UNIVERSIDAD NACIONAL AUTONOMA
DE MEXICO

FACULTAD DE INGENIERIA

ADQUIRIDOR DE DATOS PARA CONTROL DE VEHICULOS

T E S I S

QUE PARA OBTENER EL TITULO DE:
INGENIERO MECANICO ELECTRICISTA
P R E S E N T A N :
CORAS GARCIA RONALD MARTIN
SOSA LUJANO GILBERTO ALEJANDRO

FACULTAD DE INGENIERIA
U N A M

CD. UNIVERSITARIA, D. F. 1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

¡Mira a este día!

Porque es la vida, la mismísima vida de la vida.

En su breve curso
están todas las verdades y realidades de tu existencia:

La bendición del desarrollo,
la gloria de la acción,
el esplendor de las realizaciones...

Porque el ayer es sólo un sueño
y el mañana sólo una visión,

perc el hoy bien vivido hace de todo ayer un sueño
de felicidad
y de cada mañana una visión de esperanza.

¡Mira bien, pues, a este día!

Tal es la salutación del alba.

Kalidasa

Este trabajo es fruto del apoyo de grandes personas:

Queremos agradecer al Ing. Ernesto Suárez Sport su enorme paciencia e inmenso apoyo durante el desarrollo de este trabajo de tesis: gracias por su confianza.

Agradecemos a la M. en C. Amanda O. Gómez González por su amistad, su invaluable ánimo e incondicional apoyo a lo largo de todo este trabajo (y en nuestra vida).

También deseamos agradecer su tiempo, apoyo y ejemplo al Ing. Javier Gómez Castellanos, ya que sin su ayuda, este trabajo no hubiera sido el mismo.

Gracias al Ing. Raymundo Hugo Rangel por las facilidades prestadas para la realización de nuestro trabajo.

Igualmente, agradecemos al Ing. Román Osorio Comparán su cooperación y todas las facilidades proporcionadas.

Emitimos un profundo agradecimiento al sr. José y también al demás personal que labora en la División de Estudios de Posgrado de la Facultad de Ingeniería.

De la misma forma, agradecemos al personal que labora en "la puerta 4", y al personal de "Reparación de velocímetros", todo su apoyo, tiempo y explicaciones que tuvieron a bien brindarnos.

Deseamos agradecer a los profesores de la Facultad de Ingeniería, que nos hayan demostrado que la U.N.A.M. es sencillamente ¡la mejor!.

A Dios Todopoderoso,
por la oportunidad de
vivir...

A la persona que más admiro,
ya que de él soy lo que soy,
por que llevo su ejemplo en mí,
y me ha enseñado a ser un hombre;
a mí padre, Jesús.

A la mujer más maravillosa
de la Tierra, por todo su
cuidado,y por enseñarme
lo que debe ser una mujer;
a mi madre, Delia.

A ellos, que me dieron todo
cuando más lo necesitaba,
y me han dado la felicidad...

Gilberto

Deseo pronunciar un profundo agradecimiento a cinco personas que tuvieron gran influencia en mi vida durante mi desarrollo profesional:

A Luis, por su ejemplo.

A Reyna María, que me ha atendido como una amiga y madre; gracias por todo su cariño y preocupación.

A José Luis, por todas sus atenciones y amistad.

A Miriam, por su ternura, inocencia y dulzura; por ser una niña muy linda.

A Chary, por hacer que mi vida universitaria fuera singular, y envidiablemente divertida; no la cambiaría por nada; ¡Te amo!

A la familia Barragán Paz, por adoptarme como un miembro más, muchas gracias...Gilberto.

A Leonardo, por ser un ejemplo de fortaleza y conocimiento para mí.

A Luis, por todo lo que hemos vivido (y bien vivido), (¡ y lo que falta !). Eres especial, muy especial.

A la niña más linda, alegre, simpática, sincera, cariñosa, ingeniosa, optimista, amena, por su confianza, cooperación, apoyo en los momentos tristes y alegres; por sus alegrías y enojos; por ser una persona fantástica (hermosa), la mejor amiga y novia que un hombre puede tener: a Chary.

A Ronald, por soportarme; por involucrarme en este trabajo y por ser un compañero de tesis singular (me divertí mucho).

A Ana e Itha, por que son grandiosas. A Jorge y Poncho, por su entusiasmo. A Lalo, Fernando y Gerardo, por todo su apoyo, su sinceridad y su confianza. Por que en ellos veo el ejemplo de un trabajo bien hecho (y la confianza de un amigo). A Juan Carlos, Oscar, Victor, Jorge, Angel y demás miembros de MASA, por soportarme, en las buenas y en las malas.

A Alicia, Tomy, Aurora, Fernanda y Luis, por tantos ratos agradables (¡que sean más!). Gracias por su paciencia.

A mis padrinos José Luis, Wilma, Manuel y Maruca.

A todo el personal de UNIVERSUM (especialmente a la sala de Matemáticas), por su amistad, comprensión, confianza y apoyo incondicional. ¡Adelante Ale!

A Mauricio, Juan Manuel, Marce, Adriana, Saúl, Alejandro, Gabriel, y todos los demás becarios y personal (Zena, GÜerita) de la D.C.A.A., por que son un ejemplo para mí. (¡Gracias Octavio!).

Al Dr. Cornelio Robledo, y a Victor, Jorge, Angel, Alejandro, Raúl, Pepe, Toño, Chava, Raúl, Raymundo, Ingrid, Concha, (y tanto personal de Banco de México que me falta mencionar), por darme uno de los momentos más felices de mi vida: sentirme útil. ¡Gracias Gabriel, gracias Isaías!

A Chary, Bety, José, Ricardo, Fausto, Fernando, Felipe, Fabiola, Victor, Celso y demás compañeros de la Facultad de Ingeniería, por ser los mejores estudiantes de toda la Universidad. A Tony, Mauricio "Baeza", Jesús, René y Felix "Coria", por su amistad incondicional durante tanto tiempo.

A Mundo, Alberto, Mauricio, Alejandro, "Pelón", y demás amigos del Condominio del Valle; por enseñarme a ser niño. ¡Salud, Joaquín!.

Por que, sin lugar a dudas, tuve la fortuna de rodearme de personas más inteligentes que yo... Gracias.... Gilberto.

A Dios

A mis abuelos
por todo su amor.

A mis padres
por todo su amor apoyo y paciencia.

A mis hermanos
Marina
Felipe
Joaquin
Diana
Paola
por todo lo que hemos compartido juntos.

A mis tíos y primos
por todo lo que han representado en mi formación como persona.

A mis cuñados
por su amistad.

A mi padrino Dr. Pablo Perez y Fuentes
por su apoyo y consejos.

A Gilberto
por haber sido un gran compañero de tesis, por su apoyo y amistad.

A todos los miembros de MASA:
Chary
Fernando
Gerardo
Eduardo
Alfonso
por su amistad.

A mis amigos y compañeros, especialmente a:
Alejandro Gomez
Manuel Morales
Marcela Cabral
David Rivas
Norman Maza
Francisco Gutierrez
Cesar Acosta
Norma Eneida
Sergio Rosas
Araceli Del Olmo
Victor Vega
Esther
Matilde
Abel
por su amistad, apoyo y por todo el tiempo que hemos compartido.

A todos los amigos y compañeros del CECAFI porque siempre me dará gusto saludarlos y recordar el tiempo que pase con ustedes.

A mis exalumnos de informática, especialmente a Eva porque, aunque no lo sabes, me has ayudado mucho a superarme como persona.

A Olga por su apoyo y comprensión.

Ronald Coras García

A Leonardo

Mektoub

ADQUIRIDOR DE DATOS PARA EL CONTROL DE VEHICULOS

Capítulo 1

Introducción

El inicio de lo que podríamos denominar la electrónica actual, se dió con la creación del primer dispositivo fabricado con material semiconductor: el transistor, desplazando casi completamente a su antecesor, el bulbo. El transistor abrió un nuevo y amplio panorama, y provocó la "revolución" más grande que cualquier experto en electrónica hubiese imaginado, ya que sería posible construir diversos circuitos electrónicos a partir de un arreglo ordenado de transistores, y a su vez, construir una enorme cantidad de sistemas con el empleo de dichos circuitos.

En un principio los transistores eran "grandes", y por ende, los circuitos electrónicos también lo eran. Sin embargo, el gran avance que vive la tecnología de fabricación de circuitos integrados, ha permitido el desarrollo de sistemas electrónicos cada vez más eficientes, pequeños y económicos. Ahora es posible poseer un circuito conformado por miles de transistores, guardado en un encapsulado del tamaño de un paquete de cerillos.

Uno de los circuitos en el que se ha aprovechado al máximo dicho desarrollo es el microprocesador. Un microprocesador es, en forma básica, un dispositivo capaz de ejecutar automáticamente una secuencia de operaciones sobre unos datos proporcionados; el propósito de tales operaciones puede ser la resolución de problemas matemáticos, el control de ciertas funciones o de otros dispositivos, etc. Se puede decir también que un microprocesador es un conjunto de circuitos combinacionales y secuenciales, que se encarga de desarrollar una serie de operaciones aritméticas, lógicas (ejecutando un conjunto de instrucciones especificadas en un programa), y de controlar otros circuitos con los que interactúa, constituyendo así la parte primordial de lo que hoy conocemos como una computadora.

Básicamente, un microprocesador está formado por: una unidad central de procesos (que se encarga de efectuar las operaciones

lógicas, aritméticas, y de control); una memoria (que es una unidad de almacenamiento de información temporal (datos)); y, una unidad de entrada/salida (que comunica al microprocesador con dispositivos externos al mismo). Algunos dispositivos externos son: los contadores, temporizadores, convertidores analógico/digitales, puertos de entrada/salida en serie o paralelo, memoria, etc. La elección de los dispositivos a usar dependerá de la aplicación que se le asigne al microprocesador.

Con el avance tecnológico se buscó fabricar circuitos que por sí mismos pudiesen efectuar completamente las operaciones básicas de un computador, requiriendo un número mínimo de componentes externos, de ahí que surgieran los microcontroladores, que son circuitos que contienen un microprocesador y un mínimo de dispositivos auxiliares, todos integrados en un solo paquete o encapsulado, reduciendo en forma considerable el tamaño total del circuito electrónico, y haciendo su manejo más sencillo al disminuir el número de conexiones requeridas.

El hecho de disponer de una capacidad de cómputo con bajo consumo de energía, bajo costo, volumen y peso pequeños, permitió la incorporación de equipo electrónico en aplicaciones tales como: la aeronáutica, la satelital, la industrial, la médica, etc. Este avance se introdujo también al desarrollo automotriz. Hoy es común encontrar vehículos cuyo funcionamiento es regulado por un microcontrolador, supervisando el correcto trabajo del motor, brindando al usuario comodidad, seguridad y un mayor dominio tanto de la conducción del automóvil como del comportamiento de la máquina, y hasta de su mantenimiento.

Otra aplicación para los circuitos electrónicos es, por ejemplo, en empresas que manejan flotas de transporte, en donde resulte indispensable tener un registro del uso que se le dá a los vehículos, del comportamiento de la máquina en marcha, a fin

de poder evaluar al personal que tiene a cargo el manejo de dicho equipo automotor. Con este fin se han desarrollado en el mercado diferentes sistemas de monitoreo electrónico; sin embargo, considerando el avance apresurado de la tecnología, los nuevos equipos pueden llegar a ser obsoletos tan pronto, que las proyecciones de recuperación de inversión y rentabilidad del equipo ya no son tan confiables.

El objetivo del presente trabajo es el diseño y la construcción de un sistema de monitoreo electrónico, de bajo costo. Además, presenta la posibilidad de ser expandible, a fin de diversificar su uso y adaptarse a necesidades presentes y futuras, sin que ésto se refleje en un incremento significativo de su precio.

Capítulo 2

Descripción de la Necesidad

Para establecer claramente los objetivos que debe cubrir el sistema adquiridor de datos, es conveniente conocer las características de otros sistemas que se encuentran disponibles en el mercado.

2.1 DESCRIPCIÓN DE LOS SISTEMAS DE MEDICIÓN CONVENCIONALES.

Al decir sistemas de medición convencionales, se quiere describir a toda la serie de sistemas y/o aparatos que se utilizan comúnmente para el registro de tiempo, distancia, etc., concerniente a un vehículo.

Un medidor con el que cuentan todos los vehículos es el velocímetro/odómetro y está conectado a un chicote proveniente de la transmisión o de la caja de velocidades. Al girar el chicote, se hace girar a un imán permanente, localizado en el interior del velocímetro. Alrededor del imán se encuentra el tambor de velocidad. La manecilla del velocímetro está fijada al tambor (no existe una conexión directa entre el chicote y la aguja del velocímetro; es más bien una conexión magnética). Conforme gira el imán, también gira su campo magnético, produciendo corrientes de eddy que ocasionan que el tambor gire hacia el sentido del imán. A mayor velocidad del vehículo, mayor número de vueltas dará el imán permanente, y mayor será la deflexión de la aguja del velocímetro. Para completar el instrumento, se le pone una carátula que está calibrada de acuerdo con la velocidad. El odómetro sí está conectado directamente al chicote, consiste por lo general de 6 ruedas (donde cada una tiene los números del 0 al 9), y se utiliza para indicar la distancia que recorre el vehículo.

Más recientemente se han colocado en el mercado velocímetros/odómetros digitales, para realizar la misma cuenta, pero mediante circuitería electrónica.

Otro sistema de medición muy conocido es el taxímetro, aparato utilizado ampliamente por los taxistas para registrar la distancia que recorre un vehículo, el tiempo que tarda en hacerlo, y así proporciona el precio por cobrar al pasajero. Industrias Farhnos de México es la compañía que más fuerza presenta en el ámbito de los taxímetros. Esta industria también ha desarrollado un equipo enfocado a la medición de distancia recorrida por un vehículo; sin embargo, es un sistema muy limitado en su capacidad de monitoreo y registro de variables, presenta un complejo sistema de claves de acceso y además, muestra la información en displays, en forma de códigos, lo que hace incómoda su lectura.

Existe otro sistema denominado tacógrafo, que registra la velocidad de un vehículo a lo largo de toda una ruta. Es un sistema al cual se le instala un disco de papel; durante el manejo del vehículo, el disco gira y una aguja con tinta se encarga de escribir en él para registrar todas las variaciones de velocidad que sufre el vehículo. Queda como resultado una línea similar a la producida por un sismógrafo, (o por un encefalograma), pero siguiendo una trayectoria en espiral (y no recta), y de dicha línea se puede obtener las velocidades del vehículo, y al mismo tiempo, la distancia recorrida.

2.2 OBJETIVOS DEL SISTEMA ADQUIRIDOR DE DATOS.

El sistema debe tener por objetivo registrar una serie de eventos en su memoria, lo que permitirá desarrollar un análisis retrospectivo con el fin de obtener una evaluación del uso que se dá a un vehículo. Este objetivo debe apreciarse desde el punto de vista de beneficios para una empresa o propietario de flotas de transporte, registrando en todo momento el kilometraje recorrido y el tiempo empleado en un itinerario determinado. Durante el recorrido de la ruta se deberá tomar en cuenta la velocidad del

vehículo, a fin de detectar si éste circula a una velocidad mayor que la permitida. La velocidad máxima permitida debe ser un parámetro programable que el cliente determine, para adecuar el sistema a sus necesidades particulares.

Además, deberá estar diseñado para permitir, si las necesidades del servicio lo requieren, el control de paradas y destinos intermedios dentro de su ruta, lo cual se puede lograr dividiendo una ruta en tramos, registrando el kilometraje recorrido y el tiempo empleado en cada tramo de la ruta por separado.

El sistema deberá ser versátil, queriendo decir con ésto, que pueda adecuarse a cualquier vehículo de transporte con relativa facilidad. También es necesario que el sistema presente como resultado final un registro de todos los parámetros medidos, y de preferencia que sea en forma escrita, para tener un documento que conserve y compruebe la realización de las mediciones.

El diseño final deberá ser muy sencillo de usar, cómodo para el conductor, de tamaño compacto, ligero (no más de 1 kg de peso), y de bajo costo. Por otra parte, deberá estar listo para incorporarle el registro de nuevos parámetros, aportando mayor utilidad para el usuario.

Capítulo 3

El diseño en Ingeniería

La tarea esencial del ingeniero consiste en encontrar soluciones a problemas técnicos, basadas en el conocimiento de las ciencias naturales, y realizarlas de una manera óptima, tomando en cuenta las limitaciones establecidas en su momento histórico a partir del material, de la tecnología y la economía. Sus ideas, conocimientos y aptitudes determinan de manera decisiva al producto y la economía entre el productor y el consumidor. Se puede definir al diseño como la realización conceptual tendiente a resolver un problema, que trata de satisfacer de la mejor forma todas las exigencias establecidas, tomando en cuenta las posibilidades según la época. Esto se traslapa con todas las actividades humanas, se sirve de los conocimientos y leyes de las ciencias naturales y alcanza una realización material.

A continuación se detallan los asuntos concernientes al diseño de un sistema electrónico, enfocado más propiamente al uso de un microprocesador en su construcción.

3.1 PRINCIPIOS BÁSICOS DEL DISEÑO.

Los principios básicos para diseñar con microprocesadores son:

Modularidad	Es la habilidad o posibilidad de dividir un problema en problemas más simples, donde cada uno de esos problemas cuenta con una solución específica.
Homogeneidad o Regularidad	Es la posibilidad que tienen los submódulos de un problema de ser lo más semejantes entre sí.

Conectividad		Posibilidad de poder comunicar adecuada y eficientemente los datos que existen entre los diferentes módulos de un sistema, garantizando la integridad de la inforamción.
Localidad	Medio ambiente interno	Habilidad de relacionar los diferentes módulos de un sistema para que trabajen correctamente.
	Medio ambiente externo	Habilidad que tiene el diseñador para ubicar el diseño en el lugar en el que va a trabajar.

Estos 4 principios se deben tener en mente al momento de realizar el diseño, de manera que se presente la solución más adecuada, de acuerdo con las condiciones del problema.

3.2 ASPECTOS DEL DISEÑO.

Los aspectos que afectan al diseño siempre resultan importantes, y conviene que el diseñador los considere siempre. Algunos son:

*) Costo, *) cantidad de datos, *) aplicación, *) tamaño físico, *) medio, *) velocidad, *) tecnología, *) periféricos, *)compatiblidad, *)complejidad, *) sistema de seguridad, *)alimentación, *) mantenimiento.

Además de los aspectos que afectan al diseño, debemos considerar los problemas que se presentan con más frecuencia en el diseño de un sistema electrónico. Estos problemas son:

*)radiofrecuencia; *)generación de armónicas; *)ruído; y *)diferenciales de potencial, entre otros.

Otros aspectos que también intervienen en el diseño son:

- Rango (intervalo en el que fluctúa una variable) y resolución de las variables.
- Medio ambiente de operación.
- Aspectos ergonómicos: adecuación de una máquina para ser usada por una persona.
- Rapidez de variación de las variables.
- Precisión (obtención de lecturas semejantes en mediciones semejantes) y exactitud (obtención de mediciones cercanas a la realidad).
- Resolución: es la fineza de la variación; normalmente se asocia con la cantidad de niveles que maneja: a mayor resolución, mayor uso de memoria.
- Mantenimiento (preventivo y correctivo); conviene tomar en cuenta: puntos y patrones de prueba y de calibración.

3.3 NIVELES DE DISEÑO.

Existen 3 niveles de diseño: a nivel sistema, a nivel subsistema y a nivel componente. Para el diseño a nivel componente, se selecciona una familia de circuitos integrados para resolver un problema específico; éste fué el nivel considerado para el desarrollo del presente trabajo.

3.4 METODOLOGÍA DE DISEÑO.

La metodología es una serie de pasos sugerentes a seguir para el desarrollo de un diseño. Seguir un método de solución de problemas en la ingeniería de diseño, integrado al proceso de

creación de productos, permite manejar una herramienta que nos lleva a converger en soluciones óptimas. La metodología es un enfoque sistemático (organizado), y existen 2 grandes tendencias en el diseño de circuitos electrónicos:

Diseño funcional ascendente (Bottom-Up)	Se usa para problemas muy grandes o para macrosistemas; es para fines de integración de sistemas.
Diseño funcional descendente (Top-Down)	Se usa para problemas o sistemas puntuales (específicos). Parte de un problema puntual y llega a una solución puntual.

La metodología de diseño es susceptible de ser programada, aplicándose entonces herramientas como CAD, CAM Y CAE (orientados al desarrollo de ingeniería, tanto en el aspecto de cálculos como en el de simulaciones).

Para el sistema que nos concierne, resulta más conveniente seguir el diseño funcional descendente. Dicha metodología está constituída por los siguientes pasos:

1.- Planteamiento del problema.- es la etapa donde se comprenden las necesidades y la dificultad particular que lo originan, con la intensión de clarificar el origen, y el término del proyecto para dirigir correctamente los esfuerzos del diseño.

2.- Delimitación de los aspectos.
   a) Hardware: Eléctrico, electrónico, etc.
   b) Software: Programas, algoritmos, etc.

3.- Selección de las familias lógicas que van a intervenir en el diseño (familias lógicas tanto en hardware como en software). Es importante recordar que el hardware y el software se tratan en paralelo (al mismo tiempo).

4.- Planteamiento general de la estructura interna del sistema (organización). Aquí se aplica el principio de modularidad.

5.- Planteamiento de la solución específica de cada módulo. El hardware y el software de cada módulo se tratan siempre en forma paralela.

6.- Pruebas de campo para cada módulo.

7.- Integración de módulos. Aquí se verifica si los principios de modularidad y conectividad se aplicaron correctamente.

8.- Pruebas generales.

9.- Características operativas del prototipo.

3.5 CRITERIOS PARA EL RECONOCIMIENTO DE UN MICROPROCESADOR.

Es la información mínima necesaria para conocer bien a un microprocesador. Requerimos de ciertos criterios para tener una idea del trabajo que el microprocesador está acostumbrado a desempeñar, y lo que podemos esperar al usarlo. Dicha información está constituída por:

1.- Arquitectura.
2.- Conjunto de instrucciones.
3.- Modos de direccionamiento (SW).
4.- Señales del procesador.
5.- Manejo de memoria (HW).

6.- Manejo de puertos.
7.- Manejo de interrupciones.

3.6 CRITERIOS DE SELECCIÓN DE UN MICROPROCESADOR.

Una vez que se han reconocido varios microprocesadores, requerimos una serie de criterios que nos permitan seleccionar aquel que se adecúe más con el trabajo que deberá desempeñar. Los criterios de selección son:

1.- Definición del uso del procesador.
    * uso= propósito general o específico.

2.- Definición de la resolución que requiere el sistema.

3.- Capacidad de direccionamiento requerida.

4.- Velocidad de procesamiento.

5.- Definición de los requerimientos de los puertos.

6.- Definir el manejo de interrupciones que se requiere.

7.- Establecer la arquitectura que requiere el sistema (grado de paralelismo y pipeline).

8.- Recursos disponibles para desarrollo (posibilidad de expansión).

Es necesario recordar que la solución óptima a un problema no es siempre aquella que colinda con la frontera tecnológica contemporánea, sino aquella que contempla el mejor empleo de los recursos disponibles para resolver el problema particular.

Capítulo 4

Recopilación y análisis de la información.

Se decidió registrar la información en forma digital, ya que resultaba la manera más apropiada de almacenarla y de manejarla con un sistema programable. Una vez establecido ésto, se estudiaron y plantearon diferentes opciones sobre los componentes que formarían al sistema.

De acuerdo con los objetivos planteados, se requería un sistema electrónico digital que incluyera un CPU, registros, ALU, banderas así como la circuitería de reloj necesaria para establecer el oscilador de trabajo; todo ésto integrado en un chip, para lograr un manejo confiable de la información, en un espacio lo más reducido posible. Este sistema debería ser desarrollado en torno a un microcontrolador que reuniera las características requeridas, tomando como referencia los criterios para la selección de microprocesadores, ya establecidos.

Además se requería de dispositivos periféricos tales como:

+) Una memoria ROM que almacenará el programa a ser ejecutado por el procesador.

*) Una memoria RAM, ya que el sistema requeriría de espacio para leer y escribir la información procesada, o proveniente del exterior, ya sea dada por el usuario, o aportada por los sensores integrados al sistema. La memoria RAM debería tener la característica de mantener almacenada la información de sus registros, aún en ausencia de la fuente de energía del sistema.

+) El adquiridor debería operar tomando en cuenta la fecha y hora de ocurrencia de los eventos, por lo que también se necesitaría de un reloj y un calendario electrónicos (reloj en tiempo real) que pudieran ser trabajados fácilmente por el sistema. Esta parte debería ser respaldada por una circuitería que mantuviera operando al reloj y al calendario de manera permanente, con el fin de que la información se conservara actualizada. La operación

de esta circuitería debería monitorear permanentemente Vcc, de manera que cuando el sistema fuera despolarizado, se conectara automáticamente una batería de respaldo. Además, sería recomendable que el consumo de energía de la circuitería del reloj y del calendario fuera bajo, cuando el sistema estuviera sin energía de fuente, ésto es, cuando el equipo no se estuviera usando.

*) Para lograr lo anterior, sería necesario usar baterías de respaldo de larga duración, ya que se pretende que el equipo tenga una vida útil mínima de 5 años.

+) Se necesitaría un sistema de puertos que permitiera la interacción con el usuario y el registro de información de los sensores, todo al mismo tiempo.

*) Una forma conveniente de interactuar con el sistema, es a través de un display y un teclado. El teclado debería ser reducido y sencillo, fácil de manejar por el usuario, y económico. El tamaño y tipo del teclado dependería de su aplicación, que es: seleccionar las opciones que ofrece el sistema (tales como cambiar el código de acceso, impresión de datos o programar la velocidad máxima, etc.). Según la opción elegida se procedería igualmente, desde el teclado, a dar o solicitar información al sistema.

+) El display debería visualizar la información que ofrece el sistema, ya sea para solicitar algún dato al usuario, proporcionándolo si se le pide con el teclado, o indicando alguna operación o condición del equipo.

4.1 RECOPILACIÓN DE INFORMACIÓN

Una vez definidas las características que deberían cumplir los componentes integrantes del sistema, se procede ahora a comentar los circuitos más aptos para formar parte del mismo.

4.1.1 MICROCONTROLADORES.

Entre los microcontroladores destacan principalmente los dispositivos electrónicos de las firmas INTEL y MOTOROLA. También encontramos microcontroladores de la marca ZILOG; sin embargo, en la actualidad no presenta una familia tan extensa como la que proporcionan las dos primeras; además, los microcontroladores de INTEL y MOTOROLA pueden adaptarse fácilmente al sistema requerido.

A continuación se describen las características más importantes del microcontrolador de MOTOROLA que se encuentra más usado en la actualidad: el MC68HC11A8.

4.1.1.1 MC68HC11A8

Alta densidad complementada al semiconductor metal-óxido (HCMOS) hacen del MC68HC11A8 un avanzado microcontrolador de 8 bits, con una muy alta y sofisticada capacidad periférica integrada. Se usaron nuevas técnicas de diseño para lograr una velocidad de bus nominal de 2 MHz. Además, su diseño completamente estático le permite operar en frecuencias bajas, permitiendo un ahorro importante en el consumo de potencia. La tecnología HCMOS usada en el MC68HC16AB combina el tamaño más pequeño con velocidad más alta, bajo consumo y la alta inmunidad al ruído de CMOS. Integrado al chip se incluyen: 8 kbytes de memoria de sólo lectura ROM, 512 bytes de EEPROM, y 256 bytes de RAM.

Se incluye un convertidor de 8 canales, con 8 bits de resolución, una interfase de comunicación serial asíncrona (SCI) y una interfase periférica serial síncrona separada (SPI). El temporizador principal del sistema, de carrera libre de 16 bits, tiene 3 líneas de entrada captura, 5 líneas de comparación de salida, y una función de interrupción en tiempo real. Un subsistema acumulador de pulsos de 8 bits puede contar eventos o medir períodos externos. Se incluye una circuitería de monitoreo de seguridad dentro del chip para protección contra errores del sistema.

7	ACUMULADOR A	0¦7	ACUMULADOR B	0
15	DOBLE ACUMULADOR D			0
15	REGISTRO ÍNDICE X			0
15	REGISTRO ÍNDICE Y			0
15	APUNTADOR DE STACK			0
15	CONTADOR DEL PROGRAMA			0

FIGURA 4.1 Registros de código de estado

La figura 4.1 muestra los 7 registros de programa disponibles para el programador. Los 2 acumuladores de 8 bits (A y B) pueden ser usados por algunas instrucciones como un solo acumulador de 16 bits, llamado el registro D, el cual permite establecer una operación de 16 bits, no obstante que el CPU es un procesador de 8 bits. El grupo de instrucciones involucra operaciones con los registros de índice. Doce instrucciones de manipulación de bit pueden operar en cualquier memoria o registro.

Las instrucciones de cambio D con X y D con Y pueden usarse para obtener rápidamente valores índice en el doble acumulador (D), donde se puede usar aritmética de 16 bits. Se incluyen también dos instrucciones de división de 16 bits por 16 bits.

Adicional a las instrucciones del MC6800 y MC6801, el conjunto de instrucciones del M68HC11 incluye 91 códigos de operación nuevos.

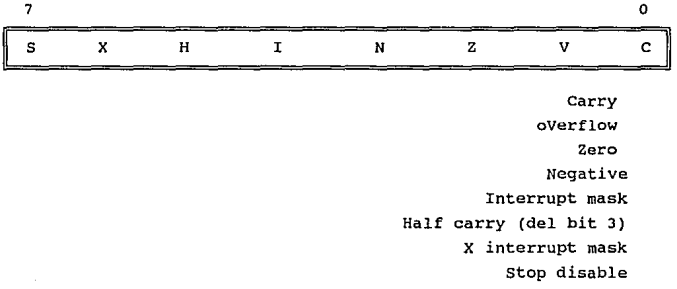


Fig. 4.2 Banderas del M68HC11

Productos derivados del M68HC11.

La familia de microcontroladores M68HC11 está compuesta de varios miembros y nuevos miembros están siendo desarrollados. En la fig. 4.3 se muestran los componentes de la familia 68HC11.

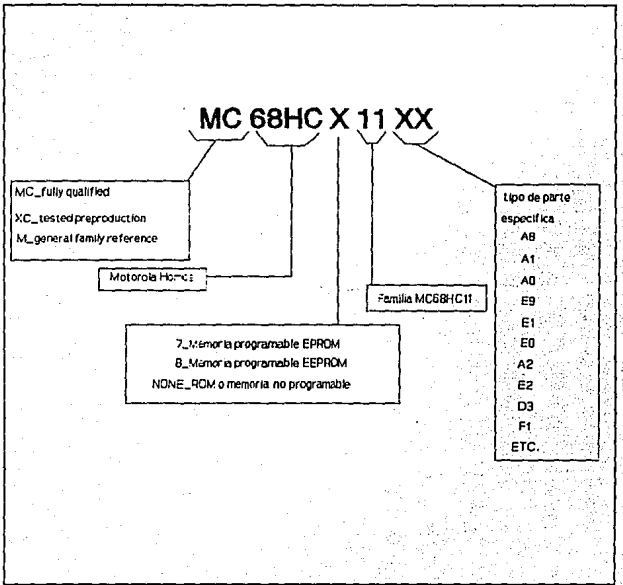


Fig. 4.3 Familia 68HC11

4.1.1.2 MCS-51

La familia de microcontroladores INTEL de 8 bits, tiene las siguientes características:

El 8051 es el miembro original de la familia MCS-51; sus atributos más importantes son:

- CPU de 8 bits, optimizado para aplicaciones de control.
- Extensa capacidad de proceso booleano (bit lógico simple).
- Direccionamiento de memoria de programa de hasta 64 kB.
- Direccionamiento de memoria de datos de hasta 64 kB.
- 4k bytes de memoria ROM integrada (8051).
- 32 líneas entrada/salida, de direccionamiento individual y bidireccional, divididos en 4 puertos, con 8 bits/puerto.
- Dos temporizadores/contadores de 16 bits.
- UART full duplex.
- Oscilador del reloj integrado.
- Canal serial programable full duplex.

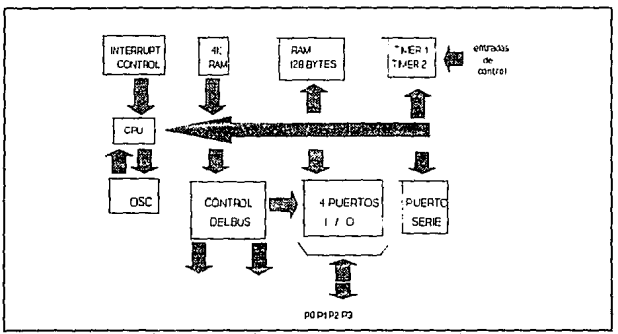


Fig. 4.4 Diagrama a bloques del 8051.

- Estructura de interrupciones 6 fuentes /5 vectores, con dos niveles de prioridad. Dos líneas de interrupción.
- La memoria de datos se direcciona separadamente de la memoria de programa.

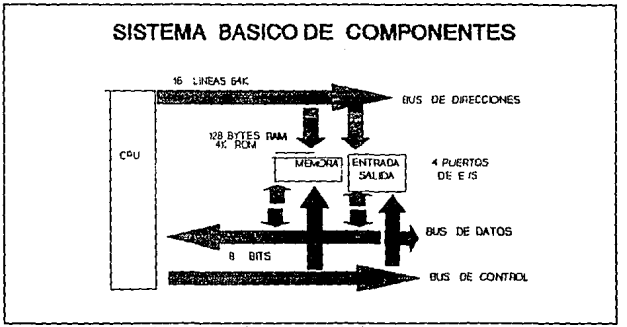


Fig. 4.5 Diagrama básico de componentes.

El CPU genera las señales de lectura (RD) y escritura (WR), que son necesarias durante el acceso a la memoria externa de datos. La memoria externa de programa y la memoria externa de datos pueden ser combinadas si se desea, aplicando las señales RD y PSEN a las entradas de una compuerta AND, y, entonces, la salida de la compuerta fungirá como el habilitador de lectura para la memoria RAM/ROM externa.

BUS	SERIE O PARALELO
MEMORIA	ROM, RAM, PROM, EPROM Y/O EEPROM
I/O	PARA COMUNICAR O GUARDAR INFORMACION EXTERNA A UN TECLADO, MONITOR, DISCO, IMPRESORA

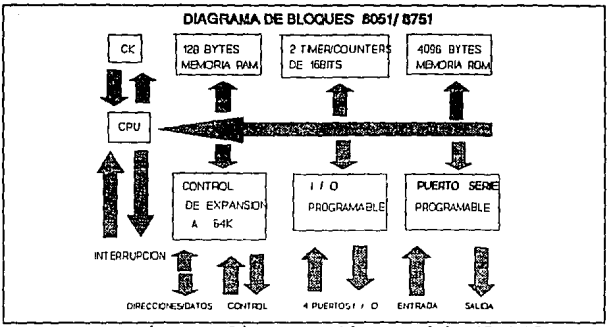


Fig. 4.6 Diagrama a bloques del 8051/8751.

Aplicaciones típicas:

- Videocassetteras;
- Contestadoras automáticas;
- Automóviles;
- Instrumentación médica;
- Hornos de microondas;
- Escalas digitales.

EL 8051H tiene 4096 bytes de memoria ROM interna; de memoria externa se direcciona un total de 64 kB. La memoria RAM interna es de 256 bytes; la mitad inferior (128 bytes) se usa para guardar datos variables del programa; la mitad superior se reserva para los registros de función especial (SFR). Los SFR se usan para guardar los valores de bytes que controlan la operación del 8051. Los SFR, cuyas direcciones son divisibles por ocho, son direccionables por bit. Además de los SFR que son direccionables por bit, 16 bytes de memoria RAM interna son también direccionables por bit, lo cual es una característica importante del 8051.

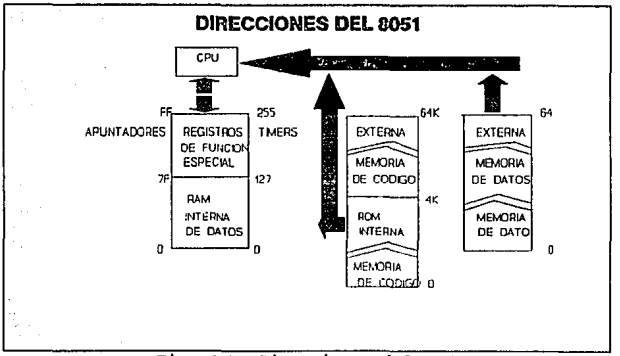


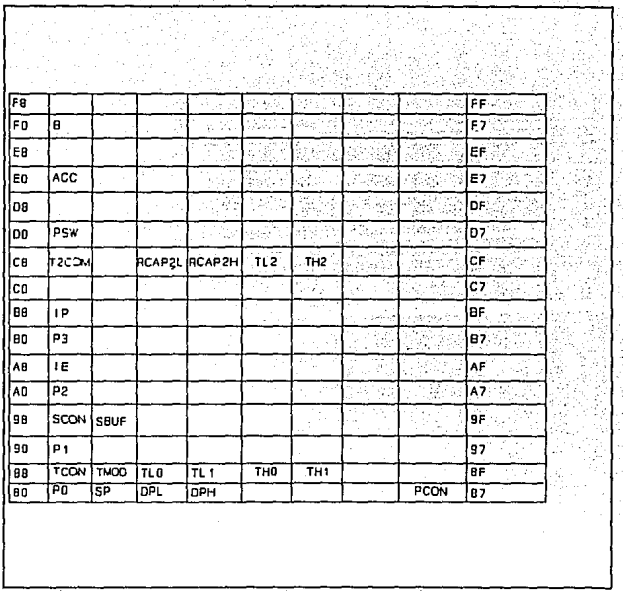
Fig. 4.7 Direcciones del 8051.

