DIVISION DE ESTUDIOS DE POSGRADO
Facultad de Ingeniería

DISEÑO DE UN COMPUTADOR AERONAUTICO TOLERANTE A FALLAS

CARLOS BADILLA CORRALES

TRABAJO

Presentado a la División de Estudios de Posgrado de la

FACULTAD DE INGENIERIA
DE LA
UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

como requisito para obtener
el grado de
MAESTRO EN INGENIERIA

(ELECTRICA)

CIUDAD UNIVERSITARIA

OCTUBRE 1991

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

INDICE

INTRODUCCION

Este trabajo presenta una propuesta de diseño para un computador tolerante a fallas orientado a aplicaciones aeronáuticas.

Dicho computador está concebido para realizar tareas de control y de propósito general a bordo de una aeronave multimisional de control remoto que se encuentran desarrollando en forma conjunta el Instituto Politécnico Nacional (IPN) y la Universidad Nacional Autónoma de México (UNAM).

Esta aeronave será utilizada para realizar vuelos de estudio en el área de percepción remota, esto es, captura de imágenes por diversos medios (video o fotografía), en forma rápida, con relación costo beneficio adecuada y resolución de terreno variable.

Dada la naturaleza de este proyecto, la operación correcta de sus diferentes elementos resulta crítica. Una avería en alguno de ellos, especialmente en los dedicados a funciones de control del avión, podría conducir a problemas tales como obtención de datos incorrectos, o bien, la pérdida del control de la aeronave.

Dadas las medidas de seguridad que requiere contemplar un proyecto de este tipo ante casos de mal funcionamiento, existe un claro interés por incrementar la confiabilidad del sistema. Uno de los métodos que pueden utilizarse para ello consiste en utilizar componentes de alta calidad, buenas técnicas de diseño y rigurosos procedimientos para controlar la calidad de los subsistemas por desarrollar. Tales medidas no solo elevan significativamente el costo del proyecto, sino que no son suficientes para reducir la probabilidad de falla en un grado apropiado [Nels 90], [Lala 85].

Un método alternativo para incrementar la confiabilidad del sistema consiste en incorporar recursos adicionales ("redundancia") en la arquitectura del sistema con el propósito de dotarlo de capacidad de respuesta ante la presencia de fallas.

Este método presenta además la ventaja de que no requiere el uso de partes de alta calidad sino que contempla el uso de componentes estándar, ya que la capacidad para enfrentar fallas reside, no en la calidad de los componentes, sino en las características incorporadas a la arquitectura (redundancia, rutinas de diagnóstico, capacidad de reconfiguración,etc).

El estudio de las técnicas de tolerancia a fallas ha evolucionado continuamente con el computador con la finalidad de soportar la complejidad creciente tanto de las estructuras de los computadores como de las herramientas computacionales. Existe incluso un conjunto de técnicas básicas de detección de error, tales como la "doble verificación", la "suma de verificación" y los "puntos de prueba", que se desarrollaron aún antes de que fuera construido el primer computador digital y que, con ligeras adaptaciones, se siguen utilizando en la mayoría de los computadores actuales [Toy 84].

Adicionalmente debe mencionarse que el éxito de estos sistemas en aplicaciones tales como procesamiento de transacciones, transferencia electrónica de fondos, comunicaciones y control de procesos ha contribuido a despertar el interés por la tolerancia a fallas en el campo de la computación de propósito general.

Debe mencionarse que, aunque los principios fundamentales en que se basa la tolerancia a fallas son conceptualmente simples, en la práctica, el diseño de un sistema computacional para alcanzar las especificaciones de confiabilidad deseadas puede resultar bastante complejo.

Entre las principales dificultades que deben considerarse al diseñar sistemas tolerantes a fallas se encuentra el hecho de que es sumamente problemático caracterizar estadísticamente, de antemano, el tipo y frecuencia de fallas, de la circuitería (hardware) y de la programación (software), que afectarán con mayor frecuencia al sistema. Debido a ello, actualmente los computadores de este tipo se diseñan, y su comportamiento se simula, mediante modelos que representan arquitecturas electrónicas complejas.

Otro de los problemas importantes que deben resolverse al seleccionar e integrar diferentes esquemas de tolerancia a fallas consiste en aislar fallas que se presenten durante la operación del sistema, a fin de que sus efectos permanezcan dentro de los límites del subsistema en que fue generada, y se evite así la falla generalizada del sistema.

De los aspectos antes enunciados puede inferirse que en la realidad no es posible diseñar un sistema 100% confiable. Ningún diseño es capaz de proporcionar un comportamiento tolerante a fallas para todos los posibles escenarios de falla, de modo que al presentarse problemas no previstos, el sistema necesariamente fallará.

Por consiguiente, en la práctica, un adecuado diseño tolerante a fallas no es el que pretenda proteger al sistema de todas las posibles fallas, sino el que contempla aquellas que tengan mayor probabilidad de ocurrencia o cuyos efectos sean especialmente perjudiciales dentro de un conjunto de restricciones dado.

Las consideraciones anteriormente expuestas constituyen el marco o contexto en el que se ha planteado el diseño del computador para la aeronave de control remoto que es objeto de este trabajo.

La incorporación de técnicas que incrementen la confiabidad en un sistema no puede ser el resultado de una decisiones improvisadas. Por el contrario, dicho objetivo solo puede ser el resultado de una labor cuidadosamente planificada y metódicamente desarrollada durante las etapas del diseño y construcción del sistema.

Este trabajo pretende ser ante todo un estudio en el área de la arquitectura de computadores tolerantes a fallas. No aspira a ser la solución única y acabada al diseño del computador aeronáutico. Más bien, pretende constituirse en una base a partir de la cual se identifiquen problemas y se analicen propuestas de solución.

RESUMEN

El comportamiento tolerante a fallas consiste en una serie de atributos de la arquitectura de un sistema computacional que involucra todos los aspectos estructurales y funcionales que hacen posible su operación correcta aún ante la presencia de fallas.

Por consiguiente, el diseño de un computador de este tipo conlleva considerar aspectos relacionados tanto con la parte electrónica ("hardware") como con su programación ("software").

En lo referente a sus alcances, el objeto de estudio de este trabajo se limita al diseño de la estructura electrónica de un computador tolerante a fallas orientado a aplicaciones aeronáuticas. Esto es, incluye la especificación de las características estructurales que han sido incorporadas para apoyar las funciones de tolerancia a fallas.

Las estrategias de la programación dirigidas a incrementar la confiabilidad del computador involucran el diseño del sistema operativo, el cual contempla los algoritmos de diagnóstico de fallas y los de reconfiguración del sistema. Los aspectos relacionados con la programación de este computador caen fuera del alcance de este trabajo.

En lo que se refiere a su estructura, este computador presenta una organización modular que emplea un esquema de redundancia modular triple con detección electrónica ("por hardware") de errores, a nivel del módulo de procesamiento, y de redundancia simple (duplicación), a nivel de los dispositivos periféricos

La arquitectura consta de tres procesadores (que operan en forma concurrente) y de un circuito de votación encargado de comparar las salidas de los procesadores para determinar posibles fallas generadas durante el procesamiento. El circuito de votación mediante el principio de votación mayoritaria.

En el diseño del computador se emplean componentes electrónicos de propósito general con alto nivel de integración; en particular, la unidad de procesamiento está basada en circuitos integrados de la familia Intel 80386.

Además, cada procesador cuenta con un bloque de memoria independiente. Esta característica permite el confinamiento de fallas ya que, en caso de error, éste no es propagado hacia otros procesadores como podría ocurrir si los procesadores utilizaran memoria compartida.

Dado que este computador está orientado a aplicaciones aeronáuticas, no requiere ser equipado con los dispositivos típicos de los sistemas de propósito general (teclado, monitor, impresora, etc.). En vez de ello, el sistema cuenta con tres dispositivos periféricos para la ejecución de sus tareas, los cuales pueden clasificarse en dos tipos: de propósito general y de apoyo a las funciones de tolerancia a fallas. Los periféricos de propósito general son el canal de comunicaciones serie, el reloj de tiempo real y el controlador de interrupciones. Los utilizados para asistir las funciones de tolerancia a fallas son un conjunto de registros mediante los cuales se define la configuración de las diferentes partes, se controla la ejecución y se define el acceso a los ductos ("buses") del sistema.

Para lograr la detección eficaz de fallas mediante la comparación de los datos de salida de los elementos del computador, se utiliza un solo circuito de generador de reloj cuya función consiste en sincronizar la operación de los componentes del sistema.

En sintesis, el computador presentado en este trabajo ha sido formulada siguiendo los principios del diseño modular de sistemas y, además, provista de un conjunto de recursos electrónicos redundantes lo cual permite caracterizarlo como un computador tolerante a fallas.

No obstante, la utilización técnicas de programación que aprovechen apropiadamente los recursos incorporados en infraestructura física diseñada será un factor clave para lograr un comportamiento realmente confiable.

CAPITULO 1

ESTRUCTURAS DE COMPUTADORES TOLERANTES A FALLAS

INTRODUCCION

En este capitulo se expone una breve reseña de la evolución de las arquitecturas de computadores tolerantes a fallas para realización tareas de propósito general construidas hasta inicios de la década de los noventa. Esta descripción se presenta con la finalidad de brindar una panorámica del contexto en el que se encuadra el diseño del sistema reportado en este documento.

1.1. CONCEPTO DE TOLERANCIA A FALLAS

La tolerancia a fallas consiste en una serie de atributos de la arquitectura de un sistema computacional que involucra todos los aspectos estructurales y funcionales que hacen posible el comportamiento defensivo del sistema. [Aviz- 90]

Dichos atributos permiten la sobrevivencia de un sistema al habilitarlo para entregar el servicio esperado, al usuario o a otro sistema, en forma continua, después de que una o más fallas se han manifestado dentro del sistema mediante errores.

Las estructuras de los computadores tolerantes a fallas se distinguen porque contiene un conjunto de elementos redundantes (es decir, no estrictamente necesarios para realizar las tareas de cálculo) que son utilizados como refacciones para sustituir partes defectuosas luego de que las fallas se vuelven activas y causan errores dentro del sistema [Aviz 90].

1.2. TIPOS DE REDUNDANCIA EN SISTEMAS TOLERANTES A FALLAS

La redundancia o provisiones requeridas para tolerar fallas pueden manifestarse en las siguientes formas: [Nels 90]

- Componentes extra
- Programación extra
- Tiempo extra
- Información extra

1.2.1. REDUNDANCIA EN COMPONENTES

Consiste en la inclusión de elementos (componentes o módulos) repetidos en la estructura del sistema que se emplean para apoyar las funciones de tolerancia a fallas (estructura tolerante a fallas).

1.2.2. REDUNDANCIA EN PROGRAMACION

Incluye las copias de un mismo programa y las distintas versiones del conjunto de algoritmos utilizados para lograr la respuesta correcta del sistema aún en presencia de defectos (computación tolerante a fallas).

Abarca además, todos los algoritmos utilizados para implantar los distintos tipos de respuesta a fallas (tales como detección de error, diagnóstico de fallas o recuperación de servicio del sistema).

1.2.3. REDUNDANCIA EN TIEMPO

Involucra ejecuciones extra de un mismo fragmento de programa. Este mecanismo es especialmente útil para la eliminación de fallas transitorias. Su implantación requiere contar con las previones en programación que permitan conservar el estado del sistema (contenido del contador de programa, registros internos y apuntador de pila), hasta un punto anterior a la aparición de la falla.

1.2.4. REDUNDANCIA EN INFORMACION

Consiste en conservar copias de los datos utilizados por el sistema. Requiere contar con dispositvos extra de almacenamiento de datos.

Para el lector interesado en la nomenclatura y diversas generalidades sobre sistemas tolerantes a fallas, se incluye información más detallada en el Apéndice A.

1.3. EVOLUCION DE LAS ARQUITECTURAS TOLERANTES A FALLAS

La evolución de las estrategias de tolerancia a fallas ha sido sumamente influenciada no solo por los avances realizados en la tecnología de fabricación de dispositivos y componentes electrónicos sino también por los adelantos logrados en el campo de la arquitectura de procesadores.

Originalmente, durante la época de la primera generación de computadores (década de los cuarenta) se utilizaron circuitos simples para la detección de error tales como verificadores de paridad y se emplearon arquitecturas monolíticas complejas.

Como consecuencia de los progresos alcanzados en el área de la arquitectura de computadores se ha llegado a la conclusión de que las claves para proveer de gran disponibilidad a un sistema son la redundancia y la modularidad [Gray 86].

La redundancia permite contar con recursos extra en caso de falla de una de las partes; por su parte, el módulo constituye la unidad de error y reemplazo, de modo que una falla no implique error a nivel de todo el sistema (avería) sino que sus efectos se circunscriban al módulo en que se originan.

Uno de los mayores esfuerzos que se han realizado en el campo de la confiabilidad ha sido en la formulación de técnicas y arquitecturas de computadores que permitan apoyar la implantación de funciones de tolerancia a fallas.

Hasta inicios de la década de los noventa se han desarrollado tres tipos fundamentales de estructuras tolerantes a fallas [Siew 90]:

- Estructuras uniprocesador
- Estructuras multicomputador
- Estructuras multiprocesador

1.3.1. ESTRUCTURAS UNIPROCESADOR

Los computadores de este tipo tienen una arquitectura uniprocesador que utilizan varios componentes de un mismo tipo para mejorar la confiabilidad del sistema.

El objetivo pretendido por estos sistemas consiste en incrementar la confiabilidad con un mínimo impacto en las características.

Las principales técnicas de respuesta a fallas provista por estos sistemas consiste en la implantación de métodos de detección y señalización de fallas

Las principales técnicas de respuesta a fallas provistas en uniprocesadores pueden agruparse en las siguientes categorías principales: [Siew 84]

a) Circuitos de detección de error interno (uso de códigos de detección y corrección de error y comprobación de paridad en las trayectorias de señales de datos, direcciones y control)
b) Diagnóstico de fallas (incluyendo programación y microcódigo)
c) Señalización de error (por ejemplo luces, bitácora y despliegue del contenido de memoria)
d) Reintento de ejecución en caso de detección de error

Las estrategias de mantenimiento utilizadas por IBM son representativas de las técnicas de tolerancia a fallas empleadas en uniprocesadores.

Las estrategias utilizadas por IBM han evolucionado desde reproducir el escenario de falla hasta capturar la falla (detectar errores y almacenar el estado del computador) para análisis subsecuentes.

En la acción correctiva seguida por los sistemas 360 y 370 de IBM se identifican 4 etapas: recuperación transparente, un usuario afectado, múltiples usuarios afectados y baja de sistema.

Las características de tolerancia a fallas presentes en los computadores de loc computadores IBM S/360 y S/370 las describen [Drou 71] y [Siew 84]. Las características de la IBM 3081 se discuten en [Boss 82] y [Tend 82].

Las perspectivas históricas sobre los atributos de tolerancia a fallas en los sistemas IBM pueden encontrarse en [Hsia 81] y [Drou 71].

El programa RAMP (Reliability, Availability and Maintainability Program) del computador VAX 8600 es representativo del diseño computacional de propósito contemporáneo general con características de tolerancia a fallas. Una descripción más amplia sobre este sistema puede encontrarse en [Siew 90] y [Bruc 85].

La UNIVAC 1100/60 presenta un conjunto de características de disponibilidad, confiabilidad y mantenibilidad (ARM = Availability, Reliability, and Maintainability) que se fundamentan tanto en características de la programación (software) como de la electrónica (hardware). Las características en detalle de este sistema son presentadas en [Siew 84] y [Bonn 80].

1.3.2. ESTRUCTURAS MULTICOMPUTADOR

Esta clase de sistemas se basan en el uso de varios uniprocesadores iguales. Cada computador tiene su propio sistema operativo y se comunica con los otros por medio de un bus de alta velocidad. Todas las comunicaciones entre computadores se realizan por medio de varias copias (como mínimo emisor y receptor) del sistema operativo. Esta característica hace que aumente considerablemente el costo ("overhead") de cooperación.

El uso de multicomputadores para implantar sistemas tolerantes a fallas ha tenido aceptación gracias a la separación física proporcionada por cada computador [Siew 90].

Este tipo de sistemas abarca todos aquellos cuya configuración puede cambiar dinámicamente en respuesta a una falla o en los cuales la redundancia de enmascaramiento, complementada por detección de errores en línea, permite la reparación en línea.

La reconfiguración puede ser realizada en forma automática por el sistema (reparación en línea) o manualmente (reparación fuera de línea).

La evolución de los sistemas multicomputador con redundancia dinámica presenta las siguientes estructuras principales:

1.3.2.1. SISTEMAS CON COMPUTADOR DE RESPALDO E INTERRUPTORES

Esta fue la técnica de tolerancia a fallas usada hasta aproximadamente 1975. Consistía de dos computadores completos capaces de realizar las mismas tareas: un principal y otro de respaldo.

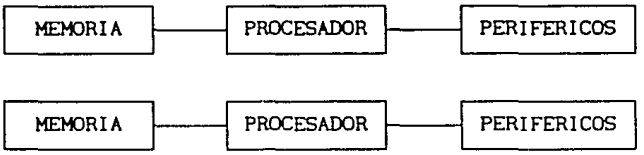


Figura 1.1. Sistema de computador de respaldo con interruptores [Figura 5(a) Siew 84]

Si el computador principal fallaba, el trabajo completo era pasado al computador de respaldo en unos "pocos" minutos. Cada computador podía ser reparado mientras el trabajo útil era realizado por el otro.

1.3.2.2. SISTEMAS CON COMPUTADOR DE RESPALDO Y CONMUTADOR DE PERIFERICO

Estos sistemas permitían conectar los dispositivos periféricos a cualquiera de los computadores en operación.

Este tipo de sistemas presentaba la ventaja de reducir las tareas del operador necesarias para transferir programas y datos, además de reducir costos por requerir únicamente duplicación de los componentes críticos.

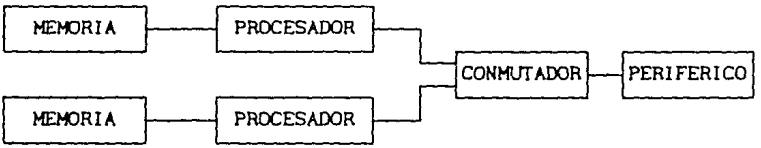


Figura 1.2 Sistema con computador de respaldo y conmutador de periférico [Figura 5(b) Siew 84]

1.3.2.3. SISTEMAS CON PUERTOS DUALES

En este caso, los periféricos estaban en linea con cada computador y la transferencia de programas y datos al computador de respaldo podía ocurrir a velocidad electrónica.

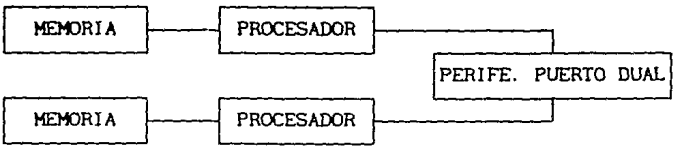


Figura 1.3. Sistema con puerto dual [Figura 5(c) Siew 84]

Un ejemplo de los periféricos de doble puerto lo presentan las familias de computadores de IBM S/360 y S/370. [Siew 82]

1.3.3. ESTRUCTURAS MULTIPROCESADOR

Este tipo de sistemas incluye computadores con varios (múltiples) procesadores que ejecutan concurrentemente (en paralelo) un mismo programa, tienen cierto grado de comunicación entre ellos y acceso a periféricos comunes. Uno de los procesadores actúa como principal y los otros como respaldo (redundantes).

El computador cuenta con un mecanismo de comparación de las salidas de los procesadores para determinar posibles diferencias ocurridas durante el tiempo de ejecución.

Algunos se basan en la igualdad permanente de sus resultados de salida; en otros se utilizan sesiones programadas (por ejemplo al final de cada tarea) para validar la consistencia entre los resultados producidos por el principal y los generados por los procesadores de respaldo. Si el procesador principal falla, alguno de los de respaldo continúa la computación.

La validación de resultados puede ocurrir bajo control de los programas de aplicación o del sistema operativo.

Los principales esquemas de computadores con multiprocesadores son los siguientes:

1.3.3.1. SISTEMAS MULTIPROCESADOR DEBILMENTE ACOPLADOS

Este tipo de sistemas se caracterizan por tener múltiples procesadores, cada uno de los cuales tiene su propia área de memoria. Adicionalmente se cuenta con algún tipo de enlace interprocesador, lo cual permite establecer comunicación débilmente acoplada entre computadores.

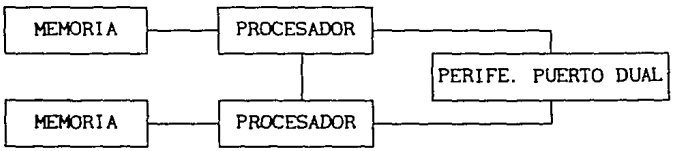


Figura 1.4 Sistema multiprocesador débilmente acoplado [Figura 5(d) Siew 84]

Estos sistemas emplean un solo sistema operativo para administrar sus recursos lo cual permite optimizar el uso de ambos procesadores. En caso de presentarse una falla, la unidad dañada es configurada fuera del sistema, el cual es reinicializado.

Entre los métodos de diagnóstico de las arquitecturas débilmente acopladas se encuentran los siguientes:

a) Utilización de programas de diagnóstico
b) Inclusión de capacidades de autodiagnóstico en cada módulo [Toy 78], [Morg 78]
c) Uso de reloj de vigilancia (watchdog) [Toy 78]
d) Uso de árbitro externo para controlar la configuración [Ihar 78]

Los sistemas débilmente acoplados presentan la ventaja de permitir un confinamiento (aislamiento) de falla más eficaz ya que, debido a que no comparten el área de memoria, es posible evitar la contaminación por parte de algún procesador en el que ocurra una falla.

Ejemplos de arquitecturas multicomputador débilmente acopladas lo constituyen los sistemas Tandem ([Siew 90], [Barl 78], [Tand 86], [Gray 90]), Stratus [Webb 91] y VAXft 3000 [Siew 90].

1.3.3.2. SISTEMAS MULTIPROCESADOR FUERTEMENTE ACOPLADOS

Estos sistemas se caracterizan porque varios procesadores comparten en mismo espacio de memoria y utilizan periféricos comunes.

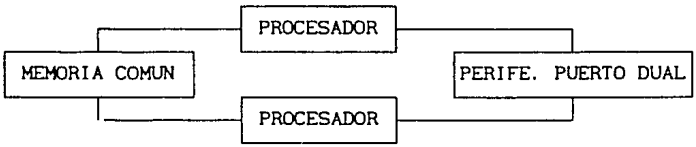


Figura 1.5 Sistema multiprocesador fuertemente acoplado [Figura 5(e) [Siew 84].

En este tipo de sistemas todos los procesadores tienen igual acceso a los recursos compartidos y cualquier unidad similar puede respaldar un componente fallado.

Un solo sistema operativo asigna la carga de trabajo a varios dispositivos sobre la base de primero en llegar, primero en servir (FIFO).

Además, los programas pueden cooperar utilizando un mínimo de recursos ("overhead"), esto es, los procesadores pueden verificar banderas en memoria sin involucrar al sistema operativo.

Dos aspectos claves de la tolerancia a fallas de las arquitecturas multiprocesador son la temprana detección y la minimización de la propagación de error.

Ejemplos de multiprocesadores con memoria compartida incluyen el Pluribus [Kats 78], C.mmp [Siew 78], Intel 432 [Siew 82], y el Synapse [Syna 83].

CAPITULO 2
OBJETIVOS DE DISEÑO DEL COMPUTADOR TOLERANTE A FALLAS

INTRODUCCION

En este capítulo se describe un conjunto de criterios que se ha utilizado como pautas para la incorporación sistemática de atributos de tolerancia a fallas en el diseño del computador aeronáutico.

Las características fundamentales que se han utilizado como guía en la definición de la estructura computacional son los siguientes:

- Actualidad tecnológica
- Niveles de degradación de servicio
- Confinamiento de fallas
- Funciones de respuesta a fallas por programación
- Dispositivos periféricos con propiedades de tolerancia a fallas

2.1. ACTUALIDAD TECNOLOGICA

El diseño de las diferentes partes del sistema debe realizarse utilizando componentes electrónicos estándar, de alto nivel de integración y que representen avances tecnológicos "recientes".

Comentarios:

a. El uso de componentes de alto nivel de integración favorece el planteamiento modular de sistemas. En el caso de computación tolerante a fallas esto es de especial importancia ya que la modularidad permite alcanzar un alto grado de confinamiento de fallas (restricción de los efectos de las fallas al módulo en que se originan).

b. Debe tenerse en cuenta que conforme avance el nivel de integración de los componentes utilizados, menor es la cantidad de líneas de conexión externas que deben ser manejadas para implementar una determinada función. Esto es importante pues una fuente importante de error tiene origen en los puntos externos de conexión física.

c. El empleo de componentes estándar no solo favorece la construcción y el mantenimiento del sistema (es posible contar con reemplazos) sino que además permite aprovechar programación disponible para operar con tales componentes.

d. Utilizar partes de tecnología "reciente" contribuye a que el producto final sea compatible con avances recientes en el campo, especialmente en lo que respecta a la programación de periféricos, de comunicaciones y del sistema operativo.

e. La implantación de sistemas con dispositivos de alto nivel de integración suele ir acompañada de una disminución en el consumo de potencia y un aumento en la frecuencia máxima de operación.

2.2. NIVELES DE DEGRADACION DE SERVICIO

En caso de detectar componentes fallados, el computador debe estar en capacidad de operar en diferentes niveles de degradación de servicio.

Comentarios.

a. La operación confiable de este computador se basa en el principio de la redundancia modular triple para el subsistema de procesamiento y redundancia simple en los dispositivos periféricos.

Este tipo de arquitectura cuenta con tres procesadores que operan en paralelo y un circuito de votación que compara las salidas de los procesadores para determinar la presencia de fallas. La comparación se basa en el principio de votación de mayoría.

Si se escriben las rutinas apropiadas, este computador puede operar con tres, dos o un solo procesador en modos de degradación creciente.

Los modos de degradación de servicio se definen mediante los registros de configuración del sistema. Cada procesador puede ser deshabilitado individualmente.

Los circuitos de votación, al igual que los dispositivos periféricos están duplicados y similarmente pueden ser deshabiltados por separado.

Dado que los dispositivos periféricos ocupan una posición subordinada respecto al procesador, es posible efectuar la detección de fallas a partir de la comparación de salidas. En caso de disparidad en las salidas, el diagnóstico de fallas puede efectuarse con apoyo del procesador.

2.3. CONFINAMIENTO DE FALLAS

La arquitectura del computador debe permitir un alto grado de confinamiento de fallas.

Comentarios. .

a. Las técnicas de confinamiento de fallas se basan fundamentalmente en el diseño modular de sistemas. Este principio permite restringir los efectos de las fallas al módulo en que se originan.

Para el caso de este computador, el objetivo de confinamiento de fallas está apoyado en el uso de un conjunto de registros de reconfiguración mediante los cuales se habilitan o deshabilitan las diferentes partes y se define la categoría (principal o respaldo) en que operan.

2.4. FUNCIONES DE RESPUESTA A FALLAS POR PROGRAMACION

Este computador debe contar con suficiente cantidad de recursos en la electrónica para poder realizar las funciones de respuesta a fallas por programación.

Comentarios.

a. Uno de los aspectos principales en que se basa la confiabilidad es el uso de recursos redundantes.

Para este sistema interesa disponer de un conjunto de recursos en su electrónica (redundancia en los periféricos, manejador de interrupciones y elementos de control tales como comparadores y circuitos de votación) que ofrezcan flexibilidad suficiente para apoyar la implantación de funciones de tolerancia a fallas

2.5. DISPOSITIVOS PERIFERICOS CON PROPIEDADES DE TOLERANCIA A FALLAS

Dado que este computador está orientado a aplicaciones aeronáuticas, debe contener los siguiente dispositivos periféricos:

a. Un puerto de entrada / salida tipo serie que permita el enlace del equipo de comunicaciones con el procesador;

b. Un reloj de tiempo real que posibilite contar con la información de reloj y calendario en los momentos que se realicen tareas de captura de información.

CAPITULO 3
DESCRIPCION DEL COMPUTADOR TOLERANTE A FALLAS

INTRODUCCION

En este capitulo se presenta una descripción de la estructura del computador redundante y tolerante a fallas con base en los módulos que lo constituyen

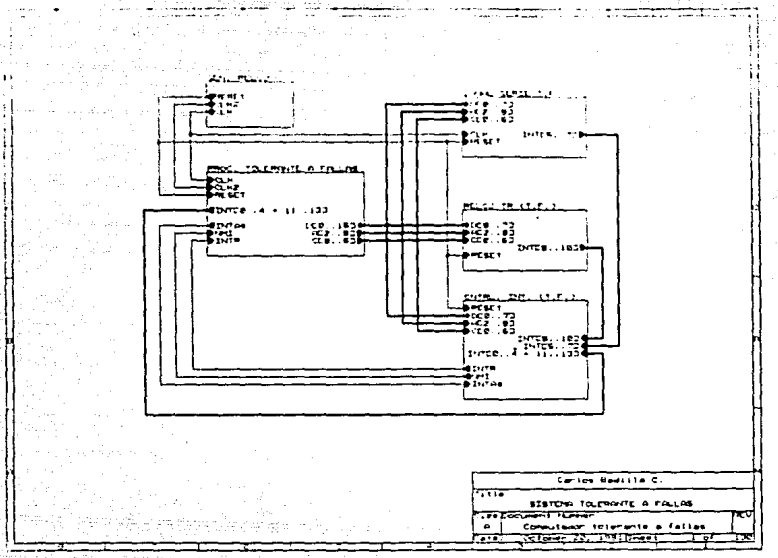
La arquitectura propuesta para el computador tolerante se ha estructurado siguiendo los principios del diseño modular de sistemas. De acuerdo con esta metodología, un sistema es visto como un conjunto de módulos o subsistemas. Cada uno de estos módulos a su vez está constituido por una serie de elementos de menor nivel y así sucesivamente hasta llegar a los componentes elementales (no divisibles).

En este diseño se consideran como componentes elementales diversos tipos de circuitos integrados (microprocesadores, decodificadores, memorias, compuertas lógicas, etc.) así como diferentes componentes eléctricos (resistencias, interruptores, cristales, etc.)

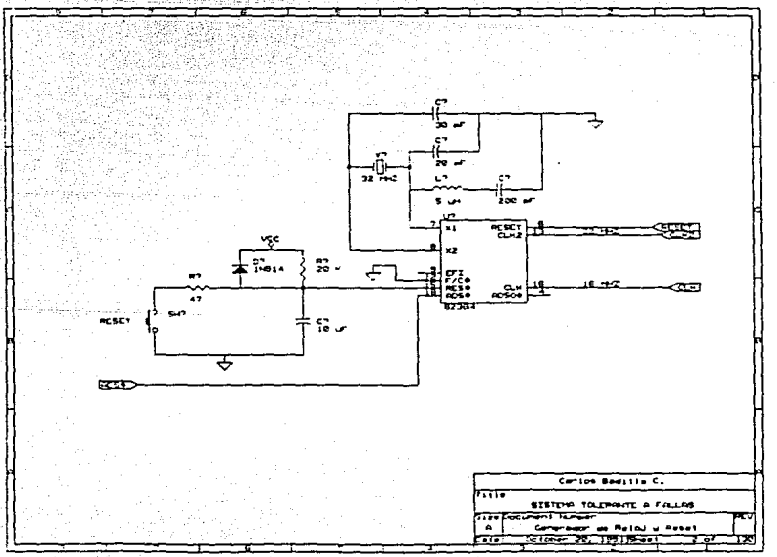
De acuerdo con el modo de representación indicado, el diagrama electrónico del sistema está elaborado en forma de una jerarquía compleja con estructura de árbol en la cual, la lámina principal (o raíz) de este esquema presenta el computador tolerante a fallas.

3.1. CARACTERISTICAS GENERALES DEL COMPUTADOR

Este computador está compuesto por un subsistema de procesamiento y por un conjunto de dispositivos periféricos: canal de comunicaciones serie, reloj de tiempo real y controlador de interrupciones (lámina 1). La operación general del sistema se sincroniza por medio de un generador de reloj.



LAMINA 1. COMPUTADOR TOLERANTE A FALLAS



LAMINA 2. GENERADOR DE RELOJ Y RESET

Tanto el procesador como los periféricos contienen características en su arquitectura que les permiten presentar un comportamiento confiable ante presencia de fallas.

No obstante, considerados como bloques funcionales, las propiedades de tolerancia a fallas son "transparentes" para los circuitos con los que interactúan, es decir, desde la perspectiva de sus terminales externos presentan un comportamiento similar al de un módulo equivalente pero no tolerante a fallas.

Conceptualmente, el procesador está basado en el principio de la redundancia modular triple. Esto significa que utiliza tres procesadores (MC80386) que ejecutan sus tareas en forma concurrente y cuya operación es permanentemente vigilada por un circuito de votación capaz de detectar diferentes tipos de fallas.

Por su parte, el diseño de los dispositivos periféricos contempla el uso de componentes duplicados y circuitos para comparar los datos de salida.

En resumen, puede considerarse que las propiedades de tolerancia a fallas del computador propuesto están basadas fundamentalmente en las características de la arquitectura física (tales como unidades redundantes, registros de configuración, circuitos comparadores) y en las de la programación del sistema (tales como operación guiada por interrupciones, rutinas de verificación, diagnóstico y reconfiguración).

Esto permite colocar en un segundo plano la calidad de las partes electrónicas en lo que respecta a lograr una operación confiable.

3.2. UNIDAD DE MEMORIA LOCAL

En lo que respecta a la unidad de memoria, este diseño plantea la utilización de un bloque de memoria independiente para cada procesador (lámina 5), con el propósito de incrementar el grado de confinamiento de fallas.

El uso de bloques separados de memoria para los procesadores presenta la ventaja de que, en caso de error, éste no es propagado hacia otros procesadores como podría ocurrir si los procesadores utilizaran memoria compartida.

El microprocesador 80386, empleado en este sistema, puede administrar un espacio de direccionamiento de memoria física de 4 Gbytes (32 líneas de direcciones); no establece zonas especiales para ubicar las áreas de memoria de tipo RAM o de tipo ROM. La definición del tipo de memoria a usar en las diferentes localidades del área de memoria la realiza el diseñador del sistema.

3.2.1 TIPOS DE MEMORIA

El módulo de memoria local consta de una sección de memoria RAM estática, para el área de memoria de contenido variable, y otra de memoria tipo EPROM para el área de almacenamiento de datos y código invariable (lámina 9).

Se prefirió el uso de memoria estática sobre dinámica ya que de acuerdo con su principio de operación (estática basada en flip-flops, dinámica basada en conservación de cargas de condensadores) puede esperarse un comportamiento más confiable de las memorias tipo RAM.

3.2.2 TAMAÑO DE LA MEMORIA

La memoria del sistema está compuesta por un espacio de 8 Mbytes (4M palabras de 16 bits) de memoria RAM estática y 256 Kbytes (128K palabras de 16 bits) de memoria EPROM. Desde la perspectiva del programador, este espacio es visto como una secuencia continua de bytes (característica de operación establecida para los procesadores de Intel).

La definición del tamaño de la memoria se ha hecho con la finalidad de dotar al computador de un espacio de memoria suficiente para adaptarse con flexibilidad a diversos requerimientos de la programación. De esta manera, es posible contar con espacio suficiente para implantar un sistema operativo con características de tolerancia a fallas por programación.

Adicionalmente, un espacio amplio en memoria permite hacer frente a necesidades variables tales como procesamiento de imágenes o correo electrónico.

3.2.3 DEFINICION DEL MAPA DE MEMORIA

El esquema de atención de interrupciones del 80386 provee un criterio para separar físicamente el espacio de memoria.

El mecanismo de atención de interrupciones utilizado por el 80386 asocia a la interrupción INTR un espacio en memoria que empieza en la dirección 00H en el cual se encuentran los vectores que contienen las direcciones iniciales de las correspondientes rutinas de atención de interrupción. Esto obliga a colocar memoria de tipo RAM en la parte baja del espacio de direccionamiento.

Por su parte, cuando se presenta la condición de RESET,el procesador es inicializado a un estado interno conocido que empieza a buscar instrucciones a partir de la dirección de reset (FFFF FFF0 H). Esto hace necesario colocar memoria de tipo ROM (o EPROM) en las direcciones altas.

Esta forma de división del espacio de memoria (SRAM en la parte baja y EPROM en la parte alta) provee un mecanismo sencillo de selección pues puede utilizarse la línea 23 como criterio de discriminación: Si A23 = 0 se direcciona memoria de tipo SRAM; si A23 = 1, se direcciona memoria de tipo EPROM.

Específicamente para este sistema, los espacios de direcciones ocupados por cada tipo de memoria han sido asignados según se indica en la siguiente tabla.

DIRECCION	TIPO DE MEMORIA
0000 0000 007F FFFF	SRAM (8 M bytes)
0080 0000 FFFB FFFF	LIBRE
FFFC 0000 FFFF FFFF	EPROM (256 K bytes)

TABLA 3.1
Distribución del espacio de memoria

3.2.3.1. MEMORIA RAM ESTATICA

La memoria SRAM se encuentra en el espacio comprendido entre las direcciones 0000 0000 H (A0..A22 = 0) hasta 007F FFFF H (A0..A22 = 1), con A23..A31 = 0 (láminas 10 a 14).

Los valores presentes en las líneas A1..A22 del módulo µp 80386 definen la localidad de palabra a ser accesada. Las líneas BEL#, BEH# seleccionan el byte direccionado (esta últimas líneas se requieren solo para escritura).

3.2.3.2. MEMORIA EPROM

La memoria EPROM se encuentra en el espacio comprendido entre las direcciones FFFC 0000 H (A0..A17 = 0) hasta FFFF FFFF H (A0..A17 = 1), con A18..A31 = 1 (láminas 15 a 17).

3.3. DISPOSITIVOS PERIFERICOS

Los dispositivos periféricos usados en este sistema pueden clasificarse en dos tipos: de propósito general y de apoyo a las funciones de tolerancia a fallas.

Los periféricos de propósito general son el canal de comunicaciones serie, el reloj de tiempo real y el controlador de interrupciones. Los utilizados para asistir las funciones de tolerancia a fallas son un conjunto de registros mediante los cuales se define la configuración de las diferentes partes, se controla la ejecución y se define el acceso a los buses del sistema.

Los dispositivos periféricos de propósito general se han diseñado incorporándoles una serie de partes que les permiten presentar un comportamiento tolerante a fallas.

Cada dispositivo periférico está formado por dos componentes de entrada-salida del mismo tipo. Por ejemplo el canal serie cuenta con dos UARTs NS16550A; similarmente, el reloj de tiempo real está constituido por dos componentes DS1287. Tales componentes duplicados están mapeados en las mismas localidades del espacio de puertos del procesador.

Adicionalmente, cada uno de los periféricos cuenta con un par de registros cuyo contenido, conocido como parámetros de configuración del periférico [estado (activo / inactivo#) y categoría (principal / respaldo#)] es utilizado para regular la operación de los elementos que integran al dispositivo periférico.

3.3.2 ESTRUCTURA DEL MAPA DE PUERTOS

El microprocesador 80386, al igual que los otros procesadores de la familia Intel puede direccionar hasta 64 K puertos. Dicho espacio se accesa utilizando las líneas del bus de direcciones de menor peso (BE0#..BE3#, A2..A15). Aunque los periféricos comparten el espacio de direcciones con los dispositivos de memoria, el procesador los maneja de forma diferente. En un momento dado, el procesador puede accesar memoria o puertos pero no ambos de manera simultánea. Además, las instrucciones utilizadas para el manejo de memoria son diferentes de aquellas utilizadas para interactuar con los puertos.

Dado que el espacio de puertos definido por el microprocesador 80386 (64 k localidades) es suficientemente grande para manejar todos los dispositivos del sistema, se cuenta con bastante libertad para asignar direcciones a los dispositivos individuales.

En consecuencia, en este diseño se definen cuatro bloques de dispositivos periféricos, cada uno de los cuales tiene asignada una extensión de 64 localidades de 1 byte, con alineamiento de doble palabra (separación de 4 bytes entre sí).

El tipo de alineamiento escogido permite facilitar la decodificación de puertos pues pueden utilizarse las líneas de direcciones A2 .. A3 y no se requiere decodificar las líneas de habilitación de byte (BE0# .. BE1#).

El tamaño de los bloques se definió con base en el espacio requerido por el reloj de tiempo real (64 localidades: 14 registros de manipulación y control y 50 localidades de propósito general).

Los bloques de periféricos son los siguientes:

a. Registros de configuración de periféricos y procesadores
b. Programación y control del canal serie
c. Programación y control del reloj de tiempo real
d. Programación y control de los manejadores de interrupciones

La ubicación de cada uno de estos grupos es definido por las líneas A8, A9 del espacio de direcciones como se indica en la siguiente tabla:

PERIFERICO	A_9 A_8
1. Módulos de configuración	0 0
2. Canal serie	0 1
3. Reloj de tiempo real	1 0
4. Manejadores de interrupciones	1 1

Tabla 3.2 Grupos de periféricos

MAPA DE PUERTOS DEL SISTEMA

REG.	PUERTO	DIR (Hex)
0	Estatus procesadores (Activo / Inactiv#)	00
1	Categoría procesadores (Princ / Respaldo#)	04
2	Modo Op. procesadores (Maestr / Esclavo#)	08
4	Estatus Circ. Votación (Activ /Inactiv#)	10
5	Categoría Circ. Votación (Princ /Respald#)	14
6	Salidas Comparadores A (Circ. Votación 0)	18
7	Salidas Comparadores B (Circ. Votación 1)	1C
8	Estatus Serializadores (Activ / Inact#)	20
9	Categoría Serializadores (Princ / Resp#)	24
10	Estatus Comparadores Serial. (Act/Inac#)	28
11	Categoría Comparadores Serial. (Prn/Resp#)	2C
12	Estatus Relojes Tiempo Real (Act/Inac#)	30
13	Categoría Relojes Tiempo Real (Prn/Resp#)	34
14	Estatus Comparadores R.T.R. (Act /Inac#)	38
15	Categoría Comparadores R.T.R. (Prn /Resp#)	3C
16	Estatus Controladores Inter. (Act/Inac#)	40
17	Categoría Controladores Inter. (Prn/Resp#)	44
18	Estatus Comparadores Ctr. In. (Act/Inac#)	48
19	Categoría Comparadores Ctr. In. (Prn/Resp#)	4C
20	Estatus procesadores (Activo / Inactiv#)	80
21	Categoría procesadores (Princ. / Respald#)	84
22	Modo Op. procesadores (Maestro /Esclavo#)	88
23	Solicitud Interrupción de Procesadores	8C
Programación y Control Serializadores		100H .. 11CH
Programación y Control Relojes Tiempo R.		200H .. 2FCH
Programación y Control PIC maestro		300H .. 304H
Programación y Control PIC esclavo		308H .. 30CH

TABLA 3.3 División del espacio de puertos

La ubicación de cada localidad dentro de un grupo se define mediante el valor de las líneas $A_2..A_7$. En total 6 líneas de direcciones que permiten accesar hasta 64 localidades.

La descripción detallada de las características y funciones de cada uno de los periféricos se incluye en los capítulos siguientes.

En la tabla 3.3 se presenta la forma en que se ha dividido el mapa de puertos del sistema.

3.4. SINCRONIA DEL SISTEMA

Con el fin de lograr que los módulos de comparación detecten verdaderas diferencias entre los datos comparados, en vez de aparentes diferencias resultantes de desfasamientos respecto a los momentos en que tales datos se producen, este computador utiliza un solo circuito de generador de reloj cuya función consiste en sincronizar la operación de los componentes del sistema.

El circuito de reloj (lámina 2) está basado en el generador de reloj 82384 el cual es un componente multifunción de la familia 80386 que proporciona las señales de reloj para la operación sincrónica del microprocesador 80386 y sus componentes de soporte.

El 82384 produce las siguientes señales:

3.4.1. CLK: Señal de reloj utilizada por los microprocesadores y otros circuitos del sistema (lógica de control del bus y periférico canal serie).

3.4.2. CLK2: Señal de reloj con frecuencia igual al doble de la frecuencia de la señal CLK. Es utilizada por los microporcesadores y por sus correspondientes circuitos de control del bus.

La fase de la señal CLK se sincroniza con la señal de reloj generada internamente por el microprocesador principal. El 82384 utiliza la señal de salida del microprocesador ADS# para ajustar la sincronia.

3.4.3. RESET: Señal utilizada para incializar (conducir a un estado inicial predefinido) a los microprocesadores y otros circuitos periféricos (registros de reconfiguración, reloj de tiempo real y canal de comunicaciones serie).

El 82384 acepta una entrada asincrona procedente de un circuito RC, sincroniza dicha entrada con la salida CLK y produce la señal RESET (activa alto).

Para manejar el 82384, puede usarse una fuente externa de frecuencia o un cristal. La entrada F/C# indica la fuente de señal. Si F/C# es puesta en alto, el 82384 reconoce a su señal de entrada EFI (External Frequency Input) como su fuente de frecuencia. Si F/C# es puesta en bajo, el cristal colocado entre los pines X1, X2 del 82384 se utiliza como fuente de frecuencia. En cualquier caso, la frecuencia fuente debe ser igual a la frecuencia CLK2 requerida. Para el caso de este computador se propone el uso de una frecuencia CLK = 16 Mhz que corresponde a la máxima frecuencia de operación aceptada por el 80386.

La entrada RESET del 80386 debe permanecer en alto al menos 15 periodos de CLK2 para asegurar la correcta inicialización y al menos 80 periodos si se ejecuta autoprueba.

CAPITULO 4
SUBSISTEMA DE PROCESAMIENTO

INTRODUCCION

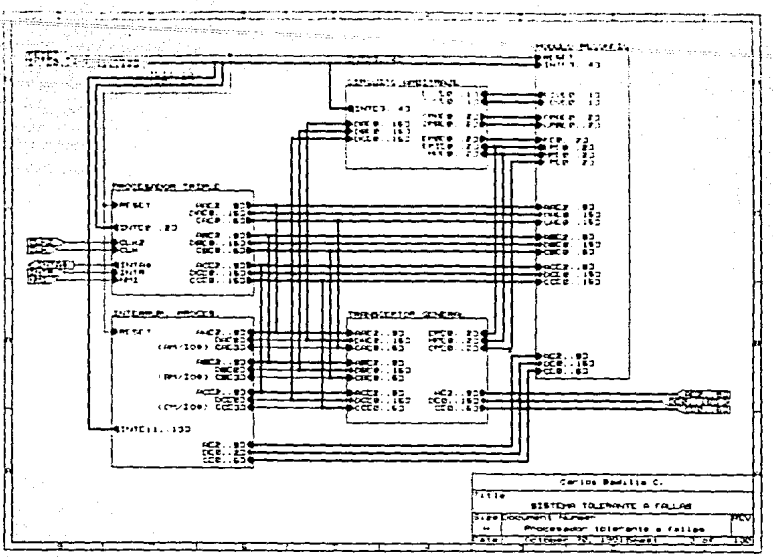
Este capítulo contiene una descripción general del módulo principal (módulo de procesamiento) del computador presentado. La incorporación de atributos de tolerancia a fallas en esta sección del computador se basa en la introducción de redundancia electrónica en la unidad de cálculo (procesador triple) y en el empleo de circuitos de control (circuito de arbitraje, modulo de reconfiguración, transceptor general y circuito de generación de interrupción de procesadores [lámina 3]) que son descritos en los capítulos siguientes.

4.1. PROCESADOR TRIPLE

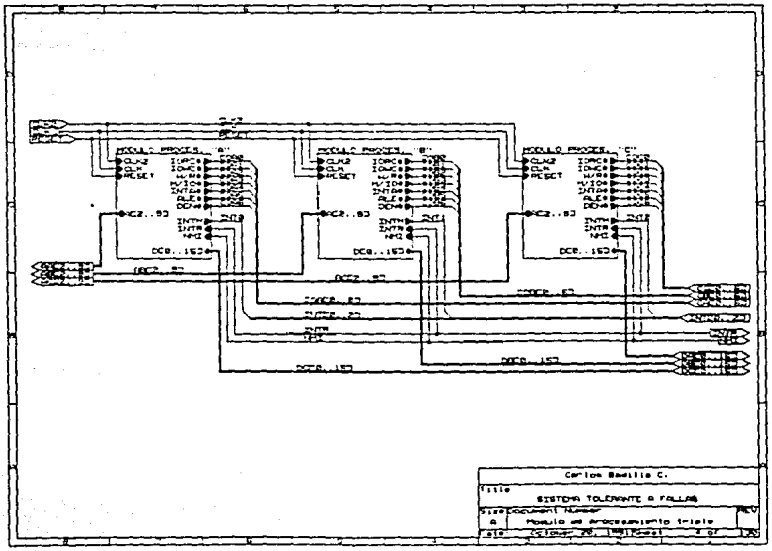
El procesador triple (lámina 4) está integrado por tres módulos iguales de procesamiento que operan en forma concurrente (en paralelo).

La operación de cada uno de los procesadores se caracteriza por medio de tres parámetros: estado, categoría y modo de operación, cuyo significado se describe en el capítulo 6 (Módulo de reconfiguración de procesadores).

Las líneas de entrada y salida de información de los procesadores están conectadas a los circuitos de control del módulo de procesamiento (circuito de arbitraje, módulo de reconfiguración, transceptor general y circuito de generación de interrupción del procesador)



LAMINA 3. PROCESADOR TOLERANTE A FALLAS



LAMINA 4. MODULO DE PROCESAMIENTO TRIPLE

Todos estos procesadores son funcionalmente equivalentes. No obstante, la función desempeñada por cada uno de ellos (acceso a los buses del sistema) es definido con base en un conjunto de parámetros (estado, categoría y modo de operación) los cuales son almacenados en el módulo de reconfiguración del sistema.

Los resultados de salida de los procesadores, conjuntamente con la información de configuración, es utilizada por los circuitos de control para efectuar los procesos de votación, arbitraje y reconfiguración del sistema.

