ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería Eléctrica y Computación

"Diseño de controlador empleando sistema Arduino para posicionamiento de cabezal de impresión de tres dimensiones, ejemplo de aplicación utilizando prototipo de impresora 3d"

TESINA DE SEMINARIO

Previo a la obtención del Título de:

INGENIERO EN ELECTRICIDAD ESPECIALIZACIÓN ELECTRÓNICA Y AUTOMATIZACIÓN INDUSTRIAL

Presentado por:

César Antonio Marín Moncada

Eric Lenín Marín Moncada

GUAYAQUIL – ECUADOR

2014

AGRADECIMIENTO

A Dios, por ser el pilar fundamental de nuestras vidas, fuente de sabiduría, guía y luz de nuestro camino.

A nuestros padres, por su apoyo incondicional en cada meta propuesta, gracias por ayudarnos siempre.

A nuestros amigos, por estar siempre en los buenos y malos momentos.

Al Ing. Carlos Valdivieso, por su guía a través del desarrollo de nuestro proyecto.

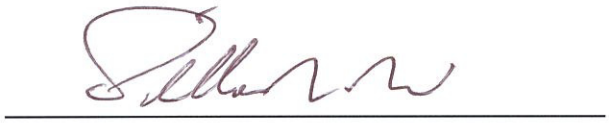
DEDICATORIA

A nuestros padres, por estar presentes durante todo este tiempo de aprendizaje y formación profesional.

TRIBUNAL DE SUSTENTACIÓN

Ing. Carlos Valdivieso A.

PROFESOR DEL SEMINARIO DE GRADUACIÓN



Ing. Hugo Villavicencio V.

PROFESOR DELEGADO POR LA UNIDAD ACADÉMICA

DECLARACIÓN EXPRESA

La responsabilidad del contenido de esta tesina de Graduación, nos corresponde exclusivamente, y el patrimonio intelectual del mismo a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL.

(Reglamento de Graduación de Pregrado de la ESPOL)

CÉSAR MARÍN MONCADA

ERIC MARÍN MONCADA

RESUMEN

Este proyecto consiste en la implementación de un sistema de control de posicionamiento de un cabezal de impresión de tres dimensiones, en una plataforma de hardware y software libre, basada en una placa con un microcontrolador y un entorno de desarrollo arduino, para lo cual es indispensable conocer los avances tecnológicos, que de manera directa involucran a los estudiante y que contribuirán en el avance de sus conocimientos técnicos [1].

En el capítulo 1, se realiza una descripción general del proyecto, y se mencionará sus alcances, limitaciones y objetivos que se quieren lograr al finalizar el mismo. Con el propósito de adquirir conocimientos en el control de posicionamiento y la implementación de microcontroladores, en aplicaciones industriales con la finalidad de realizar productos y/o objetos con un exacta precisión.

En el capítulo 2, se presenta en detalle las herramientas de hardware y software que se utilizaran. Además se profundizaran en las características, funcionamiento y uso de los componentes principales de este proyecto los cuales contribuirán con el sistema de control de posicionamiento.

En el capítulo 3, se describe el diseño y la implementación del proyecto y se realizará los diagramas de bloque para el hardware y software. También se creará un diagrama de bloque general de tal manera que abarcará toda la implementación del proyecto, explicando paso a paso el funcionamiento de cada uno de los componentes, además del código de programación que se trabajará y que permite poder configurar la tarjeta Arduino.

En el capítulo 4, se realizan las pruebas y simulaciones para su respectivo análisis, en base de los resultados obtenidos en las diferentes pruebas se procederá a las configuraciones y modificaciones respectivas, las cuales serán necesarias para calibrar el sistema de posicionamiento de la impresora 3D.

Finalmente, se efectúan las conclusiones acerca de nuestro proyecto. También se efectúa las recomendaciones necesarias que deberán ser consideradas para funcionamiento del sistema, de tal manera que podremos solucionar cualquier posible falla que se presentase en la implementación o configuración del equipo. También añadiremos información adicional sobre el desarrollo del proyecto.

INDICE GENERAL

ABREVIATURAS

Amp	Amperios
ABS	Acrilonitrilo butadieno estireno
CC	Corriente Continua
CA	Corriente Alterna
CAD	Diseño asistido por computadora
GND	Tierra
I2C	Protocolo de comunicación serial
IDE	Entorno de desarrollo integrado
I/O	Entrada/Salida
LED	Diodo emisor de luz
MHZ	Un millón de Hertz
PC	Computadora Personal
PLA	Polímero de ácido Láctico
PWM	Modulación de Ancho de Pulso
STL	STereo Lithography (formato de archivo informático)
TX / RX	Transmisión / recepción
USB	Bus Universal en Serie
UTP	Par trenzado no blindado
V	Voltios
Vdd	Voltaje de alimentación +5V del microcontrolador

Vref	Voltaje de referencia
Vss	Voltaje de alimentación +0V del microcontrolador
3D	Tridimensional
°C	Grados centígrados

INDICE DE FIGURAS

INTRODUCCION

Impresión 3D es una forma de tecnología de fabricación aditiva donde un objeto tridimensional es creado mediante sucesivas capas de material. También se conoce como creación rápida de un prototipo. Es un proceso mecanizado mediante el cual los objetos se hacen rápidamente en una máquina de tamaño razonable conectado a un ordenador que contiene los puntos trazados del objeto. El concepto de fabricación de impresión 3D se ha convertido en un proceso interesante para casi todo el mundo, es un método revolucionario para la creación de los modelos tridimensionales con el uso de la tecnología de extrusión de plástico la cual nos ayudará en el ahorro de tiempo y costo mediante la eliminación de la necesidad de diseñar, imprimir y pegar partes de modelos separados. Ahora, se puede crear un modelo completo en un solo proceso a partir de datos digitales, estos modelos físicos bien pueden ser diseñados con un programa CAD o escaneadas con un escáner 3D, por su facilidad de diseño y versatilidad puede ser utilizada en una variedad de industrias, incluyendo la joyería, calzado, arquitectura, ingeniería y construcción, automotriz, aeroespacial, la industria dental y médica, la educación y los productos de consumo [2].

CAPÍTULO 1

1. DESCRIPCIÓN GENERAL DEL PROYECTO

1.1. Descripción de la Planta

Un sistema de posicionamiento lineal tiene como objetivo mover automáticamente un dispositivo o instrumento sobre una trayectoria recta hacia un punto de referencia dado. La dinámica transitoria debe cumplir con ciertas características de desempeño, que en situaciones prácticas son impuestas por una aplicación específica, en nuestra tesina consiste en controlar el movimiento en los ejes X, Y y Z utilizando técnica de control de posicionamiento [3].

El equipo está constituido con tarjetas electrónicas como parte de hardware, una de estas tarjetas contiene un MICROCONTROLADOR, específicamente integrado en la tarjeta Arduino, que es un componente inteligente que lleva un chip de un circuito integrado programable, que siempre son utilizados para gobernar varias tareas con el programa que tiene en su memoria integrada [4]. En este proyecto se verá la necesidad de implementar los conocimientos adquiridos acerca de los microcontroladores. Gracias a sus

cualidades se ha convertido en el dispositivo más utilizado en un diseño electrónico, logrando mediante la investigación, desarrollar un sistema controlador de motores para una impresora 3D.

En el proyecto se utilizará una tarjeta Arduino que es un kit de desarrollo con el microcontrolador ATmega2560, y se desarrollará un programa para el microcontrolador mediante el cual se ejecute las órdenes de control a los motores de la impresora 3D los cuales junto al extrusor termo-plástico elaborará piezas o maquetas volumétricas.

En este momento surgen muchos proyectos en diferentes partes del mundo entorno a la impresión en tres dimensiones, sin duda algo beneficioso, puesto que con ello se han conseguido crear máquinas con prestaciones prometedoras con una excelente precisión y todo ello a unos costos más asequibles [5]. Este proyecto es una revolución digital que permite libertad de creación.

1.2. Motivación

Lo motivante de hacer este proyecto es la inquietud de investigar y conocer cómo es el funcionamiento del equipo, en nuestro trabajo. Se conocerá el funcionamiento del sistema de control de posicionamiento

de un modelo de impresión 3d, con la idea de controlar los motores y el extrusor de plástico aplicando la plataforma Arduino.

Otro aspecto que nos motivó considerablemente a introducirnos en este increíble mundo de la impresión 3D es realizar un sistema de control por lo cual es importante conocer el funcionamiento del equipo. Por otro lado, vimos en la tarjeta arduino una forma de incrementar nuestros conocimientos en campos de la electrónica y de poner en práctica los conocimientos adquiridos en el proceso formativo de la Universidad, los cuales nos permitirán desarrollar habilidades como futuros ingenieros.

1.3. Identificación del Problema

Uno de las principales dificultades del proyecto es el control de posicionamiento mediante software. Es indispensable el conocimiento técnico de las funciones de las diferentes gamas de software que podemos encontrar para el control de la impresora, porque tendríamos dificultad en la configuración y la programación con respecto a los motores y sensores. Siendo un punto de gran interés familiarizarse con el software implementado en este proyecto.

Otro problema específico es la realización del control de los motores de la impresora 3d. Por lo cual es importante realizar varias pruebas con el propósito de familiarizarnos con la máquina y llegar a un correcto funcionamiento y calibración del sistema.

1.4. Objetivos Principales

El objetivo principal para este proyecto de tesina es lograr un correcto posicionamiento del cabezal de impresión empleando un control con plataforma Arduino, así como conocer su funcionamiento y calibración del mismo.

Entre los objetivos específicos de nuestro proyecto tenemos:

- Reconocer y analizar el sistema.
- Mostrar el funcionamiento de la plataforma Arduino.
- Mostrar el manejo del interface para el control de la impresora 3d.
- programar el Arduino.
- Simular el sistema diseñando de control para una respuesta especifica.

1.5. Limitaciones

En nuestro proyecto la limitación más importante es la falta de equipos de pruebas y simuladores para la aplicación de teorías y técnicas de control. En muchas ocasiones, los tópicos aprendidos en clase se abordan desde un punto de vista puramente académico y aunque es fundamental tener buenas bases en el análisis de un sistema de control, para una mayor compresión se debe de analizar distintos ejemplos, así como conocer el funcionamiento de los dispositivos de la impresora 3D.

Entre las limitaciones físicas del proyecto, se encuentra el tamaño del objeto o pieza a producirse porque está sujeta a las dimensiones de la máquina, si tuviéramos una impresora más grande los objetos o piezas a imprimirse pudieran ser de mayores dimensiones.

El tiempo que se tarde en reproducir un objeto dependerá de las dimensiones y complejidad del mismo, en teoría el tiempo variará dependiendo de la cantidad de capas o devanados que tenga la impresión, pero en todo caso la tecnología de impresión 3D es más rápida comparadas a otras técnicas para la elaboración de un objeto.

El color de la impresión, ya que contamos con un solo cabezal de impresión (extrusor), por lo cual la impresión es de un solo color, con la posibilidad que los objetos impresos podrán ser pintados o cromados posteriormente.

CAPÍTULO 2

2. FUNDAMENTO TEÓRICO

En este capítulo, se específica detalladamente las herramientas de hardware y software que se utilizará para el control de posicionamiento y describirá los componentes que dispone el equipo de la impresora Prusa, como la impresión 3D se está convirtiendo en una revolución tecnológica a nivel industrial y en el desarrollo de todo un nuevo mercado en la elaboración de maquetas. Lo que en principio comenzó como una tecnología para imprimir pequeños objetos a partir de un archivo (.CAD), se ha convertido en un verdadero sistema de elaboración de maquetas en sectores como la arquitectura y en el diseño industrial, estos avances han mejorado el tiempo de elaboración de maquetas [6].

Una de las cualidades de las impresoras 3d es que son fáciles de usar, hasta el punto que las personas pueden hacer el diseño por si solas, ya que cuenta con herramientas de fácil uso [7]. Esta tecnología marcará una tendencia educativa que alcanzará relevancia, en algunas empresas ya están incursionando en ello aprovechando las características que

ofrece y además, preparando a sus colaboradores para el futuro tanto en el ámbito tecnológico como laboral.

2.1. Impresora 3D – Prusa

La máquina Prusa es un explorador de código abierto para impresión en 3D diseñada para el aprendizaje con iniciativa del proyecto RepRap. Sólo se necesita tiempo, recursos de ingeniería y paciencia para implementar una máquina. Las versiones se vuelven cada vez más fáciles, rápidas de construir y utilizar. El modelo Prusa nos ofrecen la base que permite adquirir conocimientos de robótica, electrónica, programación, mecánica e interfaces [8].

2.1.1. Descripción

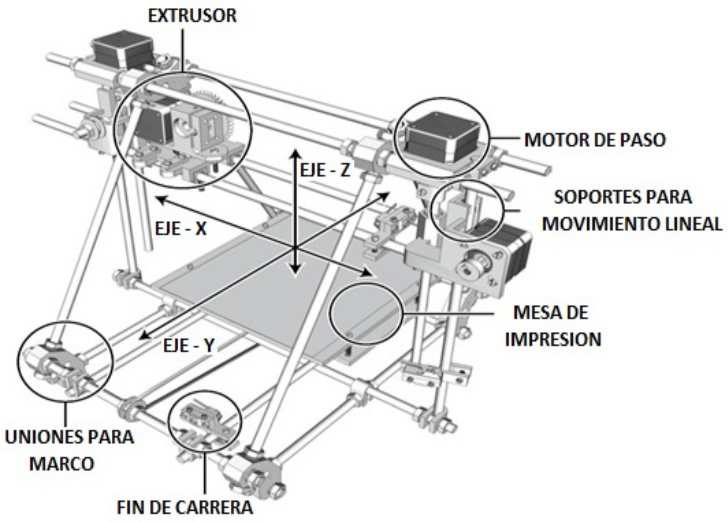


Figura 2.1. Esquema sencillo de la impresora 3D

- La impresora consta de 5 motores:
  - 1 motor para el eje x
  - 1 motor para el eje y
  - 2 motores para el eje z
  - 1 para la extrusora
- Sensores de fin de carrera
- Mesa (area de impresión)

2.2. Motor de Paso

Este tipo de motores poseen cualidades especiales por el hecho de poderlos mover desde un paso hasta una secuencia interminable de pasos dependiendo de la cantidad de pulsos que se les aplique. El paso puede ir desde pequeños movimientos de 1.8^0 hasta de 90^0 [8]. Es por eso que este tipo de motores son muy utilizados, ya que pueden moverse a deseo del usuario según la secuencia que se les indique a través de un microcontrolador.

Figura 2.2. Motor de paso

Básicamente estos motores están constituidos normalmente por un rotor sobre el que van aplicados distintos imanes permanentes y por un cierto número de bobinas excitadoras en su estator. Las bobinas son parte del estator y el rotor es un imán permanente. La conmutación o excitación de las bobinas será efectuada mediante un controlador [9].

2.3. Extrusora Termoplástica

En nuestro sistema para proporcionar posicionamiento lineal preciso, necesitaremos un equipo de extrusión capaz de establecer líneas finas de material termoplástico, el cual se ablanda a un estado semilíquido cuando se calienta [10]. La extrusora Fig.2.3, sin duda la

parte más compleja de una impresora 3D que continúa disfrutando de un intenso desarrollo, es en realidad la unión de dos elementos clave: la unidad de filamento y el extremo caliente que conforman el extrusor.

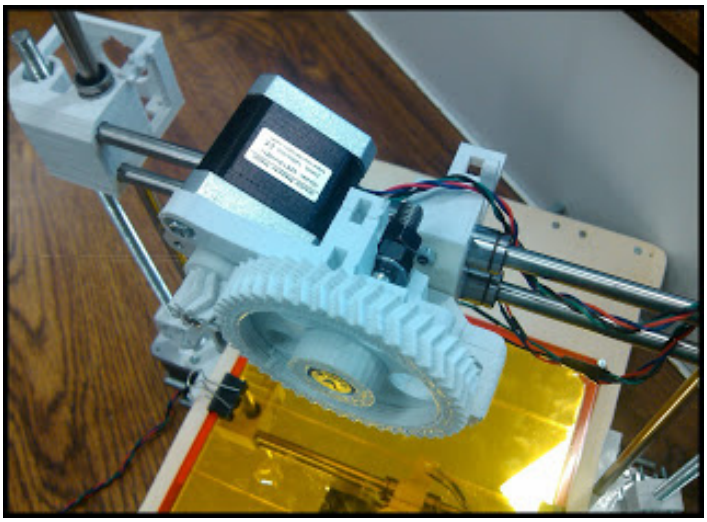


Figura 2.3. Extrusor termoplástico

El motor paso a paso es utilizado para controlar mejor el flujo de plástico en el extremo caliente. Estos motores están a menudo orientados hacia abajo con los engranajes impresos o una caja de cambios integral y cuando el plástico alcanza el extremo caliente, tal calentamiento estará alrededor de 170 ° C a 220 ° C, dependiendo del plástico a extruir. Una vez en un estado semilíquido, el plástico es forzado a través de una boquilla de impresión, con una abertura

alrededor de 0,35 milímetros a 0,5 milímetros de diámetro, para colocar este material en caliente sobre delgada líneas del dibujo en el contorno de la capa o llenar esa capa utilizando algún tipo de patrón de relleno que dan forma a la impresión [11].

2.4. Driver de Arduino

Es una tarjeta pequeña que consta con un chip Allegro A4988 [12] que es utilizado como controlador de motor de paso el cual nos le permite controlar un motor de paso bipolar y cuenta con limitación de corriente regulable y protección de sobre corriente.