La figura 4.8 muestra el espacio de direccionamiento de los registros de función especial (SFR). Dicho espacio está incluído de la dirección 128 (80H) a la dirección 255 (0FFH) y solamente es accesible por direccionamiento directo. Algunas de las direcciones en el espacio de SFR son direccionables tanto por bit como por byte; algunas son direccionables solamente por byte. Los SFR almacenan algunos valores que el 8051 usa cuando ejecuta instrucciones.

Cada puerto de entrada/salida tiene una función alterna:

- Puerto 0: tiene multiplexada la parte baja del bus de direcciones y el bus de datos, durante el acceso a memoria ROM y memoria RAM.

- Puerto 1: es de entrada/salida; tiene funciones alternas sólo sobre el 8052. Las terminales de entrada/salida P1.0 y P1.1 sirven como las funciones T2 y T2EX respectivamente.



F8									FF
F0	B								F7
E8									EF
E0	ACC								E7
D8									DF
D0	PSW								D7
C8	T2CON		RCAP2L	RCAP2H	TL2	TH2			CF
C0									C7
B8	IP								BF
B0	P3								B7
A8	IE								AF
A0	P2								A7
98	SCON	SBUF							9F
90	P1								97
88	TCON	TMOD	TL0	TL1	TH0	TH1			8F
80	P0	SP	DPL	DPH				PCON	87

Fig. 4.8 Mapa de memoria de los Registros de Función Especial (SFR).

- Puerto 2: es de entrada/salida; manda el byte de la parte alta de la dirección durante el direccionamiento de memoria RAM y ROM externa, que usa 16 bits.

- Puerto 3: es un puerto de entrada/salida, pero tiene la capacidad de usarse para funciones alternas:

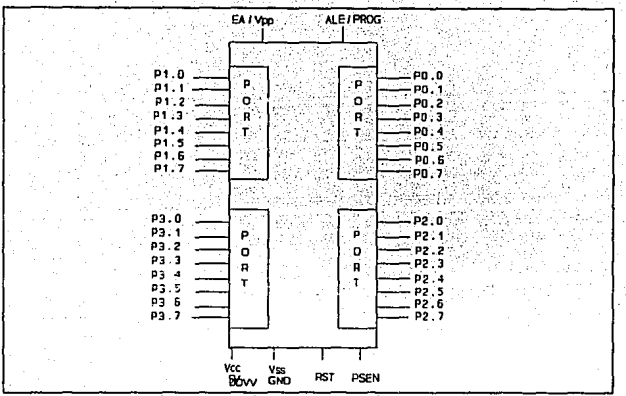


Fig. 4.9 Diagrama funcional del 8051.

Terminal	Funciones alternas
P3.0	RXD (entrada puerto serie)
P3.1	TXD (salida del puerto serie)
P3.2	INT0 (interrupción externa 0)
P3.3	INT1 (interrupción interna 1)
P3.4	T0 (Timer 0 de entrada)
P3.5	T1 (Timer 1 de entrada)
P3.6	WR (terminal de escritura a memoria)
P3.7	RD (terminal de lectura de memoria)

Productos miembros de la familia MCS-51.

Éstos son: 8031, 8051 y 8751. El 8051 y 8031 dan una solución efectiva para aplicaciones de control que requieren direccionamiento de hasta 64 kbytes de memoria RAM o ROM. Forman parte de la familia HMOS.

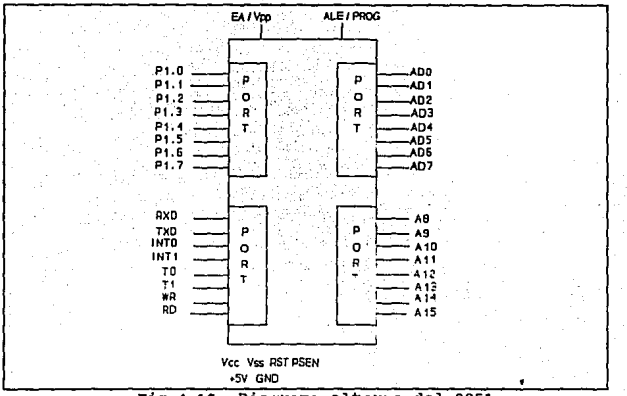


Fig 4.10 Diagrama alterno del 8051.

Versiones CHMOS.

El 8031, 8051 y 8751 están disponibles en versiones CHMOS de ahorro de energía: 80C31, 80C51 y 87C51 respectivamente. Estos componentes operan a alta velocidad, tienen una gran densidad de transición y consumen menos potencia que los componentes HMOS ya vistos. Cada uno de estos componentes tiene la misma funcionalidad que sus estándares, mas 2 modos de selección de software que reducen el consumo de potencia: Idle y Power down. Estos modos de operación los hacen ideales para aplicaciones con baterías.

Modo Idle: Este modo detiene al CPU, mientras permite que la RAM, temporizadores/contadores, puerto serie y sistema de interrupciones sigan funcionando.

Modo Power Down: Este modo guarda el contenido de la RAM, pero congela al oscilador, causando que todas las funciones del chip dejen de operar.

Miembros originales de la familia HMOS

*5 entradas de interrupción
CMOS-------Trabajan con 20 mA
IDLE--------(ocioso)---5mA
Modo de bajo consumo--20uA

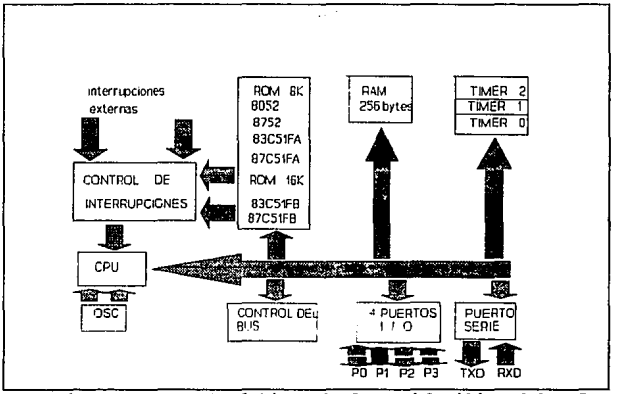


Fig. 4.11 Características de las subfamilias del 8051.

Características de la subfamilia 8052

El 8052 tiene 8 kB de memoria ROM programable de fábrica y el 8032 es una versión sin memoria ROM del 8052, mientras que el 8752 tiene 8 kB de EEPROM. Todos estos componentes pueden ser

usados en aplicaciones de control que requieren hasta 64 kB de direccionamiento de memoria RAM o ROM. Sus características comúnes son:

* 256 bytes de memoria de lectura/escritura.
* 32 líneas bidireccionales de entrada/salida, configuradas en 4 puertos de 8 bits.
* Tres temporizadores/contadores de 16 bits.
* seis fuentes, dos niveles de prioridad, estructura de interrupciones anidadas.
* Un puerto serie programable de entrada/salida.
* Un oscilador con circuito de reloj en el chip.

Subfamilias 83C51FA, 80C51FA y 87C51FA.

El 8032, 8052 y 8752 están disponibles en versiones CHMOS de baja potencia o atenuada: 83C51FA y FB, 80C51FA y 87C51FA y FB, respectivamente. Éstos tienen más velocidad, tienen una gran densidad de transición y consumen menos potencia que los HMOS ya listados. Cada uno tiene la misma funcionabilidad que sus estándares, mas los modos Idle y Power-Down seleccionables por software, que los hacen ideales para aplicaciones con baterías. Las partes FB son compatibles en terminales con los dispositivos FA. Los componentes FB tienen 16k de ROM. Tienen incluído un nuevo dispositivo: el arreglo contador programable (PCA).

El PCA (Programmable Counter Array) consiste de un contador de 16 bits y 5 módulos compara/captura de 16 bits. El compara/captura y el PCA comparten, del puerto 1, los pins del P1.2 al P1.7. Cada módulo compara/captura tiene su propio registro de modo, CCAPMn, el cual es usado para configurar el módulo n. Cada módulo es capaz de ser programado como un:

* Modulador de ancho de pulso (usado para conversión digital a analógica).

* Registro compara/captura (muy exacta medición de ancho de pulso en tiempo real).
* Gran velocidad de salida.

Resúmen de dispositivos de la familia 8051

DEVICE NAME	ROMLESS VERSION	EPROM VERSION	ROM BYTES	RAM BYTES	16 BIT TIMERS	RECKAGE
8051	8031	8751	4K	128	2	40 PIN DIP
80C51	80C31	87C51	4K	128	2	PIN DIP
8052	8032	8752	8K	256	3	
83C51FA	80C51FA	87C51FA	8K	256	3	
83C51FB	80C52FA	87C51FB	16K	256	2	
83C152	80C152	87C152	8K	256	2	48 PIN DIP 68PIN PLCC
83C452	80C452	87C452	8K	256	2	68PIN PGA
83C51FC	80C51FC	87C51FC	16K	256	3	DIP
8044	8344	8744	4K	192	2	40PIN DIP

4.1.2 LATCH OCTAL 74LS373.

Para desarrollar sistemas con el microcontrolador 80C51, INTEL especifica el uso de un latch octal, que para este sistema es el 74LS373. El latch se conecta al puerto 0 del microcontrolador, y almacena temporalmente la parte baja del bus de direcciones, cuando se realiza una lectura o escritura a un

dispositivo externo. El puerto 0 es bus de direcciones y bus de datos casi al mismo tiempo: primero muestra el valor correspondiente al bus de direcciones, y lo almacena en el latch; posteriormente, se prepara para leer el dato o para escribirlo.

4.1.3 PUERTO PROGRAMABLE DE INTERFASE.

El 8255A de INTEL es un dispositivo de entrada/salida, programable, de propósito general, diseñado para usarse con los microprocesadores de INTEL. Tiene 24 terminales de entrada/salida, los cuales pueden ser programados individualmente en 2 grupos de 12 y usados en 3 modos de operación.

En el primer modo (modo 0), cada grupo de 12 terminales de entrada/salida puede ser programado en grupos de 4, para ser entradas o salidas. En el modo 1, cada grupo puede ser programado para tener 8 líneas de entrada o salida. De los 4 pins restantes, 3 son usados para protocolo y señales de control de interrupción. El tercer modo de operación (modo 2) usa 8 pins para un bus bidireccional, y 5 líneas (tomando una prestada del otro grupo) para protocolo. Su funcionamiento es programado por el software del sistema, por lo que no necesita lógica adicional para desarrollar interfase con periféricos externos.

4.1.4 RELOJ EN TIEMPO REAL MM58274C.

El MM58274 se fabrica usando tecnologia CMOS y está diseñado para operar en sistemas con microprocesador donde se requieran funciones de reloj de tiempo real y de calendario. El oscilador controlador de cristal integrado de 32.768 kHz mantiene al cronómetro operando con un voltaje mínimo de 2.2 V, lo que permite operarlo con una batería de reserva de baja potencia.

Tiene como aplicaciones:

* Terminales de punto de venta.
* Terminales contestadoras.
* Procesadores de palabras.
* Poleo de datos.
* Control de procesos industriales.

Características del MM58274:

* Cronómetro para décimas de segundo hasta decenas de años en registros accesibles independientemente.
* Registro de años bisiestos.
* Programable para operar en modo de 12 ó 24 horas.
* Frecuencia de cristal amortiguada fuera en modo prueba para establecer la oscilación fácilmente. Se requiere un capacitor de 20 pF, un capacitor trimmer de 6pF-36pF y un cristal, para completar el circuito oscilador de tiempo de 32.768kHz.
* Bandera de cambio de datos; permite pruebas simples para cambio de hora.
* Tiempo de interrupción programable, independiente, con salida de colector abierto.
* Compatible con TTL.
* Operación de paro o amortiguamiento de baja potencia.
* Información accesible en palabras de 4 bits, aleatoriamente.
* Requiere de batería de resplado para mantener la información.

4.1.5 MEMORIAS.

Existen varios tipos de memorias semiconductoras:

Memorias de sólo lecutra (ROM): como su nombre lo dice, sólo

pueden leerse; se programan de fábrica, y se utilizan por lo general en grandes volúmenes de producción, para almacenar información, de acuerdo a un pedido elaborado por un usuario. Dicha información jamás se cambiará.

Memoria programable de sólo lectura (PROM): es similar a la ROM, sólo que en este caso el usuario la programa mediante un programador; una vez programada, la memoria no puede borrarse.

PROM borrable (EPROM): este tipo de dispositivo se usa comúnmente como un almacenamiento no-volátil. El usuario puede programarlo, y puede borrarse usando una lámpara de luz ultravioleta, para ser programado nuevamente.

Memoria estática de acceso aleatorio (SRAM): es en realidad una memoria de lectura y escritura. Está construída con flip-flop's que pueden almacenar 2 estados: 0 ó 1, y no requieren de una acción de refresco o de un reloj para mantener la información almacenada (por eso se le denomina estática). La información se pierde tan pronto como se apaga la fuente de energía que la alimenta.

Memoria dinámica de acceso aleatorio (DRAM): utiliza capacitores para almacenar la información (1 y 0 binarios para la presencia o ausencia de carga). Su celda básica de memoria consiste de un transistor de disparo y un capacitor de almacenamiento. Tiene la desventaja de que sus celdas capacitivas requieren que constantemente se estén "recargando", debido a las corrientes de fuga; es por ésto que se les llama dinámicas.

PROM borrable eléctricamente (EEPROM): se les llama también E-cuadrada PROM. Es similar a la EPROM, sólo que en este caso es programable mediante un circuito operacional. Los accesos de escritura son lentos (de 2 a 10 milisegundos para completarse) y

el número de operaciones de escritura es limitado (1,000 veces aproximadamente). Sin embargo, permite procesos rápidos de lectura.

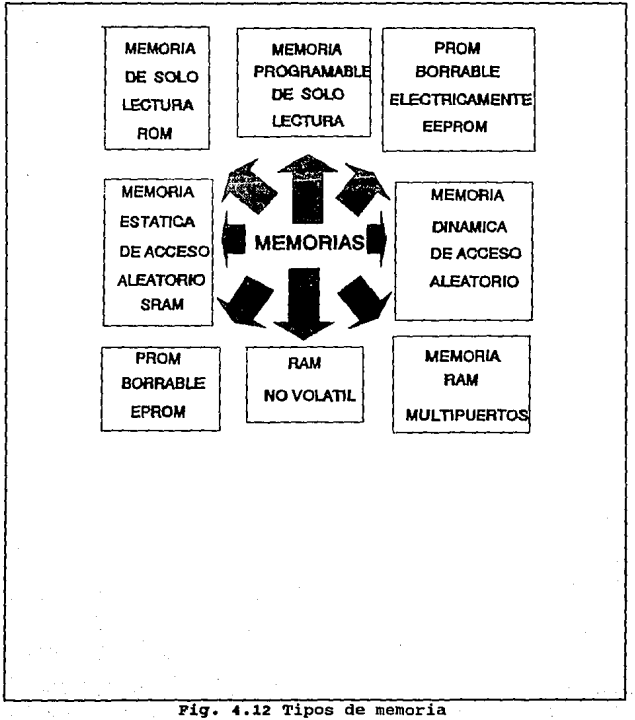


Fig. 4.12 Tipos de memoria

RAM no-volátil: es una memoria SRAM de baja potencia, que cuenta con una batería de respaldo. La batería, que es típicamente de litio, mantiene la información almacenada durante largos períodos de tiempo. Tiene la ventaja sobre la EEPROM que se lee y escribe rápidamente, sin limitación en el número de escrituras. Sin embargo, requiere de un circuito externo de apoyo y de una batería.

En el mercado podemos encontrar diferentes tamaños de memorias: de 2, 4, 8, 16 y 32 kbytes, donde 1 kbyte son 1024 bytes, y cada byte está constituído por 8 bits.

4.1.5.1 EPROM CMOS 27C64.

La 27C64 es una EPROM construída con tecnología CMOS, lo que le proporciona una alta velocidad, y un muy bajo consumo de energía. Puede almacenar 65,536 bits (8 KB x 8). Está diseñada para operar a +5V, con una tolerancia del 10%, y se encuentra disponible en un encapsulado DIP de 28 terminales. Sus características son:

- Tiempo de acceso abajo de 150 ns.
- Bajo consumo de energía:
  - Modo activo: 55 mW máximo.
  - Modo de espera: 0.55 mW máximo.
- Operación estática, no requiere reloj.
- Salida tres-estados.
- Compatible con circuitos TTL y CMOS.
- Voltajes de entrada: +6.5V a -0.6V.
- Rango de temperatura: de -10 a +80 °C.
- Voltaje de programación: 14 V.

4.1.5.2 SRAM μPD43256A.

Ésta es una típica SRAM, construída por NEC. Tiene una capacidad de 256 k-bits, organizada en 32 kbytes, de 8 bits cada byte. Es un dispositivo de alta velocidad, con un bajo consumo de energía. Sus características son:

- Alimentación con una fuente de +5V.
- Operación completamente estática.
- Terminales de entrada y salida compatibles con niveles TTL.
- Salidas tres estados.
- Voltaje mínimo de retención de datos: 2 V.
- Encapsulado plástico DIP de 28 terminales.
- 14 terminales de direcciones; 8 terminales de datos.
- Tiempo de acceso menor o igual a 150 ns.

4.1.5.3 RAM NO-VOLÁTIL.

Una RAM no-volátil (NVRAM) es una RAM estática que tiene conectada una batería de litio y un circuito de control inteligente para mantener datos, aún en ausencia de energía. El circuito de control monitorea el nivel de voltaje disponible por la memoria en todo momento, hace cambio a la pila de litio cuando es necesario, y protege a la memoria contra fluctuaciones de voltaje. El objetivo de las RAM no-volátiles es preservar la información durante días, meses o quizá hasta años. Los requisitos para construir una memoria SRAM no-volátil son:

1) No todas las SRAM pueden usarse para construir una NVRAM. Es necesario seleccionar una memoria que pueda retener datos con un voltaje bajo de alimentación. Ésto significa que debe ser capaz de conservar la información con una fuente de voltaje de 2 V, requiriendo 100 mA o menos, en su estado de retención (ésto lo cumple la μPD43256A).

2) Durante el estado de retención, se deberá proporcionar a la entrada CE de la SRAM un voltaje cercano al voltaje de polarización.

3) Al estar en el modo de retención, la terminal CE deberá estar aislada del resto del circuito en el que se encuentre trabajando la SRAM; de lo contrario, se corre el riesgo de alterar o perder la información. La reconexión deberá hacerse únicamente cuando el circuito se encuentre nuevamente polarizado.

Actualmente se cuenta con dos posibilidades: construír el circuito que controlará al sistema de respaldo de energía, o bien comprar el circuito ya hecho, verificar su forma de instalación y probar su correcto funcionamiento.

4.1.6 CIRCUITOS DE DALLAS SEMICONDUCTOR.

Actualmente se cuenta con la empresa Dallas Semiconductor, que se ha especializado en sistemas de respaldo de energía en circuitos electrónicos. Esta empresa fabrica circuitos que previenen fallas de energía, y permiten que una memoria continúe almacenando su información durante un periodo de hasta 10 años. Además, esta compañía también fabrica memorias que tienen integrado el sistema de resplado de energía; reloj en tiempo real, y hasta un microcontrolador con reloj y memoria no-volátil integrados.

4.1.6.1 MEMORIA NO-VOLÁTIL DS1243Y.

Es una memoria no-volátil con reloj fantasma. Sus características son:

- El reloj en tiempo real guarda centésimas de segundos, minutos, horas, días, fecha del mes, meses y años.
- La NVSRAM 8 KB * 8 remplaza directamente a una RAM estática volátil o a una EEPROM.
- La batería de litio integrada mantiene la operación de calendario y retiene los datos de la memoria RAM.
- Tiene integrado un circuito de control para el monitoreo de Vcc; realizará la conexión de la memoria a la batería, si el nivel de Vcc es menor a 4.0 V, protegiendo a la memoria contra escritura.
- Operación del reloj transparente a la operación de la RAM.
- El mes y el año determinan el número de días en cada mes.
- Encapsulado de 28 terminales, estándar.
- Rango de temperatura de operación (0 °C a 70 °C)
- La exactitud es mayor a +/- 1 minuto/mes @ 25 °C.
- Más de 10 años de retención de datos en ausencia de voltaje.
- Disponible con tiempo de acceso de 200ns.
- La batería viene desconectada de fábrica, y se conectará cuando se encienda el circuito por primera vez .

4.1.6.2 MICROCONTROLADOR DS5000T.

Tiene las siguientes características:

* Es un microcontrolador con un reloj/calendario incorporado.
* Batería interna de litio que mantiene el reloj funcionando en ausencia de Vcc.
* 8 ó 32 kbytes de RAM no-volátil para datos/programa.
* Partición de memoria seleccionable para memoria de datos/programa.
* 4 puertos disponibles para control de sistemas.
* Tiene incorporado una protección de programas contra piratería.
* Monitoreo de secuencia y watchdog aseguran operación de pruebas de ruptura.

* Compatible con la industria estándar del 8051 en terminales y conjunto de instrucciones.
* Exactitud del reloj superior a 2 min./mes @ 25 °C.

La combinación de un microprocesador con un reloj/calendario en tiempo real es una poderosa herramienta para aplicaciones con manejo de tiempo, lo que permite desarrollar nuevos sistemas para monitorear eventos, actividades bajo calendario y operaciones cronometradas.

4.1.7 DISPLAYS DE CRISTAL LÍQUIDO.

El AND241 es un módulo display de cristal líquido (LCD) compacto que posee un panel LCD de matriz de puntos, un controlador y un circuito manejador. Este módulo puede desplegar 160 tipos de letras, números y símbolos, así como 32 símbolos especiales, hechos por el usuario. Sus características son:

* Alto contraste, clara muestra de caracteres grandes.
* Sólo necesita 5 volts de polarización.
* Rango de tempertura amplio de 0°C a 50°C.
* Formato de caracteres de 5*10 puntos y cursor.
* Control LSI interno con RAM de display y ROM generador de caracteres.
* Interfase directa con microprocesadores de 4 y 8 bits.
* 11 comandos de control.
* Disponibles en 16, 20, 24 ó 40 caracteres por 1, 2 ó 4 renglones.

4.1.8 DISPLAYS DE 7 SEGMENTOS.

Los displays de 7 segmentos están construídos con leds. El diodo emisor de luz (LED) es, como su nombre lo indica, un diodo

que producirá luz visible cuando se encuentre energizado. Por definición, el término eficiencia es una medida de la capacidad de un dispositivo para producir un efecto deseado. Para el LED, ésto corresponde a la proporción del número de lumens (flujo luminoso) generado por watt aplicado de energía eléctrica. Por lo general, el led de color rojo es el que presenta mayor eficiencia. Además, la intensidad de luz es mayor a 0° (visto de frente) y el valor más bajo ocurre a 90° (cuando se observa el dispositivo desde un lado).

Un display de 7 segmentos es un conjunto de 7 leds (de ahí su nombre), colocados formando un 8; tienen además un led incluído para representar el punto. Con esta disposición puede formarse cualquier dígito entre el 0 y el 9, además de un gran número de las letras del alfabeto, tan solo con encender los segmentos adecuados. Estos displays están disponibles en la actualidad en muchos tamaños y conectados con cátodo común (tierra común) o ánodo común (Vcc común).

Los displays convencionales de 2 dígitos de 7 segmentos c/u tienen las siguientes características eléctricas:

Corriente por segmento en mA----------20
Disipación de potencia en mW----------800
Voltaje de pico inverso p/segmento----------5
Vida de servicio en horas----------100,000

4.1.9 MANEJADORES DE DISPLAYS.

Se tiene la opción de manejar los displays de 7 segmentos y los de cristal líquido (que no incluyen controlador) mediante los manejadores de displays.

El 7211 y 7212 constituyen una familia de manejadores y

decodificadores de displays de LCD, y de 7 segmentos, de 4 dígitos, no multiplexados. El ICM7211 está configurado para manejar un display de cristal líquido (LCD) y otorgarle una larga vida útil. Los dispositivos ICM7212 están configurados para manejar displays con leds de ánodo común y proveen una entrada de brillo que puede ser usada en niveles lógicos normales como un habilitador de display, o con un potenciómetro como un control de brillo de display. El dispositivo básico proveé 4 entradas de bit-dato y 4 entradas de selección de dígito.

Los dispositivos con interfase para microprocesador (sufijo M), proveen latches de entrada de datos y latches de código de selección de dígito. Los dispositivos básicos decodifican las 4 entradas binarias de bit en una salida alfanumérica de 7 segmentos hexadecimales.

4.1.10 DECODIFICADORES DE TECLADO.

Para lograr una interfase con el usuario se tiene la opción de usar teclados. Tenemos dos tipos de teclado básicamente: los matriciales y los decimales. Pueden usarse manejadores de teclado, para incorporarlos a sistemas con microprocesadores. Uno de los más populares es el 74HC922/923. Este decodificador de teclado CMOS contiene integrados todos los circuitos necesarios para trabajar como interfase, entre una matriz de interruptores de 16 ó 20 SPST y un sistema digital. Además anula los rebotes y es compatible con niveles TTL.

4.2 ANÁLISIS DE INFORMACIÓN Y PROPUESTA DE SOLUCIÓN.

Una vez especificados todos los dispositivos candidatos a formar parte del sistema, procederemos a comentar sobre los dispositivos elegidos, y por qué se eligieron.

Con respecto al microcontrolador a utilizar, se analizaron las dos posibilidades que se consideraron más convenientes. Debe destacarse que la firma MOTOROLA ha desarrollado microcontroladores muy poderosos para este tipo de aplicaciones automotrices. Sin embargo, dadas las finalidades de este sistema, se decidió utilizar el microcontrolador 8051 de INTEL, ya que cumple con los requerimientos necesarios, además de ser un circuito muy sencillo de utilizar, y mucho más económico. Haber utilizado el 68HC11 hubiera representado un mayor costo del sistema en general (tarjeta y componentes), además de que no se hubieran aprovechado todos los dispositivos que contiene. Ambas compañías poseen una gran familia de componentes, por lo que no hay ningún problema en la elección de una u otra. El conjunto de instrucciones de ambos dispositivos es extenso, pero el del 8051 tiene instrucciones de multiplicación y división, que le permiten usarse en el sistema sin ningún problema.

Elegir el 8051 nos permite utilizar otros dispositivos de la misma marca, como el puerto programable de interfase 8255, el cuál provee 3 puertos en un circuito integrado y ofrece sistema de control en lectura y escritura de datos entre el microcontrolador y los periféricos asociados. La incorporación de un PPI proporciona una mayor versatilidad en el manejo de los dispositivos externos del sistema. Además, es necesario utilizar un latch 74LS373, conectado al puerto 0 del microcontrolador.

Para permitir una sencilla decodificación del mapa de memoria se eligió el decodificador 3*8 74HC138, que era suficiente para la arquitectura requerida, además de resultar un dispositivo muy confiable y económico.

Se decidió utilizar una memoria EPROM 27C64 para almacenar en ella los programas, ya que se disponía del grabador necesario, y esta memoria proporciona suficientes localidades para almacenar un programa súmamente extenso sin ninguna contrariedad. Pudo

utilizarse una EEPROM, pero su precio es de 2 a 3 veces superior al de una EPROM, por lo que no fué primordial la ventaja que ofrecen las EEPROM de borrado eléctrico inmediato, contra los 15 minutos necesarios para borrar una EPROM. Por otra parte, en el mercado se encuentran microcontroladores 8051 con EPROM integrada. Sin embargo, se optó por usar memorias externas ya que sería necesario borrarlas y grabarlas varias veces, y ésto se hacía más fácilmente con varias memorias externas. Además, el costo de un microcontrolador con EPROM integrada es mucho mayor que aquél que no la tiene.

Una decisión fundamental para el diseño del sistema fué la elección del circuito encargado de preservar la información por largo tiempo. Pudo pensarse en una memoria EEPROM, pero por lo general sus voltajes de grabación son de +12V, y ésto significaría incluír más dispositivos al circuito; además, se había explicado que solamente se pueden grabar 1000 veces, por lo que su vida útil sería efímera para el sistema. Definitivamente el circuito a emplear tendría que ser una RAM no-volátil, y, aunque pudo construirse el circuito de respaldo de batería, se prefirió utilizar uno que ya lo tuviera integrado, pues el costo, a fin de cuentas, resultaba ser el mismo. Además, se requería utilizar un reloj en tiempo real para tener un registro más exacto de la información, por lo que se eligió la memoria DS1243Y, que tiene incorporado un circuito de respaldo de energía y un reloj en tiempo real, y además es sencillo de utilizar. Este circuito aportó un manejo confiable de la información. Haber adquirido por separado todos los circuitos necesarios para poder preservar la información del reloj y de la memoria SRAM, hubiera representado un costo casi equivalente al del DS1243Y.

Ahora bien, se podría pensar en usar el DS5000T para desarrollar el sistema, ya que en un solo integrado se tiene a un microcontrolador semejante al 8051, un reloj en tiempo real y una SRAM no-volátil, con el respaldo de batería adecuado. Con todo

ésto, el sistema hubiese quedado reducido a sólo 4 componentes (y no con los 7 que quedó al final). Sin embargo, el costo del DS5000T es súmamente elevado, y además, se usa el puerto serie para programarlo (siendo que, por lo general, los dispositivos se programan en paralelo), por lo que sería necesario adquirir también el kit de programación (DS5000TK), que incrementaría aún más el costo del sistema. Es por ésto que el DS5000T quedó descartado. (Ésto no significa que no se use en un futuro).

Con respecto a la forma en que el sistema desplegaría la información al usuario, se tenía contemplado usar displays de 7 segmentos, o bien, un display de cristal líquido. Se decidió utilizar 4 displays de 7 segmentos, ya que el display de cristal líquido es mucho más costoso; además, la cantidad de información a desplegar no era mucha y los displays de 7 segmentos podrían cubrir bien el trabajo. Asimismo, los displays proporcionan una excelente visibilidad, especialmente en lugares obscuros, como en el que seguramente sería instalado el monitor.

Al elegir los displays de 7 segmentos, se presentaba la opción de usar manejadores de display, los cuales están diseñados para sistemas con microprocesadores. Sin embargo, dada la función que cumplirían los displays, se decidió hacer el control y manejo de los mismos por software, ya que no resultaba complicado para el sistema. Con ésto se logró un importante ahorro en circuiteria y en costos. Además, el consumo de energía de los displays de 7 segmentos no representa una desventaja importante respecto a los displays de cristal líquido. Se eligieron displays de 2 dígitos porque son más comúnes en el mercado que los displays led de 4 dígitos, y por consiguiente más económicos. Se usaron displays de ánodo común con el fin de poner todos los segmentos de los displays a la fuente Vcc del sistema y con el PPI poner tierra a cualquiera de los 7 segmentos seleccionados, ya que es más sencillo activar un led cerrando el circuito hacia tierra que proporcionar con el sistema la corriente necesaria para que

enciendan los 28 segmentos, en el caso de que todos tuvieran que prender. Por lo que se refiere al PPI, puede soportar sin ningún problema la corriente de entrada que atraviesa a los displays.

Ahora bien, pensando en el uso y funcionamiento del sistema, llegamos a la conclusión de que no sería necesario un teclado, y con un número de 4 botones podríamos cumplir las necesidades planteadas en este trabajo. Al decidir usar sólo 4 botones fué posible prescindir del teclado, y después de desarrollar diferentes pruebas con los botones conectados al puerto 1 del microprocesador, verificamos que no sería necesario usar circuitos (como supresores de ruído o rebotes) entre el microprocesador y el juego de botones. Considerando ésto, fué posible eliminar el uso de manejadores de teclado, simplificando aún más el diseño y costo del sistema. Se eligieron botones que permitieran un manejo sencillo y fácil para el usuario.

En este punto ya se cuenta con todos los componentes que formarían el sistema. Sin embargo, aún falta por definir una forma útil de mostrar la información recolectada. La mejor forma de hacerlo es mediante su registro en un papel, por lo que se necesitaría una impresora sencilla, económica, compacta, de fácil acoplamiento al sistema y ligera para poder transportarse fácilmente. Estas características las encontramos en las sumadoras de diferentes marcas como Texas Instruments, las cuales se encuentran fácilmente en el mercado, y resulta sumamente sencillo hacer conexión entre el sistema y la sumadora para vaciar la información del primero en el rollo de papel del último. La impresión se desarrollaría con una sumadora convencional TI-5029; sin embargo, encontramos sumadoras del tamaño de una mano, cuyas características y bajo costo son semejantes, y además pueden utilizarse, ya que tienen el mismo principio de funcionamiento.

Capítulo 5

Descripción del Sistema Adquiridor de Datos.

Una vez elegidos los componentes que formarían el sistema de adquisición de datos, se procedería a establecer las conexiones entre los circuitos, y a elaborar los programas que rigirían el trabajo del sistema en su conjunto. En el presente capítulo se explican detalladamente las partes que constituyen al sistema adquiridor de datos (hardware y software).

5.1 DESCRIPCIÓN Y DISEÑO DE LAS TARJETAS.

5.1.1 TARJETA DEL SISTEMA ADQUIRIDOR DE DATOS.

La tarjeta del sistema adquiridor de datos cuenta con:

- una fuente integrada, formada por un diodo de entrada que protege al sistema de una polarización inversa, un regulador 7805 que regula a 5V la alimentación del sistema, y un capacitor de salida (para evitar oscilaciones en el regulador), con lo que se obtiene una tensión filtrada y regulada;
- un sistema de puertos y los conectores necesarios para integrar al sistema hasta 4 sensores; éstos se pueden usar, por ejemplo: para el monitoreo de consumo de combustible; para la medición de rpm's del motor; medición de distancia recorrida y violaciones de velocidad máxima; así como para el registro de pisadas de freno;
- conectores para los displays y los botones;
- un conector para la interfase con una sumadora.

El sistema está constituído por una arquitectura para el microcontrolador 80C51 y cuenta con:

1) Un microprocesador 80C51, que controla todas las funciones del sistema.

2) Un latch 74LS373 que sugiere INTEL para manejo de las primeras 8 direcciones del microprocesador.
3) Una memoria EPROM 27C64, la cual contiene el programa de todo el sistema.
4) Una memoria NVSRAM DS1243Y, con reloj/calendario integrado, la cual tiene las características y funciones ya mencionadas.
5) Un puerto paralelo de interfase (PPI) 8255, que cumple con la función de interfase con los displays y la impresora.
6) Un decodificador 74HC138, para realizar la decodificación de NVSRAM y PPI.

Se instaló un conector para cada puerto del PPI, así como para los puertos 1 y 3 del 80C51. Para cada conexión del puerto 3 (T0, T1, INT0, INT1) hay una salida de Vcc y una de GND. Las terminales RXD y TXD no se usan en el sistema diseñado; sin embargo, se encuentran disponibles para su uso futuro.

En seguida se muestra la función para cada conector.

PARA EL MICROCONTROLADOR 80C51

Puerto 0: Bus de direcciones y de datos para las memorias EPROM, NVSRAM y para el PPI.

P0.0----------(AD0)
P0.1----------(AD1)
P0.2----------(AD2)
P0.3----------(AD3)
P0.4----------(AD4)
P0.5----------(AD5)
P0.6----------(AD6)
P0.7----------(AD7)

Puerto 1: Entradas al sistema.

P1.0----------BOTÓN 1
P1.1----------BOTÓN 2
P1.2----------BOTÓN 3
P1.3----------BOTÓN 4
P1.4----------RECONOCIMIENTO DE
LA SUMADORA

Puerto 2: Bus de direcciones de las memorias EPROM y NVSRAM.

P2.0----------(A8)
P2.1----------(A9)
P2.2----------(A10)
P2.3----------(A11)
P2.4----------(A12)
P2.5----------(A13)
P2.6----------(A14)
P2.7----------(A15)

Puerto 3: Interrupciones y contadores del sistema.

P3.0----------(RXD)
P3.1----------(TXD)
P3.2----------(INT0)
P3.3----------(INT1)
P3.4----------(T0)
P3.5----------(T1)
P3.6----------(WR) Al WRITE de la memoria NVSRAM y
al WRITE del PPI.
P3.7----------(RD) Al READ de la memoria EPROM,
al READ de la memoria NVSRAM y
al READ del PPI

PARA EL PPI 8255

Puerto A: Segmentos de los displays.

PA.0----------DISPLAYS SEGMENTO A
PA.1----------DISPLAYS SEGMENTO B
PA.2----------DISPLAYS SEGMENTO C
PA.3----------DISPLAYS SEGMENTO D
PA.4----------DISPLAYS SEGMENTO E
PA.5----------DISPLAYS SEGMENTO F
PA.6----------DISPLAYS SEGMENTO G
PA.7----------DISPLAYS PUNTO

Puerto B: Salidas a la sumadora.

PB.0----------SALIDA "A" A LA INTERFASE
PB.2----------SALIDA "B" A LA INTERFASE
PB.3----------SALIDA "C" A LA INTERFASE
PB.4----------HABILITADOR DE LA INTERFASE
PB.5
PB.6
PB.7

Puerto C: Selección de los displays.

