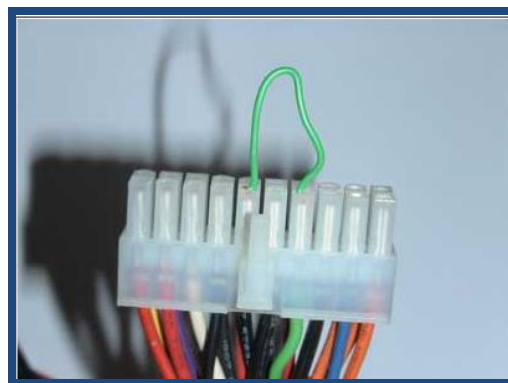


Figura 2.11 Puente entre el cable verde y negro. [2]

Los voltajes que proporciona la fuente ATX se muestran en la figura 2.12 según el esquema de colores, para verificar el voltaje que proporciona la fuente ATX, tenemos que medir el voltaje en cada terminal con un multímetro y comprobar sus valores. [2]

Pin	Señal	Color
1	3.3V	Orange
2	3.3V	Orange
3	Ground	Black
4	5V	Red
5	Ground	Black
6	5V	Red
7	Ground	Black
8	PW-OK	Grey
9	5VSB	Purple
10	12V	Yellow
11	3.3V	Brown
12	-12V	Blue
13	Ground	Black
14	PS-ON	Green
15	Ground	Black
16	Ground	Black
17	Ground	Black
18	-5V	White
19	5V	Red
20	5V	Red

Figura 2.12 Voltajes que proporciona la fuente ATX. [2]

2.3.5. Kit de la Impresora 3D Prusa Mendel

El kit de la impresora 3D Prusa Mendel es una impresora de hardware libre creada por la comunidad “RepRap”, esta impresora es proviene de la Mendel que fue la segunda impresora creada por RepRap. Esta comunidad tiene como objetivo diseñar máquinas que puedan autoreplicarse, es decir que tienen la capacidad de imprimir todas las piezas de plástico para construir otra impresora.

El kit viene con un manual de instrucciones y todos los componentes para armar la impresora tales como: tornillos, tuercas, varillas, rodamientos, muelles, correas dentadas, base de plástico para la impresión, rollo de plástico ABS, 5 motores de paso, 3 finales de carrera, una punta extrusora o hot-end y cables para las conexiones de la impresora, en la figura 2.13 vemos la impresora 3D armada. [15]



Figura 2.13 Impresora 3D Prusa Mendel armada.

2.3.6. RepRap Arduino Mega Pololu Shield (RAMPS)

La RepRap Arduino Mega Pololu Shield o también llamada RAMPS es la interfaz de conexión que usaremos para conectar el Arduino con los motores y sensores de la impresora 3D, en la figura 2.14 se muestran las características de la RAMPS 1.4 que es la versión que usaremos en nuestro proyecto de graduación.

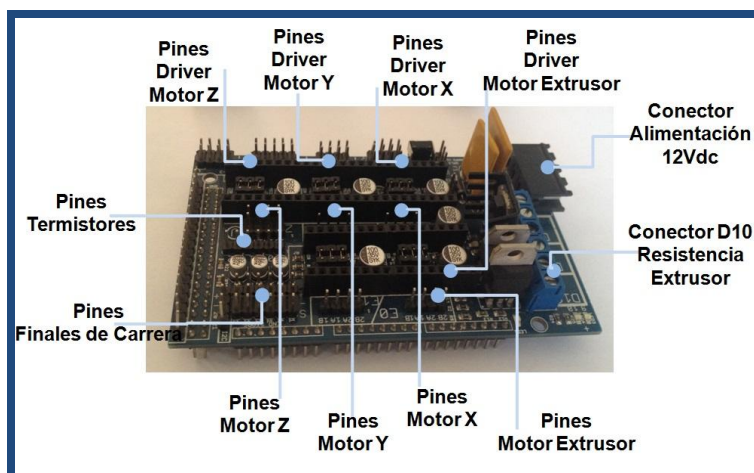


Figura 2.14 RAMPS 1.4. [5](Página 32)

Este kit electrónico RAMPS 1.4 tiene las siguientes características:

- Cuatro drivers A4988 pololu para el control de los motores paso a paso de los cuales tres son para los ejes y uno para la extrusora.
- Tres cargas conmutadas de alto poder de 5amps y 11amps para la extrusora y la cama de impresión.
- Seis conexiones de finales de carrera.
- Tres conexiones de termistores.
- Una entrada de fuente dual de 12V a 35V de hasta 16A.

A continuación en la figura 2.15 se muestra el esquema de conexiones de la RAMPS 1.4 con los motores y sensores de la impresora 3D, en la cual se muestra la conexión de los cinco motores de paso para los ejes de la impresora y para el motor que alimenta el material plástico a la extrusora, también se conectan tres finales de carrera uno para cada eje y también se conecta la punta extrusora o hot-end con su resistencia y termistor. [5](Página 30)

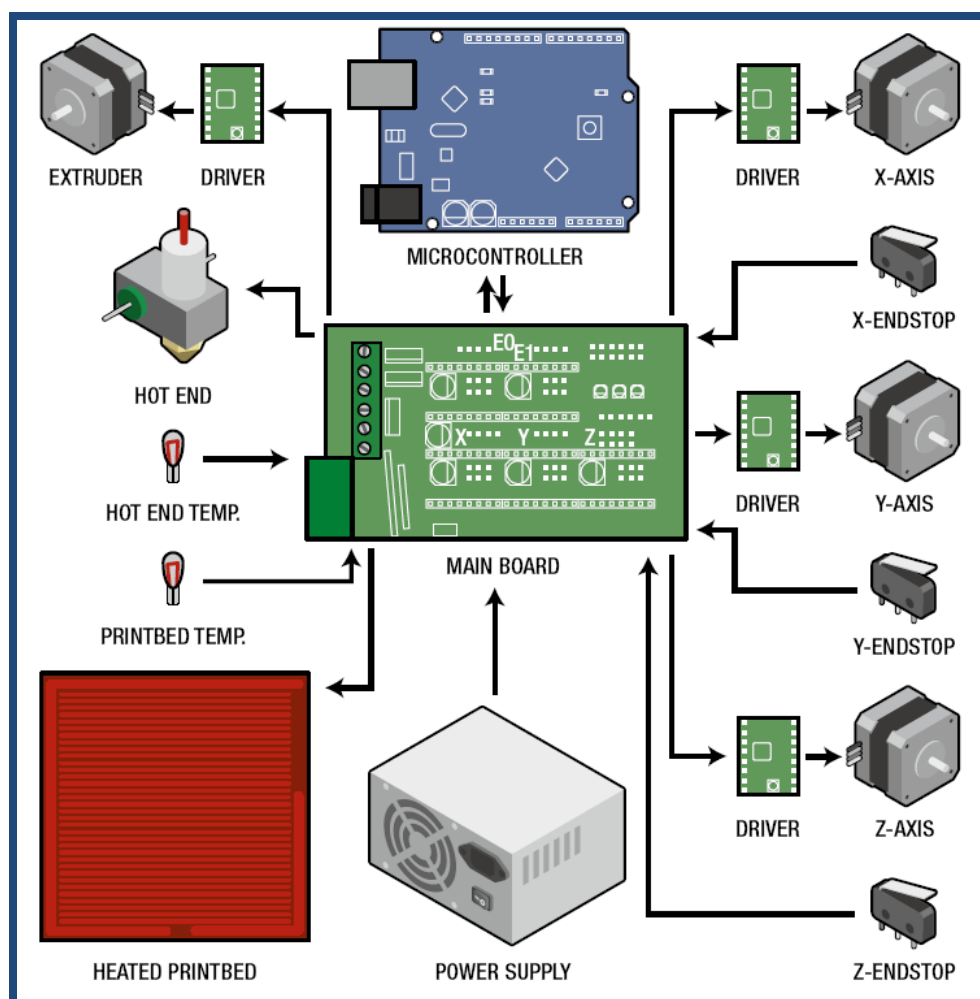


Figura 2.15 Conexión de la RAMPS 1.4. [5](Página 30)

2.3.7. Motores paso a paso

Los motores paso a paso son muy requeridos para realizar movimientos muy precisos, se caracterizan por realizar movimientos de un paso a la vez por cada pulso que se le aplique, este paso puede variar desde 90° hasta 1.8° , por lo que para realizar una vuelta completa de 360° , se necesitarían 4 pasos para el caso de 90° y 200 pasos para el caso de 1.8° .

Estos motores están constituidos por un rotor sobre el cual van distintos imanes permanentes y un estator el posee un cierto número de bobinas excitadoras. Para el desarrollo de nuestro proyecto vamos a necesitar cinco motores, de los cuales cuatro serán para controlar los ejes y uno para alimentar el material plástico a la extrusora. [16]



Figura 2.16 Motor paso a paso. [16]

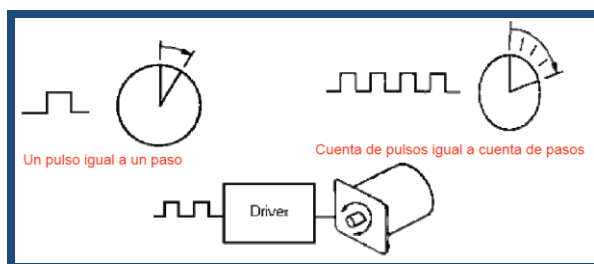


Figura 2.17 Relación entre el pulso y los pasos del motor. [16]

Este tipo de motores son conocidos como, motores paso a paso del tipo imán permanente y se clasifican en: Motores paso a paso Bipolares y Motores paso a paso Unipolares.

2.3.7.1. Motores paso a paso Bipolares

Este tipo de motores como se muestra en la figura 2.18, tienen generalmente 4 cables de salida, la secuencia para ser controlados es un poco compleja debido a que requieren del cambio de dirección del flujo de corriente a través de las bobinas para realizar un movimiento.

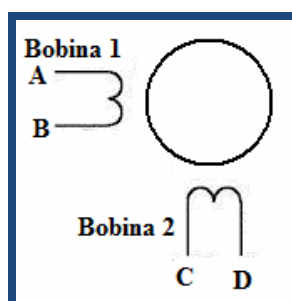


Figura 2.18 Motor paso a paso Bipolar. [16]

En la figura 2.19 se muestra un ejemplo del control de este tipo de motores usando un puente H (H-Bridge), como se puede apreciar se requiere de un puente H por cada bobina del motor, para esto se recomienda usar los drivers L293 el cual tiene dos puentes H en su interior. [16]

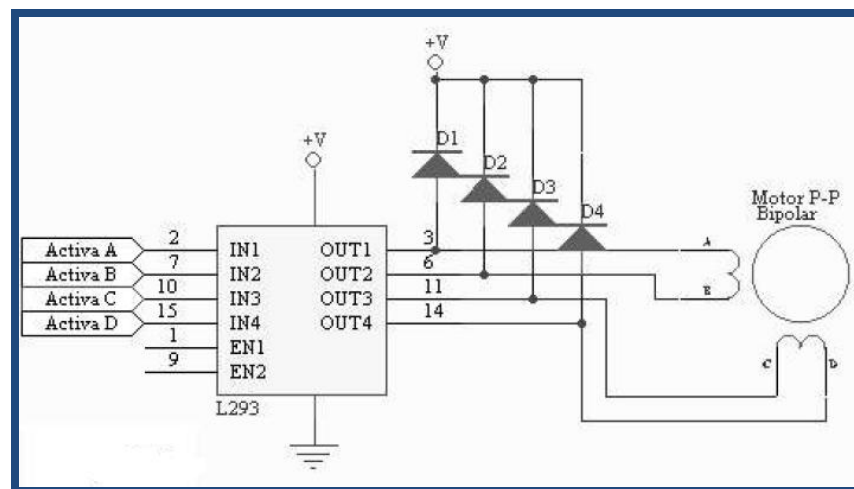


Figura 2.19 Control de un motor paso a paso bipolar mediante el uso del driver L293. [16]

2.3.7.2. Motores paso a paso Unipolares

Este tipo de motores como lo muestran las figuras 2.20 y 2.21, tienen generalmente 5 o 6 cables de salida esto depende de su conexión interna, este tipo de motor se caracteriza por ser más simple de controlar. [16]

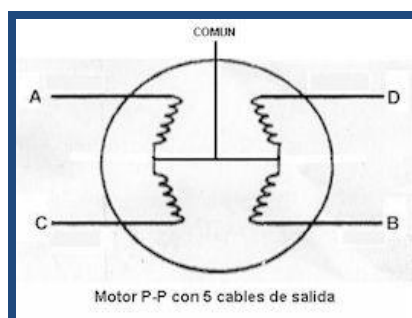


Figura 2.20 Motor paso a paso Unipolar de 5 cables de salida.

[16]

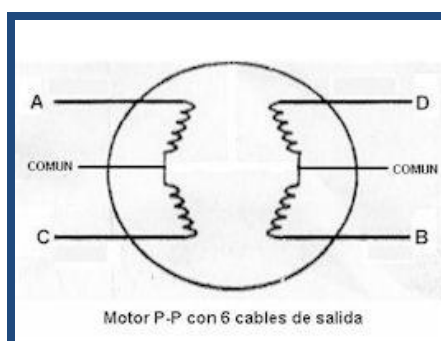


Figura 2.21 Motor paso a paso Unipolar con 6 cables de salida

[16]

En la figura 2.22 se puede apreciar un ejemplo del control de un motor paso a paso unipolar mediante el uso del driver ULN2803, cuya conexión interna es un arreglo de 8 transistores tipo Darlington, capaces de manejar cargas de hasta 500mA. Las entradas Activa A, B, C y D pueden ser activadas por medio de un microcontrolador. [16]

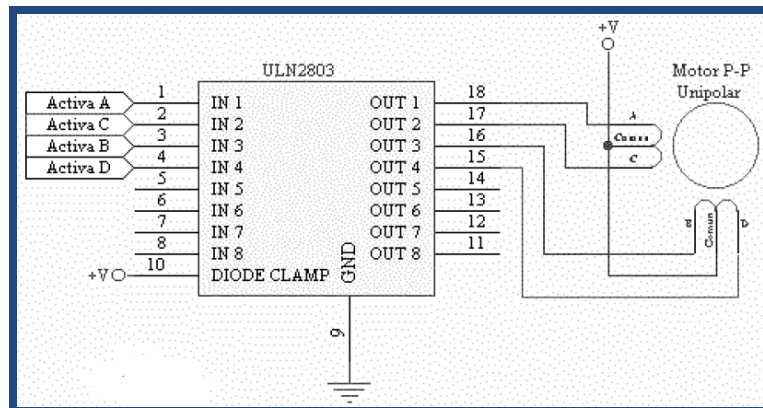


Figura 2.22 Control de un motor paso a paso unipolar mediante el uso del driver ULN2803. [16]

2.3.7.3. Secuencias para controlar motores paso a paso bipolares

Este tipo de motores requieren de la inversión de la corriente que circula a través de sus bobinas en una secuencia determinada para ser controlados, cada vez que se invierte la polaridad el rotor gira un paso y su sentido de giro depende de la secuencia como lo muestra la figura 2.23.

PASO	TERMINALES			
	A	B	C	D
1	+V	-V	+V	-V
2	+V	-V	-V	+V
3	-V	+V	-V	+V
4	-V	+V	+V	-V

Figura 2.23 Secuencia para controlar un motor paso a paso Bipolar. [16]

2.3.7.4. Secuencias para controlar motores paso a paso unipolares

Para manejar este tipo de motores existen tres métodos que dependen según la secuencia de encendido de sus bobinas.

