ESCUELA POLITÉCNICA DEL EJÉRCITO

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO EN INGENIERÍA ELECTRÓNICA

"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE CONTROL Y MONITOREO CENTRALIZADO DE FLUJO VEHICULAR Y PEATONAL"

JORGE NAPOLEÓN ALMEIDA GARZÓN
SANTIAGO FERNANDO MAFLA LEGARDA

SANGOLQUI – ECUADOR
2008

CERTIFICACIÓN

Por medio de la presente certificamos que el proyecto de grado para la obtención del título en Ingeniería Electrónica titulado **"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE CONTROL Y MONITOREO CENTRALIZADO DE FLUJO VEHICULAR Y PEATONAL"** fue desarrollado en su totalidad por los señores JORGE NAPOLEÓN ALMEIDA GARZÓN y SANTIAGO FERNANDO MAFLA LEGARDA.

Atentamente,

Ing. Rodolfo Gordillo
DIRECTOR

Ing. Wilson Yépez
CODIRECTOR

AGRADECIMIENTO

A Dios y a mis padres por ser parte esencial de mi vida, a los gerentes de MZ Sistemas Eléctricos y Electrónicos quienes hicieron posible el desarrollo de este proyecto, al director y codirector de tesis y amigos, quienes fueron respaldo constante para el desarrollo del mismo.

Y especial agradecimiento a los trabajadores de MZ Sistemas Eléctricos y Electrónicos quienes colaboraron incondicionalmente con la realización del proyecto.

Jorge Napoleón Almeida Garzón

Mi principal agradecimiento es a Dios, quien a través de mis padres me ha permitido prepararme académica. Ellos han constituido el pilar fundamental de mi vida y me debo a ellos mi formación personal y profesional. Gracias por el esfuerzo, dedicación y apoyo brindado durante cada etapa de mi vida.

Santiago Fernando Mafla Legarda

DEDICATORIA

Este trabajo esta dedicado a mis Padres que son guías y coautores de mi educación y por quienes he llegado a obtener mis logros. A mis hermanas por su apoyo en buenos y malos momentos y a mis amigos que juntos compartimos un sin numero de experiencias en el camino por lograr la culminación de la carrera.

Jorge Napoleón Almeida Garzón

El desarrollo de este proyecto se lo dedico a mis padres, Marco Ramiro y Nancy Ximena, cuyo espíritu y ejemplo vive en mi corazón; a mi abuelita Zoilita por su apoyo y consejo; a mi hermana Paulina por su compañía y alegría y a mi sobrino Daniel quien con su inocencia me ha enseñando el valor de la responsabilidad en la vida.

Santiago Fernando Mafla Legarda

PRÓLOGO

El crecimiento acelerado del parque automotor en nuestro país ha impulsado el desarrollo de métodos y procedimiento que permitan establecer el equilibrio entre el desarrollo tecnológico, la comodidad de los usuarios de las vías y el medio ambiente. Debido a esto se requiere de sistemas de control administrables, mismos que permitan encontrar soluciones óptimas a las diferentes circunstancias del entorno, considerando que estas son diversas en nuestro medio, por lo que es necesario realizar un análisis sociocultural de cada lugar.

De este modo, el presente proyecto ofrece una solución general a este fenómeno, a partir del desarrollo de un sistema de control de flujo vehicular y peatonal, cuyo diseño se basa en la utilización de software versátil y la utilización de dispositivos que brindan un mínimo impacto ambiental.

Además permite la adquisición, tratamiento y análisis de información, a partir de lo cual se busca establecer alternativas de solución a los inconvenientes de tráfico vehicular que puedan generarse. La implementación del sistema requiere de un costo de inversión mínimo y reduce gastos correspondientes al consumo de energía eléctrica.

Es posible realizar la programación del controlador de dos modos, uno remoto a través de una red Ethernet y otro local mediante el uso de un LCD y Teclado Matricial, ofreciendo al administrador alternativas de configuración del sistema.

Así, se pudo realizar un sistema fiable, altamente flexible, con capacidad de expansión, bajo el cumplimiento de normas tanto técnicas como viales, constituido de dispositivos que ofrecen un gran desempeño, alto rendimiento y larga vida útil.

ÍNDICE DE CONTENIDO

CAPÍTULO I – FUNDAMENTACIÓN TEÓRICA

CAPÍTULO II – SISTEMA CONTROLADOR

CAPÍTULO III – COMUNICACIÓN Y SOFTWARE

CAPÍTULO IV – DISEÑO DEL SISTEMA DE CONTROL DE TRÁFICO

CAPÍTULO V – DESARROLLO DE INTERFAZ GRÁFICA

CAPITULO I

FUNDAMENTACIÓN TEÓRICA

1.1 INTRODUCCION

Durante la evolución de la humanidad se ha vuelto indispensable el intercambio de productos para la subsistencia de la raza humana, es así que se crearon las vías de comunicación entre varios lugares, para la transportación de personas y productos. Mucho tiempo después, la creación de los automóviles acarreo varios problemas y soluciones para la humanidad [1].

Así se construyeron las vías de transporte, sin tomar en cuenta la señalización, ya que no se creía indispensable, descuidando de este modo un aspecto imprescindible dentro del desarrollo vial. A partir de esto, el hombre comenzó a realizar en las ciudades, el trazo urbano de la tierra y su comunicación.

De esta evolución nacen las vías e intersecciones que actualmente conocemos, convirtiéndose en un problema su uso por el aumento exorbitantemente de vehículos.

Además, es importante considerar la naturaleza del proceso innovador y de la relación entre desarrollo tecnológico y sociedad; tomando en cuenta que la innovación no es sólo la aplicación de los resultados de investigación de alto nivel, sino que también depende de las capacidades emprendedoras, estratégicas, de decisión, organizativas e imaginativas.

En nuestro país el control del transporte constituye uno de los principales problemas para las autoridades a cargo, ya que si bien se han emprendido planes de contingencia para evitar accidentes de tránsito en las vías, estos son insuficientes, debido a diversos factores, tales como la falta de educación vial, mala utilización de las señalización, desacato a las normas de tránsito y falta de sistemas modernos de control de flujo vehicular y peatonal.

Con el crecimiento del flujo vehicular surge la necesidad de que sea controlado a través de sistemas que garanticen su desempeño, optimizando la fluidez de la circulación, lo cual se hace posible a través del desarrollo de sistemas inteligentes de control vehicular.

Existen en Ecuador, ciudades como Quito y Guayaquil, que debido a la cantidad de flujo vehicular existente, poseen sistemas de monitoreo centralizado, lo cual ha facilitado el descongestionamiento de las vías, ofreciendo mayor comodidad a sus usuarios.

Como solución al rápido crecimiento del parque automotor en los últimos años se ha visto necesaria la adaptación de sistemas de control moderno de tráfico, considerando que estos son altamente confiables, además de constituirse en un beneficio económico debido a que permitirá reducir egresos correspondientes al consumo de energía eléctrica, aumento de la vida útil de los equipos, fácil mantenimiento, así como también de ser un sistema altamente flexible.

El presente proyecto constituye una alternativa de diseño e implementación de un sistema de control de tiempos, sincronismo, monitoreo, almacenamiento y procesamiento de información asociado a las condiciones de flujo vehicular, de modo que a través de la administración y gestión de las intersecciones de un área determinada, permita la optimización de la red vial para el tránsito fluido y seguro de vehículos.

Con ello se logra obtener múltiples beneficios, como la reducción de tiempos de viaje, el aumento de la capacidad vial, la disminución de accidentes de tránsito, disminución de gases y partículas contaminantes en el ambiente, mejorando así la calidad de vida, no solo de los usuarios de las vías sino de los habitantes en general, fomentando el desarrollo social y reduciendo el impacto al medio ambiente.

1.2 CARACTERISTICAS GENERALES DEL PROYECTO

El sistema de semaforización será monitoreado desde un centro de control compuesto por una estación central con sus respectivos periféricos ubicados en cada intersección, mismos que estarán comunicados a través de una red Ethernet.

El monitoreo centralizado será controlado y supervisado por un operador, quien visualizará el comportamiento del sistema en tiempo real, a través de una interfaz gráfica, a través de la cual se tendrá acceso a la configuración de los diferentes parámetros de control y a los modos de manejo, tanto local como remoto.

El software utilizado está orientado a la administración de tránsito a través de la selección de tiempos de acuerdo con la demanda del tránsito, obtenida de la adquisición de información, la cual es manejada a través de una base de datos que adicionalmente almacena los tiempos de sincronismo y horas críticas de flujo vehicular y peatonal.

El diseño de hardware y software del sistema de control permite la configuración del sistema de dos modos, remoto desde el centro de control a través de la red y local, en el periférico respectivo mediante el uso de un LCD y Teclado ubicado en el tablero de control.

El dispositivo de control del periférico es una tarjeta electrónica programable que permite controlar los flujos de tráfico vehicular y peatonal en una determinada intersección, por medio de un programa local, de acuerdo con las características de la intersección.

La tarjeta controladora de cada periférico realizará la comunicación con la estación central mediante el envío y la recepción de datos requeridos por el operador, los mismos que son almacenados en la base de datos para su posterior análisis.

El sistema utilizará semáforos con indicadores compuestos de matrices de leds, con lo que se reducirá el consumo de energía eléctrica y aumentará su vida útil. Los semáforos destinados para el flujo vehicular tendrán iluminación circular mientras que los peatonales

ofrecerán diferentes formas de acuerdo a su estado, Estos además estarán dotados de sistemas de audio imprescindibles para personas con discapacidad visual.

El semáforo periférico ofrecerá además una capacidad de control de hasta cuatro fases para constituirse como sistema adaptable a cada una de las diversas circunstancias. Su uso estará de acuerdo a la cantidad de flujo vehicular existente y las consideraciones de tipo de intersección y horas críticas.

El computador principal tiene la capacidad de control para un mayor número de intersecciones, considerando el crecimiento y necesidades a mediano y largo plazo.

En general el Sistema de Control y Monitoreo Centralizado de Flujo Vehicular y Peatonal cuenta con hardware y software versátil y moderno para realizar el control en forma coordinada, modificando las fases según la demanda vehicular o peatonal, garantizando a su vez la coherencia con cada una de las intersecciones.

1.3 NORMAS TECNICAS

Las normas técnicas permiten desarrollar proyectos basados en parámetros generales, comunes dentro de un servicio o producto; logrando así la coordinación y orden de estos, así como también garantizan su correcto funcionamiento, ofreciendo seguridad durante su desempeño.

Es de este modo que los sistemas de control vehicular deben cumplir con las normas vigentes en esta área, por lo que es necesario su análisis y consideración durante el diseño del mismo.

Cabe indicar que cada país cuenta con su sistema de normalización; en Ecuador el organismo encargado es el INEN, Instituto Ecuatoriano de Normalización; el mismo que conjuntamente con la Policía Nacional, en el Departamento de Ingeniería de Tránsito, perteneciente a su vez a la Dirección Nacional de Tránsito y Transporte Terrestre han planteado las normativas y reglamentos a seguir en la implementación de este tipo de sistemas [1].

A través de la reglamentación planteada por estos organismos, se busca establecer los diferentes aspectos que deben cumplir los sistemas de semaforización, con el fin de ofrecer seguridad a los usuarios de las vías, tanto a conductores como a peatones, además de proteger al medio ambiente, previniendo prácticas que puedan generar demoras innecesarias o incluso accidentes.

Las principales normas consideradas en el proyecto de diseño e implementación del sistema de control y monitoreo centralizado de flujo vehicular y peatonal se indican a continuación.

1.3.1 SISTEMA DE SEMAFORIZACIÓN

1.3.1.1 SEMÁFOROS VEHICULARES

- "Todos los lentes de las secciones de los semáforos tendrán un aspecto circular.
- El semáforo estándar, que tiene lentes de 200 mm de diámetro y utiliza focos incandescentes de mínimo 65 Watios y/o Leds.
- Un semáforo de funcionamiento superior, que tiene lentes de 300mm de diámetro y utiliza focos incandescentes de mínimo 110 Watios y/o leds.
- Cuando la aproximación es igual o menor de 50 Km/h, se debe utilizar semáforos con lentes de 200mm.
- Si la velocidad de aproximación es mayor de 50 Km/h se debe utilizar semáforos con lentes de 300mm.
- El material de los lentes debe ser de policarbonato"[4].

1.3.1.2 SEMÁFOROS PEATONALES

- "Los lentes de las secciones tendrán un aspecto circular, sin embargo en algunos casos existen excepciones.
- En cada terminación de un cruce peatonal marcado debe instalarse un semáforo peatonal; este debe ser localizado dentro de 1 metro de la proyección del extremo del cruce marcado y enfocado al lado opuesto del cruce.

- Si el ancho del cruce excede de 10,00 m, deben instalarse en cada terminación del cruce dos semáforos peatonales.
- Si la distancia de cruce de calzada excede de 30,00 m, es necesario instalar sobre un parterre semáforos peatonales complementarios.
- Los semáforos peatonales deben ser instalados y si es necesario enfocados, para asegurar que este es obvio al cruce que es controlado por el mismo.
- Cuando un cruce es desfasado como dos movimientos separados, cada fase debe ser semaforizado como un cruce separado" [4].

1.3.1.3 ORIENTACIÓN O ENFOQUE DE SEMÁFOROS

- "El rango de visibilidad de cada semáforo es determinado por su posición, rendimiento fotométrico y su orientación.
- Semáforos de arranque deben ser enfocados hacia un punto 3,00 m atrás de la línea de pare al centro de la aproximación.
- Semáforos de maniobra deben ser enfocados hacia el centro de línea de pare"[4].

1.3.1.4 ALTURA DE MONTAJE

- "El montaje de los semáforos dependerá del sistema más económico y funcional.
- La altura de montaje es medida desde el nivel de la calzada hasta la parte inferior del cuerpo del semáforo.
- La altura de montaje en poste o columna debe ser mínimo de 3,30m.
- Donde se requiera que el semáforo sea visible dentro de 20,00m, la altura de montaje de todos los semáforos para esa exhibición puede ser reducido a 2,40m.
- Para los semáforos peatonales, la altura de montaje debe ser de 2,40m.
- Semáforos aéreos deben ser montados a una altura de 5,30m a 5,80m desde la superficie de la calzada a la parte inferior de la pantalla de respaldo del semáforo.
- Una altura mayor o menor al rango establecido, significaría que el semáforo aéreo estaría localizado fuera de la línea de visibilidad del conductor.
- La altura de montaje esta relacionada con la línea de vista normal del conductor como se muestra en la figura 1.1" [4].

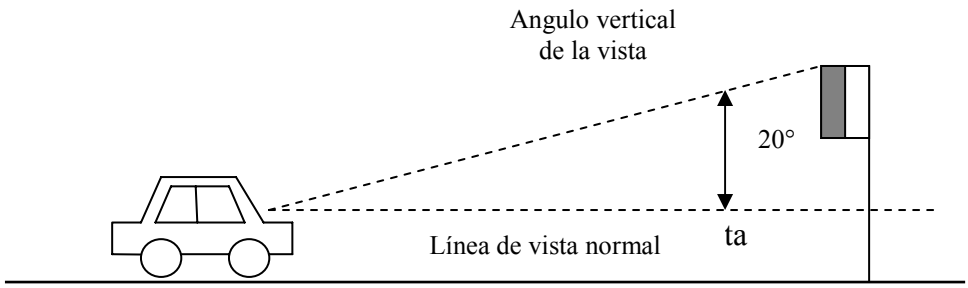


Figura 1.1 Diagrama de altura de montaje

1.3.1.5 VISERAS

"Se utilizan para modificar la cubierta visual angular del semáforo, es decir corta las luces del semáforo de la vista de conductores en otras aproximaciones y/o proteger el sistema óptico del semáforo, de la incidencia de la luz solar la cual puede producir una falsa iluminación.

Existen 2 tipos de viseras, abiertas y cerradas. Hay dos clases de viseras cerradas: cortas y largas".

1.3.1.6 POSTES DE SEMÁFOROS

"Son la estructura de soporte donde se anclan los elementos y dispositivos del semáforo. Existen 4 tipos de postes.

Tipo 1: La longitud de este poste está condicionado a las necesidades y se utiliza solamente en forma temporal o en especiales circunstancias donde no se puede usar el poste tipo 2.

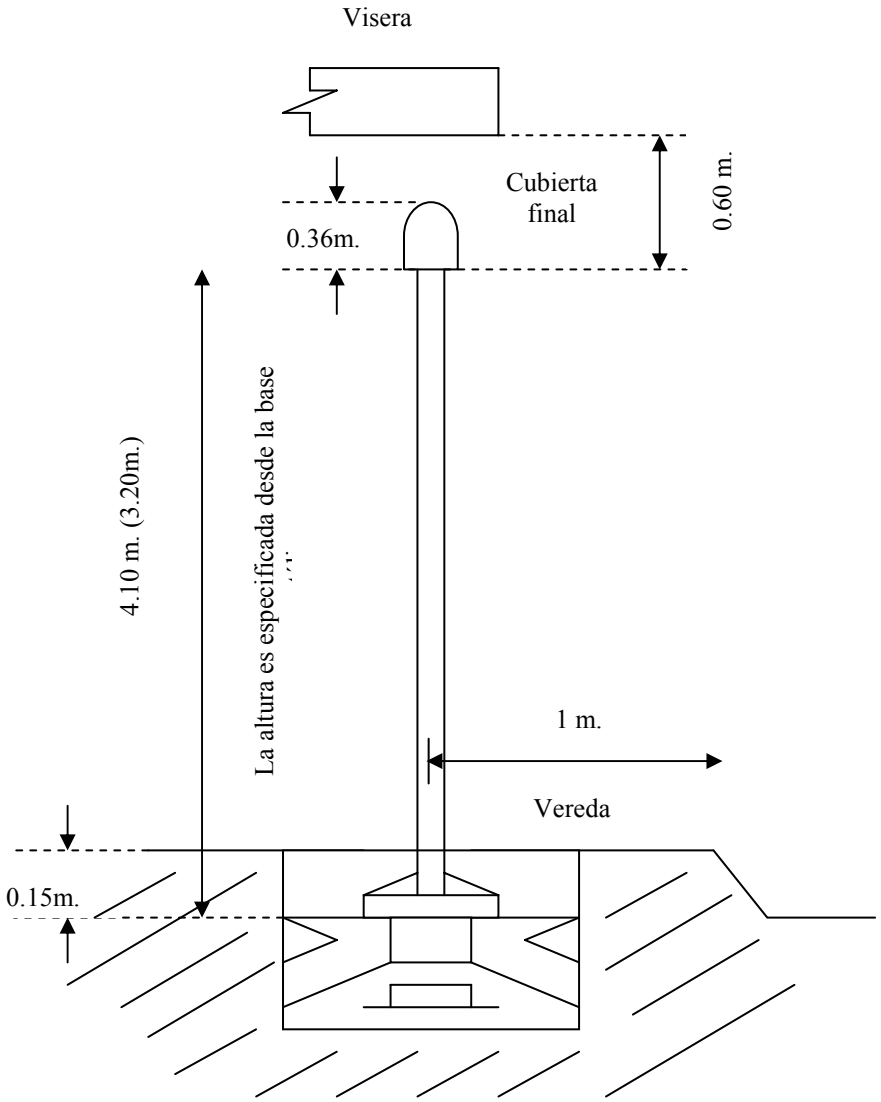


Figura 1.2 Diagrama de postes

Tipo 2: la longitud normal de este poste es de 4,10m pero cuando es utilizado para colocar exclusivamente semáforos peatonales, la longitud es de 3,20m. También se puede variar su longitud cuando es necesario evitar obstáculos como viseras; de igual forma, si no es posible utilizar un brazo aéreo, se puede emplear un poste tipo1 de 4,60m de largo. Las figuras 1.2 y 1.3 muestran los diagramas de este tipo de postes.

Tipo 3: Estos postes se utilizan cuando los semáforos peatonales para un movimiento peatonal particular son colocados en otros postes, son pequeños y sirven para colocar botones de presión peatonales, su altura es de 1,50m.

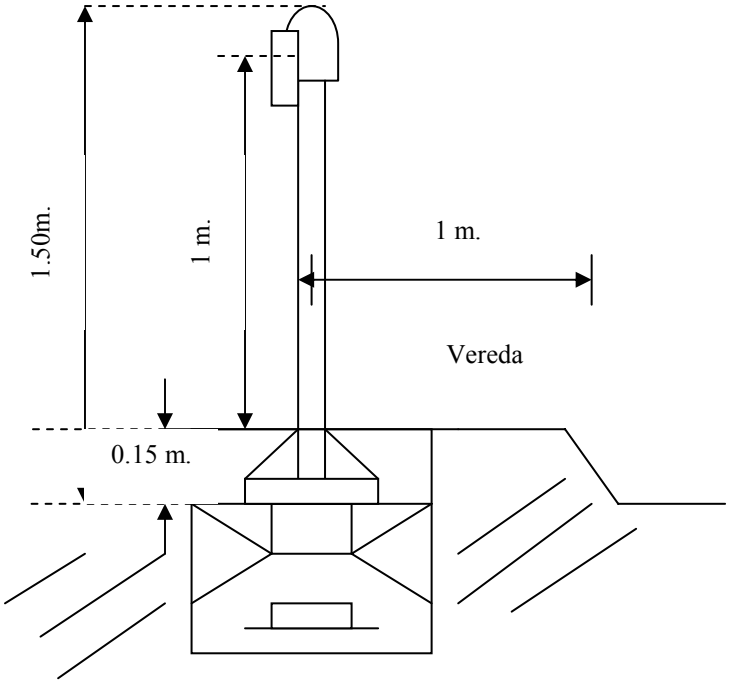


Figura 1.3 Diagrama de postes

Tipo 4 Postes Aéreos (Báculos): Existen dos tipos de postes aéreos, el tipo 4 C, brazo corto y el tipo 4L brazo largo.

Estos postes deben ser instalados manteniendo una distancia mínima de 5,3m desde la superficie de la calzada y a distancias prudentes de las líneas eléctricas.

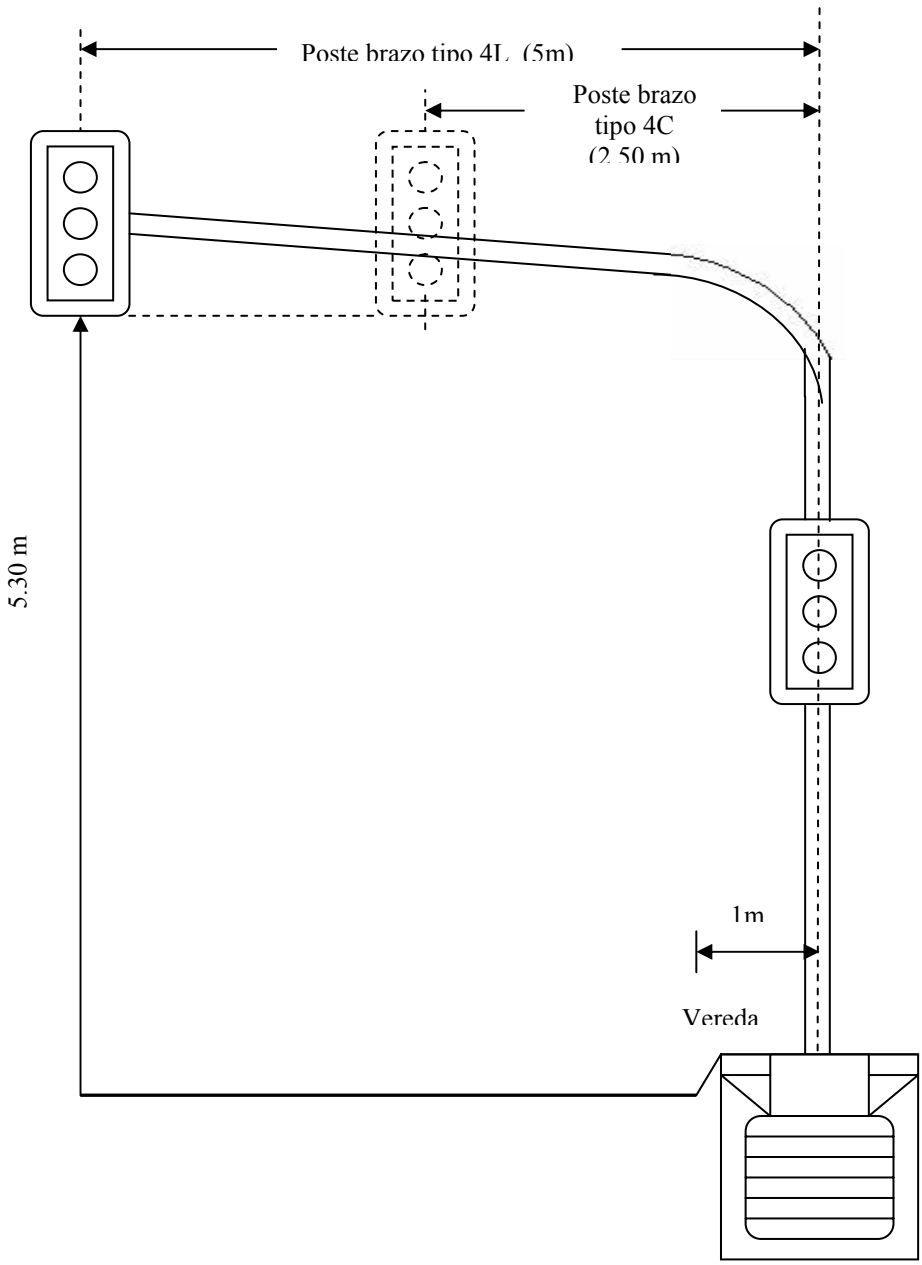


Figura 1.4 Diagrama de postes

Se deben instalar en:

- Cualquier aproximación en donde la distancia de visibilidad de parada no puede ser logrado con semáforos colocados en postes tipo 2.
- Sobre o cerca de línea de parada de la aproximación afectada.
- En el lado de salida al frente de la intersección si es que no es posible instalarlo cerca de la línea de parada.

- En cualquier aproximación de 3 carriles, si es que no se puede instalar un poste medio en el parterre.
- En cualquier aproximación de 4 carriles en asociación con un poste medio, instalado en el parterre" [4].

1.3.1.7 LOCALIZACIÓN DE LOS POSTES

"Los postes de semáforos deben instalarse de la siguiente manera:

- A 1,00m de la distancia mínima del filo o borde de la vereda, a menos que en el plano de diseño se indique en otra posición.
- En el centro de la línea de parada, cuando es necesario obtener una mejor visibilidad del semáforo, o para evitar cajas de revisión, desagües, cajas telefónicas, postes eléctricos, etc.
- Cuando las localizaciones indicadas no se pueden realizar debido a cualquier obstáculo, los postes deben colocarse preferiblemente delante de su colocación ideal antes que más cerca de la intersección; los semáforos peatonales pueden ser requeridos instalarlos en otro poste mejor situado, pero hay que tomar en cuenta que las botoneras o pulsadores, deben ser colocados dentro de 1m del cruce peatonal asociado.
- A una distancia de 1,20m del inicio del parterre central cuando este parterre está atrás del cruce peatonal y a la misma distancia cuando no hay cruce peatonal.
- A una distancia mínima de 500mm desde borde del parterre central cuando existe un movimiento peatonal a través del parterre y la punta de abre hacia delante. (mínimo 2m)" [4].

1.3.1.8 SEPARACIÓN HACIA INSTALACIONES ELÉCTRICAS

"Las separaciones varían dependiendo del voltaje de transmisión de los cables eléctricos, su aislamiento y de ordenanzas de empresas eléctricas locales; por lo tanto, si el equipo de semáforos va a ser colocado cerca de cables eléctricos, debe consultarse con la empresa eléctrica correspondiente; manteniendo una distancia de separación mínimo de 0,60m.

A continuación se muestra en la figura 1.5 la posición de los elementos que conforman los semáforos".

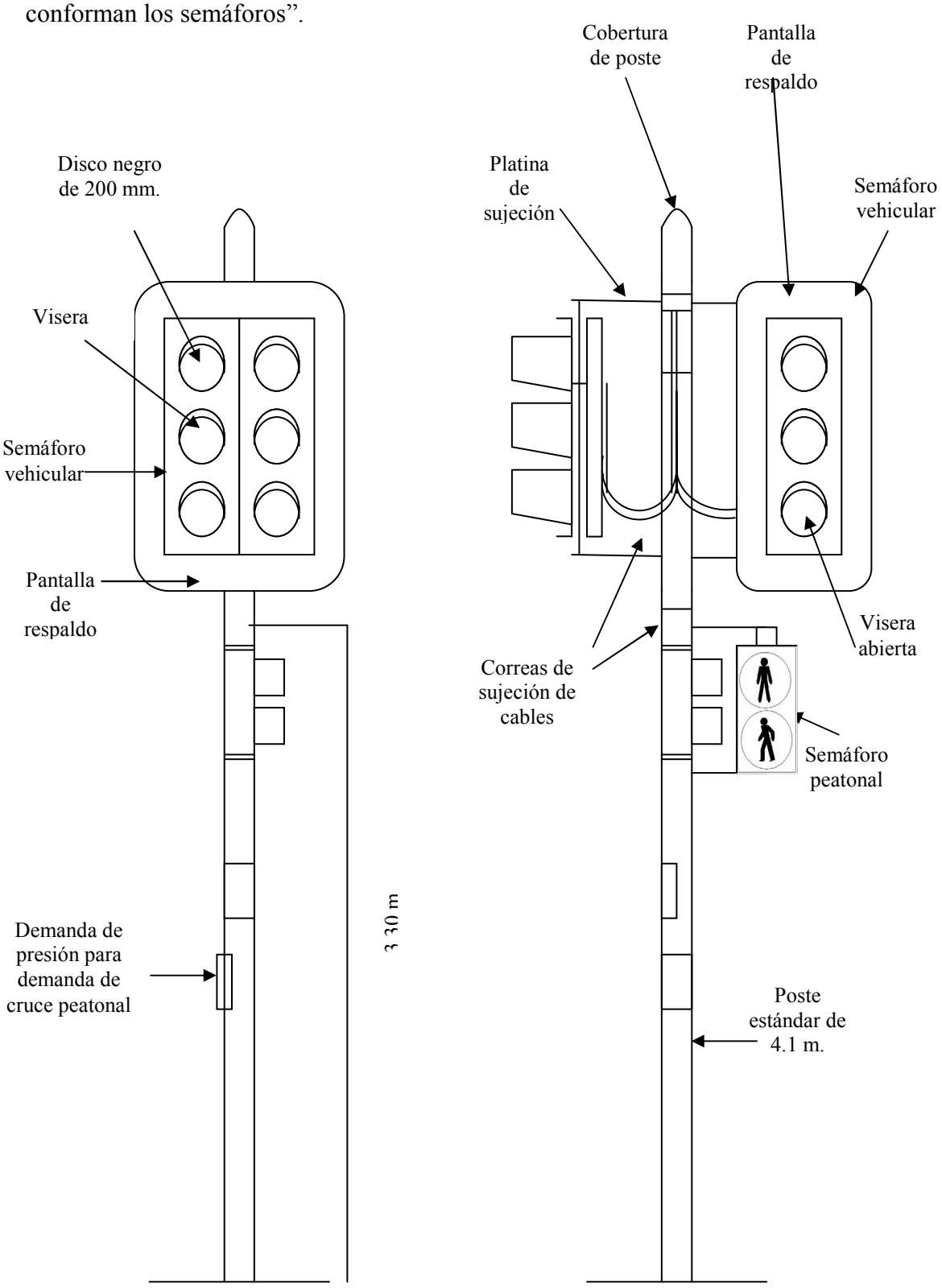


Figura 1.5 Diagrama de descripción de postes

1.3.2 CONTROLADOR

1.3.2.1 POSICIÓN

"El lugar de instalación del controlador de semáforos debe satisfacer los siguientes aspectos:

- Debe instalarse en la cercanía de la red pública.
- Estar en una posición de tal manera que desde el controlador haya una visibilidad clara de todas las aproximaciones, esto facilitará las calibraciones de tiempo y mantenimiento.
- No debe obstruir el derecho de vía de los peatones.
- No debe estar expuesto a que sea dañado por el tráfico vehicular.
- De ser posible, debe ser colocado junto a la pared más próxima.
- El ruido provocado por la operación del controlador no debe molestar a los residentes cercanos.
- Estar alejados de sitios sujetos a inundaciones.
- Debe instalarse alejado de futuras ampliaciones de vías" [4].

1.3.3 RED DEL SISTEMA

Para el diseño del sistema de ha considerado pertinente el cumplimiento de estándares de régimen internacional mismos que serán considerados en el desarrollo general del proyecto.

1.3.3.1 RED ETHERNET

Sus estándares son indispensables en el diseño e implementación de los sistemas que cableado estructurados para oficinas, o para ambientes de campo. Definen las distancias, los conectadores, las arquitecturas del sistema del cable, las características de funcionamiento y los requisitos de la instalación de cable [3].

Estándares: IEEE 802.3, IEEE 802.2 (control de enlace lógico)

Características:

- Tecnología: 100BaseTX
- Velocidad de transmisión: 100Mbps
- Tipo de cable: Par Trenzado, categoría 5UTP, Estándar ANSI/TIA/EIA 568-B.2.
- Distancia máxima: 100 m
- Tipología: Half Duplex(hub) y Full Duplex(switch)

1.4 MODOS DE OPERACIÓN DEL SISTEMA DE CONTROL DE TRÁFICO

El sistema de control presenta dos modos de operación, local y remoto, cada uno ofrecen garantía y robustez, además permiten la configuración de todos los parámetros de control, necesarios para conseguir el sincronismo y coordinación en cada una de las intersecciones. A continuación se indican los modos de operación.

1.4.1 MODO REMOTO

Este modo está destinado para realizar el control del sistema a distancia, lo cual se logra a través de la red Ethernet, la misma que permite la interconexión entre la tarjeta programable, es decir el controlador y el computador principal con sus respectivos periféricos, ubicado en el centro de monitoreo.

Para ello se desarrollan programas mediante los cuales se establece la comunicación, tanto la parte correspondiente a la coordinación, en el controlador, como la interfaz de monitoreo, la cual se encuentra en el computador principal.

1.4.2 MODO LOCAL

A través de este modo se realiza el control del sistema de forma directa, es decir en el panel ubicado en la propia intersección, en donde se encuentra el controlador.

La configuración local de los parámetros se lleva a cabo a través de un teclado y un LCD, con lo que el sistema ofrece al operador una alternativa de programación práctica y

segura para realizar los cambios respectivos de acuerdo a las circunstancias del entorno local.

Este modo es muy útil en los siguientes casos:

- Realizar pruebas de control durante la implementación y adaptación del sistema en nuevas intersecciones.
- Realizar el mantenimiento y reparación de equipos en aquellos ya instalados.
- Fallo de las comunicaciones.
- Fallo en el sistema central por suministro eléctrico.
- Uso incorrecto de los equipos de monitoreo.

CAPITULO II

SISTEMA CONTROLADOR

2.1 INTRODUCCION

El desarrollo tecnológico ha permitido realizar el control de procesos, comunicación, automatización, mediante dispositivos de hardware y software que se ajustan a las características propias de estos, es así que cada dispositivo ofrece diferentes tipos de soluciones, optimizando recursos, brindando además alta confiabilidad y eficiencia.

Estos dispositivos electrónicos, conocidos como Controladores, los cuales a través de su lógica programable se encargan de recibir señales eléctricas en sus entradas procesan la información y activan órdenes de salida que se envían hacia los correspondientes actuadotes.

En la actualidad existe gran variedad de controladores, los cuales ofrecen beneficios y soluciones prácticas a sistemas electrónicos, los mismos que están en función de la aplicación a desarrollar.

Entre los controladores de mayor uso, se encuentran los Controladores Lógicos Programables (PLC), microcontroladores, tarjetas de adquisición, y las modernas tarjetas electrónicas embebidas, las cuales ofrecen ventajas en el desarrollo de aplicaciones a un costo de inversión relativamente bajo.

Es así, que al momento de seleccionar el controlador a utilizar es necesario considerar parámetros tales como:

- Consumo de energía eléctrica.
- Niveles de tensión y corriente.
- Velocidad de respuesta.
- Número y tipo de entradas y salidas.
- Tipo de comunicación.
- Número de puertos.
- Capacidad de expansión.
- Tiempo de vida útil.
- Sistema de protección.
- Inmunidad al ruido e interferencias.

A partir del análisis de los diferentes controladores, se realiza la selección entre los dispositivos que ofrecen los recursos específicos para el desarrollo del proyecto, considerando el costo del producto en el mercado.

El presente proyecto requiere un controlador que permita establecer transmisión de datos serialmente y mediante Ethernet, manejo de interrupciones, configuración de puertos bidireccionales para señales digitales, que incluya fuentes de alimentación en la placa, y que sea inmune a interferencia y al ruido. Además que no requiera la adaptación de hardware externo.

Por lo que para el desarrollo del diseño e implementación en este proyecto particular se ha considerado la utilización de una de estas tarjetas embebidas, El dispositivo de control seleccionado es la Tarjeta Rabbit con módulo PowerCore Flex 3800, misma que posee las características necesarias para la presente aplicación.

2.2 TARJETA ELECTRONICA RABBIT

La tarjeta posee un módulo, PowerCore, que incluye un sistema microprocesador que permite trabajar en red, además brinda la posibilidad de conectar la fuente de alimentación en la misma placa. Consta de 50 pines correspondientes a entradas, salidas y a las fuentes de alimentación, además tiene soporte mecánico robusto.

El desarrollo de la programación se realiza bajo la plataforma Dynamic C, la misma que incluye extensas librerías que garantizan el rápido manejo de recursos del controlador.

Los módulos se programan en Dynamic C y se descargan a través de puerto de comunicación RS-232, existe la posibilidad de hacerlo mediante el puerto USB con la utilización del conversor RS-232/USB, o directamente mediante el enlace Ethernet utilizando el manejador de descarga Dynamic C con el enlace Rabbit [5]. La figura 2.1 muestran al módulo PowerCore Flex 3800 y la figura 2.2 indica su disposición en la placa de la tarjeta.

Figura 2.1 Módulo PowerCore Flex 3800

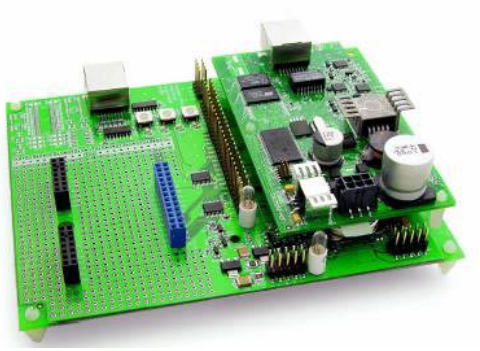


Figura 2.2 Módulo PowerCore Flex 3800 colocada sobre la placa

2.2.1 CARACTERÍSTICAS BÁSICAS

- 39 líneas de entrada/salida de propósito general configurables.
- 3 entradas digitales adicionales.
- 2 salidas digitales adicionales.
- 5 puertos seriales, 3.3V CMOS compatible, con una tasa asíncrona superior a 6.45 Mbps.
- 3 puertos son configurables como puerto serial de reloj (SPI), 2 puertos como puerto serial HDLC y 1 puerto como puerto serial SDLC.
- Uno de los puertos seriales está destinado a la programación.
- 512K de memoria flash para almacenamiento de instrucciones.
- 256K de memoria RAM estática para datos.
- Batería de respaldo.

2.2.2 CARACTERÍSTICAS ELÉCTRICAS

Alimentación

DC: no regulada de 8 a 43V DC (potencia 13.3W)

AC: 24-60V AC con transformador con derivación central (potencia 13.3W)
12-36V AC con transformador estándar (potencia 13.3W)

Corriente límite para placa a 5V DC regulados: 2A.

Corriente promedio para circuitos en placa: 400mA.

Salidas

AC/DC: +3.45V DC: I3VoutM = 350mA.
+5V DC: I5VoutM = [1600mA - I3VoutM]
No regulada AC/DC:
IunregM = 1600mA – (I5VoutA + I3VoutA) X (6.7V/Vin)

Temperatura de operación: –40°C a +70°C

Indicaciones:

- La fuente de alimentación de +3.45V DC tiene una tolerancia de ±150mA; la fuente de +5V de ±250mA.
- I3VoutM = corriente máxima de salida disponible para el circuito del usuario correspondiente a la alimentación de +3.45V.
- I3VoutA = corriente actual de salida utilizada por el circuito del usuario correspondiente a la alimentación de +3.45V.
- I5VoutM = corriente máxima de salida disponible para el circuito del usuario correspondiente a la alimentación de +5V.
- I5VoutA = corriente actual de salida utilizada por el circuito del usuario correspondiente a la alimentación de +5V.
- IunregM = máxima corriente de salida AC/DC no regulada disponible para el circuito del usuario.

2.2.3 VENTAJAS

Este tipo de tarjetas ofrece ventajas en la realización de proyectos, debido a que tienen múltiples recursos, tales como:

- Comunicación Ethernet.
- Alta capacidad de almacenamiento en memoria.
- Comunicación serial.
- Conversor Análogo Digital.

- Fuentes de alimentación incluidas en placa.
- Batería de respaldo.
- Manejo de señales análogas y digitales.
- Inmunes a interferencia y ruido.

2.2.4 MODULO POWERCORE FLEX

El módulo es de tamaño reducido, 60 mm × 102 mm × 28 mm [6].

Su disposición física se muestra en las figuras 2.3, 2.4 y 2.5.

