# UNIVERSIDAD NACIONAL DE INGENIERÍA

## FACULTAD DE INGENIERÍA MECÁNICA

# DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE CONTROL Y SUPERVISIÓN PARA PEAJES DE LA SERRANÍA DEL PERÚ, HACIENDO USO DE MICROCONTROLADORES Y UN LENGUAJE DE PROGRAMACIÓN VISUAL

## TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE:

**INGENIERO MECANICO ELECTRICISTA**

**PRESENTADO POR:**

**ALFREDO MOISÉS, QUISPE ESPINOZA**

**PROMOCIÓN 2002 - II**

**LIMA – PERÚ**

# ÍNDICE

Planos

Anexos

# PROLOGO

El presente trabajo tiene como finalidad la solución específica de un problema actual en nuestra realidad con respecto a los peajes en la sierra y selva, alejados a nuestra capital. Se detallará como desarrolló un sistema de control para el conteo de vehículos en una vía de peajes en estos lugares. Con este se hizo uso de microcontroladores, lenguaje de programación visual y diseño por medio de cálculo de elementos finitos.

***Capítulo II.*** Aquí se describe el problema y se plantea una solución para resolverlo.

***Capítulo III.*** Se detallará el diseño de los sensores inductivos, los cuales sirven para la detección de vehículo, la tarjeta acondicionadora de señal para el sensor inductivo, sensor de temperatura, sensor de presión, sensor de estado de la puerta del tablero eléctrico y el sensor de nivel de carga de las baterías. Además se verá cómo se diseño la tarjeta de control y el programa en el microcontrolador PIC16F877A de MICROCHIP. Esta tarjeta de control se encargará de clasificar los vehículos detectados y guardar dicha información con sus características de cada vehículo, así como la hora, minuto y segundo en que pasó. Además en este capítulo

se verá el rediseño de los sensores de presión para el conteo de número de ejes de los vehículos, ya que los que hay en el mercado, para este tipo de aplicación, tienen un tiempo de vida corta. Para los diseños y desarrollo usaron los software de cálculo de elementos finitos COSMOS DESIGNSTAR, PROTEUS y CIRCUIT MAKER.

***Capítulo IV.*** En este capítulo se mostrará el diseño del programa desarrollado en LABVIEW 7.1 para la recopilación de la información almacenada en las memorias de la tarjeta de control, la generación de reportes, la configuración de la tarjeta de control, el mantenimiento de los sensores y del equipo.

***Capítulo V.*** Se explicará el procedimiento a seguir para la instalación adecuada y el buen funcionamiento del equipo, además se mostrará el plan de mantenimiento seguir.

***Capítulo VI.*** Se explicará todos los resultados de las pruebas que se realizaron, observando las dificultades que se tuvieron y las soluciones que se dieron.

***Capítulo VII.*** Se detallará los costos de implementación del equipo, comparándolos con los costos de los equipos que existen en el mercado.

***Capítulo VIII.*** Se planteará una propuesta para la interconexión de los equipos a una red diseñando un sistema de supervisión similar a un SCADA para la supervisión y control. Para esto se convertirá el protocolo de comunicación RS232 al protocolo TCP/IP, conectándose el equipo a INTERNET. El equipo será supervisado

desde un computador en Lima en el que se encuentra un programa desarrollado en LABVIEW.

Finalmente se presentan las conclusiones, observaciones y recomendaciones en la elaboración y la implementación de este trabajo.

# CAPÍTULO I

# INTRODUCCIÓN

Hoy en día, con el crecimiento de la tecnología, el estudio y el análisis en ingeniería han llegado a niveles tales que los tiempos de realización de proyectos se han reducido, así como sus costos, ya que todo se va reduciendo en el análisis computacional. Por ejemplo con el desarrollo de los software CAD (Diseño asistido por computador) y CAE (Ingeniería asistido por computador) se puede llegar analizar estructuras, máquinas, transmisiones mecánicas, todo tipo de piezas mecánicas, bancos de pruebas para automóviles, etc. con un menor costo y tiempo, además de un diseño mucho más eficiente. Además, con el crecimiento de la tecnología, la electrónica se ha desarrollado, y sus aplicaciones en casi todas las áreas como la ingeniería, medicina, etc., van creciendo día a día. Debido a esto resulta difícil pensar en un sistema que sea puramente mecánico, eléctrico o electrónico; sino más bien integrado, en donde la electrónica a proporcionado un mayor control y flexibilidad para que éstos sistemas sean programables fácilmente. Aquí se pueden destacar los software de simulación para las aplicaciones en electricidad y electrónica, con los que se pueden depurar errores y confirmar datos antes del diseño de tarjetas. Una de las cosas que se han logrado con la tecnología es la fabricación de sistemas digitales embebidos con una alta escala de integración como son los microprocesadores,

microcontroladores, procesadores digitales de señales, etc.; estos dispositivos hacen posible la implementación de aplicaciones que antes eran muy costosas y complicadas de desarrollar e implementar. Otra de las cosas que también hoy en día se hace necesario, son los sistemas de comunicación; es decir, es imposible pensar en un sistema aislado, sino más bien todos están interconectados, comunicados mediante algún protocolo y con un entorno visual.

La presente tesis resolverá un problema presentado, desde ya hace mucho tiempo, en los sistemas de control para peajes en la sierra del Perú. Para lo cual se hará uso de las herramientas de software y hardware que se mencionaron; es decir microcontroladores PIC para el diseño de las tarjetas de control, software de entorno visual LABVIEW para el HMI (Interface Hombre Máquina) y generación de reportes, protocolos de comunicación, software de simulación para electricidad y electrónica PROTEUS y CIRCUIT MAKER, software CAD INVERTOR y CAE COSMOS DESIGNSTAR para la selección y diseño de sensores de presión.

# CAPÍTULO II

# PROBLEMÁTICA Y PLANTAMIENTO DE LA SOLUCIÓN

## 2.1. PROBLEMÁTICA

La falta de sistemas de control y supervisión (en peajes de la sierra y selva del Perú) eficientes, confiables y de bajo consumo de energía, han ocasionado grandes pérdidas debido a las irregularidades en la recaudación. El Ministerio de Transportes y Comunicaciones, a través de "PROVÍAS NACIONAL", es el encargado de la mayoría de los peajes en la sierra y selva, estando en su mayoría en lugares alejados de la ciudad. Debido a esto, se tiene un grupo electrógeno, el cual se enciende sólo en las noches por razones de costo. Por esta razón "PROVÍAS NACIONAL", pese a muchas veces convocar a concurso o comprar equipos de diferentes empresas para funciones puntuales, no ha conseguido resolver el problema, ya que todos los sistemas de este tipo suelen depender de un computador y PLC los cuales consumen demasiada energía; además que sus sistemas son fáciles de vulnerar y que, al depender de una computadora, los cobradores cuentan con la facilidad de interrumpir la comunicación del sistema, lo que ocasiona un cese de conteo. Hubo muchas propuestas para compensar la falta de energía; sin embargo otro problema ha sido que el costo de los equipos no justificaba el recaudo en esos lugares, ya que a esto hay que incluir el

mantenimiento mensual. Otro problema que era común es que los sensores de presión, que usan en estos peajes para la detección de ejes, tenían un tiempo de vida muy corto y esto encarecía el mantenimiento.

## 2.2. PLANTEAMIENTO DE LA SOLUCIÓN

En la actualidad diversas empresas, tanto colombianas, argentinas y peruanas (tal es el caso de SIEMMENS, SEYMA y otras) han desarrollado muchos sistemas; pero no cumplen los requerimientos que se desean para estos peajes de la sierra y selva. Además, mantener éstos equipos en esas condiciones requiere de mucha inversión, añadiendo a esto la forma como trabajan los equipos, éstos no garantiza el buen almacenamiento y seguridad de la información. Debido a esta necesidad es que se pensó en dar una adecuada solución, presentando un proyecto a "PROVÍAS NACIONAL" para desarrollar equipos apropiados y con la posibilidad de interconectarlos.

Para satisfacer todos los requerimientos, se diseñará el sistema haciendo uso de microcontroladores PIC de MICROCHIP y para el desarrollo del software de supervisión y control en la PC se usará el lenguaje de programación gráfica LABVIEW de NATIONAL INSTRUMENTS. Además se rediseñará los sensores de presión utilizando cálculo por elementos finitos. En la figura 2.1 se observa el esquema general del sistema que se desarrollará.

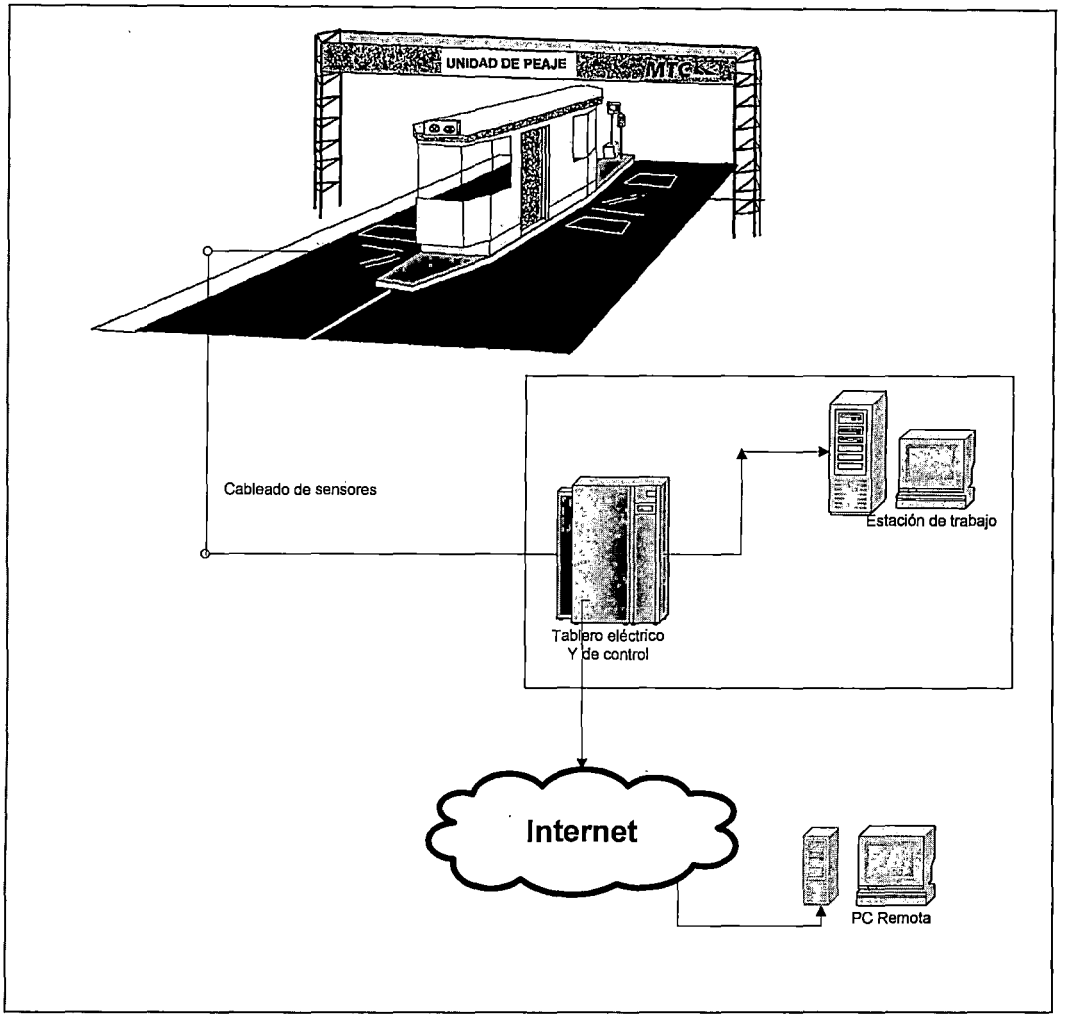



***Figura 2.1.-*** *Sistema general de supervisión y control.*

## 2.3. OBJETIVO GENERAL

Diseño e implementación de un sistema de supervisión y control para peajes en la serranía del Perú, con bajo costo y reducido consumo de energía.

## 2.4. OBJETIVOS ESPECÍFICOS

- Diseño de la tarjeta de adquisición y control basada en microcontroladores PIC de MICROCHIP.

- Sistema de almacenamiento de energía, con capacidad de abastecimiento hasta para una semana.
- Capacidad de almacenamiento de datos hasta por un mes sin pérdidas.
- Configuración del sistema de supervisión y control, desde un computador.
- Descarga de información directa y/o en forma remota.
- Sistema de ordenamiento de información y generación de reportes, por hora y por turno, desarrollado en lenguaje visual LABVIEW de NATIONAL INSTRUMENTS.
- Diseño de una aplicación en PC para mantenimiento y monitoreo del equipo completo.
- Mejora del diseño de sensores de presión para el conteo de ejes.
- Posibilidad de descarga remota de la información desde la misma central en Lima.

# CAPÍTULO III

# DISEÑO DEL SISTEMA ELÉCTRICO ELECTRÓNICO Y MECÁNICO

## 3.1. EL MICROCONTROLADOR PIC16F87X

Para el desarrollo de la tarjeta de control se usó el microcontrolador PIC16F877A de MICROCHIP, el cual pertenece a la familia 16F87X. En esta familia se encuentran los microcontroladores 16F873, 16F874, 16F876 y 16F877 quienes tienen una arquitectura y funcionamiento similar.

El microcontrolador es un dispositivo el cual posee un microprocesador, puertos de entrada y salida, así como algunos recursos tales como temporizadores, conversor analógico digital, comunicación serial, etc.

### 3.1.1. Los puertos paralelos de entrada/salida

Los integrados PIC16F874 y PIC16F877 poseen 5 puertos de entrada / salida denominados PORTA, PORTB,..., PORTE, mientras que el PIC16F873 y PIC16F876 sólo hasta el PORTC.

Estos puertos son totalmente programables, es decir, sus líneas pueden ser configuradas para trabajar como entradas o como salidas a selección del programador.

### 3.1.2. El puerto serial USART

La USART (Universal Synchronous Asynchronous Receiver Transmitter) es uno de los dos periféricos contenidos en el PIC que le permiten realizar comunicación en serie. El otro es el MSSP (Master Synchronous Serial Port), el cual no es tratado en estas notas.

La USART, también conocida como SCI (Serial Communications Interface) puede configurarse como una unidad de comunicación en serie para la transmisión de datos asíncrona con dispositivos tales como terminales de computadora o computadoras personales, o bien para comunicación síncrona con dispositivos tales como convertidores A/D o D/A, circuitos integrados o memorias EEPROM con comunicación serie, etc. Una gran cantidad de periféricos se comunican actualmente en serie con una microcomputadora: líneas telefónicas, terminales remotas, unidades de casete magnético, el ratón, teclados, etc.

### 3.1.3. La usart del PIC16F87x

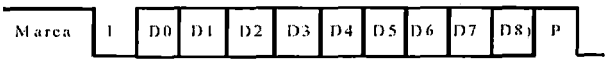
La USART del PIC puede ser configurada para operar en tres modos:

- Modo Asíncrono (full duplex (transmisión y recepción simultáneas)),
- Modo Síncrono – Maestro (half duplex)

- Modo Síncrono – Esclavo (half duplex)

**Modo Asíncrono.**

En este modo la USART usa un formato estándar **NRZ** asíncrono, el cual para la sincronización usa: 1 bit de inicio (**I**), 8 o 9 bits de datos y 1 bit de paro (**P**). Mientras no se están transmitiendo datos la USART envía continuamente un bit de marca. El modo asíncrono se selecciona limpiando el bit SYNC del registro **TXSTA (98H)**. El modo asíncrono es deshabilitado durante el modo SLEEP. Cada dato es transmitido y recibido comenzando por el LSB. El hardware no maneja bit de Paridad, pero el noveno bit puede ser usado para este fin y manejado por software.




El módulo Asíncrono de la USART consta de 4 módulos fundamentales:

- El circuito de muestreo
- El generador de frecuencia de transmisión (Baud Rate)
- El transmisor asíncrono
- El receptor asíncrono.

El circuito de muestreo.- El dato en el pin de recepción (RC7/RX/DT) es muestreado tres veces para poder decidir mediante un circuito de mayoría, si se trata de un nivel alto o un nivel bajo.

## El Generador de Baud Rate (BRG)

Este generador sirve tanto para el modo síncrono como el asíncrono y consiste de un contador/divisor de frecuencia de 8 bits controlado por el registro **SPBRG (99H)**. De tal manera que la frecuencia de transmisión se calcula de acuerdo a la tabla 3.1, en la cual la variable "X" es el registro SPBRG de 8bits el cual es un registro divisor. El bit BRGH corresponde a TXSTA<2>. En la tabla 3.2 y 3.3 se muestran un resumen de la aplicación de las fórmulas para diversos casos.

***Tabla 3.1***

*Para cálculo de la velocidad en baudios*

| SYNC | BRGH=0 (baja velocidad) | BRGH=1 (Alta velocidad) |
|---|---|---|
| 0 (modo asíncrono) | Baud rate=Fosc/(64(X+1)) | Baud rate=Fosc/(16(X+1)) |
| 1 (modo síncrono) | Baud rate=Fosc/(4(X+1)) | - |

***Tabla 3.2***

*Tabulación de valores del SPBRG para diferentes velocidades y cristales para BRGH=0*

| BAUD RATE (K) | Fosc = 20 MHz | | | Fosc = 16 MHz | | | Fosc = 10 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | - | - | - | - | - | - | - | - | - |
| 1.2 | 1.221 | 1.75 | 255 | 1.202 | 0.17 | 207 | 1.202 | 0.17 | 129 |
| 2.4 | 2.404 | 0.17 | 129 | 2.404 | 0.17 | 103 | 2.404 | 0.17 | 64 |
| 9.6 | 9.766 | 1.73 | 31 | 9.615 | 0.16 | 25 | 9.766 | 1.73 | 15 |
| 19.2 | 19.531 | 1.72 | 15 | 19.231 | 0.16 | 12 | 19.531 | 1.72 | 7 |
| 28.8 | 31.250 | 8.51 | 9 | 27.778 | 3.55 | 8 | 31.250 | 8.51 | 4 |
| 33.6 | 34.722 | 3.34 | 8 | 35.714 | 6.29 | 6 | 31.250 | 6.99 | 4 |
| 57.6 | 62.500 | 8.51 | 4 | 62.500 | 8.51 | 3 | 52.083 | 9.58 | 2 |
| HIGH | 1.221 | - | 255 | 0.977 | - | 255 | 0.610 | - | 255 |
| LOW | 312.500 | - | 0 | 250.000 | - | 0 | 156.250 | - | 0 |

| BAUD RATE (K) | Fosc = 4 MHz | | | Fosc = 3.6864 MHz | | |
|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | 0.300 | 0 | 207 | 0.3 | 0 | 191 |
| 1.2 | 1.202 | 0.17 | 51 | 1.2 | 0 | 47 |
| 2.4 | 2.404 | 0.17 | 25 | 2.4 | 0 | 23 |
| 9.6 | 8.929 | 6.99 | 6 | 9.6 | 0 | 5 |
| 19.2 | 20.833 | 8.51 | 2 | 19.2 | 0 | 2 |
| 28.8 | 31.250 | 8.51 | 1 | 28.8 | 0 | 1 |
| 33.6 | - | - | - | - | - | - |
| 57.6 | 62.500 | 8.51 | 0 | 57.6 | 0 | 0 |
| HIGH | 0.244 | - | 255 | 0.225 | - | 255 |
| LOW | 62.500 | - | 0 | 57.6 | - | 0 |

***Tabla 3.3***

*Tabulación de valores del SPBRG para diferentes velocidades y cristales para BRGH=1*

| BAUD RATE (K) | Fosc = 20 MHz | | | Fosc = 16 MHz | | | Fosc = 10 MHz | | |
|---|---|---|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | - | - | - | - | - | - | - | - | - |
| 1.2 | - | - | - | - | - | - | - | - | - |
| 2.4 | - | - | - | - | - | - | 2.441 | 1.71 | 255 |
| 9.6 | 9.615 | 0.16 | 129 | 9.615 | 0.16 | 103 | 9.615 | 0.16 | 64 |
| 19.2 | 19.231 | 0.16 | 64 | 19.231 | 0.16 | 51 | 19.531 | 1.72 | 31 |
| 28.8 | 29.070 | 0.94 | 42 | 29.412 | 2.13 | 33 | 28.409 | 1.36 | 21 |
| 33.6 | 33.784 | 0.55 | 36 | 33.333 | 0.79 | 29 | 32.895 | 2.10 | 18 |
| 57.6 | 59.524 | 3.34 | 20 | 58.824 | 2.13 | 16 | 56.818 | 1.36 | 10 |
| HIGH | 4.883 | - | 255 | 3.906 | - | 255 | 2.441 | - | 255 |
| LOW | 1250.000 | - | 0 | 1000.000 | | 0 | 625.000 | - | 0 |

| BAUD RATE (K) | Fosc = 4 MHz | | | Fosc = 3.6864 MHz | | |
|---|---|---|---|---|---|---|
| | KBAUD | % ERROR | SPBRG value (decimal) | KBAUD | % ERROR | SPBRG value (decimal) |
| 0.3 | - | - | - | - | - | - |
| 1.2 | 1.202 | 0.17 | 207 | 1.2 | 0 | 191 |
| 2.4 | 2.404 | 0.17 | 103 | 2.4 | 0 | 95 |
| 9.6 | 9.615 | 0.16 | 25 | 9.6 | 0 | 23 |
| 19.2 | 19.231 | 0.16 | 12 | 19.2 | 0 | 11 |
| 28.8 | 27.798 | 3.55 | 8 | 28.8 | 0 | 7 |
| 33.6 | 35.714 | 6.29 | 6 | 32.9 | 2.04 | 6 |
| 57.6 | 62.500 | 8.51 | 3 | 57.6 | 0 | 3 |
| HIGH | 0.977 | - | 255 | 0.9 | - | 255 |
| LOW | 250.000 | - | 0 | 230.4 | - | 0 |

Debido a que el divisor es de 8 bits, no se puede tener cualquier velocidad de transmisión deseada, ya que X se deberá redondear al entero más cercano. En las dos tablas anteriores se muestran algunos valores en baudios estándares, el divisor necesario (X=SPBRG) bajo diferentes frecuencias "Fosc" y el error producido en porcentaje.

## El transmisor asíncrono

En la figura 3.1 se muestra el diagrama de bloques del transmisor de la USART.

El corazón de este módulo es el registro de corrimiento (transmit shift register, TSR). La única manera de acceder al registro TSR es a través del registro **TXREG (19H)**.

Para transmitir un dato, el programa deberá ponerlo primero en el registro TXREG. En cuanto el TSR termina de enviar el dato que tenía (en cuanto transmite el bit de paro) lee el dato contenido en TXREG (si hay alguno) esto ocurre en un ciclo TCY. En cuanto el dato de TXREG es transferido al TSR el TXREG queda vacío esta condición es indicada mediante el bit bandera TXIF (que es el bit 4 del registro **PIR1 (0Ch)**), el cual se pone en alto. Este bit NO puede ser limpiado por software, sólo dura un instante en bajo cuando se escribe un nuevo dato a TXREG. Si se escribe un dato seguido de otro (back to back) a TXREG el primero se transfiere inmediatamente a TSR y el otro tiene que esperar hasta que el TSR termine de enviar el bit de Stop del primero. Durante esta espera TXIF permanece en bajo.

Existe otro bit, llamado TRMT (TXSTA<1>), el cual muestra el estado del TSR. TRMT se pone en alto cuando TSR está vacío, y en bajo cuando TSR está transmitiendo un dato.

Mientras que TXIF puede generar una interrupción TRMT no lo puede hacer, TRMT está pensado para ser consultado por "poleo" (sin usar interrupciones).

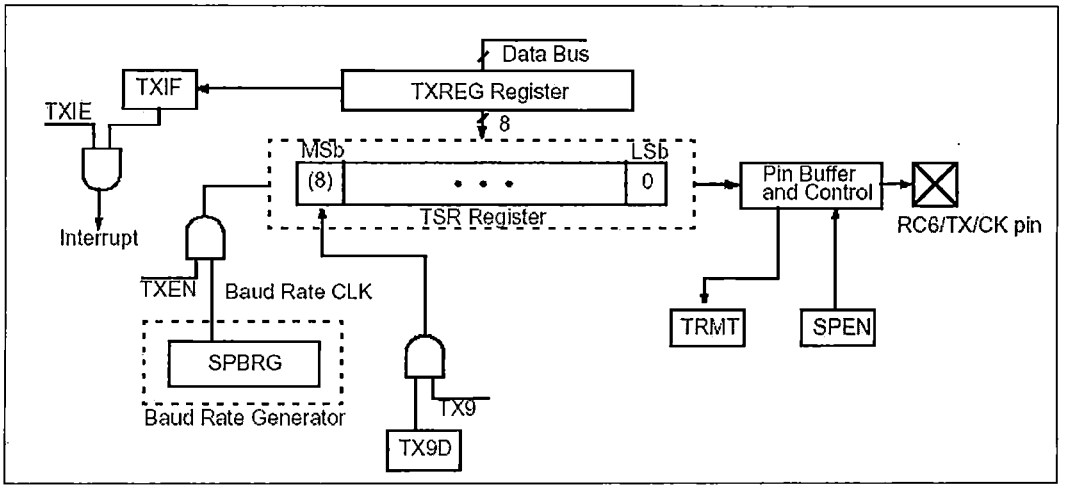



***Figura 3.1.-*** *Diagrama de bloques del transmisor de la USAT*

Para habilitar el módulo de transmisión es necesario poner en alto el bit TXEN (TXSTA<5>), mientras no se habilite el módulo, el pin de transmisión (RC6/TX/CK) se mantiene en alta impedancia. Si TXEN es deshabilitada a la mitad de una transmisión, está será abortada y el transmisor será reseteado.

Si se está usando un noveno bit TX9 (TXSTA<6>), éste deberá ser escrito antes de escribir los 8 bits restantes a TXREG, ya que en cuanto se escribe un dato a este registro inmediatamente es transferido a TSR (si éste está vacío). El diagrama de bloques del funcionamiento de la transmisión se muestra en la figura 3.1

De acuerdo a lo anterior, la **inicialización del módulo de transmisión** consiste en los siguientes pasos:

1. Inicializar baud rate escribiendo al registro SPBRG el divisor adecuado y opcionalmente al bit BRGH.
2. Habilitar comunicación asíncrona limpiando el bit SYNC y poniendo el bit SPEN.
3. Si se van a usar interrupciones, poner el bit TXIE (PIE<4>).
4. Poner el bit TX9 si se desea transmitir datos de 9 bits
5. Habilitar transmisión poniendo el bit TXEN, lo cual pondrá el bit TXIF.
6. Colocar el noveno bit del dato en TX9D si se están usando datos de 9 bits.
7. Cargar el dato al registro TXREG (inicia la transmisión).

**El receptor asíncrono**

El módulo de recepción es similar al de transmisión, en la siguiente figura se muestran los bloques que lo constituyen. Una vez que se ha seleccionado el modo asíncrono, la recepción se habilita poniendo en alto el bit **CREN** (RCSTA<4>). El dato es recibido mediante la línea RC7/**RX**/DT, la cual maneja un registro de desplazamiento de alta velocidad.

El registro de desplazamiento RSR no es accesible por software; pero cuando el dato recibido se ha completado, el dato de RSR se transfiere automáticamente al registro **RCREG** si éste está vacío y al mismo tiempo se pone en alto la bandera de recepción RCIF. La única manera de limpiar la bandera RCIF es leyendo el los datos del registro RCREG. El registro

RCREG puede contener hasta dos datos, ya que es un buffer doble que funciona como una cola de dos posiciones.

Si las dos posiciones del registro RCREG están llenas (no han sido leídas) y se detecta el bit de Stop de un tercer dato de recepción, lo cual ocasiona un transferencia automática del dato recibido a RCREG, esto destruirá el primer dato recibido y activará el indicador de sobre escritura OERR (RCSTA<1>). Para evitar esto, se deberán leer los dos datos en RSREG haciendo dos lecturas consecutivas.

La única manera de limpiar el bit OERR una vez que ha sido activado es reseteando el módulo de recepción (limpiando CREN y volviéndolo a poner), si no se limpia OERR se bloquea la transferencia de datos de RSR a RCREG y no puede haber más recepción de datos.

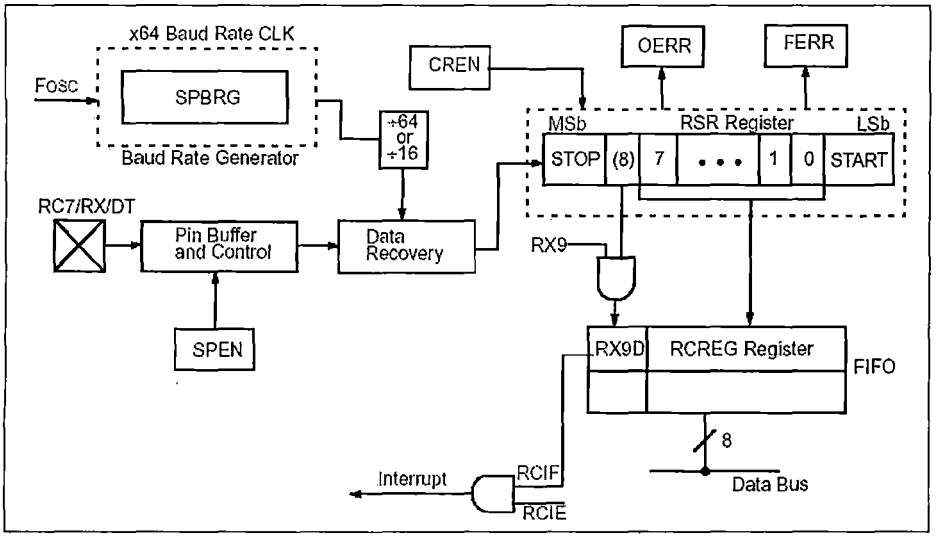



***Figura 3.2.-Diagrama de bloques de la recepción USART***

Si se detecta un bit nivel bajo en la posición del bit de stop se pone el indicador de error de encuadre (frame error) FERR RCSTA<2>. Tanto este indicador como el noveno bit RX9D de los datos están en una cola de dos posiciones al igual que los datos recibidos, de manera que al leer RCREG se actualizan FERR y RX9D con nuevos valores, por lo cual estos bits deberán ser leídos antes de leer RCREG para no perder su información. El diagrama de bloques en el que se puede apreciar la forma como se desarrolla la recepción se muestra en la figura 3.2

De acuerdo a lo anterior, la inicialización del módulo de recepción es como sigue:

1. Inicializar el baud rate escribiendo al registro SPBRG el divisor adecuado y opcionalmente al bit BRGH.
2. Habilitar el puerto serie asíncrono limpiando el bit SYNC y poniendo el bit SPEN.
3. Si se van a usar interrupciones, poner el bit RCIE (PIE<5>).
4. Si se desea recepción de datos de 9 bits se deberá poner el bit RX9 (RCSTA<0>).
5. Habilitar la recepción poniendo el bit CREN (RCSTA<4>)
6. El bit RCIF se pondrá cuando la recepción de un dato se complete y se generará una interrupción si RCIE está puesto.
7. Leer el registro RCSTA para obtener el noveno bit (si se están recibiendo datos de 9 bits) o para determinar si ha ocurrido un error de recepción.

8. Leer los 8 bits del dato recibido leyendo el registro RCREG.

9. Si ocurrió algún error este se limpia al limpiar el bit CREN, el cual deberá volver a ponerse si se desea continuar la recepción.

### 3.1.4. El conversor analógico digital

Los PIC16F87X poseen un módulo ADC interno que les permite manejar 5 entradas analógicas para los dispositivos de 28 pines y 8 para los otros dispositivos. En la siguiente figura se muestra un diagrama de bloques del módulo ADC. En la figura 3.3 se puede ver el diagrama de bloque del módulo ADC del microcontrolador.

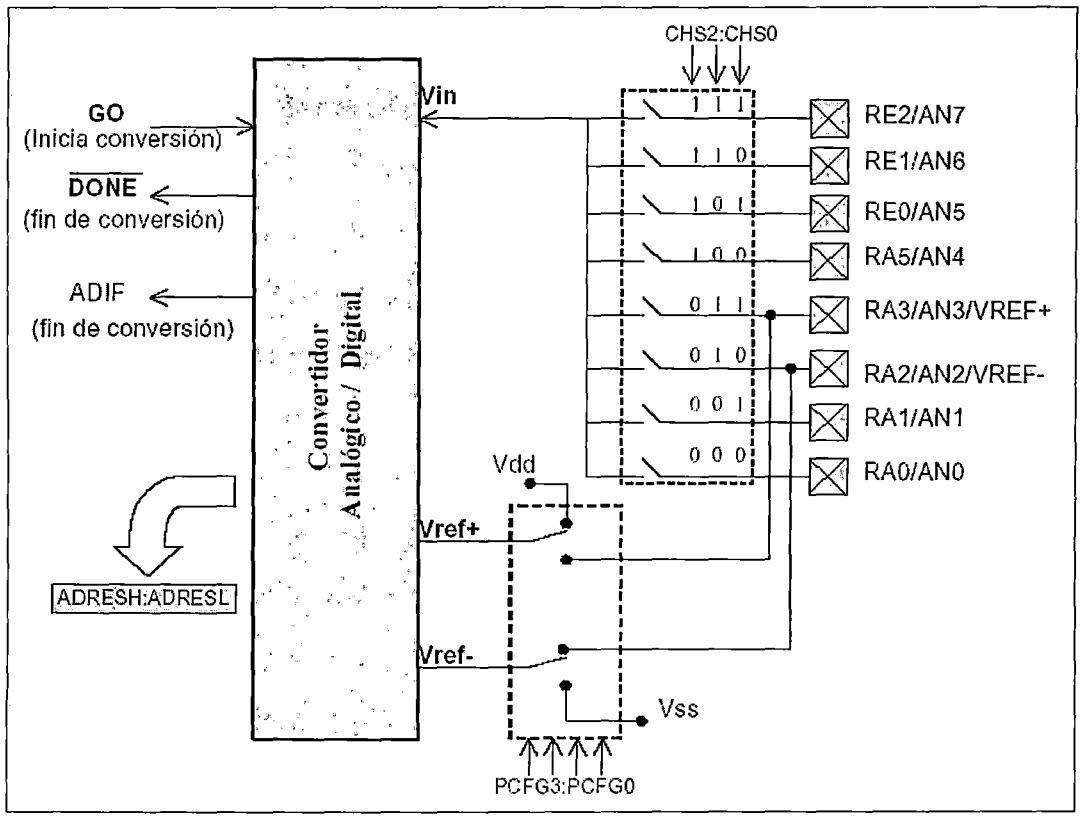



***Figura 3.3.-*** *Diagrama de bloques del convertidor AD del microcontrolador*

**El multiplexor**.- El ADC es un convertidor de aproximaciones sucesivas de 10 bits, el cual puede realizar la conversión de una de las 8 entradas (o canales) analógicas AN0,..., AN7 multiplexadas por la lógica interna que utiliza como líneas de selección del canal los bits CHS2:CHS0, en donde se coloca el número en binario del canal a convertir.

**Voltajes de Referencia**.- Todo convertidor ADC requiere voltajes de referencia que determinan el valor de mínima escala (VREF-) y el de plena escala (VREF+), de manera que la conversión de un valor de voltaje analógico Vin en el rango de VREF- a VREF+ producirá un valor equivalente binario D en el rango de 0 a $2^n$, donde n es la resolución del convertidor (n = 10). Como la relación entre escalas es lineal, una regla de tres nos da la relación entre el voltaje analógico de entrada (Vin) y el valor digital (D) obtenido por el ADC, de acuerdo a la siguiente expresión:

$$\frac{D}{2^n - 1} = \frac{V_{in} - V_{REF-}}{V_{REF+} - V_{REF-}}$$

Con la elección más común: VREF+ = VDD = 5v, VREF- = VSS= 0v, y como n=10, obtenemos:

$$D = \frac{1023}{5} V_{in} = 204.6 V_{in}$$

De donde se ve que cuando Vin varía en todo su rango, desde 0 hasta 5v, el valor obtenido D varía también en todo su rango, de 0 a 1023.

Si a la inversa, obtenemos un valor D y deseamos saber que voltaje representa, basta con despejar:

$$V_{in} = (\frac{5}{1023})D = (0.004887585533)D$$

**Observación**: Como puede verse, la conversión del dato D al voltaje correspondiente requiere una multiplicación por un número fraccionario, para lo cual el PIC no posee instrucciones, si deseamos realizar esta multiplicación en el PIC debemos hacer un programa que multiplique números de punto fijo o de punto flotante.

**El proceso de Conversión Analógico/Digital.**

En la figura 3.4 vemos que el diagrama de tiempo se muestran los eventos que tienen lugar durante el proceso de una conversión analógico / digital.

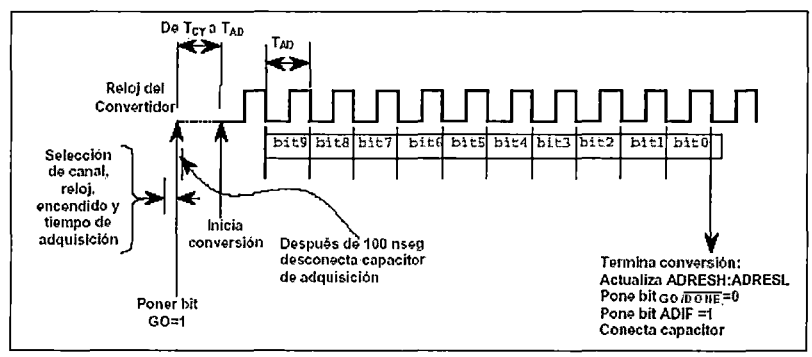



***Figura 3.4.-*** *Diagrama de tiempos del ADC*

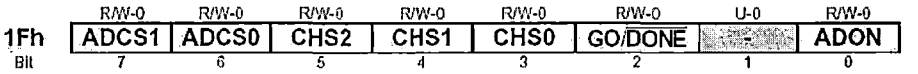
De acuerdo a la figura, para echar a andar el convertidor se deberán seguir los siguientes pasos:

1. Configurar el módulo A/D:
   - Configurar los pines analógicos y los Voltajes de referencia VREF- y VREF+, mediante el registro ADCON1 (9Fh) (y los correspondientes bits TRIS como entradas)
   - Seleccionar el canal de entrada a convertir mediante los bits CHS2:CHS0 del registro ADCON0 (1Fh)
   - Seleccionar el reloj de conversión mediante los bits ADCS1:ADCS2
   - (ADCON0<7:6>)
   - Energizar el convertidor mediante el bit ADON (ADCON0<0>)
2. Configurar interrupciones para el convertidor A/D (si se desea), para ello: limpiar ADIF y poner ADIE, PEIE y GIE.
3. Esperar mientras transcurre el tiempo de adquisición (unos 20 μseg).
4. Iniciar la conversión poniendo el bit GO/DONE (ADCON0<2>).
5. Esperar a que termine la conversión:
   - Por "poleo" (Polling): Consultando continuamente el bit GO/DONE (el cual es limpiado por el convertidor cuando la conversión está completa).
   - Por interrupciones: Cuando la conversión termina, la bandera ADIF se activa y esto genera una solicitud de interrupción, la cual deberá ser atendida por una rutina de atención a la interrupción diseñada para ello.
6. Leer el dato convertido D de los registros (ADRESH:ADRESL)

7. Para la siguiente conversión, esperar al menos 2TAD (Donde TAD es el tiempo de conversión por bit).

**Los Registros de Control**

A continuación se presenta un resumen de los registros relacionados con la operación del convertidor. En la figura 3.5 se muestra el registro ADCON0, y el registro ADCON1 se muestra en la figura 3.6.




Registro ADCON0 (1Fh)

***Figura 3.5***

**bits 7-6 ADCS1:ADCS0**.- Selección de reloj de acuerdo a la tabla 3.3.

***Tabla 3.3***
*(Para seleccionar el TAD)*

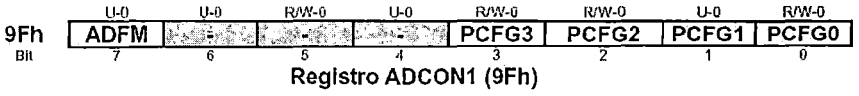
| ADCS1 | ADCS0 | Frec. seleccionada |
|---|---|---|
| 0 | 0 | $F_{OSC}/2$ |
| 0 | 1 | $F_{OSC}/8$ |
| 1 | 0 | $F_{OSC}/32$ |
| 1 | 1 | $F_{RC}$ (oscilador RC interno)* |

**bits 5-3 CHS2:CHS0**.- Selección de canal analógico a convertir. Se selecciona uno de los ocho canales AN0,..., AN7 colocando en estos tres bits el número binario correspondiente al canal deseado. (Los canales analógicos a usar deberán tener sus bits TRIS correspondientes seleccionados como entradas).

**bit 2 GO/DONE**.- Bit de inicio y fin de conversión.- Con el convertidor encendido, poniendo este bit en 1 se inicia la conversión del canal

seleccionado. Este bit permanece en 1 durante la conversión y es limpiado automáticamente por el convertidor al terminar la conversión.

**bit 0 ADON**.-Encendido del convertidor. Al poner este bit en 1 el convertidor se enciende y al ponerlo en 0 se apaga y no consume corriente.




Registro ADCON1 (9Fh)

***Figura 3.6***

**bit 7 ADFM**.- Selección de formato del resultado. Al ponerlo en 1 se selecciona resultado de 10 bits justificado a la derecha. Y con un 0 se selecciona justificación a la izquierda. En la siguiente sección se explica con mayor detalle.

**bits 3-0 PCFG3:PCFG0**.- Bits de configuración de las entradas del convertidor. Configuran los pines de entrada del convertidor de acuerdo a la tabla 3.4 (en donde A = Entrada Analógica D = Entrada /Salida digital)

***Tabla 3.4***

*(Para la configuración de canales y voltajes de referencias del ADC)*

| PCFG3: PCFG0 | AN7[1] /RE2 | AN6[1] /RE1 | AN5[1] /RE0 | AN4/ RA5 | AN3/ RA3 | AN2/ RA2 | AN1/ RA1 | AN0/ RA0 | $V_{REF+}$ | $V_{REF-}$ | Can[2] /Refs |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | A | A | A | A | A | A | A | A | VDD | VSS | 8/0 |
| 0001 | A | A | A | A | $V_{REF+}$ | A | A | A | RA3 | VSS | 7/1 |
| 0010 | D | D | D | A | A | A | A | A | VDD | VSS | 5/0 |
| 0011 | D | D | D | A | $V_{REF+}$ | A | A | A | RA3 | VSS | 4/1 |
| 0100 | D | D | D | D | A | D | A | A | VDD | VSS | 3/0 |
| 0101 | D | D | D | D | $V_{REF+}$ | D | A | A | VDD | VSS | 2/1 |
| 011x | D | D | D | D | D | D | D | D | VDD | VSS | 0/0 |
| 1000 | A | A | A | A | $V_{REF+}$ | $V_{REF-}$ | A | A | RA3 | RA2 | 6/2 |
| 1001 | D | D | A | A | A | A | A | A | VDD | VSS | 6/0 |
| 1010 | D | D | A | A | $V_{REF+}$ | A | A | A | RA3 | VSS | 5/1 |
| 1011 | D | D | A | A | $V_{REF+}$ | $V_{REF-}$ | A | A | RA3 | RA2 | 4/2 |
| 1100 | D | D | D | A | $V_{REF+}$ | $V_{REF-}$ | A | A | RA3 | RA2 | 3/2 |
| 1101 | D | D | D | D | $V_{REF+}$ | $V_{REF-}$ | A | A | RA3 | RA2 | 2/2 |
| 1110 | D | D | D | D | D | D | D | A | VDD | VSS | 1/0 |
| 1111 | D | D | D | D | $V_{REF+}$ | $V_{REF-}$ | D | A | RA3 | RA2 | 1/2 |

**Los Registros de Resultados.**

El par de registros **ADRESH:ADRESL (1Eh:9Eh)** son cargados con el dato (D) resultante de una conversión analógico / digital al terminar ésta. Cada uno de estos registros es de 8 bits, por lo tanto, juntos pueden guardar hasta 16 bits. Sin embargo, como el resultado D es de 10 bits, el módulo de conversión A/D permite justificarlo (alinearlo) en la parte izquierda o derecha de los 16 bits disponibles, para elegir alguna de las dos opciones se usa el bit ADFM (ADCON1<7>) como se muestra en la figura 3.7.

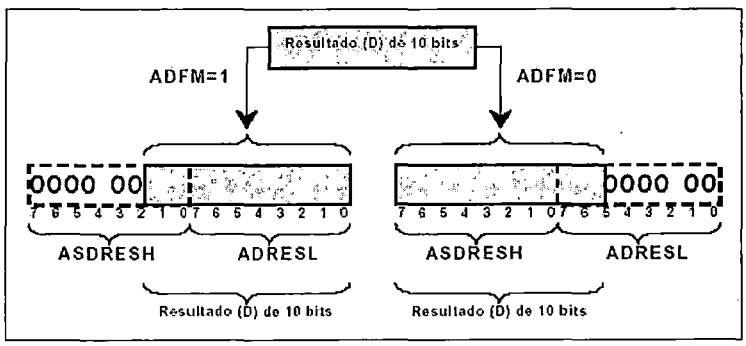



***Figura 3.7.- Justificación del dato digitalizado***

### 3.1.5. El módulo del TIMER1

El TIMER1 a diferencia del TIMER0 es un contador / temporizador de 16 bits. El conteo es realizado por dos registros de 8 bits: (**TMR1H (0Fh) y TMR1L (0Eh)**), estos dos registros se pueden leer y escribir. Al par de registros TMR1H:TMR1L los denominaremos por comodidad como si fueran un solo registro de 16 bits (**TMR1**).

Así, el registro TMR1 se incrementa de 0000h a FFFFh y en la siguiente cuenta se reinicia en 0000h y así sucesivamente, al reciclarse se activa (en alto) la bandera **TMR1IF (PIR1<0>)**, la cual puede ser utilizada para generar una interrupción, o bien, para ser consultada por poleo.

En la figura 3.8 se muestra un diagrama de bloques de este módulo, en donde se indican los bits que afectan su operación y la manera en que lo hacen.

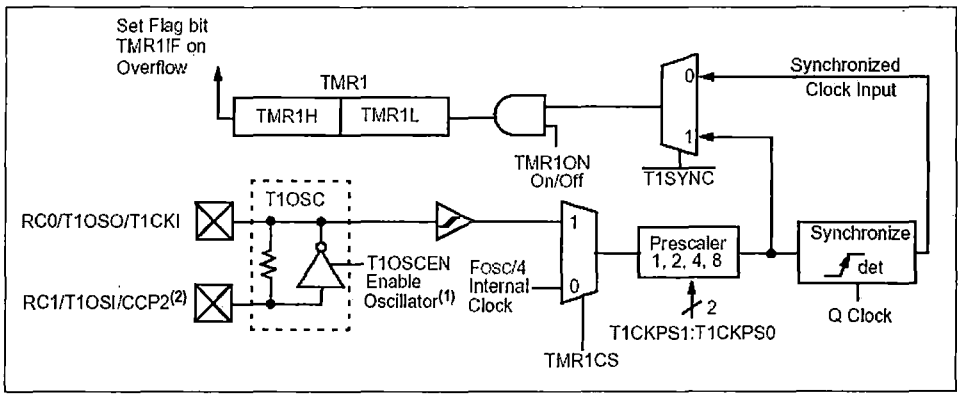



***Figura 3.8.-*** *Diagrama de bloques del TIMER1*

### 3.1.6. Manejo de interrupciones

Se le llama interrupción a un salto especial a una subrutina que no está contemplado en un punto específico del programa principal, sino que puede ocurrir en cualquier punto de éste y ***no es provocado por una instrucción en el programa, sino por un evento*** interno o externo al sistema del microcontrolador.

Los dispositivos que manejan eventos capaces de provocar una solicitud de interrupción se denominan fuentes de interrupción. La familia del PIC16F87x cuenta con hasta 14 fuentes de interrupción.

Cada fuente de interrupción posee dos bits asociados a ella:

- Una Bandera (terminada en F) de Interrupción, la cual es activada (en alto) por el evento para solicitar una interrupción.
- Una Máscara (terminada en E) Local de Interrupción, la cual si está desactivada (en bajo) bloqueará la solicitud de interrupción correspondiente, pero si está activada (en alto) permitirá la solicitud de Interrupción.
- Además existe una máscara de interrupción global **GIE (INTCON<7>)**, la cual bloqueará todas las solicitudes de interrupción si está desactivada (**GIE**=0).
- Algunas fuentes de interrupción también poseen una segunda máscara de interrupción global denominada **PEIE (INTCON<6>)**. De hecho, actúa sobre todas las fuentes de interrupción, excepto las interrupciones debidas al pin INT, el desbordamiento del TIMER0 y las interrupciones del puerto B (INTF, T0IF y RBIF).

De acuerdo a lo anterior, la única manera en que una solicitud de interrupción provoca en efecto una interrupción en el programa es cuando:

- La máscara global está activada (GIE=1).
- (En su caso) la máscara global de periféricos está activada (PEIE=1)

- La máscara local está activada
- Ocurre un evento que activa la bandera correspondiente.

La lógica de activación de máscaras y banderas descrita arriba puede entenderse en términos del diagrama lógico mostrado en la figura 3.9. En este diagrama se muestran las 14 fuentes de interrupción del PIC16F87x y se usan los nombres específicos de cada fuente de interrupción para sus respectivas banderas y máscaras de interrupción.

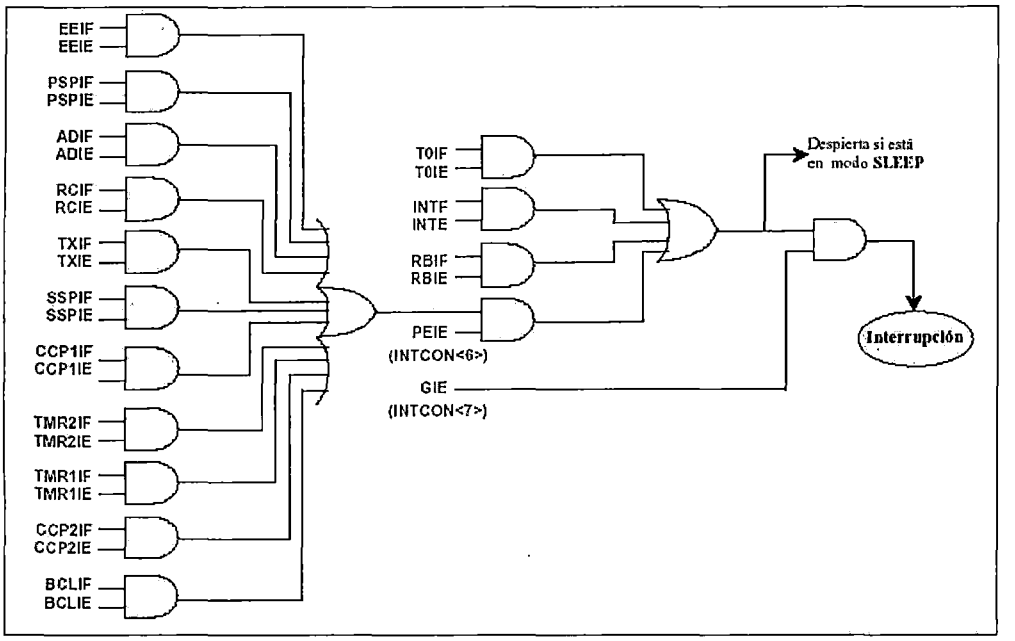



***Figura 3.9.-*** *Diagrama lógico de las interrupciones en el microcontrolador*

## 3.2. DESCRIPCIÓN GENERAL DEL SISTEMA

No se pretende desarrollar todo el sistema para el *Ministerio De Transporte* y PROVIAS NACIONAL, sino sólo para las zonas alejadas ya mencionadas,

para luego incluirlo en su sistema. Por esta razón el sistema general desarrollado será el siguiente:

### 3.2.1. Especificaciones técnicas generales

Detallaremos las especificaciones técnicas por niveles.

**A. El nivel de sensado**

Refiere a que los vehículos deberán ser registrados por dispositivos que permitan registrar su paso por la Unidad de Peaje en el carril de cobro. Es importante precisar que este registro deberá de mantenerse funcionando aún cuando la energía eléctrica del peaje sea nula; porque la alimentación de estos componentes se requiere que sea con energía continua, vale decir mediante baterías. El registro consta de clasificar y contar los vehículos, registrando la fecha, hora, ejes.

**B. El nivel de cobro**

Refiere a la emisión de tickets (comprobante de pago) para la cuál debería ser necesario contar con una computadora e impresora para el registro del cobro e impresión del ticket respectivamente, sin embargo esto se hace manualmente, es decir, mediante unos tickets que se llevan a cada peaje alejado de estas zonas.

**C. El nivel de administración**

Refiere a que las transacciones de sensado y de cobro son centralizados en un servidor de datos para un control y gestión.

**D. El Nivel de Gestión Central**

Refiere al acopio de información del peaje en un sistema existente central que consolidad la información de todas las Unidades de Peaje, pudiendo estar interconectada a través de cualquier medio tal como, línea dedicada, satélite; en el caso que no exista el medio se hará a través de archivos digitales par su integración. Aquí se ha propuesta una forma muy económica de interconexión que se tratará con mayor detalle más adelante.

### 3.2.2. Especificaciones técnicas detalladas

Detallaremos más sobre todo lo mencionado anteriormente.

**A. El nivel de sensado**

El criterio de este nivel corresponde al registro de vehículos usando equipos y componentes de que trabajen con energía continua. La actividad en las Unidades de Peaje es las 24 horas, dividido en 2 turnos de 12 horas, al término de cada turno se requiere realizar una liquidación por cambio de personal. Para la liquidación de cada turno es requisito indispensable el registro de vehículos por eventos; actividad que la realiza el administrador de turno con el cobrador, el sistema debe de estar con la disponibilidad de extraer los eventos a una computadora, sea en detalle o resumen (reporte

simple) para liquidación referida indicando fecha, hora, ejes, número de vehículos, clasificado por ejes. El sistema generará información para una liquidación de turno eficiente y eficaz en este nivel.

Se requiere componentes con alimentación continua y que permitan extraer los datos para la liquidación por turno.

Se requiere la detección de ejes, doble rueda (para diferenciar ligeros de pesados) y sentido de tráfico, el sistema debe de estar en la capacidad de detectar los vehículos que pasan en sentido contrario y emitir el reporte respectivo.

Para esto se realizará el diseño de una tarjeta basada en microcontroladores MICROCHIP de la familia 16F87X.

**Vías de Sensado**

Las vías de sensado y cobro que se implementará será solamente 1, en el que se podrá clasificar los vehículos, es decir, el sistema podrá diferenciar el tipo de vehículo que pase, sea auto o camión, así como la cantidad de ejes que tiene. En la figura 3.10 se muestra la distribución de los sensores en la pista, el cual tiene una disposición que permitirá la clasificación de vehículos.

**Componentes mínimos requeridos:**

- Sensores de presencia o inductivos (loop, antena).- este sensor detectará la presencia de un vehículo que ha pasado por la vía.
- Sensores de ejes simple y doble rueda.- estos sensores detectarán cuántos ejes tenía el vehículo que pasó y además se diferenciará y clasificará si fue un auto o un camión. Para esto se diseñará una tarjeta.
- El almacenamiento de los eventos.- Para esto se usarán una tarjeta basada en microcontroladores y memorias EEPROM para el almacenamiento.
- Programa en computador para la extracción de los eventos y generación de reportes.

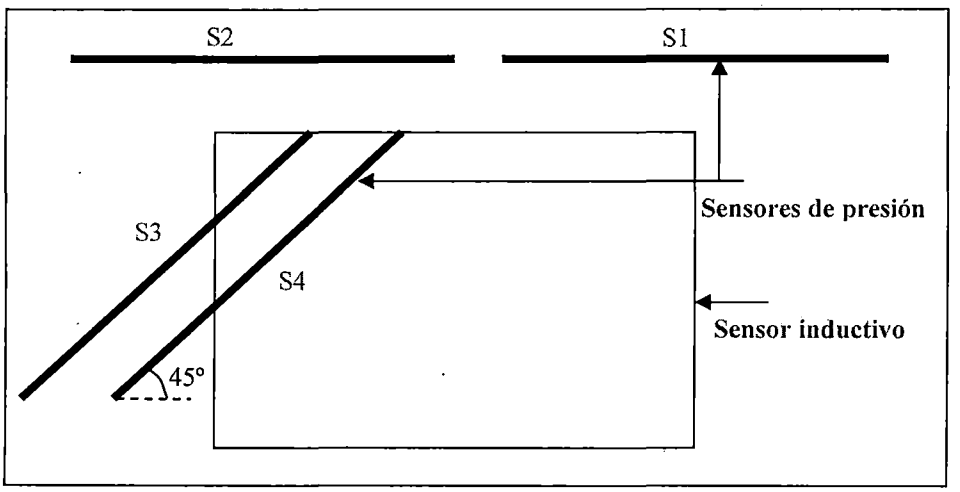



***Figura 3.10.-*** *Disposición de los sensores de presión e inductivo en la loza*

En la figura 3.11 se puede apreciar como por la disposición de los sensores se podrá determinar la clasificación de los vehículos, es decir si el que pasó fue un auto o un camión, estos se dispondrán en la pista unos metros después de la isla de la caseta de cobranza.

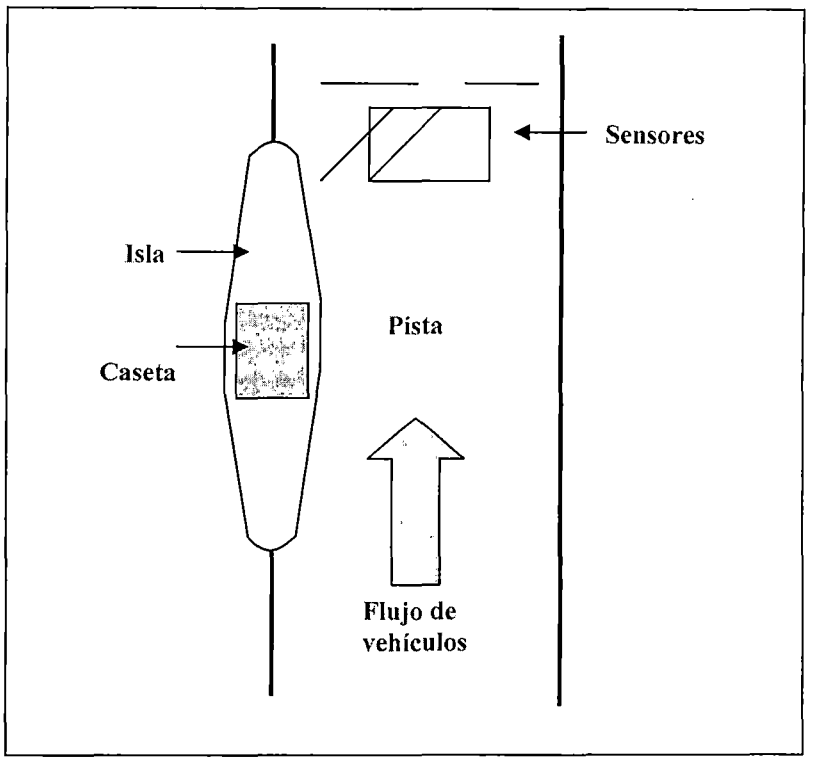



***Figura 3.11.-*** *Paso de vehículos en la unidad de peajes*

**B. El nivel de cobro**

El criterio de este nivel corresponde a la emisión de tickets que se hacen y harán manualmente de acuerdo a la categoría y clasificación. El ingreso o anulación de registros de tarifa diferenciada lo hará la administración superior.

**C. El nivel de administración**

El criterio de este nivel corresponde a la concentración de datos del nivel de sensado, contando así con una herramienta administrativa para la gestión de control de la Unidad de Peaje.

En la oficina de administración deberá de contar con una computadora, que servirá de control y gestión al administrador de peaje para la liquidación de turno, liquidación del día, reporte de detección de flujo, recaudación de discrepancias, reporte clasificado, vale decir acceso al sistema de gestión de peaje, que incluye la programación de turnos, cobradores, etc. Adicionalmente la administración de altas y bajas de registros de vehículos autorizados para circular (cobro de tarifa con descuento) así como las autorizaciones para el cobro de la tarifa diferenciada, cuya vigencia es de un año.

Se requiere equipamiento que centralice y almacene los datos registrados diariamente en los niveles de sensado y cobranza; componentes que forman la actividad de cobranza en la Unidad de Peajes y que permitan la interacción con el centro de gestión, en el ingreso o en la salida de información.

Es impresionante que la actividad de gestión de cobranza no sea dependiente del servidor; deberá ser independiente del funcionamiento del servidor o del sistema de comunicaciones. Cuando se reestablezca la comunicación o la operatividad del servidor los datos deberán ser actualizados al servidor para mantener los datos históricos de las actividades del peaje el registro del flujo vehicular, es decir, los eventos registrados en la tarjeta de control principal.

En la figura 3.12 se presenta un cuadro en el que se almacenará los datos de cada vehículo que pasó, incluyendo el año, mes, día, hora, minuto y segundo del momento en del sensado, como el tipo de vehículo, es decir, si fue un auto o camión y de cuantos ejes, además de un número correlativo al tipo de vehículo que paso, cada tipo tendrá una numeración diferente. Esta información podrá generarse para una hora específica así como para un turno completo.

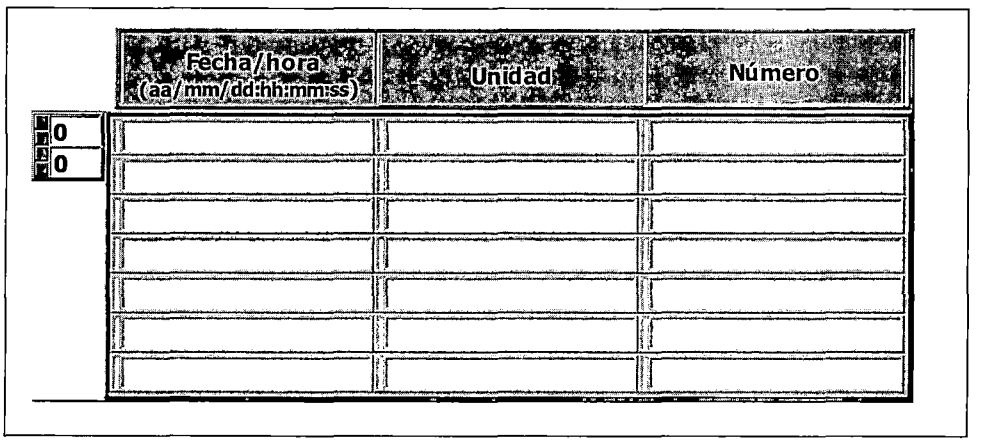



***Figura 3.12.-*** *Tabla de datos descargados por eventos*

En la figura 3.13 se puede ver cómo también se generará un reporte por hora y turno, totalizando y clasificando la información, así como la cantidad de veces que se encuentra apagado el equipo por hora.

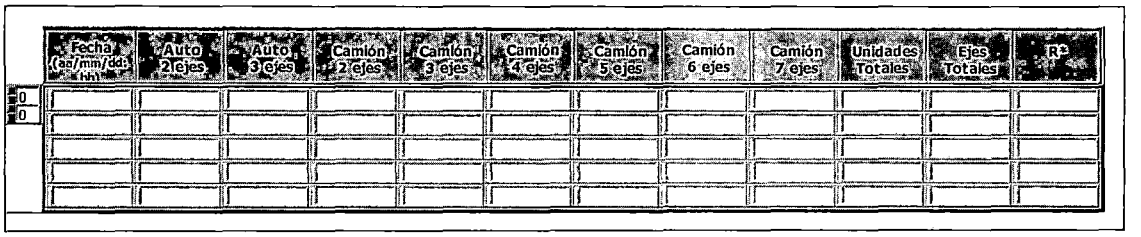



***Figura 3.13.-*** *Tabla de datos descargados clasificado por hora y turno*

## D. El nivel de gestión Central

La unidad de Peaje contará con una identificación única que permitirá incorporar a la Base de Datos dentro de la estructura general de todos los peajes para el acceso y gestión de la información, generada en cada Unidad de Peaje.

La especificación de la estructura de datos será proporcionada por el departamento de Informática de Provías Nacional. La base de datos de todos los peajes que generan información está ubicada en la sede central. Como se indicó anteriormente aquí se piensa interconectar los equipos de control directamente a un servidor principal que se encuentre en la central en Lima, esto se puede apreciar en la figura 3.14.

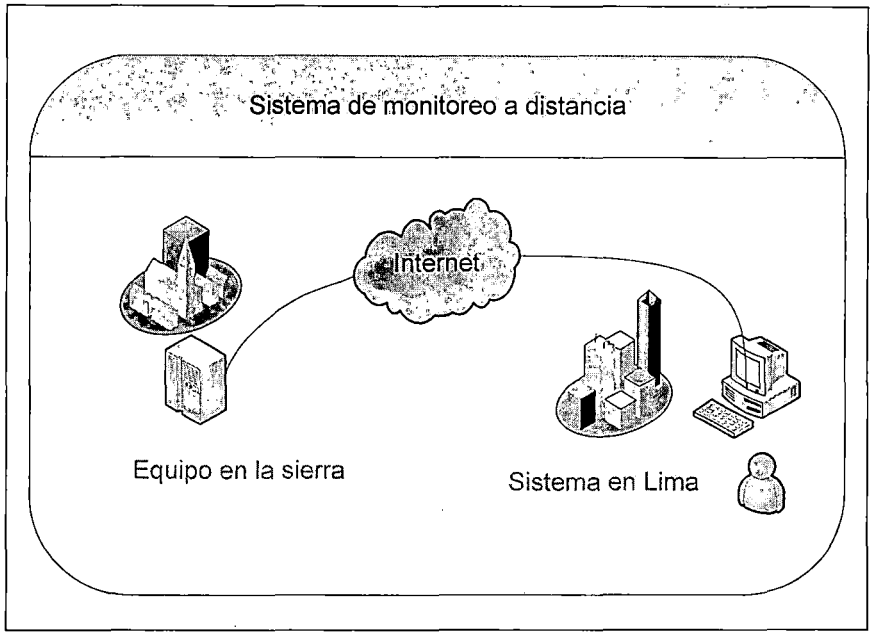



***Figura 3.14.-*** *Esquema de propuesta de sistema de monitoreo a distancia*

## E. Especificaciones de alimentación eléctrica

- **El nivel de sensado**

Requiere de alimentación de energía continua, vale decir un banco de baterías para el sensado y registro de eventos.

- **Tablero eléctrico**

Se requiere de un tablero eléctrico de distribución para alimentar los equipos que se alimentan de energía alterna de manera exclusiva del Sistema de Peaje, el esquema deberá de configurarse en circuitos separados con una línea de alimentación independiente con línea a tierra, caja metálica con tapa y cerradura de seguridad, con interruptores magnéticos dimensionados a la carga respectiva.

- **Sistema de Puesta a Tierra**

La puesta a tierra será de acuerdo al requerimiento de resistividad del suelo del lugar y de la necesidad del Sistema de Peaje y se debe garantizar una resistividad menor de 5 ohmios.

## F. Especificaciones del software

Todo el software será desarrollado en el lenguaje de programación LABVIEW de NATIONAL INSTRUMENT, el cual consistirá:

- Programa de administración de las herramientas.
- Programa de configuración de las tarjetas de control.
- Programa de generación de reportes.
- Programa de descarga de información.
- Programa de supervisión en tiempo real.
- Programa de pruebas de estado de sensores.
- Programa de supervisión a distancia

## G. Especificaciones de obra civil

El proyecto no consiste en implementar un peaje, sino implementar su sistema de control, por lo que el local de las oficinas, las computadoras, la isla, y la caseta ya se encuentran en cada unidad de peaje; por lo tanto, la única obra civil que se hará será el rompimiento de la pista para poner los sensores y la loza de concreto que tendrá un resistencia de 280kgf/cm$^2$.

## 3.3. DISEÑO ELÉCTRICO

Se mostrarán los cálculos justificativos para el diseño del sensor de bobina y del tablero eléctrico.

### 3.3.1. Diseño de los sensores de bobina

Para el diseño de este sensor, se tuvo que recurrir a hojas técnicas y recomendaciones de la empresa **RENO A&E** que diseña este tipo de sensores y equipo para su calibración (véase anexo A). Este sensor servirá para detectar la presencia de un vehículo. A este sensor también lo llamaremos sensor de antena, sensor loop o sensor inductivo.

**¿Cómo se detecta el vehículo?**

Cuando un vehículo entra en un lazo inductivo (bobina), la masa metálica proporciona un mejor camino conductor para el campo magnético. Esto produce un efecto de carga, que hace que la inductancia del lazo varíe. La variación de la inductancia hace que la frecuencia resonante aumente de su valor nominal. Si el cambio de frecuencia excede el umbral puesto por el ajuste de sensibilidad, el módulo recibe una señal como consecuencia de ello.

Colocando un cable alrededor del perímetro del sensor y cortocircuitando sus terminales, puede producir un cambio en la inductancia, por esa razón se debe calibrar la sensibilidad del circuito de una manera adecuada, teniendo en cuenta estos efectos.

**¿Cuál es el mínimo valor de inductancia aceptable del loop?**

El sensor loop se sintonizará a valores de inductancia que van desde 20 a 1000 uH. Es preferible que la combinación del lazo y la masa metálica tengan un mínimo de aproximadamente 50 uH para la estabilidad. Por regla general, el lazo inductivo debe ser igual o superior al de la masa metálica en inductancia.

**¿Cuántas vueltas de cable deberían ser instaladas en el lazo?**

El número de vueltas necesario en el loop depende de su tamaño. La inductancia del loop se puede calcular de la siguiente manera:

$$\boxed{L = P \times \frac{(t^2 + t)}{4}} \ldots\ldots\ldots (3.1)$$

Donde:

L= Inductancia (uH)

P= perímetro (pies)

T= Número de vueltas.

Esta fórmula puede ser simplificada a $L = P \times K$ para sustituir la constante K por $(t^2 + t)/4$.

Las dimensiones que tendrá el lazo inductivo han sido elegidas en una serie de pruebas, tomando en cuenta el tiempo en que se demoraría en pasar el vehículo, las dimensiones del carril y la probabilidad de cuál sea el lugar por donde pase la mayor masa metálica. Estas pruebas fueron tomadas en pleno

proceso después de la observación de inconvenientes que se iban presentando y se llegó a la más adecuada, cuyas dimensiones será de 2.5m x 1.5m. Para estas medidas los cálculos de la inductancia se muestran en la tabla 3.5. Esto significa que tendremos una inductancia de unos 39.8uH aproximadamente. Este valor será tomado en cuenta para el diseño del circuito detector de vehículo, que se regulará con bobina variable cuyo valor depende de esta inductancia. Aquí se verá la posibilidad de regulación y el valor de la bobina variable.

***Tabla 3.5***

*(Resumen de cálculos para el sensor de antena para diferentes números de vueltas y dimensiones)*

| *Perímetro* | | | *Número de vueltas* | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **metros** | **pies** | **1** | **2** | **3** | **4** | **5** | **6** | 7 |
| | 3.048 | 10 | 5 | 15 | 30 | 50 | 75 | 105 | 140 |
| | 6.096 | 20 | 10 | 30 | 60 | 100 | 150 | 210 | 280 |
| | 8 | 26.246719 | 13.12336 | 39.370079 | 78.740157 | 131.2336 | 196.85039 | 275.59055 | 367.45407 |
| | 12.192 | 40 | 20 | 60 | 120 | 200 | 300 | 420 | 560 |
| | 15.24 | 50 | 25 | 75 | 150 | 250 | 375 | 525 | 700 |
| | 18.288 | 60 | 30 | 90 | 180 | 300 | 450 | 630 | 840 |
| | 21.336 | 70 | 35 | 105 | 210 | 350 | 525 | 735 | 980 |
| | 24.384 | 80 | 40 | 120 | 240 | 400 | 600 | 840 | 1120 |
| | 27.432 | 90 | 45 | 135 | 270 | 450 | 675 | 945 | 1260 |
| | 30.48 | 100 | 50 | 150 | 300 | 500 | 750 | 1050 | 1400 |

### 3.3.2. Tablero eléctrico

En el tablero eléctrico estarán la tarjeta del sensor inductivo, acondicionadores de señal y tarjetas de control. Además tendremos como respaldo de energía 2 baterías de 12V y 17Ah en serie, un cargador con control de corriente, una llave térmica AC de 10A y 220V para la alimentación principal, una llave térmica DC de 5A y 24V, un tomacorriente y su respectiva conexión a tierra. El tablero se colocará en un gabinete metálico y los cables pasarán por canaletas, además en la puerta se colocarán un indicador luminoso de 220V AC para saber si hay energía, un amperímetro y voltímetro analógico para registrar el consumo y carga de las baterías. Además se dispondrá la salida para el timbre de cambio de hora y una salida DB9 para la conexión al puerto serial de la PC. (Ver figura 3.15 y 3.16).

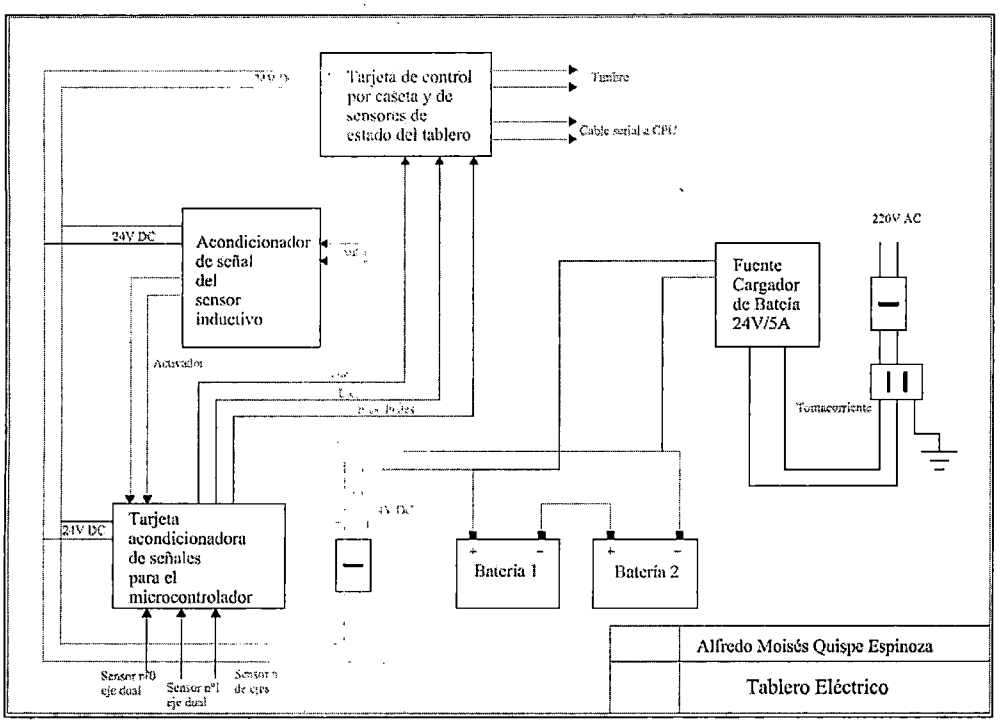



***Figura 3.15.-*** *Diagrama del tablero eléctrico*

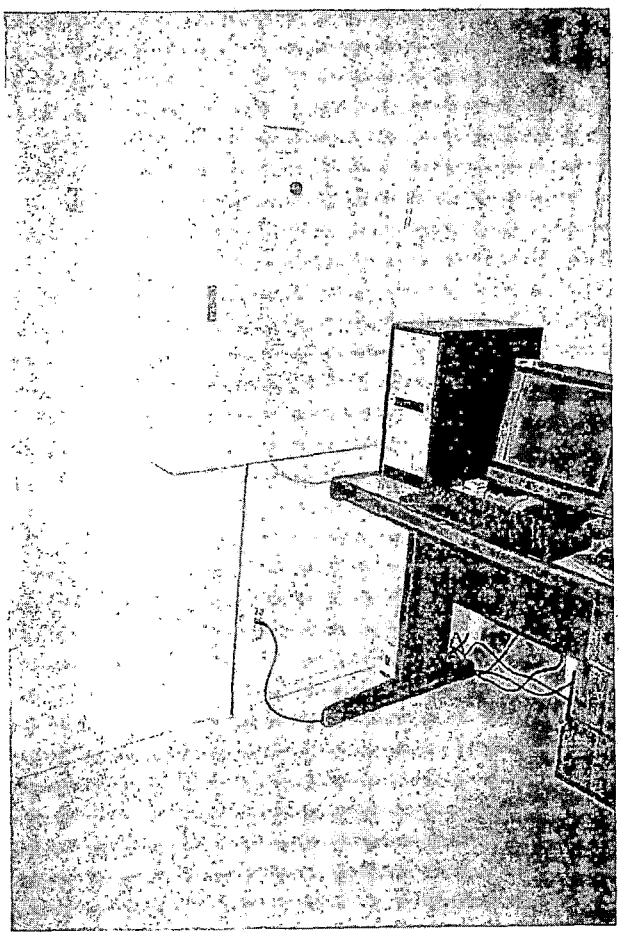


***Figura 3.16.-*** *Foto del equipo y tablero eléctrico en el peaje de Chalhuapuquio*

## 3.4. DISEÑO ELECTRÓNICO

En esta parte se presentarán los caculos justificativos para el diseño de los acondicionadores de señal de todos los sensores, así como la tarjeta de control.

El sistema de control básicamente se divide en 2 partes: una la hace el microcontrolador en la tarjeta de control y la otra es realizada por el software de control diseñado en la PC.

A continuación veremos en detalle todas las tarjetas involucradas en este sistema de control.

### 3.4.1. Tarjeta acondicionadora señal

Toda la etapa de acondicionamiento de señal se puede resumir en el esquema visto en la figura 3.17. Veremos paso a paso cada una de estas etapas.

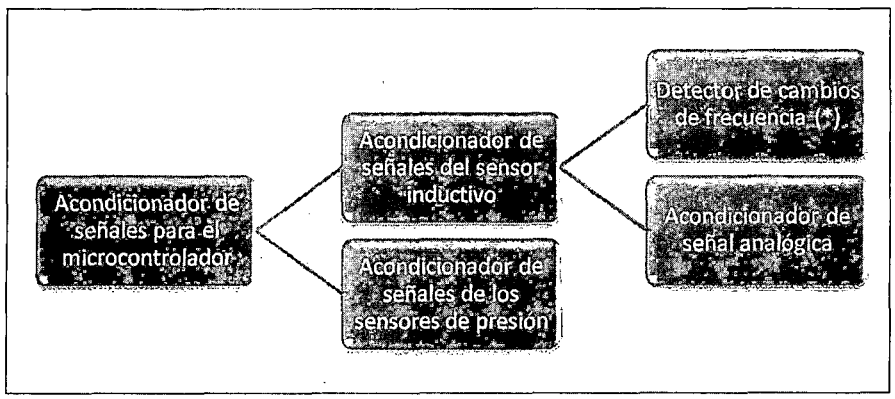



***Figura 3.17.-*** *Diagrama de la etapa de acondicionamiento de señales*

**A. Acondicionador de señal del Sensor inductivo de antena o loop**

Esta consta de 2 partes:

- Tarjeta detectora de cambios señal de frecuencia
- Tarjeta acondicionadora de señal analógica.

**a) Tarjeta detectora de cambios de señal de frecuencia**

Esta tarjeta genera una señal a una determinada frecuencia juntamente con el sensor inductivo.

El sensor de lazo inductivo será conectado como parte de un circuito oscilador el cual generará una señal de una determinada frecuencia. Este es un circuito oscilador, el cual generará una frecuencia de acuerdo a la inductancia L1. (Ver figura 3.18)

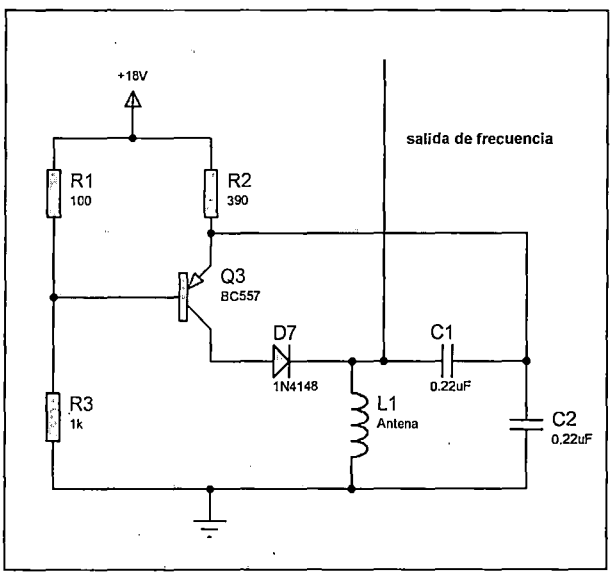



***Figura 3.18.-*** *Circuito oscilador para el sensor del lazo inductivo*

El inductor que se muestra (L1) es el sensor de lazo inductivo como parte de un circuito oscilador, cuya señal irá a otro circuito analógico.

Este circuito genera una frecuencia constante en función a la inductancia del sensor inductivo conectada a él, la cual se encuentra enterrada en el asfalto o concreto de la vía; esta inductancia sufre una variación al ingresar un material metálico dentro del área que cubre dicho lazo, que en este caso será un vehículo automotor. Esta frecuencia pasará por un circuito

detector de desfasaje (ver figura 3.19) la cual se traducirá en una señal de voltaje variable que será función de dicho valor de desfasaje.

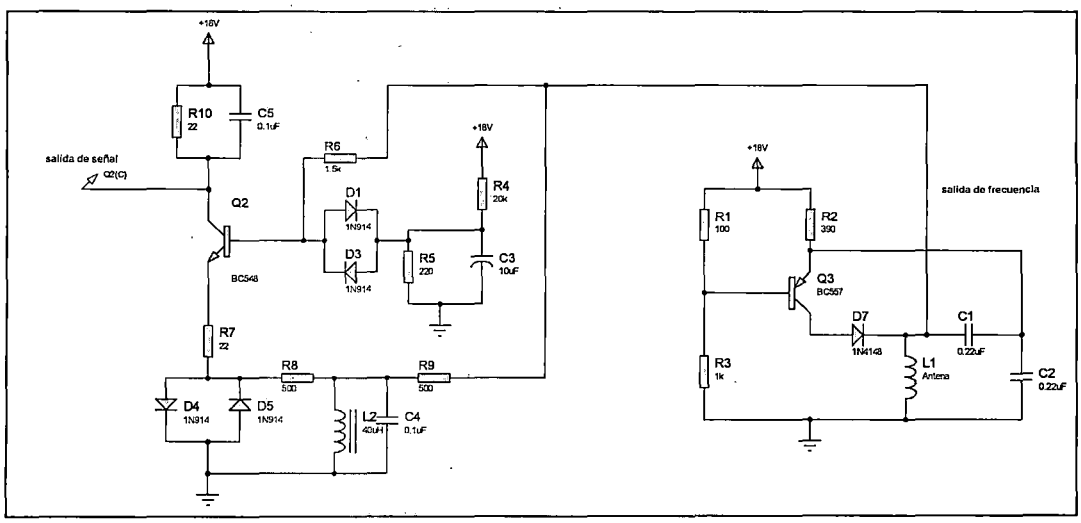



***Figura 3.19.-*** *Circuito detector del cambio de inductancia por el paso de un vehículo*

El sensor de bobina pasará por un circuito, con el cual se buscará un valor de inductancia de tal manera que a la salida de este (colector del transistor Q) se obtenga un voltaje constante alrededor de 5V a 8V. Se debe sintonizar el valor de la inductancia del inductor variable (L), de tal manera que cuando pase un vehículo por el lazo inductivo, se produzca un aumento del voltaje en colector del transistor Q. Tenemos que considerar que existe un valor de inductancia resonante que antes de este el voltaje puede subir, pero después de este el voltaje en lugar de subir disminuye. Esto se puede reflejar en el siguiente gráfico mostrado en la figura 3.20.

Del circuito mostrado en la figura 3.19 podemos obtener los resultados de la simulación, en CIRCUIT MAKER, con valores reales de la inductancia calculada por el sensor de lazo colocado en la pista. Estos resultados se aprecian en las figuras 3.21 a la 3.25.

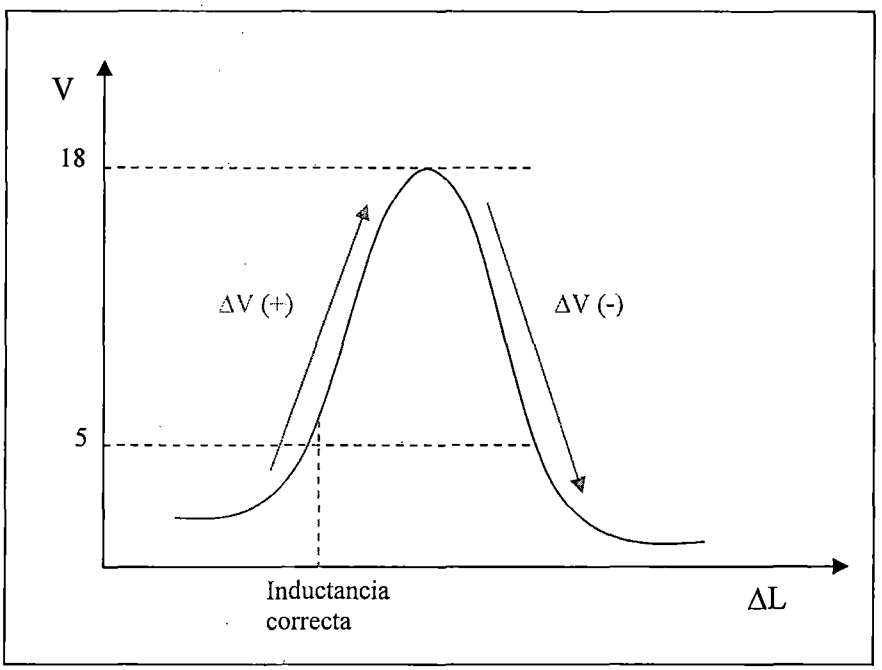



***Figura 3.20.-*** *Relación de voltaje con el cambio de inductancia*

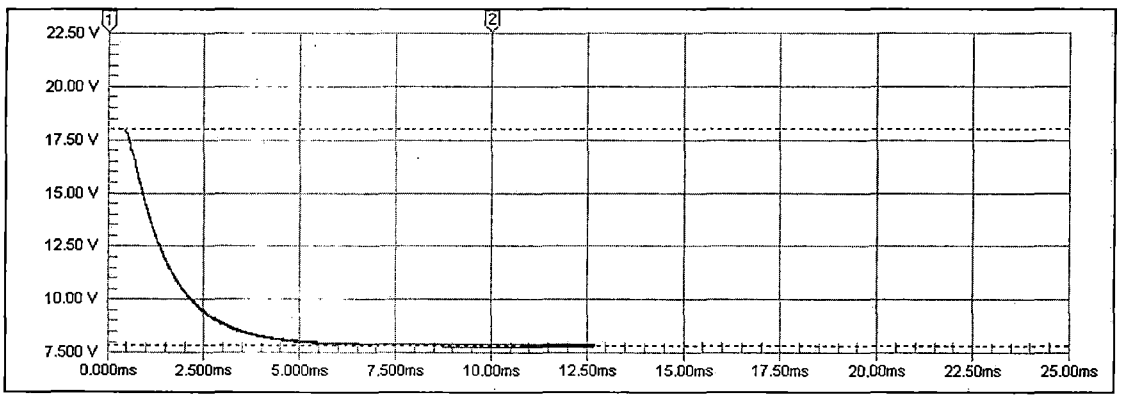



***Figura 3.21.-*** *Gráfico de simulación para inductancia de 42uH*

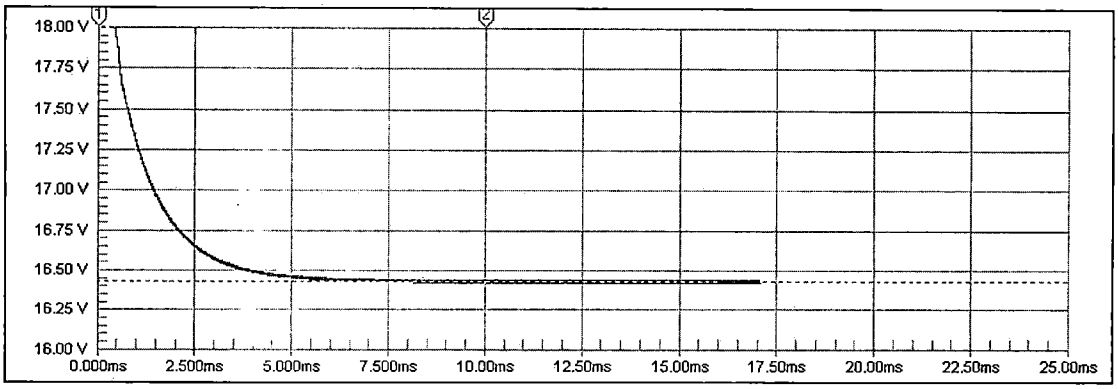



***Figura 3.22.-*** *Gráfico de simulación para inductancia de 48uH*

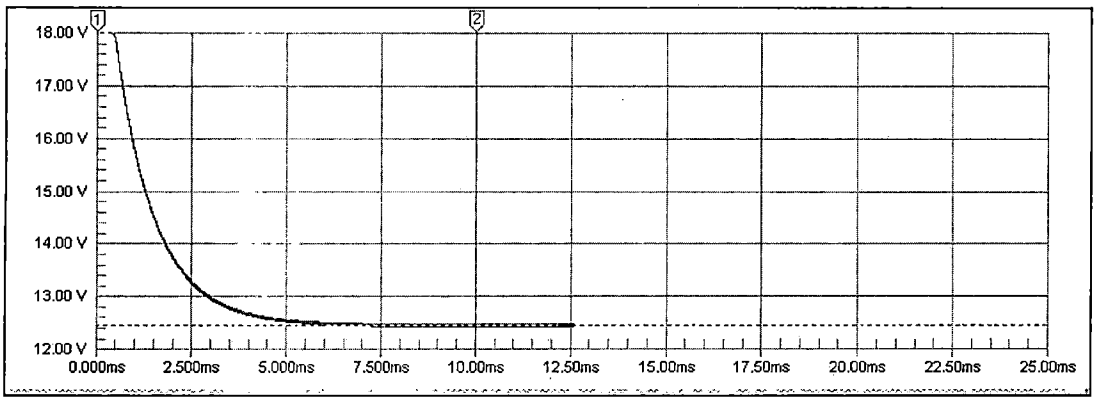



***Figura 3.23.-*** *Gráfico de simulación para inductancia de 43uH*

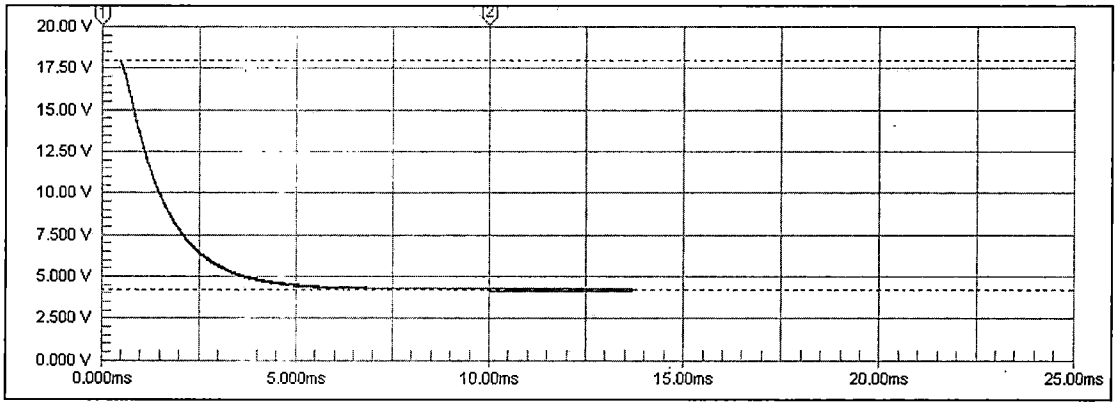



***Figura 3.24.-*** *Gráfico de simulación para inductancia de 41uH*

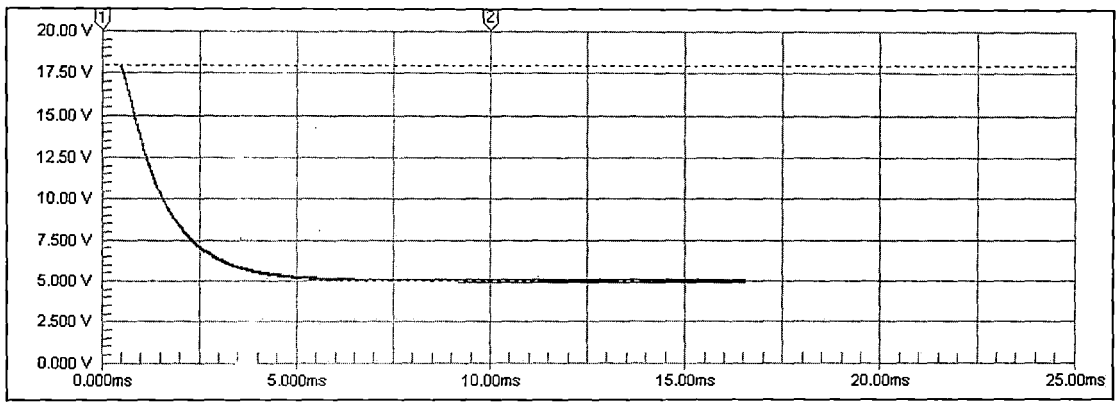



***Figura 3.25.-*** *Gráfico de simulación para inductancia de 42.7uH*

Como se puede observar, de lo analizado, el inductor variable (hecho con núcleo de ferrita) debe calibrarse muy lenta y delicadamente, ya que pequeños cambios de inductancia ocasionan cambios rápidos de voltaje de la señal de control. El valor de la bobina de inductancia variable estará entre **42 y 43uH.**

**b) Tarjeta acondicionadora de señal analógica.**

Para detectar el paso de un vehículo se diseñará un circuito que detecte el cambio de voltaje que envía el circuito detector de masa metálica con el lazo antena. Este voltaje se regula como se vio alrededor de más o menos 6V, por lo tanto se tendrá un circuito que detecte el cambio de este valor. Para esto pensemos primeramente en un circuito que envíe el mismo valor de la entrada a la salida, es decir de 6V, que servirá para compararla. Para esto la señal de salida de circuito del sensor de antena la dividiremos para la entrada a dos seguidores de voltaje, que ingrese a un circuito de comparación de tal manera que ante un cambio una señal (la de comparación) sea mucho más

lenta que la otra para que así el circuito se sature en más bajo valor. Estos seguidores de voltaje se colocan por seguridad y así tengamos alta impedancia y evitar que el voltaje disminuya a su paso, estos se diseñaran con amplificadores operacionales.