PC.0----------ÁNODO DISPLAY 1
PC.1----------ÁNODO DISPLAY 2
PC.2----------ÁNODO DISPLAY 3
PC.3----------ÁNODO DISPLAY 4
PC.4
PC.5
PC.6
PC.7

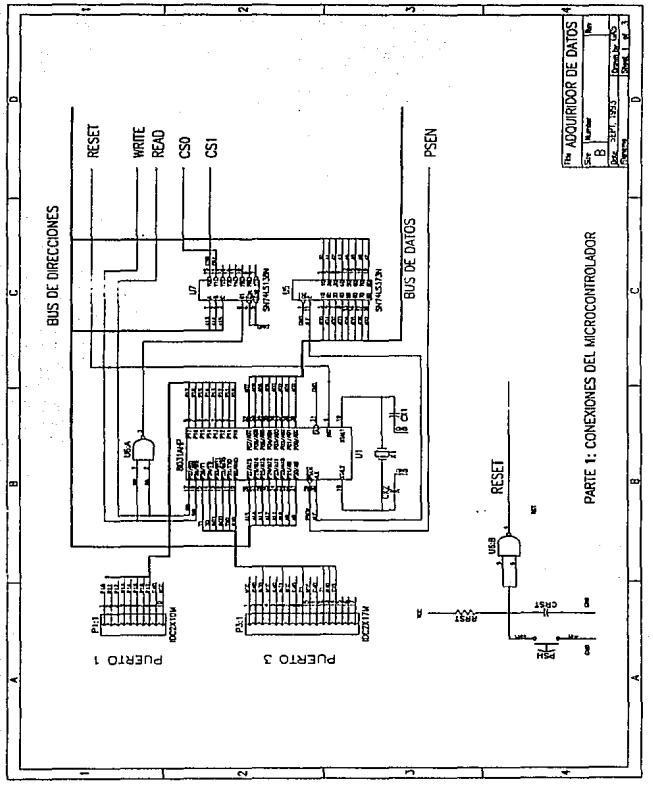


Fig. 5.1a Diagrama esquemático del sistema adquiridor de datos (parte 1).

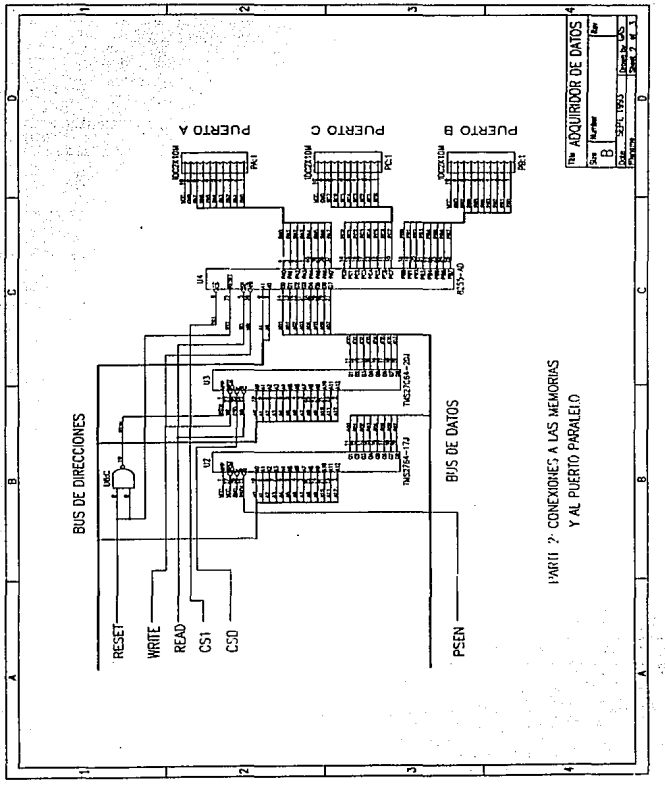


Fig. 5.1b Diagrama esquemático del sistema adquiridor de datos (parte 2).

CIRCUITOS INTEGRADOS

U1-8031
U2-2764
U3-DS1243Y
U4-8255
U5-74LS373
U6-4093
U7-74LS138
U8-LM7805

CONECTORES

PA-JUMPERS PUERTO A
PB-JUMPERS PUERTO B
PC-JUMPERS PUERTO C
P1-JUMPERS PUERTO 1
P3-JUMPERS PUERTO 3
JP1-JUMPER VIN
JP2-JUMPER GND

COMPONENTES

C0-CAPACITOR FUENTE
C1-C7-CAPACITORES 0.1 uF
RRESET- RESIST RESET 10K
CRESET- CAPACIT RESET 10uF
D1-DIODO 1N4001
CX1-CX2- CAPACIT. CRISTAL 33pF
X1-CRISTAL 12 MHz

PARTE 3: FUENTE DE ALIMENTACION Y LISTA DE COMPONENTES

Title: ADQUIRIDOR DE DATOS		
Size: B	Number	Rev
Date: SEPT/1995	Drawn by: UAS	
File name:	Sheet 3 of 3	

Fig. 5.1c Diagrama esquemático del sistema adquiridor de datos (parte 3).

5.1.2 TARJETA DE VISUALIZACION (DISPLAYS).

Los displays y los botones se montaron en una tarjeta de circuito impreso independiente del sistema, y se conecta al mismo mediante un cable plano. Este circuito impreso tiene los segmentos multiplexados y el segmento de un dígito en particular se habilita por la activación del ánodo común del display correspondiente. Los botones están conectados a tierra en un punto, y en otro al puerto 1 del microprocesador, de acuerdo con la distribución de recursos asignada.

5.1.3 TARJETA DE INTERFASE CON LA SUMADORA.

Para desarrollar la interfase de la sumadora se partió de la ventaja que ofrece el circuito impreso de ésta, mostrando claramente los puntos que conecta cada tecla al ser presionada, evitando con ésto el trabajo de tener que analizar su sistema electrónico. Los puntos que se conectan son los siguientes, de acuerdo al diagrama mostrado.

0---------(3) - (4)
1---------(8) - (5)
2---------(4) - (5)
3---------(5) - (1)
4---------(5) - (7)
5---------(6) - (8)
6---------(4) - (6)
7---------(1) - (6)
8---------(7) - (6)
9---------(3) - (8)
+---------(11) - (8)
*/T-------(11) - (7)
↑ --------(12)-(13)
. --------(2)-(8)

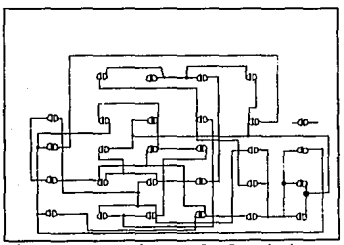


Fig.5.2 Conexiones de los botones de la sumadora TI-5029.

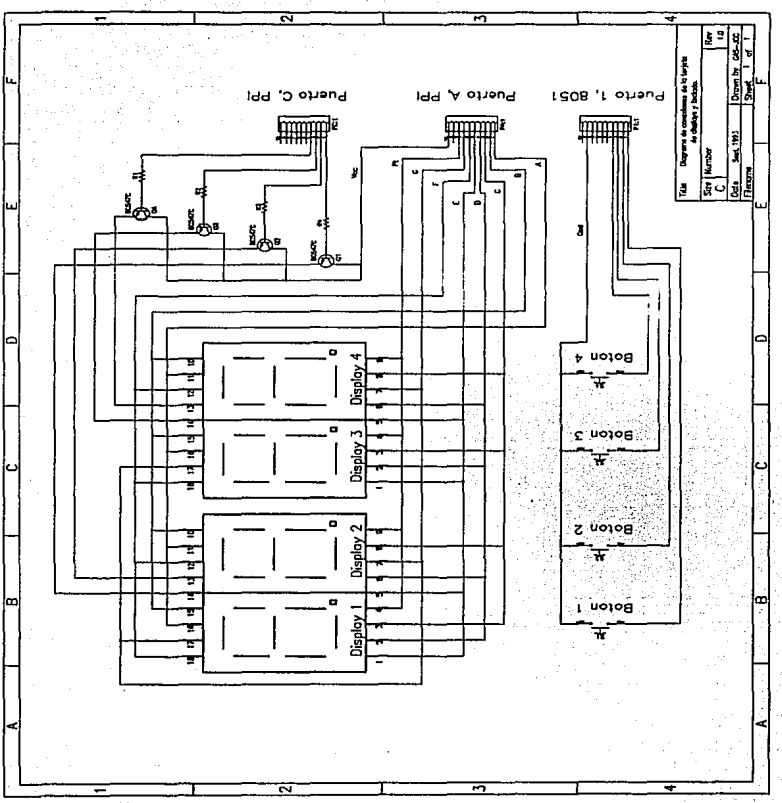


Fig. 5.3 Diagrama esquemático de la tarjeta de displays.

La interfase consiste en un circuito que toma como señales de activación las 4 primeras terminales del puerto B del 82C55, más una señal adicional para habilitar la operación del circuito. Estas 4 señales entran a un decodificador 4*16 74LS154, el cual, por la combinación de estas entradas, presenta 16 señales individuales; cada una de éstas se conectó a un circuito inversor, debido a que el decodificador dá salidas activas con nivel bajo. Cada inversor activa un MOC3011, que cumple con una función similar a una tecla de la sumadora al ser presionada. La terminal P1.4 del microcontrolador se encarga de reconocer si la sumadora está conectada o no. El habilitador del decodificador está conectado a la terminal PB.4 del 82C55, y controla el paso de señales del sistema hacia la interfase de la sumadora.

Cabe mencionar que, como la mayor parte las sumadoras actuales tienen el mismo principio de funcionamiento, esta tarjeta de interfase se puede adaptar prácticamente a cualquier sumadora, teniendo así la posibilidad de imprimir la información registrada por el sistema, usando un circuito de muy bajo costo, además de ser portátil. Por otra parte, la sumadora puede seguir empleándose para hacer cuentas sin ningún problema.

5.2 DESCRIPCIÓN DE LOS SENSORES.

El sistema desarrollado emplearía un solo sensor, para medir la distancia recorrida y los excesos de velocidad. Sin embargo, durante la elaboración del presente trabajo se planteó la elaboración de un nuevo programa para registrar un mayor número de variables, lo cual involucraría el uso de otros sensores externos. Las nuevas variables posibles de medir fueron: el número de pisadas del freno de un vehículo; las revoluciones por minuto (rpm's) del motor; y el consumo de combustible del vehículo.

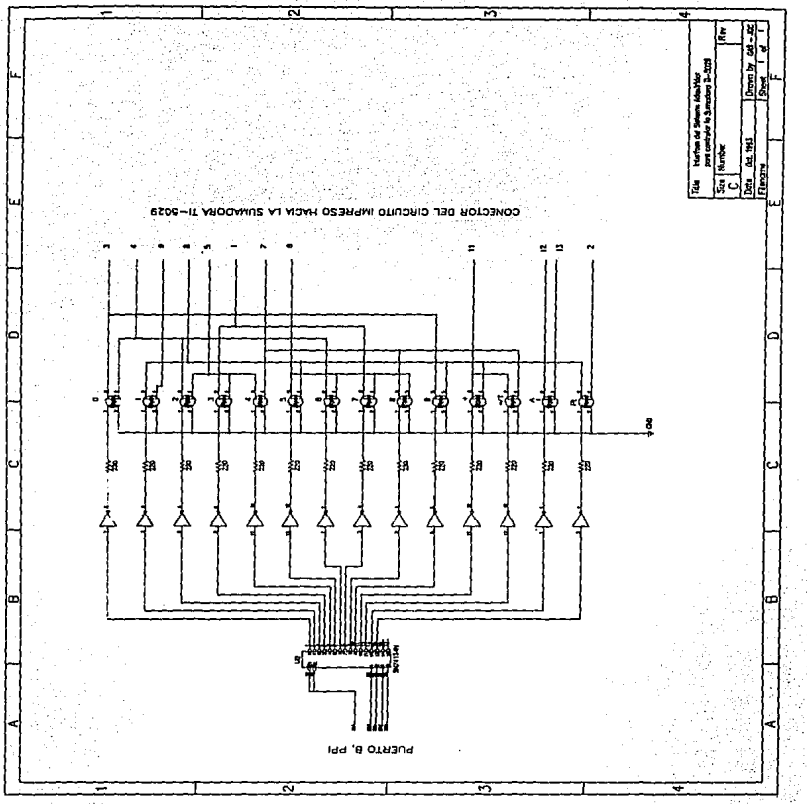


Fig. 5.4 Diagrama esquemático de la tarjeta de interfase con la sumadora.

Para medir la distancia recorrida y la velocidad, los velocímetros/odómetros digitales emplean un sensor conectado al chicote del velocímetro, que contiene un imán y un embobinado. Al cruzar el imán por dicho embobinado, produce una corriente eléctrica, que puede ser transformada a pulsos digitales, y así manejar la información con circuitos electrónicos.

En esta tesis se planteó y construyó un nuevo tipo de sensor para medir la distancia recorrida y la velocidad, y se instaló en el chicote del velocímetro. Se sabe que, debido a la conexión que presenta el chicote a la caja de velocidades del vehículo (o al diferencial, según la marca y el modelo), siempre se obtendría una relación lineal entre el número de vueltas de éste y la distancia recorrida por el vehículo. El sensor consistiría en un disco circular, que presenta 8 ranuras equidistantes, instalado en el chicote, y junto a dicho disco se dispondría de un sensor óptico capaz de apreciar el paso de las ranuras del disco. Dicho sensor (fotomicrosensor) presentaría un número de pulsos igual al número de ranuras que ha detectado. Así, se proporcionaría directamente una señal digital útil para el microcontrolador.

Para registrar las pisadas de freno, se podría usar como sensor un botón instalado directamente en el pedal del freno. Así se obtendría un pulso cada vez que el conductor lo pisara.

Para registrar las rpm's del motor, se podría usar un sensor conectado al distribuidor del automóvil, si el motor es de gasolina. Sin embargo, también se podría instalar un sensor óptico en el eje del motor, para contar su número de vueltas en un período de tiempo determinado (un segundo). Ésto se aplicaría, por ejemplo, en casos de motor de ciclo Diesel. Otra opción es instalar un sensor, similar al utilizado en la medición de velocidad, en el chicote del medidor de rpm's que tienen algunos vehículos (especialmente los camiones).

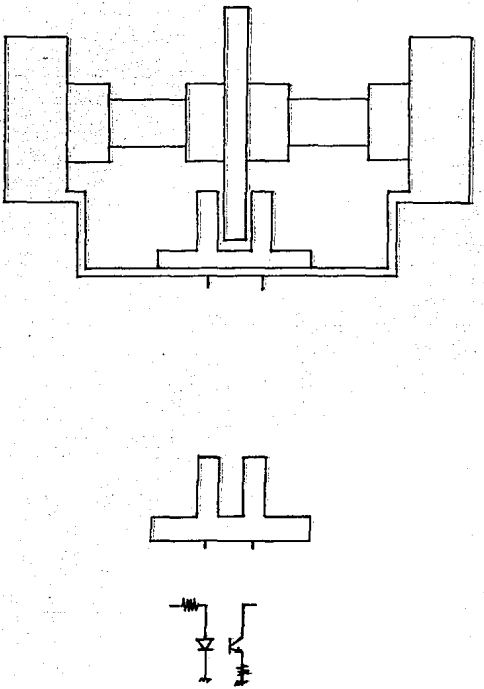


Fig. 5.5a Sensor de velocidad y distancia (diagrama de instalación y conexiones).

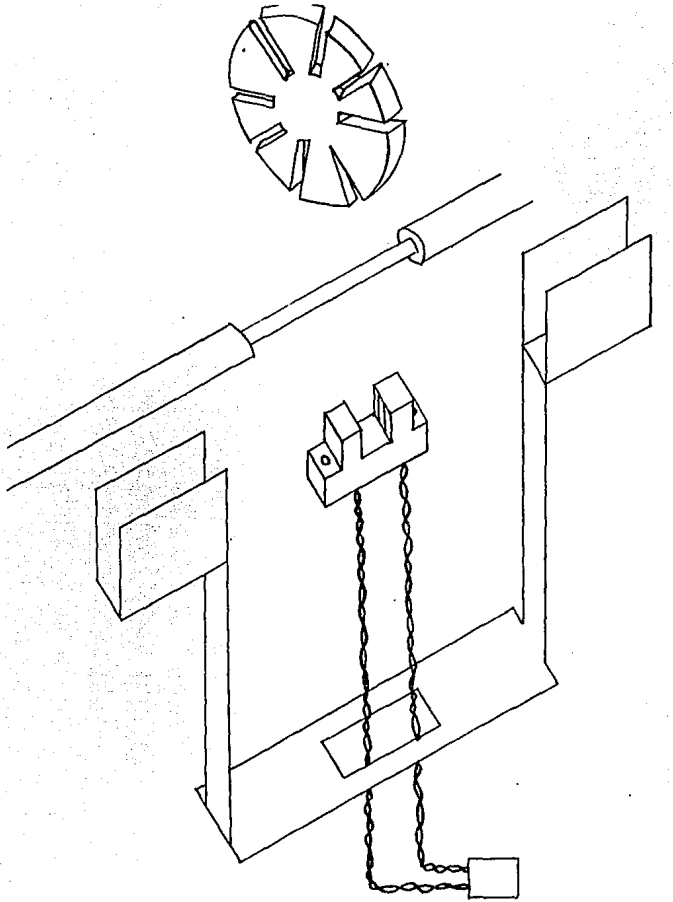


Fig. 5.5b Sensor de velocidad y distancia
(Diagrama explotado)

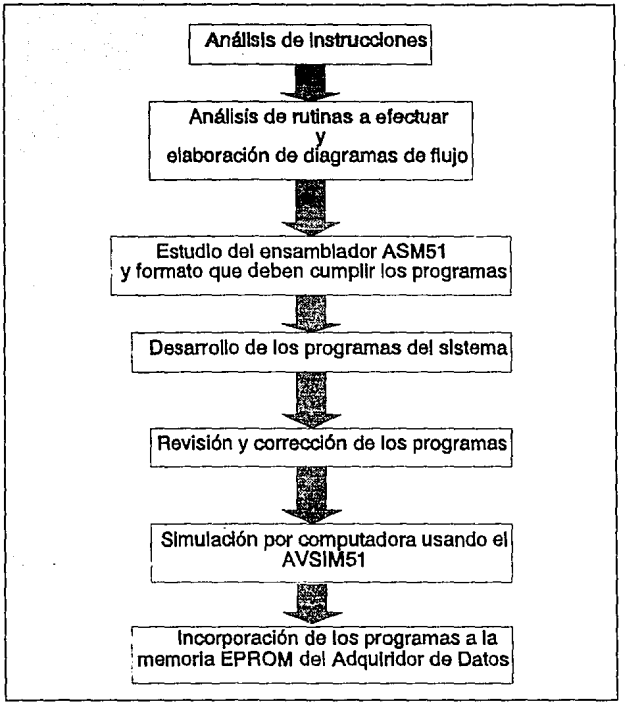
5.3 PROGRAMAS Y RUTINAS DE PRUEBA.

Un componente esencial del sistema son los programas que lo constituyen. A continuación se describe el trabajo desempeñado para elaborar rutinas y programas, que permitieran verificar el correcto funcionamiento del sistema, y se comentan algunas de importancia fundamental para el desarrollo del presente trabajo.

5.3.1 PROCEDIMIENTO DE ESCRITURA Y PRUEBA DE LOS PROGRAMAS.

El microcontrolador alrededor del cual se construyó el sistema trabaja siguiendo las instrucciones que se le indiquen. Se cuenta con un conjunto de instrucciones válidas, que son las que el microcontrolador puede ejecutar, y cada una tiene su representación en código hexadecimal, que es el usado por el microcontrolador. Dichas instrucciones deben grabarse en una memoria EPROM; el microcontrolador las leerá y ejecutará tan pronto como el sistema se encienda.

Antiguamente se tenía que escribir el conjunto de instrucciones (constituyendo un programa) en un lenguaje entendible por un ser humano (usando las instrucciones permitidas), y después, a mano, traducir todas las instrucciones a su equivalente en código hexadecimal. Ésto ofrecía muchos inconvenientes ya que era una rutina de trabajo muy tardada y laboriosa, además de que era muy factible cometer un error (y frecuentemente más de uno), y no era sencillo detectarlo(s) y corregirlo(s). Ahora el procedimiento es mucho más sencillo y seguro, gracias al empleo de computadoras y programas específicos, que actualmente brindan un gran apoyo a los estudiantes y diseñadores de sistemas digitales. A continuación se muestra un diagrama ilustrando los pasos que se siguieron para escribir cada uno de los programas, haciendo que el microcontrolador trabaje de acuerdo con su objetivo.



Todo esto nos permitió trabajar en una forma más sencilla y segura, brindando la corrección oportuna de los programas, y garantizando de antemano el correcto funcionamiento del sistema.

5.3.2 DESCRIPCIÓN DE LAS RUTINAS DE PRUEBA.

Una vez ensamblado el sistema en las tarjetas, fue necesario elaborar una serie de pequeñas rutinas o programas para probar que trabajaba correctamente. A continuación se describen las rutinas escritas para realizar las pruebas adecuadas para revisar el correcto desempeño del sistema. En el apéndice B: "Listados de programas y rutinas de prueba", que se encuentra al final de esta tesis, pueden consultarse algunos de los programas que se elaboraron, y que se construyeron usando las rutinas descritas en esta sección.

Programa de Despliegue de informacion.- Se escribió una rutina encargada de desplegar el letrero "code" en los displays, para verificar las conexiones a los mismos. Se encuentra incorporada en algunos de los programas utilizados.

Rutina de lectura de botones.- Para verificar que los botones funcionaban correctamente, se creo un programa para mostrar en los displays el número del botón que se oprimía. Una vez probado que se leían correctamente, se extrajo de dicho programa una rutina titulada APRIETA, la cual se encuentra incorporada en varios de los programas usados.

Programa para probar la conexión con la sumadora TI-5029.- Como ya se ha comentado, para generar un reporte impreso de las lecturas obtenidas se decidió utilizar la sumadora TI-5029, por lo que se escribió un programa para verificar que las conexiones entre el sistema y la sumadora estaban correctas, y para verificar también que la impresión se efectuaba correctamente. El programa se encarga de mandar a imprimir todos los números (0 al 9) y el punto decimal, además de probar las teclas de suma, total y de dejar un renglón en el papel de impresión. El listado del programa y los resultados impresos de su ejecución se encuentran

contenidos en el apéndice B: "Listados de programas y rutinas de prueba"; y en el apéndice D: "Pruebas de impresión usando la sumadora TI-5029", respectivamente.

Rutina de acceso al reloj de tiempo real.- La memoria RAM no-volátil que se adquirió contiene un reloj en tiempo real, y para tener acceso a dicho reloj es necesario que el sistema escriba una clave predeterminada, en alguna de las localidades de la RAM. Esta rutina se encarga precisamente de escribir la clave en la memoria RAM no-volátil, que nos permita tener acceso a su reloj.

Programa para poner a funcionar al reloj de Dallas DS-1243Y.- Se desarrolló un programa para actualizar la fecha y la hora del reloj, así como para ponerlo a trabajar. El programa se encarga de escribir en el reloj la información necesaria, en el orden adecuado.

Rutina para contabilizar el transcurso de un segundo.- Esta rutina se encarga de verificar cuando ya ha transcurrido un tiempo de un segundo.

Programa para leer la hora del reloj de Dallas.- Para verificar que el reloj de Dallas recibió adecuadamente la información de la fecha y la hora, se escribió un programa para leer dicha información y mostrarla en los displays. Para el desplazamiento a lo largo de todas las localidades del reloj de Dallas, se utilizan los botones 1 y 4.

Programa para apreciar y definir el brillo de los displays.- Se desarrolló un programa para variar el tiempo que permanecen encendidos los displays, ocasionando así variación en su brillo. Este programa permitió precisamente definir los valores adecuados para que los displays muestren mejor la información. Aprovechando la existencia del programa para leer la hora del reloj de Dallas, se generó un nuevo programa al que se incorporaron rutinas para

valorar el brillo que pueden emitir los displays, mientras se muestra en ellos la información contenida en el reloj; para controlar el brillo (aumentarlo o disminuírlo) se usan los botones 2 y 3.

Programa para probar el correcto acceso a la memoria RAM.- Se diseñó un programa para verificar que se tenía una correcta comunicación con la memoria RAM, tanto para leer datos como para escribirlos. El programa fue diseñado para escribir datos en la RAM, para posteriormente leerlos y mostrarlos en los displays.

Rutina para manejo de interrupciones.- Se elaboró una rutina para la familiarización con la activación y desactivación de las interrupciones que tiene el microcontrolador. La rutina se elaboró para su manejo en el simulador exclusivamente.

5.3.3 DESCRIPCIÓN DE OTRAS RUTINAS AUXILIARES.

Mientras se encontraba en desarrollo el presente trabajo de tesis, surgió la opción de desarrollar un sistema cuya función fuera la de registrar algunos parámetros presentes en un vehículo, a lo largo de un trayecto específico; dichos parámetros eran: la distancia recorrida por el vehículo; las veces que se oprimía el freno durante el trayecto; las revoluciones por minuto que presentaba el motor; y el consumo de combustible del vehículo. Las lecturas registradas deberían mostrarse en los displays únicamente, mediante la acción adecuada de las teclas del sistema.

El programa escrito para el presente trabajo de tesis contempla además algunos otros aspectos, tales como seguridad en el acceso de la información, manejo de menúes y vaciado de la información en un medio impreso. Sin embargo, las rutinas elaboradas para la opción comentada en el párrafo anterior,

resultaron una guía muy útil al momento de escribir el programa definitivo, además de que muestran un aspecto muy amplio de la versatilidad que se puede encontrar en el sistema que se diseñó y construyó. Por tanto, a continuación se describe esa opción desarrollada y las rutinas que la conforman.

Programa de registro de parámetros en vehículos.

Como ya se comentó, esta opción de programa debería encargarse de registrar los siguientes parámetros:

- Número de veces que se oprime el pedal del freno
- Consumo de combustible
- Distancia recorrida por el vehículo
- RPM's del motor

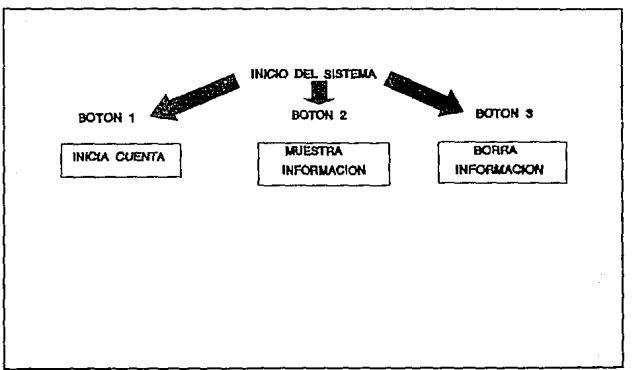


Fig. 5.6 Opciones del programa auxiliar.

Se desarrollaron cuatro rutinas independientes; cada una contempló uno de los parámetros a medir, y cada una se probó por separado.

El sistema diseñado permite llevar el registro de las 4 lecturas al mismo tiempo, sin ningún problema, ya que el microcontrolador empleado cuenta con 2 contadores independientes, y dos terminales de interrupción, que pueden usarse junto con rutinas diseñadas específicamente para llevar la cuenta de eventos. Si se analizan las rutinas, puede apreciarse que son idénticas, ya que desempeñan un propósito similar. Posteriormente se procedió a incorporar cada una a un programa, que se encargaría de llevar el registro de los 4 parámetros al mismo tiempo.

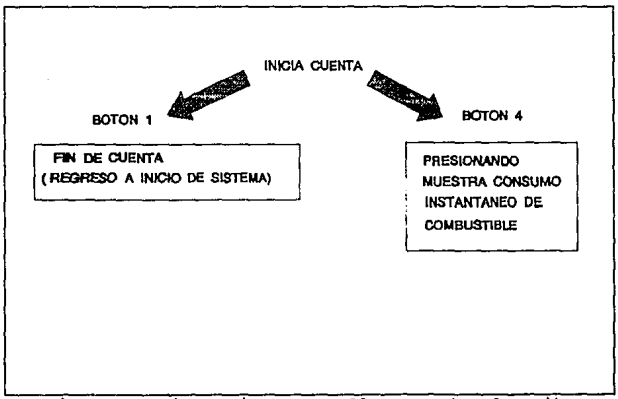


Fig. 5.7 Opciones mientras se lleva a cabo el monitoreo.

Éste programa se encargaría de iniciar el proceso de monitoreo al oprimirse el botón 1 del sistema. Comenzaría

entonces el registro de las rpm's del motor, que se actualizarían cada segundo y se mostrarían en los displays. El sistema también registraría el consumo de combustible y la distancia que va recorriendo el vehículo; estos registros también se actualizarían cada segundo. El sistema también contaría el número de veces que el conductor pisa el freno del vehículo; las pisadas podían ocurrir en cualquier momento, y cuando ocurrieran se actualizaría un contador dedicado a llevar dicho registro.

Al oprimirse el botón 4 mientras transcurría el monitoreo, los displays mostrarían el consumo de combustible registrado hasta el momento, apareciendo nuevamente las rpm's del motor al soltarse dicho botón. Para suspender el monitoreo, debería oprimirse nuevamente el botón 1.

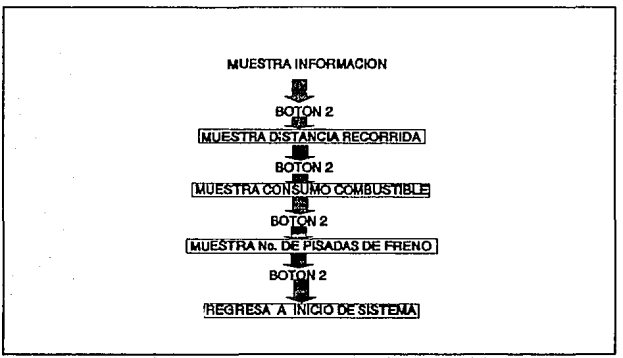


Fig. 5.8 Consulta de la información.

La consulta de la información total registrada se realizaría una vez concluído el monitoreo, oprimiendo el botón 2 para consultar primero la distancia recorrida en el trayecto,

oprimiéndolo nuevamente para consultar el consumo total de combustilbe; oprimirlo una vez más para consultar el número de veces que se pisó el freno; y oprimirlo una última vez para concluír la consulta de información.

Una vez concluído el monitoreo, al oprimirse el botón 4, se borraría la información registrada, y entonces el sistema se encontraría listo para ejecutar un nuevo monitoreo.

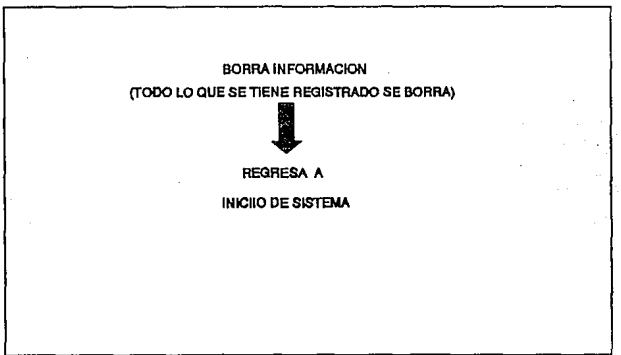


Fig. 5.9 Borrado de la información mediante la tecla 4.

En el apéndice B: "Listados de programas y rutinas de prueba" pueden consultarse las rutinas descritas, y el listado del programa que se encargaría de ejecutar el monitoreo.

Con respecto a los sensores, todos entregarían directamente pulsos digitales, por lo que se necesitaría realizar mediciones para establecer la relación entre el número de pulsos y el parámetro por medir (para el caso del combustible, la distancia y las rpm's).

Para eso, también se desarrolló un programa encargado de contar un número determinado de pulsos, y mostrarlo en los displays. Dicho programa permitía consultar el total de número de pulsos registrados y borrarlos para iniciar una nueva cuenta; su listado se puede consultar en el apéndice B del presente trabajo.

Todas estas rutinas resultaron de gran apoyo para el diseño del sistema definitivo.

5.4 DISEÑO Y DESCRIPCIÓN DEL PROGRAMA MONITOR.

Para desarrollar un programa, es necesario seguir un criterio de diseño. A continuación se describe lo utilizado para diseñar el programa monitor, así como su funcionamiento.

5.4.1 DISEÑO DEL PROGRAMA MONITOR

Para desarrollar un diseño era fundamental seguir los denominados principios de diseño. Para el caso de la programación, los fundamentales son: la modularidad, la regularidad y la conectividad, y se intentaron seguir de la forma que se consideró más adecuada. Para el diseño del programa monitor, la tarea total a desempeñar se dividió en tareas más pequeñas (principio de modularidad), que tuvieran aproximadamente una función lo más semejante entre sí (principio de regularidad), para posteriormente conectarse, una a una, entre sí, probando que su desempeño no variara (principio de conectividad). El programa monitor se dividió en las siguientes rutinas:

Rutina de acceso al sistema.- Esta rutina se encarga de mostrar en los displays un mensaje, indicando el inicio de su ejecución, y espera a que el usuario escriba, mediante el uso de los botones, la clave correcta que le permita tener acceso a las

diferentes opciones del sistema. La clave consta de 6 dígitos, donde cada uno puede ser 1, 2, 3 ó 4 (cada número corresponde a uno de los botones del sistema). Si el usuario ha escrito correctamente la clave, se muestra un mensaje que así lo indica.

Rutina de contabilización de violaciones al sistema.- Se considera como violación el equivocarse al ingresar la clave del sistema. Esta rutina muestra un mensaje indicando el intento de violación, y se actualiza un contador que registra el número de violaciones que ha sufrido el sistema.

Rutina de contabilización de la distancia recorrida.- Esta rutina se encarga de preparar al sistema para realizar la cuenta de la distancia que va recorriendo el vehículo. La distancia se actualiza cada segundo.

Rutina de contabilización de violaciones de velocidad.- Esta rutina se encarga de verificar que no se haya excedido la velocidad máxima permitida. Si se excedió, se realizará la actualización de un contador dedicado a registrar el número de violaciones ocurridas.

Rutina de cambio de velocidad máxima.- Esta rutina permite al usuario cambiar el valor de la velocidad máxima permitida al vehículo.

Rutina de cambio de clave del sistema.- Esta rutina permite al usuario cambiar la clave de acceso al sistema.

Rutina para borrar información de la RAM no-volátil.- Esta rutina se encarga de borrar la información guardada en la memoria RAM no-volátil. Ésto ocurre cuando ya no se desea conservar dicha información, y su lugar será sustituído por nueva información.

Rutina de impresión.- Esta rutina realiza la impresión de la

información recopilada por el sistema en el transcurso de los trayectos.

Rutina para cambiar la clave del sistema desde la pantalla inicial.- Pudiera existir un caso en el que el usuario haya olvidado su clave de acceso al sistema. Para eso, se creo esta rutina que permite cambiar la clave desde la pantalla inicial. El procedimiento consiste en: oprimir los botones 1 y 4 (los de los extremos) durante aproxiamadamente 5 segundos, hasta que desaparezca el mensaje mostrado en los displays. Después se sueltan los botones 1 y 4, y se oprimen el 2 y el 3 (los de enmedio) también durante 5 segundos, y se sueltan. Entonces se podrá realizar el cambio de la clave del sistema.

Para la elaboración de cada una de estas rutinas, se emplearon algunas de las mencionadas en el punto 5.3.2, además de otras más pequeñas, que pueden apreciarse en los listados mostrados en el apéndice B. Cada rutina se realizó y probó por separado, primero en el simulador, y después directamente en el sistema, cuando se consideró conveniente, para posteriormente conectarlas entre sí y constituír el programa monitor.

5.4.2 FUNCIONAMIENTO DEL PROGRAMA MONITOR.

El funcionamiento del programa monitor, que controla al sistema adquiridor de datos, se puede dividir en 2 secciones:

- una sección de monitoreo, en la que el sistema realiza la medición de los parámetros correspondientes; y,

- una sección de programación, que abarca la impresión de la información registrada, cambio de la velocidad máxima permitida, etc.

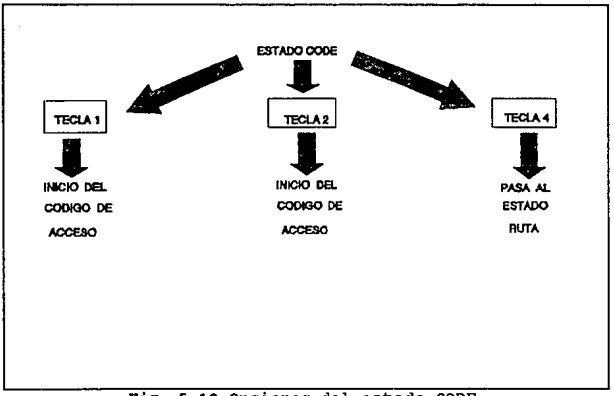


Fig. 5.10 Opciones del estado CODE.

Al encender el sistema, éste se sitúa en el estado CODE, y se muestra en displays precisamente el mensaje CODE. Desde este estado se puede accesar a cualquiera de las dos secciones mencionadas previamente, mediante la pulsación de las teclas adecuadas. A continuación se describen detalladamente ambas secciones, y cómo tener acceso a las mismas.

SECCIÓN DE MONITOREO.

Al encender el sistema, muestra inicialmente la pantalla CODE; para entrar a la sección de monitoreo, deberá pulsarse el botón 4.

Aparecerá entonces el letrero RUTA, para indicar que el sistema iniciará el proceso de monitoreo de información.

Oprimiendo la tecla 1, se le indica al sistema que ha concluído el monitoreo de la ruta actual, iniciando inmediatamente un nuevo monitoreo para una nueva ruta o tramo.

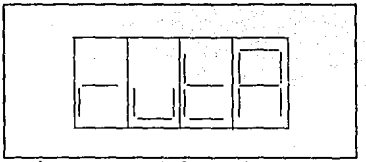


Fig. 5.11 Pantalla del estado RUTA

Para concluir definitivamente el monitoreo, deberá oprimirse el botón 4. El sistema almacenará en la memoria no-volátil los parámetros que se registraron. El sistema se situará entonces en el estado CODE, mostrando dicho mensaje en los displays.