4.2. MODULO DE PROCESAMIENTO

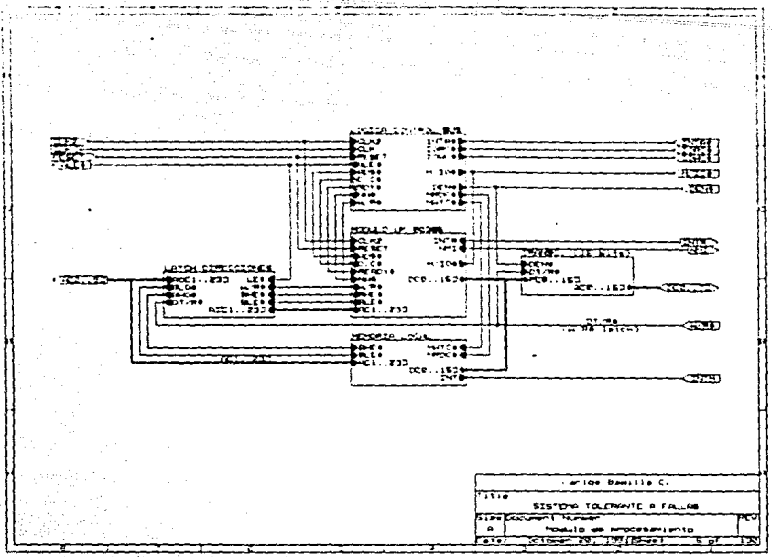
Cada uno de los procesadores que forman el procesador triple está constituido por un circuito de microprocesador basado en Intel 80386, un circuito de lógica de control del bus, un módulo de memoria local, un candado ("latch") de direcciones y un transceptor ("transceiver") de datos (lámina 5).

4.2.1. CIRCUITO MICROPROCESADOR

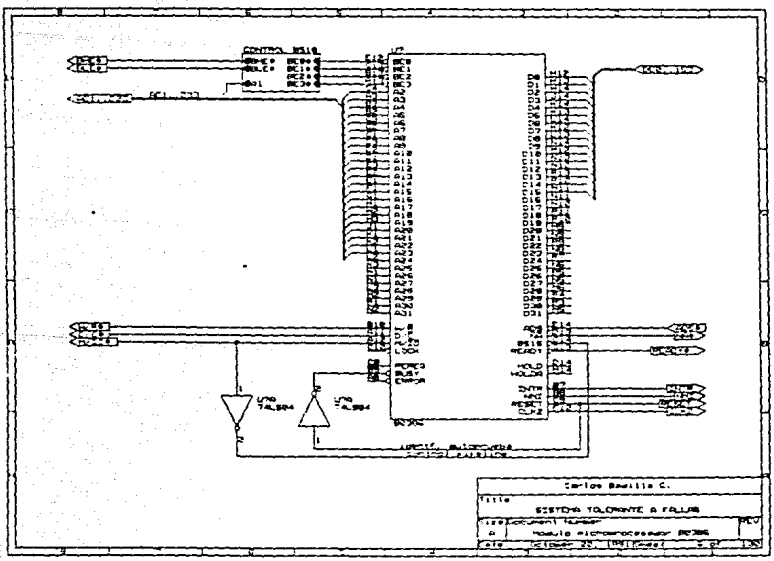
Para realizar las tareas de procesamiento de datos el subsistema procesador utiliza un microprocesador de propósito general 80386 (lámina 6). Esta sección se encarga de ejecutar los programas de aplicación y las rutinas de diagnóstico y reconfiguración del sistema.

Las características más sobresalientes del microprocesador 80386 son las siguientes: bus de datos (interno y externo) de 32 bits, bus de direcciones de 32 bits, líneas de datos y direcciones separadas (bus no multiplexado).

El 80386 maneja tipos de datos de 8, 16 y 32 bits y tiene 8 registros de 32 bits de propósito general. Cuenta con un espacio de direccionamiento de 4 Gbytes de memoria física, 64 Terabytes de memoria virtual y un máximo tamaño de segmento de 4 Gigabytes.



LAMINA 5. MODULO DE PROCESAMIENTO



LAMINA 6. MODULO MICROPROCESADOR 80386

El código objeto es compatible con el de todos los procesadores de la familia 86. Tiene circuitos de apoyo para depuración.

Presenta características de la arquitectura superiores a las de sus predecesores: ejecución encauzada de instrucciones ("pipeline"), memorias caché de traducción de direcciones, reloj interno de 12.5 y 16 Mhz (externo de 25 y 32 Mhz), ancho de banda del bus de 32 Megabytes / Sec y un rendimiento de 3 a 4 MIPS.

Puede trabajar con los coprocesadores matemáticos 80287 y 80387 para incrementar su capacidad de procesamiento numérico.

Está construido con tecnología CHMOS III y está encapsulado en un circuito de 132 pines ordenados en rejilla.

Sobre las conexiones del 80386 en este sistema pueden hacerse las siguientes observaciones:

a. ENTRADA BS16#

A esta entrada se conecta la salida M/IO# del 80386 (lámina 6). Dado que el sistema propuesto cuenta únicamente con memorias de 16 bits, no es necesario generar diferentes valores para la entrada BS16# para diferentes accesos a memoria. Por tanto, el diseño se simplifica: siempre que se realicen accesos a memoria, se activa la entrada BS16#. Si se realizan accesos a dispositivos IO, esta entrada permanece inactiva.

b. ENTRADA BUSY#

La entrada BUSY# utilizada para establecer la comunicación con el coprocesador matemático (80287 o 80387). Se utiliza para informar al 80386 que el coprocesador se encuentra ejecutando una instrucción y que, por lo tanto, no puede aceptar otra.

Adicionalmente esta señal se utiliza para solicitar la ejecución de la rutina de autoprueba durante la etapa de inicialización del sistema.

Dado que el sistema propuesto no contempla el uso de coprocesador, esta entrada puede ser conenctada por medio de un inversor a la entrada RESET (lámina 6), lo cual programa en forma permanente la autoverificación durante la secuencia de RESET.

c. GENERACION DE BHE#, BLE#, A1

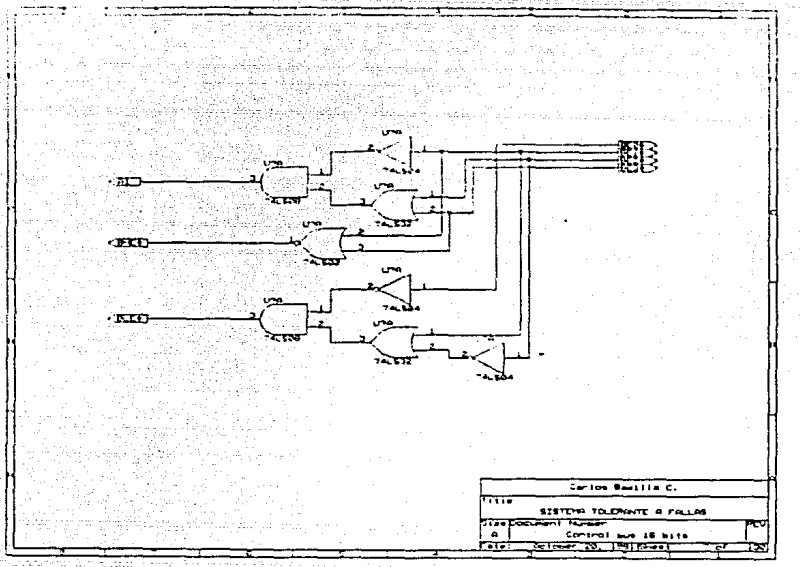
El bus de direcciones del 80386 está formado por las líneas A2..A31 y BE0# ..BE3#. Las líneas A2..A31 corresponden a las líneas más significativas del bus y las líneas BE0#..BE3# corresponden a la decodificación de las líneas A0..A1 que son proporcionadas como salidas del 80386 para facilitar el direccionamiento de memoria orientado a bytes.

Dado que el sistema aquí estudiado utiliza solo memoria de 16 bits, se emplea un circuito combinacional (lámina 7) para generar las señales A1, BHE# y BLE#, tomando como base para ello la información técnica sobre el microprocesador [INTE 86] (Tabla 3-5, pag. 3-23)

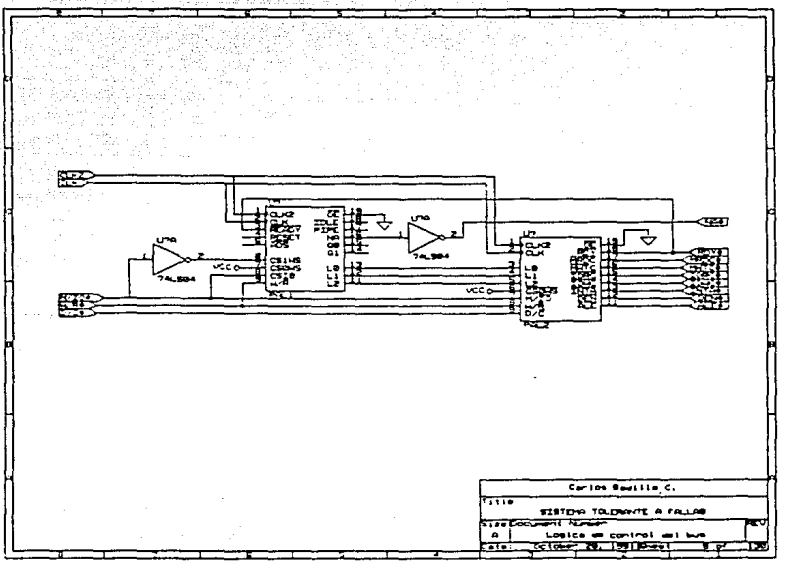
4.2.2. LOGICA DE CONTROL DEL BUS

El circuito de control del bus provee las señales requeridas para manejar los "latches" de direcciones, los transceptores de datos, los dispositivos de memoria y los dispositivos de entrada - salida. Además se encarga de generar las señales NA# y RDY# para el control del 80386.

El microprocesador 80386, a diferencia de otros predecesores de la misma familia, no cuenta con líneas de salida para control del bus externo ni con dispositivos específicos para generar las señales necesarias para ello. En vez de eso tiene cinco líneas de estado del bus (ADS#, W/R#, M/IO#, D/C#, LOCK#) que pueden utilizarse como entradas a dispositivos lógicos programables que reúnan los requerimientos de un sistema particular.



LAMINA 7. CONTROL DE BUS DE 16 BITS



LAMINA 8. LOGICA DE CONTROL DEL BUS

El sistema tolerante a fallas propuesto, utiliza los arreglos lógicos programables PAL-1 y PAL-2 para implantar el circuito de la lógica de control del bus (lámina 8). El uso de estos dispositivos se basa en las sugenrencias planteadas por los circuitos de aplicación incluidos en las notas técnicas de Intel ([Int 86a], Figura 6.4., página 6-7).

La programación de estos componentes se indica en el Apéndice A de la misma referencia (Descripciones de las PALs de la lógica de control del bus).

El circuito de control del bus utiliza la información sobre selección de dispositivos, además de la información sobre estado del bus para definir las señales NA#, RDY#, ALE#, DEN#, INTA# y los comandos de lectura y escritura a puertos (IORC#, IOWC#) y a memoria (MRDC#, MWTC#)

a. CONTROL DEL "PIPELINE" DE DIRECCIONES

La señal NA# se utiliza para gobernar el "cauce" ("pipeline") de direcciones. Esto es, la generación concurrente de las direcciones y señales de estado del siguiente ciclo durante el ciclo de bus actual.

La señal NA# es generada por la lógica de control del bus con base en los estados de las señales CS1W#, CSWO#, CSIO#; es decir, con base en el tipo de dispositivo accesado.

El valor presente en la entrada CSIO# identifica si el ciclo es de memoria (CSIO# = 1) o de dispositivo externo (CSIO# = 0).

Las señales CS1W# y CSOW# son relevantes únicamente durante los ciclos de acceso a memoria (M/IO# = 1).

La entrada CSOW# debe conectarse a la referencia para solicitar a la lógica de control del bus que genere las señales requeridas para acceso a memoria de 32 bits. Dado que el sistema propuesto solo utiliza memoria de 16 bits, la entrada CSOW# se conecta en forma permanente a VCC.

Si CSOW# = 1 (siempre en este diseño) y CS1W# = 0, se realiza acceso a memoria de 16 bits. En este caso, la lógica de control del bus genera la señal NA# en forma apropiada para gobernar el pipeline de direcciones en sistemas con memoria de 16 bits.

Esto permite utilizar "encauzado" de direcciones incluso en el caso de requerir únicamente bus de datos externo de 16 bits.

b. INTRODUCCION DE ESTADOS DE ESPERA EN ACCESOS EXTERNOS

Para el caso de este sistema, la distinción entre acceso a dispositivos de memoria y de entrada/ salida se realiza mediante el estado de la línea M/IO#, la cual se conecta a la entrada CSIO#. Cuando esta línea se encuentra en cero, no se toman en cuenta los valores presentes en las entradas CSOW# y CS1W#.

Cuando CSIO# = 0 (acceso a dispositivo de IO), la lógica de control del bus cuenta los estados y activa la señal RDY# después de la cantidad de estados de espera requeridos por el dispositivo accesado.

En caso de utilizar los arreglos lógicos programables sugeridos por Intel, se introduce una cantidad fija de estados de espera para todos los accesos a dispositivos IO, lo cual simplifica la programación de las PALs y no degrada significativamente el desempeño del sistema.

4.2.3. CIRCUITO DE MEMORIA LOCAL

La memoria local está definida como un bloque de memoria RAM estática y otro de memoria EPROM.

Tal como se indica en los diagramas del circuito (láminas 9 a 17) la memoria local está diseñada en forma modular, a partir de dispositivos HM66205L (SRAM 512K x 8 bits) y HM6027 (SRAM 256K x 1 bit) para el caso de la sección de memoria estática y 27C256 (EPROM 32K x 8) para la sección de memoria de solo lectura.

4.2.3.1. CIRCUITO DE MEMORIA RAM ESTATICA

Esta zona de la memoria es vista por el procesador como un espacio de 4M palabras de 16 bits (8 Mbytes). Está integrada por dos módulos de 4M x 8 bits y otros dos de 4M x 1 bit. Esta forma de organización permite que el procesador accese la memoria como una secuencia continua de bytes (palabras de 8 bits).

Adicionalmente, el módulo de memoria estática cuenta con un circuito generador y detector de paridad (74HC180) (lámina 10) el cual calcula, y almacena en forma concurrente, la información de paridad al momento de escribirse un dato en memoria. Para este sistema se utiliza paridad par.

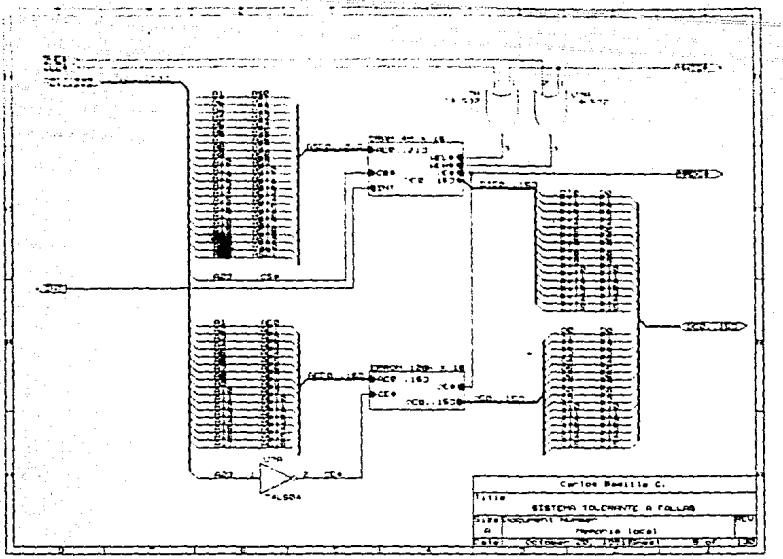
Durante los procesos de lectura de información almacenada en memoria, la paridad del dato leído se recalcula y se compara con la que fue previamente almacenada al escribirlo. Si estos valores no coinciden se genera una señal de interrupción.

Dado que cada procesador tiene su espacio de memoria particular, el subsistema de procesamiento cuenta con tres fuentes de interrupción INT[0..2] para señalar errores de paridad. Esto permite que cada procesador pueda reportar, en forma independiente, los errores de paridad detectados en su memoria local.

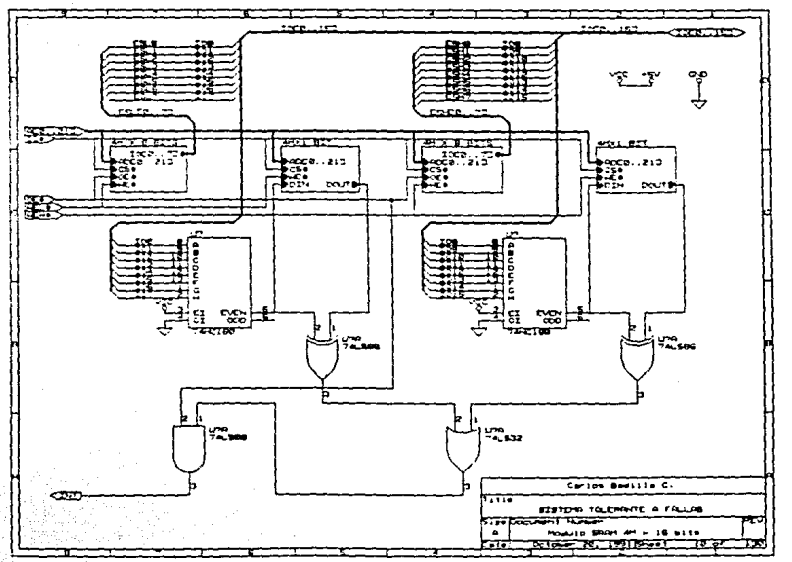
Las acciones que se realicen como consecuencia de la detección de una falla de este tipo deben ser definidas dentro de la correspondiente rutina de atención de interrupción.

4.2.3.2. CIRCUITO DE MEMORIA EPROM

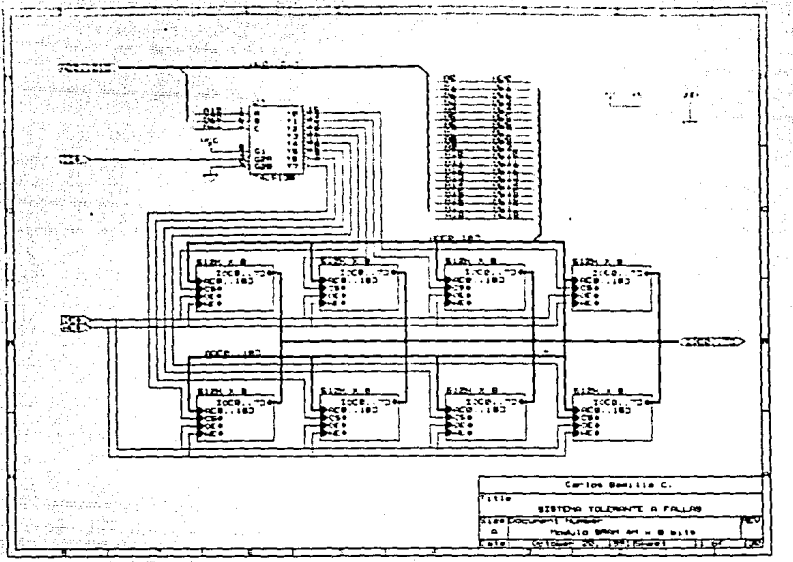
La memoria de tipo EPROM está organizada como un espacio de 128K palabras de 16 bits. Sin embargo también puede accesarse en modo byte.



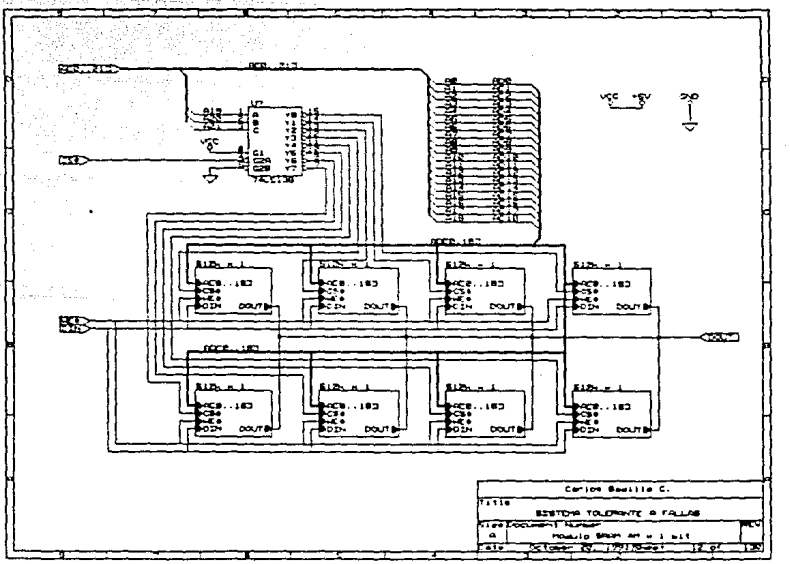
LAMINA 9. MEMORIA LOCAL



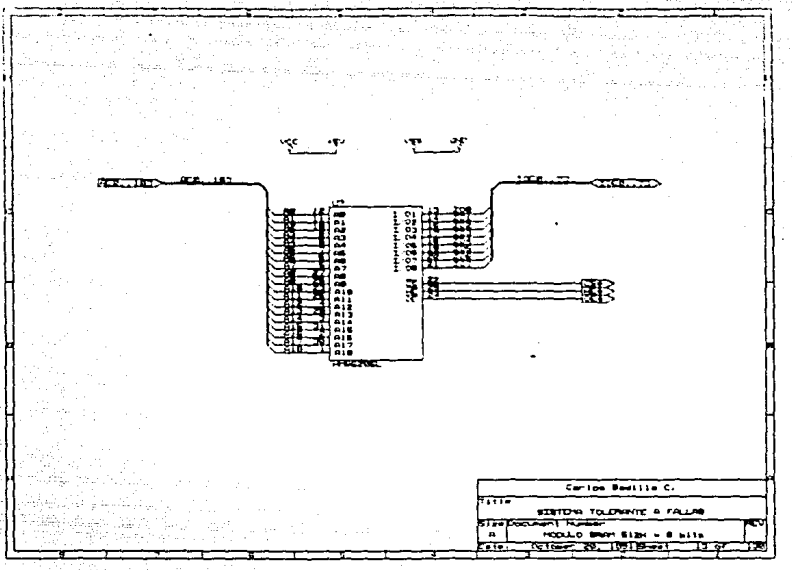
LAMINA 10. MODULO SRAM X 16 BITS



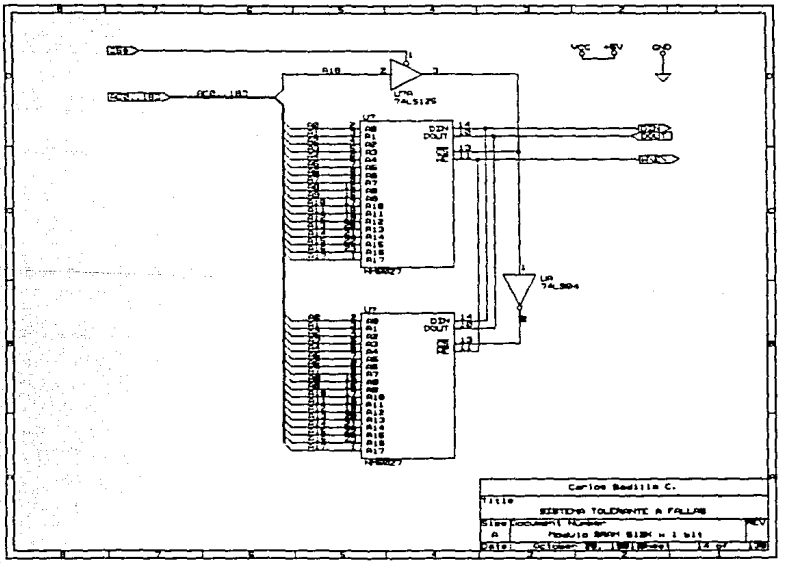
LAMINA 11. MODULO SRAM 4M X 8 BITS



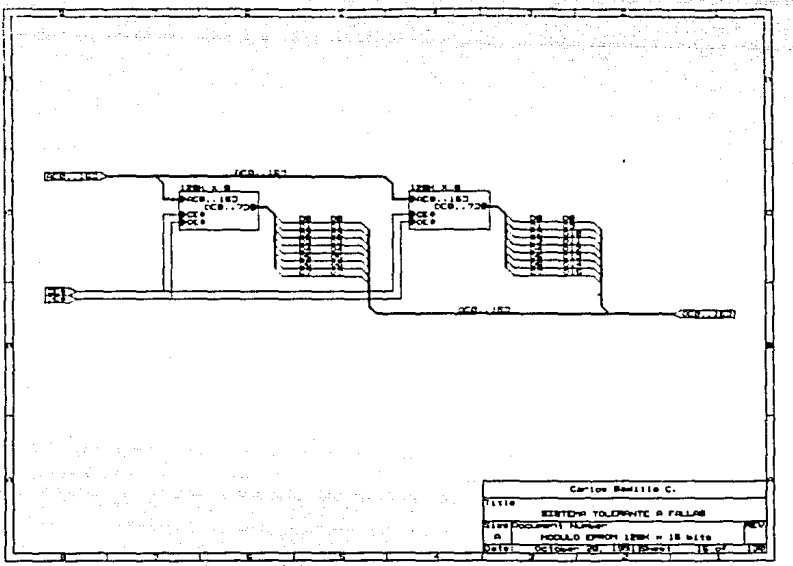
LAMINA 12. MODULO SRAM 4M X 1 BIT



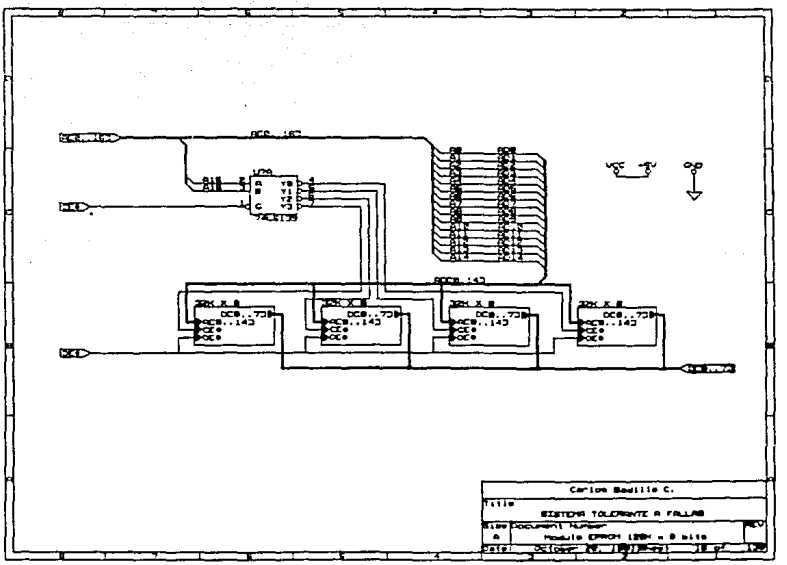
LAMINA 13. MODULO SRAM 512K X 8 BITS



LAMINA 14. MODULO SRAM 512K X 1 BIT



LAMINA 15. MODULO EPROM 128K X 16 BITS



LAMINA 16. MODULO EPROM 128K X 8 BITS

Para este tipo de memoria no se utiliza la verificación de paridad. En principio este método de detección de error también puede emplearse para examinar la consistencia de datos durante los procesos de lectura. No obstante, dicho mecanismo no se incluye en el diseño por no contar con información sobre memorias EPROM de un bit y no verse justificado el uso de memorias con mayor tamaño de palabra de salida.

4.2.4. LATCH DE DIRECCIONES

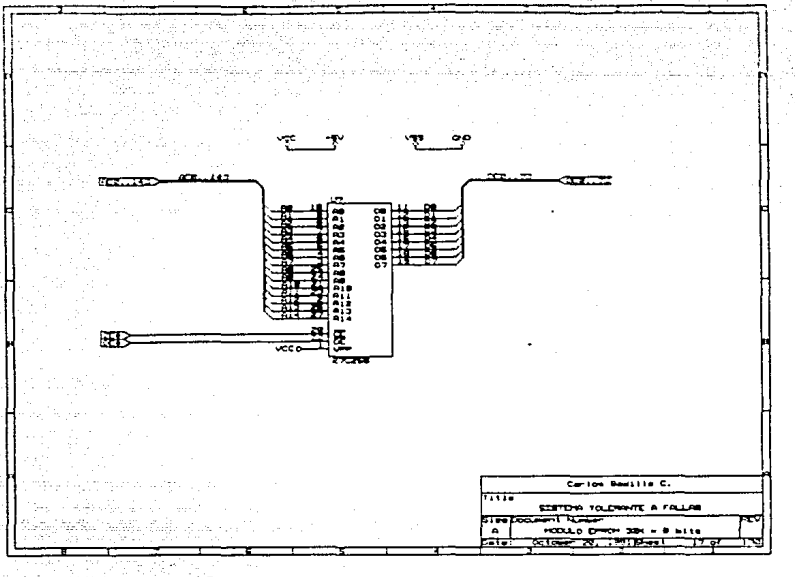
Por este circuito (lámina 18) pasan las líneas del bus de direcciones (A1..A23, BHE#, BLE#) así como la línea W/R#. Este circuito es necesario para permitir la operación del encauzado ("pipelining") de direcciones.

4.2.5. TRANSCEPTOR DEL BUS DE DATOS

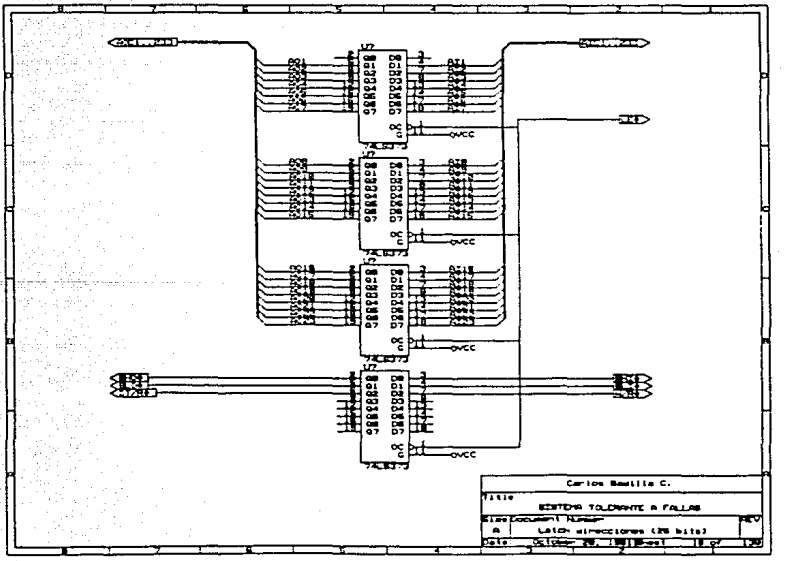
Las líneas del bus de datos se conectan al resto del sistema por medio de un conjunto de transceptores. Para controlar estos transceptores se utiliza la línea DEN# (data enable) procedente de la lógica de control del bus así como la línea DT/R# proveniente del "latch" de direcciones.

La señal DEN# indica el tiempo durante el cual los datos presentes en el bus de datos son válidos. La señal DT/R# indica el sentido (entrante o saliente) de los datos presentes en el bus. Esta señal es la versión "latch" de la línea de salida W/R# del procesador 80386.

Los transceptores son utilizados para evitar la contención del bus de datos en caso de que los dispositivos periféricos sean lentos para quitar los datos proporcionados al procesador después de un ciclo de lectura.



LAMINA 17. MODULO EPROM 32K X 8 BITS



LAMINA 18. LATCH DE DIRECCIONES

CAPITULO 5
CIRCUITO DE ARBITRAJE:
DETECCION DE ERROES A NIVEL DEL PROCESADOR

INTRODUCCION

En este capítulo se describe la estructura y el funcionamiento del mecanismo utilizado para realizar la detección de errores originados en el subsistema de procesamiento; también se explica el tipo de información suministrada por este módulo al procesador para conducir las rutinas de diagnóstico de fallas y reconfiguración del sistema.

5.1. CIRCUITO DE ARBITRAJE

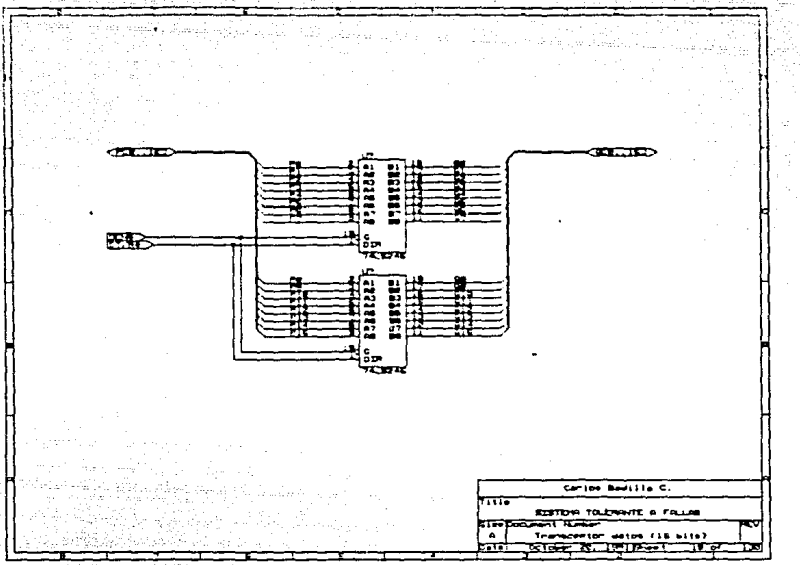
Este circuito es uno de los principales elementos en los que se sustenta la capacidad defensiva del sistema ante la presencia de fallas; está constituido por dos circuitos de votación, un manejador de bus de entrada y otro de bus de salida. (lámina 3).

La tarea de este módulo consiste en examinar la información de salida producida por los microprocesadores que forman el procesador triple con el fin de detectar posibles errores en su operación.

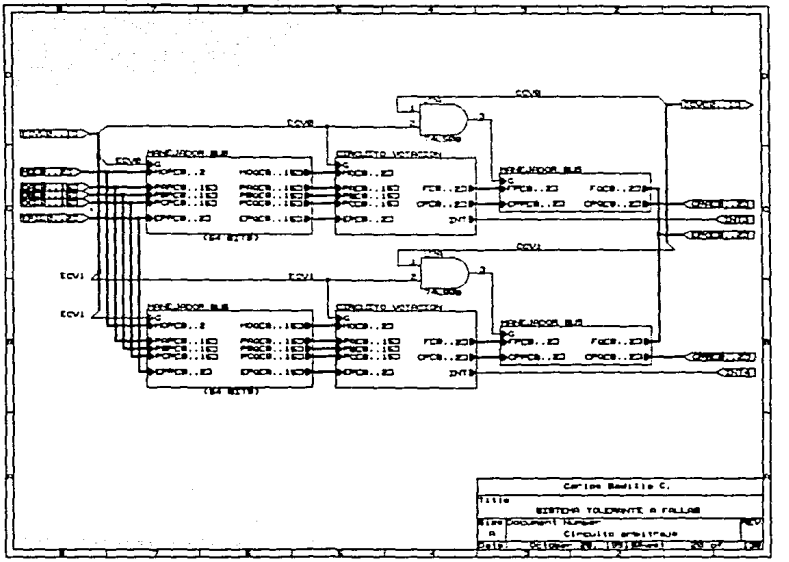
Los resultados de ese proceso de análisis son almacenados en la sección de registros de reconfiguración y en los registros de salida de los circuitos de votación.

La información producida por el circuito de arbitraje es utilizada para redefinir el estado de operación de los procesadores, para gobernar los controladores del bus del sistema y para guiar la ejecución de las rutinas de tolerancia a fallas.

Los manejadores de bus ubicados a la entrada y a la salida de los circuitos de votación son usados para aislar o habilitar los procesos de votación de acuerdo con la información almacenada en los registros de configuración de procesadores y de circuitos de votación.



LAMINA 19. TRANSCEPTOR DE DATOS



LAMINA 20. CIRCUITO DE ARBITRAJE

5.2. MANEJADOR DEL BUS DE ENTRADA

Esta sección del circuito de arbitraje consiste en un manejador de bus de 54 bits (lámina 22) el cual está constituido por dos manejadores de 3 bits y uno de 48 bits.

Los manejadores de 3 bits se usan para conectar las salidas de los registros de configuración de procesadores (estado y modo de operación) al circuito de votación.

El manejador de 48 bits se emplea para conectar las salidas de datos de los procesadores a los circuitos de votación (tres procesadores con 16 líneas de datos cada uno).

Estos manejadores son habilitados mediante la línea de estado de circuitos de votación. Normalmente a ambos circuitos de votación se les asigna el estado activo por lo que los respectivos transceptores están habilitados.

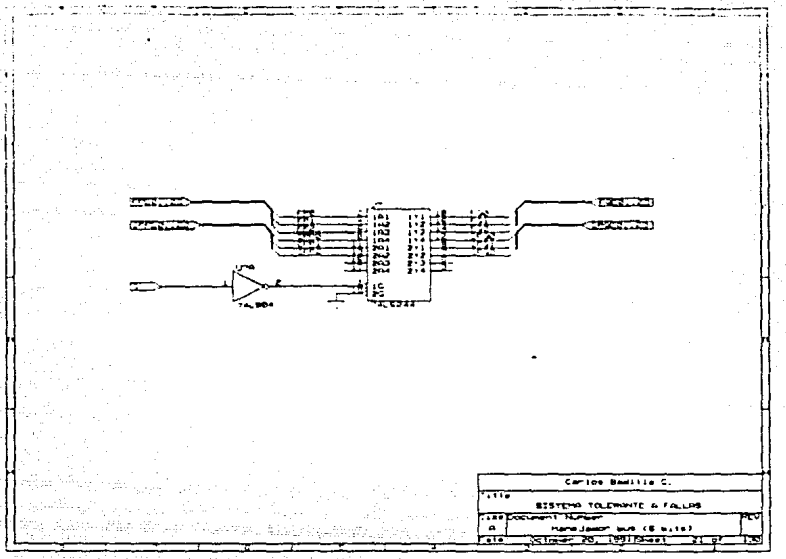
5.3. MANEJADOR DEL BUS DE SALIDA

El manejador de bus de salida (lámina 21) es un transceptor (buffer) que permite enlazar las líneas de salidas de los circuitos de votación con las entradas de los circuitos de salidas de comparadores de circuitos de votación (A, B), y con las entradas de datos del registro de estado de procesadores.

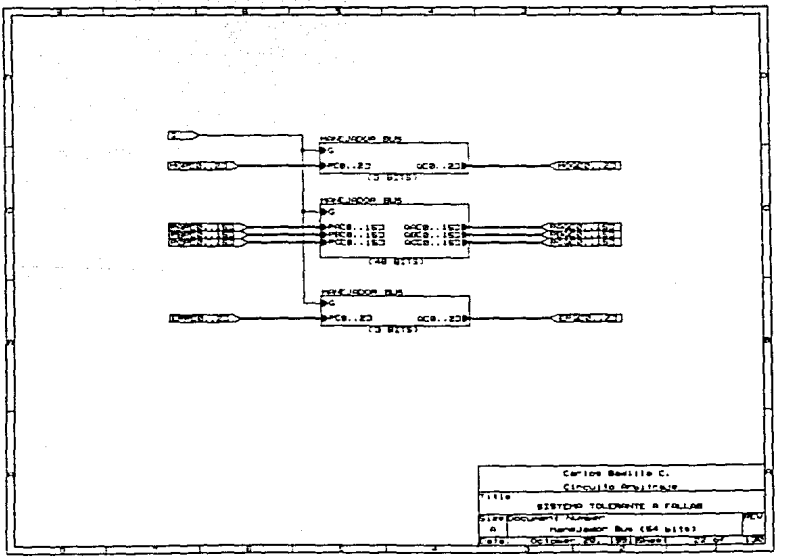
Cada circuito de votación tiene acceso a un diferente registro de salidas de comparadores. Aunque ambos tienen acceso al registro de estado de procesadores, solo el circuito votador con categoría principal está autorizado para modificar el contenido de dicho registro.

5.4. CIRCUITO DE VOTACION

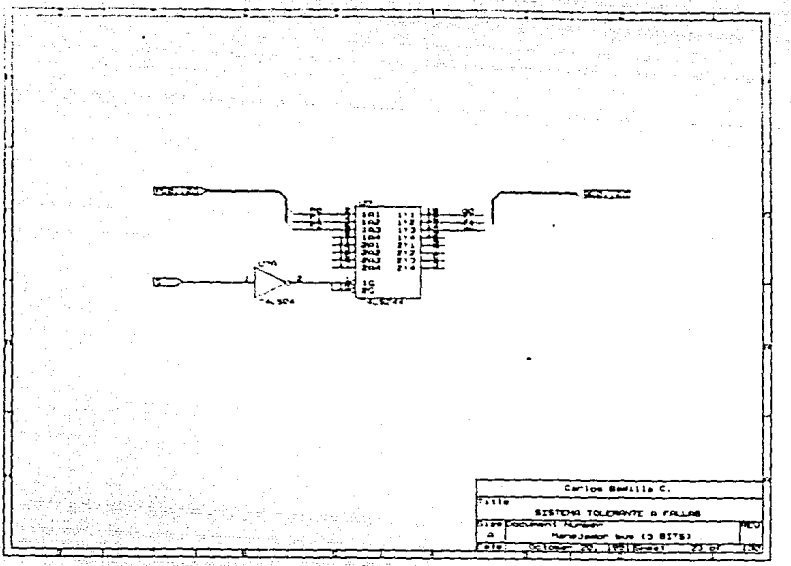
El computador diseñado cuenta con dos circuitos de votación (lámina 20). Cada uno de ellos consta de dos partes: un módulo de comparadores y un generador de interrupción (lámina 26).



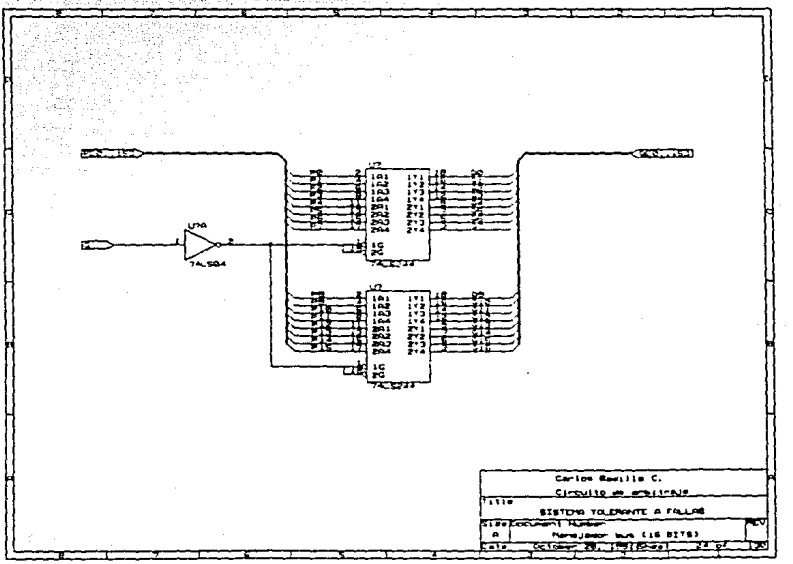
LAMINA 21. MANEJADOR DE BUS



LAMINA 22. MANEJADOR DE BUS



LAMINA 23. MANEJADOR DE BUS



LAMINA 24. MANEJADOR DE BUS

El circuito de votación tiene como propósito comparar los datos de salida de los procesadores para determinar posibles errores con base en el principio de votación de mayoría.

La operación de los circuitos votadores se controla mediante el contenido del registro de estado de circuitos de votación.

Si un circuito votador tiene estado activo, puede comparar las salidas de los procesadores y generar solicitudes de interrupción en casos de falla. No obstante, solo el que tenga categoría principal puede modificar el contenido del registro de estado de los procesadores para deshabilitar los procesadores detectados como fallados.

En caso de detectar diferencias en los datos de salida de los procesadores, el circuito de votación debe realizar las siguientes acciones:

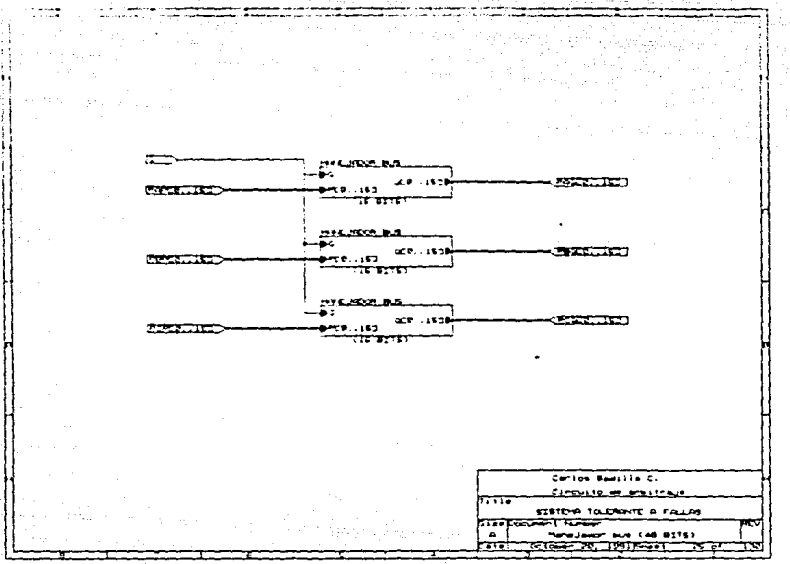
a) Generar una solicitud de interrupción a los procesadores para indicar la presencia de un error;

b) Almacenar el valor de las salidas de los comparadores internos en el registro definido para tal fin.

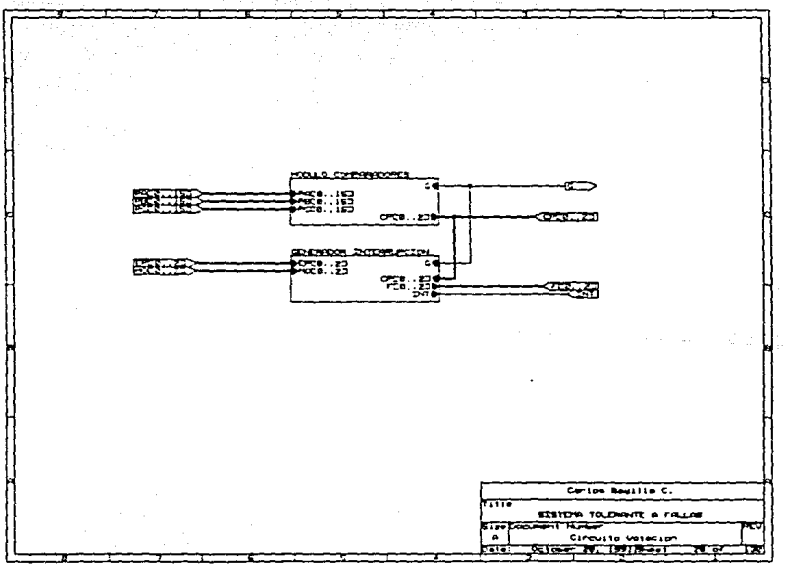
5.4.1. MODULO DE COMPARADORES

Este circuito está formado por tres comparadores de 16 bits cada uno, los cuales se encargan de cotejar los valores presentes en los buses de salida de los procesadores (lámina 27). En caso de error (disparidad en los datos confrontados), los valores presentes en las salidas de igualdad ("P=Q") de los circuitos de comparación son almacenados en el "registro de salidas de comparadores A" (láminas 60..62), si el error es detectado por el circuito de votación 0 (cero), y en el "registro de salidas de comparadores B" (láminas 63..65), si el error es detectado por el circuito de votación 1 (uno)

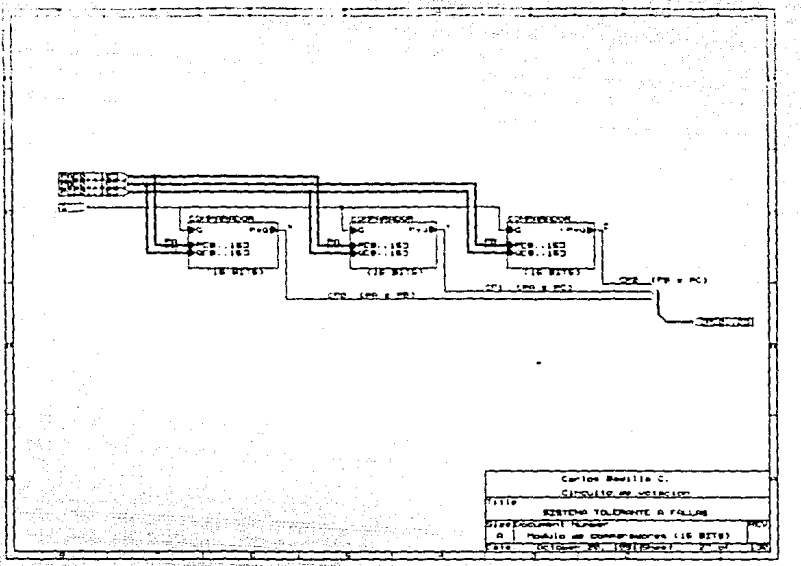
Dichos registros son accesibles a los procesadores y su contenido puede ser utilizado para dirigir las rutinas de dignóstico de fallas.



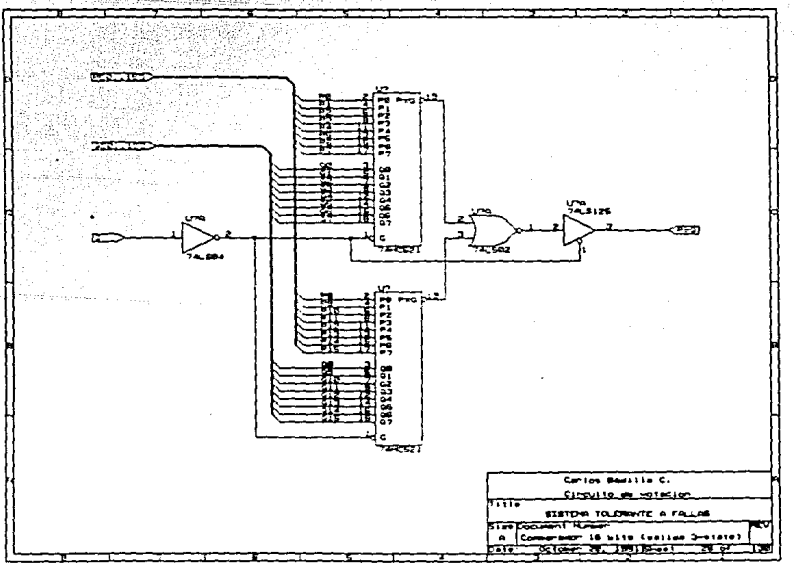
LAMINA 25. MANEJADOR DE BUS



LAMINA 26. CIRCUITO DE VOTACION



LAMINA 27. MODULO DE COMPARADORES



LAMINA 28. COMPARADOR DE 16 BITS

El contenido de este registro y el de los registros de configuración de procesadores (estado, categoria y modo de operación) son utilizados por las rutinas de diagnóstico para determinar la causa (falla) de diversos tipos de errores.

