SUBSECRETARÍA DE EDUCACIÓN SUPERIOR
DIRECCIÓN GENERAL DE EDUCACIÓN SUPERIOR TECNOLÓGICA
INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

INGENIERIA ELECTRICA

INFORME TECNICO DE RESIDENCIA PROFECIONAL

DISEÑO Y CONSTRUCCIÓN DE UN ELEVADOR DE DOS NIVELES UTILIZANDO EL SISTEMA MAKEBLOCK.

ASESOR

Dr. RAFAEL MOTA GRAJALES.

RESIDENTE

Bautista Ocampo Wilberto

PERIODO DE REALIZACION:

ENERO-JUNIO 2014

Indice General

Indice de Figuras

Índice de Gráficas y Tablas

1. Introducción

La retícula en el área de ingeniería considerando el desarrollo de prácticas de laboratorio, en donde el diseño de nuevos sistemas de automatización, control e instrumentación es fuertemente requerido de un conjunto de conocimientos técnicos, ordenados científicamente, que permiten diseñar y crear bienes y servicios que facilitan la adaptación al medio ambiente y satisfacer tanto las necesidades esenciales como los deseos de las personas. La actividad tecnológica influye en el progreso social y económico.

La necesidad de lograr una mejor calidad en el desarrollo del modelo de competencias en la ingeniería hace que se realicen sistemas de automatización basados para resolver problemas de grado tiempo real para una inspección que se puede definir como el proceso mediante el cual se determina si el producto es desviado de un conjunto de especificaciones dadas. La tabla 1.1 **muestra** en la actualidad los productos y procesos valoran las características que aseguran la calidad del producto a comercializar.

Tabla 1. 1 Tabla clasificación de calidad

Este es un contexto, la industria de la automatización en introducción de nuevos sistemas de elevación utilizando nuevas técnicas basadas en Prototipado mecatrónicos con fundamentos reales con forme a nuestra vida cotidiana tal cual es la implementación de sistemas de automatización con calidad y precisión con la solvencia de sistemas de desarrollo didácticos con herramientas basadas en sistemas abiertos

En la mayoría de los sectores de hoy en día, los márgenes de beneficio son estrechos y continuarán estrechándose aún más. Incluso en segmentos en los que los márgenes son relativamente sustanciales, la competencia y la subcontratación global obligan a la reducción de costos .Desde siempre, el alto costo de la ingeniería ha contribuido de forma tan elocuente a la merma de los márgenes de beneficio, que se han realizado numerosos intentos de reducir el tiempo de proceso o el costo de las actividades de ingeniería. La mayoría de estos enfoques han sido soluciones puntuales, que pueden ser de gran importancia por derecho propio pero no se pueden aplicar de forma general. La automatización del diseño, sin embargo, destaca como un medio eficaz de reducir radicalmente los costos de una gama bien definida y probada de actividades de ingeniería. Esto es especialmente cierto donde las necesidades empresariales exigen un presupuesto rápido y preciso, una ingeniería coherente y, lo más importante, un plazo mínimo para la entrega del producto acabado.

Mediante la implementación de la automatización del diseño, los fabricantes de ingeniería bajo pedido pueden completar días de ingeniería personalizada en sólo unos minutos. La automatización del diseño también acelera y simplifica la creación de modelos, dibujos y datos de fabricación de Solid Works (**Fig. 1.1),** es decir prácticamente cualquier requisito del proceso de ventas personalizado.

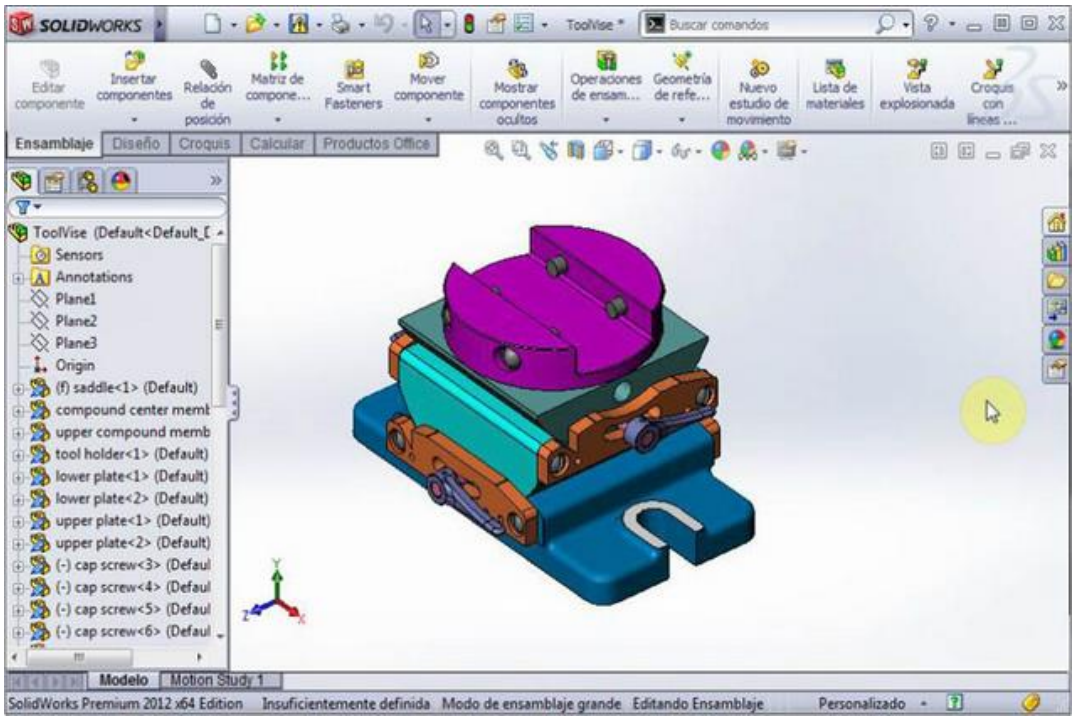


Fig. 1.1 Dibujo de diseño en Solid Works

Su carácter abrumadoramente comercial hace que esté más orientada a satisfacer los deseos de los más prósperos, sin embargo, la tecnología también puede ser usada para proteger el medio ambiente y evitar que las crecientes necesidades provoquen un agotamiento o degradación de los recursos materiales y energéticos del planeta. Introducción en el mercado de un nuevo bien o servicio, el cual los consumidores no están aún familiarizados. Introducción de un nuevo método de producción o metodología organizativa. Creación de una nueva fuente de suministro de materia prima o productos semielaborados Apertura de un nuevo mercado en un país. Implantación de una nueva estructura en un mercado.

Una gran variedad de empresas de distintos sectores buscan formas de ahorrar tiempo y reducir costos en el proceso de ingeniería. De ahí que diversos proveedores denominen “automatización del diseño” a distintas operaciones o utilidades. Desafortunadamente, la mayoría no son mecanismos útiles para la automatización porque no pueden gestionar todo el trabajo. En el caso específico de

las empresas que ofrecen productos de ingeniería bajo pedido, la automatización del diseño es un sistema que captura y más o menos aplica de forma automática esa actividad de ingeniería a variantes de productos para producir diseños acabados en un mínimo de tiempo.

1.1 Estado del Arte

En lo particular las automatizaciones son presentados en diferentes modelos y diseños, esto ha originado investigaciones centradas en sistemas Automáticos de elevación considerando defectos con base en la forma de operar. Utilizando técnicas en diseños nuevos de instrumentación y automatización de robustos sistemas automatizados de control por medio de interfaz hombre máquina, implementando open hardware .Algunas de las áreas de interés, donde se han desarrollado estas investigaciones son: (Open Hardware crowfunding' & 'Kickstarter, 2011), **(**Raspberry-pi modelo-B 2012), Automotriz (Tornero & Armesto, 2012), Metalúrgica (Luque, Ruiz, & Hogert, 2011, ElMasry & Cubero, 2012),Gadgeteer (2013), Makeblock (2011).

Open Hardware, (2011). En la actualidad conocemos y utilizamos, fomentado en algunos casos por la administración y por las grandes corporaciones. Su implantación y penetración se llevó a cabo de una manera muy gradual, en un periodo de tiempo muy largo. Sin embargo, los open hardware se ha abierto camino mucho más rápido y, aunque todavía queda mucho por hacer, se puede decir que el 2011 y 2012 han sido los años clave de los open hardware; gran parte de ello se lo debemos a los italianos creadores de Arduino. El 'crowfunding' y 'Kickstarter 'también han contribuido con el modelo básico en gran medida al auge del open hardware (Figura 1.1.1). Al igual que existen muchas distribuciones de Linux.

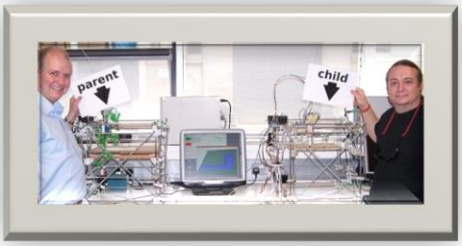


Fig. 1.1.1 Modelo crowfunding & Kickstarter

Raspberry-pi modelo-B (2012). Este es una placa realizada con presupuesto de bajo costo diseñada en Reino Unido para la enseñanza de sistemas de computación en las escuelas (Figura 1.1.2). Basado en una CPU de la familia ARM11. Desde su lanzamiento en el 2012 ha recibido cientos de miles de peticiones en escuelas y centros de educación mecatrónicos. Soporta como sistema operativo distribuciones basadas en Linux. Se pueden encontrar muchos proyectos basados en esta placa debido al elevado número de usuarios interesados en ella.

Fig. 1.1.2 Raspberry-pi modelo-B

Tornero (2012) desarrolló un sistema para la detección de defectos en carrocerías de vehículos. Proyectó un patrón de luz estructurada, mediante lámparas fluorescentes sobre la carrocería de automóviles, con una capa inicial de pintura; posteriormente, mediante un sistema de visión compuesto por una cámara, previamente calibrada a una resolución de 0.896 mm/px, situada sobre el automóvil, capturó una serie de imágenes, realizando mediciones del ancho de las líneas en pixeles, reflejadas sobre los paneles de la carrocería; de esta manera determinó las variaciones contenidas en la imagen; detectando alteraciones de 0.2, 0.3 y 0.6 mm, con una eficiencia del 100 % (Tornero & Armesto, 2012).

Luque (2011) detectó deformaciones en tuberías, utilizando una técnica óptica denominada triangulación activa con luz estructurada. Proyectó un patrón laser en forma de anillo y analizó las imágenes registradas; realizó una procesamiento de la imagen y obtuvo el contorno del círculo proyectado, a partir de este contorno y el centro, calculó la magnitud del radio; comparando el ángulo de muestreo, respecto a la longitud del radio, para detectar variaciones, le fue posible detectar validad, fisuras y corrosión en ductos (Luque, Ruiz, & Hogert, 2011)

Gadgeteer (2013). Se desarrollaron prototipos Gadgeteer (Fig. 1.1.3). Está basada también en el entorno .NET en los que utilizaron Visual C# para la creación de programas. Su componentes son; una placa principal Y una serie de módulos distintos que se conectan a la placa principal. Basado también en la arquitectura ARM de la familia 7 o 9. Están creando proyectos de plataforma. El uso de estas placas también está facilitando que muchos de los proyectos que se crean estén en desarrollo.

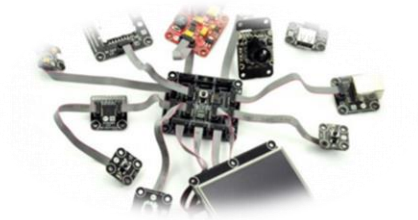


Fig. 1.1.3 Gadgeteer (2013).

Makeblock 2011. Desarrollo un sistema de construcción de elementos basado en Arduino**,** con elementos tales como actuadores, motores, interruptores y hasta controladoras bluetooth. Todos estos módulos electrónicos son de código abierto. En el Starter Robot Kit V2.0 nos encontramos un conjunto de piezas que nos va a permitir la construcción de algunos modelos básicos para familiarizarnos con Makeblock (Fig 1.1.4). En el manual de referencias técnicas, que utiliza el mismo sistema visual de Lego para guiarnos en el montaje, nos encontramos dos modelos diferentes. Con una buena selección de piezas el Starter Robot Kit V2.0 es una buena plataforma de) construcción de robots móviles, plataformas de diseño y

programación. Los motores, engranajes y ruedas de aluminio, sensor de ultrasonidos**,** placa de control de Makeblock basada en Arduino Leonardo y las vigas o elementos rígidos que van a permitirnos la construcción del chasis de los modelos. Así tenemos un receptor de infrarrojos para recibir las órdenes del mando a distancia también incluido, cables de conexión de los motores y los sensores a la placa Arduino, y por supuesto tornillos, tuercas y demás elementos para el montaje y las herramientas necesarias para tal fin. Con una buena selección de piezas.

Fig. 1.1.4 Robot Electronic Kit Pro

1.2 Contexto del trabajo de investigación

La inspección de defectos en la forma de operar es un objeto de estudio en los modernos procesos de control de calidad en la industria autómata, debido a la necesidad de contar cada vez con métodos y técnicas que aseguren la alta calidad sin defectos del producto. En los sistemas de elevación industrial es un área de interés para este tipo de análisis, debido a la necesidad de generar productos que soporten estrictos puntos de calidad.

La Real Academia de las Ciencias Físicas y Exactas define la automática como el conjunto de métodos y procedimientos para la substitución del operario en tareas físicas y mentales previamente programadas. De esta definición original se desprende la definición de la automatización como la aplicación de la automática al control de procesos industriales.

En un sistema de mando como proceso, se entiende aquella parte del sistema en que, a partir de la entrada de material, energía e información, se genera una transformación sujeta a perturbaciones del entorno, que da lugar a la salida de material en forma de producto. Los procesos industriales se conocen como procesos continuos, procesos discretos y procesos batch. Los procesos continuos se caracterizan por la salida del proceso en forma de flujo continuo de material, como por ejemplo la purificación de agua o la generación de electricidad (Fig. 1.2).

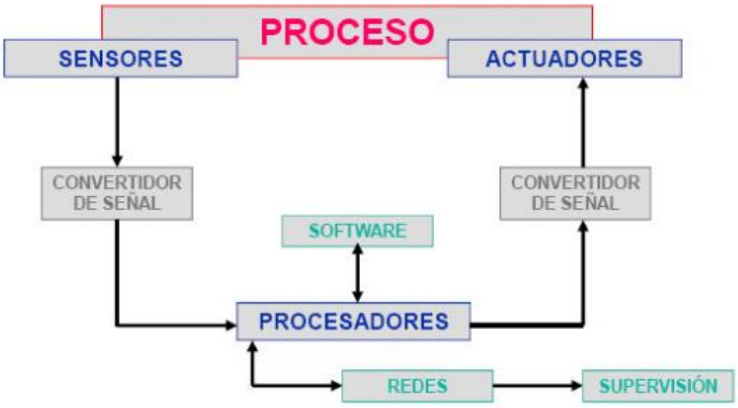


Fig. 1.2. Sistema de Mando

Los procesos discretos contemplan la salida del proceso en forma de unidades o número finito de piezas, siendo el ejemplo más relevante la fabricación de automóviles. Finalmente, los procesos batch son aquellos en los que la salida del proceso se lleva a cabo en forma de cantidades o lotes de material, como por ejemplo la fabricación de productos farmacéuticos o la producción de cerveza.

El concepto de proceso está claramente relacionado con los conceptos de productos, programas, así como con la planificación de plantas. La estructura organizativa de la empresa debe contar con una clara relación entre estos conceptos, y para ello el ciclo de diseño está basado en la idea de ingeniería concurrente en la que diversos equipos desarrollan de forma coordinada cada uno de los diseños En concreto es relevante centrarse en qué se va a producir, como y cuando se fabricarán los productos, qué cantidad de producto debe fabricarse, así como especificar el tiempo empleado y el lugar en que se llevarán a cabo dichas operaciones. Estas cuestiones sobrepasan los límites del presente libro (Tompkins et. al., 2006), (Velasco, 2007).

1.2.1 Principio ultrasónico

Un sensor ultrasónico de distancia mide empleando un transductor que emite “paquetes “de ultrasonido que contienen una serie de ondas sonoras intermitentes. El paquete se emite forma cónica, se rebota o refleja en la superficie u objetivo y se recibe el regreso en un transductor. El tiempo requerido por el sonido para ir y volver se mide y se convierte a unidades de distancia. Varios factores afectan la medición con ultrasonido: la naturaleza de la superficie, el ángulo del cono y la distancia de sensor objetivo.

Las condiciones ambientales como son temperatura, humedad relativa, gases, vapores y la presión también afectan. Los sensores están diseñados con ajustes ya sean manuales o automáticos para compensar la mayoría de estas condiciones cambiantes. El patrón del haz producido por el sensor se expresa en número de grados que el haz se separa de la línea central del sensor. El haz se desparrama con un patrón cónico a partir del transductor y aunque el haz ultrasónico continúa desparramándose, el área de detección del sensor empieza a acortarse respecto al rango de operación publicado. El área de censado se ve afectada por el número de pulsos enviado por el sensor y por el nivel de sensibilidad. A nivel alto de pulsos y sensibilidad, mayor superficie que a niveles bajos (Fig. 1.2.1).

Superficie

La superficie-objetivo ideal es dura y lisa. Esta reflejará una mayor cantidad de señal que una superficie suave y rugosa. Un eco débil, resultante de un objetivo pequeño o suave reduce la distancia de operación del sensor y disminuye su exactitud.

Distancia

Mientras más corta sea la distancia al objetivo, será más fuerte el eco. De modo que si la distancia aumenta requerimos mejores características Reflejantes en la superficie objetivo.

Tamaño

Un objetivo grande tendrá mayor superficie para rebotar la señal que Un objetivo pequeño. La porción de superficie reconocida como “objetivo” es normalmente la más cercana al sensor.

Ángulo

La inclinación de la superficie objetivo afecta la efectividad. La parte que sea perpendicular (90°) al sensor es la rebota el eco. Si la superficie total está inclinada fuertemente la señal será rebotada alejándola del sensor y no detectará eco.

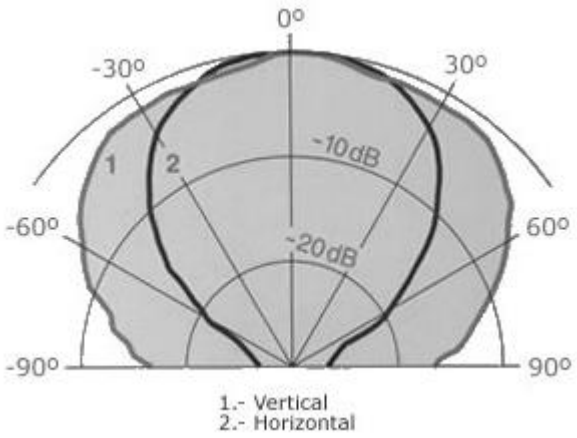


Fig. 1.2.1. Patrón de Haz

1.2.2 Modelo Físico del ultrasónico

El sensor ultrasónico emite cíclicamente un impulso acústico de alta frecuencia y corta duración. Este impulso se propaga a la velocidad del sonido por el aire. Al encontrar un objeto, es reflejado y vuelve como eco al sensor ultrasónico. Este último calcula internamente la distancia hacia el objeto, basado en el tiempo transcurrido entre la emisión de la señal acústica y la recepción de la señal de eco (Figura 1.2.2).

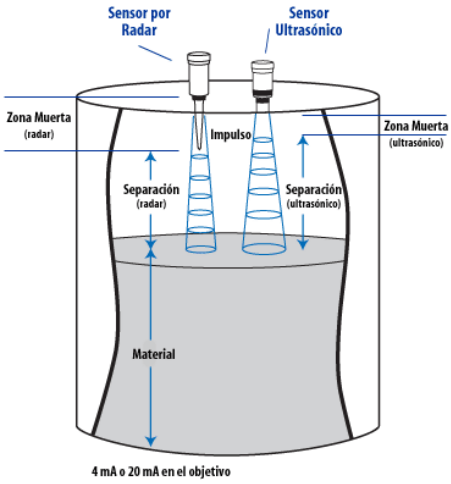


Figura 1.2.2 Aplicación Modelo ultrasónico

Ultrasonido Terapéutico

Es una forma de energía que proviene de las vibraciones mecánicas. Esta energía se propaga en forma de ondas de compresión longitudinal y necesita de un medio elástico para ser transmitido. Se entiende por tratamiento ultrasónico el empleo de vibraciones sonoras en el espectro no audible, con fines terapéuticos. Se documenta su empleo a partir de los años treinta. En los años cincuenta se generaliza su uso como una nueva forma de diatermia. A partir de los años sesenta, se introduce la forma pulsante. Se emplea como agente de diatermia selectiva, antiinflamatorio y analgésico. Al ultrasonido como terapia, también se le conoce como una terapia de las superficies límites.

1.2.3 Características Físicas

Los sensores ultrasónicos micro Sonic (Fig. 1.2.3). Permiten medir distancias entre 20 mm y 10 m, pudiendo indicar el valor medido con una precisión de milímetro, gracias a la medición del tiempo de recorrido. Algunos sensores pueden inclusive obtener una precisión de la medición de distancia de 0,025 mm. Los sensores con una zona ciega de sólo 20 mm y con un haz acústico extremadamente delgado abren en la actualidad un abanico de aplicaciones completamente nuevas: Las mediciones de estado de llenado en pocillos de placas microtiter y tubos de ensayo como también el escaneado de botellas pequeñas en la industria de los embalajes pueden llevarse a cabo sin problemas. Incurro alambres finos son reconocidos con seguridad. Prácticamente todos los materiales que reflejan el sonido son detectados, independientemente de su color. Aún materiales transparentes o láminas delgadas no presentan problemas para los sensores ultrasónicos. El área de censado se ve afectada por el número de pulsos enviado por el sensor y por el nivel de sensibilidad .A nivel alto de pulsos y sensibilidad, mayor superficie que a niveles bajos en (fig. 1.2.3.1) se ilustran estos principios.

Fig. 1.2.3 Sensor ultrasónico

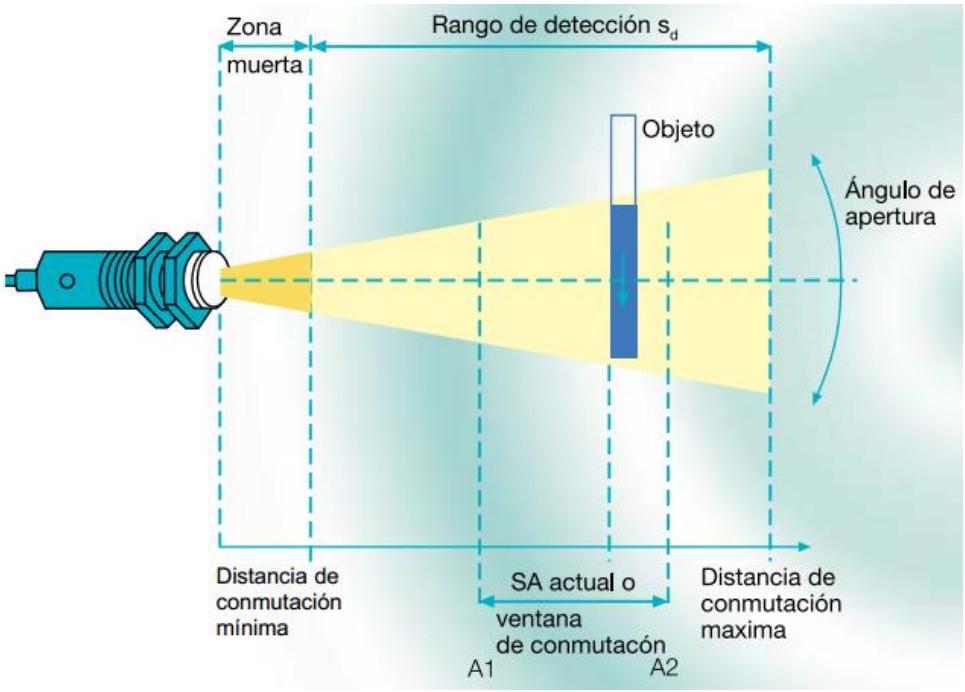


Fig. 1.2.3.1 Efecto en operación sensor ultrasónico

1.3. Justificación

La industria está en un proceso de integración hacia adelante donde pasa convertirse en proveedor de servicios tecnológicos, incluyendo compromiso asociados a la mejora de los flujos de trabajo, rendimiento de los activos e incremento de productividad en los procesos integrales. No tiene la necesidad de hacer frente a una cuantiosa inversión inicial, tiene costos fijos o predecibles para toda la tecnología y para todo el periodo del acuerdo, tiene accesos, se beneficia del conocimiento y experiencia técnica. Permite estar siempre perfectamente actualizado y al día con los últimos desarrollos en tecnología. En la construcción de los modelos open hardware.

Este proyecto se llevó a cabo con la finalidad de desarrollar sistemas más adecuados, basados en prácticas de laboratorio .Que requieren un fuerte impulso dinámico en tiempo real para mejorar la eficiencia de la ingeniería en el plan o modelo de competencias de educación que es ejercido en los institutos tecnológicos. Con la implementación de prototipos de entrenamiento ampliaremos la dinámica de aprendizaje en sistemas de ensamblado de piezas de aluminio extruido (Make Block), utilizando nuevas técnicas de aprendizaje.

La calidad de los materiales y de la construcción de las distintas piezas es realmente notable. Las piezas de aluminio han sido pensadas para necesitar el menor número de tornillería y tuercas posible. Así, un único tornillo fija una pieza de aluminio, ya que esta última cuenta con roscas, evitando tener que agregar una tuerca para fijarlo. Las vigas tienen una muesca en la que podemos atornillar sin necesidad de tuercas para la fijación de los elementos, y en sus extremos también nos encontramos dos agujeros con rosca. La fijación de elementos que necesitemos quitarse realiza con dispositivos de presión realizados en plástico, lo que hace que por un lado sea muy sencillo y rápido el montaje.

Al estar atornillado el conjunto y sujeto con materiales de calidad, utilizamos las piezas justas y necesarias para asegurar la robustez del modelo. El modelo que se puede montar en alrededor de hora y media, probablemente reduciéndose el tiempo si ya tenemos experiencia con Makeblock. Como es bastante sólido y transmite

calidad. En la siguiente parte nos centraremos en la programación del robot y las posibilidades de desarrollo de Makeblock. Dispone de muchas piezas fáciles de ensamblar entre ellas, gracias a su diseño modular y agujeros equidistantes que incluso son compatibles con LEGO.

Además se trabajó con la plataforma de Arduino Uno, ya que brinda una completa flexibilidad en el uso, tanto a nivel físico, como a nivel software. Dicho entorno nos facilita el uso de sensores, comunicación inalámbrica, pues cuenta con un lenguaje de programación open source bastante amigable, sencillo de comprender, y lo mejor de este sistema, cuenta con micro controladores AVR, los cuales superan por mucho a los comúnmente conocidos PLC de micro chip, tanto en memoria, como en puertos analógicos, serial, etc.

La programación es de código abierto, por lo que puede hacer propias o mejorar las existentes. Todos los módulos electrónicos son compatibles con Arduino, y con la placa base especial podrá conectar los sensores y los motores de una forma sumamente sencilla gracias a su código de colores y conexiones RJ11.

Código Abierto

```
sensor HC-SR04 Ping distancia:
VCC para arduino 5v
GND para arduino GND
Echo a Arduino pin 7
Trig para Arduino pin 8


# Define echo Pin 7 / / Echo Pin
# define trig Pin 8 / / Disparador Pin
# define led Pin 13 / / Onboard LED

Into maximum Range = 200; / / Alkane Maximo necessary
Inti minimumRange = 0; / / Rango mínimo necesario
De largo duración, distancia; / / Duración utilizado para calcular la
distancia

Void setup () {
 Serial. Begin (9600);
 Pin Mode (trig Pin, OUTPUT);
 Pin Mode (echo Pin, INPUT);
```

```
 Pin Mode (led Pin, OUTPUT); / / indicador de uso de LED (si es
necesario)
}

Void loop () {
 / * El siguiente ciclo trigPin / echoPin se utiliza para determinar la
Distancia del objeto más cercano haciendo rebotar ondas de sonido fuera
de ella. * /
 Digital Write (trig Pin, LOW);
 Delay Microseconds (2);

 Digital Write (trig Pin, HIGH);
 Delay Microseconds (10);

 Digital Write (trig Pin, LOW);
 Duration = Pulse In (echo Pin, HIGH);

 / / Calcular la distancia (en cm) sobre la base de la velocidad del
sonido.
 Distancia = duración/58.2;

 Si (distancia> = maximumRange | | distancia <= minimumRange) {
  / * Enviar un número negativo para ordenador y LED gira
Para indicar "fuera de rango" * /
 Serial.println ("-1");
 digital Write (led Pin, HIGH);
 }
 Más {
  / * Enviar la distancia a la computadora usando el protocolo de
serie, y
Apagará el LED para indicar una lectura exitosa. * /
 Serial.println (distancia);
 DigitalWrite (led Pin, LOW);
 }

 / / Delay 50 ms antes de la próxima lectura.
 De retardo (50);
}
```

1.4 Objetivo

Diseñar y construir un elevador de dos niveles didáctico utilizando el sistema mecánico abierto (open mechanic´s). Implementando el control del mismo con sistemas digitales, micros controladores y autómatas programables.

1.5 Objetivo específicos

Diseñar un sistema de elevación a diferentes niveles utilizando la herramienta makeblock de aluminio extruido para la enseñanza de ingeniería en sistema de automatización e instrumentación.

Diseñar un sistema del tipo HMI (Interfaz hombre maquina) de bajo costo en el monitoreo y control de sistemas mecatrónicos.

Diseñar y construir un sistema para adquisición de posición en el estudio de niveles exactos.

Diseño de un ascensor autómata de dos niveles utilizando herramienta makeblock.

2. Marco Teórico

2.1 Makeblock

Es una plataforma de construcción de código abierto para Realizar diferentes estructuras de aluminio extruido basadas en blocks y ensambles prefabricados para convertir las ideas en prototipos de utilidad como prototipos de entrenamiento. No importa el estilo o diseño que se tenga en mente, makeblock ofrece diferentes piezas mecánicas para poder realizar las multitudes de estructuras a escala milimétrica tales como son vigas de aluminio extruido que viene con agujeros lista para ensamblar y formar diseños, placas, conectores, motores y los soportes (Fig. 2.1). Makeblock es una plataforma integrada, las partes mecánicas son convenientes por que se pueden conectar unas con otras para que estas puedan ser integradas con diferentes módulos electrónicos de sistema abiertos

Fig. 2.1 Set completo Makeblock

Siempre era mal momento para la toma de las ideas en la realidad, sobre todo cuando se relaciona con el hardware. Así, muchas de las ideas son siempre permanecer en la mente de la gente, nunca tiene la oportunidad a hacerse realidad. Makeblock tiene como objetivo ayudar a las personas tomar sus ideas en realidad, hacer cosas interesantes con facilidad y rapidez, reducir el dolor de hacer que las cosas reales, y disfrutar de todo el proceso. Somos una empresa de nueva creación con sede en Shenzhen, China. Actual mente la empresa Makeblock consta con 10

elementos que forman parte de un quipo estructurado y fundamentado que se enorgullecen del producto makeblock Creemos en la ideología de la tecnología de código abierto, todos los productos diseñados por Makeblock es opensource, a partir de piezas mecánicas para módulos electrónicos, todo el archivo de origen está abierto Sabemos que no hay limitación en la imaginación de las personas, que necesitan miles de diferentes piezas y módulos para hacer realidad sus ideas, así que vamos a seguir desarrollando más y más piezas mecánicas nuevas y módulos electrónicos, tratando de satisfacer las necesidades de las personas de la creación. También alentamos a la comunidad de código abierto para ayudar a desarrollar nuevos componentes y módulos Arduino (Fig. 2.1.1 a y b).

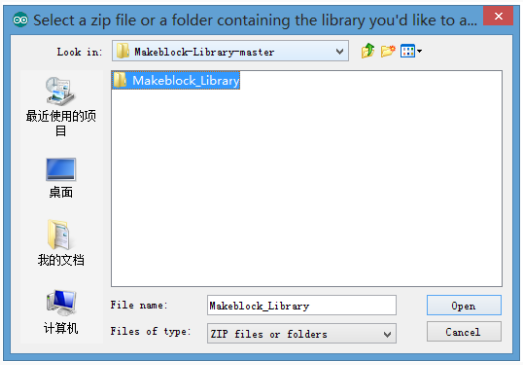


Fig. 2.1.1 (a). Código Abierto

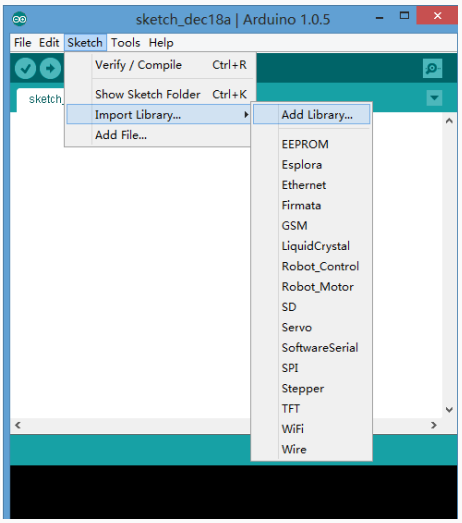


Figura 2.1.1 (b). Código Abierto

2.2 Arduino Uno

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Se creó para artistas, diseñadores, aficionados y cualquiera interesado en crear entornos u objetos interactivos (Italia 2003). Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces, motores y otros actuadores.

Software ampliable y de código abierto- El software Arduino está publicado bajo una licencia libre y preparado para ser ampliado por programadores experimentados. El lenguaje puede ampliarse a través de librerías de C++, y si se está interesado en profundizar en los detalles técnicos, se puede dar el salto a la programación en el lenguaje AVR C en el que está basado. De igual modo se puede añadir directamente código en AVR C en tus programas si así lo deseas.

Hardware ampliable y de Código abierto - Arduino está basado en los micro controladores ATMEGA168, ATMEGA328 yATMEGA1280. Los planos de los módulos están publicados bajo licencia Creativa Commons, por lo que diseñadores de circuitos con experiencia pueden hacer su propia versión del módulo, ampliándolo u optimizándolo. Incluso usuarios relativamente inexpertos pueden construir la versión para placa de desarrollo para entender cómo funciona

El micro controlador en la placa Arduino (Fig. 2.2). Se programa mediante el lenguaje de programación Arduino(basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing).

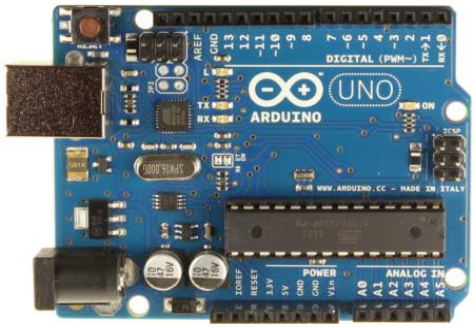


Fig. 2.2. Arduino Uno

Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un ordenador, si bien tiene la posibilidad de hacerlo y comunicar con diferentes tipos de software (Flash, Processing, Max MSP).El software de Arduino consiste en un entorno de desarrollo (IDE) y las bibliotecas de núcleo.