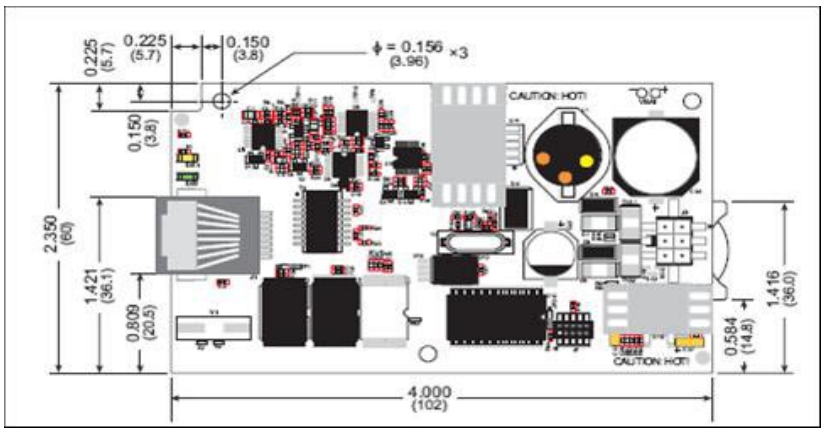


Figura 2.3 Vista superior del módulo

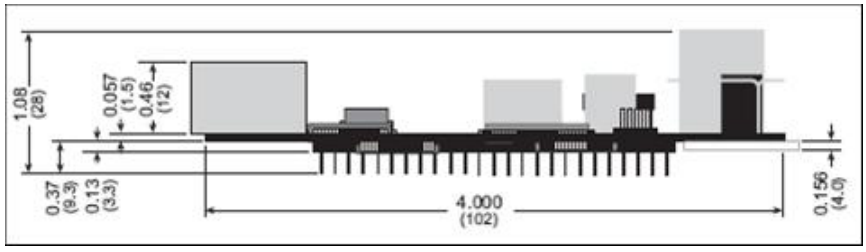


Figura 2.4 Vista frontal del módulo

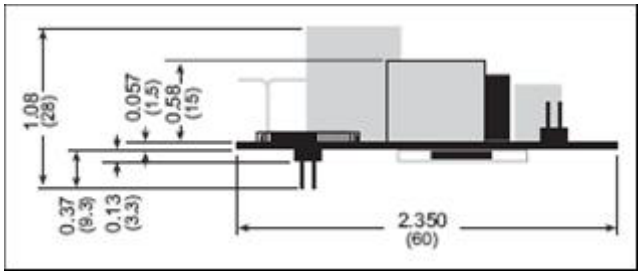


Figura 2.5 Vista lateral del módulo

2.2.4.1 SUBSISTEMAS

El módulo posee subsistemas, mediante los cuales realiza las operaciones requeridas. Cada una tiene sus funciones y características específicas y en conjunto garantizan su correcto funcionamiento [7]. Estos subsistemas se disponen del siguiente modo:

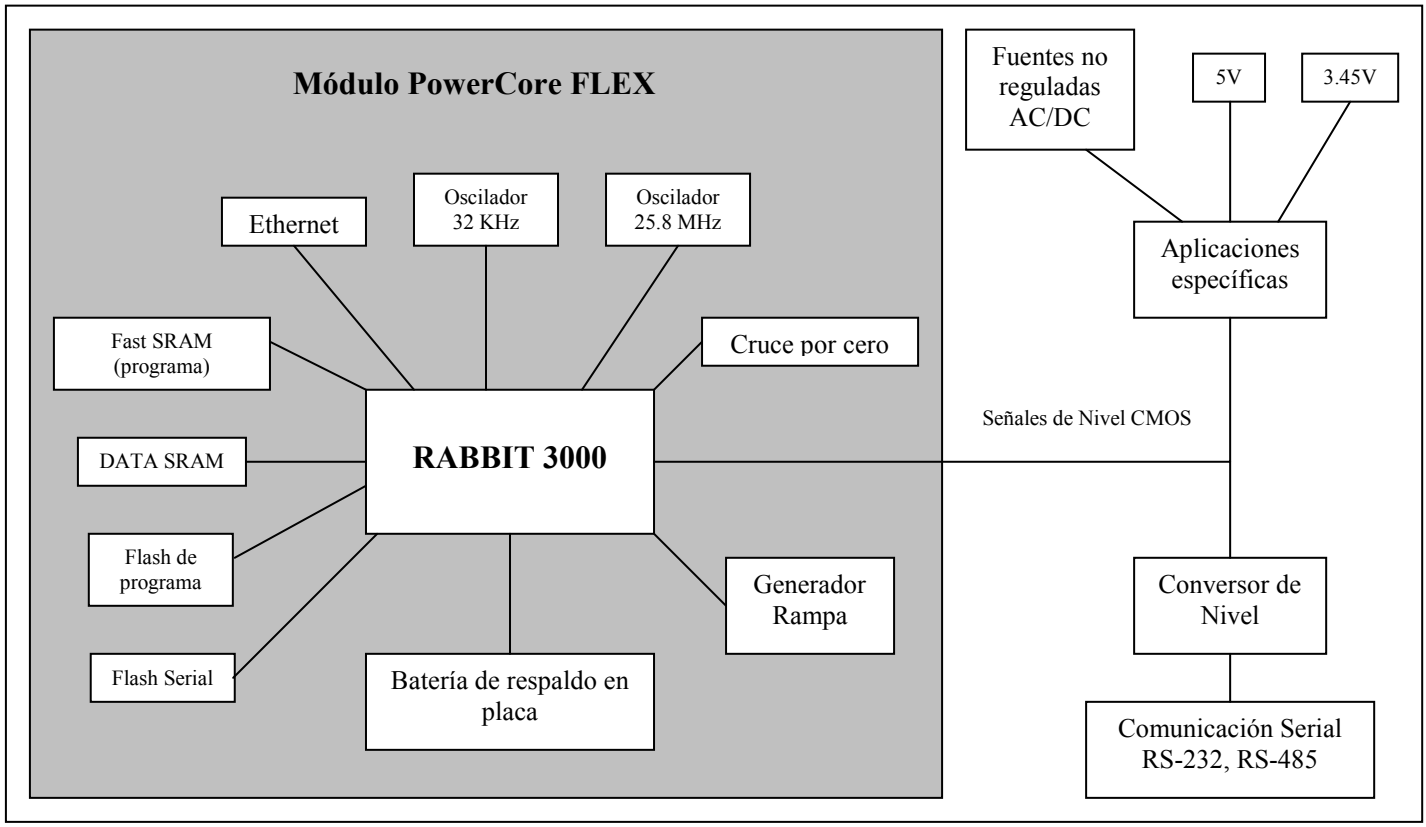


Figura 2.6 Subsistemas del Módulo PowerCore Flex

SISTEMA DE RELOJ

El oscilador principal utiliza un cristal externo con una frecuencia típica en el rango de 1.8 MHz a 26 MHz. El reloj del procesador se deriva de la salida del oscilador mediante un doblador de frecuencia, usando la frecuencia directamente, o por división de frecuencia para 2, 4, 6 u 8. El reloj del procesador puede también ser manejado por un oscilador en tiempo real de 32.768 KHz para operaciones de muy baja potencia, en cuyo caso puede ser desactivado mediante el control de software [8].

ENTRADA DE OSCILADOR DE 32.768 KHZ

Esta es diseñada para aceptar un reloj de 32.768 KHz, el mismo que maneja la batería de respaldo. Además, es usada para manejar el watchdog timer y para generar el reloj en Baudios para el puerto serial A durante la secuencia de arranque.

BUSES INTERNOS Y EXTERNOS

Existen líneas de dirección y de datos que son utilizadas por la Flash memory, SRAM y los chips de Ethernet, estas corresponden a las A0-A19 y D0-D7, respectivamente. Existe un bus separado de entradas y salidas, el cual es implementado mediante el uso de seis líneas de dirección y ocho de datos. Varios strobes pueden ser implementados para dato de reloj desde o hacia el bus. El bus de entrada/salida es una opción que puede ser habilitada para programas de usuario.

Las ocho líneas bidireccionales de entrada/salida comparten pines con el puerto paralelo A y seis líneas de dirección comparten pines de parte del puerto paralelo B, aunque 64 direcciones de lectura o escritura son directamente disponibles, facilitando de este modo la expansión del espacio de registro del bus que es tan grande como se requiera añadiendo direcciones adicionales de bits, utilizando un registro cargado de 64 registros. Otra alternativa es para usar líneas adicionales de strobe y crear 64 espacios de registros adicionales.

Las instrucciones de entrada y salida Rabbit son utilizadas para acceder a los registros creados en el bus de entrada y salida.

El puerto paralelo A, puede ser usado como un bus de datos externo de entrada y salida para aislar entradas y salidas externas desde el bus de datos principal. Los pines en el puerto B usados como líneas de dirección de bus de entrada y salida pueden ser utilizados como líneas externas cuando el bus de entrada y salida no está habilitado. Los pines del puerto paralelo B, PB2-PB7, pueden ser utilizados también como un bus de direcciones auxiliar [9].

2.2.4.2 DISPOSICION DE PUERTOS Y MEMORIA

La figura muestra el uso de los puertos del microprocesador Rabbit 3000 en el módulo PowerCore. [10]

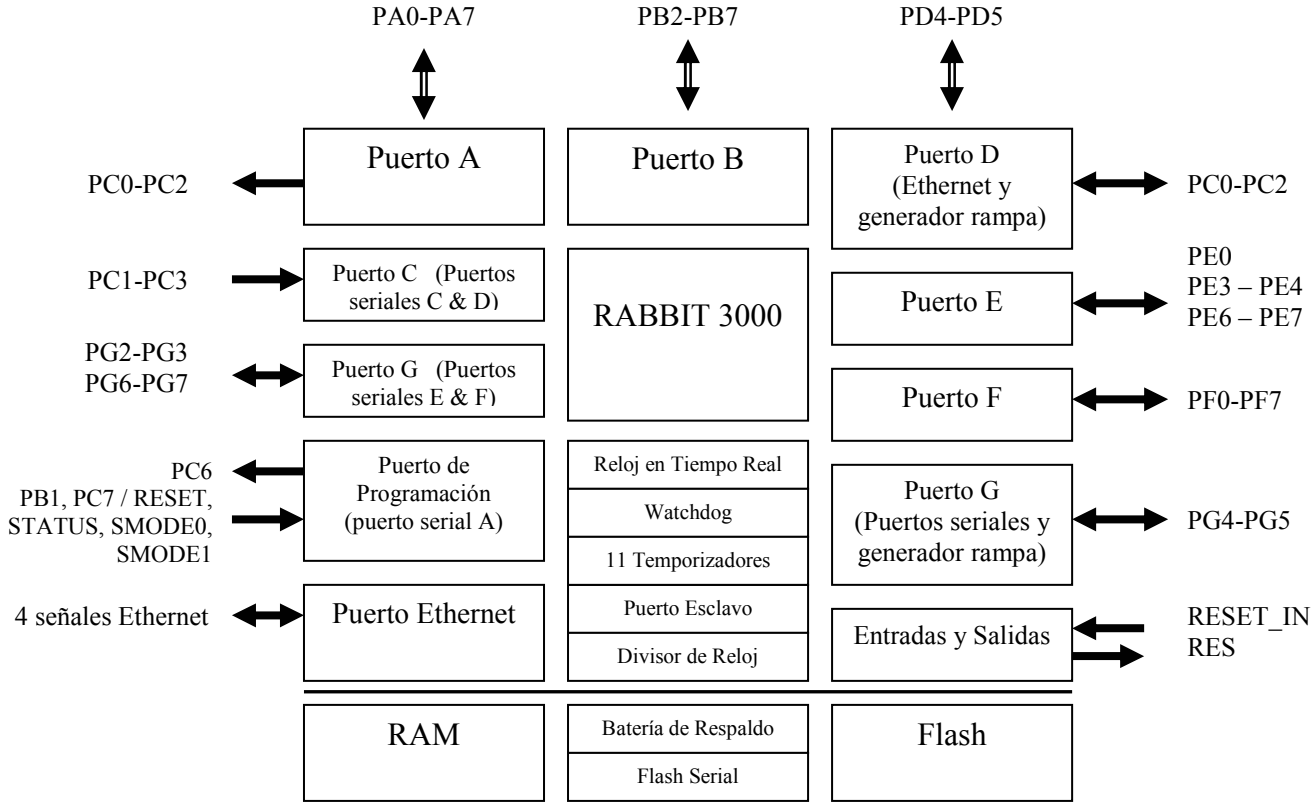


Figura 2.7 Disposición de puertos en módulo PowerCore Flex

2.2.4.3 DESCRIPCIÓN DE RECURSOS

COMUNICACIÓN SERIAL

El módulo PowerCore Flex no posee ningún conector directamente en la placa. Sin embargo, una interfaz serial debe ser incorporada de la placa al módulo, ya que el módulo tiene conectores RS-232 y RS-422. [11]

PUERTOS SERIALES

Existen 6 puertos seriales en la Rabbit 3800, los cuales son: A, B, C, D, E y F. todos pueden operar en un modo asincrónico con una tasa en baudios correspondiente al sistema de reloj dividido para 8. Un puerto asincrónico puede manejar 7 u 8 bits de datos.

El puerto serial A es normalmente utilizado como puerto de programación, pero debe ser usado como asincrónico o como un puerto serial de reloj, una vez que el módulo PowerCore Flex ha sido programado y está operando en el modo *Run*.

El puerto serial B es utilizado para comunicar con la flash serial en el Módulo PowerCore Flex y no está disponible para ninguna otra aplicación.

Los puertos seriales C y D pueden operar en modo de reloj serial, de este modo el dato de entrada o salida tiene sincronismo.

Los puertos seriales E y F, pueden ser configurados como puertos seriales HDLC. El puerto E puede ser configurado como SDLC.

PUERTO DE PROGRAMACIÓN

El Puerto A tiene características especiales que permiten el arranque del sistema después de ser reseteado. Es utilizado además para el desarrollo de software bajo Dynamic C.

El puerto de programación utiliza el puerto serial A del microprocesador para la comunicación y para operaciones de programación, tales como depuración, clonación y

descarga, además para depuración remota sobre una conexión Ethernet, para lo cual se requiere de una placa auxiliar. El pin de status de la Rabbit 3800 también se encuentra en el puerto de programación, ya que esta es una salida que puede ser usada para enviar una señal digital general.

El puerto de programación transmite la información hacia o desde la PC mientras un programa está siendo depurado.

MEMORIA

SRAM

El módulo PowerCore FLEX trabaja a 51.6 MHz y necesitan 512K de SRAM para programación y ejecución.

Flash EPROM

El módulo PowerCore FLEX tiene 512K de flash EPROM.

Flash Serial

El módulo PowerCore FLEX dispone de una memoria serial flash para almacenar datos y páginas web.

GENERADOR RAMPA

El módulo PowerCore Flex tiene un generador rampa, el cual provee una función continua “diente de sierra”. La calibración de esta función se realiza con un voltaje de referencia de 2.5V. La rampa de 400Hz tiene un crecimiento lineal de 0 a 3.1V en un tiempo aproximado de 1.9ms y de decrecimiento de 0.45ms.

A través de este es posible la medición de voltajes análogos mediante comparadores LM339 y las capacidades de captura de pulso que ofrece el microprocesador Rabbit 3800.

2.2.5 PLACA

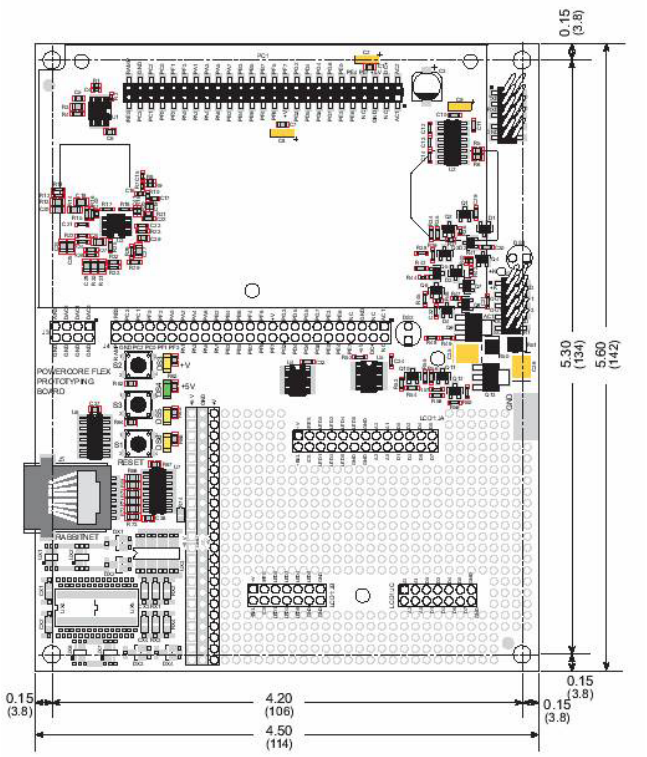


Figura 2.8 Vista superior de la placa

La placa incluye un kit de desarrollo que provee de algunas entradas y salidas como de periféricos (RS-232, triacs. Leds y switches), además de un área para realizar el desarrollo e incorporación de hardware. El prototipo se puede utilizar sin modificaciones, sin embargo es posible realizar cambios de acuerdo a las exigencias y necesidades previstas por el diseñador [12].

2.2.5.1 DISPOSICIÓN DE ELEMENTOS

La disposición de los principales elementos en la placa en la figura 2.9, mayores detalles revisar [13].

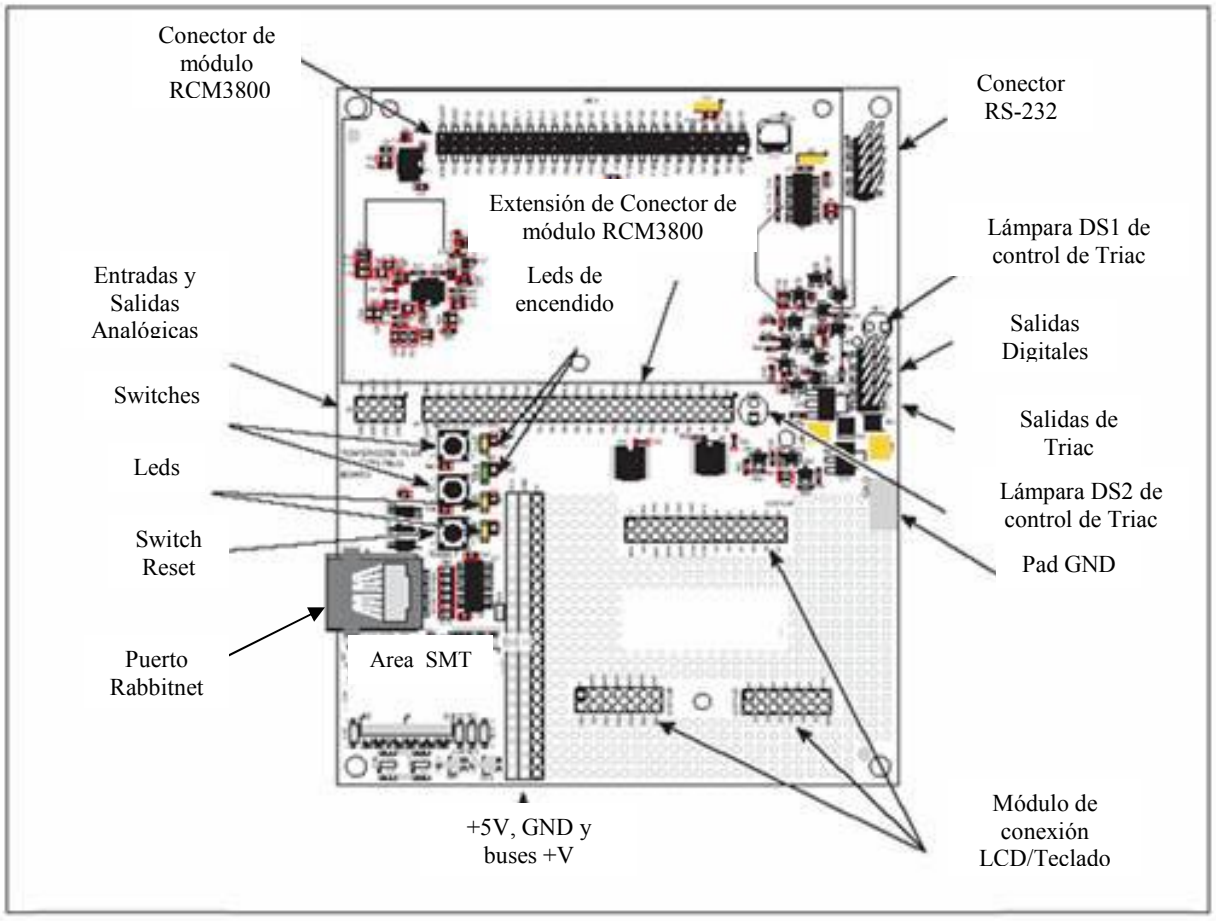


Figura 2.9 Disposición de elementos en la placa

Conexión de Encendido: Los niveles de voltaje de +5V y +3.45V son suministrados a la placa por el módulo PowerCore. Estos voltajes pueden ser utilizados para la alimentación de las partes colocadas por el diseñador dentro del área de la placa.

Encendido de Leds: Corresponden a los leds DS3 y DS4, los mismos que se encienden con cualquiera de los niveles de voltaje dados por la fuente (+3.45V o +5V).

Área del Prototipo: Dispone de una gran área, la cual permite la instalación de componentes alimentados con +3.45V y +5V y buses de tierra los cuales se disponen a lo largo de esta área.

Switch Reset: Permite el reinicio del sistema, a través del contacto de un switch normalmente abierto que va directamente al pin RESET_IN del módulo PowerCore.

Switches de Entrada/Salida y Leds: Dos switches normalmente abiertos son conectados a los pines PC3 y PG4 del módulo PowerCore y deben ser leídos como entradas para aplicaciones sencillas.

Dos leds (DS5 y DS6) son conectados a los pines de entrada/salida PD5 y PC2 del módulo PowerCore.

Conector de Extensión del Módulo: El pin set del módulo PowerCore es duplicado en el conector J4, con lo cual el usuario tiene la posibilidad de soldar cables directamente en los agujeros apropiados.

Entradas y Salidas Digitales: Cuatro salidas digitales y una conexión +K que son disponibles en el conector J4.

Triacs: Dispone de 2 triacs Z0107MN, que pueden ser utilizados para aplicaciones de switch de propósito general AC, tales como electroválvulas, bombas, entre otras. Sus salidas se encuentran en el conector J2.

RS-232: Tiene 4 puertos seriales y 2 puertos seriales RS-232 dispuestos en el conector J1.

Puerto Rabbitnet: Dispone de un puerto RS-422 RabbitNet mediante el cual es posible la conexión de tarjetas periféricas a la placa.

2.2.5.2 DISPOSICION DE PUERTOS EN LA PLACA

Los puertos indicados se disponen de la siguiente manera en la placa, con lo que se pude tener un manejo ordenado y práctico de cada uno de ellos, optimizando espacio dentro de la tarjeta [14].

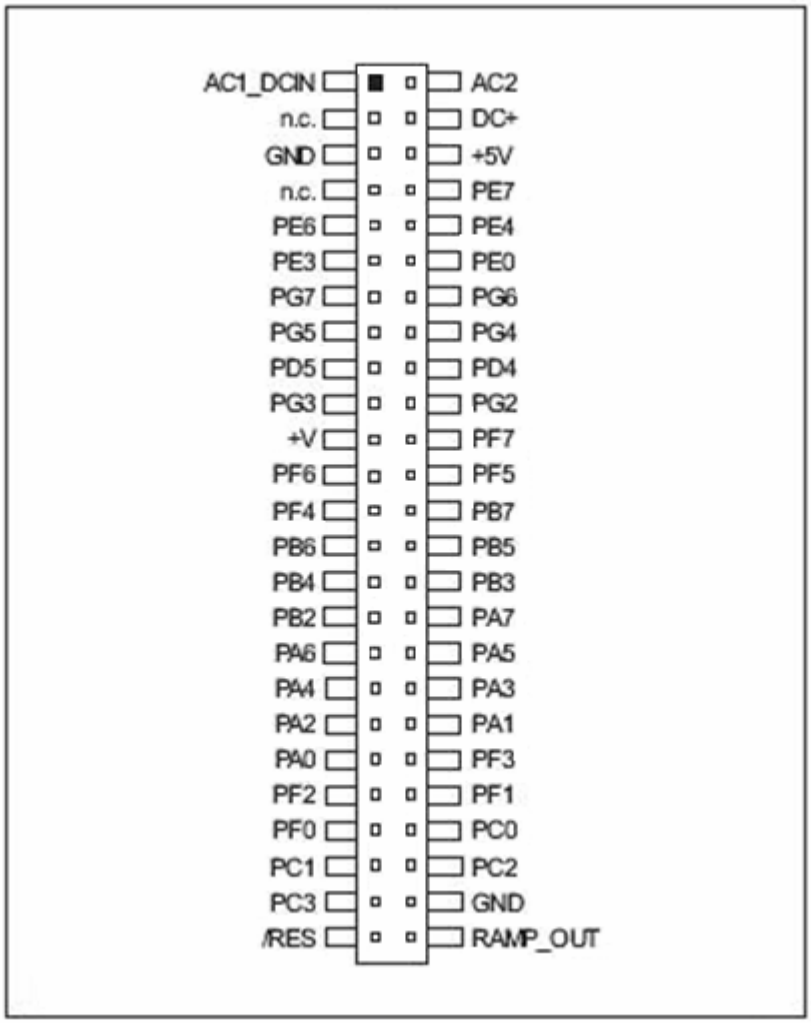


Figura 2.10 Disposición de puertos en la placa

2.3 SISTEMA ELECTRICO DE CONTROLADOR

2.3.1 DESCRIPCIÓN DE COMPONENTES

El principal dispositivo constituye el controlador Rabbit, el mismo que recibe señales en su entrada y envía señales eléctricas de salida hacia un circuito de disparo, a través del cual se realiza su acoplamiento para activar finalmente cada una de las unidades electrónicas de los semáforos. Para ello es necesario el acondicionamiento de la señal, debido a que los niveles de voltaje y corriente requeridos varían en cada una de las etapas.

2.3.2 RED ELÉCTRICA Y FUENTE DE ALIMENTACIÓN

La red eléctrica suministra un nivel de tensión de 121 Vac y 60 Hz, por lo que es necesario el acondicionamiento de esta señal, lo cual se realiza mediante su rectificación y regulación. Este acondicionamiento es necesario debido a que el controlador requiere una alimentación diferente a la de la red pública.

2.3.3 CIRCUITO DE DISPARO

Este permite enviar la señal de 121 Vac a partir de la activación de la salida del controlador hacia cada una de las electrónicas de los semáforos, dispuestas en la intersección, las cuales requieren una alimentación en su entrada.

2.3.4 ELECTRÓNICA DE SEMAFORIZACIÓN

Los circuitos electrónicos de las unidades de semaforización reciben las señales enviadas por el controlador mediante el circuito de disparo, a partir de lo cual realizan el encendido de las matrices.

2.4 SISTEMA ELÉCTRICO DE SEMÁFOROS

2.4.1 DESCRIPCIÓN DE COMPONENTES

Cada una de las electrónicas de semaforización posee matrices de diodos led, las mismas que ofrecen la iluminación requerida para cada cruce de acuerdo a las normas establecidas por los organismos de control respectivos, en aspectos tales como dimensiones, alimentación, cantidad de diodos led, distribución, forma y luminosidad, por lo que es necesario realizar un riguroso diseño, tomando en cuenta cada parámetro indicado.

Además es necesaria la adaptación de las señales que son enviadas desde la unidad principal a través del circuito de disparo hacia los semáforos.

2.4.2 SEMAFORIZACIÓN VEHICULAR

Las matrices correspondientes a los semáforos vehiculares se componen de grupos de diodos led, los cuales varían en función de la prioridad de la vía, por lo que es necesario diferente cantidad para cada semáforo. A partir de ello se indica la siguiente tabla [15].

	Dimensión de lámpara (mm)			Cantidad de leds		
Color	Rojo	Amarillo	Verde	Rojo	Amarillo	Verde
Vía Principal	200	200	200	127	127	127
Vía Secundaria	200	200	200	127	127	127

Tabla 2.1 Consideraciones de Semaforización Vehicular

2.4.3 SEMAFORIZACIÓN PEATONAL

La semaforización peatonal se compone de grupos de diodos led en formas determinadas, ya que es necesario que el usuario o peatón tenga una apreciación práctica del modo de utilización del sistema, evitando confusión y posibles accidentes.

Por lo que el diseño de figuras fijas y animadas brinda un servicio de mayor seguridad que los sistemas tradicionales. En la tabla 2.2. se muestra las consideraciones indicadas [16].

Color	Dimensión (mm)	Figura
Rojo	200	Fija: Hombre detenido
Verde	200	Animada: Hombre caminando

Tabla 2.2 Consideraciones de Semaforización Peatonal

2.5 ADAPTACIÓN

La señal enviada por el circuito de disparo correspondiente a cada una de las señales de control realiza la activación de un relé de 110V AC, el cual, mediante una fuente de voltaje, alimenta a cada una de las matrices de leds. Se debe tomar en cuenta que es necesaria únicamente una fuente de alimentación para las tres matrices implementadas en cada semáforo, debido a que en ningún momento se activan las tres a la vez, por lo que es posible la optimización de espacio y costo.

Considerando el diseño del sistema, la alimentación requerida por las matrices, es de 24V DC y 2A, la cual es suministrada por una fuente de voltaje fija. Esta fuente esta compuesta de un transformador de 121V AC a 24V AC, un puente de diodos que rectifica la señal a 24 V DC, para ser luego ser filtrada mediante el uso de capacitores, obteniendo finalmente a la salida el suministro de voltaje necesario para cada una de las matrices.

CAPITULO III

COMUNICACIÓN Y SOFTWARE

3.1 INTRODUCCIÓN

Dentro del desarrollo del proyecto de diseño e implementación del sistema de control centralizado de flujo vehicular y peatonal, la comunicación, constituye un aspecto imprescindible para coordinar, controlar y compartir datos. Es por ello que se requiere establecer una comunicación sólida entre cada uno de los dispositivos, garantizando de este modo el funcionamiento del sistema, confiabilidad en la información y seguridad en el tipo de datos adquiridos.

Para obtener un buen desempeño durante la transmisión y recepción de datos, es necesario considerar cada aspecto relacionado con ésta, tales como protocolos, normas y estándares. Además es necesario considerar los modos de comunicación que ofrece la tarjeta Rabbit, tanto en hardware como en software, ya que a partir de ello es posible alcanzar las metas del proyecto, optimizando.

3.2 PROTOCOLOS DE COMUNICACIÓN

Los protocolos de comunicación son considerados como reglas que han sido establecidas por organismos internacionales con el fin de permitir el flujo de información e incluso órdenes entre computadores o dispositivos que formen parte de la red. [17]. Estos pueden ser implementados tanto en software como en hardware o en su combinación, es decir, que cualquier dispositivo que utilice un protocolo dado debería poder funcionar con otros que utilicen el mismo protocolo.

Para ello se requiere tomar en cuenta tres aspectos fundamentales, a partir de los cuales los protocolos se desarrollan, estos son:

- Semántica, es decir el significado de lo que se comunica.
- Sintaxis, corresponde al modo en que se expresa.
- Sincronización, indica quién transmite y cuándo lo hace.

Existen características técnicas, que los protocolos, en su gran mayoría los considera, entre los cuales se pueden indicar los siguientes:

- Detección de la conexión física sobre la que se realiza la conexión.
- Procedimiento para establecer la comunicación.
- Negociación de las características de la conexión.
- Modo de inicio y fin del mensaje.
- Formato de los mensajes.
- Tratamiento de mensajes con errores. Detección y corrección.
- Modo de detección de pérdida de conexión
- Terminación de la sesión de conexión.

Así, el módulo PowerCore Flex permite la comunicación con otros dispositivos bajo el protocolo TCP/IP, con lo cual es posible el intercambio de datos entre cada intersección y la estación central.

3.2.1 TCP/IP

Descripción

Hace referencia al protocolo de control de transmisión, el mismo que se basa en un lenguaje que rige la comunicación entre computadores o dispositivos que formen parte de la red, permitiendo el envío de paquetes de información a diferentes destinos, para lo cual requiere verificar y ordenar los paquetes en su llegada.

En caso de no haber sido recibido en el sitio de destino correctamente, este es enviado nuevamente hasta superar el inconveniente. El sitio de destino de los diferentes paquetes es indicado mediante el Protocolo de Internet (IP) [18].

Capas

Existen varias capas mediante las cuales se realiza la transmisión y recepción de datos dentro de la red; sin embargo estas no actúan por si solas y requieren de protocolos de comunicación, mismos que permiten el entendimiento entre cada una, estableciendo de este modo la comunicación entre cada host.

De este modo, se realiza el envío del software entre capas a través de la recepción de información almacenada en las cabeceras. Cada capa añade su cabecera en la parte delantera del mensaje. La cabecera es retirada por parte de su correspondiente capa en el lado receptor. La figura 3.1 muestra las diferentes capas del modelo OSI.

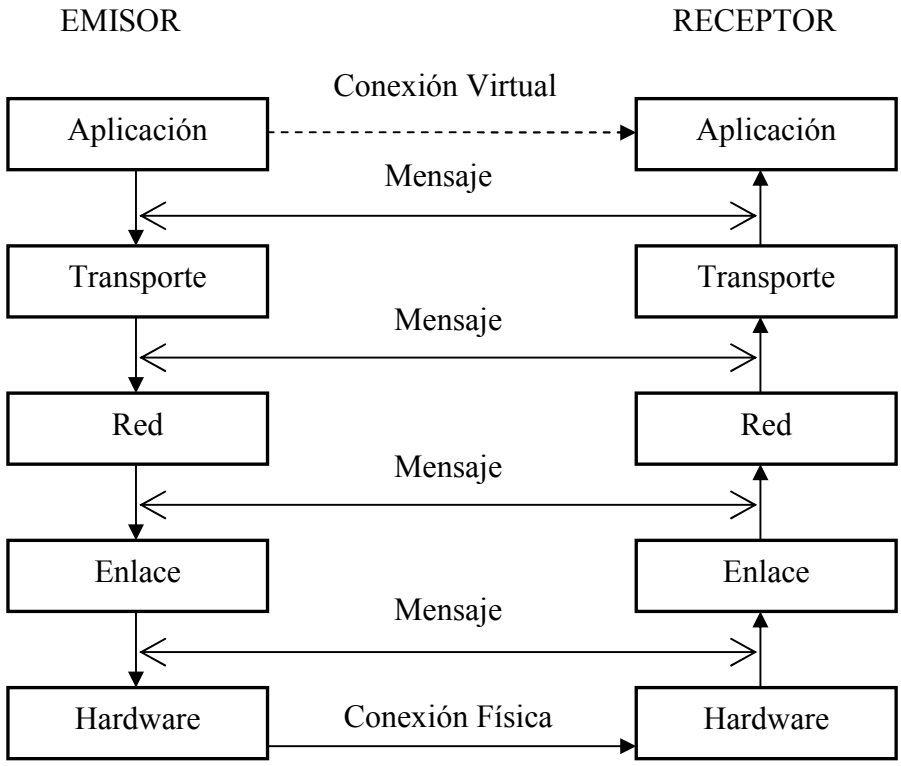


Figura 3.1 Modo de Comunicación entre capas de red

- **Capa de Aplicación:** Permite la interconexión del usuario u operador dentro de la red.
- **Capa de Transporte:** Define los servicios de segmentación, transferencia y reensamblaje de datos entre dos dispositivos finales.
- **Capa de Red:** Provee servicios para el intercambio de partes de datos sobre la red entre los identificadores de los dispositivos finales, es decir de la tarjeta de control Rabbit 3800.
- **Capa de Enlace:** Indica los métodos en el intercambio de tramas entre los dispositivos sobre el medio de comunicación.
- **Capa de Física:** Describe el funcionamiento mecánico y eléctrico, además de activar, mantener y desactivar las conexiones físicas para la transmisión de bits con el dispositivo final.

Ethernet

Está definido por la capa de Enlace de Datos y la Física de los protocolos. La figura 3.2 muestra su posición dentro del modelo OSI.

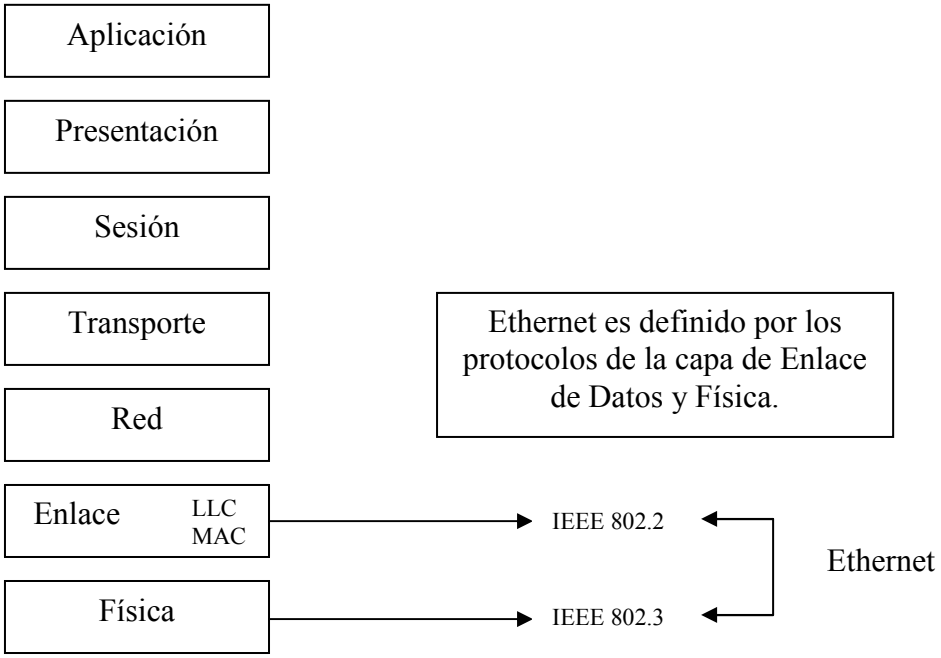


Figura 3.2 Capa de Ethernet en el modelo OSI

Ethernet es un estándar, a través del cual se realiza la comunicación a nivel físico, para lo que es necesario la transmisión de información mediante tramas de datos, las mimas que deben ser configuradas correctamente.

Mediante esta trama es posible el envío de datos sobre el medio físico, para lo cual se requiere de la dirección de destino y de la detección de error. La trama está formada por tres grupos llamados Header, Packet y Trailer, integrados por un conjunto de bytes que realizan una función específica. La figura 3.3 muestra a cada uno de los grupos de la trama y su ubicación durante la transmisión de información.

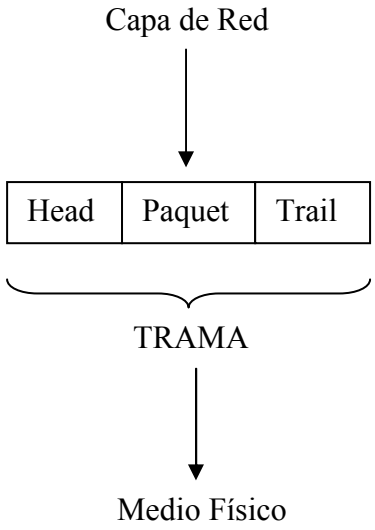


Figura 3.3 Grupos de la Trama de Ethernet

El Header indica el inicio de la trama, la dirección, el tipo y la calidad de control; el Packet contiene los datos que van a ser transmitidos; y el Trailer se encarga de la detección de errores y del bit de parada [19]. La trama de Ethernet, con cada uno de sus componentes esta descrita en la tabla 3.1.

PREAMBULO	SOF	DESTINO	ORIGEN	TIPO	DATOS	FCS
7 bytes	1 byte	6 bytes	6 bytes	2 bytes	46 a 1500 bytes	4 bytes

Tabla 3.1 Trama de Comunicación Ethernet

Preámbulo

Campo de 7 bytes (56 bits) con una secuencia de bits usada para sincronizar y estabilizar el medio físico antes de iniciar la transmisión de datos. El patrón del preámbulo es:

10101010 10101010 10101010 10101010 10101010 10101010 10101010

Estos bits se transmiten en orden, de izquierda a derecha y en la codificación Manchester representan una forma de onda periódica.

SOF (Start Of Frame - Inicio de Trama)

Campo de 1 byte (8 bits) con un patrón de 1s y 0s alternados y que termina con dos 1s consecutivos. El patrón del SOF es: 10101011. Este indica que el siguiente bit será el más significativo del campo de dirección MAC de destino. Aunque se detecte una colisión durante la emisión del preámbulo o del SOF, el emisor debe continuar enviando todos los bits de ambos hasta el fin del SOF.

Dirección de destino

Campo de 6 bytes (48 bits) que especifica la dirección MAC hacia la que se envía la trama. Esta dirección de destino puede ser de una estación, de un grupo multicast o la dirección de broadcast de la red. Cada estación examina este campo para determinar si debe aceptar el paquete.

Dirección de origen

Campo de 6 bytes (48 bits) que especifica la dirección MAC desde la que se envía la trama. La estación que deba aceptar el paquete conoce por este campo la dirección de la estación origen con la cual intercambiará datos.

Tipo

Campo de 2 bytes (16 bits) que identifica el protocolo de red de alto nivel asociado con el paquete o, en su defecto, la longitud del campo de datos. La capa de enlace de datos interpreta este campo.

Datos

Campo de 46 a 1500 Bytes de longitud. Cada Byte contiene una secuencia arbitraria de valores. El campo de datos es la información recibida del nivel de red. Este campo, también incluye los Header 3 y Header 4 (cabeceras de los niveles 3 y 4), provenientes de niveles superiores.

FCS (Frame Check Sequence - Secuencia de Verificación de Trama)

Campo de 32 bits (4 bytes) que contiene un valor de verificación CRC. El emisor calcula este CRC usando todo el contenido de la trama y el receptor vuelve a calcularlo y luego lo compara con el recibido a fin de verificar la integridad de la trama.

