

# **UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**

## **FACULTAD DE INGENIERÍA**

TESIS NUMERO DE REGISTRO: 0602011

**DISEÑO Y FABRICACIÓN DE UN PROTOTIPO A ESCALA  
PARA EMPAQUETAMIENTO, SELECCIÓN Y TRASLADO DE  
DIFERENTES PRODUCTOS EN UNA PLANTA INDUSTRIAL**

Anaya González José Carlos  
Díaz Morales Adrián



Universidad Nacional  
Autónoma de México



**UNAM – Dirección General de Bibliotecas**

**Tesis Digitales**

**Restricciones de uso**

**DERECHOS RESERVADOS ©**

**PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL**

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Esta tesis engloba todo mi esfuerzo, respeto, agradecimiento y admiración a todos y todo lo que de una u otra forma contribuyeron a que yo cumpliera esta meta.

A mi papá, mamá y mi hermana, que siempre incansables están ahí para apoyarme, orientarme y quererme, que siempre con un abrazo, una sonrisa o un consejo me guían para yo conseguir las cosas que más me benefician como persona en todos aspectos.

A mis amigos que me hicieron disfrutar de mi carrera por medio del estudio, el esfuerzo, compañerismo, pero principalmente por medio de la amistad que sin darnos cuenta surgió entre nosotros, una amistad que comenzó en las aulas, permanece hasta hoy y deberá seguir por siempre, como siempre ha sido sana y honesta, una amistad que nunca espera nada a cambio de los demás y que siempre esta ahí para los mejores y los peores momentos.

A mis profesores, que siempre demostraron su capacidad, aprecio a sus alumnos y amor a la Universidad a todo momento dentro y fuera de los salones de clases, gracias a ellos por transmitir sus conocimientos esperando a cambio únicamente nuestro esfuerzo para salir adelante para nuestro beneficio propio.

A mi Universidad y Facultad, de las cuales estoy muy orgulloso de formar parte, que son un lugar de cobijo y aprendizaje, donde encontré desde el principio los mejores ejemplos para poder salir adelante, los mejores amigos y los mejores profesores que pude haber tenido.

Adrián

Gracias a todas y cada una de las personas que me han visto crecer como persona a lo largo de estos años, a las que han estado conmigo en los ratos de éxito y júbilo, pero sobre todo, a las que estuvieron conmigo en mi difícil inicio por allá por los primeros semestres.

Gracias al Adri que es el mejor amigo que cualquier persona desearía tener.

Gracias a todas las personas que trabajaron dura y desinteresadamente en nuestra tesis: a Borre, que siempre brindo esa calma, ayuda y serenidad, al Jeimio, que nunca dejo de tener esa sonrisa alivianadora, esa inteligencia nata y esa pasividad excesiva, al Ulisten, que nos presto sus motores, su cochecito y su laboratorio para trabajar, a Gabriel, que nos brindo su conocimiento y apoyo. Pero por encima de todos, a Caro, que nos brindo su conocimiento, su dinero, su ayuda, su apoyo, su cariño, su entusiasmo, sus ganas de que todo funcionara a la perfección, su conocimiento metódico y limpio de mal formaciones mentales (por no decir p..... que a Adrian y a mi se nos ocurrían); si fuéramos justos, esta tesis seria mas suya que nuestra (bueno tal vez en la misma proporción) pero ella decidió tener una tesis más elevada.

A todos mis profesores que siempre estuvieron llenos de sabiduría y de generosidad de compartir su conocimiento. A la H.H. Universidad Nacional Autónoma de México y a la Facultad de Ingeniería, que me vieron llegar como adolescente y graduarme como ingeniero.

A mi Papá, que me brindo las bases de mi formación, a mi Mamá, que me brindo el amor desinteresado y la ayuda para formar mi propia personalidad, a Francisco, que me brindo su respaldo y su cariño, y a mi hermana, que no ha dejado de fastidiar y apoyarme todos los años que llevo de vida (tal vez más de apoyarme y de quererme que de lo otro).

A todos los que aquí se mencionan y a los que no, gracias por haberme dado la posibilidad de conocerlos y gracias por formar parte de mi vida y, aunque no mencione su nombre aquí, en persona agradeceré su gran apoyo.

José Carlos

# Índice

|                                                                             | Pág. |
|-----------------------------------------------------------------------------|------|
| <b>Objetivos</b>                                                            | i    |
| <b>Resumen del proyecto de tesis</b>                                        | ii   |
| <b>Introducción</b>                                                         | iii  |
| <b>Antecedentes</b>                                                         | v    |
| <b>Capítulo 1.- Diseño, elaboración y lógica de control del montacargas</b> |      |
| 1.1    Diseño y fabricación del montacargas autoguiado                      | 1    |
| 1.2    Diseño del órgano terminal                                           | 4    |
| 1.2.1  Capacidad de carga del órgano terminal                               | 11   |
| 1.3    Sistema electrónico del AGV                                          | 13   |
| 1.4    Lógica del movimiento del AGV                                        | 26   |
| <b>Capítulo 2.-Automatización del proceso de distribución</b>               |      |
| 2.1    Diseño y configuración del sistema                                   | 44   |
| 2.2    Percepción y acción del proceso                                      | 47   |
| 2.3    Lógica de discriminación de productos                                | 58   |
| <b>Capítulo 3.- Integración</b>                                             |      |
| 3.1    Acoplamiento de sub-sistemas                                         | 82   |
| 3.2    Toma de decisiones del AGV                                           | 89   |
| 3.3    Interfaz con el usuario                                              | 94   |
| <b>Fabricación y pruebas</b>                                                | 99   |
| <b>Conclusiones y comentarios</b>                                           | 103  |
| <b>Bibliografía</b>                                                         | 105  |
| <b>Apéndice I</b> Datasheets                                                | 106  |
| <b>Apéndice II</b> PIC16F877                                                | 117  |
| <b>Apéndice III</b> Visual Basic 6.0                                        | 156  |
| <b>Apéndice IV</b> PLC Micro 1                                              | 157  |

### **OBJETIVOS**

Diseñar y construir el prototipo a escala de un montacargas industrial autoguiado, el cual deberá recoger un producto de una línea de distribución para trasladarlo a una zona específica.

Diseñar y construir una línea de distribución a escala, la cual será capaz de despachar dos tipos de productos diferentes, que serán seleccionados por el usuario, según el producto que necesite.

Diseñar, construir e implementar un software que actúe como una interfaz que permitirá al usuario comunicarse con el montacargas para indicarle el tipo de producto que deberá recoger y, a la vez, también permitirá la comunicación del usuario con la banda de distribución para ordenar el tipo de producto a despachar.

## **RESUMEN DEL PROYECTO DE TESIS**

El presente proyecto fue desarrollado a través del conjunto de varios subsistemas: mecánicos, electroneumáticos, electrónicos, informáticos y de control, y con la sinergia de todos ellos se logró conformar un solo sistema que se encarga de cubrir el objetivo de este proyecto. El sistema en su conjunto cuenta con un montacargas, una banda de distribución y una interfaz que los comunica con el usuario.

El subsistema que realiza la función de trasladar al montacargas es un AGV, es decir, tiene como tracción un vehículo autoguiado seguidor de línea, sobre el cuál esta montado un mecanismo de cuatro barras, que gracias a las propiedades de cuerpo rígido, permite al montacargas levantar los paquetes que le sean asignados.

El montacargas cuenta con sensores que lo dotan de percepción, las señales que los sensores transmitan se procesarán y de esta forma se asigna gran parte de las funciones que deben realizar tanto el AGV como el elevador del montacargas. Esta tarea es posible mediante el uso y previa programación de un microprocesador.

Por otra parte, las bandas transportadoras son controladas por un PLC, por sus siglas en inglés, Controlador Lógico Programable, que se encarga de asignar las tareas de los actuadores (pistones y motores de CD) que despachan los paquetes.

Finalmente, se desarrolló una interfaz que comunica al usuario de manera sencilla con el microprocesador y con el PLC, para así poder asignar la ruta del AGV y el producto que se deba despachar

## **INTRODUCCIÓN**

El presente trabajo nace de la necesidad de buscar soluciones a algunos de los problemas más comunes presentados en la industria. De esta forma se pretenden cubrir los objetivos anteriormente citados, los cuales están enfocados a dar una alternativa que sea segura y eficaz para el personal y los productos durante su empaquetamiento y traslado.

Como ya se mencionó, la satisfacción inmediata de los pedidos y la seguridad en el traslado de los productos constituyen los principales puntos a atacar; dando como consecuencia el haber planteado la idea de diseñar el prototipo de un robot móvil a escala que permita ejemplificar la solución que se ha propuesto para esta necesidad. De igual forma, se ha diseñado una banda de empaquetamiento y distribución, también a escala, que se encargue de despachar dos productos diferentes. La banda y el montacargas serán capaces de recibir órdenes del usuario por medio de la comunicación con una computadora para poder asignar el producto a despachar y a recoger, respectivamente.

A continuación se hace una breve descripción de los apartados referentes a los antecedentes y capitulado que comprende el desarrollo de este proyecto.

De manera previa, se cuenta con los antecedentes que involucra el proyecto, en él se encuentran tipos de robots móviles, marco histórico y una breve referencia de la tendencia a nivel mundial en cuanto al desarrollo de este tipo de alternativas para la industria.

Posteriormente, en el primer capítulo se tratará la parte respectiva al diseño, elaboración y lógica de control del montacargas, incluyendo su órgano terminal.



El segundo capítulo aborda los temas referentes a la automatización del proceso, la banda de distribución, actuadores y sensores utilizados.

El tercer capítulo trata sobre la integración de los subsistemas teniendo en consecuencia, el proyecto funcionando en conjunto, esto involucra el diseño de la interfaz entre el usuario y el sistema.

Para finalizar se encontrarán los siguientes apartados: conclusiones, apéndice y bibliografía.

## **ANTECEDENTES**

Existe una larga historia y tradición en la que el hombre ha querido fabricar sistemas y seres animados que pretenden asemejar su figura y movimientos, desde los griegos hasta nuestros días, pasando por autómatas contruidos por artesanos franceses y suizos del siglo XVIII, que ya incorporaban dispositivos mecánicos para el control automático de movimientos.

En nuestros días, el desarrollo de las máquinas ha estado cada vez más involucrado con procesos tecnológicos, dando lugar a máquinas que tienen como objetivo la amplificación de la potencia muscular del hombre, sustituyéndola en su trabajo físico, y a máquinas o instrumentos capaces de procesar información, complementando o incluso sustituyendo, al ser humano en algunas actividades intelectuales.

El concepto de robot refleja imágenes de máquinas enfocadas a la realización de trabajos productivos, con imitación de movimientos y de comportamientos de seres vivos, con esta idea cabe mencionar que el término “robot” proviene de la palabra checa “robota” que significa esclavo o fuerza de trabajo y que nace asociado a la idea de producción. Actualmente estos esclavos tecnológicos son creaciones ingenieriles concebidas para producir bienes y servicios o explotación de recursos naturales.

Los robots industriales nacen a partir de la convergencia de tecnologías de control de máquinas herramientas, de los manipuladores operados a distancia y aplicaciones en tiempo real. Por este medio, el control pretende gobernar una tarea sin la intervención de agentes externos, en específico del hombre.

La automatización industrial se hace presente en el siglo XIX, pero es hasta el siglo XX después de la Segunda Guerra Mundial, cuando comienza a expandirse de manera importante en el sector industrial.

En todo este desarrollo se han empleado, desde hace muchos años, diferentes formas de tecnologías como son la neumática, hidráulica y eléctrica. Pero fue a principios de los años setentas cuando se da la aparición del microprocesador, para ser específico, en 1972, con el cual se dio un gran impulso al control de robots y procesos industriales. Los avances en electrónica en los siguientes años y la creciente aplicación de circuitos integrados, acentuaron esta tendencia.

Un robot tiene elementos propios que lo distinguen, un órgano terminal, brazo articulado y un sistema que los desplaza; sin embargo, algunos robots no cuentan con todos esos elementos, por ejemplo, el órgano terminal y el brazo articulado pueden estar fijos a una base. Además, un robot consta de un sistema de procesamiento de información que atiende las funciones de control de movimientos, percepción y planificación.

Los sensores brindan información sobre las condiciones del entorno, es decir son los sentidos del robot, con lo cual hace posible su adaptación al medio, todo este desarrollo parte de procesos de visión, tacto, audición, entre otros; estos elementos no sólo conforman parte de la percepción, sino el tratamiento e interpretación de la información del entorno.

Estos sistemas enfocan también su uso a la planificación de la trayectoria, desde una posición inicial a la final, minimizando riesgos y principalmente colisiones, todos estos movimientos serán realizados por actuadores. Los actuadores son los elementos que realizan el trabajo para cada una de las partes del sistema, son quienes le dan vida física a un robot o proceso, existen

diferentes tipos ellos dependiendo de las condiciones de trabajo y tarea que se tenga prevista.

Conjuntamente con lo descrito anteriormente, en los términos generales de un robot también se encuentra el trabajo de programación a diferentes niveles, todos ellos orientados a las funciones que debe realizar tratando de dotar de “inteligencia” que le permita tomar decisiones, coordinar sus actividades, obteniendo principalmente una fabricación flexible.

Particularmente, el desarrollo de robots móviles responde a la necesidad de extender el campo de aplicación de la Robótica, la cual estaba en sus inicios restringida a dispositivos mecánicos fijos que únicamente permitían el desplazamiento de objetos según su alcance propio sin dejar de estar anclados en alguno de sus extremos. Ahora el hombre se empeña en dar autonomía a las máquinas para que el hombre tenga menos injerencia al momento de su desplazamiento.

Hoy en día, en la construcción de los vehículos autoguiados se emplean los avances más recientes en cuanto a desarrollo de materiales, tecnología y automatización. La gran mayoría de los vehículos autómatas emplea cables, líneas y elementos magnéticos que son capaces de dirigirlos durante el recorrido de su ruta. Sin embargo, carecen de inteligencia que les permita recorrer más de una ruta, o elegir entre ellas.

Esta necesidad ha originado que surja el interés por desarrollar alternativas que permitan desempeñarse en diferentes sectores y tareas, marítima y portuaria, agricultura, minería y sistemas militares principalmente, sin embargo, ha sido el sector académico en colaboración con la industria quien han tenido mayor interés en su evolución, teniendo como ejemplo la Universidad de Oxford con el apoyo de Caterpillar que desde 1988 trabaja en la serie de los vehículos GEC o

la Universidad de Sevilla con el vehículo Romeo (Fig. 1), entre otros. Todos estos casos están basados en problemas comunes en la industria, que necesitan para su evolución nuevo software y hardware que sean viables para atacar dichos problemas. Entre estos retos se encuentra el de fabricar sensores inteligentes (sonar, visión, proximidad), estos elementos junto con el sistema de control son la llave que permite día a día realizar labores más complejas en tiempo real.



Fig. 1, Romeo III Universidad de Sevilla

Dentro de las funciones básicas que puede involucrar el uso de un AGVs se encuentran:

- La navegación y dirección, las cuales permiten que el vehículo siga una ruta predeterminada que sea óptima para el patrón del traslado del producto o material.
- El encaminamiento, éste se refiere a la capacidad del vehículo de tomar decisiones a lo largo de la trayectoria para seleccionar las rutas óptimas a los destinos específicos.
- El manejo del tráfico, es el sistema que se encarga de evitar colisiones con otros vehículos, de esta forma optimiza el flujo vehicular y por lo tanto también optimiza el tiempo de entrega de los productos.
- La transferencia de la carga, es el método de recolección y de la entrega para un sistema de AGVs, que puede ser simple o integrado con otros subsistemas.
- El manejo del tráfico, es el control de sistema para dictar cada operación a realizar.

El diseño apropiado de cada una de estas funciones y su capacidad de trabajo determinan en gran medida el éxito de un sistema AGV.

Con respecto al grado de autonomía, los robots móviles pueden clasificarse en teleoperados, de funcionamiento repetitivo, y autónomos o inteligentes.

En los robots teleoperados, las tareas de percepción del entorno y planificación, además de la manipulación compleja de objetos, son realizadas por seres humanos, cuentan con un operario que en tiempo real se encarga de asignarle las tareas cerrando el bucle de control de alto nivel. Sistemas evolucionados entregan información sensorial al operador para realimentarse con el entorno.

Principalmente, el uso de robots teleoperados está dirigido a trabajos de localización remota en lugares de difícil acceso, lugares riesgosos por su configuración o peligro que representen, en trabajos difíciles de automatizar o en lugares no estructurados.

Entre sus desventajas se encuentran las limitantes propias del operario en procesamiento numérico, precisión y, principalmente, la coordinación y “entendimiento” con el robot en sí, aspectos que usualmente representan la mayor complejidad de la tarea. El retraso en las transmisiones de la información, es un factor que también interviene en los procesos de control, haciéndolo más complejo.

Por otra parte, los robots de funcionamiento repetitivo, son los de mayor uso en cadenas industriales, normalmente tienen tareas invariantes y predecibles, tienen una corta percepción de su entorno, y son relativamente rápidos y precisos. Le ahorran al hombre tiempo en tareas repetitivas, en ocasiones peligrosas y hasta riesgosas para él.

Finalmente, dentro de los robots más evolucionados se encuentran los autónomos o inteligentes, que pueden percibir, modelar su alrededor, planificar y actuar para alcanzar sus objetivos, los cuales debe conseguir con muy poca o sin la intervención del ser humano. Deben ser capaces de desempeñarse en lugares dinámicos o no estructurados, teniendo alternativas para librar contingencias y percances que puedan ocurrir dentro de su desempeño.

Los esfuerzos en los últimos años van orientados a la “Inteligencia Artificial” con altos sistemas de procesamiento para atacar las dificultades que surgen del alto procesamiento de información, influye mucho la realimentación sensorial que juega un papel importante en la captación de problemas. Sus limitaciones se encuentran en la configuración del robot y su sistema de control.

Para planificar las funciones del robot, la arquitectura y configuración física son factores de suma importancia, los cuales proveerán de movimiento y accesibilidad.

Estos robots presentan diferentes tipos de locomoción que les confieren características y propiedades variadas respecto a su eficiencia energética, dimensiones, cargas útiles y maniobrabilidad.

Los vehículos con ruedas son los que dan la solución más eficiente en terrenos duros y libres de obstáculos, consiguen velocidades relativamente altas y gran estabilidad, no obstante son los más ineficientes en terrenos blandos.

La mayor maniobrabilidad se logra con los vehículos omnidireccionales. Un vehículo omnidireccional en el plano (Fig. 2), es capaz de trasladarse simultánea e independientemente en cada eje del sistema de coordenadas y rotar según el eje perpendicular.

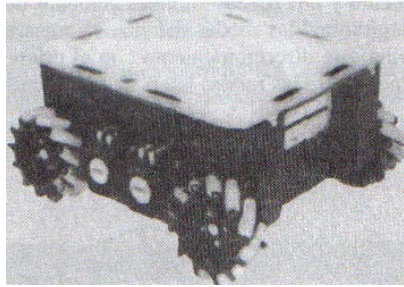


Fig. 2, Vehículo omnidireccional  
(Universidad Carnegie Mellon)

Las configuraciones más comunes de robots móviles son: Ackerman (Fig. 3) por ejemplo, el desarrollado por el Robotics Institute de Carnegie Mellon University (Fig. 4), otra configuración bastante usual es la del Triciclo Clásico (Fig. 5), por direccionamiento diferencial (Fig. 6), la configuración Skid Steer para la cual el robot Terregator (Fig. 7) es un claro ejemplo, que fue desarrollado para aplicaciones mineras en 1993, síncronos (Fig. 8-a y 8-b) y locomociones por patas, donde se encuentran varios, según la cantidad de apoyos.

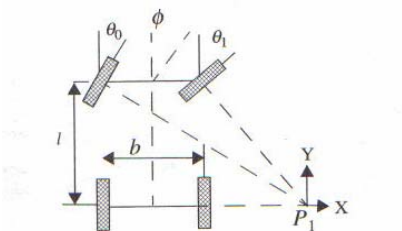


Fig. 3, Sistema Ackerman



Fig. 4, NavLab2, Robot tipo Ackerman

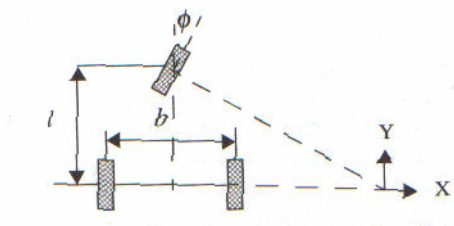


Fig. 5, Locomoción de triciclo clásico

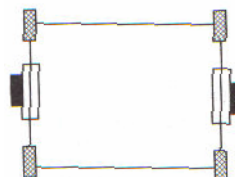


Fig. 6, Locomoción con  
direccionamiento diferencial





Fig. 7, Robot Terregator  
(Carnegie Mellon University)



Fig. 8-a, Locomotion Emulator,  
(Carnegie Mellon University)

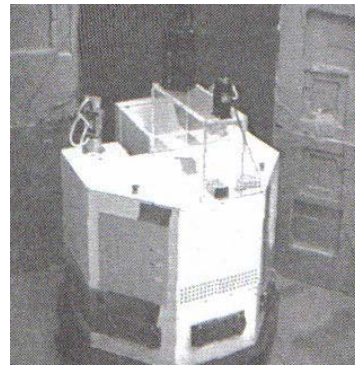


Fig. 8-b, Ram 1, Universidad de Málaga.

Existen otros tipos de tracción, además de las ruedas, los vehículos tipo oruga (Fig. 9) consiguen trasladarse mediante pistas de deslizamiento, lo que es útil en campos irregulares donde presenta un buen rendimiento.

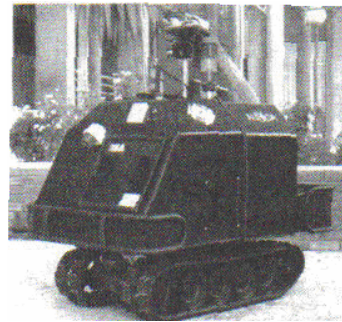


Fig. 9, Robot Auriga  
(Instituto Andaluz de Automática y Robótica)

Dentro de la clasificación de robots móviles, se encuentra la locomoción por extremidades (patas), esta configuración aísla al robot de la superficie, es mejor que las ruedas y las pistas de deslizamiento para trasladarse en lugares con obstáculos, también se puede conseguir omnidireccionalidad y casi no presenta deslizamiento. Su complejidad es mucho mayor debido a los mecanismos que emplea para soportar el cuerpo del robot, esto genera un alto

consumo de energía y los problemas de control suelen ser mucho más complicados.

Estos robots se clasifican comúnmente por el número de patas que los sostienen (Fig. 10-a y 10-b).

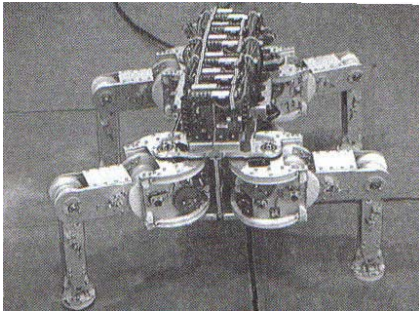


Fig. 10-a, Robot Cuadrúpedo  
(Tokio Institute of Technology)

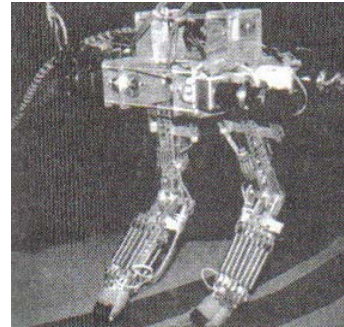


Fig. 10-b, Robot Bípedo  
(Massachussets Institute of Technology)

Cabe mencionar que existen también robots “trepadores” para realizar tareas de desplazamiento vertical (Fig. 11), utilizan garras o ventosas para su sujeción. Así mismo existen robots articulados, que están en completo contacto con el suelo la mayoría de las veces y por último encontramos robots submarinos y aéreos, que, como su nombre lo indica, son diseñados para trabajos en la superficie o dentro del agua, para grandes profundidades o para navegación en el aire, principalmente usados en investigación y funciones militares.

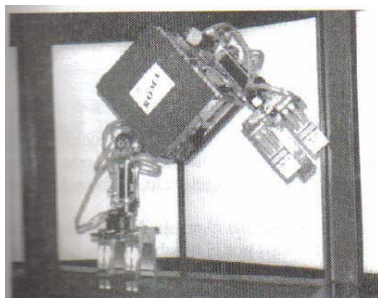


Fig. 11, Robot Trepador  
(Universidad Carlos III de Madrid)

Dentro de la categoría de robots móviles terrestres, existen diversos tipos de control: seguimiento mediante persecución pura, control geométrico mediante generación de polinomios de orden quíntico y control de tipo reactivo.

El control de tipo reactivo es el que representa a los robots seguidores de línea, en ellos las actuaciones de control del vehículo se obtienen directamente de la percepción sensorial del entorno (sensores de tipo externo que representan a la tercera generación de robots). Para este caso de configuración pueden contemplarse dos situaciones diferentes:

- Acoplamiento directo entre los sensores externos y los actuadores, de forma que ante determinadas lecturas de los sensores se generen consignas a los actuadores de dirección y velocidad.
- Generación de maniobras cuya ejecución es necesario controlar.

Por otra parte, es importante mencionar la ubicación actual referente al desarrollo de estas alternativas de transporte de carga en la industria, por lo tanto, es pertinente dar una muestra del estado del arte. Una de las compañías de mayor desarrollo tecnológico a nivel mundial es Siemens y de la cual se presentan a continuación los sistemas AGV's más innovadores y que actualmente ya se emplean en sectores manufactureros de plástico, textiles y de almacenamiento y distribución en general.

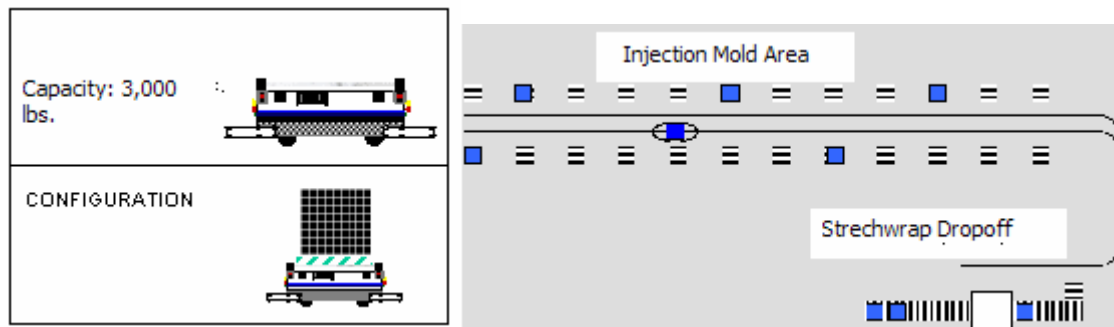
## **EJEMPLOS DE AGVs COMERCIALES:**

### **Ejemplo 1.**

#### **Portador de la Carga en la Unidad de Manufactura de Plásticos Manufacturing/Plastics**

Un solo AGV bidireccional mantiene un área de la producción del moldeo de inyección. Su propósito es transportar tinas con piezas moldeadas de plástico a una estación para que sean empacadas. Cada máquina de moldeo tiene un soporte no-automático de la carga de un transportador de rodillo del AGV. El AGV se equipa con una cubierta que acciona un rodillo junto con un mecanismo auxiliar independiente al otro rodillo. Cuando el AGV llega al lugar de carga,

acciona el rodillo impulsor del rodillo de despegue, para hacer que éstos soporten la carga. La carga finalmente se transfiere automáticamente al AGV. Como consecuencia se obtiene una reducción de costos significativa del sistema, la cuál resultó porque los soportes de la recolección son simples y no automáticos.



En el área de moldeo de inyección, varios operadores son responsables de grupos de máquinas de la producción. Cuando una máquina del moldeo ha llenado una tina, el operador utiliza una terminal próxima para llamar al AGV. El AGV acude a un llamado en cuanto se le indique, a menos que exista una actividad prioritaria. Así mismo el AGV responde a las llamadas de servicio escogiendo los contenedores que se encuentren llenos y con las piezas listas para empacarse.

Este sistema de AGV ha automatizado la operación de transporte de los productos, como consecuencia, mejora la eficiencia total de la planta gracias a la respuesta oportuna a las demandas.

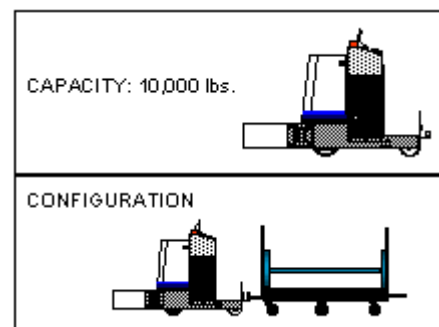
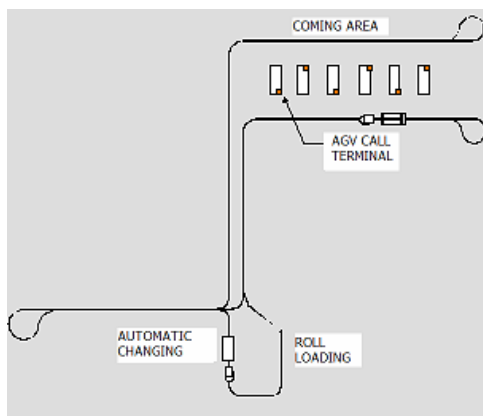
| Información Del Sistema |                                                             |                                                                                              |
|-------------------------|-------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| Estadística             | Controles                                                   | Ventajas                                                                                     |
| Vehículos: 1            | • Navegación inalámbrica del AGV                            | • Interfaz automática con las máquinas de la producción                                      |
| Estaciones: 26          | • PC del operador del AGV                                   | • Reducción del trabajo del transporte, y del riesgo durante la transportación del producto. |
| Trayectoria: 800 pies   | • Sistema de comunicación                                   | • Reducción en costos de operación                                                           |
|                         | • Sistema gráfico que supervisa y muestra el funcionamiento |                                                                                              |

## EJEMPLO NO. 2:

### Vehículo de Remolque en la Unidad Textil Manufacturera Manufacturing/Textile

El sistema transporta los rodillos de tela del área de producción a la zona de corte y posteriormete a los almacenes. Cada vehículo cuenta con un sistema de acoplamiento para cargar y descargar un rodillo del paño. El área del corte cuenta con un sistema de detección múltiple que se comunica con las terminales que llaman al AGV. De no recibir ningún llamado el AGV espera en la estación de carga de producción hasta que reciba una llamada de un nuevo pedido de algún rodillo de tela que se entregará en una estación específica.

Así mismo, cuando un operador termina el proceso de producción de un rodillo, utilizará la terminal para llamar otro AGV para que otra carga sea entregada. Por otra parte, mientras un AGV espera en el área de producción, es cargada su batería, posteriormente un operador coloca en el AGV el siguiente rodillo, le indica su nuevo destino y finalmente acciona el inicio de su procedimiento. El AGV también puede recibir su destino desde la PC del operario del vehículo sin que el AGV se encuentre en el área de producción, es decir, a distancia. En la estación de corte, el operador quita la carga y coloca en el AGV los rodillos que se han desocupado. Posteriormente, por medio del panel a bordo del AGV selecciona el almacén de destino y envía al AGV. En el almacén otro operador quita las cargas y envía el AGV de nuevo a la estación de producción.



| Información Del Sistema                                       |                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                            |
|---------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Estadística                                                   | Controles                                                                                                                                                                                                                                                                                                     | Ventajas                                                                                                                                                                                                                                                                                   |
| Vehículos: 2<br>Estaciones: 18<br>Trayectoria:<br>1, 850 pies | <ul style="list-style-type: none"> <li>• Navegación inalámbrica del AGV</li> <li>• PC Del Encargado Del Vehículo</li> <li>• 32 estaciones de destino.</li> <li>• Cinco terminales de la llamada de AGV</li> <li>• Carga automática de la batería de AGV</li> <li>• Sistema de configuración rápida</li> </ul> | <ul style="list-style-type: none"> <li>• Transporte automático de los rodillos completos y vacíos del paño</li> <li>• Eliminación del recorrido de abastecimiento del vehículo</li> <li>• Reducción del trabajo del transporte.</li> <li>• Entregas programadas justo a tiempo.</li> </ul> |

### EJEMPLO NO. 3

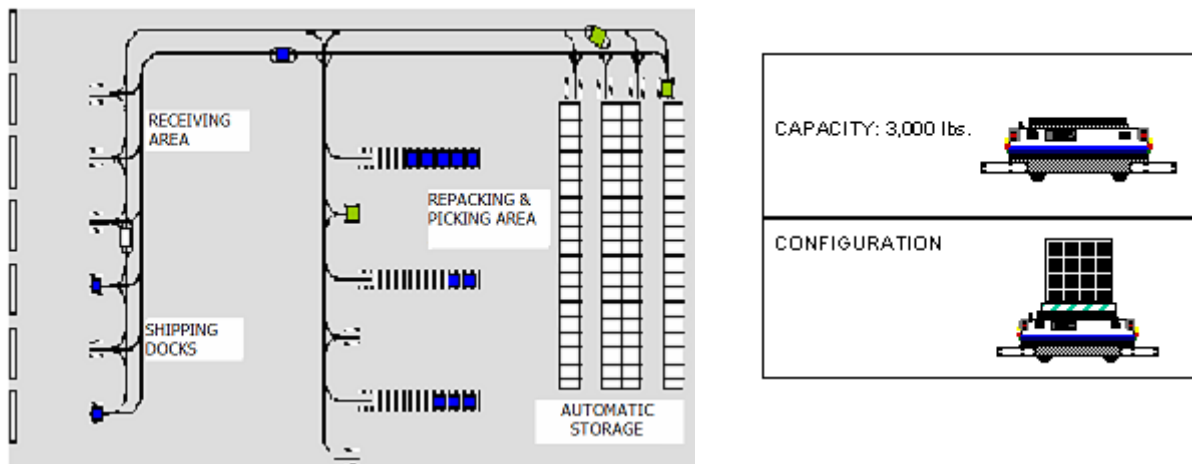
#### Portador de la Carga de la Unidad de Almacenamiento y Distribución Warehousing / Distribution

En este caso se tienen AGVs que cuentan con mecanismos de elevación y descenso que se utilizan para transportar las plataformas entre los muelles del carro a la zona de almacenaje y a las áreas de distribución. Las plataformas que se transportarán cuentan con un sistema de control para su identificación, con dicho sistema, por medio de la PC del operario de distribución se comunica el destino de la plataforma a la PC del encargado del vehículo.

El encargado del vehículo es responsable de seleccionar el AGV más cercano, y así asignar la siguiente ruta para el transporte de la siguiente plataforma. Los AGVs comunican constantemente su localización, si ellos están cargados o vacíos, el estado de la batería y su destino. Un AGV seleccionado por el encargado del vehículo se ordena a la estación apropiada de la recolección de la plataforma. A su llegada, los impulsores del AGV elevan su cubierta, levantando la plataforma del soporte, y posteriormente se les indica su destino.

En el resto de las áreas, la carga es recibida por el sistema automatizado de bandas transportadoras o de almacenaje.

El sistema, con cuatro AGVs se diseña para apoyar hasta 50 plataformas por hora y el reembolso del sistema es de aproximadamente dos años.



### Información Del sistema

| Estadística                                               | Controles                                                                                                                                                                                                                                       | Ventajas                                                                                                                                                                                                                                                                                            |
|-----------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Vehículos: 4<br>Estaciones: 36<br>Trayectoria: 1,100 pies | <ul style="list-style-type: none"> <li>Navegación inalámbrica del AGV</li> <li>PC Del Encargado Del Vehículo y del destinatario</li> <li>Recorrido bidireccional de AGV</li> <li>Soportes estáticos y automáticos del transportador.</li> </ul> | <ul style="list-style-type: none"> <li>Transporte automático de la recepción y envío del producto.</li> <li>Interfaz automatizada el área de almacenaje</li> <li>Reduce riesgos durante el transporte del producto</li> <li>Seguimiento en tiempo real de todas las cargas transportadas</li> </ul> |

Es prudente señalar que este apartado se ha dedicado a brindar una visión acerca de los diferentes tipos de robots existentes, según sus funciones y condiciones de trabajo, además de dar a conocer las herramientas tecnológicas para sistemas de automatización, principalmente enfocados al sector industrial.

Cabe mencionar que la mayor parte del texto ha sido enfocado al tipo de robot y sistema de distribución que se empleará a lo largo de este proyecto donde se demostrarán las aplicaciones y soluciones a los temas expuestos anteriormente.



## **DISEÑO, ELABORACIÓN Y LÓGICA DEL CONTROL DEL MONTACARGAS.**

### **1.1 Diseño y fabricación del montacargas autoguiado.**

Dentro de este capítulo se desarrollará el diseño y fabricación del prototipo AGV donde también se incorporará el diseño del órgano terminal para levantar los paquetes y posteriormente se añadirán los demás elementos de procesamiento, y con la unión de todos estos subsistemas se llegará a la elaboración final del montacargas.

El montacargas contará con una tracción omnidireccional, con dicha tracción se conseguirá tener un buen manejo en cuanto a la toma de decisiones para negociar las vueltas a lo largo de la trayectoria, todo esto partiendo desde el sistema de control que se le implementará y que será desarrollado en apartados subsecuentes.

Para la realización del AGV, es decir, la parte que únicamente comprende el sistema que dotará de movimiento al montacargas, se tomaron en cuenta todos los factores que pudieran intervenir al momento de realizar dicha función, entre ellos se encuentran principalmente: la forma de la ruta, el tamaño de las bandas del sistema de distribución y el tipo de producto a trasladar con sus dimensiones y peso máximo, con estos parámetros se eligió un carro previamente fabricado, que pudiera ser utilizado dentro de estas condiciones y que además por su configuración se le pudieran añadir los componentes de percepción, procesamiento de la información y comunicación para que poco a poco pudiera ser acondicionado como AGV y posteriormente incluir el órgano terminal para levantar los paquetes y así llegar a ser un montacargas.

En la Figura 1.1, se presenta el modelado final del carro, dicho modelado fue realizado por computadora con el software Solid Edge. El modelar el auto y sus

componentes, permitirá tener una visualización previa a la construcción de todo el conjunto y así minimizar errores de manufactura por no dimensionar de manera adecuada las partes que se le irán añadiendo al auto conforme se vaya fabricando.

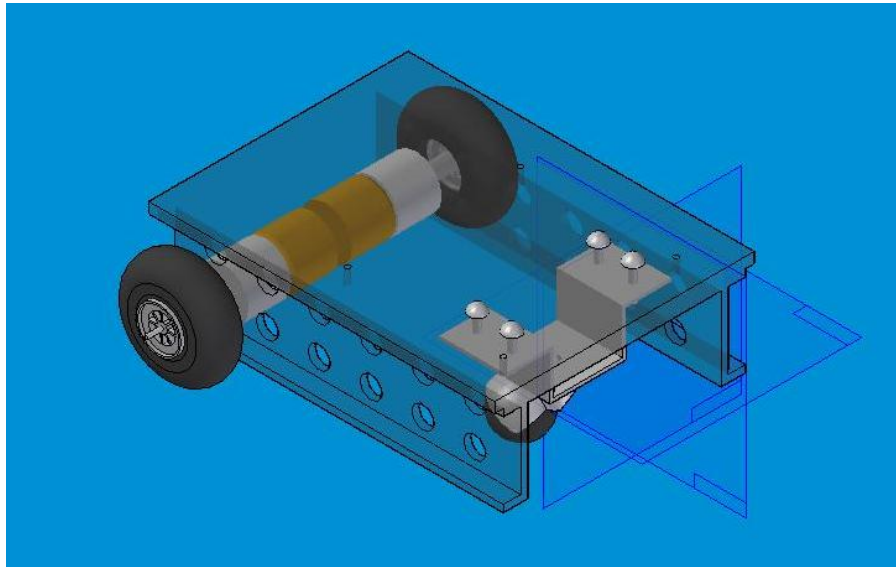


Fig. 1.1, Vista en isométrico

De esta manera el carro cuenta con las siguientes dimensiones en su plataforma de apoyo: 200[mm] de largo, 150[mm] de ancho, y finalmente 78 [mm] del piso a la parte plana del carro, (Fig. 1.2 y Fig. 1.3.)

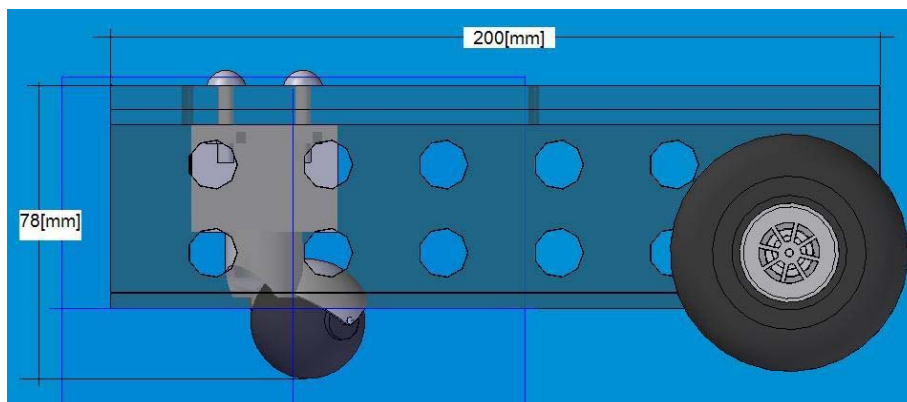


Fig. 1.2, Vista lateral

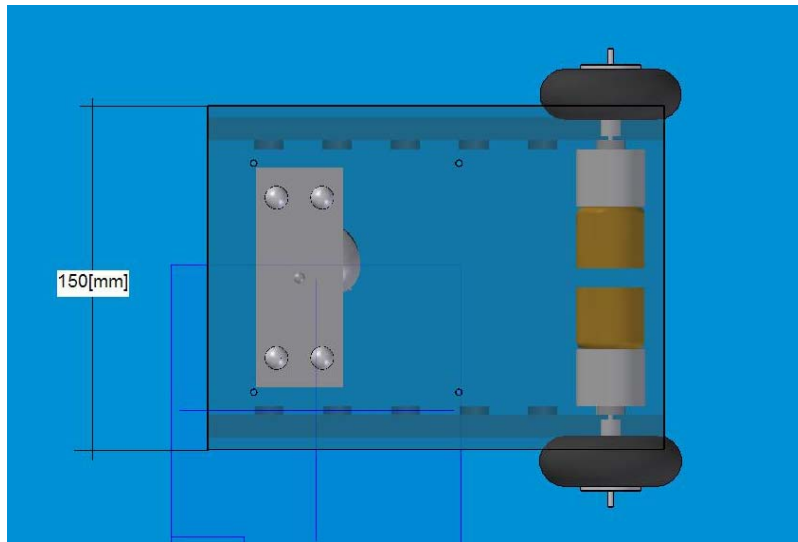


Fig. 1.3, Vista superior

El carro cuenta con un par de motores que brindarán potencia a la tracción, los motores están acoplados a un sistema de engranes que se encargará de reducir las revoluciones de salida del eje y por consiguiente aumentar el par de tracción de salida a las llantas, al frente se encuentra una rueda loca que proporcionará soporte a la carga del producto y del montacargas mismo, gracias a que se contará con un punto de contacto entre el AGV y la pista. La rueda loca se ubica lo más al frente posible sin intervenir en el espacio designado para el sistema de percepción que se colocará al frente de la rueda loca. Aunado a esto, el tener una rueda loca permite que la negociación de las vueltas esté únicamente controlada por la tracción omnidireccional.

Como se pudo observar, el carro tiene una superficie plana en la sección superior, es ahí donde se colocarán todos los elementos de procesamiento y el órgano terminal. Por otra parte, las baterías que energizarán tanto a los motores como al microprocesador, se encontrarán debajo de esta plataforma, ambos componentes, el microprocesador junto con su tableta fenólica, así como los "presets" o calibradores de los sensores ópticos del AGV y las baterías estarán ubicados en la parte posterior de la plataforma con la finalidad de desplazar hacia atrás el centro

de gravedad de todo el montacargas para que no se desestabilice al momento de levantar los paquetes y mucho menos al momento de realizar los traslados que se le ordenen.

En la Figura 1.4, se esquematiza de manera general la ubicación de estos componentes y la base final sobre la que se vendrá trabajando en los siguientes subtemas.

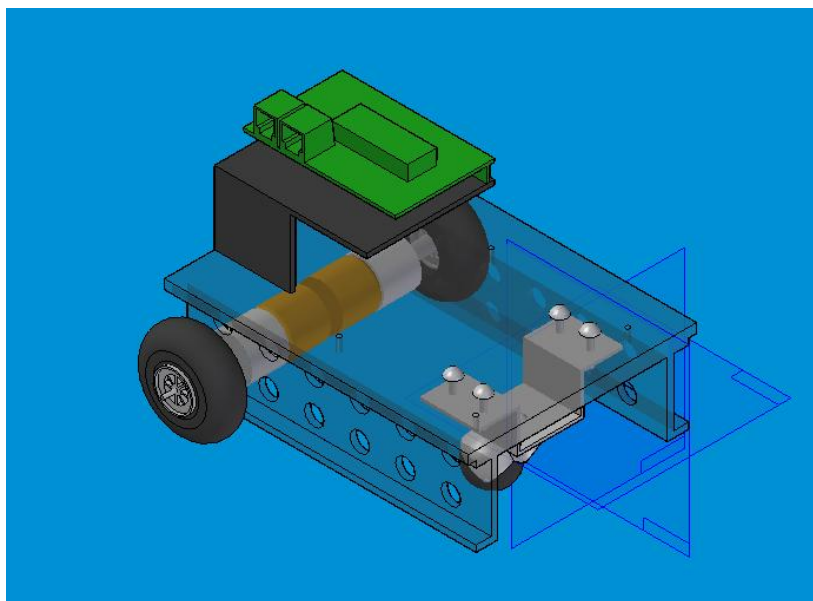


Fig. 1.4. AGV y componentes



## 1.2 Diseño del órgano terminal.

El objetivo del órgano terminal, es el de elevar desde el suelo la carga, a la salida de las bandas transportadoras, mantenerla a una altura adecuada sobre el piso y a esa altura transportar los paquetes, para que así libre sin problemas las

imperfecciones de la superficie, finalmente el órgano terminal debe descargar el producto al terminar su trayectoria.

Para realizar esta función se necesitó diseñar una trayectoria de levantamiento, esta trayectoria debía involucrar parámetros de traslación y de rotación de la carga, con la que se debía levantar al producto y alejarlo un poco del frente del carro, todo el traslado debe ser horizontal.

La trayectoria describe los siguientes puntos y ángulo de inclinación ( $\beta$ ), esto tomando como origen el borde del frente del AGV en la vista lateral, estos puntos se ubican a la altura de la plataforma del coche, (Fig. 1.5), y posteriormente será trasladada la trayectoria hasta la altura del suelo por propiedades del cuerpo rígido y así cumplir las condiciones de levantamiento:

$$R_0 = (-2, 5) \quad \beta = 0^\circ$$

$$R_1 = (-4, 6) \quad \beta = 0^\circ$$

$$R_2 = (-5, 8) \quad \beta = 0^\circ$$

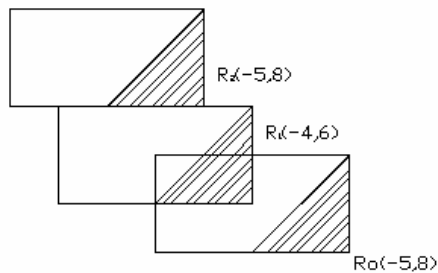


Fig. 1.5, Trayectoria de levantamiento

Se consideró que la manera más segura de realizar esta función era con un “mecanismo de cuatro barras”, ya que es una herramienta sencilla, de diseño confiable y principalmente porque permite desarrollar la trayectoria que se planteó.

Para determinar las dimensiones y ubicación de las barras se utilizó el método de: Síntesis de Conducción de Cuerpo Rígido, este método permite trazar una trayectoria de desplazamiento, tomando en cuenta ubicación e inclinación, a lo largo de tres puntos.

Por medio de la siguiente ecuación:

$$|a_0 - b|^2 = |r_n - b + (a_0 - r_0)e^{i\beta n}|^2$$

Se va conformando un sistema con las condiciones, con  $n=1$  y  $n=2$ , es decir, desde la posición del primer punto a la del segundo ( $b$ ), donde  $n=0$  ( $r_0$ ) es la posición inicial del movimiento, posteriormente el desarrollo entrega los puntos de contacto entre la barra dos y la barra acopladora y de igual forma entrega los puntos para el contacto de la barra 3 con la barra de salida. El único requisito es el de proponer los parámetros de la barra 1, es decir, los puntos en tierra ( $a_0$ ).

Con base en este método se tabularon diferentes puntos en tierra, para observar qué resultados entregaba el sistema sobre las otras barras, una vez habiendo observado el comportamiento se propuso que los puntos en tierra fueran (2,2) y (11,7), (Fig. 1.6).

Teniendo así la trayectoria y los puntos en tierra, el desarrollo para encontrar los demás parámetros es:

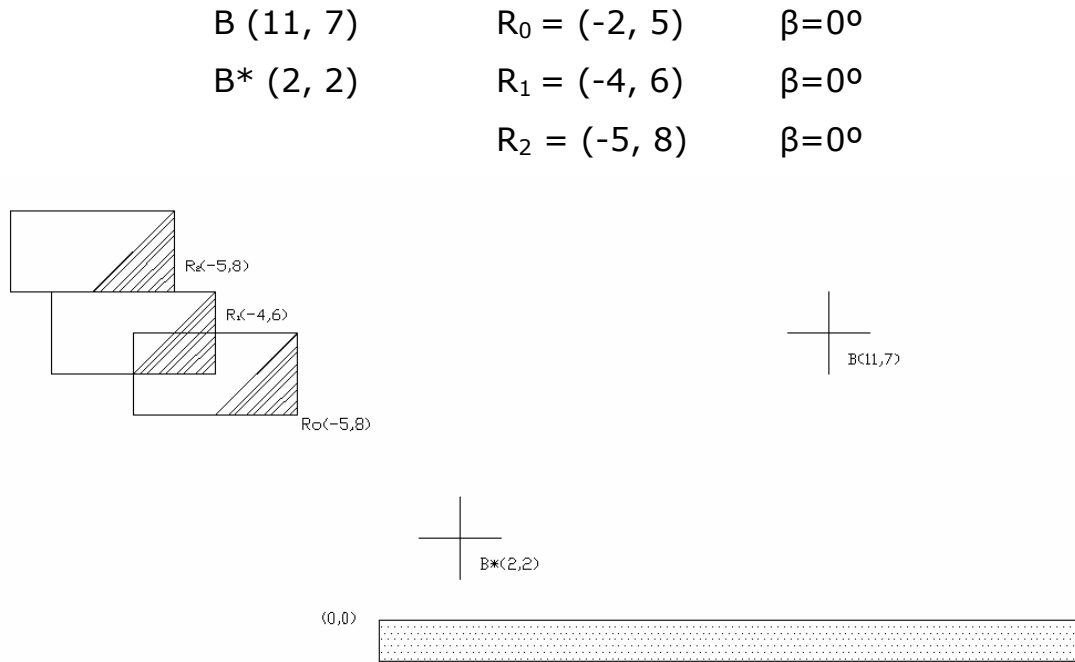


Fig. 1.6, Puntos de la barra 1 (tierra)

$$|a_0 - b|^2 = |r_n - b + (a_0 - r_0)e^{i\beta n}|^2$$

$$n = 1$$

$$|a_x + a_y - 11 - 7i|^2 = |-4 + 6i - 11 - 7i + a_x + a_y i + 2 - 5i|^2$$

$$(a_x - 11)^2 + (a_y - 7)^2 = (-13 + a_x)^2 + (a_y - 6)^2$$

$$a_x^2 - 22a_x + 121 + a_y^2 - 14a_y + 49 = 169 - 26a_x + a_x^2 + a_y^2 - 12a_y + 36$$

$$4a_x + 170 = 205 + 2a_y$$

$$4a_x = 35 + 2a_y \dots \dots \dots (1)$$

$$n = 2$$

$$|a_x + a_y - 11 - 7i|^2 = |-5 + 8i - 11 - 7i + a_x + a_y i + 2 - 5i|^2$$

$$(a_x - 11)^2 + (a_y - 7)^2 = (-14 + a_x)^2 + (a_y - 4)^2$$

$$a_x^2 - 22a_x + 121 + a_y^2 - 14a_y + 49 = 196 - 28a_x + a_x^2 + a_y^2 - 8a_y + 16$$

$$-6a_x = 33 + 6a_y$$

$$a_x = \frac{11}{3} + a_y \dots \dots \dots (2)$$

Sustituyendo la ecuación (2) en (1) y resolviendo el sistema, para obtener el punto de contacto de la barra 2 con la barra acopladora:

$$4\left(\frac{11}{3} + a_y\right) = 33 + 2a_y$$

$$\frac{44}{3} + 4a_y = 33 + 2a_y$$

$$2a_y = 18\frac{1}{3}$$

$$a_y = \frac{55}{6}$$

$$a_y = 9.166[cm]$$

$$a_x = 12.833[cm]$$

Desarrollando para el siguiente punto en tierra:

$$|a_0 * -b|^2 = |r_n - b + (a_0 * -r_0)e^{i\beta n}|^2$$

$$n = 1$$

$$|a_x * +a_y * -2 - 2i|^2 = |-4 + 6i - 2 - 2i + a_x * +a_y * i + 2 - 5i|^2$$

$$(a_x * -2)^2 + (a_y * -2)^2 = (-4 + a_x *)^2 + (a_y * -1)^2$$

$$a_x *^2 - 4a_x * + 4 + a_y *^2 - 4a_y * + 4 = 16 - 8a_x * + a_x *^2 + a_y *^2 - 2a_y * + 1$$

$$4a_x * = 2a_y * + 9 \dots \dots \dots (3)$$

$$n = 2$$

$$|a_x * +a_y * -2 - 2i|^2 = |-5 + 8i - 2 - 2i + a_x * +a_y * i + 2 - 5i|^2$$

$$(a_x * -2)^2 + (a_y * -2)^2 = (-5 + a_x *)^2 + (a_y * -1)^2$$

$$a_x *^2 - 4a_x * + 4 + a_y *^2 - 4a_y * + 4 = 25 - 10a_x * + a_x *^2 + a_y *^2 - 2a_y * + 1$$

$$-6a_x * = 18 + 2a_y$$

$$3a_x * = 9 + a_y \dots \dots \dots (4)$$

Sustituyendo (4) en la ecuación (3) y resolviendo el sistema para obtener el punto de contacto de la barra acopladora con la barra 4:



$$4\left(3 + \frac{a_y^*}{3}\right) = 9 + 2a_y^*$$

$$12 + \frac{4}{3}a_y^* = 9 + 2a_y^*$$

$$9a_y^* = 2$$

$$a_y^* = 4\frac{1}{5}$$

$$a_y^* = 4.5[cm]$$

$$a_x^* = 4.5[cm]$$

Con estos resultados, se obtiene el largo de la barra acopladora y desde cada punto se extienden dos líneas hasta el punto inicial de la trayectoria (Fig. 1.7).

