# UNIVERSIDAD NACIONAL DE INGENIERÍA

## FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA


# DISEÑO Y CONCEPCIÓN DE UN SISTEMA DE CONTROL REMOTO VÍA ETHERNET PARA EL ENCENDIDO Y APAGADO DE UN MOTOR DE 250 HP PARA APLICACIONES MINERAS

**INFORME DE SUFICIENCIA**

**PARA OPTAR EL TÍTULO PROFESIONAL DE:**

**INGENIERO ELECTRÓNICO**

**PRESENTADO POR:**

**JOSÉ LUIS ALBERCO SÁNCHEZ**

**PROMOCIÓN**
**2007-I**

**LIMA-PERÚ**
**2011**

# DISEÑO Y CONCEPCIÓN DE UN SISTEMA DE CONTROL REMOTO VÍA ETHERNET PARA EL ENCENDIDO Y APAGADO DE UN MOTOR DE 250 HP PARA APLICACIONES MINERAS

Un agradecimiento a mis padres
por su vuestro apoyo y confianza

## SUMARIO

El presente trabajo propone y analiza la implementación de un sistema electrónico que permita controlar los diferentes tipos de arranque de motores de inducción de manera remota sobre la red ethernet para aplicaciones mineras. Este control remoto abarca desde el diseño del circuito electrónico, manejo de protocolos seriales y TCP/IP, armado y ensamblado de los componentes que están distribuidos en las cajas de protección IDE, la programación en los microcontroladores PIC y, además la programación en .net que es el software utilizado para el desarrollo de la interfaz de usuario

La interface de usuario fue desarrollada utilizando el software de visual .net. Dicha interface permite el envío de comandos de manera remota, los cuales serán recibidos por el microcontrolador PIC a través de la red ethernet. La programación del microcontrolador PIC fue desarrollada en lenguaje C. La función de este microcontrolador será controlar relays que permita el encendido y apagado del motor. Sin embargo estos comandos de encendido y apagado serán enviados remotamente desde la interface de usuario a través de la red ethernet.

Este prototipo ha sido evaluado de manera real mediante la implementación de un hardware a base del microcontrolador PIC que controlará el arranque y el parado del motor. Además, se implementó el circuito de control de relevadores el cual está conformado por un conjunto de relays y por un ventilador de 24 Vdc. El diseño de los circuitos, tanto para el circuito de control PIC y el circuito de control de reveladores (manual) para el arrancado y apagado del motor, están dentro de dos cajas IDE, respectivamente. Sin embargo, para pruebas de campo (unidad minera) para el control de arranque del motor de 250 Hp, se utilizó el hardware del circuito de control PIC protegido por caja de fibra de vidrio. La versión de visual .net utilizada es la 2008, en c#, que es un lenguaje orientado a objetos y tiene un importante papel en la arquitectura de Microsoft Framework y la concepción de interfaces de usuario.

# ÍNDICE

# PRÓLOGO

El presente proyecto se trata del diseño, simulación, implementación y pruebas del circuito de un control remoto on/off. Para efectos de prueba local se construyó un hardware que se le llama circuito de control de relevadores que contiene el circuito base de control de arranque en la cual se conecta al hardware circuito de control PIC; a su vez este hardware circuito de control PIC ha sido construido para pruebas en la minería, acoplando al tablero de control del ventilador minero. Este trabajo contendrá:

El capítulo I trata sobre el planteamiento de ingeniería del problema y síntesis de trabajo.

El capítulo II trata en forma resumida acerca de los tipos de arranque y apagado para motores AC trifásico, breve explicación del protocolo TCP/IP y, además, las definiciones de términos del trabajo.

El capítulo III trata sobre el diseño, simulación, programación, construcción del hardware control PIC y del circuito de control de relevadores para un motor dc de 24 V para su demostración, además breve explicación de la programación en c# de visual .net 2008.

El capítulo IV trata sobre la verificación de las pruebas del sistema del control on/off del motor dc y también con el ventilador minero 250HP, se indicará cuánto es el presupuesto y propuesta de un proyecto óptico para bombas de agua.

Finalmente, se presentará las conclusiones, observaciones y así como recomendaciones al respecto.

# INTRODUCCIÓN

El motivo de la elección del tema ha sido en construir un prototipo electrónico y programación en software que permita facilitar al usuario el control de arranque y parada de los ventiladores mineros y además proponer un sistema de menor costo para este fin, usando los conocimientos adquiridos para armar este trabajo propuesto. Futuramente se busca desarrollar el monitoreo de sus estados de todos los ventiladores mineros en interior mina, controlarlos el arrancado y apagado del motor, verificar si necesita mantenimiento y sacar reportes estadísticos para su control del día. Estos conocimientos a utilizar para la construcción del sistema propuesto, tiene que ser analizada en los conceptos de los diferentes tipos de arranques de motores eléctricos, los conceptos de comunicación serial, los conceptos TCP/IP, equipos de red a usar, instalación y configuración, conceptos de tramas, desarrollo del prototipo electrónico basado en el microcontrolador PIC de Microchip y desarrollo de la programación del interfaz de usuario, ya que en este proyecto se usó la programación c# de visual .net 2008. Con estos conocimientos se puede armar infinidad de alternativas o caminos para la construcción del sistema de control propuesto.

Este trabajo de control propuesto es importante en el mundo industrial como la minería, en otros; ya que se puede controlar y automatizar los motores eléctricos que cuya función es de mover y/o desplazar sólido, líquidos y gases. Ya que estos motores cuentan con un circuito de control cuya función es apagar y encender el motor. Para este presente trabajo se usó el microcontrolador PIC que tengan un módulo funcional de comunicación serial USART/SCI (Universal Synchronous Asynchronous Receiver Transmitter/Serial Comunication Interface), que puede acceder a los recursos del sistema completo, pero hay desventaja en trasmitir los datos mediante comunicación serial que son distancia limitada, los costos son altos en instalación y además es asignada a sola una pc, en cambio con las redes TCP/IP se pasó a conectar todo tipo de dispositivos directamente a la red LAN. Esto permite reducir los costos de instalación, controlar los dispositivos desde cualquier PC y eliminar las limitaciones de distancia, entonces se consiguió un equipo que convierte los protocolos seriales a protocolos TCP/IP, para que se conecte con el PIC, así obtener la facilidad de controlar a distancia el arrancado y apagado del motor desde una pc.

# CAPÍTULO I
# PLANTEAMIENTO DE INGENIERÍA DEL PROBLEMA

## 1.1. Descripción de problema

La importancia en el sistema de ventilación, que es la distribución de aire en un sistema de ventilación de minas, es la asignación de caudales de aire en cantidad y calidad al interior de los diversos sectores de la mina, demandantes del recurso, de manera que se pueda lograr medioambiente subterráneos aptos para el normal desempeño de los trabajadores y una óptima operación de las instalaciones y equipos. En caso de que uno de los ventiladores se averíe en el sistema no cumpliría dicha función de acuerdo con el criterio de la distribución de aire de los ventiladores en un sistema de ventilación (equipamiento requerido como ventiladores principales, ventiladores auxiliares reforzados, dispositivos de control de flujo, reguladores, tableros de control, otros), por lo tanto estaría en riesgo la salud de los trabajadores, por no obtener ciertos estándares de oxígeno necesario y un correcto desalojo de gases. Es importante identificar la posición y el estado del motor en el interior mina y además recopilar otros datos posibles que permitan visualizar el usuario para su control del día.

## 1.2. Objetivos del trabajo

Facilitar el trabajo al usuario de controlar a distancia el estado de los motores a partir de los circuitos de control para los diferentes tipos de arranques, usando equipos de red ethernet, construcción del hardware electrónico y una programación para el interfaz de usuario y teniendo en cuenta la señal de realimentación del mismo para saber si está funcionando adecuadamente.

## 1.3. Evaluación del problema

Dado que la instalación de ventiladores de mediana capacidad, actuando como reforzadores para atender niveles de producción, reducción y hundimiento, es una opción de alta probabilidad de implementación, se hace necesario que en la eventualidad de proponer la instalación y operación masiva de un alto número de tales ventiladores al interior de los sectores, se considere la implementación de un sistema de monitoreo y

control centralizado del estado y operación de estos equipos. El mismo concepto es válido para la eventualidad de que al interior del proyecto se proponga instalar reguladores de flujos de aire, los cuales además de poder ser operados en forma manual (control local), puedan también ser conectados a un sistema de monitoreo y control a distancia (actuación de tipo telecomando). Las funciones mínimas con que debería implementarse el sistema de monitoreo y control centralizado de un sistema de distribución aire, en mina subterránea, son las siguientes: captura de información del estado de operación de ventiladores reforzadores y reguladores de flujos con servomotores incorporados, es decir, monitoreo en tiempo real; actuación (control del tipo ON/OFF) sobre los comandos de todos los motores de ventiladores reforzadores de caudal fijo, por sector; y actuación sobre los servomotores, de manera tal de controlar las diferentes posiciones de abertura de reguladores y comando de operación ventiladores reforzadores de caudal variable, por sector.

## 1.4. Limitaciones de trabajo

Este proyecto generalmente usan PLC, LABVIEW; sin embargo para poder reducir los costos se adoptó usar microcontroladores PIC y visual .net en c#, y luego hacer las pruebas de caso. El servidor serial "TIBBO" que es un convertidor de protocolos TCP/IP a protocolos seriales, se adquirió este equipo en la cual se conecta con el microcontrolador PIC, debido a que el microcontrolador PIC tiene un módulo funcional de comunicación serial USART/SCI. Si se hubiera diseñado el servidor serial se hubiera utilizado el PIC 18F97J60 y el ENC28J60. El uso del sensor de temperatura LM35, forrado de encapsulamiento metálico no se podía adherir en el rotor del motor, debido que el orificio era pequeño y estrecho hacia al rotor del motor, por lo cual se necesitaría de un sensor de temperatura como una cinta adhesiva delgada o tal vez buscar un sensor adecuado que pueda introducirse y pegarlo lo más cercano al rotor o tal vez al mismo rotor, para poder así adquirir datos de temperatura del rotor del motor eléctrico, no se pudo adquirir dicho sensor, entonces para este trabajo se hizo un control on/off en lazo directo.


**Figura 1.1** Diagrama de bloque en lazo directo para el control a distancia el arranque y apagado de un motor eléctrico

## 1.5. Síntesis del trabajo

Consiste en controlar un motor de prueba de 250 HP, el cual es usado para controlar un ventilador de interior mina. El proyecto busca futuramente el control de ventilación de todo el interior mina, y controlar los sistemas de arranque de los motores en forma remota, luego hacer la automatización en todo el sistema de ventilación. El proyecto consta de las siguientes partes:

- Control de encendido y apagado: Se diseñó una circuito electrónico que mediante relays es capaz de comunicarse con el motor eléctrico en poder arrancarlo y apagarlo, verificando sus diversos procesos de arranque y teniendo en cuenta la señal de realimentación del mismo para saber si está funcionando adecuadamente. Se tendrá que diseñar una adquisición electrónica con lo cual se logra obtener datos de temperatura del bobinado del motor para hacer tablas y posteriormente un mantenimiento preventivo.
- Servidor serial: Es una de los componentes del sistema de comunicación que se encarga de transformar de protocolos ethernet a protocolos serial y viceversa.
- Comunicación ethernet: Este equipo se comunica con toda la LAN a través de un protocolo ethernet, por el cual permite el programa visual .net hacer el monitoreo desde cualquier punto de red.
- Interfaz de usuario en Microsoft .net: Fue construida netamente en la programación c# de visual .net 2008, así mismo la publicación del sistema para el usuario.

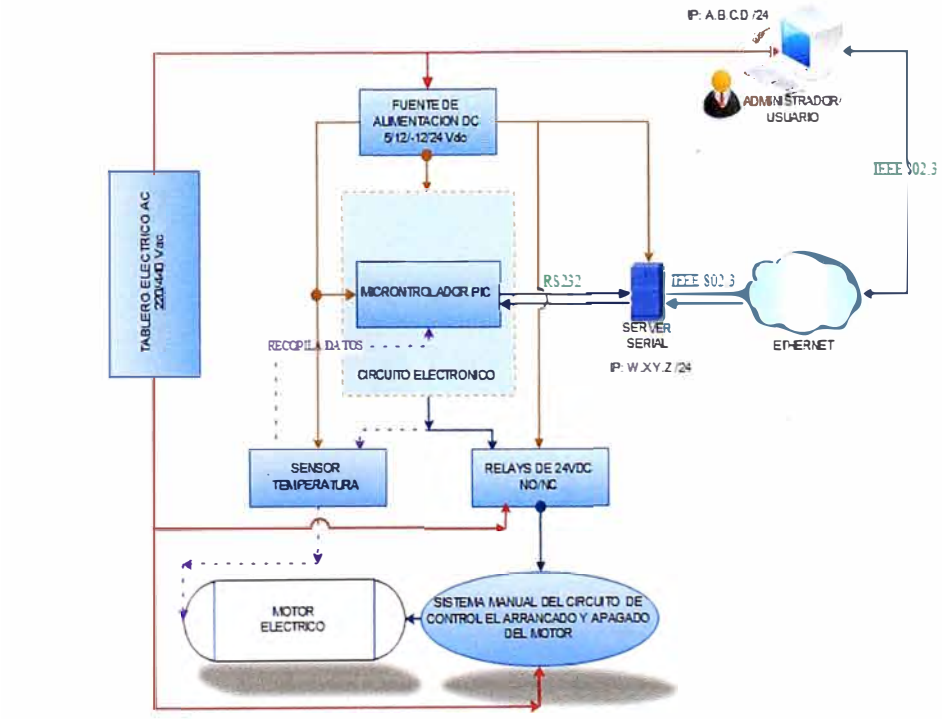


**Figura 1.2** Esquema del sistema para el control on/off para el motor eléctrico

# CAPÍTULO II
# MARCO TEÓRICO CONCEPTUAL

## 2.1. Marco histórico de los motores de inducción

La máquina de inducción es el convertidor electromecánico más utilizado en la industria. Las ideas fundamentales sobre los motores de inducción fueron desarrolladas por Nicola Tesla, hacia fines del siglo XIX, en la década de 1880, quien en esa época presentó un artículo ante el American Institute of Electrical Engineers (AIEE, predesor de hoy IEEE – Institute of Electrical and Electronics Engineers) en el cual describió los principios básicos del motor de inducción de rotor devanado, junto con ideas para desarrollar otros dos importantes motores de corriente alterna: el motor sincrónico y el motor de reluctancia.

Tesla había sugerido la idea de las ventajas que poseía la corriente alterna, cuyos niveles de tensión pueden ser variados mediante transformadores sobre la corriente continua cuyas dificultades de transmisión para la época ya comenzaban a ser evidentes. Desde ese crucial momento y hasta la actualidad, la máquina de inducción ha ido copando la inmensa mayoría de aplicaciones en la industria, en el comercio y en el hogar. En la figura 2.1 se muestra un modelo de la máquina diseñada por Tesla, cuyo original está expuesto en el museo Smithsoniano de Washington DC [9].

**Figura 2.1** Modelo de la máquina de inducción bifásica diseñada por Tesla [9]

La idea básica del motor de inducción fue descrita en 1888; el motor en sí no estaba aún completo. Hubo un periodo inicial de rápido desarrollo seguido de una serie de lentos mejoramientos que continuaron evolucionando hasta hoy.

El motor de inducción moderno se construyó entre 1888 y 1895. Durante ese periodo se desarrollaron fuentes de potencia de dos y tres fases para producir campo magnético rotacionales dentro del motor, devanados estatóricos distribuidos y se introdujo el rotor de jaula de ardilla. Hacia 1896 estuvieron disponibles en el comercio motores de inducciones trifásicas plenamente reconocidos y funcionales.

Entre aquella época y comienzos de los años de 1970 hubo progresos continuos en la calidad de los aceros, las técnicas de fundición, los aislamientos y otros elementos utilizados en los motores inducción. Estas tendencias dieron como resultado un motor más pequeño con una potencia de salida determinada, que proporcionó ahorro considerable en los costos de construcción. En efecto, un motor moderno de 100 hp es igual en tamaño físico a uno de 7.5hp de 1897 [8].

Estos motores requieren un mantenimiento mínimo; pueden operar convenientemente en ambientes peligrosos y tienen una tasa de falla muy reducida. Algunas limitaciones tales como el ajuste de la característica par-velocidad, la intensidad de las corrientes durante el arranque, la regulación de velocidad y el rendimiento han sido resueltas o mejoradas con diseños ingeniosos e incorporando controladores electrónicos de potencia. En la figura 2.2 se muestra un despiece de la máquina de inducción con rotor de jaula de ardilla donde se indican las principales partes constitutivas [9].

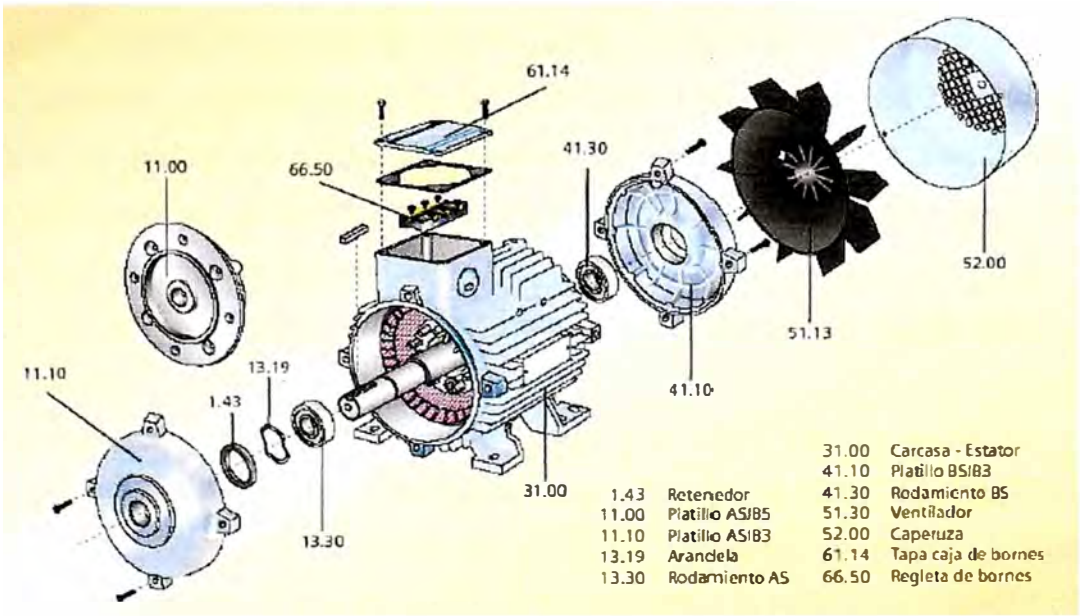


**Figura 2.2** Despiece de un motor de inducción industrial de rotor de jaula de ardilla [9]

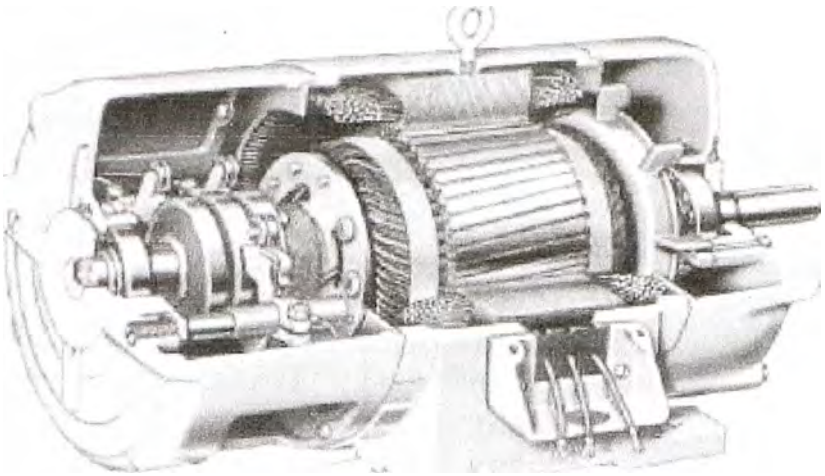

**Figura 2.3** Máquina de inducción de rotor bobinado con anillos deslizantes [9]

## 2.2. Bases teóricas

### 2.2.1. Fundamentos básicos sobre el manejo de un motor trifásico de inducción

Para poder manejar y controlar un motor trifásico de inducción, de manera correcta, primero se debe conocer las partes constitutivas del mismo, sus características y los diferentes comportamientos que presenta durante su funcionamiento, todo esto con la finalidad de encontrar la mejor manera de utilizarlo sacando provecho al máximo de su ventaja y desventajas [7].

#### 2.2.1.1. Principio de funcionamiento del motor de inducción

El motor de inducción es el motor de corriente alterna más utilizado, debido a su fortaleza y sencillez de construcción, buen rendimiento y bajo costo así como a la ausencia de colector y al hecho de que sus características de funcionamiento se adaptan bien a una marcha a velocidad constante. El motor de inducción no necesita escobillas ni colector. Su armadura es de placas de metal magnetizable. El sentido alterno de circulación, de la corriente es las espiras del estator genera un campo magnético giratorio que arrastra las placas de metal magnetizable y las hace girar.

El principio de funcionamiento de un motor de inducción se puede mostrar de la siguiente forma:

Se suspende un imán permanente de un hilo sobre una tornamesa de cobre o aluminio que gira en un cojinete colocado en una placa fija de hierro. El campo del imán permanente se completa así a través de la placa de hierro. El pivote debería estar relativamente sin fricción y el imán permanente debe tener la suficiente densidad de flujo. Cuando gira el imán en el hilo, se observará que el disco que está debajo gira con él.

El disco sigue en movimiento del imán, como se muestra en la figura 2.4 debido a las corrientes parásitas inducidas que se producen por el movimiento relativo de un conductor (el disco) y el campo magnético [7].

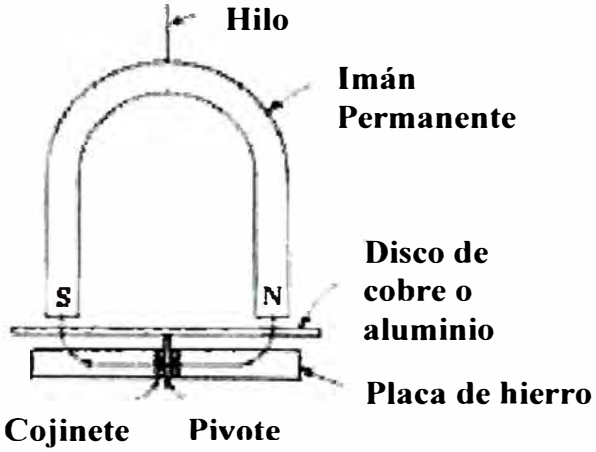


**Figura 2.4** Funcionamiento del motor de inducción [7]

Por la ley de Lenz, la dirección del voltaje inducido y de las corrientes parásitas consecuentes produce un campo magnético que tiende a oponerse a la fuerza o movimiento que produjo el voltaje inducido.

Las corrientes parásitas que se producen tienden a producir a su vez un polo S unitario en el disco en un punto bajo el polo N giratorio del imán y un polo N unitario en el disco bajo el polo S giratorio del imán, figura 2.5. Por lo tanto, siempre que el imán continúe moviéndose, continuará produciendo corrientes parásitas y polos de signo contrario en el disco que está abajo. El disco, por lo tanto, gira en la misma dirección que el imán, pero debe girar a velocidad menor que la del imán. Si el disco girara a la misma velocidad que la del imán, no habría movimiento relativo entre el conductor y el campo magnético y no se producirían corrientes parásitas en el disco.

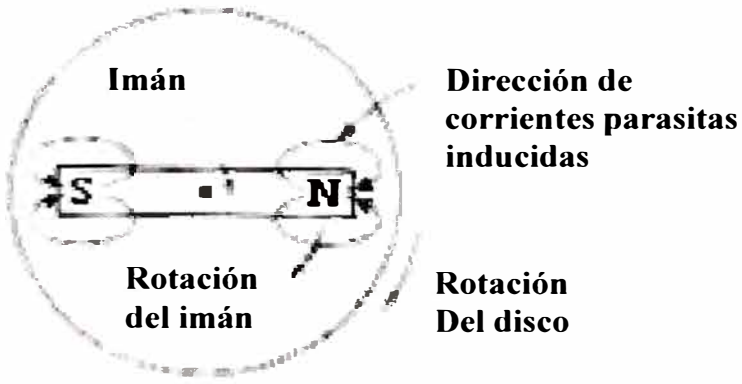


**Figura 2.5** Corrientes parasitas inducidas [7]

### 2.2.1.2. Arrancadores de Motores de inducción AC

Cuando los motores de inducción AC arrancan, hay una gran cantidad de corriente que fluye en cuanto la potencia se aplica al motor. Es típicamente de 6 a 10 veces en su valor nominal. Esto significa que el dispositivo es capaz de manejar una gran cantidad de corriente que lo convierte a una potencia en HP. En la siguiente tabla nos indica qué corriente de arranque se necesita para arrancar un motor en Horse Power según el código eléctrico nacional del 2005 [4].

**TABLA N° 2.1** Corriente de plena carga para motores trifásicos de CA [4]

| *Horsepower* | *Corriente de 460 Voltios, 3 fases* | *Horsepower* | *Corriente de 460 Voltios, 3 fases* |
|---|---|---|---|
| ½ HP | 1.1 Amp. | 50 HP | 65 Amp. |
| 3/4 | 1.6 | 60 | 77 |
| 1 | 2.1 | 75 | 96 |
| 1 1/2 | 3 | 100 | 124 |
| 2 | 3.4 | 125 | 156 |
| 3 | 4.8 | 150 | 180 |
| 5 | 7.6 | 200 | 240 |
| 7 1/2 | 11 | 250 | 302 |
| 10 | 14 | 300 | 361 |
| 15 | 21 | 350 | 414 |
| 20 | 27 | 400 | 477 |
| 25 | 34 | 450 | 515 |
| 30 | 40 | 500 | 590 |
| 40 | 52 | | |

Incluso para un relativamente pequeño motor de 10 caballos de fuerza, la corriente de funcionamiento es de 14 amperios. Eso haría que el corriente de arranque estaría entre 84 y 140 amperios. Claramente esto requiere un relativo interruptor pesado. Este interruptor pesado consta de tres partes principales. La primera parte, se trata de una bobina que se energiza para operar; la segunda parte, los 3 polos de relé o contactor; y la tercera parte es un relé de sobrecarga. El relé de sobrecarga consiste en una corriente de

medición de dispositivo y un contacto auxiliar. Por lo general, también hay contactos auxiliares que soportan baja intensidad de corriente que operan al mismo tiempo que los contactos auxiliares principales de gran intensidad de corriente. En la figura 2.10 muestra un dibujo esquemático de las principales partes de un motor trifásico de arranque.

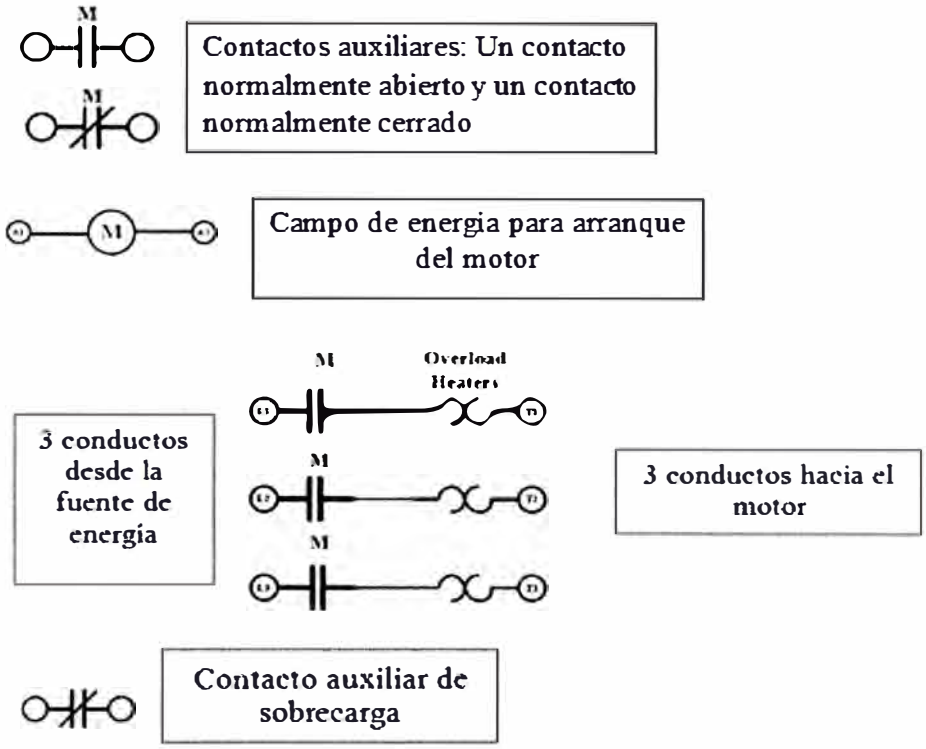


**Figura 2.10** Muestra el esquema de las partes de un arrancador [4]

La función de cada una de las partes es la siguiente:

a. La alimentación se suministra a la bobina M a través de los terminales A1, A2.

b. Todos los contactos M cambian de estado, los tres contactos principales se cierra y los 2 más pequeños contactos auxiliares cambia de estado. El normalmente abierto (NA) el contacto se cierra y el contacto normalmente cerrado (NC) el contacto se abre.

c. Si se energiza el campo de energía de M, entonces los contactores principales M se cierran por lo tanto, la energía eléctrica en los terminales L1, L2, L3 llega a los terminales T1, T2 y T3 (ver figura 2.10).

d. Si hay un motor conectado a los terminales T, entonces la corriente fluye hacia el motor, lo cual desarrolla el torque y comienza a girar el motor.

e. La corriente hará que adquiera calor a los calentadores de sobrecarga.

f. Cuando fluye demasiada corriente hacia el motor por mucho tiempo entonces los calentadores de sobrecarga hacen que el contactor de sobrecarga se abra [4].

En la figura 2.11, se muestra un esquemático de un circuito típico que es usado para arrancar y parar un motor.

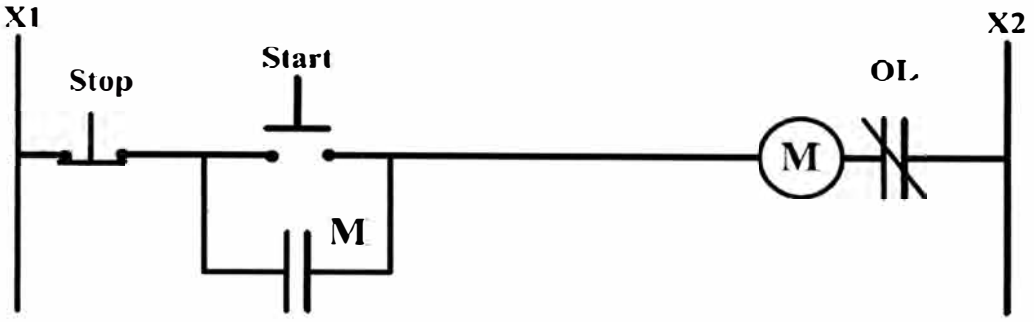


**Figura 2.11** Circuito star y stop [4]

Hay que destacar que X1, X2 son para ciertos voltajes generalmente magnitudes bajas, así como la potencia cuya magnitud es baja. En X1, X2 los voltajes pueden ser 440,220, 120 Voltios; en AC o en DC los voltajes serían 24 o 12 Voltios. La única función de X1, X2 es la fuente de alimentación para suministrar una corriente que necesita para energizar la bobina M. En operación, si el botón star está presionado, la corriente fluye a través de la bobina M es decir empieza a energizarse, causando así el cierre de los contactos M. A este efecto se le llama frecuentemente *sellos en los contactos*, cuando los contactos M están cerrados y hay flujo de corriente a través de la bobina M, que es continuo y, se encuentra energizado. Si se presiona el botón stop, el flujo de corriente a través de la bobina M se interrumpe (deja de energizar) y entonces los contactos M se abren. Por lo tanto cuando se presiona el botón stop, la bobina M deja de energizar [4].

Se observa de la figura 2.11 que el contacto OL se encuentra en serie con la bobina M. Si los calentadores de sobrecarga se encuentra a una temperatura caliente entonces el contacto OL se abre, la bobina M perderá energía eléctrica. El contacto de sobrecarga puede autoresetearse o no autoresetea. Si no autoresetea, el operador haría el reseteo de los contactos de sobrecarga por consiguiente el motor se reinicia.

