UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
"ARAGON"

" ENSAMBLADO, PRUEBAS, OPERATIVAS Y PROGRAMACION DE LAS RUTINAS DE CONTROL DE LA UNIDAD DE DETENCION DE FALLAS Y CONTROL DE LA COMPUTADORA INDUSTRIAL TOLERANTE A FALLAS DEL II UNAM"

T E S I S

Que para obtener el Título de:
INGENIERO MECANICO ELECTRICISTA
P r e s e n t a :
GERARDO TORRES RODRIGUEZ

Asesor: M. I. Esaú Vicente Vivas

San Juan de Aragón, Edo. de Méx. 1995

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

Dedicatoria

A mis padres y hermanos, por los esfuerzos y el apoyo recibido durante mis estudios.

Agradecimientos

En primer lugar quiero agradecer a mi asesor Esaú Vicente Vivas, por todo el apoyo que recibí en la realización de éste trabajo. De igual forma agradezco a la coordinación de Automatización del Instituto de Ingeniería de la UNAM todas las facilidades otorgadas para llevar a cabo mi tesis.

También agradezco toda la ayuda que me brindaron mis compañeros y amigos que trabajan en el Instituto, en especial a Rafael, Martín y Alfredo, quienes compartieron muchos de sus conocimientos conmigo.

Por último quiero agradecer a todas aquellas personas que de alguna manera me motivaron y alentaron en todos estos años de actividad escolar, gracias por su amistad , por su comprensión y por su amor.

CAPITULO UNO

INTRODUCCIÓN

1.1 Introducción

Con la creciente producción y comercialización de equipo de computo aunado a la corriente de globalización económica, hoy en día muchos sectores productivos tales como la industria automotriz [Shladover 1993, Zanoni 1993], la banca, la industria química, la industria de la transformación, entre otras, requieren, por un lado, la automatización parcial o total de la gran mayoría de sus procesos de producción, y por otro demandan altos índices de confiabilidad en el manejo, procesamiento y distribución de su información. Este tipo de industrias busca mejorar la calidad de sus productos o servicios, producir a grandes volúmenes (para reducir costos de producción), o bien, brindar un mejor servicio y tener una mayor cobertura del mismo, para poder competir y alcanzar liderazgos en mercados internacionales.

Para ello, mucha de la automatización de sus procesos se basa en la asistencia y supervisión mediante equipo de computo. Las características de estos equipos varía en tamaño, capacidad, operación continua, tolerancia a fallas y en capacidad de respuesta cuando el sistema incurre en errores. Si bien es cierto que el proceso de delegar responsabilidades a las computadoras puede generar catástrofes cuando ellas incurren en comportamientos erróneos (como por ejemplo en las transacciones bancarias, los sistemas para el control aéreo comercial y militar, en control industrial, aplicaciones espaciales o bien cuando de ellas dependen vidas humanas) las técnicas de tolerancia a fallas han evolucionado y continúan desarrollándose para tratar de reducir este tipo de problemas, en la medida de lo posible.

La computación tolerante a fallas supone que los sistemas digitales son susceptibles de incurrir en diferentes tipos de fallas y pretende alcanzar altos índices de confiabilidad, incorporando en el equipo varios tipos de redundancias. Estas redundancias pueden, en algunos casos, permitir que el sistema continúe sus operaciones aun ante la presencia de una o más fallas. En algunos casos, la redundancia puede ayudar a minimizar el daño originado por una falla sin que necesariamente el sistema siga operando. En otras situaciones, la redundancia facilita el diagnóstico y la reparación de un sistema para reducir el tiempo que permanece sin operar. Las cantidades y tipos de redundancias requeridas dependen de la aplicación y de sus posibles repercusiones en caso de que el sistema falle [Vivas 1994]

En esta tesis se describe el desarrollo, ensamble y prueba de la electrónica de supervisión y control de una computadora tolerante a fallas que se construye en el Instituto de Ingeniería de la UNAM..

En este capitulo se dan conceptos y aplicaciones de tolerancia a fallas y en el capitulo dos se describe el diseño global de la computadora de la UNAM. En el tercer capítulo se presenta el diseño de la electrónica que controla la arquitectura tolerante a fallas. El capítulo cuatro trata los aspectos de ensamble y pruebas, tanto para las unidades de detección de fallas como para sus respectivas unidades de aislamiento. En el capítulo cinco se presentan los programas desarrollados en esta tesis para la unidad de detección de fallas. En el sexto capítulo se dan algunos avances de las pruebas preliminares de integración de la computadora de la UNAM; y finalmente, en el capítulo siete se dan las conclusiones y recomendaciones que emanan del presente trabajo de tesis.

En las siguientes secciones se hablará sobre el concepto de tolerancia a fallas en equipo de computo, se mencionan algunas filosofías y técnicas de diseño tolerante a fallas (TF) basadas en el uso de redundancias por hardware y por software, también se presenta un breve panorama de los campos de aplicación donde el computo tolerante a fallas ha cubierto de manera satisfactoria las necesidades de los sectores productivos; en la última sección se dan las principales características de la computadora tolerante a fallas (CTF) que se ha desarrollado en el instituto de ingeniería de la UNAM y en el apéndice B se muestran algunas arquitecturas (prácticas y teóricas) propuestas por otros autores para lograr el atributo de tolerancia a fallas.

1.2 Conceptos de Tolerancia a Fallas

Los sistemas tolerantes a fallas (STF) son aquellos que realizan correctamente su tarea encomendada aun en presencia de anomalías en la electrónica (hardware) o errores en la programación (software). Por ejemplo, el efecto de un error en algún módulo electrónico de un STF se sobrelleva de tal forma que no perjudica la operación correcta y continua del sistema. En el campo de la tolerancia a fallas se hace una clara distinción entre los conceptos de fallas, errores y averías. Una falla es un defecto físico, una imperfección o el caso de una ruptura dentro de un módulo de hardware o software, ejemplos de fallas son: cortos circuitos entre conductores eléctricos, rupturas en conductores, impurezas en los semiconductores; una falla de software puede ser la ejecución de un ciclo iterativo del cual no se pueda salir [Vicente 1994]. El error es la manifestación de una falla, es decir, es una desviación de lo exacto; como ejemplo de un error se puede considerar la salida de una compuerta que ha quedado fija a un nivel bajo, y aunque las entradas cambien la salida sigue presentando el nivel bajo, entonces se dice que la salida tiene un error. Finalmente si el error tiene

repercusiones en las actividades del sistema, entonces el sistema está averiado, es decir, una avería es la no ejecución de una tarea que tenía que realizarse.

Para los lectores interesados en tener un panorama más amplio acerca del desarrollo de la computación tolerante a fallas ver [Rennels 1984, Vicente 1993].

1.3 Filosofías de tolerancia a fallas

Existen tres metodologías básicas para mejorar o mantener el buen desempeño de un sistema cuyo ambiente no está exento de fallas:

- Prevención de fallas. Utilizada en primer término, se vale de la revisión del diseño, verificación y en el control de calidad de los componentes que utilizará el sistema.
- Enmascaramiento de fallas. Evita la introducción de fallas en la estructura de la información de un sistema, es decir, se supervisa la información que será utilizada con el fin de no procesar datos erróneos, evitando con ello la avería del sistema.
- Tolerancia a fallas. Se puede lograr de varias formas a saber, con el enmascaramiento de fallas, con la detección y localización de la falla ocurrida y con la reconfiguración del sistema.

1.4 Técnicas de diseño en tolerancia a fallas

En la tolerancia a fallas todas las filosofías se apoyan en el uso de elementos de redundancia, estos elementos no son necesariamente réplica de los componentes utilizados, pues el término redundancia se extiende a la adición de información, recursos, o tiempo necesarios para la operación normal y continua del sistema, de esto se desprende que existen cuatro tipos de redundancias: por hardware, por software, de información y en el tiempo. A continuación se describirán brevemente las técnicas de redundancia por hardware y por software debido a que son las más frecuentes de encontrar; si el lector desea profundizar en las técnicas de redundancia de información y en el tiempo ver [Johnson W. 1989].

1.4.1 Redundancia por hardware

Por redundancia de hardware se entiende la adición de componentes complementarios que sirven para la detección de fallas, es una de las más comunes. Existen tres formas básicas de llevarla a cabo, la primera es la técnica pasiva que consiste en el enmascaramiento de fallas, en ésta se tienen N módulos idénticos operando concurrentemente y comparando sus salidas para detectar algún módulo con falla y no permitir que su información salga al exterior;

la segunda es la técnica activa que consiste en tener dos módulos iguales cuyas salidas se comparan continuamente y al existir diferencias entre los datos comparados se ejecutan rutinas de diagnóstico para ubicar la falla y remover el módulo fallado del sistema; la tercera es la técnica híbrida la cual es la combinación de las técnicas pasivas y activas, donde el enmascaramiento de fallas se usa para prevenir la generación de resultados erróneos, mientras que la detección, localización y recuperación de fallas actúan para aislar el módulo dañado.

1.4.2 Redundancia por software

La redundancia por software consiste en agregar software extra a las rutinas de un sistema. Se hace con la finalidad de detectar y posiblemente tolerar algunas fallas; esta redundancia no deber ser necesariamente la duplicación del programa si no que puede ser un conjunto de líneas adicionales de programa para verificar la magnitud de una señal o pequeñas rutinas que periódicamente prueben una memoria por medio de accesos a localidades específicas. Existen tres técnicas de redundancia por software las cuales son las más empleadas:

1.4.2.1 Examinador de consistencia

Usa el conocimiento *a priori* de las características de la información para verificar su validez.

1.4.2.2 Examinador de capacidad

Para comprobar que un sistema posee la capacidad esperada es necesario aplicar algunas pruebas. Por ejemplo, para verificar la memoria, el procesador escribe y lee patrones de datos en ciertas localidades para saber si los datos se escriben y leen correctamente. Otra técnica consiste en ejecutar periódicamente instrucciones especificas con datos particulares, para comparar sus resultados con los valores esperados que se almacenan en una ROM. En procesamiento en paralelo se ejecutan transferencias de información conocida entre los procesadores interconectados con el fin de verificar que la comunicación entre ellos es confiable.

1.4.2.3 Programación de N versiones

Las técnicas descritas anteriormente se basan en el uso extra de software para detectar fallas en la electrónica, pero en sí no se ha mencionado nada acerca de la detección o tolerancia de fallas en programación; para soportar fallas en

software, muchas de las veces se recurre a la duplicación de los programas que se ejecutarán en el sistema, pero los errores de programación se deben al mal diseño del software o errores de lógica en los programas, por lo que la duplicación de un programa no es una técnica muy confiable de tolerar fallas en software ya que si un programa contiene errores su contraparte también los contendra y el sistema incurrirá en errores.

1.5 Aplicaciones de la computación tolerante a fallas

Las aplicaciones de la tolerancia a fallas pueden dividirse en cuatro áreas: aplicaciones de larga vida, computo crítico, mantenimiento aplazable y alta disponibilidad. Cada aplicación presenta diferentes requisitos de diseño y diversos retos.

Las aplicaciones de larga vida se utilizan en sondas espaciales y satélites, por lo cual deben tener una probabilidad de 0.95 de ser operativos al final de un período de diez años debido a la dificultad y al gran costo implicado para realizar mantenimiento en el espacio [Rennels 1980].

A diferencia de otras aplicaciones, los sistemas de larga vida permiten la extensión de su vida útil al ser puestos en operación una vez más. Además este tipo de aplicaciones permiten a veces ser reconfigurados remotamente por un operador.

Las aplicaciones de computo crítico son las que tienen un mayor auge hoy en día, ejemplo de ellas son aplicaciones donde los cálculos y la operación continua son críticos en diversas ramas productivas, ya sea para la seguridad humana, el cuidado del medio ambiente, etcétera. Como ejemplo tenemos a los servidores de redes de sistemas bancarios, controladores industriales, control aéreo y sistemas militares. Un requisito típico para estas aplicaciones es tener una disponibilidad del 0.97 al final de un periodo de tres horas.

En ocasiones se necesita que los sistemas tolerantes a fallas contengan dos parámetros que aseguren la eficiencia de dichos sistemas en cierto tipo de aplicación (control de reactores nucleares y en la comunicación entre aeronaves militares), estos parámetros son la confiabilidad y la seguridad; existen esquemas que aseguran una alta confiabilidad pero su seguridad es pobre o viceversa. Lo anterior nos lleva a un gran compromiso si se requieren esquemas que proporcionen índices de confiabilidad muy altos y que además cuenten con un alto índice de seguridad; algunas estrategias para estos diseños han sido propuestas por [Nitin H. y Dhiraj K. 1993].

La industria automotriz tiene proyectos en los que aplicará este tipo de sistemas para controlar digitalmente algunos subsistemas como dirección, suspensión, frenado, etc. [Shladover 1993, Zanoni 1993].

Las aplicaciones de mantenimiento aplazable se encuentran frecuentemente donde el mantenimiento es extremadamente costoso, inconveniente o difícil de llevar a cabo.

El principal objetivo de la tolerancia a fallas en es tipo de aplicaciones es la sustitución de partes defectuosas para que el equipo continúe operando; entre una sesión de mantenimiento y otra los sistemas deben manipular las averías y las discontinuidades de servicio de manera autónoma.

En cuanto a la disponibilidad, es un parámetro que se está convirtiendo en un factor clave en muchas aplicaciones, los sistemas bancarios y otros sistemas de tiempo compartido son ejemplo de esta clase de aplicaciones. Los usuarios de estos sistemas deben tener una alta probabilidad de obtener el servicio cuando así lo pidan. El sistema de procesamiento de transacciones NonStop de Tandem [Katzman 1977] es un ejemplo de diseño de alta disponibilidad, otro ejemplo de diseño desarrollado para soportar aplicaciones de propósito general de alta disponibilidad es el procesador 432 de Intel [Johnson 1984] el cual integró en su diseño técnicas de tolerancia a fallas.

Otro tipo de aplicaciones de la redundancia son los esquemas de árbol, los cuales se enfocan al control en tiempo real en sistemas de instrumentación o medición y en la clasificación del procesamiento de bases de datos [Shantanu Dutt 1990].

De gran importancia han sido las técnicas de tolerancia a fallas desarrolladas por Ahmed E. para la fabricación de VLSIs ya que han reducido el número de compuertas que el encapsulado contendrá, además de conferirles el atributo de tolerancia a fallas en su estructura interna [Ahmed E. 1992] [Irith Pomeranz y Sudhakar M. 1993]. También en el desarrollo de "Chips" de Memoria, empleando la redundancia de circuitos y circuitos de códigos de corrección de error se han logrado desarrollar memorias de alto desempeño y alta confiabilidad [Charles H. y Hsing-San Lee 1992].

La tolerancia a fallas se está orientando hacia el mejoramiento en el funcionamiento de sistemas con un gran número de componentes, para ofrecer un alto funcionamiento y confiabilidad. La eficiencia de estas técnicas han sido probadas para conocer su repercusión en tales sistemas [Tein-Hsiang y Kang G. 1990].

Otro campo de gran desarrollo de la tolerancia a fallas es en las actividades relacionadas con los sistemas multiprocesador (Hipercubos, Redes, etc.) debido a que la probabilidad de que el sistema falle se incrementa rápidamente con el

número de procesadores que el sistema tiene. Aquí existen compromisos muy grandes, tales como la reconfiguración del procesador con falla, preservar el orden de interconexión de la estructura multiprocesador y el tiempo de respuesta del sistema para poder estar en disponibilidad cuando se le requiera [Shantanu Dutt 1992].

1.6 La computadora tolerante a fallas del Instituto de Ingeniería

Dentro del Instituto de Ingeniería de la UNAM se ha desarrollado una computadora tolerante a fallas, en su diseño se conjuntaron algunas de las filosofías y técnicas de la tolerancia a fallas que se describieron en secciones anteriores tales como el enmascaramiento de fallas, mantenimiento en línea y la posibilidad de reconfigurarse autónomamente, lo que incluye detección de fallas, localización y aislamiento.

Las características principales de ésta computadora son :

a) Detección de fallas de manera transparente para el usuario.
b) Confinamiento y aislamiento de las zonas de falla en su arquitectura.
c) Operación continua del sistema aun ante la presencia de fallas en algunos de sus subsistemas.
d) Diferentes posibilidades de reconfiguración automática o a petición del usuario.
e) Mantenimiento en línea.
f) Manejo de componentes externos, tales como teclado matricial, pantalla de cristal líquido, bocina y línea de reset compartido; todos ellos sin necesidad de conexiones externas.
g) Completa compatibilidad con PCs tipo IBM, por lo que puede manejar toda la gama de paquetes y periféricos que existen en el mercado.

