ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"DISEÑO DE RUTINAS DE CONTROL EMPLEANDO SISTEMA ARDUINO PARA EL USO DE DIFERENTES EXTRUSORAS EN IMPRESIÓN 3D, APLICACIÓN PRÁCTICA EN PROTOTIPO DE IMPRESIÓN 3D"

EXAMEN COMPLEXIVO

Previa a la obtención del Título de:

INGENIERO EN ELECTRICIDAD ESPECIALIZACIÓN ELECTRÓNICA Y AUTOMATIZACIÓN INDUSTRIAL.

Presentado por:

Jaime Omar Bustamante Alarcón.

Iván Mauricio Salazar Carrión.

Guayaquil – Ecuador
2014

AGRADECIMIENTO

Primeramente a Dios por sus bendiciones, a mi madre querida la Sra. Betty Alarcón que es la piedra angular de este logro obtenido, a mi padre Jaime por su apoyo para que pueda terminar la carrera, a mis hermanos Andrés y Karem que siempre creyeron en mi y finalmente a esos buenos amigos que me alentaron hasta el final.

Jaime Bustamante Alarcón.

A Dios, a mi mamá, a mi papá, a mi hermana, a mi novia, a mis primas y a toda mi familia en general. También le agradezco al Ing. Carlos Valdivieso por su ayuda y su guía oportuna.

Iván Salazar Carrión.

DEDICATORIA

Dedico este logro a mis padres Betty y Jaime, a mis hermanos Karem y Andrés, a mi tía Nelly Alarcón, a mi abuelita María Ruth, a mis tíos paternos y maternos y a todas esas personas que de una u otra manera me ayudaron en los momentos más difíciles de este largo camino.

Jaime Bustamante Alarcón.

Le dedico la obtención de este título principalmente a mi mamá la Sra. Marianita Carrión, a mi tía Olguita Carrión que está en el cielo y que fue mi segunda mamá y también a todos los que siempre me apoyaron.

Iván Salazar Carrión.

TRIBUNAL DE SUSTENTACIÓN

Ph. D. Boris Vintimilla B.

PRESIDENTE DEL TRIBUNAL

Ing. Carlos Valdivieso A.

DIRECTOR

Ing. Hugo Villavicencio V.

VOCAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de este trabajo final de titulación, nos corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL".

(Reglamento de exámenes y títulos profesionales de la ESPOL)

Jaime Omar Bustamante Alarcón

Iván Mauricio Salazar Carrión

RESUMEN

La implementación se hará tomando como modelo base una impresora PRUSA DIY que proporcionará las estructuras, bases y rieles que se utilizan en la construcción de una impresora 3D; adicionalmente la PRUSA DIY trae consigo termistores, sensores ópticos, finales de carrera y motores de paso; los cuales se controlarán a través de una tarjeta Arduino Mega 2560 que será el cerebro de nuestro sistema. Una tarjeta madre RepRap Arduino Mega Pololu Shield será la encargada de controlar la electrónica antes mencionada. A través de Marlin un firmware que se instala en la tarjeta Arduino se podrán comunicar el programa controlador de la impresora (Host) con todos los componentes electrónicos que funcionan en la impresora.

Pronterface, el programa controlador, será el encargado de preparar el archivo .STL que es la gráfica en 3D, la cual se convierte a un archivo en código G bajo unas condiciones de parámetros que se establecen en el software Cortador de capas. Luego de calibrar parámetros mecánicos y de instalar el software adecuado podremos imprimir objetos en 3 dimensiones y con el control de extrusora seleccionaremos el color del filamento para nuestra impresión.

ÍNDICE GENERAL

AGRADECIMIENTO……II

DEDICATORIA……..……...III

TRIBUNAL DE SUSTENTACIÓN…….……...IV

DECLARACIÓN EXPRESA……....V

RESUMEN……...VI

INDICE GENERAL……....VII

ABREVIATURAS……XII

INDICE DE FIGURAS……..……XV

INTRODUCCIÓN……..…….XVIII

CAPÍTULO 1: DESCRIPCIÓN GENERAL DEL PROYECTO……..……....1

1.1 OBJETIVOS……..……..…….....1

1.2 DESCRIPCIÓN DE LA PLANTA……....2

1.3 IMPRESORAS 3D……..……..……3

1.3.1 PROGRAMA DE CONTROL PARA LA IMPRESORA…..……..4

1.3.2 TÉCNICA DE IMPRESIÓN…..…….…….….4

1.3.3 MATERIAL DEL FILAMENTO……..…….……..……7

1.4 ARDUINO…….……..……8

1.4.1 ¿POR QUÉ ARDUINO?……..…….……...9

1.5 FIRMWARE…….……....11

1.6 IMPRESORAS 3D ACTUALMENTE EN EL MERCADO…..……...……12

1.7 LIMITACIONES……..……..……...……...17

1.8 PASOS PARA REALIZAR UNA IMPRESIÓN……..……..……..…17

CAPÍTULO 2: FUNDAMENTO TEÓRICO……..……19

2.1 PROGRAMA PYTHON…….....20

2.2 PYSERIAL……...22

2.3 PYTHON UNICODE……....22

2.4 PRONTERFACE…….....….23

2.5 SLIC3R……..…….....25

2.6 ARDUINO MEGA…….……..27

2.6.1 RESUMEN DE CARACTERÍSTICAS DEL ATMEGA 2560…..…….....28

2.6.2 ALIMENTACIÓN……...…29

2.6.3 MEMORIA……..…..31

2.6.4 ENTRADAS Y SALIDAS……...31

2.6.5 COMUNICACIONES…….....…...33

2.6.6 PROGRAMACIÓN…….……33

2.6.7 REINICIO AUTOMÁTICO POR SOFTWARE……34

2.6.8 PROTECCIÓN CONTRA SOBRETENSIONES EN USB……...…35

2.6.9 PINES DEL ARDUINO MEGA 2560……..36

2.7 MANEJO DE MOTORES DC…….……...38

2.7.1 MOTORES DE PASO……..……..39

2.8 LA ELECTRONICA…….……..….41

2.8.1 CONTROLADOR POLOLU……43

2.8.2 TRAJETA CONTROLADORA RAMPS…….....44

CAPÍTULO 3: FUNCIONAMIENTO DE LA IMPRESORA, EJERCICIOS PREVIOS Y PROYECTO FINAL ……..…….……46

3.1 CARACTERÍSTICAS DEL SISTEMA DE IMPRESIÓN…….……...46

3.2 CADENA DE HERRAMIENTAS EN LA IMPRESIÓN……..……48

3.3 EJERCICIOS PREVIOS…….……49

3.3.1 COMUNICACIÓN CON LA TARJETA ARDUINO……49

3.3.1.1 DIAGRAMA DE BLOQUES…….……53

3.3.1.2 DIAGRAMA ASM……..…54

3.3.1.3 CÓDIGO FUENTE……...55

3.3.2 SEMÁFORO USANDO 3 COLORES DE LED……56

3.3.2.1 DIAGRAMA DE BLOQUES……....57

3.3.2.2 DIAGRAMA ASM……..58

3.3.2.3 CÓDIGO FUENTE……59

3.4 SELECTOR DE TRES EXTRUSORAS DE DIFERENTE COLOR MEDIANTE ARDUINO Y L293D…..…….……60

3.4.1 CONFIGURACIÓN DE PARÁMETROS DE MARLIN……62

3.4.2 DESCRIPCIÓN DEL PROYECTO FINAL……..…….64

3.4.3 DIAGRAMA DE BLOQUES…….65

3.4.4 DIAGRAMA ASM……..66

3.4.5 DESCRIPCIÓN DEL ALGORITMO……...67

3.4.6 CODIGO FUENTE……68

4. CAPÍTULO 4: IMPLEMENTACIÓN DE EJERCICIOS DE PRUEBA Y SIMULACIÓN DEL PROYECTO……..76

4.1 “COMUNICACIÓN CON LA TARJETA ARDUINO”……77

4.1.1 IMPLEMENTACIÓN…….78

4.1.2 LISTADO DE MATERIALES……...78

4.1.3 CONCLUSIONES……..……..79

4.1.4 RECOMENDACIONES……79

4.2 “SEMÁFORO USANDO 3 COLORES DE LED”…….79

4.2.1 IMPLEMENTACIÓN…….80

4.2.2 LISTA DE MATERIALES……82

4.2.3 CONCLUSIONES……82

4.2.4 RECOMENDACIONES……...83

4.3 SELECTOR DE TRES EXTRUSORAS DE DIFERENTE COLOR MEDIANTE ARDUINO Y L293D……....83

4.3.1 IMPLEMENTACIÓN……...…93

4.3.2 LISTA DE COMPONENTES…….93

4.3.3 DIAGRAMA DE CONEXIONES…….....95

CONCLUSIONES……..……....96

RECOMENDACIONES…….……....97

ANEXOS……..98

BIBLIOGRAFÍA……..…101

ABREVIATURAS

°C	Grados centígrados
3D	3 dimensiones
3V3	3.3 voltios
ABS	Acrilonitrilo butadieno estireno
AC/DC	Corriente alterna/Corriente continua
AREF	Voltaje de referencia para señales analógicas
ASM	Máquina de estados algorítmicos
AVR C	Programación AVR en C
C++	Lenguaje de programación
CAD	Diseño asistido por computadora
CNC	Control numérico computarizado
DSPC	Proyección aglutinante
DTR	Terminal de Datos Lista
EEPROM	ROM programable y borrable eléctricamente
FDM	Deposición de hilo fundido
FTDI	Empresa Future Technology Devices International
GND	Tierra de un circuito eléctrico

I/O	Entrada/ Salida
ICSP	Programación serial en circuito
IOREF	Voltaje de entrada/salida referencial
KB	Mil Bytes
LED	Diodo emisor de luz
mA	miliamperios
MHz	Un millón de Hertz
mm	Milímetros
NPN	Transistor tipo NPN
OHM	Unidad de resistencia eléctrica
PE	Polietileno
PLA	Polímero de ácido láctico
PVC	Policloruro de Vinilo
PWM	Modulación por ancho de pulso
RAM	Memoria de acceso aleatorio
RAMPS	Protección RepRap Mega Pololu
RX	Recepción de señal
SRAM	Memoria estática de acceso aleatorio
TTL	Lógica transistor a Transistor
TX	Transmisión de señal

UARTS	Transmisor Receptor Asíncrono Universal
USB	Bus serial Universal
V	Voltios
Vin	Voltaje de entrada

ÍNDICE DE FIGURAS

FIGURA 1.1: Prusa Mendel……..……13

FIGURA 1.2: Cube 3D Printer…….……14

FIGURA 1.3: MetalBot 3D Printer……14

FIGURA 1.4: M.O.B. 3D Printer……15

FIGURA 1.5: UP! Plus 3D Printer……15

FIGURA 1.6: Object 24 3D Printer……..……16

FIGURA 1.7: FORM 1 3D Printer……16

FIGURA 2.1: Instalación de Python……21

FIGURA 2.2: Proceso de compilación Python Unicode……..……22

FIGURA 2.3: Opciones de Pronterface……..…25

FIGURA 2.4: Entorno de Slic3r……...27

FIGURA 2.5: Arduino Mega2560……27

FIGURA 2.6: Mapeo de pines del Arduino Mega 2560 parte 1…… …36

FIGURA 2.7: Mapeo de pines del Arduino Mega 2560 parte 2…… …37

FIGURA 2.8 Diferentes tipos de motores de paso……40

FIGURA 2.9 Plano de la electrónica de una impresora 3D……41

FIGURA 2.10 RepRap Arduino Mega Pololu Shield……..…44

FIGURA 3.1: Flujo organizacional del proceso para impresión 3D……47

FIGURA 3.2: Cadena de herramientas de la impresora 3D……..….48

FIGURA 3.3: Configuración del software Arduino, selección del modelo.……50

FIGURA 3.4: Configuración del software Arduino, puerto serial……..…….51

FIGURA 3.5: Botón cargar programa…….……52

FIGURA 3.6: Blink en la ventana de programación Arduino …..……52

FIGURA 3.7: Tarjeta Arduino conectada al CPU vía puerto USB.……53

FIGURA 3.8: Diagrama de Bloques de BLINK ……53

FIGURA 3.9: Diagrama ASM para cargar BLINK en la tarjeta…….....….54

FIGURA 3.10 Esquemático de las conexiones de Semáforo……..56

FIGURA 3.11 Compilación exitosa del código fuente “Semáforo”.……..….57

FIGURA 3.12 Diagrama de Bloques del programa Semáforo……57

FIGURA 3.13 Diagrama ASM para el programa Semáforo……58

FIGURA 3.14 Cambio en la configuración del Baudrate en Marlin……..……..…62

FIGURA 3.15 Configuración de la lógica de los finales de carrera……63

FIGURA 3.16 Esquema de conexiones del selector de colores……..…...64

FIGURA 3.17 Diagrama de bloques del selector de colores……...65

FIGURA 3.18 Diagrama ASM del selector de colores……..66

FIGURA 4.1 LED del Pin #13 encendido, comunicación exitosa con el PC…….78

FIGURA 4.2 Semáforo tradicional temporizado utilizando 3 diodos LED…….81

FIGURA 4.3 Entorno de Pronterface……..…85

FIGURA 4.4 Configuración de parámetros antes de imprimir……..…86

FIGURA 4.5 Impresión de un cubo de 4 caras laterales……..……...87

FIGURA 4.6 Primeros diseños 3D impresos……..……...…88

FIGURA 4.7 Escorpión con diferentes configuraciones de impresión……..…….89

FIGURA 4.8 Switch selector de 3 posiciones y Arduino ATmega 2560…….....90

FIGURA 4.9 Sensores ópticos y motor de paso del eje principal……..……91

FIGURA 4.10 Tarjeta controladora del motor formada por 2 integrados L293D..92

FIGURA 4.11 Maqueta del proyecto final junto a la impresora Prusa Mendel….93

FIGURA 4.12 Diagrama de conexiones del selector de colores utilizando tarjeta ATmega2560……..……..…95

INTRODUCCIÓN

El siguiente proyecto tiene como finalidad aplicar técnicas utilizadas en el control de extrusoras en equipos de impresión 3D a través de microcontroladores. La técnica que se va a utilizar para controlar el movimiento de la extrusora está basada en movimientos de fracciones de milímetro, desplazándose sobre los 3 ejes del plano coordenado X, Y, Z. La tarjeta Arduino va a controlar todo el funcionamiento de la impresora, esta tarjeta posee un software mediante el cual podemos programar diferentes aplicaciones basadas en lenguaje C. El firmware es el programa que va a servir como el canal de comunicación entre el programa controlador, instalado en la tarjeta Arduino, y todos los componentes electrónicos y mecánicos que trabajan en la impresora 3D.

Para poder emular un cabezal con 3 extrusoras hemos construido una maqueta la cual trabajará con Arduino y otros componentes electrónicos para mostrar la selección de la extrusora mediante un switch de 3 posiciones. Unos sensores ópticos deberán cumplir con la condición impuesta por el switch y buscar la posición seleccionada. Arduino nos ayudará con el control del motor de paso bipolar que mueve el eje principal del cabezal con las 3 extrusoras.

CAPÍTULO 1

DESCRIPCIÓN GENERAL DEL PROYECTO

1.1 OBJETIVOS

- Entender cómo funciona una impresora 3D.

- Conocer los diferentes tipos de software que se necesitan y poder hacer impresiones de cuerpos en 3 dimensiones.

- Diseñar rutinas de control para el uso de diferentes extrusoras en una impresora 3D utilizando la plataforma Arduino.

1.2 DESCRIPCIÓN DE LA PLANTA

El modelo de impresora 3D que vamos a utilizar es la Reprap Prusa Mendel, la cual viene con una boquilla de extrusión lo que nos va a permitir extrudir un filamento del material conocido como PLA.

Utilizando Arduino como parte central de nuestro proyecto vamos a implementar un circuito electrónico, que con el uso de sensores ópticos, simularemos en una maqueta una 3 extrusoras, pero de las cuales se deberá utilizar solamente una para crear el objeto deseado mediante un switch de 3 posiciones, es decir el objeto final solamente será de un color.

Dentro de los beneficios que representa este diseño está el hecho de que si se tiene solamente una extrusora al momento de que el material, que tiene un color en particular, ingresa a la boquilla y esta se calienta, el material deberá ser siempre el mismo ya que siempre quedará material dentro de la boquilla. Claro que si es posible limpiar la boquilla pero se debe tener paciencia y algo de técnica para no dañar la boquilla ni el sensor de temperatura que esta posee.

El alcance de este diseño llega solamente a la selección de los colores mediante una lógica de programación C++ instalada en el Arduino que controla el movimiento de un motor de paso y analiza el estado de 3 sensores ópticos.

1.3 IMPRESORAS 3D

La parte central de una impresora 3D tiene la idea de un robot cartesiano. Esta es una máquina que se puede mover en 3 direcciones lineales, a través de los ejes X, Y y Z, también conocidas como coordenadas cartesianas. Para hacer esto la impresora utiliza unos pequeños motores de paso que se pueden mover con bastante precisión y exactitud, usualmente 1.8° por paso, lo cual se traduce a un rango de resolución de fracciones de milímetros y que es la única forma a través de la cual estos motores de paso pueden ser controlados. [24] [pág. 2]

La idea de crear las impresoras 3D nace del hecho de poder materializar diseños creados en programas CAD. En la actualidad se fabrican piezas utilizadas en arquitectura y en varios tipos de industria como la armamentista, fotográfica y de telefonía celular; en la medicina también se está empezando a utilizar en la fabricación de prótesis y órganos, su campo de desarrollo es bastante elevado ya que el material con el que estas impresoras fabrican los objetos es muy variado. Uno de los materiales que puede ser impreso en tercera dimensión es el plástico. El término “plástico” abarca una amplia variedad de materiales por lo que se lo considera un término genérico.

1.3.1 PROGRAMA DE CONTROL PARA LA IMPRESORA

También conocido como el programa administrador o interfaz de la impresora, es donde toda la cadena de herramienta se une. Desde esta aplicación podemos conectarnos con la impresora y hablar con su firmware, mover los tres diferentes ejes, leer y configurar la temperatura para la boquilla de la extrusora, ejecutar nuestro programa cortador, e imprimir nuestros modelos 3D. [24] [pág. 40]

Pronterface es un tipo de programa administrador, que para impresoras Reprap Prusa Mendel, es el programa más utilizado que brinda una interfaz sencilla y muy práctica.

Debido a que las restricciones del uso de uno u otro programa administrador (Host) están dadas por el tipo de electrónica que se esté usando, para este proyecto es necesaria la utilización de Pronterface como programa administrador debido al modelo del prototipo con el cual vamos a trabajar.

1.3.2 TÉCNICA DE IMPRESIÓN

Existen varios métodos de impresión 3D, en nuestro proyecto utilizaremos el método FDM que es la tecnología más usada luego de la estereolitografía. El método FDM utiliza un filamento de plástico (PLA) que se desenrolla de una

bobina y abastece material hacia una boquilla de extrusión. La boquilla se alimenta con un filamento de un tamaño alrededor de 1,25mm que es calentado a una temperatura entre 0,5°C - 1ºC por debajo de la temperatura de fusión del material. La boquilla queda montada en la extrusora, que puede moverse en dirección horizontal (plano X, Y) y vertical (eje Z). [4]

Conforme la boquilla se desplaza por la mesa de acuerdo con la geometría que se le ha indicado, deposita un fino hilo de plástico extruido que va formando capas. Cuando el plástico sale de la boquilla se adhiere a la capa de abajo y luego de unos segundos se endurece. Esta tecnología de extrusión de termoplásticos permite fabricar piezas duras, resistentes y lo precisas. [4]

Ventajas del FDM:

- Se pueden realizar piezas que reflejan fielmente el diseño, tanto en forma, como dimensiones. Margen de error máximo de 0,1 mm en 400mm de longitud.

- Los moldes creados no se deforman, son estables dimensionalmente y con muy buenas características mecánicas, se utilizan mucho para ensayos funcionales, montajes, etc.

- Los materiales con los que se realiza tienen excelente resistencia a la temperatura, desde 85 ºC hasta 220 ºC

- Debido a los materiales que se utilizan, estos modelos son aptos para pintar, cromar, mecanizar.

- Los productos finales son más livianos que los que son construidos mediante el método de la estereolitografía.

- Por su velocidad relativa y su bajo costo de impresión, es recomendable para realizar pequeñas series. [4]

Limitaciones del FDM:

- El producto final presenta un aspecto granulado, por lo que el acabado superficial no es demasiado bueno.

- Presenta escasa consistencia vertical y esto constituye un gran problema en la impresión por capas.

- Debido al tamaño de los filamentos con los que se trabaja, tiene una exactitud restringida

- En términos generales, la velocidad de gestión no es elevada, es por esto que si se trata de piezas grandes o gruesas se vuelve un proceso muy lento. [4]

1.3.3 MATERIAL DEL FILAMENTO

Un termoplástico es un material que, como su nombre lo indica, a temperaturas relativamente altas, se vuelve plástico, deformable o flexible; se derrite cuando se calienta y se endurece en un estado de transición vítreo cuando se enfría lo suficiente. Los más usados comercialmente son: el polietileno (PE), el policloruro de vinilo (PVC), etc. Dentro del área de las impresiones en 3D los más utilizados son: Acrilonitrilo butadieno estireno (ABS), Polylactic acid (PLA). [5]

En nuestro proyecto emplearemos filamento de PLA, el cual es un polímero biodegradable derivado del ácido láctico; que se hace a partir de recursos renovables al 100%, como son el maíz, el trigo y otros productos ricos en almidón. Este ácido tiene muchas características iguales e incluso superiores a muchos plásticos derivados del petróleo, lo que lo hace muy eficaz para una

gran variedad de usos dentro de la industria textil, alimenticia y en la medicina. [20]

Su temperatura de fusión dentro del área de las impresiones en 3D oscila entre los 200°C y 230°C dependiendo de la calidad del material y las especificaciones de trabajo para la boquilla de la extrusora.