El dibujo de la figura 2.11 es llamado diagrama de escalera. En este esquema de control puede haber muchas líneas de lógica de control. Este dibujo sólo muestra una línea que representa el circuito start y stop de un motor. El esquema de control se complicaría haciendo una docena o más de mil líneas de lógica de control.

El circuito de potencia de un circuito de control del motor se utiliza para conectar una fuente de energía de alto voltaje. Estos contactores son completamente diferentes del contactor auxiliar de energía baja. Estos contactos y contactores están hechos de plata,

porque este metal es un buen conductor y es relativamente resistente a la corrosión [4]. En la figura 2.12 se muestra el circuito de potencia para un motor de arranque. Se observa que el control de energía X1, X2 viene justo a la salida de un transformador de control, que esta energía hará funcionar el motor. La desconexión del circuito principal no sólo se apaga la alimentación del motor, también se apaga la alimentación del transformador de control. El transformador de control es usado para reducir o mantener el control de voltaje obteniendo un nivel de voltaje seguro y para aislar el voltaje de control de las 3 fases de energía de voltaje. También, se observa que los tres fusibles principales son las misma corriente que alimenta hacia al motor. Estos son tipo especial de fusible llamados retardo de tiempo o fusibles de doble elemento. Están diseñadas para transportar una corriente alta por un pequeño periodo de tiempo. En los motores de inducción tiene 6 a 10 veces de la corriente nominal para arrancar el motor, este tipo de fusible es necesario para los circuitos de arranque de motor.

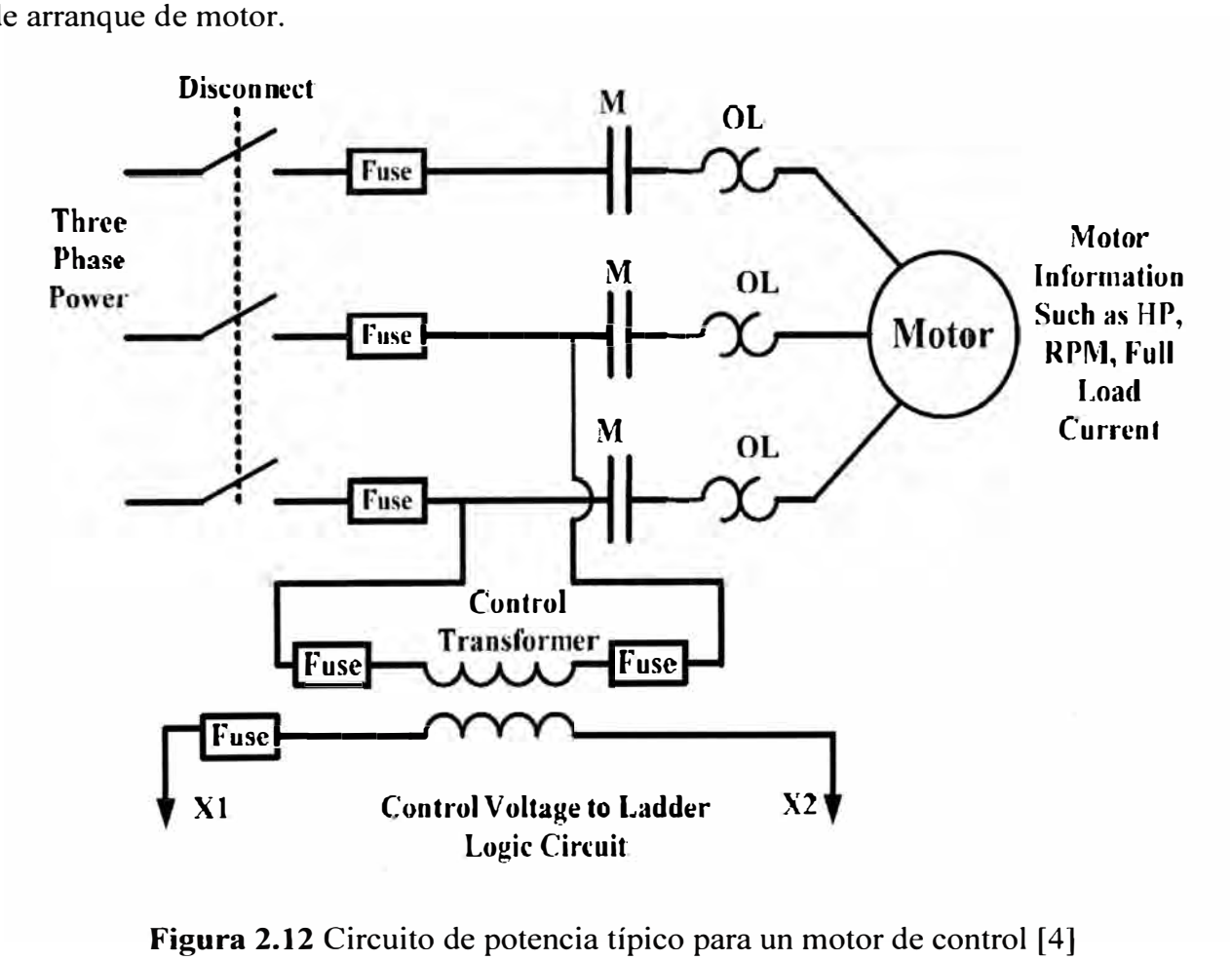


**Figura 2.12** Circuito de potencia típico para un motor de control [4]

Cuando el circuito de potencia y el circuito de control se ponen juntos, se puede verificar el resultado en la figura 2.13. Para recordar, los contactos de potencia llevan corriente hacia los motores son más altos que los contactos auxiliares y de los contactos

overload (sobrecarga). Los contactos de overload y los calentadores son únicos y frecuentemente llamados relay de sobrecargas. Si observamos, en la figura 2.13 uno de los terminales de control de voltaje X2 ha sido conectado a tierra.

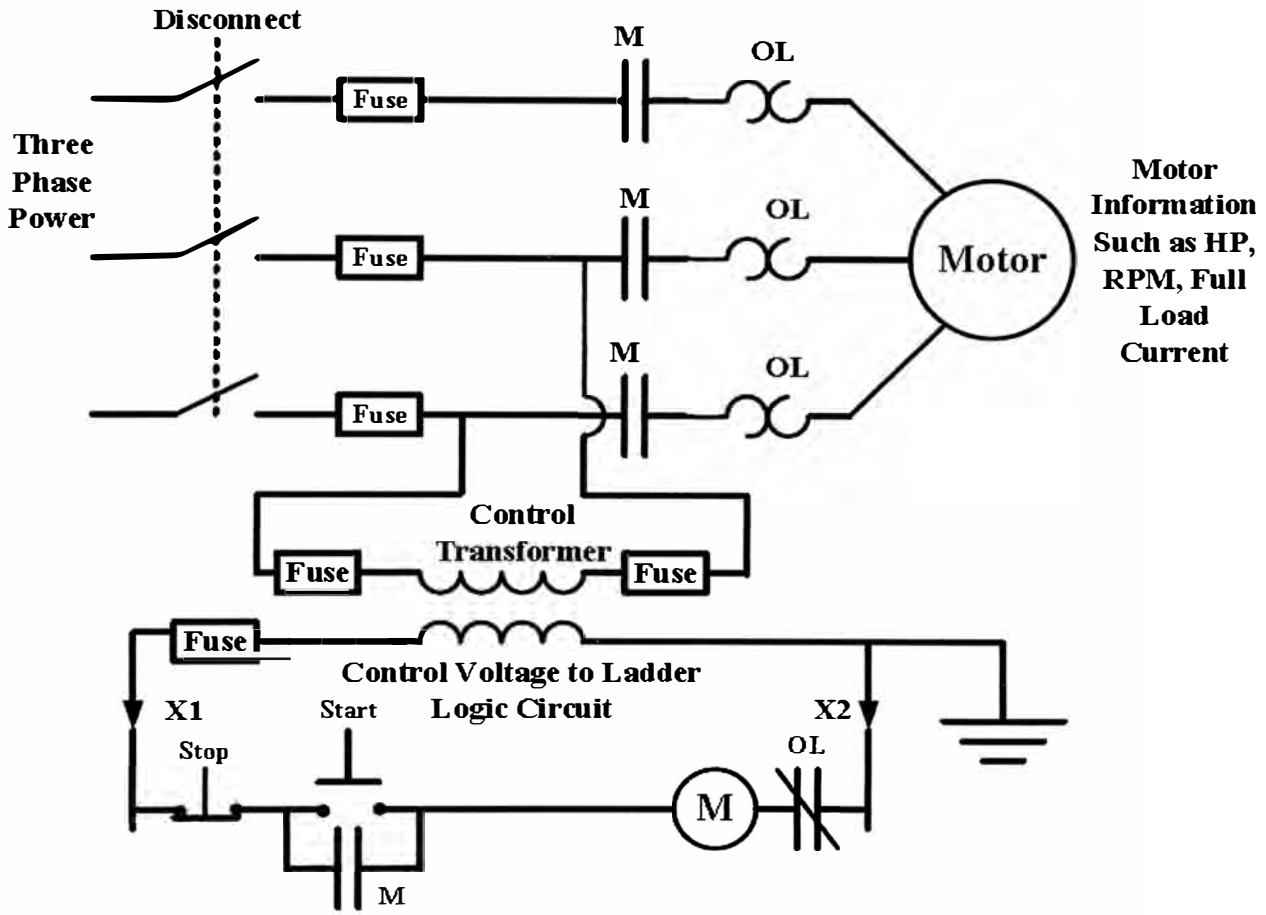


**Figura 2.13** Circuito de control completo para un motor [4]

Hay 2 tipos de reles de sobrecarga térmica. Estos son bi-metal y eutéctica tipo fusión de aleación. En los tipo bi-metal, una tira bi-metálica es calentada por el flujo de corriente que va hacia el motor. A medida que se calienta, se dobla y si se dobla demasiado hace que el contacto de sobrecarga se abra. El tipo de fusión de la aleación eutéctica en realidad tiene dos piezas de metal soldadas entre sí. La corriente que fluye al motor hace que las dos piezas de metal se calienten; y si se ponen demasiado calientes la soldadura se derrite y hace que el contacto de sobrecarga se abra. Ambos tipos de calentadores de sobrecarga están para soportar la corriente de operación a plena carga del motor. El código nacional de electricidad (NEC) permite a los calentadores soportar la corriente a 25 % más de su corriente nominal. Lo mismo es cierto para los fusibles de proteccion del motor.

La desconexión del interruptor y fusibles puede ser reemplazado por un tipo y tamaño del disyuntor. Las nuevas noticias vienen con un relé de sobrecarga electrónico [4].

### 2.2.1.3. Arrancador estrella delta

El método más simple para reducir la tensión de una máquina consiste en conectarla inicialmente en estrella y cuando el deslizamiento es pequeño se cambia la conexión del motor a delta. La tensión final sobre cada bobina de la máquina debe ser su propia tensión nominal. Este método de arranque reduce en 3 veces la tensión nominal de la máquina y la corriente se reduce en esta misma proporción. Los torques eléctricos se reducen a un tercio del torque a tensión nominal. Este procedimiento es uno de los más económicos, pero es necesario disponer de un sistema adecuado de tensiones que permita la conexión delta de la máquina durante el régimen permanente. El cambio de conexión se realiza cuando la máquina alcanza el deslizamiento de operación en la conexión estrella. La orden de cambio puede ser dada por un temporizador si se conoce la inercia de la carga o el tiempo de aceleración a tensión reducida. Si el cambio de conexión se realiza antes de que las corrientes disminuyan, el arrancador pierde efectividad. El tiempo total de arranque con este dispositivo es aproximadamente tres veces mayor que el arranque en directo de la máquina; esto es importante al momento de especificar las protecciones del motor.

A partir de motores de grandes potencias, se requiere alguna forma de limitar las grandes corrientes que fluyen en el inicio que se conoce como corriente de arranque del motor. Una manera de hacer consiste en conectar las tres fases de un motor en una conexión en estrella para el arranque. Esto limitará la tensión en cada bobina y por consecuencia la corriente. Si se conecta el devanado en una configuración delta obtendría el 58 % de la corriente. La figura 2.15 muestra el diagrama que es un esquemático de conexiones de estrella y delta de un motor que pueden ser arrancados por este método [4].

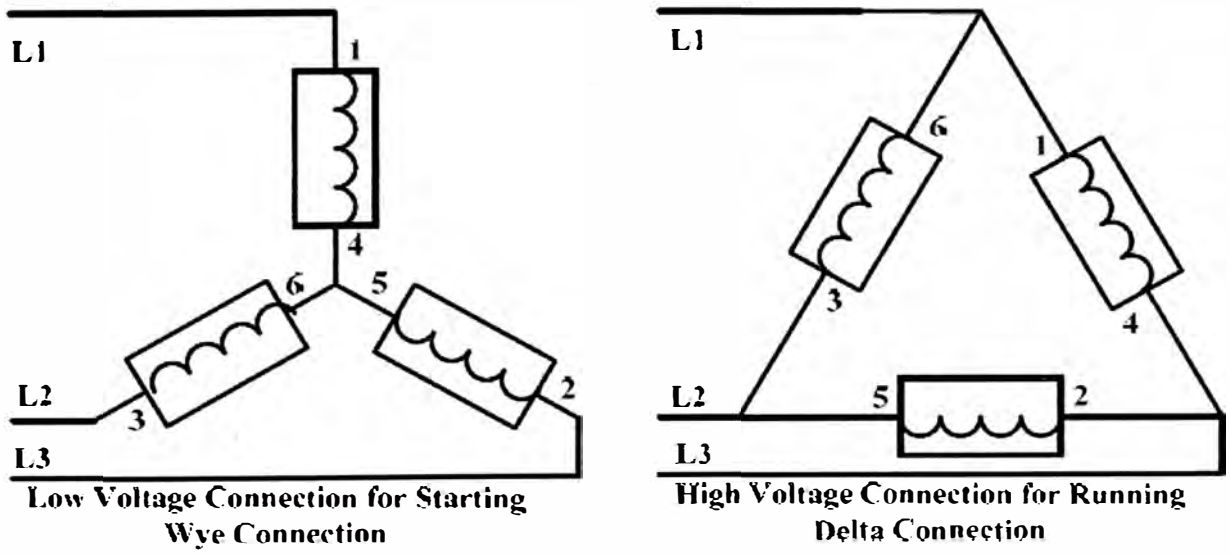


**Figura 2.15** Mostrando esquema a partir de una técnica de arranque estrella-delta [4]

La tensión en cada bobina para la conexión en estrella es uno dividido por la raíz cuadrada de 3 o 58% de la tensión en la bobina o devanado delta conectado. Podemos obtener esto directamente de pretender que la estrella está incluida en un delta, así como lo muestra en la figura 2.16 a continuación.

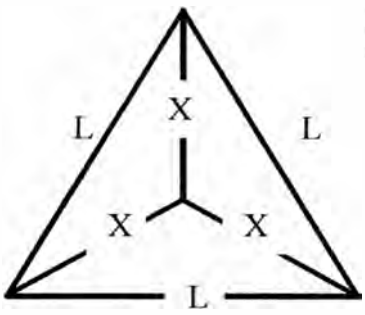


**Figura 2.16** La tensión en estrella se reduce hasta 58% de la tensión en delta en la técnican de arranque de motor [4]

Por ser un triángulo equilátero cada lado del borde del triángulo sería L, mediante la ecuación (2.1) realizada, el resultado sería 0.577 que equivale a un porcentaje de 58% del lado del triángulo .

$$X = \sqrt{3}/3\ L \quad \text{.....} \quad \textbf{(2.1)}$$

No sólo tiene que ser un motor especial, pero tiene que ser un arranque especial para el arrancador estrella Delta.

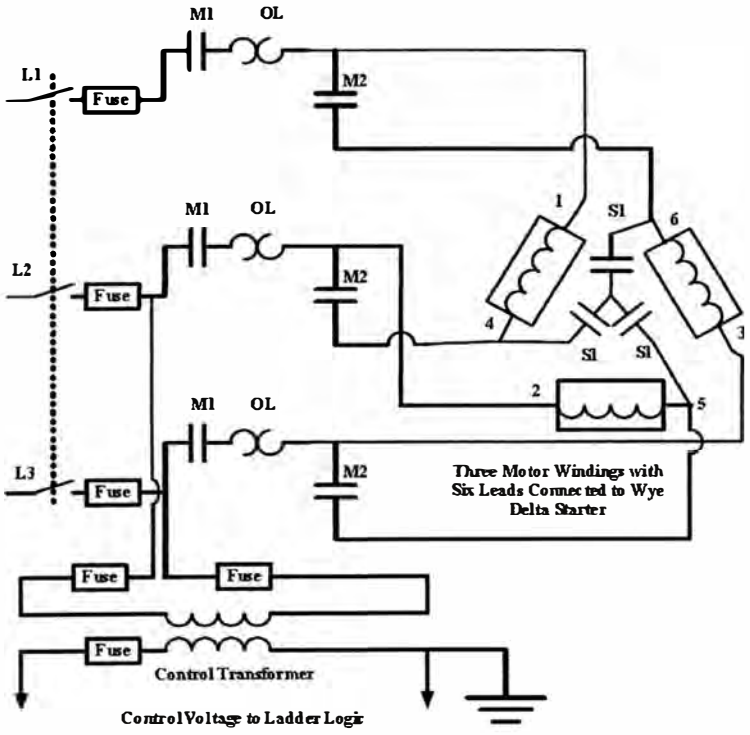


**Figura 2.17** Esquema de potencia para el arrancador estrella Delta [4]

Si el motor no se está utilizando como la aplicación de inicio de voltaje reducido, entonces se conectará los tres conductores para ser llevados a cabo a conexión. Tener en cuenta que hay 3contactores de potencia, M1, M2, y S1. Cada uno de ellos tiene 3 contactos de potencia nominal. El diagrama lógico escalera del control de un típico arrancador estrella delta se muestra en la figura 2.18 [4]

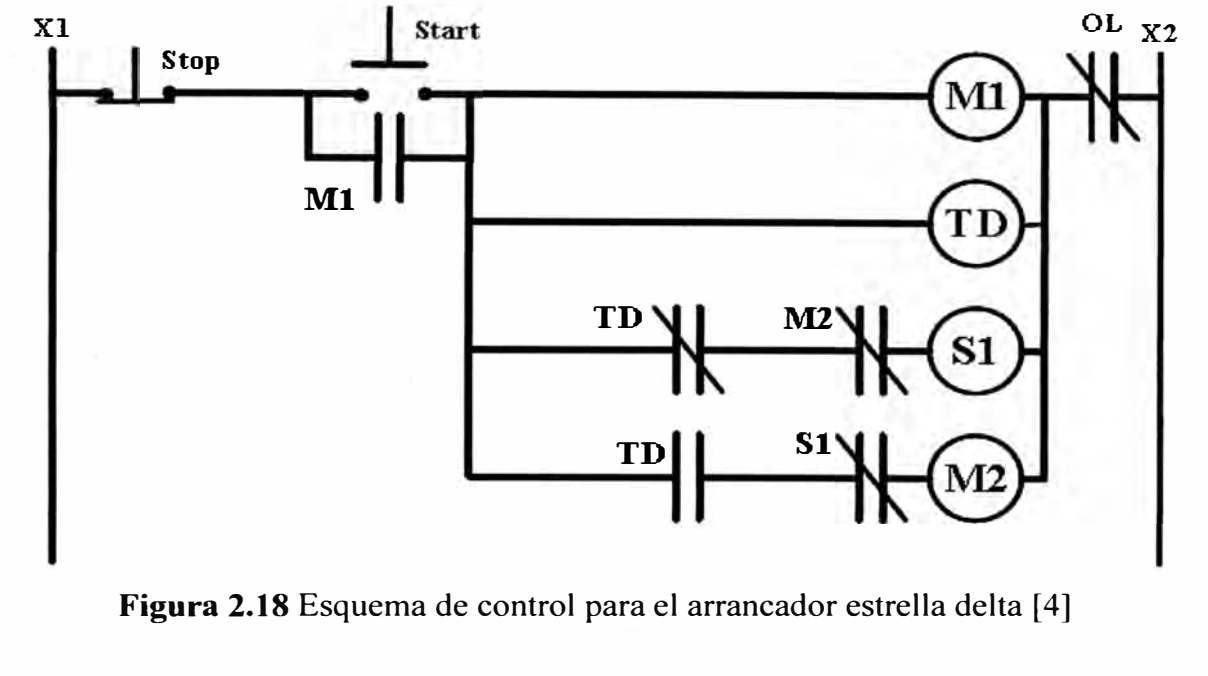


**Figura 2.18** Esquema de control para el arrancador estrella delta [4]

La secuencia de operaciones es el siguiente:

1. El pulsador de star se pulsa (se inserta).
2. Los contactores M1 y S1 cerrados; el relay de la bobina TD se energiza y comienza a sincronizar.
3. Esto hace que el motor conecte a la configuración estrella y la corriente fluye a través de las bobinas del motor. La figura 2.15 muestra el esquema de ésta configuración de conexión, y está configuración de conexión puede también tracear al esquema de potencia para el arrancador estrella delta (ver figura 2.17). Este motor está arrancando un torque cerca de 33% del par a plena carga y con una corriente aproximada de 200 % de la corriente total en carga.
4. Después del tiempo de retardo determinado por TD, el contactor S1 se abre y el contactor M2 se cierra. Si el tiempo de retardo se ajusta apropiadamente, entonces el motor alcanza su velocidad, y la corriente se mantiene un nivel de valor razonable. El tiempo de retardo TD es un tiempo que retarda la operación [4].

Está técnica se utiliza sobre todo a partir de motores de 50 HP y mucho más. Debido a que utiliza tres contactores de potencia con cada tres contactos, es mucho más

caro que uno a través de la línea de arranque. La principal ventaja de esta técnica de arranque estrella delta es que limita la corriente de arranque a 200% de la corriente de funcionamiento, en lugar de una corriente de arranque de 600% que es lo habitual cuando no se usará está técnica. El torque limitado (a 33%) también ayuda a prevenir daños a los componentes mecánicos [4].

### 2.2.1.4. Arrancador con un autotransformador a voltaje reducido

Otra forma de obtener un voltaje reducido, por lo actualmente analizado en el arranque del motor es con un autotransformador que reduce la tensión de arranque. Un autotransformador típico de arranque de voltaje reducido utiliza dos autotransformadores y dos contactores de potencia. Uno de los contactores de potencia tiene tres contactos y el otro tiene cinco contactos. Los autotransformadores tienen típicamente tres golpes. Son el 50%, 65% y el 80% de la tensión total [4].

Un diagrama esquemático de un autotransformador de arranque de voltaje reducido se muestra en la figura 2.19. Una ventaja de esta técnica es que con esto es posible seleccionar la cantidad de reducción de la corriente y del torque que deseas. Tenga en cuenta que el dibujo muestra 5 M1 contactos de potencia y 3 contactos de potencia M2. Como la mayoría de circuito de control, hay varias técnicas que pueden trabajar, para hacer este trabajo de arranque en forma correcta.

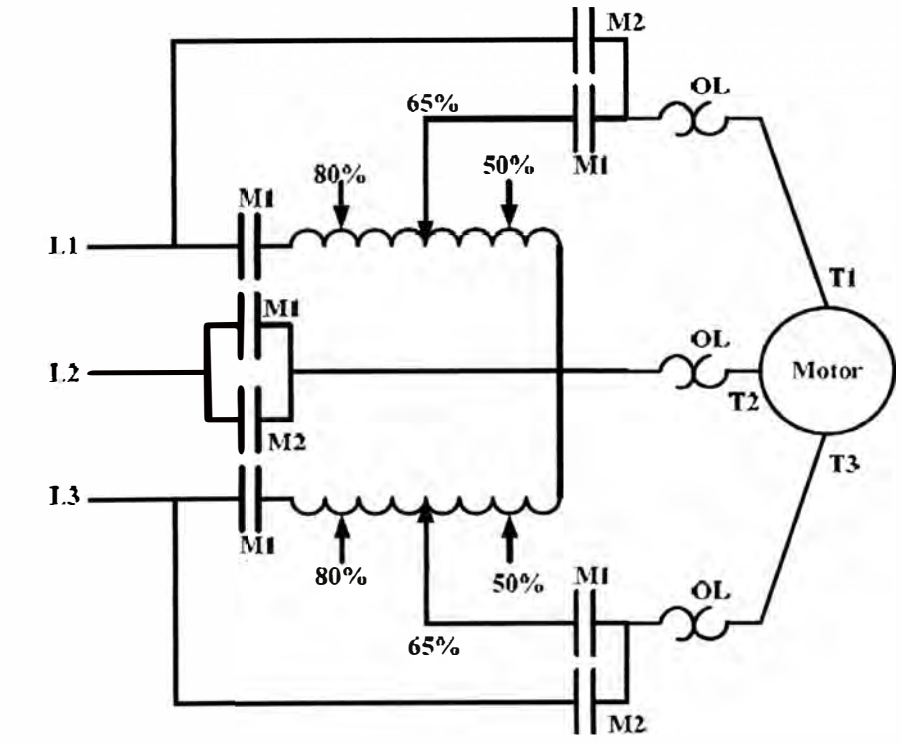


**Figura 2.19** Esquema de potencia para el arrancador de un autotransformador a voltaje reducido [4]

Este arranque requiere del contactor M1 que cierre los 5 contactos (ver figura 2.19), luego de un breve período de tiempo, el contactor M1 de potencia se abrirá y el contacto M2 de potencia se cerrará, y quedaría cerrado porque el motor está operando o está en marcha [4].

En este esquema de control (ver figura 2.20), cuando el botón de arranque (STAR) se aprieta, la bobina del relay CR se energiza, entonces los contactos CR se cierran; luego el relevador de tiempo (TD) se activa dentro de un determinado tiempo, así pues los contactos de potencia M2, estarán en estado normalmente cerrado y los contactos de potencia M1 estarán en normalmente abierto. Durante el tiempo de retardo antes de activar la bobina TD, los contactos de potencia M1, están en estado normalmente cerrado; esto, hace que el motor se gradua la tensión por partes hasta obtener la tensión plena de alimentación de la red eléctrica, que queda conectada directamente con el estator del motor. El retardo de tiempo (TD) es un tiempo de retardo a la conexión, que significa que los contactos TD cambiarán de estado debido a su energización despues de un período de tiempo en que se suministre la alimentación de la bobina [4].

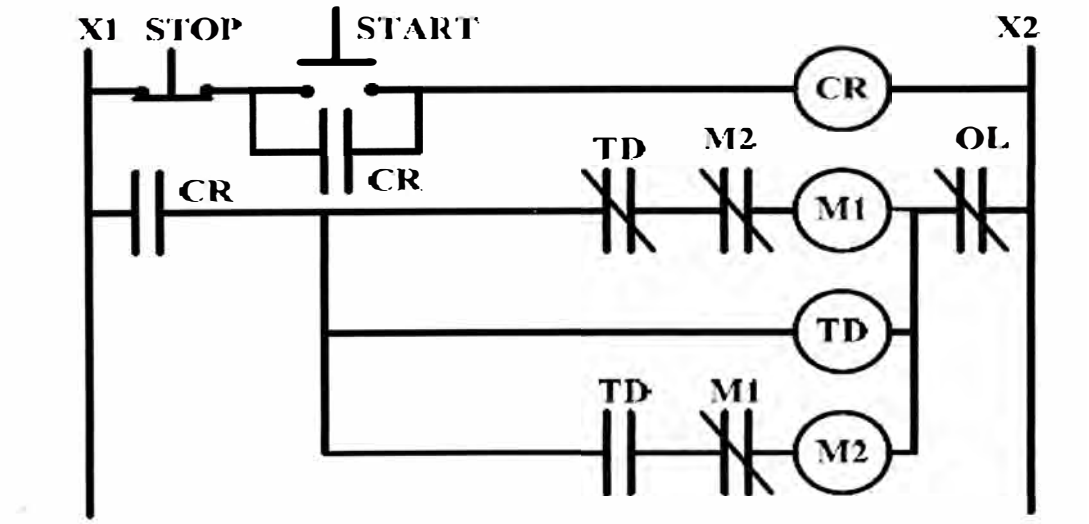


**Figura 2.20** Esquema de control para el arrancador de un autotransformador a voltaje reducido [4]

### 2.2.2. Modelo de referencia OSI

El modelo de referencia OSI (Interconexión de Sistemas Abiertos) es un modelo de siete capas desarrollado por la Organización Internacional de Estándares (ISO). Muchas arquitecturas basadas en capas partieron del modelo de referencia OSI y a partir de éste se generaron muchas otras arquitecturas como TCP/ IP. El modelo de referencia OSI es un estándar global de la industria, utilizado en las operaciones de red, con el fin de definir

cómo se comunican los protocolos a través de ésta; se desarrolló para ayudar a crear aplicaciones que sean compatibles en productos de distintas marcas. OSI utiliza un modelo de pila de protocolos donde cada nivel soluciona una serie de problemas relacionados con la transmisión de datos y proporciona un servicio bien definido a los niveles más altos. Los niveles superiores son los más cercanos al usuario y tratan con datos más abstractos, dejando a los niveles más bajos la labor de traducir los datos de forma que sean físicamente manipulables.

### 2.2.3. Protocolo TCP/IP

No existe un modelo oficial de protocolos TCP/IP; al contrario que en OSI los protocolos se han ido definiendo anárquicamente. En el modelo TCP/IP no es estrictamente necesario el uso de todas las capas, por ejemplo, hay protocolos de aplicación que operan directamente sobre IP y otros que lo hacen por encima de IP.

Cuando se habla de redes TCP/IP, siempre estará presente el término datagrama. En la figura 2.17 se pueden apreciar los 4 niveles de la arquitectura, comparados con los siete de la torre OSI.

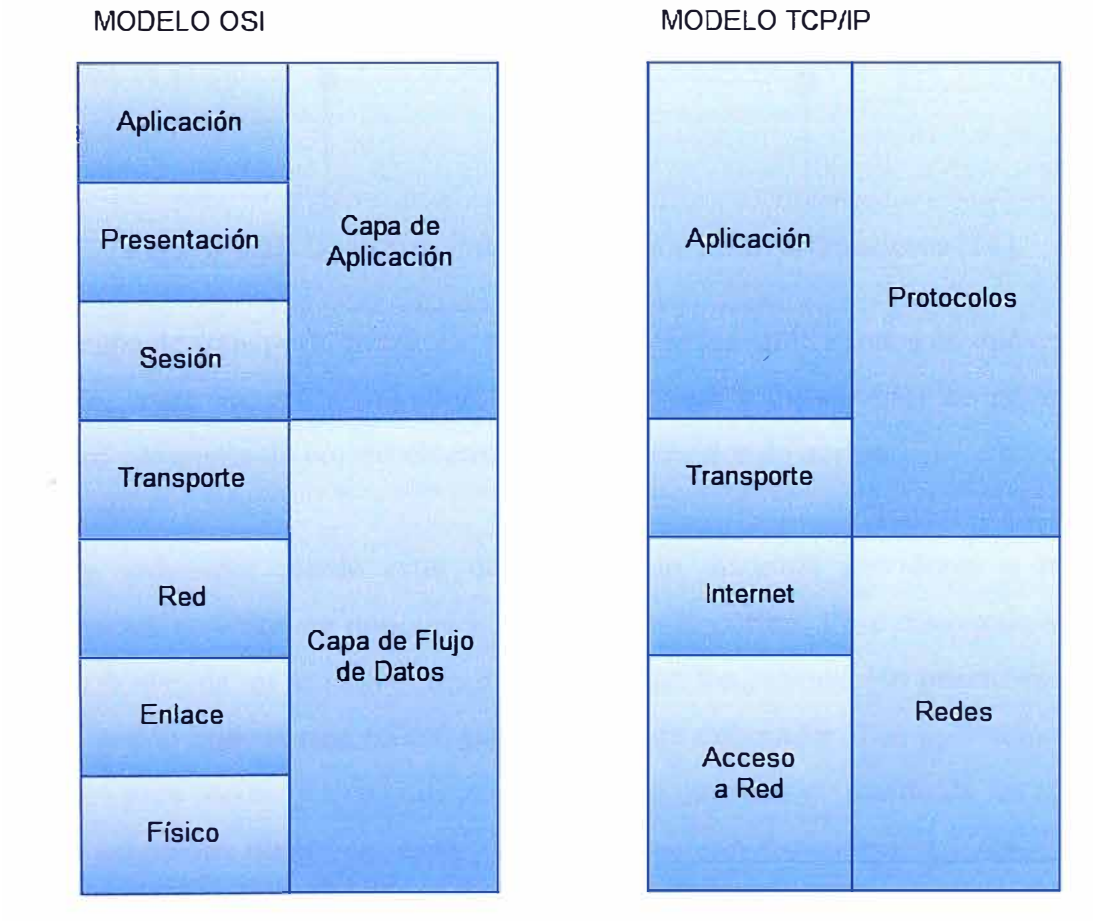


**Figura 2.17** Comparación del Modelo OSI y TCP/IP [12]

La función de la pila de protocolos TCP/IP es transferir información de un dispositivo de red a otro. Al hacerlo así, asigna estrechamente el modelo de referencia OSI en las capas más bajas y soporta todos los protocolos estándar de enlace de datos y físico.