CAPITULO DOS

DESCRIPCIÓN GLOBAL DE LA ARQUITECTURA DE LA COMPUTADORA TOLERANTE A FALLAS (CTF) DEL II DE LA UNAM

2.1 Introducción

Como se describió en el primer capítulo, existen varias técnicas para diseñar sistemas tolerantes a fallas, en este capítulo se hablará sobre la arquitectura de la computadora tolerante a fallas (CTF) que se ha desarrollado en la UNAM. Se dará un panorama general de la arquitectura de la CTF y en el siguiente capítulo se describirá de manera más amplia una de las tarjetas que componen al sistema tolerante a fallas, la cual lleva por nombre Unidad de Detección de Fallas y Control (UDFC).

2.2 Arquitectura de la CTF

El principio de funcionamiento de la CTF utiliza la técnica de voteo mayoritario, por lo que su arquitectura contempla subsistemas redundantes, debido a esto la computadora está constituida por 11 tarjetas electrónicas las cuales se muestran en la figura 2.1.

A las tarjetas se les asignaron los siguientes nombres, los cuales serán utilizados a lo largo del texto.

- Tarjetas de aislamiento de procesadores (TAUPs).
- Unidades de procesamiento (UPs).
- Tarjetas de aislamiento de unidades de detección de fallas y control (TAUDFC).
- Unidades de detección de fallas y control (UDFCs).
- Unidad de interfaz múltiple (UIM).

Como se aprecia en la figura 2.1 se necesitan dos ductos para que la CTF pueda operar de manera eficiente, dichos ductos se denominaron:

- Ducto AT.
- Ducto TF.

En las siguientes secciones se dará una descripción general de cada uno de los subsistemas que componen a la CTF y en el siguiente capítulo se tratará con mayor detalle a la UDFC, que es el propósito de esta tesis.

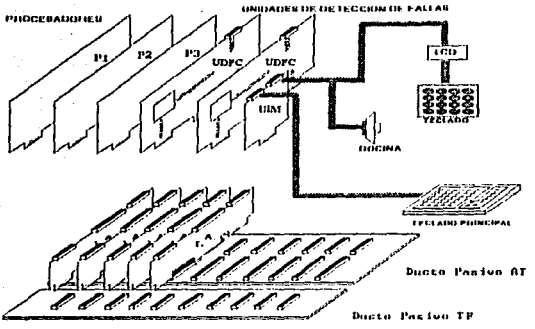


Figura 2.1 Tarjetas que componen a la CTF

2.3 Tarjetas de aislamiento para procesadores (TAUPs)

Una de las características más importantes de la computadora es el mantenimiento en línea, es decir, poder insertar tarjetas nuevas sin fallas o quitar tarjetas dañadas del sistema TF sin ocasionar cortos circuitos o colisiones en los ductos de la CTF y sin detener la operación del equipo.

Este atributo fue logrado con la ayuda de las TA, en particular las TAUPs se encargan de llevar el control del paso de señales desde procesadores hacia los ductos y de los demás subsistemas hacia procesadores.

Debido a que la técnica de voteo mayoritario necesita de un mínimo de tres procesadores operando de manera concurrente para poder realizar las comparaciones, detectar fallas y saber con seguridad que procesador falló, se necesitó de tres TAUPs.

Además, como solo un procesador puede llevar el control de los periféricos, acceso a memoria, salida a puertos, etc., la TA del procesador principal de la CTF permitirá que este procesador lleve toda su información hacia los ductos, mientras que las otras TA impedirán que sus respectivos procesadores reflejen sus señales hacia los ductos logrando así el aislamiento de señales y evitando posibles cortos. Como los tres procesadores deben capturar los mismos datos y generar los mismos resultados, las TAUPs deben permitir el paso de información

proveniente de los periféricos externos o disco duro hacia los tres procesadores, es aquí donde las TAUPs cobran una mayor importancia pues actúan como llaves que permiten o no el paso de señales.

Para que las TAUPs sepan cual de ellas estará asociada a un procesador principal y cuales a uno redundante, contienen lógica de control implementada en circuitos denominados Gals. En la figura 2.2 se muestra una TAUP ensamblada completamente.

2.4 Unidades de procesamiento (UPs)

Como se dijo anteriormente, la arquitectura de la CTF contempla tres UPs para realizar comparaciones de datos, detectar fallas y saber sin dudas si algún procesador presenta falla; cabe mencionar que los datos por comparar son los que se asocian a transferencias de memoria y accesos a puertos. La electrónica que se encarga de detectar fallas y realizar alguna reconfiguración se discutirá en el siguiente capítulo.

Si bien se compara exclusivamente la información del bus de datos, ello redunda en la detección indirecta de fallas en memoria, líneas de direcciones y en señales de control; pues ante la falla de alguna de ellas se originan accesos incorrectos que se manifiestan en diferencias entre los datos durante el proceso de comparación. Es por esto que la detección y el diagnóstico de fallas de la arquitectura propuesta se extiende a todos y cada uno de los componentes integrados en las UPs.

Cada UP está constituida por una tarjeta madre 386 SX comercial (modificada), la cual está montada sobre otra tarjeta en la cual se agregaron diversos dispositivos electrónicos, entre ellos algunos para realizar la detección de operaciones relacionadas con transferencias a memoria y accesos a puertos, la cual será discutida en otro trabajo de tesis. Por lo anterior, a lo largo de este trabajo a los procesadores se les verá exclusivamente como la fuente que provee la información a compararse. En la figura 2.3 aparece la UP ensamblada.

2.5 Tarjeta de aislamiento de unidades de detección de fallas y control (TAUDFCs)

Al igual que las TAUPs, las TAUDFCs controlan el paso de señales provenientes de UDFCs hacia los ductos, de UDFC principal hacia UDFC redundante y de los demás subsistemas hacia la UDFC principal. Pero a diferencia de las TAUPs las TAUDFCs necesitan un proceso de inicialización, el cual será tratado en el capítulo cuatro. Dicha inicialización fija a una de las dos UDFCs como principal para que tome el mando de la arquitectura de la CTF y para que la otra actúe

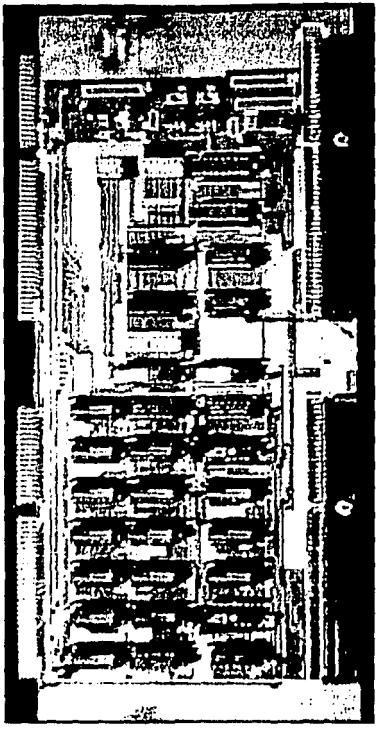


Figura 2.2 Tarjeta de Aislamiento de Unidad de Procesamiento (TAUP).

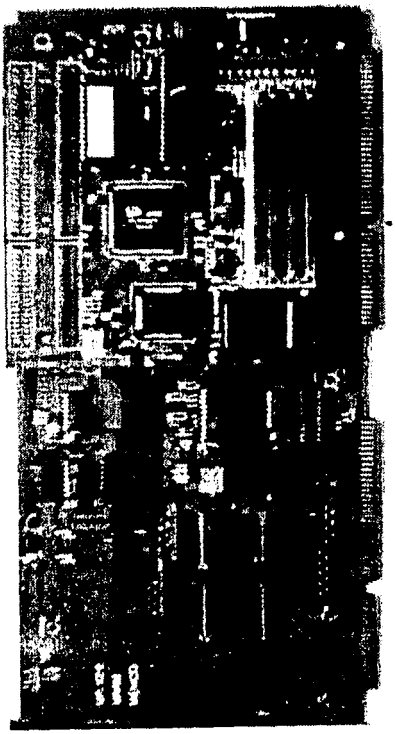


Figura 2.3 Tarjeta de Procesador (UP).

como redundante en espera de tomar el mando de la arquitectura en caso de requerirse. Además, en este período de inicialización se prevé la detección de errores en la asignación de categorías a UDFCs. En la figura 2.4 se muestra la TAUDFC ensamblada y con componentes.

2.6 Unidad de detección de fallas y control (UDFCs)

Esta tarjeta es sin duda el subsistema de mayor importancia en la CTF ya que es la encargada de realizar la comparación de datos provenientes de las UPs, la detección de fallas y la reconfiguración de la arquitectura de la computadora. Todo esto en tiempo real y de manera totalmente transparente para el usuario, logrando así otros de los atributos descritos en el capítulo uno, tales como la autoreconfiguración y la operación continua del sistema aun ante la presencia de fallas en alguno de los subsistemas. Además, el diseño de la UDFC contempla circuitos vigía encargados de la supervisión de la electrónica contenida en la misma tarjeta, la cual se discutirá en el siguiente capítulo.

La validación de la electrónica contenida en esta tarjeta así como la descripción de su comportamiento y la puesta en marcha de la misma, interactuando con los demás subsistemas, se verán en los siguientes capítulos. En la figura 2.5 se muestra a la UDFC ensamblada y con componentes.

2.7 Unidad de interfaz múltiple (UIM)

Esta tarjeta es el subsistema más pequeño de la arquitectura de la CTF, ella contiene electrónica que permite interaccionar con el usuario de forma totalmente dedicada e independiente de las UPs instaladas. A través de ella se permitirá que el usuario solicite algún tipo de cambio o de reconfiguración en el sistema ya sea para labores de mantenimiento o para propósitos de demostración. La UIM se comunica con la UDFC principal a través del bus TF y por medio de cable plano se le conectan : un teclado de membrana, una pantalla de cristal líquido, la bocina y el reset compartido que maneja la CTF; todos ellos alojados en el gabinete de la computadora. En la UIM se tienen cuatro CIs de baja escala de integración por lo que es el único subsistema que no posee redundancia ya que se considera que sus componentes tienen poca probabilidad de faltar.

El diseño de la UIM permite que ambas UDFCs puedan manejar sus periféricos externos (en caso de alguna conmutación de UDFC) sin tener que realizar conexiones externas, esta característica es otra cualidad que el diseño de la CTF ofrece en su arquitectura.

De lo anterior se desprende que debe existir alguna forma de saber si la UIM está activa y lista para interactuar con el usuario y la arquitectura. Para dar solución a

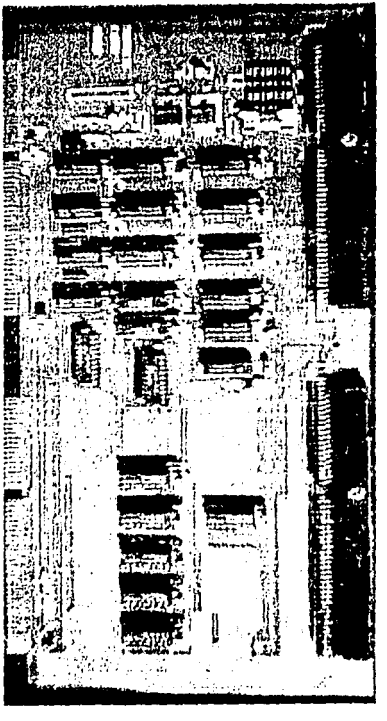


Figura 2.4 Tarjeta de Aislamiento de UDFC (TAUDFC).

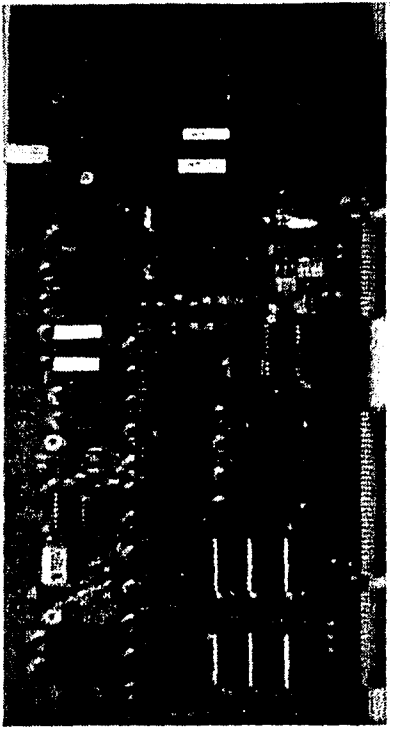


Figura 2.5 Tarjeta de Unidad de Detección de Fallas y Control (UDFC).

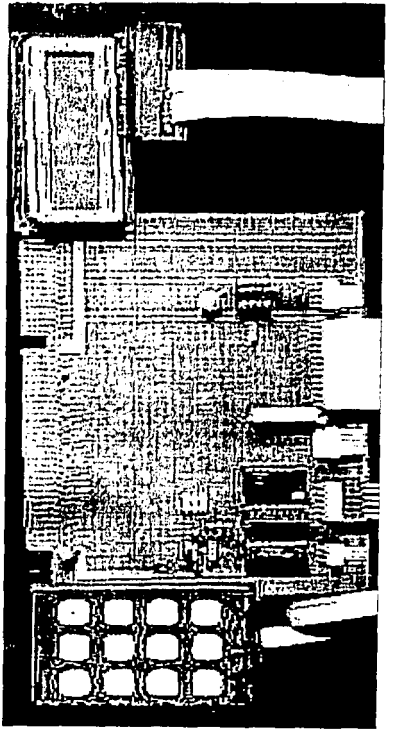


Figura 2.6 Unidad de Interfaz Múltiple (UIM).

este aparente problema, en la programación de las rutinas de control de la UDFC se contempla un despliegue intermitente de puntos en el módulo LCD; que indicará que la UIM opera correctamente y que está lista para desempeñar sus labores encomendadas, esto se verá en el capítulo cinco. En la figura 2.6 aparece la UIM ensamblada y con componentes.

2.8 Ductos utilizados

El comportamiento externo de la CTF con todas sus tarjetas integradas, es similar al de una computadora personal convencional, pero con atributos de auto diagnóstico y de autoreconfiguración en caso de fallas. Debido a esto la CTF es compatible en circuitería y en programación con la amplia disponibilidad de periféricos y de paquetes de programación que hay en el mercado. Para lograr estas dos importantes características (compatibilidad y tolerancia a fallas) la CTF utiliza dos ductos, los cuales conforman la columna vertebral del sistema. Uno de los ductos es el PC-AT y el otro es un ducto propietario denominado TF.

2.8.1 Ducto PC-AT

Este ducto es completamente compatible con el estándar ISA, el cual contiene 24 líneas de direcciones, 16 de datos y líneas de control y protocolo, ver Figura 2.7. Para mayores detalles sobre el estándar ISA ver [Solari 1991].

2.8.2 Ducto TF

En los apartados anteriores se pudo ver que para el correcto funcionamiento de la CTF existen protocolos y líneas de comunicación entre los diferentes módulos que constituyen a la CTF, para que esto se logre de manera eficiente y sin tener problemas de ruido o colisiones entre estas señales, se diseñó un ducto totalmente propietario el cual además de contener las señales de protocolo incluye señales que controlan las zonas de contención de fallas. El diseño de este ducto benefició enormemente a la modularidad de la CTF ya que de no haber sido implementado, los protocolos de comunicación entre las tarjetas, el diseño electrónico y la programación se habrían complicado de sobremanera.

Entre las señales más importantes del ducto TF están 16 líneas para el multiplexaje de datos desde UPs hacia UDFCs, líneas de estado y categoría para UPs, señales de protocolo entre autómatas, señales de teclado y bocina, así como líneas de datos y control para el despliegue alfanumérico y teclado matricial utilizados en tareas de mantenimiento. También se encuentran en este ducto las líneas de protocolo y control entre UDFCs, que permiten la sustitución automática

de una UDFC anómala por su contraparte redundante en buen estado y, finalmente, líneas de control entre TAs y UDFCs. En la figura 2.8 se muestran las líneas que conforman al ducto TF.

