23
29

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES

"ARAGON"

SISTEMA DE TIEMPO REAL PARA LA INFORMACION DE PUNTOS DE OZONO EN EL D.F. Y AREA METROPOLITANA

T E S I S

Que para obtener el Título de:

INGENIERO EN COMPUTACION

P r e s e n t a n

ARTURO MARES GOMEZ

ROBERTO MONTES MONROY

Asesor: Ing. Juan Gastaldi Pérez

San Juan de Aragón, Edo. de Méx.

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

A mi padre, Margarito Montes, y, mi madre, María Monroy, por todo el apoyo, ejemplo y amor que me brindaron para culminar con éxito este primer gran objetivo de mi vida.

Roberto.

A mis padres y familia por todo el apoyo y cariño que me brindaron, ya que este titulo es el fruto de la unión que tenemos en la familia.

Arturo.

AGRADECEMOS

A nuestro Director de Tesis, Ing. Juan Gastaldi Peréz, con un especial reconocimiento por su acertada dirección e invaluable amistad;

A la Ing. Silvia Vega Muytoy por el apoyo que nos brindó para realizar el presente trabajo;

A los Sinodales por sus valiosos comentarios y correcciones;

A todos los profesores de la Escuela Nacional de Estudios Profesionales Aragón por su dedicación y empeño en proporcionarnos una educación integral.

AMG, RMM

DIRECTOR DE TESIS:

ING. JUAN GASTALDI PEREZ

JURADO:

ING. JUAN GASTALDI PEREZ
ING. SILVIA VEGA MUYTOY
ING. LILIA ENCISO GARCIA
ING. RAFAEL MORGAN VAZQUEZ
ING. JULIO BERNAL VAZQUEZ

I N D I C E

III. DISEÑO DEL SISTEMA DE TIEMPO REAL IMECA/STR.

IV. DESCRIPCION DEL SISTEMA IMECA/STR.

V. MANUAL DE USUARIO.

INTRODUCCION

El presente trabajo trata sobre el diseño e implementación de un sistema de Tiempo real. El STR construido consiste en un programa en lenguaje PASCAL que puede ser utilizado tanto por instituciones educativas (Universidades y escuelas técnicas) como por la industria. Debido a sus características interactivas, el STR (IMECA/STR) proporciona una herramienta invaluable a la gente responsable de las mediciones de ozono, en el medio ambiente permitiéndole verificar el número de puntos de ozono así como obtener reportes impresos.

La tesis consta de 5 capítulos y 2 apéndices :

En el Capitulo I se da una breve reseña histórica de los sistemas de información así como de las computadoras en general y se definen los conceptos básicos como son modelos, sistemas reales y computadoras.

En el Capitulo II se presentan las bases teóricas en el diseño de un sistema de tiempo real: Diagrama de transición de estados, estados del sistema, etc.

En el Capítulo III se discuten las consideraciones de diseño así como los posibles problemas que se presentan en el diseño e implementación de un STR.

En el Capitulo IV Se describen algunas características adicionales tales como despliegue de pantallas utilizando gráficos, así como la impresión de reportes.

El Capitulo V Constituye un manual de Usuario que contiene las indicaciones y los pasos necesarios para la correcta operación de IMECA/STR.

Finalmente, el apéndice A contiene listado fuente de la versión básica del programa y el apéndice B proporciona algunos ejemplos de la utilización de IMECA/STR.

Es nuestro deseo que el presente trabajo no solamente sirva como una invaluable herramienta para estudiantes y gente relacionada con el área de la Medición e Instrumentación, sino que constituya un punto de partida para futuros trabajos de tesis.

AMG/RMM

CAPITULO I.

I.- ANTECEDENTES Y GENERALIDADES.

I.1. INTRODUCCION.

Debido a la creciente contaminación en el valle de México, científicos e investigadores han visto la necesidad de construir sistemas automatizados para verificar y mantener con ciertas medidas los niveles tolerables de contaminación. En los últimos años, la cantidad de herramientas disponibles así como el avance en la tecnología nos permiten realizar estos sistemas con una alta confiabilidad, pero a su vez cada día se exige más rapidez, seguridad y eficiencia. Lo anterior ha provocado que los diseñadores utilicen las computadoras para satisfacer las necesidades de diseño, automatización y prueba.

Debido a la rápida evolución en la tecnología de los procesadores, es posible utilizar la computadora no solo como un medio para procesar datos a una gran velocidad, sino también para automatizar sistemas y procesos además de poder determinar la mejor solución en caso de desastre.

El término Proceso a Tiempo Real se refiere a que en estos sistemas (de Tiempo Real) se procesan todos los datos inmediatamente; El proceso instantáneo es una característica integral de los a sistemas de tiempo.

Este tipo de procesos de gran utilidad en todas aquellas industrias en las que se necesite un proceso de datos instantáneo como es el caso de las aerolíneas en su sistema de reservaciones a nivel mundial, en donde cualquier retraso en el manejo de dichas reservaciones podría causar molestias a los clientes o la mala asignación de los recursos de la aeronave, otro ejemplo de la utilización de este tipo de proceso es en las industrias que tienen sistemas para el control de procesos, como son calderas y hornos dónde cualquier aumento en la temperatura podría causar un desastre y las pérdidas incalculables. Por último, en las bolsas de valores donde el volumen de acciones comerciales y la velocidad con la cual estas deben reportares, y los sistemas de tiempo real son los únicos que pueden cumplir con la tarea. Imaginemos el caos que podría producirse debido a la incapacidad para reportar oportunamente los intercambios de acciones a toda la comunidad financiera.

I.2. SISTEMAS.

Por sistema se entiende una colección de entidades relacionadas, cada una de las cuales se caracteriza por atributos o características que pueden estar relacionadas entre si.

Todo sistema consta de tres características: tiene fronteras, existe dentro de un medio ambiente y tiene subsistemas.

I.2.1. Tipos comunes de sistemas.

Como podemos ver de la definición anterior, existen muchos tipos diferentes de sistemas; de hecho casi todo aquello con lo cual entramos en contacto durante nuestra vida cotidiana es un sistema o bien parte de un sistema (o ambas cosas).

El objetivo de esta tesis es dar a conocer los diferentes tipos de sistemas computacionales y en especial los sistemas de tiempo real, pero empezaremos por dividir todos los sistemas en dos categorías: sistemas naturales y sistemas hechos por el hombre.

I.2.2. Sistemas naturales.

La gran mayoria de los sistemas no están hechos por el hombre; existen en la naturaleza y sirven a sus propios fines. Podemos dividir los sistemas naturales en dos subcategorias básicas: sistemas físicos y sistemas vivientes. Los sistemas físicos incluyen ejemplos como:

- Sistemas estelares : galaxias, sistemas solares, etc.
- Sistemas geológicos : rios, cordilleras, etc.
- Sistemas moleculares: organizaciones complejas de átomos.

Los sistemas vivientes comprenden toda la gama de animales y plantas que nos rodean, al igual que la raza humana. Esta categoría también comprende jerarquías de organismos vivientes individuales, por ejemplo hierbas, manadas, tribus, grupos sociales, compañías y naciones.

I.2.3. Sistemas hechos por el hombre.

Como se puede apreciar de la definición que se encuentra al principio de este punto, un buen numero de sistemas son construidos, organizados y mantenidos por el hombre, e incluyen:

- Sistemas sociales: organizaciones de leyes, doctrinas, costumbres, etc.
- Sistemas de comunicación: teléfono, telex, señales de humo, etc.

- Sistemas de manufactura : fabricas, lineas de ensamblado, etc.
- Sistemas financieros: contabilidad, inventarios, libro mayor, etc.

I.2.4. Sistemas automatizados.

Un sistema automatizado es un sistema hecho por el hombre que interactua con o son controlados por una o mas computadoras. Aunque hay diferentes tipos de sistemas automatizados, todos tienden a tener componentes en común.

Con esta definición surge la siguiente pregunta : Porque no debieran automatizarse algunos sistemas de procesamiento de información ?. Puede haber muchas razones; he aquí algunas de la mas comunes:

- Costo: Puede ser más barato continuar llevando a cabo las funciones y almacenando la información del sistemas en forma manual. No siempre es cierto que las computadoras sean mas rápidas y económicas que el método "anticuado".

- Conveniencia: Un sistema automatizado puede ocupar demasiado espacio, hacer demasiado ruido, generar demasiado calor o consumir demasiada electricidad

- Seguridad: Si el sistema de información mantiene datos confidenciales, el usuario pudiera no creer que el sistema automatizado sea lo suficientemente seguro; tal vez desee mantener la información físicamente protegida y bajo llave.

- Facilidad de mantenimiento: el usuario pudiera argumentar que un sistema de información computarizada seria costeable, excepto por el hecho de que no hay ningún miembro del personal que pudiera encargarse del mantenimiento del hardware y/o el software de la computadora.

- El hardware de la computadora: los procesadores, los discos, terminales, impresoras, unidades de cinta magnética , etc.

- El software de la computadora: los programas de sistemas tales como sistemas operativos, sistemas de bases de datos, programas de control de telecomunicaciones, etc.

- Las personas: los que operan al sistema, los que proveen su material de entrada y consumen su material de salida, y los que proveen actividades de procesamiento manual en un sistema.

- Los datos: las información que el sistema recuerda durante un periodo.

- Los procedimientos: la políticas formales e instrucciones de operación del sistema.

Una división en categorias de los sistemas automatizados es la siguiente:

- Sistemas en linea
- Sistemas de tiempo real
- Sistemas de apoyo a decisiones
- Sistemas basados en el conocimiento

A continuación se examinaran brevemente cada uno de ellos.

Sistemas en línea.

Un sistema en linea es aquel que acepta material de entrada directamente del área donde se creo. También es el sistema en el que el material de salida, o el resultado de la computación, se devuelve directamente a donde es requerido. Una caracteristica común de los sistemas en linea es que entran datos a la computadora o se les recibe de ella en forma remota.

Es decir, los usuarios del sistema computacional normalmente interactúan con la computadora desde terminales que pueden estar localizadas a cientos de kilómetros de la computadora misma.

Sistemas de Tiempo Real.

Un sistema computacional de tiempo real puede definirse como aquel que controla un ambiente recibiendo datos, procesándolos y devolviéndolos con la suficiente rapidez como para influir en dicho ambiente en ese momento.

Sistemas de apoyo a decisiones.

Como lo implica el termino, estos sistemas computacionales no toman decisiones por si mismos, sino ayudan a los administradores y a otros profesionistas "trabajadores del conocimiento" de una organización a tomar decisiones inteligentes y documentadas acerca de los diversos aspectos de la operación.

Sistemas basados en el conocimiento.

Un termino relativamente nuevo en la industria de las computadoras es el de "sistemas expertos" o "sistemas basados en el conocimiento". Dichos sistemas se asocian con el campo de la inteligencia artificial.
Un sistema experto en un programa de computadora que contiene el conocimiento y la capacidad necesarios para desempeñarse en un nivel de experto. El desempeño experto significa, por ejemplo, el nivel de desempeño de médicos que llevan a cabo diagnósticos y procesos terapéuticos.

El sistema experto es un apoyo de alto nivel intelectual para el experto humano, lo cual explica su otro nombre "asistente inteligente"

I.3. MARCO HISTORICO.

Durante las primeras tres décadas de la era de la computación, los cambios primarios fueron para desarrollar hardware para la computadora que redujera el costo del almacenamiento y procesamiento de datos.

Durante la década de los 80's, los avances en microelectrónica han dado como resultado en más poder de procesamiento de datos y haciendo los costos más bajos.

Hoy el problema es diferente. El cambio principal es reducir el costo e incrementar la calidad de las soluciones basadas en la computadora, soluciones que son implementadas con software.

Todo el poder de antaño de los mainframes es disponible ahora en un solo circuito integrado. El procesamiento increíble y las capacidades de almacenamiento del hardware moderno representan un gran potencial de computo. El software es el mecanismo que nos permite aprovechar y explotar este potencial.

El contexto en el cual el software ha sido desarrollado lo podemos resumir en cuatro décadas de evolución de los sistemas de computadora. La mejor ejecución de hardware, tamaño pequeño, y bajo costo han ocasionado sistemas mas sofisticados basados en la computadora.

Hemos dejado los procesadores con tubos al vacío para dispositivos microelectrónicos. La figura 1.1 describe la evolución del software dentro del contexto de las áreas de aplicación de los sistemas basados en computadora. Durante los primeros años del desarrollo de los sistemas de computadora, el hardware paso por cambios continuos mientras el software fue visto por muchos como una aplicación para el futuro . Los programas de computadora fueron una técnica rara para los cuales existian pocos métodos sistemáticos. El desarrollo de software fue virtualmente inmanejable a menos que se calendarizaran o los costos comenzaran a escalar.

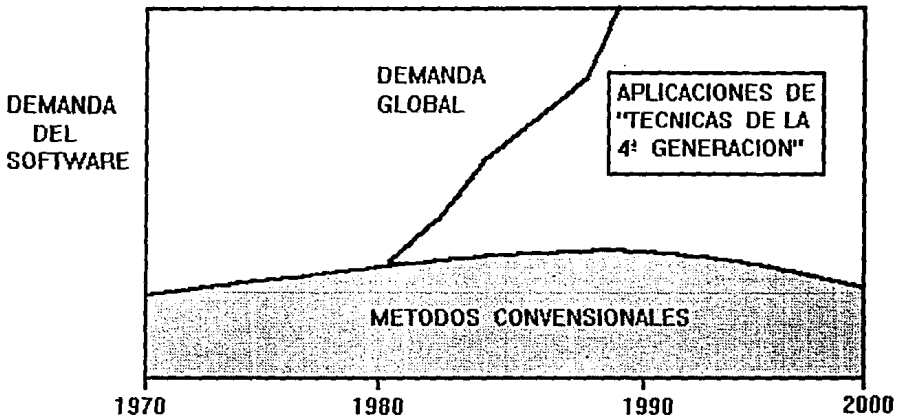


Fig. 1.1 La Evolución del Software.

Durante este periodo una orientación en lotes fue utilizada para más sistemas. Excepciones notables fueron los sistemas interactivos tales como el sistema de reservaciones anticipadas de la American Airlines y los sistemas de tiempo real orientado a la defensa tal como SAGE, sin embargo, el hardware fue dedicados para la ejecución de un solo programa que en su momento fue utilizado para una aplicación específica.

Durante los primeros años el hardware de propósito general llego a ser común y corriente. El software, por el contrario, fue diseñado para cada aplicación y tuvo una relativa distribución limitada.

Los productos de software (por ejemplo programas desarrollados para ser vendidos a uno o más clientes) estaban comenzado a aparecer. Mucho más software fué desarrollado y usado finalmente por la misma persona u organización. La misma persona que lo escribió, podía ejecutarlo, y si este fallaba, él mismo lo modificaba. Porque la movilidad del trabajo fue baja, los gerentes podían descansar seguros si es que esta persona estaba ahí cuando algunos errores fueran encontrados.

Así que en este medio ambiente de software personalizado, el diseño fué un proceso implícito ejecutado en una sola persona y la documentación fué frecuentemente inexistentes.

A través de los primeros años aprendimos mucho sobre la implementación de sistemas basados en la computadora, pero relativamente poco sobre los sistemas de computadora para la ingeniería. Sin embargo, debemos reconocer que muchos sistemas basados en computadora destacaron y fueron desarrollados durante esta era. Algunos de esos restos se usan actualmente y dan a sus compañías una admiración justificada.

La segunda era en la evolución de los sistemas de computadora se extendió en la década de mediados de los 60's y finales de los 70's. Multiprogramando, sistemas multiusuarios introduciendo nuevos conceptos de interacción hombre-máquina. Las técnicas interactivas abrieron un nuevo mundo de aplicaciones y nuevos niveles de sofisticación del hardware y del software. Los sistemas en tiempo real pueden colectar, analizar, y transformar datos de múltiples fuentes, con lo que controlando procesos y produciendo salidas en el rango de milisegundos. Avances en dispositivos de almacenamiento en línea llevaron a la primera generación de sistemas manejadores de bases de datos.

La segunda era fué también caracterizada por el uso de productos de software y la llegada de "casas de software". El software fué desarrollado para una extensa distribución en un mercado multidiciplinario. Programas para mainframes y computadoras fueron distribuidos a cientos y algunas veces miles de usuarios. Empresarios de la industria, gobierno, y academias se unieron para "desarrollar el último paquete de software" y poder ganar una gran cantidad de dinero.

Como el número de sistemas computacionales creció, las librerías de software para computadora se empezaron a expandir. En las casas desarrolladoras los proyectos produjeron decenas de miles de programas fuentes.

Los productos de software fueron comprados del exterior agregándoles cientos de miles de nuevas instrucciones.
Una sombra obscura apareció en esos momentos para todos los programas, todas esas instrucciones fuente, tuvieron que ser corregidas cuando algunas fallas fueron detectadas, modificandose para los requerimientos del usuario, o adaptando para el nuevo hardware que fuera adquirido.

Esas actividades fueron colectivamente llamadas mantenimiento del software. Muchos esfuerzos fueron realizados sobre el mantenimiento del software comenzando por absorber recursos en un rango alarmante.

Aún peor, la naturaleza personalizada de muchos programas se hicieron virtualmente inmantenibles. Una crisis de software había comenzado.

La tercera era de los sistemas de computación a mediados de los 70's y continua ahora. Los sistemas distribuidos (computadoras múltiples), cada vez ejecutan funciones concurrentemente y comunicándose con alguna otra computadora; se ha incrementado grandemente la complejidad de los sistemas computacionales.

Redes de área local y globales, comunicaciones digitales con un ancho de banda muy grande, y el incremento en la demanda para el acceso de datos "instantáneo" demanda mas desarrolladores de software.

La tercera generación también se caracteriza por el avance y el extenso uso de los microprocesadores y computadoras personales. El microprocesador es una parte integral de una extensa gama de productos "inteligentes" incluyendo automóviles, hornos de microondas, robots industriales, y equipos para análisis clínicos.

En muchos casos la tecnología del software ha sido integrada dentro de productos por un equipo técnico el cual entiende el hardware pero frecuentemente son principiantes en el desarrollo de software.

Las computadoras personales han sido el producto por excelencia para el crecimiento de muchas compañías de software. Mientras las compañías de software de la segunda era vendían cientos o miles de copias de sus programas, las compañías de la tercera generación venden decenas o centenas de miles de copias. En las computadoras personales el hardware es diferente. En efecto, el rango de ventas de las computadoras personales ha crecido muy poco desde mediados de los 70's, las ventas de los productos de software continúan hacia arriba.

La cuarta era de la computación está comenzando. Autores como Feigenbaum y McCorduck predicen que la "quinta generación" de computadoras y su respectivo software tendrán un gran impacto sobre el balance de la política y la industria en todo el mundo. Ya existen las técnicas de Cuarta Generación (4GT) para el desarrollo de software que están cambiando la manera en la cual algunos segmentos de la comunidad del software desarrollan sus sistemas computacionales. El software para los Sistemas Expertos y la Inteligencia Artificial se han finalmente desplazado de un laboratorio a una gran gama de problemas en el mundo real.

I.4. DEFINICION DE SISTEMA DE TIEMPO REAL.

Las formas más elaboradas y rápidas de procesamiento de datos son los sistemas de tiempo real. Estos sistemas no sólo proporcionan a los usuarios acceso inmediato a una amplia gama de archivos sino también permiten que varios usuarios tengan acceso a sus sistemas de cómputo. Los sistemas de tiempo real ofrecen acceso inmediato a la información y manejan todas las transacciones en el momento en el que ocurren.

Un sistema computacional de Tiempo Real puede definirse como aquel que controla un ambiente recibiendo datos, procesándolos y devolviéndolos con la suficiente rapidez como para influir en dicho ambiente en ese momento.

La expresión "con la suficiente rapidez" está, desde luego, sujeta a muchas interpretaciones. Ciertamente, existe muchos sistemas en línea (sistemas bancarios, de reservaciones aéreas y sistemas de bolsa) que se espera reaccionen en uno o dos segundos a un mensaje tecleado en la terminal.

I.5. CARACTERISTICAS DE LOS SISTEMAS DE TIEMPO REAL.

De las características de los sistemas de tiempo real podemos mencionar las siguientes:

- Los sistemas de tiempo real usualmente interactúan tanto con personas como un ambiente que casi siempre es autónomo.

- Los datos recibidos en estos sistemas como su nombre lo indica son los que se encuentra actualmente y de acuerdo un determinado tiempo según el software los actualiza

- Simultáneamente se leva a cabo el proceso de muchas actividades.

- Se le pueden asignar prioridades a los diferentes procesos, ya que algunos requieren servicio inmediato mientras que otros pueden aplazarse por periodos razonables.

- En estos sistemas se puede interrumpir alguna tarea para comenzar otra que tenga mayor prioridad.

- Se tiene una gran comunicación entre tareas, ya que muchas de estas tratan diferentes aspectos de un proceso general, como pudiera ser el control de un proceso de manufactura.

- Se tiene simultáneamente acceso a datos comunes, tanto en memoria como en el almacenamiento secundario, por lo cual es preciso elaborar procesos de sincronización y "semáforos" para asegurar que los datos comunes no se pierdan o corrompan.

- Se tiene también un uso y una asignación dinámica de memoria RAM en el sistema computacional, ya que puede resultar poco económico asignar suficiente memoria fija para manejar situaciones pico de alto volumen.

- El proceso se datos es instantáneo.

- Los sistemas de tiempo real son por lo regular bastante grandes puesto que dan apoyo a organizaciones que requieren proceso instantáneo de datos. Estos sistemas cuestan una gran cantidad de dinero y tienen las más altas velocidades.

I.5.1. Ventajas de los sistemas de tiempo real.

Debido a las características que presentan los sistemas de tiempo real presentadas en la Sección I.5, estos sistemas tienen un gran número de ventajas, las cuales se explicaran mediante dos ejemplos de aplicaciones de los sistemas de tiempo real.