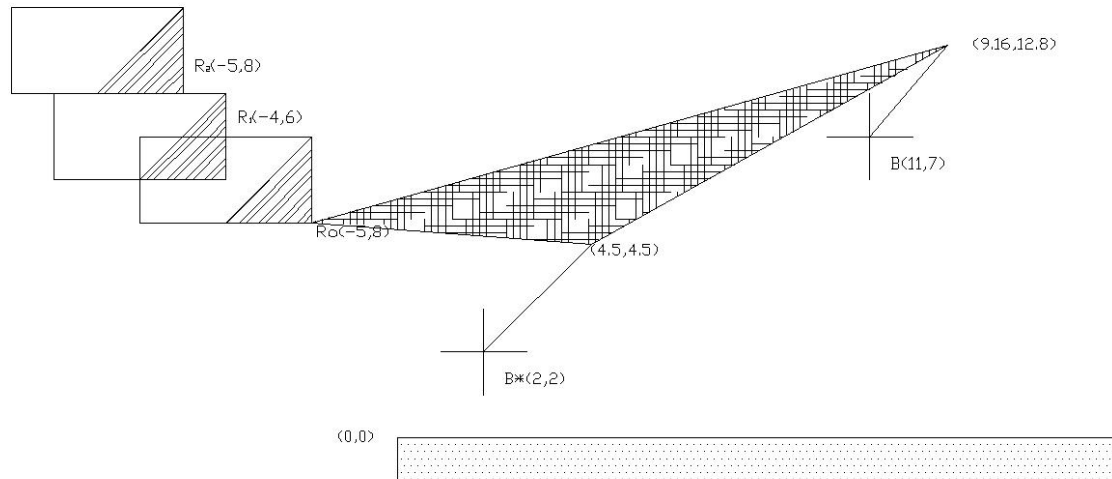


Fig. 1.7, Mecanismo de cuatro barras

Ahora que el mecanismo es capaz de realizar el movimiento que implica la trayectoria, por propiedades de cuerpo rígido, se trasladará hasta el suelo el extremo de la barra acopladora, justo donde sea posible poder levantar los paquetes sin golpearlos y sin tocar el suelo Fig. 1.8.

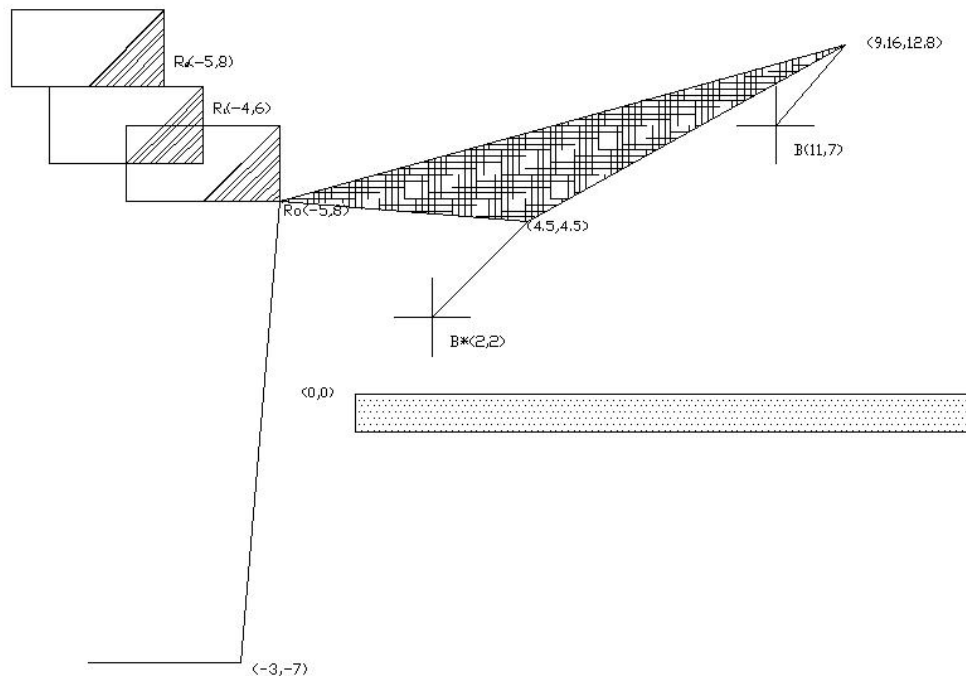


Fig. 1.8, Plano del órgano terminal

Finalmente, se tiene que dar un ancho suficiente que soporte de buena manera al mecanismo, además del punto debe contar con una superficie de contacto con el producto lo suficientemente amplia para que los paquetes sean trasladados de una manera segura, entonces, como el volumen de cada paquete será de  $4[\text{cm}] \times 4[\text{cm}] \times 4[\text{cm}]$  tomando en cuenta estas condiciones, y junto con el mecanismo de levantamiento, el órgano terminal tendrá el siguiente diseño (Fig 1.9):

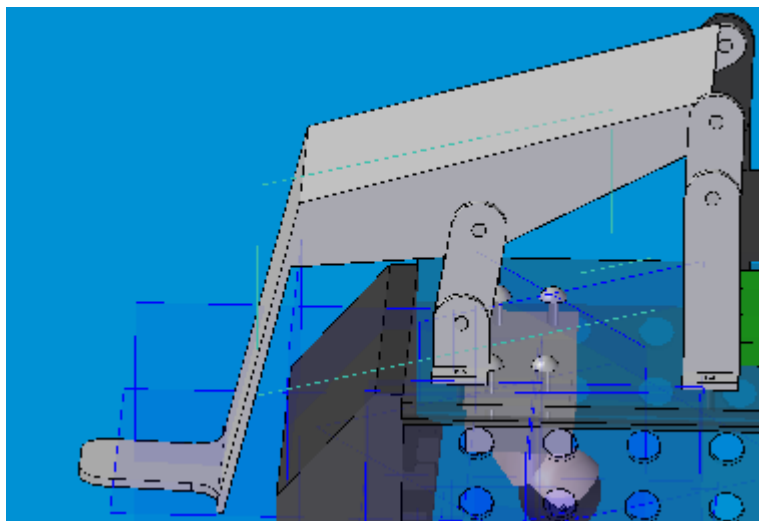


Fig. 1.9, Modelado definitivo del órgano terminal

### **1.2.1 Capacidad de carga del órgano terminal.**

El órgano terminal debe contar con la resistencia suficiente para poder soportar la carga que se le aplique sin que se vea afectado en su desempeño. Teniendo como objetivo lo anterior, se analizaron las ventajas y desventajas de varios tipos de materiales que fueran capaces de responder a esta necesidad, plásticos y metales principalmente conformaron esta lista.

La evaluación para elegir el material involucró aspectos de diseño y de resistencia, para eso se realizaron varios análisis de esfuerzo en el software Unigraphics en la sección de Estructuras. Resultando elegida una lámina de acero comercial C1015 de 1mm de espesor para que se encargara de realizar este trabajo.

Como se podrá observar en el diseño, esta lámina no comprenderá las secciones laterales del montacargas, es decir, las partes en contacto con el mecanismo de cuatro barras, esto se decidió tomando en cuenta que el acero tiene una alta densidad e incrementaría en gran medida el peso del montacargas. Las secciones laterales, así como el mecanismo de cuatro barras serán fabricados con PVC espumado, el cual brindará soporte y ligereza con respecto a los demás materiales utilizados, permitiendo también facilidad en la movilidad de las articulaciones de cada barra.

Una vez habiendo modelado el órgano terminal en Solid Edge, se exportó la pieza a Unigraphics, donde se especificaron las condiciones de trabajo:

Temperatura: 23 [°C]  
Carga: 1 [kg] por efecto de la gravedad (negativo en eje el "Z")

De igual forma se le proporcionaron al programa las propiedades del acero laminado C1015:

Densidad: 7870 [kg/m<sup>3</sup>]  
Módulo de Poisson: 0.29  
Módulo de Elasticidad: 200 [GPa]  
Esfuerzo último: 420 [MPa]

Posteriormente se realizó el mallado de elemento finito y se procedió al análisis, donde los resultados obtenidos fueron (Fig. 1.10 y Fig. 1.11):

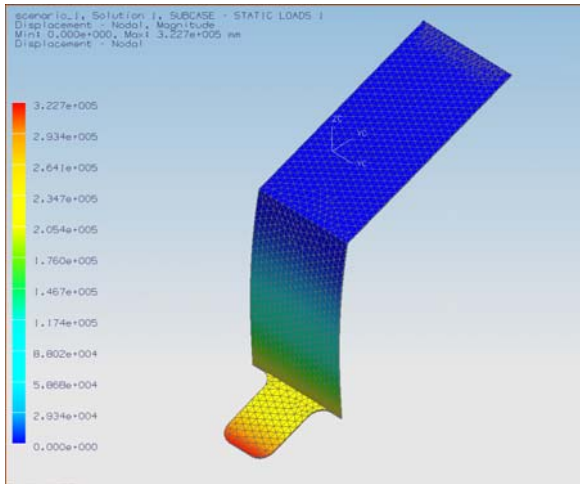


Fig. 1.10, Desplazamiento

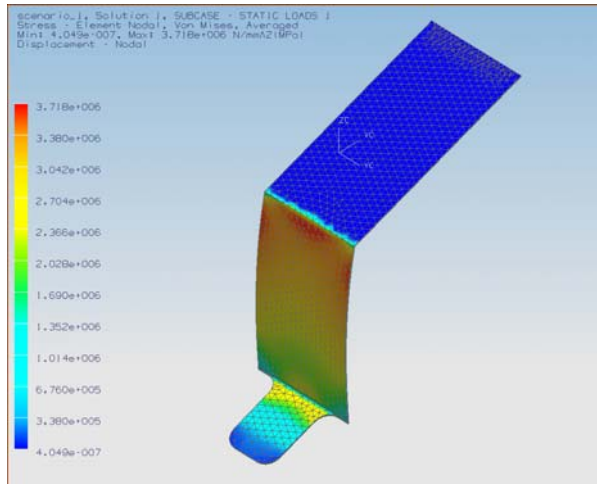


Fig. 1.11, Esfuerzo cortante

Desplazamiento total:  $3.227 \times 10^{-5}$  [m]

Esfuerzo cortante: 3.71 [MPa]

Dividiendo el esfuerzo último entre el esfuerzo aplicado, se obtiene como resultado un factor de seguridad de:

$$N = 113.2$$

Aunado a este resultado se debe recordar que el órgano terminal cuenta a los lados con los soportes de la barra acopladora, los cuales además de brindar estabilidad a la carga, incrementan el soporte a la lámina de acero.

### 1.3 Sistema electrónico del AGV.

El sistema electrónico del robot (en conjunto con la lógica del movimiento) es la parte más importante del diseño del AGV, éste brindará la capacidad de visión, percepción e interacción con su entorno.

Gracias a esto se tendrá la capacidad de conocer la dirección del vehículo en cada instante, el estado del movimiento así como la capacidad de medir el error que se va cometiendo en el seguimiento de la ruta.

El sistema sensorial está conformado por tres dispositivos de percepción óptica, los cuales deberán de ser ubicados en la parte frontal del vehículo para que la respuesta del robot sea lo más rápida posible y corrija el error que pueda presentarse durante el seguimiento de la ruta. Estos sensores deberán de estar aislados de todo tipo de luz que pueda alterar la lectura de la pista, de esta manera se realizó la construcción de una cubierta que se encargará de envolver los sensores (Fig. 1.12) y así poder aumentar la sensibilidad y disminuir los cambios bruscos en la percepción del vehículo.

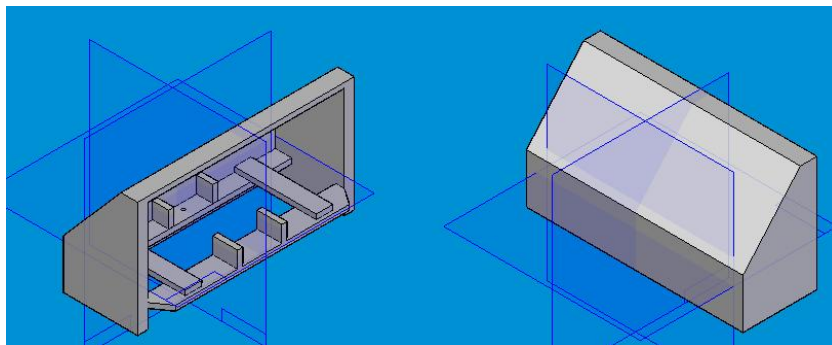


Fig. 1.12, Cubierta de los sensores

En este caso se utilizará el sensor con fototransistor reflexivo a objetos con número de serie QRB1113 de Fairchild<sup>1</sup> (Fig. 1.13). Este dispositivo consiste en un diodo infrarrojo emisor y un fototransistor de silicio NPN montados en cada lado

---

<sup>1</sup> QRB1113, Apéndice I

del sensor y que convergen en un eje óptico. El fototransistor responde a la radiación del diodo emisor cuando un objeto reflejante pasa por su campo de visión. Para que la respuesta de los sensores sea la óptima, deberán de estar montados a un centímetro del piso. El fotosensor cuenta con un filtro de luz natural que disminuye el error de visión.

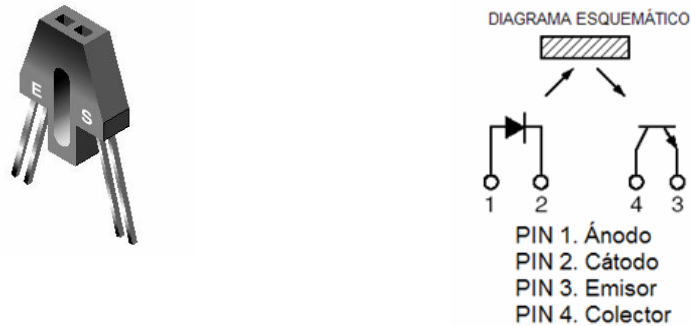


Fig. 1.13, Sensor Fairchild QRB1113

Cabe destacar que la alimentación de los sensores y la alimentación de todos los circuitos en general es provista por la tarjeta en la que se encuentra el microcontrolador, ésta nos provee de un potencial de 5 [V] y una respectiva tierra.

Los tres sensores deberán de estar orientados hacia el piso donde se desplaza el AGV detectando principalmente dos tipos de estados: blanco y negro. La facilidad de detectar sólo estados binarios nos ayudará más adelante a ocupar lógica digital para poder realizar la toma de decisiones. De esta forma, en el piso se encontrará una línea blanca que será la ruta a seguir, esta línea no deberá de ser más ancha que la distancia que exista entre cada sensor, este criterio de ubicación es con el objetivo de que en la mayoría de los eventos que se produzcan, sean únicamente detectados por un sólo sensor a la vez.

El desempeño de un montacargas dentro de una planta industrial involucra también aspectos propios de las condiciones de trabajo, principalmente lo

referente a la iluminación, la cual es de suma importancia por efectos de la detección de la línea reflejante que indica la ruta a los sensores, y que a su vez, en la mayoría de los casos, varía a lo largo del día debido al movimiento terrestre de rotación. Por esta razón, el sistema de visión deberá ser flexible, y así tener la capacidad de ser calibrado fácilmente a las condiciones de luminosidad durante sus jornadas de trabajo. Para poder regular la cantidad de luminosidad necesaria y que los sensores den validación a los estados lógicos, se necesita tener un valor de comparación entre el estado que ven los sensores y el estado base de luminosidad necesaria. Esto se realizará con un comparador que nos entregará una salida lógica, la cual se interpretará como la visión actual de los sensores.

El comparador que se ocupará en este caso es el LM339A<sup>2</sup> que es un comparador cuádruple de baja potencia (Fig. 1.14). Este dispositivo consta de cuatro comparadores independientes, con un voltaje de precisión tan pequeño como 2mV como máximo.

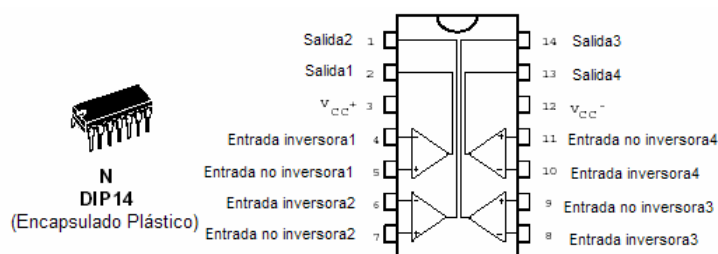


Fig. 1.14, Comparador cuádruple de baja potencia LM339A

En el siguiente diagrama se esquematiza el acoplamiento de un comparador con uno de los sensores (Fig. 1.15). La salida del sensor se conecta a la entrada no inversora de uno de los comparadores y el potenciómetro que dará el voltaje de referencia se conecta a la entrada inversora. El voltaje de entrada diferencial (diferencia de voltaje entre las terminales de entrada mencionadas) hará que el

<sup>2</sup> LM339A, Apéndice I

transistor del circuito a la salida prenda o apague dependiendo del voltaje de referencia, si la diferencia es positiva la salida irá a nivel alto y si la diferencia entre la entrada inversora y la no inversora es negativa también la salida irá al nivel alto.

La diferencia depende de manera directa del potenciómetro de 10 [k $\Omega$ ], por lo tanto este potenciómetro será el calibrador del campo de visión del sensor al que se le conecta.

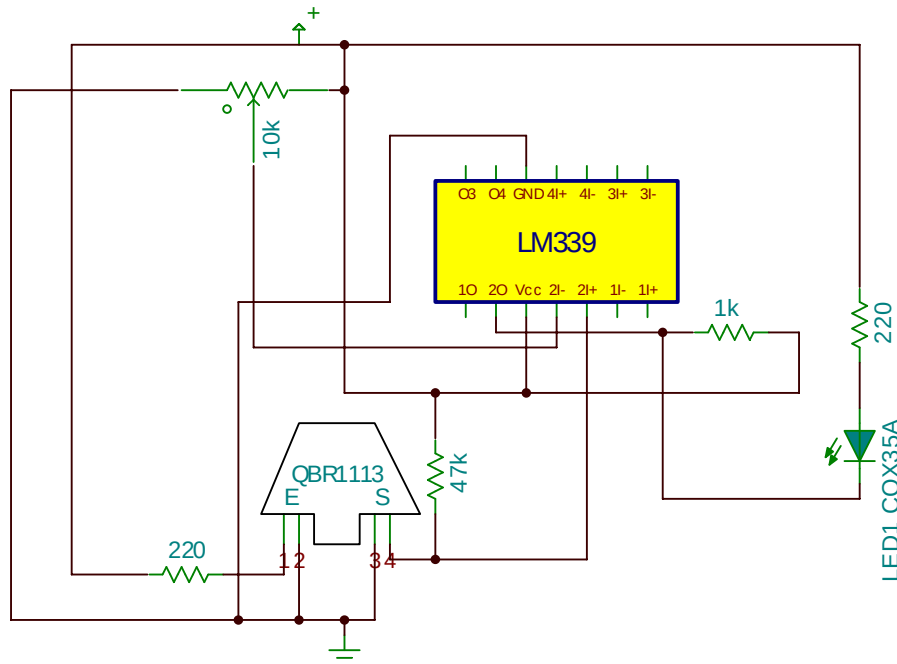


Fig. 1.15, Diagrama esquemático de un comparador sencillo.

Aún con el estado de comparación anteriormente mencionado, y para reducir errores en la lectura de los sensores, se implementará un circuito integrado más. Este circuito llamado Schmitt-Trigger<sup>3</sup> con el número de parte 7414 (Fig. 1.16), tiene la capacidad de fijar los valores de voltaje en estados de ceros y unos lógicos (0 [V] y 5 [V]). Así, dependiendo de la entrada que reciba, restaura el valor digital a un nivel lógico válido y lo invierte. Para valores de voltaje entre 0 [V] y 2.5 [V]

<sup>3</sup> Schmitt Trigger (SN74LS14), Apéndice I



la salida del circuito presentará 5 [V] y para valores entre 2.5 [V] y 5 [V] la salida que presentará será de un cero lógico (0 [V]).

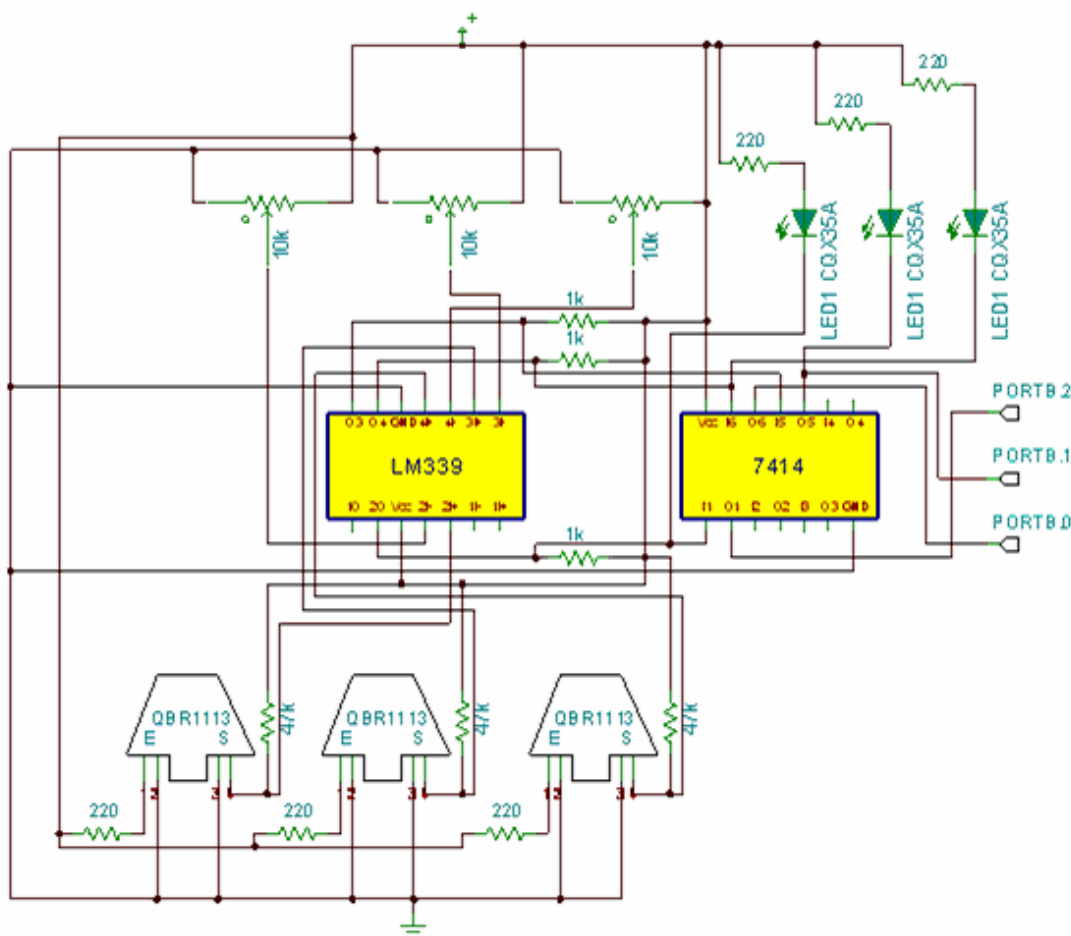


Fig. 1.16, Conexión eléctrica de la etapa de sensado y recepción de los valores.

La salida del Schmitt-Trigger es enviada directamente a los bits 0, 1 y 2 del puerto B del microcontrolador PIC16F877 para pasar a la etapa de procesamiento de la información, esta etapa así como su lógica de funcionamiento serán desarrolladas posteriormente.

Ahora que ya se ha explicado la forma en la que el vehículo realiza el sensado de su posición a lo largo de su trayectoria, se tratará la forma en que se responderá a los estímulos del medio.

Para que el prototipo pueda trasladarse y levantar el órgano terminal, se requieren elementos que realicen estos trabajos. Para la función de traslado se utilizará un par de motores de corriente directa de 12[V] cada motor contará con un sistema de engranes de reducción con la finalidad aumentar el torque de las ruedas, entregando 45[RPM] a la salida de la flecha. Y para mover el órgano terminal se utilizará un servomotor de 5[V] que es capaz de realizar el movimiento de la pinza con una velocidad de 0 a 60° en 0.22[s] con un torque de 174[oz/in] para 6 [V] (Fig. 1.17).



Fig. 1.17, Motores de CD y servomotores

Es pertinente recordar que el voltaje lógico que se está manejando en el microcontrolador es de 0 [V] y 5 [V] por lo cual necesitamos seis señales de salida para controlar los dos motores de corriente directa que proporcionarán el movimiento del robot. Dos de estas señales serán para el motor izquierdo, dos para el motor derecho y dos para habilitar o inhabilitar el movimiento de los motores (*enables*).

Para evitar saturar puertos dentro del microcontrolador únicamente se enviarán cuatro señales de salida. Una señal para cada motor y dos para los *enables*. Una señal de uno lógico para un motor de corriente directa significará que el motor debe de girar en sentido positivo (hacia adelante), caso contrario a cuando se le mande un cero lógico.

Las salidas para los motores serán entonces los bits 0 y 2 del puerto D del microcontrolador, estos controlarán el sentido de giro de los motores, además de los bits 1 y 2 del puerto C, los cuales habilitarán o inhabilitarán los motores.

Se sabe que para los motores de corriente directa se necesitan dos terminales para controlar su movimiento, una con un voltaje más elevado que otra para producir el movimiento del rotor. También se sabe que si se requiere cambiar el sentido de giro al motor sólo se debe de cambiar la polaridad de la alimentación.

Así, para controlar un motor con sólo una señal que nos envía el microcontrolador, debemos de poder crear una señal inversa para poder controlar el motor. Nos auxiliaremos de un inversor 74LS04M que es un inversor séxtuple<sup>4</sup>.

Al inversor se le enviarán las dos señales de movimiento de los motores que, como ya se mencionó, provendrán de los bits 0 y 2 del puerto D del microcontrolador. La siguiente tabla explica el sentido de movimiento para los diferentes casos de las señales del motor derecho:

| BIT 0 PUERTO D | INVERSOR | SENTIDO DE GIRO             |
|----------------|----------|-----------------------------|
| 5 [V]          | 0 [V]    | Motor derecho hacia delante |
| 0 [V]          | 5 [V]    | Motor derecho hacia atrás   |

Así se obtienen las dos señales que nos auxiliarán en el movimiento de los motores.

Debido a que las corrientes demandadas por los motores de tracción y los circuitos digitales son diferentes, se añadirá una etapa de potencia a los motores. Así mismo, la potencia suministrada a los motores deberá ser independiente a los

---

<sup>4</sup> Inversor 74LS04M, Apéndice I

componentes electrónicos para que sean protegidos de alguna descarga de corriente mayor a la que necesiten.

Primeramente, se debe de tomar en cuenta que son pequeñas las señales de salida de información que envía el microcontrolador, de 5 [V] y muy baja corriente, por lo cual utilizaremos un puente H para abastecer de energía a los motores.

El puente H ó L293D<sup>5</sup> (Fig. 1.18) posee la capacidad de poder mandar una salida de potencia entre 4 [V] y 36 [V] para voltajes de entrada de -3 [V] a 7[V]. Naturalmente, para entradas de 0 [V], la salida en el puente H será del voltaje mínimo de alimentación 0 [V] y para entradas de 5 [V] mandará el voltaje de potencia máximo requerido.

Además el circuito integrado L293D posee dos entradas para habilitadores (*enables*), estos interruptores son igualmente controlados con voltajes lógicos, 5 [V] para activado y 0 [V] para desactivado.

Las señales de entrada que se aplicarán al puente H serán: dos salidas de la dirección de los motores (provenientes del microcontrolador), sus complementos (provenientes del inversor), las señales de los *enables* (provenientes del microcontrolador), el voltaje al que se desea que trabajen los motores y el voltaje de alimentación del puente H que es de 5 [V] (provenientes de la tarjeta del microcontrolador).

---

<sup>5</sup> L293D, Apéndice I

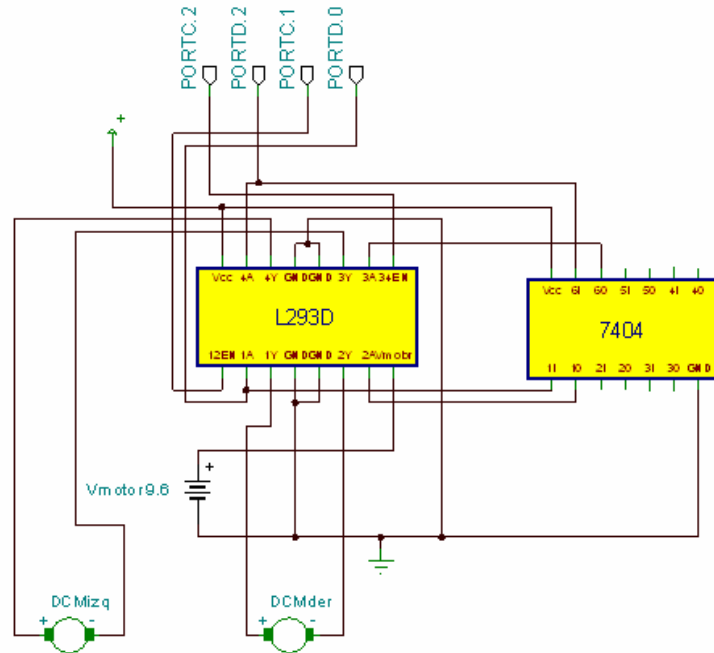


Fig. 1.18, Diagrama de conexión eléctrica de la etapa de potencia.

El voltaje de trabajo de los motores será suministrado por una batería de 9.6 [V] que se encontrará en la parte inferior del robot. Esta pila contará además con un regulador de voltaje que se podrá manejar con un potenciómetro para un mejor control de la alimentación de los motores.

Este circuito regulador de voltaje consta de un dispositivo regulador ajustable de tres terminales, con número de parte LM317, el cual está diseñado para proveer una corriente en la carga de más de 1.5 [A] con un voltaje de salida ajustable entre 1.2 y 37 [V] (Fig. 1.19). El circuito también cuenta en su configuración con un capacitor en la entrada y otro en la salida para eliminar picos en el voltaje y obtener una mejor respuesta transitoria, también contará con un potenciómetro para así poder ajustar el voltaje que se obtendrá a la salida del circuito regulador.

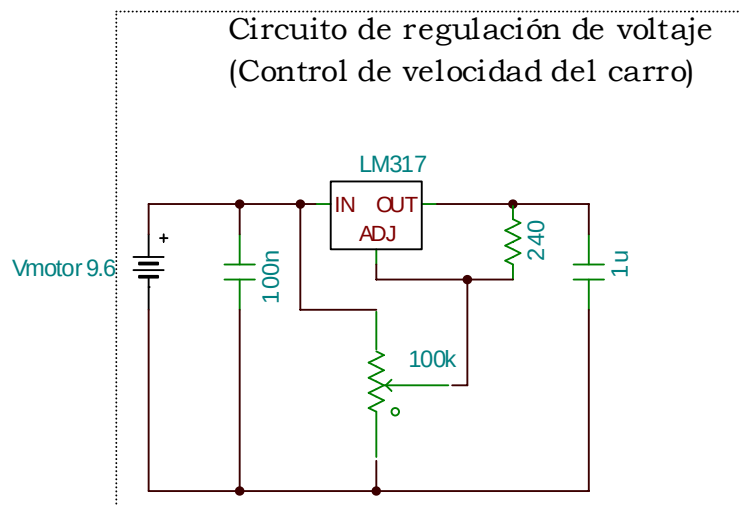


Fig. 1.19, Circuito de voltaje

El poder controlar el voltaje que llega a los motores de DC permite controlar también la velocidad a la salida de las ruedas, gracias a esto se evitarán posibles errores que se pudieran generar al sensar a altas velocidades. El circuito de potencia completo queda de la siguiente forma (Fig. 1.20):

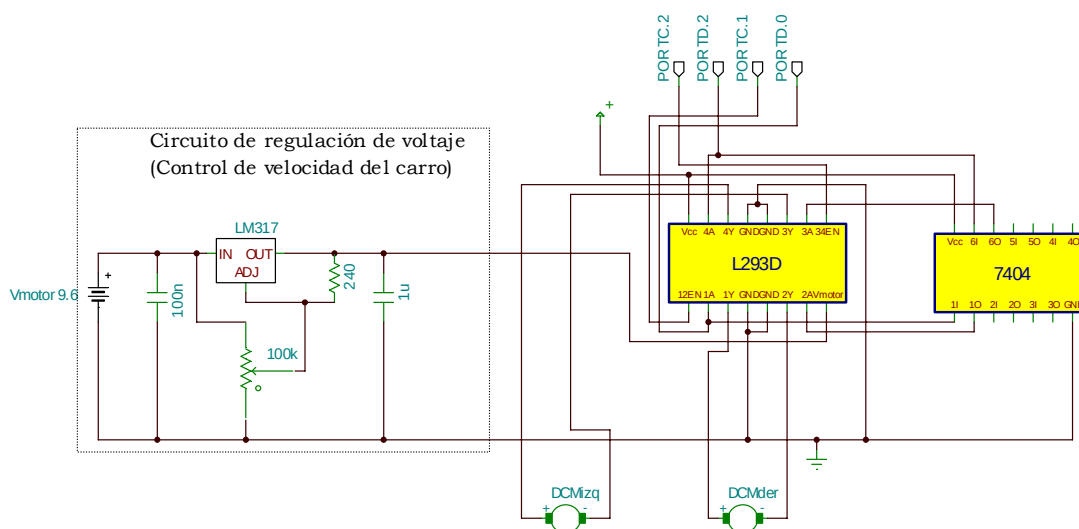


Fig. 1.20, Circuito de potencia.

Para los servomotores el control es más complejo. Ellos se controlan mediante tres distintas señales. Una de ellas es de 5[V] proveniente de la tarjeta del microcontrolador, otra de las señales es tierra y por último la señal de control.

La señal de control será proporcionada por el bit 6 del puerto D del microprocesador. El Servomotor que se empleará es el correspondiente al modelo HS-700BB de Hitec<sup>6</sup> que está diseñado para cumplir con las funciones para objetos de aeromodelismo y proyectos de robótica proporcionando toda la exactitud requerida para control de precisión. Se controla por modulación de ancho de pulso y tiene un rango de operación de voltaje de 4.8 a 6 [V]. Diagrama de conexión del servomotor (Fig. 1.21):

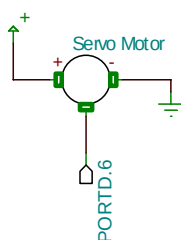


Fig. 1.21, Conexión del servomotor

Los colores de los cables en los motores de Hitec siguen una norma: negro para “tierra”, rojo (central) para “alimentación” y amarillo es la “señal de control”. Todos estos motores requieren de una señal de onda cuadrada de 3-5 [V] pico a pico, con una duración de pulso de 0.9 [ms] a 3.5 [ms] con 2 [ms] como centro. El pulso se refresca a 50 [Hz] (20 [ms]). Su sentido de giro es horario (CW).

Esta señal de control debe de ser enviada por un modulador de ancho de pulso (PWM *Pulse Width Modulator*) a una frecuencia de 20 [ms] por pulso. Los moduladores de ancho de pulso tienen la finalidad de dar una posición específica a los servomotores. El rango de movimiento de los servomotores varía entre 0 y 180° dependiendo del tiempo en que se mande la señal de voltaje.

---

<sup>6</sup> HS-700BB, Apéndice I

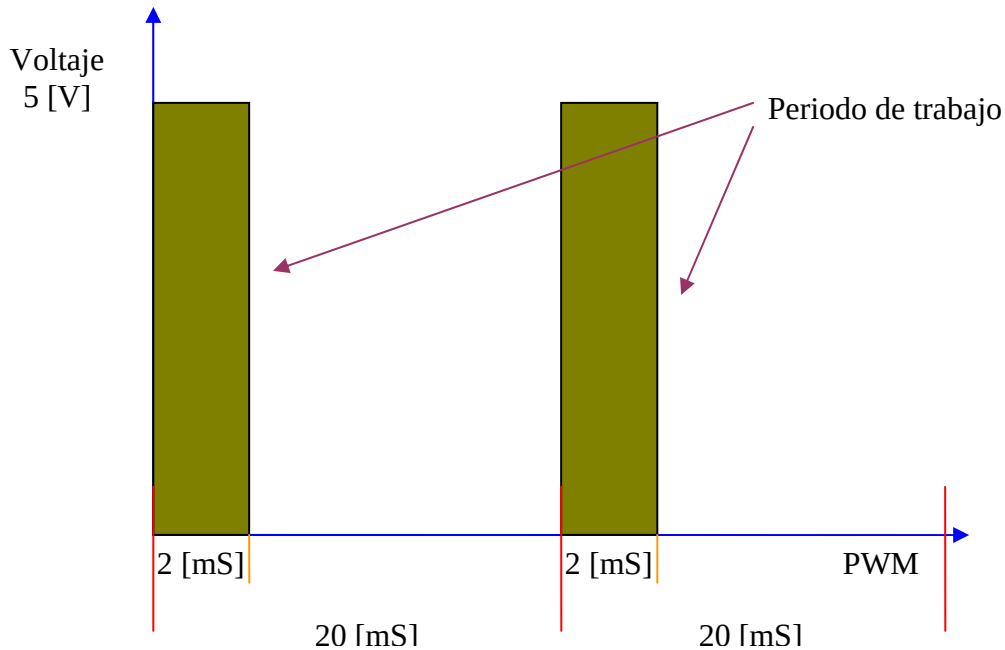


Fig. 1.22, Modulación de ancho de pulso para un servomotor

La Figura 1.22 ejemplifica el envío de la señal de control para un servo con un modulador de ancho de pulso. Durante los primeros 2 [ms] se envía una señal de 5 [V] por el puerto D y durante los siguientes 18 [ms] se envía una señal de 0 [V] por el mismo puerto.

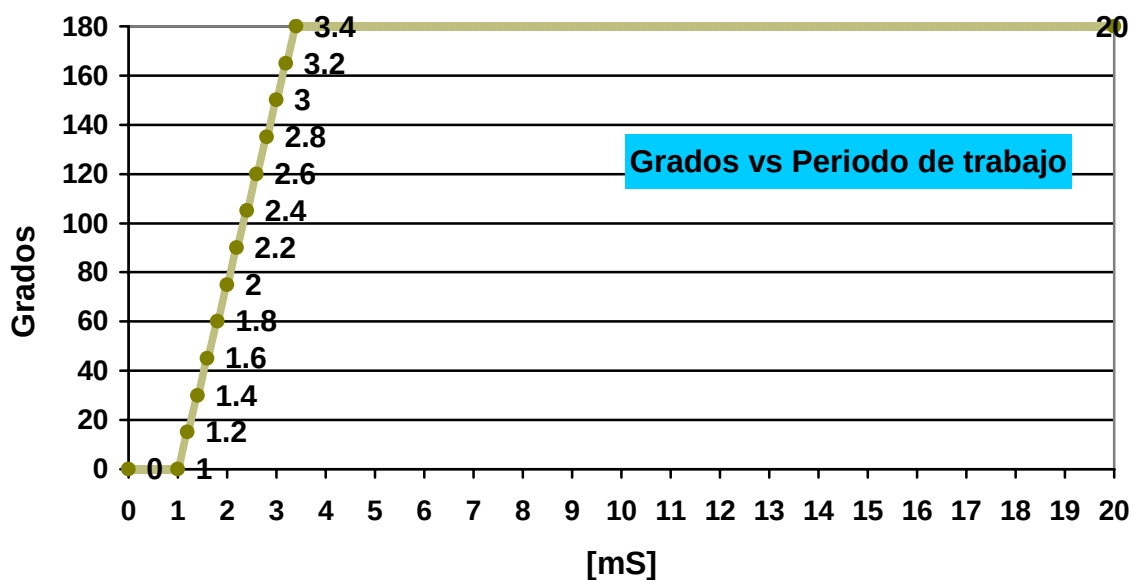
Al enviar de esta forma la señal, se completa todo el ancho de pulso que es de 20 [ms]. Con el periodo de trabajo (se refiere al tiempo donde se manda la señal de 5 [V]) se está enviando una señal para que el servomotor aumente su desplazamiento en grados. Desafortunadamente esta escala no es de tipo lineal.

El periodo de trabajo de un servomotor varía de 1 [ms] a 3.4 [ms] siendo su movimiento correspondiente a los siguientes periodos de trabajo:



| PWM     | PERIODO DE TRABAJO | GRADOS DE MOVIMIENTO |
|---------|--------------------|----------------------|
| 20 [ms] | 1 [ms]             | 0°                   |
| 20 [ms] | 1.2 [ms]           | 15°                  |
| 20 [ms] | 1.4 [ms]           | 30°                  |
| 20 [ms] | 1.6 [ms]           | 45°                  |
| 20 [ms] | 1.8 [ms]           | 60°                  |
| 20 [ms] | 2.0 [ms]           | 75°                  |
| 20 [ms] | 2.2 [ms]           | 90°                  |
| 20 [ms] | 2.4 [ms]           | 105°                 |
| 20 [ms] | 2.6 [ms]           | 120°                 |
| 20 [ms] | 2.8 [ms]           | 135°                 |
| 20 [ms] | 3.0 [ms]           | 150°                 |
| 20 [ms] | 3.2 [ms]           | 165°                 |
| 20 [ms] | 3.4 [ms]           | 180°                 |

Se puede apreciar que existen tres tipos de comportamiento para los servomotores, de 0 a 1 [ms] el resultado son 0°, de 1[ms] a 3.4 [ms] se comporta de forma lineal desde posición inicial de 0° hasta llegar a la posición final de 180°, y de 3.4[ms] a 20[ms] se queda en la posición de 180°. Esto se ejemplifica en la siguiente gráfica, (Fig 1,23) .



Fi

Fig. 1.23, Pulso vs desplazamiento

Como se observa para periodos de trabajo mayores a 3.4 [ms] el valor del ángulo desplazado se conserva en máximo ( $180^\circ$ ) y no cambiará a pesar que el periodo de trabajo sea del mismo ancho de pulso que el PWM.

El ángulo que se necesita para levantar la pinza dado el mecanismo de cuatro barras calculado es  $40^\circ$  por lo tanto se debe de utilizar un periodo de trabajo de 1.533 [ms]. La forma en la que se conseguirá enviar el periodo de trabajo indicado a través de puerto D del microcontrolador así como la lógica del movimiento se tratarán en el siguiente subtema.

#### **1.4 Lógica de movimiento del AGV**

Ahora que ya se ha tratado a profundidad la forma en la que el robot seguidor de línea recibe las señales para seguir su ruta y la manera en la que envía la corriente y voltaje necesario a los motores, se pasará a la manera en que el robot analiza e interpreta los datos que se le envían por medio de los sensores.

Para empezar debemos de hablar un poco de las características del microcontrolador que se utilizará así como de la tarjeta en la que está montado y que facilitará su funcionamiento.

El microcontrolador que se utilizará es un PIC 16F877 de Arizona Microchip<sup>7</sup>. Este dispositivo es un procesador de tipo RISC, es decir, con un conjunto reducido de instrucciones (su conjunto de instrucciones consta de 35 instrucciones) y arquitectura de tipo Harvard, por lo que cuenta con una memoria de datos y una de programa lo que hace posible un procesamiento encauzado, trayendo como ventajas la reducción de tiempo de procesamiento de los programas. Este tipo de arquitectura hace posible que todas las instrucciones se ejecuten en un solo ciclo de instrucción a excepción de los saltos que ocupan dos ciclos.

---

<sup>7</sup> PIC16F877, Apéndice II

Este dispositivo cuenta con periféricos muy útiles e importantes entre los cuales destacan: temporizadores, convertidor analógico-digital, comunicación serie (transmisión y recepción), puertos paralelos, memorias programables: 8 Kb de memoria flash y 256 bytes de memoria EEPROM, manejo de hasta 14 fuentes de interrupción, módulos de captura, comparación, PWM, entre otros.

El encapsulado que se empleará será del tipo PDIP de 40 pines (Fig. 1.24)

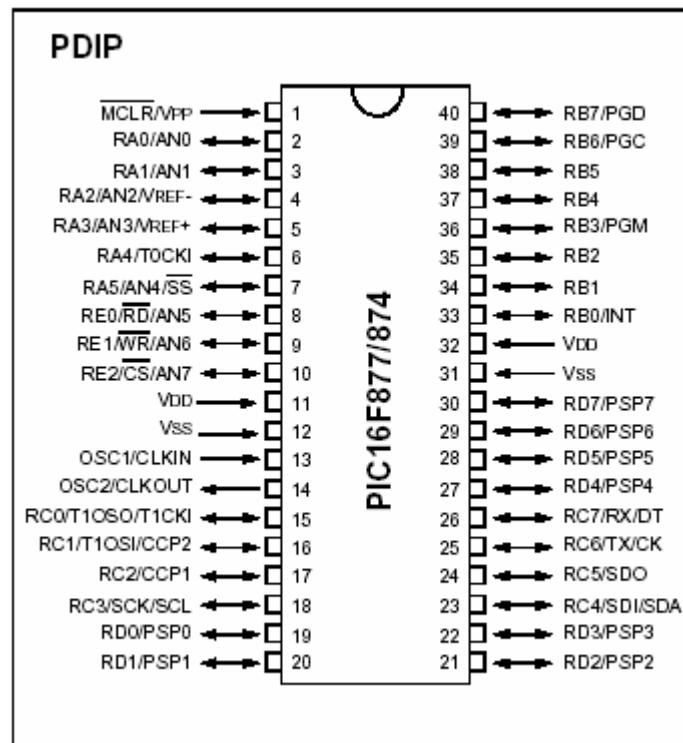


Fig. 1.24, Microcontrolador PIC16F877

De este diagrama se puede apreciar la ubicación de los 5 puertos paralelos del PIC, los cuales comparten su uso con funciones específicas del chip. El puerto A consta de 6 bits y comparte su uso con el convertidor analógico-digital; por ejemplo, el puerto B es de 8 bits y es por medio de este puerto que se realiza la programación de este dispositivo; el puerto C consta de 8 bits y comparte funciones de transmisión-recepción serie; el puerto D de 8 bits es también puerto paralelo esclavo de 8 bits con sus respectivas señales de control que se encuentran compartiendo entradas analógicas en el puerto de 3 bits E.

Todos estos puertos son bidireccionales y existe en memoria una localidad específica para su configuración como entrada o salida, dicha configuración puede realizarse por palabra entera o bit a bit. El valor de entrada o salida del puerto (dependiendo de la manera en que esté configurado) se encuentra mapeada en una localidad de la memoria de datos.

La memoria de datos de 368 palabras de 8 bits se encuentra dividida en 4 partes denominadas "bancos" a las cuales se puede acceder de manera directa con la dirección real o con la relativa al banco con el que se esté trabajando. El banco elegido se configura mediante dos bits del registro de estado.

El microcontrolador se polariza con 5 [V] y para su funcionamiento también requiere de un oscilador, un circuito de reset, un par de resistencias y capacitores. Sus salidas son compatibles con dispositivos digitales que trabajen con 5 [V]. En la mayoría de los casos estos tipos de circuitos se trabajan con una tarjeta de control ya previamente diseñada que posee todos estos elementos.