Secuencia Normal: Esta secuencia es la más usada y recomendada, en esta secuencia el motor avanza un paso por vez y debido a que siempre hay al menos dos bobinas activadas como lo muestra la figura 2.24, se logra un alto torque de paso y retención.

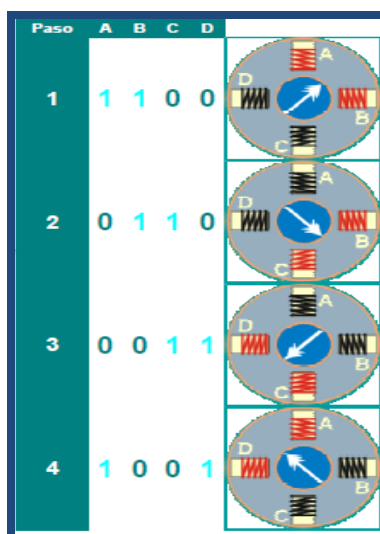


Figura 2.24 Secuencia normal. [16]

Secuencia del tipo wave drive: Esta secuencia de paso es la más simple y consiste en activar las bobinas una por una y por separado, con esta secuencia no se obtiene mucha fuerza, debido a que, solo es una bobina cada vez la que arrastra y sujeta el rotor del eje del motor como se puede apreciar en la figura 2.25. [16]

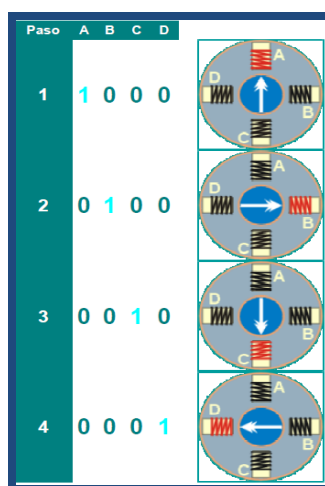


Figura 2.25 Secuencia del tipo wave drive. [16]

Secuencia del tipo paso medio: En esta secuencia se activan las bobinas de tal forma que dan un movimiento igual a la mitad del paso real y consiste en activar primero dos bobinas y luego una y así sucesivamente, como lo muestra la figura 2.26. [16]

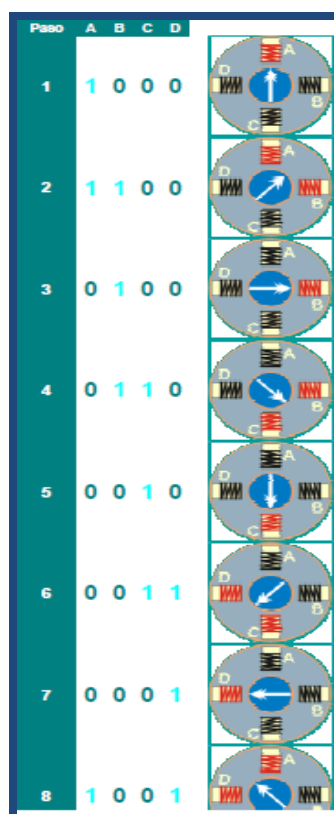


Figura 2.26 Secuencia del tipo medio paso. [16]

2.3.7.5. Identificación de los terminales de un motor paso a paso

Motor con 4 cables de salida: Para identificar los terminales de este tipo de motor cogemos un multímetro y medimos la resistencia entre un terminal y otro, si el resultado nos da infinito entonces los terminales son de diferentes bobinas y si no entonces los terminales pertenecen a la misma bobina, como se puede apreciar en la figura 2.27.

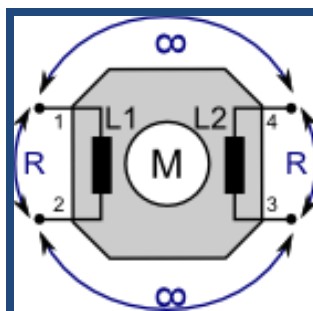


Figura 2.27 Identificación de un motor de 4 terminales. [17]

Motor con 5 cables de salida: Como podemos apreciar en la figura 2.28 se han unido los terminales intermedios de las dos bobinas, por lo que es sencillo determinar cual terminal es el común y cuáles son los terminales de las bobinas. [17]

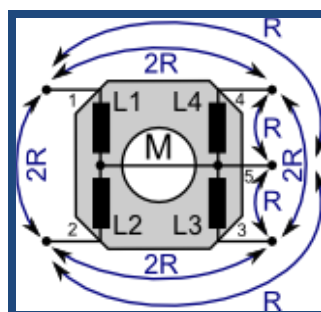


Figura 2.28 Identificación de un motor de 5 terminales. [17]

Motor con 6 cables de salida: Para identificar los terminales de este tipo de motor cogemos y medimos resistencia entre dos terminales, si el resultado es infinito entonces los terminales son de diferentes bobinas, si el resultado es R o $2R$ entonces los

terminales pertenecen a la misma bobina, para identificar el terminal intermedio el resultado debe ser R, como se puede apreciar en la figura 2.29. [17]

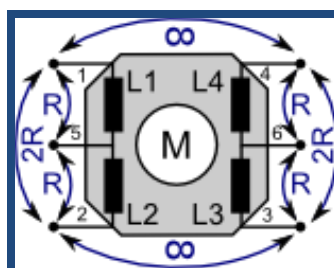


Figura 2.29 Identificación de un motor de 6 terminales. [17]

2.3.8. Drivers para controlar los motores de la impresora

Uno de los drivers que podemos usar para controlar los motores son los Chips A4988 de Pololu como el que se muestra en la figura 2.30, los Pololu son unos drivers electrónicos que se usan para controlar motores paso a paso, permiten decidir la dirección de giro y velocidad de estos, estos drivers llevan integrado un potenciómetro el cual regula el límite de corriente para los motores. [2]



Figura 2.30 Driver A4988 de Pololu. [2]

2.3.8.1. Conexiones del driver A4988

El Arduino controla los Chips A4988 de pololu mediante el uso de los pines de salidas digitales PWM, las entradas digitales de los chips que vamos a controlar con las salidas de Arduino son tres (STEP, DIR y RESET), cuando Arduino envíe un pulso a STEP el motor se moverá un paso, cuando DIR reciba una señal el motor se moverá en sentido de las manecillas del reloj, si no recibe una señal el motor se moverá en sentido contrario, para resetear el chip se usa el pin RESET.

A continuación en la figura 2.31 se muestra el esquema de conexiones de los chips A4988 de pololu, donde la alimentación del driver es de (3-5.5V) entre los pines VDD y GND, la alimentación del motor va de (8-35V) entre los pines VMOT y GND. [2]

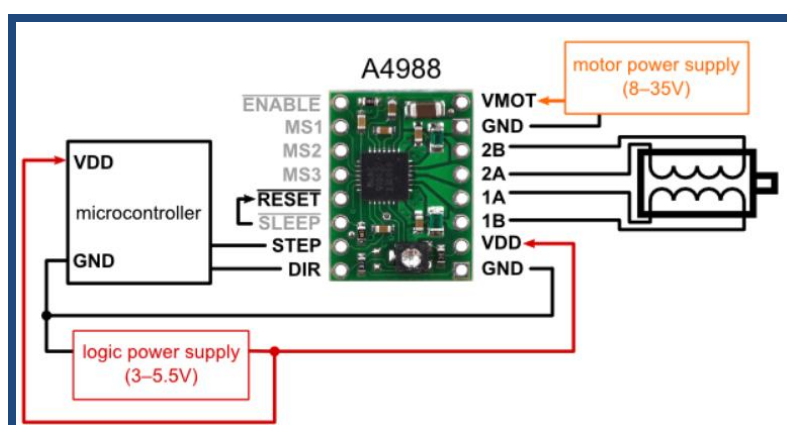


Figura 2.31 Esquema de conexión del driver A4988. [2]

2.3.9. Otros dispositivos y periféricos

Otros dispositivos y periféricos que usaremos para el desarrollo de nuestro proyecto son: HUB USB que es un dispositivo que nos permite tener más puertos USB disponibles ya que la Raspberry Pi solo tiene dos puertos disponibles, Cable convertidor HDMI a VGA el cual nos sirve para conectar un monitor VGA, tarjeta SD, teclado y mouse.

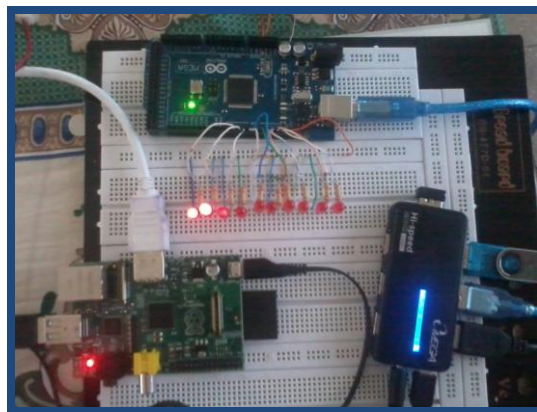


Figura 2.32 Dispositivos y periféricos.

CAPÍTULO 3

DISEÑO E IMPLEMENTACIÓN DEL PROYECTO

3.1. Introducción

En este capítulo detallaremos todo lo concerniente a cada una de las etapas para la consecución de nuestro objetivo el de diseñar e implementar rutinas para el control de una impresora 3D mediante Sistema Arduino y Raspberry Pi, Así como ejercicios y pruebas básicas de comunicación entre los dos dispositivos mencionados, control de leds con PWM así como también realizaremos el control de motores de paso mediante el driver L293D y RAMPS 1.4.

Para dar a conocer cada uno de los ejercicios procederemos a dar una pequeña introducción de cada uno de los mismos con su respectivo diagrama de bloques, así como también el diagrama de flujo y la descripción del algoritmo de cada ejercicio a realizar y para concluir agregamos los respectivos códigos de programación.

3.2. Ejercicio 1: Comunicación entre Raspberry Pi y Arduino vía puerto USB

Este ejercicio es fundamental y tiene como objetivo lograr comunicar las dos tarjetas mencionadas, la comunicación vía puerto USB entre los dos dispositivos es muy sencilla, por lo que antes de comunicar nuestra Raspberry Pi con cualquier otro dispositivo tenemos que modificar dos líneas de configuración en los siguientes archivos de RasPi.

Lo primero que debemos hacer es modificar dentro de nuestra RasPi el archivo `/boot/cmdline.txt` y eliminar la parte que dice:

```
console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
```

Luego dentro del archivo `/etc/inittab`, debemos comentar dos líneas de configuración, es decir añadir “#” delante de las dos líneas quedando de esta forma:

```
#Spawn a getty on Raspberry Pi serial line  
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Una vez que modificamos los dos archivos mencionados reiniciamos la RasPi, para luego elegir el modelo de la tarjeta Arduino y se nos habilitara la opción para elegir el puerto serial con el COM que nos asigno Raspberry Pi, con esto ya podemos verificar y compilar cualquier código de programación en Arduino. [14]

3.2.1. Diagrama de bloques del Ejercicio 1

En la figura 3.1 como podemos apreciar se muestra el diagrama de bloques correspondiente a la comunicación entre Raspberry Pi y Arduino.



Figura 3.1 Diagrama de bloques del Ejercicio 1. [14]

3.2.2. Diagrama de flujo del Ejercicio 1

A continuación se describe las instrucciones del algoritmo mediante un diagrama de flujo como se observa en la figura 3.2. [14]

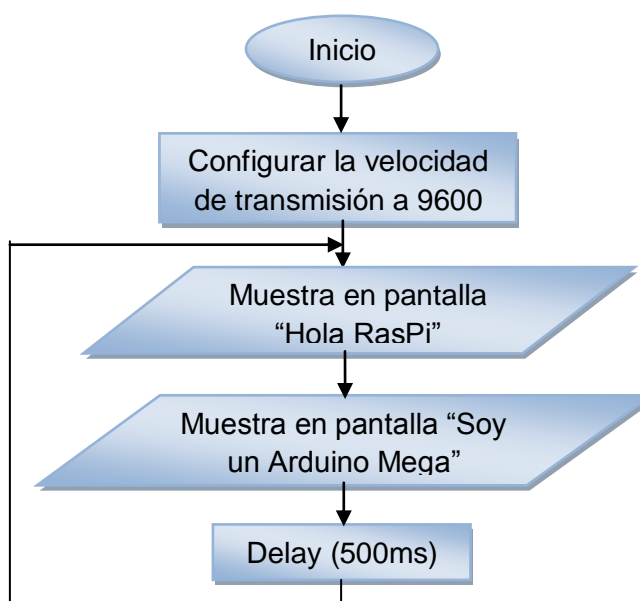


Figura 3.2 Diagrama de flujo del Ejercicio 1. [14]

3.2.3. Descripción del algoritmo del Ejercicio 1

A continuación se describe las instrucciones del algoritmo para la comunicación entre la RasPi y el Arduino. [14]

1. Se inicia la función setup().
2. Se configura la velocidad de transmisión a 9600bps.
3. Se inicia la función loop().
4. Se imprime en pantalla el mensaje "Hola RasPi!".
5. Se imprime en pantalla el mensaje "Soy un Arduino Mega".
6. Espera un retardo de 500ms.

3.2.4. Código fuente del Ejercicio 1