Un sistema de reservaciones a tiempo real permite que la aerolínea consulte y actualice continuamente los archivos de reservaciones. Debido a que dichas reservaciones de los pasajeros son muchas y muy importantes para la planeación y para la atención de la aeronave, esos datos deben procesarse inmediatamente. Cualquier retraso en el manejo de las reservaciones podría causar molestias a los clientes o la mala asignación de los recursos de apoyo a la aeronave, con posibles repercusiones financieras. Una necesidad de computación con tiempo real similar se presenta en el sistema que da servicio a las bolsas de valores. Considerando el volumen de acciones comerciales y la velocidad con la cual estos datos deben reportarse, los sistemas con tiempo real son los únicos que pueden cumplir con la tarea. Imaginemos el caos que podría producirse debido a la incapacidad para reportar oportunamente los intercambios de acciones a toda la comunidad financiera.

I.6. EJEMPLOS DE SISTEMAS DE TIEMPO REAL.

Entre los ejemplos de sistemas de tiempo real que podemos mencionar son los siguientes:

Sistemas de control de procesos: los sistemas computacionales que se utilizan para verificar y controlar refinerías, algunos procesos químicos , operaciones de maquinado, etc.

Sistemas de cajeros automáticos: estos sistemas son más conocidos ya que su uso es muy común para todos nosotros en las diferentes instituciones bancarias para hacer depósitos y retiros sencillos en el banco.

Sistemas de alta velocidad para adquisición de datos: los sistemas computacionales que obtienen datos de telemetría a alta velocidad, satélites en órbita, las computadoras que capturan cantidades enormes de datos de experimentos de laboratorio, etc.

Sistemas de guía de proyectiles: los sistemas computacionales que deben rastrear la trayectoria de un proyectil y hacer ajustes continuos a la orientación y empuje de los propulsores.

Sistemas de conmutación telefónica: estos sistemas controlan la transmisión de voz y datos en miles de llamadas telefónicas, detectando los números marcados, condiciones de ocupado y todas las demás condiciones de una red telefónica típica.

Sistemas de vigilancia de pacientes: estos sistemas son muy importantes ya que en se están haciendo muy comunes en los diferentes hospitales , algunos de estos realizan actividades tales como: detectar los signos vitales de los diversos pacientes y otros más sofisticados son capaces de ajustar el medicamento administrado o de hacer sonar la alarma si los signos vitales se mantienen fuera de ciertos límites predeterminados.

I.7. ELEMENTOS FISICOS QUE INTERVIENEN EN UN SISTEMA DE TIEMPO REAL.

El equipo disponible para ejecutar procesamiento de datos en tiempo real son terminales, lineas de comunicación y dispositivos de almacenamiento auxiliares.

TERMINALES

Son los dispositivos en los cuales se hace posible la operación manual de la computadora. Las terminales son los dispositivos de entrada y salida del sistema. Cuando tales dispositivos son físicamente localizados en cualquier otro lugar diferente a donde se encuentra la consola (CPU), es llamado terminal. La finalidad de estas terminales es :

a) Facilitar la entrada de datos dentro del sistema, y

b) Recibir información de la computadora (CPU).

DISCOS

Los disquetes (o discos flexibles) asi como los discos duros almacenan archivos de datos, textos y programas. Los discos flexibles, se llaman asi porque están hechos de un material flexible, almacenan la información como si fueran secuencias de impulsos magnéticos en la superficie del disco. Esta información se puede escribir en otro disco, o leer del disco, mediante una cabeza magnética de lectura/escritura.
En la actualidad, los discos duros son un medio magnético indispensable en los sistemas de computo personales.

CINTAS

La cinta magnética es una tira o cinta de material plástico recubierto de una pelicula de material ferromagnetico (oxido de hierro), lo que permite que sea grabada en forma continua sobre toda una superficie. Para indicar la longitud utilizable de la cinta se le colocan al inicio y al final marcas reflectantes que pueden ser detectadas por la unidad de cinta.

IMPRESORAS

Una impresora es un dispositivo electrico-mecanico capaz de plasmar sobre un soporte físico las informaciones (textos o gráficos) creados en una computadora. Estos equipos están englobados dentro de la categoria informática de periféricos de salida, pues a través de ellos la computadora nos entrega la respuesta a nuestras demandas de información, generalmente sobre papel.

LINEAS DE COMUNICACIÓN

Todo el procesamiento en línea depende de la interacción directa del usuario con la computadora. Para facilitar la telecomunicación de datos debe existir algún tipo de línea de comunicación.

Para la conexión de computadoras y terminales ubicadas en diferentes lugares se tienen muchos tipos de servicios de comunicación, desde la línea telegráfica convencional hasta las formas mas complejas de transmisión, como son los satélites, pero el medio más común de comunicación de datos es la línea telefónica.

Los servicios de datos en forma de líneas telefónicas por lo general se dividen en las categorías: líneas rentadas (línea privada) y líneas conmutadas. Esos servicios los proporciona una compañía telefónica para apoyar el proceso en línea.

MICROPROCESADOR

Desde un punto de vista funcional, un microprocesador es un circuito integrado LSI que contiene la Unidad Central de Procesamiento (CPU) de un microcomputador o maquina programable.

La unidad central de proceso interpreta y ejecuta las instrucciones que comprende un programa de un computadora. Una CPU puede ser un microprocesador de una sóla pastilla, un conjunto de pastillas o una caja de tarjetas llenas de transistores, pastillas, cables y conectores. Las diferencias en las CPU's distinguen los grandes de los minis y de los microcomputadores.

Como tal, el microprocesador es el encargado de suministrar las señales de control para los demás elementos del sistema, buscar y traer instrucciones y datos desde la memoria, transferir datos hacia y desde dispositivos de entrada/salida, decodificar instrucciones, realizar operaciones aritméticas, etc.

MEMORIA RAM

Memoria de Acceso Aleatorio. Aunque, en efecto, a este tipo de memoria se puede acceder de manera aleatoria, un mejor nominativo seria Memoria para Leer y Escribir, puesto que el usuario puede leer y escribir sobre cualquier locación de la memoria RAM.
Las unidades RAM son algo así como los pizarrones del mundo de la informática. En ellas lo programas y los datos se almacenan con carácter provisional, mientras la computadora esta en funcionamiento, y en ellas también se escriben temporalmente los resultados y otros datos.

SISTEMA OPERATIVO

Escoger un sistema operativo para tiempo real para una aplicación especifica no es fácil. Hay que considerar que algunos coinciden en capacidades, objetivos de los sistemas y otras características.

Todos los sistemas operativos deberán tener un mecanismo de prioridades, pero un Sistema Operativo de Tiempo Real (RTOS) deberá dar un mecanismo de prioridades que permita interrupciones de alta prioridad para tomar y reconocer aquellas con mayor importancia que otras.

Además, porque las interrupciones ocurren en respuesta a eventos asíncronos, no recurrentes, estos deberán ser atendidos sin importar que proceso se este realizando. Por consiguiente, para garantizar la respuesta en el tiempo, un sistema operativo de tiempo real deberá tener un mecanismo para bloquear memoria, esto es, mantener en ella algunos programas en memoria principal para que el sobrecargo de transferencia sea evitado.

El contexto conmutación en el tiempo y espera por una interrupción determinaran la capacidad para el manejo de interrupciones, el aspecto más importante de un sistemas de tiempo real. Conmutación en el tiempo es el tiempo que el sistema operativo toma para almacenar el estado de la computadora y el contenido de los registros de manera que sea posible regresar a continuar la tarea de proceso después de haber sido atendida la interrupción.

La espera por una interrupción, es el tiempo máximo de espera antes que el sistema cambie o conmute una tarea, esto ocurre porque en un sistema operativo hay frecuentemente rutas no reentrantes o rutas de proceso criticas que deberán ser completadas antes que una interrupción pueda ser procesada.

La longitud de esas rutas (el número de instrucciones requeridas antes de que el sistema puede atender una interrupción) indicará el peor caso de tiempo de intervalo o de espera. El peor caso usaría una interrupción de alta prioridad es generada inmediatamente después de que el sistema entra en una ruta critica entre una interrupción y el servicio de interrupciones.

Muchos sistemas operativos - ejecutan multitarea, o procesos concurrentes, otro de las mayores necesidades para los sistemas de tiempo real.

BUFFER DE ENTRADA/SALIDA

Para obtener una eficiente entrada/salida asincrona, es necesario algún tipo de buffer. Algunas computadoras puede usar una memoria principal de núcleos magnéticos para realizar la función de un buffer.

Una entrada por un periférico llegara a la memoria al mismo tiempo que esta procesando la computadora y viceversa para la salida, por este motivo son de gran utilidad los buffers.
Las lineas de telecomunicación tienen una particular necesidad de los buffers. Los mensajes pueden llegar a la computadora y estos pueden ser independientes de los procesos; estos mensajes deberán ser puestos en algún lugar si la computadora no los puede procesar inmediatamente. Además, cada uno de los equipos ha comenzado a leer el mensaje, y los caracteres regresaran en un regular o irregular intervalo hasta que el mensaje sea completado. En algún lugar los caracteres deberán ser almacenados para formar el mensaje completo.

Los buffers para telecomunicación están alojados para muchos sistemas en la memoria principal de la computadora, pero otros sistemas tienen buffers de telecomunicación separados. Algunos de ellos utilizan programas separados de almacenamiento de propósito general que manejen todas las líneas de telecomunicación. Estos pasaran por completo checando y editando los mensajes de entrada hacia la computadora principal y aceptara los mensajes de salida de esta.

MANEJO DE INTERRUPCIONES.

Una de las características que nos sirven para distinguir a los sistemas de tiempo real de cualquier otro tipo de sistemas es el manejo de interrupciones. Un STR debe responder a estímulos externos - interrupciones - en un tiempo fijado por el mundo externo. Porque múltiples estímulos (interrupciones) se presentan frecuentemente, prioridades y una de estas debe ser establecida para las interrupciones deben de ser establecidas. En otras palabras la tarea de mayor importancia deberá siempre de ser atendida dentro de un tiempo establecido pase lo que pase con otros eventos.

El manejo de interrupciones implica no solamente el almacenamiento de información sino también que la computadora pueda correctamente restablecerce a la tarea de interrupción, pero también debe evitar interbloqueos y ciclos sin fin.

Una forma de englobar al manejo de interrupciones de presenta en la figura 1.2. Cuando el flujo de un proceso normal es "interrumpido" por un evento que es detectado por el procesador. Un evento es cualquier ocurrencia que requiera una atención inmediata y puede ser generada por hardware o

software. El estado del programa interrumpido es salvado, y el control es transferido a la rutina encargada de atender a las interrupciones que se dirigirá a la rama adecuada de software para el manejo de esa interrupción. Una vez terminada la atención a la interrupción, el estado de la máquina es restablecido y el flujo del proceso normal es restablecido.

En muchas situaciones, el manejador de interrupciones para un evento puede ser interrumpido por el mismo por otro evento de mayor prioridad. Los niveles de prioridad para interrupciones pueden ser establecidos.

Un posible esquema involucraría un sistema de buffers para los datos de modo que estos puedan ser procesados rápidamente cuando el sistema lo requiera.

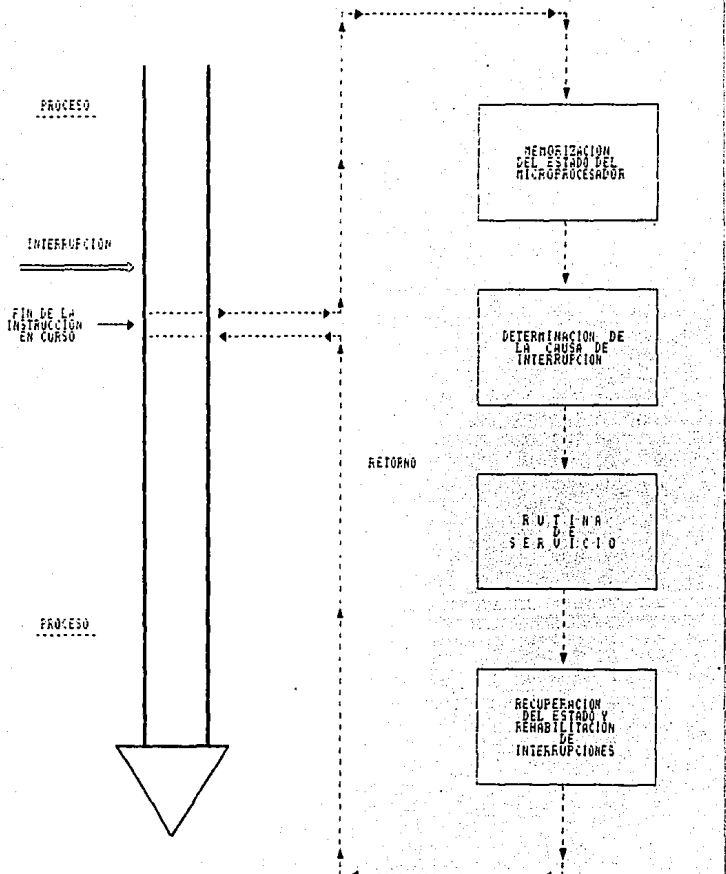


Fig. 1.2 Diagrama de servicio para una interrupción.

BASES DE DATOS EN TIEMPO REAL.

Como muchos de los sistemas que procesan datos, los STR frecuentemente son acoplados con una función para administrar la base de datos. Sin embargo, las bases de datos distribuidas deberán ser utilizadas de preferencia en los sistemas de tiempo real porque la multitarea es ordinaria y los datos son frecuentemente procesados en paralelo. Si la base de datos es distribuida y no centralizada, las tareas individuales pueden accesar a sus datos mas rápidamente, mas confiable y con pocos "cuellos de botella". La utilización de bases de datos distribuidas para aplicaciones en tiempo real divide el "tráfico" de E/S en colas pequeñas de tareas esperando para accesar la base de datos. Además, una falla de una base de datos raramente causará la falla total del sistema si una redundancia es realizada.
La redundancia en los datos improvisara repuestas en el tiempo debido a múltiples fuentes de información.

LENGUAJE DE PROGRAMACIÓN EN TIEMPO REAL.

Debido a los requerimientos especiales para el rendimiento y seguridad para los sistemas de tiempo real, elegir un lenguaje de programación es muy importante. Muchos lenguajes de programación de propósito general (ej. C, PASCAL, MODULA) pueden ser utilizados efectivamente para aplicaciones en tiempo real.
Una combinación de características hace a un lenguaje de tiempo real sea diferente de un lenguaje de propósito general. Esas incluyen la capacidad de multitarea, instrucciones para implementar directamente funciones de tiempo real, y características modernas de programación que ayuden a garantizar exactitud en los programas.

Un lenguaje de programación que soporta directamente multitarea es importante porque un STR deberá responder a eventos asincronos que ocurran simultáneamente.

Finalmente, características que faciliten una programación fiable es necesario porque los programas para tiempo real son frecuentemente largos y complejos. Esas características incluyen programación modular, un poderoso manejo para diferentes tipos de datos.

I.8. EL GAS OZONO.

Diez años después de que Priestley descubriera el oxígeno, se dio a conocer otra forma de oxígeno gaseoso. A diferencia de la forma común y estable del oxígeno, el nuevo gas poseía un olor característico. Pero sólo en 1840 pudo el químico alemán, Chistian F Schönbein aislar este gas, dándole el nombre de ozono, (del vocablo griego que significa "oler"). Su olor picante se puede percibir alrededor de las máquinas movidas eléctricamente que despiden chispas cuando están funcionando.
El ozono es un gas azul pálido, más pesado y mucho más soluble en agua que el oxígeno, y, químicamente, más activo.

Químicamente, el ozono es inestable; espontáneamente se convierte en oxígeno, y dos volúmenes de ozono producen tres de oxígeno.
El ozono se encuentra en mayores concentraciones en los niveles superiores de la estratosfera. Un acontecimiento importante ocurre en esta parte de la atmósfera, y es la conversión del ozono en oxígeno, esto libera energía calorífica. En la región de la atmósfera comprendida entre los 45 y 55 kilómetros sobre la superficie terrestre, la temperatura es unos 70 Grados Centígrados, más alta que a 10 kilómetros.

A causa de su potente actividad química, el ozono se ha utilizado como purificador; matab-acterias y otro microorganismos que están en el agua, por reaccionar con sus componentes químicos. En fuertes concentraciones es tóxico para el hombre.

I.9. PASOS PARA REALIZAR UN SISTEMA DE TIEMPO REAL.

Los pasos necesarios para realizar un sistema en tiempo real se describen a continuación:

1. Definir el problema. La formulación debe incluir las preguntas que se van a contestar, las hipótesis que se van a probar, los cálculos que se van a hacer, etc.

2. Construir un modelo matemático. El modelo debe abstraer las características esenciales del sistema y hacer posible la predicción de estados futuros del sistema basados en varios estados iniciales.

3. Validación. Si el modelo no representa al sistema lo suficientemente bien para los propósitos de estudio.

4. Construir el programa. Escribir un programa del modelo que permita introducir datos, fijar condiciones iniciales, corregir errores y generar soluciones.

5. Verificación. Si el programa no representa correctamente el modelo, regresar al paso 4.

6. Planear los experimentos de prueba. Formular un plan cuidadoso de los experimentos que se van a efectuar con el modelo para entregar la información requerida en el paso 1.El plan debe incluir los datos y condiciones iniciales que se van a usar y el tipo de resultados deseados.

7. Efectuar las corridas de prueba. Ejecutar el programa de acuerdo con el paso anterior para producir los resultados.

8. Analizar los resultados. Si los resultados obtenidos son los esperados, terminar. En caso contrario, regresar a los puntos 2 y 4 según convenga.

El desarrollo de un Sistema en tiempo real requiere un análisis detallado del problema así como la realización de una serie de actividades bien definidas como se menciona en este capítulo. No obstante, los beneficios que presentan los sistemas en tiempo real justifican ampliamente el costo y tiempo de su realización.

CAPITULO II.

SISTEMAS DE TIEMPO REAL.

II.1. INTRODUCCION.

En el diseño de un sistema de tiempo real se deben de considerar varios puntos de los cuales podemos mencionar lo especializado que va estar nuestro sistema, ya que sabiendo esto nos mostrará el grado de dificultad que vamos encontrar para llevar a cabo nuestro sistema.

Es muy importante tener en cuenta las necesidades que va a cubrir nuestro sistema y cuidar que no se nos escape ninguna ya que de esto dependerá el buen funcionamiento del mismo, así como buena utilización.

En este capítulo se expondrán las principales características de los sistemas de tiempo real de forma que se puedan establecer algunos criterios básicos de diseño para luego determinar una relación de requerimientos del lenguaje.

Antes de discutir las características de los sistemas de tiempo real, nos resultará muy útil definir qué se conoce por el término "tiempo real", especialmente a la vista del hecho de que se viene utilizando en una amplia gama de diferentes contextos con significados diferentes.

En su sentido más amplio, el término "tiempo real" puede utilizarse para describir cualquier actividad de proceso de información que deba responder a estímulos de entrada generados externamente dentro de un intervalo (delay) especificable, finito y breve.

II.2. MODELADO DE UN SISTEMA DE TIEMPO REAL.

El termino "modelo" puede parecer algo formal y atemorizante, pero representa un concepto que hemos manejado toda la vida y se pueden considerar como ejemplos de modelos los siguientes:

- Mapas
- Globos terráqueos
- Diagramas de flujo
- Dibujos arquitectónicos
- Partituras musicales

¿Por que construimos modelos?. ¿Por que no se construye simplemente el sistema mismo?. La respuesta es que podemos construir modelos de manera tal que enfatizamos ciertas propiedades criticas del sistema mientras que simultáneamente desatendamos otros de sus aspectos.

Al realizar un modelo del sistema nos podemos dar cuenta que nuestra comprensión de los requerimientos del usuario o usuarios no fué la correcta (o de que el cambio de parecer acerca de sus requerimientos), podemos hacer cambios en el modelo o desecharlo y hacer uno nuevo, de ser necesario.

El modelado del comportamiento dependiente del tiempo.

El comportamiento dependiente del tiempo, es decir, la secuencia con la cual se hará el acceso a los datos y se ejecutaran las funciones es un tercer aspecto de muchos sistemas complejos. Para algunos sistemas computacionales de empresas este aspecto no es importante, puesto que la secuencia es esencialmente trivial.

Asi, en muchos sistemas computacionales (aquellos que ni son de tiempo real, ni están en línea), la función N no puede llevar a cabo su labor hasta que reciba la entrada que requiere; y esta entrada se produce como salida de una función N-1, y así sucesivamente.

Sin embargo, muchos sistemas en línea y de tiempo real, tanto en el campo de los negocios como en el de la ciencia y la ingeniería, tienen complejas relaciones en el tiempo que deben modelarse tan cuidadosamente como las funciones y las relaciones entre datos.

Por ejemplo, muchos sistemas de tiempo real deben responder dentro de un margen breve, posiblemente de tan sólo unos microsegundos, a ciertas entradas provenientes del ambiente exterior. Y deben estar preparadas para recibir diversas combinaciones y secuencias de entradas a las cuales se debe responder adecuadamente.

Para el modelado de un sistema de tiempo real se debe considerar los siguientes puntos:

1.- Si nuestro sistema se va haciendo cada vez más especializado, menos capaz será de adaptarse a circunstancias diferentes.

2.- Entre más general sea nuestro sistema, menos óptimo será para una situación determinada; pero entre más óptimo sea, para tal situación menos adaptable será a nuevas circunstancias.

3.- Cuanto mayor sea nuestro sistema mayor es el número de sus recursos que deben dedicarse a su mantenimiento diario.

4.- Debe cumplir con los objetivos planeados o necesidades requeridas por la empresa o por el usuario ya que esto nos indicará si nuestro sistema cumplió o no .

5.- Un punto general que se maneja en cualquier tipo de sistemas es que, los sistemas siempre forman parte de sistemas mayores y siempre pueden dividirse en sistemas menores. Este punto es importante por que sugiere una forma obvia de organizar un sistema computacional que estemos tratando de desarrollar, por el procedimiento de dividirlo en sistemas menores.