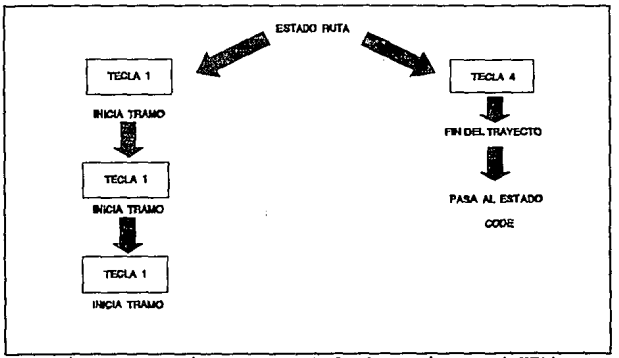


Fig. 5.12 Opciones del estado de monitoreo (RUTA).

SECCIÓN DE PROGRAMACIÓN.

Para tener acceso a la sección de programación, desde el estado CODE se deberá pulsar la clave de acceso al sistema, que consiste de 6 digitos, cada uno de los cuales puede ser 1, 2, 3 ó 4.

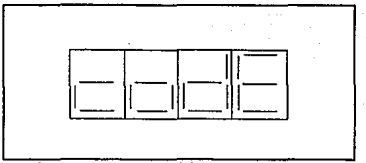


Fig. 5.13 Pantalla del estado CODE

El primer dígito de la clave no puede ser 3 ó 4; necesariamente debe ser 1 ó 2. Conforme se va pulsando la clave, se muestra en los displays el número de dígitos que se han ingresado.

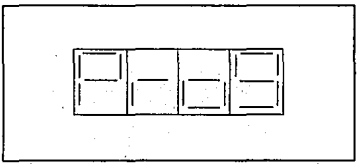


Fig. 5.14 Pantalla de la sección de PROGRAMACIÓN.

Si se comete algún error al escribir la clave, el sistema registra en su memoria un intento de violación al equipo, e inmediatamente regresará al estado CODE.

Una vez que se dá la clave correctamente, el adquiridor de datos entra a la sección de programación, mostrándose la pantalla PROG durante un breve lapso de tiempo (aproximadamente 2 segundos).

Posteriormente se muestran en los displays las opciones que la comprenden, y son:

a - para regresar a la pantalla anterior, es decir, a CODE;

i - para ingresar a la pantalla de impresión;

v - para ingresar a la pantalla de cambio de velocidad máxima permitida;

c - para ingresar a la pantalla de cambio de clave.

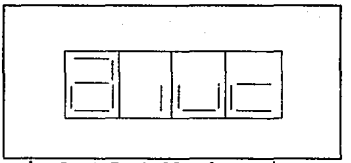


Fig 5.15 Pantalla de opciones del estado de PROGRAMACIÓN.

En el diagrama se muestran dichas opciones. Como puede apreciarse, la forma en que se muestran en la pantalla corresponde al botón que deberá oprimirse.

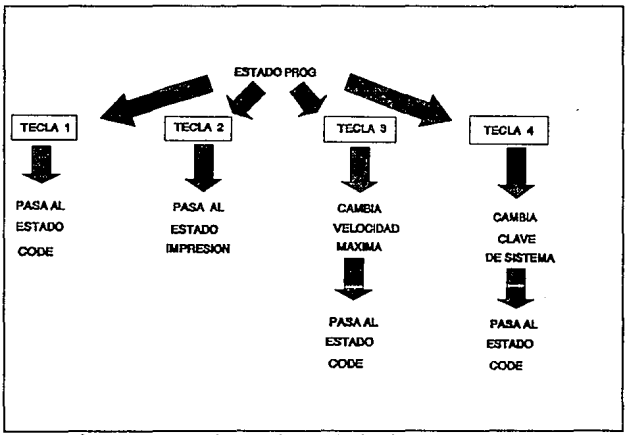


Fig. 5.16 Opciones del estado de PROGRAMACION.

Opción de cambio de velocidad máxima.

Al oprimirse el botón 3 en la pantalla "aivc", se ingresa inmediatamente a la opción para cambiar la velocidad máxima permitida. Se muestra la pantalla "v 00". La velocidad se ingresará en km/hr, y deberá ser mayor a 60; si se escribe una velocidad menor, el sistema asigna automáticamente como velocidad máxima el valor de 60.

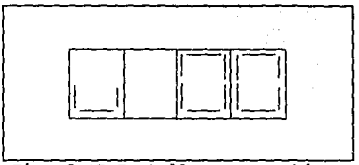


Fig. 5.17 Pantalla para cambiar velocidad máxima permitida.

El uso de los botones es el siguiente: el botón 4 se utiliza para incrementar las unidades, el botón 3 para incrementar las decenas, el botón 2 para prender o apagar las centenas, y el botón 1 para indicar al sistema que se ha terminado de ingresar la velocidad máxima. Así, por ejemplo, para situar una velocidad máxima de 95 km/hr, se deberá pulsar 5 veces la tecla 4, 9 veces la tecla 3, y la tecla 1 al terminar. Al pulsar la tecla 1, el sistema almacena la nueva velocidad, y se sitúa en la pantalla de programación "PROG". La velocidad máxima que se puede programar es de 199 km/hr.

Opción de Cambio de clave del sistema.

Al oprimir la tecla 4 en la pantalla "aivc", el sistema ingresa inmediatamente a la opción de cambio de clave del sistema; se mostrará en la pantalla "CC " (Cambio de Clave), indicando que el sistema está listo para recibir la nueva clave.

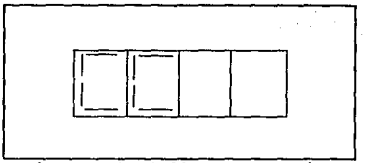


Fig 5.18 Pantalla para realizar cambio en la clave del sistema.

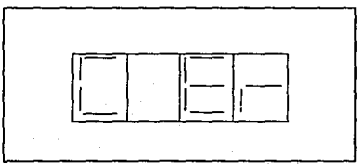


Fig. 5.19 Pantalla para indicar error al escribir la nueva clave.

Como se había comentado, la clave consta de 6 dígitos, donde cada uno puede tener un valor del 1 al 4 (uno por cada botón). El primer dígito de la clave sólo puede ser 1 ó 2. Si se pulsa el botón 3 ó el 4 como primer dígito de la clave, se mostrará en displays el letrero "C Er", indicando que hay un error. Dos segundos después se mostrará de nuevo la pantalla "CC ".

Conforme se oprimen cada una de las teclas de la nueva clave, se apreciará que los números mostrados en los displays se desplazan un lugar, de derecha a izquierda, y que el display del extremo derecho muestra el número correspondiente a la última tecla que se oprimió. Ésto sucede para indicar al usuario que el proceso se está efectuando adecuadamente.

Al terminar de pulsar los 6 dígitos que comprenden la clave,

el sistema automáticamente mostrará el letrero "List" durante 2 segundos, para posteriormente trasladarse al estado CODE, que es el estado inicial del sistema. Si se desea ingresar al estado de programación, el usuario tendrá que digitar la nueva clave.

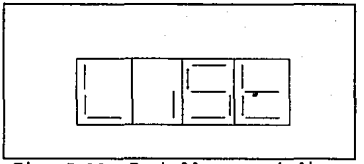


Fig. 5.20 Pantalla para indicar la aceptación de la nueva clave.

Estado de Impresión.

Al oprimir el botón 2 en la pantalla "aivc", el sistema ingresará automáticamente a la estado de "Impresión de

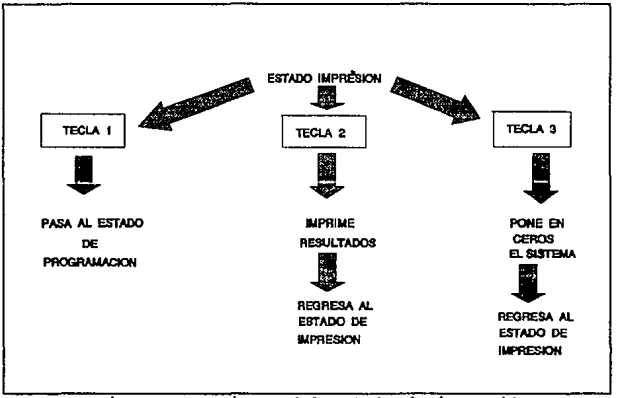


Fig. 5.21 Opciones del estado de impresión.

Resultados", mostrándose en displays el letrero "Imp" durante 2 segundos, para posteriormente mostrar el letrero "ai b", referente a las acciones que se pueden desempeñar; éstas son:

a - para regresar al estado anterior (estado de programación "PROG");

i - para imprimir las lecturas registradas por el sistema;

b - para limpiar la memoria no-volátil, y dejar espacio disponible para almacenar nueva información.

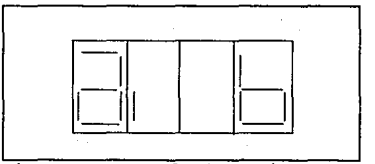


Fig. 5.22 Pantalla de opciones del estado de impresión.

Opción de borrado de información.

Al oprimir el botón 4 en la pantalla "ai b", correspondiente al estado de impresión, el sistema realiza la limpieza de la memoria no-volátil, borrando las lecturas de violaciones de velocidad y de distancias recorridas que había registrado para todas las rutas. Al terminar, muestra en displays el letrero "LiSt" durante 2 segundos, y posteriormente regresa al estado de impresión, mostrando el letrero "ImP".

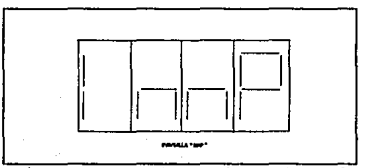


Fig. 5.23 Pantalla del estado de impresión.

Opción de impresión de resultados.

Al oprimir el botón 2 en la pantalla "ai b", correspondiente al estado de impresión, el sistema se prepara para imprimir la información que registró a lo largo de los trayectos recorridos. Primero verifica que la sumadora se encuentre conectada; si no lo está, muestra en displays el letrero "I Er", para después mostrar de nuevo el letrero "ImP", y las opciones "ai b".

Si la sumadora está conectada, aparecerá primero en los displays el mensaje "PPPP", indicando que el sistema está listo, y posteriormente los displays mostrarán "----", mientras la impresión se efectúa.

La impresión se realiza en una sumadora, la cual no puede escribir letras. Por tanto, para identificar la información es necesario ver el orden en el que se imprime y el código que le corresponde.

Primero imprime la información de identificación inicial:

Información	Código
Número de identificación del operador.	Sin código
Fecha en que se realiza la impresión	0.ddmmaa dd-día mm-mes aa-año
Hora en que se realiza la impresión	hh.mm hh-hora mm-minutos

Para cada ruta se imprime lo siguiente:

Información	Código
Número de Ruta	Sin código
Fecha de inicio de Ruta	0.ddmmaa
Hora de inicio de Ruta	hh.mm
Clave de violación de veloc. y distancia en que ocurrió	dddd.1 dddd-distancia en km
Número de viol. de veloc. en toda la Ruta	nn.2 nn-número
Distancia recorrida en la Ruta	dddd.3 dddd-distancia en km
Hora de finalización de la Ruta	hh.mm

Al final, se imprimen los siguientes datos:

Información	Código
Distancia total recorrida por el vehículo	dddd.11 dddd-distancia en km
Número total de viol. de vel.	nn.12
Número total de violaciones al monitor	nn.13 nn-número

Continúa en la sig. página ...

Información	Código
Número de veces que se ha limpiado la memoria no-volátil	nn.14 nn-número
Número de veces que se ha cambiado la clave del sistema, desde la última limpieza de la NVRAM	nn.15 nn-número

Para identificar plenamente la información, por favor consúltese el apéndice D: "Pruebas de impresión usando la sumadora TI-5029", al final de la tesis.

Al terminar la impresión, el sistema se ubica inmediatamente en el estado de impresión "ImP", para posteriormente mostrarse en pantalla el mensaje "ai b".

Para regresar a la condición inicial del sistema (salir al estado CODE), es necesario pasar primero al estado de programación (si se está en el estado de impresión, oprimiendo el botón 1). Una vez en el estado de programación, se deberá oprimir el botón 1 (opción de regresar al estado anterior).

Capítulo 6

Pruebas
y
Resultados

Se procedió a instalar las tarjetas del sistema en un gabinete construído especialmente para ello. Por ser un sistema prototipo, se decidió hacer un gabinete de dimensiones mayores a las requeridas, por si se incorporaba algún nuevo circuito. Se contempló la instalación de un disipador de calor para el regulador de voltaje, un portafusibles, conector del sistema de alimentación, conector de los sensores, además de espacio para el conector de la impresora (sumadora), y leds dispuestos para un uso futuro. La disposición de los 4 botones corresponde a los 4 displays utilizados (un botón debajo de cada display), con el fin de proporcionar una mayor facilidad en la operación del sistema.

También se le instaló un gabinete a la sumadora, para proteger su tarjeta de interfase.

Antes de instalar el sistema en el vehículo, se realizaron pruebas para verificar:

- el encendido adecuado de los displays, y su brillo óptimo;
- el correcto funcionamiento de los botones;
- posibles fallas en el reloj en tiempo real de Dallas Semiconductor; y,
- la correcta impresión de los resultados.

Todo ésto se probó con programas específicamente escritos con dicho objetivo. Esta parte de trabajo resultó exitosa.

Se construyó un "simulador" del chicote del velocímetro, en el cual se instaló el sensor, para verificar que el sistema podía registrar el número de pulsos del mismo sin ningún problema. Esta parte resultó también exitosa.

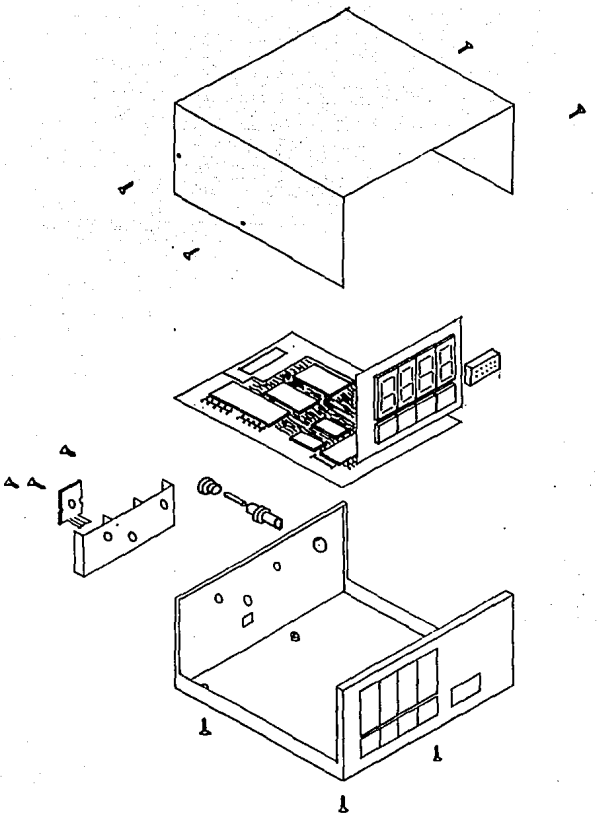


Fig. 6.1 Sistema adquiridor de datos.

Posteriormente, se adaptó el sistema a un automóvil marca Chevrolet, Malibú Classic, modelo 1980, realizando las conexiones de fuente de alimentación. Una vez que se contaba con la alimentación, se realizó la instalación del sensor óptico en el chicote del velocímetro, en el interior del vehículo. Debido al ligero peso del sensor, éste no afectaba al desempeño del chicote. Se conectó la salida del sensor a una de las entradas de señal que tiene el sistema. En el mercado se utilizan sensores magnéticos; sin embargo, para esta tesis se diseñó y probó un sensor óptico, para comprobar su uso.

Se colocó en el sistema un programa especialmente diseñado para llevar la cuenta del número de pulsos que proporcionaba el sensor del chicote, conforme el vehículo avanzaba. Con este programa se realizó la cuantificación del número de pulsos del sensor correspondientes a una unidad de distancia. Se tomó como distancia a medir, primero, 100 mts y 10 mts, obteniéndose una relación de 4.95 pulsos por metro; éstas pruebas se realizaron a una velocidad baja (10 a 20 km/hr). Posteriormente se realizaron pruebas, tomando como referencia una distancia de 1 km. La relación obtenida fué también de 4.95 pulsos por metro; estas pruebas se realizaron a velocidades altas y bajas (desde 20 hasta 90 km/hr). Ésto es así debido a que existe una relación lineal entre el número de vueltas del chicote y la distancia que recorre el vehículo.

El programa monitor diseñado puede adaptarse a cualquier vehículo, ya que sólo es necesario especificarle la relación del número de pulsos proporcionados por el sensor instalado en el chicote, por metro recorrido. Además, contiene una adaptación que permite considerar también fracciones de vuelta. Se realizó la actualización necesaria en el programa monitor, y se probó el sistema en el vehículo, obteniéndose resultados satisfactorios.

El microcontrolador es un sistema digital, por tanto, maneja variables y realiza operaciones aritméticas en forma discreta. Ésto ocasiona que algunas rutinas proporcionen ciertos errores en el monitoreo. Por ejemplo, en la rutina utilizada para programar la velocidad máxima permitida, existe un error de +3km/hr (en el peor de los casos); es decir, si se programa una velocidad máxima de 90 km/hr, el programa registrará una violación para velocidades mayores a 93 km/hr. Este error se considera severo para velocidades bajas; sin embargo, resulta no muy importante para velocidades altas, y al considerar que el sistema está programado para no recibir velocidades máximas menores de 60 km/hr, podemos concluir que no se presentará un problema grave en ese sentido.

La memoria no-volátil utilizada tiene un espacio disponible de 8 Kbytes. Considerando que se almacenan 42 bytes de información por cada ruta (asumiendo 15 violaciones de velocidad), sería posible almacenar en la memoria la información de 195 rutas, antes de que el cliente desee leerla. Si se considera que se recorren 2 rutas por día, diríamos que el sistema puede realizar un monitoreo de más de 3 meses, sin leer la memoria y sin borrarla (es decir, transcurren más de 3 meses sin que se haya agotado la capacidad de la memoria). Sin embargo, se tiene considerada una lectura de la información, por parte del cliente, de mínimo una vez al mes.

El consumo máximo de corriente que registró el sistema fue de 300 mA, lo cual es similar al de los sistemas existentes. Se eligieron circuitos cuya temperatura de operación estuviera dentro de las existentes en la República Mexicana (de 0° a +70° C, con excepción del norte de la república, en los días invernales, que puede ser menor a 0° C). Se dejó instalado el sistema en el vehículo, para verificar que resistiera las condiciones de temperatura en el Distrito Federal; el sistema no presentó ninguna variación en su desempeño.

6.1 OPCIONES A FUTURO

El equipo se desarrolló cuidando que sus componentes y circuitos tuvieran gran duración y bajo costo. El sistema y el software que lo componen están desarrollados para permitir el control de flotas de transporte y carros de alquiler.

Se piensa integrar al sistema sensores y programas para incrementar sus características como equipo auxiliar en el uso y monitoreo de un vehículo. Las nuevas características contempladas se están incorporando a solicitud y con el apoyo de la Comisión Nacional de Ahorro de Energía. Los nuevos parámetros a medir son:

a) consumo de gasto de combustible
b) rpm del motor
c) pisadas de freno

Registrar el consumo de combustible de un vehículo refleja inmediatamente su utilidad, ya que, con dicha lectura, el usuario puede cambiar sus hábitos de manejo y disminuir el consumo, ocasionando así un ahorro de dinero (y de contaminación). Sin embargo, la medición del consumo de combustible presenta varios inconvenientes, ya que será diferente para cada tipo de motor, y también para las condiciones en las que se encuentra trabajando (tiempo de marcha, aceleración o velocidad constante, etc).

En algunos casos es conveniente tener un tacómetro instalado en un vehículo, ya que con su ayuda se puede proporcionar un ahorro de combustible al efectuar los cambios de velocidad en el momento adecuado (ésto es más significativo en los camiones, sin embargo, también es válido para los motores de ciclo Otto).

Con respecto a los frenos, sabemos que hay de 2 tipos: frenos de disco y frenos de tambor. Ambos actúan por el material

de fricción denominado balata, contra una superficie metálica giratoria directamente acoplada a la rueda. Cuando se oprime el pedal de freno, un sistema hidráulico compuesto por tubería rígida y de mangeras transmite la presión del líquido desde el cilindro maestro montado debajo del cofre (en la parte posterior del motor) a los cilindros de menor tamaño situados en las 4 ruedas. Los pistones de estos cilindros oprimen las balatas del freno contra los tambores o discos giratorios, por lo que una fuga en cualquier parte del sistema provoca la falla de los frenos. Algunos coches modernos van provistos de doble sistema de frenos, uno para las ruedas delanteras y otro para las traseras, de manera que la falla de uno u otro sistema deja aun dos frenos disponibles.

El uso excesivo del freno se refleja en descomposturas del mismo, las cuales son, por lo general:

- el gasto o aflojamiento de los baleros;
- la pérdida de tensión de los resortes; y,
- el gasto de las balatas o las pastillas.

Estas fallas se detectan con el comportamiento del freno, ya que ofrecerá resistencia, el pedal vibrará, existirá un funcionamiento irregular del mismo, o bien, el coche se desviará al frenar. Entonces, mediante el conocimiento del número de pisadas del freno, se puede establecer una forma de identificar el momento en el que sea propicio realizar una reparación o un cambio en los mismos, para asegurarse de su correcto estado y así prevenir posibles accidentes. El monitoreo de las pisadas de freno se obtiene mediante un interruptor de botón instalado en el pedal del freno.

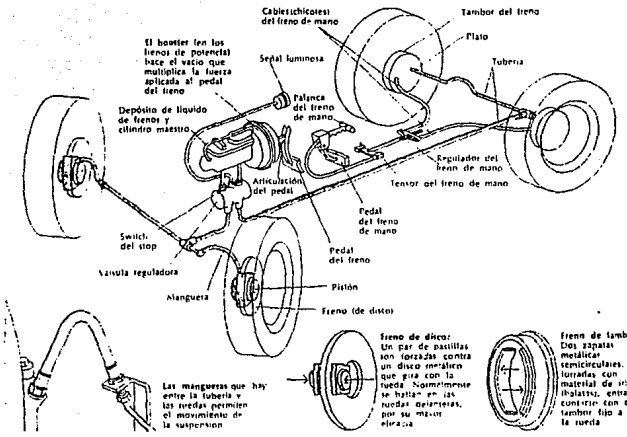


Fig. 6.2 Instalación del sistema de frenos de un automóvil.

Con respecto al sistema, se continúa trabajando y considerando diferentes opciones para hacerlo más accesible y atractivo al mercado. Las opciones inmediatas son: reducir el tamaño del gabinete a la mitad, incorporar una interfase para PC y cambiar la presentación de los diferentes mensajes en los displays; por ejemplo, que sean palabras y frases completas que se van recorriendo a través de los mismos, para hacerlo más "amigable".

Es pequeña la posibilidad de incorporar displays de cristal líquido ó procesadores como el DS5000T, ya que el costo del

equipo, factor que se ha cuidado mucho, se incrementaría en casi el 50%. Ahora bien, no se debe olvidar que, lejos de cualquier ahorro, en algunos casos al cliente le interesa más la apariencia, y ésto lo proporciona un display de cristal líquido.

El uso del microprocesador DS5000T resulta súmamente atractivo, ya que contiene muchos circuitos integrados en un solo encapsulado (microcontrolador, reloj en tiempo real, NVSRAM y circuito de respaldo de energía), sin embargo, requiere de un programador especial, y se tendría que estudiar si resulta más barato comprarlo o construírlo. Además, en el caso de usar el DS5000T, se necesitaría tener una gran cantidad de clientes interesados, ya que sólo así podría obtenerse una ganancia significativa por su costo.

Fueron varias las veces que se grabó un programa en una EPROM, para probar su funcionamiento en el sistema, y al ser un dispositivo sensible a las cargas estáticas, cada ocasión representaba una oportunidad para dañar dicho circuito. Existen en el mercado microcontroladores 8051 con EPROM integrada; sin embargo, se utilizó uno sin EPROM en la construcción y programación del prototipo, ya que era menos costoso comprar una nueva EPROM que un nuevo microcontrolador. Por otra parte, resultó más cómodo contar con varias EPROM y grabarlas conforme fuera necesario, que contar con una sola EPROM (la del microcontrolador), y borrarla cada vez que se requiriera un nuevo programa. Además, fue más sencillo conseguir un programador de EPROM que un programador del microcontrolador 8051.

Sin embargo, una vez definido plenamente el programa a instalar en el sistema, resultaría más conveniente adquirir microcontroladores con EPROM integrada, ya que:

1) Se reduciría el número de componentes del sistema.
2) La tarjeta de circuito impreso se verá reducida en sus dimensiones, lo cual, a la larga, resultaría más económico.

3) El microcontrolador poseé la alternativa de activar un sistema de seguridad, que impide que se lea el programa grabado en su EPROM interna, anulando así la posibilidad de que alguien intente copiar el programa escrito en ella.

Así, el único problema sería conseguir un grabador especial para microcontroladores 8051 (un grabador que, por cierto, está disponible para los estudiantes de la Facultad de Ingeniería).

Un obstáculo que queda por salvar es el desarrollo de sensores de paso de combustible, lo cual implica no sólo tener conocimientos firmes sobre mecánica de fluidos, densidad de combustible, viscosidad, etc., mecánica de materiales y diseño de piezas mecánicas, sino que se requiere también contar con el equipo necesario para este desarrollo, lo cual escapa de nuestras actuales posibilidades. Sin embargo, el sistema electrónico y el software están desarrollados para incorporar estos sensores, ya sea que se adquieran en el mercado y no existan problemas para su uso, o bien, que otras personas los desarrollen a la brevedad posible, ya que el tiempo es factor importante para su incorporación en el mercado.

Capítulo 7

Conclusiones

Este trabajo presenta como resultado un sistema de adquisición de datos en vehículos económico, versátil y sencillo de usar.

Para conseguir dichos objetivos, se tomaron en cuenta la filosofía, la metodología y los aspectos de diseño que se proponen para la construcción de un sistema electrónico. Se realizó una búsqueda profunda de los circuitos electrónicos más adecuados, obteniéndose como resultado varios sistemas constituídos por diferentes componentes.

Indiscutiblemente, dentro del área electrónica, los componentes que resultan más útiles solamente es posible adquirirlos en los Estados Unidos; afortunadamente en México se cuenta con compañías enfocadas exclusivamente a la importación de componentes electrónicos, bajo pedido, aunque resulta más costoso que mandarlos traer directamente. Definitivamente, una de las cualidades más importantes que requiere un ingeniero es la búsqueda de soluciones, formulando el mayor número posible de las mismas. Se tiene la idea de que en esta tesis, dicha búsqueda se efectuó en la forma más adecuada. Siguiendo los criterios de reconocimiento y selección de componentes, se eligieron aquellos que se acercaban más al cumplimiento de los objetivos, y ésto se ve reflejado en el diseño final que se presenta.

Mientras en los sistemas de medición convencionales se utiliza un sensor magnético, para el presente trabajo se construyó uno empleando un fotomicrosensor y un disco ranurado, comprobando que se puede emplear en la medición de velocidad y distancia sin ningún problema.

El sistema diseñado mejora a los disponibles en el mercado, al presentar una opción con gran capacidad de almacenamiento, sencillo de construír y de manejar, y con un costo atractivo. Además brinda la posibilidad de obtener una impresión de la

información recopilada, para llevar un historial del vehículo.

El trabajo construído es solamente un prototipo, y las acciones que ejecuta fueron definidas por los diseñadores; sin embargo, el sistema se puede adaptar a las necesidades del usuario sin ningún problema. Para demostrar ésto se incorporó al escrito un programa desarrollado para registrar 4 parámetros (distancia, rpm, pisadas de freno y consumo de combutible) al mismo tiempo, lo cual demuestra su versatilidad. Gracias a los componentes elegidos, el sistema construído puede emplearse en muchas otras aplicaciones diferentes a la del monitoreo de vehículos, como lo son: sistemas de control que requieren de un reloj para trabajar; adquisición de datos en procesos industriales, etc. Ésto es prueba de la gran versatilidad conseguida en el sistema.

Por otra parte, con el hecho de reducir el tamaño del sistema, incorporando además una interfase para PC, consideramos que puede entrar al mercado proporcionando muchos más grandes beneficios al cliente, a un costo más accesible que el ofrecido por los sistemas actuales.

Apéndice A

Circuitos Impresos

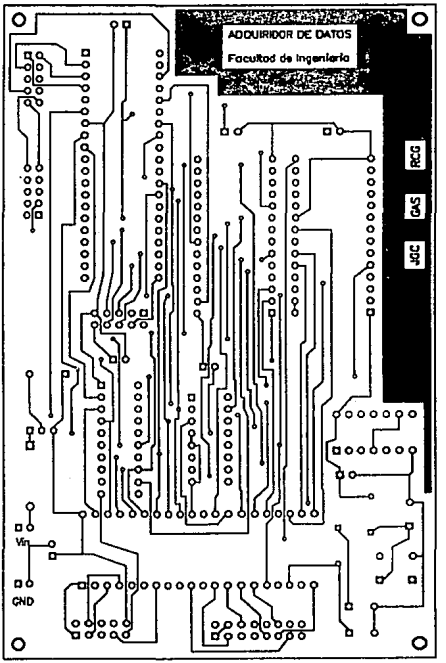


Fig. A.1 Sistema adquiridor de datos, mascarilla de circuito impreso, lado de componenetes.

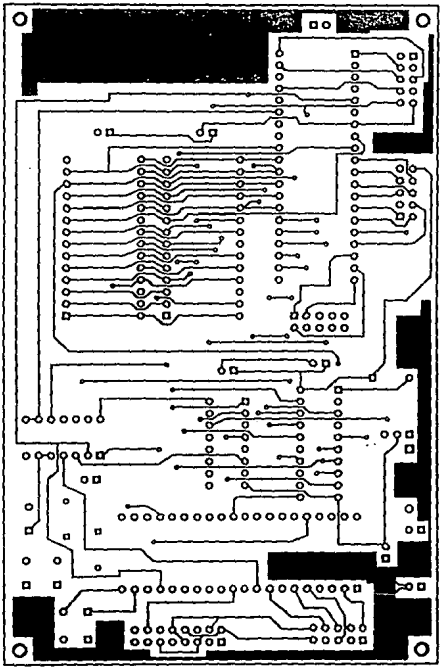


Fig. A.2 Sistema adquiridor de datos, mascarilla de circuito impreso, lado de soldadura.

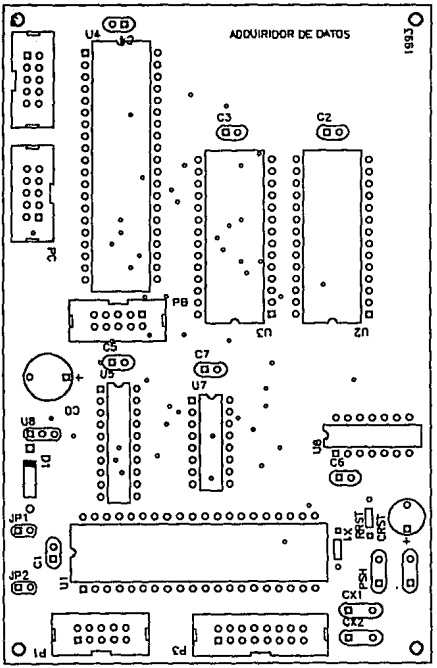


Fig. A.3 Sistema adquiridor de datos, mascarilla de componentes.

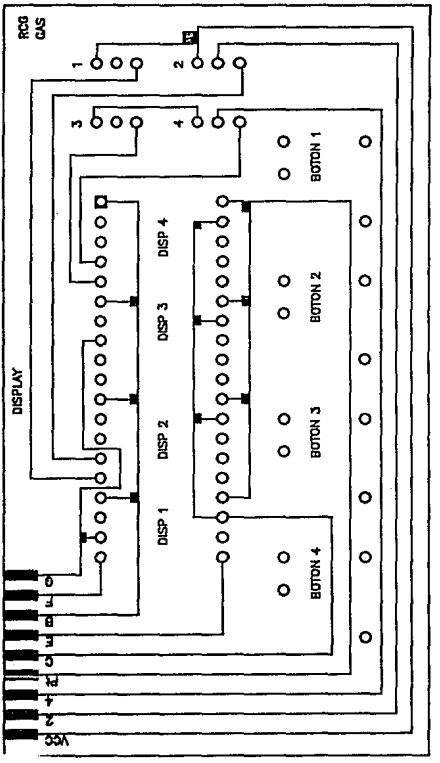


Fig. A.4 Displays del sistema, mascarilla de circuito impreso, lado de componentes.

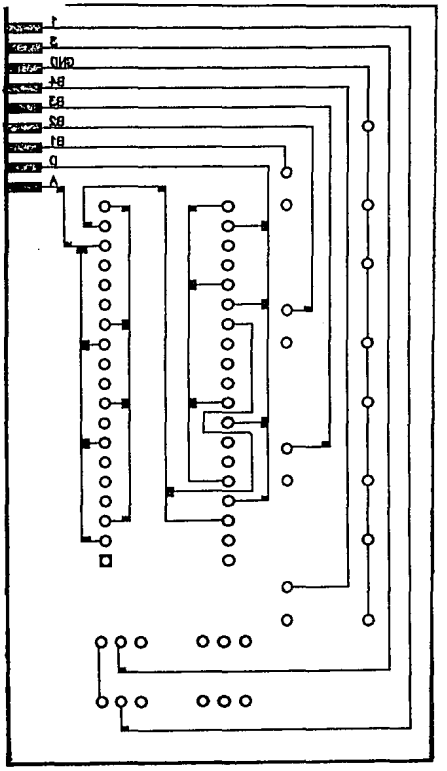


Fig. A.5 Displays del sistema, mascarilla de circuito impreso, lado de soldadura.