```
// Comunicación entre Raspberry Pi y Arduino vía puerto USB.
void setup() {
  Serial.begin(9600); // Se configura la velocidad de transmisión
                      // a 9600 bits por segundo
}
void loop() {
  Serial.write("Hola RasPi!\n"); // Se imprime "Hola RasPi!"
  Serial.write("Soy un Arduino Mega\n"); // Se imprime
                                         // "Soy un Arduino Mega"
  delay(500); // Espera un retardo de 500ms
}
```

3.3. Ejercicio 2: Control de leds con PWM

Este pequeño ejercicio tiene como objetivo comprobar el correcto funcionamiento de los pines de entrada y salida de Arduino, para esto realizaremos una secuencia básica de 3 leds la cual consiste en encender y apagar 3 leds colocados en las salidas PWM de Arduino.

A continuación en la figura 3.3 se muestra la técnica de modulación por ancho de pulso PWM para encender y apagar los leds en secuencia según lo programado en Arduino. [19](Página 8)

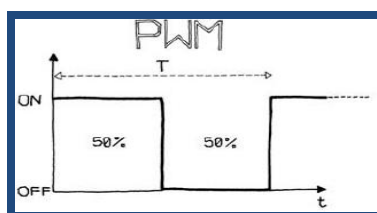


Figura 3.3 Modulación por Ancho de Pulso PWM. [19](Página 8)

3.3.1. Diagrama de bloques del Ejercicio 2

Mediante el siguiente diagrama de bloques podemos apreciar las etapas de conexión para el control de los leds con PWM como lo muestra la figura 3.4. [19](Página 8)



Figura 3.4 Diagrama de bloques del Ejercicio 2. [19](Página 8)

3.3.2. Diagrama de flujo del Ejercicio 2

En la figura 3.5 como podemos apreciar se muestra el diagrama de flujo que describe las instrucciones del algoritmo el cual controla los leds con la técnica PWM. [19](Página 8)

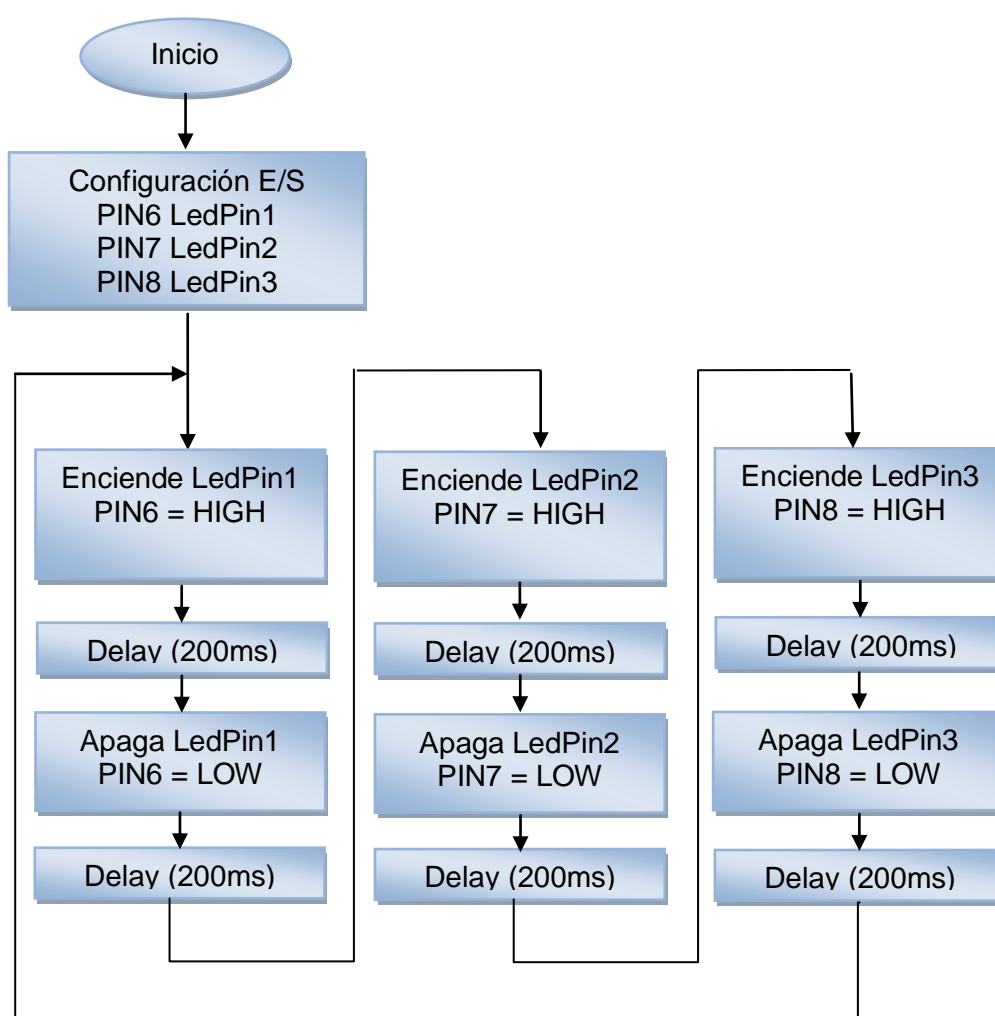


Figura 3.5 Diagrama de flujo del Ejercicio 2. [19](Página 8)

3.3.3. Descripción del algoritmo del Ejercicio 2

A continuación se describe las instrucciones del algoritmo para el control de los leds con PWM. [19](Página 8)

1. Se define las salidas de cada uno de los leds.
2. Se inicia la función setup().
3. Se declara las variables asignadas de los leds como salidas.
4. Se desactiva los leds.
5. Se inicia la función loop().
6. Se enciende y se apaga los leds con un retardo de 200ms.

3.3.4. Código fuente del Ejercicio 2

```
// Control de leds con PWM

// Encendido y apagado de 3 LED s

int ledPin1 = 6; // Define las salidas de los LED´s

int ledPin2 = 7;

int ledPin3 = 8;

void setup() { // Configura las SALIDAS

pinMode(ledPin1, OUTPUT); // declarar LEDs como SALIDAS
```

```
pinMode(ledPin2, OUTPUT);

pinMode(ledPin3, OUTPUT);

digitalWrite(ledPin1, LOW); // Apaga los LEDs

digitalWrite(ledPin2, LOW);

digitalWrite(ledPin3, LOW);

}

void loop(){ //Bucle de Funcionamiento

digitalWrite(ledPin1, HIGH); // Apaga y enciende los leds cada 200 ms

delay(200);

digitalWrite(ledPin1, LOW);

digitalWrite(ledPin2, HIGH);

delay(200);

digitalWrite(ledPin2, LOW);

digitalWrite(ledPin3, HIGH);

delay(200);

digitalWrite(ledPin3, LOW);

}
```

3.4. Ejercicio 3: Control de un motor paso a paso bipolar mediante el uso del driver L293D

Este ejercicio tiene como objetivo lograr controlar la velocidad y dirección de giro de un motor paso a paso con el driver L293D, mediante el uso de una rutina programada en Arduino y la circuitería para controlar un motor bipolar, con lo cual estaremos a un paso de lograr nuestro gran objetivo que es poder controlar los motores de los ejes X, Y, Z así como también el control del motor de la extrusora de nuestra impresora 3D. [18](Página 207)

3.4.1. Diagrama de bloques del Ejercicio 3

A continuación se muestra el diagrama de bloques de las etapas de conexión para el control de un motor paso a paso bipolar con el driver L293D, como se observa en la figura 3.6.

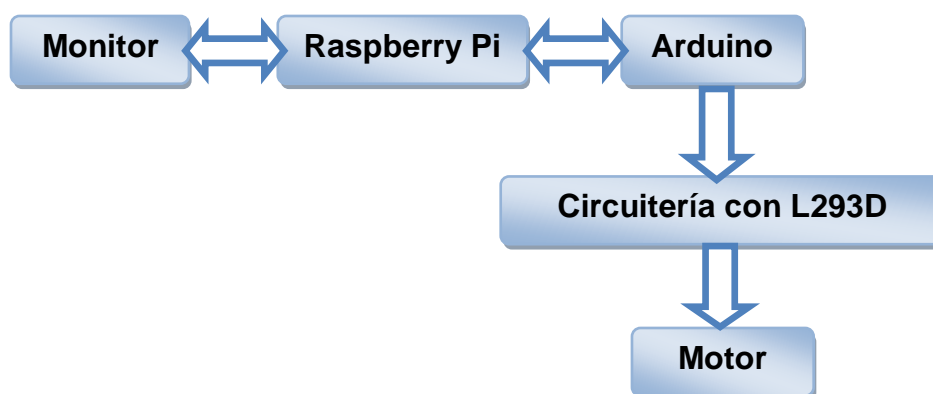


Figura 3.6 Diagrama de bloques del Ejercicio 3. [18](Página 207)

3.4.2. Diagrama de flujo del Ejercicio 3

En la figura 3.7 como podemos apreciar se muestra el diagrama de flujo que describe las instrucciones del algoritmo para controlar un motor paso a paso con el driver L293D. [18](Página 209)

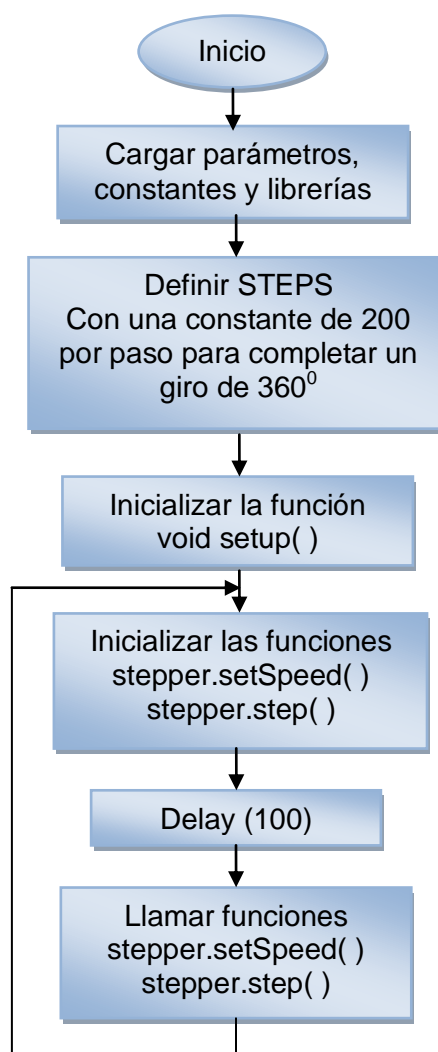


Figura 3.7 Diagrama de flujo del Ejercicio 3. [18](Página 209)

3.4.3. Descripción del algoritmo del Ejercicio 3

A continuación se describe las instrucciones del algoritmo para el control de un motor paso a paso mediante el driver L293D. [18](Página 209)

1. Se declara la librería stepper.h para iniciar la programación.
2. Se define STEPS 200 para completar una vuelta de 360° debido a que el motor es de 1.8° por paso.
3. Se declara los pines 4, 5, 6 y 7 de Arduino mediante la función stepper dentro de la librería stepper.h.
4. Se inician las funciones setup() y loop().
5. Se inicia la función setSpeed, la cual realiza las rpm que gira el motor.
6. Se inicia la función step, la cual realiza la cantidad de pasos que hará el motor hasta completar los grados de giro requeridos, el sentido del giro depende si la cantidad de pasos es positivo o negativo.
7. Se realiza un retardo 100ms para observar el cambio de giro del motor de un sentido al otro.

3.4.4. Código fuente del Ejercicio 3

```
// Control de un motor paso a paso con el driver L293D.

#include <stepper.h>

#define STEPS 200 // steps value is 360 / degree angle of motor

// create a stepper object on pins 4, 5, 6 and 7

stepper stepper(STEPS, 4, 5, 6, 7);

void setup()

{

}

void loop()

{

stepper.setSpeed(60);

stepper.step(200);

delay(100);

stepper.setSpeed(20);

stepper.step(-50);

delay(100);

}
```

3.5. Proyecto final de graduación: Diseño de rutinas empleando Sistema Arduino y minicomputadora Raspberry Pi para el control de impresoras 3D con aplicación práctica utilizando prototipo

En nuestro proyecto final el cual tiene como objetivo diseñar rutinas para el control de la impresora 3D usaremos la Raspberry Pi como ordenador, es decir desde nuestra RasPi controlaremos las rutinas programadas en Arduino mediante el software de control Pronterface el cual contiene el software Skeinforge que genera el código G al abrir un archivo.stl del diseño 3D a imprimir, para esto previamente se instaló el software de Arduino con su firmware.

Para una mejor visualización del diseño 3D podemos modificar los parámetros y dimensión del diseño 3D antes de mandar a imprimir el diseño, para esto usaremos el software Repetier-Host el cual contiene el Slic3r que es un software que también genera el código G.

También usaremos la Arduino Mega 2560 cuya tarjeta contiene un microcontrolador el cual será cargado con el código fuente de las rutinas programadas, para poder realizar el control de la impresora usaremos la RAMPS 1.4, que es una interfaz de conexión entre el Arduino, los motores paso a paso y los sensores de la impresora 3D. Para alimentar los motores usaremos una fuente ATX la cual nos proporciona 12Vdc con 4A de corriente suficiente para mover los motores de los ejes y la extrusora.

3.5.1. Diagrama de bloques del proyecto final

A continuación en la figura 3.8 se muestra el diagrama de bloques de la conexión de las etapas del proyecto final de graduación, como se puede apreciar la interface de conexión entre el Arduino, los motores, los sensores y extrusora de la impresora 3D, es la tarjeta RAMPS 1.4.

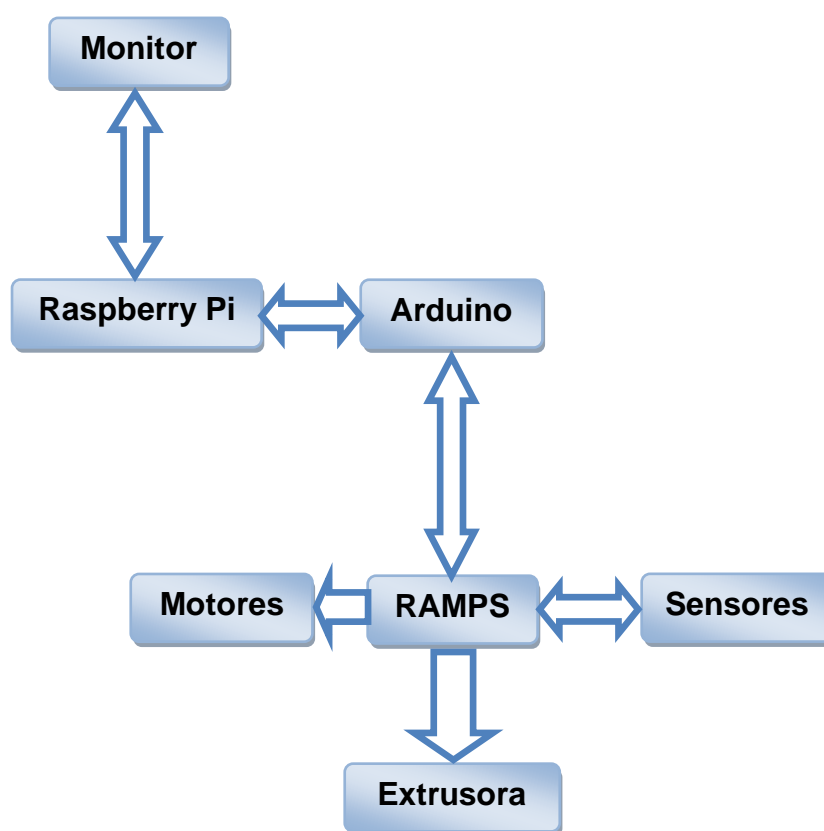
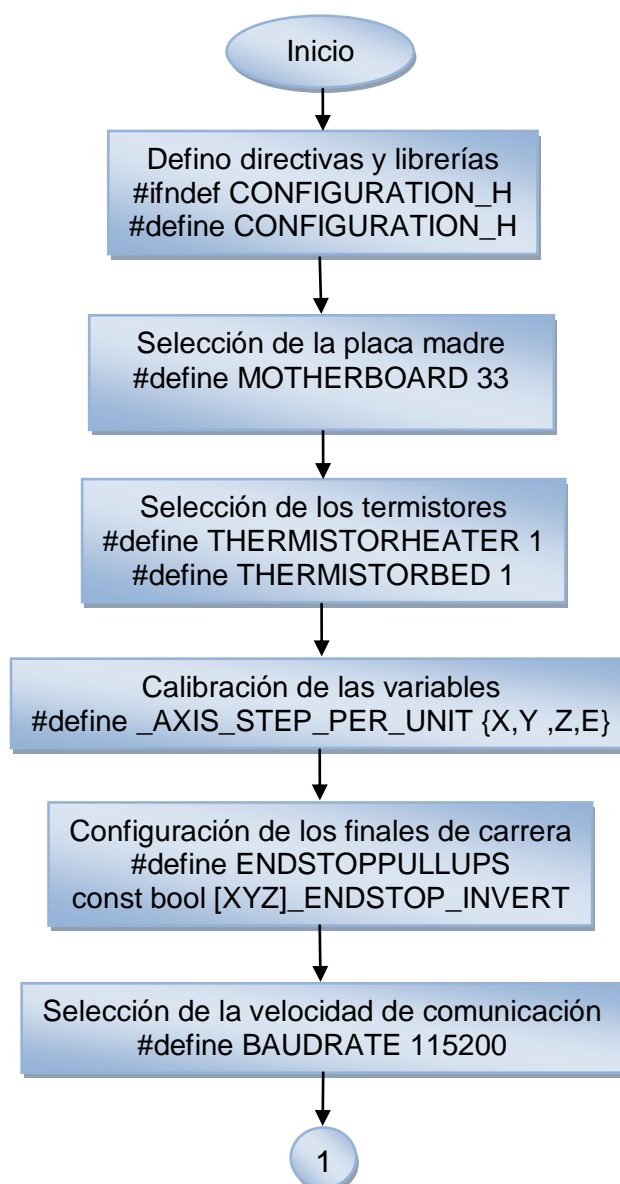


Figura 3.8 Diagrama de bloques del proyecto final.

3.5.2. Diagrama de flujo del proyecto final

En la figura 3.9 se muestra el diagrama de flujo del proyecto final, en este diagrama se describe las instrucciones principales del algoritmo para controlar los motores, extrusora y sensores de la impresora 3D.



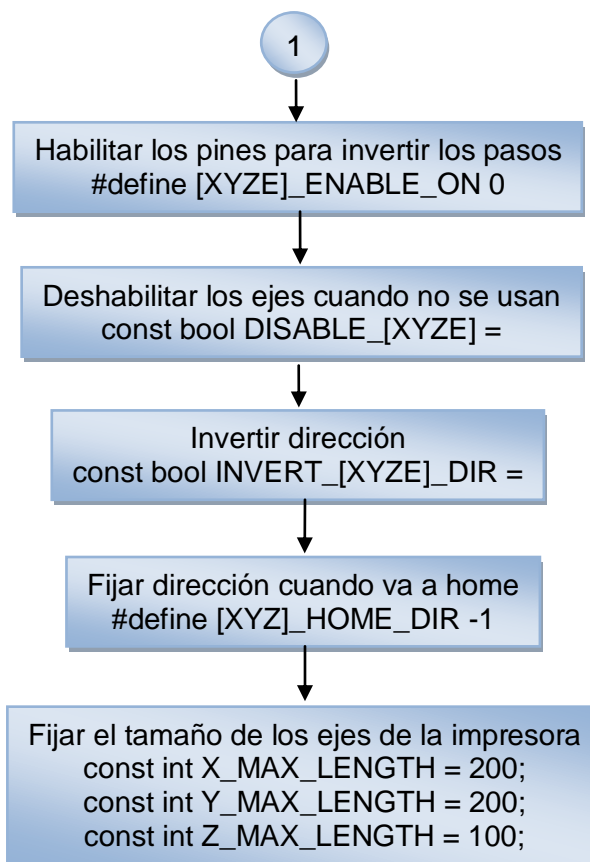


Figura 3.9 Diagrama de flujo del proyecto final.

3.5.3. Descripción del algoritmo del proyecto final

A continuación se describe las instrucciones del algoritmo para controlar los motores, sensores y extrusora de la impresora 3D.

1. Se declaran directivas y librerías.
2. Se selecciona la placa madre RAMPS que vamos a utilizar.

3. Se seleccionan los termistores que vamos a utilizar.
4. Se calibran las variables X, Y, Z, E.
5. Se configura los finales de carrera.
6. Se selecciona la velocidad de comunicación.
7. Se habilitan los pines para invertir los pasos.
8. Se deshabilita los ejes cuando no se usan.
9. Se invierte la dirección.
10. Se fija la dirección cuando se va a home.
11. Se fija el tamaño de los ejes de la impresora.

3.5.4. Código fuente del proyecto final