La trama Ethernet es de una longitud variable entre 64 y 1518 octetos (encabezado, datos y CRC). Como en todas las redes de conmutación de paquetes, cada trama Ethernet contiene un campo con la información de la dirección de destino. La figura 6 muestra que la trama Ethernet contiene la dirección física de la fuente y también la dirección física del destino [20].

Después de que es transmitida una trama, todos los dispositivos de la red compiten por la siguiente oportunidad de transmitir una trama. La disputa por la oportunidad de transmitir entre los dispositivos es pareja, para asegurar que el acceso al canal de comunicaciones sea justo, ningún dispositivo puede bloquear a otros dispositivos.

El acceso al canal de comunicaciones compartido es determinado por la subcapa MAC. Este control de acceso al medio es conocido como CSMA/CS [21]. La transmisión de datos se basa en tramas de Ethernet, para lo cual es necesario establecer una topología, sea esta tipo bus o anillo.

Control de Acceso al Medio

El método utilizado por Ethernet es CSMA/CD, con Detección de Colisiones, en el cual se busca compartir los recursos [22]. Asegurando que únicamente un nodo de red se transmita en cualquier momento.

Mediante este proceso, Ethernet "escucha" en el medio físico antes de intentar transmitir, en caso de que esté ocupado este, espera un tiempo aleatorio y vuelve a intentarlo (Carrier Sense); además permite escuchar y transmitir a la vez (Multiple Access); en caso de que múltiples dispositivos transmitan al mismo tiempo, es posible la detección del error (Collision Detection) [23].

Estándar RJ-45

Su acrónimo es Registered Jack, es regulado por el Código Federal de Regulaciones de Estados Unidos; constituye una interfaz física, es decir es un medio utilizado para la conexión de un computador o dispositivo de control, en este caso la tarjeta Rabbit, con el medio de transporte de la red. Es usado además para conectar redes de cableado estructurado, (categorías 4, 5, 5e y 6).

Consta de 8 conexiones eléctricas, que normalmente se usan como extremos de cables de par trenzado, es utilizado con estándares TIA/EIA-568-B, la misma que define la disposición de los pines; es empleado en cables de red Ethernet [24]. Su disposición se muestra a continuación.

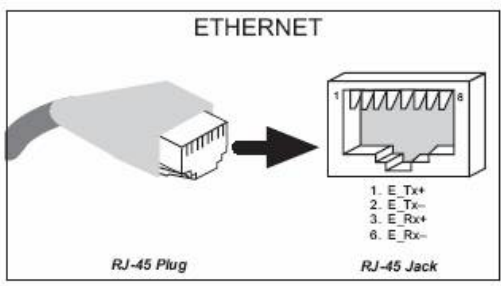


Figura 3.4 Conector de Salida Ethernet

3.2.2 COMUNICACIÓN SERIAL

El módulo PowerCore Flex ofrece comunicación serial a través de sus puertos que se basan en el estándar RS-232, RS-422, RS-485. El estándar RS-232 es usado para la transmisión de datos en el modo local hacia el LCD el mismo que se encuentra colocado en las intersecciones de interés [25]. Estos estándares se indican a continuación.

RS-232

Corresponde al estándar ANSI/EIA-232, éste conector permite la transmisión de datos a través del puerto serial de forma asíncrona. Tiene múltiples aplicaciones por lo que sean desarrollado líneas y dispositivos de transmisión de mayor velocidad y que permiten mayor distancia. Sin embargo está limitado a comunicaciones de punto a punto entre los dispositivos y el puerto serial de la computadora El hardware de RS-232 se puede utilizar para comunicaciones seriales en distancias de hasta 15 metros.

Se envían datos de 7, 8 o 9 bits. La velocidad se mide en baudios (bits/segundo) y sólo son necesarios tres cables, correspondientes a la transmisión y recepción y tierra (GND). Es imprescindible la comprensión de la función que cumple cada pin tanto de entrada como de salida de datos durante la comunicación, ya que existen dos tipos de conectores para este estándar, los de 25 pines y los de 9 pines, siendo éste último el más común, a pesar de que la de 25 permite la transferencia de una mayor cantidad de datos.

Las señales con la que actúa el puerto son digitales (0 - 1) y la tensión a la que trabaja es de 12 Voltios, es decir 12 [V] corresponde a un “0” lógico, mientras que -12[V] indica un “1” lógico.

Las características de los pines y su nombre típico son:

- **TXD** Transmitir Datos Señal de salida.
- **RXD** Recibir Datos Señal de entrada.
- **RTS** Solicitud de envió Señal de salida.
- **DTR** Terminal de datos listo Señal de salida.

- **CTS** Libre para envió Señal de entrada
- **DSR** Equipo de datos listo Señal de entrada
- **DCD** Detección de portadora Señal de entrada
- **SG** Tierra Referencia para señales
- **RI** Indicador de llamada Señal de entrada
- **RTxC** Reloj de Recepción / Transmisión.

La configuración de los pines para cada uno de los tipos de conectores se muestra en la tabla 3.2

Conector 25 pines	Conector 9 pines	Nombre
1	1	-
2	3	TxD
3	2	RxD
4	7	RTS
5	8	CTS
6	6	DSR
7	5	SG
8	1	DCD
15	-	TxC
17	-	RxC
20	4	DTR
22	9	RI
24	-	RTxC

Tabla 3.2 Configuración de conectores DB9 y DB25 (Estándar RS-232)

Antes de iniciar cualquier comunicación con el puerto RS232 se debe de determinar y configurar el protocolo a utilizar.

Siendo los parámetros a configurar los siguientes:

- Protocolo serie (numero bits-paridad-bits stop).
- Velocidad de puerto.
- Protocolo de control de flujo (RTS/CTS o XON/XOFF).

Para realizar la verificación, pruebas de comunicación y visualización de las señales es necesario un programa que permita gestionar la comunicación entre la tarjeta Rabbit con módulo PowerCore Flex y el computador. Esto es posible realizarlo a través del Hyper Terminal, el cual reúne las características requeridas.

RS-422

Corresponde al estándar EIA RS-422-A. Este conector serial utiliza señales eléctricas diferenciales, en comparación con señales referenciadas a tierra como en RS-232. La transmisión diferencial, que utiliza dos líneas para transmitir y recibir, tiene inmunidad al ruido y puede trabajar a mayores distancias que RS-232.

Siendo importante su consideración, por cuanto dentro de aplicaciones industriales es fundamental el mejoramiento de parámetros tales como inmunidad al ruido o el alcance de mayores distancias, estableciendo de este modo una comunicación fiable.

3.3 COMUNICACIÓN CON LA TARJETA ELECTRÓNICA RABBIT

Existen dos formas consideradas para la comunicación con el controlador, es decir con la tarjeta embebida Rabbit; la primera es mediante la red Ethernet, la misma que sirve para realizar el control de modo remoto; mientras que la segunda es serial y tiene como fin la comunicación del modo local. Las figuras 3.5 y 3.6 indican de forma gráfica estos dos tipos de comunicación.

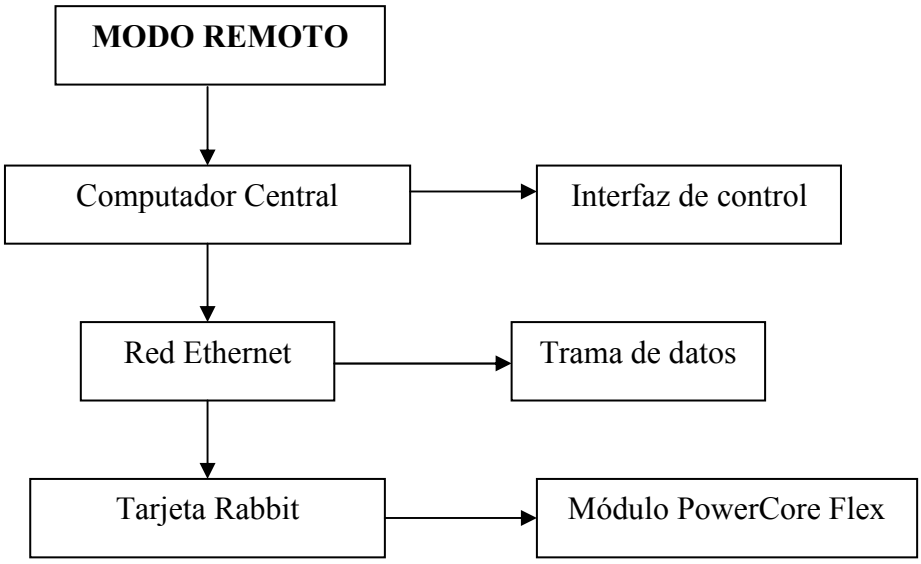


Figura 3.5 Esquema de Comunicación mediante Red Ethernet

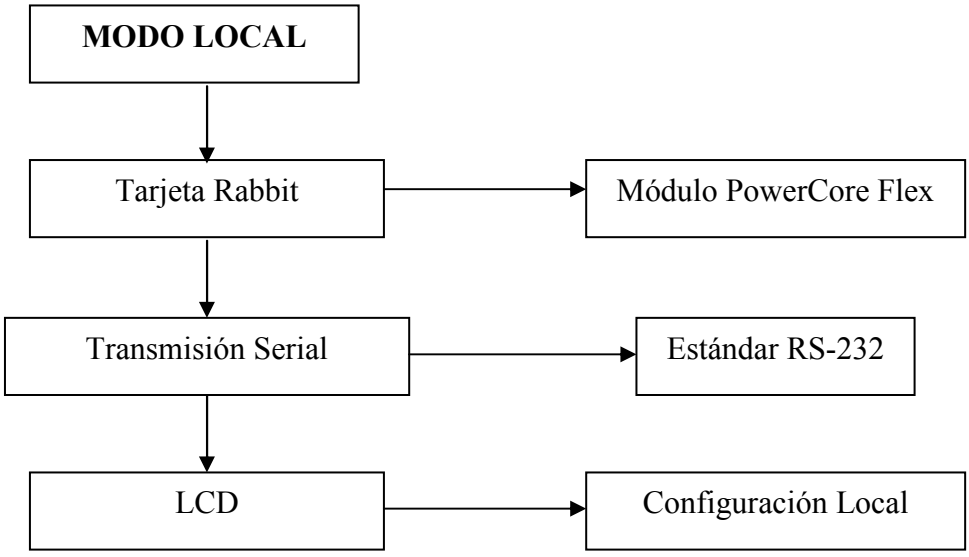


Figura 3.6 Esquema de Comunicación Serial

Para establecer la comunicación, a través de la red Ethernet y serial es necesario desarrollar la programación respectiva mediante Dynamic C.

3.3.1 RED ETHERNET

La conexión considerada para el proyecto de semaforización es entre un ordenador, quien realizará el monitoreo del sistema y la tarjeta Rabbit con módulo PowerCore Flex, encargada del control, de modo que es necesario utilizar un cable UTP categoría 5, punto a punto para Ethernet, conectando cada uno de sus extremos a los dispositivos mediante la utilización de un Switch.

Es importante citar que los modos de conexión indicados permiten la transmisión de datos dentro de una red de área local (LAN).

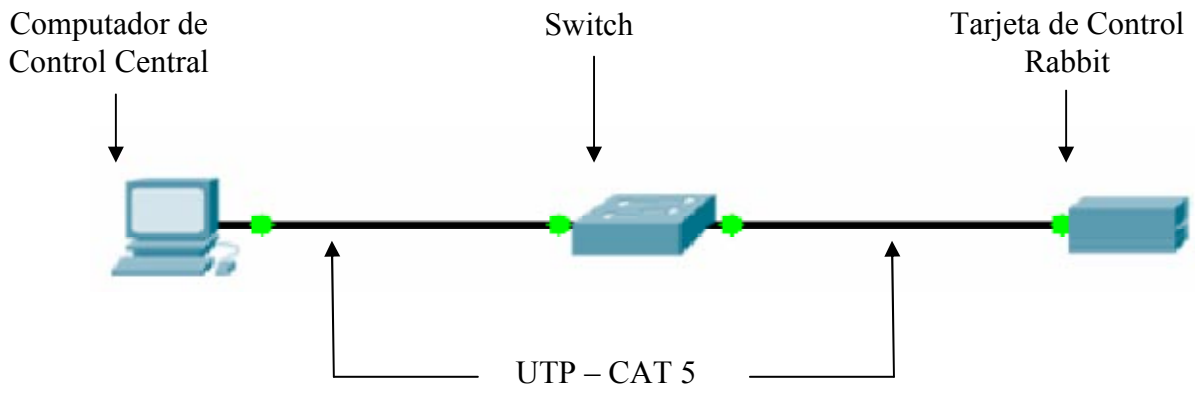


Figura 3.7 Conexión de Red Ethernet

La versión Dynamic C utilizada para realizar la programación del sistema, ofrece un completo paquete de librerías, las mismas que permiten el desarrollo de aplicaciones, de este modo cada una de ellas tienen una función específica, por lo que para establecer el intercambio de información mediante el protocolo TCP/IP estas deben ser consideradas.

SOCKET TCP

Los drivers TCP/IP están localizados en la carpeta **LIB\TCPIP.** Para la versión de Dynamic C 6.57 y superiores, cada socket debe estar asociado a *tcp_Socket* de 145 bytes. Los buffers de entrada y salida se encuentran en memoria extendida [26].

APERTURA DEL SOCKET

En este modo, cuando se requiere establecer contacto con el dispositivo controlador, es necesario abrir el socket con *tcp_listen()*. Esta forma es comúnmente utilizada para servidores Internet que escuchan en un puerto conocido, como el 80 para HTTP.

Es necesario suministrar *tcp_listen()* con un puntero de estructura *tcp_Socket* y la dirección IP viene dada por la tarjeta. Se debe configurar con 0 (cero), en caso de que se requiera realizar la conexión desde cualquier dirección IP.

Para manejar conexiones simultaneas múltiples, cada nueva conexión requerirá su propia estructura *tcp_Socket* y una llamada separada a *tcp_listen,* pero usando el mismo número de puerto local.

La llamada *tcp_listen()* retornará inmediatamente y se debe verificar la conexión entrante. Es posible el uso de la macro *sock_wait_established,* con la cual se llamará a *tcp_tick()* y bloqueará hasta que la conexión sea establecida o se puede probar manualmente el socket utilizando *sock_established.*

Funciones TCP Socket

Existen diversas funciones que pueden ser aplicadas a *tcp_Socket*, entre las cuales se encuentran las de Control, las cuales han sido consideradas durante el desarrollo del proyecto. Además de *tcp_open* y *tcp_listen()* , también está *sock_close()* que debe ser llamada cuando se espera finalizar una conexión. Una llamada a *sock_close()* no cierra inmediatamente la conexión, porque esta toma cierto tiempo en enviar la solicitud de finalización y recibir el reconocimiento. Si se asegura el cierre completo de la conexión, es posible realizar el llamado de la función *tcp_tick()* con la dirección del socket. Cuando esta función retorna un 0 (cero), indica que el socket esta completamente cerrado. Cabe indicar que si existe un dato saliendo para ser leído en el socket, éste no se cerrará completamente.

Hay varias causas para cancelar una conexión, o incluso hay ocasiones que es necesario volver a establecerla, para lo cual se llama a la función *sock_abort()*. Esta función reseteará al TCP y los paquetes enviados posteriormente serán ignorados [27].

3.3.2 SERIAL

Para la comunicación serial Dynamic C ofrece un rango muy amplio de soporte. Existen librerías como *RS232.LIB*, que provee un conjunto de buffer circular basado en funciones seriales. La librería *PACKET.LIB* provee funciones seriales basadas en paquetes, en donde estos paquetes pueden ser delimitados por el noveno bit, por brechas de transmisión, o pueden ser configuradas de acuerdo a la aplicación.

Estas dos librerías proveen el bloqueo de funciones, la cual no retorna hasta que la transmisión o recepción ha finalizado, permitiendo que otras funciones se desempeñen entre llamadas.

La función *voidserMode(int mode)* permite desarrollar líneas de comunicación serial para el módulo PowerCore Flex. Es necesario para esto, llamar a la función *serXpen()* por cualquiera de los puertos seriales. Cabe recordar que estos puertos son el E y F.

El modo se determina mediante el valor 0 o 1. La tabla 3.3 resume su configuración.

Modo	Puerto Serial	
	E	F
0	RS-232, 3 cables	RS-232, 3 cables
1	RS-232, 5 cables	RTS/CTS

Tabla 3.3 Configuración de Comunicación Serial bajo Dynamic C

3.4 SOFTWARE

La comunicación mediante los modos indicados y a través de la plataforma de programación de Dynamic C permite que el sistema desarrollado tenga el desempeño esperado, sin embargo el operador responsable del monitoreo, no requiere tener control sobre esto, por el contrario él debe manejar información confiable y actualizada proveniente del sistema implementado en cada intersección, para lo cual se requiere del desarrollo de una base de datos, la misma que será manejada a través de una interfaz, y que debe constar de todas las funciones que ofrece el sistema brindando un control práctico del mismo.

CAPITULO IV

DISEÑO DEL SISTEMA DE CONTROL DE TRÁFICO

4.1 INTRODUCCIÓN

El diseño del sistema de control constituye la parte fundamental del proyecto, debido a que representa la estrategia que permite desarrollar y construir su solución. Esto se lleva a cabo a través de la asignación de sistemas y subsistemas, con sus respectivos componentes de hardware y software.

Para llevar a cabo esto, fue necesario tener en cuenta los siguientes aspectos:

- Desglosar el sistema en subsistemas.
- Asignar los subsistemas a tareas.
- Diseño de Base de Datos.
- Implementación del software.

4.2 DISEÑO INTEGRAL

Para realizar el diseño del sistema es necesaria la consideración de características eléctricas de cada uno de los componentes y su disposición física. A continuación se describe de forma particular las características de cada uno de los subsistemas que lo componen.

4.2.1 UNIDAD PRINCIPAL

En la Unidad Principal, están contenidos los dispositivos de control y de potencia, la alimentación de la red eléctrica, la protección del sistema, así como también las tarjetas correspondientes al manejo y funcionamiento del modo local de configuración a través del LCD y del teclado matricial. La disposición de estos componentes se indica en la figura 4.1.

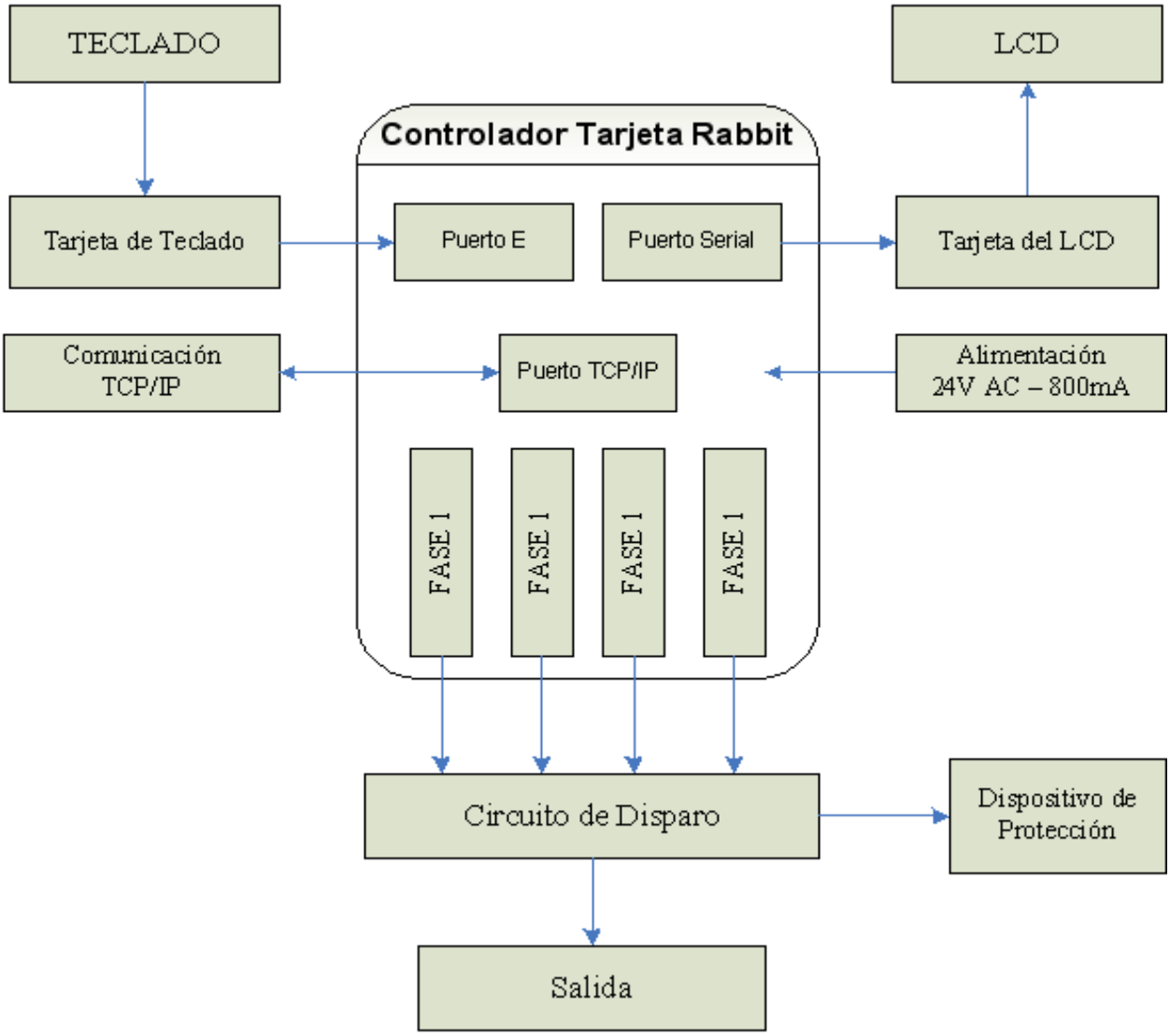


Figura 4.1 Diagrama de Bloques de la Unidad Principal

4.2.1.1 CONTROLADOR - TARJETA RABBIT

La tarjeta Rabbit realiza el control central del sistema, esta posee diversos recursos, mismos que han sido considerados dentro del diseño general. La disposición y uso esta determinado de acuerdo al número de señales a controlar. En la planeación y desarrollo del sistema de control de flujo vehicular y peatonal, se tomaron en cuenta el número de entradas y salidas de cada una de las fases, las mismas que corresponden a los puertos B y F del controlador. Para poder visualizar cada una de estas señales de control, se colocaron diodos led en la placa que viene incluida en el controlador. Además se especifica el tipo de comunicación para poder realizar el control del sistema mediante los dos modos de configuración, tanto remoto mediante el puerto TCP/IP y el local con el manejo del puerto serial [30].

4.2.1.2 DISPOSICIÓN DE PUERTOS Y FASES

La tabla 4.1 muestra la disposición de cada una de las fases con sus correspondientes puertos:

FASE	PUERTO
1	F0
1	F1
1	F2
2	F3
2	F4
2	F5
3	F6
3	F7
3	B2
4	B3
4	B4
4	B5

Tabla 4.1 Disposición de Fases y Puertos en Controlador Rabbit

4.2.1.3 CONFIGURACIÓN LOCAL

Para realizar el diseño del modo local de configuración se consideró la utilización tanto de un teclado matricial como de un LCD (Liquid Cristal Display), los cuales a través de sus circuitos permiten el ingreso y salida de datos, facilitando al administrador u operador la programación, en función de las circunstancias de cada intersección.

4.2.1.4 TARJETA DE TECLADO

Las señales ingresan desde el teclado matricial a la placa del Teclado a través de su decodificador, 74922 [35]. Las señales de salida son enviadas hacia el Controlador, el mismo que las recibe a través del puerto E. La figura 4.2 indica su modo de conexión (Ver Anexo A1).

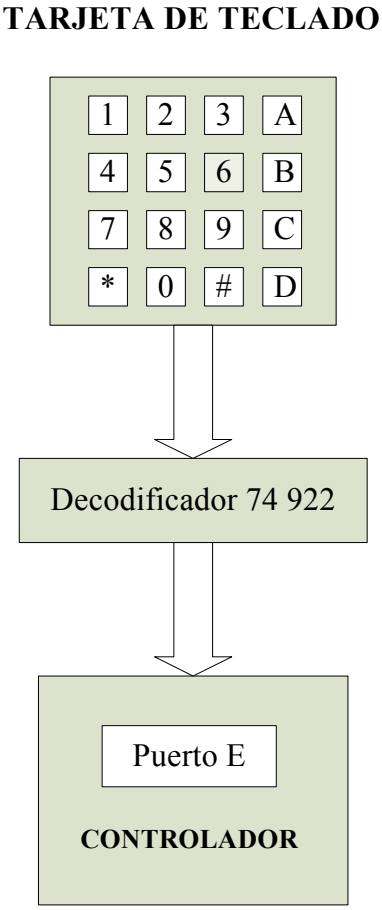


Figura 4.2 Disposición de Puertos del Controlador para Teclado

4.2.1.5 TARJETA DEL LCD

El LCD permite al operador del sistema la visualización de los cambios de configuración, para lo cual se ha desarrollado su presentación a través del menú, el mismo que facilita su uso, garantizando de este modo eficiencia y rapidez durante la programación de cada uno de las intersecciones [33].

Los datos mostrados en el LCD son enviados desde el controlador Rabbit mediante el puerto serial del controlador (puerto F) hacia el PIC 16F628A [34], el cual esta programado como decodificador, para luego ser enviados al Display (Ver Anexo A2).

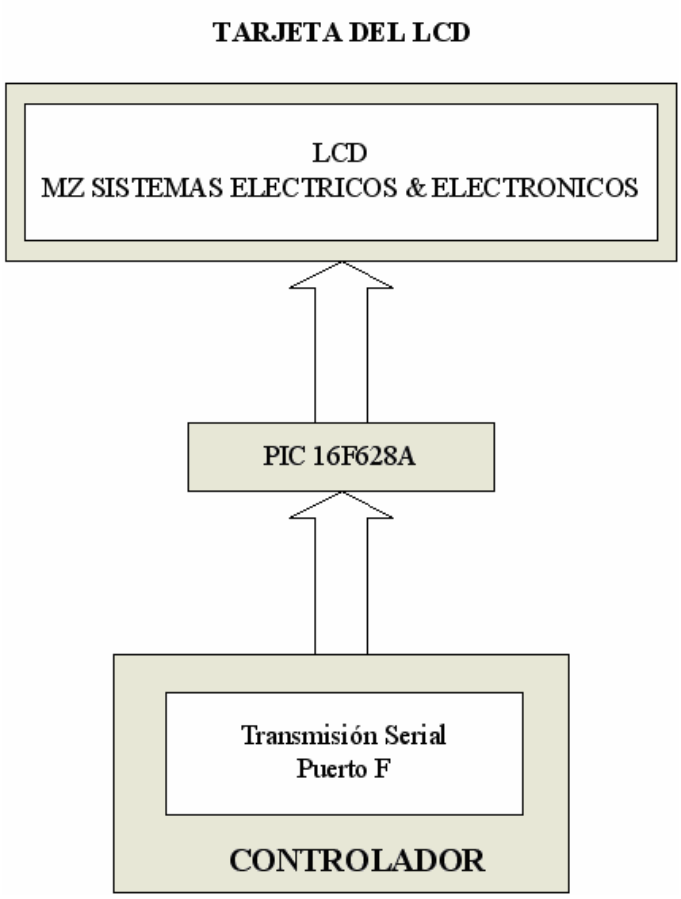


Figura 4.3 Disposición de Puertos del Controlador para LCD

4.2.1.6 SISTEMA DE PROTECCIÓN

DISPOSITIVO DE PROTECCIÓN

El dispositivo de protección de la Unidad Principal está en función de la corriente nominal de la carga, es decir de cada una de las luminarias. A partir de ellos se dimensiona considerando un 25% de valor adicional, debido a que se pueden generar transitorios que afecten el funcionamiento de los componentes que conforman el sistema.

El valor de corriente máximo de las luminarias es de 1.2 [A] por lo que el valor del breaker es de 1.5 [A]. Sin embargo es necesario ajustar a un valor que se encuentre en el mercado, por lo que se consideró uno de 2 [A].

PUESTA A TIERRA

El sistema de puesta a Tierra es muy importante dentro de toda instalación eléctrica, debido a que a través de esta es posible mantener altos niveles de seguridad del personal, operación de los equipos y su correcto desempeño, parámetros necesarios para garantizar el óptimo desarrollo y comportamiento de los sistemas.

Además, hay que tomar en cuenta la forma en que el sistema es conectado a tierra, por el efecto directo en la magnitud de los voltajes que deben ser mantenidos en condiciones normales y bajo condiciones transitorias.

Es necesario considerar un parámetro muy relevante dentro del análisis y diseño de puestas a tierra, la resistividad del electrodo, la misma que a su vez depende de factores tales como son la resistividad propia del electrodo (metal), la resistividad del contacto del electrodo con la tierra y la resistividad del suelo, es decir a partir de la superficie del electrodo hacia afuera, en el espacio por donde circula la corriente.

De este modo, en la implementación realizada el potencial propio del circuito se referencia a tierra mediante la utilización de una barra de Cobre, el cual es altamente conductivo y ofrece baja resistencia óhmica, permitiendo la descarga de cualquier señal externa, ajena a las consideradas dentro del diseño y que pudieran afectar su correcto

desempeño. Otro factor influyente es la de resistividad del terreno, el mismo que debe cumplir normas internacionales, para lo que la IEEE estableció a través del Estándar 142-1991 [28].

4.2.1.7 CIRCUITO DE DISPARO

Fue considerado el circuito de disparo debido a que la Unidad de Control Principal se encuentra alejada de los semáforos tanto vehiculares como peatonales, siendo, por lo tanto, la distancia un factor importante, ya que es posible la caída de tensión en el trayecto de la señal desde el controlador hasta cada una de los actuadores, causadas por la resistividad del cable.

Su diseño se realizó tomando en cuenta que la señal de entrada, proveniente de la tarjeta Rabbit es de 5V DC y que se requiere en su salida una tensión de 121V AC, los mismos que serán transmitidos sin inconvenientes a través del cable; para lo cual se utilizaron MOC [31] y Triac BJ136 [32], además de fusibles como dispositivos de protección.

Se elaboraron dos placas, correspondientes a cada una de las fases a controlar; en cada una de las cuales se colocaron 6 circuitos de disparo. La figura 4.4 muestra los circuitos de disparo para cada una de las fases (Ver Anexo A3).

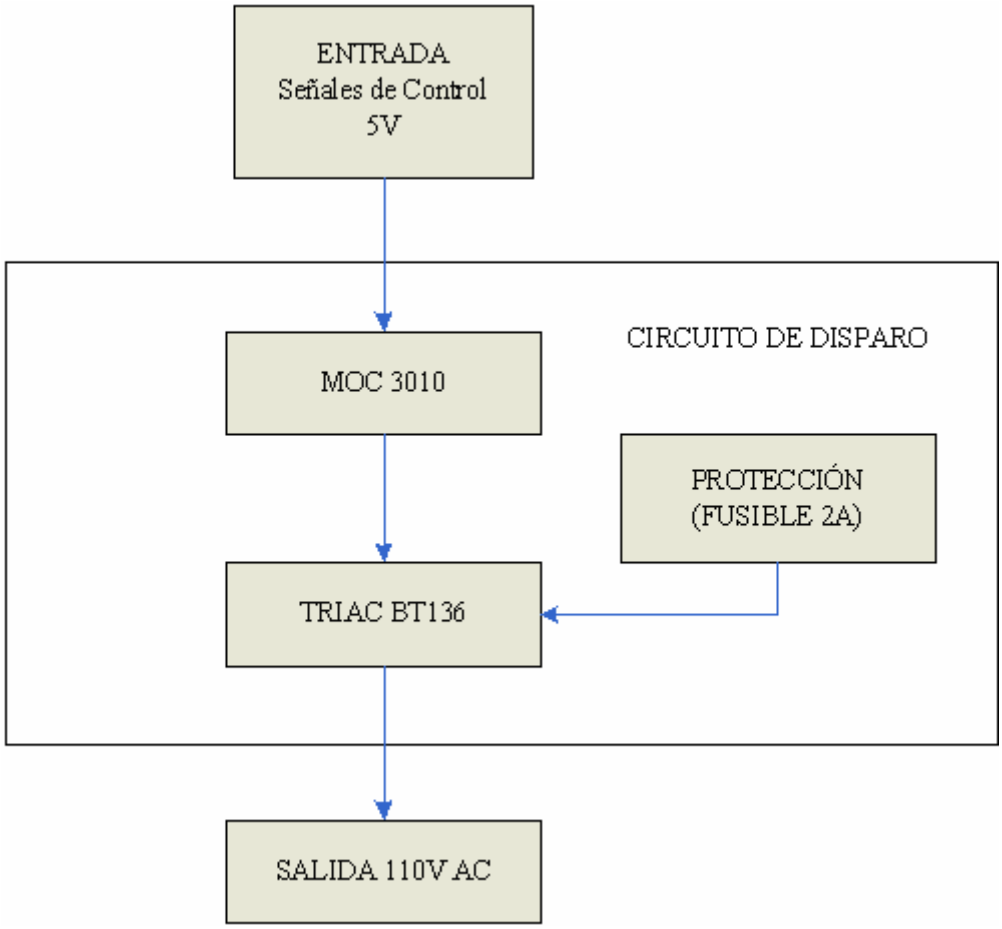


Figura 4.4 Diagrama de Bloques del Circuito de Disparo

4.2.2 SUBSISTEMA DE SEMAFORIZACION

4.2.2.1 SEMAFORIZACIÓN VEHICULAR

Las señales de 121V AC enviadas, desde la Unidad de Control Central a través del circuito de disparo llegan a la electrónica de los semáforos los mismos que se componen de relés de 110V AC de activación de bobina [36], mediante los que se habilitan las fuentes de alimentación de cada una de las luces, es decir depende de la señal de control enviada por el controlador y amplificadas por el circuito de disparo para que los actuadores se activen. En la figura 4.5 se muestra la disposición de los dispositivos (Ver Anexo B).

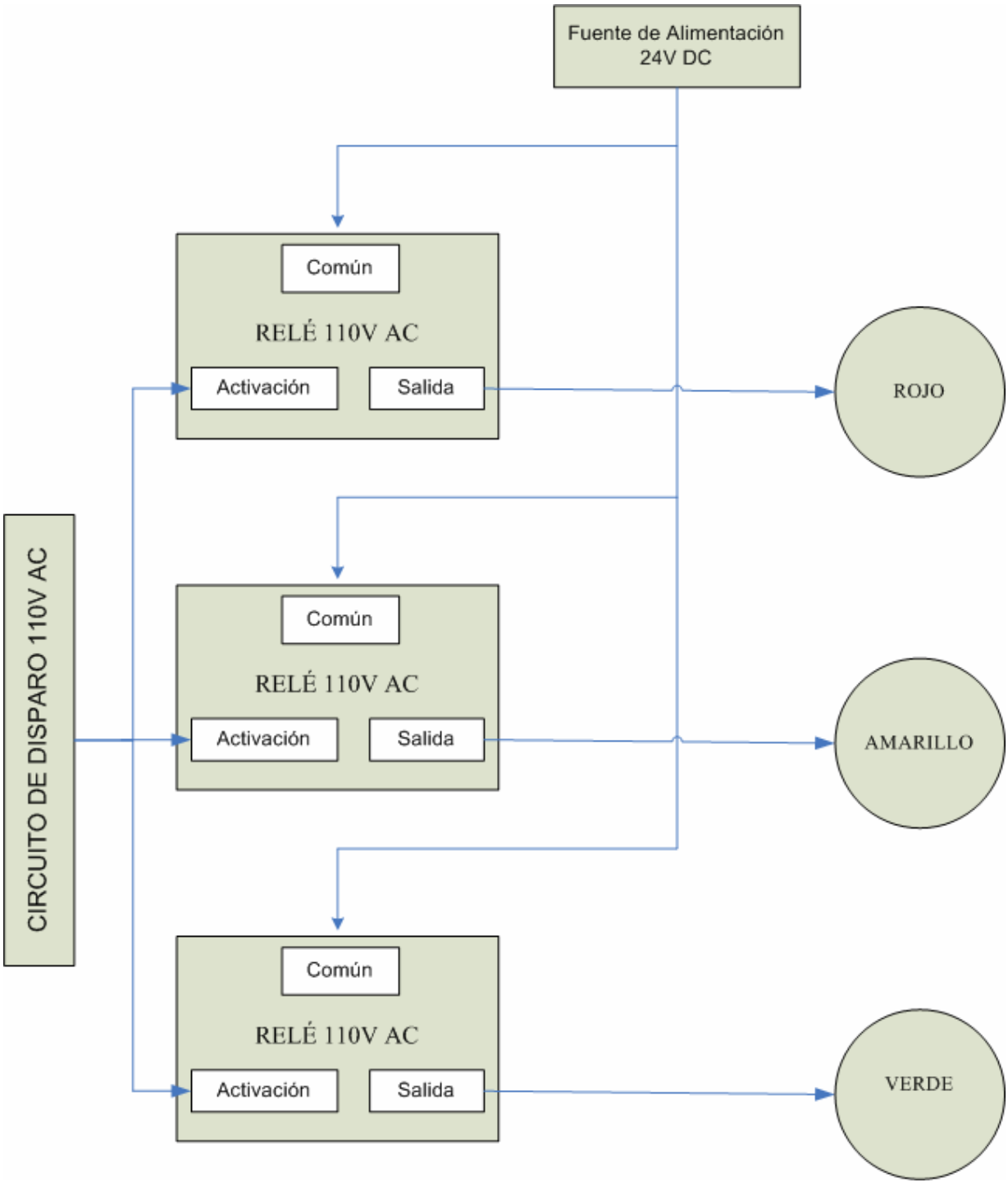


Figura 4.5 Diagrama del Subsistema de Semaforización Vehicular.

DISTRIBUCIÓN DE LEDS

La distribución de diodos led se realiza en función del voltaje que cada uno de las luminarias requiere, por lo que los grupos considerados para las luminarias son diferentes. El diseño es el siguiente:

Luminaria Roja

$$\text{Vfuente} = 24\,[\text{V}]\,\text{DC}$$
$$\text{Iconsumo} = 14\,[\text{mA}]$$
$$\text{Vled} = 2\,[\text{V}]$$

Por lo tanto considerando grupos de 10 leds, se obtiene:

$$\text{Vtotal} = 10(\text{leds}) * 2\,[\text{V}]$$
$$\text{Vtotal} = 20\,[\text{V}]$$

El cálculo de Resistencia para cada uno de los grupos es el siguiente:

$$\text{Voltaje} = \text{Corriente} * \text{Resistencia}$$
$$\text{Resistencia} = \frac{\text{Diferencia de Voltaje}}{\text{Corriente de consumo}}$$
$$\text{Resistencia} = \frac{\text{Vfuente - Vtotal}}{\text{Corriente de consumo}}$$
$$\text{Resistencia} = \frac{24\,[\text{V}] - 20\,[\text{V}]}{14\,[\text{mA}]}$$
$$\text{Resistencia} = 285.714\,[\Omega]$$

Realizando el sobredimensionamiento de 10% del valor calculo, se obtiene finalmente:

$$\text{Resistencia} = 285.714\,[\Omega] + 28.571\,[\Omega]$$
$$\underline{\text{Resistencia} = 314.285\,[\Omega]}$$

Por lo tanto, se consideró usar resistencias de 330 $[\Omega]$

Luminaria Amarilla

$$Vfuente = 24\,[V]\,DC$$
$$Iconsumo = 20\,[mA]$$
$$Vled = 2.03\ [V]$$

Por lo tanto considerando grupos de 9 leds, se obtiene:

$$Vtotal = 9(leds) * 2.03\,[V]$$
$$Vtotal = 18.27\,[V]$$

El cálculo de Resistencia para cada uno de los grupos es el siguiente:

$$Voltaje = Corriente * Resistencia$$
$$Resistencia = \frac{Diferencia\ de\ Voltaje}{Corriente\ de\ consumo}$$
$$Resistencia = \frac{Vfuente - Vtotal}{Corriente\ de\ consumo}$$
$$Resistencia = \frac{24\,[V] - 18.27\,[V]}{20\,[mA]}$$
$$Resistencia = 286.5\,[\Omega]$$

Realizando el sobredimensionamiento de 10% del valor calculo, se obtiene finalmente:

$$Resistencia = 286.5\,[\Omega] + 28.65\,[\Omega]$$
$$\underline{Resistencia = 315.15\,[\Omega]}$$

Por lo tanto, se consideró usar resistencias de 330 $[\Omega]$

Luminaria Verde

$$Vfuente = 24\,[V]\,DC$$
$$Iconsumo = 40\,[mA]$$
$$Vled = 2.3\,[V]$$

Por lo tanto considerando grupos de 10 leds, se obtiene:

$$Vtotal = 7\,(leds) * 2.3\,[V]$$
$$Vtotal = 16.1\,[V]$$

El cálculo de Resistencia para cada uno de los grupos es el siguiente:

$$Voltaje = Corriente * Resistencia$$
$$Resistencia = \frac{Diferencia\ de\ Voltaje}{Corriente\ de\ consumo}$$
$$Resistencia = \frac{Vfuente - Vtotal}{Corriente\ de\ consumo}$$
$$Resistencia = \frac{24\,[V] - 16.1\,[V]}{40\,[mA]}$$
$$Resistencia = 197.5\,[\Omega]$$

Realizando el sobredimensionamiento de 10% del valor calculo, se obtiene finalmente:

$$Resistencia = 197.5\,[\Omega] + 19.75\,[\Omega]$$
$$\underline{Resistencia = 217.25\,[\Omega]}$$

Por lo tanto, se consideró usar resistencias de 220 $[\Omega]$

4.2.2.2 SEMAFORIZACIÓN PEATONAL

La semaforización peatonal es diseñada bajo el mismo criterio, sin embargo para la animación requerida para la luz verde, es necesario añadir un circuito astable, el mismo que muestre al peatón de forma clara y fácil la indicación de que es posible realizar el cruce de acera.