El circuito comparador también será diseñado a base de amplificadores operacionales. Tal que se tiene una primera propuesta, mostrada en la figura 3.26.

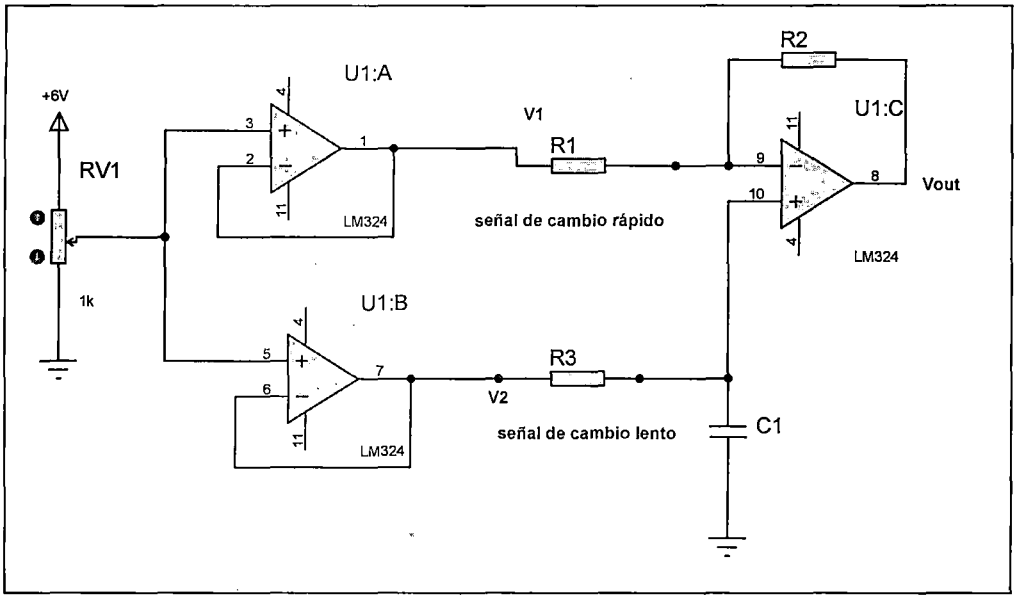



***Figura 3.26.-*** *Circuito básico para el acondicionamiento de señal*

En el circuito propuesto se observa en el operacional U1:C en la entrada positiva se tiene un condensador que después se elegirá su valor, de tal manera que cambie más lentamente que en la entrada positiva. De aquí aplicando superposición al circuito, tendremos un amplificador inversor y otro no inversor como se desarrollará a continuación. (Ver figura 3.27).

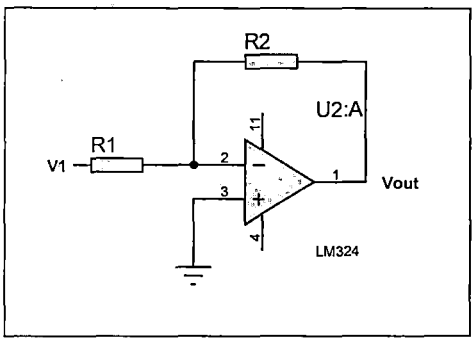



*Figura 3.27.- Amplificador inversor*

Trabajando con V1 y V2=0 tendremos un amplificador inversor

$$\frac{V1}{R1} = \frac{0 - Vout}{R2}$$

$$Vout = -\frac{R2}{R1} V1 \ldots\ldots\ldots (3.2)$$

Luego con V1=0 y trabajando con V2 tendremos un amplificador no inversor como lo muestra la figura 3.28.

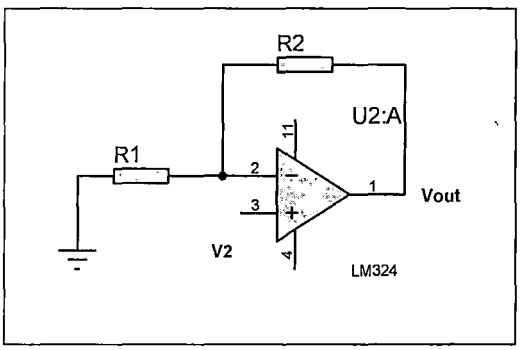



*Figura 3.28.- Amplificador no inversor*

$$\frac{V2}{R1} = \frac{Vout - V2}{R2}$$

$$Vout = (1 + \frac{R2}{R1}).V1 \quad \text{.....(3.3)}$$

Finalmente tendremos:

$$\boxed{Vout = (1 + \frac{R2}{R1}).V2 - \frac{R2}{R1}.V1} \quad \text{.....(3.4)}$$

De donde tenemos lo que esperamos. Si V1=V2=V entonces la Vout será igual al voltaje de las entradas, de acuerdo a lo mostrado en la expresión siguiente.

$$Vout = (1 + \frac{R2}{R1}).V - \frac{R2}{R1}.V = V$$

Y si la señal del sensor de loop antena cambia, entonces la señal V1 cambiará más rápidamente que la señal V2, variando rápidamente la señal a la salida Vout.

Sin embargo para que la señal V1 no crezca muy rápidamente, pondremos algunos condensadores, y para que la amplificación sea más suave y no sea muy brusca también podremos un condensador paralelo a R2, de aquí el circuito planteado se muestra en la figura 3.29.

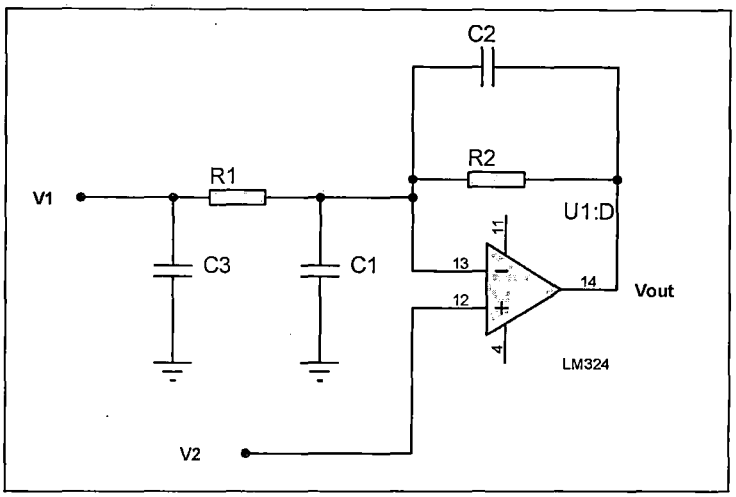



*Figura 3.29.- Amplificador completo*

Aplicaremos la transformada de Laplace para hallar la Función de transferencia del sistema a analizar y así conocer su comportamiento en estado transitorio y estado estable. Previamente aplicaremos el teorema de Thevenin para el circuito de la figura 3.29, específicamente el arreglo Π mostrado en la figura 3.30.

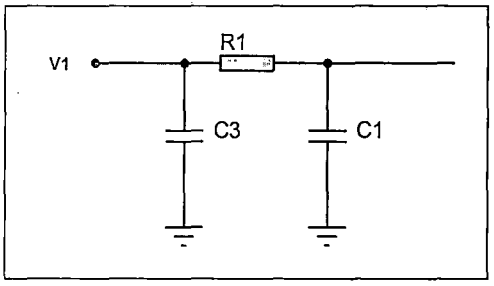



*Figura 3.30.- Circuito a aplicar Thevenin*

Se trabajará el circuito haciendo una transformada de Laplace para hallar la expresión de Vth y el Zth. Aquí trabajaremos con el siguiente circuito de impedancias de la figura 3.31.

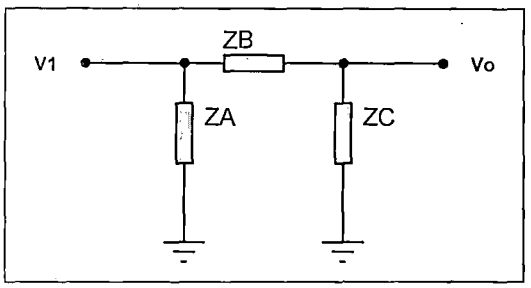



***Figura 3.31.-*** *Circuito a aplicar Thevenin, con impedancias*

$$ZA = \frac{1}{sC}$$
$$ZB = R1$$
$$ZC = \frac{1}{sC1}$$

De la figura 3.31 se tendrá que Vth=Vo, de aquí, por divisor de tensión, tendremos lo mostrado en la siguiente ecuación:

$$Vth = V1.\frac{\frac{1}{sC1}}{R1+\frac{1}{sC1}} = V1.\left(\frac{1}{s.R1.C1+1}\right) \ldots\ldots\ldots (3.5)$$

El Zth lo tendremos a partir de la figura 3.32.

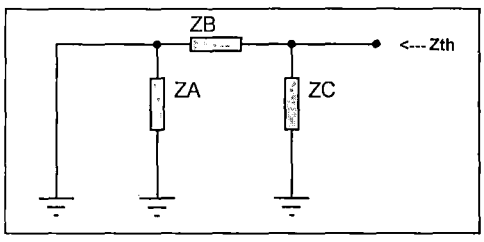



***Figura 3.32.-*** *Para el Zth de Thevenin*

$$Zth = \frac{R1.\frac{1}{sC1}}{R1+\frac{1}{sC1}} = \frac{R1}{s.R1.C1+1} \quad \text{.....(3.6)}$$

Llamaremos a Zth=Z1 y hallaremos la impedancia en Laplace de R2 y C2, luego:

$$Z2 = \frac{R2.\frac{1}{sC2}}{R2+\frac{1}{sC2}} = \frac{R2}{s.R2.C2+1} \quad \text{..... (3.7)}$$

Finalmente el circuito con impedancias, en Laplace, se puede apreciar en la figura 3.33.

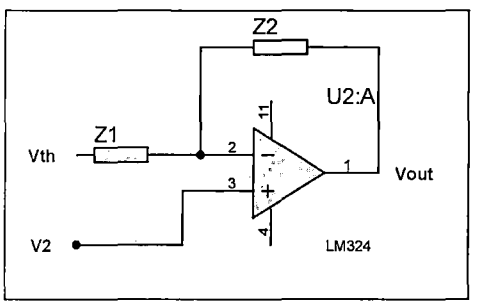



***Figura 3.33.-*** *Amplificador con impedancias con entrada del Vth de Thevenin*

Igualmente como se hizo con las resistencias, aplicaremos superposición y tendremos en la salida total como la suma de las señales de un amplificador no inversor e inversor, de aquí:

$$Vout = (1+\frac{Z2}{Z1}).V2 - \frac{Z2}{Z1}.Vth \quad \text{.....(3.8)}$$

De donde:

$$Z1 = Zth = \frac{R1}{s.R1.C1+1}$$

$$Z2 = \frac{R2}{s.R2.C2+1}$$

$$Vth = V1.\left(\frac{1}{s.R1.C1+1}\right)$$

Luego reemplazando tendremos:

$$\boxed{Vout = \left(1+\frac{R2}{R1}.(\frac{s.R1.C1+1}{s.R2.C2+1})\right).V2 - \frac{R2}{R1}.\left(\frac{1}{s.R2.C2.+1}\right).V1} \quad \text{.....(3.9)}$$

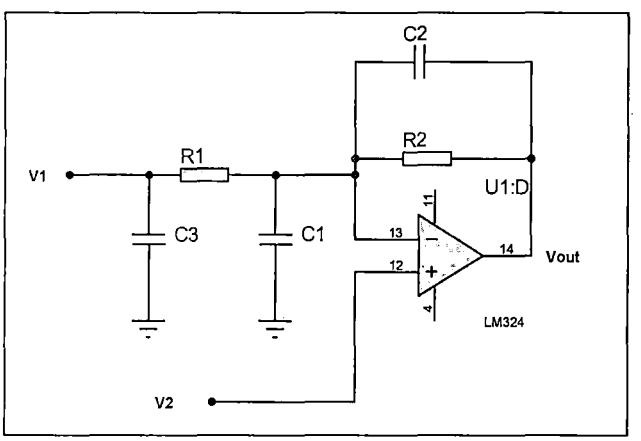



***Figura 3.34.-*** *Circuito amplificador diferenciador con filtros*

Del circuito que debemos recordar, el cual se muestra en la figura 3.34, es un circuito de dos entradas y una salida, es decir por superposición tendremos 2 funciones de transferencia mostradas en las ecuaciones 3.10 y 3.11, además los diagramas de bloques de las funciones de transferencia se muestran en la figura 3.35

$$\frac{Vout}{V1} = \frac{R2}{R1}.\left(\frac{1}{s.R2.C2.+1}\right) = G_1 \quad para\ V2 = 0 \ldots\ldots\ldots\ldots\ldots (3.10)$$

$$\frac{Vout}{V2} = \left(1+\frac{R2}{R1}.(\frac{s.R1.C1+1}{s.R2.C2+1})\right) = G_2 \quad para\ V1 = 0 \ldots\ldots\ldots\ldots (3.11)$$

Para la respuesta en estado estable para entrada escalón tendremos para s=0 en teorema de valor final:

$$Vss = \lim_{S->0} s.Vout$$

$$Vout = (1+\frac{R2}{R1}).V2 - \frac{R2}{R1}.V1$$

Por lo tanto con la consideración de que las entradas sean iguales, tendremos en el estado estable:

Vout=V1=V2=V

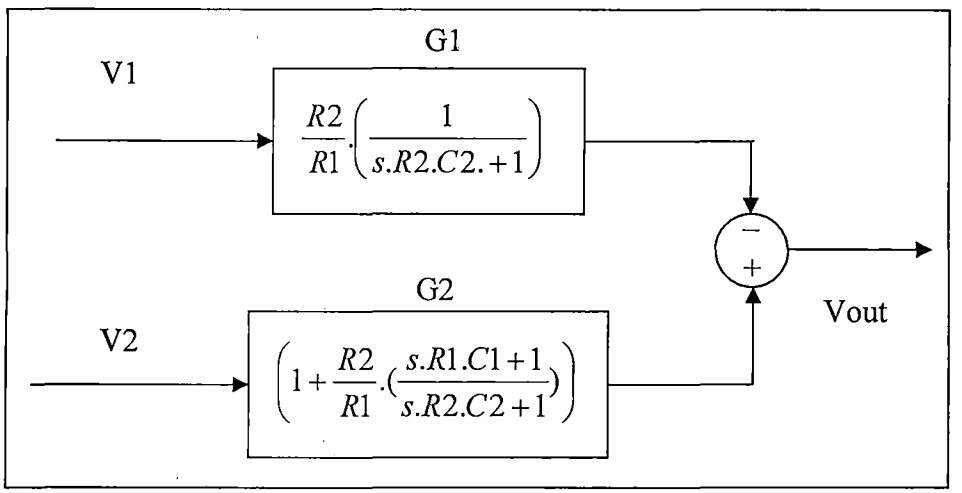



***Figura 3.35.-*** *Diagrama de bloques en Laplace del amplificador restador*

Primeramente trabajaremos con la función de transferencia G1, mostrada en la figura 3.36.

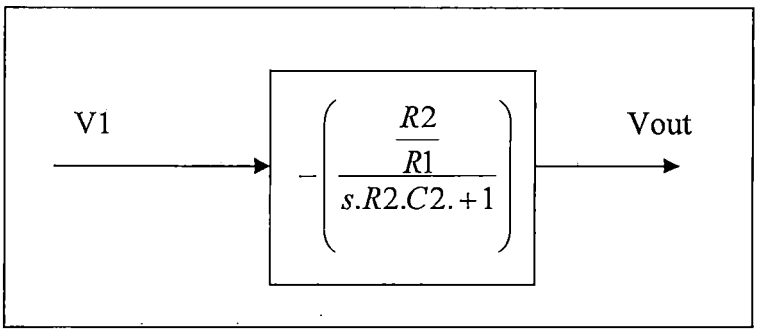



***Figura 3.36.-*** *Diagrama de bloques en Laplace para la entrada de voltaje V1*

Es un sistema de 1° orden con polo en $-\frac{1}{R2.C2}$. De aquí el tiempo de establecimiento será $tss = 5\tau = 5.R2.C2$. Ahora trabajaremos con la función de transferencia G2, la cual se puede expresar como lo muestra la figura 3.37. Podemos apreciar que su respuesta temporal depende prácticamente del bloque superior de la figura mencionada.

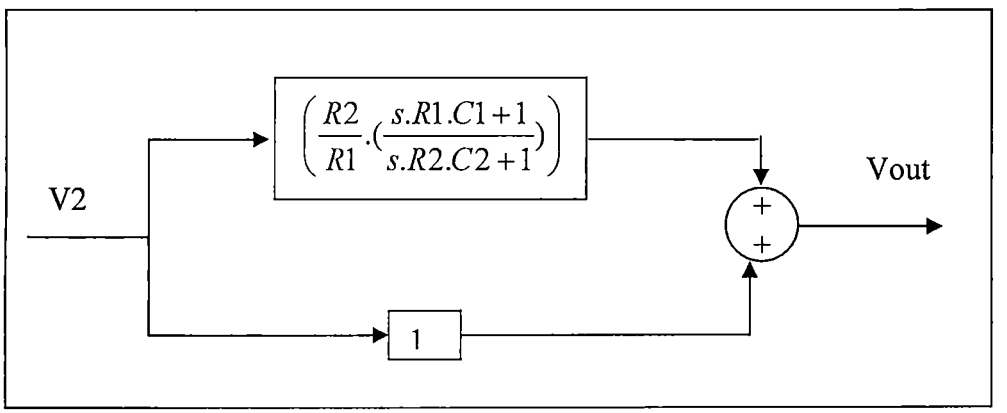



***Figura 3.37.-*** *Diagrama de bloques en Laplace para la entrada de voltaje V2*

De aquí el tiempo de establecimiento lo determina el polo $-\frac{1}{R2.C2}$

El tiempo de establecimiento será $tss = 5\tau = 5.R2.C2$

Pero en este caso tenemos adicionalmente un cero en $-\frac{1}{R1.C1}$

En el plano "S", podemos observar que el sistema es estable (ver figura 3.38); sin embargo buscaremos que el polo sea el dominante y no el cero, ya que el cero produce sobreimpulso y esto no se desea.

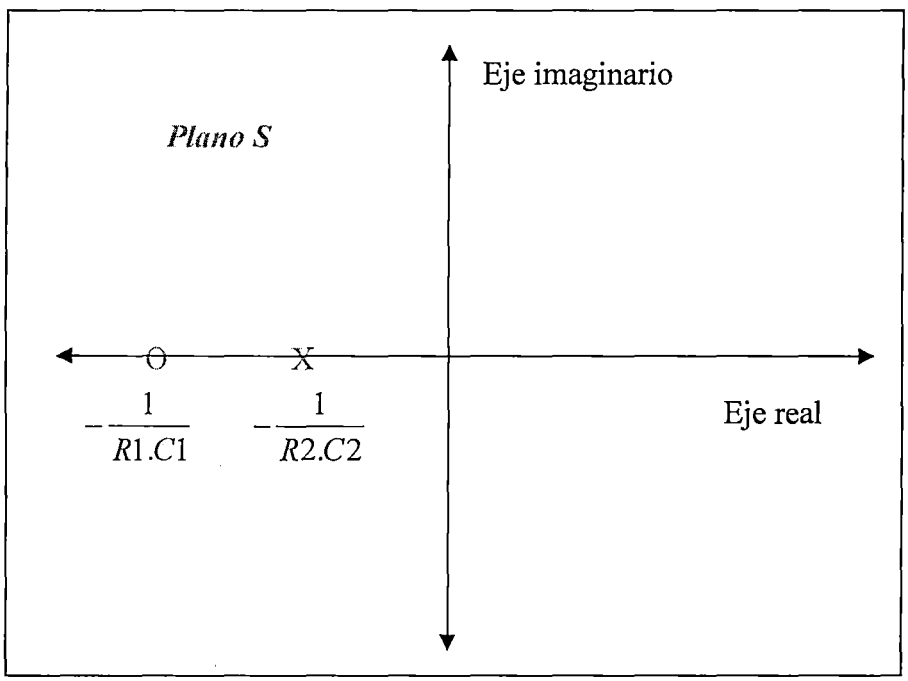



***Figura 3.38.-*** *Polo y cero del diagrama de bloques*

Para esto se escogerá la relación de 1 a 20, para que el polo sea dominante y el efecto del cero sea insignificante, de esa manera la respuesta corresponderá a un sistema de 1° orden sin sobreimpulso.

Luego los valores de resistencias y capacitares seleccionados serán:

R1=100k

C1=0.1uF

R2=2000k

C2=0.1uF

$$\frac{polo}{cero} = \frac{-\frac{1}{R2.C2}}{-\frac{1}{R1.C1}} = \frac{R1.C1}{R2.C2} = \frac{R1}{R2} = \frac{1}{20}$$

La respuesta depende del polo, siendo el efecto del cero despreciable con respecto al efecto del polo. Para probar lo analizado se uso la herramienta Control Design para LABVIEW, y apreciamos el resultado mostrado en la figura 3.39.

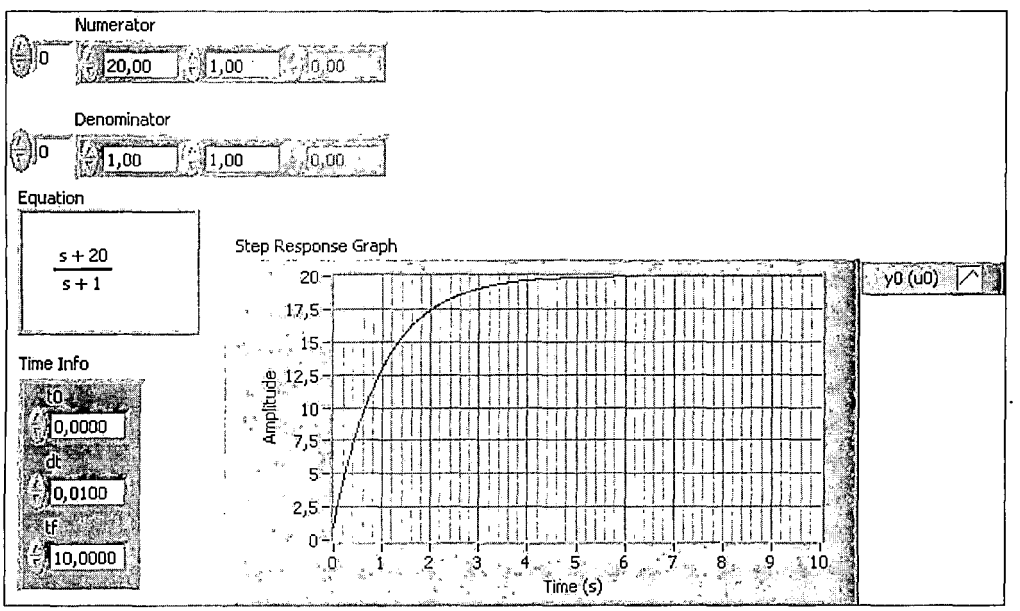



***Figura 3.39.-*** *Respuesta temporal del sistema de 1° orden*

Ahora veremos qué sucede cuando cambia V1 rápidamente, se puede apreciar que V2 cambia lentamente ya que es un voltaje producto de la carga del condensador 470uF, esto lo apreciamos en la figura 3.40.

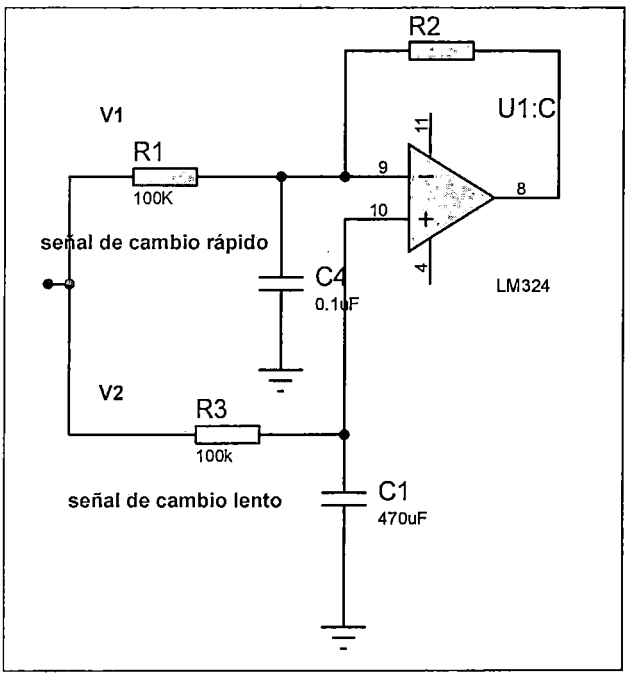



***Figura 3.40.-*** *Circuito de las señales de los voltajes V1 y V2*

$$\frac{t_{ss2}}{t_{ss1}} = \frac{5\tau_2}{5\tau_1} = \frac{100k.(470uF)}{100k.(0.1uF)}$$

$$\frac{t_{ss2}}{t_{ss1}} = 4700$$

$$t_{ss2} = 4700 t_{ss1}$$

$$\therefore t_{ss2} << t_{ss1}$$

Por esta razón al inicio, cuando se encienda el equipo y se calibre o sintonice el circuito del sensor loop antena, el voltaje V2 cambia muy

lentamente, pero se necesita que al inicio los voltajes sean iguales, entonces se pondrá un pulsador para obligar la carga rápida del condensador de 470uF.

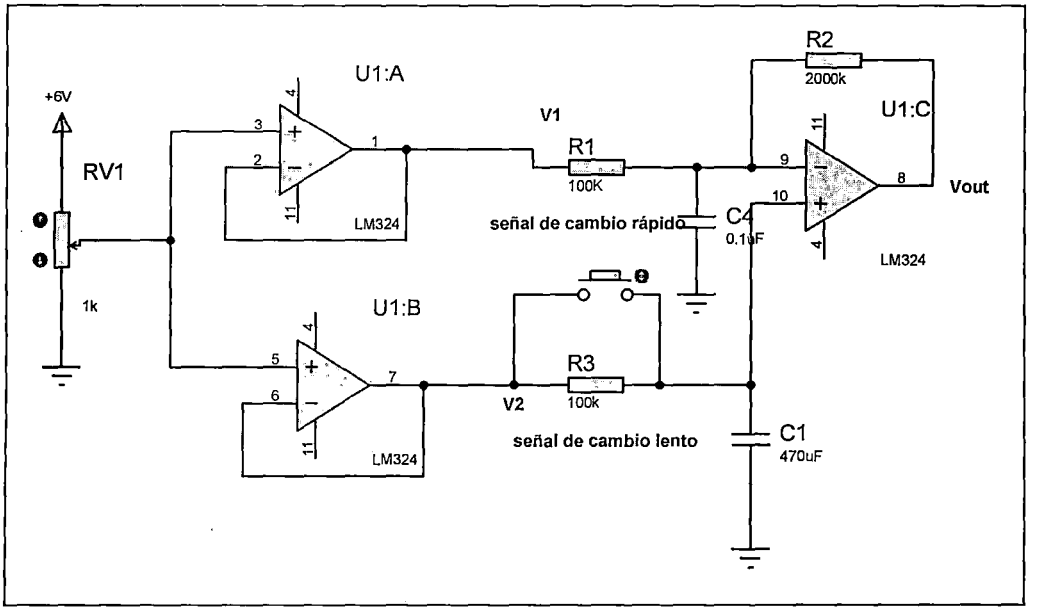



***Figura 3.41.***- *Circuito amplificador con seguidores de voltaje para cada señal V1 y V2*

Cuando haya un aumento de voltaje por la presencia de un vehículo, el voltaje V2 subirá más lentamente que V1, como se aprecia en la figura 3.41 Luego:

tss1=5x100x103x 0.1x10-6=50ms

tss2=4700.tss1=235s

Como un vehículo no demora más de 10s en pasar por el sensor loop, significa que sólo el 4.25% (10/235) del tiempo que demoraría el voltaje V2

en subir de 5V a 6V hasta 18V, entonces el voltaje subirá hasta un valor menor que Va durante el tiempo de paso de ese vehículo, de donde Va será:

Va=5+ (18-5) x0.0425=5.55v , si el voltaje inicial fuera 5V

Esto garantiza que, al cambiar el voltaje V1 (en la entrada negativa del operacional) más rápidamente que el voltaje V2 (en la entrada positiva), dará la impresión que en la entrada negativa es un escalón y en la entrada positiva es casi constante, siendo la diferencia de aproximadamente en un instante 18-5= 13V, y como la respuesta se estabiliza suave y rápidamente, se tendrá prácticamente un estado estable con salida, asumiendo que el voltaje inicial fue 5V igual a:

$$Vout = (1+\frac{R2}{R1}).V2 - \frac{R2}{R1}.V1$$
$$Vout = (1+\frac{2000k}{100k}).5 - \frac{2000k}{100k}.18$$
$$Vout = -255V$$

Como la alimentación de la tarjeta es de 18V y 0V (tierra), el voltaje a la salida se saturará en 0V cuando pase un vehículo, ya que no puede llegar a -255V.

La realimentación negativa de un amplificador tiende a mantenerlo dentro de la zona lineal y una realimentación positiva fuerza a ese amplificador a operar en la región de saturación. Como ejemplo de esto es el comparador, cuyo se muestra en la figura 3.42.

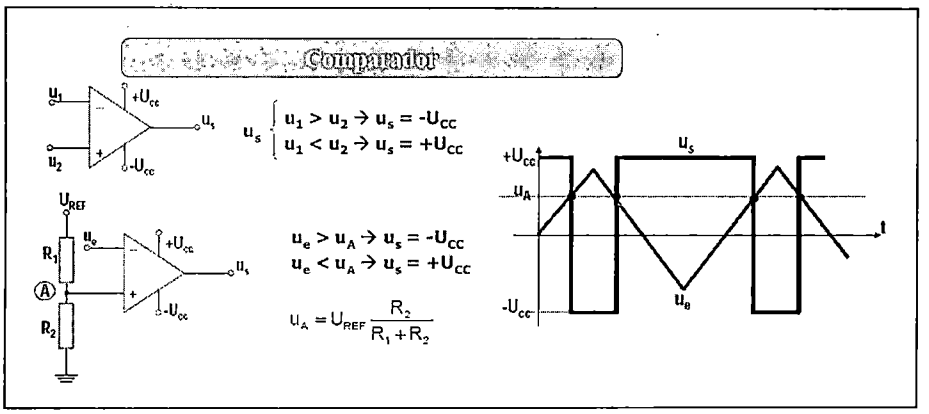



*Figura 3.42.- Comparadores*

**Inductancia variable al paso de un vehículo**

Cuando un vehículo pasa por el sensor loop antena, la inductancia cambia, pero no se mantiene uniforme, ya que un vehículo presenta ventanas, o alturas variables en el caso de los camiones, etc., esto da como resultado que la señal de voltaje que ingresa al circuito desarrollado anteriormente varíe en tanto pase el vehículo; esto podría ocasionar un mal conteo, es decir, una unidad más cada vez que pase esta situación. Para corregir este inconveniente utilizaremos un comparador Schmitt. Un comparador Schmitt es un comparador regenerativo con realimentación positiva, que presenta dos tensiones de comparación a la entrada +Va y Va, en función del estado de la salida. El diagrama de voltaje de salida versus voltaje de entrada de estos circuitos presenta histéresis, y por ello también se les denomina comparador con histéresis. Su principal importancia para el uso de este tipo de comparador será su capacidad de eliminar ruidos. Debemos recordar para esto las relaciones mostradas en la figura 3.43.

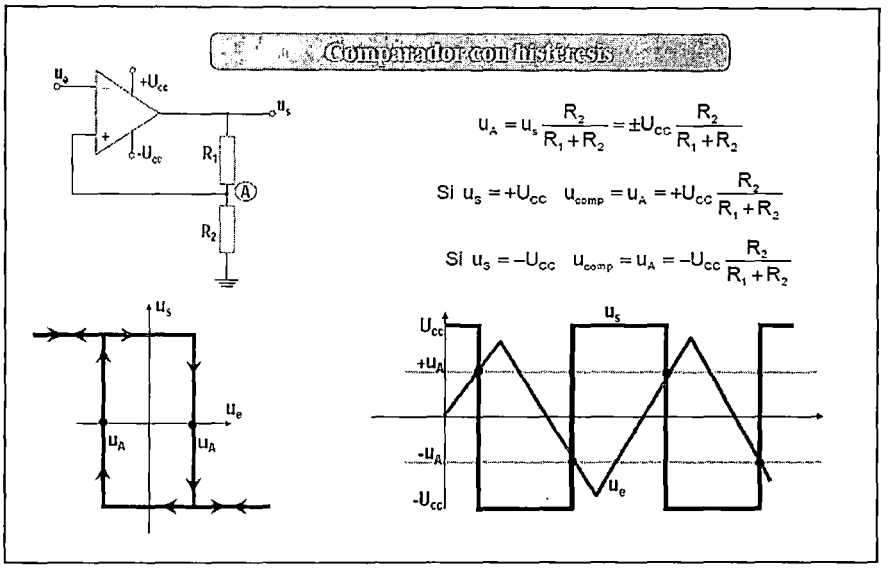



*Figura 3.43.- Comparador con histéresis*

Sin embargo para este circuito se utilizará el siguiente comparador con histéresis mostrado en la figura 3.44.

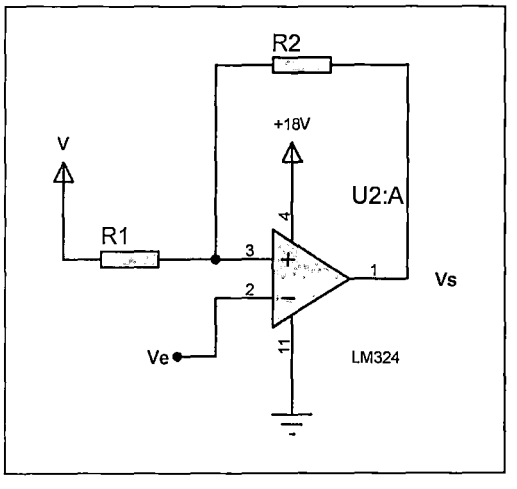



*Figura 3.44.- Circuito del comparador de histéresis a utilizar*

$$+Va = (18 - V).\frac{R1}{R1+R2}$$

$$-Va = (V - 0).\frac{R2}{R1+R2}$$

Luego debe cumplirse:

$$V.\frac{R2}{R1+R2} < (18-V).\frac{R1}{R1+R2}$$

V.R2<18.R1-V.R1

$$V < 18.\left(\frac{R1}{R1+R2}\right).....(\alpha)$$

También:

$$(18-V).\frac{R1}{R1+R2} < 5.....(\beta)$$

Probaremos para diferentes relaciones de resistencias:

- Para R1/(R1+R2)=1/2 tendremos 8<V<9
- Para R1/(R1+R2)=1/4 tendremos -2<V<4.5.

  Para V=3V tendremos:

  +Va=3.75V

  -Va=2.25V
- Para R1/(R1+R2)=1/5 tenemos -7<V<3.6

  Para V=2V tendremos:

  +Va=3.2

  -Va=1.6V

Para esto elegiremos la relación 1/5 y V=2V, con las siguientes resistencias:

R1=10k

R2=39k

$$\frac{R1}{R1+R2}=\frac{10}{49}$$

$$+Va=(18-2).\frac{10}{49}=3.265V$$

$$-Va=2.\frac{39}{49}=1.59V$$

El circuito final y su simulación en el software PROTEUS se pueden apreciar a continuación en las figuras 3.45 y 3.46.

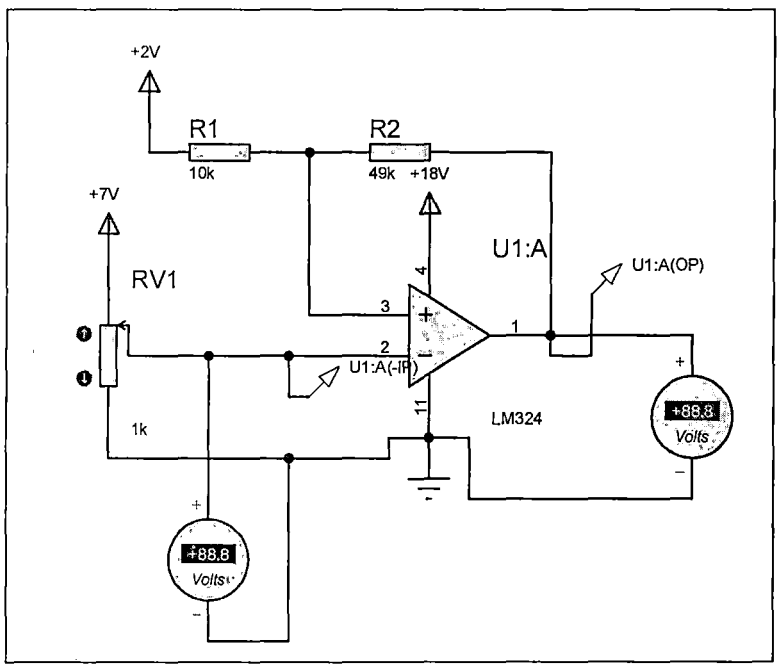



***Figura 3.45.-*** *Circuito comparador con parámetros calculados*

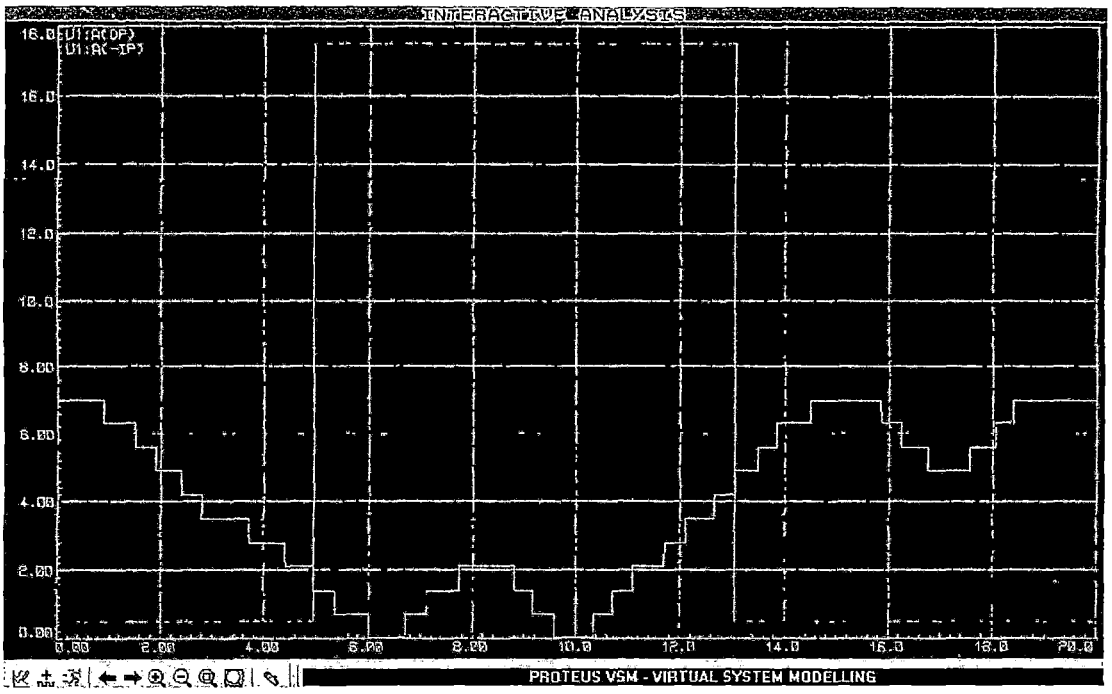



*Figura 3.46.- Simulación de la señal producida por un vehículo al pasar por el sensor inductivo. Señal analógica y digital*

Esto se acerca a los valores teóricos, cumpliendo satisfactoriamente con el propósito trazado.

**Compensador de temperatura**

Es imposible pensar que la inductancia se mantenga constante durante todo el día y el año, ya que en realidad éstos siempre sufrirán cambios físicos por dilatación térmica, compresión, etc. Para compensar este efecto, el voltaje de comparación de la señal del sensor de bobina no será constante, sino variable de acuerdo al circuito de la figura 3.41; pero además debemos recordar que éste cambia lentamente correspondiendo así a la velocidad de cambio de la temperatura, como ya lo hemos visto en el diseño anterior. Esto es importante ya que el valor nominal del sensor de bobina podría cambiar de

6V a más por efectos de la temperatura. Además para que la señal se "enganche" una vez que fue activada por un vehículo se añadió el circuito mostrado en la figura 3.47.

La señal con la que se comparará pasará por un opto triac (U2), el cual recibirá la señal de control de la señal invertida del bloque digital, esto se logrará con un transistor (Q5) trabajando en corte y saturación. Cuando no pasa un vehículo aquí la señal de control será un "1" lógico, activando el opto, el cual carga un condensador hasta 5V suavemente. Cuando pasa un vehículo la señal de control será "0" lógico, teniendo en cuenta que la señal en la entrada negativa del amplificador comparador cambia rápidamente, pero la entrada positiva cambia lentamente debido al condensador C10 que se carga por la resistencia R23. Una vez que el estado de la señal de control sea "0" el opto se abre, dejando la señal de comparación estable a un valor ligeramente encima de 5V. Este mismo efecto ocurre si las propiedades físicas del sensor de lazo han cambiado, la cual se verá reflejada en una variación de voltaje, sin embargo la señal de comparación positiva cambiará por medio del condensador que se cargará por la resistencia.

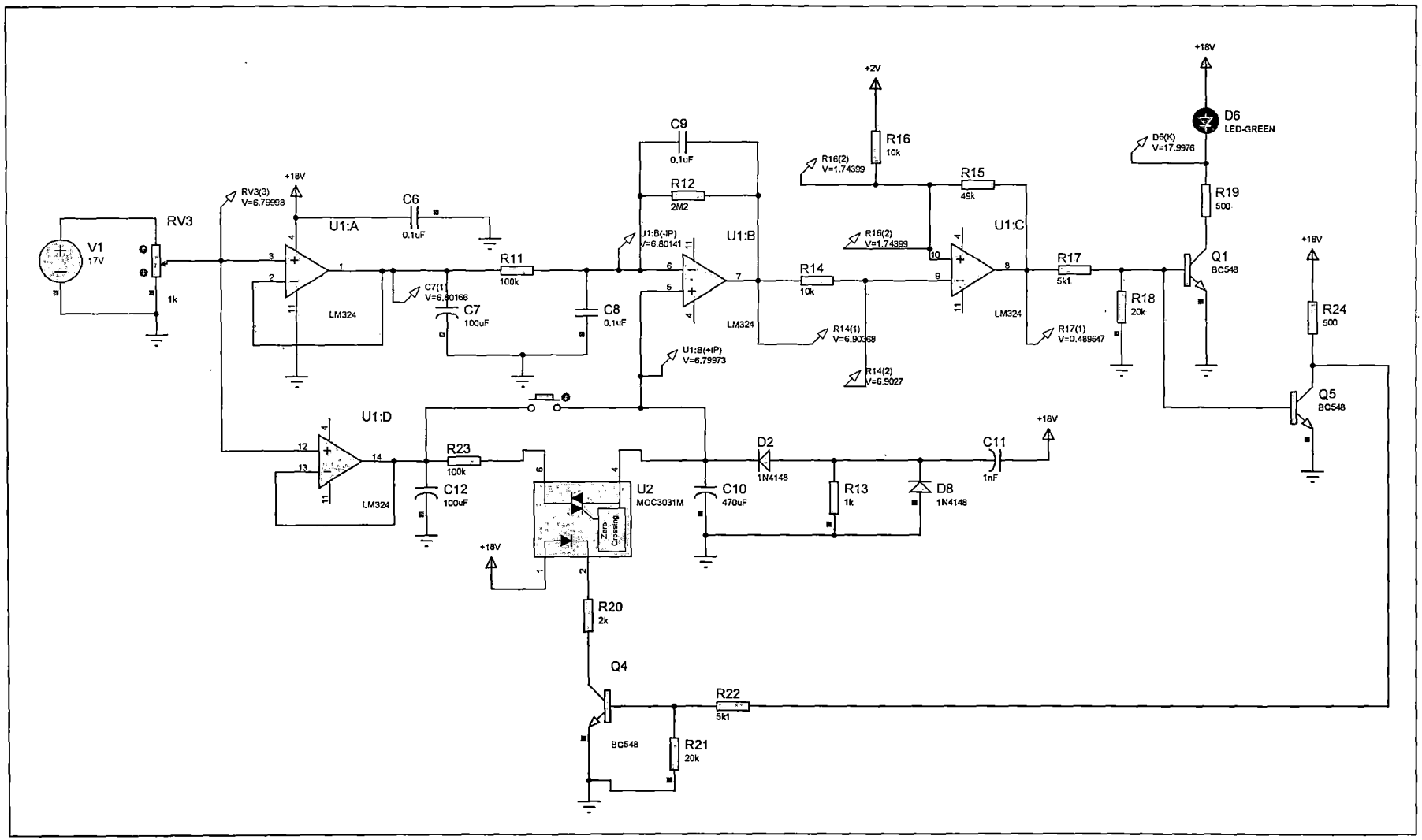



***Figura 3.47.-*** *Circuito completo con compensador de temperatura*

## B. Acondicionador de señales para el microcontrolador

Por último todas las señales, de los sensores de presión y la salida de la tarjeta acondicionadora de señales del sensor inductivo, pasarán por una tarjeta que finalmente digitalizará y preparará las señales digitales para la tarjeta del microcontrolador.

El diseño de esta tarjeta es simple, cuyo circuito se muestra en la figura 3.48. Las señales que llegan a esta tarjeta son sólo digitales:

- **Sensor de unidades.-** esta señal es la del sensor loop de antena ya acondicionada por el circuito detector de masa metálica, ya explicada anteriormente. Recordemos que esta señal es de 0V cuando no hay vehículo y 18V cuando pasa un vehículo, porque la tarjeta anterior sólo es alimentada con 18V.
- **Sensor de eje B0.-** proviene de uno de los sensores de peanas dispuestos a 45° para detectar ejes duales.
- **Sensor de eje B1.-** proviene de otro de los sensores de peanas dispuestos a 45° para detectar ejes duales.
- **Sensor de eje B2.-** proviene de la unión en paralelo de los dos sensores alineados en forma horizontal para detectar ejes totales.

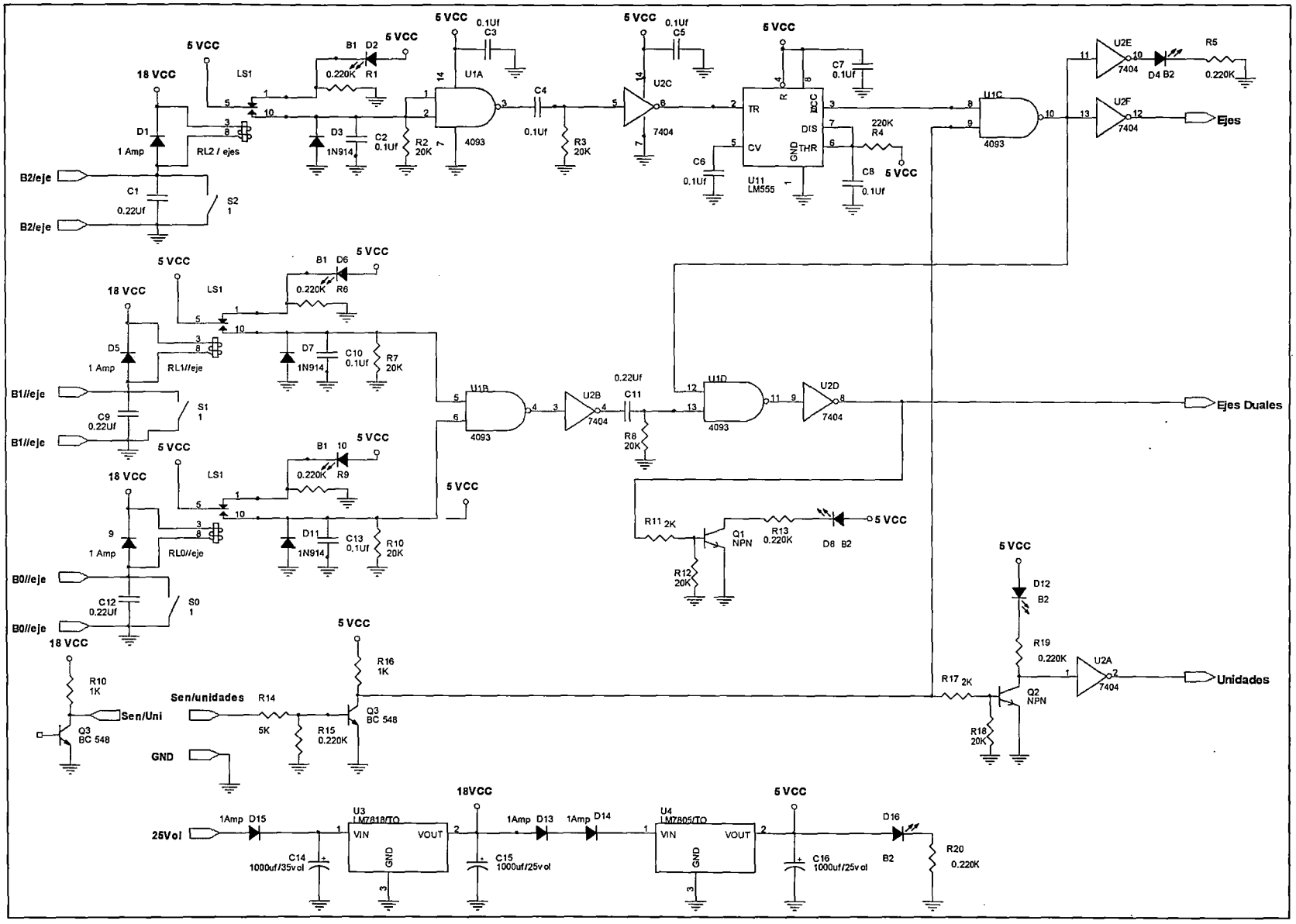



***Figura 3.48.-*** *Circuito acondicionador de señales para el microcontrolador*

La señal de los sensores de peanas sólo son ON, OFF, ya que funcionan como un switch normalmente abierto, el cual se cierra cuando un eje lo presiona. Esta tarjeta recibe dos cables por cada sensor de peana (sensor de presión para la detección de ejes. Ver capítulo 3.4). Todos funcionan de manera similar, es decir que están conectados a un relay, con el cual la bobina quedará energizada en el momento que se cortocircuiten los dos cables que vienen de los sensores. Después de esto sólo pasarán por filtros los cuales eliminarán significativamente los rebotes debido al accionamiento mecánico, sin embargo habrá ruido que no podrían eliminarse, sin embargo esto no es importante para los sensores encargados de detectar los ejes duales (B0 y B1) ya que no se necesita saber con exactitud la cantidad de ejes duales, sino saber si el vehículo tiene ejes duales, ya que de esa manera se podrá saber si es o no un camión.

**Ejes duales.**

Debemos recordar que cualquier unidad que pase puede activar los sensores de ejes duales B0 y B1, sin embargo será sólo uno a la vez; pero lo camiones, que son los vehículos que normalmente tienen ejes duales, activarán los dos sensores B0 y B1 a la vez, por la disposición geométrica en la que se encuentran en la loza de la pista. Por esta razón se utilizará una compuerta NAND y filtros (buffer, resistencias y condensadores) para conseguir estas condiciones.

**Ejes totales.**

Aquí si es importante mejorar el filtro, de tal manera que no se pierda ejes, ni existan ejes de más, por esta razón que se dispuso de un TIMER 555 que trabajará como un MONOESTABLE (ver figura 3.49), el cual su tiempo de señal en alto se calcula mediante la siguiente fórmula:

$Ton = (1.1)R1.C1$ :.....(3.12)

Ton= (1.1) x (220k) x (0.1u)= 24.2ms

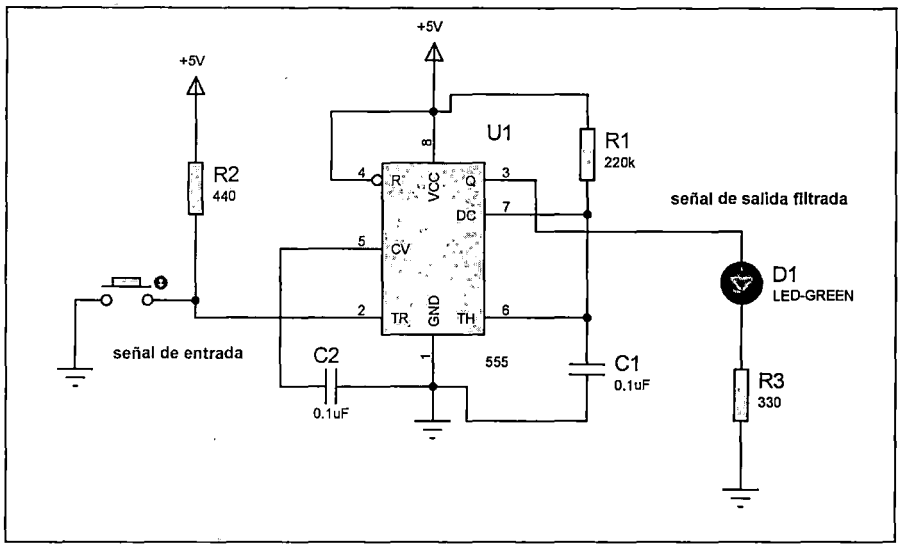



***Figura 3.49.-*** *Circuito temporizador para eliminar rebotes del sensor de ejes*

Tiempo suficiente y no tan largo como para filtrar correctamente la señal y contar como un eje por el microcontrolador. Esto es, la señal puede entrar ruidosa, con muchos pulsos, sin embargo esto representa sólo la presión de un eje, pero luego de pasar por el TIMER 555 sólo saldrá un pulso temporizado al valor que se calculó y así la señal será correcta.

Finalmente se debe recordar que, de acuerdo a la disposición de los sensores en la pista, las señales de los sensores de ejes totales y de ejes duales nunca se darán al mismo tiempo, sin embargo, es posible que en lugar de usar sensores para ejes duales y así saber si el vehículo fue un camión, se use sensores de altura (infrarrojo) para saber si fue un auto o un camión. En este caso el sensor dará muchos pulsos debido a las rendijas o a las ventanas, incluso cuando se está contando un eje, esto podría conllevar a un doble conteo de ejes totales, el cual si es importante que no suceda, por esta razón es que la señal de ejes será como una señal de control para una compuerta inhibidora (U1D 4093) para la señal de ejes duales.

**Unidades**

Recordemos que la señal que llega es de 0 y 18V, esta se convertirá a 0 y 5V con un transistor. Esta señal permanecerá en alto todo el tiempo que el vehículo se demore en pasar por la antena, y durante este tiempo es que los sensores de ejes se podrán activar; sin embargo para que no se realice un conteo erróneo, es que la señal del sensor de unidades (sensor loop de antena) servirá como una señal de control para una compuerta inhibidora (U1C 4093) para las señales de ejes totales.

Finalmente todas tendrán indicadores de luz (LED) e inversores que realizarán la labor de buffer (U2 7404) por si se requiera más fuerza en la señal. Las señales acondicionadas se muestran en la figura 3.50.

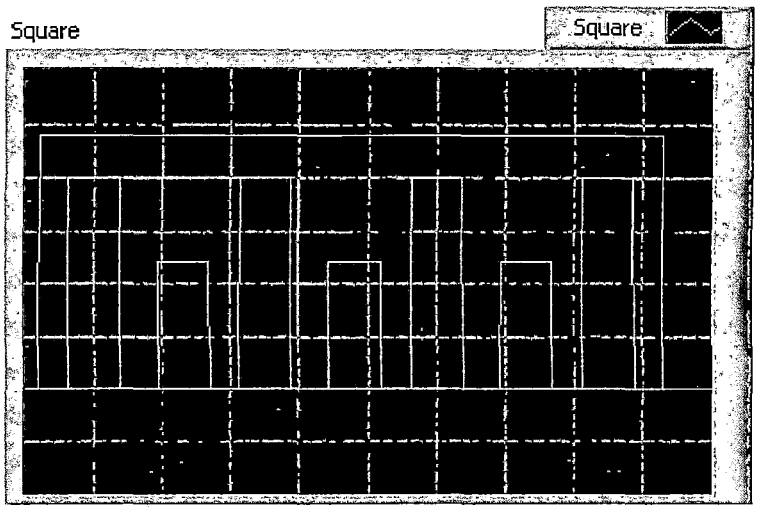



***Figura 3.50.-*** *Señales acondicionadas para el microcontrolador. Blanco, unidad. Rojo, ejes totales. Verde, ejes duales.*

Donde:

- La señal del sensor de antena es de color blanco.
- Los ejes totales es la de color rojo.
- Los ejes duales de color verde.

### 3.4.2. Tarjeta de sensores de estado del tablero eléctrico electrónico

Es importante tener en cuenta ya el diseño de esta tarjeta el cual será útil cuando se implemente el sistema interconectado de supervisión a distancia.

Las señales que se piensan sensar para determinar el estado del tablero de control serán:

- Temperatura
- Nivel de voltaje de baterías

- Estado de la puerta del tablero de control

## A. Diseño de circuito para el sensor de Temperatura

Se pueden utilizar cristales de germanio dopados de forma homogénea para detectar temperaturas próximas al cero absoluto. La curva de respuesta resistencia-temperatura del dispositivo se parece a la del termistor y tiene un coeficiente de temperatura negativo y no es muy lineal.

También se emplean cristales de silicio como sensores de temperatura. Su rango de utilización se extiende desde –67° a 275° F. A diferencia de los cristales de germanio, los cristales de silicio tienen un coeficiente de temperatura positivo en este rango. Por debajo del rango, la curva de respuesta resistencia-temperatura se hace negativa y bastante no lineal.

Una unión PN polarizada en inversa tiene una pequeña corriente de fuga, o de portadores minoritarios. La magnitud de ese flujo de corriente depende de la temperatura. Cuando la temperatura aumenta, la corriente de fuga aumenta exponencialmente. Este efecto se utiliza para medir la temperatura con transistores y diodos polarizados en inversa.

En este caso usaremos el transistor de base metálica 2N2222 en cuya hoja de datos se puede observar la curva de variación de voltaje base-emisor (VBE) con respecto a la temperatura ambiente, esto está caracterizado

mediante un coeficiente de temperatura Rθ (ver figura 3.51). En esta curva dicho coeficiente expresa como lo muestra la expresión 3.13.

$$\boxed{R\theta = \frac{\Delta V_{BE}}{\Delta T}} \ldots\ldots\ldots\ldots (3.13)$$

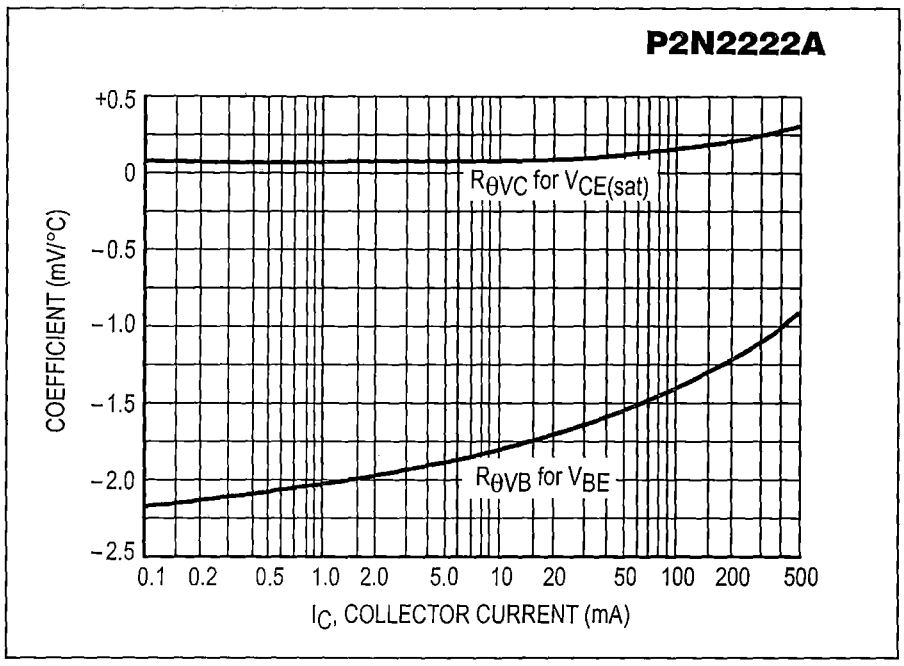



***Figura 3.51.-*** *Curva de variación de voltaje base-emisor con respecto a la temperatura*

De acuerdo a la curva característica este coeficiente es negativo y esto significa que conforme la temperatura aumenta el VBE disminuye. Con este propósito convertiremos el transistor en un diodo, es decir que cortocircuitaremos la base con el colector, para que llegue la suficiente corriente a la base como para que el transistor conduzca y se comporte como un diodo. Como el comportamiento es no lineal, se tratará de linealizar

colocando una resistencia en serie R1. De aquí tendremos el circuito de la figura 3.52.

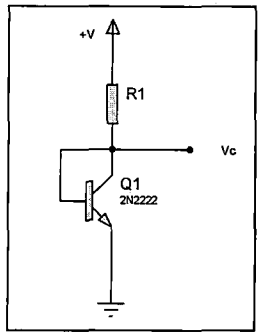



***Figura 3.52.-*** *Transistor 2N2222, en disposición de diodo para ser usado como sensor de temperatura*

Buscaremos una resistencia para generar una corriente en el colector adecuada dentro del rango permitido. Luego el circuito que tendremos se muestra en la figura 3.53.

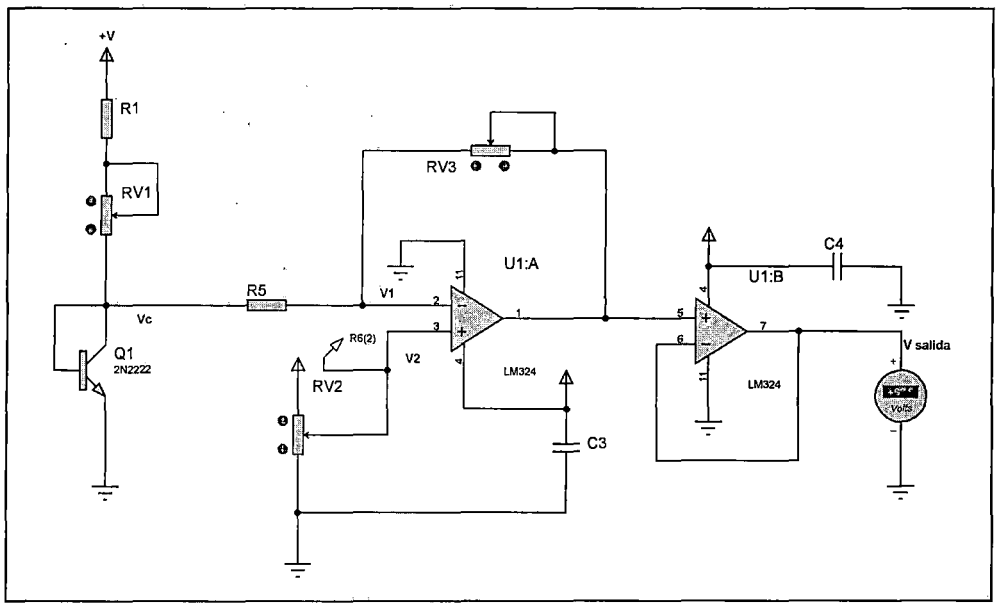



***Figura 3.53.-*** *Circuito acondicionador de señales para el sensor de temperatura, transistor 2N2222 de base metálica*

Al polarizar el transistor Q1 circula una corriente de colector, que cuando pasa por el resistor (R1+RV1) produce una caída de voltaje que dependerá de la corriente del colector y este a su vez depende de la caída de voltaje base-emisor.

Debido al coeficiente negativo de temperatura que presenta este dispositivo activo este voltaje disminuirá. Dicho voltaje, por ser tan pequeño, es amplificado por U1A, un amplificador operacional de alimentación única contenido en el circuito integrado LM324, y el cual está configurado como un amplificador restador, los cuales variando el potenciómetro RV3 se puede controlar 2la ganancia de dicho amplificador restador, y mediante el potenciómetro RV2 se puede controlar el voltaje V2 de referencia a diferenciar. Por mejorar y eliminar cualquier ruido se pondrán condensadores como filtro y además un seguidor de voltaje al final para asegurar que no pasará una corriente excedente al microcontrolador. El voltaje de salida del circuito por lo tanto será:

$$Vout = (1 + \frac{RV3}{R5}).V2 - \frac{RV3}{R5}.V1 \ldots\ldots\ldots(3.14)$$

Las curvas características de este tipo de transistor se muestran en la figura 3.54. De acuerdo a esto se elegirá trabajar con un voltaje base-emisor $V_{BE}$= 0.72V que corresponde a una corriente de colector Ic= 8mA del gráfico. Para esto tendremos una resistencia (R1+RV1) de:

$$R1 + RV1 = \frac{9 - 0.72}{0.008} = 1035\Omega$$

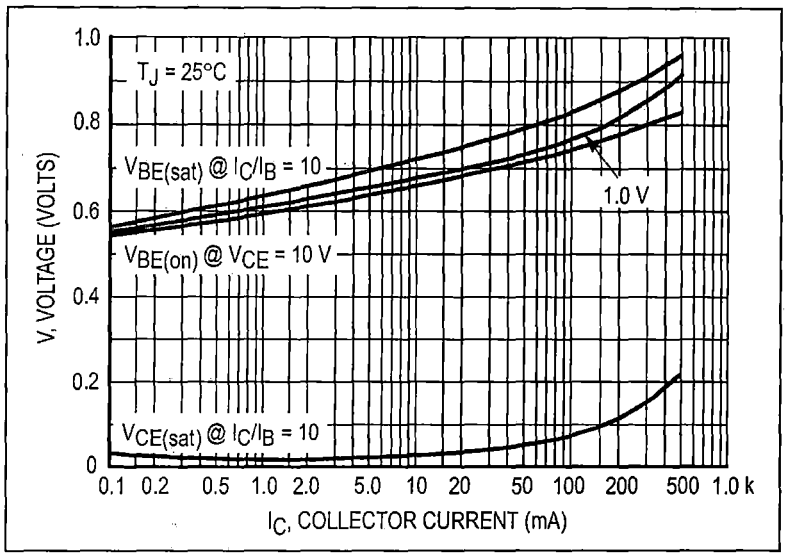



***Figura 3.54.-*** *Curva de relación de voltaje base-emisor y corriente del colector del transistor 2N2222*

Esto significa que pondremos una resistencia base R1=50Ω y un RV1=1kΩ. Finalmente simulamos este circuito en el software PROTEUS y tendremos lo mostrado en la figura 3.55. De aquí podemos ver que los valores calculados y los simulados corresponden.

Se eligió además del potenciómetro RV3=100k una resistencia de de base R2=10k y la resistencia R5=10k, de acuerdo a lo simulado podemos observar que tenemos un voltaje de 2V que ingresaría al controlador.

Como sabemos que el voltaje base-emisor disminuirá con el aumento de temperatura debido al coeficiente negativo y por lo tanto el voltaje de 2V aumentará de acuerdo a la expresión:

$$Vout = (1 + \frac{RV3}{R5}).V2 - \frac{RV3}{R5}.V1$$

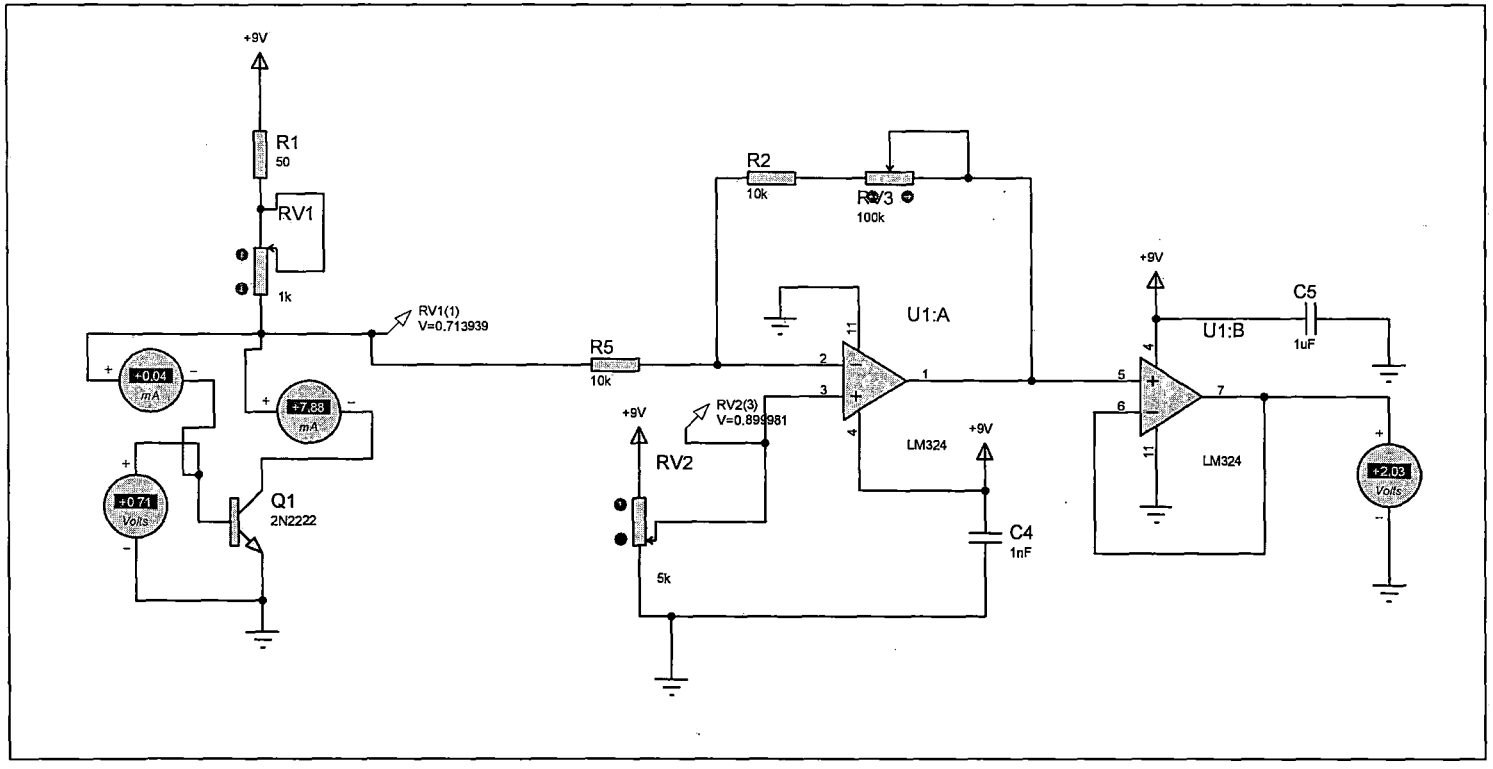



***Figura 3.55.-*** *Simulación del circuito acondicionador de señales para el sensor de temperatura*

Para Ic=8mA tendremos Rθ=-2.1mV/°C. Citemos el siguiente ejemplo para poder comprender.

***Ejemplo***

Si la temperatura del ambiente aumenta de 25°C a 30°C entonces:

$$\Delta T = 30 - 25 = 5°C$$
$$\Delta V_{BE} = -2.1(5) = -10.5mV$$
$$V_{BE} = 720mV - 10.5mV = 709mV$$

Si tenemos en cuenta que la variación de la temperatura podría estar entre -10°C y +100°C, entonces tendremos que el voltaje podría variar de la siguiente manera:

$$desde \quad \Delta V_{BE} = -2.1(-35) = 73.5mV \quad hasta \quad \Delta V_{BE} = -2.1(75) = -157.5mV$$

Lo cual nos dará un rango de variación de voltaje de base-emisor de:

$$V_{BE}\max = 720mV + 73.5mV = 793.5mV \quad para \quad T = -10°C$$
$$V_{BE}\min = 720 - 157.5 = 562.5mV \quad para \quad T = 100°C$$

Por lo tanto se buscará que regular la salida de voltaje de manera que no se sature en alto o bajo para los voltaje de referencia del convertidor análogo digital que dispone el PIC16F877A. Para esto se simuló en PROTEUS si con esto valores de resistencias y potenciómetro podríamos llegar a lo deseado (ver circuito en figura 3.56). Se buscará que el voltaje

Vout de salida final del circuito del sensor no supere los +5V o esté debajo de 0V.

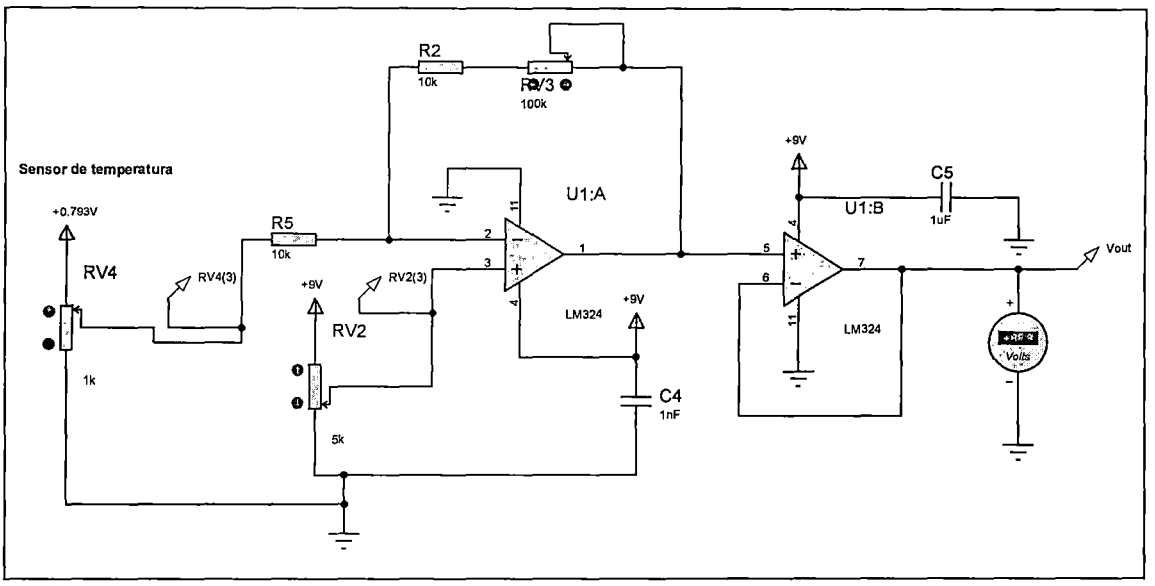



***Figura 3.56.-*** *Circuito para la simulación del acondicionamiento de señal del sensor de temperatura*

Para simular el sensor de temperatura (2N2222) se pondrá un potenciómetro el cual lo haremos variar de acuerdo a:

0.793V para -10°C

0.562V para +100°C

Ya que en los lugares de la sierra la temperatura podría bajar considerablemente.

Es importante recordar que en este caso sólo es importante verificar que no se esté produciendo un sobrecalentamiento, y sólo nos bastaría que la temperatura se encuentre entre 10°C y 90°C.

Simulando podemos observar las señales del sensor y de la salida amplificada y acondicionada del circuito los cuales en un extremo varía desde +2V hasta +6V. (Ver figura 3.57).

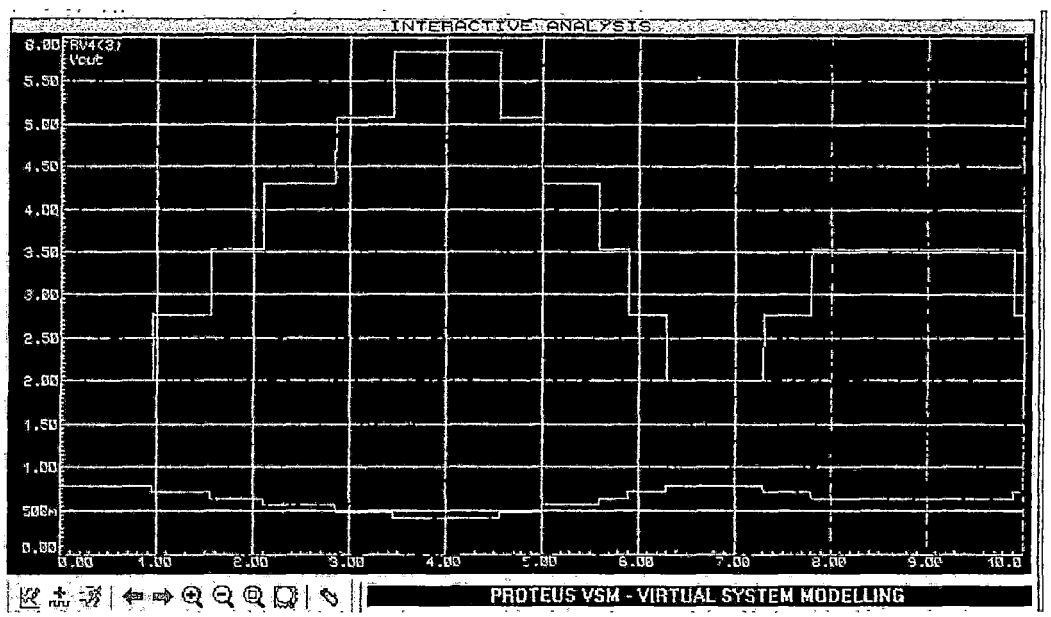



***Figura 3.57.-*** *Simulación de señales del sensor de temperatura con el acondicionamiento de señal. La línea verde representa la señal del sensor de temperatura y la roja es la señal acondicionada*

**Nota**

Se debe recordar que el hecho de monitorear la temperatura no tiene mucha importancia sino si se trabaja con un sistema de monitoreo a distancia, es decir vía Internet desde la central en Lima; sin embargo como la propuesta es diseñar este sistema de monitoreo, se está preparando las tarjetas para este propósito, pudiendo detectar cualquier evento mediante una alarma. La calibración del sensor de temperatura se verá en la sección de monitoreo a distancia.

Este sistema se explicará con mayor detalle más adelante, sin embargo este esquema se puede apreciar en la figura 3.58.

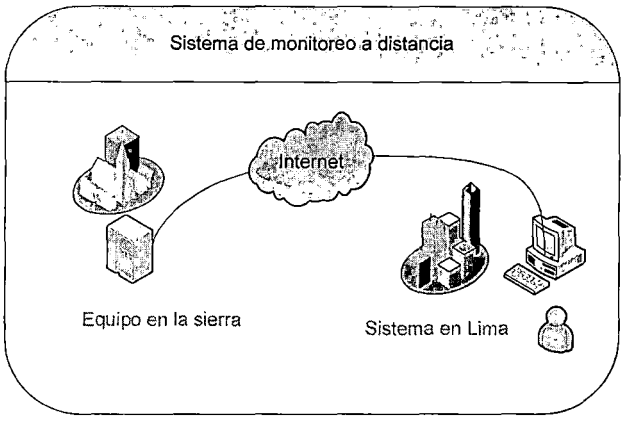



*Figura 3.58*

**B. Diseño de circuito para los sensores de estado de puerta y nivel de carga de baterías**

Para esto se usará simplemente un divisor de voltaje y así convertir los 18V hasta 5V, de esta manera la señal irá al conversor analógico digital del microcontrolador para luego enviar el valor al computador cuando se desee. Asimismo para verificar el estado de la puerta del tablero eléctrico sólo se dispondrá de un switch, el cual conmutará el estado de la entrada de un pin del puerto de microcontrolador.

### 3.4.3. Tarjeta de control por caseta

Esta es la tarjeta principal la cual hace muchas labores que iremos viendo. La tarjeta está diseñada a base del microcontrolador de MICROCHIP de la familia 16F87X.

## A. Componentes

La tarjeta tiene los siguientes componentes:

- Un microcontrolador PIC16F877A de MICROCHIP
- Un cristal de cuarzo de 20MHz
- Dos condensadores de 22pF
- Un cristal de cuarzo de 32.768kHz
- Siete condensadores de 0.1uF
- Una pantalla LCD 2x16
- Cuatro memorias I2C EEPROM 24LC512
- Dos resistencias de 20kΩ
- Un puente de diodos
- Dos condensadores de 1000uF de 25V electrolíticos
- Cinco condensadores de 1uF de 50V electrolíticos
- Una resistencia de 1kΩ
- Una resistencia de 20kΩ
- Un diodo de 1amperio
- Un MAX232
- Un transistor BC549
- Un Relay de 12V
- Un regulador 7805
- Un regulador 7812
- Un conector DB9 hembra

## B. Protocolo de comunicación

El microcontrolador de MICROCHIP PIC16F877A dispone de pines para el protocolo de comunicación serial RS232, para lo cual se configurará para transmisión y recepción FULL DUPLEX 8bits de datos un bit de START y uno de STOP, sin bit de paridad y a una velocidad de 57600 baudios.

## C. Subrutinas usadas

El programa está desarrollado en lenguaje asembler en el entorno MPLAB y compilador MPASM. Los archivos *includes* para las diversas subrutinas fueron:

- Macros.inc
- include "lcd4c.inc"
- include "24LCXXX2.inc"
- include "serial.inc"
- include "delay.inc"
- include "fecha.inc"
- include "matematica.inc"
- include "interrupciones.inc"
- include "configuracion.inc"

### a) Macro.inc:

En ellas se crearon macros para la simplificación de códigos repetitivos dentro del programa así como funciones.

**Macros para funciones**

En este caso se usó una lógica para manejar variables locales o temporales, es decir, variables que al igual que en el lenguaje "C" cuando se ingresa a una función se crean mientras que están en la función y luego se destruyen al salir de ella, así nos ahorramos espacio en la memoria RAM.

Esto se logró mediante el direccionamiento indirecto, es decir las rutinas que vienen a continuación es para crear una pila es decir almacenamiento de registros tipo LIFO en una parte de la RAM, con un registro que lleve la cuenta que servirá de puntero de pila para reservar espacio en ellas para las variables temporales y luego de terminada la función desechar ese espacio reservado para así que esté disponible para otras variables temporales o locales.

En este caso se usa un registro de la RAM para que sirva de variable de puntero de pila al cual se le llamó "puntero". Entonces cada vez que se llame a una función, el puntero se debe incrementar en la cantidad de variables que se usarán para esa función, para esto es el siguiente código:

```
num_var_local macro cantidad
        movlw cantidad
        addwf puntero,f
        movf puntero,w
        movwf FSR
        endm
```

Luego mediante los registros para direccionamiento indirecto FSR e INDF, se usarán las variables que están por debajo de la dirección que contiene el puntero de pila, decrementándolo o incrementándolo.

```
var1_local macro
        endm
var1_local_fin macro
        endm

var2_local macro
        decf FSR,f
        endm
var2_local_fin macro
        incf FSR,f
        endm
var3_local macro
        decf FSR,f
        decf FSR,f
        endm
var3_local_fin macro
        incf FSR,f
        incf FSR,f
        endm
var4_local macro
        decf FSR,f
        decf FSR,f
        decf FSR,f
        endm
var4_local_fin macro
        incf FSR,f
        incf FSR,f
        incf FSR,f
        endm
var5_local macro
        decf FSR,f
        decf FSR,f
        decf FSR,f
        decf FSR,f
        endm
var5_local_fin macro
        incf FSR,f
        incf FSR,f
        incf FSR,f
        incf FSR,f
        endm
```

Finalmente terminada la función se debe volver a decrementar el puntero en la misma cantidad de variables que se utilizó, para así regrese a su dirección anterior.

```
fin_subrutina macro cantidad
        movlw cantidad
        subwf puntero,f
        endm
```

**Macro para grabar en la memoria EEPROM del PIC**

MICROCHIP ya da en la hoja de datos técnica el código necesario para escribir, borrar y/o leer la memoria EEPROM, pero para no estar cargando de continuamente el dato y la dirección en la que se debe grabar, se elaboró las siguientes macros:

- **Macro para retardos**.- aquí se tienen las rutinas para los retardos de microsegundos, milisegundos y segundos.
- **Macros adicionales auxiliares.-** macro que convierte un número en binario natural de 16bits (2bytes) en BCD de 5 dígitos. El formato es: Convertir<byte_h>,<byte_l>,<dígito1>,<dígito2>,<dígito3>,<dígito4>,<dígito5>

**b) Matemática.inc:**

Aquí se disponen de dos rutinas: multiplicar_por_10 y conversion_bcd_2bytes. Estas rutinas se usarán para mostrar datos en la pantalla LCD y para la fecha.

### c) lcd4.inc

Presenta un conjunto de rutinas que permiten realizar las tareas de control del módulo de visualización LCD. Con la llamada a LCD_UP se configura para su uso y con la rutina LCD_INI se inicializa el LCD. Y la rutina MENSAJE visualiza un mensaje dado en una tabla. Se define con "Define" los puertos LCD_DATA y LCD_CTRL. La forma como se conectará la pantalla LCD al microcontrolador se puede ver en la figura 3.59.

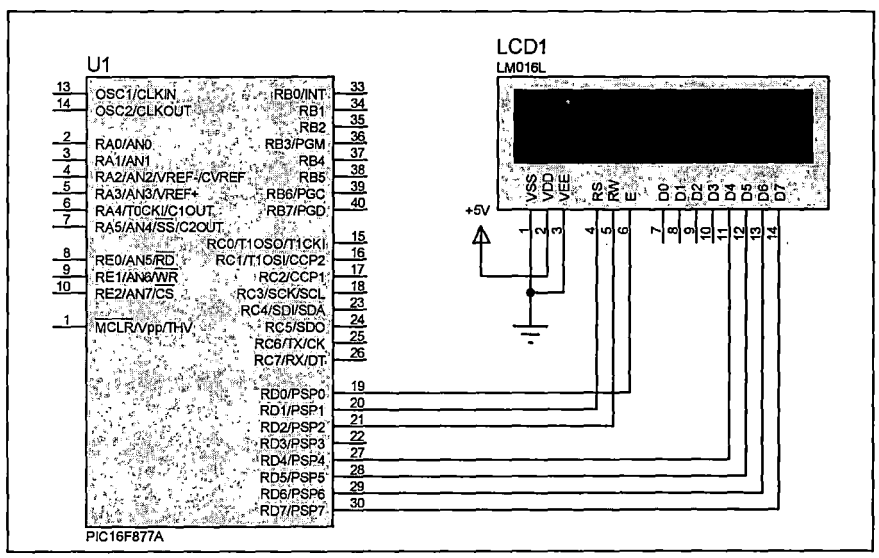



***Figura 3.59.-*** *Circuito de conexión con la pantalla LCD*

## D. Funcionamiento

La tarjeta trabajará de la siguiente manera:

- Información de seguridad
- Configuración de las memorias EEPROM
- Configuración de la fecha y hora

- Almacenamiento de datos de vehículos
- Descarga de datos guardados
- Supervisión en tiempo real desde la PC
- Supervisión de estado de sensores

**a) Información de seguridad**

A pesar del sistema de seguridad eléctrico que respalda su fuente, el microcontrolador además tiene algunos sistemas de seguridad por software como son:

**Pow-Up-Timer:** este es un temporizador de arranque para que el microcontrolador, una vez energizado, no inicie inmediatamente sino que espere unos milisegundos hasta que los niveles de voltaje de alimentación se nivelen.

