Brazo Robótico
(Autómata y Manual)

Por
Stricker Rubén Darío

El prototipo diseñado es totalmente de mi autoría, y fue creado solo para fines didácticos, aunque es totalmente funcional, pero con limitaciones.
Igual tranquilamente con algunos ajustes y pequeñas modificaciones se pueden hacer muy buenas automatizaciones usando como referencia el mismo.

Antes que empecemos, daré algunas sugerencias por si usted desea armarlo:

Riesgos eléctricos.

Es imprescindible tener conciencia del riesgo que engendra la corriente eléctrica. Ya que, si bien no es la mayor fuente de accidentes, se trata generalmente de accidentes graves, en muchos casos mortales.

Tipos de corriente.

Básicamente existen dos tipos de corrientes:
Corriente Continua (DC) y Corriente Alterna (AC).

La DC puede producir electrólisis dependiendo del tiempo de exposición y de la tensión.

La AC es, en igualdad de condiciones, de 3 a 4 veces menos peligrosa que la DC.

No obstante, en términos generales una DC o AC de intensidad 80 mA o superior es peligrosamente mortal. La susceptibilidad es mayor si la persona afectada está en buen contacto con la tierra o apoyada en superficies húmedas o mojadas. Los ambientes con alta temperatura añaden un riesgo adicional debido a la transpiración, ya que la resistencia nominal se ve reducida por la humedad.

Shock eléctrico
Un shock eléctrico puede producir desde una sensación de cosquilleo, hasta un desagradable estímulo doloroso, resultado de una pérdida total del control muscular y llegar a la muerte.

Por otra parte, aun cuando el shock eléctrico pueda ser leve, la reacción refleja de sobresalto puede hacer que el afectado pierda el control del material que está manipulando y sea causa de otro accidente.
Otra cosa más, si durante el desarrollo del proyecto usted cree que pudiera presentar algún riesgo potencial en la seguridad, pregunte, infórmese, hay muchos medios donde poder ver y sacar las dudas que tenga, Internet, Libros, etc. Aunque en el tutorial que estoy brindando, podrá ver las instrucciones y recomendaciones del proyecto con muchos detalles, pero siempre algo puede faltar.
Si en algún momento se bloquea, confundiendo las conexiones, lo mejor en estos casos, es no insistir más por el momento y relajarse, luego más tranquilo seguro podrá solucionarlo.
Cuando no esté seguro del manejo o conexión de un componente, solicite ayuda a alguien que, si sepa o si no dispone a quien pedirla, busque información en Internet o Libros, recuerde que, aunque trabajemos con voltajes de 5 y 12 volts, igual podría sufrir lesiones, por quemaduras o explosiones de componentes.

Verificar el armado correcto de un equipo o circuito antes de utilizarse.

No lo conecte a la alimentación si no está seguro, siempre verifique.

Nunca se apure a terminar ningún proyecto.

Así que sea prudente.

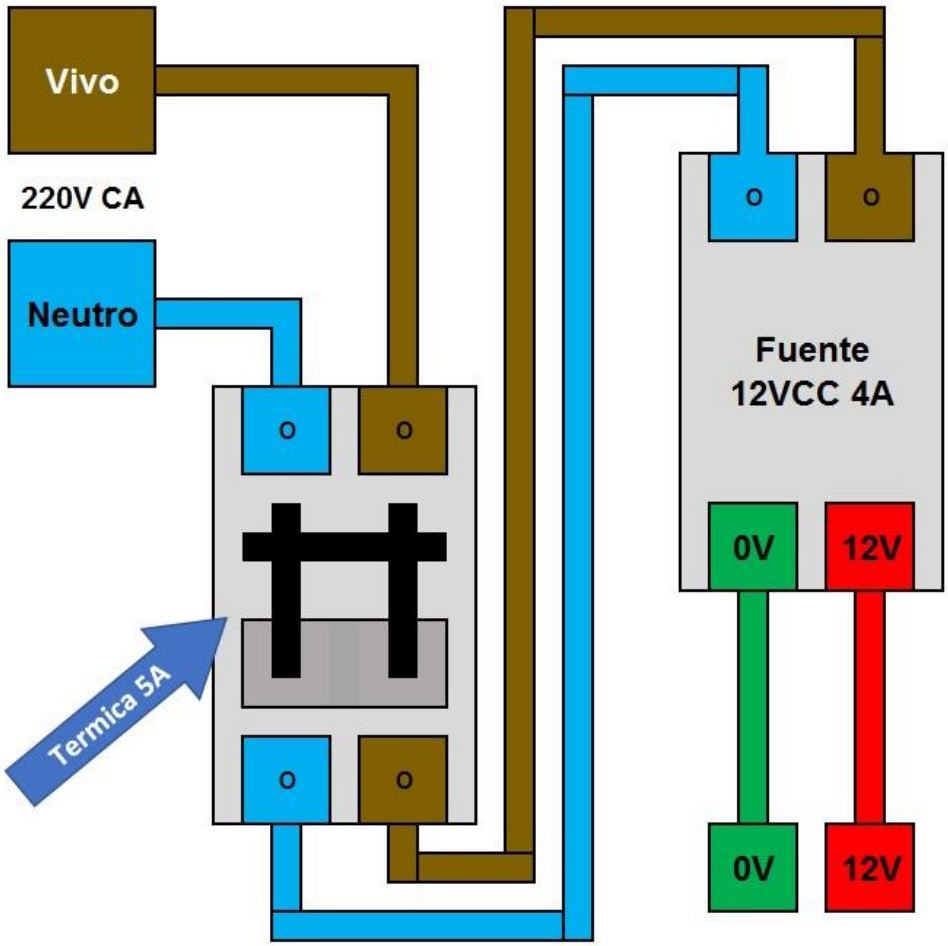
Yo no estudie electricidad ni electrónica, he aprendido con la ayuda de internet y libros, y por experiencia propia, que se puede hacer mucho si uno quiere, pero primero debe familiarizarse con los fundamentos más básicos de electricidad y electrónica.

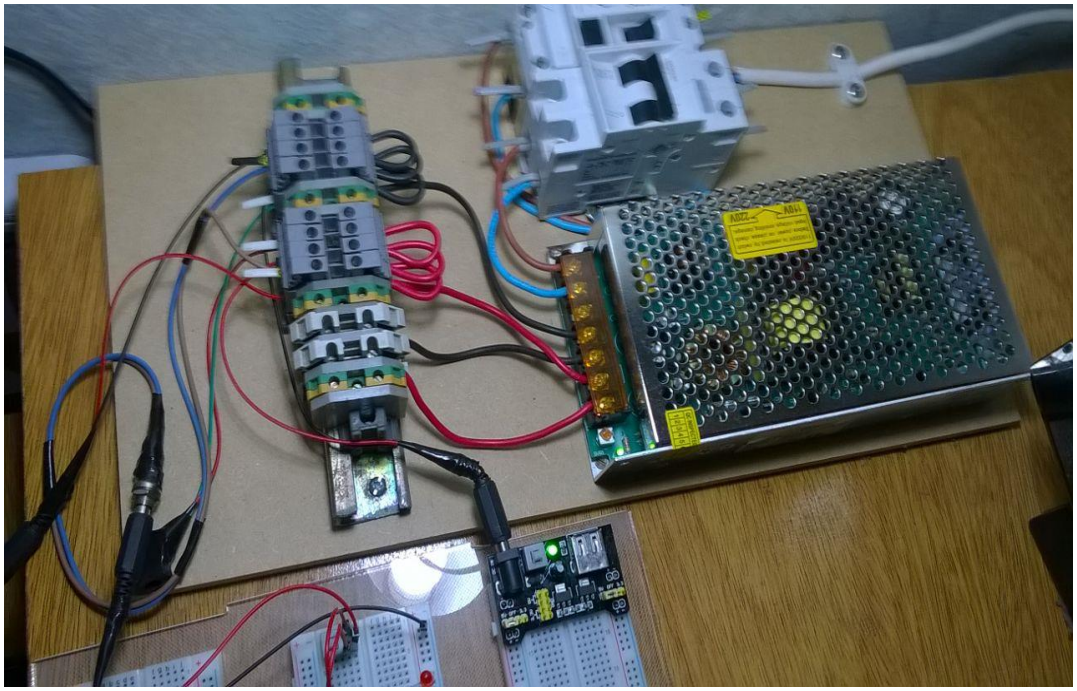
Habiendo dicho esto, comenzamos con el proyecto.

Primero de todo crearemos la sección segura de suministro de energía eléctrica de 220 volts corriente alterna para la fuente de 12 volts corriente continua 4 amperes o más.

Materiales:
1 Base de madera 20 x 40 (fibrofacil) de 5 mm.
1 Llave Térmica de 5 Amperes, o más, no importa, es solo para corte de energía.
1 Fuente de 220 VCA a 12 VCC 5A o más.
1 Riel din para la térmica.
1 Enchufe Macho de 3 patas.
1 Metro o más de cable de 3 polos tipo taller de 2,5 mm.
4 Tornillos para madera de menos de 1 cm.
Los tornillos son 2 para el riel din y 2 para la fuente.

El VIVO Y NEUTRO va el cable de 3 polos tipo taller de 2,5 mm al enchufe macho de 3 patas, que es el que enchufaremos para alimentar el proyecto.

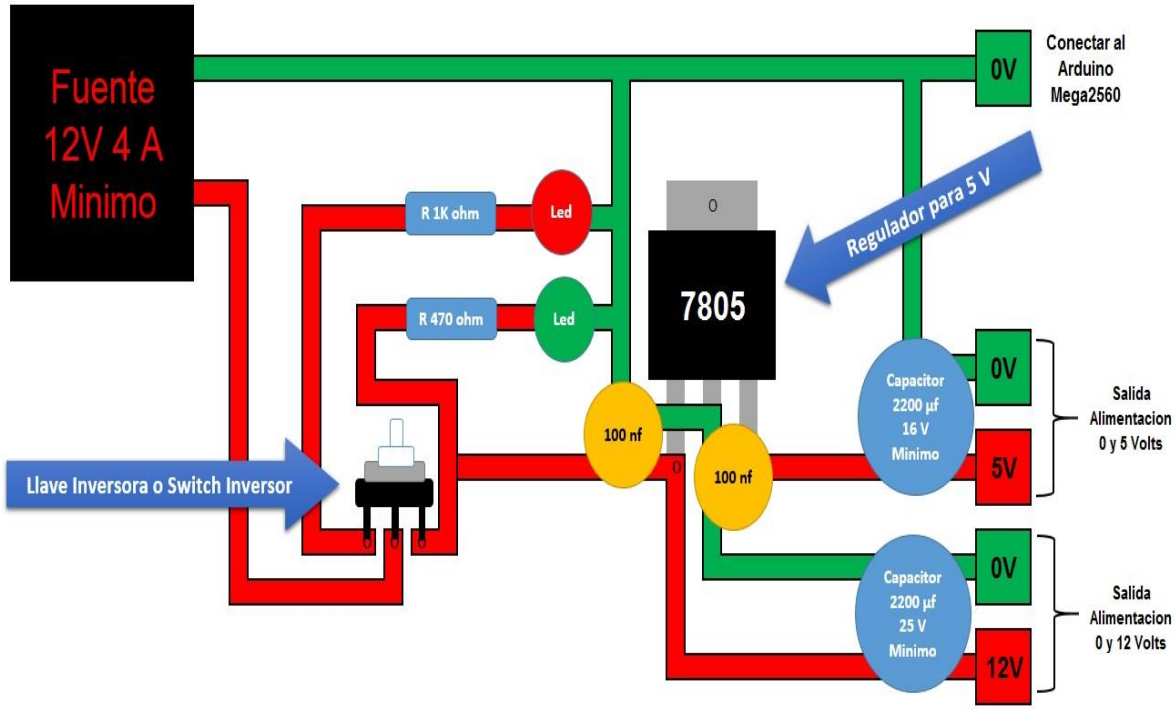

Vivo
220V CA
Neutro
Termica 5A
Fuente
12VCC 4A
0V
12V
0V
12V



Ahora la FUENTE DE ALIMENTACION

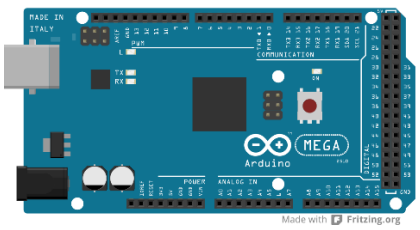
Haremos un pequeño circuito para sacar 5 volts de los 12 volts, asi alimentamos todos los demás componentes, ya que los 12 volts son solo para los 2 motores paso a paso.

Materiales:
1 Llave inversora de perilla o tipo pulsador.
1 Resistencia de 1K Ω.
1 Resistencia de 470 Ω.
1 Led Rojo 5 mm.
1 Led Verde 5 mm.
1 Regulador de voltaje C.I 7805.
2 Capacitores cerámicos de 100 nf 25V o más.
2 Capacitores electrolíticos de 2200 µf 25V o más.



Se aconseja que, en las salidas de voltaje, usar borneras así más fácil a la hora de usar o modificar, aparte de poder usar cables más gruesos para cuando se necesite suministrar más potencias y de esta manera no recaliente los cables.

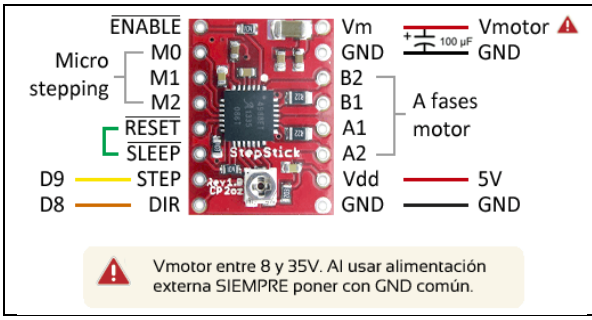
Para este proyecto usaremos un Arduino Mega 2560, ya que usaremos varios pines de entrada y salida, para poder monitorear con leds y un LCD 20 x4, aunque esto es opcional, queda muy profesional de esta manera.

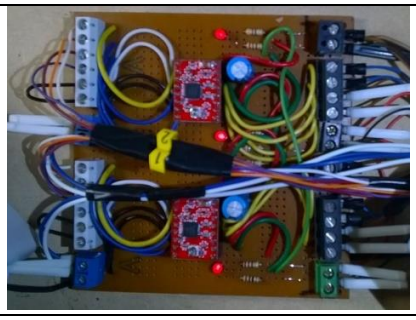


Ahora procederemos a conectar los drivers A4988 controladora de motores paso a paso.