6.- Debemos de tener en cuenta cuanto puede crecer nuestro sistema y definirlo cuidadosamente para que todo mundo conozca su contenido es una actividad muy importante, ya que como analistas o diseñadores de sistemas, debemos de tomar en cuenta este punto al comenzar a crear nuestro sistema.

EL DIAGRAMA DE FLUJO DE DATOS (DFD).

Un viejo adagio de la profesión de desarrollo de sistemas dice que un sistema de proceso de datos involucra tanto los datos como el proceso, y no se puede construir un sistema exitoso sin considerar ambos componentes. El aspecto de proceso de un sistema ciertamente es algo importante de modelar y de verificar con el usuario. El modelado que llevamos a cabo puede describiese en una variedad de maneras:

- ¿Con que funciones debe desempeñarse el sistema?
- ¿Cuales son las interacciones entre dichas funciones ?
- ¿Que transformaciones debe llevar a cabo el sistema ?
- ¿Que entradas se transforman en que salidas ?
- ¿Que tipo de labor debe realizar el sistema ?
- ¿De donde obtiene la información para llevar a cabo dicha labor?
- ¿Donde entrega los resultados de su labor?

La herramienta de modelado que utilizamos para describir la transformación de entradas a salidas es un diagrama de flujo de datos. En la figura 2.1 se muestra un diagrama de flujo de datos típico.
Los diagramas de flujo de datos consisten en procesos, agregados de datos, flujos y terminadores:

Los procesos se representan por medio de círculos, o "burbujas", en el diagrama. Representan las diversas funciones individuales que el sistema lleva a cabo. Las funciones transforman entradas en salidas.

Los flujos se muestran por medio de flechas rectas. Son las conexiones entre los procesos (funciones del sistema) y representan la información que dichos procesos requieren como entrada o la información que generan como salida.

Los agregados de datos se representan por medio de dos líneas paralelas o mediante una elipse. Muestran colecciones (o agregados) de datos que el sistema debe recordar por un periodo de tiempo. Cuando los diseñadores de sistemas y los programadores terminan de construir el sistema, los agregados existirán como archivos o bases de datos.

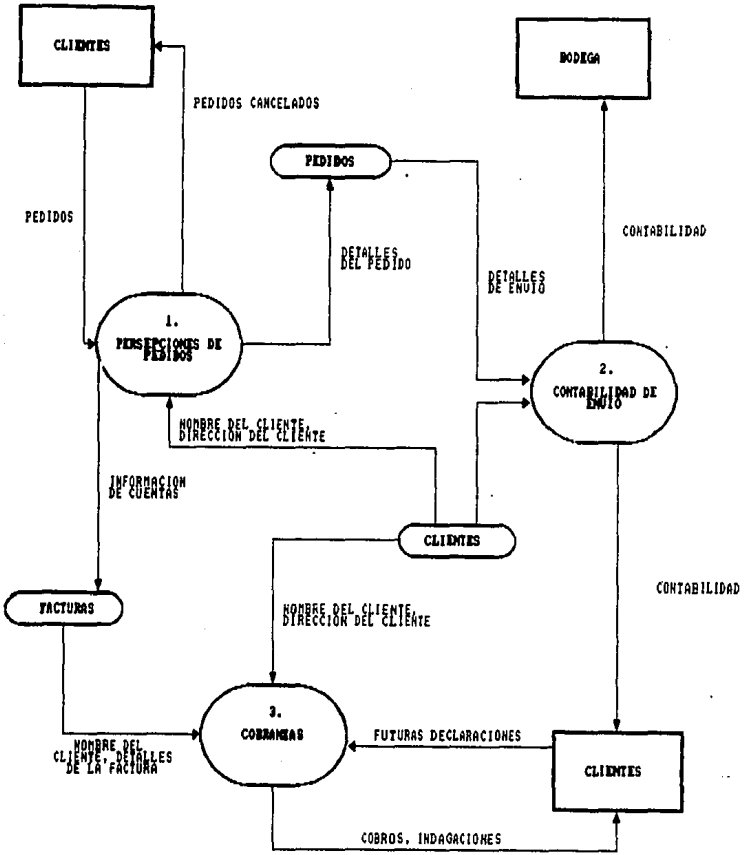


Fig. 2.1 Un Diagrama de Flujo de Datos Tipico.

Los terminadores muestran las entidades externas con las que el sistema se comunica. Típicamente se trata de individuos o grupos de personas (por ejemplo, otro departamento o división dentro de la organización), sistemas de computo externos y organizaciones externas.

DIAGRAMA DE TRANSICION DE ESTADOS (DTE).

Hasta hace poco, los modelos de comportamiento dependiente del tiempo del sistema importaban sólo para una categoría especial de sistemas conocidos como sistemas de tiempo real. Como ejemplo de estos sistemas se tiene control de procesos, sistemas de conmutación telefónica, sistemas de captura de datos de alta velocidad y sistemas de control y mando militares. Algunos de estos sistemas son pasivos, en el sentido de que no buscan controlar el ambiente que los rodea, sino mas bien reaccionan a él capturan datos que le atañen. Muchos sistemas de alta velocidad de captura de datos entran en esta categoría (por ejemplo, un sistema que captura datos científicos de un satélite a alta velocidad).

Otros sistemas de tiempo real son mas activos, en el sentido que pretenden mantener el control sobre algún aspecto del ambiente que los rodea. Los sistemas de control de procesos y una variedad de sistemas interconstruidos en otros entran en esta categoría.

Los sistemas de este tipo manejan fuentes externas de datos de alta velocidad, y deben proporcionar alguna respuesta y datos de salida de manera suficientemente rápida como para manejar el ambiente externo. Una parte importante de la especificación de tales sistemas es la descripción de que sucede y cuando.

Para los sistemas enfocados a los negocios, esto normalmente no ha sido tan importante. Las entradas pueden llegar al sistema de diferentes fuentes a velocidades relativamente altas, pero habitualmente se pueden detener si el sistema esta ocupado haciendo otra cosa. Por ejemplo, un sistema de nomina no tiene que preocuparse por interrupciones y señales de unidades de radar externas. generalmente los únicos asuntos que involucran tiempos en este tipo de sistemas son especifica mente de tiempo de respuesta.

En las figuras 2.2 y 2.3 se muestra un ejemplo muy común de un diagrama de transición de estados.

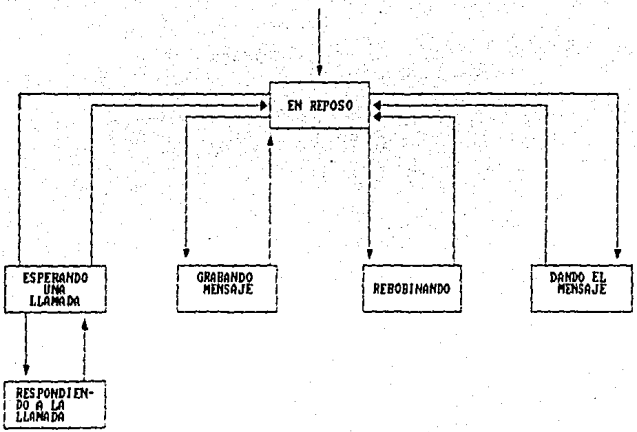


Fig. 2.2 Diagrama típico de Transición de Estados.

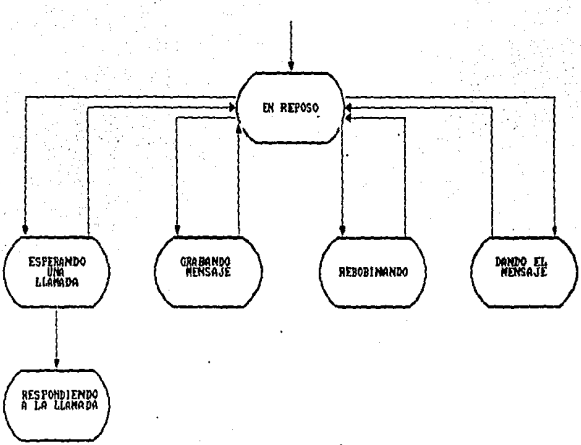


Fig. 2.3 Diagrama Alternativo de Transicion de Estados.

ESTADOS DEL SISTEMA.

Estado :

Un conjunto de circunstancias o atributos que caracterizan a una persona o cosa en un tiempo dado; forma de ser, condición.

Por lo tanto, los estados típicos de un sistema pueden ser:

- Esperar a que el usuario de su contraseña
- Calentar una mezcla de sustancias químicas
- Esperar la siguiente orden
- Acelerar el motor
- Mezclar los ingredientes
- Esperar los datos del instrumento
- Llenar el tanque
- Aguardar en reposo

Nótese que muchos de estos ejemplos implican que el sistema esta esperando a que algo ocurra, y no se expresan en términos de que la computadora este haciendo algo. Esto se debe a que el diagrama de transición de estados se usa para desarrollar un modelo esencial del sistema, es decir, un modelo de como se comportaria el sistema si hubiera tecnología perfecta.

Así, cualquier estado observable en el que el sistema se pueda encontrar solo puede corresponder a periodos en los que:

1) está esperando que algo ocurra en el ambiente externo o 2) está esperando a que alguna actividad que se esta dando en ese momento en el ambiente (como mezclar, lavar, llenar, acelerar, etc.) cambie a otra.

Esto no significa que nuestro sistemas sean incapaces de ejecutar acciones o que no pretendamos mostrarlas, sino sólo que las acciones, que ocurren instantáneamente en nuestro modelo de tecnología perfecta, no son lo mismo que los estados, que representan condiciones observables en las que el sistema se puede encontrar. Esto es, un estado representa algún comportamiento del sistema que es observable y que perdura durante algún periodo finito.

CAMBIOS DE ESTADO.

Un sistema que existió en un sólo estado no sería un objeto de estudio muy interesante: sería estático. De hecho, los sistemas de información que modelamos normalmente pueden tener docenas de estados diferentes. Pero ¿Como cambia un sistema de un estado a otro?. Si tiene reglas ordenadas que gobiernan su comportamiento, entonces generalmente sólo algunos tipos de cambio de estado serán significativos y validos.

Se muestran los cambios de estado validos en el DTE conectando pares relevantes de estado con una flecha. Así la figura 2.4 muestra que el sistema puede ir del estado 1 al estado 2. También muestra que cuando el sistema se encuentre en el estado 2 puede ir al estado 3 o regresar al 1. Sin embrago, de acuerdo con este DTE, el sistema no puede ir directamente del estado 3 de regreso al 1. Nótese que el estado 2 tiene dos estados sucesores. Esto es muy común en los DTE; de hecho, cualquier estado puede llevar a un numero arbitrario de estados sucesores.

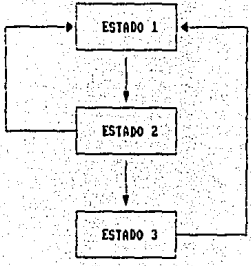


Fig. 2.4 Cambios de Estado.

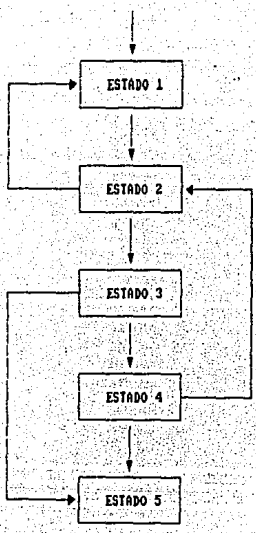


Fig. 2.5 Estados Inicial y Final.

A pesar de que la figura 2.4 proporciona información interesante acerca del comportamiento dependiente del tiempo de un sistema, no dice algo que pudiera resultar ser muy importante: cuales son los estados inicial y final del sistema, de hecho, la figura 2.4 es un modelo de estado estable de un sistema que ha estado siempre activo y que continuara siempre estándolo. La mayoría de los sistemas tienen un estado inicial reconocible y un estado final reconocible; esto se muestra en la figura 2.5.

El estado inicial tipicamente suele ser el que se dibuja en la parte superior del diagrama, aunque no es obligatorio. Lo que realmente identifica al estado 1 en la figura 2.4 como inicial es la flecha "desnuda" que no esta conectada a ningún otro estado, de manera similar, el estado final suele ser el que se dibuja en la parte inferior, pero tampoco esto es obligatorio. Lo que realmente identifica al estado 5 como final es la ausencia de una flecha que salga de el. en otras palabras, una vez que se llega al estado 5 ya no se puede ir a ninguna otra parte.

El sentido común dice que un sistema sólo puede tener un estado inicial; sin embrago, puede tener múltiples estados finales. Los diversos estados finales son mutuamente excluyentes, lo cual significa que sólo uno de ellos puede ocurrir durante alguna ejecución del sistema. La figura 2.6 muestra un ejemplo en el que los posibles estados finales son el 4 y el 6.

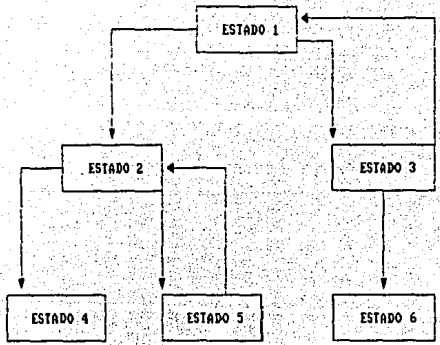


Fig. 2.6 Estados Finales Multiples de un Sistema.

CONDICIONES Y ACCIONES.

Para completar nuestro DTE necesitamos añadir dos cosas mas: las condiciones que causan un cambio de estado y las acciones que el sistema toma cuando cambia de estado. Como ilustra la figura 2.7, las condiciones y acciones se muestran junto a la flecha que conecta dos estados relacionados.

Una condición es un acontecimiento en el ambiente externo que el sistema es capaz de detectar; tipicamente es una señal, una interrupción o la llegada de un paquete de datos. Esto usualmente hace que el sistema cambie de un estado de esperar X a un estado de esperar Y; o de llevar a cabo la actividad X a llevar a cabo la actividad Y.

Como parte de un cambio de estado, normalmente el sistema hará una o más acciones; producirá una salida, desplegara una señal en la terminal de usuario, llevará a cabo un calculo, etc. Así que las acciones que se muestran en un DTE son respuestas regresadas al ambiente externo o bien cálculos cuyos resultados el sistema recuerda (tipicamente en un almacén de datos que se muestra en el DFD) para poder responder a algún acontecimiento futuro.

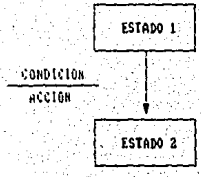


Fig. 2.7 Muestra de Condiciones y Acciones.

EXTENSIONES DEL DFD PARA SISTEMAS DE TIEMPO REAL.

Los flujos que se discuten a lo largo de este capitulo son flujos de datos, son simplemente los conductos a lo largo de los cuales viajan los paquetes de datos entre procesos y almacenes. Similarmente, las burbujas en los DFD que hemos visto hasta ahora pudieran considerarse como procesadores de datos. Para una amplia clase de sistemas, sobre todo de negocios, existen sólo dos tipos de flujos necesarios en el modelo de sistema; pero para otra clase de sistemas, los de tiempo real, necesitamos alguna manera de modelar flujos de control (es decir señales o interrupciones). Y se requiere una manera de mostrar procesos de control (esto es, burbujas cuya única labor es coordinar y sincronizar las actividades de otras burbujas del DFD).
Un flujo de control puede imaginarse como un conducto que porta una señal binaria (esto es, esta encendido o esta apagado). A diferencia de otros flujos, el flujo de control no porta datos con valores. El flujo de control se manda de un proceso a otro (o de algún terminador externo a un proceso) como una forma de decir: "Despierta, es hora de trabajar". Esto, desde luego, implica que el proceso ha estado "dormido" hasta la llegada del flujo de control.

Un proceso de control puede considerarse como una burbuja supervisora o ejecutiva, cuya labor es coordinar las actividades de otras burbujas en el diagrama; sus entradas y salidas consisten sólo de flujos de control. Los flujos de control salientes del proceso de control se utilizar para despertar a otras burbujas; los flujos de control entrantes generalmente indican que una de las burbujas ha terminado su labor o que se ha presentado alguna situación extraordinaria, de la cual necesita informarse a la burbuja de control. Por lo común sólo hay un proceso de control de estos en un DFD dado.

Como se indico anteriormente, un flujo de control se utiliza para despertar un proceso normal; una vez despierto, el proceso normal procede a llevar a cabo su labor como lo describe la especificación del proceso. Sin embargo, el comportamiento interno de un proceso de control es diferente; aquí es donde el comportamiento dependiente del tiempo del sistema se modela con detalle. El interior del proceso de control se modela con un diagrama de transición de estados que muestra los varios estados en los que se puede encontrar todo el sistema y las circunstancias que lo llevan a cambiar de estado.

En IMECA/STR el Diagrama de Flujo de Datos (DFD), el Diagrama de Transición de Estados (DTE), así como los Cambios de Estado, se aplican directamente en el algoritmo de dicho sistema (Secc. III.4.).
En el presente trabajo se dan las bases teóricas que pueden ser aplicadas para el diseño de cualquier Sistema de Tiempo Real, por lo que se presentan figuras ejemplificando a dichos diagramas y no directamente los que corresponden a IMECA/STR, ya que como se menciono, estos ya están incluidos en el algoritmo.

II.2.1. Caracteristicas del software de tiempo real.

a) El software debe ser fiable, con esto estaremos más seguros de su buen funcionamiento, ya que un fallo del sistema puede ser muy caro en términos de pérdida de producción en el caso de control de proceso o en términos de la propia vida humana en situaciones extremas, como la navegación aérea o como pudiera ser el control de alguna estación de energía eléctrica o nuclear.

b) Es casi siempre voluminoso y complejo. Esto combinado con el problema de la fiabilidad, significa que los costos de mantenimiento y desarrollo estos puede ocasionar que sean muy elevados.

c) Debe responder a una variedad de eventos externos con tiempos de respuesta garantizados, y con esto también tendrá que responder a una gran variedad de dispositivos de entrada y salida no estandares, así como de los periféricos convencionales.
Con independencia del tamaño y amplitud del lenguaje, existen algunas reglas básicas que deben observarse si se quiere conseguir un buen diseño.

d) El lenguaje debe tener una sintaxis clara y exenta de ambigüedades. Así mismo, cuantas menos reglas de sintaxis se necesiten para describir el lenguaje, será mucho mejor. Una sintaxis reducida facilita el aprendizaje del lenguaje, es más fácil de recordar y generalmente indica que el diseñador ha integrado cuidadosamente todas las características del lenguaje.

Se necesita un lenguaje de programación que permita evitar los accesos simultáneos al buffer o se producirá una corrupción de datos. La solución tradicional para asegurar un acceso mutuamente exclusivo a un recurso compartido, tal como un buffer común, es a través de la utilización de una especie de semáforo. Un semáforo es una variable binaria cuyo estado indica si esta disponible o no el recurso que protege. Para un semáforo dado s, si s=1 el recurso esta disponible, de otro modo no lo estará y un proceso que desee acceder a él deberá esperar.

A diferencia de los software científicos y comerciales, el de tiempo real tiene que funcionar, muy a menudo, ininterrumpidamente aún en presencia de condiciones de error. Por ejemplo, una computadora para control de procesos responsable de la operación de un horno de gas no puede permitir la parada del horno cuando se produce una situación de error. Por el contrario, debe intentar que la operación continúe de la mejor forma posible.

Los requerimientos de alta fiabilidad y de operación ininterrumpida en un sistema de tiempo real implican que un lenguaje de tiempo real debe facilitar mecanismos que permitan responder a condiciones de error y a programar acciones de recuperación. Existen cinco requerimientos básicos de un mecanismo de tratamiento de errores: simplicidad, claridad, evitar sobrecargas en tiempo de ejecución a menos que se le invocara, tratamiento uniforme de los errores detectados en el programa y en la implantación, y la capacidad para programar acciones de recuperación.

La posibilidad de compilar por separado los componentes individuales de un sistema software es esencial para un lenguaje de tiempo real practico. La compilación independiente permite compilar grandes programas sobre máquinas que disponen de unos requerimientos de memoria modestos, evita volver a compilar un programa entero cuando se efectúa una única modificación y ayuda a la dirección del proyecto, ya que los componentes de software pueden ser construidos y probados de forma aislada.

La especificación de las facilidades estandares para programar las operaciones de E/S sobre dispositivos periféricos estandar, tales como consolas, discos e impresoras, es especialmente difícil en el área del tiempo real. En muchos lenguajes o se ignora totalmente, de forma que los responsables de la implantación deben confeccionar su propio conjunto de facilidades, o se facilita por medio de características ya construidas que no son expresables en el propio lenguaje (ej. la sentencia write de PASCAL).

Una vez expuesto el perfil de un sistema típico ya podemos enumerar las características del software de tiempo real .En primer lugar, debe ser extremadamente fiable. Un fallo del sistema puede ser muy caro en términos de perdida de producción en el caso de un control de proceso o en términos de la propia vida humana en situaciones extremas, como la navegación aérea o el control de una central de energía nuclear. En segundo lugar es generalmente voluminoso y complejo. Esto, combinado con el problema de la fiabilidad, significa que los costos de mantenimiento y desarrollo pueden ser muy elevados. En tercer lugar debe responder a una variedad de eventos externos con tiempos de respuesta garantizados.

e) El software debe admitir la conexión con una gran variedad de dispositivos de E/S no estandares, así como de los periféricos convencionales de una computadora. Debe ser eficiente para poder hacer la mejor utilización posible del equipo hardware disponible.

El programador no debe verse sumido en la obscuridad debido a restricciones del lenguaje tales como una longitud limitada del identificador o unos tipos de datos inadecuados.

Las estructuras algoritmicas deberian estar expresadas con claridad en términos de una fácil comprensión de las estructuras de control tales como if ... then ... else ... etc. Las palabras clave no deberán abreviarse y los simbolos de los operadores se deben de escoger cuidadosamente para que reflejen su significado de forma que cualquier lector que no este totalmente familiarizado con el lenguaje pueda comprender la principales caracteristicas de un programa sin tener que consultar el manual de referencia del lenguaje.