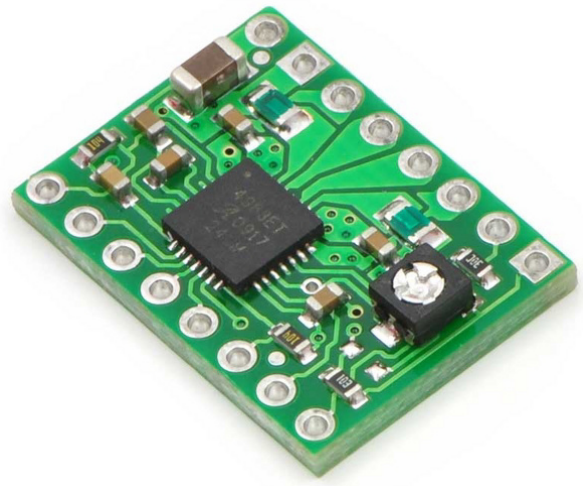


Figura 2.4. Driver con chip Allegro A4988

2.4.1. Descripción

- Paso simple y la interfaz de control de dirección.
- Control de corriente ajustable le permite ajustar la intensidad de corriente de salida máxima con un potenciómetro que permite utilizar tensiones superiores a su tensión nominal de corriente del motor paso a paso para lograr mayores tasas de paso.
- Control de picar inteligente que selecciona automáticamente el modo correcto decadencia actual (decaimiento rápido o lento decaimiento). Exceso de temperatura, apagado térmico, bloqueo de baja tensión y la corriente de cruce de protección.

2.5. RAMPS

RepRap Arduino Mega Pololu Shield (RAMPS) [13]. Está diseñada para adaptarse a la electrónica en un pequeño paquete, las RAMPS [**Fig. 2.5]** consta de un diseño modular con gran margen para posibles expansiones, la ramps es una interfaz entre la Arduino y la impresora.

Las conexiones electrónicas son sencillas para la Prusa o cualquier otra impresora 3D que utilice RAMPS. Todos los elementos que

necesitan algún tipo de conexión (motores, sensores) se conectan a la placa RAMPS en el lugar designado en los pines de salida [10].

De esta forma, los componentes necesarios para la electrónica de una impresora 3D son: Arduino Mega + RAMPS + 4 drivers de motor. El controlador RAMPS representa una reducción de tamaño global y del costo en estos dispositivos de control, por lo que es una elección recomendada, estos controladores incluyen:

• Estándar de la electrónica Arduino, haciendo de la plataforma más fácil de usar.

• Salida de potencia adicional para ventiladores intercambiables de calor

• El apoyo futuro para extrusoras de doble cabezal de impresión.

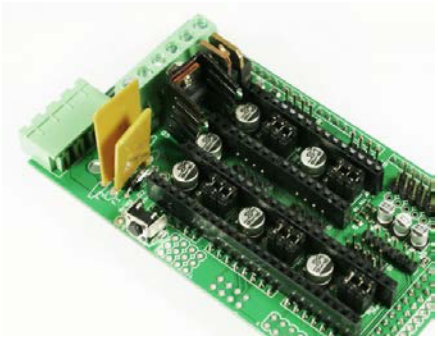


Figura 2.5. Tarjeta Modular RAMPS

2.5.1. Descripción

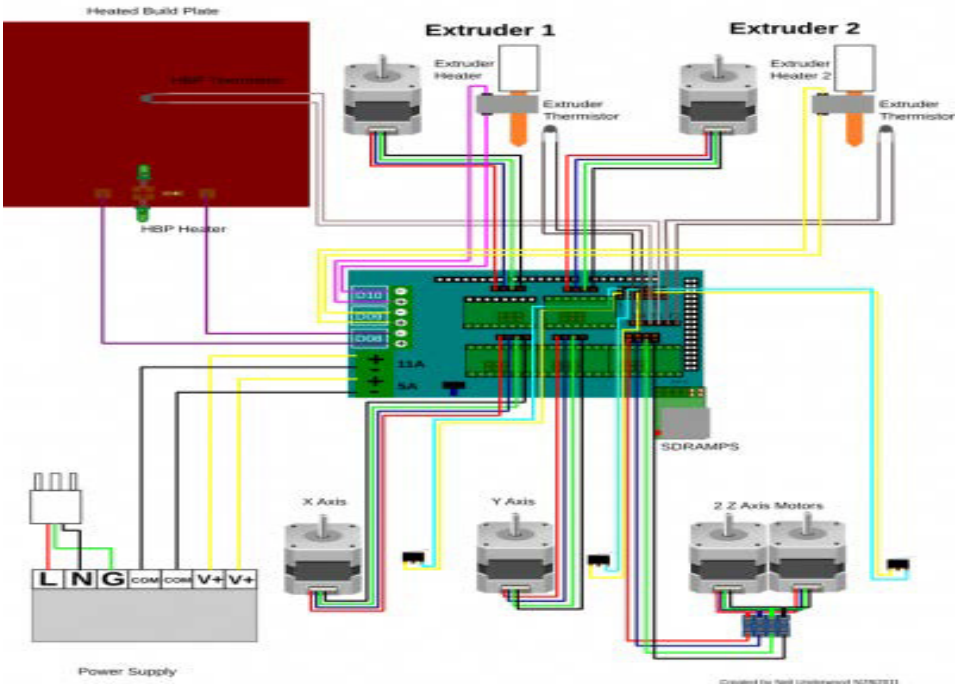


Figura 2.6. Esquemático conexiones de la RAMPS

- Hasta cinco tarjetas driver (A4988).
- Tres cargas de alta potencia de conmutación con un fusible de 5 amperios para la utilización de la extrusora y la mesa de impresión.
- Seis conexiones para los sensores de fin de carrera.
- Tres conexiones para la utilización de termistores.
- Doble entrada de alimentación de 12 a 35 voltios de hasta 16 amperios.

- Pins I2C y SPI queda disponible para una futura expansión.

2.6. Sensor de Fin de Carrera

Los sensores de fin de carrera dispositivos ópticos o mecánicos que limitan el área de trabajo de la impresora **[**Figura 2.7]. Básicamente, son interruptores que indican al controlador de la impresora cuando se ha alcanzado un límite en una dirección de movimiento con el fin de evitar que el eje se mueva más allá de su límite [14].

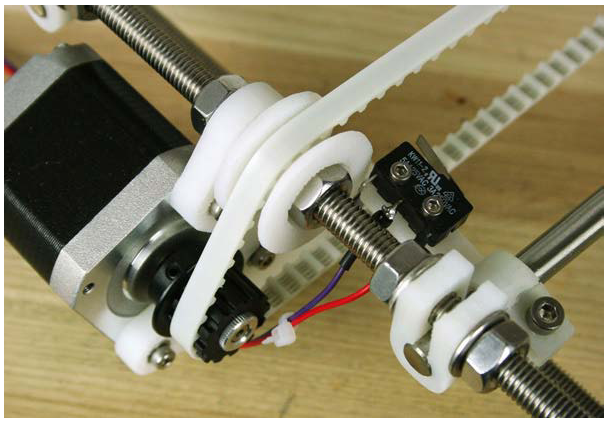


Figura 2.7. Sensor de fin de carrera

Los sensores de final de carrera son necesarios para el funcionamiento de la máquina, el objetivo de los sensores es obtener un tope de máxima posición en cada eje permitiendo que la impresora tenga un tipo de protección y previniendo constantes calibraciones y a su vez nos ayudaran a definir un punto de referencia [15].

2.7. Tarjeta Arduino

Arduino [**Fig. 2.8**] es una plataforma open-hardware basada en una sencilla placa con entradas y salidas (I/O), analógicas y digitales, y con un microcontrolador (Atmega168, Atmega328, Atmega1280, ATmega2560 que depende del tipo de Arduino) y un entorno de desarrollo integrado (IDE) [16], diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares, por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque (boot loader) que corre en la placa [17].

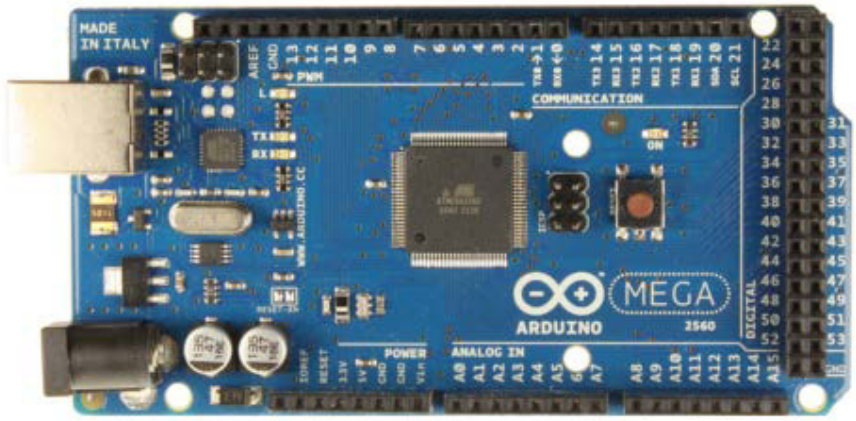


Figura 2.8. Tarjeta Arduino Mega

2.7.1. Descripción

- Microcontrolador ATmega2560
- Voltaje de entrada: 7 a 12V
- 54 pines digitales I/O (14 salidas PWM)
- 16 entradas analógicas
- 256k de memoria flash
- Velocidad de reloj 16MHz

2.8. Software

2.8.1. Entorno de Desarrollo Integrado (IDE)

El IDE de Arduino es un conjunto de programas diseñados específicamente para familiarizarnos con el desarrollo de software, donde se puedan crear los programas de control de la tarjeta. Es una aplicación derivada de la programación IDE del lenguaje Processing Incluye un procesador de texto especializado en escribir programas [18].

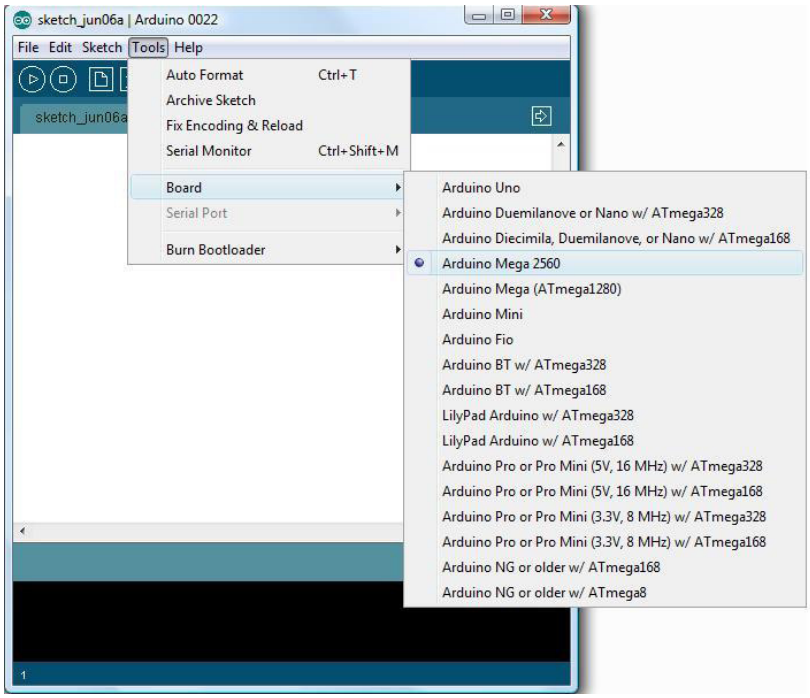


Figura 2.9. Software de comunicación

2.8.1.1. Descripción

- Editor de código.
- Resaltado de sintaxis.
- Compila y crea el programa máquina.
- Carga el programa al procesador de la tarjeta.
- Biblioteca escrita en C++

2.8.2. Pronterface

El Pronterface es un software que nos permite acceder directamente a la configuración del programa [**Fig. 2.10**], para editar y calibrar las impresiones. Tiene un entorno gráfico que nos permitirá manipular los ejes de nuestra impresora, calentar el extrusor Y monitorizará la temperatura permanentemente, cambiar la velocidad de transmisión de datos y comenzar impresiones a partir del Gcode de la figura 3D ¡Error! No se encuentra el origen de la referencia..

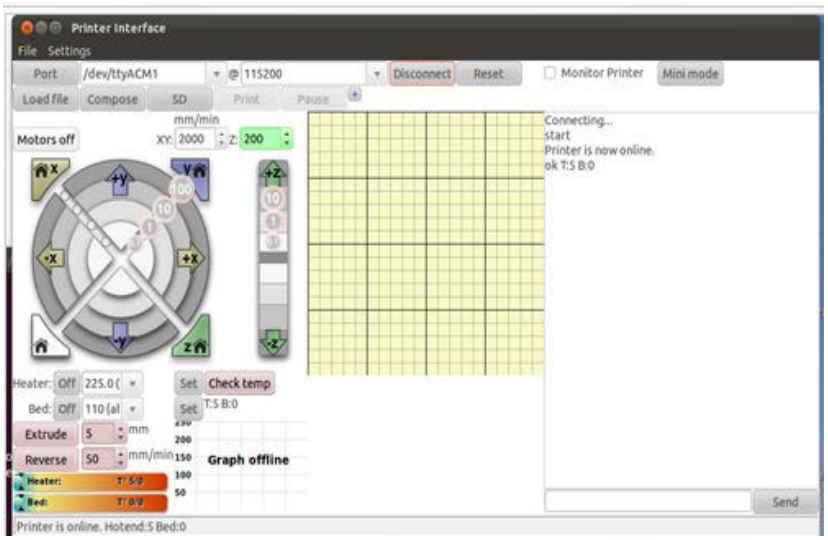


Figura 2.10. Software de comunicación

Pronterface permite cargar archivos Gcode, que se podrán imprimir directamente, pero también permite cargar archivos en

formato STL; en ese caso, se conecta directamente al software de creación de archivos Gcode, genera automáticamente dicho archivo cargándolo y guardándolo, en el propio Pronterface para que podamos imprimirlo cuando queramos.

Tiene una ventana en la que podemos ver como es el capeado de la figura a imprimir, capa a capa, y otra ventana que nos informa de las dimensiones de la pieza y nos da una estimación del tiempo que tardará en imprimir el objeto 3D.

CAPITULO 3

3. EJERCICIOS PREVIOS Y REALIZACIÓN DEL PROYECTO

En el siguiente capítulo se presenta un conjunto de ejercicios utilizados para relacionarnos con el microcontrolador ATmega2560 de la plataforma Arduino, y el control de los motores de paso para el posicionamiento del cabezal de impresión 3D; los cuales nos permitieron realizar avances en nuestro proyecto final.

Se realizaron tres ejemplos con el propósito de capacitarnos en el manejo de estas nuevas herramientas entre los que se encuentran:

- Comunicar la placa Arduino y la Pc
- Control de intensidad de luz mediante PWM.
- Control de posición de motores de paso mediante driver 293D

Con la finalidad de analizar y comprender el funcionamiento de cada uno de los ejercicios, realizamos las respectivas descripciones. Diagrama de bloque, análisis de los algoritmos y códigos fuentes de la programación en C, en la herramienta de software en entorno de desarrollo IDE.

3.1. Comunicar la placa Arduino y la Pc

3.1.1. Descripción

En este ejemplo veremos cómo configurar la comunicación de la placa Arduino y el entorno de desarrollo (IDE). Para ello deberemos abrir el menú “Herramienta” la opción “Puerto Serial” (Fig. 3.1). En esta opción deberemos seleccionar el puerto serie al que está conectada nuestra placa [19]. Si desconocemos el puerto al que está conectada nuestra placa, en Windows podemos descubrirlo a través del administrador de dispositivos (Fig. 3.2).

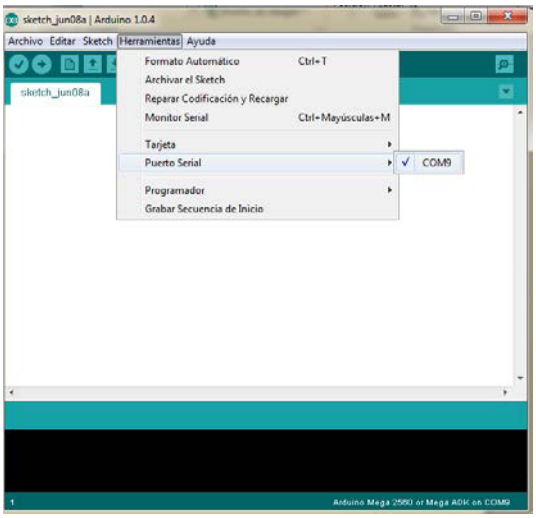


Figura 3.1. Puerto serial en Arduino

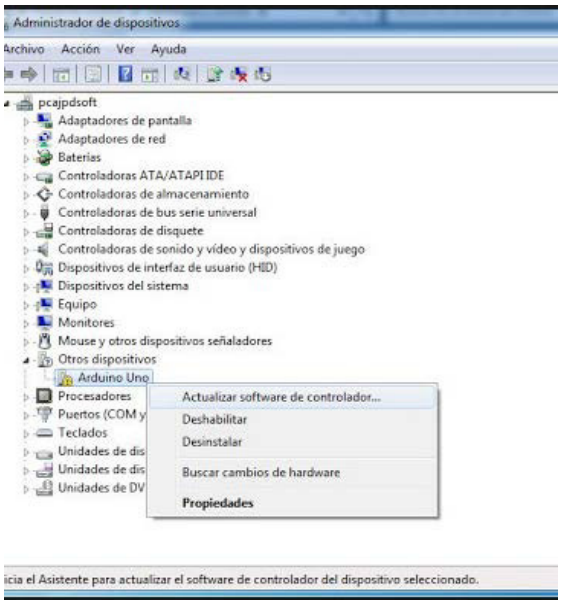


Figura 3.2. Puerto serial en PC

El primer paso para comprobar que todo lo que hemos hecho hasta ahora esté bien y familiarizarnos con la interfaz de desarrollo, se debe abrir un ejemplo. Se recomienda abrir el ejemplo “Blink”. Para ello debemos acceder a través del menú Archivo → Ejemplos → 0.1basic → Blink (Fig. 3.3).