Materiales:
2 drivers A4988.
2 MOTORES NEMA17

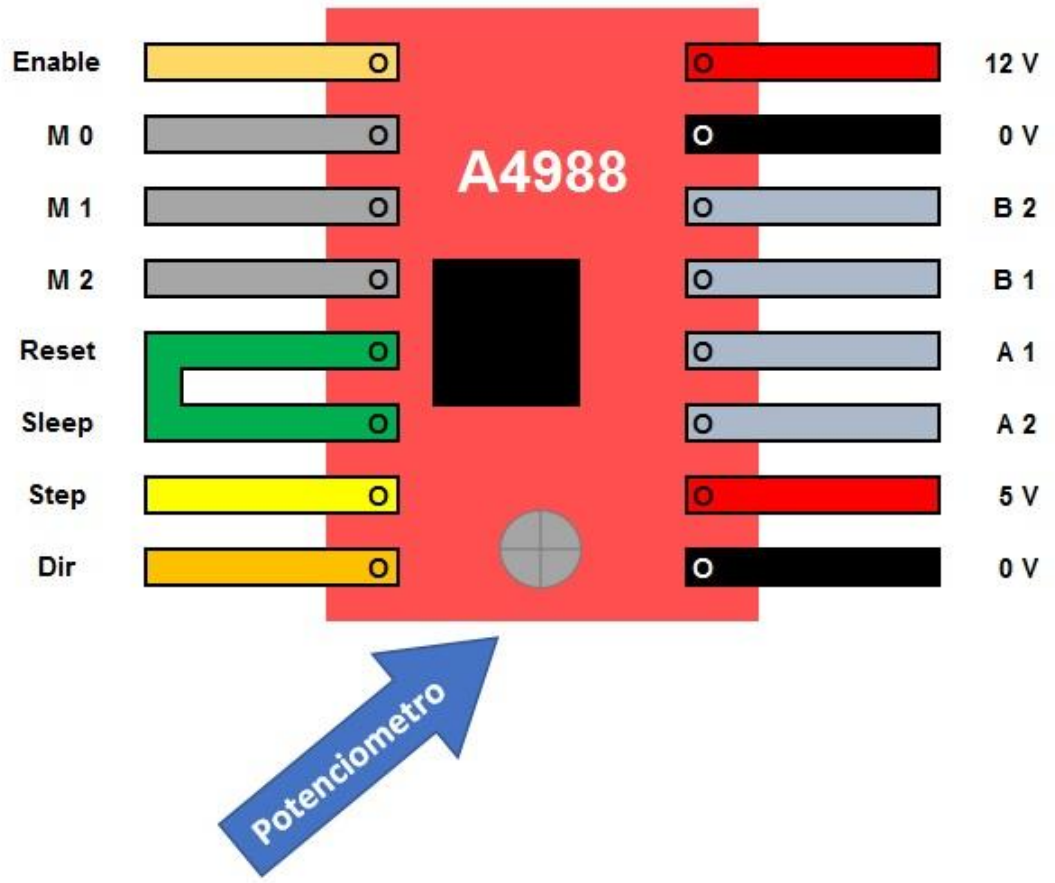
Yo he diseñado una plaqueta para poder conectar todo más fácil con cables más gruesos, pero es un diseño propio y bastante fácil de hace en una placa pcb con perforaciones y unos cuentos pines hembras y borneras, nada más.
Yo le agregue unos leds para poder ver cuando tengo tención y cuando no, pero no es necesario.





Esta es la imagen del A4988, y su descripción.

Yo lo he descripto de otra manera para que la conexión de cableado sea más sencilla.

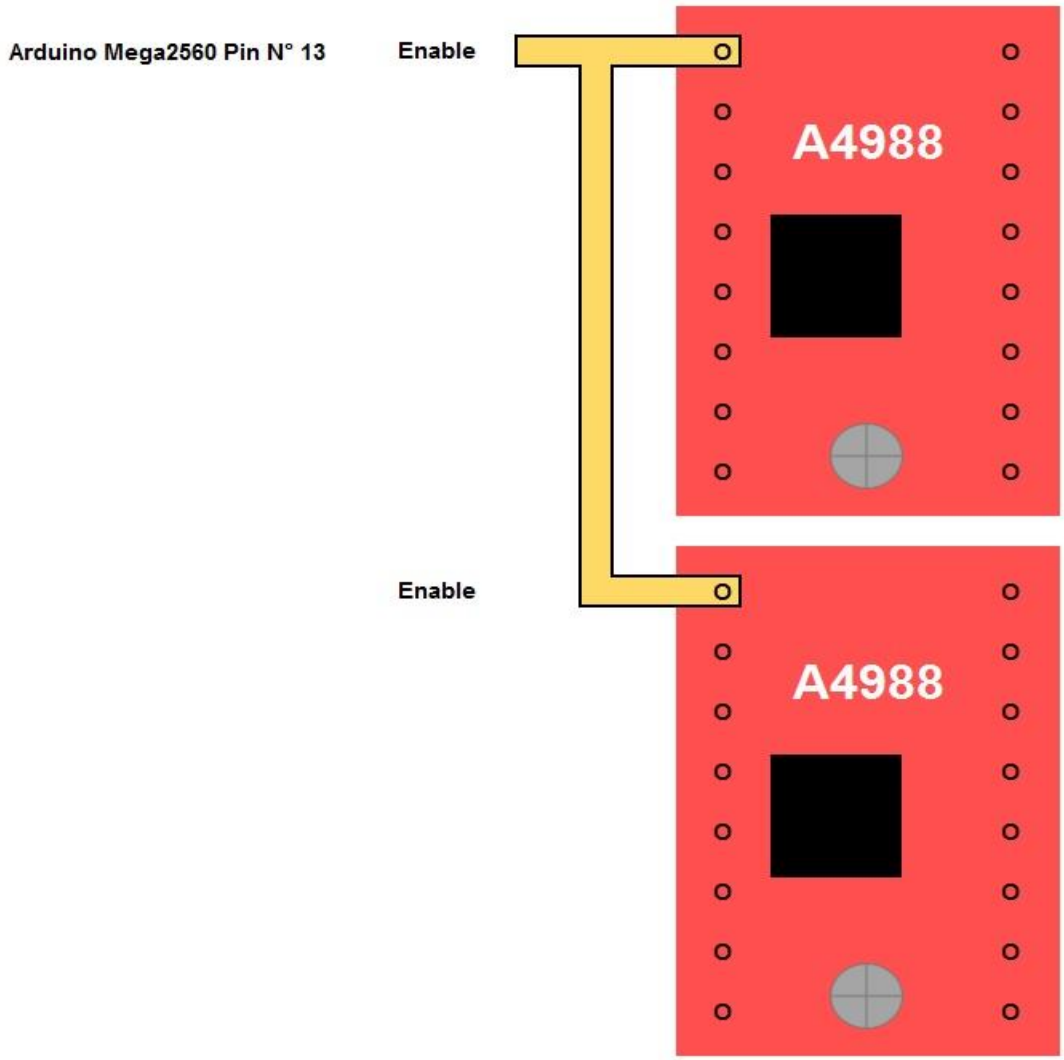


Como verán, es una imagen muy precaria, pero tiene todo lo que necesitamos ver, y con la ventaja de tener los cables identificados con colores específicos y mostrando el potenciómetro como referencia de cómo van los pines, ya que una mala conexión implica en la destrucción del driver A4988, ya que son muy delicados.
Luego más adelante deberemos regular ese potenciómetro para poder suministrar el voltaje adecuado a nuestro motor.

Bueno, dejaremos los pines de alimentación para lo último, por razones de seguridad.
Ya que un descuido y podríamos perder los drivers A4988.

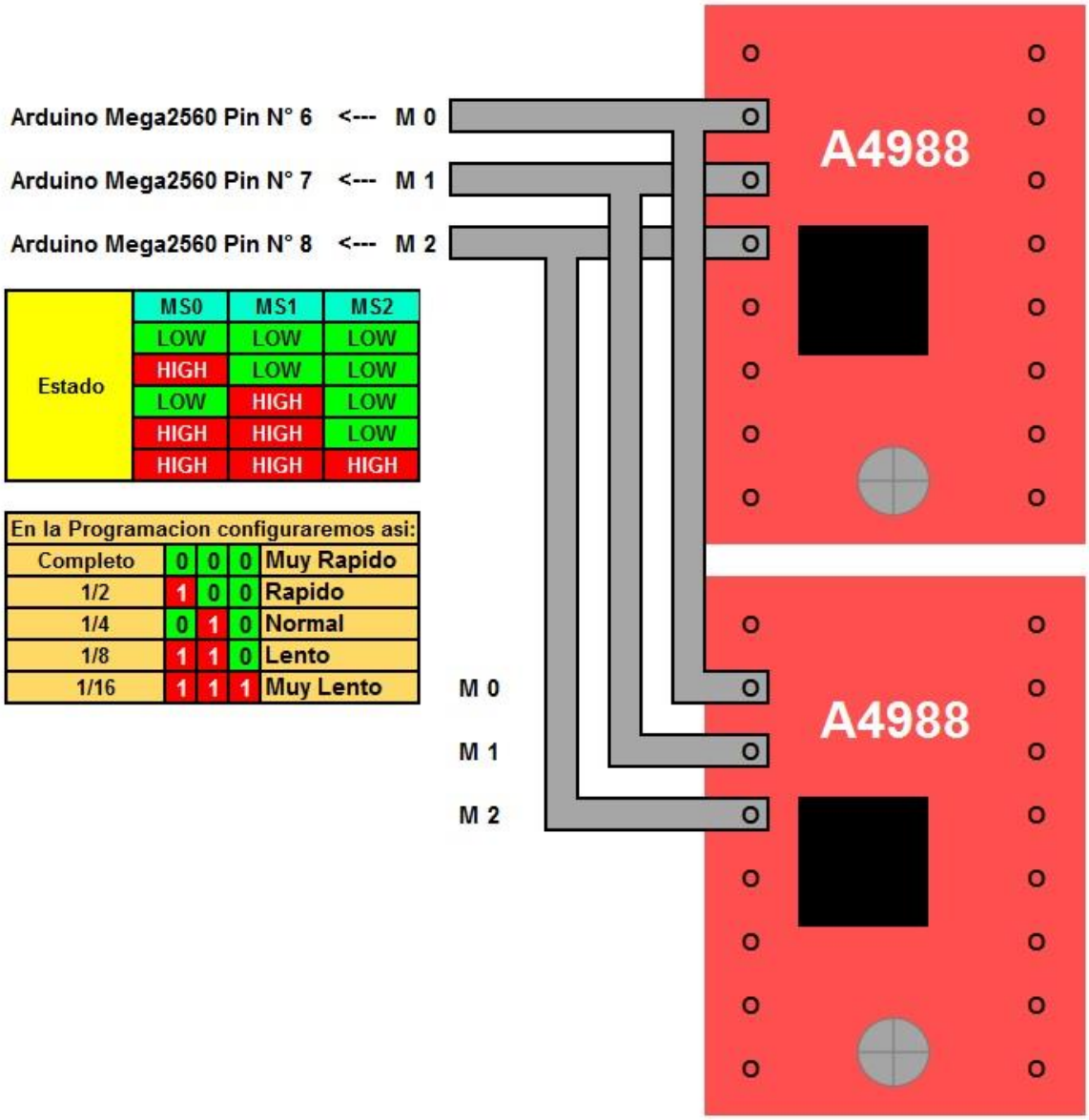
Entonces comenzaremos con los pines de señal.

Enable



El ENABLE Permite que el driver A4988 pueda enviar corriente al motor.
En este proyecto usaremos un solo control para los 2 Enable.
LOW o 0V habilita.
HIGH o 5V deshabilita.
El enable va conectado al pin 13 de Arduino Mega 2560, para que no tengo tensión los motores cuando no sean usados, pero también se pueden dejar a 0V en la prueba y calibración, para mayor facilidad.

M0, M1 y M2



Estado	MS0	MS1	MS2
	LOW	LOW	LOW
	HIGH	LOW	LOW
	LOW	HIGH	LOW
	HIGH	HIGH	LOW
	HIGH	HIGH	HIGH

En la Programacion configuraremos asi:				
Completo	0	0	0	Muy Rapido
1/2	1	0	0	Rapido
1/4	0	1	0	Normal
1/8	1	1	0	Lento
1/16	1	1	1	Muy Lento

Los M0, M1 y M2 son los encargados de la Configuración de los micropasos.
Estos 3 pines (M1, M2 y M3) son para seleccionar una de las resoluciones de cinco pasos de acuerdo con la tabla de verdad anterior.
Estas clavijas tienen resistencias pull-down internas así que, si las dejamos desconectadas, el tablero funcionará en modo de paso completo.
Pero en nuestro proyecto las usaremos para poder seleccionar distintas velocidades, más adelante lo explicaré.
El M0 va conectado al pin 6 de Arduino Mega 2560.
El M1 va conectado al pin 7 de Arduino Mega 2560.
El M2 va conectado al pin 8 de Arduino Mega 2560.

Como verán en el esquema, al igual que enable, los conectamos juntos a cada M de cada **driver A4988,** (M0 driver1 con M0 driver 2) y este al pin 6 de **Arduino Mega 2560**, y asi con los demás.
Según la tabla tenemos las siguientes posibilidades

Estado	MS0	MS1	MS2
	LOW	LOW	LOW
	HIGH	LOW	LOW
	LOW	HIGH	LOW
	HIGH	HIGH	LOW
	HIGH	HIGH	HIGH

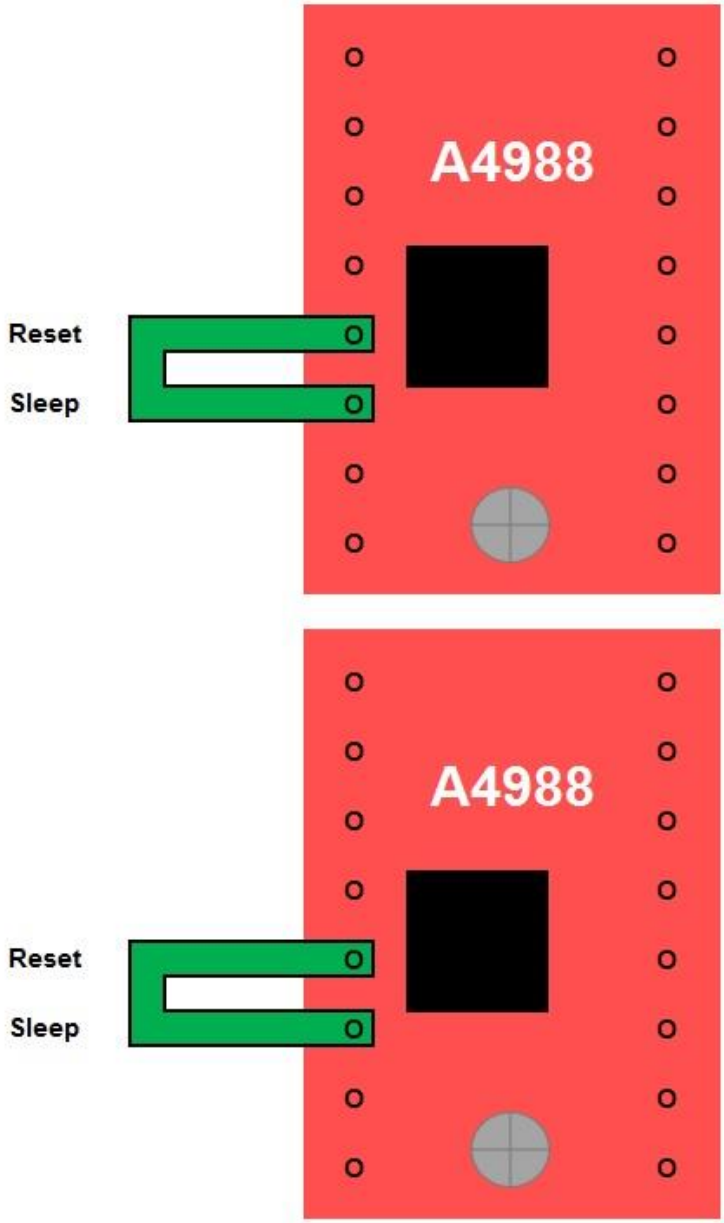
LOW = 0 Volts
HIGH = 5 Volts

Con LOW, LOW, LOW paso completo.
Con HIGH, LOW, LOW 1/2 paso.
Con LOW, HIGH, LOW 1/4 paso.
Con HIGH, HIGH, LOW 1/8 paso.
Con HIGH, HIGH, HIGH 1/16 paso.

En la Programacion configuraremos asi:				
Completo	0	0	0	Muy Rapido
1/2	1	0	0	Rapido
1/4	0	1	0	Normal
1/8	1	1	0	Lento
1/16	1	1	1	Muy Lento

Para la automatización, lo dejaremos LOW, HIGH, LOW, a 1/4 de pasos, así no es tan lento, pero teniendo bastante precisión.
Los pasos más lentos son los de más precisión, y los pasos más rápidos son menos precisos dificultando a veces el exacto posicionamiento.
El paso 1/16 es el más apropiado para usar, cuando las tareas demanden mucha precisión.