El IDE está escrito en java y basado en el entorno de desarrollo processing las bibliotecas están escritas en C y C++ y compilado usando AVR-GCC y librería AVR. El código fuente para Arduino está alojado en GITHUB.

2.3 Conceptos Básicos

El Arduino Uno R3 es una placa electrónica basada en el ATmega328 (Tarjeta de desarrollo Arduino Uno R3). Cuenta con 14 entradas y/o salidas digitales, pines de los cuales 6 son utilizados como salidas PWM, 6 entradas analógicas, un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, una cabecera de ICSP, y un botón de reinicio.

Contiene lo necesario para apoyar al micro controlador, sólo tiene que conectarlo a un ordenador con un cable USB o con un adaptador AC-DC. El Arduino Uno R3 difiere de todas las placas anteriores en que no utilizan el chip controlador USB FTDI a serie. Por el contrario, cuenta con la Atmega16U2 programado como un convertidor de USB a serie. Dicho de otra manera este nuevo cambio le permite tener más memoria y más velocidad en procesamientos de datos.

Pines de Alimentación

VIN.- El voltaje de entrada a la placa Arduino cuando se utiliza una fuente de alimentación externa (a diferencia de 5 voltios de la conexión USB o de otra fuente de alimentación regulada). Se suministra la alimentación a través de este pin.

5V.- La fuente de alimentación regulada utilizada para alimentar el micro controlador y otros componentes en el tablero. Esto puede provenir ya sea de VIN a través de un regulador a bordo, o se suministra a través de USB o de otra fuente de 5V regulada.

3V3.- Un suministro voltios 3,3 generada por el regulador de a bordo. Consumo de corriente máxima es de 50 mA.

GND...- Pines de tierra.

Entrada y salida

Serie.- 0 (RX) y 1 (TX). Se utiliza para recibir (RX) y transmisión (TX) datos serie TTL. Estos se encuentran conectados a los pines correspondientes de la ATmega8U2 USB-ha-chip de serie TTL.

Interrupciones externas 2 y 3.- Estas patillas se configuran para desencadenar una

interrupción en un valor bajo, un borde ascendente o descendente, o un cambio en el valor. Con la función attach Interrupt ().

PWM.- 3, 5, 6, 9, 10 y 11. Proporcionar 8-bit de salida PWM con la función analog Write ().

SPI.- 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Estos pines apoyo a la comunicación SPI utilizando la biblioteca de SPI.

LED.- 13. Hay un built-in LED conectado al pin digital 13. Cuando el pin es de alto valor, el LED está encendido, cuando el pasador es bajo, es apagado.

El Arduino Uno tiene 6 entradas analógicas, etiquetados A0 a A5, cada una de las cuales proporcionan 10 bits de resolución. Por defecto miden desde 0 a 5 voltios, aunque es posible cambiar el extremo superior de su rango con el pasador y el AREF analog Reference (). Además, algunos pines tienen funciones especializadas:

IST.- A4 o A5 y pin SDA o pin SCL. Apoyar la comunicación TWI utilizando la librería Wire.

2.4 Lenguaje de Programación

El lenguaje Arduino está basado en C/C++ y soporta todas las construcciones de C estándar y algunas funcionalidades de C++.Vincula la librería AVR LIBC y permite el uso de todas sus funciones. Es una implementación de Wiring, una plataforma de computación física parecida, que a su vez se basa en Processing, un entorno de programación multimedia. Arduino está basado en C y soporta todas las funciones del estándar C y algunas de C++. A continuación se muestra un resumen en todas las estructuras del lenguaje Arduino.

Estructuras control
if *(comparador si-entonces)*
if...else *(comparador si...sino)*
for *(bouclé con contour)*
switch case *(comparator multiple)*

Sintaxis

- ; (punto y coma)
- {} (llaves)
- // (comentarios en una línea)
- /* */ (comentarios en múltiples líneas)

Operadores Aritméticos

- = (asignación)
- + (suma)
- - (resta)
- * (multiplicación)
- / (división)
- % (resto)

Operadores Comparativos

- == (igual a)
- != (distinto de)
- < (menor que)
- > (mayor que)
- <= (menor o igual que)

>= (mayor o igual

Operadores

Booleanos
&& (y)
|| (o)
! (negación)
Operadores de Composición
++ (incrementa)
-- (decremento)
+= (composición suma)
-= (composición resta)
*= (composición multiplicación)
/= (composición división)

Variables

Constantes
HIGH | LOW
INPUT | OUTPUT
true | false

Constantes

Numéricas
Tipos de Datos
bolean *(booleano)*
chart *(carácter)*
byte
int *(entero)*
unsigned int *(entero sin signo)*

2.5 Entorno de Desarrollo

El entorno de Desarrollo Arduino está constituido por un editor de texto para escribir el código, un área de mensajes, una consola de texto, una barra de herramientas con botones para las funciones comunes, y una serie de menús. Permite la conexión con el hardware de Arduino para cargar los programas y comunicarse con ellos. Arduino utiliza para escribir el software lo que denomina "sketch (*programa*)" (Fig. 2.5). Estos programas son escritos en el editor de texto. Existe la posibilidad de cortar/pegar y buscar/remplazar texto. En el área de mensajes se muestra información mientras se cargan los programas y también muestra errores. La consola muestra el texto de salida para el entorno de Arduino incluyendo los mensajes de error completos y otras informaciones. La barra de herramientas permite verificar el proceso de carga, creación, apertura y guardado de programas, y la monitorización serie:

Verifica/Compile
Chequea el código en busca de errores.

Stop
Finaliza la monitorización serie y oculta otros botones

New
Crea un nuevo *sketch*.

Open
abrirá en la ventana actual.

Save
Salva el programa *sketch*.

Upload to I/O Board
Compila el código y lo graba en placa Arduino.

Serial Monitor
Inicializa la monitorización serie

.

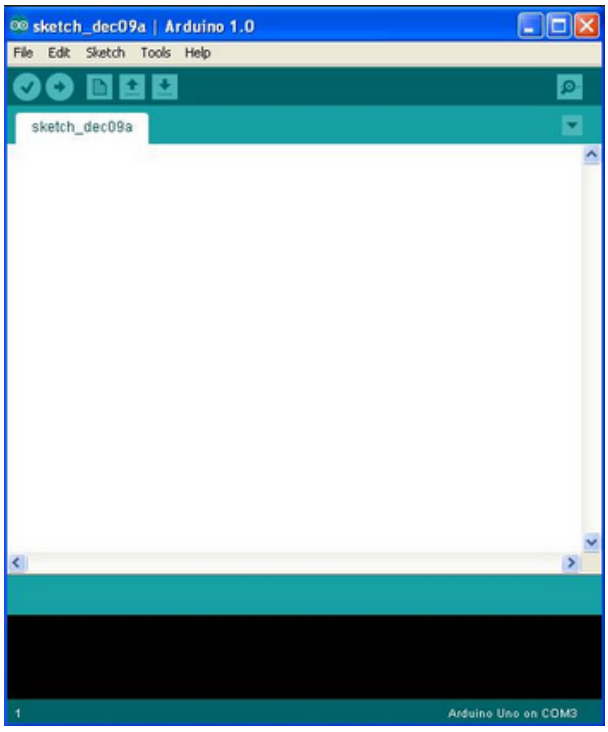


Fig. 2.5 Entorno Arduino

El entorno de Arduino incluye el concepto de "sketchbook": que es el lugar estándar para el almacenamiento de sus programas (o "sketch"). Los "sketches" dentro de su "sketchbook" pueden abrirse desde el menú File > Sketchbook o desde el botón dela barra de herramientas Open. La primera vez que arranque el software Arduino, se creará un directorio para su "sketchbook". Puede visualizar o cambiar su localización dentro de "sketchbook locación"

2.6. Librerías y Comunicación

Las librerías proporcionan funcionalidad extra para la utilización en "sketches", por ejemplo para trabajar con hardware o manipular datos. Para utilizar una librería en un "sketch", seleccione el menú Sketch > Importa Library. Esto insertará una o más sentencias incluye al principio del "sketch" y compilará la librería con su "sketch". Debido a que las librerías se vuelcan a la placa junto con su "sketch", incrementan la ocupación del espacio disponible. Si un "sketch" no precisa de una librería, simplemente borra su sentencia # incluye en la parte inicial del código. Algunas librerías están incluidas en el software Arduino (Fig. 2.6), otras pueden ser descargadas desde una gran variedad de fuentes.

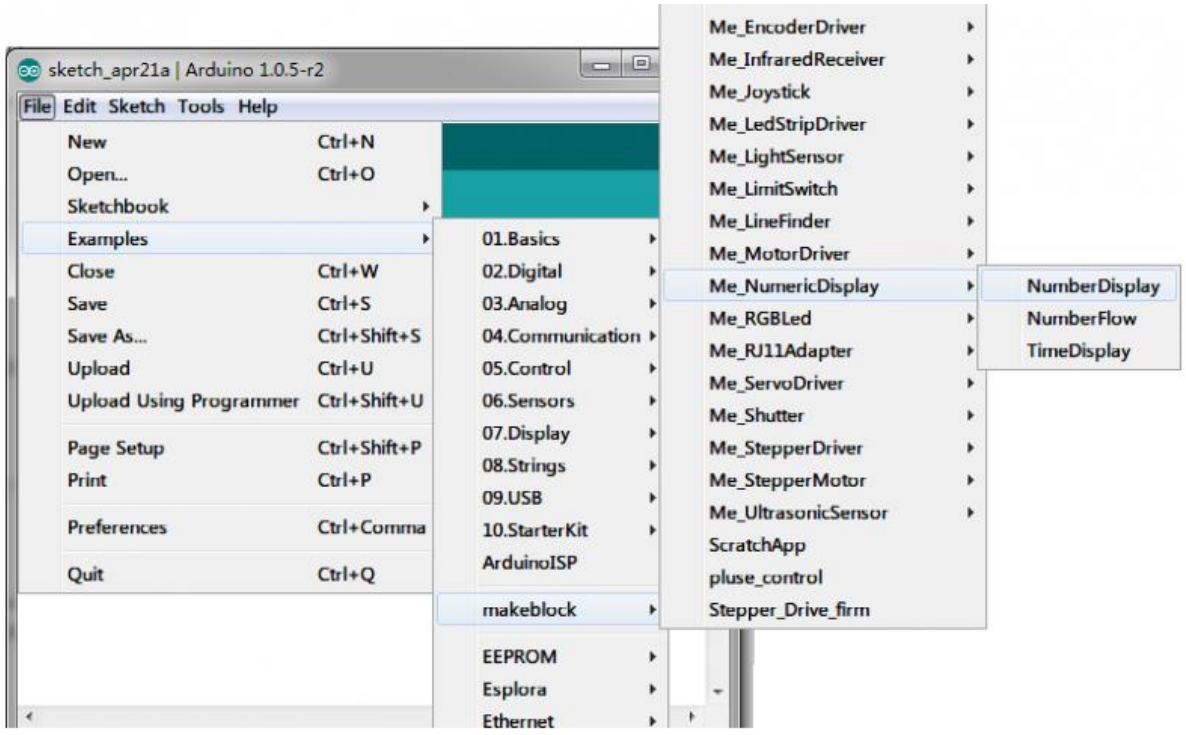


Fig. 2.6 software Arduino

2.6.1 Librerías

EEPROM - Para leer y escribir en memorias "permanentes".

Ethernet - Para conectar a internet usando el *Ethernet Shield**?***.

Firmata - Para comunicarse con aplicaciones en la computadora usando un protocolo estándar *Serial*.

Liquida Cristal - Para controlar Displays de cristal líquido (*LCD*)

Servo - Para controlar *servomotores*

Software Serial - Para la comunicación serial de cualquier pin digital.

Stepper - Para controlar motores paso a paso (*Stepper motor*)

Wire - Interfaz de dos cables, o *Two Wire Interface (TWI/I2C)*, para enviar y recibir datos a través de una red de dispositivos y sensores (Fig. 2.6.1).

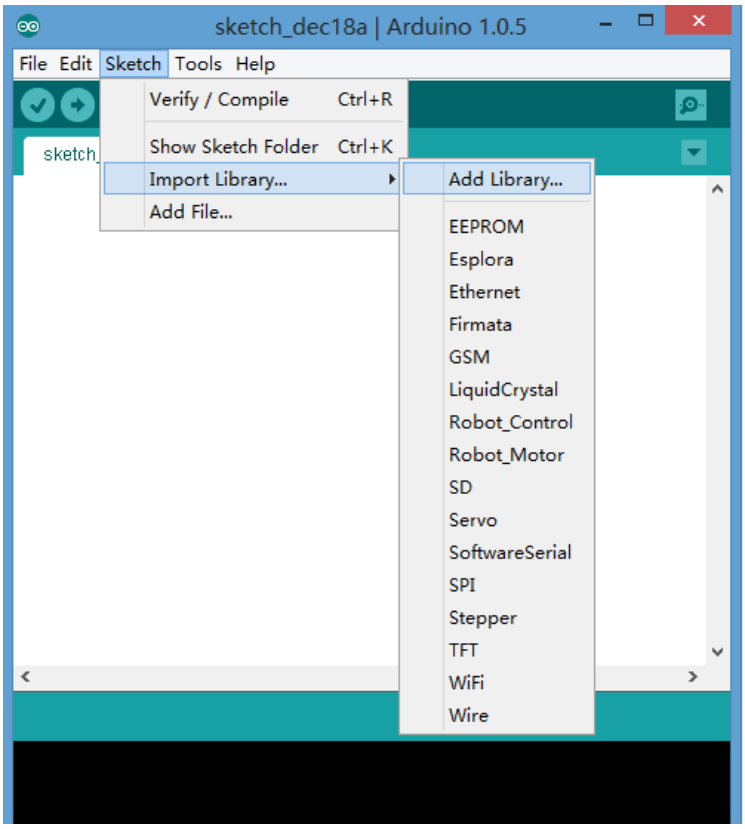


Fig. 2.6.1 Librería

2.6.2 Comunicación Serial

Se utiliza para la comunicación entre la placa Arduino y un ordenador u otros dispositivos. Todas las placas Arduino tienen al menos un puerto serie (también conocido como UART o USART): Serial. Se comunica a través de los pines digitales 0 (RX) y 1 (TX), así como con el ordenador mediante USB. Por lo tanto, se utiliza estas funciones, no puede usar los pines 0 y 1 como entrada o salida digital. Puede utilizar el monitor del puerto serie incorporado en el entorno Arduino para comunicar con la placa Arduino. Haz clic en el botón del monitor de puerto serie en la barra de herramientas y selecciona la misma velocidad en baudios utilizada en la llamada a begin ().La placa Arduino Mega tiene tres puertos adicionales de serie: Serial1 en los pines 19 (RX) y 18 (TX), Serial2 en los pines 17 (RX) y 16 (TX), Serial3 en los pines 15 (RX) y 14 (TX). Para utilizar estos pines para comunicarse con el ordenador personal, necesita de un adaptador USB adicional a serie, ya que no están conectados al adaptador USB-Serie de la placa Arduino. Para usarlos para comunicarse con un dispositivo serie externo TTL, conecta el pin TX al pin RX del dispositivo, el puerto serie RS232, que operan a +/- 12V y esto puede dañar la placa Arduino.)RX al pin TX del dispositivo, y el GND del Arduino a masa del dispositivo. (No conectar estos pines directamente a un puerto serie RS232, que operan a +/- 12V y esto puede dañar la placa Arduino.

2.7 Modulo Ultrasónico HCSR.

El sensor SRF04 funciona emitiendo impulsos de ultrasonidos inaudibles para el oído humano. Los impulsos emitidos viajan a la velocidad del sonido hasta alcanzar un objeto, entonces el sonido es reflejado y captado de nuevo por el receptor de ultrasonidos. Lo que hace el controlador incorporado es emitir una ráfaga de impulsos ya continuación empieza a contar el tiempo que tarda en llegar el eco. Este tiempo se traduce en un pulso de eco de anchura proporcional a la distancia a la que se encuentra el objeto. Desde un punto de vista práctico, lo que hay que hacer es mandar una señal de arranque en el pin 3 del SRF04 y después leer la anchura del impulso que nos proporciona en el pin 2. El pulso de disparo tiene que tener una anchura mínima de 10 uS. Después leemos el pulso de salida de Eco y medimos su longitud que es proporcional al eco recibido. En caso de que no se produzca ningún eco, porque no se encuentra un objeto, el pulso de eco tiene una longitud aproximada de 36 ms. Hay que dejar un retardo de 10 ms desde que se hace una lectura hasta que se realiza la siguiente, con el fin de que el circuito se estabilice.

Ultrasónico van HC module - SR04 (Fig. 2.7). Ofrece 2cm - 400cm sin contacto función de medición, la precisión que van puede llegar a 3 mm. Los módulos incluyen transmisores ultrasónicos, el receptor y el circuito de control. El principio básico de los trabajos: (1) El uso de IO desencadenante de al menos 10us señal de alto nivel, (2) El módulo envía automáticamente ocho 40 kHz y detectar si hay un la señal de vuelta del pulso. (3) Si la parte posterior de la señal, a través de alto nivel, el tiempo de salida de alta duración IO es el tiempo desde el envío de ultrasonidos de volver. Pruebe la distancia = (tiempo de alto nivel × velocidad del sonido (los 340M / S) / 2.

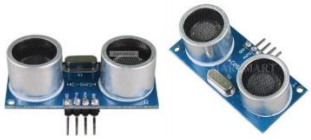


Figura 2.7 Sensor SR04

2.7.1 Especificaciones Modulo Ultrasónico

- Voltaje de funcionamiento: 5V (DC)
- Corriente estática: <2 mA
- Trabajo actual: 15mA
- Frecuencia de trabajo: 40KHz
- Señal de salida: frecuencia de la señal eléctrica, 5V de alto nivel, bajo nivel de 0V
- Ángulo Eficaz: <15 º
- Distancia de detección: 2 cm - 450 cm
- Resolución: 0,3 cm
- Medición de ángulo: 30 º
- Disparo de la señal de entrada: TTL pulso 10µs
- Echo señal de salida: señal PWL de TTL
- Angulo de cobertura: 2cm – 500cm
- Resolución : 0.3 cm
- 4 pines :VCC,Trig,Echo,GND

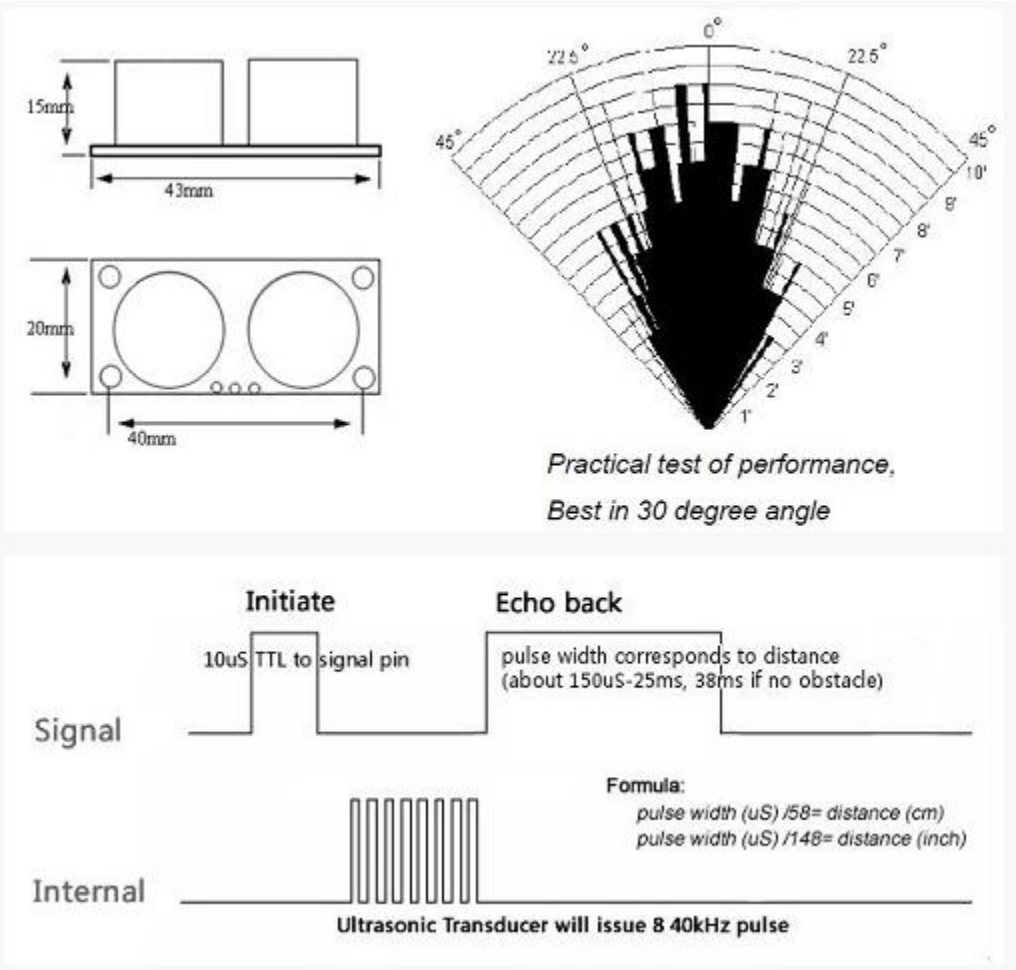
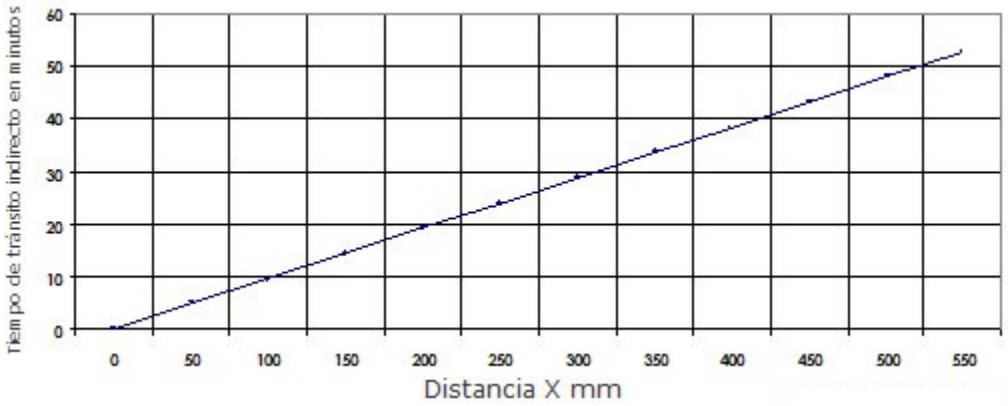


Fig. 2.7.1 prueba en Angulo de 30 grados

Grafica 2.7.1 (a). Método para determinar la distancia con arreglo indirecto



Grafica 2.7.1 (b). Método Grafico de distancia en contraposición con el tiempo

2.7.2. Clasificación de la calidad por medio de la velocidad de onda.

Tabla 2.7.2 Tabla clasificación de calidad.

Velocidad de la onda longitudinal m/seg	**Condición**
Más de 4570De 3050 a 4570 De 3050 a 3650 De 2130 a 3050 Menos de 2130	ExcelenteBuena Regular a dudosa Pobre Muy pobre
Evaluación la calidad mediante la velocidad de pulso	
Velocidad de pulso m/seg	**Condición**
Más de 3000De 2500 a 3000 Menos de 2130	BuenaRegular Pobre
Velocidad mínima de pulso en estructuras típicas.	
Tipo de obra	**Velocidad mínima de pulso para su aceptación m/seg**
Selecciones T de hormigón reforzadoUnidades de anclaje de hormigón reforzado Marcos de edificios de hormigón reforzado Losas de entre piso	45704360 4110 4720

Modo. 1 Señal de activación y eco independientes

Este modo utiliza pines independientes para la señal de inicio de la medición y para retorno del eco, siendo el modo más sencillo de utilizar este modo, simplemente deberá dejar sin conectar el pin de modo

Modo2.Pin único para la señal de activación y eco.

Este modo utiliza un único pin para las señales de activación y eco, y está diseñado para reducir el número de pines en el micro controladores. Para utilizar este modo, conecte el pin de modo al pin de tierra de 0v. La señal de eco aparecerá en el mismo pin que la señal de activación. Dispone de ese tiempo para cambiar el pin del disparador y convertirlo en una entrada para preparar el código de medición de pulsos. El comando PULSIN integrado en la mayor parte de los controladores del mercado lo hace automáticamente.

2.7.3 Código Serial para distancia con módulo HC-RS04.

```
#include <Ultrasonic.h>
Ultra Sonic (9,8); // (Trig PIN, Echo PIN)
Void setup () {
  Serial. Begin (9600);
}
Void loop ()
{
  Serial. Print (ultrasonic. Ranging (CM)); // CM or INC
  Serial.println (" cm”);
  Delay (100);
}
// Ultrasonic - Library for HR-SC04 Ultrasonic Ranging Module.
#include <Ultrasonic.h>
#include <LiquidCrystal.h>
Liquid Crystal LCD (12, 11, 5, 4, 3, 2); // LCD Arduino library
Ultrasónico (9,8); // (Trig PIN, Echo PIN)
Void setup () {
Lcd. Begin (16, 2);
}
Void loop ()
{
  Lcd. Clear ();
  lcd.setCursor (0, 0);
  Lcd. Print (ultrasonic. Ranging (CM)); // CM or INC
  Lcd. Print ("cm");
  Delay (100);
```

```
}
// Ultrasonic - Library for HR-SC04 Ultrasonic Ranging Module.
// more info at www.ardublog.com
#include <Ultrasonic.h>
Ultrasonic (9,8); // (Trig PIN, Echo PIN)
Void setup () {
  Serial. Begin (9600);
}
Void loop ()
{
  Serial. Print (ultrasonic. Timing ());
  Serial.println (" ms”); // milliseconds
  Delay (100);
}
// Ultrasonic - Library for HR-SC04 Ultrasonic Ranging Module.
#include <Ultrasonic.h>
Ultrasonic ultraleft (9,8);   // (Trig PIN, Echo PIN)
Ultrasonic ultraright (6,7);  // (Trig PIN, Echo PIN)
Void setup () {
  Serial. Begin (9600);
}
Void loop ()
{
  Serial. Print ("Left: ");
  Serial. Print (ultraleft.Ranging (CM)); // CM or INC
  Serial. Print (" cm     “);
  Delay (50);
```

```
  Serial. Print ("Right: ");
  Serial. Print (ultraright.Ranging (CM)); // CM or INC
  Serial.println (" cm”);
  delay(50);
}
```

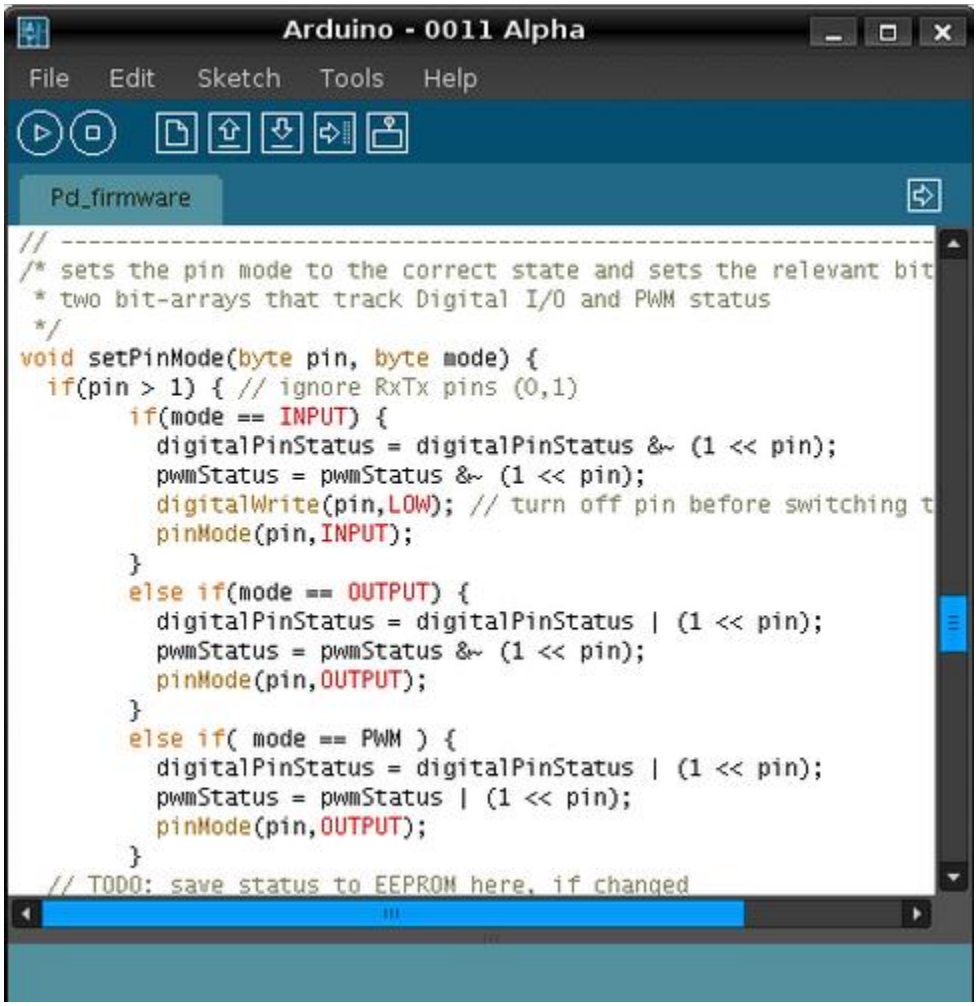


Fig. 2.7.3 código en plataforma Arduino

Las aplicaciones del sensor srf04.

Son múltiples, pero sobre todas ellas, destaca su utilización como detector de obstáculos en robots con navegación autónoma, es decir, en aquellos robots que se mueven encontrando el camino a seguir y sorteando obstáculos. En los robots de pequeño tamaño, es suficiente con un solo detector, ya que su cono de detección de unos 30 grados es suficiente para cubrir el frontal del robot. En las plataformas de mayor tamaño, son necesarias varias unidades para cubrir de una forma segura todo el perímetro. Para un robot de unos 30 cm es necesario un mínimo de 2 unidades, para cubrir solo el frontal. Si queremos cubrir todo el perímetro de avance, es necesario de 3 a 5 unidades para el mismo tamaño. Una posibilidad es la de montar el sensor en un servo y mover este 180 grados a la vez que se efectúan diversas mediciones a modo de radar.

3. Metodología

3.1 Desarrollo

Para poder comprender la complejidad, y los distintos tipos de soluciones del sistema hemos desarrollado dos prototipos utilizando hardware libre

- Elevador de dos niveles automático
- Silla móvil con trasmisión de datos vía blue tooth

El diseño de los prototipos se realizó tomando como base métodos de open hardware, teniendo en cuenta que se busca la mejor manera de resolver la problemática que tienen los sistemas automáticos alámbricos para resolver las decisiones tomadas por el usuario .En este proyecto trabajaremos métodos inalámbricos sensor izados y de trasmisión vía blue tooth..

El método que se optó por utilizar es el método de open hardware muestra las siguientes ventajas:

- Son sistemas distribuidos lineales
- Son sistemas tolerantes a fallos
- Adaptabilidad
- Establecen relaciones no lineales entre datos

La forma más práctica de implementar sistemas de open hardware es utilizando un computador convencional, puesto permite visualizar y modificar fácilmente diversas características de programación. A pesar de ser un sistema de procesamiento secuencial el computador puede ser utilizado para implementar un código nuevo diferente al que ya está en función, ya que se puede programar para realizar cualquier tipo de proceso algorítmico, incluyendo la simulación de un proceso en paralelo

En este trabajo se realizó un código programado en c++, y posteo mente fue exportado a la plataforma Arduino, para ser implementado en el sistema en tiempo real controlando los movimientos y disposiciones del elevador.

3.2 Diagrama a Bloques del Funcionamiento de Elevador

En la **figura 3.2.** Se observa el diagrama a bloques, donde observamos los dispositivos que se encuentran conectados a nuestra placa Arduino, la cual tiene conectado a sus entradas los sensores ultrasónicos .A la salidas, se puede observar que solo se encuentra conectado el driver que controla el motor del elevador, debido a que este driver (L293D) es un puente H, y necesita 2 señales digitales para poder utilizarlo.

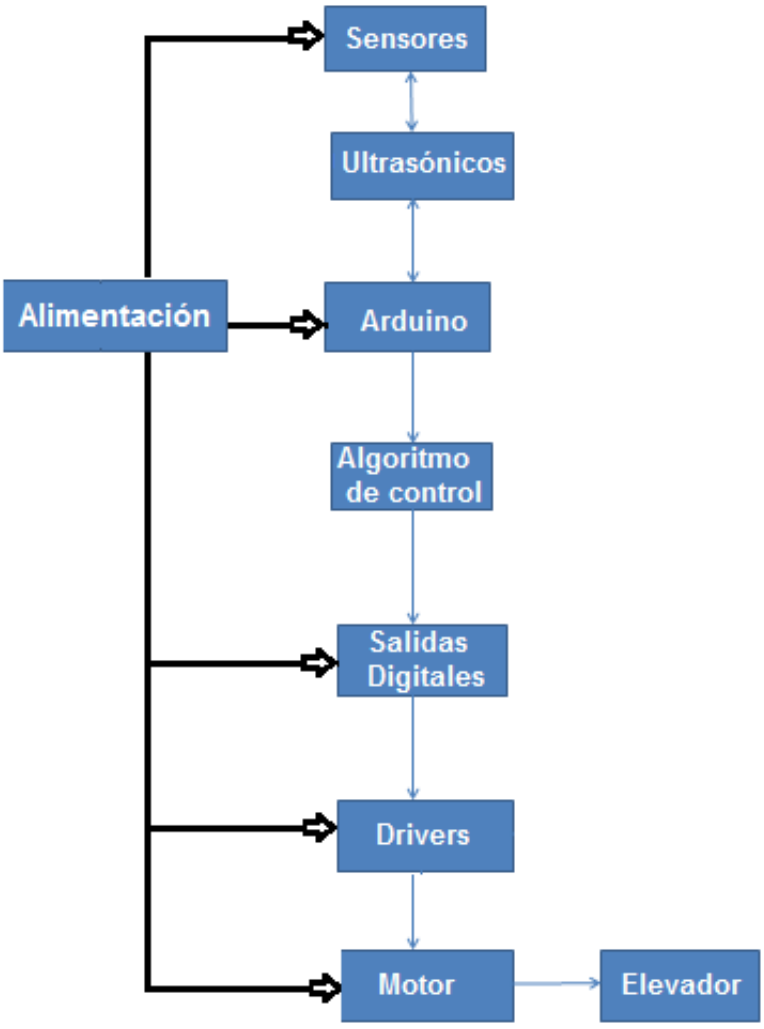


Fig. 3.2 Diagrama a bloques

3.3 Análisis de Algoritmos

Antes de comenzar a hacer la programación, es necesario hacer un análisis y tomar una decisión que lógica utilizaremos para que el elevador resuelva las disposiciones. En el código abierto a continuación se explica como al principio se emite una señal del sensor y cuando el sensor recibe la señal de vuelta esta cuenta el tiempo que transcurre, dicho tiempo es el tiempo del sonido. Al tener el tiempo del sonido podemos calcular su distancia, en los comentarios del código se explica más a detalle que es lo que sucede en el programa. Después hacemos uso de la consola serial para imprimir la distancia que existe entre el objeto y el sensor. Y de esta forma es como posicionamos el motor.