**Información de reset:** esta es una información muy importante, ya que la tarjeta indicará la cantidad de veces que se reseteo o apagó el equipo. Esto se podrá mostrarlo en el software de la PC, indicando la hora, fecha de y la cantidad de veces de dicho evento. Esta información se guarda en la dirección 0x03 de la memoria EEPROM interna del microcontrolador. En la figura 3.60 se muestra el diagrama de flujo de la inicialización del sistema.

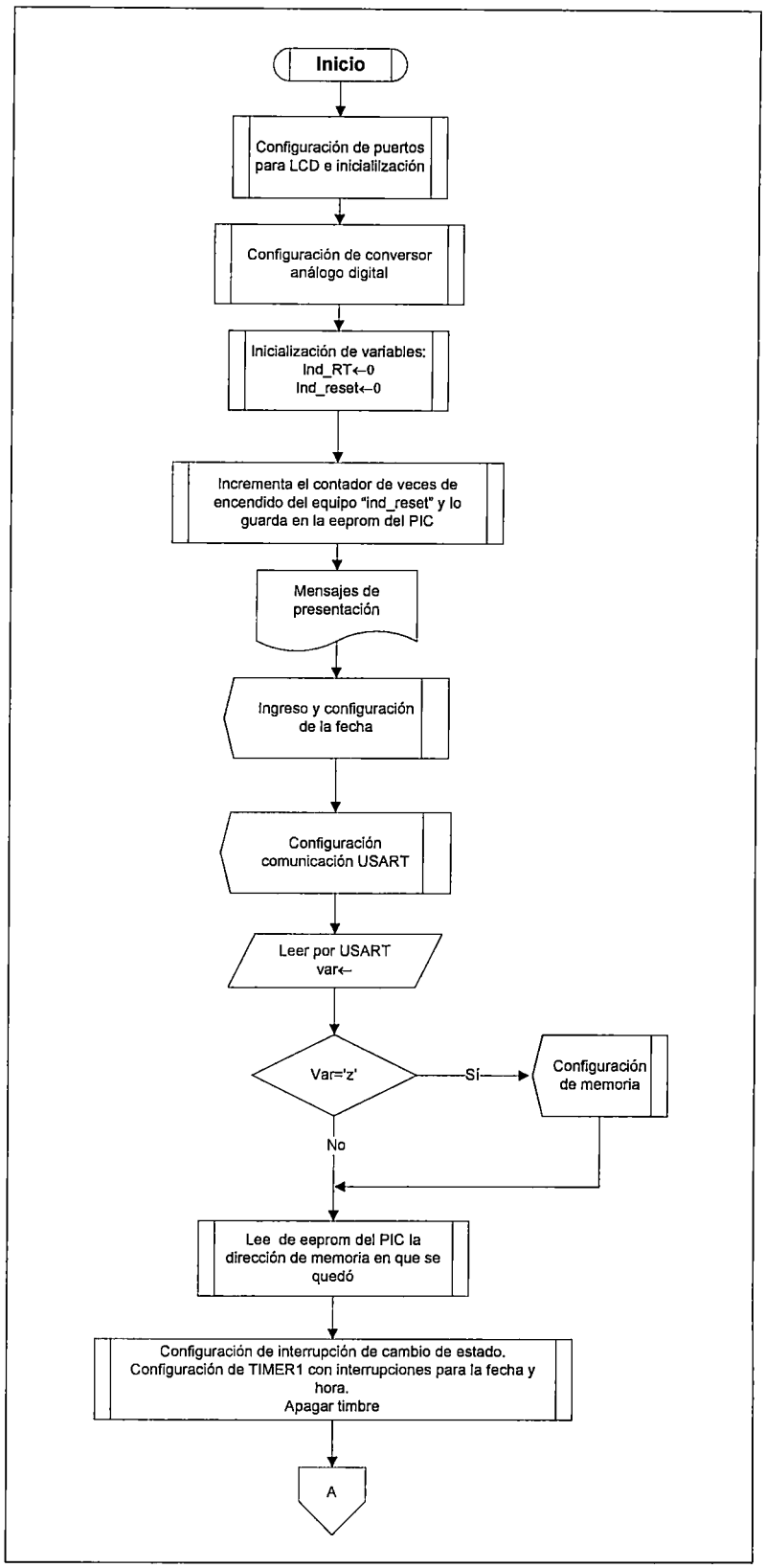



***Figura 3.60.-*** *Diagrama de flujo de configuraciones iniciales*

El código del programa en este caso es:

```
;**********************************
bcf banderas,ind_RT
clrf ind_reset
lee_eeprom 0x06
movwf num_reset
incf num_reset,1
graba_eeprom 0x06,num_reset
;**********************************
```

**b) Configuración de las memorias EEPROM 24LCXXX**

En el momento en que se enciende la tarjeta de control saldrá un mensaje de presentación en la pantalla LCD (ver figura 3.61) y luego de 2s el mensaje: "Pulse OK....", esto se muestra en la figura 3.62.

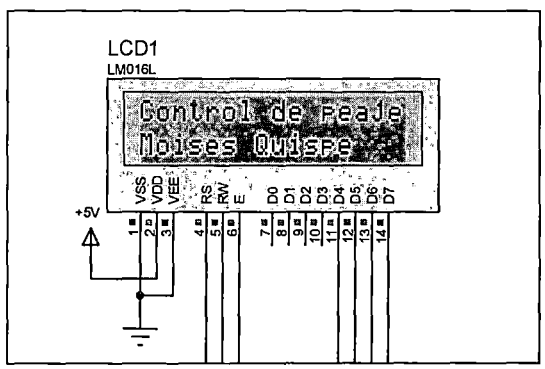



***Figura 3.61.-*** *Mensajes de presentación de pantalla LCD*

En este segundo mensaje el microcontrolador está esperando la posibilidad del envío de 2 diferentes códigos ASCII. Si se envía, desde el computador, la letra 'z' entonces pasará a otra subrutina en la que esperará el envío del número de memoria y la dirección del registro donde iniciar, y con cualquier otro caracter continuará con el funcionamiento normal.

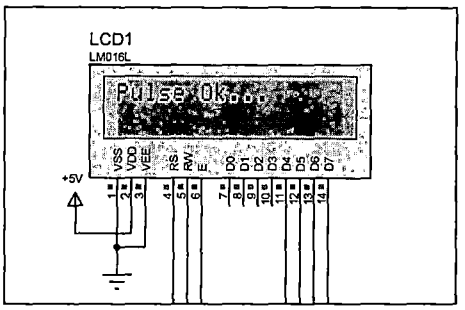



***Figura 3.62.-*** *Mensaje para confirmación de configuraciones*

Toda la información que se requiere almacenar se harán en las memorias EEPROM 24LCXXX. Existen varios tipos de estas memorias (ver figura 3.63), para este caso podremos usar cualquiera las siguientes:

- 24LC128 (128kbits, esto es 16kbytes o 16384bytes)
- 24LC256 (256kbits, esto es 32kbytes o 32768bytes)
- 24LC512 (512kbits, esto es 64kbytes o 65536bytes)

Además se puede usar desde 1 hasta 4 memorias por tarjeta de control. El tipo de memoria a usar, la cantidad de memorias y la dirección del registro de inicio de grabación de los datos pueden ser configurados desde el programa de control en PC. Es recomendable empezar a grabar desde la dirección cero de la primera memoria. Las memorias se controlan mediante el protocolo I2C; es decir todas las memorias están unidas a dos pistas, una de ellas es el clock (PIN6) para la sincronización de la velocidad y la otra (PIN5) es para enviar y recibir datos.

Los pines 5 y 6 del integrado, para comunicación I2C, se conectan con el microcontrolador por medio de resistencias pull-up de 10 a 20kΩ. Ver figura 3.64.

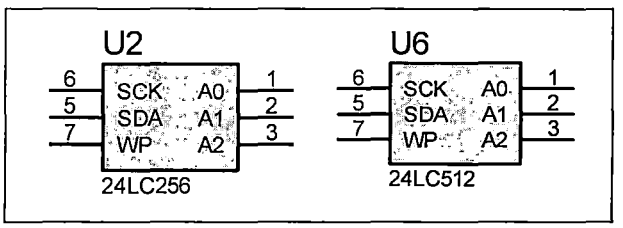



***Figura 3.63.-*** *Memorias EEPROM I2C*

A cada una de las memorias se le asignan una dirección por hardware, es decir por medio de los pines 1(A0), 2(A1) y 3(A2). Ver tabla 3.6.

***Tabla 3.6***
*(Direcciones de las memorias EEPROM I2C)*

| ***Nombre*** | ***Dirección*** | | |
|---|---|---|---|
| | ***A2*** | ***A1*** | ***A0*** |
| Memoria 1 | 0 | 0 | 0 |
| Memoria 2 | 0 | 0 | 1 |
| Memoria 3 | 0 | 1 | 0 |
| Memoria 4 | 0 | 1 | 1 |

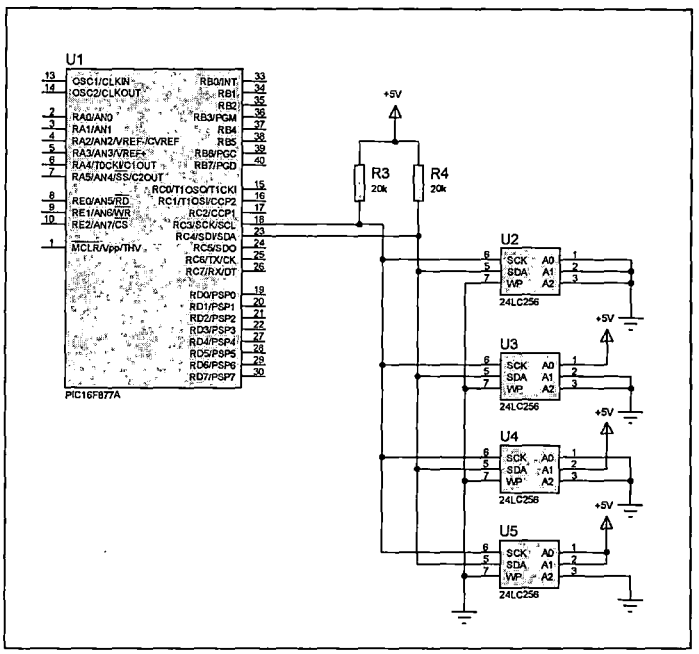



***Figura 3.64.-*** *Circuito de conexión de las memorias I2C al microcontrolador*

Inmediatamente después del último mensaje visto, existen 2 alternativas:

- Continuación normal del programa. Al enviar cualquier caracter.
- Configuración de la tarjeta. Al enviar el caracter 'z'.

Una vez enviada el caracter 'z' desde el computador, el microcontrolador espera una cadena de 6 bytes en hexadecimal y el caracter return (0x0A). El diagrama de flujo para la configuración de las memorias se puede ver en la figura 3.65.

La cadena de los 6 bytes en hexadecimal lleva la siguiente información:

- **Primer byte.** Contiene la dirección del número de memoria a partir del cual se iniciará a grabar los datos, esto es lo que se conoce como el

CHIPSET, este dato se guardará en la dirección 0x00 de la memoria EEPROM del microcontrolador.

- **El segundo y tercer byte.** Esta información es la dirección del registro dentro de la memoria a partir del cual se comenzará la grabación de datos, por la capacidad de información a grabar es que se usa 2bytes para direccionar los datos, primero se envía el más significativo y luego el menos significativo los cuales se guardarán respectivamente el las direcciones 0x01 y 0x02 de la memoria EEPROM del PIC.
- **El cuarto byte.** Contiene la dirección del último registro de la memoria hasta la que se desee grabar. Se guarda en la EEPROM del PIC en la dirección 0x04.
- **El quinto y sexto byte.** Estos dos bytes definen el tamaño de la memoria con las que se desea trabajar, pudiendo ser 16384 para la memoria 24LC128, 32768 para la memoria 24LC256 y 0 para la 24LC512, estos números se usan en el programa. Esta información también se guarda en la memoria EEPROM del PIC el más significativo y luego el menos significativo en las direcciones 0x05 y 0x06 respectivamente.
- **El sétimo byte**. Es sólo el "return" para confirmar la finalización de envío de información.

El microcontrolador, una vez recibido los datos, reenvía lo recibido a la PC para que confirme si se recibió el dato correctamente.

Esto es solamente necesario hacerlo una vez, por la persona autorizada para configurar el equipo, ya que después de ello la dirección del registro en la que se quedó se guardará y no se perderá aún apagado el equipo, ya que las memorias EEPROM son no volátiles.

Después de esto sólo se accederá a esta función en caso de mantenimiento, si fuera necesario, o en casos especiales.

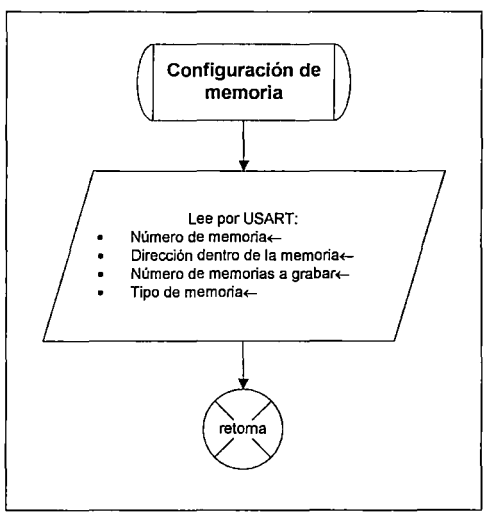



***Figura 3.65.-*** *Diagrama de flujo para la configuración de memorias*

En funcionamiento normal el microcontrolador leerá inmediatamente la información guardada en su EEPROM interna y se actualizará constantemente cada hora. La subrutina para la lectura de la posición en la que se quedó es la siguiente:

```
lee_direccion
        lee_eeprom 0x00
        movwf I2CCHIPSET
```

```
lee_eeprom 0x01
movwf I2CADDRESS_H
lee_eeprom 0x02
movwf I2CADDRESS_L
lee_eeprom 0x04
movwf tamaño_h
lee_eeprom 0x05
movwf tamaño_l
lee_eeprom 0x06
movwf num_memo
return
```

En caso de no hacer al menos una vez esta operación, el microcontrolador no encontrará ninguna dirección válida y no podrá grabar ningún dato en las memorias EEPROM externas, de esta manera el programa no funcionará, ya que inmediatamente después de poner la fecha y hora ésta se guarda en memoria externa, y al no estar direccionada, no conseguirá hacerlo.

**c) Configuración de la fecha y hora**

En caso de haber enviado cualquier otro byte se mostrará el mensaje como en la figura 3.66. Se puede ingresar la fecha y la hora de acuerdo a lo indicado en la pantalla (ver figura 3.67), con la opción de corregirla hasta pulsar "enter". Este ingreso se hace mediante protocolo RS232 por el COM1 del PC con un programa hecho en LABVIEW. El diagrama del ingreso de la fecha se muestra en la figura 3.68.

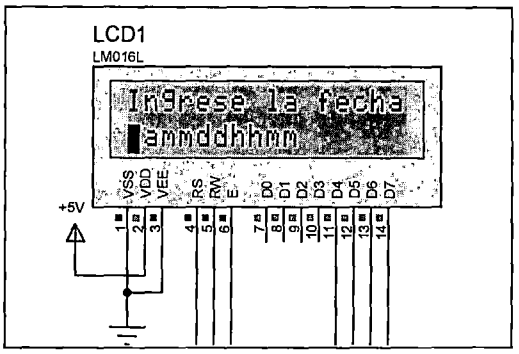



***Figura 3.66.-*** *Mensaje de ingreso de fecha*

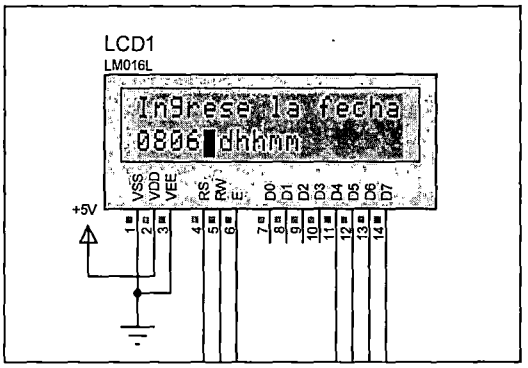



***Figura 3.67.-*** *Simulación de ingreso de fecha*

Una vez ingresada la fecha y hora en pantalla el microcontrolador leerá la información que se encuentra en la pantalla LCD, para guardarla en su memoria RAM y además guardará en las memorias I2C 4bytes en hexadecimal (0x80), los cuales serán señalizadores útiles para las funciones de búsqueda, también se guardará el año, mes, día y hora en 4bytes, incrementando constantemente la dirección de la posición en la que se quedó.

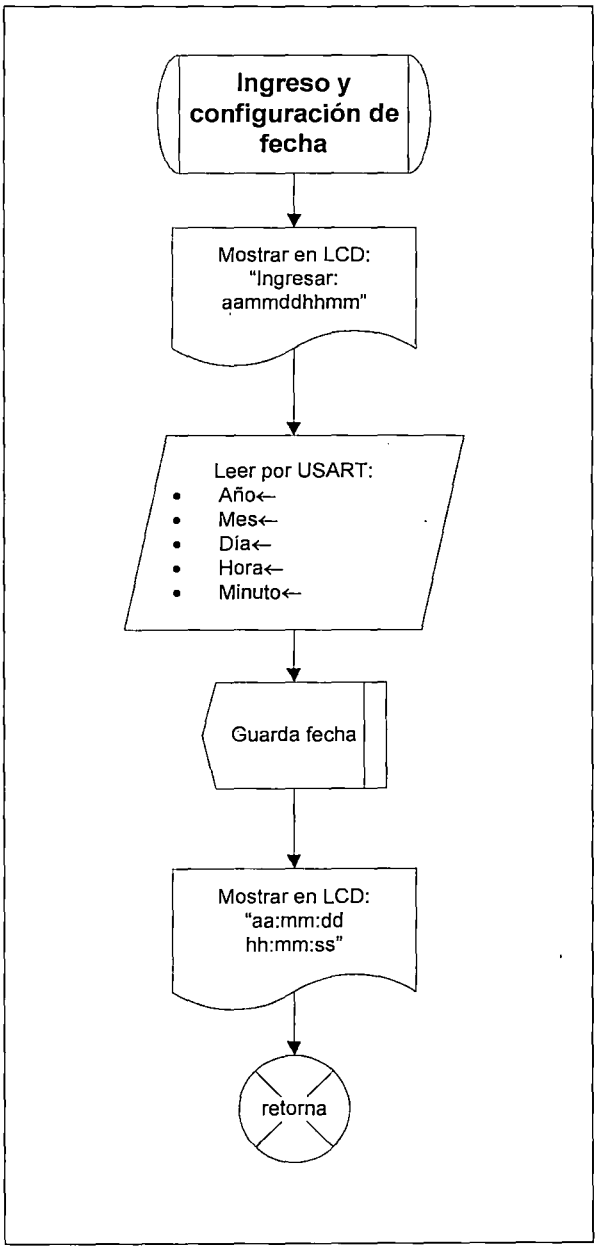



***Figura 3.68.-*** *Diagrama de flujo del ingreso de fecha*

En esta subrutina se pudo apreciar la forma en el que se guarda la trama de datos; además se puede verificar que cada vez que alguien quiera modificar la fecha, una vez encendido el equipo se aumentará el número de

veces que se reseteó al equipo, el cual está almacenado en una variable para después ser mostrada en el programa de reportes. El código es el siguiente:

```
btfss ind_reset,0
goto $+5
clrf num_reset
graba_eeprom 0x03,num_reset
bsf PORTA,3
goto $+2
bsf ind_reset,0
movlw 0x0F
call LCD_SETDDADDRE4
movf num_reset,w
call LCD_WRITEDATO4
return
```

Todos los datos se prepararon para que el microcontrolador los guarde en grupo de 8bytes dentro de las memorias I2C EEPROM. Se escogió grupos de 8bytes, porque es divisor de 16kbytes, 32kbytes y 64kbytes, es decir para poder cambiar de memoria en el código del programa no era necesario de preguntar byte a byte para saber si se llenó una memoria y así poder cambiar de memoria (ya que pueden ser más de 1), sino que se preguntará cada vez que se guarda el grupo completo de los 8bytes.

Es importante también mencionar que para poder generar con exactitud 1s para el reloj fue necesario usar un cristal de frecuencia 32.768kHz y configurar el TIMER1 del microcontrolador para recibir los pulsos generados externamente por este cristal. Una vez cumplido la cantidad de pulsos para 1s, se habilitó la interrupción de este TIMER para poder modificar la hora.

El código necesario para poder hacer esto se muestra a continuación:

```
timer1_up
        movlw b'00111110'
        movwf T1CON
        movlw 0x00
        movwf TMR1L
        movlw 0x10
        movwf TMR1H
        return

inicia_timer1
        bcf     PIR1,TMR1IF
        BANCO1
        bsf     PIE1,TMR1IE
        bsf TRISC,0
        bsf TRISC,1
        BANCO0
        bsf     T1CON,TMR1ON
        bsf INTCON,PEIE
        return
```

Dado la interrupción se hará una serie de preguntas para localizar el motivo de ello. En el caso del TIMER1 se incrementará la hora en 1s.

El número cargado en el TIMER es 0x1000 ó 4096 la cual multiplicada por 4 da 32768 justo el valor del cristal usado, por esa razón la selección de esa frecuencia.

El diagrama de flujo para la generación y el control de la fecha y hora se muestran en la figura 3.69 y 3.70.

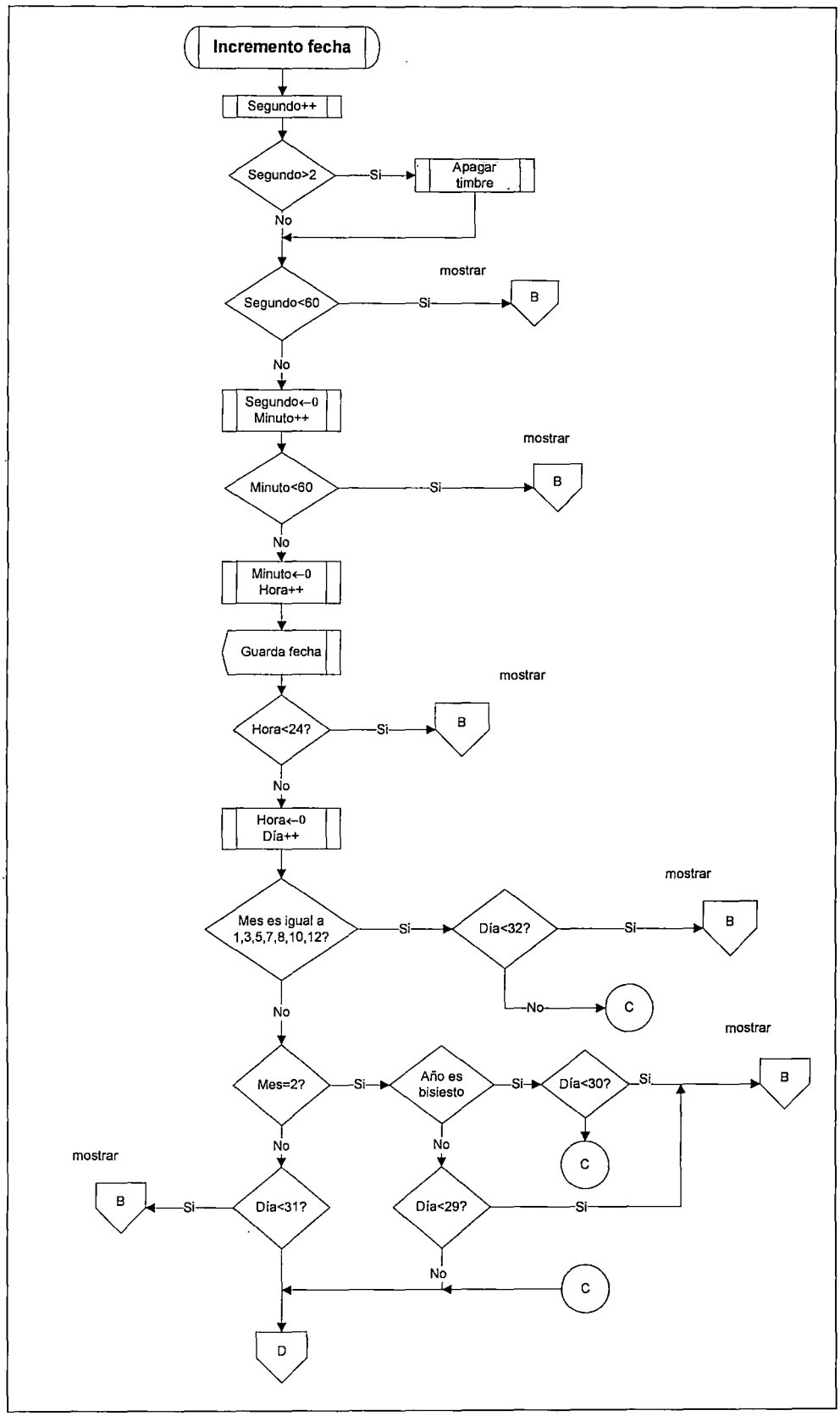



*Figura 3.69.- Diagrama de flujo para generar la fecha*

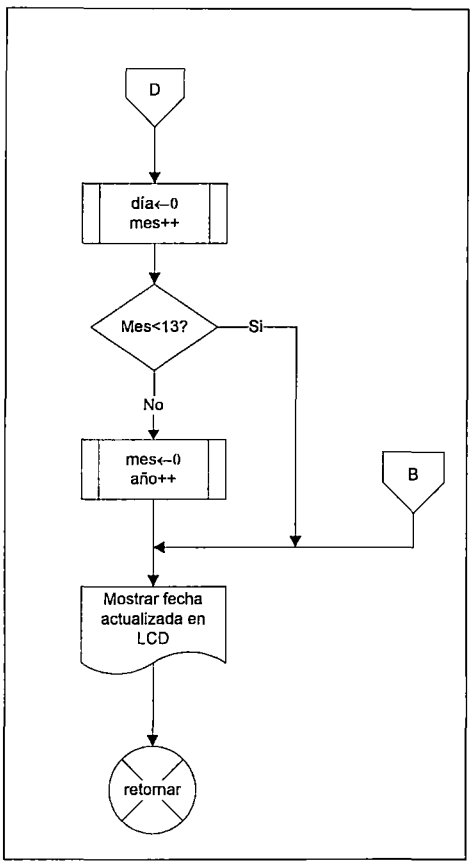



***Figura 3.70.-*** *Continuación de diagrama de flujo para generar la fecha*

**Menú principal**

Terminado los pasos anteriores el microcontrolador muestra en la pantalla la información como en la figura 3.71.

Internamente se tendrá el siguiente menú:

1. Cambiar fecha
2. Descargar
3. Supervisar en tiempo real
4. Supervisión de estado de sensores

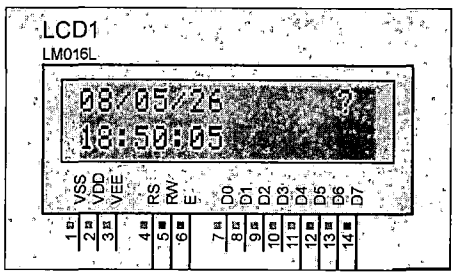



***Figura 3.71.-*** *Pantalla principal de la tarjeta de control*

El diagrama de flujo para el código de este menú en el microcontraldor se muestra en la figura 3.72. Esto significa que está esperando cualquier causa de interrupción para la cual fue programada y/o esperando si se ingresa alguna opción del menú por el puerto de comunicación desde PC.

**d) Almacenamiento de datos de vehículos**

Para esto se debe conocer cómo es que se acondicionará las señales enviadas desde la tarjeta acondicionadora de señal, el cual recibirá las señales de los sensores de presión y del sensor inductivo, que previamente pasa por otra tarjeta.

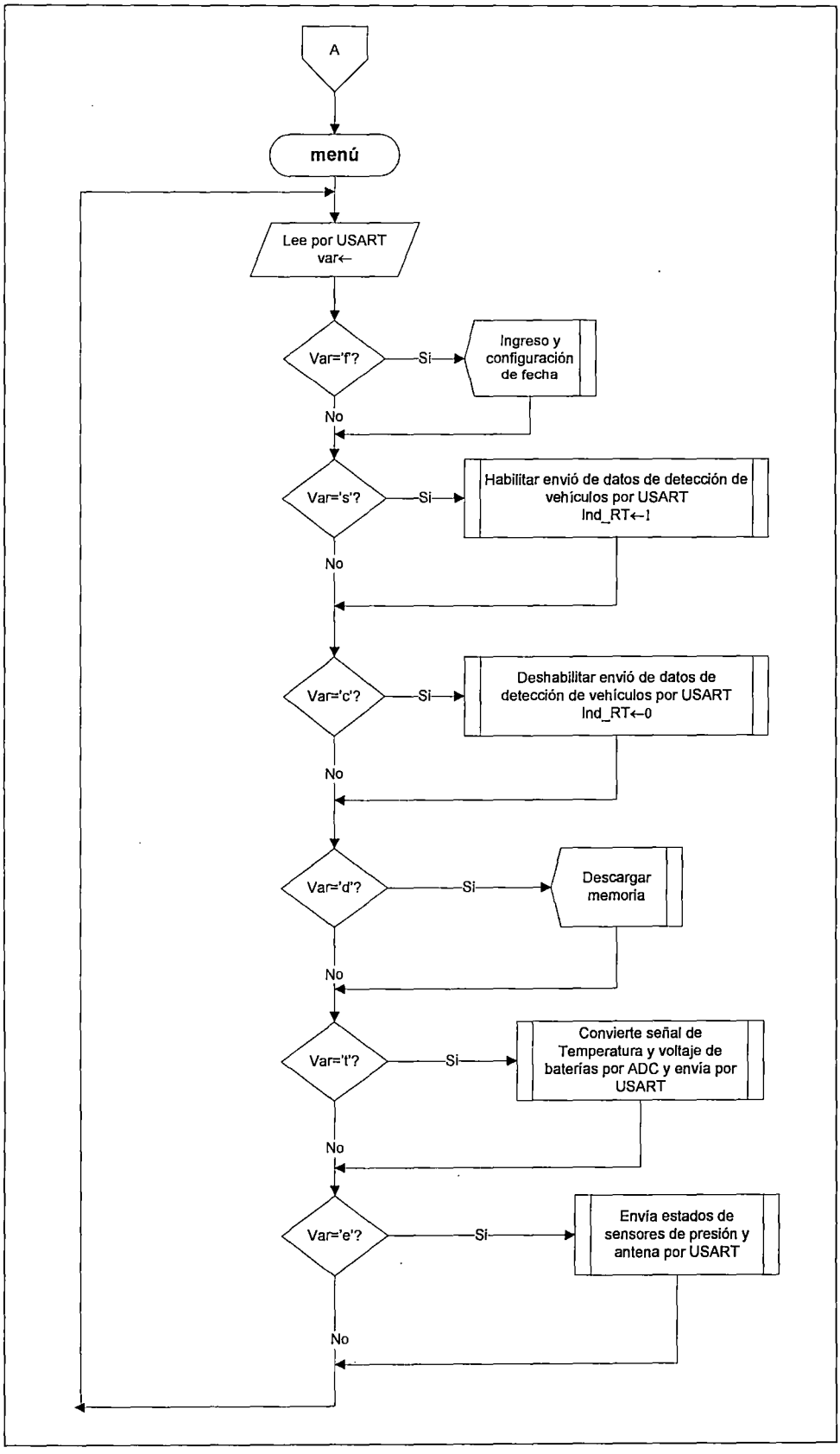



***Figura 3.72.-*** *Diagrama de flujo del menú interno del microcontrolador*

El microcontrolador de esta tarjeta finalmente recibe 3 tipos de señales, de acuerdo al diagrama temporal de la figura 3.73.

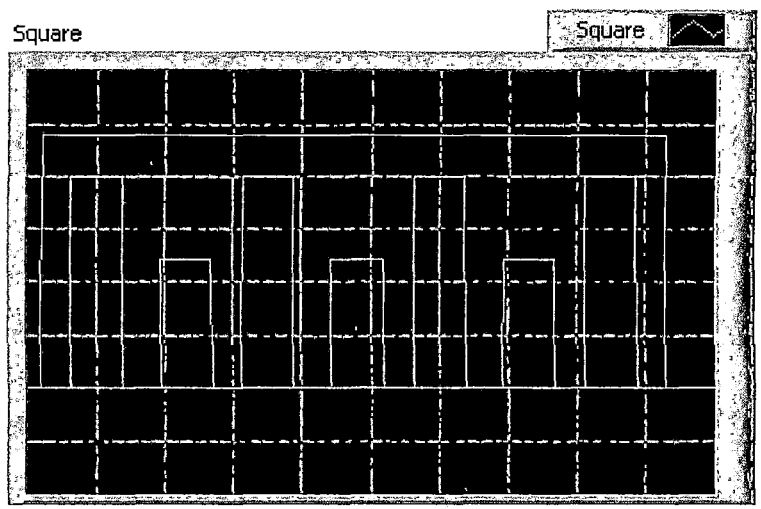



***Figura 3.73.-*** *Señales acondicionadas de los sensores*

La señal del sensor inductivo es la de color blanco, la señal roja es la del sensor de presión detector de ejes totales y la señal verde es la señal de detección de ejes duales, en caso de camiones.

El nivel de señales serán todas de 5V; pero en el esquema mostrado se mostraron con diferente amplitud todas con la intención de diferenciarlas claramente.

En muchos peajes tienen dos tipos de casetas para diferenciar el tipo de vehículos, es decir una caseta de autos y otra de camiones; pero como en la sierra del Perú la cantidad de vehículos que pasan no justifica tener estas 2

casetas, por eso es que se dispusieron los sensores de lo muestra la figura 3.74.

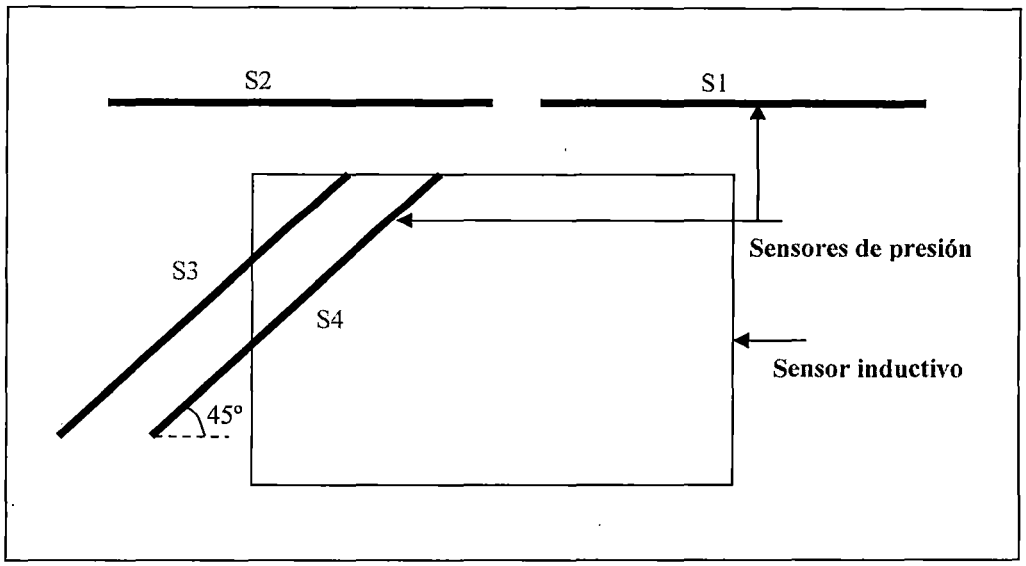



***Figura 3.74.-*** *Disposición de los sensores en la loza de peaje*

Esta es una vista horizontal de los sensores que se encuentran debajo de la pista por donde pasan los vehículos. Al pasar una unidad primero activa el sensor inductivo, señal que se mantendrá en alto todo el tiempo que permanezca allí; luego al seguir avanzando, las llantas pisarán uno a uno los sensores S1 y S2 simultáneamente ya que están unidos, por lo tanto estos detectarán la cantidad de ejes que tiene el vehículo. Los sensores S3 y S4 se encuentran formando un ángulo de 45° con la horizontal con la intención de que cuando pase un vehículo que tiene en la parte posterior llantas dobles, presionarán al mismo tiempo estos sensores, detectando así este tipo de ejes (de doble llanta), a estos ejes se le llaman ejes dobles y es característica de los camiones, de esta manera se puede diferenciar cuando pasan autos o

camiones. La disposición de los pines que se usaron del microcontrolador se puede apreciar en la figura 3.75

Esta disposición es necesaria ya que no solo es suficiente saber cuántos ejes tiene un vehículo sino si son dobles, ya que la tarifa es diferenciada.

También es necesario mencionar que la señal, ya acondicionada, no se permanece en alto aunque el vehículo siga presionando el sensor ya que todas las señales de los sensores de presión son pulsos y tienen el mismo ancho de tiempo.

Para que el microcontrolador pueda detectar el tipo de vehículo, estas señales de los sensores se dispondrán para el microcontrolador de la siguiente manera:

- Sensor de unidad. Permanece mientras permanece el vehículo.
- Sensor de ejes totales. Pulsos que cuenta ejes totales.
- Sensor de ejes duales. Pulsos que cuentan ejes dobles.

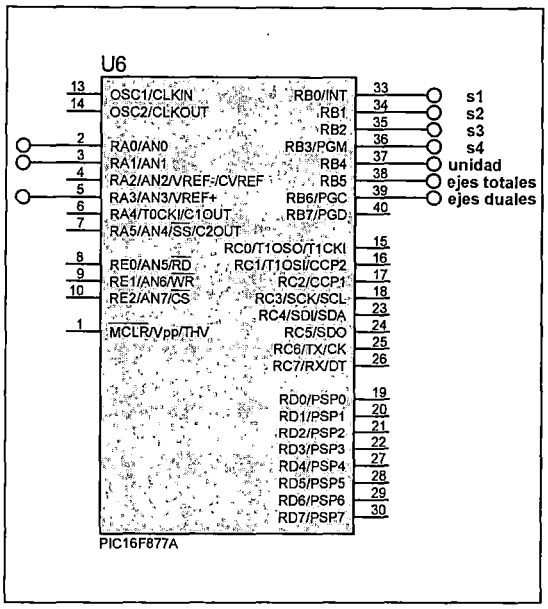



***Figura 3.75.-*** *Disposición de pines para los sensores de presión e inductivos*

Los pines usados para la detección de estas señales serán:

RB4: unidad

RB5: ejes totales

RB6: ejes duales

El microcontrolador no contará la cantidad de ejes mientras no se active el sensor de unidad, y al finalizar el paso del vehículo el PIC guardará en memoria el tipo de vehículo, la fecha y la hora a la que paso. Es decir si la señal no tiene la forma mostrada en la figura 3.73, la tarjeta acondicionadora de señal no enviará la señal a la tarjeta principal de control, ya que si no se da la secuencia correcta, es porque algo está mal. Para estas señales se habilitó la interrupción por cambio de estado que tiene los bits más significativos del puerto "B", estos pines son: RB4, RB5, RB6, RB7.

Se utilizó interrupciones, ya que existe la posibilidad de que mientras se está sacando información de las memorias, podría estar pasando un vehículo, y al estar habilitado el conteo por causa de interrupción, no interesa que operación esté realizando, esta se interrumpirá momentáneamente para atender el conteo y luego regresará ha continuar la operación que dejó momentáneamente. El diagrama de flujo para la atención del servicio de rutinas de interrupción se muestra en la figura 3.76.

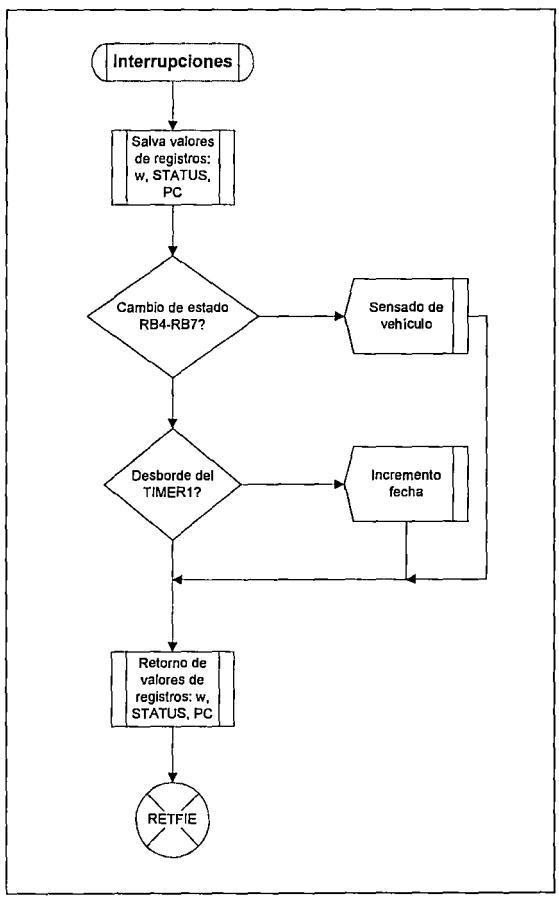



***Figura 3.76.-*** *Diagrama de flujo de las interrupciones: conteo de vehículos y generación de tiempo*

Cada vez que se detecte un vehículo estará esperando a que se detecte la cantidad de ejes totales y ejes duales, los cuales estarán siendo guardando en 2 variables. Una vez terminado el paso del vehículo, el microcontrolador clasificará para determinar si fue un auto o un camión y cuantos ejes tuvo, esto lo hace por el sensor dual de ejes, inmediatamente asociará un número secuencial para el tipo de vehículo que pasó.

***Tabla 3.7***
*(Números correlativos para cada uno de los tipos de unidades vehiculares)*

| ***Tipo de vehículo*** | ***Numeración*** |
|---|---|
| • Auto de 2 ejes | • 0 a 65535 |
| • Camión de 2 ejes | • 0 a 65535 |
| • Camión de 3 ejes | • 0 a 65535 |
| • Camión de 4 ejes | • 0 a 65535 |
| • Camión de 5 ejes | • 0 a 65535 |
| • Camión de 6 ejes | • 0 a 65535 |
| • Camión de 7 ejes | • 0 a 65535 |
| • Otros: auto 3 ejes, camión más de 7 ejes, auto de 1 ejes, etc. | • 0 a 65535 |

El diagrama de flujo de la forma como sensa las unidades vehiculares y las clasifica se muestra en la figura 3.77, 3.78 y 3.79.

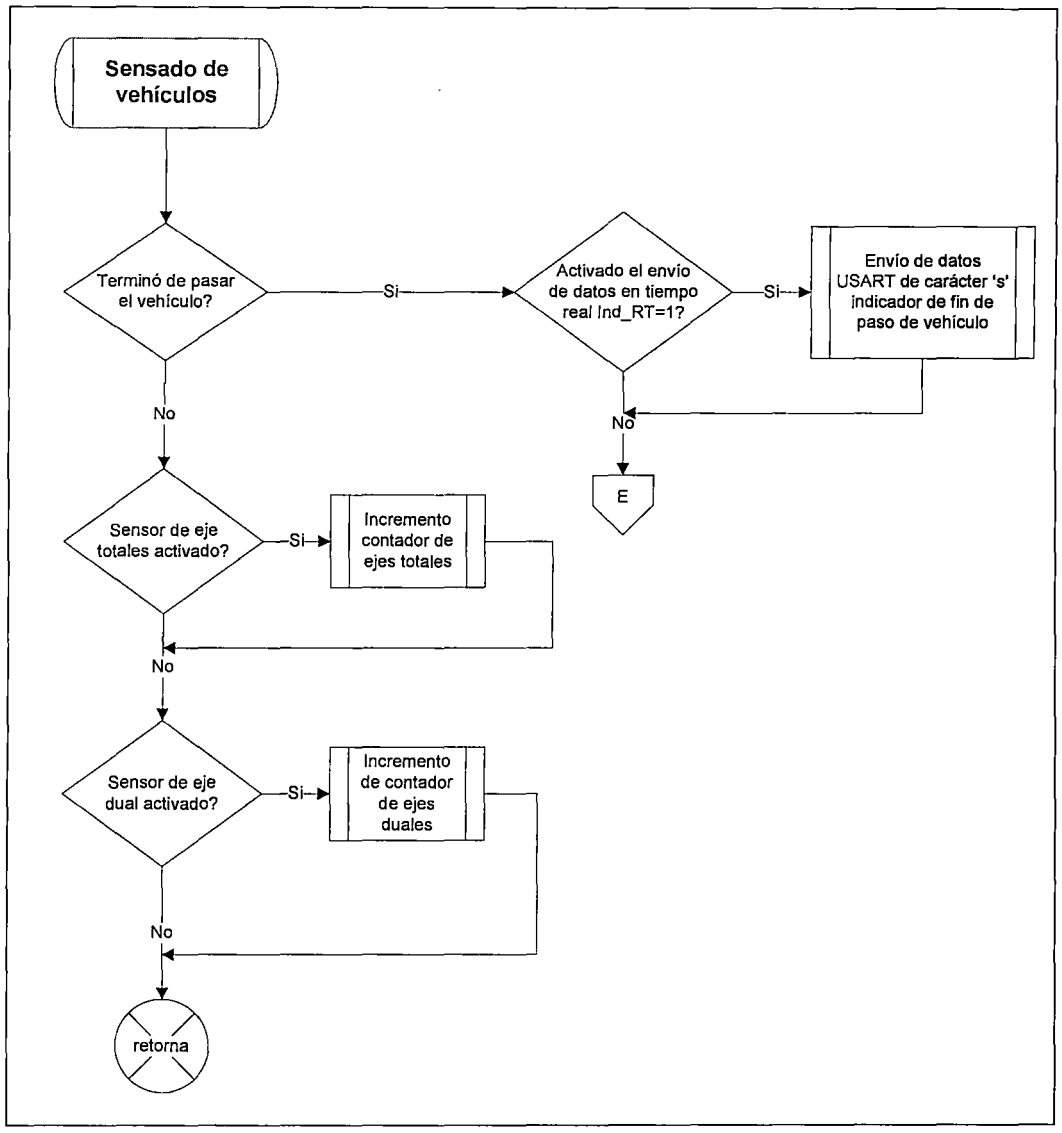



***Figura 3.77.-*** *Diagrama de flujo para el conteo y sensado de vehículos*

En la tabla 3.7 se puede apreciar que cada tipo de vehículo tiene una numeración propia, la cual podría variar de acuerdo a los requerimientos. La información de cada vehículo se guardará en las memorias EEPROM 24LCXXX externas al microcontrolador. La información asociada a cada vehículo se guardará en grupo de 8bytes, cuyo formato se muestra en la tabla 3.8.

*Tabla 3.8*
*(Descripción de la trama de datos guardados por vehículo)*

| *Minuto (1byte)* | *Segundo (1byte)* | *Tipo de unidad (1byte)* | *Numero de ejes (1byte)* | *Número de vehículo (2bytes)* | *Bytes libres (2bytes)* |
|---|---|---|---|---|---|
| Numero de minuto en que pasó el vehículo. Formato hexadecimal. | Número de segundo en que pasó el vehículo. Formato hexadecimal. | Es de tipo carácter, código ASCCI pudiendo ser: 'L': ligero, auto 'P': pesado, camión | Cantidad de ejes del vehículo en cuestión. Formato decimal. | Formato hexadecimal, número asociado a la cantidad de vehículo de ese tipo, estando comprendido desde 0 a 65535 (0000h-FFFFh) | Bytes libres para completar los 8bytes |

La razón por la cual se usaron grupos de 8bytes para guardar la información es para no estar preguntando cada vez que se guarda 1byte si ya se llenó una memoria para pasar a otra, ya que 8 es múltiplo de 16384bytes, 32768bytes o 65536bytes, por esta razón cada vez que se guarden 8bytes recién se preguntará si ya llegó al límite de la memoria.

Finalmente se guardará el siguiente número como marcador 08h. Esto servirá para búsqueda de información por PC a la hora que se crearán los reportes.

Inmediatamente se mostrará en la pantalla LCD la información de las características del vehículo que pasó. (Ver figura 3.80).

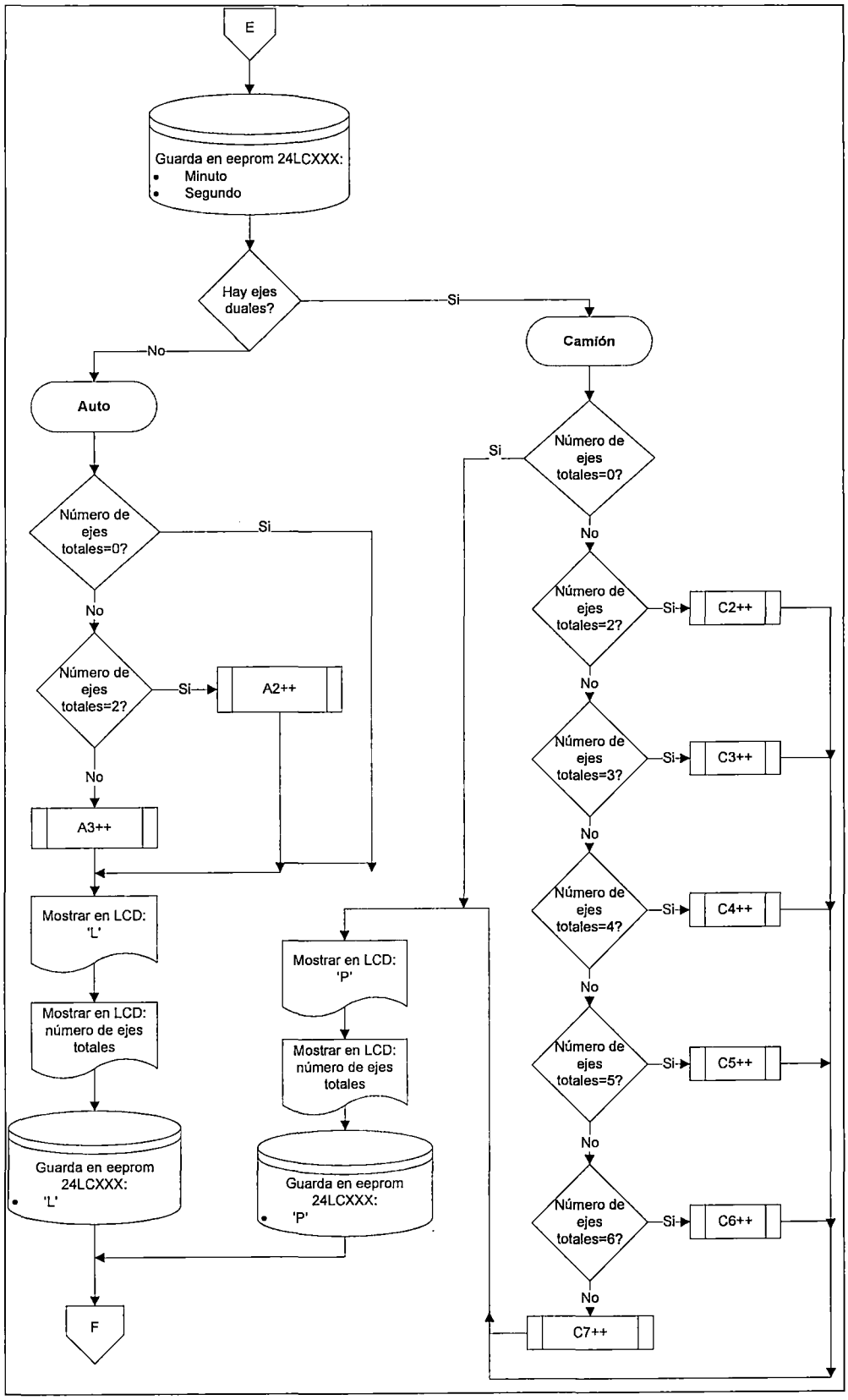



***Figura 3.78.-*** *Diagrama de flujo para la determinación del tipo de vehículo*

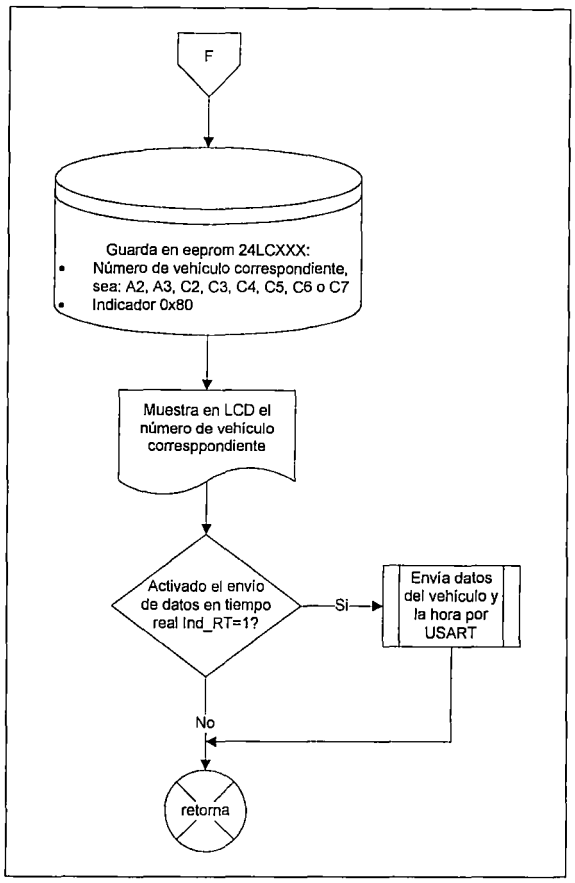



***Figura 3.79.-*** *Continuación del diagrama de flujo para la determinación del tipo de vehículo y almacenamiento*

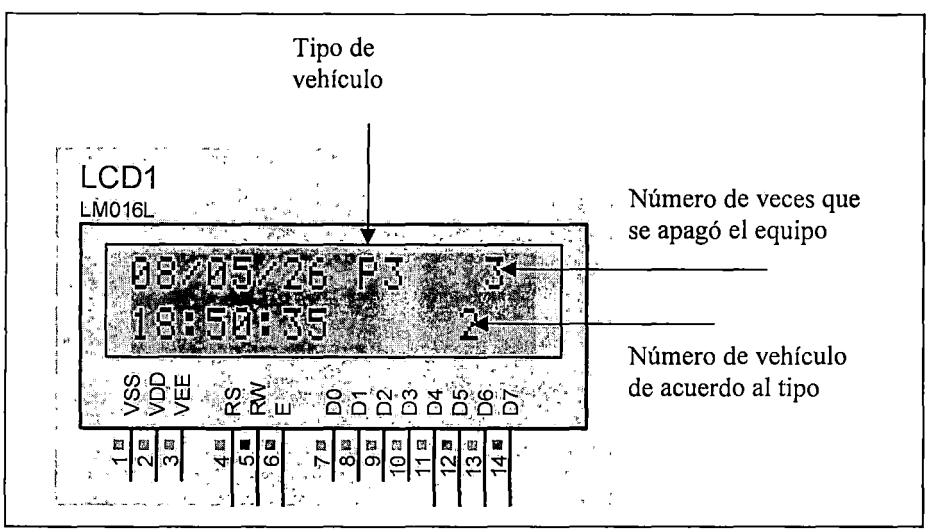



***Figura 3.80.-*** *Descripción de datos en pantalla LCD*

**Cambio de hora**

Cada vez que se cumpla una hora sucederá lo siguiente:

1) Se guardará la fecha.

2) Se guardará el número de veces que se hubiera apagado o reiniciado el equipo en esa hora.

3) Se guardará la posición de cada memoria así como en qué memoria estamos.

4) Se tocará un timbre anunciando cambio de hora o de turno durante 2 segundos.

El diagrama de flujo de todos estos eventos se observa en la figura 3.81.

**Guardado de la fecha.-** Esto se realizará igual en un grupo de 8bytes de acuerdo al siguiente formato:

***Tabla 3.9***

*(Trama de datos de fecha para almacenamiento en memorias I2C)*

| ***Señalizadores (4bytes)*** | ***Año (1byte)*** | ***Mes (1byte)*** | ***Día (1byte)*** | ***Hora (1byte)*** |
|---|---|---|---|---|
| Para la búsqueda de información a la hora de generar reportes. 4bytes iguales en formato hexadecimal 80h. | Formato hexadecimal | Formato hexadecimal | Formato hexadecimal | Formato hexadecimal |

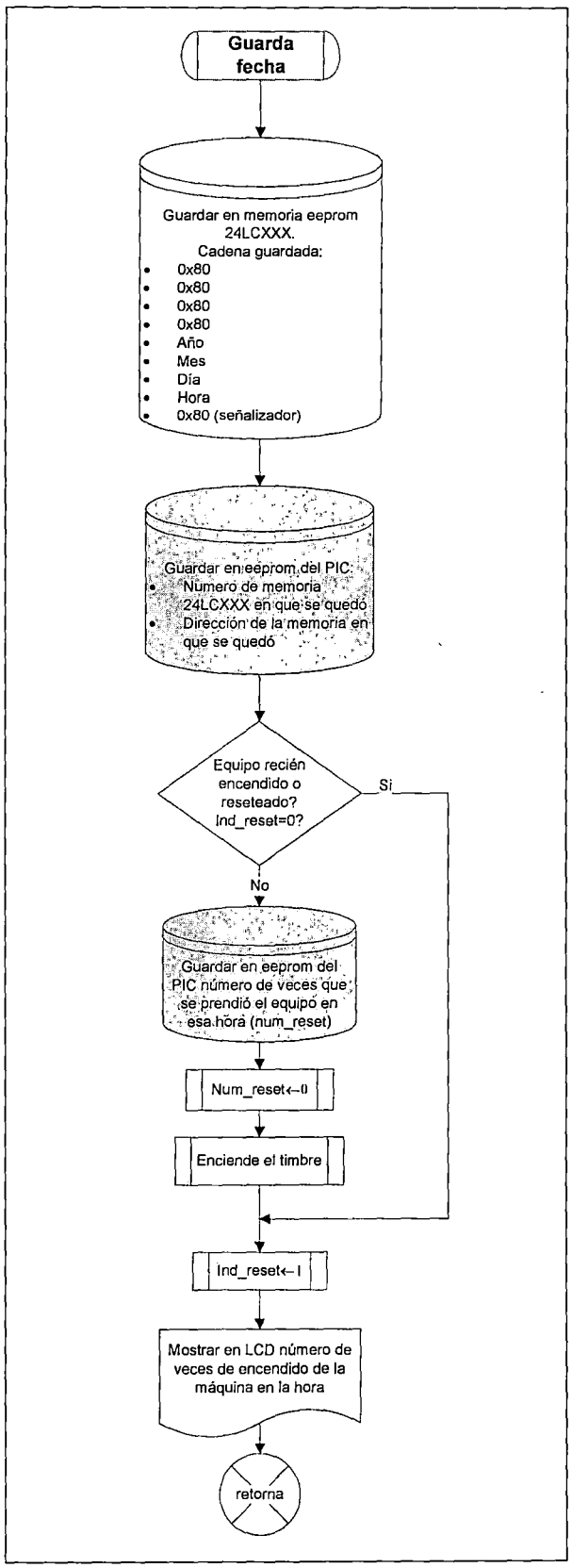



***Figura 3.81.-*** *Diagrama de flujo de eventos tras el cambio de hora*

**Guardado de evento de reset.-** Este dato también se guardará en grupo de 8bytes siendo simplemente el primero el que contiene la información, en formato hexadecimal, del número de veces que se reinició el equipo o que se apagó durante esa hora, los siguientes 7bytes no contienen información alguna. Se debe recordar que siempre que se guarda estos grupos de datos al final se guarda el señalizador 08h. Inmediatamente después la variable que tiene la cantidad de reset se volverá a cero, mostrándose cero en la pantalla LCD. A la hora de la descarga para los reportes, entonces se sabrá la cantidad de veces que se reinició el equipo y la hora en la que se produjo esto.

**Almacenamiento de puntero de dirección.-** será necesario almacenar en memoria EEPROM del PIC la dirección de las memorias EEPROM externas; es decir el número de la memoria y la dirección del registro en la que se encuentra. Si por cualquier razón se apaga completamente el equipo, al reiniciarse el sistema, no comenzará a guardar la información desde el inicio, borrando así los datos anteriores, sino desde la posición en que se quedó.

**Timbre.-** Cada hora, y por lo tanto cada cambio de turno siempre se tocará un timbre indicando este cambio, ya que muchas veces los cobradores cambian cada hora y/o turno.

**e) Descarga de datos guardados**

Recordemos el menú interno de microcontrolador:

1) Cambiar fecha

2) Descargar

3) Supervisar en tiempo real

4) Estado de sensores

Para diferentes tipos de peajes los momentos en que se descarga la información para los reportes suelen ser diferentes, siendo en algunos sitios cada turno, cada día, cada semana, etc. Esto significa que el equipo, dependiendo del promedio de la cantidad de vehículos que pasan al día tener memoria suficiente para tener la información hasta para un mes. De acuerdo a análisis estadísticos de los peajes en las zonas de la sierra tal como: Huacrapuquio, Chalhuapuquio, Tambogrande, Rumichaca, etc., al día pasan alrededor de 500 a 600 vehículos, este es un factor, junto con las fechas en las que se generan los reportes, para determinar memorias de qué capacidad y cuántas utilizar.

Trabajando con las memorias de mayor almacenamiento 24LC512, el cual su capacidad es de 65536 bytes, y sabiendo que la información por vehículo lo comprende 8bytes y usando la máxima capacidad de memorias (4) tendremos la siguiente capacidad:

$$\boxed{capacidad\ en\ días = \frac{n \times m}{8 \times v}.f} \quad \ldots\ldots\ldots\ldots \quad (3.15)$$

Donde:

n: número de memorias

m: capacidad de memoria en bytes

v: promedio de número de vehículos por día

f: factor de corrección

El factor de corrección está dado por la razón de que también se guarda la fecha y el número de veces de reiniciado el equipo, ocupando así un lugar en la memoria. Este factor experimentalmente esta alrededor de 0.91 a 0.95. Para la máxima capacidad tendremos:

$$\text{Capacidad en días} = \frac{4x65536}{8x600} \times 0.92$$

Capacidad en días = 50 días

Por lo tanto el equipo tendrá para aproximadamente 1mes y medio para guardar datos, sin correr el riesgo de que esta se pierda información.

Esta descarga se inicia al enviar un caracter 'd'. El diagrama de flujo del código para la descarga se muestra a continuación en la figura 3.82. El microcontrolador también devuelve el caracter 'd'. Luego se inician otras variables que van a contener la dirección y el número de memoria para leer las memorias, como existe la posibilidad que mientras se esté descargando pase un vehículo y se tenga que guardar la información por eso es necesario que las variables, tanto para lectura y escritura, sean diferentes.

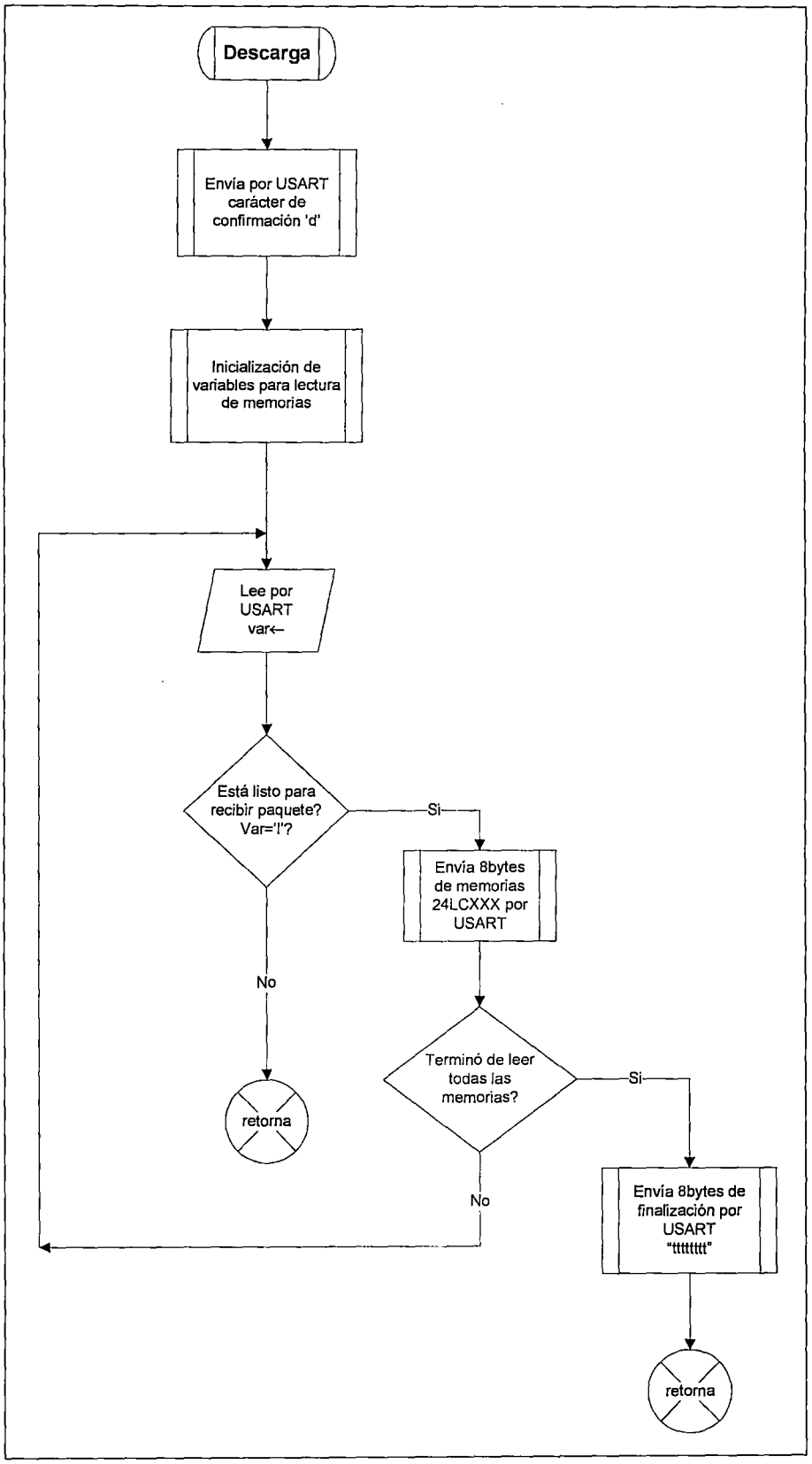



***Figura 3.82.-*** *Diagrama de algoritmo para la descarga de información a la PC*

La descarga se dará en grupos de 8bytes desde la dirección 0000h de la primera memoria hasta la última que fue configurada. Cada vez que reciba un caracter diferente de 'l' enviará el grupo de 8bytes, de lo contrario significa que algo falló y el microcontrolador regresará al menú principal, esto hasta la última dirección, y cuando halla finalizado de enviar todos los datos enviará una cadena de 8bytes de caracteres 't' indicando a la PC que ya terminó la descarga.

**f) Supervisión en tiempo real desde la PC**

Cuando el microcontrolador recibe el caracter 's' estará preparado para que además de guardar la información correspondiente a las características del vehículo que pasó, también enviará a la PC esta información, el número de veces que se reinició el equipo en esa hora y además en qué memoria está y en que posición se encuentra. Esta información se visualizará en la PC y será explicado más adelante. Siempre enviará los datos hasta que se envíe el carácter 'c' el cual deshabilitará esta función.

La rutina de interrupción para el conteo de vehículos se puede apreciar en la figura 3.83.

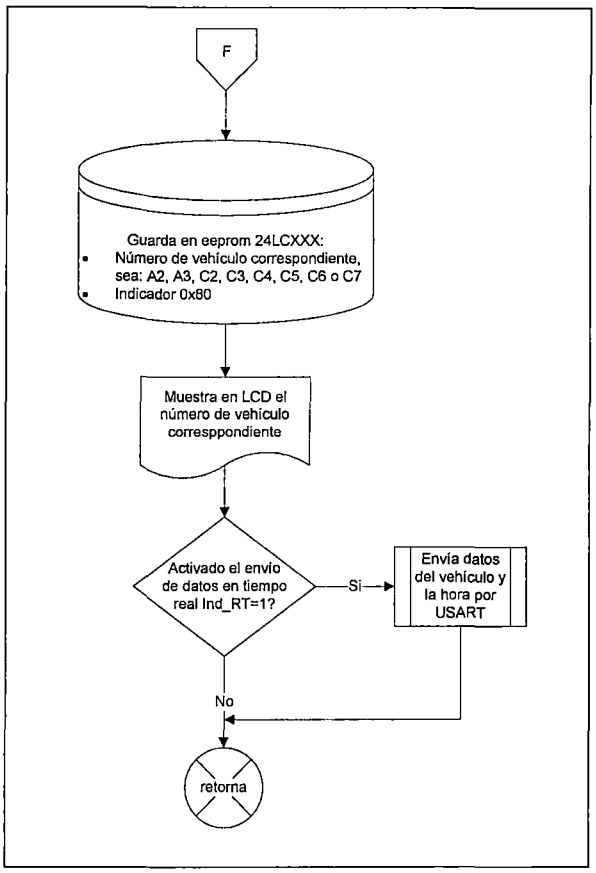



***Figura 3.83.-*** *Diagrama de flujo de almacenamiento de datos del vehículo sensado, para mostrarlo en pantalla LCD y envío a PC para supervisión RT*

Se puede observar que la variable "banderas" tiene un bit (el bit cero) que se llama "ind_RT", que si está en "1" la lectura en tiempo real estará habilitada, y si en "0" deshabilitada.

La trama de datos para cada vehículo es de la siguiente manera:

| *Año* | *Mes* | *Día* | *Hora* | *Minuto* | *Segundo* | *Unidad* | *número de ejes* | *número de vehículo* | *Número de memoria* | *Dirección dentro de memoria* |
|---|---|---|---|---|---|---|---|---|---|---|

- Año (1byte): en hexadecimal

- Mes (1byte): en hexadecimal
- Día (1byte): en hexadecimal
- Hora (1byte): en hexadecimal
- Minuto (1byte): en hexadecimal
- Segundo (1byte): en hexadecimal
- Unidad (1byte): formato carácter. 'L' para ligero o auto, 'P' para pesado o camión.
- Número de ejes (1byte): en hexadecimal
- Número de vehículo (2bytes): en hexadecimal
- Número de memoria (1byte): pudiendo ser A0h, A2h, A4h, A6h
- Dirección dentro de memoria (2bytes): depende de la capacidad de la memoria.

**g) Supervisión de Estado de sensores**

Siempre es importante saber el estado en el que se encuentra el tablero eléctrico, para esto se usaron algunos sensores para tomar medidas de algunos parámetros importantes:

- Sensor de temperatura.
- Sensor de nivel de voltaje de baterías.
- Sensor de puerta.

Para esto el microcontrolador dispone de un conversor análogo digital, del cual usaremos 2 canales para el sensor de temperatura y el voltaje de las

baterías. Se usará también una entrada digital más para saber el estado de la puerta del tablero eléctrico, es decir detectar si la puerta está abierta o cerrada.

Esta información cobra mayor importancia cuando se supervise mediante la red del internet desde la central en Lima el estado de todos los equipos instalados en la serranía del Perú en tiempo real. Esto se explicará con mayor detalle cuando se explique la propuesta del sistema interconectado diseñado.

En este caso se usarán el canal 0 (RA0) y el canal 1 (RA1), así como se usará también un pin para voltaje de referencia positivo (RA3) y así poder calibrar con mayor precisión y resolución las señales. El código en asembler de la configuración de conversor análogo digital se muestra a continuación:

```
adc_config
        BANCO1
        movlw b'00000101'
        movwf ADCON1
        BANCO0
        movlw b'10000001'
        andwf ADCON0,1
```

El microcontrolador enviará la información de temperatura, voltaje de baterías y estado de la puerta (3bytes) en cuanto cada vez que reciba el caracter 't'. También, con la intención de poder dar un mejor mantenimiento, se separaron 4 entradas digitales para que ingresen las señales de los sensores: inductivo y presión, sin pasar por la tarjeta acondicionadora de señal y así

verificar el estado de dichos sensores. Para esto el microcontrolador espera recibir el caracter 'e'.

Estas señales digitales usarán los bits menos significativos del puerto B como se aprecia en la figura 3.84.

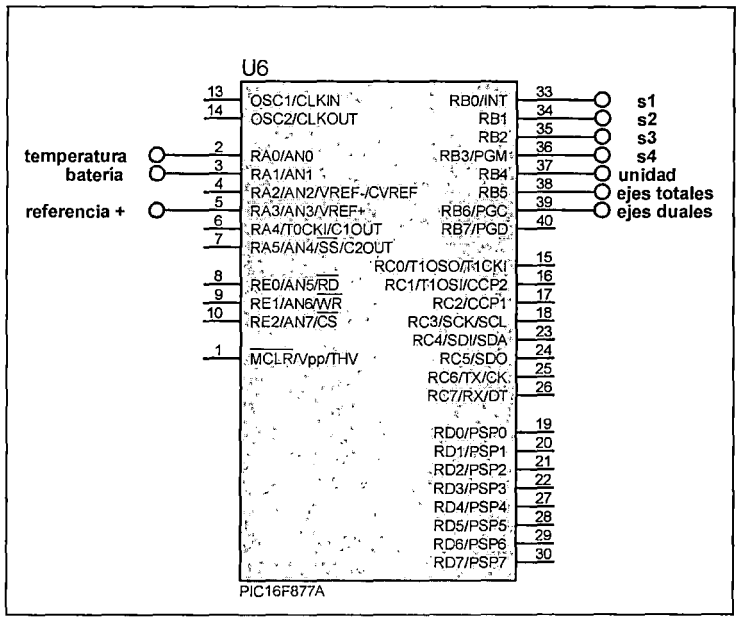



***Figura 3.84.-*** *Disposición de pines para los sensores de sensado de vehículos y sensores de temperatura, batería y estado de la puerta del tablero eléctrico de control*

S1: sensor inductivo de unidad

S2: sensor de presión de ejes totales

S3: sensor oblicuo 1

S4: sensor oblicuo 2

Cada vez que reciba ese caracter el microcontrolador enviará el estado del puerto B a la PC.

## 3.4. CÁLCULO MECÁNICO PARA EL DISEÑO DE SENSORES DE EJES DE PEANAS

Para detectar el eje de un vehículo se tienen diferentes sensores, siendo los más usados los sensores piezoeléctricos, de fibra óptica y mecánicos (de presión). El costo de sensores de fibra óptica son los más elevados; sin embargo ofrece mayor confiabilidad, vida útil larga y puede detectar vehículos que viajan altas velocidades. Los sensores piezoeléctricos también ofrecen un periodo de vida al igual que los de fibra óptica, esto es de alrededor de 5 años; este periodo de vida útil es mayor que los sensores mecánicos, sin embargo la diferencia con los sensores de fibra óptica es que estos pueden detectar vehículos en baja velocidad. Es razonable pensar que cuando un vehículo pasa extremadamente lento ambos sensores (los de fibra y los piezoeléctricos) podrían fallar; por esta razón se debe evitar que los vehículos se detengan cuando pasan por ellos. Los sensores de fibra y piezoeléctricos fueron probados en estos peajes y por alguna razón resultaron poco confiables en las condiciones de trabajo de estos peajes, ya que muchas veces detectaban ejes demás y ejes de menos. Cuando un sistema falla de esta manera, genera desconfianza, y ocurren irregularidades en el ajuste de cuentas; el cobrador (o alguna otra persona que desea extraer dinero ilícitamente) se justifica echando toda responsabilidad al sistema.

Los sensores mecánicos (de presión) brindaron mayor confiabilidad en la condiciones de trabajo de estos peajes, ya que incluso sensaban correctamente los ejes de vehículos aún cuando estos se detenían al pasar por ellos. Sin embargo, el

problema de estos sensores era su periodo de vida útil, siendo muy corta en comparación con los otros sensores. También, con estos sensores de presión, se puede de disponer de un arreglo para que el sistema sea capaz de clasificar (diferenciar) los vehículos, esto se vio a inicios del capítulo III (ver figura 3.10).

### 3.4.1. Objetivo

Se pretende mejorar el diseño de sensores de peanas, para la verificación del buen funcionamiento de estos sensores, así como mejorar su vida útil. Estos sensores trabajan como si fueran pulsadores, de manera que al ser presionados por las llantas de los vehículos, detectarán la presencia de un eje al superar un umbral de presión. De esta manera se puede conseguir un contador de ejes. A este sensor también lo llamaremos sensor de presión o sensor de ejes. Para este fin se hará uso de software CAD y CAE. Debemos recordar que CAD (Diseño Asistido por Ordenador) es un conjunto de Técnicas que permiten a los diseñadores, arquitectos, ingenieros, etc., utilizar en su trabajo herramientas informáticas para acortar los tiempos necesarios en el diseño de productos y CAE (Ingeniería asistido por computador) permite a los ingenieros simular en el ordenador los modelos que se piensan poner en práctica con el objetivo de apreciar su validez sin incurrir en costes de fabricación.

### 3.4.2. Alcance

Los sensores de peanas son fabricados por SIEMENS, algunas empresas colombianas y argentinas; sin embargo éstos sólo duran 2 a 3 meses

en las condiciones de trabajo de estos peajes, poco después empiezan a fallar. En base a las dimensiones exteriores que se fabrican estos sensores se mejorará el diseño para prolongar su vida útil.

### 3.4.3. Criterios de cálculo

#### A. Modelo analizado

Como se sabe, estos sensores trabajan como switch, los cuales se cortocircuitan cada vez que el eje de un móvil pasa. Se buscará las dimensiones adecuadas y el material de las dos planchas de metal que se ubican al interior del sensor. La parte exterior del sensor será de neopreno, de dimensiones iguales a las de los otros sensores. Estos sensores se ubicarán en la loza de concreto en unos soportes de acero, los cuales serán las restricciones en el análisis.

#### B. Hipótesis de Carga

La carga aplicada se considerarán estáticas y sólo debido al peso del automóvil en las llantas sobre el área en contacto, y al esfuerzo generado producto de la tracción.

#### C. Aplicación de Cargas

- Las cargas aplicadas serán sobre una de las caras del sensor, para esto se aplicará una fuerza a toda el área del sensor de tal manera que

genere el mismo esfuerzo que el que genera al aplicarse la fuerza sobre el área en contacto de las llantas.

- No se considerarán los esfuerzos térmicos.
- Tampoco se considerará el peso del sensor ya que el peso es despreciable en comparación con las cargas aplicadas debido al peso de los móviles.

**D. Cargas aplicadas**

- **Carga del peso de automóviles (P)**

  La fuerza normal está constituida por el peso del vehículo, ésta es asignada por el software de análisis sobre la superficie curva del sensor, la cual está hecho de neopreno.

- **Cargas de tracción (V)**

  La fuerza de tracción depende de la fricción y del peso del vehículo. Dicha fuerza será asignada tangencialmente, por el software, sobre la misma superficie que actúa el peso del vehículo. Para poder encontrar su valor, nos basaremos en resultados observados en ***"BANCOS DE POTENCIA (DINAMOMÉTROS) A RODILLOS PARA VEHÍCULOS CON PESO MÁXIMO DE 4000 kg POR EJE"*** (ver figura 3.1M), en el cual la fuerza de tracción es de 6kN. Luego la relación entre la fuerza de tracción y peso se puede apreciar en la expresión 3.1M. Usaremos esta relación para el cálculo de la fuerza de tracción en el modelo que se analizará en el software.

$$\frac{F.Tracción}{F.Peso} = \frac{3}{20} \text{.....} (3.1M)$$

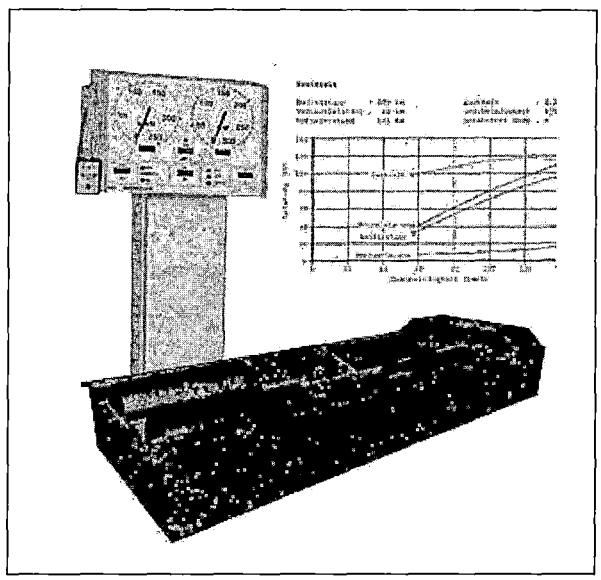


***Figura 3.1M.-*** *Banco de potencia a rodillos para medir la tracción de vehículos*

### 3.4.4. Combinación de cargas

Las combinaciones de cargas aplicadas para el cálculo es la siguiente: Peso de automóvil + fuerza de tracción, éstos vectores se muestran en la figura 3.2M.

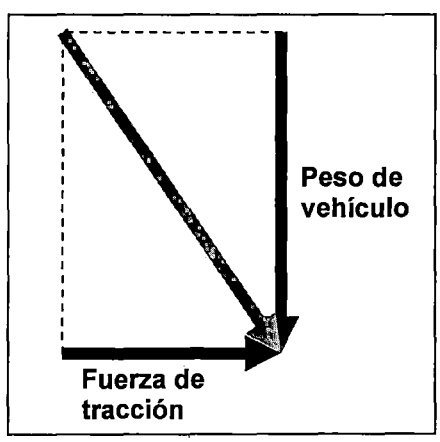



***Figura 3.2M.-*** *Vectores fuerzas debido al paso de vehículos*

### 3.4.5. Evaluación de cargas internas

Para el cálculo de las cargas internas sobre los elementos se ha utilizado el programa "Cosmos DesignStar", en el cual se han modelado en ensamble de partes.

El modelo ha sido hecho con las dimensiones en milímetros y las cargas en kilogramos fuerza y todas las cargas internas se evalúan aplicando cargas externas factoradas.

### 3.4.6. Verificación de elementos

Como no existen normas para la construcción de este tipo de sensores, sólo nos apoyaremos en los siguientes enunciados:

- El esfuerzo aplicado por los automóviles deben ser lo suficiente como para poder conseguir que el sensor se active (las láminas metálicas entren en contacto); sino sucede esto, diremos que al sensor le falta sensibilidad o que "está duro".
- El esfuerzo no debe ocasionar una deformación permanente de las láminas (que las láminas quedan en contacto permanente), de lo contrario el sensor siempre estaría activado.
- Tampoco es bueno de que sea tan sensible que se active con el peso de una persona.

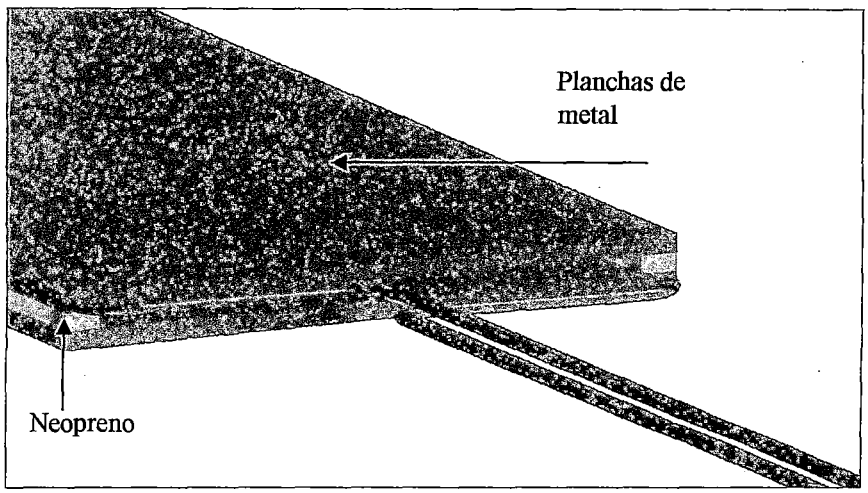



***Figura 3.3M.-*** *Planchas de acero separadas del sensor de presión, para la determinación de ejes de vehículos*

## A. Datos técnicos

El material del cual se diseño el sensor será de neopreno con unas láminas de metal en su interior. En primer lugar se mostrará las cargas aproximadas de los vehículos:

**Camiones**

Peso de un camión con tres ejes =25 a 33.5 toneladas

Número de llantas = 10

Área de contacto de la llanta = (15cm x 21cm) a (15cm x 26cm)

***Autos***

Peso = 900kgf a 1800kgf

Número de llantas = 4

Área de contacto de la llanta = 15cm x 12cm

$$Esfuerzo = \frac{Peso}{(Número\ llantas)x(Area\ llanta)} \quad .....(3.2M)$$

$$Esfuerzo\ máximo = \frac{33500kgf}{(10)x(15x21)cm^2} = 10.635\frac{kgf}{cm^2}$$

$$Esfuerzo\ mínimo = \frac{900kgf}{(4)x(15x12)cm^2} = 1.25\frac{kgf}{cm^2}$$

## B. Diseño básico de las partes

Como se puede apreciar todo parte de un diseño básico, es decir se toma como referencia medidas externas estándares de otros sensores. También el material exterior será neopreno. El ensamblado fue hecho en INVENTOR y tendrá la forma como se muestra en la figura 3.4M.

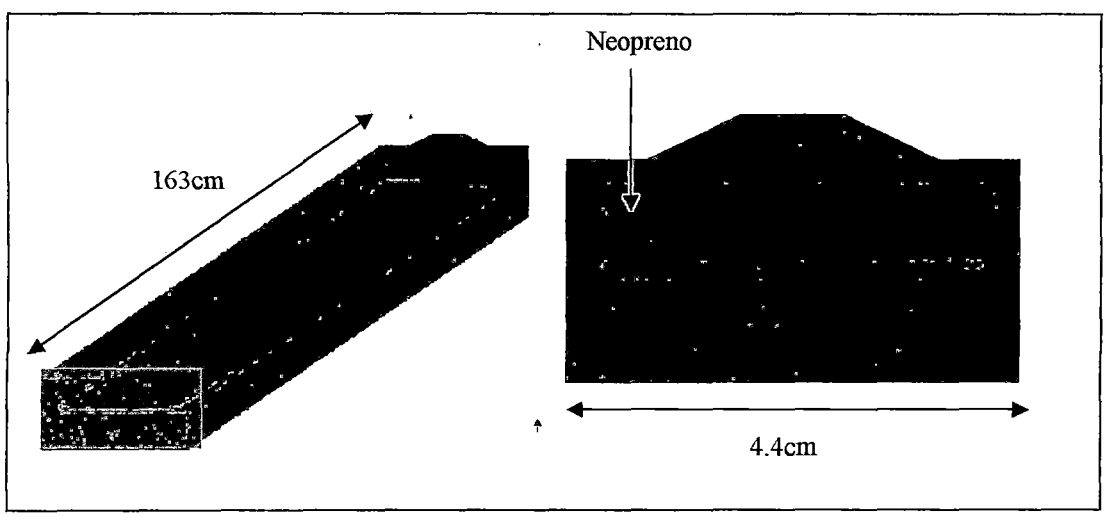



***Figura 3.4M.-*** *Sensores de presión, para la detección de ejes*

Se determinará cuál debe ser el material y dimensiones adecuadas de las placas metálicas que van paralelas dentro del caucho. El sensor dispone de

una porción de cilindro de caucho que sobresaldrá del nivel de la pista, para que de esta manera se pueda transmitir la carga aplicada a las placas de metal. La superficie de la pista estará al mismo nivel del sensor como se muestra en el gráfico de la figura 3.5M.

## C. Cálculo de las fuerzas máximas y mínimas aplicadas

Las dimensiones exteriores del sensor de presión, hecho de caucho neopreno, son de 4.4cm de ancho por 163cm de largo como lo muestra la figura 3.4M. Entonces la mejora del diseño de este sensor será sobre las láminas de metal. Para esto, solo por simplicidad en el uso del software de cálculo elegido, supondremos que la fuerza se distribuye en toda el área del sensor, es decir: Área = 4.4cm x 163cm = 717.2cm$^2$

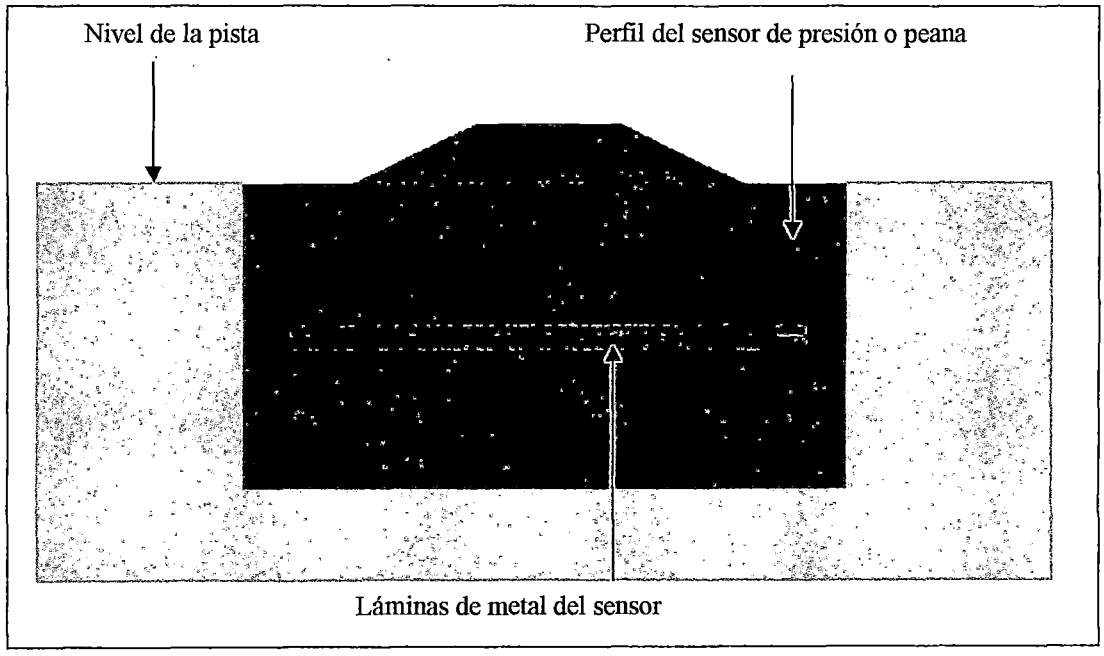



***Figura 3.5M.-*** *Vista de perfil del sensor de presión en la pista*

De acuerdo a las presiones ejercidas por los autos y camiones, tendremos la fuerza máxima y la mínima generada por camiones y autos:

Para camiones, las componentes de la fuerza máxima serán:

$$\text{Fnormal} = 10.635x717.2 = 7627.422\ kgf$$

$$\text{Ftangente} = 7627.422x\frac{3}{20} = 1144.1133\ kgf$$

La fuerza mínima es dada por los autos, cuyas componentes serán:

$$\text{Fnormal} = 1.25x717.2 = 896.5kgf$$

$$\text{Ftangente} = 896.5x\frac{3}{20} = 134.475kgf$$

Sin embargo, no toda la fuerza actúa sobre esa área sino sólo sobre una parte del área (ver figura 3.6M), el cual es diferente para autos o camiones. Se determinó experimentalmente el porcentaje de área sobre el cual actúa esta fuerza, obteniéndose para esto los siguientes factores de corrección:

- Autos: factor de corrección = 0.25
- Camiones: factor de corrección = 0.5

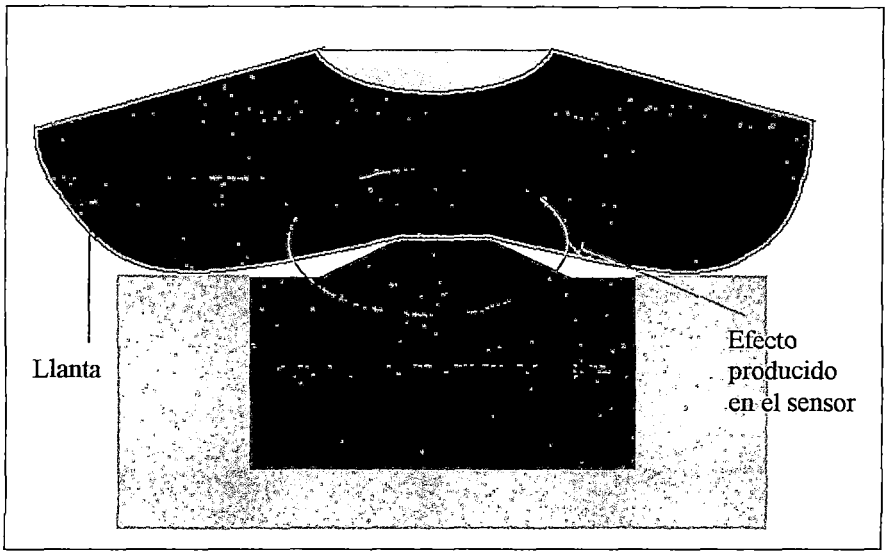



***Figura 3.6M.-*** *Sensor de presión durante el paso de un vehículo*

De acuerdo a esto el esfuerzo normal y tangencial se verá incrementado y por consiguiente la fuerza.