En este caso el Microcontrolador está montado en una tarjeta fenólica diseñada por el Departamento de Ingeniería Mecatrónica de la Facultad de Ingeniería. Esta tarjeta se encarga de la buena distribución de los puertos, la protección y alimentación del microcontrolador, así como de la comunicación RS-232 de cualquier elemento que quiera establecer contacto con el microcontrolador. (Fig. 1.25)

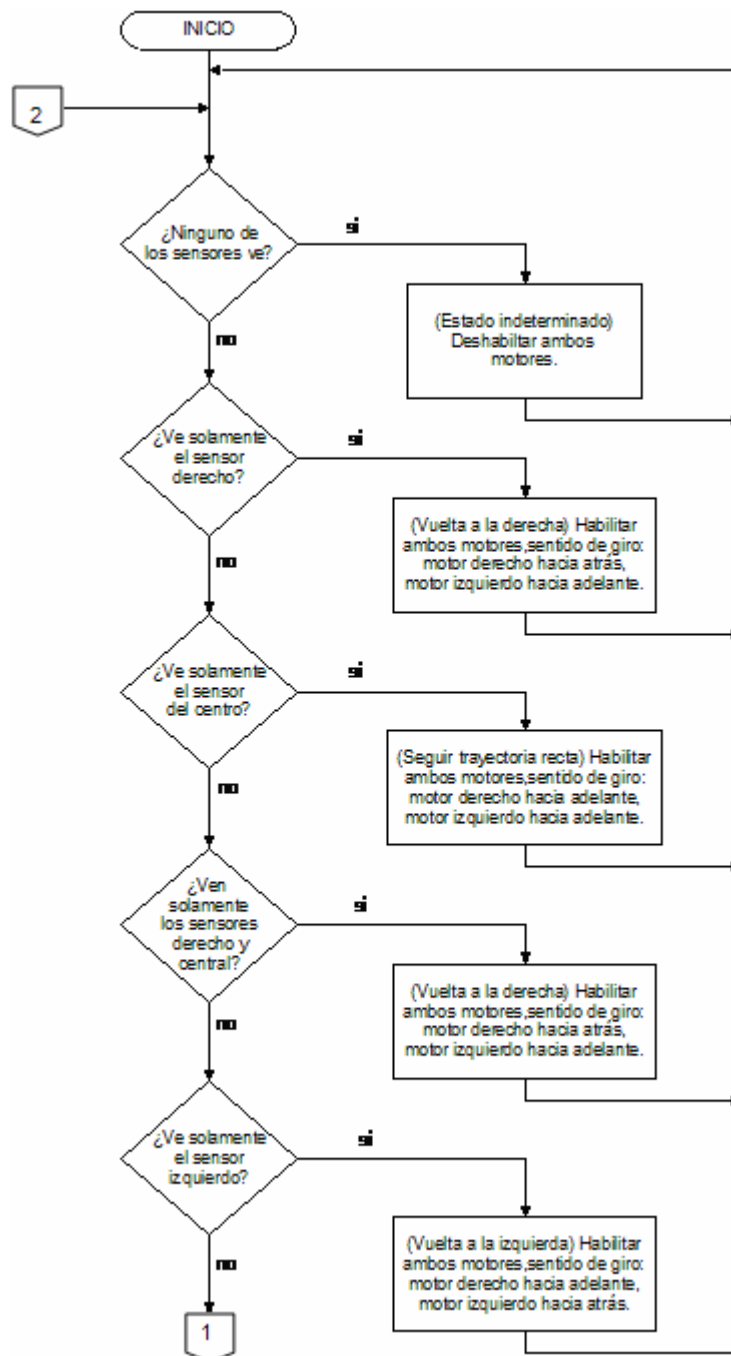


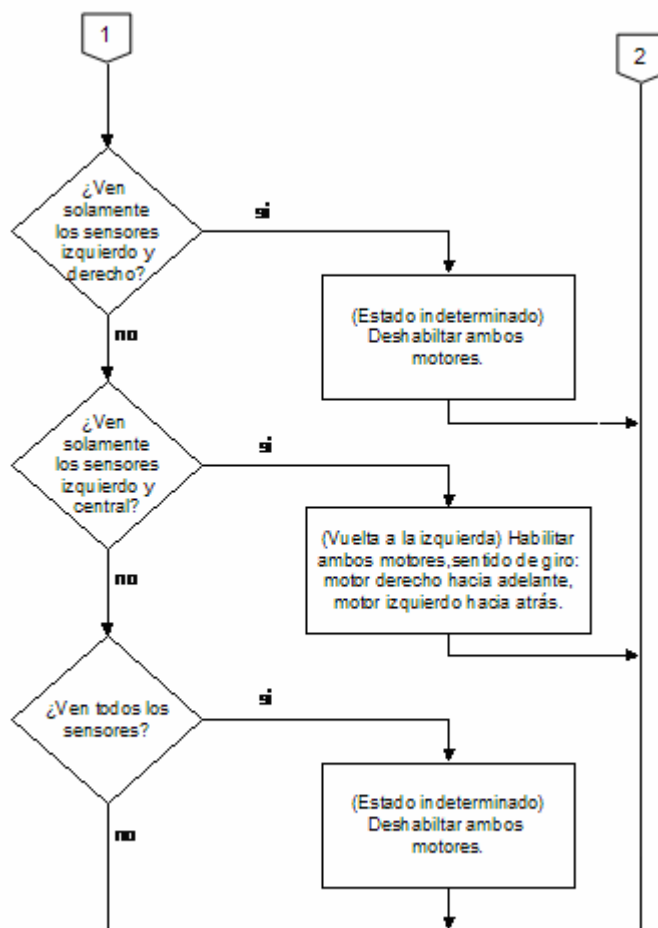
Fig. 1.25, Circuitería en tarjeta fenólica.

Este microprocesador funciona con un circuito oscilador de alta velocidad (HS), que asimismo podría ser utilizado con una red RC, un oscilador de baja frecuencia o uno de mediana frecuencia. Por otra parte, la tarjeta de control utilizada cuenta con dos cristales uno de 20 [MHz] y otro de 4[KHz] para establecer las frecuencias de trabajo. Además el circuito proporciona líneas de voltaje de 5 [V] y tierra para cualquier aplicación del usuario. Estos voltajes son los que se utilizan como alimentación de los sensores y demás circuitos que han sido mencionados en el subtema anterior.

Los sensores de percepción que se incorporaron al AGV, brindan el estado en que se encuentra el montacargas en cualquier instante del movimiento, gracias a esta información se puede controlar la trayectoria que se esté siguiendo. Para lograr esta meta, se corregirá el movimiento mediante la variación de la velocidad de manera independiente a la salida de cada motor de la tracción.

El diagrama de flujo siguiente muestra la corrección para los diferentes niveles de sensado que presenta el robot:





El robot continúa en un ciclo infinito de lectura de sensores hasta que ocurre un cambio significativo en alguno de ellos, con esta información el microcontrolador habilita o inhabilita los motores y cambia la polaridad de alimentación de los motores, controlando así su sentido de giro.

La ubicación de los tres sensores de percepción, los motores de desplazamiento del motor y los *enables* de los motores se encuentran en la siguiente forma:

#### SENSORES DE ENTRADA

|          |                  |
|----------|------------------|
| Puerto B |                  |
| Bit 2    | Sensor izquierdo |
| Puerto B |                  |
| Bit 1    | Sensor central   |
| Puerto B |                  |
| Bit 0    | Sensor derecho   |

**PUERTOS DE SALIDA**

|                    |                                         |
|--------------------|-----------------------------------------|
| Bit 2,<br>puerto C | Habilitador del motor izquierdo         |
| Bit 2,<br>puerto D | Sentido de giro para el motor izquierdo |
| Bit 1,<br>puerto C | Habilitador del motor derecho           |
| Bit 0,<br>puerto D | Sentido de giro para el motor derecho   |

El diagrama lógico de los estados de las entradas y salidas del microcontrolador se presenta a continuación:

| Entradas                  |                |                | Salidas del microcontrolador (PIC16F877) |                           |                                     |                                   | Acción                |
|---------------------------|----------------|----------------|------------------------------------------|---------------------------|-------------------------------------|-----------------------------------|-----------------------|
| Respuesta de los sensores |                |                |                                          |                           |                                     |                                   |                       |
| Puerto B                  |                |                | Puerto C                                 |                           | Puerto D                            |                                   |                       |
| Bit 2                     | Bit 1          | Bit 0          | Bit 2                                    | Bit 1                     | Bit 2                               | Bit 0                             |                       |
| Sensor izquierdo          | Sensor central | Sensor derecho | Habilitador motor izquierdo              | Habilitador motor derecho | Sentido de giro del motor izquierdo | Sentido de giro del motor derecho |                       |
| 0                         | 0              | 0              | 0                                        | 0                         | 0                                   | 0                                 | Estado indeterminado  |
| 0                         | 0              | 1              | 1                                        | 1                         | 1                                   | 0                                 | Vuelta a la derecha   |
| 0                         | 1              | 0              | 1                                        | 1                         | 1                                   | 1                                 | Trayectoria recta     |
| 0                         | 1              | 1              | 1                                        | 1                         | 1                                   | 0                                 | Vuelta a la derecha   |
| 1                         | 0              | 0              | 1                                        | 1                         | 0                                   | 1                                 | Vuelta a la izquierda |
| 1                         | 0              | 1              | 0                                        | 0                         | 0                                   | 0                                 | Estado indeterminado  |
| 1                         | 1              | 0              | 1                                        | 1                         | 0                                   | 1                                 | Vuelta a la izquierda |
| 1                         | 1              | 1              | 0                                        | 0                         | 0                                   | 0                                 | Estado indeterminado  |

A continuación, comenzará el desarrollo de la programación del microcontrolador PIC16F877, el cual será programado en lenguaje ensamblador.



La programación y algunas instrucciones se ubicarán en el apéndice 2. Dentro de esta sección se mencionarán únicamente las partes significativas de código y de las instrucciones necesarias.

Las variables significativas dentro del microcontrolador se declaran con la función *#define*, esto brindará claridad al hacer referencia a las entradas y salidas del sistema. Esta función se ocupará por facilidad y para poder interpretar mejor el código. A continuación muestra el código donde se definen las variables significativas:

```
#DEFINE SENSORD    PORTB,0    ;SENSOR DERECHA
#DEFINE SENSORC    PORTB,1    ;SENSOR CENTRAL
#DEFINE SENSORI    PORTB,2    ;SENSOR IZQUIERDA
#DEFINE MOTORD     PORTD,0    ;SENTIDO MOTOR DERECHA
#DEFINE MOTORI     PORTD,2    ;SENTIDO MOTOR IZQUIERDA
#DEFINE ENABLED    PORTC,1    ;ENABLE MOTOR DERECHA
#DEFINE ENABLEI    PORTC,2    ;ENABLE MOTOR IZQUIERDA
```

Los puertos deben de ser declarados del tipo en que serán usados (entradas o salidas de datos). Estos se declaran en el banco 1 del microcontrolador, para cambiar de banco debe de cambiar de estado al bit RPO del STATUS:

```
BSF        STATUS,RP0    ;CAMBIA A BANCO1 PARA CONFIGURAR

CLRFB      TRISD          ;CONFIGURA PORTD COMO SALIDA
BCF        PORTC,1        ;CONFIGURA PC BIT 1 COMO SALIDA, ALLI SE ENCUENTRA EL
                        ;ENABLE DEL MOTOR DERECHO
BCF        PORTC,2        ;CONFIGURA PC BIT 2 COMO SALIDA ALLI SE ENCUENTRA EL
                        ;ENABLE DEL MOTOR IZQUIERDO
MOVLW     b'11111111'    ;MUEVE VALOR DE 256 BINARIO
MOVWF     TRISB          ;ASIGNA EL VALOR PARA QUE PUERTO B SEA ENTRADA

BCF        STATUS,RP0    ;REGRESA AL BANCO 0
```

Como se puede apreciar se ha asignado el valor de 0 a la función TRISD esto convierte los 8 bits del puerto D en salidas; caso contrario de TRISB que se le asignó el valor binario 11111111 con lo cual se convierten los 8 bits del puerto B en entradas de datos. La configuración del puerto C es diferente. En este caso se apagan dos bits el 1 y 2 del puerto C con lo cual se convierten en salidas.

Para poder leer la información que presentan los sensores se debe tener en cuenta que todo el puerto B ha sido declarado como entrada de datos pero para este caso sólo es de relevancia la información que proporcionan los bits 1, 2 y 3 de dicho puerto. La información adicional que se presente en los bits restantes se tiene que eliminar, para lograr esto se debe aplicar una "máscara" que se encargue de discriminar los bits en uso.

```

MOVLW B'00000111'    ; SOLO SON DE INTERES LOS BITS 0, 1 Y 2
MOVWF MASCARA         ; MUEVE EL VALOR A LA MASCARA

MOVF  PORTB,W         ; MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
                      ; COMPARARLO
ANDWF  MASCARA,W       ; APLICA LA MASCARA Y SOLO RECIBIR LOS VALORES DEL PB0, PB1
                      ; Y PB2

MOVWF  TRASH          ; MUEVE EL VALOR AL LUGAR DE COMPARACION,

```

MASCARA es una localidad de memoria RAM que se ha creado para poder hacer la discriminación y recibir sólo los valores de bits en operación. Se aplica la operación de discriminación (AND) entre el puerto B y la MASCARA y el resultado de la discriminación se almacena en la variable TRASH.

TRASH es otra localidad de memoria que como su nombre lo indica, sólo se emplea para hacer operaciones momentáneas y después desechar el resultado cuando ya no tenga relevancia. TRASH en este momento tiene almacenado el resultado de la discriminación realizada, es decir, la información que le envían de los sensores.

Se debe trabajar con TRASH para saber de qué manera responderá el microcontrolador. El diagrama de flujo muestra la comparación que se debe de realizar para saber si el robot se encuentra en un estado u otro del programa, por consiguiente se compara TRASH y los estados probables del sistema:

```

MOVWF  TRASH          ; MUEVE EL VALOR AL LUGAR DE COMPARACION
MOVLW  B'00000001'    ; REvisa si el robot esta viendo solo el sensor
                      ; DE LA DERECHA
XORWF  TRASH,F         ; COMPARA EL VALOR
BTFSC  STATUS,Z        ; PREGUNTA SI ESTA PRENDIDO EL SENSOR
GOTO   VUELTADEDER     ; SI ESTA VIENDO LA DERECHA, DA VUELTA A LA DERECHA
                      ; NO, CONTINUA CON EL PROGRAMA

```

En el código anterior se compara el valor de TRASH contra el valor 00000001 que significa que el robot sólo está viendo el sensor de la derecha, la operación se lleva acabo con la operación XOR que arroja un cero si los valores de TRASH y el puerto son iguales. Posteriormente se pregunta al bit Z del STATUS el resultado de la última operación, si fue cero y si son iguales se va a vuelta derecha, en caso contrario da un salto de línea.

En dado caso que se dé un salto de línea el programa seguirá comparando hasta que halle la opción correspondiente al caso actual:

|          |                   |                                                                 |
|----------|-------------------|-----------------------------------------------------------------|
| PROGRAMA | MOVLW B'00000111' | <i>;MUEVE ESTE VALOR DE DISCRIMINACION, YA QUE ESOS SON</i>     |
|          |                   | <i>; LOS UNICOS VALORES DE INTERES</i>                          |
|          | MOVWF MASCARA     | <i>;MUEVE EL VALOR A LA MASCARA</i>                             |
|          | MOVF PORTB,W      | <i>;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA</i>            |
|          |                   | <i>COMPARARLO</i>                                               |
|          | ANDWF MASCARA,W   | <i>;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES</i>        |
|          |                   | <i>DE LOS SENSORES</i>                                          |
|          | MOVWF TRASH       | <i>;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,</i>           |
|          | MOVLW B'0000000'  | <i>;MUEVE EL VALOR PARA SABER SI NO VE NINGUN SENSOR</i>        |
|          | XORWF TRASH,F     | <i>;COMPARO EL VALOR</i>                                        |
|          | BTFSK STATUS,Z    | <i>;NO ESTA VIENDO LA LINEA NINGUN SENSOR???</i>                |
|          | GOTO INDETER      | <i>,SI, VA A LA SUBROUTINA INDETERMINADO</i>                    |
|          | MOVF PORTB,W      | <i>;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA</i>            |
|          |                   | <i>COMPARARLO</i>                                               |
|          | ANDWF MASCARA,W   | <i>;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES</i>        |
|          |                   | <i>DE LOS SENSORES</i>                                          |
|          | MOVWF TRASH       | <i>;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,</i>           |
|          | MOVLW B'0000001'  | <i>;MUEVE EL VALOR PARA SABER SI VE EL SENSOR DE LA DERECHA</i> |
|          | XORWF TRASH,F     | <i>;COMPARA EL VALOR</i>                                        |
|          | BTFSK STATUS,Z    | <i>;ESTA VIENDO EL SENSOR DE LA DERECHA???</i>                  |
|          | GOTO VUELTADE     | <i>,SI, VE A LA SUBROUTINA VUELTA DERECHA</i>                   |
|          | MOVF PORTB,W      | <i>;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA</i>            |
|          |                   | <i>COMPARARLO</i>                                               |
|          | ANDWF MASCARA,W   | <i>;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES</i>        |
|          |                   | <i>DE LOS SENSORES</i>                                          |
|          | MOVWF TRASH       | <i>;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,</i>           |
|          | MOVLW B'0000010'  | <i>;MUEVE EL VALOR PARA SABER SI ESTA VIENDO EL SENSOR DEL</i>  |
|          |                   | <i>CENTRO</i>                                                   |
|          | XORWF TRASH,F     | <i>;COMPARA EL VALOR</i>                                        |
|          | BTFSK STATUS,Z    | <i>;ESTA VIENDO EL SENSOR DEL CENTRO???</i>                     |
|          | GOTO RECTA        | <i>,SI, VE A LA SUBROUTINA RECTA</i>                            |
|          | MOVF PORTB,W      | <i>;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA</i>            |
|          |                   | <i>COMPARARLO</i>                                               |
|          | ANDWF MASCARA,W   | <i>;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES</i>        |
|          |                   | <i>DE LOS SENSORES</i>                                          |
|          | MOVWF TRASH       | <i>;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,</i>           |
|          | MOVLW B'0000011'  | <i>;MUEVE EL VALOR PARA SABER SI VE EL SENSOR DEL CENTRO Y</i>  |
|          |                   | <i>EL DE LA DERECHA</i>                                         |
|          | XORWF TRASH,F     | <i>;COMPARA EL VALOR</i>                                        |

```

BTFSC STATUS,Z      ;ESTA VIENDO EL SENSOR DEL CENTRO Y LA DERECHA???
GOTO  VUELTAIDER     ,SI, VA A LA SUBROUTINA VUELTA DERECHA

MOVF  PORTB,W        ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
                     ;COMPARARLO
ANDWF MASCARA,W      ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES
                     ;DE LOS SENSORES
MOVWF TRASH          ;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,
MOVLW B'0000100'     ;MUEVE EL VALOR PARA SABER SI VE EL SENSOR DE LA IZQUIERDA
XORWF TRASH,F        ;COMPARA EL VALOR
BTFSC STATUS,Z      ;ESTA VIENDO EL SENSOR DE LA IZQUIERDA???
GOTO  VUELTAIZQ      ,SI, VA A LA SUBROUTINA VUELTA IZQUIERDA

MOVF  PORTB,W        ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
                     ;COMPARARLO
ANDWF MASCARA,W      ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES
                     ;DE LOS SENSORES
MOVWF TRASH          ;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,
MOVLW B'0000101'     ;MUEVE EL VALOR PARA SABER SI ESTA VIENDO EL SENSOR DE LA
                     ;IZQUIERDA Y EL DE LA DERECHA
XORWF TRASH,F        ;COMPARO EL VALOR
BTFSC STATUS,Z      ;ESTA VIENDO EL SENSOR DE LA IZQUIERDA Y EL DE LA
                     ;DERECHA???
GOTO  INDETER        ,SI, VA A LA SUBROUTINA INDETERMINADO

MOVF  PORTB,W        ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
                     ;COMPARARLO
ANDWF MASCARA,W      ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES
                     ;DE LOS SENSORES
MOVWF TRASH          ;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,
MOVLW B'0000110'     ;MUEVE EL VALOR PARA SABER SI VE EL SENSOR DE LA DERECHA Y
                     ;EL DEL CENTRO
XORWF TRASH,F        ;COMPARA EL VALOR
BTFSC STATUS,Z      ;ESTA VIENDO EL SENSOR DE LA IZQUIERDA Y EL DEL CENTRO???
GOTO  VUELTAIZQ      ,SI, VA A LA SUBROUTINA VUELTA IZQUIERDA

MOVF  PORTB,W        ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
                     ;COMPARARLO
ANDWF MASCARA,W      ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES
                     ;DE LOS SENSORES
MOVWF TRASH          ;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,
MOVLW B'0000111'     ;MUEVE EL VALOR PARA SABER SI VEN TODOS LOS SENSORES
XORWF TRASH,F        ;COMPARO EL VALOR
BTFSC STATUS,Z      ;ESTAN VIENDO TODO LOS SENSORES???
GOTO  INDETER        ,SI, VA A LA SUBROUTINA INDETERMINDO

GOTO  PROGRAMA

```

En el código anterior se realizan todas y cada una de las comparaciones para encontrar a la subrutina a la que debe de dirigirse el seguidor de línea en cualquier estado del sensado. El código se comporta de la misma forma que el diagrama de flujo que se presentó anteriormente.

En este punto cabe destacar que cada línea de programa tarda en ser ejecutada cuatro veces el periodo del oscilador principal, la frecuencia del oscilador principal que se utiliza es de 20 [MHz] por consiguiente el tiempo de procesamiento es de 200 [ns]. Así, en ejecutarse todo este fragmento de código se tarda 11.8 [μs] y como la velocidad de los motores es muy baja, garantizamos que prácticamente el microcontrolador siempre está trabajando con la información que estén enviando los sensores durante su desplazamiento, y por consiguiente no habrá retraso en la señal en proceso.

Las subrutinas tendrán contedida la información de las salidas del sistema en cada momento, de acuerdo con el diagrama lógico:

|            |      |          |                                     |
|------------|------|----------|-------------------------------------|
| VUELTA DER | BCF  | ENABLED  | <i>;APAGA EL MOTOR DERECHO</i>      |
|            | BSF  | ENABLEI  | <i>;ENCIENDE EL MOTOR IZQUIERDO</i> |
|            | BSF  | MOTORI   | <i>;EL MOTORI VA HACIA ADELANTE</i> |
|            | GOTO | PROGRAMA | <i>;REGRESA AL PROGRAMA</i>         |
| RECTA      | BSF  | ENABLED  | <i>;ENCIENDE EL MOTOR DERECHO</i>   |
|            | BSF  | ENABLEI  | <i>;ENCIENDE EL MOTOR IZQUIERDO</i> |
|            | BSF  | MOTORD   | <i>;EL MOTORD VA HACIA ADELANTE</i> |
|            | BSF  | MOTORI   | <i>;EL MOTORI VA HACIA ADELANTE</i> |
| VUELTA IZQ | GOTO | PROGRAMA | <i>;REGRESA AL PROGRAMA</i>         |
|            | BSF  | ENABLED  | <i>;ENCIENDE EL MOTOR DERECHO</i>   |
|            | BCF  | ENABLEI  | <i>;APAGA EL MOTOR IZQUIERDO</i>    |
|            | BSF  | MOTORD   | <i>;EL MOTORD VA HACIA ADELANTE</i> |
| INDETER    | GOTO | PROGRAMA | <i>;REGRESA AL PROGRAMA</i>         |
|            | BCF  | ENABLED  | <i>;APAGA EL MOTOR DERECHO</i>      |
|            | BCF  | ENABLEI  | <i>;APAGA EL MOTOR IZQUIERDO</i>    |
|            | GOTO | PROGRAMA | <i>;REGRESA AL PROGRAMA</i>         |

De esta forma se controlan cada uno de los movimientos del AGV dependiendo de la señal de los estados que envíen en los sensores.

A continuación se tratará la forma en la que se controlará el órgano terminal. Como ya se ha mencionado, se necesita control mediante ancho de pulso a una frecuencia de 20 [ms] y con un periodo de trabajo establecido de 1.5333 [ms] para que se mueva la pinza 40°.

El microcontrolador cuenta con dos PWM ya configurados que dependen del timer 2 del microcontrolador. Sin embargo, el tiempo de retardo del timer 2 del microcontrolador depende directamente de la frecuencia del cristal de 20[MHz]; entonces, el tiempo de retardo máximo que proporciona el timer 2 es de 13.1072 [ms], que es mucho menor que el ancho de pulso necesario, 20 [ms].

Por lo tanto se tiene que implantar un PWM regulador del voltaje de salida a través del puerto D pin 6 que trabaje con el periodo de tiempo de trabajo deseado. Se eligió trabajar con el timer 0 de microcontrolador y trabajar mediante interrupciones.

Las interrupciones son acciones que atienden operaciones que se solicitan en cualquier momento del programa y al terminar de ejecutarse continúan realizando las acciones que estaban realizando antes del llamado. Existen 14 diferentes tipos de interrupciones para el PIC16F877, en este caso se ocupará la interrupción por cuenta máxima en el timer 0.

La interrupción se llevará a cabo cada 0.1 [ms] con el objetivo de hacer más suave el levantamiento de la plataforma, la forma en la que la interrupción trabaja se muestra en el siguiente esquema (Fig. 1.26):

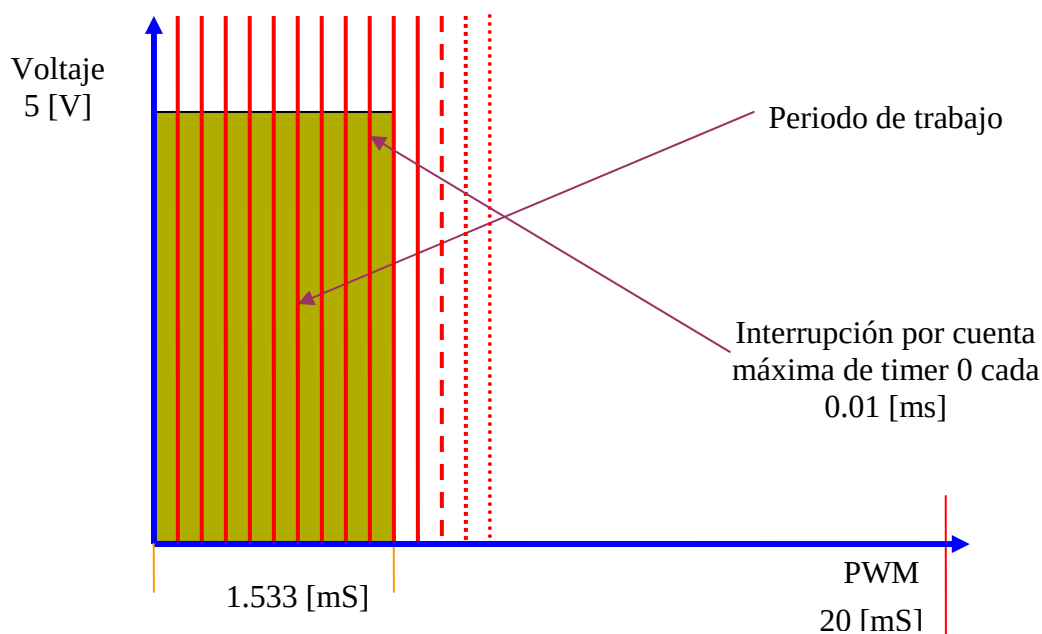


Fig. 1.26, Interrupción

En el esquema anterior se resaltan 2 aspectos, el tiempo del periodo de trabajo 1.533 [ms] y el tiempo del ancho de pulso 20 [ms], además es pertinente recordar que el elevador se encuentra en su posición inicial, sin levantar, entonces, debe de tener un ángulo de 0° en la flecha del servomotor, lo que indica que se debe mantener un periodo de trabajo de 1 [ms]. Los valores significativos se definen como constantes de la siguiente forma:

|           |     |      |                                                                                                |
|-----------|-----|------|------------------------------------------------------------------------------------------------|
| MAX       | EQU | .200 | <i>;VALOR PARA EL VALOR DEL CICLO DE TRABAJO DE LOS SERVOS (20 [ms]) EN ESCALA DE 0.1 [ms]</i> |
| MIN1      | EQU | .100 | <i>;VALOR MINIMO PARA EL PERIODO DE TRABAJO DE LOS SERVOS (0.97[ms]) EN ESCALA DE 0.01[ms]</i> |
| REQUERIDO | EQU | .150 | <i>;VALOR QUE QUEREMOS PARA EL MOVIMIENTO DE 40°, 1.45[ms] EN ESCALA DE 0.01[ms]</i>           |

La interrupción debe de estar programada para que se lleve acabo cada 0.01 [ms]. La programación de las interrupciones así como la habilitación de las banderas de interrupción podrán ser consultadas en el apéndice.

A cada entrada a la interrupción se incrementa un contador llamado CUENTA2. Este contador debe de ser comparado contra el periodo de trabajo que se quiere,

puede ser que el periodo de trabajo que se desee sea para 0° o para 40°, el periodo de trabajo que se necesita está almacenado en el registro PWM1:

|       |       |           |                                                                                            |
|-------|-------|-----------|--------------------------------------------------------------------------------------------|
| INTER | INCF  | CUENTA2   | <i>;INCREMENTA LA CUENTA2</i>                                                              |
|       | MOVF  | CUENTA2,W | <i>;CARGA EL VALOR DE CUENTA EN EL ACUMULADOR</i>                                          |
|       | XORWF | PWM1,W    | <i>;COMPARA EL VALOR REQUERIDO CON LA CUENTA HASTA EL MOMENTO</i>                          |
|       | BTFSS | STATUS,Z  | <i>;EL VALOR DEL PWM REQUERIDO ES IGUAL AL VALOR DEL LA CUENTA QUE LLEVA???</i>            |
|       | GOTO  | CONTINUA  | <i>;NO, NO HA PASADO COMPLETO EL PERIODO DE TRABAJO, CONTINUA CHECANDO LA INTERRUPCION</i> |
|       | BCF   | SERVO1    | <i>;SI, YA TENEMOS EL PWM REQUERIDO, APAGA EL PUERTO</i>                                   |
|       | GOTO  | VOLVER    | <i>;REGRESA A LA RUTINA. DEL PROGRAMA</i>                                                  |

Se realiza la comparación de la cuenta contra el valor requerido del periodo de trabajo que está almacenado en el registro PWM1. Si el valor de PWM1 y de la cuenta de tiempo es el mismo significa que ya ha pasado todo el periodo de trabajo por consiguiente se apaga el bit del puerto D, de lo contrario se continua revisando la interrupción. Para completar un periodo de trabajo para 40°, el programa deberá entrar 150 veces a interrupción con el puerto D encendido y 100 veces para 0°. De ahí salen los valores de los registros MIN1 y REQUERIDO.

Posteriormente se revisará si ya pasó todo el PWM. Existe otro registro de cuenta llamado TIME, este registro revisa si la interrupción ha pasado 10 veces, esta cuenta se hace para cambiar de escala, si TIME se cumple quiere decir que ha ocurrido una interrupción de 0.1 [ms] y no una de 0.01 [ms]. Este cambio se hace por practicidad debido a que si se manejara un contador de tiempo normal para intervalos de 0.01 [ms] hasta 20 [ms] se necesitaría un registro que fuera capaz de contar hasta 2000 y los registros normales sólo son de 8 bits y cuentan hasta 256.

Así con valores de 0.1[ms] sólo es necesario contar hasta 200 para completar el PWM, de ahí sale el valor de la constante MAX:

|          |       |        |                                                                                          |
|----------|-------|--------|------------------------------------------------------------------------------------------|
| CONTINUA | INCF  | TIME   | <i>;INCREMENTA TIME QUE LLEVA LA CUENTA DEL TIEMPO QUE HA PASADO POR LA INTERRUPCION</i> |
|          | MOVLW | .10    | <i>;EL VALOR PARA SABER SI YA PASO 10 VECES</i>                                          |
|          | XORWF | TIME,W | <i>;YA PASO 10 VECES?</i>                                                                |



|       |          |                                                                                               |
|-------|----------|-----------------------------------------------------------------------------------------------|
| BTFSS | STATUS,Z | <i>;YA PASÓ 1 [ms] DESDE LA ULTIMA VEZ QUE ENTRASTE???</i>                                    |
| GOTO  | VOLVER   | <i>;NO, VA A LA RUTINA DEL PROGRAMA</i>                                                       |
| CLRF  | TIME     | <i>;SI, ENTONCES CONTINUA Y LIMPIA TIME</i>                                                   |
| INCF  | CUENTA,F | <i>;CUENTA = CUENTA + 1, LO CUAL SIGNIFICA QUE HA PASADO 1 ms MAS DESDE LA ULTIMA ENTRADA</i> |
| MOVF  | CUENTA,W | <i>;ASIGNA EL VALOR DE LA CUENTA AL ACUMULADOR</i>                                            |
| XORLW | MAX      | <i>;COMPARA EL VALOR DE LA CUENTA CON 20 MS (LO QUE DURA EL PERIODO DE LOS SERVOMOTORES)</i>  |
| BTFSS | STATUS,Z | <i>;CUENTA LLEGO AL MAXIMO?, HAN PASADO 20 MS???</i>                                          |
| GOTO  | VOLVER   | <i>;NO, CONTINUA CHECANDO EL PROGRAMA</i>                                                     |
| BSF   | SERVO1   | <i>;SI, ES UN NUEVO CICLO, DEBE DE INICIAR CON FLANCO,</i>                                    |
|       |          | <i>;PRENDE EL SERVO</i>                                                                       |
| CLRF  | CUENTA   | <i>;LIMPIA LA CUENTA</i>                                                                      |
| CLRF  | CUENTA2  | <i>;LIMPIA LA CUENTA2</i>                                                                     |

La cuenta de tiempo para saber si ha pasado completo el PWM se almacena en el registro CUENTA2, este registro es comparado contra el valor MAX para saber si ya ha pasado el PWM completo. Si ya ha pasado todo el PWM se vuelve a encender SERVO1 (puerto D bit 6) y se inicializan las variables de cuenta.

En consecuencia se tratará la forma en la que adquiere el valor del periodo de trabajo el registro PWM1. Primero el valor inicial para el ángulo de la flecha de salida del servomotor debe de ser 0° por lo cual se asigna a PWM1 el valor del periodo de trabajo mínimo que se encuentra en MIN1 y además se inicializan las variables:

|       |         |                                                                                                                              |
|-------|---------|------------------------------------------------------------------------------------------------------------------------------|
| CLRF  | TIME    | <i>;LIMPIA EL TIME QUE SIRVE PARA EL SERVO</i>                                                                               |
| CLRF  | ESTADO  | <i>;ASIGNA EL ESTADO INICIAL DE LA PINZA, COMO ES CERO ESTA ABAJO</i>                                                        |
| CLRF  | CUENTA  | <i>;INICIALIZA REGISTRO DE CONTEO DE EVENTOS, INICIAMOS EN 0 LO CUAL SON 0 MS TRANSCURRIDOS</i>                              |
| CLRF  | CUENTA2 | <i>;LIMPIA UNA DE LAS CUENTAS DEL SERVO</i>                                                                                  |
| MOVLW | MIN1    | <i>;CARGA EL VALOR MINIMO EN EL ACUMULADOR, ESTE VALOR ES PARA QUE LA PINZA NO SE LEVANTE, EL ANCHO DE PULSO ES DE 1[MS]</i> |
| MOVWF | PWM1    | <i>;MUEVE EL VALOR PARA QUE EL PWM ARRANQUE EN LA POSICION DE 0°</i>                                                         |
| BSF   | SERVO1  | <i>;EL VALOR DEL PWM SIEMPRE INICIA EN FLANCO ALTO</i>                                                                       |

También se enciende el PWM1 porque se debe de iniciar el PWM en flanco alto.

En el caso del programa del ancho de pulso que se encuentra en el apéndice, el cambio de estado de la pinza ocurrirá cuando ninguno de los sensores esté viendo la línea; el estado de la pinza se almacena en un registro llamado ESTADO en su bit 0, cuando el bit 0 de estado esté encendido la pinza estará arriba, en el caso contrario estará retraída:

|       |             |                                                                   |
|-------|-------------|-------------------------------------------------------------------|
| MOVF  | PORTB,W     | ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA COMPARARLO          |
| ANDWF | MASCARA,W   | ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES |
| MOVWF | TRASH       | ;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,                    |
| MOVLW | B'00000000' | ;MUEVE EL VALOR PARA SABER SI NO VE NINGUN SENSOR                 |
| XORWF | TRASH,F     | ;COMPARA EL VALOR                                                 |
| BTFSC | STATUS,Z    | ;NO ESTA VIENDO LA LINEA NINGUN SENSOR???                         |
| GOTO  | LEVANTAR    | ;SI, VE A LA SUBROUTINA LEVANTAR                                  |

La subrutina LEVANTAR contiene la forma en la que se va a levantar la pinza. Esta contiene un retardo, este retardo modulado hace que se incremente el periodo de trabajo muy lentamente para que el cambio de estado en la pinza no sea muy brusco:

|          |        |            |                                                  |
|----------|--------|------------|--------------------------------------------------|
| LEVANTAR | BTFSC  | ESTADO,0   | ;ESTA EN EL ESTADO INICIAL, CON LA PINZA ABAJO?? |
|          | GOTO   | PROGRAMA   | ;NO, ENTONCES ESTA YA CON LA PINZA LEVANTADA,    |
|          |        |            | ;REVISA EL PROGRAMA                              |
| PWM      | INCF   | PWM1       | ;INCREMENTA EL PWM                               |
|          | MOVLW  | .255       | ;VALOR QUE SE LE VA A MOVER AL DELAY             |
|          | MOVWF  | VALOR1     | ;VALOR DEL DELAY                                 |
|          | MOVLW  | .4         | ;VALOR QUE SE LE VA A MOVER AL DELAY             |
|          | MOVWF  | VALOR2     | ;VALOR DEL DELAY                                 |
|          | MOVLW  | .31        | ;VALOR QUE SE LE VA A MOVER AL DELAY             |
|          | MOVWF  | VALOR3     | ;VALOR DEL DELAY                                 |
|          | GOTO   | DELAY      | ;SE VA A UN RETARDO                              |
| SIGO     | MOVLW  | REQUERIDO  | ;COMPARA A VER SI YA LLEGO AL MAXIMO INCREMENTO  |
|          | XORWF  | PWM1,TRASH | ;COMPARA EL VALOR DE ACULADOR CON PWM1           |
|          | BTFSS  | STATUS,Z   | ;YA ALCANZO EL VALOR REQUERIDO???                |
|          | GOTO   | PWM        | ;NO, VE Y SIGUE INCREMENTANDO                    |
|          | BSF    | ESTADO,0   | ;YA ESTA EN EL ESTADO FINAL, CON LA PINZA        |
|          |        |            | ;LEVANTADA                                       |
|          | GOTO   | PROGRAMA   | ;AHORA REVISA SI CAMBIO DE ESTADO EL PROGRAMA    |
| ;RETARDO |        |            |                                                  |
| DELAY    | MOVF   | VALOR1,W   |                                                  |
|          | MOVWF  | TMP0       |                                                  |
| DEL0     | MOVF   | VALOR2,W   |                                                  |
|          | MOVWF  | TMP1       |                                                  |
| DEL1     | MOVF   | VALOR3,W   |                                                  |
|          | MOVWF  | TMP2       |                                                  |
| DEL2     | DECFSZ | TMP2,F     |                                                  |
|          | GOTO   | DEL2       |                                                  |
|          | DECFSZ | TMP1,F     |                                                  |
|          | GOTO   | DEL1       |                                                  |
|          | DECFSZ | TMP0,F     |                                                  |
|          | GOTO   | DEL0       |                                                  |
|          | GOTO   | SIGO       |                                                  |

El retardo que se utiliza para el levantamiento de la pinza es de tres variables el cual responde a la siguiente formula:

$$\text{Ciclos de retardo} = (3 \cdot \text{VALOR1} \cdot \text{VALOR2} \cdot \text{VALOR3} + 4 \cdot \text{VALOR1} \cdot \text{VALOR2} + 4 \cdot \text{VALOR1} + 5)$$

$$\text{Tiempo de retardo} = \text{ciclos de retardo} \cdot (200 \times 10^{-9})$$

$$\text{Tiempo de retardo} = (99965)(200 \times 10^{-9}) = 0.01999[\text{s}]$$

Así, en el programa se incrementa el valor del PWM1 (cuyo valor inicial es de 100) de 1 en 1. Cada cambio toma 0.01999[s], como el valor que se necesita alcanzar es de 150, se deben de realizar 50 retardos y 50 incrementos en PWM1. Por consiguiente, el cambio de estado de la pinza de la posición de 0° a 40° toma 1 segundo en completarse. Y se cambia el registro ESTADO para indicar que se encuentra en la posición de levantada la pinza.

En este capítulo se ha tratado el diseño e implementación de un AGV, así como el control del elevador mediante ancho de pulso. Estos temas se retomarán más adelante al hacer la integración de los sistemas.

NOTA: En el apéndice II se encuentra una explicación más técnica y detallada de los comandos, operaciones y procedimientos utilizados en esta sección para la programación del Microcontrolador PIC16F877

## AUTOMATIZACIÓN DEL PROCESO DE DISTRIBUCIÓN

### 2.1 Diseño y configuración del sistema

En este capítulo se desarrollará el diseño correspondiente a la configuración del sistema de automatización. El sistema involucra la implementación de 3 bandas de hule como medio de transporte en el plano horizontal. Las bandas deben estar tensadas por un par de rodillos colocados en los extremos de ellas y uno de los rodillos debe estar acoplado a un motor con la suficiente potencia para mover las bandas gracias a la fuerza de fricción que exista entre él y las bandas de hule (Fig. 2.1, Fig. 2.2). A su vez, las bandas deben tener una inclinación de cero grados con respecto de la pista. Cabe mencionar que los motores deberán ser colocados en lugares en los que no estorben la trayectoria del montacargas.

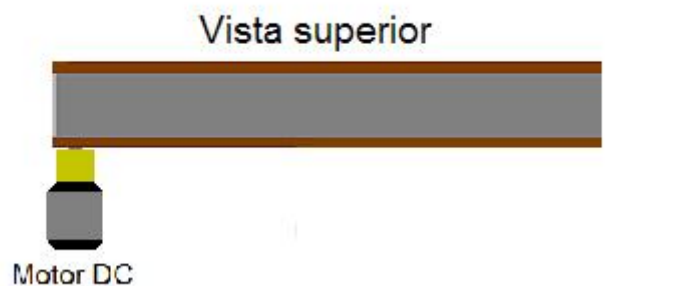


Fig. 2.1, Banda vista superior



Fig. 2.2, Banda vista frontal

Para los fines de este proyecto, el sistema deberá contar únicamente con un par de despachadores que contendrán dos diferentes productos. La cantidad de cada producto se surtirá por tiempo mediante la apertura y cierre de las

boquillas de los despachadores, éste trabajo será realizado por dos pistones neumáticos, cada uno colocado en la misma posición de cada despachador, de tal forma que el pistón esté por debajo de la salida de los despachadores (Fig. 2.3).

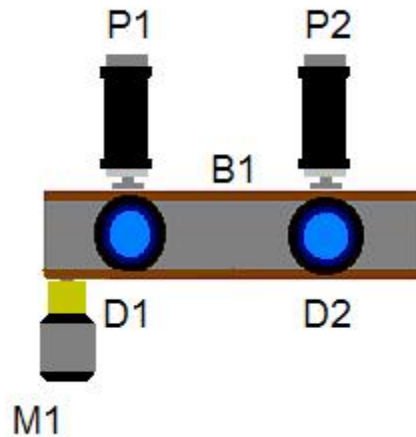


Fig. 2.3, Despachadores

Es necesario que las bandas estén acopladas entre sí, de tal manera que se puedan comunicar unas con otras para transportar los productos de una a otra sin desestabilizar los productos. El sistema debe tener la facultad de discriminar entre los dos productos que se puedan despachar, entonces, después del final de la primera banda, se colocará un pistón que se encargará de negociar hacia qué banda debe continuar cada producto según su tipo.

Finalmente, se debe contar con elementos de percepción que son fundamentales para la toma de decisiones a lo largo del proceso. Los sensores son ópticos y en todos los casos serán activados mediante la presencia o al paso de los productos que se encuentren dentro de su rango de visión, sin embargo existirán tres tipos de funciones dependiendo de cuál sensor sea el que dispare una señal, una de las funciones será para detener las bandas al momento de surtir el producto dentro de los contenedores, otra señal activará al discriminador dependiendo del tipo de producto y por último el tercer tipo de señal detendrá todo el proceso, ya que indicará que existe un paquete al final del proceso.

Los sensores se colocarán de la siguiente manera:

S1: frente al despachador 1

S2: frente al despachador 2

S3: frente al pistón 3

S4: a un lado de la rampa 1

S5: a un lado de la rampa 2

El diagrama final de distribución es el siguiente (Fig. 2.4):

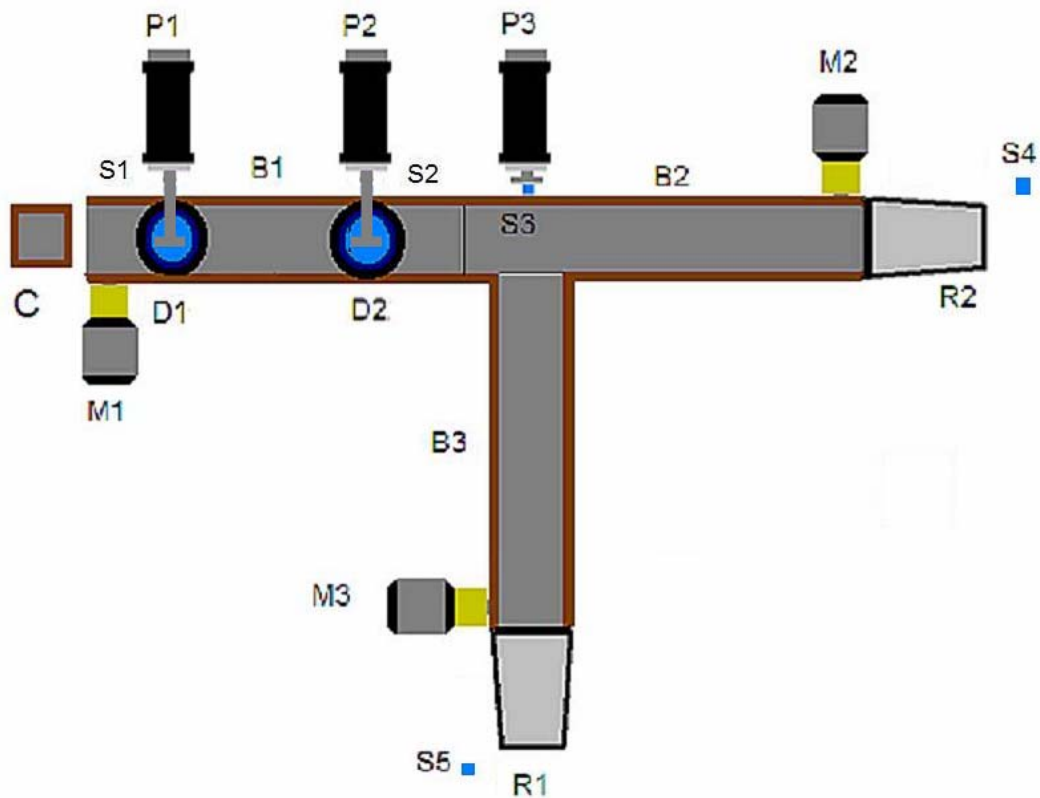


Fig. 2.4, Diagrama de distribución

Nomenclatura:

- B#: Banda
- C: Contenedor a despachar
- D#: Despachador
- P#: Pistón
- M#: Motor DC
- R#: Rampa
- S#: Sensor

## 2.2 Percepción y acción del proceso

Ahora que se ha definido la forma en que las bandas de distribución, los motores y los pistones estarán ubicados dentro del proceso, se mostrará la forma en la que está constituido el sensado del proceso y la forma en que se distribuyen y trabajan las señales de control.

Para el proceso de llenado se necesitó crear un tipo de percepción el cuál tuviera la capacidad identificar los paquetes que se envían por la banda y así, ésta pudiera ser llenada con el producto correspondiente. En este caso se utilizaron sensores de presencia de dos tipos distintos, son sensores ópticos reflexivos con salida de transistor, con números de parte QBR1113 y CNY70. El sensor QBR1113 se ha descrito anteriormente.

El sensor CNY70 (Fig. 2.5) posee una estructura compacta donde una fuente emisora de luz y el detector, que consiste de un fototransistor, están acomodados en la misma dirección para detectar la presencia de un objeto mediante el rayo de luz que se refleja del objeto.

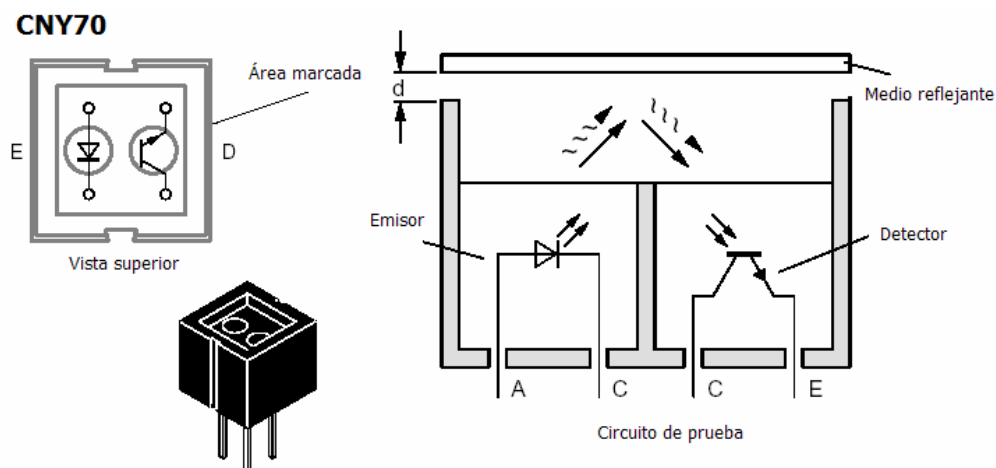


Fig. 2.5 Sensor con fototransistor CNY70

Es necesario conocer la ubicación de los contenedores a lo largo de la primer banda, por esta razón, se colocarán los sensores justamente en donde se encuentran los vertederos de producto.

De igual forma, se debe percibir la posición de la caja en la banda de distribución número 2 y así poder enviar la caja a la rampa 2 o dejarla continuar su camino para que llegue a la rampa número 1, ahí será recogida por el montacargas (Fig. 2.6):

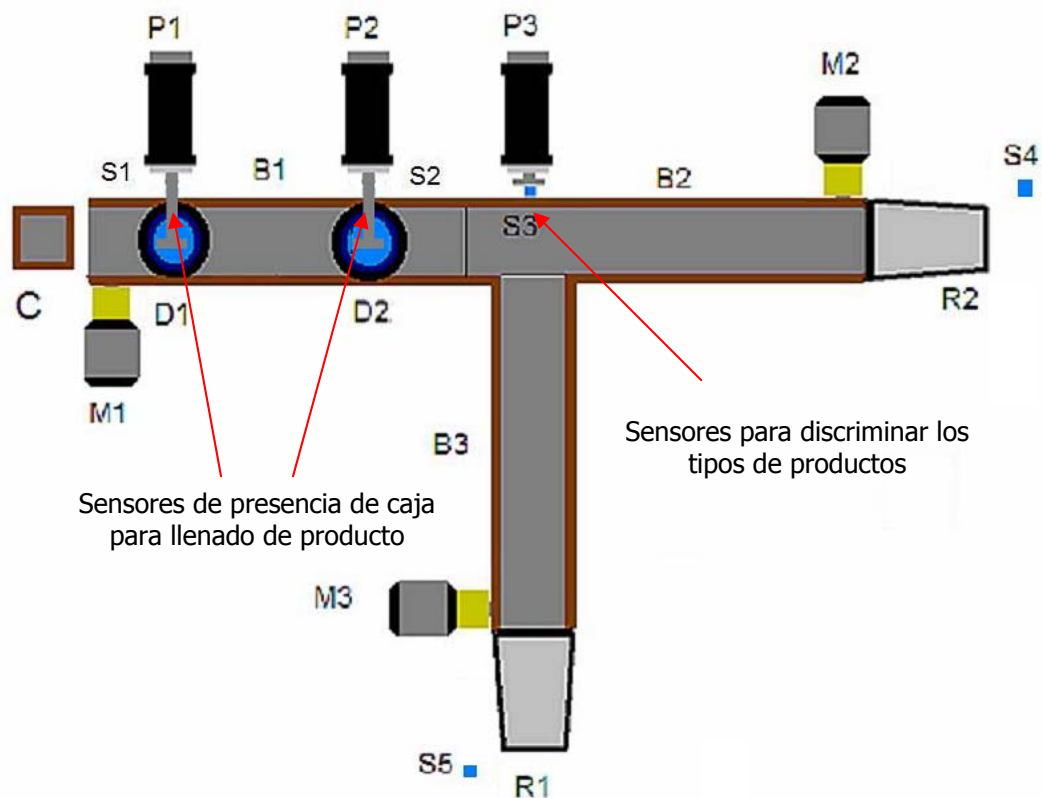
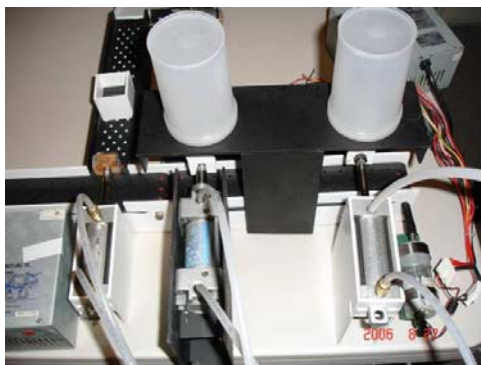


Fig. 2.6, Ubicación de sensores de presencia





Los paquetes también serán sensados al terminar su trayectoria, ya que al final de cada rampa se encontrará un sensor QRB1113<sup>1</sup>, del mismo tipo de los que fueron utilizados en el AGV, éste dispositivo tiene una mayor sensibilidad porque las cajas se encontrarán a una mayor distancia de la posición del sensor para que no afecte con el movimiento del seguidor de línea. Además, se debe tener la certeza de que existe un producto listo para ser recogido y evitar la expedición de un producto del mismo tipo lo cual provocaría problemas en el proceso. El sensor también tiene la función de percibir ausencia de producto (Fig. 2.7), ante lo cual el proceso tiene que expedir inmediatamente un producto del tipo faltante.