### 2.2.3.1 Capa de transporte

La capa de red transfiere datagramas entre dos ordenadores a través de la red utilizando como identificadores las direcciones IP. La capa de transporte añade la noción de puerto para distinguir entre los muchos destinos dentro de un mismo host. No es suficiente con indicar la dirección IP del destino, además hay que especificar la aplicación que recogerá el mensaje. Cada aplicación que esté esperando un mensaje utiliza un número de puerto distinto; más concretamente, la aplicación está a la espera de un mensaje en un puerto determinado (escuchando un puerto).

Pero no sólo se utilizan los puertos para la recepción de mensajes, también para el envío: todos los mensajes que envíe un ordenador debe hacerlo a través de uno de sus puertos. El siguiente diagrama representa una transmisión entre el ordenador 194.35.133.5 y el 135.22.8.165. El primero utiliza su puerto 1256 y el segundo, el 80.

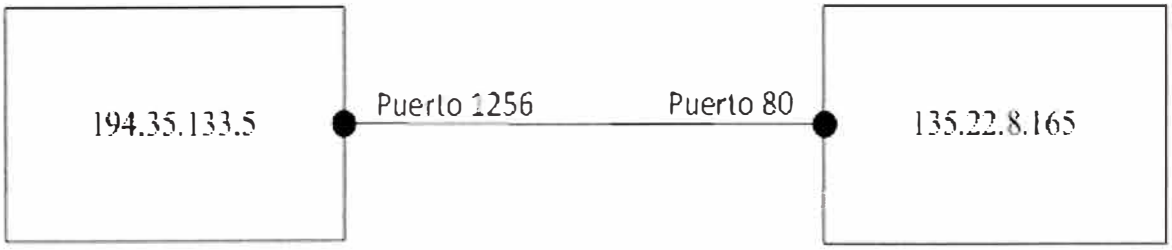


**Figura 2.18** Diagrama entre transmisión entre ordenadores [11]

La capa de transporte transmite mensajes entre las aplicaciones de dos ordenadores. Por ejemplo, entre nuestro navegador de páginas web y un servidor de páginas web, o entre nuestro programa de correo electrónico y un servidor de correo.

### 2.2.3.2 Puertos

Un ordenador puede estar conectado con distintos servidores a la vez; por ejemplo, con un servidor de noticias y un servidor de correo. Para distinguir las distintas conexiones dentro de un mismo ordenador se utilizan los puertos. Un puerto es un número de 16 bits, por lo que existen 65536 puertos en cada ordenador. Las aplicaciones utilizan estos puertos para recibir y transmitir mensajes. Los números de puerto de las aplicaciones cliente son asignados dinámicamente y generalmente son superiores al 1024. Cuando una aplicación cliente quiere comunicarse con un servidor, busca un número de puerto libre y lo utiliza. A continuación se enumeran los puertos bien conocidos más usuales [11]:

**TABLA N° 2.2** Descripción de los puertos TCP/UDP [11]

| Palabra Clave | Puerto | Descripción |
|---|---|---|
| Tcpmux | 1/tcp | TCP Port Service Multiplexer |
| Rje | 5/tcp | Remote Job Entry |
| Echo | 7/tcp/udp | Echo |
| Discard | 9/tcp/udp | Discard |
| Systat | 11/tcp/udp | Active Users |
| Daytime | 13/tcp/udp | Daytime |
| Qotd | 17/tcp/udp | Quote of the Day |
| Chargen | 19/tcp/udp | Character Generator |
| ftp-data | 20/tcp | File Transfer[Default] |
| ftp | 21/tcp | File Transfer[Control] |
| telnet | 23/tcp | Telnet |
| Smtp | 25/tcp | Simple Mail Transfer |
| Time | 37/tcp/udp | Time |
| nameserver | 42/tcp/udp | Host Name Server |
| Nicname | 43/tcp/udp | Who Is |
| Domain | 53/tcp/udp | Domain Name Server |
| Bootps | 67/udp/udp | Bootstrap Protocol Server |
| Tftp | 69/udp | Trivial File Transfer |
| Finger | 79/tcp | Finger |
| www-htp | 80/tcp | World Wide Web HTTP |
| Dcp | 93/tcp | Device Control Protocol |
| Hostname | 101/tcp | NIC Host Name Server |
| Gppitnp | 103/tcp | Genesis Point-to-Point Trans Net |
| Rtelnet | 107/tcp/udp | Remote Telnet Service |
| Pop3 | 110/tcp | Post Office Protocol – Version 3 |

## 2.3 Definición de términos

**Arranque**: El motor puede arrancar conectándolo directamente a través de la línea. Sin embargo, la máquina impulsada se puede dañar si se arranca con ese esfuerzo giratorio repentino. El arranque debe hacerse lenta y gradualmente, no sólo para proteger la máquina, sino porque la oleada de corriente de la línea durante el arranque puede ser demasiado grande. La frecuencia del arranque de los motores también comprende el empleo del controlador [6].

**Paro:** Los controladores permiten el funcionamiento hasta la detención de los motores y también imprimen una acción de freno cuando se debe detener la máquina rápidamente. La parada rápida es una función para casos de emergencia [6].

**Inversión de la rotación:** Se necesitan controladores para cambiar automáticamente la dirección de la rotación de las máquinas mediante el mando de un operador en una estación de control. La acción de inversión de los controladores es un proceso continuo en muchas aplicaciones industriales. Esta puede hacerse por medio de estaciones de botones, un interruptor de tambor o un módulo inversor de giro [6].

**Marcha**: Las velocidades y características de operación deseadas, son, función y propósito directos de los controladores. Éstos protegen a los motores, operadores, máquinas y materiales, mientras funcionan [6].

**Control de velocidad:** Algunos controladores pueden mantener velocidades muy precisas para propósitos de procesos industriales, pero se necesitan de otro tipo para cambiar las velocidades de los motores por pasos o gradualmente [6].

**Seguridad del operador:** Muchas salvaguardas mecánicas han dado origen a métodos eléctricos. Los dispositivos piloto de control eléctrico afectan directamente a los controladores al proteger a los operadores de la máquina contra condiciones inseguras [6].

**Protección contra daños:** Una parte de la función de una máquina automática es la de protegerse a sí misma contra daños, así corno a los materiales manufacturados o elaborados. Por ejemplo, se impiden los atascamientos de los transportadores. Las máquinas se pueden hacer funcionar en reversa, detenerse, trabajar a velocidad lenta o lo que sea necesario para realizar la labor de protección [6].

**Mantenimiento de los dispositivos de arranque:** Una vez instalados y ajustados adecuadamente, los arrancadores para motor mantendrán el tiempo de arranque, voltajes, corriente y troqué confiables, en beneficio de la máquina impulsada y el sistema de energía. Los fusibles, cortacircuitos e interruptores de desconexión de tamaño apropiado

para el arranque, constituyen buenas prácticas de instalación que se rigen por los códigos eléctricos [6].

**Contactor principales:** Son los contactos que tienen por finalidad realizar el cierre o apertura del circuito principal, a través del cual se transporta la corriente al circuito de utilización (carga). Deben estar debidamente calibrados para permitir el paso de intensidades requeridas por las carga sin peligro de deteriorarse. Por la función que debe realizar estos contactos serán únicamente abiertos [10].

**Contactor auxiliares:** Son aquellos contactos que tienen por finalidad el gobierno del contactos (específicamente de la bobina) y de sus señalización. El número de contactos auxiliares por contactos varía de acuerdo a las necesidades de las diferentes maniobras, desde uno normalmente abierto, hasta varios abiertos y cerrados. Pueden ser abiertos o cerrados y, como están hechos para dar paso únicamente a pequeñas corrientes (alimentación de la bobina y elementos de señalización), suelen ser normalmente más pequeños que los contactos principales [10].

**Protocolo UDP:** El protocolo UDP (User Datagram Protocol, protocolo de datagrama de usuario) proporciona una comunicación muy sencilla entre aplicaciones de dos ordenadores. Al igual que el protocolo IP, UDP es no orientado a conexión y no fiable. UDP utiliza el protocolo IP para transportar sus mensajes. Como vemos, no añade ninguna mejora en la calidad de la transferencia; aunque sí incorpora los puertos origen y destino en su formato de mensaje [11].

**Protocolo TCP:** El protocolo TCP (Transmission Control Protocol, protocolo de control de transmisión) está basado en IP que es no fiable y no orientado a conexión, sin embargo, es orientado a conexión y es fiable. El protocolo TCP permite una comunicación fiable entre dos aplicaciones [11].

**Orientado a conexión:** Es necesario establecer una conexión previa entre las dos máquinas antes de poder transmitir ningún dato. A través de esta conexión los datos llegarán siempre a la aplicación destino de forma ordenada y sin duplicados. Finalmente, es necesario cerrar la conexión [11].

**Fiable:** La información que envía el emisor llega de forma correcta al destino [11].

**No orientado a conexión:** No se establece una conexión previa con el otro extremo para transmitir un mensaje UDP. Los mensajes se envían sin más y éstos pueden duplicarse o llegar desordenados al destino [11].

**No fiable:** Los mensajes UDP se pueden perder o llegar dañados [11].

# CAPÍTULO III
# METODOLOGÍA PARA LA SOLUCIÓN DEL PROBLEMA

## 3.1 Alternativas de solución

### 3.1.1. Primera alternativa de solución

De las figuras 2.11 y 2.12 se obtiene en la figura 2.13 el circuito de control completo para el arranque y apagado del motor. Si queremos controlar en forma remota de acuerdo de las figuras antes mencionadas, se tendrá que hacer una modificación en el esquema del circuito de control básico, se adicionará, de la siguiente manera:

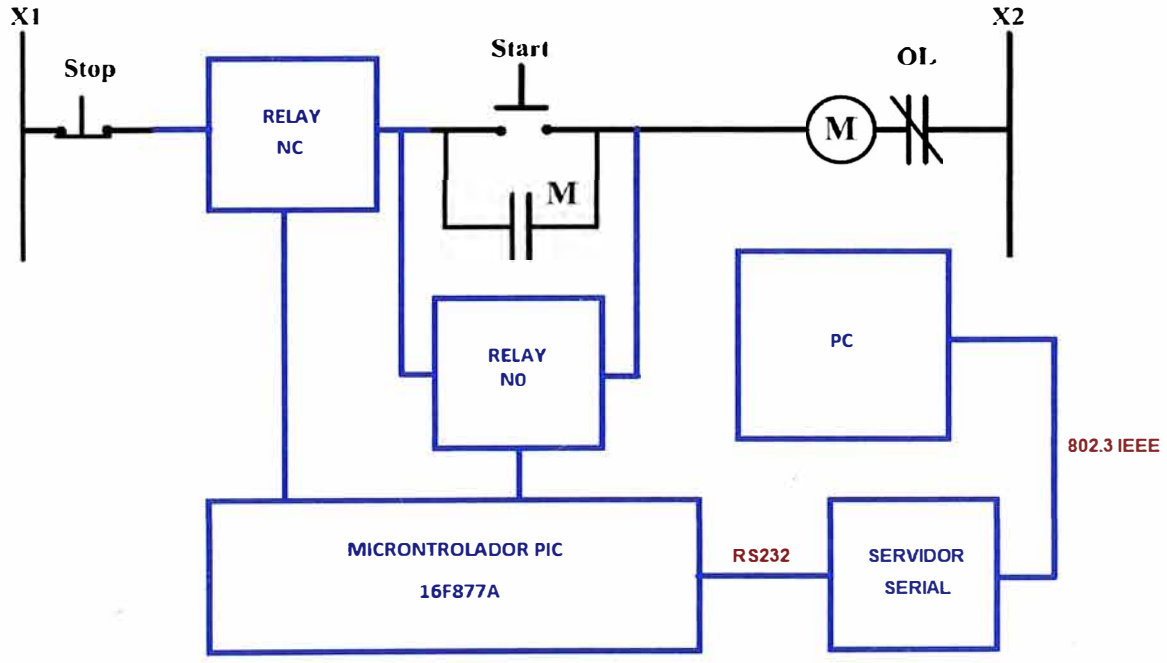


**Figura 3.1** Propuesta para el control a distancia del arranque y apagado de un motor

Como nos describe en la figura 3.1, el relay NC se instalará en serie y el relay NO se instalará en forma paralela; éstos estarán controlado por el microcontrolador pic 16f877A, que son obedecidos por los comandos ejecutados desde la pc, en la cual se lograría controlar el arrancado y apagado en forma remota. El software tiene que ser un diseño capacitado para controlar el accionamiento de los contactores que activen los diferentes circuitos de control de arranques, revisar en el fundamento teórico antes

mencionado. Luego de recibir la instrucción por medio del ordenador, se visualizará la respuesta rápida del PIC. Ello hace que el control del sistema sea inmediato y eficaz, y además el controlador mediante un software de programación que puede ser visual Basic, C++, JAVA, cualquier el lenguaje de visual .net 2008 que puede C#, J#, entre otros, y además en labview.

### 3.1.2. Segunda alternativa de solución

Todos los elementos electromecánicos y electrónicos que permiten el funcionamiento conjunto del módulo están gobernados por un microcontrolador PIC 16F877A, que será el encargado de administrar el funcionamiento de los actuadores obedeciendo los comandos de las tareas que son recibidas por el usuario, ya sea a través del computador o mediante los pulsadores del mismo módulo, y la programación en el ordenador para el interfaz de usuario puede ser en visual basic [7], y también se puede hacer con el labview.

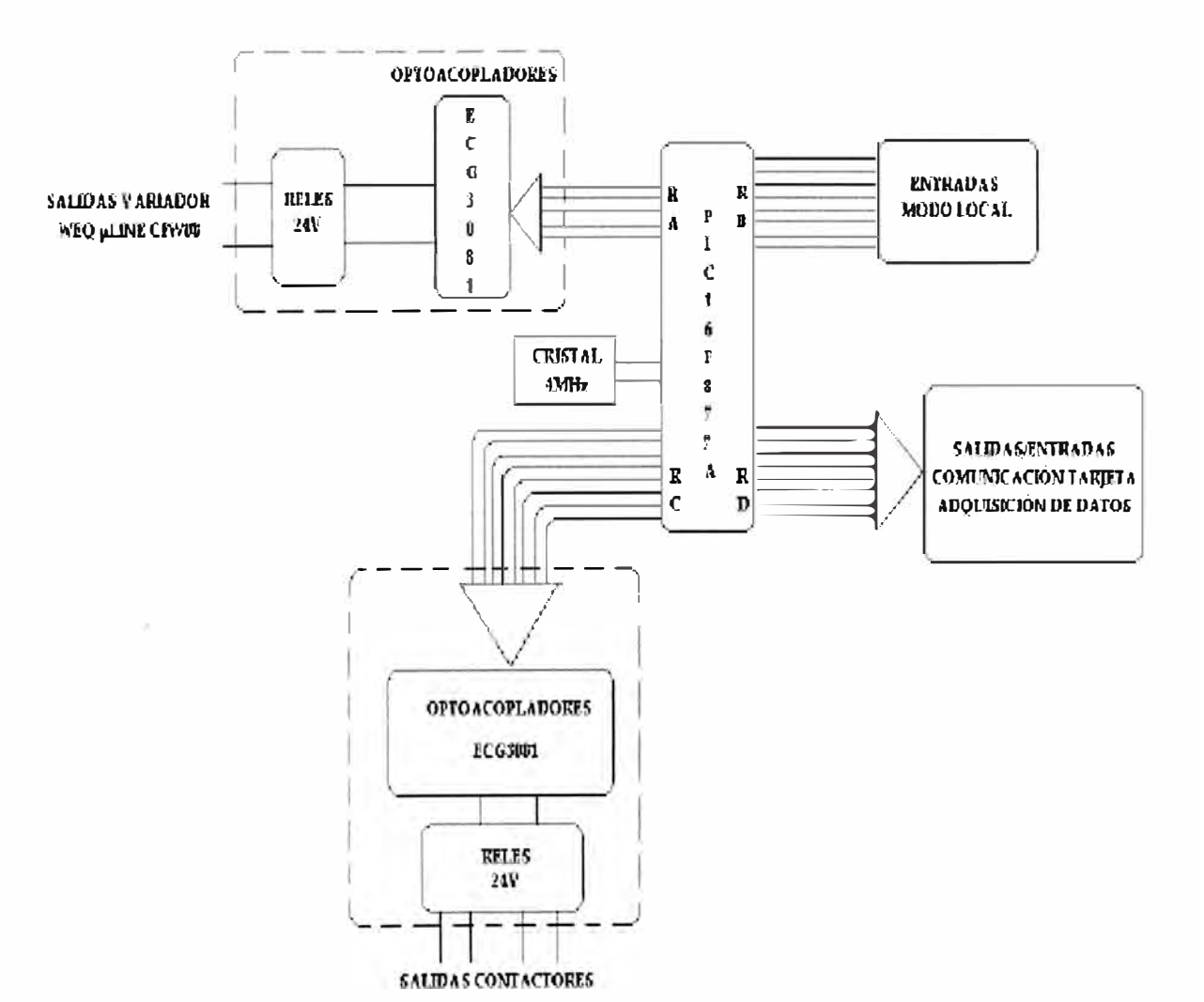


**Figura 3.2** Esquema de hardware de la tarjeta principal [7]

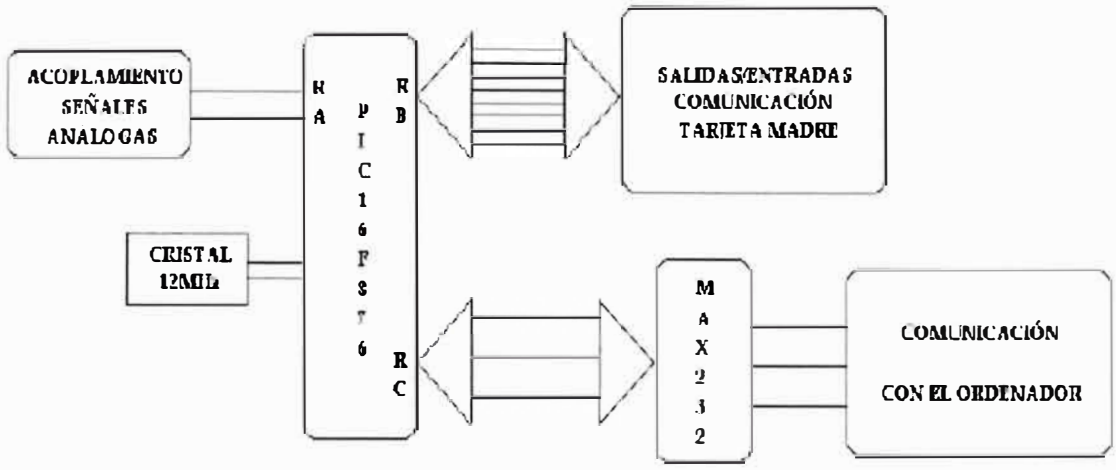


**Figura 3.3** Esquema de hardware de la tarjeta de adquisición de datos [7]

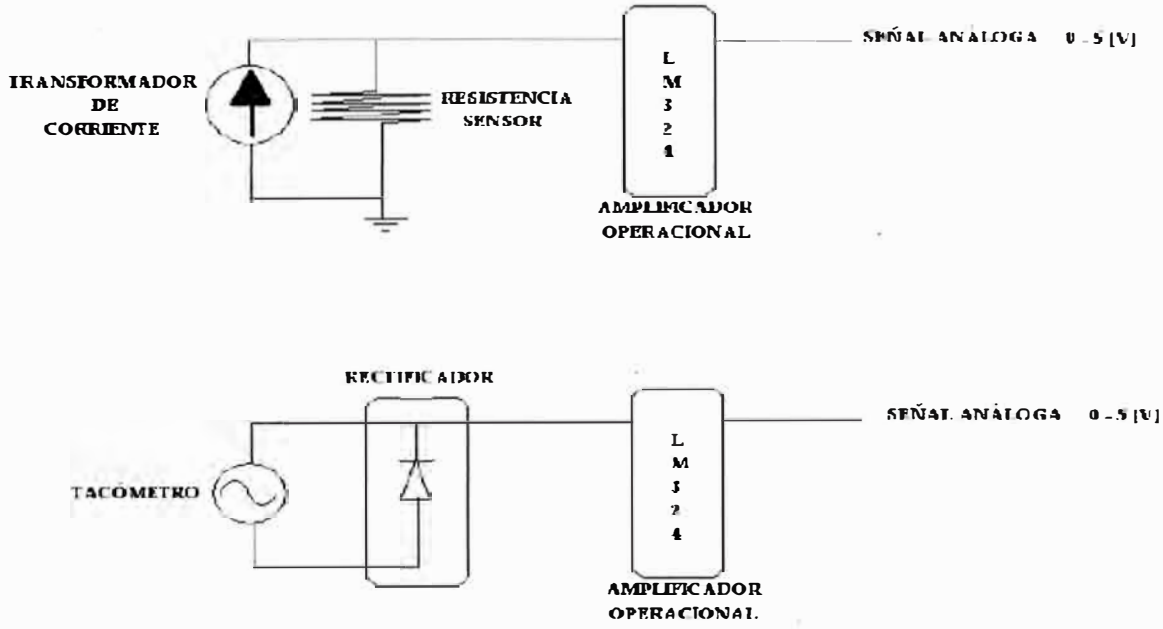


**Figura 3.4** Esquema de hardware del acoplamiento de señales análogas [7]

## 3.2. Solución del problema

Se muestra a continuación la propuesta de solución del sistema de control el arrancado y apagado del motor en forma remota mediante TCP/IP a un menor costo, se muestra en la figura1.2, que es un control de lazo cerrado, debido a las limitaciones (revisar el capítulo 1.4) para la construcción del proyecto, entonces no se consideró el bloque de sensor de temperatura obteniendo así un control de lazo abierto. Para las pruebas en campo se construyó el prototipo como indica la siguiente figura 3.5, a continuación se va explicar el control de arranque y apagado del motor eléctrico en forma remota.

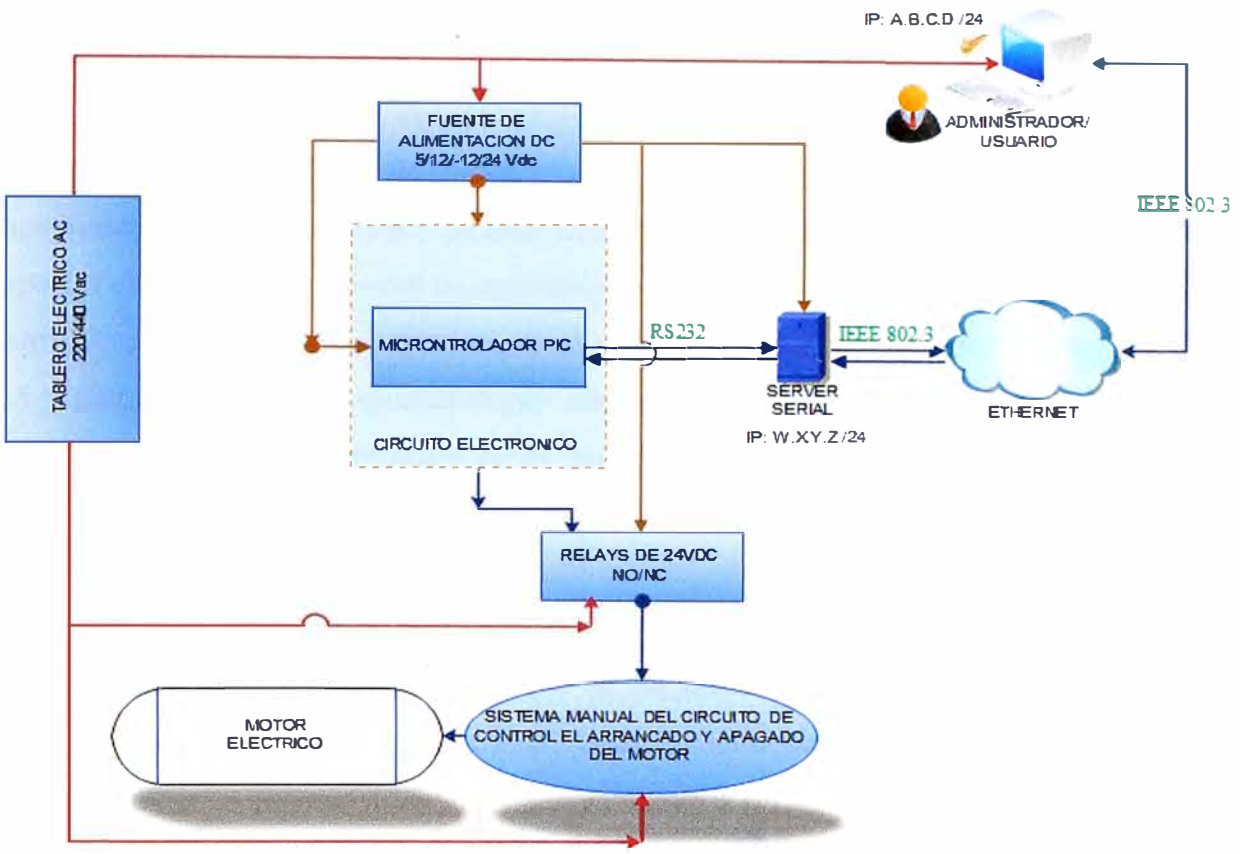


**Figura 3.5** Esquema del sistema para el control de lazo abierto on/off para el motor eléctrico ac

De acuerdo con la figura mostrada, el proyecto está conformada por el diseño de alimentación DC de 3 salidas que son 5V, 12V y -12V; diseño de un circuito de control basado con el microcontrolador PIC, comunicación serial, servidor serial TIBBO DS 100 , comunicación ethernet (IEEE 802.3), el interfaz de usuario que se programó en C# de visual .net 2008, entendimiento de funcionamiento del sistema manual del circuito de control el arrancado y apagado del motor y para efectos de pruebas en campo , se previó hacer pruebas locales en la cual se construyó un hardware que conforma relays en función del circuito de control básico de arranque y apagado del motor a que se le llama el control de relevadores para un ventilador de 24 Vdc. Se describe en forma resumida el funcionamiento y configuración de este esquema que se muestra la figura 3.5, entonces comenzamos en configurar las ips tomando en cuenta se tiene que formar parte del sistema de red tcp/ip LAN, y si es necesario diseñarlos, ya que por estos equipos transportarán la trama ethernet, y parte de esa trama se encuentra guardada el dato que contiene las instrucciones encomendadas desde el interfaz de usuario. En el recorrido de los equipos mostrados en la figura 3.5 se encuentra con el servidor serial, ya que éste tiene la función

de convertir los protocolos ethernet a protocolos seriales, es decir TCP/IP (IEEE 802.3) a RS232, a su vez estas señales de protocolos RS232 lo recibe el integrado electrónico MAX232, ya que este se encarga de convertir a señales ttl (tramas convertidas en ttl) que lo recibe el microcontralador PIC. Para poder conseguir la manipulación del arrancado y apagado del motor desde el PIC, se hace una programación en alto nivel C en el PIC, cuyo propósito es de activar mediante energización de las bobinas de relay o relé para este fin de control (ver figura 3.7). Los relays que se activan a causa del PIC, las cuales son: RL4, RL5 y RL6. Terminada la ejecución por unos segundos la instrucción, entonces valida la transmisión del PIC y se muestra en la pantalla de la pc por el interfaz de usuario, una cadena de letras, en cual nos muestra que ya reviso el comando, por lo tanto, ha sido ejecutada dicha tarea.

### 3.2.1. Diseño de un circuito de la fuente de alimentación dc de 3 salidas

En este sistema experimental que se ha diseñado en la fuente de energía de 3 salidas, que son $+5V, -12V, +12V$, esta tensión puede tener un error de $\pm 10\%$, por lo tanto, por lógica se utilizaron los reguladores para 5V (el regulador LM7805), para 12V se utilizó el LM7812; para -12V el LM7912. En la siguiente figura 3.6 se muestra la simulación del diseño de la fuente, cuál nos muestra 2 fuentes de 24Vdc en forma independiente.

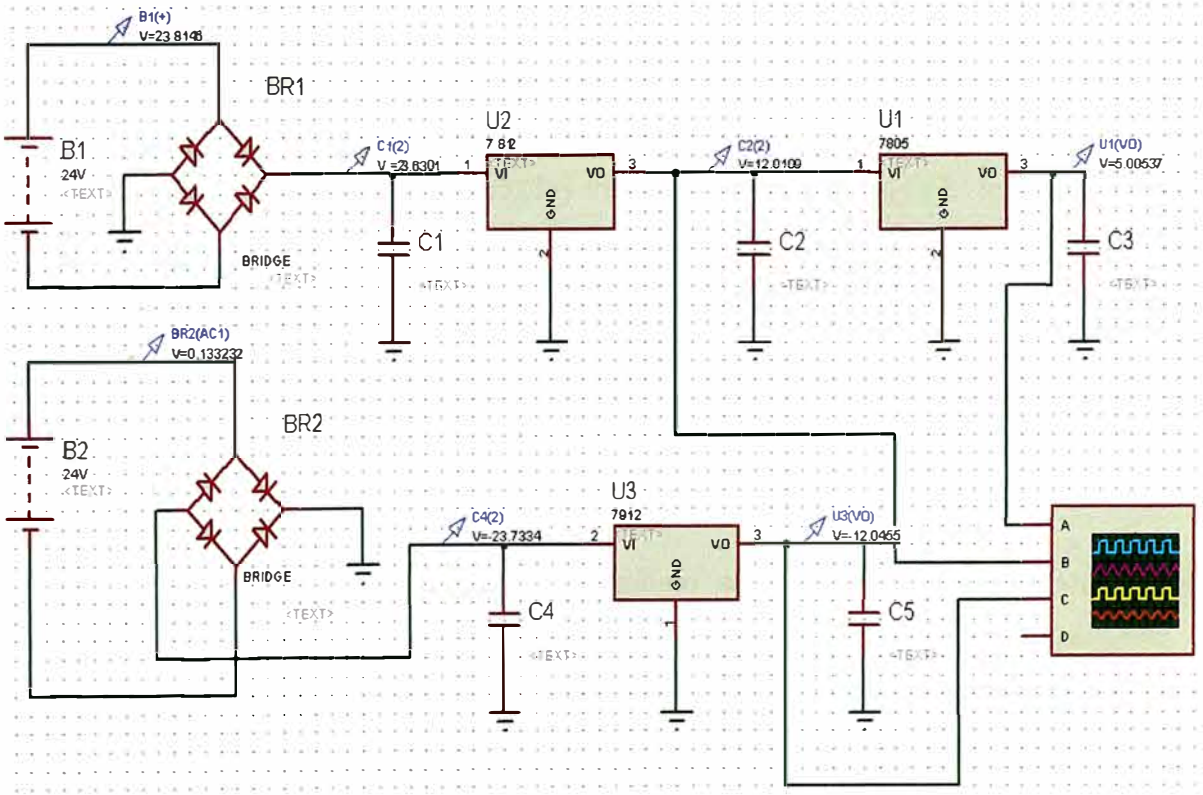


**Figura 3.6** Esquema de la fuente de alimentación de 3 salidas

Cada fuente está conectada por un puente diodo, en ese presente simulación no están todos los elementos implementado, pero aquí se muestra cómo se va obteniendo estos 3 voltajes, para luego hacer la tarjeta impresa se muestra el esquema diseñado en la figura B.3. Para modo de implementación se conecta el transformador de 220 a 24 vdc a una de la bornera de 24 Vdc (ver figura B.3), y están conectados con diodo puente cada uno, y para cada lado, para protección del circuito se colocó un termoresistencia de 500 mA , luego un conjunto de capacitadores electrolíticos para que pueda filtrar en la salida del puente y/o entrada del reguladores y/o también su salida, debido a los efectos de ruidos, señales armónicos picos para un cierta frecuencia, etc.

En este esquema (ver figura 3.6), que ha sido simulado en proteus, el 7812 está en cascada con 7805 y el 7912 está separado. Con este esquema se tiene una fuente de alimentación de 3salidas, complementando así, una de las partes del proyecto; que es el esquema electrónico de 3 salidas de 5v, 12v, y -12v, revisando datasheet tiene un máximo de corriente de 1000 mA para cada regulador.