```
#ifndef CONFIGURATION_H

#define CONFIGURATION_H

// BASIC SETTINGS: select your board type, thermistor type, axis
scaling, and endstop configuration

// RAMPS 1.3/1.4 = 33

#define MOTHERBOARD 33

//// Thermistor settings:

#define THERMISTORHEATER 1
```

```
#define THERMISTORBED 1

//// Calibration variables

// X, Y, Z, E steps per unit - Metric Prusa Mendel with Wade extruder:

#define _AXIS_STEP_PER_UNIT {80, 80, 3200/1.25,700}

//// Endstop Settings

#define ENDSTOPPULLUPS // Comment this out (using // at the start
of the line) to disable the endstop pullup resistors

//If your axes move in one direction ONLY when the endstops are
triggered, set [XYZ]_ENDSTOP_INVERT to true here:

const bool X_ENDSTOP_INVERT = false;

const bool Y_ENDSTOP_INVERT = false;

const bool Z_ENDSTOP_INVERT = false;

// This determines the communication speed of the printer

#define BAUDRATE 115200

//#define BAUDRATE 250000

// For Inverting Stepper Enable Pins (Active Low) use 0, Non Inverting
(Active High) use 1

#define X_ENABLE_ON 0

#define Y_ENABLE_ON 0

#define Z_ENABLE_ON 0

#define E_ENABLE_ON 0
```

```
// Disables axis when it's not being used.

const bool DISABLE_X = false;

const bool DISABLE_Y = false;

const bool DISABLE_Z = true;

const bool DISABLE_E = false;

// Inverting axis direction

const bool INVERT_X_DIR = false;

const bool INVERT_Y_DIR = false;

const bool INVERT_Z_DIR = true;

const bool INVERT_E_DIR = false;

//// ENDSTOP SETTINGS:

// Sets direction of endstops when homing; 1=MAX, -1=MIN

#define X_HOME_DIR -1

#define Y_HOME_DIR -1

#define Z_HOME_DIR -1

// #define ENDSTOPS_ONLY_FOR_HOMING // If defined the endstops
// will only be used for homing

const bool min_software_endstops = false; // If true, axis won't move to
coordinates less than zero.
```

```
const bool max_software_endstops = true; //If true, axis won't move to
coordinates greater than the defined lengths below.
```

```
//Max Length for Prusa Mendel, check the ways of your axis and set
this Values
```

```
const int X_MAX_LENGTH = 200;
```

```
const int Y_MAX_LENGTH = 200;
```

```
const int Z_MAX_LENGTH = 100;
```

```
//// MOVEMENT SETTINGS
```

```
const int NUM_AXIS = 4; // The axis order in all axis related arrays is
X, Y, Z, E
```

