Facultad de Ingeniería Eléctrica
Especialidad Automática

Trabajo de Diploma

Título: *Control secuencial de un manipulador neumático utilizando PLCs.*

Diplomante: *Elides Medina Frómeta.*

Tutores: *Dr. Israel Benítez Pina.*
Dr. Orlando Obregón Pacheco.

Consultantes: MSc. Luis García Faure.
MSc. Luisa Villafruela Loperena.

Curso 2001-2002

Resumen.

El uso de los dispositivos electroneumáticos en la Automática ocupa un lugar importante dentro de los medios de automatización actuales. Estos forman parte de los accionamientos presentes en Robótica, Sistemas de Manufactura flexible, etc

En el presente trabajo se describe cómo se realiza el montaje y puesta en marcha de una instalación de accionamiento y supervisión de un manipulador neumático con ventosa utilizando un PLC. Para esto se parte de un estudio previo sobre el funcionamiento de los principales componentes de los circuitos neumáticos, así como de los autómatas programables. Además se explica cómo se logra la visualización remota de esta instalación desde varios ordenadores.

Se detallan los programas para la simulación y supervisión en tiempo real, realizados en el lenguaje LD y el G del LabView.

Esta instalación permite el entrenamiento de nuestros estudiantes de pre y postgrado en la utilización de dispositivos electroneumáticos gobernados por PLCs y PCs dentro de sistemas de automatización, lo cual no se disponía en nuestros laboratorios de Automática.

Introducción

La Robótica es un componente esencial de la automatización de la producción, que afecta a la mano de obra humana a todos los niveles, desde los trabajadores no especializados hasta los técnicos profesionales y directores de producción. Los futuros robots pueden encontrar aplicaciones fuera de la fábrica en bancos, restaurantes e incluso en los propios hogares. Es posible, y quizá probable, que la Robótica llegue a ser un campo, como el de la tecnología informática actual, que invada nuestra sociedad.

Actualmente los robots son manipuladores mecánicos muy automatizados, donde el control se realiza por computadoras u otros dispositivos programables.

La automatización y la robótica son dos tecnologías estrechamente relacionadas. En un contexto industrial podemos definir la automatización como una tecnología que está relacionada con el empleo de sistemas mecánicos, electrónicos y basados en computadoras en la operación y control de la producción. En consecuencia, la robótica es una forma de automatización industrial. Hay tres clases amplias de automatización industrial: automatización fija, automatización programable y automatización flexible.

El presente proyecto describe la realización de la automatización programable de un manipulador neumático utilizando para ello el autómata TSX17-20 de la firma francesa TELEMECANIQUE, así como la visualización remota del proceso desde una red de área local (LAN), utilizando para ello el Labview, una poderosa herramienta que proporciona la National Instruments para generar aplicaciones de control de procesos.

El propósito de la instalación tanto de lógica cableada como de lógica programada, no es solo que el alumno conozca la base de la automatización, sino que comprenda el gran avance que supone la automatización y la incorporación de la lógica programada en la industria.

Para una mayor comprensión sobre el funcionamiento del manipulador se introducirá primero el concepto de Neumática y Electroneumática como base para lograr la creación de una maqueta de laboratorio con este manipulador. El proyecto concluye con el montaje de este manipulador en una instalación.

Los objetivo del trabajo fueron:

- El estudio del funcionamiento de los componentes de automatización presentes en la instalación de laboratorio.
- El montaje de los componentes neumáticos y electroneumáticos que forman parte de la instalación y la organización de éstos, para asegurar su correcto funcionamiento.
- Automatización de la instalación con un PLC TSX17-20, de la firma francesa TELEMECANIQUE.
- La visualización del proceso en una PC utilizando para ello el Labview.
- La visualización remota desde una red de área local (LAN).

Este último no era realmente un objetivo del trabajo pero con el uso tan difundido que está teniendo en nuestros días, se decidió incorporarlo dentro del Sistema Docente de Automatización Industrial (SDAI) en desarrollo dentro de nuestro laboratorio.

La maqueta ha sido montada en el laboratorio de control de procesos ubicado en el segundo piso de la Facultad de Ingeniería Eléctrica de la Universidad de Oriente, y tendrá otra versión igual instalada en el laboratorio de mecánica de los fluidos de la Facultad de Ingeniería Mecánica, ubicado en la sede central de esta Universidad. Esto obedece a que este trabajo surge como colaboración con dicha facultad para utilizar la donación electroneumática de la firma japonesa SMC que ellos recibieron.

La estructura del trabajo es la siguiente:

En el capítulo 1 se comenzará introduciendo el concepto de Neumática para introducir el de Electroneumática, con el objetivo de realizar una descripción de los componentes que intervienen en la neumática: cilindros, válvulas y otros.

Para comprender el principio de funcionamiento de los elementos neumáticos es imprescindible conocer algunas de las propiedades del aire comprimido, elemento básico de un circuito neumático. Además en este capítulo se incluye la explicación del funcionamiento manual de la instalación de laboratorio.

En el capítulo 2 se trata la necesidad de automatizar el proceso, el ISaGRAF como lenguaje de simulación, así como de las características del PLC utilizado con este propósito y de su lenguaje de programación, terminando con la descripción del proceso automatizado.

En el capítulo 3 y final se detallan las principales características del Labview que lo hacen una herramienta efectiva para la visualización de aplicaciones de control de procesos, así como del DataSocket para la visualización remota. Este capítulo concluye con la explicación del trabajo realizado con los mismos.

De forma general este trabajo ha sido realizado para crear las condiciones en nuestro Laboratorio de Automática para permitir la familiarización de nuestros estudiantes con los componentes básicos de los circuitos neumáticos y con el uso de los PLCs para su automatización, los cuales no se disponían anteriormente. Esto permite incentivar su creatividad e interés por la automatización industrial que es en definitiva la vida de nosotros los automáticos, a la vez que los forma integralmente para su futura profesión.

1 La Neumática en la Automatización.

1.1 Introducción.

En esta parte, se comenzará introduciendo el concepto de Neumática como base para introducir el de Electroneumática, concepto más moderno pero fundamentado en la neumática tradicional. Para comprender el principio de funcionamiento de los elementos neumáticos es imprescindible conocer algunas de las propiedades del aire comprimido, elemento básico de un circuito neumático.

Una vez conocidos los fundamentos teóricos de la Neumática y de Electroneumática se realizará una descripción de los elementos que intervienen en la Neumática: cilindros, válvulas y otros.

En la actualidad la aplicación de los principios de la Neumática son de vital importancia para la industria ya que la utilización de los manipuladores que trabajan bajo este principio hacen más eficiente, cómodo y rápido el trabajo a realizar. Entre los principales componentes que realizan esta función se encuentran los cilindros neumáticos, las válvulas electroneumáticas, equipos de vacío como los eyectores entre otros que son los principales exponentes de este amplio conjunto.

Desde hace algunos años, la utilización de equipos de vacío en las automatizaciones industriales han aumentado de forma considerable. Los diseños más compactos de los componentes, sus rendimientos, la limpieza del medio y la economía de los mismos ha colaborado a este incremento de utilización.

Las ventajas de la utilización de la técnica de vacío frente a otras técnicas de manipulación radica fundamentalmente en :

- Simplicidad de los componentes básicos.
- Fácil posicionamiento.
- Altas frecuencias de trabajo .
- Fácil adaptación a superficies diversas sin necesidad de mecanizados adicionales para su adaptación.

Un ejemplo de la aplicación de la Neumática a los procesos industriales es la utilización de las ventosas las cuales al adherirse al objeto a trasladar van creando un vacío capaz de levantar grandes masas con un mínimo de esfuerzo.

La técnica Neumática constituye hoy por hoy el complemento ideal de la mecánica en cualquier proceso de producción moderno. Muchos problemas de ingeniería, a lo largo de los años, han sido resueltos mediante la técnica tradicional, pero con la incorporación relativamente reciente de estas tecnologías se ha conseguido simplificar las máquinas haciendo más sencillos los procesos, a la vez que se ha logrado cierto grado de automatización de forma rápida y económica.

En la actualidad, la necesidad de automatizar la producción no afecta únicamente a las grandes empresas, sino también a la pequeña industria. Las tareas manuales deben sustituirse por la fuerza y precisión mecánica para de está manera aumentar los índices de eficiencia y calidad así como disminuir la posibilidad de accidentes fatales para la vida humana en los procesos de alto riesgo. Esto se logra automatizando las diversas labores productivas.

La energía neumática puede realizar muchas funciones mejor y más rápidamente, de forma más regular y sobre todo durante mucho más tiempo sin sufrir los efectos de la fatiga.

Entre los trabajos que puede realizar la energía neumática se pueden citar los siguientes:

- Accionamiento: empuje, tracción,...
- Elevación.
- Alimentación y expulsión de materiales.
- Transporte.
- Inspección y comprobación de medidas o cantidad de piezas.
- Mecanización.
- Operaciones de seguridad y protección.

La energía neumática no es aplicable a todos los casos de automatización. Las posibilidades técnicas de esta tecnología están sometidas a ciertas limitaciones en lo que se refiere a fuerza, espacio, tiempo y velocidad en el proceso de la información, sin embargo, la Neumática tiene su ventaja más importante en la flexibilidad y variedad de aplicaciones en casi todas las ramas de la producción industrial.

1.2 ¿Qué es la Neumática?

El concepto moderno de Neumática trata sobre los fenómenos y aplicaciones de la sobrepresión o depresión (vacío) del aire. La mayoría de las aplicaciones neumáticas se basan en el aprovechamiento de la sobrepresión.

La energía neumática, que emplea como fuente de energía el aire comprimido, tiene cualidades excelentes entre las que destacan:

- El aire es abundante y barato.
- Se transforma y almacena fácilmente.
- Es limpio, no contamina y carece de problemas de combustión con la temperatura.

Los elementos neumáticos pueden alcanzar velocidades de trabajo elevadas pero, dada la compresibilidad del aire, su regulación no es constante.

1.3 Propiedades del aire comprimido

El aire comprimido empleado en la industria es aire de la atmósfera sometido a presiones de hasta unos 12 bar (12 Kp / cm2) aproximadamente. Es una energía fácilmente transportable, pero no se recomiendan grandes distancias en su distribución debido a las pérdidas de carga que se originan en tuberías y rácores de unión.

El aire es un gas casi perfecto caracterizándose esencialmente por su fluidez, compresibilidad y elasticidad. La fluidez permite a sus partículas no ofrecer resistencia apenas al desplazamiento; la compresibilidad hace que una determinada cantidad de gas pueda reducir su volumen si éste se encuentra en un recinto herméticamente cerrado; la elasticidad permite que al comprimirlo en ese mismo recinto, ejerza sobre sus paredes una determinada presión, normal a las superficies de contacto.

Al someter a compresión el aire encerrado dentro de un cilindro, el pistón cede y, al igual que un resorte helicoidal mecánico, la deformación experimentada será directamente proporcional a la fuerza ejercida. Los mecanismos accionados con este medio son mecanismos que poseen cierta elasticidad y capacidad de amortiguamiento. Esta cualidad resulta negativa por la imposibilidad de detener la carrera del cilindro cuando éste deja de ser alimentado. Si el caudal de aire se detiene bruscamente el cilindro no para, sino que sigue avanzando hasta que las fuerzas de una y otra cámara se igualan o bien un tope mecánico lo detiene.

1.4 Elementos neumáticos.

Los elementos básicos en automatismos neumáticos son los actuadores, dentro de los cuales se hayan los cilindros, y las válvulas neumáticas.

Actuadores neumáticos:

En un sistema neumático los receptores son los llamados actuadores neumáticos o elementos de trabajo, cuya función es la de transformar la energía neumática del aire comprimido en trabajo mecánico.

Los actuadores neumáticos se clasifican en dos grandes grupos:

- Cilindros.
- Motores.

Aunque el concepto motor se emplea para designar a una máquina que transforma energía en trabajo mecánico, en Neumática sólo se habla de un motor si es generado un movimiento de rotación, aunque es también frecuente llamar a los cilindros, motores lineales.

1.4.1 Cilindros neumáticos:

El cilindro neumático es un dispositivo capaz de convertir la energía contenida en el aire comprimido en trabajo mecánico de movimiento rectilíneo, que consta de carrera de avance y carrera de retroceso.

Generalmente el cilindro está constituido por un tubo circular cerrado en los extremos mediante dos tapas, entre las cuales se desliza un émbolo que separa dos cámaras. Al émbolo va unido un vástago que, saliendo a través de una o ambas tapas, permite utilizar la fuerza desarrollada por el cilindro en virtud de la presión del fluido al actuar sobre las superficies del émbolo.

Los dos volúmenes de aire en que queda dividido el cilindro por el émbolo reciben el nombre de cámaras. Si la presión del aire se aplica en la cámara posterior de un cilindro, el émbolo y el vástago se desplazan hacia adelante (carrera de avance). Si la presión del aire se aplica en la cámara anterior del cilindro, el desplazamiento se realiza en sentido inverso (carrera de retroceso).

Según la forma en que se realiza el retroceso del vástago, los cilindros neumáticos se dividen en dos grupos:

- Cilindros de simple efecto.
- Cilindros de doble efecto.

1.4.1.1 Cilindros de simple efecto

Los cilindros de simple efecto sólo pueden realizar el trabajo en un único sentido, es decir, el desplazamiento del émbolo por la presión del aire comprimido tiene lugar en un solo sentido, pues el retorno a su posición inicial se realiza por medio de un muelle recuperador que lleva el cilindro incorporado o bien mediante la acción de fuerzas exteriores.

Según la disposición del muelle, los cilindros de simple efecto pueden aplicarse para trabajar a compresión (vástago recogido en reposo y muelle en cámara anterior) o para trabajar a tracción (vástago desplazado en reposo y muelle en cámara posterior).

Mediante el resorte recuperador incorporado, queda limitada la carrera de los cilindros de simple efecto; por regla general la longitud de la carrera no supera los 100mm. Por razones prácticas, son de pequeño diámetro y la única ventaja de estos cilindros es su reducido consumo de aire, por lo que suelen aplicarse como elementos auxiliares en las automatizaciones.

1.4.1.2 Cilindros de doble efecto

Al decir doble efecto se quiere significar que tanto el movimiento de salida como el de entrada son debidos al aire comprimido, es decir, el aire comprimido ejerce su acción en las dos cámaras del cilindro, de esta forma puede realizar trabajo en los dos sentidos del movimiento.

El campo de aplicación de los cilindros de doble efecto es mucho más amplio que el de los cilindros de simple efecto; incluso si no es necesario ejercer una fuerza en los dos sentidos, el cilindro de doble efecto es preferible al de simple efecto.

Al aplicar aire a presión en la cámara posterior del cilindro y comunicando la cámara anterior con la atmósfera a través de una válvula, el cilindro realiza la carrera de avance.

La carrera de retroceso se efectúa introduciendo aire a presión en la cámara anterior y comunicando la cámara posterior con la atmósfera, igualmente a través de una válvula para la evacuación del aire contenido en esa cámara del cilindro.

Para una presión determinada en el circuito, el movimiento de retroceso en un cilindro de doble efecto desarrolla menos fuerza que el movimiento de avance, ya que la superficie del émbolo se ve ahora reducida por la acción transversal del vástago. Normalmente, en la práctica no se requieren fuerzas iguales en los dos movimientos opuestos.

Los cilindros de doble efecto pueden ser: sin amortiguación o con amortiguación.

En la práctica el empleo de uno u otro depende de factores como la carga y la velocidad de desplazamiento. Por ejemplo, cuando la carga viene detenida por topes externos pueden aplicarse los cilindros sin amortiguación.

Sin embargo, cuando la carga ni viene detenida por tales topes se debe recurrir a la utilización de los cilindros con amortiguación.

Los cilindros de doble efecto presentan las siguientes ventajas frente a los cilindros de simple efecto:

- Posibilidad de realizar trabajo en los dos sentidos.
- No se pierde fuerza para comprimir el muelle.
- No se aprovecha toda la longitud del cuerpo del cilindro como carrera útil.

Por el contrario, tienen el inconveniente de que consumen doble cantidad de aire comprimido que un cilindro de simple efecto.

1.5 Válvulas neumáticas:

Las válvulas neumáticas son las que gobiernan el movimiento de los cilindros.

Según la función que realizan pueden ser:

1- Válvulas distribuidoras: son las que gobiernan el arranque, paro y sentido de circulación del aire comprimido.

La misión que se encomienda a los distribuidores dentro de un circuito de automatización es la de mantener o cambiar, según unas órdenes o señales recibidas, las conexiones entre los conductos a ellos conectados, para obtener unas señales de salida de acuerdo con el programa establecido. Simultáneamente actúan como transductores o como amplificadores, ya que controlan una potencia neumática con otra menor, también neumática (amplificación), o de otra naturaleza (transducción y amplificación).

De acuerdo con su uso pueden dividirse en:

- Distribuidores de potencia o principales: su función es suministrar aire directamente a los actuadores neumáticos y permitir igualmente el escape. Se sitúan lo más próximos posible a los cilindros.

- Distribuidores fin de carrera: abren o cierran pasos al aire cuya función no será la de ir directamente al actuador, sino que se utilizan solamente para accionamiento de otros mecanismos de control, tales como los distribuidores de potencia. Su situación viene fijada por el punto y la manera en que han de ser controlados.
- Distribuidores auxiliares: utilizados en los circuitos y que, en combinación con válvulas fin de carrera y de potencia, se utilizan para dirigir convenientemente las señales de presión del aire. Su colocación es independiente.

2- Válvulas reguladoras.

3- Válvulas de seguridad.