Además, fue necesaria la colocación de un dispositivo auditivo, el cual permite la advertencia a los peatones con discapacidad visual. La distribución de los diodos led para los semáforos peatonales se realizó bajo el mismo criterio de diseño que los vehiculares, siendo necesaria la utilización únicamente de diodos led rojos y verdes, debido a que los amarillos no han sido normalizado para este fin (Ver Anexo C).

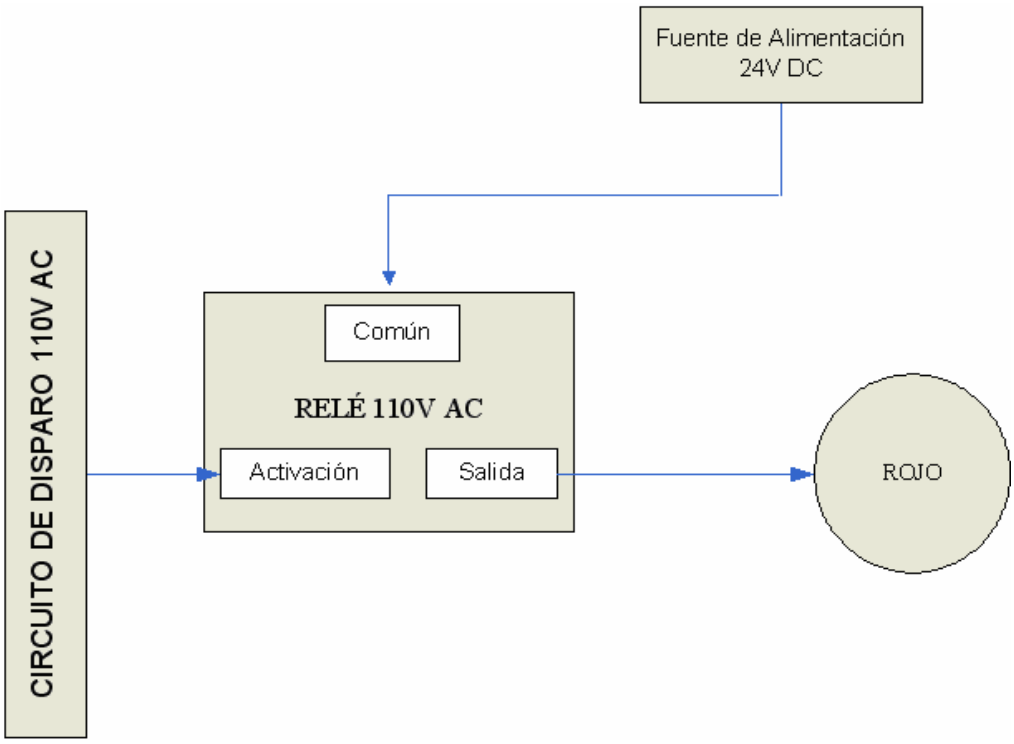


Figura 4.6 Diagrama de Disposición Electrónica de Semaforización Peatonal - Rojo

ANIMACIÓN

La animación requerida para la semaforización peatonal correspondiente al color verde se realizó a través del uso de un circuito astable, mediante el uso del integrado LM555 [37], que permite obtener en la salida la intermitencia necesaria para simular el movimiento de la figura. Sin embargo, al tener en la salida de dicho circuito una señal de 12V, se consideró la utilización de un Relé de 12V DC de activación que permite el encendido de las luminaria mediante el paso de la fuente de 24V DC, es decir, la salida del circuito astable sirve de señal de activación para el relé, obteniendo de este modo los dos estados a través de sus contactos, normalmente abierto y normalmente cerrado. La figura 4.7 muestra la disposición de los dispositivos indicados.

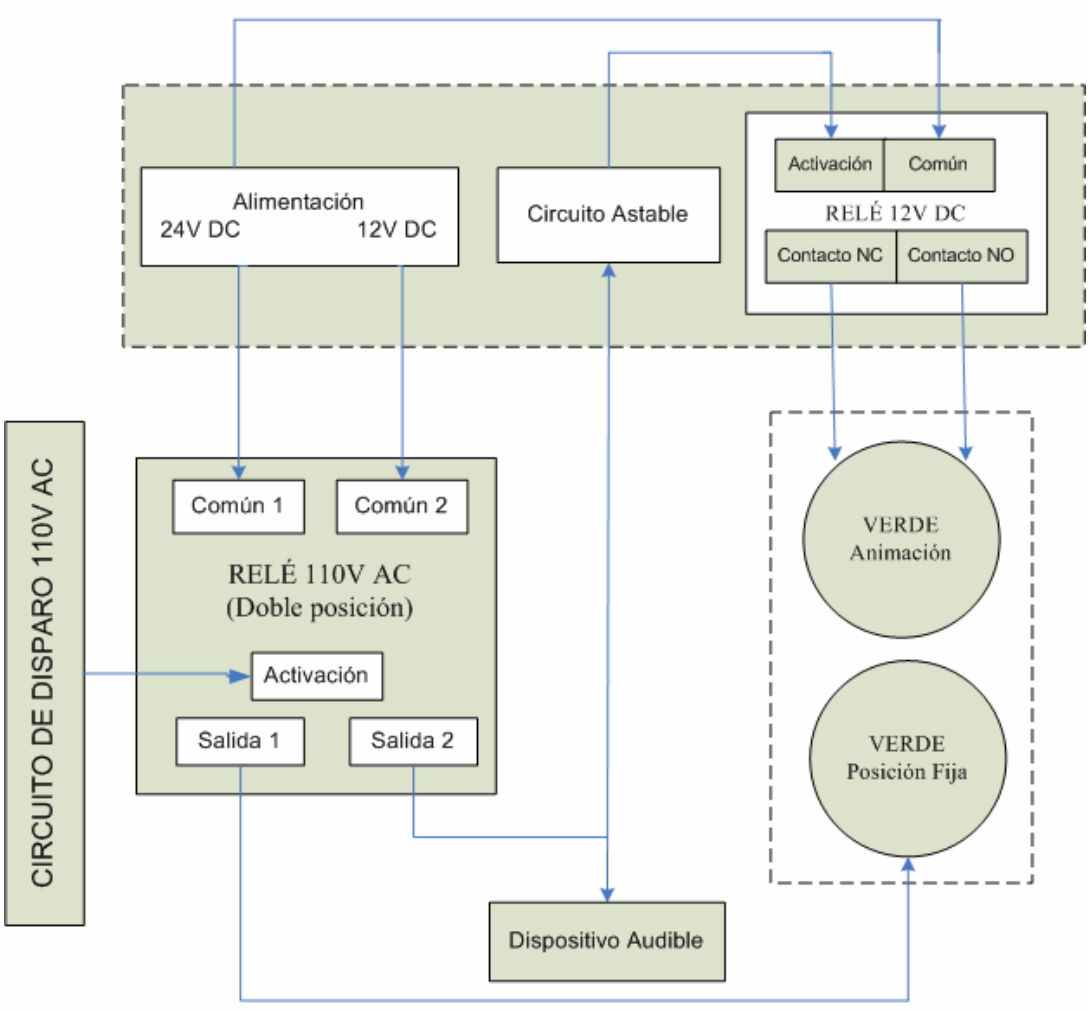


Figura 4.7 Diagrama de Disposición Electrónica de Semaforización Peatonal – Verde

4.3 PROGRAMACIÓN DEL CONTROLADOR

4.3.1 DISEÑO

Actualmente es muy común utilizar microcontroladores para pequeños sistemas de control sin embargo tarjetas electrónicas en los últimos tiempos han modificado esa costumbre por la facilidad de poseer recursos embebidos incluidos en la misma, así en este capitulo se describe el manejo del dispositivo utilizado para el control de señales digitales dirigido a esta aplicación, una tarjeta Rabbit PowerCore Flex 3800.

El manejo lógico del dispositivo es a través de un programa propio llamado Dynamic C 9.20, que se fundamenta en la compilación de código en lenguaje C y grabación en la memoria de programa de la tarjeta Rabbit, tomando en cuenta que posee sus propias librarías adaptadas para el manejo respectivo de cada uno de los recursos del dispositivo.

4.3.1.1 DYNAMIC C

Dynamic C constituye una plataforma de programación a través de la cual se desarrolla la parte lógica del proyecto, por lo que su comprensión es fundamental. La figura 4.7 muestra la pantalla principal de la interfaz de Dynamic C en donde se indican las funciones de los botones principales.

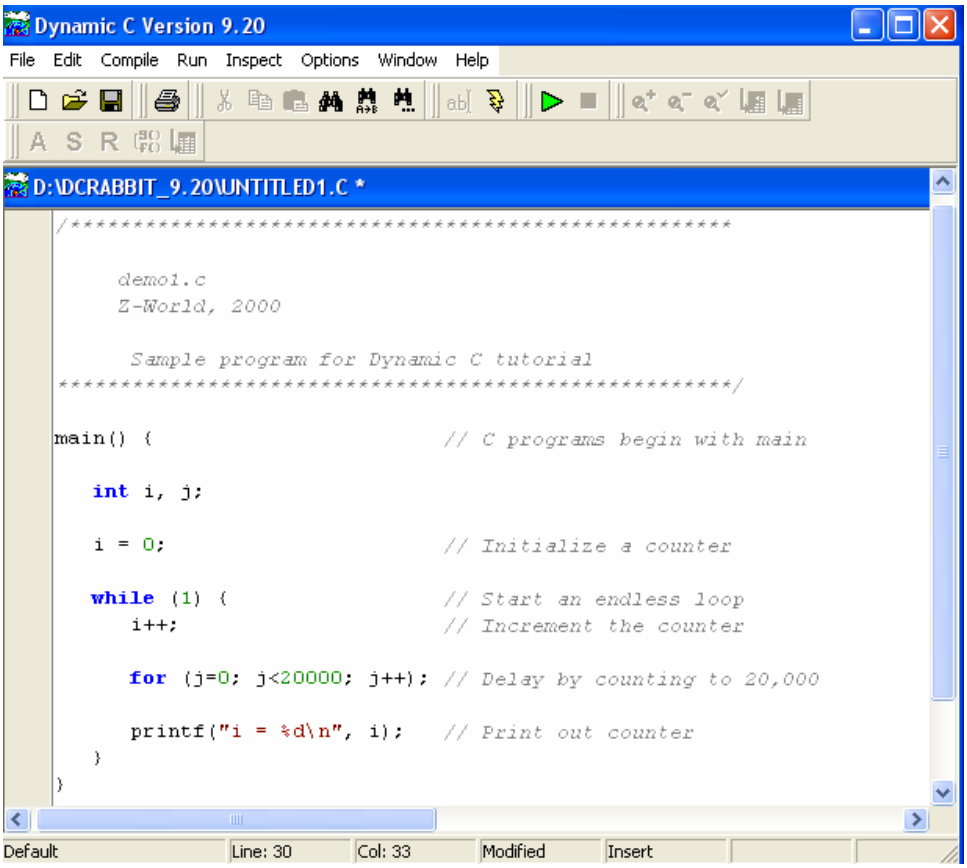


Figura 4.8 Pantalla Principal de Dynamic C

Permite compilar y ejecutar el programa, es posible realizar esta acción mediante el uso de la tecla F9.

Luego de realizar la compilación y ejecución se despliega una ventana llamada Stdio, la misma que indica las salidas en caso de haber utilizado la instrucción *printf()*.

Sirve para compilar únicamente el programa, no para ejecutarlo. Se puede hacer uso de esta acción mediante la tecla F5.

Permite detener la ejecución del programa, mientras se encuentre conectado a la tarjeta.

Sirve para realizar cambios en el código del programa.

Permite desplazarse en el programa en ejecución a través de las instrucciones, sin ingresar a los bucles.

4.3.1.2 DIAGRAMA GENERAL DEL FUNCIONAMIENTO DEL SISTEMA

El desarrollo del software es una parte esencial para el manejo de la tarjeta Rabbit, el cual se basa en un sistema operativo básico para el control de semáforos. Es así que durante el desarrollo del mismo se acoplan varias funciones para la optimización del código, sin embargo existen comandos que no pueden ser llamados desde una función por lo se encuentran dentro de el código main().

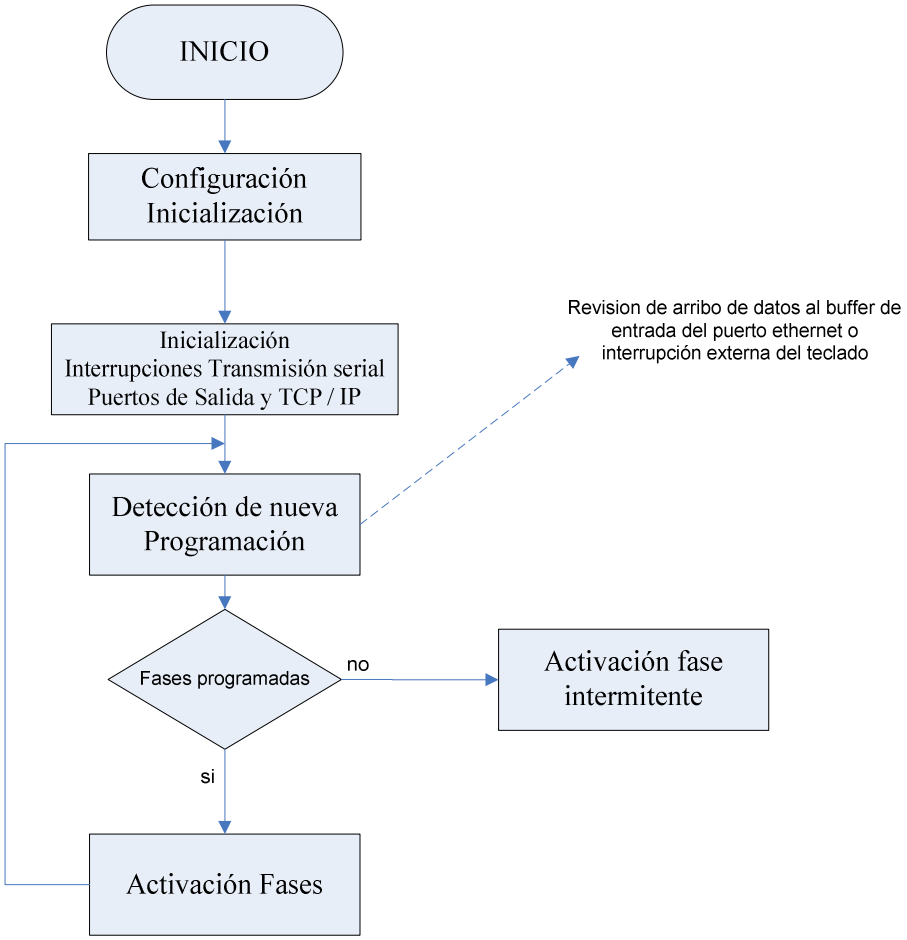


Figura 4.9 Diagrama de flujo general del sistema

La optimización se realiza mediante el uso de subrutinas o funciones, para que encabezado del programa se reduzca a la inicialización de los recursos a utilizar como se muestra en la figura 4.9.

En la cabecera del código se llama a las librerías y declaraciones de inicialización tanto del tablero como de cada recurso utilizado como son TCP/IP, interrupciones y transmisión serial. A continuación se muestra el encabezado y declaraciones del programa.

```
void main()
{
  int bytes_read;
  struct tm rtc;                    // Estructura de tiempo
  longword  destIP;                 // Variable para cambio de dato ip
  tcp_Socket socket;                //puntero al socket tcp
  brdInit();                        // function de inicialización de tablero
  sock_init();                      // Función de inicialización del socket
                                                destIP=resolve(ADDRESS);          //cambio
                                          de la direccion ip tipo texto en una          variable
                                          longword
 WrPortI(PEDDR, &PEDDRShadow, 0x00); // Puerto E como entradas
  #if __SEPARATE_INST_DATA__ && FAST_INTERRUPT
          interrupt_vector ext0_intvec my_isr0; // activación condicionada de
                                                  // interrupción del puerto PE0
                                                  //en compilación de la tarjeta
  #else
          SetVectExtern3000(0, my_isr0);
          SetVectExtern3000(0, GetVectExtern3000(0));
  #endif
  WrPortI(I0CR, &I0CRShadow, 0x09);   //activacion de la interrupción INT0 con PE0
  serFopen(BAUD232);                // activación de Puerto serial  F
                                          serMode(0);       // activación del modo de trabajo del
                                        Puerto serial  // F como serial de 3 hilos
                                                WrPortI(SPCR, &SPCRShadow, 0x84);       //
                                          registro de trabajo del Puerto A como //salidas digitales
  WrPortI(PBDDR, 0, 0xFF);          // Puerto B como salidas
  WrPortI(PFDDR, 0, 0xFF);          // Puerto F como salidas
  WrPortI(PGDDR, 0, 0xFC);          // Puerto G como salidas
  WrPortI(PGFR, 0, 0x00);           // Puerto G como salidas digitales
  WrPortI(PFFR, 0, 0x00);           // Puerto F como salidas digitales
  serFwrFlush();                    // vacía el buffer de escritura del serial F
  serFrdFlush();                    // vacía el buffer de lectura del serial F
```

A partir del encabezado, La secuencia de bucle infinito consta de tres partes principales que son la detección de nueva programación, activación de fases e intermitencia. Para conocer el estado de programación se verifica si los tiempos de fase son diferentes de cero.

4.3.1.3 DETECCIÓN DE PROGRAMACIÓN

Para realizar una nueva programación es necesario que se ingresen tiempos de fase, horarios pico y de intermitencia, a través de dos métodos, el primero en línea por medio de una red TCP/ IP y el segundo a través de la operación manual directamente en el tablero de control.

Si la programación se realiza mediante el software de manejo, es necesario que el dispositivo de control este conectado a la red por lo que el sistema operativo del controlador esta solicitando conexión con el programa master periódicamente y en el caso de existir un arribo de datos en el buffer de entrada del puerto Ethernet, se carga la nueva programación en memoria, reemplazando los tiempos de fase existentes anteriormente.

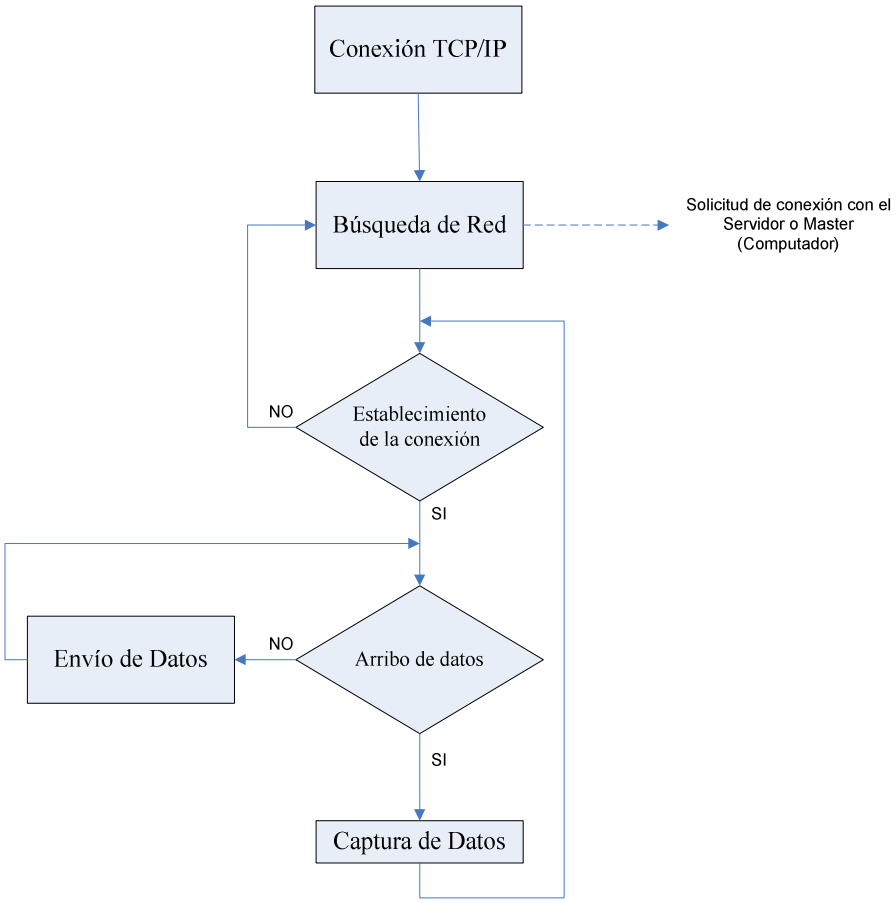


Figura 4.10 Diagrama de flujo Programación por red

El Diagrama de flujo de la figura 4.10 muestra el proceso realizado por el dispositivo lógico para verificar el arribo de datos.

```
…..
        tcp_tick(NULL);
if(sock_established(&socket)==1){ // comprobación de conexion
if(sock_dataready(&socket)>0){  // comprobación de arribo de datos
  sock_read(&socket,buffer,109);// captura de datos en memoria
  program();                    // Función adquisición de datos
}}
else{tcp_open(&socket,1401,destIP,PORT,NULL);} // Solicitud de Conexión
……
```

Si se desea manejar manualmente, el armario de control posee la conexión de un teclado y un LCD que permite la programación directa de las fases, reloj en tiempo real de la tarjeta y de horarios de horas pico.

Para el manejo de estos nuevos datos se hace uso de un menú que inicia con el ingreso de una clave y carga datos por medio del teclado y su visualización en un LCD como se muestra en la figura 4.11.

El proceso inicia a partir de presionar la tecla “*” para el ingreso de la clave, donde, si la clave es correcta se despliegan las opciones del menú.

Dentro de la opción de configuración de fases el sistema operativo pide el ingreso de tiempos y la asignación a la fase deseada. En la opción de fecha y horarios se puede configurar el reloj del sistema, horarios de intermitencia y horarios pico con sus respectivas programaciones.

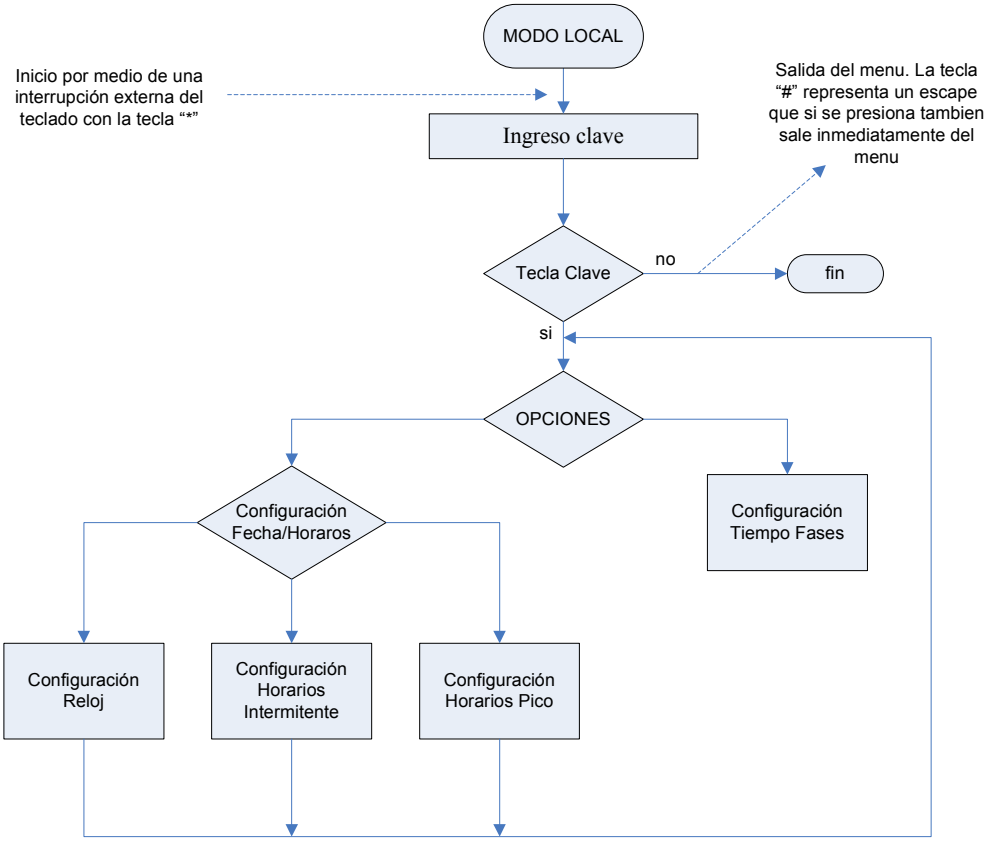


Figura 4.11 Diagrama de flujo programación modo local

4.3.1.4 VISUALIZACIÓN DE DATOS

Durante un proceso de activación de fases, el LCD despliega la fecha y hora del controlador mediante el uso de un reloj en tiempo real y el envío de datos a través de la transmisión serial como se indica en la figura 4.12.

La obtención de datos del reloj en tiempo real es a partir de una estructura propia de la tarjeta, llamando comandos asignados por las librerías de la misma una función independiente llamada print_time(), que incluye el proceso de comparación para la activación de los horarios picos e intermitentes.

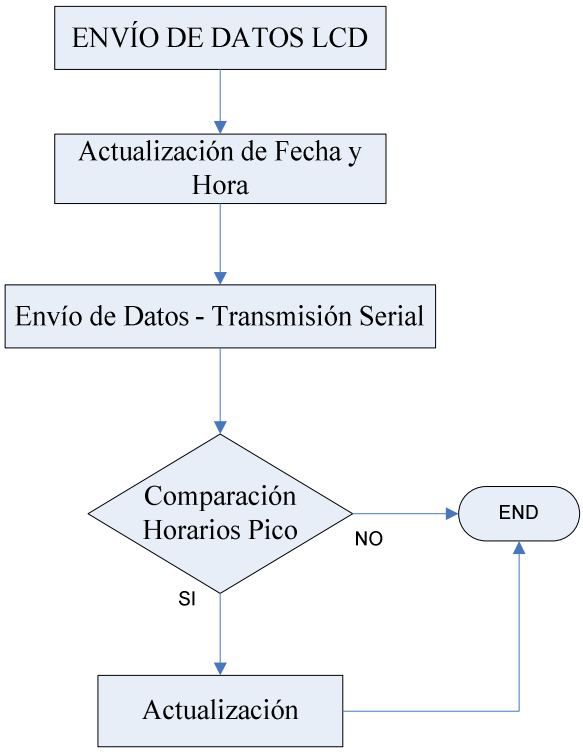


Figura 4.12 Diagrama de flujo envío de Datos al master

Del mismo modo la función llamada mensajes(int) permite el envío de datos serialmente mediante los comandos serFput c y serFputs . Parte de los comandos son descritos a continuación.

```
void print_time(unsigned long thetime)
{  …….
  struct tmthetm; //creacion de la estructura
  mktm(&thetm, thetime); // asignación de valores al puntero
  a1=((thetm.tm_year-100)/10); // conversion para despliegue en lcd
  a2=((thetm.tm_year-100)-(a1*10)); //  conversion para despliegue en lcd
  me1=(thetm.tm_mon/10); // conversion para despliegue en lcd
  me2=(thetm.tm_mon-(me1*10)); // conversion para despliegue en lcd
  d1=(thetm.tm_mday/10); // conversion para despliegue en lcd
  d2=(thetm.tm_mday-(d1*10)); // conversion para despliegue en lcd
  if (a11!=a1 || a22!=a2 || me11!=me1 || me22!=me2 || d11!=d1 || d22!=d2){
    mensajes(31); // llamado a función de envio de datos a LCD
    for(i=0; i<20000; i++);
    a11=a1;
  ……..
```

4.3.1.5 FUNCIONAMIENTO DE FASES

Dentro del proceso principal se encuentra el funcionamiento de fases que cumple una lógica simple de activar la fase que esta programada. Sin embargo en cada activación se mantiene un proceso constante de revisión de conexión para programación ya definido anteriormente. El proceso que cumple esta descrito en la figura 4.13.

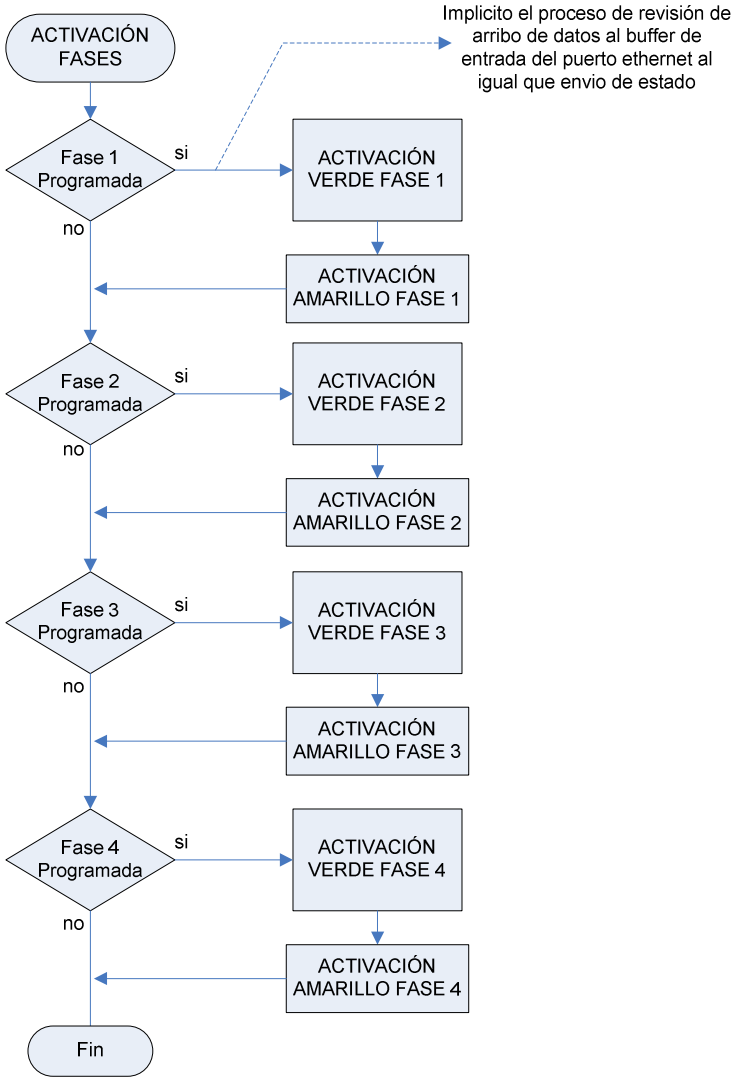


Figura 4.13 Diagrama de flujo Funcionamiento de Fases

Las comparaciones son realizadas para verificar el estado de las fases. En caso de estar activadas, cada fase se mantiene habilitada durante el tiempo programado antes de concluir su proceso, caso contrario realiza el proceso de intermitencia mostrado en el main().

CAPITULO V

DESARROLLO DE INTERFAZ GRÁFICA

5.1 CONSIDERACIONES

5.1.1 INTERFAZ GRÁFICA

El programa considerado para la Interfaz gráfica es Visual Basic, debido a que ofrece múltiples ventajas, especialmente de tipo gráfico, debido a que es necesario tener un control visual del funcionamiento del sistema. Para el desarrollo de la interfaz del proyecto han sido considerados los siguientes aspectos:

- Análisis
- Creación de un interfaz de usuario.
- Definición de las propiedades de los controles
- Generación del código asociado a los eventos que ocurran a estos controles.
- Generación del código del programa.
- Creación de un interfaz de usuario.

5.1.2 BASE DE DATOS

El desarrollo y la administración de la base de datos se realiza mediante MySQL, ya que es un sistema que permite realizar estas funciones mediante el tratamiento de datos. MySQL, como base de datos relacional, utiliza múltiples tablas para almacenar y organizar la información.

Cabe resaltar, que la condición de open source de MySQL, hace que su utilización sea gratuita y se pueda modificar con total libertad, es decir ofrece una amplia aplicabilidad en sistemas de monitoreo y control [29].

5.2 DISEÑO DE HMI

Para la realización del HMI es necesario realizar una presentación del programa por medio de una pantalla la cual posee un semáforo que cambia de color cada cierto periodo de tiempo.

Figura 5.1 Pantalla de presentación del HMI

El ingreso del usuario se realiza por medio de una pantalla auxiliar previo al manejo del programa como tal.

Figura 5.2 Pantalla de ingreso de usuario

La pantalla principal del interfaz posee los semaforos para visualización de las fases que esten activadas, además de tener la fecha, hora, opción de conexión con los cruces y selección de programación a cargar.

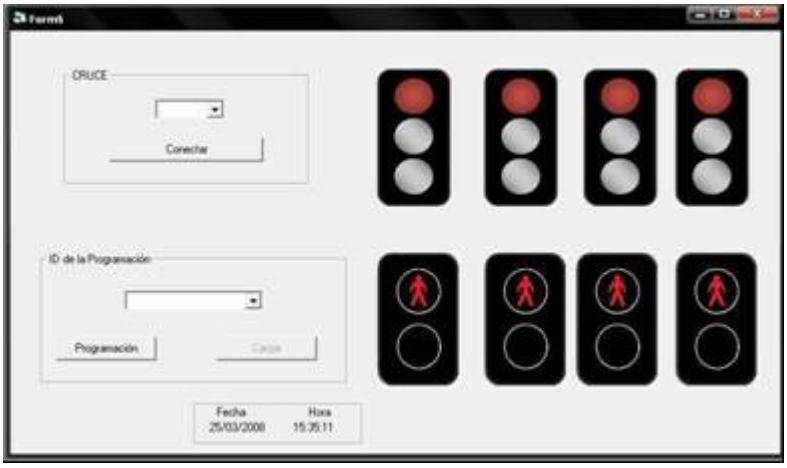


Figura 5.3 Pantalla principal del HMI

La interfaz para almacenar nuevas programaciones posee cuadros de texto para cargar el valor de tiempos de fase, horarios intermitentes y horarios de horas pico asi como las fases a cargar en horarios pico. Y posee un navegador (ADODC) para el manejo de la base de datos.

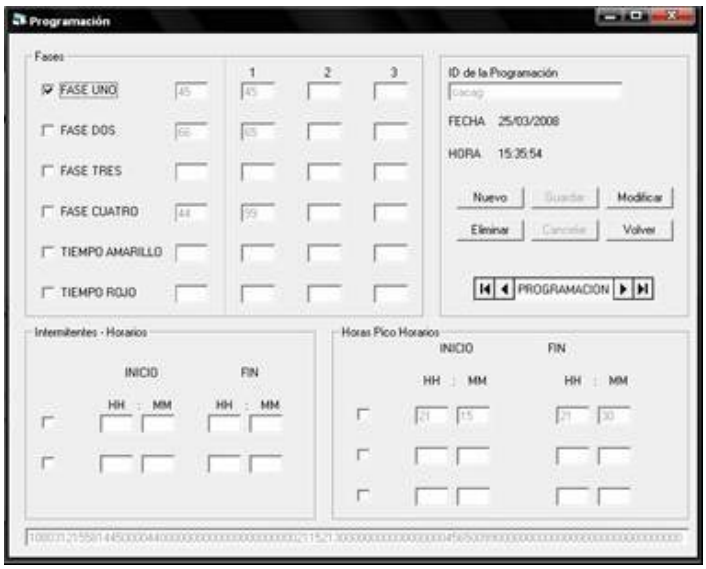


Figura 5.4 Pantalla para almacenamiento de programaciones

El MDIForm posee un menu para salir del programa, ingresar a la pantalla principal y el ingreso de nuevos usuarios.

Figura 5.5 Menú superior del interfaz

La pantalla de ingreso de nuevos usuarios y modificación de los mismos ya almacenados anteriormente posee cajas de texto para el ingreso de los datos y botones llamados por la función que cumplen. Y posee un navegador (ADODC) para el manejo de la base de Datos exclusivamente en la tabla donde esta ubicada la información de los usuarios.

Figura 5.6 Pantalla para almacenamiento de usuarios

La pantalla de reporte es necesaria para obtener la información de los cambios sucedidos durante el turno de un operador, para ello se realiza una lectura de la base de datos, que se actualiza cada vez que existe un cambio de programación en cualquiera de los cruces.

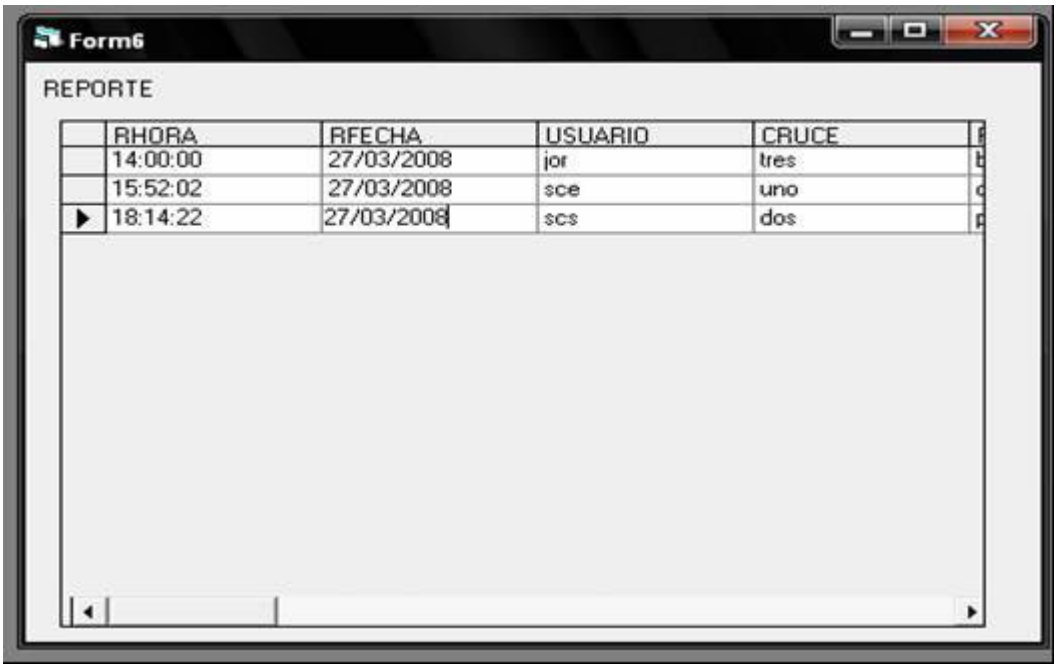


Figura 5.7 Pantalla para almacenamiento de usuarios

5.3 DIAGRAMA GENERAL

La HMI diseñada se fundamenta en el intercambio de datos con cada cruce conectado, además posee una base de datos en donde registra cambios en los controladores, operadores y hora de cambió en conjunto con datos de programación de cruces.

Al iniciar el interfaz carga la primera pantalla en donde se realiza una presentación del producto, mostrando un cambio de luces y esperando un click en la pantalla para continuar. Dentro de la programación esta implícito el dimensionamiento y estética de la presentación en general. A partir de esto se realiza el ingreso de Usuario en un proceso de comparación con la base de Datos. Si se ingresa en el sistema, se puede entablar la conexión con los cruces de semáforos, además de tener la posibilidad de realizar el almacenamiento de nuevos Datos y modificación de antiguos. Así, una vez obtenida la conexión con cualquiera de los cruces se puede realizar el envío y recepción de datos.

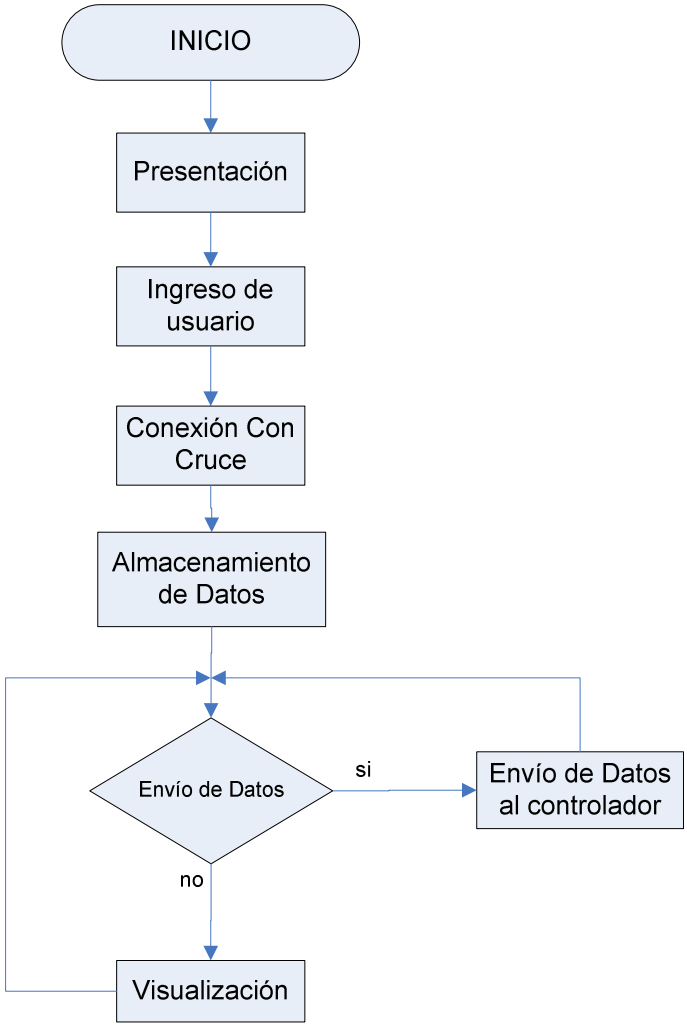


Figura 5.8 Diagrama general de funcionamiento del interfaz

5.3.1 INGRESO DE USUARIO

Para realizar un proceso de registro de usuario del software para conocer quien realizo los cambios en los controladores, se procede a partir del ingreso de una clave y una comparación con la base de Datos.