La función de cada uno de los registros de configuración de procesadores se explica en el capitulo 6.

5.4.2. GENERADOR DE INTERRUPCION

Este circuito es implantado mediante un arreglo lógico programable (PAL10H8) y es utilizado para detectar y indicar (mediante generación de una solicitud de interrupción) las condiciones de falla (errores) en las salidas de procesadores detectadas durante la ejecución del sistema.

Utiliza como entradas las variables de estado de procesadores (activo, inactivo#), las de modo de operación de procesadores (maestro, esclavo#) y las salidas de igualdad de los comparadores (x,y,z), las cuales son conocidas también como el bus interno CP[0..2]

El generador de interrupción produce las siguientes salidas:

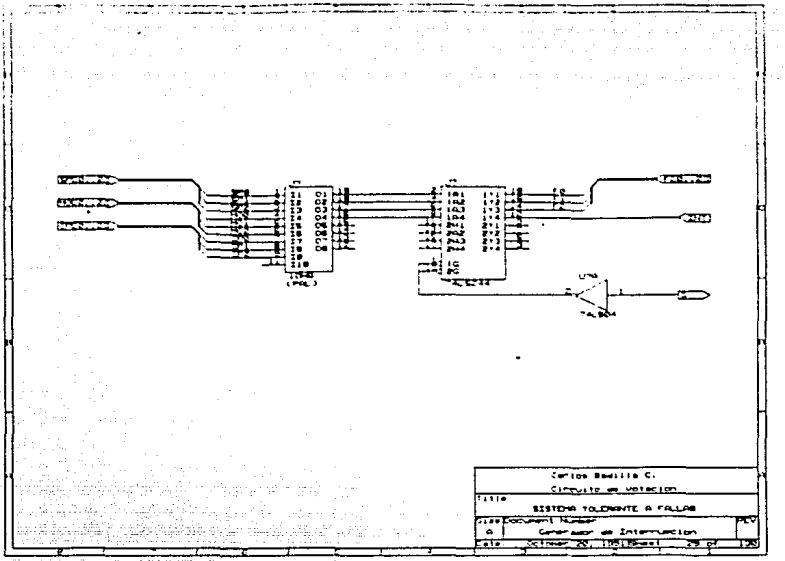
a) Las líneas F[0..2] utilizadas para modificar el contenido del registro de estado de procesadores.

b) Las líneas de solicitud de interrupción INT[3..4];

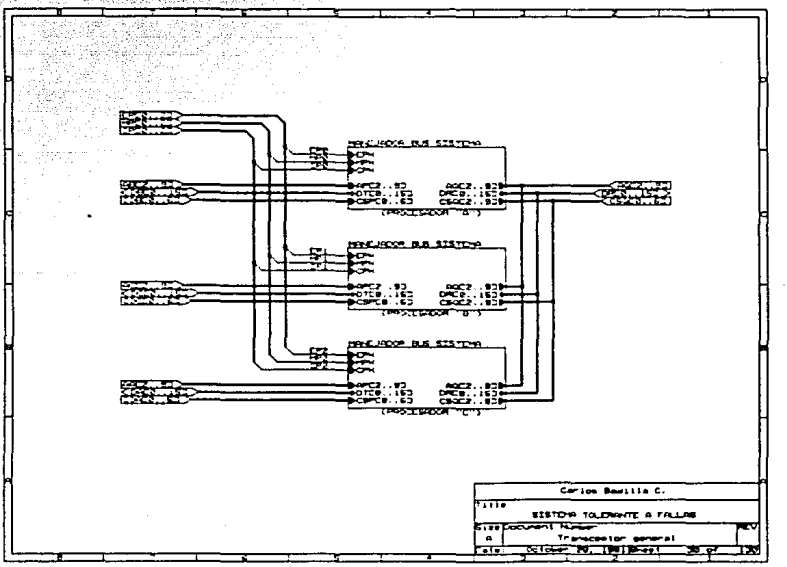
5.4.2.1. LINEAS DE MODIFICACION DE ESTADO DE PROCESADORES F[0..2]

Estas líneas son utilizadas para desactivar a un procesador detectado como fallado durante el desarrollo del proceso de votación.

El contenido de estas líneas es almacenado en el registro de estado de procesadores por medio de la línea de solicitud interrupción en el momento en que se detecte una falla



LAMINA 29. GENERADOR DE INTERRUPCION



LAMINA 30. TRANCEPTOR GENERAL

Las líneas F[0..2] se diferencian de las líneas de salida de los comparadores y de la línea de solicitud de interrupción en que, solo las que proceden del votador principal, tienen efecto fuera del circuito de votación. Esto es debido a que están separadas del sistema por medio de los transceptores de aislamiento del circuito de votación (lámina 20).

En consecuencia, solamente las líneas que proceden del votador con categoría principal pueden modificar el contenido del registro de estado de procesadores.

5.4.2.2. LINEAS DE SOLICITUD DE INTERRUPCION INT[3..4]

Líneas utilizadas por los circuitos de votación (principal y respaldo) para solicitar la interrupción de la tarea actual, a efecto de que se ejecute una rutina de diagnóstico de fallas, en el momento de detectar una anomalía (error) en alguna de las salidas de los procesadores.

A cada circuito de votación está asociada una línea de solicitud de interrupción: la línea INT[3] se conecta al circuito de votación cero (0) y la línea INT[4] al circuito de votación uno (1). Cada una de estas líneas está conectada como entrada a la unidad controladora de interrupciones.

En caso de detectarse un error, tanto la línea INT[3] como INT[4], pueden solicitar la atención del procesador independientemente de la categoría (principal o respaldo) en que se encuentre el circuito de votación que detectó el error.

Esto es posible debido a que las líneas de solicitud de interrupción generadas por los votadores están conectadas directamente a la unidad controladora de interrupciones (no pasan por los circuitos transceptores del circuito de votación).

5.4.3. FUNCIONAMIENTO DEL CIRCUITO DE VOTACION

En esta sección se explican, con base en un diagrama de flujo que representa el comportamiento lógico del módulo generador de interrupciones, las diferentes situaciones de falla a nivel del procesador que puede detectar el circuito de votación.

5.4.3.1 VARIABLES USADAS PARA EFECTUAR EL PROCESO DE VOTACION

Para determinar la función de ejecución correcta o incorrecta de las operaciones de salida del procesador, el circuito combinacional de votación opera, en forma concurrente, sobre seis variables de entrada y tres variables adicionales que son generadas internamente por el módulo de comparadores.

a) VARIABLES DE ENTRADA AL CIRCUITO DE ARBITRAJE

i) Líneas de estado de procesadores (activo - inactivo#)
SP0, SP1, SP2

ii) Líneas de modo de operación de procesadores (maestro - esclavo#)
M00, M01, M02

b) VARIABLES INTERNAS

i) Líneas de salida de los comparadores de procesadores
x, y, z

Las variables antes mencionadas pueden combinarse, para obtener 3 variables intermedias adicionales no necesarias desde el punto de vista electrónico pero que permiten explicar, en forma más clara el funcionamiento del circuito de votación.

Las variables utilizadas para describir la operación del circuito votador son las siguientes:

f, g, h, k: Variables usadas para representar el estado de los procesadores donde

f = SP0 & SP1 & SP2
g = SP0
h = SP1
k = SP2

p,q,r,s: Variables usadas para representar el modo de operación de los procesadores donde

```
p = MOO & MO1 & MO2
q = MOO
r = MO1
s = MO2
```

w,x,y,z: Variables usadas para representar el estado de las salidas de los comparadores donde

```
w = x & y & z
x ===> Salida PO = Salida P1
y ===> Salida PO = Salida P2
z ===> Salida P1 = Salida P2
```

Estas 12 variables son utilizadas, en las siguientes secciones para explicar, el comportamiento combinacional del circuito de votación.

5.4.3.2 PROCESO DE VALIDACION DE OPERACION CORRECTA

Para deducir las ecuaciones que representan condiciones de falla se ha construido un árbol binario, presentado en la Figura 5.1, que simboliza "un diagrama de flujo" del circuito de votación.

En dicho esquema, las ecuaciones de falla pueden deducirse como funciones de las variables (f,g,h,k), (p,q,r,s,), (w,x,y,z) o como funciones de las variables (SPO,SP1,SP2), (MOO, MO1, MO2), (x,y,z) siguiendo las trayectorias desde el "nodo raíz" hasta cada una de las "hojas" que representan condiciones de falla en el árbol binario.

Para fines descriptivos, la deducción de condiciones de falla se presenta como un proceso de decisiones secuenciales (diferidas en el tiempo). No obstante, las variables utilizadas para inferir fallas son analizadas combinacionalmente (en forma concurrente) por el circuito de votación.

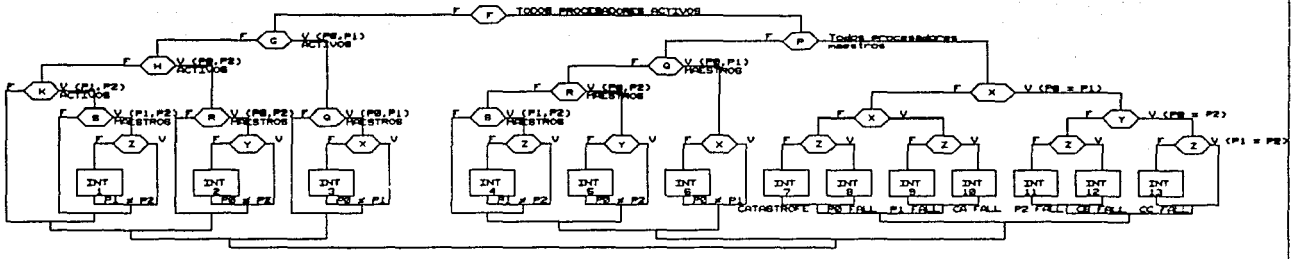


Figura 5.1. Diagrama de flujo de las condiciones de falla detectadas por el circuito de votación.

Estatus procesadores (activo / inactivo#)	Modo Operac. Proces. (maestro / esclavo#)	Salidas comparadores (igual / diferente#)
$f = P_0\ P_1\ P_2$	$p = P_0\ P_1\ P_2$	$w = x\ y\ z$
$g = P_0\ P_1$	$q = P_0\ P_1$	$x : P_0 = P_1$
$h = P_0\ P_2$	$r = P_0\ P_2$	$y : P_0 = P_2$
$k = P_1\ P_2$	$s = P_1\ P_2$	$z : P_1 = P_2$

El siguiente fragmento de programa, escrito en pseudo código, representa el esquema de decisiones seguido por el circuito de votación para detectar errores en los procesadores y en los comparadores. El diagrama de flujo correspondiente a este proceso se presenta en la figura 5.1

```
PROCESO DE VALIDACION DE OPERACION CORRECTA
BEGIN
IF (SP0 & SP1 & SP2)  { f = todos los procesadores activos }

{Análisis del subárbol derecho del flujograma:
 Sección correspondiente a todos los procesadores activos}

THEN IF (M00 & M01 & M02) { p = todos los procesadores maestros }
     THEN IF (x) {P0 = P1}
          THEN IF (y) {P0 = P2}
               THEN  IF (z) {P1 = P2}
                     THEN  CT {Salidas iguales en 3 procesadores
                                activos y maestros}
                     ELSE  INT-13 {Comparador C en mal estado}
               ELSE IF (z) {P1 = P2}
                    THEN  INT-12 {Comparador B en mal estado}
                    ELSE  INT-11 {Procesador 2 en mal estado}
                          SP2 < — 0 {Apaga bit 2 del registro
                                     de estado de procesadores}
          ELSE IF (y) {P0 = P2}
               THEN IF (z) {P1 = P2}
                    THEN  INT-10 {Comparador B en mal estado}
                    ELSE  INT-9  {Procesador 1 en mal estado}
                          SP1 < — 0 {Apaga bit 1 del registro
                                     de estado de procesadores}
               ELSE IF (z) {P1 = P2}
                    THEN  INT-8  {Procesador 0 en mal estado}
                          SP0 < — 0 {Apaga bit 0 del registro
                                     estado de procesadores}
                    ELSE  INT-7  {Estado catastrófico}
     THEN IF (M00 & M01) { q = procesadores P0, P1 maestros}
          THEN IF (x) {P0 = P1}
               THEN  CT    {Salidas iguales en procesadores
                            P0, P1 maestros}
               ELSE  INT-6 {Salidas diferentes en procesadores
                            P0, P1 maestros}
          ELSE IF (M00 & M02) { r = procesadores P0, P2 maestros}
               THEN IF (y) {P0 = P2}
                    THEN  CT    {Salidas iguales en procesadores
                                 P0, P2 maestros}
                    ELSE  INT-5 {Salidas diferentes en procesad.
                                 P0, P2 maestros}
               ELSE IF (M01 & M02) { r = proces. P1, P2 maestros}
                    THEN IF (z) {P1 = P2}
                         THEN  CT {Salidas iguales en procesadores
                                   P1, P2 maestros}
```

```
                                ELSE  INT-4 {Salidas diferentes en
                                           procesad. P1, P2 maestros}
-------------------------------------------------------------------

{Análisis del subárbol izquierdo del flujograma:
 Sección correspondiente a uno o dos procesadores activos}

ELSE IF (SP0 & SP1 )  { g = procesadores P0, P1 activos }
     THEN IF (M00 & M01) { q = procesadores P0, P1 maestros}
          THEN IF (x) {P0 = P1}
               THEN  CT    {Salidas iguales en procesadores
                            P0, P1 maestros}
               ELSE  INT-3 {Salidas diferentes en procesadores
                            P0, P1 maestros}
          ELSE  CT {Un procesador maestro y otro esclavo}
     ELSE IF (SP0 & SP2 )  { h = procesadores P0, P2 activos }
          THEN IF (M00 & M02) { r = procesadores P0, P2 maestros}
               THEN IF (y) {P0 = P2}
                    THEN  CT    {Salidas iguales en procesadores
                                 P0, P2 maestros}
                    ELSE  INT-2 {Salidas diferentes en procesad.
                                 P0, P2 maestros}
               ELSE  CT {Un procesador maestro y otro esclavo}
          ELSE IF (SP1 & SP2 )  { k = procesadores P1, P2 activos }
               THEN IF (M01 & M02) { s = proces. P1, P2 maestros}
                    THEN IF (z) {P1 = P2}
                         THEN  CT {Salidas iguales en procesadores
                                   P1, P2 maestros}
                         ELSE  INT-1 {Salidas diferentes en
                                      procesad. P1, P2 maestros}
                    ELSE CT {Un proces. maestro y otro esclavo}
               ELSE  CT {Un procesador maestro y otro esclavo}

END; {PROCESO DE VALIDACION DE OPERACION CORRECTA}
```

5.4.4. ECUACIONES BOOLEANAS DEL GENERADOR DE INTERRUPCION

En esta sección se especifican las ecuaciones de las trayectorias, en un recorrido de izquierda a derecha, que identifican condiciones de falla en el árbol binario que representa al circuito de votación.

Estas ecuaciones representan el contenido del programa a utilizar para definir la estructura interna del arreglo lógico programable (PAL combinacional) requerido para implementar el generador de interrupción del circuito de votación.

De acuerdo con esta especificación, se necesita una PAL mínima de 9 entradas y 13 salidas. El tamaño de este circuito deberá ser mayor si se decide implementar los comparadores dentro de la misma PAL en vez de utilizar circuitos independientes. En la propuesta de circuito se emplea la PAL 10H8 (10 entradas x 8 salidas).

Del total de posibles combinaciones de las variables utilizadas para realizar el proceso de validación de operación correcta, existen 13 funciones que permiten definir un conjunto razonable de condiciones previsibles de falla del sistema y que pueden ser utilizadas para generar interrupciones al procesador para ejecutar rutinas de diagnóstico que permitan identificar si la falla es transitoria o permanente.

Estas 13 funciones originan todas una solicitud de interrupción y son producidas mediante un arreglo lógico programable cuyo contenido, de ser necesario, podría modificarse para contemplar nuevas condiciones de falla.

De estas funciones existen 3 que corresponden a falla "inequívoca" en un procesador y que, por consiguiente, son utilizadas para modificar el registro de estado de procesadores.

Los valores de las variables de entrada al circuito de votación así como las salidas de los comparadores de procesadores pueden ser leídos por el procesador para diagnosticar la falla (determinar la causa del error) ocurrida en cada caso.

1. $(\overline{SP0} \cdot SP1 \cdot SP2) (M01 \cdot M02) \overline{Z}$
   Diferencia en las salidas de los procesadores P1 y P2

2. $(SP0 \cdot \overline{SP1} \cdot SP2) (M00 \cdot M02) \overline{Y}$
   Diferencia en las salidas de los procesadores P0 y P2

3. $(SP0 \cdot SP1 \cdot \overline{SP2}) (M00 \cdot M01) \overline{X}$
   Diferencia en las salidas de los procesadores P0 y P1

4. $(SP0 \cdot SP1 \cdot SP2) (\overline{M00} \cdot M01 \cdot M02) \overline{Z}$
   Diferencia en las salidas de los procesadores P1 y P2

5. $(SP0 \cdot SP1 \cdot SP2) (M00 \cdot \overline{M01} \cdot M02) \overline{Y}$
   Diferencia en las salidas de los procesadores P0 y P2

6. (SP0 . SP1 . SP2) (M00 . M01 . $\overline{M02}$) $\overline{X}$
   Diferencia en las salidas de los procesadores P0 y P1

7. (SP0 . SP1 . SP2) (M00 . M01 . M02) $\overline{X}$ $\overline{Y}$ $\overline{Z}$
   Estado catastrófico: Salidas diferentes en las salidas de todos los procesadores

8. (SP0 . SP1 . SP2) (M00 . M01 . M02) $\overline{X}$ $\overline{Y}$ Z
   Procesador 0 fallado: SP0 < — 0
   Apaga bit 0 del registro de estatus de procesadores

9. (SP0 . SP1 . SP2) (M00 . M01 . M02) $\overline{X}$ Y $\overline{Z}$
   Procesador 1 fallado: SP1 < — 0
   Apaga bit 1 del registro de estatus de procesadores

10. (SP0 . SP1 . SP2) (M00 . M01 . M02) $\overline{X}$ Y Z
    Comparador A fallado

11. (SP0 . SP1 . SP2) (M00 . M01 . M02) X $\overline{Y}$ $\overline{Z}$
    Procesador 2 fallado: SP2 < — 0
    Apaga bit 2 del registro de estatus de procesadores

12. (SP0 . SP1 . SP2) (M00 . M01 . M02) X $\overline{Y}$ Z
    Comparador B fallado

13. (SP0 . SP1 . SP2) (M00 . M01 . M02) X Y $\overline{Z}$
    Comparador C fallado

Cada una de las rutas que conduce a generar una solicitud de interrupción (ruta de falla) requiere el diseño de una rutina de servicio de interrupción especifica.

En principio, al atender una interrupción, el procesador debe utilizar la información presente en los registros de configuración [estado de procesadores (activo / inactivo#) y modo de operación (maestro / esclavo#)] y la del registro de salidas de comparadores, analizarla (seguir el árbol de decisiones del circuito de votación) y proceder en consecuencia.

La interpretación del significado asociado al contenido del registro de salidas de comparadores se indica en la Tabla 5.1.

El árbol binario del circuito de votación carece de secciones que sean equivalentes en sentido estricto. En consecuencia, no es válido establecer "enlaces de simplificación" para eliminar secciones del árbol.

Adicionalmente debe tenerse en cuenta que mediante la utilización de arreglos lógicos programables es posible implementar no solo las funciones lógicas de detección de errores, sino también los circuitos comparadores.

ENTRADAS			SALIDAS
X	Y	Z	SIGNIFICADO DEL CODIGO
0	0	0	Estado catastrófico
0	0	1	Procesador 0 en mal estado
0	1	0	Procesador 1 en mal estado
0	1	1	Comparador A en mal estado
1	0	0	Procesador 2 en mal estado
1	0	1	Comparador B en mal estado
1	1	0	Comparador C en mal estado
1	1	1	Sistema en buen estado

TABLA No. 5.1

Códigos de falla representados por las salidas de los comparadores (X, Y, Z)

CAPITULO 6

MODULO DE RECONFIGURACION DE PROCESADORES

INTRODUCCION

El adecuado funcionamiento de los mecanismos de tolerancia a fallas propuestos para este sistema se basa, en una proporción considerable, en el uso del concepto de parámetros de operación de los diferentes componentes de la arquitectura .

En este capitulo se describe el significado de los parámetros de operación del subsistema de procesamiento (procesadores, circuitos de votación y salidas de comparadores de circuitos de votación) y los circuitos empleados para almacenar tales indicadores.

6.1. PARAMETROS DE OPERACION DEL SISTEMA

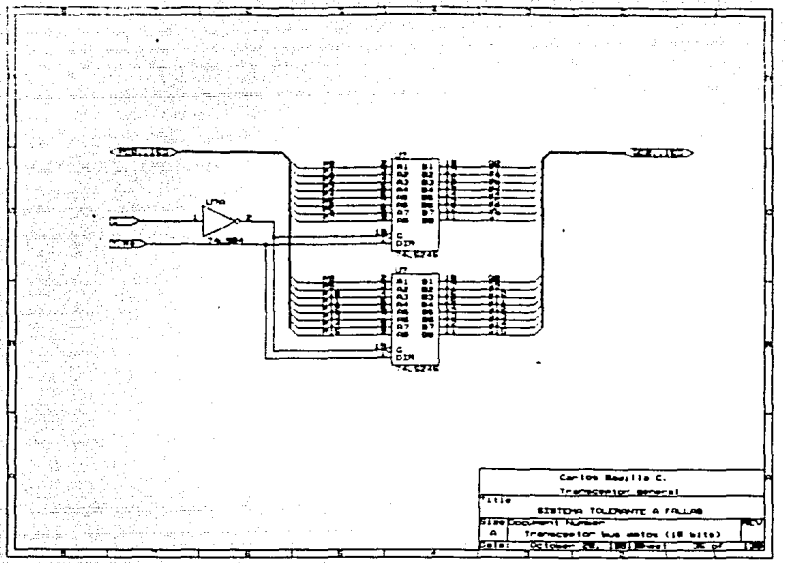
Los indicadores utilizados para representar la operación del subsistema de procesamiento son almacenados en un módulo de registros de configuración (lámina 36). Este módulo incluye:

a. Registros de configuración de procesadores
b. Registros de configuración de circuitos votadores
c. Registros de salidas de comparadores de los circuitos de votación

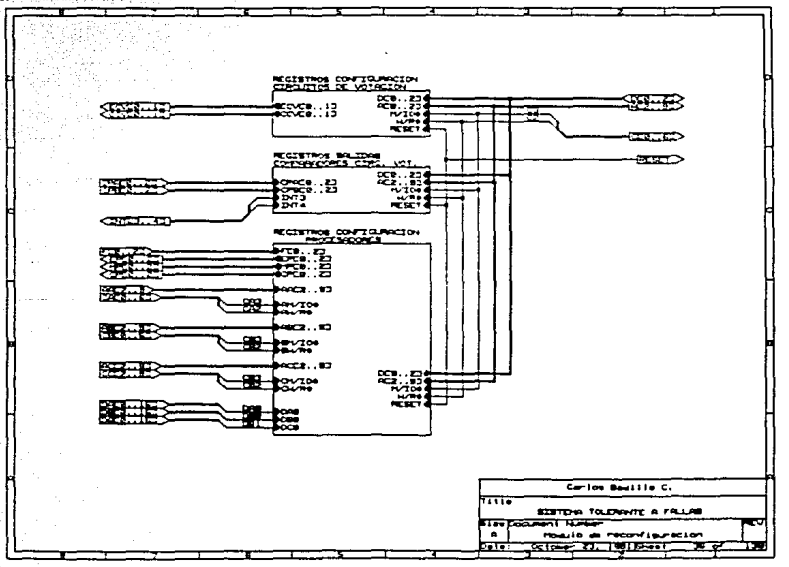
Esta información se utiliza para realizar el proceso de votación, para definir cuál procesador tiene acceso a los buses del sistema y para conducir los procesos de diagnóstico de fallas.

6.2. PARAMETROS DE OPERACION DE PROCESADORES

La operación de cada procesador está caracterizada por medio de tres parámetros: estado, categoría y modo de operación.



LAMINA 35. TRANSCEPTOR DE BUS DE DATOS



LAMINA 36. MODULO DE RECONFIGURACION

6.2.1. ESTADO DE OPERACION

Un procesador puede encontrarse en uno de dos posibles estados: activo o inactivo#. El estado en que se encuentra un procesador se indica por medio del registro de estado de procesadores.

Un procesador está activo cuando se encuentra ejecutando alguna tarea bajo control del sistema operativo. Un procesador activo puede estar en modo de operación maestro o esclavo# y puede tener categoría principal o respaldo#.

Si un procesador está activo y se encuentra en modo de operación maestro sus salidas son tomadas en cuenta para realizar el proceso de votación (está ejecutando la tarea principal).

Normalmente cuando un procesador es desactivado por uno de los circuitos de votación, la correspondiente rutina de interrupción debe contener instrucciones para hacerlo entrar en un proceso de autodiagnóstico. Durante este periodo, los datos que produzca no son tomados en cuenta por los circuitos de votación.

6.2.2. CATEGORIA DE OPERACION

Cada procesador activo trabaja en una de dos posibles categorías: principal o respaldo#. La categoría en que se encuentre un procesador se indica por medio del registro de categoría de procesadores.

Si un procesador opera en categoría principal puede colocar información (datos, control, direcciones) sobre los buses de salida del módulo de procesamiento (sus salidas se utilizan para gobernar los dispositivos periféricos)

En el sistema solo puede haber un procesador que opere en categoría principal.

Cuando un procesador opera en estado activo y categoría de respaldo# se cumple lo siguiente:

- recibe los mismos datos entrantes que el procesador principal;
- ejecuta las mismas operaciones que el procesador principal
- los resultados que produce se toman en cuenta en el proceso de votación
- sus salidas no tienen acceso a los buses del sistema

Si un procesador está en modo de operación inactivo#, la calificación de principal o respaldo# es irrelevante.

6.2.3. MODO DE OPERACION

Cada procesador activo puede trabajar en uno de dos posibles modos de operación: maestro o esclavo#. El modo de operación en que se encuentre un procesador se indica por medio del registro de modo de operación de procesadores.

Si un procesador opera en modo de operación maestro y se le ha asignado categoría principal, dicho procesador puede colocar información (datos, control, direcciones) sobre los buses de salida del módulo de procesamiento.

Cuando un procesador activo está en modo de operación esclavo# se cumple lo siguiente:

- puede recibir ("escuchar") la información de entrada presente en los buses del sistema;
- sus salidas no se toman en cuenta para el proceso de votación;
- la categoría de operación (principal o respaldo#) es irrelevante
- sus salidas no tienen acceso a los buses del sistema

Típicamente un procesador entra en modo de operación esclavo# para ejecutar una rutina de autodiagnóstico y luego recibir información (datos e instrucciones) desde el procesador con categoría principal, en caso no haber encontrado fallas en su operación.

La transferencia de información de un procesador en modo de operación maestro a uno en modo esclavo# puede realizarse por ejemplo para recargar un programa que se haya dañado. Estas operaciones de transferencia deben hacerse bajo control del sistema operativo.

Si un procesador está en modo de operación inactivo#, la calificación de maestro o esclavo# es irrelevante.

6.3. REGISTROS DE CONFIGURACION DE PROCESADORES

Referencia: lámina 37.

Los registros de configuración de procesadores se caracterizan porque pueden operar como dobles puertos: un "puerto externo" y un "puerto interno"

El "puerto externo" es accesible al procesador con categoría principal tanto para operaciones de lectura como de escritura. Permite a dicho procesador modificar los parámetros de cualesquiera procesadores.

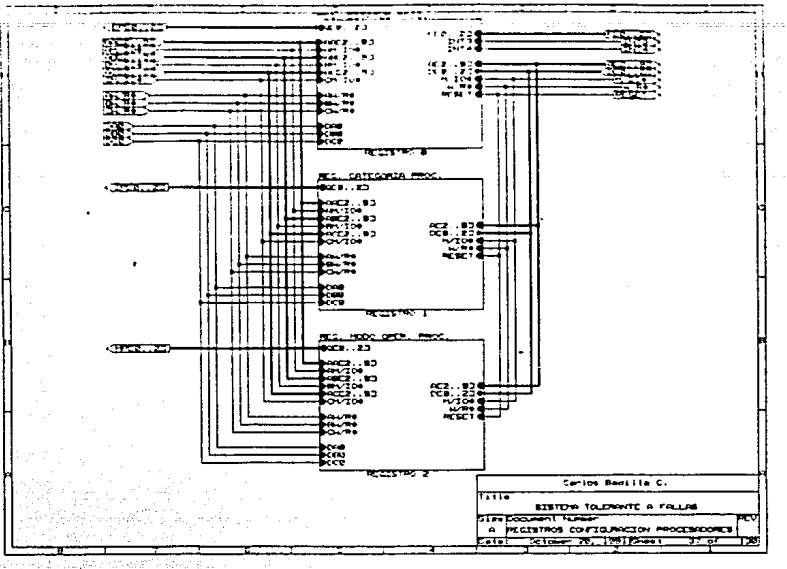
El "puerto interno" es accesible a todos los procesadores, para operaciones de escritura, independientemente de su categoría, bajo la restricción de poder modificar únicamente el bit que tengan asociado en el puerto accesado.

Las salidas de los registros de configuración de procesadores son de tipo "latch". Esta característica permite tener disponible, en forma permanente, sus estados lógicos durante los procesos de votación.

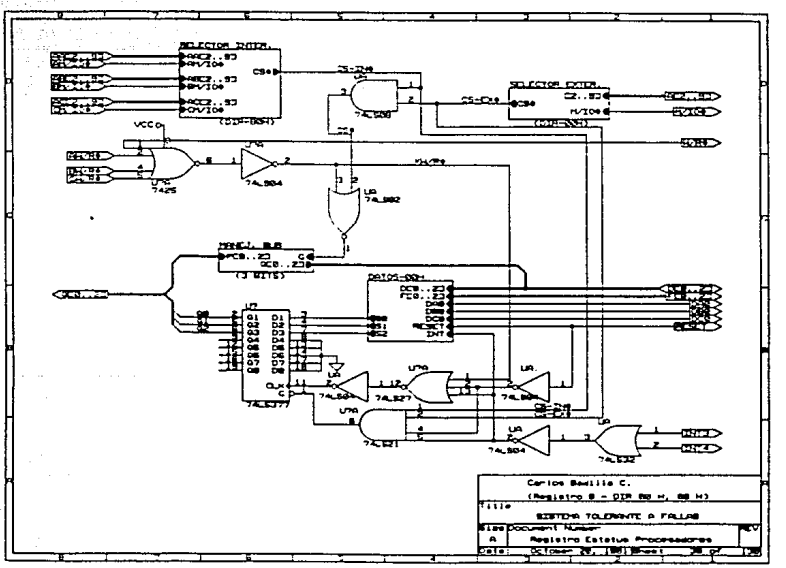
6.3.1. REGISTRO DE ESTADO DE PROCESADORES

Registro número 0. Como puerto externo ocupa la localidad 00 H del espacio de puertos del sistema y como puerto interno la localidad 80 H (láminas 38..42).

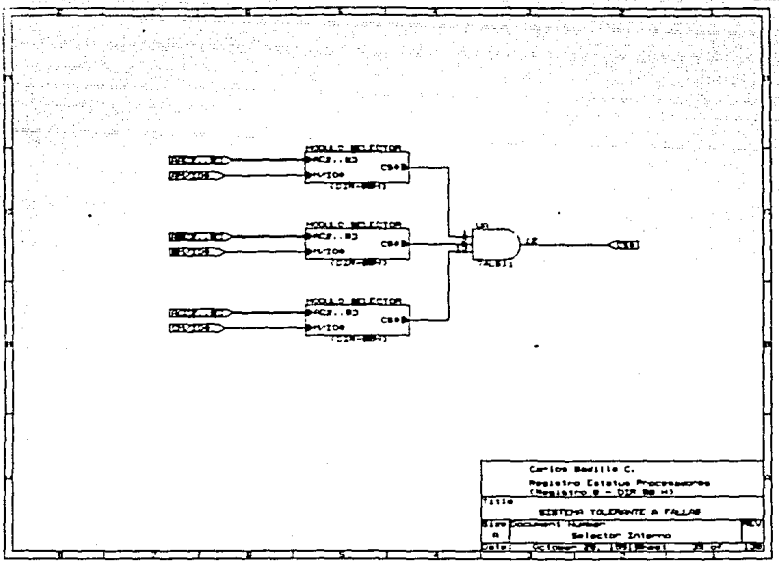
Tiene 3 bits de ancho, cada uno de ellos asociado a uno de los procesadores (Figura 6.1).



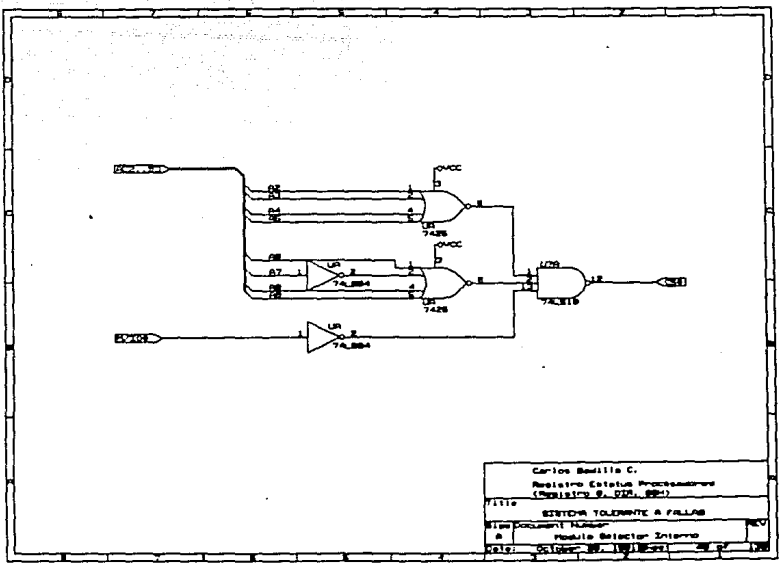
LAMINA 37. REGISTROS DE CONFIGURACION DE PROCESADORES



LAMINA 38. REGISTRO DE ESTATUS DE PROCESADORES



LAMINA 39. SELECTOR INTERNO



LAMINA 40. MODULO SELECTOR INTERNO

El contenido de este registro informa acerca del estado de operación (activo - inactivo#) en que se encuentra cada uno de los procesadores.

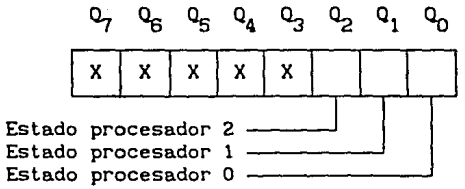


Figura 6.1. Registro de estado de procesadores

La información de este registro puede ser establecida de cuatro maneras diferentes (lámina 42):

a. Por medio del circuito de "reset"

Durante la rutina de iniciación todos los procesadores son declarados activos: se escribe un "uno" en todos sus bits.

b. Por medio del circuito de votación

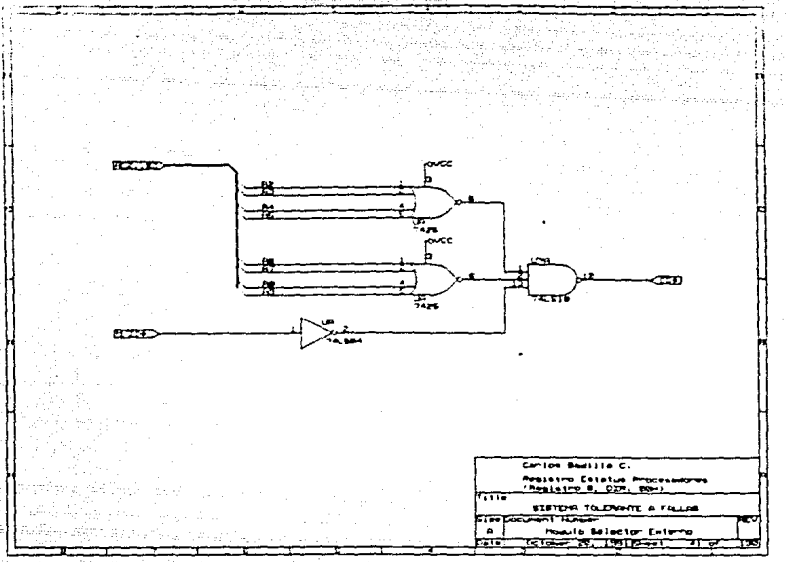
El estado de un procesador puede ser modificado, durante los procesos de votación, por las salidas F[0..2] del circuito de votación. Esto se realiza si se detecta que el valor de sus salidas difiere de las restantes; el procesador se declara como fallado borrando el bit correspondiente en el registro de estado.

c. Por medio de propio procesador

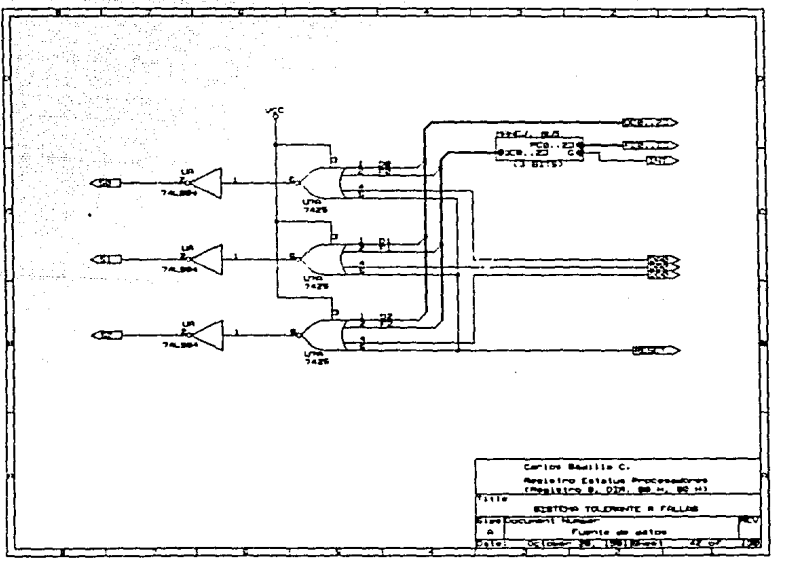
Cada procesador puede modificar su propio estado de operación escribiendo sobre el puerto interno asociado al registro de estado de procesadores.

d. Por medio del procesador principal

El procesador que opere en categoría principal puede modificar el estado de cualquiera de los procesadores escribiendo sobre el puerto externo asociado a dicho registro.



LAMINA 41. MODULO SELECTOR EXTERNO



LAMINA 42. FUENTE DE DATOS

6.3.2. REGISTRO DE CATEGORIA DE PROCESADORES

Registro número 1. Como puerto externo ocupa la localidad 04 H del espacio de puertos del sistema y como puerto interno la localidad 84 H (láminas 43..47).

Tiene 3 bits de ancho, cada uno de ellos asociado a uno de los procesadores (Figura 6.2).

El contenido de este registro informa acerca de la categoría de operación (principal / respaldo#) en que se encuentra cada uno de los procesadores.

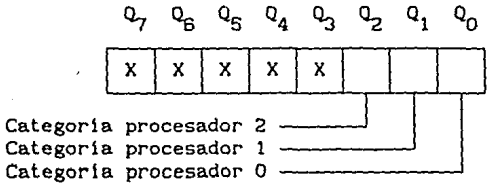


Figura 6.2. Registro de categoría de procesadores

La información de este registro puede ser establecida de tres maneras diferentes (lámina 47):

a. Por medio del circuito de "reset"

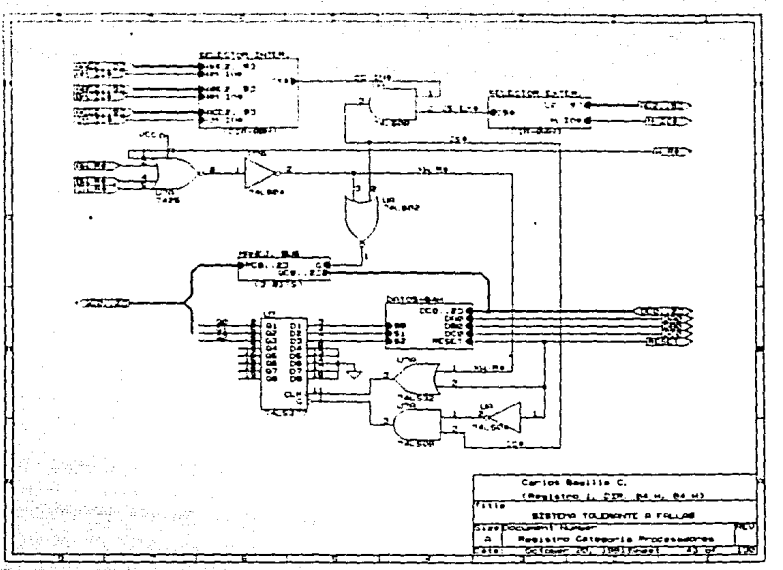
Durante la rutina de iniciación se declara principal al procesador 0 (cero) y como respaldo# a los procesadores 1 y 2: se escribe un "uno" en el bit 0 y "ceros" en los bits 1 y 2.

b. Por medio de propio procesador

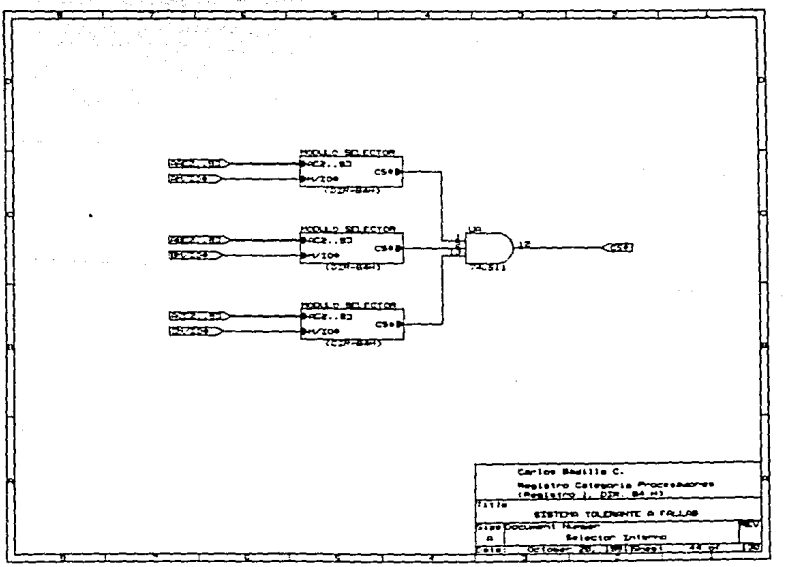
Cada procesador puede modificar su propia categoría de operación escribiendo sobre el puerto interno asociado al registro de categoría de procesadores.

c. Por medio del procesador principal

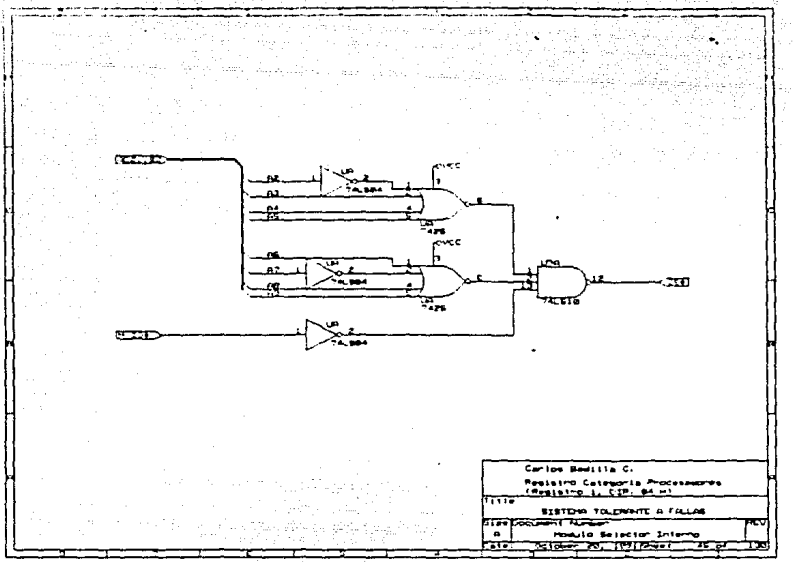
El procesador que opere en categoría principal puede modificar la categoría de cualquiera de los procesadores escribiendo sobre el puerto externo asociado a dicho registro.



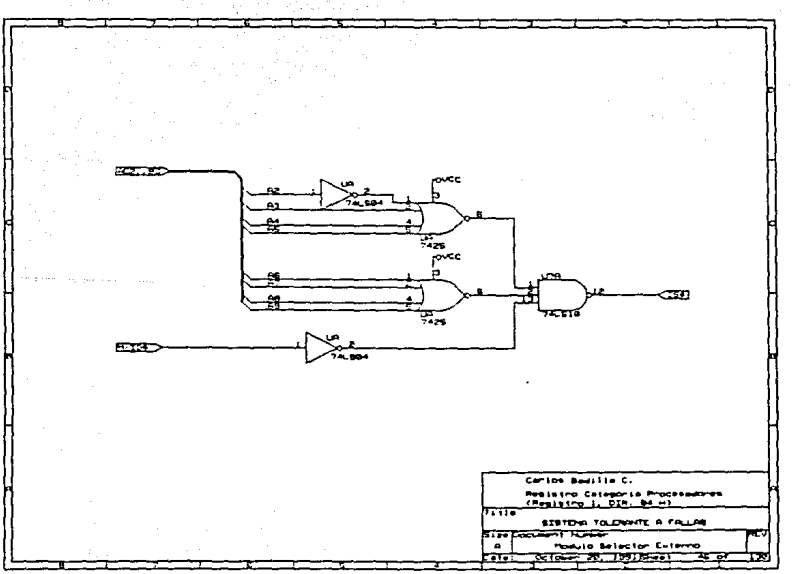
LAMINA 43. REGISTRO DE CATEGORIA DE PROCESADORES



LAMINA 44. SELECTOR INTERNO



LAMINA 45. MODULO SELECTOR INTERNO



LAMINA 46. MODULO SELECTOR EXTERNO

6.3.3. REGISTRO DE MODO DE OPERACION DE PROCESADORES

Registro número 2. Como puerto externo ocupa la localidad 08 H del espacio de puertos del sistema y como puerto interno la localidad 88 H (láminas 48..52).

Tiene 3 bits de ancho, cada uno de ellos asociado a uno de los procesadores.

El contenido de este registro informa acerca del modo de operación (maestro, esclavo#) en que se encuentra cada uno de los procesadores.