Inicio código abierto

```
Int pingpong = 13; //Pin que emite los sonidos, Trig
Int entrada Pin = 12; //Pin que va a recibir de vuelta la onda
 Int led Rojo = 5; //LED rojo va en pin 5
Int led Verde = 7; //LED verde va en pin 7
 /*Con esta variable podremos encender el LED rojo cuando
La distancia sea demasiado corta, puede ser
Cambiada con solo cambiar el valor, ese valor está en cm
*/
Int zona Segura = 10;
 Void setup () {
   //Inicializamos los pines como entradas y salidas
  Pin Mode (pigpen, OUTPUT);
  Pin Mode (Led Raju, OUTPUT);
  Pin Mode (Led Verde, OUTPUT);
  Pin Mode (entrada Pin, INPUT);
    Serial. Begin (9600); //Inicializamos la comunicación serial
```

```
}
Void loop () {
    //Creamos 2 variables, una para la duración y otra para la distancia
  Long duración, distancia en Cm;
   /*
  Hacemos un pulso bajo-alto-bajo para encender el sensor. Al encender y apagar esperamos en microsegundos, de esta manera enviaremos nuestra primera onda
  */
  Digital Write (pingpong, LOW); // Envía un pulso bajo
  Delay Microseconds (2);       // Espera dos microsegundos
  Digital Write (pingPin, HIGH);// Envía un pulso alto
  Delay Microseconds (5);       // Espera 5 microsegundos
  Digital Write (pingPin, LOW); // Se queda en espera
   /*Obtenemos la duración de tiempo mientras el sensor este recibiendo la información
  */
  Duración = pulse (entrada Pin, HIGH);
    /*
  Convertimos la duración del tiempo a distancia La velocidad del sonido es de 340metros/segundo que es igual a 29 microsegundos por centímetro es por es que vamos a dividir la duración entre 29.  Después se divide entre 2 porque es el tiempo que viaja el sonido de ida y de vuelta, solo queremos un valor pero ambos son iguales, es por eso que solo dividimos entre 2
  */
  DistanciaEnCm = (duración/29)/2;
   //Imprimimos la distancia en consola
  Serial. Print (distancia en Cm);
  Serial. Print ("cm");
  Serial.println ();
   /*Prendemos los Leds, cuando la distancia es
```

```
Mayor a la zona segura se prende el LED verde y
Se apaga el rojo. Cuando la distancia es menor
A la zona segura se prende el LED rojo y se apaga el LED
Verde
*/
If (distanciaEnCm > zona Segura){
Digital Write (led Verde, HIGH);
  Digital Write (led Rojo, LOW);
     }
Else {
  Digital Write (led Verde, LOW);
  Digital Write (led Rojo, HIGH);
   }
  //Hacemos un delay antes de volver a censar
Delay (100);
}
```

3.4 Simulaciones y primera prueba

Una vez ya que comprendemos como programamos procedemos a calibrar la distancia para posicionar la plataforma del elevador entonces procedemos hacer simulación en marcha accionando el prototipo basándonos en el programa que ya subimos a la plataforma Arduino.

Para crear un nuevo código nos basamos en los cálculos de la velocidad del en que viaja las ondas de sonido que emite el sensor ultrasónico y ya censado procedemos hacer un recalculo basado en la formula= (tiempo de alto nivel × velocidad del sonido (los 340M / S) / 2.**Proceso:**

Paso 1 .inicio en primer lugar es necesario instalar el **Modulo Ultrasónico HCSR** en el equipo e instalar los controladores. El módulo en uno de los puertos USB de repuesto. Windows detectará automáticamente el dispositivo y si se les pregunta a los conductores, apuntan a la unidad de CD-ROM. Una vez instalados los controladores, Windows puede tener que reiniciar el sistema. Utilizaremos los pines digitales y lógicos (**Figura 3.4)**. Para obtener los pulsos digitales los cuales le llamaremos alto o 1 lógico, bajo o 0 lógico (High and Low) esto lo detallo a continuación en código de programación.

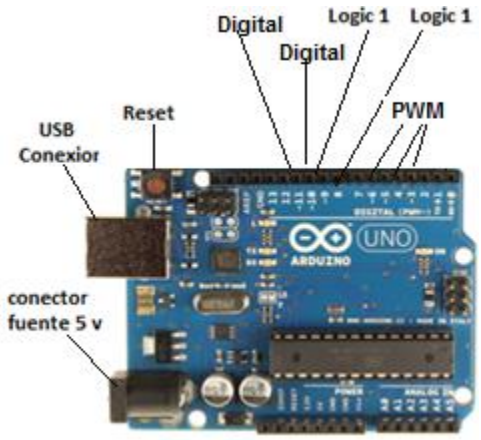


Fig. 3.4 pines de conexión en placa Arduino

3.4.1 Programación

```
Long distancia;
Long tiempo;
Void setup () {
  Serial. Begin (9600);
  Pin Mode (9, OUTPUT); /*activación del pin 9 como salida: para el pulso
ultrasónico*/
  Pin Mode (8, INPUT); /*activación del pin 8 como entrada: tiempo del rebote del
ultrasonido*/
}

Void loop (){
  DigitalWrite (9, LOW); /* Por cuestión de estabilización del sensor*/
  Delay Microseconds (5);
  DigitalWrite (9, HIGH); /* envío del pulso ultrasónico*/
  Delay Microseconds (10);
  Tiempo=pulsen (8, HIGH); /* Función para medir la longitud del pulso entrante.
Mide el tiempo que transcurrido entre el envío
  Del pulso ultrasónico y cuando el sensor recibe el rebote, es decir: desde que el
pin 12 empieza a recibir el rebote, HIGH, hasta que
  Deja de hacerlo, LOW, la longitud del pulso entrante*/
  Distancia= int (0.017*tiempo); /*fórmula para calcular la distancia obteniendo un
valor entero*/
  /*Monitorización en centímetros por el monitor serial*/
  Serial.println ("Distancia ");
  Serial.println (distancia);
  Serial.println (" cm");
  Delay (1000);
}
```

Paso 2.

MEDICION: para iniciar con la medición se debe conectar el pin trig a un nivel alto por 10 ms o más, esperar un pulso de retorno (**Fig. 3.4.2).** La duración del pulso de salida en echo es proporcional a la distancia medida según la fórmula; esto será igual a ((duración pulso de salida)*(1/29uS))/2.La velocidad de sonido es considerada 340 m/s.

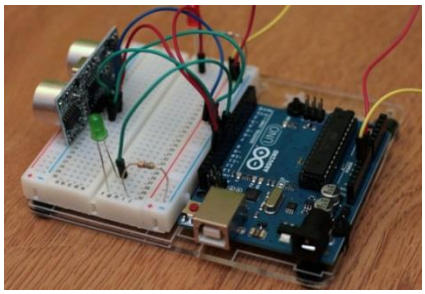


Fig. 3.4.2 Conexión pin trig

Paso 3.

En el código se explica paso a paso lo que hacemos. Nuestro programa constantemente estará midiendo la distancia que hay entre algún objeto y el sensor ultrasónico, al detectar que el objeto está a menos de 10 cm de distancia el LED de rojo o de alerta se va a encender, puedes modificar el código para establecer otras distancias (Fig. 3.4.3 a y b).

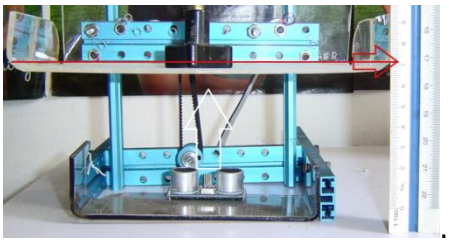


Fig. 3.4.3 (a) Objeto en detección

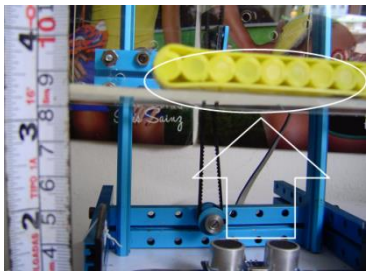


Fig. 3.4.3 (b) Objeto en detección

Paso 3.

En el código se explica como al principio se emite una señal del sensor y cuando el sensor recibe la señal de vuelta esta cuenta el tiempo que transcurre, dicho tiempo es el tiempo del sonido. Al tener el tiempo del sonido podemos calcular su distancia, en los comentarios del código se explica más a detalle que es lo que sucede en el programa. Después hacemos uso de la consola serial para imprimir la distancia que existe entre el objeto y el sensor.

```
  //Inicializamos los pines como entradas y salidas
  Pin Mode (pingPin, OUTPUT);
  Pin Mode (led Rojo, OUTPUT);
  Pin Mode (led Verde, OUTPUT);
  Pin Mode (entrada Pin, INPUT);
    Serial. Begin (9600); //Inicializamos la comunicación serial
}
Void loop () {
    //Creamos 2 variables, una para la duración y otra para la distancia
  Long duración, distanciaEnCm;
   /*
  Hacemos un pulso bajo-alto-bajo para encender el sensor. Al encender y apagar esperamos en microsegundos, de esta manera enviaremos nuestra primera onda
  */
  Digital Write (pingPin, LOW); // Envía un pulso bajo
  Delay Microseconds (2);       // Espera dos microsegundos
  Digital Write (pingPin, HIGH);// Envía un pulso alto
  Delay Microseconds (5);       // Espera 5 microsegundos
  Digital Write (pingPin, LOW); // Se queda en espera
   /*Obtenemos la duración de tiempo mientras el sensor este recibiendo la información
  */
  Duración = pulse (entrada Pin, HIGH);
    /*
  Convertimos la duración del tiempo a distancia La velocidad del sonido es de 340metros/segundo que es igual a 29 microsegundos por centímetro es por es que vamos a dividir la duración entre 29.  Después se divide entre 2 porque es el tiempo que viaja el sonido de ida y de vuelta, solo queremos un valor pero ambos son iguales, es por eso que solo dividimos entre 2
  */
```

```
DistanciaEnCm = (duración/29)/2;
 //Imprimimos la distancia en consola
Serial. Print (distancia en Cm);
Serial. Print ("cm");
Serial.println ();
 /*Prendemos los Leds, cuando la distancia es
Mayor a la zona segura se prende el LED verde y
Se apaga el rojo. Cuando la distancia es menor
a la zona segura se prende el LED rojo y se apaga el LED
Verde
*/
If (distanciaEnCm > zona Segura){
Digital Write (led Verde, HIGH);
  Digital Write (led Rojo, LOW);
     }
Else {
  Digital Write (led Verde, LOW);
  Digital Write (led Rojo, HIGH);
   }
  //Hacemos un delay antes de volver a censar
Delay (100);
}
```

Paso 4.

En este paso continuamos con el programa que cargamos en plataforma Arduino pero en este caso quitaremos el led verde y el led rojo para utilizar los pulsos de salida lógicos .Y en esta etapa también incrementamos un motor dc que será

acoplado a un puente h que hice un diseño casero en placa fenólica (Fig. 3.4.4).Estos arreglos sean realizado con técnicas de apoyo y desarrollo profesional utilizando las herramientas y equipo necesario tales como son Simuladores ARES E ISIS Profesional. Para poder realizar un material didáctico. Ya que tenemos terminada nuestra placa ensamblaremos los componentes electrónicos mencionados anteriormente (LM293D) este es el integrado que mejor desempeño rinde en sistemas de automatización por el consumo de corriente y el uso rudo al a aplicación que tenemos en acción

Fig. 3.4.4 Diseño en placa puente H

3.5 El Sistema Makeblock

Iniciamos con **el sistema makeblock** damos las primeras vistas de lo que es el trabajo de ensamblar piezas de aluminio extruido .Es llamado también set de inicio (Fig. 3.5) para principiantes en el mundo de la robótica este set nos servirá para referenciar las piezas por el contorno y las dimensiones en milímetros es un poco difícil de diseñar partes muy pequeñas en base a lo que es el sistema de ferretería que también viene en el estuche de inicio para sujetar las estructuras así como los tornillos prisioneros y las micro poleas que estas son necesarias para el sistema mecánico

Fig. 3.5 set de Inicio Makeblock

3.5.1 Vigas de Aluminio

Las Vigas con agujeros de aluminio extruido las utilizamos para formar una estructura firme y resistente en la cual nos servirá de base para el inicio de la construcción del elevador. Estas partes son las más usadas para formación de plataformas, tiene una longitud de 144 y 192 mm (Fig. 3.5.1).

Fig. 3.5.1 vigas medianas

3.5.2 Estructura Base de Elevador.

En este paso juntamos las vigas para formar la base del elevador que le llamaremos pierna izquierda y pierna derecha hacemos una forma tipo “T” en Angulo de90 grados.

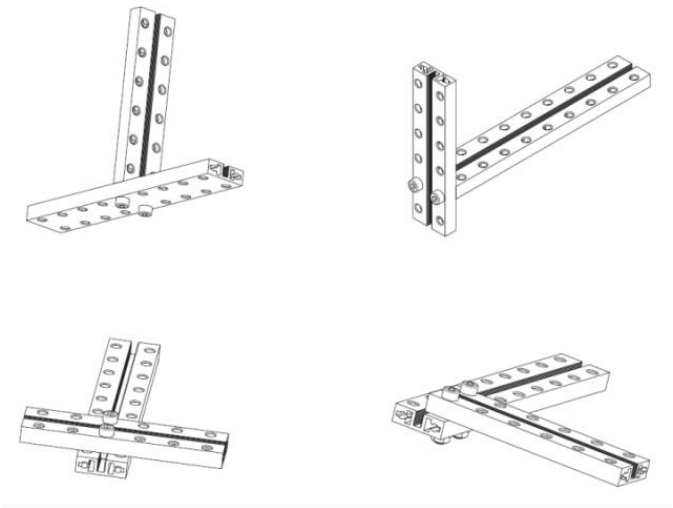


Fig. 3.5.2 Formas de la primera estructura

3.5.3 Plano de Viga con agujeros.

Para las soleras de aluminio extruido (Fig. 3.5.3). A continuación se detallan en la forma de cómo está constituido la distancia que existe entre agujeros así como también el grosor del material utilizado se muestra la escala en la que se trabaja para la fabricación de estas que es la escala milimétrica (mm)

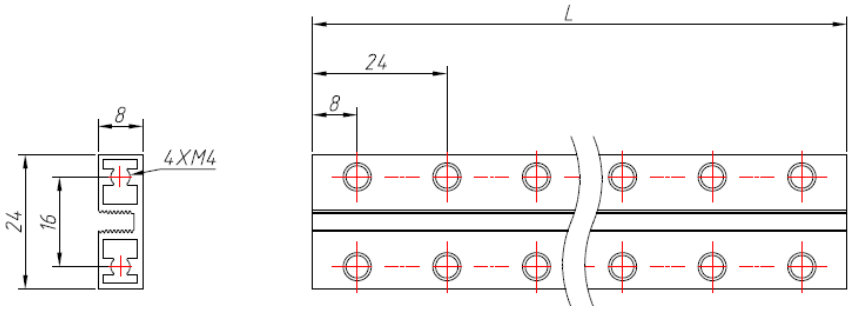


Fig. 3.5.3 vigas con agujeros

Fig. 3.5.3 (a) Estructura real con vigas

3.5.4 Motor Cd con Ensamble de Aluminio y Cables de Conexión

Diámetro de 37 mm de metal de engranajes CC del motor, junto con un soporte del motor, un conector del eje y una rueda de usos múltiples. El soporte del motor incluido es adecuado para 37mm motor. La rueda de usos múltiples se puede utilizar como una polea de temporización, o un cubo de rueda, o una rueda de depósito. (Fig. 3.5.4).

Fig. 3.5.4 Set de Motor cd

3.5.5 Plano para motor cd de 37 mm.

El motor que se utilizara en el elevador es uno de corriente directa con una dimensión de 37 milímetros (Fig. 3.5.5). A continuación se detallan en la forma de cómo está fabricado y se muestra la escala en la que se trabaja para la fabricación de estos que es la escala milimétrica (mm)

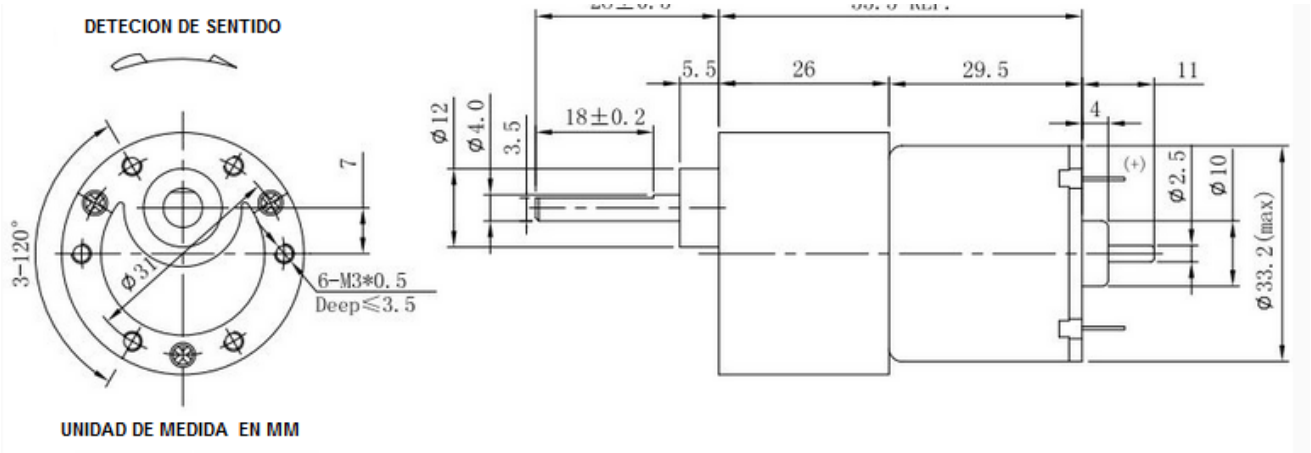


Fig. 3.5.5 plano motor CD

3.5.6 Motor montado en estructura

Paso básicos para el modelo de motor (Fig. 3.5.6), que será el que le dará vida a nuestro prototipo de elevación, seguiremos estos pasos indicados en numeración ordenadamente par que centremos de una manera adecuada y no dejar desnivelado la posición del ensamble para evitar fricción o rozamiento que en esta tipo de prototipos necesitamos de mucha precisión porque son motores muy delicados y el torque está calculado par bajo peso

Fig. 3.5.6 pasos de ensamble de motor

Tabla 3.5.4. Especificaciones para motor cd

Model	Voltage(V)	Gear Ratio	No Load		Load		
			Speed(rpm)	Current(mA)	Torque(Kg.cm)	Speed(rpm)	Current(Ma)
DC Motor-37 12V/50RPM	12.0	1 : 90	50±10%	<=100	4.5±	40±10%	<=300

3.5.7 Cálculos de Potencia y Corriente para Motor de Elevador.

Conforme a la tabla de especificaciones se realizaron comprobaciones matemáticas utilizando fórmulas de resistencia conocida como: Victoria, Reyna de Inglaterra donde necesitamos los datos de voltaje, resistencia y corriente. (Conocimientos adquiridos en la materia de circuitos I).Según lecturas reales estos son los cálculos realizados. (V=voltaje, R=resistencia, I=corriente y P=potencia). El voltaje al que deberá operar el motor será a 7.2 volts, por los datos adquiridos con la lectura que nos proporcionó el Óhmetro que fue de 13.5 Ω, con estos dos datos se pudo adquirir el consumo de corriente que es 0.533 mA (verificado con amperímetro en tiempo real) y por ultimo calculamos la potencia con los datos de voltaje entre la corriente de consumo dando un resultado de 13 Watts.

V=7.2 vcd

R=13.5 Ω

I=0.533 mA

P=13 W

Formula

$V = R.I$

P= $\frac{V}{I}$

Nota .Estos datos pueden variar con forme al peso de la cabina.

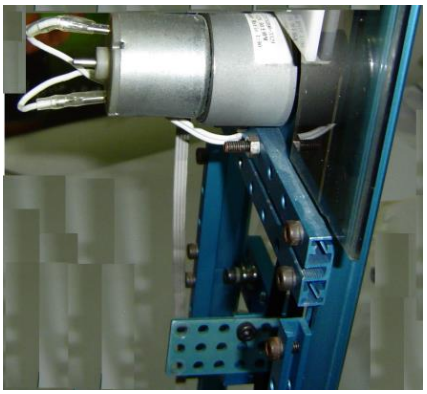


Fig. 3.5.7 (a) Motor montado en estructura Real (vista lateral)

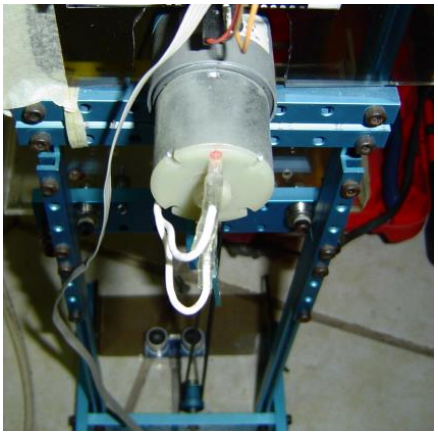


Fig. 3.5.7 (b) Motor montado en estructura Real (Vista Superior)

3.5.8 Flecha tipo D

Esta pieza nos servirá para fijar la contra parte del motor y se encontrara situada en la parte inferior donde ensamblaremos una micro polea para posteriormente le daremos el trabajo con la banda de hule por la cual se trasmitirá el movimiento para la cabina de elevador (Fig. 3.5.8)

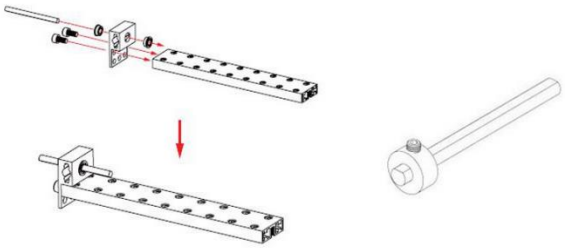


Fig. 3.5.8 Flecha tipo D

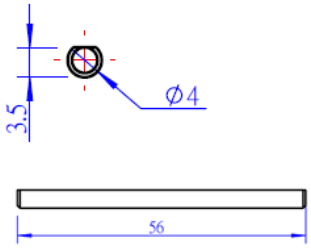


Fig. 3.5.9 planos de flecha

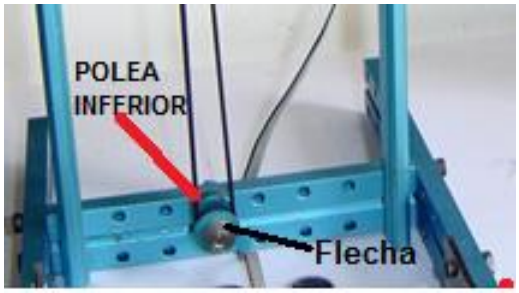


Fig. 3.5.10 Polea Montada (Estructura Real)

3.5.9 Deslizador

Esta pieza está fabricada con material de aluminio extruido en escala de milímetros y de suma resistencia (Figura 3.5.11). Fácil de ensamblar y como su nombre dice es muy buen deslizador por el trabajo que desempeñara en este prototipo acá pondré las direcciones que debemos de llevar acabo para que formemos un deslizador bien alineado y seguro para que en este elevador haga la función del guiado y estabilizador de la cabina del elevador (fig. 3.5.12).

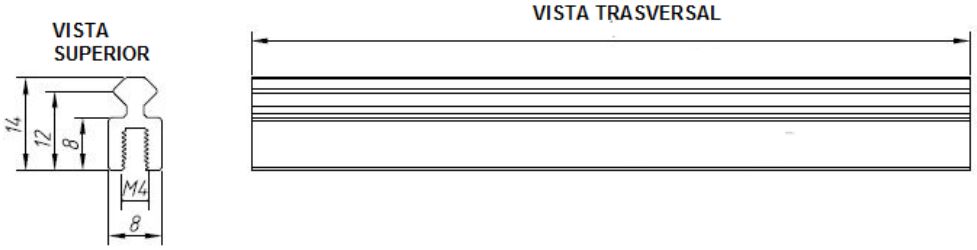


Fig. 3.5.11 Plano De Deslizador

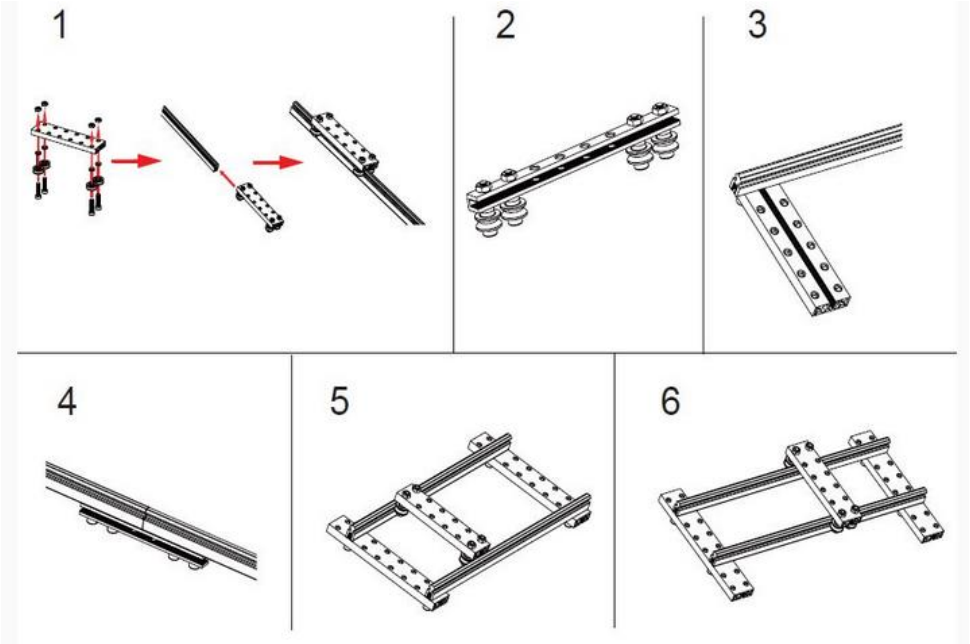


Fig. 3.5.12 Ensamble por pasos del deslizador

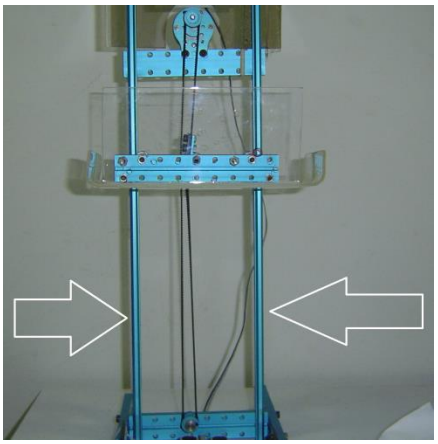


Fig. 3.5.13 Deslizadores en estructura real

3.5.10 Correa de Neopreno dentada.

Esta correa de Neopreno tipo MXL de700mm de longitud, dentada con espacios de paso 2.03 mm con 123 dientes y ancho de 6.6 mm está hecha de material resistente muy maleable ,este material es especial para que no patine la polea por la adherencia que tiene al material con el aluminio .Y no se dilata por la fricción y calor que produce la polea del motor este diseño es de tipo larga vida útil en el modo que lo emplearemos necesitamos seguir las instrucciones que a continuación se describen primeramente verificaremos que la polea del motor este al 100 por ciento alineada con la polea inferior para que no surjan efectos secundarios como saltos de dientes en tracción ,porque es muy importante que recordemos .este motor es para tracción trasera y delantera que mejor hacer un buen trabajo incrementaremos unos minutos más pero es mejor para la calidad del prototipo .

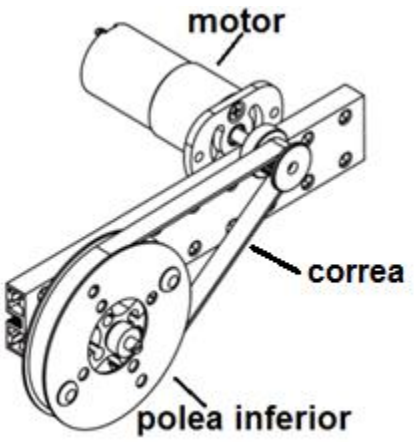


Fig. 3.5.14 Alineado de Correa

Fig. 3.5.15 Correa Montada y Alineada (estructura real)

3.6 Programación del algoritmo en Arduino

Después de simular el programa en las entradas y salidas de datos en la plataforma para poder hacer pruebas físicas por lo cual se requiere para hacer un sketch con este software Arduino, del cual dispone de la versión 1.0.1 para el Arduino Uno. Teniendo en cuenta que estamos haciendo uso de sensores ultrasónicos, comunicación serial y salidas digitales, es necesario configurar la plataforma Arduino Uno, incluir las librerías necesarias, Y crear las variables que vallan a utilizarse en el programa.

La idea es muy sencilla enviar una señal sonora casi inaudible (para la mayoría totalmente inaudible), que rebote en el obstáculo y al regresar de nuevo al sensor medir cuanto ha tardado en hacer el trayecto. De esta forma se puede medir la distancia. Velocidad del Sonido a 20ºC 340 metros por segundo. 340 m/s
El eco tiene que ir y volver. Sabemos que el eco ha tardado t segundos. Entonces tenemos que la distancia en metros d es igual:
2d = 340 * t de donde obtenemos d = (340 * t) / 2 .Se conectó el TRING al pin 7 y el ECHO al pin8.

```
#define echo Pin 7
#define trig Pin 6
#define LED Pin 13
Long duration, cm;

Void setup () {
  Serial. Begin (9600);
  Pin Mode (trig Pin, OUTPUT);
  Pin Mode (echo Pin, INPUT);
  Pin Mode (LED Pin, OUTPUT);
}

Void loop () {
 Digital Write (trig Pin, LOW);
 Delay Microseconds (2);
 Digital Write (trig Pin, HIGH);
 Delay Microseconds (5);
```

```
 Digital Write (trig Pin, LOW);
 Duration = pulse in (echo Pin, HIGH);

 cm = microsecondsToCentimeters (duración);

 Serial.print ("Milisegundos: ");
 Serial.print (duración);
 Serial.print ("  Distancia estimada: ");
 Serial.print (cm);
 Serial.println (" cm");

 Delay (1000);
}

Long microsecondsToCentimeters (long microseconds) {
  // La velocidad del sonido a 20º de temperatura es 340 m/s o
  // 29 microsegundos por centímetro.
  // La señal tiene que ir y volver por lo que la distancia a
  // La que se encuentra el objeto es la mitad de la recorrida.
  Retorno microseconds / 29 /2;
}
```

Esta librería tal cual viene tiene un time out o tiempo de espera de 3ms por lo que la distancia máxima que medirá será de 51cm algo escaso para mi gusto. La ventaja de tener un time out tan bajo es que se pueden hacer muchas mediciones en poco tiempo. Si parece poco el tiempo se cambiara.

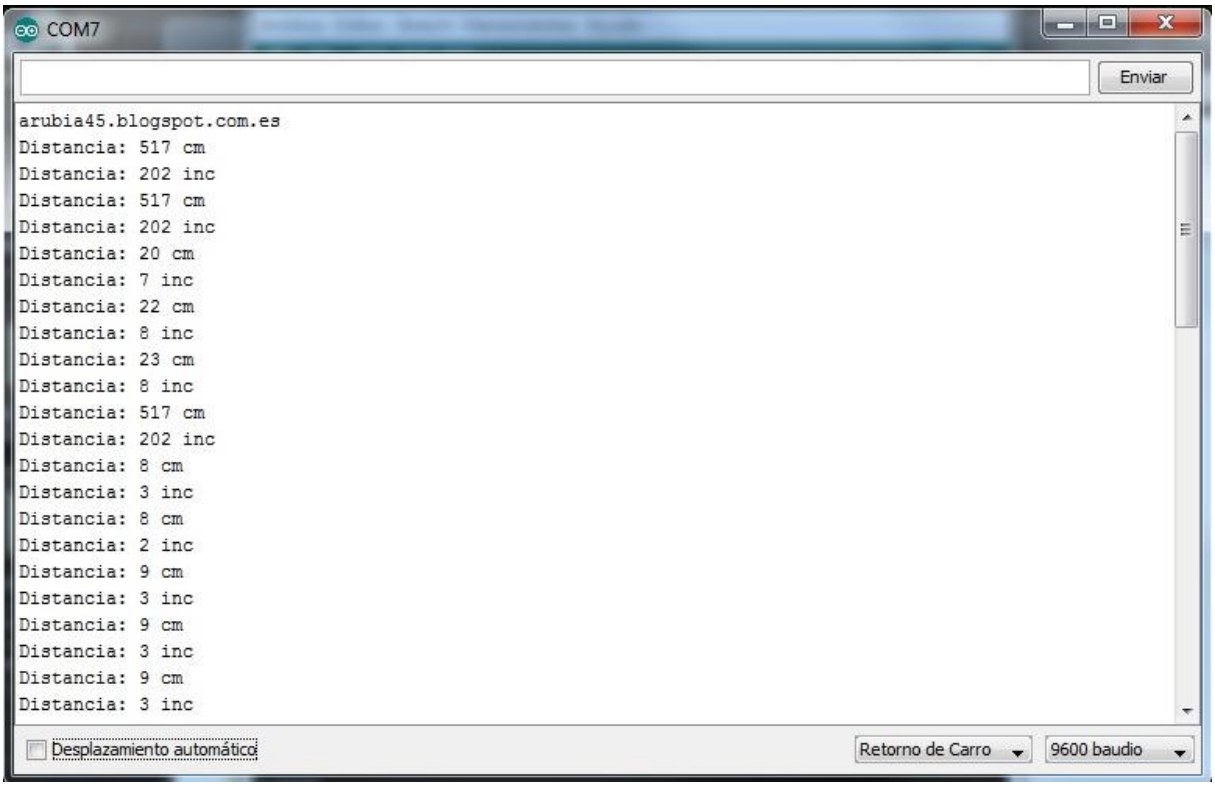


Figura 3.6 Sensor Midiendo

3.7 Desarrollo dos

Para poder comprender la complejidad, y los distintos tipos de soluciones en sistemas automatizados, iniciamos con el complemento. Desarrollando el sistema de silla de Ruedas Automática vía Blue Tooth.