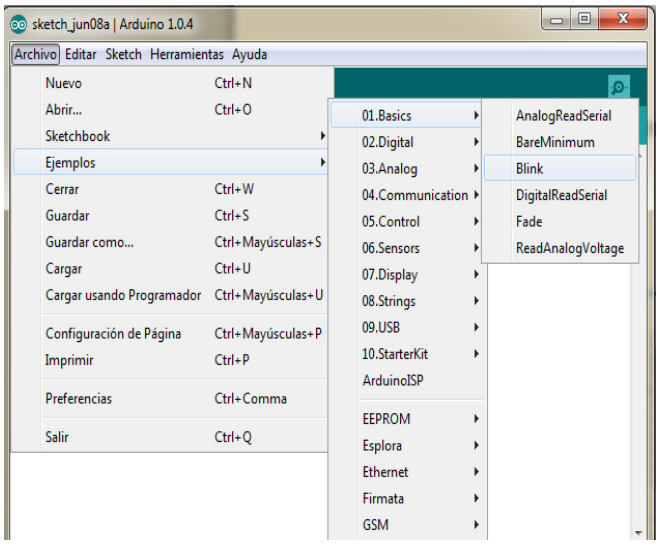


Figura 3.3. Puerto serial

3.1.2. Ejemplo BLINK

El funcionamiento del ejemplo “Blink” el cual hará parpadear un LED que está colocado en el pin número 13 de la placa. Lo primero que se debe realizar es la comprobación del código fuente y así comprender el funcionamiento del ejemplo. Luego pulsamos el botón de verificación indicado en la (Fig. 3.4). Veremos aparecer un mensaje en la parte inferior de la interfaz indicando “Done compiling”. Una vez que el código ha sido verificado procederemos a cargarlo en la placa. Para ello tenemos que pulsar el boton de reset de la placa (Fig. 3.5). E

inmediatamente después pulsar el botón que comienza la carga (Fig. 3.6).

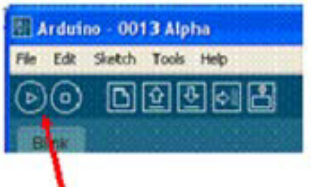


Figura 3.4. Verificación

Figura 3.5. Botón Reset

Figura 3.6. Cargar

Mediante el proceso en el que se va cargando el programa, se encenderán los LEDs TX/RX (Fig 3.7) los cuales indican que los datos han sido enviados y recibido por el puerto serie. Si todo se ha realizado correctamente debe aparecer el mensaje “done uploading”. Ahora solo queda esperar unos 8 segundos aproximadamente para comprobar que el LED colocado en el pin 13 de la placa se enciende y se apaga cada segundo entonces el ejemplo se ha cargado correctamente.

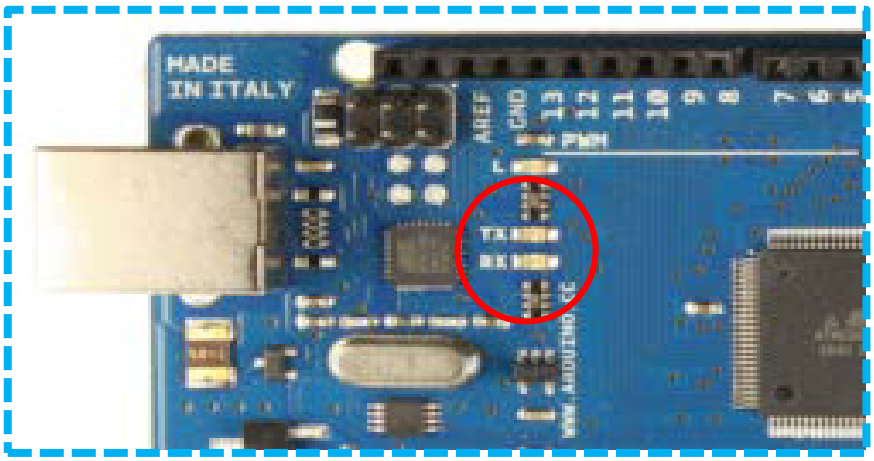


Figura 3.7. LEDs TX- RX

3.1.3. Diagrama de bloques

Figura 3.8. Diagrama de bloques del ejemplo ¨Blink

3.1.4. Diagrama ASM

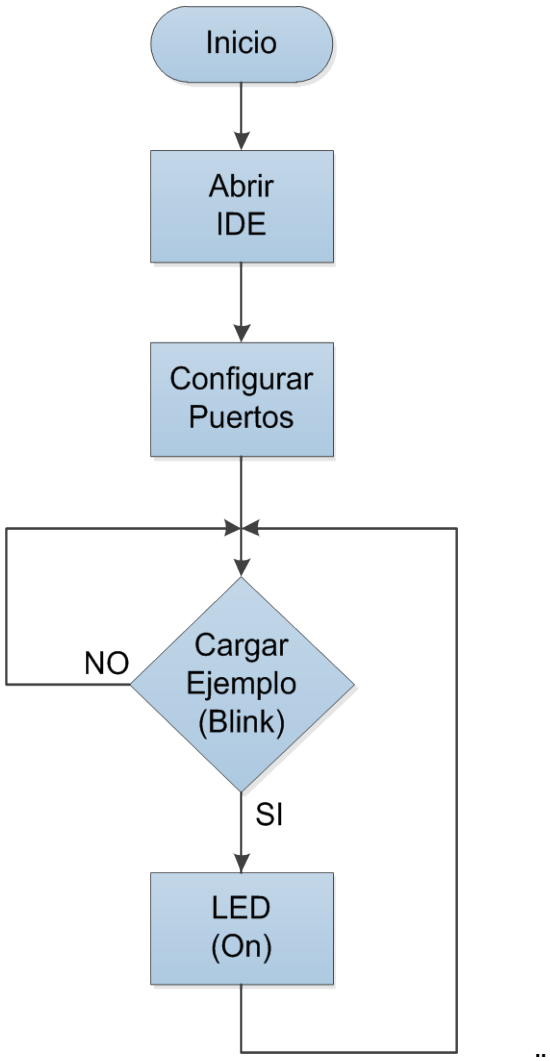


Figura 3.9. Diagrama de flujo para cargar ejemplo ¨Blink¨

3.1.5. Descripción del algoritmo

- Abrir el entorno de desarrollo (IDE)

- Conectar la tarjeta al PC
- Configurar IDE (seleccionar puerto)
- Cargar ejemplo (BLINK)
- Comprobar parpadeo del LED (pin 13)

3.1.6. Código Fuente

```
/* Blink
int led = 13;
void setup() {
  pinMode(led, OUTPUT);
}
void loop() {
  digitalWrite(led, HIGH);          / / enciende el LED ( nivel alto
de tensión)
delay(1000/ / esperar un segundo
  digitalWrite(led, LOW);           / / apagar el LED ( nivel bajo
de tensión)
  delay(1000);                      / / esperar un segundo
}
```

3.2. Control de Intensidad de Luz mediante PWM

3.2.1. Descripción

En este ejemplo se desarrollara el control de intensidad de luz por medio de PWM, simulando el encendido mediante la representación de 4 led's de diferentes colores, los cuales cambiarán su intensidad de iluminación en un período determinado de tiempo [20].

Se conectarán los LEDs con el ánodo de cada uno de los pines digitales 2, 4, 6 y 8 a través de una resistencia de 330Ω cada uno. Tomamos un cable de puente desde GND de la Arduino para el carril de tierra en la parte superior de la placa de prueba; un cable de tierra va desde el cátodo de cada LED para el carril de tierra común a través de una resistencia.

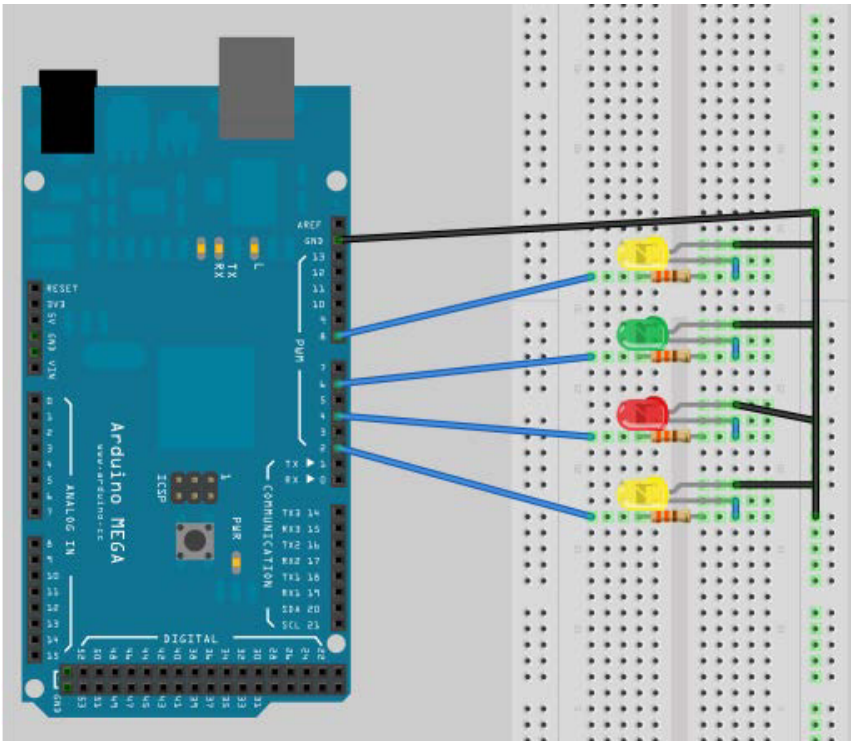


Figura 3.10. Esquema de conexiones

3.2.2. Diagrama de bloques

Figura 3.11. Diagrama de bloques del control de intensidad de luz mediante PWM

3.2.3. Diagrama ASM

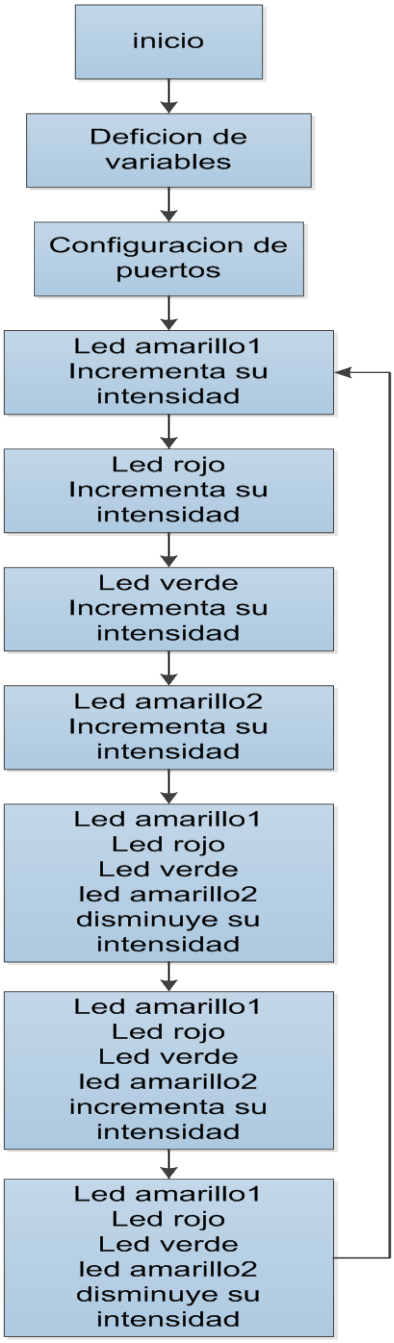


Figura 3.12. Diagrama de bloques del control de intensidad de luz mediante pwm

3.2.4. Descripción del algoritmo

- Se inicializan las variables.
- Se configuran los puertos referente a la variables que se estas utilizando.
- Se ejecuta el programa:
    - Se habilita el puerto 2, enviando una señal de PWM a led amarillo 1 incrementando hasta llegar su máxima intensidad luminosa.
    - Se habilita el puerto 4, enviando una señal de PWM a led rojo incrementando hasta llegar su máxima intensidad luminosa.
    - Se habilita el puerto 6, enviando una señal de PWM a led verde incrementando hasta llegar su máxima intensidad luminosa.
    - Se habilita el puerto 8, enviando una señal de PWM a led amarillo 2 incrementando hasta llegar su máxima intensidad luminosa.
    - Después que todos los leds hayan llegado hasta su máxima intensidad, comenzará a disminuir su intensidad hasta su apagado total.

- Con todos los leds apagado comienzan otra vez a incrementar su intensidad todos al mismo tiempo.
- Después que todos los leds hayan llegado hasta su máxima intensidad, comenzará a disminuir su intensidad hasta su apagado total.
- El procedimiento se volverá a repetirse en el lapso infinito.

3.2.5. Código Fuente

```
//        ERIC MARÍN MONCADA
//       CÉSAR MARÍN MONCADA
//        REALIZACIÓN DE EJERCICIO UTILIZANDO
//        PWM CON LED

//        UTILIZACIÓN DEL PWM PARA CONTROLAR
//        LA INTENSIDAD DE LUZ DE UN ELEMENTO
//        EN ESTE CASO MEDIANTE LOS LED.

//        DECLARACIONES DE VARIABLES QUE USAREMOS
//        EN EL SEGUNDO EJEMPLO DE PRÁCTICA

const int ledamarillo1     = 2;              //  DECLARAMOS VARIABLES
COMO LOS LEDS QUE SE
const int ledrojo          = 4;              //  ESTÁN UTILIZANDO EN EL
PROTO PARA LA
const int ledverde         = 6;             //  SIMULACIÓN.
const int ledamarillo2     = 8;              //
const int tiempo           = 50;            //  tiempo es el delay del programa
 //   ARREGLOS DE NUMEROS DEL 0.....255
 //   QUE SIMULARÁN COMO EL PORCENTAJE DE LOS
 //   CICLOS DE TRABAJO DEL PWM.
const int steps[] = {
0,1,2,3,4,5,6,7,8,9,10,
11,12,13,14,15,16,17,18,19,20,
21,22,23,24,25,26,27,28,29,30,
```

```
31,32,33,34,35,36,37,38,39,40,
41,42,43,44,45,46,47,48,49,50,
51,52,53,54,55,56,57,58,59,60,
61,62,63,64,65,66,67,68,69,70,
71,72,73,74,75,76,77,78,79,80,
81,82,83,84,85,86,87,88,89,90,
91,92,93,94,95,96,97,98,99,100,
101,102,103,104,105,106,107,108,109,110,
111,112,113,114,115,116,117,118,119,120,
121,122,123,124,125,126,127,128,129,130,
131,132,133,134,135,136,137,138,139,140,
141,142,143,144,145,146,147,148,149,150,
151,152,153,154,155,156,157,158,159,160,
161,162,163,164,165,166,167,168,169,170,
171,172,173,174,175,176,177,178,179,180,
181,182,183,184,185,186,187,188,189,190,
191,192,193,194,195,196,197,198,199,200,
201,202,203,204,205,206,207,208,209,210,
211,212,213,214,215,216,217,218,219,220,
221,222,223,224,225,226,227,228,229,230,
231,232,233,234,235,236,237,238,239,240,
241,242,243,244,245,246,247,248,249,250,
251,252,253,254,255 };
//
//   EN VOID SETUP SE DECLARARÁN LAS ENTRADAS
//       Y SALIDAS DE LOS PUERTOS PWM
//
 void setup()
 {
  pinMode(ledamarillo1, OUTPUT);         // ledamarillo1 = 2;
  pinMode(ledrojo, OUTPUT);             // ledverde     = 4;
  pinMode(ledverde, OUTPUT);              // ledrojo      = 6;
  pinMode(ledamarillo2, OUTPUT);         // ledamarillo2 = 8;

  }
//
//  PROGRAMA PRINCIPAL Y EJECUCION DE LAS
//       RUTINAS DEL EJERCICIO
//
 void loop() {

   for (int pos = 0; pos < 255; pos++)
 {
   analogWrite(ledamarillo1, steps[pos]);
   delay(tiempo);
 }
```

```
   for (int pos = 0; pos < 255; pos++)
{
  analogWrite(ledrojo, steps[pos]);
  delay(tiempo);
}
   for (int pos = 0; pos < 255; pos++)
{
  analogWrite(ledverde, steps[pos]);
  delay(tiempo);
}
   for (int pos = 0; pos < 255; pos++)
{
  analogWrite(ledamarillo2, steps[pos]);
  delay(tiempo);
}
 for (int pos = 255; pos > 0 ; pos--)
 {
  analogWrite(ledamarillo1, steps[pos]);
  analogWrite(ledrojo, steps[pos]);
  analogWrite(ledverde, steps[pos]);
  analogWrite(ledamarillo2, steps[pos]);
  delay(tiempo);
}
   for (int pos = 0; pos < 255; pos++)
{
  analogWrite(ledamarillo1, steps[pos]);
  analogWrite(ledrojo, steps[pos]);
  analogWrite(ledverde, steps[pos]);
  analogWrite(ledamarillo2, steps[pos]);
  delay(tiempo);
}
  for (int pos = 255; pos > 0 ; pos--)
 {
  analogWrite(ledamarillo1, steps[pos]);
  analogWrite(ledrojo, steps[pos]);
  analogWrite(ledverde, steps[pos]);
  analogWrite(ledamarillo2, steps[pos]);
  delay(tiempo);
 }
}
```

3.3. Control de Posición de Motores de Paso mediante Driver L293D

3.3.1. Descripción

En este ejemplo vamos a ver el Control de dirección de giro de dos motores paso a paso, que simulará una aplicación real de una banda transportadora y selladora de botellas, para lo cual utilizamos el driver de potencia basado en un L293D. El funcionamiento es sencillo si le das al pulsador se iniciara la secuencia, el motor 1 simulara el movimiento de una banda transportadora y el motor 2 simulará el equipo de selladora de tapas de botellas [21].