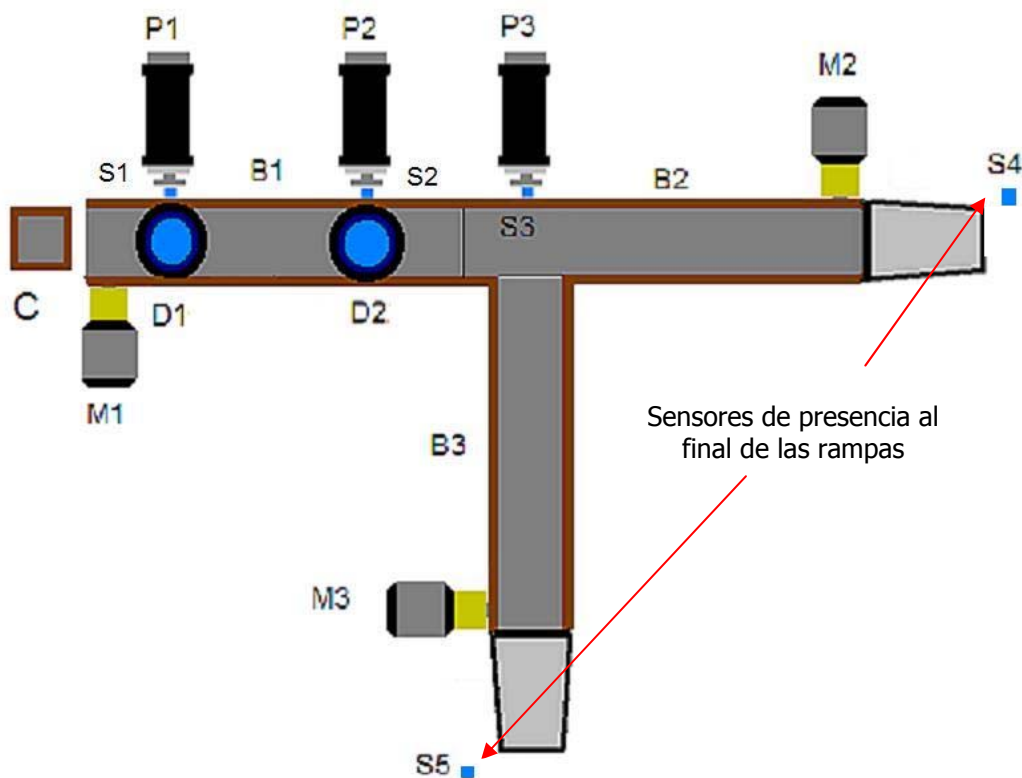


Fig. 2.7, Ubicación de los sensores de presencia en las tarimas

<sup>1</sup> QRB1113, Apéndice I

La señal de salida de los sensores es acondicionada para el manejo del PLC que se utilizará.

Debido a que la señal que envían los sensores es muy débil se debe amplificar. Así mismo se utiliza también una etapa de comparación, para garantizar que se está sensando según la etapa en la que se encuentre el proceso; finalmente, antes de enviar la señal al PLC de igual forma que se hizo en los sensores en el AGV, se acoplará un Schmitt-Trigger para restablecer los voltajes lógicos<sup>2</sup>.

Las salidas de los sensores se conectan a un circuito que contiene un arreglo de comparadores, potenciómetros (mini presets) y resistencias que ayudan a calibrar los sensores a las condiciones de trabajo, como la cantidad de luz del área de trabajo y proximidad del objeto, de igual forma son utilizados para las variaciones de voltaje que se obtienen a su salida, ya que es necesario que se entreguen valores digitales en 0 y 5 [V]. Además cada salida de comparador va conectado a una entrada del circuito Schmitt-trigger inversor para reestablecer el valor lógico de la señal y poder trabajar con valores lógicos válidos y correctos. En el siguiente diagrama se muestra el diagrama de conexiones de la tarjeta que contiene el circuito descrito. (Fig. 2.8)

---

<sup>2</sup> LM339A, Schmitt-Trigger SN74LS14, Apéndice I

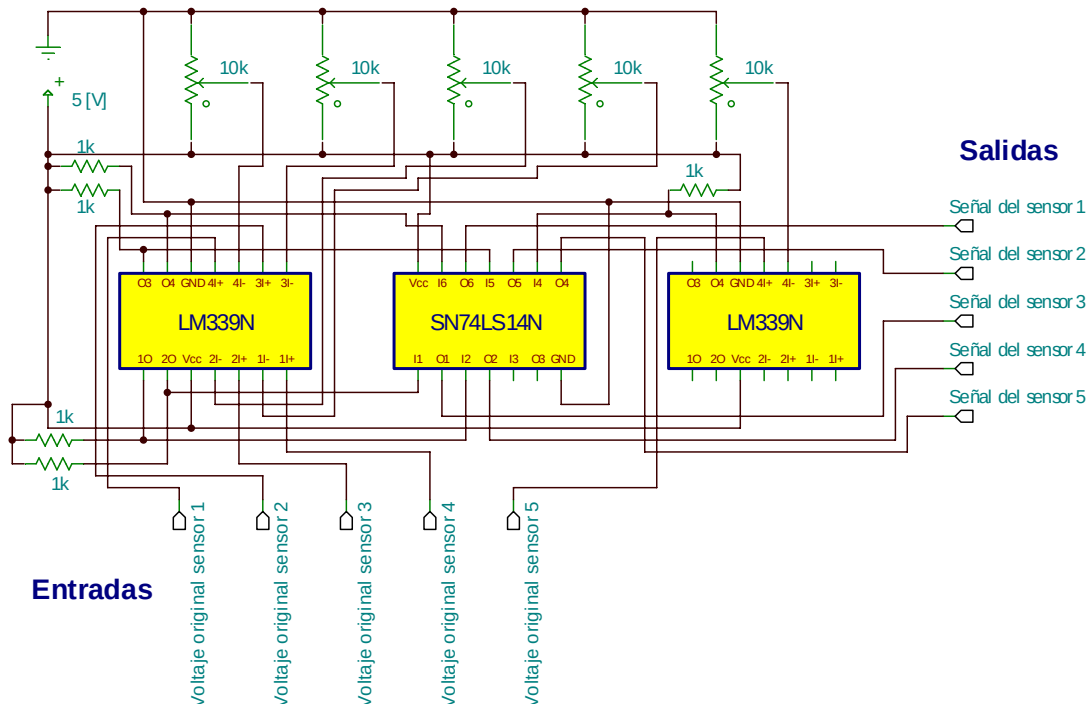


Fig. 2.8, Circuitería de comparación

Las salidas de la tarjeta descrita anteriormente son las señales que indican la presencia o ausencia de una caja en algún punto determinado del sistema. Dichas señales no poseen la corriente necesaria demandada para continuar, así que es necesario colocar un circuito adicional como etapa de potencia que consiste de un transistor PNP y una resistencia para cada salida.

El transistor cuyo número de parte es TIP32C es un transistor de silicio de tipo PNP cuya ventaja principal es que proporciona la corriente necesaria para activar el relevador que se conecta a él, con el fin de preparar la señal para interactuar con el PLC. En el siguiente diagrama se muestra dicho transistor (Fig 2.9).

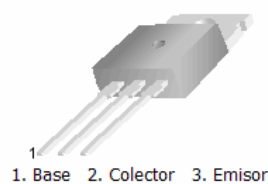


Fig. 2.9, Transistor de silicio PNP, TIP32

La forma en que se conecta el transistor a cada señal de cada sensor obtenida de la tarjeta de calibración explicada anteriormente y su salida se muestra a continuación. Como se puede observar la configuración actúa como un circuito inversor que a la salida nos dará 5 [V] cuando el sensor detecte un objeto y 0 [V] cuando no detecte nada. (Fig. 2.10)

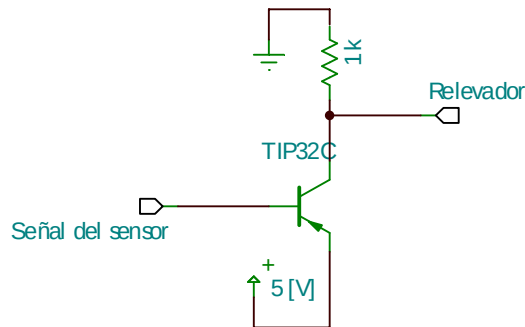


Fig. 2.10, Acoplamiento TIP32

Después de que la señal ha sido amplificada, ésta se manda a un relevador que cierra el circuito. El relevador que se utilizará tiene número de parte RAS – 0510 (Fig. 2.11). El relevador es un interruptor operado magnéticamente. Éste se activa o desactiva (dependiendo de la conexión) cuando el electroimán, que forma parte del relevador, es energizado. Esta operación causa que exista conexión o no, entre dos terminales del relevador. Esta conexión se logra con la atracción o repulsión de un pequeño brazo, llamado común, con el que se conectan o desconectan las terminales mencionadas.

### Relevador RAS - 0510

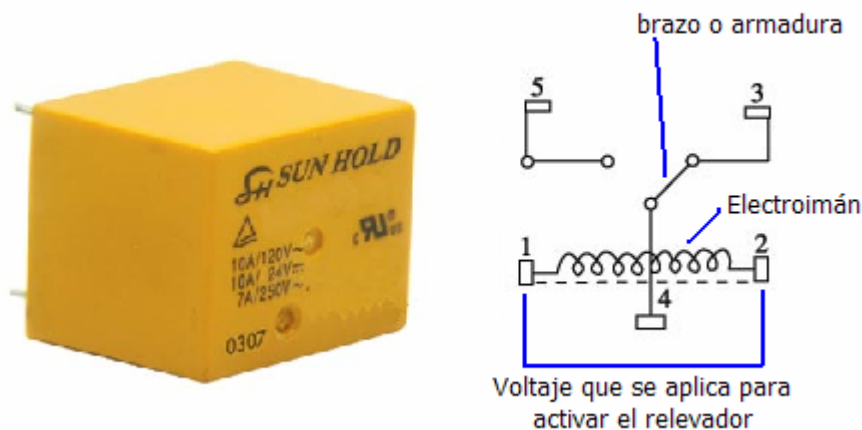


Fig. 2.11, Relevador

Las características del dispositivo vienen especificadas por el número de parte que lleva, RAS especifica el tipo, 05 es el voltaje del relevador que son 5 [V] de DC y el 10 que soporta 10 [A] de corriente, como no lo acompaña ninguna otra letra, se trata de un tipo 1c y su diagrama se muestra a continuación (Fig. 2.12):

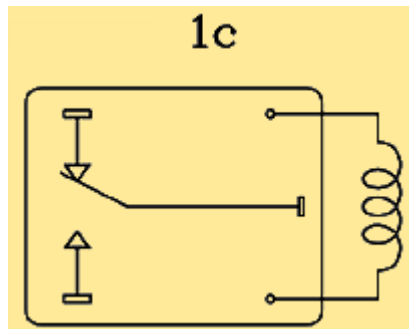


Fig. 2.12, tipo 1c

El brazo se mueve de una posición de normalmente cerrado hacia la otra para hacer contacto. Este dispositivo posee una resistencia interna de 69 [ $\Omega$ ] y consume una potencia de 0.36 [W].

Las salidas del circuito son enviadas a las entradas 0 ,1 ,2 ,3 y 4 del PLC para pasar a su etapa de procesamiento. La configuración que se ha empleado es utilizar el interruptor normalmente cerrado, para que cuando sense un objeto cambie de posición y cierre el circuito que dará la indicación al PLC de que ha pasado una caja en alguna posición ya definida del sistema. El diagrama eléctrico de conexiones completo del sistema se muestra a continuación (Fig. 2.13):

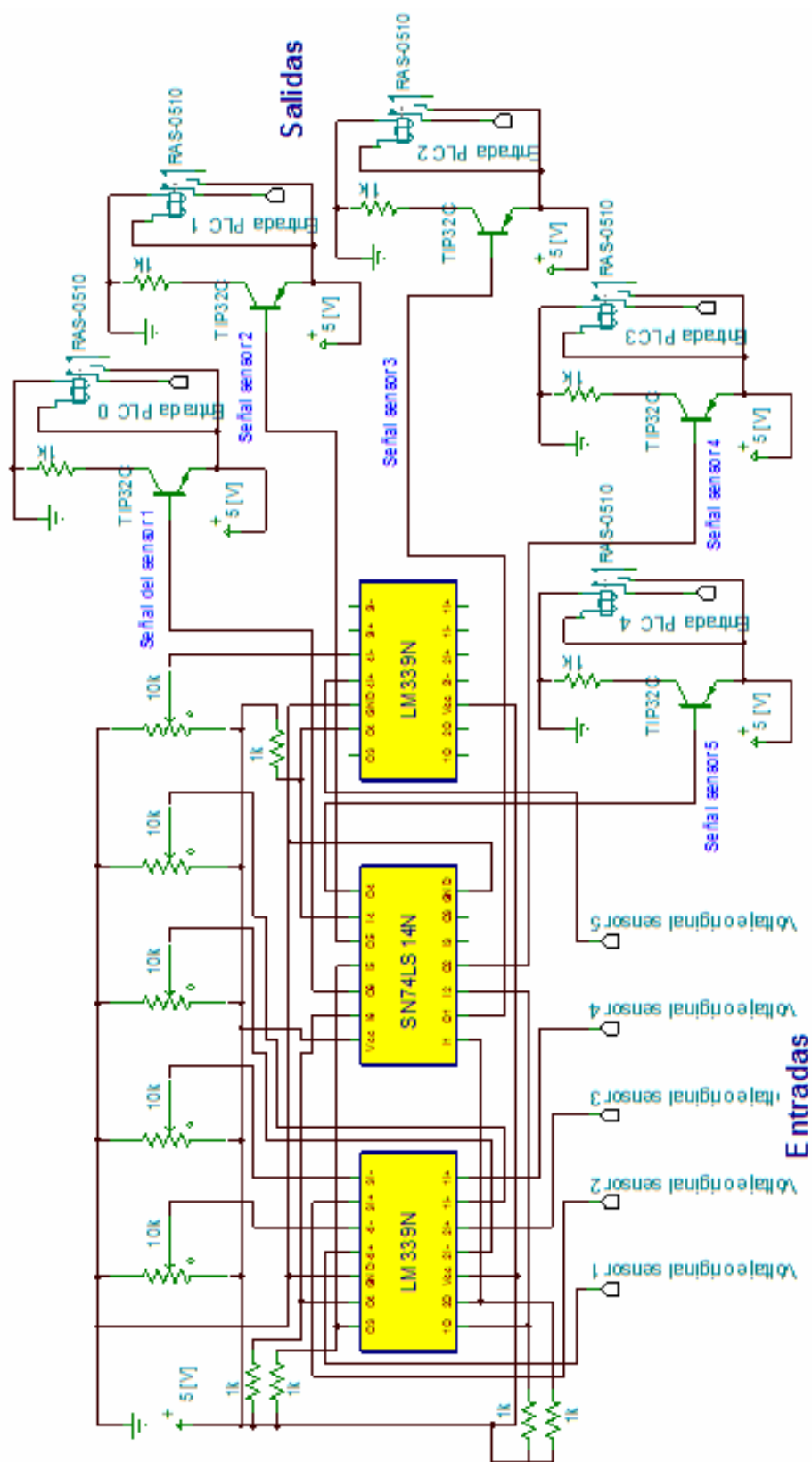


Fig. 2.13, Diagrama de conexión

El PLC que se empleará es el modelo Micro-1<sup>3</sup> de Square-D que cuenta con dos módulos, ocho entradas y seis salidas discretas cada uno. El PLC tiene las siguientes entradas y salidas:

|                 | <b>Micro 1</b> | <b>Expansión</b> |
|-----------------|----------------|------------------|
| <b>Entradas</b> | 0-7            | 10-17            |
| <b>Salidas</b>  | 200-205        | 210-215          |

Además de las entradas de los sensores ya mencionadas, se necesita un botón enclavado para lograr paro total del sistema que se utilizará como medida precautoria. Al oprimir este botón se detendrá el proceso y regresará el estado de los pistones a su posición inicial. Por lo tanto las entradas totales del sistema son:

| <b>Entradas</b> | <b>Descripción</b>                         |
|-----------------|--------------------------------------------|
| <b>0</b>        | Sensor de llenado producto 1               |
| <b>1</b>        | Sensor de llenado producto 2               |
| <b>2</b>        | Sensor de cambio de banda de distribución. |
| <b>3</b>        | Sensor de rampa producto 1                 |
| <b>4</b>        | Sensor de rampa producto 2                 |
| <b>5</b>        | Botón enclavado de paro total del sistema  |

Los actuadores del sistema son los pistones neumáticos, ellos realizarán el trabajo para poder abrir las líneas de los vertederos de producto y para poder provocar el cambio de banda de distribución de la caja. Trabajan bajo las

<sup>3</sup> PLC Micro 1, Apéndice IV

señales de salida del PLC y van directo a electroválvulas (fig. 2.14) 5/2 monoestables y biestables.

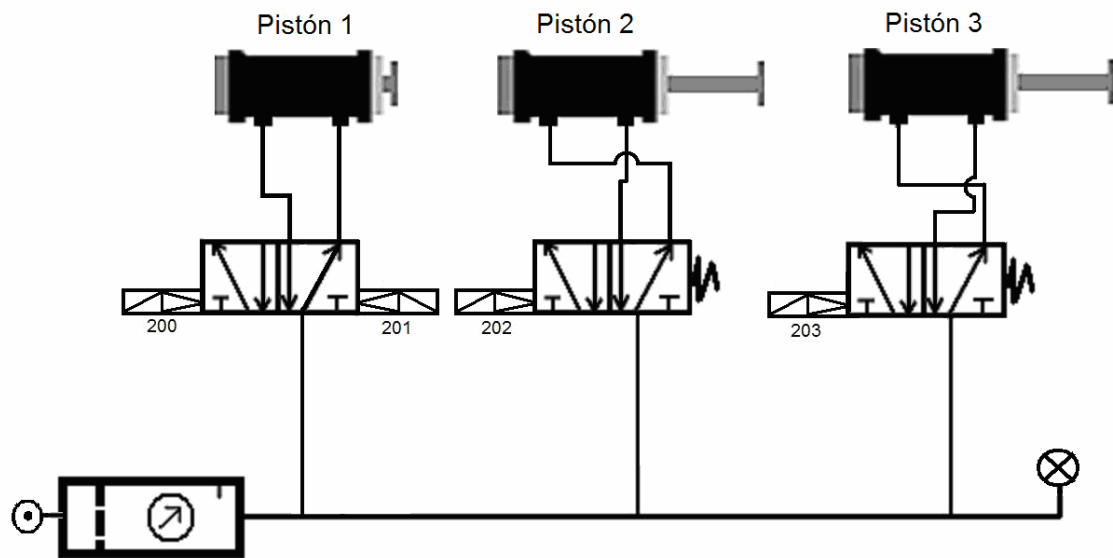


Fig. 2.14, Diagrama neumático

Para el pistón de cambio de banda de distribución, se empleará una válvula biestable, para los pistones de apertura y cierre de los vertedores, serán electroválvulas monoestables. Estos estarán en las siguientes salidas del PLC:

| Salidas    | Descripción                                                         |
|------------|---------------------------------------------------------------------|
| <b>200</b> | C+ (Señal de salida del pistón de cambio de banda de distribución)  |
| <b>201</b> | C- (Señal de entrada del pistón de cambio de banda de distribución) |
| <b>202</b> | A+ (Señal de acción pistón de llenado 1)                            |
| <b>203</b> | B+ (Señal de acción pistón de llenado 2)                            |



Por último se tiene la salida a los motores que mueven las bandas de distribución, ellos son controlados por las señales de salida:

| Salidas    | Descripción                                                                                |
|------------|--------------------------------------------------------------------------------------------|
| <b>204</b> | Motor de DC para la banda de distribución 1 (banda donde se encuentran los vertederos)     |
| <b>205</b> | Motor de DC para la banda de distribución 2 (banda donde se encuentra el pistón de cambio) |
| <b>210</b> | Motor de DC para la banda de distribución 3 (banda recta perpendicular)                    |

El PLC Micro-1 tiene salidas a relevador que funcionan para corriente directa o alterna, por lo tanto para poder accionar los motores se debe de realizar la siguiente conexión de los motores (Fig. 2.15). Este caso está esquematizado para la salida 204, motor de la banda 1, sin embargo la configuración debe ser la misma para las otras dos salidas.

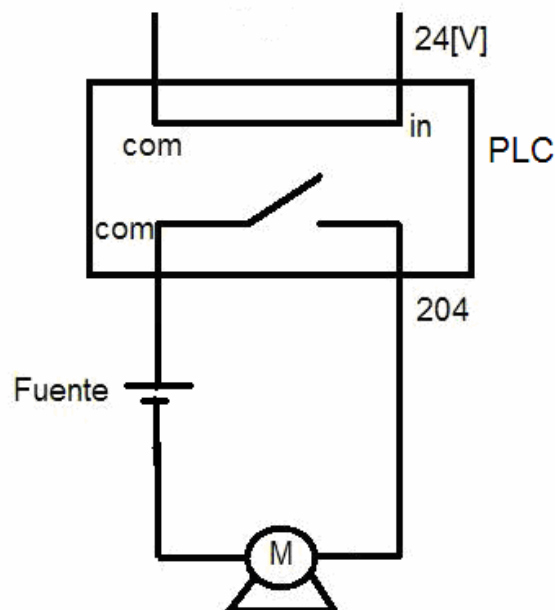


Fig. 2.15, Conexión de motores

Este subtema ha comprendido la forma en la que se realiza el sensado de las cajas, como se hace la rectificación de la corriente para llevarla a un estado de trabajo, la conexión de los pistones neumáticos y la conexión de los motores que mueven las bandas de distribución. Se dio la forma en la que se envían las señales al PLC y la forma en que se reciben. En el próximo apartado se tratará la forma en la que son interpretadas estas señales y los resultados que se obtendrán de ellas.

### **2.3 Lógica de discriminación del proceso**

Ya se ha tratado a fondo la forma en la que los sensores del sistema perciben los estados del sistema, así como la manera en que estos envían las señales de salida. Una vez conociendo esto, se procederá a definir cómo se desea que el sistema de distribución responda a la información enviada por los sensores.

El dispositivo que se empleará para controlar el proceso de discriminación es un PLC de la marca Square-D, modelo Micro 1. Este PLC cuenta con 8 entradas y 6 salidas por sí solo, a su vez se le puede añadir un módulo de expansión con la finalidad de contar con el doble de entradas y salidas, el módulo de expansión sí será necesario en este caso<sup>4</sup>.

En el apartado anterior se tiene ya definido el orden en que irán las señales a las entradas del PLC, posteriormente se expondrá la forma en la que el PLC responde a estas señales.

La discriminación del proceso se debe de hacer con base en reglas que permitan diseñar un algoritmo lógico que cubra las funciones del sistema. Se tiene que definir qué producto se debe de llenar primero cuando haya ausencia de ambos productos y también se debe de llenar un segundo producto, siempre y cuando el primero haya llegado a su posición final.

---

<sup>4</sup> PLC Micro 1, Apéndice IV

Los diagramas que muestran en qué situación se encuentra el sistema a cada momento se llaman "Diagramas de Estados". En un diagrama de estados, cuando se tiene un flanco alto quiere decir que el actuador que se emplea se encuentra realizando una función, se encuentra encendido, es decir es igual a **1** por el contrario, si se encuentra en un flanco bajo, dicho actuador se encuentra inhabilitado o apagado, por lo tanto es igual a **0**.

En este caso se cuenta con tres diagramas de estados: cuando existe la ausencia del producto 1, cuando hay ausencia del producto 2 y cuando se tiene ausencia de ambos productos.

El estado inicial del sistema se presenta con los dos pistones de llenado extendidos para poder tapar a los despachadores de llenado, también presenta las bandas transportadoras detenidas y el pistón de cambio de banda contraído esperando. (Fig. 2.16).

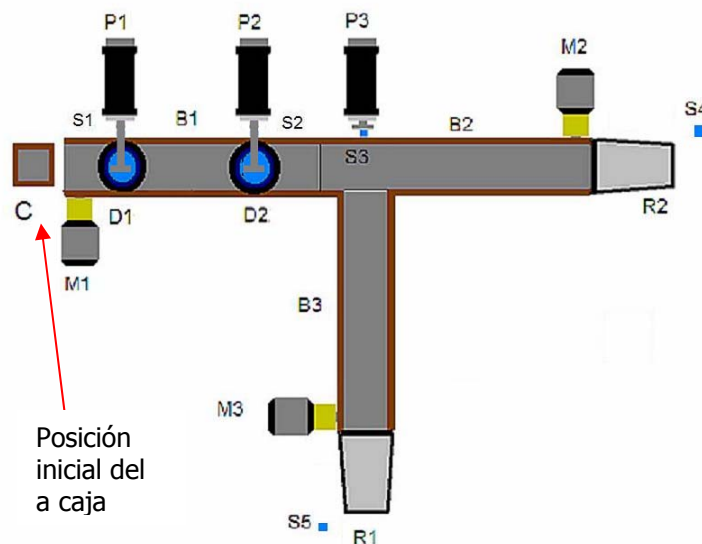


Fig. 2.16, Estado inicial del sistema.

Con la finalidad de que el programa que ejecutará el PLC sea sencillo en cuanto a su explicación y entendimiento, se decidió que la programación sea de forma modular, por lo tanto, se contará con (subrutinas) que se ejecutarán por separado y en conjunto conformarán el programa total del sistema.

La programación se realizó mediante diagramas de escalera, los cuales refieren a una conexión basada en símbolos eléctricos que utilizan la mayoría de las familias de PLCs. Al haberse programado en módulos, se tuvo como consecuencia la existencia de tres subrutinas diferentes: uno cuando hay ausencia del producto 1 y presencia del producto 2 (dicho programa será explicado a continuación), otro cuando hay presencia del producto 1 y ausencia del producto 2, y finalmente el tercer programa para el caso en el que haya ausencia de ambos productos.

Dentro de las restricciones propias de un PLC, naturalmente basadas en su funcionamiento y para que no se caiga en contradicciones durante la ejecución de los programas, la regla fundamental cuando se programa un PLC es que no se deben de repetir las salidas, solo puede existir una salida correspondiente a un grupo de condiciones e instrucciones. Dichas señales de salida serán enviadas hacia cada uno de los motores o pistones según les corresponda. Por esta razón, las condiciones parciales o salidas indirectas serán enviadas a los relevadores internos, posteriormente se asociará un relevador diferente a cada una de las salidas del sistema y así se irá conformando el programa modularmente.

La función de los relevadores es la de encender las salidas. Existe un relevador diferente para cada uno de los programas modulares, pero todos van asociados a una misma salida. Es así como podemos encender el motor de la banda de distribución número 1 (salida 204) con los relevadores 404 o 414. Los relevadores asociados a cada uno de los programas son:

#### Programa 1

| Salidas    | Descripción                                                    | Relevador  |
|------------|----------------------------------------------------------------|------------|
| <b>200</b> | C+ (Extensión del pistón de cambio de banda de distribución)   | <b>400</b> |
| <b>201</b> | C- (Contracción del pistón de cambio de banda de distribución) | <b>401</b> |

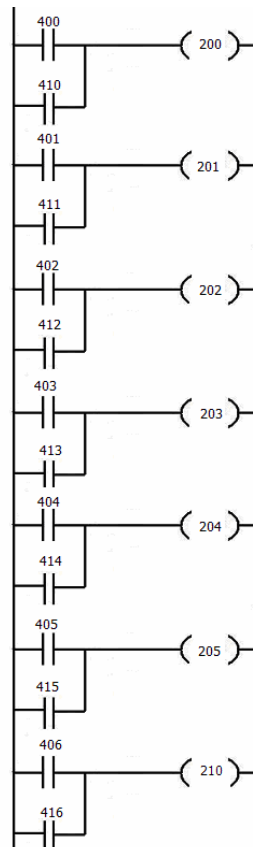
|            |                                                                               |            |
|------------|-------------------------------------------------------------------------------|------------|
| <b>202</b> | A+ (Señal de acción pistón de llenado 1)                                      | <b>402</b> |
| <b>203</b> | B+ (Señal de acción pistón de llenado 2)                                      | <b>403</b> |
| <b>204</b> | Motor de DC para la banda 1<br>(banda donde se encuentran los despachadores)  | <b>404</b> |
| <b>205</b> | Motor de DC para la banda 2<br>(banda donde se encuentra el pistón de cambio) | <b>405</b> |
| <b>210</b> | Motor de DC para la banda 3<br>(banda perpendicular a las otras dos)          | <b>406</b> |

## Programa 2

| <b>Salidas</b> | <b>Descripción</b>                                                            | <b>Relevador</b> |
|----------------|-------------------------------------------------------------------------------|------------------|
| <b>200</b>     | C+ (Extensión del pistón de cambio de banda de distribución)                  | <b>410</b>       |
| <b>201</b>     | C- (Contracción del pistón de cambio de banda de distribución)                | <b>411</b>       |
| <b>202</b>     | A+ (Señal de acción pistón de llenado 1)                                      | <b>412</b>       |
| <b>203</b>     | A+ (Señal de acción pistón de llenado 2)                                      | <b>413</b>       |
| <b>204</b>     | Motor de DC para la banda 1<br>(banda donde se encuentran los vertederos)     | <b>414</b>       |
| <b>205</b>     | Motor de DC para la banda 2<br>(banda donde se encuentra el pistón de cambio) | <b>415</b>       |
| <b>210</b>     | Motor de DC para la banda 3<br>(banda perpendicular a las otras dos)          | <b>416</b>       |

El programa número tres depende de los programas uno y dos, como se verá más adelante, por lo cual no se plantean relevadores distintos para su ejecución.

El diagrama de escalera es capaz de asociar cada uno de los relevadores a la salida a la que están referidos, el diagrama de escalera que asocia cada salida con sus relevadores es el siguiente:



Se asociaron relevadores para cada uno de los programas, cada relevador indica qué programa está activo, con ello se evita que los programas puedan correr de forma simultánea y ocurran colisiones o estados no deseados. Los relevadores son accionados por los sensores del sistema dependiendo de que al final de la rampa exista o no un producto.

Así, si el sensor de la rampa 1 no detecta producto, se ejecutará el programa 1, en el caso de ausencia al final de la rampa 2 se accionará el 2 y cuando haya ausencia en ambos sensores se ejecutará el programa 3. Los relevadores asociados a cada uno de los programas son:

| Programa | Relevador asociado | Condición                                           |
|----------|--------------------|-----------------------------------------------------|
| 1        | 500                | Ausencia del producto 1<br>Presencia del producto 2 |
| 2        | 501                | Presencia del producto 1<br>Ausencia del producto 2 |
| 3        | 502                | Ausencia de ambos productos                         |

**Programa 1: Ausencia del producto 1 y presencia el producto 2**

Cuando exista este estado lo primero que se tiene que hacer es accionar este programa para que sea ejecutado, con ello es garantizado el ejecutar sólo esa parte del programa. Después se activa la banda transportadora hasta que la caja que contiene el producto llegue a la posición de llenado. Cuando el sensor detecte que la caja ha llegado al despachador correspondiente, se detendrá la banda transportadora y se retraerá el pistón de llenado del producto 1 durante un periodo de 10[s] para que por gravedad se libere el producto y la caja sea llenada.

Éste estado del proceso es mostrado en la siguiente figura (Fig. 2.17):

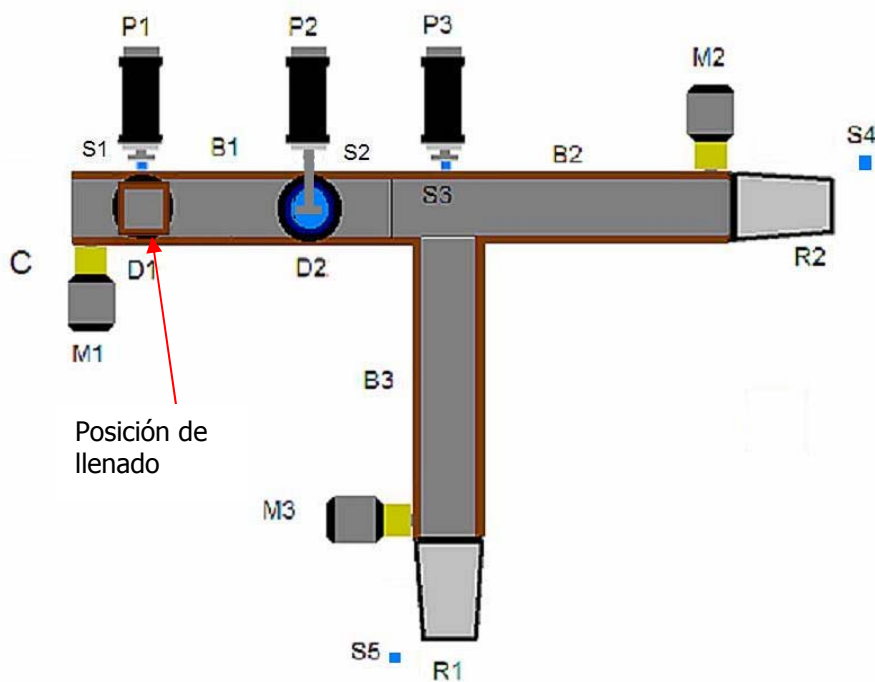
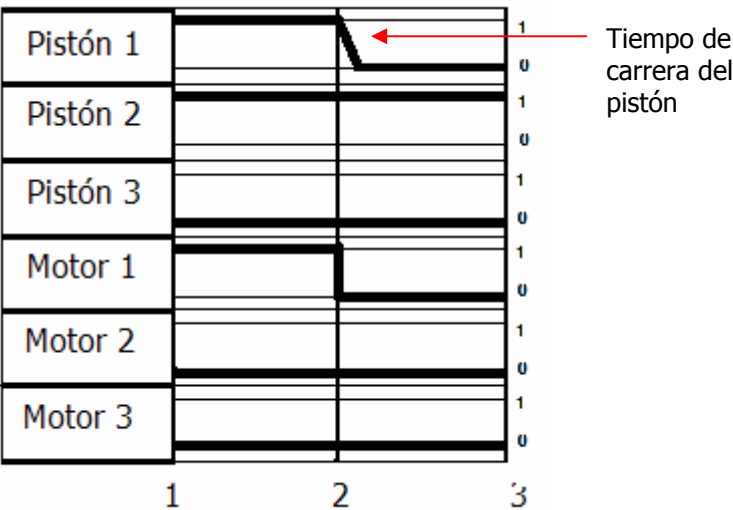


Fig. 2.17, Programa 1 - Posición siguiente del paquete.

El diagrama de estados del segmento explicado se presenta a continuación:



Después de transcurrido el tiempo de llenado, se cierra el despachador y se activa de nuevo la banda 1 y ahora también la banda 2 para que la caja pueda continuar con su desplazamiento.

El siguiente estado se presenta llegando al sensor del pistón 3, se activa la banda número 3, se desactiva la banda 1 y el pistón 3 cambia de estado. Y con esto el pistón 3 mueve la caja de la banda 1 a la banda 3.

La caja sigue su rumbo hasta llegar al final de la rampa 1. Al llegar al sensor de la rampa 1, éste detecta su presencia, desactiva la banda 3, la banda 2 y regresa el pistón 3 a su posición inicial (Fig. 2.18).



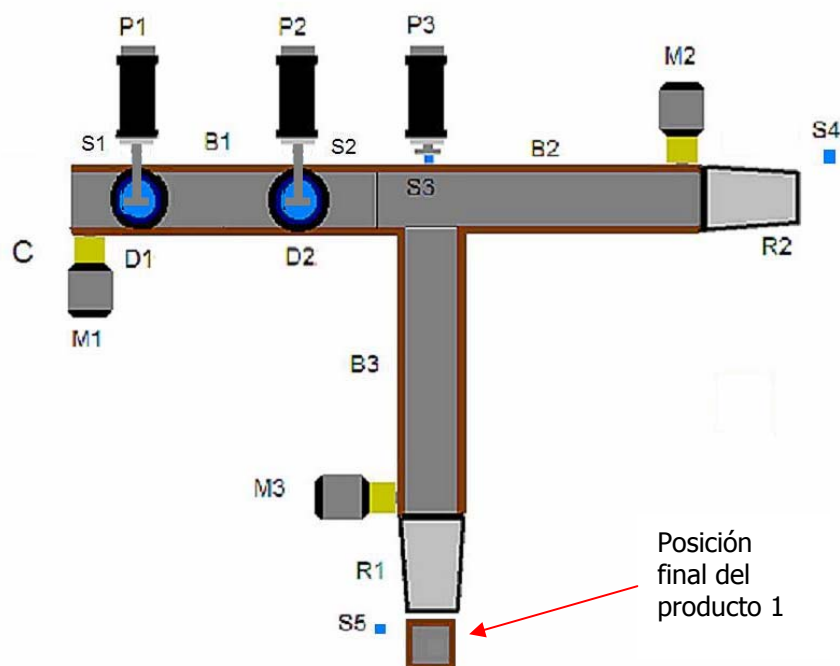
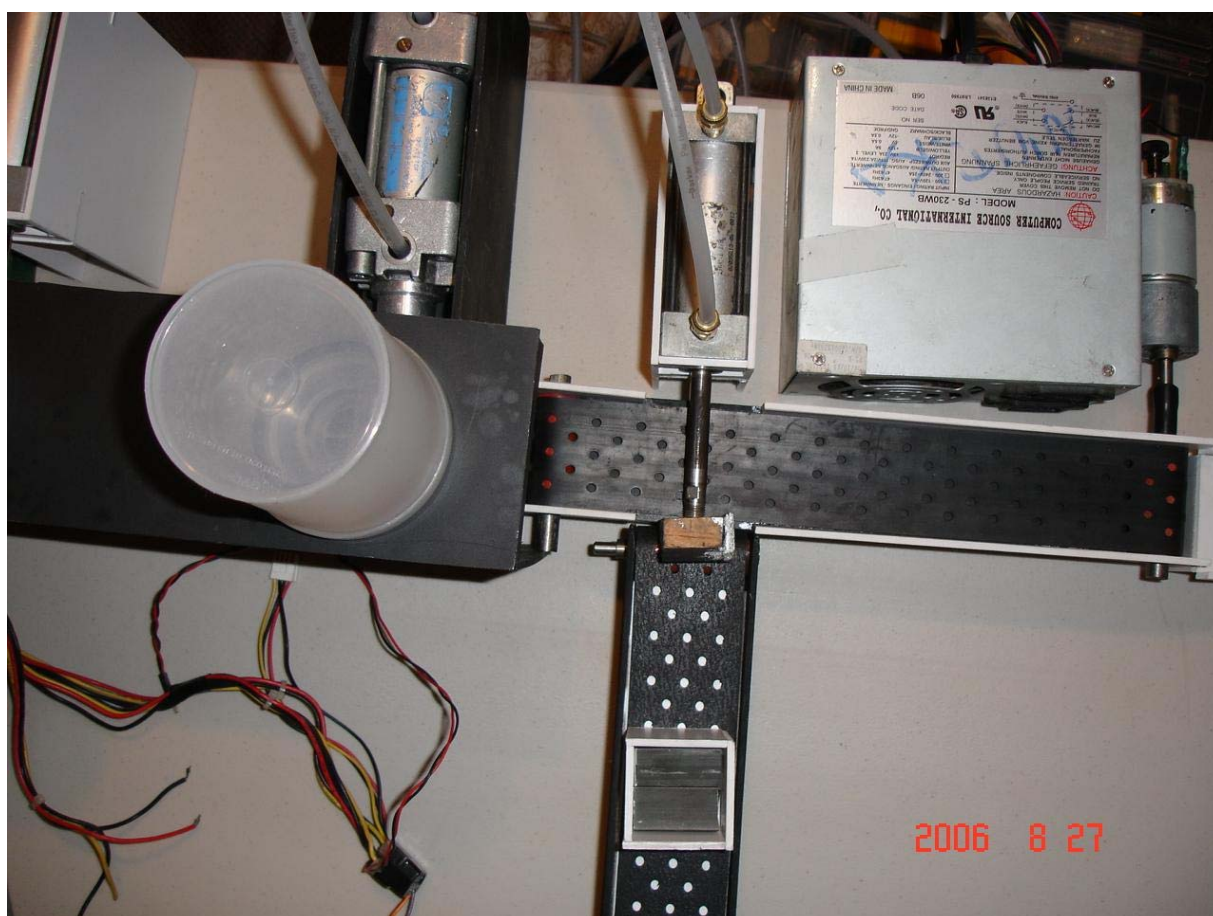
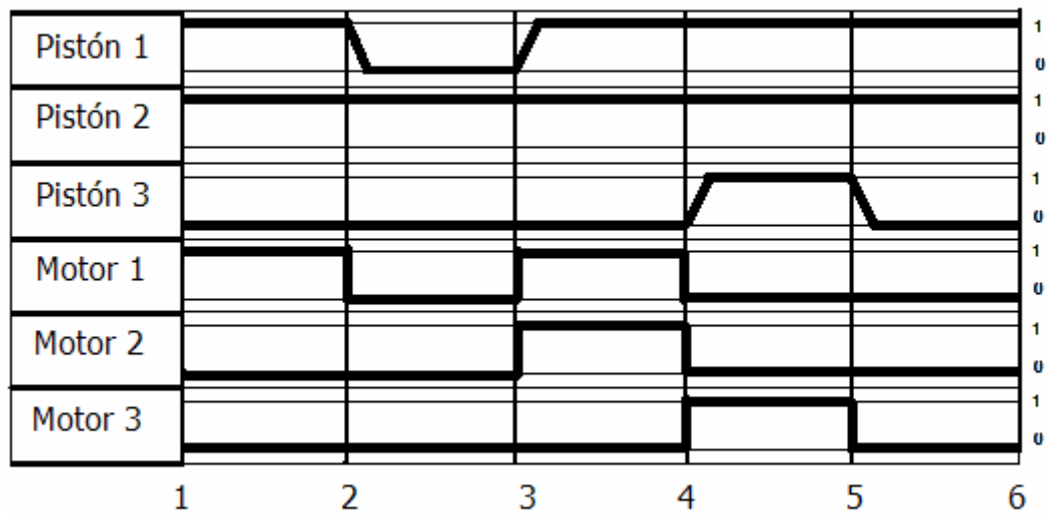


Fig. 2.18, Programa 1 – Posición final del paquete.



El diagrama de estados del programa 1 es el siguiente:



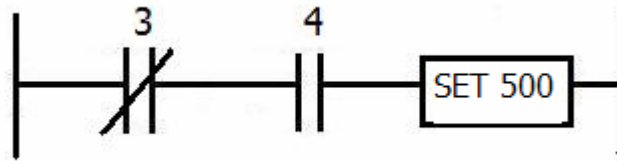
Cada cambio de estado está referido a algún suceso detectado dentro del sistema, en este caso cada hecho que interviene está referido a un número del 1 al 6:

- 1.- Inicio del proceso.
- 2.- Sensor debajo del Despachador 1 **detecta al paquete** . (Inicio del Timer)
- 3.- Tiempo cumplido del Timer, llenado de la caja concluido.
- 4.- Sensor de cambio de banda **detecta al paquete**.
- 5.- Sensor de final de rampa 1 **detecta al paquete**.
- 6.- Fin del proceso

Ahora se explicará el programa que será cargado en el PLC. La explicación se realizará de manera similar a cómo se explicaron los programas del microcontrolador.

Las condiciones que se deben de cumplir para que corra este programa son: el sensor de la rampa producto 2 sensando (entrada 4), sensor de rampa producto 1 sin sensar (entrada 3) al cumplirse estos 2 estados se activa el relevador asociado al programa 1 (relevador 500). En el siguiente diagrama de escalera el interruptor normalmente abierto es el que va a hacer que se

encienda el relevador 500 debido a que el interruptor normalmente cerrado permite la continuidad del peldaño.



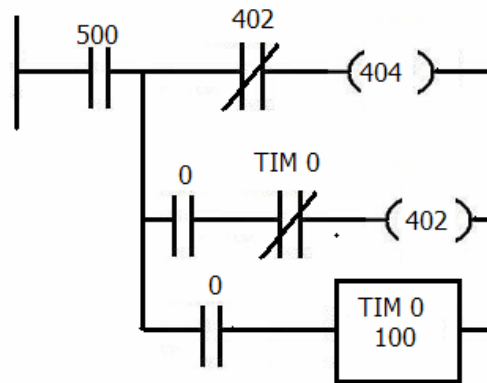
El relevador 500 cubre un master control set (MCS)<sup>5</sup>, por lo tanto las condiciones que se encuentren después de él únicamente se cumplirán si el relevador está activo.

El programa 1 arranca, lo primero que se debe de hacer es accionar la banda 1 (salida 204 = relevador 404) y dejarla en movimiento hasta que la caja llegue a la posición del sensor de llenado del producto 1 (entrada 0). En ese momento se acciona un Timer<sup>6</sup> durante 10[s] y se retrae el pistón de llenado de producto 1 (salida 202 = relevador 402) para llenar la caja. La banda debe de mantenerse detenida mientras el pistón de llenado se encuentre retraído y el producto esté siendo despachado.

En el diagrama de escalera se muestra el peldaño en el cual se activa la banda mientras el pistón se encuentra retraído, en el siguiente peldaño se muestra cómo se activa el pistón 1 (salida 202 = relevador 402) cuando el sensor de llenado está detectando al paquete frente a él (entrada 0) y mientras tanto el Timer0 continúa con su conteo. En el siguiente peldaño se enciende el Timer0 cuando el sensor de llenado detecta al paquete.

<sup>5</sup> MCS (Master Control Set), Apéndice IV

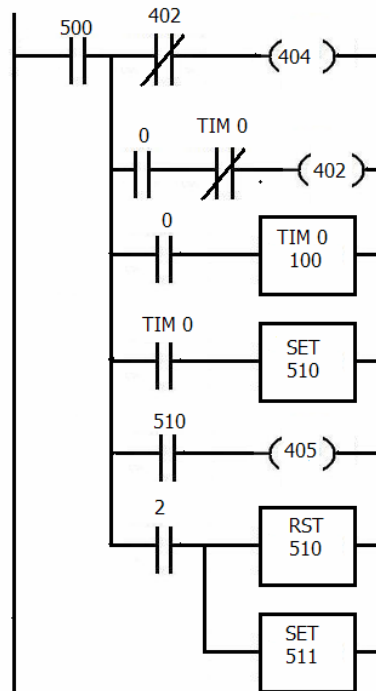
<sup>6</sup> Programación de los Timers en el PLC, Apéndice IV



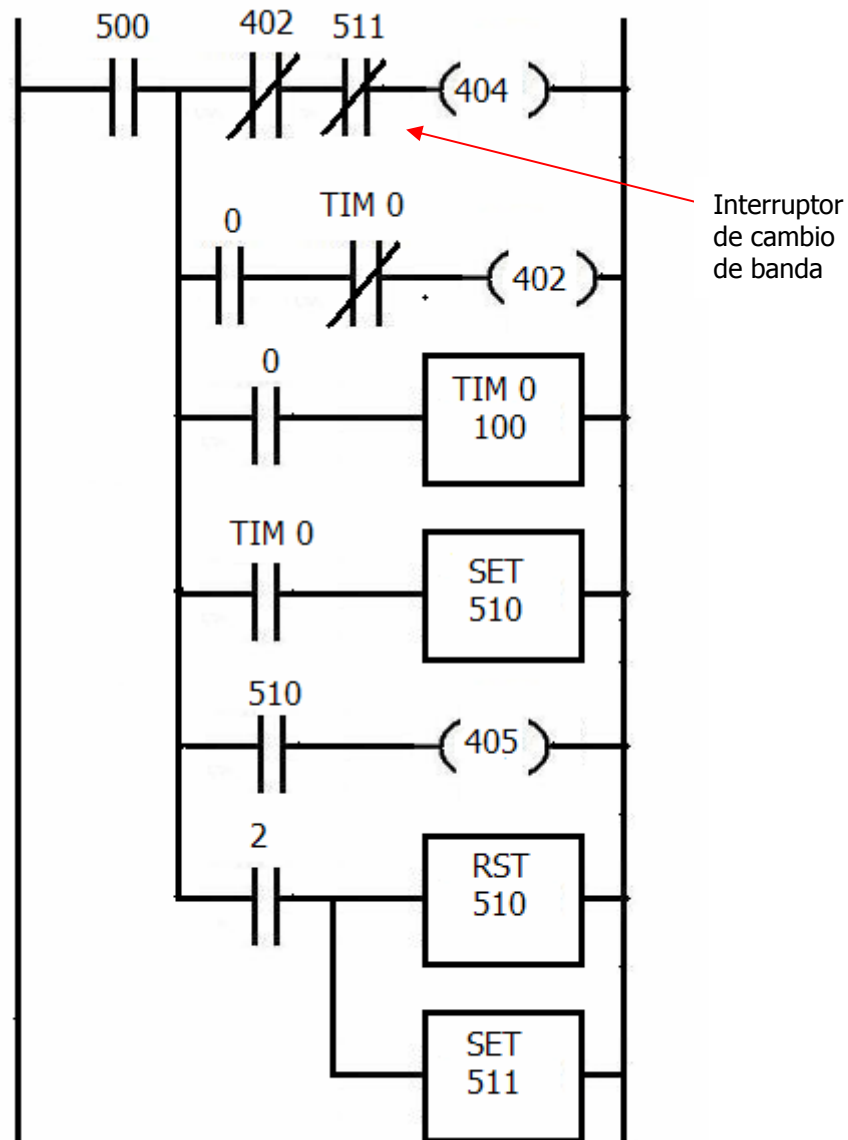
Una vez que es llenada la caja, se activa de nuevo el motor de la banda 1 debido a que la señal de acción del pistón se ha retirado como se puede observar en el primer peldaño del diagrama anterior.

Cuando el Timer0 ha terminado su cuenta se enciende otro relevador, este relevador enciende la bandera que indica que el proceso de llenado ha concluido, para esta función se utilizará el relevador número 510. La banda 2 (salida 205 = relevador 405) debe de funcionar justo cuando el proceso de llenado ha terminado, así que se asocia al relevador que se accionó (510) y debe permanecer en movimiento hasta que el sensor de cambio de banda (entrada 2) detecte la presencia del paquete.

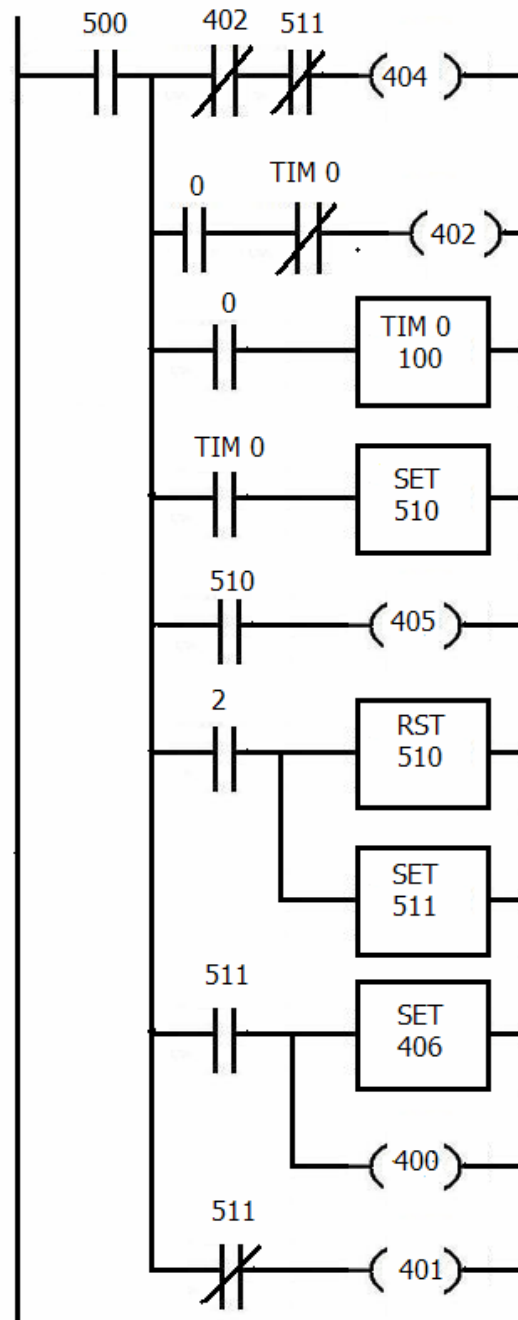
En el peldaño cuatro se cuenta con el relevador que indica que terminó el proceso de llenado (510) asociado la finalización del conteo en el Timer0. En el peldaño cinco se enciende el motor de la banda 2 (salida 205 = relevador 405), así como el relevador de finalización de llenado (510). En el peldaño seis se apaga el relevador de término de llenado (510) con ello se detiene la banda 2 (salida 205) y se asocia un relevador nuevo que indica que se pasa al estado cambio de banda, para este efecto se utilizará el relevador 511.