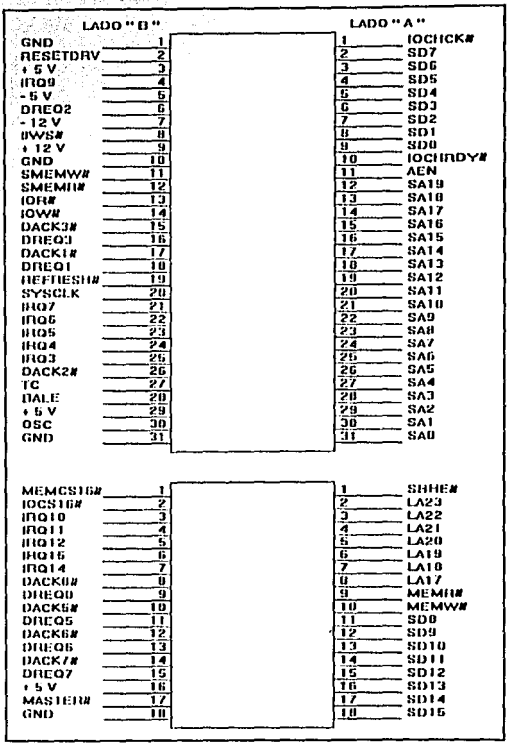


Figura 2.7 Terminales del Ducto PC-AT.

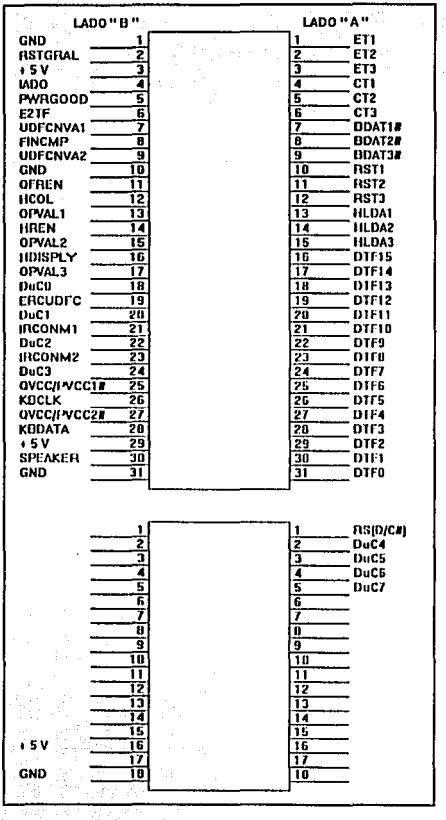


Figura 2.8 Terminales del Ducto TF.

Como se mencionó anteriormente existen TAs dedicadas para UPs y para UDFCs. Las TAs se conectan directamente al ducto TF, pero no todas las señales contenidas en él llegan a las UPs o a las UDFCs. Dentro de las TAs se generan diferentes señales para UPs y para UDFCs, de ello se desprende la existencia de dos ductos intermedios que denominamos:

- **Ducto TA-UPs.**
- **Ducto TA-UDFCs.**

Estos ductos conducen líneas de datos y control específicas para cada tarjeta objetivo. En las Figuras 2.9 y 2.10 se muestra la disposición de las terminales del ducto TA-UPs y el ducto TA-UDFCs, respectivamente.

A continuación se da una breve descripción de las señales que integran el ducto propietario:

IADO Inicio del Autómata de Detección de Operaciones. Inicializa al autómata que se encarga de decodificar señales que indican lectura o escritura a puerto o memoria.

PWRGOOD Power Good. Señal proporcionada por la fuente de alimentación que llega a la unidad manejadora del ducto AT.

E2TF Existen dos Tarjetas con Falla. Línea necesaria para proporcionar dicha información al Autómata de Detección de Operaciones.

FINCMP Fin de Comparación. Indica el final del proceso de comparación de los datos multiplexados.

QFREN Quita Freno. Señal que libera a los procesadores (cuando se encuentran detenidos) después de efectuarse la comparación de datos en la UDFC.

OPVALi Operación Válida de la UPi. Indica que se efectuó una lectura o escritura en la iésima UP.

KBCLK Keyboard Clock. Señal de reloj del teclado.

KBDATA Keyboard Data. Señal de datos del teclado.

SPEAKER Bocina

ETi Estado de la Tarjeta i. Señala el estado (encendido o apagado) de la tarjeta correspondiente.

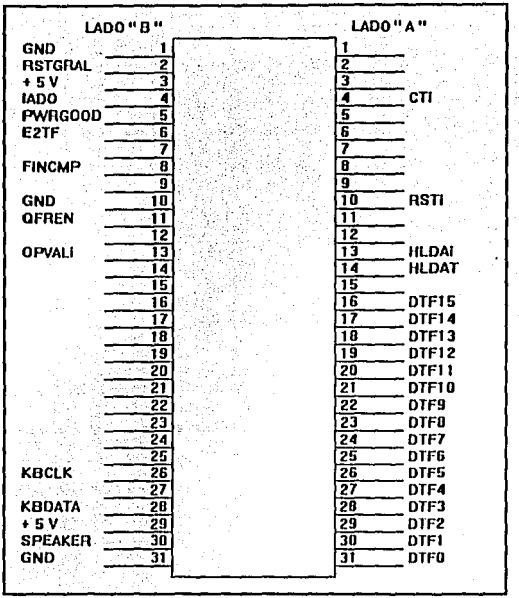


Figura 2.9 Terminales del Ducto TA-UPs.

CTI Categoría de la Tarjeta i. Indica la categoría (principal o redundante) de la UP correspondiente.

BDATI Habilitador de Datos Multiplexados. Habilita las salidas del *buffer* i de la UP i para multiplexar los datos.

RSTI Reset i. Es la línea de RESET para la UP i.

HLDAI Hold Acknowledge i. Línea de reconocimiento de DMA en la UP i.

DTF0-15 Ducto de datos multiplexados por comparar.

UDFCNVA1,2 UDFC Nueva 1 y 2. Señal que indica a la UDFC redundante que la principal falló.

HCOL Habilita Columna. Señal que habilita la columna del teclado de la UIM.

HREN Habilita Renglón. Señal que habilita el renglón del teclado de la UIM.

HDISPLY Habilita Display. Señal para habilitar la pantalla de cristal líquido de la UIM.

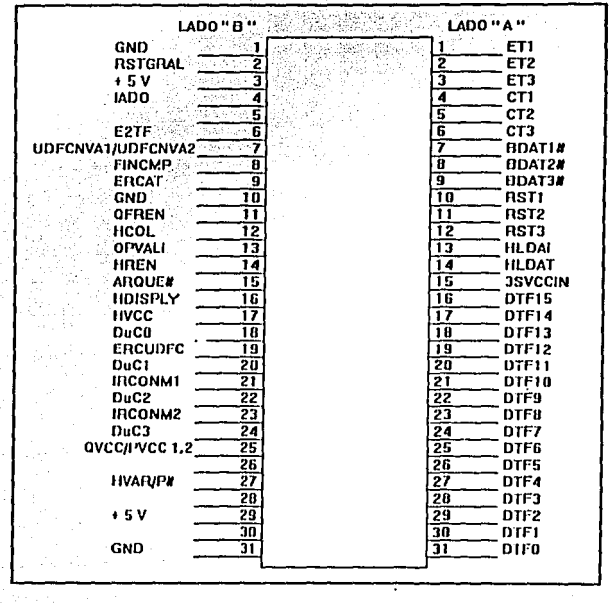


Figura 2.10 Terminales del Ducto TA-UDFCs.

DuCO-7 Ducto de datos de la UIM.

ERCUDFC Error de Categoría en UDFC. Indica un error en la asignación de categoría de las dos UDFCs instaladas cuando arranca el sistema.

IRCONM1,2 Interrupción de Conmutación. Cuando falla la UDFC principal, esta señal indica a la UDFC redundante que debe iniciar operaciones.

QVCC/PVCC1,2# Quita Vcc/ Pon Vcc 1 y 2. Señal que se encarga de conmutar los voltajes de alimentación de las UDFCs, en caso de que se requiera mantenimiento en línea.

3SVCCIN Tres Segundos de Vcc In. Señal que permite el suministro de energía a las UDFCs durante tres segundos para que se autoconfiguren al arrancar el sistema.

ERCAT Error de Categoría en UPs. Línea que se activa en caso de detectarse dos o más UPs declaradas como principales.

ARQUE Arranque. Señal que permite detectar en las UDFCs un arranque en frío o en tibio.

HVCC Habilita Vcc. Esta señal habilita el voltaje de alimentación de las UDFCs.

RS(D/C#) Habilitador que indica transferencia de datos o comando de control para la pantalla de cristal líquido.

HUAR/P# Habilitador de unidades de aislamiento que define su operación como redundante o como principal.

CAPITULO TRES

ARQUITECTURA DE LA UNIDAD DE DETECCIÓN DE FALLAS Y CONTROL (UDFCs) DE LA CTF

3.1 Introducción

En el capítulo anterior se describieron a *grosso modo* los subsistemas que integran a la CTF y se hizo mención de la interacción entre UPs y UDFCs para poder detectar fallas, realizar diagnóstico y si es necesario reconfigurar la arquitectura.

En este capítulo se hará una descripción más extensa de la electrónica que compone a las UDFCs, destacando aspectos como su inicialización. Otro de los aspectos importantes que se discutirán en este capítulo es la operación de los autómatas de alta velocidad, los cuales pueden operar a 2, 4, 8 ó 16 MHz, cuyas responsabilidades comprenden el multiplexaje de datos, la comparación de información y la toma de alguna decisión en la configuración de la arquitectura.

También se explicará el funcionamiento del microcontrolador (μC) integrado en la tarjeta y sus registros asociados lo cual se complementará con la programación de las rutinas de control de la UDFC que se verá en el capítulo cinco.

Los demás apartados del capítulo hablarán sobre los protocolos de comunicación entre UPs y UDFCs para la correcta puesta en marcha del sistema TF y por último se describe superficialmente el diseño del circuito impreso de la UDFC.

3.2 Inicialización del sistema tolerante a fallas

Para el buen funcionamiento de la CTF se debe realizar un procedimiento de inicialización de algunos subsistemas que integran a la CTF, dicha inicialización tiene como objeto definir cual de las dos UDFCs será la que lleve el control de la arquitectura (UDFC PRINCIPAL) y cual será su contraparte (UDFC REDUNDANTE) la cual permanece en estado de espera hasta que se requiera que tome el control de la arquitectura.

El diseño de la UDFC integra interruptores programables por el usuario los cuales se deben fijar antes del encendido de la CTF (switch S1); dichos interruptores programan la categoría de la UDFC correspondiente e indirectamente la categoría de la unidad de aislamiento, ver tabla 3.1.

Switch S1			Categoría de UDFC y TAUDFC	PUERTOS	DIRECCIONES	
SW3	SW2	SW1				
ON	ON	ON	PRINCIPAL	1	LECRA#	0300 H - 1 0308 H - 2 0310 H - 3 0318 H - 4
ON	ON	OFF		2	RDRE#	0301 H - 1 0309 H - 2 0311 H - 3 0319 H - 4
ON	OFF	ON		3	FINITA#	0302 H - 1 030A H - 2 0312 H - 3 031A H - 4
ON	OFF	OFF		4	LDRER	0303 H - 1 030B H - 2 0313 H - 3 031B H - 4
OFF	ON	ON	REDUNDANTE	1		
OFF	ON	OFF		2		
OFF	OFF	ON		3		
OFF	OFF	OFF		4		

Tabla 3.1 Posición de interruptores y Actividad en la UDFC.

Durante el arranque del sistema el microcontrolador contenido en cada UDFC examina su categoría y posteriormente establece las variables requeridas para la inicialización de las unidades de aislamiento.

Como quizás se pueda comprender, de acuerdo a la explicación anterior, existe un breve lapso de tiempo durante el inicio del sistema en que ambas TAUDFCs deben permitir el paso de señales provenientes del ducto pasivo para que éstas puedan llegar a las tarjetas objetivo y de esta forma se pueda realizar su inicialización. Para dar seguridad al equipo, durante este lapso de tiempo ambas tarjetas operan como redundantes.

El tiempo estimado en el que ambas UDFCs serán redundantes, es aproximadamente de 3 segundos, en este lapso cada UDFC lee su categoría y genera la lógica de control de su respectiva TA, este detalle de diseño se ampliará en el próximo capítulo y la rutina de inicialización se describirá en el capítulo cinco.

Además, como se mencionó en el capítulo dos, durante los tres segundos se puede detectar algún error en la asignación de categorías de las UDFCs (cuando por error el usuario programe los interruptores de ambas UDFCs como principales).

3.3 Arquitectura de UDFCs

Para facilitar la descripción de la electrónica que compone a las UDFCs, ésta se dividió en dos bloques, tal como se muestra en el diagrama de bloques de la figura 3.1.

El primer bloque lo constituyen los autómatas que se encargan de realizar labores de detección, diagnóstico y reconfiguración (si es necesaria) de la arquitectura de la CTF.

El segundo bloque lo constituye el µC 68HC11E1, su "latch" para demultiplexar datos y direcciones, EPROM de 8 Kbytes y electrónica adicional para poder comunicarse con los demás subsistemas de la CTF e interaccionar con el usuario en labores de mantenimiento y envío de alarmas cuando se presenten fallas en el sistema. En la figura 3.2 se muestra un diagrama de bloques de la electrónica que rodea al 68HC11, en la figura 3.3 se muestra el diagrama electrónico de la UDFC.

3.4 Sección de autómatas

La sección de autómatas se encarga de multiplexar los datos provenientes de las UPs instaladas así como de comparar los diversos datos para detectar fallas, las que de suceder, se diagnostican para efectuar las reconfiguraciones necesarias obteniendo así la operación continua del sistema. Su electrónica está compuesta por dos autómatas que operan a 2, 4, 8 ó 16 MHz, ambos vigilados por electrónica dedicada para detectar sus posibles fallas, las que de presentarse, obligan por medios electrónicos la conmutación y delegación de la responsabilidad de control hacia la UDFC redundante.

3.4.1 Autómata de Multiplexaje

La filosofía de diseño de este autómata consiste en detectar datos por comparar provenientes de 3 UPs, contiene además "flip-flops" que se encargan de retener los datos a comparar y electrónica para decodificar salidas de control para encadenar el inicio de operaciones del siguiente autómata de control.

De lo anterior se desprende que debe existir un protocolo entre el autómata de detección de operaciones (ADO) contenido en cada UP y el autómata de Multiplexaje de datos (AMD) de la UDFC principal; esta comunicación se efectúa por medio de las líneas OPVAL 1,2 y 3 que se describieron en el capítulo dos. En la figura 3.4 aparece el autómata de multiplexaje de datos y en la figura 3.5 aparecen los GALS que se emplearon para la implementación del mismo, junto aparecen las ecuaciones que describen la lógica de control del AMD.