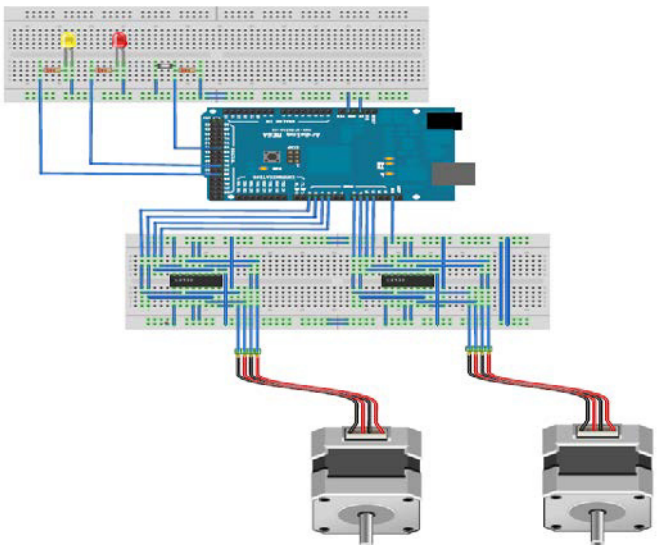


Figura 3.13. Esquema de implementación del control de posición de motores de paso mediante driver L293D

3.3.2. Diagrama de bloques

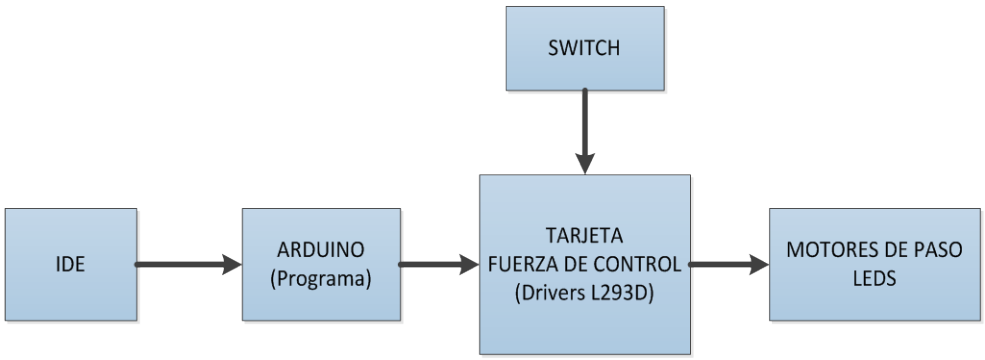


Figura 3.14. Diagrama de bloques del control de posición de motores

3.3.3. Diagrama ASM

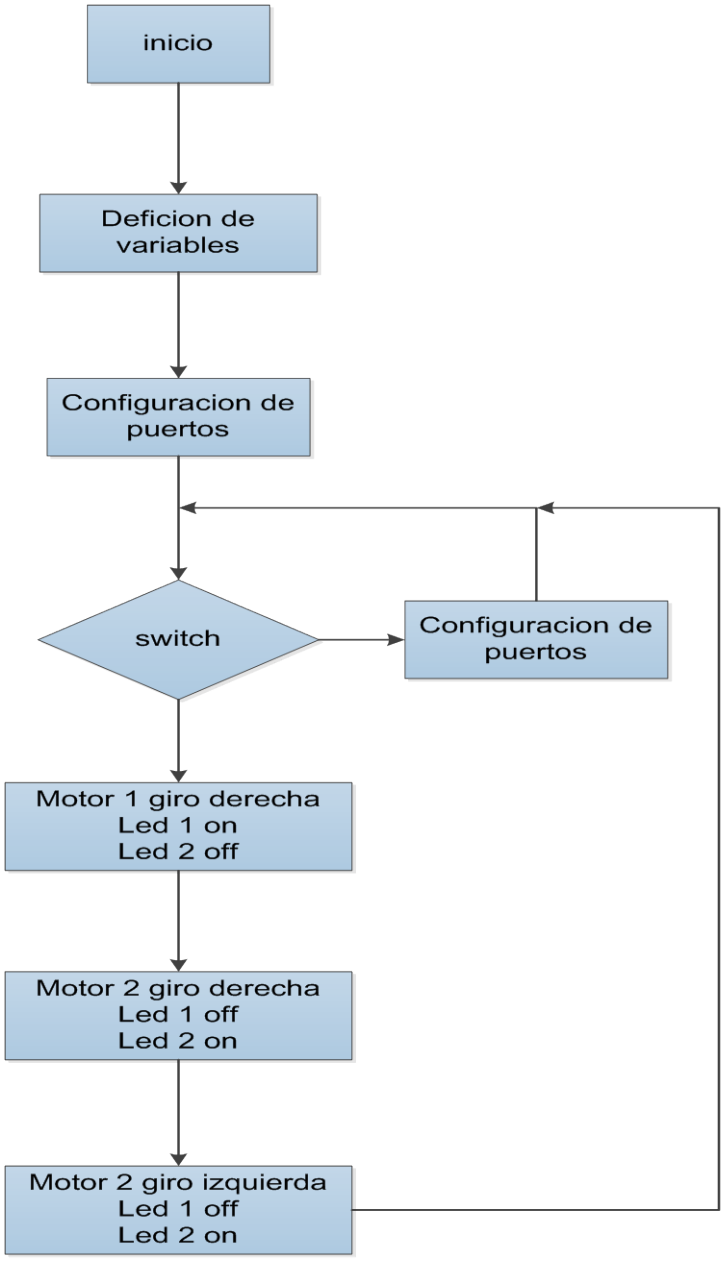


Figura 3.15. Diagrama de flujo del ejemplo control de posición de motores

3.3.4. Descripción del algoritmo

- Se inicializan las variables.
- Se configuran los puertos referente a la variables que se estas utilizando.
- Se ejecuta el programa
    - El programa espera la condición del SWITCH si se encuentra en on u off.
    - SWITCH (on) el primer motor de paso se moverá a una distancia determinada con diez vueltas a su vez se activa el led 1 indicando su activación,
    - Después que el primer motor se haya detenido se moverá el segundo motor tres vueltas hacia la derecha y luego tres vueltas hacia la izquierda y su vez se activará el led 2 indicando su activación.
    - El proceso se repetirá continuamente hasta que el SWITCH cambien de estado en (off), mantendrá todas las salidas en bajo.

3.3.5. Código Fuente

```
//      ERIC MARÍN MONCADA
//      CÉSAR MARÍN MONCADA
//      REALIZACIÓN DE EJERCICIO
//      CONTROL DE DOS MOTORES DE PASO
```

```
//        DECLARACIONES DE VARIABLES QUE USAREMOS
//        EN EL TERCER EJEMPLO DE PRÁCTICA

int switch1=42; int boton1=38;
int led1=34;    int led2=30;
int A=2; int B=3; int C=4; int D=5;        // VARIABLES PARA LA
BOBINAS DEL PRIMER MOTOR DE PASO
int E=8; int F=9; int G=10; int H=11;     // VARIABLES PARA LA
BOBINAS DEL SEGUNDO MOTOR DE PASO
int mSecPaso=1500;                         // VARIABLE DE TIEMPO
DE CONMUTACIÓN

//
//        EN VOID SETUP SE DECLARARÁN LAS ENTRADAS
//        Y SALIDAS DE LOS PUERTOS
//

void setup()
{
//        VARIABLES DE ENTRADAS Y SALIDAS
//        PARA CONMUTADORES Y INDICADORES
pinMode(switch1, INPUT);
pinMode(boton1, INPUT);
pinMode(led1, OUTPUT);
pinMode(led2, OUTPUT);
// VARIABLES DE BOBINAS DE MOTOR 1
pinMode(A, OUTPUT);
pinMode(B, OUTPUT);
pinMode(C, OUTPUT);
pinMode(D, OUTPUT);
//        INICIALIZACIÓN DE BOBINAS
//        DE MOTOR 1 EN BAJO
digitalWrite(A, LOW);
digitalWrite(B, LOW);
digitalWrite(C, LOW);
digitalWrite(D, LOW);
//        VARIABLES DE BOBINAS DE MOTOR 2
pinMode(E, OUTPUT);
pinMode(F, OUTPUT);
pinMode(G, OUTPUT);
pinMode(H, OUTPUT);
//        INICIALIZACIÓN DE BOBINAS
//        DE MOTOR 2 EN BAJO
digitalWrite(E, LOW);
digitalWrite(F, LOW);
```

```
digitalWrite(G, LOW);
digitalWrite(H, LOW);
}

//        PROGRAMA DE GIRO DE MOTOR 1
//        HACIA LA IZQUIERDA
void izquierdamotor1(){
//paso 1
digitalWrite(A, LOW);
digitalWrite(B, HIGH);
digitalWrite(C, HIGH);
digitalWrite(D, LOW);
delayMicroseconds(mSecPaso);
//paso 2
digitalWrite(C, LOW);
digitalWrite(D, HIGH);
delayMicroseconds(mSecPaso);
//paso 3
digitalWrite(A, HIGH);
digitalWrite(B, LOW);
delayMicroseconds(mSecPaso);
//paso 4
digitalWrite(C, HIGH);
digitalWrite(D, LOW);
delayMicroseconds(mSecPaso);
}//FIN izquierda

//        PROGRAMA DE GIRO DE MOTOR 1
//        HACIA LA DERECHA
void derechamotor1(){
//paso 1
digitalWrite(A, HIGH);
digitalWrite(B, LOW);
digitalWrite(C, HIGH);
digitalWrite(D, LOW);
delayMicroseconds(mSecPaso);
//paso 2
digitalWrite(C, LOW);
digitalWrite(D, HIGH);
delayMicroseconds(mSecPaso);
//paso 3
digitalWrite(A, LOW);
digitalWrite(B, HIGH);
delayMicroseconds(mSecPaso);
//paso 4
digitalWrite(C, HIGH);
```

```
digitalWrite(D, LOW);
delayMicroseconds(mSecPaso);
}//FIN derecha

//        PROGRAMA DE GIRO DE MOTOR 2
//        HACIA LA IZQUIERDA
void izquierdamotor2(){
//paso 1
digitalWrite(E, LOW);
digitalWrite(F, HIGH);
digitalWrite(G, HIGH);
digitalWrite(H, LOW);
delayMicroseconds(mSecPaso);
//paso 2
digitalWrite(G, LOW);
digitalWrite(H, HIGH);
delayMicroseconds(mSecPaso);
//paso 3
digitalWrite(E, HIGH);
digitalWrite(F, LOW);
delayMicroseconds(mSecPaso);
//paso 4
digitalWrite(G, HIGH);
digitalWrite(H, LOW);
delayMicroseconds(mSecPaso);
}//FIN izquierda

//        PROGRAMA DE GIRO DE MOTOR 2
//        HACIA LA DERECHA
void derechamotor2(){
//paso 1
digitalWrite(E, HIGH);
digitalWrite(F, LOW);
digitalWrite(G, HIGH);
digitalWrite(H, LOW);
delayMicroseconds(mSecPaso);
//paso 2
digitalWrite(G, LOW);
digitalWrite(H, HIGH);
delayMicroseconds(mSecPaso);
//paso 3
digitalWrite(E, LOW);
digitalWrite(F, HIGH);
delayMicroseconds(mSecPaso);
//paso 4
digitalWrite(G, HIGH);
```

```
digitalWrite(H, LOW);
delayMicroseconds(mSecPaso);
}//FIN derecha

//
//        PROGRAMA PRINCIPAL Y EJECUCION DE LAS
//        RUTINAS DEL EJERCICIO
void loop()
{
  //      CONDICIÓN PARA LA INICIALIZACION
  //      DEL PROGRAMA
  if (digitalRead(switch1)==LOW)
  {
    for(int i=0; i<1000; i++)
    {
    digitalWrite(led1,HIGH); // ENCIENDE EL LED 1
    digitalWrite(led2,LOW);  // MANTIENE APAGADO EL LED 2
    derechamotor1();              // PROGRAMA DEL MOTOR 1 GIRA
HACIA LA DERECHA
    }
    for(int i=0; i<300; i++)
    {
    digitalWrite(led2,HIGH); // ENCIENDE EL LED 2
    digitalWrite(led1,LOW);  // APAGA EL LED 1
    derechamotor2();              // PROGRAMA DEL MOTOR 2 GIRA
HACIA LA DERECHA
    }
    for(int i=0; i<300; i++)
    {
    digitalWrite(led2,HIGH); // ENCIENDE EL LED 2
    digitalWrite(led1,LOW);  // MANTIENE APAGADO EL LED 1
    izquierdamotor2();            // PROGRAMA DEL MOTOR 2 GIRA
HACIA LA IZQUIERDA
    }
  }
  else
  {
    //SWITCH NO ES ACTIVADO
    digitalWrite(led1,LOW);  //APAGA EL LED 1
    digitalWrite(led2,LOW);  //APAGA EL LED 2
    //SALIDAS DEL MOTOR DE PASO 1 EN BAJO
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
    digitalWrite(D, LOW);
    //SALIDAS DEL MOTOR DE PASO 2 EN BAJO
```

```
    digitalWrite(E, LOW);
    digitalWrite(F, LOW);
    digitalWrite(G, LOW);
    digitalWrite(H, LOW);

  }
}
```

3.4. Control de Posición de Cabezal de Impresión de Tres Dimensiones

La realización de este proyecto se basa en el control y manejo de los motores de paso de la impresora 3D, con el propósito de crear objetos tridimensionales, mediante la superposición de capas de material PLA o ABS empleando un sistema con Arduino y Pronterface que es un interfaz de visualización e iteración con el usuario [10].

Para implementación del proyecto se hará uso de un programa que se cargara en el arduino, el cual por medio del interfaz que permite la transmisión y recepción de datos para el control de los diferentes motores de la impresora Prusa, es necesario conocer el funcionamiento del sistema de impresión 3D, por lo cual se explicará en forma detallada y estructurada el proceso para generar un pieza utilizando nuestra tarjeta arduino [12].

3.4.1. Descripción

Haciendo uso del kit Prusa DIY serie V06-0702 se procedió al ensamblado de la impresora 3D, perteneciente a la línea de impresoras STUFFMAKER, se realizaron cambios necesarios en la impresora para la adaptación de la tarjeta que utilizáremos en la implementación del proyecto, en el diagrama de bloques figura 3.14 se observa las partes que está conformada la impresora tales como: motores, sensores, tarjetas, drivers y el extrusor [22].

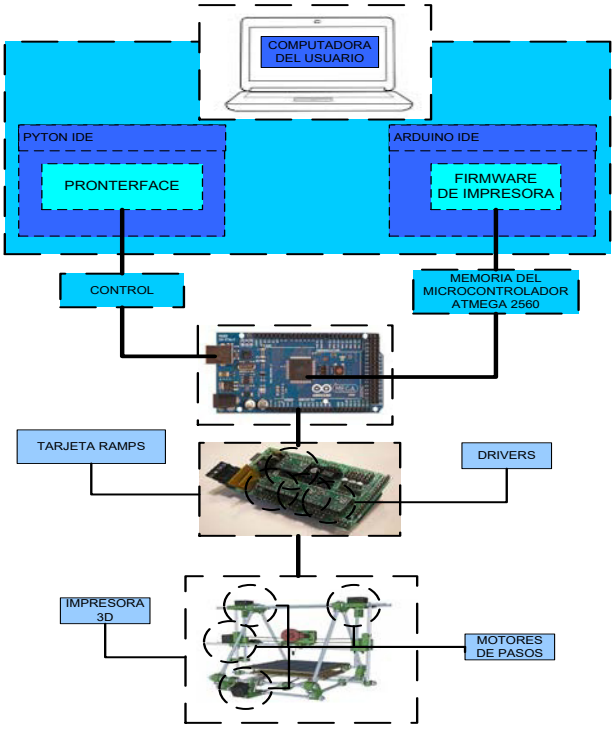


Figura 3.16. Partes de la impresora Prusa

El hardware está estructurada de la siguiente forma, en el primer plano la computadora donde se instalara el entorno de desarrollo en el cual se podrá configurar y modificar el programa antes de la instalación en la tarjeta , tambien se instalara pronterface, que es el software de interacion con el usuario y controlador de nuestro sistema.