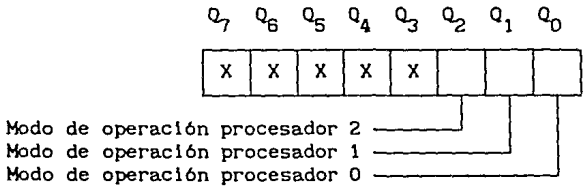


Figura 6.3 Registro de modo de operación de procesadores

La información de este registro puede ser establecida de tres maneras diferentes (lámina 52):

a. Por medio del circuito de "reset"

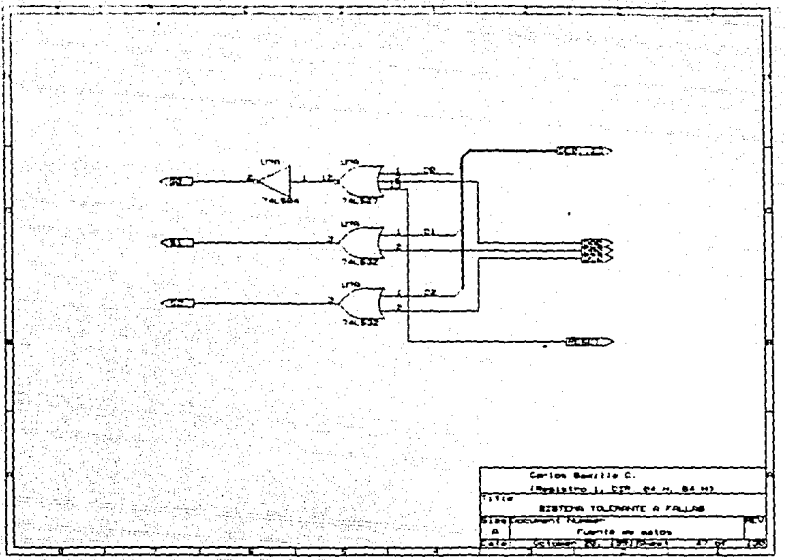
Durante la rutina de iniciación todos los procesadores son declarados maestros: se escribe un "uno" en todos sus bits.

b. Por medio del propio procesador

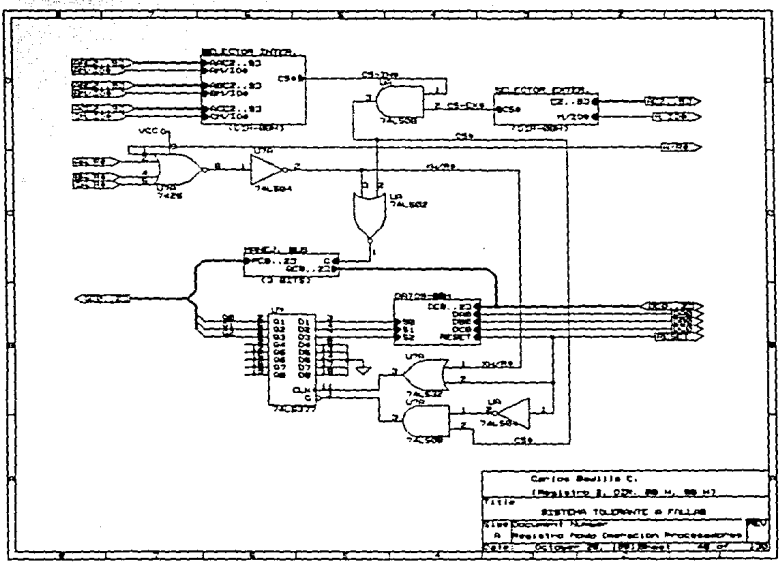
Cada procesador puede modificar su propio modo de operación escribiendo sobre el puerto interno asociado al registro de modo de operación de procesadores.

c. Por medio del procesador principal

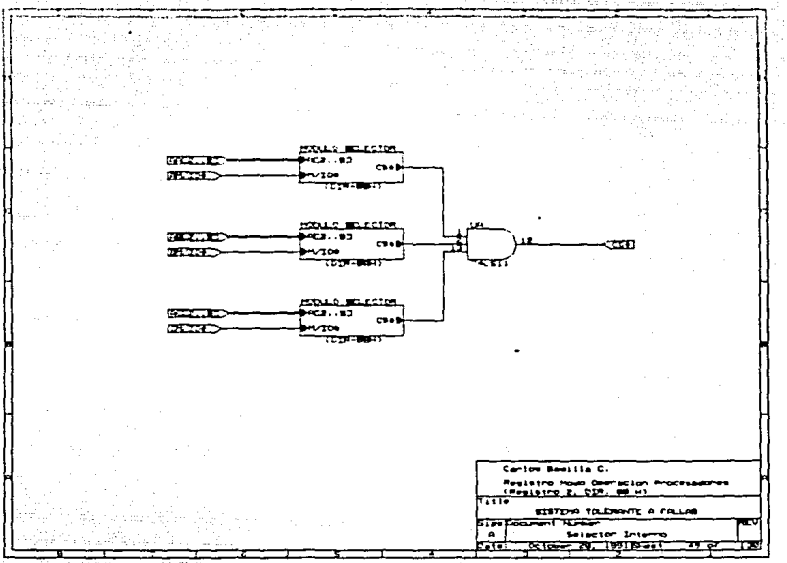
El procesador que opere en categoría principal puede modificar el modo de operación de cualquiera de los procesadores escribiendo



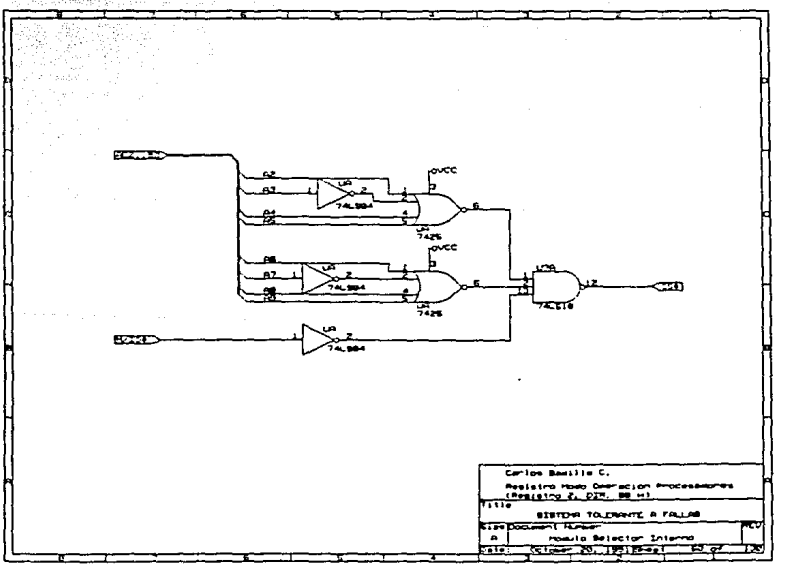
LAMINA 47. FUENTE DE DATOS



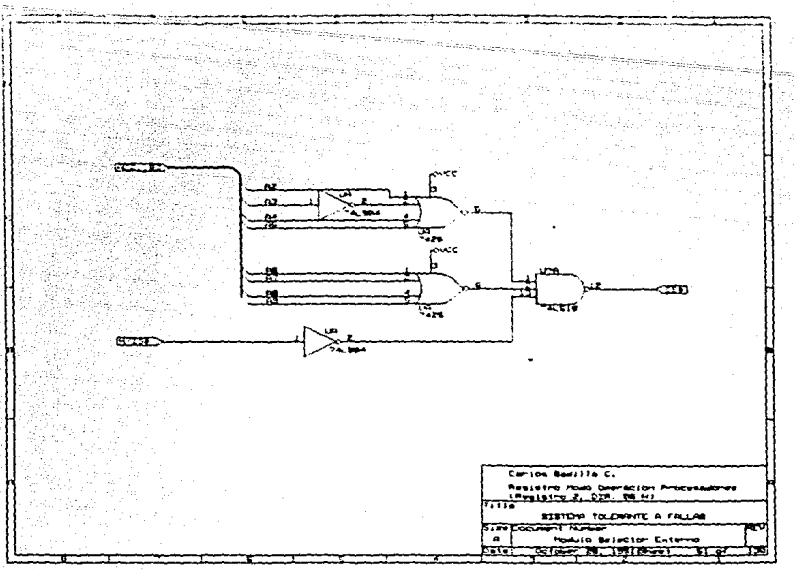
LAMINA 48. REGISTRO DEL MODO DE OPERACION DE PROC.



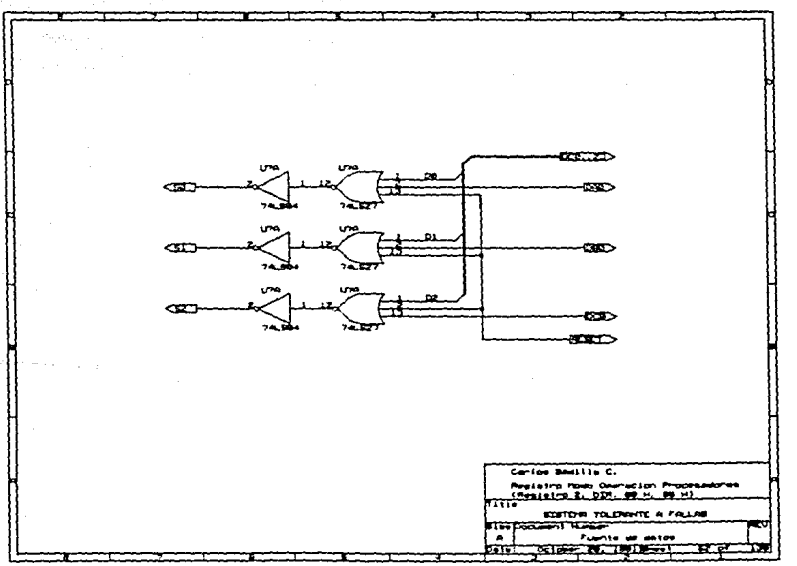
LAMINA 49. SELECTOR INTERNO



LAMINA 50. MODULO SELECTOR INTERNO



LAMINA 51. MODULO SELECTOR EXTERNO



LAMINA 52. FUENTE DE DATOS

6.4. REGISTROS DE CONFIGURACION DEL CIRCUITO DE VOTACION

Referencia: lámina 37.

La operación de los circuitos de votación se define por medio de dos parámetros (estado y categoría). Esta información se almacena en los registros de configuración de los circuitos votadores.

Un circuito de votación puede encontrarse en uno de dos posibles estados de operación: activo o inactivo#. El estado en que se encuentra cada votador se indica por medio del registro de estado de circuitos de votación.

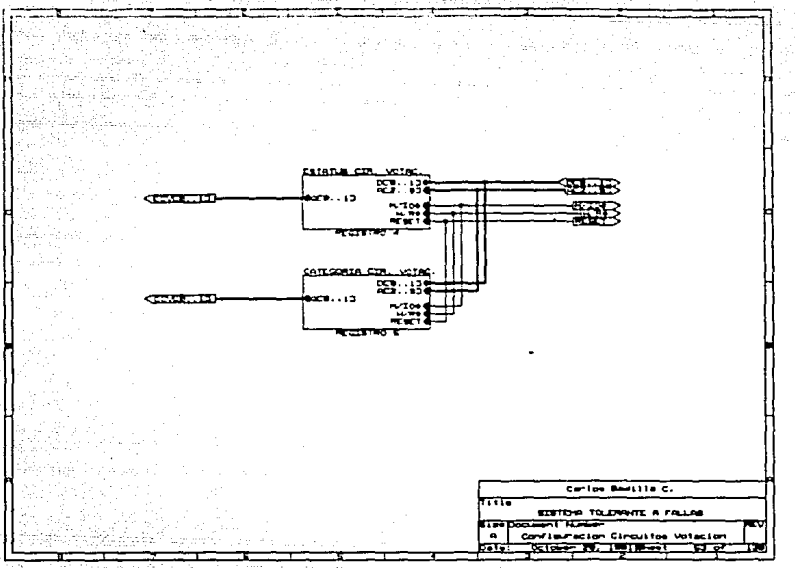
Además, cada circuito de votación activo puede operar en una de dos posibles categorías: principal o respaldo#. La categoría en que se encuentre cada votador se indica por medio del registro de categoría de circuitos de votación.

Si un circuito votador se encuentra en estado activo puede comparar salidas de procesadores y generar solicitudes de interrupción en casos de detectar error en uno de los procesadores. Si además se encuentra en categoría principal, en caso de detectar falla en un procesador, sus salidas F[0..2] se usan para modificar el contenido del registro de estado de procesadores.

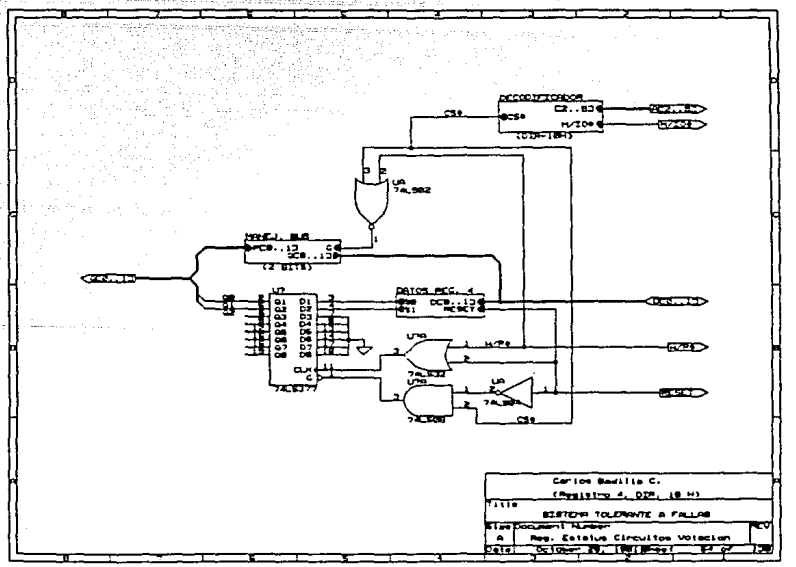
En el sistema solo puede haber un circuito de votación con categoría principal. Si un circuito de votación está en modo inactivo#, la calificación de principal o respaldo# es irrelevante.

Si un circuito de votación opera en estado activo y categoría de respaldo# se cumple lo siguiente:

- recibe los mismos datos entrantes que el circuito de votación principal;
- compara las salidas de los procesadores maestros y efectúa el proceso de votación;
- genera una solicitud de interrupción en caso de detectar un error;



LAMINA 53. CONFIGURACION DE CIRCUITOS DE VOTACION



LAMINA 54. REGISTRO DE ESTATUS DE CIRCUITOS DE VOTACION

- sus salidas F[0..2] NO pueden modificar el contenido del registro de estado de procesadores.

Los registros de configuración de circuitos de votación se caracterizan porque solo pueden operar como "puertos externos", es decir, son accesibles solo al procesador con categoría principal (tanto para operaciones de lectura como de escritura).

Las salidas de los registros de configuración de circuitos de votación son de tipo "latch". Esto permite tener disponible, en forma permanente, sus estados lógicos para controlar la operación de los circuitos de votación.

6.4.1. REGISTRO DE ESTADO DE CIRCUITOS DE VOTACION

Registro número 4. Ocupa la localidad 10 H del espacio de puertos del sistema (láminas 54..56).

Tiene 2 bits de ancho, cada uno de ellos asociado a uno de los circuitos de votación. Se utiliza para habilitar (o deshabilitar) los circuitos de votación.

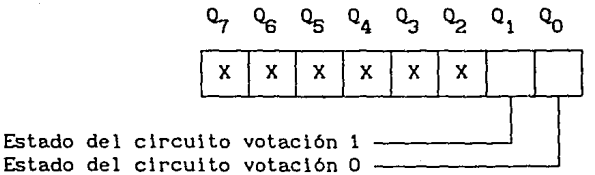
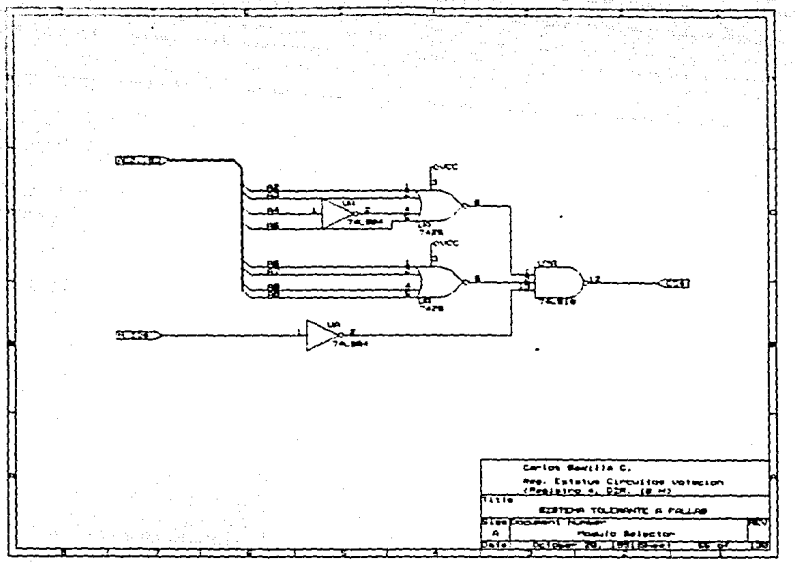


Figura 6.4 Registro de estado de circuitos de votación

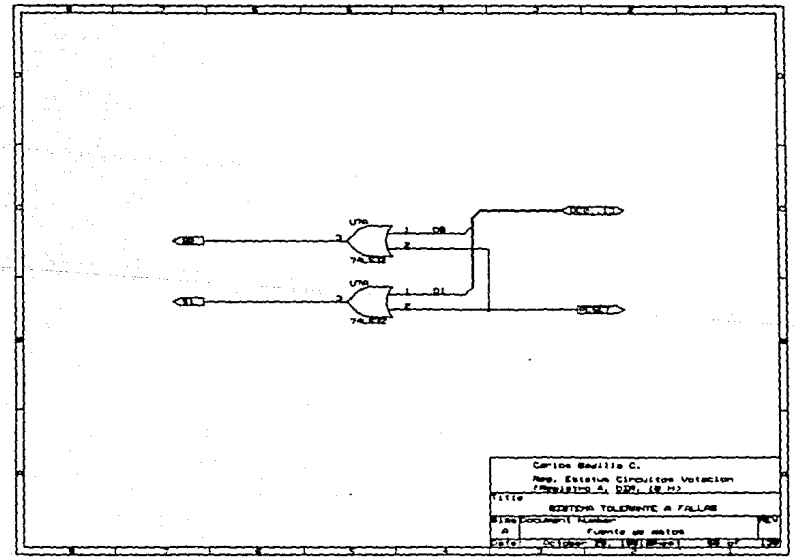
El contenido de este registro puede ser establecido de dos maneras diferentes (lámina 56):

a. Por medio del circuito de "reset"

Durante la rutina de iniciación ambos circuitos de votación son declarados activos: se escribe un "uno" en sus bits de estado.



LAMINA 55. MODULO SELECTOR



LAMINA 56. FUENTE DE DATOS

b. Por medio del procesador principal

El procesador que opere en categoría principal puede modificar el estado de operación de los circuitos de votación escribiendo sobre el puerto asociado a dicho registro.

6.4.2. REGISTRO DE CATEGORIA DE CIRCUITOS DE VOTACION

Registro número 5. Ocupa la localidad 14 H del espacio de puertos del sistema (láminas 57..58).

Tiene 2 bits de ancho, cada uno de ellos asociado a uno de los circuitos de votación. Se usa para definir la categoría (principal o respaldo#) en que opera cada uno de los circuitos de votación.

El votador principal se caracteriza porque sus salidas pueden modificar el contenido de uno de los registros de configuración del sistema (registro de estado de los procesadores) y con ello indirectamente, controlar transceptores que permiten el acceso de los procesadores a los los buses del sistema.

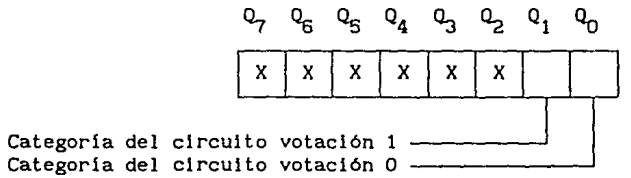
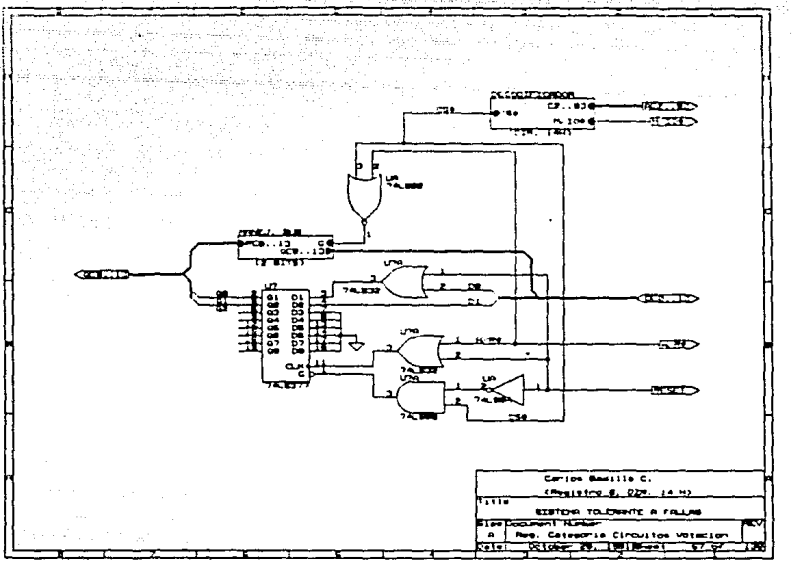


Figura 6.5 Registro de categoría de circuitos de votación

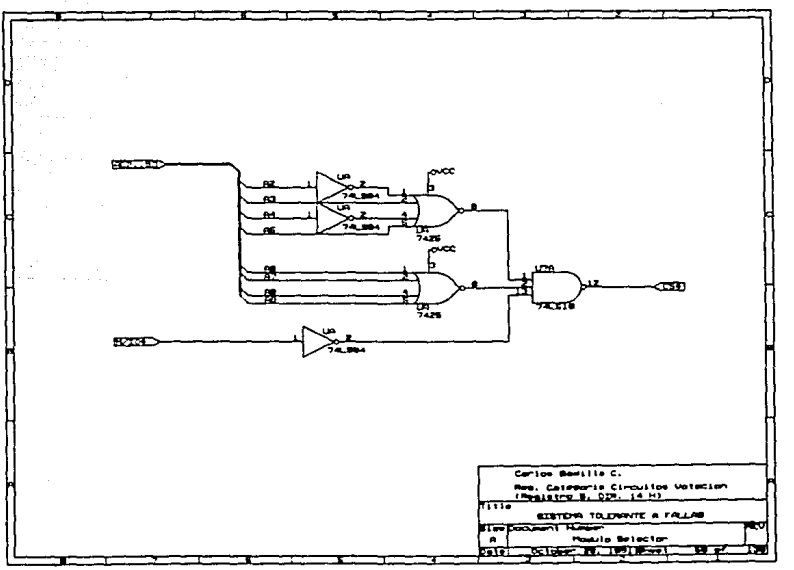
El contenido de este registro puede ser establecido de dos maneras diferentes (lámina 57):

a. Por medio del circuito de "reset"

Durante la rutina de iniciación se declara principal al circuito de votación 0 (cero) y como respaldo# al número 1: se escribe un "uno" en el bit 0 y "cero" en el bit 1.



LAMINA 57. REGISTRO DE CATEGORIA DE CIRCUITOS DE VOTACION



LAMINA 58. MODULO SELECTOR

b. Por medio del procesador principal

El procesador que opere en categoría principal puede modificar la categoría de operación de los circuitos de votación escribiendo sobre el puerto asociado a dicho registro.

6.5. REGISTROS DE SALIDAS DE COMPARADORES DE CIRCUITOS DE VOTACION

Referencia: lámina 59.

Los registros de salidas de comparadores de circuitos de votación se utilizan para almacenar los valores de las salidas de cada uno de los comparadores internos del circuito de votación en el momento de detectarse una falla (en uno de los procesadores o en uno de los comparadores internos del circuito de votación). La interpretación del significado asociado al dato almacenado en ellos se indica en la Tabla 5.1.

Estos registros se caracterizan porque solo operan como "puertos externos", es decir, son accesibles solo al procesador con categoría principal (tanto para operaciones de lectura como de escritura).

Las salidas de estos registros son de tipo tercer estado. Esto es debido a que sus estados lógicos no son utilizados para controlar en forma directa la operación de partes del sistema sino que son empleados como entradas por el procesador para ejecutar rutinas de diagnóstico en caso de interrupción originada por el circuito de votación.

Estos registros tienen 3 bits de ancho, cada uno de ellos asociado a uno de los comparadores del circuito de votación.

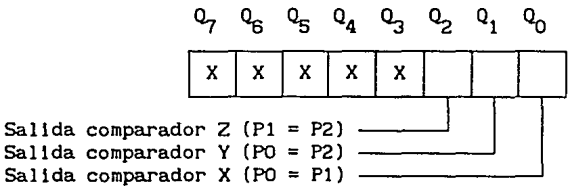
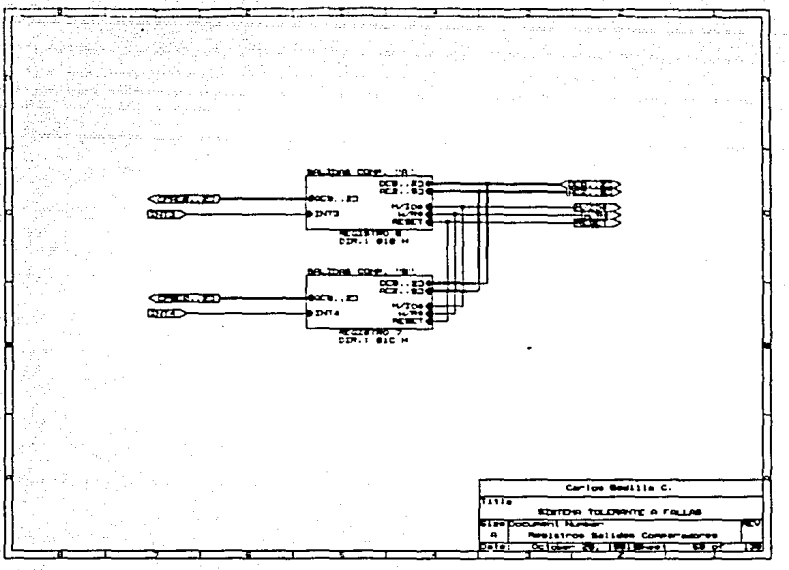
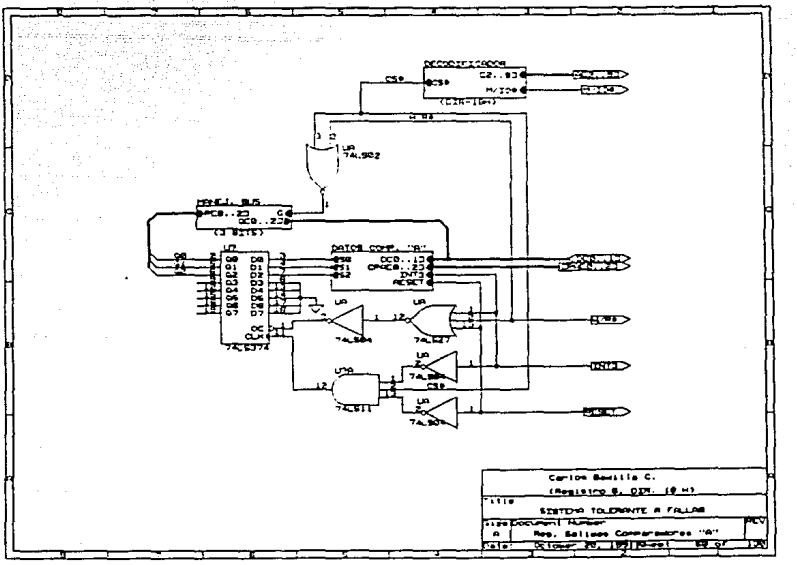


Figura 6.6 Registro de salidas comparadores "A" (Circuito Votación "0")



LAMINA 59. REGISTROS DE LAS SALIDAS DE LOS COMPARADORES



LAMINA 60. REGISTRO DE LAS SALIDAS DE LOS COMPARADORES"A"

El contenido de estos registros puede ser establecido de tres maneras diferentes (lámina 62 y 65):

a. Por medio del circuito de "reset"

Durante la rutina de iniciación se almacena el código correspondiente a sistema sin fallas: se escribe un "uno" en sus bits del registro.

b. Por medio del circuito de votación.

El contenido de estos registros se actualiza por medio la línea de la solicitud de interrupción cada vez que el circuito votador con el que está asociado encuentra una anomalía (en uno de los procesadores o en uno de los comparadores del circuito de votación).

Los valores presentes en las salidas de los comparadores internos del circuito de votación se almacenan en estos registros en el momento de detectarse un error.

c. Por medio del procesador principal

El procesador que opere en categoría principal puede modificar el contenido de los registros de salidas de comparadores escribiendo sobre el puerto asociado a cada uno de ellos.

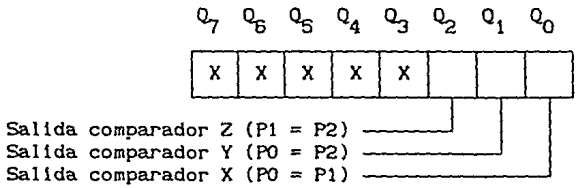
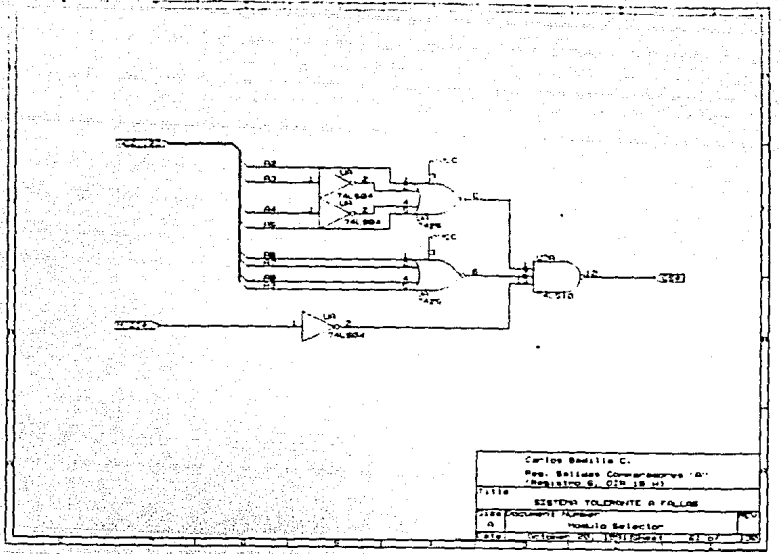


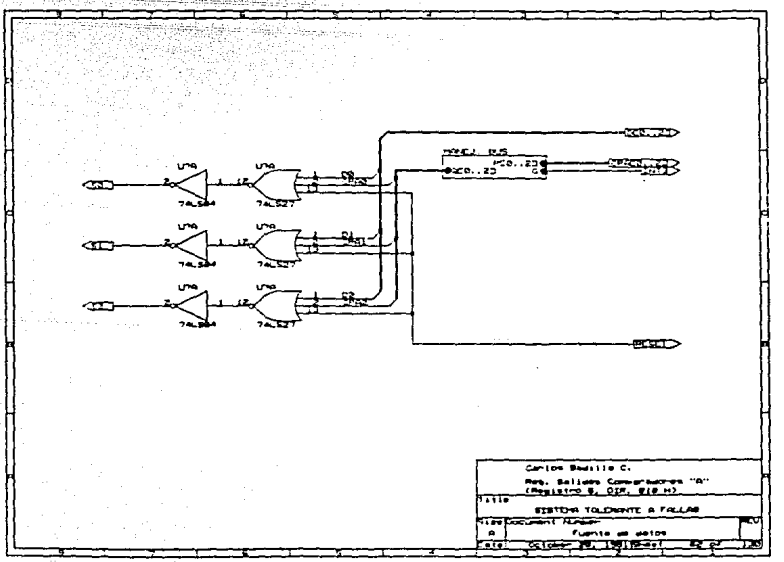
Figura 6.7 Registro de salidas comparadores "B"

A cada uno de los circuitos de votación está asociado un registro de salidas de comparadores.

Esta provisión permite a cada circuito de votación almacenar, en forma independiente, las condiciones de falla que detecte.



LAMINA 61. MODULO SELECTOR



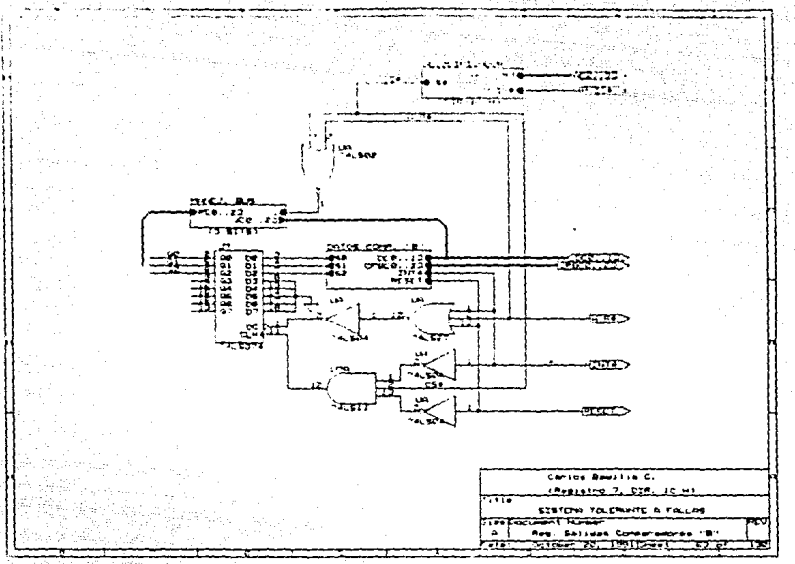
LAMINA 62. FUENTE DE DATOS

6.5.1. REGISTRO DE SALIDAS DE COMPARADORES "A"

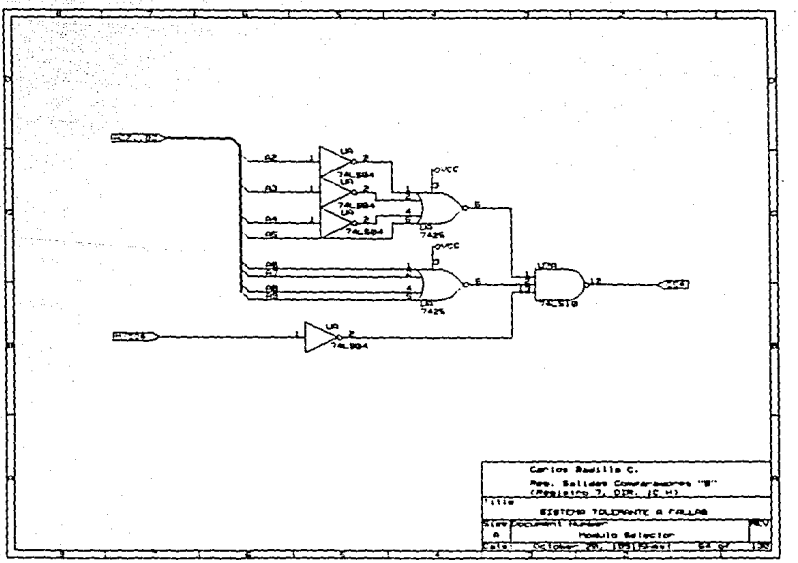
Registro número 6. Está asociado al circuito de votación 0 (cero) Ocupa la localidad 18 H del espacio de puertos del sistema (láminas 60..62).

6.5.2. REGISTRO DE SALIDAS DE COMPARADORES "B"

Registro número 7. Está asociado al circuito de votación 1 (uno) Ocupa la localidad 1C H del espacio de puertos del sistema (láminas 63..65).



LAMINA 63. REGISTRO DE SALIDAS DE COMPARADORES "B"



LAMINA 64. MODULO SELECTOR

CAPITULO 7

TRANSCEPTOR GENERAL DEL SISTEMA

Y

REGISTRO DE SOLICITUD DE INTERRUPCION DE PROCESADORES

INTRODUCCION

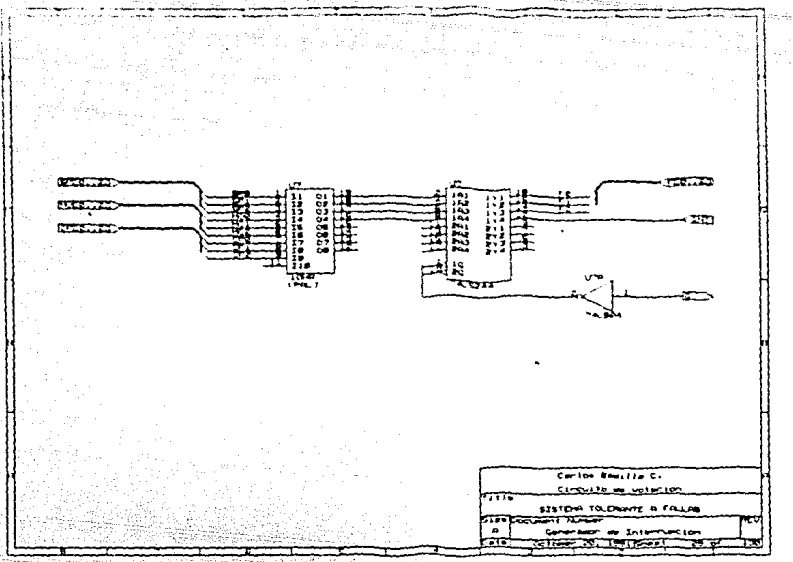
Este capítulo contiene la descripción de los módulos "transceptor general del sistema" y "registro de solicitud de interrupción de procesadores" que forman parte del subsistema de procesamiento. El primero de estos módulos se utiliza para gobernar el acceso de los procesadores al bus del sistema; el otro permite a un procesador solicitar la atención de los restantes.

7.1. TRANSCEPTOR GENERAL DEL SISTEMA

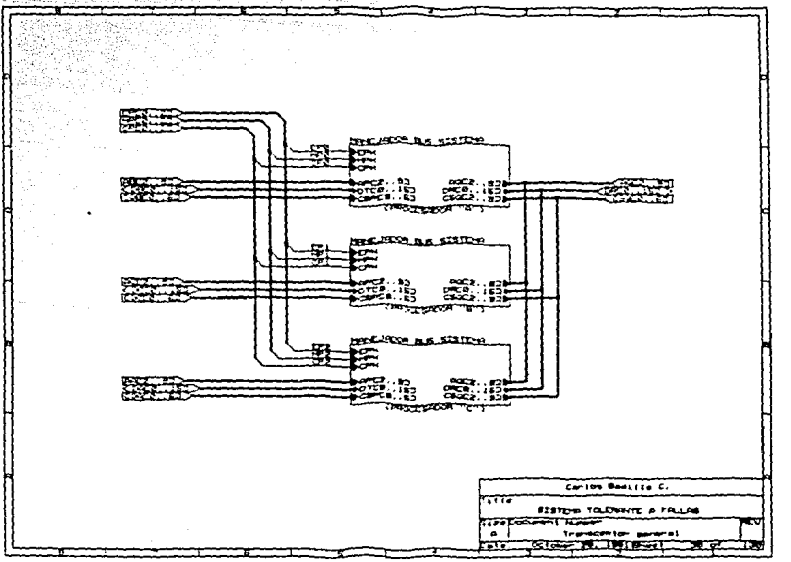
El transceptor general es la sección de la unidad de procesamiento cuya función es comunicar (o aislar) los buses de los procesadores (datos, direcciones, control) con el bus del sistema. Está constituido por un grupo de manejadores de bus y sus correspondientes circuitos de control.

Los transceptores asociados a cada procesador son gobernados por un circuito específico para dicho módulo. Esto permite aislar a los procesadores por separado con el fin de permitir la operación continua aún en el caso de que alguno de los procesadores se encuentre dañado.

El circuito lógico utilizado para administrar la operación de los transceptores de cada procesador es puramente combinacional. Las entradas recibidas por cada circuito así como sus salidas son nominalmente las mismas. Se diferencian solamente en cuanto al procesador que las origina. Por consiguiente se incluye la descripción del circuito prototipo utilizado para realizar la función indicada.



LAMINA 29. GENERADOR DE INTERRUPCION



LAMINA 30. TRANCEPTOR GENERAL

7.2. LINEAS DE ENTRADA AL CIRCUITO DE CONTROL DE TRANSCEPTORES

Para generar las funciones de control de los transceptores (habilitación [G] y sentido [T/R#]), el circuito de control de cada manejador de bus utiliza la información contenida en los registros de configuración de procesadores (estado, modo de operación y categoría) y dos líneas de estado del bus (M/IO#, W/R#).

7.2.1. SEÑALES DE ESTADO DEL BUS

Las líneas de estado del bus usadas como entradas por los circuitos de control de transceptores son las que contienen las señales generadas por el procesador 80386 (que opera con con categoría principal) para gobernar los procesos de entrada/salida de datos.

a) M/IO# : Línea utilizada para diferenciar los accesos a memoria (M) de los accesos a dispositivos periféricos (IO#).

b)WR/RD# : Línea empleada para identificar el tipo de proceso en desarrollo: escritura (WR) o lectura (RD#).

7.2.2. LINEAS DE CONFIGURACION DE PROCESADORES

En este grupo se incluyen las líneas de salida de los registros de configuración asociadas a un procesador particular que son usadas como líneas de entrada al circuito de control de transceptores de dicho procesador.

a) Estado de procesador (activo - inactivo#)
b) Modo de operación de procesador (maestro - esclavo#)
c) Categoría de procesador (principal - respaldo#)

7.3. LINEAS DE SALIDA DEL CIRCUITO DE CONTROL DE TRANSCEPTORES

Las líneas de salida de cada circuito de control son utilizadas para controlar únicamente los transceptores asociados a un procesador en particular.

a) Habilitación de transceptores (G): Linea utilizada para activar los transceptores.

b) Sentido de conducción (T/R#): Linea utilizada para definir el sentido u orientación en que viajan los datos en un transceptor bidireccional habilitado.

7.4. FUNCIONAMIENTO DEL CIRCUITO COMBINACIONAL DE CONTROL DE TRANSCEPTORES

Análisis de la tabla de verdad del circuito combinacional de control de transceptores.

a) MINTERMINOS 0 A 15

Si a un procesador se le ha asignado estado inactivo#, todos los transceptores que unen a dicho procesador con el bus del sistema quedan deshabilitados y, con ello, el procesador aislado.

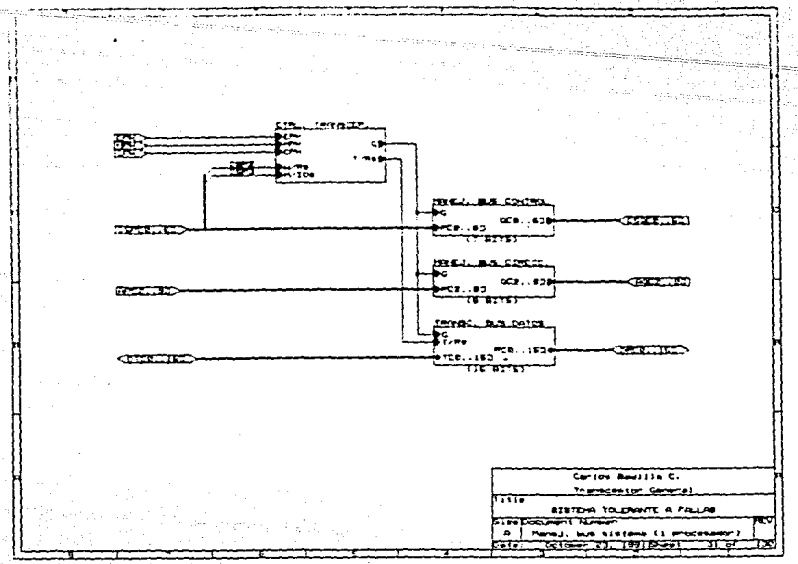
No obstante, en este caso, el procesador puede solicitar reingreso al sistema escribiendo un "uno" sobre el bit asociado en el registro de solicitud de interrupción de procesadores. También puede recuperar el estado activo escribiendo un "uno" sobre el puerto interno asociado al registro de estado de procesadores.

b) MINTERMINOS 16 A 23

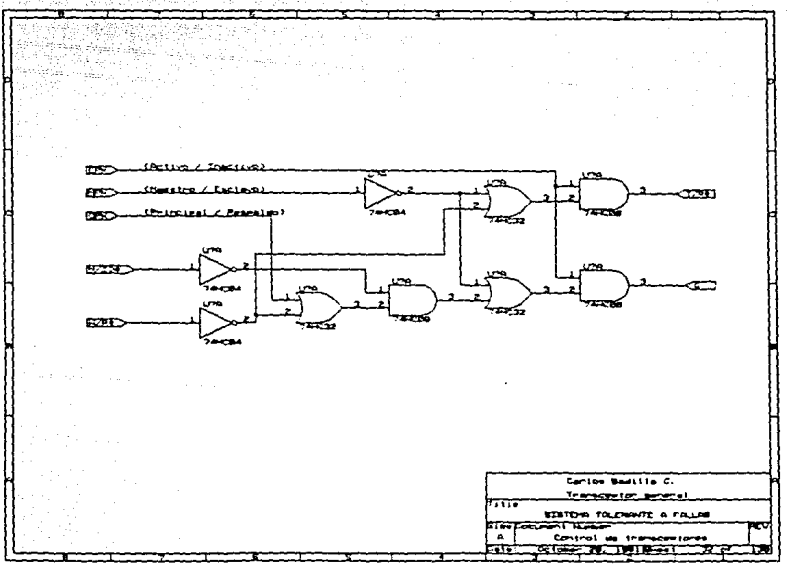
Esta parte de la tabla define el valor previsto para las funciones de salida del transceptor (G, T/R#) cuando el procesador asociado opera en modo esclavo y recibe información del procesador que se encuentra en modo maestro.

Conviene enfatizar que este es solo un comportamiento previsto ya que esta versión del computador contempla el modo de operación esclavo únicamente para ejecución de tareas de autodiagnóstico, pero no para la comunicación interprocesador.

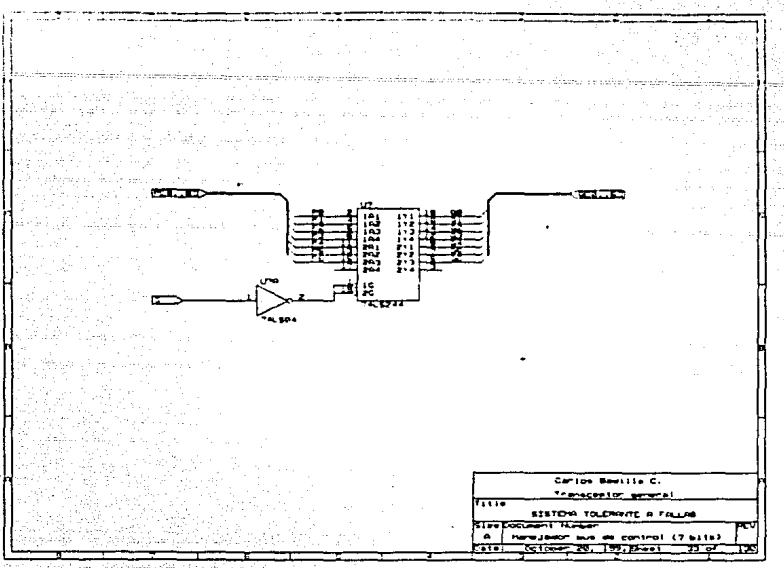
Bajo esta consideración la operación del transceptor en esta parte de la tabla es la siguiente:



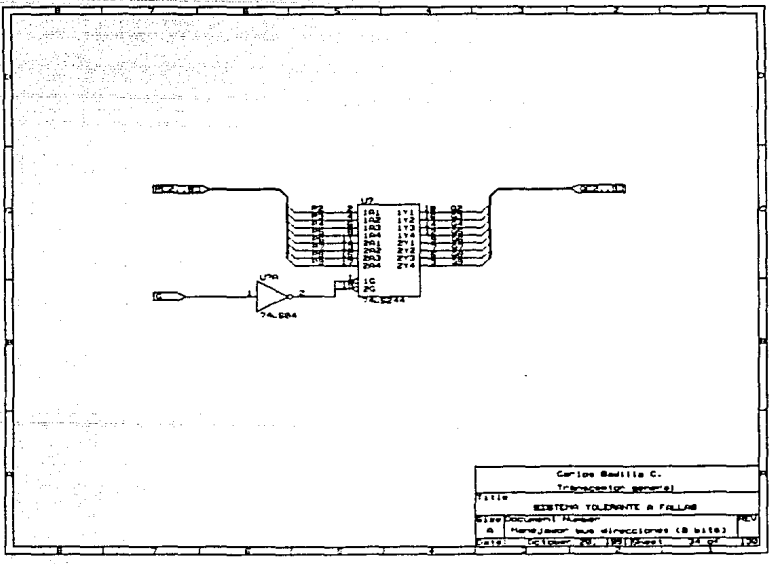
LAMINA 31. MANEJADOR DE BUS DEL SISTEMA



LAMINA 32. CONTROL DE TRANSCEPTORES



LAMINA 33. MANEJADOR DE BUS DE CONTROL



LAMINA 34. MANEJADOR DE BUS DE DIRECCIONES

	ENTRADAS					SALIDAS	
MINT	ESTADO PROCES.	MODO OPERAC.	ESPACIO DIRECC.	ACCION I/O	CATEGOR. PROCES.	HABILIT. TRANSC.	SENTIDO CONDUCC.
	ACT/INA#	MST/ESC#	M/IO#	WR/RD#	PRI/RES#	HAB/DHA#	ENT/SAL#
0	0	0	0	0	0	0	x
1	0	0	0	0	1	0	x
2	0	0	0	1	0	0	x
3	0	0	0	1	1	0	x
4	0	0	1	0	0	0	x
5	0	0	1	0	1	0	x
6	0	0	1	1	0	0	x
7	0	0	1	1	1	0	x
8	0	1	0	0	0	0	x
9	0	1	0	0	1	0	x
10	0	1	0	1	0	0	x
11	0	1	0	1	1	0	x
12	0	1	1	0	0	0	x
13	0	1	1	0	1	0	x
14	0	1	1	1	0	0	x
15	0	1	1	1	1	0	x

Tabla 7.1. Tabla de verdad del control de transceptores (I. parte)

	ENTRADAS					SALIDAS	
MINT	ESTATUS PROCES.	MODO OPERAC.	ESPACIO DIRECC.	ACCION I/O	CATEGOR. PROCES.	HABILIT. TRANSC.	SENTIDO CONDUCC.
	ACT/INA#	MST/ESC#	M/IO#	WR/RD#	PRI/RES#	HAB/DHA#	ENT/SAL#
16	1	0	0	0	0	1	1
17	1	0	0	0	1	1	1
18	1	0	0	1	0	1	1
19	1	0	0	1	1	1	1
20	1	0	1	0	0	1	1
21	1	0	1	0	1	1	1
22	1	0	1	1	0	1	1
23	1	0	1	1	1	1	1
24	1	1	0	0	0	1	1
25	1	1	0	0	1	1	1
26	1	1	0	1	0	0	x
27	1	1	0	1	1	1	0
28	1	1	1	0	0	0	x
29	1	1	1	0	1	0	x
30	1	1	1	1	0	0	x
31	1	1	1	1	1	0	x

Tabla 7.1. Tabla de verdad del control de transceptores (II. parte)

Cuando un procesador trabaja en modo de operación esclavo#, el procesador en categoría principal puede transferir datos hacia el procesador esclavo# bajo control del sistema operativo. Para que esta operación pueda llevarse a cabo, los transceptores del procesador esclavo# (datos, direcciones, control) se habilitan en sentido entrante, para recibir la información procedente del maestro, durante el lapso que dure la transferencia de datos.

La ejecución de operaciones de transferencia de información debe realizarse bajo control del sistema operativo y permite realizar tareas tales como recargar una de las áreas de memoria (datos o programa) del procesador esclavo#.

En modo de operación esclavo#, la calificación de categoría (principal - respaldo#) es irrelevante, por lo que esta variable es indiferente para definir la habilitación de sus transceptores y el sentido en que conducen.

c) MINTERMINOS 24 Y 25

Cuando se realiza una lectura de puerto, los procesadores en modo de operación maestro pueden aceptar un dato de entrada independientemente de si tienen categoría principal o de respaldo#.

d) MINTERMINOS 26 Y 27

Si un procesador tiene categoría de respaldo# y se realiza una operación de escritura, sus transceptores deben permanecer deshabilitados y la especificación de sentido de transferencia es irrelevante (mintérmino 26).

Por el contrario, si el procesador opera en categoría principal sus transceptores deben estar habilitados como salidas (mintérmino 27).

e) MINTERMINOS 28 A 31

Durante las operaciones de movimiento de datos al interior de un módulo de procesamiento, en modo de operación maestro o esclavo#, los transceptores permanecen deshabilitados. En este caso, el sentido de la transferencia es irrelevante.

7.5. RELACION ENTRE EL CIRCUITO DE VOTACION Y EL CONTROL DE TRANSCEPTORES

El circuito de votación no puede deshabilitar, en forma directa, los transceptores que enlazan a los procesadores con el bus del sistema. Esto es debido a que, en caso de producirse una condición de deshabilitación general, los procesadores quedarían aislados y no habría posibilidad de reactivar al sistema ya que ninguno de los procesadores tendría acceso al archivo de registros de configuración.

En vez de ello, el circuito de votación, puede deshabilitar indirectamente los procesadores, en caso de identificar error en alguno de ellos. Para ello debe modificar el contenido del registro de estado de procesadores, mediante las líneas de salida F[0..2] y la de solicitud de interrupción INT.

Las líneas F[0..2] del circuito de votación con categoría principal se utilizan para apagar los bits del registro de estado de procesadores. Esta acción ocasiona que el (los) procesador(es) fallado(s) queden aislados del bus del sistema.