1.4 ARDUINO

Arduino es una pequeña tarjeta microcontroladora que posee un puerto USB, para conectarse a un computador y un número de sockets que pueden ser cableados con dispositivos externos tales como: motores, relés, sensores de luz, diodos láser, altavoces, micrófonos y más. Estos equipos pueden también ser alimentados a través de la conexión USB desde el computador, desde una batería de 9V o desde una fuente de poder. Estos equipos pueden ser controlados desde el computador, utilizando a Arduino como una tarjeta de interfaz, o programados a través de un computador y luego ser desconectados para trabajar independientemente. [21] [pág. 15]

El corazón del Arduino es un microcontrolador, que es en realidad una pequeña computadora en un chip. Tiene un procesador, memoria de acceso aleatorio

(RAM), una EEPROM y unos pines de entrada y salida que conectan al microcontrolador con el resto de los dispositivos. [21] [pág. 19]

En la familia de Arduinos existen varios modelos pero uno de los más poderosos es el Arduino Mega, el cual aumenta su número de puerto de I/O en comparación con otras tarjetas de la misma familia. Debido a su diseño es compatible con todos los shields (tarjetas adicionales de uso específico) aumentando sus características y su uso dentro de diferentes áreas de trabajo. Su procesador es el ATmega2560, cuya superficie está fijada de forma permanente a la tarjeta; por lo que no se podrá cambiar en caso de algún daño, lo cual representa una desventaja que no afecta a otros modelos.

1.4.1 ¿POR QUÉ ARDUINO?

Existen muchos microcontroladores y plataformas con microcontroladores disponibles para la física computacional, las ventajas de trabajar con Arduino son:

- **Asequible**.- El bajo costo de las placas Arduino comparadas con otras plataformas de microcontroladores, es una de sus más grandes ventajas. La versión más cara de un módulo de Arduino ya montada cuesta alrededor de $100.

- **Multi-Plataforma**.- El software de Arduino funciona en los sistemas operativos Windows, Macintosh OSX y Linux. La mayoría de los entornos para microcontroladores están limitados a Windows.

- **Entorno de programación simple y directo.**- La plataforma de programación para el Arduino es muy sencilla de aprender para usuarios principiantes y se adapta a las exigencias de usuarios avanzados.

- **Software ampliable y de código abierto.**- El software Arduino es de código abierto por lo que puede ser ampliado, en caso de ser necesario, por programadores experimentados. El lenguaje para usuarios avanzados es AVR C, pero para cualquier persona con un conocimiento medio avanzado en programación se puede ampliar la programación de la placa Arduino mediante las librerías de C++.

- **Hardware ampliable y de Código abierto**.- Arduino está basado en los microcontroladores ATMEGA168, ATMEGA328, ATMEGA1280 y

ATMEGA2560. La licencia de los módulos construidos a base de Arduino le pertenecen a Creative Commons, gracias a esto, todos los diseñadores de circuitos que lo requieran, pueden crear sus propias versiones de placas para ampliar o mejorar sus funciones. [7]

1.5 FIRMWARE

Cada tarjeta de control necesita un código especializado llamado Firmware, cargado en su microcontrolador, para hacer que la electrónica de la impresora 3D cobre vida. El firmware es responsable de interpretar los comandos de código-G enviados a la electrónica desde el software de control de la impresora. Que tan preciso el firmware haga esto, determinará que tan elaborados construirá la impresora los objetos. En ocasiones, sólo con pasar a una versión más actual del firmware se aumentará la calidad de impresión.

Marlin en un firmware muy popular en las plataformas RAMPS el cual está teniendo mucho desarrollo ya que es un firmware CNC de código abierto para microcontroladores Arduino. Al igual que Sprinter (un firmware más común), Marlin está diseñado para algunas de las plataformas más estándar como Reprap y Ultimakers. [24] [pág. 36]

Sin embargo Marlin posee algunas mejoras por sobre otros programas las cuales resultan en impresiones más limpias y suavizadas. Una de las mejoras que posee Marlin es que posee una mejor interpretación de la aceleración que usa la característica (look-ahead) para anticipar el movimiento que sigue para prevenir alguna pausa innecesaria en medio de la impresión.

Algunas ventajas:

- Aceleración con look-ahead.
- Sobremuestreo de temperatura para lecturas más precisas.
- Control de temperatura PID con sintonización PID automática. [24] [pág. 36]

1.6 IMPRESORAS 3D ACTUALMENTE EN EL MERCADO

Existen impresoras 3D personales en un rango de precios que varía entre $ 550 y $ 2500. Con bajos costos de impresión, una variedad de materiales de fabricación y una cantidad considerable de modelos, podremos escoger la impresora 3D personal para nuestras necesidades.

RepRap Prusa Mendel $554.00

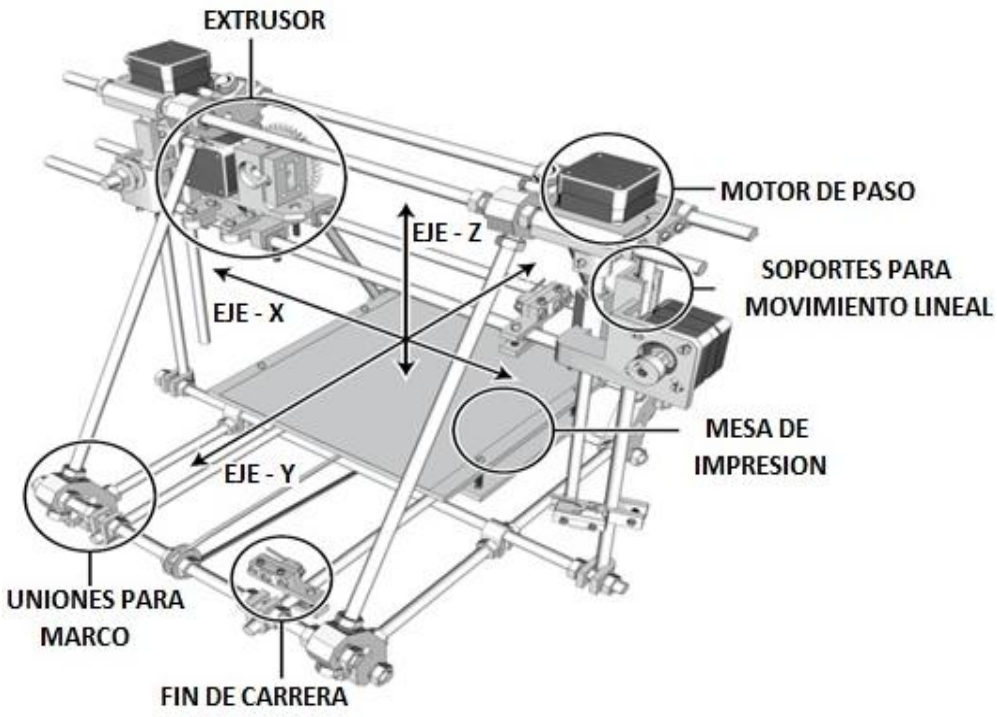


Figura 1.1 Prusa Mendel [12]

Este es un modelo básico, que sirve como modelo didáctico y que bien calibrado puede fabricar piezas bastante exactas y con apariencia aceptable.

La clásica Reprap Prusa Mendel consta de 5 motores:

- 1 motor para el eje x
- 1 motor para el eje y

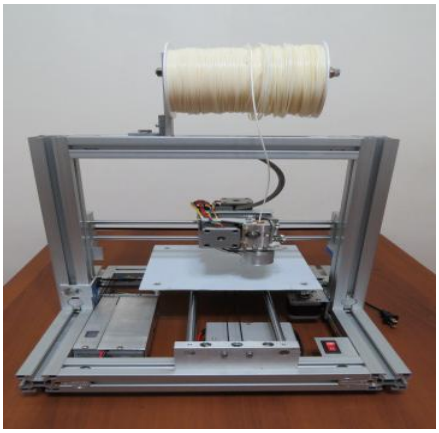
- 2 motores para el eje z
- 1 motor para la extrusora

Cube 3D Printer $1399.00

La impresora Cube 3D es una máquina fabricada para el hogar que permite realizar tus sueños de manera simple con el uso del WiFi.

Figura 1.2 Cube 3D Printer [17]

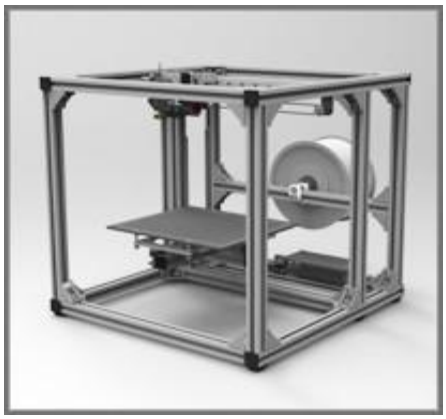
MetalBot 3D Printer $840.00



La MetalBot ofrece durabilidad y precisión a un costo muy bajo. Con Marco de aleación de Aluminio rígido con rodamientos lineales de grado industrial.

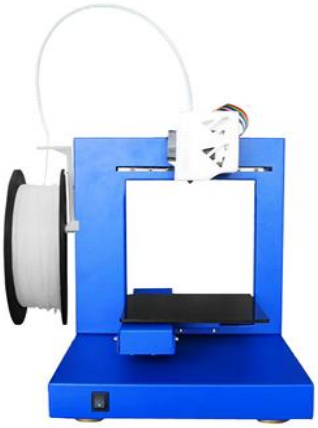
Figura 1.3 MetalBot 3D Printer [12]

M.O.B. 3D Printer $960



Innovadora y elegante, construida en aluminio y acero inoxidable sólido y resistente al clima, la M.O.B. ofrece una mayor resistencia y precisión a un bajo costo. Materiales: ABS y PLA.

Figura 1.4 M.O.B 3D Printer [12]



UP! Plus 3D Printer $1490.00

La nueva UP! Plus Desktop 3D Printer es una máquina simple y asequible. Usando tecnología adicional para añadir sucesivas capas de ABS construyendo modelos tridimensionales.

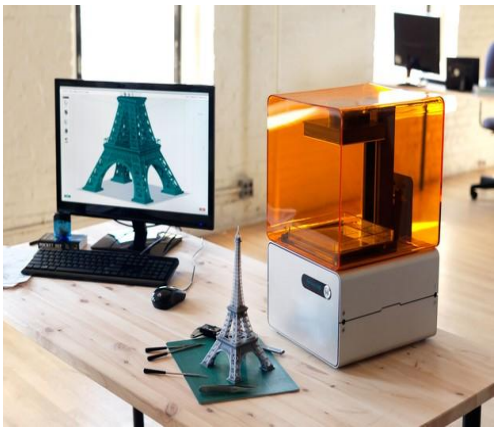
Figura 1.5 UP!Plus 3D Printer [18]

Object 24 3D Printer $19,900

La Objet 24 permite impresiones con material resistente, piezas móviles, paredes delgadas y superficies lisas para pintar. La Objet 24 utiliza un material blanco rígido (VeroWhitePlus), que ofrece estabilidad dimensional superior.

Figura 1.6 Object24 3D Printer[13]

FORM 1 3D Printer $2500.00



Impresora 3D de Formlabs de bajo costo que utiliza tecnología de estereolitografía “tan fácil como presionar un botón”. Definitivamente una opción interesante.

Figura 1.7 Form 1 3D Printer [19]

1.7 LIMITACIONES

- Dentro de las limitaciones que encontramos para este proyecto están el conseguir el Arduino, ya que no existe mucha demanda en nuestro mercado, por esta razón su stock es limitado y poco comercializado.

- Otro detalle a la hora de trabajar con estas impresoras es la materia prima con la que se fabrican los cuerpos en 3D, filamentos de ABS y PLA son también muy escasos en Ecuador, por lo que se tienen que hacer pedidos personalizados de los rollos con las medidas del filamento.

No se consideran una limitación las partes que conforman la impresora, ya que en la misma impresora podemos reproducir ciertas piezas si previamente las diseñamos en un archivo CAD con las medidas precisas.

1.8 PASOS PARA REALIZAR UNA IMPRESIÓN.

Los pasos básicos para imprimir un objeto 3D son:

- Abrir el programa administrador y conectar la impresora.

- Precalentar la boquilla de la extrusora.

- Descargar el modelo 3D para imprimirlo.

- Establecer las configuraciones de corte para tu impresora y el modelo a cortar.

- Cargar en el control de la impresora el archivo de código G generado.

- Verificar que la impresora esté lista para arrancar.

- Empezar la impresión.

CAPÍTULO 2

FUNDAMENTO TEÓRICO

Para poder trabajar con el software controlador de la impresora se requiere primero instalar otros software que van a realizar funciones intrínsecas de comunicación entre nuestro programa controlador y el hardware que vamos a utilizar.

Antes de utilizar Pronterface debemos instalar los siguientes programas en el orden dado:

1. Python 2.7.3

2. Pyserial-2.5.win32

3. Python2.8-win32-unicode

4. Pyreadline1.7.1.win32

5. Pyglet 1.1.4 (se debe ejecutar el archivo setup.py desde la ventana de comandos escribiendo la ruta de la carpeta con el mismo nombre) [25]

Si queremos evitarnos todo el proceso de instalar cada uno de los programas tenemos otra opción más rápida que además del Pronterface incluye Slic3r que es el convertidor de STL a Código G directamente como programa pre determinado; en la sección “Getting Printrun” descargar la versión pre compilada para Windows. [25]

2.1 PROGRAMA PYTHON

Python es el entorno de programación que usted necesitará instalar antes de poder ejecutar Slic3r, que es el programa clave para la impresión. La versión más recomendada es Python 2.7.3 Windows Installer. [8]

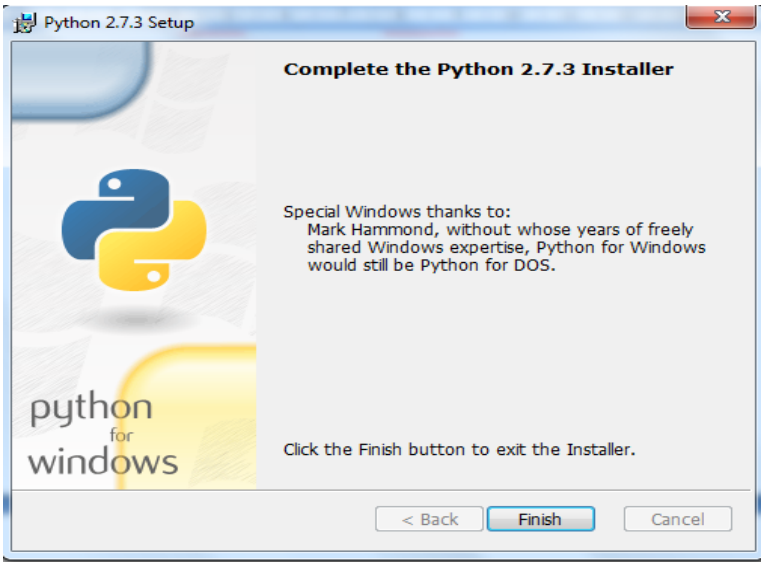


Figura 2.1 Instalación de Python [8]

Python es un lenguaje de programación que permite trabajar con mayor rapidez e integrar sus sistemas con mayor eficacia. Python es un lenguaje de programación multiparadigma, esto significa que se adapta a los estilos de los programadores para permitir varios tipos de programación como: programación orientada a objetos, programación imperativa y programación funcional. [6]

Este programa funciona en Windows, Linux / Unix, Mac OS X, y ha sido llevado a Java y a las máquinas virtuales .NET.

2.2 PYSERIAL

Pyserial es un plugin de Python que le permite a Python comunicarse vía USB. Este módulo encapsula el acceso para el puerto serial y proporciona los recursos para que Python pueda funcionar en Windows y Linux. [14]

La instalación detallada de Pyserial la podrá encontrar en la referencia [8].

2.3 PYTHON UNICODE

Otorga un numero único para cada caracter, sin importar cuál sea la plataforma, sin importar cual sea el programa, sin importar cual sea el lenguaje.

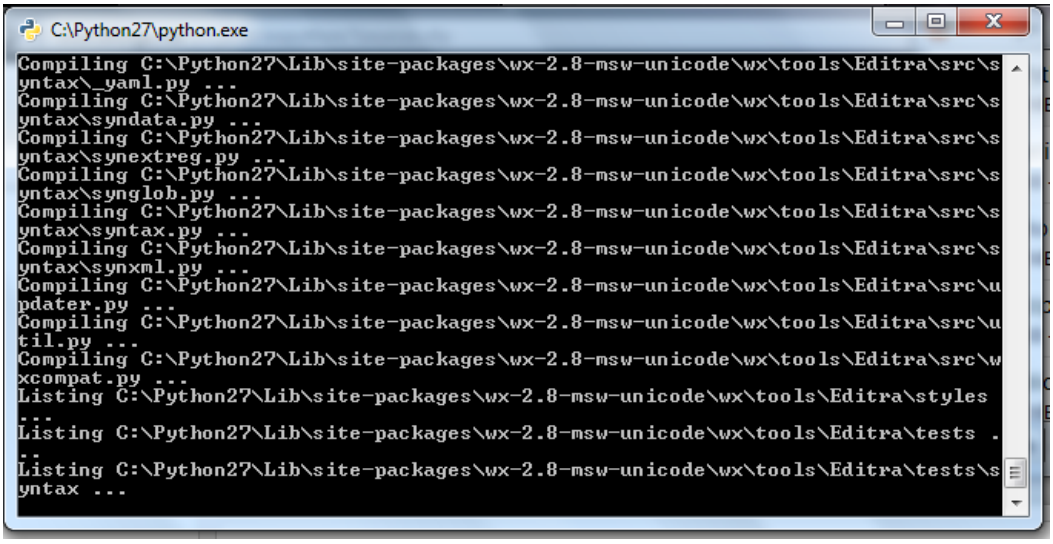


Figura 2.2 Proceso de compilación Python Unicode [8]

La incorporación de Unicode en las aplicaciones cliente-servidor o multi-niveles y sitios web ofrece ahorros significativos en los costos por el uso de juegos de caracteres heredados.

Unicode permite a un solo producto de software o un solo sitio web dirigirse a través de múltiples plataformas, idiomas y países sin necesidad de re-ingeniería. Unicode permite a los datos ser transportados a diferentes sistemas sin alteraciones en su código original. [22]

2.4 PRONTERFACE

Aunque no está cargado de muchas características avanzadas como otros programas controladores, Pronterface es el programa controlador más utilizado en máquinas RepRap y trabaja igual de bien con otras impresoras que usen una electrónica compatible. [24] [pág 43]

En Pronterface utilizamos el panel de control para conectarnos a nuestra impresora, mover sus ejes, fijar y monitorear valores de temperatura. Pronterface no es tan robusto por lo que no nos permite una visualización del objeto desde varios ángulos (podemos utilizar replicatorG como alternativa para realizar esta

función), pero también tiene una ventaja que es que permite visualizar las capas creadas por el código G y la secuencia de cómo estas se van construyendo antes de mandar a imprimir.

Debido a que tiene una arquitectura más abierta, Pronterface puede ser configurado para trabajar con cualquiera de las aplicaciones cortadoras disponibles en la actualidad.

Para integrar a Pronterface el cortador de nuestra preferencia y hacer que directamente se cargue un archivo .STL para ser cortado automáticamente, se debe abrir la ventana “edit settings” en setting > options, y colocar la ruta donde está instalado nuestro contador en las opciones slicecommand y sliceoptscommand; en nuestro caso instalamos el programa Slic3r. [24] [pág 43]

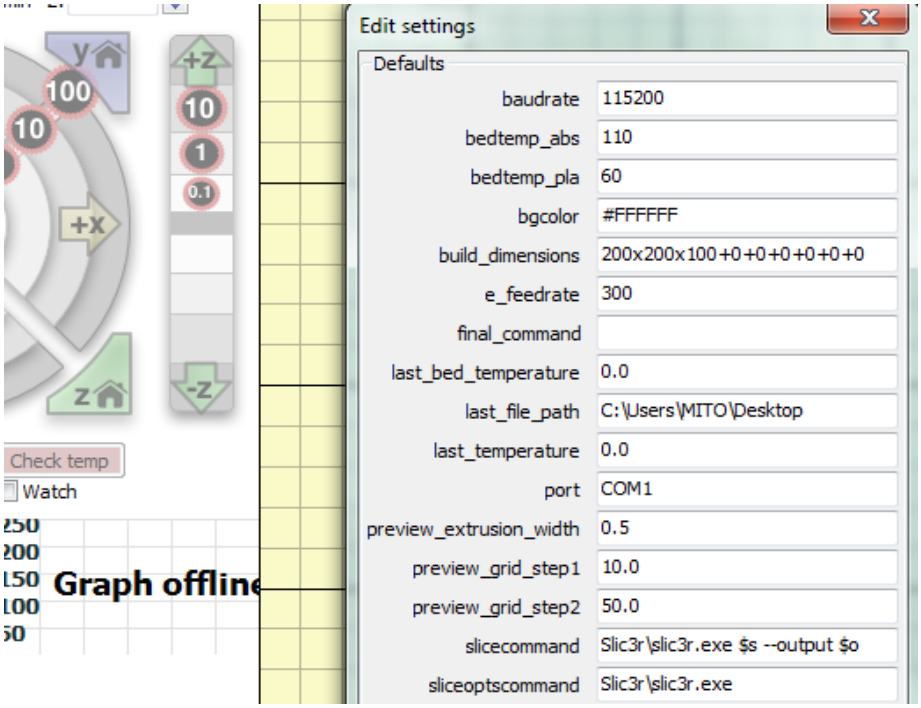


Figura 2.3 Opciones de Pronterface

2.5 SLIC3R

Para generar el patrón de la extrusora, necesitamos utilizar una aplicación llamada rebanador (slicer) la cual toma nuestro modelo entero en 3D y lo corta en capas adecuándolo para la impresión 3D. Este proceso crea el código que le dice a la impresora 3D donde mover la extrusora, cuando extrudir el plástico y cuanto plástico debe extrudir. Estos comandos son conocidos como código-G, los cuales son enviados desde nuestro software de control de la impresora al

firmware de nuestra electrónica, la cual es responsable de interpretar estos códigos para controlar los motores y calentadores de la impresora. [24] [pág. 37]

Este programa cortador es uno de los más populares debido a que es fácil de usar, posee asombrosos tiempos de impresión y notable calidad de impresión. Anteriormente carecía de muchas características que otros cortadores poseían, pero con el tiempo estas falencias se solucionaron.