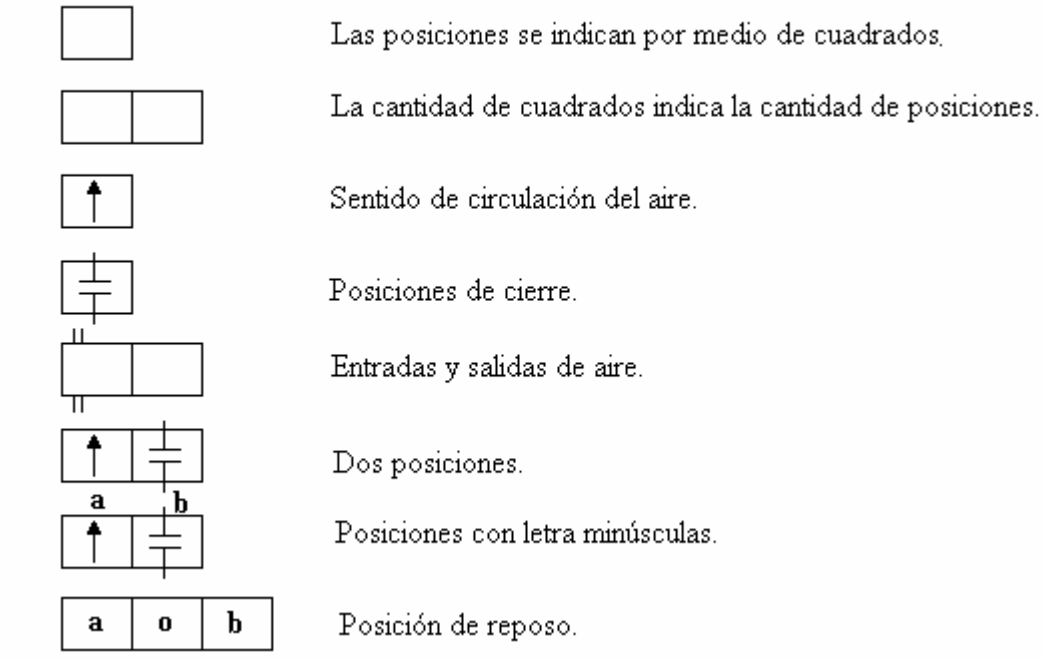
4- Válvulas de secuencia.

5- Válvulas temporizadoras.

1.5.1 Concepto de vías y posiciones

Se entiende por número de vías el número máximo de conductos que pueden interconectarse a través del distribuidor.

El número de posiciones es el de conexiones diferentes que pueden obtenerse de manera estable entre las vías del distribuidor.



Las válvulas de vías se designan en los catálogos de los fabricantes por el número de vías controladas y de posiciones de maniobra estables. Así, una válvula 3/2 vías quiere decir que posee tres vías y dos posiciones de maniobra.

Para evitar errores durante el montaje y además identificarlos, se designan con letras mayúsculas o números:

Según DIN 25300, se indica:

P = Alimentación de aire comprimido.

A, B, C = Salidas de trabajo.

R, S, T = Escape de aire.

X, Y, Z = Conexiones de mando.

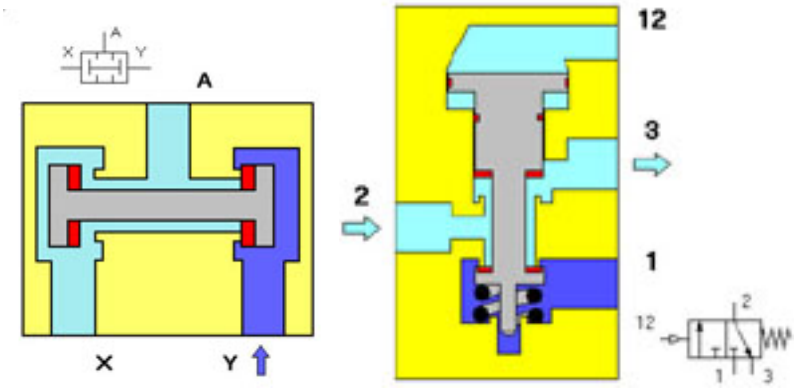
Según normas CETOP, es:

1 = Alimentación de aire comprimido.

2 y 4 = Salidas de trabajo.

3 y 5 = Escape de aire.

12 y 14 = Conexiones de mando.



Ejemplos de representación de válvulas

1.5.2 Accionamientos de los distribuidores.

Una característica importante de toda válvula es su clase de accionamiento, de acuerdo con ello, dentro de la cadena de mando de un equipo neumático se le empleará como elemento emisor de señal, órgano de control o de regulación.

Los accionamientos comprenden dos mecanismos, el de mando y el de retorno, que pueden ser distintos o iguales. Se debe tener siempre presente que para cambiar el estado de un distribuidor, es preciso, que se ejerza una acción en un solo extremo del distribuidor.

Podemos dividir los accionamientos en:

- Accionamientos mecánicos: son necesarios en todas aquellas partes en las que la válvula deba ser accionada mediante un órgano mecánico del equipo, por ejemplo: las levas en el vástago de un cilindro, carros de las máquinas... A veces, las válvulas con este dispositivo de mando actúan como finales de carrera.
- Accionamientos de fuerza muscular: por medio de este mando es posible supeditar una acción neumática a lo ordenado por el operario que se encarga de accionarla. Entre estos accionamientos figuran todos los que son realizados con la mano o con el pie.
- Accionamiento neumático: estos accionamientos utilizan aire a presión, se emplean en accionamientos a distancia. En el mando a distancia de un distribuidor el elemento emisor de señales está separado del punto de accionamiento.

El accionamiento neumático puede realizarse por impulso del aire a presión- accionamiento o pilotaje positivo o por reducción de la presión- accionamiento o pilotaje negativo. Las válvulas accionadas por medios neumáticos con posición de reposo automática (válvulas monoestables), utilizan exclusivamente pilotaje positivo, un accionamiento de este tipo se dice que es de mando permanente, y la inversión de la válvula permanece en tanto dure la presión de pilotaje. A diferencia de las anteriores, en válvulas de impulso, de inversión positiva o negativa, es suficiente una señal momentánea de duración mínima establecida para efectuar la inversión, permaneciendo la válvula en la posición de maniobra adoptada hasta que se presenta un impulso contrario (válvulas biestables).

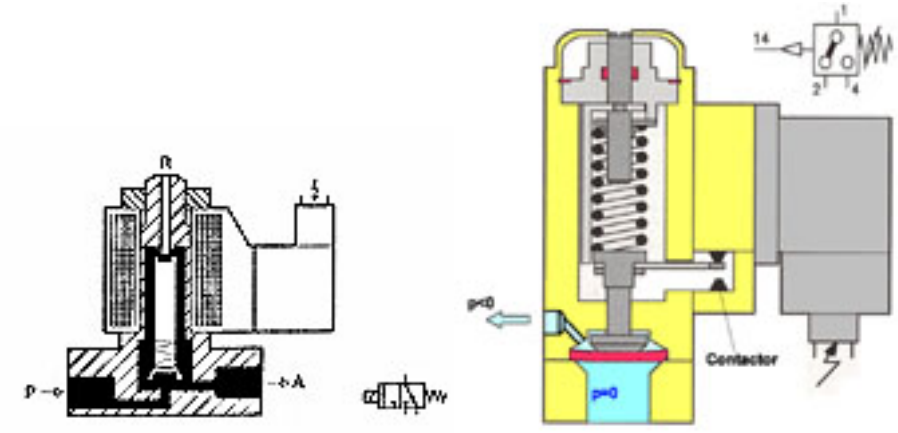
Las tuberías de mando en las válvulas de accionamiento neumático no deben ser demasiado largas, pues de lo contrario se hacen demasiado largos los tiempos de respuesta y el consumo de aire también es demasiado grande.

Accionamiento eléctrico: por medio de este mando se subordina una acción neumática por el paso de la corriente a través de un electroimán. Las válvulas provistas de este sistema de mando reciben el nombre de *válvulas magnéticas o electroválvulas*.

En el accionamiento eléctrico de una válvula, la longitud de la línea de mando es independiente de la completa eficiencia del funcionamiento pudiendo preverse líneas de mando de varios centenares de metros. Los tiempos de mando son muy cortos.

Como emisores de señales se emplean preferentemente interruptores de final de carrera, pudiendo servir además todos los dispositivos que entregan una señal eléctrica. En ambientes con peligro de explosión todos los componentes eléctricos deben tener una protección adecuada.

A continuación se muestra un esquema del funcionamiento de electroválvulas.



Esquema de funcionamiento de electroválvulas.

1.6 Los captadores

Para detectar posición, movimiento, etc. en manipuladores se utilizan distintos tipos de captadores neumáticos, eléctricos y/o electrónicos.

Los finales de carrera son muy similares en neumática y en electricidad. Sin embargo, en versiones miniaturizadas los neumáticos son más robustos.

En Neumática, como en electricidad, las funciones de los finales de carrera pueden estar integradas en los cilindros:

- En neumática se utilizan las presiones internas de los cilindros.
- En electricidad, con un cilindro especial, el émbolo magnético conmuta un contacto tipo *reed*..

Los captadores de fuga neumáticos permiten detectar un movimiento de pequeña amplitud.

Los detectores de proximidad electrónicos, en sus variantes de corriente continua, son pequeños y dan mejores resultados frente a los detectores fluídicos de proximidad, que se utilizan solo en casos especiales.

1.7 Aplicaciones industriales

Para dar una idea general de las posibilidades de aplicación de la Neumática, expondremos una lista de varios procesos industriales y sus aplicaciones; no obstante esta lista se ve ampliada constantemente debido a la investigación y desarrollo de nuevas tecnologías.

- Centrales eléctricas: dispositivos de ventilación para edificios de calderas, correderas telemanipuladas, mandos de interruptores neumáticos,...
- Centrales nucleares: entrada y salida de barras de combustible y dispositivos de frenado, cierre de compuertas, dispositivos de control y de medición.
- Abastecimiento de agua: control de nivel y servomecanismos de corredera, accionamiento de válvulas y de rejillas en instalaciones de depuradoras y de suministro.
- Industria química: dispositivos de cierre de tapas, instalaciones de dosificación, accionamiento de rodillos en mezcladores de laboratorios, dispositivos de elevación y descenso para baños, accionamiento de compuertas, mandos de balanzas, técnica de embalaje, regulaciones de nivel, dispositivos de regulación de procesos.
- Materiales para la construcción: accionamiento de moldes, cierre de silos, dispositivos de alimentación de lijadoras, multivibradores contra la formación de atascos en depósitos de arena, instalaciones de transporte,...
- Fundición: moldeadoras, dispositivos para extracción, dispositivos de transporte y de almacenamiento de cuchara, accionamientos auxiliares en máquinas de moldeo, mandos de puertas de hornos,...
- Industria de la madera: desplazamiento de rodillos en sierras alternativas, accionamiento de sierras tronzadoras, prensas de bastidor, dispositivos de alimentación,...
- Manipulación de papel y cartón: dispositivos de transporte, de sujeción, de plegado, de prensado y de empaquetado, dispositivos de empaquetado, accionamiento de prensas de corte, control de cinta,...

1.8 Descripción del proceso a controlar.

En el laboratorio de mecánica de los fluidos de la Facultad de Ingeniería Mecánica de la Universidad de Oriente y perteneciente al Centro de Estudio de Eficiencia Energética (CEEFE) existe una maqueta cuyo funcionamiento esta basado en los principios de la neumática. Basado en este mismo diseño se elaboró otra maqueta igual para el laboratorio de control automático de la Facultad de Ingeniería Eléctrica.

Esta maqueta, se encuentra en el anexo #5, cuenta con tres válvulas solenoides (electroválvulas), un eyector, un regulador de vacío, dos cilindros de doble efecto, uno de ellos con una ventosa, un detector magnético y tres microswitchs como sensores.

El aire comprimido (flujo de aire) que sale del compresor llega al mismo tiempo a las tres electroválvulas, gracias a la utilización de rácores de unión, (las tres electroválvulas poseen además de su accionamiento eléctrico uno manual que cuando es apretado realiza el mismo efecto que cuando es activado por los 24V que necesita para su alimentación). La primera de ellas del tipo VZ2130 de la firma japonesa SMC con una presión de trabajo en un rango de 1-9 Kgf/cm^2, es la encargada de suministrar el flujo de aire al eyector. Este último es un dispositivo encargado de realizar vacío extrayendo aire por una de sus entradas y expulsándola por la salida. Este vacío logra extraer aire de una ventosa lo que posibilita que se pueda adherir a algún objeto. De esta manera cuando la ventosa se une al objeto a trasladar, gracias al eyector, puede levantar el objeto primero y trasladarlo después.

Para controlar si el vacío creado es suficiente se encuentra ubicado entre el eyector y la ventosa un regulador de vacío. Este regulador es un dispositivo con un diafragma que sufre un desplazamiento según el vacío debido a que él se encuentra tomando una muestra del flujo de aire del conducto que va hasta la ventosa. Cuando ésta no ha cogido ningún objeto el diafragma se encuentra en su posición de equilibrio, o sea, arriba logrando que el obturador unido a ella se mantenga presionando a un microswitch indicando de esta manera que la ventosa esta libre. Si por el contrario la ventosa ha logrado adherirse a algún objeto el aire que se extraía de la ventosa, pasa a ser extraído del regulador provocando un desplazamiento del diafragma hacia abajo y con él del obturador, liberando así al microswitch indicando que ya tiene el objeto.

Por otro lado tenemos que el flujo de aire llega a la segunda electroválvula modelo VZ3223 también de la firma japonesa SMC con la presión de trabajo entre 1-7Kgf/cm^2 esta segunda electroválvula cuenta con dos vías de salida que van a las tomas de aire del cilindro de doble

efecto, modelo C82ADB20-125 dicho cilindro aguanta una presión de trabajo de 10 bar y una temperatura de 5/60 °C con desplazamiento vertical (en cuyo extremo se encuentra la ventosa). El sentido del movimiento del cilindro con la ventosa depende de la salida accionada en la electroválvula. Este cilindro posee un detector magnético encargado de avisar la llegada de su émbolo arriba. El mismo se encuentra unido mediante tornillos a otro cilindro también de doble efecto pero modelo C92SDB con las mismas características del anterior. Este realiza el movimiento horizontal del conjunto para la traslación de los objetos. Este último posee dos tomas de aire similar al anterior y el aire le es suministrado a través de la tercera electroválvula cuyo modelo es igual a la mencionada anteriormente y que su alimentación neumática también es proveniente del compresor.

Una vez conocidos los elementos que intervienen en el proceso pasemos a la secuencia de trabajo del mismo:

Cuando el sistema posee alimentación neumática se pueden accionar de forma manual las perillas de las electroválvulas según sea la secuencia deseada. En este caso especifico partiremos de la siguiente secuencia: se debe buscar el objeto a trasladar, en la posición abajo y derecha, luego se sube y se desplaza el conjunto a la izquierda para soltar aquí el objeto.

Esto cumple el objetivo de tomar una pequeña pelota en la parte inferior de una canaleta para llevarla hasta el inicio superior de la misma. Desde este último lugar la pelota se desliza por gravedad a través de toda la canal para retornar a su posición inicial. Esto se resuelve de forma sencilla, partiendo de la posición izquierda, es accionada la electroválvula del eyector para poder coger los objetos y dejándola así. Luego se actúa sobre la electroválvula del movimiento horizontal derecho para lograr que el conjunto se desplace por la posición superior hasta el extremo derecho. Cuando llegue allí se suelta esta y se actúa en la electroválvula del movimiento vertical hacia abajo para que el extremo del cilindro vertical con la ventosa tope al objeto (pelota). Cuando se coge el objeto, se presiona la perilla de la otra electroválvula del movimiento hacia arriba para desplazar la pelota unida a la ventosa a la posición superior derecha. Después se acciona el desplazamiento horizontal izquierdo para llegar a la posición inicial de la canaleta donde se va a soltar el objeto. En esta posición se actúa sobre el eyector para liberar el objeto de la ventosa. Luego se repite cuantas veces se desee.

Esta instalación presenta como inconveniente el accionamiento manual del sistema por eso este trabajo fue dedicado a su control mediante un PLC para evitar las molestias que ocasiona el

mismo y además permitir el estudio del accionamiento totalmente automatizado tanto por los estudiantes de Ingeniería Mecánica como por los de Ingeniería Automática.

2 Herramientas utilizadas para la programación y simulación de la instalación.

En este capítulo se estudiarán los paquetes de software y la metodología utilizada para el diseño e implantación del sistema de automatización de la maqueta del manipulador neumático. Previamente se diseña el sistema de automatización utilizando un paquete de programación y simulación de sistemas automatizados con PLCs, el ISaGRAF. Luego se realiza la implantación del accionamiento local automatizado mediante el autómata programable TSX17-20. Por lo anterior se explican a continuación estas herramientas, concluyendo al final con la descripción de su uso para desarrollar el sistema automático de la maqueta del manipulador neumático.

2.1 El ISaGRAF como herramienta de simulación.

Con la aparición de los diversos lenguajes de programación de autómatas ha hecho falta una herramienta que se encargue de la simulación de los mismos. Esto permite saber antes de montar la aplicación, cual será el posible resultado que arrojará el programa real, lográndose esta forma una garantía en el diseño realizado.

Es por ello que se escoge el ISaGRAF como simulador de programas de PLCs en varios lenguajes según la norma IEC 1131 como el IL, ST, LD, FBD y SFC. El LD se ha popularizado por su sencillez a tal punto que en una encuesta realizada en el año 2000 arrojó que el 93% de los encuestados usaba este tipo de programación.

Un autómata que utiliza el lenguaje de programación LD es el TSX17-20 de TELEMECANIQUE. Como el presente trabajo es precisamente el control de un manipulador utilizando este autómata, hemos recurrido a esta herramienta de simulación para saber los resultados de antemano, detectar cualquier posible error y así como lograr su solución.

A continuación se muestran las características de la programación en LD, el programa de control del autómata y su simulación gráfica lograda gracias al SpotLight del ISaGRAF. Se explica además como fue posible su realización.

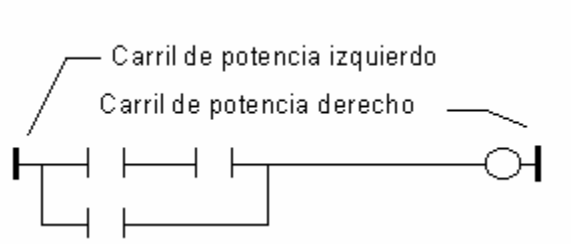
2.1.1 El lenguaje LD.