En el caso anterior, los procesadores pueden reincorporarse al sistema mediante la ejecución de una rutina de reincorporación que permita realizar los ajustes de reconfiguración que sean necesarios en el sistema.

7.6. REGISTRO DE SOLICITUD DE INTERRUPCION DE PROCESADORES

Este registro se utiliza para solicitar la atención de un procesador por parte de otro durante alguna etapa de ejecucción.

Un uso típico de este registro se presenta durante la etapa de reincorporación de un procesador al sistema. Cuando el circuito de votación descubre un error de procesamiento, el procesador "fallado" es puesto temporalmente fuera de operación para ejecutar una rutina de diagnóstico. Si como resultado de esta rutina se concluye que el procesador involucrado opera libre de fallas, dicho procesador puede escribir un "uno" en el bit asociado al registro de solicitud de interrupción para "levantar la bandera de solicitud de reincorporación".

Este registro tiene el número 23 y ocupa la localidad 8C H del espacio de puertos del sistema (láminas 66..69).

Tiene 3 bits de ancho, cada uno de los cuales está asociado a uno de los procesadores.

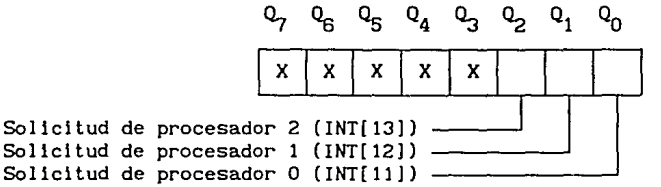
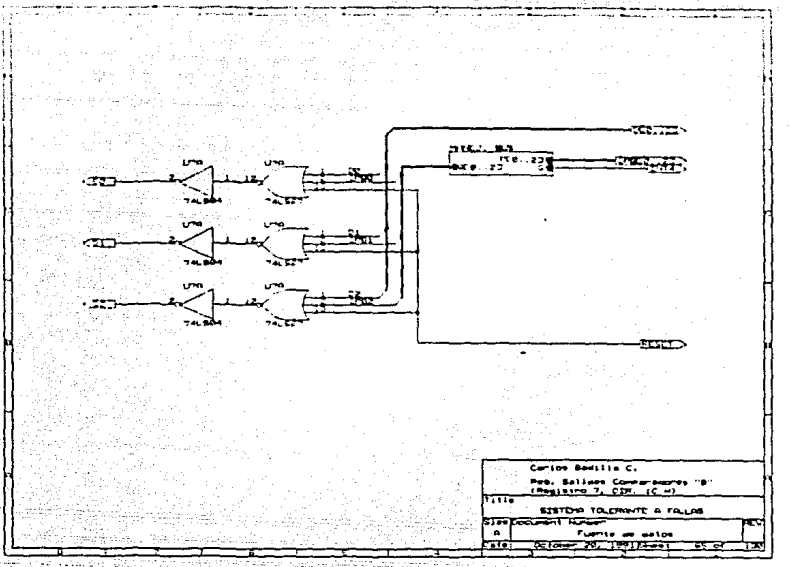
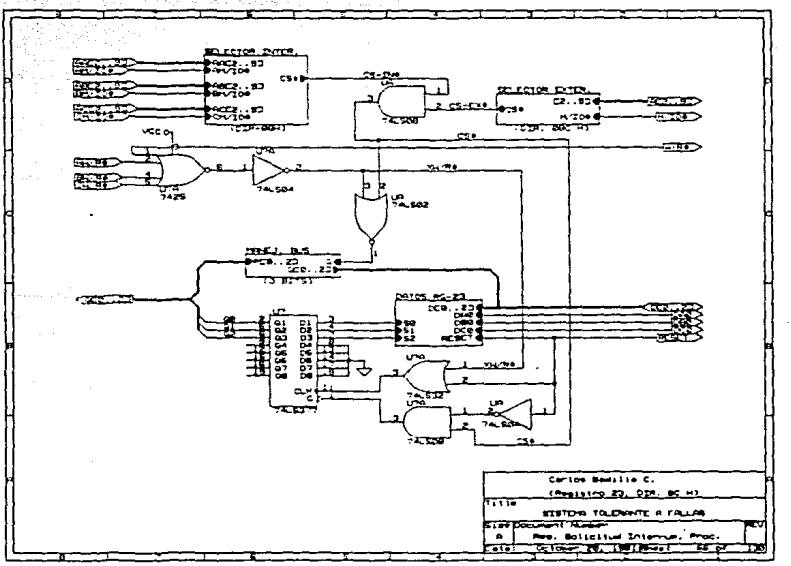


Figura 7.1 Registro de solicitud de interrupción de procesadores

Las salidas de este registro corresponden a las líneas INT[11..13] del bus de fuentes de solicitud de interrupción y están conectadas a las entradas de la unidad de control de interrupciones. Esto permite contar con un método de identificación inequívoca del procesador que solicita la atención del sistema.



LAMINA 65. FUENTE DE DATOS



LAMINA 66. REGISTRO DE SOLICITUD DE INTERRUPCION DE PROCESADORES

Este registro tiene salidas de tipo "latch", lo cual permite tener disponibles sus estados lógicos durante el tiempo requerido por el controlador de interrupciones para procesar la solicitud.

El registro de solicitud de interrupción de procesadores puede operar como doble puerto ("externo" e "interno") mapeado sobre la misma dirección del espacio de puertos (08C H)

Como "puerto externo" es accesible al procesador con categoría principal tanto para operaciones de lectura como de escritura. Este tipo de operación resulta útil, por ejemplo, durante la etapa de depuración del sistema.

Como "puerto interno" es accesible para operaciones de escritura a todos los procesadores independientemente de su categoría con la restricción de que pueden modificar únicamente el bit que tengan asociado en el puerto accesado.

El contenido de este registro puede establecerse de tres maneras diferentes (lámina 67):

a. Por medio del circuito de RESET

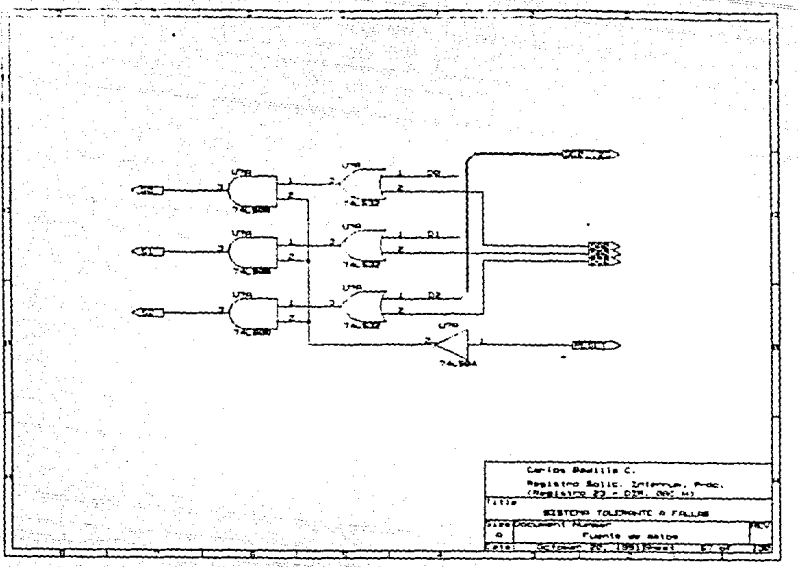
Durante la rutina de iniciación ningún procesador solicita interrupción: se escribe un "cero" en todos los bits de este registro.

b. Por medio del propio procesador

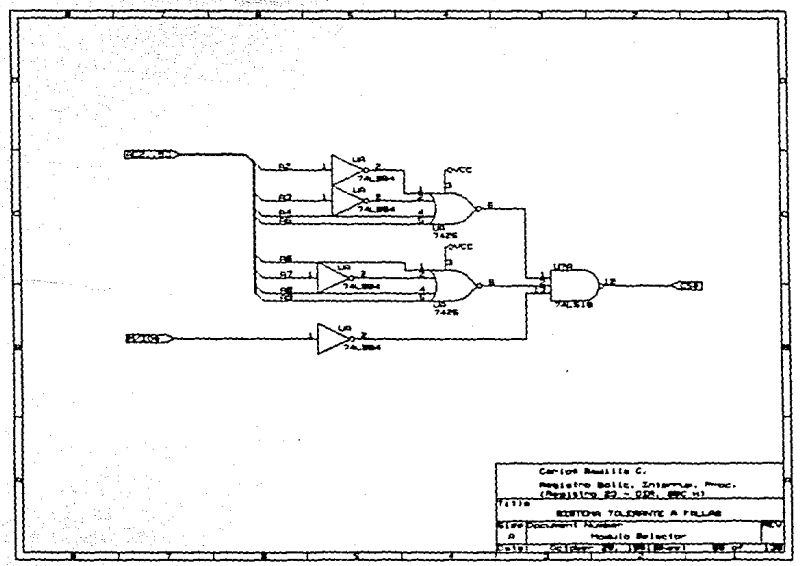
Cada procesador puede solicitar servicio de interrupción escribiendo en el puerto interno asociado a este registro.

c. Por medio del procesador principal

El procesador que opere en categoría principal puede modificar el contenido del registro de solitud de interrupción de procesadores escribiendo sobre el puerto externo que tiene asociado dicho registro.



LAMINA 67. FUENTE DE DATOS



LAMINA 68. MODULO SELECTOR

CAPITULO 8
CANAL DE COMUNICACIONES SERIE

INTRODUCCION

El puerto en serie permite al computador recibir programas, datos y comandos de control (por medio del equipo de comunicaciones) procedentes de una estación de control ubicada en tierra. También le permite enviar información (estado, variables físicas capturadas, resultados de cálculo, etc.) telemétrica a tierra.

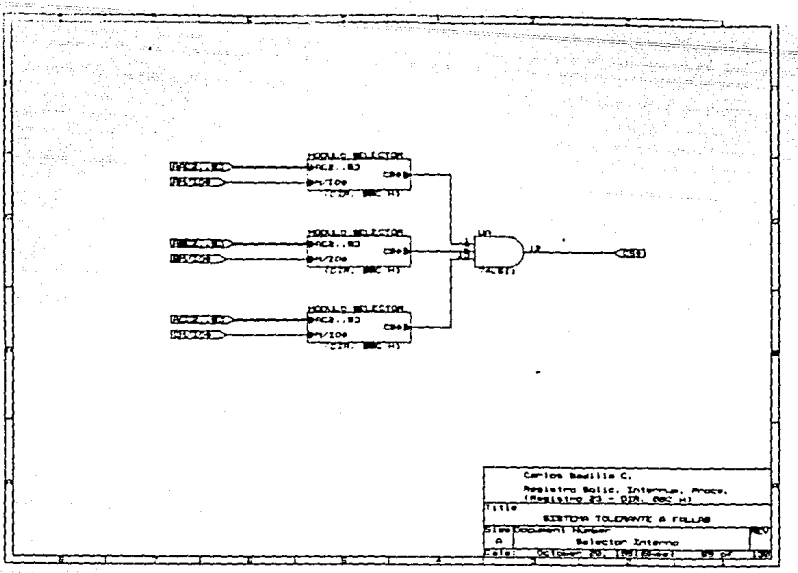
El canal de comunicaciones presentado en este diseño tiene elementos que le permiten realizar ciertas funciones de tolerancia a fallas. En particular, cuenta con dos dispositivos de transmisión / recepción asíncrona (UART NS16550A) y un bloque de comparadores que vigila la igualdad de salidas del módulo de comunicaciones. Esto ocurre cada vez que el procesador efectúa la lectura de datos recibidos o de estado de los UARTs.

Cuando el manejador de interrupciones informa al procesador sobre una anomalía en las salidas del canal serie, el procesador entra en una rutina de servicio de interrupción que permita diagnosticar la falla u origen del error detectado y, eventualmente, proceder a aislar al periférico fallado.

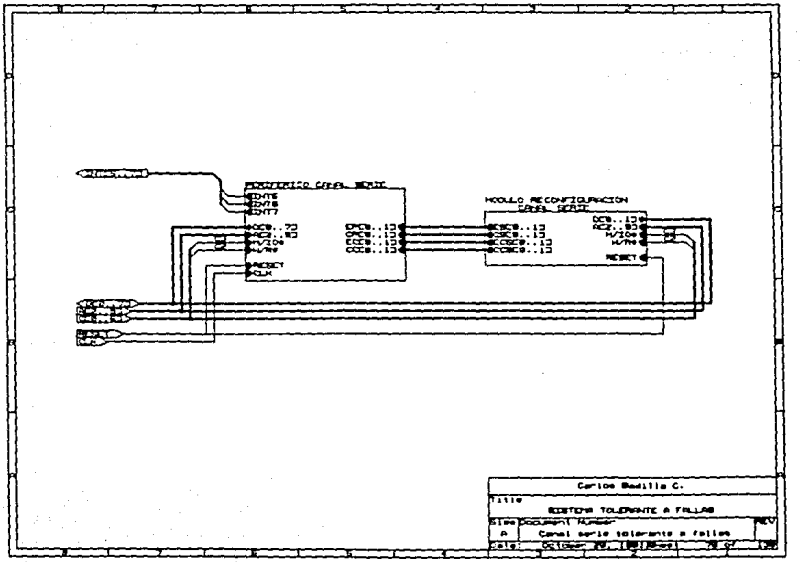
Las rutinas de diagnóstico de fallas pueden realizarse en forma local o remota de acuerdo con las opciones que sean programadas en el sistema operativo.

8.1. DISEÑO DEL CANAL DE COMUNICACIONES SERIE CON UARTs

Al desarrollar aplicaciones que utilicen UARTs deben tenerse en consideración las siguientes criterios: [Mic 88a]



LAMINA 69. SELECTOR INTERNO



LAMINA 70. CANAL DE SERIE TOLERANTE A FALLAS

El CPU es usualmente mucho más rápido que el UART al transferir datos. Un procesador de alta velocidad puede transferir un byte hacia o desde una UART en un tiempo mínimo de 280 nsec. El UART podría tomar un tiempo de más de 1800 veces mayor para transmitir o recibir serialmente si fuera operado a 19.2 kbauds.

El CPU requiere un tiempo mínimo fijo para dar servicio a la UART debido al cambio de contexto requerido para realizar las transferencias

Tomando en cuenta estas consideraciones puede concluirse que el tiempo que el CPU tiene disponible para ejecutar otras funciones aumenta conforme se incrementa el desempeño de los dispositivos empleados para el manejo de las comunicaciones.

Estas ventajas han sido las que se han tomado en cuenta para escoger la UART NS16550A para el manejo del canal serie en este sistema.

8.2. TRANSMISOR RECEPTOR UNIVERSAL ASINCRONO NS16550A

El NS16550A (UART) es un convertidor serie - paralelo que puede utilizarse como puerto de entrada - salida en un sistema basado en microprocesador.

El UART realiza conversiones serie-a-paralelo sobre los caracteres de datos recibidos de dispositivos periféricos o de un modem y conversiones paralelo-a-serie sobre los caracteres recibidos del CPU.

El procesador puede leer el estado completo del UART en cualquier momento de operación. La información reportada incluye el tipo y condición de las operaciones de transferencia que están siendo realizadas por el UART, así como cualquier condición de error (paridad, sobreflujo, marco o suspensión de la transferencia).

El UART tiene un generador programable de baudaje capaz de dividir la entrada del reloj de referencia temporal (CLK) por divisores de 1 a (2**16 -1) y producir una señal de 16 veces la frecuencia del reloj utilizado para manejar la lógica del transmisor interno.

El NS16550A presenta la ventaja, sobre sus antecesores más conocidos (NS8250, NS16450), de que cuenta con dos memorias de tipo FIFO con una profundidad de 16 bytes cada una (la primera contiene datos para el transmisor y otra para el receptor) y la correspondiente circuiteria para su manejo. Esta característica permite reducir la programación y tiempo de procesador requeridos para realizar las transferencias de datos debido a que reduce el número de interrupciones presentadas al CPU.

Desde la perspectiva del procesador, el NS16550A es visto como un dispositivo de 8 localidades, es decir, requiere tres líneas de direccionamiento A0..A2.

Este componente tiene dos registros utilizados para realizar la transmisión de datos y otros dos para realizar la recepción, lo cual elimina la necesidad de sincronización precisa entre el procesador y el equipo de comunicaciones.

En el circuito de entrada, el registro de desplazamiento RX se utiliza para recibir el dato de entrada y el registro RX-hold para almacenar el dato anteriormente recibido.

En el circuito de salida, el registro de desplazamiento TX se utiliza para transmitir el dato saliente y el registro TX-hold para almacenar el próximo dato a transmitir.

El UART NS16550A tiene un conjunto de líneas que permite manejar todas las funciones de modem (CTS, RTS, DSR, DTR, RI, DCD).

Las características de la interfase serial son totalmente programables:

- Caracteres de 5,6,7 u 8 bits
- Generación y detección de paridad par, impar o no paridad
- Generación de bits de fin (stop) (1, 1.5 o 2)
- Generación de baudaje (DC a 256 k baud)

Además, puede detectar la ocurrencia de falso bit de inicio.

El mecanismo de manejo de interrupciones permite generar solicitudes en forma totalmente priorizada.

Cuenta con dos salidas OUT1#, OUT2# las cuales se comportan como puerto de salida serie de propósito múltiple. Una aplicación típica de estas líneas consiste en utilizar una de ellas para habilitar un buffer tri-state que conecte la salida INT con una de las entradas de la unidad de control de interrupciones (8259A).

El NS16550A tiene todos los registros de programación y control de sus antecesores (NS8250 y NS16450), lo cual proporciona compatibilidad total con la programación escrita para cualquiera de ellos. Esta característica es crítica debido a la gran cantidad de programación que ha sido escrita para manejo de UARTs.

Además de las características anteriores, el UART NS16550A tiene una interfase incorporada para la operación con DMA (acceso directo a memoria) y una temporización de bus más rápida ,lo cual ofrece mayores ventajas para ser utilizado con procesadores rápidos tales como el 80386.

8.3. OPERACIONES DE TRANSFERENCIA DE DATOS EN EL NS16550A

El intercambio de información entre el procesador y los serializadores se fundamenta en el uso intensivo de las interrupciones al procesador.

8.3.1. PROCESO DE RECEPCION DE CARACTERES

En los procesos de recepción de datos, el NS16550A puede leer en forma autónoma hasta 16 caracteres (profundidad de la FIFO del receptor) antes de solicitar la atención del procesador para transferirle los datos recibidos.

Una vez que la UART está preparada para transferir datos al CPU, genera una solicitud de servicio (activa la línea INT[7] del bus de interrupciones) y espera la atención por parte del procesador.

La cantidad de bytes que se almacenan en la FIFO de recepción puede redefinirse por programación durante la operación del sistema. Esta característica permite ajustar los niveles de disparo de interrupción dependiendo de la tarea o de la carga de trabajo actual.

La RxFIFO puede mantener hasta 16 bytes independientemente del nivel de disparo seleccionado por el CPU. Esto permite que la FIFO continúe llenándose aún después de haber solicitado la interrupción.

8.3.2. PROCESO DE TRANSMISION

La profundidad de la TxFIFO asegura que el UART puede transferir 16 caracteres hacia el equipo de comunicaciones antes de generar una nueva solicitud de interrupción de transmisión para que el procesador envíe más datos al convertidor.

Utilizar un convertidor con memoria de tipo FIFO permite que el CPU cargue cierta cantidad de caracteres en la UART (máximo 16 para el NS16550A) cada vez que cambia de contexto para dar servicio a la interrupción. Esta característica reduce el tiempo requerido por el procesador para la conmutación de contexto.

8.4. TOLERANCIA A FALLAS EN EL CANAL DE COMUNICACIONES SERIE

Para aumentar la confiabilidad de la información entregada por el canal serie al procesador, este diseño cuenta no solo con redundancia en los dispositivos de transmisión / recepción asíncrona (UART NS16550A), sino también con dos comparadores de 8 bits para validar la consistencia de la información enviada por el convertidor serie - paralelo hacia el procesador y un conjunto de registros que permiten controlar la operación de las diferentes partes del puerto de comunicaciones.

Cada vez que el procesador inicia una operación de lectura del NS16550A una unidad de comparadores valida la igualdad de la información de salida del convertidor (información a colocar sobre el bus de datos para ser leída por el procesador).

Si durante el proceso de comparación se detecta una disparidad, el comparador correspondiente genera una solicitud de interrupción de tipo INTR. Las interrupciones generadas por inconsistencia en las salidas del reloj de tiempo real corresponden a las líneas INT[5..6] del bus de interrupciones.

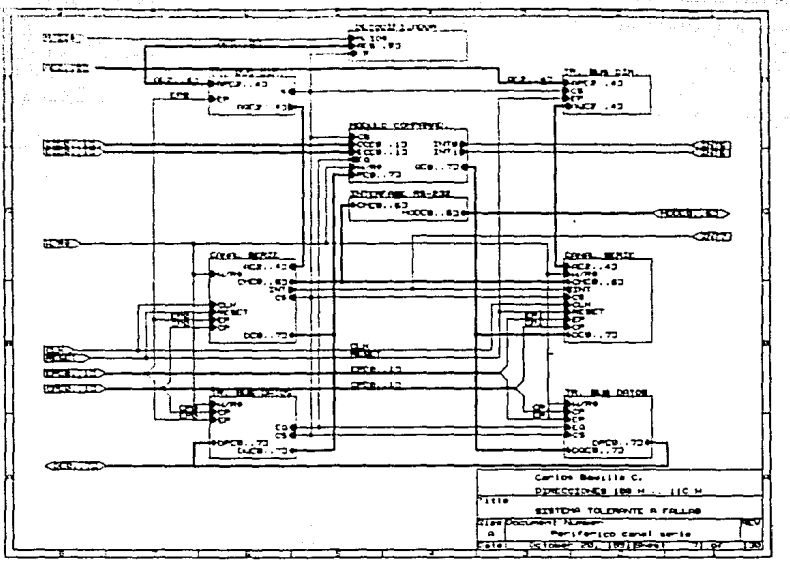
8.5. CIRCUITO DEL CANAL DE COMUNICACIONES SERIE

Referencia: láminas 71 a 80

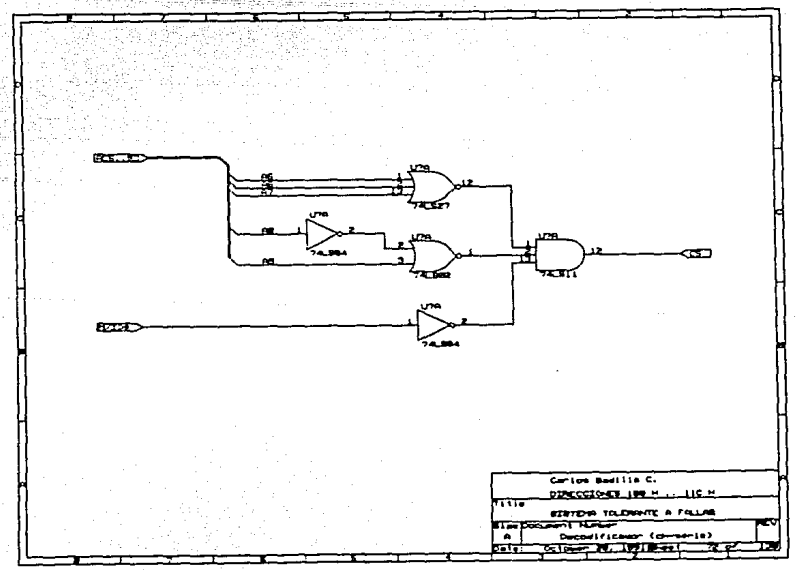
8.5.1. BUS DE DATOS

El convertidor serie - paralelo se comunica con el procesador por medio del bus de datos D0..D7. A través de este bus, el procesador envía información de inicialización, comandos de programación y los datos por enviar al equipo de comunicaciones. También se emplea para enviar hacia el procesador información de estado y los datos recibidos en serie desde el equipo de comunicaciones.

Dado que el UART NS16550A tiene buses de datos y direcciones separados, no se requiere circuitería adicional para diferenciar ambos tipos de información.



LAMINA 71. PERIFERICO CANAL SERIE



LAMINA 72. DECODIFICADOR

Al igual que en los demás periféricos, la detección de fallas se basa en el análisis de los datos que fluyen del periférico hacia el procesador.

8.5.2. LOGICA DE LECTURA / ESCRITURA

El NS16550A tiene cuatro entradas que pueden utilizarse para identificar los comandos de lectura y escritura emitidos por el procesador. Dos entradas se usan para recibir órdenes de lectura (RD, RD#) y otras dos para las órdenes de escritura (WR, WR#). En contraste, el procesador utiliza una sola línea de salida (W/R#) para activar ambos comandos.

En vista de lo anterior, la entrada RD# del NS16550A se conecta directamente a la salida W/R# del procesador 80386; la entrada WR del NS16550A se conecta a esa misma línea; la entrada RD del serializador se conecta a referencia (GND) y la entrada WR# a VCC.

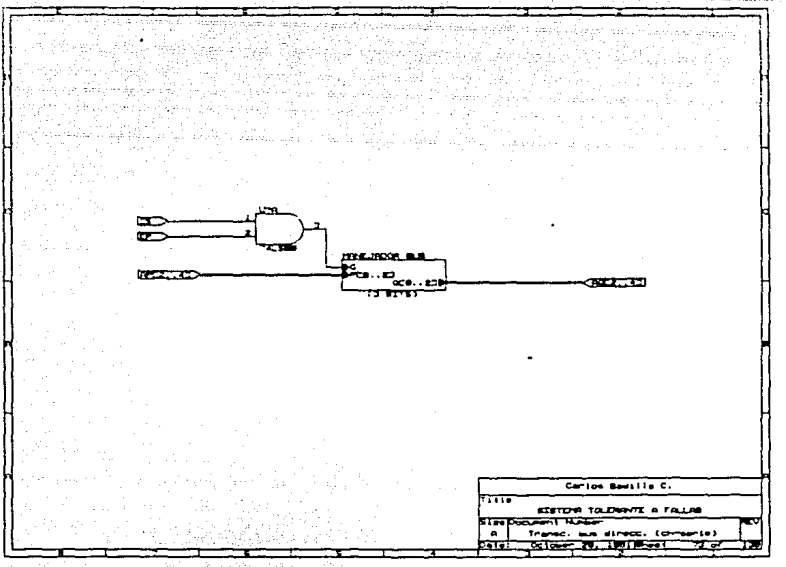
8.5.3. HABILITACION DEL CONVERTIDOR SERIE-PARALELO

El puerto de conversión serie-paraleo consta de dos dispositivos NS16550A (uno principal y otro de respaldo#) los cuales están mapeados en las mismas direcciones del espacio de puertos (100 H a 11C H)

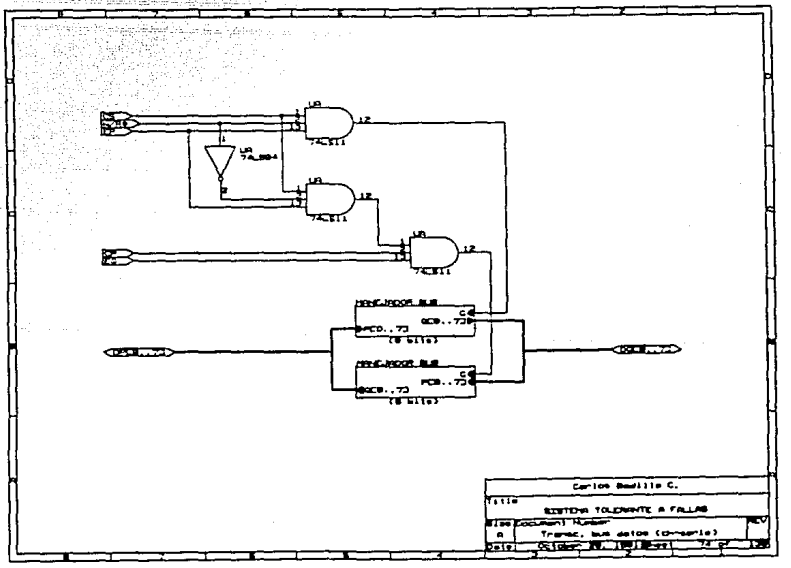
Cuando el procesador envía información al canal serie, los registros internos (sean de recepción de datos o de programación) de ambos serializadores se actualizan; durante los procesos de lectura, solo el serializador con categoría principal puede colocar información sobre el bus de datos.

Para seleccionar los convertidores se utilizan las líneas A5 a A9 del bus de direcciones. El circuito de decodificación genera la línea CS la cual se activa cada vez que el procesador entra en comunicación con el dispositivo.

La salida CS del decodificador (activa alto) se conecta a la entrada CS0 del NS16550A. Las otras líneas de selección de este dispositivo se conectan así: CS1 a VCC y CS2# a referencia (GND).



LAMINA 73. TRANSCEPTOR DEL BUS DE DIRECCIONES



LAMINA 74. TRANSCEPTOR DEL BUS DE DATOS

8.5.4. HABILITACION DE DIRECCIONES

Las líneas A2 a A4 se utilizan para direccionar los registros internos del UART. Dado que las direcciones son estables durante las operaciones de lectura / escritura, la línea ADS# (Address Strobe) se conecta físicamente a referencia ([Nati 88], Section 6.0, Pin descriptions).

8.5.5. SOLICITUDES DE SERVICIO GENERADAS POR EL SERIALIZADOR

Para solicitar la atención del procesador (para realizar procesos de transmisión o de recepción), el UART activa la línea de salida INTR (activa alto). En el sistema presentado, esta salida se conecta a la entrada INT[7] de la unidad controladora de interrupciones.

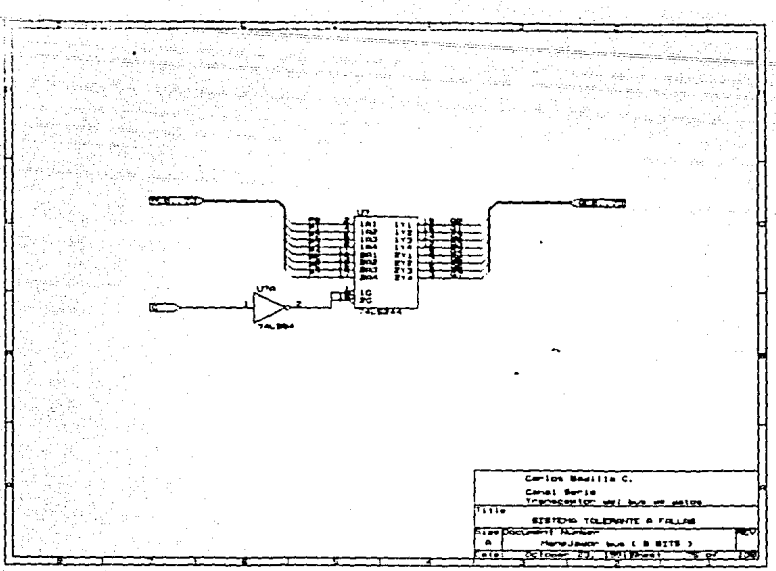
La señal OUT2# se utiliza para gobernar un buffer conectado entre la unidad de control de interrupciones y la UART. Esta conexión se basa en la aplicación presentada en [Mic 88-b]. En esta versión del diseño, la línea OUT1# se deja desconectada.

Las señales de solicitud de servicio son generadas por los dos serializadores. No obstante, solo el UART con categoría principal puede solicitar interrupciones al procesador. La categoría de estos dispositivos se define mediante el contenido de los registros de configuración de serializadores.

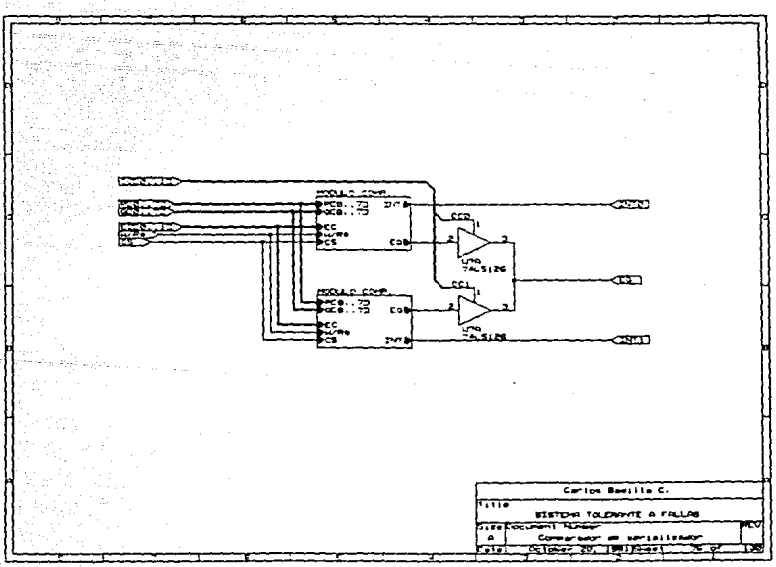
8.5.6. INICIALIZACION (Reset)

La línea de inicialización MR (master reset) del NS16550A se habilita en alto. Cuando esta línea se activa se limpia el contenido de los registros internos de programación de la UART y varía el estado de varias líneas de salida [Nati 88].

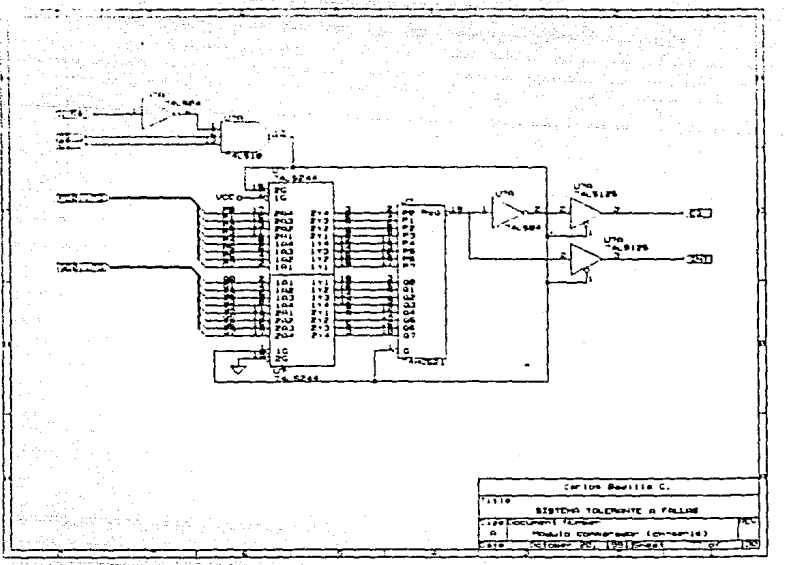
Esta entrada está conectada en forma directa a la línea de salida RESET del generador de reloj del sistema, la cual es igualmente activa en alto.



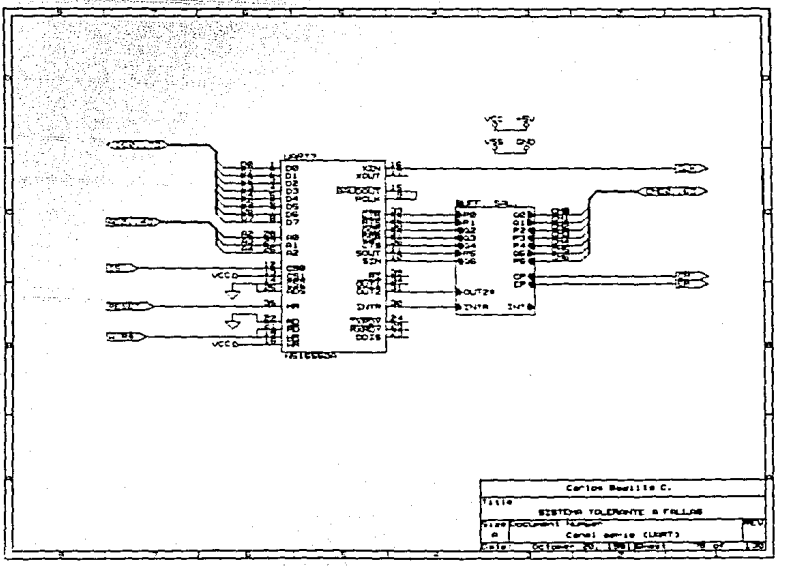
LAMINA 75. MANEJADOR DE BUS



LAMINA 76. COMPARADOR DE SERIALIZADOR



LAMINA 77. MODULO COMPARADOR



LAMINA 78. CANAL SERIE

8.5.7. FUENTE DE RELOJ

Si se utiliza un oscilador independiente para generar la referencia fundamental de tiempo, la salida de dicho oscilador debe conectarse a las líneas XIN, XOUT.

En el caso del sistema aquí propuesto, se utiliza como reloj la señal CLK generada por el 82384. Esta línea se conecta directamente a la entrada XIN de la UART NS16550A. Dado que la línea CLK se utiliza para proporcionar la señal de sincronía, la línea XOUT no se requiere y, por tanto, se deja desconectada.

8.5.8. GENERADOR DE BAUDAJE

La salida (BAUDOUT#) del NS16550A tiene una señal de reloj correspondiente a 16 veces la frecuencia del reloj de la sección del transmisor de la UART. Esta salida se conecta a la entrada RCLK (receiver clock) para manejar la lógica del transmisor interno. Esta conexión ahorra la construcción de un reloj específico para la entrada RCLK.

La frecuencia de la señal de reloj presente en BAUDOUT# corresponde a la frecuencia del oscilador principal (CLK) dividida por el divisor programado.

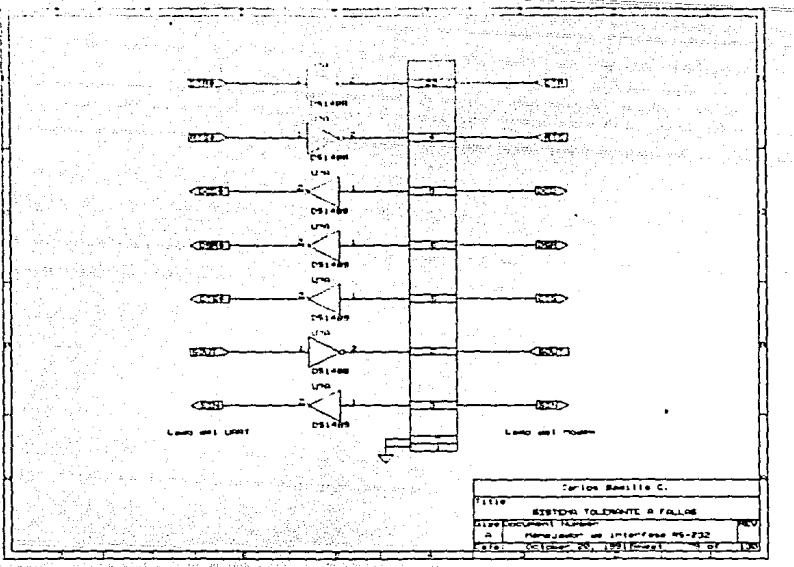
8.5.9. SEÑALES DE COMUNICACION CON MODEM

Las señales de datos y control utilizadas para la comunicación con el modem o el dispositivo externo de comunicaciones (DTR#, RTS#, DCD#, DSR#, CTS#, SIN, SOUT) se unen a un conector RS-232 de 25 pines.

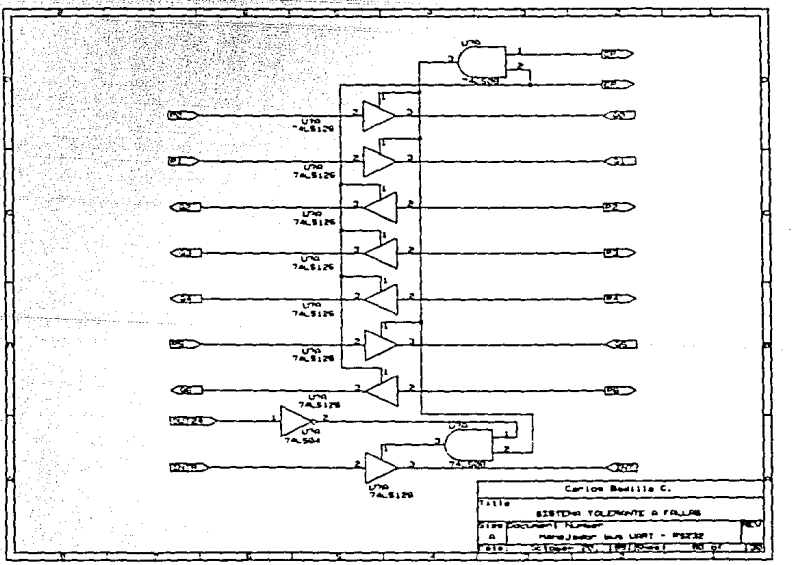
Entre el conector y el UART se conecta un conjunto de manejadores (drivers) (DS1488 para la salida; DS1489 para la entrada) que permiten ajustar los niveles TTL a las tensiones especificadas por el protocolo RS-232.

8.5.10. LINEAS RI#, TxRdy, RxRdy, DDIS

La línea RI# (Ring Indicator) se deja inactiva (conectada a VCC) ([Nati 88], Section 9.0, Typical Applications)



LAMINA 79. MANEJADOR DE INTERFASE



LAMINA 80. MANEJADOR DEL BUS UART-RS232

Las líneas TxRdy y RxRdy utilizadas para señalizar la interacción DMA - UART se dejan desconectadas ya que este diseño no contempla la utilización de dispositivos DMA.

La línea DDIS (Driver Disable) se usa, en aplicaciones típicas, para deshabilitar o para controlar la dirección de los transceptores del bus de datos colocados entre el CPU y el UART. ([Nati 88], Section 6.0, Pin descriptions).

En el sistema propuesto, la activación de los transceptores que enlazan al procesador principal con el bus del sistema se define con base en las señales de estado del procesador y el contenido de los registros de configuración de procesadores. Por consiguiente, la línea DDIS requiere ser utilizada para este fin y por ello se deja desconectada.

8.6. MODULO DE REGISTROS DE CONFIGURACION DEL SERIALIZADOR

(Referencia: lámina 81)

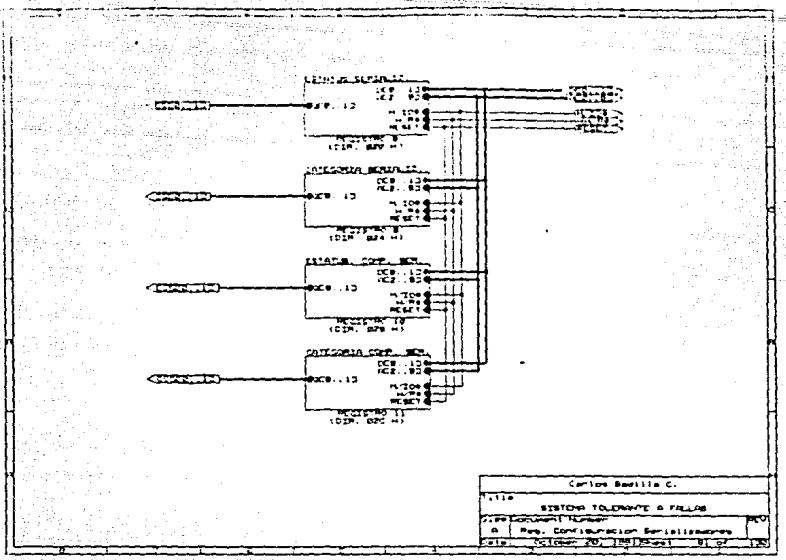
Para controlar la operación del serializador tolerante a fallas se utilizan cuatro registros de configuración cuyo contenido original se define durante la etapa de inicio (reset) del sistema. Estos registros pueden ser accesados por el procesador ya sea para operaciones de lectura o de escritura.

Las salidas de los registros de configuración de los relojes de tiempo real son de tipo latch. Esta característica permite tener disponible, en forma permanente, un conjunto de líneas independientes cuyos estados lógicos son empleados para definir las características de operación (estado y categoría) en que trabajan los convertidores serie-a-paralelo.

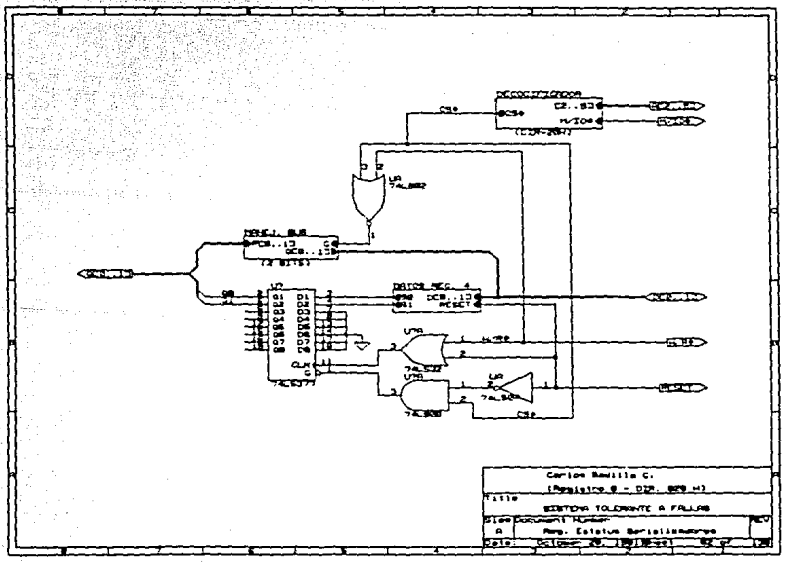
8.6.1. REGISTRO DE ESTADO DE SERIALIZADORES

Referencia: lámina 82.

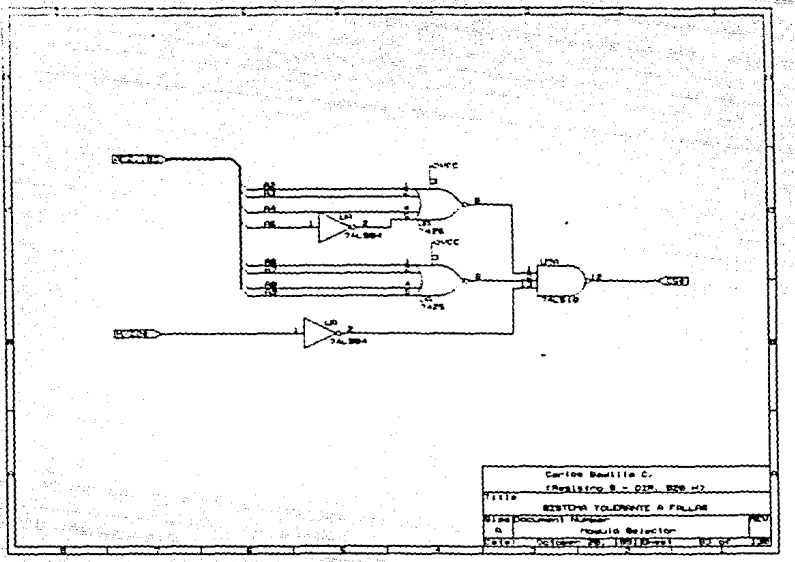
Registro número 8. Ocupa la localidad 20 H en el espacio de puertos del sistema. (lámina 83)



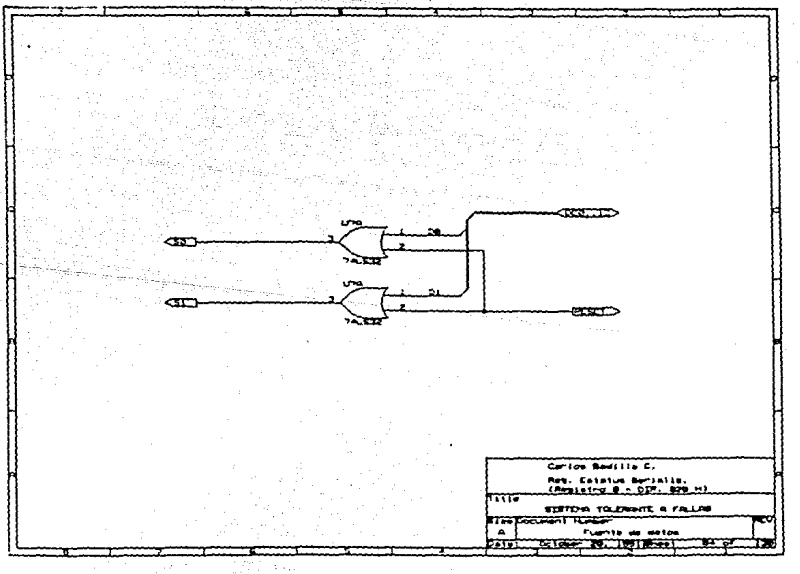
LAMINA 81.REGISTRO DE CONFIGURACION DE SERIALIZADORES



LAMINA 82. REGISTRO ESTATUS DE SERIALIZADORES



LAMINA 83. MODULO SELECTOR



LAMINA 84. FUENTE DE DATOS

Tiene 2 bits de ancho, cada uno asociado a uno de los serializadores.

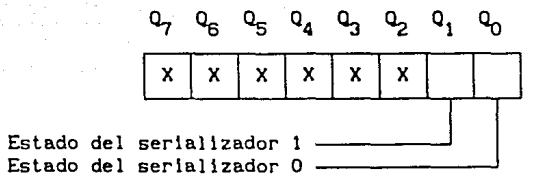


Figura 8.1 Registro de estado de serializadores

Cada bit representa uno de dos posibles estados de operación que pueden ser asignados a un módulo de conversión serie - paralelo: activo / inactivo#.

Un convertidor al que le ha sido asignado el estado activo, participa en transferencias de datos provenientes del procesador; es decir, está preparado para recibir datos de programación o información de propósito general.

Inicialmente a ambos convertidores se les asigna el estado activo. (lámina 84).

8.6.2. REGISTRO DE CATEGORIA DE SERIALIZADORES

Referencia: lámina 85.

Registro número 9. Ocupa la localidad 24 H en el espacio de puertos del sistema. (lámina 86).

Tiene un ancho de dos bits, cada uno asociado a uno de los convertidores serie-a-paralelo.