```
#define _MAX_FEEDRATE {400, 400, 2, 45} // (mm/sec)
```

```
#define _HOMING_FEEDRATE {1500,1500,120} // (mm/min) !!
```

```
#define _AXIS_RELATIVE_MODES {false, false, false, false}
```

```
#define MAX_STEP_FREQUENCY 30000 // Max step frequency
```

CAPÍTULO 4

SIMULACIÓN Y PRUEBAS DEL PROYECTO

4.1. Introducción

En este capítulo procederemos a explicar de forma detallada la realización de las conexiones físicas y los elementos físicos utilizados tanto en los ejercicios de prueba como en nuestro proyecto final de graduación, también se mostrarán imágenes de la implementación en protoboard de cada uno de los ejercicios realizados, así como la simulación de los mismos.

4.2. Simulaciones y pruebas en protoboard

A continuación procederemos a realizar unos ejercicios de prueba, dando a conocer sus respectivas conexiones y mostrando las simulaciones de los mismos en protoboard.

4.2.1. Ejercicio 1: Comunicación entre Raspberry Pi y Arduino vía puerto USB

Existen dos formas para comunicar la Raspberry Pi con otros dispositivos, vía puerto USB y por el puerto serial GPIO, para ambos casos debemos modificar dos archivos de nuestra Raspberry Pi, en nuestro proyecto usaremos la forma más sencilla que es vía puerto USB.

Por otro lado si queremos utilizar el puerto GPIO de la RasPi para comunicarnos con cualquier otro dispositivo como Arduino, tendremos que realizar una interfaz de conexión para igualar los voltajes, ya que la tarjeta Arduino Mega trabaja con 5V mientras que el puerto GPIO de la RasPi trabaja con 3.3V, para esto podemos realizar el siguiente circuito como lo indica la figura 4.1. [14]

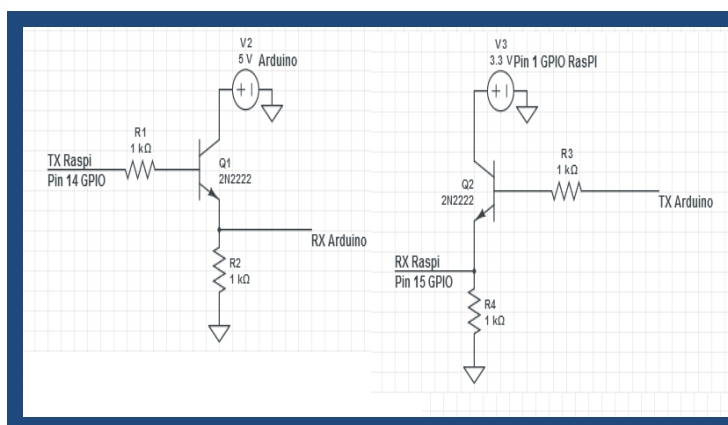


Figura 4.1 Igualando voltajes. [14]

4.2.1.1. Conexiones del Ejercicio 1

A continuación en la figura 4.2, se muestra la conexión de la Raspberry Pi con Arduino vía puerto USB, lo primero que tenemos que hacer es insertar la tarjeta SD a la RasPi previamente ya configurada, luego conectamos el cable convertidor HDMI a un monitor VGA para poder apreciar el escritorio de la Raspberry Pi.

También debemos conectar un HUB USB ya que la RasPi solo nos proporciona dos puertos USB, con esto tendremos más puertos disponibles para conectar otros dispositivos como: el teclado, mouse, pendrive y lógicamente el Arduino. Con todo esto solo nos queda conectar la alimentación de la Raspberry Pi y cargar el código de programación en nuestro Arduino. [14]

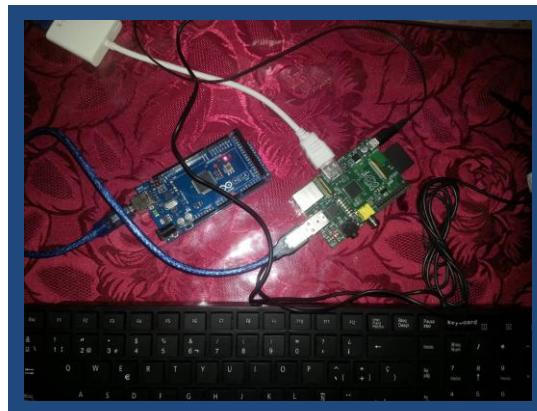


Figura 4.2 Comunicación entre Raspberry Pi y Arduino vía puerto USB.

4.2.1.2. Simulación del Ejercicio 1

En la figura 4.3 se puede apreciar la simulación del ejercicio 1, es decir la comunicación de Raspberry Pi con Arduino vía puerto USB, realizadas las conexiones correspondientes solo nos queda verificar y compilar el código de programación cerciorándonos de que no existe ningún error en el código y para poder apreciar lo programado y confirmar la comunicación entre los dos dispositivos le damos clic en la pestaña monitor serial de Arduino para apreciar el envío de los mensajes entre las dos tarjetas. [14]

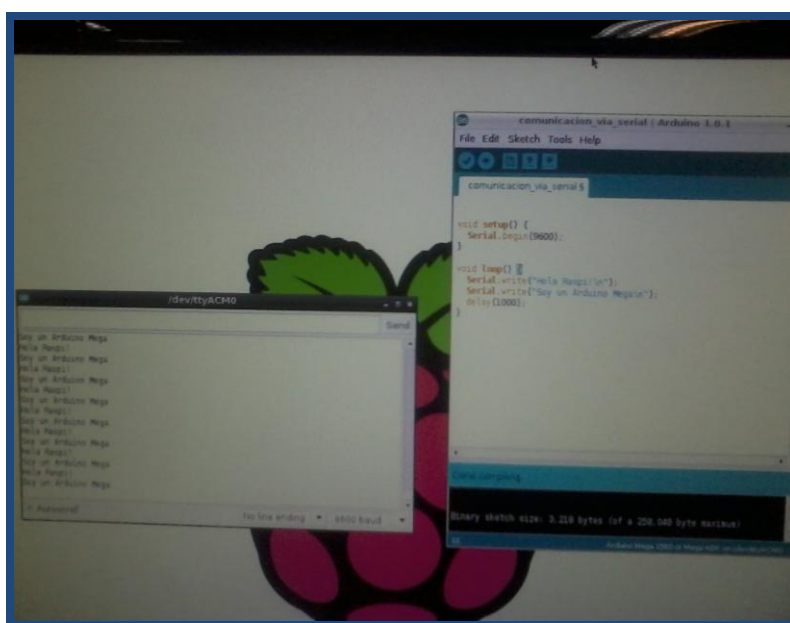


Figura 4.3 Envío de mensajes entre Raspberry Pi y Arduino.

4.2.1.3. Conclusiones del Ejercicio 1

Mediante este ejercicio se logró comprobar la comunicación entre los dos dispositivos, es decir se logró realizar la interfaz de conexión para que Raspberry Pi reconozca la tarjeta Arduino Mega, como nos pudimos dar cuenta al dar clic en la pestaña “Tools” se eligió el modelo de la tarjeta Arduino y se habilitó la opción “Serial Port” para elegir en COM dev/tty/ACM0 el cual fue asignado por la Raspberry Pi.

Se comprobó el envío de mensajes entre Raspberry Pi y Arduino Mega, al desplegar el Serial Monitor de Arduino notamos que se imprimen los mensajes en pantalla según el código programado con lo cual se pudo comprobar que la comunicación entre los dos dispositivos se realizó de una manera exitosa.

Como se esperaba el entorno de Raspberry Pi nos resultó un poco complejo debido a que su sistema operativo que está basado en Linux, la mayoría de los usuarios que trabajamos con el sistema operativo Windows no estamos acostumbrados a ejecutar comandos específicos para instalar un programa y muchas veces los comandos encontrados en internet no están correctos es decir muchas veces están mal escritos.

4.2.2. Ejercicio 2 : Control de Leds con PWM

Este ejercicio básicamente consiste en encender y apagar 3 leds con la técnica de modulación por ancho de pulso, al cargar el código de programación al Arduino se activaran las salidas PWM de Arduino según lo programado, para realizar la implementación de este ejercicio de prueba implementamos un pequeño circuito con 3 leds de colores amarillo, verde y rojo con 3 resistencias de 220 Ohm. [19](Página 8)

4.2.2.1. Conexiones del Ejercicio 2

En la figura 4.4 se muestran las conexiones para este ejercicio de prueba, en la cual los leds y resistencias están colocados a las salidas PWM 6, 7 y 8 del Arduino es decir en el PIN6, PIN7 y PIN8 respectivamente. El tiempo de encendido y apagado de cada led tiene un retardo de 200 ms, las variables asignadas a cada led son las siguientes LedPin1 (Led amarillo), LedPin2 (Led rojo) y LedPin3 (Led verde). [19](Página 8)

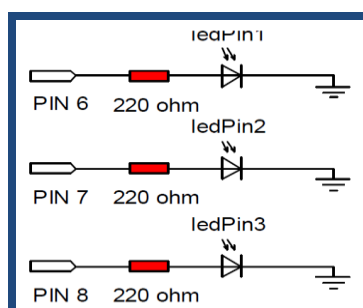


Figura 4.4 Esquema de conexión del Ejercicio 2. [19](Página 8)

4.2.2.2. Simulación del Ejercicio 2

A continuación como podemos apreciar en la figura 4.5 se muestra la simulación del ejercicio 2 en protoboard, en este circuito implementado con Raspberry Pi y Arduino tenemos 3 resistencias de 220ohm con 3 leds de colores amarillo, rojo y verde los cuales se encienden en secuencia con la técnica de Modulación por Ancho de Pulso PWM mediante la rutina programada en Arduino. [19](Página 8)

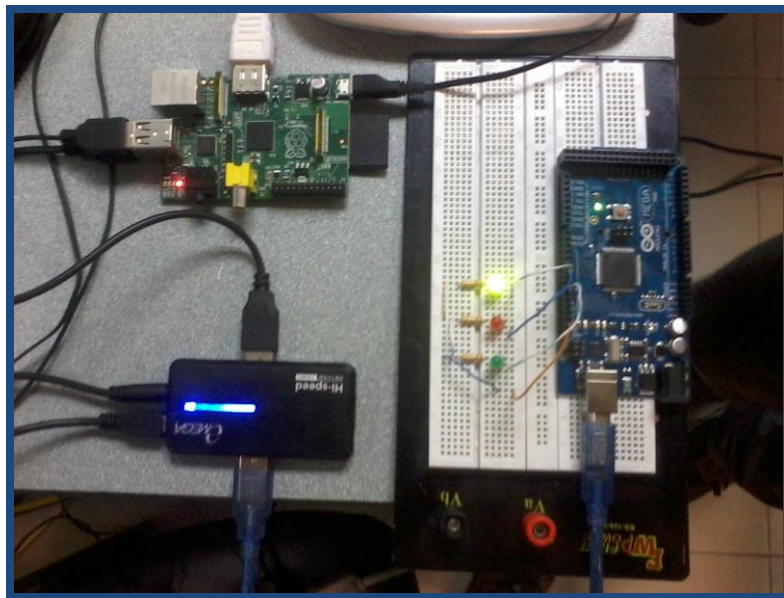


Figura 4.5 Simulación del Ejercicio 2.

4.2.2.3. Conclusiones del Ejercicio 2

Con este ejercicio se logró comprobar que efectivamente el Arduino genera señales PWM utilizando la capacidad de su microprocesador, esto se lo pudo realizar mediante el código de programación, en el cual al pin seleccionado se le asigna el valor HIGH o LOW mediante la función `digitalWrite()`, por lo que al cargar la rutina programada observamos como los leds se encienden y apagan en secuencia con un pequeño retardo según lo programado en Arduino.

El código de programación es sumamente sencillo, por lo que si deseamos utilizar las demás salidas digitales PWM solo tendremos que declararlas como salidas y asignar variables a los demás pines correspondientes.

La técnica de modulación por ancho de pulso es muy usada y Arduino nos proporciona 14 salidas digitales PWM y las podemos aprovechar para controlar la velocidad de los motores de paso como lo veremos en el siguiente ejercicio en el que se usara el driver L293D para controlar un motor paso a paso bipolar.

4.2.3. Ejercicio 3: Control de un motor de paso bipolar mediante el uso del driver L293D

Los motores bipolares tienen generalmente 4 cables de salida es decir tienen dos bobinas y para controlar simultáneamente su dirección de giro y su velocidad necesitaremos un circuito integrado o chip llamado de forma general puente H. Para este ejercicio de prueba usaremos el driver L293D, debido a que requiere de un puente H por cada bobina del motor para ser controlado, es decir usaremos dos puentes H como nos muestra la figura 4.6, la diferencia entre el L293D y el L293, es que el primero ya viene con diodos de protección para evitar los daños producidos por los picos de voltaje que puede producir el motor. [19](Página 66)

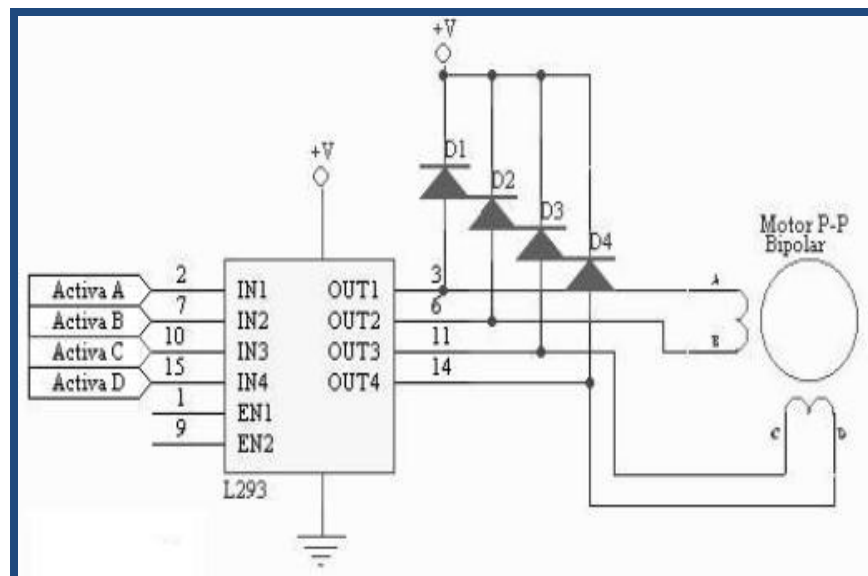


Figura 4.6 Control de un motor bipolar mediante el driver L293D. [16]

Los parámetros del motor bipolar que queremos controlar son la dirección de giro y la velocidad, la dirección se controla cambiando su polaridad es decir al cambiar la dirección del flujo de corriente a través de sus bobinas, mientras que la velocidad se controla mediante la técnica de modulación por ancho de pulso PWM, es decir al controlar el duty cycle tiempo en que el pulso esta activo ON.

A continuación en la figura 4.7 se muestra la relación entre la señal PWM y el voltaje efectivo, si el duty cycle es del 50% el voltaje efectivo será la mitad del voltaje total de entrada, mientras que si reducimos el duty cycle al 25% el voltaje efectivo será un cuarto del voltaje total de entrada lo que quiere decir que la velocidad del motor se reducirá, de esta forma si controlamos el duty cycle lograremos controlar la velocidad del motor. [19](Página 66)

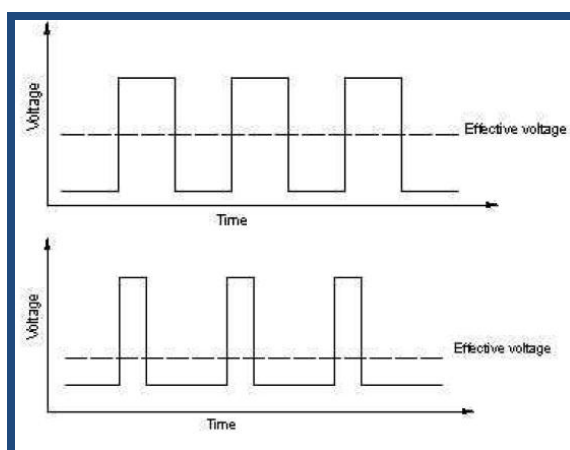


Figura 4.7 Relación de la señal PWM con el voltaje efectivo.

[19](Página 66)

4.2.3.1. Conexiones del Ejercicio 3

Los pines digitales 4, 5, 6 y 7 del Arduino van conectados a las entradas Input 1, Input 2, Input 3 e Input 4 del L293D respectivamente, mientras que las salidas Output 1, Output 2, Output 3 y Output 4 del driver van conectadas hacia los terminales de cada bobina del motor.

Por otro lado el pin 5V del Arduino va conectado a los pines 1, 9 y 16 del controlador, mientras que el pin Vin es conectado al pin Vc es decir al pin 8 del controlador. En el caso de usar un motor unipolar debemos conectar el común a Vin como se indica en la figura 4.8. [18](Página 209)

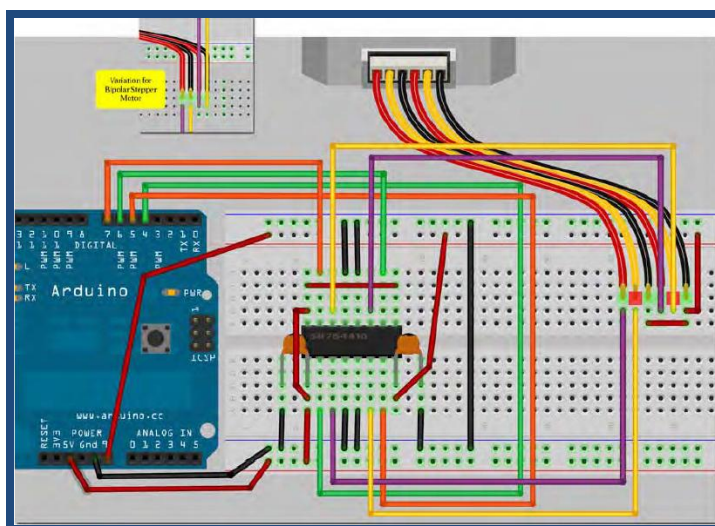


Figura 4.8 Esquema de conexiones del motor con el driver L293D. [18](Página 208)

4.2.3.2. Simulación del Ejercicio 3

A continuación en la figura 4.9 se muestra la implementación del ejercicio de prueba 3 en protoboard, el cual consiste en controlar simultáneamente la dirección de giro y velocidad de un motor paso a paso bipolar mediante el driver L293D, al cargar la rutina de programación en Arduino como el motor gira en un sentido hasta completar los pasos requeridos con una cierta velocidad y luego cambia la dirección de giro con otra velocidad diferente.

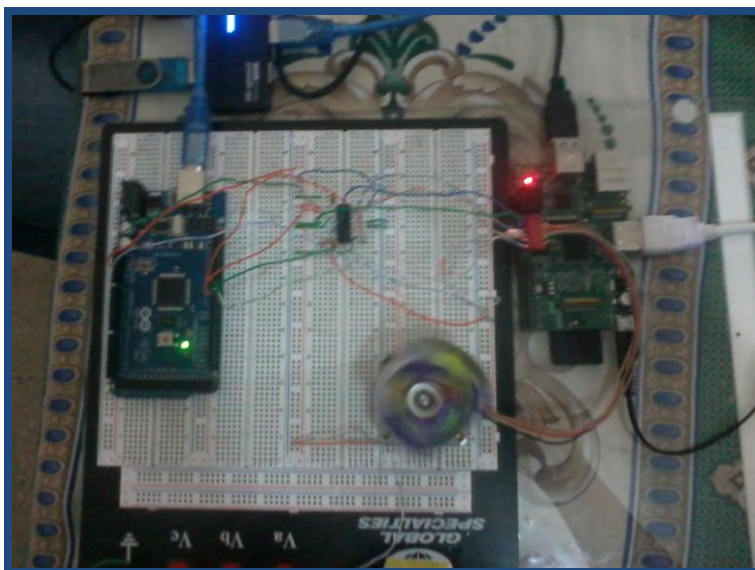


Figura 4.9 Simulación del Ejercicio 3.

4.2.3.3. Conclusiones del Ejercicio 3

Con este ejercicio de prueba que consistió en controlar la velocidad y la dirección de giro de un motor paso a paso bipolar con el driver L293D mediante una rutina programada en Arduino estamos muy cerca de lograr nuestro gran objetivo que es controlar los motores tanto de la extrusora como los de los ejes de la impresora 3D.