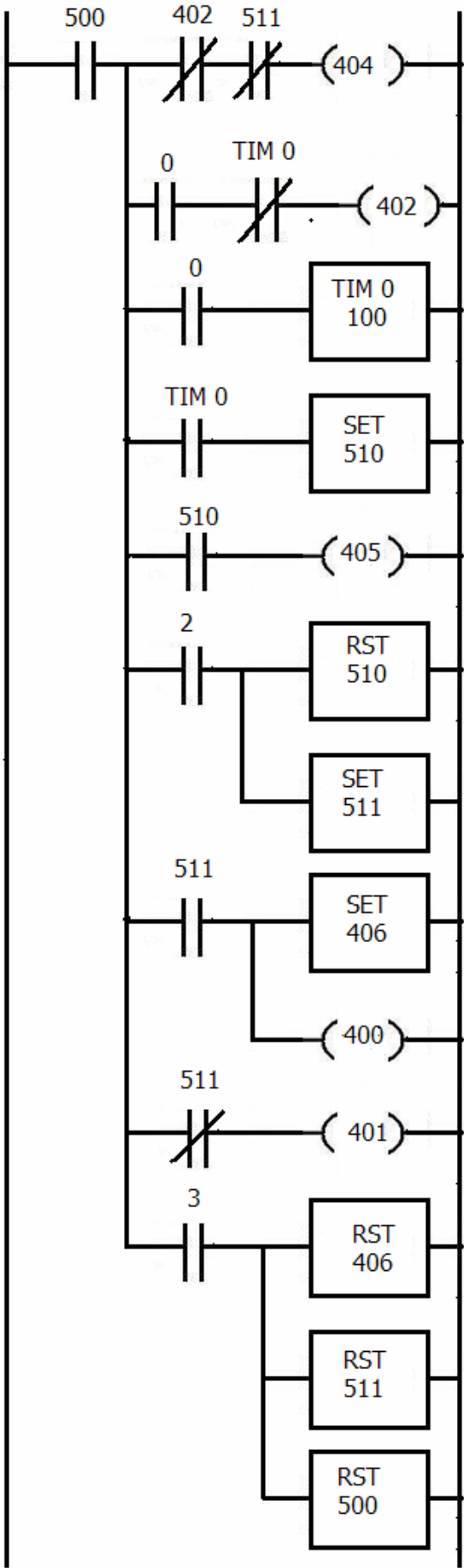
Finalmente se agrega una condición más al programa, ya no que no es deseado que la banda 1 (salida 204) siga en movimiento cuando se va a realizar el cambio a la banda 3 por lo tanto se añadirá un interruptor al primer peldaño del programa. Se asocia la banda 1 (salida 204) al relevador que indica que se pasó a la fase de cambio de banda (511) y con ello se apaga cuando se realiza el cambio de banda.



Con el relevador de cambio de banda (511) se enciende el motor de la banda número 3 (salida 206 = relevador 406) y se acciona el pistón de cambio de banda. Como el pistón de cambio de banda es un pistón de doble efecto, se necesita enviar una señal por uno de los pilotajes (salida 200 = relevador 400) y retirar la señal del otro (salida 201 = relevador 401).



La caja continúa su movimiento por la banda 3 (salida 210 = relevador 416) hasta que el sensor de la rampa 1 (entrada 0) detecta la caja en esa posición. En ese punto se detiene la banda 3 (salida 210 = relevador 416), se apaga el relevador 511 (con ello regresa a su posición inicial el pistón de cambio de banda) y se apaga la bandera del programa 2 (relevador 500).





Programa 2: Presencia del producto 1 y ausencia el producto 2

En este estado se debe activar la banda transportadora hasta que la caja llegue a la posición de llenado del producto numero 2, cuando el sensor detecte que la caja ha llegado a la posición de llenado, se tiene que detener la banda transportadora y retraer el pistón de llenado del producto 2 durante 10 [s].

Este estado del proceso es mostrado en la siguiente figura (Fig. 2.19):

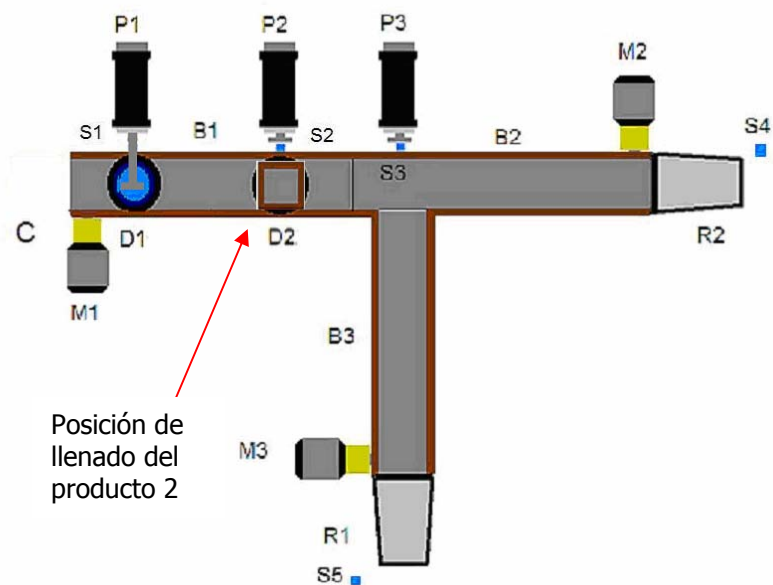
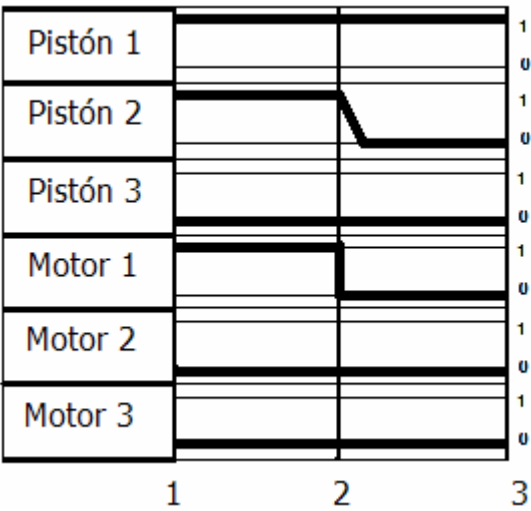


Fig. 2.19, Programa 2 - Posición de llenado.

El diagrama de estados del proceso para el programa 2 es el siguiente:



Después de transcurrido el tiempo de llenado, se cierra el despachador, se activa de nuevo la banda 1 y se acciona la banda número 2 para que la caja pueda continuar con su movimiento.

Al llegar a la posición donde se encuentra el sensor de cambio de banda, se desactiva la banda 1 y la caja sigue el rumbo hasta llegar a la rampa de recolección 2. Cuando el sensor de la rampa 2 detecta la presencia de la caja, se desactiva la banda 2. (Fig. 2.20)

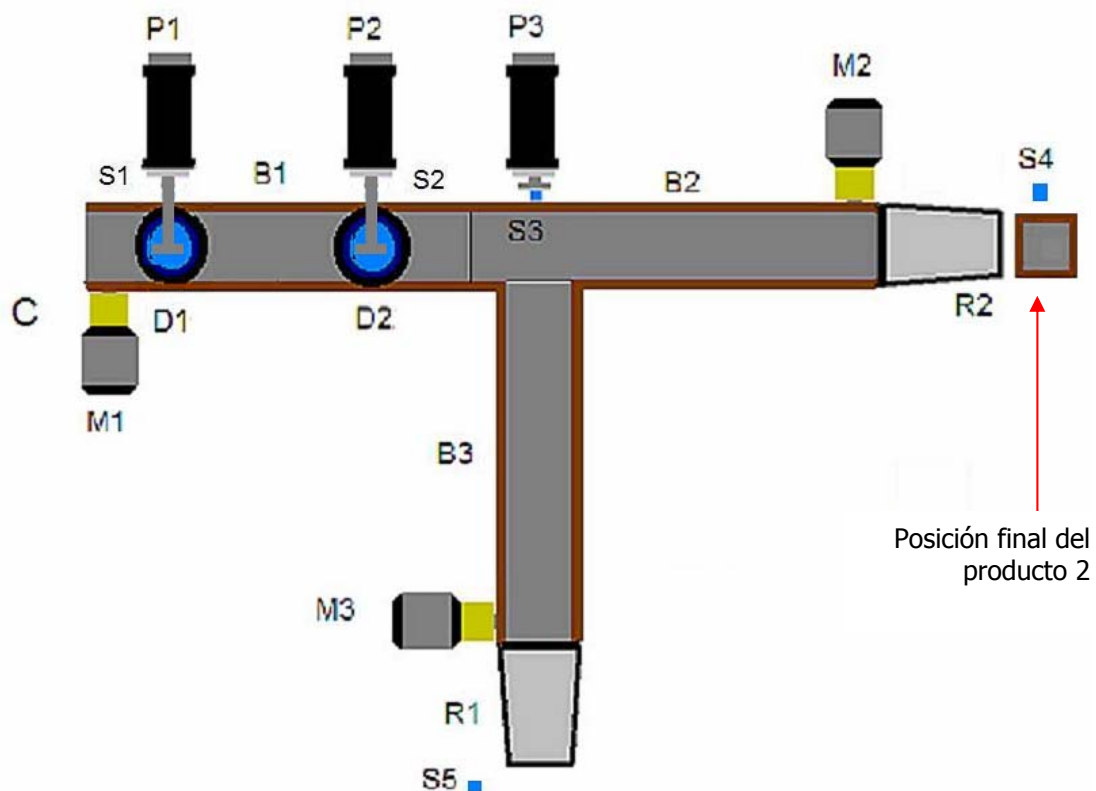
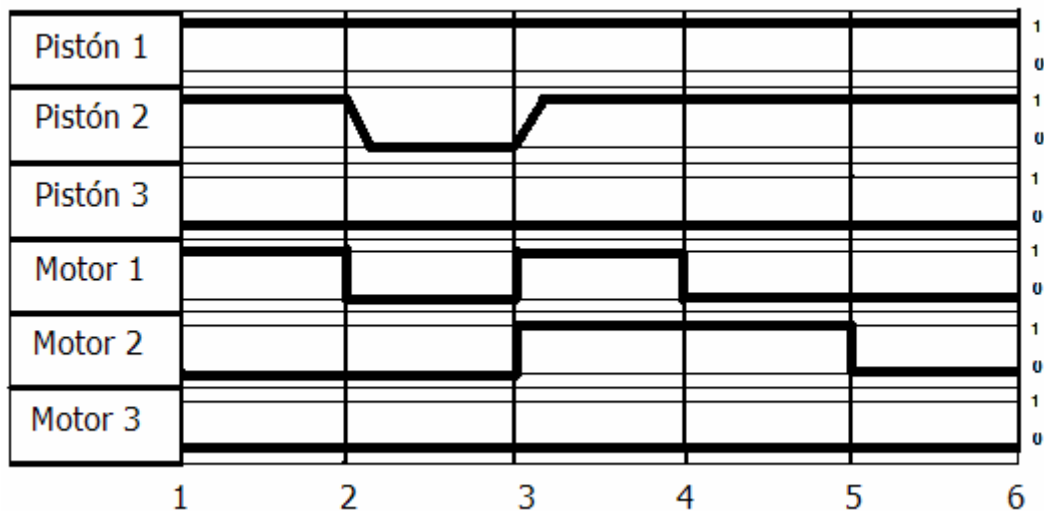


Fig. 2.20, Programa 2 – Posición final.



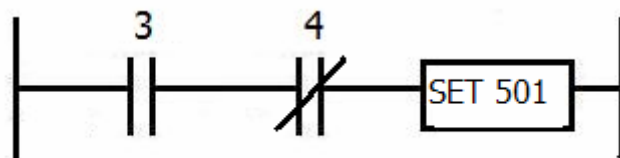
El diagrama de estados del programa 2 es el siguiente:



Donde los números refieren a:

- 1.- Inicio del proceso.
- 2.- Sensor debajo del Despachador 2 **detecta al paquete** . (Inicio del Timer)
- 3.- Tiempo cumplido del Timer, llenado de la caja concluido.
- 4.- Sensor de cambio de banda **detecta al paquete**.
- 5.- Sensor de final de rampa 2 **detecta al paquete**.
- 6.- Fin del proceso

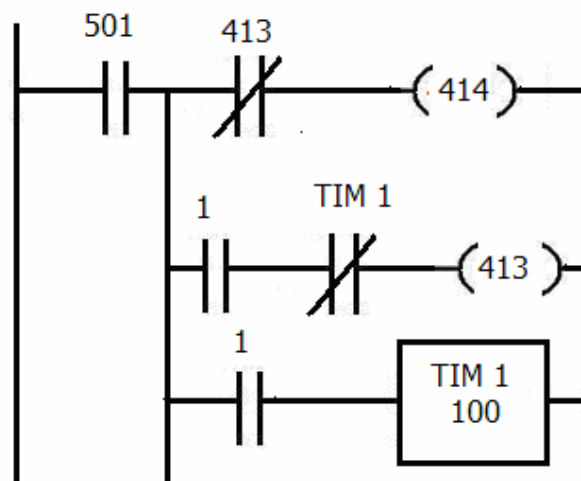
Las condiciones que se deben de cumplir para que corra este programa son: el sensor de la rampa del producto 1 activado (entrada 3) y sensor del final de la rampa del producto 2 desactivado (entrada 4). Al cumplirse estos dos estados automáticamente se enciende el relevador asociado al programa numero 2 (501).



El relevador 501 funciona, al igual que en el programa 1, es una condición primordial de todas las demás, por lo tanto se utiliza un master control set para asociar dicha condición a las demás. Lo primero que debe de hacer es encender el motor de la banda 1 (salida 204 = relevador 414) y dejarlo

encendida hasta que la caja llegue a la posición del sensor de llenado del producto 2 (entrada 1). En ese momento se activa el Timer1 durante 10 [s] y se retrae el pistón de llenado de producto 2 (salida 203 = relevador 413) para llenar la caja. La banda debe de mantenerse apagada mientras el pistón de llenado se encuentre contraído.

En el diagrama de escalera se muestra el peldaño en el cual se activa la banda mientras el pistón se encuentre retraído, en el siguiente peldaño se muestra cómo se activa el pistón 2 (salida 203 = relevador 413) cuando el sensor de llenado detecta (entrada 1) y mientras el Timer1 no ha llegado a su cuenta máxima. En el siguiente peldaño se enciende el Timer1 cuando el sensor de llenado detecta la presencia del producto.

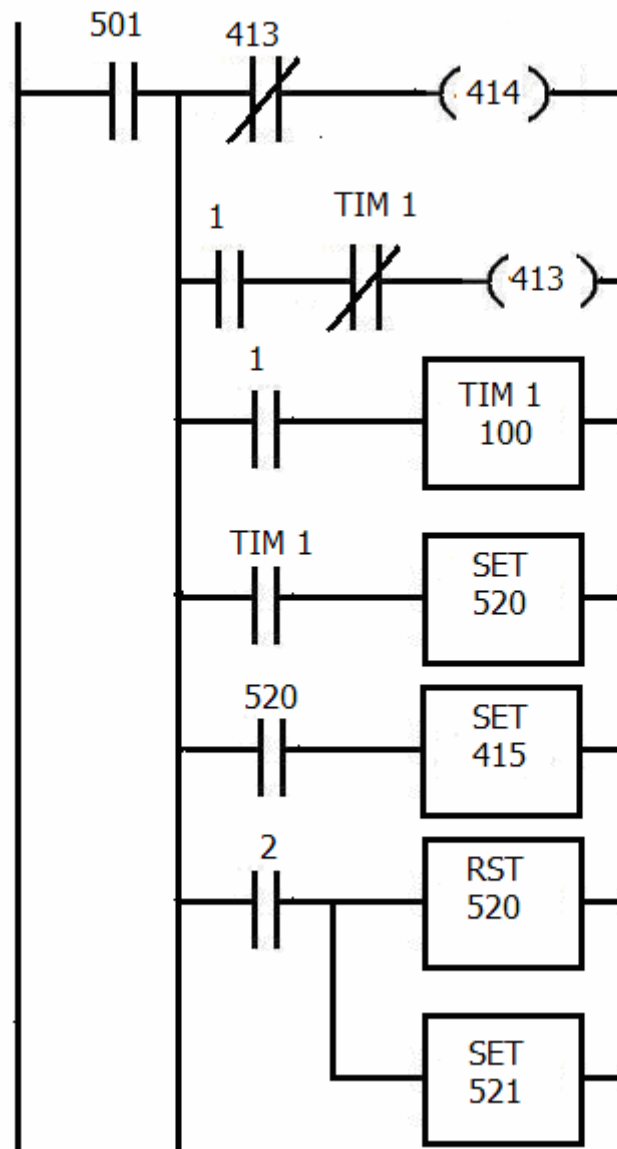


Una vez que se llena la caja, se enciende el motor de la banda 1 debido a que la señal de acción del pistón se ha retirado como se puede observar en el primer peldaño del diagrama anterior.

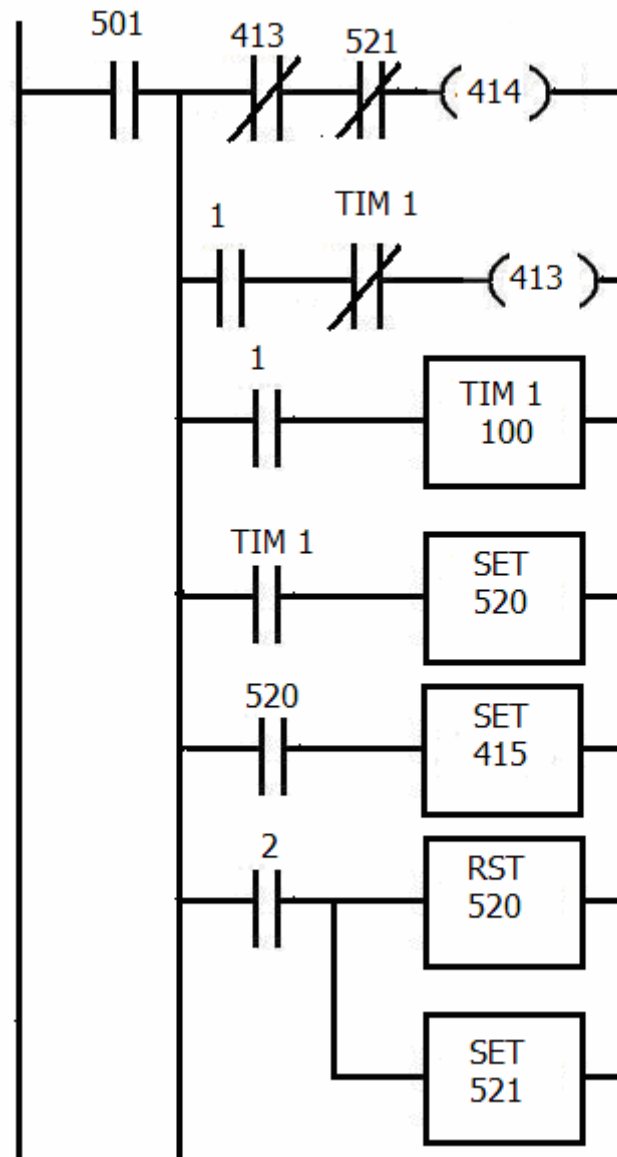
Cuando el Timer1 ha terminado su cuenta se enciende otro relevador, éste indica que el proceso de llenado ha concluido, se utiliza para este efecto el relevador 520. La banda 2 (salida 205 = relevador 415) debe de accionarse justo cuando el proceso de llenado haya terminado, así que se asocia al relevador que se encendió (520) y la banda debe de permanecer en

movimiento hasta que el sensor de cambio de banda (entrada 2) detecte al paquete.

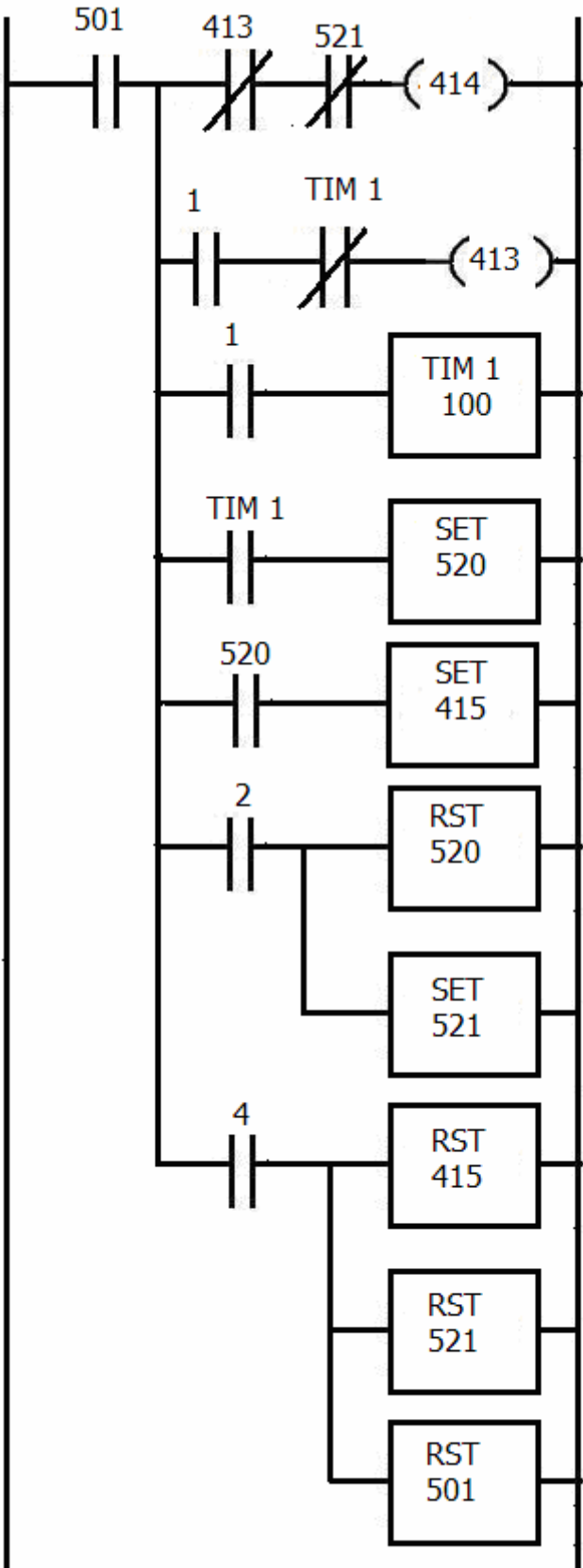
En el peldaño cuatro se tiene al relevador que indica que terminó el proceso de llenado (520) asociado con la finalización del conteo en el Timer1. Posteriormente, en el peldaño cinco se activa la banda 2 (salida 205 = relevador 415) junto con el relevador de finalización de llenado (520). Cuando termina esa fase, en el peldaño seis se desactiva el relevador de término de llenado (520) con ello se detiene la banda 2 (salida 205) y se asocia un relevador nuevo que indica que se está en el pistón de cambio de banda, para este efecto se empleará al relevador 521.



De igual manera que en el programa 1, se agregará una condición más al programa, ya que no se desea que la banda 1 (salida 204) siga activada debido a que el producto se encuentra en la banda 3, entonces se añade un interruptor al primer peldaño del programa. Ahora se asocia la banda 1 (salida 204) al relevador que indica que se pasó a la fase de cambio de banda (521), de hecho este relevador sólo tiene la función de detener la banda número 1.

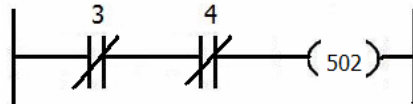


La caja continúa su movimiento por la banda 2 (salida 204 = relevador 414) hasta que el sensor del final de la rampa 2 (entrada 4) detecta la caja en esa posición. En ese punto es detenida la banda 2 (salida 205 = relevador 415), se apaga el relevador 521 y es apagada la bandera del programa 2 (501).

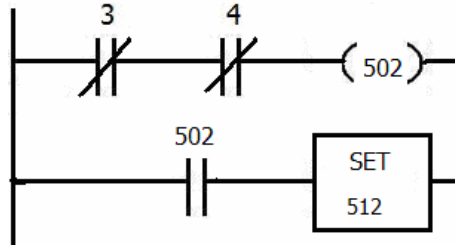


**Programa 3: Ausencia de ambos productos.**

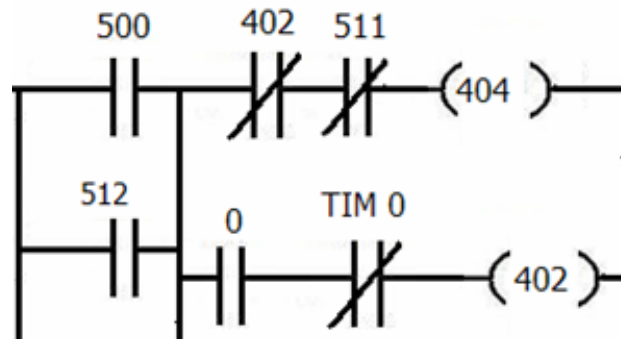
Las condiciones que se deben de cumplir para que se ejecute este programa son: el sensor de final de rampa del producto 1 desactivado (entrada 3) y sensor de final de rampa del producto 2 desactivado (entrada 4). Al cumplirse estas dos condiciones, automáticamente se activa el relevador asociado al programa número 3 (502).



En este caso el programa número tres está asociado directamente al programa 1. Cuando el programa 3 sea activado (502) el programa 1 es ejecutado (501).



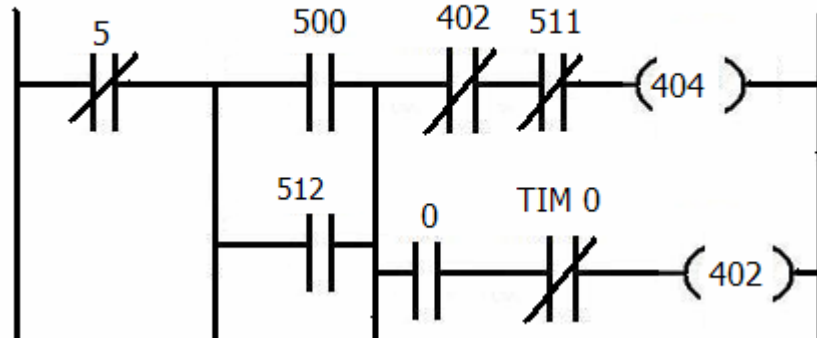
Al terminar de correr el programa 1, el estado final del sistema del sensor de la rampa 1 es activado (entrada 0), sensor de la rampa 2 desactivado (entrada 1), este estado es el del programa 2, enciende el programa 2 y se completan las rutinas de cada producto. Por lo tanto el programa tres es sólo una rutina progresiva primero del programa 1 y luego del programa 2. En consecuencia se coloca en paralelo el relevador 500 y el 512 al inicio de la rutina 1.



Por último se agrega un botón enclavado para el paro total del sistema. El botón de paro (entrada 5) funciona como un master control set. Cuando éste es



accionado, se regresan todos los pistones a su posición inicial y son detenidos todos los motores.



En este capítulo se trató el tema de diseño y automatización de un proceso de llenado por medio de electroneumática. Se cumplieron todos los objetivos esperados, la integración con el montacargas seguidor de línea se tratará en el siguiente capítulo.

Nota: Con la finalidad de contar con una mayor explicación en cuanto a las funciones del PLC y/o conocer el código más a fondo, se pueden consultar las instrucciones del PLC así como el programa completo en el apéndice IV.

## **Integración**

### **3.1 Acoplamiento de sub-sistemas**

En los anteriores capítulos se ha explicado profundamente la forma en la que se diseñaron y elaboraron los sistemas que intervienen individualmente en el proceso de llenado y recolección de productos. Sin embargo, cada uno de estos subsistemas depende de acciones que se realizan paralelamente a su funcionamiento, estas acciones a su vez son de importancia para cada sub-sistema y, por lo tanto, repercuten al momento de realizar sus tareas y en algunos casos en sus decisiones. Por consiguiente, este apartado tratará la forma en que se integran para que puedan trabajar todos como uno solo.

Concretamente, para el caso de las bandas transportadoras, no se tiene la ventaja de flexibilidad, ya que su configuración es rígida, además de que conforma la parte mecánica de todo el sistema que emula directamente un proceso industrial, y en la mayoría de los casos, resultaría muy difícil realizar un cambio significativo.

Por lo tanto, la adaptación está basada en la ruta de desplazamiento del montacargas seguidor de línea. El seguidor de línea tiene la capacidad de desplazarse en cualquier dirección y puede llegar a cualquier lugar requerido siempre y cuando las rutas que deba seguir el AGV no presenten curvas muy pronunciadas.

En consecuencia, el AGV tiene la capacidad de llegar a la posición donde se encuentran las rampas finales que reciben los contenedores de cada producto.

El AGV debe de salir de un punto determinado al cual se le llamará, a partir de este momento, "HOME". El "HOME" se encuentra a una distancia significativa del punto de recolección de productos, y el montacargas deberá de permanecer en ese lugar hasta que el usuario de la orden de recolección.

El “HOME” es el mismo indistintamente del producto que se desea recoger. Cerca del HOME se encuentran los puntos de entrega de producto después de terminado su recorrido a lo largo de la pista. (Fig. 3.1)

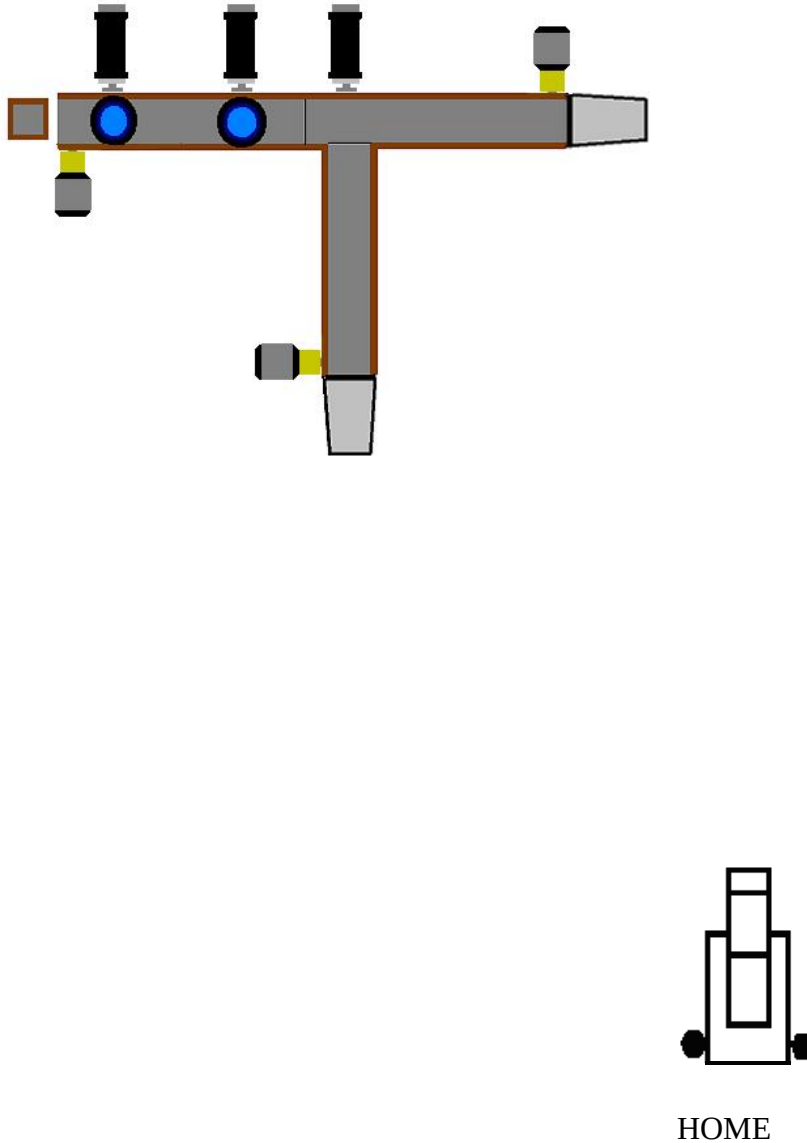


Fig. 3.1, Posición de “HOME”

El AGV tiene la capacidad de recolectar sólo un producto cada vez que realiza la trayectoria de recolección; el producto que el AGV debe de recolectar debe de

definirse antes de que el montacargas comience su recorrido a lo largo de la pista de desplazamiento.

El montacargas seguidor de línea sigue una trayectoria recta en dirección de las bandas de distribución, hasta llegar a un punto de decisión. En este punto el montacargas decide la trayectoria que ha de seguir entre dos rutas posibles, una que lleva la dirección de producto 1 y otra que lleva a la rampa en la que se encuentra el producto 2.

Se decidió que el punto de toma de decisión sea al momento en el que el AGV llegue a una discontinuidad de línea en la pista. Así, el AGV revisará la información que se le haya enviado con anterioridad y tomará la decisión que le corresponda. Esta discontinuidad dará pie a una combinación en las señales de los sensores ópticos que técnicamente serán tres ceros, uno por cada sensor. Por lo tanto, la toma de decisiones ocurrirá al momento de que por primera vez no detecte nada ningún sensor (Fig. 3.2).

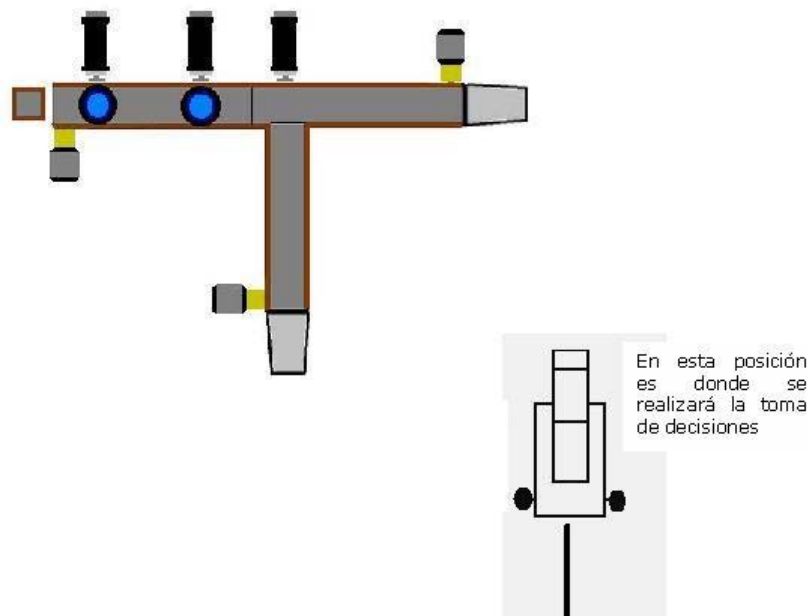


Fig. 3.2, Toma de decisiones.

Posteriormente el vehículo tomará una ruta, la de la izquierda o la que está frente a él en línea recta, en caso de que elija seguir por la ruta de la izquierda irá a recoger el producto 1, para tomar esta ruta deberá girar, hacia la izquierda, sobre el eje perpendicular al eje de la tracción, y continuará girando hasta que el sensor del centro encuentre la línea blanca de la ruta del producto. Por el contrario, si al vehículo se le ordenó recoger el producto 2, únicamente deberá seguir derecho hasta encontrar la ruta que lo lleve a su destino.

En ambos casos el montacargas seguidor de línea continuará su trayectoria a lo largo de la línea trazada en el piso, hasta encontrar el siguiente punto de decisión. Este punto de decisión tiene la configuración opuesta al anterior, en este caso los tres sensores detectan el color de la pista (Fig. 3.3).

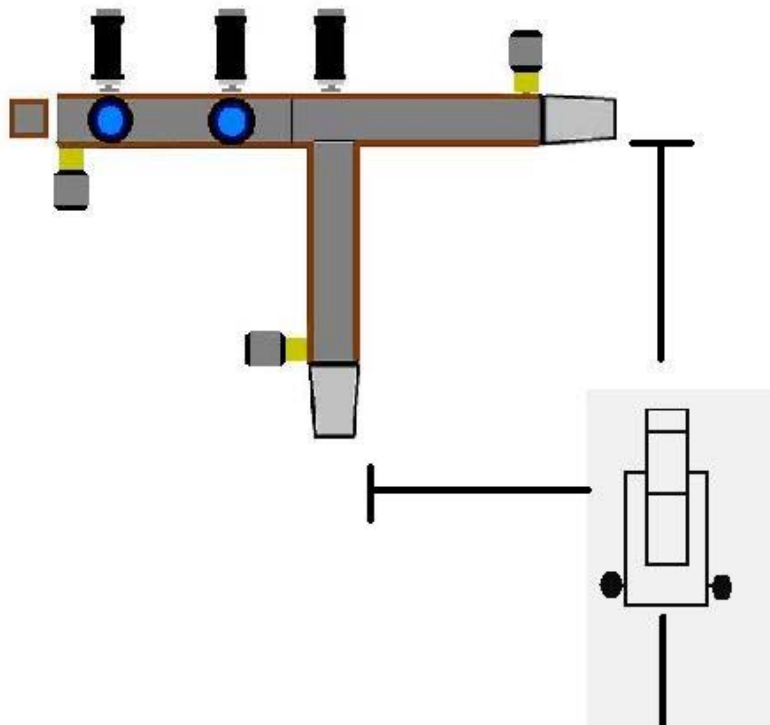


Fig. 3.3, Elección de la ruta.

En el caso de que el AGV fuera a recoger el producto 2, el montacargas debe de levantar la pinza cuando encuentre el segundo punto de decisión. Ya con el

producto levantado, el montacargas girará a su derecha hasta encontrar otra línea la cual debe de seguir hasta encontrar el punto de entrega del producto número 2 (Fig. 3.4).

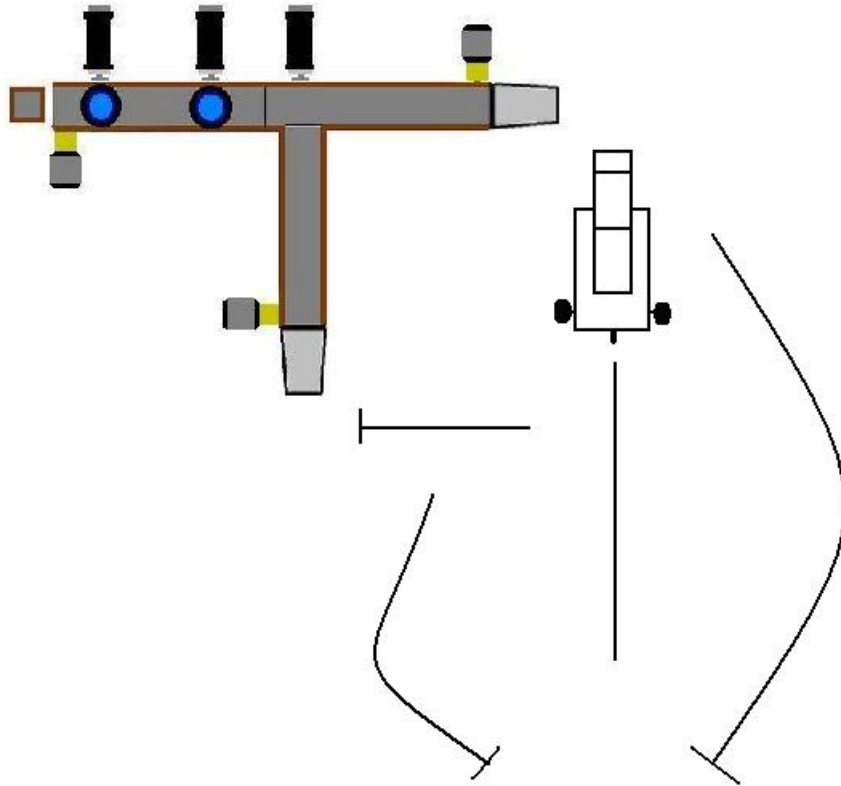


Fig. 3.4, Posición del producto uno.



Caso contrario, cuando el AGV se encuentra en la posición de carga del producto 1, el AGV gira hacia la izquierda hasta encontrar otra línea que lo lleva a la posición de entrega del producto 1 (Fig. 3.5).

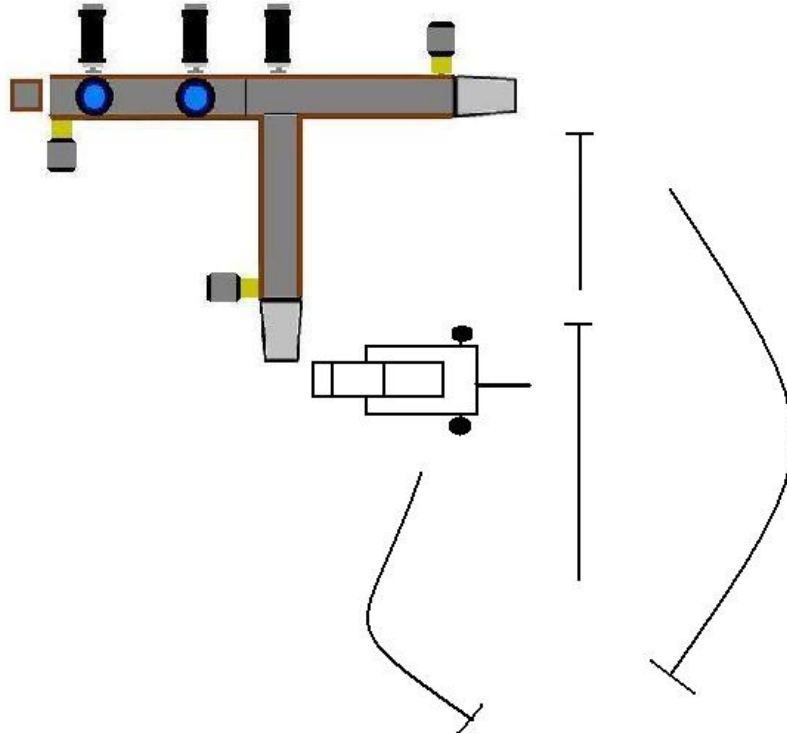


Fig 3.5, Posición del producto dos.

Finalmente, el diagrama siguiente esquematiza la configuración final de las trayectorias (Fig. 3.6).

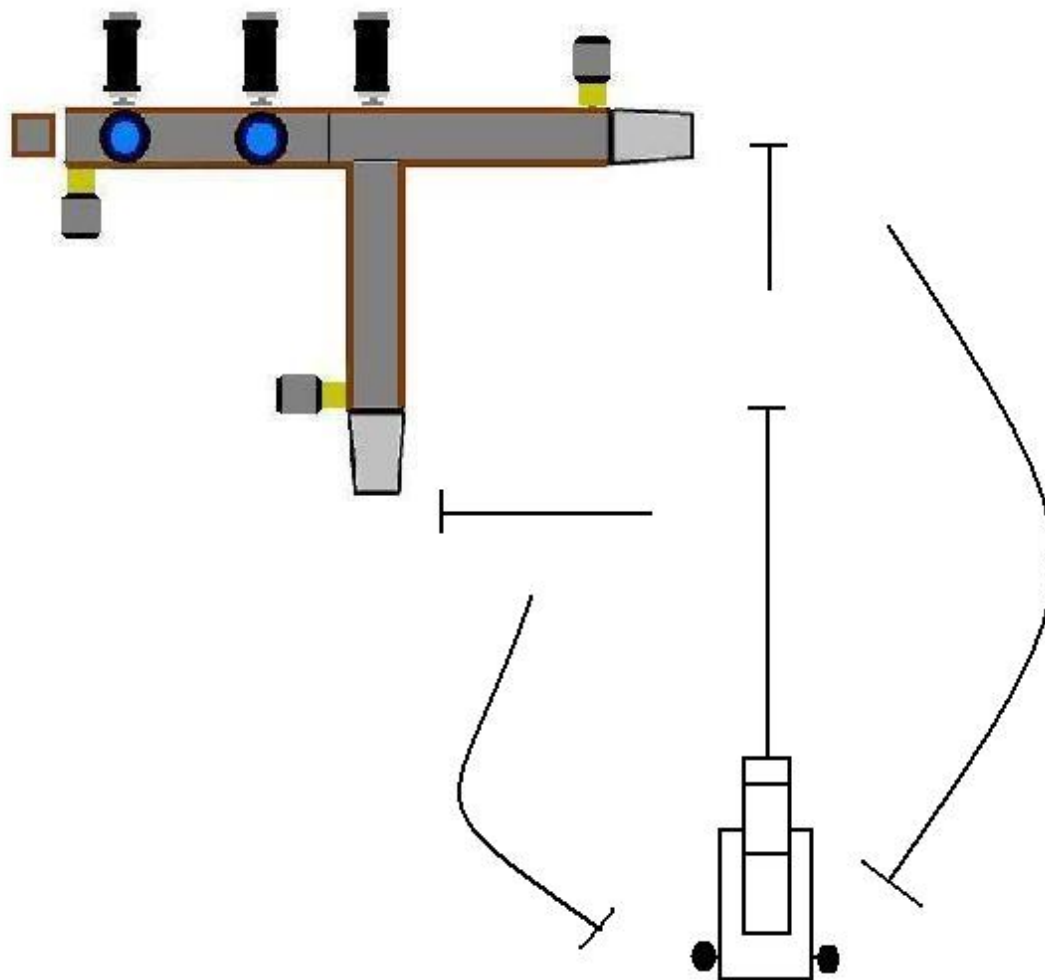
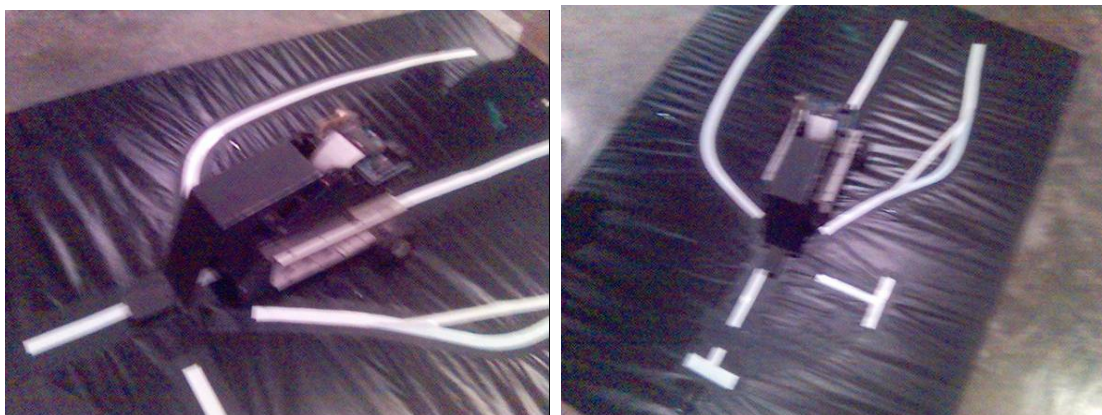


Fig. 3.6, Configuración final.





### 3.2 Toma de decisiones del AGV

En el subtema anterior se han establecido las posiciones en las que tomará las decisiones el AGV, una de ellas ocurre cuando ninguno de los tres sensores detecta la pista del AGV. Recordando lo mencionado en el primer capítulo (Diseño, elaboración y lógica de control del montacargas), en la sección que trata la lógica de movimiento, se mencionó que cuando los tres sensores no detectaban presencia de la ruta, los motores se detendrían por completo. Con esta información se debe diseñar un programa que comunique las decisiones al AGV dependiendo de la instrucción que se le haya asignado al inicio. Las instrucciones que envía el usuario serán muy básicas, y sólo hay dos tipos de ellas:

- Recoger el producto uno
- Recoger el producto dos.

La forma en la que se enviarán las instrucciones al microcontrolador es un tema que se tocará más adelante, por este momento se tratará la forma en la que el seguidor de línea interpreta las instrucciones que le son enviadas y la forma en la que éstas se ejecutan.

Como ya se había dicho, existen dos instrucciones que se le mandarán al microcontrolador para que sean ejecutadas, por lo cual se explicará a continuación cómo se interpreta la información dada a cada una de ellas.

En la localidad de memoria RUTA del microcontrolador se almacenará la ruta que ha de seguir el AGV para recoger alguna de las cajas de producto. La variable ruta puede tomar valores de 00000001, para la ruta 1, y 00000010, para la ruta 2.

Existe otra localidad de memoria del sistema que es de suma importancia, ésta es la variable ESTADO, en ella se guardará la posición del sistema en cada momento

de la ruta. La variable ESTADO puede adoptar 3 valores, cuando la variable ESTADO tiene el valor cero quiere decir que el vehículo se encontrará en su posición inicial, para el valor de uno se encontrará en el punto de decisión donde se elige la opción de la ruta a seguir y por último cuando el estado adquiere el valor de dos se encontrará con el producto levantado por el montacargas y listo para dirigirse a la entrega del mismo.

Este número de ruta lo enviará el usuario mediante la interfaz que se tiene con la computadora y el programa del microcontrolador sólo responderá al valor que se haya asignado en RUTA. Es importante mencionar que el vehículo sólo iniciará su movimiento cuando el usuario le envíe la ruta que debe de seguir y ésta se almacenará en la localidad de memoria RUTA. Por lo cual, la localidad de memoria RUTA tiene como valor inicial el número 0 binario; a menos que el valor de la localidad de memoria RUTA cambie, el seguidor de línea permanecerá en un *loop* infinito o ciclo infinito esperando una orden.

```

ESPERANDO      MOVLW    .0           ;VALOR INICIAL DE LA RUTA
                XORWF    RUTA, TRASH ;COMPARA EL VALORE DE LA RUTA
                BTFSC    STATUS,Z    ;YA SE HA ASIGNADO LA RUTA???
                GOTO     ESPERANDO   ;ESPERA RUTA

```

Existen dos puntos de decisión importantes para el seguidor de línea, el primero se encuentra en la ausencia de línea en la ruta principal. Para que este punto represente un estado de decisión se necesita cambiar la subrutina que es llamada al entrar en este punto.

```

MOVF           PORTB,W           ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA COMPARARLO
ANDWF          MASCARA,W         ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
MOVWF          TRASH             ;MUEVE EL VALOR AL LUGAR DE COMPARACION TRASH,
MOVLW          B'00000000'       ;MUEVE EL VALOR PARA SABER SI NO VE NINGUN SENSOR
XORWF          TRASH,F           ;COMPARA EL VALOR
BTFSC          STATUS,Z          ;NO ESTA VIENDO LA LINEA NINGUN SENSOR???
GOTO           DECISION          ;SI, VE A LA SUBROUTINA DECISIÓN

```

Ahora se cambia la rutina anterior llamada INDETERMINADO por la rutina de DECISIÓN. En este punto se debe de evitar el ruido ocasionado por la rapidez de

visión de los sensores, ya que podría ocurrir este estado por una milésima de segundo en un punto de recorrido diferente al que se desea representar. Por lo tanto, se debe tener la seguridad de que se encuentra en la posición correcta para tomar la decisión, de esta manera los sensores deben detectar dicho estado por lo menos durante 5 segundos para que pueda continuar ejecutando su rutina.