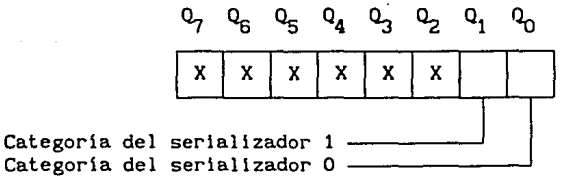
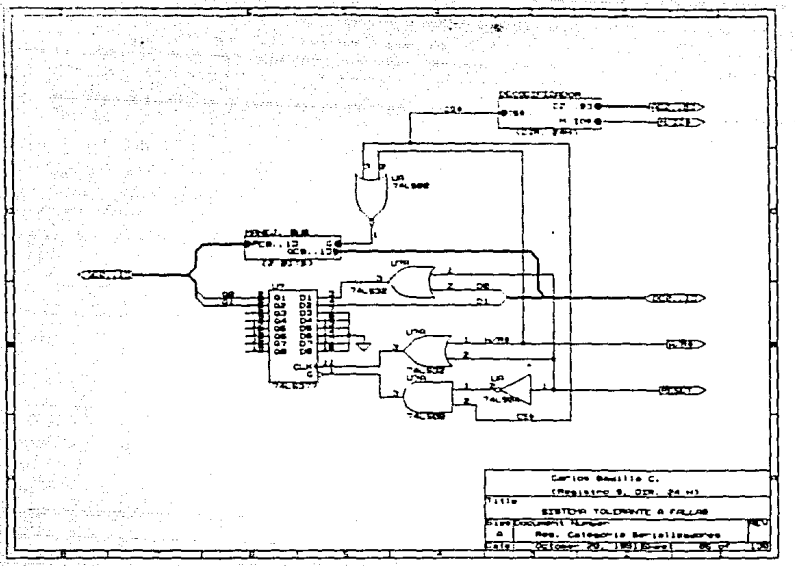
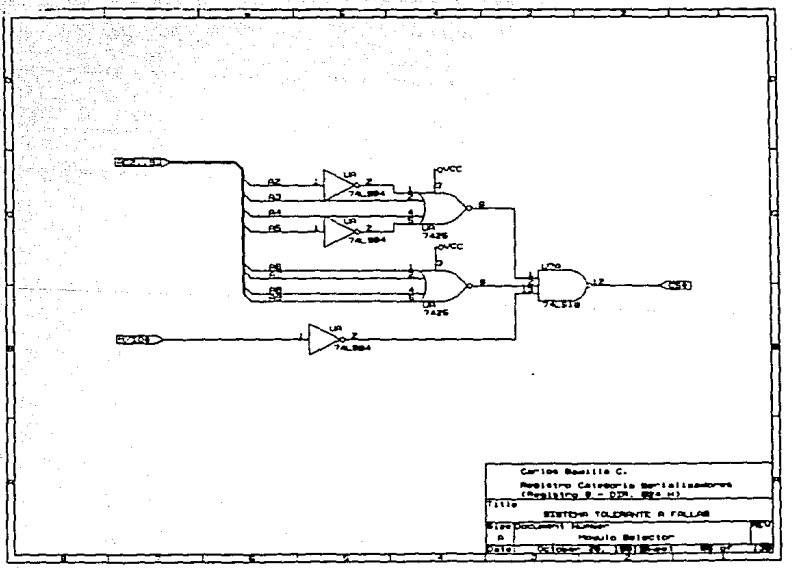


Figura 8.2 Registro de categoría de serializadores



LAMINA 85. REGISTRO CATEGORIA DE SERIALIZADORES



LAMINA 86. MODULO SELECTOR

Cada bit representa una de dos posibles categorías que pueden ser asignadas a uno de dichos módulos: principal / respaldo#.

Un convertidor al que se le asigne categoría principal está autorizado tanto para enviar como para recibir información (de control o general) del procesador, es decir, para colocar información sobre el bus de datos.

Si un convertidor se encuentra en estado activo pero tiene categoría de respaldo#, ese convertidor está preparado para recibir información del procesador; es decir, el procesador puede escribir información sobre él. No obstante, en este caso, el convertidor no está autorizado para enviar información hacia el procesador.

Inicialmente al convertidor número cero (0) se le asigna la categoría principal y al número uno (1) la categoría de respaldo#. (lámina 85).

Si uno de los comparadores de salidas de los serializadores identifica una situación anómala, dicho comparador notifica dicha condición al controlador de interrupciones (activa una solicitud de interrupción: INT[5] o INT[6])

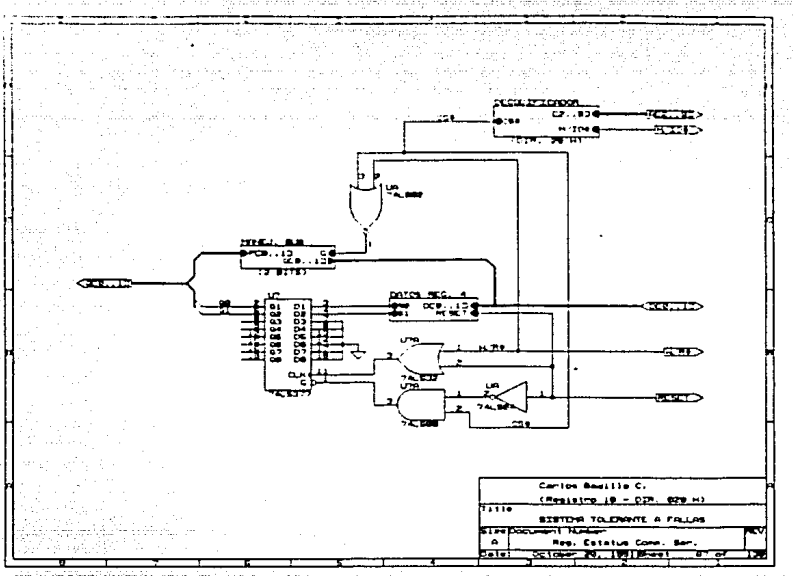
La línea de solicitud de interrupción que se active identifica al comparador que detectó la inconsistencia. Si ambas solicitudes se activan simultáneamente, se atiende primero (o exclusivamente) a la de mayor jerarquía. El orden en que sean atendidas las interrupciones debe ser definido mediante la programación del sistema operativo.

8.6.3. REGISTRO DE ESTADO DE COMPARADORES DE SERIALIZADORES

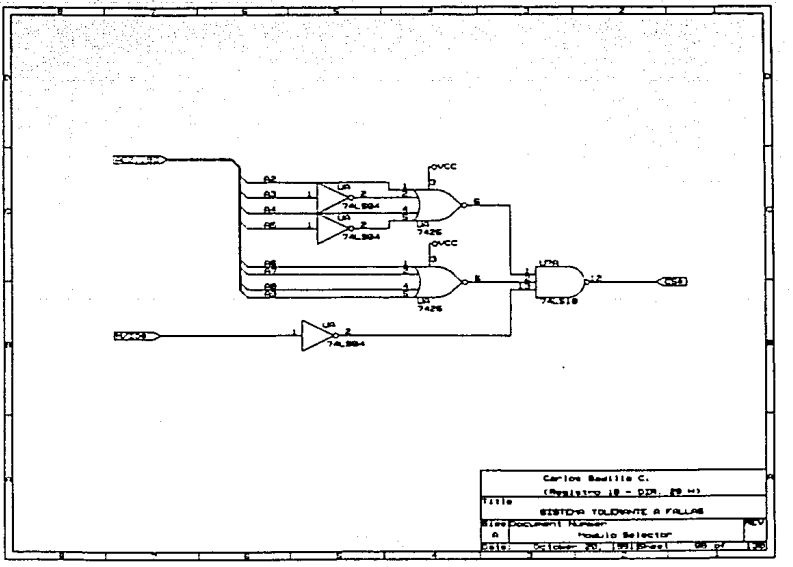
Referencia: lámina 87.

Registro número 10. Ocupa la dirección 28 H del espacio de puertos del sistema. (lámina 88)

Tiene un ancho de dos bits, cada uno asociado a uno de los



LAMINA 87. REGISTRO DE ESTATUS DE COMPARADORES DE SERIALIZADORES



LAMINA 88. MODULO SELECTOR

comparadores de salida de los convertidores serie-a-paralelo.

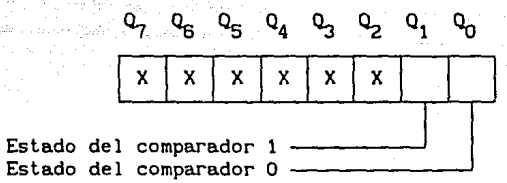


Figura 8.3 Registro de estado de comparadores de serializadores

Cada bit de este registro representa dos posibles estados de operación que pueden ser asignados a los comparadores: activo / inactivo#.

Un comparador al que le ha sido asignado el estado activo puede evaluar los datos de salida de los convertidores serie-a-paralelo.

Además, en caso de encontrar diferencias entre las salidas de los convertidores, cada comparador puede generar una solicitud de interrupción para señalizar la anomalía detectada.

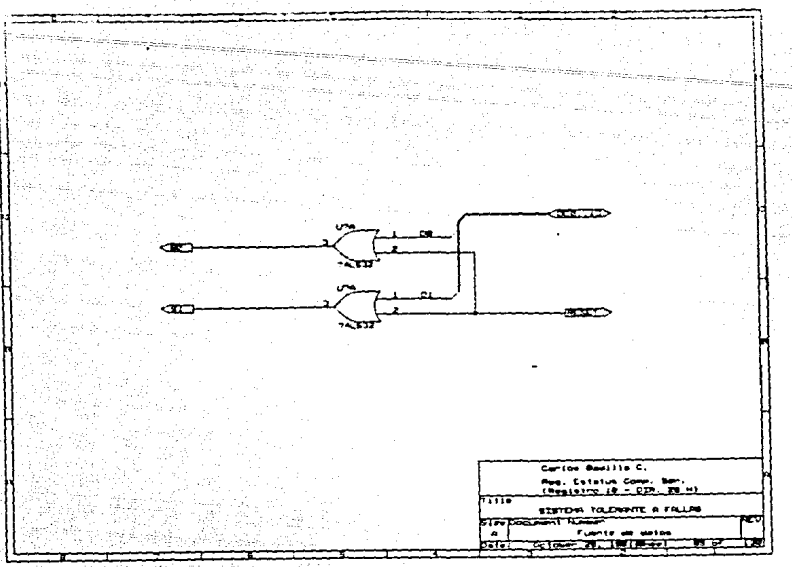
Inicialmente a ambos comparadores se les asigna el estado activo. (lámina 89)

8.6.4. REGISTRO DE CATEGORIA DE COMPARADORES DE SERIALIZADORES

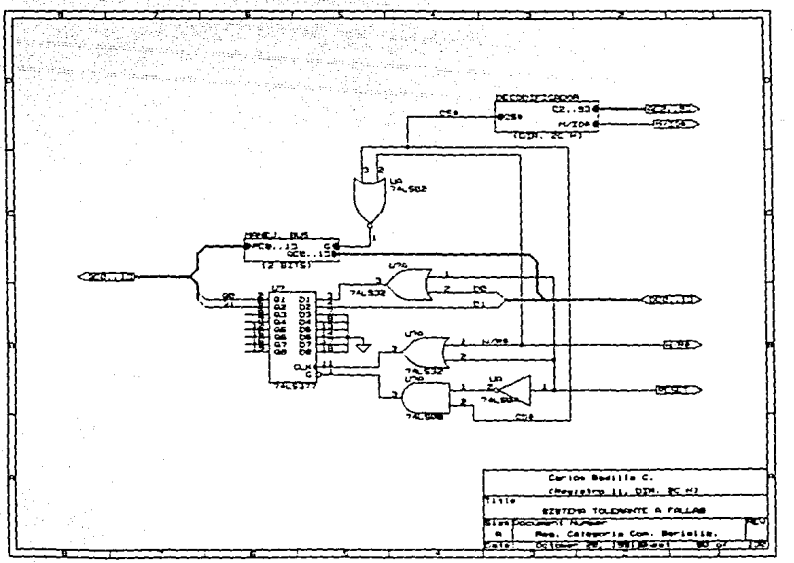
Referencia: lámina 90.

Registro número 11. Ocupa la dirección 2C H del espacio de puertos del sistema. (lámina 91)

Tiene un ancho de dos bits, cada uno asociado a uno de los comparadores de salida de serializadores.



LAMINA 89. FUENTE DE DATOS



LAMINA 90. REGISTRO DE CATEGORIA DE COMPARADORES DE SERIALIZADORES

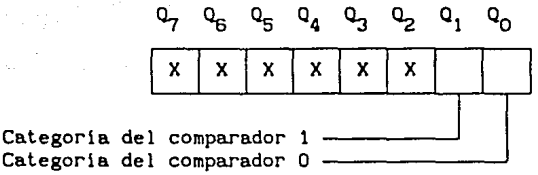


Figura 8.4 Registro de categoría de comparadores de serializadores

Cada bit de este registro representa una de dos posibles categorías que pueden ser asociadas a los comparadores: principal / respaldo#.

Las líneas de salida del registro de categoría de comparadores se emplean para habilitar o deshabilitar los circuitos de aislamiento conectados a las salidas de igualdad ("equality") de los comparadores; esto es, para identificar al comparador cuya salida de igualdad tiene efecto fuera del módulo de comparación (lámina 76).

La categoría principal identifica al comparador cuya salida de igualdad ("equality") se utiliza para gobernar los transceptores que unen el bus de datos del canal serie con el bus del sistema (lámina 71); es decir, para definir si se autoriza al serializador con categoría principal para colocar información sobre el bus de datos.

Inicialmente al comparador cero (0) se le asigna la categoría principal y al uno (1) la de respaldo#. (lámina 90).

CAPITULO 9
RELOJ DE TIEMPO REAL

INTRODUCCION

En vista de que el computador aquí reportado está diseñado para aplicaciones de percepción remota, es necesario contar con un dispositivo que proporcione la información de tiempo y calendario para registrar el momento en que se realiza la captura de los datos físicos.

Para realizar esta tarea se utiliza un reloj de tiempo real al cual le han agregado una serie de recursos que le permiten realizar ciertas funciones de tolerancia a fallas.

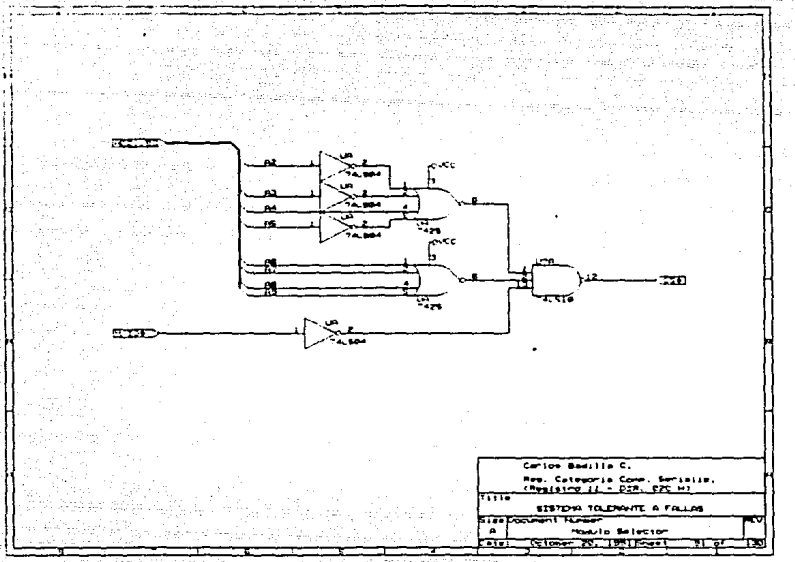
Este subsistema está provisto de dos dispositivos de medición de tiempo DS1287 y un bloque de comparadores que vigila la igualdad en los datos de salida del módulo de tiempo real cada vez que el procesador lo accesa en operaciones de lectura (de datos almacenados o estado de los relojes), de forma similar al circuito de serializadores.

Cuando el manejador de interrupciones informa al procesador sobre una anomalía en las salidas del reloj de tiempo real, el procesador entra en una rutina de servicio de la interrupción que permita diagnosticar la falla u origen del error detectado y eventualmente proceder a aislar al periférico fallado.

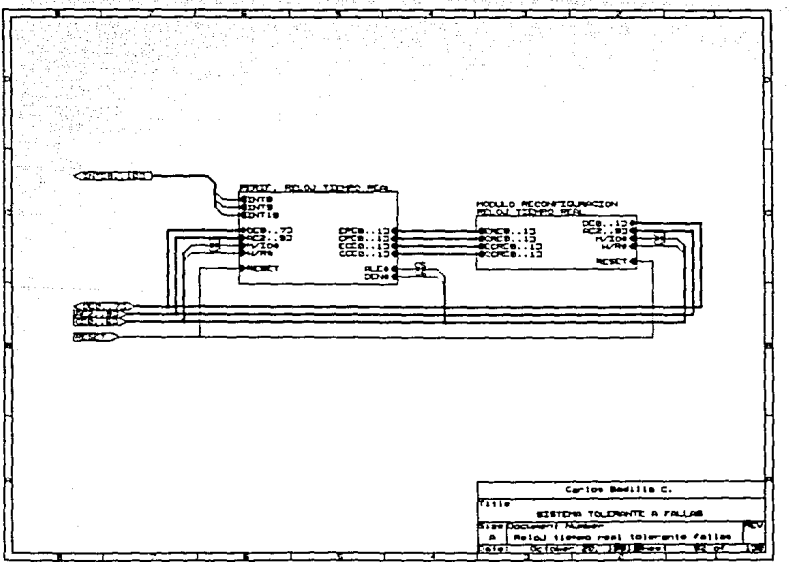
Las rutinas de diagnóstico de fallas pueden efectuarse de forma local o remota de acuerdo con las opciones que se incorporen en la programación del sistema operativo.

9.1. RELOJ DE TIEMPO REAL DS1287

El DS1287 es un reloj de tiempo real que presenta una estructura de entrada - salida y líneas de control que facilitan la interfase con un microprocesador. Incluye pila de litio, oscilador de cuarzo y otros circuitos de apoyo. Está encapsulado dentro de un chip de 24 pines



LAMINA 91. MODULO SELECTOR



LAMINA 92. RELOJ DE TIEMPO REAL TOLERANTE A FALLAS

Las funciones del DS1287 incluyen:

a. reloj no volátil de hora-del-día (segundos, minutos, horas)
b. calendario de 100 años (día de la semana, fecha, mes y año (con compensación automática para cálculo del año bisiesto)
c. alarma

Ofrece la salida de tiempo, calendario y alarma en representación binaria o en BCD. Puede presentar la información de tiempo en formato de 12 o 24 horas con AM y PM en el modo de 12 horas.

El DS1287 presenta bajo consumo de potencia y tiene una pila recargable de litio que ofrece retención de datos durante 10 años en ausencia de alimentación externa.

El reloj cuenta con 64 localidades de 8 bits de memoria RAM no volátil, de las cuales, 14 bytes se utilizan para almacenar la información de reloj y para los registros de control y 50 bytes de RAM para usos de propósito general.

En total este dispositivo utiliza 6 líneas de direccionamiento (A0..A5) y está mapeado en las localidades 200 H ..2FC H.

Cuenta con una señal de salida de onda cuadrada programable.

Los datos y direcciones utilizan una trayectoria multiplexada. Durante el direccionamiento son significativas las líneas A0..A5; durante las transferencias de datos son relevantes todas las líneas D0..D7.

Ofrece tres tipos de interrupciones separadamente enmascarables y verificables por programación:

a) Alarma de hora del día desde una vez por segundo hasta una vez por día.
b) Tasas periódicas desde 122 µsec hasta 50 msec.
c) Fin de ciclo de actualización de reloj

Permite seleccionar el tipo de temporización a utilizar: bus Motorola o bus Intel.

9.2. PROCESOS DE LECTURA - ESCRITURA EN EL DS1287

En los procesos de lectura - escritura, durante la primer parte del ciclo de bus, las líneas AD0..AD5 se comportan como bus de direcciones. La información presente en el bus en ese momento debe validarse antes del flanco negativo de la señal AS/ALE. Para realizar la captura de direcciones, el DS1287 utiliza la señal ALE# proveniente de la lógica de control del bus del microprocesador 80386.

Durante la segunda parte del ciclo, las líneas AD0..AD7 se comportan como bus de datos. La información de datos debe validarse durante el tiempo en que las señales en los pines DS o WR# sea válida.

9.3. TOLERANCIA A FALLAS EN EL RELOJ DE TIEMPO REAL

Para verificar la consistencia de la información enviada por el reloj de tiempo real hacia el procesador se utilizan dos comparadores de ocho bits.

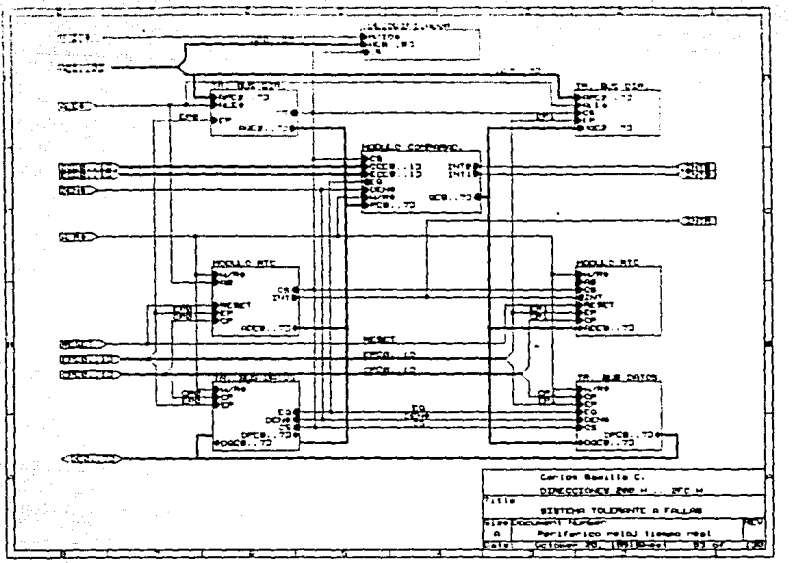
Si durante este proceso de comparación alguno de los comparadores detecta disparidad, dicho comparador genera una solicitud de interrupción de tipo INTR. Las interrupciones generadas por inconsitencia en las salidas del reloj de tiempo real corresponden a las líneas INT[8..9] del bus de interrupciones.

9.4. CIRCUITO DEL RELOJ DE TIEMPO REAL

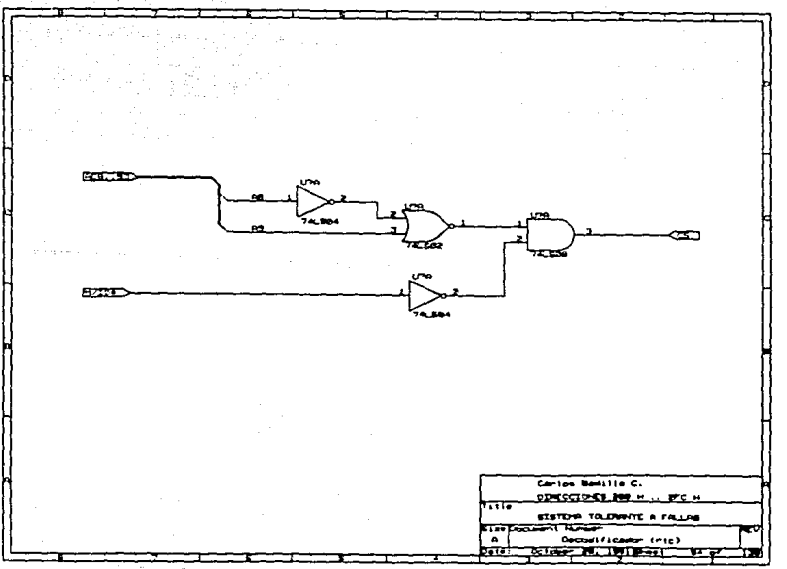
Referencia: láminas 92 a 100.

9.4.1. BUS DE DIRECCIONES Y DATOS MULTIPLEXADO

El reloj de tiempo real se comunica con el procesador mediante el bus multiplexado de datos y direcciones AD0..AD7. Es decir, la información de datos y la de direcciones utilizan la misma trayectoria.



LAMINA 93. PERIFERICO RELOJ TIEMPO REAL



LAMINA 94. DECODIFICADOR

Esta ruta se usa para recibir información de inicialización, comandos o datos de propósito general procedentes del procesador y para enviar información de estado y contenido de las localidades de memoria interna del DS1287 (archivo de registros) hacia el CPU.

La información presente en las líneas AD0..AD5 se interpreta como direcciones durante la primera parte del ciclo de bus. En ese momento las líneas AD6..AD7 no se utilizan. Durante la segunda parte del ciclo de bus las líneas AD0..AD7 se comportan como bus de datos.

Dado que el procesador 80386 utiliza trayectorias diferentes para datos y direcciones, se utiliza un par de circuitos transceptores (lámina 95..97) para establecer el enlace entre los buses del procesador y el bus del DS1287.

Para gobernar los procesos de comunicación con el DS1287 se utilizan las líneas DEN# y ALE# generadas por la lógica de control del bus del procesador 80386.

La línea ALE# informa al DS1287 el momento en que la información de direcciones proveniente del procesador es válida. La línea DEN# se emplea para identificar presencia de datos válidos en el bus de datos

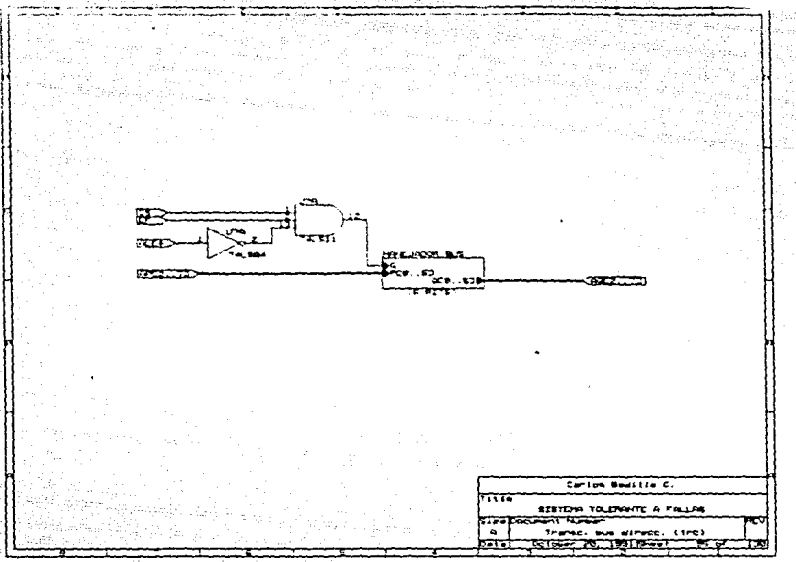
9.4.2. LOGICA DE LECTURA / ESCRITURA

Dado que este sistema está basado en el procesador 80386 la entrada MOT del DS1287 se conecta a referencia (GND) para seleccionar temporización Intel.

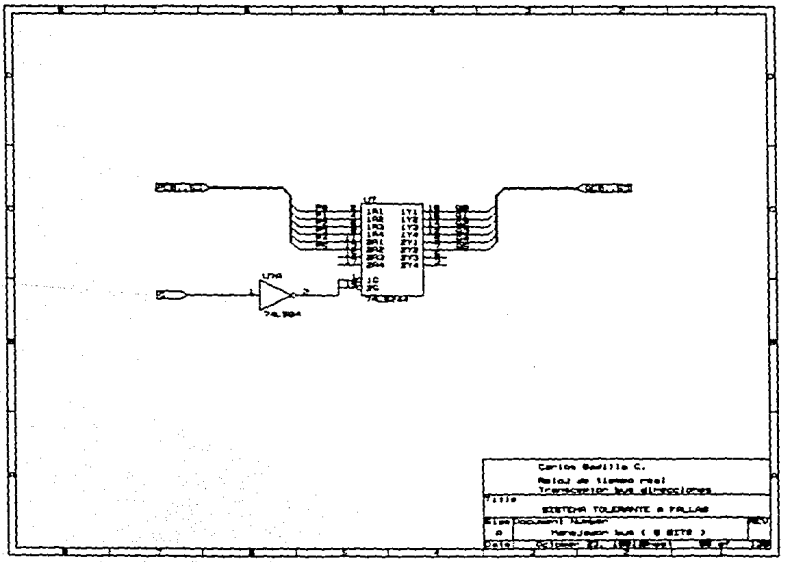
El tipo de temporización escogida determina el nombre y modo de operación de los pines usados para realizar los procesos de lectura - escritura.

9.4.2.1. LOGICA DE LECTURA

Dado que la entrada MOT = 0 (está conectada a referencia), la línea DS (Data Strobe) del DS1287 se llama RD# (Read) y es activa en bajo.



LAMINA 95. TRANSCEPTOR DE BUS DE DIRECCIONES



LAMINA 96. MANEJADOR DE BUS

Esta entrada se utiliza para identificar el comando de lectura emitido por el procesador, por lo cual está conectada en forma directa a la salida W/R# del módulo de procesamiento.

El tiempo durante el que RD# esté activo identifica el tiempo durante el cual el DS1287 presenta datos válidos en el bus AD0..AD7, es decir, el tiempo durante el cual el procesador está autorizado para leer información del reloj de tiempo real.

La entrada RD# del DS1287 tiene el mismo significado que la entrada OE# (output enable) en una memoria típica.

9.4.2.2 LOGICA DE ESCRITURA

Puesto que se utiliza temporización Intel (MOT = 0), la entrada R/W# del DS1287 se llama WR# y es activa en bajo. Esta entrada se utiliza para reconocer los comandos de escritura emitidos por el procesador.

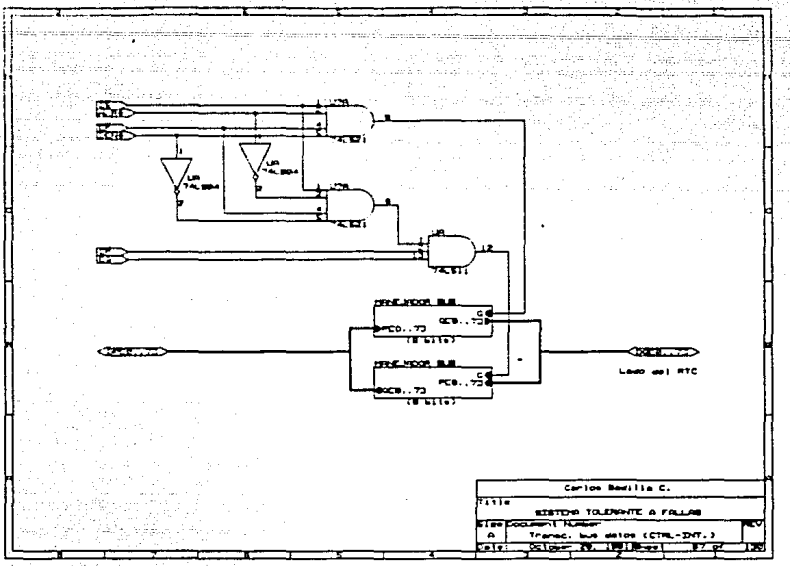
Puesto que el procesador usa una sola línea para gobernar los procesos de lectura - escritura, la entrada WR# se conecta a la línea W/R# del módulo de procesamiento por medio de un negador.

La entrada WR# del DS1287 tiene el mismo significado que la entrada WE# (write enable) en una memoria típica.

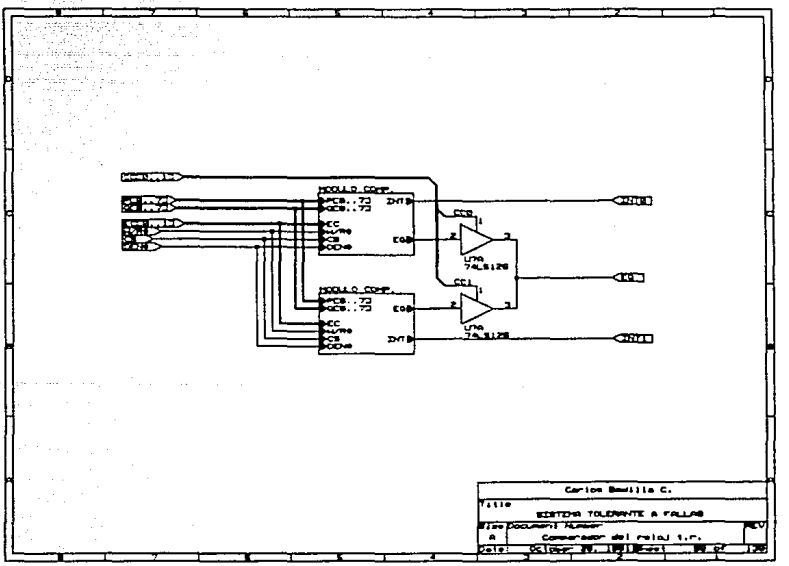
9.4.3. HABILITACION DEL RELOJ DE TIEMPO REAL

Para realizar transferencias del procesador hacia o desde el reloj de tiempo real, se requiere activar la línea de CS#. Las funciones de cuenta de tiempo y calendario son independientes de esta señal, la cual es necesaria únicamente cuando se realizan procesos de comunicación con el procesador.

Los dos relojes utilizados para llevar la cuenta de tiempo y calendario están mapeados en las mismas localidades del espacio de puertos. A estas unidades se les ha asignado las direcciones 200 H hasta 2FC H.



LAMINA 97. TRANSCEPTOR DEL BUS DE DATOS



LAMINA 98. COMPARADOR DEL RELOJ

Cuando el procesador envía información hacia el reloj se actualizan los registros internos de ambas unidades; no obstante, durante los procesos de lectura, solo uno de ellos (el que está en categoría principal) coloca información sobre el bus de datos.

Para seleccionar los manejadores de interrupción se utilizan las líneas A2 a A9 del bus de direcciones. El circuito de decodificación genera la línea CS# la cual se activa para identificar los procesos de comunicación con el procesador.

9.4.4. SOLICITUDES DE SERVICIO GENERADAS POR EL RELOJ DE TIEMPO REAL

El reloj de tiempo real está en capacidad de generar solicitudes de servicio (interrupciones de alarma) cuando se cumpla alguna de las condiciones definidas por programación

Las señales de alarma son generadas por los dos relojes. No obstante, solo el reloj con categoría principal puede solicitar interrupciones al controlador 8259A.

9.4.5. INICIALIZACION (Reset)

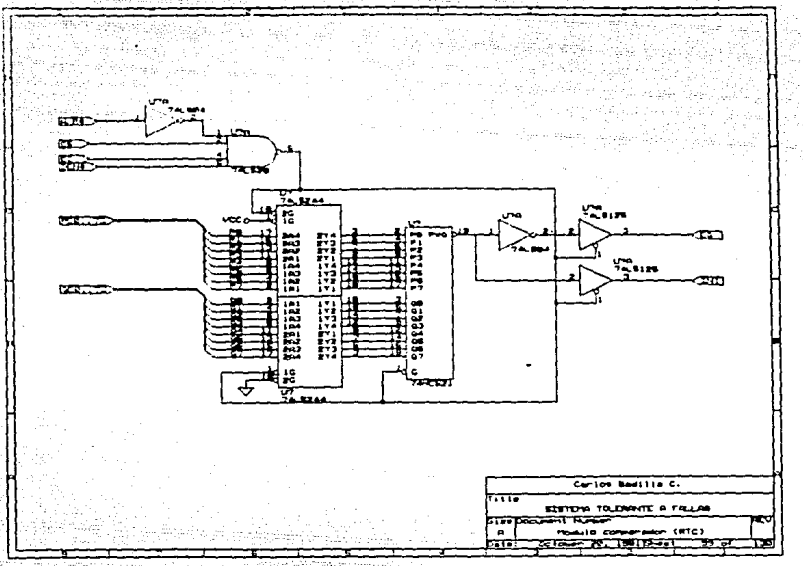
La línea de inicialización RESET# es activa en bajo. Esta línea debe ser activada para inicializar los registros de programación y control.

Esta entrada se conecta mediante un circuito negador a la línea de salida RESET del generador de reloj del sistema, la cual es activa en alto.

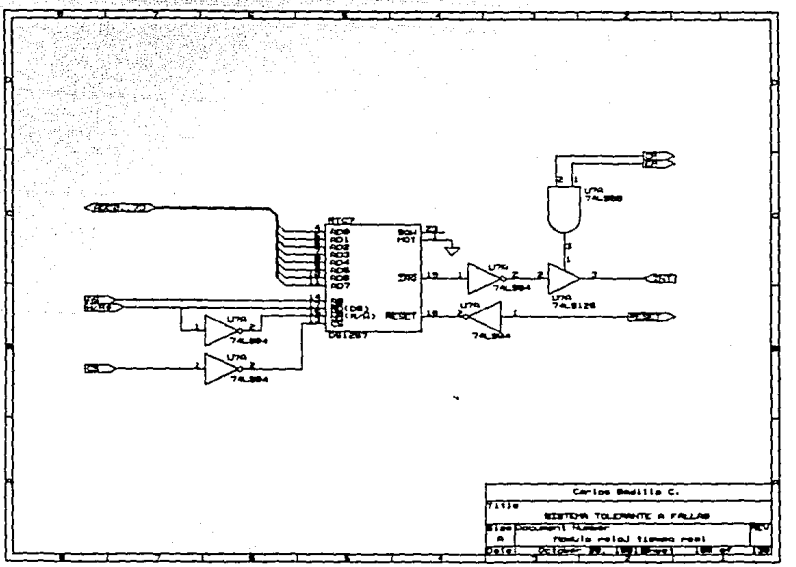
La activación de la entrada RESET# no tiene efecto sobre el reloj, el calendario y el área de memoria RAM interna.

9.4.6. MEMORIA RAM INTERNA

El reloj de tiempo real DS1287 cuenta con un espacio de 50 localidades de memoria RAM no volátiles que pueden usarse para propósitos generales.



LAMINA 99. MODULO COMPARADOR



LAMINA 100. MODULO RELOJ DE TIEMPO REAL

Un posible uso de esta memoria es dar apoyo a la ejecución de las funciones de tolerancia a fallas.

Como ejemplo de este uso puede mencionarse el almacenamiento de constantes que representen resultados de algoritmos de diagnóstico. Asimismo esta memoria puede utilizarse para almacenar códigos de iniciación de diferentes periféricos e incluso para almacenar los últimos códigos de error (transitorios o permanentes) detectados por el circuito de votación.

9.5. MODULO DE REGISTROS DE CONFIGURACION DE RELOJES DE TIEMPO REAL

(Referencia: lámina 101)

Para controlar la operación del reloj de tiempo real tolerante a fallas se utilizan cuatro registros de configuración cuyo contenido original se define durante la etapa de iniciación (reset) del sistema. El procesador puede accesar estos registros ya sea para operaciones de lectura o de escritura.

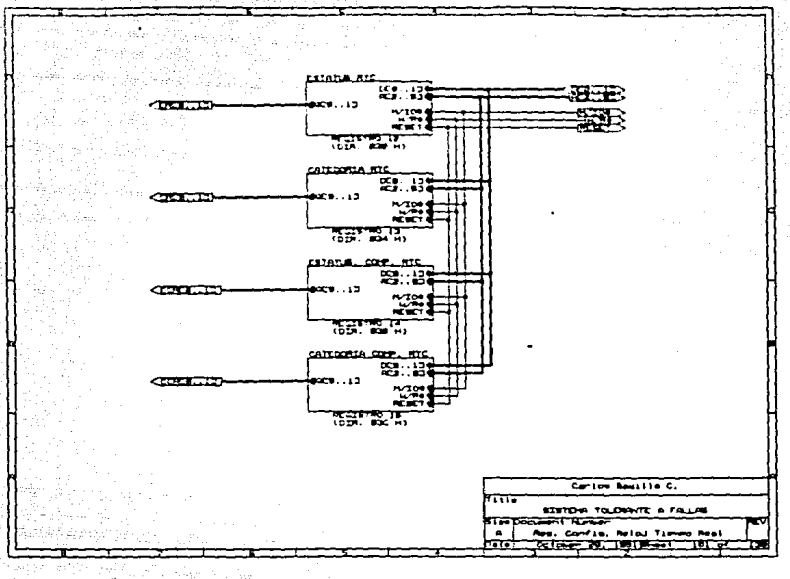
Las salidas de los registros de configuración de los relojes de tiempo real son de tipo latch. Esto permite tener disponible, en forma permanente, un conjunto de líneas independientes cuyos estados lógicos se emplean para definir las características de operación (estado y categoría) en que trabajan las unidades de manejo de tiempo y calendario.

9.5.1. REGISTRO DE ESTADO DE RELOJES DE TIEMPO REAL

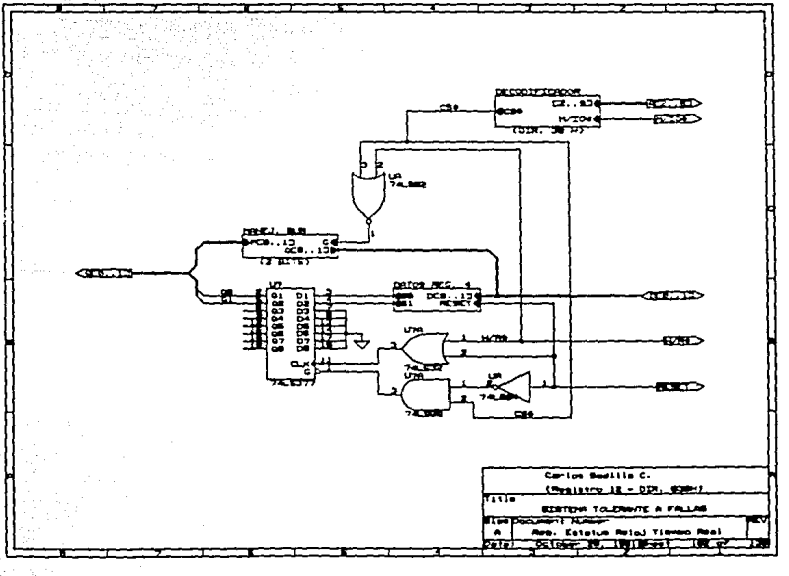
Referencia: lámina 102.

Registro número 12. Ocupa la localidad 30 H en el espacio de puertos del sistema. (lámina 103)

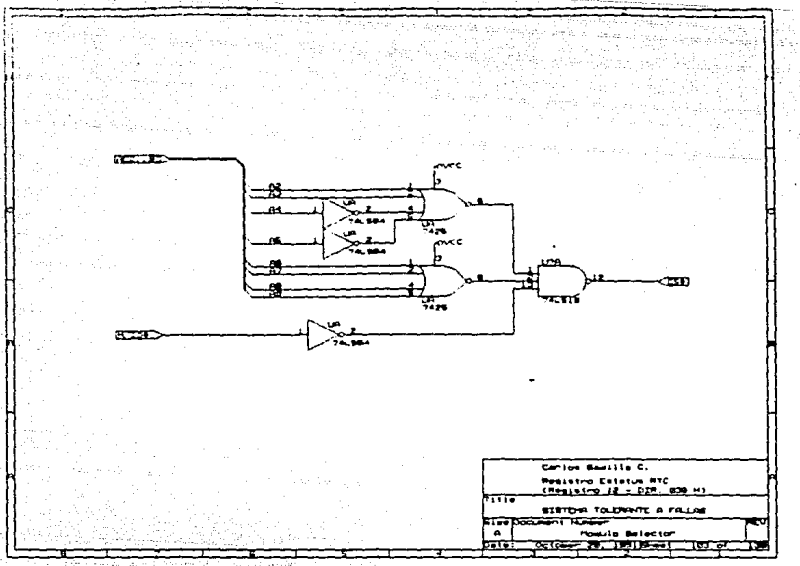
Tiene 2 bits de ancho, cada uno de ellos asociado a uno de los relojes de tiempo real.



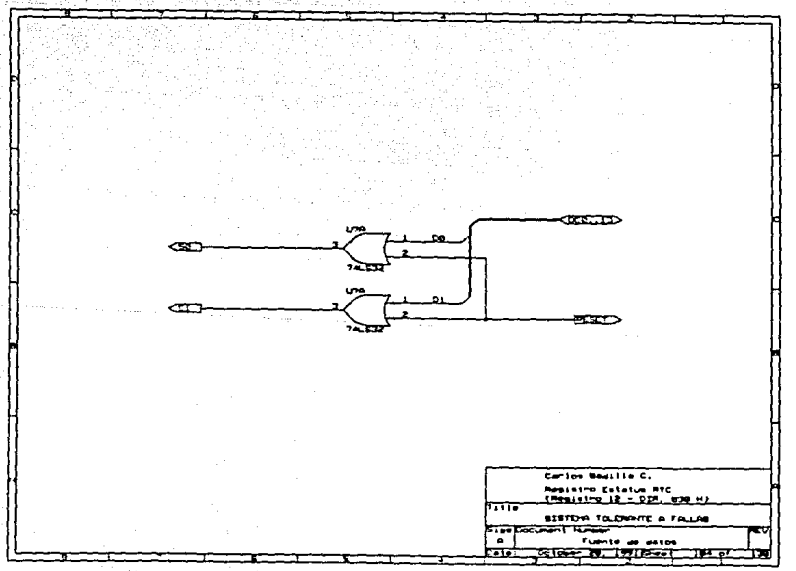
LAMINA 101. REGISTRO CONFIG. RELOJ DE TIEMPO REAL



LAMINA 102. REGISTRO ESTATUS RELOJ DE TIEMPO REAL



LAMINA 103. MODULO SELECTOR



LAMINA 104. FUENTE DE DATOS

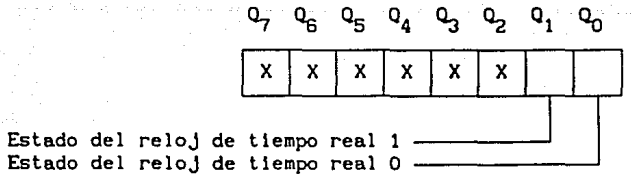


Figura 9.1 Registro de estado de relojes de tiempo real

Cada bit representa uno de dos posibles estados de operación que pueden asignarse a un de reloj de tiempo real: activo / inactivo#.

Un módulo de reloj programado como activo, puede participar en transferencias de datos provenientes del procesador; es decir, está preparado para recibir datos de programación o información de propósito general.

Inicialmente a ambos módulos de reloj se les asigna el estado activo. (lámina 104).

9.5.2. REGISTRO DE CATEGORIA DE RELOJES DE TIEMPO REAL

Referencia: lámina 105.

Registro número 13. Ocupa la localidad 34 H en el espacio de puertos del sistema. (lámina 106).

Tiene un ancho de dos bits, cada uno de ellos asociado a uno de los módulos de reloj.

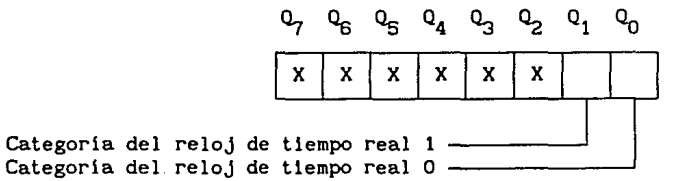
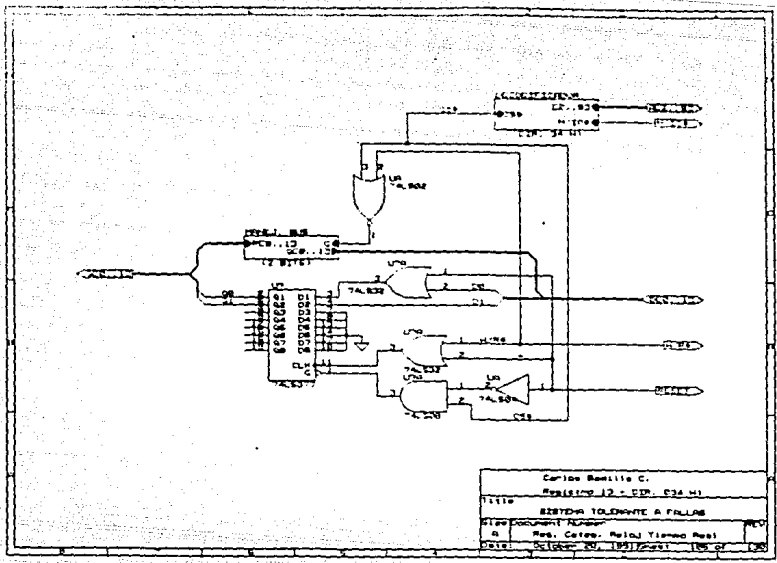
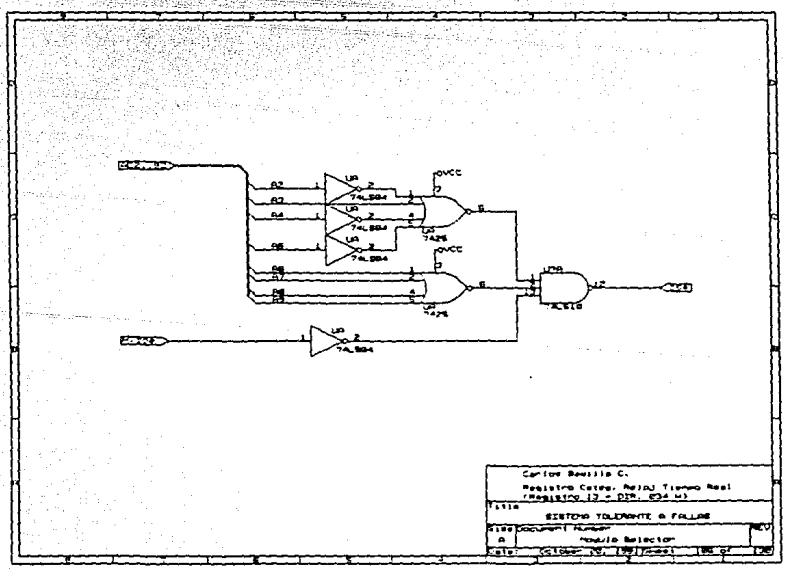


Figura 9.2 Registro de categoría de relojes de tiempo real

Cada bit representa una de dos posibles categorías que pueden ser asignadas a uno de dichos módulos: principal / respaldo#.



LAMINA 105. REGISTRO CATEGORIA RELOJ DE TIEMPO REAL



LAMINA 106. MODULO SELECTOR

Un reloj al que se le asigne categoría principal puede enviar y recibir información (de control o general) del procesador, es decir, para colocar información sobre el bus de datos.

Si un reloj se encuentra en estado activo pero tiene categoría de respaldo#, dicho reloj puede recibir información del procesador; es decir, el procesador puede escribir información sobre él. No obstante, en este caso, el reloj no puede enviar información hacia el procesador.

Inicialmente al reloj número cero (0) se le asigna la categoría principal y al número uno (1) la de respaldo#. (lámina 105).