Para camiones, las componentes corregidas de la fuerza máxima serán:

$$\text{Fnormal} = \frac{7627.422}{0.5} = 15254.844 = 15260 kgf$$

$$\text{Ftangente} = \frac{1144.1133}{0.5} = 2288.2266 = 2290 kgf$$

Las componentes de fuerza mínima corregidas de los autos serán:

$$\text{Fnormal} = \frac{896.5}{0.25} = 3586 = 3580 kgf$$

$$\text{Ftangente} = \frac{134.475}{0.25} = 536.9 = 530 kgf$$

## D. Análisis de casos por medio de software de elementos finitos

Para el análisis se usará el software de cálculo de elementos finitos por lo tanto con esto se trabajará con el software **Cosmos DesignStar**, con los datos obtenidos en los cálculos y consideraciones anteriores. Podemos apreciar en la figura 3.7M y 3.8M el entorno de trabajo del software así como el CAD del sensor de presión en el que se aplican las fuerzas.

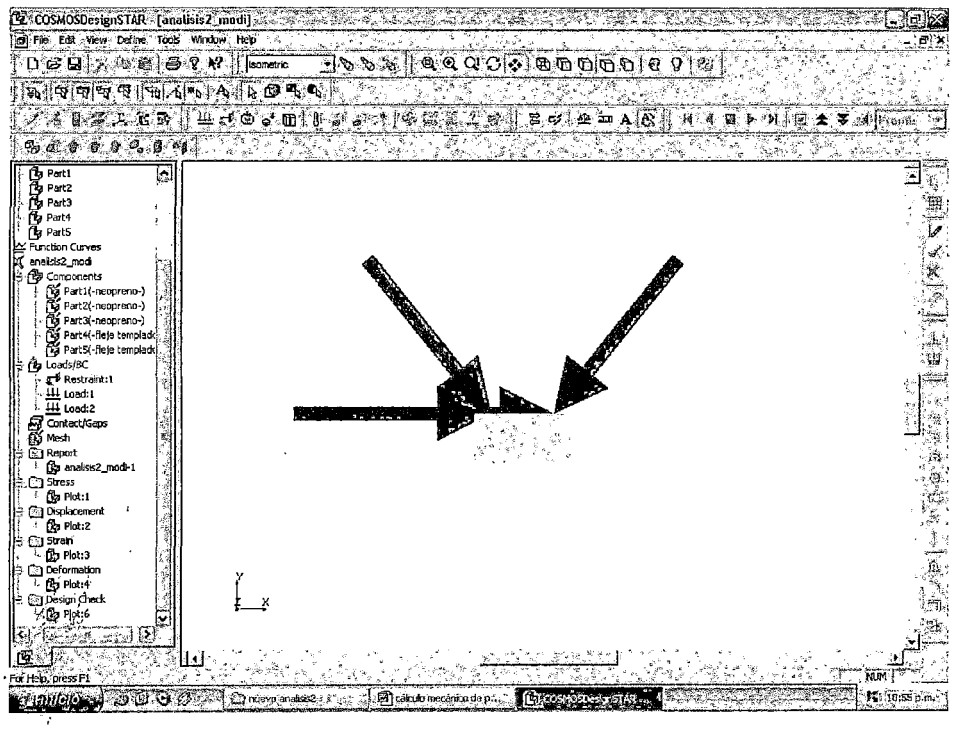



***Figura 3.7M.-*** *Representación de cargas en software de análisis por elementos finitos Cosmos DesignStar*

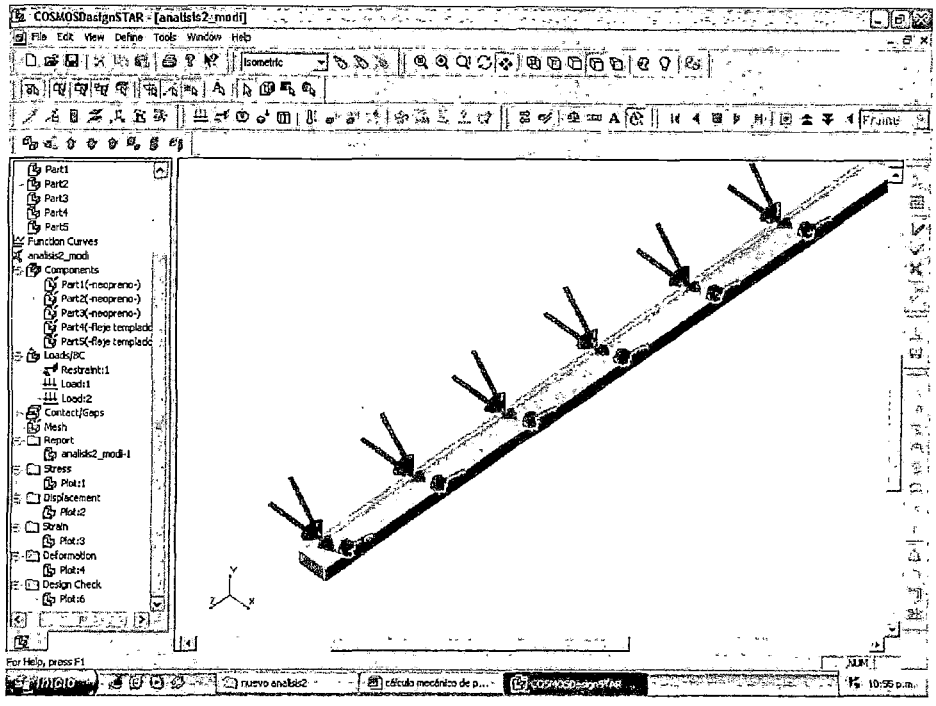



***Figura 3.8M.-*** *Representación de cargas en software de análisis por elementos finitos Cosmos DesignStar, vista isométrica*

Las dimensiones exteriores del sensor se sacaron en base a sensores con medidas estándares usados como se mencionó anteriormente, de tal manera que para el cliente sea fácil encontrar repuesto. Las dimensiones internas se tuvieron que analizar para diferentes medidas. La forma de las láminas internas de metal se muestra en la figura 3.9M.

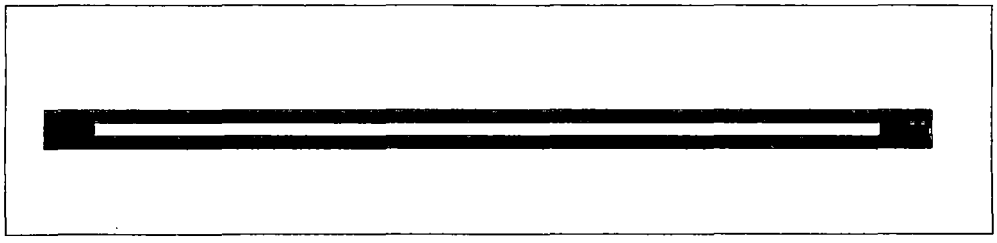


***Figura 3.9M.-*** *Vista de perfil de las planchas de acero que detectarán por contacto la presencia de un eje*

Se apreciará además que los máximos esfuerzos y deformaciones estarán concentrados en estas dos placas para cualquiera de los casos que analicemos. Como se puede ver en los gráficos de las figuras 3.10M y 3.11M, los colores demuestran esto. Los análisis que se hicieron es sobre todo el sensor, es decir, las fuerzas se aplicaron sobre el caucho de neopreno y las placas de metal. Las restricciones serán en todas las caras menos por la cara superior que es por donde pasa las llantas de los vehículos, estas serán como consecuencia de que se acomodarán esto sensores en unas bases metálicas muy resistentes en la loza de concreto. Estas restricciones se pueden apreciar como flechas de color verde en el gráfico de la figura 3.12M.

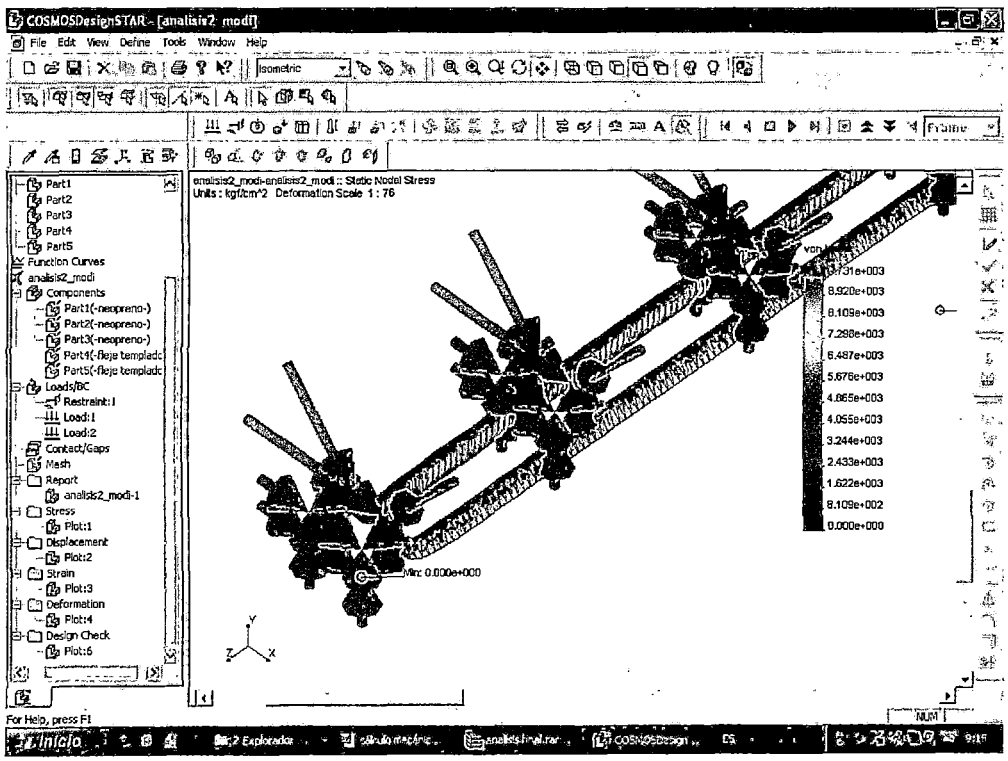



***Figura 3.10M.-*** *Vista de cargas, restricciones y esfuerzos sobre los separadores de las planchas metálicas*

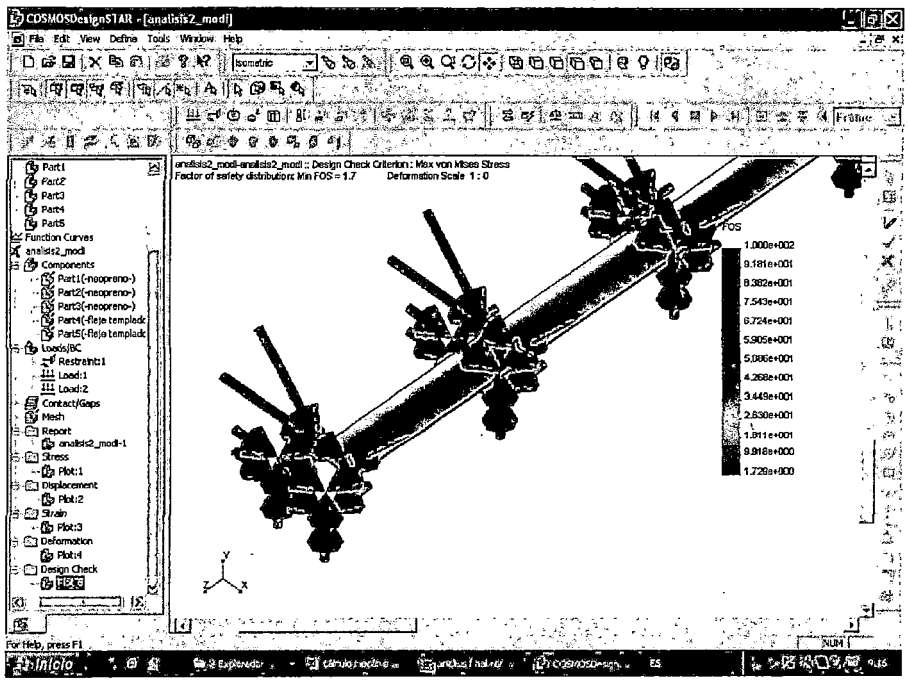



***Figura 3.11M.-*** *Vista de cargas, restricciones y esfuerzos sobre las planchas metálicas*

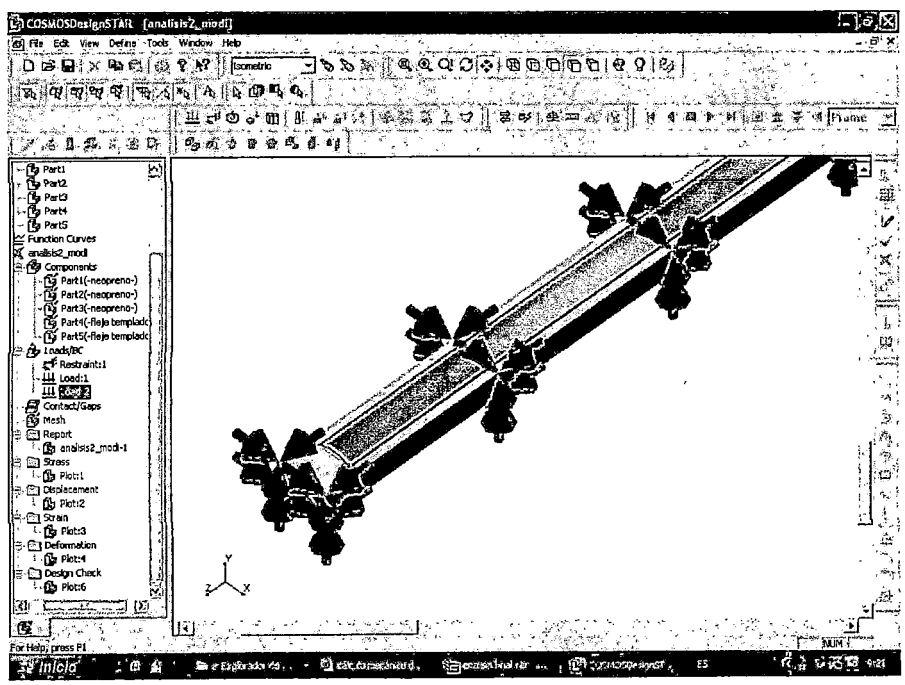



***Figura 3.12M.-*** *Representación de fuerzas para las restricciones en el análisis estático*

Para llegar a la conclusión sobre el tipo de material que se usará y las dimensiones de las planchas metálicas se hicieron diversas pruebas, de las cuales mostraremos un resumen especificada en 4 casos.

**Caso1**

Primero se partió de planchas de 1mm de espesor y el neopreno que se usa de separación de las placas será también de 1mm, y el ancho de las placas será de 31mm. (Ver figura 3.13M).

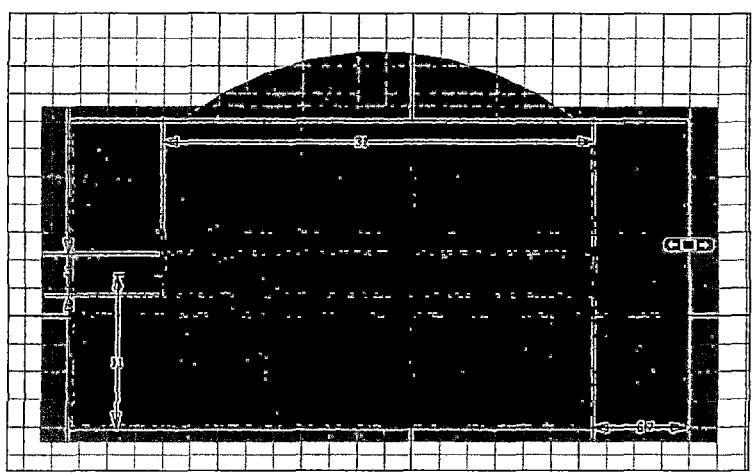


***Figura 3.13M.-*** *Perfil del caso 1*

Aplicaremos sobre el sensor la carga máxima, cuyas componentes son:

Fnormal =15260kgf

Ftangencial =2290kgf

Se puede apreciar que, de acuerdo al esfuerzo aplicando en todo el área, tendríamos que buscar un material que supere el esfuerzo de 8055 $kgf/cm^2$ indicado en la figura 3.14M.

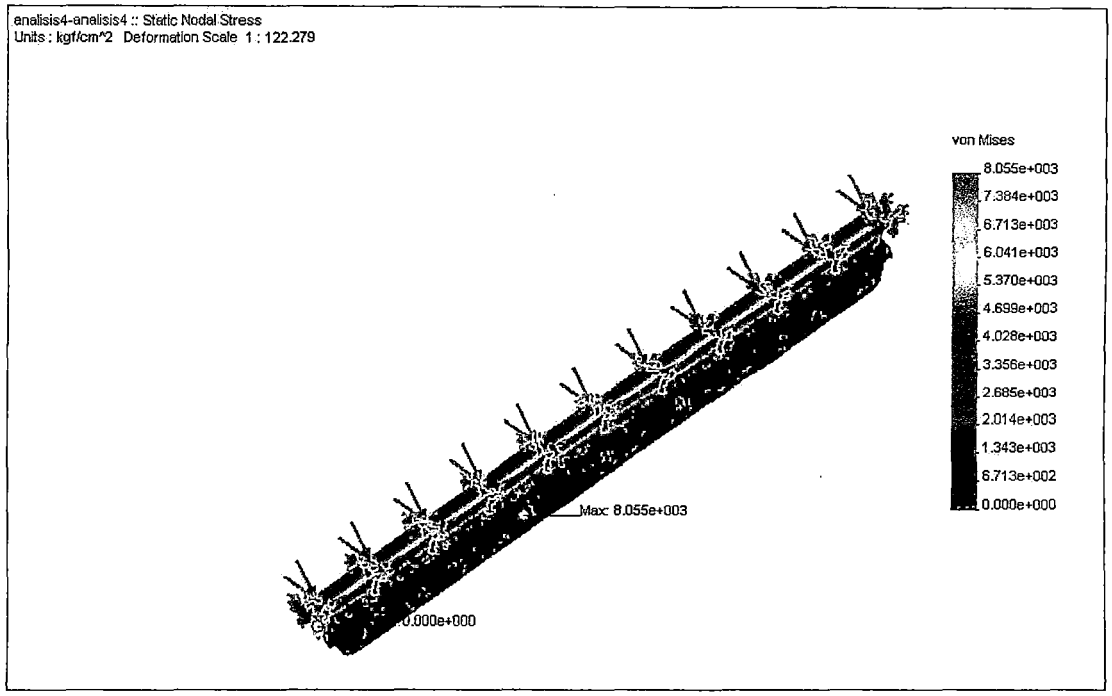



***Figura 3.14M.-*** *Esfuerzos en las planchas metálicas para el caso 1, debido a las fuerzas máximas*

En cuanto al desplazamiento, podemos apreciar en la figura 3.15M, que toma su máximo valor de 1.354mm en el centro de las placas, y como la separación de éstas placas es de 1mm, entonces podemos decir que el sensor puede detectar esta carga ya que las placas entran en contacto.

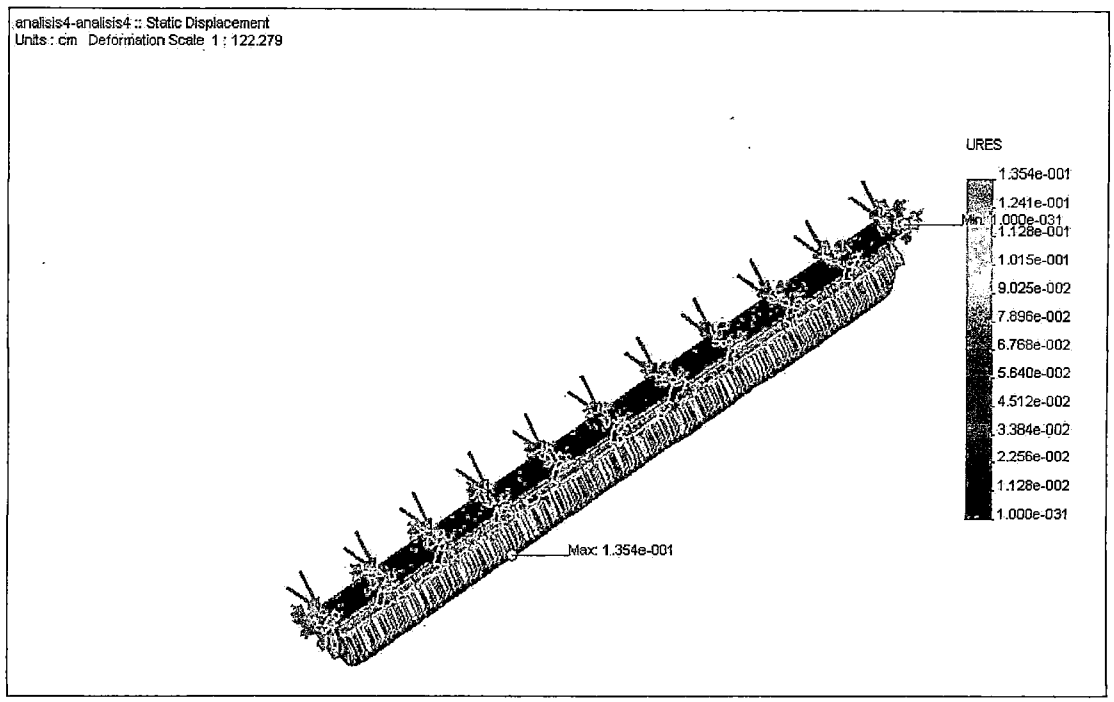



***Figura 3.15M.-*** *Desplazamientos en las planchas metálicas para el caso 1, debido a las fuerzas máximas*

Aplicaremos ahora en el sensor la carga mínima, cuyas componentes son:

Fnormal =3580kgf

Ftangencial =530kgf

Luego para esto tendremos los resultados de esfuerzos y desplazamiento en las figuras 3.16M y 3.17M.

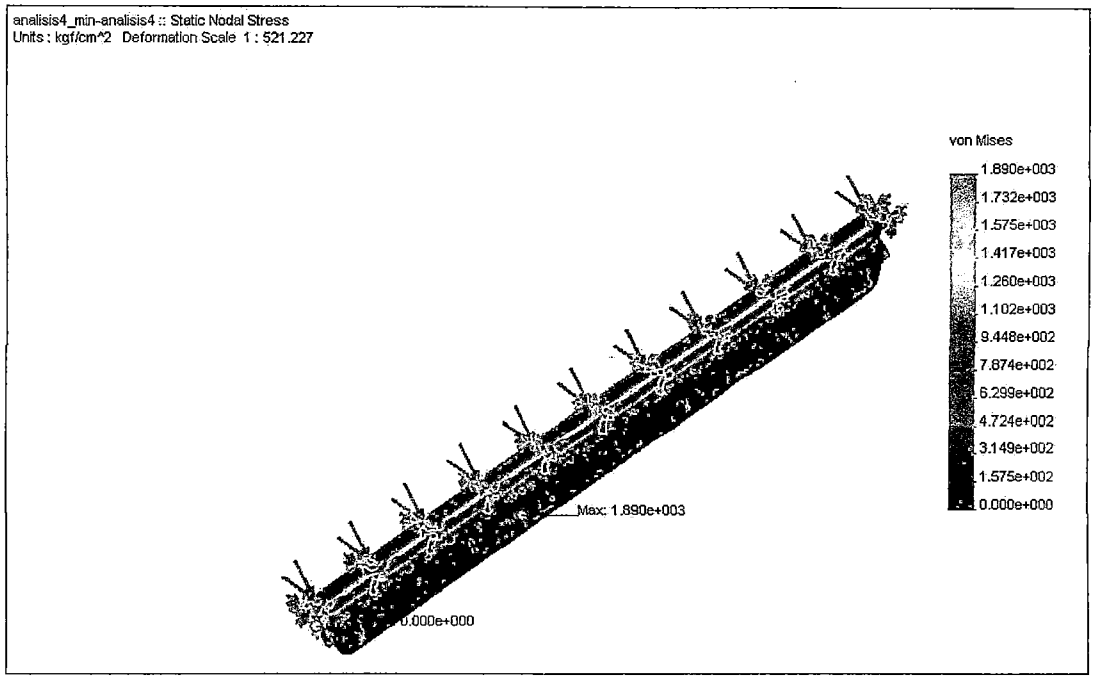



***Figura 3.16M.-*** *Esfuerzos en las planchas metálicas para el caso 1, debido a las fuerzas mínimas*

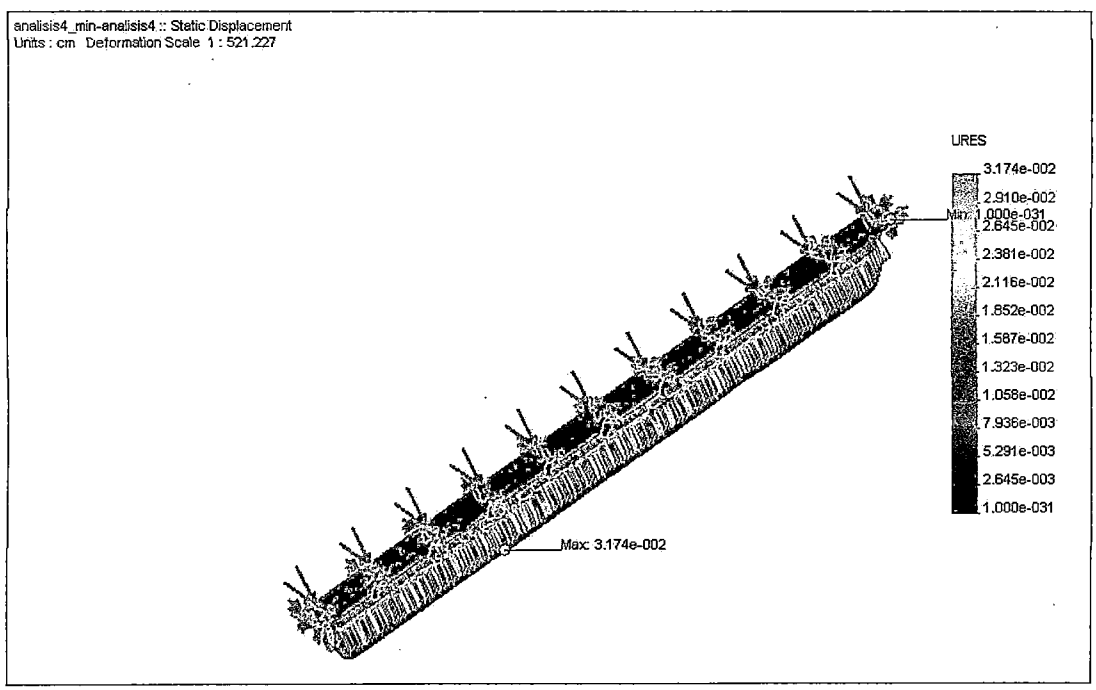



***Figura 3.17M.-*** *Desplazamientos en las planchas metálicas para el caso 1, debido a las fuerzas mínimas*

Como es de suponer el esfuerzo máximo es menor; sin embargo el desplazamiento no supera la separación de las placas de 1mm, solo llega a 0.317mm, lo cual significaría que las placas de metal no se juntarían, por lo que el sensor no conseguiría detectar esta fuerza.

***Tabla 3.1M***

*Propiedades de principales materiales*

| N° SAE o AISI | Resistencia a la tracción Rm | | Límite de fluencia Re | | Alargamiento en 50 mm | Dureza Brinell |
|---|---|---|---|---|---|---|
| | Kgf / mm2 | Mpa | Kgf/mm2 | Mpa | % | |
| 1010 | 40,0 | 392,3 | 30,2 | 292,2 | 39 | 109 |
| 1015 | 42,9 | 420,7 | 32,0 | 313,8 | 39 | 126 |
| 1020 | 45,8 | 449,1 | 33,8 | 331,5 | 36 | 143 |
| 1025 | 50,1 | 491,3 | 34,5 | 338,3 | 34 | 161 |
| 1030 | 56,3 | 552,1 | 35,2 | 345,2 | 32 | 179 |
| 1035 | 59,8 | 586,4 | 38,7 | 377,5 | 29 | 190 |
| 1040 | 63,4 | 621,7 | 42,2 | 413,8 | 25 | 201 |
| 1045 | 68,7 | 673,7 | 42,2 | 413,8 | 23 | 215 |
| 1050 | 73,9 | 724,7 | 42,2 | 413,8 | 20 | 229 |
| 1055 | 78,5 | 769,8 | 45,8 | 449,1 | 19 | 235 |
| 1060 | 83,1 | 814,9 | 49,3 | 483,5 | 17 | 241 |
| 1065 | 87,0 | 853,2 | 51,9 | 509,0 | 16 | 254 |
| 1070 | 90,9 | 891,4 | 54,6 | 535,4 | 15 | 267 |
| 1075 | 94,7 | 928,7 | 57,3 | 560,9 | 13 | 280 |
| 1080 | 98,6 | 966,9 | 59,8 | 586,4 | 12 | 293 |

Todas las pruebas de cálculo fueron hechas primero con aceros conocidos como AISI 304, 1060, 1070, 1075, (ver tabla 3.1M) sin embargo cuando eran sometidos a los máximos esfuerzos, estos sobrepasaban la zona elástica, es decir, fallaban y quedaban deformados permanentemente. Por esta razón es que se opto por un acero templado para mejorar su resistencia mecánica, AISI 1075. En la tabla 3.1M se observa que la resistencia mecánica

es de 9860kgf/cm$^2$, superando los 8055 kgf/cm$^2$ de máxima carga; sin embargo al templar este acero mejoraremos sus propiedades para darle un mejor factor de seguridad. Las condiciones de templado de AISI 1075 se muestran a continuación.

**Material seleccionado para la planchas de metal**

Se llegó a la conclusión que el material más adecuado para el diseño de las placas de metal del sensor de presión es ACERO AISI 1075 TEMPLADO cuyas características serán:

Acero AISI 7075

- Tipo de aleación: C 0.77 Si 0.25 Mn 0.7%
- Estado de suministro: Templado y revenido 47-50HCR aprox.
- Acero fino al carbono de alta calidad

Instrucciones del tratamiento térmico:

- Forjado: 1050-850°C
- Normalizado: 820 – 850°C
- Recocido: 650 – 700°C
- Temple: al aceite 810 – 840°C
- Dureza obtenible: 60 – 65HCR
- Revenido: 100 – 500°C
- Resistencia a la tracción máxima= 1600-1650N/mm2=163.155-168.253kgf/mm2

El cual es mucho mayor que cualquiera de los antes mencionados en tabla. Después de hacer todas las pruebas, y de acuerdo a los espesores comerciales en los que hay de estas planchas de acero templado, se seleccionó las de espesor de 0.5mm.

A continuación se procederá a determinar y presentar el análisis, para determinar cuál debe ser el ancho y separación adecuado para las planchas, con el material seleccionado.

**CASO2**

Trabajaremos en este caso con planchas de 0.5mm de espesor y el neopreno que se usa de separación de las placas será también de 0.5mm, y el ancho de las placas será de 31mm. (Ver figura 3.18M)

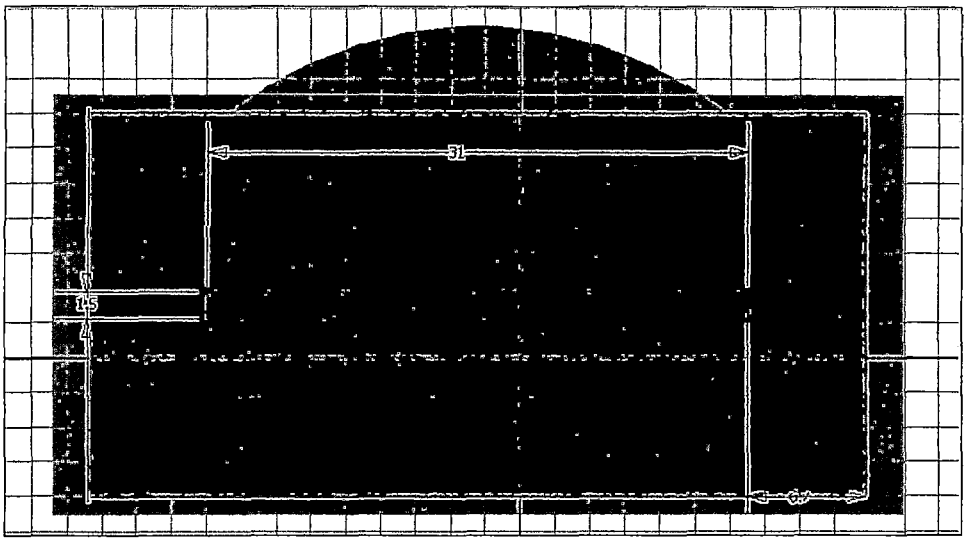



***Figura 3.18M.-*** *Perfil del caso 2*

Veamos los resultados de esfuerzo y desplazamiento que se obtuvieron, para la fuerza máxima, en las figuras 3.19M y 3.20M respectivamente.

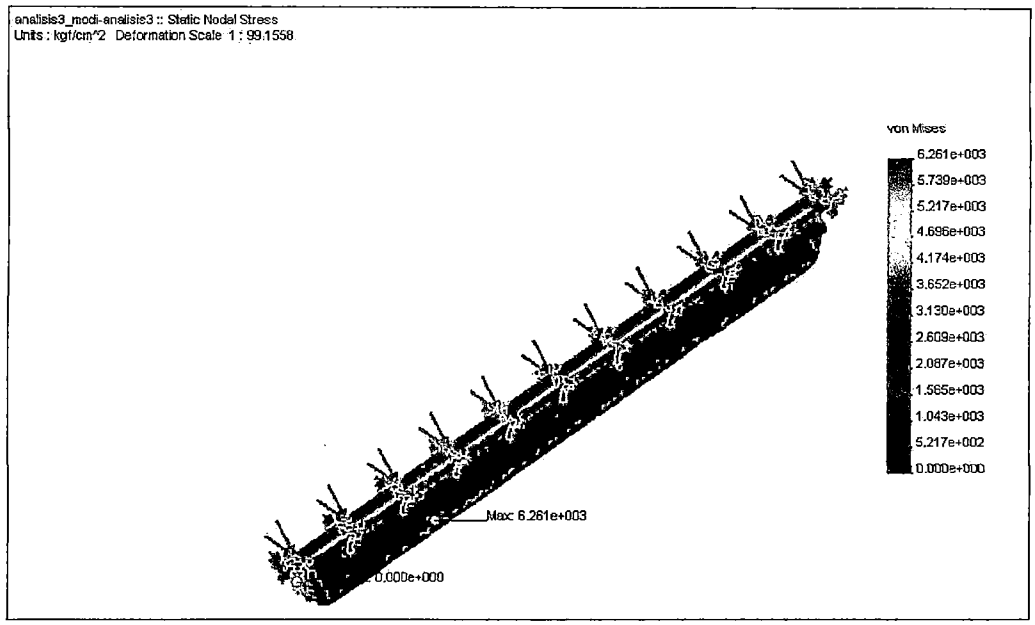



***Figura 3.19M.-*** *Esfuerzos en las planchas metálicas para el caso 2, debido a las fuerzas máximas*

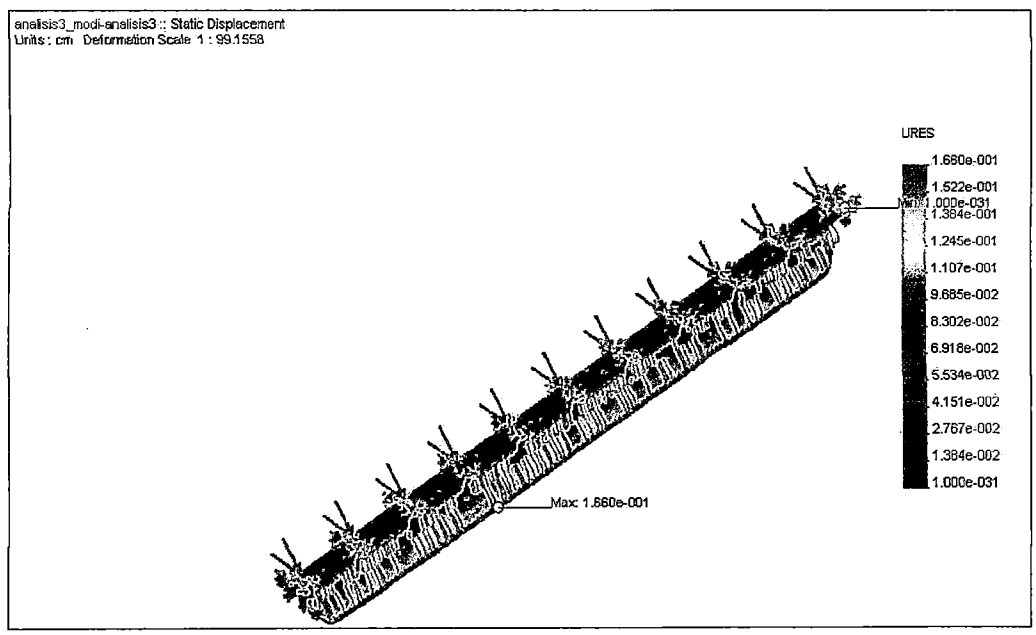



***Figura 3.20M.-*** *Desplazamientos en las planchas metálicas para el caso 2, debido a las fuerzas máximas*

Como es de suponer no se encuentra ningún problema con respecto al esfuerzo máximo. Observemos además, en la figura 3.20M, que el desplazamiento es mayor que 0.5mm, lo que hace al sensor adecuadamente sensible.

Ahora veamos los resultados para la mínima fuerza en las figuras 3.21M y 3.22M de esfuerzo y desplazamiento.

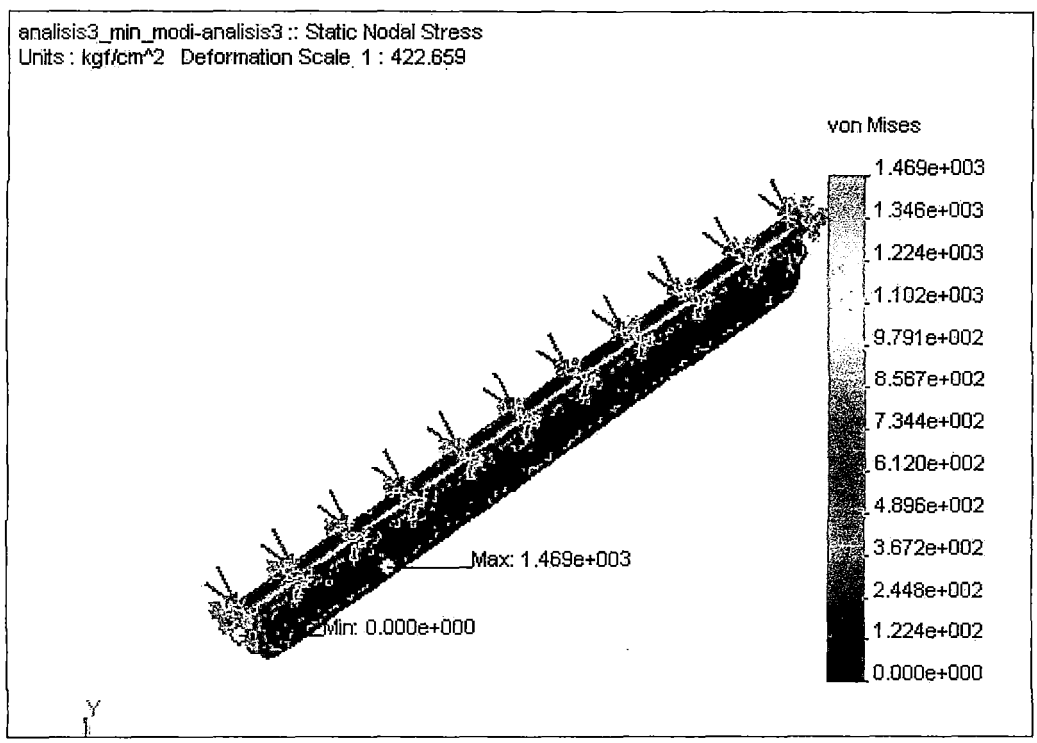



***Figura 3.21M.-*** *Esfuerzos en las planchas metálicas para el caso 2, debido a las fuerzas mínimas*

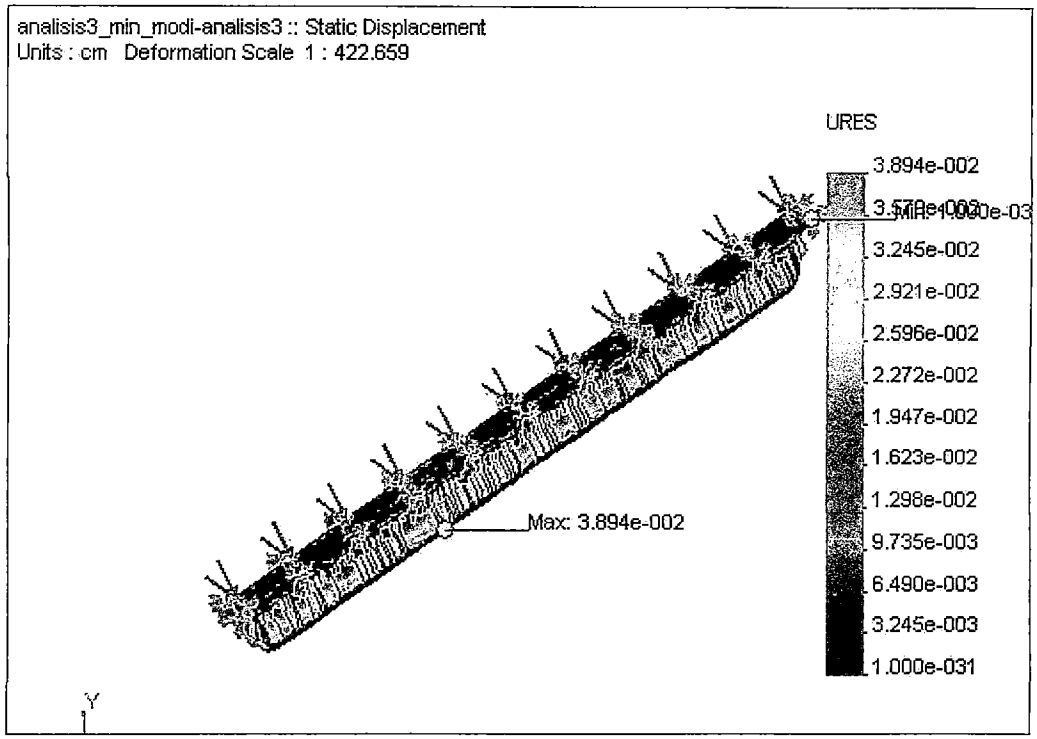



***Figura 3.22M.-*** *Desplazamientos en las planchas metálicas para el caso 2, debido a las fuerzas mínimas*

En este caso el desplazamiento todavía es menor que 0.5mm, esto es 0.389mm, lo que significaría que el sensor no detectaría un vehículo; por lo tanto no satisface las condiciones.

**CASO3**

Con la intensión de incrementar la sensibilidad del sensor; es decir, que con la misma carga el desplazamiento pueda superar los 0.5mm, el ancho de las placas serán de 35mm y los espesores y separación seguirán siendo de 0.5mm como se muestra en la figura 3.23M.

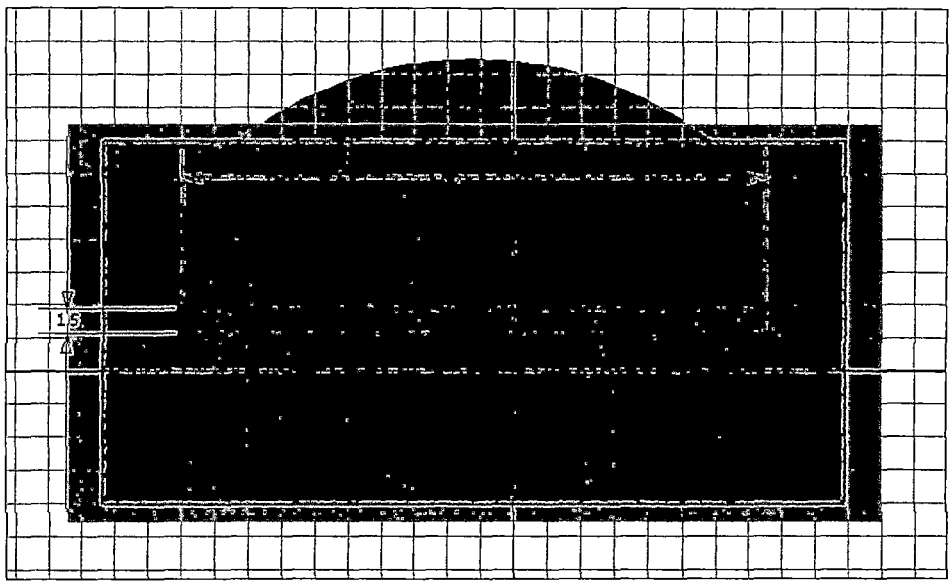


***Figura 3.23M.-*** *Perfil caso 3*

Los resultados para la fuerza máxima se muestran en las figuras 3.24M y 3.25M tanto para el esfuerzo como para el desplazamiento. Se aprecia, en la figura 3.25M, que no existe problema con el desplazamiento, sin embargo es necesario analizar para la fuerza mínima.

Los resultados obtenidos para la aplicación de la mínima fuerza pueden ser mostrados en las figuras 3.26M y 3.27M.

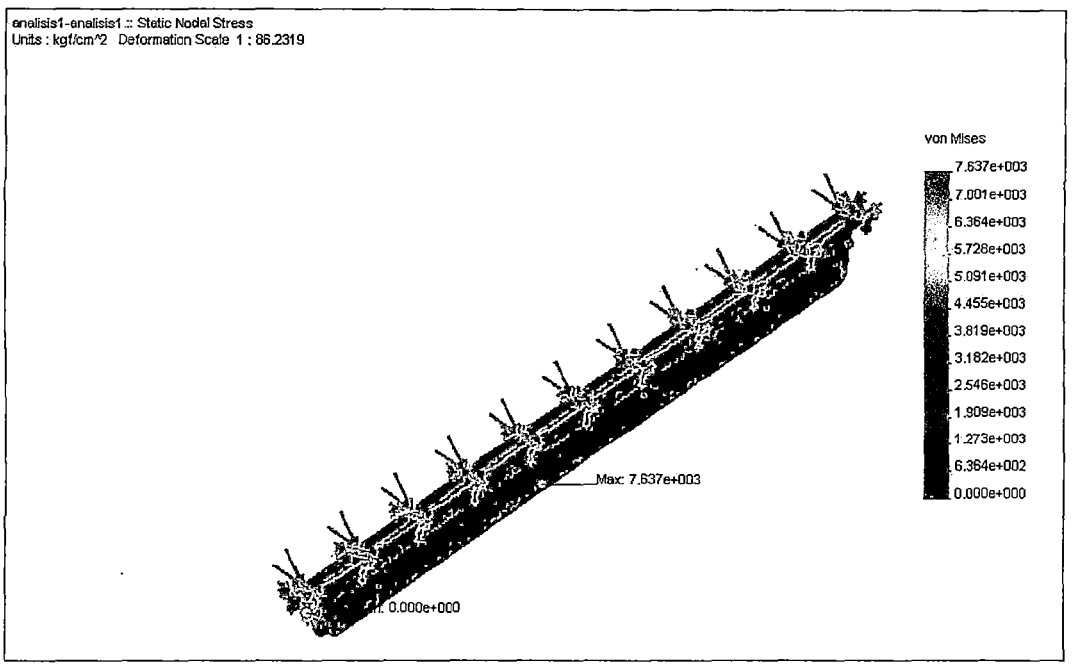



***Figura 3.24M.-*** *Esfuerzos en las planchas metálicas para el caso 3, debido a las fuerzas máximas*

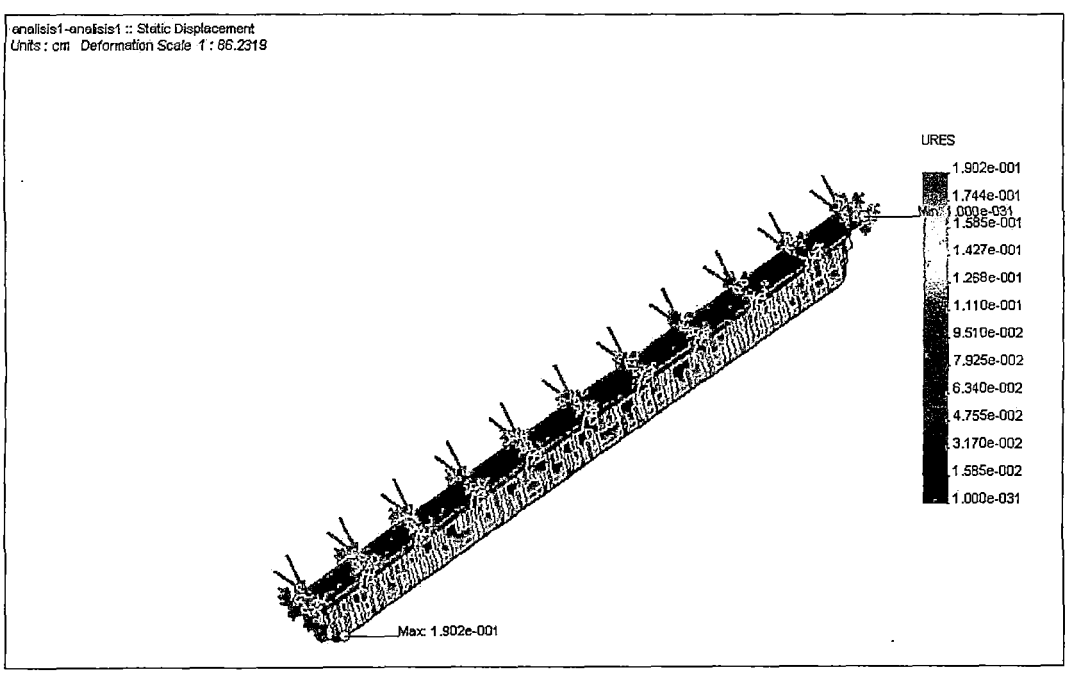



***Figura 3.25M.-*** *Desplazamientos en las planchas metálicas para el caso 3, debido a las fuerzas máximas*

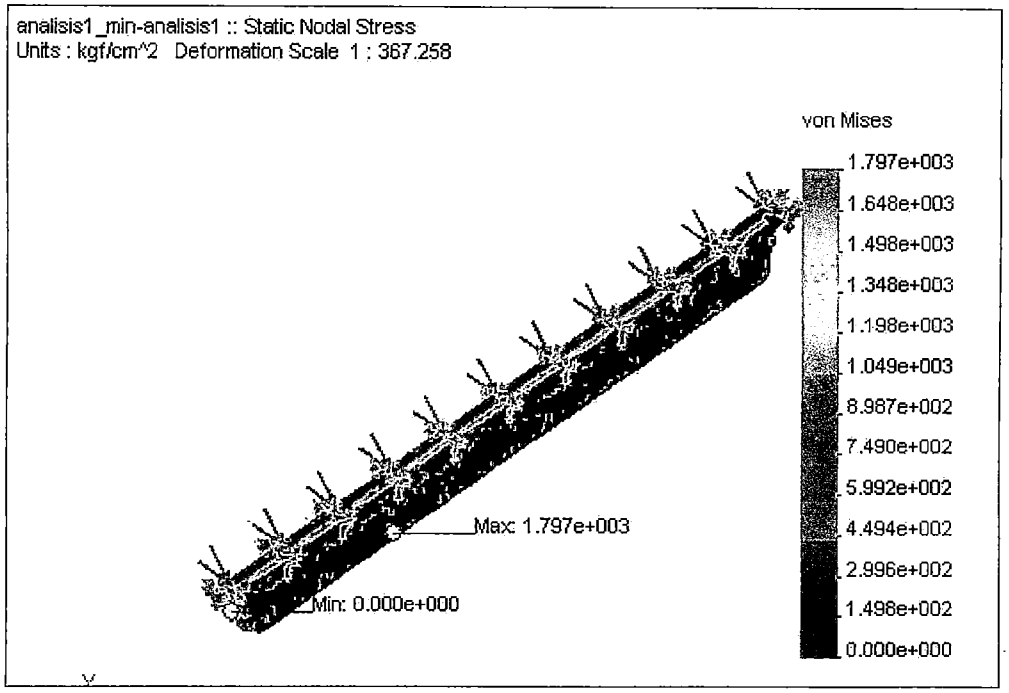



***Figura 3.26M.-*** *Esfuerzos en las planchas metálicas para el caso 3, debido a las fuerzas mínimas*

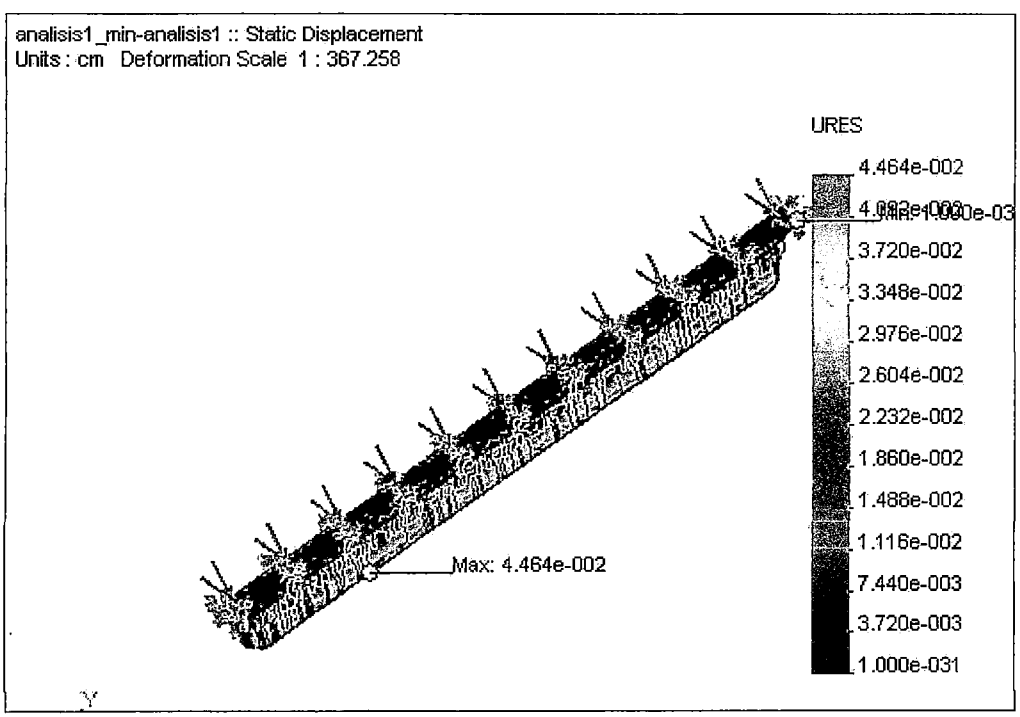



***Figura 3.27M.-*** *Desplazamientos en las planchas metálicas para el caso 3, debido a las fuerzas mínimas*

Aún podemos apreciar, para la fuerza mínima, que las planchas no consiguen desplazarse 0.5mm, sino sólo 0.45mm. Quizás por la aproximación finita de elementos de este tipo de cálculo es que no se consigue llegar al objetivo, sin embargo estas medidas podrían ser las adecuadas realmente.

**CASO4**

Finalmente, para conseguir mayor sensibilidad (esto significa que el sensor detecte cargas más bajas, como el peso de autos ligeros), es necesario aumentar el ancho de las placas y con eso conseguiremos mayor flecha. Por lo tanto, trabajaremos con placas de 37mm de ancho con el mismo espesor de 0.5mm y una separación de 0.5mm (Ver figura 3.28M). Debemos recordar que, en lo que respecta a la resistencia a la compresión, se seleccionó un material adecuado, así que ha de esperarse que en este caso también el material resista los esfuerzos como en los casos anteriores. Pese a todo esto, de todas maneras, se realizaran las pruebas de simulación para el esfuerzo.

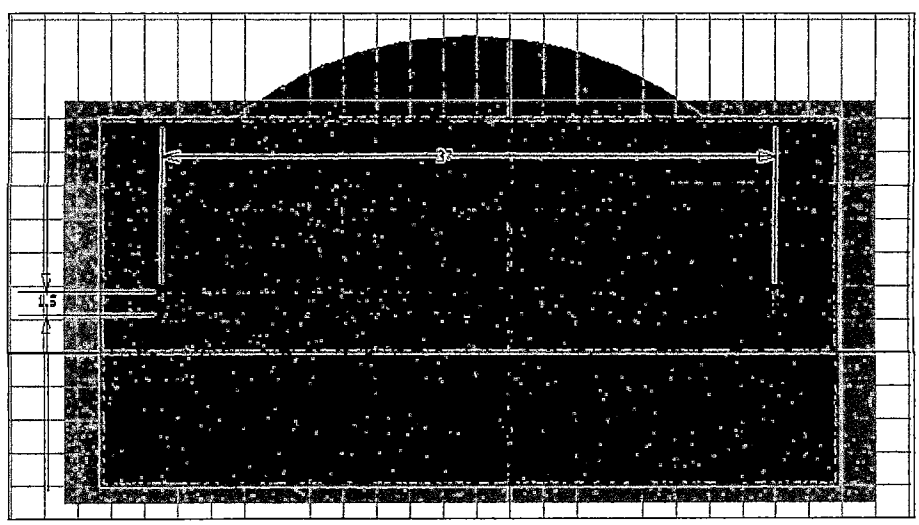



***Figura 3.28M.-*** *Perfil caso 4*

Así como lo hicimos en los casos anteriores, mostramos los resultados obtenidos para la máxima carga (camiones de varios ejes) en las figuras 3.29M y 3.30M y para la carga mínima (autos ligeros) en las figuras 3.32M y 3.33M para los esfuerzos y desplazamientos respectivamente.

Se muestra en la figura 3.31M el gráfico de esfuerzos en donde sólo se muestran las placas de metal, en donde se puede observar de que la zona crítica esta justo en la línea transversal que pasa por el centro de las placas.

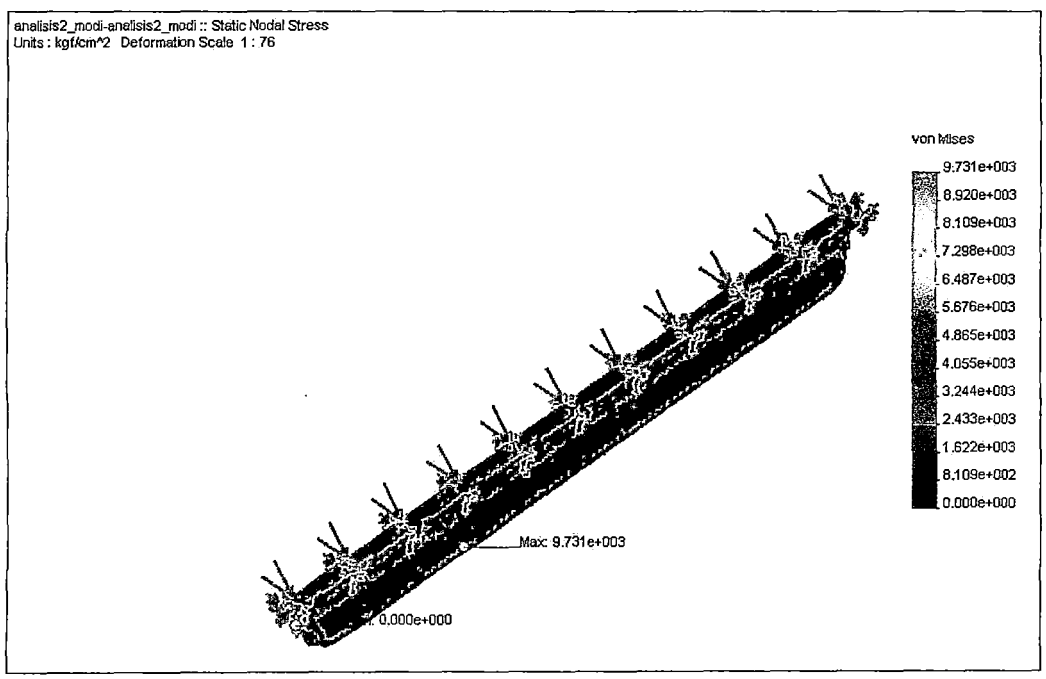



***Figura 3.29M.-*** *Esfuerzos en las planchas metálicas para el caso 4, debido a las fuerzas máximas*

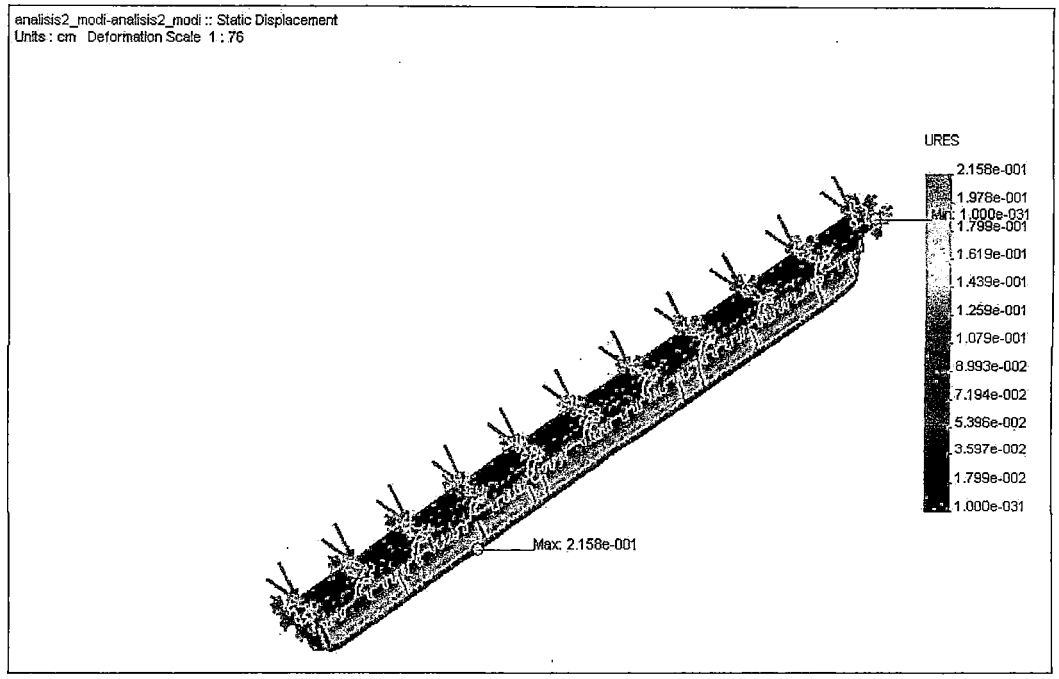



***Figura 3.30M.-*** *Desplazamientos en las planchas metálicas para el caso 4, debido a las fuerzas máximas*

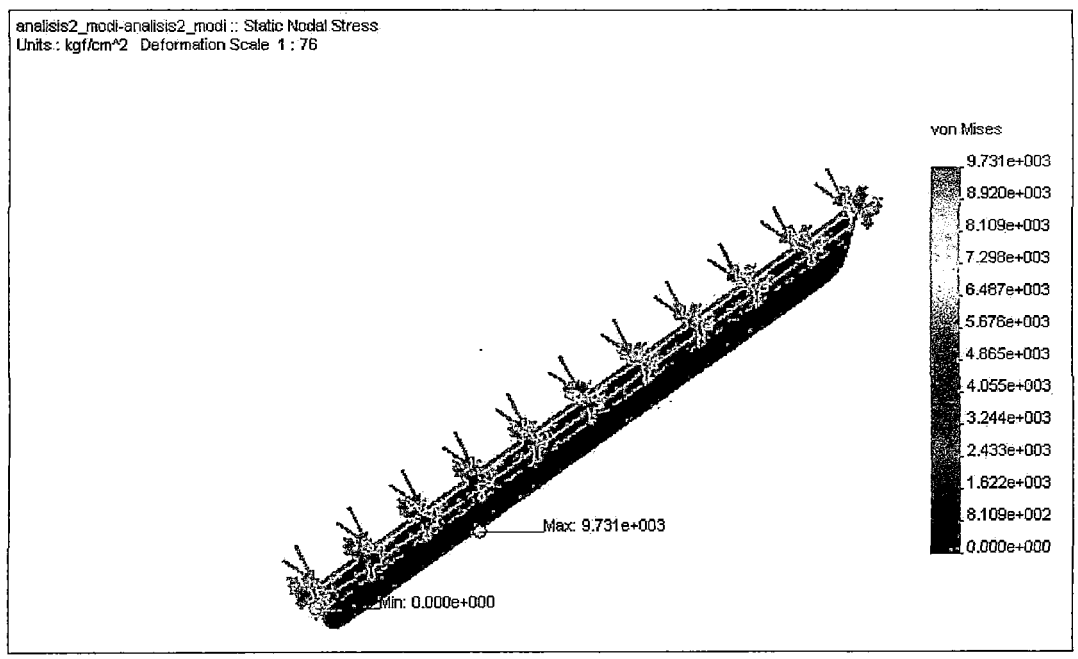



***Figura 3.31M.-*** *Detalle de placas de metal del caso 4 para fuerzas máximas*

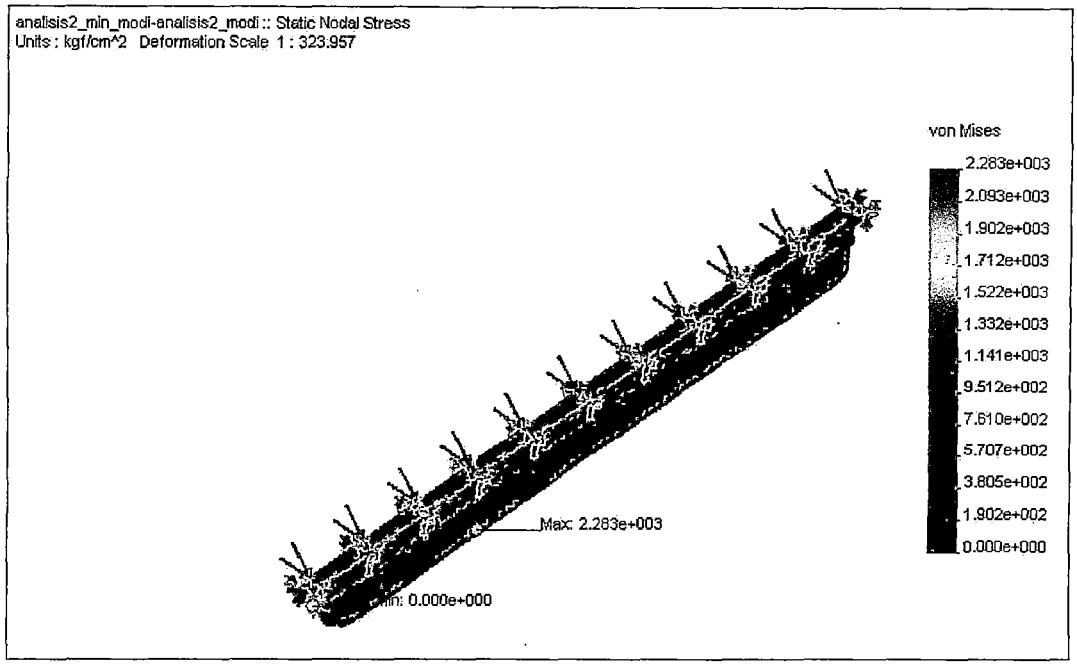



***Figura 3.32M.-*** *Esfuerzos en las planchas metálicas para el caso 4, debido a las fuerzas mínimas*

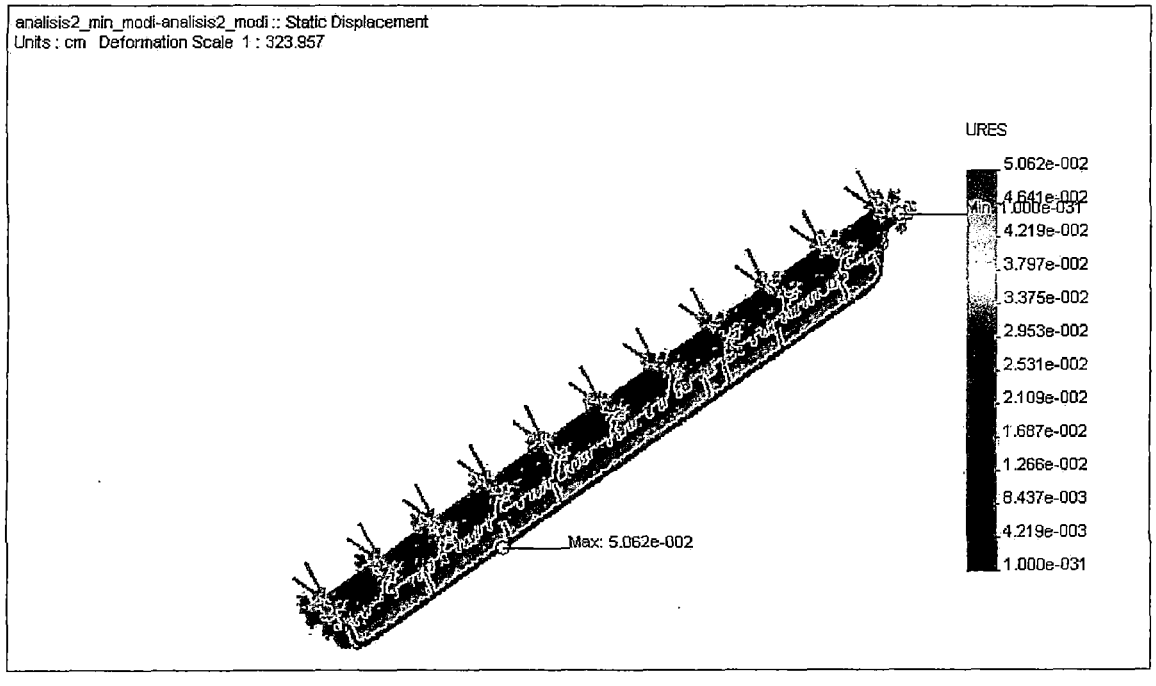



***Figura 3.33M.-*** *Desplazamientos en las planchas metálicas para el caso 4, debido a las fuerzas mínimas*

En este caso podemos apreciar que el máximo desplazamiento de las placas cuando se aplica la mínima carga se da en el medio y además supera los 0.5mm, es decir llega a 0.5062mm. Esto significa que el sensor es sensible a los vehículos ligeros como los autos.

### 3.5.7. Análisis y decisiones finales

Finalmente se verificarán los factores de seguridad. Podemos apreciar, para cada uno de los casos estudiados, los factores de seguridad mínimo (ver figuras del 3.34M al 3.38M) para la fuerzas máximas. Esto fue calculado según el criterio de Von Mises. De acuerdo a este criterio el factor de seguridad nos dice en cuanto la resistencia al máximo esfuerzo del material supera el esfuerzo calculado debido a las cargas aplicadas, es decir el factor de seguridad mínimo debe ser mayor que la unidad para garantizar que el material no supere la zona elástica y así no quede con una deformación permanente. Sólo mostraremos los casos de fuerza máxima.

Es importante resaltar que anteriormente ya se hizo este cálculo para otros tipos de materiales y para diversos casos. Esos resultados no se presentan aquí; pero en la mayoría de los casos el factor de seguridad con dificultad llegaba a la unidad, por esta razón se optó por el acero templado de las características mencionadas anteriormente. En la figura 3.38M se muestra en detalle el caso 4 para las placas metálicas.

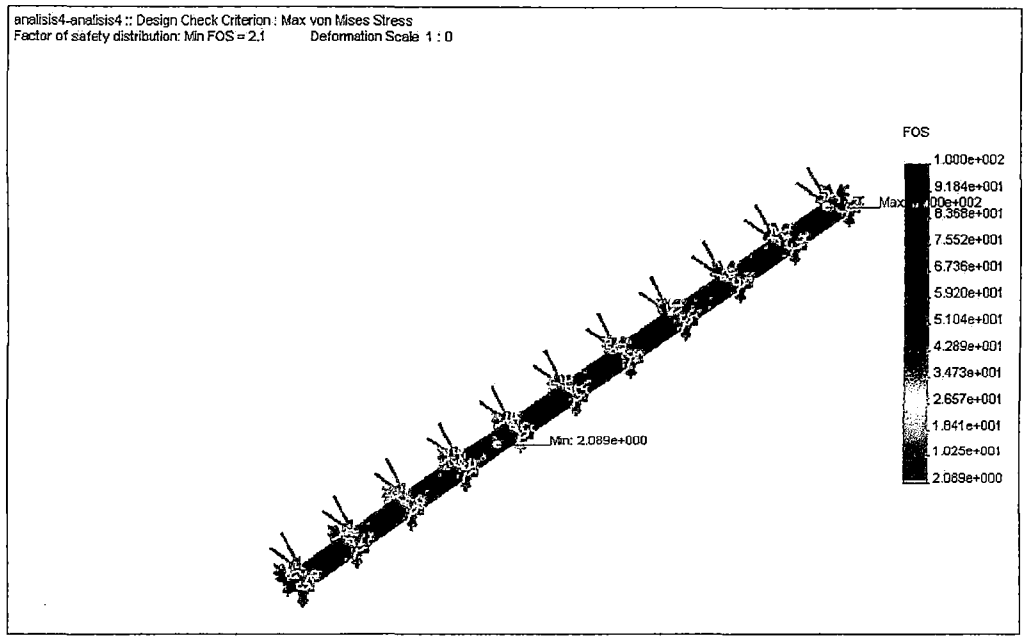



*Figura 3.34M.- Caso 1*

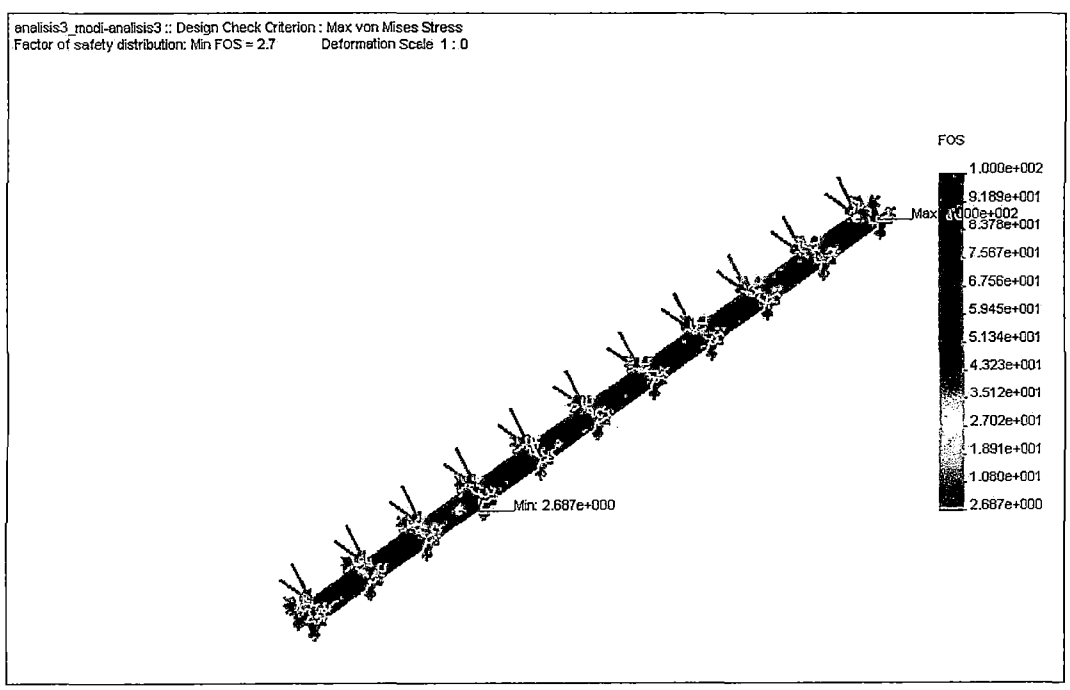



*Figura 3.35M.- Caso 2*

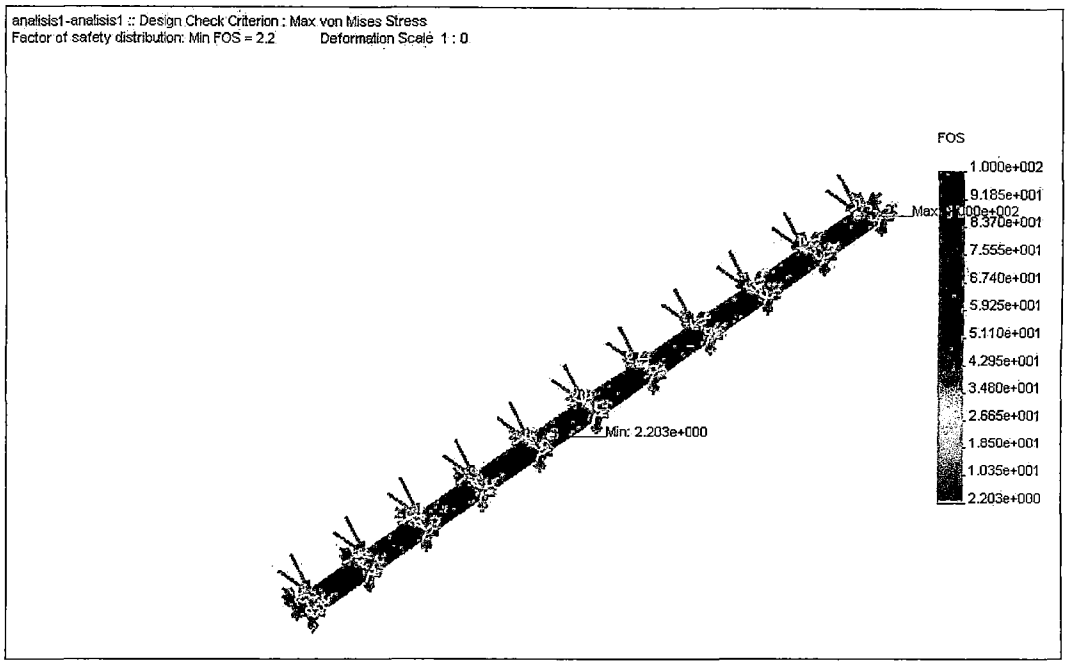



*Figura 3.36M.- Caso 3*

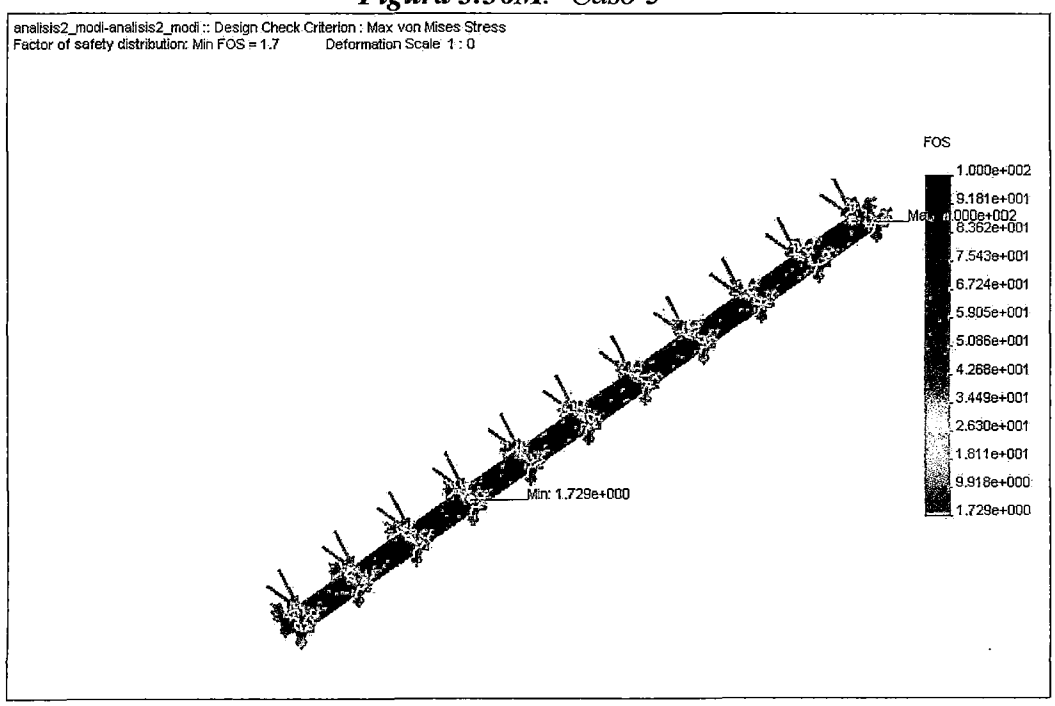



*Figura 3.37M.- Caso 4*

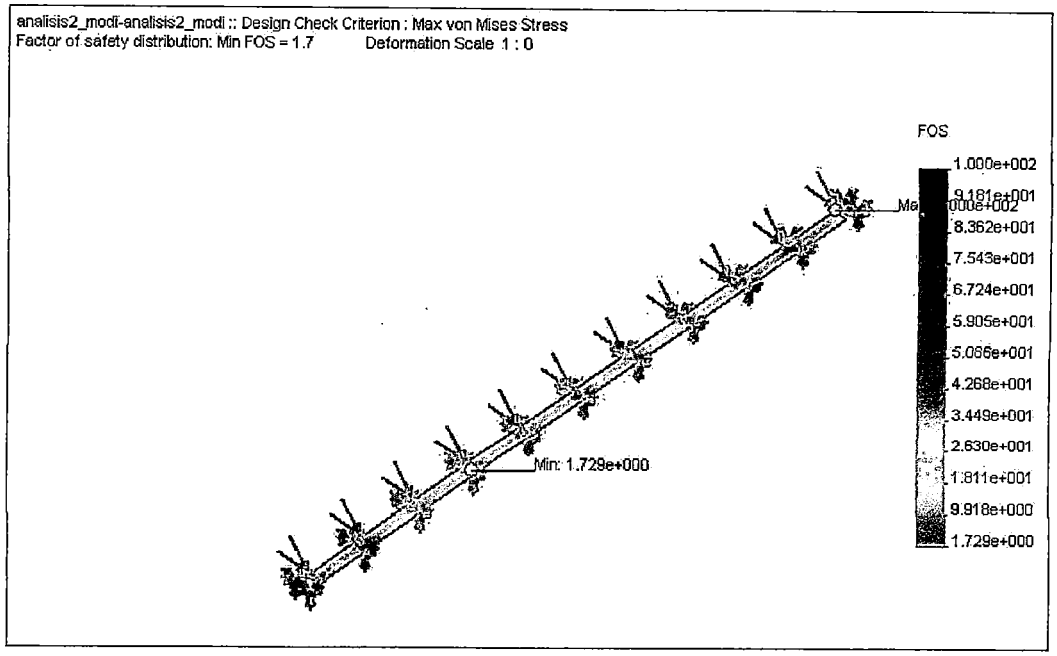



*Figura 3.38M.- Caso 4*

*Tabla 3.2M*
*(Resumen de los análisis para cada uno de los casos)*

| | descripción | | desplazamiento | Factor de seguridad mínimo | resultado | conclusión |
|---|---|---|---|---|---|---|
| *caso 1* | espesor: 1mm<br>ancho: 31mm<br>separación: 1mm | Fuerza máxima<br>Fuerza mínima | 1,354mm<br>0,317mm | 2,1<br>8,9 | √<br>X | falla |
| *caso 2* | espesor: 0,5mm<br>ancho: 31mm<br>separación: 0,5mm | Fuerza máxima<br>Fuerza mínima | 1,66mm<br>0,3894mm | 2,7<br>11 | √<br>X | falla |
| *caso 3* | espesor: 0,5mm<br>ancho: 35mm<br>separación: 0,5mm | Fuerza máxima<br>Fuerza mínima | 1,902mm<br>0,464mm | 2,2<br>9,4 | √<br>X | falla |
| *caso 4* | espesor: 0,5mm<br>ancho: 37mm<br>separación: 0,5mm | Fuerza máxima<br>Fuerza mínima | 2,158mm<br>0,5062mm | 1,7<br>7,4 | √<br>√ | óptimo |

### 3.4.8. Conclusión

De acuerdo a todo lo analizado, así como lo muestra la tabla 3.2M se usará las dimensiones, para el sensor de presión, del **caso** número **4**.

A continuación un resumen que genera el software "Cosmos DesignStar":

## A. Materiales

*Tabla 3.3M*

| No. | Part Name | Material | Mass | Volume |
|---|---|---|---|---|
| 1 | Part1 | neopreno | 2.40308 kg | 1657.3 cm^3 |
| 2 | Part2 | neopreno | 0.0022765 kg | 1.57 cm^3 |
| 3 | Part3 | neopreno | 0.0022765 kg | 1.57 cm^3 |
| 4 | Part4 | fleje templado | 0.227422 kg | 29.045 cm^3 |
| 5 | Part5 | fleje templado | 0.227422 kg | 29.045 cm^3 |

## B. Información de carga máxima

*Tabla 3.4M*

| **Restricciones** | |
|---|---|
| **Restraint:1 <Part1>** | on **5 Face(s)** Fixed. |
| **Description:** | |
| **Cargas** | |
| **Load:1 <Part1>** | on **1 Face(s)** apply normal force **15260 kgf** using uniform distribution |
| **Description:** | |
| **Load:2 <Part1>** | on **1 Face(s)** apply force **2290 kgf** along plane Dir 1 using uniform distribution |
| **Description:** | |

## C. Propiedades de estudio

*Tabla 3.5M*

| Información del enmallado | |
|---|---|
| Mesh Type: | Solid Mesh |
| Mesher Used: | Standard |
| Automatic Transition: | Off |
| Include Mesh Controls: | On |
| Smooth Surface: | On |
| Jacobian Check: | 4 Points |
| Element Size: | 0.5 cm |
| Tolerance: | 0.025 cm |
| Quality: | High |
| Number of elements: | 127771 |
| Number of nodes: | 198020 |

## D. Resultados de esfuerzos

*Tabla 3.6M*

| Name | Type | Min | Location | Max | Location |
|---|---|---|---|---|---|
| Plot-1 | VON: von Mises stress | 0 kgf/cm^2 Node: 3602 | (2.46326 cm, -0.801034 cm, 163 cm) | 9730.87 kgf/cm^2 Node: 191860 | (0.243256 cm, 0.323966 cm, 112.25 cm) |

## E. Resultados de deformación

*Tabla 3.7M*

| Name | Type | Min | Location | Max | Location |
|---|---|---|---|---|---|
| Plot-3 | ESTRN : Equivalent strain | 0 Element: 92770 | (2.33992 cm, -0.691034 cm, 162.875 cm) | 0.500418 Element: 96657 | (1.9041 cm, 0.304933 cm, 141.857 cm) |

## F. Desplazamiento

*Tabla 3.8M*

| Name | Type | Min | Location | Max | Location |
|---|---|---|---|---|---|
| Plot-2 | URES: Resultant displacement | 0 cm<br>Node: 2218 | (2.46326 cm, 1.39897 cm, 0 cm) | 0.215827 cm<br>Node: 965 | (0.243256 cm, 1.89897 cm, 112.5 cm) |

## G. Información de carga mínima

*Tabla 3.9M*

| Restricciones | |
|---|---|
| **Restraint:1 <Part1>** | on **5 Face(s)** Fixed. |
| **Description:** | |
| **Load** | |
| **Load:1 <Part1>** | on **1 Face(s)** apply normal force **3580 kgf** using uniform distribution |
| **Description:** | |
| **Load:2 <Part1>** | on **1 Face(s)** apply force **530 kgf** along plane Dir 1 using uniform distribution |
| **Description:** | |

## H. Resultados de esfuerzo (Para carga mínima)

*Tabla 3.10M*

| Name | Type | Min | Location | Max | Location |
|---|---|---|---|---|---|
| Plot-1 | VON: von Mises stress | 0 kgf/cm^2<br>Node: 3602 | (2.46326 cm, -0.801034 cm, 163 cm) | 2282.91 kgf/cm^2<br>Node: 191860 | (0.243256 cm, 0.323966 cm, 112.25 cm) |

## I. Resultados de deformación (Para carga mínima)

*Tabla 3.11M*

| Name | Type | Min | Location | Max | Location |
|---|---|---|---|---|---|
| Plot-3 | ESTRN : Equivalent strain | 0<br>Element: 92770 | (2.33992 cm,<br>-0.691034 cm,<br>162.875 cm) | 0.117331<br>Element: 96657 | (1.9041 cm,<br>0.304933 cm,<br>141.857 cm) |

## J. Desplazamiento (para carga mínima)

*Tabla 3.12M*

| Name | Type | Min | Location | Max | Location |
|---|---|---|---|---|---|
| Plot-2 | URES: Resultant displacement | 0 cm<br>Node: 2218 | (2.46326 cm,<br>1.39897 cm,<br>0 cm) | 0.0506241 cm<br>Node: 965 | (0.243256 cm,<br>1.89897 cm,<br>112.5 cm) |

## K. Materiales

*Tabla 3.13M*

**Nombre del material:** Neopreno

**Tipo de modelo:** Linear Elastic Isotropic

L.

| Property Name | Value |
|---|---|
| Elastic modulus | 700 kgf/cm^2 |
| Poisson's ratio | 0.34 |
| Shear modulus | 9.1775 kgf/cm^2 |
| Mass density | 0.00145 kg/cm^3 |
| Tensile strength | 3600 kgf/cm^2 |
| Yield strength | 2200 kgf/cm^2 |
| Thermal expansion coefficient | 0 /Centigrade |
| Thermal conductivity | 0 Cal/(cm.s.C) |
| Specific heat | 0 Cal/(kg.C) |

*Tabla 3.14M*

**Nombre del material:** fleje templado

**Tipo de modelo:** Linear Elastic Isotropic

| Property Name | Value |
|---|---|
| Elastic modulus | 2.1414e+006 kgf/cm^2 |
| Poisson's ratio | 0.3 |
| Shear modulus | 8.0864e+005 kgf/cm^2 |
| Mass density | 0.00783 kg/cm^3 |
| Tensile strength | 16825 kgf/cm^2 |
| Yield strength | 9700.8 kgf/cm^2 |
| Thermal expansion coefficient | 1.5e-005 /Centigrade |
| Thermal conductivity | 0.1195 Cal/(cm.s.C) |
| Specific heat | 100.38 Cal/(kg.C) |

# CAPÍTULO IV

# DISEÑO DEL SISTEMA DE SUPERVISIÓN Y GENERADOR DE REPORTES

## 4.1. LENGUAJE VISUAL LABVIEW COMO HERRAMIENTA PARA EL DISEÑO DEL SISTEMA DE CONTROL

Antes de explicar cómo se desarrolló el sistema de supervisión, se revisarán algunos aspectos teóricos del software usado.