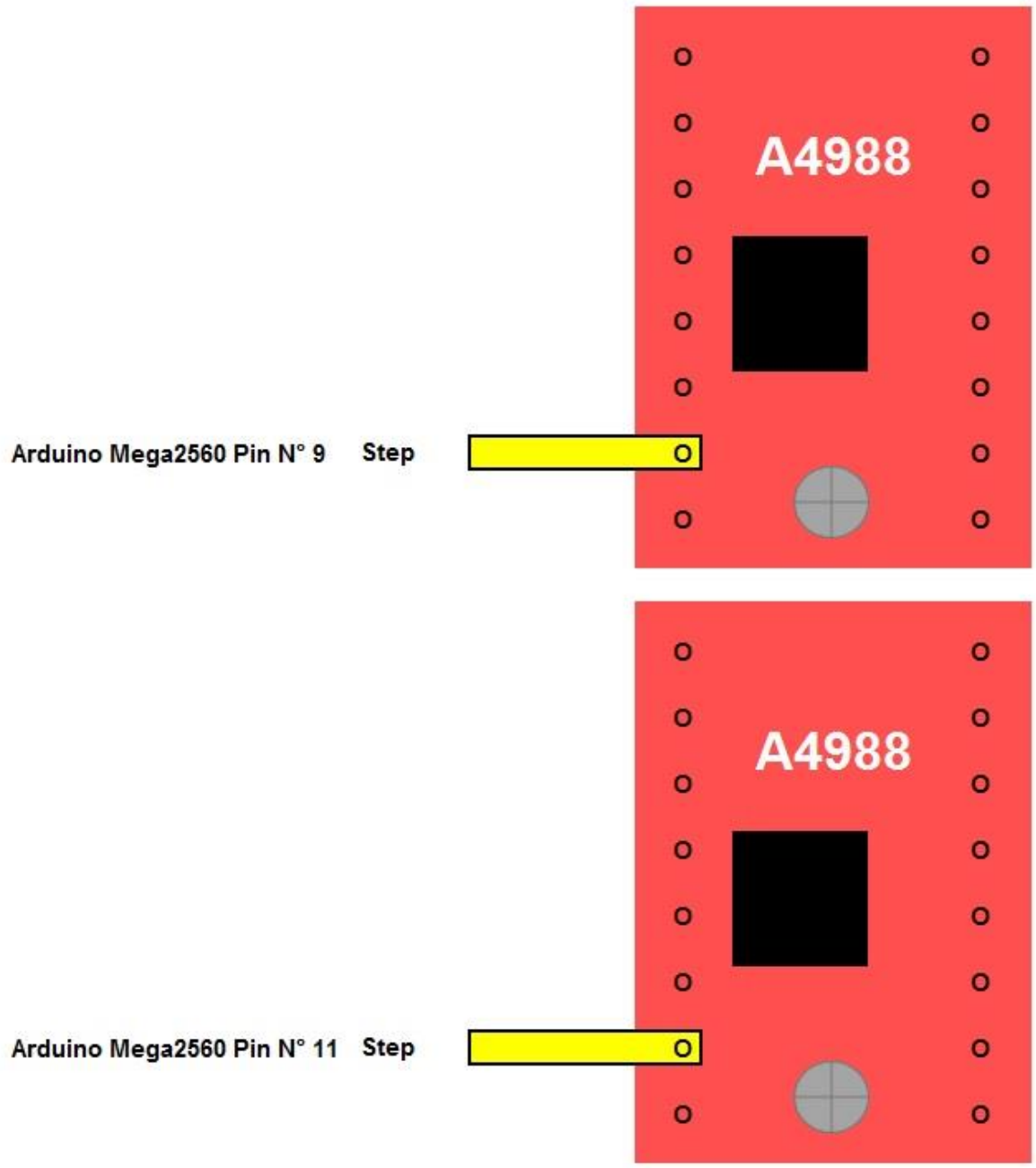
Reset y Sleep



Acá es solo conectar 2 pines entre sí de cada driver A4988.
Sleep = Modo de sueño
El Pin Sleep y un nivel lógico bajo pone el driver A4988 en modo de reposo para minimizar el consumo de energía cuando el motor no está en uso.

El pin RESET establece el traductor en un estado de inicio predefinido.
Este Estado de origen o Posición de Microstep en el Hogar se puede ver a partir de estas Figuras de la Hoja de Datos A4988. Así que estas son las posiciones iniciales desde donde empieza el motor y son diferentes dependiendo de la resolución del microstep. Si el estado de entrada a este pin es un nivel lógico bajo, todas las entradas STEP serán ignoradas. El pin de Restablecimiento es un pin flotante, así que si no tenemos intención de controlarlo con nuestro programa necesitamos conectarlo al pin SLEEP para llevarlo alto y habilitar el tablero.

Step

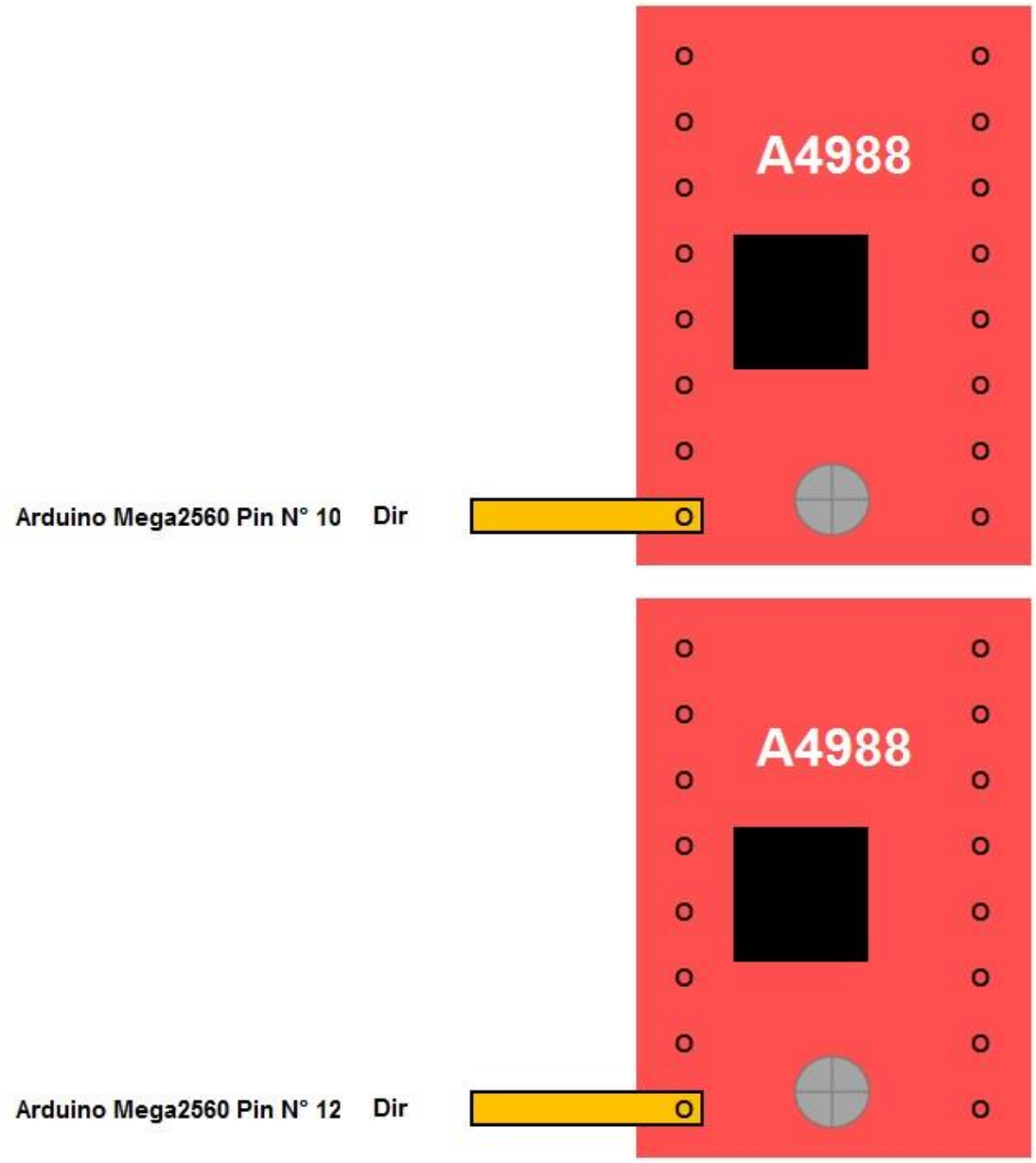


Step = Paso

Con el Step controlamos los microstep del motor y con cada pulso enviado a este pin el motor mueve un paso. Esto significa que no necesitamos ninguna programación compleja, tablas de secuencia de fases, líneas de control de frecuencia, etc., porque el traductor incorporado del controlador driver A4988 se encarga de todo.
Aquí debemos mencionar que este pin no debemos dejarlos flotando en nuestro proyecto, porque filtros parasito son afectado y podría funcionar de manera indeseada.

Cada uno de los Steps deberán ir a un pin diferente, pin 9 del Arduino Mega 2560 y pin 11 del Arduino Mega 2560.

Dir



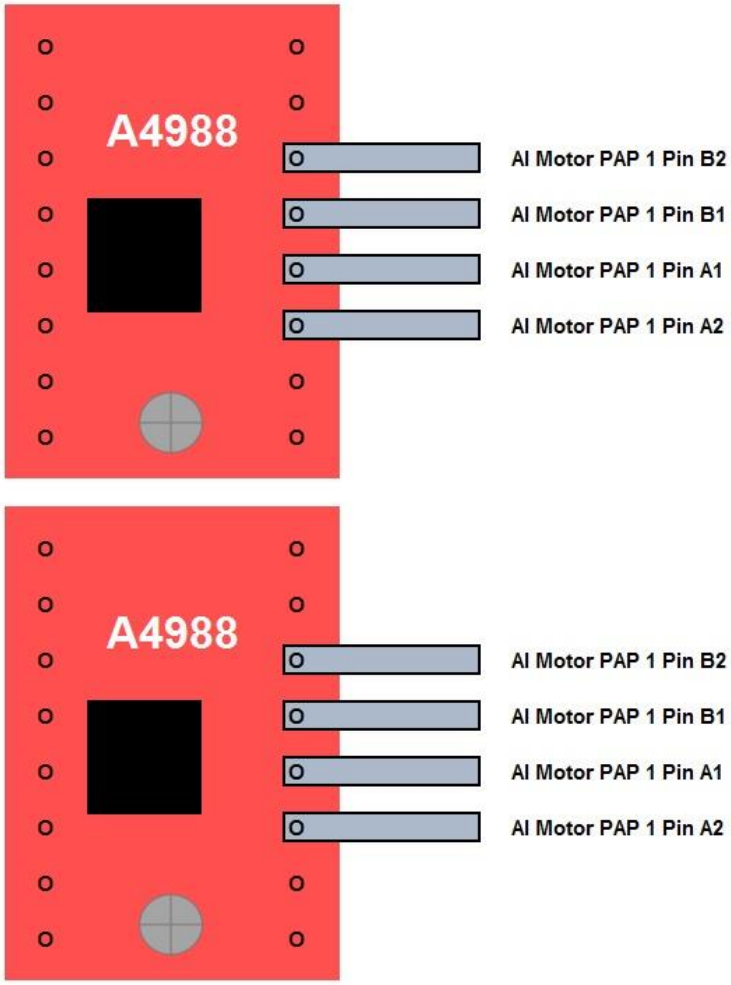
El pin de Dir controla la dirección de rotación del motor y necesitamos conectarlo a uno de los pines digitales de nuestro microcontrolador Arduino Mega 2560.

En este proyecto, conectaremos el primer pin Dir, al pin número 10 del Arduino Mega 2560 y el segundo pin Dir al pin número 12 del Arduino Mega 2560.

Podríamos haberlo puesto con alguna llave o botón pulsador para cambiar el sentido de giro, pero como lo usaremos en automatismo y luego en manual, lo usamos con el controlador, ya que en el momento que lo usemos manual, con un Joystick analógico es mucho más fácil y cómodo de hacerlo.

El pin Dir y Step son los pines que realmente utilizamos para controlar los movimientos del motor.

B2, B1, A1 y A2



Los pines B2, B1, A1 y A2, son los que llevan la corriente al motor paso a paso, y dependiendo la secuencia el motor

Tabla de orden de fases. En este caso concreto el motor tendrá un paso angular de 90º y un semipaso de 45º (al excitarse más de una bobina)

Paso	Terminal 1 Bobina A	Terminal 2 Bobina A	Terminal 1 Bobina B	Terminal 2 Bobina B
Paso 1	5	0		
(Semi-)Paso 2	5	0	5	0
Paso 3			5	0
(Semi-)Paso 4	0	5	5	0
Paso 5	0	5		
(Semi-)Paso 6	0	5	0	5
Paso 7			0	5
(Semi-)Paso 8	5	0	0	5

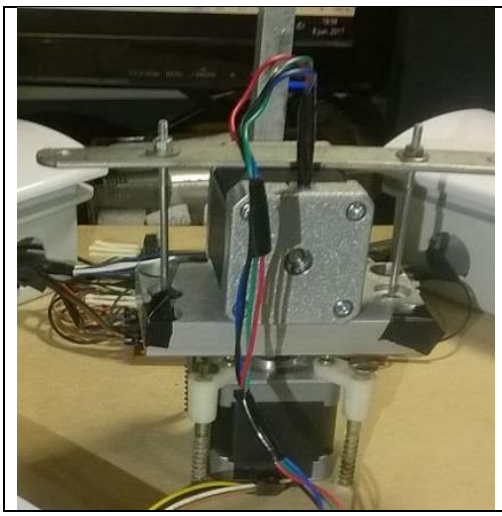
Como en este proyecto usaremos 2 motores paso a paso, las conexiones son en total 8 cables, 4 a cada motor, desde cada driver A4988.

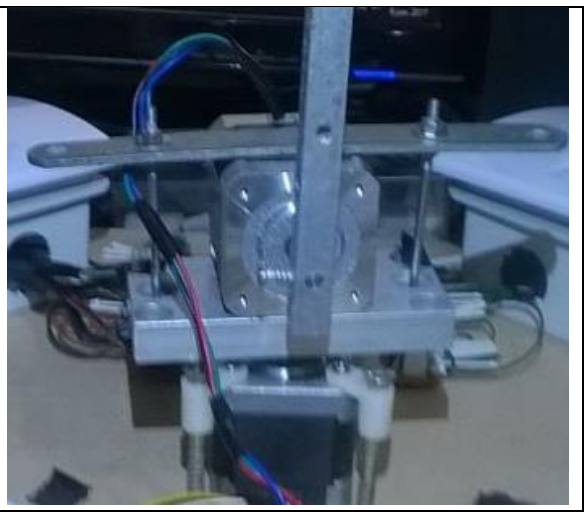
B2x, B1x, A1x, A2x y B2y, B1y, A1y A2y.

Los Motores NEMA17

Características motor paso a paso NEMA17 más común:

Tamaño: 42 mm x 38 mm cuadrados, sin incluir el eje (NEMA 17)
Peso: 285g (10 onzas)
Diámetro del eje: 5 mm.
Pasos por revolución: 200.
Giro 360° / Pasos por revolución: 200 = 1.8°
Corriente: 1.68A por bobina.
Voltaje: 2.7 V.
Resistencia: 1.65 Ohm por bobina.
Torque: 3.7Kg-cm (51 onzas-in)

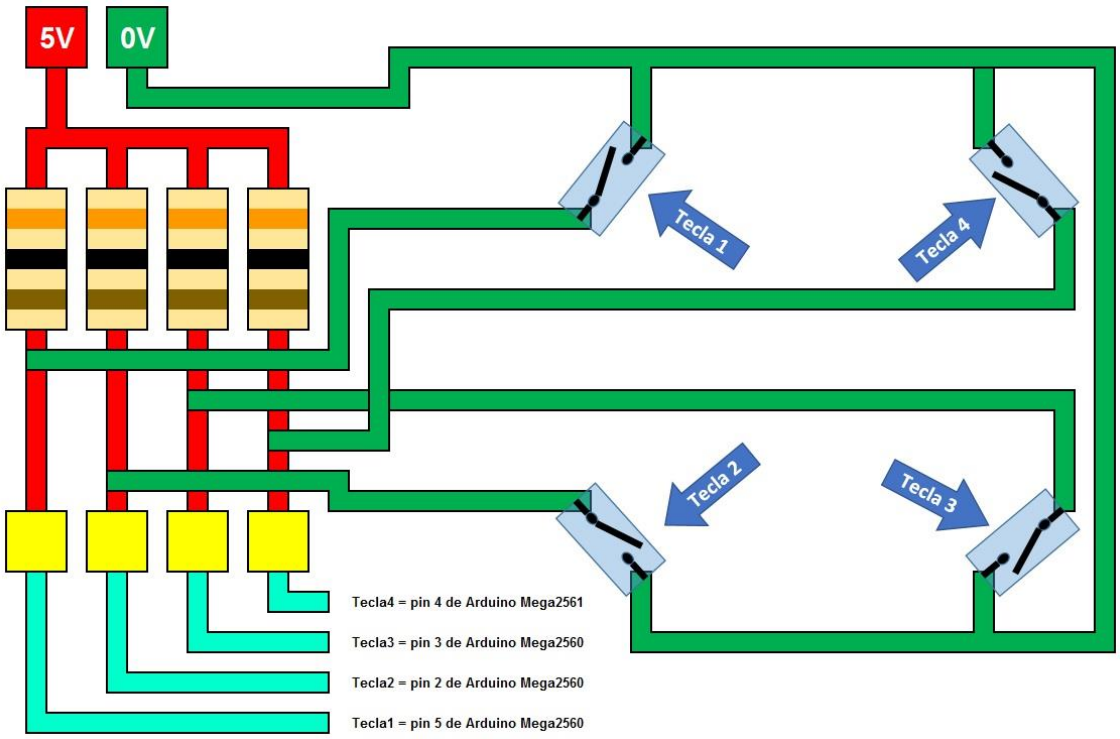




Las Teclas

Estas teclas serán las encargadas de hacer las tareas de forma automática de este proyecto, cuando el caso lo requiera.