Los driver L293D o más conocidos como puentes H, son muy requeridos para controlar este tipo de motores, ya sean estos unipolares como bipolares como se aprecia en el esquema de conexiones, debido a que estos motores tienen dos bobinas en su interior por lo cual se requieren de dos puentes H uno por cada bobina del motor para ser controlados.

Se logro aprovechar dos funciones que nos proporciona la librería stepper de Arduino, la función `setSpeed()` para controlar la velocidad del motor según el parámetro ingresado y `step()` para controlar la dirección de giro el cual depende si el parámetro ingresado es positivo o negativo.

4.3. Proyecto final de graduación: Diseño de rutinas empleando Sistema Arduino y minicomputadora Raspberry Pi para el control de impresoras 3D con aplicación práctica utilizando prototipo

A continuación adaptaremos cada uno de los ejercicios realizados anteriormente al proyecto final de graduación, desde la comunicación entre los dos dispositivos, la utilización de las salidas PWM de Arduino y el control de la dirección de giro y la velocidad de un motor de paso con la RAMPS 1.4 la cual como explicamos anteriormente es la interfaz de conexión de Arduino con los motores y sensores de la impresora 3D.

Antes de realizar la conexión de la RAMPS 1.4 directamente con la impresora 3D procederemos a realizar unas pruebas iniciales para verificar el correcto funcionamiento del firmware y el software de control, para esto utilizaremos los motores paso a paso que usamos en los ejercicios de pruebas anteriores.

El objetivo de realizar estas pruebas es calibrar el firmware “Sprinter” y el software de control “Pronterface”, una vez que calibremos el firmware procederemos mediante los controles de Pronterface a comprobar el funcionamiento de los motores de cada uno de los ejes, así como el de la extrusora, el cual alimentara con el material plástico para que este sea extruido por medio de la punta extrusora o hot-end, también aprovecharemos este software para comprobar el funcionamiento de los finales de carrera.

4.3.1. Calibración del firmware Sprinter

- Lo primero que se hace es elegir la placa madre que estamos utilizando, dentro del código del firmware debemos habilitar la opción que dice `#define MOTHERBOARD 33`, ya que estamos utilizando la RAMPS 1.4, cuya opción es la misma si utilizamos la RAMPS 1.3.
- Luego elegimos los termistores que estamos usando, es decir el del hot-end y el de la base caliente si es que la llegamos a usar, para este proyecto final, habilitamos las opciones, `#define THERMISTORHEATER 1` y `#define THERMISTORBED 1`.
- Configuramos las variables de calibración, esta línea de código es para indicar los pasos que dan los motores de cada uno de los ejes y el de la extrusora, para lo cual habilitamos la opción que dice, `float axis_steps_per_unit[] = {80, 80, 3200/1.25,700}`.
- Configuramos los finales de carrera, en el caso de que los motores solo giren en una sola dirección, debemos ajustar los finales de carrera, para esto debemos cambiar las opciones que inicialmente están en false a, `[XYZ]_ENDSTOP_INVERT = true`.
- Por último elegimos la velocidad de comunicación de la impresora, para esto habilitamos la opción `#define BAUDRATE 115200`. [10]

4.3.2. Conexión de la RAMPS 1.4 con la impresora 3D

A continuación en la figura 4.10 se muestra la conexión de la RAMPS 1.4 con la impresora 3D, para esto debemos colocar y ajustar los drivers correctamente, luego debemos conectar los motores paso a paso, los finales de carrera y conectar la punta extrusora.

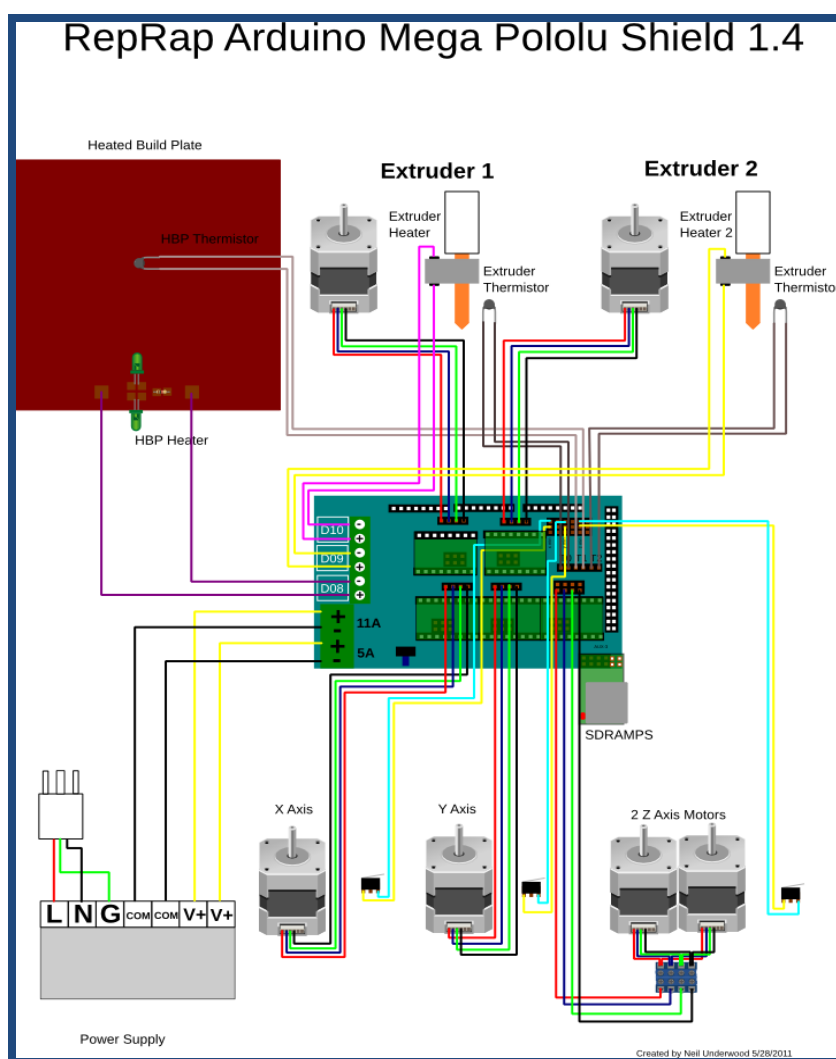


Figura 4.10 Conexión de la RAMPS 1.4 con la impresora 3D.

4.3.2.1. Conexión de los drivers A4988

En este proyecto de graduación usaremos 4 drivers A4988 para controlar los motores de los ejes X, Y, Z y también la extrusora, estos drivers deben ir colocados como lo indica la figura 4.11, como se puede apreciar tienen dos tornillos de ajuste, que sirven para regular el límite de corriente que le suministramos a los motores, para ajustar los drivers debemos girar el potenciómetro de la parte superior, debemos dar una vuelta completa en sentido contrario de las manecillas del reloj y luego debemos girar 1/4 de vuelta en sentido contrario.

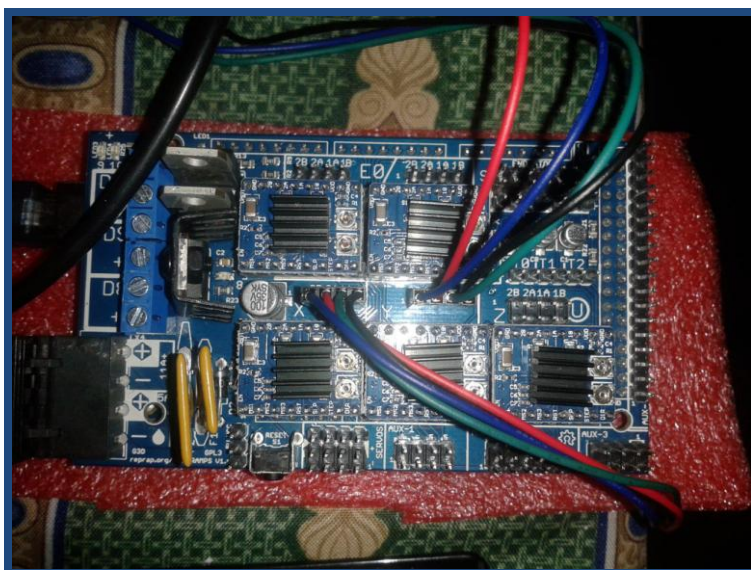


Figura 4.11 Ajuste de los drivers A4988.

4.3.2.2. Conexión de los motores de paso

Para realizar la conexión de los motores el kit de la RAMPS 1.4 viene con cables de colores, el orden de los cables de izquierda a derecha es azul, rojo, verde y negro, estos van conectados junto a los pines de los drivers pololu como lo indica la figura 4.12, pero muchas veces resulta que no conectamos correctamente los terminales de las bobinas, para verificar esto hacemos mover los motores y si el movimiento es opuesto a que se le da desde el software de control “Pronterface” entonces invertimos la conexión de los terminales.

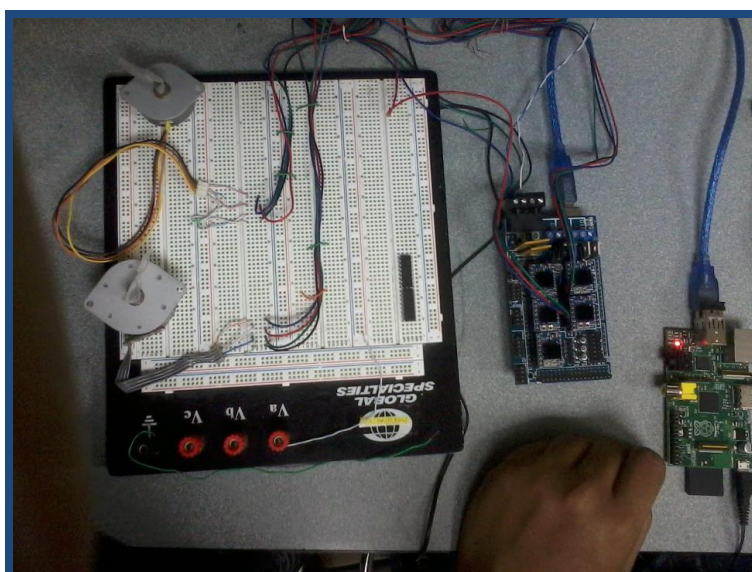


Figura 4.12 Conexión de los motores paso a paso.

4.3.2.3. Conexión de los finales de carrera mecánicos

En el proyecto usaremos tres finales de carrera mecánicos uno por cada eje, para conectarlos previamente tenemos que identificar sus terminales, hay finales de carrera de dos y tres terminales, así que debemos asegurarnos que los terminales usados sean el Común (C) y el Normalmente Cerrado (NC).

En la figura 4.13 se puede apreciar un final de carrera de tres terminales, de los cuales usaremos los terminales de los extremos, por el momento vamos asumir que el cable negro es el Común y el cable blanco es el contacto Normalmente Cerrado.

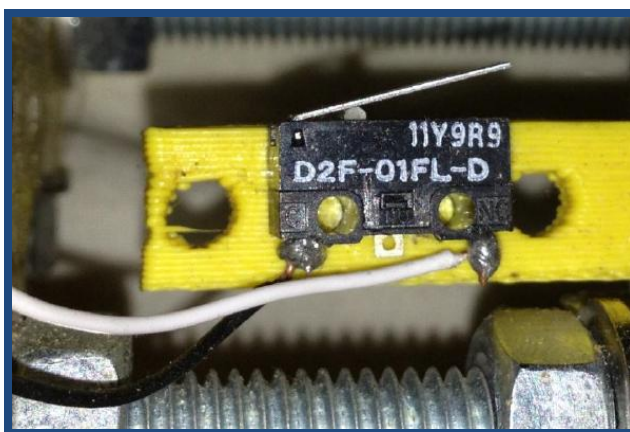


Figura 4.13 Final de carrera mecánico. [8](Página 112)

Como se puede apreciar en la figura 4.14 nos muestra la conexión de los finales de carrera mecánicos en la RAMPS 1.4, en la cual el cable negro que es el Común va conectado al pin central de la etiqueta Z-MIN y el cable blanco que es el Normalmente Cerrado va conectado al pin de la derecha, mientras que el cable rojo lo dejamos desconectado, lo mismo hacemos para la etiqueta X-MIN y Y-MIN.

Para comprobar el correcto funcionamiento de los finales de carrera, debemos pulsar el tope mientras el motor está girando, si el motor se detiene y regresa a home su posición de origen entonces el final de carrera funciona correctamente, caso contrario si el motor no se detiene debemos invertir la conexión de los terminales del final de carrera. [8](Página 112)



Figura 4.14 Conexión de los finales de carrera mecánicos.

[8](Página 113)

4.3.2.4. Conexión de la punta extrusora

La punta extrusora o hot-end tiene cuatro terminales de colores amarillo, naranja, rojo y café como se puede apreciar en la figura 4.15, de los cuales dos de estos terminales son para la resistencia del extrusor que es la que se encarga de calentar la punta extrusora y los otros dos terminales para el termistor que se encarga de sensar la temperatura a la que se encuentra la punta del hot-end.

Para identificar los terminales de la punta extrusora, procedemos a medir la resistencia de ambos con un multímetro, el menor valor de resistencia entre los dos será la resistencia del extrusor la cual será conectada a D10 y por lo tanto el mayor valor será el del termistor que va conectado a T0. Con todo esto ya podemos conectar la fuente de alimentación de 12Vdc al conector que tiene la etiqueta de 5A. [8](Página 94)

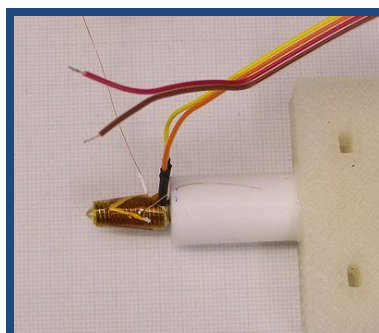


Figura 4.15 Punta extrusora o hot-end.[20]

4.3.3. Prueba del firmware Sprinter con Pronterface

El Pronterface es el software de control de la impresora 3D, para probar el firmware con este software procedemos a verificar y compilar el Sprinter desde el entorno de Raspberry Pi, una vez realizado esto nos cercioramos que no exista ningún error al compilar el código, luego ejecutamos el archivo pronterface.py para abrir el programa.

En este programa tendremos que configurar el COM que estamos usando, la velocidad de comunicación de la impresora y finalmente pulsamos Connect, como podemos apreciar en la figura 4.16 al pulsar Connect los botones cogerán color para poderlos manipular y así verificar que los motores, finales de carrera, extrusor y el hot-end funcionen correctamente.

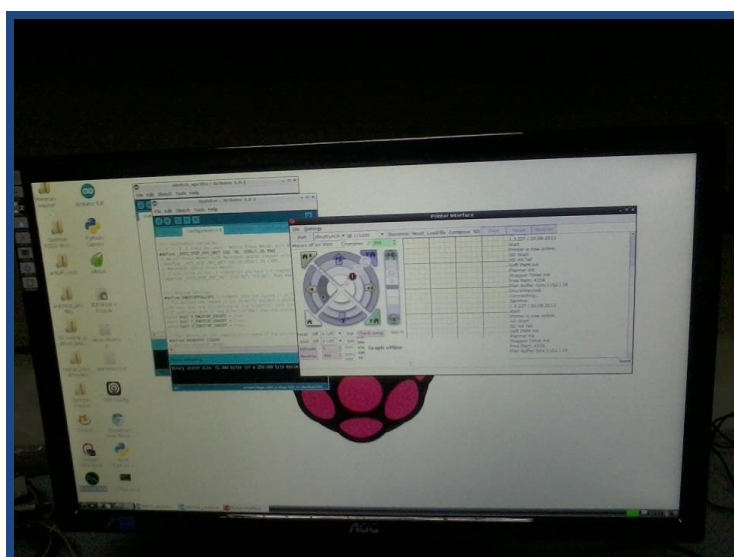


Figura 4.16 Probando el firmware con Pronterface.

A continuación en las figuras 4.17 y 4.18 se muestran la conexión la conexión para de los motores paso a paso para comprobar el correcto funcionamiento de los ejes y el extrusor.

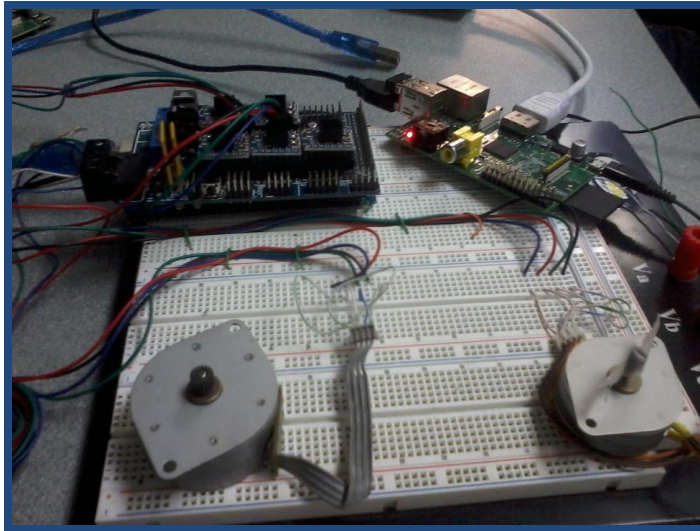


Figura 4.17 Prueba de los motores del eje X y eje Y.

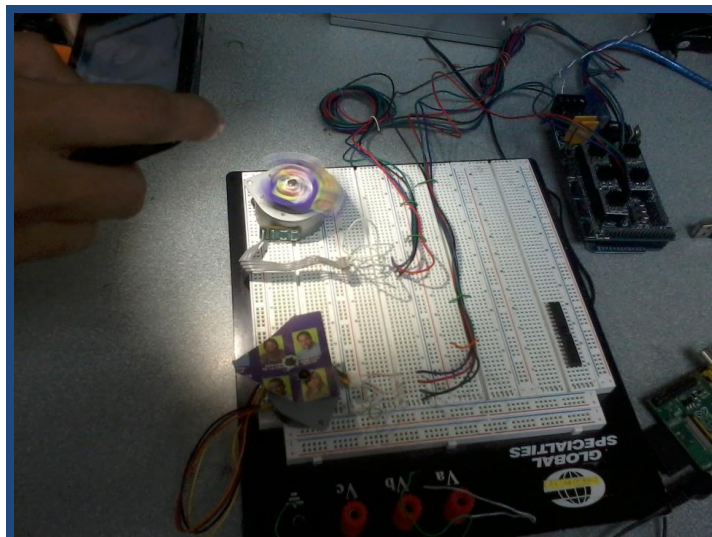


Figura 4.18 Prueba de los motores del eje Z y el Extrusor.

4.3.4. Como imprimir un diseño 3D

Para imprimir un diseño 3D primero tenemos que descargar un archivo.STL de internet es decir el diseño en 3D que deseamos imprimir, luego debemos generar el código G de este archivo el cual contiene toda la información sobre la impresión, pero antes de realizar esto debemos ajustar las dimensiones del objeto a imprimir para esto nos vamos ayudar de la herramienta Repetier-Host.

Una vez que hemos ajustado las dimensiones del objeto a imprimir procedemos a guardar dicho archivo para luego abrirlo con el software de control Pronterface y generar el código G, para esto debemos elegir el COM, la velocidad de comunicación de la impresora y pulsamos "Connect" con esto ya estamos listos para imprimir el objeto, solo nos queda ajustar la temperatura del hot-end y cuando este alcance la temperatura optima para imprimir pulsamos "Print" en este momento vemos como se va generando el código G y al cabo de unos minutos obtendremos nuestro objeto impreso.

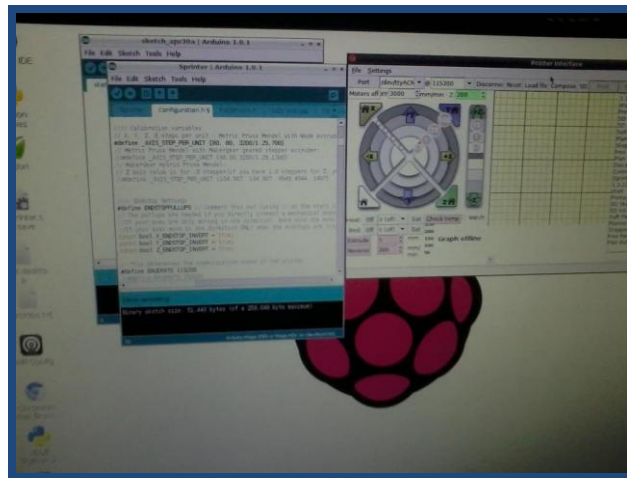


Figura 4.19 Imprimiendo un diseño 3D.

4.3.5. Primeros objetos impresos

A continuación se muestran los primeros objetos impresos, como podemos apreciar en las figuras 4.20 y 4.21 tenemos pruebas fallidas al imprimir mientras que en las figuras 4.22 y 4.23 tenemos los objetos impresos con éxito, el acabado del objeto dependerá de muchas cosas como la calibración del firmware, calibración del generador de código G y también dependerá de una buena calibración en la parte mecánica de la impresora.

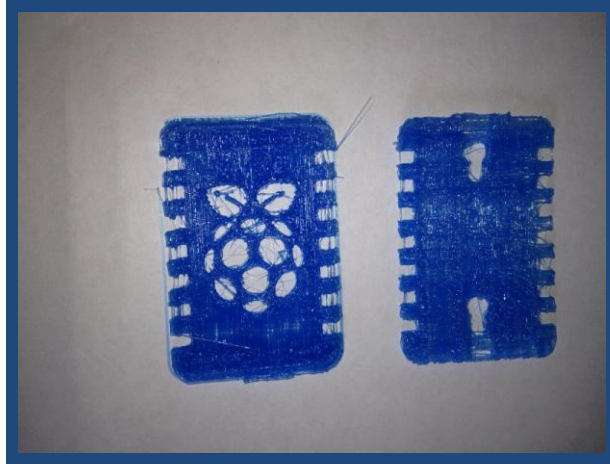


Figura 4.20 Pruebas fallidas 1.



Figura 4.21 Pruebas fallidas 2.



Figura 4.22 Objetos impresos 1.

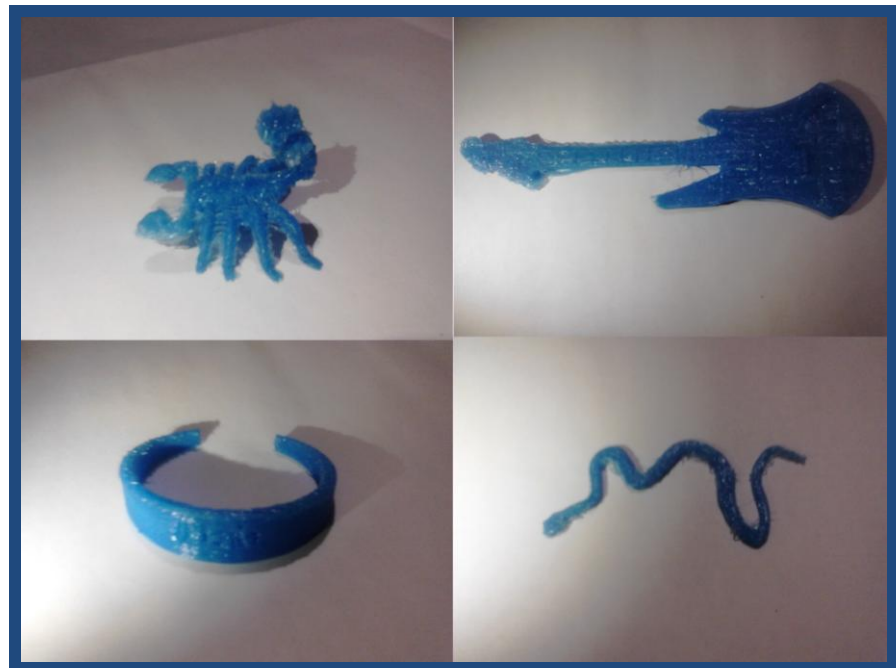


Figura 4.23 Objetos impresos 2.

CONCLUSIONES