### 4.1.1. Instrumento Virtual

- Los instrumentos virtuales pueden ser definidos dinámicamente por el usuario, mientras que instrumentos tradicionales tienen funcionalidad fija, no definida por el usuario.
- Cada instrumento virtual tiene dos partes – software y hardware.
- Al no utilizar software y hardware preestablecido, ingenieros y científicos obtienen máxima flexibilidad definida por el usuario. Un instrumento tradicional proporciona tanto software como circuitos de medición empacados en un producto con lista finita o funcionalidad fija utilizando el instrumento del panel frontal

- Una tendencia básica en la industria de pruebas automatizadas es un cambio muy marcado hacia sistemas de pruebas basados en software. Por ejemplo, el Departamento de la Defensa de los Estados Unidos es uno de los mayores clientes de equipo para pruebas automatizadas (ATE).

### 4.1.2. Usando LABVIEW

Con más de 6 nuevos millones de canales de medición vendidos en el último año, National Instruments es líder mundial en la instrumentación virtual. Ingenieros han usado la instrumentación virtual por más de 25 años para traer la potencia de software flexible y tecnología PC para probar, controlar y diseñar aplicaciones haciendo mediciones análogas y digitales exactas de corriente directa hasta 2.7 GHz.

### 4.1.3. ¿Qué es LABVIEW?

Podemos definirlo de la siguiente forma:

a) Es un lenguaje de programación gráfico, orientado al control, monitoreo, procesamiento de señales, aplicaciones en tiempo real, etc.

b) Utiliza dos ventanas: la de programación llamada " Panel de Diagramas de Bloques" y la de HMI llamada " Panel Frontal"

c) Tiene una mejora continua en la potencia del Software como:

- La mayoría de sus libreras están hechas en C, mejorando la velocidad de procesamiento.

- Tiene integrado la facilidad de programar en ANCI C, MATLAB, sin la necesidad de disponer esos programas.
- Tiene la posibilidad de importar diseños en CAD y CAE (análisis mecánicos, análisis térmicos, etc.)
- La tecnología Express, es un conjunto de herramientas diseñadas para que los ingenieros sean más exitosos, sin importar la experiencia en programación o la plataforma de hardware.

d) LABVIEW introduce nuevos Express VIs. Este lanzamiento también marca la innovación más significativa del Módulo LABVIEW Real Time desde su introducción en 1999, con soporte de NI-DAQmx, herramientas avanzadas de disparo y tiempo, y la expansión de LABVIEW Real Time para ejecutarse en PC's de escritorio.

e) A la fecha tiene una amplia variedad de librerías, como:

- PID
- FUZZY (ver figura 4.2)
- ANÁLISIS DE SONIDO Y VIBRACIÓN
- ANÁLISIS DE ORDEN
- VISIÓN Y PROCESAMIENTO DE IMÁGENES
- MATRIX –X
- SYSTEM IDENTIFICATION
- ETC.

LABVIEW es un lenguaje de programación gráfico para ingeniería, el cual tiene innumerables herramientas para matemáticas, control, simulación, comunicación, etc. En la figura 4.1 podemos ver como se trabajan con las herramientas de control PID.

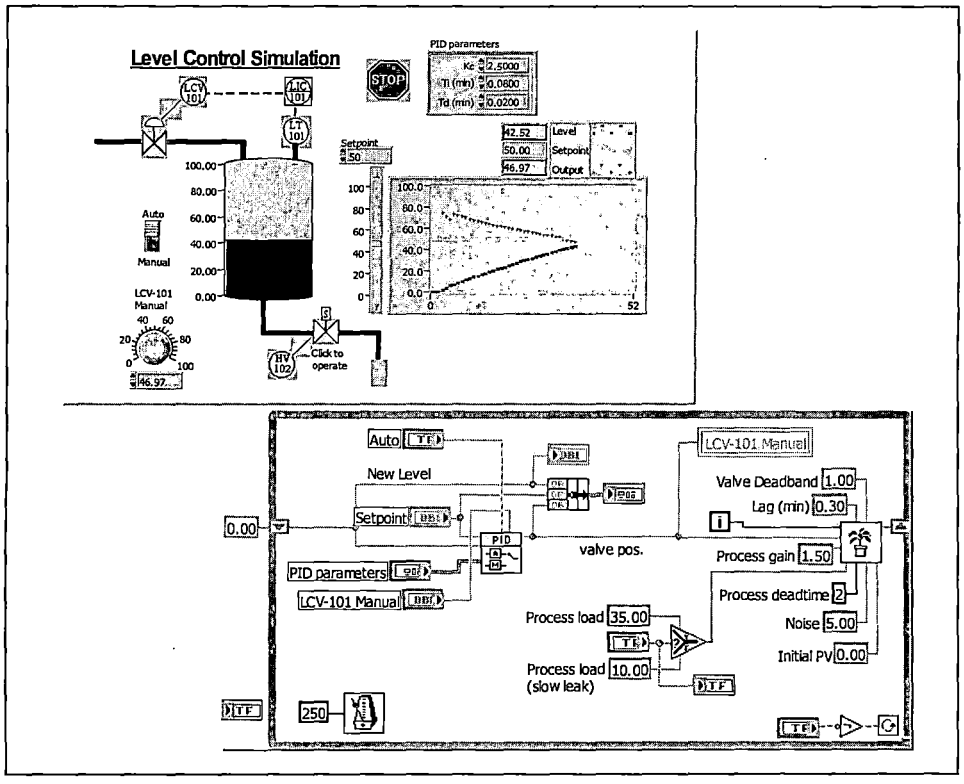



***Figura 4.1.-*** *Ejemplo de control PID*

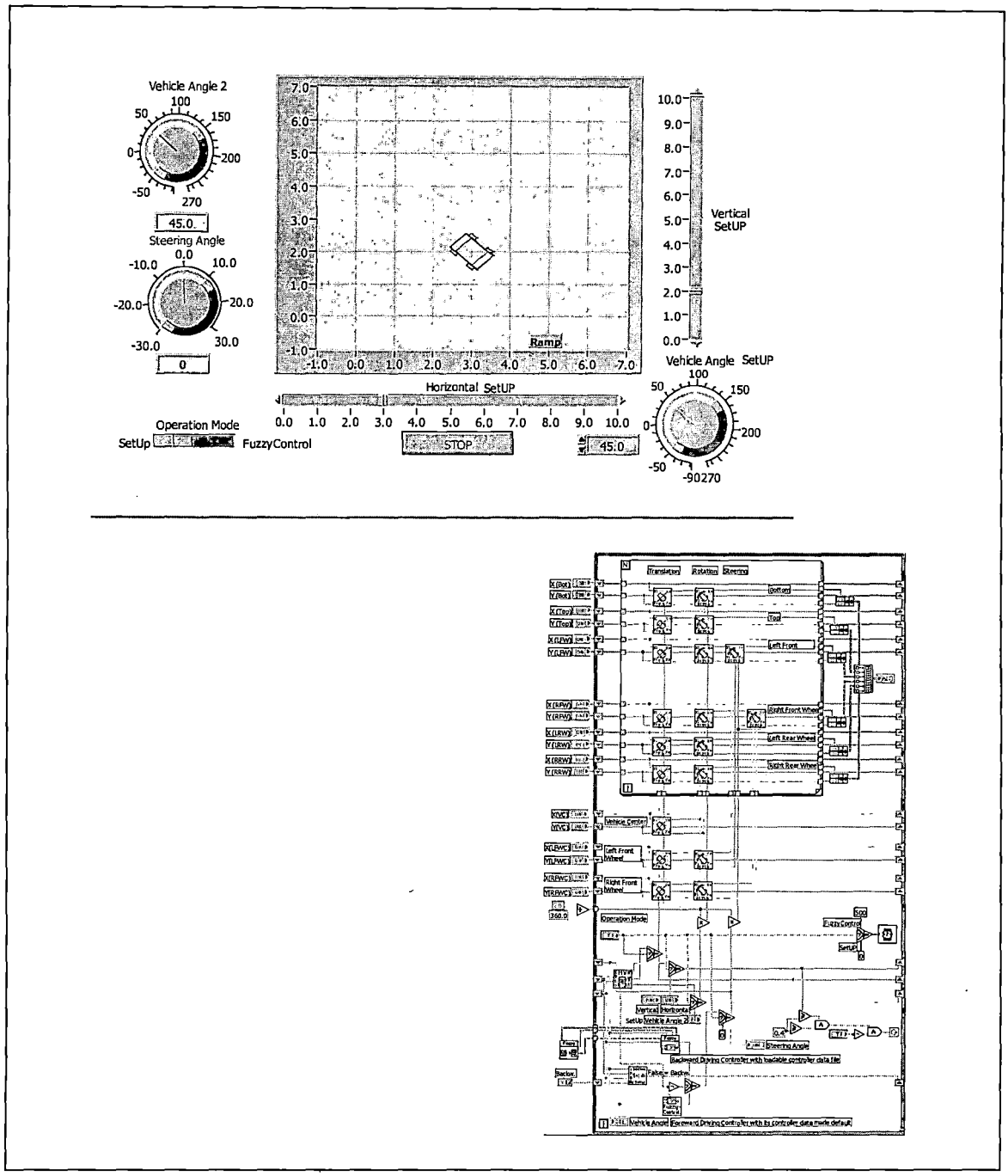



***Figura 4.2.-*** *Ejemplo de Lógica Fuzzy*

### 4.1.4. Manejo de puertos con LABVIEW

En este lenguaje de programación se pueden usar diversos puertos para la comunicación con diversos hardwares de National Instruments y otros. Estos pueden ser:

- El puerto serial, para soportar protocolos de comunicación como el RS232, RS485, MODBUS, PROFIBUS, etc.
- Puerto PCI. Este puerto se usa para la comunicación también de algunas tarjetas DAQ como la PCI 1200 de National Instruments, tarjetas que se insertan internamente a una PC.
- Puerto RJ45. Que la mayoría de hardware que soportan el protocolo TCP/IP, UDP, DataSocket, Logos (que son usados para sistemas de supervisión, etc.
- El puerto USB, que soportan protocolos para algunas tarjetas de adquisición de datos, instrumentos de medición y ahora también los computadores industriales PXI como los PAC (Procesadores Automáticos Programables) ya vienen con estos puertos para la extracción de datos de una forma práctica.
- Puerto paralelo. Que se usan preferentemente para el protocolo GPIB para instrumentos de medición, también cuando se tienen procesadores digitales (no de puntos flotantes), para FPGA como el Compact Rio, etc.

De todos estos puertos usaremos el puerto serial COM y trabajaremos con el protocolo RS232, ya que el microcontrolador que usamos soporta este protocolo y además es fácil de implementar. Debido a esto detallaremos como se trabaja con este puerto en LABVIEW.

### 4.1.5. Puerto serial

Primero veremos cómo configurar este puerto para el protocolo RS232. Para esto usaremos el subVI VISA como se aprecia en la figura 4.3, con el cual se puede configurar:

- El COM o puerto a usar como el COM1, COM2, etc.
- La velocidad en baudios, que por defecto se encuentra a 9600baudios pero puede llegar hasta 115200 baudios.
- Número de bits de datos. Por defecto está a 8 bits pero puede trabajar a 9 bits esto es cuando se trabajan a multipunto.
- Bit de paridad. Por defecto no está habilitado pero puede trabajar con bit de paridad.
- Bit de stop. Pudiendo ser 1, 1.5 y 2.
- Control de flujo. Trabajo con varios tipos, de lo cual si se desea transmitir lo más rápido posible se elegirá ninguno.
- Habilitación de carácter final. Si se desea enviar una cadena de caracteres se puede habilitar para que automáticamente envíar un caracter en especial al final, y cuando recibe esperar también este carácter.

- Tipo de carácter de final de cadena. Aquí se especifica cuál es el caracter que se desea enviar al final de una cadena.
- Tiempo de espera. Se puede especificar el tiempo que esperará la PC para recibir el dato que se pidió.

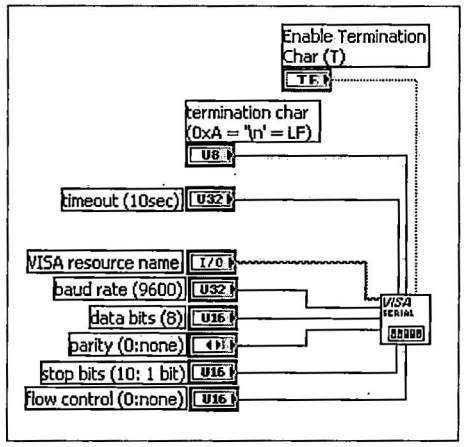



***Figura 4.3.-*** *Configuración del puerto serial para el protocolo RS232*

## A. Lectura del puerto serial

Para recibir datos por este puerto se usará el VI VISA READ, en donde se especificará la cantidad de datos que se espera recibir y en un indicador, en formato cadena, se mostrará el dato recibido. Esto se hace a partir de la configuración que se hizo anteriormente. En la figura 4.4 se aprecia la paleta para la comunicación serial. El nodo de "byte count" recibe el tamaño de buffer que se escribió en el puerto. (Ver figura 4.4).

Por último se debe cerrar la sesión VISA para liberar el puerto y poderle dar otra función. Esto se logra con "VISA close" en Functions >>

Instrument I/O >> Serial >> Visa Close, (ver figura 4.5) y, como buena costumbre de programación se coloca un controlador de errores.

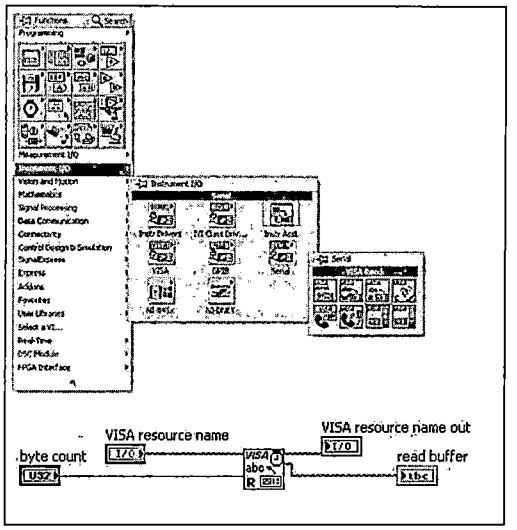



***Figura 4.4.-*** *Herramientas VISA Read*

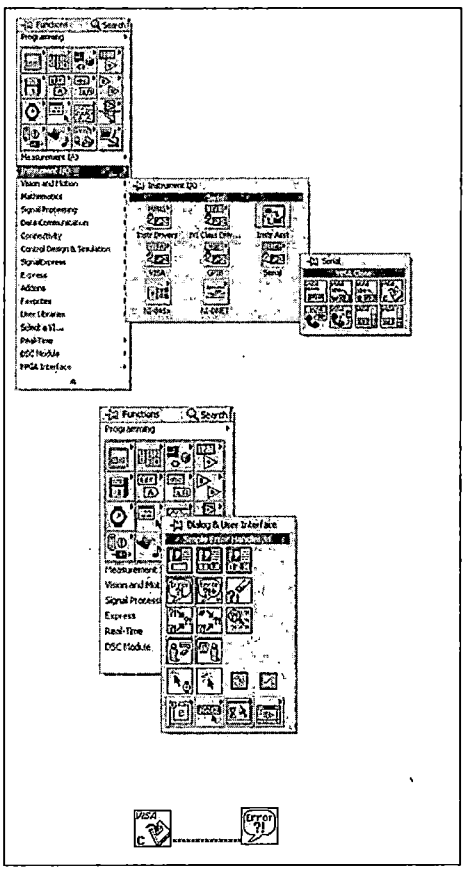


***Figura 4.5.-*** *Cierre y liberación del puerto*

## B. Escritura en el puerto serial

Primero, se inicializa una sesión VISA de la misma manera que se hizo al leer el puerto, con un "VISA Configure Serial Port". Luego, se coloca un "VISA Write" que se puede encontrar en Functions >> Programming >> Instrument I/O >> VISA Write.

Por último se cierra la sesión VISA con un "VISA close" y un "Simple Error Handler". El VI de escritura en puerto serial puede quedar como se muestra en la figura 4.6.

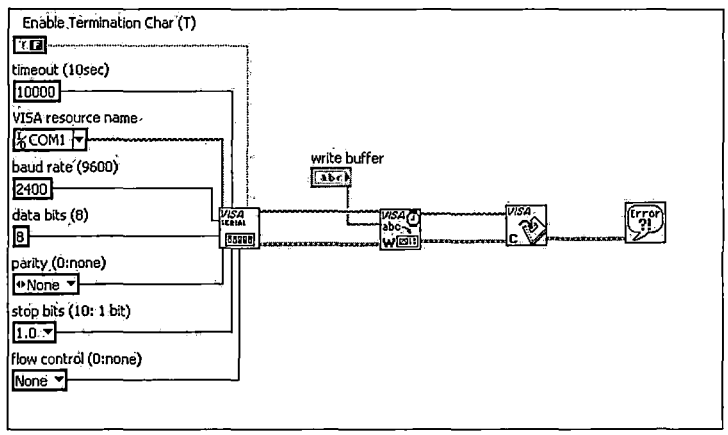



***Figura 4.6.-*** *Escribiendo al Puerto serial con una sesión VISA en LABVIEW*

Al igual que en la lectura del puerto serial, la velocidad de transferencia es primordial.

En ambos casos, es importante sincronizar las tareas de escritura y lectura respectivamente, de tal manera que el microcontrolador o la computadora estén listos para enviar o recibir un dato en el puerto serial. Para

ello se puede programar un lazo de control que no permita que el programa avance hasta que no se reciba cierto texto.

## 4.2. DESCRIPCIÓN GENERAL DEL SISTEMA

### 4.2.1. Organización

Para diseñar el software de control y supervisión se tendrán en cuenta lo siguiente:

- Seguridad.
- Jerarquía par la descarga de información y generación de reportes.
- Protocolo de comunicación.

El programa fue desarrollado en LABVIEW, para lo cual se crearon los siguientes VI's principales.

- Clave
- Inicio_final
- Reportes
- Configurar_final
- Descargar
- Supervisar_final
- Sensores

También se usó los siguientes subVI's:

- Nuevo_save_report
- Reemplazar
- Aumento1h
- Oper_orden3
- Variables

Todos los VI's principales y los subVI's trabajan en común, es decir, realmente el VI principal es "clave.vi" y el resto son subVI's, sin embargo, como cada uno de los 7 primeros VI's se visualizan en pantalla por eso es que se les están llamando VI's, y los 5 siguiente realizan operaciones que no se ven en pantalla, por eso les llamaremos subVI's.

Es importante también recordar que la mayoría de funciones en LABVIEW son también subVI's; pero como estos son de National Instrument, no se mencionan, sólo se mencionaron los creados para este proyecto.

De acuerdo al orden jerárquico de los VI's tendremos el diagrama mostrado en la figura 4.8.

### 4.2.2. Sistema de comunicación

LABVIEW dispone de muchos protocolos de comunicación; pero como se mencionó que el modelo de microcontrolador usado dispone del

RS232, entonces se dispondrá de ese protocolo, para lo cual se configuró como se muestra la figura 4.7.

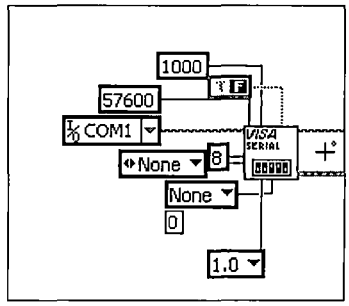



***Figura 4.7.-*** *Configuración del COM1*

- Puerto utilizado: COM1
- Velocidad: 57600 baudios
- Número de bits de datos: 8
- Bit de paridad: ninguno (None)
- Bits de stop: 1
- Control de flujo: no (None)

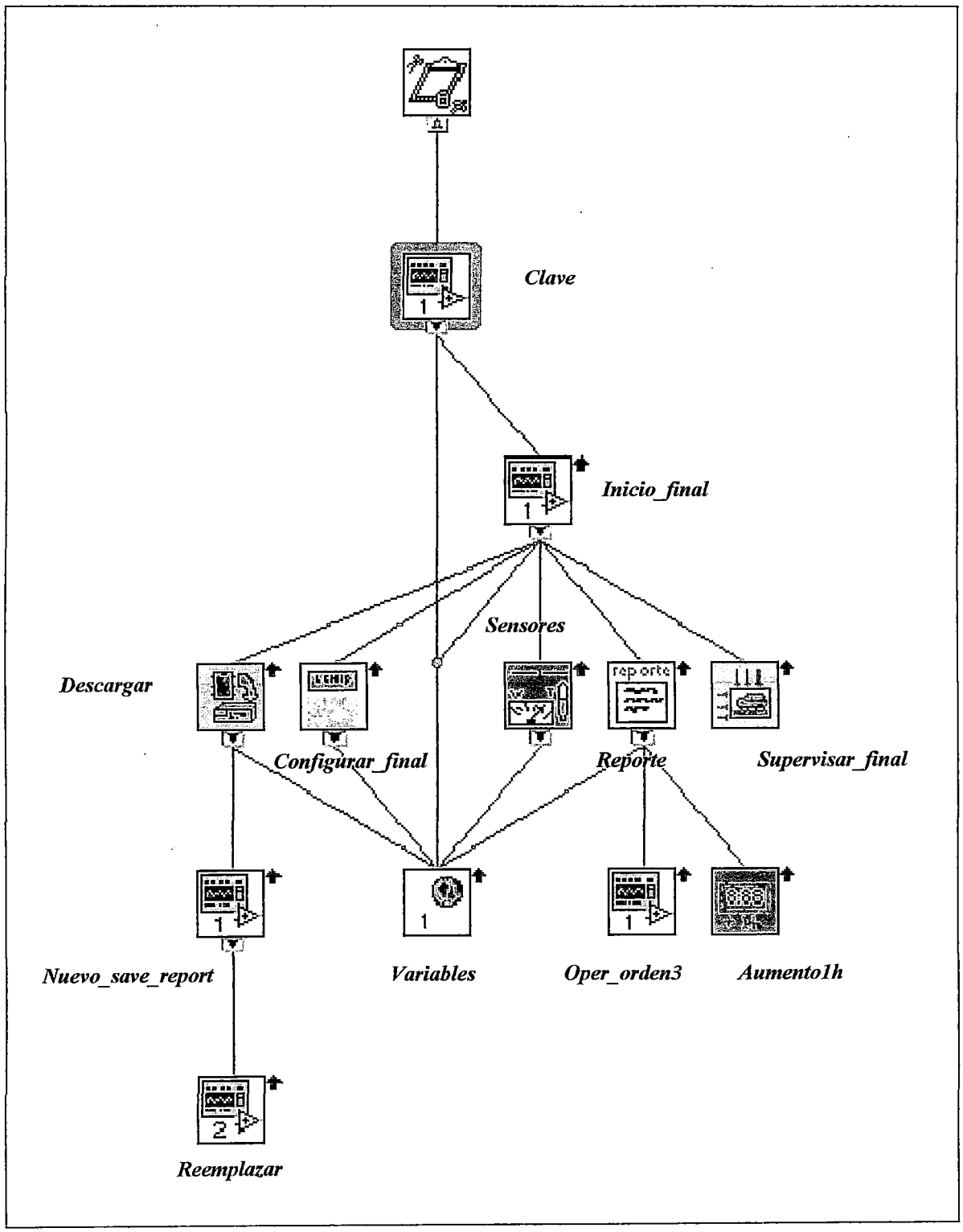



***Figura 4.8.-*** *Árbol de VI's y sub VI's del programa*

## 4.3. DESCRIPCIÓN DEL FUNCIONAMIENTO DEL SOFWARE

De acuerdo a las funciones que cumple podremos dividirlo en las siguientes partes:

- Acceso mediante clave
- Configuración de la tarjeta de control 
- Supervisión en tiempo real 
- Descarga de información 
- Verificación de sensores 
- Generación de reportes 


### 4.3.1. Acceso mediante clave

Como todo software de programación, aún con mayor razón siendo para una aplicación de ingeniería, el primer VI principal llamado "clave", fue diseñado para poder acceder al sistema con diferentes tipos de usuario:

- Invitado
- Administrador
- Mantenimiento

Donde para cada usuario se crearon sus propias claves, así como las funciones que el sistema le permitirá acceder dependerá del usuario seleccionado. La apariencia del panel frontal se muestra en la figura 4.9, y el código se muestra en la figura 4.10.

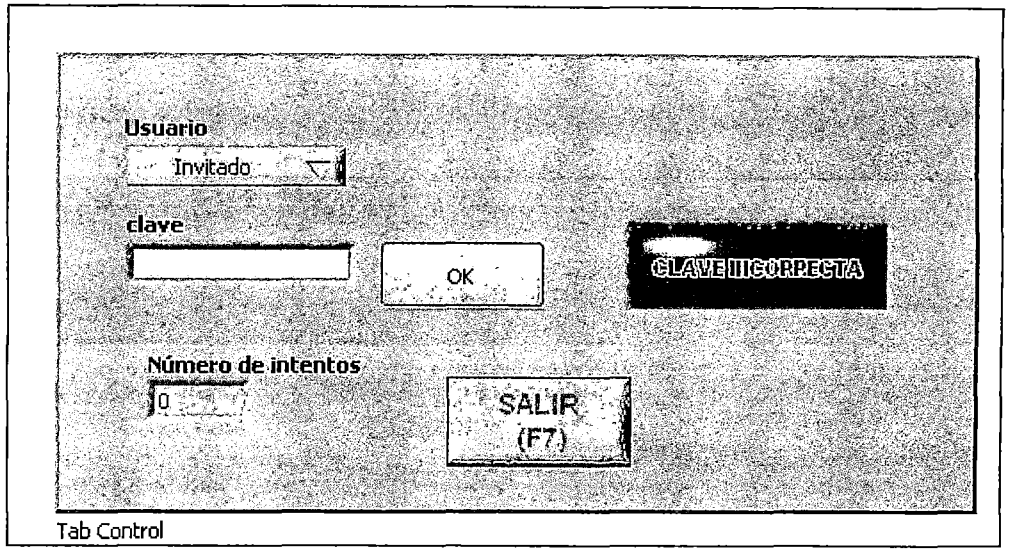



***Figura 4.9.-*** *Panel principal de acceso al programa*

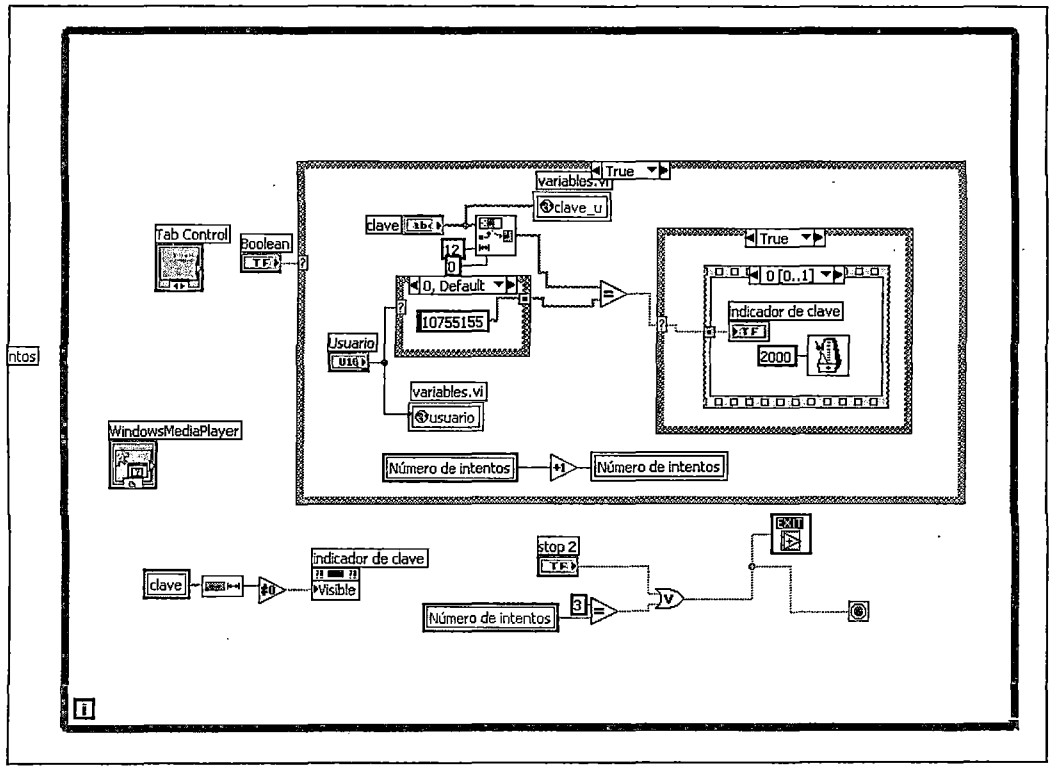



***Figura 4.10.-*** *Panel de diagrama del panel principal de acceso al programa*

En todo el sistema se usarán las siguientes variables globales:

- Usuario 


A cada usuario se le asignó una clave y un número de acuerdo al tipo, esto se muestra en la figura 4.11.

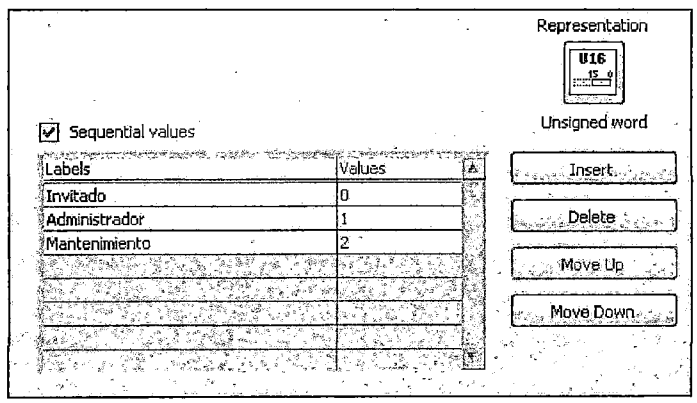
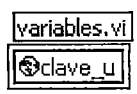


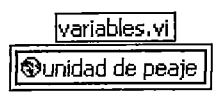

***Figura 4.11.-*** *Configuración del menú de acceso para usuarios*

- Clave_u 


Así también se le asignaron claves para cada tipo de usuario

- Lectura 


Por seguridad hay algunas funciones especiales, más superiores aún del mantenimiento, esto con la intención de proteger más el sistema y que un usuario no autorizado pueda afectar al sistema y las tarjetas.

- Unidad de paje 


Dependiendo del nombre del peaje se dispondrá de este dato a los largo de todo el sistema, sobre todo en la generación de reportes.

Luego saldrá, si la clave es correcta correspondiente al tipo de usuario como lo muestra la figura 4.12.

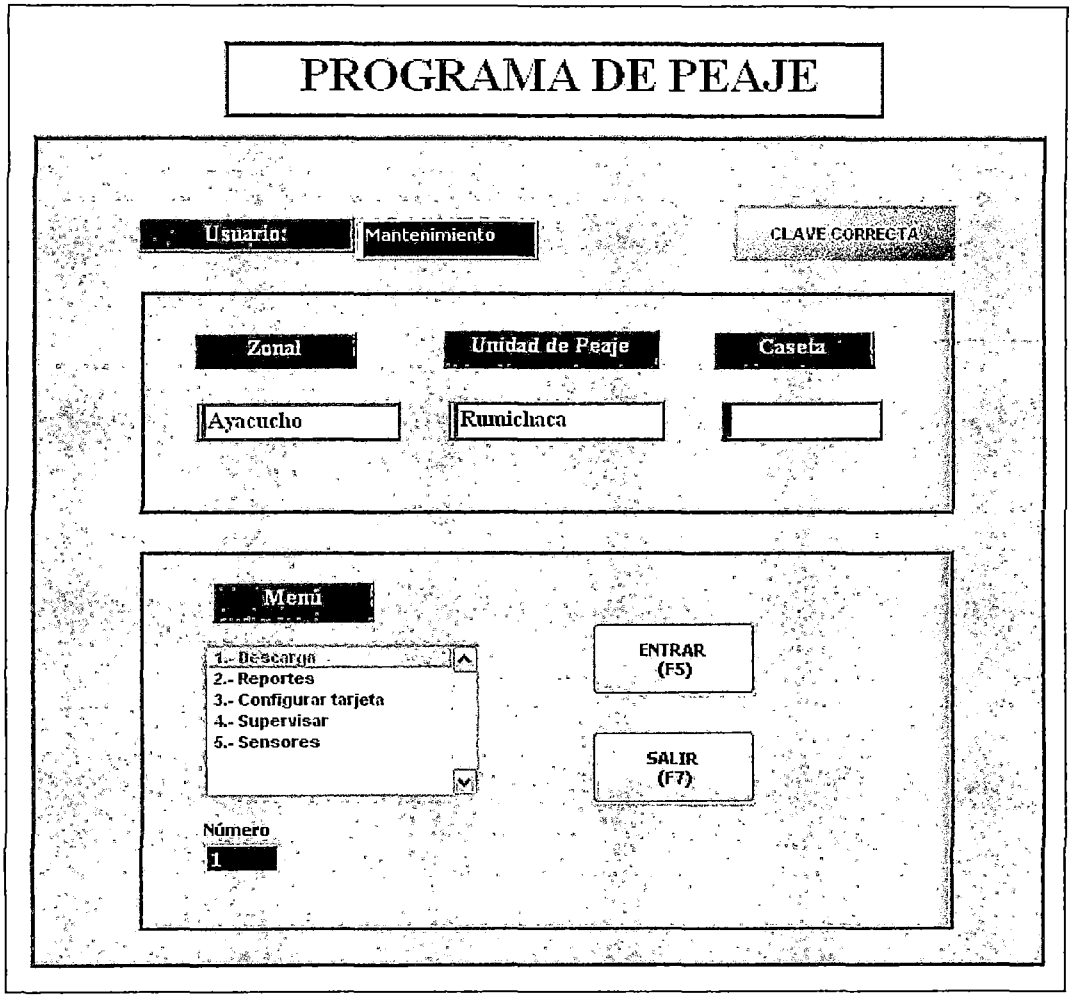



***Figura 4.12.-*** *Panel de menú de opciones para generación de reportes y mantenimiento*

### 4.3.2. Usuarios y niveles de seguridad

El menú que se muestra cambiará para diferentes usuarios:

**Invitado:** realmente esta puede ser una persona encargada por la empresa para descargar la información, pero no con la jerarquía del administrador, siendo las funciones disponibles para el menú:

1.-Descargar

2.-Reportes

**Administrador:** esta es la persona que estará a cargo del peaje por lo tanto tendrá las siguientes funciones en el menú:

1.-Descargar

2.-Reportes

3.-Configurar tarjeta

4.-Supervisar

**Mantenimiento:** esta es la persona encargada de dar el mantenimiento al equipo por lo cual dispondrá del siguiente menú:

1.-Descargar

2.-Reportes

3.-Configurar tarjeta

4.-Supervisar

5.-Sensores

No existe el peligro de que un usuario acceda a una función que no corresponda, ya que simplemente de acuerdo a la clave se mostrará sólo el

menú correspondiente, para esto serán útiles las variable globales, ya que este VI se puede leer. El código respectivo será mostrado en la figura 4.13.

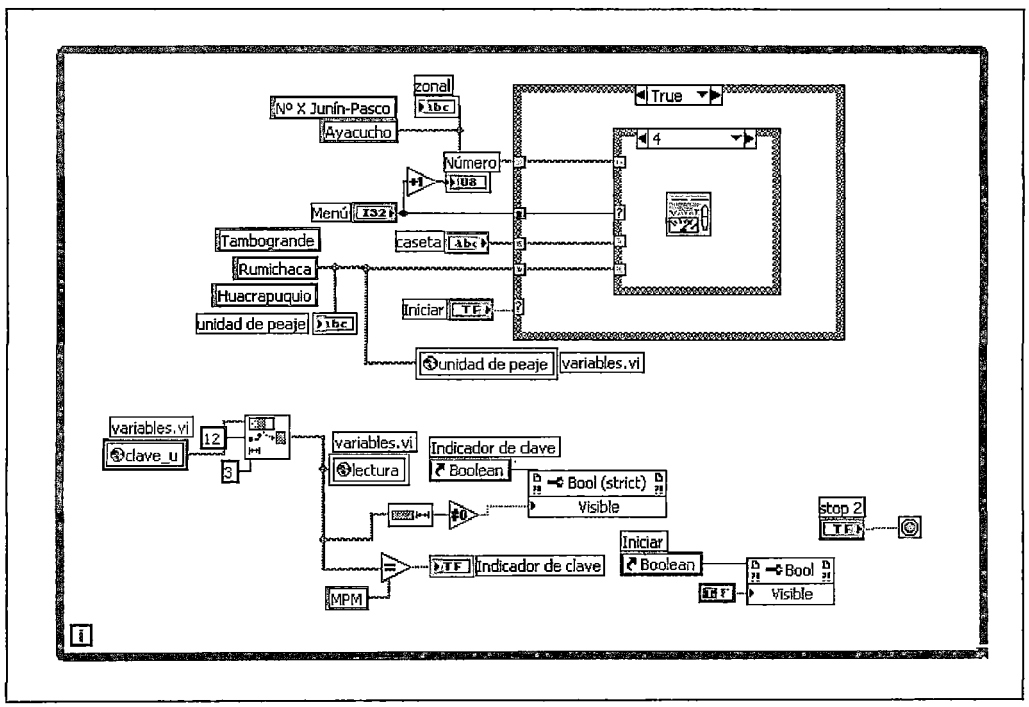



***Figura 4.23.-*** *Panel de diagrama de bloques del menú de opciones para los diferentes usuarios*

## 4.4. CONFIGURACIÓN DE LA TARJETA DE CONTROL

Para esto se accede al VI "configurar_final"  . A esta función sólo tienen acceso dos tipos de usuario: administrado y mantenimiento. Sin embargo los dos sólo pueden modificar la fecha y hora; pero existe una función especial para el ingeniero que va a instalar el equipo, y estas funciones tienen que ver con cuántas memorias va a usar en la tarjeta, de qué capacidad son y a partir de que posición se empezará a guardar la información, por lo tanto la apariencia del panel frontal cambiará, de acuerdo al gráfico de la figura 4.14.

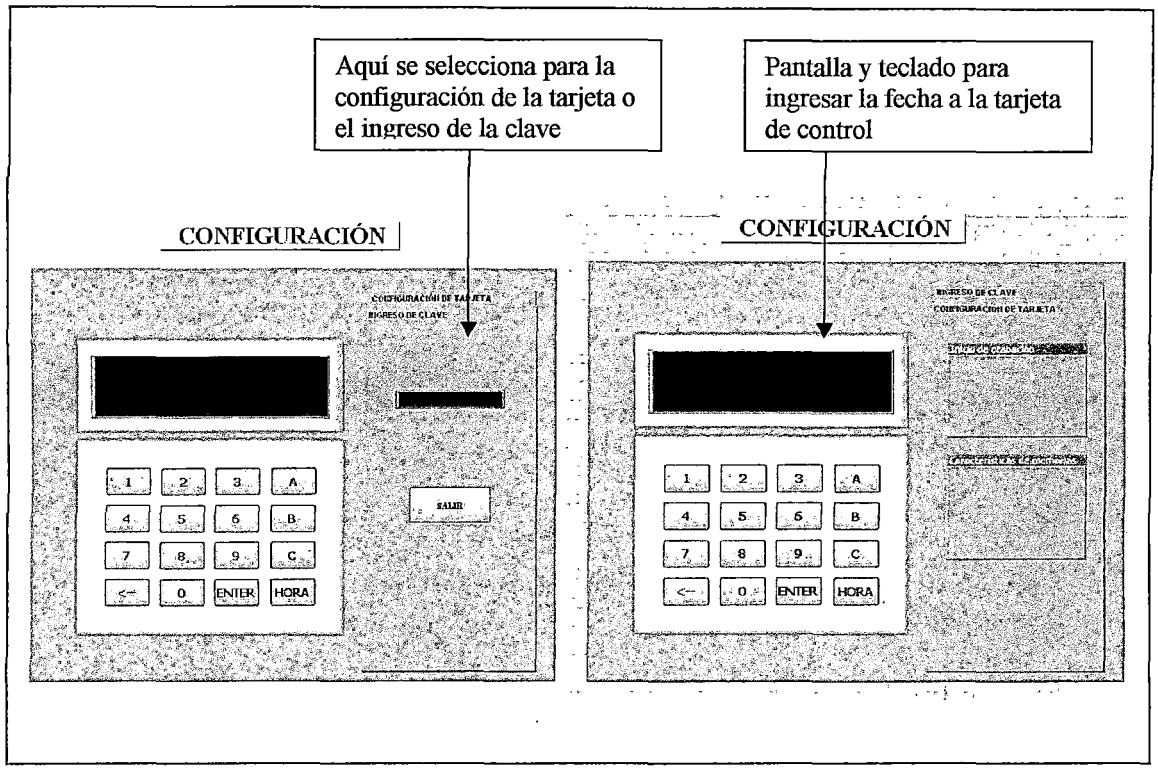



***Figura 4.14.-*** *Panel frontal para la configuración de la tarjeta de control, ingreso de fechas y configuración de memorias*

Como se puede apreciar, se podrá ingresar la fecha y hora, sin embargo, no se tendrá acceso a configuración que tienen que ver con el funcionamiento de las memorias. Para esto existe una clave especial, que es la misma del usuario "mantenimiento" pero con unos caracteres adicionales. Por seguridad aquí se vuelve a pedir la clave del usuario para confirmar que sea la misma que fue ingresada anteriormente.

Para el ingeniero que tiene acceso a estas claves se dispondrá de las funciones adicionales que muestra en la figura 4.15.

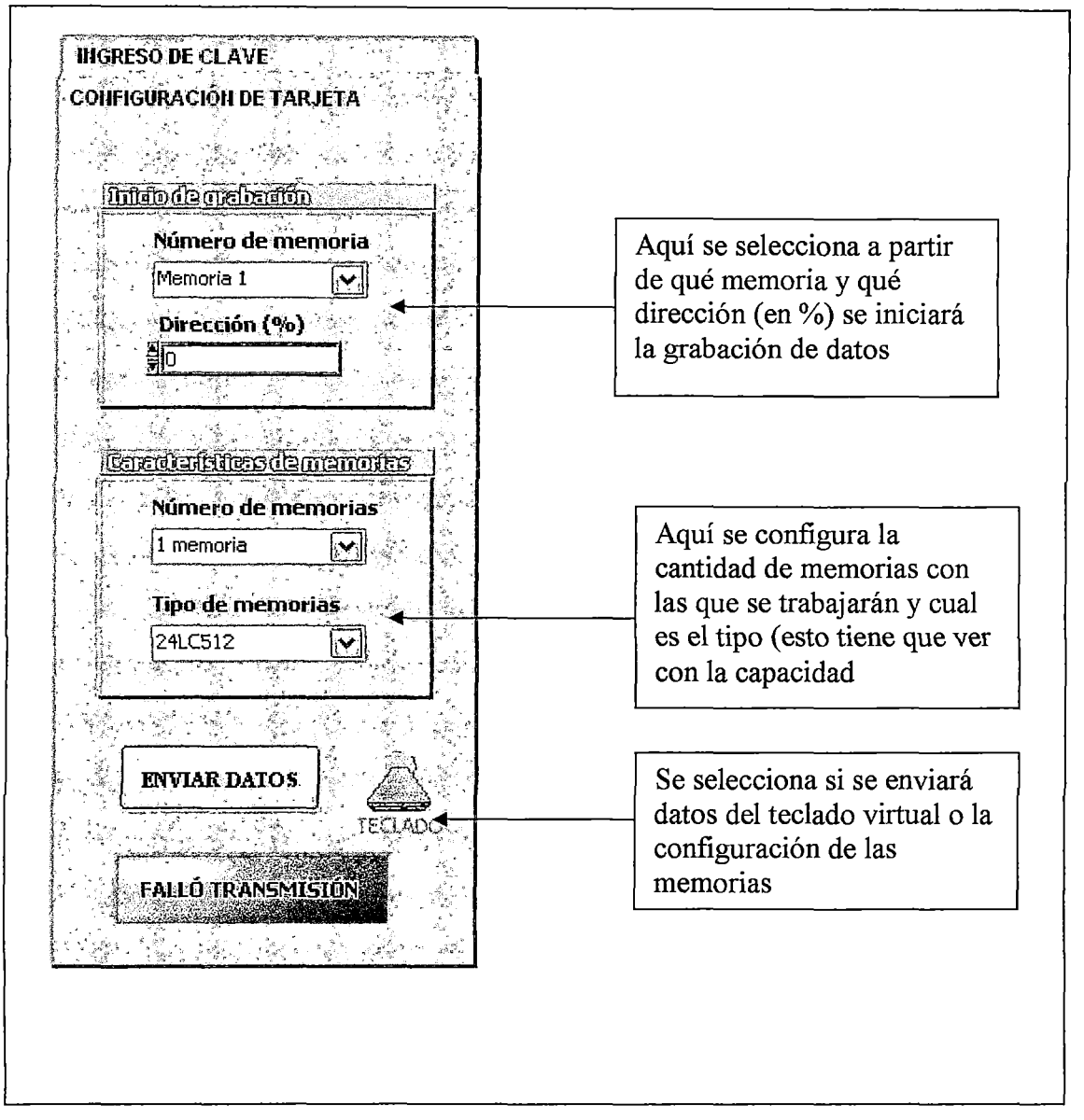



***Figura 4.15.-*** *Tab Control de la configuración de las memorias a utilizar*

Toda la información del PC se enviará mediante el protocolo de comunicación RS232 y su diagrama de bloques se ve en la figura 4.16.

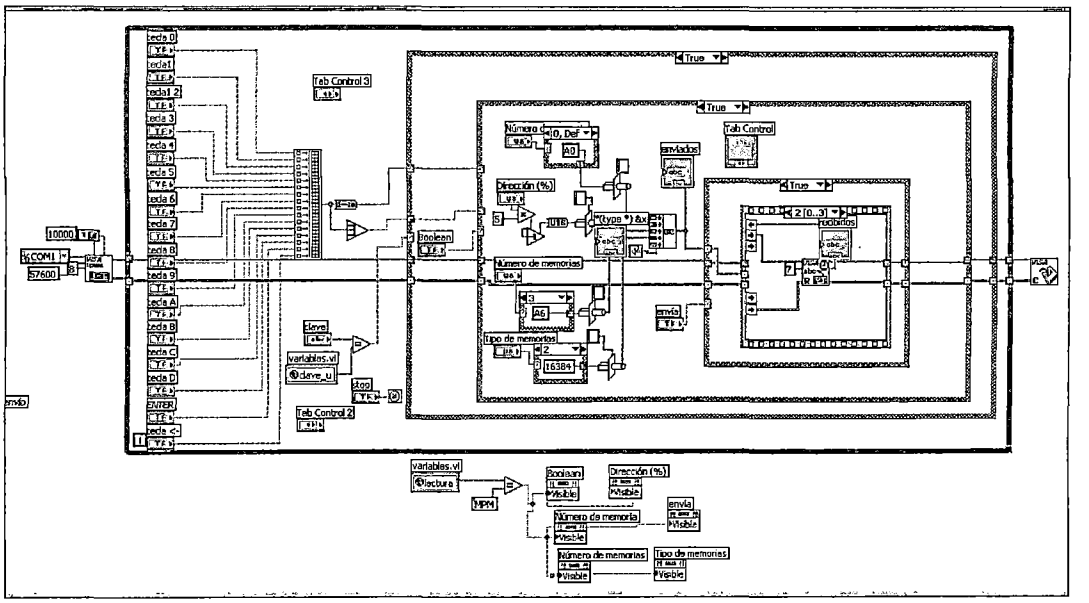


*Figura 4.16.- Panel de diagrama de bloques de la configuración de la tarjeta, ingreso de fecha y memorias*

## 4.5. SUPERVISIÓN EN TIEMPO REAL

A esta función pueden acceder los usuarios "administrador" y "mantenimiento", los cuales se podrá supervisar el funcionamiento de la tarjeta de control en tiempo real de la siguiente manera:

- Detección y visualización de las características del vehículo que pase por el peaje, así como se mostrará la fecha, hora, minuto y segundo en que pasó la unidad.
- Estado de las memorias; es decir, cantidad disponible y llena.
- Clasificación de los datos recibidos.

En el panel frontal, visto en la figura 4.18, se pueden apreciar 4 partes:

- Datos
- Estado de memoria

- Botones
- Datos en tiempo real

### 4.5.1. Datos

Son los datos del peaje cargados en la pantalla principal del menú, tales como zonal, unidad de peaje y caseta.

### 4.5.2. Estado de memoria

Aquí se muestra, cada vez que pasa un vehículo, en qué memoria y en qué parte de ella se encuentran los datos, esto es importante sobre todo para mantenimiento, así cualquier cambio de memoria que se haga podría ser cualquiera dependiendo de las decisiones que se tomen para diferentes situaciones. Como ejemplo de esto podría decirse que si estuvieran los últimos datos en la mitad de la segunda memoria, entonces este estado se muestra en la figura 4.17.

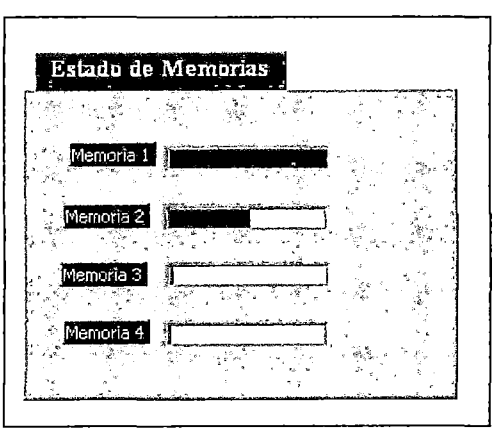



***Figura 4.17.-*** *Indicador gráfico que refleja la cantidad de memoria usada y disponible en la tarjeta*

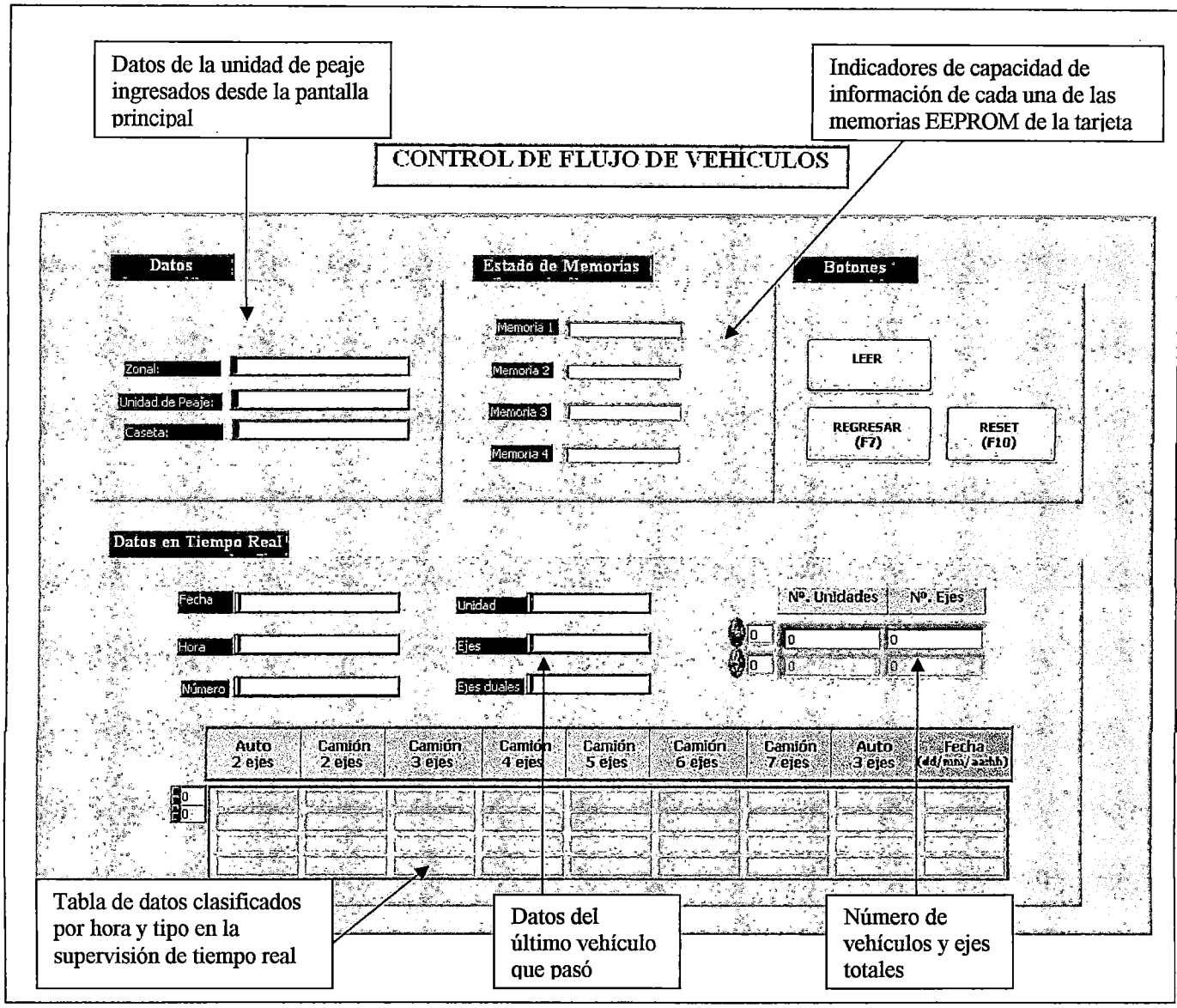



***Figura 4.18.-*** *Panel frontal para la supervisión en tiempo real de las unidades y del estado de las memorias eeprom*

Por supuesto debemos recalcar que esto está en porcentaje de acuerdo a un factor de conversión dependiendo del tipo de memorias usadas. Se pueden apreciar 3 botones:

**Regresar.-** Para que retorne a la pantalla anterior.

**Reset.-** La PC siempre recibe una cadena de datos con un tamaño definido cada vez que pasa una unidad vehicular, y en un tiempo definido a partir desde el momento en que el vehículo empieza a pasar; pero si por alguna razón no recibe los caracteres completos, para evitar que los datos se desorganicen, entonces se puede usar este botón, si fuera necesario, para restablecer el orden de los datos.

**Leer.-**Este es el botón más importante, ya que con ella empieza la supervisión en tiempo real. Con esta función cada vehículo que pase se registrará y se clasificará de acuerdo al tipo de vehículo y a su cantidad de ejes en la tabla de la figura 4.19.

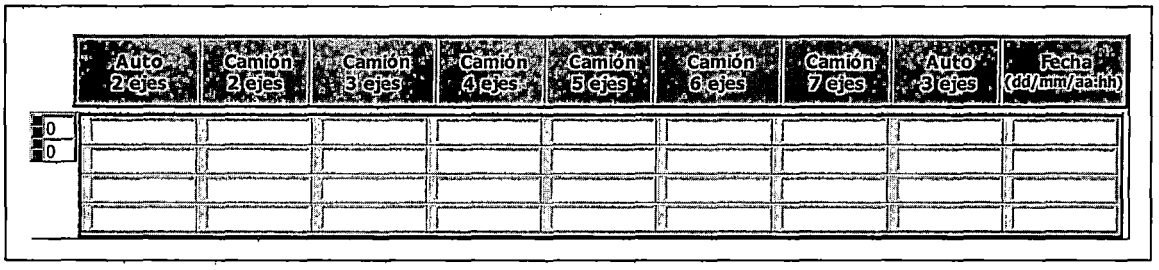



***Figura 4.19.-*** *Tabla de almacenamiento y clasificación de unidades, para la supervisión en tiempo real*

Esta tabla almacenará el total por hora de acuerdo al tipo de vehículo, a partir del momento en que se pulsó leer, cada fila representa una hora. También Se puede tener la cuenta total de vehículos, así como la cantidad de ejes totales, ya que por experiencia este dato puede conllevar a conclusiones

importantes a la hora de hacer el análisis en la administración central. El total de unidades y ejes se muestra a continuación en la figura 4.20. Todo el algoritmo de programación se muestra a continuación en la figura 4.21.

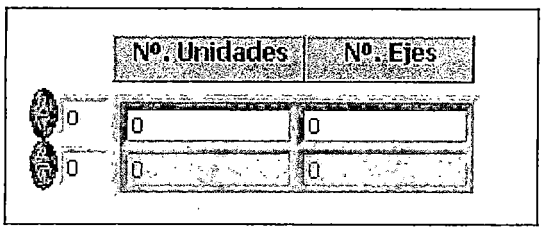



***Figura 4.20.-*** *Tabla del total de número de ejes y unidades sin clasificar, desde el momento en que se puso la supervisión en tiempo real*

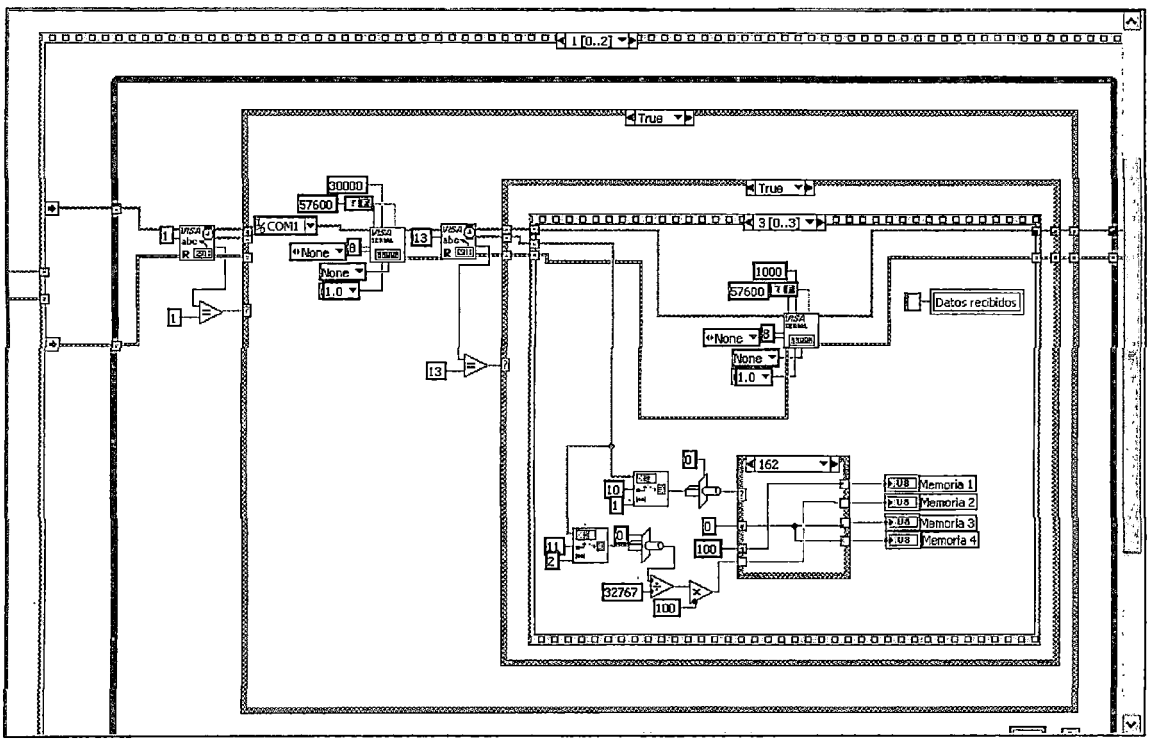



***Figura 4.21.-*** *Diagrama de bloques de la supervisión en tiempo real*

## 4.6. PROCESO DE COMUNICACIÓN Y TRANSMISIÓN DE DATOS

El botón **Leer** es un switch, es decir, se queda enganchado una vez pulsado y no regresa, por lo que se mantendrá supervisando hasta que se vuelva a pulsar el botón para desengancharlo.

Lo primero que debemos recordar es que si el microcontrolador recibe el carácter "s", empezará a mandar datos cada vez que pase un vehículo; por lo tanto este debe ser el primer caracter que envíe. Después de esto el sistema se quedará en un bucle permanente hasta que se vuelva a presionar el botón **Leer**, momento en el cual el programa enviará el caracter de finalización "c".

La PC siempre recibirá 14bytes. El primero es el carácter 's' que indica el inicio del paso del vehículo, y los 13 restantes realmente contiene la información de importancia, está cadena de datos será:

| Año | Mes | Día | Hora | Minuto | Segundo | Unidad | número de ejes | número de vehículo | Número de memoria | Dirección dentro de memoria |
|---|---|---|---|---|---|---|---|---|---|---|

- Año (1byte): en hexadecimal
- Mes (1byte): en hexadecimal
- Día (1byte): en hexadecimal
- Hora (1byte): en hexadecimal
- Minuto (1byte): en hexadecimal
- Segundo (1byte): en hexadecimal

- Unidad (1byte): formato carácter. 'L' para ligero o auto, 'P' para pesado o camión.
- Número de ejes (1byte): en hexadecimal
- Número de vehículo (2bytes): en hexadecimal
- Número de memoria (1byte): pudiendo ser A0h, A2h, A4h, A6h
- Dirección dentro de memoria (2bytes): depende de la capacidad de la memoria.

Para esto en LABVIEW se genera una secuencia mostrado a continuación en la figura 4.22.

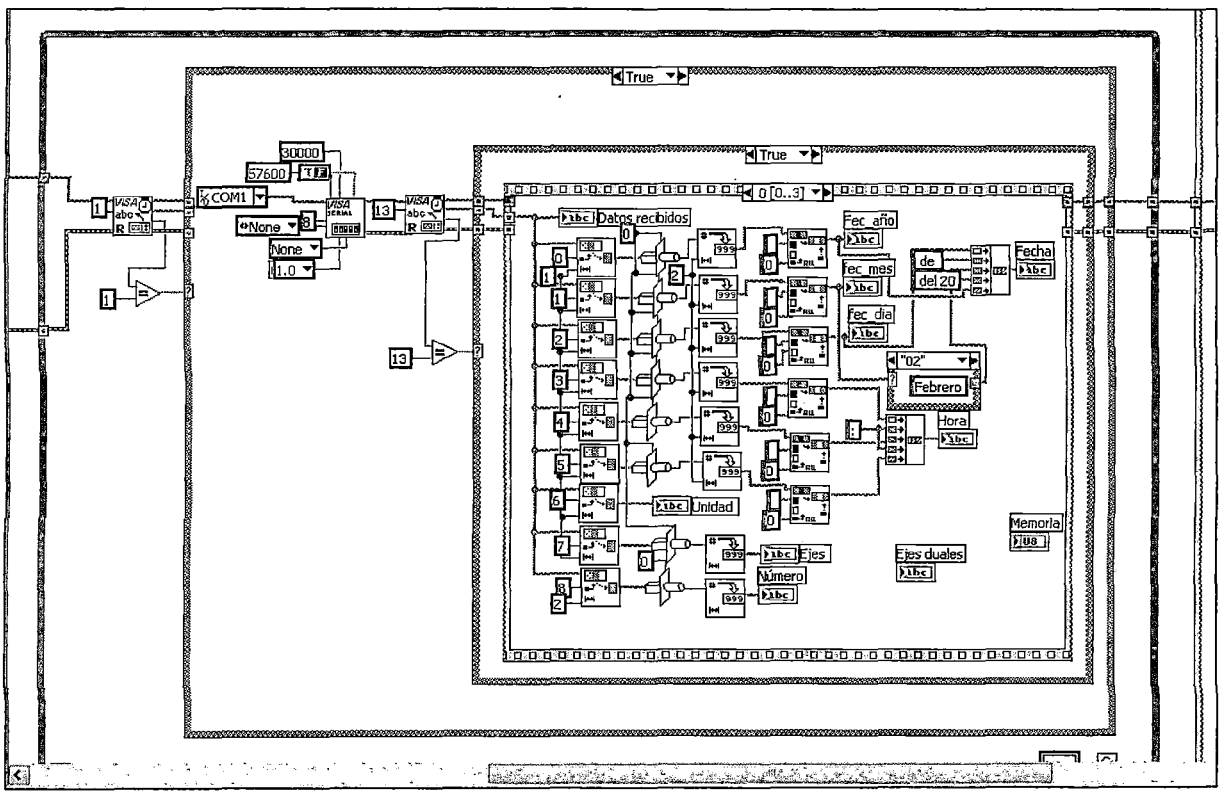



***Figura 4.22.-*** *Panel de diagrama de bloques de la recepción y decodificación de los 14 caracteres recibidos de la tarjeta de control*

Se puede apreciar el ordenamiento de la cadena de datos recibidos. Ahora se realizarán 3 secuencias más. En la figura 4.23 se muestra la rutina para comparar si la hora del vehículo que recién pasa y el anterior son iguales; si no son, se añade una nueva fila a la tabla, el cual será una nueva hora para almacenar la cuenta.

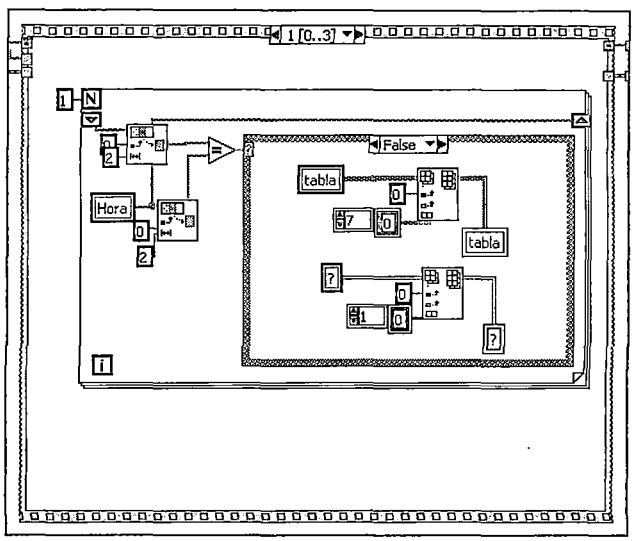



***Figura 4.23.-*** *Especificación del diagrama de bloques para la clasificación y ordenamiento de datos en las tablas*

Aquí se usa un "case" para poder saber a cual de las columnas correspondientes del tipo de vehículo tiene que sumarle una unidad. Todo se trabaja con arreglos bidimensionales en caracteres y decimal para poder hacer el incremento numérico. Esto lo apreciamos en la figura 4.24.

Ahora mostraremos el algoritmo, que en base a la información del número de memoria y la dirección, se usará para mostrar (por medio de las barras indicadoras) hasta qué parte de las memorias se encuentra lleno de información. (Ver figura 4.25)

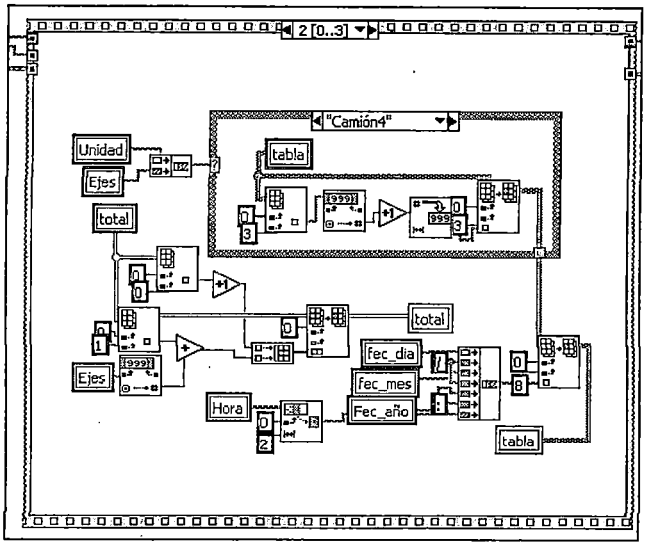



***Figura 4.24.-*** *Especificación del diagrama de bloques para la clasificación y ordenamiento de datos en las tablas, para el cambio de hora*

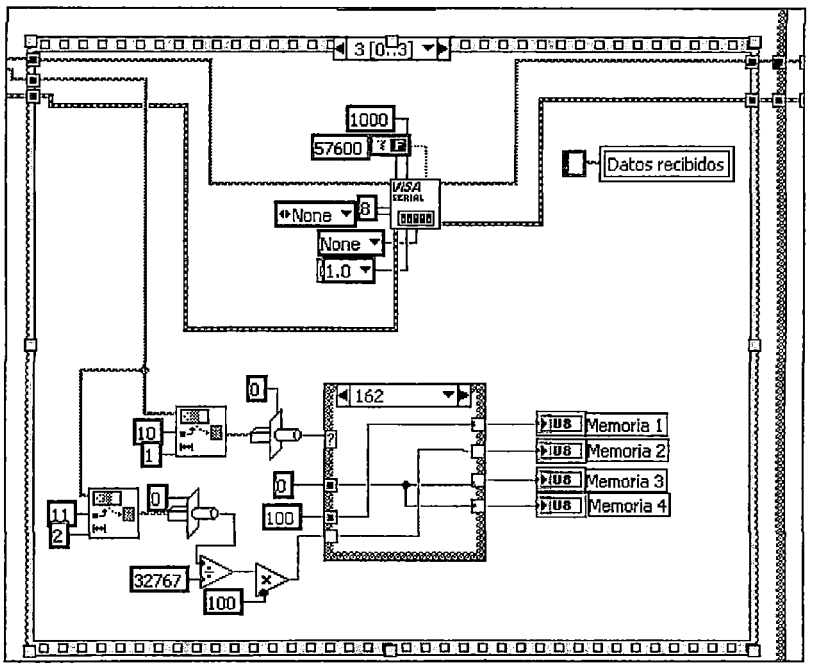



***Figura 4.25.-*** *Diagrama de bloques para la lectura de la capacidad de las memorias, tanto el espacio usado como el disponible, de acuerdo los tipos de memorias configurados en las funciones anteriores*

## 4.7. DESCARGA DE INFORMACIÓN

Esta es una de las más importantes funciones, ya que a partir de aquí se generarán los reportes. En primer lugar por seguridad se vuelve a pedir la clave de acceso, dependiendo del usuario, en el cual hasta entonces no aparecerá ningún botón para descargar. Esta pantalla tendrá la apariencia como se muestra en la figura 4.26.

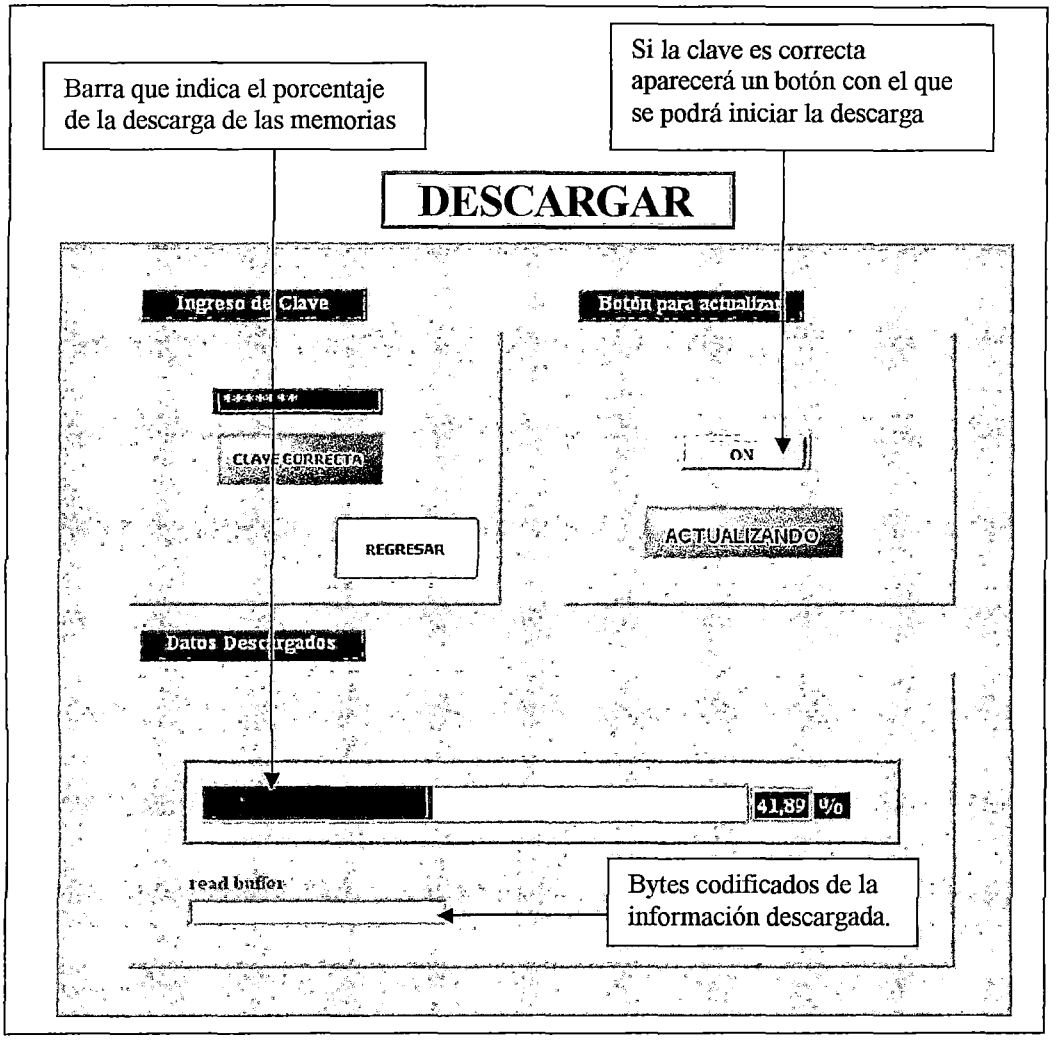



***Figura 4.26.-*** *Panel frontal para la descarga de información de la tarjeta de control a la PC*

Sólo poniendo la clave correcta se podrá mostrar el botón para acceder a función de descargar datos desde las memorias. Para esto el programa creará un archivo como para una base de datos personal. y funciones para reportes. De todas las formas en las que LABVIEW dispone para trabajar y guardar archivos, entre las cuales son: archivo de texto, word, excel, html, etc. Usaremos "html"; pero para proteger la información de este archivo se guardará codificado, así aunque alguien lo abra para entenderlo, no podrá, sino sólo mediante el programa para reportes creado en LABVIEW, el cual decodificará la información para luego presentarlo en formato "html".

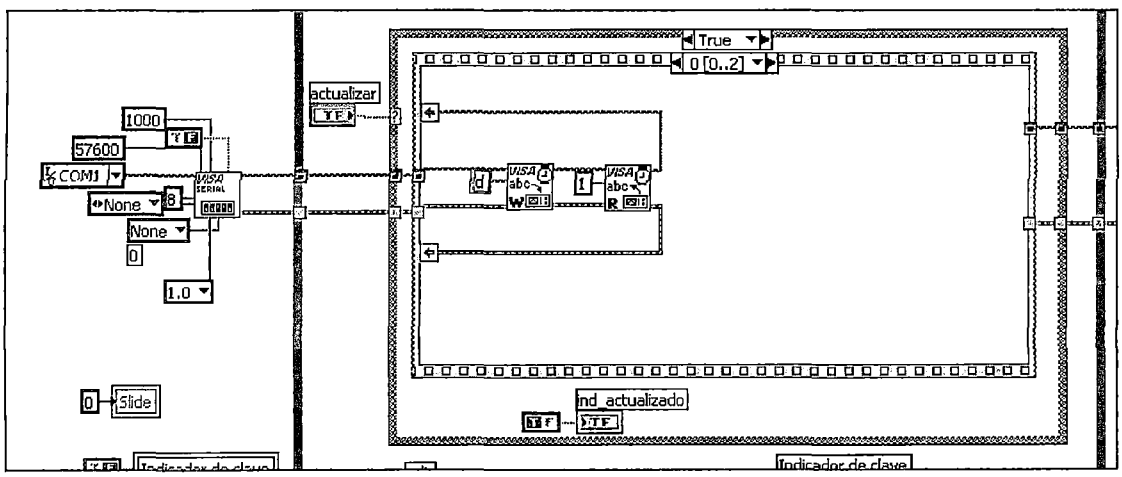



***Figura 4.27.-*** *Diagrama de bloques del inicio de la descarga de la información*

A continuación se explicará la forma en la que se decodifica:

- Primero el programa envía el caracter 'd' al microcontrolador que le indica la función de descargar. Esto se observa en la figura 4.27. A continuación hay dos pasos más, el primero es el más importante, el cual consiste en enviar los datos en grupos de 8bytes, esto lo vemos en la figura 4.28.

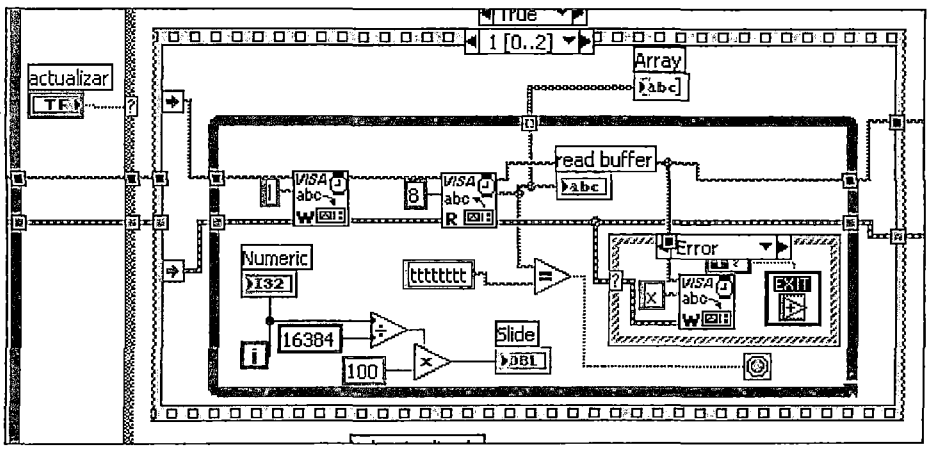



***Figura 4.28.-*** *Diagrama de bloques del proceso de comunicación y descarga de la información de la tarjeta de control*

- Existe un caracter ('l') para control de flujo el cual el programa enviará cada vez que le diga al microcontrolador que envíe un grupo de 8bytes. Estos grupos de 8bytes se indexan en un array (arreglo), indicando el fin del arreglo una cadena de 8bytes de puros caracteres 't', esto es "tttttttt", que enviará el microcontrolador. Recibido esta cadena de final de envío de datos, se pasará a la etapa de codificación. La otra posible causa para que finalice el programa de descarga es algún error en la comunicación serial, el cual en el programa se resolverá enviando el caracter 'x', indicándole al microcontrolador que hubo un error e inmediatamente se cerrará el programa y el microcontrolador saldrá de la rutina de descarga volviendo al funcionamiento normal. El panel de diagrama de bloque se observa en la figura 4.28.
- Finalmente llega la etapa de codificación, en la cual lo único que se hace es pasar a convertir el arreglo al formato "html", que como los datos son pasados desde el microcontrolador en su mayoría en hexadecimal, entonces se verán a la hora de abrirse como caracteres desconocidos. (Ver figura 4.29)

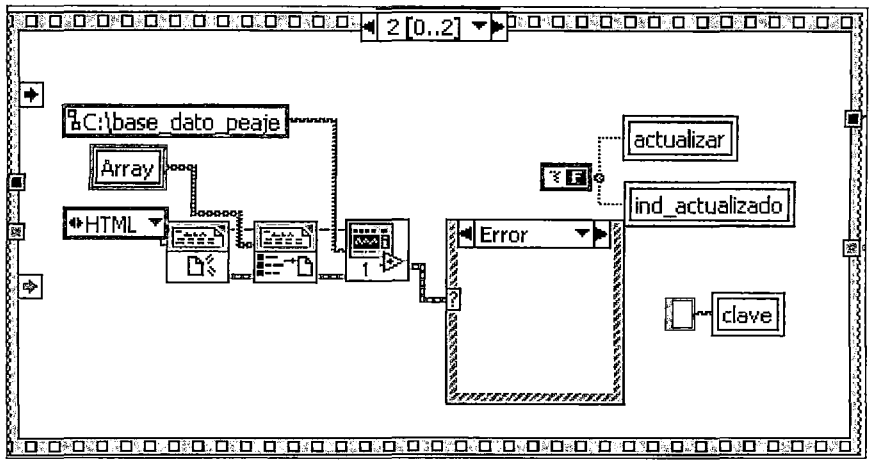



*Figura 4.29.- Diagrama de bloques para almacenamiento de la información en disco duro*

## 4.8. VERIFICACIÓN Y PRUEBA DE SENSORES

A esta función sólo tiene acceso el usuario "mantenimiento", sin embargo por seguridad, realmente se tiene que añadir una clave especial para hacer funcionar esta función, o de lo contrario sólo podrá visualizarlo. En esta operación se podrán realizar 2 tipos de pruebas:

- Verificación de los sensores de presión e inductivo.
- Lectura de parámetros de temperatura, voltaje de baterías y estado de la puerta del tablero de control.

### 4.8.1. Verificación de los sensores

En la figura 4.30 se puede apreciar la ventana correspondiente a estas funciones. Se puede apreciar la disposición de los sensores, aquí es el programa quien realizará una inspección de los sensores y verificará la secuencia correcta, independiente del conteo de la tarjeta, mostrando el tipo de vehículo que pasó, para esto dispone de una rutina para clasificar el auto y

además se podrá ver el momento en el que los sensores se van activando en tiempo real, en tanto que por su cuenta la tarjeta de control guarde la información clasificada. Aquí se podrá ver si los sensores están respondiendo correctamente o si de repente algún sensor no está funcionando correctamente, para su posterior mantenimiento.

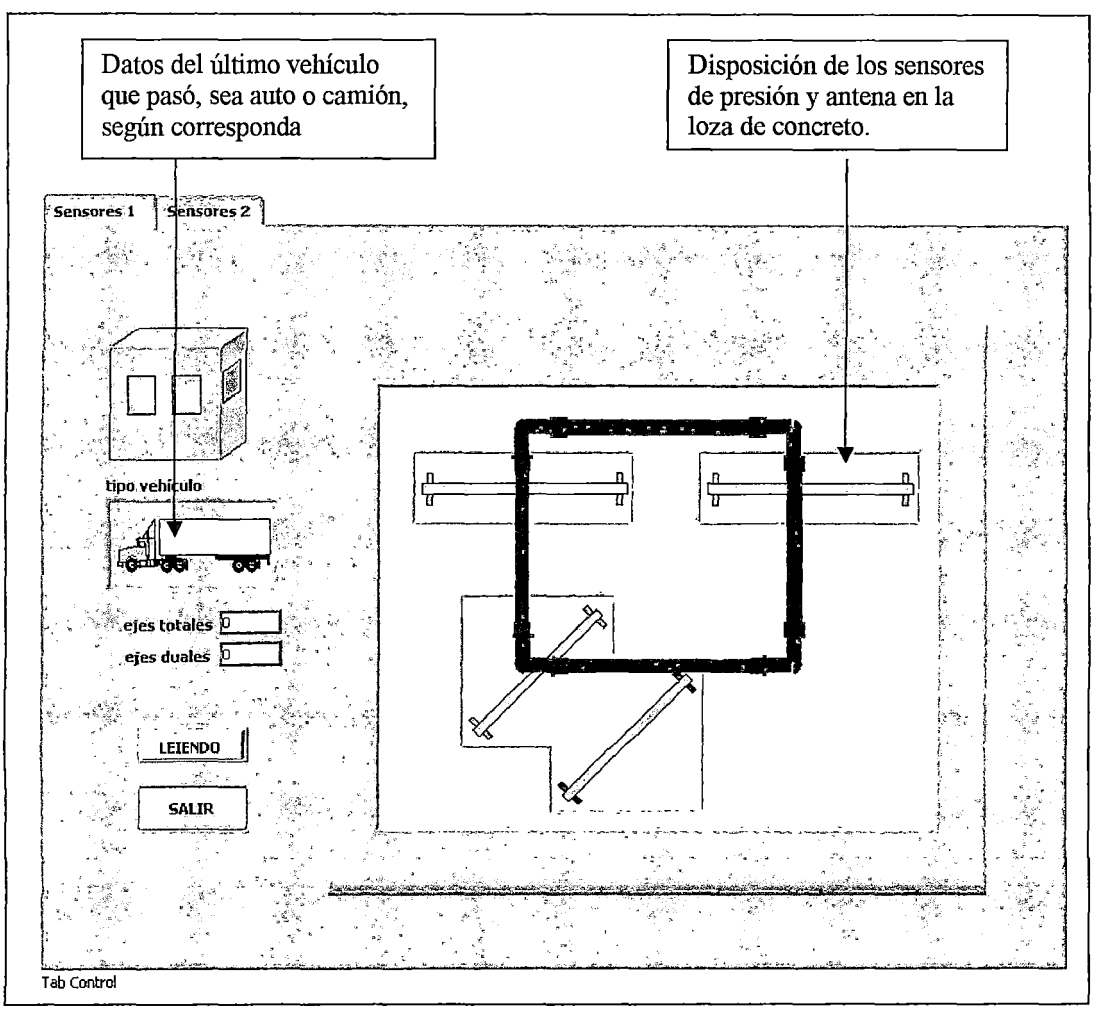



***Figura 4.30.-*** *Panel frontal para las pruebas de verificación del estado de sensores*

El algoritmo que se usa y la comunicación son simples. Cada ciertos milisegundos el programa enviará el caracter 'e' para lo cual el

microcontrolador responderá con un byte en el que se encuentra el estados de los sensores mencionados. Esta información será procesada a nivel de bits ya que LABVIEW tiene las herramientas para este ello. Se mostrará en la pantalla la información procesada de acuerdo al tipo de vehículo que pasó. El panel de diagrama de bloques de esta etapa se muestra en la figura 4.31.

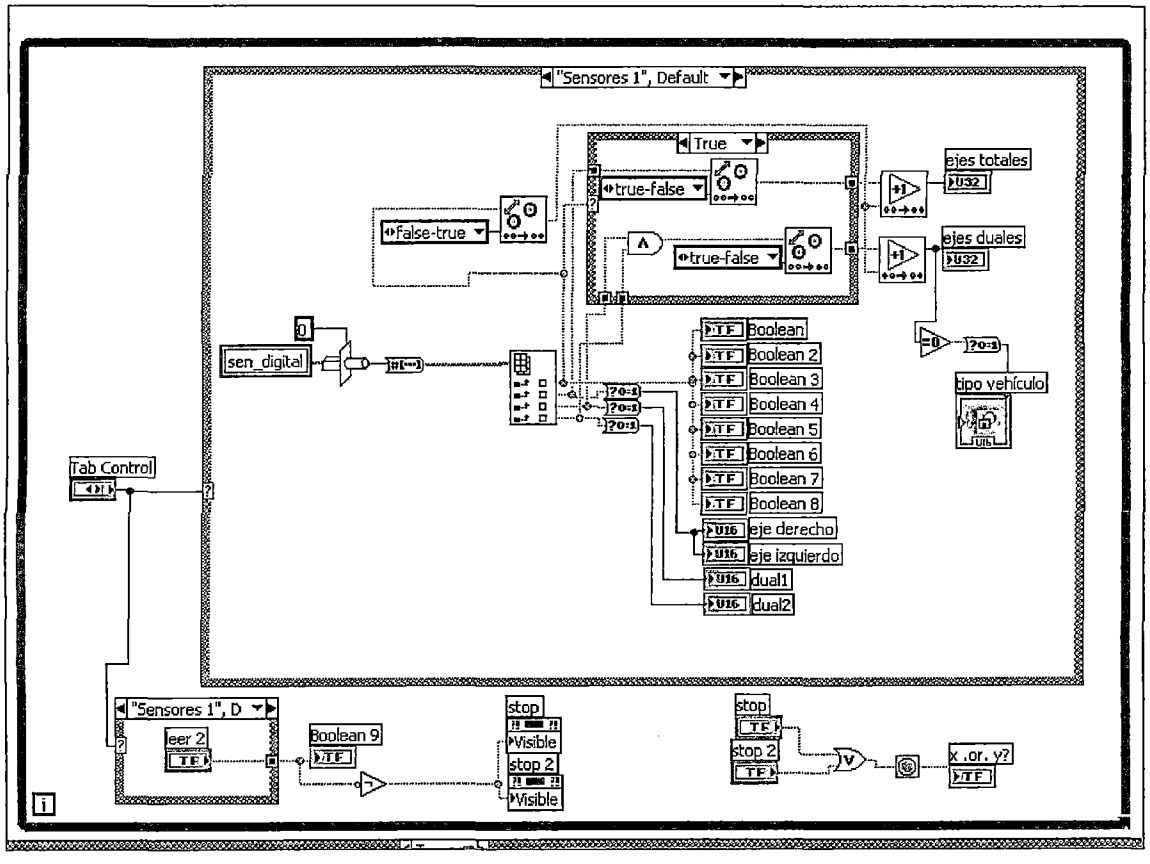



***Figura 4.31.-*** *Diagrama de bloques del programa de verificación del estado de sensores*

### 4.8.2. Lectura y verificación de parámetros importantes

Aquí se puede apreciar una función especial que es la lectura de los siguientes parámetros:

- Temperatura (Señal analógica)
- Voltaje de baterías (señal analógica)
- Estado de la puerta del tablero de control (señal digital)

Por tratarse de señales analógicas entonces la información irá en 3 bytes. Cada vez que el programa envíe el caracter 't', el microcontrolador enviará 3bytes.

- El primer byte contiene el estado de la puerta.
- El segundo byte contiene el valor de la temperatura.
- El tercer byte contiene el valor del voltaje de la batería.

Realmente esta función está preparada cuando se proponga el sistema de monitoreo a distancia, el cual se presentará posteriormente. El panel frontal se puede apreciar en la figura 4.32 y el panel de diagrama de bloques en la figura 4.33.