Donde otros programas piden a los usuarios hacer muchos cálculos sobre la extrusión del filamento, Slic3r calcula todos estos valores sólo con ingresar ciertos parámetros de la impresora, del filamento y de la configuración de la impresión deseada. [24] [pág. 40]

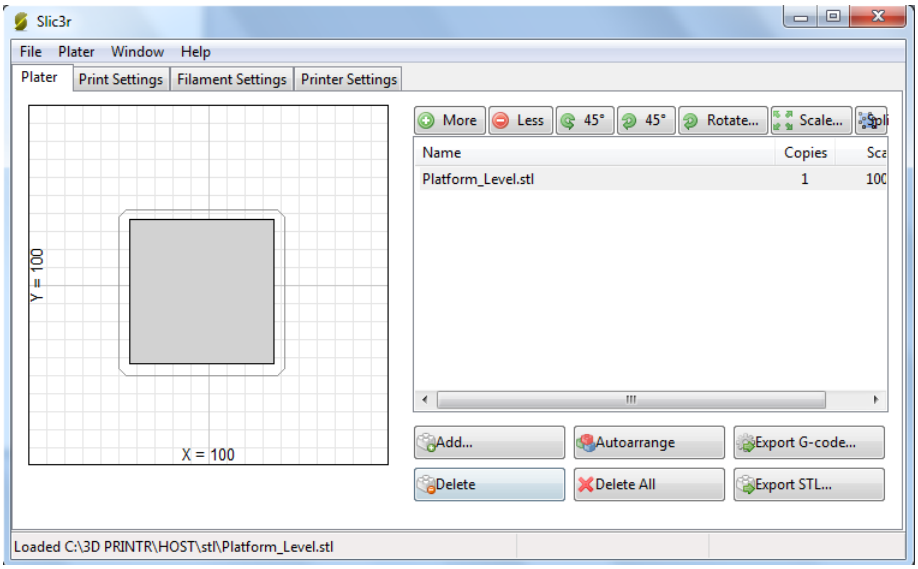


Figura. 2.4 Entorno de Slic3r

2.6 ARDUINO MEGA

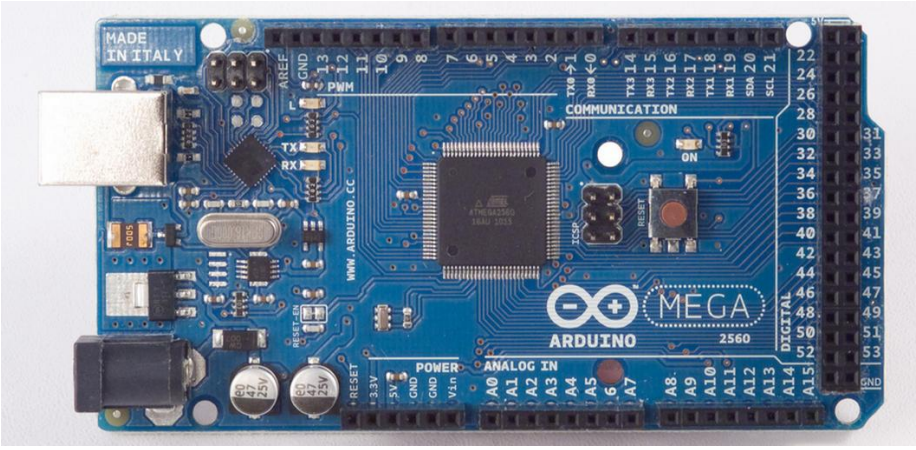


Figura 2.5 Arduino Mega2560 [11]

El Arduino Mega es una tarjeta microcontroladora basada en el ATmega2560, tiene 54 pines de entradas/salidas digitales (de las cuales 15 pueden ser usadas como salidas PWM), 16 entradas analógicas, 4 UARTS (puertos serial para hardware), un cristal oscilador de 16MHz, conexión USB, entrada de corriente, conector ICSP y botón de reset. [11]

El Mega es compatible con la mayoría de shields (tarjetas adaptables de uso específico) diseñados para el Arduino Duemilanove o Diecimila. [11]

2.6.1 RESUMEN DE CARACTERÍSTICAS DEL ATMEGA2560

- Microcontrolador: ATmega2560
- Voltaje de Funcionamiento: 5V
- Voltaje de entrada (recomendado): 7-12V
- Voltaje de entrada (límites): 6-20V
- Pines E/S digitales: 54 (15 proporcionan salida PWM)
- Pines de entrada analógica: 16
- Corriente DC por pin E/S: 40mA
- Corriente DC para pin de 3.3V: 50mA
- Memoria flash: 256 KB de las cuales 8 KB las usa el gestor de arranque (bootloader)

- SRAM: 8KB
- EEPROM: 4KB
- Velocidad de reloj: 16MHz

2.6.2 ALIMENTACIÓN

El Arduino Mega puede ser alimentado vía USB o con una fuente de alimentación externa. El origen de la alimentación se selecciona automáticamente.

Las fuentes de alimentación externas (no-USB) pueden ser un adaptador AC/DC o una batería. El adaptador se puede conectar usando un conector macho de 2.1mm con centro positivo en el conector hembra de la placa. La placa puede trabajar con una alimentación externa de entre 6 a 20 voltios. Si el voltaje suministrado es inferior a 7V el pin de 5V puede proporcionar menos de 5 Voltios y la placa puede volverse inestable, si se usan mas de 12V los reguladores de voltaje se pueden sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios. [11]

Los pines de alimentación son los siguientes:

- **VIN.** La entrada de voltaje a la placa Arduino cuando se está usando una fuente externa de alimentación (en opuesto a los 5 voltios de la conexión USB). Se puede proporcionar voltaje a través de este pin, o, si se está alimentado a través de la conexión de 2.1mm, acceder a ella a través de este pin.

- **5V.** Este pin de salida entrega un voltaje regulado de 5V desde el regulador de la tarjeta. La tarjeta también puede ser alimentada a través del conector jack (7-12V), el conector USB (5V), en el pin VIN de la tarjeta (7-12V). Al entregar voltaje a través de los pines de 5V o 3,3V se evita el regulador, y se puede dañar la tarjeta.

- **3V3.** Una fuente de voltaje a 3.3 voltios generada en la tarjeta. La corriente máxima soportada 50mA.

- **GND.** Pines de toma de tierra. [11]

- **IOREF.** Este pin en la placa del Arduino entrega el voltaje de referencia con el cual opera el microcontrolador. Una tarjeta configurada apropiadamente puede leer el voltaje del pin IOREF y seleccionar la fuente de alimentación adecuada o habilitar los traductores de voltaje en las salidas para trabajar con 5V o 3,3V.

2.6.3 MEMORIA

El ATmega2560 tiene 256KB de memoria flash para almacenar código (de los cuales 8KB son usados para el arranque del sistema), 8 KB de memoria SRAM y tiene 4KB de EEPROM (los cuales puede ser leídos y escritos con la librería EEPROM). [11]

2.6.4 ENTRADAS Y SALIDAS

Cada uno de los 54 pines digitales pueden utilizarse como entradas o como salidas usando las funciones pinMode(), digitalWrite(), y digitalRead() . Las E/S operan a 5 voltios, cada pin puede proporcionar o recibir una intensidad máxima de 40mA y tienen una resistencia pull-up interna (desconectada por defecto) de 20-50kOhms. Además, algunos pines tienen funciones especializadas:

- **Serial: 0 (RX) y 1 (TX), Serial 1: 19 (RX) y 18 (TX); Serial 2: 17 (RX) y 16 (TX); Serial 3: 15 (RX) y 14 (TX).** Usado para recibir (RX) transmitir (TX) datos TTL serial. Los pines 0 y 1 están también conectados a los pines correspondientes del chip serial ATmega16U2 USB-TTL. [11]

- **Interrupciones Externas: 2 (interrupción 0), 3 (interrupción 1), 18 (interrupción 5), 19 (interrupción 4), 20 (interrupción 3), y 21 (interrupción 2)**

Estos pines se pueden configurar para lanzar una interrupción en un valor LOW(0V), en flancos de subida o bajada (cambio de LOW a HIGH(5V) o viceversa), o en cambios de valor. Ver la función attachInterrupt() para más detalles.[11]

- **PWM: de 2 a 13 y de 44 a 46.** Proporciona una salida PWM de 8bits con la función analogWrite().[11]

- **LED: 13.** Hay un LED integrado en la placa conectado al pin digital 13, cuando este pin tiene un valor HIGH(5V) el LED se enciende y cuando este tiene un valor LOW(0V) este se apaga. [11]

El Mega2560 tiene 16 entradas analógicas, y cada una de ellas proporciona una resolución de 10bits (1024 valores diferentes). Por defecto se miden de tierra a 5 voltios, aunque es posible cambiar la cota superior de este rango usando el pin AREF y la función analogReference().

Otros pines en la placa son:

- **AREF.** Voltaje de referencia para las entradas analógicas. Usado con analogReference().

- **Reset.** Se le debe suministrar un valor LOW (0V) para reiniciar el microcontrolador. Típicamente usado para añadir un botón de reset a los shields que bloquean el acceso a este botón en la placa. [11]

2.6.5 COMUNICACIONES

El software incluye un monitor de puerto serie que permite enviar y recibir información textual de la placa Arduino. Los LEDS RX y TX de la placa parpadearan cuando se detecte comunicación transmitida través del chip FTDI y la conexión USB (no parpadearan si se usa la comunicación serie a través de los pines 0 y 1). [11]

2.6.6 PROGRAMACIÓN

El Arduino Mega se puede programar con el software Arduino. El ATmega2560 en el Arduino Mega viene precargado con un gestor de arranque (bootloader) que permite cargar nuevo código sin necesidad de un programador por hardware externo. Se comunica utilizando el protocolo STK500 original.

También podemos omitir el gestor de arranque y programar directamente el microcontrolador a través del puerto ICSP (In Circuit Serial Programming).

2.6.7 REINICIO AUTOMÁTICO POR SOFTWARE

En vez de necesitar reiniciar presionando físicamente el botón de reset, el Arduino Mega está diseñado de manera que es posible reiniciar por software desde el ordenador donde esté conectado.

Una de las líneas de control de flujo (DTR) del ATmega8U2 está conectada a la línea de reinicio del ATmega2560 a través de un condensador de 100 nanofaradios. Cuando la línea se pone a LOW (0V), la línea de reinicio también se pone a LOW el tiempo suficiente para reiniciar el chip.

El software de Arduino utiliza esta característica para permitir cargar los sketches con solo apretar un botón del entorno. Dado que el gestor de arranque tiene un lapso de tiempo para ello, la activación del DTR y la carga del sketch se coordinan perfectamente. [11]

El Mega contiene una pista que puede ser cortada para deshabilitar el auto-reset. Las terminaciones a cada lado pueden ser soldadas entre ellas para rehabilitarlo. Están etiquetadas con "RESET-EN".

También podemos deshabilitar el auto-reset conectando una resistencia de 110 ohms desde el pin 5V al pin de reset. [11]

2.6.8 PROTECCIÓN CONTRA SOBRETENSIONES EN USB

El Arduino Mega tiene un multifusible reiniciable que protege la conexión USB de tu ordenador de cortocircuitos y sobretensiones. Aparte que la mayoría de ordenadores proporcionan su propia protección interna, el fusible proporciona una capa extra de protección.

Si más de 500mA son detectados en el puerto USB, el fusible automáticamente corta la conexión hasta que el cortocircuito o la sobretensión desaparezcan. [11]

2.6.9 PINES DEL ARDUINO MEGA 2560

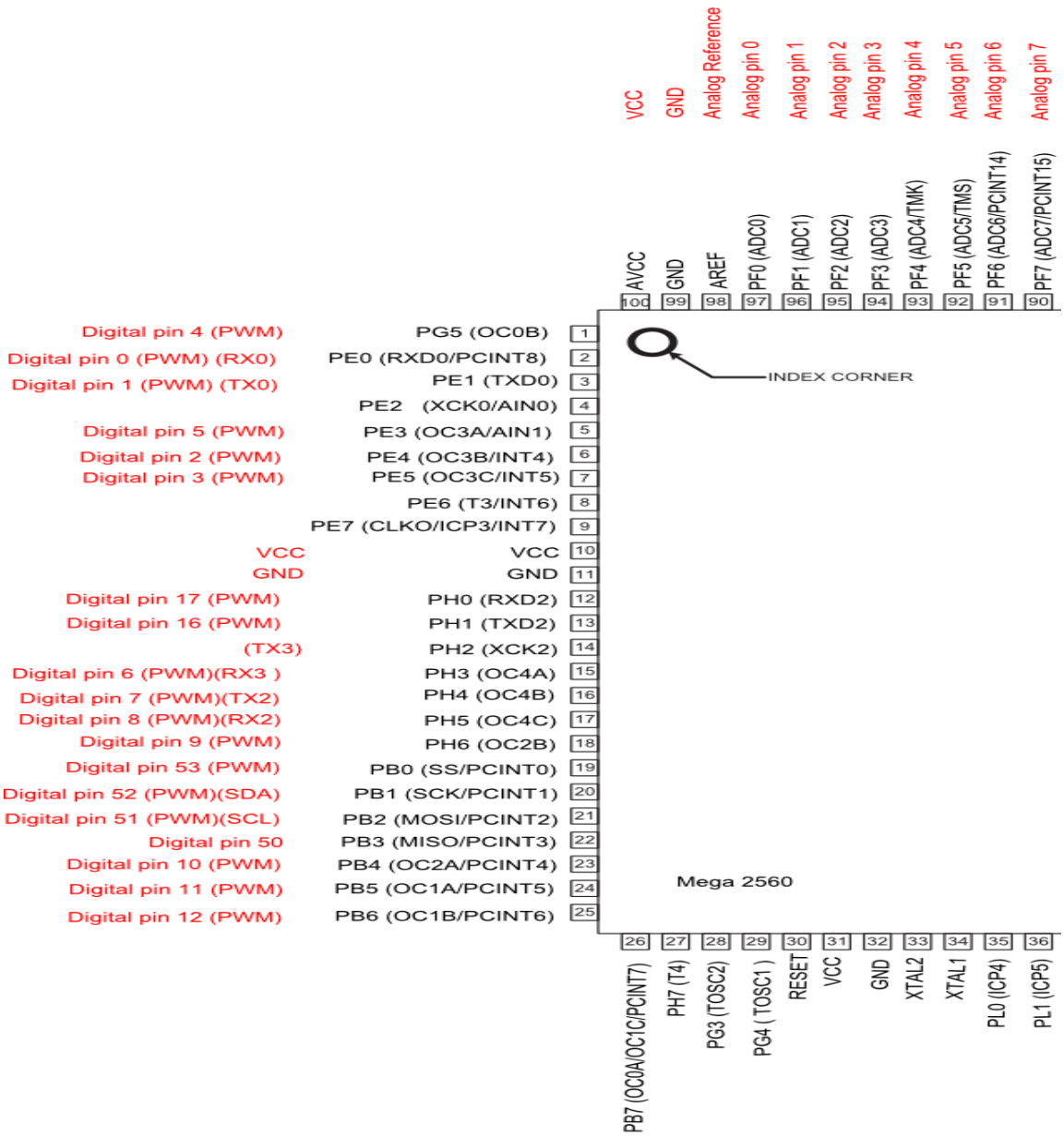


Figura 2.6 Mapeo de pines del Arduino Mega 2560 parte 1. [26]

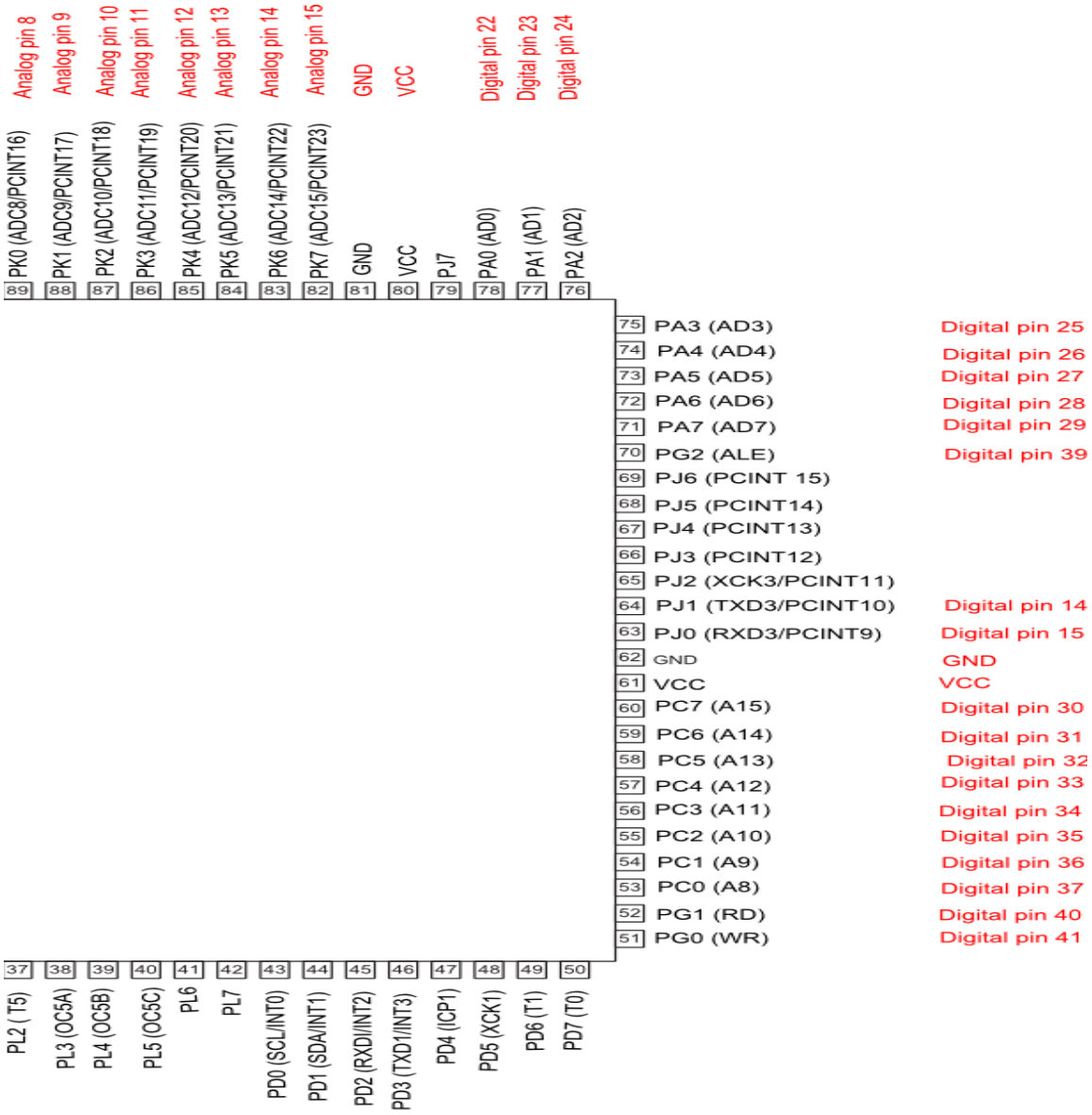


Figura 2.7 Mapeo de pines del Arduino Mega 2560 parte 2. [26]

2.7 MANEJO DE MOTORES DC

Cuando se planea un proyecto en el cual exista algún tipo de movimiento en la mayoría de los casos será necesario el uso de motores. El manejo de motores requiere corrientes más elevadas de las que el Arduino puede entregar de manera segura en sus pines de salida, por lo que se necesitará utilizar un transistor para asegurar de que se tendrá suficiente corriente para el motor y diodos para la protección del Arduino. [23] [pág. 99]

Una fuente de alimentación externa (que no exceda el voltaje de alimentación del motor), un potenciómetro de 10K ohm para el control de velocidad, un adecuado transistor de potencia NPN diseñado para cargas de corriente elevadas y un diodo 1N4001 son materiales comúnmente utilizados en la electrónica para formar este circuito controlador de un motor DC. [23] [pág. 100]

El transistor funcionará como un switch digital, el cual manejará una alta corriente (la corriente del motor) mediante una pequeña corriente que se aplicará a la base. [23] [pág. 102]

Cuando la alimentación del motor es removida, los campos magnéticos colapsan, estos campos que colapsan pueden producir un voltaje inverso el cual

se aplicará en los conductores y puede dañar seriamente el Arduino. [23] [pág. 103]

Si el motor produjera una fuerza contra electromotriz enviaría una corriente en retroceso a través del conductor, es aquí cuando el diodo actuaría como una válvula de una vía para prevenir que esto ocurra, permitiendo que sólo exista flujo de corriente en un sentido (del Arduino hacia el motor). [23] [pág. 103]

Por lo tanto hay que siempre intentar usar diodos en los circuitos donde sea que haya peligro de que el voltaje se invierta, o por error del usuario o vía fenómenos tales como la fuerza contra electromotriz. [23] [pág. 104]

2.7.1 MOTORES DE PASO

Un motor de paso es un motor DC en el cual la rotación puede dividir en un número de pasos más pequeños. Esto se puede hacer gracias a que se tiene un rotor de hierro con forma de engranaje unido a los ejes dentro del motor. Alrededor del exterior del motor son electroimanes con dientes. Cuando se energiza una bobina hace que los dientes del rotor de hierro se alineen con los dientes del electroimán. Los dientes del siguiente electroimán están ligeramente desplazados del primero, cuando se energiza y la primera bobina es apagada provoca que el eje gire ligeramente hacia el siguiente electroimán. Este proceso

es repetido por muchos de los electroimanes que están dentro hasta que los dientes estén casi alineados con el primer electroimán y el proceso se repite.