Materiales:
4 Resistencias de 10K Ω.
4 Teclas de luz comunes.
4 Caja para las teclas.
1 Base de madera 40 x 40 (fibrofacil) de 5 mm.
8 Tornillos para madera de menos de 1 cm.
Los tornillos son 2 para caja de las teclas.
1 Bornera x 6 polos



Las borneras se distribuyen de esta manera:

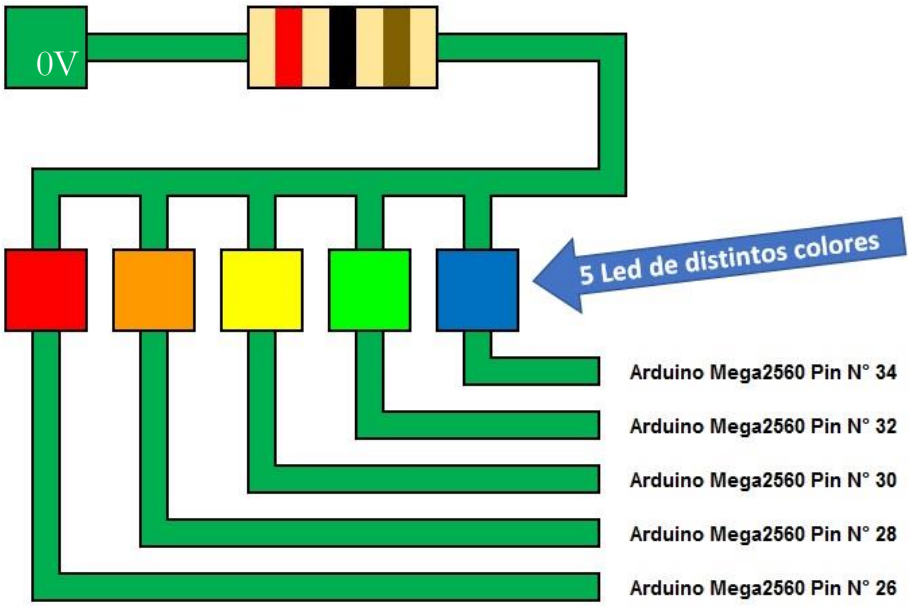
2 para la alimentación, 0V y 5V.
4 para las señales de entrada al Arduino Mega2560.

Las salidas de las teclas son las 4 señales a nuestro Arduino Mega2560, y las conectaremos de esta manera:
La bornera de la tecla 1 al pin 5 del Arduino Mega2560.
La bornera de la tecla 2 al pin 2 del Arduino Mega2560.
La bornera de la tecla 3 al pin 3 del Arduino Mega2560.
La bornera de la tecla 4 al pin 4 del Arduino Mega2560.
De esta manera es más fácil para no confundirse, usamos el pin 5 en vez del pin 1, ya que este último tiene otras configuraciones que podría ocasionar el mal funcionamiento de nuestro proyecto, por eso fue reemplazado.

Led de Aviso de Velocidad

Estos leds son los encargados de mostrar visualmente el estado de velocidad en que estan trabajando los motores paso a paso.

Materiales:
1 Resistencias de 1K Ω.
1 Led Rojo de 5mm.
1 Led Naranja de 5mm.
1 Led Amarillo de 5mm.
1 Led Verde de 5mm.
1 Led Azul de 5mm.



Cada led encendido, muestra la velocidad actual en uso o en espera.
La velocidad por defecto la usaremos en VELOCIDAD NORMAL, que es el led Amarillo.
La conexión es muy sencilla, una conexión a 0V con una resistencia de 1K Ω y desde esta a los cátodos de cada uno de los 5 leds.
Los Ánodos de cada led los conectaremos de la siguiente manera:

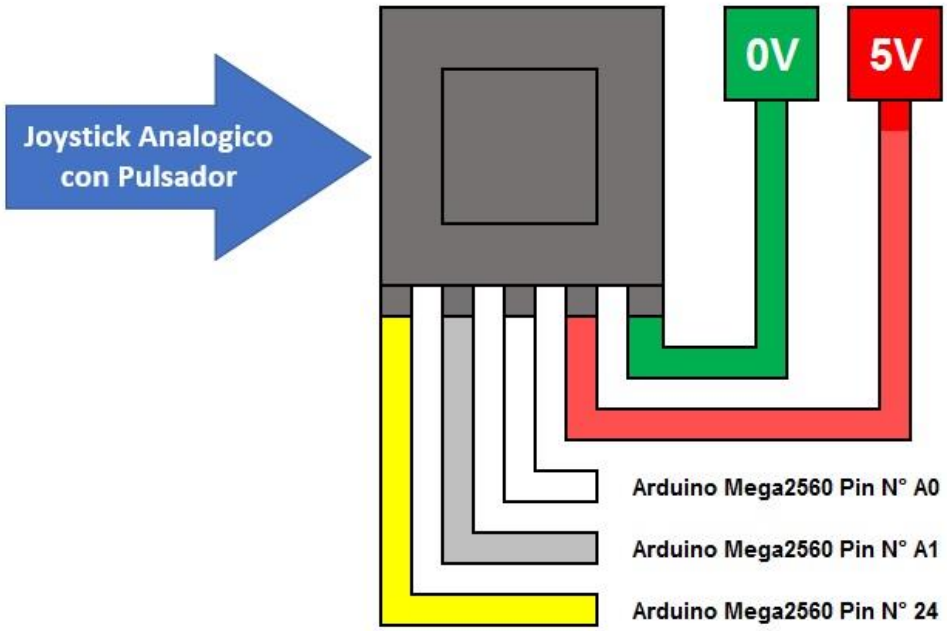
Led Rojo al pin 26 del Arduino Mega2560.
Led Naranja al pin 28 del Arduino Mega2560.
Led Amarillo al pin 30 del Arduino Mega2560.
Led Verde al pin 32 del Arduino Mega2560.
Led Azul al pin 34 del Arduino Mega2560.

El led Rojo avisa VELOCIDAD MUY RAPIDO
El led Naranja avisa VELOCIDAD RAPIDO
El led Amarillo avisa VELOCIDAD NORMAL
El led Verde avisa VELOCIDAD LENTO
El led Azul avisa VELOCIDAD MUY LENTO

JOYSTICK

El joystick es el encargado de mover los motores paso a paso de manera manual.
Esto es muy útil, para poder calibrar o hacer movimientos que no fueron programados.

En este proyecto usaremos un Módulo Joystick Analógico con Pulsador Ps2 es muy sencillo su uso, aparte de ser económico, es muy fácil de conseguir.
Ejemplo: algún Joystick de Play 2 en desuso.



Como verán las conexiones son muy simples.
2 pines a 0V y 5V.
2 pines son los encargados de las coordenadas X/Y
1 pin es el pulsador, se lo conoce como (R3)

Las conexiones son de esta manera.

El pulsador (R3) al pin 24 del Arduino Mega2560.
El pin X al pin A0 del Arduino Mega2560.
El pin Y al pin A1 del Arduino Mega2560.

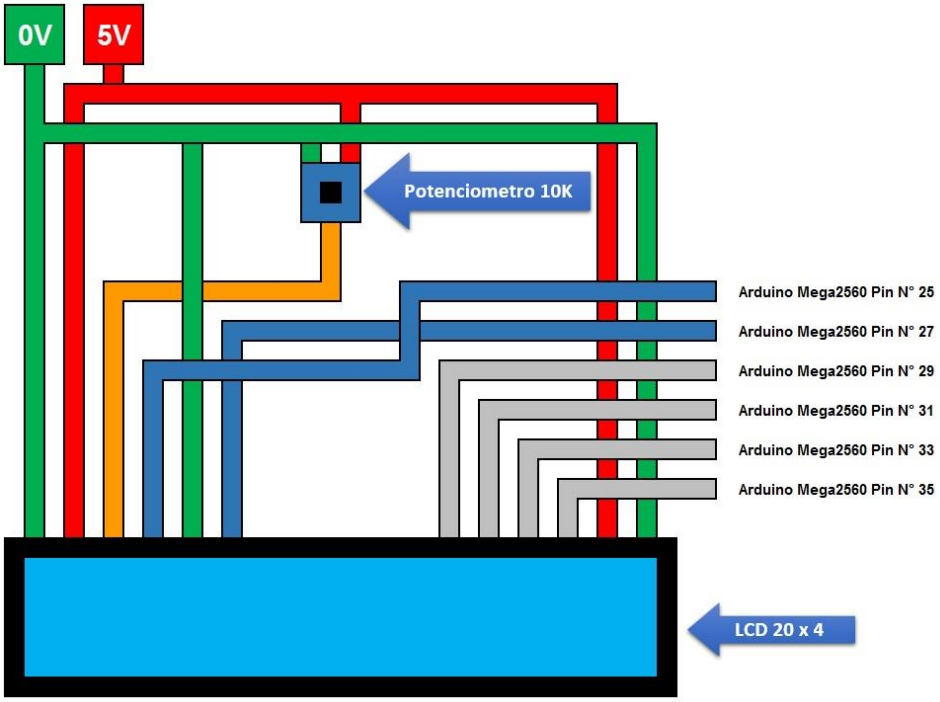
El pulsador (R3) será el encargado de cambiar las velocidades, como dijimos, configuramos en este proyecto la velocidad por defecto a NORMAL, y con cada pulso accionado en el pulsador (R3) bajará la velocidad respectivamente a la siguiente tabla:

El led Rojo avisa VELOCIDAD MUY RAPIDO
El led Naranja avisa VELOCIDAD RAPIDO
El led Amarillo avisa VELOCIDAD NORMAL
El led Verde avisa VELOCIDAD LENTO
El led Azul avisa VELOCIDAD MUY LENTO
Luego de llegar a VELOCIDAD MUY LENTO con otro pulso del pulsador (R3), cambiara a VELOCIDAD MUY RAPIDO, continuando así de esta manera el ciclo.

LCD

Acá debo aclarar que esta parte no es esencial, pero contribuye al aprendizaje y queda mucho más profesional al poder monitorizar todos los eventos, en un Display Lcd.

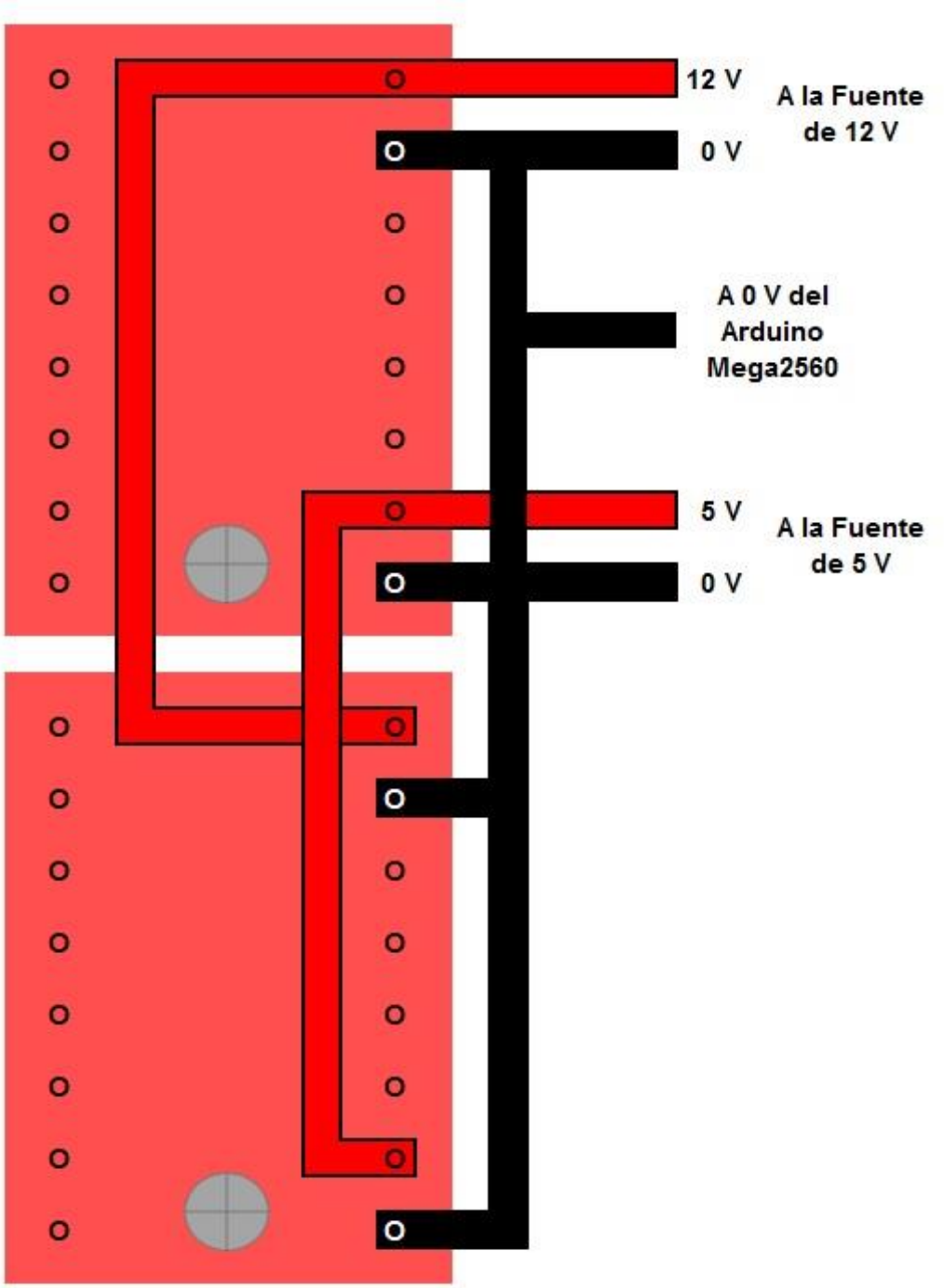
Materiales:
Display Lcd 2004 Back light 20×4 Hd44780.
1 Potenciómetro de 10K Ω.
Usamos este Display Lcd ya que es bastante económico y fácil de conseguir.