El Diagrama de Escalera (Contactos) (LD) es una representación gráfica de ecuaciones booleanas que combina contactos (argumentos de entrada) con bobinas (resultados de salida). El lenguaje LD permite la descripción de pruebas y modificaciones de datos booleanos mediante la

colocación de símbolos gráficos en el diagrama del programa. Los símbolos gráficos LD se organizan dentro del diagrama del mismo modo que en un diagrama de contactos eléctricos. Un diagrama LD está conectado en sus laterales izquierdo y derecho con carriles de potencia verticales.

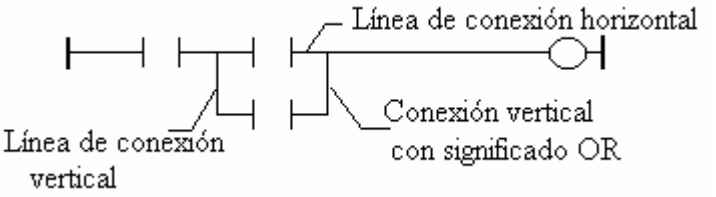
Carriles de potencia y líneas de conexión:

Un diagrama LD está delimitado en sus laterales izquierdo y derecho por unas líneas verticales, denominadas carril de potencia izquierdo y carril de potencia derecho, respectivamente.



Los símbolos gráficos de los diagramas LD están conectados a carriles de potencia o a otros símbolos por medio de líneas de conexión. Las líneas de conexión pueden ser horizontales

o verticales.



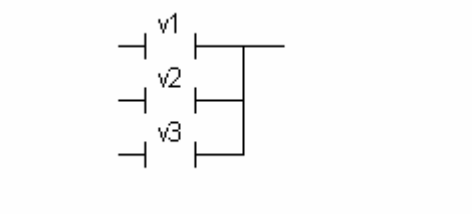
Cada segmento de línea tiene un estado booleano FALSO o VERDADERO. El estado booleano es el mismo para todos los segmentos que estén conectados directamente entre sí. Cualquier línea horizontal que esté conectada al carril de potencia izquierdo tiene el estado VERDADERO.

Conexiones múltiples

El estado booleano que se atribuye a una única línea de conexión horizontal es el mismo que en las extremidades izquierda y derecha de la línea. La combinación de líneas de conexión horizontales y verticales permite la construcción de conexiones múltiples. El estado booleano de las extremidades de una conexión múltiple obedece unas reglas lógicas.

Una conexión múltiple a la izquierda combina más de una línea horizontal conectada en el lateral izquierdo de una línea vertical, y una línea conectada en su lateral derecho. El estado booleano de la extremidad derecha es la 'O' (OR) LÓGICA entre todas las extremidades de la izquierda.

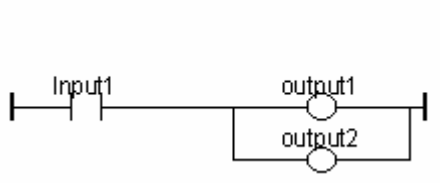
(* Ejemplo de conexión múltiple a la IZQUIERDA *)



(* estado de extremidad derecha es (v1 OR v2 OR v3) *)

Una conexión múltiple a la derecha combina una línea horizontal conectada al lateral izquierdo de una línea vertical, con más de una línea conectada a su lateral derecho. El estado booleano de la extremidad izquierda se propaga a cada una de las extremidades derechas.

(* Ejemplo de una conexión múltiple a la DERECHA *)



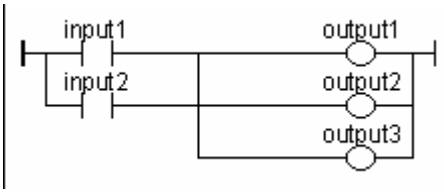
(* Equivalencia ST: *)

output1 := input1;

output2 := input1;

Una conexión múltiple a la derecha y a la izquierda combina más de una línea horizontal conectada al lateral izquierdo de una línea vertical, y más de una línea conectada a su lateral derecho. El estado booleano de cada una de sus extremidades de la derecha es el 'OR' LÓGICO del conjunto de extremidades de la izquierda.

(* Ejemplo de conexión múltiple IZQUIERDA y DERECHA *)



(* Equivalencia ST: *)

output1 := input1 OR input2;

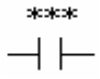
output2 := input1 OR input2;

output3 := input1 OR input2;

2.1.1.1 Contactos y bobinas básicos del lenguaje LD

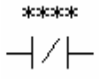
Se dispone de diversos símbolos para los contactos de entrada

- Contacto directo:



contacto normalmente abierto. La instrucción es verdadera cuando el bit correspondiente está activado(“1”).

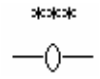
- Contacto invertido:



contacto normalmente cerrado. La instrucción es verdadera cuando el bit correspondiente está activado(“0”).

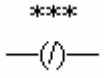
Se dispone de diversos símbolos para las bobinas de salida

- Bobina directa.



bobina de relé. Mientras la fila de instrucciones es cierta se activa el bit correspondiente; si la fila deja de ser cierta(deja de cumplirse al menos una condición) se desactiva.

- Bobina invertida.



bobina inversa. Mientras la fila de instrucciones es falsa se activa el bit correspondiente; si la fila pasa a ser cierta entonces se desactiva.

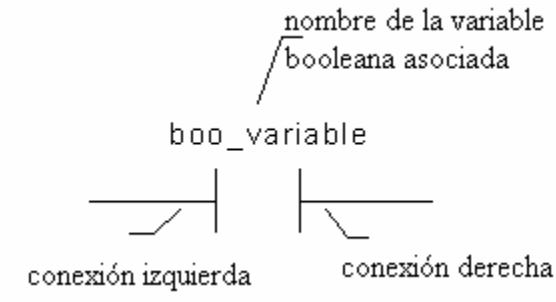
- Bobina SET.

***
—(S)— bobina tipo set. Si se cumplen las condiciones de la fila correspondiente está se activa y se mantiene así hasta que se desactiva por una bobina tipo reset.

- Bobina RESET.

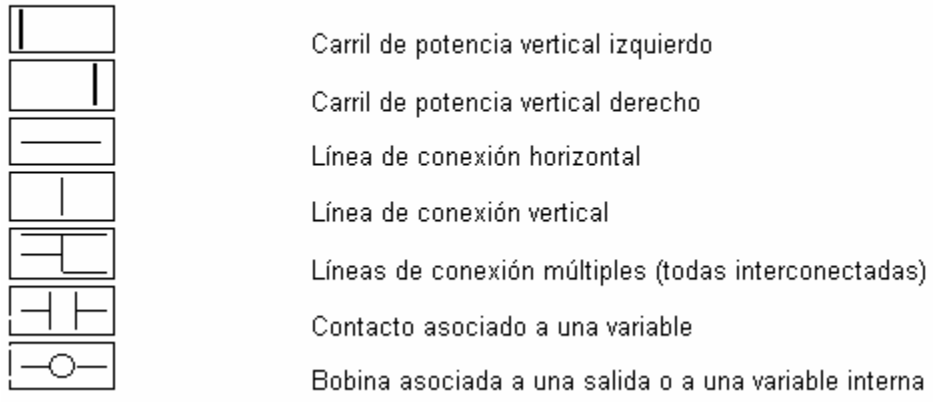
***
—(R)— bobina tipo reset. Si se cumplen las condiciones de la fila correspondiente está se desactiva y se mantiene así hasta que sea activada por una bobina tipo set.

El nombre de la variable se escribe por encima de cualquiera de estos símbolos gráficos



Estos son los componentes gráficos básicos de un diagrama LD:

Estos son los componentes gráficos básicos de un diagrama LD:

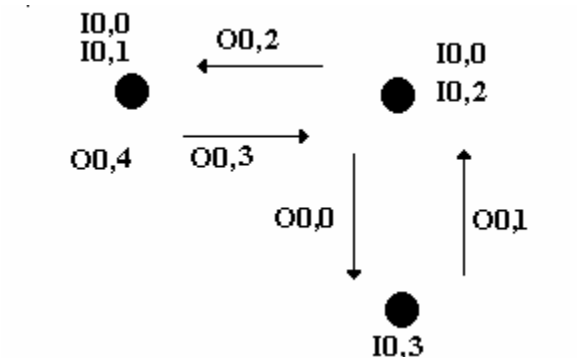


Luego de conocer cuestiones básicas de este lenguaje pasemos a analizar el programa del autómata realizado en el editor de programas del ISaGRAF.

El programa realizado en el ISaGRAF se encuentra en el anexo #1.

2.1.1.2 Explicación del programa de control del manipulador.

Para poder comprender mejor este tipo de programación de contactos debemos partir de cómo se comienza a trabajar con él. Lo primero que se realiza es asociar a los estados físicos del proceso estado de salidas y entradas del autómata para que de esta manera sea más sencilla la programación a realizar. Analicemos el siguiente esquema :



En nuestro programa específicamente existen tres estados físicos deseados ya que la secuencia de trabajo es la siguiente: el manipulador debe bajar a buscar la pelota, partiendo de la posición inicial de encontrarse arriba a la derecha y no tener la pelota, subir desplazarse a la izquierda y sólo entonces soltar la pelota, o sea, los tres estados son arriba y derecha, arriba e izquierda y abajo a la derecha, abajo a la izquierda no es una posición válida pues rompería la canal por lo que dicho estado debe ser validado.

Después de conocer los estados deseados debe partirse de una posición inicial como decía anteriormente. A cada estado asocio las salidas y/o entradas correspondientes y según la secuencia de operación propuesta por Ud. procede a realizar su programa. El programa que se realizó en el presente trabajo se desarrollo de la siguiente forma:

- Se partió de la posición inicial arriba (que se le asoció la entrada **I0,0)**, derecha(**I0,2**) sin pelota, o sea, cuando el manipulador se encuentre en está posición el debe bajar a buscar la misma y esto se logra energizando la electroválvula que permite pasar el aire proveniente del compresor al cilindro con la ventosa, acción esta que se ejecuta por medio de **O0,0** (salida cero del autómata).
- Cuando el baja entonces se encuentra una nueva posición ‘no arriba’, derecha(**I0,2**) y cuando coge la pelota (**I0,3**), entonces para subir necesita “apagar” a **O0,0** y encender a **O0,1** que provoca el movimiento deseado.

- Luego al encontrarse en la nueva posición arriba(**I0,0**), derecha(**I0,2**) pero esta vez con la pelota(**I0,3**) se quiere desplazar el sistema a la izquierda para posteriormente soltar la pelota. Esta acción se lleva a cabo "encendiendo" la electroválvula correspondiente y que responde al nombre de **O0,2**.
- Una vez que el manipulador ha llegado a la izquierda las entradas del autómata toman otra configuración, o sea, continua arriba(**I0,0**), ahora a la izquierda(**I0,1**) y con la pelota(**I0,3**), pero por la secuencia prefijada el sistema debe dejar caer a esta última proceso ejecutado al energizarse la salida **O0,4** esta encargada de la electroválvula que realiza dicha acción.

Sólo nos queda entonces retornar a nuestra posición inicial "apagando" las demás electroválvulas y energizando la que propicia el movimiento a la derecha, o sea, **O0,3** así al volver al estado de comienzo se repetirá la secuencia hasta que se desee cambiar. Se debe tener en cuenta los estados no deseados para su validación igual que en la programación de tipo texto.

Este es el primer paso a seguir para la programación en lenguaje de contacto, escribir literalmente las acciones que se quieren ejecutar y en la secuencia deseada para luego transformarla al lenguaje de escalera que es verdaderamente muy sencillo porque se trata de un programa con condiciones. Ahora se muestra el programa de nuestro manipulador traducido a diagrama de bloques. En los anexos se haya el programa íntegro en PL7-2.

Después de haber realizado la automatización de la instalación el sistema funciona de la siguiente manera:

Cuando el autómata es energizado, como posee una memoria RAM de 8K mantenida por una pila de 3V, el programa presente en el mismo fuerza al manipulador a desplazarse para buscar la pelota, posición abajo y derecha donde se mantiene hasta adherirse a la misma. Cuando esto sucede se activa la entrada I0,3 del PLC activándose así la salida O0,1 que es la encargada de energizar la electroválvula que lleva la alimentación neumática al cilindro de la ventosa produciendo de esta forma la subida del émbolo y con él de la ventosa además se resetea la salida O0,0 para evitar la ocurrencia de dos órdenes inversas, esto se realiza en todos los casos. Una vez que el manipulador llega arriba con la pelota se setea la salida O0,2 del PLC dándole paso a los 24V provenientes de esta salida hasta la electroválvula que suministra el flujo de aire al cilindro de movimiento horizontal trasladando al conjunto hacia la derecha. Cuando llega a su nuevo estado se acciona el microswitch correspondiente este envía la señal a la entrada I0,1 del

autómata con lo cual se resetea el movimiento hacia la izquierda y se activa la salida O0,4 esta última encargada de energizar la electroválvula que da el paso de aire al eyector cortándose el suministro del mismo a la ventosa y por ende la pelota es soltada, entonces es cuando el regulador de vacío vuelve a su estado de equilibrio cortando los 24V a la entrada I0,3 del PLC condición que produce entonces que se active la salida O0,3 de este y por lo tanto la electroválvula que produce el recogimiento del émbolo del cilindro de desplazamiento horizontal, también se energiza llevando al manipulador a la derecha. Después de esta situación se repetirá el proceso una y otra vez.

Los resultados obtenidos fueron los esperados por lo que se procedió inmediatamente a la programación en PL7-2.

Una vez analizado el programa pasemos a explicar cómo fue realizada la simulación usando el SpotLight del ISaGRAF.

El simulador del Kernel ISaGRAF se inicia junto con el depurador cuando se ejecuta el comando "Simular" del menú "Depurar" en la ventana del Gestor de Programas. El simulador del Kernel es un sistema objeto ISaGRAF completo que soporta las funciones estándares de ISaGRAF y todas las funciones y los bloques de función "C" de la librería estándar suministrada por CJ International. Las tarjetas de E/S se simulan gráficamente en una ventana. Se puede simular cualquier tipo de tarjeta de E/S. Las tarjetas definidas como "tarjetas virtuales" durante la conexión de E/S también aparecen en la ventana de simulación.

2.1.2 Simulación de E/S

Las tarjetas de E/S aparecen en la ventana de simulación, tituladas por su nombre y número de ranura. Se soporta cualquiera de los tipos estándares ISaGRAF de E/S (booleanas, analógicas o de mensaje). Se muestran los canales de las tarjetas de entrada con botones y campos especiales. Se muestran los canales de las tarjetas de salida con indicadores gráficos de estado y campos de datos.

 Entradas booleanas: Las entradas booleanas están representadas por botones cuadrados de color verde. Se muestra el número del canal con el botón de E/S. El valor de entrada es VERDADERO cuando se pulsa el botón. Al hacer click sobre el botón, se cambia el valor de E/S correspondiente. Únicamente cuando está pulsado el botón de entrada, se puede utilizar el botón derecho del ratón para configurar la entrada.

Salidas booleanas: Las salidas booleanas están representadas por círculos pequeños. Se muestra el número del canal con la E/S. El valor de salida es VERDADERO cuando está resaltado el símbolo gráfico.

La herramienta SpotLight de ISaGRAF permite definir al usuario listas de observación (watch) que pueden ser presentadas bien como dibujos gráficos o como listas durante la depuración. Los ítems gráficos deben estar enlazados a variables del proyecto ISaGRAF. El dibujo gráfico se define y anima "en línea".

2.1.3 Construyendo la composición gráfica.

Un diagrama esta constituido por dibujos de fondo (bitmaps o metafiles), y un conjunto de ítems gráficos que serán animados durante la depuración. Para crear el diagrama, se deben realizar las siguientes operaciones: Insertar dibujos de fondo, insertar ítems gráficos, enlazar objetos a las variables del proyecto.

2.1.3.1 Dibujos de fondo

Los dibujos de fondo son ficheros "bitmap" (.BMP) o "metafile" (.WMF). El número de dibujos incluidos en la composición gráfica no está limitado. Los dibujos se pueden mover o redimensionar en la composición gráfica. No aparecen en la composición de lista. Los dibujos se construyen con otras herramientas. SpotLight no incluye una herramienta de dibujo. El comando "Opciones / Color de fondo" se utiliza para seleccionar un color sólido para el espacio vacío en una composición gráfica.

Los bitmaps consumen una elevada cantidad de memoria. Por esto es altamente recomendable dimensionar correctamente el dibujo, y limitar el espacio no utilizado dentro del rectángulo del bitmap.

2.1.3.2 Composición gráfica animada.

Se pueden insertar los siguientes ítems gráficos en la composición gráfica:

- Presentación solo texto
- Gráficos de barras unipolares y bipolares
- Curvas
- Iconos booleanos

Un ítem "Icono booleano" se utiliza para presentar un estado binario. Se define un fichero icono (.ICO) para FALSO o valor 0. Se define otro icono para el resto de los valores distinto a cero. Como SpotLight no incluye un editor de iconos, los ficheros de iconos deben ser preparados con otra herramienta.

- Campos de bits.

Primeramente es necesario saber que en este simulador no pueden importarse iconos de un tamaño mayor a (40*40)cm lo que dificulta el trabajo para la visualización de la figura deseada. Estos iconos son realizados en el Paint e importados una vez que sean requeridos por el SpotLight.

Con la ejecución del simulador aparece un cuadrado en forma de autómata con las entradas y salidas definidas como se explica anteriormente. Cuando las entradas están siendo activadas según sea el programa se activan las salidas correspondientes encendiéndose las mismas. Cuando esto sucede recurrimos a la opción de insertar icono booleano armando de esta manera el dibujo que deberá aparecer cuando se active la salida correspondiente o la entrada ó ambas según se haya escogido. Así se logra realizar una simulación muy parecida a lo que realmente esta ocurriendo en la práctica ya que al desactivarse las E/S asociadas estos dejan de ser visibles y se le vuelve a asignar al nuevo estado otro dibujo. Como la desaparición y aparición ocurre de manera rápida nos da la impresión de movimiento. En el anexo #3 se muestra esta simulación.