Si uno de los comparadores de salidas de los relojes identifica una situación anómala, dicho comparador notifica dicha condición al controlador de interrupciones (activa una solicitud de interrupción: INT8 o INT9)

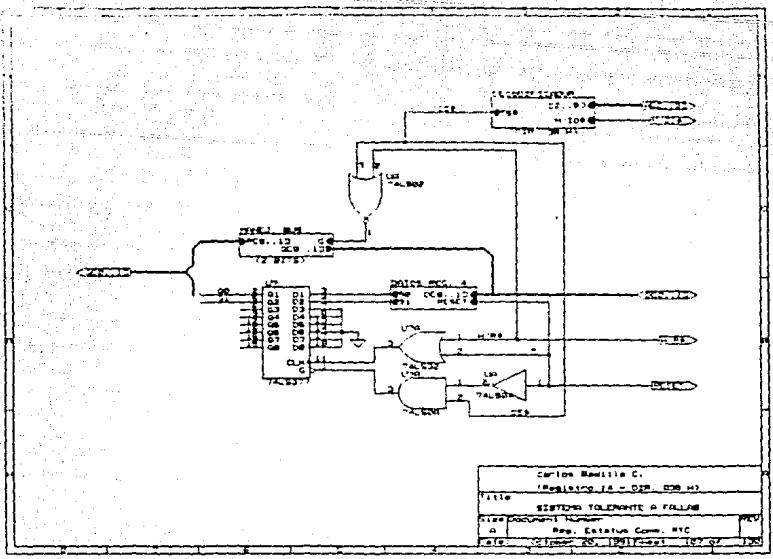
La línea de interrupción que se active identifica al comparador que detectó la inconsistencia. Si ambas solicitudes de interrupción se activan simultáneamente, se atiende primero (o exclusivamente) a la de mayor jerarquía. El orden en que sean atendidas las interrupciones debe definirse en la programación del sistema operativo.

9.5.3. REGISTRO DE ESTADO DE COMPARADORES DE RELOJES DE TIEMPO REAL

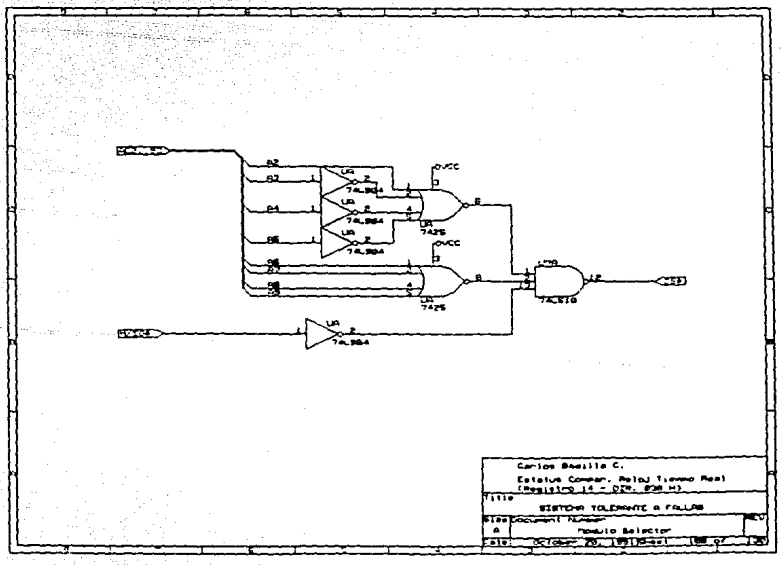
Referencia: lámina 107.

Registro número 14. Ocupa la dirección 38 H del espacio de puertos del sistema. (lámina 108)

Tiene un ancho de dos bits, cada uno de ellos asociado a uno de los comparadores de salida de relojes de tiempo real.



LAMINA 107. REGISTRO ESTATUS COMPARADOR



LAMINA 108. MODULO SELECTOR

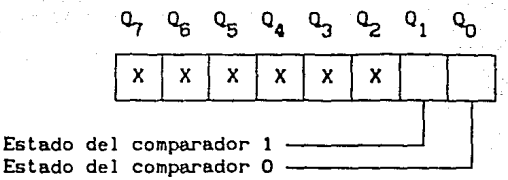


Figura 9.3 Registro de estado de comparadores de relojes de tiempo real

Cada bit de este registro representa dos posibles estados de operación que pueden ser asignados a los comparadores: activo / inactivo#.

Un comparador al que le haya sido asignado el estado activo puede evaluar los datos de salida de los relojes de tiempo real.

Además, en caso de encontrar diferencias entre las salidas de los relojes de tiempo real, cada comparador puede generar una solicitud de interrupción para indicar la anomalía detectada.

Inicialmente a ambos comparadores se les asigna el estado activo. (lámina 109)

9.5.4. REGISTRO DE CATEGORIA DE COMPARADORES DE RELOJES DE TIEMPO REAL

Referencia: lámina 110.

Registro número 15. Ocupa la dirección 3C H del espacio de puertos del sistema. (lámina 111)

Tiene un ancho de dos bits, cada uno de ellos asociado a uno de los comparadores de salida relojes de tiempo real.

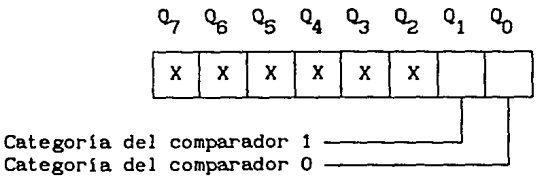
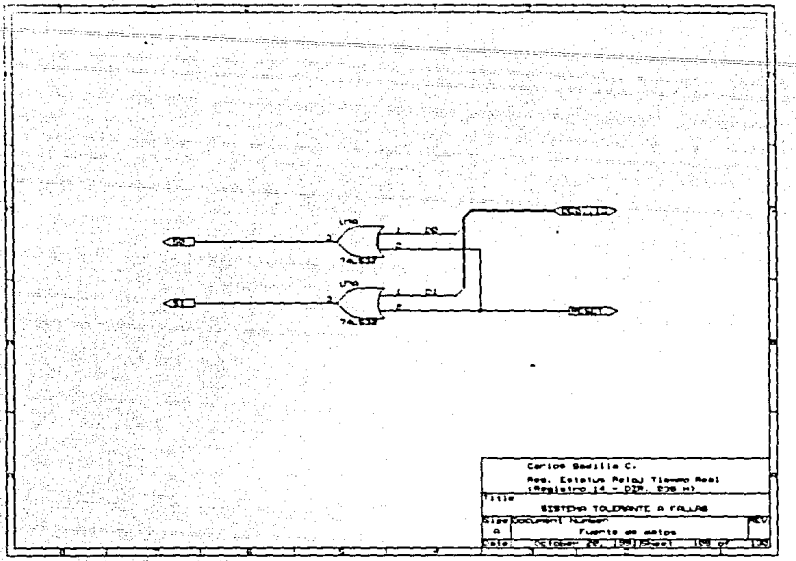
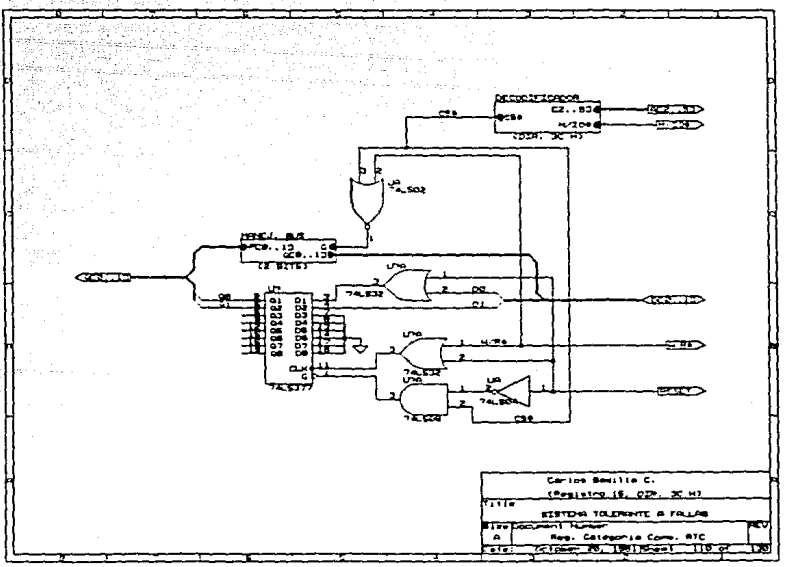


Figura 9.4 Registro de categoría de comparadores de relojes de tiempo real



LAMINA 109. FUENTE DE DATOS



LAMINA 110. REGISTRO CATEGORIA COMPARADOR

Cada bit de este registro representa una de dos posibles categorías que pueden asociarse a los comparadores: principal / respaldo#.

Las líneas de salida del registro de categoría de comparadores se emplean para habilitar o deshabilitar los circuitos de aislamiento conectados a las salidas de igualdad ("equality") de los comparadores; esto es, para identificar al comparador cuya salida de igualdad tiene efecto fuera del módulo de comparación (lámina 98).

La categoría principal identifica al comparador cuya salida de igualdad ("equality") se utiliza para gobernar los transceptores que unen el bus de datos del reloj de tiempo real con el bus del sistema (lámina 93); es decir, para definir si se autoriza al reloj de tiempo real con categoría principal para colocar información sobre el bus de datos.

Inicialmente al comparador cero (0) se le asigna la categoría principal y al uno (1) la de respaldo#. (lámina 110).

CAPITULO 10
UNIDAD DE CONTROL DE INTERRUPCIONES DEL SISTEMA

INTRODUCCION

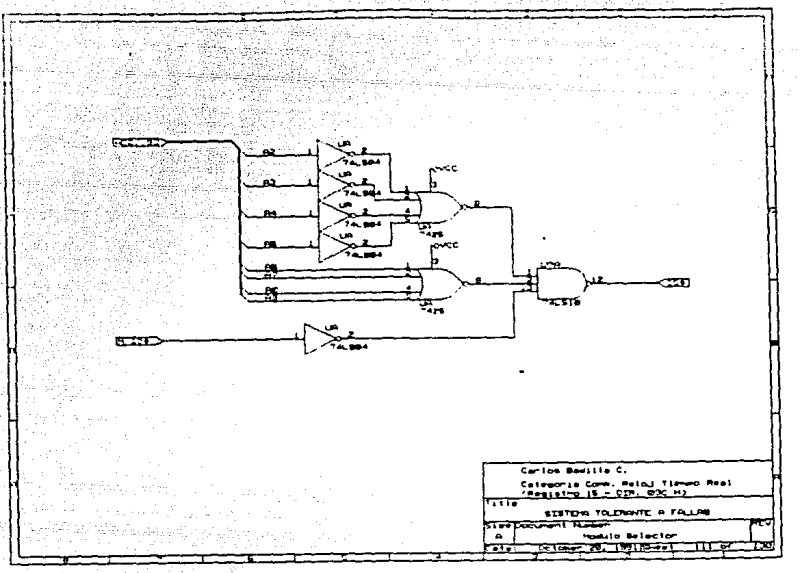
Este sistema hace uso extensivo de interrupciones al procesador para para atender las operaciones de servicio a los dispositivos periféricos, así como para apoyar las funciones de tolerancia a fallas.

Bajo este esquema, la mayoría de las funciones de tolerancia a fallas (tales como diagnóstico de fallas y recuperación de sistema) son rutinas del sistema operativo que se invocan por la unidad de control de interrupciones en el momento en que los circuitos de detección de errores descubran una condición de operación anómala.

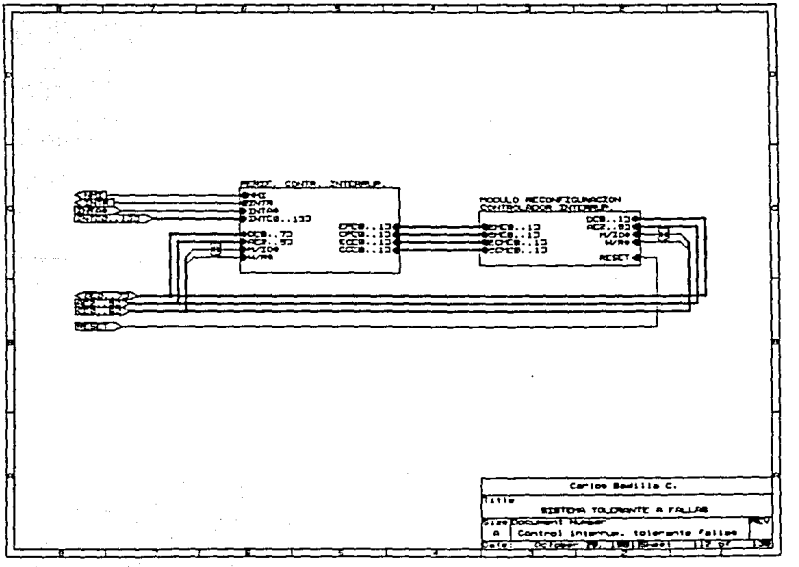
La unidad controladora de interrupciones utilizada en este computador maneja niveles de prioridades, puede atender una cantidad significativa de interrupciones e incluye la posibilidad de crecimiento de las fuentes de interrupción.

En este sistema, en el que la tolerancia a fallas se conduce por interrupciones, cada una de las posibles fuentes de error previstas en el diseño del sistema tiene asociada una línea de solicitud de interrupción. Estas líneas se conectan como entradas al manejador de interrupciones, el cual se encarga de procesarlas (organizar su jerarquía de atención) de acuerdo con su prioridad.

Al igual que los demás periféricos, la unidad controladora de interrupciones cuenta con elementos que le permiten realizar ciertas funciones de tolerancia a fallas.



LAMINA 111. MODULO SELECTOR



LAMINA 112. CONTROL INTERRUPTOR TOLERANTE A FALLAS

La unidad cuenta con dos manejadores programables de interrupciones PIC 8259A y un bloque de comparadores que vigila la igualdad de las salidas de los controladores de interrupciones cada vez que el procesador los accesa en operaciones de lectura (de estado de los PICs) o cuando el controlador de interrupciones coloca un vector sobre el bus de datos (que equivale a un proceso de lectura pero conducido por medio de la señal INTA#).

Cuando se presenta un error en las salidas de la unidad de control de interrupciones, este se indica al procesador activando la señal NMI que es la línea de solicitud de interrupción de mayor jerarquía.

En el momento en que se active la entrada NMI, el procesador entra en una rutina de servicio de interrupción que permita diagnosticar la falla u origen del error detectado y eventualmente proceder a aislar al periférico dañado.

10.1. FUENTES DE INTERRUPCION

De acuerdo con su uso, las fuentes de interrupción instaladas en este sistema pueden clasificarse en dos grupos:

a. Interrupciones de apoyo a las funciones de tolerancia a fallas

b. Interrupciones de servicio a los dispositivos periféricos

10.1.1. INTERRUPCIONES DE APOYO A LAS FUNCIONES DE TOLERANCIA A FALLAS

A este grupo pertenecen las interrupciones generadas por los circuitos de detección de error y las utilizadas para solicitar la reincorporación de procesadores al sistema.

Las fuentes de interrupciones empleadas para indicar detección de error están triplicadas para el caso de los circuitos de verificación de paridad en la memoria local de los procesadores y duplicadas para el caso de los circuitos de votación y los de comparación de salidas de periféricos.

ORIGEN	SIGNIFICADO
INT0	Error paridad procesador 0
INT1	Error paridad procesador 1
INT2	Error paridad procesador 2
INT3	Falla detectada por circuito votacion 0
INT4	Falla detectada por circuito votacion 1
INT5	Diferenc. detectada compar. serializ. 0
INT6	Diferenc. detectada compar. serializ. 1
INT7	Solicitud servicio serializador princ.
INT8	Diferenc. detectada compar. R.T.C. 0
INT9	Diferenc. detectada compar. R.T.C. 1
INT10	Solicitud servicio reloj. t.real princ.
INT11	Solicitud reincorporacion procesador 0
INT12	Solicitud reincorporacion procesador 1
INT13	Solicitud reincorporacion procesador 2
NMI	Diferenc. detectada comparadores de PIC

Tabla 10.1 Fuentes de interrupción del sistema

Si dos o más circuitos de detección de error activan simultáneamente una solicitud de interrupción, se atiende primero (o exclusivamente) la solicitud de mayor jerarquía. El orden en que sean atendidas las interrupciones debe definirse en la la programación del sistema operativo.

Las solicitudes atendidas por el controlador de interrupciones, originadas por detección de error, son generadas por los circuitos de validación de paridad (INT[0..2]), los circuitos de votación (INT[3..4]), los comparadores de serializadores (INT[5..6]) y los comparadores de relojes de tiempo real (INT[8..9]).

Para indicar la presencia de errores en las salidas del manejador de interrupciones, la salida de interrupción del comparador correspondiente se conecta a la entrada NMI de los procesadores.

En vista de que la eficacia de los mecanismos de tolerancia a fallas incorporados al "hardware" del sistema se basa esencialmente en la confiabilidad de la unidad controladora de interrupciones, se ha asignado la máxima prioridad a la indicación de error en el manejador de interrupciones.

Para que un circuito de detección de error pueda evaluar los datos recibidos como entradas y generar pedidos de interrupción, previamente le debe asignársele, por programación, el estado activo.

La función de reincorporación al sistema puede ser invocada por los procesadores que hayan sido excluidos de los procesos de votación escribiendo un "uno" sobre el bit asociado en el registro de interrupción de procesadores. Los bits de este registro actúan como las fuentes de interrupción INT[11..13].

10.1.2 INTERRUPCIONES DE SERVICIO A LOS DISPOSITIVOS PERIFERICOS

En este grupo se incluyen la línea INT[7] utilizada por el canal serie para guiar los procesos de recepción y transmisión de información y la línea INT[10] utilizada por el reloj de tiempo real para generar señales de alarma cuando se cumpla alguna de las condiciones definidas por programación.

A diferencia de las solicitudes de interrupción asociadas a los circuitos empleados para implantar los atributos de tolerancia a fallas, las cuales están duplicadas, existe una sola línea por cada periférico para solicitar la atención del procesador mediante interrupción. La razón de esto es que cada periférico es visto por el procesador como una unidad; desde esa perspectiva la redundancia incorporada en el periférico es transparente para el procesador

Aunque los dos dispositivos iguales con los que cuenta cada periférico generan simultáneamente la señal de solicitud de servicio, solo el dispositivo al que le haya sido asignada la categoría principal puede activar la línea conectada como entrada al controlador de interrupciones.

La lista de todas las posibles causas de interrupción contempladas en esta versión del sistema se indican en la tabla 10.1.

10.2. SISTEMA DE INTERRUPCIONES EN LA FAMILIA 86 DE INTEL

Los microprocesadores de la familia 86 de Intel cuentan con un sistema de interrupciones relativamente simple y versátil. En estos procesadores, las interrupciones pueden ser generadas por los dispositivos externos al CPU, por programación ("por software") o, bajo ciertas condiciones, por el CPU mismo.

En el contexto de sistemas basados en microprocesadores, una interrupción es una suspensión o discontinuidad en el flujo de ejecución de instrucciones, en respuesta a un estado activo en una de las entradas de solicitud de interrupción (INTR o NMI).

Al recibir una solicitud de interrupción, el procesador realiza dos diferentes acciones:

i. Procesa la interrupción: salva el estado del procesador en la pila y, si requiere conmutar de ambiente, salva la información de tarea.

ii. Da servicio a la interrupción, esto es, transfiere la ejecución del programa a una de las 256 posibles rutinas de interrupción.

A cada interrupción se le asigna un código (índice o vector) que la identifica ante el procesador. El CPU utiliza el índice para apuntar a una localidad en la tabla de vectores de interrupción almacenados en memoria. La tabla de vectores de interrupción puede contener hasta 256 vectores para diferentes tipos de interrupciones.

10.3. MANEJO DE INTERRUPCIONES EXTERNAS EN EL 80386

El 80386 tiene dos terminales de entrada que pueden usar los dispositivos externos para solicitar interrupciones: INTR (interrupt request) y NMI (non maskarable interrupt).

Típicamente, en un sistema con múltiples interrupciones externas, basado en microprocesador, la línea INTR es manejada por una unidad de manejo de interrupciones (tal como un dispositivo 8259A), a la cual se conectan como entradas las fuentes de interrupción.

Dicha unidad procesa las solicitudes de acuerdo con su prioridad y las envía a la entrada INTR del 80386, una a la vez.

Para que el sistema atienda interrupciones, debe previamente haberse inicializado el área de vectores de interrupción con las direcciones iniciales de las rutinas de interrupción.

Para accesar una rutina particular de interrupción, el 80386 debe obtener un vector (o índice) que apunte a la localidad de la tabla que contiene la dirección de la subrutina correspondiente.

La fuente de este vector depende del tipo de interrupción. Si la interrupción es mascarable (entrada INTR activa), el vector lo suministra la unidad de manejo de interrupciones (8259A); si la interrupción es no mascarable (entrada NMI activa), se apunta automáticamente la localidad 2 de la IDT (Interrupt Descriptor Table).

La interrupción INTR tiene asociada en memoria un área que empieza en 00H, en la que se encuentran los vectores que contienen las direcciones iniciales de las rutinas de atención de interrupción asociadas.

La solocitud NMI y la INTR difieren en que el procesador 80386 puede programarse para ignorar solicitudes INTR (borrando la bandera de interrupción del 80386), mientras que una solicitud NMI siempre provoca una respuesta por parte del procesador a menos que éste se encuentre dando servicio a una solicitud NMI anterior.

Además, la solicitud INTR ocasiona que el 80386 ejecute dos ciclos de reconocimiento de interrupción para obtener el vector de la rutina de servicio. Estos dos ciclos no se requieren para una solicitud NMI debido a que la posición de su vector es fija. Si una solicitud MNI y una INTR llegan simultáneamente al 80386, la NMI se procesa primero.

En el computador aquí presentado, se utiliza la entrada NMI para indicar la ocurrencia de una falla a nivel del manejador de interrupciones (salidas distintas en los buses de datos de la unidad principal y la redundante).

10.4. UNIDAD DE CONTROL DE INTERRUPCIONES (8259A)

Uno de los elementos fundamentales del sistema presentado es el manejador de interrupciones. Esta unidad se comporta como un dispositivo periférico cuya función consiste en administrar las solicitudes de interrupción de las tareas del procesador.

El computador tolerante a fallas propuesto cuenta con dos unidades de manejo de interrupciones. Bajo condiciones normales una de ellas actúa con categoría de controlador principal y la otra en categoría de respaldo.

El 8259A es un dispositivo especialmente diseñado para uso en en sistemas de microcomputadores conducidos por interrupciones. Cada circuito 8259A maneja hasta ocho niveles o solicitudes. Presenta características incorporadas que posibilitan la conexión en cascada de varios componentes 8259A sin requerir circuitería adicional, lo que permite expander la capacidad de atención hasta 64 niveles o fuentes de interrupción.

Durante los procesos de lectura - escritura del procesador, el manejador de interrupciones 8259A se accesa como un par de localidades en el espacio de 64K puertos. Durante tales procesos, recibe como entradas los comandos RD# y WR# para leer o escribir información en él. En este sistema, la unidad de control de interrupciones está mapeada en las direcciones 300 H y a 30C H.

Las características de operación del 8259A pueden ser definidas por programación. El programador del sistema tiene posibilidad de establecer un modo de selección de prioridad, de manera que la secuencia en que se procesan las diferentes solicitudes de interrupción, puede ajustarse a los requerimientos del sistema.

La jerarquía de atención puede cambiarse dinámicamente en cualquier momento durante la operación del sistema. En consecuencia, la estructura completa de interrupciones puede redefinirse según se requiera.

El 8259A está diseñado para minimizar el costo de programación y el tiempo de ejecución requerido para el manejo de interrupciones priorizadas multinivel.

Las solicitudes de interrupción de las diferentes fuentes previstas en el sistema se conectan a las entradas de interrupción del 8259A. Además, este circuito se comunica con el procesador por medio del bus de datos.

10.5. UNIDAD CONTROLADORA DE INTERRUPCIONES DEL SISTEMA

Esta versión del sistema cuentacon 14 distintas fuentes interrupción (INT[0..13]) atendidas por el controlador de interrupciones.

Cada unidad manejadora de interrupciones está formada por dos dispositivos 8259A conectados en cascada, uno de los cuales actúa como maestro y el otro como esclavo.

En este tipo de configuración, las tres líneas CAS0 .. CAS2 se comportan como un bus privado que permite controlar la estructura de la unidad de manejo de interrupciones. Las líneas del bus de cascada permiten seleccionar los dispositivos esclavos durante la secuencia de reconocimiento de interrupción.

En el 8259A maestro, las líneas de cascada actúan como salidas que sirven para direccionar los dispositivos esclavos; en el 8259A esclavo, dichas líneas se comportan como entradas para recibir la dirección proveniente del maestro.

En este esquema, si un dispositivo periférico requiere servicio, solicita la atención del manejador esclavo al que se encuentre conectado. Este a su vez solicita interrupción al 8259A maestro, el cual se encarga finalmente, de gestionar la solicitud ante el procesador. Similarmente, durante el proceso de respuesta, el procesador responde al maestro, éste al esclavo y éste al dispositivo externo.

Para que durante el proceso de reconocimiento de interrupción, se direccione el 8259 apropiado y éste coloque el vector de interrupciones sobre el bus de datos sin causar contención, cada manejador 8259A debe programarse con el modo de operación adecuado (maestro o esclavo).

Cada manejador 8259A debe seguir una secuencia separada de incialización y puede programarse para operar de diferente modo. La programación en modo cascada se realiza durante la inicialización de cada 8259A. El dispositivo programado como maestro debe recibir información, durante la fase de inicio, acerca de cuáles de sus entradas IR están conectadas a salidas INT de un esclavo.

A cada esclavo se le debe programar la dirección de la entrada IR del maestro que ocupa. Esto es necesario para que las líneas CAS0..CAS2 puedan identificar a cada esclavo individual.

10.6. CIRCUITO DE LA UNIDAD DE CONTROL DE INTERRUPCIONES

Referencia: láminas 113 a 119.

10.6.1. BUS DE DATOS

La unidad de control de interrupciones se comunica con el procesador por medio del bus de datos. Esta ruta la utiliza para recibir información de inicialización o comandos procedentes del procesador y para transmitir información de estado y vectores de interrupción hacia el CPU.

Dado que cada vez que el manejador de interrupciones da curso a una solicitud debe enviar un vector hacia el procesador, se cuenta con un módulo de comparadores que analiza las salidas de las unidades manejadoras hacia el bus de datos. Si durante el proceso de comparación se detecta una disparidad, el comparador correspondiente genera una solicitud de tipo NMI.

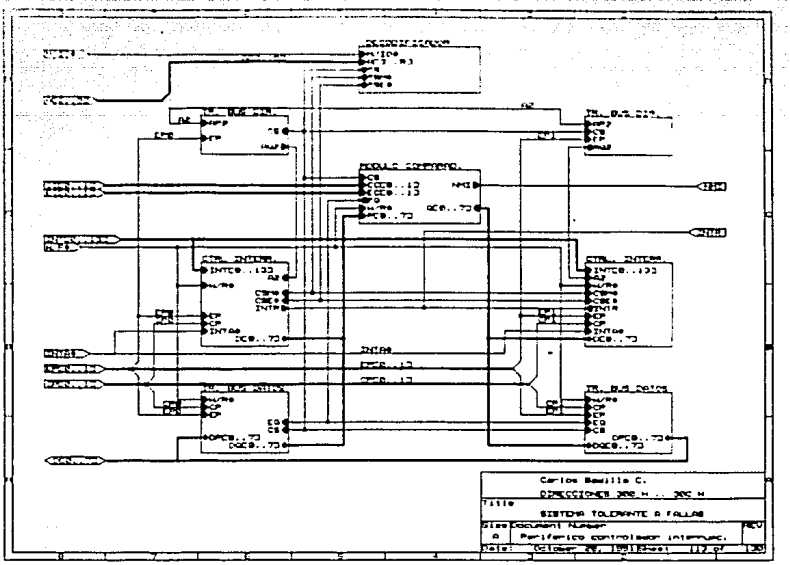
10.6.2. LOGICA DE LECTURA / ESCRITURA

Las entradas RD# y WR# del 8259A se utilizan para identificar los comandos de lectura y escritura emitidos por el procesador. En vista de que el microprocesador utiliza una sola línea para activar ambos comandos, la entrada RD# del 8259A se conecta directamente a la línea W/R# del procesador 80386; la entrada WR# del 8259A se conecta a esa misma salida del 80386 por medio de un negador.

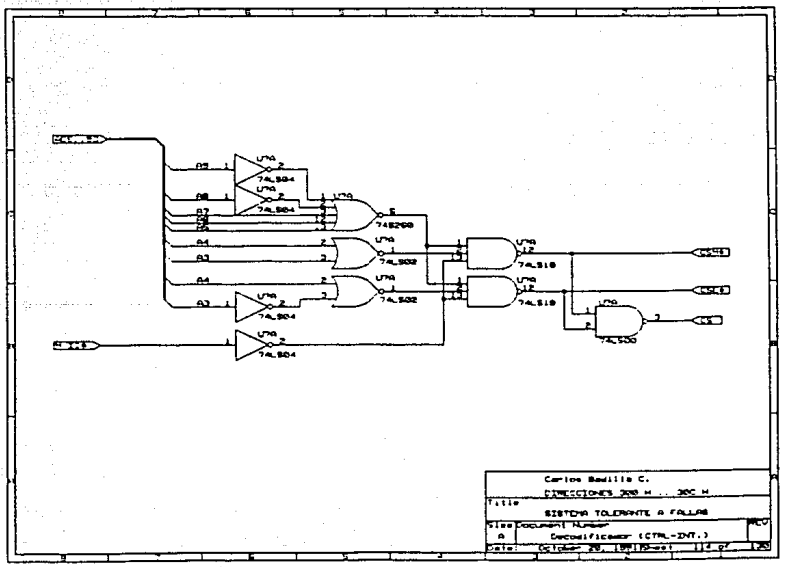
10.6.3. HABILITACION DEL CONTROLADOR

Cada unidad controladora de interrupciones consta de dos dispositivos 8259A. El primer de ellos actúa como maestro y está mapeado en la direcciones 300H y 304H; el segundo se comporta como esclavo y utiliza las direcciones 308H y 30CH.

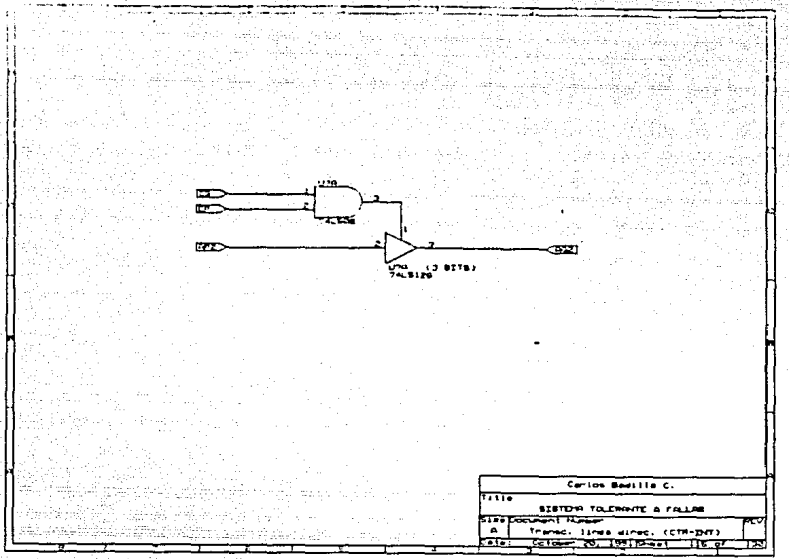
Las dos unidades controladoras de interrupciones (una principal y la otra de respaldo) están mapeadas en las mismas direcciones del espacio de puertos.



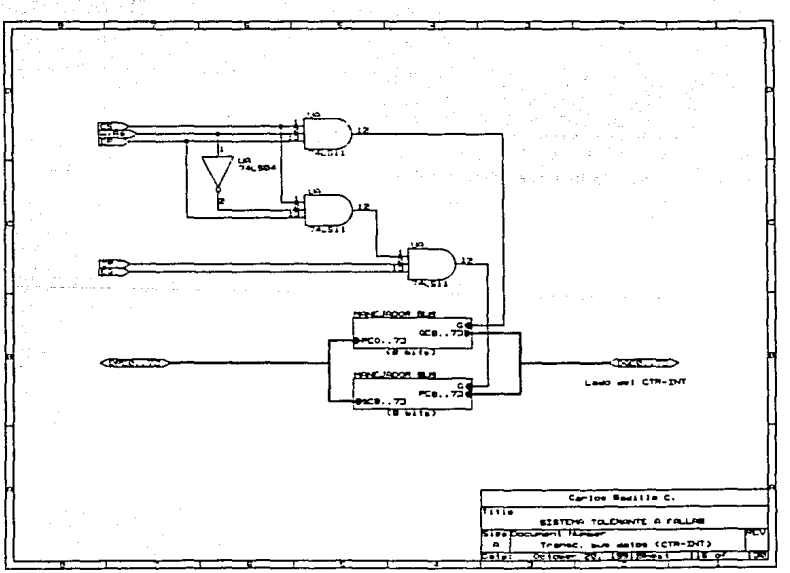
LAMINA 113. PERIFERICO CONTROLADOR DE INTERRUPCIONES



LAMINA 114. DECODIFICADOR



LAMINA 115. TRANSCEPTOR DE LA LINEA DE DIRECCION



LAMINA 116. TRANSCEPTOR DE BUS DE DATOS

Cuando el procesador envía información hacia el controlador de interrupciones, se actualizan los registros internos de ambas unidades; durante los procesos de lectura, solo una de ellas (la que está en categoría principal) coloca información sobre el bus de datos.

Para seleccionar los manejadores de interrupción se utilizan las líneas A2 a A9 del bus de direcciones. El circuito de decodificación genera las líneas CSM# (para seleccionar al maestro) y CSE# (para escoger al esclavo). La línea A2 se emplea para gobernar directamente la entrada A0 de ambos 8259A, la cual discrimina los correspondientes registros internos.

10.6.4. IDENTIFICACION DE MAESTRO Y ESCLAVO

Para identificar el modo de operación de cada dispositivo 8259A, se emplea la línea SP#/ES# . Esta línea se conecta a VCC para identificar al 8259A que presenta modo de operación maestro. Dicha entrada se conecta a referencia (GND) para identificar al manejador esclavo.

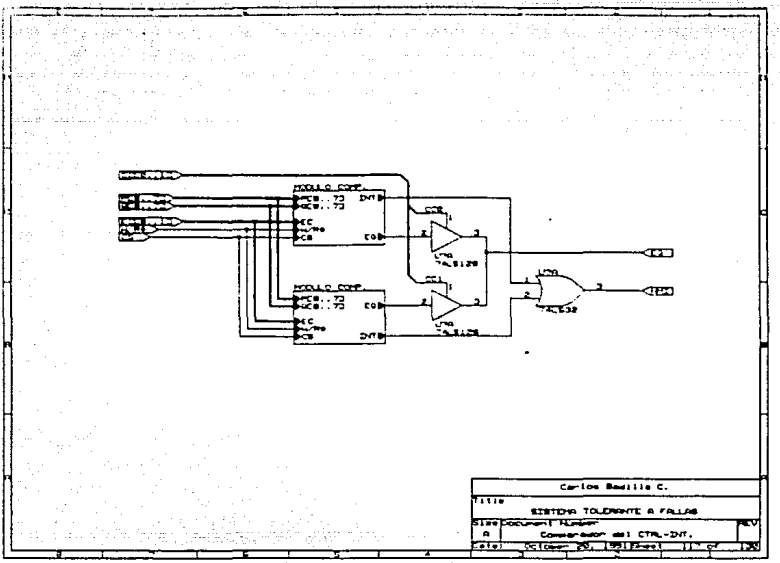
10.6.5. INTERCONEXIONES ENTRE MAESTRO Y ESCLAVO

Las líneas CAS0..CAS2 del maestro actúan como salidas y están conectadas a las líneas del mismo nombre del esclavo. Este bus se emplea para el direccionamiento del manejador esclavo durante los ciclos de reconocimiento de interrución.

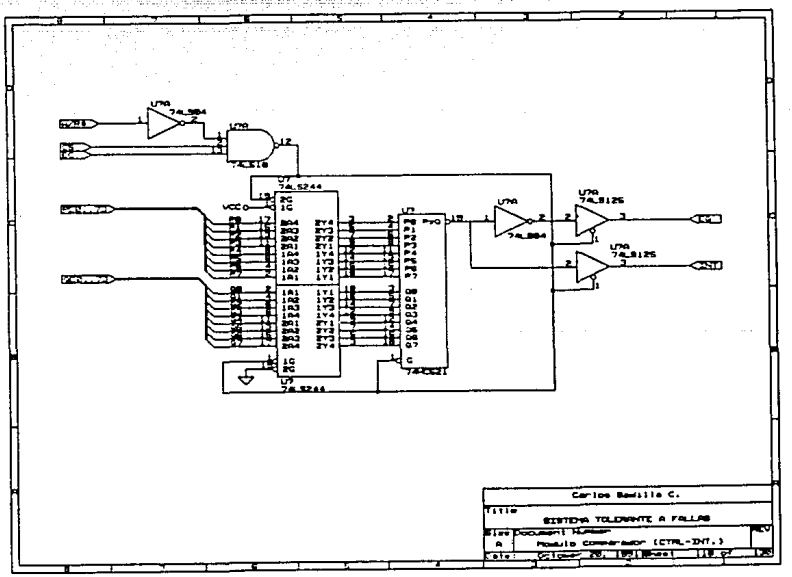
La salida INT del esclavo es conectada a la entrada IR7 del maestro (para el maestro actúa como fuente de interrupción).

10.6.6. CONEXION DE LA LINEA DE RECONOCIMIENTO DE INTERRUPCION

La señal INTA# procedente del procesador llega a ambos 8259A (tanto maestro como esclavo). Durante el primer ciclo, la entrada INTA# solicita al maestro que coloque la dirección del esclavo sobre el bus interno CAS. Durante el segundo ciclo autoriza al manejador encargado de la interrupción para que coloque el correspondiente vector sobre el bus de datos.



LAMINA 117. COMPARADOR DEL CTRL-INT



Las entradas restantes IR, tanto del maestro como del esclavo, reciben las líneas de solicitud de interrupción del sistema según se indica en la siguiente tabla:

M. Int.	ORIGEN	SIGNIFICADO
M/IR0	INT0	Error paridad procesador 0
M/IR1	INT1	Error paridad procesador 1
M/IR2	INT2	Error paridad procesador 2
M/IR3	INT3	Falla detectada por circuito votacion 0
M/IR4	INT4	Falla detectada por circuito votacion 1
M/IR5	INT5	Diferenc. detectada compar. serializ. 0
M/IR6	INT6	Diferenc. detectada compar. serializ. 1
M/IR7	INT/E	Salida interrupcion 8259A esclavo
E/IR0	INT7	Solicitud servicio serializador princ.
E/IR1	INT8	Diferenc. detectada compar. R.T.C. 0
E/IR2	INT9	Diferenc. detectada compar. R.T.C. 1
E/IR3	INT10	Solicitud servicio reloj. t.real princ.
E/IR4	INT11	Solicitud reincorporacion procesador 0
E/IR5	INT12	Solicitud reincorporacion procesador 1
E/IR6	INT13	Solicitud reincorporacion procesador 2

Tabla 10.2 ASIGNACION DE FUENTES DE INTERRUPCION

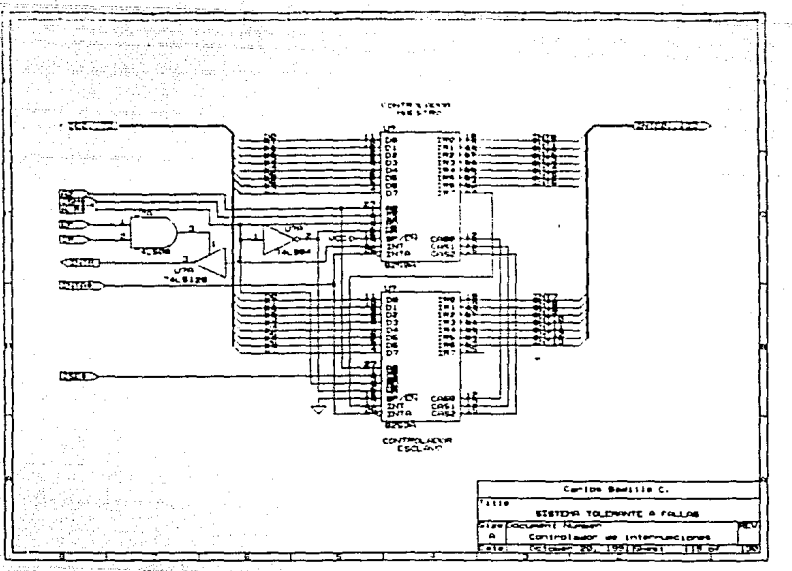
Notación:

M/IRi : i-ésima entrada IR del 8259A maestro; i= 1..7

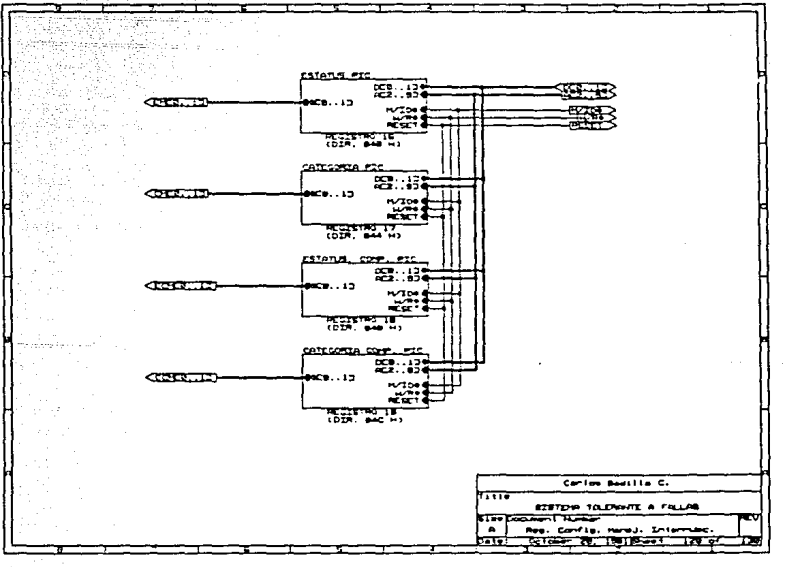
E/IRj : j-ésima entrada IR del 8259A esclavo; j= 1..6

INTk : k-ésima línea del bus de fuentes de interrupción; k = 0..13

INT/E : Salida INT del 8259A esclavo



LAMINA 119. CONTROLADOR DE INTERRUPCIONES



LAMINA 120. REGISTRO DE CONFIGURACION DE MANEJADORES INTERRUPCIONES

10.7. MODULO DE REGISTROS DE CONFIGURACION DE CONTROLADORES DE INTERRUPCIONES

(Referencia: lámina 120)

La operación del módulo controlador de interrupciones tolerante a fallas se basa en el uso de cuatro registros de configuración cuyo contenido original se define durante la etapa de iniciación (reset) del sistema. El procesador puede accesar estos registros ya sea para operaciones de lectura o de escritura.

Las salidas de los registros de configuración de los controladores de interrupciones son de tipo latch. Esto permite tener disponible, en forma permanente, un conjunto de líneas independientes cuyos estados lógicos se emplean para definir las características de operación (estado y categoría) en que trabajan las unidades controladoras de interrupciones.

10.7.1. REGISTRO DE ESTADO DE CONTROLADORES DE INTERRUPCION

Referencia: lámina 121.

Registro número 16. Ocupa la localidad 40 H en el espacio de puertos del sistema. (lámina 122)

Tiene 2 bits de ancho, cada uno de ellos asociado a una unidad controladora de interrupciones.

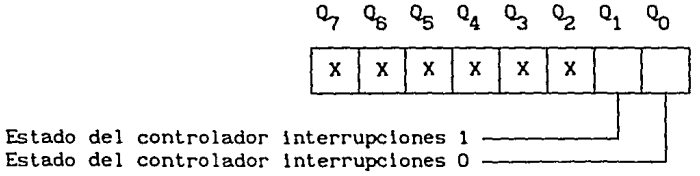
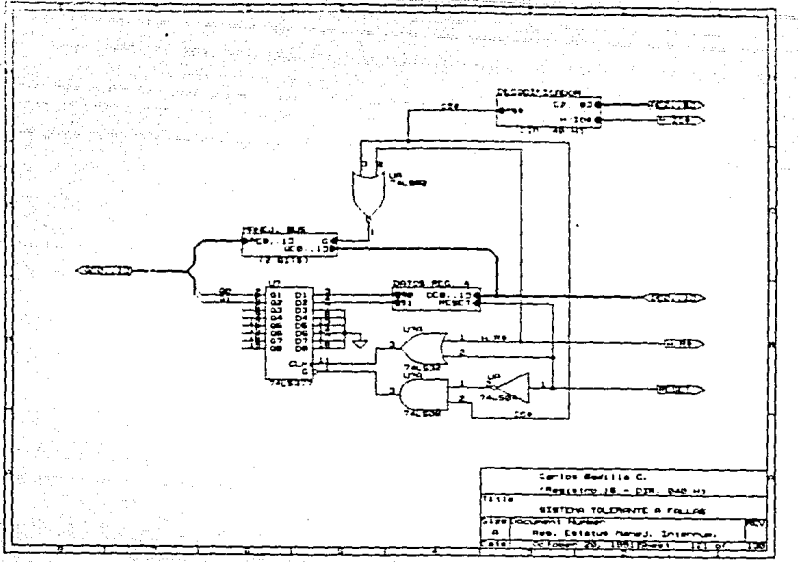
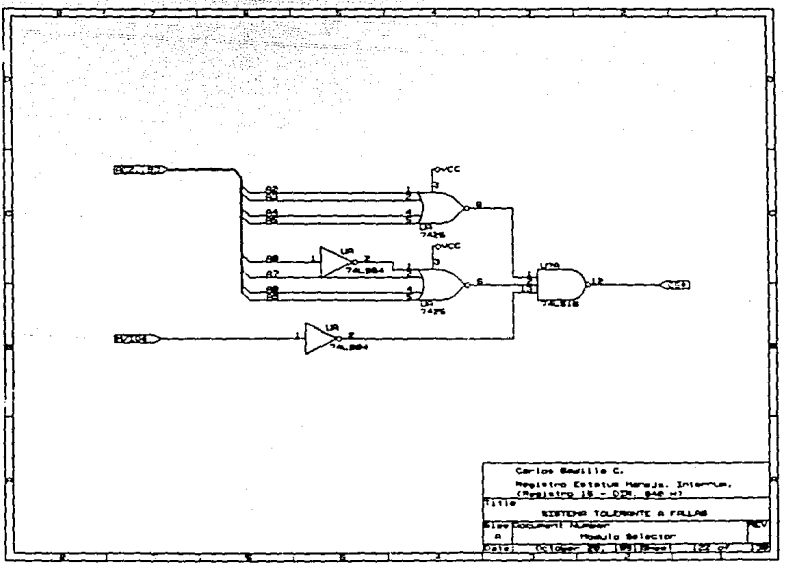


Figura 10.1 Registro de estado de controladores de interrupciones

Cada bit representa uno de dos posibles estados de operación que pueden asignarse a una unidad controladora de interrupciones: activo / inactivo#.



LAMINA 121. REGISTRO ESTATUS DEL MANEJADOR DE INTERRUPCIONES



LAMINA 122. MODULO SELECTOR

Un módulo controlador de interrupciones al que se le asigne el estado activo, puede recibir solicitudes de interrupción e iniciar su procesamiento; esto es, activar la línea INTR para gestionar el inicio del servicio de interrupción ante el procesador

Inicialmente a ambos controladores se les asigna el estado activo. (lámina 123).

10.7.2. REGISTRO DE CATEGORIA DE CONTROLADORES DE INTERRUPCIONES

Referencia: lámina 124.

Registro número 17. Ocupa la localidad 44 H en el espacio de puertos del sistema. (lámina 125).

Tiene un ancho de dos bits, cada uno de ellos asociado a una unidad controladora de interrupciones.

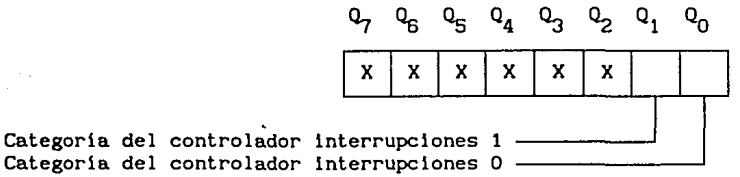
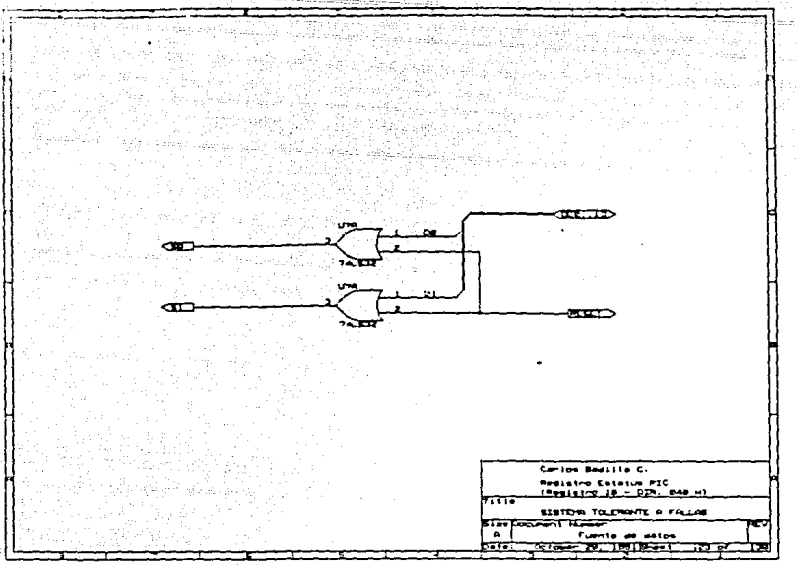


Figura 10.2 Registro de categoría de controladores de interrupciones

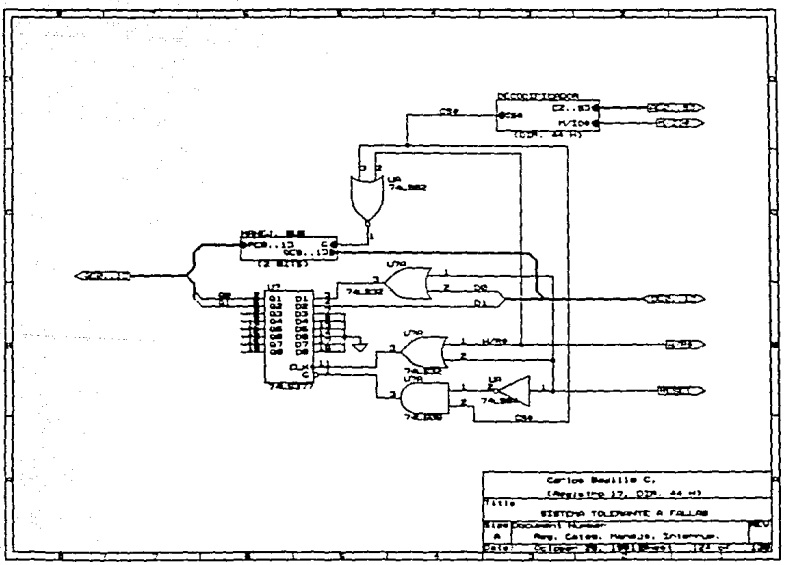
Cada bit representa una de dos posibles categorías que pueden asignarse a una unidad controladora de interrupciones: principal / respaldo#.