En la parte intermedia del hardware se encuentra constituida por las tarjetas arduino, board ramps, y los driver. La tarjeta arduino contiene el controlador atmega2560, el cual hara lectura de los diferentes sensores y activará los dispositivos necesario para la impresión. Las board ramps es un tarjeta modular donde se encuentra la parte de fuerza, y adaptación de las drivers [23].

Finamente tenemos la impresora que esta constituida por el motor del extrusor que es el encargado de proveer de material al cabezal de impresion, los motores de los ejes (x,y,z) los cuales daran los movimiento del cabezal en el transcurso de impresion, los sensores fin de carrera y la mesa de impresión.

Los primeros pasos en el desarrollo del proyecto es la instalación del entorno de desarrollo integrado (IDE) de arduino 1.1 en la PC, luego cargamos el firmware (Marlin) y se procede

hacer los cambios en la configuración para adaptar la tarjeta arduino al sistemas, unos de los cambio se basó en la velocidad de comunicación que se tiene en el pronterface y el firmware, por las pruebas realizadas de comunicación e impresión, se aconseja utilizar la máxima velocidad de comunicación de transmisión de datos de 250000 **BAUDRATE**.

 # Define BAUDRATE 250000

// Define BAUDRATE 115200

Por requerimientos de la tarjeta se procedió hacer el cambio de los sensores fin de carrera ópticos por los sensores mecánicos, debido la diferencia de lógica de los sensores en el programa, se procede al cambio de la logica TRUE por FALSE en definición de variables para los finales de carrera.

```
Const bool X_ENDSTOP_INVERT = false;
Const bool Y_ENDSTOP_INVERT = false;
Const bool Z_ENDSTOP_INVERT = false;
```

Con los cambios realizados procedemos a cargar el firmware en el arduino como se describió en el capítulo anterior ver fig. 3.17, luego procedemos a comunicar el pronterface con el arduino mega ver fig. 3,18, para verificar si está funcionando

correctamente, se accede a las aplicaciones de pronterface y verificamos los movimientos del cabezal en los ejes (x,y,z), el funcionamiento del extrusor y la temperatura del cabezal.

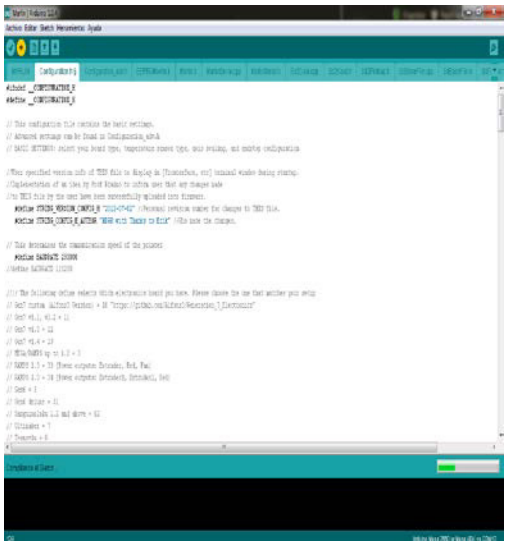


Figura 3.17. firmware cargado en el entorno de desarrollo

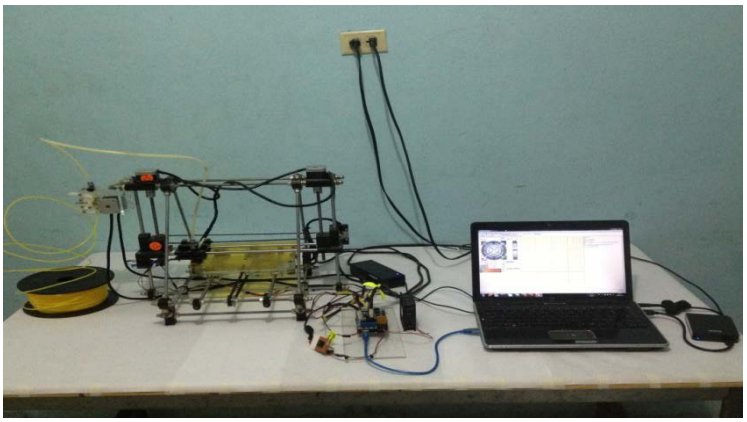


Figura 3.18. Comunicación Arduino-Pc

Comenzaremos con las conexiones de la impresora a la tarjeta, y luego probaremos y calibraremos los drivers cuando terminemos con las conexiones.

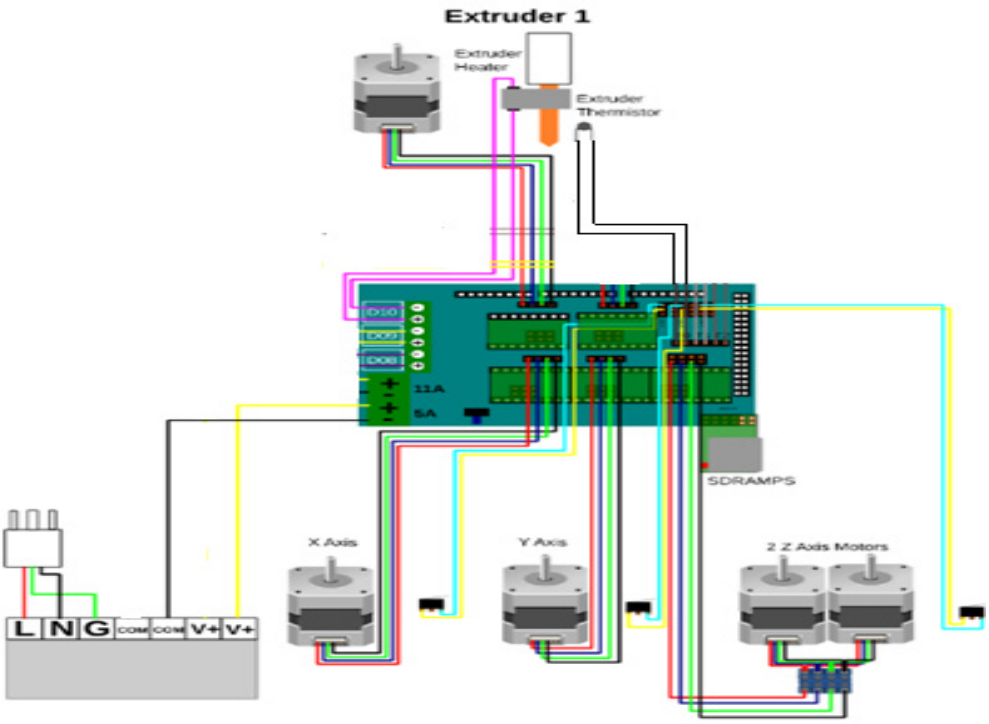


Figura 3.19. Diagrama de conexiones con la tarjeta Ramps

En las conexiones comenzaremos con el eje X con su respectivo sensor de fin de carrera, después el eje Y y por consiguiente el eje Z. luego de finalizar las conexiones de los ejes sigue el motor que envía material a la extrusora.

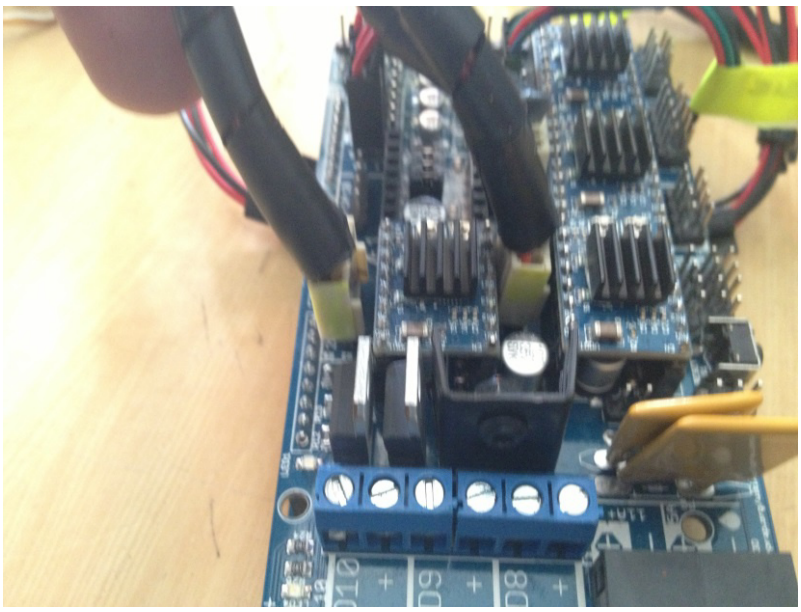


Figura 3.20. Conexión de los motores de paso

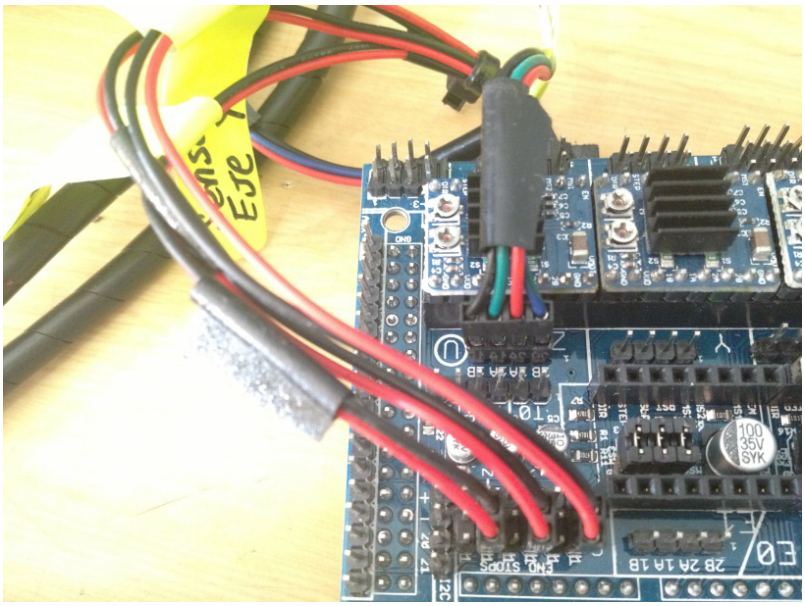


Figura 3.21. conexión de los sensores fin de carrera

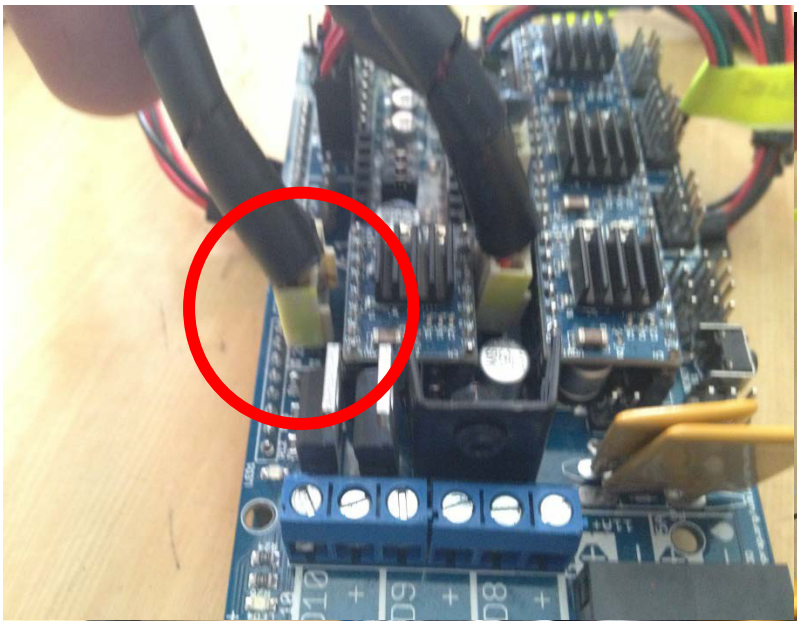


Figura 3.22. Conexión del motor extrusor

Finalmente terminaremos con la conexión de la extrusora. Es importante porque la extrusora consta de 4 cables, 2 cables para la resistencia que derretirá el material y 2 cables más para el sensor de temperatura.

3.4.2. Diagrama de bloques

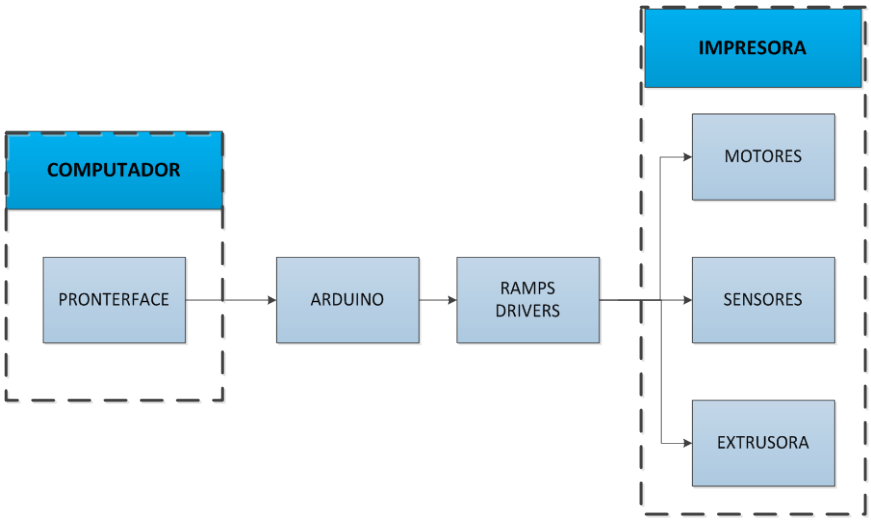


Figura 3.23. Diagrama de la implementación de la impresora

3.4.3. Diagrama ASM

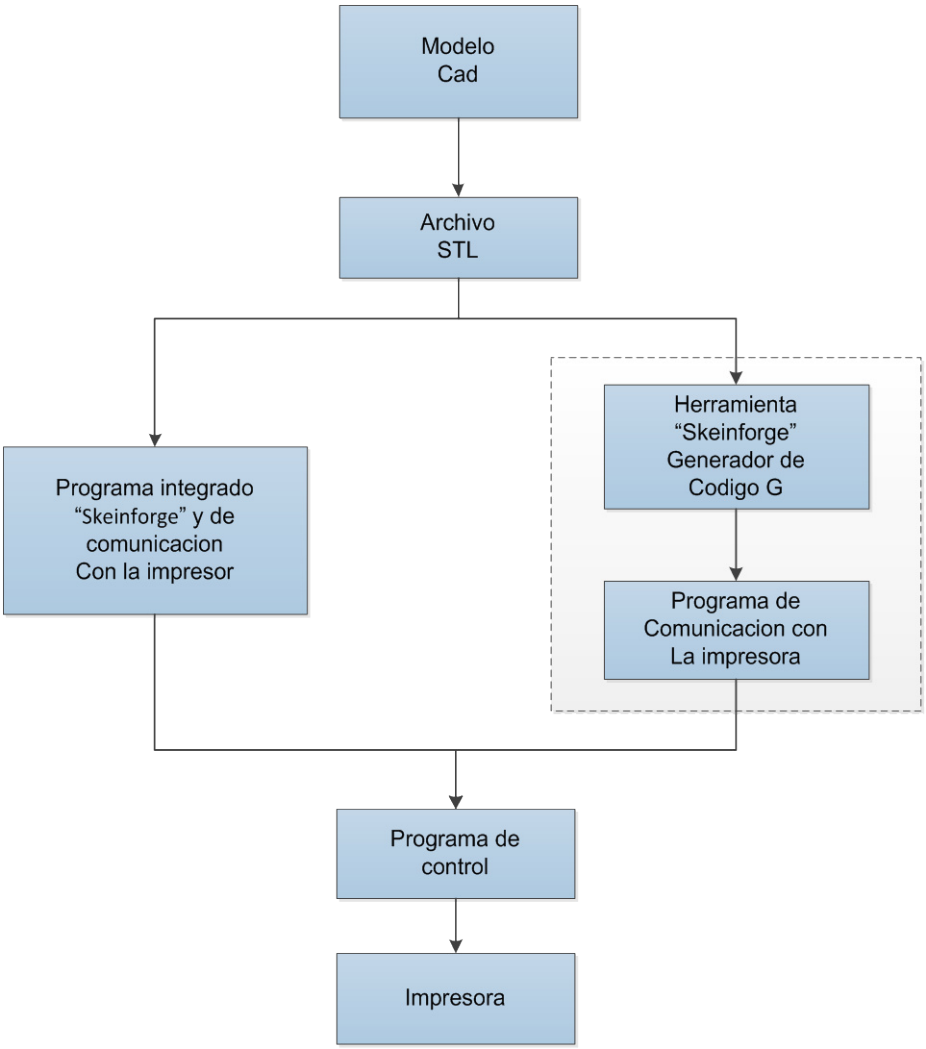


Figura 3.24. Diagrama bloques para la generación de los comandos de control.

3.4.4. Descripción del algoritmo

- Diseño 3d con software, realizados en Autocad, solidEdge, Solidworks, etc.
- Se guarda o convierte el archivo en formato *.STL.
- Generar las instrucciones en código G, utilizando programas skeinforge, slic3r, Kisslicer, etc.
- Cargar las instrucciones en el programa de comunicación y control Pronterface.
- Transmitir los datos al controlador ATM2560 de la tarjeta Arduino.