2.2 Descripción del autómata TSX17-20 de la firma francesa TELEMECANIQUE.

El autómata programable TSX17-20 de TELEMECANIQUE es un PLC del tipo compacto, esto significa que la unidad central de procesamiento (CPU), la fuente de alimentación y las entradas/salidas se encuentran en el mismo módulo, puede tener 12, 22, o 24 entradas y 8, 12 o 16 salidas. Estos se pueden programar en lista de instrucciones (PL7-1), diagrama de contactos y GRAFCET. Para la programación en diagrama de contactos (PL7-2) o en GRAFCET el autómata debe portar un cartucho del lenguaje. A cada autómata se le pueden conectar tres módulos o bloques de extensión si se programa en PL7-2 o en GRAFCET y dos si se programa en PL7-1.

2.2.1 Mapa de memoria

Las entradas del autómata se nombran de la siguiente forma I0,,j donde I significa que es una entrada y j es el número de la entrada que puede ser (j=0..23) y las salidas O0,i donde de forma similar la O significa que es una salida y la i es el número de la salida y se mueve en el intervalo

(i=0..15). Las entradas y salidas de los bloques de extensión se nombran de la siguiente manera Ix,j y Ox,i con la x (x=1..3) que define el número del módulo referenciado en el orden que van conectados al autómata. Por ejemplo O0,5 es la salida 5 del módulo 0, o sea, del autómata y O3,1 es la salida 1 del módulo de extensión número 3.

Este autómata presenta además dos entradas adicionales (I0,24 y I0,25) que pueden ser configuradas como entradas normales o como entradas de interrupción ó respuesta rápida.

La entrada I0,0 puede configurarse como una entrada normal o como una señal de comienzo/parada. La salida O0,0 puede también ser una entrada normal o programarse para que sea una señal de funcionamiento correcto (activa la salida si el procesador está corriendo la aplicación y se encuentra desactivada si se ha producido algún error).

El módulo inteligente de entradas y salidas(e/s) analógicas puede tener e/s del tipo palabra (16bits) y se direccionan respectivamente como sigue: IWx,i y OWx,i donde x es el número del módulo correspondiente. Por ejemplo si el módulo 1 es de entradas analógicas, la palabra IW1,0 será la entrada analógica 0 de este módulo.

Los relés internos son los Bi(i=1..255) y sirven tanto como relés que no han de actuar, como salidas o para guardar estados binarios. Los primeros 127 bits internos guardan su valor después de una pérdida de tensión. Las palabras internas se llaman por Wi(i=1..127) y sirven para guardar valores provenientes de cálculos o entradas analógicas.

Para aquellos valores que no van a cambiar a lo largo del programa podemos emplear las constantes internas , nombradas como CWi,(i=1..127). tanto de constantes como de variables es posible llegar a 1024 efectuando un tratamiento tipo texto.

Los bits de etapa llamados como Xi(i=0..95), indican cuales etapas de GRAFCET (si se esta trabajando con este lenguaje) están activas. Si no se esta programando en GRAFCET todos estarán a cero. Para cada etapa hay una palabra, direccionada como Xi,v(i=0..95), que indica el tiempo que está activada cada una.

Los bits del sistema llamados como SYi, ellos tienen la información del estado de trabajo del autómata. Los bits de defecto Sx,i indican el estado en que está trabajando cada módulo. Las palabras del sistema llamadas como SWi(i=0..63) y contienen datos sobre el funcionamiento del sistema.

Las palabras pueden ser direccionadas bit a bit escribiendo Wi,j(i=0..127, j=0..15) para las palabras internas, CWi,j(i=0..127, j=0..15) para las constantes internas y SWi,j(i=0..63, j=0..15) para las palabras del sistema. Por ejemplo W12,3 es el bit 3 de la palabra interna 12.

2.2.2 Principales características del autómata TSX17-20.

El TSX17-20 es un autómata de tipo compacto que posee como principales características las que a continuación mencionamos:

- posee el cartucho de programación en PL7-2.
- la conexión de cartuchos EEPROM se hace directamente en el alojamiento reservado para ello en el propio autómata.
- todas las conexiones se realizan a través de una tira de bornes fácilmente recambiables en caso de deterioro mecánico o eléctrico.
- alimentación 24V cc.
- el mantenimiento de la RAM se realiza mediante una pila de 3V cc, su tamaño es de 8k.
- las salidas son a transistores que trabajan en corte o saturación y cuya fuente es de 24V cc que al activarse entregan este voltaje a su salida.
- puede expandirse a través de un conectador de 9 pines.

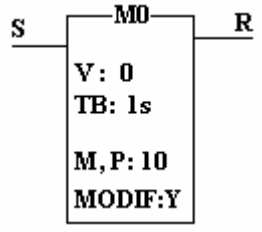
2.2.3 Programación en lenguaje de contactos: instrucciones tipo relé, bloques funcionales y saltos.

La programación en lenguaje de contacto es por páginas. Cada página tiene 4 líneas de 10 columnas. Las primeras 9 columnas están reservadas a contactos y funciones tipo texto como temporizadores, comparadores, etc. Mientras que la última es ocupada por bobinas y funciones tipo acción como saltos, operaciones, transferencias. Cada página debe tener una etiqueta cuyo nombre está entre 1..999 el valor no tiene importancia mientras todas tengan un valor diferente. Además cada página puede tener un comentario de 15 caracteres.

A continuación se explican como funcionan algunos bloques funcionales (los que se utilizan en el trabajo) e instrucciones tipo relé para que se comprenda mejor el funcionamiento del programa.

Mi (i=0..7)

- Monoestable**:** Palabras:



Mi,P: Preselección BCD.

Ci,V: Valor actual.

TB: base de tiempo (1mn, 1s, 100ms, 10ms).

Bit

S: Activación

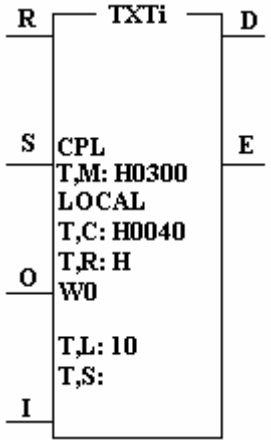
. R: salida

El monoestable activa la salida durante un cierto tiempo. Cada vez que la entrada S pasa de desactivar a activar cargara el valor de tiempo preseleccionado (Ci,V=Ci,P) y a partir de este momento va a decrementar el valor de Ci,V hasta llegar al valor 0, activando de esta forma la salida R.

Cuando Ci,V=0 la salida se desactiva. Si mientras esta temporizando, la entrada S pasa de desactiva a activa, cargará nuevamente el valor predeterminado (Ci,V=Ci,P) y seguirá temporizando.

- Bloque de texto.

El bloque de texto es el que permite la comunicación, este puede ser de tres tipos: TER, CPL y TXT. El tipo TER es para comunicarse a través de la conexión de terminal. Para conectar una red UNI-TELWAY adaptada a una red INTEL se debe usar el CPL.



TXT (i=0..7)

Bits de entrada	Bits de salida
R: reset.	D: fin de intercambio
S: inicio de intercambio.	E: error
O: emisión.	
I: recepción.	

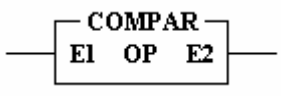
Palabras:

TXTi, M: módulo y dirección destino.

Si colocamos el cursor sobre la entrada R y le damos ZOOM podremos modificar el nombre del cuadro de texto (por ejemplo TXT1, TXT2). Si volvemos a dar ZOOM entraremos en una tabla en la cual podremos modificar el tipo de bloque de texto

(TER, CPL o TXT), la longitud de la tabla de emisión (TXTi, L), el módulo de acoplamiento y dirección destino (TXTi, M), entre otras palabras.

- Comparador.



E1: palabra

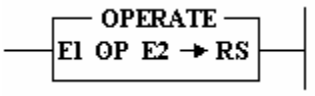
OP: operador

E2:palabra o valor

Efectúa una comparación entre dos palabras o entre una palabra y un valor. El resultado de la operación es el resultado de la instrucción.

Los operadores pueden ser: mayor(>), menor(<), igual(=), mayor igual(>=), menor igual(<=), diferente(<>).

- Operaciones.



E1: palabra.

OP: operador.

E2: palabra o valor.

RS: resultado.

Con este operador podemos realizar operaciones aritméticas (+, -, *, /), operaciones lógicas (AND, OR, XOR, CPL), operaciones de cambio de código, transferencias, operaciones circulares, entre otras. En este programa el **operate** se utilizó para realizar transferencias de las salidas y las entradas del autómata a palabras internas del mismo para su posterior lectura. Más adelante se explica con detalles estos bloques funcionales.

Las instrucciones tipo relé de este lenguaje son las mismas que utiliza el LD del PL7-2 del ISaGRAF y son explicadas en el punto **2.1.1.1.**

El programa realizado en este lenguaje se encuentra de forma completa en los anexos #1 y 2.

3 Herramientas para supervisión local y remota de la instalación.

Para realizar la supervisión local y remota del sistema automático del manipulador neumático de la maqueta del laboratorio desde una PC, se utilizó el Labview como herramienta de programación por las facilidades de comunicación de su biblioteca de VIs.

A continuación se explica dicha herramienta.

3.1 Introducción a Labview

Labview es un entorno de programación basado en el lenguaje de programación gráfico G para desarrollo de programas bajo plataforma Windows, Linus, SUN, HP-UX, Macintosh, Solaris, enfocado a la adquisición de datos, el proceso y el almacenamiento de mismos.

Este entorno de programación es una poderosa herramienta para generar aplicaciones en el control de procesos, permitiendo la amplia automatización de laboratorios de áreas experimentales por medio de equipo de cómputo utilizando una amplia biblioteca de funciones y subrutinas para mejorar las tareas de programación.

Labview a la vez es compatible con herramientas de desarrollo similares y puede trabajar con programas de otra área de aplicación, como por ejemplo MatLab. Tiene la ventaja que permite una amplia integración con hardware, específicamente con tarjetas de medición, adquisición y procesamiento de datos y de imágenes.

Labview permite desarrollar de manera más rápida cualquier aplicación, especialmente de instrumentación, en comparación con lenguajes de programación tradicionales basados en textos.

Una de las principales características de labview es su modularidad, es decir, la capacidad de utilizar bloques funcionales para la definición de la especificación. Labview permite conectarse a otras aplicaciones mediante un intercambio de datos como active x, librerías dinámicas, bases de datos, Excel y/o a protocolos de comunicación como DataSocket, TCP/IP, UDP, RS-232, entre otras.

Una característica se encuentra en el flujo de datos, que muestra la ejecución secuencial del programa, es decir, una tarea no se inicia hasta que las antecesoras terminen de ejecutarse. Debido al lenguaje gráfico el compilador con que cuenta labview es más versátil ya que sobre el mismo código de programación se puede ver fácilmente el flujo de datos, así como su contenido.

Labview también puede ser un programa en tiempo real donde la aplicación trabaja sin la necesidad de otro sistema operativo, este programa denominado labview RT viene con su propio Kernel que se encarga de la administración de las tareas.

Labview tiene su mayor aplicación en sistema de medición, como monitoreo de procesos y aplicaciones de control.

Cuando se diseñan programas con labview está trabajando siempre bajo el denominado VI (instrumento virtual). Este VI puede utilizarse en cualquier otra aplicación como una subfunción dentro de un programa general. Los VIs se caracterizan por: tener una interfaz de usuario, tener entradas con su color de identificación de dato, tener una o varias salidas y por supuesto ser reutilizables.

3.1.1 El labview como herramienta para la visualización.

Poder interactuar con un dispositivo de control programable que se encuentre conectado directamente a algún proceso resulta relativamente sencillo cuando se cuenta con una herramienta como el labview ya que este permite el intercambio de datos entre el periférico y la PC a través del puerto serie de esta última. Desde el labview se pueden procesar fácilmente los datos provenientes del proceso, así como enviarles configuraciones al PLC que respondan de manera eficiente a la respuesta esperada por el sistema ante los diferentes cambios.

Una de las principales aplicaciones del labview además de la antes mencionada es que este permite crear una interfaz de usuario interactiva y en tiempo real, esto nos da la posibilidad de realizar la supervisión de una instalación que no se encuentre ubicada en nuestra área de trabajo para la detección de posibles fallas en el funcionamiento de la misma así como su solución desde dicha área si necesidad de ir directamente al proceso.

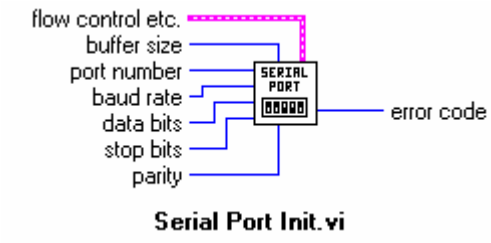
En el presente trabajo se realizará la supervisión desde una PC de una instalación que cuenta con un manipulador neumático con ventosa, controlado por el autómata TSX17_20 de la firma francesa TELEMECANIQUE, esta instalación se encuentra conectada directamente por el puerto serie al ordenador y al PLC mencionado. Además se realizará la visualización remota desde otras PC conectadas a dicho ordenador mediante una red de área local (LAN) utilizando para ello el protocolo de comunicación DataSocket.

A continuación de describe el procedimiento que se llevó a cabo para obtener los resultados satisfactorios en la labor realizada.

3.1.2 Comunicación por el puerto serie entre la PC y el PLC utilizando el labview.

El labview como se había mencionado anteriormente posibilita el intercambio de información entre el ordenador y el PLC a través del puerto serie. En el caso específico del TSX17_20 se realiza la conexión con este terminal de forma directa pues posee un adaptador DB_25 macho(conector del autómata) a DB_9 macho(terminal de la PC).

El labview tiene herramientas que realizan la lectura, escritura y limpieza de los buffers de entradas y salidas del puerto serie. A continuación se relacionan algunas de las mismas.



Antes de poder utilizar el puerto serie para la transmisión o recepción de datos es imprescindible configurarlo. De esta forma se le indica a la PC como va a manejar la comunicación. Al ejecutar el icono **Serial Port Init.vi** se eligen las características de comunicación deseadas para el puerto serie a continuación se detallan cada una de las entradas así como su única salida.

Flow control etc. Los parámetros que se introducen por esta conexión de entrada son los relativos a los protocolos de comunicación. Si se prescinde de ellos esta conexión se deja al aire.

Buffer size: es una constante que indicará al puerto el tamaño que se desea para sus buffers de entrada y salida en la recepción y transmisión de datos. El tamaño mínimo es de 1KB y de no indicársele ninguno toma dicho valor

Port number: es un número entero entre 0 y 13 con el cual se indica al PC el puerto serie a configurar. Por ejemplo 0:COM1, 1:COM2,...,8:COM9, 10:PRN1, ..,13:PRN4

Baud rate: valor de la velocidad de transferencia de datos en baudios. Sus valores típicos son 300, 600, 1200, 2400, 4800y 9600 baudios.

Data bits: indica el número de bits que deben tomarse como bits de datos en cada byte recibido. Los valores van de 5 a 8 bits.

Stop bits: indica el número de bits de parada que se desea para las transferencias. Los valores que toma son: 0 para 1 bit de stop ó 1 para 2 bits de parada.

Parity: tipo de paridad que se quiere en la comunicación. Se introducirá 0 para ninguna paridad, 1 para paridad impar ó 2 para paridad par

Error code: es la única conexión de salida del icono y entregará por ella un –1 en caso de cometerse un error en algunos de los parámetros asignados.

Otro icono importante es el **Bytes at Serial Port.vi**

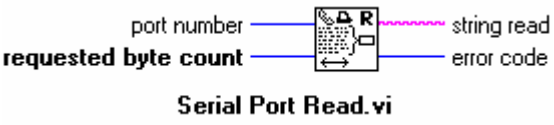
Este entrega la cantidad de bytes que tiene almacenado en su buffer de entrada en espera de ser leídos, dado un número de puerto determinado sus especificidades se relacionan a continuación.

Port number: puerto serie que se quiere consultar.

Byte count: entrega el número de bytes almacenados en el buffer de entrada del puerto indicado.

Error code: cuando por esta salida se recibe un valor distinto de 0 indica que se ha producido un error al tratar de acceder al puerto serie especificado.

Después de configurar el puerto serie y establecer la conexión es posible que el periférico haya transferido los datos al PC. Para el programador este proceso de recepción es transparente. Cuando se quiera acceder a dicha información debe de programarse el **Serial port read.vi** sus entradas y salidas se detallan a continuación:



debe tenerse en cuenta que cuando se lee la información del buffer de recepción esta dejará de estar almacenada en él, dejándose espacio para la llegada de posibles datos.

Port number: puerto serie a consultar.

Requested byte count: número de bytes a leer, este no puede ser superior al tamaño del buffer de recepción configurado con el **Serial Port Init.vi** ni superior a la cantidad de bytes almacenados en dicho buffer. Para evitar esto se recomienda cablear la salida **byte count** del

Bytes at Serial Port.vi con lo que siempre serán leídos todos los todos los bytes almacenados en el buffer.

String Read: por esta salida se entregan en forma de cadena de caracteres, los datos leídos en el buffer de recepción. Se debe conocer que si la información recibida es de tipo texto, o sea, son cadena de caracteres, esta se puede visualizar directamente, si por otro lado son bytes con un significado específico se debe realizar un cambio de formato adecuado para poder interpretarla de manera correcta.

Error code: se entrega un valor diferente de 0 si se ha producido un error en la lectura de los datos en el puerto.

Además de estos iconos existen otros como el Serial Port Write.vi, el Serial Port Beak y el Closed Serial Driver que no se detallan por no ser utilizados en esta aplicación.

Una vez conocidos estos elementos pasemos a la descripción del programa realizado para la visualización usando el labview.

Lo primero que se hizó fue inicializar el puerto teniendo en cuenta que los valores asignados estuvieran en total acuerdo con los valores de las trasferencias que realizaría el PLC, en el programa hecho en PL7-2 se detallan estas especificidades. Para ello los valores asignados fueron los siguientes:

- Puerto serie COM1.
- El tamaño del buffer de 1K.
- Ningún protocolo de comunicación.
- La velocidad de transferencia de 9600 baudios.
- Se tomaron 8 bits de datos.
- Ningún tipo de paridad.
- Dos bits de stop.