3.8 Diagrama a bloques del funcionamiento de sistema Blue Tooth

En la figura se observa el diagrama a bloques, donde observamos los dispositivos que se encuentran conectados a nuestra placa Arduino lo cual tiene conectado a sus entradas el modulo blue tooth y a su puerto serial. A la salida se observa el modulo donde se encuentra conectado el driver que es un puente H, que necesita 2 señales digitales para que controle los motores de izquierda y derecha.

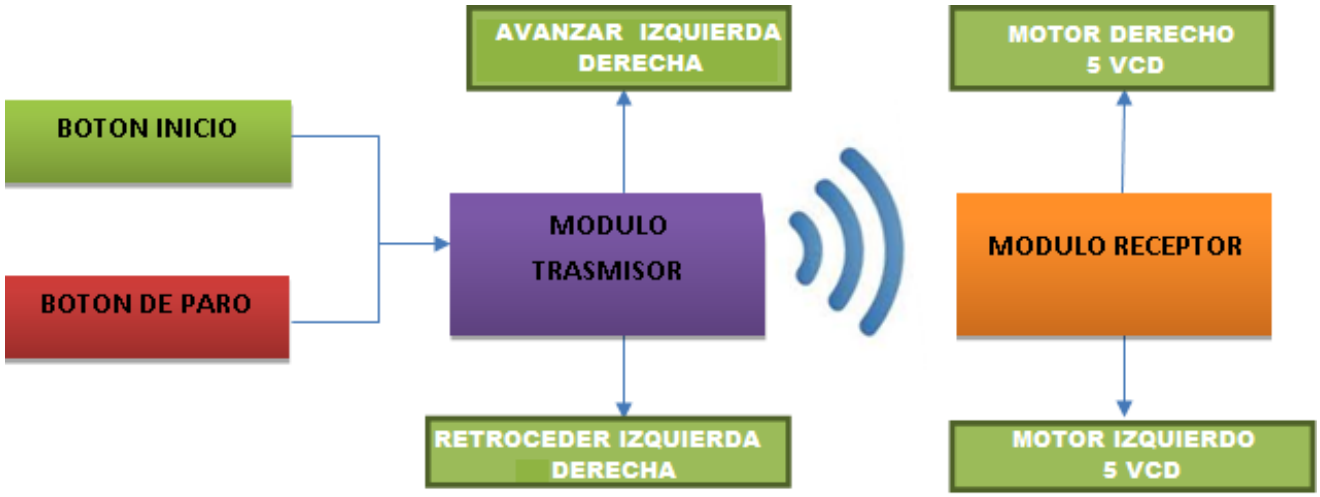


Fig. 3.8 Diagrama a Bloques Sistema Vía Blue Tooth

Proceso:

Paso 1. Primer paso es necesario instalar el módulo Bluetooth en el equipo e instalar los controladores. el módulo en uno de los puertos USB de repuesto. Windows detectará automáticamente el dispositivo y si se les pregunta a los conductores, apuntan a la unidad de CD-ROM. Una vez instalados los controladores, Windows puede tener que reiniciar el sistema.

Fig. 3.8.1(a) Modulo Blue Tooth Transmitiendo datos

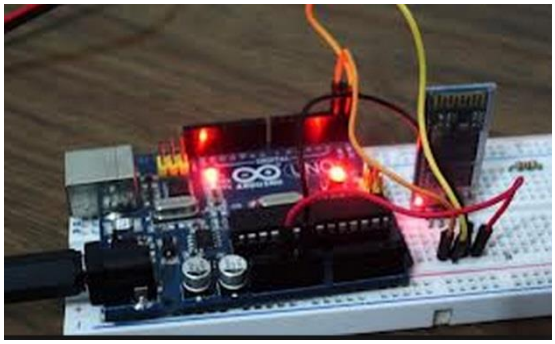


Fig. 3.8.1 (b) Modulo Blue Thooth en espera

Paso 2. Una vez que su equipo se ha reiniciado, conecte el módulo Bluetooth (Fig. 3.8.2). Un icono de Bluetooth debe aparecer en la barra de tareas (Windows XP SP2 tiene esta característica de soporte de Bluetooth). Su módulo Bluetooth ya está configurado y funcionando como un puerto serie. Ahora tenemos que configurar el módem en la placa de circuito.

Fig. 3.8.2 modulo blue Tooth

Paso 3. Vamos a crear la configuración. En primer lugar conectar el LED a la placa de circuito en el que pin que desea EXCEPTO pin 2 o 3 (hemos utilizado Pin 8 en este tutorial). Vea este ejemplo para ver cómo conectar un LED .Una vez que haya conectado el LED a la placa es el momento de conectar el módem Bluetooth. El módem Bluetooth que nos ocupa tiene 4 patas marcadas, PWR, GND, Rx y Tx. Es bastante sencillo: conectar el pin PWR al pin 5 V situado en el cableado, conecte el pin GND al pin de tierra en el cableado. A continuación, conecte el pin Rx al pin Tx de Cableado (pin digital 3) y conectar el pin Tx a la patilla Rx del cableado (pin digital 2). Una vez conectado a continuación, enchufe el adaptador de CA en el tablero, si todo está conectado a la derecha un pequeño LED verde comenzará a parpadear en el módem Bluetooth.

3.8.1 programación de modulo blue tooth

Inicio de código

```
/ / Clase de importación para configurar la conexión en serie con la placa
de cableado
. processing.serial import *;

Puerto serie;

/ / Botón de configuración
de color currentcolor; RectButton rect1, rect2; bolean bloqueadas =
false; void setup () {
 / / Configurar la ventana
  tamaño (200, 200);
  base Color = Color (102, 102, 102);
  currentcolor = base Color;
  / / Lista de todos los puertos serie disponibles en el panel de salida.
  / / Usted tendrá que elegir el puerto de la tarjeta .
  / / Conectado a partir de esta lista. El primer puerto de la lista es
  / / Puerto # 0 y el tercer puerto en la lista es el puerto # 2.
  print (Serial.list ());
  / / Abre el puerto que la tarjeta está conectado (caso 1)
  / / Que es el segundo puerto abierto (en la matriz)
  / / Asegúrese de abrir el puerto a la misma velocidad de cableado está
utilizando (9600bps)
  porta = nueva serie (esto, Serial.list () [2], 9600);
  / / Definir y crear botón rectángulo # 1
  int x = 30;
  int y = 100;
  int size = 50;
  button color = Color (153, 102, 102);
  highlight color = Color (102, 51, 51);
  rect1 = new RectButton (x, y, el tamaño, botón color, resalte);
  / / Definir y crear botón rectángulo # 2
  x = 90;
  y = 100;
  tamaño = 50;
  botón color = Color (153, 153, 153);
rect2 = new RectButton (x, y, el tamaño, botón color, resalte);


}
```

```
void draw () {


void update (int x, int y) {
  if (bloqueado == false) {
    rect1.update ();
    rect2.update ();
  } Else {
    cerrado = false;
  }
  / / Activar LED encendido y apagado si los botones presionados donde
  / / H = en (alta) y L = off (bajo)
  if (mouse Pressed) {
    if (rect1.pressed ()) {/ / button ON
      current color = rect1.basecolor;
      Port. Write ('H');
    } Else if {button (rect2.pressed ()) / / OFF
      current color = rect2.basecolor;
      Port. Write ('L');
    }
  }


}


class Button {
  int x, y;
  int size;
  base color, high light Color;

  Boolean sobre = false;
  Boolean pulsed = false;
  void update ()
  {
if (sobre ()) {
      current color = highlight Color;
    } Else {
      current color = base color;
    }
  }
  Boolean pulsed ()
  {
    if (más) {
      bloqueado = true;
      return true;
```

```
    } Else {
      cerrado = false;
      return false;
    }
  }
  Boolean sobre ()
  {
    return true;
  }
Void display ()
  {
  }

}

class Rect Button extended Button {
  Rect Button (int ix, iy int, int ISIZE, color , high light color)
  {
    x = IX;
    y = iy;
    size = ISIZE;
    base color = color;
    high light Color = high light;
    current color = base color;
  }
  bolean sobre ()
  {
    si (over Rect (x, y, tamaño, tamaño)) {
      sobre = true;
      return true;
    } Else {
      sobre = false;
      return false;
    }
  }
  void display ()
  {
    accidente cerebrovascular (255);
    llenar (currentcolor);
    rect (x, y, el tamaño, el tamaño);
  }

}
```

```
Boolean over Rect (int x, int y, int ancho, int altura) {
  if (mouse> = x && mouse <= x + && ancho
      mouse Y> = y && mouse Y <= y + alto) {
    return true;
  } Else {
    return false;
  }

}
```

3.8.2 Etapa de Control

La etapa de Control se realizó mediante la utilización de la placa Arduino UNO. Placa Arduino UNO). El diagrama de conexión de la activación de los motores está formado por un botón normalmente abierto (Fig. 3.8.2) Diagrama de Control de Activación). El diagrama está integrado por la detección de cambio de estado de entrada al pin del micro controlador, si este cambia, el Arduino detecta ese cambio como estado alto y se inicia el ciclo de inicio de los motores. Por lo contrario el Arduino no haría nada.

Al ser el estado en alto se le ordena al Arduino que active sus salidas previamente ya declaradas en el lenguaje de su programación, por lo tanto estas salidas son activadas y al mismo tiempo excitan la base de los transistores que se encuentran en la parte de etapa de potencia del módulo de motores y, estos, a su vez; se activan, haciendo el papel de girar derecha o izquierda.

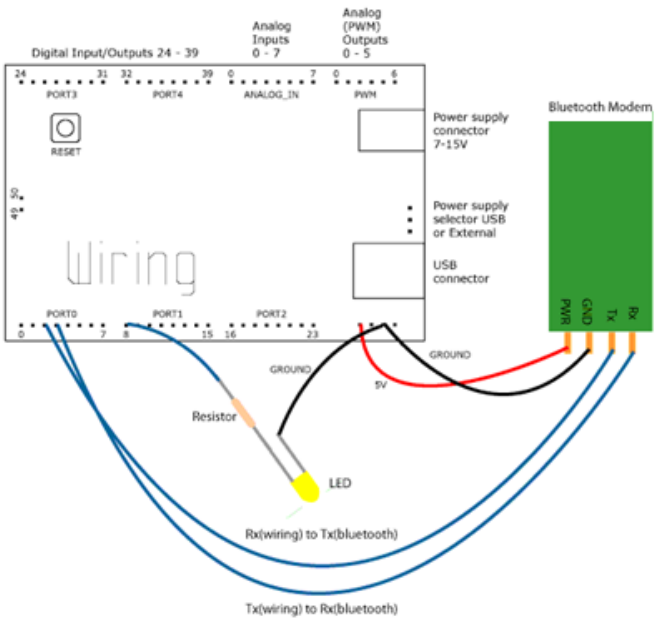


Fig. 3.8.2 Diagrama de Control de Activación

Para determinar la profundidad de una fisura, se cuentan con dos tiempos t_1 y t_2 para distancias X y 2X, respectivamente, dicha profundidad se obtiene mediante la siguiente expresión:

$$C= X\,(4(t_1^2 + t_2^2)/(t_2^2 - t_1^2))^{0.5}$$

Dónde:

C = profundidad de la grieta

X = distancia inicial

t1 = tiempo de la distancia inicial (X)

t2 = tiempo del doble de la distancia (2X)

Todos los datos y resultados obtenidos se anotan en la tabla de interpretación de datos.

Para obtener el módulo de elasticidad dinámico a partir de la velocidad de pulso, se cuenta con las siguientes expresiones:

1. *Para probetas de laboratorio : $Ed = 1.02 * V^2 * W * 10^5$*
2. *Para losas : $Ed = 0.961 * V^2 * W * 10^5$*
3. *Para hormigón en masa : $Ed = 0.866 * V^2 * W * 10^5$*

Dónde:

Ed = módulo dinámico de elasticidad del hormigón

V = velocidad de pulso

W = Peso volumétrico del hormigón

No es fácil estimar la relación que existe entre el pulso ultrasónico y la resistencia del hormigón; pues el tipo de agregado, la relación agregado-cemento, la edad del agregado y las condiciones de curado influyen en ella.

El equipo puede emplearse para llevar el control del hormigón en una construcción, esto se logra mediante el uso de cilindros de prueba. En ellos se hacen mediciones de la velocidad de pulso y resistencia a compresión, con estos

datos se hace una gráfica de resistencia en contraposición con la velocidad de pulso que servirá como referencia y así poder hacer ensayos al hormigón ya colocado en elementos estructurales, para lo cual basta con medir la velocidad de pulso en cada elemento y compararla con la gráfica obtenida de antemano en los cilindros de prueba.

3.8.3 Etapa de Trasmisión de Datos

Esta etapa se compone por un circuito codificador de señal, que es trasmitida a través de una antena. Esta trasmisión es posible gracias a que el modulo transmisor (Fig. 3.8.3). Diagrama de Conexión del Trasmisor) contiene un pin de entrada de datos, que previamente han sido codificados por su correspondiente circuito de codificación. En la antena se envían pares de paquetes que se es propagado a través del medio.

Posee un direccionamiento que de acuerdo a su configuración, dicha señal solo es reciba por los módulos que previamente ya han sido programadas con ese mismo direccionamiento. Es por ello que se realiza esto para evitar ruidos en la señal trasmitida, se direcciona y se codifica la señal. Se alimenta por escasos 9 V en corriente directa, y en la etapa de codificación se utiliza una antena con una resistencia menor a los 50 Ohm para evitar pérdidas por caídas de voltaje.

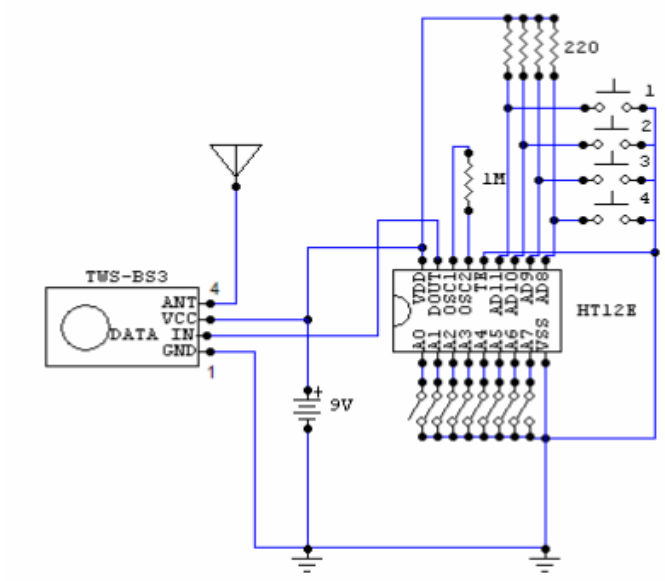


Fig. 3.8.3 Diagrama de Conexión Trasmisor

3.8.4 Etapa de Recepción de Datos

Para la etapa de recepción de (Fig. 3.8.4). Diagrama de Recepción) datos se utilizó un blindaje para evitar ruido en la recepción, estos ruidos pueden ser ocasionados por señal motores o transformadores, entre otros aparatos que usen una trasmisión de armónicos. Este tipo de blindaje se trata de decodificar la señal, que es recibida por su antena en la entrada de datos del receptor; esta señal viene codificada por el módulo de un transmisor y, gracias a este se puede lograr una comunicación sin ruidos y reduciendo así la perdida de comunicación.

Su alimentación es alrededor de los 5 V de corriente de CD y su máximo de voltaje para su perfecto trabajo es de 12 V. En este módulo se trabajó a 5V para evitar calentamiento en el circuito de recepción, y a su vez alargar el periodo de vida de los receptores. El modulo receptor usado fue el RWS-371-6; este trabaja a una frecuencia de los 433 MHz y, en lugares ideales su recepción alcanza los 100 metros de longitud entre modulo transmisor y receptor.

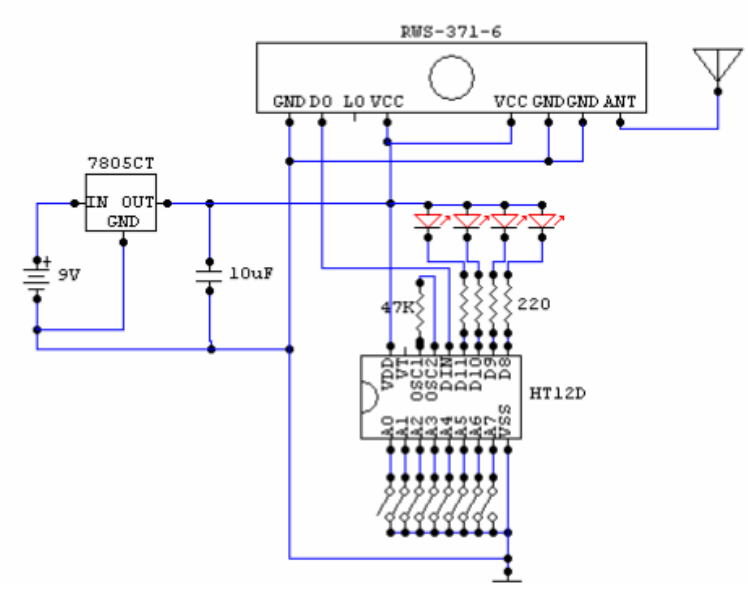


Figura 3.8.4 Diagrama de Recepción

3.8.5 Etapa de Alimentación

Para la alimentación de los módulos se usó una batería de 7.2 vcd, seguidamente se conectó al pin de entrada que alimenta al Arduino y que este tiene una etapa de regulación de voltaje el cual convierte los 7.2 volts a 5 volts, que es lo que necesita el módulo de blue tooth para que funcione al 100 por ciento., obteniendo como resultado una corriente directa que alimentara a los componentes electrónicos. y en conjunción con sus respectivos capacitores que permitirán una salida constante de 5 V de CD.

Se obtiene un Voltaje regulado, ya que se cuenta con una resistencia de tipo pre set conectado a la entrada 2 del regulador de voltaje; que regula el aumento o la disminución del voltaje de salida, logrando así un voltaje variable y ajustable a lo que se requiera

3.8.6 Etapa de Programación

Programación del Trasmisor.- En esta etapa se declara el ciclo que se ejecutara durante el resto del programa, que será previamente cargado al micro controlador. Al iniciar el Arduino, se declaran las variables que se usaran durante el resto del programa; entre ellas se declaran también la salidas, así como las entradas del micro controlador, dejando claro que se usó retardos para evitar una lectura errónea en las entradas del Arduino.

El micro controlador está constantemente leyendo y comparando si existe algún cambio en el estado de su pines de entrada, si esta es correcta; ejecuta su código de programación correspondiente y a su vez envía comandos a través de la señal de radiofrecuencia; cuyo comando será activar a los módulos receptores y la activación del motor. Una vez trascurrido este ciclo, esperara 3 minutos para desactivar todo y dejar al micro controlador con sus entradas y salidas al estado inicial del Arduino.

INICIO DE CODIGO

```
//const int button Pin [] = {2,3};    // PINES DE SALIDA
//int button State = 0;         // VARIABLE QUE LEO EL ESTADO DEL BOTON
Int sensor=4; //SENSOR VERDE-BLANCO
Int s5=5; //TR1 BLANCO-CAFE
Int s6=6; //TR2 BLANCO-AZUL
Int s7=7; //TR3 AZUL
Int s8=8; //TR4 CAFE
Int s9=9; //R1 MOTOR UNO 5VCD
Int s10=10; //R2 MOTOR DOS
Int s11=11; //R3 MOTOR TRES
Int s12=12; //LED INICIO
```

```
int manual=2;
Int automatic=3;
Int x=0;
Int button State1=0;
Int button State2=0;
Void setup () {
 //SE DECLARAN LAS SALIDAS
Pin Mode (s5, OUTPUT);//TR1
Pin Mode (s6, OUTPUT);//TR2
Pin Mode (s7, OUTPUT);//TR3
Pin Mode (s8, OUTPUT);//TR4
Pin Mode (s9, OUTPUT);//R1 MOTOR UNO 5 VCD
Pin Mode (s10, OUTPUT); //R2 MOTOR DOS
Pin Mode (s11, OUTPUT); //R3 MOTOR TRES
Pin Mode (s12, OUTPUT); //LED INICIO
Digital Write (s12, LOW); //APAGAR LED
Digital Write (s11, LOW); //APAGAR MOTOR UNO
Digital Write (s10, LOW); //APAGAR MOTOR
Digital Write (s9, LOW); //APAGAR MOTOR UNO 5 VCD
Digital Write (s8, LOW); //APAGAR TR1
Pin Mode (manual, INPUT);
Pin Mode (automatic, INPUT);
  buttonState1=0;
  buttonState2=0;
Delay (1000);

}
```

```
Void loop () {
    buttonState1= digital Read (manual);
    buttonState2= digital Read (automatic);
Delay (1000);
If (buttonState1== HIGH)
  {
    x=1;
  }
if (buttonState2== HIGH)
   {
     x=2;
   }

if(x==1)
   {
Digital Write (s12, HIGH); //LED INDICADOR DE TRASMISION
Digital Write (s11, HIGH); //Encender MOTOR UNO
Digital Write (s10, HIGH); //Encender MOTOR DOS
Digital Write (s9, HIGH); //Encender MOTOR UNO 5VCD
Digital Write (s8, HIGH); //Enviar TR1
Delay (180000); // ESPERA 3 MINUTOS
Digital Write (s12, LOW); //LED INDICADOR DE TRASMISION
Digital Write (s11, LOW); //Apaga MOTOR UNO
Digital Write (s10, LOW); //Apaga MOTOR DOS
Digital Write (s9, LOW); //Apaga MOTOR UNO 5 VCD
Digital Write (s8, LOW); //Apaga TR1
```

```
   }

If(x==2) {
Digital Write (s12, HIGH); //LED INDICADOR DE TRASMISION
Digital Write (s11, HIGH); //Encender MOTOR UNO
Digital Write (s10, HIGH); //Encender MOTOR DOS
Digital Write (s9, HIGH); //Encender MOTOR UNO 5 VCD
DigitalWrite (s8, HIGH); //Enviar TR1
    }


}
```

3.8.7 Programación del Receptor

Se declara el ciclo que se ejecutara durante el resto del programa, que será previamente cargado al micro controlador. Al iniciar el Arduino, le decimos mediante código de programación que se declaren entradas, variables; salida y/o datos de comparación que serán de utilidad para que el micro controlador ejecute una orden previamente ya programada. Al recibir el dato correcto del transmisor se ejecutara una acción, lo cual dará como lugar a la activación del motor.

Si por lo contrario no recibe ninguna señal de radiofrecuencia, o en su peor caso no es el comando que se espera este no se activara. Estos receptores han sido programados para que solo acepte una señal que sea del módulo transmisor, ya que ha sido codificada para evitar ruidos en el proceso de trasmisión y a su vez usa un decodificar en paralelo para poder descifrar esa señal y poder realizar la ejecución de código correspondiente.

```
/*CODIGO   RA2-TA5
RA2-->Receptor Entrada Arduino UNO
*/
Int e2 = 2;//RA2-TA8
Int e3= 3;//RA3-TA7
Int e4 = 4;//RA4-TA6
Int s5 = 5;//RA5-TA5
Int s6 =6;//MOTOR UNO 5 VCD
Int s7 =7;//MOTOR DOS
Int s8 =8;//MOTOR TRES
Int inl2 = 0;
Int inl3 = 0;
Int inl4 = 0;
```

```
Int inl5 = 0;
Int activado=0;

Void setup (void) {
Pin Mode (e2, INPUT);//RA2-TA8
Pin Mode (e3, INPUT);//RA3-TA7
Pin Mode (e4, INPUT);//RA4-TA6
Pin Mode (s5, INPUT);//RA5-TA5
Pin Mode (s6, OUTPUT); //R1 MOTOR UNO
Pin Mode (s7, OUTPUT); //R2 MOTOR DOS
Pin Mode (s8, OUTPUT); //R3MOTOR TRES
inl2=digital Read (2) ;//---> LEER EL ESTADO DE BOTON MANUAL
inl3=digital Read (3) ;//---> LEER EL ESTADO DE BOTON MANUAL
inl4=digital Read (4); // ---> LEER EL ESTADO DE BOTON AUTOMATICO
inl5=digital Read (5); // ---> LEER EL ESTADO DE BOTON AUTOMATICO

  }

Void loop () {

inl2=Digital Read (2) ;//---> LEER EL ESTADO DE BOTON MANUAL
inl3=Digital Read (3) ;//---> LEER EL ESTADO DE BOTON MANUAL
inl4=Digital Read (4); // ---> LEER EL ESTADO DE BOTON AUTOMATICO
inl5=Digital Read (5); // ---> LEER EL ESTADO DE BOTON AUTOMATICO

If (inl2==HIGH & inl3==HIGH & inl4==HIGH & inl5==HIGH)
{
```

```
Digital Write (s6, HIGH); //Encender MOTOR UNO 5VCD
 }

Else if (inl2==LOW & inl3==LOW & inl4==LOW & inl5==LOW &activado==0)
{
Digital Write (s6, LOW); //ENCENDER MOTOR UNO 5 VCD
 }

Else if (inl2==HIGH & inl3==LOW & inl4==LOW & inl5==LOW)
{
Digital Write (s8, HIGH); //Encender MOTOR UNO
Digital Write (s7, HIGH); //Encender MOTOR DOS
Digital Write (s6, HIGH); //Encender MOTOR TRES
Activado=1;
 }

else if (inl2==LOW & inl3==LOW & inl4==LOW & inl5==LOW &activado==1)
{
Digital Write (s8, LOW); //Encender MOTOR
Digital Write (s7, LOW); //Encender MOTOR
Digital Write (s6, LOW); //Encender MOTOR
Delay (100);
Digital Write (s8, HIGH);
Delay (100);
Digital Write (s8, LOW);
Delay (100);
Digital Write (s8, HIGH);
```

```
Delay (100);
Digital Write (s8, LOW);
Delay (100);
Digital Write (s8, HIGH);
Delay (100);
DigitalWrite (s8, LOW); //Apagar MOTOR UNO
Activado=0;
 }
else if (inl2==LOW & inl3==LOW & inl4==LOW & inl5==LOW &activado==0)
{
Digital Write (s8, LOW); // on MOTOR UNO
Digital Write (s7, LOW); // on MOTOR DOS
Digital Write (s6, LOW); // on MOTOR TRES
 }
```

3.9 Modelo Hardware Abierto

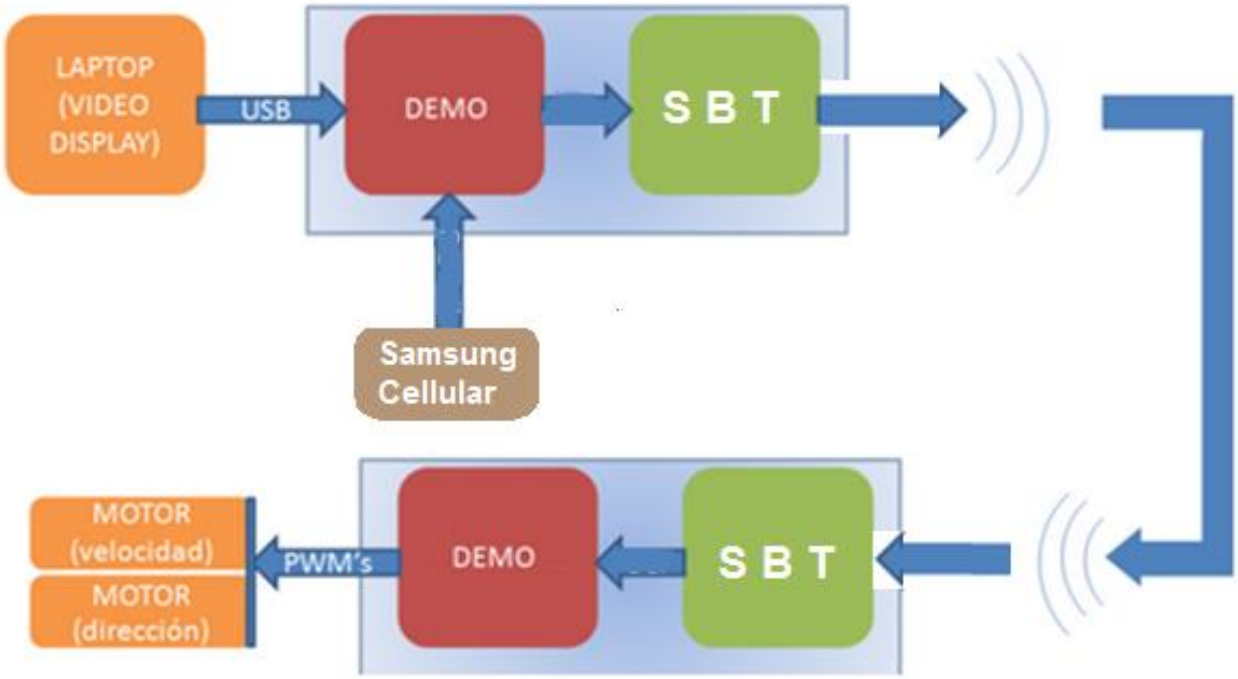


Fig. 3.9 Diagrama a Bloques de Silla de Ruedas

3.9.1 Hardware

Esta sección presenta el hardware necesario para el desarrollo del proyecto "Manejo por Blue Tooth de Silla de Ruedas a Escala". (Fig. 3.9.1):

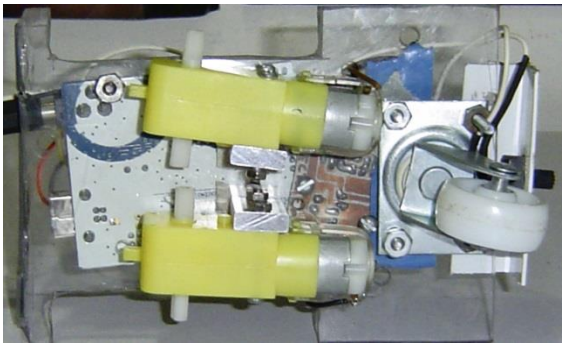


Fig. 3.9.1 Estructura de silla de Ruedas a escala.

Se hizo uso de una estructura de material acrílico a escala sobre el cual se montaron las tarjetas con el Micro controlador, el módulo Blue Tooth y la etapa de potencia; que hicieron posible el control y movimiento de la silla de ruedas. Cabe mencionar que el prototipo es alimentado con una batería de 7.2 Volts a 2.2 Amperes, estas baterías son comúnmente utilizadas en prototipos con motores de alto torque como los componentes y motores actualmente acoplados en este prototipo (Fig. 3.9.1.1).

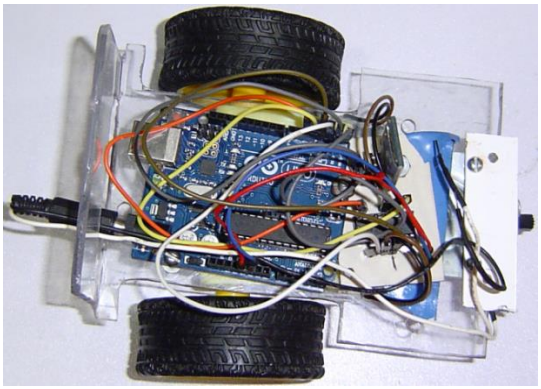


Figura 3.9.1.1 Silla de Ruedas escala con los módulos montados

3.9.2 Tarjeta Código Abierto

Esta Tarjeta de código abierto nos permitió utilizar las capacidades del Micro controlador de una manera integral, pues tiene muchas herramientas ya diseñadas para poder acceder a los recursos más importantes y comunes del micro como es la tarjeta Arduino que integran un grupo de pines que son los que tiene salidas PWM (Pulsos anchos de modulación), el puerto serial, etc.(Fig. 3.9.2).

Los módulos mencionados anteriormente, fueron los principalmente utilizados en nuestro proyecto.

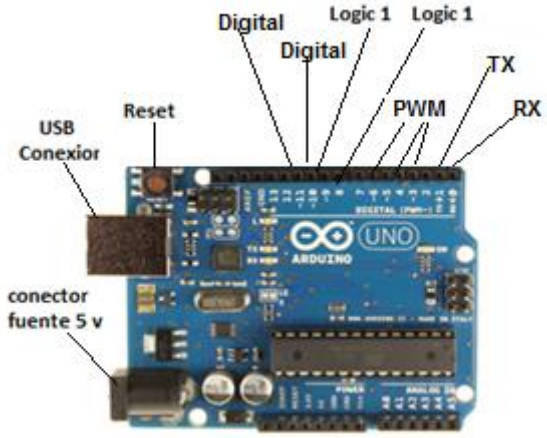


Fig. 3.9.2 Tarjeta de desarrollo Arduino Uno.

3.9.3 Tarjeta Blue tooth

Esta tarjeta cuenta con un módulo de Blue Tooth (Fig. 3.9.3) .mediante el cual se envía un puerto serial Inalámbrico. Usando esta herramienta se logra eliminar el cable que comunica el Micro controlador que genera y mantiene el control de los motores del prototipo para poder manejarlo de manera inalámbrica desde la comodidad del usuario.