Cada vez un electroimán es energizado y el rotor se mueve ligeramente, esto quiere decir que se está llevando a cabo “un paso”. Para hacer girar el rotor en sentido contrario se invierte la secuencia de los electroimanes energizando el rotor. El trabajo del Arduino es aplicar el comando apropiado (High or Low) a las bobinas en la secuencia y velocidad correctas para permitirle girar al eje. Todo esto es lo que sucede de manera oculta en la librería stepper.h. [23] [pág. 211]

Figura. 2.8 Diferentes tipos de motores de paso. [23] [pág. 212]

2.8 LA ELECTRÓNICA

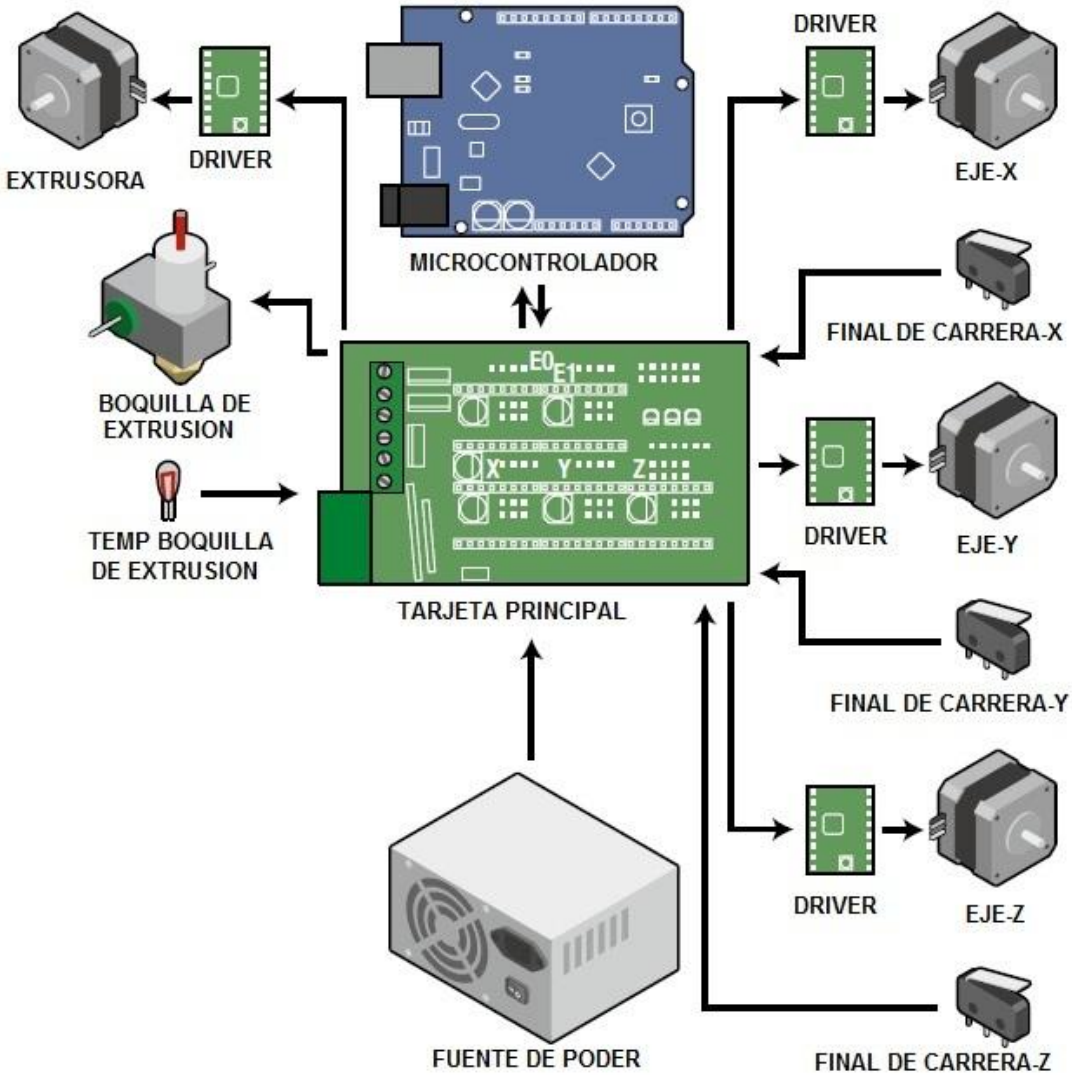


Figura 2.9 Plano de la electrónica de una impresora 3D [24] [pág. 30]

La electrónica de nuestra impresora 3D incluye muchas partes diferentes, trabajando juntas, para construir las impresiones 3D. Estos componentes incluyen un microcontrolador, una tarjeta principal, controladores de motores y motores de paso, una boquilla de extrusión y partes como finales de carrera o interruptores de límite y sensores de temperatura. Para tener una idea de cómo estas piezas diferentes trabajan juntas observemos la Fig. 3.3 [24][pág. 30]

En este gráfico de la electrónica, podemos ver las diferentes partes del sistema electrónico interconectadas con flechas que representan la dirección de control desde un componente al siguiente. La parte central de toda la electrónica es una tarjeta principal controladora que conecta todos los diferentes componentes, requeridos por la impresora 3D, con el microcontrolador; que es en esencia el cerebro de todo el sistema. Para conmutar la alta corriente asociada con el calentador de la extrusora, la tarjeta principal tendrá un hardware especializado de switcheo estimado para las cargas máximas necesarias.

La tarjeta principal tiene la capacidad de leer sensores de temperatura basados en resistencias llamados Termistores. La tarjeta principal es también la fuente de energía central de todo el sistema, tomando la energía de una fuente de poder y distribuyendo la misma a otros sistemas como sea necesario. La tarjeta principal necesitará también un modo de comunicación con los interruptores de final de

carrera en cada de uno de los ejes para permitir a la impresora posicionar su cabezal de impresión antes de imprimir. [24][pág. 30]

El microcontrolador, se encuentra en una tarjeta completamente separada, es una pequeña y simple computadora que ejecuta código especializado llamado firmware que le permite leer e interpretar sensores como los de temperatura y final de carrera, así como el manejo de motores utilizando drivers de para motores y conmutar altas cargas usando transistores para altas corrientes llamados MOSFET's. La mayoría del paquete electrónico para impresoras 3D personales utiliza el microcontrolador Arduino, que está basado en C (lenguaje de programación). [24][pág. 31]

2.8.1 CONTROLADOR POLOLU

Para controlar motores de paso, que son los que usamos en las impresoras 3D, la electrónica de control utiliza un controlador de motor de paso, para cada motor por separado. Esta pequeña tarjeta puede manejar solamente un motor de paso y es usada por unas pocas tarjetas controladoras para manejar cada uno de los ejes de la impresora.

2.8.2 TARJETA CONTROLADORA RAMPS

La RepRap Arduino Mega Pololu Shield conocida también como RAMPS es una tarjeta controladora que trabaja en conjunto con la tarjeta microcontroladora Arduino Mega, ambas se muestran en la Fig 2.10

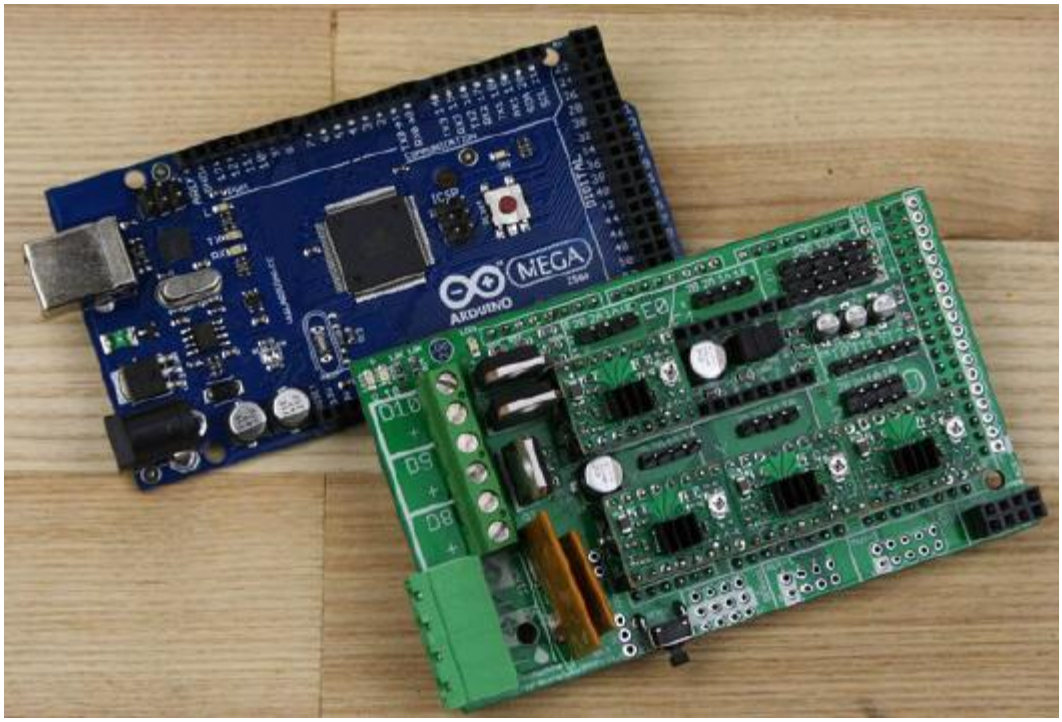


Figura 2.10 RepRap Arduino Mega Pololu Shield [24] [pág. 32]

Este paquete electrónico tienes las siguientes características:

- Soporta hasta 5 tarjetas controladoras para motores de paso Pololu 4988 o similares.

- 3 conmutadores de carga de alta potencia fusionados a 5amp y 11 amp para la extrusora.

- 6 conexiones para final de carrera.

- 3 conexiones para termistores.

- Una entrada de alimentación dual de 12 a 3 voltios a 16 amp como máximo.

RAMPS es una solución de dos tarjetas que consiste en una tarjeta microcontroladora de Arduino Mega y una placa especialmente diseñada que se conecta encima del Arduino. El Arduino mega proporciona el cerebro a la plataforma, mientras que la placa proporciona el hardware de conmutación para los calentadores. [24] [pág. 32]

CAPÍTULO 3

FUNCIONAMIENTO DE LA IMPRESORA, EJERCICIOS PREVIOS Y PROYECTO FINAL

3.1 CARACTERÍSTICAS DEL SISTEMA DE IMPRESIÓN

Para poder interactuar con nuestra impresora 3D debemos utilizar las diferentes aplicaciones que forman la cadena de herramientas para el manejo de la impresora. Dentro de los elementos que forman esta cadena estan: componentes electrónicos, el firmware, el software de control y el software de corte.

Necesitamos seguir algunos pasos de forma ordenada para convertir una gráfica digital 3D en un objeto plástico real. [24][pág.27]

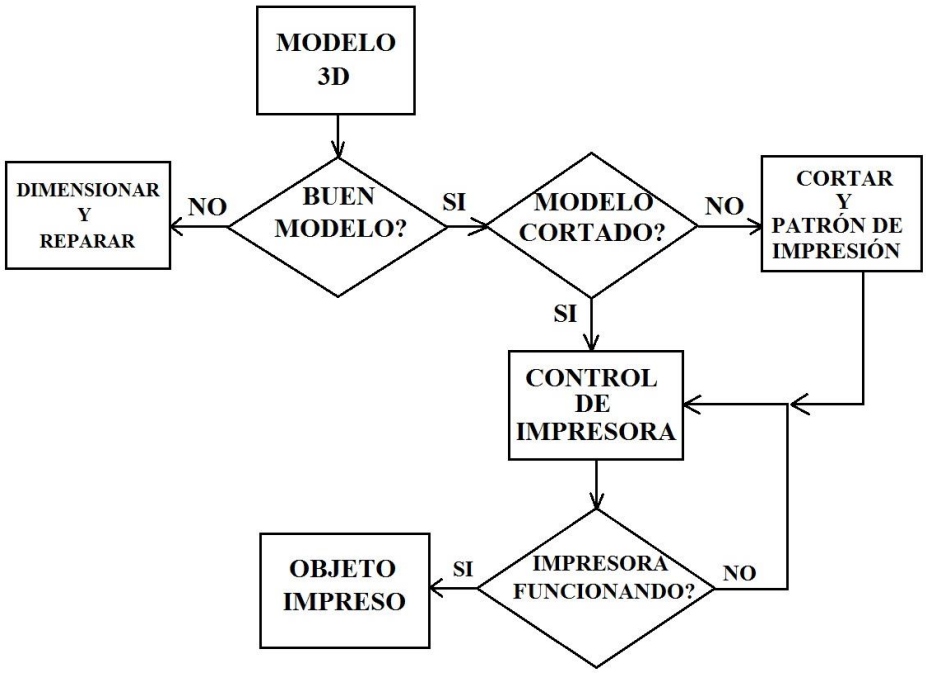


Figura. 3.1 Flujo organizacional del proceso para impresión 3D [24] [pág. 28]

El proceso entero comienza con un buen modelo 3D. Generalmente el modelo que buscamos estará en formato de archivo .stl, además este modelo deberá estar dimensionado apropiadamente para el tamaño de nuestra impresora, de no ser así debemos arreglarlo con la ayuda de algún software. [24][pág. 28]

3.2 CADENA DE HERRAMIENTAS EN LA IMPRESIÓN

El diagrama de flujo de trabajo no sería posible sin una serie de aplicaciones y hardware que necesitaremos usar en cada etapa. Estas piezas individuales juegan un papel crucial en el proceso de impresión y son las que conforman la cadena de herramientas de la impresora 3D.

La cadena de herramientas comienza con nuestro programa administrador y controlador (host software) que incluye: el programa cortador; el código, conocido como firmware cargado en las tarjetas electrónicas; así como la electrónica y el cableado. [24][pág. 29]

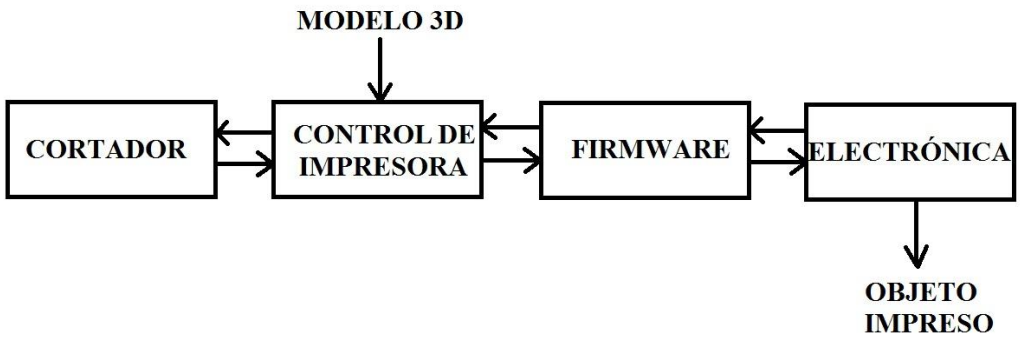


Figura. 3.2 Cadena de herramientas de la impresora 3D [24] [pág. 29]

La aplicación de control de la impresora procesará el modelo 3D y lo enviará al software cortador si es necesario. Luego el control de la impresora se comunica con un grupo especializado de código llamado firmware, el cual se ejecuta en la plataforma de la electrónica. El firmware controla el hardware electrónico para construir nuestro objeto 3D de acuerdo a las instrucciones recibidas desde el control de la impresora y envía datos como, temperatura, posicionamiento y otra información de regreso a la aplicación de control. [24][pág. 29]

3.3 EJERCICIOS PREVIOS

3.3.1 COMUNICACIÓN CON LA TARJETA ARDUINO

En este ejercicio vamos a configurar 2 opciones básicas dentro del software de Arduino, para establecer una correcta conexión con la tarjeta y luego cargaremos un programa de prueba para verificar que el programa fue cargado correctamente en el Arduino.

Dentro del programa Arduino en la pestaña “Herramientas” debemos configurar las opciones “Tarjeta” y “Puerto Serial”; en “Tarjeta” seleccionamos el modelo de la tarjeta Arduino que poseemos, en este caso seleccionamos Arduino Mega

2560; en “Puerto Serial” debemos seleccionar el puerto en el cual está conectado nuestro Arduino, para esto la tarjeta debe estar conectada a un puerto USB del CPU y en la ventana “Administrador de equipos” seleccionamos “Administrador de dispositivos” y dentro de este menú seleccionamos “Puertos (COM y LPT)” en donde podremos ver en qué puerto COM está conectada nuestra tarjeta con lo que regresamos al software Arduino en “Herramientas” y “Puerto Serial” escogemos el número de puerto COM del Arduino.

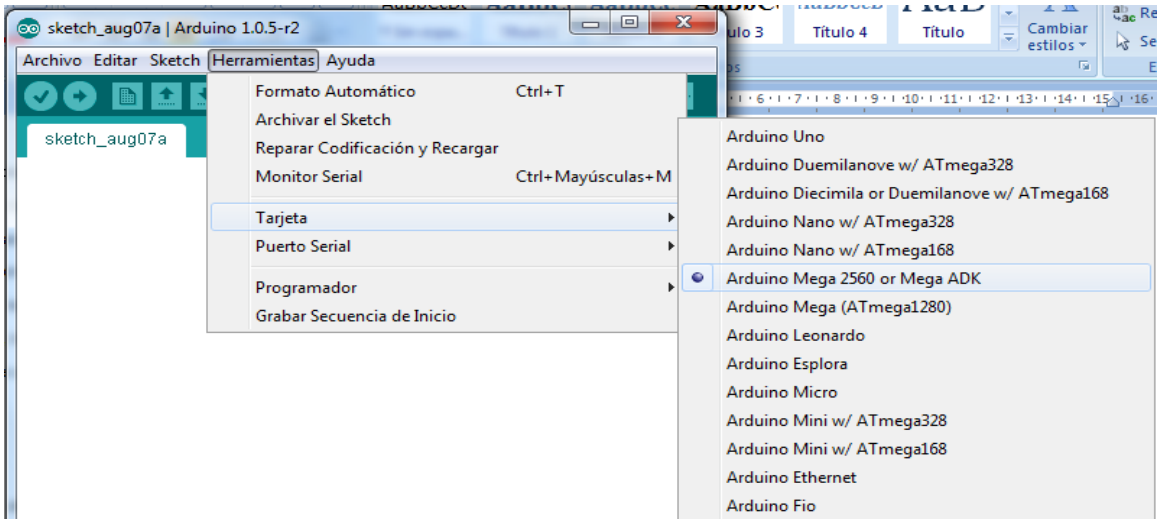


Figura 3.3 Configuración del software Arduino, selección del modelo [23] [pág. 9]

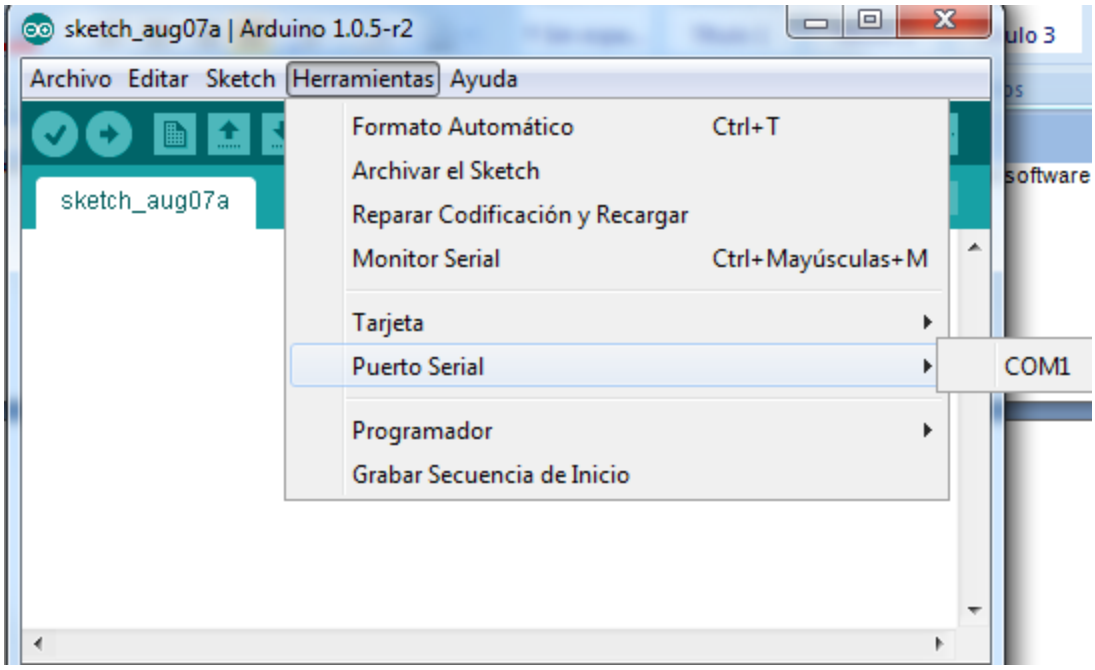


Figura 3.4 Configuración del software Arduino, puerto serial [23] [pág. 10]

Luego de realizar estos pasos estamos listos para cargar nuestro primer programa en la tarjeta. Para esto vamos a “Archivo”, “Ejemplos”, “Basics” y escogemos “Blink”. Para cargar el programa escogido vamos a presionar el botón Cargar que es una flecha hacia la derecha.