Apéndice B

Listados de programas y rutinas de prueba

```
;**  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **
;*                                                            *
;*      PROGRAMA PARA PROBAR TODAS LAS LOCALIDADES             *
;*      DEL RELOJ DE DALLAS, Y PARA PROBAR EL BRILLO DE       *
;*                        LOS DISPLAYS                        *
;*                                                            *
;*                                                            *
;*                    Coras Garcia Ronald                     *
;*                    Sosa Lujano Gilberto Alejandro          *
;**  **  **  **  **  **  **  **  **  **  **  **  **  **  **  **

;------------------- Declaracion de Etiquetas ----------------

VRESET   EQU 00H        ;Lugar Vector de Reset <- 00H
RRESET   EQU 33H        ;Lugar Rutina de Reset <- 33H
PTO_A    EQU 2000H      ;Puerto A
PTO_B    EQU 2001H      ;Puerto B
PTO_C    EQU 2002H      ;Puerto C
PTO_CTRL EQU 2003H      ;Puerto de Control

;-------------------- Declaracion de Rutinas -----------------
         ORG VRESET     ;Vector de Reset
         AJMP RRESET    ;Brinca a Rutina de Reset

;---------------------- Programa Principal --------------------
         ORG RRESET
                        ; Carga valores de 7 segm.
         MOV 40H,#0C0H  ; 0  Dir. 40H
         MOV 41H,#0F9H  ; 1
         MOV 42H,#0A4H  ; 2
         MOV 43H,#0B0H  ; 3
         MOV 44H,#99H   ; 4
         MOV 45H,#92H   ; 5
         MOV 46H,#82H   ; 6
         MOV 47H,#0F8H  ; 7
         MOV 48H,#80H   ; 8
         MOV 49H,#98H   ; 9
                        ; Carga clave de acceso al reloj de Dallas
         MOV 60H,#88H
         MOV 61H,#66H
         MOV 62H,#55H
         MOV 63H,#77H
         MOV 64H,#22H
         MOV 65H,#44H
         MOV 66H,#88H
         MOV 57H,#22H
                        ; Programacion puertos
         MOV DPTR,#PTO_CTRL
         MOV A,#80H     ; Puertos A, B y C de salida
         MOVX @DPTR,A

         MOV R4,#08H    ; Reg 4 <- indica la localidad a leer
         MOV R7,#01H    ; Reg 7 <- Tiempo mínimo (#01H)
         MOV R3,#01H    ; Reg 3 <- Controla el brillo

;----------------- Comienza Rutina Principal ----------------

APRIETA: CALL LLAMA     ; Obtiene informacion del reloj
         CALL DISPLAY   ; Espera a que se oprima un boton
         MOV A,P1       ; Botones alambrados a Pto. 1 del micro
         XRL A,#0FFH
         JZ APRIETA     ; Si NO se oprimio boton: Brinca a APRIETA

         MOV 3CH,A      ; Dir 3CH <- Guarda el boton.
         ANL A,#04H     ; Transforma boton a BCD
         JZ AQUI1
         MOV 3CH,#03H
AQUI1:   MOV A,3CH
         ANL A,#08H
         JZ SUELTA
         MOV 3CH,#04H ; Dir 3CH <- Guarda boton en BCD
```

```
SUELTA:  CALL LLAMA    ; Obtiene informacion del reloj
         CALL DISPLAY  ; Espera a que se suelte el boton
         MOV A,P1
         XRL A,#0FFH
         JNZ SUELTA    ; Si NO se ha soltado boton: Brinca a
                       ;  SUELTA
         MOV A,3CH     ; Si oprimio boton 4:
         XRL A,#04H    ;  Se ubicara una localidad inferior
         JZ RESTA
         MOV A,3CH     ; Si oprimio boton 1:
         XRL A,#01     ;  Se ubicara una localidad superior
         JZ SUMA
         MOV A,3CH     ; Si oprimio boton 2:
         XRL A,#02H    ;  Aumenta el brillo
         JNZ BAJBRIL   ;
         INC R3
         AJMP APRIETA
BAJBRIL: DEC R3
         AJMP APRIETA

; Para Sumar dos localidades (8 bits) de informacion:
SUMA:    MOV A,R4      ; Si esta en la ultima localidad superior:
         XRL A,#08H    ;    ya no suma
         JNZ SISUMA    ;  si no
         AJMP APRIETA  ;    suma 8 (se ubica en la sig. local.)
SISUMA:  MOV A,R4
         ADD A,#01H
         MOV R4,A
         AJMP APRIETA

; Para Restar dos localidades de informacion (8 bits):
RESTA:   MOV A,R4      ; Si esta en la ultima localidad inferior:
         XRL A,#00H    ;    ya no resta
         JNZ SIRESTA   ;  si no
         AJMP APRIETA  ;    resta 8 (se ubica en local. inferior)
SIRESTA: MOV A,R4
         CLR C
         SUBB A,#01H
         MOV R4,A
         AJMP APRIETA

;************ Rutina: Lee_informacion_del_reloj **************
; Con esta rutina se lee la informacion del reloj de Dallas,
; se convierte a 7 segmentos y se guarda en las direcciones
; 38H a 3BH.

LLAMA:   CALL ACCESO   ; Llama a rutina de acceso al reloj
         MOV A,R4      ; Reg A <- Reg 4
         MOV R0,A      ; Reg 0 <- Reg A (valores a brincar)
         XRL A,#00H
         JZ HERE05
HERE0:   MOVX A,@DPTR
         DJNZ R0,HERE0
HERE05:  MOVX @DPTR,A
         MOV R0,#57H   ; Reg 0 <- Dir. que contendra el dato
HERE1:   MOV R1,#01H   ; Reg 1 <- # de datos que se leeran
HERE2:   MOVX A,@DPTR  ; Escribe los valores en cuestion
         SWAP A
         ANL A,#0FH
         MOV @R0,A
         DEC R0
         MOV A,R0
         XRL A,#53H
         JNZ HERE1
         MOV A,#08H
         CLR C
         SUBB A,R4
         MOV R0,A
         XRL A,#00H
```

```
        JZ HERE35
HERE3:  MOVX A,@DPTR
        DJNZ R0,HERE3
HERE35: MOV R2,#04H   ; Conversion de numeros a 7 segmentos
HERE4:  MOV A,R2
        ADD A,#53H    ; Guarda los valores en 7 segmentos
        MOV R1,A      ; en las dirs. 38H a 3BH.
        MOV A,R2
        ADD A,#37H
        MOV R0,A
        MOV A,@R1
        ADD A,#40H
        MOV R1,A
        MOV A,@R1
        MOV @R0,A
        DJNZ R2,HERE4
        RET

;****************** Rutina: Acceso_al_Reloj ********************
;       Rutina para entrar al reloj de Dallas
ACCESO: MOV DPTR,#0000H
        MOVX A,@DPTR
        MOV R0,#57H     ; Rutina acceso a reloj Dallas
ALL11:  MOV 2H,@R0
        MOVX A,@DPTR
        ANL A,R2
        MOVX @DPTR,A
        DEC R0
        MOV A,R0
        XRL A,#5FH
        JNZ ALL11
        RET

;****************** Rutina: Muestra_en_Display ***************
; Rutina para mostrar informacion en displays de 7 segm.
; Muestra la informacion contenida en las direcciones 38H a 3BH.

DISPLAY: MOV R0,#38H
        MOV R1,#01H
        MOV R2,#04H     ; R4 <- Contador
DENUEVO: MOV DPTR,#PTO_A
        MOV A,@R0
        MOVX @DPTR,A    ; Displays conectados al pto. A
        MOV DPTR,#PTO_C
        MOV A,R1
        MOVX @DPTR,A    ; Seleccion de display en pto. C
        CALL ESPERA
        MOV A,#00H
        MOVX @DPTR,A
        INC R0
        MOV A,R1
        RL A
        MOV R1,A
        DJNZ R2,DENUEVO
        DJNZ R7,DISPLAY
        MOV R7,#01H
        RET

;****************** Rutina: Retardo_de_Tiempo ****************
ESPERA:  MOV R6,#0FFH ; Usa R5 y R6 para contar
AQUI7:   MOV A,R3     ; 1 ciclo
         MOV R5,A     ; 1 ciclo
AQUI8:   NOP          ; 1 ciclo
         NOP          ; 1 ciclo
         NOP          ; 1 ciclo
         DJNZ R5,AQUI8 ; 2 ciclos
         DJNZ R6,AQUI7 ; 2 ciclos
         RET
         END;
```

```
;******* Rutina para imprimir informacion en la sumadora. ******

;                          Coras Garcia Ronald
;                          Sosa Lujano Gilberto Alejandro

VRESET   EQU 00H
RRESET   EQU 33H
PTO_A    EQU 2000H
PTO_B    EQU 2001H
PTO_C    EQU 2002H
PTO_CTRL EQU 2003H
HAS      EQU 0AH
TOTAL    EQU 0BH
PAPEL    EQU 0CH

         ORG VRESET
         AJMP RRESET

         ORG RRESET
                  ; Programacion PPI
         MOV DPTR,#PTO_CTRL
         MOV A,#80H        ;Ptos A, B y C de salida
         MOVX @DPTR,A

         MOV DPTR,#PTO_B ;Deshabilita sumadora
         MOV A,#0FFH
         MOVX @DPTR,A

                  ; Guarda informacion a imprimir
         MOV 50H,#12H  ; Se guarda de la dir 50H a la 59H
         MOV 51H,#34H
         MOV 52H,#56H
         MOV 53H,#78H
         MOV 54H,#9DH
         MOV 55H,#0D3H
         MOV 56H,#44H
         MOV 57H,#0D7H
         MOV 58H,#81H
         MOV 59H,#99H

         MOV R7,#0FFH  ; Retardo de tiempo


DISPLAY: MOV DPTR,#PTO_A ;Muestra "C" en displays
         MOV A,#0C6H
         MOVX @DPTR,A   ;Displays conectados al pto. A
         MOV DPTR,#PTO_C
         MOV A,#01H
         MOVX @DPTR,A   ;Seleccion de display en pto. C
         CALL ESPERA
         MOV A,#00H
         MOVX @DPTR,A

         MOV R3,#03H ;Reg 3 guarda No. Total cifras a imprimir
         MOV R0,#4FH ;Reg 0 <- apunta a informacion

         MOV DPTR,#PTO_B ;Impresora conectada al Pto B del PPI

AGUI0:   MOV R2,#03H  ;Reg 2 guarda el # de par de Nos. a
                      ; imprimir
AGUI1:   INC R0
AGUI2:   MOV A,@R0
         JB 0,AGUI3
         SWAP A
AGUI3:   ANL A,#0FH
         MOVX @DPTR,A ;Escribe num. en calculadora
         CALL ESPERA
         CPL 0
         MOV A,#0FFH  ;Deshabilita calculadora
```

```
        MOVX @DPTR,A
        CALL ESPERA
        CALL ESPERA
        JB 0,AQUI2
        DJNZ R2,AQUI1

        MOV A,#MAS
        MOVX @DPTR,A  ;Manda imprimir
        CALL ESPERA
        MOV A,#0FFH   ;Deshabilita calculadora
        MOVX @DPTR,A
        CALL ESPERA
        MOV A,#PAPEL
        MOVX @DPTR,A  ;Deja un renglon
        CALL ESPERA
        MOV A,#0FFH   ;Deshabilita calculadora
        MOVX @DPTR,A
        ACALL ESPERA
        ACALL ESPERA
        ACALL ESPERA
        ACALL ESPERA
        DJNZ R3,AQUI0
        AJMP DISPLAY

;****************** Rutina: Retardo_de_Tiempo *****************
; Se debe especificar un numero en Reg 7, antes de entrar a esta
; rutina. Los valores recomendados son los siguientes:
;     01H <- Mostrar numeros en display
;     080H <- Enviar numeros a la sumadora
;     0FFH <- Dejar un renglon en impresora
;

ESPERA:  MOV R6,#0FFH  ; Usa Regs 6 y 7 para contar
AQUI6:   NOP           ; 1 us
         NOP           ; 1 us
         NOP           ; 1 us
         DJNZ R6,AQUI6 ; 2 us
         DJNZ R7,ESPERA; 2 us
         MOV R7,#0FFH  ; Retardo de tiempo para sumadora
         RET

         END;
```

```
;*********************************
;*    Programa para contar el número de vueltas que dan    *
;*   los engranes del sensor de combustible. Puede usarse  *
;*    para contar el número de vueltas de cualquier sensor. *
;*                                                          *
;*                    Coras Garcia Ronald                   *
;*                    Sosa Lujano Gilberto Alejandro        *
;*                                                          *
;*********************************

;-------------------- Zona de Equivalencias ----------------

VRESET   EQU 00H     ; Direccion del Vector de Reset
RRESET   EQU 33H     ; Direccion de la Rutina de Reset
VINTO    EQU 03H     ; Direccion del Vector de Interrupcion
PIO_A    EQU 2000H   ; Ubicacion del Pto. A del PPI
PIO_B    EQU 2001H   ; Ubicacion del Pto. B del PPI
PIO_C    EQU 2002H   ; Ubicacion del Pto. C del PPI
PIO_CTRL EQU 2003H   ; Ubicacion del Pto. de Control del PPI

;-------------------- Declaracion de rutinas ---------------

         ORG VRESET
         AJMP RRESET


;*** La cuenta del numero de vueltas que registra el sensor
;*** de combustible se lleva con ayuda de la interrupcion 0 del
;*** microcontrolador (entrada 2 del puerto 3).

         ORG VINTO      ; Cuenta vueltas sensor combustible
         INC 69H        ; Dir. 69H <- Guarda la cuenta
         RETI


;-------------------- Inicio de Programa -------------------

         ORG RRESET

                        ;Carga clave de acceso al reloj
         MOV 30H,#88H   ; La clave se guarda en las Dirs 30H a 37H
         MOV 31H,#66H
         MOV 32H,#55H
         MOV 33H,#77H
         MOV 34H,#22H
         MOV 35H,#44H
         MOV 36H,#88H
         MOV 37H,#22H
                        ;Carga valores de 7 segmentos
         MOV 40H,#0C0H ; 0  Dir. =CH
         MOV 41H,#0F9H ; 1
         MOV 42H,#0A4H ; 2
         MOV 43H,#0B0H ; 3
         MOV 44H,#99H  ; 4
         MOV 45H,#92H  ; 5
         MOV 46H,#82H  ; 6
         MOV 47H,#0F8H ; 7
         MOV 48H,#80H  ; 8
         MOV 49H,#98H  ; 9

         MOV DPTR,#PIO_CTRL  ;Definicion de puertos del PPI:
         MOV A,#80H          ; Los Ptos. A, B y C se definen como
         MOVX @DPTR,A        ; puertos de salida.

         CLR 0          ; Limpia Bit 0 (Indica monitoreo)
                        ;  "0" <- Indica Inicio del Sistema
                        ;  "1" <- Indica Ejecucion del Monitoreo

         CLR 1          ; Limpia Bit 1 (Indica cumplimiento 1 seg)
                        ;  "0" <- Aun no transcurre el seg
```

```
                        ;  "1" <- Ha transcurrido 1 seg

        CLR 2           ; Limpia Bit 2 (Consumo Instantaneo/Total)
                        ;  "0" <- Consumo Instantaneo
                        ;  "1" <- Consumo Total

        CALL BORRA      ; Limpia localidades de memoria usadas
                        ; para llevar la cuenta.


;*** Muestra pantalla inicial, esperando que el usuario
;*** seleccione la opcion deseada.

                        ; Carga en Displays de 7 segm. la
                        ;  pantalla que indica las opciones:

                        ; que indica las opciones:
BIENVEN: MOV 38H,#0EFH  ;   Inicia cuenta
        MOV 39H,#0ABH   ;   muestra iNformacion
        MOV 3AH,#0FFH   ;
        MOV 3BH,#83H    ;   Borra informacion

PIDE:   CALL APRIETA    ; Espera a que se apriete un boton
                        ; Dir 3CH <- Boton que se apreto
        MOV A,3CH
        XRL A,#01H      ; Si oprimio boton 1, brinca a INICIA
        JZ INICIA       ;   (Inicia monitoreo)
VEBOT2: MOV A,3CH       ; Si oprimio boton 2, brinca a MUESTRA
        XRL A,#02H      ;   (Muestra los resultados)
        JZ IMPRES
VEBOT3: MOV A,3CH       ; Si oprimio boton 4, brinca a BORRA
        XRL A,#04H      ;   (Borra informacion almacenada)
        JZ BORRA
        AJMP BIENVEN    ; Si oprimio boton 3, regresa a PIDE
                        ;   (El boton 3 no tiene asignada
                        ;         ninguna rutina)


;*** Este es el inicio de la rutina de Monitoreo.
;*** Este programa esta adecuado para calibrar el medidor de
;*** combustible.

INICIA: MOV 3EH,#00H  ; Dir 3EH <- Para verificar cambio de seg

                      ;Asigna valores normales a los contadores

        MOV 59H,#02H  ; Hoyos/Lt : 02H (Dir 59H) (Hoyos expres. en Hex)
        MOV IE,#10000001B     ; Enciende interrupciones
        MOV TCON,#00000001B   ; Interrupcion por flanco
                              ; Enciende contadores

        SETB 0        ; Ejecucion del monitoreo

;*** Monitoreo.....

REGIS:  CALL APRIETA  ; Espera a que se apriete un boton
                      ; Dir 3CH <- Boton que se apreto

        CLR 2         ; Display muestra RPH's
        MOV A,3CH     ; Si oprimio boton 1, brinca a BIENVEN
        XRL A,#01H    ;   (Fin del monitoreo)
        JNZ REGIS     ; Si oprimio cualquier otro boton,
        CLR 0         ;   brinca a REGIS (continua Monitoreo)
        MOV IE,#00H   ; Deshabilita interrupciones y contadores.
        AJMP BIENVEN

;*** Rutina para mostrar en los displays el consumo total de
;*** combustible.
```

```
IMPRES:  MOV R0,#6FH     ; El consumo total de combustible esta
         CALL SEGMEN     ; en la Dir 6FH
         AJMP PIDE

;*** Rutina para borrar el consumo total de combustible que se
;*** tiene registrado.

BORRA:   MOV 6FH,#00H    ; Dirs 6E y 6FH <- Consumo total de combustible
         MOV 6EH,#00H
         MOV 6BH,#00H    ; Dirs 6A y 6BH <- Consumo en 1 seg.
         MOV 6AH,#00H
         MOV 69H,#00H    ; Dir 69H <- Lleva Cuenta
         AJMP BIENVEN


;********************* Rutina: Aprieta_boton *****************
;              Espera a que el usuario oprima un boton.
;                Botones alambrados al Pto. 1 del uC
;          El boton seleccionado se guarda en la Dir. 3CH

APRIETA: CALL DISPLAY    ; Despliega informacion 7 segmentos
         JNB 0,VEBOTON   ; Si esta en Inicio del Sistema,
                         ;    brinca a VEBOTON
         CALL UNSEG      ; Llama a contador de 1 segundo
         JNB 1,VEBOTON   ; Si aun no transcurre 1 segundo,
                         ;    brinca a VEBOTON
         CALL ACTUAL     ; Actualiza informacion

VEBOTON: MOV A,P1
         XRL A,#0FFH     ; Si no se ha apretado boton,
         JZ APRIETA      ;    brinca a APRIETA

         MOV 3CH,A       ; Dir 3CH <- Guarda el boton.
         ANL A,#04H      ; Transforma boton a BCD
         JZ NOES3
         MOV 3CH,#03H
NOES3:   MOV A,3CH
         ANL A,#08H
         JZ SUELTA
         MOV 3CH,#04H    ; Dir 3CH <- Guarda boton en BCD
         SETB 2          ; Si se oprimio el boton 4, se mostrara
                         ; el Consumo Total de Combustible

SUELTA:  CALL DISPLAY    ;Espera a que se suelte el boton
         JNB 0,VEBOT     ; Si esta en Inicio del Sistema,
                         ;    brinca a VEBOT
         CALL UNSEG      ; Llama a contador de 1 segundo
         JNB 1,VEBOT     ; Si aun no transcurre 1 segundo,
                         ;    brinca a VEBOT
         CALL ACTUAL     ; Actualiza informacion
VEBOT:   MOV A,P1
         XRL A,#0FFH     ; Si no se ha soltado el boton
         JNZ SUELTA      ;    brinca a SUELTA
FIN:     RET


;***************** Rutina: Muestra_en_Display ****************
;    Rutina para mostrar informacion en displays de 7 segm.

DISPLAY: MOV R0,#38H     ; R0 <- Dir. informacion disp. 7 segm
         MOV R2,#01H     ; R2 <- Para habilitar display
         MOV R4,#04H     ; R4 <- Numero de numeros a desplegar.
DENUEVO: MOV DPTR,#PTO_A
         MOV A,@R0
         MOVX @DPTR,A    ; Displays conectados al Pto. A del PPI
         MOV DPTR,#PTO_C
         MOV A,R2
         MOVX @DPTR,A    ; Seleccion de display en Pto. C del PPI
         CALL ESPERA     ; Espera un rato
```

```
        INC R0          ; R0 <- Apuntara al sig. digito
        MOV A,R2
        RL A
        MOV R2,A
        DJNZ R4,DENUEVO
        RET

;******************** Rutina: Retardo_de_Tiempo ****************
;  Se realizaron pruebas sobre el brillo del display con el
;  programa BRILREL.ASM, y  se encontro un brillo optimo
;  para los valores:
;          Reg 6 <- #0FFH
;          Reg 5 <- #03H
;  Sin embargo, despues de ver el funcionamiento de este programa
;  se llego a la conclusion de que el brillo del los displays
;  variara de acuerdo con la rutina que se este llevando a cabo,
;  por lo que el valor a asignar a R5 y R6 variara... es decir,
;  los valores de R5 y R6 se deberan asignar antes de entrar
;  a la rutina.

ESPERA:  MOV R6,#0FFH ; Usa R5 y R6 para contar
AQUI7:   MOV R5,#03H  ; 1 ciclo
AQUI8:   NOP          ; 1 ciclo
         NOP          ; 1 ciclo
         NOP          ; 1 ciclo
         DJNZ R5,AQUI8 ; 2 ciclos
         DJNZ R6,AQUI7 ; 2 ciclos
         RET


;****************** Rutina: Espera_Un_Segundo ******************
;               Espera a que transcurra 1 segundo

UNSEG:   CALL ACCESO    ; Llama a rutina de acceso al reloj

                        ;Rutina lectura de fecha y hora
HERE0:   MOVX A,@DPTR   ; Se brinca 1/100 y 1/10 seg

HERE2:   MOVX A,@DPTR
         MOV 3FH,A      ; Dir. 3FH <- Segundo en cuestion

         MOV R0,#06H    ;Culmina los ciclos de acceso al reloj
HERE3:   MOVX A,@DPTR
         DJNZ R0,HERE3

         MOV A,3FH      ; Reg A <- Segundo en cuestion
         XRL A,3EH      ; Compara con el segundo registrado,
         JZ NOSEG       ; y si no ha cambiado:  Brinca a NOSEG

         MOV 3EH,3FH    ; Actualiza valor Dir. 3EH
         SETB 1         ; Indica que ya se cumplio un segundo
NOSEG:   RET

;****************** Rutina: Acceso_al_Reloj ******************
;              Rutina para tener acceso al reloj de Dallas

ACCESO:  MOV DPTR,#0000H
         MOVX A,@DPTR   ; Inicia apuntador reloj Dallas

         MOV R0,#37H
ALLI1:   MOV 2H,@R0
         MOVX A,@DPTR
         ANL A,R2
         MOVX @DPTR,A
         DEC R0
         MOV A,R0
         XRL A,#2FH
         JNZ ALLI1
         RET
```

```
;**************** Rutina: Actualiza_Informacion **************
;       Guarda la cuenta en su lugar correspondiente.

ACTUAL:  CLR 1          ; Limpia bit que indica transcurso 1 seg.

                        ;Actualiza consumo de combustible.
         MOV B,59H      ; Reg B <- Hoyos/lt
         MOV A,69H      ; Reg A <- No. hoyos leidos (TL0)
         DIV AB         ; Reg A <- Consumo en 1 seg, en Hex
         MOV 68H,A      ; Dir 68H <- Guarda Consumo
         MOV 6AH,#00H   ; Dir 6AH <- Ceros
         MOV 69H,B      ; Dir 69H <- Guarda vueltas no
                        ;   consideradas en el calculo anterior

MOVIOL:  CALL HEXABCD   ; Convierte Consumo de Hex a BCD

         MOV R1,#6FH    ; Reg 1 <- Dir donde se guarda la Suma
         CALL SUMABCD   ; Suma consumo a la cuenta total

; Actualiza consumo en la memoria de Dallas

         MOV R1,#02H    ; El consumo se guarda en Dirs. 12H y 13H
         MOV R0,#6EH
         MOV DPTR,#0012H
ACTVEL:  MOV A,@R0
         MOVX @DPTR,A
         INC DPTR
         INC R0
         DJNZ R1,ACTVEL

         MOV R0,#68H    ; Mostrara Consumo Instant. de Combustible
         JNB 2,CALSEGM  ; Si bit 2 esta encendido
         MOV R0,#6FH    ;    Mostrara entonces Consumo Total

CALSEGM: CALL  SEGMEN   ; Actualiza displays 7 segmentos

         RET

;************ Rutina: Convierte_de_Hexadecimal_a_BCD **********
; Convierte de Hexadecimal a BCD el dato localizado en la Dir 68H
; de la memoria interna del uC.

HEXABCD: MOV A,68H   ; Dir 68H <- Consumo en Hexadecimal
         SWAP A
         ANL A,#0FH
         MOV R2,A    ; Reg 2 <- Guarda el nibble mas significativo
         MOV R5,#06H ; Reg 5 <- Guarda dato a sumar (es #06H)
         MOV A,68H
         ANL A,#0F0H ; Si consumo > 10H, brinca a AJUSTE
         JNZ AJUSTE
         MOV R2,#01H
         MOV R5,#00H ; Reg 5 <- Cambia su dato a #00H
AJUSTE:  MOV A,68H
SUMA:    ADD A,R5
         DA A
         JNC NOOVER  ; Dir 6AH <- Centenas
         INC 6AH
NOOVER:  DJNZ R2,SUMA
         MOV 68H,A   ; Dir 68H <- Guarda el resultado
         RET

;************ Rutina: Suma_Dos_Numeros_en_BCD ***************
; Esta rutina suma el numero contenido en la Dir 68H, al total
; guardado en la Dirs. 6EH a 6FH.
; Dir 69H <- Contiene el dato a sumar
; Reg 0 <- Guarda el dato a sumar, en el instante
; Reg 1 <- Apunta a la Dir. en la que va la suma.
; Reg 2 <- Contador del No. de bytes que faltan por sumar
; Reg 3 <- Ayuda para hacer la suma (Dir 03H)
```

```
; Bandera 3 <- Lleva el "carry"
; Bandera 4 <- Ve si es nibble 0 o 1 (Der. o Izq.)
; Dir 6EH a 6FH <- Guarda la suma total.
SUMA8CD: CLR 3          ; Limpia bandera "carry"
         CLR 4          ; Limpia bandera "nibble"
         MOV R2,#02H    ; Reg 2 <- No. bytes a sumar
         MOV R0,68H     ; Reg 0 <- Dato a Sumar (Contenido Dir 68H)
SUM8CD:  MOV A,R0
         JNB 4,AQUI0    ; Ve si hay intercambio de nibbles
         SWAP A
AQUI0:   ANL A,#0FH     ; Obtiene nibble menos signif. dato a sumar
         MOV R3,A       ; Guarda nibble en Reg 3
         MOV A,@R1
         JNB 4,AQUI1    ; Ve si hay intercambio de nibbles
         SWAP A
AQUI1:   ANL A,#0FH     ; Obtiene nibble menos signif. de suma total
         ADD A,R3       ; Suma los nibbles
         DA A           ; Ajuste decimal
         JNB 3,AQUI2    ; Si no hay "carry" anterior brinca a AQUI2
         ADD A,#01H     ; Suma "carry"
         DA A           ; Ajuste decimal
         CLR 3          ; Limpia "carry"
AQUI2:   MOV R3,A       ; Reg 3 <- Suma parcial
         ANL A,#0F0H
         JZ AQUI3       ; Si hay carry,    Bit 3 <- 1
         SETB 3         ; Si NO hay carry, Bit 3 <- 0
AQUI3:   ANL 03H,#0FH   ; Reg 3 <- Reg 3 .AND. #0FH
         MOV A,@R1      ; Reg A <- Dato antiguo (suma total)
         JNB 4,AQUI4    ; Ve si hace intercambio de nibbles
         SWAP A
AQUI4:   ANL A,#0F0H    ; Actualiza suma total
         ORL A,R3
         JNB 4,AQUI5    ; Ve si hace intercambio de nibbles
         SWAP A
AQUI5:   MOV @R1,A
         CPL 4          ; Complementa bandera de "nibble"
         JB 4,SUM8CD
         DEC R1
         MOV R0,6AH  ; Reg 0 <- Dir 6AH (Nuevo dato a sumar)
         DJNZ R2,SUM8CD
         RET

;*************** Rutina: Conversion a 7 segmentos ************
;         Rutina para actualizar el valor que aparecera
;                en los displays de 7 segmentos
;     Reg 0 <- 6FH, Para mostrar Consumo Total de Combustible
;    Reg 0 <- 61H, Para mostrar Consumo Parcial de Combustible
;     El valor de Reg 0 se asigna antes de entrar a la Rutina.
SEGMEN:  MOV R2,#02H    ; Reg 2 <- No. bytes a convertir
         MOV R3,#3BH    ; Reg 3 <- Lugar en que almacena
         CLR 3          ; Aquí bit 3 indica si hay SWAP o no
CONVER:  MOV A,@R0      ; Reg 0 <- Dato que se mostrara en displays
         JNB 3,CONT
         SWAP A
CONT:    ANL A,#0FH
         ADD A,#40H
         MOV R1,A       ; Reg 1 <- Dir del valor en 7 segmentos
         MOV A,@R1
         MOV R4,A       ; Reg 4 <- Valor en 7 segmentos
         MOV A,R3       ; Reg A <- Dir en que almacenara el valor
         MOV R1,A
         MOV @R1,04H
         DEC R3
         CPL 3
         JB 3,CONVER
         DEC R0
         DJNZ R2,CONVER
         RET
         END;
```

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *      Programa para medir: rpm's, distancia recorrida,       *
; *     no. de pisadas del freno y consumo de combustible..     *
; *                                                             *
; *                          Coras Garcia Ronald                *
; *                          Sosa Lujano Gilberto Alejandro     *
; *                                                             *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

;--------------------- Zona de Equivalencias -------------------
VRESET   EQU 00H     ; Direccion del Vector de Reset
RRESET   EQU 33H     ; Direccion de la Rutina de Reset
VINT0    EQU 03H     ; Direccion del Vector de Interrupcion 0
VINT1    EQU 13H     ; Direccion del Vector de Interrupcion 1
PTO_A    EQU 2000H   ; Ubicacion del Pto. A del PPI
PTO_B    EQU 2001H   ; Ubicacion del Pto. B del PPI
PTO_C    EQU 2002H   ; Ubicacion del Pto. C del PPI
PTO_CTRL EQU 2003H   ; Ubicacion del Pto. de Control del PPI

;--------------------- Declaracion de rutinas ------------------

;*** Rutina principal...

         ORG VRESET
         AJMP RRESET

;*** La cuenta del numero de vueltas que registra el sensor
;*** de combustible se lleva con ayuda de la interrupcion 0 del
;*** microcontrolador (entrada 2 del puerto 3).

         ORG VINT0       ; Cuenta vueltas del medidor de combustible
         INC 69H         ; Dir. 69H <- Guarda la cuenta
         RETI

;*** Se utiliza la interrupcion 1 para llevar la cuenta del
;*** numero de pisadas del freno (entrada 3 del puerto 3).

         ORG VINT1
         MOV A,57H     ; La cuenta se lleva en las Dirs. 62H, 63H
         ADD A,#01H    ;  y 64H.
         DA A          ; La cuenta se lleva directamente en BCD.
         MOV 57H,A
         JNC REGRESO
         MOV A,56H
         ADD A,#01H
         DA A
         MOV 56H,A
         JNC REGRESO
         MOV A,55H
         ADD A,#01H
         DA A
         MOV 55H,A

REGRESO: RETI

;--------------------- Inicio de Programa ----------------------
         ORG RRESET

;*** Prepara el sistema para trabajar

                       ;Carga clave de acceso al reloj
         MOV 30H,#88H  ; La clave se guarda en las Dirs 30H a 37H
         MOV 31H,#66H  ; de la memoria del uC.
         MOV 32H,#55H
         MOV 33H,#77H
         MOV 34H,#22H
         MOV 35H,#44H
         MOV 36H,#88H
         MOV 37H,#22H
```

```
                    ;Carga valores de 7 segmentos
        MOV 40H,#0C0H ; 0  Dir. 40H
        MOV 41H,#0F9H ; 1
        MOV 42H,#0A4H ; 2
        MOV 43H,#0B0H ; 3
        MOV 44H,#99H  ; 4
        MOV 45H,#92H  ; 5
        MOV 46H,#82H  ; 6
        MOV 47H,#0F8H ; 7
        MOV 48H,#80H  ; 8
        MOV 49H,#98H  ; 9

        MOV DPTR,#PTO_CTRL  ;Definicion de puertos del PPI:
        MOV A,#80H          ; Los Ptos. A, B y C se definen como
        MOVX @DPTR,A        ; puertos de salida.

        CLR 0           ; Limpia Bit 0 (Indica monitoreo)
                        ;  "0" <- Indica Inicio del Sistema
                        ;  "1" <- Indica Ejecucion del Monitoreo

        CLR 1           ; Limpia Bit 1 (Indica cumplimiento 1 seg)
                        ;  "0" <- Aun no transcurre el seg
                        ;  "1" <- Ha transcurrido 1 seg

        CLR 2           ; Limpia Bit 2 (Informacion a mostrar)
                        ;  "0" <- Muestra RPM's
                        ;  "1" <- Muestra Consumo Instan. de
                        ;         Combustible

;*** Actualiza la informacion de la memoria interna del uC,
;*** copiandola de la memoria de Dallas.

        ACALL ACTuC

;*** Muestra pantalla inicial, esperando que el usuario
;*** seleccione la opcion deseada.

                        ; Carga en Displays de 7 segm. la
                        ; pantalla que indica las opciones:
BIENVEN: MOV 38H,#0EFH  ;   Inicia cuenta
        MOV 39H,#0ABH   ;   muestra INformacion
        MOV 3AH,#0FFH   ;
        MOV 3BH,#83H    ;   Borra informacion

PIDE:   ACALL APRIETA   ; Espera a que se apriete un boton
                        ; Dir 3CH <- Boton que se apreto
        MOV A,3CH
        XRL A,#01H      ; Si oprimio boton 1, brinca a
        JZ INICIA       ;       INICIA (Inicia monitoreo)
VEBOT2: MOV A,3CH       ; Si oprimio boton 2, brinca a
        XRL A,#02H      ;       MUESTRA (Muestra los resultados)
        JZ IMPRES       ; Si oprimio boton 4, brinca a
VEBOT3: MOV A,3CH       ;       BORRA (Borra informacion
                        ;         almacenada)
        XRL A,#04H      ; Si oprimio boton 3, regresa a PIDE
        JZ BORRA        ;    (El boton 3 no tiene asignada
        AJMP BIENVEN    ;          ninguna rutina)

;*** Inicio de la rutina de Monitoreo.

INICIA:  MOV 3EH,#00H  ; Dir 3EH <- Para verificar cambio de seg

        MOV TL0,#00H    ;Limpia regs. Timer/Counters
        MOV TL1,#00H    ; Contador 0: Distancia Recorrida
        MOV TH0,#00H    ; Contador 1: RPMs
        MOV TH1,#00H

             ;*** Actualiza ajustes para cuentas registradas.
```

```
        MOV 50H,#02H  ; Hoyos/mt : 02H (Dir 50H) (Hoyos expres.
                      ;            en Hex)
        MOV 51H,#02H  ; Hoyos/Lt : 02H (Dir 51H) (Hoyos expres.
                      ;            en Hex)
        MOV 52H,#02H  ; Hoyos/rpm : 02H (Dir 52H) (Hoyos expres.
                      ;             en Hex)
        MOV IE,#10000101B    ; Enciende interrupciones
        MOV TMOD,#01010101B  ; Enciende contadores
        MOV TCON,#01010101B  ; Interrupciones por flanco

        SETB 0          ; Indica Ejecucion del monitoreo


;*** Monitoreo.....

REGIS:  ACALL APRIETA  ; Espera a que se apriete un boton
                       ; Dir 3CH <- Boton que se apreto

        MOV A,3CH      ; Si oprimio boton 1,
        XRL A,#01H     ;    Deshabilita interrupciones y
                       ;       contadores.
        JNZ REGIS      ;    Brinca a BIENVEN (Fin del monitoreo)
        CLR C          ; Si oprimio cualquier otro boton,
        MOV IE,#00H    ;    Brinca a REGIS (continua Monitoreo)
        MOV TCON,#00H
        AJMP BIENVEN

;*** Rutina para mostrar en los displays la informacion
;*** registrada.

IMPRES: ACALL ACTuC
        MOV R4,#04H    ; Reg 4 <- No. de Datos a Mostrar
        MOV R0,#5FH    ; Dir 5FH <- Distancia Recorrida
        SJMP ACTDISP

SIGDATO: ACALL APRIETA  ; Espera a que se apriete un boton
                        ; Dir 3CH <- Boton que se apreto
        MOV A,3CH
        XRL A,#02H     ; Si NO se apreto Boton 2
        JNZ SIGDATO    ;    Brinca a SIGDATO

        MOV A,R4
        XRL A,#03H
        JNZ ESFRENO
        MOV R0,#6FH    ; Dir 6FH <- Consumo Total de Combustible
        SJMP ACTDISP

ESFRENO: MOV R0,#57H   ; Dir 64H <- Pisadas de Freno

ACTDISP: ACALL SEGMEN
DECREM:  DJNZ R4,SIGDATO

        AJMP BIENVEN

;*** Rutina para borrar toda la informacion registrada.

BORRA:  MOV A,#00H     ; Escribe #00H de la Dir. 15 a la 1FH
        MOV R1,#11D    ; de la memoria de Dallas.
        MOV DPTR,#15H
REPITE: MOVX @DPTR,A
        INC DPTR
        DJNZ R1,REPITE

        MOV R0,#77H    ; Escribe #00H de la Dir 55 a la 77H
        MOV R1,#35D    ; de la memoria interna del uC
REBORRA: MOV @R0,#00H
        DEC R0
        DJNZ R1,REBORRA
```

```
        AJMP BIENVEN

;************** Rutina: Actualiza_de_Dallas_a_uC ***************
;      Rutina para actualizar la informacion de la memoria
;                       interna del uC.

ACTuC:  MOV R0,#5CH     ;Distancia recorrida:
        MOV R1,#04H     ; Origen: Dirs. 1C a 1FH, memoria
                        ;   Dallas
        MOV DPTR,#001CH ; Destino: Dirs. 5C a 5FH, memoria uC
        ACALL COPIAuC

        MOV R0,#55H     ;No. pisadas del freno:
        MOV R1,#03H     ; Origen: Dirs. 15 a 17H, memoria
                        ;   Dallas
        MOV DPTR,#15H   ; Destino: Dirs. 55 a 57H, memoria uC
        ACALL COPIAuC

        MOV R0,#6EH     ;Consumo de combustible:
        MOV R1,#02H     ; Origen: Dirs. 1A a 1BH, memoria
                        ;   Dallas
        MOV DPTR,#1AH   ; Destino: Dirs. 6E a 6FH, memoria uC
        ACALL COPIAuC

        RET

;**************** Rutina: Copia_de_Dallas_a_uC ****************
; Copia la informacion de la memoria de Dallas a la memoria RAM
;                       interna del uC.

COPIAuC: MOVX A,@DPTR  ; Reg 0 <- En donde se escribe la
                       ;   informacion
        MOV @R0,A      ; Reg 1 <- No. de bytes que se copian
        INC R0         ; DPTR  <- De donde se toma la informacion
        INC DPTR
        DJNZ R1,COPIAuC
        RET

;********************** Rutina: Aprieta_boton *****************
;             Espera a que el usuario oprima un boton.
;               Botones alambrados al Pto. 1 del uC
;          El boton seleccionado se guarda en la Dir. 3CH

APRIETA: ACALL DISPLAY  ; Despliega informacion 7 segmentos
        CLR 2           ; Display muestra RPM's
        JNB 0,VEBOTON   ; Si esta en Inicio del Sistema,
                        ;   brinca a VEBOTON
        ACALL UNSEG     ; Llama a Contador de 1 segundo
        JNB 1,VEBOTON   ; Si aun no transcurre 1 segundo,
                        ;   brinca a VEBOTON
        ACALL ACTUAL    ; Actualiza informacion

VEBOTON: MOV A,P1
        XRL A,#0FFH     ; Si no se ha apretado boton,
        JZ APRIETA      ;   brinca a APRIETA

        MOV 3CH,A       ; Dir 3CH <- Guarda el boton.
        ANL A,#04H      ; Transforma boton a BCD
        JZ NOES3
        MOV 3CH,#03H
NOES3:  MOV A,3CH
        ANL A,#08H
        JZ SUELTA
        MOV 3CH,#04H    ; Dir 3CH <- Guarda boton en BCD
        SETB 2          ; Si se oprimio el boton 4, se mostrara
                        ; el consumo de combustible

SUELTA:  ACALL DISPLAY  ;Espera a que se suelte el boton
        JNB 0,VEBOT     ; Si esta en Inicio del Sistema,
```

```
                        ;     brinca a VEBOT
        ACALL UNSEG     ; Llama a contador de 1 segundo
        JNB 1,VEBOT     ; Si aun no transcurre 1 segundo,
                        ;     brinca a VEBOT
        ACALL ACTUAL    ; Actualiza informacion

VEBOT:  MOV A,P1
        XRL A,#0FFH     ; Si no se ha soltado el boton
        JNZ SUELTA      ;     brinca a SUELTA
        RET


;****************** Rutina: Espera_Un_Segundo ******************
;               Espera a que transcurra 1 segundo

UNSEG:  ACALL ACCESO    ; Llama a rutina de acceso al reloj

                        ;Rutina lectura de fecha y hora
HEREO:  MOVX A,@DPTR    ; Se brinca 1/100 y 1/10 seg

        MOVX A,@DPTR
        MOV 3FH,A       ; Dir. 3FH <- Segundo en cuestion

        MOV R0,06H      ;Culmina los ciclos de acceso al reloj
HERE1:  MOVX A,@DPTR
        DJNZ R0,HERE1

        MOV A,3FH       ; Reg A <- Segundo en cuestion
        XRL A,3EH       ; Compara con el segundo registrado,
        JZ NOSEG        ; y si no ha cambiado:  Brinca a NOSEG

        MOV 3EH,3FH     ; Actualiza valor Dir. 3EH
        SETB 1          ; Indica que ya se cumplio un segundo
NOSEG:  RET


;****************** Rutina: Acceso_al_Reloj ******************
;           Rutina para tener acceso al reloj de Dallas

ACCESO: MOV DPTR,#0000H
        MOVX A,@DPTR

        MOV R0,#37H
ALLI:   MOV 2H,@R0
        MOVX A,@DPTR
        ANL A,R2
        MOVX @DPTR,A
        DEC R0
        MOV A,R0
        XRL A,#2FH
        JNZ ALLI
        RET


;****************** Rutina: Muestra_en_Display ******************
;    Rutina para mostrar informacion en displays de 7 segm.