Aquí se puede apreciar la existencia de dos lazos de control "do-while" en paralelo. Una es usada para la comunicación serial y no perder la atención en ello, la otra se encarga de ordenar los datos para presentarlos. Para poder controlar variables de un lazo a otro se usan variables locales.

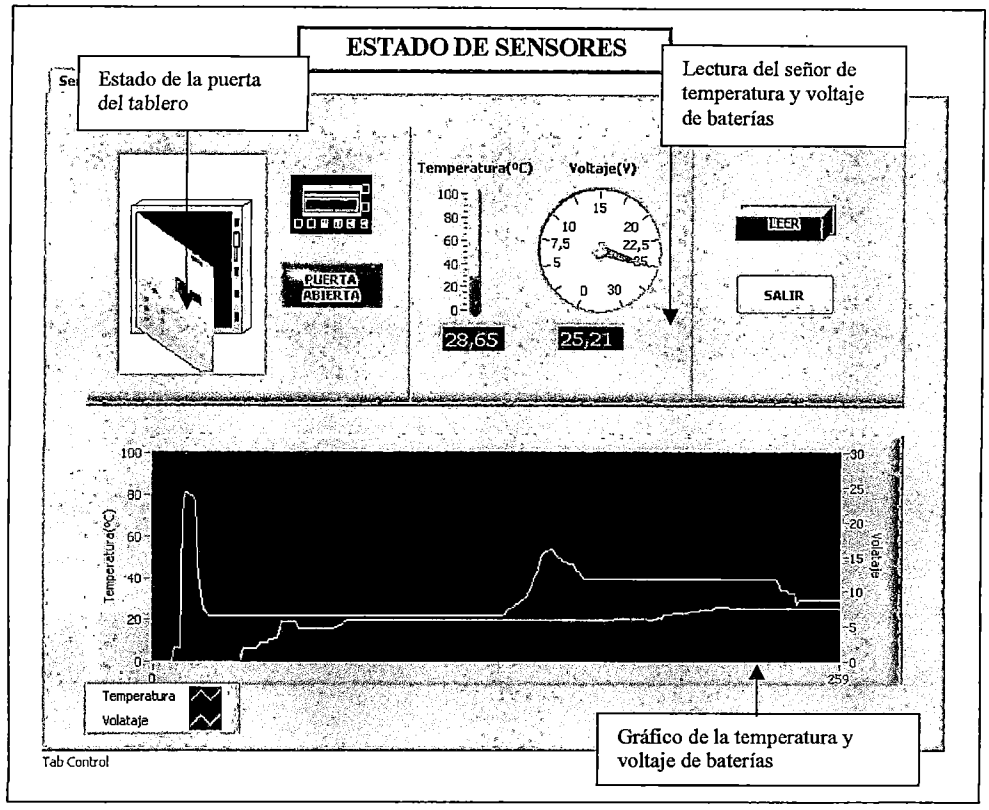



***Figura 4.32.-*** *Panel frontal para la lectura de los sensores de: temperatura, estado de la puerta y voltaje de baterías*

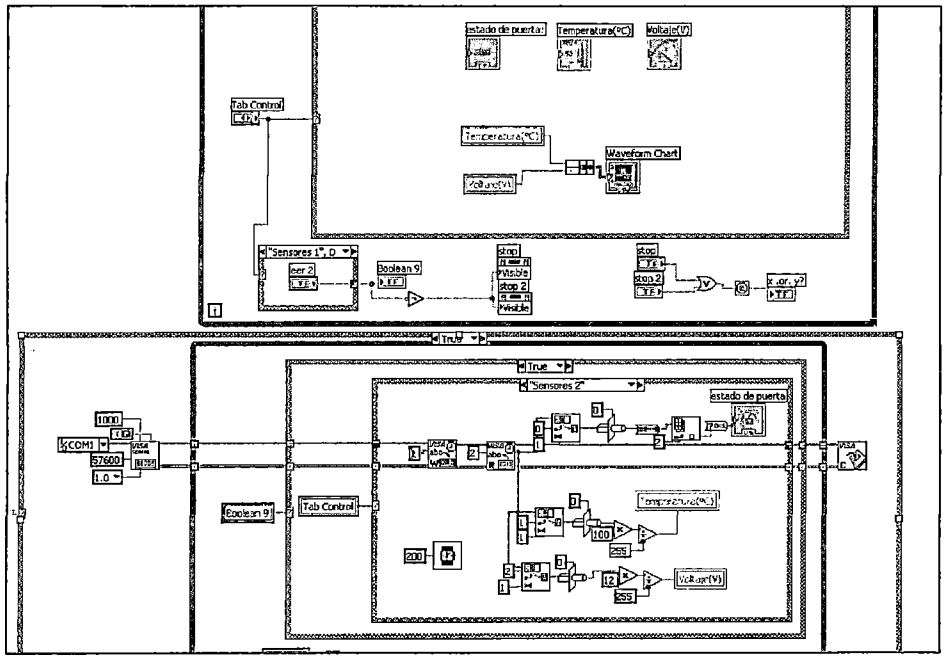



***Figura 4.33.-*** *Diagrama de bloques del programa para la lectura de los sensores de: temperatura, estado de la puerta y voltaje de baterías*

### 4.8.3. Calibración del sensor de temperatura

Para realizar la calibración del sensor de temperatura, se dispuso de un recipiente con agua al que se fue calentándolo hasta llegar a la temperatura de ebullición. Mientras se hacía esto, se iba midiendo la temperatura con un termómetro patrón y con el transistor 2N2222 que se usó como sensor de temperatura para el proyecto, al cual se le diseñó su tarjeta (ver figura 4.34). Estas dos mediciones se tabularon en Excel para luego encontrar la curva característica que relaciona la tensión del sensor y la temperatura en grados Celcius.

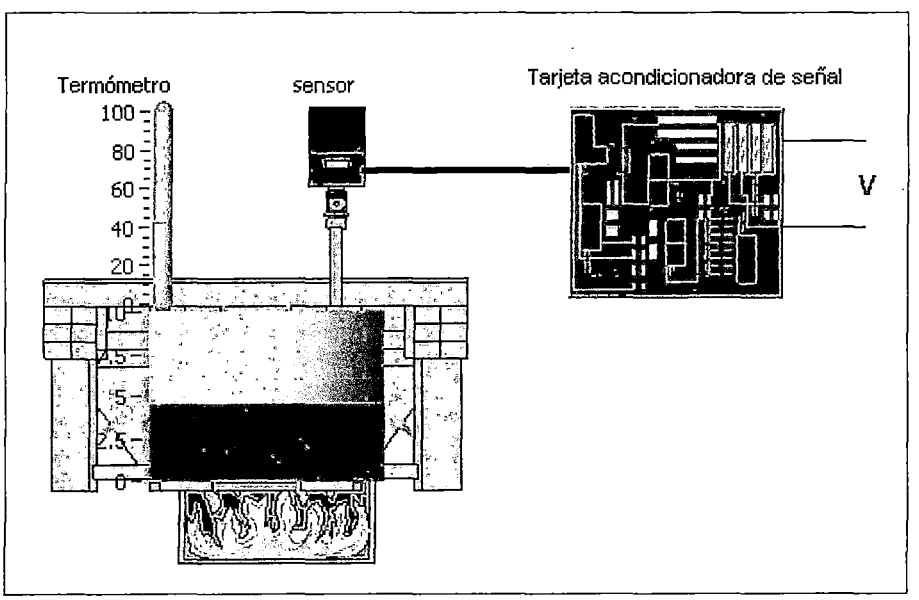



***Figura 4.34.-*** *Esquema de las pruebas para la calibración de sensor de temperatura*

El instrumento patrón usado fueron dos ***termómetro de mercurio*** y una ***termocupla*** cuya lectura se podía visualizar en un multímetro. Podemos observar en la tabla 4.1 los datos tomados.

***Tabla 4.1***
*(Tabulación de los resultados de la prueba del sensor de temperatura)*

| *Temperatura (°C)* | *Voltaje* |
|---|---|
| 25 | 1.96 |
| 26 | 2 |
| 27 | 2.04 |
| 29 | 2.12 |
| 30 | 2.16 |
| 32 | 2.24 |
| 34 | 2.32 |
| 35 | 2.36 |
| 37 | 2.44 |
| 40 | 2.56 |
| 41 | 2.6 |
| 42 | 2.64 |
| 45 | 2.76 |
| 46 | 2.8 |
| 47 | 2.84 |
| 48 | 2.88 |
| 50 | 2.92 |
| 51 | 2.96 |
| 53 | 3.04 |
| 56 | 3.16 |
| 58 | 3.242 |
| 59 | 3.283 |
| 60 | 3.323 |
| 61 | 3.364 |
| 62 | 3.405 |
| 63 | 3.446 |
| 65 | 3.528 |
| 68 | 3.651 |
| 69 | 3.692 |
| 70 | 3.733 |
| 74 | 3.897 |
| 76 | 3.979 |
| 78 | 4.061 |
| 81 | 4.184 |
| 85 | 4.348 |
| 87 | 4.43 |
| 91 | 4.594 |
| 94 | 4.717 |
| 97 | 4.84 |
| 100 | 4.96 |

Estos datos fueron pasados a Excel para un ajuste de curva el cual presenta en el gráfico de la figura 4.35.

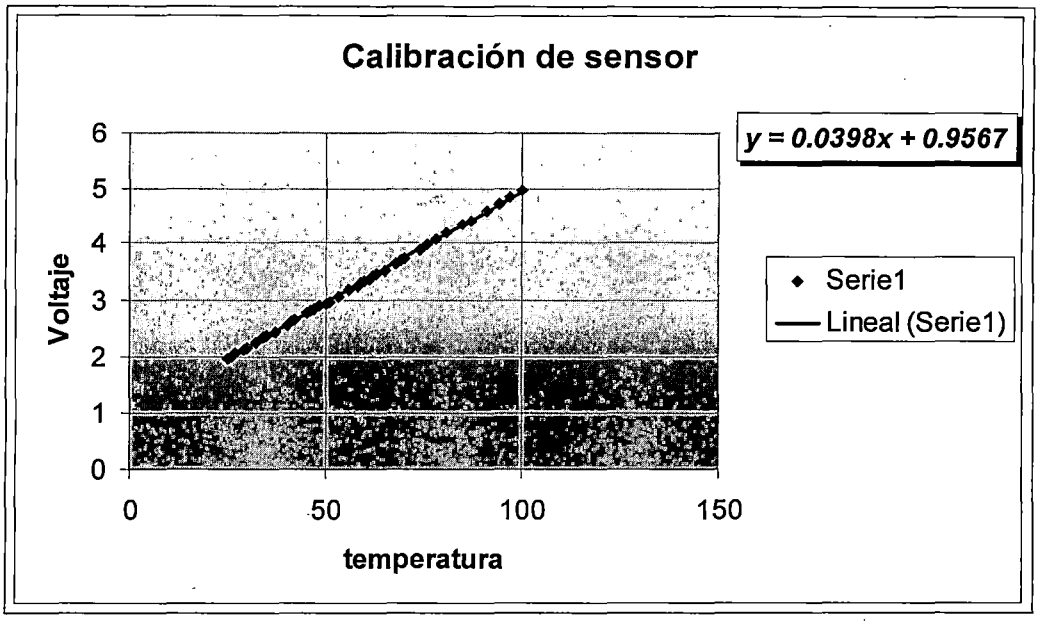



***Figura 4.35.-*** *Ajuste de curva de la relación voltaje-temperatura*

Luego estos datos, de acuerdo a la digitalización del conversor análogo digital del microcontrolador, deben pasar por una relación de conversión, el cual está determinado por los voltajes de referencias que se conectó al microcontrolador y la resolución de bits que se escogió, como se escogió trabajar con una resolución de 8bits, entonces tendremos:

$$\Delta V = \frac{[(Vref+)-(Vref-)]}{2^n - 1} = \frac{(5-0)}{2^8 - 1} = 19.6mV$$

Luego el número enviado por el microcontrolador que está entre 0 y 255 se convertirá a voltaje para reemplazarlo en la ecuación antes deducida:

$$\Delta V = \frac{[(Vref+)-(Vref-)]}{2^n - 1} = \frac{(5-0)}{2^8 - 1} = 19.6mV$$

V=0.0398T+0.9567

T=25.12563V-24.037688

T=25.12563*0.0196N-24.037688

$$\boxed{T = 0.49266N - 24.037688}$$

Donde:

T: temperatura en °C

N: número entero digitalizado por el conversor análogo digital del PIC

Esta fórmula es pasada a LABVIEW (ver figura 4.36) para que al recibir los datos del microcontrolador se pueda convertir de código decimal a unidades de temperatura en grados Celsius.

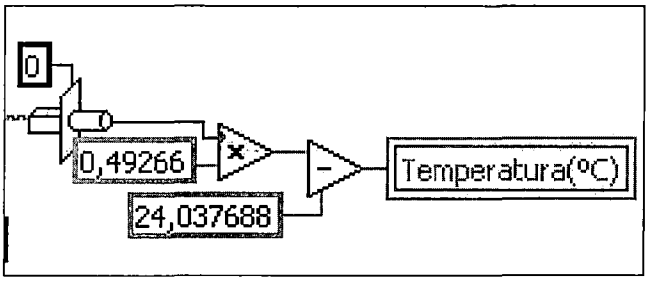



*Figura 4.36.- Ecuación de ajuste en LABVIEW*

## 4.9. GENERACIÓN DE REPORTES

Finalmente esta es la parte más importante desde el punto de vista de la supervisión, ya que para administración aquí se puede verificar la buena o mala marcha en la operación de un peaje, sobre todo estando alejado de la capital. Para esto en esta etapa se puede apreciar dos tipos de reportes:

- Reporte detallado
- Reporte clasificado

### 4.9.1. Reporte detallado

Aquí se puede apreciar vehículo por vehículo el tipo la secuencia de número, la hora, el minuto y el segundo de la hora que se desea y que se ingresa desde la pantalla, así como de todas las horas de un turno completo.

Los turnos para estos peajes usualmente son dos turnos de 12 horas cada uno. Esta operación se puede hacer en la página 1 del "tab control". También se puede apreciar en la parte superior el número de veces que se reseteó el equipo en esa hora respectiva.

La búsqueda de información se realiza por fecha y hora o turno. En el caso en que por alguna razón existan dos horas que se guardaron en diferentes partes de las memorias, y el sistema sólo visualice una parte se puede aumentar el "número de insistencias" para buscar y seguir buscando. Esto se explicará más adelante. En gráfico de la figura 4.37 se ven todas las partes que tiene.

Un ejemplo del reporte detallado, presentado en formato "html", se puede apreciar en la figura 4.38.

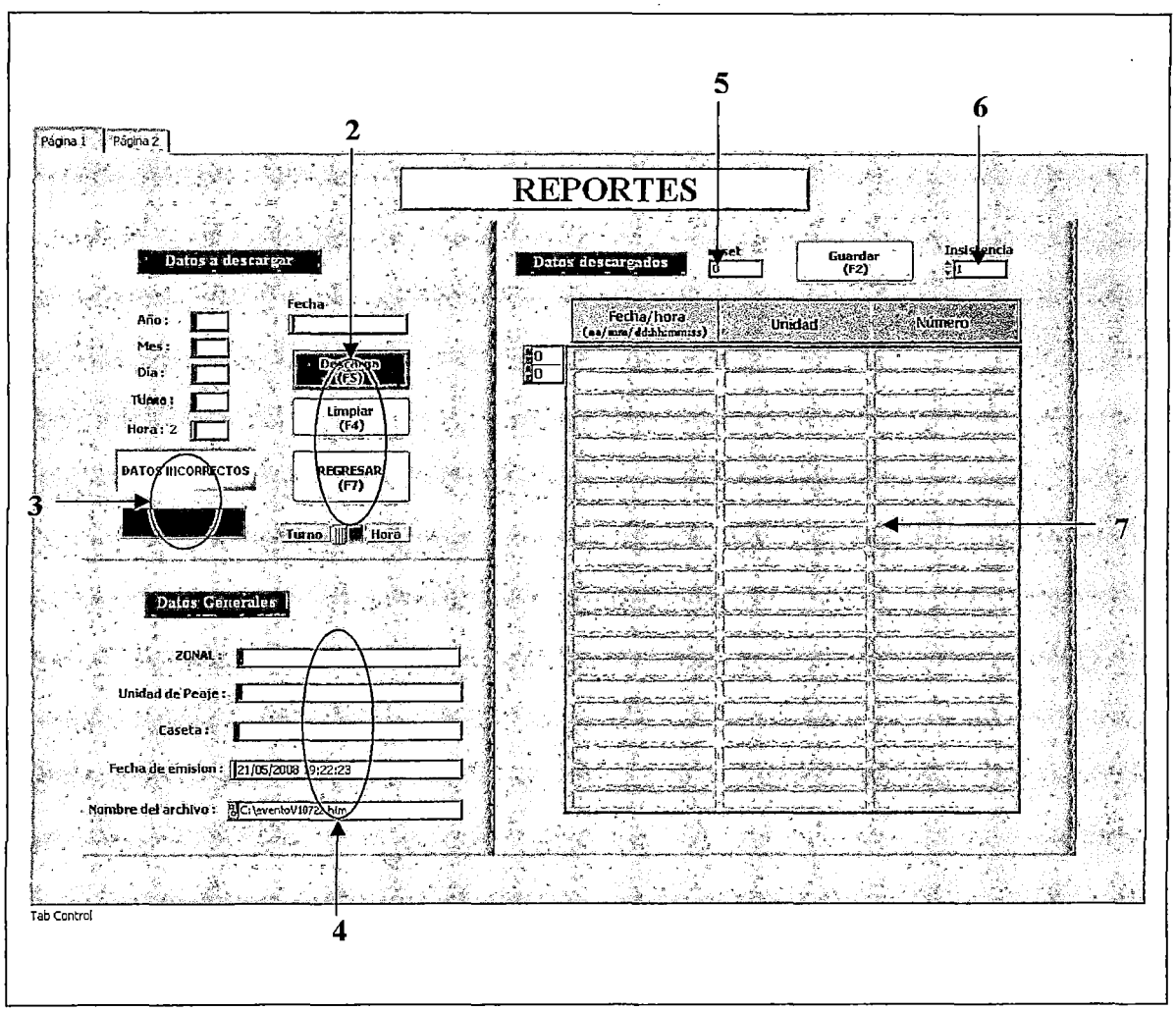



***Figura 4.37.-*** *Panel Frontal para la generación de reportes por eventos, no clasificado*

1. Aquí se pone la fecha y la hora o el turno que se quiere descargar

2. Tenemos cuatro controles por botones:

   ***Descargar***. Solo se podrá si los datos de la fecha se introdujeron correctamente, esto es en 2 dígitos.

***Limpiar***. Los datos descargados serán visualizados en una tabla, y si la tabla es muy extensa y se desea limpiarla se puede hacer con este botón.

***Regresar***. Sirve para volver al menú

***Turno hora***. Es un seleccionador. Si se pone para hora se podrá descargar la información de 1 hora en particular y la casilla de datos de turno desaparecerá. Si se selecciona para turno entonces el sistema no permitirá poner la hora sino seleccionar el turno.

3. Indicadores, los cuales dirán si los datos solicitados están en el formato especificado de 2 dígitos, y el indicador verde indicará si se encontraron dato o no existen dichos datos.
4. Son datos que vienen del menú principal, pero dos de ellos los coloca el sistema los cuales son la fecha de extracción de la información y el nombre del archivo en función a esta fecha.
5. Número de veces que se reinicializó la tarjeta.
6. Número de insistencias, en caso que haiga un dato aislado.
7. Tabla (arreglo bidimensional) en el que se puede visualizar la fecha, hora, minuto y segundo en el que pasó un vehículo, así como el tipo de vehículo que es y el número correlativo a ello.

PROVIAS NACIONAL

JL Controles y Sistemas Electrónicos SAC
(Teléfonos: 015537689-0199920822)

ZONAL : N° X Junín-Pasco
UNIDAD DE PEAJE : Chalhuapuquio
CASETA :
Fecha de emisión: 23/08/2006
Hora de emisión: 03:17:29 p.m.
Turno: 1

| Fecha/hora (aa/mm/dd:hh:mm:ss) | Unidad | Número |
|---|---|---|
| 06/08/04:12:00:52 | L2 | 0042 |
| 06/08/04:12:01:13 | L2 | 0043 |
| 06/08/04:12:02:25 | P2 | 0021 |
| 06/08/04:12:03:19 | P3 | 0003 |
| 06/08/04:12:06:13 | P2 | 0022 |
| 06/08/04:12:07:38 | P5 | 0002 |
| 06/08/04:12:07:58 | P2 | 0023 |
| 06/08/04:12:08:17 | L2 | 0044 |
| 06/08/04:12:11:15 | P2 | 0024 |
| 06/08/04:12:11:32 | L2 | 0045 |
| 06/08/04:12:11:46 | L2 | 0046 |
| 06/08/04:12:12:25 | P2 | 0025 |
| 06/08/04:12:12:40 | L2 | 0047 |

***Figura 4.38.-*** *Archivo generado por eventos del peaje de Chalhuapuquio*

### 4.9.2. Reporte clasificado

Descargada la información se puede apreciar en la página dos, a todos los vehículos clasificados y ordenados en horas ascendentes, de acuerdo al turno o las horas específicas que se pidieron. En la figura 4.39 se puede apreciar el panel frontal del informe clasificado, el cual se genera después de la descarga de datos de la tarjeta de control. Un ejemplo del reporte clasificado, presentado en formato "html", se puede apreciar en la figura 4.40.

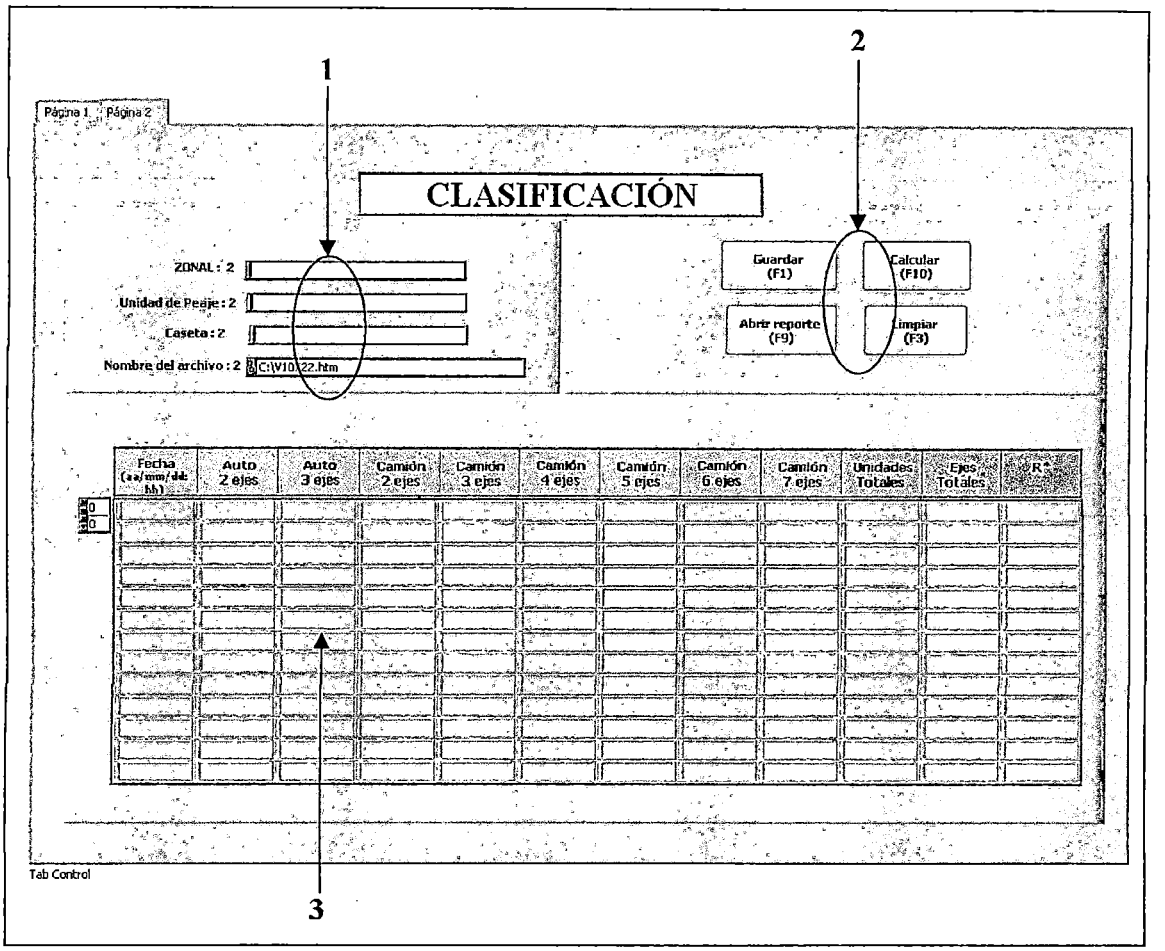



***Figura 4.39.-*** *Panel Frontal para la generación de reportes clasificado y por turno o por hora*

1. Al igual que en la página anterior éstos son los datos cargados en el menú principal, además del nombre del archivo que se genera para el reporte clasificado. Es importante mencionar que el reporte generado aquí es diferente que el reporte generado en la página 1, por lo tanto el archivo será diferente y con otro nombre, en función de la fecha y hora de la extracción, así como del tipo de archivo.

2. Los botones que se disponen cumplen las siguientes funciones:

   ***Guardar:*** guarda el reporte

***Abrir reporte:*** se puede hacer una vista previa al reporte generado y se puede imprimir desde aquí

***Calcular:*** totaliza la cantidad de vehículo por tipo en el turno o las horas extraídas así como el total de ejes y el total de unidades.

***Limpiar:*** si se requiere sacar más información, se podrá limpiar pantalla y volver a hacerlo para ser clasificada.

3. Es una tabla (arreglo bidimensional) en el cual se muestra toda la información clasificada, así como los totales calculados.

**PROVIAS NACIONAL**

**JL Controles y Sistemas Electrónicos SAC**
**(Teléfonos: 015537689-0199920822)**

**ZONAL :** Junin-Pasco
**UNIDAD DE PEAJE :** Chalhuapuquio
**CASETA :** 1
**Fecha de emisión :** 01/09/2007
**Hora de emisión :** 08:08:18 a.m.
**Turno :** 2

| Fecha (dd/mm/aa:hh) | Auto de 2 ejes | Camión de 2 ejes | Camión de 3 ejes | Camión de 4 ejes | Camión de 5 ejes | Camión de 6 ejes | Camión de 7 ejes | Autos de 3 ejes | Unidades totales | Ejes totales | R* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 31/08/07:20 | 0019 | 0004 | 0002 | 0001 | 001 | 001 | 000 | 00 | 28 | 67 | 0 |
| 31/08/07:21 | 0007 | 0006 | 0000 | 0001 | 001 | 001 | 000 | 00 | 16 | 41 | 0 |
| 31/08/07:22 | 0008 | 0010 | 0001 | 0000 | 001 | 001 | 000 | 00 | 21 | 50 | 0 |
| 31/08/07:23 | 0005 | 0007 | 0004 | 0000 | 000 | 000 | 000 | 00 | 16 | 36 | 0 |
| 01/09/07:00 | 0007 | 0008 | 0002 | 0000 | 000 | 000 | 000 | 00 | 17 | 36 | 0 |
| 01/09/07:01 | 0003 | 0008 | 0003 | 0001 | 001 | 000 | 000 | 00 | 16 | 40 | 0 |
| 01/09/07:02 | 0002 | 0011 | 0000 | 0001 | 000 | 000 | 000 | 00 | 14 | 30 | 0 |
| 01/09/07:03 | 0001 | 0009 | 0006 | 0001 | 000 | 000 | 000 | 00 | 17 | 42 | 0 |
| 01/09/07:04 | 0000 | 0006 | 0001 | 0001 | 001 | 000 | 000 | 00 | 9 | 24 | 0 |
| 01/09/07:05 | 0000 | 0008 | 0010 | 0000 | 002 | 001 | 000 | 00 | 21 | 62 | 0 |
| 01/09/07:06 | 0007 | 0004 | 0000 | 0001 | 002 | 002 | 000 | 00 | 16 | 48 | 0 |
| 01/09/07:07 | 0009 | 0006 | 0002 | 0000 | 002 | 000 | 000 | 00 | 19 | 46 | 0 |
| Suma: | 68 | 87 | 31 | 7 | 11 | 6 | 0 | 0 | | | |

***Figura 4.40.-*** *Archivo generado clasificado del peaje de Chalhuapuquio*

### 4.9.3. Código en LABVIEW

Para poder explicar la forma como se desarrollo el código en LABVIEW, lo dividiremos de la siguiente manera:

- Búsqueda de información y decodificación
- Ordenamiento de datos
- Generación del reporte en formato "htlm" y guardado de información.

### A. Búsqueda de información y decodificación

Esta operación se realiza básicamente con el subVI "oper_orden3". El cual se puede observar en las figuras 4.41, 4.42 y 4.43.

En primer lugar se tiene que codificar la fecha y hora de la misma forma como lo codificó el microcontrolador en la tarjeta de control. Esto se puede observar en la figura 4.41.

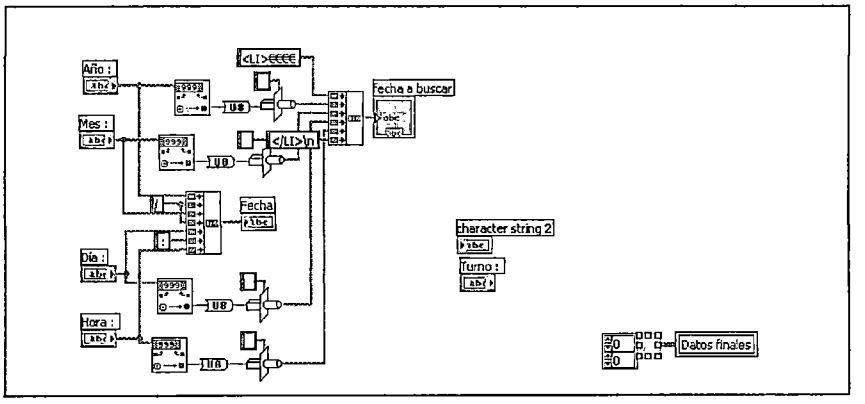



***Figura 4.41.*** *- Diagrama de bloques de la codificación de datos de la fecha para la búsqueda de la información*

Luego debemos recordar que cuando se descargo la información en la PC, esta se guardó en un archivo en la memoria "C" del PC, ahora se tiene que abrir este archivo. Esto lo podemos observar en la figura 4.42.

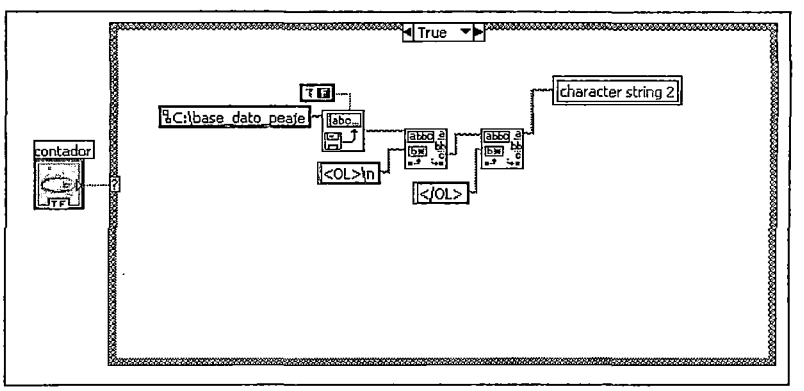



***Figura 4.42.-*** *Diagrama de bloques para abrir el archivo descargado del microcontrolador, para el inicio de la búsqueda*

Después se buscan los cuatro marcadores, grabados por el microcontrolador, que indican un cambio de hora, para comparar la fecha que deseamos con la que leemos inmediatamente después de estos marcadores.

Una vez encontrada la fecha y hora buscada, extraemos los datos del archivo que están inmediatamente después de esta hora ya que estos corresponden a los vehículos hasta que nuevamente encontremos un marcador. El panel de diagrama de bloques que corresponde a esta búsqueda se muestra en la figura 4.43.

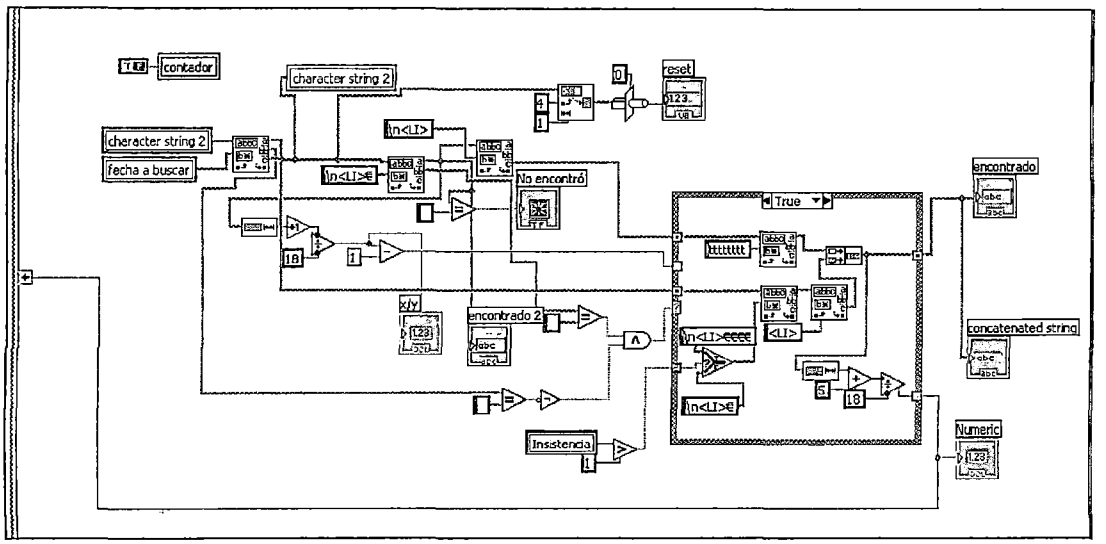



***Figura 4.43.-*** *Diagrama de bloques para la decodificación y búsqueda de los datos de las fechas indicadas*

Recordemos, como se puede apreciar en el gráfico 4.43, que la búsqueda termina hasta que encontremos otro marcador o el final de información que son los o caracteres 't'.

Finalmente los datos son ordenados para ser usados en el VI que lo contiene. También es importante destacar que en cada uno de los *CASE* no se muestran el cao falso porque en la mayoría de ellos no se realiza operación sino sólo de reinicialización de algunas variables.

En la figura 4.44 se observa una parte del código que sirve para la búsqueda de los vehículos en 1 hora, por lo que si se desea que se realice en un turno simplemente se realiza un lazo "FOR" para la cantidad de horas que tiene un turno.

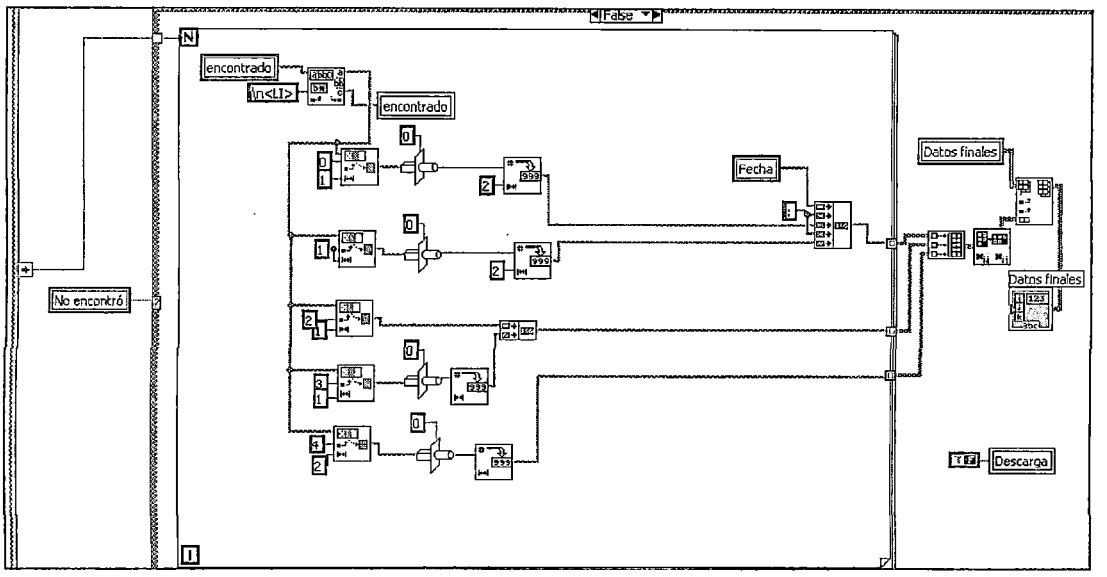



*Figura 4.44.- Diagrama de bloques de la decodificación de datos encontrados*

## B. Ordenamiento de datos

Los datos que se descargan por hora se muestran en un arreglo bidimensional de cadenas. Al subVI que se encarga de buscar se accede tantas veces como se pida y estos son añadidos en dicho arreglo bidimensional. (Ver figura 4.45)

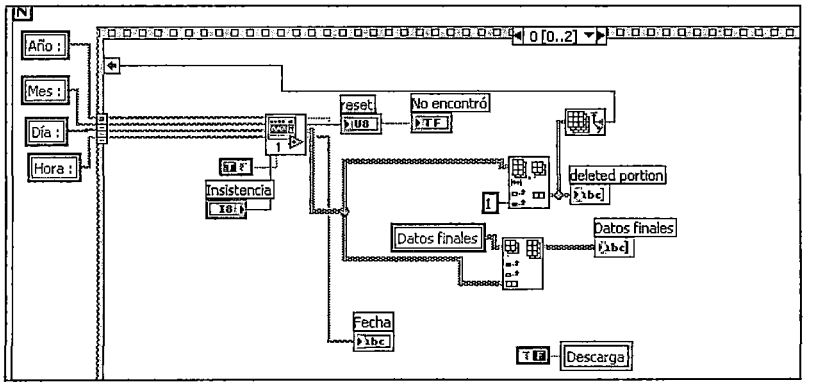



*Figura 4.45.- VI para la búsqueda de la información codificada*

Luego la información es clasificada, preguntando por cada elemento qué tipo de vehículo es y creando un contador para cada uno de ellos, teniendo así el total de vehículos de acuerdo al tipo y por hora. (Ver figura 4.46)

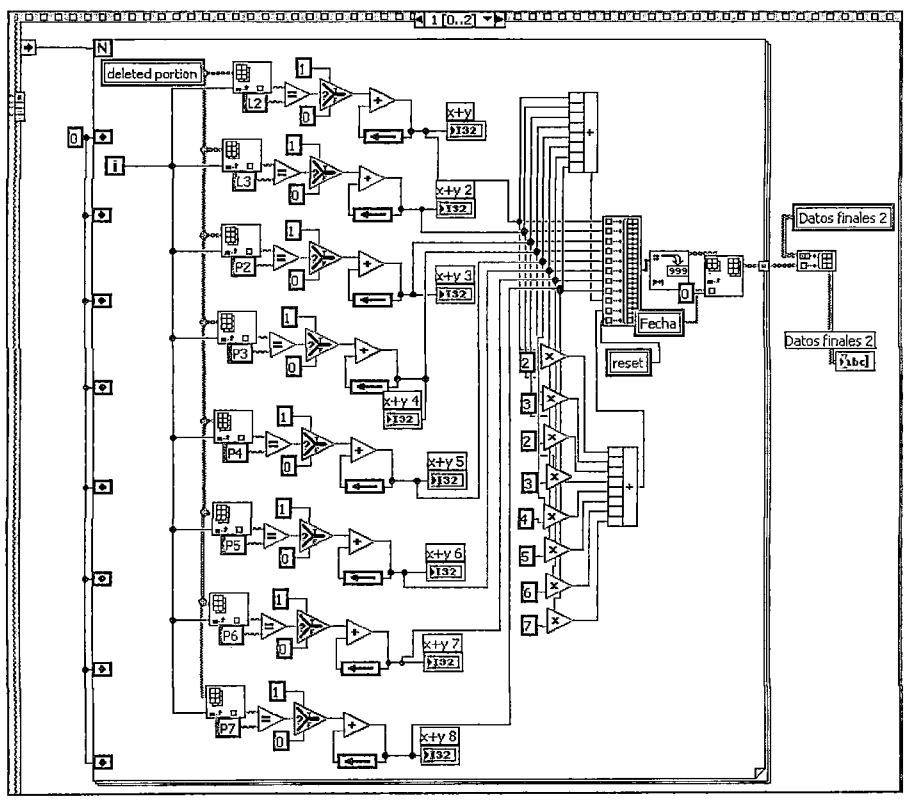



***Figura 4.46.*** *- Diagrama de bloques para el cálculo del total de unidades y ejes en cada hora y cada turno*

Después, si es que se pidió la descarga un turno, desde el programa tenemos que generar la siguiente hora, teniendo en cuenta que esta operación podría cambiar de hora o de mes o incluso de año al finalizar la última hora del día, para esto se hizo un subVI que genera la siguiente hora. Éste subVI se llama "aumento1h"

## C. Generación del reporte y guardado de información

Una vez buscado y extraído la información de la PC se procederá a generar el informe en formato "html". Debemos recordar que estos informes pueden ser 2, uno el detallado y otro el clasificado. LABVIEW tiene subVI's para generar este tipo de reportes, es decir crear listas, tablas, tipos de letras, cabeceras, etc. Finalmente se guardará el archivo y por versiones si es que se graba más de una vez. Este código se puede apreciar en la figura 4.47

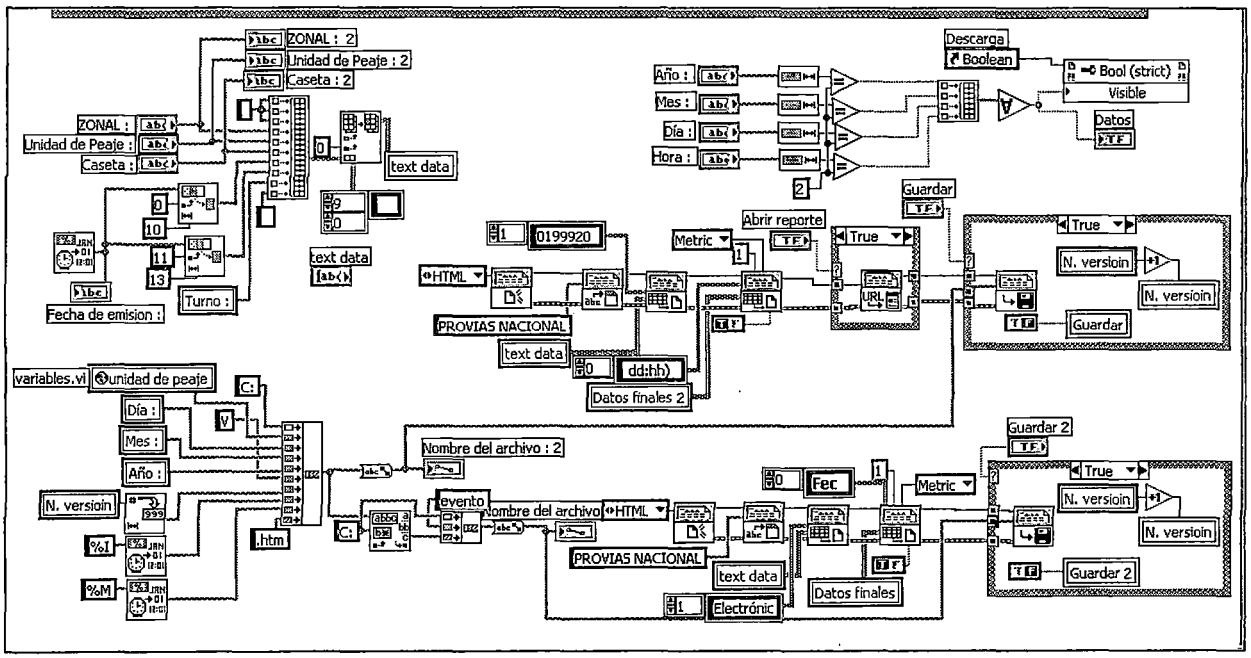



***Figura 4.47.-*** *Diagrama de bloques para la generación y presentación de reportes por eventos y clasificado según corresponda, por hora o por turno*

# CAPÍTULO V

# INSTALACIÓN Y PUESTA EN MARCHA DEL SISTEMA

PROVIAS NACIONAL solicitó que se instalara el sistema de control desarrollado en los peajes de Chalhuapuquio y Huacrapuquio para ponerlos a prueba; luego, si el sistema trabajaba bien, se procedería la instalación en otros peajes, y así fue. Con el fin de explicar el proceso de instalación y puesta en marcha, detallaremos lo que se hizo en estos peajes.

## 5.1. OBRA CIVIL

Primeramente se hizo una losa de concreto y una caja de paso, para lo cual las especificaciones fueron las siguientes:

- Las cajas de paso deben ser de concreto armado con tapa reforzada de 60 x 60cm o de 40 x 40cm con perfil de metal en la base de descanso de la caja para mayor seguridad de deterioro, no debe construirse en la Unidad de Peaje, debe de ser prefabricado, a fin de evitar malos acabados y demoras de tiempo.

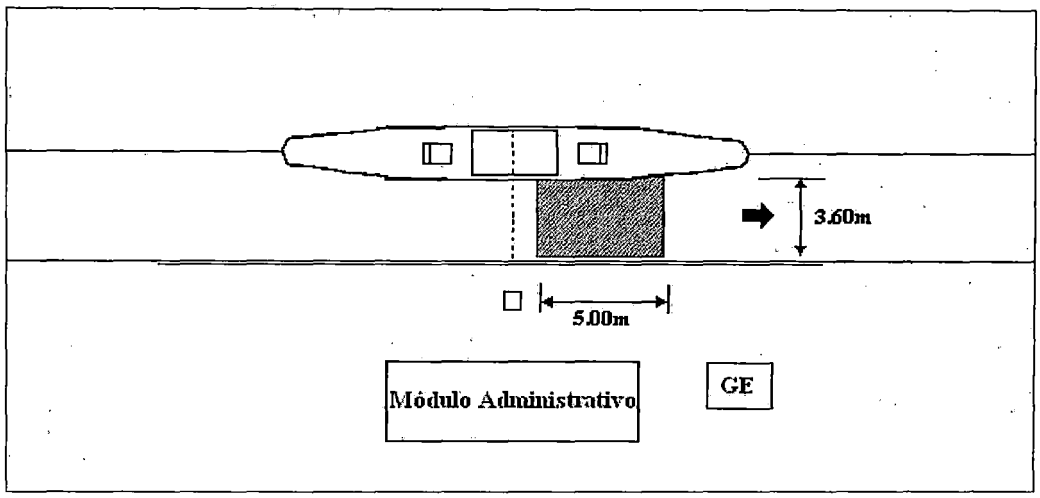
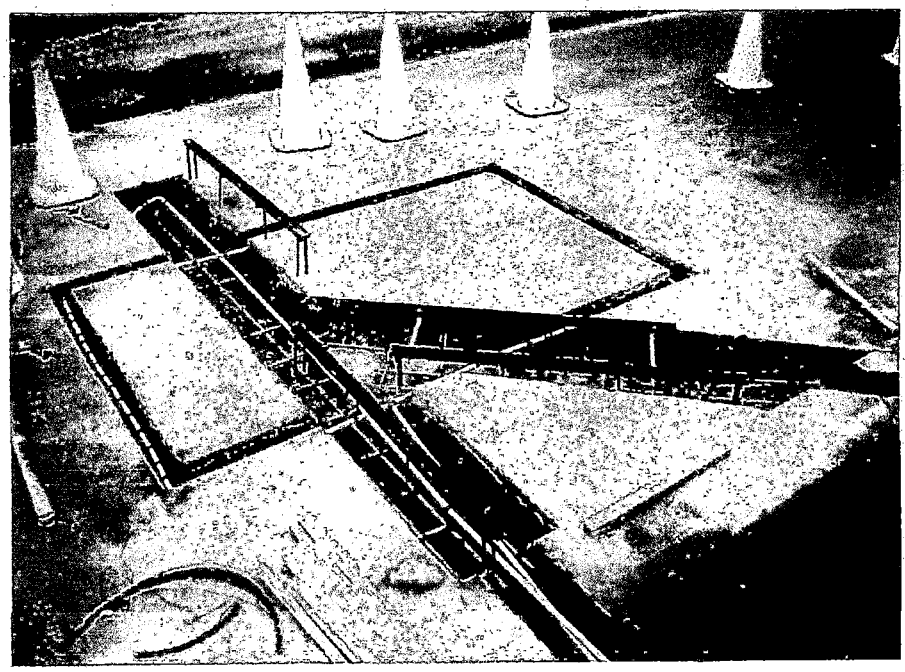



***Figura 5.1.-*** *Vía de paso de vehículos*

- La zanja para el pase de tubería de conduit, o ductos de concreto, deberá de estar a una profundidad de 0.2 x 0.30m, como mínimo.
- La resistencia del concreto debe para la loza de concreto debe ser de 280kg/cm$^2$ y del acero de 4200kg/cm$^2$. Esta loza se puede apreciar en la figura 5.1, dispuesta en la pista, con respecto a la posición de la caseta de cobro. En la figura 5.2 se puede apreciar la preparación de la loza para colocar los sensores de presión y antena. La loza terminada se puede apreciar en la figura 5.3, con los sensores ya instalados y preparados para la conexión eléctrica.
- Los acabados finalmente deben ser similares al acabado de la caseta de cobranza y oficinas, por razones de estética, esto es algo que solicita PROVIAS desde un inicio.



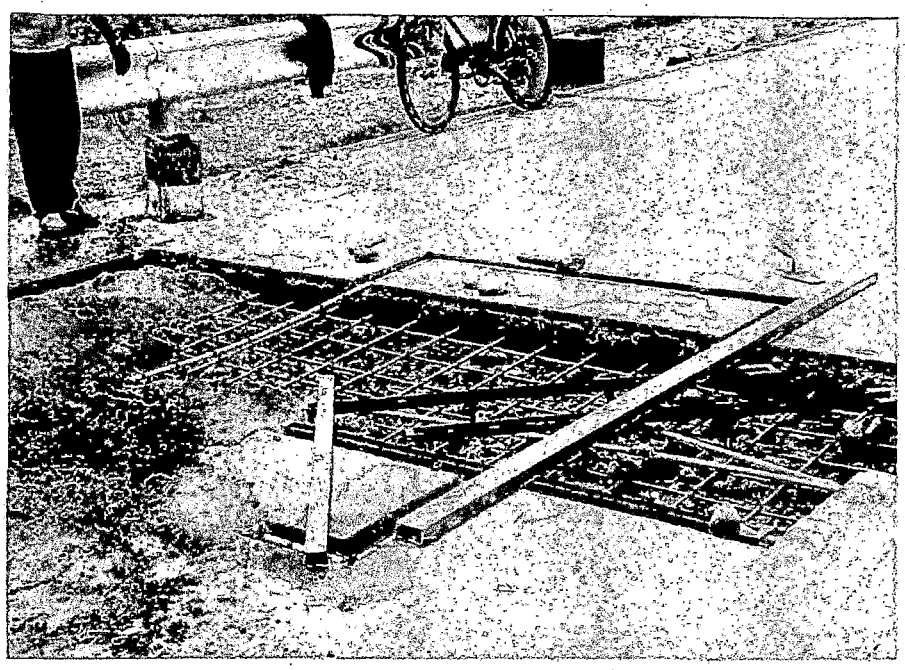


***Figura 5.2.-*** *Instalación de sensores de presión y de antena*

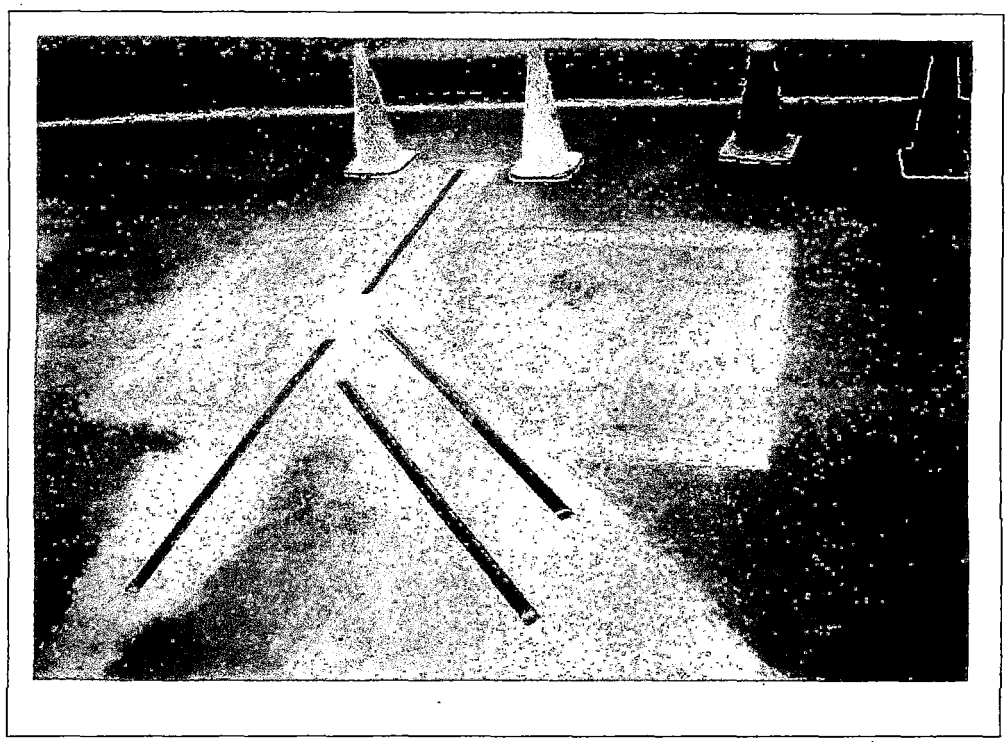


*Figura 5.3.- Obra civil e instalación de sensores terminada*

## 5.2. INSTALACIÓN ELÉCTRICA

- Los tableros deberán ubicarse en un lugar apropiado de manera que sea funcional para cualquier tipo de maniobra y/o mantenimiento que se requiera se realice, como se observa en la figura 5.4, en donde se observa que este se encuentra junto al computador desde donde se hace toda labor de control y supervisión.
- Asimismo se debe tener en cuenta, por razones de mantenimiento, de que los sensores de presión deben ser fáciles de retirar y reemplazar cuando esto así se requiera como se puede apreciar en la figura 5.4.

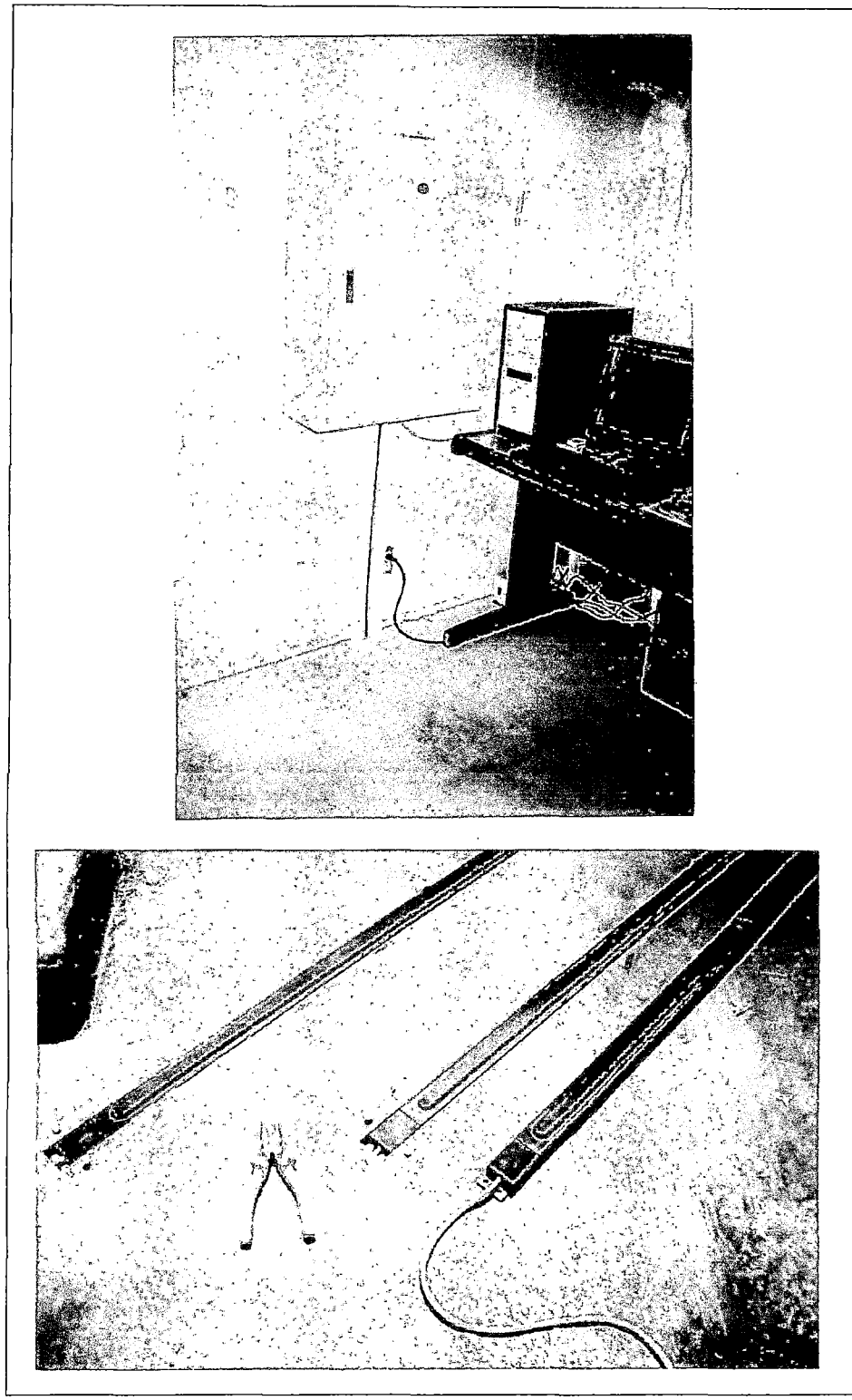


***Figura 5.4.-*** *Tablero eléctrico de control y sensores de presión del peaje de Chalhuapuquio*

- Todo tipo de cableado de será a través de tuberías, de acuerdo a las normas eléctricas, evitando que éstas vayan por la intemperie.
- El equipo e instalaciones eléctricas deberá tener su instalación de puesta a tierra. (Ver figura 5.5)

***Figura 5.5.-*** *Fosa para el pozo de tierra del equipo*

- El pozo de tierra debe llegar como las especificaciones que se hicieron al inicio a un valor menor de 5 Ohmios. (Ver figura 5.6)
- El pozo de tierra fue instalado por medio de la empresa ***J.L. Controles y Sistemas Electrónicos S.A.C.*** y es por esta razón que no se detalla su instalación y diseño.

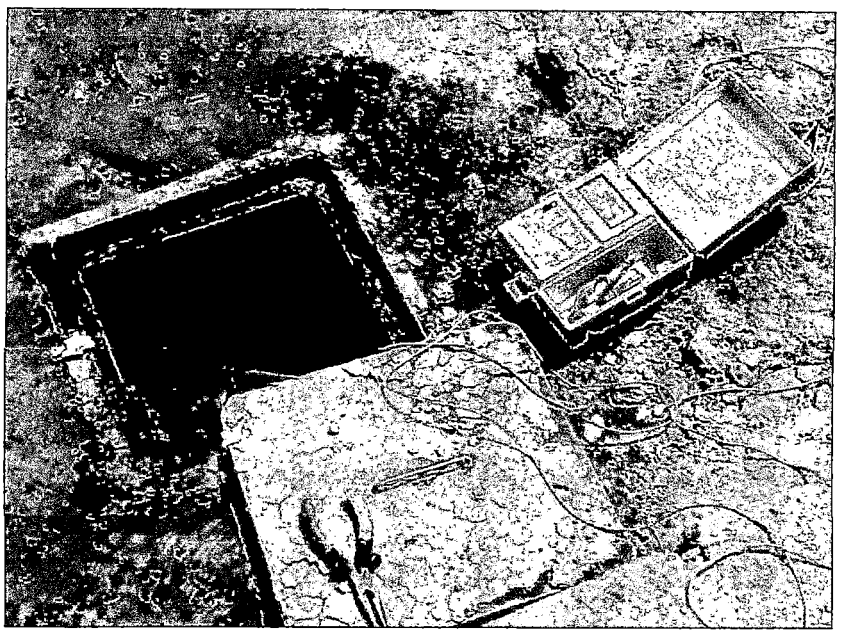


***Figura 5.6.-*** *Medición del pozo de tierra*

- La caja metálica del tablero eléctrico se llevará previamente preparada, para la conexión de los sensores respectivos en su momento y su verificación.
- Las tarjetas deben ser instaladas, calibradas y debidamente probadas (ver figura 5.7), siguiendo la orden de instalación programada:
  - ✓ Verificación de la carga de las baterías.
  - ✓ Verificación del cargador de baterías.
  - ✓ Mediciones de voltaje a cada tarjeta
  - ✓ Calibración y verificación del voltaje de la tarjeta del sensor de antena por medio de la inductancia variable.
  - ✓ Encendido de la tarjeta principal y configuración de ella por medio del software.

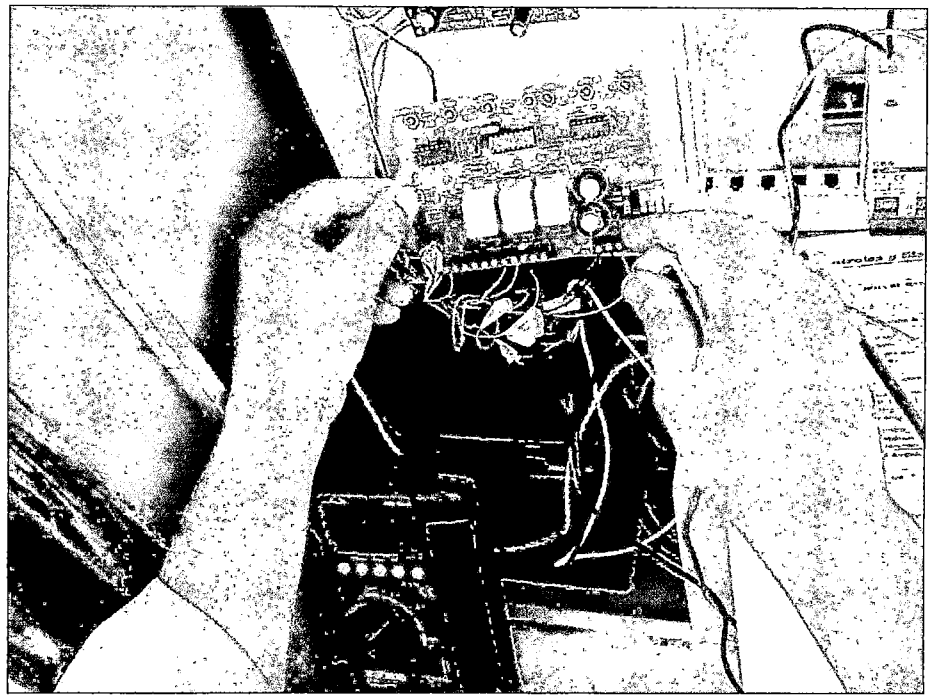


***Figura 5.7.-*** *Instalación de las tarjetas de control*

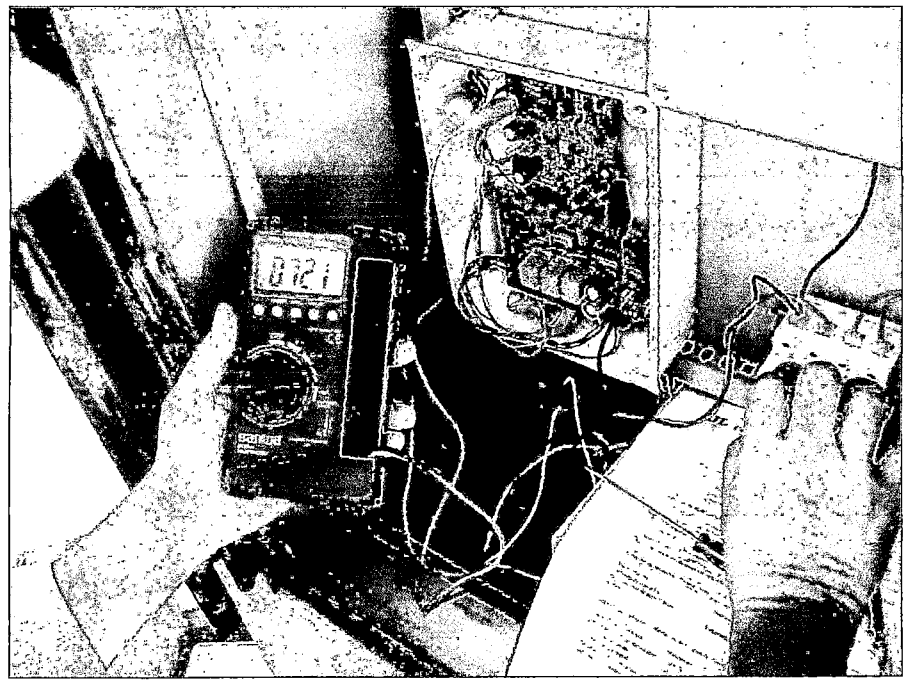


***Figura 5.8.-*** *Medición de la tensión del equipo*

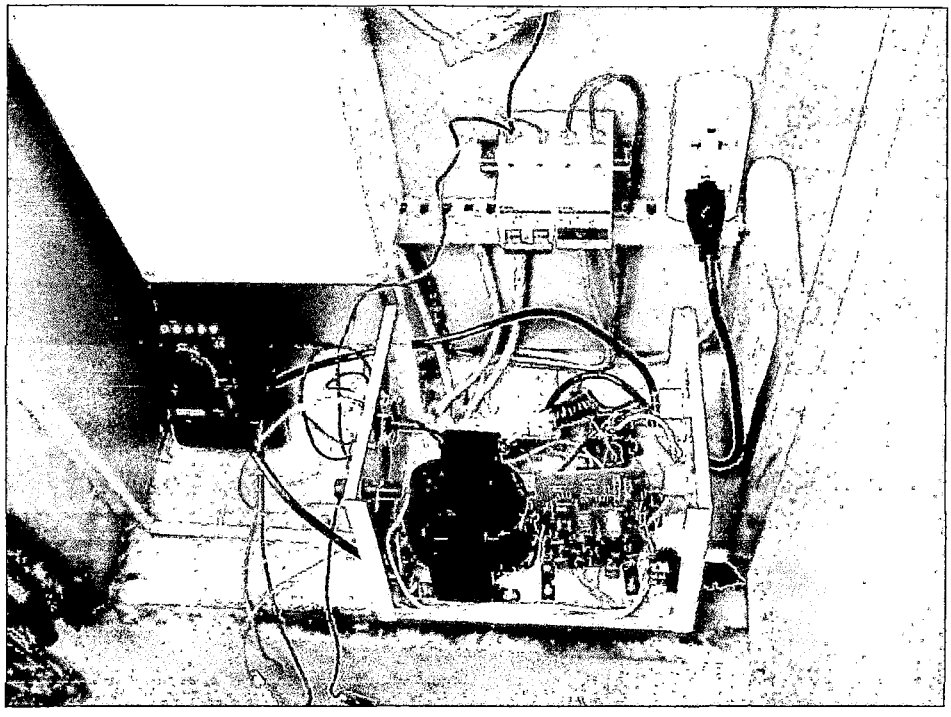


***Figura 5.9.-*** *Verificación de estado del cargador de baterías*

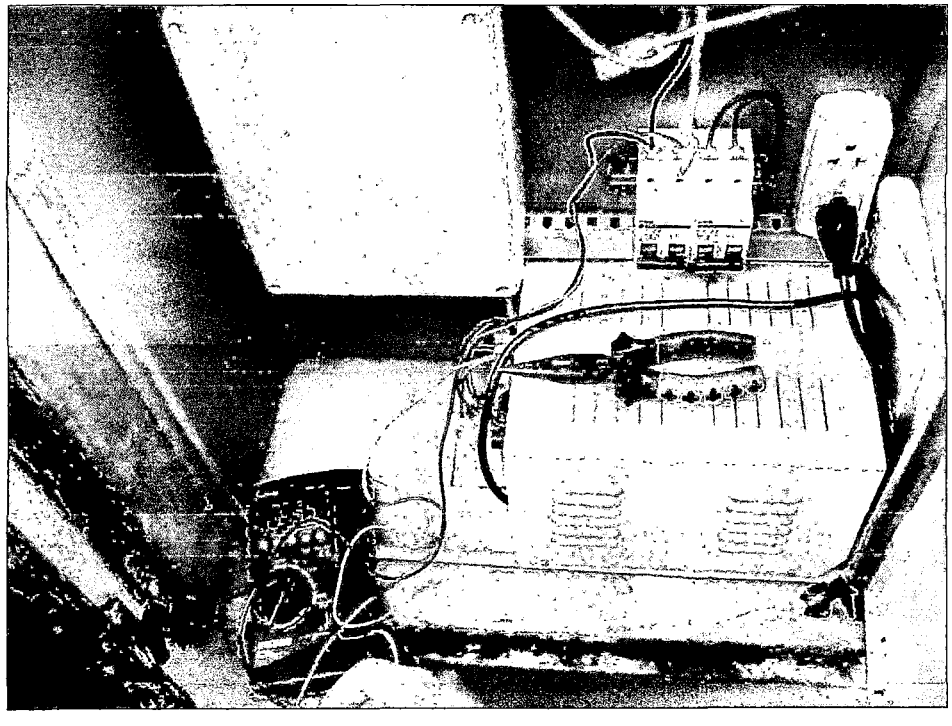


***Figura 5.10.-*** *Verificación de tensiones con el equipo en funcionamiento*

- Prueba del correcto sensado de vehículos durante un periodo de 1 a 2 horas

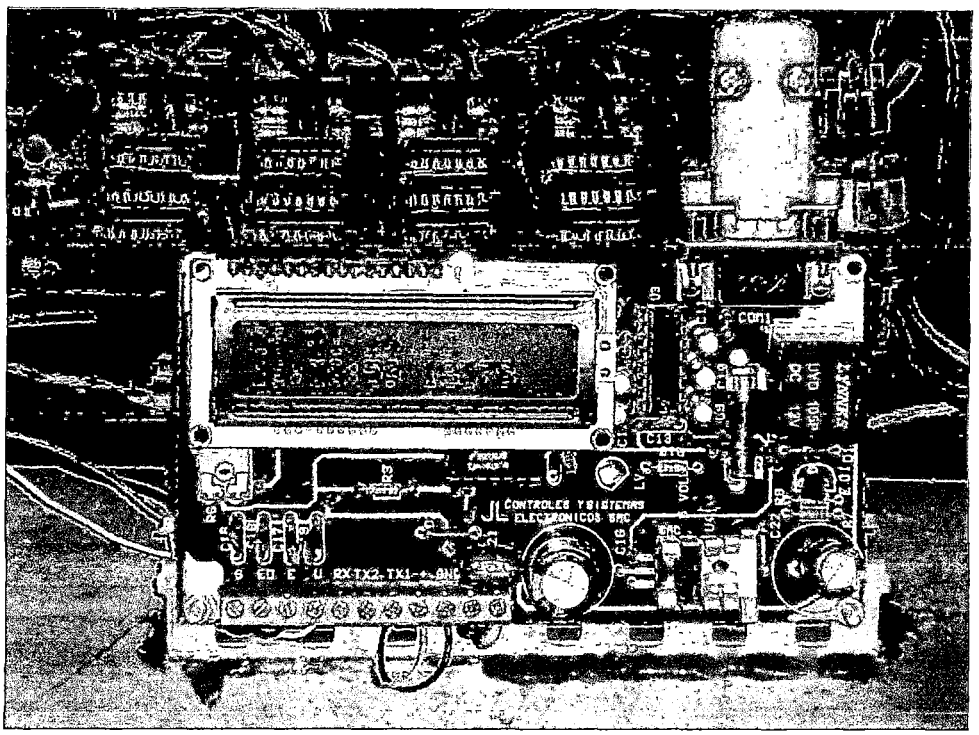


***Figura 5.11.-*** *Verificación del buen funcionamiento del equipo, antes de concluir con la instalación*

## 5.3. INSTALACIÓN DEL SOFTWARE

El software a instalar consiste sólo en un CD instalador en el que se encuentra todo lo necesario para su total y completo funcionamiento. La instalación se procederá ejecutando el archivo setup.exe que se encuentra dentro de la carpeta "installer".

Setup.exe

Luego de ello simplemente aceptar y pulsar siguiente (next), esperar mientras se termina la instalación.

Reiniciar la PC y luego buscar el acceso directo al programa "traffic counter" que se encuentra en inicio/todos los programas/traffic counter. Iniciar el programa de control, encendiendo la tarjeta y configurando la tarjeta mediante el usuario de mantenimiento, esto lo hará la persona encargada para este propósito, recordar que la configuración de la tarjeta consiste en:

- Ingresar la fecha y hora
- Configurar la cantidad de memorias a usar, así como a partir de memoria y parte iniciar la grabación.

Recordemos que la segunda operación sólo lo hace la persona encargada (ingeniero) de la instalación y esto una sola vez.

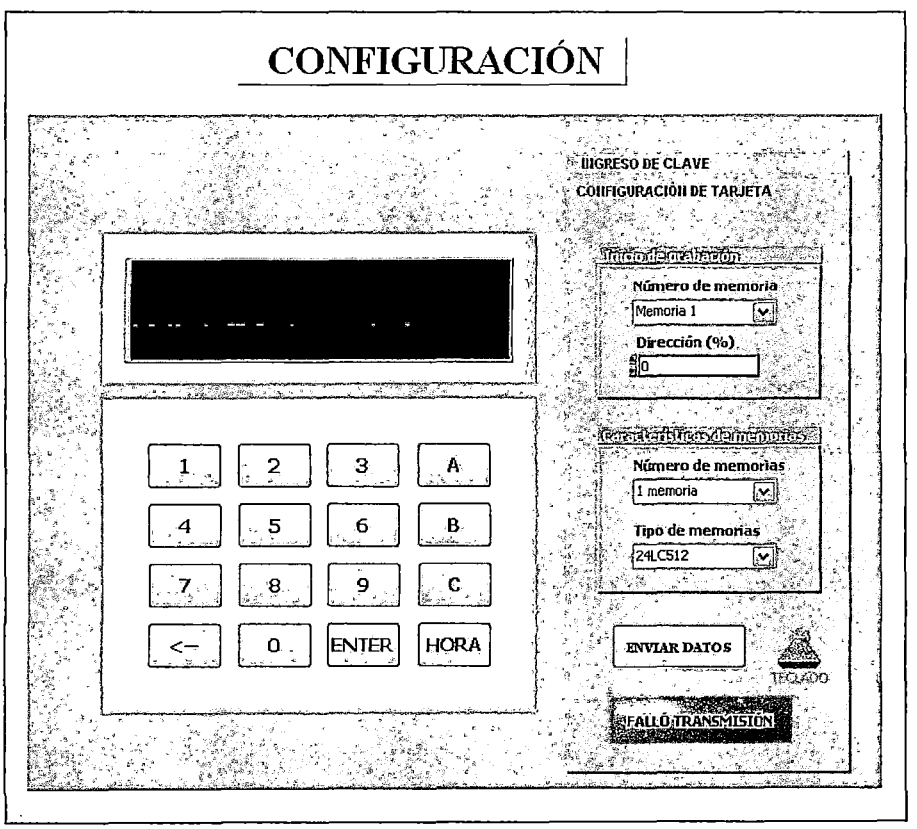



***Figura 5.12.-*** *Configuración de la tarjeta de control por software*

Luego se debe probar uno a uno todas las funciones de los subvi.

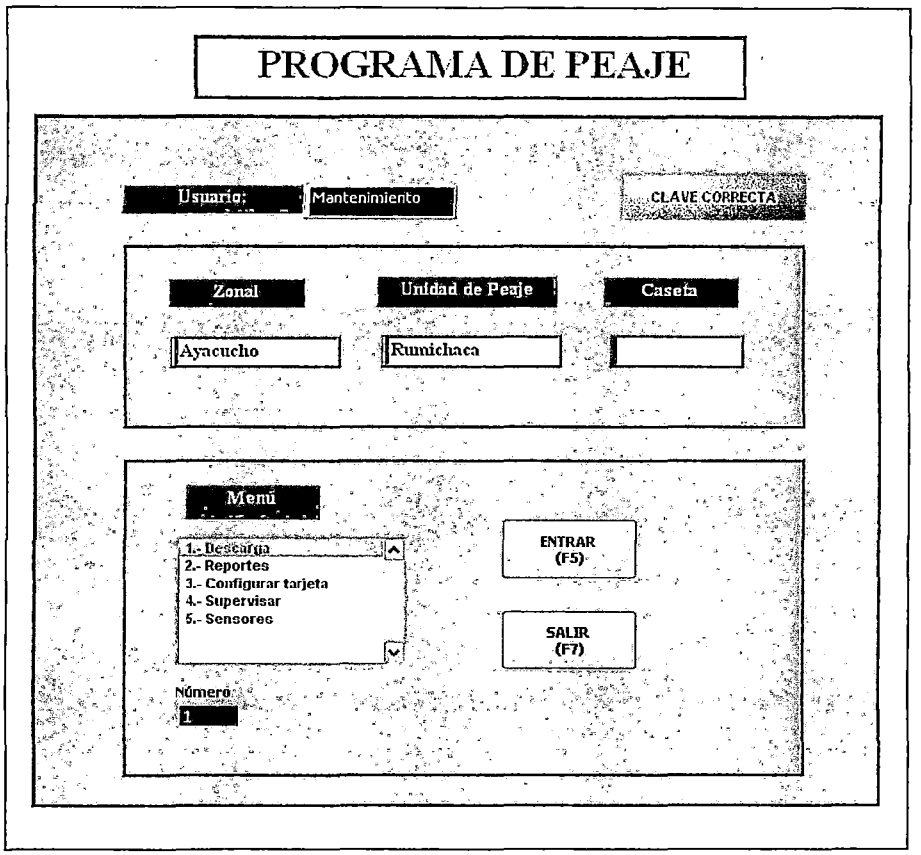



***Figura 5.13.-*** *Verificación del buen funcionamiento del software en todas sus aplicaciones, sobre todo en la descarga*

1. Descarga
2. Reportes
3. Configurar tarjeta
4. Supervisar
5. Sensores

Las pruebas realizadas deben asegurar el buen funcionamiento del sistema, para esto se debe supervisar el equipo trabajando durante un periodo de 2horas, finalizando las pruebas con la descarga de información de datos almacenados e imprimiendo los respectivos reportes; además se deben realizar las pruebas del estado de los sensores de antena y de presión.

Las pruebas de verificación del correcto funcionamiento serán las siguientes:

- Verificación de la sensibilidad de los sensores de presión. Esto asegurará que al paso del vehículo más liviano, como el caso de un auto, sea sensado; además no debe ser sensible a una bicicleta, moto o persona.
- También se debe verificar que el sistema no cuente eventos que no guarden el formato estudiado anteriormente. Por ejemplo, si una masa metálica pasa por el sensor de antena, pero al no tener ejes no debe ser considerado, ya que podría ser una carga arrastrándose.
- Se debe verificar el hecho de que no se pierda ejes en la cuenta.
- Se verificará el orden correlativo del tipo de vehículo.
- Se verificará que un vehículo sea sensado correctamente en el momento que se esté realizándose la descarga de información, sin interrumpirla.
- Se verificará si el timbre toca cada cambio de hora, a pesar de que esté realizándose la descarga de información o sensado de vehículo.
- Finalmente, se debe verificar si un vehículo fue sensado correctamente justo cuando hay un cambio de hora, ya que en ese momento se toca el timbre y esto produce ruido electrónico, pudiendo alterar la información.

# CAPÍTULO VI

# PRUEBAS Y RESULTADOS

El desarrollo de todo este sistema se realizó en varias etapas, para los cuales se hicieron diversas pruebas que tuvieron una duración de alrededor de 6 meses. Las pruebas se realizaron en el laboratorio como en el campo. Como resultado de estas pruebas se realizaron mejoras del sistema tanto en software como en hardware. Los lugares en donde inicialmente se instalaron fueron:

- Chalhuapuquio
- Huacrapuquio

Posteriormente también se instalaron en:

- Lunahuaná
- Socos
- Rumichaca
- Tambogrande
- El fiscal

Posteriormente la empresa EMAPE, para el plan verano de playas que tienen todos los años, solicitó equipos con similares cualidades de funcionamiento; además

la instalación permanente en los nuevos accesos del "Puente San Pedro" y "Arica" en Lurín. A EMAPE también se le está proponiendo el monitoreo a distancia de los equipos.

## 6.1. DISEÑO DE LOS SENSORES DE PRESIÓN

Pese a que el análisis por elementos finitos nos dio la solución de las medidas y los tipos de materiales; sin embargo en su fabricación se presentó las siguientes situaciones:

- El reencauche del neopreno fue en dos piezas; estas piezas cubrirían las láminas de acero las cuales que son las que detectan el paso de las llantas. Estas dos piezas de neopreno se unirían por pegamento; sin embargo no se encontró el adecuado, ya que el clima el la sierra y selva es extremo añadiendo a esto la gran carga y tracción de los camiones, esto disolvía el epóxico, filtrando agua y marcando continuidad como si los sensores estuvieran activados. La solución que se buscó fue de reencauchar en una sola pieza la cobertura del sensor, dejando en el extremo una abertura por donde introducir las láminas de acero, y así se usaba pegamento sólo en el extremo. Este método mejoró la vida útil del sensor, sin embargo esto sólo significó unos 4 a 5 meses. A pesar de esto se superó el diseño de los sensores que PROVIAS NACIONAL adquirió de empresas extranjeras, los cuales sólo duraban 2 a 3 meses. Finalmente ahora todo se hace de tal manera que salga en una sola pieza poniendo las láminas en el mismo reencauche, en un proceso muy delicado de fabricación. Este proceso se va perfeccionando progresivamente, ya que esto se debe supervisar cuidadosamente debido al

poco cuidado que se tienen en estos lugares de reencauche. En un procedimiento de fabricación adecuadamente supervisado, los sensores han llegado ya a superar los 2 años trabajando correctamente.

- Otro inconveniente que se presentó fue que el caucho que se adquiría, ya que algunas veces se encontraba ionizado, es decir, tenía propiedades conductoras. Esto hacía que el sensor marcará continuidad, mostrándose así activo como si existiera la presencia de un eje permanente. Esta particularidad del neopreno se constató después exhaustivas pruebas, en las que se tuvieron que destruir varios sensores.

## 6.2. DISEÑO DE LAS TARJETAS DE CONTROL

Este proceso fue en paralelo al diseño de sensores de presión, debido a su gran importancia. Desde el primer diseño que se realizó, el cual sólo clasificaba el total de vehículos que pasaban durante una hora, hasta llegar al almacenamiento de información evento por evento. Entre las dificultades mayores presentadas fueron:

- La comunicación serial USART y la I2C con las memorias no tuvieron muchos problemas cuando trabajaban en forma separada, es decir en momentos diferentes; pero cuando trabajaban simultáneamente el sistema se colgaba; por esta razón se tuvo que reestructurar el programa en el microcontrolador para que trabajara con interrupciones, además de una continua supervisión a las memorias, ya que podría darse el caso en que se esté descargando la información a la PC, pasando un vehículo (guardando la información del vehículo en las memorias) y a la vez un cambio de hora

(guardando la configuración de las memorias así como los cambios de fechas y cambios de turnos) momento en el cual el timbre tocará, pudiendo ocasionar que en algún momento que se filtre demasiado ruido y el sistema se cuelgue. Se tuvo que hacer por software un protocolo en el cual es sistema a pesar de haberse quedado en un bucle, éste siga guardando la información, y desde la PC (con el programa en LABVIEW) envíe un caracter de aviso y por precaución se cierre el programa de la PC. Antes de esto se recomendaba al personal administrativo, que iba a trabajar con el equipo, que evitaran descargar la información en un cambio de hora, no obstante esto no se podía evitar, hasta que se llegó a la solución ya antes mencionada. Esto conllevó a realizar varios viajes a la sierra para finalmente corregir el inconveniente, ya que aquí en Lima no podíamos (algunas veces) provocar el mismo evento que generaba el error, incluso había soluciones que aquí en Lima funcionaban correctamente, pero a la hora de llevarlo a la sierra por alguna razón fallaban en algún momento. El sistema no se ha vuelto a colgar o fallar desde que se llegó a la solución. Estos cambios se hicieron luego en todos los peajes mencionados; sin embargo particularmente había un peaje que no fallaba con las primeras soluciones, pero por precaución también se le hizo el cambio del programa.

## 6.3. DISEÑO DE LAS TARJETAS ACONDICIONADORAS DE SEÑALES

El principal inconveniente que se tuvo fue en afinar la sensibilidad del sensor de antena o loop. En un inicio no se había considerado los cambios de temperatura, esto ocasionaba, en algunos casos, que el sensor de antena no

detecte algunos vehículos livianos, y en otros casos se volvía muy sensible detectando masas muy pequeñas. Esto se daba porque la señal de comparación para la detección del vehículo era constante, y el voltaje del circuito de salida del sensor variaba con la temperatura dado que el clima en la sierra y selva varía drásticamente. Por eso después se puso el circuito compensador detallado en el capítulo III.

## 6.4. DISEÑO DEL SOFTWARE

Es sensato pensar que cada vez que se mejoraba o cambiaba el programa en el microcontrolador, el software de la PC también lo hacía, ya que las pruebas eran continuas. Además, dependiendo del requerimiento del usuario, se tuvo que ir añadiendo funciones, hasta llegar a lo que se explicó detalladamente en el CAPÍTULO IV.

- Una de las cosas importantes era mantener, en lo posible, la información codificada y así evitar alguna forma de manipularla. Esto se logró combinando códigos en ASCCI con números en hexadecimal, y además marcas a nivel de código binario. Muchos subVI fueron creados para este propósito, además cada vez que se descarga la información desde la PC, se crea un archivo en un formato que sólo será leído por el programa. Se crearon subVI mediante el cual se puede buscar información sobre este archivo codificado, el cual se guarda en el disco **C**, para luego decodificarlo y generar los reportes.
- Uno de los inconvenientes más resaltantes fue el hecho de la posibilidad que existan horas repetidas; es decir, si por alguna razón el conteo se detuvo y

luego se volvió a encender el equipo en esa misma hora que se detuvo, entonces esto ocasionaba horas repetidas, en las cuales hay información no repetida. Este detalle, a la hora de buscar información por el software, provocaba que sólo mostraba la información de la primera hora la cual no era la única correspondiente a esa hora. Esto fue superado mejorando el programa en LABVIEW para buscar esa hora reiteradamente tantas veces como el usuario crea conveniente. Esto aparece en el programa como ***número de insistencias***, que por defecto aparece con el valor de 1; esto significa que si el programa encuentra la hora respectiva, descarga la información hasta la marca respectiva y luego vuelve ha realizar la búsqueda pero a partir de la posición en que se quedó.

- También existe la posibilidad de que la comunicación falle en algún momento y el programa se cuelgue al perder un carácter. Esto ocasionaba que el microcontrolador se quede en un bucle. Se superó esto haciendo que el programa en LABVIEW envíe un caracter al PIC que significará que la comunicación falló e inmediatamente se cerrará el programa de la PC.
- También para proteger la manipulación inapropiada de la tarjeta se implementó un teclado virtual en LABVIEW, todo por comunicación USART.

## 6.5. PLAN DE MANTENIMIENTO

El plan de mantenimiento se empezará a regir inmediatamente después al mes siguiente de la conformidad y buen funcionamiento del equipo. El mantenimiento se realizará mensualmente siguiendo la siguiente rutina:

- Limpieza interna del tablero eléctrico.
- Chequeo de la carga y/o consumo
- Control de fecha
- Control de hora
- Control de carga y estado de baterías
- Control del programa de extracción de información
- Chequeo y control del software por usuario
- Chequeo de cableado
- Chequeo de calibración de sensores de antena
- Chequeo de la sensibilidad de sensores de presión
- Verificación de timbre
- Ajuste general
- Control y mantenimiento del Pozo de tierra.
- Impresión de reportes

Además es importante mencionar la actualización de la mejora continua del programa del microcontrolador, para prever ciertos eventos imprevistos que pudieran afectar la seguridad de la información.

En caso de que alguna tarjeta sufra algún desperfecto, esta será reemplazada por una tarjeta adicional inmediatamente, llevándose la defectuosa para su análisis.

La revisión y mantenimiento de los equipos electrónicos interrumpirá el normal conteo de vehículos inmediatamente iniciado un nuevo turno, no se puede interrumpir una hora ya iniciada, sino que se tendrá que esperar el cambio de hora por la señal del timbre.

Se levantará un informe de mantenimiento recalcando la hora a la que fue realizado para posteriormente indicar que el conteo electrónico fue suspendido momentáneamente en la respectiva hora y se tuvo que hacer manualmente. Además se debe incluir en el informe la hora en que se realizó el diagnostico del equipo. Si este fue apagado alguna vez durante ese turno, además de la hora de mantenimiento, será fácil de ver por el informe extraído.

Finalmente se sabe que las memorias EEPROM del microcontrolador y las externas 24LCXXX tiene una cantidad limitada de veces borrado/escritura, por esta razón pese a que esta es un millón y por la cantidad de vehículos que pasan tendremos suficiente para mucho más de 10 años, sin embargo estas se cambiarán totalmente cada 2 a 3 años, solo por prevención.

# CAPÍTULO VII

# COSTOS

Los costos aquí presentados no representan el precio de venta del equipo, sino son sólo los materiales usados.

A continuación se muestra en tablas el detalle de costos para cada una de las partes del proyecto.