Flexibilidad.

El lenguaje de programación debe proporcionar al programador la suficiente flexibilidad como para que pueda expresar todas las operaciones.

Eficiencia.

Los sistemas de tiempo real deben procurar frecuentemente un alto grado de flujo interno para satisfacer las restricciones impuestas por el equipo externo a monitorizar o controlar. Un lenguaje debe , por tanto, ser capaz de una implantación eficiente

Una de las caracteristicas peculiares de un software de tiempo real es el hecho de que, con mucha frecuencia se deben tratar dispositivos de E/S no estandares. Esto implica que un lenguaje debe aportar las facilidades necesarias para poder programar directamente dispositivos de bajo nivel.

Finalmente, para conseguir una mayor fiabilidad, el software de tiempo real debe ser capaz de una recuperación de las condiciones de error.

II.3. ESTRUCTURA BASICA DE UN SISTEMA DE TIEMPO REAL.

Modulo interno de entrada/salida.

La función principal del modulo interno de entrada es la de leer o consumir los datos producidos por cualquier acción de entrada, para ilustrar este modulo utilizaremos un ejemplo muy común de consumo de datos por parte del CPU como lo es la lectura de un disco duro o disco flexible.
El modelo resultante para la operación de entrada está representado en la figura 2.8.

Por otro lado la función principal del modulo interno de salida es la de consumir los datos producidos por el CPU o por cualquier otro dispositivo. Como ejemplo para ilustrar este modulo utilizaremos el conjunto de acciones que se producen al momento de imprimir. El modelo para representar la operación de salida se muestra en la figura 2.9.

Modelo abstracto.

La actividad realizada por un dispositivo de E/S puede considerarse como la de un procesador que ejecuta una tarea fija. Por lo tanto, un modelo que describa la programación de dispositivos de E/S debe facilitar una representación abstracta para uno o mas procesadores periféricos, trabajando en paralelo con el procesador central. Es claro que, a nivel lenguaje, el concepto de proceso, es la abstracción apropiada para representar este tipo de sistema.

ESTRUCTURA BASICA DE UN SISTEMA DE TIEMPO REAL

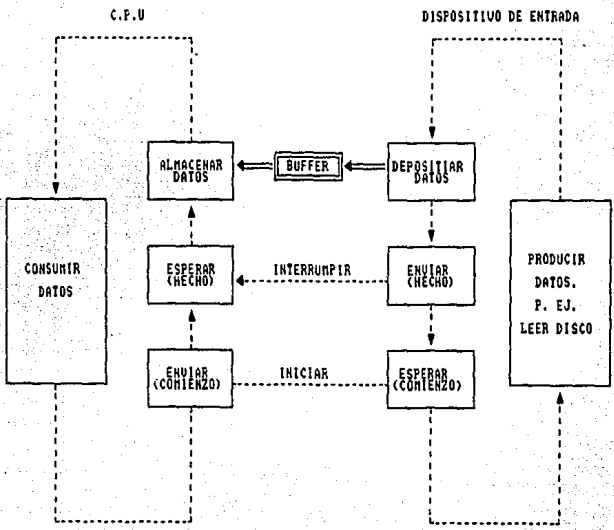


Fig. 2.8 Operacion de Entrada bajo Control de Interrupciones.

ESTRUCTURA BASICA DE UN SISTEMA DE TIEMPO REAL

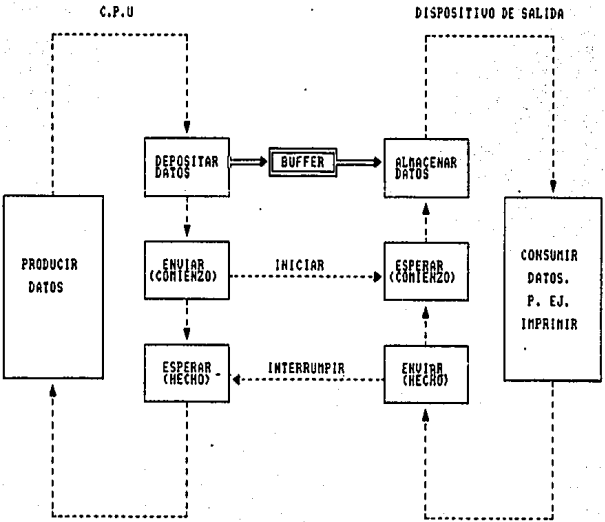


Fig. 2.9 Operacion de Salida bajo Control de Interrupciones.

Dado que la actividad del CPU esta representada por un proceso y la de un dispositivo de E/S también esta representada por un proceso, el acto de escribir o leer en o desde un registro de dispositivo corresponde a una comunicación de bajo nivel entre procesos. Como la sincronización entre procesos se consigue por medio de una interrupción, esta corresponde a una señal hardware. Conviene observar que el nombre interrupción sugiere un papel distinto al de una simple señal. De hecho, el nombre presupone un modelo que carece de cualquier forma de estructura multi-proceso; de ahí que la interrupción deba considerarse como un mecanismo para estimular actividades asíncronas. La complejidad resultante de dicho modelo se incrementa cuando no existen facilidades de programación de dispositivos de bajo nivel, como sucede en los tradicionales lenguajes secuenciales de tiempo real. Solamente el lenguaje ensamblador parece aportar la flexibilidad suficiente para soportar este modelo.

Dentro de un marco de multiproceso, el modelo resultante para las operaciones de E/S dirigidas por interrupciones aparece representado en los flujogramas de las figuras 2.8 y 2.9.

El CPU y los dispositivos periféricos trabajan concurrentemente utilizando señales para sincronizar sus actividades. Una señal de comienzo representa las acciones del CPU para iniciar el dispositivo de E/S y una señal de hecho representa la emisión de una interrupción por parte del dispositivo de E/S para indicar la terminación. En la práctica, el CPU no se encuentra parado esperando la señal hecho, sino que ejecuta otro proceso.

II.4. MECANISMOS HARDWARE.

A nivel hardware, una operación de E/S tiene lugar ejecutando una serie de intercambios de información entre dos elementos de lógica secuencial esencialmente asíncrona, es decir, el CPU y un dispositivo de E/S. La información se intercambia vía un bus de datos que con frecuencia es compartido por todos los dispositivos. El primer requerimiento esencial para programación de E/s, por tanto, poder direccionar los registros individuales de un dispositivo específico de E/S.

A fin de encajar los mecanismos del hardware actual en un modelo que se pueda utilizar para diseñar características abstractas del lenguaje de programación, el aspecto más fundamental es el hecho de que el hardware de un dispositivo de E/S pueda operar en paralelo con el hardware del CPU. Las antiguas arquitecturas de computadoras no poseían esta facilidad; en su lugar, el CPU se construía para que ejecutara un ciclo de espera de ocupación mientras que un dispositivo realizaba una operación. El estado del dispositivo era determinado por el CPU, enviando continuamente señales de estado (polling) al registro de estado del dispositivo. Esta clase de operación de E/S, por lo tanto, se conoce por el modo "polling" de E/S.

El inconveniente del modo "polling" es, por supuesto, que el CPU desperdicia un tiempo, justamente el tiempo que tarda en ejecutarse cada operación de E/S. Mas tarde, se introdujo un mecanismo por el que se podía obviar ese inconveniente; las interrupciones. Bajo el modo de interrupción de E/S, el CPU comienza una operación de E/S y luego puede ir a ejecutar otras tareas. El dispositivo de E/S funciona en paralelo y cuando ha completado la operación envía una señal al CPU activado una interrupción. El mecanismo de interrupción realiza dos funciones; la primera de ellas, salva el contexto del programa interrumpido; la segunda, hace que el CPU ejecute una rutina de servicio de la interrupción para efectuar cualquier operación requerida por el dispositivo que produjo la interrupción (interrumpió). Cuando concluye esta rutina de servicio, se restaura el contexto del programa interrumpido.

El mecanismo de interrupción facilita el medio de automatizar el protocolo "hand-shaking" (saludo) necesitado para permitir el que CPU realice la comunicación asíncrona y los dispositivos de E/S se comuniquen entre si. Sin embargo, la transferencia real de datos se sigue efectuando bajo el control del CPU. Cuando se trata de transferir grandes cantidades de datos, se puede utilizar un mecanismo hardware más sofisticado denominado Direct Memory Access (DMA - Acceso Directo a Memoria). Con este sistema el CPU indica al dispositivo DMA una dirección de memoria adonde desea transferir un bloque de datos (o desde donde), y el número de palabras del bloque. El dispositivo DMA ejecuta luego la transferencia del bloque de datos independientemente de la acción del CPU, "robando" ciclos de acceso a memoria a dicho CPU (por ejemplo, ejecutando un ciclo de memoria cada vez que el CPU ejecuta una operación interna del acumulador). Solamente cuando se ha transferido todo el bloque, el DMA enviará una señal de finalización interrumpiendo el CPU.

En resumen, los mecanismos hardware de transferencia de E/S pueden distinguirse principalmente por dos características fundamentales. La primera, el protocolo de E/S requerido puede implantarse en el software por medio de un "polling" o en el hardware por medio de un mecanismo de interrupción. La segunda, la transferencia de datos puede realizarse dato a dato siendo supervisada por el CPU o en bloque supervisada por un hardware DMA dedicado. La combinación encontrada más comúnmente en los sistemas de tiempo real es, sin duda, la transferencia dato a dato bajo control de interrupción; este mecanismo es la base del modelo abstracto desarrollado y que aparece en las figuras 2.8 y 2.9.

DISEÑO DEL SISTEMA DE TIEMPO REAL IMECA/STR.

III.1. Introducción.

Existen 3 principales aplicaciones para un STR:

- El objetivo de un sistema en tiempo real es dar apoyo inmediato a la carga de trabajo de una organización. Todo en un sistema de tiempo real esta enfocado a la obtención de información que pueda ser utilizada inmediatamente por la organización. Todos los programas contribuyen a este fin, y los usuarios tienen acceso a la información contenida en el sistema.

- Por lo general, los sistemas de tiempo real son utilizados por cualquier organización que maneje grandes volúmenes de datos importantes, los cuales deben de ser procesados inmediatamente, y sólo un sistema de tiempo real puede manejar tales volúmenes de información.

- Los sistemas de tiempo real dan apoyo a organizaciones dinámicas, que manejan información que cambia rápidamente y que proviene de diversas fuentes. La información se procesa en el momento en que se produce, proporcionando a los usuarios información lo más actualizada posible.

El presente trabajo contempla la elaboración de un STR, que sea fácil de manejar, fácil de entender, que sea rápido, eficiente, y que además que simule exactamente el funcionamiento de un Sistema de Tiempo Real de grandes dimensiones.

III.2. CONSIDERACIONES DE DISEÑO.

No obstante el propósito tangible de IMECA/STR, se persigue otro igualmente importante, que se refiere a que el usuario interesado en los sistemas de Tiempo Real pueda analizar fácilmente el programa de IMECA/STR, pueda modificarlo y sobretodo introducirle nuevas cualidades. Esto se logra mediante la utilización de un lenguaje tan versátil como lo es PASCAL (Secc. III.2.1) y esta es la razón para la utilización del mismo en STR realizado.

Es muy importante mencionar, que en el presente trabajo únicamente se presenta la versión básica de un sistema de mayores dimensiones y con un costo muy elevado, debido a esta situación, sólo es considerado el diseño y desarrollo del programa para computadora central, pero se mencionan algunas de las posibles características de los periféricos que se pueden utilizar para transferir la información de las estaciones recolectoras hacia la computadora central, además se presenta una Interface Programable de Periféricos (PPI 8255), la cual, viene a sustituir tanto a las estaciones recolectoras de información como a los dispositivos utilizados para transferir la información; debido a las características de la Interface, será posible simular a la perfección la adquisición de datos, su transmisión y su llegada a la computadora central.

Por último, debido a la situación antes descrita, los datos deberán ser introducidos en forma manual, esto es, tanto el número de estación como la cantidad de puntos de ozono serán introducidos manualmente con la ayuda de un jumper colocado en una tableta para el desarrollo de prototipos (ProtoBoard) y una fuente de alimentación, la cual, suministrara la energia suficiente (5 Volts) para alimentar a la Interface Programable (PPI). Este procedimiento se describe a detalle en la Sección V.2.1.

A continuación se presentan algunos aspectos a considerar para el diseño y posterior utilización de un IMECA/STR Y DE UN Sistema de Tiempo Real en general.

III.2.1. Consideraciones de la computadora.

La ejecución del STR relaciona a la computadora con el sistema natural utilizado, por lo que es necesario considerar algunas cualidades que debe reunir la computadora utilizada.

La computadora a ser utilizada puede tener alguna de las siguientes configuraciones de fábrica:

- Computadora AT 286, 386 o 486, 1MB de memoria RAM, disco duro de 40 MBytes, monitor VGA, puerto serie COM1 y puerto para la impresora.

- Computadora XT, 8088, 640 Kbytes de RAM, disco duro de 20 MB, monitor CGA, EGA, VGA o Hercules, puerto COM1 y puerto para impresora.

La mayoría de las computadoras tienen puertos serie, que utilizan para las impresoras, modems, redes de área local y otros propósitos de comunicaciones.
Aunque estos dispositivos facilitan el envío y la recepción de datos a través de los puertos serie, están demasiado limitados para la mayoría de los propósitos. Los programas de comunicaciones, por ejemplo, necesitan saltarse normalmente los dispositivos del DOS y utilizar directamente el puerto serie.

IMECA/STR tiene la cualidad de poder ser utilizado en cualquiera de las computadoras antes mencionada, incluso, esta computadora, pudiera no tener disco duro, pero lo que si es necesario tener en la computadora es un slot libre para poder colocar la interface programable PPI 8255 con la que será posible introducir datos a IMECA/STR y poder obtener el resultado de los datos procesados.

Lenguaje de programación utilizado.

Como se menciono en Capítulo I, después de la utilización de lenguajes de bajo nivel, se procedió a utilizar lenguajes muy poderosos de propósito general como el FORTRAN, BASIC, PASCAL, etc.

El lenguaje PASCAL es un lenguaje de alto nivel, de propósito general muy difundido y aceptado porque es fácil de aprender y de aplicar, aún para el usuario que no este muy familiarizado con la ciencia de la computación, además, PASCAL ofrece muchas ventajas sobre otros lenguajes de programación totalmente dedicados a lo que son los Sistemas de Tiempo Real (RTL/2 o MODULA), como por ejemplo, la flexibilidad, legibilidad, sencillez, transportabilidad, facilidad de programar dispositivos hardware, pero lo más importante, es un lenguaje prácticamente conocido por todos los desarrolladores de sistemas.

El propósito del presente trabajo es que IMECA/STR sea interactivo de tal forma que aunque el conocimiento del usuario respecto a sistemas computacionales sea muy pobre, le sea posible utilizarlo exitosamente, ya que este lo guia paso a paso para su operación.

No obstante este propósito tangible, se persigue otro igualmente importante, que se refiere a que el usuario interesado en los STR pueda analizar fácilmente el programa de IMECA/STR, pueda modificarlo y sobretodo introducirle nuevas cualidades. Esto se logra mediante la utilización de un lenguaje tan versátil como lo es el PASCAL y está es la razón para la utilización del mismo en el STR realizado.

III.3. PROBLEMATICA DE DISEÑO.

Como en todos los proyectos de desarrollo e investigación es prácticamente imposible encontrar una formula que nos permita librarnos de situaciones de error, así como para evitar algunos problemas en el diseño e implementación de los sistemas. A continuación mostraremos algunos puntos que es necesario mencionar sobre la problemática que representa desarrollar un STR, así como los detalles que principalmente pueden ocasionar problemas en su diseño y también algunas posibles soluciones para disminuir estos problemas.

El problema del tratamiento de errores.

Un requerimiento importante en muchos sistemas de tiempo real, si no es que en todos es la posibilidad de tratamiento de situaciones de error. Un sistema, siempre que ello sea posible, debe ser capaz de recuperase de situaciones de error. Debido a esto es necesario considerar todas, las posibles fallas o situaciones de error para poder implementar las soluciones y programar las rutinas necesarias para el manejo de dichos errores.

Los requerimientos de alta fiabilidad y de operación ininterrumpida en un sistema de tiempo real implican que un lenguaje de tiempo real debe facilitar mecanismos que permitan responder a condiciones de error y a programar acciones de recuperación. Un mecanismo de tratamiento de errores debe tener 5 requerimientos básicos: simplicidad, claridad, evitar sobrecargas en tiempo de ejecución a menos que se le invocara, tratamiento uniforme de los errores detectados en el programa y en la implementación, y la capacidad para programar acciones de recuperación.

Las acciones de recuperación de errores puede programarse con el mecanismo de escape, diseñando el manejador para que vuelva a calcular todo el procedimiento abortado, después de haber reparado los parámetros de entrada, o bien diseñándolo o para que restaure la causa del error y vuelva luego a invocar el procedimiento abortado.

Sea cual fuere el mecanismo aportado para el tratamiento de errores en un lenguaje de tiempo real, se deben satisfacer ciertos requerimientos . En primer lugar, el mecanismo debe ser sencillo de comprender y utilizar. En segundo lugar, la codificación para el tratamiento de errores no debe ser tan extensa como para que dificulte sin errores la comprensión de la operación normal, del programa. En tercer lugar, el mecanismo debería estar diseñado de tal forma que solamente se incurra en sobrecarga en el tiempo de ejecución cuando se este tratando el error, de forma que la ejecución no quede afectada bajo condiciones de operación normales. En cuarto lugar, el mecanismo debería permitir un tratamiento uniforme de los errores detectados por la implantación y por el programa. Finalmente, el mecanismo debería permitir la programación de acciones de recuperación.

El problema de distribuir en el tiempo un procesador.

La solución tradicional para distribuir en el tiempo (scheduling) un procesador entre varios procesos es la del tiempo compartido (time-sharing). El procesador se conecta a través de una línea de interrupción a un reloj. En cada interrupción, el procesador suspende la ejecución del proceso en curso y selecciona otro para su ejecución.

El problema de elegir un lenguaje de programación.

En primer lugar, debe ser extremadamente fiable. Un fallo del sistema puede ser muy caro en términos de perdida de producción en el caso de un control de proceso o en términos de la propia vida humana en situaciones extremas , como la navegación. En segundo lugar, es generalmente voluminoso y complejo. Esto, combinado con el problema de la fiabilidad, significa que los costos de mantenimiento y desarrollo pueden ser muy elevados. En tercer lugar, de debe responder a una variedad de eventos externos con tiempos de respuesta garantizados. En cuarto lugar debe admitir la conexión con una gran variedad de dispositivos de E/S no estandares, así como de los periféricos convencionales. En quinto lugar, debe ser eficiente para poder hacer la mejor utilización posible del equipo hardware disponible.

El problema de elegir el hardware para tiempo real.

De las muchas características que distinguen un sistema de tiempo real de otros sistemas basados en una computadora, es de fundamental importancia la singular relación entre la computadora y su entorno.

Precisamente, lo que caracteriza a una computadora de tiempo real es la necesidad de sincronizar sus operaciones con los estímulos externos y de comunicarse con una amplia gama de dispositivos "extraños" de E/S. Esta necesidad implica que un lenguaje de tiempo real debe incorporar facilidades para poder programar directamente los dispositivos hardware como son los puerto COM1, COM2, LPT, tarjetas controladoras, etc.

III.4. ALGORITMO DE IMECA/STR

Una vez considerados los aspectos más importante de un STR, de la problemática que se presenta cuando se quiere realizar un STR y la computadora a utilizar, el siguiente paso en el desarrollo de IMECA/STR es la construcción del diagrama a bloques que incluye todas las funciones necesarias para llevar a cabo el buen funcionamiento del sistema.

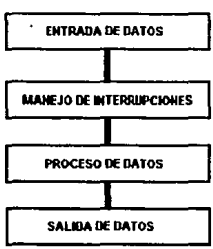


Fig. 3.1 Módulos de IMECA/STR.

Un STR consta de 4 elementos principales: La entrada de datos, el manejo de interrupciones, el proceso de datos y la salida de los datos procesados. Estos elementos se presentan en la figura 3.1.

En base a los elementos anteriores, se construye el algoritmo de IMECA/STR que se muestra en la figura 3.2.

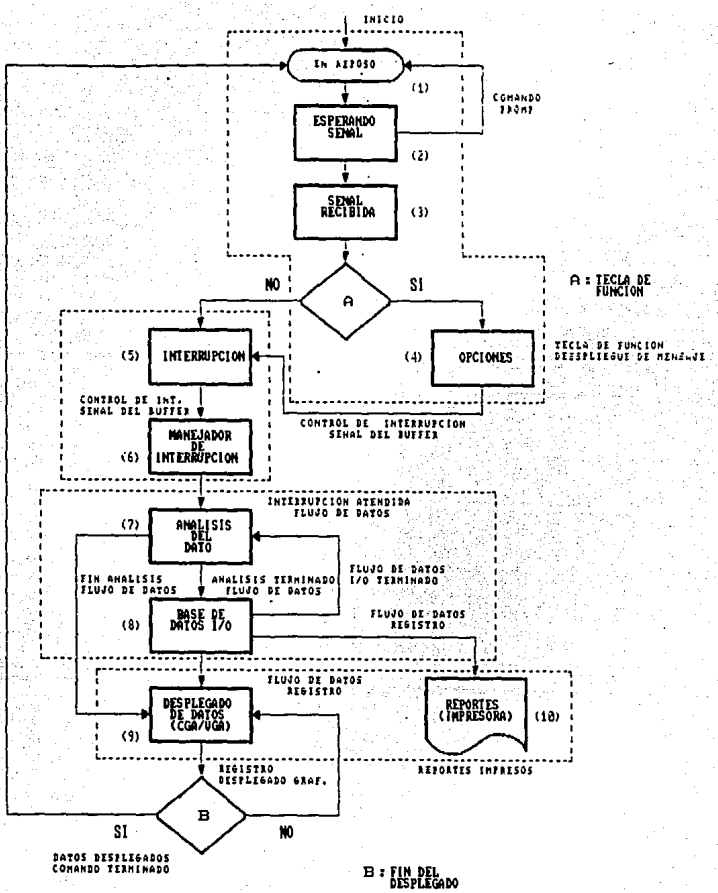


Fig. 3.2. Algoritmo de IMECA/SIR.