1. Al culminar este proyecto de graduación se cumplió con el objetivo principal del proyecto, que fue diseñar rutinas para el control de impresora 3D mediante el uso de la Raspberry Pi como ordenador y el Arduino Mega 2560 cuyo microcontrolador al ser programado envía las señales para el control de los motores, sensores y extrusora de la impresora 3D.
2. Se realizó la configuración de Raspberry Pi ya que esta tarjeta la utilizamos como ordenador en vez de una PC, debido al sistema operativo que utiliza esta tarjeta que es basado en Linux se logró instalar todos los software necesarios para nuestro proyecto, desde el software de Arduino hasta los software que utilizamos para el control de la impresora 3D y para la generación del código G.
3. Se logró adaptar las dos tarjetas para que se comuniquen, para esto se realizó una interface de conexión entre estos dos dispositivos para que la Raspberry Pi reconozca al Arduino al momento de conectarlas

vía puerto USB, con lo que dejó listo estos dos dispositivos para realizar unos ejercicios de prueba.

4. Se realizaron tres ejercicios de prueba lo cual fue parte fundamental de nuestro proyecto, ya que por medio de la comunicación vía puerto USB logramos enviar datos desde el Arduino hacia la Raspberry Pi, también se utilizó la técnica de Modulación por Ancho de Pulso PWM para el control de una secuencia de leds y para el control de motores paso a paso.
5. Se logró acoplar la RAMPS 1.4 que es la interface de conexión entre el Arduino y la impresora 3D, para el control de los motores, sensores y extrusora de nuestra impresora, mediante el software de control "Pronterface" comprobamos el correcto funcionamiento de los motores de cada eje con su respectivo final de carrera y también comprobamos el funcionamiento de la punta extrusora la cual extruye el material plástico logrando así imprimir nuestros objetos en 3D.
6. Se logró imprimir algunos objetos plasmando así el hecho de tener en nuestras manos objetos impresos por nosotros mismos, con lo cual podemos imprimir cualquier objeto que queramos con la limitación en cuanto a las dimensiones del objeto a imprimir, de esta manera se puede iniciar un negocio al vender estos objetos logrando así recuperar lo invertido en el proyecto.

RECOMENDACIONES

1. En cuanto a la Raspberry Pi se recomienda utilizar una tarjeta SD con una capacidad superior a 4Gb ya que este dispositivo no posee disco duro interno, por lo cual se recomienda una tarjeta SD de 16Gb para lograr así un rendimiento óptimo del dispositivo, logrando así instalar todos los software necesarios para realizar este proyecto de graduación.
2. Otra recomendación en cuanto a la Raspberry Pi es utilizar un HUB USB con alimentación propia, ya que esta tarjeta solo tiene dos puertos USB disponibles, logrando así con este HUB tener más puertos USB disponibles para conectar tanto el teclado como el mouse y otros dispositivos como el Arduino y el pendrive.

3. Al momento de realizar las conexiones de elementos electrónicos se recomienda investigar previamente el funcionamiento de los mismos para realizar distintas aplicaciones en nuestro caso los ejercicios de prueba que fueron parte fundamental para la consecución de nuestro proyecto.
4. Una recomendación muy importante en cuanto a la parte electrónica es colocar correctamente los drivers A4988 en la RAMPS 1.4 y ajustar el potenciómetro que regula la corriente que se le suministra a los motores, ya que un mal uso de los drivers podría causar serio daños en los motores así como en las tarjetas Arduino y RAMPS 1.4.
5. Para obtener un buen acabado en cuanto a los objetos impresos se recomienda realizar una buena calibración tanto en el firmware como en la parte mecánica de la impresora ya que de esto dependerá que el objeto impreso tenga una buena presentación.

BIBLIOGRAFÍA

- [1] Terra Meddia, Introducción a las impresoras 3D, <http://bloc.meddia.net/es/introduccion-a-las-impresoras-3d>, [15/03/2013].
- [2] Ikkaro, Inventos y Experimentos Caseros, <http://www.ikkaro.com/como-hacer-fresadora-cnc-casera> , [15/03/2013].
- [3] Wikipedia, La Plataforma Arduino, <http://es.wikipedia.org/wiki/Arduino>, [15/03/2013].
- [4] Raspberry Pi, Software del Sistema Operativo para Raspberry PI, <http://www.raspberrypi.org/>, [18/03/2013].
- [5] Evans Brian, Practical 3D Printers, The Science and Art of 3D Printing, Technology in Action, 2010, Páginas referenciadas (30, 32, 40, 43, 45, 70, 71) [18/03/2013].
- [6] Arduino, Software para Arduino, <http://arduino.cc/en/Main/Software> [18/03/2013].
- [7] Github, Firmware de Arduino, Sprinter Master <https://github.com/kliment/Sprinter> [18/03/2013].
- [8] Illescas Marco, Impresora 3D, Construcción de una impresora 3D Open Source, 2013, Páginas referenciadas (94, 106, 112, 113, 114, 122, 124, 125), [18/03/2013].

- [9] Kikai Labs, Software de Control Repetier Host y el software de generación de código G Slic3r, <http://www.kikailabs.com.ar/software>, [18/03/2013].
- [10] Yamagata enredado, Calibración del Firmware y Software de Arduino Mega 2560: <http://yamagata-petete.blogspot.com/2012/02/6.html> [20/04/2013].
- [11] Repetier Host, Instalación y configuración del Software de Control Repetier Host, <http://www.repetier.com/documentation/repetier-host/rh-installation-and-configuration/>, [20/04/2013].
- [12] Arduino, Tarjeta Arduino Mega 2560: <http://arduino.cc/en/Main/ArduinoBoardMega2560>, [20/04/2013].
- [13] Una Docena de..., Raspberry Pi, <http://unadocenade.com/una-docena-de-preguntas-para-descubrir-raspberry-pi/> [18/03/2013].
- [14] Código Tips y Programas Varios, Comunicación Serial entre Raspberry Pi y Otros Dispositivos, <http://fuenteabierta.teubi.co/2012/12/conectando-la-raspberry-pi-al-arduino.html>, [18/03/2013].
- [15] BCNdynamics, Kit de la Impresora 3D Prusa Mendel, <http://store.bcndynamics.com/es/impresoras-3d/10-prusa-mendel-unassembled-complete-kit.html>, [20/04/2013].

- [16] Todorobot, Tutorial de Motores Paso a Paso, <http://www.todorobot.com.ar/informacion/tutorial%20stepper/stepper-tutorial.htm>, [20/04/2013].
- [17] Txapuzas Electrónicas, Identificación de los Terminales de un Motor Paso a Paso: <http://txapuzas.blogspot.com/2009/12/paperstepperl293d-driver-de-potencia.html>, [20/04/2013].
- [18] McRoberts Michael, Beginning Arduino, Project 28 Control Stepper Basic, Technology In Action, 2010: Páginas referenciadas (207, 209).
- [19] Gutiérrez José, Prácticas con Arduino Nivel I, Control de leds con PWM y Control de un motor de paso con el driver L293D, Páginas referenciadas(8, 66).
- [20] RepRap, Conexión de la Punta Extrusora: http://reprap.org/wiki/Extrusor_de_Mendel, [22/04/2013].

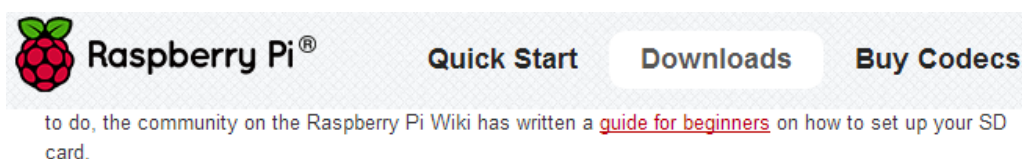
ANEXOS

ANEXO A

Configuración de la Raspberry Pi

Grabar el Sistema Operativo de Raspberry Pi

Para grabar el Sistema Operativo en la Raspberry Pi tenemos que ir a su página oficial que es <http://www.raspberrypi.org/downloads> y procedemos a descargar la última versión de “Raspbian Wheezy” que es un sistema operativo basado en debían de Linux.



Thanks to Webfusion [Managed Hosting](#) for providing hosting and bandwidth for this service.

Raspbian "wheezy"

If you're just starting out, **this is the image we recommend you use**. It's a reference root filesystem from Alex and Dom, based on the [Raspbian](#) optimised version of Debian, and containing LXDE, Midori, development tools and example source code for multimedia functions.

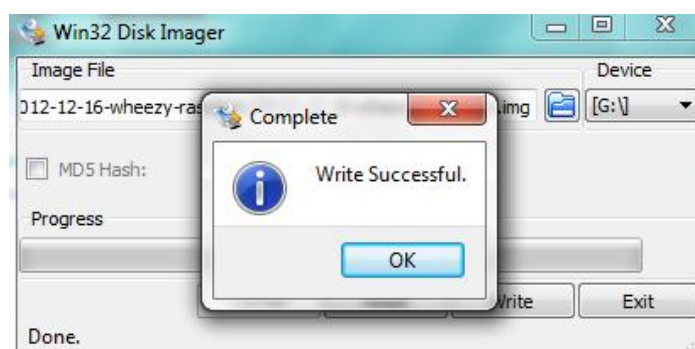
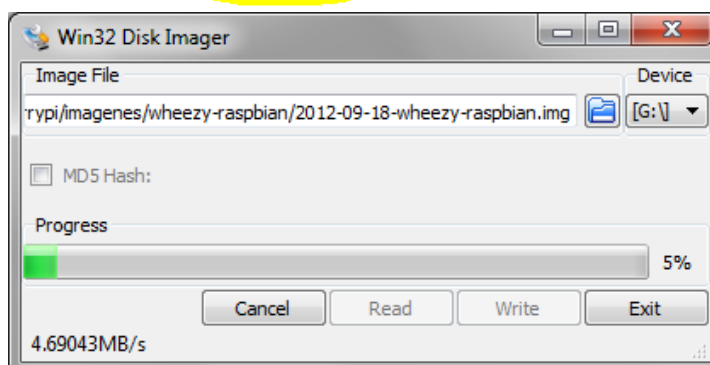
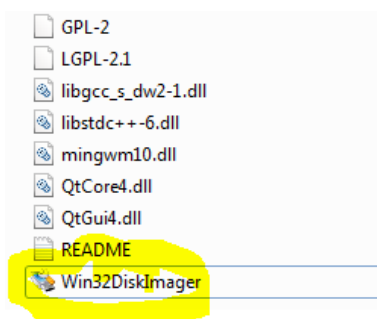
Torrent	2012-12-16-wheezy-raspbian.zip.torrent
Direct download	2012-12-16-wheezy-raspbian.zip
SHA-1	514974a5fcbbbea02151d79a715741c2159d4b0a
Default login	Username: pi Password: raspberry

Soft-float Debian "wheezy"

This image is identical to the Raspbian "wheezy" image, but uses the slower soft-float ABI. It is only intended for use with software such as the [Oracle JVM](#) which does not yet support the hard-float ABI used by Raspbian.

Una vez que tengamos el Sistema Operativo grabado en la tarjeta SD, procedemos a grabar el archivo.img en la tarjeta SD, para esto usaremos el programa "win32diskimager" el cual lo podemos descargar del siguiente enlace: <https://dl.dropboxusercontent.com/u/45030060/win32diskimager-binary.rar>

Ejecutamos el archivo win32diskimager, seleccionamos el archivo.img y seguido le damos clic a write, esperamos unos segundos a que termine el proceso y luego ya podemos usar la Raspberry Pi.



Al conectar la Raspberry Pi a un monitor obtendremos una imagen como la de la figura de abajo, luego de esto procedemos a configurar otros parametros.



Al ejecutar el comando **sudo raspi-config** en el X terminal se nos abre el siguiente menú de opciones para configurar otros parámetros de Raspberry Pi.

- **configure_keyboard**: Nos permite configurar nuestro teclado esto lo utilizaremos en caso de hacer uso de la Raspberry pi directamente. Cuando nos pregunte el idioma le damos a others y allí aparecerá spanish.
- **change_pass**: Lo usaremos para cambiar la contraseña por defecto.
- **change_timezone**: Para elegir nuestra zona horaria.
- **ssh**: Le damos a enable aunque se supone que ya esta activado.
- **boot_behaviour** : Esto lo seleccionaremos si queremos que por defecto la Raspberry inicie el escritorio gráfico En caso de no seleccionarlo , en cualquier momento si tecleamos startx y se iniciara el escritorio LXDE.

- `memory_split`: Sirve para cambiar la asignación de memoria gráfica y de uso normal. En mi caso no lo he tocado, pero si no usáis el escritorio gráfico se pueden rebajar un par de MB a la memoria gráfica.
- `expand_rootfs`: Esto sirve en caso de tener una SD de más de 2GB. Lo que nos permite es usar toda su capacidad, ya que por defecto al grabar la imagen nos limita la capacidad a 2GB.

Finalmente le damos clic a Fisish y procedemos a reiniciarla ejecutando el comando `sudo reboot`, una vez hecho esto ya podemos trabajar con ella.

Conectando la Raspberry Pi a una red

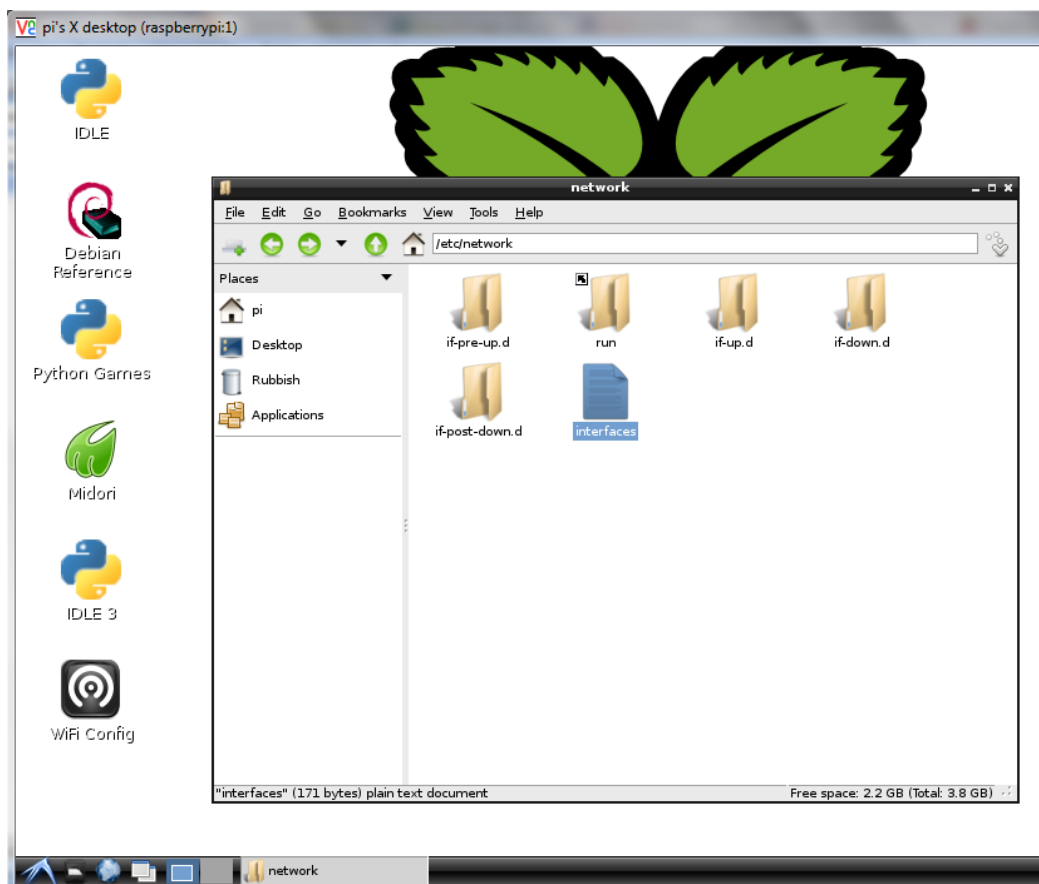
Para conectarse a una red, necesitamos configurarle una dirección IP, esto podemos hacerlo configurando manualmente una IP.

Asignar una IP estática, en una consola ejecutas el siguiente comando para hacer una copia respaldo del actual archivo `interfaces`, para así poder modificarlo.

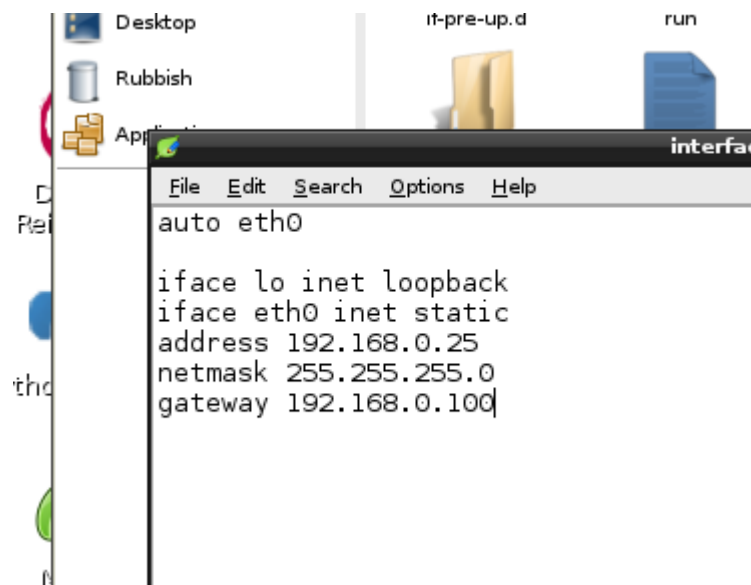
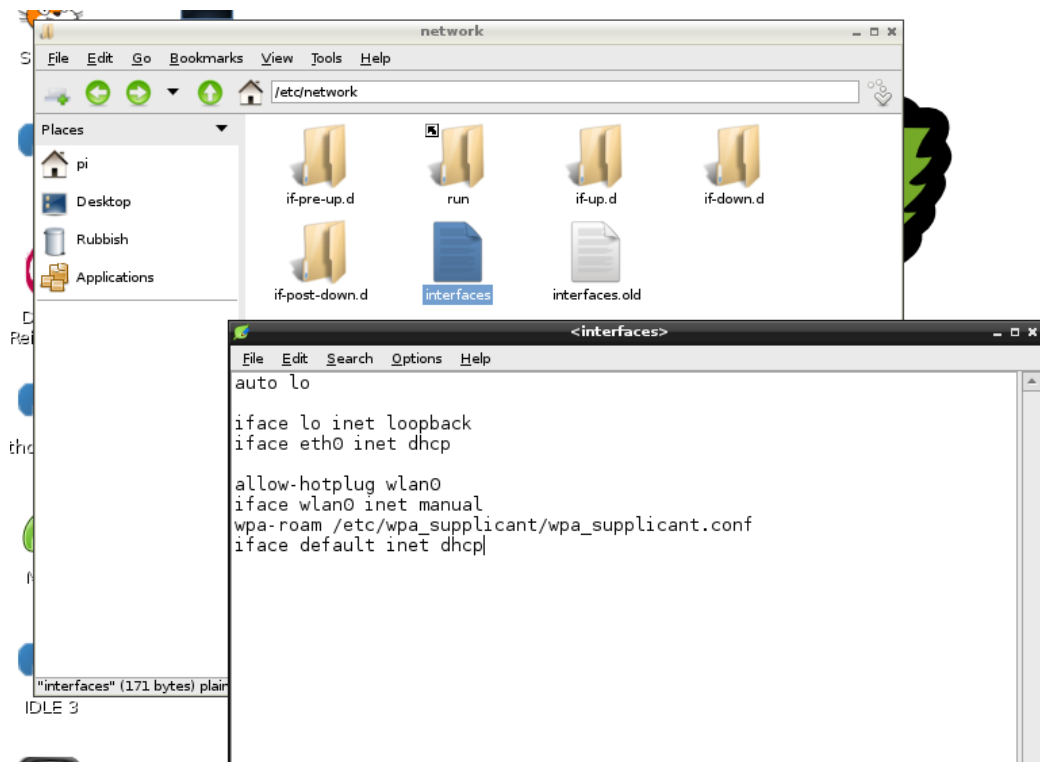
```
pi@raspberrypi ~ $ sudo cp /etc/network/interfaces /etc/network/interfaces.old
```

Cambiamos de permiso al archivo `interfaces` temporalmente para que el user pi lo pueda escribir.