Después de esto se abrió una secuencia con la cual se garantiza en la primera escena 0(0..1) la limpieza del buffer de entrada al decirle al **Bytes at Serial Port** que cada vez que el número de bytes presentes en el COM1 fuera distinto de 0 lo leyera. Recordemos que anteriormente se explicó que al realizar la lectura del puerto serie este se vaciaba. Aquí se hace además una espera de 200 milisegundos para volver a leer el mismo.

En la segunda escena se indica que si la cantidad de bytes es mayor o igual a 2 los leyera. Como esta lectura es una string y lo que se necesita es conocer el estado de las entradas y las salidas del autómata, los bits, para procesar la información debe realizarse un cambio de formato adecuado para ello. Lo primero fue transformarla en un arreglo de bytes sin signo utilizando para ello el **String to Bytes Array** con lo cual en el byte 0 de este arreglo se entrega el valor ASCII del primer carácter en la cadena. Luego este arreglo se llevó a un cluster usando para eso el **Array to Cluster** el cual convierte un arreglo unidimensonal a un cluster en el cual cada elemento es del mismo tipo de los elementos del arreglo. Después de esto y utilizando el **Unbundle** logramos separar el cluster en cada uno de los componentes individuales, en el mismo orden en el que entraron y organizados de arriba hacia abajo. Una vez llevado a cabo este proceso se multiplica el componente 0 *255 y se le suma el segundo para transformarlo a número. Una vez realizado esta acción se lleva finalmente a arreglo booleano para trabajar directamente con los bits, usando el **Number to Boolean Array** el cual convierte la entrada a un arreglo booleano de 8, 16 ó 32 elementos. En este el elemento 0 corresponde al bit menos significativo de la representación binaria del número entero. Luego de lograr esto y trabajando con el **Index Array** se obtienen indicadores de encendido de los bits igual que los presentes en el autómata, esto se logró gracias a que en el TSX17-20 con la ayuda de los bloques de comunicación(**TXT**) se procedió a enviar al PC el estado de sus entradas y salidas mostradas en forma íntegra con el **Index Array**. Luego se hace una lógica cableada para lograr la visualización de las figuras realizadas en el Paint. Según sea el estado de las entradas y por ende de las salidas así será la foto que aparecerá en la pantalla. Este proceso se muestra a continuación.

Después de realizar la lógica se pone un indicador que nos enseñará el estado de los bits correspondientes. Para lograr hacer un control que cuando estos bits se activen aparezca lo que deseamos vamos al panel frontal de labview, hacemos click **derecho/controls/boolean** y sacamos un led entonces haciendo click derecho encima de este, seleccionamos la opción **advanced/Customize,** en el nuevo panel nos apoyamos en la opción **operate/Change to Customize Mode**, con la misma se logrará separar el led indicador del fondo que expandimos hasta lograr el tamaño deseado para la foto hecha en Paint que será importada por la opción **Edit/Import Figure from File**, luego poniendo el mouse encima del led apretamos el click derecho y con la opción **Import Figure** gracias a esto aparece la foto en la pantalla, después damos nuevamente click derecho pero ahora la opción será **Copy to Clipboard** con ella se copia la figura, una vez realizada la operación se vuelve a dar click derecho **picture ítem** aquí salen

ahora 4 estados 2 encendidos y 2 apagados entonces debemos asignar la figura al mismo estado pero en la posición nueva. Cuando terminamos este proceso ya el indicador se encuentra totalmente listo para mostrar la figura cuando se active el estado correspondiente a la misma, o sea, hemos hecho nuestro propio control. Para evitar que una figura que no es la correcta aparezca en el display debemos hacerla invisible en su estado de apagado, es por eso que vamos al diagrama apoyamos el ratón encima del indicador en cuestión y haciendo click derecho escogemos la opción **Create/Property Node**, nos entonces un pequeño cuadro, posamos al ratón encima del nuevamente damos click **derecho/Change to Write** con lo que se garantiza el apagado de la figura cuando sea falsa, o sea, cuando el bit correspondiente es 0.

El programa realizado y la visualización obtenida se encuentran en el anexo #4.

3.2 El DataSocket como vehículo para la visualización remota.

3.2.1 Introducción

El Internet continúa volviéndose más integrado en nuestras vidas diarias. Esto es particularmente cierto para científicos e ingenieros, porque diseñadores de sistemas de desarrollo ven el costo-precio neto como una norma mundial rentable para distribuir los datos. Hoy, los clientes de Nacional Instrument, pueden publicar los datos fácilmente de sus programas en la Web usando el Labview y LabWindows/CVI las herramientas de Internet. Con éstas herramientas de Internet, los programadores, crean aplicaciones que sirven imágenes de los paneles frontales de sus aplicaciones como páginas web con muy poca o ninguna programación. El paso de imágenes sobre Internet es fácil; sin embargo, muchos usuarios están buscando soluciones más interactivas en que ellos realmente puedan controlar experimentos remotos usando los navegadores de web. Los usuarios también quieren aumentar al máximo la actuación, lo cual frecuentemente significa el paso de los valores de datos en lugar de las imágenes grandes. Mejorando el desarrollo de aplicaciones Web es posible, pero sólo con una red de computadoras especializada o con la experiencia de la programación en Internet.

La Nacional Instrument ofrece ahora el DataSocket, una nueva tecnología de programación en Internet que simplifica el intercambio de datos entre las computadoras y aplicaciones. Con DataSocket, los programadores pueden pasar los datos crudos(sin necesidad de llevar a algún formato específico) eficazmente por Internet y responder a múltiples usuarios sin la complejidad de programación de TCP de bajo nivel.

Esta es la razón por la cual se utilizó dicha herramienta con el fin de realizar la visualización remota de la instalación del manipulador neumático con ventosa.

A continuación se detallan algunas características de la comunicación a través de DataSocket para luego explicar su implementación en este trabajo específico.

3.2.2 ¿Por qué DataSocket?

Cogiendo todo su hardware y los componentes del software linkiandolos juntos se ha tenido pocos cambios.

Por ejemplo, enganchar al hardware requiere consideración de niveles de señal, la impedancia, etc., El software tiene su propio juego de cambios. Un buen ejemplo es portando datos dentro y fuera de las aplicaciones. Primero, usted debe medir los datos crudos que se hace usando herramientas tales como bibliotecas para el control del instrumento y adquisición de los datos. Segundo, usted, debe comunicarse entre programas usando otro juego de tecnologías. Algunas aplicaciones simplemente salvan los resultados en un archivo, otros pueden usar las soluciones del cliente TCP/IP de una red de computadoras, DDE, o Active X. Cada mecanismo de I/O tiene sus propios problemas y requiere cierta especialización para la implementación.

DataSocket, una tecnología que es parte de la colección de la Nacional Instrument, es una interfaz fácil de usar que proporciona el fácil acceso a varios mecanismos de I/O sin enredar al usuario en los detalles de bajo nivel. Él une la tecnología de comunicación establecida para la medida y automatización de la misma manera que un navegador de Web une diferentes tecnologías de Internet en una herramienta de fácil uso.

3.2.3 Ejemplo de transmisión de datos.

Las herramientas de la Nacional Instrument lo hacen sencillo para configurar un sistema de la medida que lo hace todo sin ayuda.. Ahora tomemos el próximo paso. Suponga que usted quiere compartir las mediciones con varias máquinas. Un guión típico es un laboratorio de la universidad dónde una máquina controla el experimento mientras varios estudiantes hacen su propio análisis en tiempo real desde los puestos de trabajo individuales. Históricamente, para hacer esto usted se habría vuelto a la biblioteca de TCP para distribuir los datos. Mientras estas bibliotecas pueden resolver el problema, hay varios pasos para completar:

- Escoger un TCP/IP número de puerto (y espera que ese no este en uso por cualquier otra aplicación del sistema).

- Define el protocolo (por ejemplo cuando coge, cuando envía)
- Configura el Servidor para escuchar en el puerto seleccionado y crea la conexión cuando el cliente iniciándola demanda.
- Configura el Servidor para allanar los datos y escribe a todas las conexiones.
- Determina cualquier error.
- Configura las aplicaciones del Cliente para conectar al puerto seleccionado, evita la colisión de los datos, y los muestra en la pantalla.

Claro, cuando usted hace los cambios al servidor, por ejemplo sumar un nuevo dato, usted tiene que arreglar a todos los clientes también. Con bastante trabajo, usted puede

realizar una aplicación muy robusta, pero tiene que agregar gran cantidad de código a su simple programa.

Realizar la misma tarea usando la tecnología de DataSocket requiere realizar dos pasos básicos:

- Abrir una conexión del DataSocket usando un nombre para identificar los datos.
- Escribir los datos a esa conexión como usted computa los nuevos resultados

En este caso, la programación TCP/IP de bajo nivel no se ha hecho para usted.

3.2.4 La medición de los datos.

DataSocket fue diseñada para satisfacer las necesidades de medida y automatización. Por ejemplo, con TCP/IP usted tiene que escribir el código para convertir sus datos medidos a una cadena no estructurada de bytes en la aplicación de la transmisión, así como el código para analizar la cadena de bytes atrás en su forma original subscribiendo las aplicaciones.

DataSocket, sin embargo, transfiere los datos en el mismo formato descrito lo que puede representar los datos en un número ilimitado de formatos, incluyendo cadenas, escalares, booleanas, y forma de onda.

Las operaciones de lectura y escritura en DataSocket transparentemente convierten sus mediciones a y desde la cadena de bytes fundamental para usted, eliminando la necesidad de escribir el código analizado de forma complicada. Además, usando el formato de datos de DataSocket, usted puede asociar los atributos definidos por el usuario con los datos. Por ejemplo, usted podría asociar una estampa de tiempo con una temperatura medida, o una velocidad de

muestreo con un arreglo. DataSocket simplifica grandemente el trabajo con la medición de los datos.

3.2.5 Un URL a Cualquier Fuente de los Datos.

Antes de que usted pueda ir muy adelante, es importante entender brevemente cómo DataSocket conecta a las I/O de las diferentes tecnologías. Todo comienza con cómo usted nombra el dispositivo o el recurso al cual usted está transfiriendo o recibiendo los datos. Típicamente una biblioteca de I/O tendrá una función abierta a la que usted pasa un nombre o numera para identificar la fuente que usted quiere leer de o escribir a. Para el archivo de I/O, el nombre del recurso es un camino del archivo, para TCP/IP hay dos partes para el nombre–un nombre de la máquina y número del puerto. Con DataSocket el nombre del recurso es en la forma de un URL (el localizador uniforme del recurso) muchos como la dirección Web usada por un navegador Web. Considere cómo un navegador Web interpretaría el URL.

Él le dice al navegador usar el protocolo TCP/IP básico llamado HTTP (protocolo de transferencia de hipertexto) para conectar a la máquina nombrada y para sacar la página Web nombrada DataSocket. El URL es diferente del nombre usado por la mayoría de las tecnologías de I/O en eso no sólo define en lo que usted está interesado, también indica cómo conseguirlo. El "cómo" codifica en la primera parte del URL se llama el método de acceso o protocolo. Los navegadores Web usan típicamente algunos métodos de acceso, como HTTP, HTTPS (el codificador de HTTP), FTP (el protocolo de transferencia de archivo), y ARCHIVO (por leer los archivos en su máquina local). Claro, usted normalmente no cuida cómo la página está cargada, usted simplemente ¡Quiere verla! DataSocket toma el mismo acercamiento para la medida de los datos. Por ejemplo, en los datos que comparten el ejemplo descrito sobre, DataSocket podría usar el URL siguiente para conectar al elemento de datos: el **dstp://mytestmachine/wave1**

el "el dstp" en el frente DataSocket dice abrir una conexión protocolar de trasferencia de enchufe de datos a **mytestmachine** y saca una señal llamada **wave1**. Si el URL hubiera empezado con "el archivo," se habrían sacado los datos de un archivo en lugar del servidor de DataSocket.

3.2.6 ¿ Qué es DataSocket?

DataSocket, una nueva tecnología de programación basada en la standard industrial de TCP/IP, simplifica el intercambio de los datos vivos entre las diferentes aplicaciones en una computadora o entre las computadoras conectadas mediante una red. Aunque una variedad de tecnologías diferentes existe hoy para compartir los datos entre las aplicaciones, como TCP/IP y DDE, la

mayoría de estas herramientas no están dotadas para la transferencia de los datos vivos. DataSocket lleva a cabo un uso fácil, una interface de programación diseñada para compartir y publicar los datos vivos en la medida y las aplicaciones de automatización.

DataSocket consiste en dos pedazos–el API de DataSocket y el Servidor de DataSocket.

El API de DataSocket presenta una sola interface para comunicar con diferentes tipos de datos, de múltiples lenguajes. El Servidor de DataSocket simplifica la comunicación de Internet por el manejo TCP/IP que programa para usted.

Es un API con protocolo-independiente, idioma-independiente, y OS-independiente diseñado para simplificar la publicación binaria de los datos. El API de DataSocket es implementado como un ActiveX, una biblioteca de LabWindows/CVI , y un juego de Labview VIs, para que usted pueda usarlo en cualquier ambiente de programación.

El API de DataSocket automáticamente convierte los datos de la medida del usuario en una cadena de bytes que se envían por la red. La aplicación de DataSocket subscribiendo, automáticamente convierte la cadena de bytes en su forma original. Esta conversión automática elimina la complejidad de la red que considera una cantidad sustancial de código que usted debe escribir al usar las bibliotecas de TCP/IP.

Aprender el API de DataSocket es simple. Consiste en cuatro acciones básicas (abrir, leer, escribir, y cerrar) eso es similar a los archivos standard de I/O llamados. Usted puede usar el mismo API de DataSocket en sus programas para leer los datos de:

- Elementos de datos en los servidores de HTTP.
- Elementos de datos en los servidores de FTP.
- Los archivos Locales.
- Elementos de datos en OLE para los servidores de control de Procesos (OPC).
- Elementos de datos en los servidores de DSTP.

3.2.7 El Servidor de DataSocket.

El Servidor de DataSocket es un peso ligero, el componente autosuficiente con que los programas usando el API de DataSocket pueden transmitir los datos vivos de la medida a altas velocidades por el Internet a varios clientes remotos concurrentemente. El Servidor de DataSocket simplifica la programación TCP de la red por el manejo automático de conexiones a los clientes.

La transmisión de datos con el Servidor de DataSocket requiere tres “actores”–publicador, el Servidor de DataSocket, y un subscriptor. Una aplicación de la publicación usa el API de DataSocket para escribir los datos al servidor. Una aplicación subscribiendo usa el API de DataSocket para leer los datos del servidor. La publicación y " las aplicaciones subscribiendo son “los clientes” del Servidor de DataSocket. Los tres actores pueden residir en la misma máquina, pero muy a menudo ellos corren en máquinas diferentes. La habilidad de ejecutar el servidor de DataSocket en otras máquinas mejora la actuación y proporciona la seguridad aislando las conexiones de la red de su aplicación de medida.

El Servidor de DataSocket restringe el acceso a los datos administrando seguridad y permisos.

Con DataSocket, usted puede compartir los datos confidenciales de la medida sobre Internet mientras prevenga el acceso desautorizado por los espectadores.

En esencia, el Servidor de DataSocket es de fácil uso, la solución general a la programación de TCP/IP, que reemplaza a los usuarios comúnmente escritos, en la creciente casa de códigos de red.

3.2.8 Aplicaciones del DataSocket.

Porque DataSocket es una herramienta de la programación de uso general para reforzar la medida las aplicaciones, puede usarse en una variedad de aplicaciones diferentes. Algunos ejemplos del uso de DataSocket se muestran a continuación.

3.2.8.1 Construyendo un Laboratorio del Estudiante Interactivo.

Imagine una universidad con un laboratorio de procesamiento de señal. La próxima semana es el primer día de clase, y usted, el profesor, debe preparar un experimento del laboratorio que introduce a los estudiantes en el análisis de señal. El laboratorio tiene 30 puestos de trabajo del estudiante, que conforman una red de computadoras y todas están conectadas al servidor del laboratorio. Usted quiere demostrar los principios de análisis adquiriendo y analizando una señal en uno de los puestos de los estudiantes y transmitiéndolo entonces por la red al resto de las computadoras, para que los estudiantes puedan ver los efectos del análisis de señal sin la necesidad de adquirir los datos en cada máquina.

Usted ya ha escrito una aplicación que adquiere y publica datos usando DataSocket.

Usted se desafía ahora con diseminar los datos al resto de los estudiantes. Usted decide usar el navegador Web como un vehículo para visualizar los datos vivos. Para construir la página Web,

usted usa el Visual Basic para construir una interface de usuario y convertirlo a un documento de HTML.

Usted usa los componentes de DataSocket para leer los datos publicados al servidor.

Leer los datos de un servidor de DataSocket consiste en tres pasos fáciles:

- abrir una sesión al servidor de DataSocket usando el DataSocket ActiveX control,
- leer el elemento del servidor de DataSocket,
- cerrar su conexión de DataSocket cuando su aplicación termina.

Usted puede ver los datos en cualquier parte de una red o en el Internet. Usted puede escribir aplicaciones que leyeron la información de una red o Internet con poco o ningún código.

3.2.8.2 DataSocket usando las Variables del Proceso.

Imagine una fábrica de la galleta que hace varios tipos diferentes de galletas. Cada tipo de galleta tiene su propia línea de la producción, y cada línea tiene una computadora que supervisa las variables del proceso.

Usted es ingeniero de los sistemas escogido para escribir una aplicación de Labview que continuamente supervisa cada uno de las variables del proceso. Su aplicación escribe los datos vivos a la oficina central sobre la red local. Una computadora en la oficina central recoge los datos y muestra un resumen vivo de la línea de producción y del proceso inconstante para que se tenga un cuadro moderno de cómo la fábrica está trabajando.

Sin DataSocket, usted tendría que escribir un servidor de TCP/IP y aplicación del cliente para transferir los datos de la fábrica a la oficina central. La aplicación del servidor habría de adquirir los datos del proceso, ponerlos en una cadena de bits, y transferirlos al servidor.