Descripción del algoritmo.

El Bloque 1 "Esperando señal" , el STR se encuentra en un estado de "reposo" en aparente calma. El Bloque 2 "Esperando señal" el STR se encuentra en espera de una señal de entrada o requisición del sistema,. El Bloque 3 "Señal recibida" efectúa la lectura de la señal externa aplicada al sistema. El Bloque 4 "Opciones" alguna tecla especial ha sido pulsada. Estos cuatro bloques (1-4) forman parte del Modulo de Entrada de Datos, que se describe en la Secc. III.4.1.

El Bloque 5 "Interrupción" es detectada una señal (dato de entrada o tecla de función) porque se produce una interrupción . El Bloque 6 "Manejador de interrupción" se controla la interrupción a través de un buffer y se le asigne una prioridad. Los Bloques 5 y 6 forman parte del Modulo de manejo de interrupciones, que se describe en la sección III.4.2.

El Bloque 7 "Análisis del datos" la interrupción generada es atendida y el dato es analizado y se le aplica el proceso correspondiente. El Bloque 8 "Base de Datos I/O" el dato analizado es almacenado en una base de datos si esto es necesario. Estos dos Bloques constituyen el Modulo de Proceso de Datos (Secc. III.4.3).

El Bloque 9 y el Bloque 10 conforman el Modulo de Salida de datos. Estos datos pueden provenir tanto de la Base de Datos como del proceso directo de los datos, este Modulo se describe en la Secc. III.4.3.

III.4.1 Modulo para la Entrada de Datos.

El Modulo de entrada de datos efectúa la lectura de todos los datos introducidos por el usuario. Estos datos pueden ser de dos tipos, el primero de ellos es un dato de entrada vía puerto de la computadora (COM1 - PPI8255) y el segundo es una requisición del usuario como lo son los disponibles en las teclas de función (F1-F10), donde el usuario en cualquier momento pude hacer uso de los recursos del sistema. Al pulsar cualquier tecla de función un mensaje asociado a dicha tecla será desplegado.

Después de haberse recibido una señal, esta se canaliza al Modulo de Manejo de Interrupciones, para que dichas señales sean atendidas conforme vallan llegando. Dicho modulo se describe en la Secc. III.4.2.

III.4.2. Modulo de Manejo de Interrupciones.

Como se menciono en la Sección I.7. una de las características fundamentales de los sistemas de tiempo real es contar con una manejador de interrupciones. Cuando una interrupción se presenta debido a un dato de entrada un sistema de buffers actúa de modo que dichos datos puedan ser procesados rápidamente cuando el sistema lo requiera. En los Sistemas de tiempo real se presentan múltiples estímulos por lo que el manejador de interrupciones deberá de conocer la tarea de mayor importancia y siempre está deberá ser atendida dentro de un tiempo establecido pase lo que pase con otros eventos.

Existen dos tipos de interrupciones: hardware y software. Las interrupciones hardware, generadas por acciones como la pulsación de una tecla, un pulso de reloj del sistema o la entrada de datos por el puerto serie, se originan en la circuiteria de la computadora y están controladas por un chip especial, el controlador de interrupciones 8259.
Las interrupciones software se generan con programas que solicitan servicios especiales del DOS (Disk Operating System). En Turbo Pascal, las ordenes Intr y MsDos crean interrupciones software.

IMECA/STR utiliza la siguiente interrupción para tener control sobre el puerto serial:

Interrupción ROM/BIOS	Función
14h	E/S del puerto serie

Después de que la interrupción ha sido atendida, el dato fluye al Modulo de Procesamiento de los Datos que se describe en la Secc. III.4.3.

III.4.3. Modulo Proceso de Datos.

Con este Modulo se procesan los datos provenientes del manejador de interrupciones. Este proceso va desde cambio de fecha para el sistema hasta el almacenamiento de la información en un archivo (Base de Datos) para un posterior proceso o reportes requeridos por el usuario.

El Análisis del Dato lo realiza IMECA/STR a través de este modulo dicho dato puedé ser enviado directamente al Modulo de Salida de Datos o primero pueden ser enviados a la Base de Datos utilizada por IMECA/STR y luego pasar al Modulo de Salida, este modulo se describe en la Secc. III.4.4.

III.4.4. Modulo de Salida de Datos.

El Modulo para la salida de datos esta formado por dos Bloques, los cuales tienen la función de presentar al usuario los datos procesados por el sistema. El primero de estos bloques (Bloque 9) tiene la función de desplegar los datos en la pantalla de la computadora. Dichos datos pueden provenir de la base de datos utilizada por el STR o directamente de la entrada de datos, pasando por el modulo de manejo de interrupción y por el Modulo de procesamiento de datos. El segundo Bloque (Número 10) tiene misma función que el bloque anterior sólo que el resultado de los procesos solamente provienen de la Base de Datos y son enviados directamente a la impresora (Reportes impresos). Estas opciones se describen a detalle en el manual de usuario (Capítulo V).

CAPITULO IV.

DESCRIPCION DEL SISTEMA.

IV.1. INTRODUCCION.

El STR que se describe a continuación (IMECA/STR básico), es un sistema completo y eficiente. Sin embargo, debido a sus fines demostrativos y didácticos, tiene algunas limitaciones, pero es posible aumentar sus capacidades para hacerlo más fácil de usar, más rápido y más versátil.

Para aumentar las capacidades y facilitar su uso, es necesario mejorar la entrada de datos, es decir, hacer más sencillo para el usuario la introducción de los datos por el dispositivo de entrada hacia el sistema (IMECA/STR). Una forma de lograr lo anterior es mediante la utilización de la interface RS232 y un MODEM.

En el presente trabajo sólo es considerada la versión básica de IMECA/STR, por lo que, sólo se explica brevemente la forma de trabajar del puerto serial y los tipos de comunicación que hay a través de las líneas telefónicas. Dando un énfasis a la interface programable de periféricos 8255, que es utilizada como el dispositivo para la adquisición de datos por la computadora y a su vez por IMECA/STR.

IV.2. ENTRADA DE DATOS.

La versión básica de IMECA/STR tiene como objetivo principal dar a conocer las funciones, ventajas y desventajas de todos los sistemas de tiempo real, así como los aspectos más importantes que se deben considerar para el diseño de un STR.

En general, todos los sistemas computacionales necesitan consumir datos, procesarlos y producir una salida. Para un sistema de tiempo real, estos procesos deben de realizarse a una gran velocidad y por lo general se utiliza algún dispositivo de entrada externo para la obtención de dichos datos.

Los grandes sistemas de tiempo real reciben datos del exterior a través de dispositivos alejados de la computadora central, como son los satélites, una antena de radio frecuencia, o incluso otra computadora.
El medio más común para la transferencia de información son las líneas telefónicas, pero además, estas señales deben de ser controladas y traducidas por un Modem (Modulador/Demodulador), que debe de estar en ambos lados de la transferencia (Transmisor/Receptor).
Un Modem es una unidad que incorpora una técnica para mandar y recibir señales de computadoras sobre una portadora común de comunicaciones.

En IMECA/STR hemos sustituido dichos dispositivos por una Interface Programable de Periféricos, el 8255. Esta interface con la ayuda de un microswitch nos proporcionara los datos de entrada necesarios para el funcionamiento de IMECA/STR.

La pantalla principal de IMECA/STR presenta el estado actual de cada una de las 6 estaciones que existen para el sistema, cada estación se encuentra dentro de un recuadro, los seis recuadros presentan 3 etiquetas principales y que son:

NIVEL MAX
NIVEL INS
NIVEL MIN

El NIVEL MAX (125 puntos) permanece invariable en la pantalla y nos indica el nivel máximo de puntos de ozono que son aceptables de acuerdo a normas internacionales, por lo que no tienen consecuencias graves para la salud. Cuando este nivel es rebasado, es deber de los organismos autorizados proporcionar y aplicar medidas para informar y auxiliar a la población.

El NIVEL INS es el nivel instantáneo, esto es, los puntos de ozono recolectados en ese instante por las estaciones recolectoras de información . Cabe notar que no necesariamente todas las estaciones se deben reportar al mismo tiempo, esto es debido a diferentes factores ajenos al sistema, como son la distancia entre las estaciones y el puesto central, las condiciones meteorológicas, etc.

El NIVEL MIN (10 puntos) sólo es un marco de referencia para el usuario, ya que, a menor número de puntos de ozono, mejores serán las condiciones de salud para toda la población.

IV.2.1. La tarjeta de Bus para PC.

A menudo nos encontramos los conectores estandar de las computadoras, ocupados por dispositivos diversos: impresora, ratón, etc. Por ello, a la hora de controlar elementos externos a través de la computadora es recomendable poder disponer de una interface propia y especialmente diseñada para este cometido.

En el presente trabajo se muestra un montaje que accede directamente al bus de datos PC (8 bits) a través del chip 8255, que genera 3 puertos de 8 bits programables como entradas o salidas y puede operar de tres diferentes modos (0,1, y 2). En el modo 0 el puerto A puede operar como entrada o como salida, IMECA/STR sólo utiliza a este puerto como entrada.

El 8255 tiene 4 puertos. El puerto A esta siempre en la dirección base, el puerto B esta en la base+1, el puerto C esta en la base+2, y el puerto de control esta en la base+3. Por ejemplo, si el jumper de la tarjeta esta en la posición 3, la dirección base debe se 576, por lo que se podrá accesar al puerto B (577), al puerto C (578), y el puerto de control en la 579.
IMECA/STR utiliza este conjunto de direcciones para su operación.

El Bus PC tiene 62 contactos tipo borde de tarjeta, 31 por cada cara (A/B), en pasos de 0.1 pulgadas.
El Bus AT ISA y el EISA-386 tienen un mayor tamaño, pero son compatibles con el Bus PC, mediante las ranuras para 8 bits.

Descripción.

Utiliza sólo 2 integrados.

1 Decodificador/Demultiplexor 74LS138. El dispositivo es ideal para ser utilizado para los chips de memoria bipolares de alta velocidad que selecciona direcciones decodificando señales.

1 PIA 8255 que es el elemento principal del diseño y proporciona 3 puertos de 8 bits programables como entradas o salidas. Estos puertos son compatibles con TTL y este puede operarse de 3 diferentes modos. Dependiendo el modo las líneas en cada puerto actúan diferente.
Utiliza una dirección para cada puerto y otra adicional para el control de puertos.

1 Microswitch de 8 posiciones para manejar 8 posibles direcciones.

IMECA/STR es únicamente un sistema informativo, por lo que no es necesario regresar o mandar datos ya procesados a un dispositivo externo para controlar o verificar dispositivos.
Debido a esto, IMECA/STR solamente utilizará uno de los 3 puertos de que dispone el PPI 8255, el puerto A, que en el modo 0 opera como entrada.

En la figura 4.1 se muestra el diseño completo de esta interface.

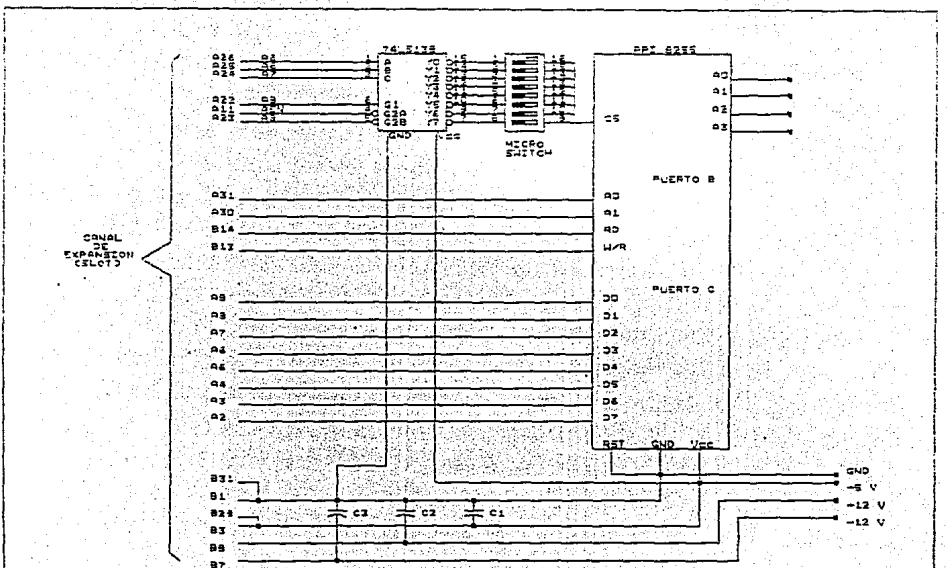


Fig. 4.1 Configuracion general para la tarjeta de Bus PC.

El microswitch utilizado en el diseño tiene 8 posiciones y el cual es utilizado para mandar los datos a la computadora. Con estas 8 posiciones podemos producir 256 datos binarios, esto es, 0, 1, 2, 4, 8, 16, 32, ..,255 combinando las 8 posiciones en el microswitch. En la figura 4.2 se muestra el microswitch utilizado para el diseño.

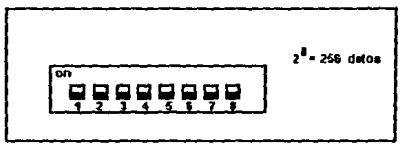


Fig. 4.2 Microswitch de 8 posiciones.

La forma de como utilizar el microswitch para introducir los datos a la computadora y que estos puedan ser consumidos por IMECA/STR se describe en el Capítulo V.

IV.3. DESPLIEGUE DE PANTALLAS UTILIZANDO GRAFICOS.

Una de las caracteristicas que ayudan apreciablemente la comunicación hombre-máquina es el uso de gráficos, que son utilizados ampliamente en los sistemas por microcomputadora debido a la facilidad de interpretación de información.

En el presente trabajo se han empleado gráficos para facilitar la captura e interpretación de datos.
Se presentan en la pantalla los gráficos correspondientes a todas las estaciones manejadas por el STR.

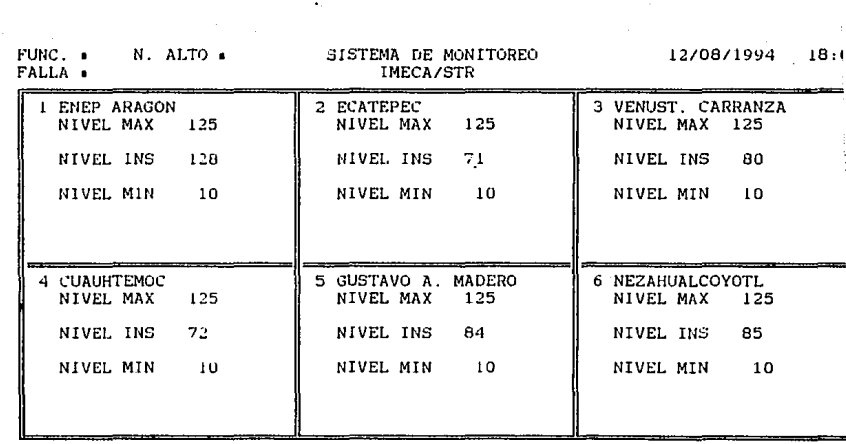
Cuando es necesario capturar datos, debajo de cada gráfico aparece un campo de captura donde el usuario puede introducir los datos correspondientes. Para seleccionar alguno de los elementos que aparecen en la parte inferior de la pantalla, el usuario simplemente deberá utilizar la tecla correspondiente en el teclado.

Una vez seleccionada la opción con su tecla asignada, aparece una nueva pantalla con el gráfico correspondiente y campos de captura si es que es necesario introducir nuevos datos. Para regresar al gráfico principal o primario basta con pulsar << intro >>.

El ejemplo más representativo en el presente trabajo en la utilización de gráficos se muestra en el mapa del Distrito Federal y la parte Norte del área metropolitana, donde se muestra el lugar exacto de las estaciones recolectoras de información (puntos de ozono).

La figura 4.3 muestra la pantalla principal de IMECA/STR con sus diferentes opciones en la parte inferior.

Fig. 4.3 Pantalla principal de IMECA/STR.



IV.4. Impresión de reportes sobre el estado que tiene el sistema.

El Sistema Operativo DOS (Disk Operating System) soporta varios dispositivos de impresión: PRN, LPT1, LPT2 y LP3 (LPT1 y PRN se refieren al mismo dispositivo). La mayoría de la gente utiliza una sola impresora, por lo que sólo utilizan los dispositivos LPT1 y PRN. Turbo Pascal ofrece una forma de dirigir la salida a la impresora: la unidad Printer. Esta unidad declara una variable de archivo de texto de llamada LST que dirige la salida a la impresora.

IMECA/STR es un sistema básico de información, por lo que es necesario dar a conocer el resultado del procesamiento de los datos que entran a la computadora y son consumidos por IMECA/STR.

Estos datos provienen de la Base de Datos utilizada por IMECA/STR, esta Base de Datos guarda todos los reportes (datos) recibidos por la diferentes estaciones conectadas al sistema y les asigna la fecha correspondiente al día, mes y año en que se reportaron, así como la cantidad medida de puntos de ozono.

Muchas veces dicha información es solicitada por organismos externos o departamentos alejados de la computadora central y que no tienen conexión vía red a esta computadora, por lo que es necesario proporcionar la información mediante un reporte impreso.

IMECA/STR nos permite dar este servicio a través de su Modulo de Salida de Datos (Sección III.4.4).
Para obtener dicha información sólo es necesario presionar la tecla de función correspondiente y proporcionar la fecha del reporte para enviarlo a la cola de impresión

En el Capítulo V se describe a detalle la forma en que se puede utilizar este modulo para la impresión de reportes.

I.V.5. SISTEMA DE DETECCION DE GASES TOXICOS (SDGT).

En esta sección se describe un sistema que puede ser utilizado por IMECA/STR para la recolección de información (gases tóxicos), su análisis, y su posterior transmisión en forma conjunta con un Modem. En la versión básica de IMECA/STR no es utilizado este sistema debido a su alto costo, sumado a que es necesario utilizar para la transmisión de información una línea telefónica y un Modem, dicho sistema podrá ser útil en una versión posterior de IMECA/STR ya que en este trabajo sólo se presenta la parte que corresponde a la computadora central y no a los periféricos que utiliza el Sistema de Tiempo Real IMECA/STR.

CARACTERISTICAS PRINCIPALES

Detección del nivel de concentración de gas tóxico de la atmósfera en el punto de muestreo.

Sistema de sensado de gases intrínsecamente seguro para atmósferas según normas NEC CLASE 1, DIVISION 1, GRUPOS A, B, C y D.

Operación continua 24 horas al día, 365 días del año.

Toma de muestra hasta una distancia de 50 metros del sensor utilizando un aspirador intrínsecamente seguro, lo que permite el muestreo en áreas de difícil acceso o riesgosas.

Respaldo de batería para operación contínua en caso de fallas de energía.

Los sensores pueden colocarse a una distancia de hasta 1500 metros de terminal remota.

No se requiere que la instalación sea a prueba de explosión ni de procedimientos especiales para la instalación y mantenimiento.

Empleo de sensores redundantes para asegurar que la información sea siempre confiable y que se activen alarmas cuando realmente exista una condición de emergencia, ademas, el mismo sistema reporta cuando algún sensor requiere calibración.

Procedimiento de calibración sencillo, que no requiere personal especializado y que se realiza por una sóla persona.

APLICACIONES

Plantas Industriales.
Complejos industriales.
Tanques de Almacenamiento de Combustibles.
Agencias de Gas Combustible.
Expendios de Gasolina y diesel.
Detección de Gases Explosivos en drenajes.

COMPONENTES

Sensor.-

Sistema intrínsecamente seguro.
Calibración sencilla con 1 sola persona.
Distancia entre terminal y sensor hasta 1.5 Km

Terminal Remota.-

Capacidad para conexión de hasta 15 sensores.
Alarma local.
Panel luminoso para indicación local.
Unidad inteligente para conversión de unidades y detección de alarmas.
Respaldo de batería.

Anunciador de Alarmas.-

Con opción de primera alarma por grupo de alarmas. Puede conectarse en red con varias terminales remotas y con otro anunciadores.
Con memoria de primera alarma.

OPCIONES

Sellado redundante con aspirador.- Ideal para muestreos en áreas de difícil acceso o peligrosas; para aplicación en drenajes, cisternas, pozos y donde se requiere proteger al sensor de la intemperie.

Pantalla luminosa de cristal líquido.- Presenta indicación contínua de las mediciones en %LEL o ppm, para integrar en anunciador de alarmas o terminal remota.

Comunicaciones .- Puerto Serie RS-232C como estandar
Serial RS-422/485
Línea privada o hilo piloto.
Radio UHF o VHF.
Modem de 9600 bps o 14400 bps.

Estación de supervisión.-

Indicación de los valores instantáneos y de alarmas.
Archivos históricos
Desplegados gráficos de curvas de tendencias.
Capacidad para hasta 256 terminales remotas.

GLOSARIO DE TERMINOS PARA EL SDGT.

INTRISECAMENTE SEGURO: NEC(National Electrical Code) define al equipo y alambrado intrínsecamente seguro como "incapaz" de descargar suficiente energía eléctrica o térmica bajo condiciones normales o anormales para causar ignición en una atmósfera con mezclas explosivas.

NEC: National Electrical Code de Estados Unidos. El cual provee definiciones básicas para áreas peligrosas. En el articulo 500, define CLASE 1, DIVISION 1 como "Areas" donde liquidos inflamables son manejados o transferidos; y áreas adyacentes a calderas, tanques, mezcladores, etc.