CAPÍTULO 4

4. IMPLEMENTACIÓN EJERCICIOS DE PRUEBA Y SIMULACIÓN DEL PROYECTO

En este capítulo se muestra los diagramas de conexiones de cada uno de los ejercicios realizados y detallados en el capítulo anterior, se presenta el listado de materiales necesarios para cada ejercicio con su respectiva ilustración gráfica.

Las simulaciones correspondientes de cada ejercicio también forman parte de este capítulo, con el objetivo de observar que la transmisión de datos mediante comunicación del Arduino y la Pc sea realizada en forma correcta.

Los diagramas de conexiones son basados en las hojas de datos esquemáticos del microcontrolador ATmega2560 de la plataforma Arduino, estos diagramas han sido diseñados bajo nuestra creatividad luego de estudiar físicamente e identificar las entradas y salida que cuenta dicha tarjeta. La simulación y realización de esquemáticos de los ejemplos fue realizada en fritzing, programa diseñado para la elaboración de esquemáticos [10].

4.1. Comunicar la placa Arduino y la Pc

La finalidad de este procedimiento básico, esencial es para involucrarnos en el entorno de arduino, realizando la comunicación de la placa arduino y la pc como se muestra en la figura 4.2 y afianzarnos antes de la implementación del proyecto.

En este ejercicio nuestro principal objetivo es analizar la comunicación entre la tarjeta Arduino Atme2560 y la Pc utilizando el entorno de desarrollo (IDE) versión 1.0.4, para lo cual aprovechamos el LED que está colocado en el pin número 13 de la placa [20].

Para la transmisión hacemos uso del cable USB para conectar la placa y la Pc, el cual nos ayudar a cargar el ejemplo “blink” el ejemplo está diseñado para enviará una señal al pin 13 y gracias al encendido del led de la tarjeta podemos observar que se realiza la comunicación de manera óptima.

4.1.1. Lista de Componentes

- Tarjeta Arduino ATmega2560.
- Cable Adaptador USB.
- Diodo Led integrado en la tarjeta.

4.1.2. Diagrama de conexiones

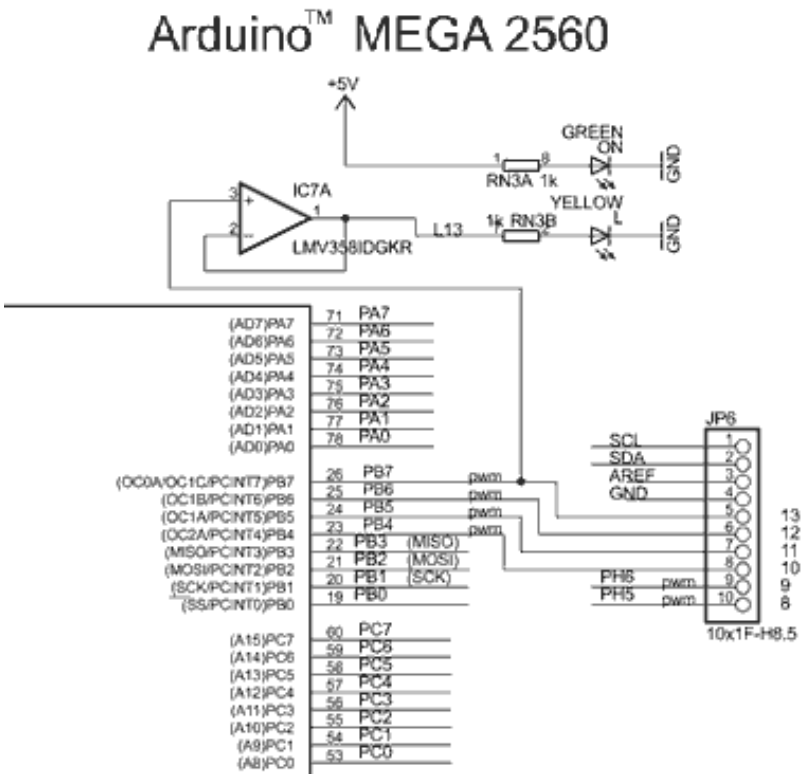


Figura 4.1. Diagrama interno de la tarjeta led pin 13.

4.1.3. Implementación

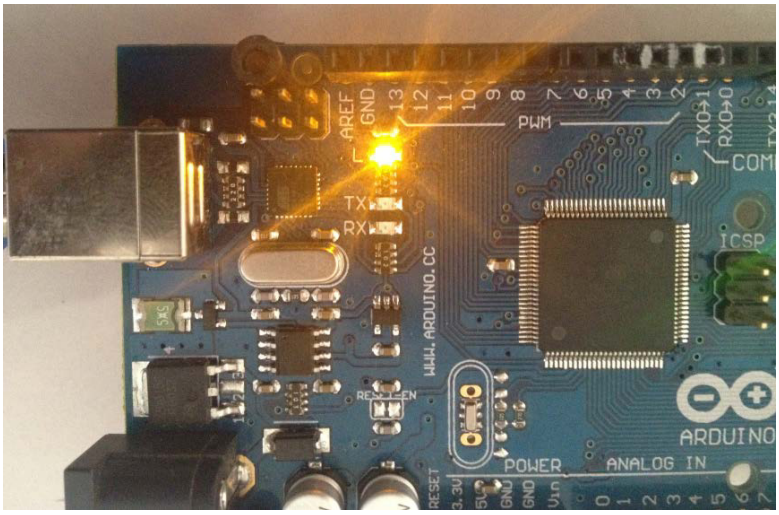


Figura 4.2. Comunicación la placa Arduino y la Pc.

4.1.4. Observaciones

Al realizar la comunicación con la placa de arduino, procedimos a cargar el ejemplo blink y se puede observar que led L del pin 13 comienza a parpadear a razón de un segundo, comprobando la correcta comunicación que se llevó acabo en este procedimiento.

4.2. Control de Intensidad de Luz mediante PWM

El desarrollo de este ejercicio tiene como propósito familiarizarnos con las entradas/salidas digitales que cuenta el Atmega2560 de las cuales emplearemos las salidas digitales que pueden ser utilizadas para PWM de la tarjeta e interactuar con las diferentes funciones que esta nos ofrece.

El funcionamiento consiste en el control de la intensidad luminosa de leds, los cuales aumentarán su intensidad de luz de forma secuencial manteniéndose en la intensidad más alta, luego disminuirán y aumentarán su intensidad para finalmente disminuir su intensidad en el nivel más bajo.

4.2.1. Lista de Componentes

- Tarjeta Arduino ATmega2560
- Cable Adaptador USB.
- 4 Diodos LED’s.
- 4 Resistencias de 330 Ω.
- Cable UTP.

4.2.2. Diagrama de conexiones

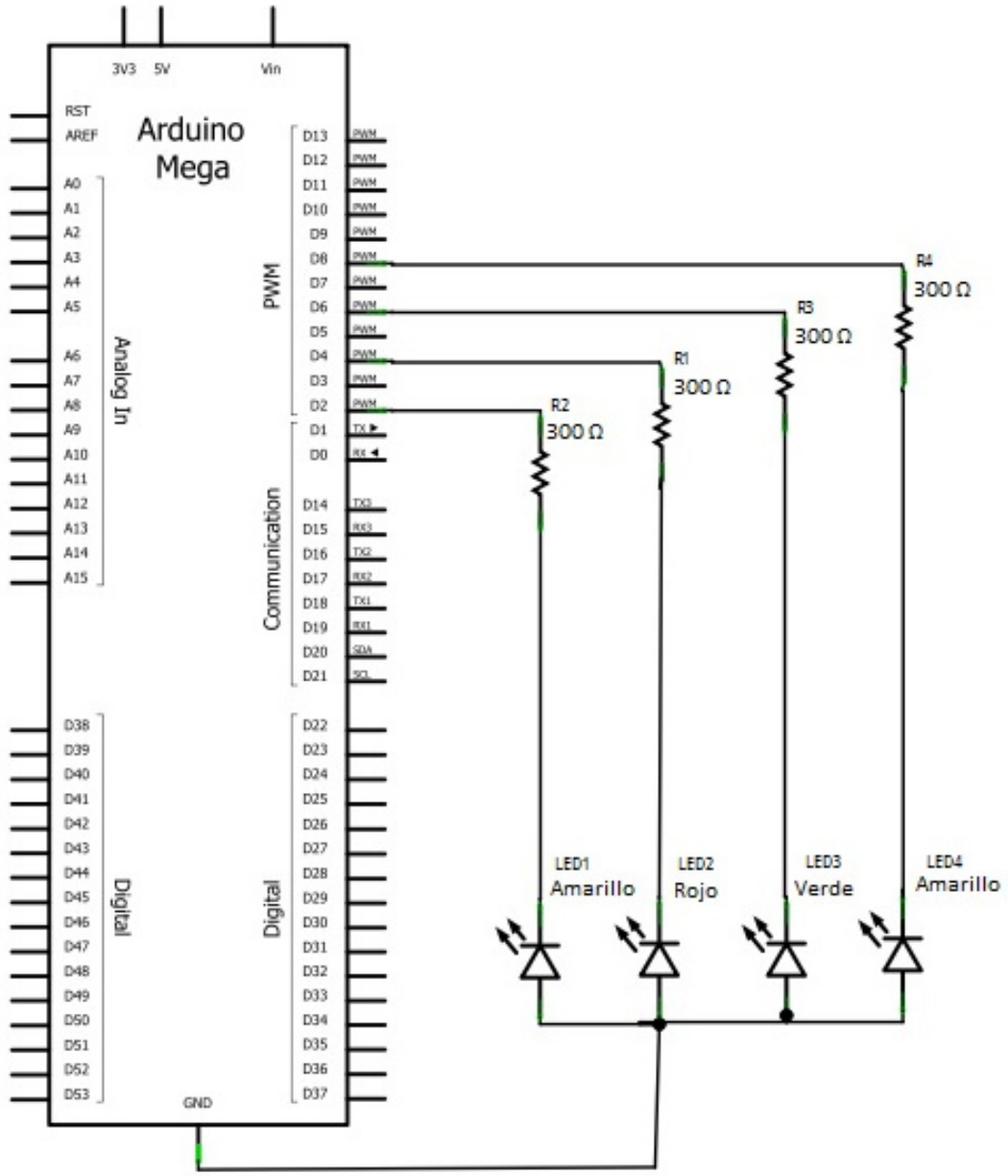


Figura 4.3. Diagrama de conexiones para el control de intensidad de leds usando la tarjeta ATmega2560.

4.2.3. Implementación

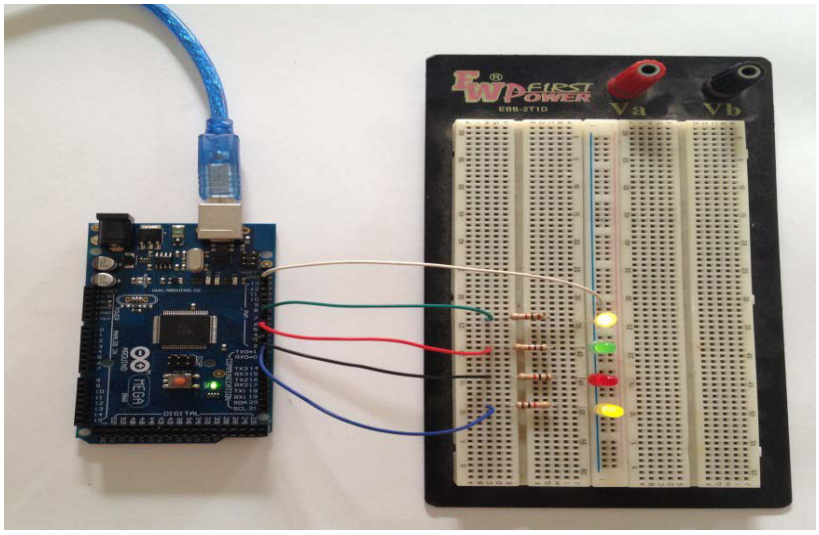


Figura 4.4. Intensidad luminosa de leds mediantes PWM.

4.2.4. Observaciones

Al ejecutar el programa cargado en el microcontrolador, se observa el cambio de intensidad de los leds como se muestra en la figura 4.4 se aprecia un incremento de intensidad hasta su máxima luminosidad y también haciendo y su descenso hasta mínima intensidad en forma individualmente y luego de forma conjunta.

4.3. Control de Posición de Motores de Paso mediante Driver 289D

En este ejercicio el objetivo principal es el control de motores de paso mediante la implementación de 2 motores de paso y 2 leds para la respectiva simulación, haciendo uso de la tarjeta ATMEGA2560 y la tarjeta de fuerza diseñada para los motores [24].

El ejercicio se basa fundamentalmente en la realización de dos movimientos simulando el movimiento de una banda transportadora y una selladora de tapas de botellas; el motor1 simulará el movimiento de la banda transportadora el cual girará un determinado cantidad de grados, luego el motor 2 el equipo sellador de tapas.

4.3.1. Lista de Componentes

- Tarjeta Arduino ATmega2560
- Cable Adaptador USB.
- 2 Diodos LED’s.
- 2 Resistencias de 330 Ω.
- Cable UTP.
- Fuente de 12V.
- 2 Motores de paso.

4.3.2. Diagrama de conexiones

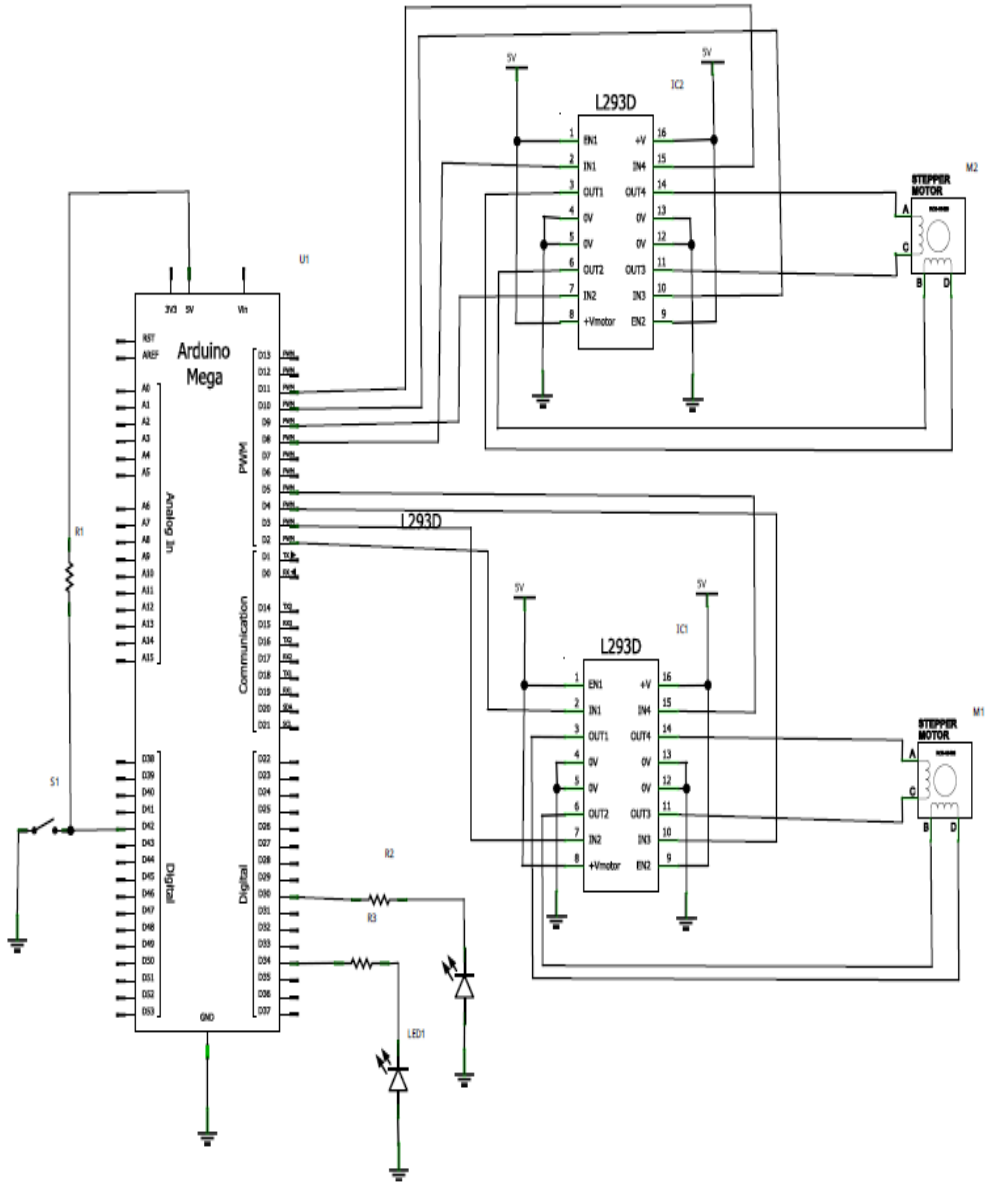


Figura 4.5. Diagrama de conexiones para el control de posición de motores de paso usando la tarjeta ATmega2560.