Figura 3.5 Botón cargar programa.

Blink es un programa que va a poner a parpadear un Led de la tarjeta, con lo que vamos a comprobar si las configuraciones previas estuvieron correctas.

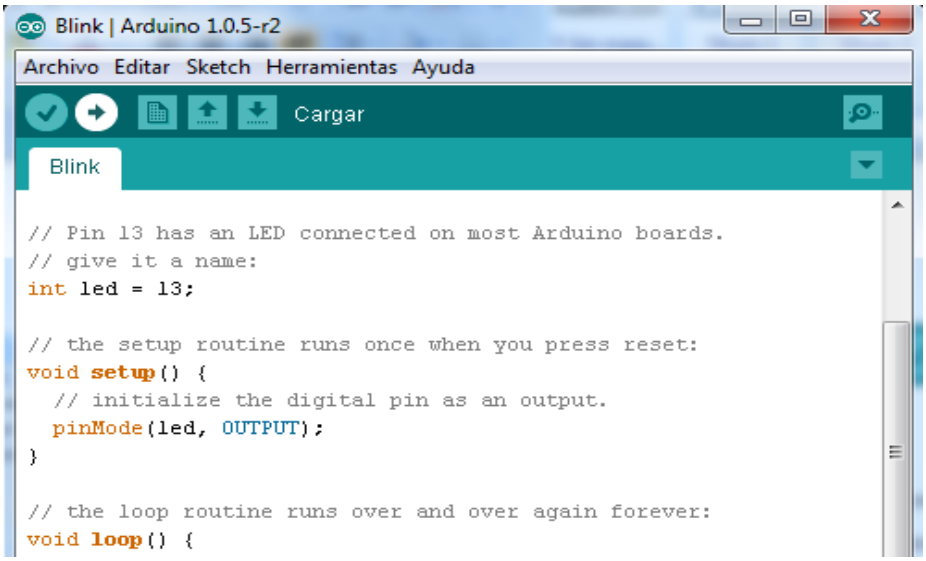


Fig. 3.6 Blink en la ventana de programación Arduino

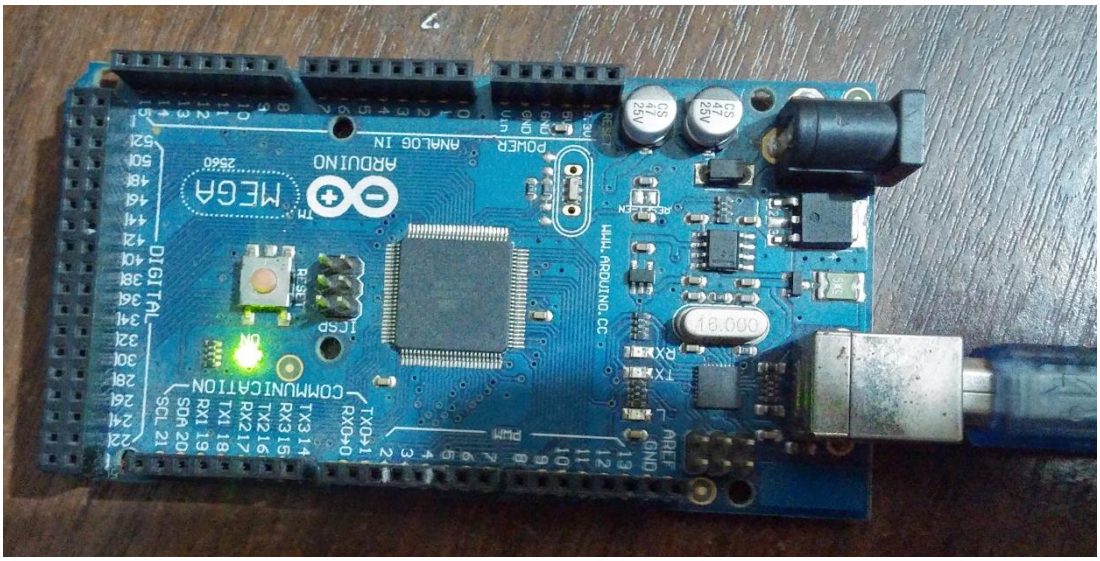


Fig. 3.7 Tarjeta Arduino conectada al CPU vía puerto USB

3.3.1.1 DIAGRAMA DE BLOQUES

Fig. 3.8 Diagrama de Bloques de BLINK

3.3.1.2 DIAGRAMA ASM

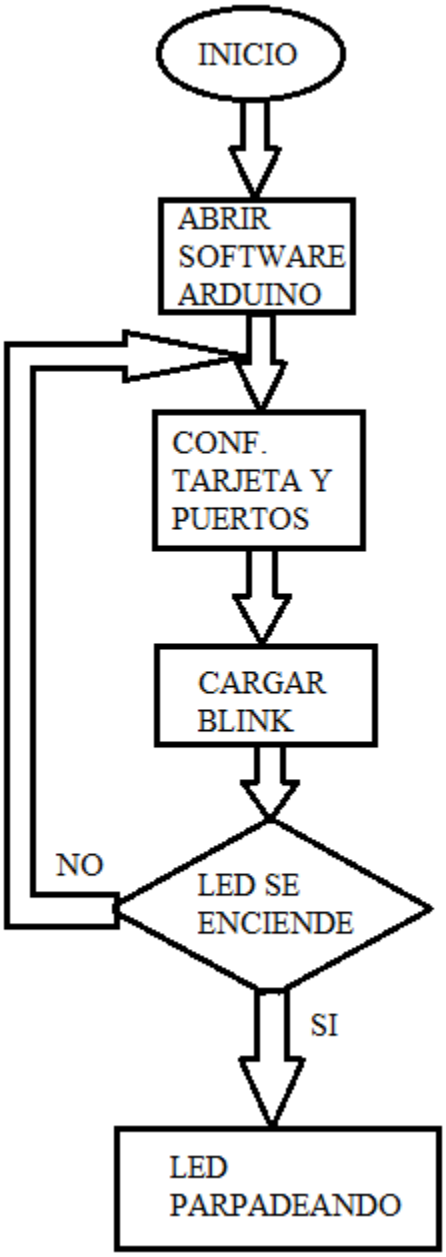


Fig. 3.9 Diagrama ASM para cargar BLINK en la tarjeta

3.3.1.3 CÓDIGO FUENTE

```
/* Blink
Enciende un LED durante un segundo, luego apaga el LED durante un segundo y asÍ repetitivamente
*/
 // Pin 13 tiene conectado un LED en la mayoría de las tarjetas Arduino.
// le damos un nombre al puerto #13
int led = 13;
// La rutina setup se ejecuta una vez cuando se presiona Reset:
void setup() {
  pinMode(led, OUTPUT); // inicializa el pin digital como una salida.
}
// La rutina loop se ejecuta una y otra vez infinitamente:
void loop() {
  digitalWrite(led, HIGH);   // enciende el Led (HIGH es el nivel del voltaje)
  delay(1000);               // provoca un retardo de 1000 mili segundos
  digitalWrite(led, LOW);    // apaga el Led haciendo que el voltaje este en LOW
  delay(1000);               // provoca un retardo de 1000 mili segundos
}
```

3.3.2 SEMÁFORO USANDO 3 COLORES DE LED

En el siguiente ejercicio mostraremos uno de las aplicaciones que posee la tarjeta Arduino como declarar puertos digitales como salidas, establecer los puertos digitales en una señal de ALTO o BAJO y también el uso de funciones temporizadoras que provocan retardos de milisegundos en la ejecución del programa.

Para esto vamos a utilizar 3 LEDs de 3 colores diferentes y 3 resistencias de 150ohm, la conexión la realizaremos tal como se muestra en la figura.

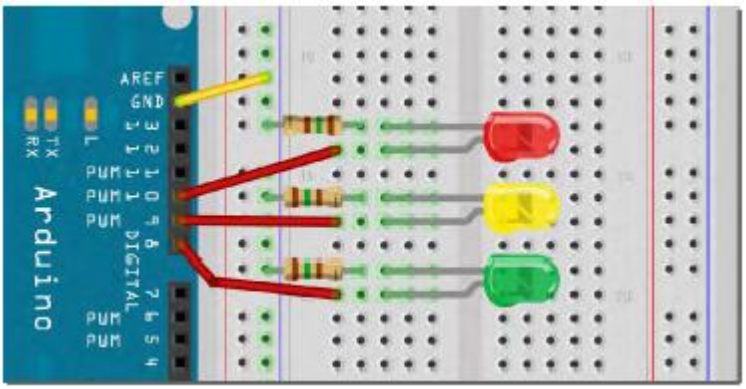


Figura 3.10 Esquemático de las conexiones de Semáforo [23][Pág. 36]

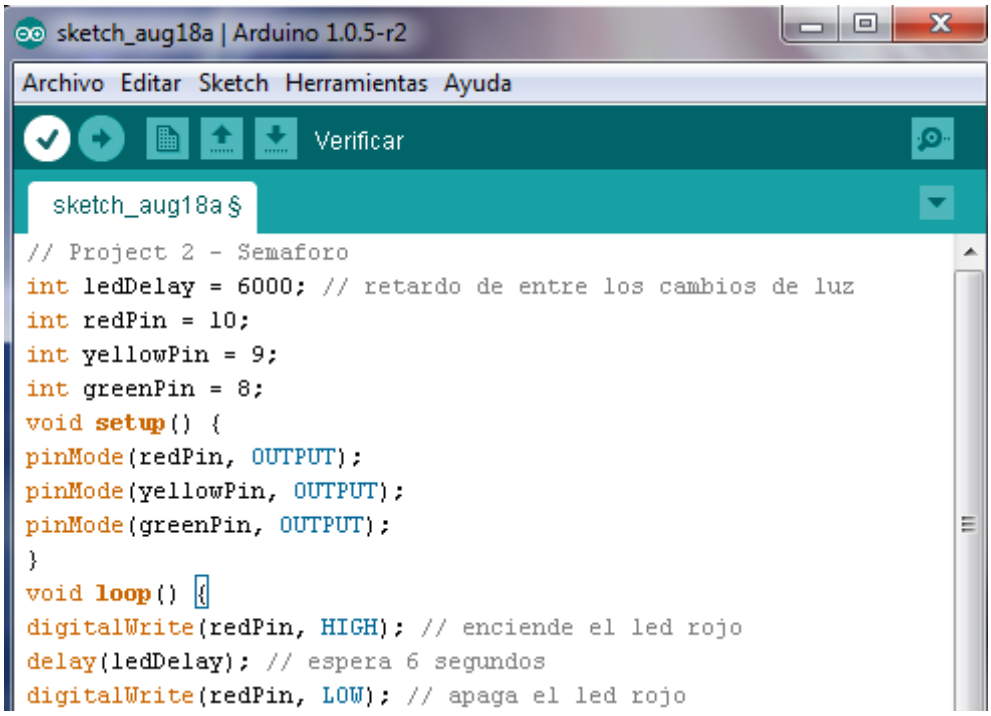


Figura 3.11 Compilación exitosa del código fuente “Semáforo”.

3.3.2.1 DIAGRAMA DE BLOQUES

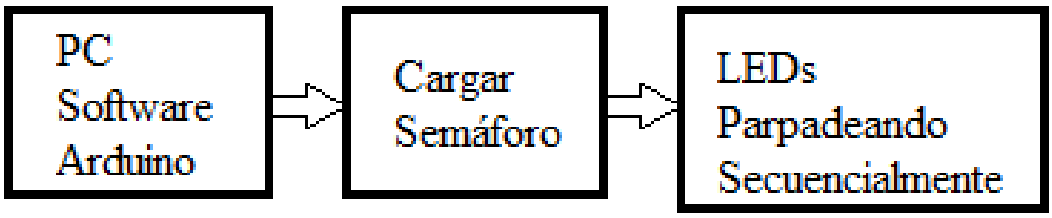


Figura 3.12 Diagrama de Bloques del programa Semáforo

3.3.2.2 DIAGRAMA ASM

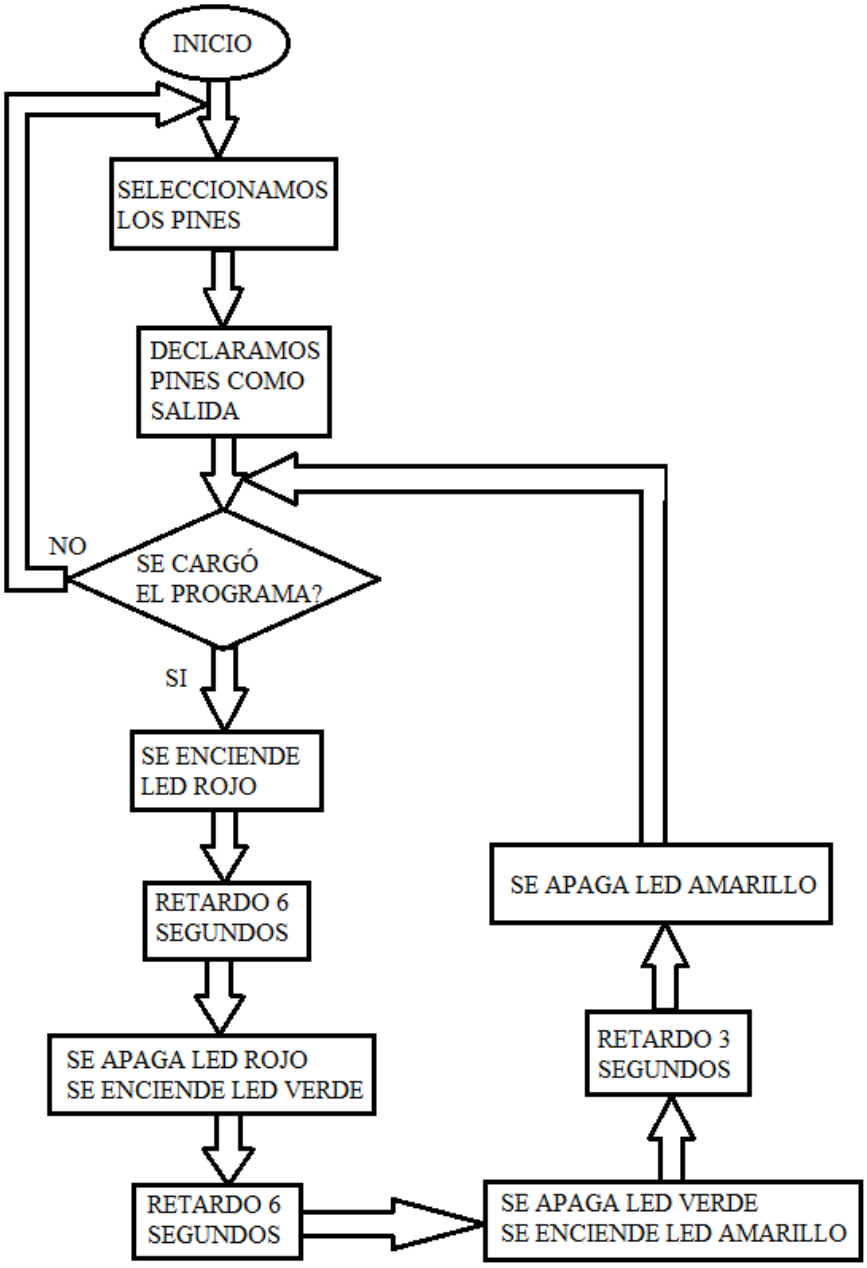


Figura 3.13 Diagrama ASM para el programa Semáforo

3.3.2.3 CÓDIGO FUENTE

```
// Project 2 - Semaforo
int ledDelay = 6000;        // retardo de entre los cambios de luz
int redPin = 10;            // se le asigna un valor a la variable entera redpin
int yellowPin = 9;
int greenPin = 8;
void setup() {
pinMode(redPin, OUTPUT);    // el pin 10 es declarado como salida
pinMode(yellowPin, OUTPUT);
pinMode(greenPin, OUTPUT);
}
void loop() {
digitalWrite(redPin, HIGH);    // enciende el led rojo
delay(ledDelay);               // espera 6 segundos
digitalWrite(redPin, LOW);     // apaga el led rojo
digitalWrite(greenPin, HIGH);  // enciende el led verde
delay(ledDelay);               // espera 6 segundos
digitalWrite(greenPin, LOW);   // apaga el led verde
digitalWrite(yellowPin, HIGH); // enciende el led amarillo
```

```
delay(3000);                    // espera 2 segundos
digitalWrite(yellowPin, LOW);   // apaga el led amarillo
// Ahora nuestro lazo se repite
}
```

3.4 SELECTOR DE TRES EXTRUSORAS DE DIFERENTE COLOR MEDIANTE ARDUINO Y L293D.

En nuestro proyecto vamos a utilizar la impresora Prusa DIY 3D Stuffmaker o también conocida como Reprap Prusa Mendel. Dentro de los cambios que vamos a realizar a esta impresora están:

- Reemplazar la tarjeta controladora original por una Arduino Mega 2560.
- Reemplazar los sensores ópticos de los finales de carrera por unos mecánicos.
- Reemplazar el firmware Sprinter por el firmware Marlin.
- Configuración de parámetros dentro de Marlin.

En lo que respecta a la electrónica y componentes que conforman esta impresora podemos apreciar sus elementos en la figura 2.9; de igual manera la lógica del funcionamiento para imprimir un objeto 3D se refleja en la gráfica de la figura 3.1.

Tal como se menciona, una parte del desarrollo de este trabajo fue instalar una nueva tarjeta controladora y la que utilizaremos es la Arduino Mega 2560 cuyos pines de conexión se detallan en la figura 2.6 y 2.7.

Otros cambios que se realizaron en el hardware de la impresora fueron los finales de carrera que de fábrica vienen de tipo óptico, fueron reemplazados por unos de tipo mecánico. El firmware que utiliza la tarjeta original es Sprinter y debido a que hemos encontrado que existe un programa que suaviza los acabados de los objetos impresos y que también los imprime a mayor velocidad nos decidimos por instalar Marlin en nuestro controlado Arduino.

Debido a todos estos cambios de hardware y software fue necesario realizar una configuración en le Firmware para el correcto funcionamiento de todos estos parámetros.

3.4.1 CONFIGURACIÓN DE PARÁMETROS DE MARLIN

Por defecto el firmware de Marlin viene configurado para 115200 Baudrate nosotros deshabilitamos este valor y activamos 250000 Baudrate.

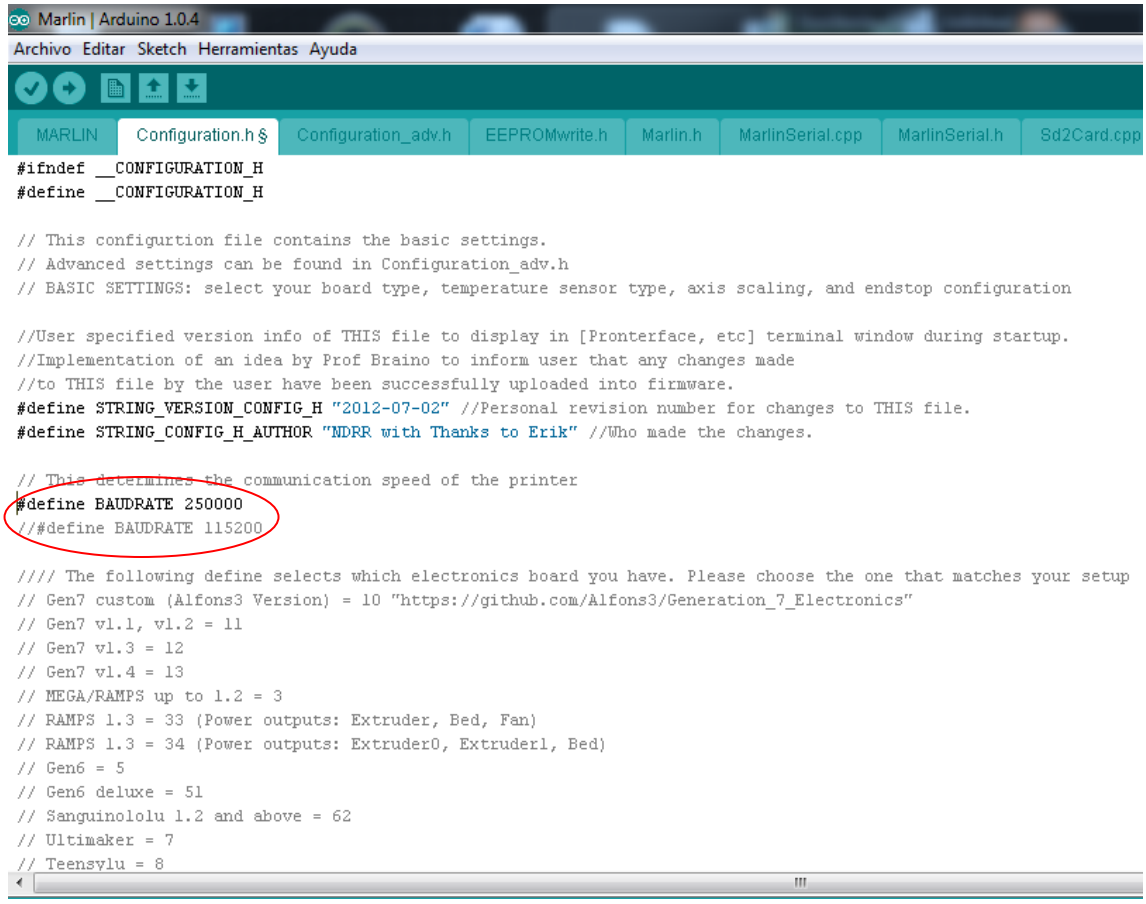


Figura 3.14 Cambio en la configuración del Baudrate en Marlin.