NIVEL MINIMO DE EXPLOSIVIDAD (%LEL): Para gases o vapores que forman mezclas inflamables con aire u oxigeno, hay una concentración mínima de gas o vapor, por abajo de la cual no puede ocurrir la propagación de la flama al contacto con una fuente de ignición.

SENSOR: En este trabajo, sensor es un dispositivo que monitorea continuamente la concentración de un gas en un punto determinado. (a diferencia de un analizador, el cual determina la cantidad, calidad y/o tipo de substancia o substancias especificas en una mezcla.

CAPITULO V.

MANUAL DE USUARIO.

V.1. INTRODUCCION.

El presente trabajo tiene como objetivo que IMECA/STR sea interactivo, esto es, que permita establecer una relación amable entre el usuario y la máquina. Dicho en otras palabras, el objetivo es que el usuario sea guiado por el sistema para la operación del mismo.

En el presente Capítulo se dan las instrucciones necesarias para la operación de IMECA/STR en su versión básica y se mencionan algunas alternativas para hacer los cambios más comúnmente deseados.

Para la operación de IMECA/STR el usuario debe conocer la computadora a utilizar; tener presente que el Sistema Operativo utilizado es MS-DOS; y que el STR esta escrito en PASCAL.

V.2. OPERACION DE IMECA/STR VERSION BASICA.

En esta sección se describe el procedimiento para la correcta operación de IMECA/STR, así como de la computadora utilizada.

A continuación se enumeran los pasos para cargar el STR en la memoria de la computadora.

1.- Insertar diskette conteniendo los siguientes archivos:

- IMECA.EXE
- EST.DAT
- REPORT.DAT
- DF.DAT
- ESTACION.DAT
- EDOMEX.DAT
- DIRECTO.DAT
- ESTIMP.DAT

2.- Ejecutar el programa tecleando el siguiente comando:

IMECA << return >>

Con lo que aparecerá en la pantalla el siguiente mensaje:

BIENVENIDOS AL SISTEMA DE TIEMPO REAL

A continuación aparecerá un gráfico de presentación para el sistema, posteriormente se presenta en la pantalla las teclas de función F1 a F10, con sus correspondientes tareas asociadas y las estaciones de monitoreo presentado sus respectivos punto máximos (NIVEL MAX), mínimos (NIVEL MIN), y los datos medidos y enviados por cada una de las 6 estaciones existentes.

V.2.1. Teclas de función.

En IMECA/STR se encuentran a disposición del usuario una serie de teclas especiales llamadas teclas de función marcadas en los teclados de las computadoras compatibles con la tecla " F " y un número consecutivo (generalmente del 1 al 10). Dichas teclas pueden realizar diversas funciones en IMECA/STR.

Tecla F1 (RELOJ).

La tecla de función F1 permite establecer una nueva hora para el sistema, la cual aparece en la parte superior derecha de la pantalla. Al presionar la tecla F1 se desplegara una pantalla donde se deberá teclear la nueva hora para el sistema, así como los minutos. El sistema estará esperando a que se introduzca la nueva hora como se muestra:

TECLEE LA HORA

Después de haber introducido la hora se deberá pulsar << return >>, con lo que aparecerá un desplegado como el siguiente:

TECLEE LOS MINUTOS

Posteriormente se pulsara nuevamente << return >> y el sistema tendrá la nueva hora establecida por el usuario.

Tecla F2 (LOCAL).

La tecla de función F2 tiene como objetivo el de proporcionar al usuario el nombre, y la dirección exacta de la estación recolectora y transmisora de información. Al presionar esta tecla, se desplegara en la pantalla un gráfico con la información mencionada anteriormente. Para salir de esta pantalla, basta con pulsar << return >>.

Tecla F3 (RPAN).

La tecla de función F3 tiene la función de presentar en la pantalla un reporte sobre la cantidad medida de puntos de ozono en una fecha dada por el usuario. Al pulsar esta tecla, se desplegara una pantalla de captura, en la cuál, será necesario teclear la fecha del reporte que se desea conocer con los siguientes formatos:

a) DMAAAA
b) DMMAAAA
c) DDMAAAA
d) DDMMAAAA

Ejemplos:

121994 => 1 de Febrero de 1994.
5121994 => 5 de Diciembre de 1994.
1561994 => 15 de Junio de 1994.
30101994 => 30 de Octubre de 1994.

Para salir de esta opción basta con pulsar << return >>.

Tecla F4 (RIMP).

La tecla de función F4 tiene el mismo objetivo que la tecla F3 (RPAN), solo que la información es enviada directamente a la cola de impresión, por lo que, con esta opción, es posible obtener reportes impresos sobre los puntos de ozono medidos en cada una de las 6 estaciones.
Al igual que con la tecla F3 aparecerá una pantalla, donde se deberá introducir la fecha del reporte a imprimir, con el mismo formato de fecha que en la opción anterior.

Tecla F5 (PRE).

La tecla de función F5 tiene el objetivo de presentar al sistema IMECA/STR, mostrando el nombre completo del proyecto, así como el nombre de los autores y el del asesor del proyecto. Esta información aparecerá durante algunos segundos, posteriormente retornara a la pantalla principal sin necesidad de pulsar alguna tecla.

Tecla F6 (MAPA).

La tecla de función F6 tiene la función de presentar al usuario un mapa del Distrito Federal en su totalidad, así como la zona noreste del Area Metropolitana. Este mapa, además, contiene la localización de las diferentes estaciones de monitoreo ambiental.
Para salir de esta opción y retornar a la pantalla principal, basta con pulsar << return >>.

Tecla F7 (EXP).

La tecla de función F7 tiene la función de explorar o detectar el número de la estación transmisora, así como la cantidad de puntos de ozono.

Debemos recordar que la Interface Programable de Periféricos (PPI) viene sustituyendo a los dispositivos automatizados para la transmisión de información, por lo que tanto la estación transmisora como la cantidad de puntos de ozono serán introducidos manualmente con ayuda del jumper localizado en una tarjeta de prototipos (ProtoBoard).

El procedimiento es el siguiente:

Seleccionar el número de la estación con el jumper y pulsar F7, con lo que se desplegará durante algunos segundos en la pantalla un letrero como el siguiente:

EXPLORANDO ESTACIONES

Posteriormente se desplegará un segundo mensaje, como este:

RECIBIENDO NUMERO DE LA ESTACION

Para lo que se deberá pulsar << return >>, apareciendo en la pantalla el número de la estación transmisora, ahora se deberá introducir la cantidad de puntos de ozono deseados con la ayuda del jumper, y se desplegará un tercer mensaje como este:

RECIBIENDO INFORMACION DE PUNTOS DE OZONO

Nuevamente se pulsará << return >> y se desplegará en la pantalla la cantidad de puntos de ozono recolectados por dicha estación. Para regresar a la pantalla principal de IMECA/STR pulsar nuevamente << return >>, desplegándose dicha información en la estación correspondiente.

Tecla F8 (GRAB).

La tecla de función F8 guarda la información que hasta ese momento tiene el sistema para futuras consultas, y se debe seguir el siguiente procedimiento:
Se pulsará F8 y en la pantalla se desplegará un mensaje, indicando el formato en el que se deberá teclear la fecha para dicho reporte, esta información se grabará en un histórico de datos.

Tecla F9 (AYU).

La tecla de función F9 tiene como objetivo el de proporcionar al usuario que no está familiarizado con el sistema una pequeña ayuda en línea sobre algunos aspectos importantes del sistema, así como una breve descripción de la teclas de función que pueden ser utilizadas en IMECA/STR.

Tecla F10 (FIN).

La tecla de función F10 nos permite salir del sistema. Al pulsar esta tecla se desplegará en la pantalla un mensaje como este:

DESEA GUARDAR EL ARCHIVO

En el caso de no querer guardar la información, se deberá pulsar la letra N, por el contrario, si se desea conservar está información, basta con pulsar la letra S y posteriormente IMECA/STR pedirá la fecha con la cuál se guardará el reporte en un histórico de datos.

V.2.2. Mensajes de error.

En cuanto a los mensajes de error, resulta conveniente señalar la diferencia entre los mensajes producidos por IMECA/STR y los mensajes originados por el compilador PASCAL o por DOS. Los primeros mensajes son aquellos que facilitan la operación de IMECA/STR y los últimos son generados por el compilador de PASCAL y DOS cuando no se cumple con el formato preestablecido para la introducción de datos. Los mensajes de ambos tipos pueden aparecer por separado o en combinación.

Mensajes generados por IMECA/STR y de DOS.

ERROR	DESCRIPCIÓN
Archivo no encontrado.	El nombre o la fecha del archivo no existen.
Formato numérico no válido.	Se introdujeron letras en lugar de números o viceversa.

CONCLUSIONES.

La contaminación en el Valle de México cada día aumenta a gran velocidad por lo que es necesario contar con un sistema automatizado de alta velocidad para tener al tanto a los diferentes organismos encargados de informar a la población y de aplicar los diferentes operativos creados para proteger a esta población en caso de un alto grado de contaminación.

Los Sistemas de Tiempo Real tienen un amplio campo de aplicación, pero tienen el inconveniente de ser en general muy caros, debido a que deben interactuar con grandes líneas y equipos de comunicación de datos, por lo que es necesario realizar una correcta planeación del proyecto y fijar los objetivos a corto, mediano, y largo plazo.

En el presente trabajo se dan las bases teóricas del diseño y funcionamiento de los Sistemas de Tiempo Real (STR), y se implementa la versión básica de IMECA/STR que puede ser utilizado tanto en las universidades como en la industria, para mostrar las ventajas de los STR, debido a la facilidad de operación y a la confiabilidad de los resultados.

El objetivo final de IMECA/STR es el de proporcionar un paquete de información confiable y disponible a cualquier hora del día y los 365 días del año, pero, debido al gran gasto económico que representa un sistema de este tipo en el presente trabajo sólo es considerado el proyecto piloto de un sistema de mayores dimensiones y que se puede utilizar no sólo en la recolección y transmisión de información sobre contaminantes, si no que además puede ser utilizado para proporcionar información tan variada como son los niveles de agua, presencia de gases nocivos en plantas industriales, etc.

IMECA/STR puede ser fácilmente utilizado por estudiantes y operadores de sistemas, aún cuando estos carezcan de conocimientos en computación. Lo anterior es posible debido a que el STR es de tipo interactivo puesto que permite establecer un diálogo entre el usuario y la computadora para su operación. Además, el STR está estructurado de tal forma que permite al usuario con conocimientos en computación, entender el programa, modificarlo, y mejorarlo.