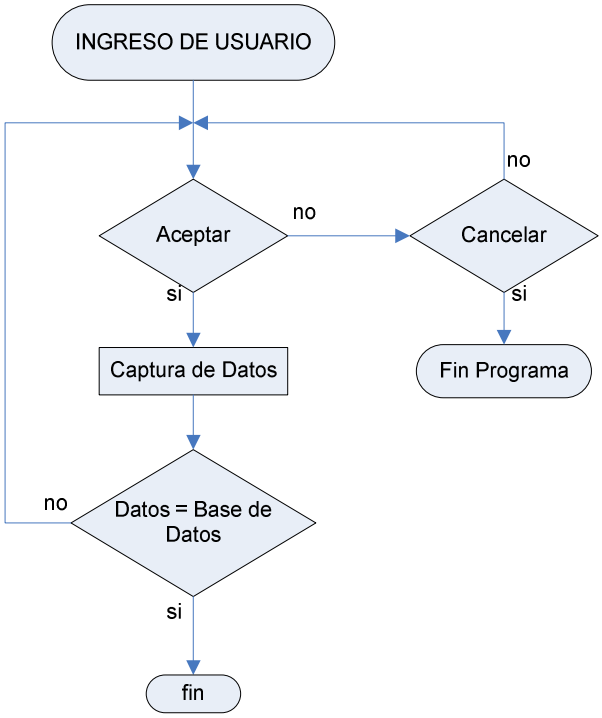


Figura 5.9 Diagrama de flujo de Datos para el ingreso de usuario

En el caso de no existir coincidencia con los campos pre – enlazados se despliega un mensaje de error y se retorna a la condición inicial. Dentro de la programación de los procesos se realiza la evaluación de las cajas de texto, el dimensionamiento y ubicación de la pantalla. El enlace de Visual Basic con la base de datos se realiza a partir de la creación del ODBC respectivo y una herramienta propia de visual Basic llamada ADODC la cual es un manejador de datos, con el cual se realiza el enlace con el ODBC y con la tabla y columna deseada. Un ejemplo de cómo se llama al ADODC esta dentro de la programación de este primer proceso.

```
Private Sub Command1_Click()
criterio = "clave='" & Text2 & "'"
Adodc1.Recordset.MoveFirst  // ubicarse en el primer dato de la tablas
Adodc1.Recordset.Find criterio // búsqueda del dato almacenado en criterio
If Adodc1.Recordset.EOF Then // comparación para saber si es el ultimo dato
…
…
```

El enlace realizado es semejante en todos los procesos, ya que el manejador actúa de la misma manera. Sin embargo para poder hacer uso de este enlace es necesario que se encuentre creada la base de Datos y las tablas necesarias para que no exista error alguno en la búsqueda de datos.

5.3.2 CONEXIÓN CON CRUCE

La conexión con cada cruce depende de la selección del cruce y de la existencia del mismo. El proceso realizado es activar la función que espera levantar una conexión con el controlador del cruce seleccionado, ya que el mismo esta intentando conectarse periódicamente.

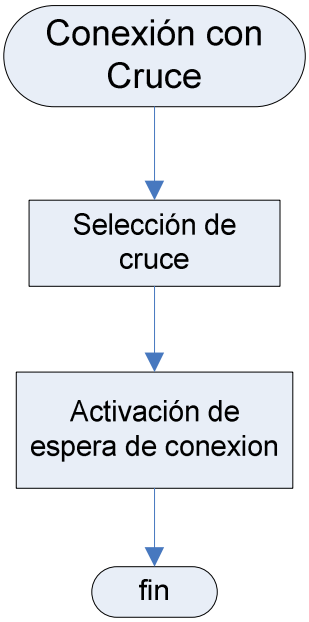


Figura 5.10 Diagrama de flujo de Datos para la conexión de red

A conexión se realiza mediante una herramienta de Visual Basic llamada Microsoft Winsock el cual es un manejador de red, que es necesario configurar en sus propiedades los campos como dirección IP local, dirección IP remota, numero de puerto local, numero de puerto remoto y el protocolo de comunicación a utilizar, ya sea UTP o TCP/IP. Una vez configurado se puede hacer uso tanto de funciones como comandos destinados a comunicación y arribo de datos por ejemplo:

```
……
……
Private Sub Winsock1_ConnectionRequest(ByVal requestID As Long)  // función de conexión
Winsock1.Close                  // Cierra el socket utilizado
Winsock1.Accept requested       // acepta el requerimiento de conexión enviado por el controlador
End Sub                         // fin de función
……
……
Private Sub Command1_Click()
If Combo1.Text = "UNO" Then  // Selección de cruce
   Winsock1.Listen              //Comando de activación para espera de conexión
End If
…….
…….
```

Cada uno de los comandos usados se reiteran en su uso dependiendo de la intersección el cual sea solicitado conexión, ya que es posible el uso de varios winsock a la vez en el mismo programa, recalcando que cada uno de ellos debe usar su propio puerto de socket para evitar conflicto.

5.3.3 ALMACENAMIENTO DE DATOS

Existen dos pantallas en las cuales hay la posibilidad de creación, modificación, borrado y visualización de Datos de ciertas tablas, las cuales cumplen el mismo procedimiento, pero cada una apuntada a su propia tabla.

Para poder realizar cada proceso es necesario llamar los comandos del ADODC de cada pantalla para la modificación de la tabla apuntada. Sin embargo estos procesos pueden ser realizados paralelamente al proceso general.

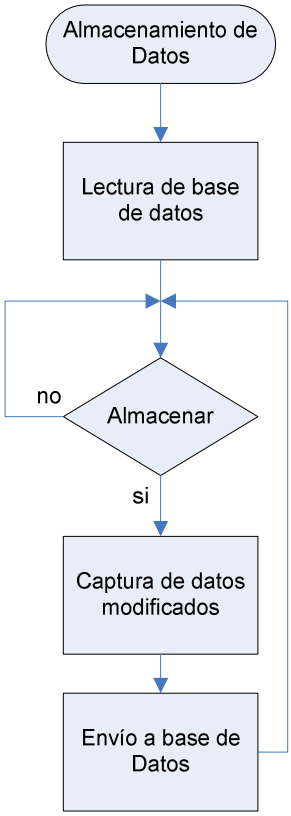


Figura 5.11 Diagrama de flujo de Datos para el almacenamiento de datos

5.3.4 ENVÍO AL CONTROLADOR

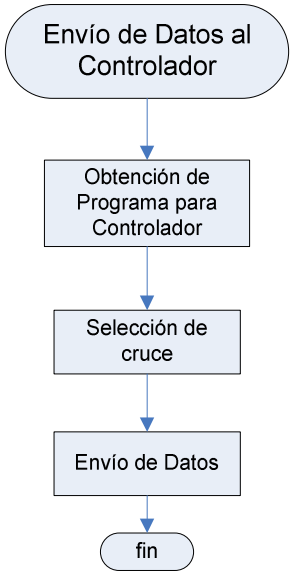


Figura 5.12 Diagrama de flujo de Datos para el envío de datos al controlador

Para el envío de datos es solo necesario el llamado de un comando que apunta al winsock, sin embargo incluye también el proceso de selección de programación, y obtención de información de la base de datos para el envío hacia el controlador.

La obtención del programa se realiza de igual manera que la búsqueda de usuario, una vez obtenida la trama de información del programa para enviar, se almacena en una variable del programa que después de seleccionar el cruce, se envía usando el comando **WinsockX.SendData enviar** , donde **X** es el cruce a enviar la información y **enviar** es la variable que posee la información. La trama de información enviada consta de 109 datos en hexadecimal, descrito a continuación:

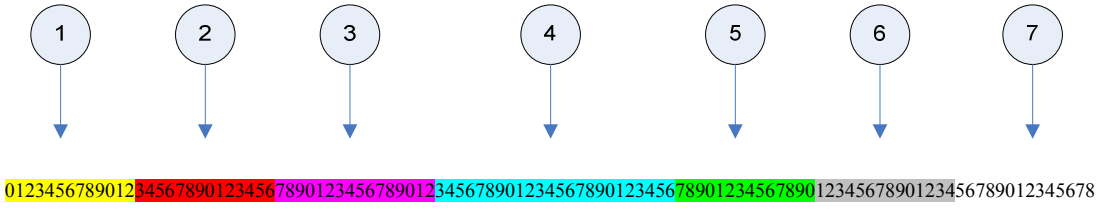


Figura 5.13 Trama de datos enviados a cada controlador

Parte	Descripción	Datos
1	Fecha y Hora tomada del calendario	13
2	Datos de las cuatro Fases , de tiempo de amarillo y tiempo de rojo emergente	14
3	Horarios de Intermitentes	16
4	Horarios de Horas pico 1, 2 y 3	24
5	Datos de las cuatro Fases , de tiempo de amarillo y tiempo de rojo emergente del horario pico 1	14
6	Datos de las cuatro Fases , de tiempo de amarillo y tiempo de rojo emergente del horario pico 2	14
7	Datos de las cuatro Fases , de tiempo de amarillo y tiempo de rojo emergente del horario pico 3	14

Tabla 5.1 Distribución de la trama de datos de envío para los cruces

5.3.5 VISUALIZACIÓN

Una vez conectado el controlador, el arribo de datos realiza cada vez que los semáforos cambian de estado. Una vez que se obtiene el dato se realiza la comparación para conocer en que estado se encuentra el cruce.

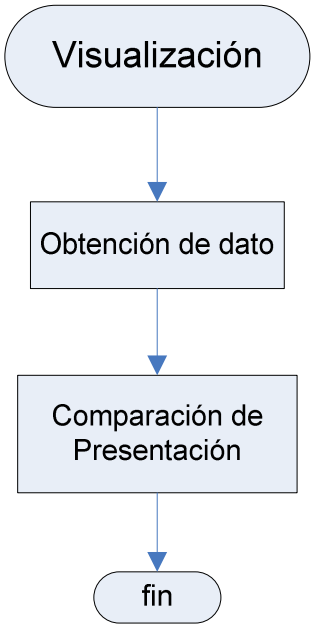


Figura 5.14 Diagrama de flujo de Datos para la visualización del estado del cruce

El arribo del dato se da mediante una función propia del winsock. Esta función actúa como interrupción ya que se ejecuta el momento en que el buffer de entrada posee un nuevo dato.

```
Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)// Función de arribo de dato
Winsock1.GetData datos1     // obtención del dato del buffer de entrada
End Sub
```

5.4. DISEÑO DE BASE DE DATOS

El almacenamiento de datos en un sistema de control es indispensable para la administración del mismo, es así que se vuelve necesario la creación de una base de Datos que almacene todos los cambios sucedidos en la programación y los sujetos participes de la acción.

A partir de este concepto se uso el programa MySQL que es un software gratuito para bases de datos por ser diseñado inicialmente para Linux.

Este es un software muy completo el cual a más de almacenar posee las características necesarias para realizar un enlace con visual Basic y por lo tanto capaz de satisfacer las necesidades de almacenamiento del sistema de semaforización.

La base de Datos posee tres tablas básicas que almacenan los datos necesarios para la administración del sistema.

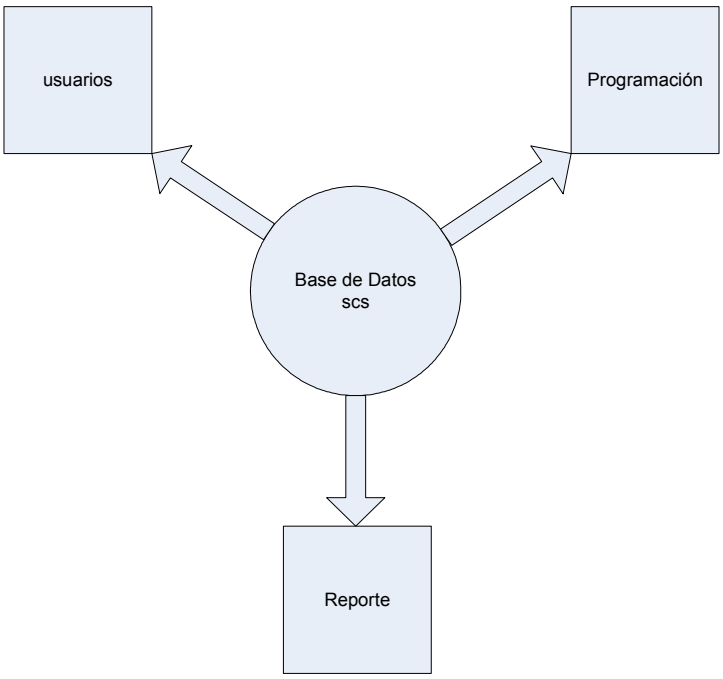


Figura 5.15 Diagrama de flujo de distribución de Base de Datos

Cada una de las tablas se crearon a partir de las necesidades de almacenamiento del programa, debido a esto las tablas constan de los siguientes campos:

Tabla	Campos	Descripción
Usuario	Nombre	Nombre del usuario
	Apellido	Apellido del Usuario
	Usuario	Iniciales del usuario
	Clave	Clave para usuario
	Tipo	Tipo de usuario – administrador/operador
Programación	idprog	Nombre de la programación
	F1	Tiempo de fase
	F2	Tiempo de fase
	F3	Tiempo de fase
	F4	Tiempo de fase
	Ta	Tiempo de amarillo
	Tr	Tiempo de rojo
	Ihi1	Hora de inicio de intermitente 1
	Imi1	Minuto de inicio de intermitente 1
	Ihf1	Hora de fin de intermitente 1
	Imf1	Minuto de fin de intermitente 1
	Ihi2	Hora de inicio de intermitente 2
	Imi2	Minuto de inicio de intermitente 2
	Ihf2	Hora de fin de intermitente 2
	Imf2	Minuto de fin de intermitente 2
	Hphi1	Hora de inicio de hora pico 1
	Hpmi1	Minuto de inicio de hora pico 1
	Hphf1	Hora de fin de hora pico 1
	Hpmf1	Minuto de fin de hora pico 1
	Hphi2	Hora de inicio de hora pico 2
	Hpmi2	Minuto de inicio de hora pico 2
	Hphf2	Hora de fin de hora pico 2
	Hpmf2	Minuto de fin de hora pico 2
	Hphi3	Hora de inicio de hora pico 3

	Hpmi3	Minuto de inicio de hora pico 3
	Hphf3	Hora de fin de hora pico 3
	Hpmf3	Minuto de fin de hora pico 3
	Fp11	Tiempo de fase pico 1
	Fp12	Tiempo de fase pico 1
	Fp13	Tiempo de fase pico 1
	Fp14	Tiempo de fase pico 1
	Fp1a	Tiempo de amarillo pico 1
	Fp1r	Tiempo de rojo pico 1
	Fp21	Tiempo de fase pico 2
	Fp22	Tiempo de fase pico 2
	Fp23	Tiempo de fase pico 2
	Fp24	Tiempo de fase pico 2
	Fp2a	Tiempo de amarillo pico 2
	Fp2r	Tiempo de rojo pico 2
	Fp31	Tiempo de fase pico 3
	Fp32	Tiempo de fase pico 3
	Fp33	Tiempo de fase pico 3
	Fp34	Tiempo de fase pico 3
	Fp3a	Tiempo de amarillo pico 3
	Fp3r	Tiempo de rojo pico 3
	Trama	Trama para envío a controlador
Reporte	RHORA	Hora de cambio en cruce
	RFECHA	Fecha de cambio en cruce
	USUARIO	Usuario que realizo la programación
	CRUCE	Cruce en el que se programó
	PROGRAMACIÓN	Programación cargada en el cruce

Tabla 5.2 Distribución de campos en las tablas de la base de datos

CAPITULO VI

PRUEBAS Y RESULTADOS

6.1 DESCRIPCIÓN

Dentro de los parámetros necesarios para el correcto funcionamiento del sistema de semaforización es necesario realizar pruebas para comprobar la fiabilidad, conectividad y funcionamiento en general del mismo.

A partir de esto, durante la realización, avance y desarrollo del proyecto se llevaron a cabo distintas pruebas con tiempos establecidos. Las aprobación de pruebas esta detallada en la tabla.

Pruebas	**Duración**	**Elementos**	**Resultado**
Luces LED	4 meses	Luces LED, Fuentes de Semáforos	✓
Circuito de Disparo	1 mes	Circuitos de Disparo, Tarjeta Rabbit, Luces LED, Fuentes de Semáforos	✓
Visualización de Datos LCD	2 meses	Tarjeta Rabbit, Circuito LCD	✓
Conexión De la Tarjeta en la Red	2 meses	Tarjeta Rabbit, Servidor, Switch.	✓
Conexión De la Tarjeta al Interfaz Gráfica	1 mes	Tarjeta Rabbit, Servidor, Switch.	✓
Paso de Datos a la Tarjeta Controladora	2 semanas	Tarjeta Rabbit, Servidor, Switch, Circuitos de Disparo, Luces LED, Fuentes de Semáforos	✓

Lógica de Programación del Controlador	1mes	Tarjeta Rabbit, Servidor, Switch, Circuitos de Disparo, Luces LED, Fuentes de Semáforos	✓

Tabla 6.1. Detalle de pruebas por períodos

6.2 PROCEDIMIENTO

6.2.1 LUCES LED

Después de la fabricación de las matrices de diodos led fue necesario comprobar su funcionamiento mediante la activación de las mismas durante varias horas, es así que se mantuvo en funcionamiento por veinte y cuatro horas durante dieciséis semanas.

6.2.2 CIRCUITO DE DISPARO

Luego de realizar el diseño y fabricación del circuito de disparo se verificó su funcionamiento mediante mediciones a la salida del mismo, tomando en cuenta que de ésta depende la activación de las luminarias en el subsistema de semaforización, a partir de la señal de control enviada por la tarjeta Rabbit. Su prueba se realizó durante las veinte y cuatro horas en el transcurso de cuatro semanas.

6.2.3 VISUALIZACIÓN DE DATOS LCD

A través de la utilización de las tarjetas para el manejo del LCD y teclado matricial, se procedió a realizar la comprobación del envío y recepción de datos y caracteres mediante la transmisión serial, además del ingreso y visualización de los mismos en el display. Estas pruebas se realizaron durante ocho semanas.

6.2.4 CONEXIÓN DE LA TARJETA EN LA RED

Para la verificación del funcionamiento de la tarjeta en la red fue necesario programarla con su respectiva dirección IP, de este modo al hacer un ping de comprobación, se comprobó que la conexión se completo sin errores.

Esta prueba fue llevada a cabo durante ocho semanas, sin que sucedieran desconexión o problemas de comunicación.

6.2.5 CONEXIÓN DE LA TARJETA AL INTERFAZ GRÁFICA

Luego de establecer la comunicación de la tarjeta a la red, se realizó la conexión con la interfaz gráfica mediante el uso de herramientas que el software sobre el cual se programó ofrece, verificando las banderas del registro manejador de Red. Estas pruebas se llevaron a cabo en el transcurso de cuatro semanas, comprobando además el modo de reconexión en caso de que existan fallos en el suministro de energía eléctrica en el servidor, por lo que debe ser reiniciado el programa.

6.2.6 PASO DE DATOS A LA TARJETA CONTROLADORA

Al establecer la conexión del controlador con el interfaz gráfica se realizó la comunicación bidireccional de datos, para lo cual se realizaba una envío de espejo, es decir que se enviaron datos hacia el controlador y en el momento de ser recibidos, éste los reenviaba hacia el servidor, comprobándose de este modo que la información enviada se recepte sin pérdidas durante su envío. Su verificación se realizó durante dos semanas.

6.2.7 LÓGICA DE PROGRAMACIÓN DEL CONTROLADOR

Una vez realizadas las pruebas sobre cada uno de los subsistemas, se procedió a realizar su interconexión y verificación integral, considerando la lógica de programación y el funcionamiento en conjunto tanto de hardware como de software.

Se hicieron pruebas respecto a los tiempos de activación programados, horarios pico, intermitentes, acceso a la programación manual y conectividad en red. Esta verificación general del sistema se realizó durante cuatro semanas.

CAPITULO VII

ASPECTO ECONOMICO

7.1 INTRODUCCIÓN

El diseño del proyecto de Sistema de Control de Monitoreo Centralizado de Flujo Vehicular y Peatonal no se lo realizó únicamente en base al aspecto técnico, sino que, se llevó a cabo considerando la optimización de recursos económicos; esto debido a que a pesar de que actualmente existen sistemas que permiten realizar el monitoreo, estos requieren de un alto costo de instalación, funcionamiento y mantenimiento, por lo que fue necesario desarrollar un sistema que además de ser de bajo presupuesto garantice su control y funcionalidad, ofreciendo seguridad tanto a los usuarios de las vías como a los responsables de la administración vial.

El análisis económico tiene dos enfoques, el primero hace referencia a la Inversión Inicial y el segundo a la Reducción de Gastos que genera la utilización del sistema, debido al bajo consumo de energía. Estos son analizados a continuación.

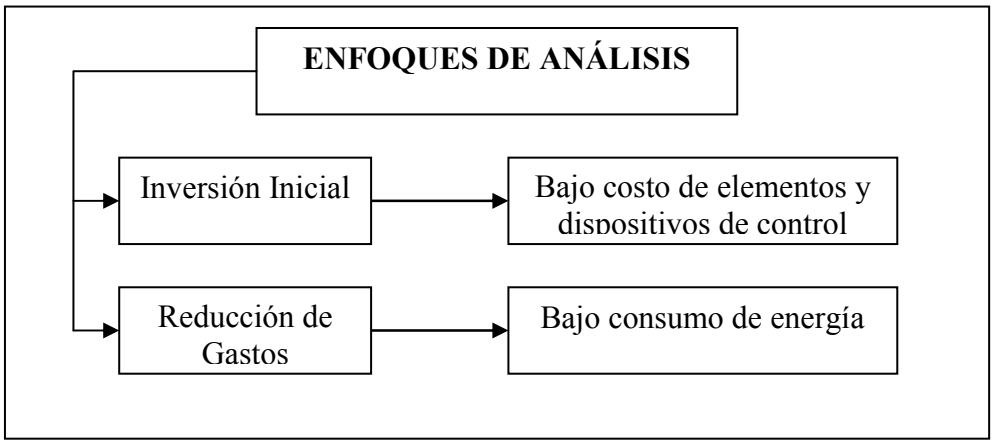


Figura 7.1 Enfoques de análisis económicos

7.2 INVERSIÓN INICIAL

Los dispositivos electrónicos, así como los elementos mecánicos de soporte considerados dentro del diseño son diversos, cada uno con un objetivo en común, garantizar mediante su correcta utilización el funcionamiento general del sistema.

Las siguientes tablas muestran la inversión realizada, esta información es detallada en función de cada uno de los subsistemas que forman parte del proyecto en general.

SUSBSISTEMA DE SEMAFORIZACIÓN			
DISPOSITIVO ELECTRÓNICO	**COSTO UNITARIO US$**	**CANTIDAD**	**COSTO TOTAL US$**
Led de alta luminosidad rojo	0,20	1152	230.4
Led de alta luminosidad amarillo	0,20	800	160
Led de alta luminosidad verde	0,30	1248	374.4
Placa de matriz vehicular	15	12	180
Placa de matriz peatonal roja	12	4	48
Placa de matriz peatonal verde (animación)	12	4	48
Tarjeta de alimentación de 24 V.	2	4	8
Resistencia 330Ω a 1/2W	0.04	204	8.16
Resistencia 220 Ω 1/2W	0.04	176	7.04
Bornera de dos entradas	0,40	20	8
Bornera de tres entradas	0,60	8	4.8
Puente de diodos KBL404	1,10	4	4.4
Puente de diodos 2A	0,95	4	3.8
Capacitor electrolítico 100uF, 250V	0,16	4	0.64

Transformador de 24V con tap central	6,50	8	52
CI LM555	0,45	4	1.8
Capacitor electrolítico 10uF, 50V	0,16	4	0.64
Capacitor cerámico de 1pF	0,12	4	0.48
Potenciómetro de precisión	1,60	4	6.4
Transistor N2222	0,20	4	0.8
Dispositivo Audible	1,50	4	6
DISPOSITIVO ELECTROMECÁNICO			
Relé de 110VAC de activación	3,60	20	72
DISPOSITIVO ELECTRÓNICO	**COSTO UNITARIO US$**	**CANTIDAD**	**COSTO TOTAL US$**
Relé de 12VDC de activación	1,20	4	4.8
DISPOSITIVO DE SOPORTE			
Sockets para Relé	4.20	20	84
Estructura de hierro para 3 luminarias (semáforos vehiculares)	160	4	640
Estructura de hierro para 2 luminarias (semáforos peatonales)	110	4	440
Lunas de protección	5	20	100
Empaques de caucho	0,50	20	10
Sujetadores	0,25	20	5
Riel DIN	2,80	8	22.4
TOTAL			**US$ 2531.96**

Tabla 7.1 Inversión Económica del Subsistema de Semaforización

SUSBSISTEMA DE CONTROL LOCAL			
DISPOSITIVO ELECTRÓNICO	**COSTO UNITARIO**	**COSTO TOTAL**	**OBSERVACIÓN**
Tarjeta Rabbit	198	1	198
Placa de transmisión serial LCD	5	1	5
Placa de Teclado	5	1	5
Placa de Circuitos de Disparo	12	2	24
Borneras de alimentación AC	4	1	4
LCD	8.90	1	8.90
Teclado matricial	8.20	1	8.20
PIC 16F628A	3.95	1	3.95
Decodificador MM74C922	14.90	1	14.90
Led rojo	0.10	4	0.40
Led amarillo	0.10	4	0.40
DISPOSITIVO ELECTRÓNICO	**COSTO UNITARIO US$**	**CANTIDAD**	**COSTO TOTAL US$**
Led verde	0.10	4	0.40
Resistencia 200 Ω, 1/4W	0.03	12	0.36
Resistencias 1KΩ, 1/4W	0.03	24	0.72
Capacitor cerámico de 22 pF	0.12	2	0.24
Cristal de 4 MHz.	0.90	1	0.90
Potenciómetro de precisión	1.60	1	1.60
MOC 3010	1	12	12
Triac BTA16	1.10	12	13.2

Bornera de dos entradas	0.40	2	0.80
Bornera de tres entradas	0.60	4	2.40
Fusibles de 2A	0.10	12	1.20
Borneras para transmisión de datos de 4 bits	0.60	2	1.20
Borneras para transmisión de datos de 8 bits	0.85	8	6.80
Borneras para transmisión de datos de 16 bits	1.20	2	2.40
DISPOSITIVO DE SOPORTE			
Caja de Control con Tablero metálico (40x60)	65	1	65
Sujetadores de placas (topes)	0.20	21	4.20
TOTAL			**US$ 113.42**

Tabla 7.2 Inversión Económica del Subsistema Control

SUSBSISTEMA DE MONITOREO REMOTO			
DISPOSITIVO ELECTRÓNICO	**COSTO UNITARIO**	**COSTO TOTAL**	**OBSERVACIÓN**
CPU	600	1	600
Switch	80	1	80
periféricos	500	1	500
TOTAL			**US$ 1180**

Tabla 7.3. Inversión Económica del Subsistema de Monitoreo Remoto

ELEMENTOS ADICIONALES			
ELEMENTO	**COSTO UNITARIO**	**COSTO TOTAL**	**OBSERVACIÓN**
Cable de Red	0,60	10	6
Cable flexible #12	0,80	30	24
Cable gemelo	0,90	30	27
Cable plano	1,40	5	7
TOTAL			**US$ 64**

Tabla 7.4 Inversión Económica de Elementos Adicionales

SUBSISTEMA	COSTO US$
Semaforización	2531.96
Control Local	113.42
Monitoreo Remoto	1180
Adicional	64
INVERSIÓN TOTAL	**3889.38**

Tabla 7.5 Inversión Total del Proyecto

7.3 REDUCCIÓN DE GASTOS

Dentro del diseño del sistema se consideró la implementación a través de la utilización de diodos led de alta luminosidad, debido a su bajo consumo de energía eléctrica, es decir ofrecen una gran ventaja sobre los sistemas tradicionales, que trabajan con luminarias incandescentes, los cuales son usados aún actualmente y que generan grandes pérdidas económicas.

Esto se refleja a través del cálculo de consumo de potencia por parte de los dos sistemas. De este modo se puede observar los beneficios de consumo que ofrece el sistema que utiliza diodos led con respecto a los sistemas tradicionales.

Los sistemas que emplean luces incandescentes, tiene un consumo por luminaria de 100[W], mientras que la potencia de consumo del sistema de diodos led permite reducir los costos de energía como se muestra a continuación:

Potencia de Consumo del Sistema de Leds

Sistema de Leds

$$\text{Potencia} = \text{Voltaje} * \text{Corriente}$$
$$\text{Voltaje} = 24\,[\text{V}]\text{DC}$$
$$\text{Corriente Máxima} = 1.2\,[\text{A}]$$
$$\text{Potencia} = 24\,[\text{V}] * 1.2\,[\text{A}]$$
$$\underline{\underline{\text{Potencia} = 28.8[\text{W}]}}$$

Como se puede observar el consumo de potencia es menor en el sistema de control, ofreciendo un ahorro de más del 71.2% respecto a los sistemas tradicionales.

Existe otro beneficio de utilizar diodos led de alta luminosidad, esta constituye su vida útil, la cual es de 100.000 horas, lo cual permite reducir costos de mantenimiento, e incluso, cuando sea necesario hacerlo se realizará una inversión mucho más conveniente, ya que adquirir estos dispositivos es más económico que las luminarias incandescentes.

Además, el diseño realizado en matrices de diodos led, permite ofrecer, tanto a los administradores u operadores del sistema, así como a los usuarios de las vías tener un sistema en permanente funcionamiento, ya que si existe algún inconveniente con algún grupo de leds dentro de una matriz, los demás seguirán desenvolviendo su función, sin mayores variaciones dentro del normal desempeño.

CAPÍTULO VIII

CONCLUSIONES Y RECOMENDACIONES

8.1 INTRODUCCIÓN

El permanente crecimiento del flujo vehicular ha impulsado la búsqueda de nuevos sistemas que permitan brindar beneficios a los usuarios de las vías, es decir establecer procesos que puedan ser adaptados de forma práctica a sus diversas necesidades, siendo necesario realizar diseños a largo plazo a partir del análisis de los permanentes cambios a los que está inevitablemente sometido nuestro entorno.

Es así, que se diseñó e implementó un sistema a través del cual fue posible cubrir los diferentes requerimientos que exige nuestro medio, ya que se pudo llegar a establecer un modelo que además de tener un bajo costo inicial, tiene una garantía de funcionamiento a largo plazo; esto fue posible conseguir mediante la selección adecuada no solo de dispositivos electrónicos sino que también de la determinación de metodologías de investigación que permitieron cumplir con las metas planteadas al inicio del proyecto.

De este modo, se pudo realizar un sistema fiable, altamente flexible, con capacidad de expansión, bajo el cumplimiento de normas tanto técnicas como viales, constituido de dispositivos que ofrecen un gran desempeño, alto rendimiento y larga vida útil.

8.2 CONCLUSIONES

El sistema permite reducir gastos referentes al consumo de energía debido a que han sido considerados dentro del diseño diodos led de alta luminosidad, los cuales manejan corrientes muy bajas. Además estos elementos tienen una vida útil de 100.000 horas, lo

cual garantiza su permanente funcionamiento, requiriendo baja inversión en su reposición o cambio y extendiendo los períodos correspondientes a manteniendo.

Existe un crecimiento considerable del uso de tarjetas electrónicas programables a raíz de la funcionalidad, beneficios y garantía que estas ofrecen, facilitando el acoplamiento y adaptación en sistemas de control.

En el caso de que el controlador pierda conectividad con el servidor, la lógica de la tarjeta esta diseñada para que se vuelva a levantar la conexión en caso de que el usuario de la interfaz gráfica lo requiera sin dejar de realizar el proceso en el que se encuentre.

Cuando se usa TCP/IP de la tarjeta rabbit es necesario revisar si existen posibles registros compartidos que bloqueen el uso de otros puertos y funciones de la tarjeta a pesar de que estén habilitados, para esto es necesario revisar mapas de memoria y la configuración de registros generales.

La tarjeta Rabbit puede ser programada únicamente mediante Dinamic C, ya que este posee las librerías necesarias para manejo de todas las características y recursos que poseen estas tarjetas.

Para verificar la funcionalidad del sistema a través del desarrollo de la programación del controlador, es necesario realizar pruebas por módulos para luego integrar todo el código y efectuar las pruebas generales respectivas.

Existen varios puertos de la tarjeta que son de uso específico que no pueden ser utilizados para la programación, sin embargo es posible deshabilitar sus funciones principales utilizando buses auxiliares que deben ser activados mediante comandos específicos del Dynamic C.

Es necesaria la espera de conexión cuando se ha perdido la misma ya que la apertura del socket de la tarjeta se realiza después de haber culminado un ciclo completo de programa.

La tarjeta Rabbit, como controlador, ofrece múltiples recursos, los cuales fueron considerados dentro de la transmisión de datos, ya que el sistema ofrece dos tipos de configuración, una local mediante la utilización de uno de los puertos seriales y otra remota a través del manejo del puerto Ethernet, manteniendo una alta eficiencia en la comunicación.

Es posible realizar a través de la interfaz la programación de rutinas en función de las circunstancias existentes, cambios que son respaldados en una base de datos que puede ser analizada, con hora y fecha, nombre del operador o administrador a cargo, información necesaria dentro de todo sistema de monitoreo y control.

8.3 RECOMENDACIONES

La permanente necesidad por desarrollar planes de educación vial en nuestro país, además del rápido crecimiento del parque automotor, ha sido motivo para que los gobiernos seccionales consideren conveniente la implementación de sistemas de control de flujo vehicular y debido a la falta de estos en nuestro medio, han tenido que importar equipos y solicitar personal extranjero para su instalación, convirtiéndose en un gasto muchas de las veces demasiado elevado para los fines pretendidos, por lo que consideramos imprescindible el desarrollo de proyectos por parte de los institutos de educación superior que beneficien el desarrollo social y no vernos en la obligación de depender de agentes externos a nuestra realidad.

Es necesario que se establezcan normativas que rijan el funcionamiento y desempeño de los sistemas de control vial en nuestro país, ya que únicamente existen modelos que hacen referencia a otros países como Colombia y Chile, sin embargo la culturas son diferentes por lo que se debe determinar políticas propias para nuestro entorno.

Dentro de la planeación del proyecto es imprescindible realizar la selección del controlador en función de la aplicación a desarrollar, ya que existe una amplia gama de tarjetas pertenecientes a la familia Rabbit, las mismas que a pesar de poseer características propias, trabajan bajo la misma plataforma de programación, Dynamic C.

Se debe realizar una revisión general de la tarjeta Rabbit, debido a que existen múltiples puertos que están siendo utilizados internamente por otras funciones, siendo necesaria su habilitación a través de comandos específicos para activar funciones auxiliares.

Es necesario considerar el desarrollo de sistemas de respaldo de energía eléctrica que permita su permanente funcionamiento, constituyendo una alternativa la utilización de paneles solares, los cuales requieren de una mínima inversión reduciendo el impacto en el medio ambiente.

Se recomienda dar capacitación al personal encargado del manejo del sistema, ya que conlleva una gran responsabilidad la administración vial, ya que de ello dependen no solo el evitar altos niveles de tráfico sino que una mala utilización puede conducir a la provocación de accidentes en la vía pública.

La instalación del presente proyecto debe llevarse a cabo considerando las características viales y de crecimiento vehicular de cada ciudad, además de los factores sociales, culturales y ambientales que puedan afectar o incidir en el normal funcionamiento del sistema de control presentado.

REFERENCIAS BIBLIOGRÁFICAS

CAPITULO I

[1] Cal, Rafael, "Ingeniería de Tránsito, fundamentos y Aplicaciones", página 4, 2000.

[2] "Propuesta de Normalización del Transporte Terrestre", Policía Nacional, Dirección Nacional de Tránsito y Transporte Terrestre, páginas 6, 24, 26, 45, 53; 2007.

[3] http://cisco.com/web/learning/netacad/index.html, CCNA1, Capítulo 9

[4] "Propuesta de Normalización del Transporte Terrestre", Policía Nacional, Dirección Nacional de Tránsito y Transporte Terrestre. Páginas 1-109; 2007

CAPITULO II

[5] PowerCore Flex, Introduction, página 1, user's manual, 2004, www.rabbitsemiconductor.com.

[6] PowerCore Flex, appendix A, Electrical and Mechanical Characteristics, página 94, user's manual, 2004, www.rabbitsemiconductor.com.

[7] PowerCore Flex, Hardware Reference, página 27, user's manual, 2004, www.rabbitsemiconductor.com.

[8] Rabbit 3000 Microprocessor, Rabbit 3000 Design Features, página 12, user's manual, 2004, www.rabbitsemiconductor.com.

[9] PowerCore Flex, Internal and External Buses, página 32, user's manual, 2004, www.rabbitsemiconductor.com.

[10] PowerCore Flex, use of Rabbit 3000 Ports, página 29, user's manual, 2004, www.rabbitsemiconductor.com.

[11] PowerCore Flex, Hardware Reference, páginas 34-38, user's manual, 2004, www.rabbitsemiconductor.com.

[12] Prototyping Board, Appendix B, página 112, user's manual, 2004, www.rabbitsemiconductor.com.

[13] Prototyping Board, Appendix B, páginas 110-111, user's manual, 2004, www.rabbitsemiconductor.com.

[14] PowerCore Flex, PowerCore Digital Inputs and Outputs, página 28, user's manual, 2004. www.rabbitsemiconductor.com.

[15] "Propuesta de Normalización del Transporte Terrestre", Policía Nacional, Dirección Nacional de Tránsito y Transporte Terrestre, página 6; 2007.

[16] "Propuesta de Normalización del Transporte Terrestre", Policía Nacional, Dirección Nacional de Tránsito y Transporte Terrestre, página 5; 2007.

CAPITULO III

[17] http://es.wikipedia.org/wiki/Protocolo_de_red

[18] http://www.learnthenet.com/spanish/glossary/tcpip.htm

[19] http://es.wikipedia.org/wiki/Ethernet

[20] http://www.paginasprodigy.com/campechedigital/arielmedina1978/ethernet.doc

[21] http://www.textoscientificos.com/redes/ethernet/principios-operacion-ethernet

[22] Software, networking solutions TCP/IP, introduction, Ethernet Basics, página 3, 2004, www.rabbitsemiconductor.com

[23] http://www.tech-faq.com/lang/es/csma-cd.shtml

[24] http://es.wikipedia.org/wiki/RJ-45

[25] http://digital.ni.com/public.nsf/allkb/0390012581#232

[26] PowerCore Flex, Using the TCP/IP Features, página 86, user's manual, 2004. www.rabbitsemiconductor.com.

[27] Networking Solutions TCP/IP, Dynamic C TCP/IP Implementation, TCP Socket Function, páginas 29-31, user's manual, 2004, www.rabbitsemiconductor.com.

CAPITULO IV

[28] AGULLEIRO, Ignacio, MARTÍNEZ, Miguel, "Técnicas modernas para la medición de sistemas de puesta a tierra en zonas urbanas", página 7-10, 2001.

CAPITULO V

[29] http://www.esepestudio.com/articulo/desarrollo-web/bases-de-datos-mysql/Que-es-MySQL.htm.