Las conexiones a pesar que podría parecer difícil, es bastante sencilla. Solo se debe prestar mucha atención a las conexiones.
El pin 1 del LCD a 0V.
El pin 5 del LCD a 0V.
El pin 16 del LCD a 0V.
El pin 2 del LCD a 5V.
El pin 2 del LCD a 5V.
El pin 15 del LCD a 5V.
El pin 3 del LCD al pin del centro del potenciómetro de 10K Ω, los 2 pines sobrantes del potenciómetro van a 0V y 5V.
El pin 4 del LCD al pin 25 del Arduino Mega2560.
El pin 6 del LCD al pin 27 del Arduino Mega2560.
El pin 11 del LCD al pin 29 del Arduino Mega2560.
El pin 12 del LCD al pin 31 del Arduino Mega2560.
El pin 13 del LCD al pin 33 del Arduino Mega2560.
El pin 14 del LCD al pin 35 del Arduino Mega2560.

Cuando encienda el LCD, mover el potenciómetro hasta lograr el mejor ajuste de visibilidad de la pantalla.

Ahora, por último y más importante, la conexión de alimentación de los drivers A4988, para que puedan funcionar los 2 motores paso a paso.



Mucha precaución al momento de conectar los voltajes.
(Se recomienda sacar los drivers A4988 antes de darle voltaje)
Y una vez terminado, volver a conectarlos.
Por este motivo, he construido una placa para la facilitación y manipulación de dichos drivers.
No olvide conectar los 0V de los 3 suministros de voltaje entre sí, (5V) + (12V) + (ARDUINO). Esta especificado en las conexiones, pero lo reitero, ya que la falta de uno de ellos podría causar fallos en el funcionamiento.

Calibración del driver A4988

¿Por qué ajustar la corriente?

Porque l fuerza (realmente, el par motor) que va a ejercer un motor paso a paso depende directamente de la corriente que circula por él.

Más corriente = más fuerza.

Pero también, nuestros drivers A4988 tienen un valor máximo de corriente que pueden entregar con seguridad.

El valor óptimo de corriente al que debemos regular los drivers estará comprendido entre ambos valores.

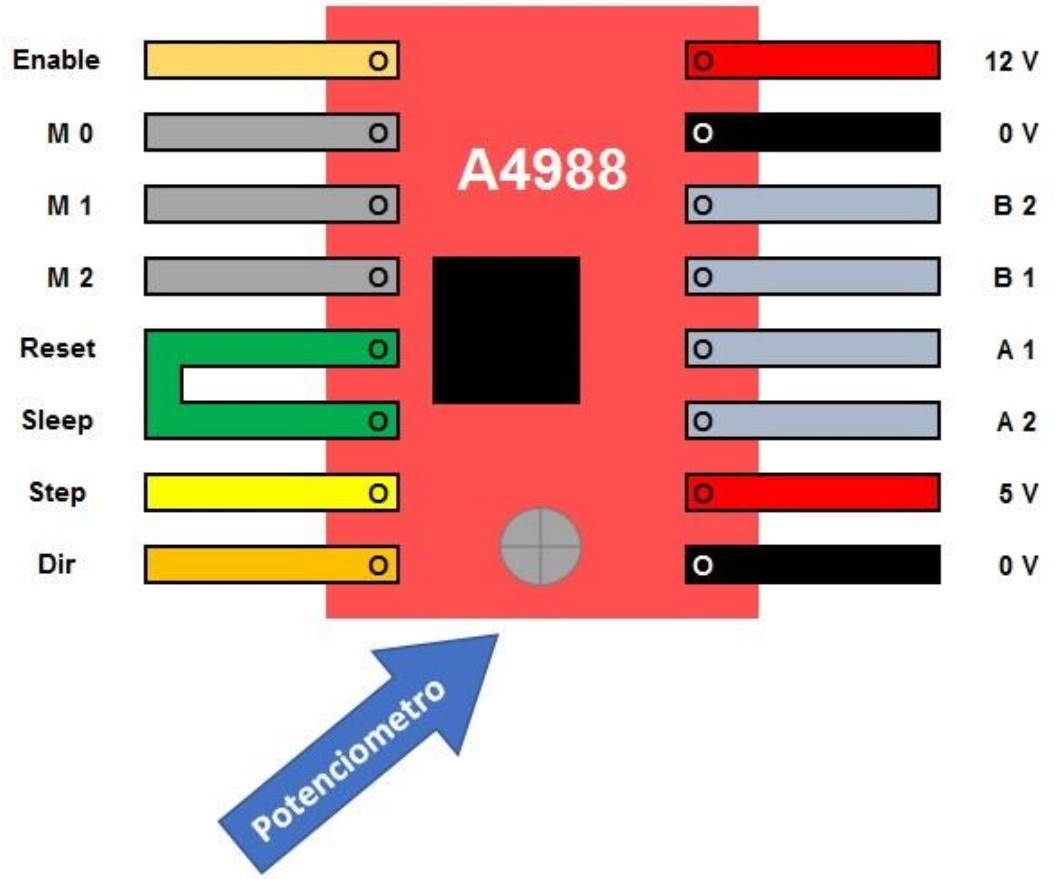
(Regla de oro):

MINIMA CORRIENTE que necesita el motor paso a paso, para ejercer suficiente fuerza para el proyecto.	<	Valor al que calibraremos la corriente del driver A4988.	<	MAXIMA CORRIENTE que puede suministrar nuestro driver A4988.

Como he explicado con anterioridad, yo no estudié nada de esto, he aprendido buscando información en internet o libros.
Y por tal motivo he aprendido que hay siempre más de una manera de hacer las cosas.
Ahora explicare muy básicamente como calibrar el driver A4988, de la manera que yo lo hago, no es muy profesional, pero hablando con personal que se dedica a esto, me han enseñado a hacerlo de esta manera, ya que así es una de las maneras más fáciles de hacerlo, pero si ustedes quieren hacerlo correctamente, pueden buscar información en internet y verán que los resultados son los mismos.
Con el driver A4988 alimentado y con todas las conexiones terminadas, pondremos en marcha la calibración.
Dato:
(Para calibrar el driver A4988 se necesitará un destornillador punta Phillips muy chico).
Si en su caso, les pasa como a mí que tienen de estos destornilladores que son todos metálicos, deben cubrir con cinta aislante todo el mango al momento de usarlo.

	No tienen nada de peligro, pero al tocar el potenciómetro con estos destornilladores, genera mucho ruido por interferencia y aparte de ser molesto, no dejará escuchar bien la calibración, así que aislarlos ayuda a reducir el ruido.

Usaremos un driver solo y el otro desconectado, así es más fácil.
Dejamos en automático el giro del motor paso a paso que vamos a calibrar.
Con todo preparado, comenzamos la calibración.
Debemos poner el driver visto de esta manera.



Con ayuda del destornillador Phillips aislado, vamos girando en sentido horario, subiendo el amperaje entregado, o anti horario, bajando el amperaje entregado, y vamos viendo la fuerza del motor paso a paso y el ruido.
Nos daremos cuenta que esta calibrado cuando vemos que gira y queremos frenarlo con la mano y cuesta, (pero sin ruido), si hay ruido vamos bajando de a poquito hasta no escuchar ningún chirrido, si está bien el ruido, vemos si podemos frenar el motor paso a paso con facilidad si es así, subimos un poco sentido horario, pero muy poco hasta ver que tenga fuerza y listo, en menos de 4 o 5 intentos seguro lo calibran, yo ya acostumbrado lo calibro en 2 o 3 intentos, luego lo dejo marcado con fibra indeleble y procedo a calibrarlo con las fórmulas como debe ser y queda en la misma posición, así que esta manera es mucho más práctica para quienes no sabemos tanto, o somos más perezosos.
Luego de hacerlo con un driver, hacemos lo mismo con el otro.
Yo lo que siempre hago si es que los 2 motores paso a paso son idénticos, es calibrar un driver A4988 sacarlo y poner el otro driver A4988 y el calibrado dejarlo para el otro motor paso a paso, así es más fácil, ya que uso el mismo código para las 2 calibraciones.
Ojo, al momento de sacar el driver A4988 para poner el otro, deben desconectar la alimentación de 12 volts.
De esta manera se aseguran que no podrán ocasionar ningún desperfecto por tocar algún pin involuntariamente haciendo que el driver A4988 tenga mal funcionamiento en la posterioridad.

Código para el **Arduino Mega2560**, aunque podríamos usar el **Arduino Uno R3,** solo para poder calibrar los **Driver A4988.**

```
//  Pin 6, 7 y 8 Habilitador de los pasos del Driver A4988 = M0, M1 y M2
int PasosM0 = 6;
int PasosM1 = 7;
int PasosM2 = 8;

//  Divisor de los pasos por vueltas   Ideal 128
int Dividisor = 128;

// Pin Stepper Motor Paso a paso 1
const int Step = 9;

// Pin Dirección motor Paso a paso 1
const int Direccion = 10;

//   Velocidad en RPM     60 a 600,    200 Ideal
const int PasosRPM = 200;

//  Pasos por vuelta de nuestro motor 200 es una vuelta 1.8°     50 a 400     Ideal 200
const int PasosPorVueltas = 200;

//  Con la velocidad y el número de pasos por vuelta, podemos calcular el periodo con el que debemos actuar sobre el pin Steepper.
//  PasosPeriodo = 600 * (PasosRPM / 60) / PasosPorVueltas;  //   100 a 600        Ideal 200
const int PasosPeriodo = 300 * (PasosRPM / 100) / PasosPorVueltas;

void setup() {
//  Habilitamos como salida
  pinMode(Habilitador, OUTPUT);
  pinMode(PasosM0, OUTPUT);
  pinMode(PasosM1, OUTPUT);
  pinMode(PasosM2, OUTPUT);

 //  Vamos a escribir sobre los pines Step y Dir, que son salidas
pinMode(Step, OUTPUT);
pinMode(Direccion, OUTPUT);
}

void loop() {
  digitalWrite(Direccion, HIGH);
  for (uint32_t i = 0; i < PasosPorVueltas / Dividisor ; i++) {
    digitalWrite(Step, HIGH);
    delay(PasosPeriodo / 0.65 );
    digitalWrite(Step, LOW);
    delay(PasosPeriodo / 0.65 );
}
```

Listo, con este pequeño código, podremos calibrar el driver A4988.

Luego de calibrar, borraremos este código, ya que el que usaremos tiene 2 motores paso a paso con 2 driver, y algunos parámetros los nombramos con terminación de 1 y 2.
Ejemplo: Step se reemplaza por Step1 y Step2.
Ahora que todo está conectado, y funcionando, procedemos al código del proyecto final.

La programación la he creado totalmente desde cero.
Así que sepan disculpar a los más experimentados las pequeñas imperfecciones que tiene este proyecto, pero como dije antes, es solo con fines didácticos.

La información buscada y usada ha sido suministrada por internet sobre:

Estructuras:
- Estructuras de Control
- Operadores de comparación
- Operadores aritméticos
- Operadores booleanos
- Operadores compuestos

Variables:
- Constantes
- Tipos de datos
- Conversión
- Alcance variable y calificadores

Funciones
- Digital I/O = E/S (Entrada / Salida)
- Análogo I/O = E/S (Entrada / Salida)
- Avanzada I/O = E/S (Entrada / Salida)
- Time = Hora
- Math = Matemática
- Comunicación

Desde la propia web de Arduino.
https://www.arduino.cc/en/Reference/HomePage

Y también algunos valores, rutinas o partes de códigos para poder entender algunos conceptos básicos.

Este proyecto lo había terminado mucho antes, pero no me gustaba la programación, así que tardé un poco más para poder hacerlo más prolijo, ya que el anterior código era mucho más largo casi el triple, y tenía muchas desprolijidades que a la hora de modificar algunos parámetros se complicaba, ya que no era tan legible y aparte no había documentado casi nada, y a veces no tenía ni idea de que hacia algún fragmento de código o cuando buscaba algo para modificar no encontraba donde estaba programado, y perdía mucho tiempo y a veces terminaba reprogramándolo ya que era más fácil que buscar el error.

Igual seguramente encontraran algunas asperezas más que limar, pero bueno, para eso están ustedes lectores que ayudaran a la causa.

CODIGO FINAL:

Librería del LCD:

Esta librería LiquiddCrystal.h permite a una placa Arduino controlar las pantallas de LiquidCrystal (LCD) basadas en el Hitachi HD44780 (o un chipset compatible), que se encuentra en la mayoría de las pantallas LCD basadas en texto. La biblioteca funciona con el modo de 4 o 8 bits (es decir, usando 4 o 8 líneas de datos además de las rs, enable y opcionalmente las líneas de control rw).

https://www.arduino.cc/en/Reference/LiquidCrystal

Incluimos la librería LiquiddCrystal.h y definimos los pines a usar del Arduino.

```
//-------------------------------------------------------------------------------------------------------------------------
//**************************
//**                       *
//**  (1)           (4)    *   Tecla 1 y 2
//**                       *
//**                       *
//**        (M)            *   Motores
//**                       *
//**                       *
//**  (2)           (3)    *   Teclas 3 y 4
//**                       *
//**************************
//-------------------------------------------------------------------------------------------------------------------------
#include <LiquidCrystal.h>
//-------------------------------------------------------------------------------------------------------------------------
// Pin 4 LCD al pin 25 Arduino
// Pin 6 LCD al pin 27 Arduino
// Pin 1 LCD al pin 29 Arduino
// Pin 12 LCD al pin 31 Arduino
// Pin 13 LCD al pin 33 Arduino
// Pin 14 LCD al pin 35 Arduino
LiquidCrystal lcd(25, 27, 29, 31, 33, 35);
//-------------------------------------------------------------------------------------------------------------------------
```

Declaramos las Variables:

```
//-------------------------------------------------------------------------------------------------------------------------
int PasosM0 = 6;                //   Pin 6 Habilitador de los pasos del Driver A4988 = M0
int PasosM1 = 7;                //   Pin 7 Habilitador de los pasos del Driver A4988 = M1
int PasosM2 = 8;                //   Pin 8 Habilitador de los pasos del Driver A4988 = M2
//-------------------------------------------------------------------------------------------------------------------------
int CoordenadaX = A0;           //   Pin Analogico A0 Comando X del joystick
int CoordenadaY = A1;           //   Pin Analogico A1 Comando Y del joystick
//-------------------------------------------------------------------------------------------------------------------------
int ContadorPulsosBotonR3 = 2;  // Contador con 0 así cuando arranca comienza con velocidad normal de las pulsaciones de botón R3 del joystick
//-------------------------------------------------------------------------------------------------------------------------
int EstadoDelBotonR3 = 0;       // Estado actual del botón R3 del joystick
//-------------------------------------------------------------------------------------------------------------------------
int UltimoEstadoDelBotonR3 = 0; // Estado anterior del botón R3 del joystick
//-------------------------------------------------------------------------------------------------------------------------
int Habilitador = 13;            // Pin 13 Habilitador de los Driver A4988
//-------------------------------------------------------------------------------------------------------------------------
int Dividisor = 128;             //   Divisor de los pasos por vueltas   Ideal 128
//-------------------------------------------------------------------------------------------------------------------------
int PosicionX = 0;     //   Posicion X del jostick
int PosicionY = 0;     //   Posicion Y del jostick
//-------------------------------------------------------------------------------------------------------------------------
int Boton1 = 5;             // Definimos el pin de la entrada a la tecla 1
int Boton2 = 2;             // Definimos el pin de la entrada a la tecla 2
int Boton3 = 3;             // Definimos el pin de la entrada a la tecla 3
int Boton4 = 4;             // Definimos el pin de la entrada a la tecla 4
//-------------------------------------------------------------------------------------------------------------------------
```