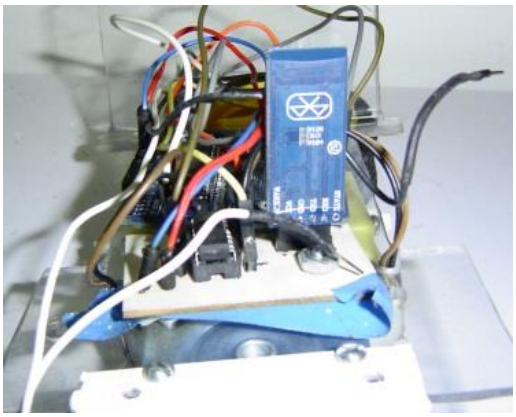


Fig. 3.9.3 Blue Tooth montado en placa

3.9.4 Puente H

Este circuito (LM 293D).permite por medio de un PWM controlar el sentido de giro y dirección de un motor de corriente directa, esto se realiza controlando el ancho de pulso de la señal y el sentido en el que la corriente fluye en el circuito para el control de los dos motores como se lee en el diagrama de instalación para dos motores cd (**Fig. 3.9.4 a y b**)

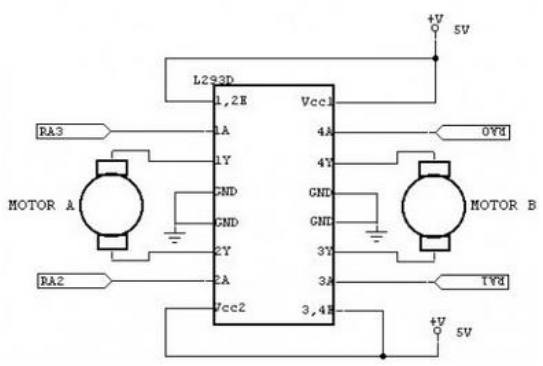


Fig. 3.9.4 (a) Diagrama puente H

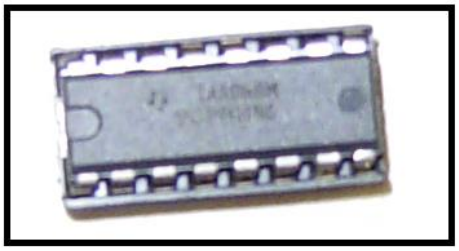
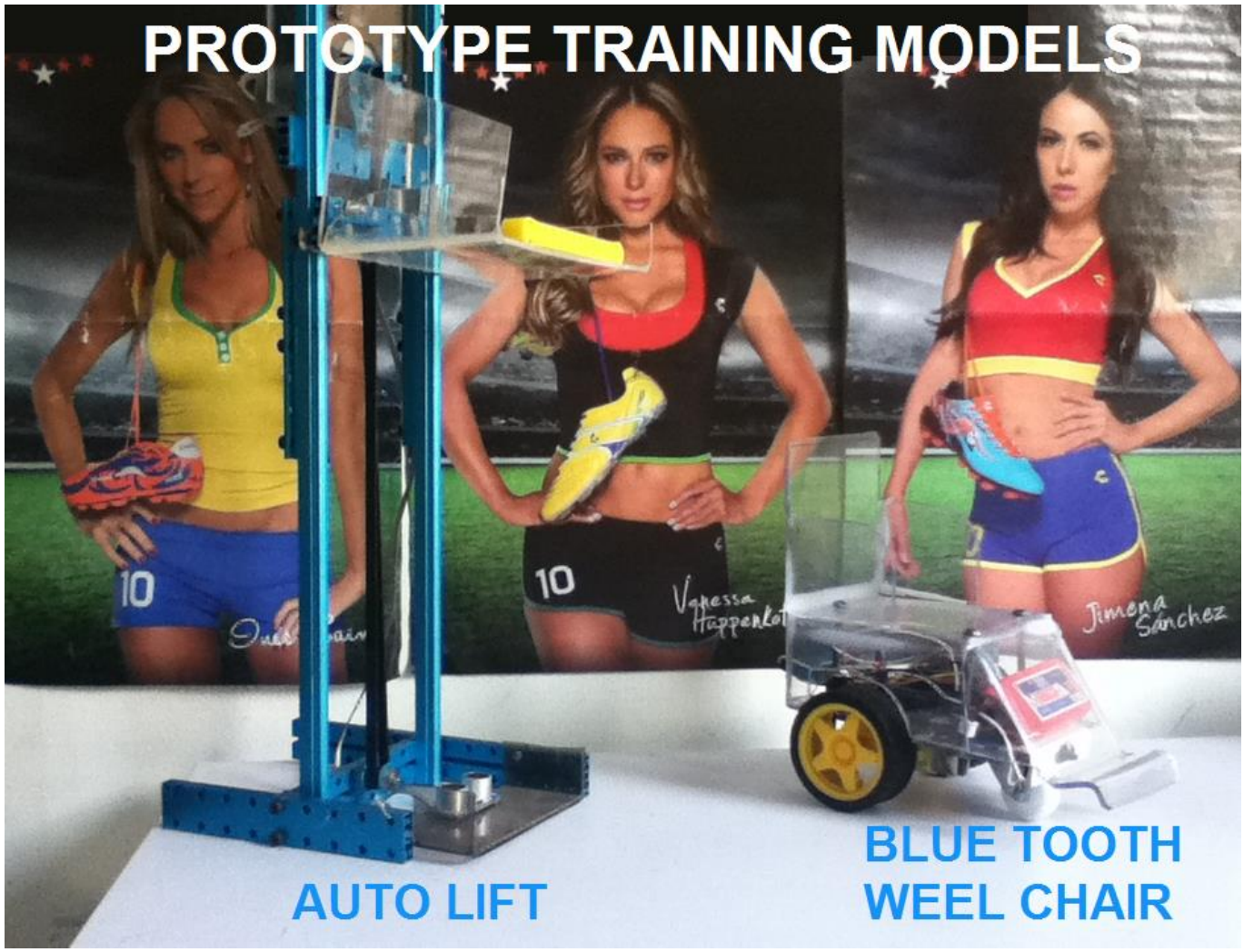


Fig. 3.9.4 (b) LM293D

4. Análisis de Resultado


PROTOTYPE TRAINING MODELS
AUTO LIFT
BLUE TOOTH
WEEL CHAIR

4.1 Análisis de Resultados.

En el uso de herramienta como es la plataforma Arduino, obtuvimos resultados muy satisfactorios, pues gracias al diseño de la placa Arduino UNO (**Fig. 4.1**), no es necesaria la creación de placa para conectar el micro controlador, ni la interfaz entre el micro controlador y los distintos dispositivos que utilizamos como los sensores con la excepción del puente H que para el circuito si fue necesario la creación de un pequeña placa con diseños y manufacturas realizadas en casa.

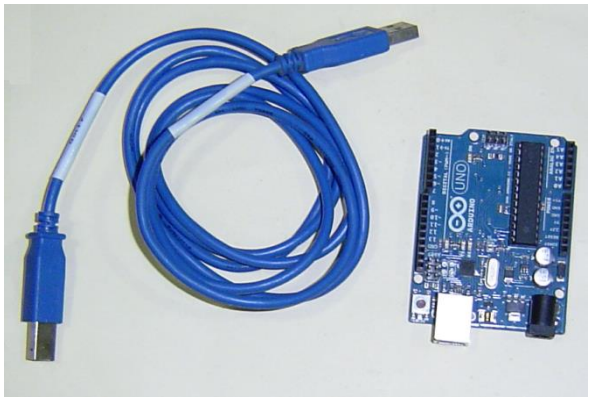


Fig. 4.1 (a) Arduino Uno

El contar con entradas analógicas, salidas digitales, puerto serial y serial USB en un solo sistema, nos permite ahorrar trabajo, tiempo y dinero y espacio en la elaboración de sistemas de entrenamiento, pues es sistema bastante amigable con el programador y simulador electrónico para el diseño de componentes electrónicos utilizados en la placa de control de giro de motor cd para estos prototipos y que fue compatible para realizarlo en esta plataforma.(Fig. 4.1.1) Podemos observar detalladamente el sistema realizado con los dispositivos conectados en las entradas y salidas correspondientes.

Figura 4.1.1 Arduino Uno –Puertos

No tuvo ningún problema con el uso del sistema Arduino, pues tanto las entradas digitales como analógicas, fue de mucha ayuda, al igual que los puertos de comunicación serial, y gracias a todo esto os dos sistemas que se construyeron se obtuvieron buenos resultados; el conexionado de los dispositivos digitales y analógicos(fig. 4.1.2),no proporciono problema alguno gracias a los conocimientos adquiridos en el proceso de preparación antes de iniciar con este proyecto .Y la colaboración del director y asesor de trabajo profesional.

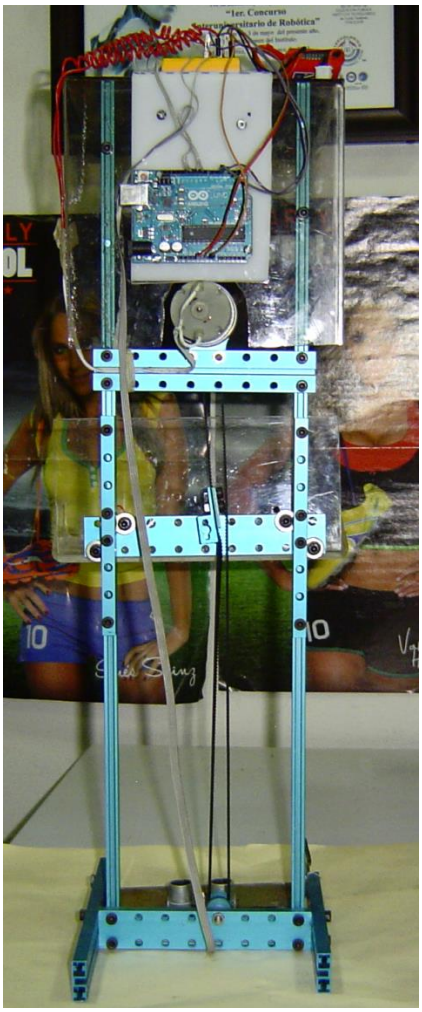


Fig. 4.1.2 Placa Arduino montada en elevador

Con el sistema propuesto anteriormente se resolvió el problema de caída de tensión, provocado por largas distancias de conexión de los cables. Se obtuvo mayor rapidez de activación. Se eliminaron los diferentes tipos de interruptores al realizar un sistema automático. Los interruptores fueron sustituidos por sensores ultrasónicos, donde se manda la activación y desactivación del sistema. Se colocó un sensor para activar el llamado de nivel uno o nivel dos del elevador automáticamente (Auto Lift).

En la figura 4.1.3. Observamos el análisis para el nivel uno, que obtuvimos precisión y exactitud al llamado de la plataforma de elevación y se realizó la comprobación con un instrumento de medición donde se obtuvo un 3 % de error al cual se obtuvo el 97% de efectividad

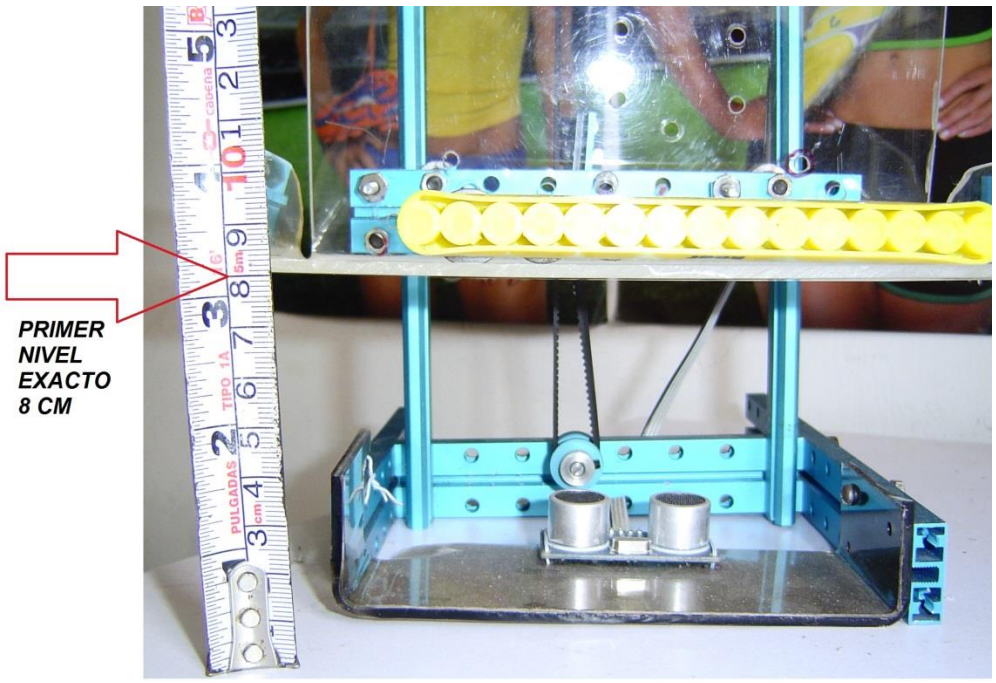


Fig. 4.1.3 Elevación a primer nivel

En la figura 4.1.4. Observamos el análisis para el nivel dos, que obtuvimos precisión y exactitud al llamado de la plataforma de elevación y se realizó la comprobación con un instrumento de medición midiendo 24 cm de altura después de 10 veces que se tuvo en función el elevador continuaba con la misma exactitud en la medida de la altura.

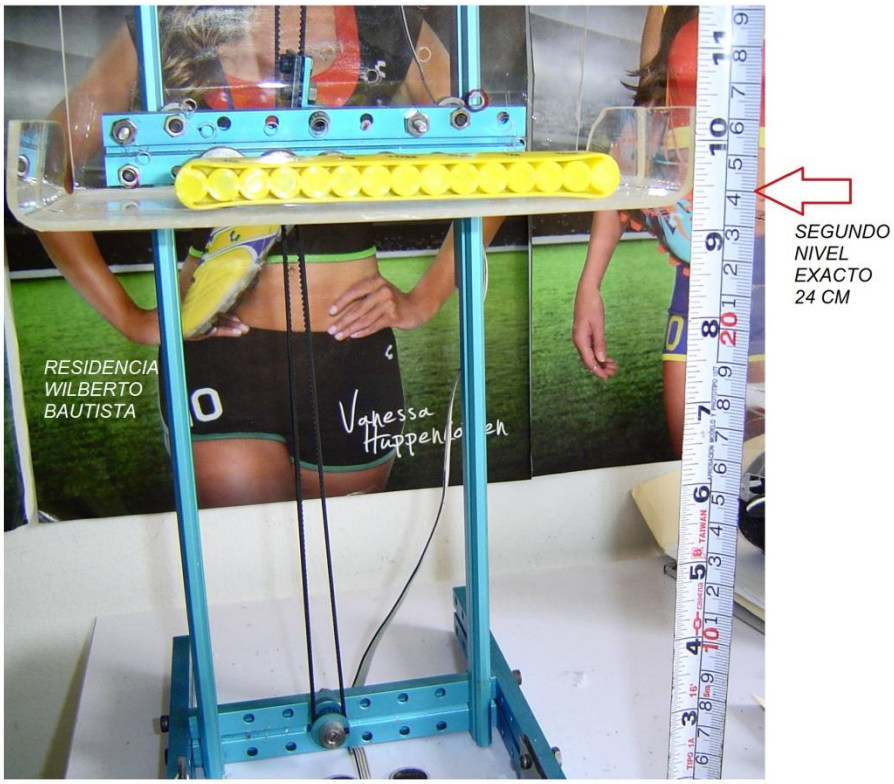


Fig. 4.1.4 Elevación a segundo nivel

Con el sistema de trasmisión vía blue tooth mejoramos el control manual del móvil

dejando a un lado el sistema anterior que operaban con Joystick. Y este sistema también reduce perdidas en las caídas de tensión por el cableado .este sistema también extendemos al doble la vida de la batería ya que es operado con batería recargable de larga duración.

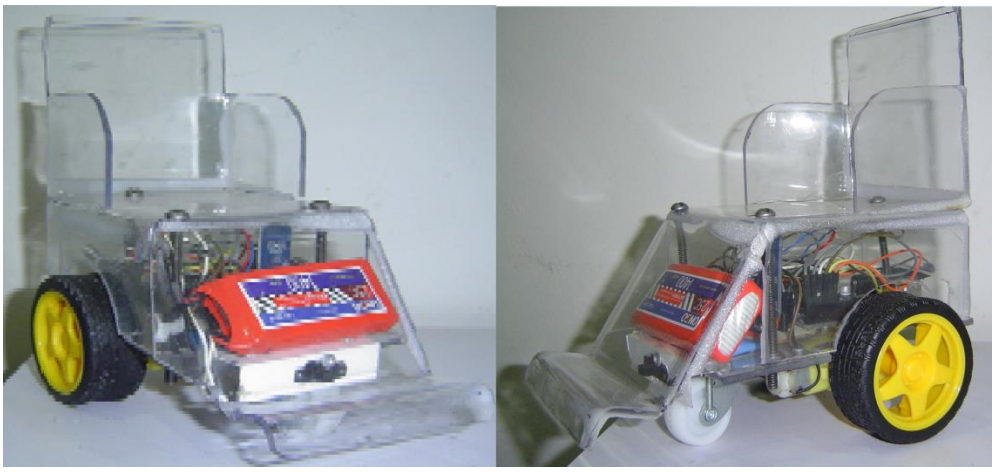


Figura 4.1.5 Silla de Ruedas Blue Tooth

Figura 4.1.6 Silla de Ruedas Vista frontal

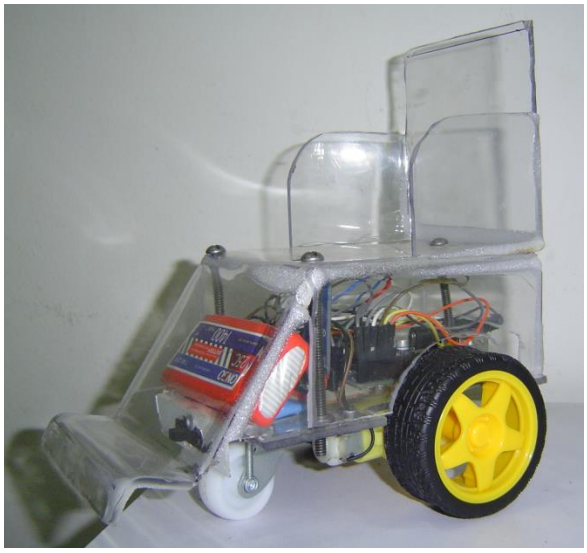


Figura 4.1.7 Silla de Ruedas Vista lateral

4.2 Conclusiones.

En este trabajo se ha explorado la utilización de sistemas Open Mechanics y Open Hardware como una nueva herramienta para solucionar problemas de faltas de prácticas de laboratorio con la elaboración de diseños que se implementaron. Así como la construcción de programas en el Open Sourse de la plataforma Arduino, de lo cual se obtuvieron resultados favorables gracias a las herramientas de Sistema Abierto.

Dentro de los resultados obtenidos, podemos destacar que el uso de la herramienta makeblock, me fue indispensable en la implementación de

estructuras con deslizadores, pues facilito el diseño gracias a las vigas de aluminio extruido que son maleables y fácil de ensamblar. Con ello se comprobó que la imaginación que mantenemos en la mente podemos dejarla correr y realizar infinidad de modelos creativos con este sistemas de open mecanics, la imaginación es la perspectiva a un diseño y esta construcción a un proyecto (modelo didáctico)

En general, el trabajo realizado en este semestre específicamente en sistemas de automatización, con sistemas digitales y sensores ultrasónicos, me sirvió para actualizar e incrementar mis conocimientos en la rama de le programación y la Robótica. El uso de estos métodos basados en lógica digital que se trabajó demostró la solución de un problema a nivel lógico. Todo esto se logró al implementarse en un algoritmo que es capaz de resolver el problema.

Anexos

Anexo 1. Código de programa de control automático para elevador de 7 niveles utilizando PLC

Se realizó una programación en lenguaje (LADDER) escalera para PLC de la marca siemens .El sistema esta creado para siete nivele y también cambiaría el modelo de la cabina y la puerta que también será automática

RESUMEN

Un ascensor o elevador es un aparato que sirve para trasladar personas o cosas de unos niveles de altura a otros. La cabina debe acudir a cada altura cuando sea solicitado por un usuario desde la planta o desde el interior. El objetivo del proyecto es diseñar un control eléctrico y electrónico que controle e integre todos los elementos propios de un ascensor. Se establecerán las bases para una implantación real, marcando rutinas y esquemas de trabajo a posibles operarios de montaje y mantenimiento, para la localización rápida de dispositivos y Averías. Otro fin es el de establecer un coste de materiales y herramientas necesarias para la elaboración de la maniobra del ascensor descrita. Para diseñar los dispositivos y el control, se ha tenido especial atención en cumplir la normativa en materia de aparatos elevadores publicada por el Departamento de Industria. El “cerebro” del control lo forma un Autómata Programable Siemens de la serie S7-200 apoyado por un micro controlador Microchip de la serie 16 que realiza las funciones de gestión y Comunicación de errores.

Palabras clave

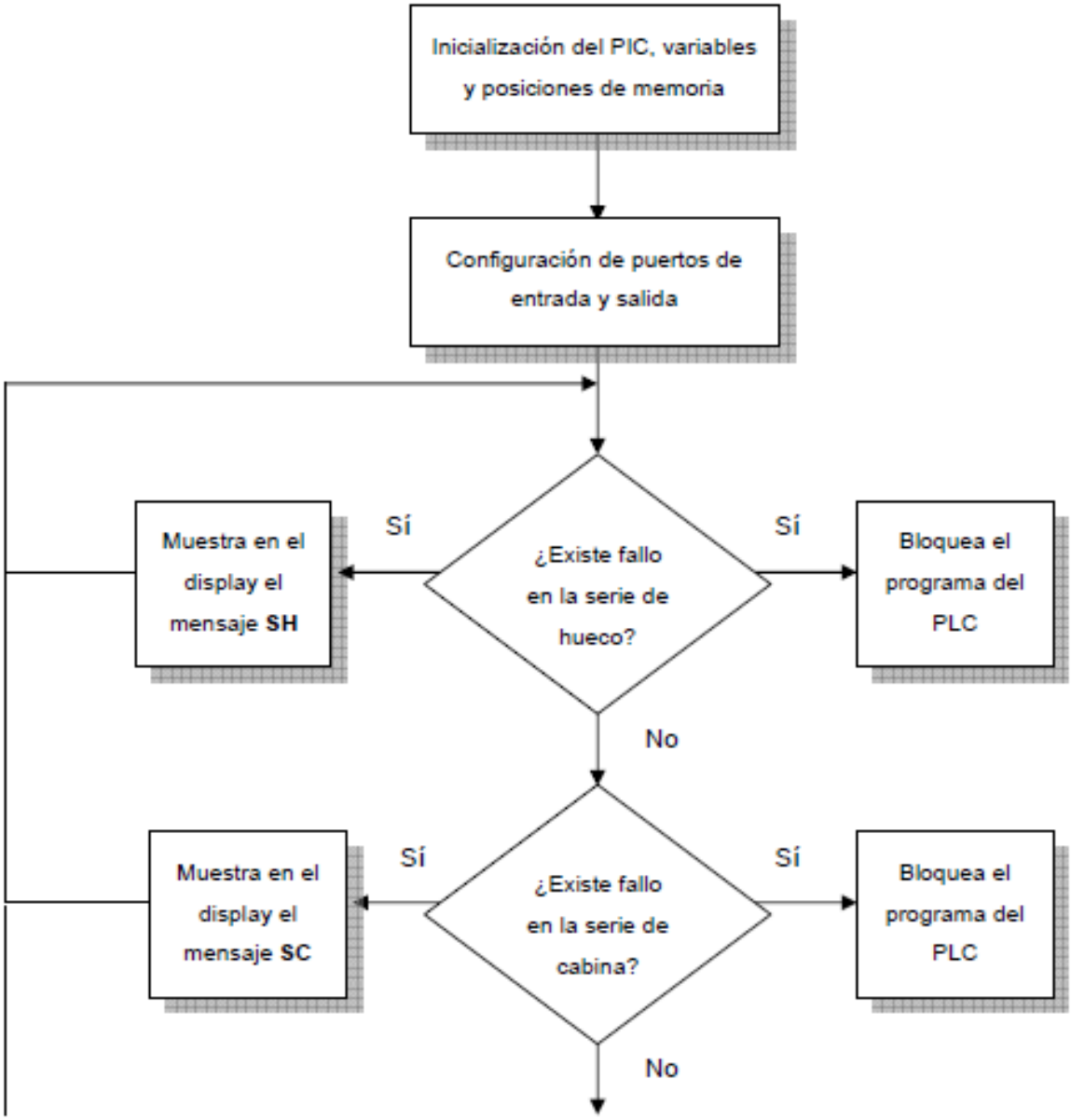
Ascensor, maniobra, Control, Siemens, PLC, Microchip, Micro controlador,

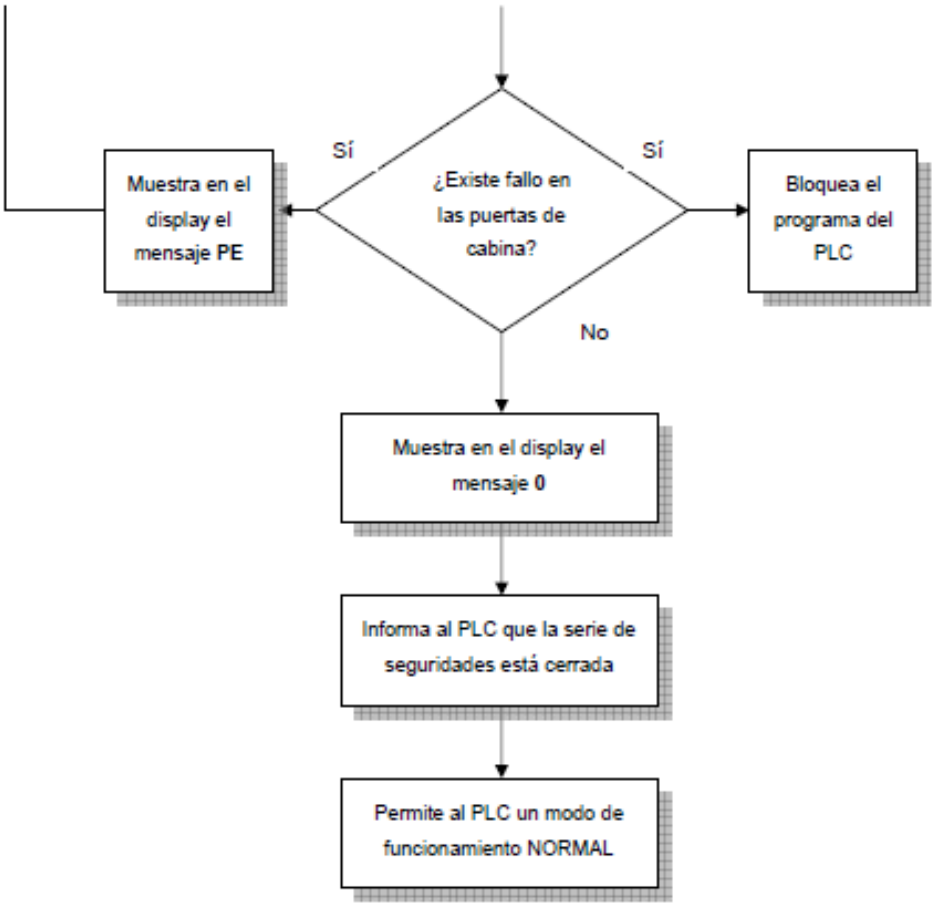
Clasificación y características de las instalaciones

Un ascensor o elevador, es un aparato que sirve para trasladar personas o cosas de unos niveles de Altura a otros. Se pueden clasificar según el tipo de tracción en electromecánicos o hidráulicos. A grandes rasgos, puedo establecer cuatro partes diferenciadas:

- Hueco: Es el espacio destinado en un edificio o estructura para ubicar el ascensor.
- Cuarto de máquinas: Es el local destinado a ubicar la máquina tractora, los dispositivos de Control, y todos los demás componentes que gobiernan el ascensor.
- Cabina: Plataforma cerrada o abierta que alberga la carga y que se desplaza a través del Hueco.
- Foso: Parte inferior del hueco del ascensor. La cabina debe acudir a cada altura cuando sea solicitado por un usuario desde la planta o desde el Interior.

Diagrama de flujo





Elementos que componen la maniobra

A continuación se listan los dispositivos y elementos que componen el diseño de la maniobra, según los esquemas eléctricos

Descripción

Elementos situados en cabina

Nombre	**Descripción**
KOPEN	Pulsador de reapertura de puertas en cabina
BARRERA	Barrera fotoeléctrica anti aplastamiento
LED1	Indicador de sobrecarga en cabina
Z1	Zumbador indicador de sobrecarga en cabina

LUZC	Iluminación de cabina
PULSC’5’	Pulsador en cabina de llamada a piso 5
PULSC’4’	Pulsador en cabina de llamada a piso 4
PULSC’3’	Pulsador en cabina de llamada a piso 3
PULSC’2’	Pulsador en cabina de llamada a piso 2
PULSC’1’	Pulsador en cabina de llamada a piso 1
PULSC’0’	Pulsador en cabina de llamada a piso 0
BCD2	Decodificador BCD-7 segmentos
DISPLAY2	Posicional indicador de piso en acceso cabina
POSUP	Sentipara de subida en cabina
POSDW	Sentipara de bajada en cabina
COM	Comunicador-marcador telefónico
LED2	Indicador de llamada del comunicador
LED3	Indicador de comunicación
ALT	Altavoz del comunicador
MIC	Micrófono del comunicador
ALM	Zumbador de alarma en cabina
KALM	Pulsador de alarma en cabina
EM	Dispositivos de control de la iluminación de emergencia
LUZEM	Iluminación de emergencia en cabina

Elementos situados en el hueco

Nombre	**Descripción**
KPUERTAS’n’	Contacto de cierre de puertas exteriores
PULSH’5’	Pulsador exterior de llamada a piso 5
PULSH’4’	Pulsador exterior de llamada a piso 4

PULSH’3’	Pulsador exterior de llamada a piso 3
PULSH’2’	Pulsador exterior de llamada a piso 2
PULSH’1’	Pulsador exterior de llamada a piso 1
PULSH’0’	Pulsador exterior de llamada a piso 0
KPUERTA’5’	Contacto de cierre de puertas exteriores en planta 5
KPUERTA’4’	Contacto de cierre de puertas exteriores en planta 4
KPUERTA’3’	Contacto de cierre de puertas exteriores en planta 3
KPUERTA’2’	Contacto de cierre de puertas exteriores en planta 2
KPUERTA’1’	Contacto de cierre de puertas exteriores en planta 1
KPUERTA’0’	Contacto de cierre de puertas exteriores en planta 0
LUZH	Luces de hueco. Rosario
POSUP	Sentí-para de subida en piso
POSDW	Sentí-para de bajada en piso
BCD1	Decodificador BCD-7 segmentos
DISPLAY1	Posicional indicador de piso en acceso principal

Elementos situados en techo de cabina

Nombre	**Descripción**
MOTORP	Motor de puertas
KAC1	Contacto de acatamiento
KFIN	Contacto final de carrera en subida y bajada
STOP2	Paro de seguridad en techo de cabina
KCERRAR	Contacto de cierre de puertas de cabina
INSPECCIÓN1	Botonera de funcionamiento en revisión en techo de cabina
PAROINSUP	Contacto de paro superior en inspección

PAROINSDW	Contacto de paro inferior en inspección
INS1	Conmutador inspección/normal en techo de cabina
PARADA1	Detector magnético de parada 1
PARADA2	Detector magnético de parada 2
IMPULSOUP	Detector magnético de impulso en subida
IMPULSODW	Detector magnético de impulso en bajada
PAROUP	Detector magnético de paro en extremos superior
PARODW	Detector magnético de paro en extremos inferior
KABRIR	Contacto límite de apertura de puertas
SENS	Contacto de sensibilidad de reapertura de puertas
PES	Dispositivo pesacartas
CTC	Cuadro de conexiones en techo de cabina
CM1	Cordón de maniobra 1
CM2	Cordón de maniobra 2
CMC	Cordón de maniobra del mandador

Descripción del diseño

A continuación explicaré el funcionamiento de la maniobra en función de los esquemas eléctricos Que se han diseñado y que se pueden comprobar en el anexo en el apartado anterior.

Nota: Por seguridad, reitero que se debe implementar en la fase de montaje que todas las partes Metálicas de la maniobra estén conectadas a tierra

El interruptor magneto térmico trifásico **MT1** acomete por una parte a los contacto res trifásicos que alimentan directamente al motor y por otra parte al bobinado primario del transformador multe-salida **TRAFO**. Mediante él, se puede desconectar la maniobra manualmente y asegurarnos que el motor no va a ser alimentado. Además protege contra sobretensiones. En tareas de mantenimiento, por seguridad, se deberá impedir cualquier operación del interruptor magneto

térmica mediante un útil de bloqueo un candado. El interruptor magneto térmico monofásico **MT2** acomete los circuitos de iluminación del ascensor. No es necesario impedir la activación de este dispositivo en tareas de mantenimiento, ya que su uso es independiente de un posible movimiento involuntario del ascensor, y además nos permite tener iluminados los trabajos.
El operador de puertas **(Fig. 1)** es el su circuito que controla la apertura y cierre de puertas de Cabina, y por arrastre, la apertura y cierre de las puertas exteriores. El dispositivo magneto térmico **MTP** permite anular la posibilidad de movimiento de puertas El relé **RE3** activa el motor de puertas (**MOTORP**), mientras que el relé **RE2** de doble circuito Permuta las dos fases de acometida del motor para efectuar los dos sentidos de giro de puertas, Apertura y cierre. Se deberá utilizar como operador mecánico de puertas uno que incluya un motor de tipo monofásico.

El contacto de la tensa limitador (**KAC3)** actúa si los cables del limitador de velocidad se cortan o se destensan. La polea inferior del limitador de velocidad se encuentra suspendida por su propio peso, si baja o Cae se acciona **KAC3**.El contacto de muelles (**KM**) actúa si la cabina se pasa de recorrido y se apoya en los muelles de Emergencia del foso.

STOP3 es un pulsador con enclavamiento (Seta) situado en el foso, accesible desde la luz de Puertas que permite al operario anular cualquier posibilidad de movimiento de la cabina. Si el grupo que forman los tres contactos de seguridad en foso **KAC3, KM** y **STOP3** se encuentran Cerrados, se ataca la entrada RB0 del micro controlador lo que le informa que la serie de hueco se encuentra correctamente. El contacto de acuña miento (**KAC1)** se activa cuando actúa el limitador de velocidad y la cabina se grapa a las guías impidiendo su movimiento incluso en caída libre. **FIN** se acciona mediante un resbalón si la cabina se pasa de recorrido tanto en el extremo superior como en el inferior, deteniendo eléctricamente de inmediato el ascensor.**STOP2** es un pulsador con enclavamiento (Seta) situado en el techo de cabina para el operario que realiza tareas de mantenimiento o montaje Si el grupo que forman los tres contactos de

seguridad en el techo de cabina **KAC1, KFIN** y **STOP2** se encuentran cerrados, se ataca la entrada RB1 del micro controlador lo que le informa que la serie de cabina se encuentra correctamente.

KAC2 es el contacto que detecta el disparo del limitador de velocidad en el cuarto de máquinas. Cuando se activa se ataca la entrada RB2 del micro controlador indicando de sobre velocidad de la cabina. **KTM** y **KTC** con termo sondas que indican que informan al micro controlador de sobre temperatura en el motor y en el cuadro de maniobras respectivamente. A tal efecto, informan al micro controlador a través de su entrada RB3 de apertura de la serie de térmicos.**STOP1** es un pulsador con enclavamiento (seta) que permite al operario de mantenimiento detener el ascensor manualmente en caso de emergencia desde el cuadro de maniobras.

KAF es un contacto colocado en el cuarto de máquinas dispuesto de tal forma que actúa cuando los cables de tracción de la máquina se cortan o se destensan, deteniendo inmediatamente el funcionamiento del ascensor.**STOP1** y **KAF** atacan al micro controlador por su entrada RB4 para informarle que la serie del cuarto de máquinas se encuentra correcta. **KPUERTAS'n'** es una serie de seguridades en si misma que conecta todas las puertas exteriores. En cuanto una de ellas sea abierta, se abre la serie. **KPUERTAS'n'** ataca la entrada RB5 del micro controlador y al entrada I0.0 del PLC Siemens.