### 3.2.2. Diseño de un circuito de control con microcontrolador PIC

Para ver el comportamiento de este diseño del circuito de control con el microcontrolador PIC se hizo una simulación con el programa proteus (ver figura 3.7) que contienen: 1 microcontrolador PIC16F877A (U1), 4 optocopladores (U2; U3; U4; U5), 1 circuito oscilador externo (C2 ;C3; X1), 1 circuito de master clear (D4; R7; R8; C4; pulsador; VP5), 3 relés de 12Vdc voltios (RL1; RL2; RL3), 3 relays de 24Vdc voltios (RL4; RL5; RL6); y una consola para la ejecución de comandos.

Este circuito se simulará y se explayará a partir de la figura 3.7, teniendo en cuenta que la figura B.1 es un esquema electrónico final para construir la tarjeta electrónica impresa. Entonces, se describirá sus partes del diseño que conforma el circuito de control PIC; estas partes son las siguientes: uso del microcontrolador PIC, empleo del circuito oscilador, uso del circuito master clear, descripción del circuito de entrada y circuito de salida.

#### a. Uso del microcontrolador PIC

Para esta solución de propuesta de trabajo se escogió el microcontrolador PIC porque tiene un manejo sencillo, hay bastante información, fácil de conseguir, es económica frente a sus competidores, posee elevada velocidad de funcionamiento, tiene varios parámetros a manipular: velocidad; consumo; tamaño; alimentación; código compacto; entre otros, tiene herramientas fáciles y baratas para desarrollar, existe una gran

variedad de herramientas de hardware que permiten grabar, depurar, borrar y comprobar el comportamiento de los PIC, se diseña rápidamente, hay una gran variedad de modelos de PIC que permite elegir el que mejor responde al requerimiento de la aplicación. Entonces para este trabajo se necesita el mínimo requerimiento y requisitos para escoger el PIC adecuado, las cuales son: tenga comunicación serial, 4 entradas y salidas, además que haya un pin que permita ingresar señales análogo/digital, que permita generar ondas cuadradas de acuerdo al diseño del circuito oscilador, entonces el microcontrolador PIC a usar sería el 16F877A, ya que tiene las características suficientes para este fin, porque contiene el receptor transmisor serie síncrono-asíncrono universal (USART) y además el USART consta de 3 módulos fundamentales: el circuito de muestreo, el generador de frecuencia de transmisión (Baud Rate), el transmisor asíncrono y el receptor asíncrono. El uso del compilador CCS c se programa la lógica que desea para este fin. La programación en el microcontrolador PIC se explicará en el capítulo 3.2.3. Se describe en forma breve el PIC 16F877A., entonces, este microcontrolador posee 5 puertos de entrada/salida denominados PORTA, PORTB, PORTC, PORTD, PORTE; de las cuales 2 de esos puertos de entrada/salida se usan, que son: el PORTA y el PORTC. El puerto A posee 6 líneas bidireccionales, y tiene 3 registros asociados a este puerto; el registro PORTA (05H), el registro TRISA (85H), y el registro ADCON1 (9FH). El puerto C posee 8 líneas bidireccionales que están asociados a los registros PORTC (07H), TRISC (87H). Entre las características más importantes que se pueden resaltar de este microcontrolador PIC 16F877A se pueden mencionar los siguientes: USART integrado, manejo de hasta 14 interrupciones de diferentes fuentes, tres contactos/temporizadores, ADC de 10 bits, generador de PWM, 8kx14 double bytes de memoria flash disponible, 368x8 bytes de memoria RAM disponible, 256x8 bytes de memoria tipo EEPROM disponibles para datos, ciclo de máquina de hasta 20 Mhz. Sin embargo, para nuestro objetivo de controlar mediante TCP/IP se tiene que adaptar un equipo que convierta los protocolos seriales a protocolos TCP/IP, se usó el TIBBO DS100, se explicará con más detalle sus características en el capítulo 3.2.7.

**b. Empleo del circuito oscilador**

Se emplea un oscilador de 4 MHz debido a que la transferencia de datos por el puerto serial va a ser de 9600 baudios. La comunicación bidireccional se realiza por un único cable y es necesario para lograr esto una serie de rutinas precisas para obtener una temporización adecuada que le permita a ambos dispositivos comunicarse.

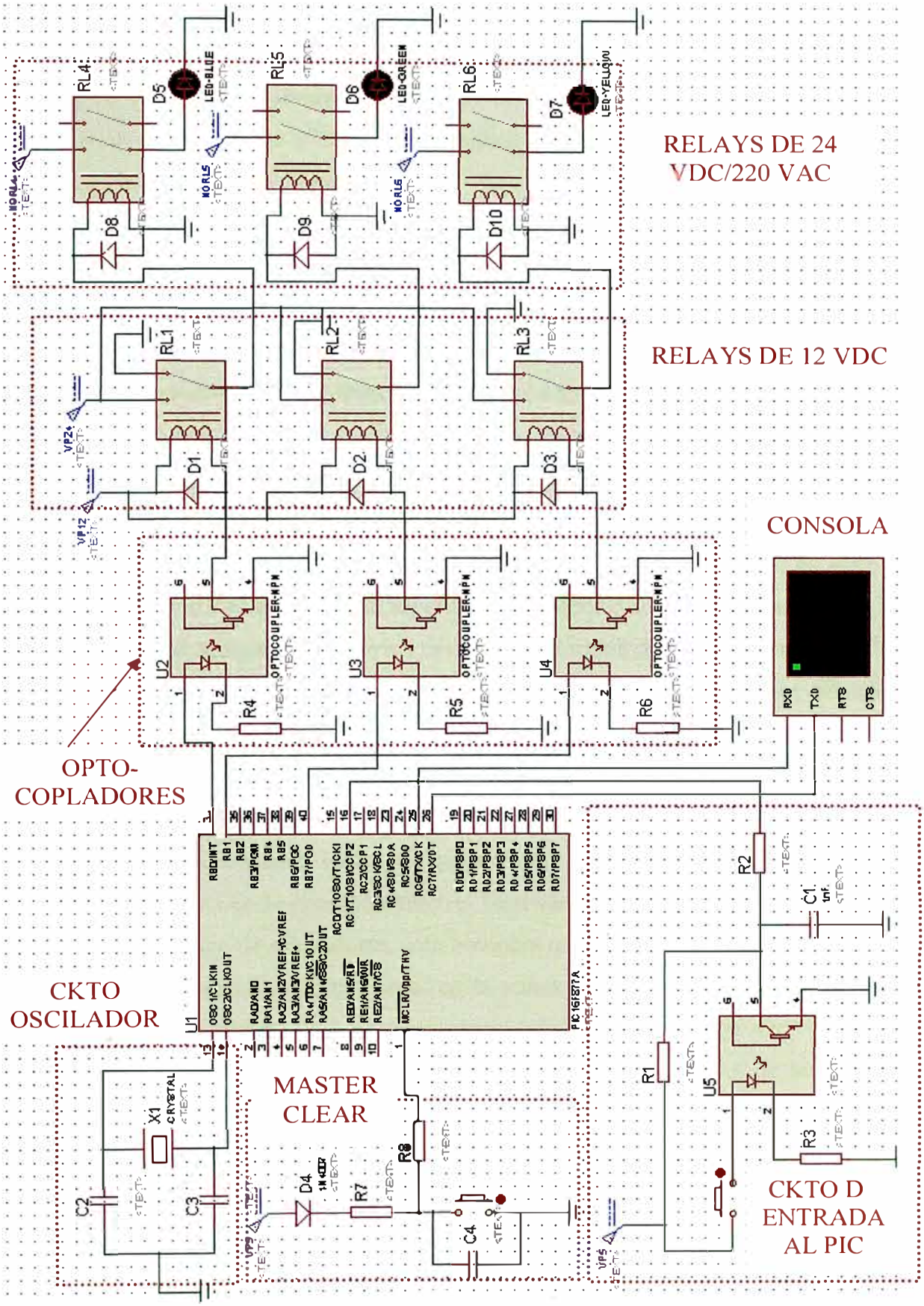


**Figura 3.7** Esquema de simulación del circuito de control basado con un microcontrolador PIC

**c. Uso del circuito master clear**

El circuito master clear consigue el reset llevando momentáneamente este pin a un estado lógico bajo. Esto hace que al encender el sistema el microcontrolador quede en estado de reset por un tiempo mientras se estabilizan todas las señales del circuito.

**d. Descripción del circuito de entrada**

En el circuito de entrada está conformado como muestra el esquema electrónico hecho en proteus (ver figura 3.7): un optocoplador U5; resistores R1, R2 y R3; capacitor C1; una fuente de 5 voltios VP5 y una botonera. El optocoplador cuya función es aislar eléctricamente el circuito, por tal razón es colocado en el circuito de control PIC, para proteger de las señales picos armónicos externas y de las tensiones relacionadas con la red eléctrica que pueden dañar al microcontrolador PIC. La botonera representa en la simulación del programa proteus como un interrupción de la tensión de 5 voltios, si analizamos la botonera en el circuito de entrada se presentan 2 casos que son: el primer caso es, si la botonera está abierta (interrumpe el paso de corriente), es decir en la entrada del microcontrolador PIC se aplica un nivel bajo (0 voltios), cuya respuesta a través de la comunicación serial transferida por bytes mediante una trama definida ( trama lo define el programador de PIC), cuya información binaria nos señala que el motor no está en funcionamiento; el segundo caso es, si la botonera está cerrada (permite circular corriente), es decir en la entrada del microcontrolador PIC se aplica un nivel alto (5 voltios), entonces la respuesta del microcontrolador PIC a través de la comunicación serial en bytes, transfiere la información binaria indicando que el motor está en funcionamiento.

Tener en cuenta en la conexión entre el hardware electrónico de control PIC y con el hardware electrónico de relevadores, está botonera que se ve en la figura 3.7, representa el relay Kp (ver figuraB.8). Sin embargo, en la conexión en el campo con el tablero de control del ventilador minero, está botonera representa un circuito electrónico que se tiene que diseñar y construir la tarjeta de adquisición de datos con 2 niveles de tensión 0 y 5 voltios, cuya señal externa es aplicada al circuito de entrada hacia al microcontrolador PIC sería la tensión eléctrica equivalente por temperatura (uso del sensor de temperatura).

**e. Descripción del circuito de salida**

El circuito de salida está conformado por optocopladores (U2, U3, U4); relés de 12 Vdc - 100 mA (RL1, RL2, RL3) y relays de 24 Vdc - 10 A/ 220Vac – 5A (RL4, RL5, RL6). En la figura 3.7 para poder explayar la descripción lo llamaremos "subcircuito" a las partes que abarca el circuito de salida, así: el "subcircuito1" lo conforma U2, RL1y RL4; el

"subcircuito2" lo conforma U3, RL2 y RL5; y el "subcircuito3" lo conforma U4, RL3 y RL6.Entonces, a continuación se describirá su funcionamiento del circuito de salida a partir de las 3 salidas del microntrolador PIC (RB0, RB1, RB7) conectada con cada subcircuito:

- En el subcircuito1: cuando en la salida de la patita 33 (RB0) del microcontrolador PIC 16F877A obtiene una tensión de nivel lógico alto (5 voltios), circula una corriente por el led emisor del optocoplador U2 emitiendo un haz de luz que incide sobre la base del transistor receptor del optocoplador U2 hace que lo sature; es decir, que el colector y emisor del transistor receptor se juntan, esto implica que pasa a conducción la tensión de 12 voltios dc VP12 (funcionamiento similar al interruptor) ya que este efecto permite la excitación de la bobina del relé RL1; entonces, se activa el relé RL1facilitandoa conducir la tensión de 24 voltios dc VP24, y por consiguiente se energiza la bobina del relay RL4, debido a esta activación del RL4 permita la conducción de la señal NORL4 a fin de que el led D5 se prenda; ya que nos señala la ejecución de la desactivación del motor ( se para el motor).

Si en la salida de la patita RB0 del microcontrolador obtiene un nivel lógico bajo (0 voltios), entonces no hay efectos de saturación del optocoplador U2, energización de la bobina RL1 y excitación de la bobina R4 por lo tanto el led D5se encuentra apagada; así pues nos indica el led D5 que no hay señal de control para la parada del motor.

- En el subcircuito 2: cuando en la salida de la patita 40 (RB7) del microcontrolado PIC obtiene una tensión de nivel lógico alto (5 voltios), manera a que se satura el optocoplador U3, debido a este efecto de saturación facilite a energizar la bobina RL2 con la conexión de la tensión VP12, por consiguiente activa RL2 para que permita conducir la tensión VP24, por tanto conduciendo la tensión VP24 active el relay RL5, así pues permita acceder la señal NORL5 cuyo fin se prenda el led D6; ya que con esta acción representa ejecución de arranque del motor.

Si en la salida de la patita RB7 del microcontrolador obtiene un nivel lógico bajo (0 voltios), pues no surge ningún efectos de cambio para el optocoplador U2, energización de la bobina RL2 y excitación de la bobina RL4 por lo tanto el led D5 se encuentra apagada; ya que este led D5 nos muestra que no hubo ejecución de arranque del motor.

- En el subcircuito3: cuando en la salida de la patita 34 (RB1) obtiene una tensión de nivel lógico alto (5 voltios), entonces se satura el optocoplador U4, que a su vez activa el relé RL3 permitiendo así facilitar la conducción de la tensión VP24, por lo cual se activa el

relay RL5, permitiendo la conducción de la señal NROL6, con el propósito que se prenda el led D7, ya que está acción representa la desconexión de la energía de alimentación del motor.

Si en la salida de la patita RB1 obtiene un nivel lógico bajo, que por consiguiente el led D7 se encuentre apagado; ya que el led nos indica que no hay acción de desconectar la alimentación de energía del motor.

Tener en cuenta que es indispensable conectar un diodo en paralelo con la bobina del relé o relay, como protección frente a los picos de fuerza contra electromotriz producidos por la carga inductiva de la bobina en el momento de la conmutación, se describirá la colocación de los diodos de la siguiente manera: en el caso del subcircuito1; el diodo D1 está en paralelo con la bobina de RL1 y el diodo D8 está en paralelo con la bobina RL4, en el caso del subcircuito2; el diodo D2 está en paralelo con la bobina de RL2 y el diodo D9 está en paralelo con la bobina de RL5, caso del subcircuito3; el diodo D3 está en paralelo con la bobina RL3 y el diodo D10 está en paralelo con RL6. Y además se usó optocopladores para este circuito de salida, para su protección del microcontrolador PIC.

### 3.2.3. Algoritmo de la programación del PIC 16F877A

La programación en el PIC se muestra la lógica en un diagrama de flujo (revisar en la figura 3.8), cuya labor es recepcionar valores, ejecutar instrucciones y validar la ejecución de la tarea (transmitir) ya que estas instrucciones están asociados a los botones del interfaz de usuario (revisar el capítulo 3.2.8). El algoritmo se explicará en forma breve que lo conforman: inicio del programa, asignación de variables, ingreso de datos, las condicionales, validación de datos ingresados, respuesta de la revisión de estado del motor y finalización del programa. En el compilador que se uso fue CCS en C, ya que este compilador contiene librerías para una mayor facilidad de programar y fácil entendimiento para la lógica que se muestra en la figura 3.8, comenzamos con la breve descripción del programa. Entonces se inicia el programa en la espera de un carácter; que luego se declara, se define y se asignan variables; mediante la función getc() captura el carácter que luego asigna a la variable "x", ya que esta función getc() espera un carácter del dispositivo RS232 y lo retorna el carácter recibido, a su vez este carácter conforma 8 bits en ascii. En las condicionales se encontraron 5 casos de las cuales 4 son condicionales en secuencia y el otro caso es para el resto de las opciones , que a continuación se explicará en forma breve la siguiente manera:

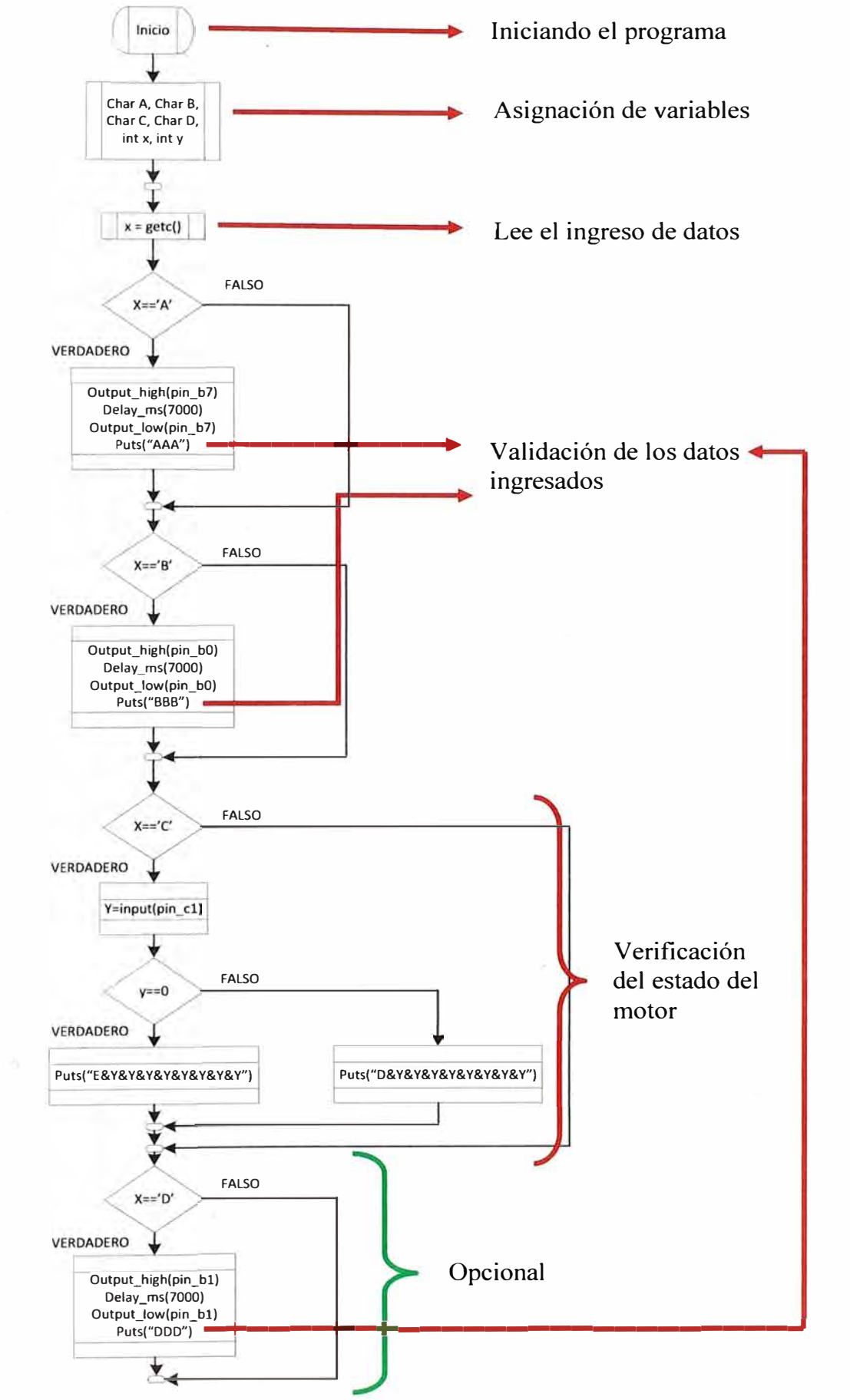


**Figura 3.8** El algoritmo de la programación en el PIC16F877A

- Para el caso, si la variable "x" es asignada el carácter "A", cuando es afirmativa obtiene una salida del pin RB7 del microcontrolador PIC en un nivel lógico alto (5 voltios) por unos 7 segundos que luego queda a un nivel lógico bajo (0 volitos), ya que está función puts() envía el carácter en cadena mostrando su respuesta de 3 caracteres "AAA"; y cuando es falso pasa la siguiente opción.
- Para el caso, si la variable "x" es asignada el carácter "B", cuando es afirmativa obtiene una salida del pin RB0 en un nivel lógico alto por unos 7 segundos, después se muestra a un nivel lógico bajo, entonces en forma inmediata responde los 3 caracteres "BBB"; y cuando es falso pasa la siguiente opción.
- Para el caso, si la variable "x" es asignada el carácter "C", cuando es verdadera; la variable "y" verifica la patita C1 del microcontrolador PIC en que nivel lógico se encuentra, entonces si la variable "y" es asignada el valor de "0, cuando es verdadera envía un conjunto de caracteres mostrando su respuesta de 9 caracteres cocatenadas "E&Y&Y&Y&Y&Y&Y&Y"; y cuando es falso envía otro conjunto de caracteres mostrando su respuesta de 9 caracteres cocatenadas "D&Y&Y&Y&Y&Y&Y&Y"; cuando es falso la asignación del carácter de la variable "x" pasa la siguiente opción.
- Para el caso, si la variable "x" es asignada el carácter "D", cuando es afirmativa obtiene una salida del pin RB1 en un nivel lógico alto por unos 7 segundos, después se muestra a un nivel lógico bajo, entonces en forma inmediata mediante la función puts() de CCS responde los 3 caracteres "DDD"; y cuando es falso fin del programa.
- Para todos los casos que no cumplen las condicionales secuenciales hace que finalice el programa, por lo cual estarán a la espera de otro carácter para su ejecución.

Cuando responde la validación de los caracteres para cada caso ante mencionado, es enviado al interfaz de usuario para su confirmación que ha sido ejecutado la tarea encomendada. Tener en cuenta que estas instrucciones enviadas por el pic se encuentran con el equipo serial que lo convierte a señales ethernet, ya que estas señales contienen el dato que serán mostradas por el interfaz de usuario (revisar el capítulo 3.2.8).

### 3.2.4. Simulación del circuito de control microcontrolador PIC

Cuando se hizo la simulación del esquema completo (ver figura 3.7), no se puede ejecutar la simulación debido a que se presentó un cuadro de mensaje de alarma mostrándonos e indicándonos la falta de recursos de memoria en el programa proteus 7.1, por lo cual se optó hacer la simulación por partes, como se muestra la siguiente figura 3.9. Estos leds D1, D2 y D3 comparando con la figura 3.7 representan el circuito que falta,

entonces si comprobamos se prenda estos leds, se verifica que la energía eléctrica está presente para la ejecución del circuito faltante. Es decir en la simulación nos importa si llega energía eléctrica o no, cuando el led D1, D2, D3 prende se acciona el relay RL1, RL2 y RL3 en forma respectiva. Para el caso mostrado en la figura 3.9 se ha accionando el relay RL1, que nos indica que hay energía eléctrica de 12 voltios.

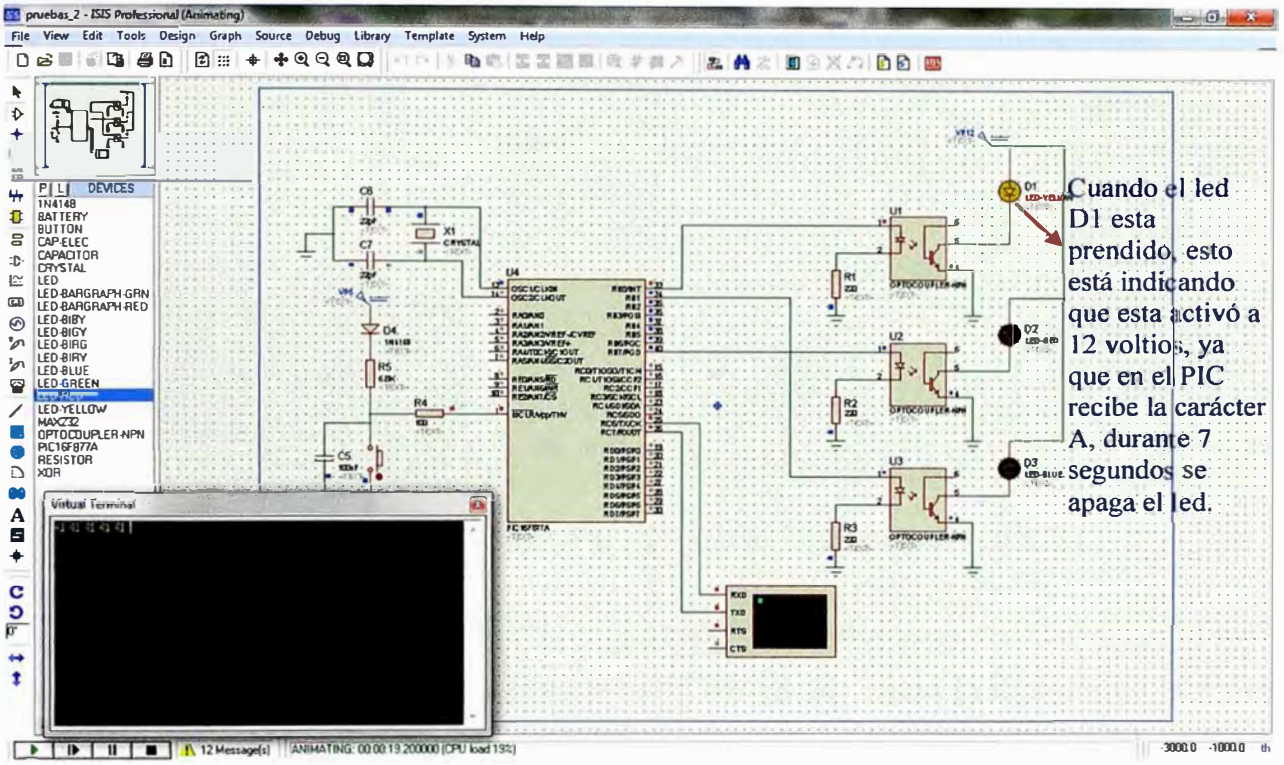


**Figura 3.9** Simulación en el PIC 16F877A habilitado en RB7

En la consola de virtual terminal, aparecen ciertos números que estan en modo hexagesimal. El número hexagesimal "41" está en modo ascii de la letra "A" mayúscula, por lo cual se muestra que cuando ingresa la letra "A" el led D1 se enciende, entonces despues de un cierto momento en el virtual terminal se visualiza con "414141" que es la cadena de letras "AAA", validando que ha cortado la energía eléctrica del circuito de control base de arranque y apagado mediante reles para hacer el proceso de parado del motor.

Para el caso de hacer la tarea de arranque, que a continuación se muestra la figura 3.10, se verifica en el led D2 este encendido, por consiguiente en la consola aparecerá el número 42, que es en modo hexagesimal de la letra "B", es decir, cuando ingresa la letra B, enseguidamente el led D2 se prende durante unos 7 segundos; entonces, despues se visualiza en el virtual terminal el 424242 que es la cadena de letras "BBB" , ya que

representa que ha accionado mediante los circuitos del relé activandole la energía eléctrica al motor, para que pueda arrancar el motor.

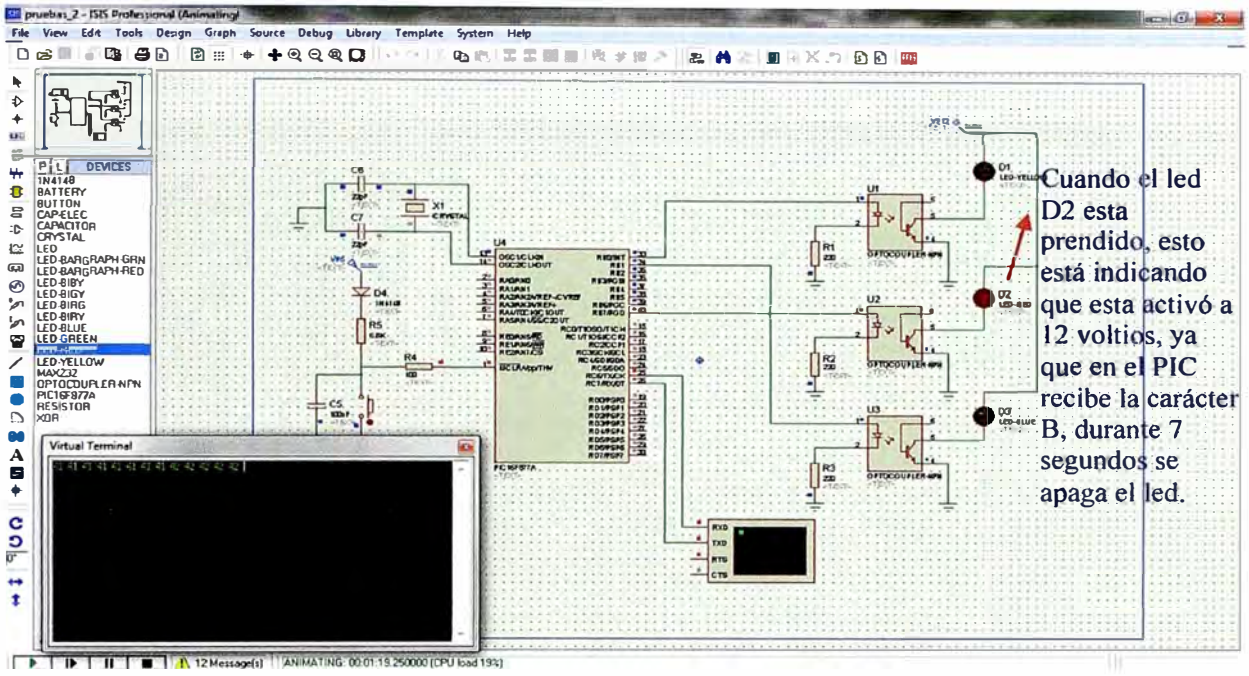


**Figura 3.11** Simulación en el PIC 16F877A habilitado en RB0

Si tecleamos introduciendo la letra mayúscula "C" en la consola "virtual terminal", y el pulsador se encuentra abierto en el circuito de entrada como lo muestra en la figura 3.12, entonces, en la consola virtual terminal aparecerá una cadena "DYYYYYYYY", cuya información es enviada al interfaz de usuario que representa una instrucción que se encuentra desactivado el motor.

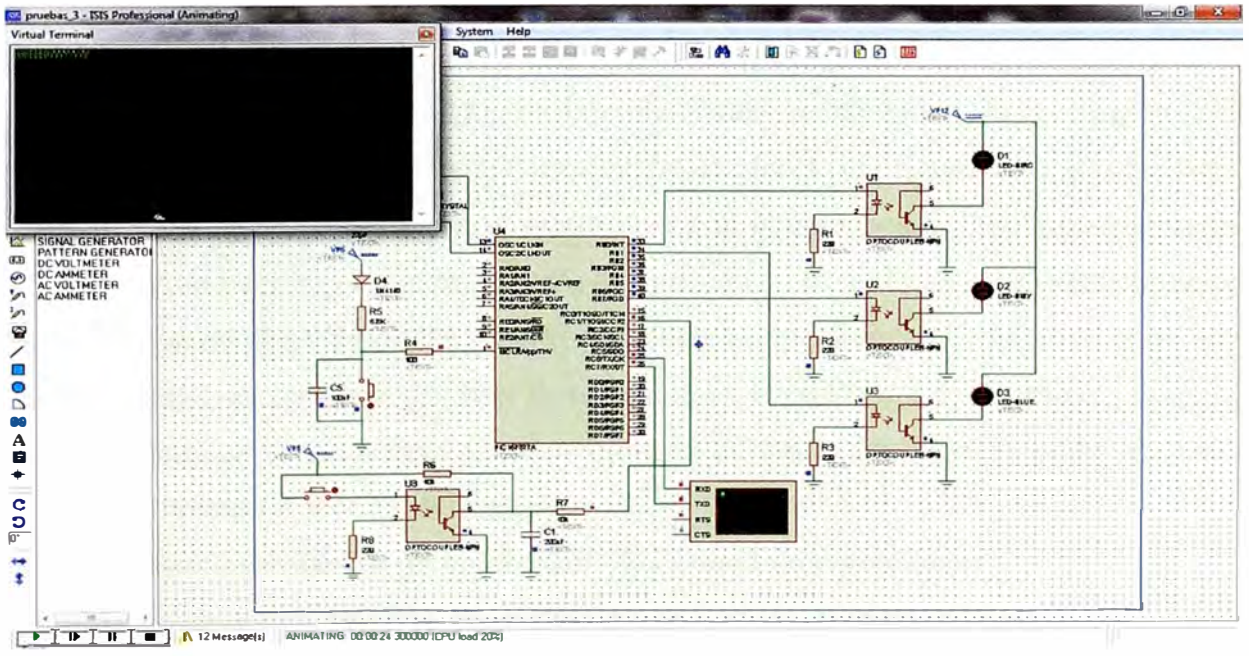

**Figura 3.12** Simulación en el PIC 16F877A verificando disable en el motor

Si tecleamos la letra mayúscula "C", y el pulsador está cerrado como lo muestra en la figura 3.13, entonces en la consola virtual terminal aparecerá una cadena "EYYYYYYYY", que representa y un aviso que está activado del motor o está en funcionamiento.