En lo que respecta a la configuración de los finales de carreras estos vienen definidos con su lógica en true para unos sensores ópticos, por seguridad al cambiar el tipo de sensores por los de tipo mecánico normalmente cerrado, cambiamos la lógica a False, para que al presionar el interruptor del final de carrera considere esta señal de corte como la orden para detener el motor del eje asignado y que además nos permita detener el movimiento del motor asignado en caso de que se desconecte uno de los cables del final de carrera.

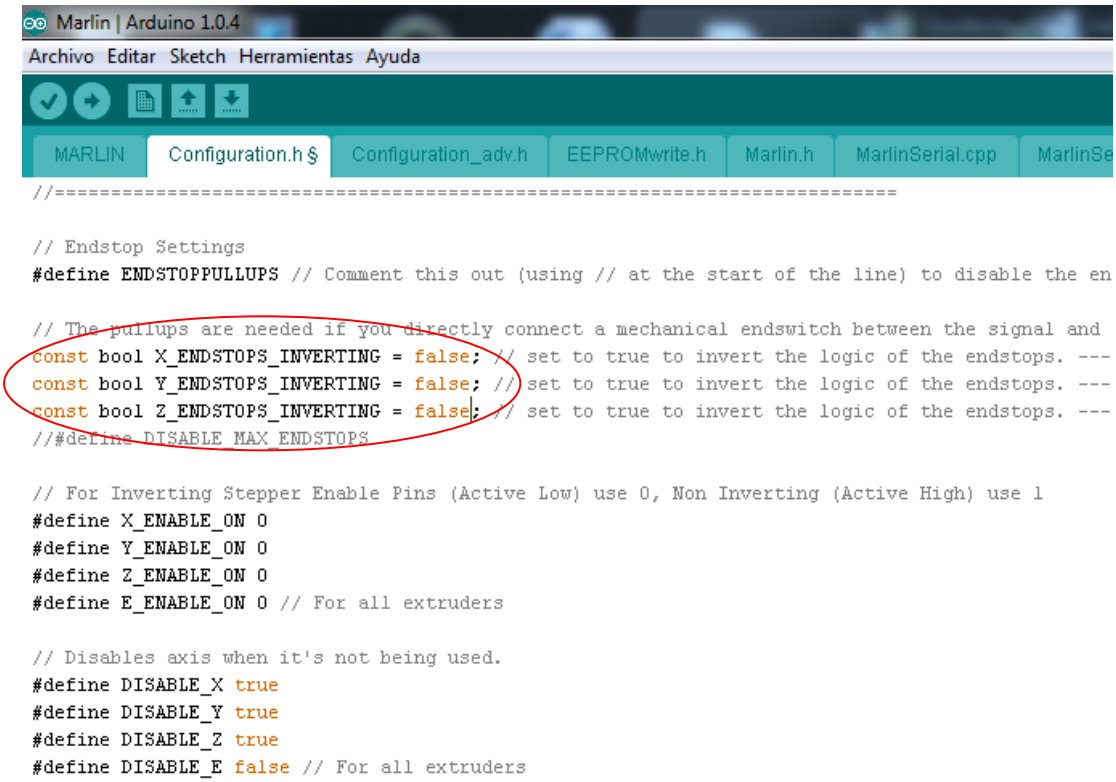


Figura 3.15 Configuración de la lógica de los finales de carrera.

3.4.2 DESCRIPCIÓN DEL PROYECTO FINAL

La selección de los colores (extrusoras) es controlada por un motor de paso bipolar mediante una tarjeta arduino y dos drivers l293d.

La preselección dependerá de las condiciones que se determinaron mediante el switch de 3 posiciones y las que son generadas por los sensores cuando se mueve el cabezal, con todas esas condiciones tendremos el color que deseamos.

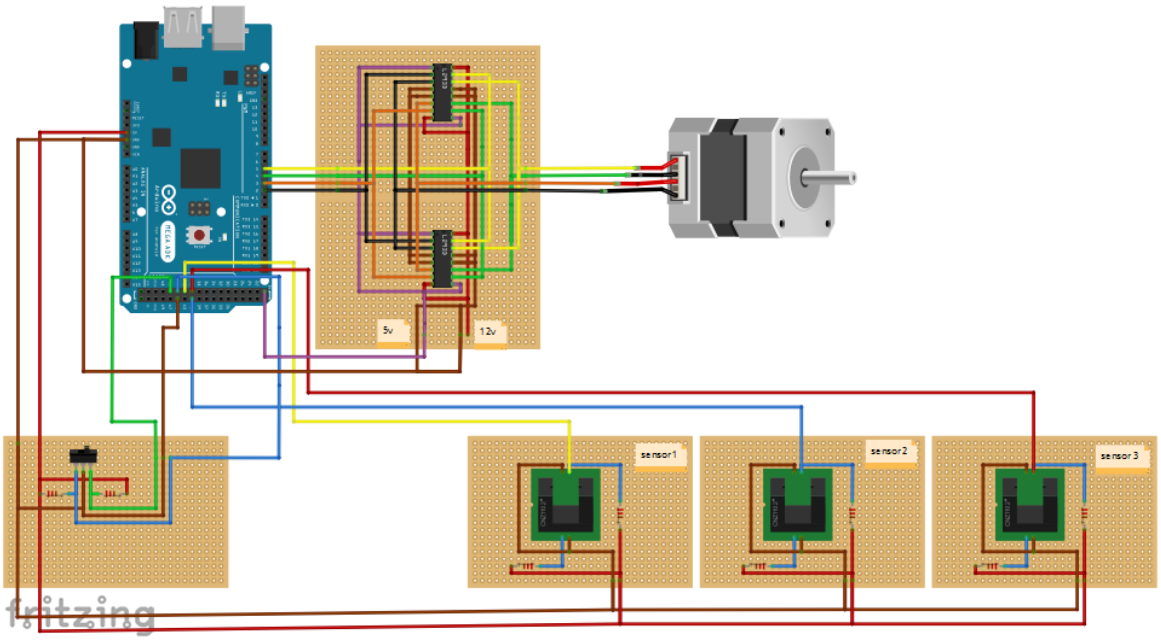


Figura 3.16 Esquema de conexiones del selector de colores

3.4.3 DIAGRAMA DE BLOQUES

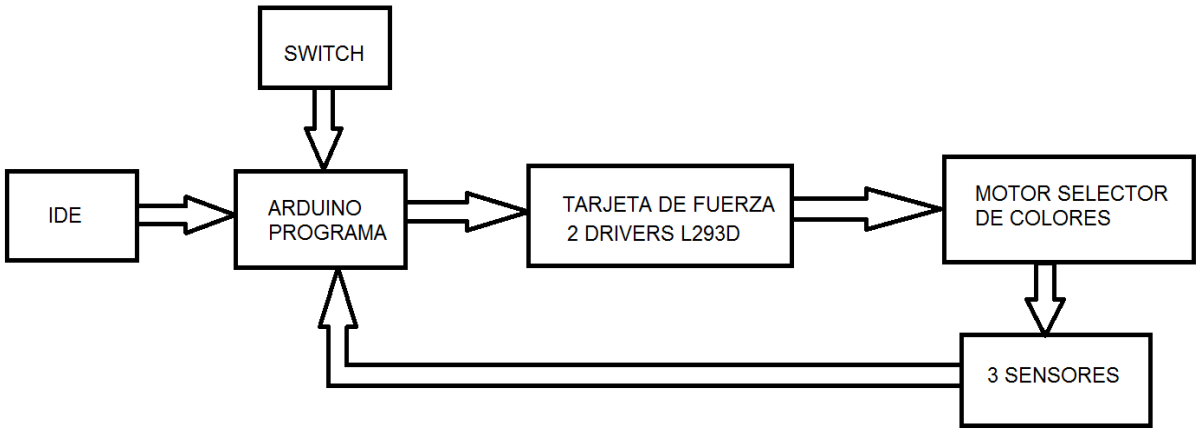


Figura 3.17 Diagrama de bloques del selector de colores

3.4.4 DIAGRAMA ASM

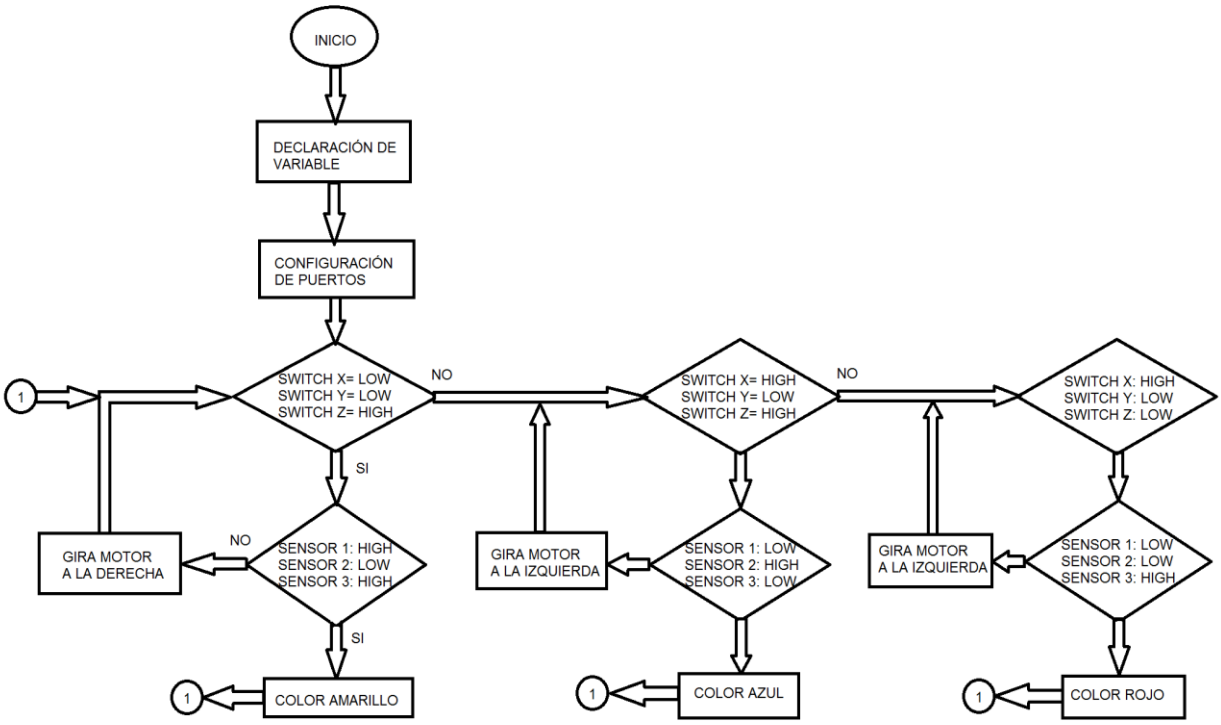


Figura 3.18 Diagrama ASM del selector de colores

3.4.5 DESCRIPCIÓN DEL ALGORITMO

- Se inicializan las variables.
- Se configuran los puertos referentes a las variables que se están utilizando.
- Se ejecuta el programa:
  - Seteo del switch (X Y Z).
  - Verificación de los sensores (1 2 3) dependiendo de la condición del switch (X Y Z).
  - Gira el motor para un determinado sentido por las condiciones previas.
  - Habilitará la salida del color elegido (amarillo, azul, rojo).

3.4.6 CÓDIGO FUENTE

```
//    SELECCION DE 3 COLORES


int sensor1=40;              //se declaran variables para los pines del arduino
int sensor2=41;
int sensor3=42;
int switchX=44;
int switchY=45;
int switchZ=46;
int salida1=48;
int salida2=49;
int salida3=50;
int A=2; int B=3;                    //variables para bobinas del motor
int C=4; int D=5;                    // variables para bobinas del motor
int mSecPaso=800;                    // variable de tiempo de conmutación
int vueltas_motor1 = 1 ;             // escribir el numero de vueltas
//   en void setup se declararán las entradas y salidas de los puertos
void setup()
{
```

```
pinMode(sensor1, INPUT);              //setea el pin 40 como una entrada digital
pinMode(sensor2, INPUT);
pinMode(sensor3, INPUT);
pinMode(switchX, INPUT);
pinMode(switchY, INPUT);
pinMode(switchZ, INPUT);
pinMode(salida1, OUTPUT);             //setea el pin 48 como una salida digital
pinMode(salida2, OUTPUT);
pinMode(salida3, OUTPUT);
pinMode(A, OUTPUT);
pinMode(B, OUTPUT);
pinMode(C, OUTPUT);
pinMode(D, OUTPUT);
digitalWrite(A, LOW);      //manda señal de Bajo=Off al pin 2 de la bobina
digitalWrite(B, LOW);
digitalWrite(C, LOW);
digitalWrite(D, LOW);
digitalWrite(salida1, LOW); //manda señal de Bajo=Off al pin 48 de la bobina
digitalWrite(salida2, LOW);
digitalWrite(salida3, LOW);
```

```
}
// Función para girar motor hacia la izquierda
void izquierdamotor1(){
//paso 1
digitalWrite(A, LOW);
digitalWrite(B, HIGH);
digitalWrite(C, HIGH);
digitalWrite(D, LOW);
//llama a la función y le da un valor en micro segundos
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
// Función para girar motor 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
//La función void loop es el lazo infinito que siempre se va a repetir
void loop()
{
// Lee las 3 señales del switch de 3 posiciones, compara cada uno de los
//estados y pregunta si cumplen con la condición
  if (digitalRead(switchX)==LOW && digitalRead(switchY)==LOW &&
digitalRead(switchZ)==HIGH)
   {
 //Lee las 3 señales que emiten los 3 sensores ópticos, compara cada uno de
//los estados y pregunta si cumplen con la condición
    if (digitalRead(sensor1)==LOW && digitalRead(sensor2)==HIGH &&
digitalRead(sensor3)==HIGH)   //Color Rojo
     {
      //Setea el pin de salida1 en Alto= ON y a los otros 2 pines en OFF
       digitalWrite(salida1,HIGH);
       digitalWrite(salida2,LOW);
```

```
      digitalWrite(salida3,LOW);
    }
   else
    {
     for(int i=0; i<vueltas_motor1; i++)
     {
      derechamotor1();          // motor gira hacia la derecha
      mSecPaso+=50;             //incrementa el valor de la variable
     }
    }
  }
 if (digitalRead(switchX)==HIGH && digitalRead(switchY)==LOW &&
digitalRead(switchZ)==HIGH)
  {
   if (digitalRead(sensor1)==HIGH && digitalRead(sensor2)==LOW &&
digitalRead(sensor3)==HIGH) //Color Azul
    {
     digitalWrite(salida1,LOW);
     digitalWrite(salida2,HIGH);
     digitalWrite(salida3,LOW);
```

```
    }
   else
    {
      for(int i=0; i<vueltas_motor1; i++)
      {
       izquierdamotor1();  // motor gira hacia la izquierda
       mSecPaso+=50;
      }
    }
  }
 if (digitalRead(switchX)==HIGH && digitalRead(switchY)==LOW &&
digitalRead(switchZ)==LOW)
  {
    if (digitalRead(sensor1)==HIGH && digitalRead(sensor2)==HIGH &&
digitalRead(sensor3)==LOW) //Color Amarillo
    {
      digitalWrite(salida1,LOW);
      digitalWrite(salida2,LOW);
      digitalWrite(salida3,HIGH);
    }
```

```
  else
   {
     for(int i=0; i<vueltas_motor1; i++)
     {
      izquierdamotor1();
      mSecPaso+=50;
     }
   }
}
```

CAPÍTULO 4

IMPLEMENTACIÓN DE EJERCICIOS DE PRUEBA Y SIMULACIÓN DEL PROYECTO

En este último capítulo detallamos que componentes usamos para armar cada uno de los circuitos. Mostraremos imágenes de todos los ejercicios implementados previamente a la construcción de nuestro trabajo final. Observaciones y recomendaciones para una correcta ejecución de cada uno de los ejercicios, son explicadas dentro de este capítulo que también involucra el proyecto final de este trabajo.

4.1 “COMUNICACIÓN CON LA TARJETA ARDUINO”

Una vez que tenemos instalado nuestro IDE (software Arduino 1.0.5), seleccionamos el modelo de la tarjeta para que se instalen los drivers y finalmente escogemos el puerto COM al cual estamos conectados; después de todo esto podemos para cargar cualquier programa en nuestra tarjeta.

El programa Blink es uno de los programas básicos que vienen pre instalados con el software Arduino y que nos permite visualizar directamente, con un led en la tarjeta, que la comunicación y el programa fueron cargados exitosamente.

Las luces de Rx y Tx deberían empezar a parpadear cuando se está transmitiendo la información desde la computadora hacia la tarjeta. Cuando en la ventana del IDE aparece “Done Uploading” o Carga Terminada, quiere decir que nuestro programa fue cargado con éxito y luego de esto las luces de Rx y Tx dejarán de parpadear.

El LED de la tarjeta Arduino viene conectado internamente al Pin #13 es por esto que el código fuente debe utilizar sólo este pin. El tiempo que dura el parpadeo del LED se puede modificar mediante el código fuente de Blink, este tiempo viene configurado inicialmente en 1segundo o 1000ms que es lo mismo.

4.1.1 IMPLEMENTACIÓN

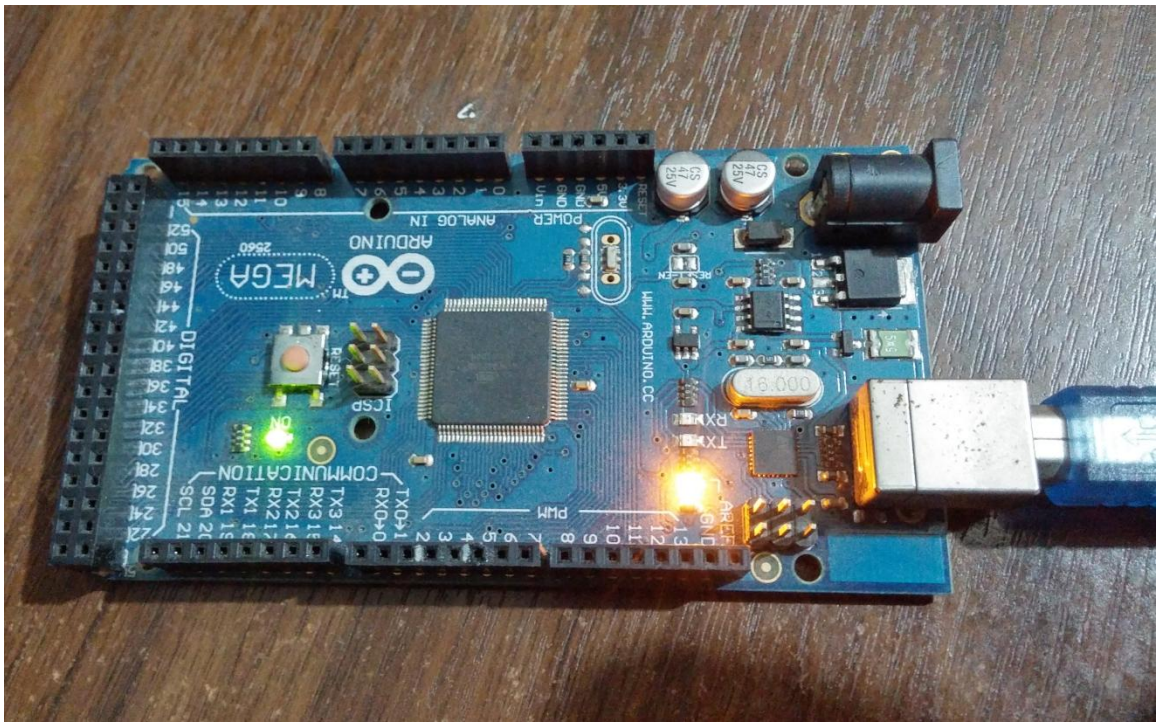


Figura 4.1 LED del Pin #13 encendido, comunicación exitosa con el PC

4.1.2 LISTADO DE MATERIALES

- Tarjeta Arduino ATmega 2560
- Cable adaptador USB

4.1.3 CONCLUSIONES

1) Siempre que conectamos la tarjeta a nuestro computador el LED de ON debe estar encendido indicando que la tarjeta esta energizada.

2) Durante la transmisión de cada programa se deben encender los LEDs de Rx y Tx y apagarse cuando finalice toda la carga.

4.1.4 RECOMENDACIONES

1) Se recomienda verificar muy bien el puerto COM que registra nuestro computador al conectarse la tarjeta y si al momento de la carga de un programa las luces de Rx y Tx no se encienden, lo mejor será probar en otro puerto USB de nuestro computador y actualizar esto en Herramientas—Puerto Serial del IDE.

4.2 “SEMÁFORO USANDO 3 COLORES DE LED”

En este proyecto vamos a simular el funcionamiento de un semáforo convencional en nuestro país pero va a trabajar de forma independiente sin la condición de un segundo semáforo.

Cuando se trabaja con diodos LED tenemos una polaridad (ánodo y cátodo) que es muy importante tener en cuenta para que estos se enciendan y no sufran daños, adicionalmente siempre que se trabaja con LED es necesario conectarles una resistencia para que fluya una corriente a través de ellos.

El semáforo que implementamos obedece directamente a la programación ya que en él vamos a determinar cuáles son los pines que vamos a utilizar como salidas digitales y también mediante la función “Delay()” o Retardo de tiempo, vamos a poder controlar los tiempos en que cada uno de los LEDs permanece encendido o apagado. El orden en que se encienden o se apagan los LEDs va a corresponder a la lógica de la programación y este también podrá ser modificado en caso de que queramos simular un semáforo de otro país.