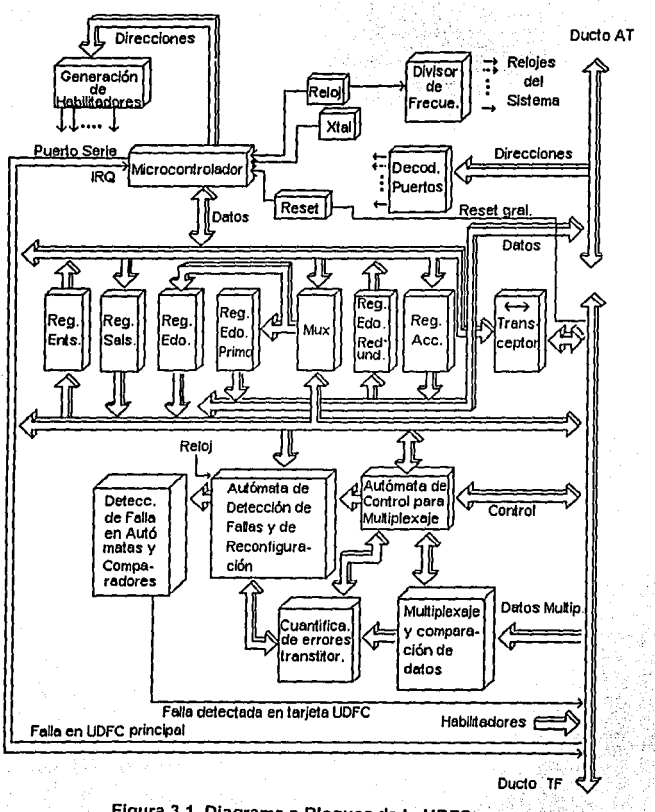
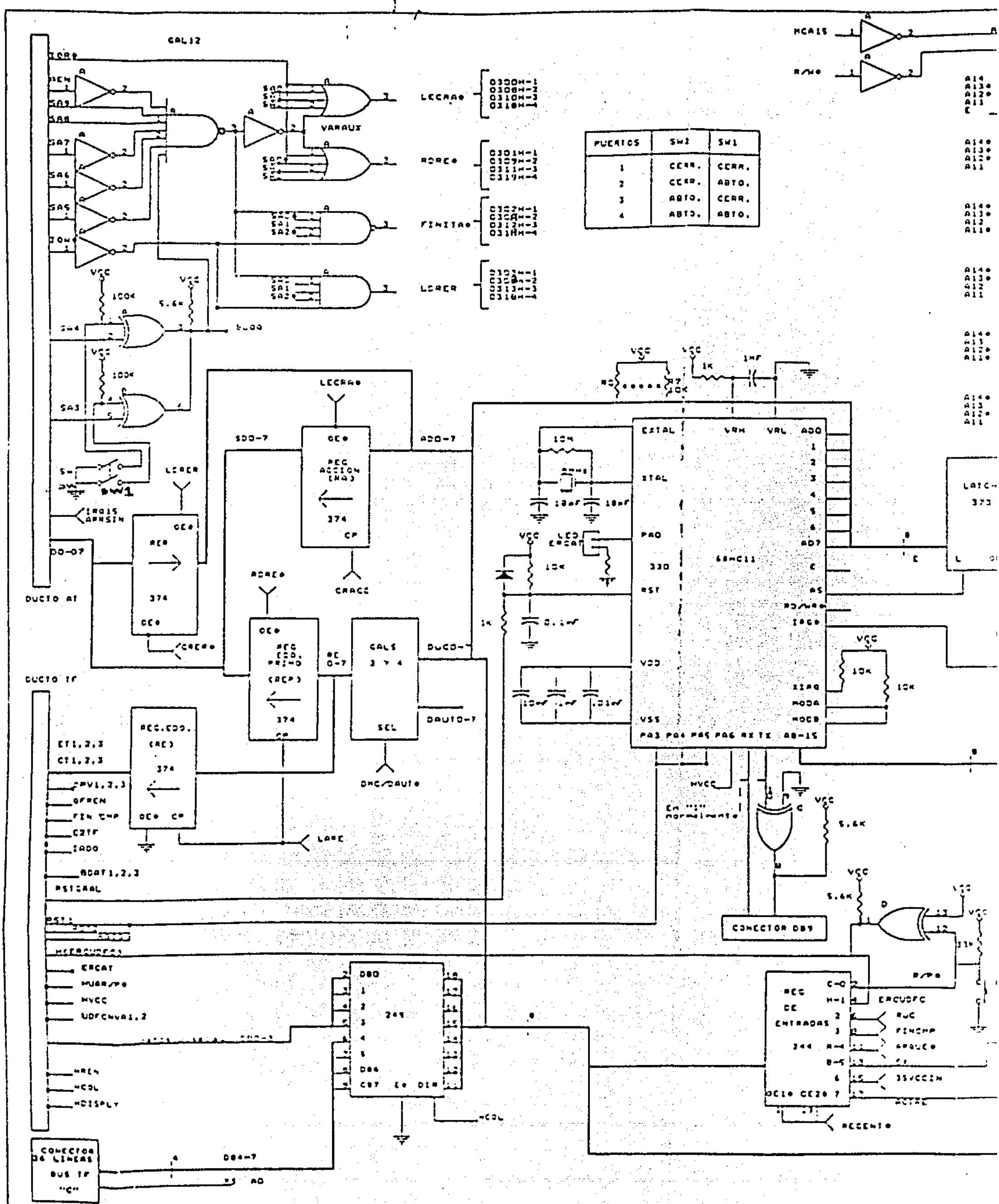
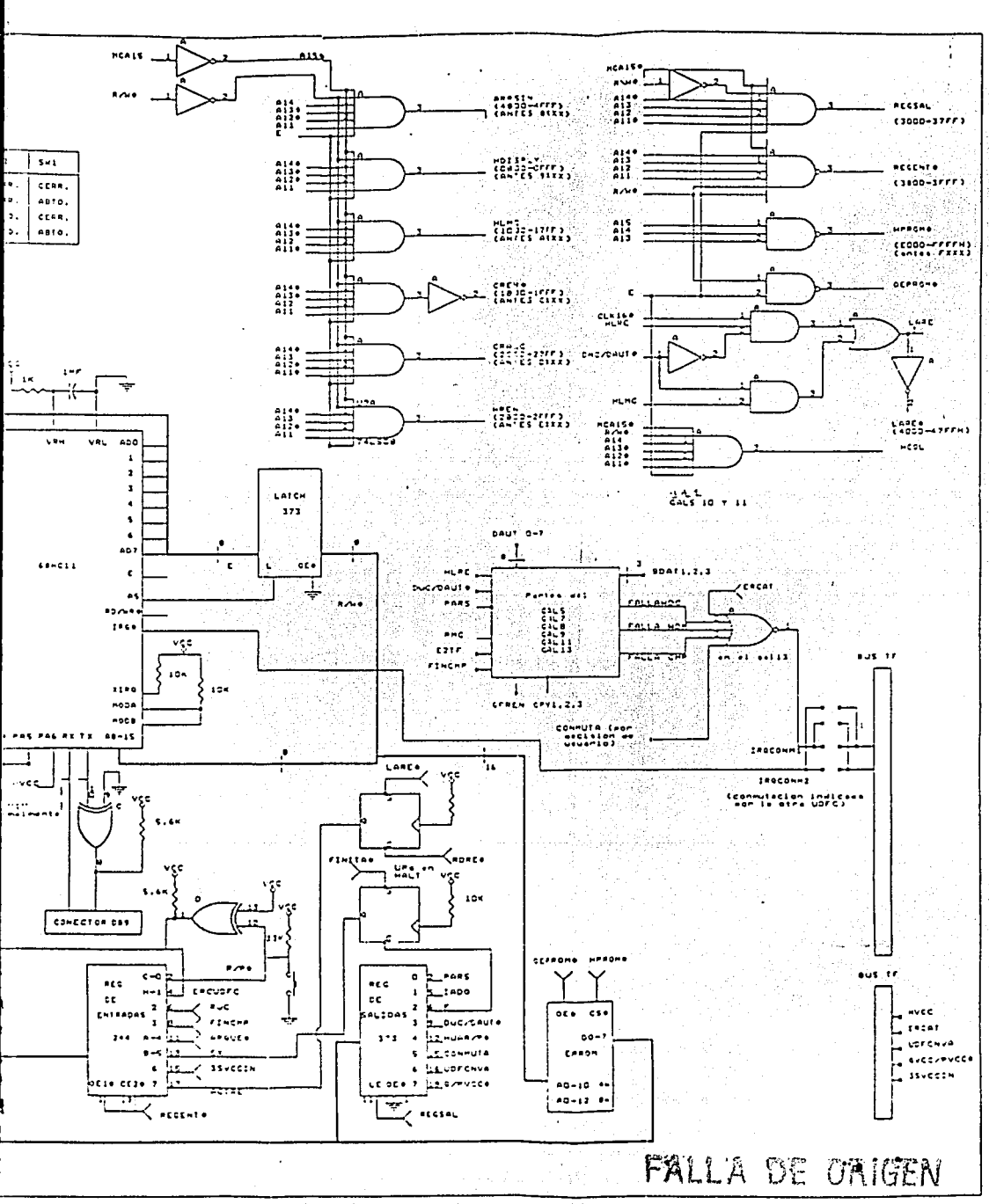


Figura 3.1 Diagrama a Bloques de la UDFC.

Figura 3.2 Diagrama a Bloques de la Electrónica del 68HC11.



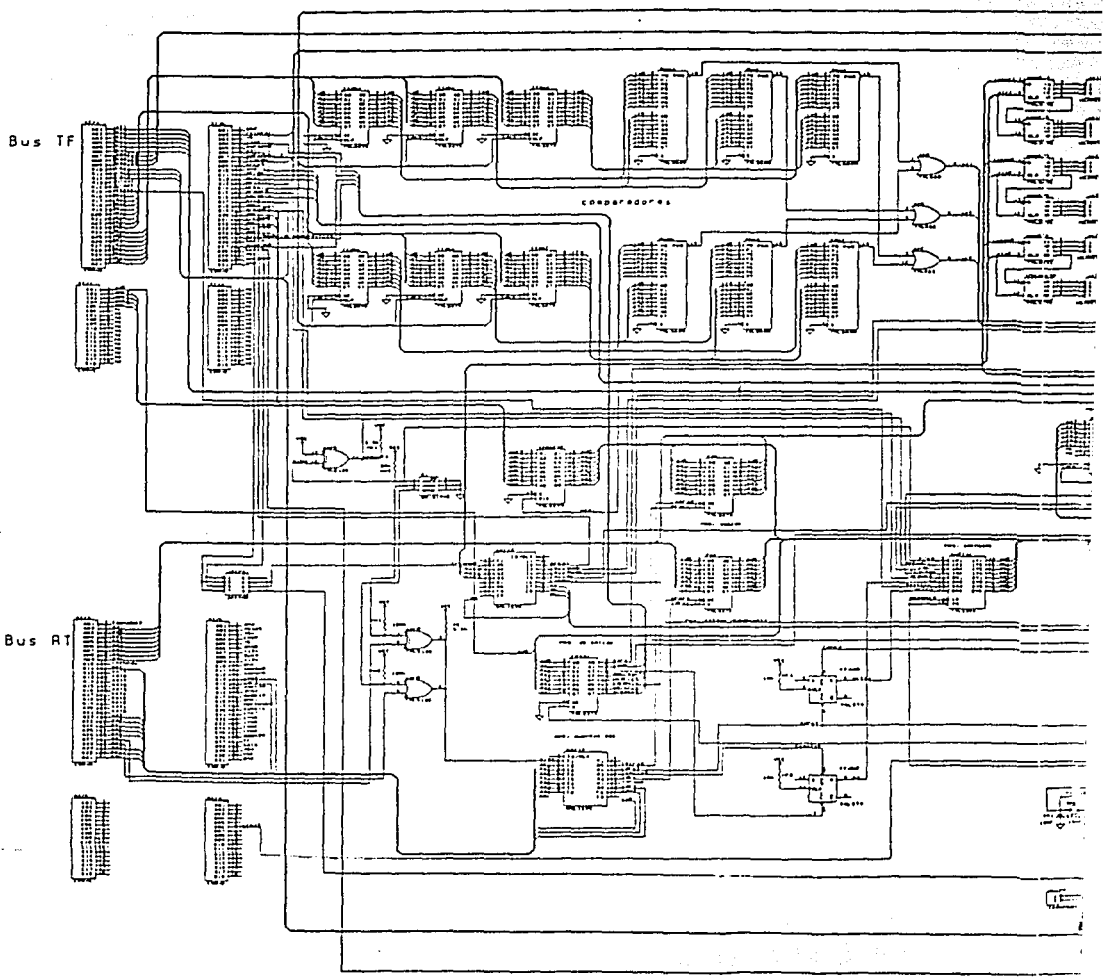
GAL12
LECRA*
ADRE*
FINITA*
LDREP
VARAUX
0300H-1
0308H-2
0310H-3
0318H-4
0301H-1
0309H-2
0311H-3
0319H-4
0302H-1
0312H-3
0303H-1
0313H-3
0318H-4
PUERTOS | SW2 | SW1
1 | CERR. | CERR.
2 | CERR. | ABTO.
3 | ABTO. | CERR.
4 | ABTO. | ABTO.
VCC
100K
5.6K
SW1
IRQ15
DUCTO AT
DUCTO IF
D0-7
SD0-7
AD0-7
REG ACCION (RA)
374
REG EDO. PRIMO (REP)
GALS 3 Y 4
SEL
DAUTO-7
DNC/DAUTO
REG.EDO. (RE)
LAPE
CRACC
10K
1K
0.1uF
18pF
68HC11
EXTAL
XTAL
VRH
VRL
AD0
AD7
AS
RD/WR
IRQ*
XIRQ
MODA
MODB
VDD
VSS
RST
LATCH
373
CONECTOR DB9
REG DE ENTRADAS
244
245
DB0
DB7
DIR
HCOL
HREN
HDISPLY
HVCC
CONECTOR 26 LINEAS BUS IF
DB4-7
A0
ENCAT
RECENT*
RSTGRAL
FIN CMP
SPREN

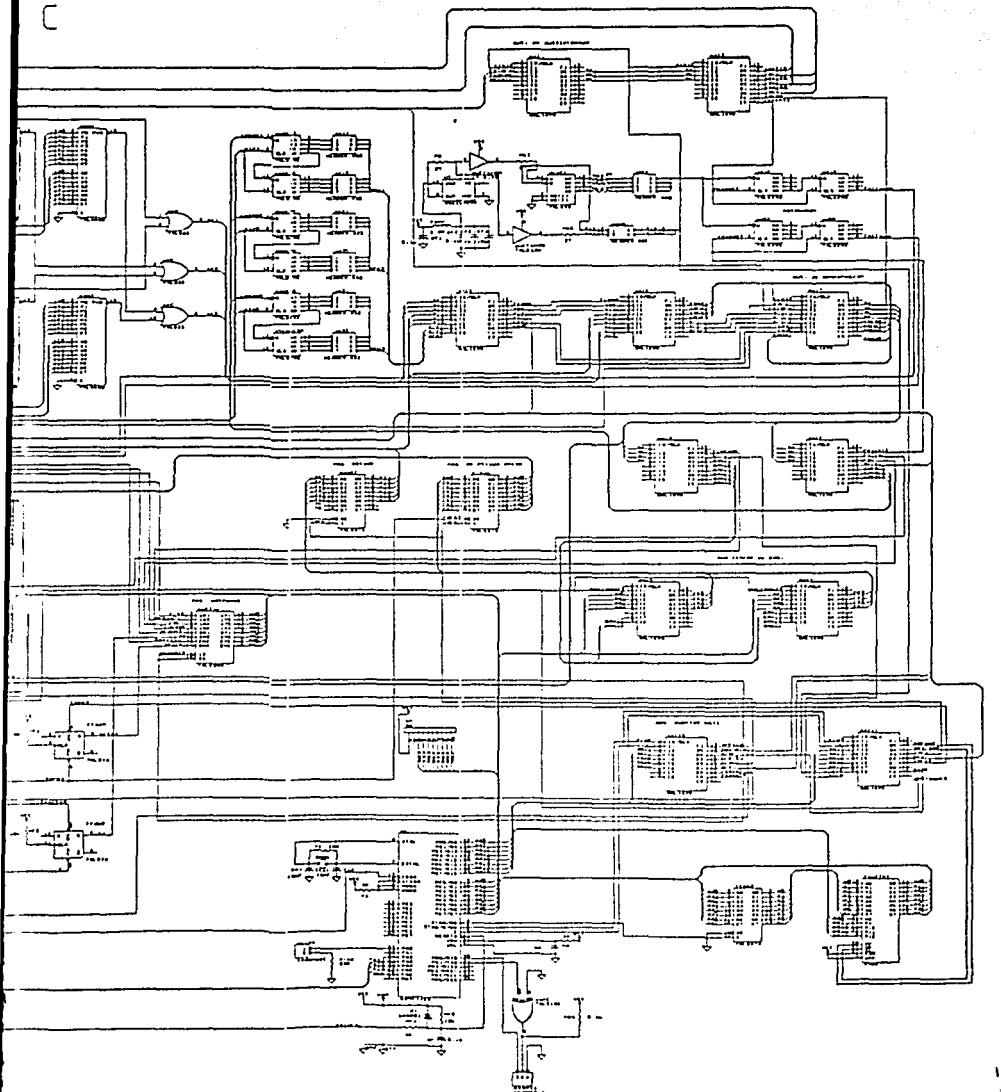


FALLA DE ORIGEN

Figura 3.3 Diagrama Electrónico de la UDFC.