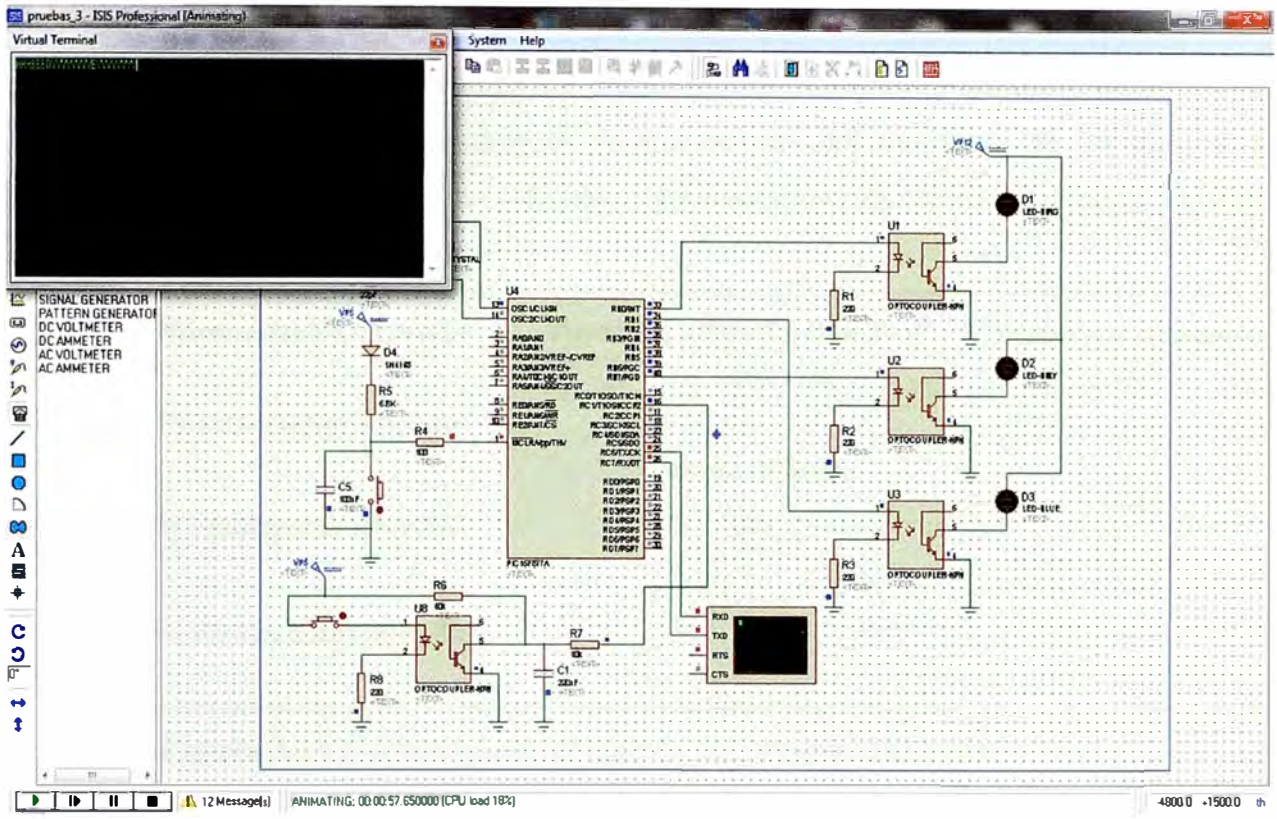


**Figura 3.13** Simulación en el PIC 16F877A verificando enable en el motor

Teniendo en cuenta que para modo de simulación se cogió el pulsador. En las pruebas el circuito de control de relevadores, circuito de control PIC, y el ventilador dc, este pulsador el que está conectado con la patita 1 del optocoplador U8 (ver figura 3.13), representa el relay kp del circuito de control de relevadores (ver figura B.8), ya que hace la misma función en las pruebas con el motor dc y con la simulación. Pero para las pruebas en campo como ventilador minero, bomba de agua, etc., este pulsador representa un tarjeta de adquisición de datos desde las señales del sensor temperatura que coge o recopila datos de temperatura, pero por razones ante mencionado en el capítulo 1.4, las pruebas con estos motores eléctricos (ventiladores minero, bombas de agua, etc.), no se pudo acoplar el sensor de temperatura al rotor del motor.

### 3.2.5. Diseño y simulación de un circuito de control de relevadores

Antes de hacer pruebas en campo en la Unidad Minera, se previó construir un hardware que abarca un conjunto de relays (relevadores) y elementos electrónicos, desde el

concepto del circuito de control básico. Se hicieron las pruebas necesarias desde el prototipo electrónico al que se lo llama circuito de control PIC al cuál se conectó con el hardware construido en base de relays al que se le llama circuito de control de relevadores cuyo propósito es controlar el arranque y apagado de un ventilador de 24 Vdc. Este diseño construido se observa en la figura B.8 (anexo B), que a continuación se explicará este esquema, con la ayuda de la simulación del programa automation studio, y para un buen entendimiento del esquema de control de relevadores del motor dc de 24 voltios, se indicará algunos símbolos comunes de automation studio como se muestra en la siguiente figura 3.14.

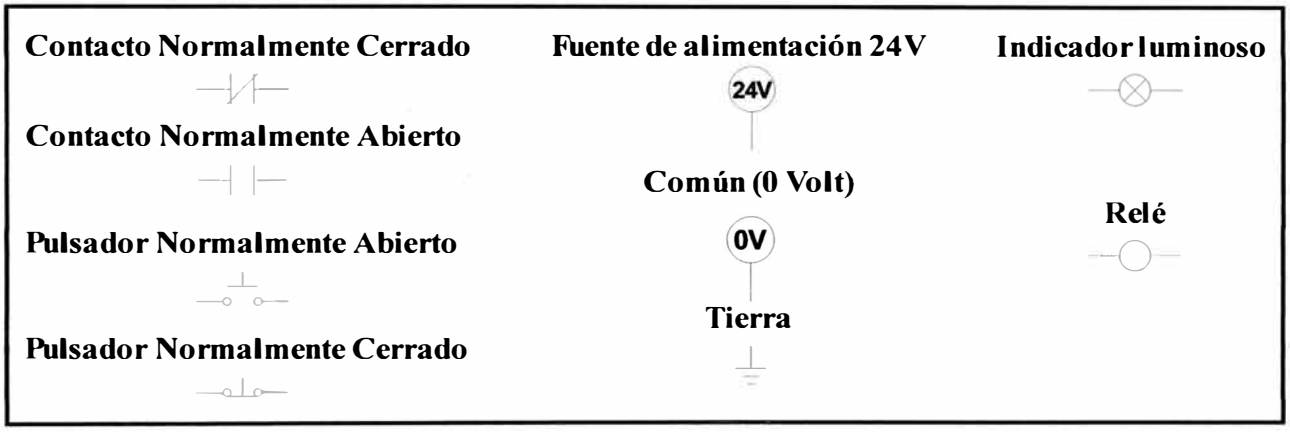


**Figura 3.14** Simbología de contactos del programa automatio studio

En el siguiente diseño simulado por el programa automatio studio se explica la secuencia del arrancamiento del ventilador dc de 24 voltios que se muestra en la figura 3.15 y luego se describe la parada del ventilador dc de 24 voltios que se muestra en la figura 3.16. Entonces, a continuación se detalla la secuencia de arranque del motor:

i) Se examina el indicador luminoso "POWER_P" se encuentra activo debido a la energización presente del voltaje de 24 Vdc.

ii) Al presionar el pulsador "start", permite pasar la corriente por la bobina Kn, cuyo efecto se energiza provocando así, sus contactos cambian de estado es decir el contacto Kn normalmente abierto se cierra.

iii) Mientras el pulsador "stop" no se lo presiona, entonces pasa la corriente a la bobina Km, por tanto se cierra el contacto normalmente abierto Km.

iv) Luego que los contactos Km y Kn se encuentran cerrados debido a la energización de sus bobinas Km y Kn cuya causa se prende el indicador luminoso "START_P".

v) Como la bobina Ko está en paralelo con el indicador luminoso "START_P", ya que este indicador luminoso se encuentra prendido, entonces se energiza la bobina Ko.

vi) Debido a que se energiza la bobina Ko su contacto normalmente abierto se cierra y el contacto normalemente cerrado se abre, así pues interrumpe la corriente de tal manera no se prende el indicador luminoso "STOP_P" (en la figura 3.15 se aprecia pintado de color negro el símbolo del contacto cerrado Ko es decir se encuentra abierto).

vii) A su vez el otro contacto normalmente abierto ko se cierra (en la figura 3.15 se aprecia pintado de color rojo el símbolo del contacto abierto Ko es decir se encuentra cerrado), por lo cual permite energizarse la bobina Kp.

viii) Debido a que se energiza la bobina Kp normalmente abierto se cierra, permitiendo que se traslade la corriente eléctrica provocando que se prenda el indicador luminoso "VERIFICAR".

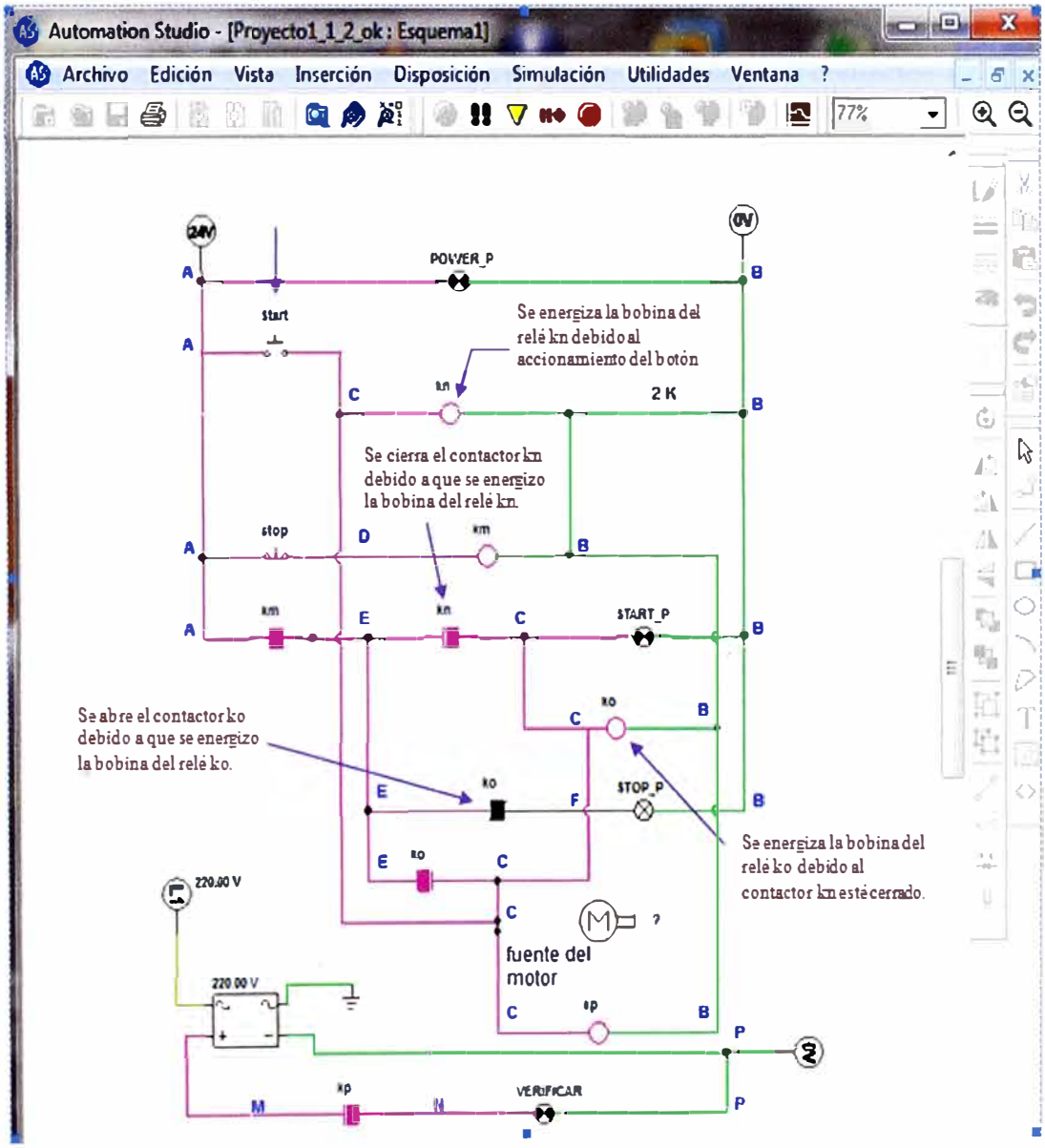


**Figura 3.15** Proceso de arrancar en la simulación de un diagrama de un sistema de control de 24 Voltios DC

A continuación se indicará la secuencia de parada del motor desde que se encuentra en operación se muestra la figura 3.15 hasta el momento que se pare el motor así se muestra en la figura 3.16, así:

i) Al presionar el botón "stop", hace que interrumpa el flujo de corriente ya que se desenergiza la bobina Km, por ende los contactos Km queda a su estado normal.

ii) Debido que el contacto Km se encuentra en contacto normalmente abierto por tanto interrumpe el flujo de corriente que alimenta al indicador luminoso "STAR_P", es decir se apaga.

iii) Y a su vez la bobina Ko que se encuentra en paralelo con el indicador luminoso "STAR_P", se desenergiza porque el indicador luminoso se encuentra apagado y además también la bobina Kn se desenergizará por falta de alimentación de corriente.

iv) Ya que la bobina Ko no se encuentra energizada entonces sus contactos Ko queda a su estado normal, por ende el indicador luminoso "STOP_P" se prende.

v) Sin embargo, la bobina Kp no se encuentra energizada entonces sus contactos está en su estado normal entonces el contacto Kp se encuentra normalmente abierto, por el cual el indicador luminoso "VERIFICAR" se encuentre apagado.

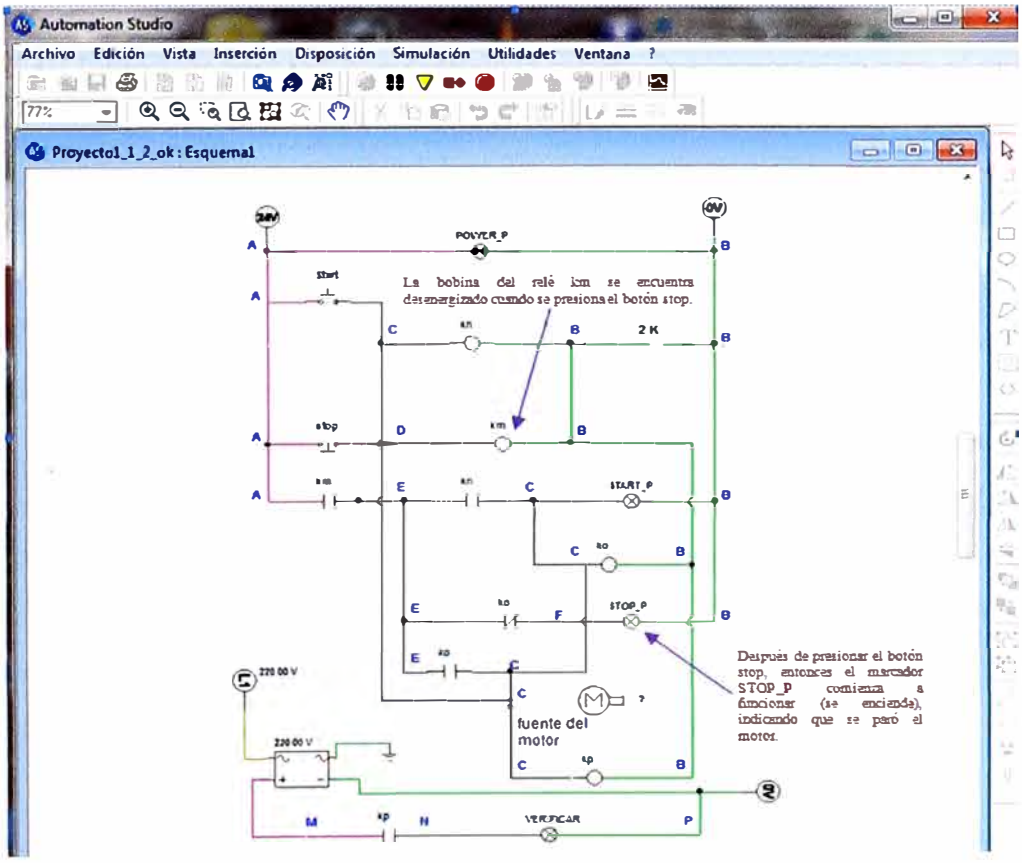


**Figura 3.16** Proceso de parar en la simulación de un diagrama de un sistema de control de 24 Voltios DC

### 3.2.6. Diseño de indicadores de leds para su verificación de las operaciones de los circuitos ya tratados

Hay 2 tarjetas electrónicas hechas en fibra de vidrio son: tarjeta electrónica conformada por leds que permite revisar las tensiones dc (ver figura B.4) 5,12 y -12 voltios y la otra tarjeta permite verificar mediante leds el estado, el arrancado, el parado del motor dc (ver figura B.2) y además la alimentación de energía, se describen a continuación:

La primera tarjeta tiene como finalidad de verificar si está funcionando correctamente los voltajes dc de 12 voltios, 5 voltios y –12 voltios de la tarjeta de alimentación de 3 salidas. Con estos indicadores, que se muestran en la figura 3.17 simulado en proteus, cuando está prendido el led muestra que dicho voltaje testeado, está en funcionamiento con el voltaje que le corresponde.

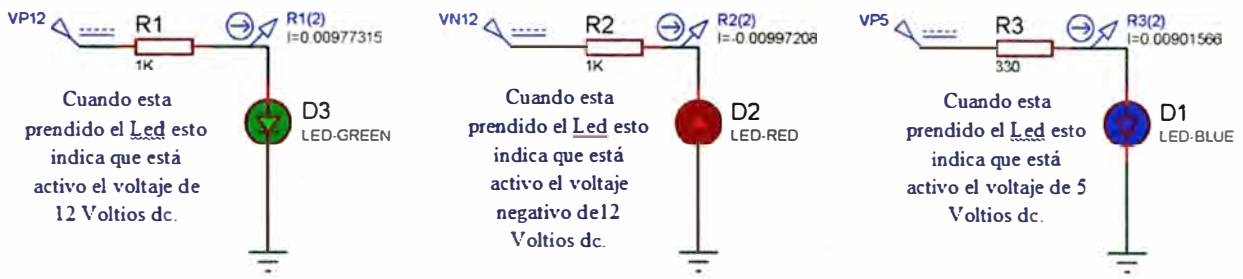


**Figura 3.17** Indicador para ver el estado de la tarjeta de alimentación para el circuito de control PIC

Para la segunda tarjeta su función es verificar el estado del circuito de control de relevadores (ver figura 3.18), cuya finalidad es de revisar el estado del motor si está funcionando o no (led D2 se prende), además, revisa la energía eléctrica que lo está alimentando al circuito de control de relevadores (led D1 se prende). También revisa si ha accionado el botón start (led D3 se prende) y, por último, si ha accionado el botón stop (led D4 se prende), para caso contrario no están prendidos los led es que hay desperfecto con la tarjeta debido a sus elementos electrónicos o está apagada el equipo de control de relevadores. En la figura 3.18 simulado en proteus, las corrientes que circulan por los 4 leds estan entre 9 a 10 mA, cuando esta activo el led D1 de tensión "in power", nos indica que la energía eléctrica alimenta al equipo. Si el led D2 de tensión "in state" esta activo, nos indica que el motor esta funcionando. Si el led D3 de tensión "in start" esta activo, indica que ha accionado el pulsador start y comenzo a arrancar el motor. Y si el led D4 de tensión "in stop" esta activo, indica que ha accionado el pulsador stop y se paro el movimiento del motor.

La diferencia en el diseño electrónico en la tarjeta que indica el estado del circuito de control de relevadores es que entre el led state y los 3 leds (star, stop, power) tienen gnd diferente, pero en el caso que hubiera pertenecido la misma tierra, el led blue (status) no hubiera podido indicarnos correctamente la lectura, dado que su indicador mostraría otra respuesta, no estaría viendo el estado del motor.

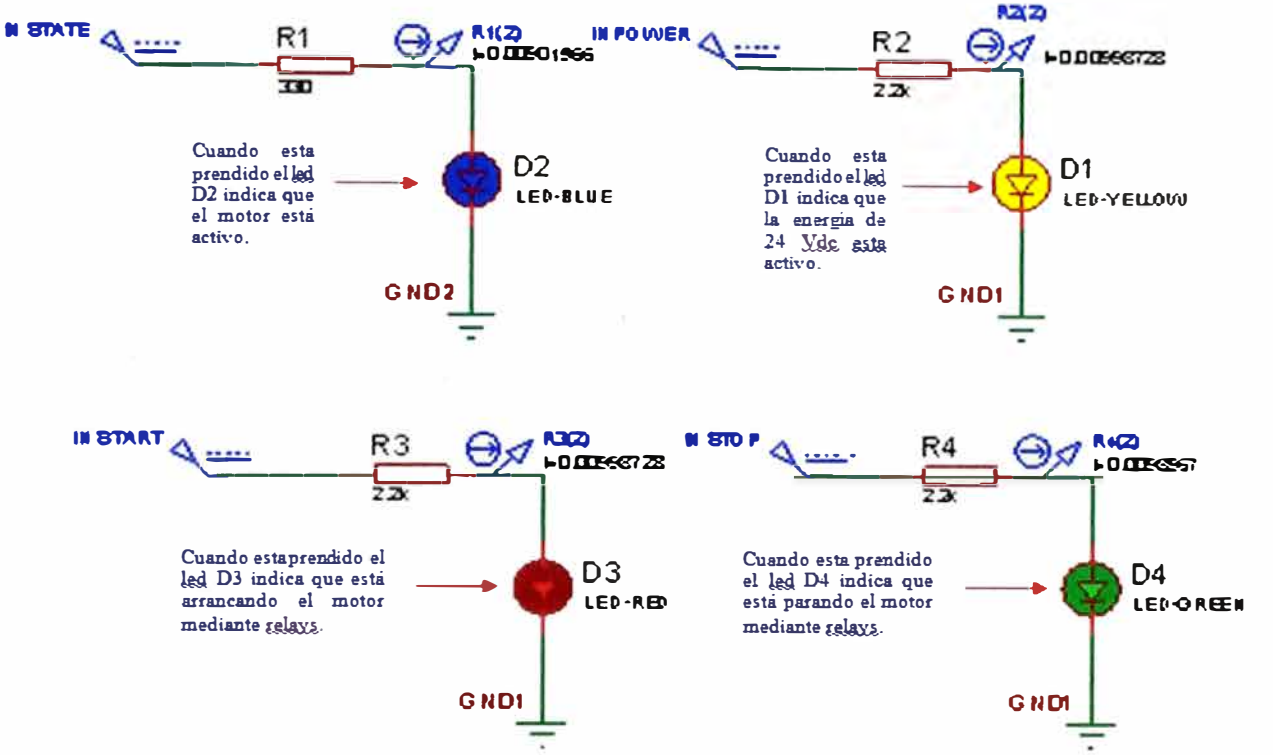


**Figura 3.18** Indicadores para ver el estado del circuito de control de relevadores

### 3.2.7. Conexión del servidor serial TIBBO

Este servidor serial se usó para adaptar al microntrolador PIC que cuya función es, convertir protocolos seriales a protocolos tcp/ip, ya que es un conversor serial-ethernet que se conecta externamente cualquier equipo serial (RS-232/RS-485) a una red LAN ethernet. El servidor DS100 se utilizó para este proyecto, que se encuentra conectado con la entrada o salida del puerto db9 del módulo de la tarjeta de control PIC. Al mismo tiempo, este servidor tiene un puerto de red RJ45 como entrada o salida por donde emite la información tramas ethernet y que se conecta con un equipo de redes compatibles (IEEE 802.3). A este servidor se le asigna una ip y una máscara como se muestra a continuación en la siguiente figura 3.19.

Las características técnicas de este servidor serial tibbo DS100 se indica a continuación:

- Enlaza en red cualquier equipo serial (RS232/RS485).

- Enruta transparentemente los datos entre su puerto serial y la red ethernet 10 Base T.
- Se comunica con otras estaciones por red usando los protocolos TCP o UDP.
- Flexible. Cuenta con muchos parámetros de funcionamiento configurables.
- Todos los parámetros pueden ser configurados a través del puerto serial o por la red.
- Soporta configuración de la dirección IP por la red.
- Soporta comandos "al vuelo" para el cambio remoto inmediato de los parámetros del puerto serial (baudrate, paridad, etc).
- Su firmware interno puede ser actualizado por el usuario.
- Software y firmware descargables desde el sitio web del fabricante [15].
- Compatible con elo módulo EM100 (de hecho, el DS100 incorpora un EM100 en su interior).


**Figura 3.19** Conexión en la conversión de protocolos usando el servidor serial tibbo DS100

Cuando se configura el servidor serial Tibbo DS100, se tiene que instalar el software que se descarga del sitio web del fabricante, esta instalación genera 5 subprogramas de las cuáles se menciona los 3 importantes que son DSManager, Port Monitor, VSP Manager. Cuando se utiliza el subprograma DSManager, mediante manipulación de ventanas y pestañas, cuyos parámetros básicos se tiene que configurar son (ver figura 3.20): el protocolo de transporte a utilizar es UDP, el modo ruteando es server (slave), aceptación de conexión desde cualquier IP; para la pestaña de puerto serial, (ver figura 3.21) interface serial debe estar en modo automático, el RTS/CTS control de flujo de estar en modo local, el modo DTR debe estar en modo libre y remoto, la velocidad del serial configurarlo con 9600 bps, los datos en bits debe estar 8 bits y además el estado de la energía DTR debe estar en bajo. En la pestaña de red se selecciona y se configura la ip del servidor serial y el nombre. El subprograma Port Monitor es para monitorear el puerto configurado. Para este caso del proyecto se configuro el COM10 en el programa TIBBO.

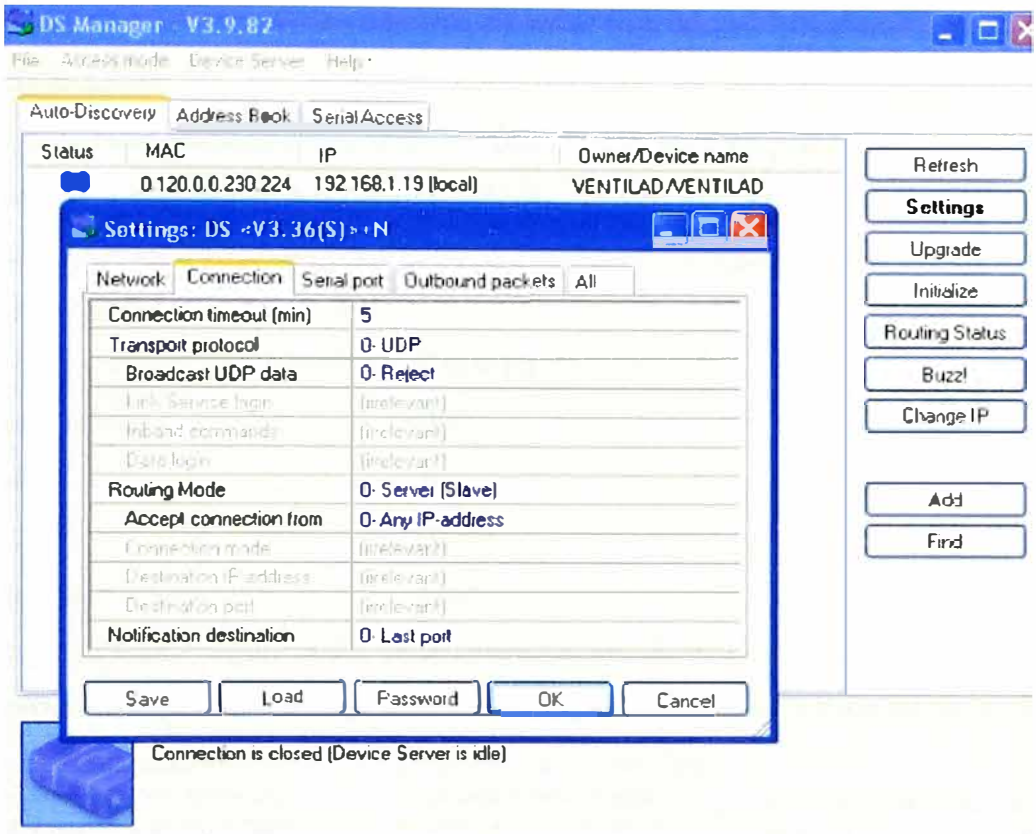


**Figura 3.20** Parámetros básicos de conexión a configurar en el DS Manager tibbo DS100

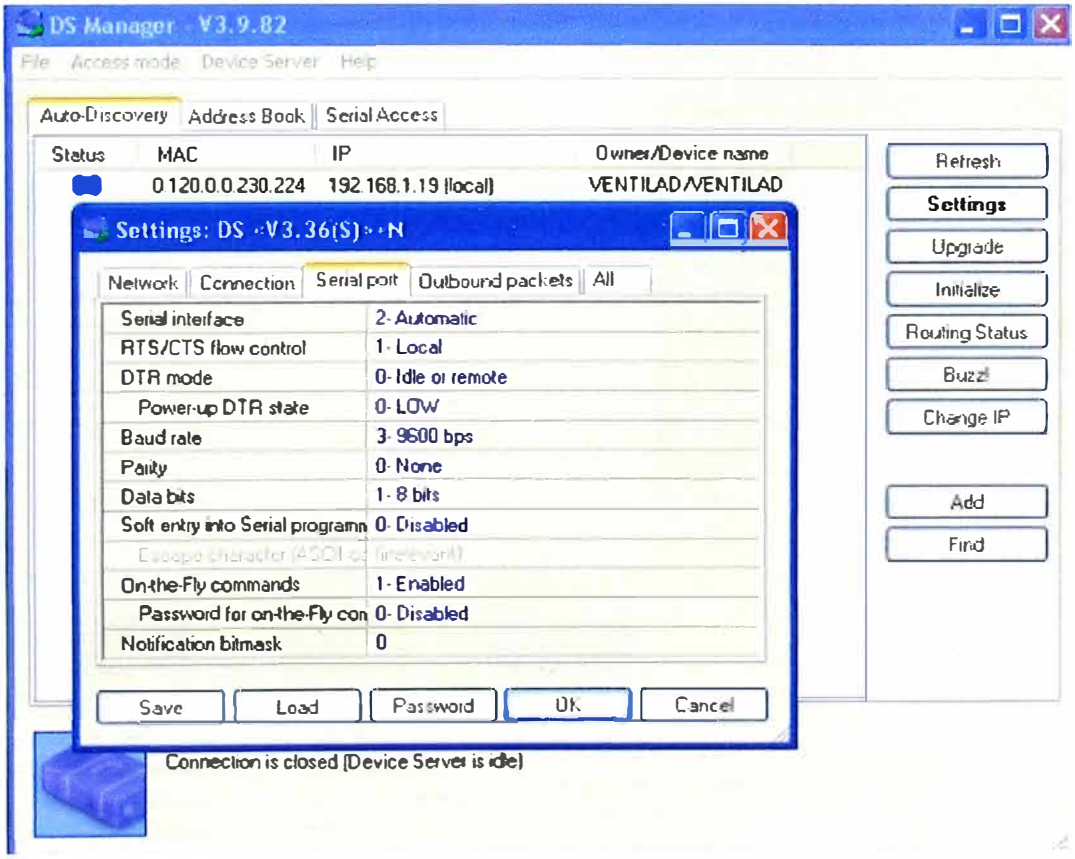


**Figura 3.21** Parámetros básicos a configurar en serial port de DS Manager tibbo DS100

### 3.2.8. Presentación del programa diseñado y una breve programación en C# del interfaz de usuario

El programa sistema de control ha sido programado en c sharp de visual .net 2008, se explicará en forma breve la programación que es más relevante. El interfaz de usuario se presenta en la figura 3.22, para configurar el puerto de configuración se tiene que llenar los datos, de este modo:

| | | |
|---|---|---|
| Puerto | : | COM10 |
| Bits por segundo | : | 9600 |
| Bits de datos | : | 8 |
| Paridad | : | None |
| Bits de Parada | : | One |

En la siguiente figura 3.22, es la presentación de usuario para el sistema de control que esta programado en c#, que a continuación se describirá la función de cada elemento:

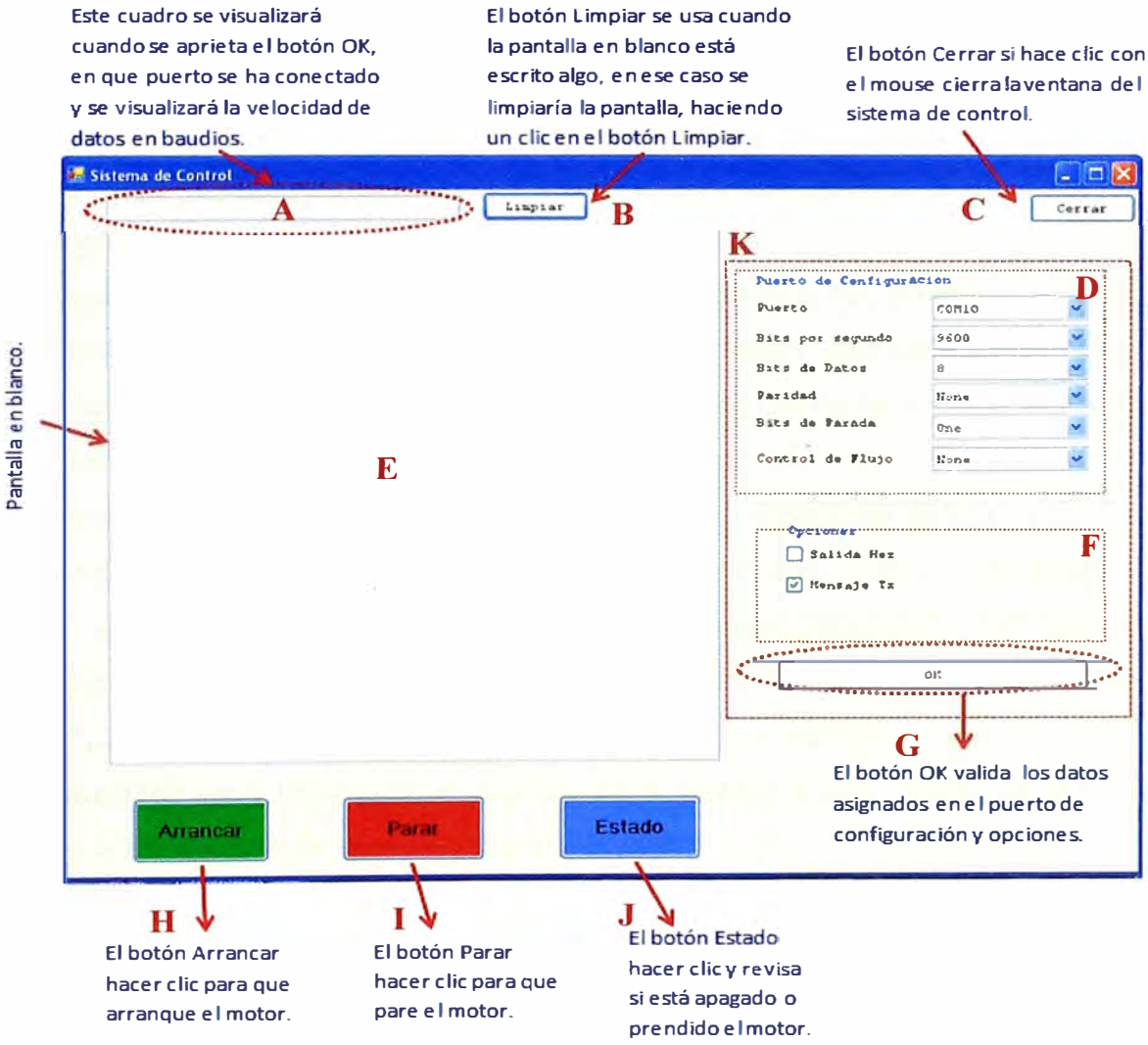


**Figura 3.22** Presentación de la ventana programado para el sistema de control

**A. Validación y operación de los datos configurados de la comunicación serial:** en este campo indica si está conectado al serial y en que com se conectó, cuyos valores tiene que coincidir con los datos configurados en el campo de puerto de configuración. Es decir para que acceda a la comunicación, coge los datos configurados, luego lo verifica y sincroniza, si es válido se visualiza en la parte derecha de la esquina superior del programa sistema de control, indicándonos la conexión con el equipo serial.