4.2.1 IMPLEMENTACIÓN

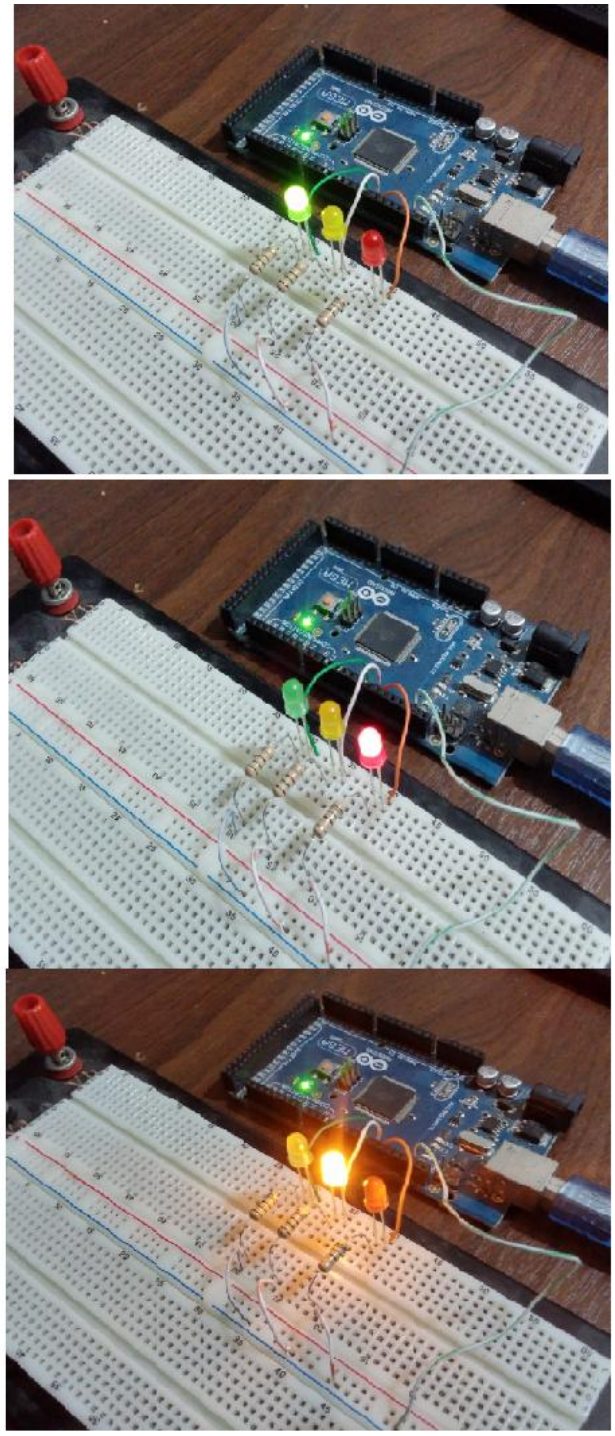


Figura 4.2 Semáforo tradicional temporizado utilizando 3 diodos LED

4.2.2 LISTA DE MATERIALES

- 3 diodos LED color Rojo, Amarillo y Verde.
- 3 resistencias de 150 ohm.
- 1 protoboard.
- Alambres de cobre.

4.2.3 CONCLUSIONES

1) La función void loop() es un lazo que se ejecuta línea por línea y que cuando llega al final repite todo desde el comienzo creando un lazo infinito que para este ejercicio funciona de manera correcta.

2) Los pines utilizados 8, 9, 10 son pines digitales, que fueron declarados como salidas con la función pinMode(Pin #, OUTPUT), debido a esto pudimos conectarles un elemento que emite una señal visual como el diodo LED. El estado ON-OFF de los LEDs lo controlamos directamente con la tarjeta y la función digitalWrite(Pin #, LOW) ó digitalWrite(Pin #, HIGH) funcionó correctamente cada vez que se ejecutó en todas las líneas.

4.2.4 RECOMENDACIONES

1) La función Delay() siempre toma valores en milisegundos por lo que se recomienda considerar valores mayores a 1000ms si queremos visualizar los cambios.

2) Se recomienda que el valor de la resistencia que se utilice no sea muy alto ya que esto baja la intensidad de luz de los LEDs.

3) No debemos olvidar que la tierra es un punto común y debe estar conectada al pin “GND” de nuestro Arduino, en caso de que uno de los LEDs no encienda se recomienda verificar la forma del punto común que tiene nuestro protoboard ya que esta puede ser una de las causas.

4.3 “SELECTOR DE TRES EXTRUSORAS DE DIFERENTE COLOR MEDIANTE ARDUINO Y L293D”.

El desarrollo de este proyecto tiene como finalidad realizar rutinas de control, basadas en una lógica del funcionamiento de 3 sensores ópticos, mediante las cuales podamos crear una nueva alternativa de selección entre 3 extrusoras, las cuales manejan 3 bobinas de PLA de diferente color; lo que nos permitiría escoger un color de nuestra preferencia para imprimir un objeto deseado en la

impresora 3D. Adicionalmente a la utilización de la tarjeta Arduino ATmega 2560, hemos considerado necesario el uso de algunos elementos electrónicos adicionales los cuales nos ayudaron a que todo este proyecto se lleve a cabo.

Para el funcionamiento del modelo base que es la impresora Reprap Prusa Mendel se deben realizar verificaciones y calibraciones previas que son necesarias antes de comenzar a imprimir, a continuación se detallan las principales:

- Correcto funcionamiento del termistor de la extrusora.

- Correcto drenaje de material a través de la boquilla de la extrusora, debe ser fluido y con una caída casi vertical.

- Correcta ubicación del final de carrera de todos los ejes principalmente del eje Z para mantener la mínima distancia entre la boquilla de la extrusora y la cama de impresión.

- Una correcta y precisa nivelación de la “cama” o base sobre la cual se van a asentar todos los modelos 3D.

El Pronterface es el IDE que vamos a utilizar y que junto con Slic3r, que es el programa que convierte los archivos .STL en GCODE, vamos a poder crear un archivo digital en un cuerpo real de 3 dimensiones.

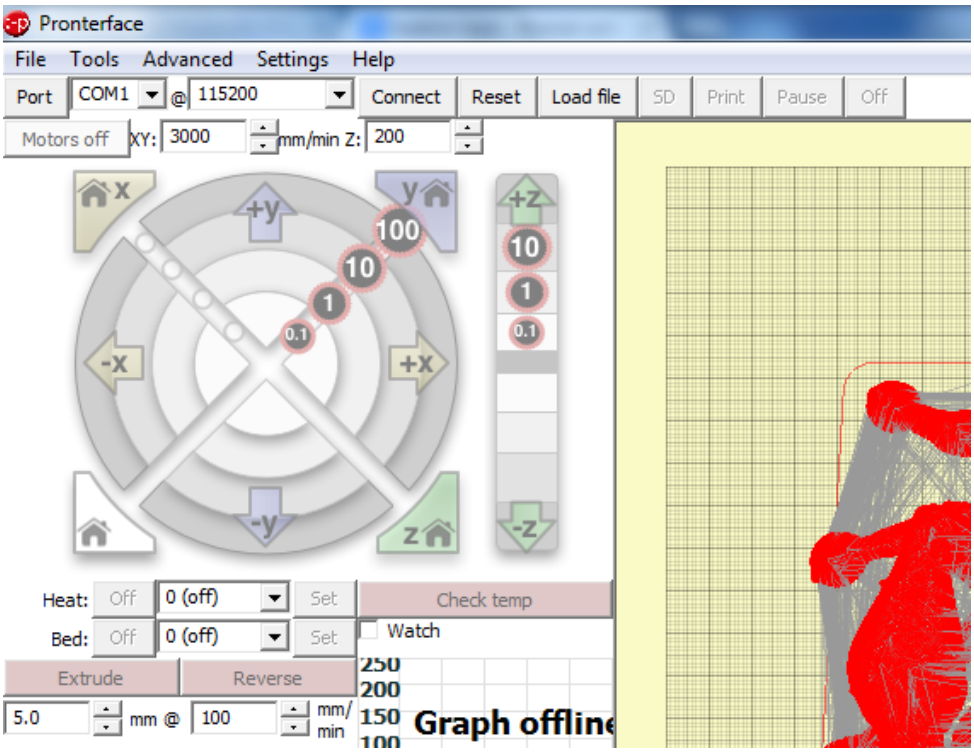


Figura 4.3 Entorno de Pronterface

En Pronterface podemos configurar muchos parámetros para una correcta impresión, la cantidad de parámetros que podamos modificar antes de realizar

una impresión va a depender del tipo de programa cortador que tengamos instalado, como ya lo mencionamos en los primeros capítulos, Slic3r, es el programa con el que vamos a trabajar.

Slic3r nos permite realizar modificaciones en la velocidad de impresión, es decir cuánto material va a extrudir por segundo, que tan grueso queremos que sea el filamento, que tan caliente queremos que este la extrusora antes de empezar la impresión, entre otros. Todos estos parámetros van a ser claves para que nuestro objeto 3D tenga finos acabados y mayor detalle en sus medidas.

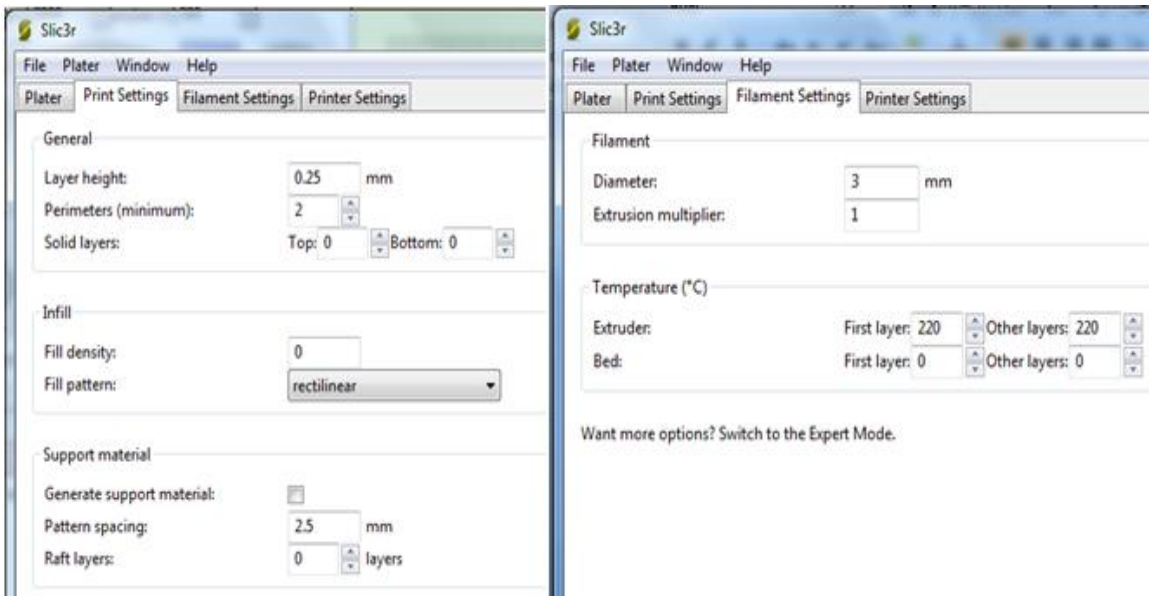


Figura 4.4 Configuración de parámetros antes de imprimir

Debemos tener en cuenta que dependiendo del tamaño y la forma que tenga nuestro objeto, será más óptimo o menos óptimo el hecho de que realicemos cambios en la configuración inicial que trae Slic3r por defecto.

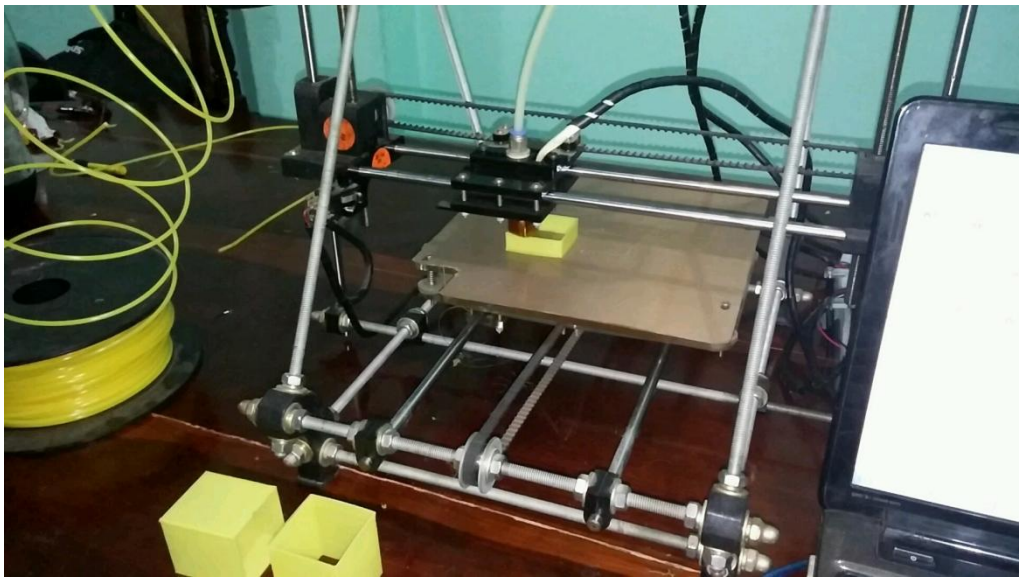


Figura 4.5 Impresión de un cubo de 4 caras laterales.

Como primeros ejercicios a imprimir es recomendable que se realicen objetos simples y de pequeño tamaño, objetos cuadrados como el cubo nos pueden servir como referencia para darnos cuenta de que nuestra impresora está alineada y que los finales de carrera están bien ubicados.

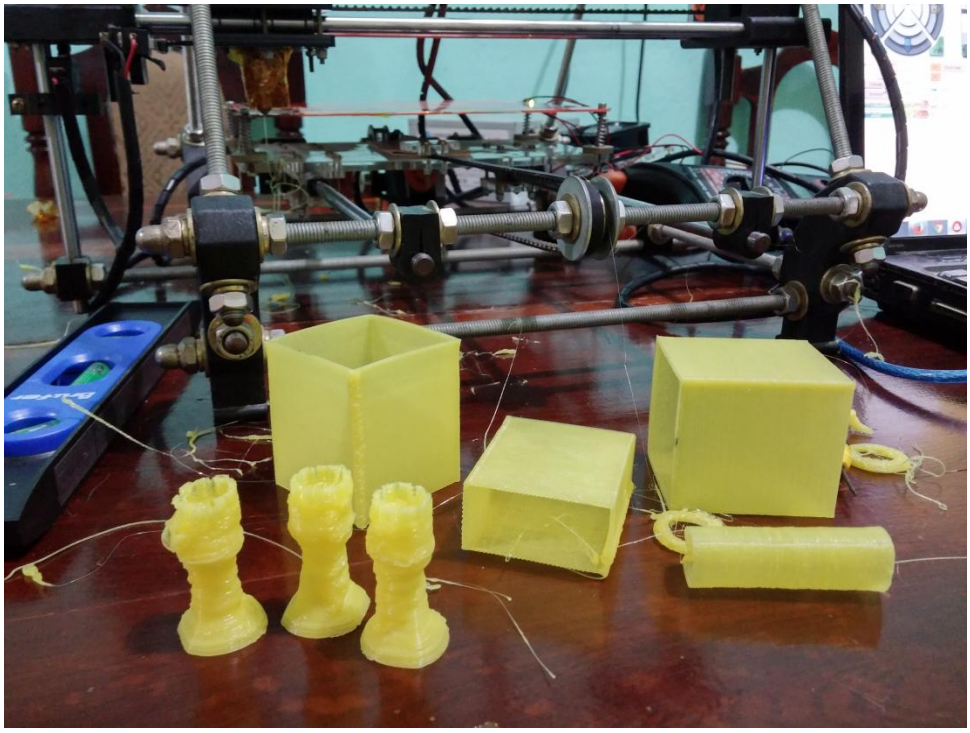


Figura 4.6 Primeros diseños 3D impresos

Parámetros como la temperatura de la extrusora y la altura de la capa o grosor del filamento se tuvieron que modificar para conseguir mayor firmeza vertical y mejores acabados en la construcción del cubo y la torre de ajedrez.

Un ejemplo claro de que al modificar algunos parámetros dependiendo del tipo de complejidad del objeto 3D que queremos construir, es el escorpión que presentamos en la siguiente imagen. Temperatura de la extrusora y velocidad de

extrusión del material fueron configuraciones claves para que el modelo vaya mejorando conforme se modificaban estos parámetros.

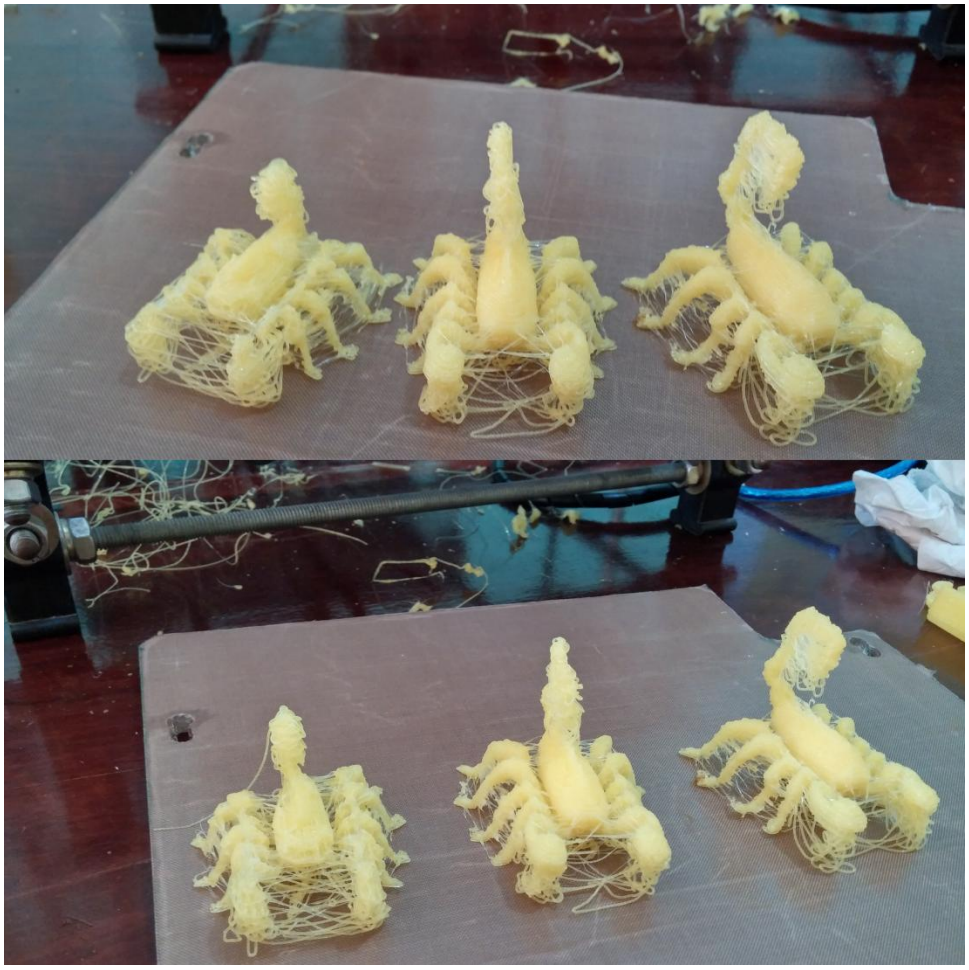


Figura 4.7 Escorpión con diferentes configuraciones de impresión.

Lo que queremos simular en este proyecto final es el mismo funcionamiento de impresora Prusa Mendel pero con una extrusora diferente la cual tendrá no sólo

una boquilla sino 3. La idea principal consiste en preseleccionar el color de impresión, es decir en seleccionar una de las 3 extrusoras y decimos preseleccionar, ya que esta es una selección previa a la orden de “Mandar a Imprimir” ya que una vez comenzada la impresión sólo se construirá el objeto con la extrusora seleccionada es decir con un solo color.

Mediante un switch de 3 posiciones y dependiendo de la posición escogida, se deberán cumplir condiciones de giro y de los sensores para el programa ejecute sus movimientos de rotación y seleccione el color deseado por el usuario.

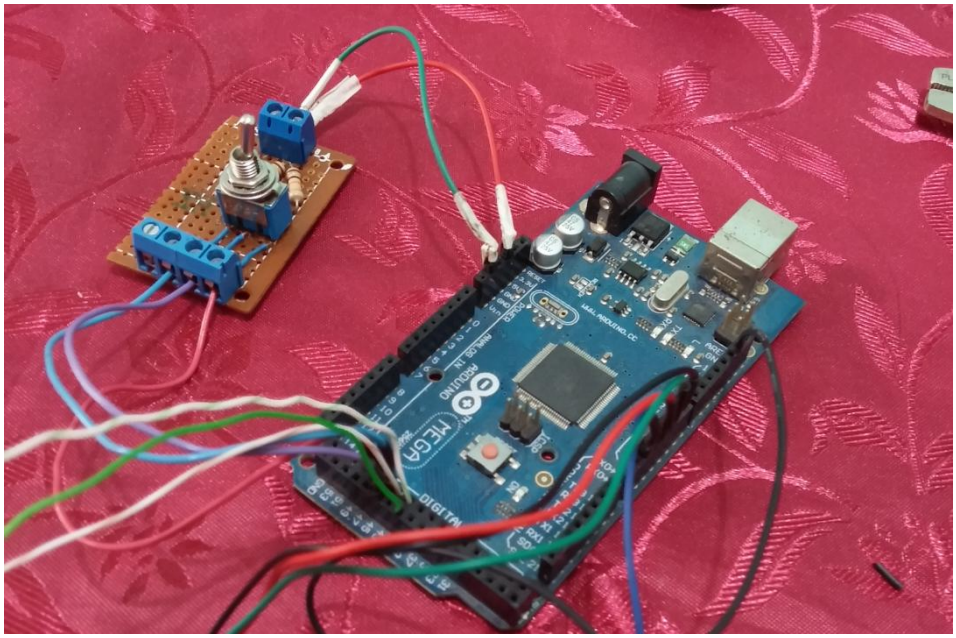


Figura 4.8 Switch selector de 3 posiciones y Arduino ATmega 2560

Una de las ideas claves para la elaboración de este selector de extrusoras fue la creación de un arreglo de 3 sensores. Estos sensores están ubicados de forma circular con una separación de 90° y en una cuarta posición tenemos un espacio en blanco que también es considerado. Se maneja una lógica de ON/OFF para cada uno de los sensores pero que requieren los estados de los demás para validar la posición final que está condicionada por la posición del Switch.