DISPLAY: MOV R0,#38H    ; R0 <- Dir. informacion disp. 7 segm
         MOV R1,#01H    ; R1 <- Para habilitar display
         MOV R2,#04H    ; R2 <- Numero de numeros a desplegar.
DENUEVO: MOV DPTR,#PTO_A
         MOV A,@R0
         MOVX @DPTR,A   ; Displays conectados al Pto. A del PPI
         MOV DPTR,#PTO_C
         MOV A,R1
         MOVX @DPTR,A   ; Seleccion de display en Pto. C del PPI
```

```
        ACALL ESPERA    ; Espera un rato
        INC R0          ; R0 <- Apuntara al sig. digito
        MOV A,R1
        RL A
        MOV R1,A
        DJNZ R2,DENUEVO
        RET


;******************** Rutina: Retardo_de_Tiempo ****************
;  Se realizaron pruebas sobre el brillo del display con el
;  programa BRILREL.ASM, y  se encontro un brillo optimo
;  para los valores:
;         Reg 6 <- #0FFH
;         Reg 5 <- #03H
;  Sin embargo, despues de ver el funcionamiento de este programa
;  se llego a la conclusion de que el brillo del los displays
;  variara de acuerdo con la rutina que se este llevando a cabo,
;  por lo que el valor a asignar a R5 y R6 variara... es decir,
;  los valores de R5 y R6 se deberan asignar antes de entrar
;  a la rutina.

ESPERA:  MOV R6,#0FFH  ;Usa R5 y R6 para contar
AQUI7:   MOV R5,#03H   ; 1 ciclo
AQUI8:   NOP           ; 1 ciclo
         NOP           ; 1 ciclo
         NOP           ; 1 ciclo
         DJNZ R5,AQUI8 ; 2 ciclos
         DJNZ R6,AQUI7 ; 2 ciclos
         RET


;***************** Rutina: Actualiza_Informacion ***************
;        Guarda los valores registrados en los sensores
;                en sus correspondientes lugares

ACTUAL:  CLR 1          ; Limpia bit que indica transcurso 1 seg.

                ;*** Actualiza Distancia Recorrida

         MOV B,50H      ; Reg B <- Hoyos/mt
         MOV A,TL0      ; Reg A <- No. hoyos (TL0)
         DIV AB         ; Reg A <- Dist. recorrida en 1 seg, en
                        ;          Hex
         MOV 59H,A      ; 59H <- Guarda Distancia recorrida
         MOV TL0,B      ; TL0 <- Guarda vueltas no consideradas
                        ;        en el calculo anterior

                ;*** Actualiza lectura de combustible.

         MOV B,51H      ; Reg B <- Hoyos/lt
         MOV A,69H      ; Reg A <- No. hoyos leidos
         DIV AB         ; Reg A <- Consumo en 1 seg, en Hex
         MOV 6BH,A      ; Dir 6BH <- Guarda Consumo
         MOV 6AH,#00H   ; Dir 6AH <- Ceros
         MOV 69H,B      ; Dir 69H <- Guarda vueltas no
                        ; consideradas en el calculo anterior

                ;*** Actualiza lectura RPMs

         MOV B,52H      ; Reg B <- Hoyos/rpm
         MOV A,TL1      ; Reg A <- No. hoyos (TL1)
         DIV AB         ; Reg A <- RPMs en 1 seg, en Hex
         MOV 65H,A      ; Dir 65H <- Guarda RPMs
         MOV TL1,B      ; TL1 <- Guarda vueltas no consideradas
                        ;        en el calculo anterior

                ;*** Hace calculos, guarda totales
                  ;*** RPMs
```

```
        MOV A,65H
        MOV B,#3CH    ; Multiplica RPMs X 60
        MUL AB
        MOV 66H,B     ; Dirs 66 y 67H <- RPMs en Hex
        MOV 67H,A

        MOV R0,#66H   ; Reg 0 <- Dir fuente (Hex)
        MOV R1,#60H   ; Reg 1 <- Dir destino (BCD)
        ACALL HEXABCD ; Convierte RPMs de Hex a BCD


                      ;*** Consumo de Combustible
        MOV R0,#6AH   ; Reg 0 <- Dir. Fuente (Hex)
        MOV R1,#74H   ; Reg 1 <- Dir. Destino (BCD)
        ACALL HEXABCD ; Convierte Consumo de Hex a BCD

        MOV R0,#75H   ; Reg 0 <- Dato a Sumar (Contenido Dir
                      ;          75H)
        MOV R1,#6FH   ; Reg 1 <- Dir donde se guarda la Suma
        MOV R2,#02H   ; Reg 2 <- No. bytes a sumar
        ACALL SUMABCD ; Suma Consumo a la cuenta total

                      ;*** Distancia Recorrida
        MOV R0,#5BH   ; Reg 0 <- Dir. Fuente (Hex)
        MOV R1,#5AH   ; Reg 1 <- Dir. Destino (BCD)
        ACALL HEXABCD ; Convierte Distancia de Hex a BCD

        MOV 5BH,#00H  ; Borra Distancia Recorrida en Hex
        MOV 59H,#00H
        MOV R0,#5BH   ; Reg 0 <- Dato a Sumar (Dir 6BH)
        MOV R1,#5FH   ; Reg 1 <- Dir donde se guarda la Suma
        MOV R2,#04H   ; Reg 2 <- No. bytes a Sumar
        ACALL SUMABCD ; Suma Distancia recorrida al Total

;       MOV R1,#71H   ; Reg 1 <- Dir donde se guarda la Suma
;       ACALL SUMABCD ; Suma consumo a la cuenta de 1 minuto
;       MOV A,3EH       ; Si ya transcurrio 1 minuto, actualiza
;       XRL A,#00H      ; la Dir. que cuenta el consumo por
;                       ; minuto.
;       JNZ REG1S
;       MOV 72H,70H
;       MOV 73H,71H
;       MOV 70H,#00H
;       MOV 71H,#00H

;*** Actualiza la informacion en la memoria de Dallas.

                      ;Consumo de combustible:
        MOV R0,#6EH   ; Origen: Dirs. 6E a 6FH, memoria uC
        MOV R1,#02H   ; Destino: Dirs. 1A a 1BH, memoria Dallas
        MOV DPTR,#1AH
        ACALL ACTDAL
                      ;No. de pisadas del freno:
        MOV R0,#55H   ; Origen: Dirs 55 a 57H, memoria uC
        MOV R1,#03H   ; Destino: Dirs. 15 a 17H, memoria Dallas
        MOV DPTR,#15H
        ACALL ACTDAL
                      ;Distancia recorrida:
        MOV R0,#5CH   ; Origen: Dirs. 5C a 5FH, memoria uC
        MOV R1,#04H   ; Destino: Dirs. 1C a 1FH, memoria Dallas
        MOV DPTR,#1CH
        ACALL ACTDAL

;*** Actualiza informacion a mostrar en Displays

        MOV R0,#61H   ; Mostrara RPM's
        JNB 2,CALSEGH ; Si bit 2 esta encendido
        MOV R0,#75H   ;    Mostrara entonces consumo
                      ;      instantaneo
```

```
CALSEGM: ACALL SEGMEN    ; Actualiza displays 7 segmentos

         RET


;********** Rutina: Actualiza_memoria_de_Dallas **************
;  Copia la informacion de la memoria RAM interna del uC a la
;                      memoria de Dallas.

ACTDAL:  MOV A,@R0       ; Reg 0 <- De donde se toma la
;                        ;       informacion
         MOVX @DPTR,A    ; Reg 1 <- No. de bytes que se copian
         INC DPTR        ; DPTR  <- A donde se escribe la
;                        ;       informacion
         INC R0
         DJNZ R1,ACTDAL
         RET


;********** Rutina: Convierte_de_Hexadecimal_a_BCD *********
; Convierte informacion de Hexadecimal a BCD:
;     Reg 0 <- Contiene direccion fuente  (Hexadecimal)
;     Reg 1 <- Contiene direccion destino (BCD)
; Los valores de R0 y R1 se ponen antes de llamar a la rutina.

HEXABCD: MOV @R1,#0EFH ; Se guarda como informacion -> -1

D1000LP: INC @R1       ; Registra cuenta de millares
         MOV A,@R0     ;
         CLR C
         SUBB A,#04H   ; Resta 1024D a RPM's
         MOV @R0,A
         JC FIN1000    ; Si el dato se vuelve negativo, brinca a
         INC R0        ; FIN1000
         MOV A,@R0
         ADD A,#240    ; Suma 240 a informacion
         MOV @R0,A
         DEC R0
         JNC D1000LP   ; Si hay exceso al sumar 240,
         INC @R0       ; incrementa el valor del MSB
         SJMP D1000LP

FIN1000: MOV A,@R0     ; Suma 1024D a informacion
         ADD A,#04H    ;
         MOV @R0,A     ;
         DEC @R1
         MOV A,@R1     ; Pone en su lugar los millares (MSn)
         SWAP A
         ORL A,#0FH    ;
         MOV @R1,A     ;

         INC R0
D100LP:  INC @R1       ; Cuenta centenas en Dir 60H
         MOV A,@R0     ; Resta 1000 a RPM's
         CLR C
         SUBB A,#1000
         MOV @R0,A
         JNC D100LP    ; Si no hay overflow, brinca a D100LP
         DEC R0
         MOV A,@R0     ; Si lo hay: Resta 1 a Dir 74H
         CLR C
         SUBB A,#01H
         MOV @R0,A     ; Si el contenido de Dir 74H NO era cero
         INC R0        ;      brinca a D100LP
         JNC D100LP
         MOV A,@R0
```

```
        ADD A,#1000  ; Suma 1000 a informacion
        MOV @R0,A

        MOV A,@R1    ; Realiza ajuste decimal en Millares
        ADD A,#00    ; y Centenas
        DA A
        MOV @R1,A

        MOV A,@R0
        INC R1
        MOV @R1,#0FFH
D10LP:  INC @R1      ; Cuenta decenas en Dir 61H
        CLR C

        SUBB A,#100  ; Resta 100 a informacion
        JNC D10LP    ; Si el resultado es positivo, brinca a D10LP
        ADD A,#100   ; Suma 100 a informacion
        ANL A,#0FH   ; Reg A <- Guarda las unidades
        SWAP A
        ORL A,@R1    ; Junta unidades con decenas
        SWAP A       ; Los pone en su respectivo orden
        MOV @R1,A    ; @R1 <- Decenas y unidades

        RET


;************** Rutina: Suma_Dos_Numeros_en_BCD ****************
; Esta rutina suma el numero contenido en la Dir apuntada por R0
; al total guardado en la Dir apuntada por R1.
; Reg 0 <- Apunta al dato a sumar
; Reg 1 <- Apunta a la Dir. en la que va la suma.
; Reg 2 <- Contador del No. de bytes que faltan por sumar
; Reg 3 <- Ayuda para hacer la suma (Dir 03H)
; Reg 5 <- Dato a sumar
; Bandera 3 <- Lleva el "carry"
; Bandera 4 <- Ve si es nibble 0 o 1 (Der. o Izq.)

; Se debe dar un valor a Regs 0,1 y 2  antes de llamar a esta
;  Rutina.


SUMABCD: CLR 3       ; Limpia bandera "carry"
         CLR 4       ; Limpia bandera "nibble"

SUMBCD:  MOV A,@R0
         JNB 4,AQUI0 ; Ve si hay intercambio de nibbles
         SWAP A

AQUI0:   ANL A,#0FH  ; Obtiene nibble menos signif. dato a sumar
         MOV R3,A    ; Guarda nibble en Reg 3
         MOV A,@R1
         JNB 4,AQUI1 ; Ve si hay intercambio de nibbles
         SWAP A

AQUI1:   ANL A,#0FH  ; Obtiene nibble menos signif. de suma
                     ;   total
         ADD A,R3    ; Suma los nibbles
         DA A        ; Ajuste decimal
         JNB 3,AQUI2 ; Si no hay "carry" anterior brinca a AQUI2
         ADD A,#01H  ; Suma "carry"
         DA A        ; Ajuste decimal
         CLR 3       ; Limpia "carry"

AQUI2:   MOV R3,A    ; Reg 3 <- Suma parcial
         ANL A,#0F0H
         JZ AQUI3    ; Si hay carry,    Bit 3 <- 1
         SETB 3      ; Si NO hay carry, Bit 3 <- 0

AQUI3:   ANL 03H,#0FH ; Reg 3 <- Reg 3 .AND. #0FH
```

```
        MOV A,@R1     ; Reg A <- Dato antiguo (suma total)
        JNB 4,AQUI4   ; Ve si hace intercambio de nibbles
        SWAP A

AQUI4:  ANL A,#0F0H   ; Actualiza suma total
        ORL A,R3
        JNB 4,AQUI5   ; Ve si hace intercambio de nibbles
        SWAP A

AQUI5:  MOV @R1,A
        CPL 4         ; Complementa bandera de "nibble"
        JB 4,SUMBCD

        DEC R1
        DEC R0
        DJNZ R2,SUMBCD
        RET


;*************** Rutina: Conversion_a_7_segmentos ************
;         Rutina para actualizar el valor que aparecera
;                en los displays de 7 segmentos

;       Reg 0 <- 5FH, Para mostrar Distancia Recorrida
;       Reg 0 <- 61H, Para mostrar RPM's
;       Reg 0 <- 64H, Para mostrar pisadas de Freno
;       Reg 0 <- 6FH, Para mostrar Consumo Total de Combustible


SEGMEN: MOV R2,#02H   ; Reg 2 <- No. bytes a convertir
        MOV R3,#3BH   ; Reg 3 <- Lugar en que almacena
        CLR 3         ; Aqui bit 3 indica si hay SWAP o no
CONVER: MOV A,@R0     ; Reg 0 <- Dato que se mostrara en displays
        JNB 3,CONT
        SWAP A
CONT:   ANL A,#0FH
        ADD A,#40H
        MOV R1,A      ; Reg 1 <- Dir del valor en 7 segmentos
        MOV A,@R1
        MOV R5,A      ; Reg 5 <- Valor en 7 segmentos
        MOV A,R3      ; Reg 1 <- Dir en que almacenara el valor
        MOV R1,A
        MOV @R1,05H   ; @R1 <- Reg 5
        DEC R3
        CPL 3
        JB 3,CONVER
        DEC R0
        DJNZ R2,CONVER
        RET

        END;
```

```
;*****************************************
;*      Adquiridor de Datos para el Control de Vehiculos      *
;*                                                            *
;*                        Coras Garcia Ronald                 *
;*                        Sosa Lujano Gilberto Alejandro      *
;*                                                            *
;*****************************************

;---------------------- Zona de Equivalencias -----------------
VRESET   EQU 00H     ; Direccion del Vector de Reset
RRESET   EQU 33H     ; Direccion de la Rutina de Reset
VINT0    EQU 03H     ; Direccion del Vector de Interrupcion 0
VINT1    EQU 13H     ; Direccion del Vector de Interrupcion 1
PTO_A    EQU 2000H   ; Ubicacion del Pto. A del PPI
PTO_B    EQU 2001H   ; Ubicacion del Pto. B del PPI
PTO_C    EQU 2002H   ; Ubicacion del Pto. C del PPI
PTO_CTRL EQU 2003H   ; Ubicacion del Pto. de Control del PPI
KEY      EQU 55H     ; Ubicacion de la Clave del Sistema en RAM del uC
DIGCLAV  EQU 06H     ; Numero de digitos que comprende la clave
MAS      EQU 0AH     ; Tecla + de la Sumadora
TOTAL    EQU 0BH     ; Tecla =/T de la Sumadora
PAPEL    EQU 0CH     ; Tecla para dejar un renglon de papel
PUNTO    EQU 0DH     ; Tecla de "." en la Sumadora
ITINER   EQU 2FH     ; No. identificacion del Itinerario.
CHOFER   EQU 2EH     ; No. identificacion del Chofer.
HOYMT    EQU 05H     ; Equivalencia de Hoyos/mt (en Hex)

;---------------------- Declaracion de rutinas ----------------
;*** Rutina principal...

         ORG VRESET
         AJMP RRESET

;---------------------- Inicio de Programa --------------------
         ORG RRESET

;*** Prepara e. sistema para trabajar

                  ;*** Carga clave de acceso al reloj
         MOV 30H,#88H  ; La clave se guarda en las Dirs 30H a 37H
         MOV 31H,#66H ; de la memoria del uC.
         MOV 32H,#55H
         MOV 33H,#77H
         MOV 34H,#22H
         MOV 35H,#44H
         MOV 36H,#88H
         MOV 37H,#22H
                  ;*** Carga valores de 7 segmentos
         MOV 40H,#0C0H ; 0  Dir. 40H
         MOV 41H,#0F9H ; 1
         MOV 42H,#0A4H ; 2
         MOV 43H,#0B0H ; 3
         MOV 44H,#99H  ; 4
         MOV 45H,#92H  ; 5
         MOV 46H,#82H  ; 6
         MOV 47H,#0F8H ; 7
         MOV 48H,#80H  ; 8
         MOV 49H,#98H  ; 9
                  ;*** Definicion de puertos del PPI:
         MOV DPTR,#PTO_CTRL
         MOV A,#80H          ; Los Ptos. A, B y C se definen como
         MOVX @DPTR,A        ; puertos de salida.

;*** Actualiza la informacion de la memoria interna del uC, copiandola
;*** de la memoria de Dallas. La informacion a copiar es:
;***       - Distancia recorrida hasta el momento (parcial y total)
;***       - Clave del sistema
;***       - Velocidad maxima permisible
         ACALL ACTuC
```

```
;*** Asignacion de los valores por omision:
        CLR 1           ; Limpia Bit 1 (Indica ejecucion de MUENUM)
                        ;  "0" <- No ejecuta MUENUM
                        ;  "1" <- Si ejecuta MUENUM
        CLR 2           ; Limpia Bit 2 (Indica Corrimiento del Display)
                        ;  "0" <- No recorre Displays
                        ;  "1" <- Si recorre Displays
        CLR 9           ; Limpia Bit 9 (Indica escritura de nueva clave)
                        ;  "0" <- No se escribe nueva clave
                        ;  "1" <- Si se escribe nueva clave
        CLR 0AH         ; Limpia Bit A (Indica ejecucion de monitoreo)
                        ;  "0" <- No se esta ejecutando monitoreo
                        ;  "1" <- Si se esta ejecutando monitoreo
        CLR 0BH         ; Limpia Bit B (Indica las teclas esperadas)
                        ;  "0" <- Teclas 1 y 4
                        ;  "1" <- Teclas 2 y 3
        MOV R6,#01H     ; Reg 6 <- Retardo para display
        MOV R7,#0FFH    ; Reg 7 <- Tiempo despliegue letrero completo

;*** Muestra la pantalla inicial, esperando que el usuario seleccione
;*** la opcion deseada, o que ingrese la clave para entrar a la rutina
;*** de programacion.
CODIGO: MOV 66H,#00H    ; Dir 66H <- No. Digitos ingresados
        MOV 67H,#KEY    ; Dir 67H <- Lleva ubicacion clave
        CLR 0           ; Bit 0 <- Indica si hay SWAP o no.

                        ; Carga en Displays:
PIDE:   MOV 38H,#0C6H   ;  C
        MOV 39H,#0C0H   ;  O
        MOV 3AH,#0A1H   ;  D
        MOV 3BH,#86H    ;  E
        ACALL APRIETA   ; Dir 3CH <- Boton que se apreto

VECLAVE: MOV R0,67H     ; Reg 0 <- Dir. del digito a leer
        MOV A,@R0       ;
        JB 0H,NOHAZSW
        SWAP A
NOHAZSW: ANL A,#0FH     ; Reg A <- Boton que debio oprimirse
        XRL A,3CH
        JNZ NOCLAVE     ; Si el No. leido no corresponde a la Clave,
                        ;       brinca a NOCLAVE

;*** Si el numero corresponde a la clave:

SICLAVE: INC 66H        ; Dir 66H <- No. digitos metidos
        MOV 3CH,66H
        ACALL MUENUM
        MOV R7,#1FH
        ACALL DISPLAY
        CPL 0
        JB 0H,SIGUE
        INC 67H                 ; Si aun no se termina de ingresar
SIGUE:  MOV A,66H               ; la clave, brinca a PIDE
        CJNE A,#DIGCLAV,PIDE
        AJMP PROGRA             ; Si es fin de clave, brinca a PROGRA


;*** Si el numero no correspondio a la clave:

NOCLAVE: MOV A,67H      ; Si el boton 4 fue el PRIMER boton oprimido:
        XRL A,#KEY      ;   brinca a RUTA
        JNZ VIOLA       ;     Si no:
        JB 0H,VIOLA     ;       Si el boton 3 fue el PRIMER boton oprimido:
        MOV A,3CH       ;         brinca a AUTO
        CJNE A,#04H,VEAUTO ;        Si no:
        AJMP RUTA       ;             Brinca a NUECLAV
VEAUTO: MOV A,3CH
        CJNE A,#03H,NUECLAV
        AJMP AUTO
```

```
                ;*** Ve si se intenta cambiar de clave:
NUECLAV: JNB 9,VIOLA    ; Si intenta cambiar de clave,
         JNB 5,VIOLA    ;     brinca a CAMCLAV (Cambio de Clave)
         SETB 0BH
         MOV A,3CH      ; Si no:
         XRL A,#06H     ;     Brinca a VIOLA
         JNZ CODIGO
         CLR 5
         CLR 9
         CLR 0BH
         AJMP CAMCLAV

; ****************** Sub-Menu: Violaciones ********************
;       Violaciones son los intentos de acceso al sistema
VIOLA:   MOV DPTR,#0009H ; Las violaciones se escriben directamente
         MOVX A,@DPTR    ; en las Dirs. 8 y 9H de la memoria de Dallas.
         ADD A,#01H      ; Se guardan en BCD.
         DA A            ; No. violaciones <- * + 1
         MOVX @DPTR,A
         MOV 3CH,A       ; Dir 3CH <- Para poder mostrar No. Violaciones
         JNC MUEVIOL
         MOV DPTR,#0008H
         MOVX A,@DPTR
         ADD A,#01H
         DA A
         MOVX @DPTR,A
MUEVIOL: MOV 38H,#0E3H   ; V
         MOV 39H,#0EFH   ; I
         MOV 3AH,#0A3H   ; O
         MOV 3BH,#0CFH   ; L
         ACALL DISPLAY   ; Muestra letrero "VIOL"
         ACALL MUENUM    ; Muestra No. de Violaciones
         ACALL DISPLAY
         AJMP CODIGO

; ********************* Sub-Menu: Programacion ***********************
PROGRA:  MOV 38H,#8CH    ; P
         MOV 39H,#0AFH   ; R
         MOV 3AH,#0A3H   ; O
         MOV 3BH,#9DH    ; G
         ACALL DISPLAY   ; Muestra letrero "PROG"
         MOV 38H,#0A0H   ; a
         MOV 39H,#0FBH   ; i
         MOV 3AH,#0E3H   ; v
         MOV 3BH,#0A7H   ; c   ;Muestra "aivc"
         ACALL ESPERA    ; Espera a que se apriete un boton
         MOV A,3CH
         XRL A,#01H      ; Si oprimio boton 1, brinca a CODIGO
         JNZ VEBOT2      ;     (Sale del Menu PROG)
         AJMP CODIGO
VEBOT2:  MOV A,3CH
         XRL A,#02H      ; Si oprimio boton 2,brinca a IMPRES
         JNZ VEBOT3      ;     (Impresion de Resultados)
         AJMP IMPRES
VEBOT3:  MOV A,3CH       ; Si oprimio boton 3, brinca a CAMVEL
         XRL A,#03H      ;     (Cambio de Velocidad)
         JNZ VEBOT4
         AJMP CAMVEL
VEBOT4:  AJMP CAMCLAV    ; Si oprimio boton 4, brinca a CAMCLAV
                         ;     (Cambio de Clave)

;*** Boton 2 en PROGRA: Rutina para imprimir en la Sumadora
;***  TI-5029 la informacion almacenada en la memoria
;***  NVRAM de Dallas.
IMPRES:  MOV 38H,#0CFH   ; Muestra "Im P"
         MOV 39H,#0ABH
         MOV 3AH,#0AFH
         MOV 3BH,#8CH
         ACALL DISPLAY
```

```
        MOV 38H,#0A0H ; a
        MOV 39H,#0FBH ; i
        MOV 3AH,#0FFH
        MOV 3BH,#83H  ; b   ;Muestra "ai b"

REGIS:  ACALL APRIETA   ; Espera a que se apriete un boton
        MOV A,3CH       ; Dir 3CH <- Boton que se apreto
        XRL A,#01H
        JNZ VEBOTON2    ; Si oprimio boton 1,
        AJMP PROGRA     ;    Regresa a PROGRA
VEBOTON2:MOV A,3CH
        XRL A,#02H      ; Si oprimio boton 2,
        JNZ VEBOTON4    ;    Comienza la impresion
        AJMP VERIF
VEBOTON4:MOV A,3CH      ; Si oprimio boton 4,
        XRL A,#04H      ;    Limpia memoria NVRAM
        JNZ IMPRES
        AJMP LIMNVRM

;--> Boton 2 en IMPRES: Realiza impresión de la información...

                ;*** Verifica que la sumadora este conectada:
VERIF:  MOV A,P1
        XRL A,#0EFH     ; Si esta conectada:
        JZ SICONEC      ;   Brinca a SICONEC
        MOV 38H,#85H    ; De lo contrario:
        MOV 39H,#0AFH   ;   Muestra "Err "
        MOV 3AH,#0AFH
        MOV 3BH,#0FFH
        ACALL DISPLAY
        AJMP IMPRES

SICONEC: MOV DPTR,#PTO_B ; Deshabilita sumadora
        MOV A,#0FFH
        MOVX @DPTR,A

        MOV DPTR,#PTO_A ; Muestra en Displays: "PPPP"
        MOV A,#8CH      ; Indicando ejecucion de impresion.
        MOVX @DPTR,A
        MOV DPTR,#PTO_C ; Se mostrara "----" mientras se
        MOV A,#0FH      ; ejecuta la impresion.
        MOVX @DPTR,A
        MOV R6,#0FFH
        ACALL ESPERA
        MOV DPTR,#PTO_A
        MOV A,#08FH
        MOVX @DPTR,A

        CLR 8
        MOV R6,#01H     ; Retardo de tiempo

                ;*** Obtiene Fecha y Hora de Impresion del reporte
        SETB 6          ; Prende bandera de Hora
        SETB 7          ; Prende bandera de Fecha
        ACALL HORA      ; Lee Fecha y Hora, y los guarda en RAM uC
        MOV A,6FH       ; Acomoda la Hora en orden
        MOV 6FH,6EH     ;  Dir 6FH <- Hora
        MOV 6EH,A       ;  Dir 6EH <- Minutos

                ;*** Pone Clave del operador en RAM uC...
        MOV 64H,#CHOFER ; Dir 64H <- No. Identificacion Conductor
        MOV 65H,#ITINER ; Dir 65H <- No. de Itinerario

        MOV 58H,#00H    ; Dirs 58 a 5BH <- Llevara la suma del Total
        MOV 59H,#00H    ;    de violaciones de veloc. cometidas.
        MOV 5AH,#00H
        MOV 5BH,#00H

;*** Inicia impresion...
```

```
        MOV 63H,#00H    ; Dir 63H <- Tipo de Dato a Imprimir

                ;*** Imprime Clave del Operador:
        MOV R1,#65H     ; Reg 1 <- Apunta a Informacion a Imprimir
        MOV R2,#02H     ; Reg 2 <- No. de Bytes a Imprimir
        ACALL IMPRIME   ; Realiza la Impresion

        MOV 63H,6EH     ; Para Imprimir la Fecha, primero pasa los
                        ;  minutos a la Dir 63H

                ;*** Imprime Fecha:
        MOV 6EH,#00H    ; Dir 6EH <- Tipo de Informacion
        SETB 8          ; Primero escribira el tipo de Informacion
        MOV R1,#6EH     ; Reg 1 <- Apunta a Fecha de Impresion
        MOV R2,#04H     ; Reg 2 <- No. de Bytes a Imprimir
        ACALL IMPRIME   ; Realiza la Impresion

                ;*** Imprime Hora:
        MOV R1,#6FH     ; Reg 1 <- Apunta a la Hora de Impresion
        MOV R2,#01H     ; Reg 2 <- No. Bytes a Imprimir
        ACALL IMPRIME   ; Realiza la Impresion

;*** Imprimira la Informacion guardada en NVRAM...
        MOV 66H,#00H    ; Inicia DPTR en Dir 20H de la NVRAM
        MOV 67H,#20H
                ;*** Ve si Hay Informacion a Imprimir:
VERUTA: MOV DPH,66H     ; DPTR <- Direccion Inicial
        MOV DPL,67H
        MOVX A,@DPTR
        XRL A,#0FFH     ; Lee Byte, y si no es FFH
        JNZ FINREP      ;      Brinca a FINREP

        INC DPTR        ; Incrementa Apuntador
        MOV 66H,DPH     ; Guarda DPTR en RAM uC
        MOV 67H,DPL
        MOV R1,#6FH     ; Reg 1 <- De donde se Imprimira la Informacion
                ;*** Lee No. Ruta y lo Imprime:
        MOV R2,#01H     ; Reg 2 <- No. Bytes a Leer de la NVRAM
        ACALL LEENVRAM
        MOV R2,#01H     ; Reg 2 <- No. Bytes a Imprimir
        ACALL IMPRIME
                ;*** Lee Fecha de Inicio de Ruta, y la Imprime:
        MOV R2,#03H     ; Reg 2 <- No. Bytes a Leer de la NVRAM
        ACALL LEENVRAM
        SETB 8          ; Bit 8 <- Que Imprima primero el tipo
        MOV R1,#70H     ; Reg 1 <- Apunta a Dir a Imprimir
        MOV R2,#04H     ; Reg 2 <- No. Bytes a Imprimir
        MOV 70H,#00H
        ACALL IMPRIME
                ;*** Lee Hora de Inicio de Ruta, y la Imprime:
        MOV R1,#6FH
        MOV R2,#02H     ; Reg 2 <- No. Bytes a Leer de la NVRAM
        ACALL LEENVRAM
        MOV R2,#01H     ; Reg 2 <- No. Bytes a Imprimir
        MOV 63H,6EH     ; Dir 3H <- Guarda los minutos
        ACALL IMPRIME
                ;*** Imprime la Informacion Siguiente:
LEETIPO: MOV DPH,66H    ; Tipo de Informacion, un punto y la Informacion
        MOV DPL,67H     ;   correspondiente.
        MOVX A,@DPTR    ; Tipos:
        MOV 63H,A       ;   1 <- Viol. de Veloc (Distancia en que ocurrio)
        INC DPTR        ;   2 <- No. Viol. Veloc. en la Ruta
        MOV 66H,DPH     ;   3 <- Distancia Recorrida en la Ruta
        MOV 67H,DPL
        MOV R2,#02H     ; Reg 2 <- No. Bytes a Leer
        ACALL LEENVRAM  ; Lee la Informacion
        MOV R1,#6FH     ; Reg 1 <- De donde se Imprimira la Informacion
        MOV R2,#02H     ; Reg 2 <- No. Bytes a Imprimir
        ACALL IMPRIME   ; Imprime la Informacion
```

```
                ;*** Ve el Tipo de la Informacion
        MOV A,63H
        XRL A,#02H      ; Si el Dato es de Tipo 2 (No. Viol Veloc.):
        JNZ VETIPO3     ;  Copia Informacion a Dirs 61 y 62H
        MOV 62H,6EH     ;  Suma No. Violaciones al Total almacenado
        MOV 61H,6FH     ;       en Dirs 58 a 5BH.
        MOV R0,#62H     ;  Reg 0 <- Apunta al Dato a sumar
        MOV R1,#5BH     ;  Reg 1 <- Apunta a Dir. en donde se hace la suma.
        MOV R2,#03H     ;  Reg 2 <- No. bytes a sumar
        ACALL SUMABCD
        AJMP LEETIPO

VETIPO3: MOV A,63H      ; Si el Dato es de Tipo 3:
        XRL A,#03H      ;  Cambia el Tipo a 0.
        JNZ LEETIPO     ;  Lee Hora de Finalizacion de Ruta y la Imprime
        MOV R2,#02H     ;  Brinca a VERUTA.
        ACALL LEENVRAM
        MOV R2,#02H
        MOV 63H,6EH
        ACALL IMPRIME
        AJMP VERUTA

;*** Procede a imprimir los Datos Totales:
                ;*** Imprime la Distancia Total Recorrida
FINREP:  MOV 66H,#00H
        MOV 67H,#1CH
        MOV R2,#04H
        ACALL LEENVRAM
        MOV 63H,#11H    ; Dir 63H <- Tipo de Informacion
        MOV R2,#04H
        ACALL IMPRIME
                ;*** Imprime El No. Total de Viol. de Velocidad
        MOV 63H,#12H    ; Dir 63H <- Clave Total Viol. Veloc.
        MOV 6FH,58H     ; La Suma Total se pasa a las Dirs 6F a 6CH,
        MOV 6EH,59H     ;   que son las usadas para imprimir.
        MOV 6DH,5AH
        MOV 6CH,5BH
        MOV R2,#04H
        ACALL IMPRIME
                ;*** Imprime el No. Total de Viol. al Monitor
        MOV 63H,#13H    ; Dir 63H <- Clave Total Viol. al Monitor
        MOV 67H,#08H    ; El No. de Viol. al Monitor se encuentra en
        MOV R2,#02H     ;  las Dirs. 08 y 09H de la NVRAM.
        ACALL LEENVRAM
        MOV R2,#02H
        ACALL IMPRIME
                ;*** Imprime el No. Total de Puestas en Cero
        MOV 63H,#14H    ; Dir 63H <- Clave No. Total Puestas en Cero
        MOV 67H,#0EH    ; El No. Total de Puestas en cero se encuentra
        MOV R2,#02H     ;  en las Dirs. 0E y 0FH de la NVRAM.
        ACALL LEENVRAM
        MOV R2,#02H
        ACALL IMPRIME
                ;*** Imprime el No. de veces que ha cambiado la clave
        MOV 63H,#15H    ; Dir 63H <- Tipo de Informacion
        MOV 67H,#10H    ; Dirs 66 y 67H <- Valor para el DPTR al leer
        MOV R2,#01H     ; Reg 2 <- No. Bytes a leer
        ACALL LEENVRAM  ; Lee NVRAM Dallas
        MOV R2,#01H     ; Reg 2 <- No. Bytes a imprimir
        ACALL IMPRIME
        AJMP IMPRES

;--> Botón 4 en IMPRES: Limpia la NVRAM, para dar lugar a guardar
;    nueva información...
LIMNVRM: MOV 38H,#83H   ; Muestra "borr"
        MOV 39H,#0A3H
        MOV 3AH,#0AFH
        MOV 3BH,#0AFH
        ACALL DISPLAY
```

```
                ;*** Comienza el proceso de limpieza...
        MOV DPTR,#08H
        MOV A,#00H
        MOV R0,#06H     ; Reg 0 <- No. Bytes a Borrar
SIGBOR: MOVX @DPTR,A    ; Escribe 00H en Dirs. 08 a 0DH NVRAM
        INC DPTR        ; Borra: No. viol al monitor, No. viol. veloc,
        DJNZ R0,SIGBOR  ;        No. Ruta
        MOV DPTR,#10H   ; Escribe 00H en Dir. 10H
        MOVX @DPTR,A    ; Borra No. cambios de clave
        MOV DPTR,#18H
        MOV R0,#05H
SIGBOR2: MOVX @DPTR,A   ; Escribe 00H en Dirs. 18 a 1FH
        INC DPTR        ; Borra Dist. Parcial y Total recorridas.
        DJNZ R0,SIGBOR2
        MOV DPTR,#16H   ; Se inicializa la localidad del
        MOVX @DPTR,A    ; apuntador en la Dir. 20H de la NVRAM
        INC DPTR
        MOV A,#20H
        MOVX @DPTR,A
        MOV DPTR,#0FH   ; Actualiza el contador de no. de
        MOVX A,@DPTR    ;  puestas en cero de la NVRAM.
        ADD A,#01H
        DA A
        MOVX @DPTR,A
        JNC DEFIN1
        MOV DPTR,#0EH
        MOVX A,@DPTR
        ADD A,#01H
        DA A
        MOVX @DPTR,A
DEFIN1: MOV DPTR,#20H   ; Escribe CCH en Dirs. 20 a 2FH
        MOV A,#0CCH     ; Son señales para indicar memoria vacia.
        MOV R0,#10H
ESCRCC: MOVX @DPTR,A
        INC DPTR
        DJNZ R0,ESCRCC
        AJMP INPRES

;*** Boton 3 en PROG: Rutina para cambiar la Velocidad
;***  Máxima Permitida.
CAMVEL: MOV 51H,#00H  ; Limpia velocidad (Dirs 51H, 52H y 53H)
        MOV 52H,#00H
        MOV 53H,#00H
        MOV 39H,#0E3H ; Muestra en Display: "v 00"
        MOV 3FH,#0FFH
        MOV 31H,#0C0H
        MOV 32H,#0C0H

CONTCAM: ACALL APRIETA ; Dir 3CH <- Guarda boton apretado
        DEC 3CH
        MOV A,3CH     ; Si oprimio boton 1:
        XRL A,#00H    ;  brinca a FINCVEL
        JZ FINCVEL
        MOV A,3CH     ; Si oprimio boton 3 o 4:
        XRL A,#01H    ;   brinca a ES304
        JNZ ES304
                      ; Si oprimio boton 2:
        MOV 39H,#0FFH ;    Display 2 <- 1
        MOV A,51H     ;    Actualiza valor
        INC A
        ANL A,#01H    ;    Blanco -> 1 -> Blanco
        MOV 51H,A
        JZ ESCERO     ;    Actualiza display 7 segmentos
        MOV 39H,#0F9H ;    Display 2 <- Blanco
ESCERO: AJMP CONTCAM

ES304:  MOV A,3CH     ; Los digitos de la veloc. maxima se guardan
        ADD A,#50H    ;   en las Dirs. 51H, 52H y 53H
        MOV R0,A      ; Reg 0 <- Dir. en que se incrementa el digito
```

```
        MOV A,@R0
        ADD A,#01H      ; Incrementa el digito
        DA A
        ANL A,#0FH
        MOV @R0,A       ; Guarda digito en lugar correspondiente
        ADD A,#40H      ; Ubica la representacion del digito en 7 segm.
        MOV R1,A        ; Reg 1 <- Dir del codigo en 7 segm.
        MOV A,R0        ; Ubica lugar del display correspondiente
        CLR C
        SUBB A,#18H
        MOV R0,A        ; Reg 0 <- Display correspondiente
        MOV A,@R1       ; Actualiza display correspondiente,
        MOV @R0,A       ;      colocandole el nuevo codigo de 7 segm.
        AJMP CONTCAM    ; Brinca a solicitar un cambio en la velocidad.