U D F C





FALLA DE ORIGEN

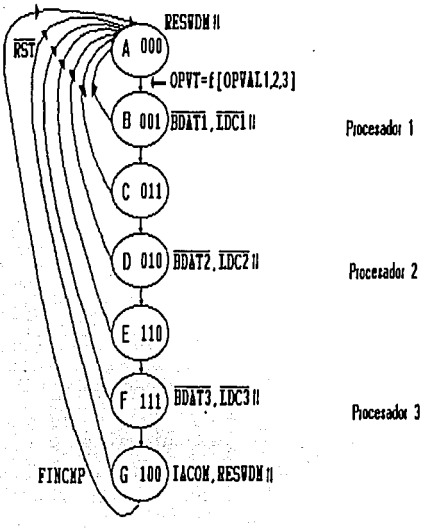


Figura 3.4 Autómata de Multiplexaje.

El AMD inicia operaciones cuando el ADO de cada UP ha detectado una operación válida, una vez satisfecha esta condición ordena la liberación del dato de alguna UP y procede a retenerlo (mediante las señales BDATi# y LDCi#). Esta operación se repite tres veces independientemente del número de UPs instaladas. Cuando una UP no se encuentra en buen estado, o bien, cuando por alguna razón se desinstala alguna de ellas la operación de este autómata no sufre alteración lo cual no afecta los resultados del voteo. En este caso quién toma las decisiones de comparación y analiza los resultados es el Autómata de Comparación de Datos y Reconfiguración (ACDR) el cual prevé las diferentes combinaciones de procesadores instalados; este autómata se describirá más adelante.

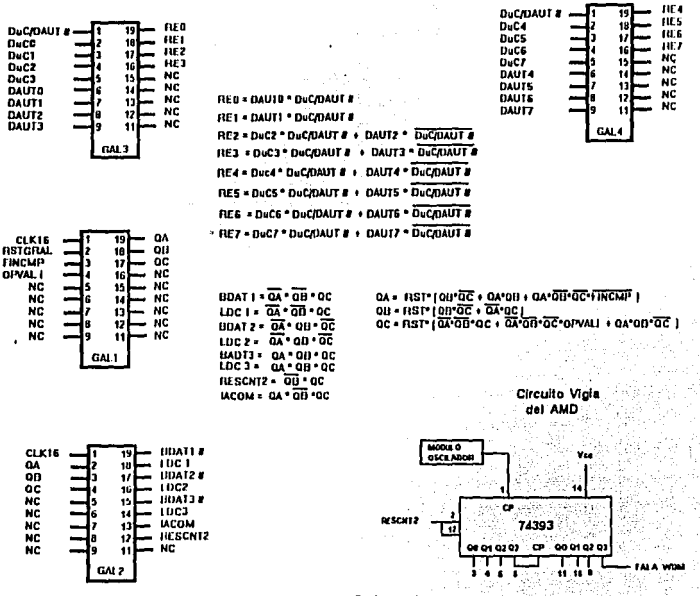


Figura 3.5 Banco de Gals del AMD.

Una vez finalizadas las transferencias de datos de las UPs se envía la señal IACOM la cual activa al ACDR y permanecerá en ese estado hasta que el ACDR le mande la señal FINCMP la cual es generada después de haber analizado el voteo y realizado los cambios necesarios (en caso de requerirse).

Como se dijo en el capítulo dos las UDFCs cuentan con electrónica dedicada para detectar fallas aun en su propia electrónica. Debido a que este autómata es el interlocutor entre UPs y UDFC principal se le asignó un circuito vigía quién detecta alguna falla en el AMD y en caso de existir enviará las señales pertinentes para conmutar de UDFC (todo esto de manera transparente para el usuario).

El principio de funcionamiento de este vigía consiste en programarle en el estado "A" el tiempo máximo permisible de operación del autómata, éste tiempo es variable (2, 4, 8 ó 16 MHz.) y puede ser seleccionado por el usuario mediante el *jumper* "jpfr" (ver figura 3.3). En caso de sobrepasar el tiempo seleccionado, indicará que el AMD no realizó el Multiplexaje de datos y por lo tanto gestiona las señales para conmutar de UDFC. El circuito vigía genera el pulso FALLWDM (que podrá aparecer a los 7.8, 16, 32 ó 62.5 Khz., aproximadamente, dependiendo de la selección hecha) cuando la electrónica que compone al autómata deja de operar por alguna razón, ver figura 3.5.

Los GALS G1 y G2 de la figura 3.5 conforman el AMD, mientras que G3 y G4 forman el multiplexor para determinar quién cambiará (el usuario o el ACDR) los bits de estado y categoría.

3.4.2 Autómata de Comparación de Datos y Reconfiguración

La técnica utilizada por la CTF para detectar fallas es por voteo mayoritario, por lo cual se requieren capturar datos, compararlos y posteriormente se necesita generar un dictamen del resultado.

Para la captura de los 16 bits de datos, de cada UP, se utilizaron 6 circuitos 74LS374, los que se agruparon en 3 pares para poder capturar los 16 bits, ver figura 3.3. Para la comparación de datos provenientes de hasta 3 UPs se utilizaron 6 CIs 74LS688, los que aparecen en la misma figura.

Con el fin de poder entender mejor los procesos de detección y diagnóstico de fallas nos apoyaremos en la figura 3.6. La cual presenta tres características importantes, las cuales se describen en los párrafos siguientes.

3.4.3 Detección de fallas en UPs

La electrónica puede detectar un dato anómalo de cualquier UP y determinando con exactitud cual procesador generó la falla.

En el circuito mostrado cuando se presenta un dato anómalo de cualquier UP se puede detectar tanto la falla como la UP que la generó. Por ejemplo, en caso de presentarse un error en la UP1, las salidas SC1 y SC3 del circuito presentan niveles altos en tanto que SC2 presenta un nivel bajo; cuando se tiene este tipo de relación se busca el factor común entre las líneas que generan niveles altos. En nuestro ejemplo, SC1 se deriva de las comparaciones de datos entre UP1 y UP2 mientras que SC3 depende de la comparación de datos entre UP1 y UP3. De lo que se deduce que la falla proviene de la UP1, que es el factor común.

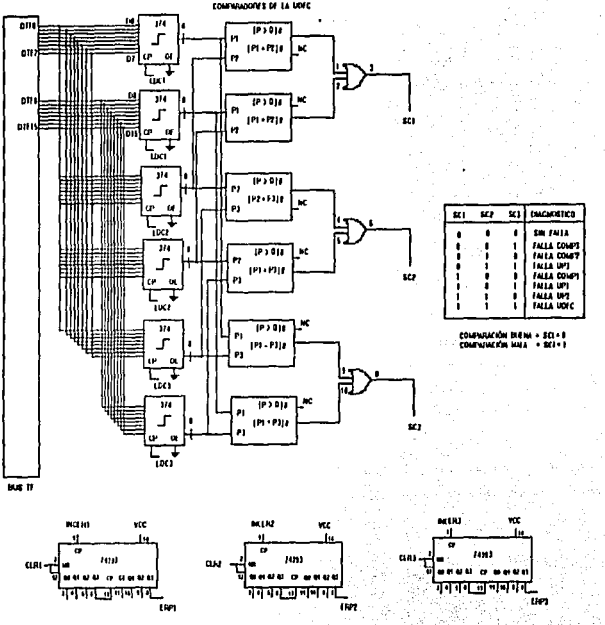


Figura 3.6 Registros de Captura y Comparadores utilizados por el ACDR.

3.4.4 Detección de fallas en comparadores

La electrónica también puede detectar fallas en los comparadores de la UDFC o en sus registros asociados.

Un detalle importante del circuito es que permite la identificación de fallas en los mismos comparadores pues como se mencionó arriba, la falla de alguna UP se manifiesta en dos de las tres salidas SC1, SC2 y SC3; por lo cual de presentarse

un solo nivel alto en alguna de las señales anteriores indicaría la presencia de una falla en alguno de los comparadores o en los registros vinculados a ellos.

La tercer característica tiene por objeto cuantificar el número de fallas detectadas en cada UP, ésta labor corre a cargo de los CIs 74LS393; estos circuitos se añadieron al diseño con la finalidad de detectar errores transitorios y de evaluar sus repercusiones en diferentes aplicaciones.

Como se vio en el capítulo uno, una de las características de la arquitectura de la CTF consiste en poder autoreconfigurarse ante la presencia de fallas en cualquiera de sus subsistemas, de manera transparente para el usuario y en tiempo real. De ello se desprende que el detectar fallas y diagnosticar el origen de la misma NO serviría de nada si no se tomara alguna decisión sobre el estado actual de la arquitectura de la CTF. Para lograr estos atributos en el ACDR se implantaron una serie de decisiones para la reconfiguración de la CTF. Estas decisiones aparecen en forma de árbol en la figura 3.7.

Los caminos que puede seguir el comportamiento de la arquitectura de la CTF, en base al árbol de decisiones, son básicamente tres:

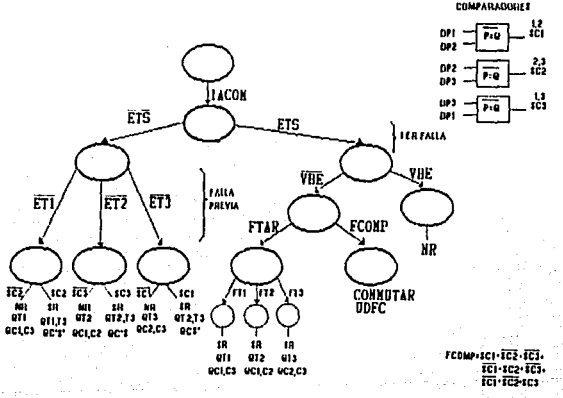


Figura 3.7 Árbol de Decisiones del ACDR.

a) No se detecto falla en el sistema, en este caso la ruta de decisiones es a través de las señales:

IACOM (Inicializa al ACDR) — ETS (estado total del sistema bueno) — VBE (voteo en buen estado) y NR (no hay reconfiguración).

b) Detección de la primera falla en UPs o en Comparadores:

b.1. Falla en UPi, en este caso la ruta de decisiones es a través de las señales:

IACOM — ETS — VBE # (voteo malo) — FTAR (falla alguna UP) y dependiendo de cual UP fallo seguirá por :

FT1 (falla UP1) teniendo como consecuencia el cambio de categoría y contabilizar, el numero de fallas transitorias que de alcanzar el máximo número permisible de errores se procederá a desactivar a UP1, seguido del envío de alarmas al usuario.

FT2 (falla UP2) similar al punto anterior sólo que la categoría principal se cede a UP1 o UP3.

FT3 (falla UP3) similar al punto anterior solo que la categoría principal se cede a UP1 o UP2.

b.2. Falla un Comparador, en este caso la ruta de decisiones es a través de las señales:

IACOM — ETS — VBE # — FCOMP (falla comparador), teniendo como consecuencia la conmutación de UDFC.

c) Detección de una segunda falla sin que se haya restaurado la primera; en este caso la ruta de decisiones es a través de las señales:

IACOM — ETS # (estado total del sistema malo) y dependiendo de cual UP se desactivó se siguen las siguientes rutas:

c.1. ET1 # (UP1 apagada) y dependiendo del resultado de la comparación de datos entre UP2 y UP3 podrá seguir por dos rutas, a saber :

Si SC2 es alto (comparación mala) esto nos llevaría a desactivar a UP1 y a UP2.

si SC2 es bajo (comparación exitosa) no hay reconfiguración.

c.2. ET2 # (UP2 apagada) similar al punto anterior solo que si SC3 es alto desactivamos a UP2 y a UP3.

c.3. ET3 # (UP3 apagada) similar a los anteriores solo que si SC1 es alto desactivamos a UP1 y a UP3.

En los párrafos anteriores se explicaron a grandes rasgos las reglas de detección de fallas y de reconfiguración. En la figura 3.8 aparece el ACDR y en la figura 3.9 aparecen los GALS en los que se implementaron las ecuaciones para el ACDR, junto a los GALS aparecen las ecuaciones lógicas del autómata.

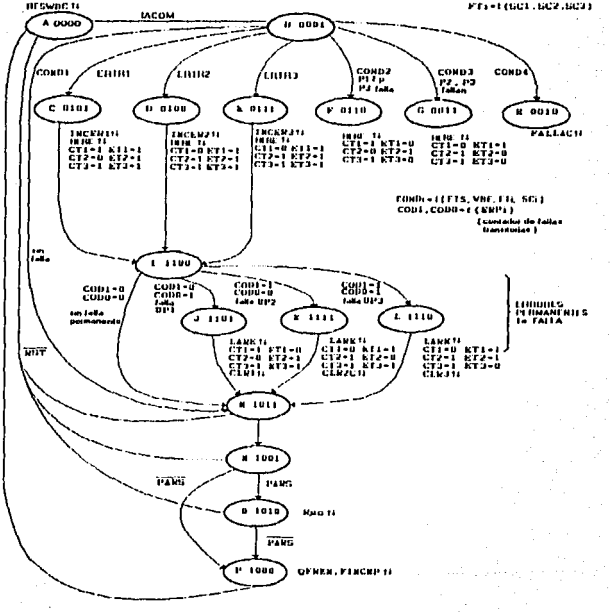
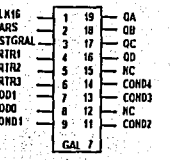
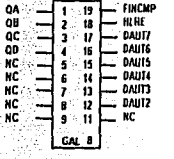
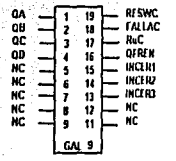
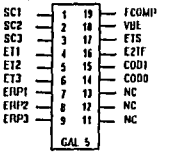


Figura 3.8 Autómata de Comparación de Datos y Reconfiguración (ACDR).









$QA = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COND1\cdot RST + QA\cdot\overline{QC}\cdot RST + \overline{QB}\cdot RST + QC\cdot\overline{QD}\cdot RST + QA\cdot\overline{QD}\cdot RST$

$QB = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QD}\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COD1\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COD0\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot ERTR1\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot ERTR2\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot ERTR3\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot COND2\cdot RST$

$QC = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot RST + QA\cdot\overline{QB}\cdot\overline{QD}\cdot RST + QB\cdot\overline{QC}\cdot\overline{QD}\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot PARS\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot PARS\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COD1\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot\overline{COD1}\cdot COD0\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot COND1\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot ERTR3\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot COND2\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COND1\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COND4\cdot RST$

$QD = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot ACOM\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot\overline{COD1}\cdot RST + QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot COD0\cdot RST + \overline{QB}\cdot\overline{QC}\cdot QD\cdot RST + QA\cdot\overline{QB}\cdot QD\cdot RST + QB\cdot\overline{QC}\cdot\overline{QD}\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot COND1\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}\cdot ERTR1\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot ERTR2\cdot RST + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD\cdot COND2\cdot RST$

$FINCMP = QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}$

$HLRE = \overline{QA}\cdot QB + \overline{QB}\cdot QD + QB\cdot QC + \overline{QA}\cdot\overline{QC}\cdot QD$

$DAUT7 = \overline{QA}\cdot\overline{QB}\cdot\overline{QC} + QA\cdot\overline{QB}\cdot QC + \overline{QA}\cdot\overline{QC}\cdot QD + QB\cdot\overline{QC}\cdot QD$

$DAUT6 = \overline{QA}\cdot QD + QB\cdot\overline{QC}\cdot QD + QB\cdot QC\cdot\overline{QD}$

$DAUT5 = QB\cdot\overline{QD} + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}$

$DAUT4 = QB\cdot\overline{QC}\cdot QD + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}$

$DAUT3 = QA\cdot\overline{QB}\cdot\overline{QC} + QB\cdot\overline{QC}\cdot QD + \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD} + \overline{QA}\cdot\overline{QC}\cdot QD$

$DAUT2 = \overline{QA}\cdot QB + \overline{QB}\cdot QD + QB\cdot\overline{QC} + \overline{QA}\cdot\overline{QC}\cdot QD$

$RESWUC = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}$

$FALLAC = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}$

$RuC = QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}$

$QFREN = QA\cdot\overline{QB}\cdot\overline{QC}\cdot\overline{QD}$

$INCER1 = \overline{QA}\cdot\overline{QB}\cdot\overline{QC}\cdot QD$

$INCER2 = \overline{QA}\cdot QB\cdot\overline{QC}\cdot\overline{QD}$

$INCER3 = \overline{QA}\cdot QB\cdot QC\cdot QD$

$FCOMP = SC1\cdot\overline{SC2}\cdot\overline{SC3} + \overline{SC1}\cdot SC2\cdot\overline{SC3} + \overline{SC1}\cdot\overline{SC2}\cdot SC3$

$VUE = \overline{SC1}\cdot\overline{SC2}\cdot\overline{SC3}$

$ETS = ET1\cdot ET2\cdot ET3$

$E2TF = \overline{ET1}\cdot\overline{ET2}\cdot ET3 + ET1\cdot\overline{ET2}\cdot\overline{ET3} + \overline{ET1}\cdot ET2\cdot\overline{ET3}$

$COD1 = ERP1\cdot ERP2 + ERP1\cdot ERP3$

$COD0 = ERP1 + \overline{ERP2}\cdot ERP3$

Figura 3.9 (a) Gals que Conforman al ACDR.