Con **KCERRAR** se detecta si la puerta de cabina está cerrada y se ataca la entrada RB6 del micro controlador y la entrada I0.1 del PLC Siemens. Si la serie de seguridades completa se encuentra cerrada, se activa directamente (sin intervención del micro controlador ni del PLC) el contacto de seguridad **CC** que permite acometer el motor de tracción. A su vez, solo si la serie de seguridades se encuentra cerrada, se acometen las botoneras de Inspección que permitirán al operario mover el ascensor en revisión La botonera de mantenimiento

INSPECCIÓN 1 se encuentra en el techo de cabina mientras que la botonera **INSPECCIÓN 2** se encuentra en el cuadro de maniobras.

Las botoneras de mando en inspección permiten al operario de mantenimiento controlar el Movimiento del ascensor impidiendo la posibilidad de un funcionamiento en normal. Mediante los conmutadores **INS1** e **INS2** se ataca la entrada I0.2 del PLC Siemens permitiendo al ascensor un funcionamiento en normal o controlado por los mandos de inspección. Las botoneras disponen, además de un conmutador Inspección/Normal de un botón de subida, uno de bajada y un botón común a ambos. Los pulsadores de subida y bajada están enclavados eléctricamente para no permitir darle al elevador dos órdenes opuestas al mismo tiempo.

Las órdenes de subida y bajada de las botoneras de mando e inspección atacan directamente a los contacto res de velocidad lenta y a los contacto res de subir y bajar respectivamente. La botonera de inspección en techo de cabina dispone de unos finales de carrera tanto en subida o en bajada de tal forma que el ascensor se detenga unas decenas de centímetros antes de llegar al extremo superior e inferior.

De esa forma se previene que el operario pueda golpearse en la cabeza por un descuido o quedarse atrapado. Una manguera para pulsadores exteriores se distribuye a través del hueco del ascensor Transmitiendo al PLC siemens si alguien ha pulsado un botón para que acuda el elevador. Para dichas señales se utilizan las entradas de la I1.0 a la I1.5.

Esquemas eléctricos

Indicadores de movimiento del ascensor tanto en subida como en bajada se deberán colocar en cada planta. La activación y desactivación de los mismos está controlada por el PLC Siemens en sus canales Q0.5 para la subida y Q0.6 para la bajada. Un display de 7 segmentos se puede colocar en la planta principal o en las

plantas en las que se desee para informar de la ubicación actual del ascensor. Se utiliza un decodificador BCD-7 segmentos para traducir las señales del PLC Siemens a través de sus canales Q0.0, Q0.1, Q0.2 y Q0.3

El punto del display de 7 segmentos se utilizará para marcar plantas negativas, como por ejemplo garajes o subterráneos. Para posicionar el ascensor se utilizan detectores magnéticos tal como se detalla en el apartado 2.5.Estos detectores magnéticos ofrecen un comportamiento inestable cerrando su circuito interno en presencia de un campo magnético generado por un imán y abriéndolo a su salida. La suma de los detectores **PARADA1** y **PARADA2** atacan a la entrada I0.3 del PLC Siemens.

El detector de impulsos en subida (**IMPULSOUP**) ataca la entrada I0.4 del PLC. El detector de impulsos en bajada (**IMPULSODW**) ataca con 24VDC la entrada I0.5 del PLC. Los detectores de paro en subida y bajada (**PAROUP** y **PARODW**) atacan a 24VDC las entradas I0.6 e I0.7 del PLC siemens respectivamente. Varios dispositivos activan la entrada I1.7 del PLC Siemens enviándole una señal de reapertura de puertas: **KOPEN** es un pulsador que se encuentra en el mandador de cabina y permite al usuario reabrir puertas cuando están cerrando.

BARRERA es una barrera fotoeléctrica que impide que un usuario del ascensor quede atrapado cuando las puertas están cerrando. **SENS** son una serie de dispositivos que incorporan la mayoría de los operadores mecánicos de puertas del mercado que se activa cuando las puertas de cabina a su cierre encuentran un obstáculo o se intenta abrir forzadamente las mismas. **ES** un dispositivo pesador de cargas que colocado bajo la cabina envía una señal de sobrecarga que impedirá al ascensor cerrar puertas de cabina y por lo tanto iniciar el movimiento.

A su vez la señal del pesacartas alimenta un zumbador y un indicador luminoso que indica a los usuarios que se debe aligerar peso. Todos estos dispositivos activan la señal de reapertura del PLC Siemens reabriendo puertas en el caso de

que estén cerrando y por lo tanto impidiendo cualquier marcha del ascensor hasta que se solucione la causa. Existirá una luz en el interior de la cabina (**LUZC**) de iluminación suficiente según normativa. Los pulsadores de cabina (**PULSC'n'**) actúan de la misma forma que los pulsadores exteriores y acometen las mismas entradas del PLC Siemens.

Un display de 7 segmentos se colocará en el interior de cabina para informar de la ubicación actual del ascensor. Se utiliza un decodificador BCD-7 segmentos para traducir las señales del PLC Siemens a través de sus canales Q0.0, Q0.1, Q0.2 y Q0.3El punto del display de 7 segmentos se utilizará para marcar plantas negativas, como por ejemplo garajes o subterráneos. Indicadores de movimiento del ascensor tanto en subida como en bajada se deberán colocar en el interior de cabina. La activación y desactivación de los mismos está controlada por el PLC Siemens en sus canales Q0.5 para la subida y Q0.6 para la bajada.

- La salida del PLC Siemens Q1.0 activa el contacto de subida
- La salida del PLC Siemens Q1.1 activa el contacto de bajada
- La salida del PLC Siemens Q1.2 activa el contacto de velocidad rápida
- La salida del PLC Siemens Q1.3 activa el contacto de velocidad lenta
- La salida del PLC Siemens Q1.4 activa el relé de activación de freno
- La salida del PLC Siemens Q1.6 activa el relé de selección de giro de puertas
- La salida del PLC Siemens Q1.5 activa el relé de activación de puertas
- La salida RA1 del micro controlador activa la entrada I0.6 del PLC Siemens, es decir, si el micro controlador no encuentra fallo en la serie de seguridades del ascensor informa al PLC. Todas las entradas de contacto res y relés están protegidas por diodos supresores de picos de tensión.

Corrección en bajada

Se ha diseñado la maniobra para conseguir un control con viaje de corrección en bajada, esto quiere decir que el ascensor en caso de perder en memoria su

posición actual, por un fallo o por un corte en el suministro de corriente, desciende automáticamente hasta la planta del extremo inferior y se resetea, colocando todas sus variables de posición a 0, realizando un retardo, y esperando una nueva llamada a piso.

El sistema de posicionamiento

Se ha diseñado un sistema de posicionamiento sin contacto, que informa a la CPU en todo momento de la posición del ascensor. Detectores magnéticos monoestables colocados en el techo de la cabina leen durante un viaje la posición de imanes colocados fijos en las guías y en soportes fijos. Existirá un imán de parada (PARADA1 y PARADA2) en todos los accesos que, con la cabina a nivel de planta, coincidirá exactamente con la posición de los detectores, de la forma descrita en la imagen adjunta. El imán debe medir al menos 150 mm de longitud, y la longitud del mismo debe coincidir Exactamente con la distancia de separación de sus dos detectores de parada.

Deberá colocarse un imán (IMPULSODW) de unos 50 mm, aproximadamente a unos 70 cm por encima del imán de parada, y otro a la misma distancia por debajo (IMPULSOUP). El paso por estos imanes marcará la entrada de la velocidad lenta y por lo tanto, la parada de la cabina al llegar al siguiente imán de parada.

Entradas y salidas de PLC y del micro controlador

IO	**Descripción PLC**
I0.0	Activado indica que la serie de puertas exteriores está cerrada.
I0.1	Activado indica que las puertas de cabina están cerradas.
I0.2	Activado indica orden de funcionamiento en modo NORMAL. Desactivado indica orden de funcionamiento en INSPECCION y el PLC no da órdenes a la maniobra.

	Detectores de parada 1 y parada 2 activados
I0.3	Detector de impulso de subida activado
I0.4	Detector de impulso de bajada activado
I0.5	Detector de paro en extremos superior activado
I0.6	Detector de paro en extremos inferior activado
I0.7	
	Activado un pulsador de llamada a piso 0
I1.0	Activado un pulsador de llamada a piso 1
I1.1	Activado un pulsador de llamada a piso 2
I1.2	Activado un pulsador de llamada a piso 3
I1.3	Activado un pulsador de llamada a piso 4
I1.4	Activado un pulsador de llamada a piso 5
I1.5	Puertas de cabina completamente abiertas
I1.6	Reapertura de puerta activada
I1.7	
I2.0	Activado indica que la serie completa de seguridades está cerrada (Procede de la salida del Micro controlador RBO)
	No utilizado
I2.1	No utilizado
I2.2	No utilizado
I2.3	No utilizado
I2.4	No utilizado
	No utilizado

I2.5	No utilizado
I2.6	
I2.7	

Salidas de PLC

IO PLC	**Descripción**
Q0.0	Activa el bit 0 de los decodificadores BCD
Q0.1	Activa el bit 1 de los decodificadores BCD
Q0.2	Activa el bit 2 de los decodificadores BCD
Q0.3	Activa el bit 3 de los decodificadores BCD
Q0.4	No utilizado
Q0.5	Activa los sentí paras de subida
Q0.6	Activa los sentí paras de bajada
Q0.7	No utilizado
Q1.0	Orden de subida
Q1.1	Orden de bajada
Q1.2	Orden de velocidad rápida´
Q1.3	Orden de velocidad lenta
Q1.4	Orden de apertura de freno
Q1.5	Activado indica que las puertas de cabina se deben mover en dirección de cierre, desactivado indica que las puertas de cabina deben moverse en dirección de apertura
Q1.6	Orden activación del motor de puertas No utilizado

Q1.7

Programa del PLC Siemens S7-200

El programa principal de control del PLC Siemens está generado con el software propio del Autómata STEP 7 Micro Win versión 32 Para dibujar el programa he elegido el editor KOP por parecerme el de más sencillo uso y el que se Adapta mejor a un diseño del programa basado en Grafcets de control. El S7-200 ejecuta cíclicamente la lógica de control del programa, leyendo y escribiendo datos. Cuando un programa se carga en la CPU y ésta se pone en modo RUN, la CPU ejecuta el programa en el siguiente orden:

- El S7-200 lee el estado de las entradas.
- El programa almacenado en el S7-200 utiliza las entradas para evaluar (o ejecutar) la Lógica.
- Tras evaluar el programa, el S7-200 almacena los resultados de la lógica en el área de Salidas, es decir, es decir, en la imagen del proceso de las salidas.
- Al final del programa, el S7-200 escribe los datos de la imagen del proceso de las salidas en las salidas físicas.
- El ciclo de tareas se repite.

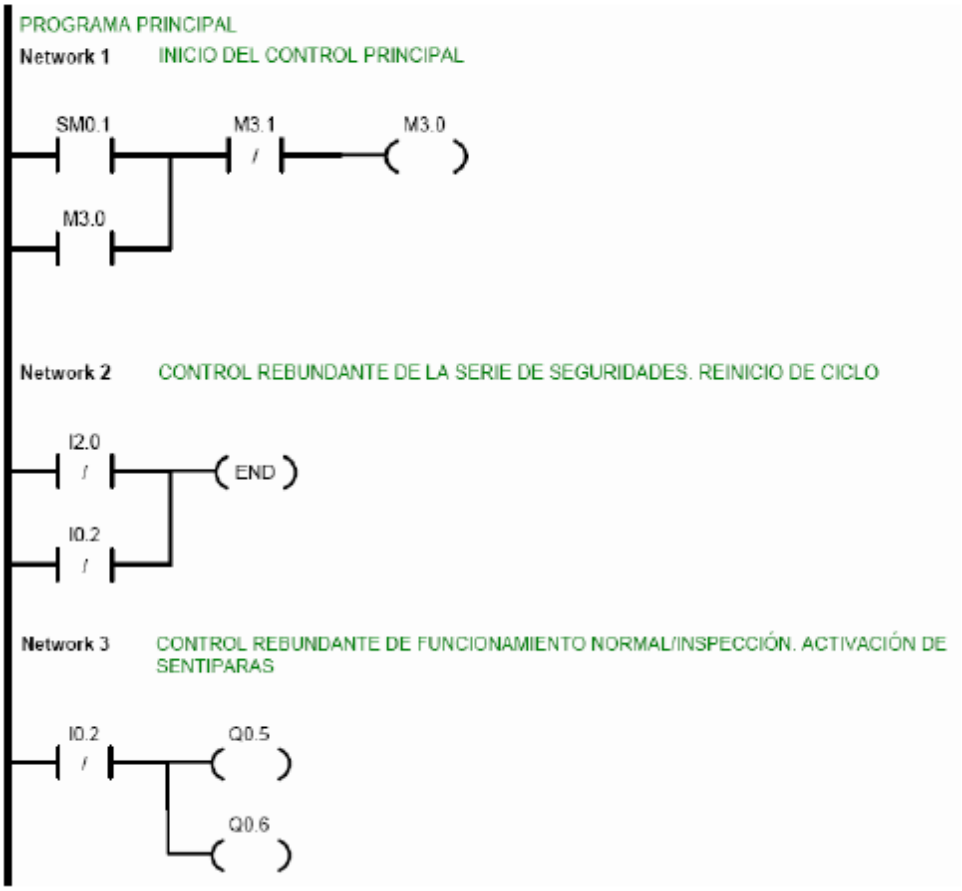
Programa principal

En prácticamente todas las transiciones se comprueba el estado de la serie de seguridades y del modo de funcionamiento “Normal” del ascensor. Cuando se activa una orden de subida o de bajada, ya sea en velocidad rápida o lenta, se activa el sentirla correspondiente de subida o de bajada en el exterior de todas las plantas y en cabina. En todos los ciclos de ejecución del programa del PLC se

transfiere el valor de la planta actual del ascensor a los decodificadores BCD que controlan los posicionales colocados en el exterior de las plantas y en cabina.

Programa del PLC

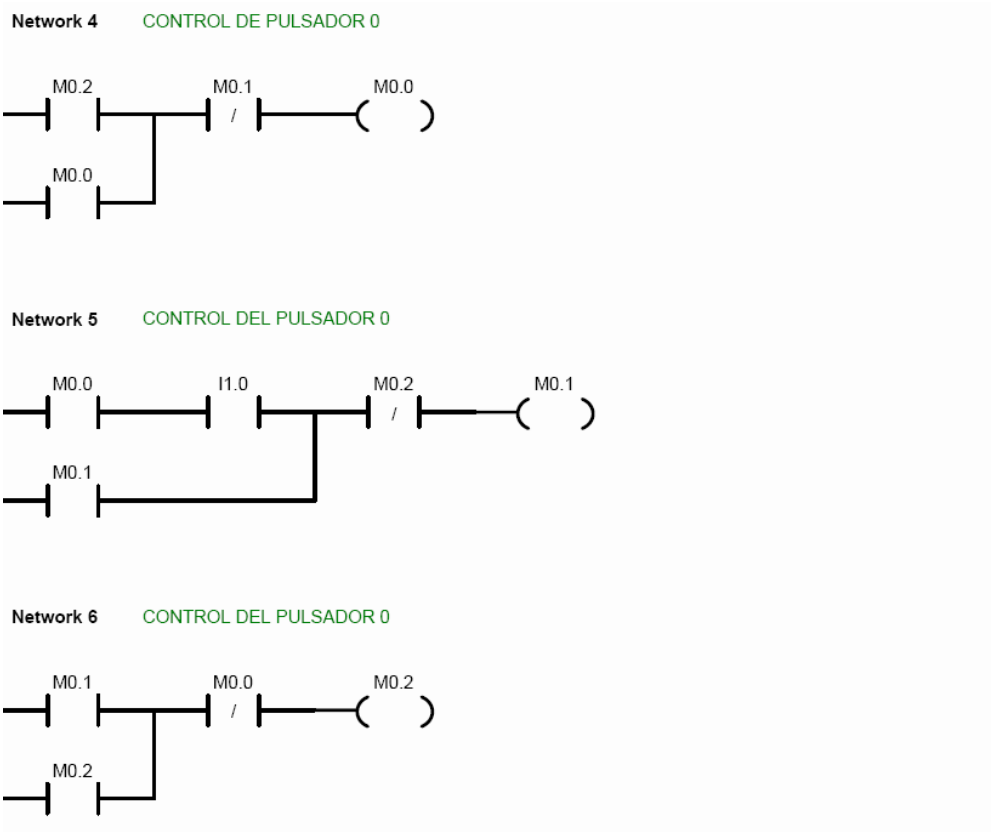
Inicio del control



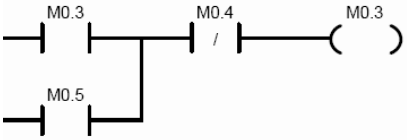
La marca especial SM0.1 inicializa el programa de control, activándose la misma al comienzo del primer ciclo del PLC. (Segmento 1) Aunque en todas las transiciones del programa se controla si la serie de seguridades está cerrada y el ascensor funcionando en modo normal, se crea una doble seguridad en el

segmento 2 en que un fallo en cualquiera de los dos dispositivos hace detener al ascensor y reiniciar el ciclo. (Segmento 2)Si el ascensor se coloca en modo de funcionamiento de Inspección, el PLC se coloca en modo de “espera” y se activan a la vez el sentirla de subida y el sentirla de bajada para informar a usuarios y operarios de montaje y mantenimiento. (Segmento 3)

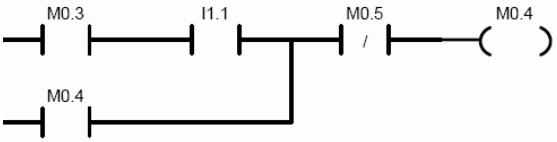
CONTROL DE PULSADORES



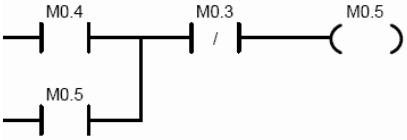
Network 7 CONTROL DE PULSADOR 1



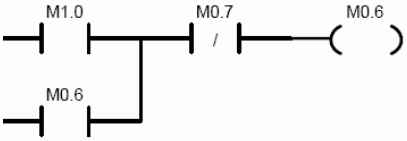
Network 8 CONTROL DEL PULSADOR 1



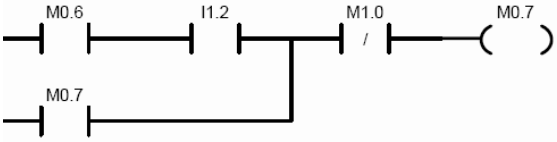
Network 9 CONTROL DEL PULSADOR 1



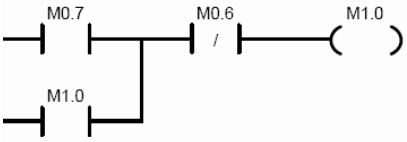
Network 10 CONTROL DE PULSADOR 2



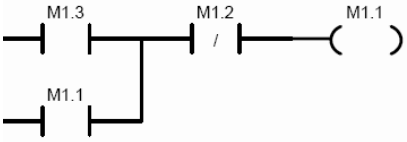
Network 11 CONTROL DEL PULSADOR 2



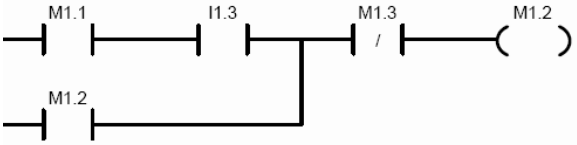
Network 12 CONTROL DEL PULSADOR 2



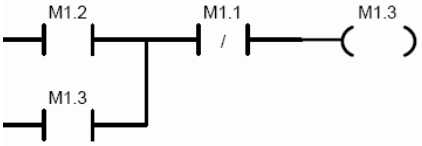
Network 13 CONTROL DE PULSADOR 3



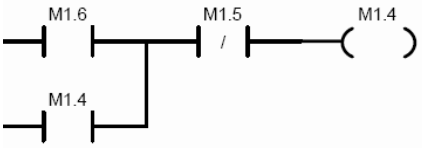
Network 14 CONTROL DEL PULSADOR 3



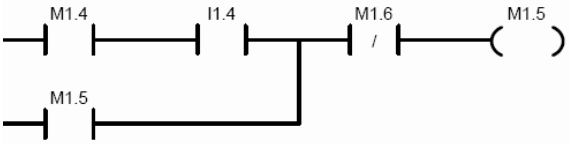
Network 15 CONTROL DEL PULSADOR 3



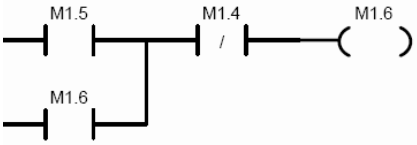
Network 16 CONTROL DE PULSADOR 4



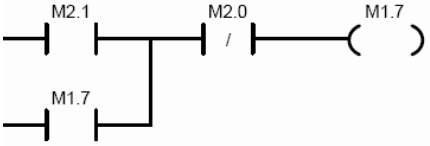
Network 17 CONTROL DEL PULSADOR 4



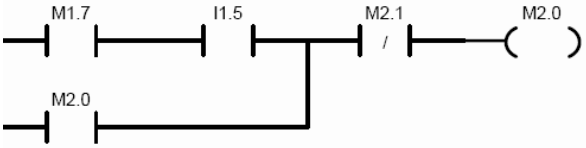
Network 18 CONTROL DEL PULSADOR 4



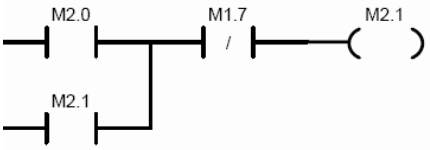
Network 19 CONTROL DE PULSADOR 5



Network 20 CONTROL DEL PULSADOR 5

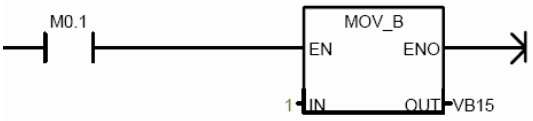


Network 21 CONTROL DEL PULSADOR 5

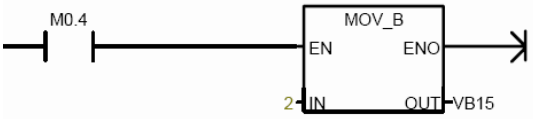


Acciones del control de pulsadores

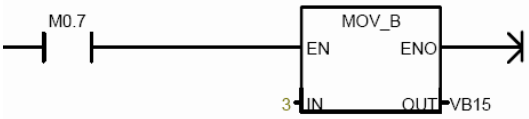
Network 22 ACCIONES DEL PULSADOR 0



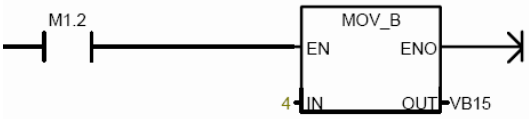
Network 23 ACCIONES DEL PULSADOR 1



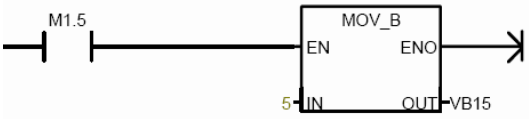
Network 24 ACCIONES DEL PULSADOR 2



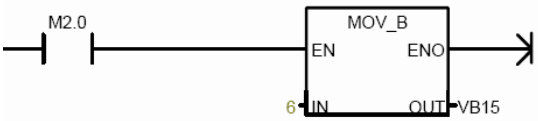
Network 25 ACCIONES DEL PULSADOR 3



Network 26 ACCIONES DEL PULSADOR 4

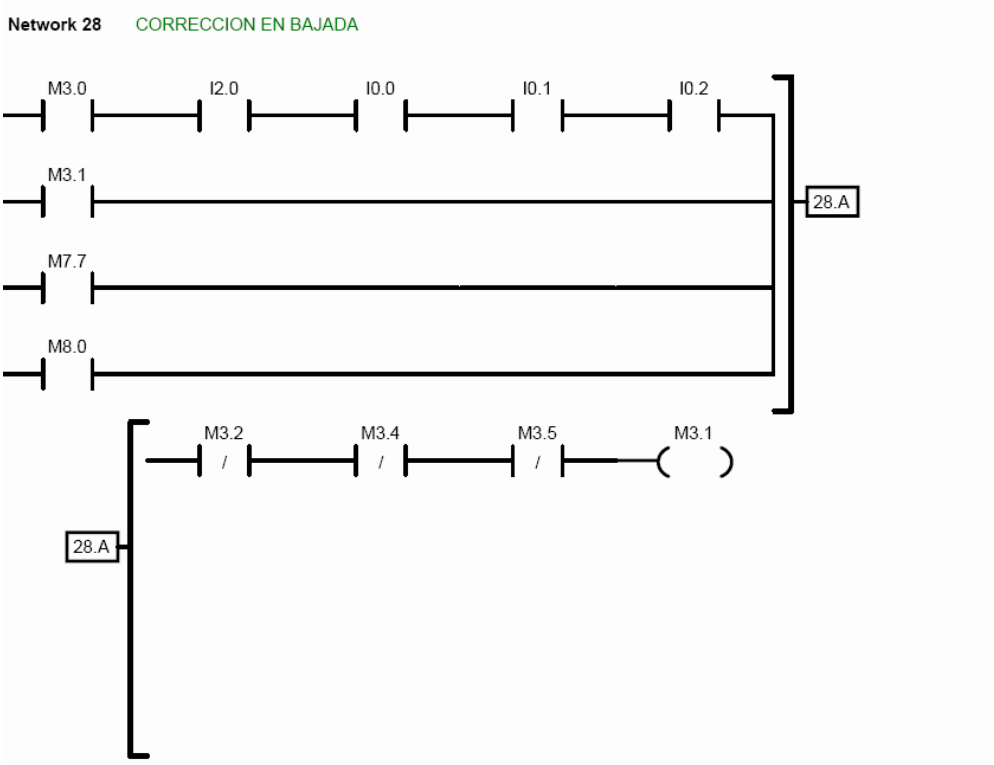


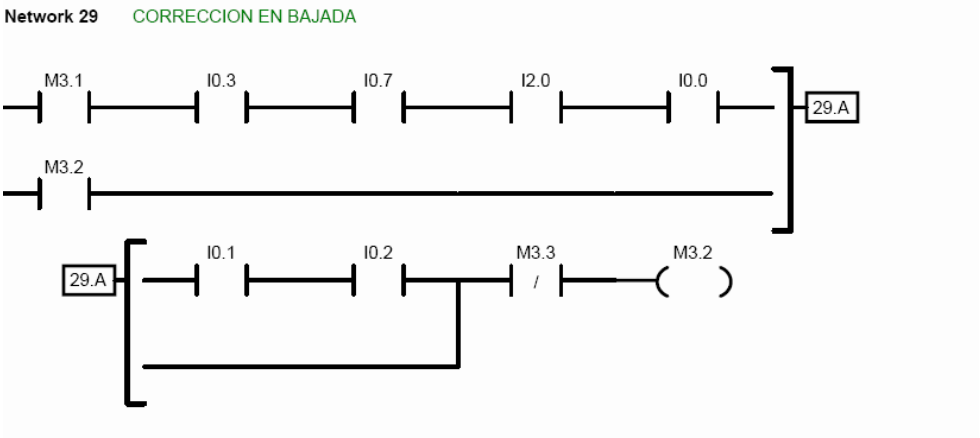
Network 27 ACCIONES DEL PULSADOR 5



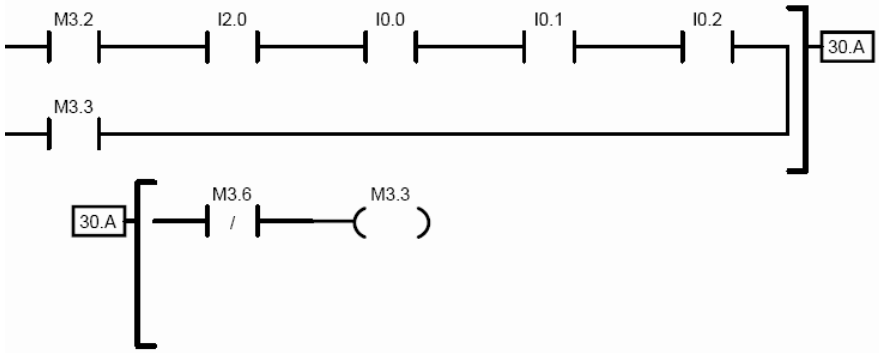
Cuando se acciona un pulsador de llamada a planta (tanto exterior como de cabina) se asigna a la variable VB15 de tipo “Byte” el valor de la planta asignada.

Corrección en bajada.