FINCVEL: MOV A,52H      ; Junta el contenido de las Dirs 52H y 53H
        SWAP A          ;       y lo guarda en la Dir. 53H
        ORL 53H,A
        MOV A,51H
        ANL A,#01H
        JNZ BIEN
        MOV A,53H
        CLR C
        SUBB A,#60H
        JNC BIEN        ; Si la veloc. es menor a 60 km/hr
        MOV 53H,#60H    ;   se hace que Veloc sea de 60 km/hr
BIEN:   ACALL BCDBIN    ; Convierte veloc. de BCD a Binario
        ACALL MSHEX     ; Convierte veloc. de km/hr a m/s
        AJMP PROGRA     ; Brinca a PROGRA


;*** Boton 4 en PROGRA: Rutina de Cambio de Clave.
CAMCLAV: MOV 38H,#0C6H ; Pone "CC  "  (Cambio de Clave)
        MOV 39H,#0C6H
        MOV 3AH,#0FFH
        MOV 3BH,#0FFH
        MOV 66H,#00H    ; Dir 66H <- Posicion digito clave
        MOV 67H,#KEY    ; Dir 67H <- ubica 1er. digito de clave
        CLR 0           ; Bit 0 <- Indica si hay SWAP o no.

INGCLAV: SETB 1         ; Prende bits 1 y 2
        SETB 2
        ACALL APRIETA   ; Lee boton
        MOV A,66H
        XRL A,#00H      ; Si no es primer digito
        JNZ CONTIN      ;    brinca a CONTIN
        MOV A,3CH
        CLR C
        SUBB A,#03H     ; Si es primer digito, no puede ser 3 o 4
        JC CONTIN       ;    Si es 3 o 4:
        MOV 39H,#0FFH   ;       Escribe "C Er"
        MOV 3AH,#86H    ;
        MOV 3BH,#0AFH   ;
        ACALL DISPLAY   ; Despliega mensaje
        AJMP CAMCLAV    ; Brinca a CAMCLAV

CONTIN:  MOV R0,67H     ; Reg 0 <- Direc. Clave en RAM uC
        MOV A,@R0       ; Reg A <- Viejo Digito
        JB 0,NOSWAP
        SWAP A
NOSWAP:  ANL A,#0F0H;
        ORL A,3CH       ; Reg A <- Contiene nuevo digito
        JB 0,NOSWAP2
        SWAP A
NOSWAP2: MOV @R0,A      ; Se guarda el nuevo digito
        INC 66H
        CPL 0
        JB 0,NOINCREM
        INC 67H
```

```
NOINCREM:MOV A,66H      ; Dir 66H <- Lleva cuenta de digitos metidos
         XRL A,#DIGCLAV ; DIGCLAV dice cuantos digitos son de clave.
         JNZ INGCLAV

                ;*** Copia clave de RAM uC a RAM Dallas.
         MOV DPTR,#04H ; DPTR <- Dir. Clave en RAM Dallas
         MOV R0,#54H   ; Reg 0 <- Dir. Clave en RAM uC
         MOV R1,#04H   ; Reg 1 <- No. bytes a copiar
COP:     MOV A,@R0
         MOVX @DPTR,A
         INC R0
         INC DPTR
         DJNZ R1,COP

               ;*** Actualiza el no. de veces que se ha cambiado la clave
         MOV DPTR,#10H
         MOVX A,@DPTR  ; Este dato se almacena en la Dir. 10H de la NVRAM
         ADD A,#01H
         DA A
         MOVX @DPTR,A

            ;*** Una vez terminada la actualizacion:
         MOV 30H,#0C7H ; L
         MOV 39H,#0EFH ; i
         MOV 3AH,#92H  ; s
         MOV 3BH,#87H  ; t
         ACALL DISPLAY  ; Muestra "List"
         AJMP CODIGO

; ********************* Sub-Menu de Ruta ************************
; Rutina para realizar el conteo de los parametros a medir.
RUTA:    MOV 38H,#0AFH ; r
         MOV 39H,#0E3H ; u
         MOV 3AH,#87H  ; t
         MOV 3BH,#88H  ; A
         ACALL DISPLAY ; Muestra letrero "RUTA"
                       ; Programacion de Timer/Counters
         SETB 0AH      ; Se prende el indicador de ejecucion de monitoreo

;         MOV 61H,#00H    ; Limpia cuenta de 26 hoyos
         MOV 3EH,#00H    ; Dir 3EH <- Para verificar cambio de seg

                       ;*** Copia Veloc. Maxima de RAM Dallas a RAM uC:
         MOV DPTR,#0002H ; La veloc. se trabaja en m/s
         MOVX A,@DPTR    ; Origen  <- Dirs. 02 y 03H de la RAM de Dallas
         MOV 52H,A       ; Destino <- Dirs. 52 y 53H de la RAM uC
         INC DPTR
         MOVX A,@DPTR    ; La veloc. esta en hexadecimal.
         MOV 53H,A

                       ;*** Limpia y prende contadores.
         MOV 60H,#05H    ; Hoyos/mt : 05H (Dir 60H) (Hoyos expres. en Hex)
         MOV TL0,#00H    ; Limpia regs. de los contadores.
         MOV TL1,#00H
         MOV TH0,#00H
         MOV TH1,#00H
         MOV IE,#00H     ; Deshabilita interrupciones.
         MOV TMOD,#55H   ; TMOD <- 55H Program. T/C como Counters
         MOV TCON,#15H   ; TCON <- 15H (Prende contador 0)

;*** Inicia medicion de velocidad y...
                ;*** Limpia No. Viol. Velocidad y Dist. Par. Rec.
LIMPIA:  MOV DPTR,#0AH   ; Limpia No. Viol. Velocidad
         MOV A,#00H
         MOVX @DPTR,A
         INC DPTR
         MOVX @DPTR,A
         MOV DPTR,#18H   ; Limpia Distancia Parcial Recorrida
         MOV R0,#04H
```

```
LIMPDPR: MOVX @DPTR,A
         INC DPTR
         DJNZ R0,LIMPDPR

                  ;*** Obtiene el No. de Ruta
         MOV DPTR,#0DH   ; Reg A <- Dir 0DH RAM Dallas (No. Ruta)
         MOVX A,@DPTR
         ADD A,#01H      ; No. Ruta <- No. Ruta + 1
         DA A
         MOV 63H,A       ; Dir 63H <- No. Ruta Actual
         MOVX @DPTR,A    ; Actualiza No. Ruta.

                  ;*** Obtiene Fecha y Hora de Inicio de Ruta.
         SETB 6          ; Prende bandera de Hora
         SETB 7          ; Prende bandera de Fecha
         ACALL HORA      ; Lee Fecha y Hora, y los guarda en RAM uC

                  ;*** Escribe inicio y No. de Ruta
         ACALL APa@DPTR  ; DPTR <- Apuntador RAM Dallas
         MOV A,#0FFH     ; Escribe FFH en la posicion correspondiente
         MOVX @DPTR,A    ;    (indica inicio de una nueva Ruta)
         INC DPTR        ; Incremento Apuntador.
         MOV A,63H       ; Reg A <- No. Ruta
         MOVX @DPTR,A    ; Escribe No. Ruta en RAM Dallas
         INC DPTR
                  ;*** Escribe Fecha y Hora en RAM Dallas
         MOV R1,#03H     ; Escribe Fecha en RAM Dallas
         MOV R0,#60H
COPFECHA:MOV A,@R0
         MOVX @DPTR,A
         INC DPTR
         DEC R0
         DJNZ R1,COPFECHA

         MOV A,6EH       ; Escribe Hora en RAM Dallas
         MOVX @DPTR,A
         INC DPTR
         MOV A,6FH
         MOVX @DPTR,A

         ACALL DPTRaAP   ; Actualiza Apuntador RAM Dallas.

                  ;*** Copia Distancias Parcial y Total Recorridas
                  ;*** de la RAM Dallas a RAM uC.
         MOV R0,#58H
         MOV DPTR,#18H   ; Origen: Dirs 18 a 1FH RAM Dallas.
         MOV R1,#08H     ; Destino: Dirs 58 a 5FH RAM uC.
COPIA:   MOVX A,@DPTR
         MOV @R0,A
         INC R0
         INC DPTR
         DJNZ R1,COPIA
                  ;*** Limpia Banderas...
         CLR 4           ; Limpia Bandera 4. Indica si esta en ejecucion
                         ; una violacion de velocidad (1) o si no (0).
         CLR 5           ; Limpia Bandera 5. Indica si ya transcurrio
                         ; un segundo (1) o no (0).

;*** Ciclo de Actualizacion de viol. veloc. y Distancia Recorrida
REGIS:   ACALL APRIETA   ; Espera a que se apriete un boton
                         ; Dir 3CH <- Boton que se apreto
         MOV A,3CH       ; Si oprimio boton 1,
         XRL A,#01H      ;    Escribe informacion de Fin de Ruta
         JNZ VE4         ;    Brinca a LIMPIA (Inicio de Ruta).
         ACALL FINRUTA
         AJMP LIMPIA
VE4:     MOV A,3CH       ; Si oprimio botones 2 o 3
         XRL A,#04H      ;    Brinca a REGIS (continua Monitoreo)
         JNZ REGIS
```

```
        CLR 0           ; Si oprimio boton 4
        MOV IE,#00H     ;    Apaga los Contadores
        MOV TCON,#00H   ;    Escribe informacion de Fin de Ruta
        ACALL FINRUTA   ;    Escribe en RAM Dallas marca de Fin de
        ACALL APaDPTR   ;       Monitoreo (CCH), sin incrementar DPTR
        MOV A,#0CCH     ;    Termina el monitoreo.
        MOVX @DPTR,A

        CLR OAH

        AJMP CODIGO


;*************** Rutina: Actualiza_de_Dallas_a_uC ***************
;       Rutina para actualizar la informacion de la memoria
;                       interna del uC
ACTuC:  MOV R0,#58H     ;Distancias recorridas (Parcial y Total):
        MOV R1,#08H     ; Origen: Dirs. 18 a 1FH, memoria Dallas
        MOV DPTR,#0018H ; Destino: Dirs. 58 a 5FH, memoria uC
        ACALL COPIAuC
        MOV R0,#52H     ;Velocidad Maxima Permisible y Clave del Sist:
        MOV R1,#06H     ; Origen: Dirs. 02 a 03H, memoria Dallas
        MOV DPTR,#02H   ; Destino: Dirs. 52 a 53H, memoria uC
        ACALL COPIAuC   ; Origen: Dirs. 04 a 07H, memoria Dallas
                        ; Destino: Dirs. 54 a 57H, memoria uC
        RET

;***************** Rutina: Copia_de_Dallas_a_uC ****************
; Copia la informacion de la memoria de Dallas a la memoria RAM
;                       interna del uC.
COPIAuC: MOVX A,@DPTR   ; Reg 0 <- En donde se escribe la informacion
        MOV @R0,A       ; Reg 1 <- No. de bytes que se copian
        INC R0          ; DPTR  <- De donde se toma la informacion
        INC DPTR
        DJNZ R1,COPIAuC
        RET

;******************** Rutina: Muestra_Numero ********************
; Con el bit 2 indicamos si la informacion del display se recorre
; de derecha a izquierda. El bit 2 debe entrar a esta rutina con
; un valor determinado; el valor por Default es 0 (no se recorre).
MUENUM:  JB 2,CORRE
        MOV 38H,#0FFH ; Limpia displays 1, 2 y 3
        MOV 39H,#0FFH
        MOV 3AH,#0FFH
        AJMP BUSNUM
CORRE:  MOV 39H,3AH   ; Recorre del Display 3 al 2
        MOV 3AH,3BH   ; Recorre del Display 4 al 3
BUSNUM: MOV A,3CH     ; Reg A <- Numero a desplegar (Dir 3CH)
        ANL A,#0FH
        ADD A,#40H
        MOV R0,A
        MOV 3BH,@R0   ; Escribe dato en Display 4
        CLR Z
        RET

;******************** Rutina: Aprieta_boton *********************
; Espera a que el usuario oprima un boton. Los botones estan
; alambrados a: Pto. 1 del uC
; El boton seleccionado se guarda en la Dir. 3CH.
; El bit 1 indica si se ejecuta MUENUM (1) o no (0). El bit 1
; debe entrar a esta rutina con un valor; el valor por Default
; es 0.

APRIETA: MOV R7,#01H    ; Despliegue Rapido
        ACALL DISPLAY   ; Despliega informacion 7 segmentos
        JNB OAH,VEBOTON ; Si no esta en proceso el monitoreo,
                        ;       brinca a VEBOTON
        ACALL HORA      ; Llama a contador de 1 segundo
```

```
        JNB 5,VEBOTON   ; Si aun no transcurre 1 segundo,
                        ;    brinca a VEBOTON
        ACALL ACTUAVEL  ; Actualiza informacion
VEBOTON: MOV A,P1
        ORL A,#10H      ; Anula la influencia de la sumadora
        XRL A,#0FFH     ; Si no se ha apretado boton,
        JZ APRIETA      ;    brinca a APRIETA
        MOV 3CH,A       ; Dir 3CH <- Guarda el boton.
        XRL A,#04H      ; Transforma boton a BCD
        JNZ NOES3
        MOV 3CH,#03H
NOES3:  MOV A,3CH
        XRL A,#08H
        JNZ PREPCC
        MOV 3CH,#04H    ; Dir 3CH <- Guarda boton en BCD
                        ;*** Ve si se quiere cambiar la clave del sistema
                        ;***  desde la pantalla inicial
PREPCC: MOV 3DH,#07H
        CLR 9
MUESTRA: JNB 1,SUELTA
        ACALL HUENUM    ; Pasa No. al Display
                        ;*** Espera a que se suelte el boton
SUELTA: MOV R7,#01H     ; Despliegue rapido
        ACALL DISPLAY
        JB 0AH,LLAMHORA ; Si estan prendidos Bit 9 o Bit AH
        JB 9H,LLAMHORA  ;   Brinca a LLAMHORA
        AJMP VESICC
LLAMHORA:ACALL HORA     ; Llama a contador de 1 segundo
        JNB 9,VEACTUA   ; Si Bit 9 prendido y ya transcurrio 1 seg
        JNB 5,VEACTUA   ;    Ejecuta LIMDIS
        ACALL LIMDIS
VEACTUA: JNB 5,VEBOT    ; Si aun no transcurre 1 segundo,
                        ;    Brinca a VEBOT
        ACALL ACTUAVEL  ; Actualiza informacion
        AJMP VEBOT

                ;*** Esta parte es util solo al cambiar la clave...
VESICC:  JB 0BH,VE2Y3   ; Verifica si se desea cambiar clave.
        MOV A,P1        ;  Bit 8 = 0 <- Botones 1 y 4
        ORL A,#10H
        XRL A,#0F6H     ;  Bit 8 = 1 <- Botones 2 y 3
        JNZ VEBOT
        MOV 3CH,#09H
        AJMP SIHACC     ; Al oprimir los botones adecuados,
VE2Y3:  MOV A,P1        ;   Se enciende Bit 9.
        ORL A,#10H
        XRL A,#0F9H
        JNZ VEBOT
        MOV 3CH,#06H
SIHACC: SETB 9

VEBOT:  MOV A,P1
        ORL A,#10H      ; Anula la influencia de la sumadora.
        XRL A,#0FFH     ; Si no se ha soltado el boton
        JNZ SUELTA      ;    brinca a SUELTA

        CLR 1           ; Bit 1 <- Valor Default = 0
        RET

;************** Rutina: Ve_si_se_cambia_la_clave ***************
; Se escribio una alternativa para realizar la reescritura de la
; clave del sistema, por si el usuario la olvido. El procedimiento
; es oprimir las teclas 1 y 4 durante 5 seg, y al ver que se borre
; el display, debera oprimir las teclas 2 y 3, tambien durante
; 5 seg. Entonces el sistema solicitara que el usuario ingrese la
; nueva clave de acceso.

;******************** Rutina: Blanquea_Display ******************
;       Rutina para apagar los Displays de 7 segmentos.
```

```
LIMDIS:  DEC 30H          ; Dir 30H <- Lleva la cuenta de 5 seg
         MOV A,30H
         XRL A,#00H
         JZ LDISP         ; Si aun no transcurren:
         CLR 5            ;   Limpia Bit 5
         RET
LDISP:   INC 30H          ; Si ya transcurrieron:
         MOV 38H,#0FFH    ;   Limpia los Displays
         MOV 39H,#0FFH
         MOV 3AH,#0FFH
         MOV 3BH,#0FFH
         RET


;********************* Rutina: Lee_la_Hora *********************
; Rutina para leer la informacion del reloj de Dallas, para:
;       1) Verificar si ha transcurrido 1 seg.
;       2) Leer Hora del Reloj.   (Bit 6 Prendido)
;       3) Leer Fecha del Reloj.  (Bit 7 Prendido)

HORA:    ACALL ACCESO     ; Llama a rutina de acceso al reloj
         MOV R0,#01H      ; Se brinca 1/100 y 1/10 seg
         JNB 5,HERE0      ; Si esta prendido Bit 6:
         MOV R0,#02H      ;    Se brinca tambien los segundos
HERE0:   MOVX A,@DPTR     ; Reg 0 cuenta # bits a brincarse
         DJNZ R0,HERE0

                  ;*** Ve la informacion a leer:
         MOV R3,#01H      ; Prepara Regs 0 y 3 para verificar
         MOV R0,#3FH      ;   el transcurso de 1 seg.
         JNB 6,HERE1      ; Si Bit 6 esta prendido:
         MOV R3,#02H      ;   Prepara Regs 0 y 3 para leer la Hora
         MOV R0,#6FH
                          ; Dir 3FH <- Segundo en cuestion
HERE1:   ACALL LEEREL     ; Dirs 6E a 6FH <- Hora
         MOV R0,#06H      ; Reg 0 <- No. bits a leer, sin registrarlos.
         JNB 7,HERE3
         MOV R0,#03H
HERE3:   JNB 6,HERE4
         MOV R0,#04H
         JNB 7,HERE4
         MOV R0,#01H
HERE4:   MOVX A,@DPTR     ; Si no lee Fecha:
         DJNZ R0,HERE4    ;    Culmina los ov ciclos de acceso al reloj
                          ;    Brinca a COMPSEG
         JNB 7,COMPSEG
         MOV R0,#6DH      ; Reg 0 <- Apunta a Dir. de almacenamiento
         MOV R3,#03H      ; Reg 3 <- No. Bytes a leer
                          ; Dirs. 6B a 6DH <- Almacenan la Fecha
         ACALL LEEREL     ; Lee Fecha
COMPSEG: MOV A,3FH        ; Reg A <- Segundo en cuestion
         XRL A,3EH        ; Compara con el segundo registrado,
         JZ FINHORA       ; y si no ha cambiado:  Brinca a FINHORA
         MOV 3EH,3FH      ; Actualiza valor Dir. 3EH
         SETB 5           ; Indica que ya se cumplio un segundo
FINHORA: CLR 6
         CLR 7            ; Limpia Banderas 6 y 7
         RET

;******************** Rutina: Acceso_al_Reloj ******************
;       Rutina para tener acceso al reloj de Dallas
ACCESO:  MOV DPTR,#0000H ; Inicia apuntador reloj Dallas
         MOVX A,@DPTR
         MOV R0,#37H
ALLI1:   MOV 2H,@R0
         MOVX A,@DPTR
         ANL A,R2
         MOVX @DPTR,A
         DEC R0
```

```
        MOV A,R0
        XRL A,#2FH
        JNZ ALLI1
        RET

;*************** Rutina: Lee_informacion_del_Reloj ************
; Rutina para leer informacion del Reloj de Dallas y guardarla
; en el lugar correspondiente.
;    Reg 0 <- Apunta al lugar en donde se almacena
;    Reg 3 <- No. de Bytes a leer
; Los valores de Reg 0 y Reg 3 se dan antes de llamar a la Rutina.
LEEREL: MOVX A,@DPTR   ; Reg 0 <- Apunta al lugar en el que se
        MOV R2,A       ;          almacena el dato.
        MOV @R0,A      ; Reg 3 <- No. de bytes a leer
        DEC R0
        DJNZ R3,LEEREL
        RET

;****************** Rutina: Muestra_en_Display ***************
; Rutina para mostrar informacion en displays de 7 segmentos.
; El Reg 7 guarda el retardo de tiempo que un letrero permanecera
; mostrado en los displays.
; El Reg 7 debe contener un valor antes de entrar a esta rutina,
; de lo contrario, entra con el valor Default -> OFFH (Tiempo largo).
; Se recomiendan los siguientes valores para Reg 7:
;       01H <- Si la informacion se debe mostrar rapido
;       OFFH <- Si se desea mostrar un letrero durante aprox 1.5 seg.

DISPLAY: MOV R0,#38H     ; Reg 0 <- Direccion Informacion del Display
        MOV R2,#01H      ; Reg 2 <- Habilitacion del Display
        MOV R4,#04H      ; Reg 4 <- Contador de Numeros a desplegar
DENUEVO: MOV DPTR,#PTO_A
        MOV A,@R0
        MOVX @DPTR,A     ; Displays conectados al Pto. A del PPI
        MOV DPTR,#PTO_C ;    (Muestra informacion)
        MOV A,R2
        MOVX @DPTR,A     ; Seleccion de display en Pto. C del PPI
        ACALL ESPERA     ;    (Habilita display)
        MOV A,#00H       ; Espera un rato
        MOVX @DPTR,A     ; Deshabilita display
        INC R0
        MOV A,R2
        RL A
        MOV R2,A
        DJNZ R4,DENUEVO
        DJNZ R7,DISPLAY ; Reg 7 <- Veces que repite la rutina
        MOV R7,#OFFH     ; Reg 7 <- Valor Default = OFFH
        RET

;****************** Rutina: Retardo_de_Tiempo ****************
; Esta rutina utiliza los Regs. 5 y 6 para contar.
; Se debe especificar un numero en Reg 6, antes de entrar a esta
; rutina. Los valores recomendados son los siguientes:
;     01H <- Mostrar numeros en display   (Valor Default)
;     OFFH <- Enviar numeros a la sumadora
ESPERA:  MOV R5,#OFFH    ; Reg 5 <- Valor Default = OFFH (siempre)
TIMEO:   NOP             ; 1 ciclo
         NOP             ; 1 ciclo
         NOP             ; 1 ciclo
         DJNZ R5,TIMEO   ; 2 ciclos
         DJNZ R6,ESPERA ; 2 ciclos
         MOV R6,#01H     ; Reg 6 <- Valor Default = 01H
         RET

;************* Rutina: Convierte_de_BCD_a_Binario *************
; Convierte a binario el numero en BCD ubicado en la Dir. 53H.
;       El numero convertido se guarda en la Dir. 53H
BCDBIN:  MOV A,53H
         SWAP A
```

```
        ANL A,#0FH
        MOV R3,A
        ANL A,#0FH  ; Si el primer digito de la Dir 53H es cero,
        JZ VECIEN   ;     brinca a VECIEN
        MOV A,53H
        CLR C
RESTA:  SUBB A,#06H
        DJNZ R3,RESTA
        MOV 53H,A   ; Guarda en Dir 53H el resultado parcial
VECIEN: MOV A,51H   ;     de la velocidad
        ANL A,#01H
        JZ FINAL
        MOV A,53H   ; Si hay centenas involucradas (Dir 51H = #01H),
        ADD A,#64H  ;   suma 64H al resultado parcial.
        MOV 53H,A
FINAL:  RET

;********** Rutina: Convierte_veloc_de_km/hr_a_m/s ************
; La Dir. 53H contiene la velocidad, en Km/hr; toma dicho valor
; y lo convierte a m/seg, sumando de 47 en 47, en hexadecimal,
; y se va guardando en el Reg A (parte fraccionaria) y en la
; Dir 52H (parte entera).
; El resultado se guarda en las Dirs. 02H y 03H de la RAM de DALLAS
; y en las Dirs. 52 y 53H de la RAM del uC.
; La velocidad en m/seg se obtiene en HEXADECIMAL.

MSHEX:  MOV 52H,#00H   ; En Dirs. 02H y 03H de la RAM de Dallas
        MOV R3,53H     ; se guarda la velocidad, en hexadecimal, en m/seg
        MOV A,#00H     ;   Dir 02H <- parte entera
SIGSUM: ADD A,#47H     ;   Dir 03H <- parte fraccionaria
        JNC NOCARRY
        INC 52H        ; Suma 47H
NOCARRY: DJNZ R3,SIGSUM
        MOV DPTR,#0003H
        MOVX @DPTR,A   ; Almacena parte fraccionaria en RAM Dallas
        MOV 53H,A      ; Alamcena parte fraccionaria en RAM uC
        MOV A,52H
        MOV DPTR,#0002H
        MOVX @DPTR,A   ; Almacena parte entera en RAM Dallas
        RET

;****************** Rutina: Imprime_Informacion ***************
; Rutina que escribe la informacion en la sumadora, y la imprime.
; Si el Bit 8 entra apagado, se escribe primero la informacion ,
; despues un punto y el tipo de dato. Si entra prendido, se
; imprimira al reves.
; Se usan los Regs 1 y 2, y la Dir 63H, que deben tener un valor
; antes de ejecutar esta rutina.
;       Reg 1 <- Apunta a la informacion que se imprimira
;       Reg 2 <- Contiene el No. de Bytes a imprimir
;       Dir 63H <- Contiene el Tipo de Informacion que se imprime.

IMPRIME: MOV DPTR,#PTO_B ; Impresora conectada al Pto B del PPI
        MOV R0,01H       ; Reg 0 <- Reg 1
        CLR 0            ; Limpia Bandera de Swap
        JB 8,IMPRE0      ; Si Bit 8 = 0
        SETB 8           ;   Primero escribe la informacion
        SJMP IMPRE1
                         ; Si no
IMPRE0: CLR 8            ;   Escribe el tipo de informacion y luego
                         ;   la informacion, en el mismo ciclo.
IMPRE1: MOV A,@R0
        JB 0,IMPRE2
        SWAP A
IMPRE2: ANL A,#0FH
        MOVX @DPTR,A     ; Escribe num. en calculadora
        MOV R6,#0FFH
        ACALL ESPERA     ; Espera un momento
        MOV A,#0FFH      ; Deshabilita calculadora
```

```
        MOVX @DPTR,A
        MOV R6,#0FFH
        ACALL ESPERA    ; Espera un momento
        CPL 0
        JB 0,IMPRE1
        DEC R0
        DJNZ R2,IMPRE1
        JNB 8,IMPRE4

IMPRE3: CLR 8           ; Ahora se escribira el tipo de informacion
        MOV R0,#64H     ; Reg 0 <- Ahora apunta al tipo de inform.
        MOV R2,#02H     ;
        MOV A,#PUNTO
        CPL 0           ; Manda escribir el punto decimal
        AJMP IMPRE2
                ;*** Imprime la informacion escrita ...
IMPRE4: MOV A,#MAS
        MOVX @DPTR,A    ; Manda imprimir
        MOV R6,#0FFH
        ACALL ESPERA    ; Espera un momento
        MOV A,#0FFH     ; Deshabilita calculadora
        MOVX @DPTR,A
        MOV R0,#07H
ESPFIN1: MOV R6,#0FFH
        ACALL ESPERA    ; Espera mucho tiempo, en lo que se imprime
        DJNZ R0,ESPFIN1 ;   la informacion.
        CLR 8           ; Limpia Bit 8
        RET

;************ Rutina: Lee_Informacion_de_la_NVRAM ************
; Rutina para leer la informacion de la NVRAM y copiarla en la
; RAM del uC. El Reg 2 debe tener un valor asignado antes de
; llamar a esta Rutina.
;       Reg 0 <- Apunta a la Dir. en donde se guardara la informacion
;                (A partir de la Dir 6FH y en decremento).
;       Reg 2 <- Contiene el No. de Bytes a Copiar
LEENVRAM:MOV DPH,66H    ; DPTR <- Toma su valor de las Dirs 66 y 67H
        MOV DPL,67H
        MOV R0,#6FH     ; La informacion se guarda a partir de la
LEESIG: MOVX A,@DPTR    ; Dir 6FH, y hacia abajo, tantos Bytes
        MOV @R0,A       ; como se requieran de acuerdo a Reg 2.
        INC DPTR
        DEC R0
        DJNZ R2,LEESIG
        MOV 66H,DPH     ; Dirs 66 y 67H <- DPTR
        MOV 67H,DPL
        RET

;************ Rutina: Suma_Dos_Numeros_en_BCD ***************
; Esta rutina suma el numero contenido en la Dir apuntada por R0
; al total guardado en la Dir apuntada por R1.
; Reg 0 <- Apunta al dato a sumar
; Reg 1 <- Apunta a la Dir. en la que va la suma.
; Reg 2 <- Contador del No. de bytes que faltan por sumar
; Reg 3 <- Ayuda para hacer la suma (Dir 03H)
; Reg 5 <- Dato a sumar
; Bandera 3 <- Lleva el "carry"
; Bandera 0 <- Ve si es nibble 0 o 1 (Der. o Izq.)
; Se debe dar un valor a Regs 0,1 y 2 antes de llamar a esta Rutina.

SUMABCD: CLR 0          ; Limpia bandera "nibble"
        CLR 3           ; Limpia bandera "carry"
SUMBCD: MOV A,@R0
        JNB 0,AQUI0     ; Ve si hay intercambio de nibbles
        SWAP A
AQUI0:  ANL A,#0FH      ; Obtiene nibble menos signif. dato a sumar
        MOV R3,A        ; Guarda nibble en Reg 3
        MOV A,@R1
        JNB 0,AQUI1     ; Ve si hay intercambio de nibbles
```

```
        SWAP A
AQUI1:  ANL A,#0FH      ; Obtiene nibble menos signif. de suma total
        ADD A,R3        ; Suma los nibbles
        DA A            ; Ajuste decimal
        JNB 3,AQUI2     ; Si no hay "carry" anterior brinca a AQUI2
        ADD A,#01H      ; Suma "carry"
        DA A            ; Ajuste decimal
        CLR 3           ; Limpia "carry"
AQUI2:  MOV R3,A        ; Reg 3 <- Suma parcial
        ANL A,#0F0H
        JZ AQUI3        ; Si hay carry,    Bit 3 <- 1
        SETB 3          ; Si NO hay carry, Bit 3 <- 0
AQUI3:  ANL 03H,#0FH    ; Reg 3 <- Reg 3 .AND. #0FH
        MOV A,@R1       ; Reg A <- Dato antiguo (suma total)
        JNB 0,AQUI4     ; Ve si hace intercambio de nibbles
        SWAP A
AQUI4:  ANL A,#0F0H     ; Actualiza suma total
        ORL A,R3
        JNB 0,AQUI5     ; Ve si hace intercambio de nibbles
        SWAP A
AQUI5:  MOV @R1,A
        CPL 0           ; Complementa bandera de "nibble"
        JB 0,SUMBCD
        DEC R1
        DEC R0
        MOV A,R0
        XRL A,60H
        JNZ AQUI6
        INC R0
        MOV 61,#00H     ; Reg 0 <- Apuntara al nuevo sumando (0)
AQUI6:  DJNZ R2,SUMBCD
        RET

;***************** Rutina: ApuntaRAM_a_DPTR ********************
;     Rutina para copiar el contenido de las Dirs 16 y 17H
;          (Apuntador RAM de Dallas) al DPTR.
APaDPTR: MOV DPTR,#16H  ; Copia el Apuntador de la RAM de Dallas a
        MOVX A,@DPTR    ; las Dirs. 66 y 67H de la RAM interna del uC.
        MOV 66H,A
        INC DPTR
        MOVX A,@DPTR
        MOV 67H,A
        MOV DPH,66H     ; Copia Dir 66 y 67H al DPTR.
        MOV DPL,67H     ; Al salir de la Rutina, el DPTR contiene la
        RET             ;   Dir en que se escribira la informacion.

;***************** Rutina: DPTR_a_ApuntaRAM ******************
;     Rutina para actualizar el valor del Apuntador de la
;                    RAM de Dallas
DPTRaAP: INC DPTR       ; Primero incrementa el DPTR
        MOV 66H,DPH     ; Copia el Apuntador a las localidades corres-
        MOV 67H,DPL     ; pondientes de la RAM de Dallas (Dirs 16 y 17H)
        MOV DPTR,#16H
        MOV A,66H
        MOVX @DPTR,A
        INC DPTR
        MOV A,67H
        MOVX @DPTR,A
        RET

;************* Rutina: Actualizacion_de_Velocidad *************
; Rutina para guardar la distancia recorrida y deteccion de un
; exceso de velocidad. Al tomar lecturas se obtuvieron 5.2 vueltas
; por metro; sin embargo, en pruebas posteriores que abarcaron una
; mayor distancia se obtuvieron 5 vueltas por metro. En comentarios
; estan las operaciones para hacer el ajuste de 5.2 vueltas/mt.
ACTUAVEL:CLR 5          ; Apaga indicador de 1 seg. transcurrido
        MOV 62H,TL0     ; Dir 62H <- No. hoyos
;        MOV A,TL0
```

```
;         ADD A,61H
;RES26:   CLR C
;         SUBB A,#260       ; Restara 1 hoyo por cada 26 hoyos
;         JC FINC26         ;  registrados.
;         DEC 62H
;         AJMP RES26

;FINC26:  ADD A,#260        ; Lo que sobre se guarda en Dir. 61H
;         MOV 61H,A

          MOV B,#HOYMT      ; Reg B <- Hoyos/mt
          MOV A,62H         ; Reg A <- No. hoyos
          DIV AB            ; A <- Veloc (m/s) en hexadecimal
          MOV 62H,A         ; 62H <- Guarda Veloc Actual
          MOV TL0,B         ; TL0 <- Guarda vueltas no usadas

;         MOV A,61H
;         CLR C             ; Se resta las vueltas no usadas al
;         SUBB A,B          ;  contenido de Dir. 61H
;         MOV 61H,A


                        ;*** Verifica si hay violacion de veloc.
          MOV A,52H         ; Dir. 52H <- Veloc. Maxima
          CLR C
          SUBB A,62H        ; A <- VelocMax - VelocActual
          JNC APABAN        ; Si No Violacion: Brinca a APABAN

          JB 4,NOVIOL       ; Si bandera prendida, Brinca a NOVIOL
                            ; La Bandera indica que ya se registro
                            ; dicha violacion.
VIOLVEL:  MOV DPTR,#000BH
          MOVX A,@DPTR      ; Actualiza No. violaciones de velocidad
          ADD A,#01H        ; en la RAM de Dallas (Dirs. 0A y 0BH).
          DA A
          MOVX @DPTR,A
          SETB 4            ; Prende Bandera que indica que se
          JNC ESCVIOL       ; registra violacion de velocidad.
          MOV DPTR,#0AH
          MOVX A,@DPTR
          ADD A,#01H
          DA A
          MOVX @DPTR,A
ESCVIOL:  ACALL APaDPTR     ; Registra en la RAM de Dallas la Distancia
          MOV A,#01H        ; que se llevaba recorrida al momento de
          MOVX @DPTR,A      ; efectuarse la violacion.
          INC DPTR          ; A la informacion le antecede el numero 1,
          MOV A,58H         ; que indica que se trata de una violacion
          MOV R0,#64H       ; de velocidad.
          MOV 64H,59H
          MOV 65H,5AH
          XCHD A,@R0        ; Escribe lsn de Dir 58H, Dir 59H y msn de
          INC R0            ; Dir. 5AH, para registrar kilometros
          XCHD A,@R0        ; solamente. Lo hacemos usando como interme-
          MOV A,64H         ; diario las Dirs 64 y 65H.
          SWAP A
          MOVX @DPTR,A
          INC DPTR
          MOV A,65H
          SWAP A
          MOVX @DPTR,A
          ACALL DPTRaAP
          AJMP MOVIOL

APABAN:   CLR 4             ; Apaga Bandera de Violacion de Velocidad
MOVIOL:   ACALL HEXABCD     ; Convierte Distancia de Hex a BCD

                  ;*** Actualiza Distancia Parcial Recorrida.
          MOV R0,#62H       ; Reg 0 <- Apunta al Dato a sumar
```

```
        MOV R1,#5FH     ; Reg 1 <- Apunta a Dir. en donde se hace la suma.
        MOV R2,#04H     ; Reg 2 <- No. bytes a sumar
        ACALL SUMABCD
              ;*** Actualiza Distancia Total Recorrida.
        MOV R0, #62H    ; Reg 0 <- Apunta al Dato a sumar
        MOV R1,#5BH     ; Reg 1 <- Apunta a Dir. en donde se hace la suma.
        MOV R2,#04H     ; Reg 2 <- No. bytes a sumar
        ACALL SUMABCD

;*** Copia Distancias Parcial y Total a la RAM de Dallas cada minuto
;*** (actualizacion por minuto) ...
        MOV A,3EH
        XRL A,#00H
        JNZ ACTDISPL
        MOV R0,#58H     ; Reg 0 <- Fuente
        MOV R1,#08H     ; Reg 1 <- No. Bytes a copiar
        MOV DPTR,#0018H ; DPTR  <- Destino
ACTVEL: MOV A,@R0
        MOVX @DPTR,A
        INC DPTR
        INC R0
        DJNZ R1,ACTVEL
        RET             ; Sale de la Rutina de Actualizacion

;************ Rutina: Convierte_de_Hexadecimal_a_BCD**********
; Convierte de Hexadecimal a BCD el dato localizado en la Dir 62H
; de la memoria interna del uC.