El servidor lee los datos, los procesa, y los muestra. Además del código requerido para leer la información del servidor, la aplicación del cliente también debe contener el

el código requerido para manejar las múltiples conexiones, una conexión para cada línea del proceso. Escribiendo todos los códigos TCP/IP de bajo nivel para manipular cada transferencia de los datos agregaría una significante cantidad de sobrecarga al desarrollo del proceso.

Con DataSocket, usted puede manipular fácilmente la comunicación de la red requerida para mover los datos de la fábrica a la oficina central. Porque los datos están escribiéndose en el servidor de DataSocket, la aplicación de la oficina central no necesita la implementación de

códigos extras para manipular las conexiones extras de las múltiples líneas de la producción. Él simplemente lee el elemento de los datos para cada línea.

Este ejemplo ilustra cómo transmitir los datos a través de una red local. Usted podría extender en el guión de la galleta pasando la información de mando de proceso de la oficina central atrás a las líneas de la producción. Porque la comunicación de DataSocket puede medir por palmos una red o el Internet, la oficina central puede localizarse en el mismo edificio o a medio camino alrededor del mundo.

3.3 Descripción del trabajo con DataSocket.

Después de haber conocido el funcionamiento básico del DataSocket pasemos a la descripción del trabajo realizado con el mismo en la visualización remota de nuestra instalación desde la Intranet.

3.3.1 Visualización remota.

Una vez que ha sido realizada la aplicación en algunos de los lenguajes como el labview o Labwindows/CVI se procede a visualizar los resultados desde otras PC. Para ello se hace lo siguiente:

- Copiar el programa original.
- Realizar las conexiones con DataSocket.
- Asignar a cada estado las imágenes correspondientes.

Después de realizar la copia del programa conectamos los elementos que deseamos sean leídos por los clientes ¿cómo se hace esto? Pues su respuesta es muy sencilla. Primero colocamos el cursor del ratón sobre dicho elemento, luego haciendo **click derecho/Data Operations/DataSocket Connection** aquí sale una ventana donde nos piden a donde conectarnos entonces le indicamos la dirección de la máquina en la cual se localiza el servidor de DataSocket y la aplicación. Esta conexión se lleva a cabo de la siguiente manera: **dstp: //máquina con servidor/nombre del elemento**, aquí también se indica el tipo de conexión, o sea, si es para publicar o para subscribir. En el caso del cliente que va a leer debe inscribirse como subscriptor. Esto se repite con cada uno de los elementos.

Para el programa original el procedimiento es muy similar sólo cambia la forma hacer la inscripción de publicador, ya que ahora en lugar de escribir la máquina con servidor se pone lo

siguiente: **dstp://localhost/nombre del elemento** y marcamos el publicador. Que significa que es una máquina local (Intranet), esta acción se hace con todas aquellos elementos que queremos sean leídos por los clientes.

Después de esto a la aplicación del cliente debe asignarse las imágenes correspondientes a cada estado y esta se hace de la misma forma a la explicada anteriormente.

De esta manera culmina nuestro trabajo al lograr la visualización remota del proceso analizado. La información se encuentra de manera más en el anexo #4.

Valoración Económica

En la actualidad los avances de la ciencia y la tecnología han alcanzado un desarrollo notable en las diferentes ramas de la producción. La automática es parte fundamental de este progreso ya que la creciente robotización industrial se ha convertido casi en una necesidad para alcanzar mejores indicadores de eficiencia y calidad.

Nuestra universidad como centro forjador de los nuevos ingenieros no debe estar exenta a este desarrollo por lo que se ha dado a la tarea de preparar a los futuros graduados de la mejor manera posible para ello se han comenzado a realizar trabajos encaminados a la creación en nuestra facultad de un laboratorio de robótica. El presente es uno de ellos. Para montar dicho laboratorio se realizó el intercambio de un manipulador mecánico, con todos sus componentes provenientes de la firma japonesa SMC, por un autómata TSX17-20 de la firma francesa TELEMECANIQUE. el manipulador está compuesto por 2 cilindros de doble efecto uno de ellos modelo C82ADB20-125 este con un valor ascendiente a $139.47 en USD, el otro modelo C92SDB32-50R con un precio de $135.39, 2 electroválvulas de doble solenoide modelo VZ3223 cada una con un costo de $161.35, una de simple solenoide modelo VZ2130 con un precio de $82.19, un eyector este de valor $240.38, un detector de vacío modelo PS1100 cuyo valor es de $78.15, una ventosa de diámetro 10mm con un precio de $11.00, posee además un detector magnético este con un precio de $37.24 y un autómata TSX17-20 con un costo de $500.00 todo esto en dólares, pero es válido señalar que estos PLCs fueron resultados de una donación de la Universidad Politécnica de Cataluña como parte de un proyecto del Centro de Cooperación para el Desarrollo. El costo total de la instalación ascendió a 1546.52 dólares.

Para calcular el costo de la mano de obra se utilizaron los datos siguientes: la tarifa de un programador “A” es de $4.90; se trabajaron 8 horas / día durante 3 meses y medio de 24 días laborales. Y hallando el valor de los costos de la mano de obra obtenemos que equivale a $3298.8.

Para calcular los costos por consumo de energía se tuvo en cuenta que una computadora personal consume aproximadamente 200 W/hora; que la tarifa por consumo de energía es de $0.09 cuando el consumo es menor de 100 kW/hora, $0.20 cuando el consumo está entre 100 y 200 kW/hora y $0.30 para consumos mayores de 200 kW/hora; que se trabajaron 8 horas / día durante 3 meses y medio. A partir de todos estos valores se plantea:

$$CE = C_{POT} \cdot Ht \cdot Dt \cdot Tce$$

donde

C_{POT}*:* Consumo de potencia por hora de trabajo del equipo.

Ht: Horas trabajadas al día.

Dt: Días de trabajo en un mes

Tce: Tarifa de consumo eléctrico.

Obtenemos como resultado que los costos por consumo eléctrico ascienden a la cantidad de *CE*=$12.69

Finalmente el resultado del costo de este trabajo es de: $1546.52 USD y $3311.49 en moneda nacional.

Por otro lado también se aplicó el Modelo Constructivo de Costo ("*COCOMO*"), el cual se basa en estimaciones matemáticas y se usa para calcular los costos "a priori" de los sistemas de cómputo, así como su tiempo de desarrollo, atendiendo a las etapas definidas para los mismos.

Este método define una serie de parámetros y ecuaciones que posibilitan tener una idea de cuanto se invierte en el desarrollo de un trabajo programativo. Estos parámetros son:

Tdes: Tiempo de desarrollo total.

Ks: Kilo sentencias (1Ks se describe como promedio en 5 días).

CFT: Cálculo diario del costo de la fuerza de trabajo.

TFT: Tarifa de fuerza de trabajo por hora (Para un programador "A" el TFT=$4.90).

CTM: Cálculo diario del costo del tiempo de máquina.

TTM: Tarifa del tiempo de máquina. Para una microcomputadora es de $7.69 h. Esta tarifa de tiempo de máquina incluye la amortización de su costo de producción y su aprovechamiento.

Teniendo en cuenta la inicial falta de documentación existente y el volumen programativo el cálculo del valor económico del aseguramiento programativo se elaboró de la forma siguiente:

Tiempo de desarrollo total del software:

Tdes = *Ks* • 5días donde *Ks* = 9

Tdes = 9• 5días

Tdes = 45 días.

Cálculo diario del costo de la fuerza de trabajo:

CFT = 0

Cálculo diario del costo del tiempo de máquina:

CTM = *TTM* • 8 horas/días

CTM = $7.69 • 8 horas/días

CTM = $61.52

Cálculo del tiempo de desarrollo por etapas:

Factibilidad:

TF = (30 • *Tdes*)/100

TF = (30 • 45 días)/100

TF = 13.5 días

Diseño:

TD = (45 • *Tdes*)/100

TD = (45 • 45)/100

TD = 20.25 días

Implantación:

TI = (25 • *Tdes*)/días

TI = (25 • 45)/días

TI = 11.25 días

Costo por etapas:

Factibilidad:

CF = CFT • TF

CF = 0

Diseño:

CD = (CFT+CTM) • TD

CD = (0+$ 61.52) • 20 días

CD = $ 1230.4

Implantación:

CI = (CFT+CTM) • TI

CI = (0+$ 61.52) •11 días

CI = $ 676.72

Costo del soporte programativo:

CT = *CF*+*CD*+*CI*

CT = 0 + 1230.4+ 676.72

CT = $ 1907.12

El costo total del soporte programativo previsto para el desarrollo de la programación del PLC, el ISaGRAF y el Labview es de 1907.12.

Es necesario aclarar que este tiempo no es absoluto ni mucho menos ya que para este cálculo no se tiene en cuenta el tiempo de aprendizaje de los tres lenguajes de programación utilizados, debido a ello el tiempo estimado no fue cumplido como se planteaba.

La posibilidad de esta instalación automatizada mediante Autómatas Programables en el laboratorio del Departamento de Control Automático y su acceso remoto, permitirá una formación más integral de los egresados de esta Universidad. El beneficio estará dado por la capacidad técnica desarrollada, acorde con los niveles actuales de automatización adquirida por los estudiantes y que será reflejada en cada industria ó puesto donde se les sitúe donde podrán sus aportes cada proceso productivo al que se enfrenten.

La repercusión económica del presente trabajo es amplia tanto a escala docente como profesional.

Conclusiones

Después de haber terminado este trabajo se logró la creación de una instalación de laboratorio del sistema automatizado de un manipulador neumático y su supervisión remota. Se obtuvieron resultados satisfactorios tanto con la programación del autómata empleado, como con las herramientas de simulación y visualización utilizadas para obtener el correcto funcionamiento de la instalación y la supervisión de ésta.

Esta instalación tiene como aspecto novedoso la programación de la automatización y supervisión de un manipulador, trabajo este que hasta el momento no había sido desarrollado en nuestra facultad. Esto es de gran utilidad para nuestros estudiantes porque realiza un acercamiento entre ellos y los procesos industriales reales lo que conlleva a una mejor preparación profesional a la salida de nuestro centro de estudios. Además al introducir prácticas asociadas a robótica en nuestro laboratorio pueden sentarse las bases para la inclusión de esta temática en los programas de estudios de pre y postgrado con bases sólidas. También favorece que se realicen trabajos de grupos científicos referidos al tema en nuestro laboratorio.

Recomendaciones.

Se recomienda se continúe la programación de manipuladores mecánicos, incorporando el control remoto de la instalación desde la Intranet y el Internet aprovechando las facilidades que ofrece el labview y los trabajos referentes al tema.

Recomendamos el desarrollo de un laboratorio de Robótica en nuestra facultad con vista de facilitar el mejor aprendizaje y preparación de nuestros estudiantes.

Recomendamos además que se realice un estudio minucioso sobre Robótica para ver las posibilidades de incluirla como una asignatura más dentro del plan de estudio, sea opcional u obligatoria, ó al menos realizar grupos científicos relacionados con el tema.

Bibliografía

Amat, J., Ayza J., Basañez, L., Ferrate, G., Ferrer, F., Huber, R., Torres, C. Robótica industrial. Barcelona-México.

Angulo, J. Robótica práctica: tecnología y aplicaciones. Madrid. España.1985.

Anguita, E. Apuntes de PLC. http://www.edsonanguita.8m.com/.

Åström, K.J., Kheir, N.A., Auslander D., Cheok K.C., Franklin G.F., Masten M. And Rabins M. "Control Systems Engineering Education". Automática, Vol 32, No 2, pp. 147-166. Great Britain. 1996.

Bartos, F. "Artificial intelligence: smart thinking for complex control". Control Engineering, pp. 44-52. Jul. 1997.

Benitez. I. y col. Estación de control de procesos en automática para un laboratorio virtual. Taller laboratorios virtuales. Dpto Informática. Universidad de Oriente. Noviembre 2001.

Introducción a las comunicaciones serie. http://www.ctv.es/pckits/tutore.html

Johnson G. Labview Grafical Programing. 2da Edición.1997.

Langages PL7-2. Manuel de référence. Francia. Sep. 1990.

Langages PL7-2. Modes opératoires V3. Francia. Feb. 1991.

Lázaro, M. LabVIEW Programación gráfica para el control. Paraninfo. España. 1997.

Manipuladores neumáticos de SMC www.smc.com

National Instrument. LabView documentation in electronic format. 2000.

Programa de la Asignatura Automática V. Departamento de Control Automático. Universidad de Oriente. 2002.

Programación serie 7 de TELEMECANIQUE TSX17 www.modicon.com\techpubs, www.scheneider-electric.com.mx\sch531800.

Rosa J.; García A. Sistema docente para la programación sobre la IEC-1131. Trabajo de diploma. Dpto CA. FIE. UO. Santiago de Cuba. Junio 2001.

Sánchez, I. Trabajo de fin de carrera. Oviedo. España. 2000

SMC_ANELI11_1.pdf, SMC_ANELI11_2.pdf .

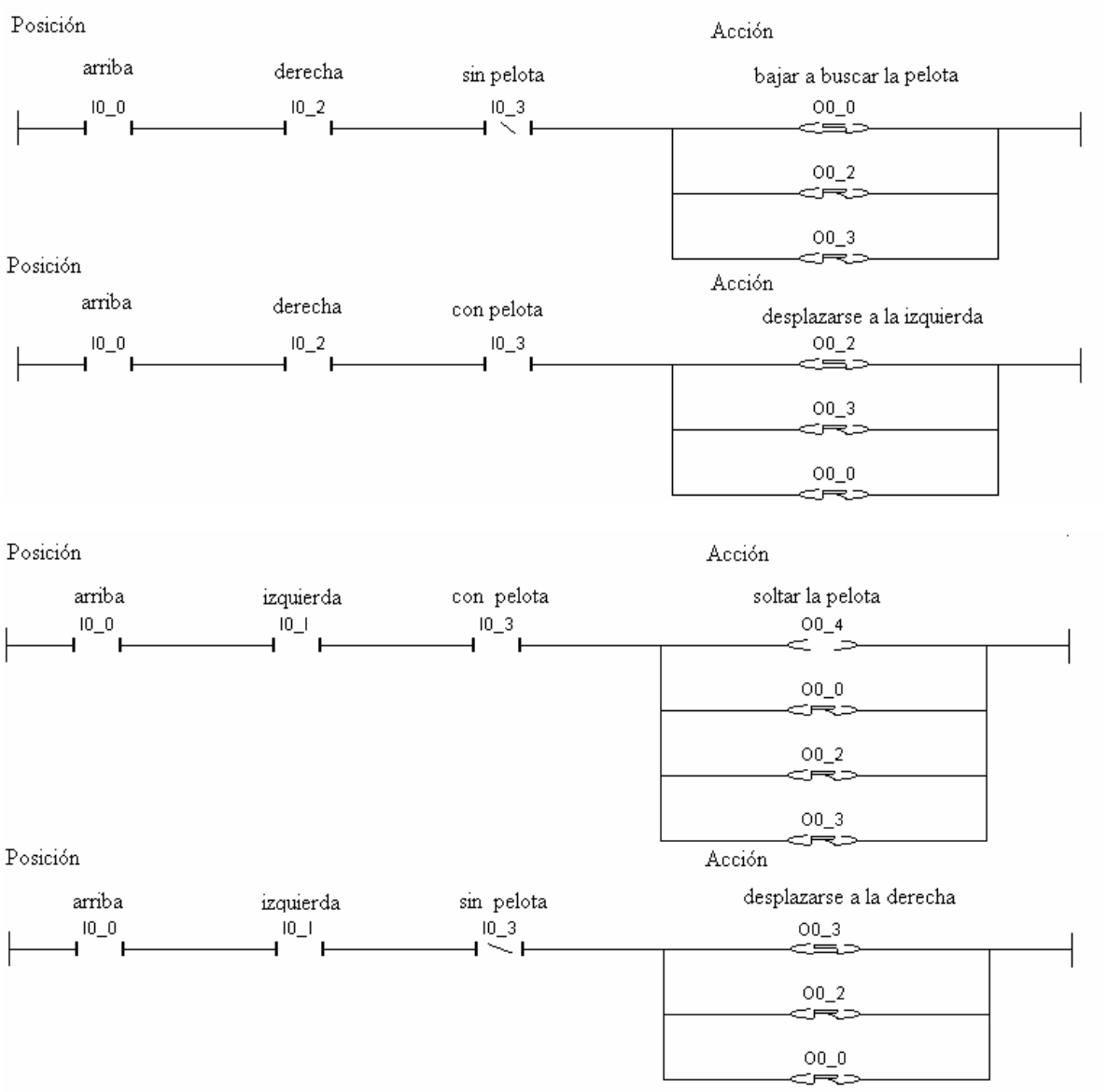
Staff G. Smaller PLCs retain popularity. Control Engeneering. March 2000. www.controleng.com

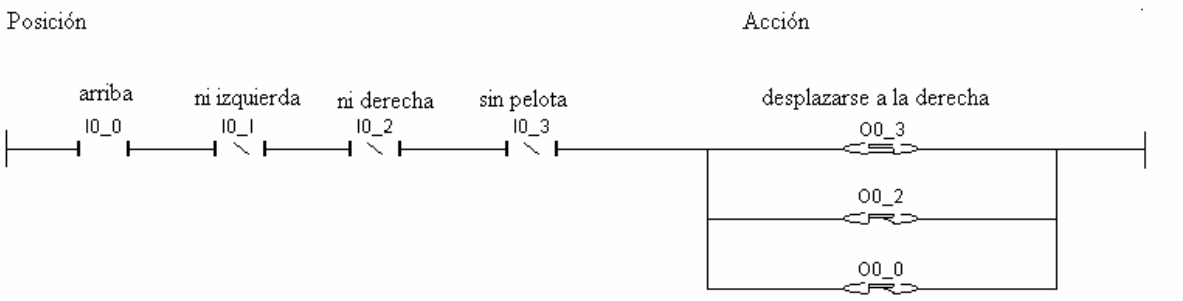
Sudriá A. y otros. Programació d´autòmats Telemecanique. Edicions UPC. España. 1994.