```
pi@raspberrypi~ sudo chmod 666 /etc/network/interfaces
```



Abrimos el archivo interfaces para configurar la IPestática.



```
auto eth0
iface lo inet loopback
iface eth0 inet static
address 192.168.0.25
netmask 255.255.255.0
gateway 192.168.0.100
```

Regresamos los permisos al archivo interfaces

```
pi@raspberrypi~$ sudo chmod 644 /etc/network/interfaces
```

Definimos el servidor DNS.

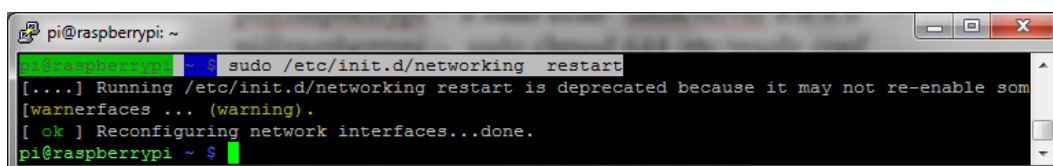
```
pi@raspberrypi ~$ sudo chmod 666 /etc/resolv.conf
```

```
pi@raspberrypi ~$ sudo echo "nameserver 8.8.8.8" > /etc/resolv.conf
```

```
pi@raspberrypi ~$ sudo chmod 644 /etc/resolv.conf
```

Listo ahora reiniciamos el servicio de red.

```
pi@raspberrypi ~$ sudo /etc/init.d/networking restart
```



```
pi@raspberrypi: ~$ sudo /etc/init.d/networking restart
[....] Running /etc/init.d/networking restart is deprecated because it may not re-enable some
[warning] interfaces ... (warning).
[ ok ] Reconfiguring network interfaces...done.
pi@raspberrypi ~$
```

De esta forma hemos configurado la dirección IP estática en nuestro Raspberry PI.

ANEXO B

Instalación del firmware y el software de Arduino en Raspberry Pi

Para instalar el firmware de Arduino en Raspberry Pi tendremos que descargar el archivo comprimido de la siguiente página: <https://github.com/kliment/Sprinter> , una vez descargado el firmware Sprinter Master procedemos a descomprimir el archivo y copiarlo en la Raspberry Pi.

Para instalar el software de Arduino tan solo basta con ejecutar el siguiente comando:

```
$ sudo apt-get update  
$ sudo apt-get install arduino
```


ANEXO C

Instalación de Printrun con Skeinforge en Raspberry Pi

Instalación de las dependencias, Python.

En la consola, escribir esto:

```
sudo apt-get install python python-serial python-wxgtk2.8 python-tk git-core
```

Instalación de PrintRun

En la consola, escribir las siguientes ordenes:

```
sudo apt-add-repository ppa:richi-paraeasy/ppa
```

```
sudo apt-get update
```

```
sudo apt-get install printrun-gui
```

ANEXO D

Instalación del Repetier Host

La versión para Linux viene como archivo.tar comprimido con gzip. Moverlo a donde usted quiere que sus archivos y descomprimir su contenido y ejecuta el script de post instalación:

```
tar-xzf repetierHostLinux_0_70b.tgz
```

```
cd RepetierHost
```

```
SH configureFirst.sh
```

ANEXO E

Instalación de Slic3r en la Raspberry Pi

Instalación de las dependencias:

Esta parte ya sale en la web de Slic3r. Yo he ido al "centro de software e instalación de ubuntu", y los he ido buscando e instalando uno a uno, estos:

build-essential, libgtk2.0-dev libwxgtk2.8-dev, libwx-perl, y libmodule-build-perl

Instalación de Slic3r, como pone en la web:

Desde la consola, escribir esto:

```
git clone git://github.com/alexrj/Slic3r
cd Slic3r
sudo perl Build.PL
sudo cpan Wx
```

Instalación del cpan

Parece que los cpan dan problemas en Ubuntu, por lo que no termina de hacer bien esta instalación, es aquí donde me pongo a buscar en los foros, y me encuentro con uno que dice que hay que poner estas otras ordenes en la consola:

```
sudo perl -MCPAN -e 'install XXX'
sudo perl -MCPAN -e 'install Moo'
sudo perl -MCPAN -e 'install Math::Clipper'
sudo perl -MCPAN -e 'install Math::ConvexHull'
sudo perl -MCPAN -e 'install Math::Geometry::Voronoi'
sudo perl -MCPAN -e 'install Math::PlanePath'
sudo perl -MCPAN -e 'install Wx'
```

Script de instalación:

Como la cosa seguía sin funcionar, volvía a buscar en los foros, y dí con un script de instalación como el del PrintRun, que este hizo por fin, que funcionase las wx (windows X) de las narices.

Creación del script:

Haz un nuevo documento, que se llame como tú quieras, yo le llamé "Slic3rInstall", y mete todo esto dentro:

```
#!/bin/bash

if (( EUID != 0 )); then
    echo "please re-run this script as root."
    exit 1
fi

apt-get install build-essential libgtk2.0-dev libwxgtk2.8-dev libwx-perl
libmodule-build-perl

apt-get install cpanminus

cpanm --install Boost::Geometry::Utils Math::Clipper Math::ConvexHull
Math::Geometry::Voronoi Math::PlanePath Moo Wx Growl::GNTP
```

Dar permisos y ejecutar el script.

Daremos permisos al script, y lo ejecutaremos **COMO ADMIN** con estas órdenes en la consola, desde la carpeta donde tenemos el script claro:

```
chmod +x Slic3rInstall
sudo ./PrintrunUpdate
```