4.3.3. Implementación

Simulación control de posición de motores de paso

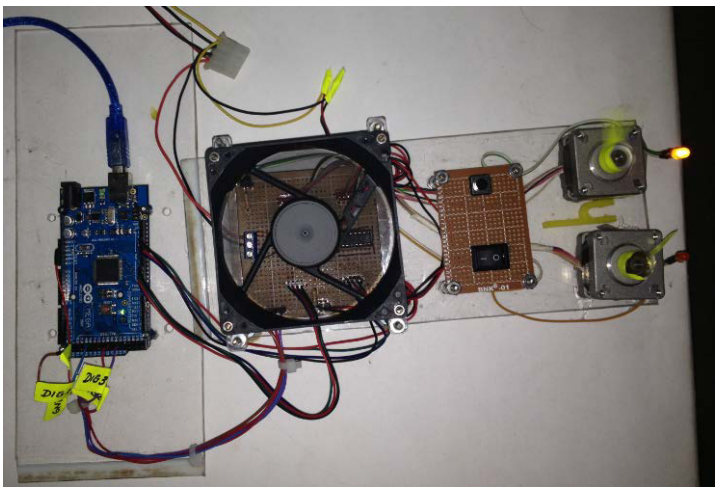


Figura 4.6. Motor que simula el movimiento de una banda transportadora.

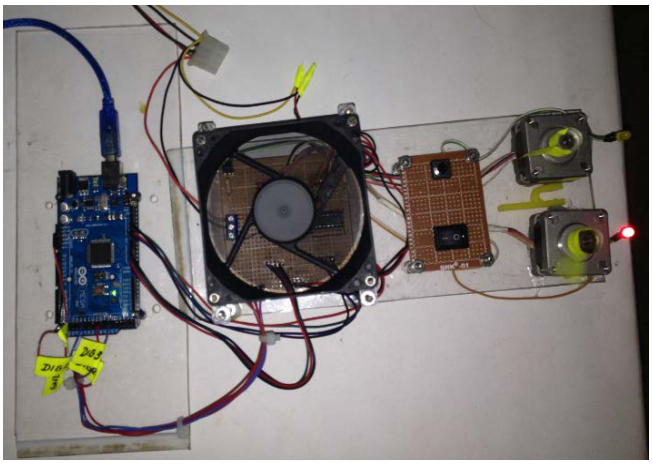


Figura 4.7. Motor que simula el movimiento del equipo selladora de botellas

4.3.4. Observaciones

El comportamiento de este ensayo se muestra las figuras 4.6a y 4.6b, las cuales nos permiten observar la secuencia de control de los dos motores de paso, este tipo de control se basa en la activación de bobinas, motivo por el cual es necesario saber la distribución de los cables a los bobinados establecidos en la hoja de datos de los motores o averiguarlo mediante inspección.

4.4. Control de Cabezal de Impresión para Elaboración de Objetos en Tres Dimensiones

El objetivo principal del proyecto es en lograr el correcto posicionamiento del cabezal de impresión empleando un control en la plataforma arduino, en este capítulo desarrollaremos un ejercicio con la finalidad de familiarizarnos con el entorno del interfaz gráfica de pronterface.

Para el desarrollo del ejercicio es importante obtener la imagen de impresión para porcesarla en el programa para generación del código que es enviado a la tarjeta arduino como se muestra en la figura 4.8.

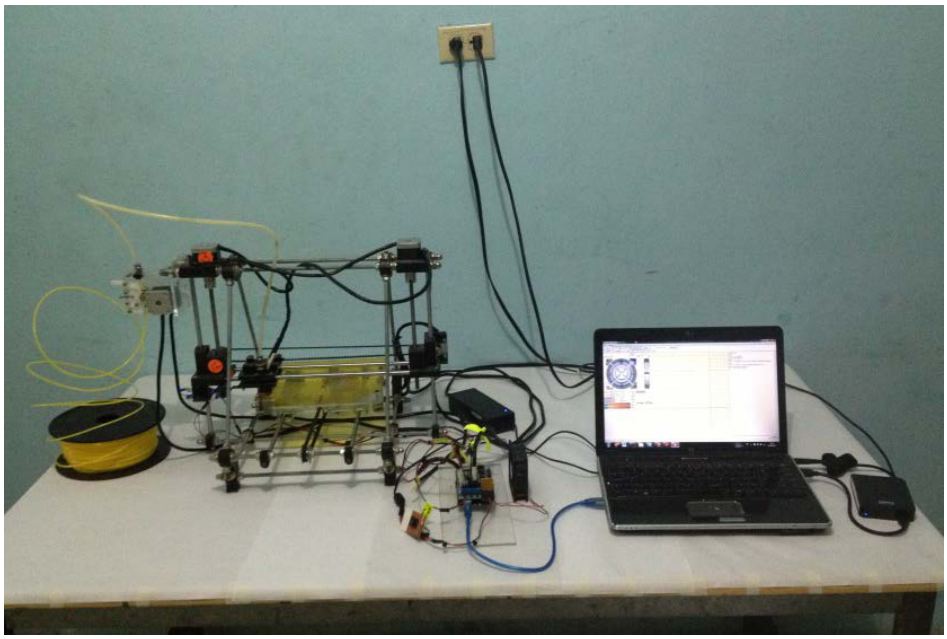


Figura 4.8. Implementación completa del sistema

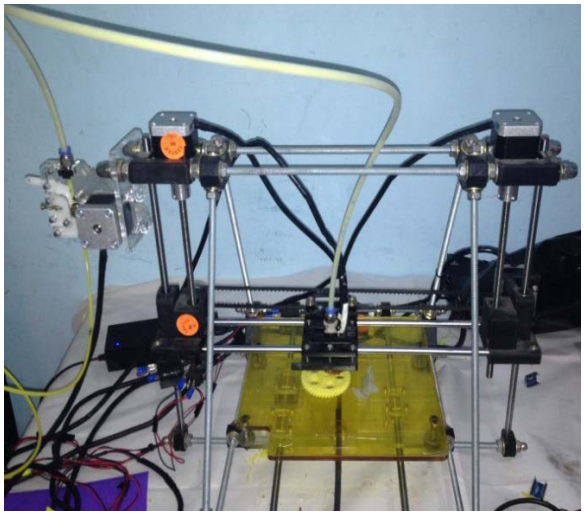


Figura 4.9. Sistema imprimiendo rueda dentada

En proceso de creación de la pieza como se muestra en la figura 4.9 el programa crea un código mediante capas. Y también calcula el tiempo que demorará en hacer la impresión, eso dependerá del acabado que se desee por ejemplo, el relleno, detalle de los bordes internos y externos. Con la finalidad de tener una buena resolución del objeto.

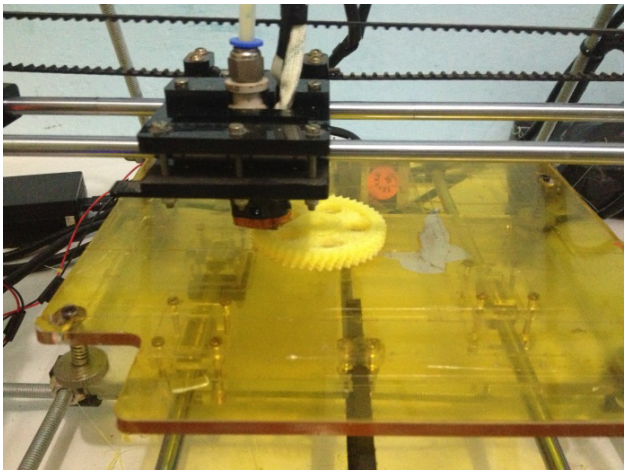


Figura 4.10. Última capa de impresión

La figura 4.10 se llevó alrededor de hora y media en culminar su elaboración, por los detalles que tiene en los bordes internos y externos, hace que reduzca su velocidad y tome un poco más tiempo. Pero tendremos una impresión de gran calidad como se muestra en la figura 4.11.

Figura 4.11. Impresión terminada de la rueda dentada.

CONCLUSIONES

Basado en lo expuesto anteriormente en esta tesina podemos obtener las siguientes conclusiones:

1. La gran mayoría de los datos, mejoras y consejos que hemos propuesto a lo largo de este proyecto han sido fruto de la investigación y experiencia obtenida en estos últimos meses, experiencia que hemos obtenido gracias al trabajo de montaje, impresión, mantenimiento y reparación de las impresoras 3D prusa DIY serie V06-0702 que disponemos en la Universidad, así como del montaje, calibración y mantenimiento de mi propia impresora 3D modelo Prusa Mendel.

2. Los microcontroladores de los fabricantes ATMega son de una máxima utilidad para el control de motores y otras funciones debido a que tiene una elevada capacidad de almacenamiento, y también en la transmisión y recepción de datos y su sencilla programación, teniendo una gama de utilidades en diferentes campos de estudios.

3. Mediante la implementación de los ejercicios y el proyecto pudimos comprender y tener una visión más amplia acerca del funcionamiento de los microcontroladores ATMega2560 y de los motores de paso, así

como la forma en que se realiza la comunicación entre la tarjeta y pronterface.

4. En general, la aplicación de microcontroladores en la impresión 3D se encuentra innovando y cambiando constantemente, y cada vez podemos observar nuevos modelos con mayores capacidades. Para cumplir con los parámetros de impresión, podemos concluir que la plata forma arduino que ha sido objeto de estudio en esta tesina es una herramientas con gran ventaja para desarrollar objetos autónomos e interactivos, capaz de comunicarse con software instalado en un computador como pronterface, Dada su rápida curva de aprendizaje y su bajo precio constituye una herramienta ideal para estudiantes, maestros, diseñadores y cualquier interesado en electrónica y robótica.

RECOMENDACIONES

Las recomendaciones que logramos obtener del proyecto son las expuestas a continuación:

1. Después de ensamblar la impresora se requiere una calibración en el eje que mantiene al motor x, y también en el plato base de la impresora, es necesario realizar este procedimiento por las dificultades que se podría obtener al crear un objeto.

2. Se recomienda chequear las tarjetas (ramp y drivers) físicamente antes de su instalación para verificar que se encuentren en un buen estado para su funcionamiento. Después realizar su respectiva calibración para limitar su corriente y frecuencia para los motores.

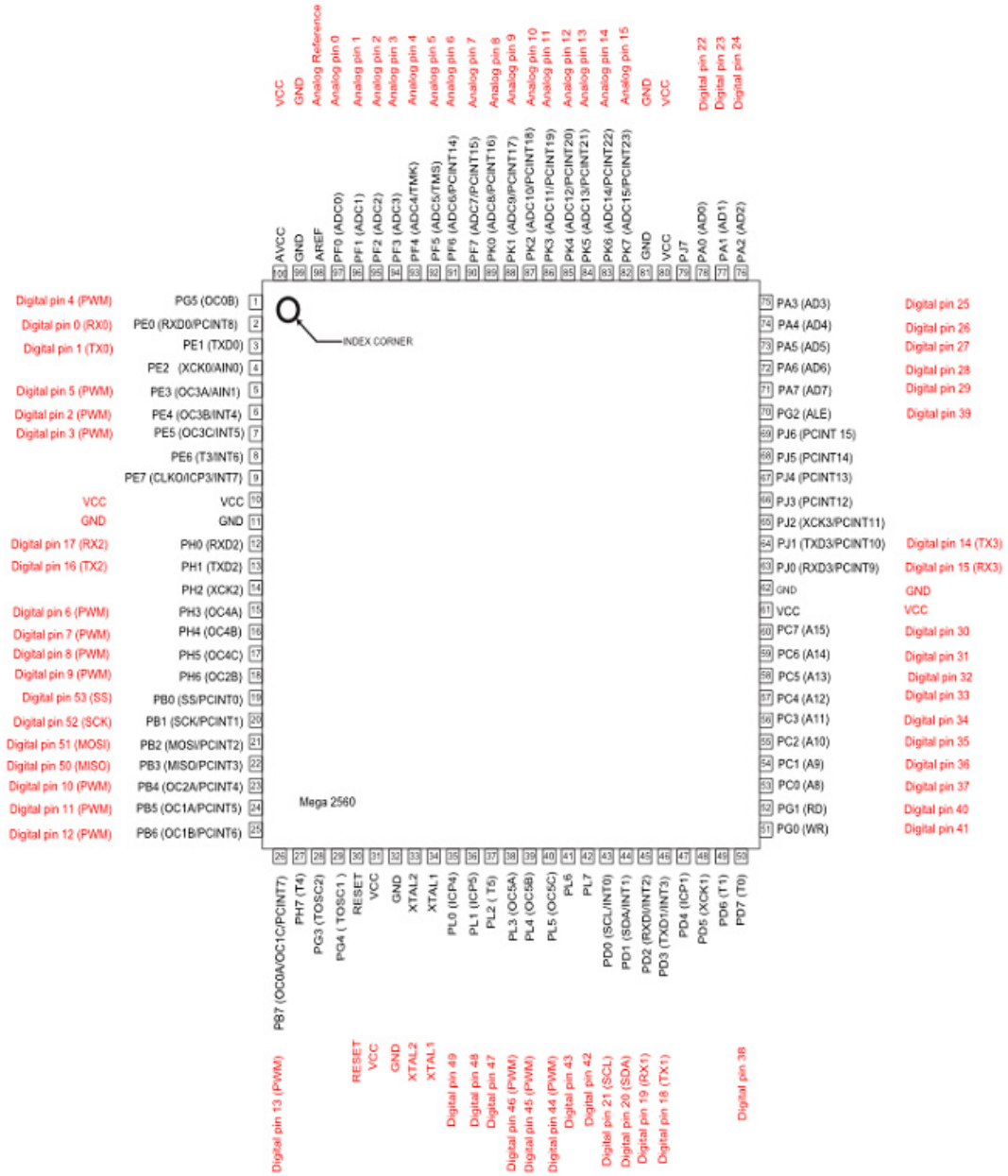
3. Mediante el desarrollo de un ejercicio de prueba, los driver llegaron un nivel de temperatura elevada, fue necesario adaptar un ventilador de mayor fuerza y fue necesario chequear los driver cada media hora para ver si requieren un segundo ventilador adicional.

4. Para la conexión y uso de los pines de la tarjeta arduino es necesario revisar cada uno de ellos y verificar cuales se encuentran reservados por el programa, y cuáles pueden ser usados como I/O y qué pines

sirven para funciones específicas, además es necesario verificar que se encuentren correctamente conectado el puerto de la tarjeta para asegurar el correcto funcionamiento del sistema.

ANEXOS

ANEXO 1.- Arduino Mega 2560 PIN diagram



ANEXO 2.- Arduino Mega 2560 PIN mapping table

Pin Number	Pin Name	Mapped Pin Name
1	PG5 (OC0B)	Digital pin 4 (PWM)
2	PE0 (RXD0/PCINT8)	Digital pin 0 (RX0)
3	PE1 (TXD0)	Digital pin 1 (TX0)
4	PE2 (XCK0/AIN0)	
5	PE3 (OC3A/AIN1)	Digital pin 5 (PWM)
6	PE4 (OC3B/INT4)	Digital pin 2 (PWM)
7	PE5 (OC3C/INT5)	Digital pin 3 (PWM)
8	PE6 (T3/INT6)	
9	PE7 (CLKO/ICP3/INT7)	
10	VCC	VCC
11	GND	GND
12	PH0 (RXD2)	Digital pin 17 (RX2)
13	PH1 (TXD2)	Digital pin 16 (TX2)
14	PH2 (XCK2)	
15	PH3 (OC4A)	Digital pin 6 (PWM)
16	PH4 (OC4B)	Digital pin 7 (PWM)
17	PH5 (OC4C)	Digital pin 8 (PWM)
18	PH6 (OC2B)	Digital pin 9 (PWM)
19	PB0 (SS/PCINT0)	Digital pin 53 (SS)
20	PB1 (SCK/PCINT1)	Digital pin 52 (SCK)
21	PB2 (MOSI/PCINT2)	Digital pin 51 (MOSI)
22	PB3 (MISO/PCINT3)	Digital pin 50 (MISO)
23	PB4 (OC2A/PCINT4)	Digital pin 10 (PWM)
24	PB5 (OC1A/PCINT5)	Digital pin 11 (PWM)
25	PB6 (OC1B/PCINT6)	Digital pin 12 (PWM)
26	PB7 (OC0A/OC1C/PCINT7)	Digital pin 13 (PWM)
27	PH7 (T4)	
28	PG3 (TOSC2)	
29	PG4 (TOSC1)	
30	RESET	RESET
31	VCC	VCC
32	GND	GND
33	XTAL2	XTAL2
34	XTAL1	XTAL1
35	PL0 (ICP4)	Digital pin 49
36	PL1 (ICP5)	Digital pin 48
37	PL2 (T5)	Digital pin 47
38	PL3 (OC5A)	Digital pin 46 (PWM)
39	PL4 (OC5B)	Digital pin 45 (PWM)
40	PL5 (OC5C)	Digital pin 44 (PWM)