La existencia de IMECA/STR proporciona al estudiante y al operador de sistemas gran experiencia en la operación y el entendimiento de los dispositivos y sistemas digitales, así como el manejo de información por computadora que es la tendencia actual en la ingeniería.

```
{ $M 65520,0,655360}
(* SISTEMA DE TIEMPO REAL PARA LA INFORMACION DE PUNTOS DE OZONO EN EL *)
(* D.F. Y AREA METROPOLITANA REALIZADO POR LOS ALUMNOS ROBERTO MONTES *)
(* MONROY Y ARTURO MARES GOMEZ                                        *)

PROGRAM IMECA;
 USES CRT,DOS,PRINTER,GRAPH;

  VAR
   EN,EC,VE,CU,GU,NE,DATO,EST,KI:INTEGER;



 PROCEDURE UNAM; (*PRESENTACION DEL SISTEMA*)
  VAR
   CONTROLGR,MODOGR:INTEGER;
   A,B:WORD;

   BEGIN
    WINDOW(1,1,80,25);
    TEXTBACKGROUND(BLACK);
    TEXTCOLOR(WHITE);
    CLRSCR;
    CONTROLGR:=DETECT;
    INITGRAPH(CONTROLGR,MODOGR,'');
    SOUND(1000);
    DELAY(100);
    NOSOUND;
    SETCOLOR(14);
    SETTEXTSTYLE(0,0,5);
    MOVETO(250,35);
    OUTTEXT( 'UNAM' );
    SETCOLOR(9);
    MOVETO(248,35);
    OUTTEXT( 'UNAM' );
    SETCOLOR(14);
    SETTEXTSTYLE(0,0,3);
    MOVETO(195,90);
    OUTTEXT( 'ENEP ARAGON');
    SETCOLOR(9);
    SETTEXTSTYLE(0,0,3);
    MOVETO(193,90);
    OUTTEXT( 'ENEP ARAGON');
    SETCOLOR(14);
    SETTEXTSTYLE(0,0,2);
    MOVETO(55,150);
```

```
 OUTTEXT('TESIS DE INGENIERIA EN COMPUTACION');
 SETCOLOR(9);
 SETTEXTSTYLE(0,0,2);
 MOVETO(54,150);
 OUTTEXT('TESIS DE INGENIERIA EN COMPUTACION');
 SETCOLOR(14);
 SETTEXTSTYLE(0,0,1);
 MOVETO(85,210);
 OUTTEXT('SISTEMA DE TIEMPO REAL PARA LA INFORMACION DE PUNTOS DE OZONO ');
 MOVETO(195,230);
 OUTTEXT(' EN EL D.F. Y AREA METROPOLITANA');
 SETCOLOR(11);
 SETTEXTSTYLE(0,0,2);
 MOVETO(85,295);
 OUTTEXT('ALUMNOS :');
 SETTEXTSTYLE(0,0,2);
 MOVETO(255,295);
 OUTTEXT('MARES GOMEZ ARTURO');
 MOVETO(255,335);
 OUTTEXT('MONTES MONROY ROBERTO');
 SETTEXTSTYLE(0,0,2);
 MOVETO(75,410);
 OUTTEXT('ASESOR: ING. JUAN GASTALDI PEREZ');
 DELAY(7000);
 SOUND(1000);
 DELAY(100);
 NOSOUND;
 CLOSEGRAPH;
END;


PROCEDURE FUNCIONES;  (* TECLAS DE FUNCIONES *)

BEGIN
 WINDOW(1,24,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);

 GOTOXY(1,2);
 TEXTCOLOR(GREEN); WRITE('F1'); TEXTCOLOR(YELLOW); WRITE(' RELOJ');
 TEXTCOLOR(GREEN); WRITE(' F2'); TEXTCOLOR(YELLOW); WRITE(' LOCAL');
 TEXTCOLOR(GREEN); WRITE(' F3'); TEXTCOLOR(YELLOW); WRITE(' RPAN');
 TEXTCOLOR(GREEN); WRITE(' F4'); TEXTCOLOR(YELLOW); WRITE(' RIMP');
 TEXTCOLOR(GREEN); WRITE(' F5'); TEXTCOLOR(YELLOW); WRITE(' PRE');
 TEXTCOLOR(GREEN); WRITE(' F6'); TEXTCOLOR(YELLOW); WRITE(' MAPA');
 TEXTCOLOR(GREEN); WRITE(' F7'); TEXTCOLOR(YELLOW); WRITE(' EXP');
 TEXTCOLOR(GREEN); WRITE(' F8'); TEXTCOLOR(YELLOW); WRITE(' GRAB');
 TEXTCOLOR(GREEN); WRITE(' F9'); TEXTCOLOR(YELLOW); WRITE(' AYU');
 TEXTCOLOR(GREEN); WRITE(' F10'); TEXTCOLOR(YELLOW); WRITE(' FIN');
END;
```

```
PROCEDURE PAUSA(N:WORD);
 VAR
 I:WORD;

  BEGIN
   N:=N*10;
   I:=0;
   REPEAT
    I:=I+1;
    DELAY(100);
   UNTIL KEYPRESSED OR (I>N);
  END;


FUNCTION CONVIER(NUM:INTEGER):STRING;
 VAR
 NUMER:STRING[2];

  BEGIN
   STR(NUM,NUMER);
   IF NUM<10 THEN NUMER:='0' +NUMER;
   CONVIER:= NUMER;
  END;


PROCEDURE LEE_FECHA(VAR D,ME:STRING; VAR ANIO:STRING; VAR H,MI:STRING);
(* SE LEE LA FECHA DEL SISTEMA *)

 VAR
  RELOJ:REGISTERS;
  ANO,MES,DIA,DIS:WORD;
  HOR,MIN,SEG,CES:WORD;

  BEGIN
   GETDATE(ANO,MES,DIA,DIS);
   GETTIME(HOR,MIN,SEG,CES);
   H:=CONVIER(HOR);
   MI:=CONVIER(MIN);
   D:=CONVIER(DIA);
   ME:=CONVIER(MES);
   STR(ANO:4,ANIO);
  END; (* LEE_FECHA *)


PROCEDURE TIEMPO_INICIAL;
(* LEE TIEMPO INICIAL *)

 VAR
  RELOJ:REGISTERS;
  HORA,TOS:INTEGER;
```

```
 BEGIN
  WITH RELOJ DO
  BEGIN
   AX:= $2C00;
   MSDOS(RELOJ);
   HORA:= HI(CX);
   TOS:= LO(CX);
  END;  (* END WITH *)
 END; (* END TIEMPO_INICIAL *)


PROCEDURE RELOJ_PANTALLA;
(* SE ESCRIBE LA FECHA Y LA HORA EN LA PANTALLA *)

 VAR
 HR,MIN,DIA,MES:  STRING;
 ANIO:        STRING;
 CADENAFECHA:    STRING;
 CADENAHORA:     STRING;
 FECHA_HORA: STRING;
 X:WORD;

 BEGIN
  CADENAHORA:= '';
  CADENAFECHA:= '';
  LEE_FECHA(DIA,MES,ANIO,HR,MIN);
  CADENAHORA:= CONCAT(HR,':',MIN);
  CADENAFECHA:= CONCAT(DIA,'/',MES,'/',ANIO);
  WINDOW(62,1,79,2);
  TEXTBACKGROUND(BLACK);
  TEXTCOLOR(CYAN);
  GOTOXY(1,1);
  WRITE(CADENAFECHA:10,CADENAHORA:8);
 END;


PROCEDURE AUTORES;  (* PRESENTACION DEL SISTEMA EN EL EJECUTABLE *)

 VAR
 COUNT:INTEGER;

 BEGIN
  WINDOW(1,1,80,25);
  TEXTBACKGROUND(BLACK);
  TEXTCOLOR(WHITE);
  CLRSCR;
  WINDOW(25,2,75,25);
  TEXTBACKGROUND(BLUE);
  TEXTCOLOR(WHITE);
  WRITELN('                    ');
  WRITELN('        UNAM        ');
  WRITELN('    E.N.E.P ARAGON  ');
```

```
 WRITELN('                              ');
 WRITELN(' TESIS DE INGENIERIA EN COMPUTACION ');
 WRITELN('                              ');
 WRITELN('   SISTEMA DE TIEMPO REAL PARA     ');
 WRITELN('                              ');
 WRITELN('   LA INFORMACION DE PUNTOS DE     ');
 WRITELN('                              ');
 WRITELN('   OZONO EN  EL  D.F.  Y  AREA     ');
 WRITELN('                              ');
 WRITELN('         METROPOLITANA          ');
 WRITELN('                              ');
 WRITELN('            ALUMNOS             ');
 WRITELN('                              ');
 WRITELN('        MARES GOMEZ ARTURO        ');
 WRITELN('                              ');
 WRITELN('       MONTES MONROY ROBERTO       ');
 WRITELN('                              ');
 WRITELN('  ASESOR: ING. JUAN GASTALDI PEREZ  ');
 WRITELN('                              ');
 FOR COUNT:= 0 TO 1 DO
  BEGIN
   SOUND(1200);
   DELAY(200);
   NOSOUND;
   SOUND(1600);
   DELAY(200);
   NOSOUND;
  END;
   DELAY(7000);
 END;



PROCEDURE MARCO;

 VAR
  I,J,K,L:INTEGER;

 BEGIN
  WINDOW(1,1,80,25);
  TEXTBACKGROUND(BLACK);
       TEXTCOLOR(WHITE);
       CLRSCR;
  GOTOXY(1,3);
  WRITE(CHR(201));
  GOTOXY(1,23);
  WRITE(CHR(200));
  GOTOXY(79,23);
  WRITE(CHR(188));
  GOTOXY(79,3);
  WRITE(CHR(187));
  FOR I:=0 TO 1 DO
  FOR J:=2 TO 78 DO
   BEGIN
```

```
    K:=3+I*20;
    GOTOXY(J,K);
    WRITE(CHR(205));
   END;
   FOR I:=0 TO 1 DO
   FOR J:=4 TO 22 DO
       BEGIN
        K:=1+I*78;
        GOTOXY(K,J);
        WRITE(CHR(186));
       END;
  END;

(* BLOQUE DE LA AYUDA *)

PROCEDURE AYUDA3;

BEGIN
 WINDOW(1,1,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);
 CLRSCR;
 MARCO;
 WINDOW(4,4,75,20);
 TEXTCOLOR(GREEN);WRITE('  F6');TEXTCOLOR(YELLOW);WRITE(' MAPA');
 TEXTCOLOR(WHITE);
 WRITE(' LA FUNCION MUESTRA UN MAPA DEL D.F. Y LA PARTE NORTE DE LA ');
 WRITELN(' AREA METROPOLITANA ASI COMO LA UBICACION DE CADA UNA DE LAS');
 WRITELN(' ESTACIONES                                    ');
 TEXTCOLOR(GREEN);WRITE('  F7');TEXTCOLOR(YELLOW);WRITE(' EXP');
 TEXTCOLOR(WHITE);
 WRITE('  LA FUNCION EXPLORA LAS ESTACIONES, ES DECIR, PIDE INFORMA-');
 WRITELN(' CION A LAS MISMAS DE LOS PUNTOS DE OZONO QUE ESTAN CAPTAN-');
 WRITELN(' DO EN ESE MOMENTO                             ');
 TEXTCOLOR(GREEN);WRITE('  F8');TEXTCOLOR(YELLOW);WRITE(' GRAB');
 TEXTCOLOR(WHITE);
 WRITE(' LA FUNCION GRABA LA INFORMACION EN ARCHIVO PARA FUTUROS RE-');
 WRITELN('          PORTES                               ');
 TEXTCOLOR(GREEN);WRITE('  F9');TEXTCOLOR(YELLOW);WRITE(' AYU');
 TEXTCOLOR(WHITE);
 WRITE('  DESPLIEGA LA AYUDA DEL SISTEMA                 ');
 WRITELN('                                               ');
 TEXTCOLOR(GREEN);WRITE('  F10');TEXTCOLOR(YELLOW);WRITE(' FIN');
 TEXTCOLOR(WHITE);
 WRITE(' LA FUNCION NOS PERMITE SALIR DEL SISTEMA INDICANDO SI SE');
 WRITELN('            QUIERE O NO GUARDAR LA INFORMACION EN  ARCHIVO');
 READLN;
END;
```

```
PROCEDURE AYUDA2;
 VAR
  OPCION:STRING[2];


BEGIN
 WINDOW(1,1,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);
 CLRSCR;
 MARCO;
 WINDOW(4,4,75,20);
 WRITELN(' EN LA PARTE INFERIOR DE LA PANTALLA PRINCIPAL SE ENCUENTRAN LAS ');
 WRITELN(' TECLAS DE FUNCION LAS CUALES REALIZAN LOS SIGUIENTE :          ');
 WRITELN;
 TEXTCOLOR(GREEN); WRITE('  F1');TEXTCOLOR(YELLOW);WRITE(' RELOJ');
 TEXTCOLOR(WHITE);
 WRITE(' LA FUNCION DEL RELOJ ACTUALIZA LA HORA DEL SISTEMA               ');
 WRITELN;
 TEXTCOLOR(GREEN); WRITE('  F2');TEXTCOLOR(YELLOW);WRITE(' LOCAL');
 TEXTCOLOR(WHITE);
 WRITE(' LA FUNCION DE LOCAL NOS PROPORCIONA LAS DIRECCIONES DE CADA ');
 WRITELN('          UNA DE LAS ESTACIONES DEL SISTEMA');
 WRITELN;
 TEXTCOLOR(GREEN);WRITE('  F3');TEXTCOLOR(YELLOW);WRITE(' RPAN');
 TEXTCOLOR(WHITE);
 WRITELN(' LA FUNCION DE RPAN NOS PRESENTA UN REPORTE GENERAL POR PAN-');
 WRITELN('  TALLA DE LOS PUNTOS DE OZONO QUE S TUVIERON O SIERON O SE TIENEN SE-');
 WRITELN('  GUN EL DIA ESPECIFICADO DEL CUAL SE REQUERIDO EL REPORTE');
 WRITELN;
 TEXTCOLOR(GREEN);WRITE('  F4');TEXTCOLOR(YELLOW);WRITE(' RIMP');
 TEXTCOLOR(WHITE);
 WRITE('  LA FUNCION DE ESTA TECLA ES SEMEJANTE A F3 SOLAMENTE QUE ES ');
 WRITE('  IMPRESO EL REPORTE                                      ');
 WRITELN('     ');
 TEXTCOLOR(GREEN);WRITE('  F5');TEXTCOLOR(YELLOW);WRITE(' PRE');
 TEXTCOLOR(WHITE);
 WRITE(' LA FUNCION DE PRE ES UNA PRESENTACION DONDE SE ENCUENTRAN ');
 WRITE(' LOS AUTORES DE LA TESIS ASI COMO EL ASESOR DE LA MISMA          ');
 READLN;

END;




PROCEDURE AYUDA;
VAR
 OPCION:STRING[2];
BEGIN
 MARCO;
 WINDOW(4,4,76,20);
 TEXTBACKGROUND(CYAN);
```

```
TEXTCOLOR(WHITE);
WRITELN('                    AYUDA                    ');
TEXTBACKGROUND(BLACK);
WRITELN('EL SISTEMA DE TIEMPO REAL IMECA/STR PROPORCIONA LA CANTIDAD DE PUNTOS');
WRITELN('DE OZONO QUE RECOPILA DE LAS DIFERENTES ESTACIONES EN EL D.F. Y AREA ');
WRITELN('METROPOLITANA.                                    ');
WRITELN('LA PANTALLA PRINCIPAL DEL SISTEMA MUESTRA LAS ESTACIONES CON SUS RES-');
WRITELN('PECTIVOS NUMEROS Y MUESTRA UNA VENTANA SEMEJANTEA LA SIGUIENTE :   ');
WRITELN('                                                  ');
WINDOW(22,12,37,18);
TEXTBACKGROUND(GREEN);
TEXTCOLOR(WHITE);
WRITELN('NIVEL MAX  125');
WRITELN('           ');
WRITELN('NIVEL INS   75');
WRITELN('           ');
WRITELN('NIVEL MIN   10');
WINDOW(47,12,65,12);
TEXTBACKGROUND(GREEN);
TEXTCOLOR(WHITE);
WRITE('FUNCIONANDO');
WINDOW(47,14,65,14);
TEXTBACKGROUND(RED);
WRITE('FALLA');
WINDOW(47,16,65,16);
TEXTBACKGROUND(CYAN);
WRITE('NIVEL ALTO');
WINDOW(4,18,76,21);
TEXTBACKGROUND(BLACK);
WRITELN('EN LA VENTANA DE LA ESTACION NOS MUESTRA EL NIVEL MAXIMO PERMITIDO ');
WRITELN('O RESPIRABLE, EL NIVEL INSTANTANEO ES DECIR, EL NIVEL MEDIDO EN ESE');
WRITELN('MOMENTO Y EL NIVEL MINIMO                         ');

END;



PROCEDURE GUARDAR; (*GUARDA LA INFORMACION EN ARCHIVO*)

 VAR
  REP:TEXT;
  FECHA,DATE:STRING;

  BEGIN
   WINDOW(1,1,80,25);
   TEXTBACKGROUND(BLACK);
   TEXTCOLOR(WHITE);
   CLRSCR;
   MARCO;
   WINDOW(23,4,60,17);
   TEXTBACKGROUND(BLUE);
   TEXTCOLOR(WHITE);
   WRITELN('                    ');
```

```
 WRITELN('PARA  GUARDAR EL ARCHIVO  DEL REPORTE');
 WRITELN('                                    ');
 WRITELN('TECLEE  EL  ARCHIVO  DESEADO  DE  LA ');
 WRITELN('                                    ');
 WRITELN('SIGUIENTE  FORMA  DIA, MES, AYO  EJ: ');
 WRITELN('                                    ');
 WRITELN('FECHA 16 DE  SEPTIEMBRE  DE 1994  SE ');
 WRITELN('                                    ');
 WRITELN('TECLEEA:1691994 16=DIA 9=MES 1994=AYO');
 WRITELN('                                    ');
 WRITELN('TECLEE LA FECHA DEL ARCHIVO A GUARDAR');
 WRITELN('                                    ');
 WINDOW(23,18,35,19);
 READLN(DATE);
 FECHA:=DATE+'.DAT';
 WINDOW(30,20,50,21);
 TEXTBACKGROUND(CYAN);
 TEXTCOLOR(WHITE);
 WRITE('GUARDANDO INFORMACION');
 SOUND(1000);
 DELAY(800);
 NOSOUND;
 ASSIGN(REP,FECHA);
 REWRITE(REP);
 APPEND(REP);
 WRITELN(REP,EN);
 APPEND(REP);
 WRITELN(REP,EC);
 APPEND(REP);
 WRITELN(REP,VE);
 APPEND(REP);
 WRITELN(REP,CU);
 APPEND(REP);
 WRITELN(REP,GU);
 APPEND(REP);
 WRITELN(REP,NE);
 CLOSE(REP);
END;


PROCEDURE TIMER;

 VAR
 HORA,MINUTO,SEGUNDO,SEC100:INTEGER;

 BEGIN
 MARCO;
 TEXTBACKGROUND(BLUE);
 TEXTCOLOR(WHITE);
 GOTOXY(32,5);
 WRITELN('SISTEMA IMECA/STR');
 GOTOXY(9,7);
 WRITELN('TECLEE LA HORA OPRIMA RETURN, ENSEGUIDA APARECERA EL LETRERO DE ');
```

```
 GOTOXY(9,8);
 WRITELN('TECLEE LOS MINUTOS, TECLEE EL DATO Y OPRIMA RETURN          ');
 WINDOW(32,10,60,15);
 TEXTBACKGROUND(CYAN);
 TEXTCOLOR(WHITE);
 WRITELN('TECLEE LA  HORA ');
 READLN(HORA);
 WRITELN('TECLEE LOS MINUTOS');
 READLN(MINUTO);
 SEGUNDO:=0;
 SEC100:=0;
 SETTIME(HORA,MINUTO,SEGUNDO,SEC100);
 WINDOW(1,1,80,25);
 GOTOXY(3,21);
 WRITELN('PULSE LA TECLA ENTER PARA CONTINUAR');
END;



PROCEDURE REPORTE;  (*REPORTE POR PANTALLA*)
 VAR
  ARCH:TEXT;
  L,NOMBRE,NOM:STRING;
  TARCH1,ARCH1,ART:TEXT;
  A,N,B,C,D:INTEGER;
  S:STRING[22];
  OPCION:STRING[1];

BEGIN
 WINDOW(1,1,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);
 CLRSCR;
 GOTOXY(25,12);
 TEXTBACKGROUND(BLUE);
 TEXTCOLOR(WHITE);
  BEGIN
   WINDOW(1,1,80,25);
   TEXTBACKGROUND(BLACK);
   TEXTCOLOR(WHITE);
   CLRSCR;
   TEXTBACKGROUND(CYAN);
   TEXTCOLOR(WHITE);
   WRITE('          REPORTE DE IMECAS REGISTRADOS EL DIA          ');
   WRITE('                U.N.A.M                  ');
   WRITE('              E.N.E.P  ARAGON          ');
   WRITE('          INGENIERIA  EN  COMPUTACION          ');
   GOTOXY(1,7);
   WRITE('NOMBRE DE LA ESTACION          CALIDAD      NIVEL DE OZONO  ');
   WINDOW(25,12,65,15);
   TEXTBACKGROUND(BLUE);
   TEXTCOLOR(WHITE);
   WRITELN('POR FAVOR TECLEE LA FECHA DEL ');
```

```
WRITELN('REPORTE QUE DESEA OBSERVAR   ');
WRITELN('                             ');
READLN(NOMBRE);
 BEGIN
  WINDOW(1,8,80,23);
  TEXTBACKGROUND(BLACK);
  TEXTCOLOR(WHITE);
  CLRSCR;
  TEXTBACKGROUND(CYAN);
  TEXTCOLOR(BLACK);
  NOM:=NOMBRE+'.DAT';
  ASSIGN(ARCH,NOM);
  {$I-}
  RESET(ARCH);
  IF IORESULT <> 0 THEN
   BEGIN
    WINDOW(1,1,80,24);
    TEXTBACKGROUND(BLACK);
    TEXTCOLOR(WHITE);
    CLRSCR;
    WINDOW(28,12,65,14);
    TEXTBACKGROUND(BLUE);
    TEXTCOLOR(WHITE);
    SOUND(1000);
    WRITELN('ARCHIVO NO ENCONTRADO ');
    DELAY(500);
    WINDOW(25,25,65,25);
    NOSOUND;
    TEXTBACKGROUND(CYAN);
    TEXTCOLOR(WHITE);
    WRITE('PULSE LA TECLA ENTER PARA RETORNAR');
    EXIT;
    END;
          BEGIN
          B:=1;
          WHILE NOT EOF(ARCH)  DO
           BEGIN
           FOR N:=1 TO 6 DO
           A:=69;
           B:=B+2;
           GOTOXY(A,B);
           READLN(ARCH,L);
           WRITE(L);
           END;

          B:=1;
          ASSIGN(TARCH1,'EST.DAT');
          RESET(TARCH1);
          TEXTCOLOR(BLACK);
          WHILE NOT EOF(TARCH1) DO
           BEGIN
           FOR N:=1 TO 6 DO
           A:=1;
           B:=B+2;
```

```
          GOTOXY(A,B);
          READLN(TARCH1,S);
          WRITE(S);
         END;
          BEGIN
           B:=1;
           ASSIGN(ARCH1,'REPORT.DAT');
           RESET(ARCH1);
           WHILE NOT EOF(ARCH1) DO
              BEGIN
               FOR N:=1 TO 6 DO
               A:=47;
               B:=B+2;
               GOTOXY(A,B);
               READLN(ARCH1,S);
               WRITE(S);
              END;
               WINDOW(70,1,80,2);
               WRITELN(NOMBRE);
               WINDOW(20,25,65,25);
               TEXTCOLOR(WHITE);
               WRITE('PULSE LA TECLA ENTER PARA RETORNAR');
          END;
  END;
 END;
END;
END;


PROCEDURE REPIMP;  (* REPORTE IMPRESO *)

 VAR
  A1,A2,DISCO,IMPRIME:TEXT;
  S1,S2,S3,S5,NOM,NOMBRE:STRING;

BEGIN
 WINDOW(1,1,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);
 CLRSCR;
 MARCO;
 WINDOW(25,12,65,15);
 TEXTBACKGROUND(BLUE);
 TEXTCOLOR(WHITE);
 WRITELN('POR FAVOR TECLEE LA FECHA DEL ');
 WRITELN('REPORTE QUE DESEA IMPRIMIR   ');
 WRITELN('                          ');
 WINDOW(25,15,33,15);
 READLN(NOMBRE);
 NOM:=NOMBRE+'.DAT';
 ASSIGN(A1,'ESTIMP.DAT');
 RESET(A1);
 ASSIGN(A2,NOM);
```

```
RESET(A2);
IF IORESULT <> 0 THEN
 BEGIN
  WINDOW(1,1,80,24);
  TEXTBACKGROUND(BLACK);
  TEXTCOLOR(WHITE);
  CLRSCR;
  WINDOW(28,12,65,14);
  TEXTBACKGROUND(BLUE);
  TEXTCOLOR(WHITE);
  SOUND(1000);
  WRITELN('ARCHIVO NO ENCONTRADO ');
  DELAY(500);
  WINDOW(25,25,65,25);
  NOSOUND;
  TEXTBACKGROUND(CYAN);
  TEXTCOLOR(WHITE);
  WRITE('PULSE LA TECLA ENTER PARA RETORNAR');
  EXIT;
 END;
   ASSIGN(DISCO,'REPORTE.DAT');
   REWRITE(DISCO);
   WHILE NOT EOF(A1) DO
    WHILE NOT EOF(A2) DO
     BEGIN
        READLN(A1,S1);
        READLN(A2,S2);
        S3:=CONCAT(S1,'          ',S2);
        WRITELN(DISCO,S3);
     END;
    CLOSE(A2);
   CLOSE(A1);
  CLOSE(DISCO);
BEGIN
 WRITELN(LST,'          REPORTE DE IMECAS REGISTRADOS EL DIA  ', NOMBRE  );
 WRITELN(LST,'                U.N.A.M');
 WRITELN(LST,'              E.N.E.P  ARAGON');
 WRITELN(LST,'            INGENIERIA  EN  COMPUTACION');
 WRITELN(LST);
 WRITELN(LST,'NOMBRE DE LA ESTACION      LOCALIDAD    NIVEL DE OZONO');
 WRITELN(LST);
 ASSIGN(IMPRIME,'REPORTE.DAT');
 RESET(IMPRIME);
 WHILE NOT EOF(IMPRIME) DO
  BEGIN
  READLN(IMPRIME,S5);
  WRITELN(LST,S5);
 END;
  CLOSE(IMPRIME);

 END;
END;
```

```
PROCEDURE MAPA; (* MAPA DEL D.F. Y AREA METROPOLITANA*)

VAR
 CONTROLGR,MODOGR,CX,CY,CZ,CW,I,A,B,C,D:INTEGER;
 TARCH2:TEXT;

 BEGIN
  CLRSCR;
  TEXTBACKGROUND(BLACK);
  TEXTCOLOR(WHITE);
  CONTROLGR:=DETECT;
  INITGRAPH(CONTROLGR,MODOGR,'');
  ASSIGN(TARCH2,'DF.DAT');
  RESET(TARCH2);
  I:=0;
  WHILE NOT EOF(TARCH2) DO
   BEGIN
    READLN(TARCH2,CX,CY,CZ,CW);
    A:=CX;
    B:=CY;
    C:=CZ;
    D:=CW;
    LINE(A,B,C,D);
   END;
    BEGIN
        ASSIGN(TARCH2,'EDOMEX.DAT');
        RESET(TARCH2);
        WHILE NOT EOF(TARCH2) DO
        BEGIN
         SETCOLOR(LIGHTGRAY);
         READLN(TARCH2,CX,CY,CZ,CW);
         A:=CX;
         B:=CY;
         C:=CZ;
         D:=CW;
         LINE(A,B,C,D);
        END;

        BEGIN
         SETFILLSTYLE(1,GREEN);
         IF EC=0 THEN
         SETFILLSTYLE(1,RED);
         IF EC>135 THEN
         SETFILLSTYLE(1,CYAN);
         BAR(477,30,481,34);

         SETFILLSTYLE(1,GREEN);
         IF EN=0 THEN
         SETFILLSTYLE(1,RED);
         IF EN>135 THEN
         SETFILLSTYLE(1,CYAN);
         BAR(460,120,464,124);
```

```
SETFILLSTYLE(1,GREEN);
IF NE=0 THEN
SETFILLSTYLE(1,RED);
IF NE>135 THEN
SETFILLSTYLE(1,CYAN);
BAR(516,175,520,179);

SETFILLSTYLE(1,GREEN);
IF VE=0 THEN
SETFILLSTYLE(1,RED);
IF VE>135 THEN
SETFILLSTYLE(1,CYAN);
BAR(357,98,361,102);

SETFILLSTYLE(1,GREEN);
IF CU=0 THEN
SETFILLSTYLE(1,RED);
IF CU>135 THEN
SETFILLSTYLE(1,CYAN);
BAR(323,140,327,144);

SETFILLSTYLE(1,GREEN);
IF GU=0 THEN
SETFILLSTYLE(1,RED);
IF GU>135 THEN
SETFILLSTYLE(1,CYAN);
BAR(351,30,355,34);


END;

BEGIN
SETTEXTSTYLE(0,0,1);
SETCOLOR(14);
MOVETO(476,21);
OUTTEXT('2');
MOVETO(459,111);
OUTTEXT('1');
MOVETO(515,166);
OUTTEXT('6');
MOVETO(350,21);
OUTTEXT('5');
MOVETO(356,90);
OUTTEXT('3');
MOVETO(322,131);
OUTTEXT('4');
END;
SETFILLSTYLE(1,YELLOW);
BAR(25,415,115,418);
BAR(69,371,72,459);
SETCOLOR(4);
SETTEXTSTYLE(0,0,2);
MOVETO(8,411);
OUTTEXT('O');
```

```
        MOVETO(120,411);
        OUTTEXT('E');
        MOVETO(65,356);
        OUTTEXT('N');
        MOVETO(65,464);
        OUTTEXT('S');

  END;

BEGIN
 SETUSERCHARSIZE(4,3,4,3);
 SETTEXTSTYLE(TRIPLEXFONT,HORIZDIR,1);
 SETCOLOR(11);
 MOVETO(02,011);
 SETTEXTSTYLE(0,0,1);
 MOVETO(55,01);
 OUTTEXT('RED DE MONITOREO DE PUNTOS DE OZONO EN EL D.F. Y AREA METROPOLITANA ');
 SETCOLOR(BLUE);
 LINE(15,35,180,35);
 LINE(15,35,15,125);
 LINE(180,35,180,125);
 LINE(15,125,180,125);
 SETFILLSTYLE(1,GREEN);
 BAR(25,44,29,48);
 SETCOLOR(WHITE);
 MOVETO(33,44);
 OUTTEXT('FUNCIONANDO NIVEL');
 MOVETO(33,53);
 OUTTEXT('NORMAL DE OZONO');
 SETFILLSTYLE(1,CYAN);
 BAR(25,75,29,79);
 MOVETO(33,75);
 OUTTEXT('FUNCIONANDO NIVEL');
 MOVETO(33,84);
 OUTTEXT('ALTO DE OZONO');
 SETFILLSTYLE(1,RED);
 BAR(25,112,29,117);
 MOVETO(33,112);
 OUTTEXT('ESTACION EN FALLA');
 END;
        FLUSH(TARCH2);
        CLOSE(TARCH2);
 END;


PROCEDURE EXPLO;  (* EXPLORA LAS ESTACIONES *)

 VAR
 COUNT:INTEGER;

 BEGIN
  WINDOW(1,1,80,25);
  TEXTBACKGROUND(BLACK);
  TEXTCOLOR(WHITE);
```

```
 CLRSCR;
 TEXTBACKGROUND(CYAN);
 WINDOW(23,4,56,5);
 WRITE('RECIBIENDO NUMERO DE LA ESTACION');
 READLN;
 EST:=PORT[576];
 TEXTBACKGROUND(CYAN);
 WINDOW(35,6,45,6);
 WRITE(EST);
 WINDOW(18,8,71,10);
 WRITE('RECIBIENDO INFORMACION DE PUNTOS DE OZONO');
 READLN;
 DATO:=PORT[576];
 WINDOW(35,11,40,11);
 WRITE(      DATO);
 READLN;
 DATO:=DATO*1;
END;


PROCEDURE INICIO; (* PANTALLA PRINCIPAL 1 *)

 VAR
 X,Y,I,Z,V,J,K,L,COUNT:INTEGER;
 TARCH,ARCH:TEXT;
 R,S:STRING[20];

  BEGIN
   TEXTBACKGROUND(GREEN);
   IF EN=0 THEN
   TEXTBACKGROUND(RED);
   IF EN>130 THEN
   TEXTBACKGROUND(CYAN);
   WINDOW(5,5,20,15);
   TEXTCOLOR(WHITE);
   WRITELN('NIVEL MAX  125');
   WRITELN('           ');
   WRITELN('NIVEL INS  ', EN);
   WRITELN('           ');
   WRITELN('NIVEL MIN   10');
   WRITELN('           ');

   TEXTBACKGROUND(GREEN);
   IF EC=0 THEN
   TEXTBACKGROUND(RED);
   IF EC>130 THEN
   TEXTBACKGROUND(CYAN);
   WINDOW(31,5,46,15);
   TEXTCOLOR(WHITE);
   WRITELN('NIVEL MAX  125');
   WRITELN('           ');
   WRITELN('NIVEL INS  ', EC);
   WRITELN('           ');
```

```
WRITELN('NIVEL MIN   10');
WRITELN('            ');

TEXTBACKGROUND(GREEN);
IF VE=0 THEN
TEXTBACKGROUND(RED);
IF VE>130 THEN
TEXTBACKGROUND(CYAN);
WINDOW(57,5,72,15);
TEXTCOLOR(WHITE);
WRITELN('NIVEL MAX  125');
WRITELN('            ');
WRITELN('NIVEL INS  ', VE);
WRITELN('            ');
WRITELN('NIVEL MIN   10');
WRITELN('            ');

TEXTBACKGROUND(GREEN);
IF CU=0 THEN
TEXTBACKGROUND(RED);
IF CU>130 THEN
TEXTBACKGROUND(CYAN);
WINDOW(5,15,20,23);
TEXTCOLOR(WHITE);
WRITELN('NIVEL MAX  125');
WRITELN('            ');
WRITELN('NIVEL INS  ', CU);
WRITELN('            ');
WRITELN('NIVEL MIN   10');
WRITELN('            ');

TEXTBACKGROUND(GREEN);
IF GU=0 THEN
TEXTBACKGROUND(RED);
IF GU>130 THEN
TEXTBACKGROUND(CYAN);
WINDOW(31,15,46,23);
TEXTCOLOR(WHITE);
WRITELN('NIVEL MAX  125');
WRITELN('            ');
WRITELN('NIVEL INS  ', GU);
WRITELN('            ');
WRITELN('NIVEL MIN   10');
WRITELN('            ');

TEXTBACKGROUND(GREEN);
IF NE=0 THEN
TEXTBACKGROUND(RED);
IF NE>130 THEN
TEXTBACKGROUND(CYAN);
WINDOW(57,15,72,23);
TEXTCOLOR(WHITE);
WRITELN('NIVEL MAX  125');
WRITELN('            ');
```

```
WRITELN('NIVEL INS  ', NE);
WRITELN('           ');
WRITELN('NIVEL MIN   10');
WRITELN('           ');

WINDOW(1,1,80,25);
TEXTBACKGROUND(BLACK);
TEXTCOLOR(WHITE);
GOTOXY(1,3);
WRITE(CHR(201));
GOTOXY(1,23);
WRITE(CHR(200));
GOTOXY(79,23);
WRITE(CHR(188));
GOTOXY(79,3);
WRITE(CHR(187));
FOR I:=0 TO 2 DO
FOR J:=2 TO 78 DO
 BEGIN
     K:=3+I*10;
     GOTOXY(J,K);
     WRITE(CHR(205));
 END;
FOR I:=0 TO 3 DO
FOR J:=4 TO 22 DO
 BEGIN
     K:=1+I*26;
     GOTOXY(K,J);
     WRITE(CHR(186));
 END;
GOTOXY(27,3);
WRITE(CHR(203));
GOTOXY(53,3);
WRITE(CHR(203));
GOTOXY(27,23);
WRITE(CHR(202));
GOTOXY(53,23);
WRITE(CHR(202));
TEXTBACKGROUND(BLACK);
TEXTCOLOR(GREEN);
GOTOXY(1,1);
WRITE('FUNC. þ');
TEXTCOLOR(RED);
GOTOXY(1,2);
WRITE('FALLA þ');
TEXTCOLOR(CYAN);
GOTOXY(12,1);
WRITE('N. ALTO þ');
TEXTBACKGROUND(BLUE);
TEXTCOLOR(WHITE);
GOTOXY(30,1);
WRITE('SISTEMA DE MONITOREO');
GOTOXY(35,2);
WRITE('IMECA/STR');
```

```
 ASSIGN(TARCH,'ESTACION.DAT');
 RESET(TARCH);
 WHILE NOT EOF(TARCH) DO
  BEGIN
      TEXTCOLOR(YELLOW);
      GOTOXY(3,14);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(3,4);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(29,14);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(29,4);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(55,14);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(55,4);
      READLN(TARCH,S);
      WRITE(S);
  END;
 FLUSH(TARCH);
 CLOSE(TARCH);
 FUNCIONES;
 RELOJ_PANTALLA
END;



PROCEDURE INICIO1; (* PANTALLA PRINCIPAL 2 *)

 VAR
 X,Y,I,Z,V,J,K,L,COUNT:INTEGER;
 TARCH,ARCH:TEXT;
 R,S:STRING[20];

 BEGIN
  IF EST=1 THEN
  BEGIN
  EN:=DATO;
  WINDOW(5,5,20,15);
  IF EN=0 THEN
  TEXTBACKGROUND(RED) ;
  IF EN>130 THEN
  TEXTBACKGROUND(CYAN)
  ELSE
  TEXTBACKGROUND(GREEN);
  TEXTCOLOR(WHITE);
  WRITELN('NIVEL MAX   125');
  WRITELN('              ');
```

```
 WRITELN('NIVEL INS   ', EN);
 WRITELN('            ');
 WRITELN('NIVEL MIN   10');
 WRITELN('            ');
END;


 IF EST=2 THEN
 BEGIN
 EC:=DATO;
 IF EC=0 THEN
 TEXTBACKGROUND(RED) ;
 IF EC>130 THEN
      TEXTBACKGROUND(CYAN)
 ELSE
 TEXTBACKGROUND(GREEN);
 TEXTCOLOR(WHITE);

 WINDOW(31,5,46,15);
 WRITELN('NIVEL MAX   125');
 WRITELN('            ');
 WRITELN('NIVEL INS   ', EC);
 WRITELN('            ');
 WRITELN('NIVEL MIN   10');
 WRITELN('            ');
END;



 IF EST=3 THEN
 BEGIN
 VE:=DATO;
 IF VE=0 THEN
 TEXTBACKGROUND(RED) ;
 IF VE>130 THEN
      TEXTBACKGROUND(CYAN)
 ELSE
 TEXTBACKGROUND(GREEN);
 TEXTCOLOR(WHITE);
 WINDOW(57,5,72,15);
 WRITELN('NIVEL MAX   125');
 WRITELN('            ');
 WRITELN('NIVEL INS   ', VE);
 WRITELN('            ');
 WRITELN('NIVEL MIN   10');
 WRITELN('            ');
END;


 IF EST=4 THEN
 BEGIN
 CU:=DATO;
 IF CU=0 THEN
 TEXTBACKGROUND(RED) ;
```

```
IF CU>130 THEN
    TEXTBACKGROUND(CYAN)
ELSE
TEXTBACKGROUND(GREEN);
TEXTCOLOR(WHITE);
WINDOW(5,15,20,23);
WRITELN('NIVEL MAX  125');
WRITELN('           ');
WRITELN('NIVEL INS   ', CU);
WRITELN('           ');
WRITELN('NIVEL MIN   10');
WRITELN('           ');
END;


IF EST=5 THEN
BEGIN
GU:=DATO;
IF GU=0 THEN
TEXTBACKGROUND(RED) ;
IF GU>130 THEN
    TEXTBACKGROUND(CYAN)
ELSE
TEXTBACKGROUND(GREEN);
TEXTCOLOR(WHITE);
WINDOW(31,15,46,23);
WRITELN('NIVEL MAX  125');
WRITELN('           ');
WRITELN('NIVEL INS   ', GU);
WRITELN('           ');
WRITELN('NIVEL MIN   10');
WRITELN('           ');
END;

IF EST=6 THEN
BEGIN
NE:=DATO;
IF NE=0 THEN
TEXTBACKGROUND(RED) ;
IF NE>130 THEN
    TEXTBACKGROUND(CYAN)
ELSE
TEXTBACKGROUND(GREEN);
TEXTCOLOR(WHITE);
WINDOW(57,15,72,23);
WRITELN('NIVEL MAX  125');
WRITELN('           ');
WRITELN('NIVEL INS   ', NE);
WRITELN('           ');
WRITELN('NIVEL MIN   10');
WRITELN('           ');
END;
```

```
WINDOW(1,1,80,25);
TEXTBACKGROUND(BLACK);
TEXTCOLOR(WHITE);
GOTOXY(1,3);
WRITE(CHR(201));
GOTOXY(1,23);
WRITE(CHR(200));
GOTOXY(79,23);
WRITE(CHR(188));
GOTOXY(79,3);
WRITE(CHR(187));
FOR I:=0 TO 2 DO
FOR J:=2 TO 78 DO
 BEGIN
     K:=3+I*10;
     GOTOXY(J,K);
     WRITE(CHR(205));
 END;
FOR I:=0 TO 3 DO
FOR J:=4 TO 22 DO
 BEGIN
     K:=1+I*26;
     GOTOXY(K,J);
     WRITE(CHR(186));
 END;
GOTOXY(27,3);
WRITE(CHR(203));
GOTOXY(53,3);
WRITE(CHR(203));
GOTOXY(27,23);
WRITE(CHR(202));
GOTOXY(53,23);
WRITE(CHR(202));
TEXTBACKGROUND(BLACK);
TEXTCOLOR(GREEN);
GOTOXY(1,1);
WRITE('FUNC. þ');
TEXTCOLOR(RED);
GOTOXY(1,2);
WRITE('FALLA þ');
TEXTCOLOR(CYAN);
GOTOXY(12,1);
WRITE('N. ALTO þ');
TEXTBACKGROUND(BLUE);
TEXTCOLOR(WHITE);
GOTOXY(30,1);
WRITE('SISTEMA DE MONITOREO');
GOTOXY(35,2);
WRITE('IMECA/STR');
ASSIGN(TARCH,'ESTACION.DAT');
RESET(TARCH);
WHILE NOT EOF(TARCH) DO
 BEGIN
```

```
      TEXTCOLOR(YELLOW);
      GOTOXY(3,14);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(3,4);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(29,14);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(29,4);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(55,14);
      READLN(TARCH,S);
      WRITE(S);
      GOTOXY(55,4);
      READLN(TARCH,S);
      WRITE(S);
   END;
  FLUSH(TARCH);
  CLOSE(TARCH);
  FUNCIONES;
  RELOJ_PANTALLA
 END;



PROCEDURE LOCALIZA;  (* DIRECCIONES DE LAS ESTACIONES *)

 VAR
 LOCAL:TEXT;
 F:STRING;
 A,B,C:INTEGER;

  BEGIN
   CLRSCR;
   TEXTBACKGROUND(BLUE);
   TEXTCOLOR(WHITE);
   ASSIGN(LOCAL,'DIRECTO.DAT');
   RESET(LOCAL);
   WHILE NOT EOF(LOCAL) DO
    BEGIN
     READLN(LOCAL,F);
     WRITE(F);
    END;
   CLOSE(LOCAL);
  END;
```

```
PROCEDURE TECLAS;

 VAR
  A:INTEGER;
  ARTU:INTEGER;
  TARCH,ART,TXT,ARCH:TEXT;
  TF:BOOLEAN;
  CAR:CHAR;
  OPCION:STRING[2];

  BEGIN
   GOTOXY(70,22);
   TF:= FALSE;
   CAR:=READKEY;
   IF CAR<> #0 THEN
   TECLAS;
   IF CAR= #0 THEN
    BEGIN
     TF:=TRUE;
     CAR:= READKEY;
     ARTU:=ORD(CAR);
     CASE ARTU OF

{F1}    59:BEGIN
             WINDOW(1,1,80,25);
             TEXTBACKGROUND(BLACK);
             TEXTCOLOR(WHITE);
             CLRSCR;
             SOUND(1000);
             DELAY(100);
             NOSOUND;
             TIMER ;
             READLN;
             WINDOW(1,1,80,25);
             TEXTBACKGROUND(BLACK);
             TEXTCOLOR(WHITE);
             CLRSCR;
             INICIO;
             TECLAS;
            END;

{F2}    60:BEGIN
             SOUND(1000);
             DELAY(100);
             NOSOUND;
             WINDOW(1,1,80,25);
             TEXTBACKGROUND(BLACK);
             TEXTCOLOR(WHITE);
             CLRSCR;
             LOCALIZA;
             READLN;
             WINDOW(1,1,80,25);
             TEXTBACKGROUND(BLACK);
             TEXTCOLOR(WHITE);
```

```
            CLRSCR;
            INICIO;
            TECLAS;
            END;

{F3}    61:BEGIN
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            WINDOW(1,1,80,25);
            REPORTE;
            READLN;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO;
            TECLAS;
            END;

{F4}    62:BEGIN
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            REPIMP;
            READLN;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO;
            TECLAS;
            END;

{F5}    63:BEGIN
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            AUTORES;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO;
            TECLAS;
            END;

{F6}    64:BEGIN
```

```
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            RELOJ_PANTALLA;
            MAPA;
            READLN;
            CLOSEGRAPH;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO;
            TECLAS;
           END;

{F7}   65:BEGIN
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            TEXTBACKGROUND(BLUE);
            TEXTCOLOR(WHITE);
            WINDOW(30,12,60,13);
            WRITELN('EXPLORANDO ESTACIONES');
            SOUND(800);
            DELAY(1500);
            NOSOUND;
            MARCO;
            EXPLO;
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO1;
            DELAY(500);
            INICIO;
            TECLAS;
           END;

{F8}   66:BEGIN
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
```

```
            CLRSCR;
            GUARDAR;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO;
            TECLAS;
           END;


{F9}    67:BEGIN
            SOUND(1000);
            DELAY(100);
            NOSOUND;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            AYUDA;
            WINDOW(3,22,55,22);
            TEXTBACKGROUND(CYAN);
            TEXTCOLOR(WHITE);
            WRITE('DESEA CONTINUAR CON LA AYUDA S/N ');
            READ(OPCION);
            IF OPCION='S' THEN
            AYUDA2
            ELSE
            EXIT;
            TEXTBACKGROUND(CYAN);
            TEXTCOLOR(WHITE);
            WINDOW(3,22,55,22);
            WRITE('DESEA CONTINUAR CON LA AYUDA S/N ');
            READ(OPCION);
            IF OPCION='S' THEN
            AYUDA3;
            WINDOW(3,22,55,22);
            TEXTBACKGROUND(CYAN);
            WRITE('PULSE LA TECLA ENTER PARA CONTINUAR');
            BEGIN
            READLN;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            INICIO;
            TECLAS;
           END;
           END;


{F10}   68:BEGIN
            SOUND(1000);
            DELAY(100);
```

```
            NOSOUND;
            WINDOW(1,1,80,25);
            TEXTBACKGROUND(BLACK);
            TEXTCOLOR(WHITE);
            CLRSCR;
            MARCO;
            TEXTBACKGROUND(BLUE);
            WINDOW(20,10,75,18);
            WRITELN('EN ESTE MOMENTO SE ENCUENTRA A PUNTO DE  ');
            WRITELN('FINALIZAR LA UTILIZACION DEL SISTEMA DE  ');
            WRITELN('TIEMPO REAL IMECA/STR.                   ');
            WRITELN('                                         ');
            WRITELN('DESEA GUARDAR LA INFORMACION EN DISCO S/N ');
            READ(OPCION);
            IF OPCION='S' THEN
            GUARDAR
            ELSE
            HALT;
           END;

    END;
   END;
  END;


(*PROGRAMA PRINCIPAL*)
BEGIN
 PORT[$79]:=136;
 WINDOW(1,1,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);
 CLRSCR;
 EN:=1;
 EC:=1;
 VE:=1;
 CU:=1;
 GU:=1;
 NE:=1;
 WINDOW(1,1,80,25);
 TEXTBACKGROUND(BLACK);
 TEXTCOLOR(WHITE);
 CLRSCR;
 UNAM;
 INICIO;
 FUNCIONES;
 TECLAS;
END.
```