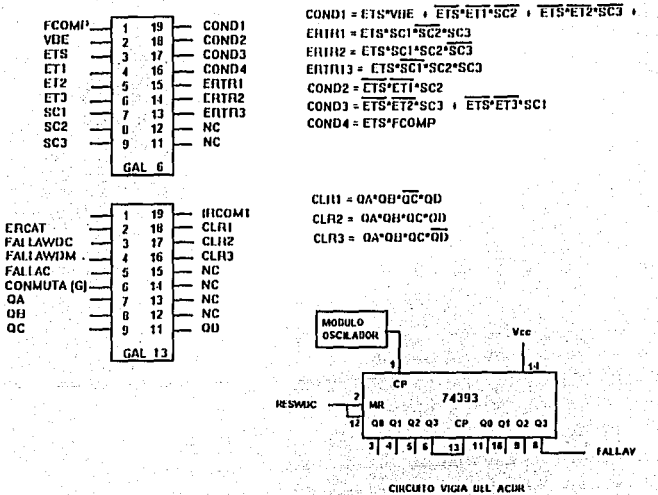


Figura 3.9 (b) Gals que Conforman al ACDR.

3.4.5 Mecanismos de autodiagnóstico en el ACDR

Debido a que este autómata es el más importante de la arquitectura de la CTF sus mecanismos de detección de fallas y diagnóstico son de vital importancia para el sistema TF. Esta electrónica dedicada contiene circuitos electrónicos para detectar anomalías tanto en la fase de detección de fallas como en la parte operativa del autómata en sí.

Como se vio anteriormente, el circuito diseñado para la comparación de datos provenientes de UPs y para el diagnóstico de fallas en comparadores o procesadores se asocia con las salidas SC1, SC2 y SC3, las cuales indican el tipo de falla que ocurre en la arquitectura, ver figura 3.6. Para esto los circuitos vigía

que evalúan a los comparadores generan la señal FALLACOMP, cuando estos fallan; el ACDR incluye también un circuito vigía que genera un pulso FALLWDC cuando el autómata presenta algún tipo de anomalía, de forma similar a la generada por el AMD de las UPs.

Como se puede ver en la figura 3.2, cada una de estas señales se lleva a una compuerta NOR cuya salida constituye la señal de solicitud de conmutación de UDFC (IRCOM 1,2). Cuando se genera cualquiera de estos pulsos se considera que el error que se ha presentado es de alta prioridad teniendo como consecuencia inmediata la conmutación de UDFC; el pulso de error que se genere en la UDFC principal viaja por el ducto TF y llega al pin IRQ de la UDFC redundante; con lo cual esta última inicia actividades como UDFC principal y desactiva a la UDFC recién fallada.

En la misma figura aparecen las señales ERCAT y G, la primera la generan las TAUDFCs cuando existen dos UPs declaradas como principales (lo cual sería catastrófico para la CTF), por lo que en caso de presentarse esta situación se obliga a la conmutación de UDFC debido a que la presencia de esta señal se asocia a un error en el RE o a parte de su electrónica.

La señal G (conmuta) es generada por el µC cuando el usuario desea realizar una conmutación de UDFC, la cual ha sido planeada principalmente para propósitos de demostración o en el caso de que el µC autodetecte problemas en su funcionamiento (opción a futuro).

Al lector se le enfatiza que las acciones de conmutación automática o solicitada por el usuario se ejecutan en tiempo real con lo que se asegura una amplia cobertura de fallas, siempre y cuando se sustituya lo más rápido posible a la tarjeta dañada por una tarjeta en buen estado y con la misma posición de interruptores.

3.5 Sección del microcontrolador

Como se mencionó en el apartado 3.3, uno de los bloques que componen a la UDFC es el del µC y su electrónica asociada que se encarga de interactuar con el usuario para labores de mantenimiento y para mandar alarmas al usuario al reportar fallas en el sistema. Cabe mencionar que a través de la UIM el µC se encarga de enviar hacia la pantalla de cristal líquido (LCD) los mensajes de interacción con el usuario.

Además, parte de su electrónica asociada cuenta con registros para leer variables de entrada y retener variables de salida, dichos registros aparecen con los nombres de Reg. Entradas y Reg. Salidas; también tiene registros para interactuar con UPs (para propósitos de inicialización y sincronización) éstos

registros aparecen con los nombres de Reg. Edo, Reg. Edo. Primo, Reg. Edo. Redundante y Reg. Acción, ver figura 3.3.

Además de la electrónica mencionada arriba el µC cuenta también con GALS para generar los bits de estado y categoría de las UPs (GAL 3 Y 4) los cuales pueden ser modificados por el ACDR o bien por el usuario a través del teclado matricial, cabe mencionar que la jerarquía más alta para modificar estos bits la tiene el ACDR. Los GALS 10, 11 y 12 son utilizados para la decodificación de memoria externa, decodificación de puertos utilizados por las UPs para leer y escribir registros, y además incluyen la lógica requerida para generar los habilitadores de escritura y lectura de registros, ver figura 3.10.

A continuación se mencionarán y se describirán cada uno de los registros integrados en la tarjeta UDFC :

- **Registro de Acción (RA).** Aquí el µC deposita un dato de protocolo previamente al envío de una señal de interrupción a las UPs; el dato especifica el tipo de acción que los procesadores deberán realizar.

- **Registro de Estado (RE).** Este registro es actualizado ya sea por el ACDR o por el µC. Sus bits ET1,ET2,ET3,CT1,CT2 y CT3 definen el estado y la categoría, respectivamente, de las UPs instaladas. Sus salidas se envían directamente al ducto TF y de ahí cada UP toma sus bits de control asignados.

- **Registro de Estado Primo (REP).** Contiene la misma información que el registro de estado sólo que sus salidas se leen exclusivamente por las UPs a través del ducto AT. Cuando existen cambios en el registro de estado se envía una interrupción a las UPs junto con la acción requerida y ellas, entre otras cosas, proceden a leer el cambio efectuado en el registro de estado primo. Posteriormente las UPs actualizan el dato leído en el registro de estado redundante de la UDFC de respaldo; este proceso se realiza para lograr que la UDFC de respaldo pueda iniciar operaciones (en caso de que la UDFC principal falle) conservando el estado y la categoría de las UPs instaladas.

- **Registro de Estado Redundante (RER).** Como se mencionó en el punto anterior este registro lo actualizan las UPs siempre y cuando la UDFC que lo contiene sea redundante. Con el registro actualizado y en caso de requerirse una conmutación de UDFC la tarjeta redundante conservará las asignaciones de categoría y estado de los procesadores debido a que una de las tareas iniciales del 68HC11 (durante la transición de UDFC redundante a principal) consiste en leer este registro y copiar su contenido en el RE y en el REP para entonces iniciar operaciones.

La lógica para generar los diversos habilitadores está contenida en los GALS 10 y 11, mientras que la decodificación de los puertos de comunicación con las UPs se implantó en el GAL 12. Los CIs y las ecuaciones que conforman la lógica de este bloque se muestran en la figura 3.10.

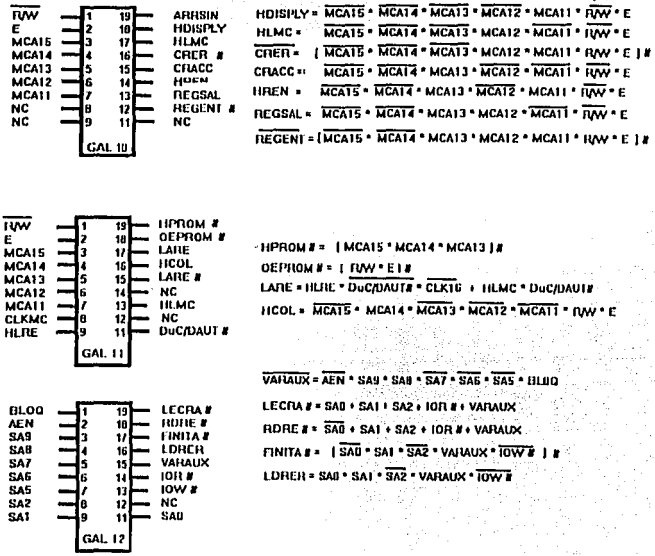


Figura 3.10 Banco de Gals del 68HC11E1.

3.6 Protocolo de comunicación entre UPs y UDFCs

Como se ha visto con anterioridad, para que la CTF realice sus labores encomendadas debe existir una comunicación entre UPs y UDFCs con el propósito de anular cualquier inconsistencia en algunas condiciones específicas de funcionamiento tales como inicialización y actualización de datos, salvar

información en disco duro del programa de aplicación y recuperación de dicha información.

Para ello el protocolo básico de comunicación entre UDFCs y UPs consiste en el envío de una señal de interrupción, desde el µC contenido en la UDFC principal hacia la UPs instaladas, las que atenderán la interrupción leyendo el RA de la UDFC principal. La información contenida en este registro es colocada por el µC antes de interrumpir a los procesadores. Los datos contenidos en ese registro proporcionan información que define el tipo de acción solicitada así como a la(s) UP(s) a quien(es) va dirigida.

En la tabla 3.1 aparecen las diferentes acciones utilizadas en el sistema y la forma de definir a las UPs involucradas. Como puede verse existen diferentes combinaciones usadas y otras que no se utilizan las cuales se pueden aprovechar en un futuro.

	REGISTRO DE ACCIÓN		
Acción	No. UP	Código de Acción	Actividad
1	0XXX	0001	Reanuda operación concurrente de UPs
2	0XXX	0010	Carga última inform. de disco duro a procesador nuevo
3	0XXX	0011	Salva última inform. de procesador actual a disco duro
4	0XXX	0100	Frena UPs por conmutación de UDFCs
5	0XXX	0101	Actualiza registro de estado
10	0XXX	1010	Enumera al procesador corriente
Donde XXX = 001 ⟹ UP1 010 ⟹ UP2 011 ⟹ UP3 100 ⟹ Todas			

Tabla 3.1 Codificación del Registro de Acción.

En seguida se explican las acciones programadas:

ACCIÓN 1 Arranque Síncrono. Se utiliza durante el arranque del sistema triplex o bien cuando por alguna razón se ha procedido a detener la operación del sistema y se desea reanudar nuevamente la operación concurrente síncrona.

ACCIÓN 2 Ordena que se cargue el archivo CTFBACK.DAT de disco a RAM del nuevo procesador por declarar en el sistema.

ACCIÓN 3 Indica respaldar toda la información contenida en RAM del procesador principal en el archivo CTFBACK.DAT de disco duro. Esta opción se usa en combinación con la anterior para inicializar una nueva UP con SO, programa de aplicación y datos de aplicación.

ACCIÓN 4 Se le avisa a todas las UPs que ha ocurrido una conmutación de UDFC y por tanto se les obliga a frenar operaciones hasta que lo ordena la nueva UDFC. Es un paso adicional para asegurar la sincronización después de la falla de la UDFC principal.

ACCIÓN 5 Esta acción le indica a las UPs activas que ha habido algún cambio en el RE de la UDFC principal. Concretamente indica que se ha modificado el valor de alguno de los bits de estado o categoría contenidos en dicho registro.

ACCIÓN 10 Asigna número de UP al procesador corriente. Esta opción se utiliza para retroalimentar a la UP respectiva un número con el que será identificada por las UDFCs. Se debe recordar que algunas acciones van dirigidas hacia UPs específicas, para ello, cada UP debe saber que número se le asignó (1,2 ó 3). Esta acción se les envía a las UPs durante el arranque del sistema TF y durante la inicialización de una UP recién instalada.

Para cerrar el lazo de comunicación entre UDFCs y UPs, el diseño de la UDFC contiene puertos digitales, con los cuales las UPs podrán escribir o leer a los registros de las UDFCs. La habilitación de estos puertos se lleva a cabo mediante las siguientes señales:

LECRA# (Lectura del Registro de Acción) Por medio de este puerto las UPs realizan la lectura de la acción de interrupción solicitada por la UDFC principal.

RDRE# (Lectura del Registro de Estado) Por medio de este puerto las UPs actualizan su información acerca del estado corriente de la arquitectura.

LDRER# (Carga el Registro de Estado Redundante) Cada vez que existe algún proceso de reconfiguración en la arquitectura TF, a las UPs se les ordena leer el registro de estado primo de la UDFC principal y almacenarlo en el RER de la UDFC redundante. La escritura al RER de la UDFC redundante también se realiza después de dar de alta a una UDFC recién instalada.

FINITA# (Fin de Iniciación de Tarjeta) Con este puerto cada procesador emite una respuesta de fin de iniciación durante el arranque de las UPs (una a la vez). Durante la operación del sistema este puerto es utilizado por la UP principal para sincronizar los procesadores en circunstancias tales como, dar de alta un nuevo

procesador, después de conmutar de UDFC y después de una lectura del RE o del RA realizada por UPs.

3.7 Diseño del Circuito Impreso de la UDFC

Para desarrollar el circuito impreso de esta tarjeta, se utilizó el paquete de diseño asistido por computadora denominado TANGO PCB PLUS + +. El impreso se diseñó en dos caras, para lo cual se elaboraron cuatro capas (soldadura, componentes, referencias y perforaciones). Este trabajo se describe en una tesis previa [Mejía 1994].

CAPITULO CUATRO

ENSAMBLADO DE CIRCUITOS IMPRESOS, PRUEBAS DIVERSAS Y PRUEBAS OPERATIVAS REALIZADAS A LAS TARJETAS DE AISLAMIENTO DE UDFCs Y A UDFCs

4.1 Introducción

En los primeros capítulos se describió la arquitectura general de la CTF y la arquitectura de la UDFC así como la manera en que opera la CTF. En este capítulo se describirán los procedimientos utilizados en el ensamblado de las TA de UDFCs y de la UDFC así como diversas pruebas del funcionamiento de ambas tarjetas.

Con las pruebas realizadas se encontraron algunos errores que tenía el diseño original, los cuales se corrigieron. Las correcciones hechas aparecen ya en todos los diagramas de esta tesis así como en las ecuaciones que se implementaron en los autómatas y en la decodificación que utiliza el μC. Las pruebas realizadas en este capítulo solo se abocan a TA de UDFCs y a UDFCs, las demás pruebas de integración con los demás módulos de la CTF serán descritas en el capítulo seis.

4.2 Ensamblado de TA de UDFCs y de UDFCs

Como se describió en el capítulo dos, las TA de UDFCs permiten el paso de señales provenientes de la UDFC principal hacia las demás tarjetas de la CTF para el control de la arquitectura TF, también permiten la extracción o inserción de UDFCs en mal y buen estado respectivamente, sin riesgos de colisiones en los ductos. De la misma manera la UDFC representa el subsistema de mayor importancia en la CTF ya que de su correcta operación depende el desempeño del sistema TF.

De aquí que para el ensamblado de las tarjetas (TA de UDFCs y UDFCs) se hayan seguido procedimientos ordenados con el fin de incrementar las posibilidades de éxito al realizar las pruebas. El procedimiento de ensamblado siguió los siguientes pasos:

a) Revisión del circuito impreso.
b) Seguimiento de planos de Vcc y GND en el impreso.
c) Verificar la polarización de capacitores de desacoplo y filtrado de ruido en el impreso.
d) Soldado de bases para CIs , resistores y demás componentes.

e) Pruebas de continuidad.
f) Limpieza de las tarjetas.

4.2.1 Revisión de circuitos impresos

Para realizar esta prueba se utilizaron los esquemáticos o diagramas eléctricos generados por el paquete ORCAD y en el gráfico de la tarjeta (PCB) que proporciona el paquete TANGO PCB PLUS.