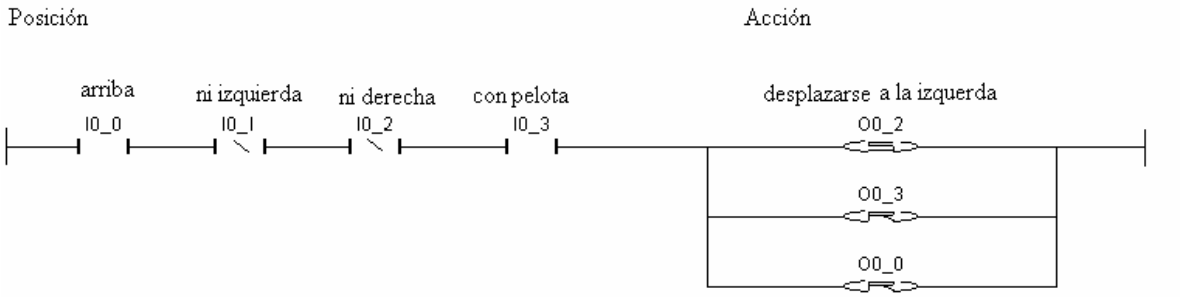
Trabajo sobre lenguajes de programación de Autómatas programables.2000. http://www.lafacu.com.

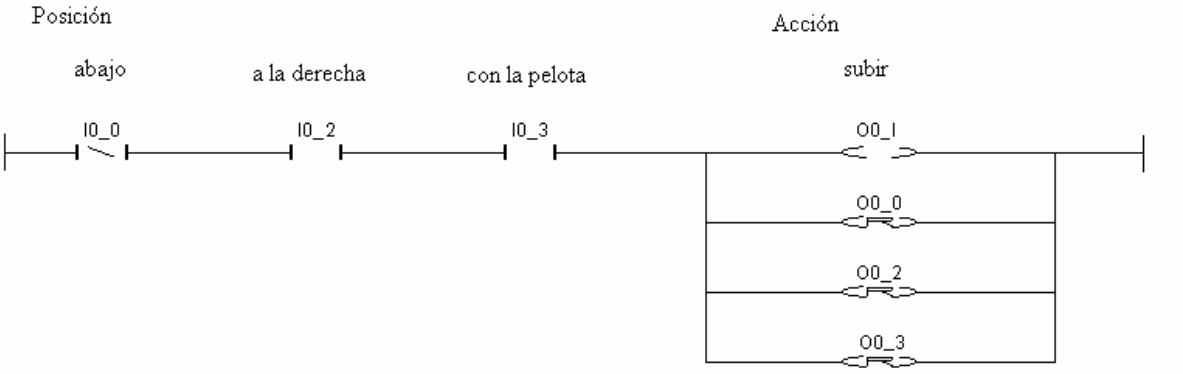
ANEXO # 1

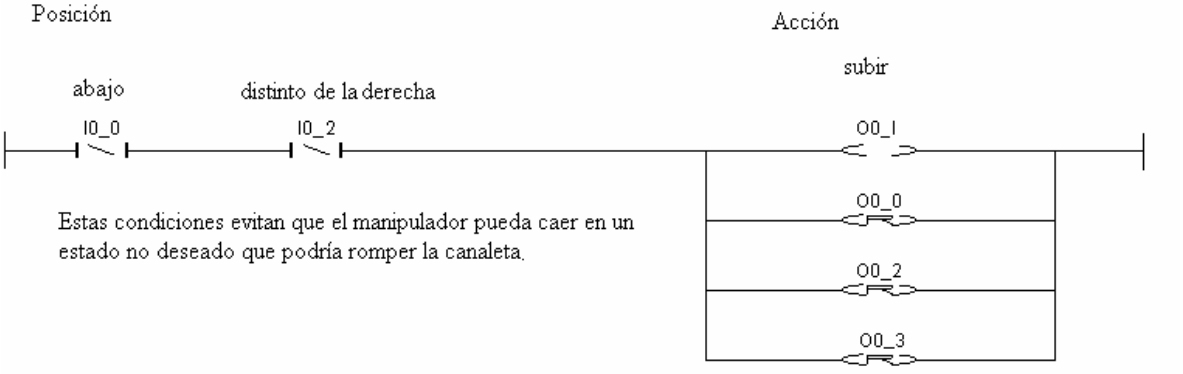
Programa realizado en el ISaGRAF para la simulación.









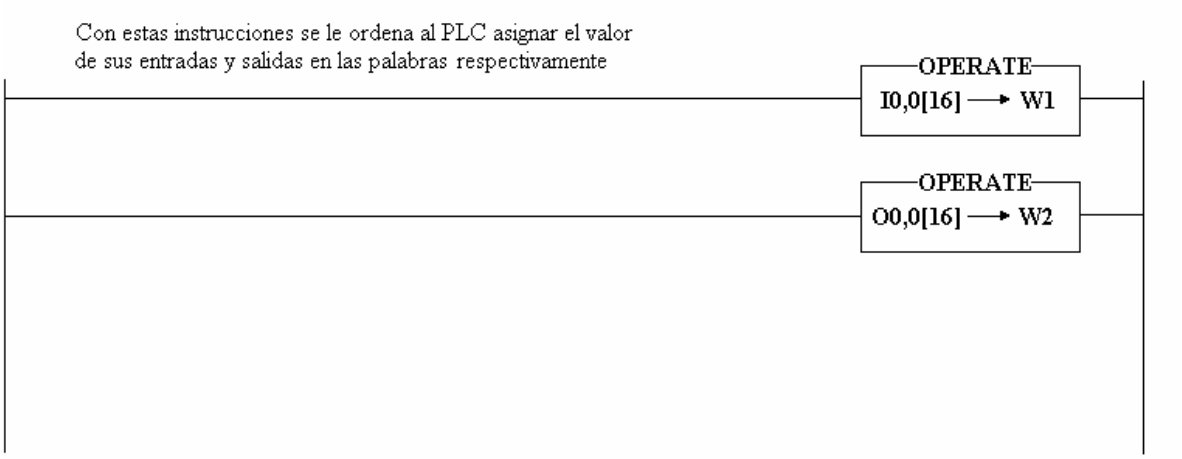


Estas condiciones evitan que el manipulador pueda caer en un estado no deseado que podría romper la canaleta.

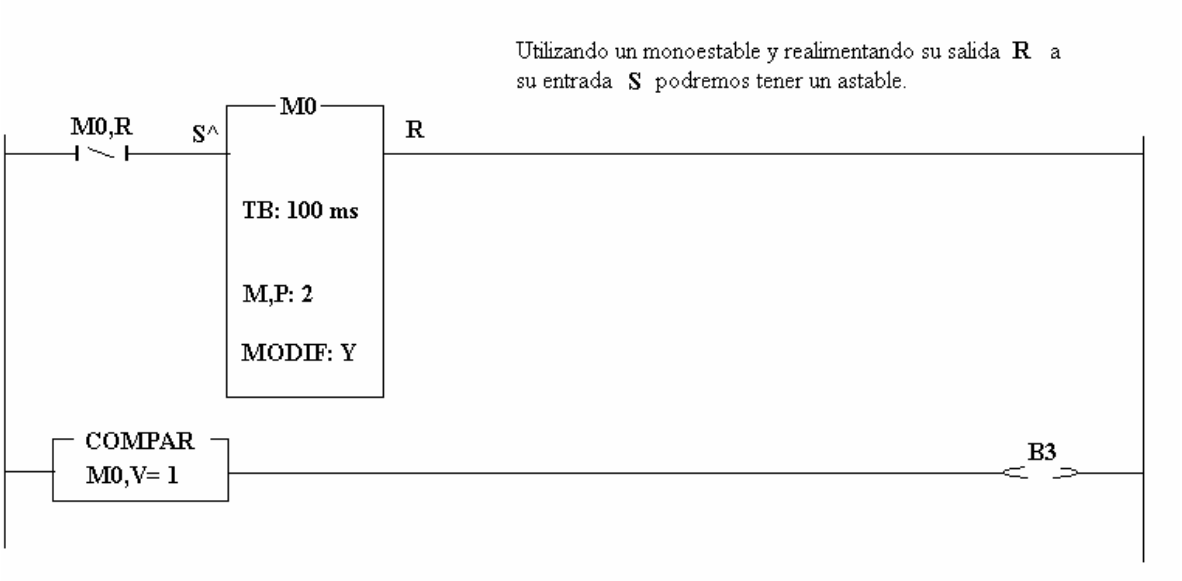
ANEXO # 2

El programa realizado en PL7-2 es exactamente igual al de ISaGRAF con la única diferencia que aparecen los bloques para la comunicación con la PC.

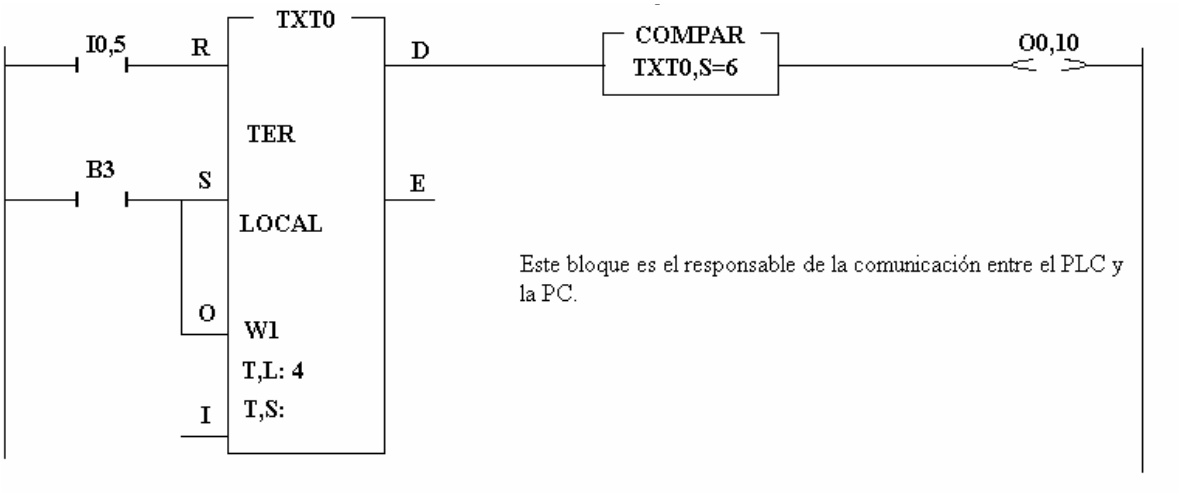
En esta primera sentencia lo que se hace es mandar el estado de las salidas y entradas desde la cero hasta la dieciséis para las palabras internas W1 y W2 respectivamente para luego leerlas.



Con el uso de el bloque monoestable logramos realizar un astable para el envio de los datos constantemente, a la PC.

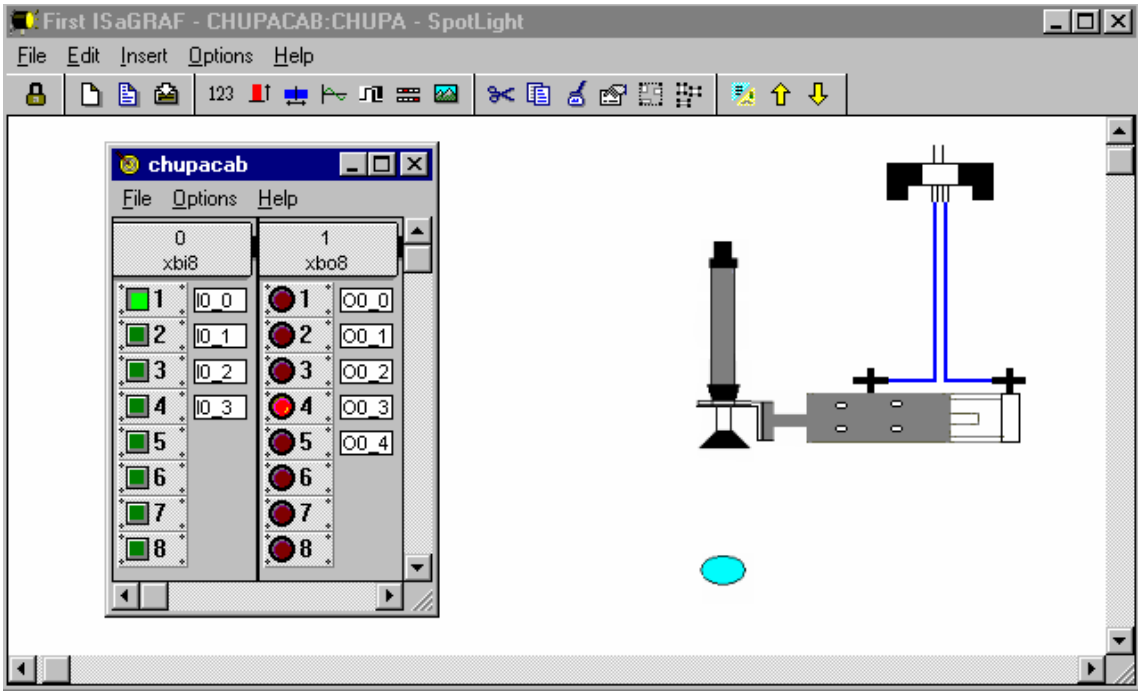


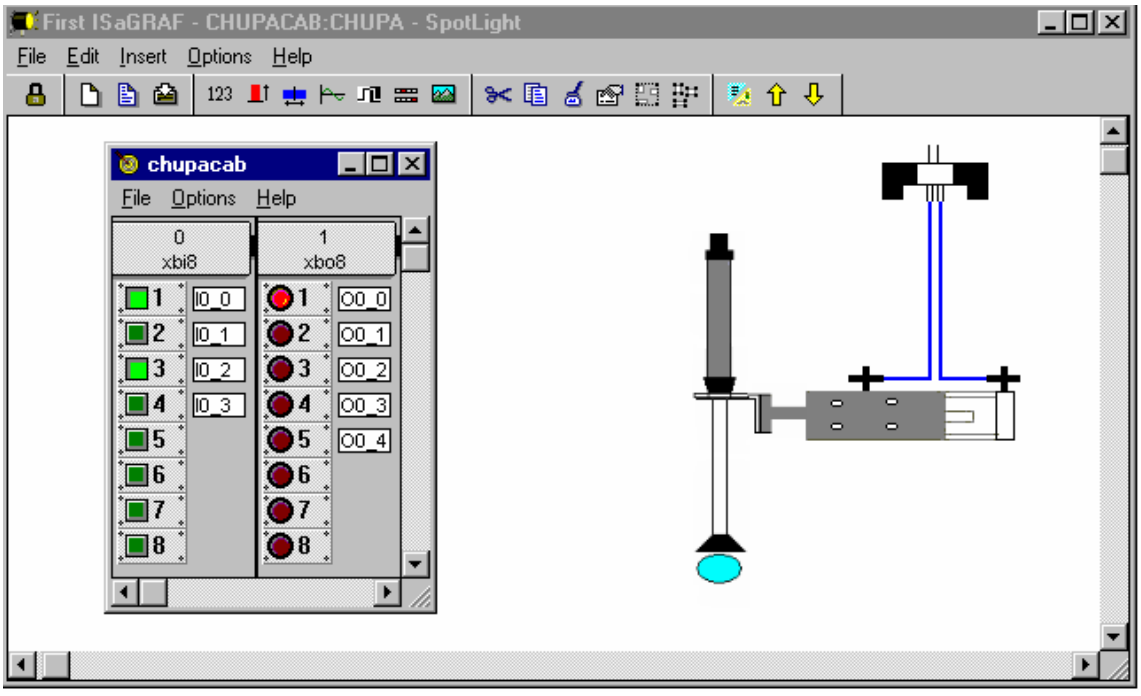
Cuando se activa el bit número tres (B3) este activa la entrada set y la entrada O del bloque texto con lo que se le indica la operación de escritura. El W1 le indica que es a partir de esa palabra lo que va a enviar y la cantidad se le manda en T,L, o sea, 4 bytes con esto se garantiza enviar las palabras W1 y W2 por el puerto serie.