Declaramos las Constantes Variables:

```
//-----------------------------------------------------------------------------------------------------------------------
const int  BotonR3 = 24;         // Pin 24 conectado el pulsador R3 del joystick
//-----------------------------------------------------------------------------------------------------------------------
const int  LedMuyRapido = 26;          // Pin para LED Rojo
const int  LedRapido = 28;             // Pin para LED Naranja
const int  LedNormal = 30;             // Pin para LED Amarillo
const int  LedLento = 32;              // Pin para LED Verde
const int  LedMuyLento = 34;           // Pin para LED Azul
//-----------------------------------------------------------------------------------------------------------------------
const int Step1 = 9;        // Pin Stepper Motor Paso a paso 1
const int Step2 = 11;       // Pin Stepper Motor Paso a paso 2
//-----------------------------------------------------------------------------------------------------------------------
const int Direccion1 = 10;  // Pin Direccion motor Paso a paso 1
const int Direccion2 = 12;  // Pin Direccion motor Paso a paso 2
//-----------------------------------------------------------------------------------------------------------------------
const int PasosRPM1 = 200;   //    Velocidad en RPM      60 a 600,    200 Ideal
const int PasosRPM2 = 200;   //    Velocidad en RPM      60 a 600,    200 Ideal
//-----------------------------------------------------------------------------------------------------------------------
const int PasosPorVueltas1 = 200;  //   Pasos por vuelta de nuestro motor 200 es una vuelta 1.8°      50 a 400     Ideal 200
const int PasosPorVueltas2 = 200;  //   Pasos por vuelta de nuestro motor 200 es una vuelta 1.8°      50 a 400     Ideal 200
//-----------------------------------------------------------------------------------------------------------------------
//   Con la velocidad y el numero de pasos por vuelta, podemos calcular el periodo con el que debemos actuar sobre el pin Steepper.
//   PasosPeriodo = 600 * (PasosRPM / 60 ) / PasosPorVueltas;  //   100 a 600         Ideal 200
//-----------------------------------------------------------------------------------------------------------------------
const int PasosPeriodo1 = 300 * (PasosRPM1 / 100) / PasosPorVueltas1;
const int PasosPeriodo2 = 300 * (PasosRPM2 / 100) / PasosPorVueltas2;
//-----------------------------------------------------------------------------------------------------------------------
```

En el Setup ()

```
//-----------------------------------------------------------------------------------------------------------------------
void setup() {
  //---------------------------------------------------------------------------------------------------------------------
  Serial.begin(9600);            //   Habilitamos el monitor serie a 9600
  //---------------------------------------------------------------------------------------------------------------------
  lcd.begin(20, 4);
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(Habilitador, OUTPUT);  //   Habilitamos como salida
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(PasosM0, OUTPUT);      //   Habilitamos como salida
  pinMode(PasosM1, OUTPUT);      //   Habilitamos como salida
  pinMode(PasosM2, OUTPUT);      //   Habilitamos como salida
  //---------------------------------------------------------------------------------------------------------------------
  pinMode( Boton1  , INPUT) ;   //   Botones como entradas
  pinMode( Boton2  , INPUT) ;   //   Botones como entradas
  pinMode( Boton3  , INPUT) ;   //   Botones como entradas
  pinMode( Boton4  , INPUT) ;   //   Botones como entradas
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(Step1, OUTPUT);       //   Vamos a escribir sobre los pines Stepper que son salidas
  pinMode(Step2, OUTPUT);       //   Vamos a escribir sobre los pines Stepper que son salidas
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(Direccion1, OUTPUT);  //   Vamos a escribir sobre los pines Direccion que son salidas
  pinMode(Direccion2, OUTPUT);  //   Vamos a escribir sobre los pines Direccion que son salidas
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(CoordenadaX, INPUT);  // Declarar pines como entrada jostick X analogico
  pinMode(CoordenadaY, INPUT);  // Declarar pines como entrada jostick Y analogico
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(BotonR3, INPUT_PULLUP);// Inicializar el pin del botón como entrada , Resistencia de pullup interna.
  //---------------------------------------------------------------------------------------------------------------------
  pinMode(LedMuyRapido, OUTPUT);     // Inicializa el pin 26 del LED Rojo como una salida
  pinMode(LedRapido, OUTPUT);        // Inicializa el pin 28 del LED Naranja como una salida
  pinMode(LedNormal, OUTPUT);        // Inicializa el pin 30 del LED Amarillo como una salida
  pinMode(LedLento, OUTPUT);         // Inicializa el pin 32 del LED Verde como una salida
  pinMode(LedMuyLento, OUTPUT);      // Inicializa el pin 34 del LED Azul como una salida
  //---------------------------------------------------------------------------------------------------------------------
}
//-----------------------------------------------------------------------------------------------------------------------
```

Acá en el Setup ingresamos lo que necesitamos que se inicia un al arrancar.
Ejemplo: para inicializar variables, modos de pin, iniciar utilizando bibliotecas, etc.
La función de configuración sólo se ejecutará una vez, después de cada encendido o reinicio de la placa Arduino.

En el Void Loop ()

```
//-------------------------------------------------------------------------------------------------------------------------
void loop() {
  //-------------------------------------------------------------------------------------------------------------------------
  //   Acciones de rutinas
  AvisoSistemaManual();
  VelocidadConJoystick();
  Joystick();
  SistemaAutomatizadoTeclas();
  //-------------------------------------------------------------------------------------------------------------------------
}
//-------------------------------------------------------------------------------------------------------------------------
void SistemaAutomatizadoTeclas() {
  //-------------------------------------------------------------------------------------------------------------------------
  int Valor1 = digitalRead(Boton1) ;   //   Leemos los Valores de las teclas
  int Valor2 = digitalRead(Boton2) ;   //   Leemos los Valores de las teclas
  int Valor3 = digitalRead(Boton3) ;   //   Leemos los Valores de las teclas
  int Valor4 = digitalRead(Boton4) ;   //   Leemos los Valores de las teclas
  //-------------------------------------------------------------------------------------------------------------------------
  if (Valor1  < 1 ||  Valor2  < 1 ||  Valor3  < 1 ||  Valor4  < 1 )   //   Condicion para habilitar los motores, velocidad y acciones
    //-----------------------------------------------------------------------------------------------------------------------
  {
    //-----------------------------------------------------------------------------------------------------------------------
    digitalWrite(Habilitador, LOW);   // Habilita corriente a los Motores PAP
    //-----------------------------------------------------------------------------------------------------------------------
    VelocidadNormal();                // Selecciona la velocidad de los Motores PAP
    AvisoSistemaAutomatico();
    //-----------------------------------------------------------------------------------------------------------------------
    Tecla1Normal();    //   Accion
    Tecla2Normal();    //   Accion
    Tecla3Normal();    //   Accion
    Tecla4Normal();    //   Accion
    //-----------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
  else
    //-----------------------------------------------------------------------------------------------------------------------
  {
    //-----------------------------------------------------------------------------------------------------------------------
    digitalWrite(Habilitador, HIGH);         // Desabilita corriente a los Motores PAP
    //-----------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
}
//-------------------------------------------------------------------------------------------------------------------------
```

Rutinas y SubRutinas:

Sistema Automatizado de las Teclas:

Esta SubRutina verifica si alguna de las 4 teclas esta activada y si es así ejecuta acciones, de lo contrario no hace nada y queda en espera, apagando el habilitador de los motores paso a paso, ahorrando energía, y también selecciona la velocidad normal para la acción y da aviso visual al LCD de que se está ejecutando en automático y a la velocidad normal.

```
//-------------------------------------------------------------------------------------------------------------------------
void SistemaAutomatizadoTeclas() {
  //-------------------------------------------------------------------------------------------------------------------------
  int Valor1 = digitalRead(Boton1) ;   //   Leemos los Valores de las teclas
  int Valor2 = digitalRead(Boton2) ;   //   Leemos los Valores de las teclas
  int Valor3 = digitalRead(Boton3) ;   //   Leemos los Valores de las teclas
  int Valor4 = digitalRead(Boton4) ;   //   Leemos los Valores de las teclas
  //-------------------------------------------------------------------------------------------------------------------------
  if (Valor1  < 1 ||  Valor2  < 1 ||  Valor3  < 1 ||  Valor4  < 1 )   //   Condicion para habilitar los motores, velocidad y acciones
    //-----------------------------------------------------------------------------------------------------------------------
  {
    //-----------------------------------------------------------------------------------------------------------------------
    digitalWrite(Habilitador, LOW);   // Habilita corriente a los Motores PAP
    //-----------------------------------------------------------------------------------------------------------------------
    VelocidadNormal();                // Selecciona la velocidad de los Motores PAP
    AvisoSistemaAutomatico();
    //-----------------------------------------------------------------------------------------------------------------------
    Tecla1Normal();    //   Accion
    Tecla2Normal();    //   Accion
    Tecla3Normal();    //   Accion
    Tecla4Normal();    //   Accion
    //-----------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
  else
    //-----------------------------------------------------------------------------------------------------------------------
  {
    //-----------------------------------------------------------------------------------------------------------------------
    digitalWrite(Habilitador, HIGH);         // Desabilita corriente a los Motores PAP
    //-----------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
}
//-------------------------------------------------------------------------------------------------------------------------
```

Tecla1

```
//-------------------------------------------------------------------------------------------------------------------------
void TeclalNormal() {
  VelocidadNormal();                        //  Selecciona la velocidad del motor PAP
  AvisoLCDVelocidad3();                     //  Visualiza en LCD la velocidad Normal
  int Valor1 = digitalRead(Boton1) ;        //  Leemos el Valor de la tecla
  if (Valor1 < 1 )  {                       //  Si el valor es menor a 1 ejecuta accion
    delay(500);                             //  Agrega un tiempo de medio segundo antes de comenzar
    for (int i = 0; i <= 140; i++) {        //  Cantidad de pusos, para el movimiento
      MotorVerticalIzquierda();             //  Accion de un motor PAP y giro expecifico
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 140; i++) {
      MotorVerticalDerecha();
    }
  }
}
//-------------------------------------------------------------------------------------------------------------------------
```

Tecla2, Tecla3 y Tecla4

```
//-----------------------------------------
void Tecla2Normal() {
  VelocidadNormal();
  AvisoLCDVelocidad3();
  int Valor2 = digitalRead(Boton2) ;
  if (Valor2 < 1 ) {
    delay(500);
    for (int i = 0; i <= 130; i++) {
      MotorVerticalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 130; i++) {
      MotorVerticalIzquierda();
    }
  }
}
//-----------------------------------------
```

```
//-----------------------------------------
void Tecla3Normal() {
  VelocidadNormal();
  AvisoLCDVelocidad3();
  int Valor3 = digitalRead(Boton3) ;
  if (Valor3 < 1 )  {
    delay(500);
    for (int i = 0; i <= 255; i++) {
      MotorVerticalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 255; i++) {
      MotorVerticalIzquierda();
    }
  }
}
//-----------------------------------------
```

```
//-----------------------------------------
void Tecla4Normal() {
  VelocidadNormal();
  AvisoLCDVelocidad3();
  int Valor4 = digitalRead(Boton4) ;
  if (Valor4 < 1 ) {
    delay(500);
    for (int i = 0; i <= 270; i++) {
      MotorVerticalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 270; i++) {
      MotorVerticalDerecha();
    }
  }
}
//-----------------------------------------
```

Estas 4 teclas accionadas en simultaneo o individualmente, ejecutan acciones preestablecidas. Vea como las 4 subrutinas son muy parecidas, aunque los motores se ejecutan casi en el mismo orden, pero distintos giros (Horario y AntiHorario), haciendo que ejecute la automatización programada.

También selecciona la velocidad normal para la acción y da aviso visual al LCD de que se está ejecutando en automático y a la velocidad normal.

Como habrán notado, Tecla1, Tecla2, Tecla3 y Tecla4 son subrutinas.

Movimiento de los 2 motores Paso a Paso (Giro Horario y Giro AntiHorario)

```
//--------------------------------------------------------------------------
//   Movimientos de las 2 posiciones de cada Motor PAP
//--------------------------------------------------------------------------
void MotorHorizontalIzquierda() {
  digitalWrite(Direccion1, HIGH);
  for (uint32_t i = 0; i < PasosPorVueltas1 / Dividisor ; i++) {
    digitalWrite(Step1, HIGH);
    delay(PasosPeriodo1 / 0.65 );
    digitalWrite(Step1, LOW);
    delay(PasosPeriodo1 / 0.65 );
  }
}
//--------------------------------------------------------------------------
void MotorHorizontalDerecha() {
  digitalWrite(Direccion1, LOW);
  for (uint32_t i = 0; i < PasosPorVueltas1 / Dividisor ; i++) {
    digitalWrite(Step1, HIGH);
    delay(PasosPeriodo1 / 0.65  );
    digitalWrite(Step1, LOW);
    delay(PasosPeriodo1 / 0.65 );
  }
}
//--------------------------------------------------------------------------
void MotorVerticalIzquierda() {
  digitalWrite(Direccion2, HIGH);
  for (uint32_t i = 0; i < PasosPorVueltas2 / Dividisor ; i++) {
    digitalWrite(Step2, HIGH);
    delay(PasosPeriodo2 / 0.65 );
    digitalWrite(Step2, LOW);
    delay(PasosPeriodo2 / 0.65 );
  }
}
//--------------------------------------------------------------------------
void MotorVerticalDerecha() {
  digitalWrite(Direccion2, LOW);
  for (uint32_t i = 0; i < PasosPorVueltas2 / Dividisor ; i++) {
    digitalWrite(Step2, HIGH);
    delay(PasosPeriodo2 / 0.65  );
    digitalWrite(Step2, LOW);
    delay(PasosPeriodo2 / 0.65 );
  }
}
//--------------------------------------------------------------------------
```

Acá no daremos tantos detalles ya que es mucha matemática, solo lo importante.
Elige la dirección de giro, (Horario o AntiHorario) = (Dirección, LOW) o (Dirección, HIGH)
Luego con estos valores:
Pasos por vuelas se divide en nuestro caso por el divisor (128).
Pasos por periodo lo divide a 0.65
Step, LOW y Step, HIGH son los pasos del motor.
Todo esto en resumen es lo que da los distintos pulsos para mover el motor.

Las 5 Velocidades

Como hemos visto en más arriba las entradas M0, M1, M2 del driverA4988 son las que manejan los pasos y micropasos.
Siendo estos los que mueven el motor con un pulso los distintos grados.

Tabla Real:

Estado	MS0	MS1	MS2
	LOW	LOW	LOW
	HIGH	LOW	LOW
	LOW	HIGH	LOW
	HIGH	HIGH	LOW
	HIGH	HIGH	HIGH

Tabla del proyecto:

En la Programacion configuraremos asi:				
Completo	0	0	0	Muy Rapido
1/2	1	0	0	Rapido
1/4	0	1	0	Normal
1/8	1	1	0	Lento
1/16	1	1	1	Muy Lento

Estos distintos micropasos son muy útiles, porque a veces necesitamos más velocidad, pero otras veces necesitamos mayor precisión.

En este proyecto declaramos la Constante Variable
Pasos por vueltas en 200
Y un giro completo tiene 360°
Si dividimos 360° en 200 pasos nos da un Angulo de 1.8° por paso.
Que es la resolución de la mayoría de los motores nemas 17.

Para la automatización, lo dejaremos LOW, HIGH, LOW, a 1/4 de pasos, así no es tan lento, pero teniendo bastante precisión.
Los pasos más lentos son los de más precisión, y los pasos más rápidos son menos precisos dificultando a veces el exacto posicionamiento.
El paso 1/16 es el más apropiado para usar, cuando las tareas demanden mucha precisión.