**B. Botón limpiar:** cuya función es hacer una limpieza en el campo de visualización de pantalla quedando en blanco, sin ninguna instrucción encomendada o respuesta.

**C. Botón cerrar:** cuya función es un medio de salir del programa.

**D. Puerto de configuración: c**uya finalidad es configurar mediante pestañas los parámetros de la comunicación serial para poder así conectarnos con el servidor serial.

E. **Pantalla del sistema de control:** cuya función es visualizar las instrucciones que se ejecuta como también a su respuesta mostrándolo la validación de la ejecución de la instrucción. Tener en cuenta que las instrucciones son solamente 3 para este caso específico que son: pulsar el botón "arrancar", "parar" y "estado".

**F. Opciones:** cuya finalidad es, como se visualiza la forma de respuesta en la pantalla en blanco, habilitando mediante opciones, las cuales son: hexagesimal (salida Hex) y transmisión (mensaje Tx).

Cuando habilitamos la opción "salida Hex", en la pantalla del programa nos muestra en modo hexagesimal por ejemplo si apretamos el botón "parar" que está asociado con la tecla mayúscula "A" en su programación entonces aparecerá en la pantalla del sistema de control 0x41 que está en modo ascii la letra mayúscula "A".

Cuando habilitamos la opción "mensaje Tx", en la pantalla del programa se visualizará el mensaje de la acción del botón que ha sido ejecutado, por ejemplo si apretamos el botón "parar" se visualiza en la pantalla del sistema de control "apagado" y además dentro de unos segundos se visualiza su respuesta de la transmisión mostrando los 3 caracteres "AAA", que nos indica su confirmación que ha sido ejecutada la instrucción apagar.

**G. Botón OK:** cuya función es validar los datos asignados en el puerto de configuración serial y la habilitación de la opción requerida.

**H. Botón arrancar:** cuando pulsa este botón habilita la instrucción que está asociado con la letra "A" en ascii es 0x41, cuyo fin es accionar el arrancado del motor.

**I. Botón parar:** cuando pulsa este botón habilita la instrucción que está asociado con la letra "B"en ascii es 0x42, cuyo fin es parar el movimiento del motor.

**J. Botón estado:** cuando pulsa este botón habilita la instrucción que está asociado con la letra "C" en ascii es 0x43, que permite visualizar en la pantalla en blanco el estado del motor, es decir, si está "encendido" o "apagado" el motor.

**K. Parámetros de configuración:** conforma puerto de configuración, opciones y botón ok. Es decir son los datos que se tiene que configurar, mostrar y aceptar para poder establecer la comunicación.

Tener en cuenta que el interfaz de usuario se programó en C Sharp (C#), y se explica con anterioridad los elementos de la plantilla que conforma este sistema de control para luego programarlo cada uno de estos objetos para su determinada finalidad. En la siguiente figura 3.23, se muestra la estructura y los componentes que se necesita para poder programar el proyecto en mención:

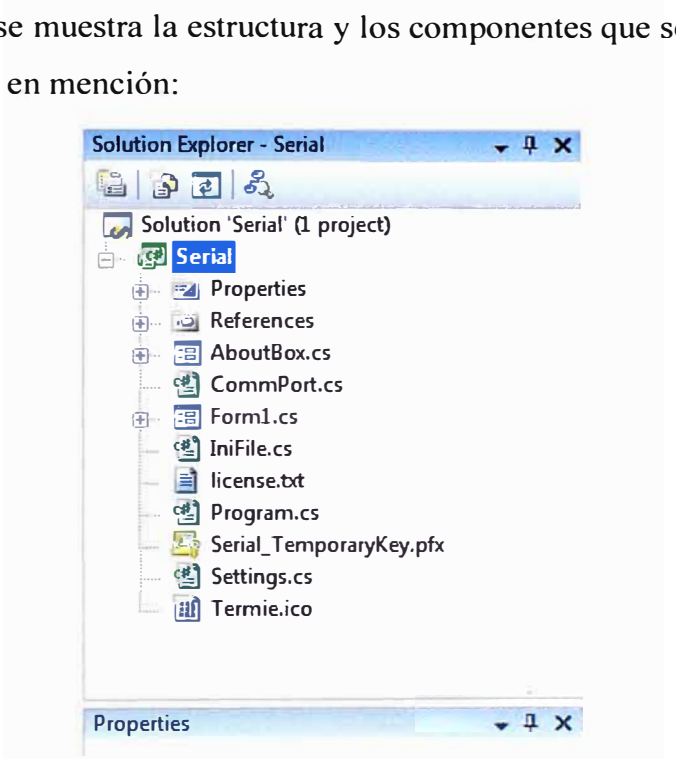


**Figura 3.23** En el presente proyecto la solución explorer serial en el lenguaje c#

Ya que la programación para este sistema de control consta de varias partes, así que haremos una breve explicación a los objetos más relevantes, ya que son de suma importancia, las cuales son: la programación botón "arrancar", la programación botón "parar", y la programación botón "estado". La plantilla a programar se encuentra en el fichero Form1.cs. y además se encuentra los códigos programados que estan asociados a los objetos que abarca la plantilla. El fichero license.txt se encuentra habilitada como lo indica el proyecto solución explorer.

A continuación se brinda la programación de los botones "arrancar", "parar" y "estado", entonces:

- La programación para el botón *"arrancar"*:

```
private void btnArrancar_Click(object sender, EventArgs e)
{
CommPort com = CommPort.Instance;
com.Send("B");// envía el cárater B = 0x42, cuando pulsa este boton verde
 if (Settings.Option.LocalEcho)
  {
   outputList_Add("Encendido" + "\n", sentColor);
   //Acá visualiza la pantalla que se encendio el motor.
  }
}
```

- La programación del botón *"parar"*:

```
private void btnParar_Click(object sender, EventArgs e)
{
 CommPort com = CommPort.Instance;
 com.Send("A"); / / envía el cárater A = 0x41, cuando pulsa este boton  parar
  if (Settings.Option.LocalEcho)
   {
    outputList_Add("Apagado" + "\n", sentColor);
    //Acá visualiza en  la pantalla, que esta apagado el motor.
   }
}
```

- La programación del botón *"estado"*:

```
private void btnEstado_Click(object sender, EventArgs e)
{
CommPort com = CommPort.Instance;
com.Send("C");/ / envía el cárater C= 0x43, cuando pulsa este boton  status
 if (Settings.Option.LocalEcho)
  {
   outputList_Add("Estado" + "\n", sentColor);
```

```
  //Acá visualiza la pantalla el estado del motor.
  }
}
```

### 3.2.9. Implementación y armado del presente proyecto

Se ha ensamblado en forma exitosa el módulo propuesto para el control de motor de inducción para sus diferentes tipos de arranque de motor a través del circuito de control básico. Esta herramienta como se explicó anteriormente tendrá una conexión en paralelo y el otro en serie al circuito de control base de arranque y apagado. Así mismo, este equipo integra al circuito de control base con sinnúmero de utilidades que brindan este circuito para poder controlarlos las técnicas de arranque para motores de inducción. Teniendo en cuenta que los reguladores de tensión están con sus respectivos disipadores, está tarjeta es de control o cerebro, que se encarga de ejecutar las tareas mediante instrucciones. Este cerebro está basado en el microcontrolador tipo PIC 16F877A de 40 pines. La función de este dispositivo es recibir los comandos de manera remota para ejecutarlas y transmitir el dato de aviso para la manipulación de control.

Este proyecto a modo de prueba consta de 2 partes:

1ra parte: El circuito de control PIC, para prueba y modo de demostración local se diseñó el circuito de control de relevadores para un ventilador dc de 24 voltios.

2do parte: Para hacer las pruebas con el ventilador minero, el circuito de control PIC (ver figura 3.24) se acoplará un conexionado de cable con el tablero eléctrico de control del arrancado y apagado del ventilador de 250 HP.

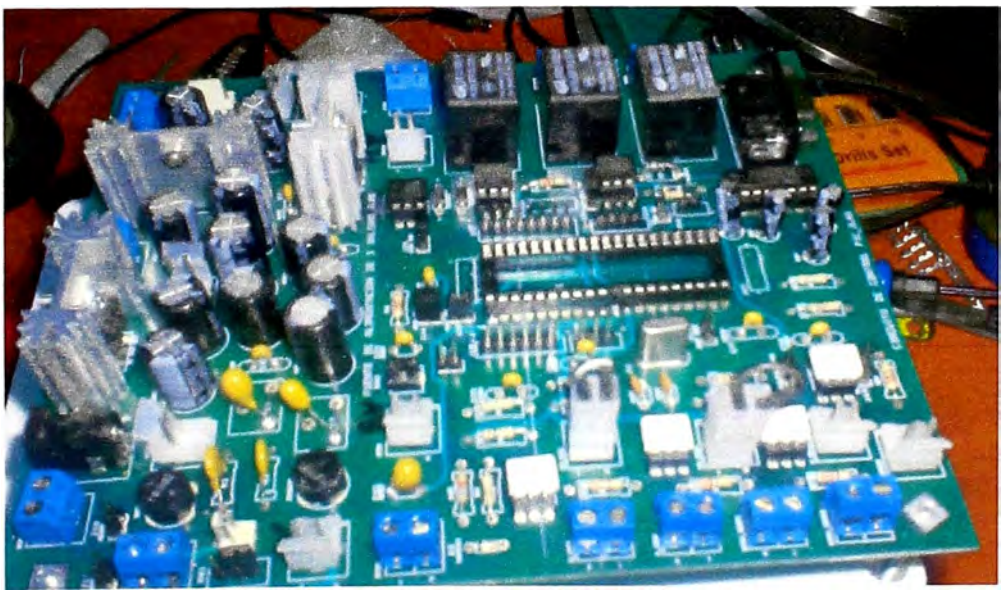

**Figura 3.24** Circuito electrónico montado en un microcontrolador PIC

En la figura 3.25 se muestra el contenido de la caja de control de relevadores que contiene un montando de base acrílica, taladreado y ajustado con tornillos; se colocaron las bases de los relays, conexiones de cables, borneras, son los que conforman el ensamblado del hardware del circuito de control de relevadores.

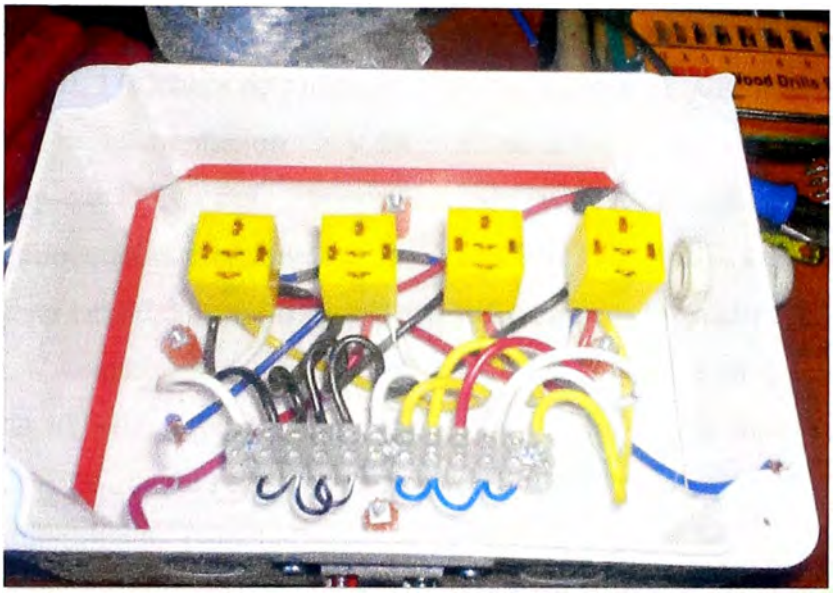

**Figura 3.25** Ensamblado del circuito de control con relevadores

**Figura 3.26** Ensamblado del circuito de control con relays

En la figura 3.26 se muestra conectados los relays a la base del relay y desde las borneras están conectadas de acuerdo con el esquema (ver figura B.8) a las botoneras star y stop, además hay un conector de 4 pines que se encuentra conectado con una tarjeta de fibra de vidrio que conforma los 4 indicadores luminosos con sus 4 resistencias. Estos leds son los indicadores del circuito de control con relevadores, que son: led power, start, stop

y status, estos leds ultraluminosos están soldados y montados en una tarjeta de circuitos electrónicos, y que cada color representa una acción.

El circuito de control PIC se muestra en la figura 3.27. El circuito de control de relevadores que conforma el ventilador dc de 24 voltios y una caja que abarca circuito de control para arrancar y apagar el motor, 2 botoneras uno normalmente abierto y el otro normalmente cerrado, 1 bornera de radio de 2 salidas es para la alimentar al ventilador dc, cables de entrada de alimentación 5 y 24 voltios, 2 cables que están en normalmente abierto, 2 cables que están en normalmente cerrado, y 2 cables que verifica el estado del motor que están conectadas mediante terminales aéreos con la otra caja que es el circuito de control PIC. Esta caja del circuito de control PIC, está conformado por el servidor serial que es un tibbo, 2 transformadores 220Vac a 24Vdc, 2 relays de 24Vdc/220 Vac y 3 pequeños relay de 12 Vdc, cable utp, conexiones de cables,1 cable de energía de 220 Vac, una entrada de 220Vac , cable serial y una bornera de 10 Amperios, por lo cual en la figura 3.27 se ve el conexionado para las pruebas en forma local a base de la alternativa de solución, y se observa que durante las pruebas en forma manual como cambia el indicador de luz en el circuito de control de relevadores que concuerda las tareas encomendadas, pulsándolo las botoneras azúl y rojo.

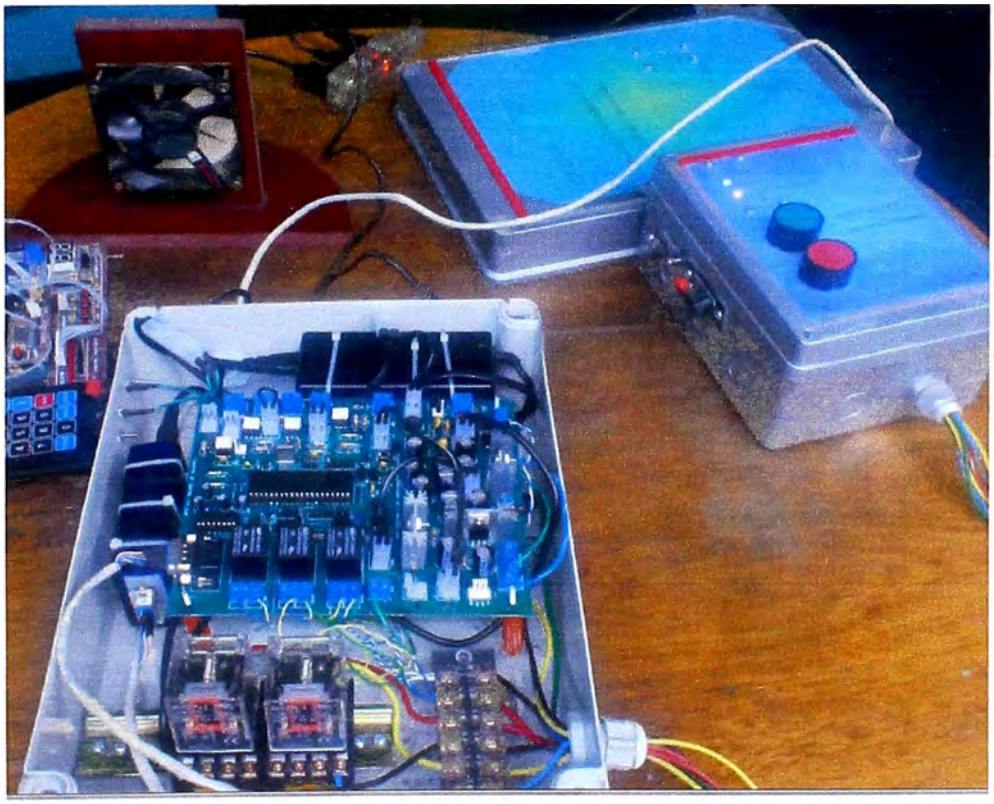

**Figura 3.27** Pruebas en forma local el circuito de control PIC y el circuito de control del relevador para arrancar y apagar el ventilador dc

### 3.3. Recursos humanos y equipamiento

El equipamiento para el circuito de control PIC se indica los siguientes:

**TABLA N° 3.1** Equipamiento para el hardware control PIC el cerebro parte 1

| EQUIPAMIENTO | DESCRIPCION | CANTIDAD |
|---|---|---|
| Leds ultrabrillantes | Indica cierta variable en operación | 3 |
| Plancha acrílico de un 3mm de ancho. | Proteger la cara del prototipo | 1/4 |
| Tornillos 1/8 de ¼,1/2,1/8 | Para ajustar y ensamblar en la caja | 20 |
| Terminales de 2,4,6 | Son terminales aéreos | 8 |
| PIC 16F877A | Microcontrolador PIC de la familia 16F | 1 |
| Max 232 | Un Integrado que permite comunicarse entre tx y rx del PIC con el serial conector de pc. | 1 |
| 4n35 | Son optocopladores | 7 |
| Puente diodos | Soporta 1 amperios | 2 |
| Termoresistencia | Fusible de 500 mA | 2 |
| Transformador 220Vac/24Vdc | Transformador de 2 amperios | 2 |
| Capacitador 10 uF 50V | Capacitador electrolítico | 4 |
| Capacitador 470uF 35V | Capacitador electrolítico | 7 |
| Capacitador 1000uF 25V | Capacitador electrolítico | 4 |
| Capacitador 0.1 uF 25V | Capacitador cerámico | 7 |
| Capacitador 27pF 35V | Capacitador cerámico | 2 |
| Cintillos de 15 cm | Por unidad | 20 |
| Bornera de 2 entradas | Bornera por unidad | 12 |
| Bornera de 3 entradas | Bornera por unidad | 3 |
| Relay de 24 Vdc/220Vac | Relays de 24 Vdc | 2 |
| Bornera Molex de 2 ,4, 6 y 8 | Bornera Molex en conjunto | 20 |
| Bornera de 15 amperios de 8 entradas y salidas | Bornera | 1 |
| Bornera de 2 amperios de 16 entradas y salidas | Bornera | 2 |
| Relay de 12Vdc | Relays de 12 Vdc para la tarjeta impresa | 3 |
| Diodo D1N4148 | Diodo rectificador | 1 |
| Diodo D1N4007 | Diodo rectificador | 3 |
| Diodo D1N4004 | Diodo rectificador | 2 |

**TABLA N° 3.2** Equipamiento para el hardware control PICel cerebro parte 2

| EQUIPAMIENTO | DESCRIPCION | CANTIDAD |
|---|---|---|
| Regulador 7805 | Limita la salida cuya voltaje es 5 V | 1 |
| Regulador 7812 | Limita la salida cuya voltaje es 12 V | 1 |
| Regulador 7912 | Limita la salida cuya voltaje es -12 V | 1 |
| Cristal XT de 4Mhz | Cristal para el oscilador | 1 |
| Resistores de 220 Ω ¼ watts | Por unidad los resistores | 8 |
| Resistores de 330 Ω ¼ watts | Por unidad los resistores | 1 |
| Resistores de 2.2 K Ω ¼ watts | Por unidad los resistores | 3 |
| Resistores de 100 Ω ¼ watts | Por unidad los resistores | 5 |
| Resistores de 6.8K Ω ¼ watts | Por unidad los resistores | 1 |
| Resistores de 1K Ω ¼ watts | Por unidad los resistores | 1 |
| Resistores de 10K Ω ¼ watts | Por unidad los resistores | 4 |
| Conector db9 macho | Por unidad | 1 |

El equipamiento para el circuito de control de relevadores para las pruebas locales son las siguientes:

**TABLA N° 3.3** Equipamiento para el hardware control de relevadores.

| EQUIPAMIENTO | DESCRIPCION | CANTIDAD |
|---|---|---|
| Botonera N.C | Botón parar el motor | 1 |
| Botonera N.A | Botón arrancar el motor | 1 |
| Caja IDE 194x144x100 mm | Cubierta del circuito de control de reveladores prueba | 1 |
| Switch on off | Apagar y Prender el motor dc en forma directa | 1 |
| Leds ultrabrillantes | Indica cierta variable en operación | 8 |
| Plancha acrílico de un 3mm de ancho | Proteger la cara del hardware y una base para sus conexiones de cables | 1/4 |
| Tornillos 1/8 de ¼,1/2,1/8 | Para ajustar y ensamblar en la caja | 25 |
| Relays de 24 VDC y la base de conexión del relay | Relays de 24 VDC de 30 amperios/40 amperios | 4 |
| Ventilador DC | Ventilador de 24 VDC | 1 |
| Terminales de 2,4,6 | Son terminales aéreos | 2 |

El equipamiento para las herramientas de construcción son los siguientes:

**TABLA N° 3.4** Equipamiento de herramientas para la construcción en el ensamblado del proyecto.

| EQUIPAMIENTO | DESCRIPCION | CANTIDAD |
|---|---|---|
| Cintillos de 15 cm | Por unidad | 20 |
| Silicona de 220ml | Pegamento | 1 |
| Manga retráctil | En metro la maga retráctil | 3 |
| Cable ethernet Sctp | En metro cable ethernet | 3 |
| Cable automotriz | En metro cable de 10 amperios | 8 |
| Separadores | Unos 5 cm de largo | 8 |
| Adaptador serial a usb | Por unidad | 1 |
| Alicate de corte | Por unidad | 1 |
| Alicate de punta | Por unidad | 1 |
| Alicate pela cable | Por unidad | 1 |
| Grimping | Por unidad | 1 |
| Cuchilla | Por unidad | 1 |
| Desarmador plano | Por unidad | 1 |
| Desarmador estrella | Por unidad | 1 |
| Perillero plano | Por unidad | 1 |
| Perillero estrella | Por unidad | 1 |
| Multímetro digital | Por unidad | 1 |
| Minitaladro | Es un minitaladro para tarjeta electrónica | 1 |
| Taladro | Taladro de 250 Wattios | 1 |
| Cautil | Por unidad cautil de 25 wattios | 1 |
| Soporte de cautil y con lupa | Por unidad | 1 |
| Barnizado para tarjeta electrónica | Por unidad | 1 |
| Alicate de punta | Por unidad | 1 |
| Limpiador de contactos | Por unidad | 1 |
| Pintura | Spray plomo y esmalte marrón | 1 |

# CAPÍTULO IV
# ANÁLISIS Y PRESENTACIÓN DE RESULTADOS

## 4.1 Descripción y resultados obtenidos

A continuación se instalará en la pc el programa ejecutable programado en visual .net 2008, y se harán las pruebas correspondientes para el hardware construido de un ventilador en dc y las pruebas reales con el ventilador minero:

En las pruebas del hardware del circuito de control PIC y del circuito de reveladores para el ventilador 24Vdc, se describe; primero conectas mediante los conectores aéreos la caja que contiene el circuito de control PIC con la caja que contiene el circuito de relevadores, habilitar el punto de red y energía de alimentación de 220 voltios ac, luego se conecta con el switch mediante el cable de red UTP, y la alimentación de 220 voltios ac, se configura y verifica la continuidad en la red, asegurase que éste pertenezca a la ip de red, es decir se verifica si existe la ip en el hardware donde se encuentra el tibbo y la ip de la pc remota, se instaló el software del servidor del hardware tibbo, se asignó el puerto serial virtual en el com10 mediante el programa configurable VSP (Virtual Serial Port) del tibbo y, por último se instaló el ejecutable programado en c#, se abre el programa ejecutable, y se rellena los campos como se indica a continuación:

| | |
|---|---|
| Puerto | COM10 |
| Bits por segundo | 9600 |
| Bits de datos | 8 |
| Paridad | None |
| Bits de Parada | One |

Se aprecia en la siguiente ventana (ver figura 4.1) el sistema de control programa, que contiene botones arrancar, parar y estado; ya que estos están asociados con el comando, si se acciona unos de estos botones asociados al comando para que ejecute la tarea que se encomendó desde la pantalla del programa de la pc. A continuación se muestran los resultados obtenidos:

- Si arrancamos el motor esto quiere decir si apretamos una sola vez el botón "arrancar" desde el sistema de control en la pc, y se visualiza que en la pantalla el sistema de control obtienes "Encendido" y además se ve el mensaje que ha transmitido esta instrucción, para este caso se visualizara en la pantalla "BBB", esto indica que ya sido ejecutado la tarea encomendada, que es encender el motor como se muestra en la figura 4.1 y figura 4.2, y si hubiéramos habilitado salida Hex, hubiera mostrado en la pantalla 424242.

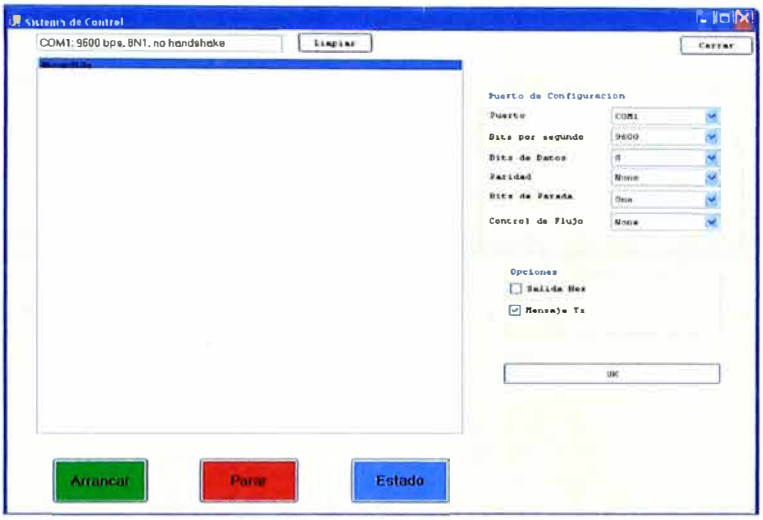


**Figura 4.1** Ventana para rellenar los datos en los campo requeridos y arranque del motor

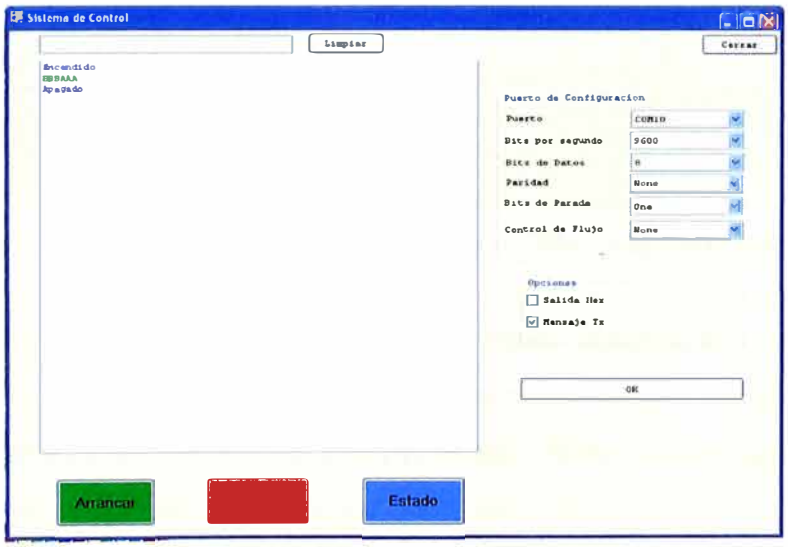


**Figura 4.2** Pruebas del software el apagado del motor

- Cuando paramos el motor del ventilador, se aprieta el botón "parar" que es de color rojo, por lo tanto, en forma inmediata desactiva el funcionamiento del motor, entonces en un tiempo determinado se visualiza en el programa como se describe la figura 4.2, obteniendo el mensaje "apagado", y además se visualizará en la pantalla "AAA", que está confirmando que ha sido ejecutado esta tarea.

- Cuando apretamos el botón "estado" que es de color celeste como se aprecia en la figura 4.3 (este botón nos da la indicación que el motor está en enable o está en disable), en la figura 4.3 la pantalla del programa se visualiza la siguiente cadena EY<STX>Y<STX>Y<STX>Y<STX>Y<STX>Y<STX>Y<STX>Y<STX> (enable) lo que indica que está habilitado o en funcionamiento el motor.