APENDICE B. EJEMPLOS.

EJEMPLO I. DESPLEGADO DE LA TECLA DE FUNCION F1.

```
                    SISTEMA IMECA/STR

TECLEE LA HORA OPRIMA RETURN. ENSEGUIDA APARECERA EL LETRERO DE
TECLEE LOS MINUTOS. TECLEE EL DATO Y OPRIMA RETURN

                    TECLEE LA  HORA
                    18
                    TECLEE LOS MINUTOS
                    00
```

EJEMPLO 2. DESPLEGADO DE LA TECLA DE FUNCION F2.

```
LA ESTACION DE ENEP ARAGON SE LOCALIZA EN LA CALLE DE HACIENDA DE RANCHO
SECO S/N COLONIA BOSQUES DE ARAGON MUNICIPIO DE NEZAHUALCOYOTL EDO. MEX.

LA ESTACION DE ECATEPEC SE LOCALIZA EN LA  AV. VIA MORELOS # 235 COLONIA
CERRO GORDO MUNICIPIO DE ECATEPEC EDO. MEX.

LA ESTACION DE VENUSTIANO CARRANZA SE LOCALIZA ENTRE LA AV. FRANCISCO DEL
PASO Y TRONCOSO  Y AV. FRAY SERVANDO TERESA DE MIER COLONIA JARDIN BALBUE
NA DISTRITO FEDERAL

LA ESTACION DE CUAHTEMOC SE LOCALIZA EN MANUEL GONZALEZ S/N COLONIA EX-HI
PODROMO DE PERALVILLO DISTRITO FEDERAL

LA ESTACION DE GUSTAVO A. MADERO SE LOCALIZA EN CALZADA DE GUADALUPE # 528
COLONIA ESTRELLA DISTRITO FEDERAL

LA ESTACION DE NEZAHUALCOYOTL SE LOCALIZA EN LA AV. RIVA PALACIO S/N COLO-
NIA FEDERAL MUNICIPIO DE NEZAHUALCOYOTL EDO. MEX.

PULSE LA TECLA ENTER PARA CONTINUAR
```

EJEMPLO 3. REPORTE CORRESPONDIENTE A LAS TECLAS F3 Y F4.

REPORTE DE IMECAS REGISTRADOS EL DIA 08121994
U.N.A.M
E.N.E.P ARAGON
INGENIERIA EN COMPUTACION

NOMBRE DE LA ESTACION	LOCALIDAD	NIVEL DE OZONO
1 ENEP ARAGON	EDOMEX	128
2 ECATEPEC	EDOMEX	71
3 VENUSTIANO CARRANZA	D.F.	80
4 CUAUHTEMOC	D.F.	72
5 GUSTAVO A. MADERO	D.F.	84
6 NEZAHUALCOYOTL	EDOMEX	85

PULSE LA TECLA ENTER PARA RETORNAR

EJEMPLO 4. DESPLEGADO DE LA TECLA DE FUNCION F5.

UNAM
E.N.E.P ARAGON

TESIS DE INGENIERIA EN COMPUTACION

SISTEMA DE TIEMPO REAL PARA

LA INFORMACION DE PUNTOS DE

OZONO EN EL D.F. Y AREA

METROPOLITANA

ALUMNOS

MARES GOMEZ ARTURO

MONTES MONROY ROBERTO

ASESOR: ING. JUAN GASTALDI PEREZ

EJEMPLO 5. DESPLEGADO DE LA TECLA DE FUNCION F8.

```
PARA  GUARDAR EL ARCHIVO  DEL REPORTE
TECLEE  EL  ARCHIVO  DESEADO  DE  LA
SIGUIENTE  FORMA  DIA. MES. AÑO  EJ:
FECHA 16 DE  SEPTIEMBRE  DE 1994  SE
TECLEEA:1691994 16=DIA 9=MES 1994=AÑO
TECLEE LA FECHA DEL ARCHIVO A GUARDAR

08121994
```

EJEMPLO 6. DESPLEGADO DE LA TECLA DE FUNCION F9.

AYUDA

EL SISTEMA DE TIEMPO REAL IMECA/STR PROPORCIONA LA CANTIDAD DE PUNTOS DE OZONO QUE RECOPILA DE LAS DIFERENTES ESTACIONES EN EL D.F. Y AREA METROPOLITANA.
LA PANTALLA PRINCIPAL DEL SISTEMA MUESTRA LAS ESTACIONES CON SUS RESPECTIVOS NUMEROS Y MUESTRA UNA VENTANA SEMEJANTEA LA SIGUIENTE :

NIVEL MAX	125	FUNCIONANDO
NIVEL INS	75	FALLA
NIVEL MIN	10	NIVEL ALTO

EN LA VENTANA DE LA ESTACION NOS MUESTRA EL NIVEL MAXIMO PERMITIDO O RESPIRABLE. EL NIVEL INSTANTANEO ES DECIR. EL NIVEL MEDIDO EN ESE MOMENTO Y EL NIVEL MINIMO

DESEA CONTINUAR CON LA AYUDA S/N

BIBLIOGRAFIA.

1. Análisis Estructurado Moderno
   Yourdon Edward
   Prentice Hall
   México 1991

2. Software Engineering
   Reges S. Pressman
   Mc. Graw Hill
   1988

3. Las Computadoras y la Información.
   Mc. Graw Hill
   Orilia S. Lawrence
   1988

4. Computación Aplicada a los Negocios
   Orilia S. Lawrence
   Mc. Graw Hill
   1989

5. Computación en la Ciencias Administrativas
   Sanders H. Donald
   Mc. Graw Hill
   1983

6. Aplicaciones en Tiempo Real: Diseño e Implementación
   Blackman Maurice
   Limusa
   1979

7. Real Time Languages: Design and Development
Young J. Stephen
Horwood
1982

8. Tutorial Hard Real-Time Systems
Stankovic A. John
IEEE Computer Society
1988

9. Design of Real-Time Computer Systems
Martin James
Prentice Hall
1990

10. Real-Time Computer Systems
Heller Philip
Birkhauser
1987

11. Real-Time Systems and Their Programing Languages
Burs Alan
Addison Wesley
1990

12. Turbo Pascal 7
O`Brien K. Stephen
Mc Graw Hill
1993

13. Revista PC Magazine
Noviembre, 1992
PP. 84-87

14. Revista PC Magazine
Marzo, 1994
PP. 86-91