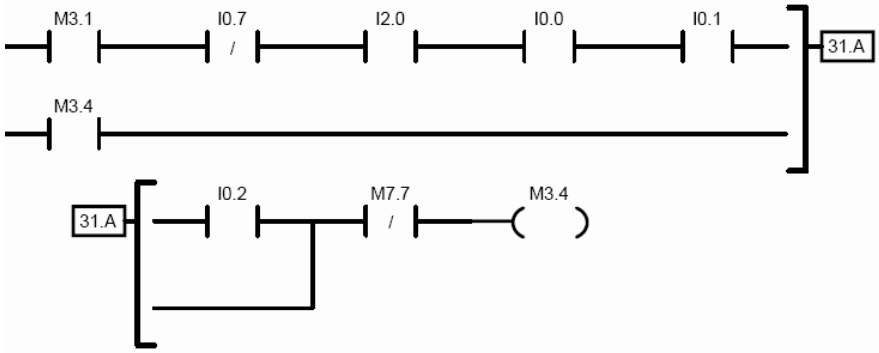




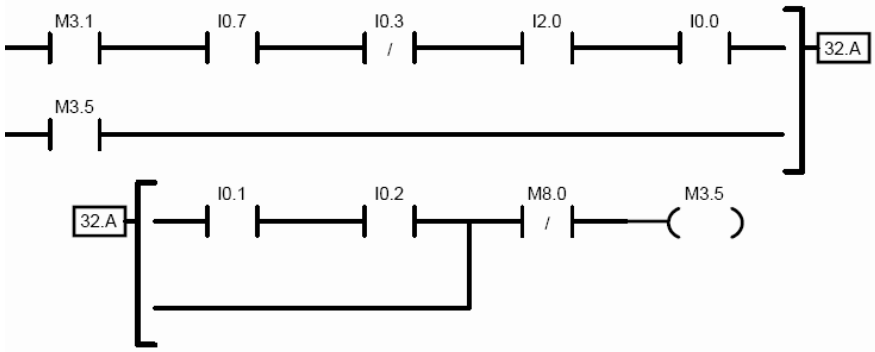
Network 30 CORRECCION EN BAJADA



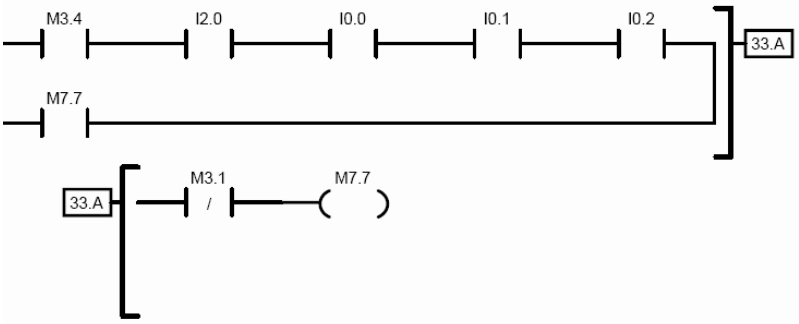
Network 31 CORRECCION EN BAJADA



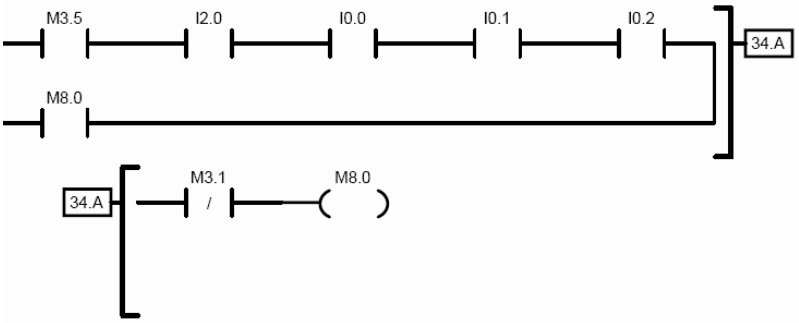
Network 32 CORRECCION EN BAJADA

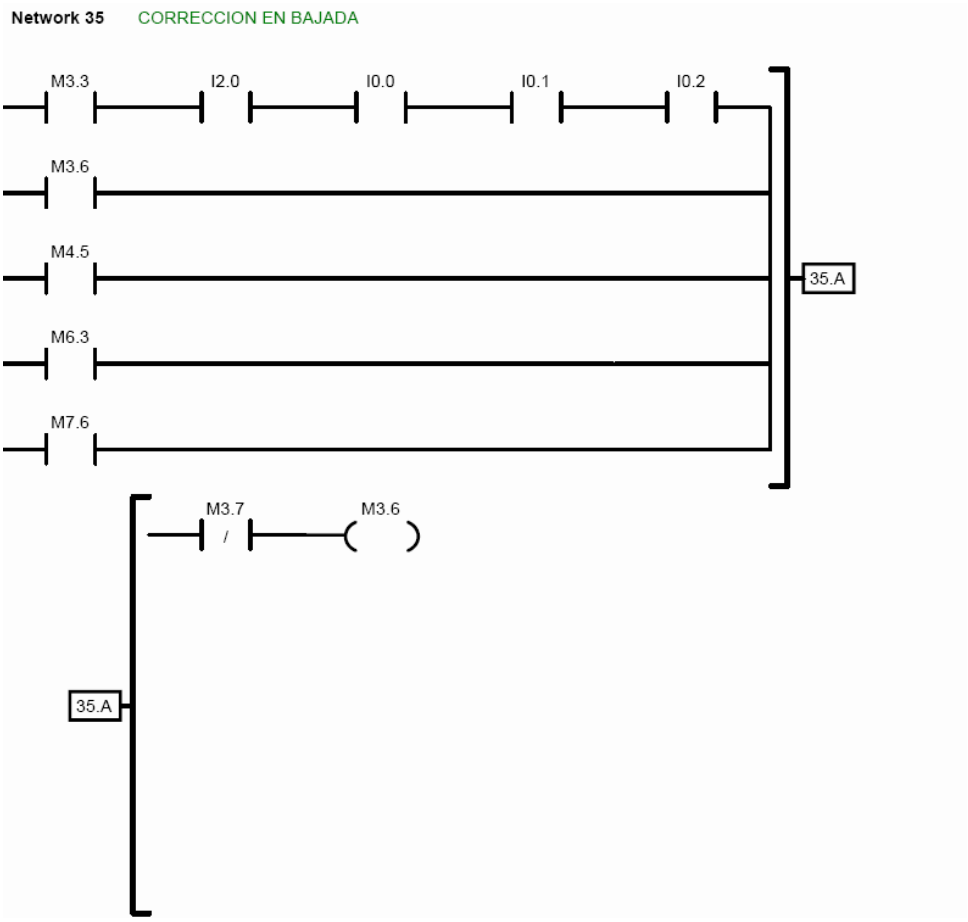


Network 33 CORRECCION EN BAJADA

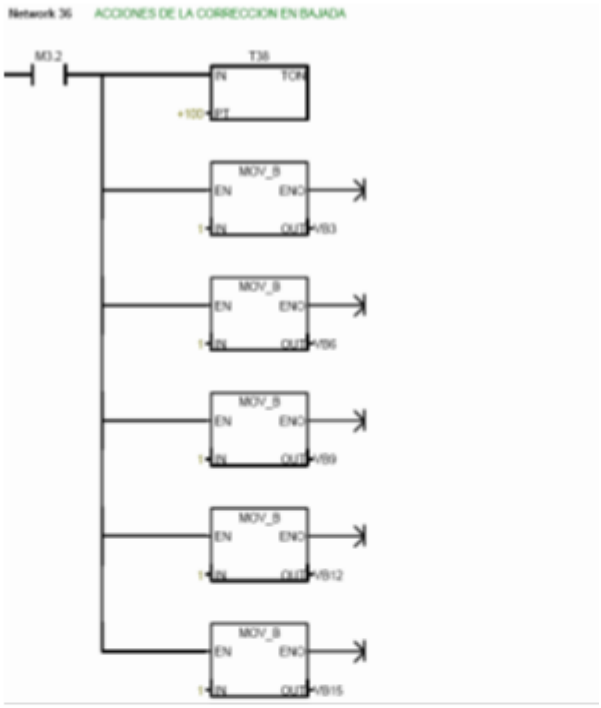


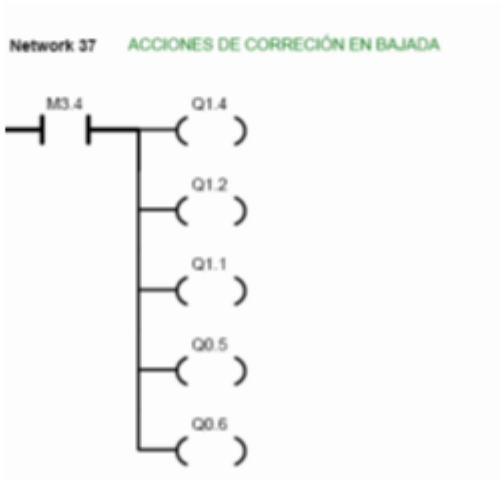
Network 34 CORRECCION EN BAJADA

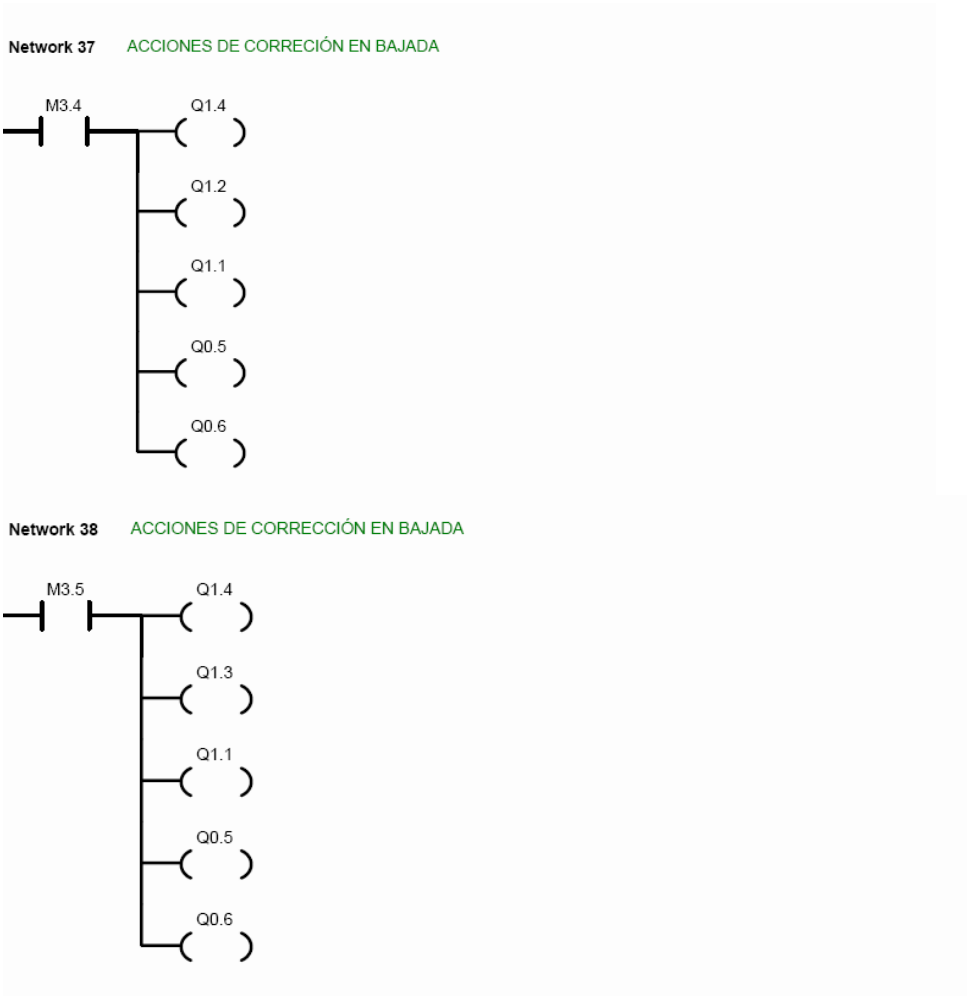



Network 35
CORRECCION EN BAJADA
M3.3
I2.0
I0.0
I0.1
I0.2
M3.6
M4.5
M6.3
M7.6
35.A
M3.7
/
M3.6
35.A

Acciones de la corrección en bajada

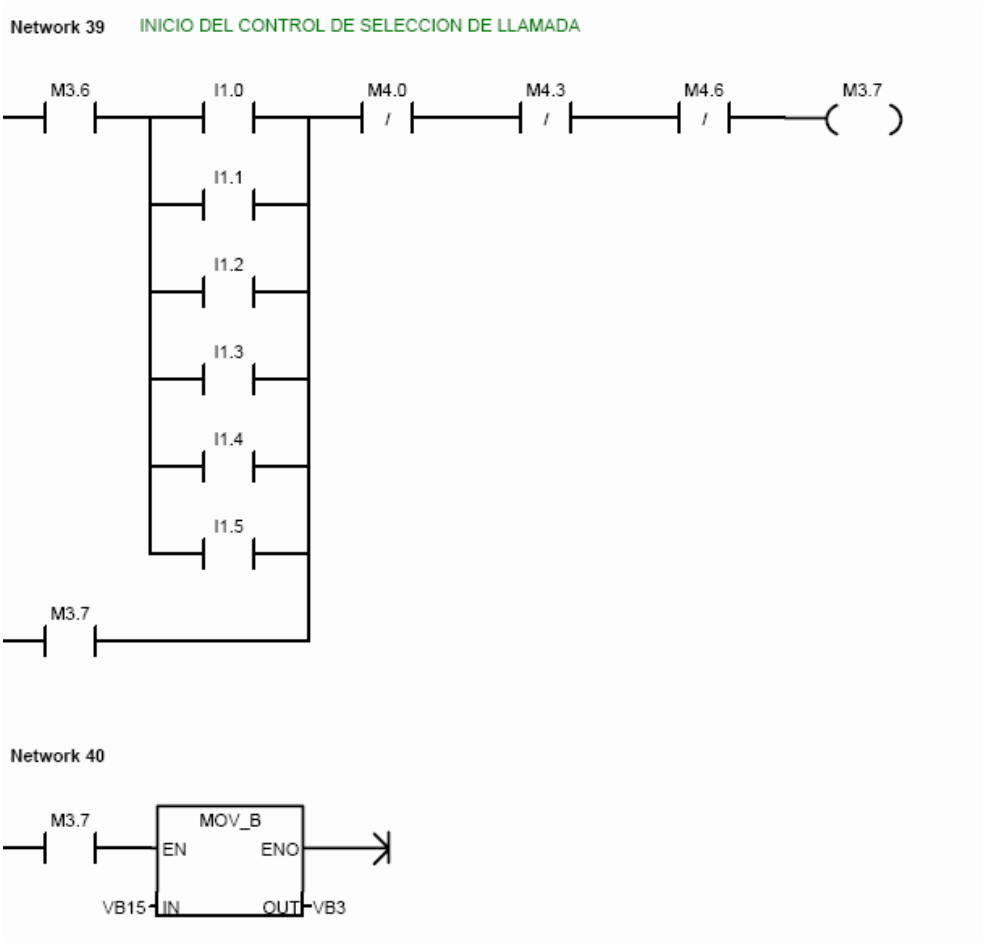






Al arrancar el programa de control del PLC ya sea en por su primera puesta en “RUN”, en caso de corte de corriente o por un error, se ejecuta una corrección en bajada. El ascensor desciende en rápida, hasta activarse el extremo inferior, en que comienza a descender pero esta vez en velocidad lenta. Cuando detecta la parada del piso extremo inferior, el programa resetea todas las variables y realiza una espera de 10 segundos. (Segmento 36)

Inicio del control de selección de llamada



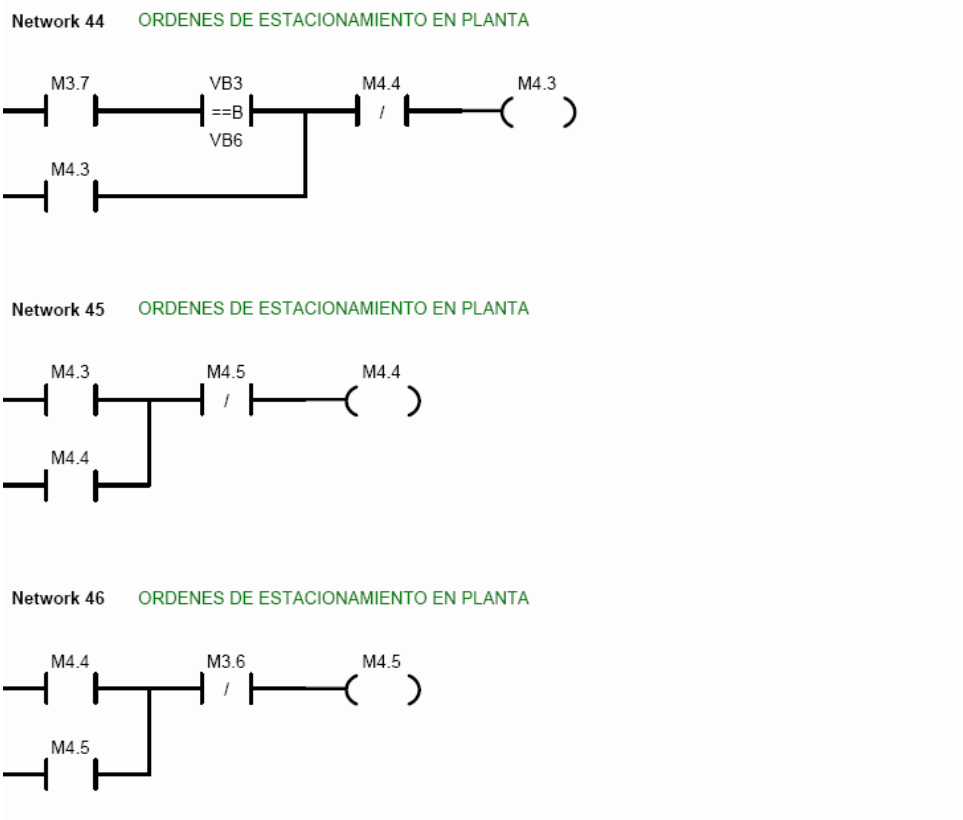
Cuando el ascensor está en reposo, espera a que un pulsador de llamada se accionado. Únicamente si el ascensor está en modo de reposo se guarda en memoria el valor del pulsador de Llamada activado, de esta forma se impide que el ascensor reciba nuevos datos mientras está Haciendo un viaje y por lo tanto se prioricen las últimas llamadas en vez de las primeras.

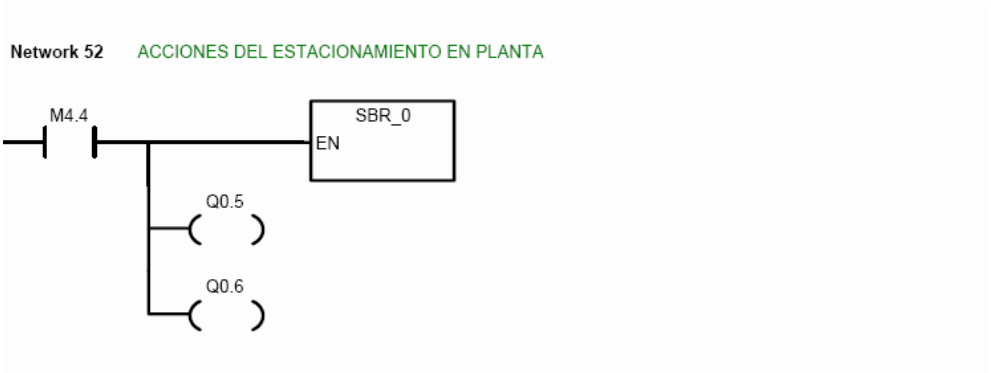
Si se solicita llamada a una planta superior a la actual

Si la llamada se produce a un piso superior a la posición actual de la cabina, se comparan ambas variables tipo “byte” (segmento 41) y se ejecuta la orden de subida.

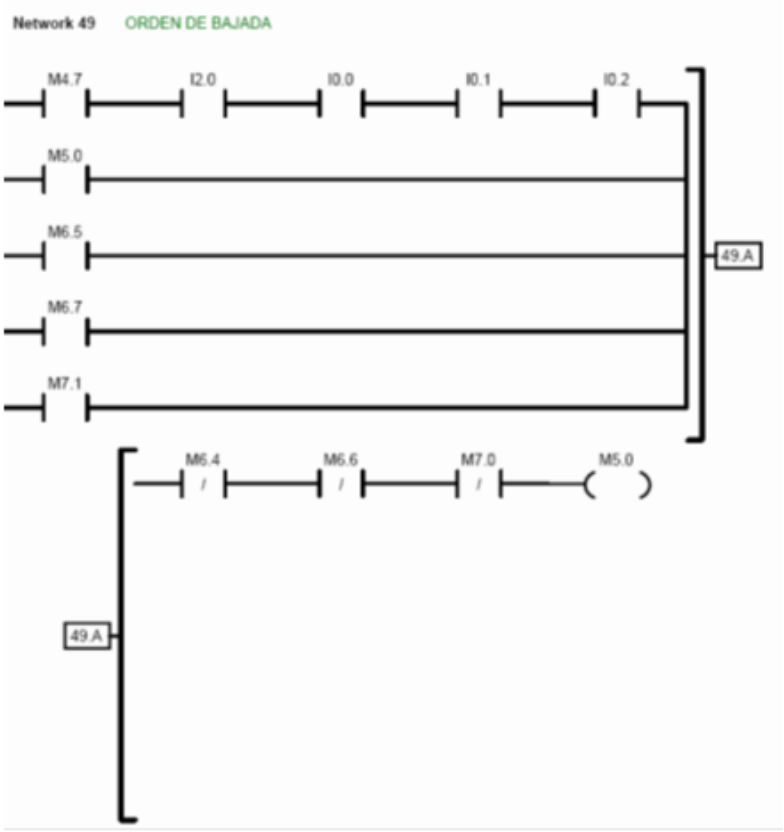
El ascensor comienza a subir en velocidad rápida.

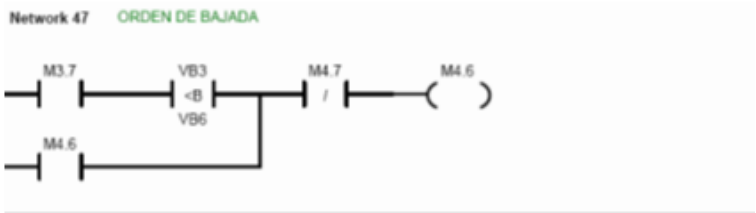
Si el ascensor se detiene en la planta solicitada o se efectúa una llamada a la misma planta donde se encuentra el ascensor

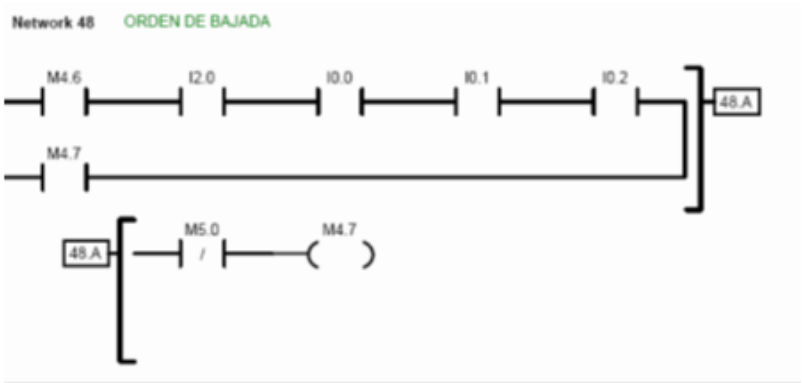


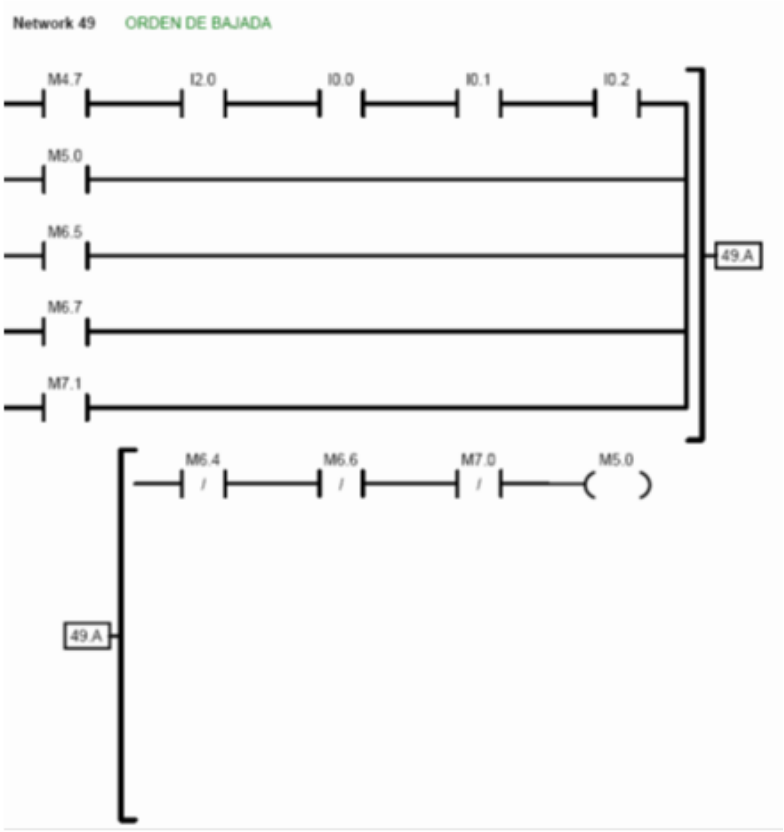


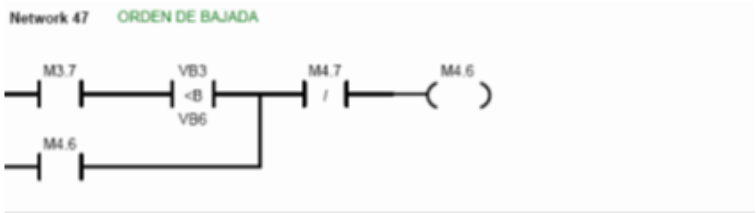
Esta parte del código se ejecuta en dos ocasiones, cuando el ascensor se detiene a nivel de planta que el usuario ha solicitado o si se efectúa una llamada en el mismo piso donde se encuentra el ascensor. Se comparan ambas variables tipo “byte” (segmento 44).Se inicializa la subrutina de apertura y cierre de puertas y se activan a la vez los sentirlas de subida y de bajada para marcar el estacionamiento e informar a los usuarios.

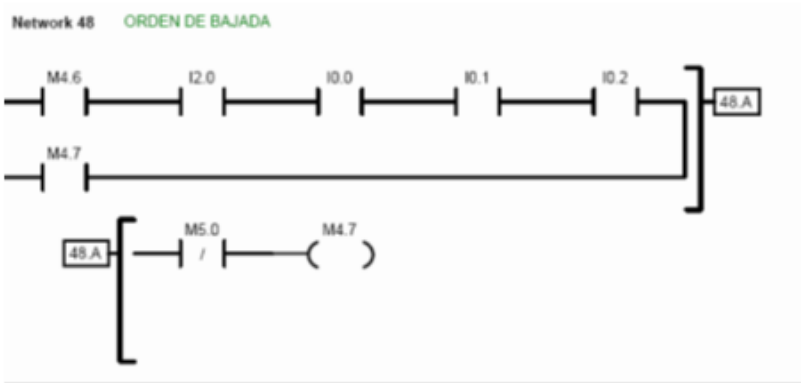

Network 49
ORDEN DE BAJADA
M4.7
I2.0
I0.0
I0.1
I0.2
M5.0
M6.5
M6.7
M7.1
49.A
M6.4
M6.6
M7.0
M5.0
49.A

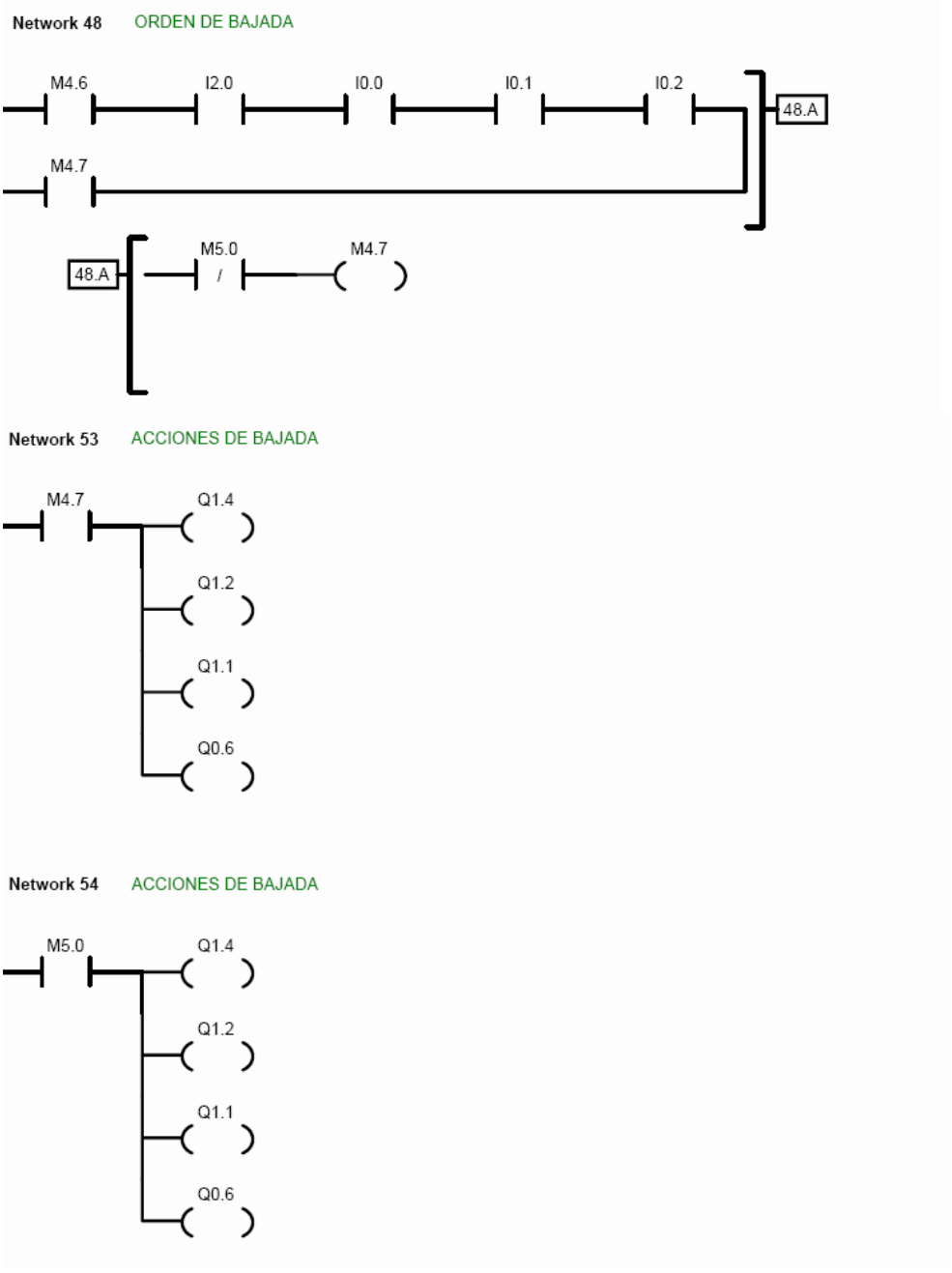

Network 47
ORDEN DE BAJADA
M3.7
VB3
<B
VB6
M4.7
M4.6
M4.6


Network 48
ORDEN DE BAJADA
M4.6
I2.0
I0.0
I0.1
I0.2
48.A
M4.7
48.A
M5.0
M4.7


Network 49
ORDEN DE BAJADA
M4.7
I2.0
I0.0
I0.1
I0.2
M5.0
M6.5
M6.7
M7.1
49.A
M6.4
M6.6
M7.0
M5.0
49.A

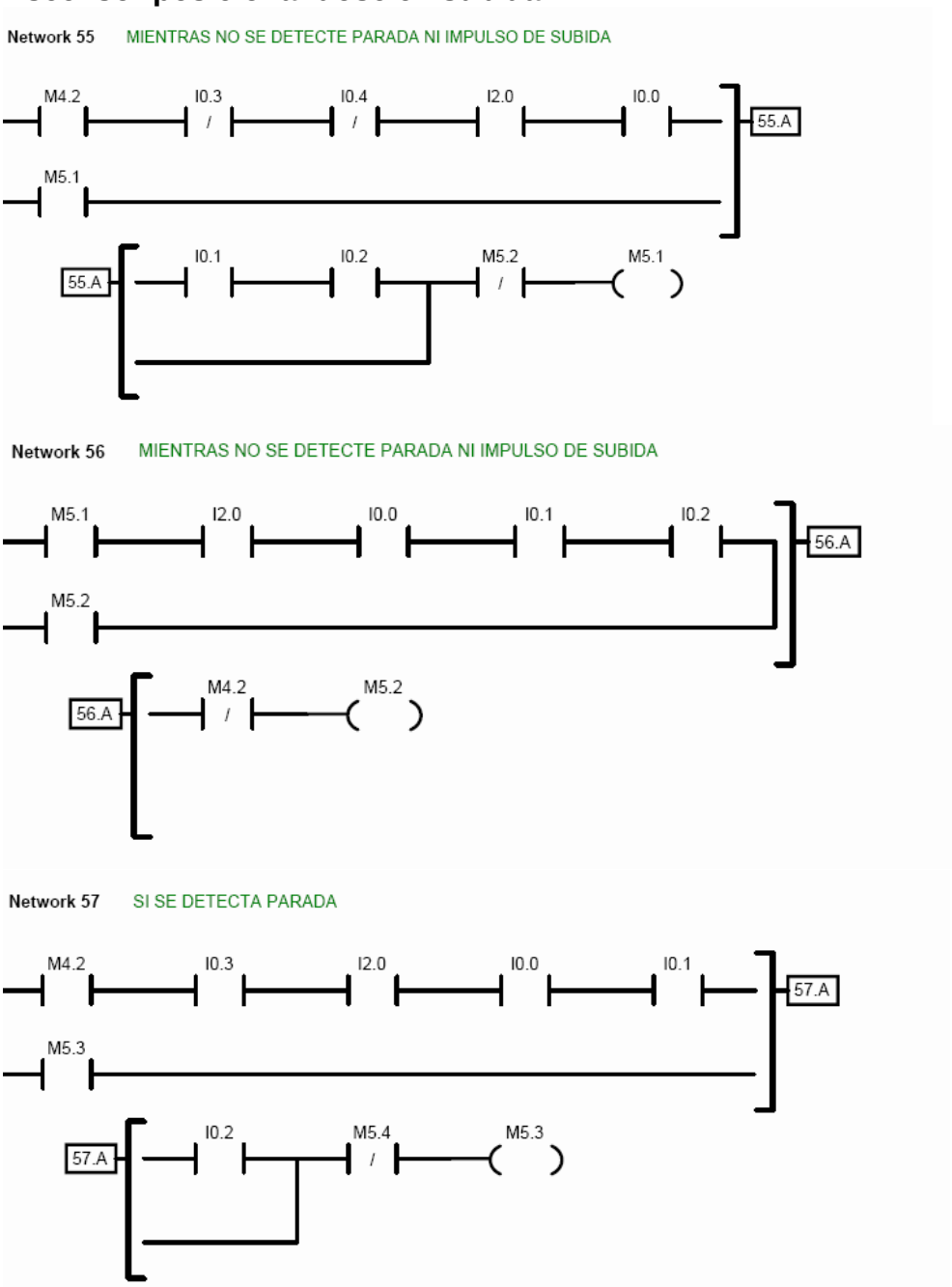

Network 47
ORDEN DE BAJADA
M3.7
VB3
<B
VB6
M4.7
M4.6
M4.6


Network 48
ORDEN DE BAJADA
M4.6
I2.0
I0.0
I0.1
I0.2
48.A
M4.7
48.A
M5.0
M4.7

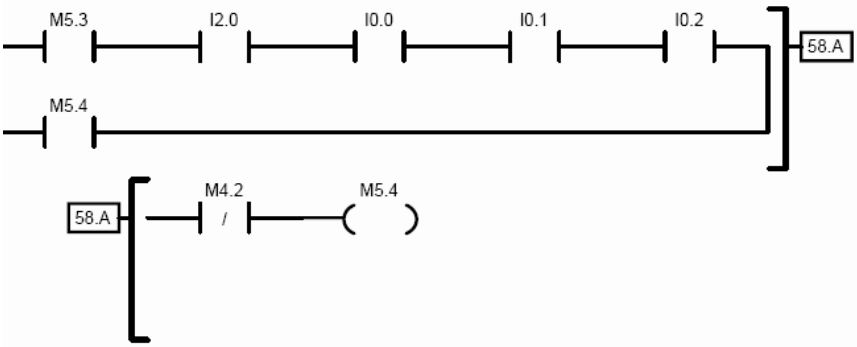


Si la llamada se produce a un piso inferior a la posición actual de la cabina, se comparan ambas variables tipo “byte” (segmento 47) y se ejecuta la orden de bajada. El ascensor comienza a bajar en velocidad rápida.