Pero si en caso se visualiza la cadena siguiente DY<STX>Y<STX>Y<STX>Y<STX>Y<STX>Y<STX>Y<STX>Y<STX> (disable) indicaría que está desabilitado o no está en funcionamiento el motor. Por ello, el botón "estado" indica el cómo está en su actividad del motor, es decir si es enable (realizable), que está operando el motor y si indica disable (desactivar), no está activo el motor.

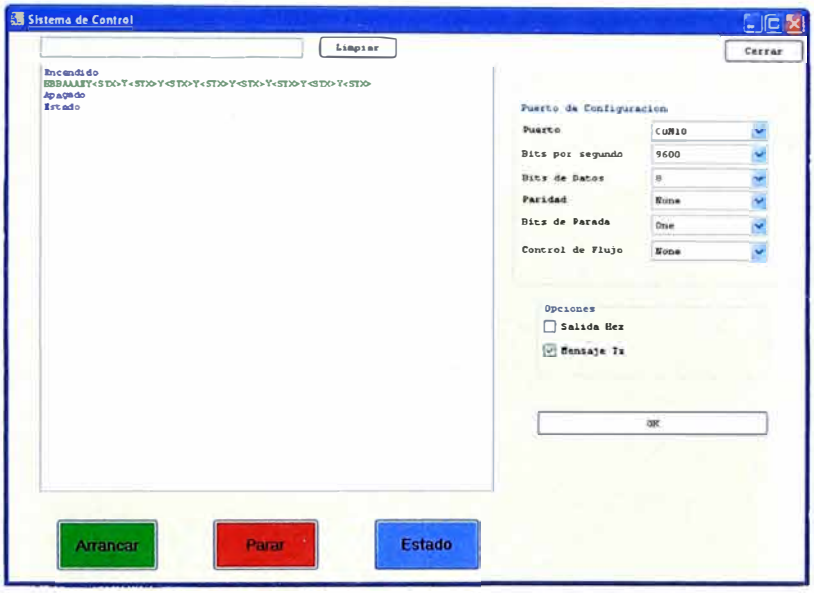


**Figura 4.3** Pruebas de software de verificación de estado

Se hicieron pruebas al software y con el hardware construido, que es un circuito de control de relevadores para el control on/off para un motor dc y el circuito de control PIC, se hicieron 5 muestras y ha funcionado a la perfección, como indican las figura 4.1, figura 4.2, figura 4.3, pero si apretamos en forma continua y rápida el botón arrancar y/o botón parar, llega un límite el consumo de memoria del procesador del PIC (se satura la memoria) y hace que ya no obedezca la instrucciones solicitadas mediante un clic al botón al que desea consultar o accionar la tarea.

También se hicieron pruebas con el ventilador de 250 HP desde un tablero de control que se adhirió con el prototipo construido (circuito de control pic) y diseñado este sistema con anterioridad explicado, verificar en el capítulo 1.2 y la figura 4.5. A continuación se describe para la realización de pruebas reales con el ventilador minero:

primero se reconoció cuál ventilador minero se va usar en las pruebas, este ventilador se encontraba en el nivel -670 del túnel San Carlos de la Minera Porvenir, luego se acordó que se habilitará un punto de red y la ubicación del tablero de alimentación de 220 voltios ac, después se instaló los elementos de red, switch, cableado de red, cableado de energía eléctrica, el hardware construido (circuito de control pic), verificación de continuidad en la red es decir se verifico si existe la ip en el hardware del tibbo, además la pc remota si esa ip pertenece a la red, luego se instaló el software del servidor del hardware tibbo, y se asignó configurando el puerto serial virtual que sería el com10, por último se instaló el ejecutable programado en c#, abrir el programa ejecutable. La instalación se quedó por lo menos un mes en la unidad Minera Porvenir en Cerro de Pasco, durante este tiempo, no hubo corrosión con la tarjeta, ni otra opción de deterioro, durante esos días hicieron varias pruebas desde el software, el arrancado y apagado del ventilador eran satisfactorios, no hubo inconveniente, pero si hubo observaciones para un mejoramiento en la programación.

**Figura 4.4** Pruebas a controlar arrancado y apagado de un ventilador de 250 HP [3]

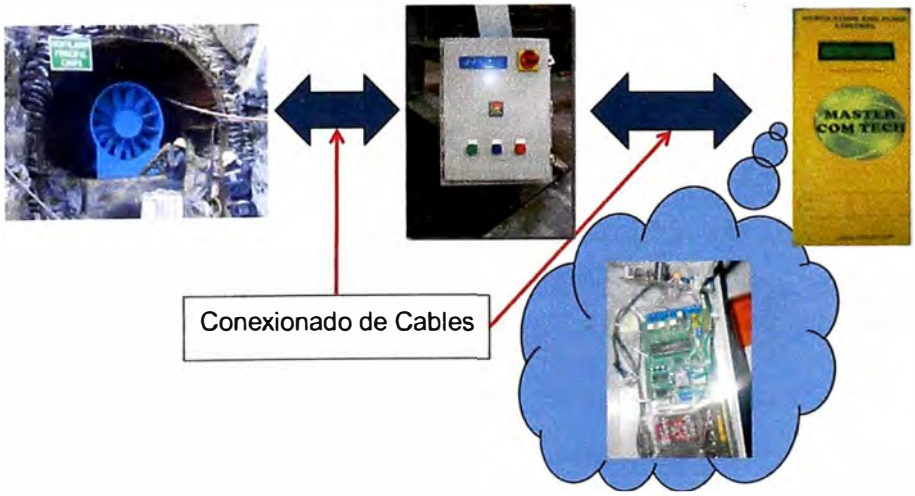


**Figura 4.5** Diagrama que se acopla el circuito de control PIC hacia el tablero de control del ventilador minero

## 4.3. Presupuesto y tiempo de ejecución

El costo de la implementación del hardware y software se considera lo siguiente:

**TABLA N° 4.1** Presupuesto de costo de la implementación del proyecto hardware

| Descripción | Cantidad/unidad | Costo Total |
|---|---|---|
| Diseño de la tarjeta electrónica de control PIC en fibra de vidrio | 1 | S/. 90.00 |
| Diseño de la tarjeta electrónica status de control y fuente de alimentación en dc | 2 | S/.40.00 |
| Diseño de un circuito de control para el motor dc | 1 | S/.120.00 |
| Componentes electrónicos | ---- | S/. 80.00 |
| Herramientas | ---- | S/. 650.00 |
| Servidor Serial | 1 | S/. 540.00 |
| Cajas IDEs | 2 | S/. 80.00 |
| Otros | ---- | S/.300.00 |

El costo total resulta la suma de los costos anteriores S/. 1800.00, faltaría recalcar la licencia el software .net cuyo costo asciende a S/. 400 a S/. 600 adicionales, por lo tanto el costo total sería de S/. 2300 aproximadamente; además esto puede bajar considerablemente si es que este hardware se fabrica en grandes cantidades, tener en cuenta, que hay 2 hardware separados que es para la demostración prueba local, y lo han considerado en el presupuesto (ver TABLA N° 4.1) pero para controlar este proyecto se necesita solamente el hardware de control PIC. Además para el servidor serial si esto se hubiera diseñado obtendría un menor costo.

Se implementó este diseño de hardware y software aproximadamente unos 6 meses, en instalar los equipos de red y el cableado, y configuración de los parámetros duró unos 3 días y por consiguiente se realizaron las pruebas reales por un mes el control on/off en la Minera Porvenir Milpo para ventiladores mineros de 250 hp. Hay observaciones pero no inconvenientes en su funcionamiento, no hubo corrosión en el equipo, ni deformaciones en los integrados que contiene ese hardware implementado.

A continuación, una de las propuestas en la instalación de los equipos para este control de lazo abierto y el costo económico estimado de este proyecto en la Minera Porvenir, como indica la siguiente figura hay 9 hardware que va controlar el on/off a los 9 motores de bomba de agua, mediante un sistema de equipos de red, hacer la

automatización que se apaguen en forma automática y gradual para las bombas a ciertas horas donde la tarifa eléctrica esté en su mayor costo debido a la fuerte demanda obteniendo un pico a esa hora y después arranque el motor cuando la tarifica eléctrica se encuentre en su costo normal o habitual, y está automatización estimada ahorraría al mes unos $5640, para este caso específico.

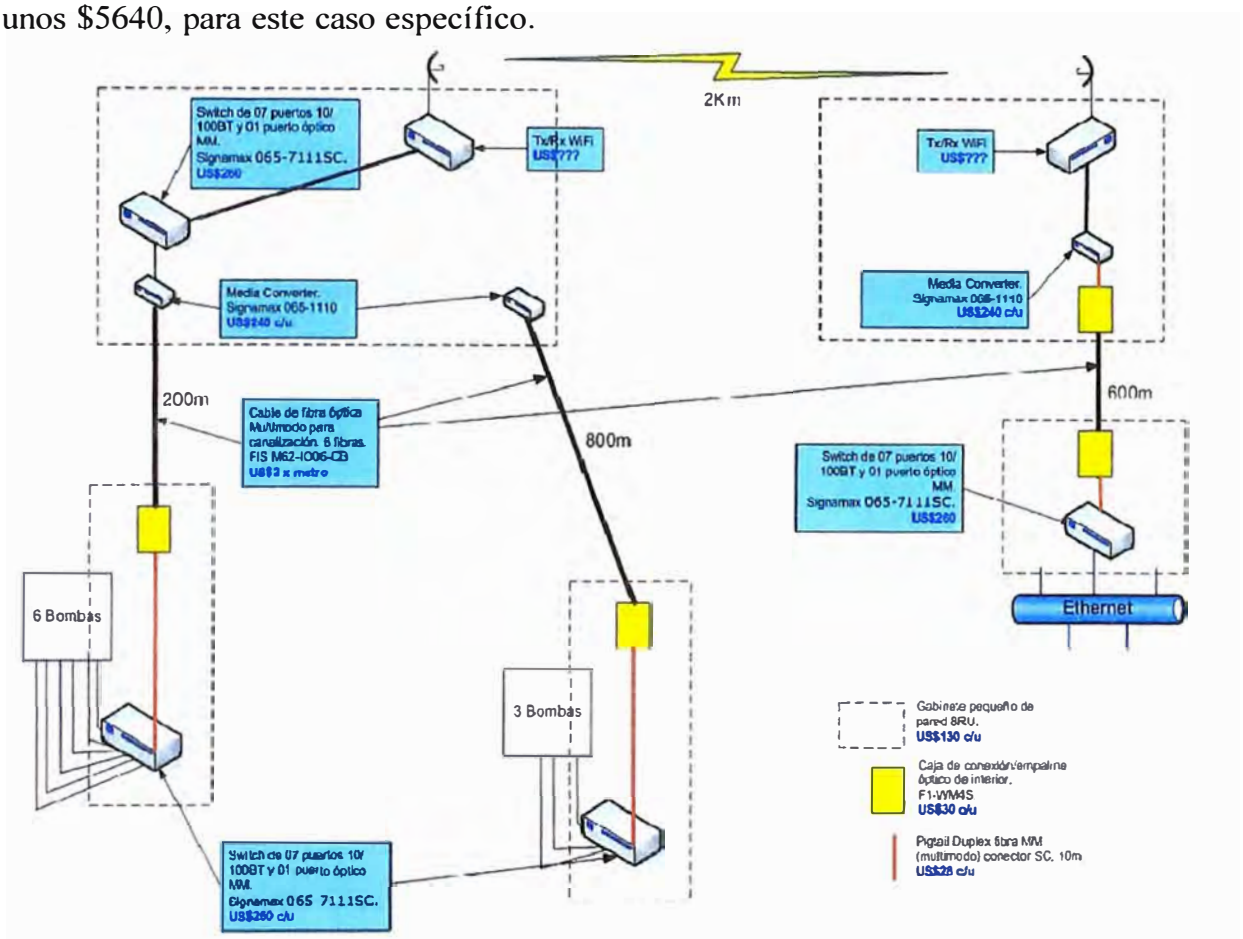


**Figura 4.5** Diagrama y propuesta de un proyecto óptico de un control a distancia para 9 bombas de agua para la Minera Porvenir

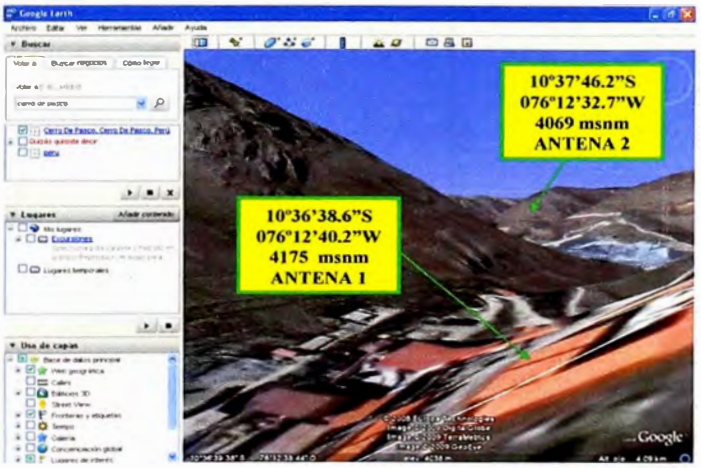


**Figura 4.6** Ubicación de la colocación de antenas del proyecto óptico propuesto para la Minera Porvenir

Este esquema propuesto en la toma de datos se verificó con el google earth obteniendo medidas aproximadamente, y ubicaciones donde se colocaría estos elementos de red, colocación de antenas (ver figura 4.6), instalación de pozo a tierra, el cableado tanto para red y energía eléctrica, colocación de las baterías y simulación aproximada de las antenas tipo grilla de frecuencia de 2.4GHz.

**TABLA N° 4.2** Presupuesto de costo del proyecto control on/off a distancia para las 9 bombas de agua a la Minera Porvenir

| Descripción de los componentes | Cantidad/unidad | Costo Total |
|---|---|---|
| Access Point EDIMAX EW- 7209APG, 18dBm | 2 | a |
| Cables Pigtail Hyperlink SMA - N-macho | 2 | b |
| Antenas Hyperlink 15dBi HG2415G | 2 | c |
| Cajas de Pase Solera REF. 686 | 2 | d |
| media converter de ethernet a Fibra | 4 | $960 |
| Switch de 7 puertos y 1 puerto óptico | 3 | $600 |
| 1 kit que abarca 9Hardwares de control a distancia cubierta de fibra de vidrio y un software | 1 | $5600 |
| Cable de Fibra Óptica de 6 fibras por metro | 1600 | $3200 |
| Cable de energía eléctrica por metro | 1000 | $600 |
| Gabinete pequeño de pared BRU | 5 | $650 |
| Instalación pozo a tierra | 1 | $600 |
| | Total | $12559 |

Dónde, a+b+c+d = $349 dólares (Kit P2P de 5 km).

De acuerdo con este cuadro, el costo estimado estaría alrededor de $12559; a esto equivale un gasto de consumo de energía eléctrica durante 3 meses en ese pico de demanda aproximadamente, por lo cual en el 1er año se ahorraría $45000 y el resto de los años se ahorraría $60000.

# CONCLUSIONES Y RECOMENDACIONES

Tras analizar los resultados obtenidos de las pruebas local y las pruebas en el campo en la unidad Minera se llega a las siguientes conclusiones:

1. Se demuestra con este trabajo propuesto e implementado en controlar desde una PC en forma remota el control arranque y apagado del ventilador minero mediante TCP/IP, sin alterar el sistema de control arranque y parada de un motor.
2. Se demuestra que el costo es menor con estas herramientas utilizadas para este presente trabajo propuesto que abarca los microcontroladores PIC, diseños de circuitos electrónicos, servidor serial y la programación visual .net 2008.
3. Para evitar saturación en la memoria del microcontrolador PIC debido a las manipulaciones muy rápidas (seguidas) de los botones de interfaz de usuario en visual .net 2008, se tiene que modificar en la programación c# agregando de la siguiente manera: durante cada instrucción ejecutado y además validado por el microcontrolador PIC la ejecución de la instrucción, inmediatamente empieza a sincronizar 7 segundos para no obedecer las instrucciones ejecutadas mediante los botones del sistema de control (interfaz de usuario).
4. Adicionar en el prototipo un circuito de reseteo de energía eléctrica de 220 Vac a distancia, sin ir a buscar el hardware para desconectar la energía eléctrica, debido cuando se presenta congelamiento en el programa de control o saturaciones de memoria en el microcontrolador PIC.
5. Se observa en el diseño del circuito electrónico de la fuente de energía de 3 salidas, para reducir espacio se debe colocar un solo transformador de 220 vac a 24Vdc cuya potencia sea de 4 amperios ya que en este presente trabajo están colocadas 2 transformadores de 24Vdc de 2 amperios cada uno, y además conseguir el regulador de 12 voltios que soporte una potencia desde 1 amperio hasta 3 amperios (por ejemplo el 78T12), ya que el regulador LM7812 se calienta demasiado rápido a pesar que está puesto con un disipador.

6. Para no estar continuamente ingresando al interior de la mina para la actualización del programa en el PIC, el prototipo (hardware electrónico construido para el arranque y apagado del motor), se debe adicionar y, incluir un circuito electrónico que haga la función de quemado o grabación del programa en el PIC en forma remota.
7. Continuamente se tiene que mejorar la programación en c# y mejorar el prototipo electrónico que sea más genérico, en función a la necesidad del usuario para el control de los motores de aplicaciones industriales, por ejemplo, que aparezcan la temperatura del motor, el reloj (Orómetro), sacar reportes, monitoreo, entre otros.
8. Para mejorar el diseño de la tarjeta electrónica del control PIC y reducir espacio se debe usar microcontrolador PIC con módulos ethernet de 16 o 32 bits de fabricación Microchip o Motorola por ejemplo el PIC MC9S12NE64 y además adicionar al proyecto del sistema de control un servidor de consulta de base de datos, que almacenará los eventos y parámetros, para poder generar reportes estadísticos necesarios.
9. El interfaz de usuario para reducir el costo de licencia se puede usar el programa de programación JAVA, ya que tiene la similitud en la programación C Sharp (C#) de visual .net 2008.

# ANEXO A
# LISTADO DE PROGRAMAS

## LISTADO DE PROGRAMAS

A continuación se presentan el código de programación en el PIC 16F877A y el código de visual .net C#, respectivamente:

### A.1. Programa en Código PIC 16F877A

```
#INCLUDE   "C:\Archivos de programa\PICC\Devices\16F877A.h"
#FUSES     NOWDT,NOLVP,XT
#USE  DELAY (CLOCK=4000000)
#USE  RS232 (baud=9600, RCV=PIN_C7, XMIT=PIN_C6)
void main(void)
{
    char A;
    char B;
    char C;
    int x;
    int y;
    PUTS("OK");
BUCLE:              x=getc();
 if (x == 'A')
{
  output_high(PIN_B7);
  delay_ms(7000);
  output_low(PIN_B7);
  puts("AAA");
}
if (x == 'B')
{
  output_high(PIN_B0);
  delay_ms(7000);
  output_low(PIN_B0);
   puts("BBB");
}
 if (x == 'C')
```

```
 {
  y=input(PIN_C1)
  if (y==0)
  {
    puts("E&Y&Y&Y&Y&Y&Y&Y");
  }
  else
  {
    puts("D&Y&Y&Y&Y&Y&Y&Y");
  }
}
 if (x == 'D')
{
  output_high(PIN_B1);
  delay_ms(7000);
  output_low(PIN_B1);
  puts("DDD");
}
  goto BUCLE;
}
```

**A.2. Programa en Código Visual. Net C#**

```
using System;
using System.IO;
using System.IO.Ports;
using System.Diagnostics;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
```

```
namespace Termie
{
  public partial class Form1 : Form
  {
/// Class to keep track of string and color for lines in output window.
//Form2 testform = new Form2();
    private class Line
    {
      public string Str;
      public Color ForeColor;
      public Line(string str, Color color)
      {
        Str = str; ForeColor = color;
      }
     };
   ArrayList lines = new ArrayList();
   Font origFont;
   Font monoFont;
   public int Var;
 public Form1()
{InitializeComponent();
  splitContainer1.FixedPanel = FixedPanel.Panel1;
  splitContainer2.FixedPanel = FixedPanel.Panel2;
//AcceptButton = button5; //Send
//CancelButton = button4; //Close
  outputList_Initialize();
  Settings.Read();
  TopMost = Settings.Option.StayOnTop;
// let form use multiple fonts
  origFont = Font;
  FontFamily ff = new FontFamily("Courier New");
  monoFont = new Font(ff, 8, FontStyle.Regular);
 Font = Settings.Option.MonoFont ? monoFont : origFont;
```

```
 CommPort com = CommPort.Instance;
 com.StatusChanged += OnStatusChanged;
 com.DataReceived += OnDataReceived;
 com.Open();
 ////////// Aqui empieza para la configuracion del puerto
if (portList.Length > 0)
comboBox1.SelectedIndex = found;
Int32[] baudRates = {100,300,600,1200,2400,4800,9600,14400,19200,38400, 56000,
57600,115200,128000,256000,0 };
found = 0;
for (int i = 0; baudRates[i] != 0; ++i)
 {
  comboBox2.Items.Add(baudRates[i].ToString());
  if (baudRates[i] == Settings.Port.BaudRate)
  found = i;
  }
comboBox2.SelectedIndex = found;
comboBox3.Items.Add("5");
comboBox3.Items.Add("6");
comboBox3.Items.Add("7");
comboBox3.Items.Add("8");
comboBox3.SelectedIndex = Settings.Port.DataBits - 5;
foreach (string s in Enum.GetNames(typeof(Parity)))
 {
  comboBox4.Items.Add(s);
  }
   comboBox4.SelectedIndex = (int)Settings.Port.Parity;
foreach (string s in Enum.GetNames(typeof(StopBits)))
 {
  comboBox5.Items.Add(s);
  }
comboBox5.SelectedIndex = (int)Settings.Port.StopBits;
foreach (string s in Enum.GetNames(typeof(Handshake)))
```

```
 {
 comboBox6.Items.Add(s);
  }
comboBox6.SelectedIndex = (int)Settings.Port.Handshake;
checkBox1.Checked = Settings.Option.HexOutput;
checkBox3.Checked = Settings.Option.LocalEcho;
textBox1.Text = Settings.Option.LogFileName;      ////////fin de la configuraciondel puerto
}
// shutdown the worker thread when the form closes
protected override void OnClosed(EventArgs e)
{
CommPort com = CommPort.Instance
com.Close();
base.OnClosed(e);
}
// output string to log file
///salida de la cadena/nombre -- - -string to output
public void logFile_writeLine(string stringOut)
{
if (Settings.Option.LogFileName != "")
 {
 Stream myStream = File.Open(Settings.Option.LogFileName,
 FileMode.Append, FileAccess.Write, FileShare.Read);
 if (myStream != null)
   {
    StreamWriter myWriter = new StreamWriter(myStream, Encoding.UTF8);
    myWriter.WriteLine(stringOut);
    myWriter.Close();
    }
 }
}
#region Output window
string filterString = "";
```

```
bool scrolling = true;
Color receivedColor = Color.Green;
Color sentColor = Color.Blue;
/// context menu for the output window
ContextMenu popUpMenu;
///  comprobar si coincide con filtro de cadenas
bool outputList_ApplyFilter(String s)
{
 if (filterString == "")
  {
   return true;
  }
else if (s == "")
  {
  return false;
  }
else if (Settings.Option.FilterUseCase)
  {
  return (s.IndexOf(filterString) != -1);
  }
 else
 {
 string upperString = s.ToUpper();
 string upperFilter = filterString.ToUpper();
 return (upperString.IndexOf(upperFilter) != -1);
 }
}
/// clear the output window
void outputList_ClearAll()
{
 lines.Clear();
 partialLine = null;
 outputList.Items.Clear();
```

```
}
/// refresh the output window
void outputList_Refresh()
{
outputList.BeginUpdate();
outputList.Items.Clear();
foreach (Line line in lines)
 {
   if (outputList_ApplyFilter(line.Str))
   {
   outputList.Items.Add(line);
   }
 }
outputList.EndUpdate();
outputList_Scroll();
}
/// add a new line to output window
Line outputList_Add(string str, Color color)
{
 Line newLine = new Line(str, color);
 lines.Add(newLine);
 if (outputList_ApplyFilter(newLine.Str))
  {
  outputList.Items.Add(newLine);
  outputList_Scroll();
  }
return newLine;
}
/// Update a line in the output window.
void outputList_Update(Line line)
{
// should we add to output?
  if (outputList_ApplyFilter(line.Str))
```

```
   {
// is the line already displayed?
   bool found = false;
   for (int i = 0; i < outputList.Items.Count; ++i)
     {
      int index = (outputList.Items.Count - 1) - i;
      if (line == outputList.Items[index])
      {
// is item visible?
      int itemsPerPage = (int)(outputList.Height / outputList.ItemHeight);
       if (index >= outputList.TopIndex && index < (outputList.TopIndex + itemsPerPage))
       {
// is there a way to refresh just one line
// without redrawing the entire listbox?
// changing the item value has no effect
       outputList.Refresh();
       }
      found = true;
     break;
      }
     }
if (!found)
 {
// not found, so add it
  outputList.Items.Add(line);
   }
  }
}
/// Initialize the output window
private void outputList_Initialize()
{
// owner draw for listbox so we can add color
outputList.DrawMode = DrawMode.OwnerDrawFixed;
```

```
outputList.DrawItem += new DrawItemEventHandler(outputList_DrawItem);
outputList.ClearSelected();
// build the outputList context menu
popUpMenu = new ContextMenu();
popUpMenu.MenuItems.Add("&Copy", new EventHandler(outputList_Copy));
popUpMenu.MenuItems[0].Visible = true;
popUpMenu.MenuItems[0].Enabled = false;
popUpMenu.MenuItems[0].Shortcut = Shortcut.CtrlC;
popUpMenu.MenuItems[0].ShowShortcut = true;
popUpMenu.MenuItems.Add("Copy All", new EventHandler(outputList_CopyAll));
popUpMenu.MenuItems[1].Visible = true;
popUpMenu.MenuItems.Add("Select &All", new EventHandler(outputList_SelectAll));
popUpMenu.MenuItems[2].Visible = true;
popUpMenu.MenuItems[2].Shortcut = Shortcut.CtrlA;
popUpMenu.MenuItems[2].ShowShortcut = true;
popUpMenu.MenuItems.Add("Clear Selected", new EventHandler
(outputList_ClearSelected) );
popUpMenu.MenuItems[3].Visible = true;
outputList.ContextMenu = popUpMenu;
}
/// draw item with color in output window
void outputList_DrawItem(object sender, DrawItemEventArgs e)
{
 e.DrawBackground();
  if (e.Index >= 0 && e.Index < outputList.Items.Count)
  {
   Line line = (Line)outputList.Items[e.Index];
// if selected, make the text color readable
   Color color = line.ForeColor;
   if ((e.State & DrawItemState.Selected) == DrawItemState.Selected)
    {     color = Color.Black;  // make it readable     }
   e.Graphics.DrawString(line.Str, e.Font, new SolidBrush(color),
   e.Bounds, StringFormat.GenericDefault);
```

```
    }
  e.DrawFocusRectangle();
}
/// Scroll to bottom of output window
void outputList_Scroll()
{
  if (scrolling)
  {
    int itemsPerPage = (int)(outputList.Height / outputList.ItemHeight);
    outputList.TopIndex = outputList.Items.Count - itemsPerPage;
  }
}
/// Enable/Disable copy selection in output window
private void outputList_SelectedIndexChanged(object sender, EventArgs e)
{
 popUpMenu.MenuItems[0].Enabled = (outputList.SelectedItems.Count > 0);
}
/// copy selection in output window to clipboard
private void outputList_Copy(object sender, EventArgs e)
{
 int iCount = outputList.SelectedItems.Count;
 if (iCount > 0)
  {
   String[] source = new String[iCount];
   for (int i = 0; i < iCount; ++i)
   {
    source[i] = ((Line)outputList.SelectedItems[i]).Str;
   }
   String dest = String.Join("\r\n", source);
   Clipboard.SetText(dest);
  }
}
/// copy all lines in output window
```

```
private void outputList_CopyAll(object sender, EventArgs e)
{
 int iCount = outputList.Items.Count;
 if (iCount > 0)
  {
   String[] source = new String[iCount];
   for (int i = 0; i < iCount; ++i)
   {
    source[i] = ((Line)outputList.Items[i]).Str;
   }
   String dest = String.Join("\r\n", source);
   Clipboard.SetText(dest);
  }
}
/// select all lines in output window
private void outputList_SelectAll(object sender, EventArgs e)
{
 outputList.BeginUpdate();
 for (int i = 0; i < outputList.Items.Count; ++i)
 {
   outputList.SetSelected(i, true);
 }
   outputList.EndUpdate();
}
/// clear selected in output window
private void outputList_ClearSelected(object sender, EventArgs e)
{
 outputList.ClearSelected();
 outputList.SelectedItem = -1;
}
#endregion
#region Event handling - data received and status changed
/// Prepare a string for output by converting non-printable characters.
```

```
private String PrepareData(String StringIn)
{
// The names of the first 32 characters
 string[] charNames = { "NUL", "SOH", "STX", "ETX", "EOT", "ENQ", "ACK",  "BEL",
"BS", "TAB", "LF", "VT", "FF", "CR", "SO", "SI", "DLE", "DC1", "DC2", "DC3", DC4",
"NAK", "SYN", "ETB", "CAN", "EM", "SUB","ESC", "FS", "GS", "RS", "US", "Space"};
 string StringOut = "";
 foreach (char c in StringIn)
  {
   if (Settings.Option.HexOutput)
   {
    StringOut = StringOut + String.Format("{0:X2} ", (int)c);
    }
   else if (c < 32 && c != 9)
   {
     StringOut = StringOut + "<" + charNames[c] + ">";
//Uglier "Termite" style
//StringOut = StringOut + String.Format("[{0:X2}]", (int)c);
   }
   else
   {
     StringOut = StringOut + c;
   }
 }
 return StringOut;
}
/// Partial line for AddData().
private Line partialLine = null;
    Add data to the output.
private Line AddData(String StringIn)
{
  String StringOut = PrepareData(StringIn);
// if we have a partial line, add to it.
```

```
  if (partialLine != null)
  {
// tack it on
   partialLine.Str = partialLine.Str + StringOut;
   outputList_Update(partialLine);
   return partialLine;
  }
  return outputList_Add(StringOut, receivedColor);
}
// delegate used for Invoke
internal delegate void StringDelegate(string data);
/// Handle data received event from serial port.
public void OnDataReceived(string dataIn)
{
//Handle multi-threading
 if (InvokeRequired)
 {
   Invoke(new StringDelegate(OnDataReceived), new object[] { dataIn });
    return;
  }
// pause scrolling to speed up output of multiple lines
bool saveScrolling = scrolling;
scrolling = false;
// if we detect a line terminator, add line to output
int index;
while (dataIn.Length > 0 &&
((index = dataIn.IndexOf("\r")) != -1 ||
(index = dataIn.IndexOf("\n")) != -1))
 {
  String StringIn = dataIn.Substring(0, index);
  dataIn = dataIn.Remove(0, index + 1);
  logFile_writeLine(AddData(StringIn).Str);
  partialLine = null;    // terminate partial line
```

```
 }
// if we have data remaining, add a partial line
if (dataIn.Length > 0)
{
  partialLine = AddData(dataIn);
}
// restore scrolling
scrolling = saveScrolling;
outputList_Scroll();
}
// Update the connection status
public void OnStatusChanged(string status)
{
//Handle multi-threading
 if (InvokeRequired)
 {
  Invoke(new StringDelegate(OnStatusChanged), new object[] { status });
  return;
  }
textBox1.Text = status;
}
#endregion
#region User interaction
/ / toggle connection status
private void textBox1_Click(object sender, MouseEventArgs e)
{
CommPort com = CommPort.Instance;
if (com.IsOpen)
  {
  com.Close();
  }
 else
 {
```

```
  com.Open();
  }
outputList.Focus();
}
/// Limpiar la ventana de Salida de impresion
private void button2_Click(object sender, EventArgs e)
{
outputList_ClearAll();
}
/// Mostrar la Ventana about
private void button3_Click(object sender, EventArgs e)
{
TopMost = false;
AboutBox about = new AboutBox();
about.ShowDialog();
//TopMost = Settings.●ption.StayOnTop;
}
 // Cerrando la aplicacion
private void button4_Click(object sender, EventArgs e)
{
Close();
}
#endregion
private void btnArrancar_Click(object sender, EventArgs e)
{
CommPort com = CommPort.Instance;
com.Send("B");
if (Setting .Option.LocalEcho)
 {
 outputList_Add("  ncendido" + "\n", sentColor);
 }
}
private void btnParar_Click(object sender, EventArgs e)
```

```
{   CommPort com = CommPort.Instance;
  com.Send("A");
  if (Settings.Option.LocalEcho)
  {
    outputList_Add("Apagado" + "\n", sentColor);
  }
}
private void btnEstado_Click(object sender, EventArgs e)
{
 CommPort com = CommPort.Instance;
 com.Send("C");
 if (Settings.Option.LocalEcho)
  {
  outputList_Add("Estado" + "\n", sentColor);
  }
}
 private void btnOk_Click(object sender, EventArgs e)
{
 Settings.Port.PortName = comboBox1.Text;//
 Settings.Port.BaudRate = Int32.Parse(comboBox2.Text);
 Settings.Port.DataBits = comboBox3.SelectedIndex + 5;
 Settings.Port.Parity = (Parity)comboBox4.SelectedIndex;
 Settings.Port.StopBits = (StopBits)comboBox5.SelectedIndex;
 Settings.Port.Handshake = (Handshake)comboBox6.SelectedIndex;
   ettings.Option.HexOutput = checkBox1.Checked;
 Settings.Option.LocalEcho = checkBox3.Checked;
 Settings.Option.LogFileName = textBox1.Text;
 CommPort com = CommPort.Instance;
 com.Open();
 Settings.Write();
}
  }
}
```

# ANEXO B
# DIAGRAMAS CIRCUITALES

## ESQUEMAS DE CIRCUITOS ELECTRÓNICOS

En el presente anexo se presentan los siguientes esquemas:

- Figura B.1: Circuito de control PIC con 3 señales de salida y con 4 señales de entrada.
- Figura B.2: Circuito para ver el estado de la tarjeta electrónica de control PIC.
- Figura B.3: Circuito para fuente de alimentación dc de 3 salidas 12, -12 y 5 voltios.
- Figura B.4: Circuito para ver el estado de la tarjeta de alimentación dc.
- Figura B.5: Silkscreen top del circuito de control y alimentación dc.
- Figura B.6: Capa bottom del circuito de control y alimentación dc.
- Figura B.7: Capa top del circuito status de la tarjeta de alimentación.
- Figura B.8: Esquema de un circuito de control para un motor dc para su demostración en prueba local el control de arranque y parada.