DATA SHEETS

[30] Tarjeta Rabbit con módulo PowerCore Flex.
www.rabbitsemiconductor.com

[31] MOC 3010.
www.datasheetcatalog.net/es/datasheets_pdf/M/O/C/3/MOC3010.shtml

[32] Triac BT136.
www.datasheetcatalog.com/datashets_pdf/B/T/1/3/BT136-500.shtml

[33] LCD HD44780U – HITACHI.
www.eisee.fr/~perrotol/LCD-HD44780.pdf

[34] PIC 16F628A.
http://ww1.microchip.com/downloads/en/devicedoc/40044b.pdf

[35] MM74C922
www.datasheetcatalog.net/es/datasheets_pdf/M/M/7/4/MM74C922.shtml

[36] Relé 56.32-0300 – FINDER
www.findernet.com/comuni/pdf/S56EN.pdf

[37] LM555
www.datasheetcatalog.net/es/datasheets_pdf/L/M/5/5/LM555.shtml

ANEXOS

ANEXO A

PLANOS ELÉCTRICOS DEL SUBSISTEMA DE CONTROL

ANEXO A1

INGRESO DE DATOS
MODO LOCAL

ANEXO A2

SALIDA DE DATOS
MODO LOCAL

ANEXO A3

CIRCUITO DE DISPARO

ANEXO B

PLANOS ELÉCTRICOS DEL SUBSISTEMA DE SEMAFORIZACIÓN VEHICULAR

ANEXO C

PLANOS ELÉCTRICOS DEL SUBSISTEMA DE SEMAFORIZACIÓN PEATONAL

ANEXO D

CÓDIGO DE PROGRAMA DEL CONTROLADOR TARJETA RABBIT

```
/*****************************************************************************
*                    ESCUELA POLITÉCNICA DEL EJÉRCITO                       *
*                 INGENIERÍA ELECTRÓNICA EN AUTOMATIZACIÓN                  *
*                              TESIS DE GRADO                               *
*                                                                           *
*                        SISTEMA DE SEMAFORIZACIÓN                          *
*                                                                           *
*                          JORGE ALMEIDA GARZÓN                             *
*                          SANTIAGO MAFLA LEGARDA                           *
*****************************************************************************/

#class auto

#use "PowerCoreFLEX.lib"
#define TCPCONFIG 1
#define PORT 1301
#memmap xmem
#define FINBUFSIZE  15
#define FOUTBUFSIZE 15
#define BAUD232 2400
#define ADDRESS "132.147.160.9"
#use "dcrtcp.lib"
#define FAST_INTERRUPT 0

void print_time(unsigned long);
void program();
void salidalcd();
void mensajes(unsigned int);
void encerar();

////////////////////////////////////////////////////////////////////////

unsigned int t1, t2,t3,t4,t5,t6,t7,t8,t9,t10,p,k,xx,z,m,w,b;
unsigned int t11,t12,t13,t14,t15,t16,t17,t18,t19,t20;
unsigned int t21,t22,t23,t24,t25,t26,t27,t28,t29,t30;
unsigned int t31,t32,t33,t34,t35,t36,t37,t38,t39,t40;
unsigned int t41,t42,t43,t44,t45,t46,t47,t48,t49,t50;
unsigned int t51,t52,t53,t54,t55,t56,t57,t58,t59,t60;
unsigned int taux1, taux2,taux3,taux4,taux5,taux6,pe;
unsigned int d11,d22,me11,me22,a11,a22,h11,h22,m11,m22;
const char s1[]="Clave:";
const char s2[]="1.Fases";
const char s3[]="2.Fecha/Horarios";
const char s4[]="Tiempo:";
const char s5[]="Cargar a:";
const char s6[]="B Opc.";
const char s7[]="Ajustar Reloj";
const char s8[]="Digite minuto:";
const char s9[]="Digite Hora:";
const char s11[]="Opcion:";
const char s12[]="Intermitente";
const char s13[]="Hora inicio:";
const char s14[]="Hora fin:";
const char s15[]="Minuto inicio:";
const char s16[]="Minuto fin:";
const char s17[]="Hora pico1:";
const char s18[]="Hora pico2:";
const char s19[]="Hora pico3:";
const char s20[]="C Aceptar";
const char s21[]="A Programar otro";
```

```
char       buffer[109];
//char  lcdfin[6];
char  salidalcd1[9];
char  salidalcd2[6];
const char          sal0[]="cero";
const char          sal1[]="uno";
const char          sal2[]="dos";
const char          sal3[]="tres";
const char          sal4[]="cuatro";
const char          sal5[]="cinco";
const char          sal6[]="seis";
const char          sal7[]="siete";
const char          sal8[]="ocho";
const char          sal9[]="nueve";
//int       entrada[11];
void my_isr0();

void main()
{
  int bytes_read;
  struct tm                 rtc;                                          // time struct
          unsigned long     tempo;   // variable used to cycle thru all the LED's
  auto unsigned int i,n,j,a,x,y,clave,tiempo,dia,hora,minuto;
  longword  destIP;
  tcp_Socket socket;
  brdInit();
  sock_init();
  destIP=resolve(ADDRESS);
 // tcp_open(&socket,1401,destIP,PORT,NULL);
  //sock_wait_established(&socket,60,tcp_tick(NULL);, int *status );
/*   while(!sock_established(&socket) && sock_bytesready(&socket)==-1) {
                    tcp_tick(NULL);
          }*/

          WrPortI(PEDDR, &PEDDRShadow, 0x00);             // set port E as all inputs
  #if __SEPARATE_INST_DATA__ && FAST_INTERRUPT
                    interrupt_vector ext0_intvec my_isr0;
          #else
                    SetVectExtern3000(0, my_isr0);
          // re-setup ISR's to show example of retrieving ISR address using GetVectExtern3000
                    SetVectExtern3000(0, GetVectExtern3000(0));
          #endif

          WrPortI(I0CR, &I0CRShadow, 0x09);                 // enable external INT0 on PE0, rising edge,
priority 1
  serFopen(BAUD232);
  serMode(0);
  WrPortI(SPCR, &SPCRShadow, 0x84);
  //WrPortI(PDDDR, 0, 0xFF);
  WrPortI(PBDDR, 0, 0xFF);
  WrPortI(PFDDR, 0, 0xFF);
  WrPortI(PGDDR, 0, 0xFC);
  WrPortI(PGFR, 0, 0x00);
  WrPortI(PFFR, 0, 0x00);
  //WrPortI(PDFR, 0, 0x00);*/
  //ledOut(0,1);
  k=12;
  salidalcd1[0]=0;
  salidalcd1[1]=0;
  salidalcd1[2]=0;
```

```
salidalcd1[3]=0;
salidalcd1[4]=0;
salidalcd1[5]=0;
salidalcd1[6]=0;
salidalcd1[7]=0;
salidalcd1[8]=0;
salidalcd2[0]=0;
salidalcd2[1]=0;
salidalcd2[2]=0;
salidalcd2[3]=0;
salidalcd2[4]=0;
salidalcd2[5]=0;
pe=0;
t30=5;
t9=1000;
clave=0;
tiempo=0;
tempo=0;
minuto=0;
hora=0;
        t1=0;
t2=0;
t3=0;
t4=0;
i=0;
j=0;
a=0;
p=0;
z=16;
y=0x12;
x=0x12;
d11=0;
d22=0;
me11=0;
me22=0;
a11=0;
a22=0;
h11=0;
h22=0;
m11=0;
m22=0;
serFwrFlush();
        serFrdFlush();
WrPortI(PFDR, 0, 0x00);
WrPortI(PADR, 0, 0x00);
WrPortI(PBDR, 0, 0x00);
mensajes(31);
for (;;) {      //Bucle Infinito
                    tcp_tick(NULL);
  if(sock_established(&socket)==1){
    if(sock_dataready(&socket)>0){
        sock_read(&socket,buffer,109);
        program();
      printf("%d\n",t30);
      sock_write(&socket,buffer,109);
    }}
                    else{tcp_open(&socket,1401,destIP,PORT,NULL);}

//-----------fase1----------------------------
                    if (t1==0){k=3;}
```

```
//-----------inicio pausa 1--------------
      while(k==0){
    // ----prender fase1----
      WrPortI(PFDR, 0, 0x4C);
     // WrPortI(PADR, 0, 0x49);
      WrPortI(PBDR, 0, 0x08);
    //------comprobacion de llegada de Dato-------
      tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
            if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
          if (pe==0) {
              sock_write(&socket,sal0,4);
            pe=1;
            }
          if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
              sock_read(&socket,buffer,109);
              program();                  //programación
            printf("%d\n",t1);
            sock_write(&socket,buffer,109);
          }
      }
      costate {
          if (i==(t1-1)) {k++;pe=0;}
    //-----inicio horario-------
    //-----fin horario-------
          i++;
          if (k==0){printf("%d\n",i);
          printf("%d\n",t1);}
          else {i=0;}
          salidalcd();
          waitfor(DelayMs(1000)); // wait one second
          }//fin costate
      } // fin while(k)
//-----------fin pausa1----------------
while(k==1){
    // ----prender fase1----
      WrPortI(PFDR, 0, 0x4A);
   //   WrPortI(PADR, 0, 0x49);
      WrPortI(PBDR, 0, 0x08);
    //------comprobacion de llegada de Dato-------
      tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
            if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
          if (pe==0) {
              sock_write(&socket,sal1,3);
            pe=1;
            }
          if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
              sock_read(&socket,buffer,109);
              program();                  //programación
            printf("%d\n",t30);
            sock_write(&socket,buffer,109);
          }}
                      costate {
          if (i==(t30-1)) {
                                          k++;
            pe=0;
    //-----inicio monitoreo-------
            }
    //-----fin monitoreo-------
          i++;
          if (k==1){printf("%d\n",i);
```

```
        printf("%d\n",t30);}
        else {i=0;}
        salidalcd();
        waitfor(DelayMs(1000)); // wait one second
        }//fin costate
    } // fin while(k)
//-----------fin pausa2----------------

//-----------inicio pausa 3--------------
   while(k==2){
   // ----prender fase1----
     WrPortI(PFDR, 0, 0x49);
  //   WrPortI(PADR, 0, 0x49);
     WrPortI(PBDR, 0, 0x08);
           costate {
        if (i==1){k++;}
        if (k==2){printf("%d\n",i);
        printf("%d\n",t1);}
        else {i=0;}
        //salidalcd();
        i++;
        waitfor(DelayMs(t9)); // wait one second
        }//fin costate
    } // fin while(k)
//-----------fin pausa3----------------
//-----------fin fase 1----------------

//-----------fase2---------------------------
                    if (t2==0){k=6;}
//-----------inicio pausa 1--------------
     while(k==3){
   // ----prender fase2----
     WrPortI(PFDR, 0, 0x61);
     //WrPortI(PADR, 0, 0x89);
     WrPortI(PBDR, 0, 0x08);
    //------comprobacion de llegada de Dato-------
     tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
           if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
       if (pe==0) {
           sock_write(&socket,sal2,3);
         pe=1;
         }
       if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
           sock_read(&socket,buffer,109);
           program();                //programación
         printf("%d\n",t2);
         sock_write(&socket,buffer,109);
       }}
                    costate {
        if (i==(t2-1)) {
                                        k++;
         pe=0;
   //-----inicio monitoreo-------
         }
   //-----fin monitoreo-------
        i++;
        if (k==3){printf("%d\n",i);
        printf("%d\n",t2);}
        else {i=0;}
        salidalcd();
```

```
        waitfor(DelayMs(1000)); // wait one second
        }//fin costate
     } // fin while(k)
//-----------fin pausa1----------------
//-----------inicio pausa 2------------
                    while(k==4){
   // ----prender fase2----
          WrPortI(PFDR, 0, 0x51);
      //   WrPortI(PADR, 0, 0x89);
          WrPortI(PBDR, 0, 0x08);

   //------comprobacion de llegada de Dato-------
          tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
                  if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
         if (pe==0) {
          sock_write(&socket,sal3,4);
         pe=1;
         }
         if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
                  sock_read(&socket,buffer,109);
                  program();                //programación
          printf("%d\n",t30);
          sock_write(&socket,buffer,109);
          }}
                              costate {
          if (i==(t30-1)) {
                                                  k++;
           pe=0;
   //-----inicio monitoreo-------
         }
   //-----fin monitoreo-------
        i++;
        if (k==4){printf("%d\n",i);
        printf("%d\n",t30);}
        else {i=0;}
        salidalcd();
        waitfor(DelayMs(1000)); // wait one second
        }//fin costate
     } // fin while(k)
//-----------fin pausa2----------------

//-----------inicio pausa 3--------------
   while(k==5){
   // ----prender Rojos----
     WrPortI(PFDR, 0, 0x49);
   //  WrPortI(PADR, 0, 0x49);
     WrPortI(PBDR, 0, 0x08);
          costate {
        if (i==1){k++;}
        if (k==5){printf("%d\n",i);
        printf("%d\n",t9);}
        else {i=0;}
        //salidalcd();
        i++;
        waitfor(DelayMs(t9)); // wait one second
        }//fin costate
     } // fin while(k)
//-----------fin pausa3----------------
//-----------fin fase 2----------------
```

```
//-----------fase3---------------------------
                        if (t3==0){k=9;}
//-----------inicio pausa 1--------------
      while(k==6){
    // ----prender fase2----
      WrPortI(PFDR, 0, 0x09);
      //WrPortI(PADR, 0, 0x4c);
      WrPortI(PBDR, 0, 0x0C);
    //------comprobacion de llegada de Dato-------
      tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
            if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
          if (pe==0) {
              sock_write(&socket,sal4,6);
            pe=1;
            }
          if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
              sock_read(&socket,buffer,109);
              program();                  //programación
            printf("%d\n",t2);
            sock_write(&socket,buffer,109);
          }}
                        costate {
          if (i==(t3-1)) {
                                              k++;
            pe=0;
    //-----inicio monitoreo-------
            }
    //-----fin monitoreo-------
          i++;
          if (k==6){printf("%d\n",i);
          printf("%d\n",t3);}
          else {i=0;}
          salidalcd();
          waitfor(DelayMs(1000)); // wait one second
          }//fin costate
      } // fin while(k)
//-----------fin pausa1-----------------
//-----------inicio pausa 2------------
                        while(k==7){
    // ----prender fase2----
              WrPortI(PFDR, 0, 0x89);
       //     WrPortI(PADR, 0, 0x4a);
              WrPortI(PBDR, 0, 0x08);
    //------comprobacion de llegada de Dato-------
              tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
                        if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
            if (pe==0) {
              sock_write(&socket,sal5,5);
            pe=1;
            }
            if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
                        sock_read(&socket,buffer,109);
                        program();                  //programación
              printf("%d\n",t30);
              sock_write(&socket,buffer,109);
              }}
                                costate {
              if (i==(t30-1)) {
```

```
                                                    k++;
              pe=0;
    //-----inicio monitoreo-------
            }
    //-----fin monitoreo-------
          i++;
          if (k==7){printf("%d\n",i);
          printf("%d\n",t30);}
          else {i=0;}
          salidalcd();
          waitfor(DelayMs(1000)); // wait one second
          }//fin costate
      } // fin while(k)
//-----------fin pausa2----------------

//-----------inicio pausa 3--------------
    while(k==8){
    // ----prender Rojos----
      WrPortI(PFDR, 0, 0x49);
    //  WrPortI(PADR, 0, 0x49);
      WrPortI(PBDR, 0, 0x08);
            costate {
          if (i==1){k++;}
          if (k==8){printf("%d\n",i);
          printf("%d\n",t9);}
          else {i=0;}
          //salidalcd();
          i++;
          waitfor(DelayMs(t9)); // wait one second
          }//fin costate
      } // fin while(k)
//-----------fin pausa3----------------
//-----------fin fase 3----------------



//-----------fase4---------------------------
                        if (t4==0){k=12;}
//-----------inicio pausa 1--------------
      while(k==9){
    // ----prender fase4----
      WrPortI(PFDR, 0, 0x49);
     // WrPortI(PADR, 0, 0x61);
      WrPortI(PBDR, 0, 0x20);
    //------comprobacion de llegada de Dato-------
      tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
            if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
        if (pe==0) {
            sock_write(&socket,sal6,4);
          pe=1;
          }
        if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
            sock_read(&socket,buffer,96);
            program();                  //programación
          printf("%d\n",t2);
          sock_write(&socket,buffer,96);
        }}
                        costate {
         if (i==(t4-1)) {
                                          k++;
```

```
          pe=0;
    //-----inicio monitoreo-------
          }
    //-----fin monitoreo-------
        i++;
        if (k==9){printf("%d\n",i);
        printf("%d\n",t4);}
        else {i=0;}
        salidalcd();
        waitfor(DelayMs(1000)); // wait one second
        }//fin costate
     } // fin while(k)
//-----------fin pausa1----------------
//-----------inicio pausa 2------------
                    while(k==10){
    // ----prender fase4----
          WrPortI(PFDR, 0, 0x49);
     //   WrPortI(PADR, 0, 0x51);
          WrPortI(PBDR, 0, 0x10);
    //------comprobacion de llegada de Dato-------
          tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
                    if(sock_established(&socket)==1){  //verifica si se establecio la comunicación
          if (pe==0) {
           sock_write(&socket,sal7,5);
          pe=1;
          }
          if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
                    sock_read(&socket,buffer,109);
                    program();                //programación
           printf("%d\n",t30);
           sock_write(&socket,buffer,109);
           }}
                              costate {
          if (i==(t30-1)) {
                                                  k++;
           pe=0;
    //-----inicio monitoreo-------
          }
    //-----fin monitoreo-------
        i++;
        if (k==10){printf("%d\n",i);
        printf("%d\n",t30);}
        else {i=0;}
        salidalcd();
        waitfor(DelayMs(1000)); // wait one second
        }//fin costate
     } // fin while(k)
//-----------fin pausa2----------------

//-----------inicio pausa 3--------------
    while(k==11){
    // ----prender Rojos----
      WrPortI(PFDR, 0, 0x49);
      //WrPortI(PADR, 0, 0x49);
      WrPortI(PBDR, 0, 0x08);
          costate {
        if (i==1){k=k+3;}
        if (k==11){printf("%d\n",i);
        printf("%d\n",t9);}
        else {i=0;}
```

```
          //salidalcd();
          i++;
          waitfor(DelayMs(t9)); // wait one second
          }//fin costate
      } // fin while(k)
//-----------fin pausa3----------------
//-----------fin fase 4----------------
     if (t1==0 && t2==0 && t3==0 && t4==0){k=12;}
//------------intermitente----------------
                        while(k==12){
    // ----prender fase----
     // if(sock_established(&socket)==0){  //verifica si se entablo la comunicación
       //  sock_close(&socket);}
      WrPortI(PFDR, 0, y);
      //WrPortI(PADR, 0, x);
      WrPortI(PBDR, 0, x);
    //------comprobacion de llegada de Dato-------
      tcp_tick(NULL);    //obtiene los datos del controlador del modulo TCP
            if(sock_established(&socket)==1){  //verifica si se entablo la comunicación
            if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
            sock_read(&socket,buffer,109);
            program();                //programación
           printf("%d\n",t30);
           sock_write(&socket,buffer,109);
         }}
      if (xx==1){if(sock_established(&socket)!=1){tcp_open(&socket,1401,destIP,PORT,NULL);}}
      if (xx>9){if(sock_established(&socket)!=1){sock_close(&socket);xx=0;}}
      // else{tcp_open(&socket,1401,destIP,PORT,NULL);}
      costate {
         if (a==0){
           if(sock_established(&socket)==1){sock_write(&socket,sal9,5);}
           y=0;
           x=0;
           a=1;}
         else{
           if(sock_established(&socket)==1){sock_write(&socket,sal8,4);}
           y=0x92;
           x=0x10;
           a=0;}
         //a=~a;
         if (k==12){printf("\nintermitente");xx=xx+1;}
         salidalcd();
     //    if(sock_established(&socket)==0){  //verifica si se entablo la comunicación
     //    sock_close(&socket);}
         waitfor(DelayMs(1000)); // wait one second
         }//fin costate
      } // fin while(k)
//-----------fin intermitente----------------
       if(sock_established(&socket)==1){  //verifica si se entablo la comunicación
            if(sock_dataready(&socket)>0){   //verifica si ya esta listo un dato en el socket
            sock_read(&socket,buffer,109);
            program();                //programación
          // printf("%d\n",t30);
           sock_write(&socket,buffer,109);
        }
       }
       else{sock_close(&socket);}

//-----------inicio de menu local------------
            while(k==13){
```

```
if(z==15){
  mensajes(31);
  for(i=0; i<60000; i++);
  printf("\n Clave:");
  //serFputs(salidalcd1);
  mensajes(10);
  z=16;
  b=1;
  clave=0;
      }
if (b==1){
  if(z<10){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
     if(j<3){
       clave=(clave*10)+z;
       z=16;
       j=j+1;
     }
     else{
       clave=(clave*10)+z;
       z=16;
       j=0;
       mensajes(31);
       if (clave==1234){
                                b=2;
         for(i=0; i<20000; i++);
         printf("\n1.Fases");
         mensajes(11);
         for(i=0; i<20000; i++);
         mensajes(32);
         printf("\n2.Fecha/Horarios");
         mensajes(12);
                                     }
       else{
         k=0;
                clave=0;
                b=0;
              }
            }
  }
}
if (b==2){
  if(z==1||z==2){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
     if(z==1){
       z=16;
       j=0;
       b=3;
       mensajes(31);
       for(i=0; i<20000; i++);
       printf("\nTiempo:");
       mensajes(13);
       tiempo=0;
     }
     if(z==2){
       z=16;
```

```
        j=0;
        b=6;
        mensajes(31);
        for(i=0; i<20000; i++);
        printf("\nOpcion:");
        mensajes(19);
      }
   }
}

if (b==3){
  if(z<10){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
     if(j<1){
        tiempo=(tiempo*10)+z;
        z=16;
        j=j+1;
     }
     else{
        tiempo=(tiempo*10)+z;
        j=0;
        z=16;
        mensajes(31);
        for(i=0; i<20000; i++);
        printf("\n Cargar a:");
        mensajes(14);
        b=4;
              }
  }
}
if (b==4){
  if(z<7){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
     if(z==1){
       t1=tiempo;
        printf("%d", t1);}
     if(z==2){
       t2=tiempo;
        printf("%d", t2);}
     if(z==3){
       t3=tiempo;
        printf("%d", t3);}
     if(z==4){
       t4=tiempo;
        printf("%d", t4);}
     if(z==5){
       t30=tiempo;
        printf("%d", t30);}
     if(z==6){
        t9=tiempo*10;
        printf("%d", t9);}
     if(z==1 || z==2 || z==3 || z==4 || z==5 || z==6){
        printf("\n A programar otro");
        mensajes(31);
        for(i=0; i<20000; i++);
        mensajes(30);
```

```
        printf("\n C aceptar");
        for(i=0; i<20000; i++);
        mensajes(32);
        for(i=0; i<20000; i++);
        mensajes(29);
        z=16;
        j=0;
        b=5;
      }
    }
  }
  if (b==5){
    if(z==10 || z==12){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      if(z==10){
        z=16;
        j=0;
        b=3;
        tiempo=0;
        mensajes(31);
        for(i=0; i<20000; i++);
        printf("\n Tiempo:");}
        mensajes(13);
      if(z==12){
        i=0;
        j=0;
        encerar();
        printf("/n chauuu");}
    }
  }
  if (b==6){
    if(z==1||z==2||z==3||z==4||z==5){
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      mensajes(z);
      //k=12;
      if(z==1){
        b=7;
        tempo=0;
        mensajes(31);
        for(i=0; i<20000; i++);
                                        printf("\nAjustar Reloj");
        mensajes(16);
        mensajes(32);
        for(i=0; i<20000; i++);
        printf("\nDigite hora:");
        mensajes(18);
        tiempo=0;
      }
      if(z==2){
        b=8;
        mensajes(31);
        for(i=0; i<20000; i++);
        printf("\nHorario Intermitente");
        mensajes(21);
        mensajes(32);
        for(i=0; i<20000; i++);
        printf("\nHora Inicio:");
```

```
      mensajes(22);
    }
    if(z==3){
      b=9;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nHorario Pico 1");
      mensajes(26);
      mensajes(32);
      for(i=0; i<20000; i++);
      printf("\nHora Inicio:");
      mensajes(22);
      tiempo=0;
    }
    if(z==4){
      b=10;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nHorario Pico 2");
      mensajes(27);
      mensajes(32);
      for(i=0; i<20000; i++);
      printf("\nHora Inicio:");
      mensajes(22);
      tiempo=0;
    }
    if(z==5){
      b=11;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nHorario Pico 3");
      mensajes(28);
      mensajes(32);
      for(i=0; i<20000; i++);
      printf("\nHora Inicio:");
      mensajes(22);
      tiempo=0;
    }
    minuto=0;
                                hora=0;
    z=16;
    j=0;
  }
}
if (b==7){
  if(j==0){
    if(z<3){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      hora=(hora*10)+z;
      z=16;
      j=j+1;
      }
  }
  if(j==1){
     if(z<10){
      mensajes(z);
      for(i=0; i<60000; i++);
                            printf("%d",z);
```

```
            hora=(hora*10)+z;
           z=16;
           j=j+1;
            printf("\nhora:%d",hora);
            //mensajes(19);
            mensajes(31);
            for(i=0; i<20000; i++);
            printf("\nDigite minuto:");
            mensajes(17);
          }
                }
        if(j==2){
          if(z<6){
            mensajes(z);
            for(i=0; i<60000; i++);
           printf("\n%d",z);
            minuto=(minuto*10)+z;
            z=16;
            j=j+1;
          }
        }
        if(j==3){
          if(z<10){
            mensajes(z);
            for(i=0; i<60000; i++);
           printf("\n%d",z);
            minuto=(minuto*10)+z;
            printf("\minuto:%d",minuto);
            z=16;
            j=j+1;
//-------------------change the time---------------------------------------
                                                rtc.tm_sec = 00;
          // change the time
            rtc.tm_min = minuto;
                                                rtc.tm_hour = hora;
                                                rtc.tm_mday = 1;
                                                rtc.tm_mon = 1;
                                                rtc.tm_year =107;
//              tempo=0;
                                                tm_wr(&rtc);
                        // set clock
                                                tempo = mktime(&rtc);
                                                printf("Setting date/time to ");
                                                print_time(tempo);
            mensajes(31);
            for(i=0; i<20000; i++);
            printf("\n B Opciones");
            mensajes(15);
            mensajes(32);
            for(i=0; i<20000; i++);
            printf("\n C aceptar");
            mensajes(29);
          }
        }
        if(j==4){
          if(z==11 || z==12){
            mensajes(z);
            for(i=0; i<60000; i++);
                    printf("\n%d",z);
            if(z==11){
```

```
                j=0;
                 b=6;
          tiempo=0;
          n=0;
          mensajes(31);
          for(i=0; i<20000; i++);
          printf("\nOpcion:");
          mensajes(19);}
              if(z==12){
          i=0;
          j=0;
          encerar();
                  printf("/n chauuu");}
        z=16;
     }
   }
}
if (b==8){
  if(j==0){
     if(z<3){
        mensajes(z);
        for(i=0; i<60000; i++);
        printf("\n%d",z);
        hora=(hora*10)+z;
        z=16;
        j=j+1;
        }
  }
  if(j==1){
      if(z<10){
      mensajes(z);
                  for(i=0; i<60000; i++);
                              printf("%d",z);
                              hora=(hora*10)+z;
       t10=hora;
       hora=0;
       z=16;
      j=j+1;
       mensajes(31);
       for(i=0; i<20000; i++);
       printf("\nhora inicio:%d",t10);
       printf("\nMinuto inicio:");
       mensajes(24);
     }
           }
  if(j==2){
     if(z<6){
        mensajes(z);
       for(i=0; i<60000; i++);
       printf("\n%d",z);
        minuto=(minuto*10)+z;
        z=16;
        j=j+1;
     }
  }
  if(j==3){
     if(z<10){
        mensajes(z);
        for(i=0; i<60000; i++);
       printf("\n%d",z);
```

```
      minuto=(minuto*10)+z;
      t11=minuto;
      minuto=0;
      z=16;
      j=j+1;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nminuto inicio:%d",t11);
      printf("\nHora fin:");
      mensajes(23);
    }
}
if(j==4){
   if(z<3){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      hora=(hora*10)+z;
      z=16;
      j=j+1;
      }
}
if(j==5){
     if(z<10){
      mensajes(z);
      for(i=0; i<60000; i++);
                              printf("%d",z);
      hora=(hora*10)+z;
      t12=hora;
      hora=0;
      z=16;
     j=j+1;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nhora fin:%d",t12);
      printf("\nMinuto fin:");
      mensajes(25);
   }
          }
if(j==6){
   if(z<6){
      mensajes(z);
      for(i=0; i<60000; i++);
     printf("\n%d",z);
      minuto=(minuto*10)+z;
      z=16;
      j=j+1;
   }
}
if(j==7){
   if(z<10){
      mensajes(z);
      for(i=0; i<60000; i++);
     printf("\n%d",z);
      minuto=(minuto*10)+z;
      t13=minuto;
      minuto=0;
      z=16;
      j=j+1;
      mensajes(31);
```

```
      for(i=0; i<20000; i++);
      printf("\nminuto fin:%d",t13);
      printf("\n B Opciones");
      mensajes(15);
      mensajes(32);
      for(i=0; i<20000; i++);
      printf("\n C aceptar");
      mensajes(29);
    }
  }
  if(j==8){
    if(z==11 || z==12){
      mensajes(z);
      for(i=0; i<60000; i++);
              printf("\n%d",z);
      if(z==11){
        j=0;
                b=6;
        tiempo=0;
        n=0;
        mensajes(31);
        for(i=0; i<20000; i++);
             printf("\nOpcion:");
        mensajes(15);}
           if(z==12){
        i=0;
        j=0;
        encerar();
                printf("/n chauuu");}
      z=16;
         }
  }
}
if (b==9){
  if(j==0){
    if(z<3){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      hora=(hora*10)+z;
      z=16;
      j=j+1;
      }
  }
  if(j==1){
     if(z<10){
      mensajes(z);
      for(i=0; i<60000; i++);
                          printf("%d",z);
                          hora=(hora*10)+z;
      t18=hora;
      hora=0;
      z=16;
     j=j+1;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nhora inicio:%d",t18);
      printf("\nMinuto inicio:");
      mensajes(24);
    }
```

```
        }
if(j==2){
  if(z<6){
                                        mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
     minuto=(minuto*10)+z;
     z=16;
     j=j+1;
  }
}
if(j==3){
  if(z<10){
                                        mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
     minuto=(minuto*10)+z;
     t19=minuto;
     minuto=0;
     z=16;
     j=j+1;
     mensajes(31);
     for(i=0; i<20000; i++);
     printf("\nminuto inicio:%d",t19);
     printf("\nHora fin:");
     mensajes(23);
  }
}
if(j==4){
  if(z<3){
                                        mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
     hora=(hora*10)+z;
     z=16;
     j=j+1;
     }
}
if(j==5){
    if(z<10){
                                        mensajes(z);
              for(i=0; i<60000; i++);
                        printf("%d",z);
                        hora=(hora*10)+z;
     t20=hora;
     hora=0;
     z=16;
    j=j+1;
     mensajes(31);
     for(i=0; i<20000; i++);
     printf("\nhora fin:%d",t20);
     printf("\nMinuto fin:");
     mensajes(25);
  }
        }
if(j==6){
  if(z<6){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
```

```
      minuto=(minuto*10)+z;
      z=16;
      j=j+1;
    }
  }
  if(j==7){
    if(z<10){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
       minuto=(minuto*10)+z;
       t21=minuto;
       minuto=0;
       z=16;
       j=j+1;
       n=0;
       //b=0;
       //k=12;
       mensajes(31);
       for(i=0; i<20000; i++);
       printf("\nminuto fin:%d",t21);
       printf("\nTiempo:");
       mensajes(13);
    }
  }
  if (j==8){
      if(z<10){
                                                    mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      if(n<1){
                  tiempo=(tiempo*10)+z;
                  z=16;
                  n=n+1;
      }
      else{
                  tiempo=(tiempo*10)+z;
         n=0;
                z=16;
         mensajes(31);
                  for(i=0; i<20000; i++);
              printf("\n Cargar a:");
         mensajes(14);
           j=j+1;
                  }
              }
         }
      if (j==9){
            if(z==1 || z==2 || z==3 || z==4){
       mensajes(z);
       for(i=0; i<60000; i++);
           printf("\n%d",z);
        if(z==1){
                  t31=tiempo;
                printf("%d", t31);}
          if(z==2){
                  t32=tiempo;
       printf("%d", t32);}
              if(z==3){
                t33=tiempo;
```

```
            printf("%d", t33);}
        if(z==4){
                t34=tiempo;
                 printf("%d", t34);}
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\n A programar otro");
      mensajes(30);
      mensajes(32);
      for(i=0; i<20000; i++);
      printf("\n B Opciones");
      mensajes(15);
      printf("\n C aceptar");
      mensajes(29);
      z=16;
      j=j+1;
      tiempo=0;
      }
      }
          if (j==10){
          if(z==10 || z==11 || z==12){
      mensajes(z);
      for(i=0; i<60000; i++);
              printf("\n%d",z);
      n=0;
      if(z==10){
          j=8;
                tiempo=0;
        mensajes(31);
                 for(i=0; i<20000; i++);
                 printf("\n Tiempo:");
        mensajes(13);}
      if(z==11){
                j=0;
                 b=6;
        mensajes(31);
                 for(i=0; i<20000; i++);
               printf("\nOpcion:");
        mensajes(15);}
            if(z==12){
        i=0;
        j=0;
        encerar();
                 printf("/n chauuu");}
      z=16;
    }
     }
}
if (b==10){
  if(j==0){
    if(z<3){
      mensajes(z);
       for(i=0; i<60000; i++);
       printf("\n%d",z);
       hora=(hora*10)+z;
       z=16;
       j=j+1;
       }
  }
  if(j==1){
```

```
    if(z<10){
    mensajes(z);
                for(i=0; i<60000; i++);
                            printf("%d",z);
                            hora=(hora*10)+z;
     t22=hora;
     hora=0;
     z=16;
    j=j+1;
     mensajes(31);
     for(i=0; i<20000; i++);
     printf("\nhora inicio:%d",t22);
     printf("\nMinuto inicio:");
     mensajes(24);
   }
         }
if(j==2){
  if(z<6){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
     minuto=(minuto*10)+z;
     z=16;
     j=j+1;
  }
}
if(j==3){
  if(z<10){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
     minuto=(minuto*10)+z;
     t23=minuto;
     minuto=0;
     z=16;
     j=j+1;
     mensajes(31);
     for(i=0; i<20000; i++);
     printf("\nminuto inicio:%d",t23);
     printf("\nHora fin:");
     mensajes(23);
  }
}
if(j==4){
  if(z<3){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
     hora=(hora*10)+z;
     z=16;
     j=j+1;
     }
}
if(j==5){
    if(z<10){
    mensajes(z);
                for(i=0; i<60000; i++);
                            printf("%d",z);
                            hora=(hora*10)+z;
     t24=hora;
```

```
      hora=0;
      z=16;
     j=j+1;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nhora fin:%d",t24);
      printf("\nMinuto fin:");
      mensajes(25);
   }
         }
if(j==6){
  if(z<6){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
     minuto=(minuto*10)+z;
     z=16;
     j=j+1;
  }
}
if(j==7){
  if(z<10){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
     minuto=(minuto*10)+z;
     t25=minuto;
     minuto=0;
     z=16;
     j=j+1;
     n=0;
     //b=0;
     //k=12;
     mensajes(31);
     for(i=0; i<20000; i++);
     printf("\nminuto fin:%d",t25);
     printf("\nTiempo:");
     mensajes(13);
  }
}
if (j==8){
    if(z<10){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
    if(n<1){
                tiempo=(tiempo*10)+z;
                z=16;
                n=n+1;
    }
    else{
                tiempo=(tiempo*10)+z;
       n=0;
              z=16;
       mensajes(31);
                for(i=0; i<20000; i++);
           printf("\n Cargar a:");
       mensajes(14);
         j=j+1;
                }
```

```
        }
    }
if (j==9){
        if(z==1 || z==2 || z==3 || z==4){
mensajes(z);
            for(i=0; i<60000; i++);
     printf("\n%d",z);
   if(z==1){
            t41=tiempo;
           printf("%d", t41);}
      if(z==2){
            t42=tiempo;
   printf("%d", t42);}
          if(z==3){
            t43=tiempo;
        printf("%d", t43);}
   if(z==4){
            t44=tiempo;
             printf("%d", t44);}
 mensajes(31);
 for(i=0; i<20000; i++);
 printf("\n A programar otro");
 mensajes(30);
 mensajes(32);
 for(i=0; i<20000; i++);
 printf("\n B Opciones");
 mensajes(15);
 printf("\n C aceptar");
 mensajes(29);
 z=16;
 j=j+1;
 tiempo=0;
}
}
     if (j==10){
     if(z==10 || z==11 || z==12){
mensajes(z);
            for(i=0; i<60000; i++);
           printf("\n%d",z);
 n=0;
 if(z==10){
     j=8;
            tiempo=0;
   mensajes(31);
             for(i=0; i<20000; i++);
             printf("\n Tiempo:");
   mensajes(13);}
 if(z==11){
            j=0;
             b=6;
   mensajes(31);
             for(i=0; i<20000; i++);
           printf("\nOpcion:");
   mensajes(15);}
        if(z==12){
   i=0;
   j=0;
   encerar();
             printf("/n chauuu");}
 z=16;
```

```
    }
     }
}
if (b==11){
  if(j==0){
    if(z<3){
      mensajes(z);
       for(i=0; i<60000; i++);
       printf("\n%d",z);
       hora=(hora*10)+z;
       z=16;
       j=j+1;
      }
  }
  if(j==1){
      if(z<10){
      mensajes(z);
                for(i=0; i<60000; i++);
                          printf("%d",z);
                          hora=(hora*10)+z;
       t26=hora;
       hora=0;
       z=16;
      j=j+1;
       mensajes(31);
       for(i=0; i<20000; i++);
       printf("\nhora inicio:%d",t26);
       printf("\nMinuto inicio:");
       mensajes(24);
    }
           }
  if(j==2){
    if(z<6){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
       minuto=(minuto*10)+z;
       z=16;
       j=j+1;
    }
  }
  if(j==3){
    if(z<10){
      mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
       minuto=(minuto*10)+z;
       t27=minuto;
       minuto=0;
       z=16;
       j=j+1;
       mensajes(31);
       for(i=0; i<20000; i++);
       printf("\nminuto inicio:%d",t27);
       printf("\nHora fin:");
       mensajes(23);
    }
  }
  if(j==4){
    if(z<3){
```

```
     mensajes(z);
      for(i=0; i<60000; i++);
      printf("\n%d",z);
      hora=(hora*10)+z;
      z=16;
      j=j+1;
     }
}
if(j==5){
    if(z<10){
    mensajes(z);
                for(i=0; i<60000; i++);
                        printf("%d",z);
                        hora=(hora*10)+z;
      t28=hora;
      hora=0;
      z=16;
     j=j+1;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nhora fin:%d",t28);
      printf("\nMinuto fin:");
      mensajes(25);
   }
         }
if(j==6){
  if(z<6){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
      minuto=(minuto*10)+z;
      z=16;
      j=j+1;
   }
}
if(j==7){
  if(z<10){
     mensajes(z);
     for(i=0; i<60000; i++);
     printf("\n%d",z);
      minuto=(minuto*10)+z;
      t29=minuto;
      minuto=0;
      z=16;
      j=j+1;
      n=0;
      //b=0;
      //k=12;
      mensajes(31);
      for(i=0; i<20000; i++);
      printf("\nminuto fin:%d",t29);
      printf("\nTiempo:");
      mensajes(13);
   }
}
if (j==8){
    if(z<10){
    mensajes(z);
    for(i=0; i<60000; i++);
    printf("\n%d",z);
```

```
if(n<1){
            tiempo=(tiempo*10)+z;
            z=16;
            n=n+1;
}
else{
            tiempo=(tiempo*10)+z;
   n=0;
          z=16;
   mensajes(31);
             for(i=0; i<20000; i++);
        printf("\n Cargar a:");
   mensajes(14);
     j=j+1;
            }
        }
  }
if (j==9){
       if(z==1 || z==2 || z==3 || z==4){
mensajes(z);
            for(i=0; i<60000; i++);
     printf("\n%d",z);
  if(z==1){
            t51=tiempo;
          printf("%d", t51);}
     if(z==2){
            t52=tiempo;
  printf("%d", t52);}
         if(z==3){
           t53=tiempo;
       printf("%d", t53);}
  if(z==4){
            t54=tiempo;
             printf("%d", t54);}
 mensajes(31);
 for(i=0; i<20000; i++);
 printf("\n A programar otro");
 mensajes(30);
 mensajes(32);
 for(i=0; i<20000; i++);
 printf("\n B Opciones");
 mensajes(15);
 printf("\n C aceptar");
 mensajes(29);
 z=16;
 j=j+1;
 tiempo=0;
}
}
     if (j==10){
     if(z==10 || z==11 || z==12){
mensajes(z);
            for(i=0; i<60000; i++);
          printf("\n%d",z);
 n=0;
 if(z==10){
     j=8;
            tiempo=0;
   mensajes(31);
             for(i=0; i<20000; i++);
```

```
                    printf("\n Tiempo:");
              mensajes(13);}
            if(z==11){
                    j=0;
                    b=6;
              mensajes(31);
                    for(i=0; i<20000; i++);
                  printf("\nOpcion:");
              mensajes(15);}
                 if(z==12){
              i=0;
              j=0;
              encerar();
                    printf("/n chauuu");}
            z=16;
          }
           }
     }
    WrPortI(PFDR, 0, y);
    WrPortI(PADR, 0, x);
    WrPortI(PBDR, 0, 0x00);
          costate {
       if (a==0){
          y=0;
         x=0;
         a=1;}
       else{
          y=0x12;
         x=0x12;
         a=0;}
       //a=~a;
       if (k==13){printf("\nintermitente teclado");}
       //salidalcd();
       waitfor(DelayMs(1000)); // wait one second
       }//fin costate
    }// fin del while
    k=0;
 /*   ledOut(0,1);
                    for(i=0; i<30000; i++); // time delay loop
    ledOut(0,0);
                    for(i=0; i<30000; i++); // time delay loop*/

          //t1=10;
          i=0;
          //p=0;
    }//fin for
    serFclose();
}


void program()
{
          struct tm          rtc;                                                    // time struct
  unsigned long   tiempo;  // variable used to cycle thru all the LED's
//---------------------cargar los tiempos recibidos-------------------

  t1=((buffer[13]-48)*10)+(buffer[14]-48);
  t2=((buffer[15]-48)*10)+(buffer[16]-48);
  t3=((buffer[17]-48)*10)+(buffer[18]-48);
  t4=((buffer[19]-48)*10)+(buffer[20]-48);
```

```
  t30=((buffer[21]-48)*10)+(buffer[22]-48);
/*  t6=((buffer[23]-48)*10)+(buffer[24]-48);
  t7=((buffer[25]-48)*10)+(buffer[26]-48);
  t8=((buffer[27]-48)*10)+(buffer[28]-48);*/
  t9=((buffer[23]-48)*1000)+((buffer[24]-48)*100)+((buffer[25]-48)*10)+(buffer[26]-48);
  t10=((buffer[27]-48)*10)+(buffer[28]-48);
  t11=((buffer[29]-48)*10)+(buffer[30]-48);
  t12=((buffer[31]-48)*10)+(buffer[32]-48);
  t13=((buffer[33]-48)*10)+(buffer[34]-48);
  t14=((buffer[35]-48)*10)+(buffer[36]-48);
  t15=((buffer[37]-48)*10)+(buffer[38]-48);
  t16=((buffer[39]-48)*10)+(buffer[40]-48);
  t17=((buffer[41]-48)*10)+(buffer[42]-48);
  t18=((buffer[43]-48)*10)+(buffer[44]-48);
  t19=((buffer[45]-48)*10)+(buffer[46]-48);
  t20=((buffer[47]-48)*10)+(buffer[48]-48);
  t21=((buffer[49]-48)*10)+(buffer[50]-48);
  t22=((buffer[51]-48)*10)+(buffer[52]-48);
  t23=((buffer[53]-48)*10)+(buffer[54]-48);
  t24=((buffer[55]-48)*10)+(buffer[56]-48);
  t25=((buffer[57]-48)*10)+(buffer[58]-48);
  t26=((buffer[59]-48)*10)+(buffer[60]-48);
  t27=((buffer[61]-48)*10)+(buffer[62]-48);
  t28=((buffer[63]-48)*10)+(buffer[64]-48);
  t29=((buffer[65]-48)*10)+(buffer[66]-48);
  t31=((buffer[67]-48)*10)+(buffer[68]-48);
  t32=((buffer[69]-48)*10)+(buffer[70]-48);
  t33=((buffer[71]-48)*10)+(buffer[72]-48);
  t34=((buffer[73]-48)*10)+(buffer[74]-48);
  t39=((buffer[75]-48)*10)+(buffer[76]-48);
  t40=((buffer[77]-48)*1000)+((buffer[78]-48)*100)+((buffer[79]-48)*10)+(buffer[80]-48);
  t41=((buffer[81]-48)*10)+(buffer[82]-48);
  t42=((buffer[83]-48)*10)+(buffer[84]-48);
  t43=((buffer[85]-48)*10)+(buffer[86]-48);
  t44=((buffer[87]-48)*10)+(buffer[88]-48);
  t49=((buffer[89]-48)*10)+(buffer[90]-48);
  t50=((buffer[91]-48)*1000)+((buffer[92]-48)*100)+((buffer[93]-48)*10)+(buffer[94]-48);
  t51=((buffer[95]-48)*10)+(buffer[96]-48);
  t52=((buffer[97]-48)*10)+(buffer[98]-48);
  t53=((buffer[99]-48)*10)+(buffer[100]-48);
  t54=((buffer[101]-48)*10)+(buffer[102]-48);
  t59=((buffer[103]-48)*10)+(buffer[104]-48);
  t60=((buffer[105]-48)*1000)+((buffer[106]-48)*100)+((buffer[107]-48)*10)+(buffer[108]-48);
 //t30=((buffer[73]-48)*10)+(buffer[74]-48);
  taux1=t1;
  taux2=t2;
  taux3=t3;
  taux4=t4;
  taux5=t30;
  taux6=t9;
  printf("%d\n",buffer[13]);
  if (t1==0 && t2==0 && t3==0 && t4==0){k=12;}
  else{k=0;}
  if (t30==0){t30=4;}
  printf("%d\n",t30);
  printf("%d\n",k);
//---------------Actualizar Reloj------------------
  rtc.tm_sec = ((buffer[11]-48)*10)+(buffer[12]-48);                                                    //
change the time
          rtc.tm_min = ((buffer[9]-48)*10)+(buffer[10]-48);
```

```
        rtc.tm_hour = ((buffer[7]-48)*10)+(buffer[8]-48);
        rtc.tm_mday = ((buffer[5]-48)*10)+(buffer[6]-48);
        rtc.tm_mon = ((buffer[3]-48)*10)+(buffer[4]-48);
        rtc.tm_year = ((buffer[0]-48)*100)+((buffer[1]-48)*10)+(buffer[2]-48);
        tm_wr(&rtc);                                                    // set clock
        tiempo = mktime(&rtc);
        printf("Setting date/time to ");
        print_time(tiempo);
  }
void salidalcd()
{
        struct tm       rtc1;                                   // time struct
  unsigned long   ti2;    // variable used to cycle thru all the LED's
  ti2 = read_rtc();                                             // read time in seconds since
1980
  printf("Fecha/ Hora:");         // NOTE that we use read_rtc() and not tm_rd()
        print_time(ti2);
  }
void encerar()
{
  d11=0;
  d22=0;
  me11=0;
  me22=0;
  a11=0;
  a22=0;
  h11=0;
  h22=0;
  m11=0;
  m22=0;
  k=0;
  b=0;
  z=16;
  }
//---------------mensajes--------------------
void mensajes(unsigned int h)
{
        unsigned char x;
  if(h==0){x='0';
  serFputc(x);}
  if(h==1){x='1';
  serFputc(x);}
  if(h==2){x='2';
  serFputc(x);}
  if(h==3){x='3';
  serFputc(x);}
  if(h==4){x='4';
  serFputc(x);}
  if(h==5){x='5';
  serFputc(x);}
  if(h==6){x='6';
  serFputc(x);}
  if(h==7){x='7';
  serFputc(x);}
  if(h==8){x='8';
  serFputc(x);}
  if(h==9){x='9';
  serFputc(x);}
  if(h==10){serFputs(s1);}
  if(h==11){serFputs(s2);}
```

```
  if(h==12){serFputs(s3);}
  if(h==13){serFputs(s4);}
  if(h==14){serFputs(s5);}
  if(h==15){serFputs(s6);}
  if(h==16){serFputs(s7);}
  if(h==17){serFputs(s8);}
  if(h==18){serFputs(s9);}
  if(h==19){serFputs(s11);}
  if(h==20){serFputs(s11);}
  if(h==21){serFputs(s12);}
  if(h==22){serFputs(s13);}
  if(h==23){serFputs(s14);}
  if(h==24){serFputs(s15);}
  if(h==25){serFputs(s16);}
  if(h==26){serFputs(s17);}
  if(h==27){serFputs(s18);}
  if(h==28){serFputs(s19);}
  if(h==29){serFputs(s20);}
  if(h==30){serFputs(s21);}
  if(h==31){x='*';
  serFputc(x);}
  if(h==32){x='#';
  serFputc(x);}
  if(h==33){serFputs(salidalcd1);}
  if(h==34){serFputs(salidalcd2);}

}
//---------------imprimir tiempo------------------
void print_time(unsigned long thetime)
{
  unsigned int i,d1,d2,me1,me2,a1,a2,h1,h2,m1,m2;
  struct tm          thetm;
          mktm(&thetm, thetime);
          printf("%02d/%02d/%04d %02d:%02d:%02d\n\n",
                                thetm.tm_mon, thetm.tm_mday, 1900+thetm.tm_year,
                                thetm.tm_hour, thetm.tm_min, thetm.tm_sec);
  //salidalcd1[0]=thetm.tm_mon;
  a1=((thetm.tm_year-100)/10);
//          printf("%02d/",a1);
  a2=((thetm.tm_year-100)-(a1*10));
//   printf("%02d/",a2);
  me1=(thetm.tm_mon/10);
  me2=(thetm.tm_mon-(me1*10));
  d1=(thetm.tm_mday/10);
  d2=(thetm.tm_mday-(d1*10));
  if (a11!=a1 || a22!=a2 || me11!=me1 || me22!=me2 || d11!=d1 || d22!=d2){
    mensajes(31);
    for(i=0; i<20000; i++);
    a11=a1;
          a22=a2;
          me11=me1;
                me22=me2;
          d11=d1;
          d22=d2;
          salidalcd1[0]=a1+48;
        salidalcd1[1]=a2+48;
          salidalcd1[2]=47;
        salidalcd1[3]=me1+48;
          salidalcd1[4]=me2+48;
        salidalcd1[5]=47;
```

```
            salidalcd1[6]=d1+48;
          salidalcd1[7]=d2+48;
    mensajes(33);
    for(i=0; i<20000; i++);
  }

  h1=(thetm.tm_hour/10);
  h2=(thetm.tm_hour-(h1*10));
  m1=(thetm.tm_min/10);
  m2=(thetm.tm_min-m1*10);
        if (h11!=h1 || h22!=h2 || m11!=m1 || m22!=m2){
        mensajes(32);
    for(i=0; i<20000; i++);
    h11=h1;
    h22=h2;
    m11=m1;
    m22=m2;
/*    salidalcd2[0]=32;
          salidalcd2[1]=32;
        salidalcd2[2]=32;
          salidalcd2[3]=32;
        salidalcd2[4]=32;*/
          salidalcd2[0]=(h1+48);
        salidalcd2[1]=h2+48;
          salidalcd2[2]=58;
        salidalcd2[3]=m1+48;
          salidalcd2[4]=m2+48;
    //salidalcd2[5]=58;
    mensajes(34);
    for(i=0; i<20000; i++);
  }

  if (t10!=0 || t11!=0 || t12!=0 || t13!=0){
        if (t10==thetm.tm_hour && t11==thetm.tm_min && thetm.tm_sec==0){k=12;}
        if (t12==thetm.tm_hour && t13==thetm.tm_min && thetm.tm_sec==0){
        k=0;
        //i=0;
      }
  }
        if (t14!=0 || t15!=0 || t16!=0 || t17!=0){
        if (t14==thetm.tm_hour && t15==thetm.tm_min && thetm.tm_sec==0){k=12;}
        if (t16==thetm.tm_hour && t17==thetm.tm_min && thetm.tm_sec==0){
        k=0;
        //i=0;
      }
  }
  if (t18!=0 || t19!=0 || t20!=0 || t21!=0){
    if (t18==thetm.tm_hour && t19==thetm.tm_min && thetm.tm_sec==0){
          taux1=t1;
        taux2=t2;
            taux3=t3;
          taux4=t4;
        taux5=t30;
            taux6=t9;
          t1=t31;
        t2=t32;
            t3=t33;
        t4=t34;
          t30=t39;
            t9=t40;
```

```
    k=0;
        //i=0;
        }
          if (t20==thetm.tm_hour && t21==thetm.tm_min && thetm.tm_sec==0){
          t1=taux1;
        t2=taux2;
            t3=taux3;
          t4=taux4;
        t30=taux5;
            t9=taux6;
    k=0;
        //i=0;
        }
}
if (t22!=0 || t23!=0 || t24!=0 || t25!=0){
          if (t22==thetm.tm_hour && t23==thetm.tm_min && thetm.tm_sec==0){
          taux1=t1;
        taux2=t2;
            taux3=t3;
          taux4=t4;
        taux5=t30;
            taux6=t9;
          t1=t41;
        t2=t42;
            t3=t43;
          t4=t44;
        t30=t49;
            t9=t50;
    k=0;
        //i=0;
        }
          if (t24==thetm.tm_hour && t25==thetm.tm_min && thetm.tm_sec==0){
          t1=taux1;
        t2=taux2;
            t3=taux3;
          t4=taux4;
        t30=taux5;
            t9=taux6;
    k=0;
        //i=0;
        }
}
if (t26!=0 || t27!=0 || t28!=0 || t29!=0){
  if (t26==thetm.tm_hour && t27==thetm.tm_min && thetm.tm_sec==0){
            taux1=t1;
        taux2=t2;
            taux3=t3;
          taux4=t4;
        taux5=t30;
            taux6=t9;
          t1=t51;
        t2=t52;
            t3=t53;
          t4=t54;
        t30=t59;
            t9=t60;
    k=0;
        //i=0;
        }
  if (t28==thetm.tm_hour && t2==thetm.tm_min && thetm.tm_sec==0){
```

```
            t1=taux1;
          t2=taux2;
              t3=taux3;
            t4=taux4;
          t30=taux5;
              t9=taux6;
      k=0;
        //i=0;
        }
  }
}

nodebug root interrupt void my_isr0()
{
  w=0;
  m=0;
 // for(b=0; b<60000; b++); // time delay loop
  m=RdPortI(PEDR);
//   for(b=0; b<30000; b++); // time delay loop
  w=m|0x27;
  if(w==0x27){z=0;}
  if(w==0x2f){z=1;}
  if(w==0x37){z=2;}
  if(w==0x3f){z=3;}
  if(w==0x67){z=4;}
  if(w==0x6f){z=5;}
  if(w==0x77){z=6;}
  if(w==0x7f){z=7;}
  if(w==0xa7){z=8;}
  if(w==0xaf){z=9;}
  if(w==0xb7){z=10;}
  if(w==0xbf){z=11;}
  if(w==0xe7){z=12;}
  if(w==0xef){z=13;}
  if(w==0xf7){
    encerar();
    k=12;
    }
  if(w==0xff)
        {z=15;
    k=13;}
}
```