***Sensores de ejes o peanas***

***Tabla 8.1***

| *Materiales* | *S/.* |
|---|---|
| Láminas de acero templado preparadas y cortadas | 50 |
| Vulcanizado de Neoprene | 75 |
| Cables de silicona | 1 |
| Total | 126 |

***Sensores inductivos de unidades de vehículos o sensor loop de antena***

***Tabla 8.2***

| *Materiales* | *S/.* |
|---|---|
| Cable número 16 (2 vueltas 20m) | 10 |
| Tubos PVC | 10 |
| Lepóxico | 100 |
| Total | 120 |

*Instalaciones eléctricas*

*Tabla 8.3*

| Materiales | S/. |
|---|---|
| Cable endoprene número 14 (rollo de 100m) | 250 |
| Cable telefónico de acometida (rollo de 300m) | 50 |
| Total | 300 |

*Tablero eléctrico*

*Tabla 8.4*

| Materiales | S/. |
|---|---|
| Gabinete metálico | 300 |
| Cargador de baterías con regulación de voltaje y corriente | 650 |
| Baterías secas de 12V 17Ah (2 unidades) | 200 |
| Llaves térmicas para DC (5A – 24V) | 30 |
| Llaves térmicas para DC (10A – 220V) | 30 |
| 1 amperímetro y 1 voltímetro indicador | 10 |
| 1 indicador luminoso de 220V AC | 5 |
| Cables | 10 |
| Regletas | 15 |
| tornillos y pernos | 3 |
| Total | 1253 |

*Tarjeta acondicionadora de señal para el sensor loop de antena*

*Tabla 8.5*

| Materiales | Cantidad | S/. |
|---|---|---|
| LM324 | 1 | 3 |
| Regulador de 7818 | 1 | 1.5 |
| Opto triac MOC3021 | 1 | 2.5 |
| Bobina de sintonía | 1 | 20 |
| Diodos (1N914, 1amp) | 8 | 1 |
| Switch | 1 | 0.5 |
| Resistencias(390, 220, 500,1k,5k,470k,100k,2M) | 38 | 1 |
| Transistor BC548 | 4 | 1 |
| Transistor BC547 | 1 | 1 |
| Capacitores (100uF, 0.1uF, 10uF, 0.22uF, 1000uF/25V, 470uF/25V) | 18 | 2 |
| Led | | 1 |
| Boneras | | 6 |
| Placa | | 20 |
| Total | | 60.5 |

*Tarjeta para sensores de temperatura, estado de puerta y voltaje de baterías*

*Tabla 8.6*

| Materiales | Cantidad | S/. |
|---|---|---|
| Transistor 2N2222 | 1 | 0.5 |
| LM324 | 1 | 3 |
| Regulador 7809 | 1 | 1.5 |
| Regulador 7805 | 1 | 1.5 |
| Regulador LM317 | 1 | 1.5 |
| Diodos | 3 | 1 |
| Switch | 1 | 0.5 |
| Condensadores(0.1uF, 10uF) | 7 | 1 |
| Resistencias | | 1 |
| Placa | | 20 |
| Borneras | | 6 |
| Total | | 37.5 |

*Tarjeta acondicionadora de señal para los sensores de ejes (peanas)*

*Tabla 8.7*

| Materiales | Cantidad | S/. |
|---|---|---|
| NAND 4093 | 1 | 1 |
| NOT 7404 | 1 | 2 |
| TIMER 555 | 1 | 2 |
| Regulador 7818 | 1 | 1.5 |
| Regulador 7805 | 1 | 1.5 |
| Transistor BC548 | 4 | 1 |
| Relay | 3 | 6 |
| Diodos (1amperio, 1n914) | 16 | 2 |
| Capacitores (0.22uF, 0.1uF, 1000uF/25V) | 16 | 3 |
| Switch | 3 | 1 |
| Resistencias (220, 20k, 220k, 1k, 2k, 5k) | 21 | 2 |
| Borneras | | 6 |
| Placa | | 20 |
| Total | | 49 |

*Tarjeta de control principal*

*Tabla 8.8*

| *Materiales* | *Cantidad* | *S/.* |
|---|---|---|
| PIC16F877A | 1 | 25 |
| Pantalla LCD 2x16 | 1 | 25 |
| MAX232 | 1 | 4 |
| Memorias 24LC256 | 4 | 20 |
| Conector DB9 hembra | 1 | 5 |
| Relay | 1 | 2 |
| Timbre | 1 | 7 |
| Cristal(32.768kHz, 20MHz) | 2 | 3 |
| Transistor BC548 | 1 | 1 |
| Resistencias (10k, 0.5k, 20k, 2k, 5k) | 9 | 2 |
| Condensadores (0.1uF, 1uF, 22pF, 33pF ) | 20 | 2 |
| Regulador 7809 | 1 | 1.5 |
| Regulador 7805 | 1 | 1.5 |
| Diodos | 3 | 2 |
| Borneras | 4 | 6 |
| Placa | | 20 |
| Total | | 127 |

**Consolidados de costos del proyecto**

Aquí se muestra el costo total del proyecto, sin considerar los costos iniciales y únicos que se tienen para el diseño de la matriz de vulcanizado del neopreno y el costo de la licencia del software.

Matriz de vulcanizado del neopreno = S/ 2000

Software LABVIEW = S/ 7500

El costo del software es único, sin embargo podría reducirse este costo valiéndose de la compra de la autorización para generar un programa ejecutable

instalador, a alguna institución que tenga la licencia de LABVIEW, como *Inducontrol*, el cual puede facilitar mediante un acuerdo convertir el programa diseñado a un ejecutable instalador.

***Tabla 8.9***

| Descripción | Cantidad | c/u | Costo (S/.) |
|---|---|---|---|
| Sensores de presión (peanas) | 4 | 126 | 504 |
| Sensores inductivos (antena, loop) | 1 | 120 | 120 |
| Canaletas metálicas para las peanas | 4 | 250 | 1000 |
| Loza de concreto (280kg/cm^2) | 1 | 3000 | 3000 |
| Instalaciones eléctricas | 1 | 300 | 300 |
| Tablero eléctrico | 1 | 1253 | 1253 |
| Tarjeta sensor loop antena | 1 | 60.5 | 60.5 |
| Tarjeta de de sensores: temperatura, estado de puerta y batería | 1 | 37.5 | 37.5 |
| Tarjeta acondicionadora de señal para los sensores de peanas | 1 | 49 | 49 |
| Tarjeta principal de control | 1 | 127 | 127 |
| Pozo de tierra | 1 | 1500 | 1500 |
| Otros | 1 | 500 | 500 |
| Total | | | 8451 |

El precio de sistemas similares son de $25 000 por vía, siendo sistemas que no clasifican vehículos, lo que significa que para detectar tanto camiones como autos necesitaremos como mínimo de 2 vías, esto significaría $50 000 de inversión en cada unidad de peaje, teniendo en cuenta que el costo de mantenimiento no está incluido, estos sistemas no serían adecuados en estos peajes de lugares alejados de la sierra y selva. Además de ello, estos equipos necesitan de fluido eléctrico continuo, ya que en otros países no existe el problema de energía como el que hay en la sierra y selva del Perú, lo que encarecería el costo. Con todo esto podemos apreciar en la figura 7.1 que el costo del diseño de este sistema es económicamente factible y deja ganancias

en un tiempo más corto que otros. Para el costo total del equipo se incluirá la licencia del software $2500 y la matriz de los sensores de presión $600, que realmente sólo es un costo único. Luego el costo total sería $8450 (2850+5000+600).

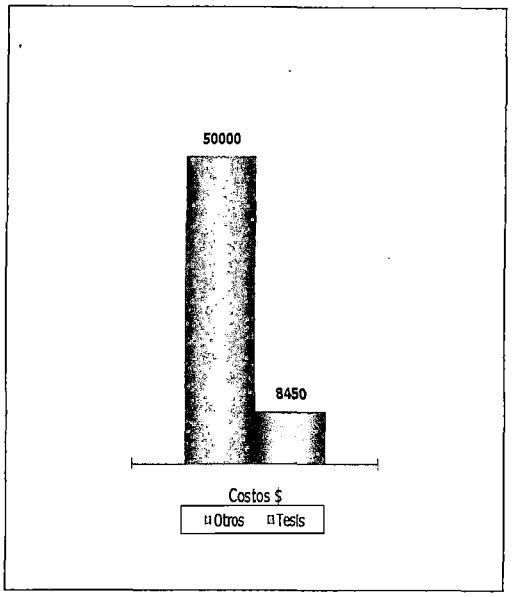



***Figura 7.1.-*** *Comparación de costos*

Para evaluar el proyecto tendremos en cuenta el ingreso promedio diario ($2000) en cada uno de estos peajes y los costos de mantenimiento.

Costo del equipo = $8450

Costo de mantenimiento = $300

***Tabla 7.1***
*Comparación del 1° mes de instalación de los equipos*

| | Tesis | Otros |
|---|---|---|
| Equipo | $8450 | $50000 |
| Mantenimiento | $300 | $1500 |
| Total 1º mes | $8750 | $51500 |

No es necesario como podemos ver un análisis de VAN o TIR ya que un mes es suficiente para recuperar y tener ganancias. Además el principal inconveniente económico sería el tener un grupo electrógeno permanentemente durante el día.

# CAPÍTULO VIII

# PROPUESTA DE AMPLIACIÓN PARA RED INTERCONECTADA DE TODOS LOS PEAJES

## 8.1. INTRODUCCIÓN

Los sistemas SCADA y de monitoreo a distancia cada vez son más necesarios, ya que conllevan al ahorro y una supervisión en tiempo real adecuada. Para esto hay muchas formas de lograrlo; pero como se trata siempre de abaratar costos, es que optó como medio de comunicación para realizar el monitoreo por INTERNET, es decir que se conectará la tarjeta de control de cada peaje a INTERNET y en la central en Lima se tendrá un programa en LABVIEW por medio del cual se podrá acceder directamente a la tarjeta (ver figura 8.1). Es decir, se dispondrá de una pantalla en el que se encuentre un mapa del Perú, desde el cual se verá el estado de las tarjetas, así como todas las funciones del programa en LABVIEW que se describieron en este proyecto.

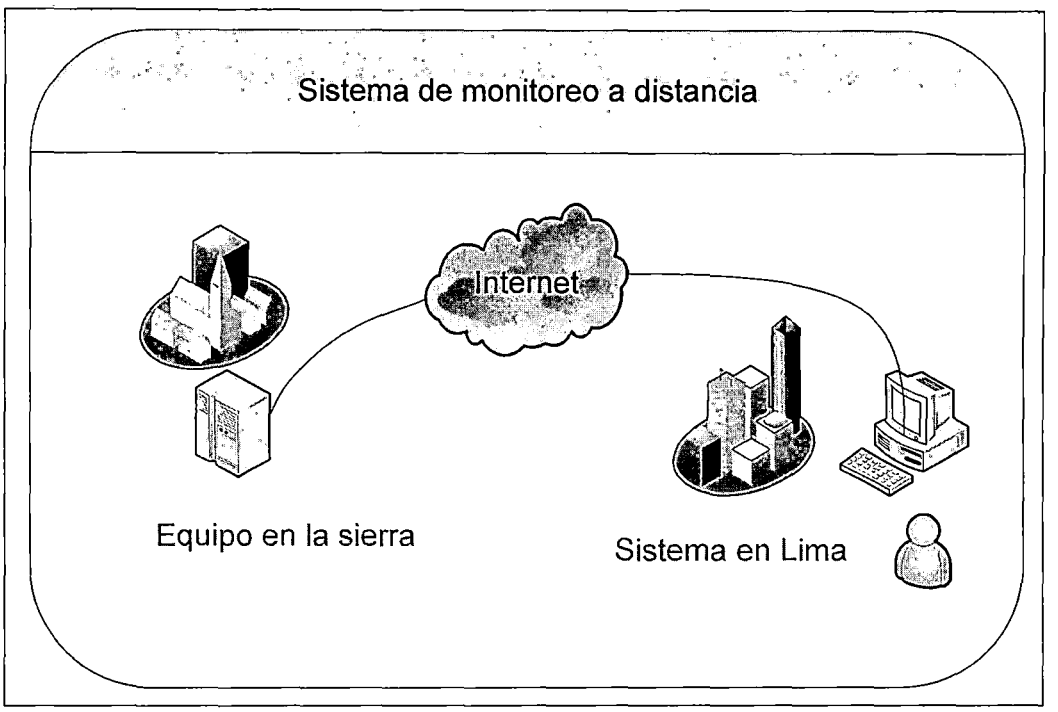



***Figura 8.1.-*** *Esquema del sistema interconectado remoto*

Para esto debemos recordar que la tarjeta ya tiene la función de enviar los siguientes datos: temperatura, nivel de voltaje de las baterías y estado de la puerta del tablero eléctrico. El programa entonces monitoreará todas las tarjetas para tener siempre éstas variables.

Es importante recalcar que para poder añadir esta función no es necesario detener el proceso permanente de cuenta, ni cambiar la tarjeta, sino que diseñar el programa supervisor a distancia en LABVIEW; pero si es necesario añadir algún elemento que convierta el protocolo RS232 en TCP/IP para que se establezca la comunicación con el servidor principal en Lima. Con este fin es que se utilizará los servidores seriales TIBBO.

## 8.2. SERVIDORES SERIALES TIBBO

### 8.2.1. Descripción

Es un conversor Serial-Ethernet que conecta externamente cualquier equipo Serial (RS-232/RS-485) a una red LAN Ethernet. El Servidor DS100 y DS202 son ideales para Integradores de Sistemas que deseen conectar en red instalaciones o equipos existentes. Sin embargo también se puede conectar a Internet mediante asignándole una IP pública, que es lo que necesitamos para este propósito. Las pruebas para este fin se realizaron y fueron exitosas.

### 8.2.2. El servidor DS202

El equipo se conectará a la red de Internet (TCP/IP) usando el Servidor Serial DS202. El DS202 es un dispositivo externo que rutea transparentemente los datos entre el puerto Serial y la red. (Ver figura 8.2).

El **DS202** es el nuevo integrante de la familia de Servidores Seriales de Tibbo. Midiendo solamente 60x47x30mm, el DS202 ofrece un conjunto poderoso de características que incluyen un puerto 100BaseT de red, una fuente de alimentación de grado industrial con un rango de entrada de 9 a 25v, grandes buffers de datos, un riel DIN de montaje (opcional), y (por último pero no menos importante) un agradable diseño. El DS202 está equipado con un firmware sofisticado y el software de PC del set de herramientas para Servidores de Dispositivos (DST – Device Server Toolkit) que hace que el conectar el dispositivo serie a la red sea un trabajo fácil. Solo

se conectará el DS202 al dispositivo serie y a la red, se correrá el Wizard de Conexión (que es parte del DST), se responde algunas simples preguntas y eso será todo, el dispositivo serie se estará comunicando a través de la red.

La tarjeta de control ahora está conectada a la red pero esto es solo la mitad del trabajo, ver figura 8.3. El software de control en LABVIEW existente, está trabajando con el protocolo para acceder al equipo serial a través del puerto serial. No hay problema, para esto se instalará el software del "**Tibbo**" que es el **Virtual Serial Port Driver** (VSPD) for Windows. Realmente lo que se hará es crear un puerto serial virtual en la red. Para cualquier programa Windows los Puertos Seriales Virtuales se verán como puertos COM reales. En realidad, el driver VSP enruta transparentemente los datos al DS202 y al equipo "detrás" de este, de esta forma el Software de control en la PC puede trabajar por la red sin ninguna modificación.

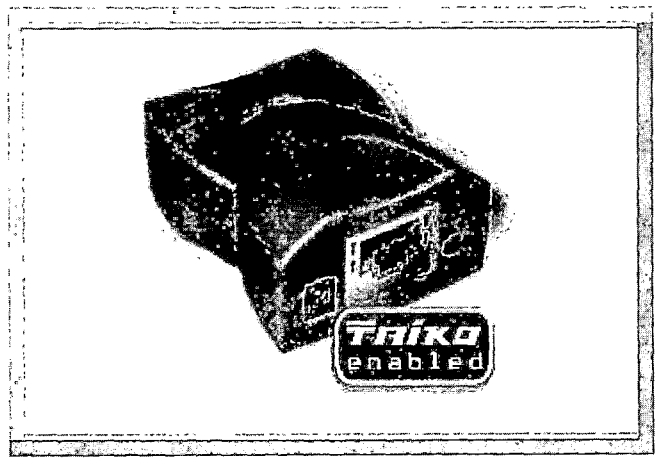



***Figura 8.2.-*** *Servidor Tibbo DS202*

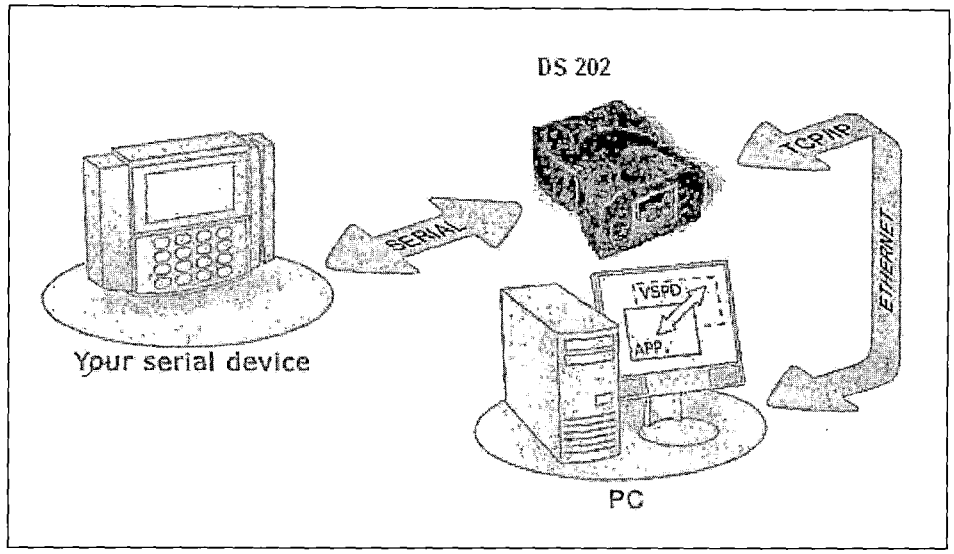



***Figura 8.3.-*** *Conexión del servidor Tibbo*

## 8.3. CONFIGURACIÓN DE LOS SERVIDORES SERIALES

El programa de supervisión se desarrollo como ejemplo para 4 peajes ubicados en diferentes partes del Perú, en los cuales se encontrarán localizados los equipos de control. El programa ha sido diseñado de tal manera que los equipos instalados en los peajes no necesitarán modificación alguna, sino sólo la a conexión del módulo tibbo DS202 previamente configurados y la instalación de un punto de internet. El software para la configuración se muestra en la figura 8.4.

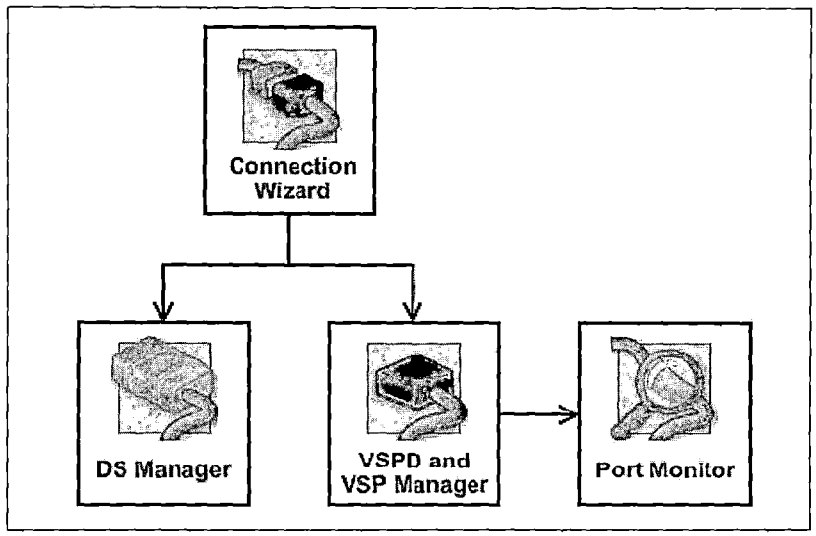



***Figura 8.4.-*** *Softwares de configuración de los servidores Tibbo*

Se crearán puertos virtuales con la aplicación VSP Manager para esto se añadirá un puerto virtual de acuerdo a la figura 8.6.

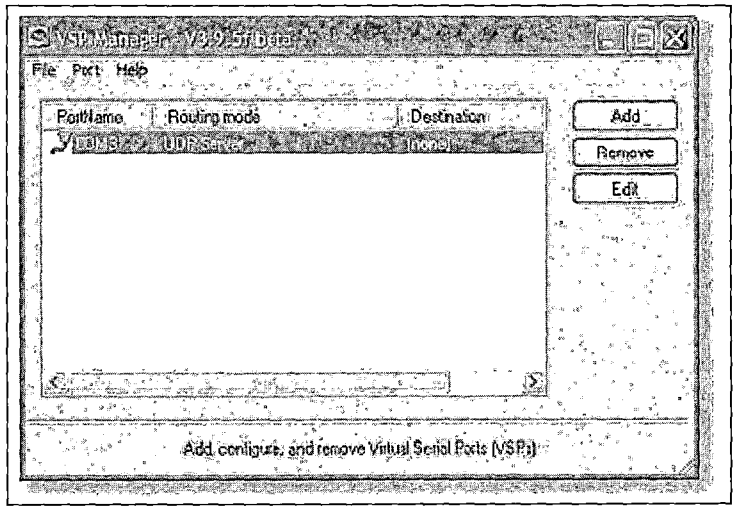



***Figura 8.5.-*** *Creación del puerto virtual*

Con el botón **Add**, que podemos ver en la figura 8.5, se pueden añadir tantos puertos virtuales como se deseen para comunicar con los servidores.

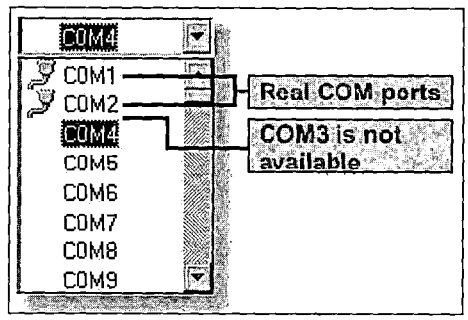



***Figura 8.6.-*** *Puertos COM virtuales a poder crearse*

Estos puertos se conectarán virtualmente a cada dispositivo de acuerdo a las siguientes posibilidades, de existir más de una tarjeta por unidad de peaje como la figura 8.8 y 8.9.

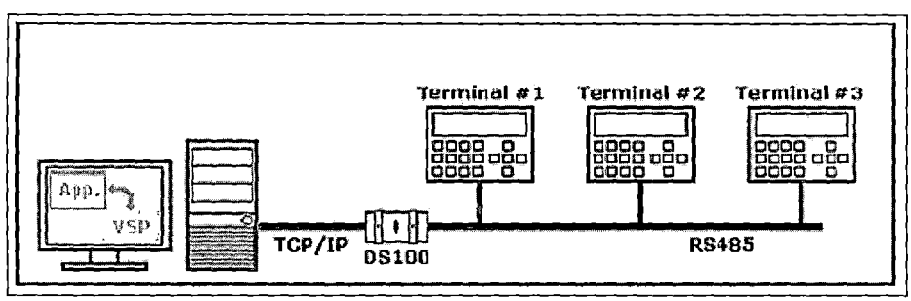



***Figura 8.8.-*** *Conexión con un Tibbo*

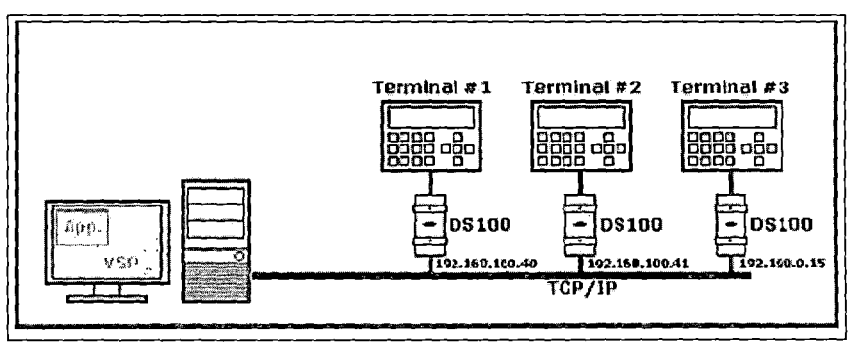



***Figura 8.9.-*** *Conexión con tres servidores Tibbo*

Cada puerto virtual se configurará a una dirección IP dentro de la red LAN o la IP pública para Internet, con el respectivo protocolo a usar para la configuración, como se muestra en la figura 8.10.

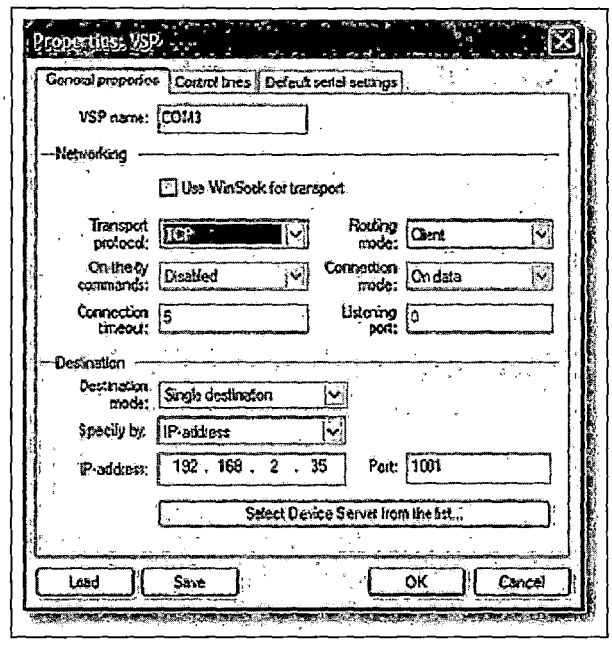



***Figura 8.10.-*** *Asignación de dirección IP*

Además se puede seleccionar la velocidad en baudios para la comunicación RS232/RS485 así como se puede apreciar en la figura 8.11.

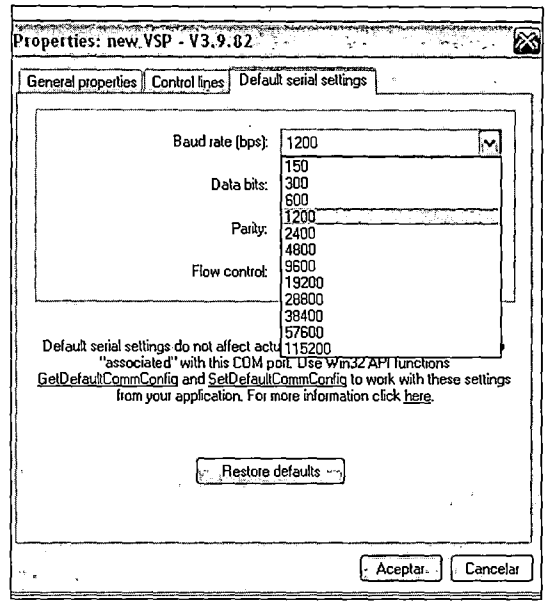



***Figura 8.11.-*** *Asignación de velocidad en baudios para el puerto virtual*

Se puede verificar los puertos virtuales creados en el administrador de dispositivos de Windows, tal como se muestra en la figura 8.12.

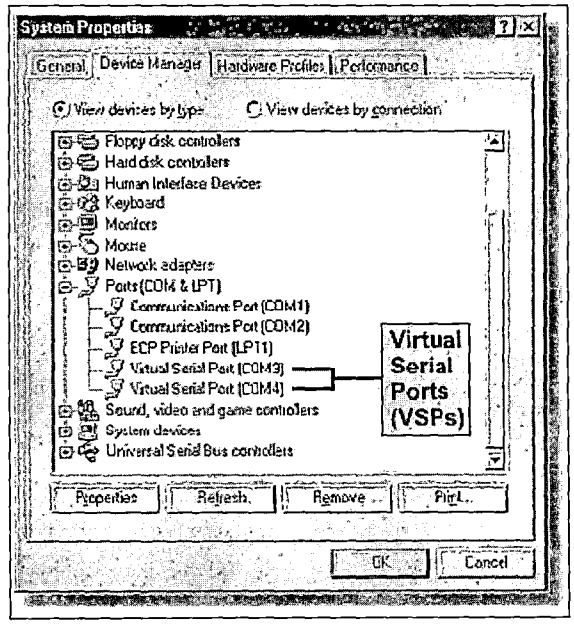



***Figura 8.12.-*** *Visualización de los puertos virtuales creados en el sistema operativo Windows*

Se tiene una aplicación que es el Port Monitor para la verificación y escaneo de éstos puertos. Ver figura 8.13 y 8.14.

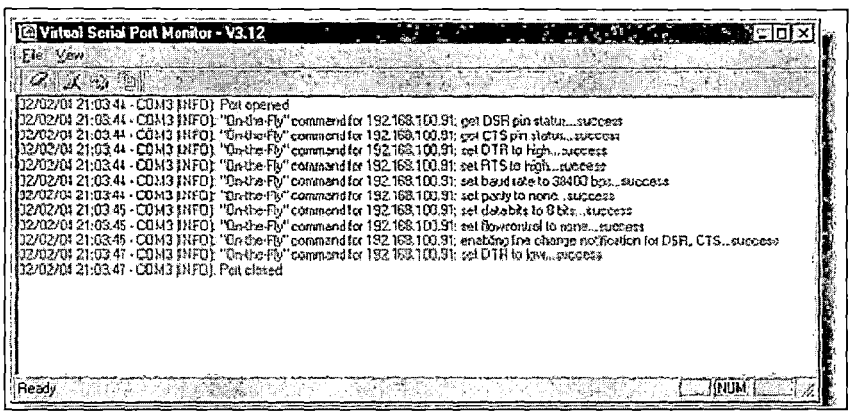



***Figura 8.13.-*** *Verificación del estado de los puertos creados*

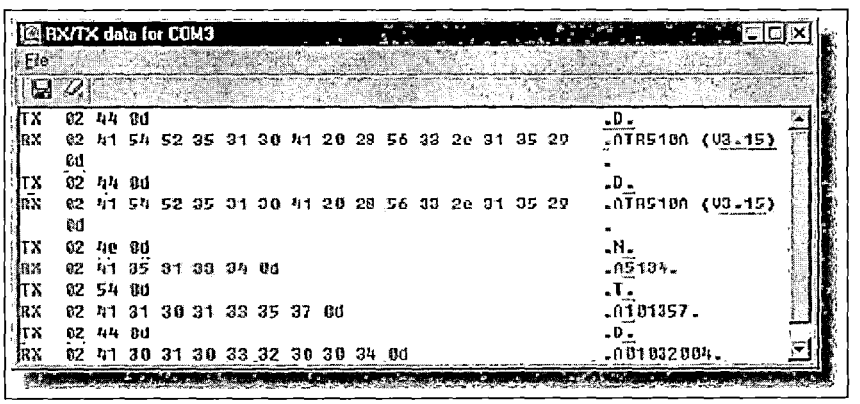



***Figura 8.14.-*** *Prueba de envío y recepción de datos por el puerto virtual*

Luego de crear estos puertos virtuales, se buscarán en la red todos los servidores seriales TIBO conectados a la red, mediante la aplicación DS Manager, de acuerdo a la figura 8.15.

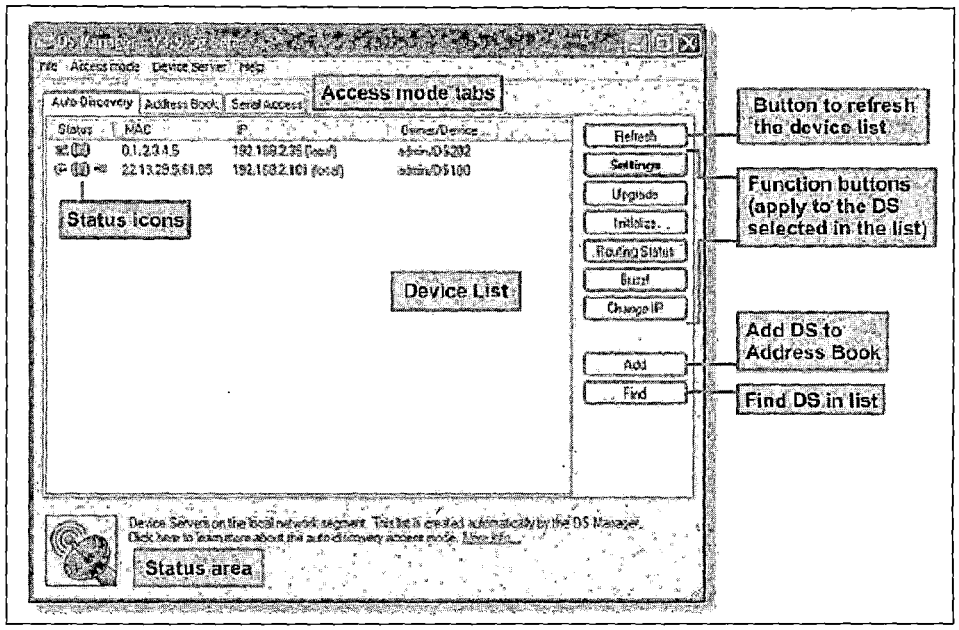



***Figura 8.15.-*** *Entorno del DS Manager para el monitoreo de los servidores tibbo conectados a red*

## 8.4. PROGRAMA SUPERVISOR EN LABVIEW

Aquí se mostrará el desarrollo del VI principal, el cual se encontrará en el computador principal de la central en Lima. Por medio de este software se supervisará los sistemas de control de los peajes y además se podrá descargar toda la información, teniendo un control total. En la figura 8.16 se muestra el panel frontal de programa.

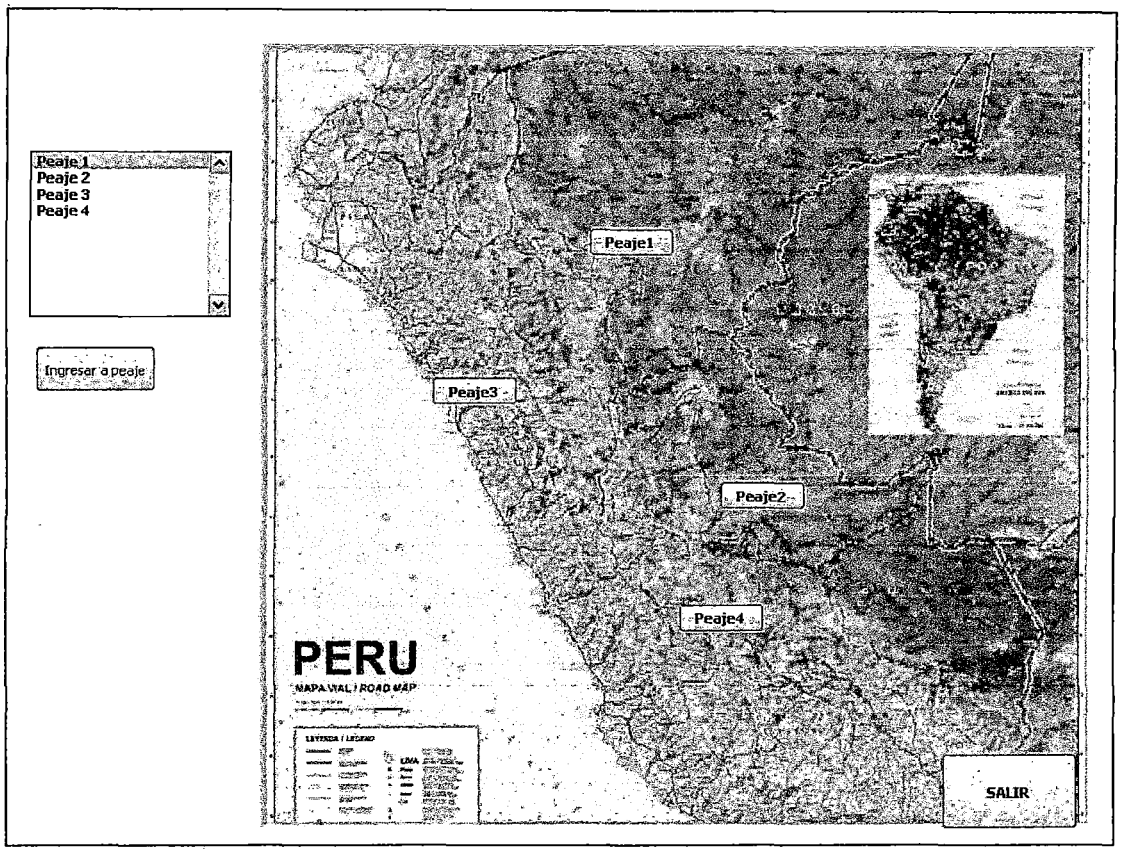



***Figura 8.16.-*** *Panel frontal de programa de supervisión interconectado, desarrollado en Labview*

Esto significará que en cada punto de los peajes ya no existirá el programa instalado. Este programa principal tendrá todas las funciones ya antes mencionadas por el programa principal y además un sistema de supervisión del estado de los siguientes parámetros:

- Estado de la puerta del tablero principal
- Temperatura
- Voltaje de la batería

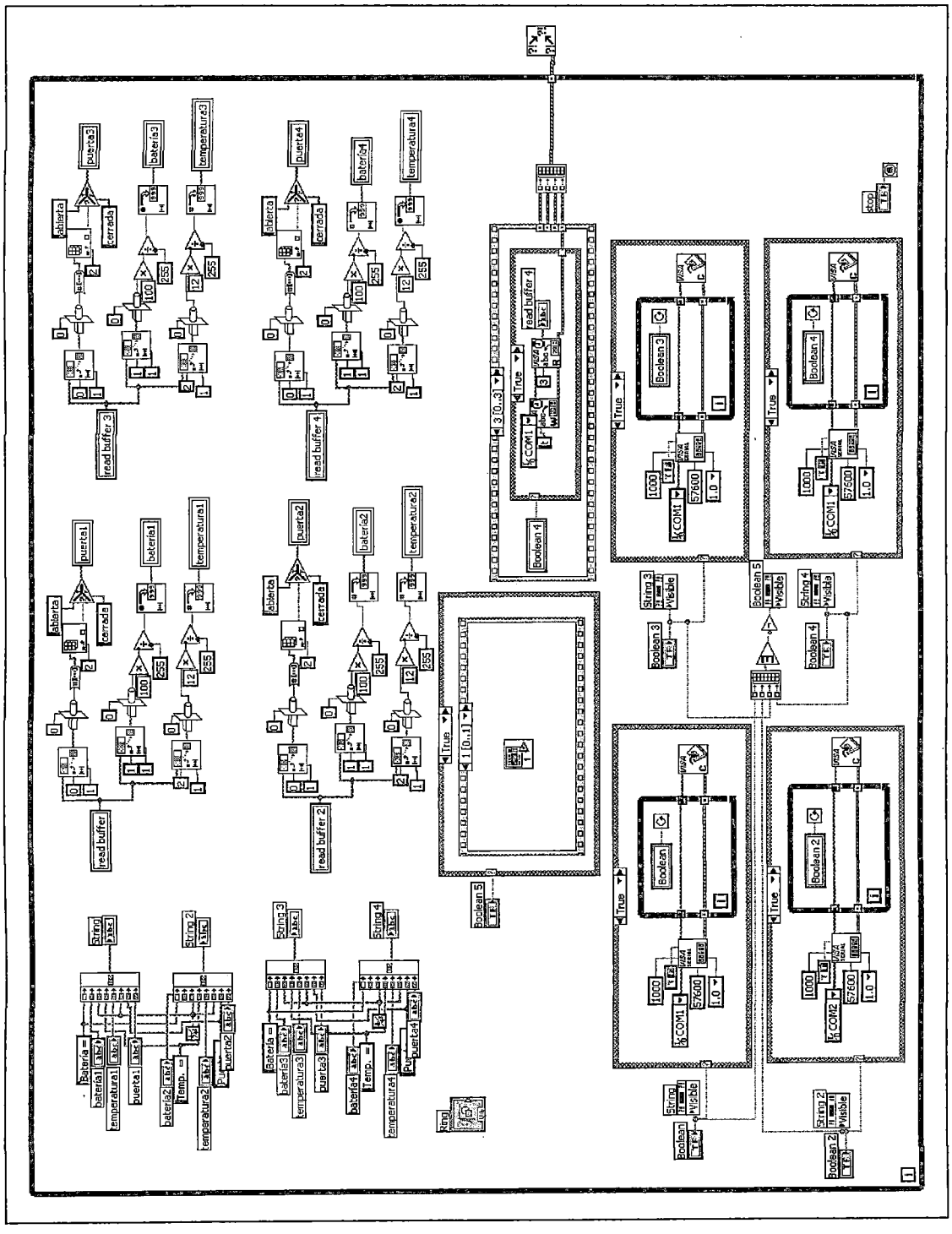


*Figura 8.17.- Diagrama de bloques del sistema de supervisión por Internet*

Esto se hace usando los puertos COM virtuales creados con uno de los software del tibbo. Para esto se tuvo el cuidado de trabajar con varios bucles

while para la lectura de cada uno de los puertos virtuales creados los cuales se conectan directamente con los equipos instalados. En la figura 8.17 se puede apreciar el diagrama de bloques del programa desarrollado.

A este programa se añadirá el software diseñado del capítulo 4, de tal manera que se tendrán todas las funciones que fueron descritas con detalles en ese capítulo y además los parámetros de temperatura, estado de puerta y nivel de carga de baterías.

# CONCLUSIONES

Finalmente, con el desarrollo del presente trabajo, se logró alcanzar los objetivos trazados:

- Se desarrolló e implementó todo el sistema de control y supervisión de peajes, para las zonas alejadas en la sierra y selva del Perú. Para conseguir esto, se hicieron diversas pruebas en algunos peajes como: Huacrapuquio (Huancayo-Junín), Challhuapuquio (San Ramón-Junín), resultando éstas satisfactorias después de hacer las correcciones debidas, perfeccionando el sistema mientras se desarrollaban estas pruebas en campo y en "laboratorio". Las pruebas demoraron hasta 6 meses, hasta llegar al 100% de confiabilidad en el almacenamiento de la información.
- Se desarrolló una tarjeta de control eficiente y confiable basado en la tecnología de microcontroladores.
- Se consiguió tener una capacidad de almacenamiento de información de hasta casi dos meses, sobrepasando el límite trazado.
- Se logró desarrollar un software de interface en una PC basado en el lenguaje de programación visual LABVIEW, con el que se hicieron las siguientes funciones: Configurar la tarjeta de control, supervisión en tiempo real, pruebas para

mantenimiento del equipo y estado de sensores, descarga de información y generación de reportes.

- Se logró mejorar el diseño de los sensores de presión, incrementando su vida útil, llegando así a superar los 2 años sin mantenimiento alguno. De esta manera se superó a los sensores existentes en el mercado, quienes en estas condiciones de trabajo llegaban sólo hasta 3 meses.
- Se está concluyendo el desarrollo del sistema de supervisión y monitoreo a distancia, realizando las pruebas y añadiendo la aplicación en el software principal.
- Finalmente si comparamos los precios del sistema de control de peajes diseñado en este trabajo, con los que se ofrecen en el extranjero, se observó una gran diferencia. Además el software de gestión para estos equipos puede llegar a costar unos $10 000, recordando también que éstos demandan de energía continua para mantenerlos operativos.

## OBSERVACIONES Y RECOMENDACIONES

Es importante mencionar, que el proceso de fabricación de los sensores de presión es toda una técnica, ya que se tuvieron muchos inconvenientes, y que después de muchas pruebas se fueron perfeccionando cada vez que se construía uno.

Es importante también una constante revisión de los equipos para anticiparse a cualquier imprevisto, mejorando así las versiones del software de la tarjeta de control y del programa en LABVIEW.

El sistema desarrollado ha resultado práctico y seguro, siendo una de las razones de esto su capacidad de clasificar vehículos con una sola vía; es decir diferenciar los autos de los camiones. Esto es importante ya que cualquier otro sistema normalmente requiere de 2 vías para esto, incrementando el costo del equipo.

Este sistema ya está siendo probado por EMAPE en el plan verano de playas y en algunos otros lugares de accesos pequeños, cumpliendo satisfactoriamente su función. Eso ha llevado a pensar muy seriamente al crecimiento de aplicaciones para grandes peajes.

Es importante tener presente, sobre todo para la instalación y funcionamiento, lo siguiente:

- Todas las instalaciones de cableado deben ser soldadas, selladas e impermeabilizadas con epóxico, posteriormente a ello se debe realizar la prueba de aislamiento con megómetro, el cual debe marcar 5Mohmios como mínimo.
- Se debe tener cuidado de que los vehículos, una vez que ingresan al área de sensado, no retrocedan o no realicen un giro exagerado porque podría ocasionar un error de sensado.
- No se debe permitir el paso de vehículos arrastrando objetos metálicos (máquinas agrícolas) porque dañarían los sensores de presión y ocasionarían errores de lectura.
- Todos los sensores se deben instalar haciendo la loza de concreto de las características señaladas, ya que de otro modo se deterioraría la pista y como consecuencia dañarse el sensor de antena incluso los sensores de presión.
- Es importante la medición del pozo de tierra antes de arrancar el sistema, ya que especialmente en la sierra las cargas estáticas son altas, y esto podría dañar los equipos.
- Para el mantenimiento e instalación de tarjetas electrónicas es indispensable el uso del brazalete de descarga estática; para esto ya se debe contar con el pozo de tierra.
- Se puede precisar que no influye significativamente la distancia desde el sensor de antena al circuito detector de masa metálica, ya que se trabaja a frecuencias relativamente altas (80kHz) y baja corriente, en orden de miliamperios. Esta corriente no la podemos medir con el multímetro por ser de frecuencia alta; pero

sí se puede medir la resistencia del cable conductor con la antena (sensor inductivo) conectada, el cual su valor debe estar entre 0.2 ohmios a una distancia de 5m hasta 5 ohmios a una distancia de más de 100m.

- Para la implementación de los circuitos y sensores se revisó varias literaturas correspondientes; sin embargo se demuestra que la implementación en laboratorio muchas veces difiere con a la del "campo". Por esto se debe tener presente la importancia de someter el equipo a las pruebas más severas antes de llevarlo al "campo". Esto se debe realizar en pequeños bancos de pruebas que traten de emular lo más cerca posible a las condiciones que estarán sometidos.

# BIBLIOGRAFÍA

***Libros:***

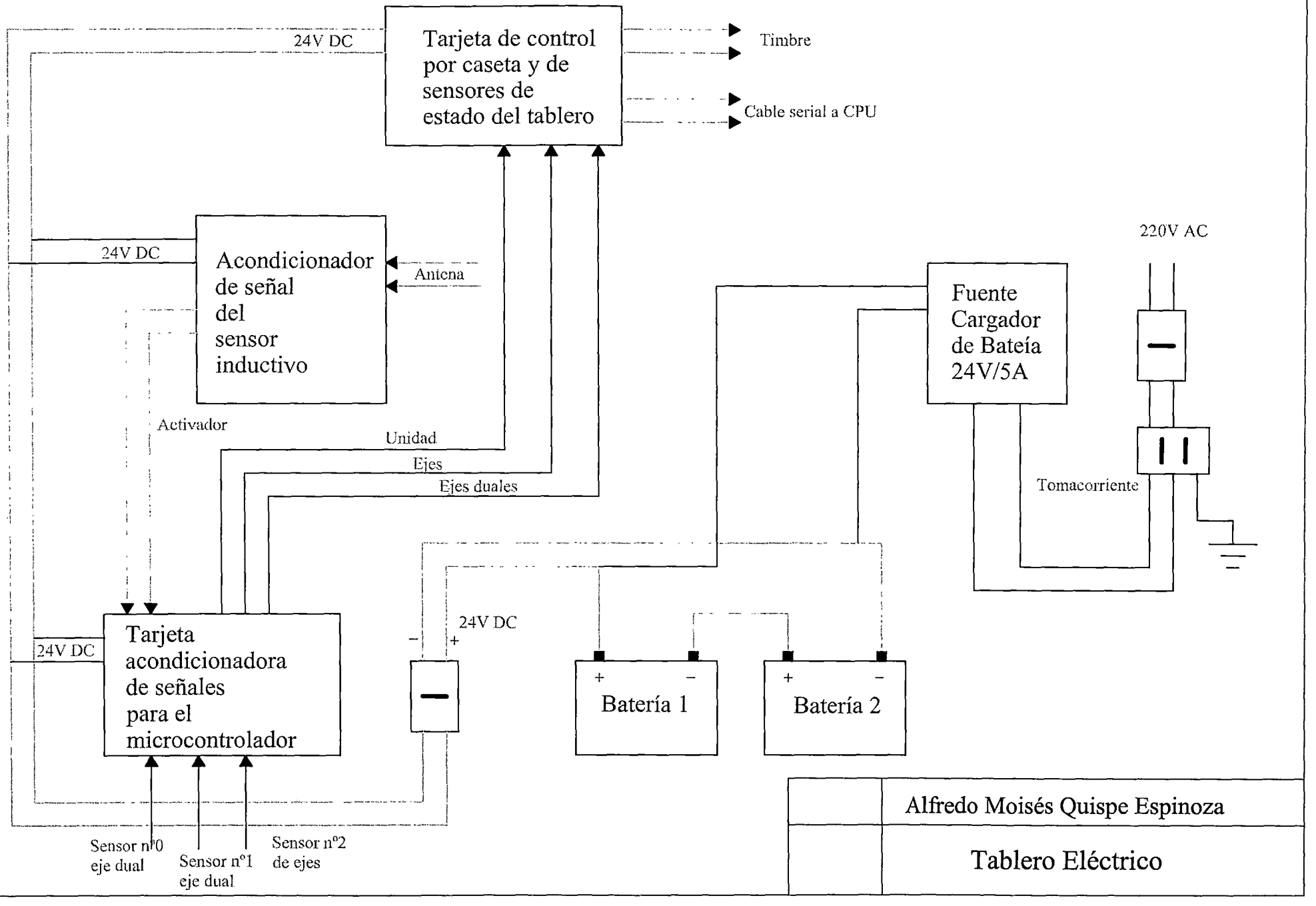

- *Labview 7.1. Programación Gráfica Para el Control de Instrumentos.* Antonio Manuel, Lázaro. Joaquín, del Río Fernández. Madrid (España). 2005.
- *Entorno Gráfico de Programación Labview 8.2 y versiones anteriores.* Lajara Vizcaino, José Rafael Pelegrí Sebastia, José. México. 2007.
- *Microcontroladores Pic 2°Parte. PIC16F87X. Diseño Práctico de Aplicaciones.* Angulo Usategui, José María Romero Yessa, Susana Angulo Martínez, Ignacio. Madrid (España). 2006.
- *Ingeniería de Control Moderna 4° Edición.* Katsuhiko Ogata. 2003.
- *Sistema de Control Moderno 10° Edición.* Dorf Richard G., Bishop Robert H. 2005.
- *Sensores y Acondicionadores de Señal 4° Edición.* Ramón Pallás Arency. España. 2005.

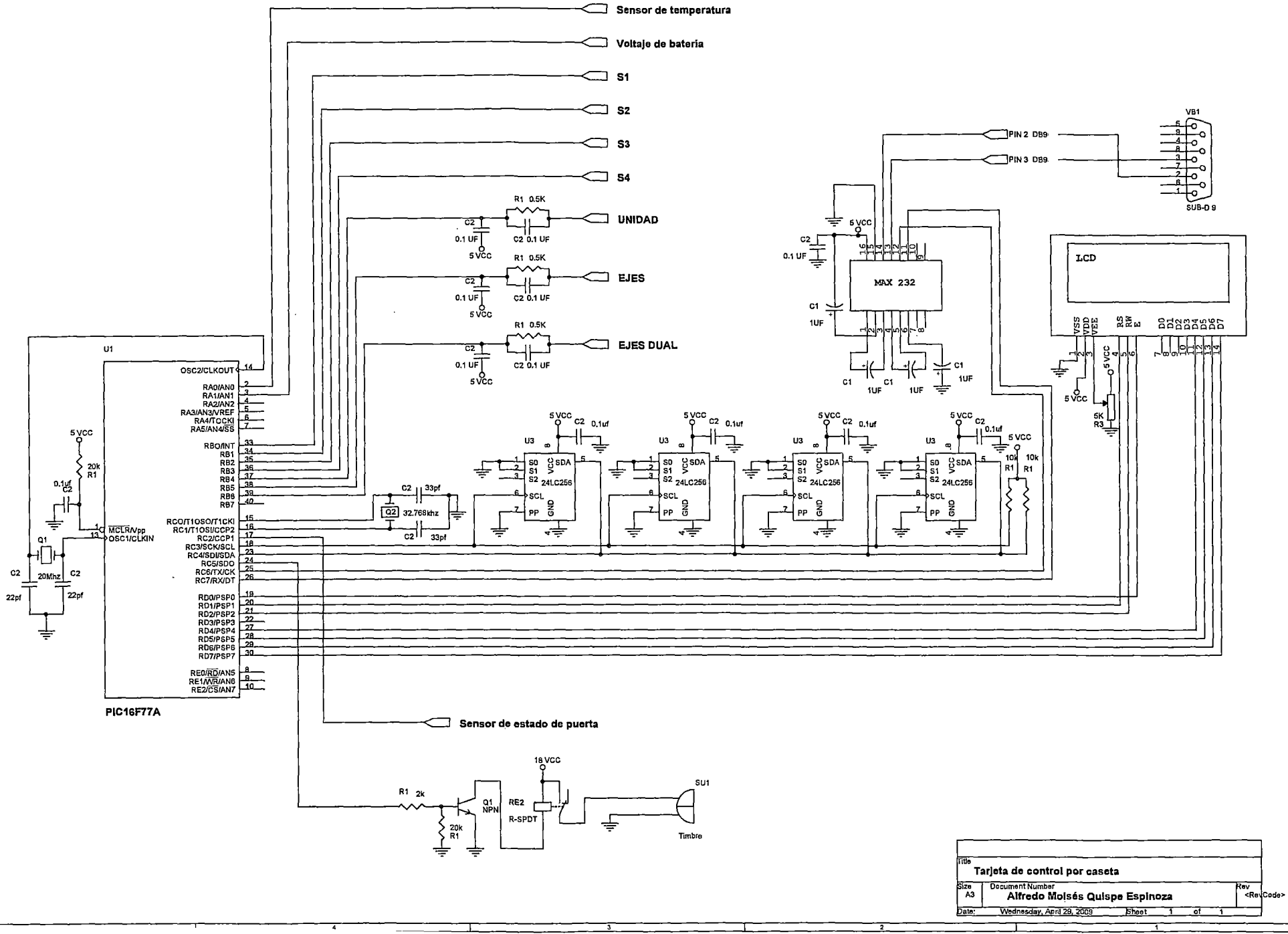

***Páginas web:***

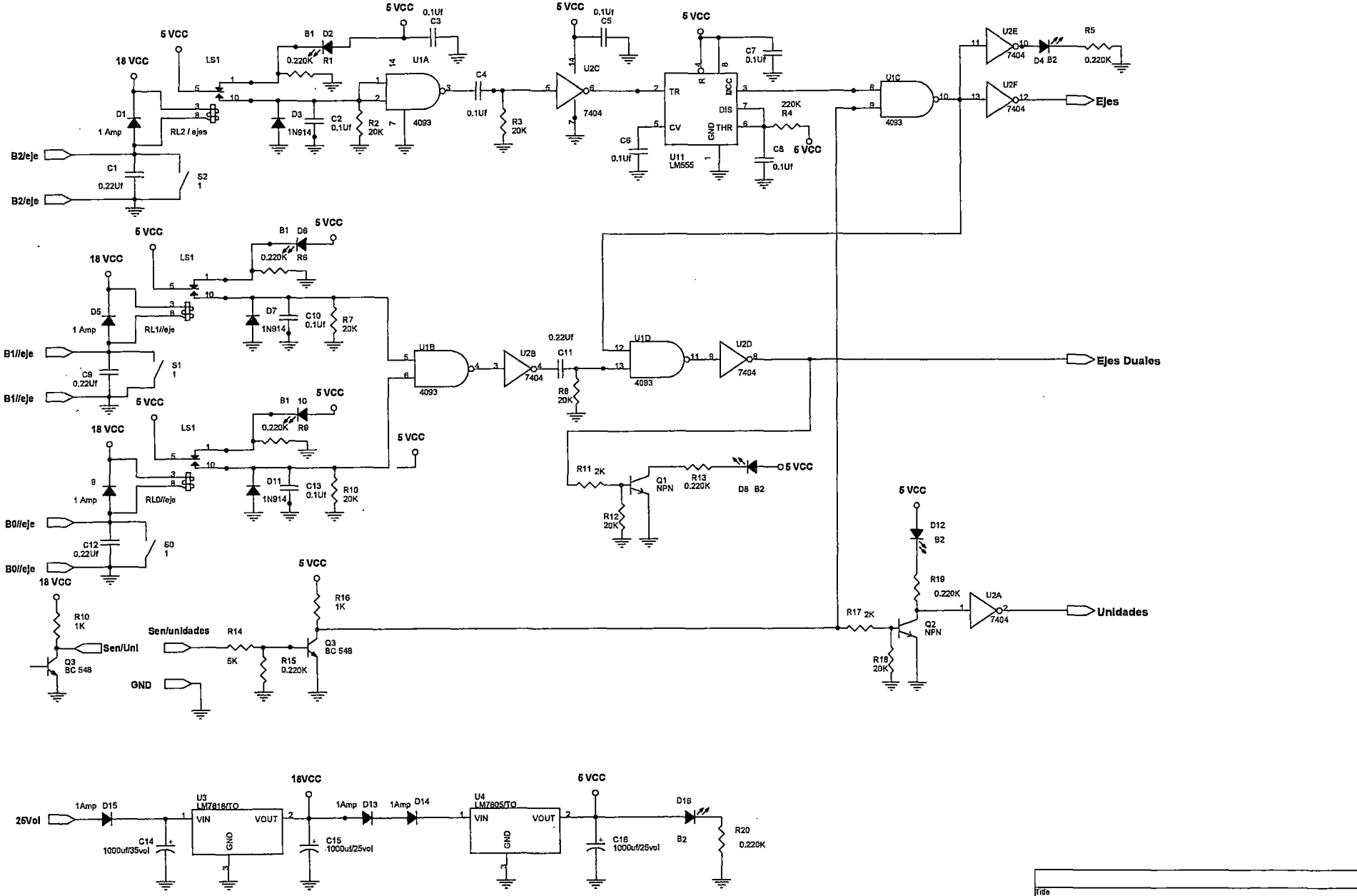

www.microchip.com

www.ni.com/latam

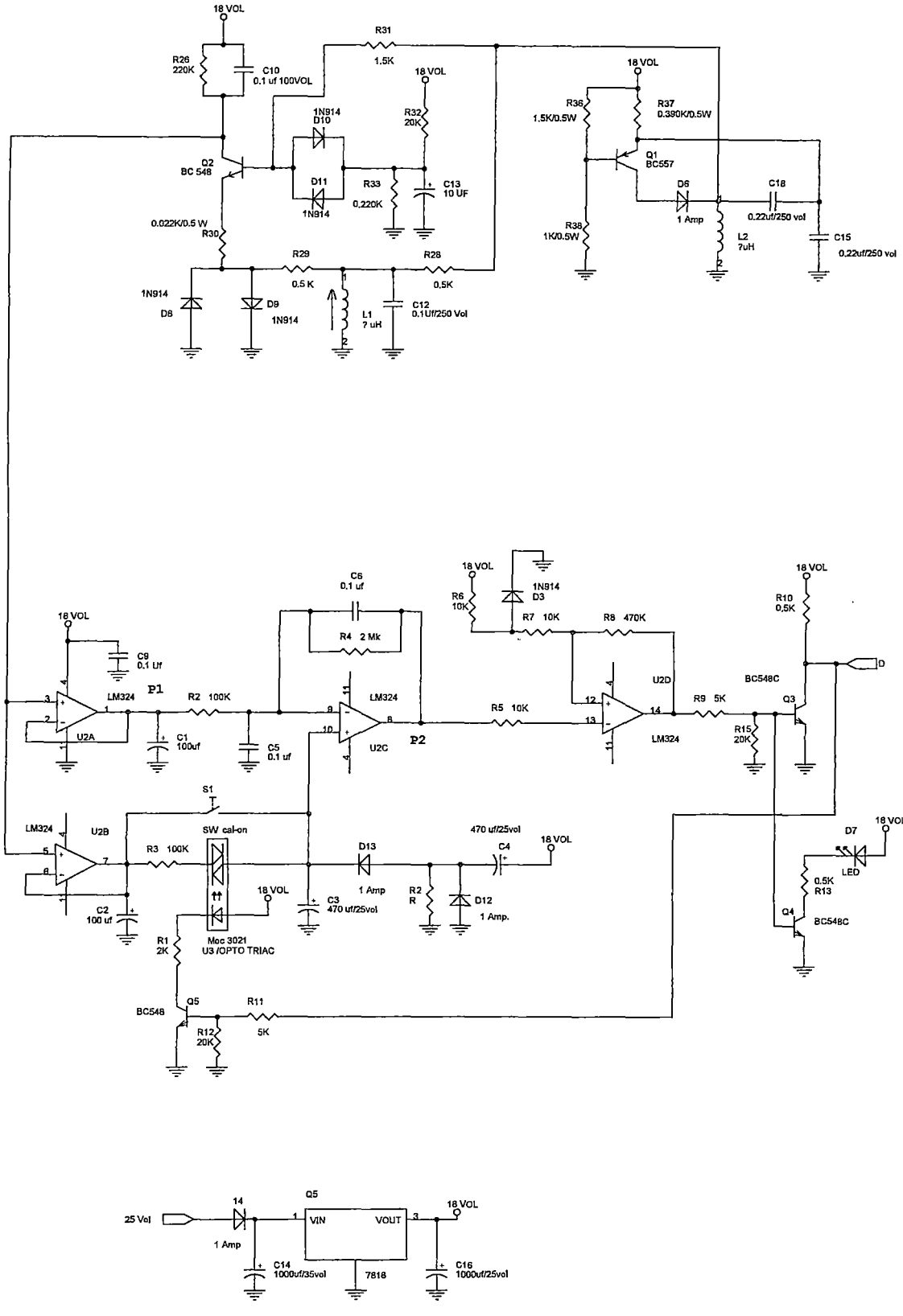
# PLANOS


Tarjeta de control por caseta y de sensores de estado del tablero
24V DC
Timbre
Cable serial a CPU
Acondicionador de señal del sensor inductivo
24V DC
Antena
220V AC
Fuente Cargador de Bateía 24V/5A
Activador
Unidad
Ejes
Ejes duales
Tomacorriente
Tarjeta acondicionadora de señales para el microcontrolador
24V DC
24V DC
Batería 1
Batería 2
Sensor nº0 eje dual
Sensor nº1 eje dual
Sensor nº2 de ejes
Alfredo Moisés Quispe Espinoza
Tablero Eléctrico

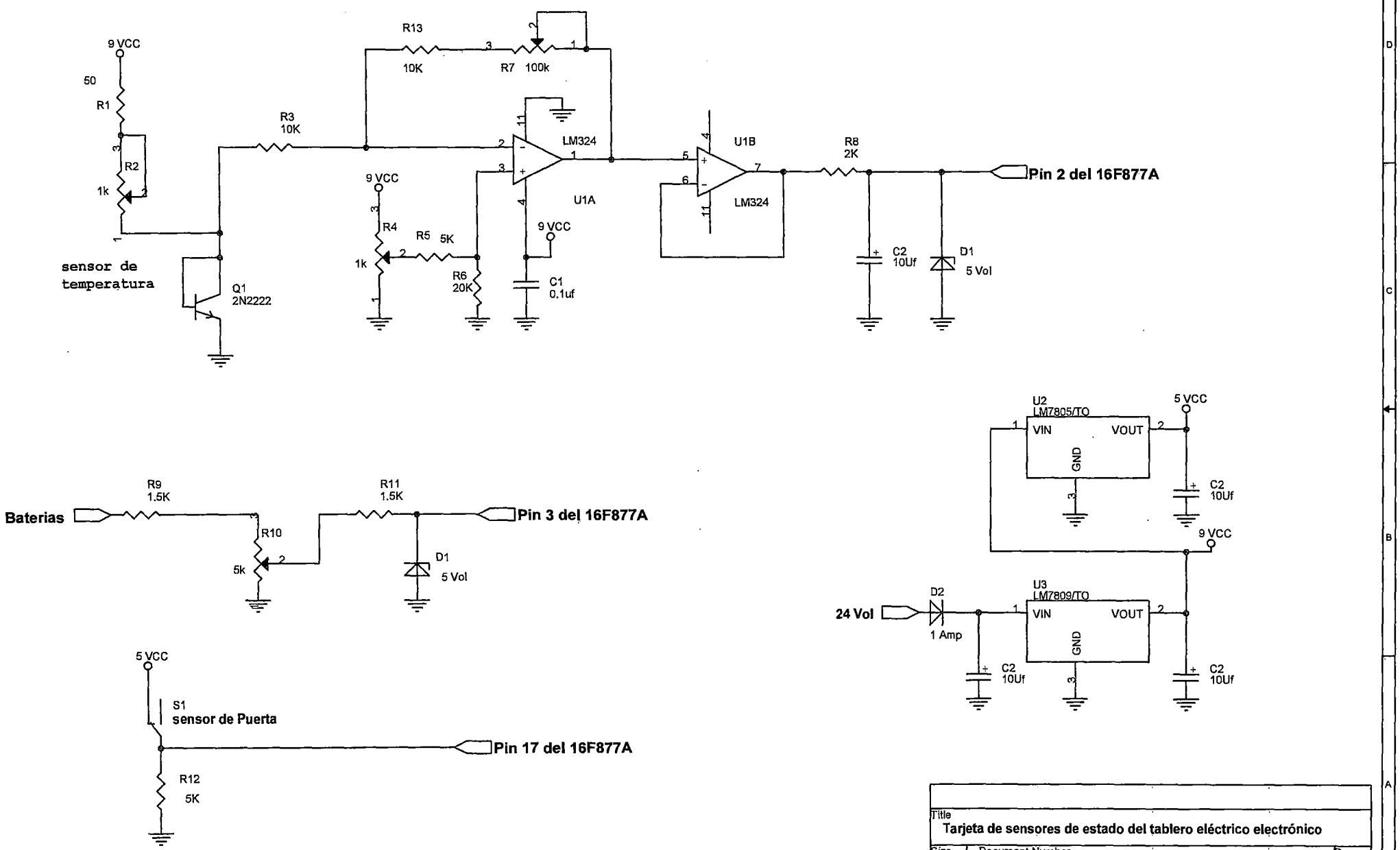

Sensor de temperatura
Voltaje de bateria
S1
S2
S3
S4
UNIDAD
EJES
EJES DUAL
R1 0.5K
C2 0.1 UF
5 VCC
U1
OSC2/CLKOUT
RA0/AN0
RA1/AN1
RA2/AN2
RA3/AN3/VREF
RA4/T0CKI
RA5/AN4/SS
RB0/INT
RB1
RB2
RB3
RB4
RB5
RB6
RB7
RC0/T1OSO/T1CKI
RC1/T1OSI/CCP2
RC2/CCP1
RC3/SCK/SCL
RC4/SDI/SDA
RC5/SDO
RC6/TX/CK
RC7/RX/DT
RD0/PSP0
RD1/PSP1
RD2/PSP2
RD3/PSP3
RD4/PSP4
RD5/PSP5
RD6/PSP6
RD7/PSP7
RE0/RD/AN5
RE1/WR/AN6
RE2/CS/AN7
MCLR/Vpp
OSC1/CLKIN
20k R1
0.1uf C2
Q1
20Mhz
22pf
Q2 32.768khz
33pf
PIC16F77A
U3
24LC256
SDA
SCL
VCC
GND
10k R1
MAX 232
1UF
VB1
PIN 2 DB9
PIN 3 DB9
SUB-D 9
LCD
VSS
VDD
VBE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
5K R3
Sensor de estado de puerta
18 VCC
R1 2k
Q1 NPN
RE2
R-SPDT
20k R1
SU1
Timbre
Title
Tarjeta de control por caseta
Size A3
Document Number
Alfredo Moisés Quispe Espinoza
Rev <RevCode>
Date: Wednesday, April 29, 2009
Sheet 1 of 1




| | |
|---|---|
| Title | Acondicionador de señales para el microcontrolador |
| Size | |
| Document Number | Alfredo Moisés Quispe Espinoza |
| Rev | 01 |
| Date: | Wednesday, April 29, 2009 |
| Sheet | 1 of 1 |



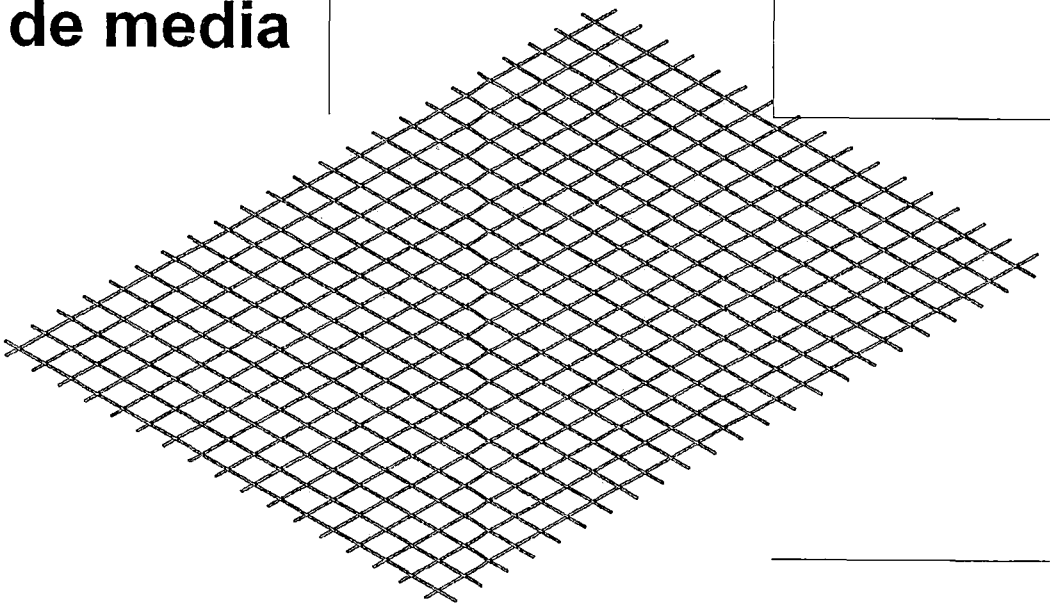

| | | |
|---|---|---|
| Title | Acondicionador de señal del sensor inductivo | |
| Size | Document Number: Alfredo Moisés Quispe Espinoza | Rev: 02 |
| Date: Wednesday, April 29, 2009 | Sheet 1 of 1 | |

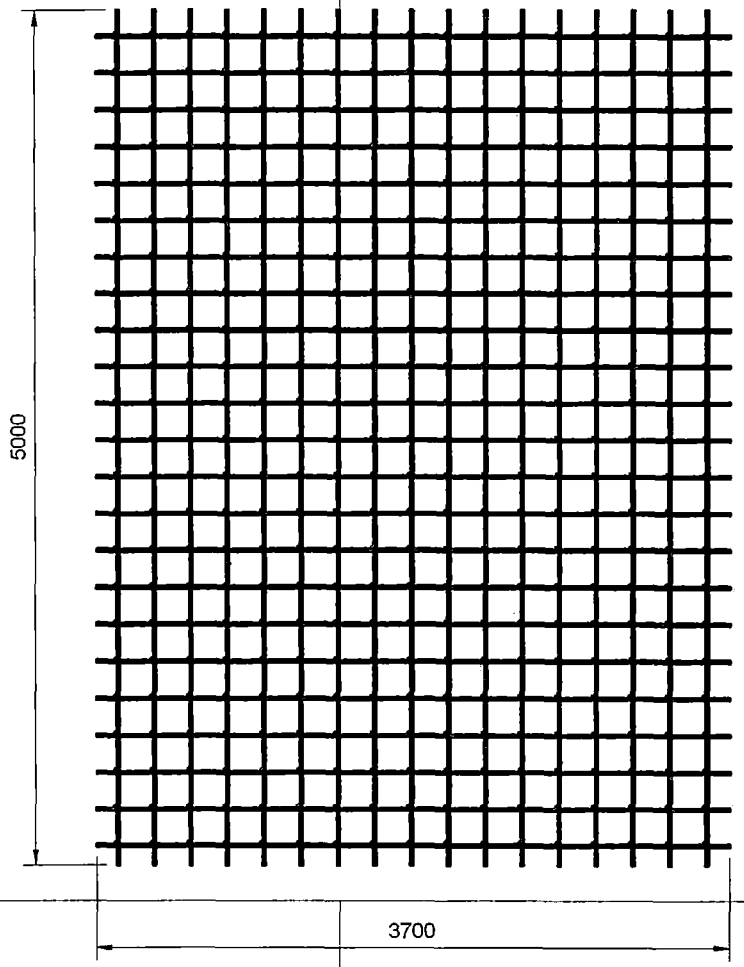



| | | |
|---|---|---|
| Title: Tarjeta de sensores de estado del tablero eléctrico electrónico | | |
| Size: A4 | Document Number: Alfredo Moisés Quispe Espinoza | Rev: 03 |
| Date: Wednesday, April 29, 2009 | Sheet 1 of 1 | |

17-vertical
23-horizontal

# malla-fierro de media



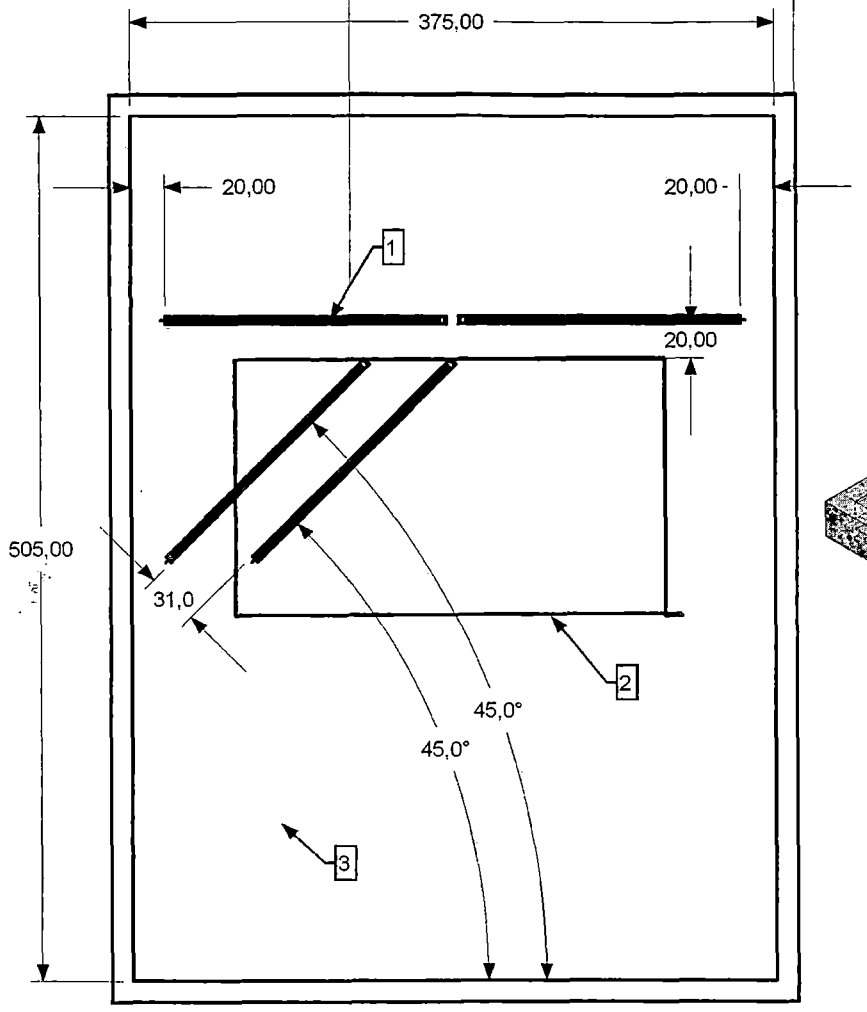



cuadrado
20x20

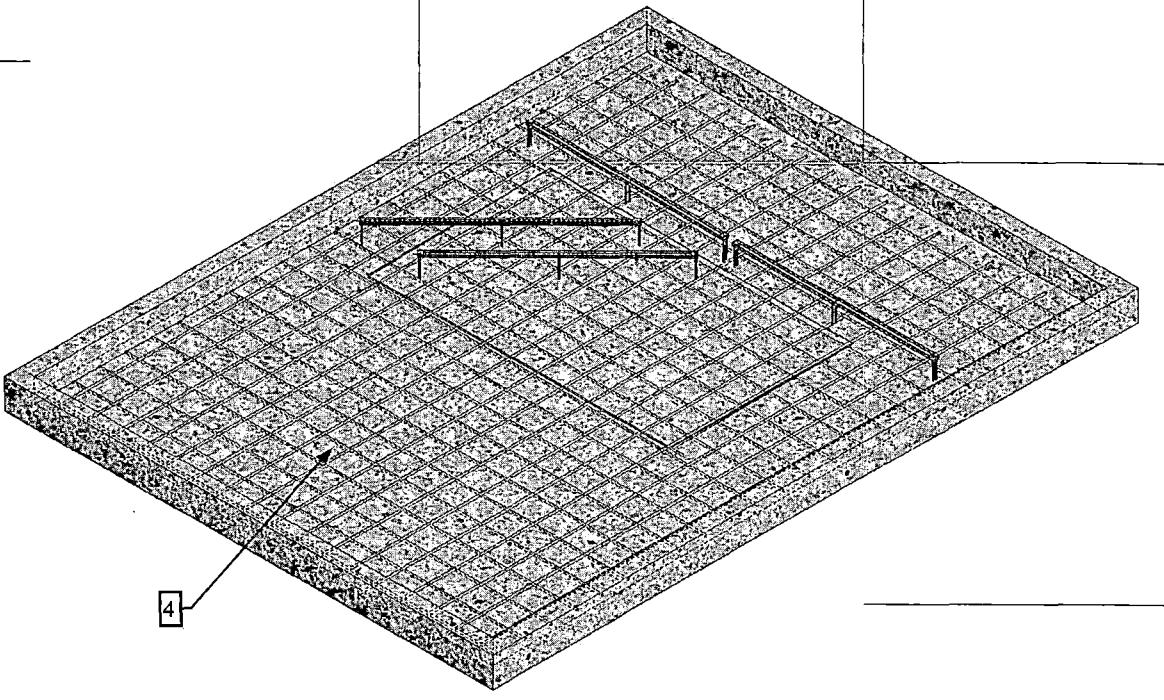
Nota: Se usará fierro corrugado de media

| | | | | |
|---|---|---|---|---|
| E | A3 | General Tolerances acc to : ISO 2768-1 medium | unid: milimetros | Doc.Nr |
| | Scale | | | |
| Alfredo Moisés Quispe Espinoza | | | tema: Malla de fierro | |

| RevNr. | Revision note | Date | Sign |
|---|---|---|---|




**Vista isométrica de los sensores en la loza**




Nota:El sensor de loop estará dentro de tubo PVC a 2,54cm de la superficie de la pista.

| Ite | Nombre | Cant | Descripción |
|---|---|---|---|
| 1 | Sensor de presión | 4 | Detecta número de ejes |
| 2 | Sensor de loop | 1 | 2 vueltas de cable Nº16 de 2,5m x 1,5m |
| 3 | Loza de concreto | 1 | Concreto de 280kgf/cm^2 |
| 4 | Malla de fierro | 1 | Fierro corrugado de media |

| | | | | | |
|---|---|---|---|---|---|
| E | A3 | General Tolerances acc to : | | | |
| | Scale | ISO 2768-1 medium | Unidades: cm | Checked Checked | Dat. Dat |
| *Alfredo Moisés Quispe Espinoza* | tema: **Disposición de sensores** | | | Sheet 1 | Rev. |
| | | | | Doc.Nr 1000 0000 0000 | |

# ANEXOS

# ANEXO A.
# INFORMACIÓN SOBRE DISEÑO DE SENSORES INDUCTIVOS DE LA EMPRESA RENO A&E

To ensure a long, trouble-free test loop life, it is very important to keep the wire wraps from becoming loose and possibly crossing each other. Clear nail polish can be used to achieve this result. Simply coat the wire wraps with a liberal amount of clear nail polish and allow it to dry. An alternative method of ensuring that the loop wire wraps remain in the proper position is to firmly wrap the wire with electrical tape. Whichever method is used, make sure that the wire wraps are not disturbed during the process of securing them.

The ends of the test loop wires can be terminated in one of two ways. The easiest method of termination is to drill two small holes near the bottom of the coupling (below the loop wraps) and feed each end of the loop wire through either hole (see Figure 3). Trim the loop wire ends to a convenient length, strip back a short length of insulation to allow connection to the test loop wire ends, and tin the exposed wire ends with solder. A second method of termination is to use Banana jacks. Once again, drill two appropriately sized holes near the bottom of the coupling (below the loop wraps), mount the Banana jacks, and terminate each end of the loop wire at either Banana jack (see Figure 4).

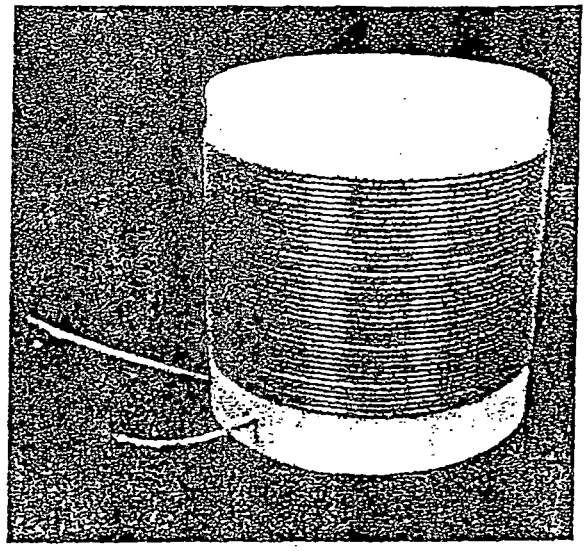


Figure 3 – Bare End Loop Wire Termination

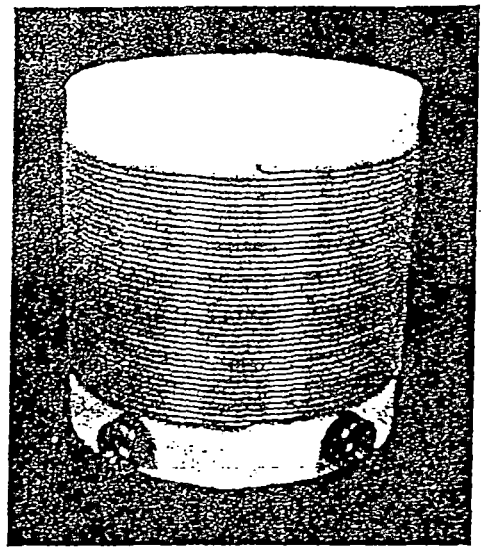


Figure 4 – Banana Jack Loop Wire Termination

The actual inductance of the test loop depends on the number of wraps (turns) of wire wound when constructing the loop. The following table lists approximate inductance values for various numbers of turns of wire wound around a 2" Schedule 40 PVC Coupling.

## Approximate Test Loop Inductance

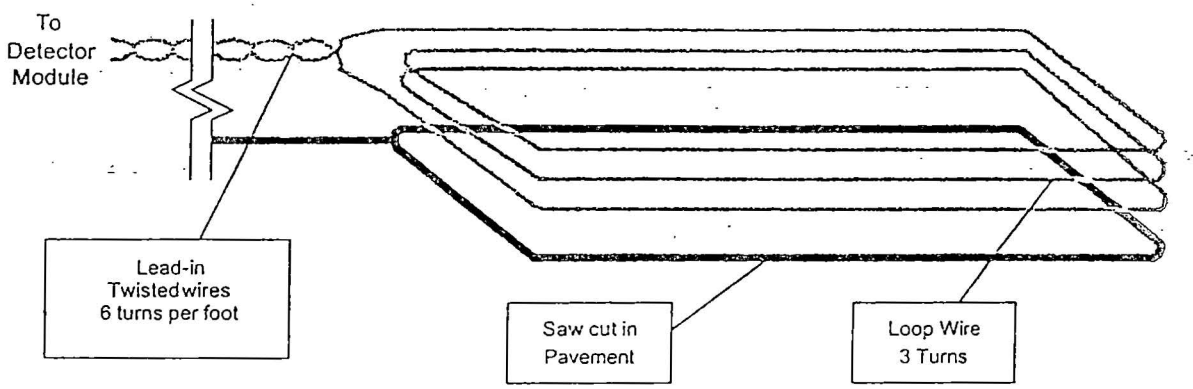
| Number of Turns | Test Loop Inductance |
|---|---|
| 40 | 100 µH |
| 38 | 95 µH |
| 37 | 90 µH |
| 35 | 85 µH |
| 34 | 80 µH |
| 32 | 75 µH |
| 30 | 70 µH |
| 29 | 65 µH |
| 27 | 60 µH |
| 25 | 55 µH |
| 23 | 50 µH |
| 21 | 45 µH |
| 19 | 40 µH |
| 16 | 35 µH |
| 14 | 30 µH |
| 12 | 25 µH |
| 9 | 20 µH |

Reno A & E
4655 Aircenter Circle
Reno, NV 89502-5948 USA
Telephone: (775) 826-2020
Fax: (775) 826-9191
Website: www.renoae.com
E-mail: contact@renoae.com

*Engineering Excellence!*

## What is an inductive loop?

An inductive loop is a wire wound in a rectangular, square, or round shape that is typically sawcut into the pavement. The ends of the wire are brought back to an enclosure, which houses an inductive loop detector module. The detector module powers the loop and causes a field to form around the loop. The loop automatically tunes to a resonant frequency. The detector module monitors this resonant frequency to determine if a vehicle is in the loop area.




## What is inductance?

Inductance is defined as the opposition to a change in current flow. When a current is applied to a conductor such as a wire, a magnetic field is formed around the wire. If the current source is removed, the magnetic field collapses into the wire trying to maintain the current flow. By winding several turns of the wire into a coil, the magnetic field is intensified, which increases the inductance.

## How is the vehicle detected?

When a vehicle enters or crosses the loop, the body and frame provide a conductive path for the magnetic field. This produces a loading effect, which in turn causes the loop inductance to decrease. The decreased inductance causes the resonant frequency to increase from its nominal value. If the frequency change exceeds the threshold set by the sensitivity setting, the detector module will output a detect signal.

There is a common misconception that an inductive loop requires a mass of metal or ferrous material for detection. Placing a single wire around the perimeter of the loop and shorting the ends together will quickly disprove this misconception. The single wire forming a shorted turn provides a current path for the magnetic field; thus causing a loading effect similar to that of a vehicle. The shorted turn effect of the single wire coil in the proximity of the loop acts much like a shorted turn secondary of a transformer.

## What is the minimum acceptable loop inductance?

An inductive loop detector will tune to inductance values ranging from 20 to 1000 microhenries. It is preferable that the combination of the loop and lead-in inductance values has a minimum of approximately 50 microhenries for stability. As a general rule, the loop inductance should be equal to or greater than the lead-in inductance.

## How many turns of wire should be installed in the loop?

The number of turns required in the loop is dependent on the loop size. Loop inductance can be calculated as follows:

$L = P/4\ (t^2 + t)$; Where: L = Inductance (microhenries)
P = Perimeter (feet)
t = Number of turns.

The formula can be simplified to: L = PK by substituting a constant K for $(t^2 + t)/4$.

Filling in the Number of Turns and calculating K:

| Number Of Turns (t) | K (constant) $K=(t^2+t)/4$ |
|---|---|
| 1 | 0.5 |
| 2 | 1.5 |
| 3 | 3.0 |
| 4 | 5.0 |
| 5 | 7.5 |
| 6 | 10.5 |
| 7 | 14.0 |

Example: 4' x 8' loop with 4 turns
L = P K
P = 4' + 4' + 8' + 8' = 24'
K = 5.0
L = 24 x 5.0
L = 120 microhenries

## Loop inductance in microhenries (? H)

| PERIMETER | NUMBER OF TURNS | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 5 | 15 | 30 | 50 | 75 | 115 | 140 |
| 20 | 10 | 30 | 60 | 100 | 150 | 230 | 280 |
| 30 | 15 | 45 | 90 | 150 | 225 | 345 | 420 |
| 40 | 20 | 60 | 120 | 200 | 300 | 460 | 560 |
| 50 | 25 | 75 | 150 | 250 | 375 | 575 | 700 |
| 60 | 30 | 90 | 180 | 300 | 450 | 690 | 840 |
| 70 | 35 | 105 | 210 | 350 | 525 | 805 | 980 |
| 80 | 40 | 120 | 240 | 400 | 600 | 920 | 1120 |
| 90 | 45 | 135 | 270 | 450 | 675 | 1035 | 1260 |
| 100 | 50 | 150 | 300 | 500 | 750 | 1150 | 1400 |

## Recommended number of turns

| PERIMETER | NUMBER OF TURNS |
|---|---|
| 10 | 5 |
| 20 | 4 |
| 30 | 3 |
| 40 | 3 |
| 50 | 2 |
| 60 | 2 |
| 70 | 2 |
| 80 | 2 |
| 90 | 2 |
| 100 | 2 |

Use the values listed in the table above to determine the number of turns required for a given size loop. Always use at least 2 turns.

## What type of wire should be used for the loop?

Number 16 or 20 AWG stranded wire can be used. The wire gauge is not critical to proper operation of the loop detector. The wire should maintain its integrity under the pavement stress. Since asphalt is more flexible than concrete, it is recommended that a heavier gauge wire be used for loop installations in asphalt.

The main consideration in selecting a wire for loop installations is the type of **insulation**. Cross-linked polyethylene (XLPE) insulation rated at 600 volts is highly recommended over PVC insulation. Under similar conditions, XLPE insulation will absorb approximately one percent of the moisture absorbed by PVC. When insulation absorbs moisture, loop drift occurs, which if great enough, can cause false detections. XLPE also has higher resistance to abrasion, heat, oils, and gasoline.

After installation, and any time there appears to be a loop related problem, the loop should be tested. Use a MegOhm Meter to test the integrity of the loop / lead-in wire insulation. Readings of 100MO or greater indicate that the insulation is intact and undamaged. Readings of 50MO or less indicate possible insulation damage. Use a Multimeter to check the total resistance of the loop / lead-in combination. Total loop / lead-in resistance should **never** exceed 4 Ohms.

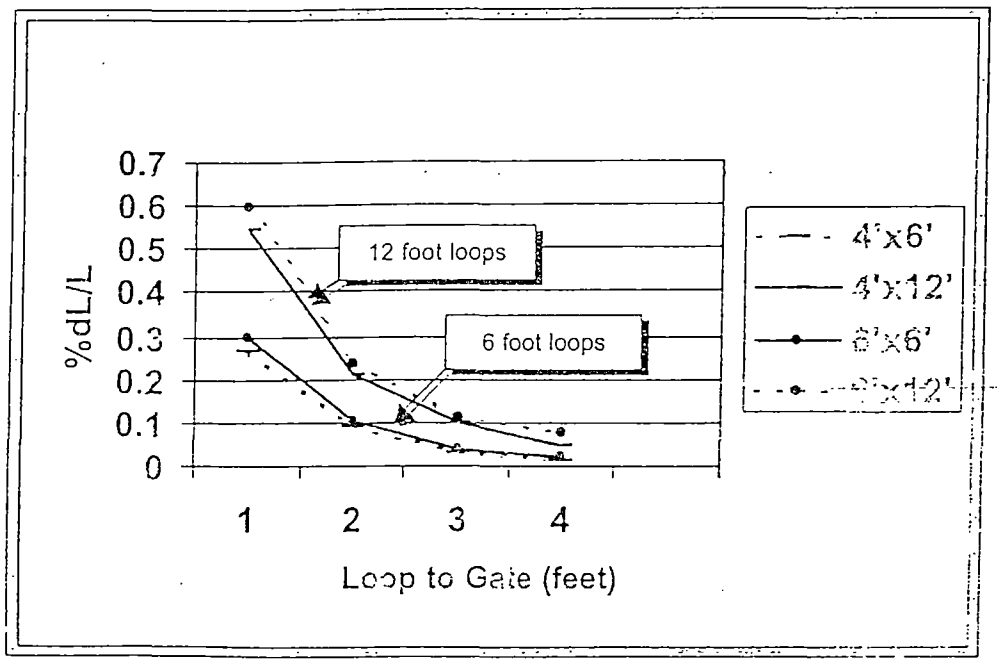
## How far from a gate should the loop be installed?