Ejemplo:
Si lo tenemos a velocidad normal LOW, HIGH, LOW, a 1/4 de pasos, nos daría estos resultados:

$$1/4 \text{ de } 1 \times 200 \times 1.8° = 0.25 \times 200 \times 1.8° = 50 \times 1.8° = 90°$$

Como verán LOW, HIGH, LOW, a 1/4 de pasos nos da un giro de 90°, pero recuerden que nosotros usamos un divisor de 128 y también dividimos el paso de periodo a 0.65, así que con un solo pulso no girará 90° sino muchísimo menos.

```
//-------------------------------------------------------------------------------
void VelocidadMuyRapido() {
  digitalWrite(PasosM0, LOW);
  digitalWrite(PasosM1, LOW);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, HIGH);              // Enciende el LED Rojo
  digitalWrite(LedRapido, LOW);                  // Apaga el LED Naranja
  digitalWrite(LedNormal, LOW);                  // Apaga el LED Amarillo
  digitalWrite(LedLento, LOW);                   // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, LOW);                // Apaga el LED Azul
}
//-------------------------------------------------------------------------------
void VelocidadRapido() {
  digitalWrite(PasosM0, HIGH);
  digitalWrite(PasosM1, LOW);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, LOW);               // Apaga el LED Rojo
  digitalWrite(LedRapido, HIGH);                 // Enciende el LED Naranja
  digitalWrite(LedNormal, LOW);                  // Apaga el LED Amarillo
  digitalWrite(LedLento, LOW);                   // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, LOW);                // Apaga el LED Azul
}
//-------------------------------------------------------------------------------
void VelocidadNormal() {
  digitalWrite(PasosM0, LOW);
  digitalWrite(PasosM1, HIGH);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, LOW);               // Apaga el LED Rojo
  digitalWrite(LedRapido, LOW);                  // Apaga el LED Naranja
  digitalWrite(LedNormal, HIGH);                 // Enciende el LED Amarillo
  digitalWrite(LedLento, LOW);                   // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, LOW);                // Apaga el LED Azul
}
//-------------------------------------------------------------------------------
void VelocidadLento() {
  digitalWrite(PasosM0, HIGH);
  digitalWrite(PasosM1, HIGH);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, LOW);               // Apaga el LED Rojo
  digitalWrite(LedRapido, LOW);                  // Apaga el LED Naranja
  digitalWrite(LedNormal, LOW);                  // Apaga el LED Amarillo
  digitalWrite(LedLento, HIGH);                  // Enciende el LED Amarillo
  digitalWrite(LedMuyLento, LOW);                // Apaga el LED Azul
}
//-------------------------------------------------------------------------------
void VelocidadMuyLento() {
  digitalWrite(PasosM0, HIGH);
  digitalWrite(PasosM1, HIGH);
  digitalWrite(PasosM2, HIGH);
  digitalWrite(LedMuyRapido, LOW);               // Apaga el LED Rojo
  digitalWrite(LedRapido, LOW);                  // Apaga el LED Naranja
  digitalWrite(LedNormal, LOW);                  // Apaga el LED Amarillo
  digitalWrite(LedLento, LOW);                   // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, HIGH);               // Enciende el LED Azul
}
//-------------------------------------------------------------------------------
```

El Joystick Analógico

```
//-------------------------------------------------------------------------------------------------------
void Joystick() {
  //-------------------------------------------------------------------------------------------------------
  PosicionX = analogRead(CoordenadaX);            //   Lee posicion de X
  PosicionY = analogRead(CoordenadaY);            //   Lee posicion de Y
  //-------------------------------------------------------------------------------------------------------
  if (PosicionX  > 950 ||  PosicionX  < 60 ||  PosicionY  > 950 ||  PosicionY  < 60 )
    //-------------------------------------------------------------------------------------------------------
  {
    //-------------------------------------------------------------------------------------------------------
    AvisoSistemaManual();
    digitalWrite(Habilitador, LOW);   // Habilita corriente a los Motores PAP
    //-------------------------------------------------------------------------------------------------------
  } else {
    digitalWrite(Habilitador, HIGH);   // Desabilita corriente a los Motores PAP
    //-------------------------------------------------------------------------------------------------------
    //-------------------------------------------------------------------------------------------------------
  }
  if (PosicionX > 950) {
    MotorHorizontalIzquierda();
  }
  //-------------------------------------------------------------------------------------------------------
  if (PosicionX < 60) {
    MotorHorizontalDerecha();
  }
  //-------------------------------------------------------------------------------------------------------
  if (PosicionY < 60) {
    MotorVerticalIzquierda();
  }
  //-------------------------------------------------------------------------------------------------------
  if (PosicionY > 950) {
    MotorVerticalDerecha();
  }
  //-------------------------------------------------------------------------------------------------------
```

Lee las coordenadas de X e Y, y las declara dentro de PosicionX y PosicionY.
Luego ya que los motores paso a paso se encuentran en estado de espera, esta rutina verifica que los valores estén en ciertos valores para que se ejecuten, y den aviso también al LCD de que esta en manual, también habilita los motores solo si algunas de las coordenadas superan ciertos limites, así ahorramos energía en los momentos de espera.

Si se da las condiciones mencionadas ejecutara las acciones que están establecidas individualmente en cada coordenada de X e Y girando el motor en sentido Horario o AntiHorario según sea el caso.

El uso del Joystick es muy útil en estos proyectos, por ejemplo, el más utilizado es el de poder poner a punto el posicionamiento de los motores para que ejecuten automatizaciones con total precisión, ya que unos pocos grados o milímetros desplazados cualquiera de los 2 motores paso a paso podría significar la falla en la tarea a realizar y quedando en un bucle infinito o tal vez la rotura de alguna parte mecánica dependiendo en que se utilice. Por ejemplo, en este proyecto cuando presionemos una tecla la automatización debería volver a presionarlo para apagar el ciclo, pero ese grado o milímetro mal posicionado haría que no pueda concretar la rutina, y al volver al punto de retorno, detectaría nuevamente la tecla que no pudo accionar y volvería a ejecutar la acción infinitamente, así hasta que corrijamos el error o presionemos la tecla nosotros.
Más adelante solucionaremos este inconveniente y que se autorregule el posicionamiento con sensores, estuve pesando mucho en hacerlo, pero como el proyecto es chico el uso de más cables cerca de los motores era un poco molesto y no era m uy estético, así que por el momento lo postergo, hasta la próxima.

Velocidad del Joystick Analógico

```
//-----------------------------------------------------------------------------------------------------------------------
void VelocidadConJoystick() {
  //-----------------------------------------------------------------------------------------------------------------------
  EstadoDelBotonR3 = digitalRead(BotonR3);
  //-----------------------------------------------------------------------------------------------------------------------
  if (EstadoDelBotonR3 != UltimoEstadoDelBotonR3)  // Compara el estado del botón con su estado anterior
  {
    //-----------------------------------------------------------------------------------------------------------------------
    if (EstadoDelBotonR3 == HIGH)                  // si el estado ha cambiado, incrementar el contador
    {
      //-----------------------------------------------------------------------------------------------------------------------
      ContadorPulsosBotonR3++;   // si el estado actual es ALTO entonces el botón incrementa
      //-----------------------------------------------------------------------------------------------------------------------
    }
  }
  //-----------------------------------------------------------------------------------------------------------------------
  UltimoEstadoDelBotonR3 = EstadoDelBotonR3;   // Guarda el estado actual como el último estado,para el próximo bucle
  //-----------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 == 1) {
    AvisoLCDVelocidad1();
    VelocidadMuyRapido();                        // Velocidad Muy Rapido
  }
  if (ContadorPulsosBotonR3 == 2) {
    AvisoLCDVelocidad2();
    VelocidadRapido();                            // Velocidad Rapido
  }
  if (ContadorPulsosBotonR3 == 3) {
    AvisoLCDVelocidad3();
    VelocidadNormal();                            // Velocidad Normal
  }
  if (ContadorPulsosBotonR3 == 4) {
    AvisoLCDVelocidad4();
    VelocidadLento();                             // Velocidad Lento
  }
  if (ContadorPulsosBotonR3 == 5) {
    AvisoLCDVelocidad5();
    VelocidadMuyLento();
  }
  if (ContadorPulsosBotonR3 > 5) {
    AvisoLCDVelocidad1();
    ContadorPulsosBotonR3 = 1;
  }
}
//-----------------------------------------------------------------------------------------------------------------------
```

Lee el estado del BotonR3 del Joystick para variar la velocidad.
Compara el estado actual del BotonR3 del Joystick con el estado anterior y si es distinto incrementa su valor.
Luego guarda el nuevo valor.

La secuencia real es así:
Por defecto arranca en 3 (VELOCIDAD NORMAL)

Con un pulso del BotonR3 del Joystick pasa a 4 y cambia la velocidad a (LENTO).

Con otro pulso del BotonR3 del Joystick pasa a 5 y cambia la velocidad a (MUY LENTO).

Con otro pulso del BotonR3 del Joystick pasa a 6 “pero” como hay un IF que dice que si el contador es mayor a 5 debe volver su valor a 1, así que pasa a 1 y cambia la velocidad a (MUY RAPIDO).

Con otro pulso del BotonR3 del Joystick pasa a 2 y cambia la velocidad a (RAPIDO).

Con un pulso del BotonR3 del Joystick pasa a 3 y cambia la velocidad a (NORMAL).

Y así vamos repitiendo el ciclo indefinidamente.
Originalmente había pensado en 2 botones distintos uno que suba y otro que baje la velocidad, pero pensé en aprovechar el BotonR3 del Joystick, ya que no tenía otro uso.

Avisos Visuales del LCD: (Automático o Manual) + (Velocidades)

```
//--------------------------------------------------------------------------------
void AvisoLCDVelocidad1() {
  lcd.setCursor(0, 3);          //  lcd.setCursor(Columna, Fila);
  lcd.print("Velocidad ");      //  Texto
  lcd.setCursor(10, 3);
  lcd.print("Muy Rapido");
}
//--------------------------------------------------------------------------------
void AvisoLCDVelocidad2() {
  lcd.setCursor(0, 3);  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);  lcd.print("Rapido    ");
}
void AvisoLCDVelocidad3() {
  lcd.setCursor(0, 3);  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);  lcd.print("Normal    ");
}
//--------------------------------------------------------------------------------
void AvisoLCDVelocidad4() {
  lcd.setCursor(0, 3);  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);  lcd.print("Lento     ");
}
//--------------------------------------------------------------------------------
void AvisoLCDVelocidad5() {
  lcd.setCursor(0, 3);  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);  lcd.print("Muy Lento ");
}
//--------------------------------------------------------------------------------
void AvisoSistemaManual() {
  lcd.setCursor(0, 2);
  lcd.print("Sistema Manual      ");
}
//--------------------------------------------------------------------------------
void AvisoSistemaAutomatico() {
  lcd.setCursor(0, 2);
  lcd.print("Sistema Automatico  ");
}
//--------------------------------------------------------------------------------
```

En esta parte tampoco entrare en muchos detalles ya que son solo subrutinas de aviso al LCD y con solo mirar se podrá apreciar que hace cada una de ellas.

Lcd.setCursor(Columna, Filas); esta parte del código selecciona desde donde comienza a cargar el texto en el LCD (Imprimir).
Ejemplo:
Las coordenadas (0,0) se ubica arriba a la izquierda, en la primera fila, y primera columna.
Las coordenadas (1,1) se ubica arriba a la izquierda, en la segunda fila, y segunda columna.
Las coordenadas (10,3) se ubica abajo en el centro, en la cuarta fila, y decima columna.

Lcd. print(“Texto”); texto a cargar (Imprimir).
Para evitar usar el clear para borrar decidí, llenar los espacios en blanco con el símbolo de espacio hasta completar las 20 columnas, así evito errores.