ANEXO E

CÓDIGO DE PROGRAMA DE LA INTERFAZ GRÁFICA

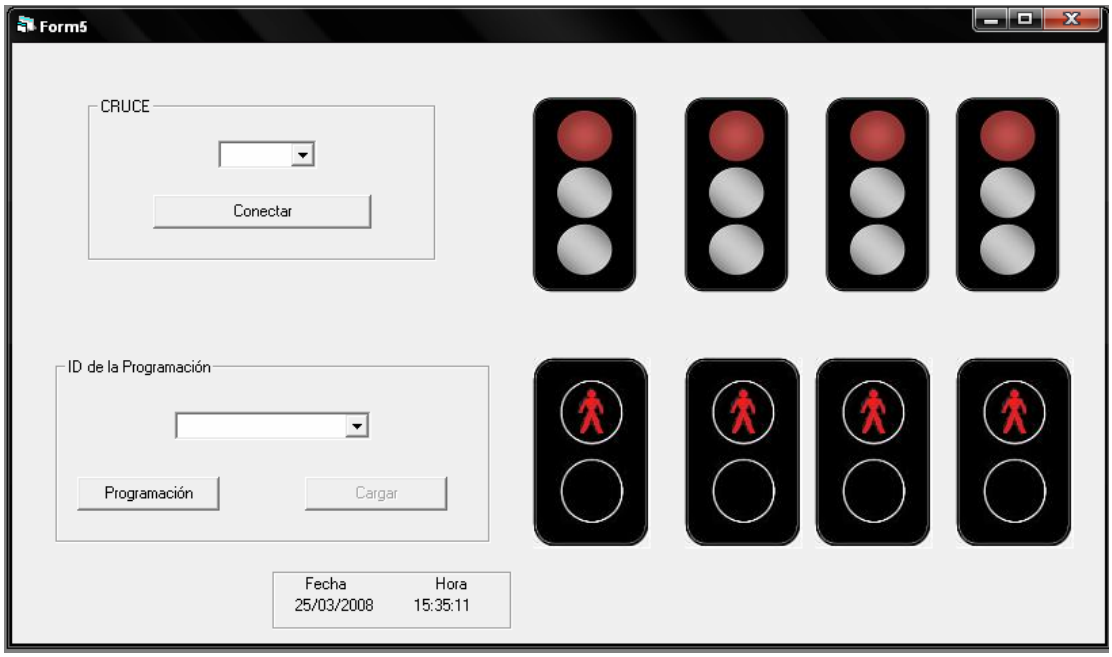
PANTALLA 1


```
Dim flap As Integer

Private Sub asemaforos()
If flap = 0 Then
   psrojo.Picture = samarillo.Picture
   flap = 1
   Exit Sub
End If
If flap = 1 Then
   psrojo.Picture = sverde.Picture
   flap = 2
   Exit Sub
End If
If flap = 2 Then
   psrojo.Picture = srojo.Picture
   flap = 0
   Exit Sub
End If

End Sub
Private Sub Form_Load()
Form1.Height = 5850
Form1.Width = 11880
Form1.Left = (Screen.Width - Width) / 2
Form1.Top = (Screen.Height - Height) / 2 - 150
End Sub

Private Sub Frame1_Click()
Form2.Show
Unload Me
End Sub

Private Sub Timer2_Timer()
    asemaforos
End Sub
```

PANTALLA 2


```
Private Sub Command1_Click()
criterio = "clave='" & Text2 & "'"
Adodc1.Recordset.MoveFirst
Adodc1.Recordset.Find criterio
If Adodc1.Recordset.EOF Then
MsgBox ("Advertencia!...Acceso Denegado..."), vbInformation, "Mensaje de Seguridad "
    Text2 = ""
    Text1 = ""
    Text1.SetFocus
Else
If Adodc1.Recordset!usuario = Text1 And Adodc1.Recordset!clave = Text2 Then
    MDIForm1.Ingreso = True
    MDIForm1.Monitoreo = True
    MDIForm1.salir = True
    Unload Me
    Form5.Show
  Else
    MsgBox ("Advertencia!...Acceso Denegado..."), vbInformation, "Mensaje de Seguridad "
    Text2 = ""
    Text1 = ""
    Text1.SetFocus
  End If
End If
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Form_Load()
Form2.Height = 2670
Form2.Width = 5070
Form2.Left = (Screen.Width - Width) / 2
Form2.Top = (Screen.Height - Height) / 2 - 150
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
  Text2.SetFocus
End If
End Sub
```

PANTALLA 3




```
Dim datos1 As String
Dim datos2 As String
Dim datos3 As String
Dim datos4 As String
Dim datos5 As String
Dim datos6 As String
Dim datos7 As String
Dim datos8 As String
Dim enviar As String
Dim ano As String
Dim mes As String
Dim dia As String
Dim hora As String
Dim minuto As String
Dim segundo As String

Private Sub Winsock1_ConnectionRequest(ByVal requestID As Long)
Winsock1.Close
Winsock1.Accept requestID
End Sub

Private Sub Winsock2_ConnectionRequest(ByVal requestID As Long)
Winsock2.Close
Winsock2.Accept requestID
End Sub

Private Sub Winsock3_ConnectionRequest(ByVal requestID As Long)
Winsock3.Close
Winsock3.Accept requestID
End Sub

Private Sub Winsock4_ConnectionRequest(ByVal requestID As Long)
Winsock4.Close
Winsock4.Accept requestID
```

```
End Sub

Private Sub Winsock5_ConnectionRequest(ByVal requestID As Long)
Winsock5.Close
Winsock5.Accept requestID
End Sub

Private Sub Winsock6_ConnectionRequest(ByVal requestID As Long)
Winsock6.Close
Winsock6.Accept requestID
End Sub

Private Sub Winsock7_ConnectionRequest(ByVal requestID As Long)
Winsock7.Close
Winsock7.Accept requestID
End Sub

Private Sub Winsock8_ConnectionRequest(ByVal requestID As Long)
Winsock8.Close
Winsock8.Accept requestID
End Sub

Private Sub Command1_Click()
If Combo1.Text = "UNO" Then
   Winsock1.Listen
End If
If Combo1.Text = "DOS" Then
   Winsock2.Listen
End If
If Combo1.Text = "TRES" Then
   Winsock3.Listen
End If
If Combo1.Text = "CUATRO" Then
   Winsock4.Listen
End If
If Combo1.Text = "CINCO" Then
   Winsock5.Listen
End If
If Combo1.Text = "SEIS" Then
   Winsock6.Listen
End If
If Combo1.Text = "SIETE" Then
   Winsock7.Listen
End If
If Combo1.Text = "OCHO" Then
   Winsock8.Listen
End If
End Sub

Private Sub Command2_Click()
Form4.Show
End Sub

Private Sub ver()
If Label13.Caption = "cero" Then
   Image1.Picture = sverde.Picture
   Image2.Picture = srojo.Picture
   Image3.Picture = srojo.Picture
   Image4.Picture = srojo.Picture
   Image5.Picture = pverde.Picture
```

```
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "uno" Then
    Image1.Picture = samarillo.Picture
    Image2.Picture = srojo.Picture
    Image3.Picture = srojo.Picture
    Image4.Picture = srojo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "dos" Then
    Image1.Picture = srojo.Picture
    Image2.Picture = sverde.Picture
    Image3.Picture = srojo.Picture
    Image4.Picture = srojo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = pverde.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "tres" Then
    Image1.Picture = srojo.Picture
    Image2.Picture = samarillo.Picture
    Image3.Picture = srojo.Picture
    Image4.Picture = srojo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "cuatro" Then
    Image1.Picture = srojo.Picture
    Image2.Picture = srojo.Picture
    Image3.Picture = sverde.Picture
    Image4.Picture = srojo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = pverde.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "cinco" Then
    Image1.Picture = srojo.Picture
    Image2.Picture = srojo.Picture
    Image3.Picture = samarillo.Picture
    Image4.Picture = srojo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "seis" Then
    Image1.Picture = srojo.Picture
    Image2.Picture = srojo.Picture
    Image3.Picture = srojo.Picture
    Image4.Picture = sverde.Picture
    Image5.Picture = projo.Picture
```

```
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = pverde.Picture
End If
If Label13.Caption = "siete" Then
    Image1.Picture = srojo.Picture
    Image2.Picture = srojo.Picture
    Image3.Picture = srojo.Picture
    Image4.Picture = samarillo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "ocho" Then
    Image1.Picture = samarillo.Picture
    Image2.Picture = samarillo.Picture
    Image3.Picture = samarillo.Picture
    Image4.Picture = samarillo.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
If Label13.Caption = "nueve" Then
    Image1.Picture = sblanco.Picture
    Image2.Picture = sblanco.Picture
    Image3.Picture = sblanco.Picture
    Image4.Picture = sblanco.Picture
    Image5.Picture = projo.Picture
    Image6.Picture = projo.Picture
    Image7.Picture = projo.Picture
    Image8.Picture = projo.Picture
End If
End Sub
Private Sub estado()
Label1.Caption = Winsock1.State
Label2.Caption = Winsock2.State
Label3.Caption = Winsock3.State
Label4.Caption = Winsock4.State
Label5.Caption = Winsock5.State
Label6.Caption = Winsock6.State
Label7.Caption = Winsock7.State
Label8.Caption = Winsock8.State
Label11.Caption = Date
Label12.Caption = Time
End Sub

Private Sub Command3_Click()
Dim enviar As String
Adodc1.Recordset.MoveFirst
Do While (Adodc1.Recordset.EOF = False)
    If Adodc1.Recordset!idprog = Combo2 Then
        ano = Year(Now) - 1900
        mes = Month(Now)
        If mes < 10 Then mes = "0" + mes
        dia = Day(Now)
        If dia < 10 Then dia = "0" + dia
        hora = Hour(Now)
        If hora < 10 Then hora = "0" + hora
```

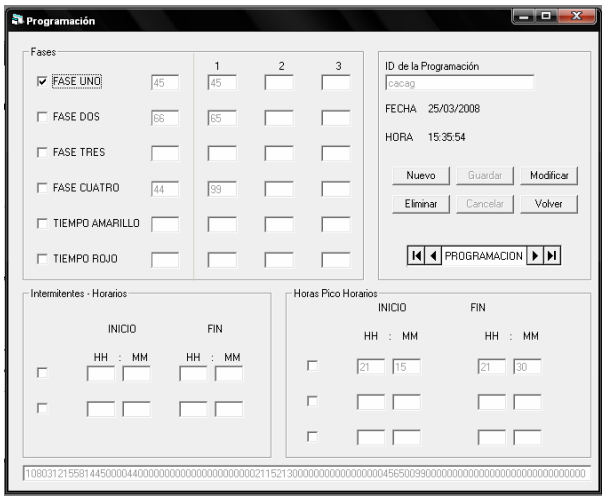
```
        minuto = Minute(Now)
        If minuto < 10 Then minuto = "0" + minuto
        segundo = Second(Now)
        If segundo < 10 Then segundo = "0" + segundo
        enviar = ano + mes + dia + hora + minuto + segundo + Adodc1.Recordset!trama
        If Combo1.Text = "UNO" Then Winsock1.SendData enviar
        If Combo1.Text = "DOS" Then Winsock2.SendData enviar
        If Combo1.Text = "TRES" Then Winsock3.SendData enviar
        If Combo1.Text = "CUATRO" Then Winsock4.SendData enviar
        If Combo1.Text = "CINCO" Then Winsock5.SendData enviar
        If Combo1.Text = "SEIS" Then Winsock6.SendData enviar
        If Combo1.Text = "SIETE" Then Winsock7.SendData enviar
        If Combo1.Text = "OCHO" Then Winsock8.SendData enviar
        Exit Sub
    End If
    Adodc1.Recordset.MoveNext
Loop

End Sub

Private Sub Form_Load()
estado
Form5.Height = 7125
Form5.Width = 12210
Form5.Left = (Screen.Width - Width) / 2
Form5.Top = (Screen.Height - Height) / 2 - 150
Adodc1.Recordset.MoveFirst
Do While (Adodc1.Recordset.EOF = False)
    Combo2.AddItem Adodc1.Recordset!idprog
    Adodc1.Recordset.MoveNext
Loop
End Sub
Private Sub Label1_Change()

If Winsock1.State = 2 Then
    MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"
End If
If Winsock1.State = 9 Then
    MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
    Winsock1.Close
End If
End Sub

Private Sub Label2_Change()
If Winsock2.State = 2 Then
    MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"
End If

If Winsock2.State = 9 Then
    MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
    Winsock2.Close
    'Winsock2.Listen
End If
End Sub

Private Sub Label3_Change()
If Winsock3.State = 2 Then
    MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"

End If
```

```
If Winsock3.State = 9 Then
   MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
   Winsock3.Close

End If
End Sub

Private Sub Label4_Change()
If Winsock4.State = 2 Then
   MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"

End If

If Winsock2.State = 9 Then
   MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
   Winsock4.Close
   'Winsock2.Listen
End If
End Sub

Private Sub Label5_Change()
If Winsock5.State = 2 Then
   MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"
End If

If Winsock5.State = 9 Then
   MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
   Winsock5.Close
   'Winsock2.Listen
End If
End Sub

Private Sub Label6_Change()
If Winsock6.State = 2 Then
   MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"
End If

If Winsock6.State = 9 Then
   MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
   Winsock6.Close
   'Winsock2.Listen
End If
End Sub

Private Sub Label7_Change()
If Winsock7.State = 2 Then
   MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"
End If

If Winsock7.State = 9 Then
   MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
   Winsock7.Close
   'Winsock2.Listen
End If
End Sub

Private Sub Label8_Change()
If Winsock8.State = 2 Then
   MsgBox ("Espere un momento para la conexión"), vbInformation, "Mensaje de Enlace"
End If
```

```
If Winsock8.State = 9 Then
    MsgBox ("Advertencia!...fallo de conexion pruebe otra vez"), vbInformation, "Mensaje de Enlace"
    Winsock8.Close
    'Winsock2.Listen
End If
End Sub
Private Sub Timer1_Timer()
estado
If Combo1.Text = "UNO" Or Combo1.Text = "DOS" Or Combo1.Text = "TRES" Or Combo1.Text =
"CUATRO" Or Combo1.Text = "CINCO" Or Combo1.Text = "SEIS" Or Combo1.Text = "SIETE" Or
Combo1.Text = "OCHO" Then
Else
  Command3.Enabled = False
End If
If Combo1.Text = "UNO" Then
  If Winsock1.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos1
End If
If Combo1.Text = "DOS" Then
  If Winsock2.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos2
End If
If Combo1.Text = "TRES" Then
  If Winsock3.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos3
End If
If Combo1.Text = "CUATRO" Then
  If Winsock4.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos4
End If
If Combo1.Text = "CINCO" Then
  If Winsock5.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos5
End If
If Combo1.Text = "SEIS" Then
  If Winsock6.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
```

```
  Label13.Caption = datos6
End If
If Combo1.Text = "SIETE" Then
  If Winsock7.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos7
End If
If Combo1.Text = "OCHO" Then
  If Winsock8.State = 7 Then
     Command3.Enabled = True
  Else
     Command3.Enabled = False
  End If
  Label13.Caption = datos8
End If
ver
End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
Winsock1.GetData datos1
End Sub

Private Sub Winsock2_DataArrival(ByVal bytesTotal As Long)
Winsock2.GetData datos2
End Sub

Private Sub Winsock3_DataArrival(ByVal bytesTotal As Long)
Winsock3.GetData datos3
End Sub

Private Sub Winsock4_DataArrival(ByVal bytesTotal As Long)
Winsock4.GetData datos4
End Sub

Private Sub Winsock5_DataArrival(ByVal bytesTotal As Long)
Winsock5.GetData datos5
End Sub

Private Sub Winsock6_DataArrival(ByVal bytesTotal As Long)
Winsock6.GetData datos6
End Sub

Private Sub Winsock7_DataArrival(ByVal bytesTotal As Long)
Winsock7.GetData datos7
End Sub

Private Sub Winsock8_DataArrival(ByVal bytesTotal As Long)
Winsock8.GetData datos8
End Sub
```

PANTALLA 4




```
Dim time1 As String
Dim time2 As String
Dim time3 As String
Dim time4 As String
Dim time5 As String
Dim time6 As String
Dim time7 As String
Dim time8 As String
Dim time9 As String
Dim time10 As String
Dim time11 As String
Dim time12 As String
Dim time13 As String
Dim time14 As String
Dim time15 As String
Dim time16 As String
Dim time17 As String
Dim time18 As String
Dim time19 As String
Dim time20 As String
Dim time21 As String
Dim time22 As String
Dim time23 As String
Dim time24 As String
Dim time25 As String
Dim time26 As String
Dim time27 As String
Dim time28 As String
Dim time29 As String
Dim time30 As String
Dim time31 As String
Dim time32 As String
Dim time33 As String
Dim time34 As String
Dim time35 As String
Dim time36 As String
Dim time37 As String
Dim time38 As String
```

```
Dim time39 As String
Dim time40 As String
Dim time41 As String
Dim time42 As String
Dim time43 As String
Dim time44 As String
Dim time45 As String
Dim time46 As String
Dim time47 As String
Dim time48 As String
Dim time49 As String
Dim time50 As String
Dim time51 As String
Dim time52 As String
Dim time53 As String
Dim time54 As String
Dim time55 As String
Dim time56 As String
Dim time57 As String
Dim time58 As String
Dim time59 As String
Dim time60 As String

Dim valor1 As Integer

Private Sub atext()
Text1.Enabled = True
Text2.Enabled = True
Text3.Enabled = True
Text4.Enabled = True
Text5.Enabled = True
Text6.Enabled = True
Text7.Enabled = True
Text8.Enabled = True
Text9.Enabled = True
Text10.Enabled = True
Text11.Enabled = True
Text12.Enabled = True
Text13.Enabled = True
Text14.Enabled = True
Text15.Enabled = True
Text16.Enabled = True
Text17.Enabled = True
Text18.Enabled = True
Text19.Enabled = True
Text20.Enabled = True
Text21.Enabled = True
Text22.Enabled = True
Text23.Enabled = True
Text24.Enabled = True
Text25.Enabled = True
Text26.Enabled = True
Text27.Enabled = True
Text28.Enabled = True
Text29.Enabled = True
Text30.Enabled = True
Text31.Enabled = True
Text32.Enabled = True
Text33.Enabled = True
Text34.Enabled = True
```

```
Text35.Enabled = True
Text36.Enabled = True
Text37.Enabled = True
Text38.Enabled = True
Text39.Enabled = True
Text40.Enabled = True
Text41.Enabled = True
Text42.Enabled = True
Text43.Enabled = True
Text44.Enabled = True
Text45.Enabled = True
Text46.Enabled = True
Check1.Enabled = True
Check2.Enabled = True
Check3.Enabled = True
Check4.Enabled = True
Check9.Enabled = True
Check10.Enabled = True
Check11.Enabled = True
Check12.Enabled = True
Check13.Enabled = True
Check14.Enabled = True
Check15.Enabled = True


End Sub
Private Sub dtext()
Text1.Enabled = False
Text2.Enabled = False
Text3.Enabled = False
Text4.Enabled = False
Text5.Enabled = False
Text6.Enabled = False
Text7.Enabled = False
Text8.Enabled = False
Text9.Enabled = False
Text10.Enabled = False
Text11.Enabled = False
Text12.Enabled = False
Text13.Enabled = False
Text14.Enabled = False
Text15.Enabled = False
Text16.Enabled = False
Text17.Enabled = False
Text18.Enabled = False
Text19.Enabled = False
Text20.Enabled = False
Text21.Enabled = False
Text22.Enabled = False
Text23.Enabled = False
Text24.Enabled = False
Text25.Enabled = False
Text26.Enabled = False
Text27.Enabled = False
Text28.Enabled = False
Text29.Enabled = False
Text30.Enabled = False
Text31.Enabled = False
Text32.Enabled = False
Text33.Enabled = False
```

```
Text34.Enabled = False
Text35.Enabled = False
Text36.Enabled = False
Text37.Enabled = False
Text38.Enabled = False
Text39.Enabled = False
Text40.Enabled = False
Text41.Enabled = False
Text42.Enabled = False
Text43.Enabled = False
Text44.Enabled = False
Text45.Enabled = False
Text46.Enabled = False
Check1.Enabled = False
Check2.Enabled = False
Check3.Enabled = False
Check4.Enabled = False
Check9.Enabled = False
Check10.Enabled = False
Check11.Enabled = False
Check12.Enabled = False
Check13.Enabled = False
Check14.Enabled = False
Check15.Enabled = False
End Sub
Private Sub abotones()
nuevo.Enabled = False
Modificar.Enabled = False
Eliminar.Enabled = False
Guardar.Enabled = True
Cancelar.Enabled = True
Adodc1.Visible = False
End Sub
Private Sub dbotones()
nuevo.Enabled = True
Modificar.Enabled = True
Eliminar.Enabled = True
Guardar.Enabled = False
Cancelar.Enabled = False
Adodc1.Visible = True
End Sub


Private Sub Cancelar_Click()
Adodc1.Recordset.CancelUpdate
dtext
dbotones
End Sub



Private Sub Eliminar_Click()
Dim r As Integer
   On Error GoTo RutinaDeError
   r = MsgBox("¿Desea borrar el registro?", vbInformation + vbYesNo, "Atención")
   If r <> vbYes Then Exit Sub
   Adodc1.Recordset.Delete        'borrar el registro actual
   Adodc1.Recordset.MoveNext      'situarse en el registro siguiente
   If Adodc1.Recordset.EOF Then
      Adodc1.Recordset.MoveLast
```

```
    End If
    Exit Sub
RutinaDeError:
    r = MsgBox(Error, vbOKOnly, "Se ha producido un error:")
    Adodc1.Recordset.CancelUpdate
End Sub

Private Sub Guardar_Click()
Dim enviar As String
Dim ano As String
Dim mes As String
Dim dia As String
Dim hora As String
Dim minuto As String
Dim segundo As String
time1 = Text1.Text
time2 = Text5.Text
time3 = Text9.Text
time4 = Text13.Text
time5 = Text17.Text
time6 = Text21.Text
time7 = Text25.Text
time8 = Text26.Text
time9 = Text27.Text
time10 = Text28.Text
time11 = Text29.Text
time12 = Text30.Text
time13 = Text31.Text
time14 = Text32.Text
time15 = Text33.Text
time16 = Text34.Text
time17 = Text35.Text
time18 = Text36.Text
time19 = Text37.Text
time20 = Text38.Text
time21 = Text39.Text
time22 = Text40.Text
time23 = Text41.Text
time24 = Text42.Text
time25 = Text43.Text
time26 = Text44.Text
time27 = Text2.Text
time28 = Text6.Text
time29 = Text10.Text
time30 = Text14.Text
time31 = Text18.Text
time32 = Text22.Text
time33 = Text3.Text
time34 = Text7.Text
time35 = Text11.Text
time36 = Text15.Text
time37 = Text19.Text
time38 = Text23.Text
time39 = Text4.Text
time40 = Text8.Text
time41 = Text12.Text
time42 = Text16.Text
time43 = Text20.Text
time44 = Text24.Text
```

```
If Check1.Value = 0 Or Text1.Text = "" Then
   time1 = "00"
Else
   If Val(Text1.Text) >= 0 And Val(Text1.Text) < 10 Then time1 = "0" + Text1.Text
End If
If Check2.Value = 0 Or Text5.Text = "" Then
   time2 = "00"
Else
   If Val(Text5.Text) >= 0 And Val(Text5.Text) < 10 Then time2 = "0" + Text5.Text
End If
If Check3.Value = 0 Or Text9.Text = "" Then
   time3 = "00"
Else
   If Val(Text9.Text) >= 0 And Val(Text9.Text) < 10 Then time3 = "0" + Text9.Text
End If
If Check4.Value = 0 Or Text13.Text = "" Then
   time4 = "00"
Else
   If Val(Text13.Text) >= 0 And Val(Text13.Text) < 10 Then time4 = "0" + Text13.Text
End If
If Check9.Value = 0 Or Text17.Text = "" Then
   time5 = "00"
Else
   If Val(Text17.Text) >= 0 And Val(Text17.Text) < 10 Then time5 = "0" + Text17.Text
End If

If Check10.Value = 0 Or Text21.Text = "" Then
   time6 = "0000"
Else
   If Val(Text21.Text) >= 0 And Val(Text21.Text) < 10 Then time6 = "0" + "0" + "0" + Text21.Text
   If Val(Text21.Text) >= 10 And Val(Text21.Text) < 100 Then time6 = "0" + "0" + Text21.Text
   If Val(Text21.Text) >= 100 And Val(Text21.Text) < 1000 Then time6 = "0" + Text21.Text
End If
If Check11.Value = 0 Or Text25.Text = "" Or Text26.Text = "" Or Text27.Text = "" Or Text28.Text = ""
Then
   time7 = "00"
   time8 = "00"
   time9 = "00"
   time10 = "00"
Else
   If Val(Text25.Text) >= 0 And Val(Text25.Text) < 10 Then time7 = "0" + Text25.Text
   If Val(Text26.Text) >= 0 And Val(Text26.Text) < 10 Then time8 = "0" + Text26.Text
   If Val(Text27.Text) >= 0 And Val(Text27.Text) < 10 Then time9 = "0" + Text27.Text
   If Val(Text28.Text) >= 0 And Val(Text28.Text) < 10 Then time10 = "0" + Text28.Text
End If
If Check12.Value = 0 Or Text29.Text = "" Or Text30.Text = "" Or Text31.Text = "" Or Text32.Text = ""
Then
   time11 = "00"
   time12 = "00"
   time13 = "00"
   time14 = "00"
Else
   If Val(Text29.Text) >= 0 And Val(Text29.Text) < 10 Then time11 = "0" + Text29.Text
   If Val(Text30.Text) >= 0 And Val(Text30.Text) < 10 Then time12 = "0" + Text30.Text
   If Val(Text31.Text) >= 0 And Val(Text31.Text) < 10 Then time13 = "0" + Text31.Text
   If Val(Text32.Text) >= 0 And Val(Text32.Text) < 10 Then time14 = "0" + Text32.Text
End If
If Check13.Value = 0 Or Text33.Text = "" Or Text34.Text = "" Or Text35.Text = "" Or Text36.Text = ""
Then
   time15 = "00"
```

```
    time16 = "00"
    time17 = "00"
    time18 = "00"
    time27 = "00"
    time28 = "00"
    time29 = "00"
    time30 = "00"
    time31 = "00"
    time32 = "00"

Else
    If Val(Text33.Text) >= 0 And Val(Text33.Text) < 10 Then time15 = "0" + Text33.Text
    If Val(Text34.Text) >= 0 And Val(Text34.Text) < 10 Then time16 = "0" + Text34.Text
    If Val(Text35.Text) >= 0 And Val(Text35.Text) < 10 Then time17 = "0" + Text35.Text
    If Val(Text36.Text) >= 0 And Val(Text36.Text) < 10 Then time18 = "0" + Text36.Text
End If
If Check14.Value = 0 Or Text37.Text = "" Or Text38.Text = "" Or Text39.Text = "" Or Text40.Text = ""
Then
    time19 = "00"
    time20 = "00"
    time21 = "00"
    time22 = "00"
    time33 = "00"
    time34 = "00"
    time35 = "00"
    time36 = "00"
    time37 = "00"
    time38 = "00"
Else
    If Val(Text37.Text) >= 0 And Val(Text37.Text) < 10 Then time19 = "0" + Text37.Text
    If Val(Text38.Text) >= 0 And Val(Text38.Text) < 10 Then time20 = "0" + Text38.Text
    If Val(Text39.Text) >= 0 And Val(Text39.Text) < 10 Then time21 = "0" + Text39.Text
    If Val(Text40.Text) >= 0 And Val(Text40.Text) < 10 Then time22 = "0" + Text40.Text
End If
If Check15.Value = 0 Or Text28.Text = "" Or Text29.Text = "" Or Text30.Text = "" Or Text31.Text = ""
Then
    time23 = "00"
    time24 = "00"
    time25 = "00"
    time26 = "00"
    time39 = "00"
    time40 = "00"
    time41 = "00"
    time42 = "00"
    time43 = "00"
    time44 = "00"
Else
    If Val(Text41.Text) >= 0 And Val(Text41.Text) < 10 Then time23 = "0" + Text41.Text
    If Val(Text42.Text) >= 0 And Val(Text42.Text) < 10 Then time24 = "0" + Text42.Text
    If Val(Text43.Text) >= 0 And Val(Text43.Text) < 10 Then time25 = "0" + Text43.Text
    If Val(Text44.Text) >= 0 And Val(Text44.Text) < 10 Then time26 = "0" + Text44.Text
End If
If Text1.Text = "" Then
    time1 = "00"
Else
    If Val(Text1.Text) >= 0 And Val(Text1.Text) < 10 Then time1 = "0" + Text1.Text
End If
If Text5.Text = "" Then
    time2 = "00"
Else
```

```
   If Val(Text5.Text) >= 0 And Val(Text5.Text) < 10 Then time2 = "0" + Text5.Text
End If
If Text9.Text = "" Then
   time3 = "00"
Else
   If Val(Text9.Text) >= 0 And Val(Text9.Text) < 10 Then time3 = "0" + Text9.Text
End If
If Text13.Text = "" Then
   time4 = "00"
Else
   If Val(Text13.Text) >= 0 And Val(Text13.Text) < 10 Then time4 = "0" + Text13.Text
End If
If Text17.Text = "" Then
   time5 = "00"
Else
   If Val(Text17.Text) >= 0 And Val(Text17.Text) < 10 Then time5 = "0" + Text17.Text
End If
If Text21.Text = "" Then
   time6 = "0000"
Else
   If Val(Text21.Text) >= 0 And Val(Text21.Text) < 10 Then time6 = "0" + "0" + "0" + Text21.Text
   If Val(Text21.Text) >= 10 And Val(Text21.Text) < 100 Then time6 = "0" + "0" + Text21.Text
   If Val(Text21.Text) >= 100 And Val(Text21.Text) < 1000 Then time6 = "0" + Text21.Text

End If
If Text2.Text = "" Then
   time27 = "00"
Else
   If Val(Text2.Text) >= 0 And Val(Text2.Text) < 10 Then time27 = "0" + Text2.Text
End If
If Text6.Text = "" Then
   time28 = "00"
Else
   If Val(Text6.Text) >= 0 And Val(Text6.Text) < 10 Then time28 = "0" + Text6.Text
End If
If Text10.Text = "" Then
   time29 = "00"
Else
   If Val(Text10.Text) >= 0 And Val(Text10.Text) < 10 Then time29 = "0" + Text10.Text
End If
If Text14.Text = "" Then
   time30 = "00"
Else
   If Val(Text14.Text) >= 0 And Val(Text14.Text) < 10 Then time30 = "0" + Text14.Text
End If
If Text18.Text = "" Then
   time31 = "00"
Else
   If Val(Text18.Text) >= 0 And Val(Text18.Text) < 10 Then time31 = "0" + Text18.Text
End If
If Text22.Text = "" Then
   time32 = "0000"
Else
   If Val(Text22.Text) >= 0 And Val(Text22.Text) < 10 Then time32 = "0" + "0" + "0" + Text22.Text
   If Val(Text22.Text) >= 10 And Val(Text22.Text) < 100 Then time32 = "0" + "0" + Text22.Text
   If Val(Text22.Text) >= 100 And Val(Text22.Text) < 1000 Then time32 = "0" + Text22.Text

End If
If Text3.Text = "" Then
   time33 = "00"
```

```
Else
   If Val(Text3.Text) >= 0 And Val(Text3.Text) < 10 Then time33 = "0" + Text3.Text
End If
If Text7.Text = "" Then
   time34 = "00"
Else
   If Val(Text7.Text) >= 0 And Val(Text7.Text) < 10 Then time34 = "0" + Text7.Text
End If
If Text11.Text = "" Then
   time35 = "00"
Else
   If Val(Text11.Text) >= 0 And Val(Text11.Text) < 10 Then time35 = "0" + Text11.Text
End If
If Text15.Text = "" Then
   time36 = "00"
Else
   If Val(Text15.Text) >= 0 And Val(Text15.Text) < 10 Then time36 = "0" + Text15.Text
End If
If Text19.Text = "" Then
   time37 = "00"
Else
   If Val(Text19.Text) >= 0 And Val(Text19.Text) < 10 Then time37 = "0" + Text19.Text
End If
If Text23.Text = "" Then
   time38 = "0000"
Else
   If Val(Text23.Text) >= 0 And Val(Text23.Text) < 10 Then time38 = "0" + "0" + "0" + Text23.Text
   If Val(Text23.Text) >= 10 And Val(Text23.Text) < 100 Then time38 = "0" + "0" + Text23.Text
   If Val(Text23.Text) >= 100 And Val(Text23.Text) < 1000 Then time38 = "0" + Text23.Text
End If
If Text4.Text = "" Then
   time39 = "00"
Else
   If Val(Text4.Text) >= 0 And Val(Text4.Text) < 10 Then time39 = "0" + Text4.Text
End If
If Text8.Text = "" Then
   time40 = "00"
Else
   If Val(Text8.Text) >= 0 And Val(Text8.Text) < 10 Then time40 = "0" + Text8.Text
End If
If Text12.Text = "" Then
   time41 = "00"
Else
   If Val(Text12.Text) >= 0 And Val(Text12.Text) < 10 Then time41 = "0" + Text12.Text
End If
If Text16.Text = "" Then
   time42 = "00"
Else
   If Val(Text16.Text) >= 0 And Val(Text16.Text) < 10 Then time42 = "0" + Text16.Text
End If
If Text20.Text = "" Then
   time43 = "00"
Else
   If Val(Text20.Text) >= 0 And Val(Text20.Text) < 10 Then time43 = "0" + Text20.Text
End If
If Text24.Text = "" Then
   time44 = "0000"
Else
   If Val(Text24.Text) >= 0 And Val(Text24.Text) < 10 Then time44 = "0" + "0" + "0" + Text24.Text
   If Val(Text24.Text) >= 10 And Val(Text24.Text) < 100 Then time44 = "0" + "0" + Text24.Text
```

```
   If Val(Text24.Text) >= 100 And Val(Text24.Text) < 1000 Then time44 = "0" + Text24.Text
End If

ano = Year(Now) - 1900
mes = Month(Now)
If mes < 10 Then mes = "0" + mes
dia = Day(Now)
If dia < 10 Then dia = "0" + dia
hora = Hour(Now)
If hora < 10 Then hora = "0" + hora
minuto = Minute(Now)
If minuto < 10 Then minuto = "0" + minuto
segundo = Second(Now)
If segundo < 10 Then segundo = "0" + segundo
enviar = time1 + time2 + time3 + time4 + time5 + time6 + time7 + time8 + time9 + time10 + time11 +
time12 + time13 + time14 + time15 + time16 + time17 + time18 + time19 + time20 + time21 + time22 +
time23 + time24 + time25 + time26 + time27 + time28 + time29 + time30 + time31 + time32 + time33 +
time34 + time35 + time36 + time37 + time38 + time39 + time40 + time41 + time42 + time43 + time44
Text46.Text = enviar
Adodc1.Recordset.Update
dtext
dbotones
End Sub

Private Sub Command5_Click()
Unload Me
Form5.Show
End Sub

Private Sub Form_Load()
Label14.Caption = Date
Label15.Caption = Time
Form4.Height = 8235
Form4.Width = 10080
Form4.Left = (Screen.Width - Width) / 2
Form4.Top = (Screen.Height - Height) / 2 - 150
End Sub

Public Sub numeros(valor As Integer)
If valor < 48 Or valor > 57 Then
   valor1 = valor
   If valor = 8 Or valor = 13 Then
      Exit Sub
   End If
   MsgBox ("solo numeros")
   valor = 0
End If
valor1 = valor
End Sub

Private Sub Modificar_Click()
atext
abotones
Text1.SetFocus
End Sub

Private Sub nuevo_Click()
atext
abotones
Adodc1.Recordset.AddNew
```

```
Text1.SetFocus
End Sub

Private Sub Text1_Change()
If Val(Text1.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text1.Text = ""
End If
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text2_Change()
If Val(Text2.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text2.Text = ""
End If
End Sub

Private Sub Text2_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub


Private Sub Text3_Change()
If Val(Text3.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text3.Text = ""
End If
End Sub

Private Sub Text3_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text4_Change()
If Val(Text4.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text4.Text = ""
End If
End Sub

Private Sub Text4_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text5_Change()
If Val(Text5.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text5.Text = ""
End If
End Sub

Private Sub Text5_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
```

```
KeyAscii = valor1
End Sub

Private Sub Text6_Change()
If Val(Text6.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text6.Text = ""
End If
End Sub

Private Sub Text6_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text7_Change()
If Val(Text7.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text7.Text = ""
End If
End Sub

Private Sub Text7_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text8_Change()
If Val(Text8.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text8.Text = ""
End If
End Sub

Private Sub Text8_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text9_Change()
If Val(Text9.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text9.Text = ""
End If
End Sub

Private Sub Text9_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text10_Change()
If Val(Text10.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text10.Text = ""
End If
End Sub

Private Sub Text10_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
```

```
End Sub

Private Sub Text11_Change()
If Val(Text11.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text11.Text = ""
End If
End Sub

Private Sub Text11_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text12_Change()
If Val(Text12.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text12.Text = ""
End If
End Sub

Private Sub Text12_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text13_Change()
If Val(Text13.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text13.Text = ""
End If
End Sub

Private Sub Text13_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text14_Change()
If Val(Text14.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text14.Text = ""
End If
End Sub

Private Sub Text14_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text15_Change()
If Val(Text15.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text15.Text = ""
End If
End Sub

Private Sub Text15_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
```

```
Private Sub Text16_Change()
If Val(Text16.Text) > 99 Then
   MsgBox ("maximo 99 segundos")
   Text16.Text = ""
End If
End Sub

Private Sub Text16_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text17_Change()
If Val(Text17.Text) > 10 Then
   MsgBox ("maximo 10 segundos")
   Text17.Text = ""
End If
End Sub

Private Sub Text17_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text18_Change()
If Val(Text18.Text) > 10 Then
   MsgBox ("maximo 10 segundos")
   Text18.Text = ""
End If
End Sub

Private Sub Text18_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text19_Change()
If Val(Text19.Text) > 10 Then
   MsgBox ("maximo 10 segundos")
   Text19.Text = ""
End If
End Sub

Private Sub Text19_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text20_Change()
If Val(Text20.Text) > 10 Then
   MsgBox ("maximo 10 segundos")
   Text20.Text = ""
End If
End Sub

Private Sub Text20_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text21_Change()
If Val(Text21.Text) > 1000 Then
   MsgBox ("maximo 1000 milisegundos")
```

```
   Text21.Text = ""
End If
End Sub

Private Sub Text21_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text22_Change()
If Val(Text22.Text) > 1000 Then
   MsgBox ("maximo 1000 milisegundos")
   Text22.Text = ""
End If
End Sub

Private Sub Text22_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text23_Change()
If Val(Text23.Text) > 1000 Then
   MsgBox ("maximo 1000 milisegundos")
   Text23.Text = ""
End If
End Sub

Private Sub Text23_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text24_Change()
If Val(Text24.Text) > 1000 Then
   MsgBox ("maximo 1000 milisegundos")
   Text24.Text = ""
End If
End Sub

Private Sub Text24_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub
Private Sub Text25_Change()
If Val(Text25.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text25.Text = ""
End If
End Sub

Private Sub Text25_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text26_Change()
If Val(Text26.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text26.Text = ""
```

```
End If
End Sub

Private Sub Text26_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text27_Change()
If Val(Text27.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text27.Text = ""
End If
End Sub

Private Sub Text27_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text28_Change()
If Val(Text28.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text28.Text = ""
End If
End Sub

Private Sub Text28_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text29_Change()
If Val(Text29.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text29.Text = ""
End If
End Sub

Private Sub Text29_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text30_Change()
If Val(Text30.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text30.Text = ""
End If
End Sub

Private Sub Text30_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text31_Change()
If Val(Text31.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text31.Text = ""
```

```
End If
End Sub

Private Sub Text31_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text32_Change()
If Val(Text32.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text32.Text = ""
End If
End Sub

Private Sub Text32_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text33_Change()
If Val(Text33.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text33.Text = ""
End If
End Sub

Private Sub Text33_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text34_Change()
If Val(Text34.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text34.Text = ""
End If
End Sub

Private Sub Text34_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text35_Change()
If Val(Text35.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text35.Text = ""
End If
End Sub

Private Sub Text35_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text36_Change()
If Val(Text36.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text36.Text = ""
```

```
End If
End Sub

Private Sub Text36_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text37_Change()
If Val(Text37.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text37.Text = ""
End If
End Sub

Private Sub Text37_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text38_Change()
If Val(Text38.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text38.Text = ""
End If
End Sub

Private Sub Text38_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text39_Change()
If Val(Text39.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text39.Text = ""
End If
End Sub

Private Sub Text39_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text40_Change()
If Val(Text40.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text40.Text = ""
End If
End Sub

Private Sub Text40_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text41_Change()
If Val(Text41.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text41.Text = ""
```

```
End If
End Sub

Private Sub Text41_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text42_Change()
If Val(Text42.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text42.Text = ""
End If
End Sub

Private Sub Text42_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text43_Change()
If Val(Text43.Text) > 23 Then
   MsgBox ("maximo 23 horas")
   Text43.Text = ""
End If
End Sub

Private Sub Text43_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub

Private Sub Text44_Change()
If Val(Text44.Text) > 59 Then
   MsgBox ("maximo 59 minutos")
   Text44.Text = ""
End If
End Sub

Private Sub Text44_KeyPress(KeyAscii As Integer)
numeros (KeyAscii)
KeyAscii = valor1
End Sub


Private Sub Check11_Click()
'If Check11.Value = 1 Then
'    Text25.Enabled = True
'    Text26.Enabled = True
'    Text27.Enabled = True
'    Text28.Enabled = True
'Else
'    Text25.Enabled = False
'    Text26.Enabled = False
'    Text27.Enabled = False
'    Text28.Enabled = False
'End If
End Sub

Private Sub Check12_Click()
```

```
'If Check12.Value = 1 Then
'    Text29.Enabled = True
'    Text30.Enabled = True
'    Text31.Enabled = True
'    Text32.Enabled = True
'Else
'    Text29.Enabled = False
'    Text30.Enabled = False
'    Text31.Enabled = False
'    Text32.Enabled = False
'End If
End Sub

Private Sub Check13_Click()
'If Check13.Value = 1 Then
'    Text2.Enabled = True
'    Text6.Enabled = True
'    Text10.Enabled = True
'    Text14.Enabled = True
'    Text18.Enabled = True
'    Text22.Enabled = True
'Else
'    Text2.Enabled = False
'    Text6.Enabled = False
'    Text10.Enabled = False
'    Text14.Enabled = False
'    Text18.Enabled = False
'    Text22.Enabled = False
'End If
End Sub

Private Sub Check14_Click()
'If Check14.Value = 1 Then
'   Text3.Enabled = True
'    Text7.Enabled = True
'    Text11.Enabled = True
'    Text15.Enabled = True
'    Text19.Enabled = True
'    Text23.Enabled = True
'Else
'    Text3.Enabled = False
'    Text7.Enabled = False
'    Text11.Enabled = False
'    Text15.Enabled = False
'    Text19.Enabled = False
'    Text23.Enabled = False
'End If
End Sub

'Private Sub Timer1_Timer()
'Label14.Caption = Date
'Label15.Caption = Time
'End Sub
```