HEXABCD: MOV A,62H      ; Dir 62H <- Veloc. en Hexadecimal.
        SWAP A
        ANL A,#0FH
        MOV R2,A        ; Reg 2 <- Guarda el nibble mas significativo
        MOV R5,#06H     ; Reg 5 <- Guarda dato a sumar (es #06H)
        MOV A,62H
        ANL A,#0F0H     ; Si veloc < 10H, brinca a AJUSTE
        JNZ AJUSTE
        MOV R2,#01H
        MOV R5,#00H     ; Reg 5 <- Cambia su dato a #00H
AJUSTE: MOV A,62H
SUMA:   ADD A,R5
        DA A
        DJNZ R2,SUMA
        MOV 62H,A       ; Dir 62H <- Guarda el resultado en BCD
        RET


;******************** Rutina: Fin_de_Ruta *********************
; Rutina para guardar el No. de Violaciones de Velocidad, la
; Distancia Recorrida en la Ruta y la Hora de Finalizacion de
; la Ruta, en la RAM de Dallas

FINRUTA: MOV R0,#58H    ; Escribe Dist. Parc. y Total a NVRAM
        MOV R1,#08H
        MOV DPTR,#0018H ; Reg 0 <- Fuente
AV:     MOV A,@R0       ; Reg 1 <- No. Bytes a copiar
        MOVX @DPTR,A    ; DPTR  <- Destino
        INC DPTR
        INC R0
        DJNZ R1,AV

        MOV DPTR,#0AH  ; Guarda el No. de violaciones de veloc
        MOVX A,@DPTR   ; en Dirs 6C y 6DH RAM uC
        MOV 6CH,A
        INC DPTR
        MOVX A,@DPTR
        MOV 6DH,A
        SETB 6
        ACALL HORA     ; Obtiene la Hora de Fin de Ruta.
```

```
            ;*** Copia informacion a RAM Dallas
ACALL APaDPTR
MOV A,#02H
MOVX @DPTR,A    ; Escribe el No. 2 y copia el No. de
INC DPTR        ;   violaciones de velocidad
MOV A,6CH
MOVX @DPTR,A
INC DPTR
MOV A,6DH
MOVX @DPTR,A
INC DPTR

MOV A,#03H      ; Escribe el No. 3 y copia la Distancia
MOVX @DPTR,A    ;  Recorrida en la Ruta.
INC DPTR
MOV A,58H
MOV R0,#59H
XCHD A,@R0      ; Escribe Isn de Dir 58H, Dir 59H y msn de
INC R0          ; Dir. 5AH, para registrar kilometros
XCHD A,@R0      ; solamente.
MOV A,59H
SWAP A
MOVX @DPTR,A
INC DPTR
MOV A,5AH
SWAP A
MOVX @DPTR,A
INC DPTR

MOV A,6EH       ; Escribe la Hora de Finalizacion del Recorrido
MOVX @DPTR,A
INC DPTR
MOV A,6FH
MOVX @DPTR,A
ACALL DPTRaAP
RET

END;
```

Apéndice C:

Hojas de

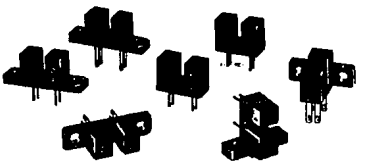
Especificaciones

Display Font	Color	Common Cathode	Common Anode
0.3 inch (7.6 mm) Character Height			
	RED GREEN	AND332R AND332G	AND333R AND333G
	RED GREEN	AND322R AND322G	AND323R AND323G
	RED GREEN	AND4125R AND4125G	AND4115R AND4115G
0.43 inch (10.9 mm) Character Height			
	RED GREEN	AND342R AND342G	AND343R AND343G
0.5 inch (12.7 mm) Character Height			
	RED GREEN	AND352R AND352G	AND353R AND353G
	RED GREEN	AND324R AND324G	AND325R AND325G
	RED GREEN	AND370R AND370G	AND371R AND371G
0.56 inch (14.2 mm) Character Height			
	RED GREEN	AND362R AND362G	AND363R AND363G
	RED GREEN	AND366R AND366G	AND367R AND367G
0.8 (20.3 mm) Character Height)			
	RED GREEN YELLOW ORANGE HI-EFF RED	AND8010RCLB AND8010GCLB AND8010YCLB AND8010OCLB AND8010HCLB	AND8010RALB AND8010GALB AND8010YALB AND8010OALB AND8010HALB
1.2 inch (45.0 mm) Character Height			
	RED (GaAsP) GREEN RED (GaP)	AND380R AND380G AND380S	—
	RED	AND4322R	—

Bar Graphs

Display Font	Color	Type No.
	RED GREEN	AND8101R AND8101G
	RED GREEN	AND8101M
	RED GREEN	AND8102M

A Series of Phototransistor Output Sensors with Excellent Sensing Position Characteristics and High Resolution



- High-resolution model detecting 0.2-mm-dia. objects and high-sensitivity model detecting 1.0-mm-dia. objects.
- Incorporating a center mark for optical axis adjustment with ease.
- Incorporating soldering terminals (EE-SH3) and PCB terminals (EE-SH3-B).
- Compact models with no mounting tab ideal to be built into various equipment (EE-SJ3-C/EE-SJ3-D/EE-SJ3-G).

Ordering Information

Appearance	Sensing method	Slot width	Sensing object	Output configuration	Model	Weight
Soldering terminals 	Transmissive	3.4 mm	Opaque, 0.5 x 2.1 mm min.	Phototransistor	EE-SH3	Approx. 0.8 g
					EE-SV3	Approx. 0.7 g
			Opaque, 1.0 x 2.1 mm min.		EE-SH3-CS	Approx. 0.8 g
					EE-SV3-CS	Approx. 0.7 g
			Opaque, 0.2 x 2.1 mm min.		EE-SH3-DS	Approx. 0.8 g
					EE-SV3-DS	Approx. 0.7 g
			Opaque, 2.1 x 0.5 mm min.		EE-SH3-GS	Approx. 0.8 g
					EE-SV3-GS	Approx. 0.7 g
PCB terminals 			Opaque, 0.5 x 2.1 mm min.		EE-SH3-B	Approx. 0.8 g
					EE-SV3-B	Approx. 0.7 g
			Opaque, 1.0 x 2.1 mm min.		EE-SH3-C	Approx. 0.8 g
					EE-SV3-C	Approx. 0.7 g
					EE-SJ3-C	Approx. 0.6 g
			Opaque, 0.2 x 2.1 mm min.		EE-SH3-D	Approx. 0.8 g
					EE-SV3-D	Approx. 0.7 g
					EE-SJ3-D	Approx. 0.6 g
			Opaque, 2.1 x 0.5 mm min.		EE-SH3-G	Approx. 0.8 g
					EE-SV3-G	Approx. 0.7 g
					EE-SJ3-G	Approx. 0.6 g

Specifications

■ ABSOLUTE MAXIMUM RATINGS (T_A = 25°C)

Item		Symbol	Rated value
Input	Forward current	I_F	50 mA*
	Reverse voltage	V_R	4 V
Output	Collector-emitter voltage	V_{CEO}	30 V
	Collector current	I_C	20 mA
	Collector dissipation	P_C	100 mW*
Ambient temperature	Operating	Topr	−25° to 85°C
	Storage	Tstg	−30° to 100°C

Refer to Engineering Data if the ambient temperature is not within the normal room temperature range.

National Semiconductor Corporation

LM78LXX Series 3-Terminal Positive Regulators

General Description

The LM78LXX series of three terminal positive regulators is available with several fixed output voltages making them useful in a wide range of applications. When used as a zener diode/resistor combination replacement, the LM78LXX usually results in an effective output impedance improvement of two orders of magnitude, and lower quiescent current. These regulators can provide local on card regulation, eliminating the distribution problems associated with single point regulation. The voltages available allow the LM78LXX to be used in logic systems, instrumentation, HiFi, and other solid state electronic equipment. Although designed primarily as fixed voltage regulators these devices can be used with external components to obtain adjustment voltages and currents.

The LM78LXX is available in the metal three lead TO-39(H) the plastic TO-92 (Z), and SO-8 plastic. With adequate heat sinking the regulator can deliver 100 mA output current. Current limiting is included to limit the peak output current to a safe value. Safe area protection for the output transistors is provided to limit internal power dissipation. If internal power dissipation becomes too high for the heat sinking provided, the thermal shutdown circuit takes over preventing the IC from overheating.

For output voltage other than 5V, 12V and 15V the LM117L series provides an output voltage range from 1.2V to 37V.

Features

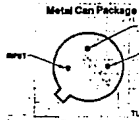
- Output voltage tolerances of ±5% (LM78LXXAC) over the temperature range
- Output current of 100 mA
- Internal thermal overload protection
- Output transistor safe area protection
- Internal short circuit current limit
- Available in plastic TO-92 and metal TO-39 and plastic SO-8 low profile packages

Voltage Range

LM78L05	5V
LM78L12	12V
LM78L15	15V

Connection Diagrams

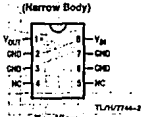
Metal Can Package



Bottom View

Order Number LM78L05ACH, LM78L12ACH or LM78L15ACH
See NS Package Number H03A

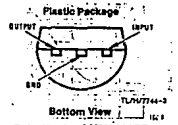
SO-8 Plastic (Narrow Body)



Top View

Order Number LM78L05ACM, LM78L12ACM or LM78L15ACM
See NS Package Number M08A

Plastic Package



Bottom View

Order Number LM78L05ACZ, LM78L12ACZ or LM78L15ACZ
See NS Package Number Z03A

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Input Voltage
- V_O = 5V: 30V
- V_O = 12V to 15V: 35V

Internal Power Dissipation (Note 1): Internally Limited

Operating Temperature Range: 0°C to +70°C
Maximum Junction Temperature: 125°C
Storage Temperature Range
- Metal Can (H Package): −65°C to +150°C
- Molded TO-92 (Z Package): −55°C to +150°C

Lead Temperature (Soldering, 10 sec.): 260°C

LM78LXXAC Electrical Characteristics

(Note 2) T_j = 0°C to 125°C, I_O = 40 mA, C_{IN} = 0.33 µF, C_O = 0.1 µF (unless noted)

LM78LXXAC Output Voltage			5V			12V			15V			Units
Input Voltage (unless otherwise noted)			10V			19V			23V			
Symbol	Parameter	Conditions	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
V_O	Output Voltage (Note 4)	T_j = 25°C	4.8	5	5.2	11.5	12	12.5	14.4	15	15.0	V
		1 mA ≤ I_O ≤ 70 mA	4.75		5.25	11.4		12.6	14.25		15.75	V
		1 mA ≤ I_O ≤ 40 mA and V_{MIN} ≤ V_{IN} ≤ V_{MAX}	4.75 (7 ≤ V_{IN} ≤ 20)		5.25	11.4 (14.5 ≤ V_{IN} ≤ 27)		12.6	14.25 (17.5 ≤ V_{IN} ≤ 30)		15.75	V
ΔV_O	Line Regulation	T_j = 25°C		10	54		20	110		25	140	mV
				(8 ≤ V_{IN} ≤ 20)			(16 ≤ V_{IN} ≤27)			(20 ≤ V_{IN} ≤ 30)		V
				18	75		30	180		37	250	mV
				(7 ≤ V_{IN} ≤ 20)			(14.5 ≤ V_{IN} ≤ 27)			(17.5 ≤ V_{IN} ≤ 30)		V
ΔV_O	Load Regulation	T_j = 25°C, 1 mA ≤ I_O ≤ 40 mA		5	30		10	50		12	75	mV
		T_j = 25°C, 1 mA ≤ I_O ≤ 100 mA		20	60		30	100		35	150	mV
ΔV_O	Long Term Stability			12			24			30		mV/1000 hrs
I_Q	Quiescent Current	T_j = 25°C		3	5		3	5		3.1	5	mA
		T_j = 125°C			4.7			4.7			4.7	mA
ΔI_Q	Quiescent Current Change	1 mA ≤ I_O ≤ 40 mA			0.1			0.1			0.1	mA
		V_{MIN} ≤ V_{IN} ≤ V_{MAX}			1.0			1.0			1.0	mA
				(8 ≤ V_{IN} ≤ 20)			(16 ≤ V_{IN} ≤ 27)			(20 ≤ V_{IN} ≤ 30)		V
V_n	Output Noise Voltage	T_j = 25°C, (Note 3) f = 10 Hz – 10 kHz		40			80			90		µV
$\frac{\Delta V_{IN}}{\Delta V_{OUT}}$	Ripple Rejection	f = 120 Hz	47	62		40	54		37	51		dB
				(8 ≤ V_{IN} ≤ 18)			(15 ≤ V_{IN} ≤ 25)			(18.5 ≤ V_{IN} ≤ 28.5)		V
	Input Voltage Required to Maintain Line Regulation	T_j = 25°C	7			14.5			17.5			V

Note 1: Thermal resistance of H package is typically 39°C/W θ_{ja} still Air, and 84°C/W θ_{ja} 400 ft/min of air. For the Z package is 60°C/W θ_{ja}, 230°C/W θ_{ja} still air, and 88°C/W θ_{ja} at 400 ft/min of air. The maximum junction temperature shall not exceed 125°C on Electrical parameters.

Note 2: The maximum steady state useable output current and input voltage are very dependent on the heat sinking and/or lead length of the package. The data shown represent pulse test conditions with junction temperatures as indicated at the initiation of test.

Note 3: Recommended minimum load capacitance of 0.27 µF to limit high frequency noise bandwidth.

Note 4: The temperature coefficient of V_{OUT} is typically within ±0.01% V_O/°C.

6-Pin DIP Optoisolators
Triac Driver Output

MOC3009
MOC3010
MOC3011
MOC3012

6-PIN DIP
OPTOISOLATORS
TRIAC DRIVER OUTPUT
250 VOLTS

CASE 730A-02
PLASTIC

These devices consist of gallium-arsenide infrared emitting diodes, optically coupled to silicon bilateral switch and are designed for applications requiring isolated triac triggering, low-current isolated ac switching, high electrical isolation (to 7500 V peak), high detector standoff voltage, small size, and low cost.

- UL Recognized File Number E54915
- VDE approved per standard 0883/6.80 (Certificate number 41853), with additional approval to DIN IEC380/VDE0806, IEC435/VDE0805, IEC65/VDE0860, VDE110b, covering all other standards with equal or less stringent requirements, including IEC204/VDE0113, VDE0160, VDE0832, VDE0833, etc.
- Special lead form available (add suffix "T" to part number) which satisfies VDE0883/6.80 requirement for 8 mm minimum creepage distance between input and output solder pads.
- Various lead form options available. Consult "Optoisolator Lead Form Options" data sheet for details.

MAXIMUM RATINGS (T_A = 25°C unless otherwise noted)

Rating	Symbol	Value	Unit
INFRARED EMITTING DIODE			
Reverse Voltage	V_R	3	Volts
Forward Current — Continuous	I_F	60	mA
Total Power Dissipation @ T_A = 25°C Negligible Power in Transistor Derate above 25°C	P_D	100 1.33	mW mW/°C
OUTPUT DRIVER			
Off-State Output Terminal Voltage	V_{DRM}	250	Volts
Peak Repetitive Surge Current (PW = 1 ms, 120 pps)	I_{TSM}	1	A
Total Power Dissipation @ T_A = 25°C Derate above 25°C	P_D	300 4	mW mW/°C
TOTAL DEVICE			
Isolation Surge Voltage (1) (Peak ac Voltage, 60 Hz, 5 Second Duration)	V_{ISO}	7500	Vac
Total Power Dissipation @ T_A = 25°C Derate above 25°C	P_D	330 4.4	mW mW/°C
Junction Temperature Range	T_J	−40 to +100	°C
Ambient Operating Temperature Range	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−40 to +150	°C
Soldering Temperature (10 s)	—	260	°C

(1) Isolation surge voltage, V_{ISO}, is an internal device dielectric breakdown rating.

COUPLER SCHEMATIC

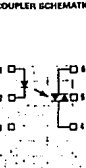


Figure 7. Static dv/dt Test Circuit

ELECTRICAL CHARACTERISTICS (T_A = 25°C unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
INPUT LED						
Reverse Leakage Current (V_R = 3 V)		I_R	—	0.05	100	μA
Forward Voltage (I_F = 10 mA)		V_F	—	1.15	1.5	Volts
OUTPUT DETECTOR (I_F = 0 unless otherwise noted)						
Peak Blocking Current, Either Direction (Rated V_{DRM}, Note 1)		I_{DRM}	—	10	100	nA
Peak On-State Voltage, Either Direction (I_{TM} = 100 mA Peak)		V_{TM}	—	1.8	3	Volts
Critical Rate of Rise of Off-State Voltage (Figure 7, Note 2)		dv/dt	—	10	—	V/μs
COUPLED						
LED Trigger Current, Current Required to Latch Output (Main Terminal Voltage = 3 V, Note 3)	MOC3009	I_{FT}	—	15	30	mA
	MOC3010		—	8	15	
	MOC3011		—	5	10	
	MOC3012		—	3	5	
Holding Current, Either Direction		I_H	—	100	—	μA

Notes: 1. Test voltage must be applied within dv/dt rating.
2. This is static dv/dt. See Figure 7 for test circuit. Commutating dv/dt is a function of the load-driving thyristor(s) only.
3. All devices are guaranteed to trigger at an I_F value less than or equal to max I_{FT}. Therefore, recommended operating I_F lies between max I_{FT} (30 mA for MOC3009, 15 mA for MOC3010, 10 mA for MOC3011, 5 mA for MOC3012) and absolute max I_F (60 mA).

TYPICAL ELECTRICAL CHARACTERISTICS

T_A = 25°C

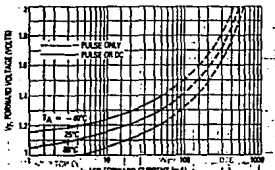


Figure 1. LED Forward Voltage versus Forward Current

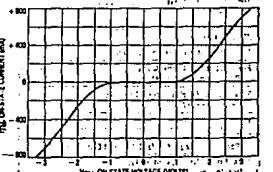
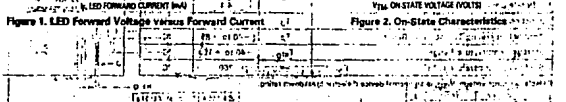


Figure 2. On-State Characteristics



- Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers and Flat Packages, and Plastic and Ceramic DIPs
- Dependable Texas Instruments Quality and Reliability

description

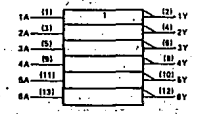
These devices contain six independent inverters.

The SN5404, SN54LS04, and SN54S04 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN7404, SN74LS04, and SN74S04 are characterized for operation from 0°C to 70°C.

FUNCTION TABLE (each inverter)

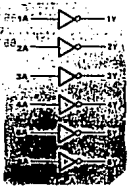
INPUTS A	OUTPUT Y
H	L
L	H

logic symbol†

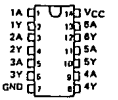


†This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12. Pin numbers shown are for D, J, and N packages.

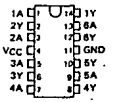
logic diagram (positive logic)



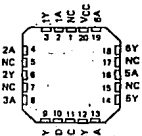
SN5404 . . . J PACKAGE
SN54LS04, SN54S04 . . . J OR W PACKAGE
SN7404 . . . N PACKAGE
SN74LS04, SN74S04 . . . D OR N PACKAGE
(TOP VIEW)



SN5404 . . . W PACKAGE
(TOP VIEW)



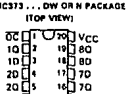
SN54LS04, SN54S04 . . . FK PACKAGE
(TOP VIEW)



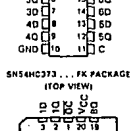
NC – No internal connection

- 8 High-Current Latches in a Single Package
- High-Current 3-State True Outputs Can Drive Up to 15 LSTTL Loads
- Full Parallel Access for Loading
- Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers, and Standard Plastic and Ceramic 300-mil DIPs
- Dependable Texas Instruments Quality and Reliability

SN54HC373 . . . J PACKAGE
SN74HC373 . . . DW OR N PACKAGE
(TOP VIEW)



SN54HC373 . . . FK PACKAGE
(TOP VIEW)



description

These 8-bit latches feature three-state outputs designed specifically for driving highly capacitive or relatively low-impedance loads. They are particularly suitable for implementing buffer registers, I/O ports, bidirectional bus drivers, and working registers.

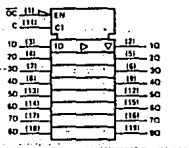
The eight latches of the 'HC373 are transparent D-type latches. While the enable (C) is high, the Q outputs will follow the data (D) inputs. When the enable is taken low, the Q outputs will be latched at the levels that were set up at the D inputs.

An output-control input ($\overline{OC}$) can be used to place the eight outputs in either a normal logic state (high or low logic levels) or a high-impedance state. In the high-impedance state the outputs neither load nor drive the bus lines significantly. The high-impedance third state and increased drive provide the capability to drive the bus lines in a bus-organized system without need for interface or pull-up components.

The output control ($\overline{OC}$) does not affect the internal operations of the latches. Old data can be retained or new data can be entered while the outputs are off.

The SN54HC373 is characterized for operation over the full military temperature range of −55°C to 125°C. The SN74HC373 is characterized for operation from −40°C to 85°C.

logic symbol†



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

FUNCTION TABLE
(EACH LATCH)

INPUTS			OUTPUT
OC	ENABLE C	D	Q
L	H	H	H
L	H	L	L
L	L	X	Q_0
H	X	X	Z

H = high level, Z = low level, X = irrelevant

absolute maximum ratings over operating free-air temperature range†

Supply voltage, V_{CC} ..... −0.5 V to 7 V
Input clamp current, I_{IK} ($V_I < 0$ or $V_I > V_{CC}$) ..... ±20 mA
Output clamp current, I_{OK} ($V_O < 0$ or $V_O > V_{CC}$ ..... ±20 mA
Continuous output current, I_O (V_O = 0 to V_{CC}) ..... ±35 mA
Continuous current through V_{CC} or GND pins ..... ±70 mA
Lead temperature 1,6 mm (1/16 in) from case for 60 s: FK or J package ..... 300°C
Lead temperature 1,6 mm (1/16 in) from case for 10 s: DW or N package ..... 260°C
Storage temperature range ..... −65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

2 HCMOS Devices

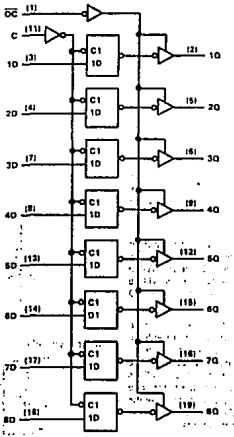
recommended operating conditions

			SN54HC373			SN74HC373			UNIT
			MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage		2	5	6	2	5	6	V
V_{IH}	High-level input voltage	V_{CC} = 2 V	1.5			1.5			V
		V_{CC} = 4.5 V	3.15			3.15			
		V_{CC} = 6 V	4.2			4.2			
V_{IL}	Low-level input voltage	V_{CC} = 2 V	0		0.3	0		0.3	V
		V_{CC} = 4.5 V	0		0.8	0		0.8	
		V_{CC} = 6 V	0		1.2	0		1.2	
V_I	Input voltage		0		V_{CC}	0		V_{CC}	V
V_O	Output voltage		0		V_{CC}	0		V_{CC}	V
t_t	Input transition rise and fall times	V_{CC} = 2 V	0		1000	0		1000	ns
		V_{CC} = 4.5 V	0		500	0		500	
		V_{CC} = 6 V	0		400	0		400	
T_A	Operating free-air temperature		−55		125	−40		85	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS	V_{CC}	T_A = 25°C			SN54HC373		SN74HC373		UNIT
			MIN	TYP	MAX	MIN	MAX	MIN	MAX	
V_{OH}	V_I = V_{IH} or V_{IL}, I_{OH} = −20 μA	2 V	1.9	1.998		1.9		1.9		V
		4.5 V	4.4	4.499		4.4		4.4		
		6 V	5.9	5.999		5.9		5.9		
	V_I = V_{IH} or V_{IL}, I_{OH} = −6 mA	4.5 V	3.98	4.30		3.7		3.84		
	V_I = V_{IH} or V_{IL}, I_{OH} = −7.8 mA	6 V	5.48	5.80		5.2		5.34		
V_{OL}	V_I = V_{IH} or V_{IL}, I_{OL} = 20 μA	2 V		0.002	0.1		0.1		0.1	V
		4.5 V		0.001	0.1		0.1		0.1	
		6 V		0.001	0.1		0.1		0.1	
	V_I = V_{IH} or V_{IL}, I_{OL} = 6 mA	4.5 V		0.17	0.26		0.4		0.33	
	V_I = V_{IH} or V_{IL}, I_{OL} = 7.8 mA	6 V		0.18	0.26		0.4		0.33	
I_I	V_I = V_{CC} or 0	6 V		±0.1	±100		±1000		±1000	nA
I_{OZ}	V_O = V_{CC} or 0	6 V		±0.01	±0.5		±10		±5	μA
I_{CC}	V_I = V_{CC} or 0, I_O = 0	6 V			8		160		80	μA
C_i		2 to 6 V		3	10		10		10	pF

logic diagram (positive logic)

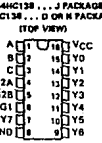


SN54HC138, SN74HC138
3-LINE TO 8-LINE DECODERS/DEMULTIPLEXERS

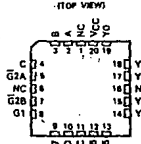
D2684, DECEMBER 1982 – REVISED SEPTEMBER 1997

- Designed Specifically for High-Speed Memory Decoders and Data Transmission Systems
- Incorporates 3 Enable Inputs to Simplify Cascading and/or Data Reception
- Package Options: Plastic and Ceramic DIPs, Plastic Small-Outline Packages, and Ceramic Chip Carriers
- Dependable Texas Instruments Quality and Reliability

SN54HC138 . . . J PACKAGE
SN74HC138 . . . D OR N PACKAGE
(TOP VIEW)



SN54HC138 . . . FK PACKAGE
(TOP VIEW)



NC—No internal connection

description

The 'HC138 circuit is designed to be used in high-performance memory-decoding or data-routing applications requiring very short propagation delay times. In high-performance memory systems this decoder can be used to minimize the effects of system decoding. When employed with high-speed memories utilizing a fast enable circuit, the delay times of this decoder and the enable time of the memory are usually less than the typical access time of the memory. This means that the effective system delay introduced by the decoder is negligible.

The conditions at the binary select inputs and the three enable inputs select one of eight output lines. Two active-low and one active-high enable inputs reduce the need for external gates or inverters when expanding. A 24-line decoder can be implemented without external inverters and a 32-line decoder requires only one inverter. An enable input can be used as a data input for demultiplexing applications.

The SN54HC138 is characterized for operation over the full military temperature range of −55°C to 125°C. The SN74HC138 is characterized for operation from −40°C to 85°C.

FUNCTION TABLE

ENABLE INPUTS			SELECT INPUTS			OUTPUTS							
G1	$\overline{G2A}$	$\overline{G2B}$	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

absolute maximum ratings over operating free-air temperature range†

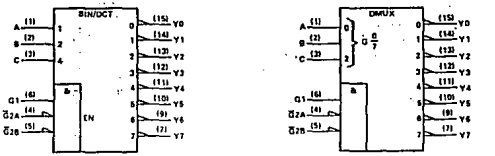
Supply voltage, V_{CC} . . . . . −0.5 V to 7 V
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{CC}) . . . . . ±20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{CC}) . . . . . ±20 mA
Continuous output current, I_O (V_O = 0 to V_{CC}) . . . . . ±25 mA
Continuous current through V_{CC} or GND pins . . . . . ±50 mA
Lead temperature 1,6 mm (1/16 in) from case for 60 s: FK or J package . . . . . 300°C
Lead temperature 1,6 mm (1/16 in) from case for 10 s: D or N package . . . . . 260°C
Storage temperature range . . . . . −65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

recommended operating conditions

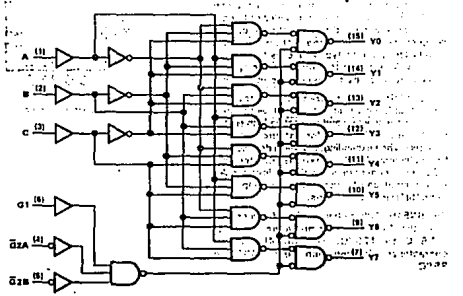
		SN54HC138			SN74HC138			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC} Supply voltage		2	5	6	2	5	6	V
V_{IH} High-level input voltage	V_{CC} = 2 V	1.5			1.5			V
	V_{CC} = 4.5 V	3.15			3.15			
	V_{CC} = 6 V	4.2			4.2			
V_{IL} Low-level input voltage	V_{CC} = 2 V	0		0.3	0		0.3	V
	V_{CC} = 4.5 V	0		0.9	0		0.9	
	V_{CC} = 6 V	0		1.2	0		1.2	
V_I Input voltage		0		V_{CC}	0		V_{CC}	V
V_O Output voltage		0		V_{CC}	0		V_{CC}	V
t_t Input transition (rise and fall) times	V_{CC} = 2 V	0		1000	0		1000	ns
	V_{CC} = 4.5 V	0		500	0		500	
	V_{CC} = 6 V	0		400	0		400	
T_A Operating free-air temperature		−55		125	−40		85	°C

logic symbols (alternatives)†



† These symbols are in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for D, J, and N packages.

logic diagram (positive logic)



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

Apéndice D:

Pruebas de impresión

usando la sumadora

TI-5029

```
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
123;456. +
7;890.3 +
44.781 +
```

Pruebas de impresión.

Información de identificación:

```
No. de identificación del chofer --------- 101• +
Fecha de impresión ------------------------ 0•210194 +
Hora de impresión ------------------------- 21•15 +
```

Información por Ruta:

```
No. Ruta ---------------------------------- 1• +
Fecha de inicio de Ruta ------------------- 0•190194 +
Hora de inicio de Ruta -------------------- 13•25 +

Distancia en que ocurrió violación vel. ----- 2•1 +
Distancia en que ocurrió violación vel. ----- 4•1 +
Distancia en que ocurrió violación vel. ----- 8•1 +
No. viol. veloc. en la Ruta ---------------- 3•2 +
Distancia  recorrida en la Ruta ------------ 12•3 +

Hora de finalización de la Ruta ------------ 13•52 +



No. Ruta ---------------------------------- 2• +
Fecha de inicio de Ruta ------------------- 0•190194 +
Hora de inicio de Ruta -------------------- 13•52 +

Distancia en que ocurrió violación vel. ----- 4•1 +
No. viol. de veloc. en la Ruta ------------- 1•2 +
Distancia recorrida en la Ruta ------------- 6•3 +

Hora de finalización de la Ruta ------------ 14•02 +
```

Ejemplo de información impresa

No. Ruta	3•	+
Fecha de inicio de Ruta	0•200194	+
Hora de inicio de Ruta	12•15	+
Distancia en que ocurrió viol. veloc.	3•1	+
Distancia en que ocurrió viol. veloc.	[illegible]	-
Distancia en que ocurrió viol. veloc.	4•1	+
Distancia en que ocurrió viol. veloc.	6•1	+
No. viol. veloc. en la Ruta	4•2	+
Distancia recorrida en la Ruta	7•3	+
Hora de finalización de la Ruta	12•31	+

Información final:

Distancia total recorrida	26•11	+
No. total de viol. de velocidad	8•12	+
No. total de violaciones al monitor	3•13	+
No. de borradas de la NVRAM	14•14	+
No. de cambios de clave al sistema	0•15	+

Ejemplo de información impresa

Bibliografía

PROGRAMMING & DESIGNING WITH THE 68000 FAMILY
MIMAR, Tibet
QA76.8 M6895 M55

MICROPROCESADORES, DISPOSITIVOS PERIFÉRICOS, OPTOELECTRÓNICOS Y DE INTERFAZ
WILLIAMS, Arthur R.
Ed. Mc. Graw-Hill
1a. edición, 1989.

LOGICA DIGITAL Y DISEÑO DE COMPUTADORES
MANO, Morris M.
Ed. Prentice-Hall Hispanoamericana, S.A.
1a. edición, 1982

CIRCUITOS INTEGRADOS LINEALES Y AMPLIFICADORES OPERACIONALES
COUGHLIN,Robert F.; DRISCOLL, Frederick F.
Ed. Prentice-Hall-Hispanoamericana, S.A.
2a. edición, 1987.

ELECTRONICA, TEORIA DE CIRCUITOS
BOYLESTAD, Robert; NASHELSKY, Louis
Ed. Prentice-Hall Hispanoamericana, S.A.
4a. edición, 1989.

METODOLOGÍA DEL DISEÑO
FERNÁNDEZ Moreno, Raymundo
Seminario de Ingeniería Mecánica, U.N.A.M.
Diciembre 7, 1993.

FUNDAMENTOS DE INGENIERÍA: MÉTODOS, CONCEPTOS Y RESULTADOS.
KRICK, Edward V.
Ed. LIMUSA
1a. edición, 1979

MICROPROCESADORES: TEORÍA Y PRÁCTICA
GARCÍA Guerra, Hugo Gilberto
Ed. LIMUSA, 1988

MICROPROCESADORES/MICROCOMPUTADORES: ARQUITECTURA, SOFTWARE Y SISTEMAS.
KHAMBATA, A.J.
Ed. Calypso, S.A. 1982

DISEÑO DE SISTEMAS MICROPROCESADORES
GARLAND, Harry
Ed. Paraninfo, 1982

THE 8080, 8085 & Z80 HARDWARE, SOFTWARE PROGRAMMING, INTERFACE & TROUBLESHOOTING
LaLOND, David
QA 76.8 I28 L35

UNDERSTANDING DIGITAL TROUBLESHOOTING
CANNON, Don L.
Ed. SAMS, 3a. Edición, 1991

FUNDAMENTALS OF ELECTRICITY & AUTOMOTIVE ELECTRICAL SYSTEMS
WEATHERS Jr., Tom / Hunter, Claude C.
Ed Prentice-Hall, 2a. edición, 1988
TL272 W43 1988

EQUIPO ELÉCTRICO DEL AUTOMÓVIL
WILLIAM, Crouse
Ed. MARCOMBO S.A., 4a edición, 1975
Barcelona, España

LA ELECTRICIDAD EN AUTOMÓVILES Y AVIONES
SAMANIEGO, José María
Ed. Aguilar, 4a. edición, 1960
Madrid, España

AUTOMOTIVE TECHNOLOGY TODAY
MITCHELL International Inc.
Ed. Prentice-Hall, 1a. edición, 1989
New Jersey, U.S.A.

8-BIT EMBEDDED CONTROLLERS
INTEL, 1990

LINEAR APPLICATIONS MANUAL
NATIONAL SEMICONDUCTOR, 1988

THE TTL DATA BOOK
TEXAS INSTRUMENTS, 1990

HCMOS DEVICES
TEXAS INSTRUMENTS

OPTOELECTRONIC DEVICES
MOTOROLA

ECG SEMICONDUCTORS MASTER REPLACEMENT GUIDE
Philips, ECG
2a. Impresión, Enero 1989

PHOTOMICROSENSORS CATALOG
OMRON, 1993

MICROELECTRONICS
SGS-THOMSON

DALLAS SEMICONDUCTOR DATA BOOK
DALLAS SEMICONDUCTOR

PERIPHERAL DEVICES
INTEL

NEWARK ELECTRONICS

DjKey

INTERSIL

CEKIT

THOMSON CATALOG