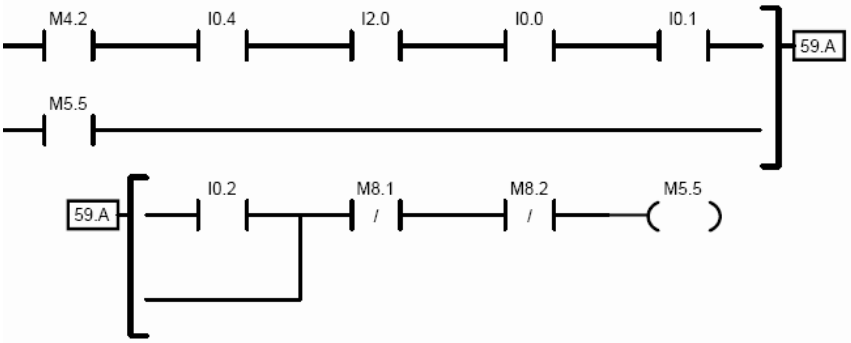
Ascensor posicionándose en subida



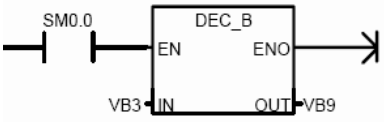
Network 58 SI SE DETECTA PARADA



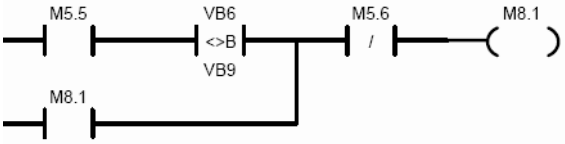
Network 59 SI SE DETECTA IMPULSO DE SUBIDA



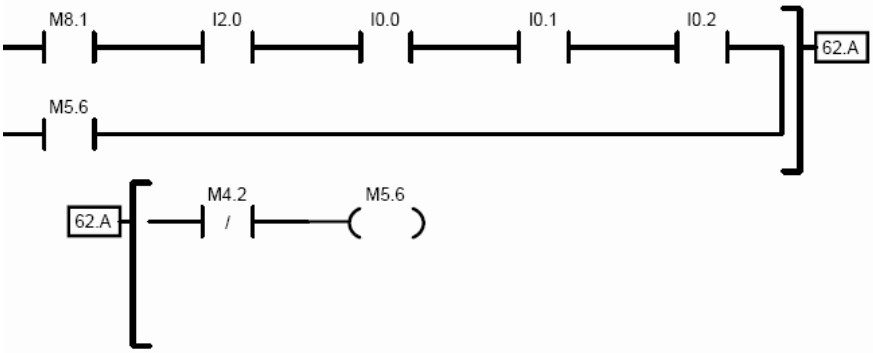
Network 60 SI SE DETECTA IMPULSO DE SUBIDA



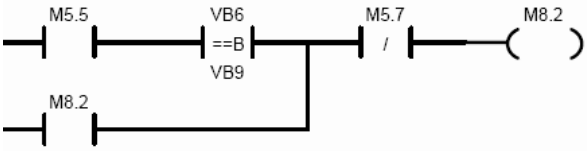
Network 61 SI SE DETECTA IMPULSO DE SUBIDA



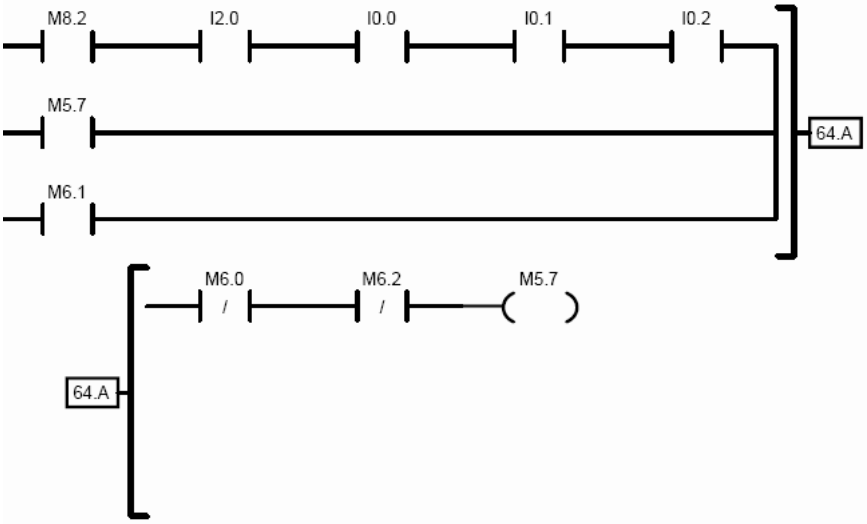
Network 62 SI SE DETECTA IMPULSO DE SUBIDA



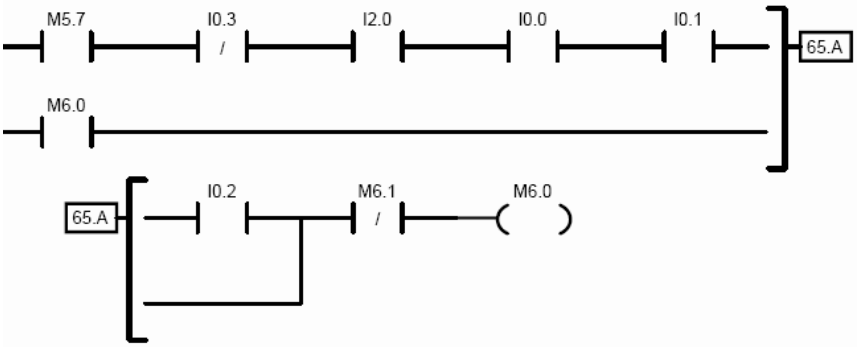
Network 63 SI SE DETECTA IMPULSO DE SUBIDA



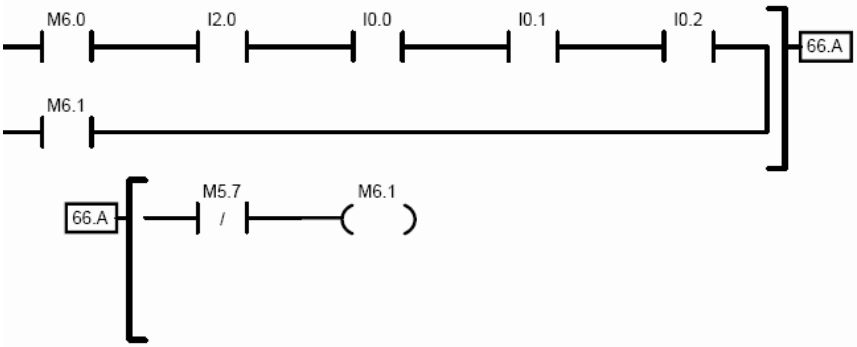
Network 64 SI PILA ES IGUAL A PLANTA



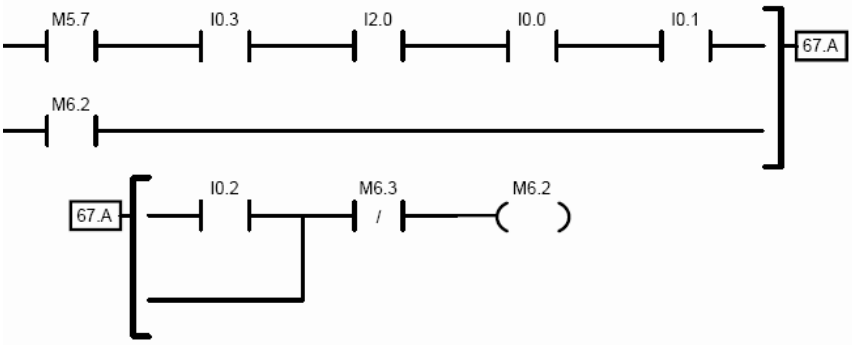
Network 65 SI PILA ES IGUAL A PLANTA Y NO SE HA LLEGADO A PARADA



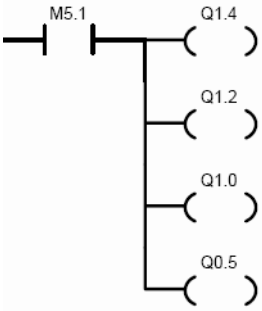
Network 66 SI PILA ES IGUAL A PLANTA Y NO SE HA LLEGADO A PARADA



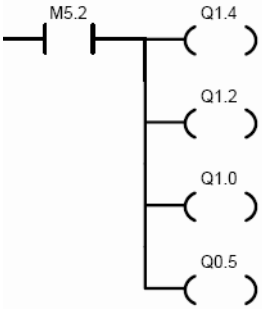
Network 67 SI PILA ES IGUAL A PLANTA Y SE HA LLEGADO A PARADA



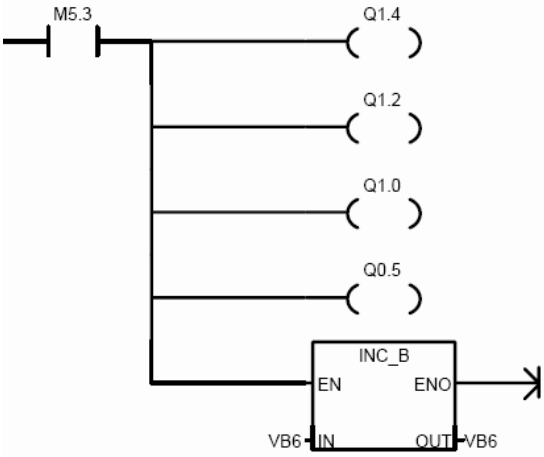
Network 69 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



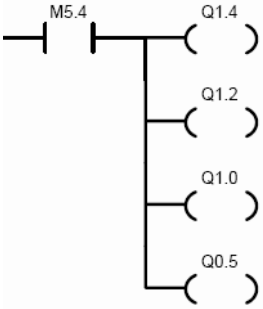
Network 70 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



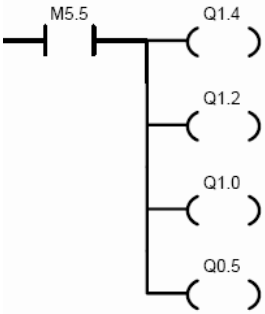
Network 71 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



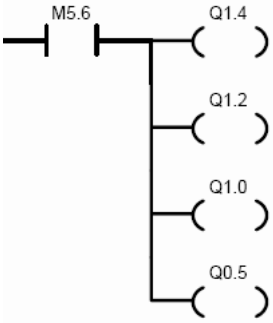
Network 72 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



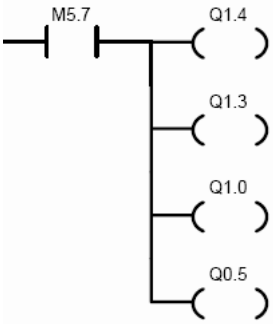
Network 73 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



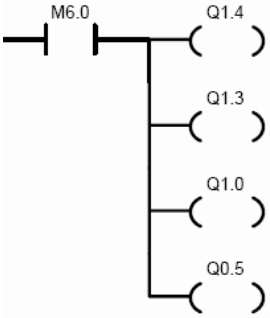
Network 74 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



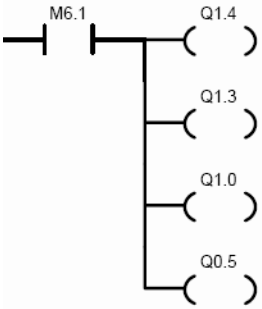
Network 75 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



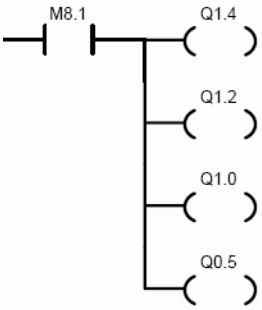
Network 76 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



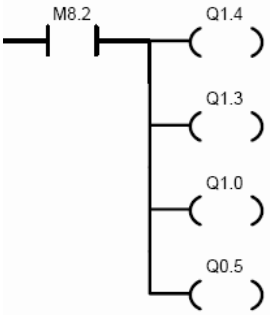
Network 77 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA



Network 78 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA

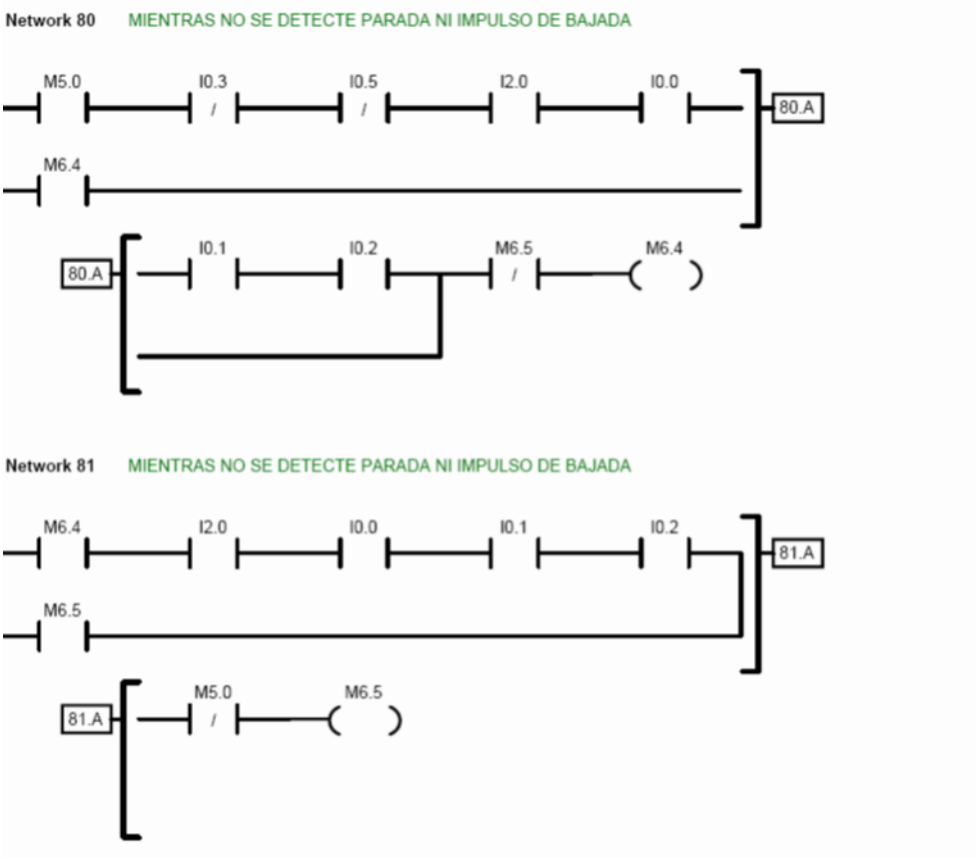


Network 79 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN SUBIDA

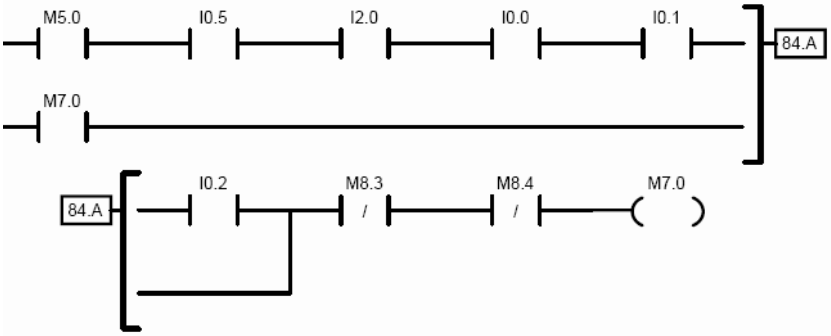


El ascensor asciende en velocidad rápida contando las veces que se activa el detector de parada e incrementando a cada activación el valor de la variable de la posición actual de la cabina (Segmento 71) Cuando se active el detector de parada justo inferior a la planta a la que se desee acceder, se cambia a subida en velocidad lenta. Para hacer esto se decremento en 1 el valor de la planta llamada y se compara con el valor de la posición actual de la cabina. (Segmentos 60, 61 y 63)Al llegar a la siguiente parada, el ascensor se detiene y se inicia la subrutina de apertura de puertas.

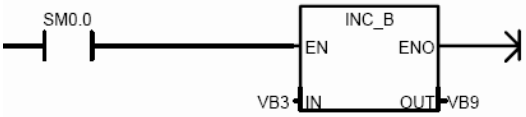
Ascensor posicionándose en bajada



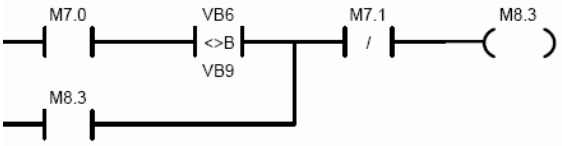
Network 84 SI SE DETECTA IMPULSO DE BAJADA



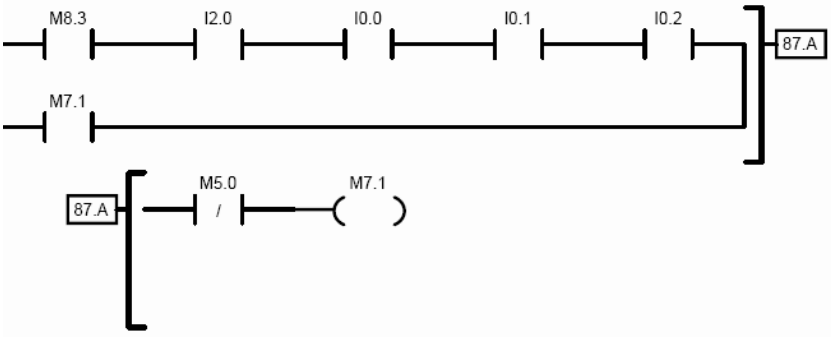
Network 85 SI SE DETECTA IMPULSO DE BAJADA



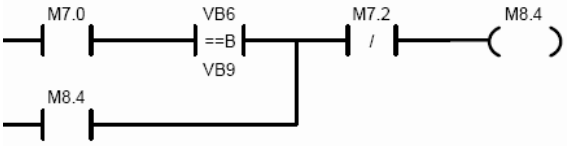
Network 86 SI SE DETECTA IMPULSO DE BAJADA



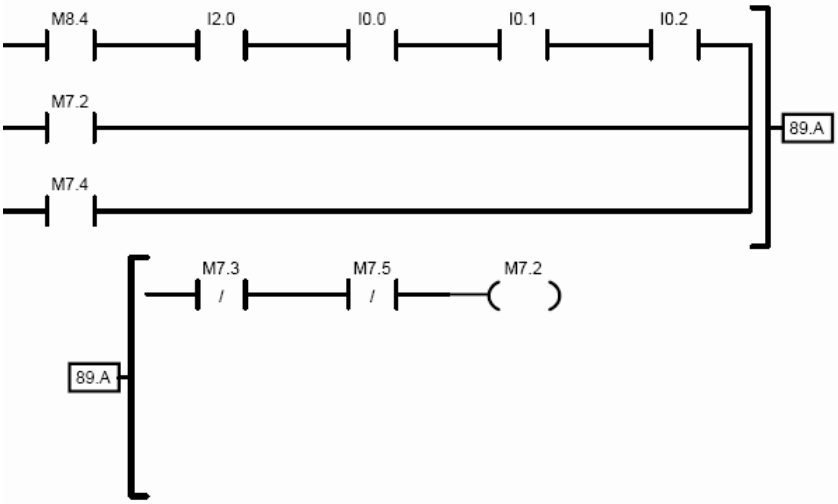
Network 87 SI SE DETECTA IMPULSO DE BAJADA



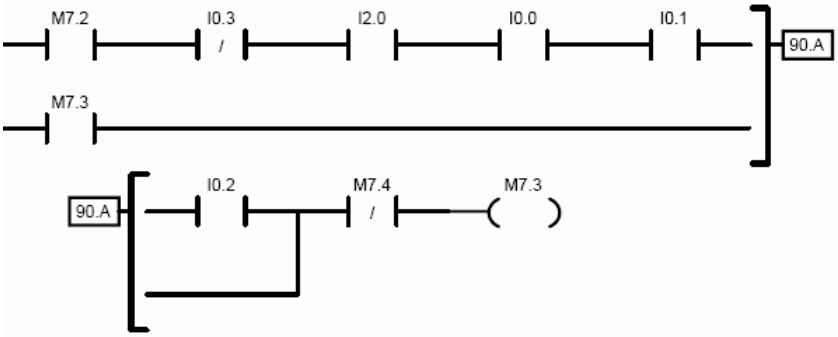
Network 88 SI SE DETECTA IMPULSO DE BAJADA



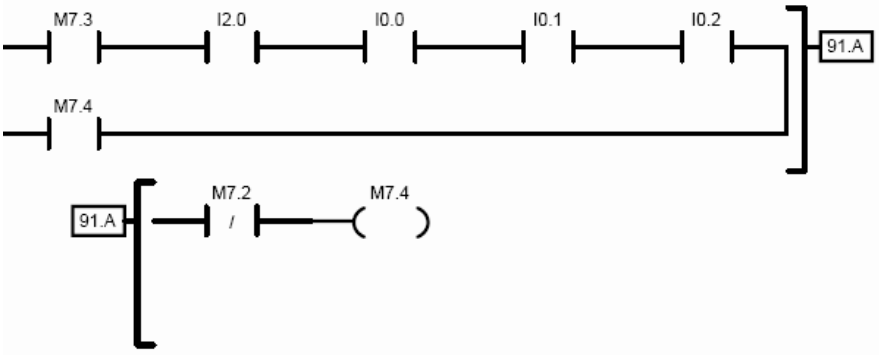
Network 89 SI PILA ES IGUAL A PLANTA



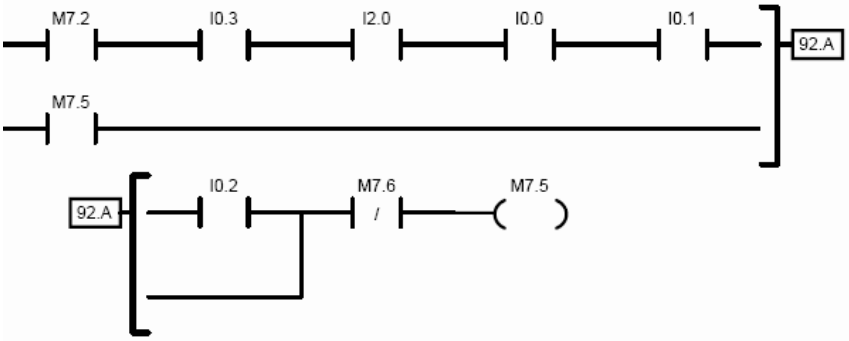
Network 90 SI PILA ES IGUAL A PLANTA Y NO SE HA LLEGADO A PARADA



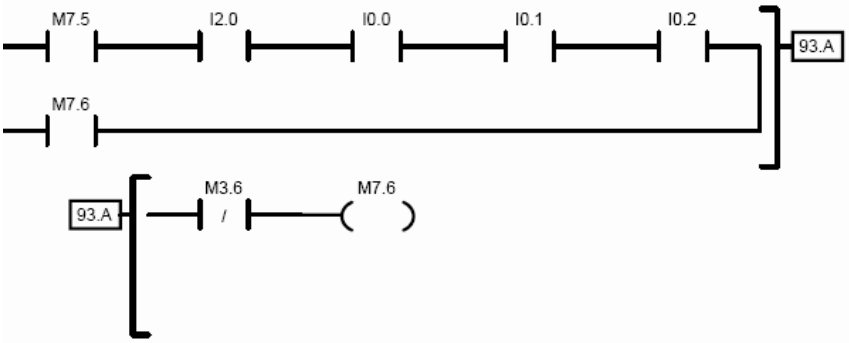
Network 91 SI PILA ES IGUAL A PLANTA Y NO SE HA LLEGADO A PARADA



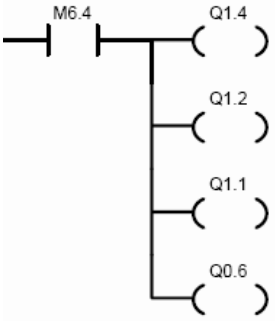
Network 92 SI PILA ES IGUAL A PLANTA Y SE HA LLEGADO A PARADA



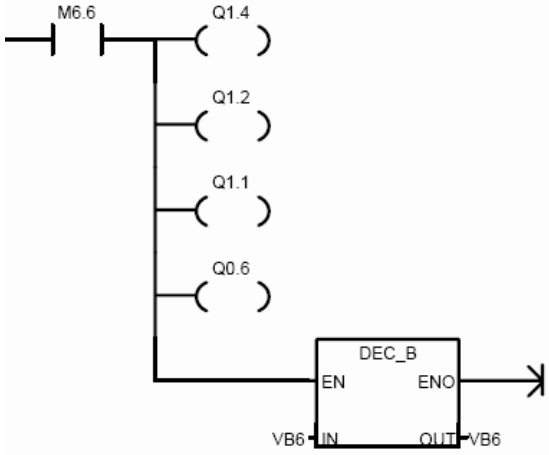
Network 93 SI PILA ES IGUAL A PLANTA Y SE HA LLEGADO A PARADA



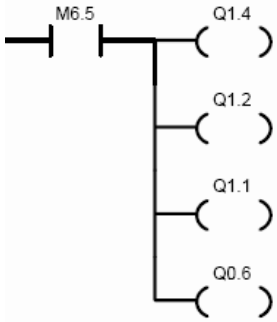
Network 94 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



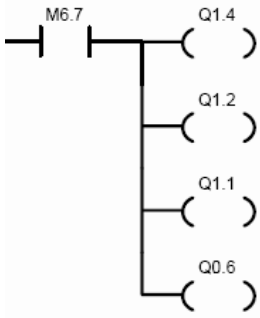
Network 96 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



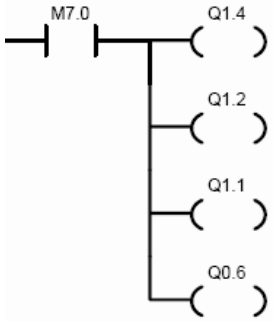
Network 95 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



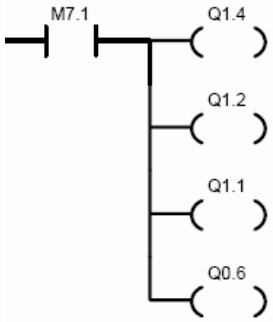
Network 97 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



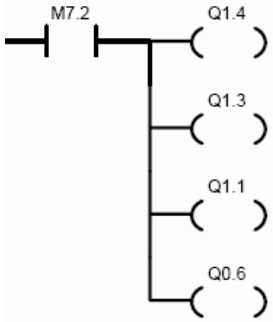
Network 98 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



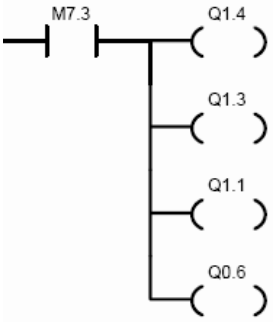
Network 99 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



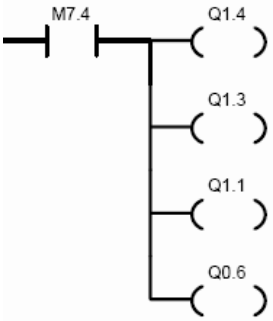
Network 100 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



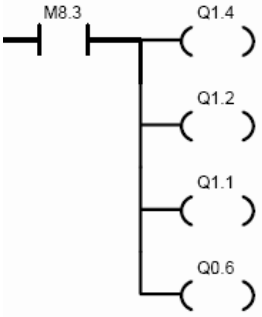
Network 101 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



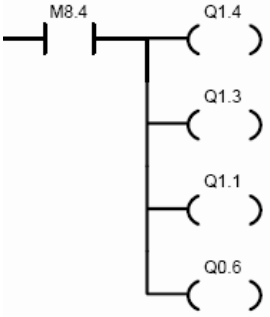
Network 102 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



Network 103 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA

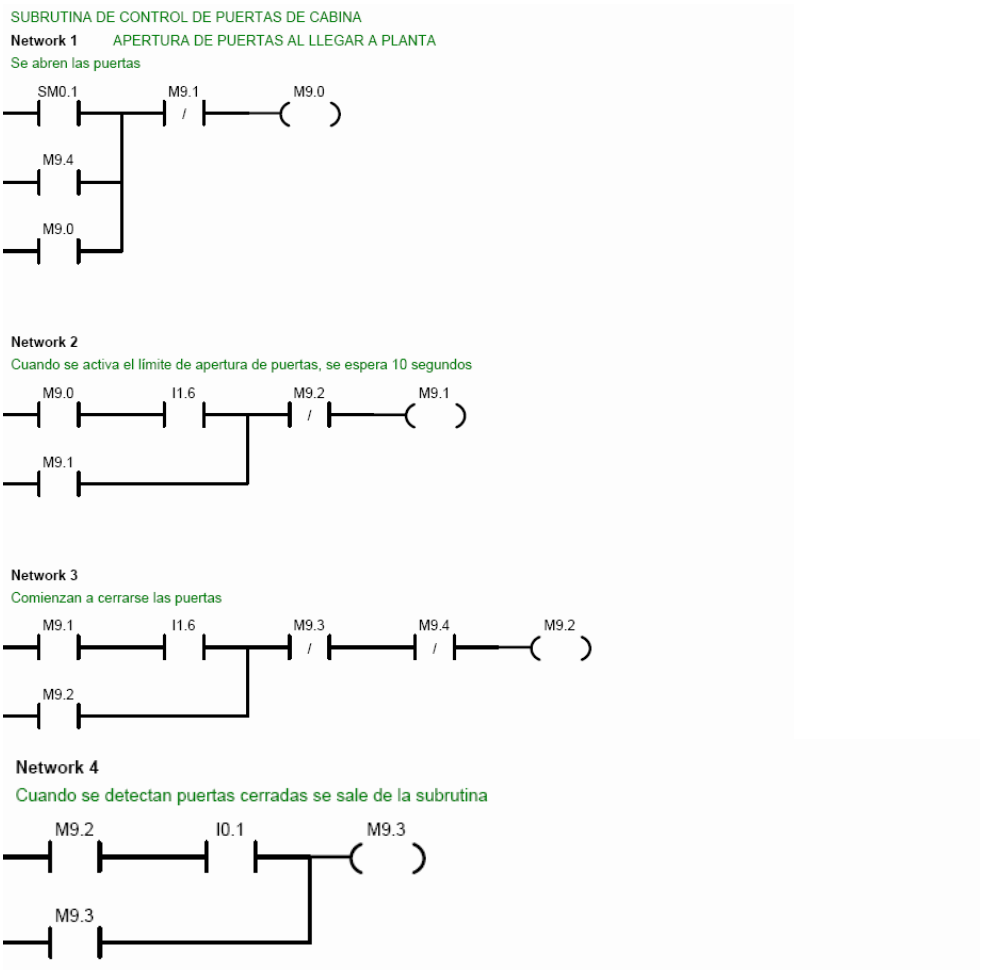


Network 104 ACCIONES DEL CONTROL DE LLEGADA A PLANTA EN BAJADA



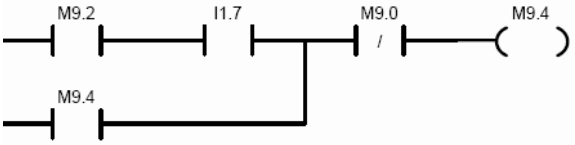
El ascensor desciende en velocidad rápida contando las veces que se activa el detector de parada y decrece a cada activación el valor de la variable de la posición actual de la cabina (Segmento 96)Cuando se active el detector de parada justo superior a la planta a la que se desee acceder, se cambia a bajada en velocidad lenta. Para hacer esto se incrementa en 1 el valor de la planta llamada y se compara con el valor de la posición actual de la cabina. (Segmentos 85, 86 y 88)Al llegar a la siguiente parada, el ascensor se detiene y se inicia la subrutina de apertura de puertas.

Subrutina de control de apertura de puertas (SBR_0)

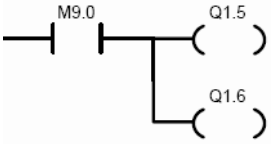


Network 5

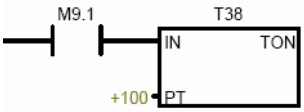
Si durante el cierre se detecta reapertura, las puertas comienzan a abrirse y se inicia de nuevo la subrutina



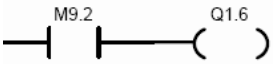
Network 6 ACCIONES DE LA SUBRUTINA DE PUERTAS



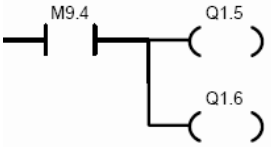
Network 7



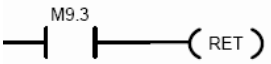
Network 8



Network 9 ACCIONES DE LA SUBRUTINA DE PUERTAS



Network 10



Cuando se inicia la subrutina de control de puertas, el relé selector de giro se coloca en posición de apertura y se activa el motor del operador de puertas, por lo que estas se abren. Cuando se detecta que se han abierto completamente, la maniobra espera 10 segundos para permitir el paso de usuarios hacia dentro y fuera de cabina. Para ello, se utiliza el Timer T38 del PLC (Segmento 7) Una vez ha pasado este tiempo, las puertas comienzan a cerrarse, activando el motor del

operador y colocando el relé selector en posición de cierre. Esta acción se mantiene en el tiempo hasta que se detecte el cierre completo de puertas, en cuyo caso se sale de la subrutina, o se active una reapertura de puertas (pulsador de reapertura, Fotocélulas, sobrecarga, etc…). En este caso las puertas invierten su giro y comienzan a abrirse de nuevo reiniciando la subrutina. Si las puertas se han cerrado completamente, se sale de la subrutina (Segmento 10)

Sugerencias.

Bus de datos

Se puede sustituir todo el cableado y la instalación de hueco y cabina (excepto seguridades) por un bus de datos tipo industrial que implemente:

- Entrada de datos de pulsadores exteriores y de cabina al PLC
- Entrada de datos de posicionamiento de los detectores magnéticos al PLC
- Salida de datos del PLC a los sentirlas, posicionales y decodificadores BCD

Esto conseguiría, por un lado, simplificar la instalación de hueco y cabina ya que el bus de datos industrial elegido utiliza únicamente 2 hilos + 2 hilos de alimentación. Por otro lado, permite una ampliación de la maniobra sencilla y económica para que el ascensor contemple el acceso a más de 6 paradas. Únicamente agregando partes de código de pulsadores al programa del PLC.Para la implementación, se colocarían entradas y salidas binarias anexas al Autómata programable, salidas y entradas binarias en pulsadores, sentirlas y posicionales a través de hueco, y salidas y entradas binarias en cabina interconectándolos con dos hilos más alimentación. Para conectar el bus de datos con cabina se utilizarían 2 hilos del cordón de maniobra. Una buena solución sería utilizar la tecnología que ofrece el producto “Dúplice” de la empresa “Carlo Gavazzi”. Se trata de un bus de datos industrial económico y de fácil programación. Con un sencillo software se asocian entradas a salidas y viceversa. Un alimentador-generador “master” de bus de datos + 6 entradas binarias + 6 salidas de libre potencia hasta 6 Amperios y el software de programación para PC .

Variador de frecuencia

Se puede sustituir la maniobra de doble velocidad del ascensor y el motor de 2 bobinas por un sistema con variación de frecuencia y tensión. Se deberá instalar un dispositivo variador de frecuencia trifásico anterior a la acometida del motor. Para controlar su funcionamiento fácilmente, simplemente se deben sustituir el

contacto res (a excepción del contacto de seguridad) e introducir dichas señales al variador como consigna de subida, de bajada, de aceleración y de desaceleración. A la puesta en marcha se deberán programar datos como valores de rampa de aceleración y desaceleración etc…

Este tipo de sistemas incrementan considerablemente el coste de realización de la maniobra pero se gana en el confort de los pasajeros y se ahorra una considerable energía, al reducir los picos de corriente en el arranque y decremento la fricción y la energía potencial del ascensor cuando está parando.

Temporización de iluminación de cabina

A modo de ahorro de energía se puede temporizar la iluminación interior de cabina cuando no existan usuarios en su interior. El control se podría realizar de dos formas:

- Temporizando la iluminación con un dispositivo adecuado en que la señal de activación la genere la maniobra del operador de puertas.
- Controlando la iluminación a través de un sensor de presencia, de tal forma que las

Luminarias se enciendan si se encuentran usuarios en el interior de cabina y se apaguen al marcharse.

REFERENCIAS BIBLIOGRÁFICAS Y VIRTUALES

[1]"CONSTRUCCION DE ROBOTS AUTONOMOS COLABORATIVOS", TESIS DE MAESTRIA EN INFORMATICA, FACULTAD DE TECNOLOGIA, UNIVERSIDAD ABIERTA INTERAMERICANA, Tesista: Ing., Néstor Adrián Balich, director: Dr. Marcelo De Vicenza.

[2] AUTOMATAS PROGRAMABLES Y SISTEMAS DE AUTOMATIZACION. **Autor:** Mandado/Acevedo/Fernández/Arresto
Editorial: MARCOMBO, S.A.

[3] Prácticas y procesos de taller de mecanizado. Mallorquín Egea, Salvador / Carrasco Moreno

[4] Inventor y su simulación con ejercicios. Younis, Wasim

[5]Análisis y diseño de piezas de aluminio extruido. Vásquez Angulo, José Antonio

REFERENCIAS VIRTUALES

[2] http://arduino.cc/es/Tutorial/HomePage

[3] http://arduino.cc/es/Reference/HomePage

[4] http://arduino.cc/es/Tutorial/HomePage

[5] http://www.idoneos.com/open_modular_hardware/

[6] http://www.makeblock.cc/

[7]http://blog.makeblock.cc/

[8] http://www.wired.com/2012/12/makeblock/

[9]http://balau82.wordpress.com/2012/12/23/makeblock-the-body-of-your-

open-hardware-projects/

[10]http://openalia.wordpress.com/2012/12/30/makeblock-is-a-new-modular-aluminum-extrusion-system/

[11]http.ro-botica.com/Arduino.asp

[12]http://www.samachar.com/makeblock-open-source-lego-for-adults-mmtvR4aeeff.html

[13] http://www.robotshop.com/en/robot-parts.html

[14]http://www.ardumania.es/midiendo-distancias-con-un-sensor-de-ultrasonidos/

[15] http://www.ardublog.com/library-for-arduino-ultrasonic-ranging-hc-sr04/

[16] http://forum.arduino.cc/index.php?topic=127105.0

[17]http://arubia45.blogspot.mx/2013/02/sensor-ultrasonico-hc-sr04-arduino.html

[18] http://www.opiron.com/portfolio/sensores-de-ultrasonidos-arduino-opiron