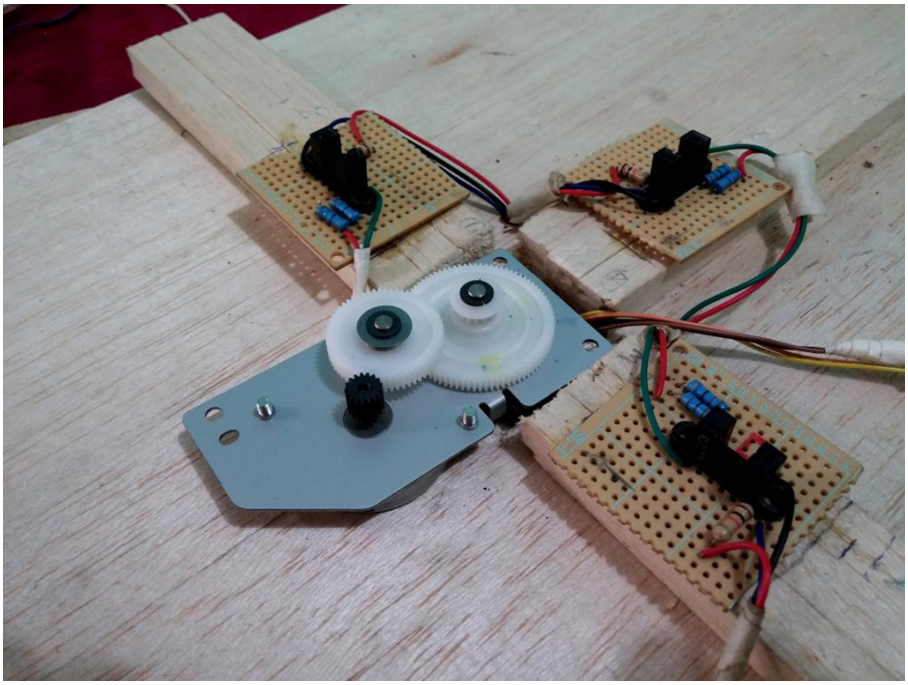


Figura 4.9 Sensores ópticos y motor de paso del eje principal.

Para el manejo de la potencia que necesitaba nuestro motor de paso bipolar fue necesaria la elaboración de una tarjeta controladora formada por 2 integrados L293D los cuales se reparten la potencia en partes iguales, evitando así el recalentamiento y posterior daño de estos integrados.

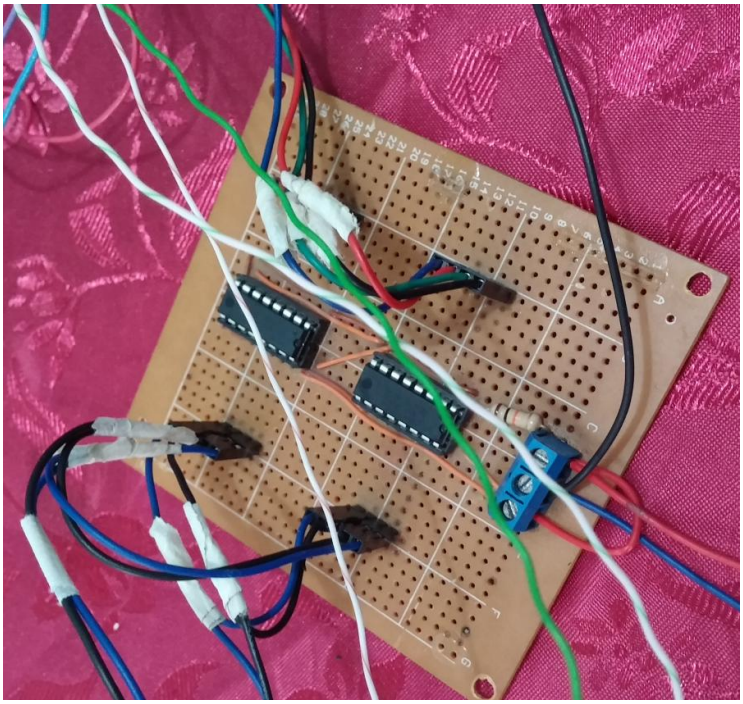


Figura 4.10 Tarjeta controladora del motor formada por 2 integrados L293D

4.3.1 IMPLEMENTACIÓN

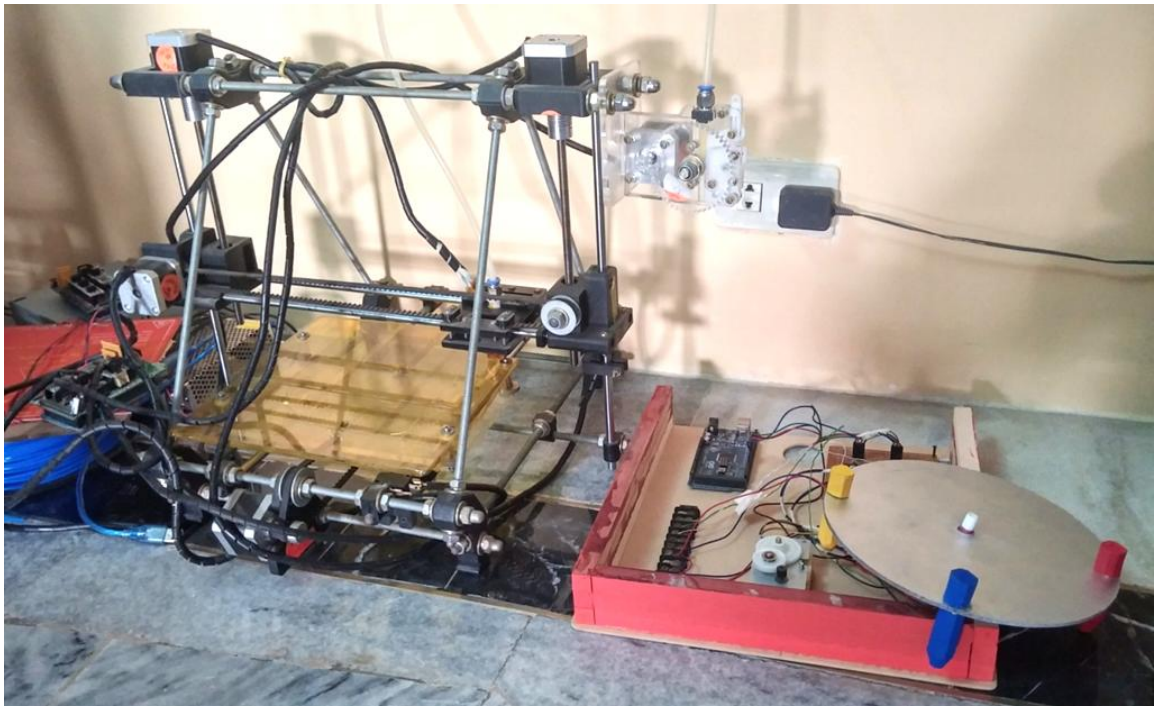


Figura 4.11 Maqueta del proyecto final junto a la impresora Prusa Mendel.

4.3.2 LISTA DE COMPONENTES

- Tarjeta Arduino ATmega2560
- Cable Adaptador USB.

- 2 integrados L293D.
- 8 Resistencias de 1KΩ.
- 1 switch de tres estados.
- Cable UTP.
- Motor de paso bipolar de 12voltios
- Fuente de 12voltios.
- 3 Sensores ópticos.

4.3.3 DIAGRAMA DE CONEXIONES

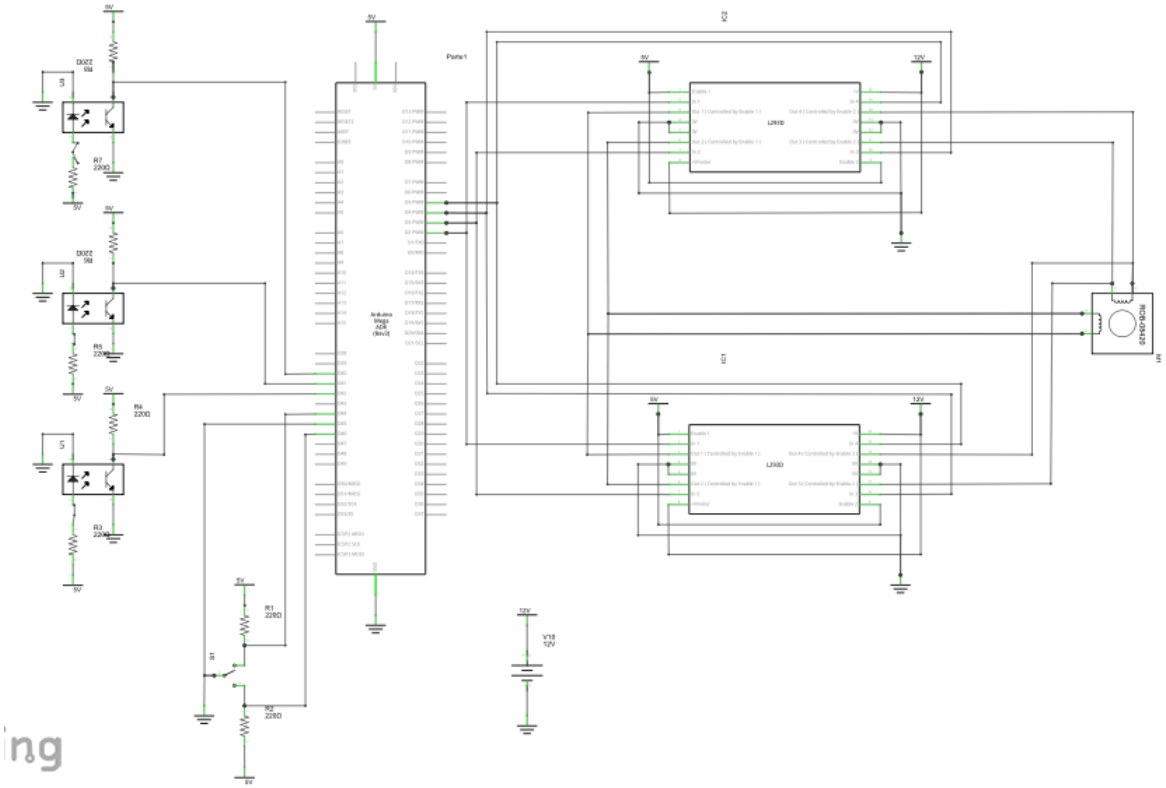


Figura 4.12 Diagrama de conexiones del selector de colores utilizando tarjeta ATmega2560.

CONCLUSIONES

1) Al ejecutar el programa en el arduino, se observa que el disco donde están las extrusoras se comienza a mover, esto se debe a las condiciones de estado que maneja el switch. El usuario debe esperar hasta que la maquina termine su ejecución, en ese momento el operador podrá seleccionar que color quiere para su impresión.

2) La frecuencia que se maneja a nivel de software no nos permitía tener un giro lento del eje de motor para así obtener una mayor precisión en el posicionamiento de cada una de las extrusoras, es por esto que decidimos trabajar con un motor cuyo eje tenga adaptado un sistema de engranes con el cual pudimos reducir la velocidad angular del nuevo eje principal.

3) Cuando se trabaja con diseños complejos como el escorpión, es necesario aumentar la temperatura de la extrusora a 220°C y reducir la velocidad de extrusión del filamento ya que con esto logramos más precisión y mejor detalle en los acabados de nuestro modelo 3D a pesar de que aumenta considerablemente el tiempo de impresión.

RECOMENDACIONES

1) Es recomendable tener siempre conectado el ventilador para el Arduino y la RAMPS ya que los Pololu manejan corrientes considerables lo que hace que se recalienten y lleguen a valores muy elevados de temperatura.

2) Se recomienda siempre colocar cinta Kapton en la superficie sobre la cual va a descansar nuestro objeto mientras es construido, debido a que por la composición química del PLA obtenemos mejor adherencia y firmeza en la base, evitando que nuestro objeto 3D se despegue antes de finalizar la impresión.

3) Se recomienda revisar constantemente la distancia entre la boquilla de extrusora y la cama sobre la que se asienta nuestro objeto, esta debe ser apenas 0,5mm para que existe una correcta adherencia del material y la cama. Esta distancia se la puede corregir deslizando ligeramente el interruptor de final de carrera del eje Z hacia arriba o hacia abajo según sea el caso.

ANEXOS

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.

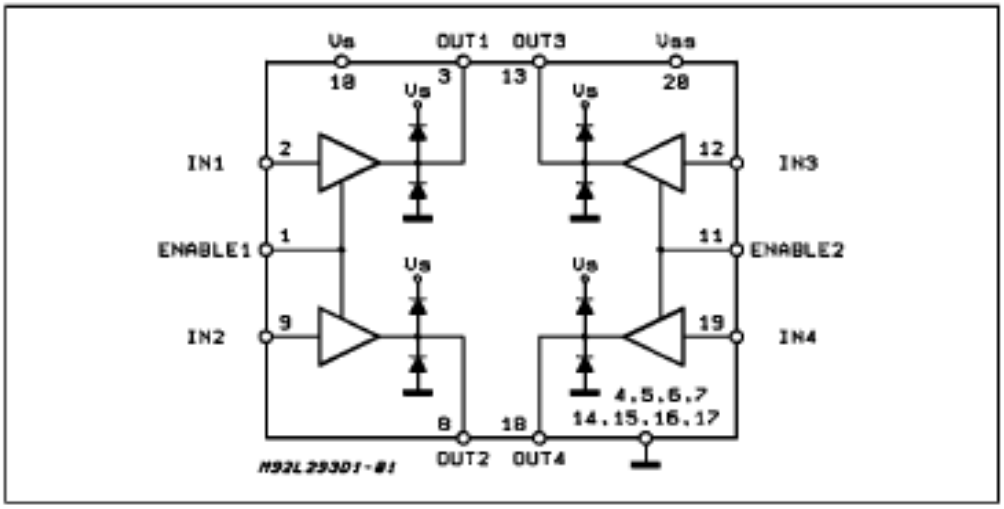
To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included.

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.

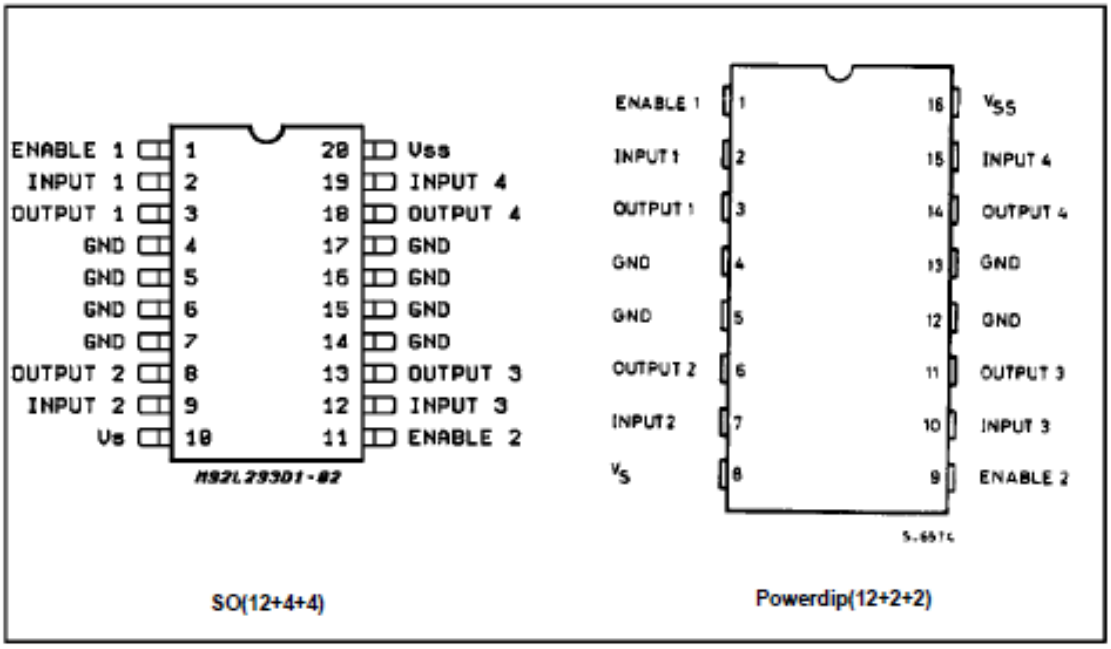
BLOCK DIAGRAM



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 µs 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 6sq. 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_{IL}	Low Voltage Input Current (pin 2, 9, 12, 19)	V_{IL} = 1.5 V			– 10	µA
I_{IH}	High Voltage Input Current (pin 2, 9, 12, 19)	$2.3\ V \leq V_{IH} \leq V_{SS} - 0.6\ V$		30	100	µA
$V_{en\,L}$	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	µA
$I_{en\,H}$	High Voltage Enable Current (pin 1, 11)	$2.3\ V \leq V_{en\,H} \leq V_{SS} - 0.6\ V$			± 10	µA
$V_{CE(sat)H}$	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.

BIBLIOGRAFÍA

[1] Vincent; Earls, Alan R., Origins: A 3D Vision Spawns Stratasys, Inc., http://en.wikipedia.org/wiki/3D_printing, fecha de consulta junio de 2013.

[2] Wikidot, ReplicatorG, http://replicat.org/, fecha de consulta junio de 2013.

[3] Smid, Peter, CNC Programming Handbook (3rd ed.), http://en.wikipedia.org/wiki/G-code, fecha de consulta junio de 2013.

[4] Rafael Infante Martín, Modelado por deposición de hilo fundido, http://tfmrimuned.wordpress.com/modelado-por-deposicion-de-hilo-fundido-fdm/, fecha de consulta julio de 2013.

[5] Baeurle SA, Hotta A, Gusev AA , On the glassy state of multiphase and pure polymer materials, http://en.wikipedia.org/wiki/Thermoplastic, fecha de consulta junio 2013.

[6] Guido Van Rossum, An Introduction to Python for UNIX/C Programmers, http://en.wikipedia.org/wiki/Python_(programming_language), fecha de consulta julio de 2013.

[7] Arduino SA, ¿Qué es Arduino?, http://arduino.cc/es/Guide/Introduction, fecha de consulta junio de 2013.

[8] Solidoodle, Software, http://www.solidoodle.com/how-to-2/how-to-install-software/, fecha de consulta junio de 2013.

[9] Wikidot, Instalación en Windows, http://replicat.org/installation-windows, fecha de consulta julio de 2013.

[10] Escuela de Ingenieros Industriales de Espala, PLA, http://www.eis.uva.es/~biopolimeros/alberto/pla.htm, fecha de consulta junio de 2013.

[11] Arduino SA, Arduino Mega 2560, http://arduino.cc/en/Main/arduinoBoardMega2560, fecha de consulta junio de 2013.

[12] 3D PrinterHub, 3D Printers Models, http://3dprinterhub.com/3d-printer-brands#personal, fecha de consulta mayo de 2013.

[13] 3D PrinterHub, Objet 3D Printer, http://3dprinterhub.com/objet-3d-printer, fecha de consulta junio de 2013.

[14] Chris Liechti, Pyserial, http://pyserial.sourceforge.net/pyserial.html, fecha de consulta julio 2013.

[15] Ultimaker, ReplicatorG Manual, http://wiki.ultimaker.com/ReplicatorG_User_Manual, fecha de consulta agosto 2013.

[16] Makerbot, ReplicatorG, http://www.makerbot.com/support/replicatorg/documentation/dualstrusion/, fecha de consulta agosto de 2013.

[17] Cubify, Cube3, http://cubify.com/cube/store.aspx, fecha de consulta agosto de 2013.

[18] PP3DP, UP! MODEL, http://www.pp3dp.com/index.php?page=shop.product_details&flypage=flypage.tpl&product_id=5&category_id=1&option=com_virtuemart&Itemid=37, fecha de consulta agosto del 2013.

[19] Formlabs, Form 1, http://www.engadget.com/2012/09/26/form-1-delivers-high-end-3d-printing-for-an-affordable-price/, fecha de consulta agosto de 2013.

[20] Laserlines, Stratasys Catalog, http://www.laserlines.co.uk/acatalog/info_36.html, fecha de consulta agosto de 2013.

[21] Simon Monk, Programming Arduino Getting Started with sketches, McGraw-Hill, 2012

[22] Unicode Inc., What is Unicode?, http://www.unicode.org/standard/WhatIsUnicode.html, fecha de consulta julio 2013.

[23] Michael McRoberts, Beginning Arduino, Springer Science Business Media, 2010.

[24] Brian Evans, Practical 3D Printers the Science and Art of 3D Printing, Springer Science Business Media, 2012.

[25] GitHub Enterprise, Download Printrun, https://github.com/kliment/Printrun, fecha de consulta julio 2014.

[26] Arduino SA, ATmega 2560 Pin Mapping, http://arduino.cc/en/Hacking/PinMapping2560, fecha de consulta julio 2014.

[25] RepRap Org., RAMPS & Firmware, http://reprap.org/wiki/RAMPS_1.4#Firmware_and_Pin_Assignments, fecha de consulta julio 2014.

[26] Harold Timmis, Practical Arduino Engineering, Springer Science Business Media, 2011.