Para ello se crea una subrutina que proporciona un retardo de visión de 5 segundos en ella se utilizarán las localidades de memoria TPRES, TPRES1 Y TPRES2. Al momento de que los sensores detecten por sólo un instante un estado diferente al del punto de decisión, estas localidades se reiniciarán.<sup>1</sup>

Cuando ya se tiene la seguridad de que el seguidor de línea no ha detectado ninguna señal durante 5 segundos, se mandará la información que envió el usuario por medio del programa que asigna la rutina de cada producto. Esto se hace mediante una comparación entre la rutina almacenada en la variable RUTA y el valor 1.

```
MOVLW      .1           ;CARGA EL VALOR DE LA RUTA 1
XORWF      RUTA, TRASH  ;COMPARA EL VALORE DE LA RUTA
BTFS      STATUS,Z      ;MANDARON LA RUTA 1?
GOTO       RUTA2         ;NO, VE A RECORRER LA RUTA 2
GOTO       RUTA1         ;SI, RECORRE LA RUTA1
```

Cuando el usuario mandó la ruta 1 el seguidor de línea debe de girar hacia la derecha hasta encontrar una línea de centro en la cual se vuelve a comportar como seguidor de línea. Así comienza a girar el vehículo preguntando a cada instante si el sensor del centro ha detectado la pista; de no ser así continuará girando hasta encontrarla, cuando esto ocurra se va al programa principal.

```
RUTA1      BSF      ENABLED      ;ENCIENDE EL MOTOR DERECHA
           BSF      MOTORD       ;EL MOTORD VA HACIA ADELANTE
           ;AHORA VA A CHECAR SI YA LLEGO AL PUNTO DONDE VE EL
           ; DEL CENTRO
           MOVF     PORTB,W      ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
           ANDWF    MASCARA,W    ;COMPARARLO
           ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES QUE
```

<sup>1</sup> Código PIC16F877, Apéndice II.

```

; NECESITA
MOVWF TRASH ;MUEVE EL VALOR AL LUGAR DE COMPARACION,
MOVLW B'00000010' ;VALOR PARA SABER SI VE EL SENSOR DEL CENTRO
XORWF TRASH,F ;COMPARA EL VALOR
BTFS STATUS,Z ;ESTA VIENDO EL SENSOR DEL CENTRO???
GOTO RUTA1 ;NO, VE A RUTA1
;YA LLEGO A LA POSICION DONDE ESTA LA CAJA
GOTO PROGRAMA ;REGRESA AL PROGRAMA

```

En el caso de la RUTA 2 el vehículo comenzará a avanzar hacia adelante hasta encontrar la línea frente a él. En ambos casos se cambia el estado de la variable ESTADO de cero a uno.

```

RUTA2
BSF ENABLED ;ENCIENDE EL MOTOR DERECHA
BSF ENABLEI ;ENCIENDE EL MOTOR IZQUIERDA
BSF MOTORD ;EL MOTORD VA HACIA ADELANTE
BSF MOTORI ;EL MOTORI VA HACIA ADELANTE
;AHORA VA A CHECAR SI YA LLEGO AL PUNTO DONDE TODOS
;VEN
MOVF PORTB,W ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
;COMPARARLO
ANDWF MASCARA,W ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES QUE
NECESITO
MOVWF TRASH ;MUEVE EL VALOR AL LUGAR DE COMPARACION,
MOVLW B'00000111' ;VALOR PARA SABER SI SE VEN TODOS LOS SENSORES
XORWF TRASH,F ;COMPARA EL VALOR
BTFS STATUS,Z ;ESTA VIENDO TODOS???
GOTO RUTA2 ;NO, VE A RUTA2
;YA LLEGO A LA POSICION DONDE ESTA LA CAJA
GOTO PROGRAMA ;REGRESA AL PROGRAMA

```

El montacargas seguirá comportándose como seguidor de línea hasta encontrar el siguiente punto de decisión, que es cuando los tres sensores detectan la ruta al mismo tiempo. De la misma forma que en la primera decisión, se debe comprobar que no se está teniendo un estado erróneo y tener certeza de que se encuentra en dicha posición. Posteriormente se apagarán los motores y se comprobará que el valor de la variable ESTADO sea 1, de igual forma se esperará por lo menos 5 segundos para asegurar que se encuentra en este punto.

```

BCF ENABLED ;APAGA EL MOTOR DE LA DERECHA
BCF ENABLEI ;APAGA EL MOTOR DE LA IZQUIERDA
MOVLW .1 ;VALOR PARA SABER SI ESTAMOS EN EL ESTADO 1 DEL
;PROGRAMA
XORWF ESTADO,F ;COMPARA EL VALOR
BTFS STATUS,Z ;YA PASÓ EL ESTADO DE DECISION DE RUTA
GOTO PROGRAMA ;NO, VE Y SIGUE VIENDO LAS ENTRADAS

```

Acto seguido, levantará la pinza con la subrutina PWM descrita en el subtema “Lógica de movimiento” en el capítulo 1. En la rutina de PWM se cambia el valor de

ESTADO de 1 a 2 y se regresa a una subrutina dependiendo de la ruta que se debe de seguir para entregar el producto.

```

MOV LW .2          ;SE ENCUENTRA EN EL ESTADO 2
MOV WF ESTADO

;SUBROUTINA CREADA PARA QUE REGRESE A LA RUTA DE LA CUAL SE MANDO LLAMAR
MOV LW .1          ;CARGA EL VALOR DE LA RUTA 1
XORWF RUTA, TRASH  ;COMPARA EL VALOR DE LA RUTA
BTFS STATUS, Z     ;MANDARON LA RUTA 1???
GOTO RUTA2SIGUE    ;NO, VE A RECORRER LA RUTA 2
GOTO RUTA1SIGUE    ;SI, RECORRE LA RUTA 1

```

La subrutina PWM nos puede regresar a dos diferentes subrutinas que marcarán el regreso del montacargas: RUTA2SIGUE Y RUTA1SIGUE.

RUTA1SIGUE hace que el montacargas gire hacia la derecha hasta que encuentra la línea de centro.

```

RUTA1SIGUE  BSF    ENABLED    ;ENCIENDE EL MOTOR DERECHO
             BSF    MOTORD    ;MOTOR DE LA DERECHA HACIA ADELANTE. PARA QUE GIRE
             ; HASTA QUE ENCUENTRE OTRA VEZ LA LINEA
             MOVF   PORTB, W   ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
             ; COMPARARLO
             ANDWF  MASCARA, W ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES DE
             ;INTERÉS
             MOVWF  TRASH      ;MUEVE EL VALOR AL LUGAR DE COMPARACION,
             MOVLW  B'00000010' ;VALOR PARA SABER SI VE EL SENSOR DEL CENTRO
             XORWF  TRASH, F    ;COMPARA EL VALOR
             BTFS   STATUS, Z   ;ESTA VIENDO EL SENSOR DE EN MEDIO???
             GOTO   RUTA1SIGUE ;NO, VE A RUTA1SIGUE
             GOTO   PROGRAMA   ;SI, CONTINUA HACIENDO EL PROGRAMA DE SEGUIDOR DE
             ; LINEA.

```

RUTA2SIGUE hace que el montacargas gire hacia la izquierda hasta que encuentra la línea de centro.

```

RUTA2SIGUE  BSF    ENABLEI    ;ENCIENDE EL MOTOR IZQUIERDA
             BSF    MOTORI    ;MOTOR DE LA IZQUIERDA HACIA ADELANTE. PARA QUE GIRE
             ; HASTA QUE ENCUENTRE OTRA VEZ LA LINEA
             MOVF   PORTB, W   ;MUEVE EL VALOR DEL PUERTO AL ACUMULADOR PARA
             ; COMPARARLO
             ANDWF  MASCARA, W ;APLICA LA MASCARA, PARA SOLO RECIBIR LOS VALORES QUE
             ; NECESITO
             MOVWF  TRASH      ;MUEVE EL VALOR AL LUGAR DE COMPARACION,
             MOVLW  B'00000010' ;VALOR PARA SABER SI VE EL SENSOR DEL CENTRO
             XORWF  TRASH, F    ;COMPARA EL VALOR
             BTFS   STATUS, Z   ;ESTA VIENDO EL SENSOR DE EN MEDIO???
             GOTO   RUTA2SIGUE ;NO, VE A RUTA2SIGUE
             GOTO   PROGRAMA   ;SI, CONTINUA HACIENDO EL PROGRAMA DE SEGUIDOR DE
             ; LINEA.

```

Finalmente, con este procedimiento se asegurará que el vehículo llegará a la posición de entrega de producto.

En seguida se explicará el diseño de la interfaz que comunicará al usuario con el montacargas y con las bandas de distribución.

### 3.3 Interfaz con el usuario

La interfaz que comunicará al usuario con el montacargas es simple y de fácil funcionamiento. En el diseño de esta comunicación no se debe de hacer uso de objetos llamativos que distraigan la operación básica del sistema.

También se debe disminuir la interacción del usuario con el software, mientras el usuario tenga la menor participación con el software se evitarán validaciones y errores en la captura de datos. La interfaz se elaborará en Visual Basic 6 gracias a las facilidades que proporciona en el ambiente de trabajo al momento de ejecutar los programas que se diseñan en este software. En la página principal del programa se desplegarán las opciones de recolección de los productos, es decir se podrá elegir uno de los dos productos. (Fig. 3.7)



Fig. 3.7, Pantalla de presentación

Esta es la pantalla que tendrán a su disposición los operarios del montacargas, cuenta con sólo 4 botones, dos de los cuales son para la selección de los productos y los otros dos restantes funcionan para terminar el programa.

Lo primero que se debe de programar en el software es comunicación RS232 con el microcontrolador, para ello se tienen que configurar varios parámetros. Estos parámetros son configurados en la subrutina cuando se carga la forma.

```
Private Sub Form_Load()

'CONFIGURACIÓN DE PUERTO SERIAL DE COMUNICACIÓN
MSComm1.CommPort = 1           'COM2
MSComm1.Settings = "57600,N,8,1" '57600 BAUD, SIN PARIDAD, 8 BITS
MSComm1.RThreshold = 1         'INTERRUPCIÓN POR RECEPCIÓN
MSComm1.InputLen = 1           'DATOS A RECIBIR PARA INTERRUMPIR
If MSComm1.PortOpen = False Then MSComm1.PortOpen = True 'ABRE EL PUERTO SI ESTÁ CERRADO
MSComm1.Output = "0"          'ENVIAMOS UN DATO POR LA SALIDA
                                'PARA QUE EL MONTACARGAS ESPERE SEÑAL

End Sub
```

También se configura la velocidad de transferencia de información que por seguridad será a 57600 baudios, sin un bit de paridad y con comunicación de 8 bits. Se configura también la interrupción por recepción. Se realiza la apertura del puerto, que en este caso se configurará para la comunicación del serial número 2 y enviar el dato 0 que, como se vio en el subtema anterior, hace que el programa entre en un *loop* infinito esperando la indicación del producto que se desea trasladar.

Se debe programar la subrutina que ocurre cuando se da el evento clic a cualquiera de los dos botones de recolección de producto.

```
Private Sub ComProducto1_Click()
    MSComm1.Output = "1" 'ENVIA UN 1 AL MICROCONTROLADOR PARA QUE SE VAYA POR EL
PRODUCTO 1
    ComProducto1.Enabled = False
    ComProducto2.Enabled = False
End Sub

Private Sub ComProducto2_Click()
    MSComm1.Output = "2" 'ENVIA UN 2 AL MICROCONTROLADOR PARA QUE SE VAYA POR EL
PRODUCTO 2
    ComProducto1.Enabled = False
    ComProducto2.Enabled = False
End Sub
```

En ambos casos se envía un valor distinto al microcontrolador, uno para la subrutina uno y dos para la subrutina dos. En cada subrutina se inhabilita cada

uno de los botones de elección para que no puedan ser presionados después de haber sido seleccionado un producto. (Fig. 3.8)



Fig. 3.8, Pantalla al momento de la ejecución.

Ahora se debe programar el código para cuando el microcontrolador envíe un dato, en este caso sólo se activarán los botones para que el programa pueda ser ejecutado nuevamente.

```
Private Sub MSComm1_OnComm()  
If MSComm1.CommEvent = comEvReceive Then  
  
    ComProducto1.Enabled = True  
  
    ComProducto2.Enabled = True  
End If  
End Sub
```

'ROUTINA QUE PASA CUANDO EL MONTACARGAS  
'HA TERMINADO SU TAREA  
'PREGUNTA SI HAY INTERRUPCION POR  
'RECEPCION DE DATO, YA SABEMOS CON ESTO 'QUE  
EL MONTACARGAS TERMINÓ  
'HABILITAMOS LOS BOTONES PARA QUE SE  
'VUELVAN A MANDAR AL MONTACARGAS

Posteriormente se programa la rutina de desactivación con el puerto serial COM2 al momento de terminar el programa.

```
Private Sub Form_Unload(Cancel As Integer) 'CIERRA EL PUERTO AL SALIR DEL PROGRAMA  
    If MSComm1.PortOpen = True Then MSComm1.PortOpen = False  
End Sub
```

Ahora se explicará cómo se programa la comunicación del montacargas seguidor de línea con el software, primeramente se configurará la transmisión de datos:

```
BCF    TRISC,6    ;CONFIGURA COMO EMISOR  
BSF    TRISC,7    ;CONFIGURA COMO RECEPTOR  
MOVLW .20         ;MUEVE EL VALOR DE BAUDIOS  
MOVWF  SPBRG      ;ASIGNA EL VALOR DE COMUNICACION
```



```

MOVLW B'00000100' ;MUEVE CONFIGURACION DE COMUNICACION
MOVWF TXSTA ;ASIGNA CONFIGURACION COMUNICACION
BSF PIE1,RCIE ;DA EL PERMISO POR RECEPCION

```

Los pines 6 y 7 del puerto C deben de ser configurados como entrada y salida, ya que ellos se encargarán de llevar a cabo la comunicación RS-232. El pin 6 se configura como emisor y el pin 7 se configura como receptor. Debido a que se configurará la velocidad de emisión y recepción de datos a 57600 baudios se debe de configurar el registro SPBRG.

El Tasa de transferencia en baudios(*baud-rate*) es igual a:

$$\text{Tasa de transferencia} = \text{Frecuencia del oscilador} / 16 (\text{SPBRG} + 1)$$

Teniendo que la frecuencia del cristal es de 20 000 000 [Hz] se tiene como resultado:

$$\text{Tasa de transferencia} = 20\,000\,000 / 16 (20 + 1) = 59523$$

Dicho resultado es mucho mayor al deseado, pero si se elige que la localidad de memoria SPBRG adquiera el valor 22 se obtendría una velocidad de comunicación menor a la requerida, preferentemente es mejor que la configuración exceda la velocidad de comunicación. También se asigna el permiso a la interrupción por recepción de datos, para el caso de la transmisión no es necesario porque sólo se enviará un dato cuando el montacargas termine su rutina y no se necesita que interrumpa una rutina.

Ahora configuramos la recepción de datos:

```

MOVLW B'10010000' ;MUEVE VALOR DE RECEPCION
MOVWF RCSTA ;MUEVE VALOR DE COMUNICACION
BCF PIR1,TXIF ;HABILITA BANDERA DE RECEPCION
BCF PIR1,RCIF ;HABILITA BANDERA DE TRANSMISION

```

Se mueven las configuraciones para la recepción y se apagan las banderas de envío y recepción de datos. Ahora lo único que espera el seguidor de línea es la indicación para que realice su movimiento, lo cual ocurrirá cuando se mande un

dato de la computadora. Cuando el dato sea enviado se entrará a la interrupción, pero se debe diferenciar cada tipo de interrupción, puesto que se tiene la interrupción por PWM y también la interrupción por recepción de datos:

```
BTFS    INTCON,T0IF      ;INTERRUMPIO POR PWM
        GOTO    RECEP    ;NO, FUE POR RECEPCION
        GOTO    PWM      ;SI, VE A LA SUBROUTINA DE PWM

RECEP    MOVF     RCREG,W  ;MUEVE EL VALOR QUE ME MANDARON
        MOVWF    RUTA     ;ASIGNE EL VALOR QUE ME MANDARON DE VISUAL EN LA VARIABLE DE
                          ; RUTA
        BCF      PIR1,RCIF ;REPONE BANDERA POR RECEPCION
```

Por último, con esta subrutina se logra discriminar entre una interrupción por recepción de datos, para que vaya a la subrutina en la cual asigna el valor mandado de la ruta (ya sea ruta 1 o ruta 2); o si fue interrupción por necesidad de ejecutar el PWM.

Nota: En el apéndice III puede consultar la programación completa de la interfaz, así como la explicación de instrucciones utilizadas en Visual Basic.

## **Fabricación y pruebas**

El algoritmo de distribución está planteado de la siguiente manera: inicialmente, se propone que el usuario tenga la necesidad de algún producto que el sistema de distribución le pueda proporcionar, de esta manera, el montacargas autoguiado será diseñado para cubrir esta necesidad, consecuentemente será quien facilite al usuario los productos, llevándolos desde la banda de distribución hasta la ubicación del usuario, que en este caso es la ubicación inicial del montacargas.

Una vez que el usuario haya ordenado el producto con la computadora, ésta comunicará a las bandas de distribución qué producto deberán despachar, las bandas lo guiarán a una zona donde el montacargas pueda recogerlo. El montacargas también estará en comunicación con la computadora para saber a cuál zona debe presentarse a recoger el producto. Después que se haya levantado el producto, el montacargas lo llevará a donde se encuentra el usuario, es decir, su posición inicial.

Antes y durante el desarrollo de la tesis surgieron diversos problemas, algunos de ellos fueron considerados antes de que sucedieran, aunque algunos otros se presentaron y se resolvieron sobre la marcha, de los cuales los más relevantes son los que se presentarán a continuación:

El AGV se desplazaba a una velocidad elevada lo cual provocaba que los sensores no detectaran con certeza en qué parte de la línea se ubicaban, por lo tanto no enviaban la señal correcta, teniendo como consecuencia que la respuesta al error fuera muy lenta. Para resolver esto, se implementó un circuito regulador de voltaje, con el cual se logró disminuir la corriente que se le suministra a los motores de tracción, sin embargo se disminuyó el par de las llantas y el montacargas no podía vencer su propia inercia para avanzar.

Como segunda solución a este problema, se acoplaron a los motores un sistema de engranes que redujera la velocidad de las llantas. Con ello se disminuyó la velocidad y se incrementó el par a las llantas, de esta forma, el AGV puede vencer su inercia y desplazarse de forma correcta.

Desafortunadamente, al aumentar el par de tracción, se incrementó el consumo de corriente y disminuyó el tiempo de uso de las baterías que alimentan a los motores, la única alternativa a esto es la de recargar la baterías con mayor frecuencia. No se encontró otra alternativa a este problema, porque las pilas con las que se cuenta son de medidas exactas para el prototipo y unas baterías más grandes alterarían la configuración total del montacargas; por otra parte, el costo de baterías más chicas es demasiado alto. En la industria, este no es un problema de relevancia, ya por el tamaño y funciones de los montacargas reales se utilizan baterías de mayor potencia, o bien, se utilizan montacargas con motor de combustible.

Un problema que se presentó durante la fabricación de las bandas de distribución, fueron los acoplamientos de los motores a los rodillos que soportan a las bandas. Como primera instancia se maquinaron en el torno acoplamientos de aluminio que se ajustaban a la flecha y al rodillo con tornillos a manera de presión, sin embargo, estos acoplamientos no eran completamente colineales y, debido a su rigidez, provocaban vibraciones en las bandas. Como alternativa se consiguieron mangueras industriales con un diámetro interior igual al diámetro de las flechas y los rodillos, y así se consiguió fabricar acoplamientos flexibles.

Otro factor importante dentro de la fabricación de todo proyecto en general fue conseguir los materiales, herramientas y componentes para su elaboración. El hecho de necesitar materiales de diferentes propiedades y componentes difíciles de conseguir, ya que en su mayoría son usados únicamente en la industria, como son los pistones, los motores, las bandas y los soportes de todos ellos, se tuvo que

echar mano de lo que se pudiera conseguir en diferentes lugares, desde establecimientos formales para los materiales hasta “tianguis y tiraderos” para los componentes, y todo esto adaptarlo de la mejor forma para satisfacer las necesidades del proyecto, como fueron: la rigidez de la maqueta, el tamaño de la misma, como también el tamaño del montacargas. Para conseguir esto, también se tuvo que recurrir a la compra de material de prueba, con el que se fabricaron diseños que no resultaron factibles, se mandaron a construir algunas piezas de carpintería, que, por falta de comunicación y entendimiento con las personas que las elaboraban no se consiguió que cumplieran la función deseada, por este motivo, finalmente se prefirió hacer todo el proyecto personalmente adaptando los elementos conseguidos de la mejor manera posible.

Lo anterior da pie a otro detalle determinante del proyecto: el tamaño del mismo. El tener un prototipo a escala involucra buscar tolerancias muy altas, tolerancias que no pueden ser alcanzadas fácilmente por el hecho de que se tuvo que adaptar el proyecto a los componentes encontrados y que por sus dimensiones se no se tiene un buen manejo de los productos.

Finalmente, el problema más grave que se tiene es el que se refiere al sensado, ni los comparadores ni los sensores no brindan la lectura correcta, ocasionando que en muchas ocasiones el montacargas se desempeñe de forma errónea. Este problema es solucionado con una perfecta calibración con respecto a la luz del ambiente en el que se trabaje, desafortunadamente para estos fines, se trabaja a diversas horas del día, y por lo tanto, transcurrido algún tiempo, el sistema de sensado incrementaba el error paulatinamente. Al mismo tiempo, la calibración demanda mucho tiempo y por las razones expuestas la única solución encontrada es la de ajustar constantemente las impedancias que en ella intervienen, esto sin contar con la garantía de su buen funcionamiento al paso de unos momentos. Naturalmente la solución ideal es instalar mejores sensores, y ajustarlos digitalmente, sin embargo para el caso de este prototipo, el implementar

dispositivos de alta sensibilidad resultaría muy costoso. Afortunadamente, esto es algo que a nivel industrial no sólo es posible, sino que es prioridad; contar con dispositivos confiables tendría como consecuencia un buen desempeño y proveerían de seguridad en la ejecución de las tareas a realizar. Por estos motivos, implementar estos elementos resultaría costeable en sistemas reales.

## Conclusiones y comentarios

Una vez finalizada esta tesis, se han conseguido los siguientes objetivos:

- Se ha diseñado y construido un prototipo a escala de un montacargas industrial autoguiado, el cual recoge productos ubicados en dos líneas de distribución para luego trasladarlos a una zona en específico.
- De igual forma, se ha diseñado y construido una línea de distribución a escala, la cual es capaz de despachar dos tipos de productos diferentes, para luego ser recolectados por el montacargas autoguiado.
- Así mismo se cumplió con el diseño, programación e implementación de un software que funciona de interfaz entre el proceso y el usuario. Esta interfaz le da la facilidad de al usuario de seleccionar el tipo de producto que el montacargas debe de recoger, el cual previamente ya ha sido despachado por el sistema de bandas de distribución.

El desarrollo del montacargas que se presentó en los capítulos anteriores se compone de un sistema de desplazamiento AGV de tracción de velocidad diferenciado, por lo que utilizará dos impulsores, cada uno fijado a una rueda trasera y así podrá variar su velocidad el impulsor de cualquier lado de la trayectoria de la guía para permitir que negocie las vueltas. Dicha tracción será controlada *onboard*, lo cual permitirá al montacargas seguir la trayectoria de línea mediante un microprocesador montado en su parte trasera. Por otra parte contará con una terminal remota, en este caso una computadora comunicada alámbricamente, con la que por medio de un sistema amigable de programación el operario elegirá y ordenará cual de dos productos despachados, por bandas de distribución, deberá recoger. Así mismo, el montacargas contará con una plataforma de levantamiento accionada por un motor acoplado a un mecanismo de cuatro barras, gracias a este sistema podrá levantar y, sobre la plataforma, trasladar paquetes de dos tipos de productos, los productos deberán estar empaquetados. El volumen de cada paquete deberá ser de 4[cm]x4[cm]x4[cm], para facilitar su carga y transportación.

Este trabajo nació de la necesidad de buscar soluciones a algunos de los problemas más comunes presentados en la industria y ahora que se ha llegado al final del mismo, se puede afirmar que se ha presentado una alternativa segura para el usuario, eficaz con respecto al empaquetamiento de los productos y eficiente en el traslado de los mismos.

Al finalizar esta tesis se cumplió con el diseño y fabricación de un prototipo a escala para empaquetamiento, selección y traslado de diferentes productos en una planta industrial, el cual cumple con sus propósitos principales, tiene ventajas y desventajas como cualquier proyecto que ha de ser desarrollado desde el inicio, y que necesita mejora continua para garantizar su funcionalidad y viabilidad, para dejar de ser un prototipo y convertirse en un producto factible para su uso diario. Es importante mencionar que la investigación y desarrollo de todas estas tecnologías, deberán tomar en cuenta cada vez más las necesidades del sector industrial, como son el cálculo de inventarios, capacidad de almacenamiento, requerimiento de productos por mencionar algunos. Una alternativa que podría ser factible es la de realizar algún tipo de comunicación que permita la transferencia de información con el sistema de distribución y traslado con la base de datos de los productos dentro de la fábrica y así tener un mayor control del producto para reducir los costos que el manejo y almacenamiento puedan involucrar.

El presente proyecto se desarrolló con la interacción de varios subsistemas: mecánicos, electroneumáticos, electrónicos, informáticos y de control, que, al unirse y complementarse unos con otros, de forma particular, enriquecen el proyecto y de forma general ejemplifican de manera adecuada la finalidad de la Mecatrónica.

En nuestro trabajo de tesis hemos aplicado gran parte de los conocimientos adquiridos durante la carrera universitaria y nos encontramos muy orgullosos por haber desarrollado un proyecto tecnológico que nos ha de dar el grado de ingenieros ante la sociedad.



## **Bibliografía**

CAMERON Stephen, PROBERT Penelope  
Advanced Guided Vehicles  
World Scientific Series in Robotics and Automated Systems. Vol. 9  
1994

OLLERO BATURONE, Aníbal  
ROBÓTICA, Manipuladores y Robots Móviles  
Boixareu Editores  
2001

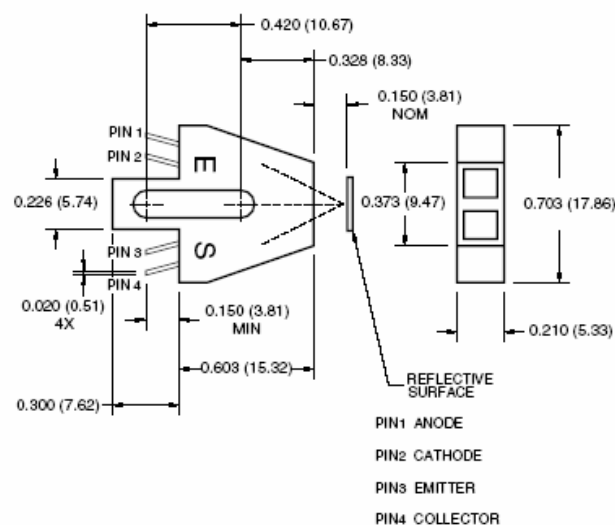
[www.agvsystems.com](http://www.agvsystems.com) (fecha de consulta: septiembre de 2005)



# PHOTOTRANSISTOR REFLECTIVE OBJECT SENSOR

QRB1113 QRB1114

## PACKAGE DIMENSIONS

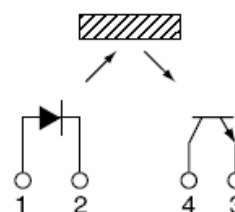


### NOTES:

1. Dimensions for all drawings are in inches (mm).
2. Tolerance of  $\pm .010$  (.25) on all non-nominal dimensions unless otherwise specified.



## SCHEMATIC



## DESCRIPTION

The QRB1113/1114 consists of an infrared emitting diode and an NPN silicon phototransistor mounted side by side on a converging optical axis in a black plastic housing. The phototransistor responds to radiation from the emitting diode only when a reflective object passes within its field of view. The area of the optimum response approximates a circle .200" in diameter.

## FEATURES

- No contact surface sensing
- Phototransistor output
- Focused for sensing specular reflection
- Daylight filter on photosensor
- Dust cover



# LM139,A LM239,A-LM339,A

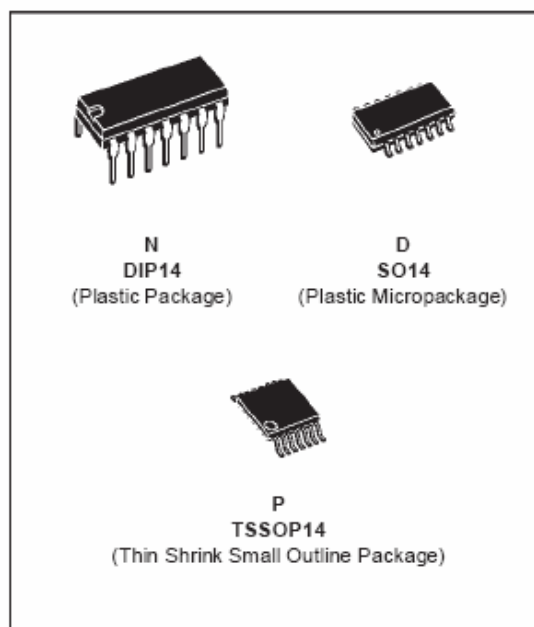
## LOW POWER QUAD VOLTAGE COMPARATORS

- WIDE SINGLE SUPPLY VOLTAGE RANGE OR DUAL SUPPLIES FOR ALL DEVICES : +2V TO +36V OR  $\pm 1$ V TO  $\pm 18$ V
- VERY LOW SUPPLY CURRENT (1.1mA) INDEPENDENT OF SUPPLY VOLTAGE (1.4mW/comparator at +5V)
- LOW INPUT BIAS CURRENT : 25nA TYP
- LOW INPUT OFFSET CURRENT :  $\pm 5$ nA TYP
- LOW INPUT OFFSET VOLTAGE :  $\pm 1$ mV TYP
- INPUT COMMON-MODE VOLTAGE RANGE INCLUDES GROUND
- LOW OUTPUT SATURATION VOLTAGE : 250mV TYP. ( $I_o = 4$ mA)
- DIFFERENTIAL INPUT VOLTAGE RANGE EQUAL TO THE SUPPLY VOLTAGE
- TTL, DTL, ECL, MOS, CMOS COMPATIBLE OUTPUTS

### DESCRIPTION

These devices consist of four independent precision voltage comparators with an offset voltage specifications as low as 2mV max for LM339A, LM239A and LM139A. All these comparators were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible.

These comparators also have a unique characteristic in that the input common-mode voltage range includes ground even though operated from a single power supply voltage.

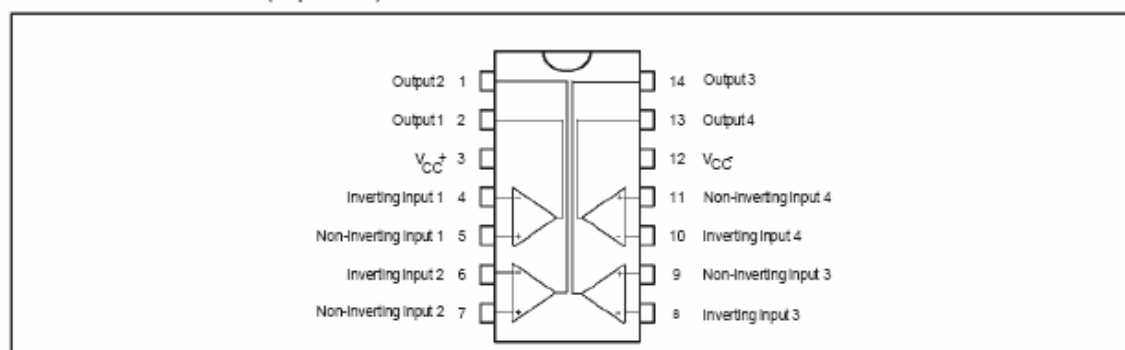


### ORDER CODES

| Part Number | Temperature Range | Package |   |   |
|-------------|-------------------|---------|---|---|
|             |                   | N       | D | P |
| LM139,A     | -55, +125°C       | •       | • | • |
| LM239,A     | -40, +105°C       | •       | • | • |
| LM339,A     | 0, +70°C          | •       | • | • |

Example : LM139AN

### PIN CONNECTIONS (top view)



SDLS049

**SN5414, SN54LS14,  
SN7414, SN74LS14**  
**HEX SCHMITT-TRIGGER INVERTERS**  
DECEMBER 1983—REVISED MARCH 1988

- Operation from Very Slow Edges
- Improved Line-Receiving Characteristics
- High Noise Immunity

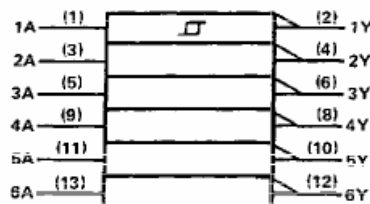
**description**

Each circuit functions as an inverter, but because of the Schmitt action, it has different input threshold levels for positive ( $V_{T+}$ ) and for negative going ( $V_{T-}$ ) signals.

These circuits are temperature-compensated and can be triggered from the slowest of input ramps and still give clean, jitter-free output signals.

The SN5414 and SN54LS14 are characterized for operation over the full military temperature range of  $-55^{\circ}\text{C}$  to  $125^{\circ}\text{C}$ . The SN7414 and the SN74LS14 are characterized for operation from  $0^{\circ}\text{C}$  to  $70^{\circ}\text{C}$ .

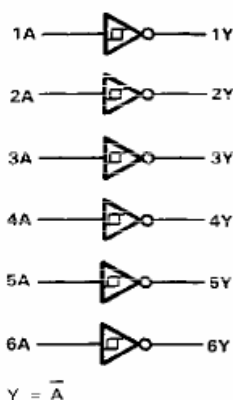
**logic symbol†**



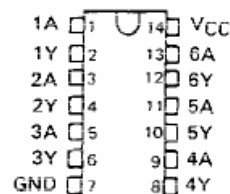
† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

Pin numbers shown are for D, J, N, and W packages.

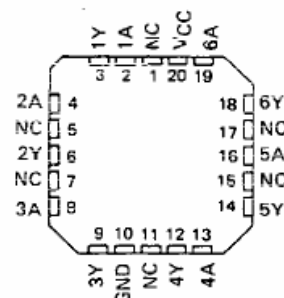
**logic diagram (positive logic)**



SN5414, SN54LS14 . . . J OR W PACKAGE  
SN7414 . . . N PACKAGE  
SN74LS14 . . . D OR N PACKAGE  
(TOP VIEW)



SN54LS14 . . . FK PACKAGE  
(TOP VIEW)



NC—No internal connection

PRODUCTION DATA documents contain information current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

**TEXAS  
INSTRUMENTS**  
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265



June 1989

## 54LS04/DM54LS04/DM74LS04 Hex Inverting Gates

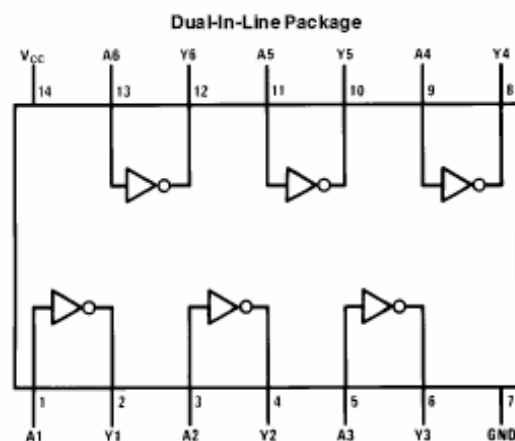
### General Description

This device contains six independent gates each of which performs the logic INVERT function.

### Features

- Alternate Military/Aerospace device (54LS04) is available. Contact a National Semiconductor Sales Office/Distributor for specifications.

### Connection Diagram



TL/F/6345-1

Order Number 54LS04DMQB, 54LS04FMB, 54LS04LMB, DM54LS04J, DM54LS04W, DM74LS04M or DM74LS04N  
See NS Package Number E20A, J14A, M14A, N14A or W14B

### Function Table

$$Y = \bar{A}$$

| Input | Output |
|-------|--------|
| A     | Y      |
| L     | H      |
| H     | L      |

H = High Logic Level

L = Low Logic Level

54LS04/DM54LS04/DM74LS04 Hex Inverting Gates

# L293 QUADRUPLE HALF-H DRIVER

SLRS005 – SEPTEMBER 1986 – REVISED MAY 1990

- 1-A Output Current Capability Per Driver
- Pulsed Current 2-A Driver
- Wide Supply Voltage Range: 4.5 V to 36 V
- Separate Input-Logic Supply
- NE Package Designed for Heat Sinking
- Thermal Shutdown
- Internal ESD Protection
- High-Noise-Immunity Inputs
- Functional Replacement for SGS L293

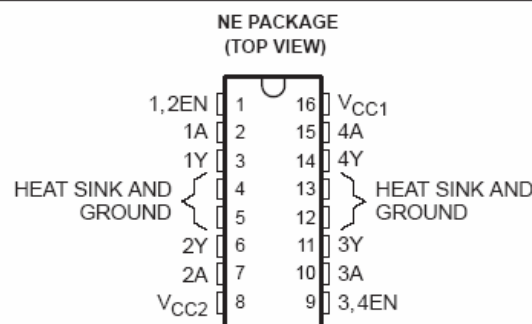
## description

The L293 is a quadruple high-current half-H driver designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. It is designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

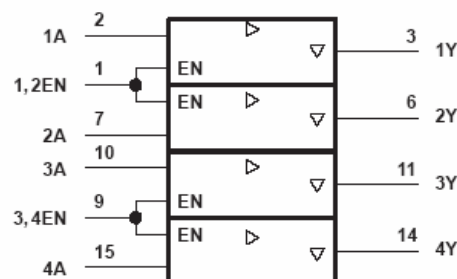
All inputs are TTL compatible. Each output is a complete totem-pole drive circuit with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled and their outputs are off and in a high-impedance state. With the proper data inputs, each pair of drivers form a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

External high-speed output clamp diodes should be used for inductive transient suppression. A  $V_{CC1}$  terminal, separate from  $V_{CC2}$ , is provided for the logic inputs to minimize device power dissipation.

The L293 is designed for operation from 0°C to 70°C.

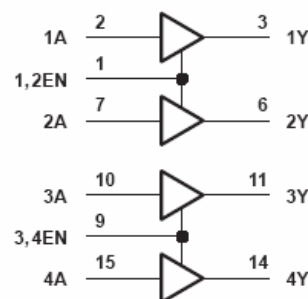


## logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC publication 617-12.

## logic diagram



FUNCTION TABLE  
(each driver)

| INPUTS‡ |    | OUTPUT<br>Y |
|---------|----|-------------|
| A       | EN |             |
| H       | H  | H           |
| L       | H  | L           |
| X       | L  | Z           |

H = high-level, L = low-level,  
X = irrelevant, Z = high-impedance (off)  
‡ In the thermal shutdown mode, the output is in the high-impedance state regardless of the input levels.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

TEXAS  
INSTRUMENTS

POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

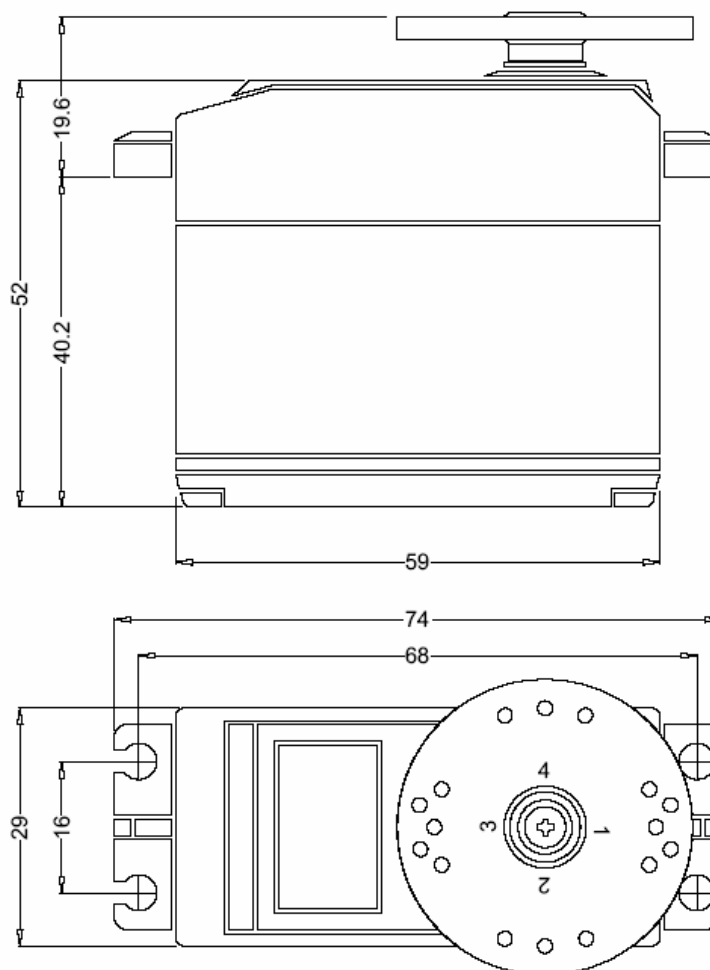
Copyright © 1990, Texas Instruments Incorporated

3-1

## ANNOUNCED SPECIFICATION OF HS-700BB 1/4 SCALE SERVO

### 1. TECHNICAL VALUES

|                             |                                               |                        |
|-----------------------------|-----------------------------------------------|------------------------|
| CONTROL SYSTEM              | : +PULSE WIDTH CONTROL 1500usec NEUTRAL       |                        |
| OPERATING VOLTAGE RANGE     | : 4.8V TO 6.0V                                |                        |
| OPERATING TEMPERATURE RANGE | : -20 TO +60°C                                |                        |
| TEST VOLTAGE                | : AT 4.8V                                     | AT 6.0V                |
| OPERATING SPEED             | : 0.22sec/60° AT NO LOAD                      | 0.17sec/60° AT NO LOAD |
| STALL TORQUE                | : 10kg.cm(138.87oz.in)                        | 12.5kg.cm(173.58oz.in) |
| OPERATING ANGLE             | : 45°/ONE SIDE PULSE TRAVELING 400usec        |                        |
| DIRECTION                   | : CLOCK WISE/PULSE TRAVELING 1500 TO 1900usec |                        |
| CURRENT DRAIN               | : 8mA/IDLE AND 230mA/NO LOAD RUNNING          |                        |
| DEAD BAND WIDTH             | : 8usec                                       |                        |
| CONNECTOR WIRE LENGTH       | : 300mm(11.81in)                              |                        |
| DIMENSIONS                  | : 59x29x52mm(2.32x1.14x2.04in)                |                        |
| WEIGHT                      | : 102g(3.59oz)                                |                        |



### 2. FEATURES

3-POLE FERRITE MOTOR  
LONG LIFE POTENTIOMETER  
1-OILITE BUSHING & 1-BALL BEARING  
WATER & DUST TIGHT  
INDIRECT POTENTIOMETER DRIVE

### 3. APPLICATIONS

GIANT SCALE AIRCRAFT  
1/4 SCALE CARS  
LARGE SCALE BOATS

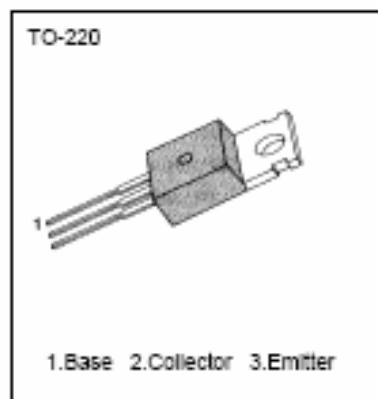
# TIP32 SERIES (TIP32/32A/32B/32C) PNP EPITAXIAL SILICON TRANSISTOR

## MEDIUM POWER LINEAR SWITCHING APPLICATIONS

\* Complement to TIP31/31A/31B/31C

## ABSOLUTE MAXIMUM RATINGS

| Characteristic                                     | Symbol           | Rating    | Unit               |
|----------------------------------------------------|------------------|-----------|--------------------|
| Collector Base Voltage : TIP32                     | $V_{CB0}$        | -40       | V                  |
| : TIP32A                                           |                  | -60       | V                  |
| : TIP32B                                           |                  | -80       | V                  |
| : TIP32C                                           |                  | -100      | V                  |
| Collector Emitter Voltage : TIP32                  | $V_{CE0}$        | -40       | V                  |
| : TIP32A                                           |                  | -60       | V                  |
| : TIP32B                                           |                  | -80       | V                  |
| : TIP32C                                           |                  | -100      | V                  |
| Emitter-Base Voltage                               | $V_{EB0}$        | -5        | V                  |
| Collector Current (DC)                             | $I_C$            | -3        | A                  |
| Collector Current (Pulse)                          | $I_C$            | -5        | A                  |
| Base Current                                       | $I_B$            | -3        | A                  |
| Collector Dissipation ( $T_C=25^{\circ}\text{C}$ ) | $P_C$            | 40        | W                  |
| Collector Dissipation ( $T_A=25^{\circ}\text{C}$ ) | $P_C$            | 2         | W                  |
| Junction Temperature                               | $T_J$            | 150       | $^{\circ}\text{C}$ |
| Storage Temperature                                | $T_{\text{stg}}$ | -65 ~ 150 | $^{\circ}\text{C}$ |



## ELECTRICAL CHARACTERISTICS ( $T_C=25^{\circ}\text{C}$ )

| Characteristic                                | Symbol          | Test Conditions                                                  | Min  | Max  | Unit          |
|-----------------------------------------------|-----------------|------------------------------------------------------------------|------|------|---------------|
| *Collector Emitter Sustaining Voltage : TIP32 | $BV_{CE0(sus)}$ | $I_C = -30\text{mA}, I_B = 0$                                    | -40  |      | V             |
| : TIP32A                                      |                 |                                                                  | -60  |      | V             |
| : TIP32B                                      |                 |                                                                  | -80  |      | V             |
| : TIP32C                                      |                 |                                                                  | -100 |      | V             |
| Collector Cutoff Current : TIP32/32A          | $I_{CBO}$       | $V_{CE} = -30\text{V}, I_B = 0$                                  |      | -0.3 | mA            |
| : TIP32B/32C                                  |                 | $V_{CE} = -60\text{V}, I_B = 0$                                  |      | -0.3 | mA            |
| Collector Cutoff Current : TIP32              | $I_{CE0}$       | $V_{CE} = -40\text{V}, V_{EB} = 0$                               |      | -200 | $\mu\text{A}$ |
| : TIP32A                                      |                 | $V_{CE} = -60\text{V}, V_{EB} = 0$                               |      | -200 | $\mu\text{A}$ |
| : TIP32B                                      |                 | $V_{CE} = -80\text{V}, V_{EB} = 0$                               |      | -200 | $\mu\text{A}$ |
| : TIP32C                                      |                 | $V_{CE} = -100\text{V}, V_{EB} = 0$                              |      | -200 | $\mu\text{A}$ |
| Emitter Cutoff Current                        | $I_{EBO}$       | $V_{EB} = -5\text{V}, I_C = 0$                                   |      | -1   | mA            |
| *DC Current Gain                              | $\beta_{DC}$    | $V_{CE} = -4\text{V}, I_C = -1\text{A}$                          | 25   |      |               |
|                                               |                 | $V_{CE} = -4\text{V}, I_C = -3\text{A}$                          | 10   | 50   |               |
| *Collector Emitter Saturation Voltage         | $V_{CE(sat)}$   | $I_C = -3\text{A}, I_B = -375\text{mA}$                          |      | -1.2 | V             |
| *Base Emitter On Voltage                      | $V_{BE(on)}$    | $V_{CE} = -4\text{V}, I_C = -3\text{A}$                          |      | -1.8 | V             |
| Current Gain Bandwidth Product                | $f_T$           | $V_{CE} = -10\text{V}, I_C = -500\text{mA}$<br>$f = 1\text{MHz}$ | 3.0  |      | MHz           |

\* Pulse Test:  $PW \leq 300\mu\text{s}$ , duty Cycle  $\leq 2\%$



# RAS SERIES

## ORDER CODE

RAS-12-10-A

□: 1c  
A: 1a  
B: 1b  
M: Automobile Relay

10: 10A  
15: 15A  
20: 20A (M type)

COIL VOLTAGE  
03: DC 3V  
05: DC 5V  
06: DC 6V  
09: DC 9V  
12: DC 12V  
18: DC 18V  
24: DC 24V  
36: DC 36V  
48: DC 48V

TYPE



### Main Features:

- UL, CUR & TUV safety approval.
- Heavy current up to 15 amps available, and special 20amps for using on the car.
- Epoxy seal type and flux free.
- Satisfying all requirements for use in car and household electric appliances.