El módulo controlador de interrupciones al que se le asigne categoría principal puede enviar información de estado al procesador y, además, completar el procesamiento de las solicitudes de interrupción (colocar el vector de interrupción sobre el bus de datos).

Inicialmente al controlador de interrupciones número cero (0) se le asigna la categoría principal y al número uno (1) la categoría de respaldo#. (lámina 124).



LAMINA 123. FUENTE DE DATOS



LAMINA 124. REGISTRO DE CATEGORIA DE LOS MANEJADORES DE INTERRUPCIONES

Si un controlador de interrupciones se encuentra en estado activo y tiene categoría de respaldo#, dicho controlador puede generar la señal INT de solicitud de interrupción. Sin embargo, no puede colocar vector sobre bus de datos.

Si ambos controladores de interrupción generan un mismo vector de interrupción (detectan una solicitud proveniente de la misma fuente), el controlador con categoría principal es el único que puede colocar el vector mencionado sobre el bus de datos a efecto de que el procesador pueda leerlo y proceda a dar el servicio correspondiente.

Sin embargo, si los vectores de interrupción generados por los controladores son diferentes, el módulo de comparadores identifica la situación anómala y genera una interrupción de tipo NMI.

Dado que la interrupción NMI tiene mayor jerarquía que las interrupciones INTR y causa siempre respuesta del procesador, los errores detectados en la unidad controladora de interrupciones se atienden antes que los que se originen en cualquier otra parte del sistema.

10.7.3. REGISTRO DE ESTADO DE COMPARADORES DE MANEJADORES DE INTERRUPCIONES

Referencia: lámina 126.

Registro número 18. Ocupa la dirección 48 H del espacio de puertos del sistema. (lámina 127)

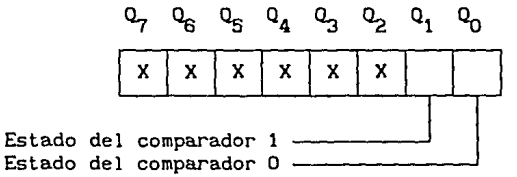
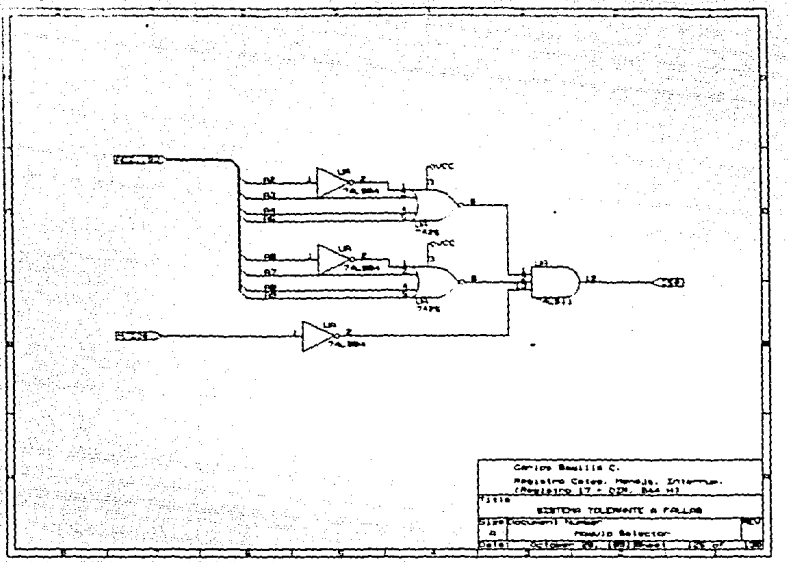
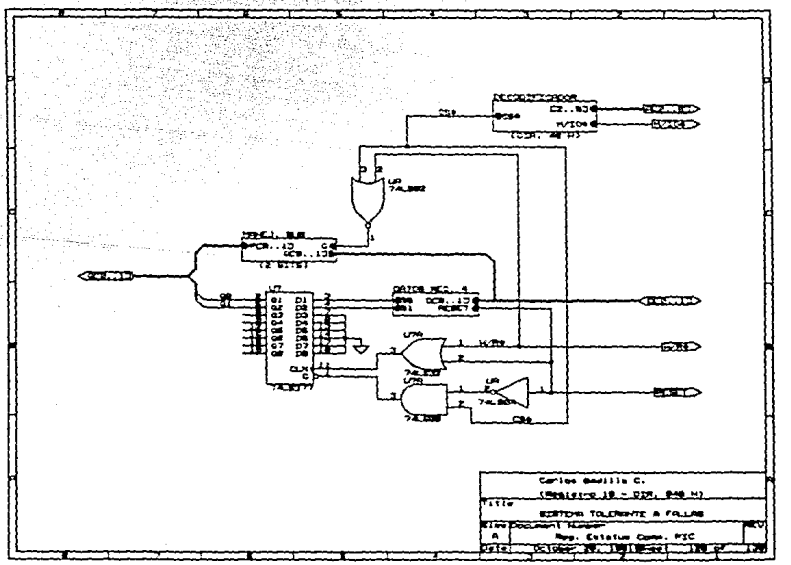


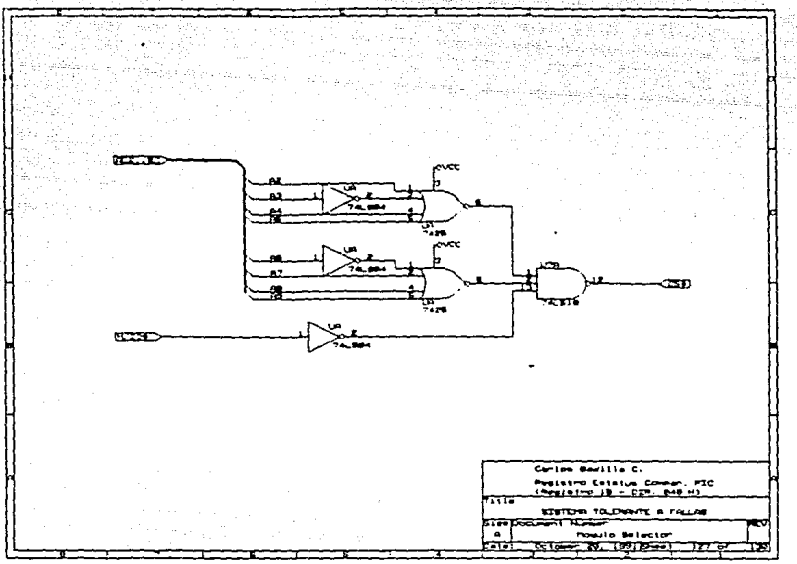
Figura 10.3 Registro de estado de comparadores de manejadores de interrupciones



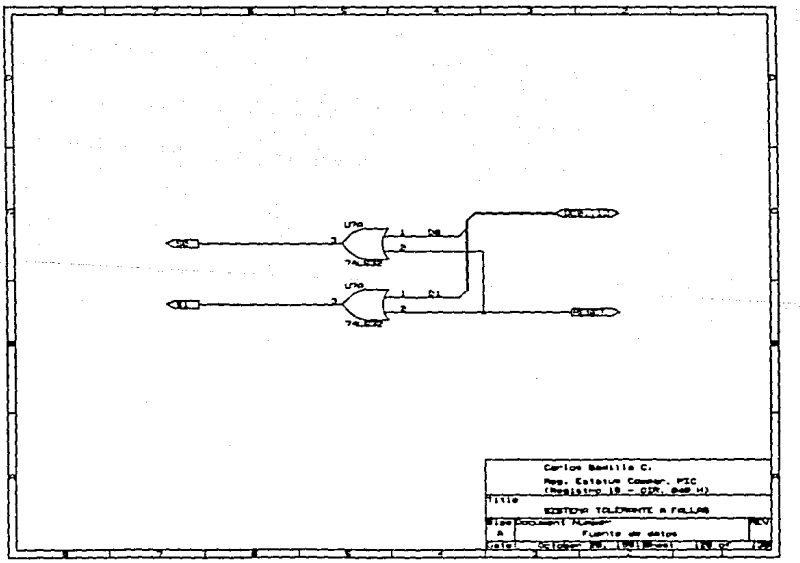
LAMINA 125. MODULO SELECTOR



LAMINA 126. REGISTRO ESTATUS DE COMPARADORES DE MANEJADORES DE INTERRUPCIONES



LAMINA 127. MODULO SELECTOR



LAMINA 128. FUENTE DE DATOS

Tiene un ancho de dos bits, cada uno de ellos asociado a uno de los comparadores de salida de controladores de interrupciones.

Cada bit de este registro representa dos posibles estados de operación que pueden asignarse a los comparadores: activo / inactivo#.

Un comparador al que se le asigne el estado activo puede evaluar los valores presentes en las salidas de los controladores de interrupciones.

Además, en caso de encontrar diferencias entre las salidas de los controladores de interrupciones, cada comparador puede generar una solicitud de interrupción para indicar la anomalía detectada.

La diferencia en las salidas de controladores de interrupción da origen a una solicitud interrupción tipo NMI. Esta es la interrupción de mayor jerarquía debido a que la operación correcta del sistema depende en alto grado de la operación correcta del susbsistema de manejo de interrupciones.

Inicialmente a ambos comparadores se les asigna el estado activo. (lámina 128)

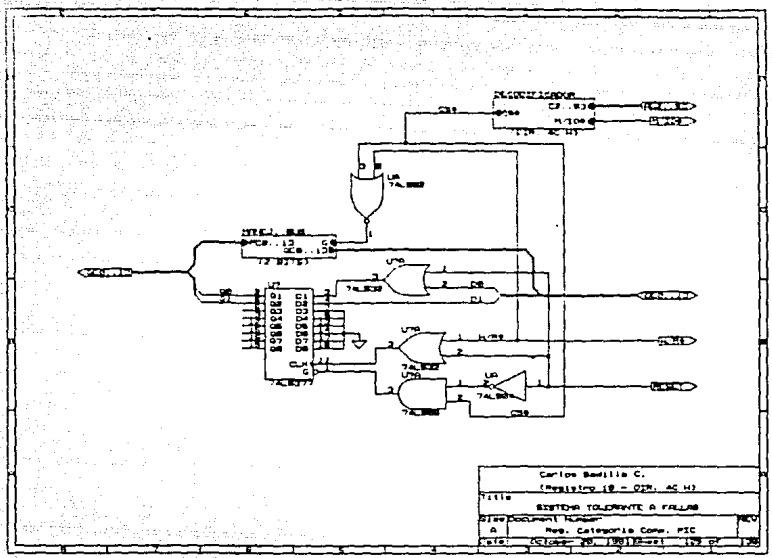
10.7.4. REGISTRO DE CATEGORIA DE COMPARADORES DE MANEJADORES DE INTERRUPCIONES

Referencia: lámina 129.

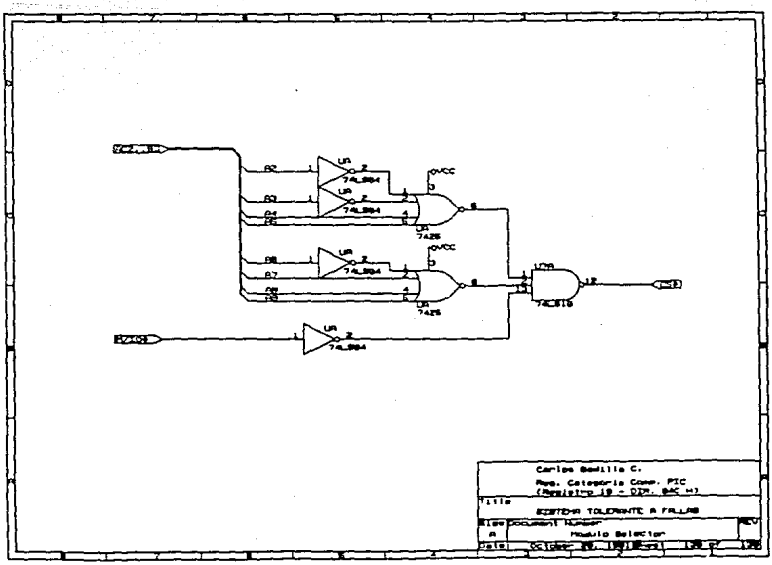
Registro número 19. Ocupa la dirección 4C H del espacio de puertos del sistema. (lámina 130)

Tiene un ancho de dos bits, cada uno de ellos asociado a uno de los comparadores de salida de controladores de interrupciones.

Cada bit de este registro representa una de dos posibles categorías que pueden asociarse a los comparadores: principal / respaldo#.



LAMINA 129. REGISTRO CATEGORIA DECOMPARADORES DE MANEJADORES DE INTERRUPCIONES



LAMINA 130. MODULO SELECTOR

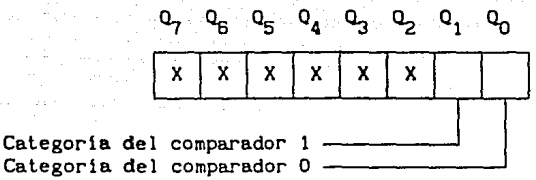


Figura 10.4 Registro de categoría de comparadores de manejadores de interrupciones

Las líneas de salida del registro de categoría de comparadores se emplean para habilitar o deshabilitar los circuitos de aislamiento conectados a las salidas de igualdad ("equality") de los comparadores; esto es, para identificar al comparador cuya salida de igualdad tiene efecto fuera del módulo de comparación (lámina 117).

La categoría principal identifica al comparador cuya salida de igualdad ("equality") se utiliza para gobernar los transceptores que unen el bus de datos del controlador de interrupciones con el bus del sistema (lámina 113); es decir, para definir si se autoriza al controlador con categoría principal para colocar información sobre el bus de datos.

Inicialmente al comparador cero (0) se le asigna la categoría principal y al uno (1) la de respaldo#. (lámina 129).

CAPITULO 11
TECNICAS UTILIZADAS PARA MEJORAR LA CONFIABILIDAD POR PROGRAMACION

INTRODUCCION

Para lograr el comportamiento confiable del sistema no solo es importante que éste cuente con recursos físicos redundantes que permitan reemplazar a los dispositivos que pudieran fallar, sino que además, la programación del sistema operativo contenga las rutinas necesarias para ejecutar las funciones de tolerancia a fallas (rutinas de diagnóstico de fallas, de reconfiguración de sistema).

Entre las técnicas que pueden emplearse para incrementar la tolerancia a fallas por programación se encuentran las siguientes [Gray 86]:

a. Programación modular basada en procesos y mensajes
b. Contención de fallas mediante módulos de respuesta rápida
c. Redundancia en programación

11.1. PROGRAMACION MODULAR BASADA EN PROCESOS Y MENSAJES

En forma similar al diseño electrónico, la clave para incrementar la confiabilidad mediante la programación consiste en descomponer jerárquicamente un sistema en un conjunto de módulos. Cada módulo debe dar servicio a un solo tipo de falla y debe estar construido de forma tal que evite la propagación de sus efectos más allá de la unidad en la cual se originó.

El esquema de respuesta a fallas basado en interrupciones facilita la programación modular pues se define una rutina (proceso) diferente de servicio de interrupción para cada falla que se presente.

De esta manera, el proceso (o subrutina) se convierte en la técnica principal de defensa ante fallas por programación.

Adicionalmente, para evitar la propagación de los efectos de fallas, las rutinas deben definirse de forma tal que no compartan información global con otros procesos, sino que la comunicación entre ellos se realice únicamente por medio de mensajes.

11.2. CONTENCION DE FALLAS MEDIANTE MODULOS DE RESPUESTA RAPIDA

Para que la contención de falla sea eficaz, el proceso que le da servicio de operar en forma rápida, esto es, debe localizar la unidad dañada y aislarla rápidamente.

Para lograr esto deben emplearse técnicas de programación defensiva en el sistema operativo, tales como la ejecución periódica de rutinas de diagnóstico de fallas.

Además, la programación de aplicaciones debe cuidar aspectos tales como verificación de entradas, resultados intermedios y límites de las estructuras de datos para detectar fallas no asociadas a la circuitería.

11.3. REDUNDANCIA EN PROGRAMACION

Al igual que en la electrónica, la redundancia (procesos extra) en la programación contribuye a mejorar la tolerancia a fallas en los sistemas computacionales. Típicamente solo se utilizan pares de procesos pues la posibilidad de falla en otras partes del sistema (electrónica o entrada de datos) es mayor.

Algunos métodos para diseñar pares de procesos son los siguientes:

11.3.1. PASO CERRADO (lockstep)

En este método, el proceso primario y el de respaldo ejecutan concurrentemente el mismo conjunto de instrucciones en procesadores independientes. Si uno de los procesadores falla, el otro continúa la computación.

Este mecanismo provee buena tolerancia a fallas electrónicas. No obstante es ineficaz ante ciertas fallas de la programación.

11.3.2. VERIFICACION PUNTUAL DE ESTADO

En este esquema se utilizan sesiones programadas de comunicación para comparar datos de prueba. Si el proceso primario falla, el control se transfiere al proceso de respaldo. Se utilizan números de secuencia de sesión para detectar mensajes duplicados y para controlar el reenvío de información.

Este método de programación provee buena tolerancia a fallas [Gray 86] pero presenta el inconveniente de que complica la programación e incrementa el tiempo requerido para el procesamiento.

11.3.3. VERIFICACION AUTOMATICA DE PUNTO

Este procedimiento es similar al anterior. Se diferencia en que la verificación se realiza automáticamente como función del sistema operativo y no como tarea del programador.

En este modelo, el sistema operativo resguarda todos los mensajes de entrada y salida del proceso principal y se utilizan para transferir el estado del principal al de respaldo hasta el punto de verificación inmediato anterior a la ocurrencia de una falla.

Este esquema requiere manejar mayor cantidad de información que la verificación programada por el usuario, por lo que su costo en tiempo de ejecución y requerimiento de memoria es mayor.

CONCLUSIONES

1. Tal como se indica en la introducción de este trabajo, no es posible construir un sistema computacional 100% confiable: siempre existe la posibilidad de que surja una falla no prevista, para la cual el sistema carezca de una respuesta predefinida.

Sin embargo, estadísticas realizadas [Gray 86] revelan que los sistemas tolerantes a fallas presentan un tiempo medio entre fallas de más de 500 veces mayor que el reportado para computadores convencionales.

Como consecuencia puede decirse que, la adecuada incorporación de atributos y técnicas de tolerancia a fallas en un computador, constituye una estrategia eficaz para incrementar la confiabilidad de un sistema computacional.

2. El diseño de este computador ha sido elaborado utilizando exclusivamente componentes electrónicos de propósito general.

Esto pone de manifiesto la viabilidad de incursionar en el campo del diseño electrónico (en este caso la arquitectura de computadores) para dar solución a necesidades concretas de nuestros países.

3. El diseño de este computador se ha desarrollado siguiendo los principios del diseño medular de sistemas.

Esta metodología de diseño conviene conservarla para introducir variantes en las siguientes versiones. La construcción modular es uno de los aspectos clave para alcanzar el comportamiento tolerante a fallas ya que presenta la ventaja de que permite confinar los efectos de las fallas al sector del circuito en que se originan.

4. Tal como se mencionó anteriormente, la tolerancia a fallas involucra todos los aspectos estructurales y funcionales del computador.

En consecuencia debe tenerse en cuenta que, el grado en el que se eleve realmente la confiabilidad del sistema depende, no solo de los elementos físicos redundantes agregados a la arquitectura (procesadores, periféricos, registros de reconfiguración, circuitos de votación, etc.), sino también del diseño de la programación (sistema operativo, rutinas de servicio de interrupciones, algoritmos de diagnóstico de fallas y recuperación del sistema, y aplicaciones de usuario).

5. Pese a que la confiabilidad es un aspecto del diseño electrónico al que se da cada vez mayor importancia, el mercado electrónico de aplicaciones comerciales no cuenta con circuitos integrados de propósito general especialmente concebidos para implantar funciones de tolerancia a fallas, tales como circuitos de votación, verificación y corrección de errores, periféricos con registros de reconfiguración incorporados.

Por esta razón, los circuitos utilizados para este fin se han especificado utilizando componentes digitales integrados que realizan funciones generales, tales como compuertas lógicas, comparadores, decodificadores, etc.

Durante la etapa de construcción convendría analizar la posibilidad de implantar tales circuitos mediante arreglos lógicos programables. Esta modalidad de implantación, además de simplificar el montaje del circuito, incrementaría la confiabilidad del sistema ya que disminuye los puntos de contacto físico, que suelen constituir una fuente importante de fallas.

6. Otra técnica que conviene considerar es el diseño de uno o varios circuitos integrados especialmente proyectados para realizar estas funciones.

Cabe mencionar que, el diseño de circuitos integrados, es un campo en el que se han realizado algunos avances en México donde ya fue construido el circuito integrado BRAMEX (microprocesador de propósito general de 8 bits) gracias a un convenio en el que participan Brasil, México y España.

7. Aunque el diseño de este computador podría realizarse usando componentes electrónicos más simples, se antepuso el criterio de actualidad tecnológica sobre el de simplicidad. Por este motivo, en este computador se usan circuitos integrados tales el microprocesador MC 80386, el transmisor receptor de datos en serie NS16550A, el reloj de tiempo real DS1287, etc.

El objetivo de esta decisión consiste en evitar que el computador diseñado presente un rezago tecnológico considerable ya desde el momento mismo de su concepción.

8. El diseño de este computador prevé el uso de un espacio de memoria amplio (8 Mega Bytes en memoria RAM estática y 256 Kilo Bytes de memoria EPROM). Esta previsión posibilita que dicho computador pueda utilizarse en una amplia gama de aplicaciones tales como procesamiento de imágenes a bordo o servicios de correo electrónico internacional.

9. El objetivo de lograr un alto grado de confinamiento de fallas obliga al desarrollo circuitería independiente para realizar el acceso selectivo a los diferentes elementos del sistema.

Este ha sido el criterio usado para especificar circuitos de decodificación separados para cada cada uno de los dispositivos periféricos, tanto para los utilizados en tareas generales como para los empleados en tareas de apoyo a las funciones de tolerancia a fallas.

10. La introducción de características de tolerancia a fallas en un computador es una labor que debe realizarse desde las primeras etapas del diseño y, en la que, las decisiones que se tomen en esa fase, condicionan en alto grado las características finales del sistema computacional.

Por este motivo, el diseño completo de un sistema computacional tolerante a fallas debería ser el resultado de la estrecha colaboración de tres grupos de especialistas: los responsables de el desempeño de la arquitectura, los diseñadores del sistema operativo y los encargados de introducir los atributos de tolerancia a fallas [Aviz 87].

De esto depende an alto grado que el producto resultante constituya una solución balanceada que se ajuste, tanto a los requerimientos de desempeño, como a los de operación confiable.

APENDICE A
CONCEPTOS FUNDAMENTALES RELACIONADOS CON ARQUITECTURAS TOLERANTES A FALLAS

INTRODUCCION

En esta sección contiene una descripción de los principales conceptos relacionados con el área de las arquitecturas de computadores tolerantes a fallas. Se incluye con la finalidad de proporcionar el marco conceptual básico asociado con el tema de la computación confiable.

1. NOMENCLATURA UTILIZADA PARA DESCRIBIR LA OPERACION DE SISTEMAS TOLERANTES A FALLAS

1.1 SERVICIO PROPIO (o esperado)

Secuencia de estados de salida que corresponden con las especificaciones de servicio que son definidas por el usuario. El servicio impropio se presenta cuando el servicio entregado es diferente del servicio especificado. La vida de un sistema es percibida por sus usuarios como la oscilación entre estos dos estados de servicio [Aviz 90].

Los eventos que constituyen las transiciones entre estos dos estados son la avería del sistema y la restauración del servicio.

1.2. ESTADO CRITICO

Información completa (datos + programas) que debe ser preservada bajo todas las condiciones de falla para las cuales el sistema tenga una respuesta de tolerancia incorporada. El estado crítico puede perderse debido a retrasos de duración inaceptable, recarga o recomputación. En tal caso se dice que se ha ocasionado una avería.

1.3 ERROR

Estado impropio de un recurso, ya sea en el límite o en un punto interno del recurso.

Un error (o estado impropio) se conoce como avería si se manifiesta en el límite o periferia del sistema o recurso.

El error puede ser latente o detectado y se origina por una falla activa.

El estado impropio asociado a un error es contrario a la especificación del recurso o a la expectativa (o requerimiento) del usuario.

Ejemplos de errores:

a) Error de paridad: Activación de la señal de paridad par durante la operación de lectura en una memoria que contiene únicamente palabras de paridad impar.

b) Error de comparación: Obtención de resultados diferentes de unidades aritméticas que ejecutan simultáneamente la misma operación sobre operandos iguales.

c) Error de temporización: Generación de una determinada señal fuera del intervalo de tiempo especificado.

d) Error de programación: Aceptación de un dato cuyo valor se encuentra fuera del intervalo de variación de determinados operandos o entradas.

1.4 FALLA

Causa identificada o hipotética de un error o de una avería. Una falla puede ser activa o durmiente. La falla es activa cuando produce un error. Una falla puede oscilar entre los estados activo y durmiente. La existencia de una falla se detecta cuando ocurre una avería en el sistema o cuando se descubre por medio de un algoritmo de detección.

En algunos casos, la falla puede ser identificada; en otros permanece como una hipótesis que no puede ser adecuadamente verificada.

Ejemplos de fallas:

a) Falla física permanente: Estado lógico invariable en la salida de una compuerta.

b) Falla física transitoria: Cambio en el estado de una celda de memoria como consecuencia de un fenómeno de radiación.

c) Falla de diseño: rutina de validación que no verifica en forma apropiada restricciones específicas.

d) Falla de especificación: especificación ambigua de las restricciones en el intervalo de variación de los datos de entrada.

1.5. AVERIA DE SISTEMA

Consiste en la pérdida del servicio propio que es experimentado por el usuario (esto es, un humano, otro subsistema o sistema) en el límite de un recurso (llamado límite de servicio).

El lugar en que sea definido el límite de servicio del recurso es lo que permite establecer la diferencia entre una avería, un error y una falla.

Si la pérdida de servicio propio es percibida por el usuario en el límite del recurso, dicha pérdida se denomina avería. Si el estado impropio se presenta dentro de un recurso se conoce como error. En ambos caso, la causa del error se llama falla.

Dado que los recursos están anidados, una falla puede ser percibida como una avería si el límite de servicio es movido hacia adentro y definido en el lugar en que se localiza la falla.

Por consiguiente, la secuencia falla, error, avería ocurre repetidamente a través del sistema conforme el error se propaga.

Ejemplos de averías:

a) El operador de una terminal (usuario) percibe que su computador (recurso) no produce la respuesta apropiada (servicio) a un determinado comando.

b) El CPU (usuario) percibe que la memoria (recurso) ha entregado una palabra (servicio) con paridad errónea.

1.6. DIFERENCIA ENTRE AVERIA, ERROR Y FALLA

La diferencia entre estos tres conceptos es definida por la localización del límite del servicio del usuario.

Si un estado impropio o pérdida de servicio aparece en el límite de servicio (límite del sistema), dicho estado se define como avería. Si un estado impropio ocurre dentro del recurso, dicho estado se define como un error.

La causa, tanto de la avería como del error, se conoce como falla.

Dado que los recursos están anidados, un error puede ser percibido como una avería si el límite del servicio es movido hacia dentro y se coloca en el punto en que está localizado el error.

1.7. CONFIABILIDAD

La confiabilidad de un sistema es una función del tiempo R(t) que indica la probabilidad condicional de que un sistema entregue el servicio apropiado en el intervalo [0,t],dado que entregó el servicio apropiado en el tiempo t=0 [Siew 84].

La confiabilidad es un parámetro utilizado para caracterizar sistemas en los cuales no es posible la recuperación en línea del sistema (tal como los computadores de satélites), o en los cuales está sirviendo una función crítica (por ejemplo, computadores de vuelo en aviones).

El vocablo "sistema confiable" es un término cualitativo utilizado para caracterizar un sistema en el que se puede justificadamente confiar que entregue el servicio requerido cuando éste se necesita.

1.8. DISPONIBILIDAD

La disponibilidad de un sistema es una función del tiempo A(t) que indica la probabilidad de que un sistema sea operacional en el instante t. Si el límite de esta función existe cuando t tiende a infinito, entonces la disponibilidad expresa la fracción esperada de tiempo de que el sistema esté disponible para realizar tareas útiles [Siew 84].

Este parámetro es usado para evaluar sistemas en los cuales el servicio puede retrasarse o negarse por cortos períodos sin provocar consecuencias irreparables.

Actividades tales como el mantenimiento preventivo y la reparación reducen la disponibilidad del sistema.

2. CONCEPTOS SOBRE FALLAS

2.1. FALLA ACTIVA

Falla que origina errores.

2.2. FALLA DURMIENTE

Falla que permanece en el sistema o recurso pero que no ha causado errores. Permanece como amenaza de error (error potencial).

2.3. FALLA TRANSITORIA (Falla "soft")

Falla causada por cambios temporales (no permanentes) en los componentes del sistema. Este tipo de falla usualmente altera los valores presentes de las variables lógicas en el sistema pero no causa daños irreversibles a los componentes. Un ejemplo de este tipo de falla es la radiación incidente sobre celdas de memoria.

2.4. FALLA INTERMITENTE - o pseudotransitoria-

Falla originada en defectos de diseño o de los componentes, los cuales requieren la presencia de una combinación especifica de un número de variables lógicas o condiciones ambientales para su manifestación, tales como las fallas sensibles a patrones en memorias de semiconductores.

2.5. FALLA PERMANENTE (falla "hard")

Falla producida por cambios irreversibles en los componentes. Este tipo de fallas conduce a una transformación permanente del diseño original en un nuevo diseño que tiene una especifición diferente y que no siempre se comporta en la forma apropiada.

2.6. LATENCIA DE FALLA

Tiempo que transcurre entre la ocurrencia de una falla y el primer error detectable que esa falla produce.

3. CONCEPTOS SOBRE ERRORES

3.1. ERROR LATENTE

Error existente en el sistema que no ha sido descubierto por un algoritmo de detección ni ha causado una avería.

3.2. LATENCIA DE DETECCION DE ERROR

Tiempo que transcurre desde que un error es causado por una falla hasta que dicho error es detectado u ocasiona una avería.

4. ETAPAS DE RESPUESTA A UNA FALLA

El diseño de un sistema tolerante a fallas involucra definir, para cada una de las fallas, una respuesta en la que se combinan una o más etapas con el fin de alcanzar el comportamiento especificado. La organización cronológica de estas etapas puede variar en respuestas diferentes. Durante la vida de un sistema pueden distinguirse las siguientes etapas [Siew 84]:

4.1. CONFINAMIENTO (O AISLAMIENTO) DE FALLA

Es el proceso realizado para limitar los efectos de una falla a un área determinada. Esta acción puede realizarse mediante la utilización de circuitos de aislamiento.

Para realizar el confinamiento de una falla, ésta debe haber sido previamente detectada por medio de circuitos o algoritmos de detección de falla.

Es necesario restringir la localización de una falla a una extensión o tamaño determinados para poder utilizar otras etapas de respuesta a fallas tales como los mecanismos de recuperación o restauración del sistema.

4.2. DETECCION DE ERROR

Procesos ejecutados con la finalidad de descubrir (determinar la parte de un sistema en que se encuentra) y señalar la existencia de un error (estado impropio dentro del sistema), causado por una falla, que pueda conducir a una pérdida de servicio en el límite del sistema (avería).

Para realizar este proceso pueden utilizarse técnicas tales como mecanismos de verificación de paridad, de consistencia o diversos tipos de protocolos.

Existen dos técnicas principales de detección de error:

4.2.1. DETECCION EN LINEA

Es el proceso de localización de errores que se ejecuta en forma concurrente (en paralelo) con la operación normal del sistema; provee capacidad de detección en tiempo real.

4.2.2. DETECCION FUERA DE LINEA

Tarea de descubrimiento de errores realizada en forma excluyente del trabajo útil del sistema. En este caso, los algoritmos de detección de error deben ejecutarse cuando los dispositivos están ociosos o multiplexados con la programación operativa. En consecuencia, la detección fuera de línea asegura la integridad antes, y posiblemente en intervalos durante la operación.

4.3. ENMASCARAMIENTO DE FALLA

Técnica consistente en ocultar los efectos de una falla. Esta acción se realiza con el propósito de evitar o impedir la propagación de error a través de los límites definidos por la partición del sistema y por los mecanismos de detección de error.

El encubrimiento de error típicamente se realiza mediante el aislamiento de una parte que se descubrió una falla. La utilización del circuito de votación de mayoría para desactivar elementos detectados como fallados es un ejemplo de enmascaramiento de falla.

4.4. REINTENTO DE EJECUCION

Segundo y sucesivos intentos de ejecución de una operación realizados con el propósito de obtener un resultado exitoso. Es especialmente útil en el caso de fallas transitorias que no causan daño físico.

4.5. DIAGNOSTICO DE FALLA

Proceso orientado a identificar y señalar la causa de los errores detectados en alguno de los recursos del sistema. Involucra el uso de algoritmos (implementados por programación o por electrónica) que permiten realizar una evaluación crítica de todas las posibles fuentes de error.

Los resultados del diagnóstico de fallas pueden utilizarse para apoyar la ejecución de otros tipos de respuesta a fallas tales como confinamiento y enmascaramiento de falla o reconfiguración de sistema.

4.6. RECONFIGURACION DE SISTEMA

Proceso mediante el cual se reemplaza o aisla un módulo o componente fallado de un sistema. Mediante este proceso, el componente fallado puede reemplazarse por refacciones de respaldo o, alternativamente, ser desconectado. En caso de proceder a su desconexión, se produce una degradación del sistema y, por tanto, se disminuye (parcialmente) la capacidad de operación o diagnóstico del sistema.

4.7. RECUPERACION

Consiste en eliminar los errores causados por una falla y restaurar el sistema a un estado válido sin errores conocidos.

Para realizar esta tarea el estado del sistema (contador de programa, contenido de los registro internos, área de datos y apuntador de la pila), es respaldado hasta algún punto anterior a la detección del error y a partir del cual se reinicia la operación.

En general, la recuperación debe ejecutarse desde un punto anterior suficientemente separado que permita eliminar los efectos de los errores que ocurrieron con anterioridad al punto en que se diagnosticó la falla.

4.8. REINICIO

Restitución o restablecimiento de las operaciones del sistema después de que la información ha sido dañada (por un error) o en el caso de que el sistema no esté diseñado para recuperación.

Existen tres tipos principales de reinicios:

4.8.1. REINICIO EN FRIO

Corresponde a una recarga completa de programa. Es la forma más drástica de reinicio y se ejecuta cuando no quedan procesos sobrevivientes.

4.8.2. REINICIO EN TIBIO

Situación en la cual solamente algunos procesos requieren ser recargados para poder continuar sin pérdida de información.

4.8.3. REINICIO EN CALIENTE

Consiste en continuar la ejecución a partir del punto en que se detectó la falla. Es posible únicamente si no ha ocurrido un daño severo (pérdida en la información).

4.9. REPARACION DE SISTEMA

Acción interna o externa de eliminación de error que consiste en reemplazar un componente o módulo fallado (o defectuoso), durante la operación del sistema, sin interrumpir la entrega de servicio.

Existen dos tipos básicos de reparación:

4.9.1. CAMBIO DE ESTADO

Consiste en recargar en el sistema un nuevo programa desde otro sistema (tal como una estación remota o una unidad de almaceanmiento externa)

4.9.2. CAMBIO DE ESTRUCTURA

Corresponde al reemplazo manual o automático de un elemento defectuoso por una refacción en buen estado.

En lo que respecta a la reparación estructural se distinguen dos tipos de sistemas:

4.9.2.1. SISTEMA TOLERANTE A FALLAS CERRADO

Sistema que emplea únicamente reparación interna (o incorporada) para efectuar el proceso de recuperación.

4.9.2.2. SISTEMA TOLERANTE A FALLAS REPARABLE

Sistema que está en capacidad de tolerar reparación tanto interna como externa durante su operación.

Los sistemas de enmascaramiento de falla son un caso particular de la reparación interna en sistemas cerrados.

De acuerdo con el momento en que se realiza, hay dos tipos de reparación:

4.9.3. REPARACION FUERA DE LINEA

En este caso, ninguno de los componentes fallados es indispensable para la operación del sistema y cualquiera puede ser reemplazado sin ocasionar la baja del sistema

4.9.4. REPARACION EN LINEA

En este caso, el componente fallado puede ser físicamente reemplazado, en forma automática o manual, por una refacción de respaldo sin interrumpir la operación del sistema durante un proceso de reconfiguración, o bien, puede ser aislado sin perturbar la operación por un proceso de enmascaramiento de falla.

4.10. REINTEGRACION DE COMPONENTE

Consiste en el proceso de reincorporación de un módulo reparado al sistema. Para reintegrar una unidad, ésta debe haber sido totalmente "educada" (habérsele transferido el estado del sistema) y puesta en sincronía con las demás. Asimismo, la reintegración debe ser realizada sin interrumpir la operación del sistema.

5. REDUNDANCIA MODULAR TRIPLE

Es uno de los métodos más usados de detección de errores en "módulos" idénticos. Típicamente, cada módulo es una unidad de procesamiento tal como un computador, un microprocesador o un coprocesador; no obstante, también puede ser una unidad menos compleja tal como un sumador o una compuerta lógica [Lala 85].

Una arquitectura basada en redundancia modular triple consta de tres procesadores (que operan en forma concurrente) y de un circuito de votación que se encarga de comparar las salidas de los procesadores para determinar, mediante el principio de votación mayoritaria, posibles fallas generadas durante el tiempo de ejecución.

BIBLIOGRAFIA

[Alla 85] Allan, Roger; "Technology Report for Fault Tolerant Computing Software is Finding a Powerfully in Hardware", *Electronic Design*, Vol. 33, No. 25, 31 Octubre 1985, pp. 111-118.

[Aviz 78] Avizienis, Algirdas (Guest Editor); "Special Issue on Fault-Tolerant Digital Systems: Is Hal Going to join us before 2001?", *Proceedings of the IEEE*, Vol. 66, No. 10, Octubre de 1978, pp. 1107-1108.

[Aviz 87] Avizienis, Algirdas; "Design paradigm for fault-tolerant systems", *AIAA Computers in Aerospace VI Conference*, Wakefield, MA, Octubre 7-9, 1987.

[Aviz 90] Avizienis, Algirdas (coordinator) y otros; Lecture Notes of course: *The application of fault tolerance technology*, University of California, Los Angeles, Julio 31- Agosto 3, 1990, pp. 1-22 / 1-23.

[Barl 78] Barlett, J.F.; "A 'Non-Stop' Operating System", *Procedures of Hawaii International Conference on System Sciences*, 1978, pp. 103-109.

[Bonn 80] Boone, L.A. y Liebergot, H.L.; "Availability, Reliability, and Maintainability Aspects of the Sperry Univac 1100/60", *Procedures of 10th International Fault Tolerant Computing Symposium*, IEEE-CS Press, Los Alamitos, California, 1980, pp. 3-8.

[Boss 82] Bossen, D.C. y Hsiao, M.Y., "Model for Transient and Permanent Error-Detection and Fault Isolation", *IBM Journal of Research and Development*, Vol. 26, No. 1, Enero 1982, pp. 67-77.

[Bruc 85] Bruckert, W.F. y Josephson, R.E.; "Designing Reliability into the VAX 8600 System", *Digital Technical Journal*, Vol. 1, Agosto 1985, pp. 71-77.

[Coop 76] Cooper, A.E. y Chow, W.T.; "Development of On-board Space Computer Systems", *IBM Journal of Research and Development*, Vol. 20, No. 1, Enero 1976, pp. 5-19

[Dall 89] *1989 Product Data Book*, Dallas Semiconductor Corporation, Dallas, Texas, 1989.

[Drou 71] Droulette, D.L.; "Recovery Through Programming System/360 - System/370", *Procedures of AFIPS SJCC*, Vol. 38, AFIPS Press, Montvale, N.J., 1971, pp. 467-476.

[Egge 87] Eggebrecht, Lewis C.; *Interfacing to the IBM Personal Computer*, Howard W. Sams & Co.,Indianapolis, 1987.

[Fuji 90] Fujiwara, Eiji y Pradhan, Dhiraj K.; "Error-Control Coding in Computers", *Computer*, Vol. 23, No. 7, Julio 1990, pp. 63-72.

[Goft 86] Gofton, Peter W.; *Mastering Serial Communications*, Sybex, San Francisco, 1986.

[Gray 86] Gray, Jim; "Why do computers stop and what can be done about it?"; *IEEE Fith Symposium On Reliability in Distributed Software and Database Systems;* Los Angeles, California, 13-15 Enero 1986, pp. 3-12.

[Gray 90] Gary, Jim; "A Census of Tandem System Availability Between 1985 and 1990", *IEEE Transactions on Reliability (Special Issue on Measssurement)*, Vol. 39, Num. 4, Octubre 1990.

[Hans 87] Hansen, Augie; "Mapping PC Address Space", *PC Tech Journal*, Marzo 1987, pp. 102-112.

[Hita 90] Hitachi; *SRAM Data Book, Hitachi America Ltd.*, Brisbane, California, Marzo 1990.

[Ihar 78] Ihara, Hirokazu y otros; "Fault-Tolerant Computer System with Three Symmetric Computers", *Proceedings of the IEEE*, Vol. 66, No. 10, Octubre de 1978, pp. 1160-1177.

[Inte 81] *iAPX 86, 88 User's Manual*, Intel Corporation, Santa Clara, California, 1981.

[Inte 83] *iAPX 286, Hardware Reference Manual*, Intel Corporation, Santa Clara, California, 1986.

[Int 86a] *80386 Hardware Reference Manual*, Intel Corporation, Santa Clara, California, 1986.

[Int 86b] *iAPX 86, 88, 186 and 188 User's Manual and Programmer's Reference*, Intel Corporation, Santa Clara, California, 1986.

[Int 86c] *80386 Programmer's Reference Manual*, Intel Corporation, Santa Clara, California, 1986.

[Int 86d] *Microsystem Components Handbook (Microprocessor Volume I)*, Intel Corporation, Santa Clara, California, 1986.

[Jigo 79] Jigour, Robin; "Using the 8259A Programmable Interrupt Controller", *iAPX 86,88 User's Manual*, Setiembre 1979, pp. A-136 / A-173.

[John 84] Johnson, Dave; "The Intel 432: A VLSI Architecture for Fault-Tolerant Computer Systems", *Computer*, Vol. 17, No. 8, Agosto 1984, pp. 40-48.

[Jour 96] Jourdain, Robert; *Programmer's Problem Solver for the IBM PC, XT & AT*, Prentice/Hall Press (Brady Book); New York, 1986.

[Kats 78] Katsuki, D. y otros; "Pluribus - An Operational Fault Tolerant Computer System", *Procedures of IEEE*, Vol. 66, No. 10, Octubre 1978, pp. 1146-1159.

[Kore 90] Koren, Israel y Singh, Adit D.; "Fault Tolerance in VLSI Circuits", *Computer*, Vol. 23, No. 7, Julio 1990, pp. 73-83.

[Lala 85] Lala, Parag K.; *Fault Tolerant and Fault Testable Hardware Design*, Prentice/Hall International, Englewood Cliffs, New Jersey, 1985.

[Mic 88a] Michael, Martin S y Durich, Daniel G.; "The NS16550A: UART, Design and Application Considerations" (Application Note 491), *Advanced peripherals - Data Communications Local Area Networks UARTs (Handbook)*, National Semiconductors, Santa Clara, California, 1988, pp. 4-58 / 4-84.

[Mic 88b] Michael, Martin S; "A comparison of the INS8250, NS16450 and NS16550A Series of UARTs" (Application Note 493), *Advanced peripherals - Data Communications Local Area Networks UARTs (Handbook)*, National Semiconductors, Santa Clara, California, 1988, pp. 4-85 / 4-89.

[Morg 78] Morganti, M.; Coppadoro, G. y Ceru, S.; "UDET7116- Common Control for PCM Telephone Exchange: Diagnostic Software Design and Availability Evaluation", *Procedures of Eighth International Fault Tolerant Computing Symposium*, IEEE-CS Press, Los Alamitos, California, 1978, pp. 16-23.

[Jour 96] Jourdain, Robert; *Programmer's Problem Solver for the IBM PC, XT & AT*, Prentice/Hall Press (Brady Book); New York, 1986.

[Kats 78] Katsuki, D. y otros; "Pluribus - An Operational Fault Tolerant Computer System", *Procedures of IEEE*, Vol. 66, No. 10, Octubre 1978, pp. 1146-1159.

[Kore 90] Koren, Israel y Singh, Adit D.; "Fault Tolerance in VLSI Circuits", *Computer*, Vol. 23, No. 7, Julio 1990, pp. 73-83.

[Lala 85] Lala, Parag K.; *Fault Tolerant and Fault Testable Hardware Design*, Prentice/Hall International, Englewood Cliffs, New Jersey, 1985.

[Mic 88a] Michael, Martin S y Durich, Daniel G.; "The NS16550A: UART, Design and Application Considerations" (Application Note 491), *Advanced peripherals - Data Communications Local Area Networks UARTs (Handbook)*, National Semiconductors, Santa Clara, California, 1988, pp. 4-58 / 4-84.

[Mic 88b] Michael, Martin S; "A comparison of the INS8250, NS16450 and NS16550A Series of UARTs" (Application Note 493), *Advanced peripherals - Data Communications Local Area Networks UARTs (Handbook)*, National Semiconductors, Santa Clara, California, 1988, pp. 4-85 / 4-89.

[Morg 78] Morganti, M.; Coppadoro, G. y Ceru, S.;"UDET7116- Common Control for PCM Telephone Exchange: Diagnostic Software Design and Availability Evaluation", *Procedures of Eighth International Fault Tolerant Computing Symposium*, IEEE-CS Press, Los Alamitos, California, 1978, pp. 16-23.

[Moor 75] Moore, William R.; "Some fault tolerance design principles of modular redundant structures", *Digest of papers FTC-5: International Symposium on Fault-Tolerant Computing;* Paris, Francia, Junio 18-20, 1975, pp. 119 - 123.

[Nati 88] *Advanced Peripherals, Data Communications Local Area Networks UARTs,* National Semiconductor Corporation, Santa Clara, California, 1988.

[Nels 90] Nelson, Victor P.; "Fault tolerant computing: Fundamental Concepts", *Computer,* Vol. 23, No.7, Julio 1990, pp. 19-25.

[Ng 80] Ng, Ying W. y Avizienis, Algirdas; "A Unified Reliability Model for Fault-Tolerant Computers", *IEEE Transactions on Computers,* Vol. C-29, No. 11, Noviembre 1980, pp. 1002-1011.

[Papp 88] Pappas, Chris H. y Murray, William H.; *Manual del microprocesador 80386,* Osborne / McGraw-Hill; Madrid; Agosto 1988.

[Pier 65] Pierce, William H.; *Failure-Tolerant Computer Design,* Academic Press, New York, 1965.

[Rect 80] Rector, Russell y Alexy, George; *The 8086 Book;* Osborne / McGraw-Hill; Berkeley, California, 1980.

[Sarr 84] Sarrazin, David B. y Malek, Miroslaw; "Fault-Tolerant Semiconductor Memories", *Computer,* Vol. 17,No. 8, Agosto 1984, pp. 49-56.

[Serl 84] Serlin, Omri; "Fault-Tolerant Systems in Commercial Applications", *Computer,* Vol. 17,No. 8, Agosto 1984, pp. 19-30.

[Siew 78] Siewiorek, Daniel P.y otros; "A Case Study of C.mmp, Cm*, and C.vmp: Part 1 - Experiences with Fault Tolerance in Multiprocessor Systems", *Procedures of IEEE*, Vol. 66, No. 10, Octubre 1978, pp. 1178-1199.

[Siew 82] Siewiorek, Daniel P.y Swarz, Robert S.; *The Theory and Practice of Reliable System Design*", Digital Press, Bedford, Massachusetts, 1982.

[Siew 90] Siewiorek, Daniel P.; "Fault Tolerance in Commercial Computers", *Computer*, Vol. 23, No. 7, Julio 1990, pp. 26-37.

[Sing 90] Singh, Adit D. y Murugesan, Singaravel; "Fault Tolerant Systems", *Computer*, Vol. 23, No. 7, Julio 1990, pp. 15-17.

[Syna 83] *Synapse Transaction Processing System Overview*, Synapse Computer Corporation, Milpitas, California, 1983.

[Tand 86] *Non-Stop Systems: System Description Manual*, Tandem Computers, Cupertino, California, Octubre 1986.

[Tend 82] Tendolkar, N.N. y Swan, R.L.; "Automated Diagnostic Methodology for the IBM 3081 Processor Complex", *IBM Journal of Research and Development*, Vol. 26, No. 1, Enero 1982, pp. 78-88.

[Thor 87] Thorne, Michael; *Progamming the 8086/8088 for the IBM PC and Compatibles*, Benjamin / Cummings Publishing Company Inc., Menlo Park, California, 1987.

[Toy 78] Toy, Wing N.; "Fault - Tolerant Design of Local ESS Processors", *Procedures of IEEE*, Vol. 66, No. 10, Octubre 1978, pp. 1126-1145.

[Toy 84] Toy, Wing N. y Morganti, Michele; "Fault Tolerant Computing", *Computer*, Vol. 17, No. 8, Agosto 1984, pp. 6-7.

[Webb 91] Webber, S.; "The Stratus Arquitecture", *The Theory and Practice of Reliable System Design* (de D.P. Siewiorek y R.S. Swarz), Capitulo 13, Digital Press, Bedford, Massachussets, 1991.

[Wens 81] Wensley, John H.; "Fault-Tolerant computers ensure reliable industrial controls", *Electronic Design*, Vol. 29, No.13, 25 Junio 1981, pp. 129-135.

[Zorp 85] Zorpette; "Computers that are 'never' down", *IEEE Spectrum*, Vol. 22, No. 4, Abril 1985, pp. 46-54.