PANTALLA 5


```
Private Sub atext()
Text1.Enabled = True
Text2.Enabled = True
Text3.Enabled = True
Text4.Enabled = True
Combo1.Enabled = True
End Sub
Private Sub dtext()
Text1.Enabled = False
Text2.Enabled = False
Text3.Enabled = False
Text4.Enabled = False
Combo1.Enabled = False
End Sub
Private Sub abotones()
nuevo.Enabled = False
Modificar.Enabled = False
Eliminar.Enabled = False
Guardar.Enabled = True
Cancelar.Enabled = True
Adodc1.Visible = False
End Sub
Private Sub dbotones()
nuevo.Enabled = True
Modificar.Enabled = True
Eliminar.Enabled = True
Guardar.Enabled = False
Cancelar.Enabled = False
Adodc1.Visible = True
End Sub

Private Sub Cerrar_Click()
Unload Me
End Sub

Private Sub Cancelar_Click()
Adodc1.Recordset.CancelUpdate
dtext
dbotones
End Sub

Private Sub Eliminar_Click()
Dim r As Integer
```

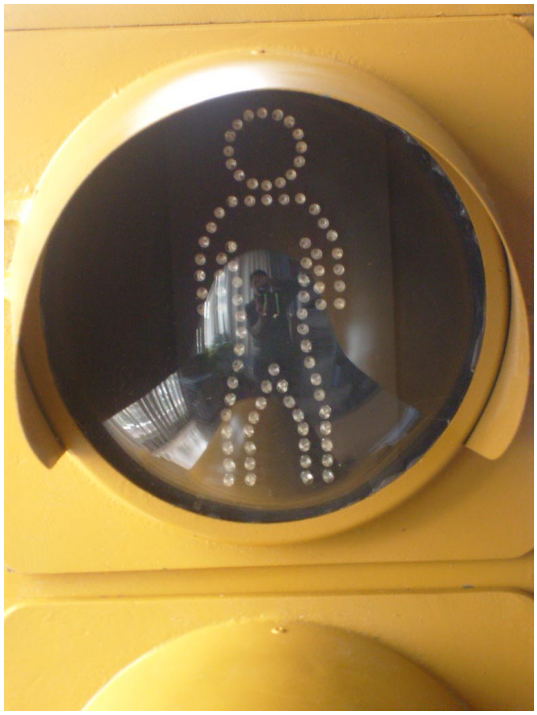
```
    On Error GoTo RutinaDeError
    r = MsgBox("¿Desea borrar el registro?", vbInformation + vbYesNo, "Atención")
    If r <> vbYes Then Exit Sub
    Adodc1.Recordset.Delete         'borrar el registro actual
    Adodc1.Recordset.MoveNext       'situarse en el registro siguiente
    If Adodc1.Recordset.EOF Then
       Adodc1.Recordset.MoveLast
    End If
    Exit Sub
RutinaDeError:
    r = MsgBox(Error, vbOKOnly, "Se ha producido un error:")
    Adodc1.Recordset.CancelUpdate
End Sub
Private Sub Form_Load()
Form3.Height = 3750
Form3.Width = 5670
Form3.Left = (Screen.Width - Width) / 2
Form3.Top = (Screen.Height - Height) / 2 - 150
End Sub

Private Sub Guardar_Click()
If Text1 = "" Or Text2 = "" Or Text3 = "" Or Text4 = "" Then
  MsgBox ("Advertencia!...Debe llenar todo los campos..."), vbInformation, "Mensaje de Seguridad "
Else
Adodc1.Recordset.Update
dtext
dbotones
End If
End Sub

Private Sub Modificar_Click()
atext
abotones
Text1.SetFocus
End Sub

Private Sub nuevo_Click()
atext
abotones
Adodc1.Recordset.AddNew
Text1.SetFocus
End Sub

Private Sub Text1_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
  Text2.SetFocus
End If
End Sub
Private Sub Text2_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
  Text3.SetFocus
End If
End Sub
Private Sub Text3_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then
  Text4.SetFocus
End If
End Sub
```

ANEXO F

FOTOGRAFÍAS

1.- Luminaria Peatonal – Matrices de Diodos Led

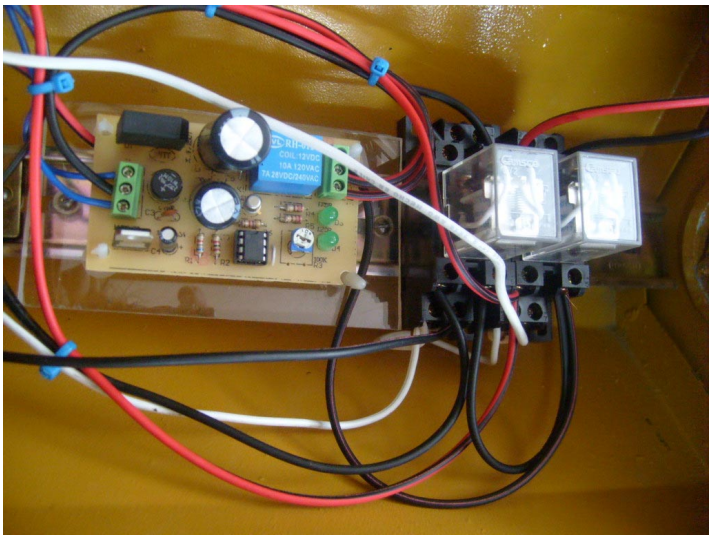
2.- Luminaria Peatonal Roja



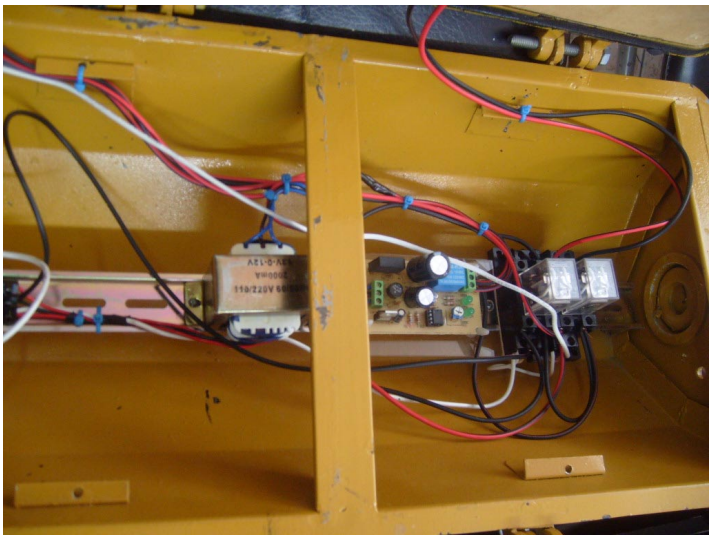
3.- Luminaria Peatonal Verde - Animación

4.- Circuitos Electrónicos del Subsistema de Semaforización Peatonal

4.1



4.2

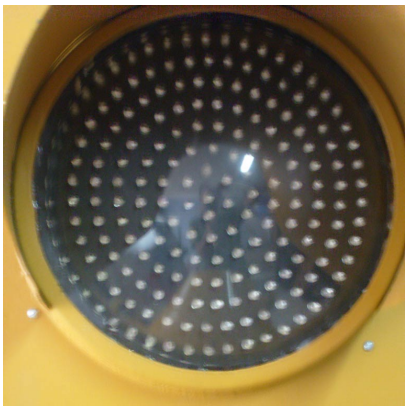


4.3

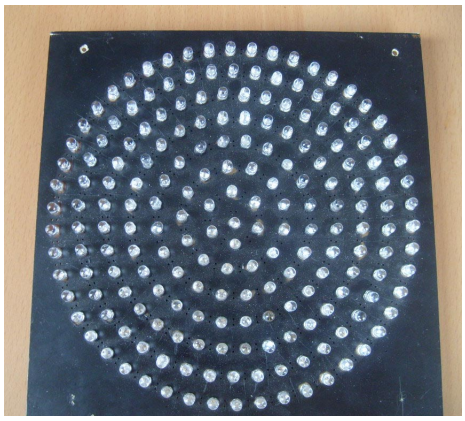
5.- Luminaria Vehicular – Matrices de Diodos Led

6.- Luminaria de Semaforización Vehicular

6.1

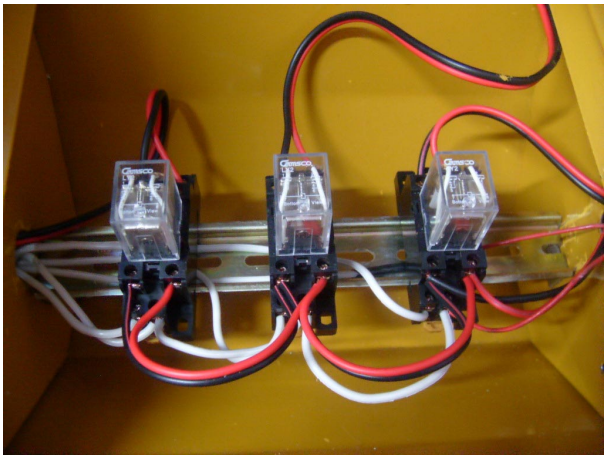


6.2

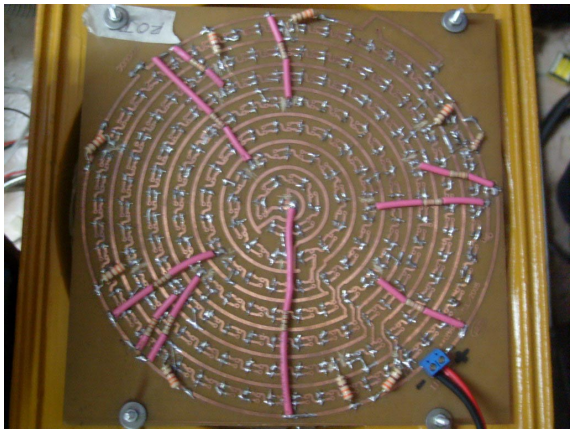


7.- Circuitos Electrónicos del Subsistema de Semaforización Vehicular

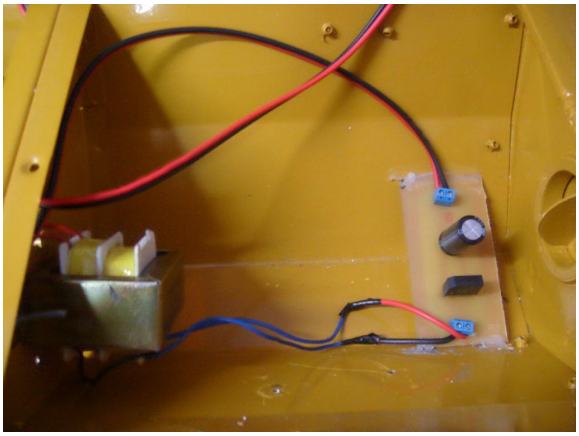
7.1



7.2

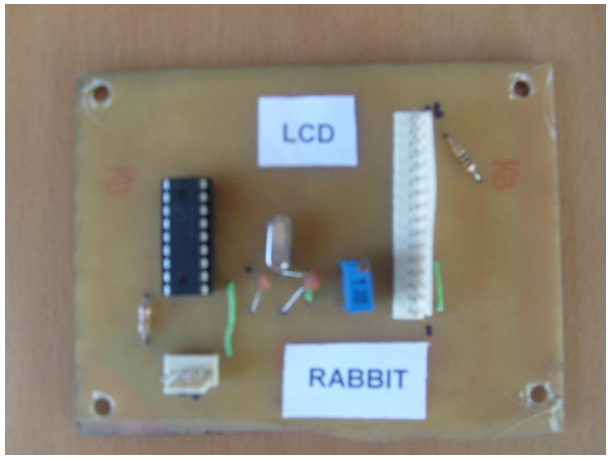


7.3

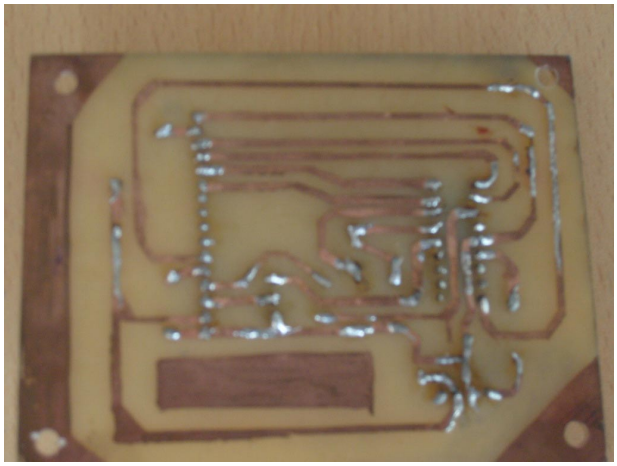


8.- Circuito de manejo de LCD

8.1 Anverso



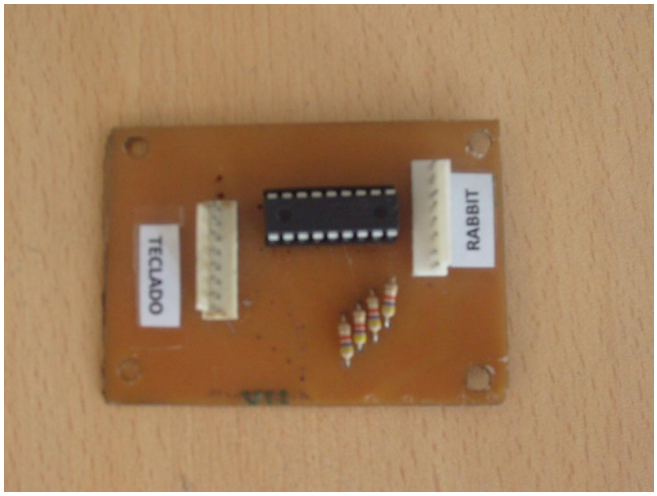
8.1 Reverso



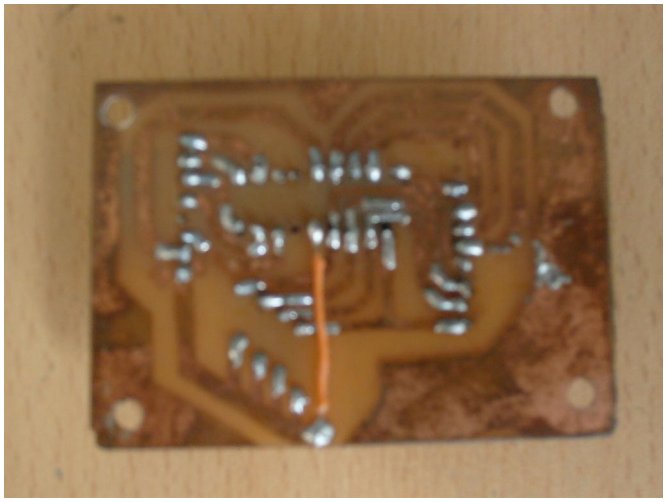
9.- LCD

10.- Circuito de manejo de teclado

10.1 Anverso

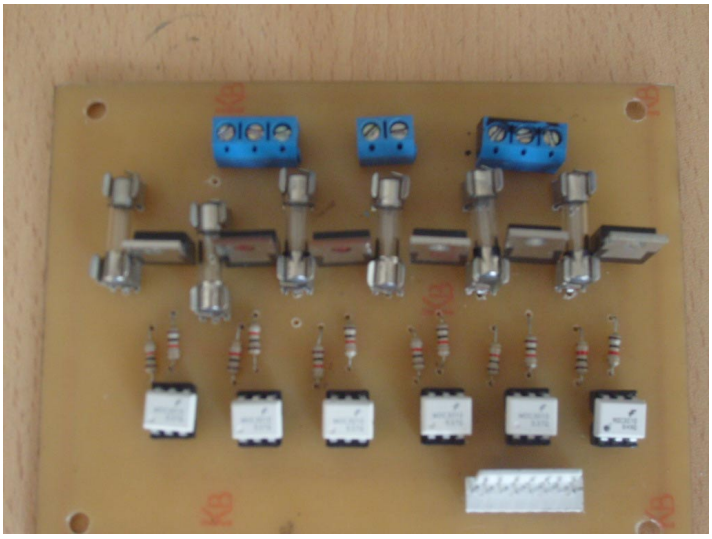


10.2 Reverso

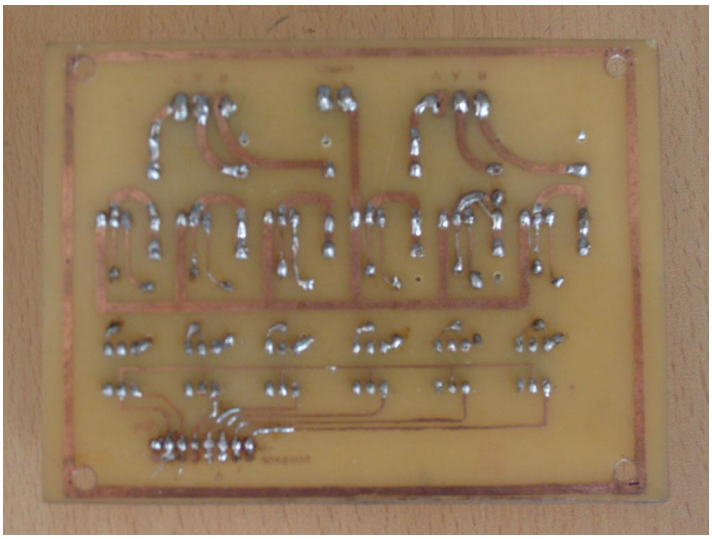


11.- Circuito de Disparo

11.1 Anverso

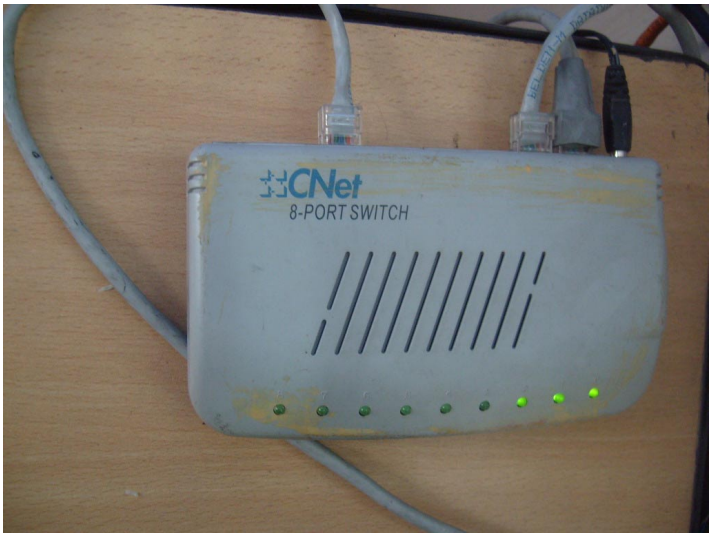


11.2 Reverso

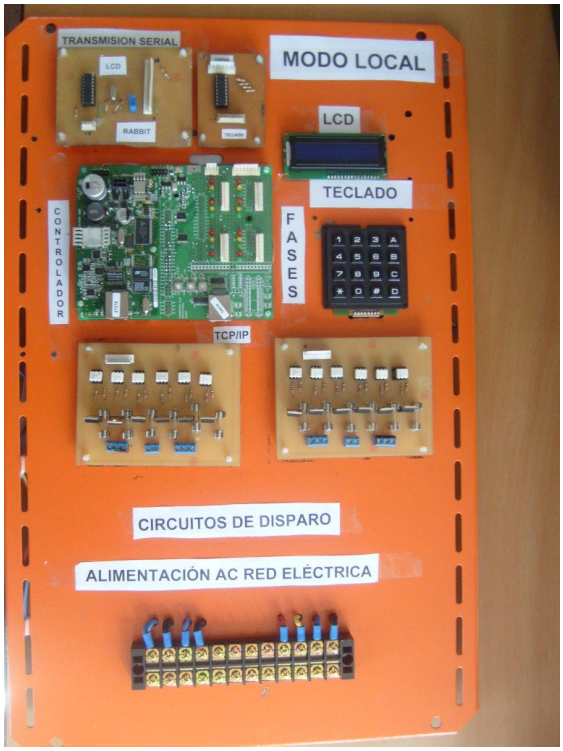


12.- Armario de Control

13.- Switch de red

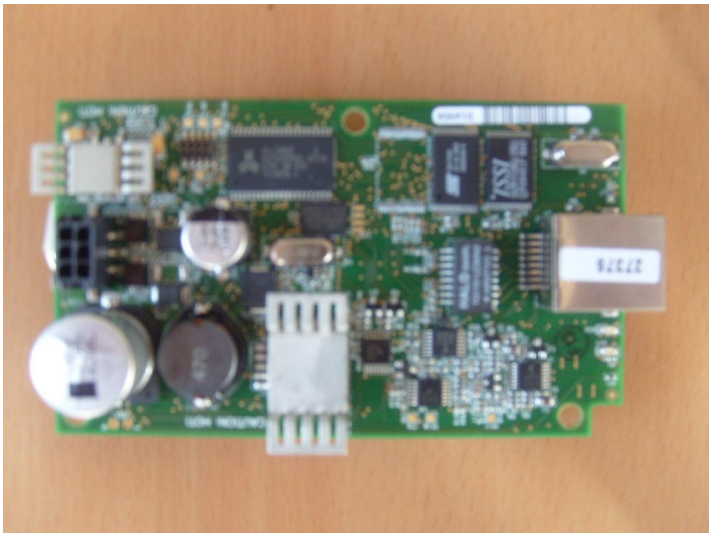


14.- Tablero de Control

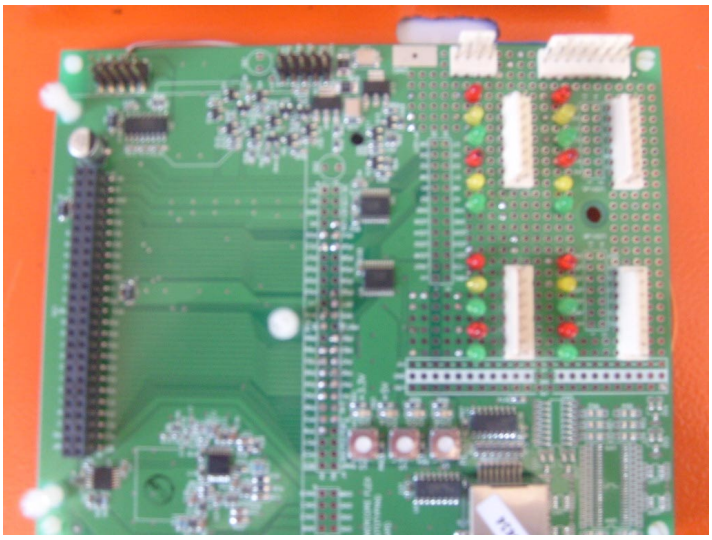


15.- Controlador

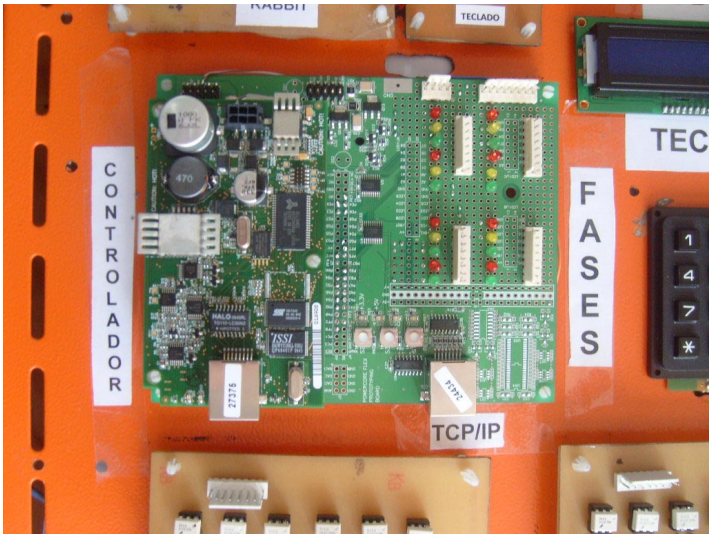
15.1 Módulo PowerCore Flex 3800



15.2 Disposición de Componentes en Placa



15.3 Disposición de La Tarjeta Rabbit en la Unidad de Control Principal



GLOSARIO

ANSI: (American National Standards Institute) Instituto de Estandarización Nacional Americano.

CRC: Control de redundancia cíclica en el envío de la trama de datos de Ethernet.

CSMA/CD: (Carrier Sense Multiple Access with Collision Detection) Acceso Múltiple con Escucha de Portadora y Detección de Colisiones, es una técnica usada en redes Ethernet para mejorar sus prestaciones.

DYNAMIC C: Compilador, editor, cargador y depurador específico de los microprocesadores Rabbit.

EIA: Estándar Europeo de Dispositivos Electrónicos**.**

ETHERNET: Es el nombre de una tecnología de redes de computadoras de área local (LANs) basada en tramas de datos. Ethernet define las características de cableado y señalización de nivel físico y los formatos de trama del nivel de enlace de datos del modelo OSI.

FCS: (Frame Check Sequence) Secuencia de Verificación de Trama de Ethernet.

HDLC: (High-Level Data Link Control) Es un protocolo de comunicaciones de datos punto a punto. Proporciona recuperación de errores en caso de pérdida de paquetes de datos o fallos de secuencia.

HYPER TERMINAL: Software de comunicaciones utilizado para conectarse a otros equipos a través de módem, serie RS-232 conexiones, o telnet

IEEE: (Institute of Electrical and Electronics) **A**sociación técnico-profesional mundial dedicada a la estandarización eléctrico.

INEN: Instituto Ecuatoriano de Normalización

LCD: (Liquid Crystal Display) Pantalla delgada y plana formada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora.

LED: (Light-Emitting Diode), diodo emisor de luz es un dispositivo semiconductor que emite luz cuasi-monocromática, es decir, con un espectro muy angosto, cuando se polariza de forma directa y es atravesado por una corriente eléctrica.

LLC: (Logical Link Control) Control de Enlace Lógico, está en la parte superior de la subcapa de enlace dentro del modelo OSI

MAC: (Media Access Control) Dirección de control de acceso al medio. Es un identificador hexadecimal de 48 bits que corresponde de forma única a una tarjeta o interfaz de red.

OSI: (Open System Interconnection) modelo de referencia de Interconexión de Sistemas Abiertos. Sirve de marco de referencia para la definición de arquitecturas de interconexión de sistemas de comunicaciones.

PLC: (Programmable Logic Controller) Controlador Lógico Programable.

RED: Es un conjunto de nodos y enlaces que proporcionan conexiones entre dos o más puntos definidos para facilitar la telecomunicación entre ellos.

RJ-45: Interfaz física comúnmente usada para conectar redes de cableado estructurado, (categorías 4, 5, 5e y 6). RJ es un acrónimo inglés de Registered Jack que a su vez es parte del Código Federal de Regulaciones de Estados Unidos.

RS-232: Estándar para la conexión serial de señales de datos binarias entre un DTE (Equipo terminal de datos) y un DCE (Equipo de terminación del circuito de datos)

RS-422: Estándar aprobado por la EIA para conectar dispositivos en forma serial.

SDLC: (Syncronous Data Link Controller) Controlador de enlace de datos síncrono para la capa 2 del modelo OSI de comunicaciones.

SOCKET: Un *socket* queda definido por una dirección IP, un protocolo y un número de puerto

SOF: (Start Of Frame) Conjunto de btis, indica el Inicio de Trama.

SWITCH: es un dispositivo electrónico de interconexión de redes de ordenadores que opera en la capa 2 (nivel de enlace de datos) del modelo OSI Interconecta dos o más segmentos de red, funcionando de manera similar a los puentes, pasando datos de un segmento a otro, de acuerdo con la dirección MAC de destino de los datagramas en la red.

TCP/IP: Protocolo de Control de Transmisión / Protocolo Internet de capa de Transporte que asegura la entrega satisfactoria de extremo a extremo de paquetes de datos sin error, como lo define el IETF.

USB: (Universal Serial Bus) Es un puerto que sirve para conectar periféricos a un computador.

UTP: Definido en el estándar EIA/TIA 568, soporta velocidades de transmisión de 10 Mbps en Ethernet 10Base-T, y 4 Mbps. Este cable tiene cuatro pares y su impedancia es de 100 Ω.

ÍNDICE DE FIGURAS

CAPITULO I

CAPITULO II

CAPITULO III

CAPITULO IV

CAPITULO V

CAPITULO VII

ÍNDICE DE TABLAS

FECHA DE ENTREGA:

_____

ING. VICTOR PROAÑO

COORDINADOR DE LA CARRERA DE
INGENIERÍA ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

_____ _____

JORGE NAPOLEÓN
ALMEIDA GARZÓN

AUTOR

SANTIAGO FERNANDO
MAFLA LEGARDA

AUTOR