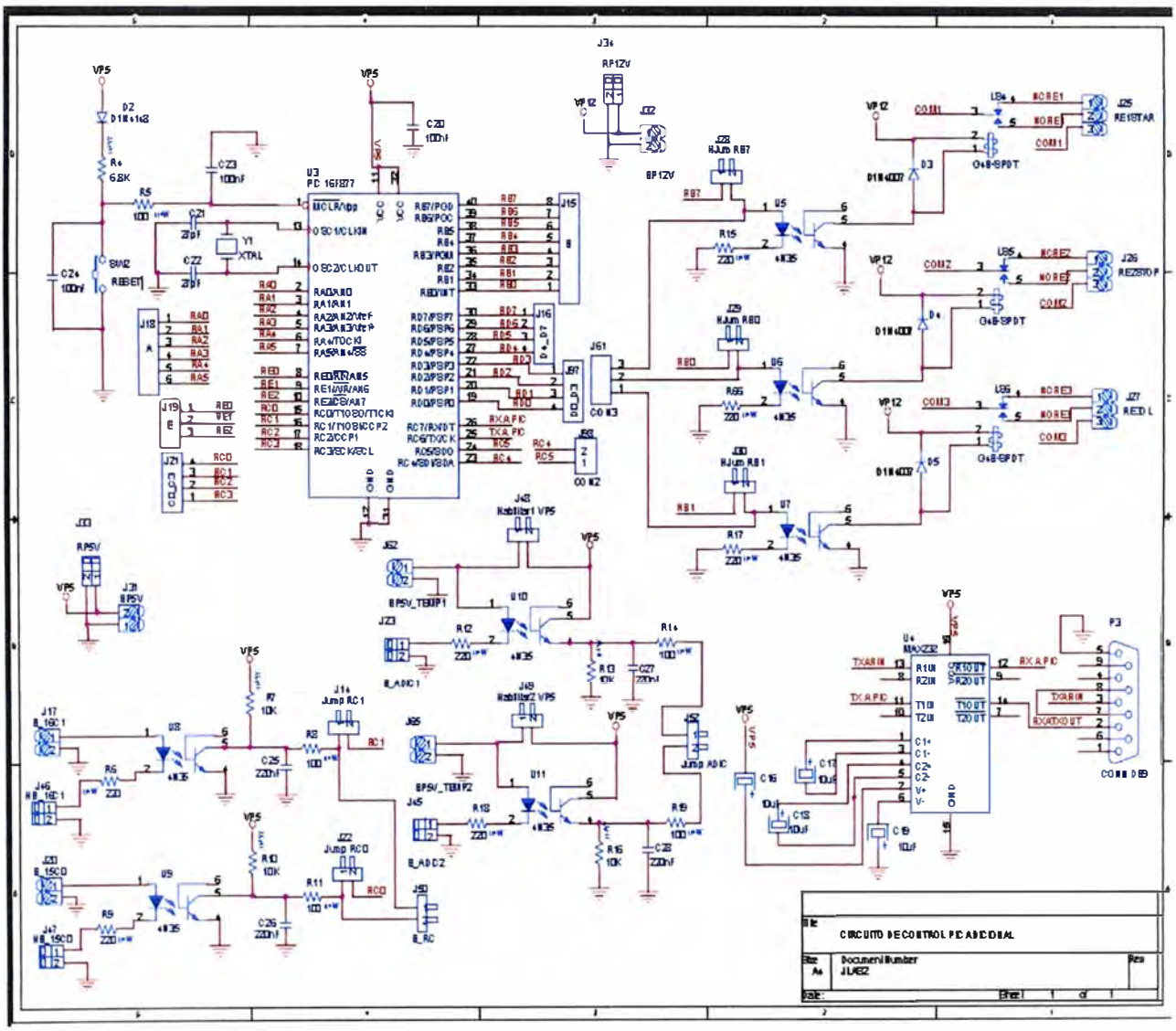


**Figura B.1** Circuito de control PIC con 3 señales de salida y con 4 señales de entrada

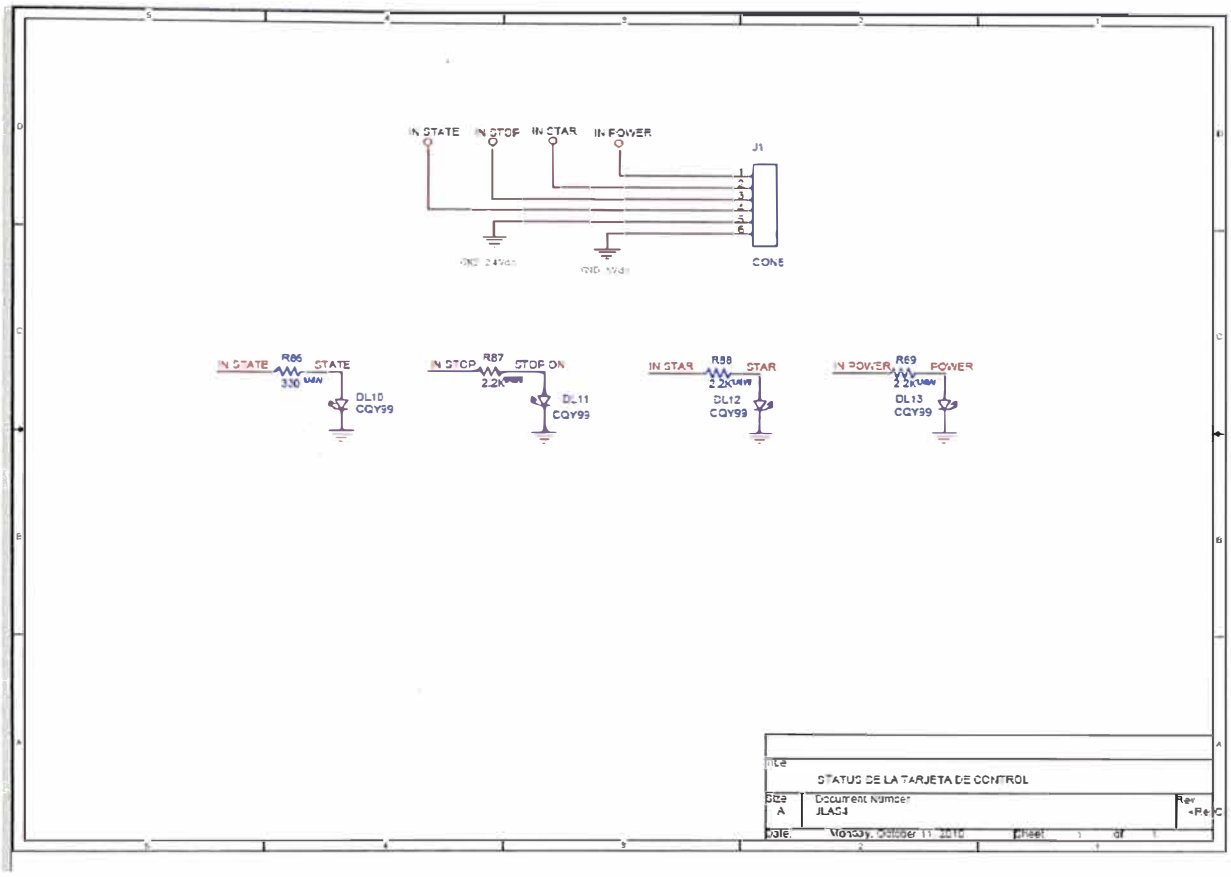


**Figura B.2** Circuito para ver el estado de la tarjeta electrónica de control PIC

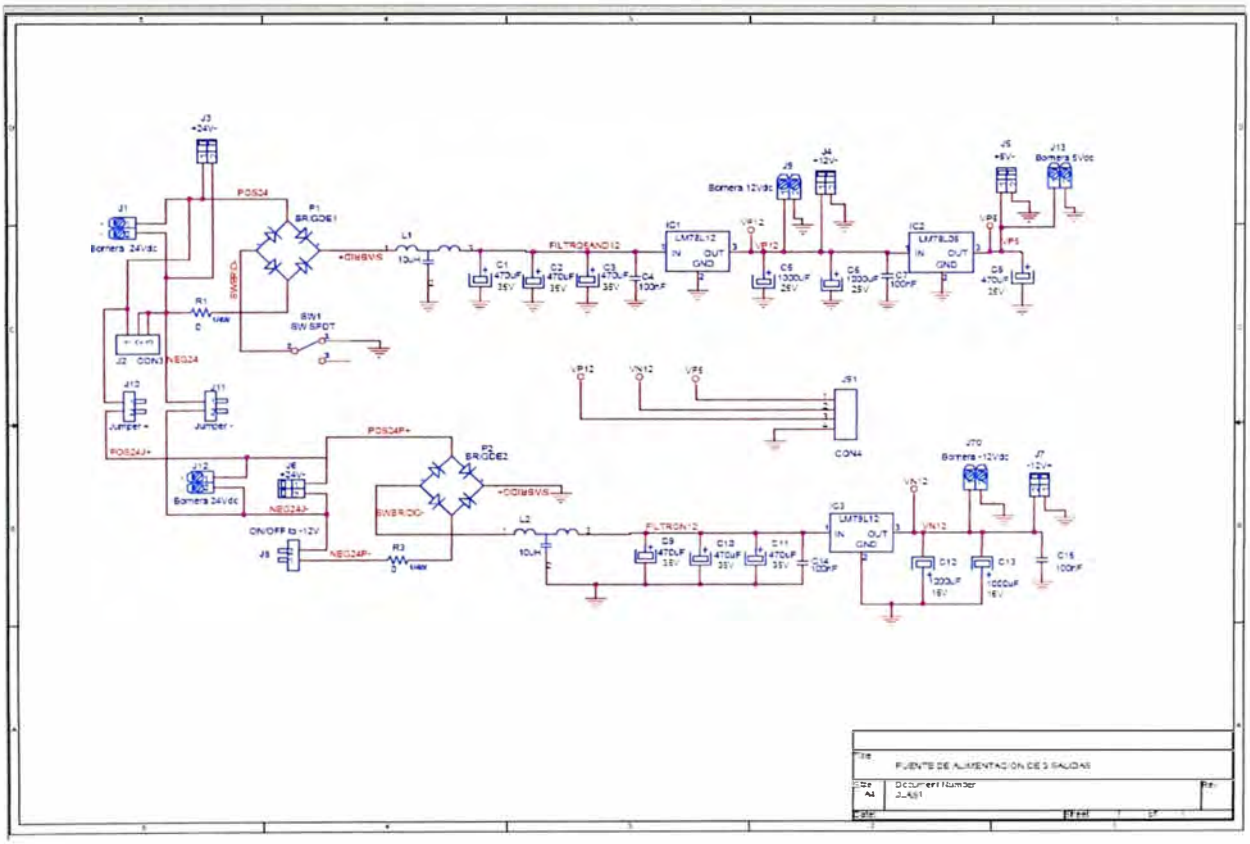


**Figura B.3** Circuito para fuente de alimentación dc de 3 salidas 12,-12 y 5 voltios

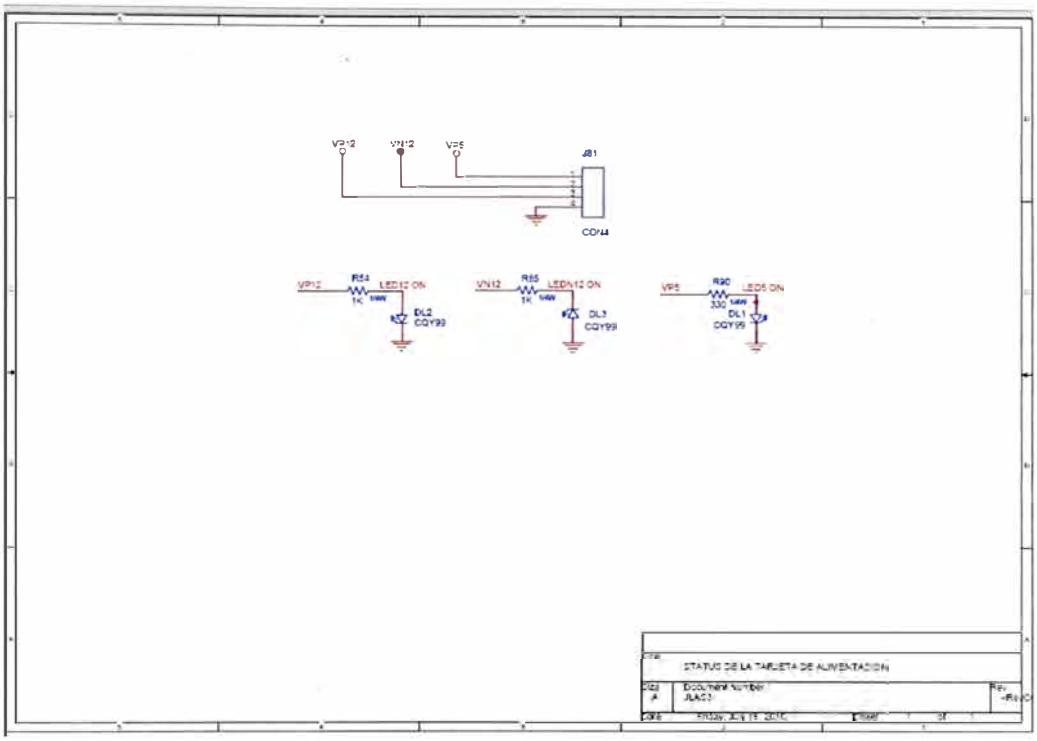


**Figura B.4** Circuito para ver el estado de la tarjeta de alimentación dc

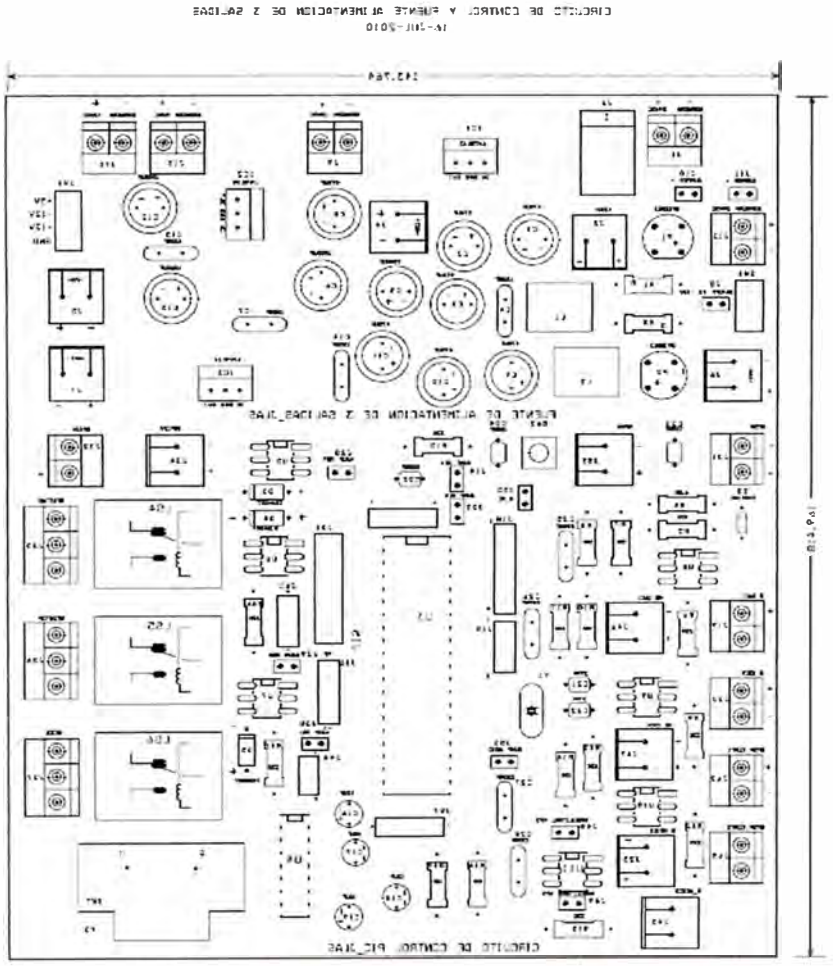

**Figura B.5** Silkscreen top del circuito de control y alimentación dc

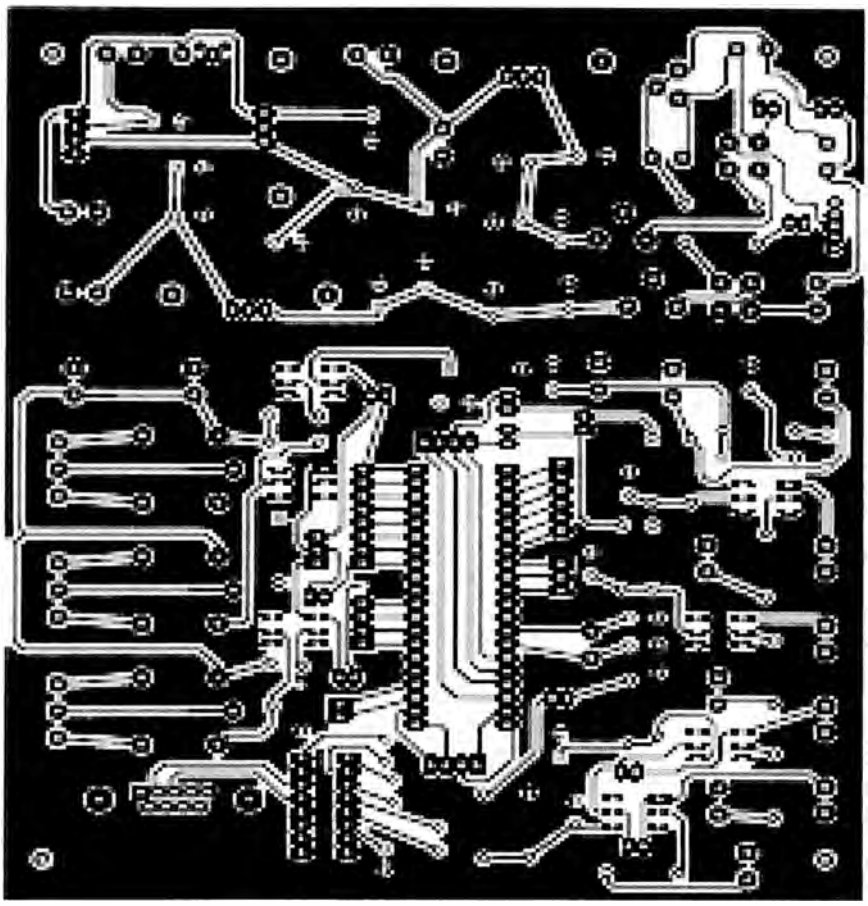

**Figura B.6** Bottom del circuito de control y alimentación dc

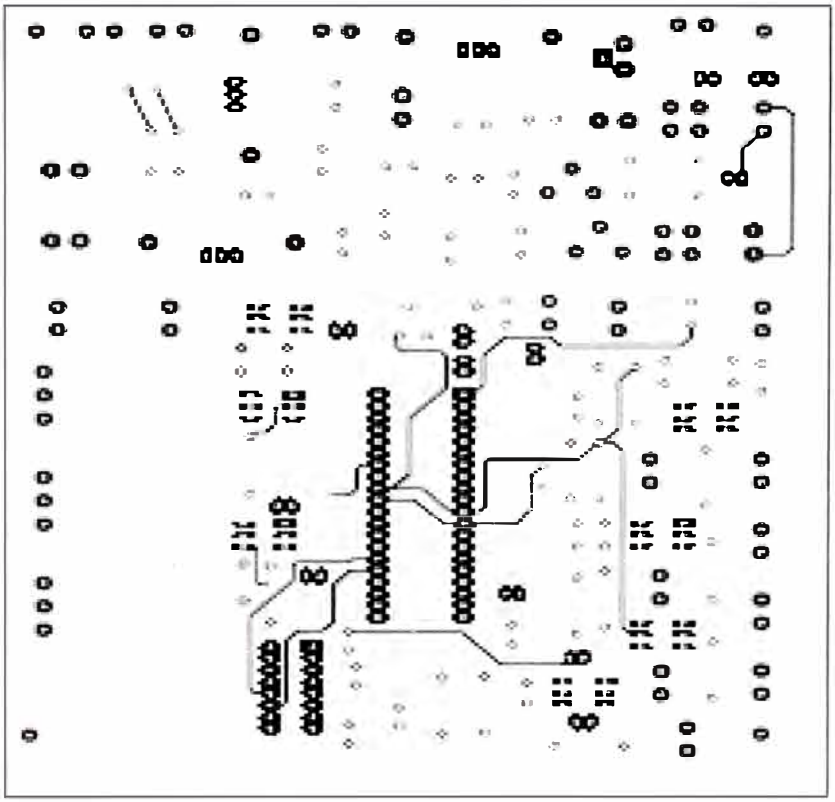

**Figura B.7** Top del circuito de control y alimentación dc

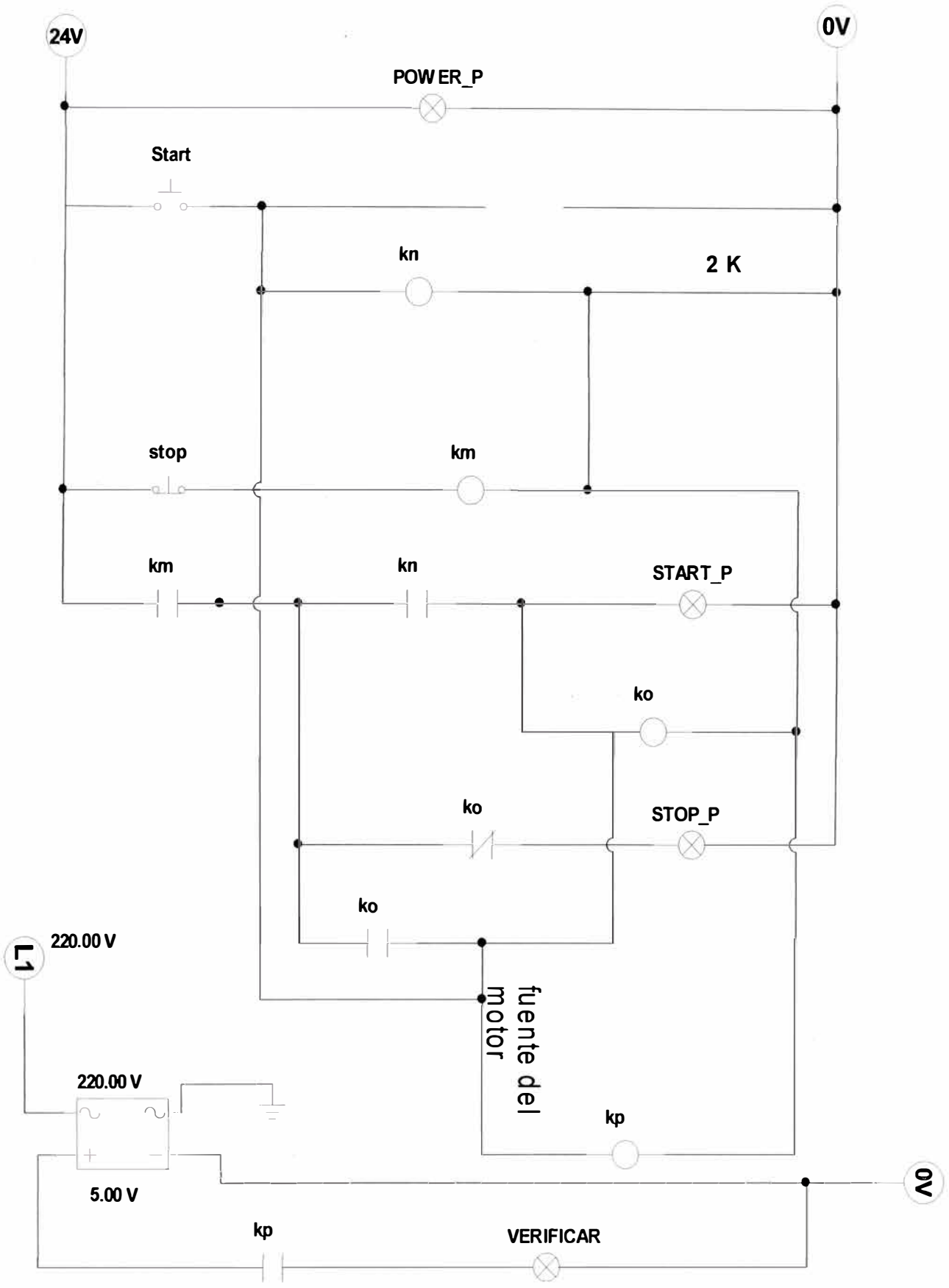


**Figura B.8** Esquema de un circuito de control para un motor dc para su demostración en prueba local el control de arranque y parada

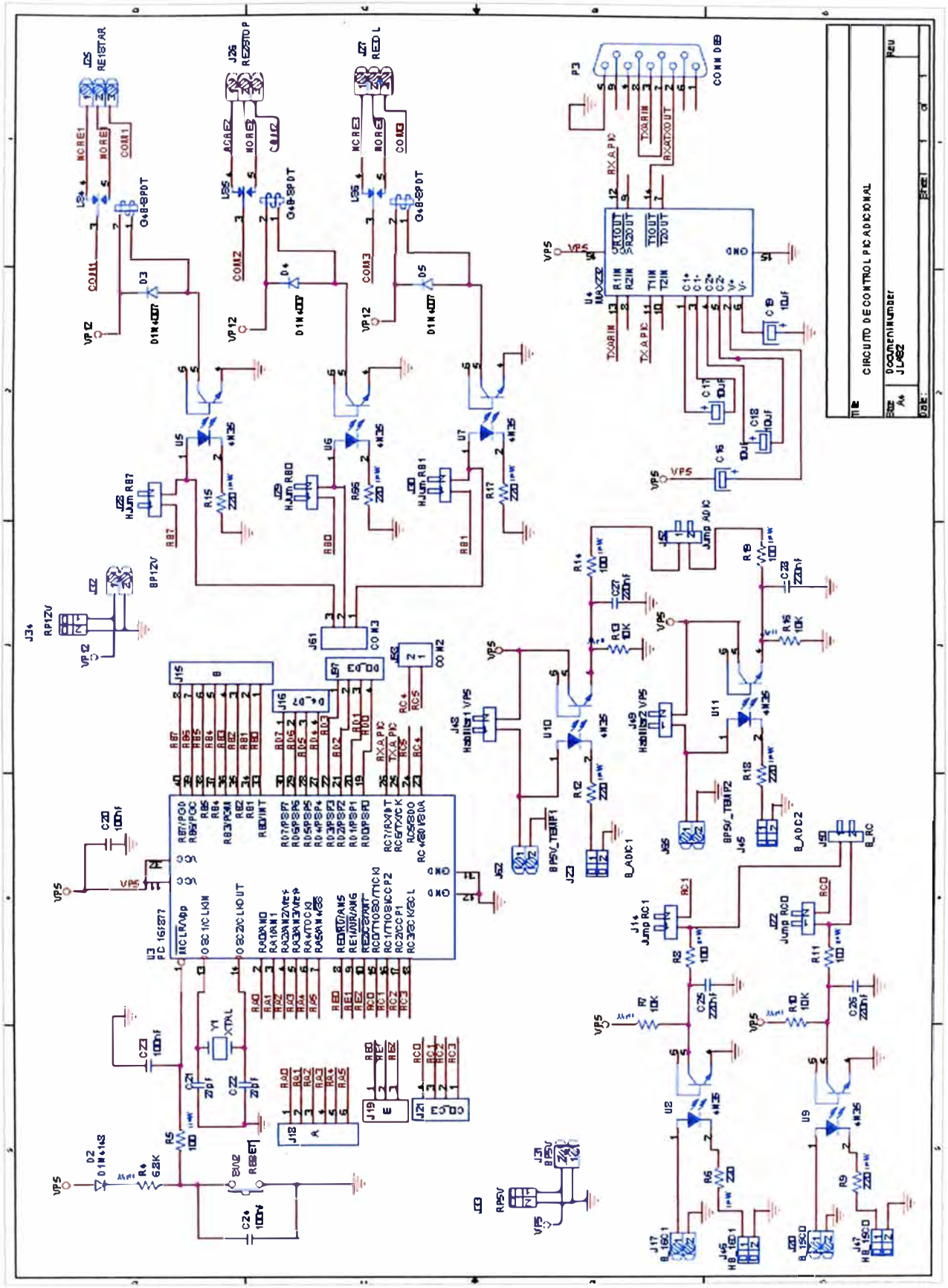


Figura ampliada del circuito de control PIC con 3 señales de salida y con 4 señales de entrada

# BIBLIOGRAFÍA


[1] Enrique Palacios, Fernando Remiro, Lucas J. López. "Microcontrolador PIC16F84 Desarrollo de Proyectos". Alfaomega.

[2] Eduardo García Breijo. "Compilador C CCS y simulador Proteus para Microcontroladores PIC". Alfaomega.

[3] Guía de Ventilación Minas http://www.semageomin.cl/pdf/guias_manuales_formularios/200812GuiaVentilacionMinas.pdf. Página 3.

[4] How to Start and How to Stop: AC/DC Motor Control http://www.pdhengineer.com/courses/e/E-3019.pdf. Páginas 4-15.

[5] Ricardo Salem, Marco Aurélio, Marley Maria. "Evolutionary Electronics Automatic Design of Electronic Circuits and System by Genetic Algorithms". Crc Press.

[6] Circuitos de control y motores eléctricos http://www.scribd.com/doc/2404238/Circuitos-de-control-y-motores-electricos-Parte-2

[7] Fundamento teórico del arranque de motores eléctricos http://bibdigital.epn.edu.ec/bitstream/15000/385/1/CD-0321.pdf. Páginas 5-7, 26-28.

[8] Stephen J. Chapman "Máquinas Eléctricas". Mc Graw Hill.

[9] José Manuel Aller Castro. "Máquinas Eléctricas Rotativas: Introducción a la Teoría General". Universidad Simón Bolivar. Páginas 163-165.

[10] Control de Motores. http://www.scribd.com/doc/6231005/Todos-Los-Diagramas

[11] Capa de Transporte y Puertos en TCP/IP http://www.saulo.net/pub/tcpip/b.htm#3-1

[12] TCP/IP modelo OSI http://www.textoscientificos.com/redes/tcp-ip/comparacion-modelo-osi

[13] JeffFerguson, Brian Patterson, Jason Beres. "La Biblia de C#". Anaya.

[14] Kraig Mitzner "Complete PCB Design using Orcad Capture and Layout". Newnes.

[15] Servidor Serial Tibbo. http://www.tibbo.com