La revisión consiste en hacer un seguimiento de las señales que manejan ambas tarjetas tomando un punto de referencia (el cual corresponderá a una señal determinada) y localizando el o los lugares a los cuales debería llegar la señal en cuestión. El seguimiento se referenció al esquemático, al PCB y siguiendo la lógica del funcionamiento de las tarjetas, con el fin de encontrar algún error en la circuitería o bien en la lógica de funcionamiento de las tarjetas.

A manera de comentario se recomienda que al realizar este tipo de prueba, se encuentre bien descansado y que cuente con tiempo suficiente para realizarla en una sesión de preferencia, se recomienda también que todos los errores encontrados sean anotados en el mismo esquemático. Una vez concluida la revisión se actualizan los cambios requeridos, una vez hecho esto se realizan una serie de revisiones hasta eliminar, o bien, reducir el número de errores al mínimo.

4.2.2 Seguimiento de planos de VCC y GND en el impreso

Otro punto importante en la revisión de circuitos impresos es la alimentación de los CIs contenidos en las tarjetas. Dicha alimentación es proporcionada por las diferentes ramificaciones de los planos de Vcc y de GND contenidos en el diseño de las tarjetas.

La prueba consiste en verificar que todos los componentes de las tarjetas reciban la polarización adecuada y que los niveles de voltaje suministrados sean los correctos, nuevamente, se utilizó un multímetro para revisar la continuidad entre líneas de polarización de cada componente y los planos de Vcc y de GND, además de verificar que los niveles de polarización fueran los correctos.

4.2.3 Capacitores de desacoplo y de filtrado

Estos componentes se incluyen en el diseño para evitar que las diferentes fuentes de ruido afecten el desempeño de los componentes alojados en las tarjetas.

Se verificó que todos los capacitores de desacoplo y de filtrado en la alimentación estuvieran polarizados con Vcc y GND.

4.2.4 Soldado de bases para CIs y demás componentes

Este paso es muy importante ya que asegura que todas las líneas que viajan en el impreso lleguen y se propaguen de manera fiel hasta los circuitos integrados, resistores, capacitores y demás componentes. Antes de realizar este proceso se deben de tomar en cuenta los siguientes factores:

a) Tipo de tecnología de los CIs a soldar con el fin de tomar las precauciones necesarias para no dañarlos.
b) Contar con la herramienta necesaria y adecuada para el soldado de bases y componentes, tales como cautín de temperatura regulable, diferentes tipos de punta para cautín, soldadura estañada del calibre adecuado, pulsera antiestática, lámpara con lupa, soporte para PCB, etcétera.
c) Tener todo el material (bases, componentes, etc.) cerca del área de soldado, ordenado y protegido (en caso de tratarse de tecnología HCMOS o CMOS). El área de soldado debe ser un lugar bien iluminado, lejos de cualquier sustancia volátil y con buena ventilación.

Una vez tomados en cuenta los factores arriba mencionados, se procede al soldado de los componentes con las siguientes consideraciones, se sueldan primeramente todos aquellos elementos cuya altura sobre el impreso sean los más pequeños y se finaliza con los más altos tales como conectores, cristales, bases para SIMS, etc. El procedimiento de soldado es el siguiente, se coloca la punta del cautín sobre el área a soldar asegurándose de que exista contacto con el componente a soldar y el área sobre la que se soldara el componente. Se deja aproximadamente unos tres segundos y se acerca la soldadura por el extremo contrario al del cautín. Se realiza una pequeña presión hacia el componente con el fin de que éste actúe como un capilar y la soldadura adquiera una forma cónica, es decir, el área de soldado será la base mientras el extremo terminal del componente es la punta. Una vez que la soldadura adquiere esta forma se retira la soldadura y el cautín se deja un instante más con el fin de evitar un soldado en frío.

Existe una forma de saber si el soldado fue en frío y es mediante una inspección visual, si el soldado fue exitoso la soldadura presenta un reflejo brilloso sobre su superficie y libre de burbujas de aire. Cuando existe un soldado en frío la soldadura adquiere un color opaco sobre su superficie y presenta porosidades, lo cual indica la presencia de burbujas de aire debido a que el cautín se retiró antes de tiempo. Para solucionar este problema se vuelve a colocar el cautín en el punto de soldado y se vuelve a calentar la soldadura hasta que adquiera la forma cónica y el brillo.

4.2.5 Pruebas de continuidad

Como se vio en el punto anterior, de un procedimiento de soldado deficiente se derivarán problemas indeseables tales como, falso contacto, corto entre pines o líneas. Para poder detectar estos problemas después de soldar componentes se procede a realizar la prueba de continuidad. Esta consiste en verificar continuidad en todas las líneas que se unieron mediante soldadura a bases, componentes, etc.

4.2.6 Limpieza de las tarjetas

Una vez que se realizaron con éxito todos los puntos anteriores el procedimiento final es la limpieza de las tarjetas. Esta limpieza se lleva acabo con el fin de quitar o remover partículas de soldadura que se hayan filtrado en bases, conectores, circuitos de montado superficial, etc., o bien retirar cualquier partícula indeseable para evitar cortos o desprendimiento de pines de algunos componentes. También sirve para remover el exceso de grasa que deja la soldadura y para ver con claridad las uniones por soldadura, así como retirar todas las partículas de polvo.

Para la limpieza de los impresos se utilizó Alcohol Isopropílico, el cual se vertió en un recipiente que pudiera cubrir a las tarjetas y con una brocha se limpió completamente el impreso. Se recomienda que el frotado en componentes de montado superficial sea muy tenue. Al finalizar el limpiado se recomienda que el secado del impreso sea con aire a presión (sopletear) para remover las partículas que el alcohol no pudo desalojar.

4.3 Pruebas operativas

En este apartado se describen las pruebas de funcionamiento para las TA de UDFCs y para las UDFCs, para ello se hace referencia a los diagramas eléctricos da cada tarjeta.

4.3.1 Pruebas realizadas a TA de UDFCs

Como se describió en el capítulo dos, las TA de UDFCs son módulos que permiten el paso de señales provenientes de la UDFC principal hacia las demás tarjetas de la CTF. Con el aislamiento se obtiene el control de la arquitectura TF y particularmente hace posible la extracción o inserción de UDFCs en mal y buen estado respectivamente, sin riesgos de colisiones en los ductos. Las TAUPs están divididas en tres secciones:

a) Aislamiento de voltaje.
b) Aislamiento del ducto AT.
c) Aislamiento del ducto TF.

Por lo anterior, las pruebas realizadas fueron básicamente dos, pruebas de aislamiento de voltaje y prueba de aislamiento de señales. Para llevar a cabo las pruebas de aislamiento de señales se necesita de una UDFC y de la UIM, esta prueba será descrita en el capítulo seis.

- Prueba de aislamiento de voltaje. La TA de UDFCs contiene 2 transistores Q1 y Q2 tipo Mosfet conectados en paralelo, los que permiten o impiden el paso de la alimentación hacia las UDFCs, estos transistores se polarizan por medio de Q3 el cual es llevado a corte mediante la señal de control ACTVCC, ver figura 4.1. Las ecuaciones que conforman a ACTVCC se muestran en la tabla 4.1

Ecuaciones del Gal 2

$$ERCAT = \overline{CT1}*\overline{CT2}*CT3 + \overline{CT1}*CT2*\overline{CT3} + CT1*\overline{CT2}*\overline{CT3}$$

$$OPVALi = OPVAL1*OPVAL2 + OPVAL1*OPVAL3 + OPVAL2* OPVAL3$$

$$HLDAT = HLDA1*HLDA2*HLDA3$$

Ecuaciones del Gal 3

$$(R/P\#)\# = HUAR/P\# + 3SVVCIN + UDFCNVA$$

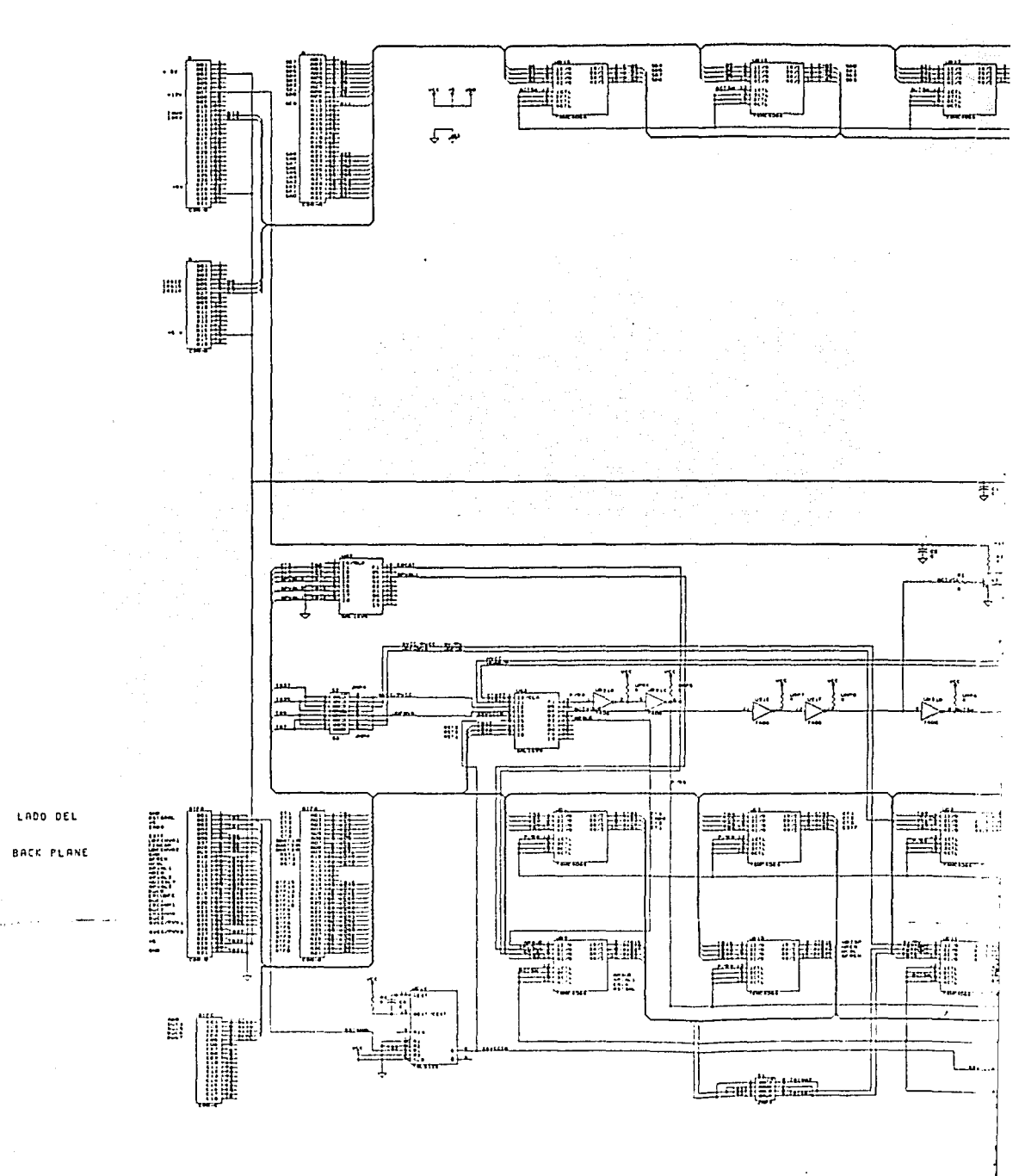
$$ARQUE = RST1 + RST2 + RST3$$

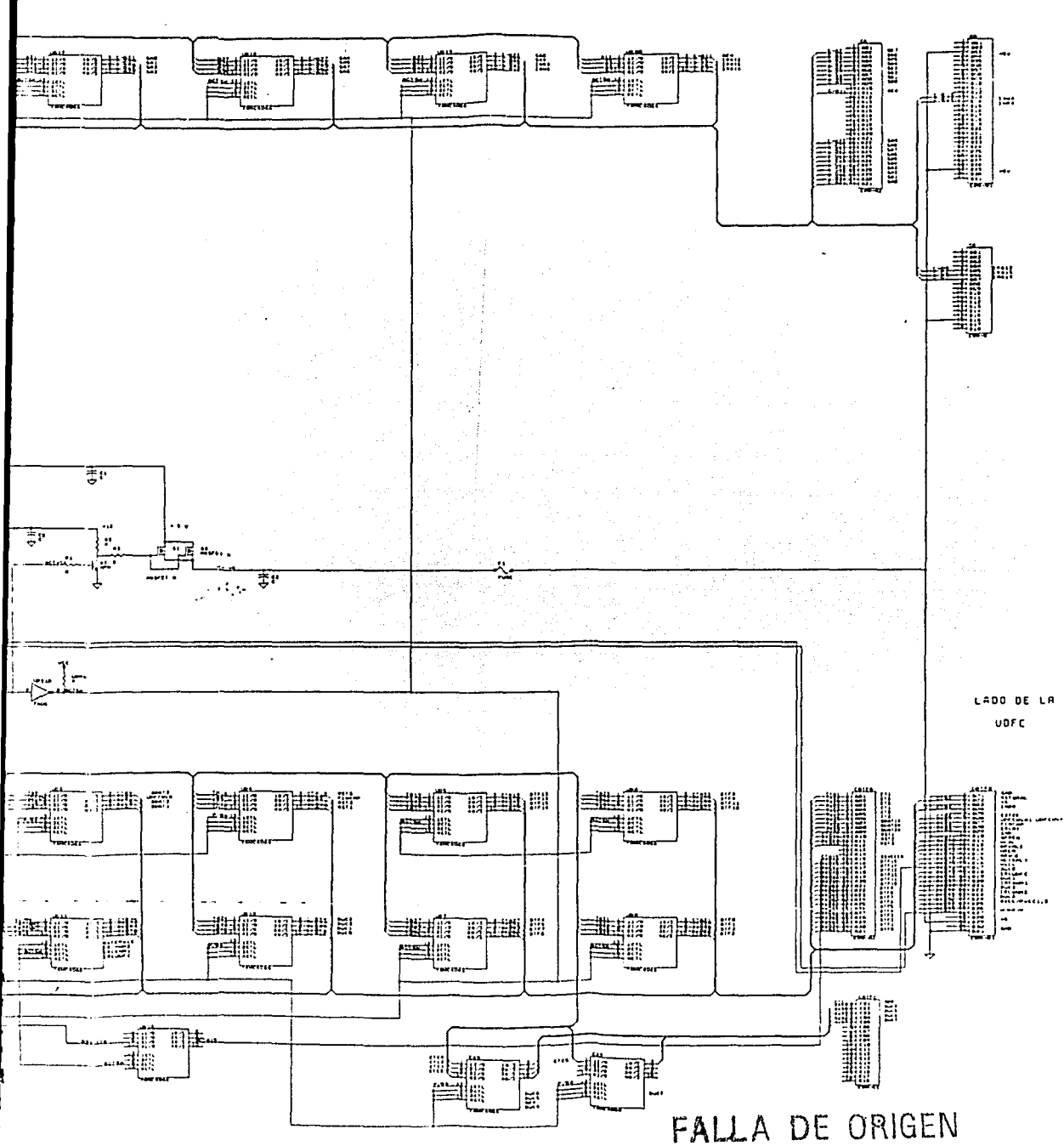
$$ATCVCC\# = (IIVCC + UDFCNVA + 3SVCCIN)*((Q/PVCC\#)\#)$$

Tabla 4.1 Lógica Programada en Gals.

En la prueba de aislamiento de voltaje se observa el comportamiento de la señal ACTVCC. En la prueba se varían las señales eléctricas que generan a la señal ACTVCC, observando los valores obtenidos para validar las ecuaciones.

Figura 4.1 Diagrama Electrónico de la TAUDFC.


LADO DEL
BACK PLANE



FALLA DE ORIGEN

Una vez validadas las ecuaciones se procedió a programarlas en el Gal definitivo, para después colocarlo en la TA y observar el control del aislamiento de voltaje hacia UDFCs; los resultados de esta prueba fueron exitosos.