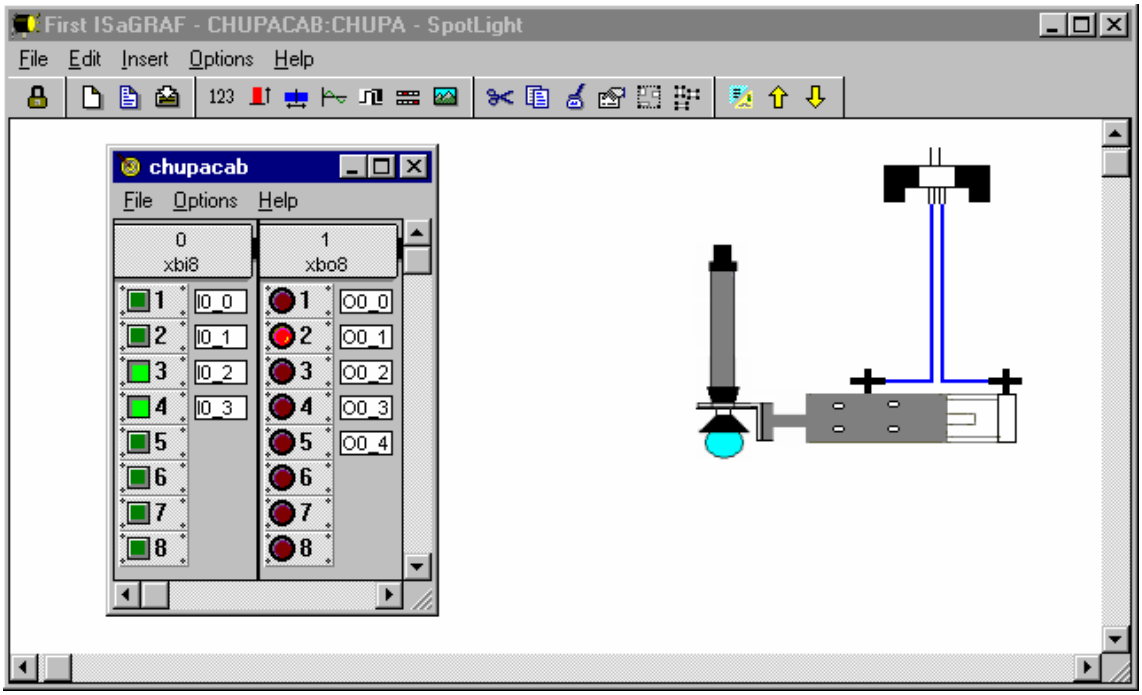


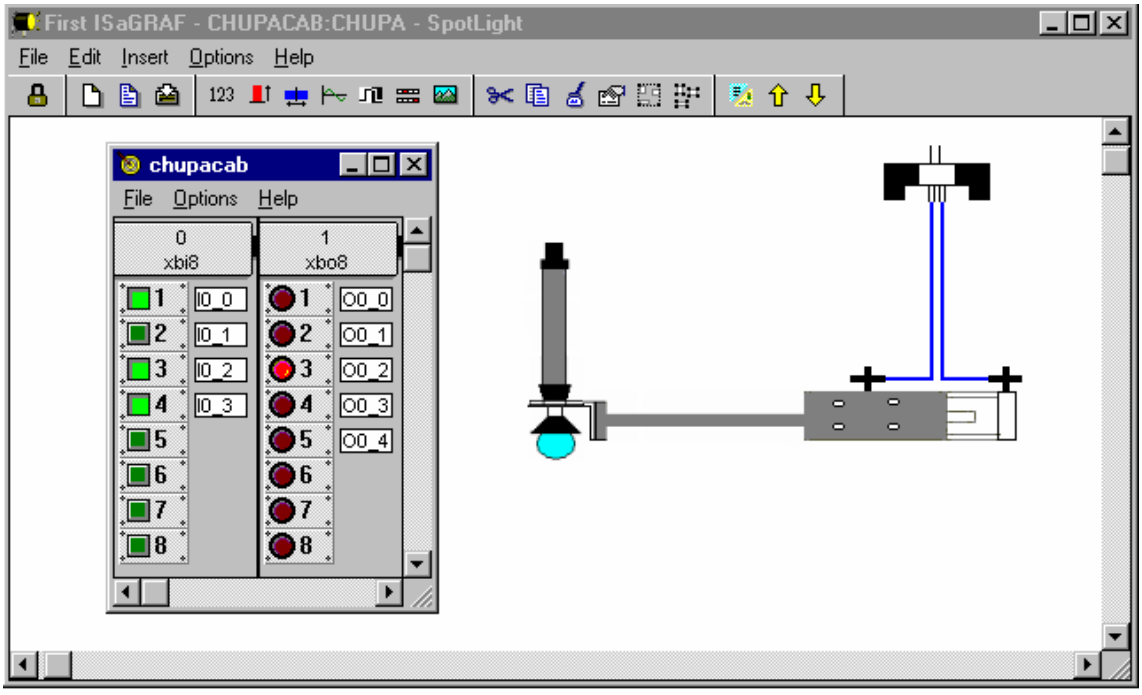
ANEXO #3

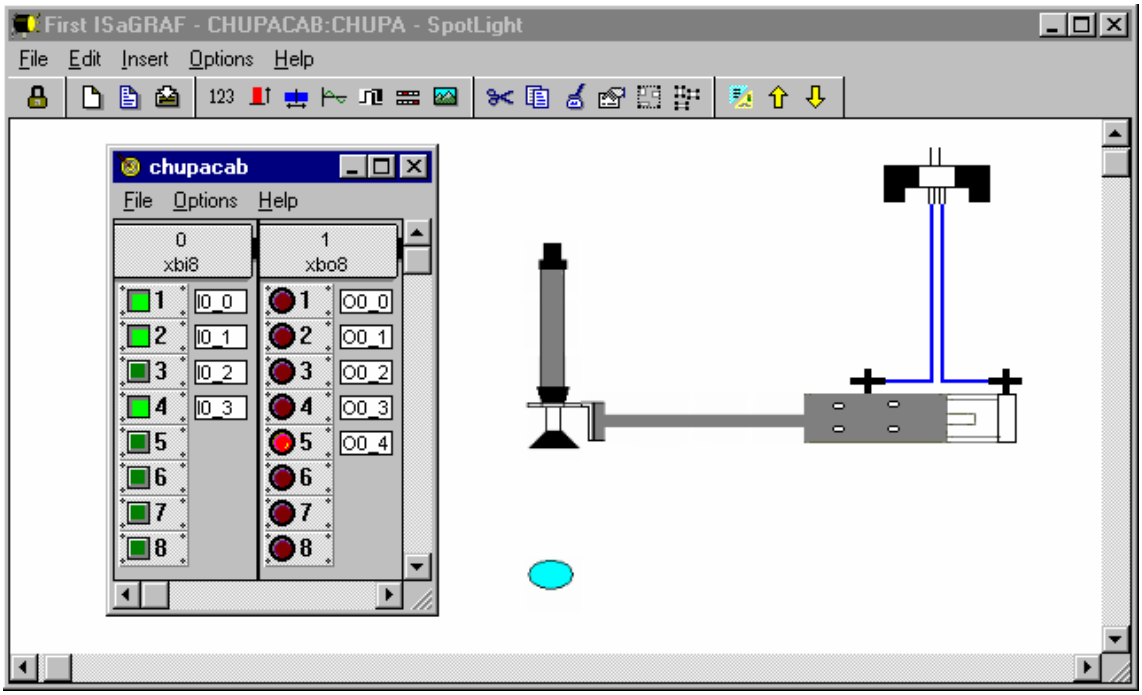
A continuación se muestra la simulación realizada en el ISaGRAF.

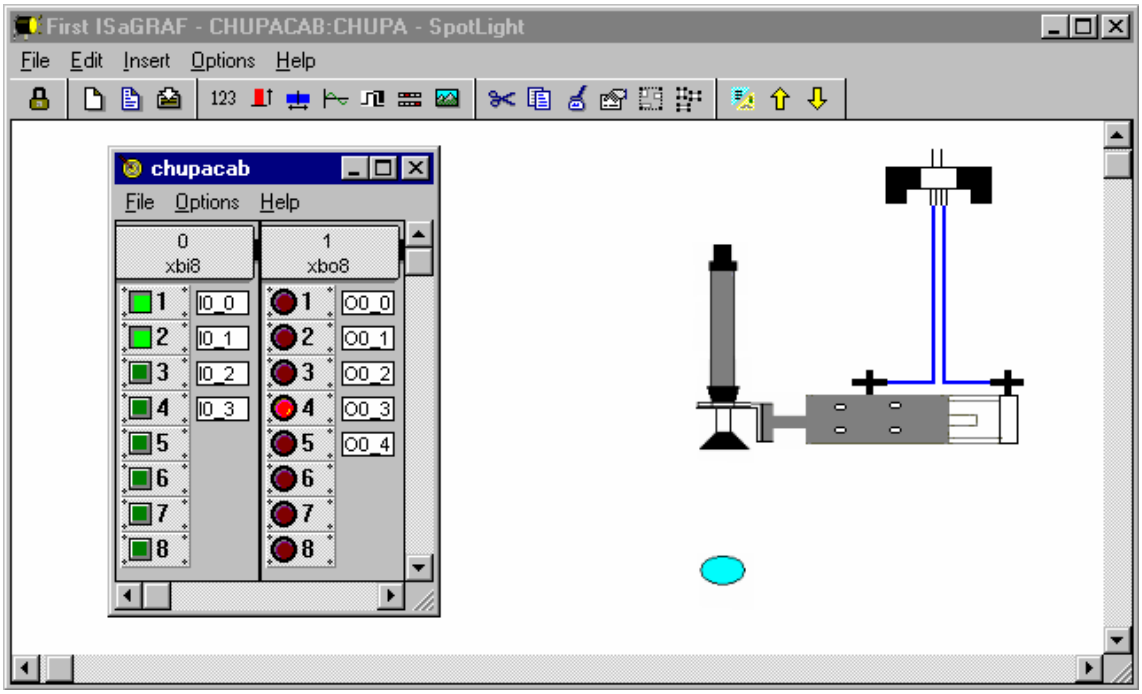





First ISaGRAF - CHUPACAB:CHUPA - SpotLight
File Edit Insert Options Help
chupacab
File Options Help
0
xbi8
1
xbo8
I0_0
I0_1
I0_2
I0_3
O0_0
O0_1
O0_2
O0_3
O0_4


First ISaGRAF - CHUPACAB:CHUPA - SpotLight
File Edit Insert Options Help
chupacab
File Options Help
0
xbi8
1
xbo8
I0_0
I0_1
I0_2
I0_3
O0_0
O0_1
O0_2
O0_3
O0_4

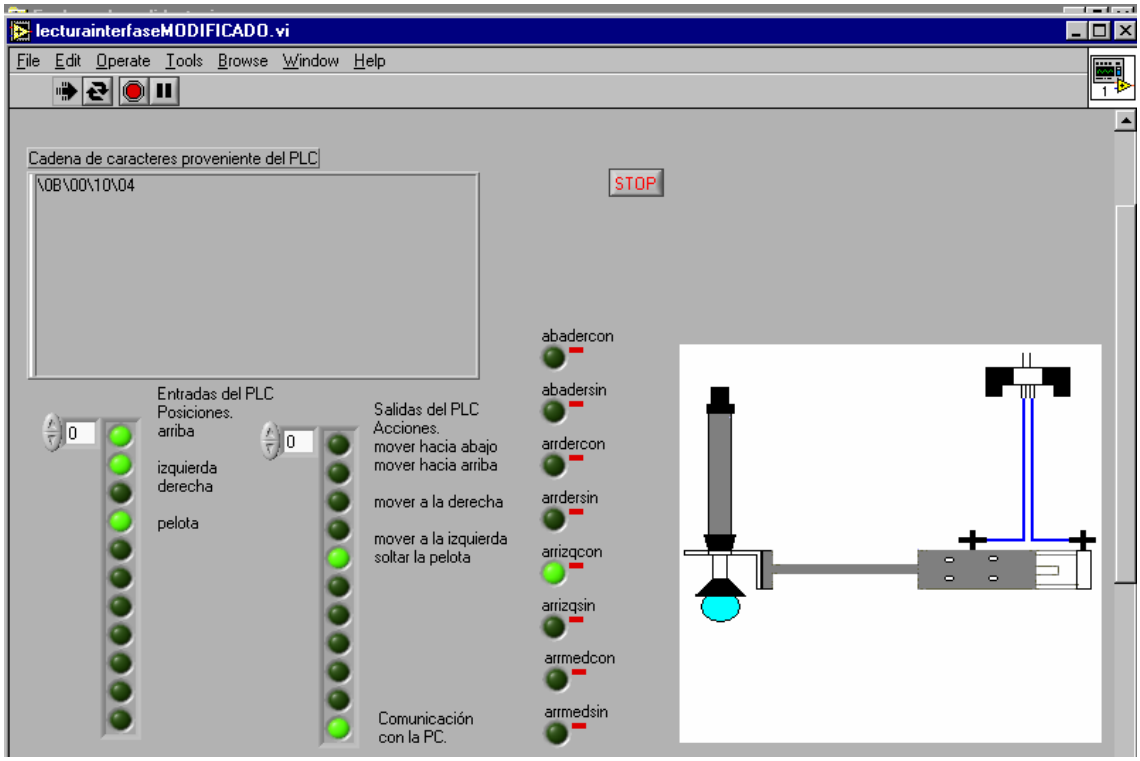

First ISaGRAF - CHUPACAB:CHUPA - SpotLight
File Edit Insert Options Help
chupacab
File Options Help
0
xbi8
1
xbo8
I0_0
I0_1
I0_2
I0_3
O0_0
O0_1
O0_2
O0_3
O0_4


First ISaGRAF - CHUPACAB:CHUPA - SpotLight
File Edit Insert Options Help
chupacab
File Options Help
0
xbi8
1
xbo8
I0_0
I0_1
I0_2
I0_3
O0_0
O0_1
O0_2
O0_3
O0_4

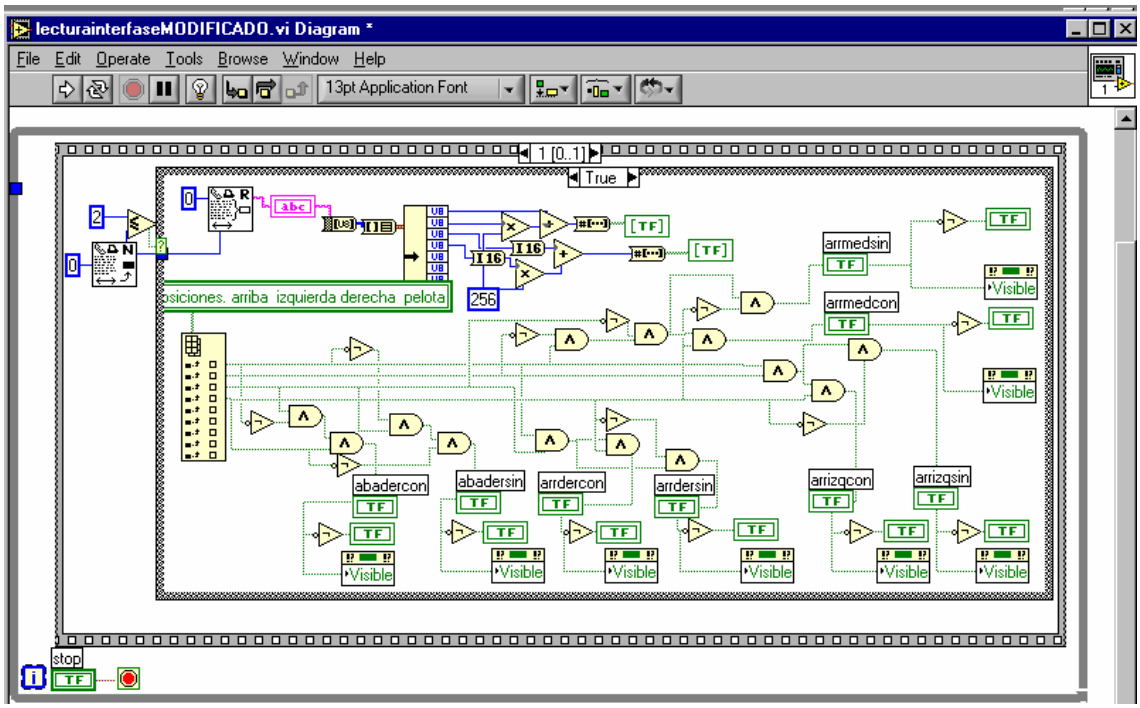
ANEXO # 4

En este anexo se encuentran la interfaz de usuario en tiempo real hecha en labview, así como los programas realizados en dicho lenguaje para lograr la visualización local y remota de la instalación de laboratorio de un manipulador neumático.

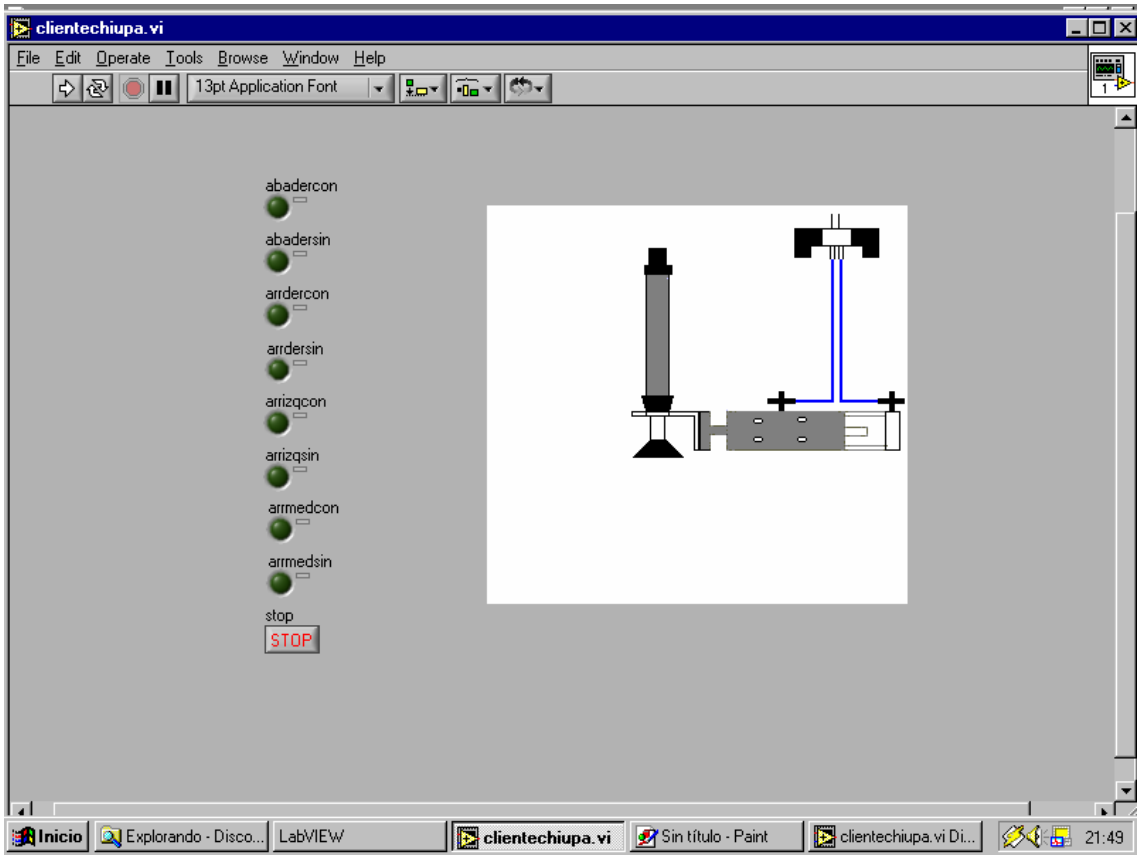
Panel de visualización.



Programa realizado.



A continuación se muestra el panel de visualización remota del cliente de DataSocket.



Este es el programa realizado para la supervisión remota usando el DataSocket.