Código Completo en formato de Texto

```
//-----------------------------------------------------------------------------------------------------------------------
//****************************
//**                        *
//**  (1)             (4)  *   Tecla 1 y 2
//**                        *
//**                        *
//**         (M)            *   Motores
//**                        *
//**                        *
//**  (2)             (3)  *   Teclas 3 y 4
//**                        *
//****************************
//-----------------------------------------------------------------------------------------------------------------------
#include <LiquidCrystal.h>
//-----------------------------------------------------------------------------------------------------------------------
LiquidCrystal lcd(25, 27, 29, 31, 33, 35);
//-----------------------------------------------------------------------------------------------------------------------
int PasosM0 = 6;                //   Pin 6 Habilitador de los pasos del Driver A4988 = M0
int PasosM1 = 7;                //   Pin 7 Habilitador de los pasos del Driver A4988 = M1
int PasosM2 = 8;                //   Pin 8 Habilitador de los pasos del Driver A4988 = M2
//-----------------------------------------------------------------------------------------------------------------------
int CoordenadaX = A0;           //   Pin Analogico A0 Comando X del joystick
int CoordenadaY = A1;           //   Pin Analogico A1 Comando Y del joystick
//-----------------------------------------------------------------------------------------------------------------------
const int  BotonR3 = 24;        // Pin 24 conectado el pulsador R3 del joystick
//-----------------------------------------------------------------------------------------------------------------------
const int  LedMuyRapido = 26;         // Pin para LED Rojo
const int  LedRapido = 28;            // Pin para LED Naranja
const int  LedNormal = 30;            // Pin para LED Amarillo
const int  LedLento = 32;             // Pin para LED Verde
const int  LedMuyLento = 34;          // Pin para LED Azul
//-----------------------------------------------------------------------------------------------------------------------
int ContadorPulsosBotonR3 = 2;   // Contador con 0 asi cuando arranca comienza con velocidad normal de las pulsaciones
de botón R3 del joystick
//-----------------------------------------------------------------------------------------------------------------------
int EstadoDelBotonR3 = 0;        // Estado actual del botón R3 del joystick
//-----------------------------------------------------------------------------------------------------------------------
int UltimoEstadoDelBotonR3 = 0;  // Estado anterior del botón R3 del joystick
//-----------------------------------------------------------------------------------------------------------------------
int Habilitador = 13;             // Pin 13 Habilitador de los Driver A4988
//-----------------------------------------------------------------------------------------------------------------------
int Dividisor = 128;              //   Divisor de los pasos por vueltas   Ideal 128
//-----------------------------------------------------------------------------------------------------------------------
const int Step1 = 9;         // Pin Stepper Motor Paso a paso 1
const int Step2 = 11;        // Pin Stepper Motor Paso a paso 2
//-----------------------------------------------------------------------------------------------------------------------
const int Direccion1 = 10;  // Pin Direccion motor Paso a paso 1
const int Direccion2 = 12;  // Pin Direccion motor Paso a paso 2
//-----------------------------------------------------------------------------------------------------------------------
int Boton1 = 5;               // Definimos el pin de la entrada a la tecla 1
int Boton2 = 2;               // Definimos el pin de la entrada a la tecla 2
int Boton3 = 3;               // Definimos el pin de la entrada a la tecla 3
int Boton4 = 4;               // Definimos el pin de la entrada a la tecla 4
//-----------------------------------------------------------------------------------------------------------------------
const int PasosRPM1 = 200;   //    Velocidad en RPM      60 a 600,    200 Ideal
const int PasosRPM2 = 200;   //    Velocidad en RPM      60 a 600,    200 Ideal
//-----------------------------------------------------------------------------------------------------------------------
const int PasosPorVueltas1 = 200;  //   Pasos por vuelta de nuestro motor 200 es una vuelta 1.8°      50 a 400     Ideal 200
const int PasosPorVueltas2 = 200;  //   Pasos por vuelta de nuestro motor 200 es una vuelta 1.8°      50 a 400     Ideal 200
//-----------------------------------------------------------------------------------------------------------------------
//   Con la velocidad y el numero de pasos por vuelta, podemos calcular el periodo con el que debemos actuar sobre el pin
Steepper.
//   PasosPeriodo = 600 * (PasosRPM / 60 ) / PasosPorVueltas;  //   100 a 600        Ideal 200
//-----------------------------------------------------------------------------------------------------------------------
const int PasosPeriodo1 = 300 * (PasosRPM1 / 100) / PasosPorVueltas1;
const int PasosPeriodo2 = 300 * (PasosRPM2 / 100) / PasosPorVueltas2;
//-----------------------------------------------------------------------------------------------------------------------
int PosicionX = 0;     //   Posicion X del jostick
int PosicionY = 0;     //   Posicion Y del jostick
//-----------------------------------------------------------------------------------------------------------------------
void setup() {
  //-----------------------------------------------------------------------------------------------------------------------
```

```
  Serial.begin(9600);           //   Habilitamos el monitor serie a 9600
  //-------------------------------------------------------------------------------------------------------------------------
  lcd.begin(20, 4);
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(Habilitador, OUTPUT);  //   Habilitamos como salida
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(PasosM0, OUTPUT);      //   Habilitamos como salida
  pinMode(PasosM1, OUTPUT);      //   Habilitamos como salida
  pinMode(PasosM2, OUTPUT);      //   Habilitamos como salida
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode( Boton1  , INPUT) ;   //   Botones como entradas
  pinMode( Boton2  , INPUT) ;   //   Botones como entradas
  pinMode( Boton3  , INPUT) ;   //   Botones como entradas
  pinMode( Boton4  , INPUT) ;   //   Botones como entradas
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(Step1, OUTPUT);       //   Vamos a escribir sobre los pines Stepper que son salidas
  pinMode(Step2, OUTPUT);       //   Vamos a escribir sobre los pines Stepper que son salidas
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(Direccion1, OUTPUT);  //   Vamos a escribir sobre los pines Direccion que son salidas
  pinMode(Direccion2, OUTPUT);  //   Vamos a escribir sobre los pines Direccion que son salidas
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(CoordenadaX, INPUT);  // Declarar pines como entrada jostick X analogico
  pinMode(CoordenadaY, INPUT);  // Declarar pines como entrada jostick Y analogico
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(BotonR3, INPUT_PULLUP);// Inicializar el pin del botón como entrada , Resistencia de pullup interna.
  //-------------------------------------------------------------------------------------------------------------------------
  pinMode(LedMuyRapido, OUTPUT);     // Inicializa el pin 26 del LED Rojo como una salida
  pinMode(LedRapido, OUTPUT);        // Inicializa el pin 28 del LED Naranja como una salida
  pinMode(LedNormal, OUTPUT);        // Inicializa el pin 30 del LED Amarillo como una salida
  pinMode(LedLento, OUTPUT);         // Inicializa el pin 32 del LED Verde como una salida
  pinMode(LedMuyLento, OUTPUT);      // Inicializa el pin 34 del LED Azul como una salida
  //-------------------------------------------------------------------------------------------------------------------------
}
//-------------------------------------------------------------------------------------------------------------------------
void loop() {
  AvisoSistemaManual();
  //-------------------------------------------------------------------------------------------------------------------------
  VelocidadConJoystick();        //   Accion
  //-------------------------------------------------------------------------------------------------------------------------
  Joystick();                    //   Accion
  //-------------------------------------------------------------------------------------------------------------------------
  SistemaAutomatizadoTeclas();   //   Accion
  //-------------------------------------------------------------------------------------------------------------------------
}
//-------------------------------------------------------------------------------------------------------------------------
void SistemaAutomatizadoTeclas() {
  //-------------------------------------------------------------------------------------------------------------------------
  int Valor1 = digitalRead(Boton1) ;   //   Leemos los Valores de las teclas
  int Valor2 = digitalRead(Boton2) ;   //   Leemos los Valores de las teclas
  int Valor3 = digitalRead(Boton3) ;   //   Leemos los Valores de las teclas
  int Valor4 = digitalRead(Boton4) ;   //   Leemos los Valores de las teclas
  //-------------------------------------------------------------------------------------------------------------------------
  if (Valor1  < 1 ||  Valor2  < 1 ||  Valor3  < 1 ||  Valor4  < 1 )   //   Condicion para habilitar los motores, velocidad y acciones
    //-------------------------------------------------------------------------------------------------------------------------
  {
    //-------------------------------------------------------------------------------------------------------------------------
    digitalWrite(Habilitador, LOW);   // Habilita corriente a los Motores PAP
    //-------------------------------------------------------------------------------------------------------------------------
    VelocidadNormal();               // Selecciona la velocidad de los Motores PAP
    AvisoSistemaAutomatico();
    //-------------------------------------------------------------------------------------------------------------------------
    Tecla1Normal();    //   Accion
    Tecla2Normal();    //   Accion
    Tecla3Normal();    //   Accion
    Tecla4Normal();    //   Accion
    //-------------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
  else
    //-------------------------------------------------------------------------------------------------------------------------
  {
    //-------------------------------------------------------------------------------------------------------------------------
    digitalWrite(Habilitador, HIGH);         // Desabilita corriente a los Motores PAP
    //-------------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
```

```
}
//-------------------------------------------------------------------------------------------------------------------------------
void Tecla3Normal() {
  VelocidadNormal();                      // Selecciona la velocidad del motor PAP
  AvisoLCDVelocidad3();
  int Valor3 = digitalRead(Boton3) ;        // Leemos el Valor de la tecla
  if (Valor3 < 1 )  {                     // Si el valor es menor a 1
    delay(500);                           // Pone un tiempo de medio segundo antes de comenzar
    for (int i = 0; i <= 255; i++) {          // Cantidad de pusos
      MotorVerticalDerecha();                 // Accion de un motor PAP y giro expecifico
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 255; i++) {
      MotorVerticalIzquierda();
    }
  }
}
//-------------------------------------------------------------------------------------------------------------------------------
void Tecla2Normal() {
  VelocidadNormal();
  AvisoLCDVelocidad3();
  int Valor2 = digitalRead(Boton2) ;
  if (Valor2 < 1 ) {
    delay(500);
    for (int i = 0; i <= 130; i++) {
      MotorVerticalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 130; i++) {
      MotorVerticalIzquierda();
    }
  }
}
//-------------------------------------------------------------------------------------------------------------------------------
void Tecla1Normal() {
  VelocidadNormal();
  AvisoLCDVelocidad3();
  int Valor1 = digitalRead(Boton1) ;
  if (Valor1 < 1 )  {
    delay(500);
    for (int i = 0; i <= 140; i++) {
      MotorVerticalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 210; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 140; i++) {
      MotorVerticalDerecha();
    }
  }
}
//-------------------------------------------------------------------------------------------------------------------------------
void Tecla4Normal() {
  VelocidadNormal();
```

```
  AvisoLCDVelocidad3();
  int Valor4 = digitalRead(Boton4) ;
  if (Valor4 < 1 ) {
    delay(500);
    for (int i = 0; i <= 270; i++) {
      MotorVerticalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalDerecha();
    }
    delay(500);
    for (int i = 0; i <= 220; i++) {
      MotorHorizontalIzquierda();
    }
    delay(500);
    for (int i = 0; i <= 270; i++) {
      MotorVerticalDerecha();
    }
  }
}
//-----------------------------------------------------------------------------------------------------------------------
//   Movimientos de las 2 posiciones de cada Motor PAP
//-----------------------------------------------------------------------------------------------------------------------
void MotorHorizontalIzquierda() {
  digitalWrite(Direccion1, HIGH);
  for (uint32_t i = 0; i < PasosPorVueltas1 / Dividisor ; i++) {
    digitalWrite(Step1, HIGH);
    delay(PasosPeriodo1 / 0.65 );
    digitalWrite(Step1, LOW);
    delay(PasosPeriodo1 / 0.65 );
  }
}
//-----------------------------------------------------------------------------------------------------------------------
void MotorHorizontalDerecha() {
  digitalWrite(Direccion1, LOW);
  for (uint32_t i = 0; i < PasosPorVueltas1 / Dividisor ; i++) {
    digitalWrite(Step1, HIGH);
    delay(PasosPeriodo1 / 0.65  );
    digitalWrite(Step1, LOW);
    delay(PasosPeriodo1 / 0.65 );
  }
}
//-----------------------------------------------------------------------------------------------------------------------
void MotorVerticalIzquierda() {
  digitalWrite(Direccion2, HIGH);
  for (uint32_t i = 0; i < PasosPorVueltas2 / Dividisor ; i++) {
    digitalWrite(Step2, HIGH);
    delay(PasosPeriodo2 / 0.65 );
    digitalWrite(Step2, LOW);
    delay(PasosPeriodo2 / 0.65 );
  }
}
//-----------------------------------------------------------------------------------------------------------------------
void MotorVerticalDerecha() {
  digitalWrite(Direccion2, LOW);
  for (uint32_t i = 0; i < PasosPorVueltas2 / Dividisor ; i++) {
    digitalWrite(Step2, HIGH);
    delay(PasosPeriodo2 / 0.65  );
    digitalWrite(Step2, LOW);
    delay(PasosPeriodo2 / 0.65 );
  }
}
//-----------------------------------------------------------------------------------------------------------------------
//   Configuracion de las Distintas Velocidades del Driver A4988
//   segun la configuracion de las entradas M0, M1, M2
//   Tabla de la Verdad
//   M0   M1   M2   Pasos
//   0    0    0    Completo
//   1    0    0    1/2
//   0    1    0    1/4
//   1    1    0    1/8
//   1    1    1    1/16
//   Binario = 000 = paso = Completo
//   Binario = 100 = paso = 1/2
//   Binario = 010 = paso = 1/4
```

```
//   Binario = 110 = paso = 1/8
//   Binario = 111 = paso = 1/16
//   M0 =  Pin N' 6   =   Naranja
//   M1 =  Pin N' 7   =   Marron
//   M2 =  Pin N' 8   =   Lila
//-------------------------------------------------------------------------------------------------------------------------
void VelocidadMuyRapido() {
  digitalWrite(PasosM0, LOW);
  digitalWrite(PasosM1, LOW);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, HIGH);           // Enciende el LED Rojo
  digitalWrite(LedRapido, LOW);               // Apaga el LED Naranja
  digitalWrite(LedNormal, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedLento, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, LOW);             // Apaga el LED Azul
}
//-------------------------------------------------------------------------------------------------------------------------
void VelocidadRapido() {
  digitalWrite(PasosM0, HIGH);
  digitalWrite(PasosM1, LOW);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, LOW);            // Apaga el LED Rojo
  digitalWrite(LedRapido, HIGH);              // Enciende el LED Naranja
  digitalWrite(LedNormal, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedLento, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, LOW);             // Apaga el LED Azul
}
//-------------------------------------------------------------------------------------------------------------------------
void VelocidadNormal() {
  digitalWrite(PasosM0, LOW);
  digitalWrite(PasosM1, HIGH);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, LOW);            // Apaga el LED Rojo
  digitalWrite(LedRapido, LOW);               // Apaga el LED Naranja
  digitalWrite(LedNormal, HIGH);              // Enciende el LED Amarillo
  digitalWrite(LedLento, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, LOW);             // Apaga el LED Azul
}
//-------------------------------------------------------------------------------------------------------------------------
void VelocidadLento() {
  digitalWrite(PasosM0, HIGH);
  digitalWrite(PasosM1, HIGH);
  digitalWrite(PasosM2, LOW);
  digitalWrite(LedMuyRapido, LOW);            // Apaga el LED Rojo
  digitalWrite(LedRapido, LOW);               // Apaga el LED Naranja
  digitalWrite(LedNormal, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedLento, HIGH);              // Enciende el LED Amarillo
  digitalWrite(LedMuyLento, LOW);             // Apaga el LED Azul
}
//-------------------------------------------------------------------------------------------------------------------------
void VelocidadMuyLento() {
  digitalWrite(PasosM0, HIGH);
  digitalWrite(PasosM1, HIGH);
  digitalWrite(PasosM2, HIGH);
  digitalWrite(LedMuyRapido, LOW);            // Apaga el LED Rojo
  digitalWrite(LedRapido, LOW);               // Apaga el LED Naranja
  digitalWrite(LedNormal, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedLento, LOW);               // Apaga el LED Amarillo
  digitalWrite(LedMuyLento, HIGH);            // Enciende el LED Azul
}
//-------------------------------------------------------------------------------------------------------------------------
void Joystick() {
  //-----------------------------------------------------------------------------------------------------------------------
  PosicionX = analogRead(CoordenadaX);          //   Lee posicion de X
  PosicionY = analogRead(CoordenadaY);          //   Lee posicion de Y
  //-----------------------------------------------------------------------------------------------------------------------
  if (PosicionX  > 950 ||  PosicionX  < 60 ||  PosicionY  > 950 ||  PosicionY  < 60 )
    //---------------------------------------------------------------------------------------------------------------------
  {
    //-------------------------------------------------------------------------------------------------------------------
    AvisoSistemaManual();
    digitalWrite(Habilitador, LOW);   // Habilita corriente a los Motores PAP
    //---------------------------------------------------------------------------------------------------------------------
  } else {
    digitalWrite(Habilitador, HIGH);   // Desabilita corriente a los Motores PAP
    //---------------------------------------------------------------------------------------------------------------------
```

```
  }
  if (PosicionX > 950) {
    MotorHorizontalIzquierda();
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (PosicionX < 60) {
    MotorHorizontalDerecha();
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (PosicionY < 60) {
    MotorVerticalIzquierda();
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (PosicionY > 950) {
    MotorVerticalDerecha();
  }
}
//-----------------------------------------------------------------------------------------------------------------------
void VelocidadConJoystick() {
  //-------------------------------------------------------------------------------------------------------------------------
  EstadoDelBotonR3 = digitalRead(BotonR3);
  //-------------------------------------------------------------------------------------------------------------------------
  if (EstadoDelBotonR3 != UltimoEstadoDelBotonR3)  // Compara el estado del botón con su estado anterior
  {
    //-----------------------------------------------------------------------------------------------------------------------
    if (EstadoDelBotonR3 == HIGH)                 // si el estado ha cambiado, incrementar el contador
    {
      //---------------------------------------------------------------------------------------------------------------------
      ContadorPulsosBotonR3++;   // si el estado actual es ALTO entonces el botón incrementa
      //---------------------------------------------------------------------------------------------------------------------
      //---------------------------------------------------------------------------------------------------------------------
    }
    //-----------------------------------------------------------------------------------------------------------------------
  }
  //-------------------------------------------------------------------------------------------------------------------------
  UltimoEstadoDelBotonR3 = EstadoDelBotonR3;   // Guarda el estado actual como el último estado,para el próximo bucle
  //-------------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 == 1) {
    AvisoLCDVelocidad1();
    VelocidadMuyRapido();                       // Velocidad Muy Rapido
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 == 2) {
    AvisoLCDVelocidad2();
    VelocidadRapido();                          // Velocidad Rapido
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 == 3) {
    AvisoLCDVelocidad3();
    VelocidadNormal();                          // Velocidad Normal
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 == 4) {
    AvisoLCDVelocidad4();
    VelocidadLento();                           // Velocidad Lento
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 == 5) {
    AvisoLCDVelocidad5();
    VelocidadMuyLento();
  }
  //-------------------------------------------------------------------------------------------------------------------------
  if (ContadorPulsosBotonR3 > 5) {
    AvisoLCDVelocidad1();
    ContadorPulsosBotonR3 = 1;
  }
  //-------------------------------------------------------------------------------------------------------------------------
}
//-----------------------------------------------------------------------------------------------------------------------
void AvisoLCDVelocidad1() {
  lcd.setCursor(0, 3);         //  lcd.setCursor(Columna, Fila);
  lcd.print("Velocidad ");     //  Texto
  lcd.setCursor(10, 3);
  lcd.print("Muy Rapido");
}
```

```
//-------------------------------------------------------------------------------------------------------------------------
void AvisoLCDVelocidad2() {
  lcd.setCursor(0, 3);
  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);
  lcd.print("Rapido    ");
}
void AvisoLCDVelocidad3() {
  lcd.setCursor(0, 3);
  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);
  lcd.print("Normal    ");
}
//-------------------------------------------------------------------------------------------------------------------------
void AvisoLCDVelocidad4() {
  lcd.setCursor(0, 3);
  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);
  lcd.print("Lento     ");
}
//-------------------------------------------------------------------------------------------------------------------------
void AvisoLCDVelocidad5() {
  lcd.setCursor(0, 3);
  lcd.print("Velocidad ");
  lcd.setCursor(10, 3);
  lcd.print("Muy Lento ");
}
//-------------------------------------------------------------------------------------------------------------------------
void AvisoSistemaManual() {
  lcd.setCursor(0, 2);
  lcd.print("Sistema Manual      ");
}
//-------------------------------------------------------------------------------------------------------------------------
void AvisoSistemaAutomatico() {
  lcd.setCursor(0, 2);
  lcd.print("Sistema Automatico  ");
}
//-------------------------------------------------------------------------------------------------------------------------

void AvisoMotorHorizontalIzquierda() {
  lcd.setCursor(0, 0);
  lcd.print("Motor1 Giro --->>>  ");
}
void AvisoMotorHorizontalDerecha() {
  lcd.setCursor(0, 0);
  lcd.print("Motor1 Giro <<<---  ");
}
void AvisoMotorVerticalIzquierda() {
  lcd.setCursor(0, 1);
  lcd.print("Motor2 Giro --->>>  ");
}
void AvisoMotorVerticalDerecha() {
  lcd.setCursor(0, 1);
  lcd.print("Motor2 Giro <<<---  ");
}
void AvisoMotor1SinActividad() {
  lcd.setCursor(0, 0);
  lcd.print("Motor1 Detenido     ");
}
void AvisoMotor2SinActividad() {
  lcd.setCursor(0, 1);
  lcd.print("Motor2 Detenido     ");
}
```