## COIL RATING (at 20°C)

| NOMINAL VOLTAGE (VDC) | COIL RESISTANCE ( $\Omega$ ) (+/- 10%) | POWER CONSUMPTION (W) | NOMINAL CURRENT (mA) (+/- 10%) | PULL IN VOLTAGE (VDC) | DROP OUT VOLTAGE (VDC) | MAX. ALLOWABLE VOLTAGE (VDC) |
|-----------------------|----------------------------------------|-----------------------|--------------------------------|-----------------------|------------------------|------------------------------|
| 3V                    | 25 $\Omega$                            | 0.36W                 | 120 mA                         | 75 % MAX.             | 10 % MIN.              | 130 %                        |
| 5V                    | 69 $\Omega$                            |                       | 72 mA                          |                       |                        |                              |
| 6V                    | 100 $\Omega$                           |                       | 60 mA                          |                       |                        |                              |
| 9V                    | 225 $\Omega$                           |                       | 40 mA                          |                       |                        |                              |
| 12V                   | 400 $\Omega$                           |                       | 30 mA                          |                       |                        |                              |
| 18V                   | 900 $\Omega$                           |                       | 20 mA                          |                       |                        |                              |
| 24V                   | 1600 $\Omega$                          |                       | 15 mA                          |                       |                        |                              |
| 36V                   | 3600 $\Omega$                          |                       | 10 mA                          |                       |                        |                              |
| 48V                   | 4500 $\Omega$                          | 0.51W                 | 10.6 mA                        | 65 %                  | 8 %                    | 110 %                        |
| 12V (M type)          | 225 $\Omega$                           | 0.64W                 | 53.3 mA                        |                       |                        |                              |

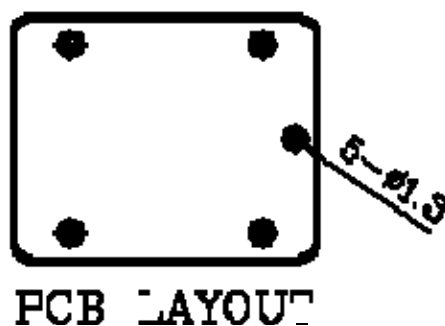
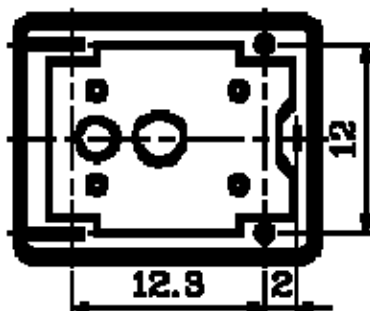
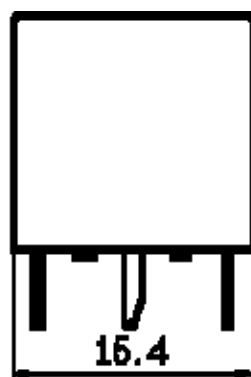
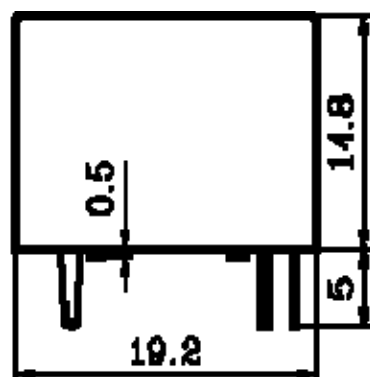
## PERFORMANCE (at initial value)

| Item                                      | Type | 10A                                                                   | 15A | 20 <sup>a</sup> |
|-------------------------------------------|------|-----------------------------------------------------------------------|-----|-----------------|
| Contact Resistance                        |      | 50m $\Omega$ Max. (initial value)                                     |     |                 |
| Operate Time                              |      | 10msec Max.                                                           |     |                 |
| Release Time                              |      | 5msec Max.                                                            |     |                 |
| Dielectric Strength                       |      | AC 1500V (1min)<br>AC 750V (1min)                                     |     |                 |
| between coil & contact<br>between contact |      |                                                                       |     |                 |
| Insulation Resistance                     |      | 100M $\Omega$ Min.(DC 500V)                                           |     |                 |
| Operating Ambient Temperature             |      | -30°C ~ +80°C                                                         |     |                 |
| Humidity                                  |      | 35 to 85% RH                                                          |     |                 |
| Vibration Resistance                      |      | 10G (10~55Hz)(Dual amplitude : 1.5mm)                                 |     |                 |
| Shock Resistance                          |      | 10G                                                                   |     |                 |
| Life Expectancy                           |      | 10,000,000 ops. Min. (1800 ops./h)<br>100,000 ops. Min. (1200 ops./h) |     |                 |
| Mechanically<br>Electrically              |      |                                                                       |     |                 |
| Weight                                    |      | 9.5g (about)                                                          |     |                 |

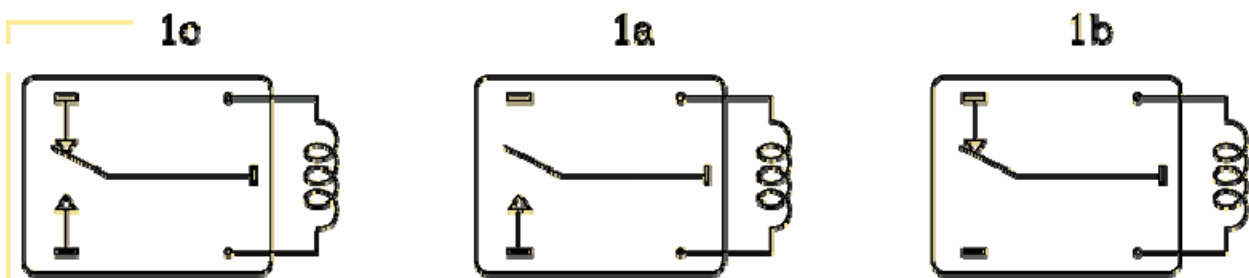
## CONTACT RATING

| Item                     | Type | 10A                                     | 15A                                      | 20 <sup>a</sup>                                     |
|--------------------------|------|-----------------------------------------|------------------------------------------|-----------------------------------------------------|
| Rated Carrying Current   |      | AC 120V 10A<br>AC 250V 7A<br>DC 24V 10A | AC 120V 15A<br>AC 250V 10A<br>DC 24V 15A | DC 14V 20 <sup>a</sup><br>DC 30V 15A<br>AC 120V 15A |
| Max. Allowable Current   |      | 15A                                     | 20A                                      | 20A                                                 |
| Max. Allowable Voltage   |      | AC 240V<br>DC 110V                      |                                          |                                                     |
| Max. Current (continual) |      | 10A                                     | 15A                                      | 20A                                                 |
| Contact Material         |      | Ag alloy                                |                                          |                                                     |

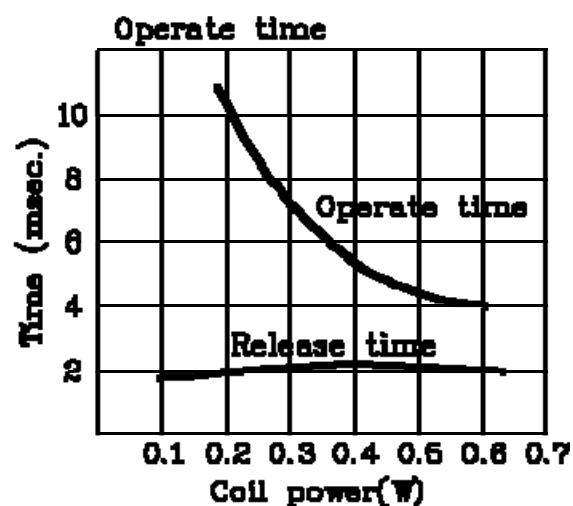
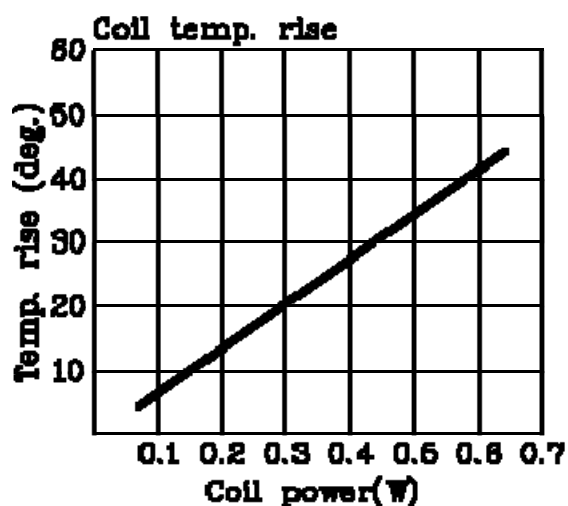
## DIMENSIONS (Unit : mm)



## WIRING DIAGRAMS (Bottom View)



## REFERENCE DATA



<http://www.sunhold.com/ras.html>

## Conjunto de Instrucciones

Cada instrucción del PIC16F877 es una palabra de 14 bits, dividida en un código de operación que especifica el tipo de instrucción y uno o más operandos que especifican aún mejor la operación de la instrucción. El conjunto de instrucciones del PIC16F877 se divide en operaciones orientadas a bytes, operaciones orientadas a bits y operaciones con literales y de control.

Para operaciones orientadas a bytes, 'f' representa el puntero al registro fuente o de archivo y 'd' representa el puntero al destino. El puntero al registro de archivo especifica que registro de archivo será utilizado en la instrucción.

El puntero del destino especifica la ubicación en donde el resultado de la operación será guardado. Si 'd' es cero, el resultado se guardará en el registro W. Si 'd' es uno, el resultado será guardado en el registro de archivo especificado en la instrucción.

Para operaciones orientadas a bits, 'b' representa el puntero al campo de bit que selecciona el número del bit afectado por la operación, mientras que 'f' representa la dirección de archivo en donde el bit está ubicado.

Para operaciones con literales y de control, 'k' representa una constante de ocho u once bits o un valor literal.

Todas las instrucciones son ejecutadas en un solo ciclo de instrucción, a menos que una prueba condicional resulte verdadera y el contador de programa sea cambiado como resultado de la instrucción. En este caso, la ejecución toma dos ciclos con un segundo ciclo ejecutado como un NOP. Un ciclo de instrucción consiste de cuatro periodos del oscilador.

Conjunto de instrucciones del PIC16F877

Operaciones orientadas a bytes

Para los ejemplos de las operaciones orientadas a bytes los estados iniciales de los registros se muestran en la siguiente tabla:

| Registro | Contenido |
|----------|-----------|
| PORTA    | 0xC2      |
| W        | 0x17      |
| Bandera  | Valor     |
| C        | 0         |

En todos estos ejemplos el formato 0xhh representa un número hexadecimal, en donde h significa un dígito hexadecimal.

- ADDWF** Suma W y f  
 Sintaxis: [etiqueta]ADDWF f, d  
 Operación:  $(W) + (f) \rightarrow \text{destino}$   
 Bits afectados en STATUS: C, DC, Z  
 Descripción: Suma los contenidos del registro W con el registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo.

```
ADDWF PORTA, 0 ; (W) ← (W) + (PORTA)
; W = 0xD9, ya que d = 0
; PORTA = 0xC2
```

- ANDWF** AND entre W y f  
 Sintaxis: [etiqueta]ANDWF f, d  
 Operación:  $(W) \text{ AND } (f) \rightarrow \text{destino}$   
 Bits afectados en STATUS: Z  
 Descripción: Operación lógica AND del registro W con el registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo.

```
ANDWF PORTA, 1 ; (PORTA) ← (W) AND (PORTA)
; W      = 0001 0111 (0x17)
; PORTA  = 1100 0010 (0xC2)
;      -----
; PORTA  = 0000 0010 (0x02), ya que d = 1
; W      = 0x17
```

- **CLRF** Limpia f
 

Sintaxis: [etiqueta]CLR f

Operación:  $00h \rightarrow f$   
 $1 \rightarrow Z$

Bits afectados en STATUS: Z

Descripción: El contenido del registro 'f' es limpiado y el bit Z se activa.

Ejemplo. `CLRF PORTA; (PORTA) ← 0x00`

- **CLRW** Limpia W
 

Sintaxis: [etiqueta]CLRW

Operación:  $00h \rightarrow W$   
 $1 \rightarrow Z$

Bits afectados en STATUS: Z

Descripción: El registro W es limpiado y el bit Z se activa.

Ejemplo. `CLRW; (W) ← 0x00`

- **COMF** Complemento de f
 

Sintaxis: [etiqueta]COMF f, d

Operación:  $(\bar{f}) \rightarrow \text{destino}$

Bits afectados en STATUS: Z

Descripción: Complemento a 1 del contenido del registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo. `COMF PORTA, 0`  
`;W = 0x3D (0011 1101), ya que d = 0`  
`;PORTA = 0xC2 (1100 0010)`

- **DECF** Decrementa f
 

Sintaxis: [etiqueta]DECF f, d

Operación:  $(f) - 1 \rightarrow \text{destino}$

Bits afectados en STATUS: Z

Descripción: Decremento del registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo. `DECF PORTA, 1; (PORTA) → (PORTA) - 1  
;PORTA = 0xC1, ya que d = 1`

- **DECFSZ** Decrementa f, salta si el resultado es 0

Sintaxis: [etiqueta]DECFSZ f, d

Operación:  $(f) - 1 \rightarrow \text{destino}$ ; salta si el resultado es 0

Bits afectados en STATUS:Ninguno

Descripción: El contenido del registro 'f' es decrementado. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'. Si el resultado es 0, entonces la siguiente instrucción es descartada.

Ejemplo. `AQUI DECFSZ PORTA, 1  
GOTO CICLO  
CONTINUA . . .  
;PORTA = 0xC1, como el resultado no es cero  
;continúa con la ejecución de la siguiente  
;instrucción que es el GOTO CICLO`

- **INCF** Incrementa f

Sintaxis: [etiqueta]INCF f, d

Operación:  $(f) + 1 \rightarrow \text{destino}$

Bits afectados en STATUS:Z

Descripción: El contenido del registro 'f' es incrementado. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo. `INCF PORTA, 0; (W) → (PORTA) + 1  
;W = 0xC3, ya que d = 0  
;PORTA = 0xC2`

- **INCFSZ** Incrementa f, salta si el resultado es 0

Sintaxis: [etiqueta]INCFSZ f, d

Operación:  $(f) + 1 \rightarrow \text{destino}$ ; salta si el resultado es 0

Bits afectados en STATUS:Ninguno



Descripción: El contenido del registro 'f' es incrementado. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'. Si el resultado es 0, entonces la siguiente instrucción es descartada.

Ejemplo.

```

AQUI      INCFSZ    PORTA, 0
          GOTO      CICLO
CONTINUA   . . .
          . . .
;PORTA = 0xC2, W = 0xC1, como el resultado
;no es cero continúa con la ejecución de la
;siguiente instrucción que es el GOTO CICLO

```

- **IORWF** OR inclusiva entre W y f

Sintaxis: [etiqueta]IORWF f, d

Operación: (W) OR (f) → destino

Bits afectados en STATUS: Z

Descripción: Operación lógica OR inclusiva del registro W con el registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo.

```

IORWF PORTA, 0 ; (W) ← (W) OR (PORTA)
; W          = 0001 0111 (0x17)
; PORTA      = 1100 0010 (0xC2)
;
; W          = 1101 0111 (0xD7) ya que d = 0
; PORTA = 0xC2

```

- **MOVF** Mueve a f

Sintaxis: [etiqueta]MOVF f, d

Operación: (f) → destino

Bits afectados en STATUS: Z

Descripción: El contenido del registro 'f' es movido al destino dependiendo del estado de 'd'. Si 'd' es 0, el destino es el registro W. Si 'd' es 1, el destino es el registro 'f'. 'd' = 1 es útil para probar un registro de archivo dado que la bandera Z del estatus es afectada.

Ejemplo.

```

MOVF PORTA, 0
; W = 0xC2, ya que d = 0
; Z = 0

```

- **MOVWF** Mueve W a f

Sintaxis: [etiqueta]MOVWF f  
 Operación:  $(W) \rightarrow f$   
 Bits afectados en STATUS: Ninguno  
 Descripción: Mueve el dato del registro W al registro 'f'.

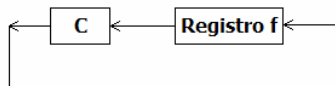
Ejemplo. 

```
MOVWF PORTA
;W      = 0x17
;PORTA = 0x17
```

- NOP No operación  
 Sintaxis: [etiqueta]NOP  
 Operación: No operación  
 Bits afectados en STATUS: Ninguno  
 Descripción: No operación.

Ejemplo. NOP

- RLF Rotación a la izquierda de f a través del acarreo  
 Sintaxis: [etiqueta]RLF f, d  
 Operación: Ver descripción  
 Bits afectados en STATUS: C  
 Descripción: El contenido del registro 'f' es rotado en un bit hacia la izquierda a través de la Bandera de Acarreo. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.



Ejemplo. 

```
RLF PORTA, 0
;PORTA = 1100 0010 (0xC2)
;W      = 1000 0100 (0x84), ya que d = 0
;C = 1
```

- RRF Rotación a la derecha de f a través del acarreo  
 Sintaxis: [etiqueta]RRF f, d  
 Operación: Ver descripción  
 Bits afectados en STATUS: C

**Descripción:** El contenido del registro 'f' es rotado en un bit hacia la derecha a través de la Bandera de Acarreo. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.



Ejemplo.

```
RRF    PORTA, 1
;PORTA = 0110 0001      (0x61), ya que d = 1
;C = 0
```

- SUBWF** Restar de W a f  
**Sintaxis:** [etiqueta]SUBWF f, d  
**Operación:** (f) - (W) → destino  
**Bits afectados en STATUS:** C, DC, Z  
**Descripción:** Resta (método de complemento a 2) del registro W al registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo.

```
SUBWF PORTA, 0 ; (W) ← (PORTA) - (W)
; W = 0xAB, ya que d = 0
```

- SWAPF** Intercambio de Nibbles en f  
**Sintaxis:** [etiqueta]SWAPF f, d  
**Operación:** (f<3:0>) → destino<7:4>,  
 (f<7:4>) → destino<3:0>  
**Bits afectados en STATUS:** Ninguno  
**Descripción:** Los cuatro bits más significativos y los cuatro bits menos significativos son intercambiados. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo.

```
SWAPF PORTA, 0
; W = 0x2C, ya que d = 0
```

- XORWF** OR exclusiva entre W y f  
**Sintaxis:** [etiqueta]XORWF f, d

Operación: (W) XOR (f) → destino

Bits afectados en STATUS: Z

Descripción: Operación lógica OR exclusiva del registro W con el registro 'f'. Si 'd' es 0 el resultado es guardado en el registro W. Si 'd' es 1 el resultado es guardado de nuevo en el registro 'f'.

Ejemplo.

```
XORWF PORTA, 0 ; (W) ← (W) XOR (PORTA)
; W           = 0001 0111 (0x17)
; PORTA       = 1100 0010 (0xC2)
;
; W           = 1101 0101 (0xD5) ya que d = 0
; PORTA = 0xC2
```

### Operaciones orientadas a bits

Para los ejemplos de las operaciones orientadas a bits los estados iniciales de los registros se muestran en la siguiente tabla:

| Registro | Contenido |
|----------|-----------|
| PORTA    | 0xC2      |

En todos estos ejemplos el formato 0xhh representa un número hexadecimal, en donde h significa un dígito hexadecimal.

- BCF Apaga un bit en f  
 Sintaxis: [etiqueta]BCF f, d  
 Operación:  $0 \rightarrow f<b>$   
 Bits afectados en STATUS: Ninguno  
 Descripción: El bit 'b' en el registro 'f' es apagado.

Ejemplo.

```
BCF PORTA, 7
; PORTA = 0100 0010 (0x42)
```

- BSF Prende un bit en f  
 Sintaxis: [etiqueta]BSF f, d  
 Operación:  $1 \rightarrow f<b>$   
 Bits afectados en STATUS: Ninguno  
 Descripción: El bit 'b' en el registro 'f' es prendido.

Ejemplo.

```
BSF PORTA, 0
```

; PORTA = 1100 0011 (0xC3)

- **BTFSC** Prueba un bit, salta si está apagado  
 Sintaxis: [etiqueta]BTFSC f, b  
 Operación: Salta si (f<b>) = 0  
 Bits afectados en STATUS:Ninguno  
 Descripción: Si el bit 'b' del registro 'f' es cero entonces la siguiente instrucción es saltada.

Ejemplo. BTFSC PORTA, 5  
 ; como el bit 5 del puerto A es 0, la siguiente  
 ; instrucción será saltada

- **BTFSS** Prueba un bit, salta si está prendido  
 Sintaxis: [etiqueta]BTFSS f, b  
 Operación: Salta si (f<b>) = 1  
 Bits afectados en STATUS:Ninguno  
 Descripción: Si el bit 'b' del registro 'f' es uno entonces la siguiente instrucción es saltada.

Ejemplo. BTFSS PORTA, 0  
 ; como el bit 0 del puerto A es 0, la siguiente  
 ; instrucción se ejecutará, es decir, no  
 ; hay salto

### Operaciones con literales y de control

Para los ejemplos de las operaciones con literales y de control los estados iniciales de los registros se muestran en la siguiente tabla:

| Registro | Contenido |
|----------|-----------|
| W        | 0x34      |

En todos estos ejemplos el formato 0xhh representa un número hexadecimal, en donde h significa un dígito hexadecimal.

- **ADDLW** Suma entre una literal y W  
 Sintaxis: [etiqueta]ADDLW k  
 Operación: (W) + k → (W)  
 Bits afectados en STATUS:C, DC, Z

Descripción: El contenido del registro W es sumado a una literal 'k' de ocho bits y el resultado es guardado en el registro W.

Ejemplo. `ADDLW 0x15; (W) ← (W) + k`  
`; W = 0x49`

- **ANDLW** AND entre una literal y W

Sintaxis: `[etiqueta]ANDLW k`

Operación:  $(W) \text{ AND } k \rightarrow (W)$

Bits afectados en STATUS: Z

Descripción: Al contenido del registro W se le aplica la operación lógica AND con una literal 'k' de ocho bits. El resultado es guardado en el registro W.

Ejemplo. `ANDLW 0x5F ; (W) ← (W) AND k`  
`; W = 0011 0100 (0x34)`  
`; k = 0101 1111 (0x5F)`  
`; -----`  
`; W = 0001 0100 (0x14)`

- **CALL** Llamada a una subrutina

Sintaxis: `[etiqueta]CALL k`

Operación:  $(PC) + 1 \rightarrow \text{TOS},$   
 $K \rightarrow \text{PC} \langle 10:0 \rangle,$   
 $(\text{PCLATH} \langle 4:3 \rangle) \rightarrow \text{PC} \langle 12:11 \rangle$

Bits afectados en STATUS: Ninguno

Descripción: Llamada a una subrutina. Primero, la dirección de retorno de 13 bits ( $PC + 1$ ) es guardada en la pila. La dirección de once bits inmediata es cargada en el PC en los bits  $\langle 10:0 \rangle$ . Los bits superiores del PC son cargados del PCLATH  $\langle 4:3 \rangle$ . CALL es una instrucción de dos ciclos de instrucción.

Ejemplo. `AQUI CALL ALLA`  
`; PC = Dirección de ALLA`  
`; TOS = Dirección de AQUI + 1`

- **CLRWDT** Apaga el temporizador Watchdog

Sintaxis: `[etiqueta]CLRWDT`

Operación:  $00h \rightarrow \text{WDT},$

0 → Cuenta del preescalador WDT,

1 →  $\overline{TO}$

1 →  $\overline{PD}$

Bits afectados en STATUS:  $\overline{TO}$ ,  $\overline{PD}$

Descripción: La instrucción CLRWDT apaga el temporizador Watchdog. También borra la cuenta del preescalador del WDT. Los bits del estatus  $\overline{TO}$ ,  $\overline{PD}$  son prendidos.

Ejemplo.

```
CLRWDT
; cuenta del WDT = 0x00
; cuenta del preescalador WDT = 0
;  $\overline{TO}$  = 1
;  $\overline{PD}$  = 1
```

- **GOTO** Salto incondicional

Sintaxis: [etiqueta]GOTO k

Operación:  $k \rightarrow PC<10:0>$

$PCLATH<4:3> \rightarrow PC<12:11>$

Bits afectados en STATUS: Ninguno

Descripción: GOTO es un salto incondicional. El valor de once bits inmediato es cargado en los bits del PC <10:0>. Los bits superiores del PC son cargados del PCLATH <4:3>. GOTO es una instrucción de dos ciclos de instrucción.

Ejemplo.

```
GOTO ALLA
; PC = dirección de ALLA
```

- **IORLW** OR inclusiva entre una literal y W

Sintaxis: [etiqueta]IORLW k

Operación:  $(W) \text{ OR } k \rightarrow (W)$

Bits afectados en STATUS: Z

Descripción: Al contenido del registro W se le aplica la operación lógica OR inclusiva con una literal 'k' de ocho bits. El resultado es guardado en el registro W.

Ejemplo.

```
IORLW 0x5F ; (W) ← (W) OR k
; W = 0011 0100 (0x34)
; k = 0101 1111 (0x5F)
; -----
; W = 0111 1111 (0x7F)
```

- **MOVLW** Mueve una literal a W  
 Sintaxis: [etiqueta]MOVLW k  
 Operación:  $k \rightarrow (W)$   
 Bits afectados en STATUS: Ninguno  
 Descripción: La literal de ocho bits 'k' es cargada en el registro W. Los don't cares serán ensamblados como ceros.

Ejemplo. `MOVLW 0x33 ; (W) ← k  
; W = 0x33`

- **RETFIE** Regreso de una interrupción  
 Sintaxis: [etiqueta]RETFIE  
 Operación:  $TOS \rightarrow PC,$   
 $1 \rightarrow GIE$   
 Bits afectados en STATUS: Ninguno  
 Descripción: Regreso de una interrupción. La dirección de 13 bits del tope de la pila (TOS) son cargados en el PC. El bit de habilitación de interrupciones globales, GIE (INTCON<7>), es automáticamente prendido, habilitando las interrupciones. Esta es una instrucción de dos ciclos de instrucción.

Ejemplo. `RETFIE  
; PC = TOS  
; GIE = 1`

- **RETLW** Regreso de una interrupción con una literal en W  
 Sintaxis: [etiqueta]RETLW k  
 Operación:  $k \rightarrow W,$   
 $TOS \rightarrow PC$   
 Bits afectados en STATUS: Ninguno  
 Descripción: El registro W es cargado con una literal de ocho bits 'k'. El contador del programa es cargado con la dirección de 13 bits en el Tope de la Pila (dirección de retorno). Esta es una instrucción de dos ciclos de instrucción.



Ejemplo. RETLW 0x07  
; W = 0x07  
; PC = TOS

- **RETURN** Regreso de una subrutina  
 Sintaxis: [etiqueta]RETURN  
 Operación: TOS → PC  
 Bits afectados en STATUS: Ninguno  
 Descripción: Regreso de una subrutina. Se obtiene el valor en el tope de la pila (TOS) y es cargado en el contador del programa. Esta es una instrucción de dos ciclos de instrucción.

Ejemplo. RETURN  
; PC = TOS

- **SLEEP**  
 Sintaxis: [etiqueta]SLEEP  
 Operación: 00h → WDT,  
 0 → Cuenta del preescalador WDT,  
 1 →  $\overline{TO}$   
 0 →  $\overline{PD}$   
 Bits afectados en STATUS:  $\overline{TO}$ ,  $\overline{PD}$   
 Descripción: El bit del estatus de apagado,  $\overline{PD}$  es apagado. El bit de estatus de tiempo fuera,  $\overline{TO}$  es prendido. El temporizador Watchdog y la cuenta de se preescalador son borrados. El procesador se pone en modo de SLEEP con el temporizador detenido.

Ejemplo. SLEEP

- **SUBLW** Resta W a una literal  
 Sintaxis: [etiqueta]SUBLW k  
 Operación:  $k - (W) \rightarrow (W)$   
 Bits afectados en STATUS: C, DC, Z  
 Descripción: El registro W es restado (por el método de complemento a 2) de una literal 'k' de ocho bits. El resultado es guardado en el registro W.

Ejemplo. SUBLW 0x4F;  $(W) \leftarrow k - (W)$

---



---

```
; W = 0x1B
```

- XORLW                      OR exclusiva entre una literal y W

Sintaxis:                      [etiqueta]XORLW k

Operación:                    (W) XOR k → (W)

Bits afectados en STATUS: Z

Descripción:                   Al contenido del registro W se le aplica la operación lógica OR exclusiva con una literal 'k' de ocho bits. El resultado es guardado en el registro W.

Ejemplo.

```
XORLW 0x5F ; (W) ← (W) XOR k
; W      = 0011 0100 (0x34)
; k      = 0101 1111 (0x5F)
;
;          -----
; W      = 0110 1011 (0x6B)
```

El registro de estado se muestra en la siguiente figura:

| R/W-0 | R/W-0 | R/W-0 | R-1             | R-1             | R/W-x | R/W-x | R/W-x |
|-------|-------|-------|-----------------|-----------------|-------|-------|-------|
| IRP   | RP1   | RP0   | $\overline{TO}$ | $\overline{PD}$ | Z     | DC    | C     |
| bit 7 |       |       |                 |                 |       |       | bit 0 |

|         |                                                                                                                                                                                                                   |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| bit 7   | IRP: Bit de selección de banco de registros<br>(utilizado para direccionamiento indirecto)<br>1 = Bancos 2, 3 (100h – 1FFh)<br>0 = Bancos 0, 1 (00h – FFh)                                                        |
| bit 6:5 | RP1:RP0: Bits de selección de banco de registros<br>(utilizado para direccionamiento directo)<br>11 = Banco 3 (180h – 1FFh)<br>10 = Banco 2 (100h – 17Fh)<br>01 = Banco 1 (80h – FFh)<br>00 = Banco 0 (00h – 7Fh) |
| bit 4   | $\overline{TO}$ : Bit de tiempo fuera<br>1 = Después de prender, instrucciones CLRWD $\overline{T}$ o SLEEP<br>0 = Un tiempo fuera de WDT ha ocurrido                                                             |
| bit 3   | $\overline{PD}$ : Bit de apagado<br>1 = Después de prender o por la instrucción CLRWD $\overline{T}$<br>0 = Por la ejecución de la instrucción SLEEP                                                              |
| bit 2   | Z: Bit de cero<br>1 = El resultado de una operación aritmética o lógica es cero<br>0 = El resultado de una operación aritmética o lógica no es cero                                                               |
| bit 1   | DC: Bit de medio acarreo<br>1 = El resultado produce un medio acarreo<br>0 = El resultado no produce un medio acarreo                                                                                             |
| bit 0   | C: Bit de acarreo<br>1 = El resultado produce un acarreo en el bit más significativo<br>0 = El resultado no produce un acarreo en el bit más significativo                                                        |

Contiene el estado aritmético de la ALU, el estado del RESET y los bits de selección de banco para la memoria de datos. Dado que la selección de los bancos de la memoria de datos es controlada por este registro, es requerida su presencia en cada banco. Incluso, este registro se encuentra en la misma posición relativa en cada banco.

El registro de estado puede ser el destino de cualquier instrucción, como cualquier otro registro. Si el registro de estado es el destino para una instrucción que afecta los bits Z, DC o C, entonces la escritura para estos tres bits estará deshabilitada. Estos bits se prenderán o apagarán de acuerdo a la lógica del dispositivo. Es más, los bits  $\overline{TO}$  y  $\overline{PD}$  no se pueden escribir. Por lo tanto, el resultado de una operación con el registro de estado como destino tal vez sea diferente al esperado.

Por ejemplo, la instrucción `CLRF STATUS` limpiará los tres bits superiores y prenderá el bit Z. Esto deja al registro de estado de la siguiente manera: 000u u1uu (donde u = sin cambio).

Por lo tanto, se recomienda que sólo se usen las instrucciones `BCF`, `BSF`, `SWAPF` y `MOVWF` para alterar el registro de estado ya que estas instrucciones no afectan los bits Z, C o DC de dicho registro. Para más instrucciones que no afecten ninguno de los bits de estado consulte el conjunto de instrucciones.

**Programa que controla al AGV**

```

;PROGRAMA DE CONTROL PARA EL SEGUIDOR DE LINEA
    ERRORLEVEL -302
    INCLUDE      "P16F877.INC"

;REGISTROS DE RAM GENERAL
    CBLOCK      0X20
        MASCARA
    VALOR DE LA MASCARA
        TRASH
    BASURA DE LAS COMPARACIONES
    ENDC

;SENSOR DERECHO
;SENSOR DEL CENTRO
;SENSOR IZQUIERDO
;SENTIDO MOTOR DERECHO
;SENTIDO MOTOR IZQUIERDO
;ENABLE MOTOR DERECHO
;ENABLE MOTOR IZQUIERDO

#DEFINE  SENSORD    PORTB,0
#DEFINE  SENSORC    PORTB,1
#DEFINE  SENSORI    PORTB,2
#DEFINE  MOTORD     PORTD,0
#DEFINE  MOTORI     PORTD,2
#DEFINE  ENABLED     PORTC,1
#DEFINE  ENABLEI     PORTC,2

;ORGANIZACIÓN DE PROGRAMA
    ORG      0X00
    GOTO     INI          ;VE AL INICIO DEL PROGRAMA, RUTINA
    DE INICIO DE PROGRAMA

INI  BSF      STATUS,RP0  ;CAMBIO A BANCO1 PARA CONFIGURAR
    CLRf      TRISD      ;CONFIGURO PORTD COMO SALIDA
    BCF       PORTC,1     ;CONFIGURO PC1 COMO SALIDA, ALLI
    ESTAN LOS ENABLE DEL MOTOR DERECHO
    BCF       PORTC,2     ;CONFIGURO PC2 COMO SALIDA ALLI
    ESTA EL ENABLE DEL MOTOR IZQUIERDO
    MOVLW    .255         ;MUEVO VALOR DE ENTRADA,
    MOVWF    TRISB        ;ASIGNO EL VALOR PARA QUE PUERTO B
    SEA ENTRADA
    BCF      STATUS,RP0  ;CAMBIO AL BANCO 0, YA CONFIGURE TODO LO
    QUE DEBIA EN EL BANCO 1
;INICIALIZACION DE VARIABLES

    BCF      ENABLED      ;PARO MOTORES
    BCF      ENABLEI      ;PARO MOTORES

;PROGRAMA GENERAL

PROGRAMA    MOVLW B'00000111' ;MUEVO ESTE VALOR DE
DISCRIMINACION, YA QUE ESOS SON LOS UNICOS VALORES QUE NOS IMPORTAN
        MOVWF MASCARA        ;MUEVO EL VALOR A LA
MASCARA

        MOVF      PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
        ANDWF     MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
        MOVWF    TRASH        ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
        MOVLW    B'00000000'  ;MUEVO EL VALOR PARA
SABER SI NO VE NINGUN SENSOR
        XORWF    TRASH,F      ;COMPARO EL VALOR

```

```

                                BTFSC      STATUS,Z      ;NO ESTA VIENDO
LA LINEA NINGUN SENSOR???
                                GOTO       INDETER      ;SI, VE A LA
SUBROUTINA INDETERMINADO

                                MOVF       PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH        ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000001'    ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DE LA DERECHA
                                XORWF     TRASH,F      ;COMPARO EL VALOR
                                BTFSC     STATUS,Z      ;ESTA VIENDO EL
SENSOR DE LA DERECHA???
                                GOTO       VUELTADER    ;SI; VE A LA SUBROUTINA
VUELTA DERECHA

                                MOVF       PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH        ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000010'    ;MUEVO EL VALOR PARA
SABER SI ESTA VIENDO EL SENSOR DEL CENTRO
                                XORWF     TRASH,F      ;COMPARO EL VALOR
                                BTFSC     STATUS,Z      ;ESTA VIENDO EL
SENSOR DEL CENTRO???
                                GOTO       RECTA        ;SI, VE A LA
SUBROUTINA RECTA

                                MOVF       PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH        ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000011'    ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DEL CENTRO Y EL DE LA DERECHA
                                XORWF     TRASH,F      ;COMPARO EL VALOR
                                BTFSC     STATUS,Z      ;ESTA VIENDO EL
SENSOR DEL CENTRO Y LA DERECHA???
                                GOTO       VUELTADER    ;SI, VE A LA SUBROUTINA
VUELTA DERECHA

                                MOVF       PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH        ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000100'    ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DE LA IZQUIERDA
                                XORWF     TRASH,F      ;COMPARO EL VALOR

```

```

                                BTFSC      STATUS,Z          ;ESTA VIENDO EL
SENSOR DE LA IZQUIERDA???
                                GOTO       VUELTAIZQ         ;SI, VE A LA
SUBROUTINA VUELTA IZQUIERDA

                                MOVF       PORTB,W           ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W         ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH              ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000101'         ;MUEVO EL VALOR PARA
SABER SI ESTA VIENDO EL SENSOR DE LA IZQUIERDA Y EL DE LA DERECHA
                                XORWF      TRASH,F            ;COMPARO EL VALOR
                                BTFSC      STATUS,Z          ;ESTA VIENDO EL
SENSOR DE LA IZQUIERDA Y EL DE LA DERECHA???
                                GOTO       INDETER            ;SI, VE A LA
SUBROUTINA INDETERMINADO

                                MOVF       PORTB,W           ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W         ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH              ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000110'         ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DE LA DERECHA Y EL DEL CENTRO
                                XORWF      TRASH,F            ;COMPARO EL VALOR
                                BTFSC      STATUS,Z          ;ESTA VIENDO EL
SENSOR DE LA IZQUIERDA Y EL DEL CENTRO???
                                GOTO       VUELTAIZQ         ;SI, VE A LA
SUBROUTINA VUELTA IZQUIERDA

                                MOVF       PORTB,W           ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF      MASCARA,W         ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH              ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW     B'0000111'         ;MUEVO EL VALOR PARA
SABER SI VEN TODOS LOS SENSORES
                                XORWF      TRASH,F            ;COMPARO EL VALOR
                                BTFSC      STATUS,Z          ;ESTAN VIENDO
TODO LOS SENSORES???
                                GOTO       INDETER            ;SI, VE A LA
SUBROUTINA INDETERMINDO

                                GOTO       PROGRAMA

VUELTA DER     BCF          ENABLED          ;APAGO EL MOTOR DERECHO
MOTOR IZQUIERDO      BSF          ENABLEI      ;ENCIENDO EL
HACIA ADELANTE      BSF          MOTORI        ;EL MOTORI VA
                                GOTO       PROGRAMA          ;REGRESO AL PROGRAMA

RECTA DERECHO      BSF          ENABLED        ;ENCIENDO EL MOTOR
MOTOR IZQUIERDO      BSF          ENABLEI      ;ENCIENDO EL

```

---



---

|                      |      |          |                         |
|----------------------|------|----------|-------------------------|
| HACIA ADELANTE       | BSF  | MOTORD   | ;EL MOTORD VA           |
| HACIA ADELANTE       | BSF  | MOTORI   | ;EL MOTORI VA           |
|                      | GOTO | PROGRAMA | ;REGRESO AL PROGRAMA    |
| VUELTAIZQ<br>DERECHO | BSF  | ENABLED  | ;ENCIENDO EL MOTOR      |
| IZQUIERDO            | BCF  | ENABLEI  | ;AOAGO EL MOTOR         |
| HACIA ADELANTE       | BSF  | MOTORD   | ;EL MOTORD VA           |
|                      | GOTO | PROGRAMA | ;REGRESO AL PROGRAMA    |
| INDETER              | BCF  | ENABLED  | ;APAGO EL MOTOR DERECHO |
| IZQUIERDO            | BCF  | ENABLEI  | ;APAGO EL MOTOR         |
|                      | GOTO | PROGRAMA | ;REGRESO AL PROGRAMA    |
| END                  | ;FIN |          |                         |



**Programa que controla todo el movimiento del AGV**

```

;PROGRAMA QUE CONTROLA TODO EL MOVIMIENTO
    ERRORLEVEL -302
    INCLUDE      "P16F877.INC"

;REGISTROS DE RAM GENERAL
    CBLOCK      0X20
        CUENTA          ;LUGAR DONDE SE LLEVA LA
CUENTA DE TIEMPO PARA SABER SI YA PASO TODO EL PERIODO DE PWM (20 ms)
        CUENTA2         ;LUGAR DONDE SE LLEVA LA
CUENTA DE TIEMPO PARA SABER SI YA PASO TODO EL PERIODO DE TRABAJO
        W_BAK           ;LUGAR DONDE SE CREA UN
RESPALDO DEL PWM
        S_BAK           ;LUGAR DONDE SE RESPALDA EL
ESTATUS
        PWM1            ;LUGAR DONDE SE VE EL VALOR
ACTUAL DEL PWM
        ;ESTADO         ;LUGAR DONDE SE ASIGAN EL
ESTADO DE LA PINZA
        TPRE            ;LUGAR DONDE SE VA A LLEVAR
EL TIEMPO PARA SABER CUANTO TIEMPO LLEVA VIENDO ESTO
        TPRE1           ;LUGAR DONDE SE VA A LLEVAR
EL TIEMPO PARA SABER CUANTO TIEMPO LLEVA VIENDO ESTO
        TPRE2           ;LUGAR DONDE SE VA A LLEVAR
EL TIEMPO PARA SABER CUANTO TIMEPO LLEVA VIENDO ESTO
        MASCARA         ;LUGAR DONDE SE GUARDARA
CUANDO SE APLIQUE UNA MASCARA
        TRASH           ;LUGAR DONDE SE GUARDA LA
BASURA DE LAS COMPARACIONES
        TIME            ;LUGAR DONDE SE GUARDA EL
TIEMPO TRANSCURRIDO
        TMP0            ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        TMP1            ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        TMP2            ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        VALOR1          ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        VALOR2          ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        VALOR3          ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        RUTA            ;LUGAR DONDE SE GUARDA LA
RUTA A SEGUIR
        ESTADO          ;EN ESTE ESPACIO DE MEMORIA
SE ALMACENA EL ESTADO EN EL QUE SE ENCUENTRA EL PROGRAMA
    ENDC

#DEFINE  SENSORD  PORTB,0      ;SENSOR DERECHO
#DEFINE  SENSORC  PORTB,1      ;SENSOR DEL CENTRO
#DEFINE  SENSORI  PORTB,2      ;SENSOR IZQUIERDO
#DEFINE  MOTORD   PORTD,0      ;SENTIDO MOTOR DERECHO
#DEFINE  MOTORI   PORTD,2      ;SENTIDO MOTOR IZQUIERDO
#DEFINE  ENABLED  PORTC,1      ;ENABLE MOTOR DERECHO
#DEFINE  ENABLEI  PORTC,2      ;ENABLE MOTOR IZQUIERDO
#DEFINE  SERV01   PORTD,6      ;SALIDA PARA EL SERVO

; DEFINICION DE CONSTANTES PARA EL MOVIMIENTO DE LA

```

```

MAX            EQU    .200            ;VALOR PARA EL VALOR DEL CICLO DE
TRABAJO DE    LOS SERVOS (20 [ms]) EN ESCALA DE 0.1 [ms]
MIN1           EQU    .97             ;VALOR MINIMO PARA EL PERIODO DE
TRABAJO DE    LOS SERVOS (0.97 [ms]) EN ESCALA DE 0.01[ms]
REQUERIDO      EQU    .145            ;VALOR QUE QUEREMOS PARA EL
MOVIEMIENTO DE 40°, 1.45[ms] EN ESCALA DE 0.01[ms]

;ORGANIZACIÓN DE PROGRAMA
    ORG    0X00
    GOTO   INI                        ;VE AL INICIO DEL PROGRAMA, RUTINA DE
INICIO DE PROGRAMA

    ORG    0X04
    GOTO   INTER                      ;LLAMADA A LA INTERRUPCION, NOS VAMOS A LA
RUTINA INTERRUPCION

;CONFIGURACIÓN DEL SISTEMA

INI    BSF        STATUS,RP0 ;CAMBIO A BANCO1 PARA CONFIGURAR LOS
PARAMETROS NECESARIOS
    CLRF          TRISD            ;CONFIGURO PORTD COMO SALIDA
    MOVLW .255                ;MUEVO VALOR DE ENTRADA PARA PUROS
UNOS,
    MOVWF TRISB                ;ASIGNO EL VALOR PARA QUE PUERTOB
SEA ENTRADA
    BCF           PORTC,1        ;CONFIGURO PC1 COMO SALIDA, ALLI
ESTAN LOS ENABLE DEL MOTOR DERECHO
    BCF           PORTC,2        ;CONFIGURO PC2 COMO SALIDA ALLI
ESTA EL ENABLE DEL MOTOR IZQUIERDO

;CONFIGURACION DEL TIMER0

    MOVLW B'00000000'          ;MOVIENDO LA CONFIGURACION DEL TIMER
    MOVWF OPTION_REG ;ASIGNO LA CONFIGURACION
    BCF          STATUS,RP0 ;CAMBIO AL BANCO 0, YA CONFIGURE TODO LO
QUE DEBIA EN EL BANCO 1

;INICIALIZACION DE VARIABLES

    CLRF          ESTADO          ;ESTAMOS EN EL ESTADO INICIAL DEL
PROCESO
    CLRF          RUTA            ;EN UN PRINCIPIO DEL PROGRAMA NO SE
TIENE UNA RUTA DEFINIDA
    CLRF          TPRES           ;ASIGNAMOS EL VALOR PARA QUE SE VEA
QUE NO LLEVA MUCHO TIEMPO PRECIONADO EL BOTON
    CLRF          TPRES1          ;ASIGNAMOS EL VALOR PARA QUE SE VEA
QUE NO LLEVA MUCHO TIEMPO PRECIONADO EL BOTON
    CLRF          TPRES2          ;ASIGNAMOS EL VALOR PARA QUE SE VEA
QUE NO LLEVA MUCHO TIEMPO PRECIONADO EL BOTON
    BCF           ENABLED         ;PARO MOTORES
    BCF           ENABLEI         ;PARO MOTORES
    CLRF          TIME            ;LIMPIO EL TIME QUE SIRVE PARA EL
SERVO
    CLRF          CUENTA          ;INICIALIZO REGISTRO DE CONTEO DE
EVENTOS, INICIAMOS EN 0 LO CUAL SON 0 MS TRANSCURRIDOS
    CLRF          CUENTA2         ;LIMPIO UNA DE LAS CUENTAS DEL
SERVO
    MOVLW MIN1                ;CARGO EL VALOR MINIMO EN EL ACUMULADOR,
ESTE VALOR ES PARA QUE LA PINZA NO SE LEVANTE, EL ANCHO DE PULSO ES DE
1[MS]

```

```

MOVWF PWM1 ;MUEVE EL VALOR PARA QUE EL PWM ARRANQUE
EN LA POSICION DE 0°
BSF SERV01 ;EL VALOR DEL PWM SIEMPRE
INICA EN FLANCO ALTO

;CONFIGURACION DEL TIMER0 PARA EL SERVO
MOVLW .236 ;MUEVO EL VALOR DEL TIMER0
MOVWF TMR0 ;ASIGNO EL VALOR DEL TIMER0 PARA QUE
OCURRA UNA INTERRUPCION CASA 0.01[ms]
BCF INTCON,T0IF ;LIMPIO LA BANDERA DE INTERRUPCION
BSF INTCON,T0IE ;DOY EL PERMISO DE INTERRUPCION, ENCIENDO
EL TIMER0

;CONFIGURACION DE INTERRUPCIONES

BSF INTCON,GIE ;PERMISO GLOBAL DE INTERRUPCION
BSF INTCON,PEIE ;PERMISO DE INTERRUPCIÓN POR PERIFERICO

;SE ASIGNA LA RUTA POR MEDIO DEL PROGRAMA

MOVLW .2 ;VALOR PARA LA RUTA 1
MOVWF RUTA ;RUTA QUE SE VA A SEGUIR

;PARTE QUE ESPERA LA INDICACION DE LA RUTA

ESPERANDO MOVLW .0 ;VALOR INICIAL DE LA RUTA
XORWF RUTA, TRASH ;COMPARO EL VALORE DE LA RUTA
BTFSC STATUS,Z ;YA SE A ASIGNADO
RUTA???
GOTO ESPERANDO ;ESPERAMOS RUTA

;PROGRAMA GENERAL

PROGRAMA MOVLW B'000000111' ;MUEVO ESTE VALOR DE
DISCRIMINACION, YA QUE ESOS SON LOS UNICOS VALORES QUE NOS IMPORTAN
MOVWF MASCARA ;MUEVO EL VALOR A LA
MASCARA

MOVF PORTB,W ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
ANDWF MASCARA,W ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
MOVWF TRASH ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
MOVLW B'00000000' ;MUEVO EL VALOR PARA
SABER SI NO VE NINGUN SENSOR
XORWF TRASH,F ;COMPARO EL VALOR
BTFSC STATUS,Z ;NO ESTA VIENDO
LA LINEA NINGUN SENSOR???
GOTO DESICION ;SI, VE A LA
SUBROUTINA DESICION

MOVF PORTB,W ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
ANDWF MASCARA,W ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
MOVWF TRASH ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
MOVLW B'00000001' ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DE LA DERECHA

```

```

                                XORWF TRASH,F      ;COMPARO EL VALOR
                                BTFSC      STATUS,Z    ;ESTA VIENDO EL
SENSOR DE LA DERECHA???
                                GOTO        VUELTADER  ;SI; VE A LA SUBROUTINA
VUELTA DERECHA

```

```

                                MOVF          PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF        MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF TRASH      ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW B'0000010'      ;MUEVO EL VALOR PARA
SABER SI ESTA VIENDO EL SENSOR DEL CENTRO
                                XORWF TRASH,F      ;COMPARO EL VALOR
                                BTFSC      STATUS,Z    ;ESTA VIENDO EL
SENSOR DEL CENTRO???
                                GOTO        RECTA        ;SI, VE A LA
SUBROUTINA RECTA

```

```

                                MOVF          PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF        MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF TRASH      ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW B'0000011'      ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DEL CENTRO Y EL DE LA DERECHA
                                XORWF TRASH,F      ;COMPARO EL VALOR
                                BTFSC      STATUS,Z    ;ESTA VIENDO EL
SENSOR DEL CENTRO Y LA DERECHA???
                                GOTO        VUELTADER  ;SI, VE A LA SUBROUTINA
VUELTA DERECHA

```

```

                                MOVF          PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF        MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF TRASH      ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW B'0000100'      ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DE LA IZQUIERDA
                                XORWF TRASH,F      ;COMPARO EL VALOR
                                BTFSC      STATUS,Z    ;ESTA VIENDO EL
SENSOR DE LA IZQUIERDA???
                                GOTO        VUELTAIZQ   ;SI, VE A LA
SUBROUTINA VUELTA IZQUIERDA

```

```

                                MOVF          PORTB,W      ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF        MASCARA,W    ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF TRASH      ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW B'0000101'      ;MUEVO EL VALOR PARA
SABER SI ESTA VIENDO EL SENSOR DE LA IZQUIERDA Y EL DE LA DERECHA
                                XORWF TRASH,F      ;COMPARO EL VALOR

```

```

                                BTFSC      STATUS,Z          ;ESTA VIENDO EL
SENSOR DE LA IZQUIERDA Y EL DE LA DERECHA???
                                GOTO        INDETER           ;SI, VE A LA
SUBROUTINA INDETERMINADO

                                MOVF        PORTB,W           ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF       MASCARA,W         ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH              ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW      B'0000110'        ;MUEVO EL VALOR PARA
SABER SI VE EL SENSOR DE LA DERECHA Y EL DEL CENTRO
                                XORWF      TRASH,F            ;COMPARO EL VALOR
                                BTFSC      STATUS,Z          ;ESTA VIENDO EL
SENSOR DE LA IZQUIERDA Y EL DEL CENTRO???
                                GOTO        VUELTAIZQ        ;SI, VE A LA
SUBROUTINA VUELTA IZQUIERDA

                                MOVF        PORTB,W           ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF       MASCARA,W         ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF      TRASH              ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW      B'0000111'        ;MUEVO EL VALOR PARA
SABER SI VEN TODOS LOS SENSORES
                                XORWF      TRASH,F            ;COMPARO EL VALOR
                                BTFSC      STATUS,Z          ;ESTAN VIENDO
TODO LOS SENSORES???
                                GOTO        PINZA              ;SI, VE A LA
SUBROUTINA PINZA

                                GOTO        PROGRAMA

VUELTADER    BCF      ENABLED          ;APAGO EL MOTOR DERECHO
                                BSF        ENABLEI            ;ENCIENDO EL
MOTOR IZQUIERDO
                                BSF        MOTORI             ;EL MOTORI VA
HACIA ADELANTE
                                GOTO        REGRESO           ;REGRESO AL
PROGRAMA

RECTA        BSF        ENABLED          ;ENCIENDO EL MOTOR
DERECHO
                                BSF        ENABLEI            ;ENCIENDO EL
MOTOR IZQUIERDO
                                BSF        MOTORD             ;EL MOTORD VA
HACIA ADELANTE
                                BSF        MOTORI             ;EL MOTORI VA
HACIA ADELANTE
                                GOTO        REGRESO           ;REGRESO AL
PROGRAMA

VUELTAIZQ    BSF        ENABLED          ;ENCIENDO EL MOTOR
DERECHO
                                BCF        ENABLEI            ;AOAGO EL MOTOR
IZQUIERDO
                                BSF        MOTORD             ;EL MOTORD VA
HACIA ADELANTE