As the length of the sides of the loop that parallel the gate increases, the inductance change caused by the gate also increases. The graph shows the inductance change for different distances between the gate and the loop for different sized loops.

The closer the loop is to a gate, the more influence the gate has on the loop! Hence, the detector sensitivity must be set lower to ensure the gate will not cause the detector to generate an output when the gate closes.

The following rule should be observed: **The longer the loop, the greater the spacing must be between the gate and the loop!**

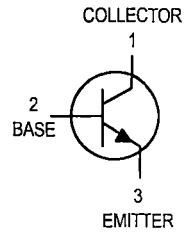
The inductance change at two feet is one third of the change at one foot. At four feet, the effects of the gate on the loop are minimal.

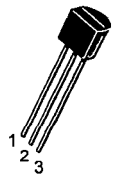


# ANEXO B.
# HOJA TÉCNICA DEL TRANSISTOR 2N2222 USADO PARA SENSOR DE TEMPERATURA

# Amplifier Transistors
## NPN Silicon

**P2N2222A**




**CASE 29–04, STYLE 17**
**TO–92 (TO–226AA)**

**MAXIMUM RATINGS**

| Rating | Symbol | Value | Unit |
|---|---|---|---|
| Collector–Emitter Voltage | $V_{CEO}$ | 40 | Vdc |
| Collector–Base Voltage | $V_{CBO}$ | 75 | Vdc |
| Emitter–Base Voltage | $V_{EBO}$ | 6.0 | Vdc |
| Collector Current — Continuous | $I_C$ | 600 | mAdc |
| Total Device Dissipation @ $T_A$ = 25°C<br>Derate above 25°C | $P_D$ | 625<br>5.0 | mW<br>mW/°C |
| Total Device Dissipation @ $T_C$ = 25°C<br>Derate above 25°C | $P_D$ | 1.5<br>12 | Watts<br>mW/°C |
| Operating and Storage Junction Temperature Range | $T_J$, $T_{stg}$ | −55 to +150 | °C |

**THERMAL CHARACTERISTICS**

| Characteristic | Symbol | Max | Unit |
|---|---|---|---|
| Thermal Resistance, Junction to Ambient | $R_{\theta JA}$ | 200 | °C/W |
| Thermal Resistance, Junction to Case | $R_{\theta JC}$ | 83.3 | °C/W |

**ELECTRICAL CHARACTERISTICS** ($T_A$ = 25°C unless otherwise noted)

| Characteristic | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| **OFF CHARACTERISTICS** | | | | |
| Collector–Emitter Breakdown Voltage ($I_C$ = 10 mAdc, $I_B$ = 0) | $V_{(BR)CEO}$ | 40 | — | Vdc |
| Collector–Base Breakdown Voltage ($I_C$ = 10 µAdc, $I_E$ = 0) | $V_{(BR)CBO}$ | 75 | — | Vdc |
| Emitter–Base Breakdown Voltage ($I_E$ = 10 µAdc, $I_C$ = 0) | $V_{(BR)EBO}$ | 6.0 | — | Vdc |
| Collector Cutoff Current ($V_{CE}$ = 60 Vdc, $V_{EB(off)}$ = 3.0 Vdc) | $I_{CEX}$ | — | 10 | nAdc |
| Collector Cutoff Current<br>($V_{CB}$ = 60 Vdc, $I_E$ = 0)<br>($V_{CB}$ = 60 Vdc, $I_E$ = 0, $T_A$ = 150°C) | $I_{CBO}$ | <br>—<br>— | <br>0.01<br>10 | µAdc |
| Emitter Cutoff Current ($V_{EB}$ = 3.0 Vdc, $I_C$ = 0) | $I_{EBO}$ | — | 10 | nAdc |
| Collector Cutoff Current ($V_{CE}$ = 10 V) | $I_{CEO}$ | — | 10 | nAdc |
| Base Cutoff Current ($V_{CE}$ = 60 Vdc, $V_{EB(off)}$ = 3.0 Vdc) | $I_{BEX}$ | — | 20 | nAdc |

**ELECTRICAL CHARACTERISTICS** ($T_A$ = 25°C unless otherwise noted) (Continued)

| Characteristic | Symbol | Min | Max | Unit |
|---|---|---|---|---|
| **ON CHARACTERISTICS** | | | | |
| DC Current Gain | $h_{FE}$ | | | — |
| ($I_C$ = 0.1 mAdc, $V_{CE}$ = 10 Vdc) | | 35 | — | |
| ($I_C$ = 1.0 mAdc, $V_{CE}$ = 10 Vdc) | | 50 | — | |
| ($I_C$ = 10 mAdc, $V_{CE}$ = 10 Vdc) | | 75 | — | |
| ($I_C$ = 10 mAdc, $V_{CE}$ = 10 Vdc, $T_A$ = –55°C) | | 35 | — | |
| ($I_C$ = 150 mAdc, $V_{CE}$ = 10 Vdc)(1) | | 100 | 300 | |
| ($I_C$ = 150 mAdc, $V_{CE}$ = 1.0 Vdc)(1) | | 50 | — | |
| ($I_C$ = 500 mAdc, $V_{CE}$ = 10 Vdc)(1) | | 40 | — | |
| Collector–Emitter Saturation Voltage(1) | $V_{CE(sat)}$ | | | Vdc |
| ($I_C$ = 150 mAdc, $I_B$ = 15 mAdc) | | — | 0.3 | |
| ($I_C$ = 500 mAdc, $I_B$ = 50 mAdc) | | — | 1.0 | |
| Base–Emitter Saturation Voltage(1) | $V_{BE(sat)}$ | | | Vdc |
| ($I_C$ = 150 mAdc, $I_B$ = 15 mAdc) | | 0.6 | 1.2 | |
| ($I_C$ = 500 mAdc, $I_B$ = 50 mAdc) | | — | 2.0 | |
| **SMALL–SIGNAL CHARACTERISTICS** | | | | |
| Current–Gain — Bandwidth Product(2) ($I_C$ = 20 mAdc, $V_{CE}$ = 20 Vdc, f = 100 MHz) | $f_T$ | 300 | — | MHz |
| Output Capacitance ($V_{CB}$ = 10 Vdc, $I_E$ = 0, f = 1.0 MHz) | $C_{obo}$ | — | 8.0 | pF |
| Input Capacitance ($V_{EB}$ = 0.5 Vdc, $I_C$ = 0, f = 1.0 MHz) | $C_{ibo}$ | — | 25 | pF |
| Input Impedance | $h_{ie}$ | | | kΩ |
| ($I_C$ = 1.0 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | 2.0 | 8.0 | |
| ($I_C$ = 10 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | 0.25 | 1.25 | |
| Voltage Feedback Ratio | $h_{re}$ | | | X $10^{-4}$ |
| ($I_C$ = 1.0 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | — | 8.0 | |
| ($I_C$ = 10 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | — | 4.0 | |
| Small–Signal Current Gain | $h_{fe}$ | | | — |
| ($I_C$ = 1.0 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | 50 | 300 | |
| ($I_C$ = 10 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | 75 | 375 | |
| Output Admittance | $h_{oe}$ | | | μmhos |
| ($I_C$ = 1.0 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | 5.0 | 35 | |
| ($I_C$ = 10 mAdc, $V_{CE}$ = 10 Vdc, f = 1.0 kHz) | | 25 | 200 | |
| Collector Base Time Constant ($I_E$ = 20 mAdc, $V_{CB}$ = 20 Vdc, f = 31.8 MHz) | $rb'C_c$ | — | 150 | ps |
| Noise Figure ($I_C$ = 100 μAdc, $V_{CE}$ = 10 Vdc, $R_S$ = 1.0 kΩ, f = 1.0 kHz) | $N_F$ | — | 4.0 | dB |

**SWITCHING CHARACTERISTICS**

| | | | | | |
|---|---|---|---|---|---|
| Delay Time | ($V_{CC}$ = 30 Vdc, $V_{BE(off)}$ = –2.0 Vdc, $I_C$ = 150 mAdc, $I_{B1}$ = 15 mAdc) (Figure 1) | $t_d$ | — | 10 | ns |
| Rise Time | | $t_r$ | — | 25 | ns |
| Storage Time | ($V_{CC}$ = 30 Vdc, $I_C$ = 150 mAdc, $I_{B1}$ = $I_{B2}$ = 15 mAdc) (Figure 2) | $t_s$ | — | 225 | ns |
| Fall Time | | $t_f$ | — | 60 | ns |

1. Pulse Test: Pulse Width ≤ 300 μs, Duty Cycle ≤ 2.0%.
2. $f_T$ is defined as the frequency at which $|h_{fe}|$ extrapolates to unity.

## SWITCHING TIME EQUIVALENT TEST CIRCUITS

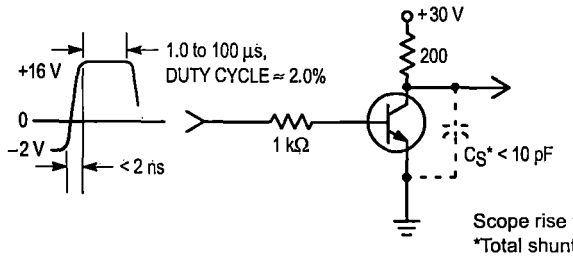



Figure 1. Turn–On Time

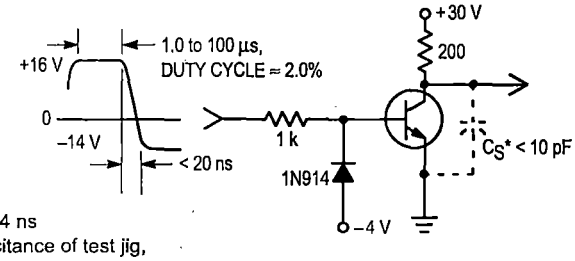



Figure 2. Turn–Off Time

Scope rise time < 4 ns
*Total shunt capacitance of test jig, connectors, and oscilloscope.

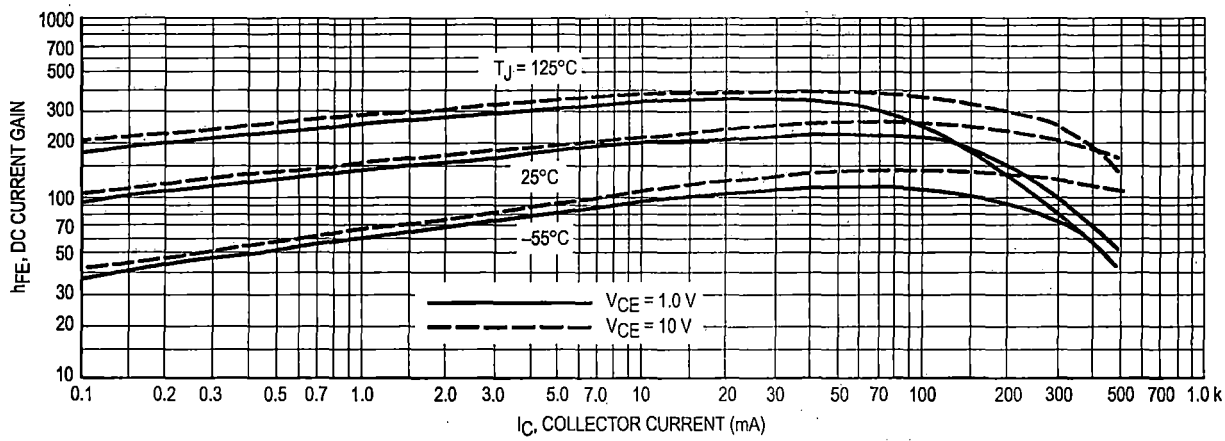



Figure 3. DC Current Gain

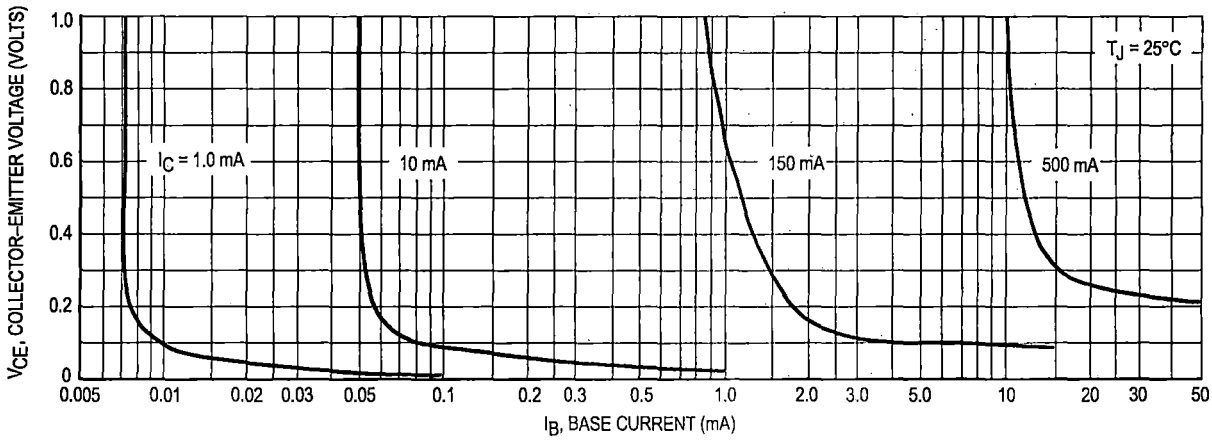



Figure 4. Collector Saturation Region

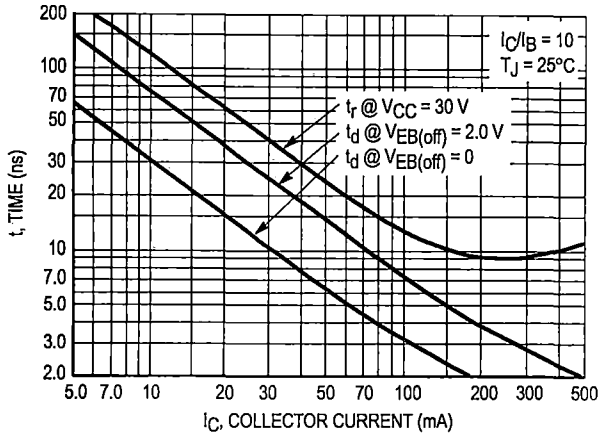



**Figure 5. Turn–On Time**

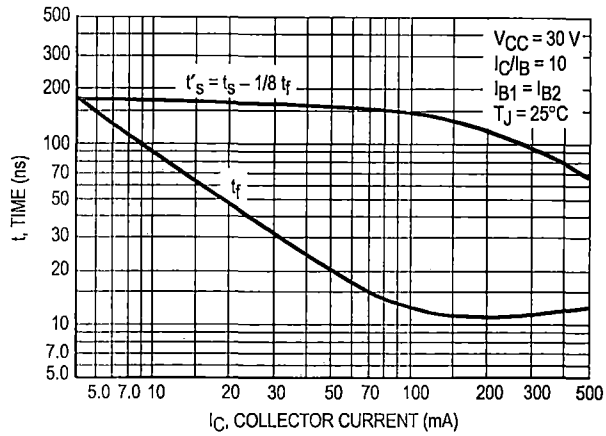



**Figure 6. Turn–Off Time**

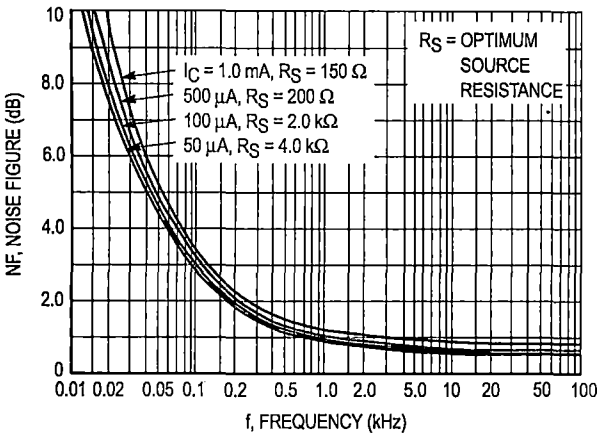



**Figure 7. Frequency Effects**

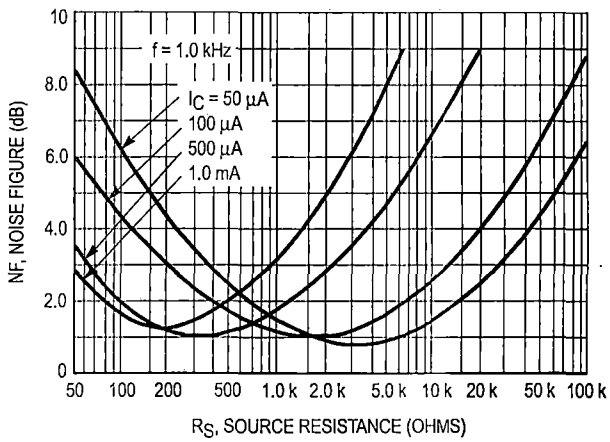



**Figure 8. Source Resistance Effects**

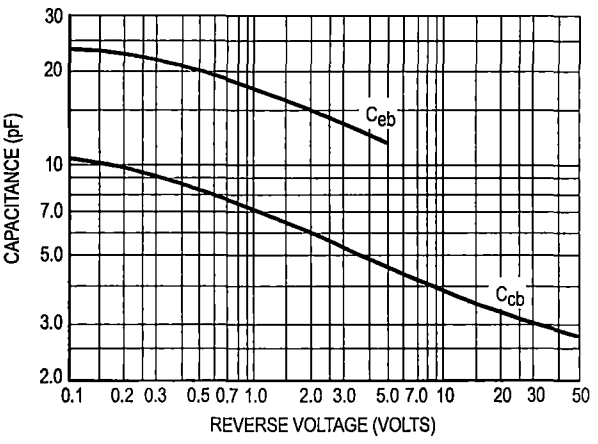



**Figure 9. Capacitances**

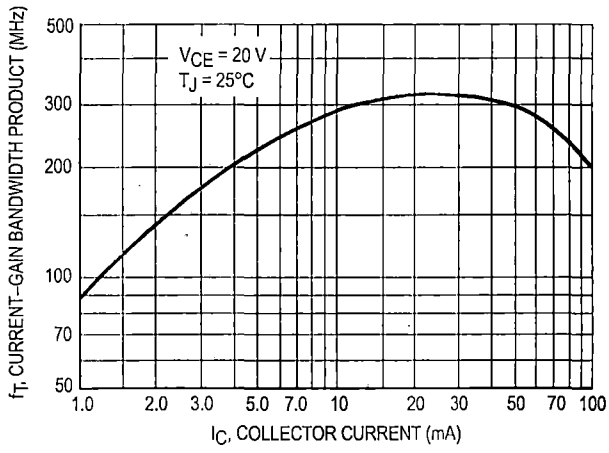



**Figure 10. Current–Gain Bandwidth Product**

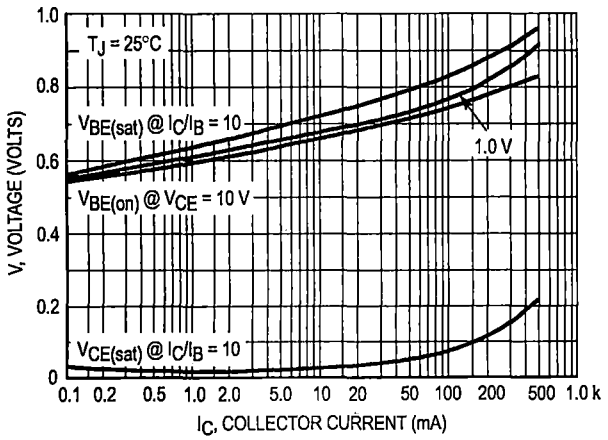



**Figure 11. "On" Voltages**

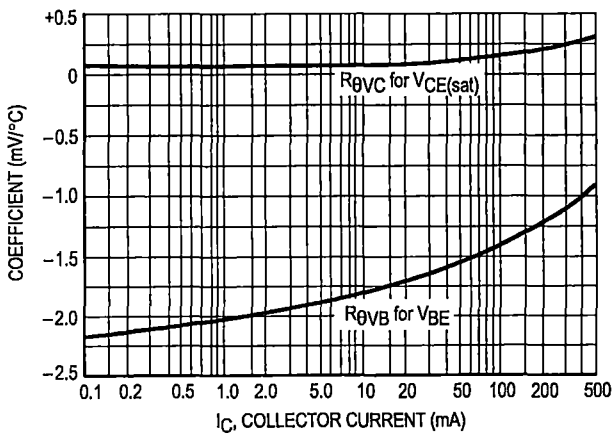
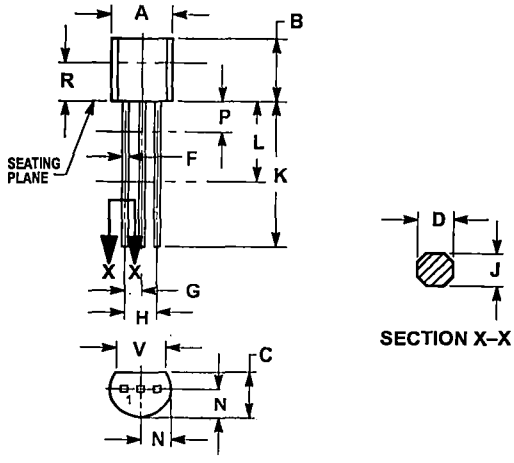



**Figure 12. Temperature Coefficients**

## PACKAGE DIMENSIONS




NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. CONTOUR OF PACKAGE BEYOND DIMENSION R IS UNCONTROLLED.
4. DIMENSION F APPLIES BETWEEN P AND L. DIMENSION D AND J APPLY BETWEEN L AND K MINIMUM. LEAD DIMENSION IS UNCONTROLLED IN P AND BEYOND DIMENSION K MINIMUM.

| | INCHES | | MILLIMETERS | |
|---|---|---|---|---|
| DIM | MIN | MAX | MIN | MAX |
| A | 0.175 | 0.205 | 4.45 | 5.20 |
| B | 0.170 | 0.210 | 4.32 | 5.33 |
| C | 0.125 | 0.165 | 3.18 | 4.19 |
| D | 0.016 | 0.022 | 0.41 | 0.55 |
| F | 0.016 | 0.019 | 0.41 | 0.48 |
| G | 0.045 | 0.055 | 1.15 | 1.39 |
| H | 0.095 | 0.105 | 2.42 | 2.66 |
| J | 0.015 | 0.020 | 0.39 | 0.50 |
| K | 0.500 | — | 12.70 | — |
| L | 0.250 | — | 6.35 | — |
| N | 0.080 | 0.105 | 2.04 | 2.66 |
| P | — | 0.100 | — | 2.54 |
| R | 0.115 | — | 2.93 | — |
| V | 0.135 | — | 3.43 | — |

**CASE 029–04**
**(TO–226AA)**
**ISSUE AD**

STYLE 17:
PIN 1. COLLECTOR
2. BASE
3. EMITTER



**How to reach us:**
**USA/EUROPE/Locations Not Listed:** Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036. 1–800–441–2447 or 602–303–5454

**MFAX:** RMFAX0@email.sps.mot.com – TOUCHTONE 602–244–6609
**INTERNET:** http://Design–NET.com

**JAPAN:** Nippon Motorola Ltd.; Tatsumi–SPD–JLDC, 6F Seibu–Butsuryu–Center, 3–14–2 Tatsumi Koto–Ku, Tokyo 135, Japan. 03–81–3521–8315

**ASIA/PACIFIC:** Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park, 51 Ting Kok Road, Tai Po, N.T., Hong Kong. 852–26629298

# ANEXO C.
# CARACTERÍSTICAS DEL MICROCONTROLADOR PIC16F877A EXTRAÍDAS DE SU HOJA TÉCNICA

MICROCHIP

# PIC16F87XA

## 28/40/44-Pin Enhanced Flash Microcontrollers

### Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

### High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

### Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and $I^2C$™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

### Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference ($V_{REF}$) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

### Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

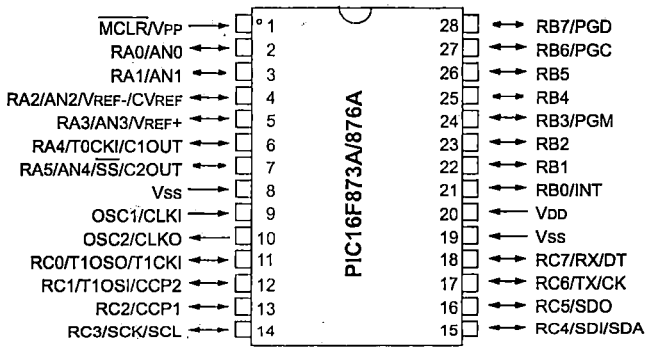
### CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

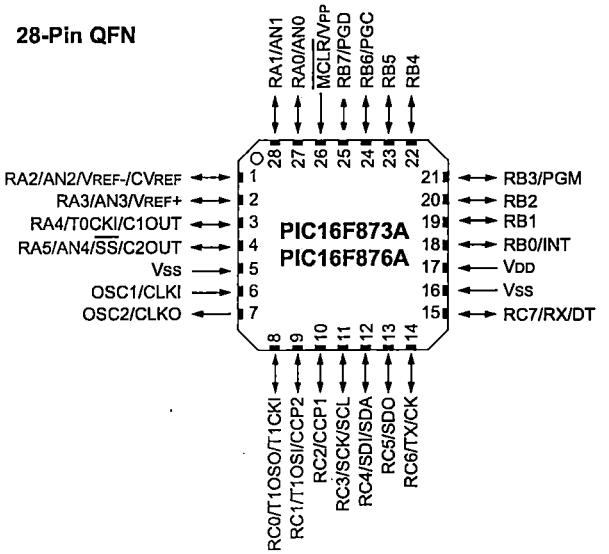
| Device | Program Memory | | Data SRAM (Bytes) | EEPROM (Bytes) | I/O | 10-bit A/D (ch) | CCP (PWM) | MSSP | | USART | Timers 8/16-bit | Comparators |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Bytes | # Single Word Instructions | | | | | | SPI | Master $I^2C$ | | | |
| PIC16F873A | 7.2K | 4096 | 192 | 128 | 22 | 5 | 2 | Yes | Yes | Yes | 2/1 | 2 |
| PIC16F874A | 7.2K | 4096 | 192 | 128 | 33 | 8 | 2 | Yes | Yes | Yes | 2/1 | 2 |
| PIC16F876A | 14.3K | 8192 | 368 | 256 | 22 | 5 | 2 | Yes | Yes | Yes | 2/1 | 2 |
| PIC16F877A | 14.3K | 8192 | 368 | 256 | 33 | 8 | 2 | Yes | Yes | Yes | 2/1 | 2 |

## Pin Diagrams

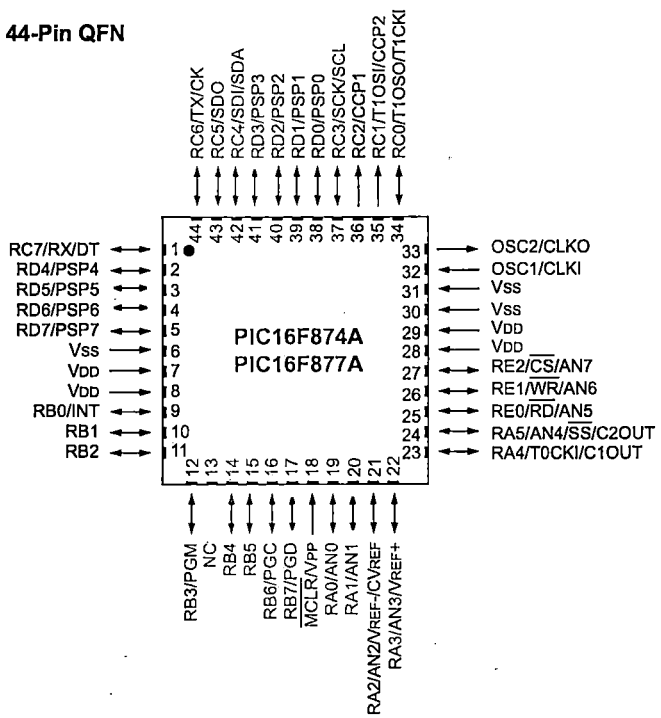
### 28-Pin PDIP, SOIC, SSOP




### 28-Pin QFN

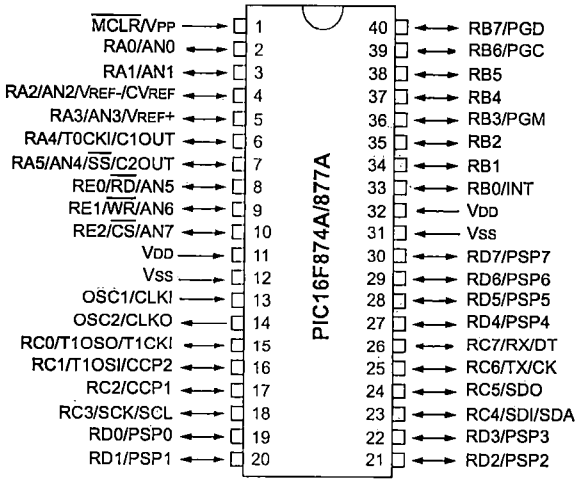



### 44-Pin QFN

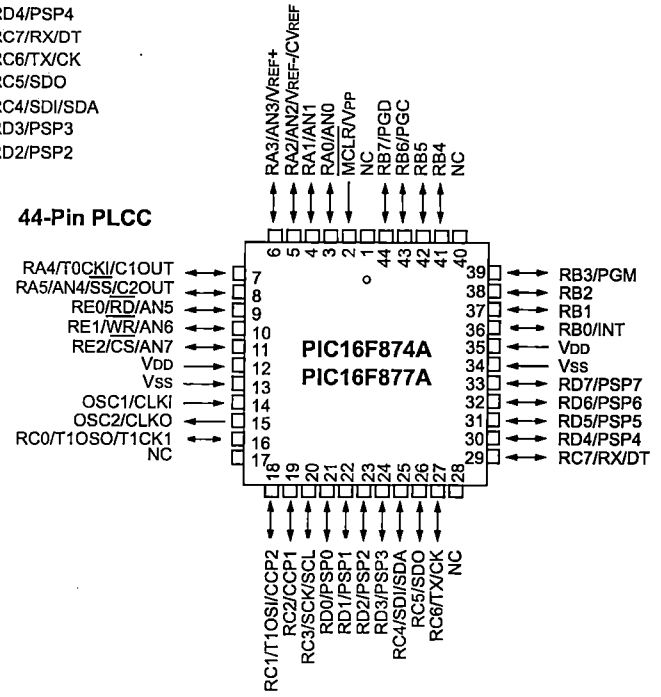


## Pin Diagrams (Continued)

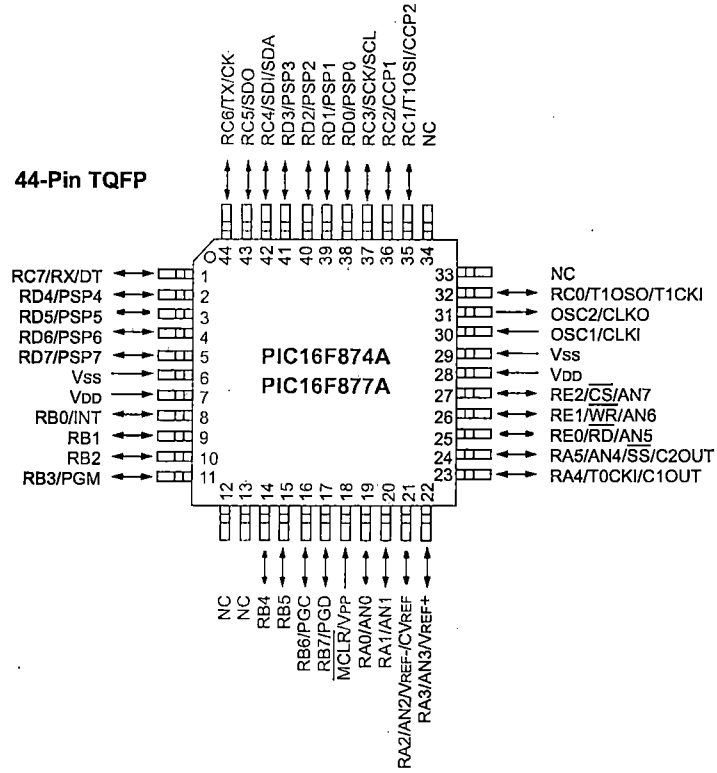
### 40-Pin PDIP




### 44-Pin PLCC




### 44-Pin TQFP



## 1.0 DEVICE OVERVIEW

This document contains device specific information about the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

PIC16F873A/876A devices are available only in 28-pin packages, while PIC16F874A/877A devices are available in 40-pin and 44-pin packages. All devices in the PIC16F87XA family share common architecture with the following differences:

- The PIC16F873A and PIC16F874A have one-half of the total on-chip memory of the PIC16F876A and PIC16F877A
- The 28-pin devices have three I/O ports, while the 40/44-pin devices have five
- The 28-pin devices have fourteen interrupts, while the 40/44-pin devices have fifteen
- The 28-pin devices have five A/D input channels, while the 40/44-pin devices have eight
- The Parallel Slave Port is implemented only on the 40/44-pin devices

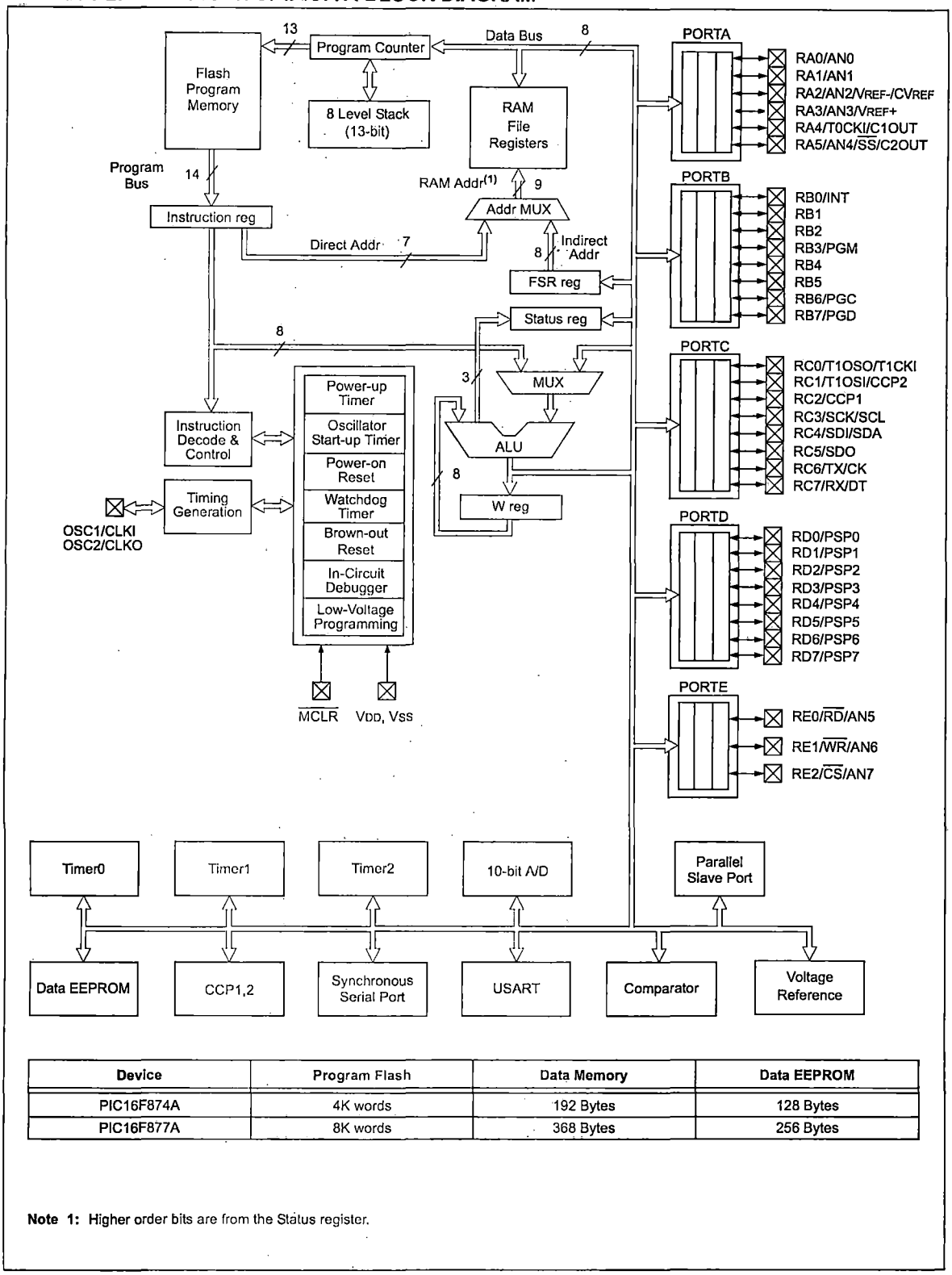
The available features are summarized in Table 1-1. Block diagrams of the PIC16F873A/876A and PIC16F874A/877A devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2 and Table 1-3.

Additional information may be found in the PICmicro® Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip web site. The Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

**TABLE 1-1: PIC16F87XA DEVICE FEATURES**

| Key Features | PIC16F873A | PIC16F874A | PIC16F876A | PIC16F877A |
|---|---|---|---|---|
| Operating Frequency | DC – 20 MHz | DC – 20 MHz | DC – 20 MHz | DC – 20 MHz |
| Resets (and Delays) | POR, BOR (PWRT, OST) | POR, BOR (PWRT, OST) | POR, BOR (PWRT, OST) | POR, BOR (PWRT, OST) |
| Flash Program Memory (14-bit words) | 4K | 4K | 8K | 8K |
| Data Memory (bytes) | 192 | 192 | 368 | 368 |
| EEPROM Data Memory (bytes) | 128 | 128 | 256 | 256 |
| Interrupts | 14 | 15 | 14 | 15 |
| I/O Ports | Ports A, B, C | Ports A, B, C, D, E | Ports A, B, C | Ports A, B, C, D, E |
| Timers | 3 | 3 | 3 | 3 |
| Capture/Compare/PWM modules | 2 | 2 | 2 | 2 |
| Serial Communications | MSSP, USART | MSSP, USART | MSSP, USART | MSSP, USART |
| Parallel Communications | — | PSP | — | PSP |
| 10-bit Analog-to-Digital Module | 5 input channels | 8 input channels | 5 input channels | 8 input channels |
| Analog Comparators | 2 | 2 | 2 | 2 |
| Instruction Set | 35 Instructions | 35 Instructions | 35 Instructions | 35 Instructions |
| Packages | 28-pin PDIP<br>28-pin SOIC<br>28-pin SSOP<br>28-pin QFN | 40-pin PDIP<br>44-pin PLCC<br>44-pin TQFP<br>44-pin QFN | 28-pin PDIP<br>28-pin SOIC<br>28-pin SSOP<br>28-pin QFN | 40-pin PDIP<br>44-pin PLCC<br>44-pin TQFP<br>44-pin QFN |

**FIGURE 1-2: PIC16F874A/877A BLOCK DIAGRAM**




| Device | Program Flash | Data Memory | Data EEPROM |
|---|---|---|---|
| PIC16F874A | 4K words | 192 Bytes | 128 Bytes |
| PIC16F877A | 8K words | 368 Bytes | 256 Bytes |

**Note 1:** Higher order bits are from the Status register.

# ANEXO D.
# CÓDIGO EN LENGUAJE ENSAMBLADOR PARA EL MICROCONTROLADOR PIC 16F877A

```
    LIST P=16F877A
    include "p16f877a.inc"
    include "macros.inc"

dato_temp_1a    equ 0x20; se cargan en el principal y se usan en la 1°. llamada
dato_temp_1b    equ 0x21
dato_temp_1c    equ 0x22
dato_temp_1d    equ 0x23
dato_temp_1e    equ 0x24


dato_int_1a     equ 0x25;se usan en las interrupciones 2F
dato_int_1b     equ 0x26
dato_int_1c     equ 0x27
dato_int_1d     equ 0x28
dato_int_1e     equ 0x29


;**************************
;variables temporales
;**************************
conver_var  equ 0x2A

;*******************************
;variables globales
;*******************************
;variables para salvar en las interrupciones
temp        equ 0x2B
w_temp      equ 0x2C
status_temp equ 0x2D
pclath_temp equ 0x2E

menu_dato   equ 0x2F
eje1        equ 0x30
dual1       equ 0x31

I2C_VAR     equ 0x32 ;8bytes (32-39)
fecha_var   equ 0x3D ;6bytes (3A-3F)


;49-58 números de autos y pesados
ind_RT      equ     0x00
banderas    equ 0x43
ind_reset   equ 0x44
num_reset   equ 0x45
cont        equ 0x46
var_aux1    equ 0x47
var_aux2    equ 0x48
puntero     equ 0x49
w2      equ 0x4A
LCD_DATA equ PORTD
LCD_CTRL equ PORTD
rs equ .1
rw equ .2
e  equ .0
#Define FOSC    .20000000                ;Hz
#Define BAUD    .57600                ;bps
#Define BAUD_I2C .100000

    org 0x00
    goto inicio
    org 0x04
    goto inter
MENS_1
    dt "JL Controles y",0x00
MENS_2
    dt "Sistemas Elec.",0x00
MENS_3
    dt "Ingrese la fecha",0x00
MENS_4
    dt "aammddhhmm",0x00
MENS_5
    dt "Pular OK...",0x00
tabla_cabina1_auto
    movf eje1,0
    sublw .3
```

```
    btfss STATUS,C
    retlw 0x4B
    movf eje1,0
    addwf PCL,1
    return
    retlw 0x49
    retlw 0x49
    retlw 0x4B
    return
tabla_cabina1_camion
    movf eje1,0
    sublw .7
    btfss STATUS,C
    retlw 0x57
    movf eje1,0
    addwf PCL,1
    return
    retlw 0x4D ;camion 1 eje
    retlw 0x4D ;camion 2 ejes
    retlw 0x4F ;camion 3 ejes
    retlw 0x51 ;camion 4 ejes
    retlw 0x53 ;camion 5 ejes
    retlw 0x55 ;camion 6 ejes
    retlw 0x57 ;camion 7 ejes
    return
    include "lcd4c.inc"
    include "24LCXXX2.inc"
    include "serial.inc"
    include "delay.inc"
    include "fecha.inc"
    include "matematica.inc"
    include "interrupciones.inc"
    include "configuracion.inc"
inicio
    BANCO0
    movlw 0x30
    movwf puntero
    call LCD_UP4
    call LCD_INI
    call LCD_CLEAR4
adc_config
    BANCO1
    movlw b'00000101'
    movwf ADCON1
    BANCO0
    movlw b'10000001'
    andwf ADCON0,1

    BANCO1
    bcf TRISC,5
    BANCO0
;************************************
    bcf banderas,ind_RT
    clrf ind_reset
    lee_eeprom 0x06
    movwf num_reset
    incf num_reset,1
    graba_eeprom 0x06,num_reset
;************************************
    bcf PORTC,5
    movlw MENS_1
    call LCD_MENSAJE4
    movlw 0x40
    call LCD_SETDDADDRE4
    movlw MENS_2
    call LCD_MENSAJE4
    movlw .2
    call DELAY_S
    call LCD_CLEAR4
    movlw MENS_5
    call LCD_MENSAJE4
    call USART_UP
;************************************
;pregunta para continuar o iniciar memoria
    btfss PIR1,RCIF
    goto $-1
```

```
    movf RCREG,0
    xorlw 'z'
    btfsc STATUS,Z
    call config_memoria
;************************************
    call lee_direccion
    call setear_fecha
    call timer1_up
    call inicia_timer1
    call rb
    bsf INTCON,GIE



;*****************************************************************
; cuerpo principal del programa "main"
;*****************************************************************
;include "main_2ca.inc";aquí está la rutina para 2 casetas
menu
    btfss PIR1,RCIF
    goto menu
    movf RCREG,0
    movwf menu_dato
    movlw 'f'
    xorwf menu_dato,0
    btfsc STATUS,Z
    call setear_fecha
    goto menu
    movlw 's'
    xorwf menu_dato,0
    btfsc STATUS,Z
    bsf banderas,ind_RT
    goto menu
    movlw 'c'
    xorwf menu_dato,0
    btfsc STATUS,Z
    bcf banderas,ind_RT
    goto menu
    movlw 'd'
    xorwf menu_dato,0
    btfsc STATUS,Z
    goto descarga
    movlw 't'
    xorwf menu_dato,0
    btfsc STATUS,Z
    goto sensores
    movlw 'e'
    xorwf menu_dato,0
    btfss STATUS,Z
    goto menu
sensores_2
    bcf INTCON,RBIE
    btfss PIR1,RCIF
    goto $-1
    movf RCREG,0
    xorlw 'n'
    btfss STATUS,Z
    goto sal_sen2
    movf PORTB,w
    andlw b'11110000'
    call TXWSERIAL
    goto menu
sal_sen2
    bsf INTCON,RBIE
    goto menu
sensores
    bcf ADCON0,3 ;canal_0
    retardo_us .20
    bsf ADCON0,2 ;inicia conversión
    btfsc ADCON0,2
    goto $-1
    BANCO1
    movf ADRESL,w
    BANCO0
    call TXWSERIAL
    retardo_ms .20
```

```
    bsf ADCON0,3 ;canal_1
    retardo_us .20
    bsf ADCON0,2 ;inicia conversión
    btfsc ADCON0,2
    goto $-1
    BANCO1
    movf ADRESL,w
    BANCO0
    call TXWSERIAL
    goto menu
descarga
    movlw 'd'
    call TXWSERIAL

    mover_num 0xA0,I2CCHIPSET_lect
    clrf I2CADDRESS_L_lect
    clrf I2CADDRESS_H_lect
repite_descarga
    movf I2CCHIPSET_lect,f
    btfsc STATUS,Z
    goto sal_descarga
    btfss PIR1,RCIF
    goto $-1
    movf RCREG,0
    xorlw 'l'
    btfsc STATUS,Z
    goto menu
    clrf cont

d1_8bytes
    call lee_i2c
    call TXWSERIAL
    call incrementa_dir_lec
    incf cont,f
    btfss cont,3
    goto d1_8bytes
    goto repite_descarga



sal_descarga
    btfss PIR1,RCIF
    goto $-1
    movf RCREG,0
    xorlw 'l'
    btfsc STATUS,Z
    goto menu
    clrf cont
d2_8bytes
    movlw 't'
    call TXWSERIAL
    incf cont,f
    btfss cont,3
    goto d2_8bytes
    goto menu

;*******************************************************************************

;************************************************
;subrutina para ingresar la fecha               *
;************************************************
setear_fecha                                    ;
    call LCD_CLEAR4                             ;
    movlw MENS_3                                ;
    call LCD_MENSAJE4                           ;
    movlw 0x40                                  ;
    call LCD_SETDDADDRE4                        ;
    movlw MENS_4                                ;
    call LCD_MENSAJE4                           ;
    movlw 0x40                                  ;
    call LCD_SETDDADDRE4                        ;
    movlw 0x07                                  ;
    call LCD_DISCURCONT4                        ;
ingreso_fecha                                   ;
    btfss PIR1,RCIF                             ;
    goto $-1                                    ;
```

```
    movf RCREG,0
    movwf dato_temp_1a
carga
    movlw '\b'
    xorwf dato_temp_1a,0
    btfss STATUS,Z
    goto $+4
;*********************************
;retrocede el cursor una posicion;*
    movlw 0x00                  ;**
    call LCD_DISCURSHIF4        ;**
;*********************************
    goto ingreso_fecha
    movlw '\r'
    xorwf dato_temp_1a,0
    btfsc STATUS,Z
    goto continua
;*****************************************
;lee y muestra caracter en la pantalla LCD;*
    movf dato_temp_1a,0                ;***
    call LCD_WRITEDATO4                ;***
;*****************************************
    goto ingreso_fecha

;*****************************************
;lee caracteres ingresados en el LCD y los ***
;guarda en año, mes, dia,hora,minuto en hexa-*
;decimal                                     *
;*****************************************
continua                                   ;*
    movlw año                              ;*
    movwf FSR                              ;*
    movlw 0x40                             ;*
    call LCD_SETDDADDRE4                   ;*
sigue_leiendo                              ;*
    call LCD_READDATO4                     ;*
    movwf dato_temp_1a                     ;*
    movlw 0x30                             ;*
    subwf dato_temp_1a,1                   ;*
    call LCD_READDATO4                     ;*
    movwf dato_temp_1b                     ;*
    movlw 0x30                             ;*
    subwf dato_temp_1b,1                   ;*
    movf dato_temp_1a,0                    ;*
    call multiplicar_por_10 ;multiplica dato_1*10
    movwf dato_temp_1a
    movf dato_temp_1b,0
    addwf dato_temp_1a,0
    movwf INDF
    movlw minuto
    xorwf FSR,0
    btfsc STATUS,Z
    goto $+3
    incf FSR,1
    goto sigue_leiendo
    movlw b'00000100'
    call LCD_DISCURCONT4
    call guarda_fecha
    call mostrar_fecha
    return
;*****************************************
;fin de subrutina par ingresar fecha
;*****************************************

;*****************************************
;subrutina para mostrar fecha
;*****************************************
mostrar_fecha
    call LCD_CLEAR4
mostrar_año
    movlw 0x00
    call LCD_SETDDADDRE4
    movf año,0
    call conversion_bcd_2bytes
    movf bcd_h,0
    call LCD_WRITEDATO4
```

```
    movf bcd_l,0
    call LCD_WRITEDATO4
    movlw '/'
    call LCD_WRITEDATO4
mostrar_mes
    movlw 0x03
    call LCD_SETDDADDRE4
    movf mes,0
    call conversion_bcd_2bytes
    movf bcd_h,0
    call LCD_WRITEDATO4
    movf bcd_l,0
    call LCD_WRITEDATO4
    movlw '/'
    call LCD_WRITEDATO4
mostrar_dia
    movlw 0x06
    call LCD_SETDDADDRE4
    movf dia,0
    call conversion_bcd_2bytes
    movf bcd_h,0
    call LCD_WRITEDATO4
    movf bcd_l,0
    call LCD_WRITEDATO4
mostrar_hora
    movlw 0x40
    call LCD_SETDDADDRE4
    movf hora,0
    call conversion_bcd_2bytes
    movf bcd_h,0
    call LCD_WRITEDATO4
    movf bcd_l,0
    call LCD_WRITEDATO4
    movlw ':'
    call LCD_WRITEDATO4
mostrar_minuto
    movlw 0x43
    call LCD_SETDDADDRE4
    movf minuto,0
    call conversion_bcd_2bytes
    movf bcd_h,0
    call LCD_WRITEDATO4
    movf bcd_l,0
    call LCD_WRITEDATO4
    movlw ':'
    call LCD_WRITEDATO4
mostrar_segundo
    movlw 0x46
    call LCD_SETDDADDRE4
    movf segundo,0
    call conversion_bcd_2bytes
    movf bcd_h,0
    call LCD_WRITEDATO4
    movf bcd_l,0
    call LCD_WRITEDATO4
    return
;*************************************************
;fin de rutina para mostrar fecha
;*************************************************

timer1_up
    movlw   b'00111110'
    movwf   T1CON
    movlw   0x00
    movwf   TMR1L
    movlw   0x10
    movwf   TMR1H
    return


inicia_timer1
    bcf PIR1,TMR1IF
    BANCO1
    bsf PIE1,TMR1IE
    bsf TRISC,0
    bsf TRISC,1
```

```
    BANCO0
    bsf T1CON,TMR1ON
    bsf INTCON,PEIE
    return


rb
    BANCO1
    bsf OPTION_REG,7
    BANCO0
    bsf INTCON,RBIE
    bcf INTCON,RBIF
    return

ini_var
    clrf eje1
    clrf dual1
    clrf banderas
    BANCO1
    movlw 0xFF
    movwf TRISB
    BANCO0
    return
    include "eprompic.inc"
    end
```

```
;*****************************************************************************************
;                  I2C MODO MAESTRO Y clock = FOSC / (4 * (SSPADD+1)) CON INTERRUPCIONES
;*****************************************************************************************
;*****************************************************************************
;                   I2C_MPSS.INC
;
;               Autor: Alfredo Moisés Quispe Espinoza
;
;
;Configura y controla la transmisión y recepción en el protocolo I2C del múdulo MSSP.
;Con la llamada a I2C_INI configura los puertos y registros de control, con la rutina ESCRBYTE
escribe
;el byte I2CDATO en la dirección I2CADDRESS del chip dado por I2CCHIPSET,  la rutina LEERBYTE lee
la
;dirección I2CADDRESS del chip dado por I2CCHIPSET y lo guarda en I2CDATO. Si ocurre un error en la
;transmisión reinicia hasta 16 intentos, en caso de no transmitir bién el valor de IDLEBUSS será
0xFE
;No olvidar definir BAUD (velocidad de transmision en Baudios) y FOSC (freciencia del oscilador en
KHz)
;con #Define. Por último poner: call I2CCONTROL en la rutina de interrupción.
;
;Este fichero se debe incluir en los futuros programas fuente mediante la directiva INCLUDE.
;
;I2C_UP:    Configura los pines SCL y SDA, el SSPCON, el SSPSTAT, registros internos y las
;       interrupciones para el uso del I2C.
;
;ESCRBYTE:  Escribe el byte I2CDATO en la dirección I2CADDRESS del chip dado por I2CCHIPSET.
;
;LEERBYTE:  lee la dirección I2CADDRESS del chip dado por I2CCHIPSET y lo guarda en I2CDATO.
;
;Maneja ciclos de lecctura y escritura de EEPROMs 24CXX conectados:
;   RC3 --> SCL
;   RC4 <-> SDA
;
;
#define I2CWRIT bcf I2CCHIPSET,0     ; R/W'=0 (escribir en la EEPROM)
#define I2CREAD bsf I2CCHIPSET,0     ; R/W'=1 (leer la EEPROM)
;
        cblock  I2C_VAR
            I2CCHIPSET
            I2CADDRESS_H
            I2CADDRESS_L
            I2CCHIPSET_lect
            I2CADDRESS_L_lect
            I2CADDRESS_H_lect
            I2CDATO
            aux_I2C
            tamaño_h
            tamaño_l
            num_memo
        endc
;
;*****************************************************************************************
;                   SUBPROGRAMA PRINCIPAL
;*****************************************************************************************
I2C_UP
    BANCO1
    bsf     TRISC, 3            ; Set SCL input
    bsf     TRISC, 4            ; Set SDA input
    movlw   (FOSC/(4*BAUD_I2C))-1   ; Calculates SSPADD Setting for
    movwf   SSPADD
    movlw   B'10000000'
    movwf   SSPSTAT
    BANCO0
    movlw   B'00111000'         ; SSP enable, I2C Master mode, clock = FOSC / (4 * (SSPADD+1))
    movwf   SSPCON
    bcf     PIR1,SSPIF          ; Clear SSP interrupt flag
    return
;
ESCRBYTE
    I2CWRIT
    clrf aux_I2C
    call    SENDSTART

ESCBYCON
    call    SENDCHIPSET
```

```
    call    SENDADDRESS_H
    call    SENDADDRESS_L
    call    SENDDATO
    call    SENDSTOP
    return
;
LEERBYTE
    bcf I2CCHIPSET_lect,0
    bsf aux_I2C,0
    BANCO1
    bsf     SSPCON2,ACKDT
    BANCO0
    call    SENDSTART

LEEBYCON
    call    SENDCHIPSET
    call    SENDADDRESS_H
    call    SENDADDRESS_L
;*********************************
    call    SENDSTOP
    call    SENDSTART
    call    SENDCHIPSET
    call    GETDATO
    call    SENDACK
    call    SENDSTOP
;*********************************
    bcf     PIR1,SSPIF
    return
;
;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;                       SUBRUTINAS
;**********************************************************************************************

SENDSTART
    bcf     PIR1,SSPIF
    BANCO1
    bsf     SSPCON2,SEN
    BANCO0
    return

SENDCHIPSET
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    btfsc aux_I2C,0
    goto $+3
    movf    I2CCHIPSET,0
    goto $+2
    movf    I2CCHIPSET_lect,0
    movwf   SSPBUF
    return
;
SENDADDRESS_H
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    btfsc aux_I2C,0
    goto $+3
    movf    I2CADDRESS_H,0
    goto $+2
    movf    I2CADDRESS_H_lect,0
    movwf   SSPBUF
    return

SENDADDRESS_L
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    btfsc aux_I2C,0
    goto $+3
    movf    I2CADDRESS_L,0
    goto $+2
    movf    I2CADDRESS_L_lect,0
    movwf   SSPBUF
    return
```

```
SENDDATO
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    movf    I2CDATO,0
    movwf   SSPBUF
    return
;
GETDATO
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    BANCO1
    bsf     SSPCON2,RCEN
    BANCO0
    return
;
SENDACK
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    movf    SSPBUF,0
    movwf   I2CDATO
    BANCO1
    bsf     SSPCON2,ACKEN
    bcf     SSPCON2,ACKSTAT
    BANCO0
    return
;
SENDSTOP
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    BANCO1
    bsf     SSPCON2,PEN
    BANCO0
    return

RESTART
    btfss PIR1,SSPIF
    goto $-1
    bcf     PIR1,SSPIF
    bsf I2CCHIPSET_lect,0
RESTENV
    BANCO1
    bsf     SSPCON2,RSEN
    BANCO0
    return
escribe_i2c
    movwf I2CDATO
    call ESCRBYTE
    return
lee_i2c
    call LEERBYTE
    movf I2CDATO,w
    return
incrementa_dir_lec
    incfsz I2CADDRESS_L_lect,1
    return
    incf I2CADDRESS_H_lect,f
    movf tamaño_h,w
    xorwf I2CADDRESS_H_lect,w
    btfss STATUS,Z
    return
    movlw 0x02
    addwf I2CCHIPSET_lect,1
    movf num_memo,w
    xorwf I2CCHIPSET_lect,w
    btfss STATUS,Z
    return
    movlw 0x00
    movwf I2CCHIPSET_lect
    return
incrementa_dir_escr
    incfsz I2CADDRESS_L,1
    return
```

```
incf I2CADDRESS_H,f
movf tamaño_h,w
xorwf I2CADDRESS_H,w
btfss STATUS,Z
return
movlw 0x02
addwf I2CCHIPSET,1
movf num_memo,w
xorwf I2CCHIPSET,w
btfss STATUS,Z
return
movlw 0xA0
movwf I2CCHIPSET
return
```

```
config_memoria
    num_var_local .7
    btfss PIR1,RCIF
    goto $-1
    movf RCREG,w
    movwf INDF
    movlw '\r'
    xorwf INDF,w
    btfsc STATUS,Z
    goto $+3
    decf FSR,f
    goto $-9
    incf FSR,f
    graba_eeprom 0x05,INDF
    movf INDF,w
    call TXWSERIAL
    incf FSR,f
    graba_eeprom 0x04,INDF
    movf INDF,w
    call TXWSERIAL
    incf FSR,f
    graba_eeprom 0x03,INDF
    movf INDF,w
    call TXWSERIAL
    incf FSR,f
    graba_eeprom 0x02,INDF
    movf INDF,w
    call TXWSERIAL
    incf FSR,f
    graba_eeprom 0x01,INDF
    movf INDF,w
    call TXWSERIAL
    incf FSR,f
    graba_eeprom 0x00,INDF
    movf INDF,w
    call TXWSERIAL
    fin_subrutina .7
    return

lee_direccion
    lee_eeprom 0x00
    movwf I2CCHIPSET
    lee_eeprom 0x01
    movwf I2CADDRESS_H
    lee_eeprom 0x02
    movwf I2CADDRESS_L
    lee_eeprom 0x03
    movwf tamaño_h
    lee_eeprom 0x04
    movwf tamaño_l
    lee_eeprom 0x05
    movwf num_memo
    return
```

```
;*****************************************************************************************
;                    RETARDOS DE TIEMPO
;*****************************************************************************************
;*****************************************************************************
;                    DELAY.INC
;
;               Autor: Alfredo Moisés Quispe Espinoza
;
;
; Se tendrá las siguientes rutinas:
;   -   DELAY_US
;   -   DELAY_MS
;   -   DELAY_S
;Estas rutinas controlan los tiempos de retardo en microsegundos,
;milisegundos y en segundos. La cantidad microsegundos, de milisegundos
;o segundos toma del acumulador.
DELAY_US
        movwf w2
        num_var_local .1
        movf w2,w
        movwf   INDF
        decf INDF,1
        decf INDF,1
        nop
        nop
nuevo_us
        decf INDF,1
        btfsc STATUS,Z
        goto $+2
        goto nuevo_us
        nop
        fin_subrutina .1
        return


DELAY_MS
        movwf w2
        num_var_local .1
        movf w2,w
        movwf   INDF
nuevo_ms
        movlw .250
        call DELAY_US
        movlw .250
        call DELAY_US
        movlw .250
        call DELAY_US
        movlw .248
        call DELAY_US
        nop
        nop
        nop
        decfsz INDF,1
        goto nuevo_ms
        fin_subrutina .1
        return

DELAY_S
        movwf w2
        num_var_local .1
        movf w2,w
        movwf   INDF
nuevo_s
        movlw .250
        call DELAY_MS
        movlw .250
        call DELAY_MS
        movlw .250
        call DELAY_MS
        movlw .250
        call DELAY_MS
        decfsz INDF,1
        goto nuevo_s
        fin_subrutina .1
        return
```

```
escribir_EE
    BANCO3
    bcf EECON1,EEPGD
    bsf EECON1,WREN
    bcf INTCON,GIE
    movlw 0x55
    movwf EECON2
    movlw 0xAA
    movwf EECON2
    bsf EECON1,WR
    BANCO0
    btfss PIR2,EEIF
    goto $-1
    bcf PIR2,EEIF
    BANCO3
    bcf EECON1,WREN
    BANCO0
    return
leer_EE
    BANCO3
    bcf EECON1,EEPGD
    bsf EECON1,RD
    BANCO2
    movf EEDATA,0
    BANCO0
    return
```

```
    cblock  fecha_var
        año
        mes
        dia
        hora
        minuto
        segundo
    endc
incrementa_segundo
    incf segundo,f
    movlw 0x02
    xorwf segundo,w
    btfsc STATUS,Z
    bcf PORTC,5
    movlw .60
    xorwf segundo,0
    btfsc STATUS,Z
    goto incrementa_minuto
    call mostrar_segundo
    return
incrementa_minuto
    clrf segundo
    incf minuto,1
    movlw .60
    xorwf minuto,0
    btfsc STATUS,Z
    goto incrementa_hora
    call mostrar_minuto
    return
incrementa_hora
    clrf minuto
    incf hora,1
    movlw .24
    xorwf hora,0
    btfsc STATUS,Z
    goto incrementa_dia
    call mostrar_hora
    call guarda_fecha
    return
incrementa_dia
    clrf hora
    bsf PCLATH,1
    incf dia,f
    movf mes,0
    addwf PCL,1
    nop
    goto mes_31dias ;enero
    goto mes_28dias ;febrero
    goto mes_31dias ;marzo
    goto mes_30dias ;abril
    goto mes_31dias ;mayo
    goto mes_30dias ;junio
    goto mes_31dias ;julio
    goto mes_31dias ;agosto
    goto mes_30dias ;setiembre
    goto mes_31dias ;octubre
    goto mes_30dias ;noviembre
    goto mes_31dias ;diciembre

mes_30dias
    movlw .31
    goto evalua
mes_31dias
    movlw .32
    goto evalua
mes_28dias
    btfss año,0
    btfsc año,1
    goto $+3;28 dias
    movlw .30 ;29 dias
    goto evalua
    movlw .29
    goto evalua
evalua
    bcf PCLATH,1
    xorwf dia,0
```

```
    btfsc STATUS,Z
    goto incrementa_mes
    call mostrar_dia
    return
incrementa_mes
    movlw .1
    movwf dia
    incf mes,1
    movlw .13
    xorwf mes,0
    btfsc STATUS,Z
    goto incrementa_año
    call mostrar_mes
    return
incrementa_año
    movlw .1
    movwf mes
    incf año,1
    call mostrar_año
    return
guarda_fecha
    movlw 0x80
    call escribe_i2c
    call incrementa_dir_escr
    movlw 0x80
    call escribe_i2c
    call incrementa_dir_escr
    movlw 0x80
    call escribe_i2c
    call incrementa_dir_escr
    movlw 0x80
    call escribe_i2c
    call incrementa_dir_escr
    movf año,w
    call escribe_i2c
    call incrementa_dir_escr
    movf mes,w
    call escribe_i2c
    call incrementa_dir_escr
    movf dia,w
    call escribe_i2c
    call incrementa_dir_escr
    movf hora,w
    call escribe_i2c
    call incrementa_dir_escr
    movf num_reset,w
    call escribe_i2c
    clrf dato_int_1a
    incf dato_int_1a,f
    call incrementa_dir_escr
    btfss dato_int_1a,3
    goto $-3
    movlw 0x80
    call escribe_i2c
    graba_eeprom 0x00,I2CCHIPSET
    graba_eeprom 0x01,I2CADDRESS_H
    graba_eeprom 0x02,I2CADDRESS_L
    btfss ind_reset,0
    goto $+5
    clrf num_reset
    graba_eeprom 0x06,num_reset
    bsf PORTC,5
    goto $+2
    bsf ind_reset,0
    movlw 0x0F
    call LCD_SETDDADDRE4
    movf num_reset,w
    call LCD_WRITEDATO4
    return
```

```
inter
    movwf w_temp
    movf STATUS,0
    movwf status_temp
    movf PCLATH,0
    movwf pclath_temp
bifurca_interrupcion
    btfsc   PIR1,TMR1IF
    goto    inter_timer1
    btfsc INTCON,RBIF
    goto inter_rb
    goto salir_interrupcion
inter_timer1
    bcf PIR1,TMR1IF
    movlw   0x10
    movwf   TMR1H
    call incrementa_segundo
    goto salir_interrupcion
inter_rb
    btfss PORTB,7
    goto guarda
    btfsc PORTB,6
    incf eje1,1
    btfsc PORTB,5
    incf dual1,1
    bcf INTCON,RBIF
    goto salir_interrupcion
guarda
    movf minuto,w
    call escribe_i2c
    call incrementa_dir_escr
    movf segundo,w
    call escribe_i2c
    call incrementa_dir_escr

    movlw 0x0A
    call LCD_SETDDADDRE4
    movf dual1,f
    btfss STATUS,Z
    goto camion
    goto auto
auto
    call tabla_cabina1_auto
    movwf FSR
    movwf dato_int_1a
    call numero_de_unidad
    movlw 'L'
    movwf dato_int_1e
    call escribe_i2c
    call incrementa_dir_escr
    movlw 'L'
    call LCD_WRITEDATO4
    goto muestra_unidad
camion
    call tabla_cabina1_camion
    movwf FSR
    movwf dato_int_1a
    call numero_de_unidad
    movlw 'P'
    movwf dato_int_1e
    call escribe_i2c
    call incrementa_dir_escr
    movlw 'P'
    call LCD_WRITEDATO4
muestra_unidad
    movf eje1,0
    addlw 0x30
    call LCD_WRITEDATO4
    movf eje1,w
    call escribe_i2c
    call incrementa_dir_escr

    incf dato_int_1a,w
    movwf FSR
    movf INDF,w
    call escribe_i2c
```

```
    call incrementa_dir_escr

    decf FSR,1
    movf INDF,w
    call escribe_i2c
    call incrementa_dir_escr
    call incrementa_dir_escr
    call incrementa_dir_escr
    movlw 0x80
    call escribe_i2c

    movlw 0x4A
    call LCD_SETDDADDRE4

;************************************************
    ;hay que convertir a bcd para mostrar en lcd
;************************************************
    mover INDF,var_aux1
    incf FSR,1
    mover INDF,var_aux2
    convertir var_aux2,var_aux1,dato_int_1a,dato_int_1b,dato_int_1c,dato_int_1d,dato_int_1e
    movlw 0x4A
    call LCD_SETDDADDRE4
    movf dato_int_1e,w
    addlw 0x30
    call LCD_WRITEDATO4
    movf dato_int_1d,w
    addlw 0x30
    call LCD_WRITEDATO4
    movf dato_int_1c,w
    addlw 0x30
    call LCD_WRITEDATO4
    movf dato_int_1b,w
    addlw 0x30
    call LCD_WRITEDATO4
    movf dato_int_1a,w
    addlw 0x30
    call LCD_WRITEDATO4

;************************************************



    btfss banderas,ind_RT
    goto salir_interrupcion
enviando_a_pc
    movlw año
    movwf FSR
    movf INDF,w
    call TXWSERIAL
    movlw segundo
    xorwf FSR,w
    btfss STATUS,Z
    goto $-5
    movf dato_int_1e,w
    call TXWSERIAL
    movf eje1,w
    call TXWSERIAL
    incf dato_int_1a,w
    movwf FSR
    movf INDF,w
    call TXWSERIAL
    decf FSR,1
    movf INDF,w
    call TXWSERIAL
    movf I2CCHIPSET,w
    call TXWSERIAL
    movf I2CADDRESS_H,w
    call TXWSERIAL
    movf I2CADDRESS_L,w
    call TXWSERIAL

    goto salir_interrupcion
numero_de_unidad
    incf INDF,1
    btfss STATUS,Z
```

```
    return
    incf FSR,1
    incf INDF,1
    return


salir_interrupcion
    movf pclath_temp,0
    movwf PCLATH
    movf status_temp,0
    movwf STATUS
    btfss status_temp,0
    goto no_carry
carry
    movlw b'11111110'
    movwf temp
    rlf temp,0
setea_z
    movlw w_temp
    btfsc status_temp,2
    incf temp,1
    incf temp,1
    retfie
no_carry
    movlw b'01111110'
    movwf temp
    rlf temp,0
    bsf temp,7
    goto setea_z
```

```
;*****************************************************************************************
;           LCD HITACHI HD44780A 2 X 16 o 1 X 16 4 bits de datos
;*****************************************************************************************
;*************************************************************************
;                   LCD4.INC
;
;               Autor: Alfredo Moisés Quispe Espinoza
;
;
;Presenta un conjuntos de rutinas que permiten realizar las tareas de control del mòdulo
;de visualización LCD. Con la llamada a LCD_UP se configura para su uso y con la rutina LCD_INI
;se inicializa el LCD. Y la rutina MENSAJE visualiza un mensaje dado en una tabla.
;Las demás rutinas son las de principal uso de control. NOTA: Definir con "Define" los puertos
;LCD_DATA y LCD_CTRL, y en caso de tener cuatro bancos poner RP1 a 0 antes de LCD_UP si es
necesario.
;
;Este fichero se debe incluir en los futuros programas fuente mediante la directiva INCLUDE.
;
;LCD_UP4:           Configura los puertos A y B, y el INTCON para el uso del LCD.
;
;LCD_INI4:          Inicializa el LCD. Espera 15ms (riso a 4.5V), configura # de bits de datos,
;       # de lineas en activas el display y tipo de fuente.
;
;LCD_MENSAJE4:      Esta rutina saca al LCD el mensaje cuyo inicio esta indicado en el acumulador.
Se
;       aconseja para cada mensaje el siguiente formato:
;           MENS    equ $   ó   MENS    equ $
;           retlw   'O'         dt  "OK",0x00
;           retlw   'K'
;           retlw   0x00
;       Y debe hacer "movlw MENS" antes de la llamada "call MENSAJE"
;
;LCD_CLEAR4:        Borra el display y retorna el cursor a la posicion 0.
;
;LCD_HOME4:         Retorna el cursor a la posicion 0.
;
;LCD_ENTRYMODE4:    Configura el modo de ingreso al display.
;       El modo de funcionamiento deseado deberá cargarse en los 2 bits menores de W:
;               x x x x x x b1 b0
;       b0 :    0 = no desplazamiento del display   1 = desplazamiento
;       b1 :    0 = autodecremento del cursor       1 = autoincremento
;       b2-7 : sin importancia
;
;LCD_DISCURCONT4:   Enciende o apaga el display, cursor y parpadeo.
;       El modo de control deseado deberá cargarse en los 3 bits menores de W:
;               x x x x x b2 b1 b0
;       b0 :    0 = no parpadeo del cursor      1 = parpadeo
;       b1 :    0 = apagar cursor           1 = encender cursor
;       b2 :    0 = apagar display          1 = encender display (los datos siguen en DDRAM)
;       b3-7 : sin importancia
;
;LCD_DISCURSHIF4:   Mueve el cursor o el display.
;       El modo de control deseado deberá cargarse en los 2 bits terceros de W:
;               x x x x b3 b2 x x
;       b2 :    0 = mueve al a izquierda        1 = a la derecha
;       b3 :    0 = mueve cursor            1 = mueve display
;       b0,1,4-7 : sin importancia
;
;LCD_SETCGADDRE4:   Fija la dirección de la CGRAM. La dirección de la CGRAM puede ser
;       leída/escrita después de este comando
;       La dirección deseada deberá estar en el registro W:
;               x x b5 b4 b3 b2 b1 b0
;       b0-5 : dirección de la CGRAM
;       b6-7 : sin importancia
;
;LCD_SETDDADDRE4:   Fija la dirección de la DDRAM. La dirección de la DDRAM puede ser
;       leída/escrita después de este comando
;       La dirección deseada deberá estar en el registro W:
;               x b6 b5 b4 b3 b2 b1 b0
;       b0-6 : dirección de la DDRAM
;       b7 : sin importancia
;
;LCD_READAC4:       Obtiene en W el valor de AC.
;
;LCD_WRITEDATO4:    Escritura de datos de W en la DDRAM o CGRAM
;       W = DATO
```

```
;       LCD_DATA <== W
;
;LCD_READDATO4:     Lectura de datos de la DDRAM o CGRAM y lo pasa a W
;       LCD_DATA = DATO
;       W <== LCD_DATA
;
;LCD_TABNUM4:       Convierte el número HEXADECIMAL que este en W a su equivalente ASCII.
;       OjO: No olvidar chequear el PCLATH por tratarse de asignación al PC por istrucción.
;
#define ENABLE      bsf LCD_CTRL,e  ;Activa E
#define DISABLE     bcf LCD_CTRL,e  ;Desactiva
#define LEER        bsf LCD_CTRL,rw ;Pone LCD en Modo RD
#define ESCRIBIR    bcf LCD_CTRL,rw ;Pone LCD en Modo WR
#define DATO        bsf LCD_CTRL,rs ;Desactiva RS (modo comando)
#define COMANDO     bcf LCD_CTRL,rs ;Activa RS (modo dato)
rs equ .1
rw equ .2
e  equ .0

;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;                       SUBRUTINAS DEL LCD
;**************************************************************************************************
LCD_UP4
    bcf INTCON,GIE
    BANCO1
    movlw   0x0F
    andwf   LCD_DATA,f      ;RB <4-7> salidas digitales
    bcf LCD_CTRL,rs
    bcf LCD_CTRL,rw
    bcf LCD_CTRL,e
    BANCO0
    COMANDO                 ;RS=0
    ESCRIBIR                ;R/W=0
    DISABLE                 ;E=0
    return
;
;-------------------------------------------------------------------------

LCD_INI
    movlw   .44
    call    DELAY_MS        ;Espera un poco más de 15ms
    movlw   B'00111000'     ;Datos de 8 bits, 2 lineas y caracteres de 5x7
    call    LCD_COMANDI
    movlw   .33
    call    DELAY_MS        ;5ms > 4.1ms.(Especificacion de Fabricante)
    movlw   B'00111000'     ;Datos de 8 bits, 2 lineas y caracteres de 5x8
    call    LCD_COMANDI
    movlw   .250
    call    DELAY_US        ;110us > 100us.(Especificacion de Fabricante)
    movlw   .250
    call    DELAY_US
    movlw   .250
    call    DELAY_US
    movlw   .52
    call    DELAY_US
    movlw   B'00111000'     ;Datos de 8 bits, 2 lineas y caracteres de 5x8
    call    LCD_COMANDI
    movlw   .250
    call    DELAY_US        ;110us > 100us.(Especificacion de Fabricante)
    movlw   .250
    call    DELAY_US
    movlw   .250
    call    DELAY_US
    movlw   .52
    call    DELAY_US
    movlw   0x20
    call    LCD_COMANDI
    movlw   .2
    call    DELAY_MS
    movlw   B'00101000'     ;Datos de 8 bits, 2 lineas y caracteres de 5x8
    call    LCD_COMAND4
    movlw   B'00000100'     ;Cursor apagado y Display prendido
    goto    LCD_DISCURCONT4

;
;-------------------------------------------------------------------------
```

```
LCD_MENSAJE4
    movwf w2
    num_var_local .2
    movf w2,w
    movwf   INDF        ;Salva posicion de la tabla
M_E_N_S
    movf    INDF,0      ;Recupera posicion de la tabla
    call    LCD_TABLA       ;Busca caracter de salida
    movwf w2
    var2_local
    movf w2,w
    movwf   INDF        ;Guarda el caracter
    movf    INDF,1
    btfss   STATUS,Z        ;¿Hay más caracteres?
    goto LCD_escribe
    var2_local_fin
    fin_subrutina .2
    return
LCD_escribe
    var2_local_fin
    call    LCD_WRITEDATO4  ;Visualiza en el LCD el caracter
    incf    INDF,1      ;Siguiente caracter
    goto    M_E_N_S         ;Repite con siguiente caracter
LCD_TABLA
    movwf   PCL             ;Desplazamiento sobre la tabla
;
;-------------------------------------------------------------------------
LCD_CLEAR4
    movlw   B'00000001'     ;Borra LCD y cursor a Home
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_HOME4
    movlw   B'00000010'     ;Cursor a Home
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_ENTRYMODE4
    andlw   0x03            ;Nos quedamos con b1-b0
    iorlw   0x04            ;Incorporamos el bit de función
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_DISCURCONT4
    andlw   0x07            ;Nos quedamos con b2-b1-b0
    iorlw   0x08            ;Incorporamos el bit de función
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_DISCURSHIF4
    andlw   0x0C            ;Nos quedamos con b3-b2
    iorlw   0x10            ;Incorporamos el bit de función
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_SETCGADDRE4
    andlw   0x3F            ;Nos quedamos con b5-b4-b3-b2-b1-b0
    iorlw   0x40            ;Incorporamos el bit de función
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_SETDDADDRE4
    iorlw   0x80            ;Incorporamos el bit de función
    goto    LCD_COMAND4
;
;-------------------------------------------------------------------------
LCD_READAC4
    num_var_local .1
    call    LCD_BUSY4
    LEER
    BANCO1
    movlw   0xF0
    iorwf   LCD_DATA,f      ;Puerto B como entrada
    BANCO0
    ENABLE
    nop
```

```
    movf    LCD_DATA,0
    DISABLE
    andlw   0x70
    movwf   INDF
    ENABLE
    nop
    movf    LCD_DATA,0
    DISABLE
    andlw   0x0F
    iorwf   INDF,f      ;Valor de la dirección de la DDRAM o CGRAM
    movf    INDF,0
    BANCO1
    movlw   0x0F
    andwf   LCD_DATA,f
    BANCO0
    ESCRIBIR
    fin_subrutina .1
    return
;-----------------------------------------------------------------------
LCD_WRITEDATO4
    movwf w2
    num_var_local .1
    movf w2,w
    movwf   INDF        ;Codigo de comando.
    DATO
    ESCRIBIR
    nop
    movf INDF,w
    andlw 0xF0
    movwf w2
    movf LCD_DATA,w
    andlw 0x0F
    iorwf w2,w
    movwf LCD_DATA
    ENABLE
    nop
    DISABLE
    swapf INDF,w
    andlw 0xF0
    movwf w2
    movf LCD_DATA,w
    andlw 0x0F
    iorwf w2,w
    movwf LCD_DATA
    nop
    ENABLE
    nop
    DISABLE
    nop
    call    LCD_BUSY4
    DATO
    fin_subrutina .1
    return
;-----------------------------------------------------------------------
LCD_READDATO4
    num_var_local .1
    call    LCD_BUSY4
    DATO
    LEER
    BANCO1
    movlw   0xF0
    iorwf   LCD_DATA,1      ;Puerto B como entrada
    BANCO0
    ENABLE
    nop
    nop
    movlw .50
    call DELAY_US
    movf    LCD_DATA,0
    nop
    DISABLE
    andlw   0xF0
    movwf   INDF
    ENABLE
    nop
    movlw .50
```

```
    call DELAY_US
    swapf   LCD_DATA,0
    nop
    DISABLE
    andlw   0x0F
    iorwf   INDF,1      ;Valor leido de la DDRAM o CGRAM
    BANCO1
    movlw   0x0F
    andwf   LCD_DATA,f
    BANCO0
    ESCRIBIR
    COMANDO
    movf INDF,0
    movwf w2
    fin_subrutina .1
    movf w2,w
    return
;-----------------------------------------------------------------------
LCD_TABNUM
    addwf   PCL,1
    dt  "0123456789ABCDEF"

;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;               SUBRUTINAS INTERNAS DEL LCD
;***********************************************************************************************
LCD_BUSY4
    num_var_local .1
    BANCO1
    movlw   0xF0
    iorwf   LCD_DATA,f      ;Puerto B como entrada
    BANCO0
    COMANDO
    LEER
    nop
otro_busy
    ENABLE
    nop
    movf    LCD_DATA,0
    nop
    DISABLE
    movwf   INDF
    ENABLE
    nop
    movf    LCD_DATA,0
    DISABLE
    andlw 0x0F
    iorwf INDF,1
    btfsc   INDF,7
    goto    otro_busy
    BANCO1
    movlw   0x0F
    andwf   LCD_DATA,f
    BANCO0
    ESCRIBIR
    fin_subrutina .1
    return
;-----------------------------------------------------------------------
LCD_E
    ENABLE
    nop
    DISABLE                 ;Desactiva E
    COMANDO
    return
;
;-----------------------------------------------------------------------
;LCD_COMANDI:   Escritura de comandos del LCD
;       W = Codigo de comando para el LCD
;       W ==> LCD_DATA
LCD_COMANDI
    movwf   LCD_DATA        ;Codigo de comando.
    call LCD_E
    return
;-----------------------------------------------------------------------
;LCD_COMAND4:       Escritura de comandos del LCD
;       W = Codigo de comando para el LCD
;       W ==> LCD_DATA
```

```
LCD_COMAND4
    movwf w2
    num_var_local .1
    movf w2,w
    movwf   INDF        ;Codigo de comando.
    COMANDO
    ESCRIBIR
    movf INDF,0
    andlw 0xF0
    movwf w2
    movf LCD_DATA,w
    andlw 0x0F
    iorwf w2,w
    movwf LCD_DATA
    nop
    ENABLE
    nop
    DISABLE
    swapf INDF,0
    andlw 0xF0
    movwf w2
    movf LCD_DATA,0 ;a prueba
    andlw 0x0F ;a prueba
    iorwf w2,w ;a prueba
    movwf LCD_DATA
    nop
    ENABLE
    nop
    DISABLE
    call    LCD_BUSY4
    COMANDO
    fin_subrutina .1
    return
;
;-------------------------------------------------------------------------
;LCD_FUNCIONSET4:   Configura ela fuente del caracter y número de lineas. No se cambia DL
;       El modo de control deseado deberá cargarse en los 2 bits terceros de W:
;               x x x x b3 b2 x x
;       b2 :    0 = caracter de 5x7         1 = de 5x10
;       b3 :    0 = display de 1 linea      1 = de 2 lineas
;       b0,1,4-7 : sin importancia
LCD_FUNCIONSET4
    andlw   0x0C            ;Nos quedamos con b3-b2 (DL no se cambia)
    iorlw   0x30            ;Incorporamos el bit de función y 8 bits
    goto    LCD_COMAND4
```

```
num_var_local macro cantidad
    movlw cantidad
    addwf puntero,f
    movf puntero,w
    movwf FSR
    endm
fin_subrutina macro cantidad
    movlw cantidad
    subwf puntero,f
    movf puntero,w ;
    movwf FSR ;
    endm

var1_local macro
    endm
var1_local_fin macro
    endm

var2_local macro
    decf FSR,f
    endm
var2_local_fin macro
    incf FSR,f
    endm

var3_local macro
    decf FSR,f
    decf FSR,f
    endm
var3_local_fin macro
    incf FSR,f
    incf FSR,f
    endm

var4_local macro
    decf FSR,f
    decf FSR,f
    decf FSR,f
    endm
var4_local_fin macro
    incf FSR,f
    incf FSR,f
    incf FSR,f
    endm

var5_local macro
    decf FSR,f
    decf FSR,f
    decf FSR,f
    decf FSR,f
    endm

var5_local_fin macro
    incf FSR,f
    incf FSR,f
    incf FSR,f
    incf FSR,f
    endm

graba_eeprom macro direc,dato
    BANCO2
    movlw direc
    movwf EEADR
    movlw dato
    movwf EEDATA
    call escribir_EE
    endm

lee_eeprom macro direc
    BANCO2
    movlw direc
    movwf EEADR
    call leer_EE
    endm

BANCO0 macro
```

```
    bcf STATUS,RP0
    bcf STATUS,RP1
    endm
BANCO1 macro
    bsf STATUS,RP0
    bcf STATUS,RP1
    endm
BANCO2 macro
    bcf STATUS,RP0
    bsf STATUS,RP1
    endm
BANCO3 macro
    bsf STATUS,RP0
    bsf STATUS,RP1
    endm

multiplicar macro num1,num2
    movlw .2
    movwf cont
    movf num1,0
    movwf dato_g
añade_mas
    movf dato_g,0
    addwf num1,1
    movlw num2
    xorwf cont,0
    btfsc STATUS,Z
    goto $+3
    incf cont,1
    goto añade_mas
    movf num1,0
    movwf dato_1
    endm

retardo_us macro valor
    movlw valor
    call DELAY_US
    endm
retardo_ms macro valor
    movlw valor
    call DELAY_MS
    endm
retardo_s macro valor
    movlw valor
    call DELAY_S
    endm
mover macro var1,var2
    movf var1,w
    movwf var2
    endm
mover_num macro var1,var2
    movlw var1
    movwf var2
    endm
mover_local macro local1,local2
    movf puntero,w
    addlw local1
    movwf FSR
    movf INDF,w
    movwf aux
    movf puntero,w
    addlw local2
    movwf FSR
    movf aux,w
    movwf INDF
    endm
mover_local_w macro local1
    movf puntero,w
    addlw local1
    movwf FSR
    movf INDF,w
    endm
mover_num_local macro numero,local1
    movf puntero,w
    addlw var1
    movwf FSR
```

```
    movlw numero
    movwf INDF
    endm


compara macro var1,var2,respuesta ;respuesta=0 (iguales)
    movf var1,w   ;respuesta=1 (diferentes)
    xorwf var2,w
    btfss STATUS,Z
    goto $+3
    clrf respuesta
    goto $+3
    movlw 0x01
    movwf respuesta
    endm

menor macro var1,var2,respuesta  ;respuesta=0 (menor)
    movf var2,w  ;respuesta=1 (mayor o igual)
    subwf var1,w
    btfsc STATUS,C
    goto $+3
    clrf respuesta
    goto $+3
    movlw 0x01
    movwf respuesta
    endm

convertir macro deci_h,deci_l,dig0,dig1,dig2,dig3,dig4
    clrf    dig0
    clrf    dig1
    clrf    dig2
    clrf    dig3
    clrf    dig4
    swapf   deci_h,w
    iorlw   0xf0
    movwf   dig3
    addwf   dig3,f
    addlw   0xE2
    movwf   dig2
    addlw   0x32
    movwf   dig0
    movf    deci_h,w
    andlw   0x0F
    addwf   dig2,f
    addwf   dig2,f
    addwf   dig0,f
    addlw   0xE9
    movwf   dig1
    addwf   dig1,f
    addwf   dig1,f
    swapf   deci_l,w
    andlw   0x0F
    addwf   dig1,f
    addwf   dig0,f
    rlf     dig1,f
    rlf     dig0,f
    comf    dig0,f
    rlf     dig0,f
    movf    deci_l,w
    andlw   0x0F
    addwf   dig0,f
    rlf     dig3,f
    movlw   0x07
    movwf   dig4
    movlw   0x0A
Lb1
    addwf   dig0,f
    decf    dig1,f
    btfss   3,0
    goto    Lb1
Lb2
    addwf   dig1,f
    decf    dig2,f
    btfss   3,0
    goto    Lb2
Lb3
```

```
    addwf   dig2,f
    decf    dig3,f
    btfss   3,0
    goto    Lb3
Lb4
    addwf   dig3,f
    decf    dig4,f
    btfss   3,0
    goto    Lb4
    endm
```

```
multiplicar_por_10;multiplica w*10->w
    movwf w2
    num_var_local .3
    movf w2,w
    movwf INDF
    var3_local
    movlw .2
    movwf INDF
    var3_local_fin
    movf INDF,0
    movwf w2
    var2_local
    movf w2,w
    movwf INDF
    var2_local_fin
añade
    var2_local
    movf INDF,0
    movwf w2
    var2_local_fin
    movf w2,w
    addwf INDF,f
    var3_local
    movlw .10
    xorwf INDF,0
    btfss STATUS,Z
    goto conti_multi10
    var3_local_fin
    movf INDF,w
    movwf w2
    fin_subrutina .3
    movf w2,w
    return
conti_multi10
    incf INDF,1
    var3_local_fin
    goto añade

dividir_entre_4

    cblock conver_var
    decimal
    bcd_h
    bcd_l
    endc

conversion_bcd_2bytes; convierte un byte en bcd de 2bytes ascci
;el número se carga previamente en w y el resultado se devuelve
;en bcd_h:bcd_l
    movwf decimal
    movlw 0x30
    movwf bcd_l
    movwf bcd_h
    movf decimal,f
    btfsc STATUS,Z
    return
incrementa_lsb
    incf bcd_l,1
    movlw 0x3A
    xorwf bcd_l,0
    btfss STATUS,Z
    goto pregunta_bcd
incrementa_msb
    movlw 0x30
    movwf bcd_l
    incf bcd_h,1
pregunta_bcd
    decfsz decimal,f
    goto incrementa_lsb
    return
```

```
USART_UP
    BANCO1
    movlw   0xc0                ;set tris bits for TX and RX
    iorwf   TRISC,F
    movlw   (FOSC/(.16*BAUD))-1 ;set baud rate
    movwf   SPBRG
    movlw   0x24
    movwf   TXSTA               ;enable transmission and high baud rate
    BANCO0
    movlw   0x90                ;enable serial port and reception
    movwf   RCSTA
    return

TXWSERIAL
    BANCO1
    btfss   TXSTA,TRMT
    goto    $-1
    BANCO0
    movwf   TXREG               ;transmit the data
    return
```