41	PL6	Digital pin 43
42	PL7	Digital pin 42
43	PD0 (SCL/INT0)	Digital pin 21 (SCL)
44	PD1 (SDA/INT1)	Digital pin 20 (SDA)
45	PD2 (RXDI/INT2)	Digital pin 19 (RX1)
46	PD3 (TXD1/INT3)	Digital pin 18 (TX1)
47	PD4 (ICP1)	
48	PD5 (XCK1)	
49	PD6 (T1)	
50	PD7 (T0)	Digital pin 38
51	PG0 (WR)	Digital pin 41
52	PG1 (RD)	Digital pin 40
53	PC0 (A8)	Digital pin 37
54	PC1 (A9)	Digital pin 36
55	PC2 (A10)	Digital pin 35
56	PC3 (A11)	Digital pin 34
57	PC4 (A12)	Digital pin 33
58	PC5 (A13)	Digital pin 32
59	PC6 (A14)	Digital pin 31
60	PC7 (A15)	Digital pin 30
61	VCC	VCC
62	GND	GND
63	PJ0 (RXD3/PCINT9)	Digital pin 15 (RX3)
64	PJ1 (TXD3/PCINT10)	Digital pin 14 (TX3)
65	PJ2 (XCK3/PCINT11)	
66	PJ3 (PCINT12)	
67	PJ4 (PCINT13)	
68	PJ5 (PCINT14)	
69	PJ6 (PCINT 15)	
70	PG2 (ALE)	Digital pin 39
71	PA7 (AD7)	Digital pin 29
72	PA6 (AD6)	Digital pin 28
73	PA5 (AD5)	Digital pin 27
74	PA4 (AD4)	Digital pin 26
75	PA3 (AD3)	Digital pin 25
76	PA2 (AD2)	Digital pin 24
77	PA1 (AD1)	Digital pin 23
78	PA0 (AD0)	Digital pin 22
79	PJ7	
80	VCC	VCC
81	GND	GND
82	PK7 (ADC15/PCINT23)	Analog pin 15
83	PK6 (ADC14/PCINT22)	Analog pin 14
84	PK5 (ADC13/PCINT21)	Analog pin 13

85	PK4 (ADC12/PCINT20)	Analog pin 12
86	PK3 (ADC11/PCINT19)	Analog pin 11
87	PK2 (ADC10/PCINT18)	Analog pin 10
88	PK1 (ADC9/PCINT17)	Analog pin 9
89	PK0 (ADC8/PCINT16)	Analog pin 8
90	PF7 (ADC7)	Analog pin 7
91	PF6 (ADC6)	Analog pin 6
92	PF5 (ADC5/TMS)	Analog pin 5
93	PF4 (ADC4/TMK)	Analog pin 4
94	PF3 (ADC3)	Analog pin 3
95	PF2 (ADC2)	Analog pin 2
96	PF1 (ADC1)	Analog pin 1
97	PF0 (ADC0)	Analog pin 0
98	AREF	Analog Reference
99	GND	GND
100	AVCC	VCC

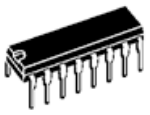
ANEXO 3.- Especificaciones Técnicas L293D

SGS-THOMSON MICROELECTRONICS

L293D
L293DD

PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES

- 600mA OUTPUT CURRENT CAPABILITY PER CHANNEL
- 1.2A PEAK OUTPUT CURRENT (non repetitive) PER CHANNEL
- ENABLE FACILITY OVERTEMPERATURE
- PROTECTION LOGICAL "0" INPUT
- VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)
- INTERNAL CLAMP DIODES



SO(12+4+4) Powerdip (12+2+2)

ORDERING NUMBERS:

L293DD L293D

DESCRIPTION

The Device is a monolithic integrated high voltage, high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as relays solenoides, DC and stepping motors) and switching power transistors.

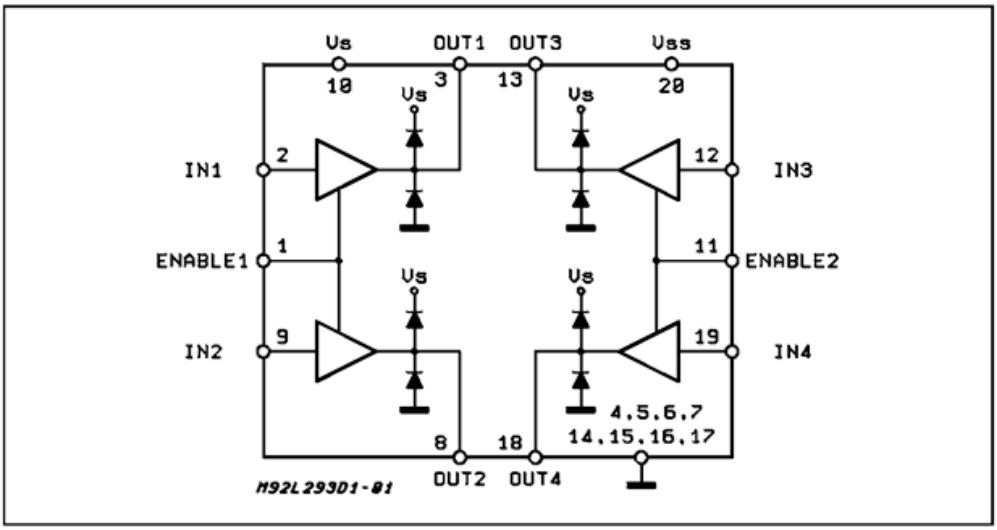
To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included.

This device is suitable for use in switching applications at frequencies up to 5 kHz.

The L293D is assembled in a 16 lead plastic packaage which has 4 center pins connected together and used for heatsinking

The L293DD is assembled in a 20 lead surface mount which has 8 center pins connected together and used for heatsinking.

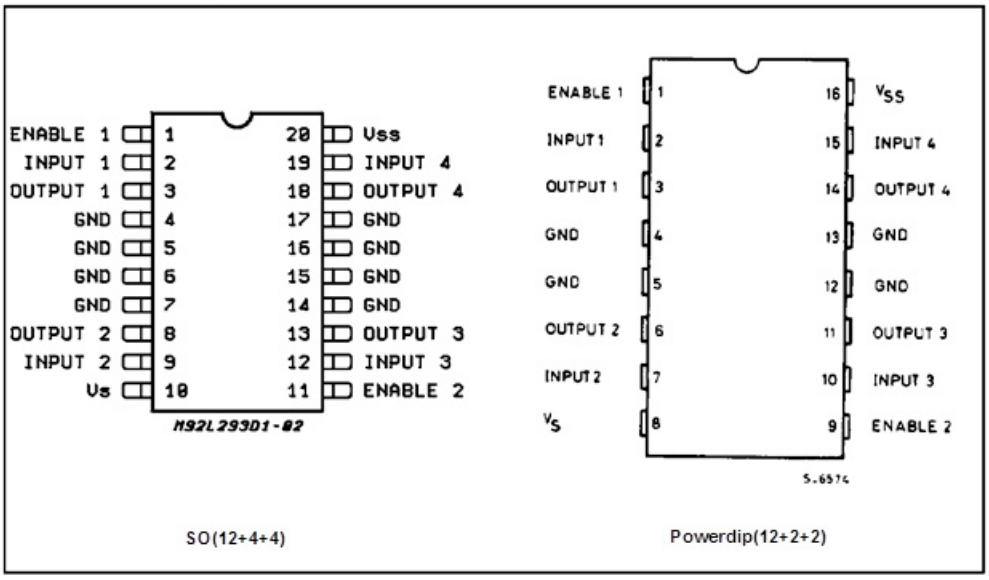
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Supply Voltage	36	V
V_{SS}	Logic Supply Voltage	36	V
V_I	Input Voltage	7	V
V_{en}	Enable Voltage	7	V
I_O	Peak Output Current (100 s non repetitive)	1.2	A
P_{tot}	Total Power Dissipation at T_{pins} = 90 °C	4	W
T_{stg}, T_j	Storage and Junction Temperature	– 40 to 150	°C

PIN CONNECTIONS (Top view)



SO(12+4+4) Powerdip(12+2+2)

THERMAL DATA

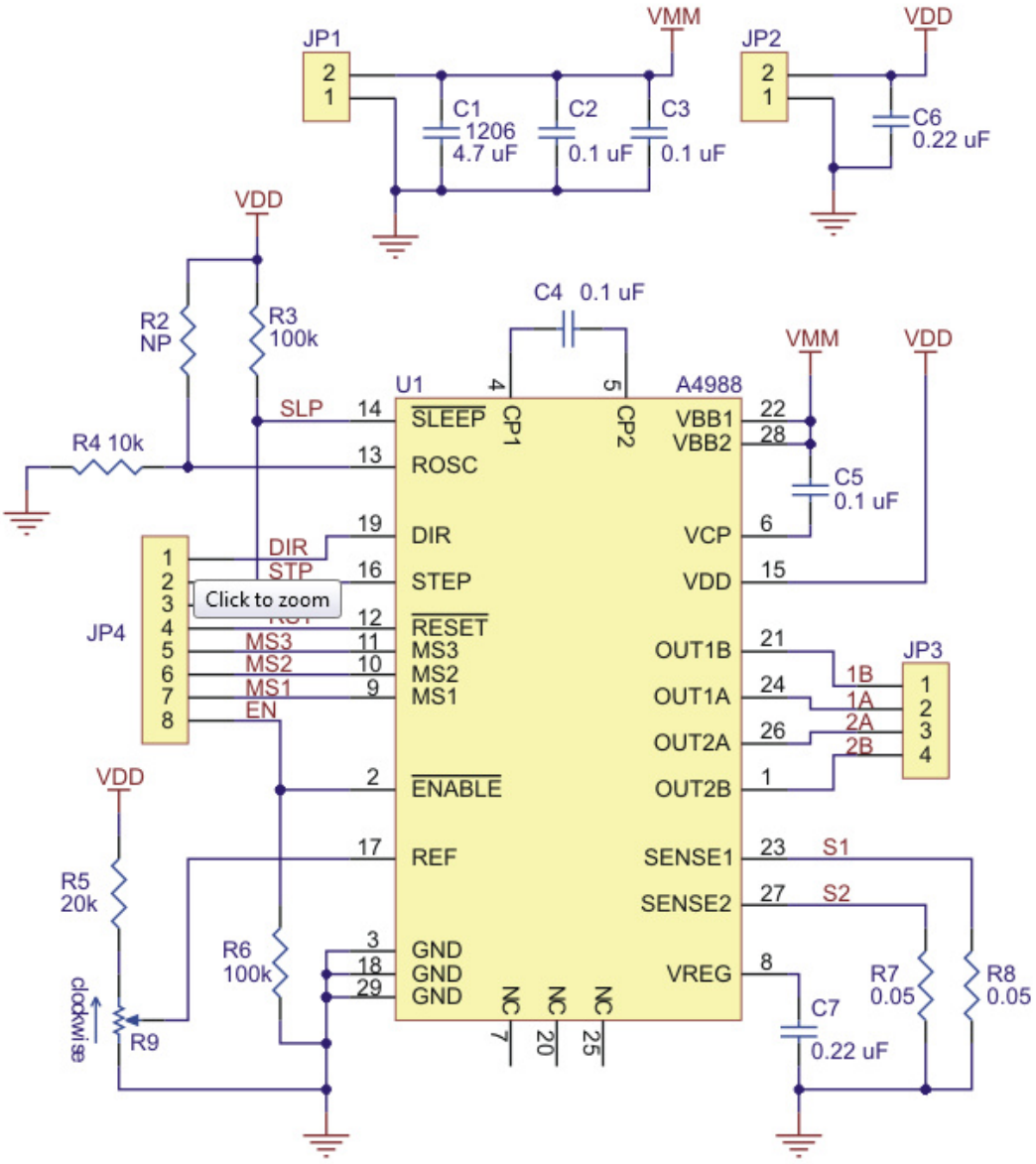
Symbol	Decription		DIP	SO	Unit
$R_{th\ j\text{-}pins}$	Thermal Resistance Junction-pins	max.	–	14	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance junction-ambient	max.	80	50 (*)	°C/W
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	max.	14	–	

(*) With 6 sq. cm on board heatsink.

ELECTRICAL CHARACTERISTICS (for each channel, V_S = 24 V, V_{SS} = 5 V, T_{amb} = 25 °C, unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 10)		V_{SS}		36	V
V_{SS}	Logic Supply Voltage (pin 20)		4.5		36	V
I_S	Total Quiescent Supply Current (pin 10)	V_I = L ; I_O = 0 ; V_{en} = H		2	6	mA
		V_I = H ; I_O = 0 ; V_{en} = H		16	24	mA
		V_{en} = L			4	mA
I_{SS}	Total Quiescent Logic Supply Current (pin 20)	V_I = L ; I_O = 0 ; V_{en} = H		44	60	mA
		V_I = H ; I_O = 0 ; V_{en} = H		16	22	mA
		V_{en} = L		16	24	mA
V_{IL}	Input Low Voltage (pin 2, 9, 12, 19)		– 0.3		1.5	V
V_{IH}	Input High Voltage (pin 2, 9, 12, 19)	$V_{SS} \leq 7$ V	2.3		V_{SS}	V
		$V_{SS} > 7$ V	2.3		7	V
I_L	Low Voltage Input Current (pin 2, 9, 12, 19)	V_L = 1.5 V			– 10	A
I_H	High Voltage Input Current (pin 2, 9, 12, 19)	2.3 V $\leq V_H \leq V_{SS}$ – 0.6 V		30	100	A
$V_{en\ L}$	Enable Low Voltage (pin 1, 11)		– 0.3		1.5	V
$V_{en\ H}$	Enable High Voltage (pin 1, 11)	$V_{SS} \leq 7$ V	2.3		V_{SS}	V
		$V_{SS} > 7$ V	2.3		7	V
$I_{en\ L}$	Low Voltage Enable Current (pin 1, 11)	$V_{en\ L}$ = 1.5 V		– 30	– 100	A
$I_{en\ H}$	High Voltage Enable Current (pin 1, 11)	2.3 V $\leq V_{en\ H} \leq V_{SS}$ – 0.6 V			± 10	A
$V_{CE(sat)H}$	Source Output Saturation Voltage (pins 3, 8, 13, 18)	I_O = – 0.6 A		1.4	1.8	V
$V_{CE(sat)L}$	Sink Output Saturation Voltage (pins 3, 8, 13, 18)	I_O = + 0.6 A		1.2	1.8	V
V_F	Clamp Diode Forward Voltage	I_O = 600nA		1.3		V
t_r	Rise Time (*)	0.1 to 0.9 V_O		250		ns
t_f	Fall Time (*)	0.9 to 0.1 V_O		250		ns
t_{on}	Turn-on Delay (*)	0.5 V_I to 0.5 V_O		750		ns
t_{off}	Turn-off Delay (*)	0.5 V_I to 0.5 V_O		200		ns

(*) See fig. 1.



Schematic diagram of the md09b A4988 stepper motor driver carrier.

REFERENCIAS BIBLIOGRÁFICAS

[1] M. McRoberts, Beginning Arduino, Primera Edición, 2010.

[2] T. D. Gauray Tyagi, «NIC-Muzaffamagar, UP, 3D Printing Technology,» [En línea]. Available: , http://123seminarsonly.com/Seminar-Reports/2013-02/83204431-3D-Printing-Technology.pdf. [Último acceso: marzo 2013].

[3] D. Girardo B. y I. Tabares G., Teoría de Control, Primera Edición, 1997.

[4] T. Olsson, D. Gaetano, S. Wiklund y J. Odhner, Open Softwear, Segunda Edición, 2011.

[5] S. Mateo, «Impresoras 3D, la nueva revolución tecnológica,» [En línea]. Available: http://sergimateo.com/impresoras-3d-la-nueva-revolucion-tecnologica/. [Último acceso: marzo 2013].

[6] «Revista Letreros N°92 - Impresoras 3D, tecnología de avanzada al alcance de la industria nacional,» [En línea]. Available: http://www.revistaletreros.com/pdf/92-052a056.pdf. [Último acceso: abril 2013].

[7] M. P. Groover, Fundamentos de Manufactura Moderna, McGraw Hill - Primera Edición, 2007.

[8] R. V. 29, Impresora de tipo RepRap, Cambridge University Press 2011, 2013.

[9] J. Floyd Kelly y P. Hood-Daniel, Printing in Plastic: Build Your Own 3D Printer, 2011.

[10] Hernández Bello y Ochoa Luna, «Sección de motores de paso,» de *Motor de Paso*, Primera Edición, 2004.

[11] «Impresoras 3D,» [En línea]. Available: http://www.impresoras3d.com/la-cabeza-de-impresion-tf3d. [Último acceso: abril 2013].

[12] Allegro, «Data sheet description,» Microsystem, Inc, [En línea]. Available: http://www.electronicaembajadores.com/datos/pdf1/sm/smci/a4988.pdf. [Último acceso: 26 marzo 2013].

[13] [En línea]. Available: reprap.org, http://reprap.org/wiki/RAMPS. [Último acceso: 27 marzo 2013].

[14] J. C. Rico, F. Ros y A. Serna Ruiz, Guía práctica de sensores, Primera Edición, 2010.

[15] J.-D. Warren, J. Adams y H. Molle, Arduino Robotics, Primera Edición, 2011.

[16] J. Pomares, Manual de programación de Arduino, Universidad de Alicante, 2013.

[17] S. Monk, Programming arduino, getting started with sketches, Mcgraw hill, 2012.

[18] T. Olsson, Arduino Wearables, Primera Edición, 2009.

[19] J. Purdum, Beginning C for Arduino: Learn C Programming for the Arduino, Primera Edición, 2012.

[20] D. Wheat, Arduino Internals, Primera Edición, 2011.

[21] S. F. Barrett, Arduino Microcontroller: Processing for Everyone, Primera Edición, 2012.

[22] B. Evans, Practical 3D Printer, Primera Edición, 2011.

[23] G. Borenstein, Making Things See: 3D Vision with Kinect, Processing, Arduino, and MakerBot, Primera Edición, 2012.

[24] M. Böhmer, Beginning Android ADK with Arduino, Primera Edición, 2012.

[25] E. Ramos Melgar y C. Castro Diez, Arduino and Kinect Projects: Design, Build, Blow Their Minds, Primera Edición, 2012.

[26] Hernandez Bello y Ochoa Luna, Motor de Paso, Primera Edición, 2004.