```

|                                                                                                           | GOTO            | REGRESO   | ;REGRESO AL                                    |
|-----------------------------------------------------------------------------------------------------------|-----------------|-----------|------------------------------------------------|
| PROGRAMA                                                                                                  |                 |           |                                                |
| INDETER                                                                                                   | BCF             | ENABLED   | ;APAGO EL MOTOR DERECHO                        |
| IZQUIERDO                                                                                                 | BCF             | ENABLEI   | ;APAGO EL MOTOR                                |
| PROGRAMA                                                                                                  | GOTO            | REGRESO   | ;REGRESO AL                                    |
| PINZA DERECHA                                                                                             | BCF             | ENABLED   | ;APAGO EL MOTOR DE LA                          |
| DE LA IZQUIERDA                                                                                           | BCF             | ENABLEI   | ;APAGO EL MOTOR                                |
| SI ESTAMOS EN EL ESTADO 1 DEL PROGRAMA                                                                    | MOVLW .1        |           | ;VALOR PARA SABER                              |
| ESTADO DE DESICION DE RUTA                                                                                | XORWF ESTADO, F |           | ;COMPARO EL VALOR                              |
| LAS ENTRADAS                                                                                              | BTFSS           | STATUS, Z | ;YA PASAMOS EL                                 |
|                                                                                                           | GOTO            | PROGRAMA  | ;NO, VE Y SUGUE VIENDO                         |
|                                                                                                           |                 |           | ;AHORA VEMOS CUANTO TIEMPO LLEVA VIENDO ESTO   |
| TIEMPO QUE LLEVA VIENDO ESTO                                                                              | INCF            | TPRE      | ;INCREMENTAMOS EL                              |
| DE COMPARACION, PARA SABER SI ES MAXIMO                                                                   | MOVLW .255      |           | ;MOVEMOS EL VALOR                              |
| COMPARACION;                                                                                              | XORWF TPRE, W   |           | ;HACEMOS AL                                    |
| ESTO MUCHO RATO???                                                                                        | BTFSS           | STATUS, Z | ;YA ESTUVO VIENDO                              |
| CHECANDO LOS SENSORES                                                                                     | GOTO            | PROGRAMA  | ;NO, SIGUE CONTINUA                            |
| LO VUELVA A HACER                                                                                         | CLRF            | TPRE      | ;LIMPIO PARA QUE                               |
| OTRA VARIABLE                                                                                             | INCF            | TPRE1     | ;INCREMENTAMOS LA                              |
| DE COMPARACION, PARA SABER SI ES MAXIMO                                                                   | MOVLW .255      |           | ;MOVEMOS EL VALOR                              |
| COMPARACION;                                                                                              | XORWF TPRE1, W  |           | ;HACEMOS AL                                    |
| ESTO MUCHO RATO???                                                                                        | BTFSS           | STATUS, Z | ;YA ESTUVO VIENDO                              |
| ESTA PRECIONADO                                                                                           | GOTO            | PROGRAMA  | ;NO, SIGUE VIENDO SI                           |
| LO VUELVA A HACER                                                                                         | CLRF            | TPRE      | ;LIMPIO PARA QUE                               |
| LO VUELVA A HACER                                                                                         | CLRF            | TPRE1     | ;LIMPIO PARA QUE                               |
| ULTIMA VARIABLE DE TIEMPO                                                                                 | INCF            | TPRE2     | ;INCREMENTO LA                                 |
| DE COMPARACION, PARA SABER SI ES MAXIMO, POR CADA DIEZ QUE AUMENTEMOS AQUI, ESTAMOS AUMENTADO APROX 2 [S] | MOVLW .8        |           | ;MOVEMOS EL VALOR                              |
| COMPARACION;                                                                                              | XORWF TPRE2, W  |           | ;HACEMOS AL                                    |
| ESTO MUCHO RATO???                                                                                        | BTFSS           | STATUS, Z | ;YA ESTUVO VINDO                               |
| SENSORES                                                                                                  | GOTO            | PROGRAMA  | ;NO, SIGUE CHECANDO LOS                        |
| PINZA                                                                                                     |                 |           | ;ESTUVO VIENDO MUCHO RATO, VAMOS A LEVANTAR LA |

```

                                GOTO      PWM                ;VE A QUE
LEVANTES LA PINZA

;PROGRAMAMOS LOS REGRESOS, QUE GIRE HASTA QUE ENCUENTRE LA OTRA LINEA
;REGRESO PARA LA RUTINA 2
RUTA2SIGUE BSF      ENABLEI          ;ENCIENDO EL MOTOR IZQUIERDA
                                BSF      MOTORI          ;MOTOR DE LA
IZQUIERDA HACIA ADELANTE. PARA QUE GIRE HASTA QUE ENCUENTRE OTRA VEZ
LA LINEA
                                MOVF      PORTB,W          ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF     MASCARA,W        ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES QUE NECESITO
                                MOVWF     TRASH             ;MUEVO EL VALOR AL
LUGAR DE COMPARACION,
                                MOVLW     B'00000010'       ;VALOR PARA SABER SI VE
EL SENSOR DEL CENTRO
                                XORWF     TRASH,F           ;COMPARO EL VALOR
                                BTFSS     STATUS,Z          ;ESTA VIENDO EL
SENSOR DE EN MEDIO???
                                GOTO      RUTA2SIGUE        ;NO, VE A RUTA2SIGUE
                                GOTO      PROGRAMA          ;SI, CONTINUA HACIENDO
EL PROGRAMA DE SEGUIDOR DE LINEA.

RUTA1SIGUE BSF      ENABLED          ;ENCIENDO EL MOTOR DERECHO
                                BSF      MOTORD            ;MOTOR DE LA
DERECHO HACIA ADELANTE. PARA QUE GIRE HASTA QUE ENCUENTRE OTRA VEZ LA
LINEA
                                MOVF      PORTB,W          ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                                ANDWF     MASCARA,W        ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES QUE NECESITO
                                MOVWF     TRASH             ;MUEVO EL VALOR AL
LUGAR DE COMPARACION,
                                MOVLW     B'00000010'       ;VALOR PARA SABER SI VE
EL SENSOR DEL CENTRO
                                XORWF     TRASH,F           ;COMPARO EL VALOR
                                BTFSS     STATUS,Z          ;ESTA VIENDO EL
SENSOR DE EN MEDIO???
                                GOTO      RUTA1SIGUE        ;NO, VE A RUTA2SIGUE
                                GOTO      PROGRAMA          ;SI, CONTINUA HACIENDO
EL PROGRAMA DE SEGUIDOR DE LINEA.

PWM          INCF      PWM1            ;QUE INCREMENTE EL PWM
                                MOVLW     .255              ;VALORA QUE SE LE
VA A MOVER AL DELAY
                                MOVWF     VALOR1            ;VALOR DEL DELAY
                                MOVLW     .4                ;VALORA QUE SE LE
VA A MOVER AL DELAY
                                MOVWF     VALOR2            ;VALOR DEL DELAY
                                MOVLW     .31               ;VALORA QUE SE LE
VA A MOVER AL DELAY
                                MOVWF     VALOR3            ;VALOR DEL DELAY
                                GOTO      DELAY              ;ME VOY A UN
RETARDO
SIGO          MOVLW     REQUERIDO        ;COMPARO A VER SI YA LLEGAMOS
AL MAXIMO INCREMENTO
                                XORWF     PWM1,TRASH        ;COMPARO EL VALOR DE
ACULADOR CON PWM1

```

```

                                BTFSS      STATUS,Z          ;YA ALCANZO EL
VALOR REUQUERIDO????
                                GOTO        PWM              ;NO, VE Y SIGUE
INCREMENTANDO
                                MOVLW      .2                ;NOS
ENCONTRAMOS EN EL ESTADO 2
                                MOVWF      ESTADO           ;NOS ENCONTRAMOS EN EL
ESTADO 2

;SUBROUTINA CREADA PARA QUE REGRESE A LA RUTA DE LA CUAL SE MANDO
LLAMAR
                                MOVLW      .1                ;CARGAMOS EL
VALOR DE LA RUTA 1
                                XORWF      RUTA, TRASH ;COMPARO EL VALORE DE LA RUTA
                                BTFSS      STATUS,Z          ;NOS MANDARON LA
RUTA 1???
                                GOTO        RUTA2SIGUE ;NO, VE A RECORRER LA
RUTA 2
                                GOTO        RUTA1SIGUE ;SI, RECORRE LA
RUTA1

;RETARDO
DELAY      MOVF      VALOR1,W
                                MOVWF      TMP0
DEL0       MOVF      VALOR2,W
                                MOVWF      TMP1
DEL1       MOVF      VALOR3,W
                                MOVWF      TMP2
DEL2       DECFSZ    TMP2,F
                                GOTO        DEL2
                                DECFSZ     TMP1,F
                                GOTO        DEL1
                                DECFSZ     TMP0,F
                                GOTO        DEL0
                                GOTO        SIG0

REGRESO    CLRf      TPRE          ;LIMPIO PARA QUE LO
VUELVA A HACER
                                CLRf      TPRE1             ;LIMPIO PARA QUE
LO VUELVA A HACER
                                CLRf      TPRE2             ;LIMPIO LA
BANDERA PARA QUE VUELVA A CHECAR
                                GOTO        PROGRAMA        ;REGRESAMOS AL PROGRAMA

DESICION   BCF      ENABLED        ;APAGO EL MOTOR DERECHA
                                BCF      ENABLEI            ;APAGO EL MOTOR
IZQUIERDA
                                ;AHORA VEMOS CUANTO TIEMPO LLEVA VIENDO ESTO
                                INCF      TPRE              ;INCREMENTAMOS EL
TIEMPO QUE LLEVA VIENDO ESTO
                                MOVLW      .255             ;MOVEMOS EL VALOR
DE COMPARACION, PARA SABER SI ES MAXIMO
                                XORWF      TPRE,W           ;HACEMOS AL
COMPARACION;
                                BTFSS      STATUS,Z          ;YA ESTUVO VIENDO
ESTO MUCHO RATO???
                                GOTO        PROGRAMA        ;NO, SIGUE CONTINUA
CHECANDO LOS SENSORES

```



```

LO VUELVA A HACER          CLRf      TPRe      ;LIMPIO PARA QUE
OTRA VARIABLE              INCf      TPRe1     ;INCREMENTAMOS LA
DE COMPARACION, PARA SABER SI ES MAXIMO
COMPARACION;              MOVLW .255          ;MOVEMOS EL VALOR
                          XORWF TPRe1,W      ;HACEMOS AL
ESTO MUCHO RATO???        BTfSS      STATUS,Z    ;YA ESTUVO VIENDO
ESTA PRECIONADO           GOTO      PROGRAMA    ;NO, SIGUE VIENDO SI
LO VUELVA A HACER          CLRf      TPRe      ;LIMPIO PARA QUE
LO VUELVA A HACER          CLRf      TPRe1     ;LIMPIO PARA QUE
ULTIMA VARIABLE DE TIEMPO  INCf      TPRe2     ;INCREMENTO LA
                          MOVLW .8            ;MOVEMOS EL VALOR
DE COMPARACION, PARA SABER SI ES MAXIMO, POR CADA DIEZ QUE AUMENTEMOS
AQUI, ESTAMOS AUMENTADO APROX 2 [S]
COMPARACION;              XORWF TPRe2,W      ;HACEMOS AL
ESTO MUCHO RATO???        BTfSS      STATUS,Z    ;YA ESTUVO VINDO
SENSORES                  GOTO      PROGRAMA    ;NO, SIGUE CHECANDO LOS
ESTADO 1 DEL PROGRAMA     MOVLW .1           ;PASAMOS AL
1 DEL PROGRAMA            MOVWF ESTADO        ;NOS MOVEMOS AL ESTADO
                          ;ESTUVO VIENDO MUCHO RATO

;AQUI PROGRAMAMOS LA RUTINA DEPENDIENDO DE LA RUTA QUE SE LE MANDO AL
SEGUIDOR DE LINEA.
VALOR DE LA RUTA 1        MOVLW .1           ;CARGAMOS EL
RUTA 1???                 XORWF RUTA, TRASH ;COMPARO EL VALORE DE LA RUTA
                          BTfSS      STATUS,Z    ;NOS MANDARON LA
RECORRER LA RUTA 2        GOTO      RUTA2      ;NO, VE A
RECORRE LA RUTA1          GOTO      RUTA1      ;SI,

;RUTAS PARA RECOGER LA CAJA DE DE REGRESO
RUTA1                     BSF         ENABLED    ;ENCIENDO EL MOTOR
DERECHA                   BSF         MOTORD     ;EL MOTORD VA
HACIA ADELANTE            ;AHORA VAMOS A CHECAR SI YA LLEGO AL PUNTO
DONDE VE EL DE ENMEDIO    MOVF        PORTB,W    ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
PARA SOLO RECIBIR LOS VALORES QUE NECESITO
LUGAR DE COMPARACION,    ANDWF      MASCARA,W    ;APLICO LA MASCARA,
EL SENSOR DEL CENTRO      MOVWF TRASH          ;MUEVO EL VALOR AL
                          MOVLW B'00000010'     ;VALOR PARA SABER SI VE
                          XORWF TRASH,F          ;COMPARO EL VALOR

```

```

SENSOR DEL CENTRO???      BTFSS      STATUS,Z          ;ESTA VIENDO EL
                           GOTO      RUTA1          ;NO, VE A RUTA1
                           ;YA LLEGO A LA POSICION DONDE ESTA LA CAJA
                           GOTO      PROGRAMA      ;REGRESA AL PROGRAMA

RUTA2
MOTOR DERECHA              BSF        ENABLED        ;ENCIENDO EL
                           BSF        ENABLEI        ;ENCIENDO EL
MOTOR IZQUIERDA            BSF        MOTORD         ;EL MOTORD VA
HACIA ADELANTE             BSF        MOTORI         ;EL MOTORI VA
HACIA ADELANTE             ;AHORA VAMOS A CHECAR SI YA LLEGO AL PUNTO
DONDE TODOS VEN            MOVF       PORTB,W        ;MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO
                           ANDWF      MASCARA,W      ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES QUE NECESITO
                           MOVWF      TRASH          ;MUEVO EL VALOR AL
LUGAR DE COMPARACION,
                           MOVLW      B'00000111'    ;VALOR PARA SABER SI SE
VEN TODOS LOS SENSORES
                           XORWF      TRASH,F        ;COMPARO EL VALOR
                           BTFSS      STATUS,Z        ;ESTA VIENDO
TODOS???
                           GOTO      RUTA2          ;NO, VE A RUTA2
                           ;YA LLEGO A LA POSICION DONDE ESTA LA CAJA
                           GOTO      PROGRAMA      ;REGRESA AL PROGRAMA

;SERVICIO DE INTERRUPCIÓN

INTER      MOVWF W_BAK      ;RESPALDO DEL VALOR ACTUAL DEL
ACUMULADOR
                           MOVF       STATUS,W        ;ASIGNO EL VALOR DE
ESTATUS AL ACUMULADOR
                           MOVWF      S_BAK          ;CREO EL RESPALDO DE ESTATUS
                           MOVLW      .236          ;ASIGNO EL VALOR DEL
TIMER0 PARA VOLVERLO A CONFIGURAR
                           MOVWF      TMR0          ;ASIGNO EL VALOR DEL TIMER0
PARA QUE ENTRE A LA INTERUPCION CADA 0.01[MS]
                           BCF        INTCON,T0IF ;LIMPIO LA BANDERA DE
INTERRUPCION
                           INCF        CUENTA2        ;INCREMENTO LA CUENTA2
                           MOVF       CUENTA2,W      ;CARGA EL VALOR DE CUENTA EN
EL ACUMULADOR
                           XORWF      PWM1,W        ;COMPARO EL VALOR REQUERIDO
CON LA CUENTA HASTA EL MOMENTO
                           BTFSS      STATUS,Z        ;EL VALOR DEL PWM
REQUERIDO ES INGUAL AL VALOR DEL LA CUENTA QUE LLEVAMOS???
                           GOTO      CONTINUA        ;NO, NO HA PASADO
COMPLETO EL PERIODO DE TRABAJO, CONTINUA CHECANDO LA INTERRUPCION
                           BCF        SERV01         ;SI, YA TENEMOS
EL PWM REQUERIDO, APAGO EL PUERTO
                           GOTO      VOLVER          ;REGRESA A LA
ROUTINA.DEL PROGRAMA

```

```

CONTINUA      INCF      TIME                ;INCREMENTO TIME QUE LLEVA LA
CUENTA DEL TIEMPO QUE HA PASADO POR LA INTERRUPCION
              MOVLW .10                    ;EL VALOR PARA SABER SI
YA PASO 10 VECES
              XORWF TIME,W                ;YA PASO 10 VECES???.
              BTFSS      STATUS,Z          ;YA PASÓ 1 [ms] DESDE
LA ULTIMA VEZ QUE ENTRASTE???
              GOTO       VOLVER            ;NO, VAMONOS A LA
RUTINA DEL PROGRAMA
              CLRF       TIME              ;SI, ENTONCES
CONTINUAMOS.Y LIMPIAMOS TIME
              INCF       CUENTA,F          ;CUENTA = CUENTA + 1,
LO CUAL SE IGNIFICA QUE HA PASADO 1 MS MAS DESDE LA ULTIMA ENTRADA
              MOVF       CUENTA,W          ;ASIGNO EL VALOR DEL
ACUMULADOR A LA CUENTA
              XORLW MAX                    ;COMPARO EL VALOR DE LA
CUENTA CON 20 MS (LO QUE DURA EL PERIODO DE LOS SERVOS)
              BTFSS      STATUS,Z          ;CUENTA LLEGO AL
MAXIMO?, HAN PASADO 20 MS???
              GOTO       VOLVER            ;NO, CONTINUA
CHECANDO EL PROGRAMA
              BSF         SERV01           ;SI, ES UN NUEVO
CICLO, DEBEMOS DE INICIAR CON FLANCO, PRENDIENDO EL SERVO
              CLRF       CUENTA            ;LIMPIO LA CUENTA
              CLRF       CUENTA2           ;LIMPIO LA CUENTA

VOLVER        MOVF       S_BAK,W           ;ASIGNO EL VALOR DEL
ACUMULADOR ANTERIOR
              MOVWF STATUS                 ;RECUPERA SU VALOR STATUS
              MOVF       W_BAK,W           ;RECUPERA SU VALOR EL
ACUMULADOR
              RETFIE                       ;REGRESO DE
LA INTERRUPCION

END           ;FIN

```

**Programa que controla al elevador**

```

;PROGRAMA QUE CONTROLA AL ELEVADOR
    ERRORLEVEL -302
    INCLUDE      "P16F877.INC"

;REGISTROS DE RAM GENERAL
    CBLOCK      0X20
        CUENTA          ;LUGAR DONDE SE LLEVA LA
CUENTA DE TIEMPO PARA SABER SI YA PASO TODO EL PERIODO DE PWM (20 ms)
        CUENTA2         ;LUGAR DONDE SE LLEVA LA
CUENTA DE TIEMPO PARA SABER SI YA PASO TODO EL PERIODO DE TRABAJO
        W_BAK           ;LUGAR DONDE SE CREA UN
RESPALDO DEL PWM
        S_BAK           ;LUGAR DONDE SE RESPALDA EL
ESTATUS
        PWM1            ;LUGAR DONDE SE VE EL VALOR
ACTUAL DEL PWM
        ESTADO          ;LUGAR DONDE SE ASIGAN EL
ESTADO DE LA PINZA
        TPRE            ;LUGAR DONDE SE VA A LLEVAR
EL TIEMPO PARA SABER CUANTO SE LLEVA PRECIONANDO EL BOTON
        MASCARA         ;LUGAR DONDE SE GUARDARA
CUANDO SE APLIQUE UNA MASCARA
        TRASH           ;LUGAR DONDE SE GUARDA LA
BASURA DE LAS COMPARACIONES
        TIME            ;LUGAR DONDE SE GUARDA EL
TIEMPO TRANSCURRIDO
        TMP0            ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        TMP1            ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        TMP2            ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        VALOR1          ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        VALOR2          ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
        VALOR3          ;LUGAR DONDE SE GUARDAN LOS
VALORES DE LOS DELAYS
    ENDC

#DEFINE  SENSORD  PORTB,0      ;SENSOR DERECHA
#DEFINE  SENSORC  PORTB,1      ;SENSOR CENTRO
#DEFINE  SENSORI  PORTB,2      ;SENSOR IZQUIERDA
#DEFINE  SERV01   PORTD,6      ;SALIDA PARA EL SERVO

; DEFINICION DE CONSTANTES PARA EL MOVIMIENTO DE LA
MAX      EQU      .200        ;VALOR PARA EL VALOR DEL CICLO DE
TRABAJO DE  LOS SERVOS (20 [ms]) EN ESCALA DE 0.1 [ms]
MIN1     EQU      .97         ;VALOR MINIMO PARA EL PERIODO DE
TRABAJO DE LOS SERVOS (0.97 [ms]) EN ESCALA DE 0.01[ms]
REQUERIDO EQU      .145       ;VALOR QUE QUEREMOS PARA EL
MOVIEMIENTO DE 40°, 1.45[ms] EN ESCALA DE 0.01[ms]

;ORGANIZACIÓN DE PROGRAMA
    ORG  0X00
    GOTO INI          ;VE AL INICIO DEL PROGRAMA, RUTINA DE
INICIO DE PROGRAMA

    ORG  0X04

```

```

        GOTO INTER          ; LLAMADA A LA INTERRUPCION, NOS VAMOS A LA
RUTINA INTERRUPCION

; CONFIGURACIÓN DEL SISTEMA

INI    BSF          STATUS, RP0  ; CAMBIO A BANCO1 PARA CONFIGURAR LOS
PARAMETROS NECESARIOS
        CLRF          TRISD          ; CONFIGURO PORTD COMO SALIDA
        MOVLW .255          ; MUEVO VALOR DE ENTRADA PARA PUROS
UNOS,
        MOVWF TRISB          ; ASIGNO EL VALOR PARA QUE PUERTO B
SEA ENTRADA

; CONFIGURACION DEL TIMER0

        MOVLW B'00000000'          ; MOVIENDO LA CONFIGURACION DEL TIMER
        MOVWF OPTION_REG          ; ASIGNO LA CONFIGURACION
        BCF          STATUS, RP0  ; CAMBIO AL BANCO 0, YA CONFIGURE TODO LO
QUE DEBIA EN EL BANCO 1

; INICIALIZACION DE VARIABLES

        CLRF          TIME          ; LIMPIO EL TIME QUE SIRVE PARA EL
SERVO
        CLRF          ESTADO          ; ASIGNO EL ESTADO INICIAL DE LA
PINZA, COMO ES CERO ESTA ABAJO
        CLRF          CUENTA          ; INICIALIZO REGISTRO DE CONTEO DE
EVENTOS, INICIAMOS EN 0 LO CUAL SON 0 MS TRANSCURRIDOS
        CLRF          CUENTA2          ; LIMPIO UNA DE LAS CUENTAS DEL
SERVO
        MOVLW MIN1          ; CARGO EL VALOR MINIMO EN EL ACUMULADOR,
ESTE VALOR ES PARA QUE LA PINZA NO SE LEVANTE, EL ANCHO DE PULSO ES DE
1[MS]
        MOVWF PWM1          ; MUEVE EL VALOR PARA QUE EL PWM ARRANQUE
EN LA POSICION DE 0°
        BSF          SERV01          ; EL VALOR DEL PWM SIEMPRE
INICA EN FLANCO ALTO

; CONFIGURACION DEL TIMER0 PARA EL SERVO
        MOVLW .236          ; MUEVO EL VALOR DEL TIMER0
        MOVWF TMR0          ; ASIGNO EL VALOR DEL TIMER0 PARA QUE
OCURRA UNA INTERRUPCION CASA 0.01[ms]
        BCF          INTCON, T0IF          ; LIMPIO LA BANDERA DE INTERRUPCION
        BSF          INTCON, T0IE          ; DOY EL PERMISO DE INTERRUPCION, ENCIENDO
EL TIMER0

; CONFIGURACION DE INTERRUPCIONES

        BSF          INTCON, GIE          ; PERMISO GLOBAL DE INTERRUPCION
        BSF          INTCON, PEIE          ; PERMISO DE INTERRUPCIÓN POR PERIFERICO

; PROGRAMA GENERAL

PROGRAMA    MOVLW B'000000111'          ; MUEVO ESTE VALOR DE
DISCRIMINACION, YA QUE ESOS SON LOS UNICOS VALORES QUE NOS IMPORTAN
        MOVWF MASCARA          ; MUEVO EL VALOR A LA
MASCARA

        MOVF          PORTB, W          ; MUEVO EL VALOR
DEL PUERTO AL ACUMULADOR PARA COMPARARLO

```

```

                                ANDWF      MASCARA,W      ;APLICO LA MASCARA,
PARA SOLO RECIBIR LOS VALORES DE LOS SENSORES
                                MOVWF TRASH                ;MUEVO EL VALOR AL
LUGAR DE COMPARACION TRASH,
                                MOVLW B'00000000'         ;MUEVO EL VALOR PARA
SABER SI NO VE NINGUN SENSOR
                                XORWF TRASH,F             ;COMPARO EL VALOR
                                BTFSC      STATUS,Z        ;NO ESTA VIENDO
LA LINEA NINGUN SENSOR???
                                GOTO       LEVANTAR         ;SI, VE A LA
SUBROUTINA LEVANTAR

GOTO PROGRAMA

LEVANTAR      BTFSC      ESTADO,0      ;ESTAMOS EN EL ESTADO
INICIAL, CON LA PINZA ABAJO??
                                GOTO       PROGRAMA        ;NO, ENTONCES ESTAMOS
YA CON LA PINZA LEVANTADA, VE CHECA EL PROGRAMA
PWM           INCF       PWM1           ;QUE INCREMENTE EL PWM
                                MOVLW .255                ;VALORA QUE SE LE
VA A MOVER AL DELAY
                                MOVWF VALOR1              ;VALOR DEL DELAY
                                MOVLW .4                  ;VALORA QUE SE LE
VA A MOVER AL DELAY
                                MOVWF VALOR2              ;VALOR DEL DELAY
                                MOVLW .31                 ;VALORA QUE SE LE
VA A MOVER AL DELAY
                                MOVWF VALOR3              ;VALOR DEL DELAY
                                GOTO       DELAY            ;ME VOY A UN

RETARDO
SIGO          MOVLW REQUERIDO            ;COMPARO A VER SI YA LLEGAMOS
AL MAXIMO INCREMENTO
                                XORWF PWM1,TRASH           ;COMPARO EL VALOR DE
ACULADOR CON PWM1
                                BTFSS      STATUS,Z        ;YA ALCANZO EL
VALOR REUQUERIDO????
                                GOTO       PWM              ;NO, VE Y SIGUE
INCREMENTANDO
                                BSF        ESTADO,0        ;YA ESTAMOS EN EL
ESTADO FINAL, CON LA PINZA LEVANTADA
                                GOTO       PROGRAMA        ;AHORA CHECA SI CAMBIO
DE ESTADO EL PROGRAMA

;RETARDO
DELAY         MOVF        VALOR1,W
                                MOVWF TMP0
DEL0          MOVF        VALOR2,W
                                MOVWF TMP1
DEL1          MOVF        VALOR3,W
                                MOVWF TMP2
DEL2          DECFSZ      TMP2,F
                                GOTO       DEL2
                                DECFSZ      TMP1,F
                                GOTO       DEL1
                                DECFSZ      TMP0,F
                                GOTO       DEL0
                                GOTO       SIGO

```

;SERVICIO DE INTERRUPCIÓN

```

INTER      MOVWF W_BAK          ;RESPALDO DEL VALOR ACTUAL DEL
ACUMULADOR

      MOVF      STATUS,W          ;ASIGNO EL VALOR DE
ESTATUS AL ACUMULADOR
      MOVWF S_BAK          ;CREO EL RESPALDO DE ESTATUS

      MOVLW .236          ;ASIGNO EL VALOR DEL
TIMER0 PARA VOLVERLO A CONFIGURAR
      MOVWF TMR0          ;ASIGNO EL VALOR DEL TIMER0
PARA QUE ENTRE A LA INTERRUPCION CADA 0.01[MS]
      BCF      INTCON,T0IF ;LIMPIO LA BANDERA DE
INTERRUPCION

      INCF      CUENTA2          ;INCREMENTO LA CUENTA2
      MOVF      CUENTA2,W      ;CARGA EL VALOR DE CUENTA EN
EL ACUMULADOR
      XORWF PWM1,W          ;COMPARO EL VALOR REQUERIDO
CON LA CUENTA HASTA EL MOMENTO
      BTFSS     STATUS,Z          ;EL VALOR DEL PWM
REQUERIDO ES INGUAL AL VALOR DEL LA CUENTA QUE LLEVAMOS???
      GOTO      CONTINUA          ;NO, NO HA PASADO
COMPLETO EL PERIODO DE TRABAJO, CONTINUA CHECANDO LA INTERRUPCION
      BCF      SERV01          ;SI, YA TENEMOS
EL PWM REQUERIDO, APAGO EL PUERTO
      GOTO      VOLVER          ;REGRESA A LA
ROUTINA.DEL PROGRAMA

CONTINUA   INCF      TIME          ;INCREMENTO TIME QUE LLEVA LA
CUENTA DEL TIEMPO QUE HA PASADO POR LA INTERRUPCION
      MOVLW .10          ;EL VALOR PARA SABER SI
YA PASO 10 VECES
      XORWF TIME,W          ;YA PASO 10 VECES???.
      BTFSS     STATUS,Z          ;YA PASÓ 1 [ms] DESDE
LA ULTIMA VEZ QUE ENTRASTE???
      GOTO      VOLVER          ;NO, VAMONOS A LA
ROUTINA DEL PROGRAMA
      CLRF      TIME          ;SI, ENTONCES
CONTINUAMOS.Y LIMPIAMOS TIME
      INCF      CUENTA,F          ;CUENTA = CUENTA + 1,
LO CUAL SE IGNIFICA QUE HA PASADO 1 MS MAS DESDE LA ULTIMA ENTRADA
      MOVF      CUENTA,W          ;ASIGNO EL VALOR DEL
ACUMULADOR A LA CUENTA
      XORLW MAX          ;COMPARO EL VALOR DE LA
CUENTA CON 20 MS (LO QUE DURA EL PERIODO DE LOS SERVOS)
      BTFSS     STATUS,Z          ;CUENTA LLEGO AL
MAXIMO?, HAN PASADO 20 MS???
      GOTO      VOLVER          ;NO, CONTINUA
CHECANDO EL PROGRAMA
      BSF      SERV01          ;SI, ES UN NUEVO
CICLO, DEBEMOS DE INICIAR CON FLANCO, PRENDIENDO EL SERVO
      CLRF      CUENTA          ;LIMPIO LA CUENTA
      CLRF      CUENTA2          ;LIMPIO LA CUENTA

VOLVER     MOVF      S_BAK,W          ;ASIGNO EL VALOR DEL
ACUMULADOR ANTERIOR
      MOVWF STATUS          ;RECUPERA SU VALOR STATUS
      MOVF      W_BAK,W          ;RECUPERA SU VALOR EL
ACUMULADOR
      RETFIE          ;REGRESO DE
LA INTERRUPCION
END      ;FIN

```

## Conceptos Básicos

Microsoft Visual Basic es un sistema de desarrollo diseñado especialmente para crear aplicaciones con interfaz gráfica, de una forma rápida y sencilla. Para soportar este tipo de desarrollos, Visual Basic utiliza fundamentalmente dos herramientas, una que permite realizar los diseños gráficos y un lenguaje de alto nivel.

Visual Basic incluye como características más sobresalientes las siguientes:

- Una biblioteca de clases que da soporte a los objetos de Windows tales como ventanas, cajas de diálogo, controles.
- Un control que permite utilizar las cajas de diálogo más comúnmente utilizadas.
- Un entorno de desarrollo integrado (editor de texto, intérprete, depurador, examinador de objetos, explorador de proyectos, compilador, etc.). Visual Basic fue diseñado para ser un intérprete, lo que favorece la creación y la depuración de una aplicación.
- El editor de textos ayuda a completar cada una de las sentencias visualizando la sintaxis correspondiente a las mismas.
- Asistentes para el desarrollo de aplicaciones: asistente para aplicaciones (que permite guardar sus configuraciones como perfiles para su uso posterior de forma que pueda crear múltiples aplicaciones con el mismo formato), para barras de herramientas (permite crear barras de herramientas personalizadas), para formularios de datos, para empaquetado y distribución (permite distribuir su aplicación en un servidor Web, en un espacio compartido de red o bien en otra carpeta), para crear la interfaz pública controles ActiveX, para páginas de propiedades, para objetos de datos, generador de clases, diseñador de complementos y para migración de documentos ActiveX.
- Galería de objetos incrustados y vinculados (OLE), esto es, software autocontenido en pequeñas y potentes unidades o componentes software para reutilizar en cualquier aplicación.



- Visualización y manipulación de datos de otras aplicaciones Windows utilizando los controles OLE.
- Una interfaz para múltiples documentos (MDI) que permite crear una aplicación con una ventana principal y múltiples ventanas de documento.
- Editar y continuar. Durante una sesión de depuración, se pueden realizar modificaciones al código de la aplicación sin tener que salir de dicha sesión.
- Creación y utilización de bibliotecas dinámicas (DLL).
- Soporte para la programación de aplicaciones para Internet.
- Soporte para el estándar COM (componente de software).
- Acceso a base de datos a través del control de datos ADO, utilizando el motor de Access o controladores ODBC.
- Acceso a bases de datos utilizando OLE DB como un proveedor de datos y objetos ADO como tecnología de acceso a datos.
- Biblioteca para SQL que permite manipular bases de datos relacionales.
- Un administrador visual de datos para manipular bases de datos.
- Un programa para añadir ayuda en línea; esta herramienta permite la creación de ficheros de ayuda estilo Windows.

### Conceptos básicos en Visual Basic

La unidad fundamental de una aplicación en Visual Basic es el formulario. Un formulario es realmente una ventana sobre la que se dibujan otros objetos llamados controles, tales como etiquetas, cajas de texto, marcos, casillas de verificación, botones de opción, botones de pulsación, etc., con fines de aceptar, ejecutar o visualizar datos.

El código de una aplicación Visual Basic se edita en módulos. Los módulos tienen incorporadas funciones automáticas de formato y comprobación de sintaxis. Hay tres tipos de módulos: de formulario (.frm), estándar (.bas) y de clase (.cls).

La base de una aplicación en Visual Basic la forman sus procedimientos conducidos por eventos. Un procedimiento conducido por evento es el código que es invocado cuando un objeto reconoce que ha ocurrido un determinado evento.

Una función es un procedimiento que cuando se ejecuta devuelve un resultado. En su sintaxis encontramos:

- nombre de la función y su tipo determina el tipo de datos que devuelve una función.
- parámetros que son una lista de variables que corresponden con los argumentos que son pasados cuando es invocada la función.
- expresión que define el valor devuelto por una función y se almacena en el propio nombre de la función, que actúa como variable dentro del cuerpo de la misma.

A diferencia de las funciones un procedimiento no puede ser utilizado en una expresión, ya que un procedimiento no retorna un valor a través de su nombre.

Una sentencia es una línea de texto que indica una o más operaciones a realizar. Una línea puede tener varias sentencias, separadas unas de otras por dos puntos.

Una estructura o registro es un nuevo tipo de datos definido por el usuario que puede ser manipulado de la misma forma que los tipos predefinidos. Una estructura puede definirse como una colección de datos de diferentes tipos evidentemente relacionados.

Un objeto es una entidad que tiene atributos particulares, los datos o propiedades, y unas formas de operar sobre ellos, los métodos o procedimientos. Cada objeto en Visual Basic está definido por una clase, entendiendo por clase un tipo de objetos.

**Programa de selección de producto**

```
Public DatoRecibido As String      'AQUI GUARDAMOS EL DATO QUE RECIBE
EL PROGRAMA CUANDO EL MONTACARGAS REGRESA
```

```
Private Sub cmdCerrar_Click()      'PARA QUE FUNCIONE EL BOTON DE CERRAR
QUE IMPLEMENTAMOS
    Form_Unload (2)                'NOS VAMOS A LA SUBROUTINA DE DESCARGA
End
End Sub
```

```
Private Sub ComProducto1_Click()
    MSComm1.Output = "1"           'ENVIAMOS UN 1 AL
MICROCONTROLADOR PARA QUE SE VAYA POR EL PRODUCTO 1
    ComProducto1.Enabled = False
    ComProducto2.Enabled = False
End Sub
```

```
Private Sub ComProducto2_Click()
    MSComm1.Output = "2"           'ENVIAMOS UN 2 AL
MICROCONTROLADOR PARA QUE SE VAYA POR EL PRODUCTO 2
    ComProducto1.Enabled = False
    ComProducto2.Enabled = False
End Sub
```

```
Private Sub Form_Load()

    'CONFIGURACIÓN DE PUERTO SERIAL DE COMUNICACIÓN
    MSComm1.CommPort = 1           'COM2
    MSComm1.Settings = "115200,N,8,1" '115200 BAUD, SIN PARIDAD, 8
BITS
    MSComm1.RThreshold = 1         'INTERRUPCIÓN POR RECEPCIÓN
    MSComm1.InputLen = 1           'DATOS A RECIBIR PARA
INTERRUMPIR
    If MSComm1.PortOpen = False Then MSComm1.PortOpen = True      'ABRE
EL PUERTO SI ESTÁ CERRADO

    MSComm1.Output = "0"           'ENVIAMOS UN DATO POR LA
SALIDA PARA QUE EL MONTACARGAS ESPERE SEÑAL
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
    'CERRAMOS EL PUERTO AL SALIR DEL PROGRAMA
    If MSComm1.PortOpen = True Then MSComm1.PortOpen = False
End Sub
```

```
Private Sub MSComm1_OnComm()      'ROUTINA QUE PASA CUANDO EL
MONTACARGAS HA TERMINADO SU TAREA
```

```
If MSComm1.CommEvent = comEvReceive Then      'PREGUNTA SI HAY
INTERRUPCION POR RECEPCION DE DATO, YA SABEMOS CON ESTO QUE EL
MONTACARGAS
TERMINÓ
    ComProducto1.Enabled = True                'HABILITAMOS LOS BOTONES
PARA QUE SE VUELVAN A MANDAR AL MONTACARGAS
    ComProducto2.Enabled = True
End If
End Sub
```

---



---

**Características generales:**

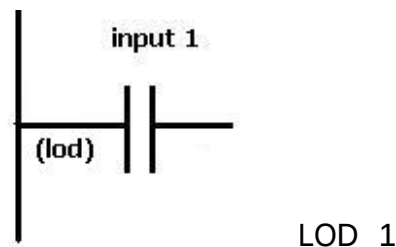
El Micro 1 cuenta con 8 entradas y 6 salidas de fabricación, y a este elemento se le puede añadir un módulo con 8 entradas y 6 salidas mas. El número de las entradas y salidas es el siguiente:

|          | PLC     | Expansión |
|----------|---------|-----------|
| Entradas | 0-7     | 10-17     |
| Salidas  | 200-205 | 210-215   |

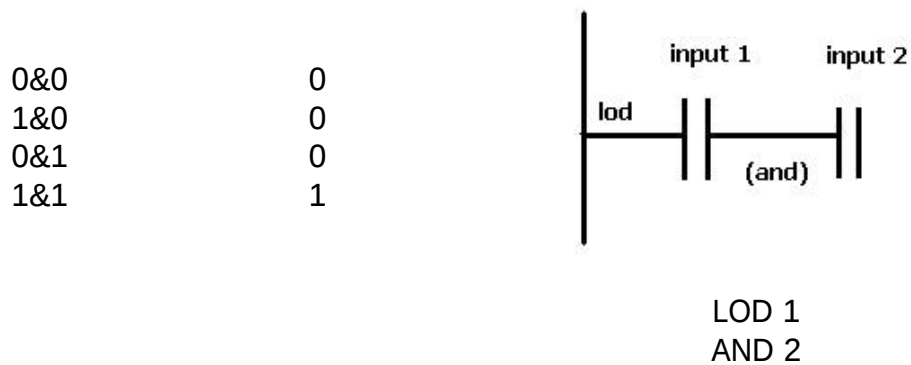
Así mismo, cuenta con relevadores internos, del número 400 al 597 yendo del 0 al 7, es decir, no hay relevadores con terminación 8 y 9. Los relevadores están ordenados en 8 bits y suman 160 en total.

Los PLC's utilizan como técnica de programación el lenguaje en escalera y las instrucciones de mayor relevancia dentro de este proyecto son:

LOD (load): da inicio a una entrada de condiciones.

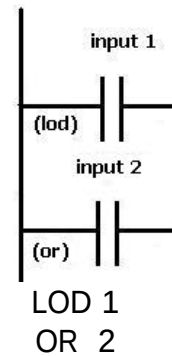


AND: encadena las condiciones de la siguiente forma:

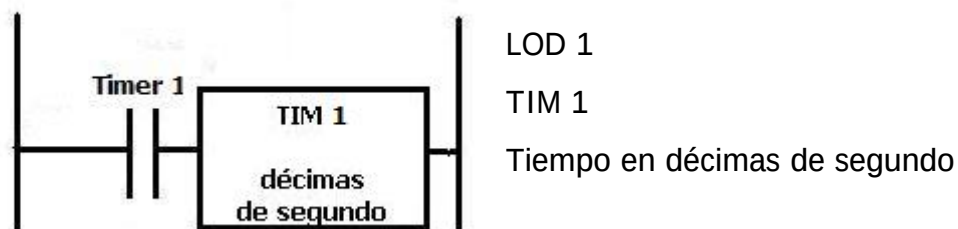


OR: Empalma las condiciones para que se cumplan de la siguiente manera:

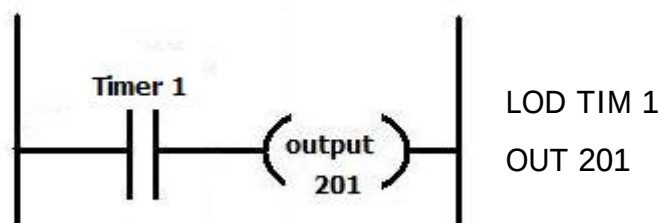
|      |   |
|------|---|
| 0  0 | 0 |
| 1  0 | 1 |
| 0  1 | 1 |
| 1  1 | 1 |



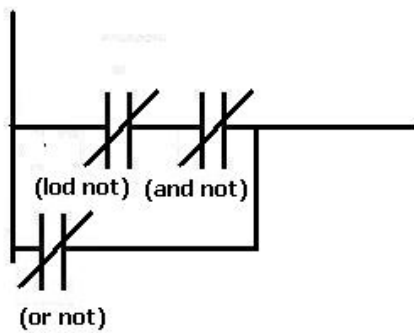
TIM: Activa un temporizador que trabaja en décimas de segundo y que se activa si se cumplen las condiciones que estén en su entrada. Durante su operación son utilizados en otras líneas de comandos como entradas del programa. Hay 80 temporizadores, del 0 al 79 y realizan una cuenta desde 0.1[s] hasta 100[s].



De esta forma se puede utilizar el timer para arrancar nuevas salidas:

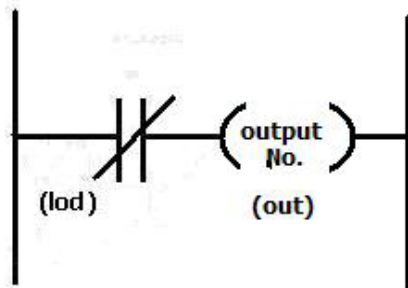


NOT: Niega la condición u operación.



LOD NOT 1  
AND NOT 2  
OR NOT 3

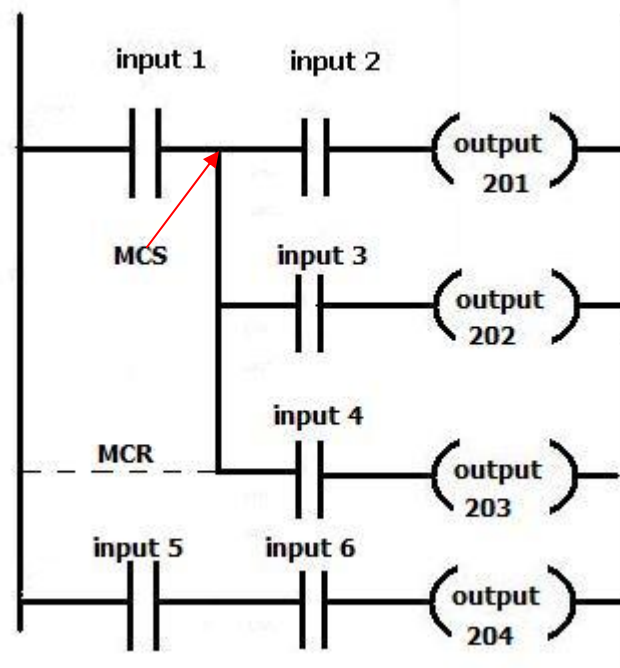
OUT: Son las salidas del sistema, debe ser una sola por línea de instrucciones, ir al final de todos los comandos y no se deben de repetir.



LOD input No.  
OUT output No.

MCS (master control set): Encadena una condición para una serie de escalones

MCR (master control reset): Regresa un escalón, es decir, cierra el MCS



LOD 1  
MCS  
LOD 2  
OUT 201  
LOD 3  
OUT 202  
LOD 4  
OUT 203  
MCR  
LOD 5  
AND 6  
OUT 204

Durante la ejecución del programa se puede monitorear el comportamiento de las salidas, relevadores, como también de los temporizadores con la instrucción MON y en seguida la operación a monitorear su número y finalmente pulsar READ.

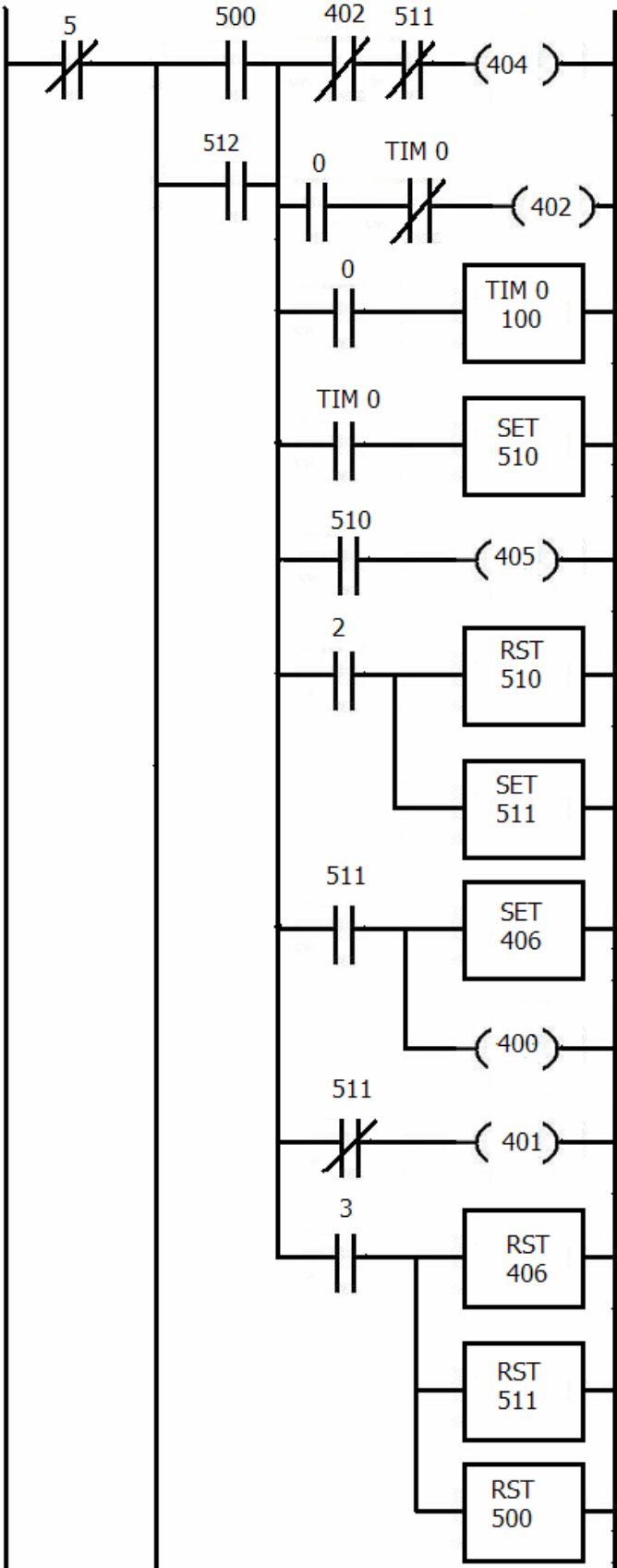
Ejemplo:

: MON 400 READ

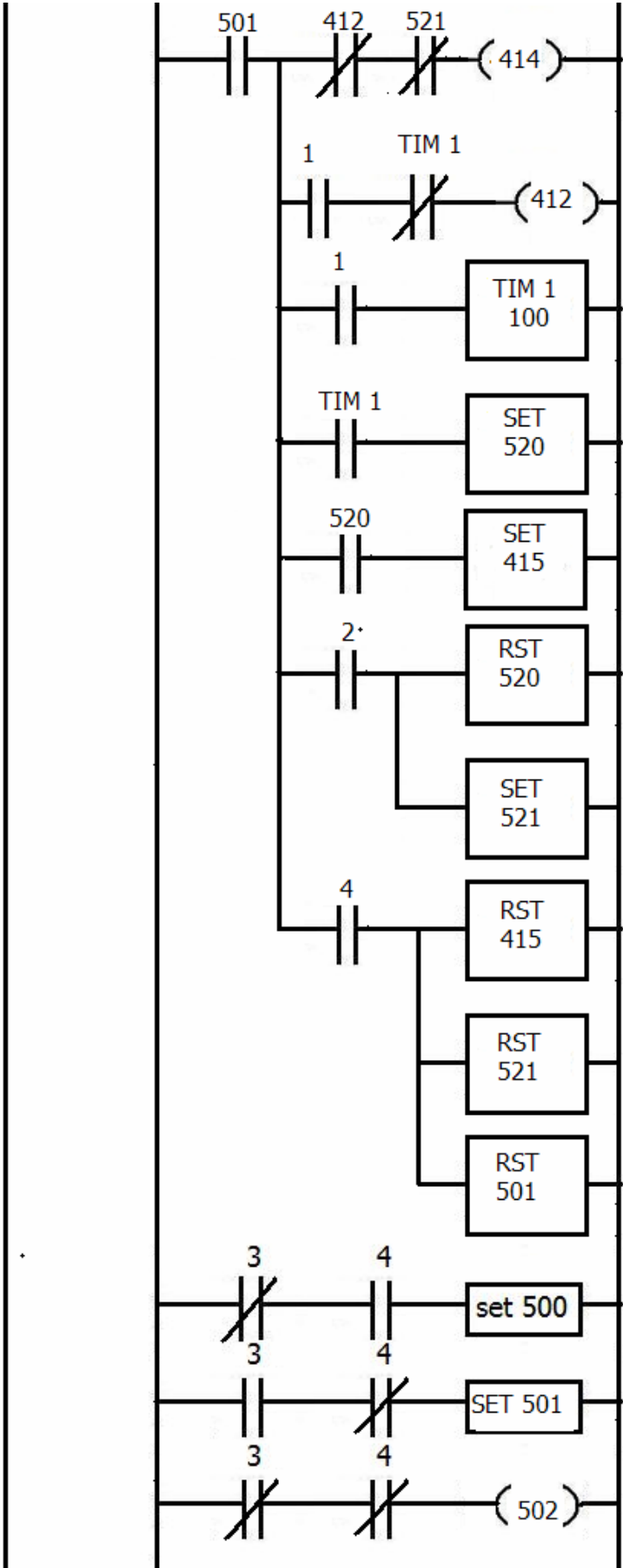
: MON (shift) TIM 1 READ



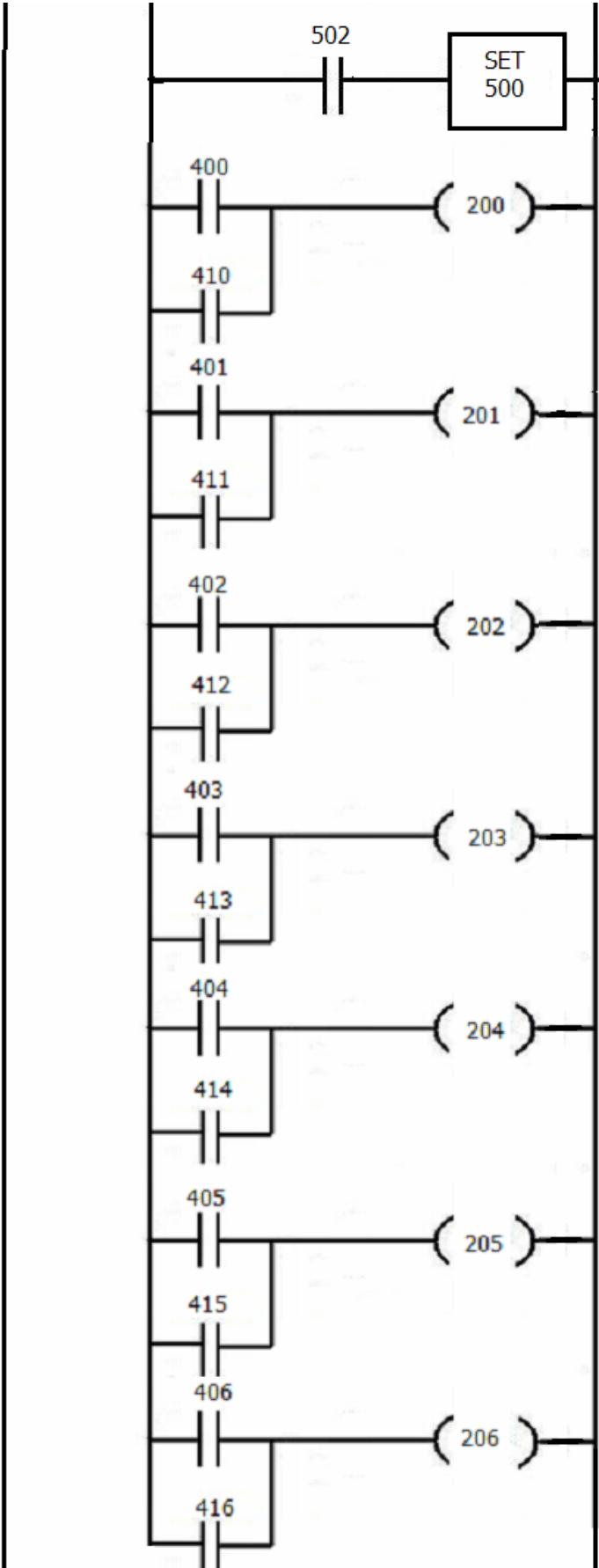
Programa cargado al PLC  
1ª Parte



2ª Parte



3ª Parte



4ª Parte