4.3.2 Pruebas realizadas a UDFCs

Como se mencionó en el capítulo dos, la UDFC se dividió en dos bloques, el de los autómatas de alta velocidad y el del μC por lo que las pruebas realizadas a la UDFC se dividieron también en dos partes :

4.3.2.1 Pruebas a los autómatas de alta velocidad

Esta prueba consistió en revisar y validar las ecuaciones que conforman a los autómatas de multiplexaje (AMD) y al autómata de comparación de datos y reconfiguración (ACDR) para ello se utilizó el paquete PALASM2 de "Advanced Micro Devices". Con éste se obtuvo la historia de la simulación y posteriormente se comparó ésta con los resultados teóricos del diseño en papel. En la figura 4.3 aparece la simulación del AMD y en la figura 4.4 la simulación del ACDR.

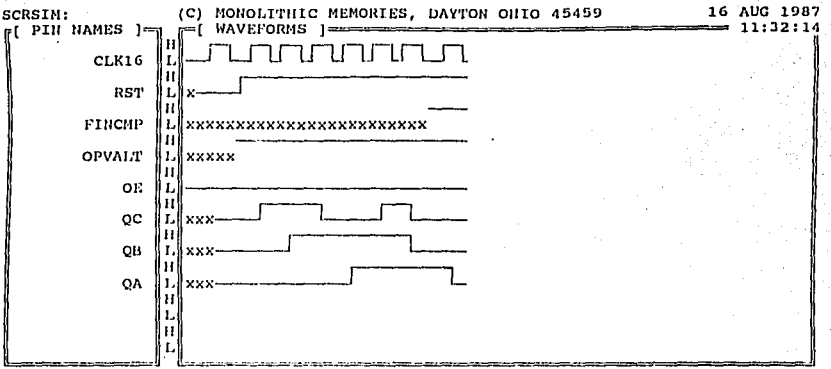
A continuación se describen los pasos que siguen las señales del AMD :

En la figura 4.3, parte superior, se presentan las señales de entrada del AMD. En el primer ciclo de reloj se fija al autómata en el estado "A". En el segundo pulso aparece la señal OPVALT, que indica que una operación válida ha sido detectada lo que obliga al autómata a ir al estado "B".

El tercer pulso es utilizado para asegurar que los primeros datos por compararse hayan sido transmitidos, pasando con ello al estado "C". El cuarto pulso nos lleva al estado "D" y en el quinto pulso se vuelve a dar tiempo para completar la transmisión de información (estado "E").

El sexto pulso conduce al estado "F" mientras que con el séptimo pulso se llega al estado "G" donde el autómata permanece hasta que la señal FINCMP lo conduzca al estado "A" (octavo pulso).

La figura inferior nos muestra las señales de salida que el AMD genera. Como puede verse la actividad empieza en el segundo pulso de reloj donde aparece la señal RESCNT2 con un nivel alto (estado "A"). En el estado "B" solo las salidas LCD1 y BDAT1 se activan con un nivel alto y la señal RESCNT2 se deshabilita (nivel bajo). En el estado "C" no está presente ninguna señal de salida. En el estado "D" solo las salidas LCD2 y BDAT2 se activan y en el estado "E" no ocurre ninguna actividad. En el estado "F" solo las señales LCD3 y BDAT3 se activan. Y por último en el estado "G" solo las señales IACOM y RESCNT2 se habilitan.



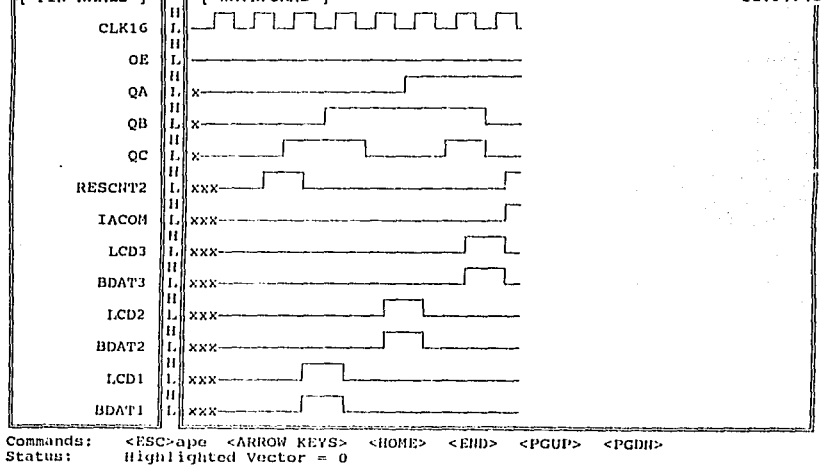


Figura 4.3 Simulación del AMD.

Para finalizar con las pruebas a los autómatas a continuación se describen los pasos que siguen las señales del ACDR :

La actividad de dicho autómata comienza al presentarse el primer pulso de reloj, ver figura 4.4.1, donde se tiene presente la señal de reset en nivel bajo, lo cual ubica al autómata en el estado "A" (figura 3.8).

En el segundo pulso de reloj las condiciones que se tienen son linea de reset en alto y se presenta la señal IACOM con un nivel alto para obligar la conmutación al estado "B". Del tercero al séptimo pulso de reloj se ilustra la trayectoria de los estados B-M-N-O-P. En los pulsos 7 y 8 se regresa al estado "B" del autómata.

Del pulso 9 al 14 se observa la trayectoria de los estados B-C-I-M-N-O-P y en los pulsos 15 y 1 de la figura 4.4.2 se retorna al estado "B" del autómata. Del pulso 2 al 8 se ilustra la trayectoria de los estados B-C-I-J-M-N-O-P mientras que en los pulsos 9 y 10 regresamos al estado "B".

De los pulsos 11 al 16 y 1 de la figura 4.4.3 se ilustra la trayectoria de los estados B-C-I-K-M-N-O-P y en los pulsos 2 y 3 se regresa al estado "B". Del pulso 4 al 10 se tiene la trayectoria B-C-I-L-M-N-O-P mientras que con los pulsos 11 y 12 se regresa al estado "B".

En los pulsos 13 y 14 se ilustra la trayectoria B-D-I; debido a que en los párrafos anteriores ya se ilustraron las trayectorias I-(J,K,L)-M-N-O-P, éstas ya no se agregaron a las simulaciones restantes. En los pulsos 15 y 1 de la figura 4.4.4 se regresa al estado "B".

Del pulso 2 al 3 se muestra la trayectoria B-E-I y se regresa al estado "B" en los pulsos 4 y 5. Por último, en los pulsos 6, 10 y 14 se recorren las trayectorias B-F , B-G y B-H respectivamente, en los pulsos intermedios se regresa al estado "B".

En la figura 4.4.5 aparece la simulación correspondiente a las salidas del autómata que se encargarán de contabilizar los errores transitorios, el reset del vigia electrónico y la señal $R_{\mu}C$.

En el primer vector se encuentran las condiciones que corresponden al estado "A" y la señal de RESWDC se hace alto. El segundo vector nos ubica en el estado "C" donde se genera la señal INCER1.

El tercer vector nos ubica en el estado "D" y se genera la señal INCER2.
El cuarto vector nos ubica en el estado "E" y se genera la señal INCER3.
El quinto vector nos ubica en el estado "O" y genera la salida $R_{\mu}C$.
El último vector nos ubica en el estado "P" y genera la señal QFREN.

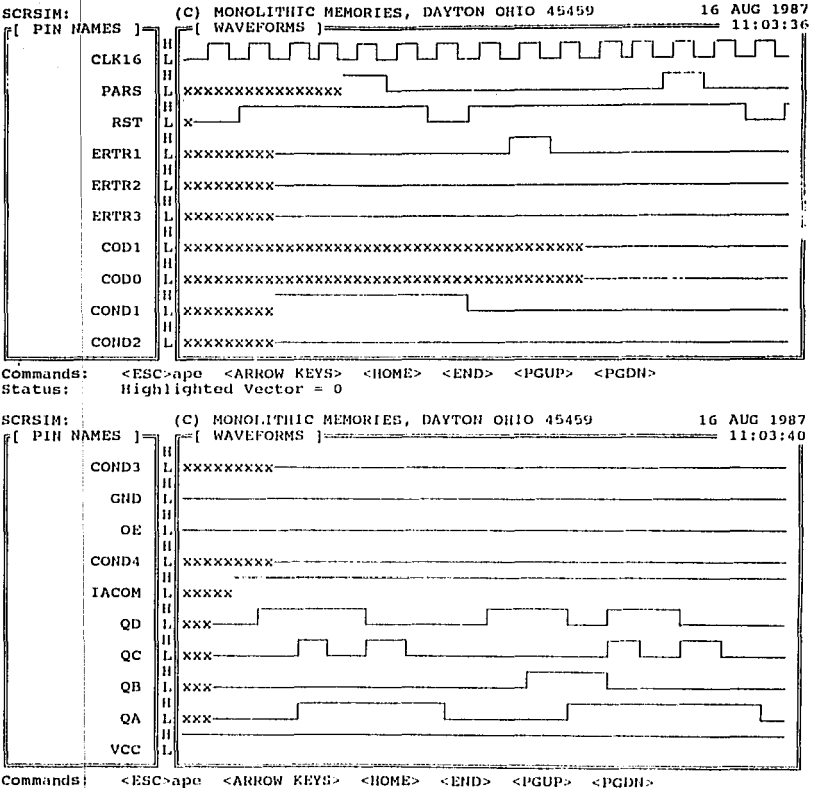


Figura 4.4.1 Simulación del ACDR.

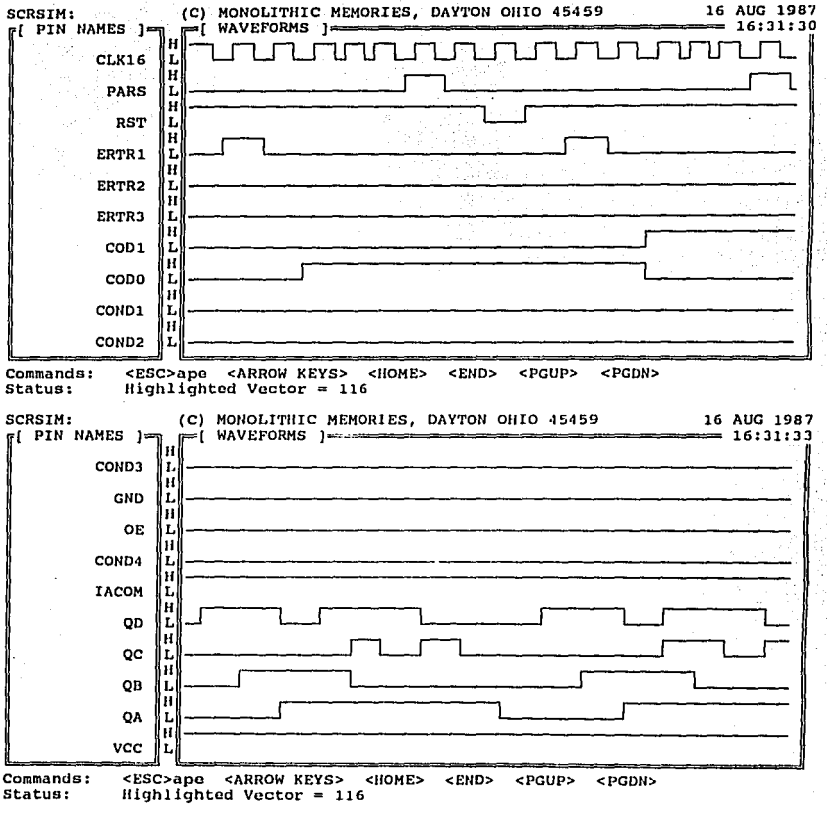


Figura 4.4.2 Simulación del ACDR.

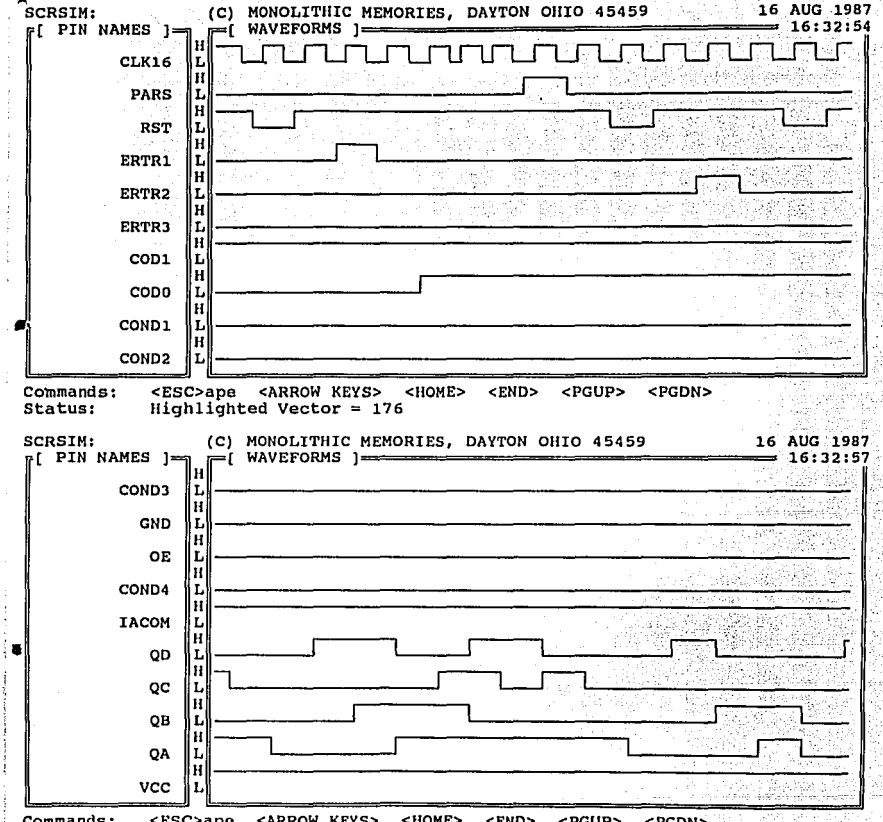


Figura 4.4.3 Simulación del ACDR.

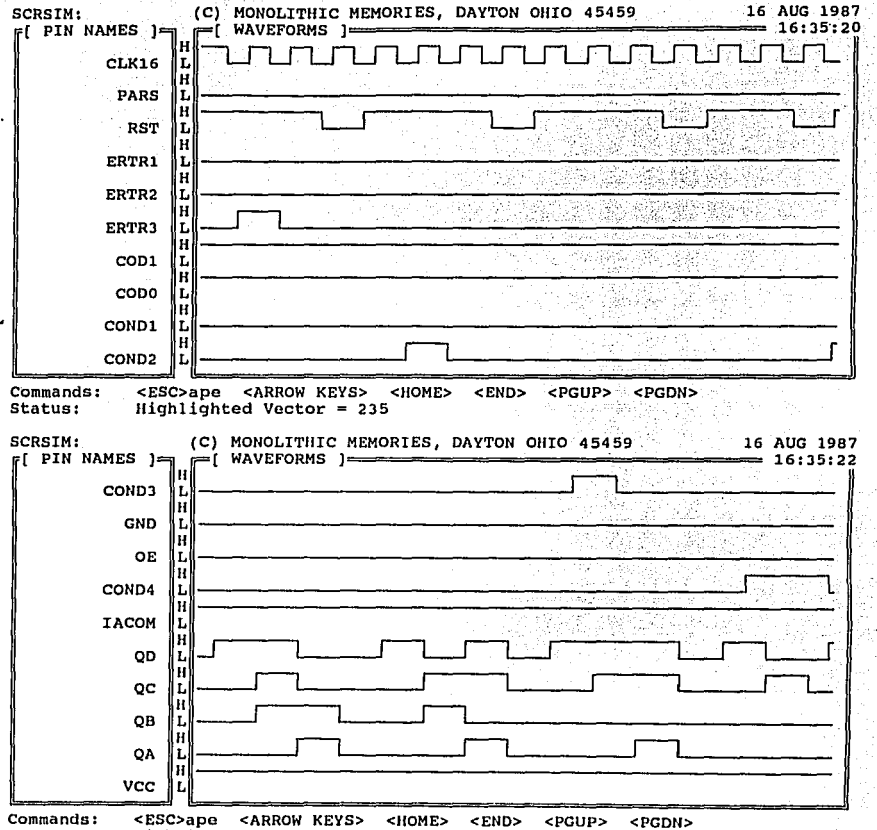


Figura 4.4.4 Simulación del ACDR.

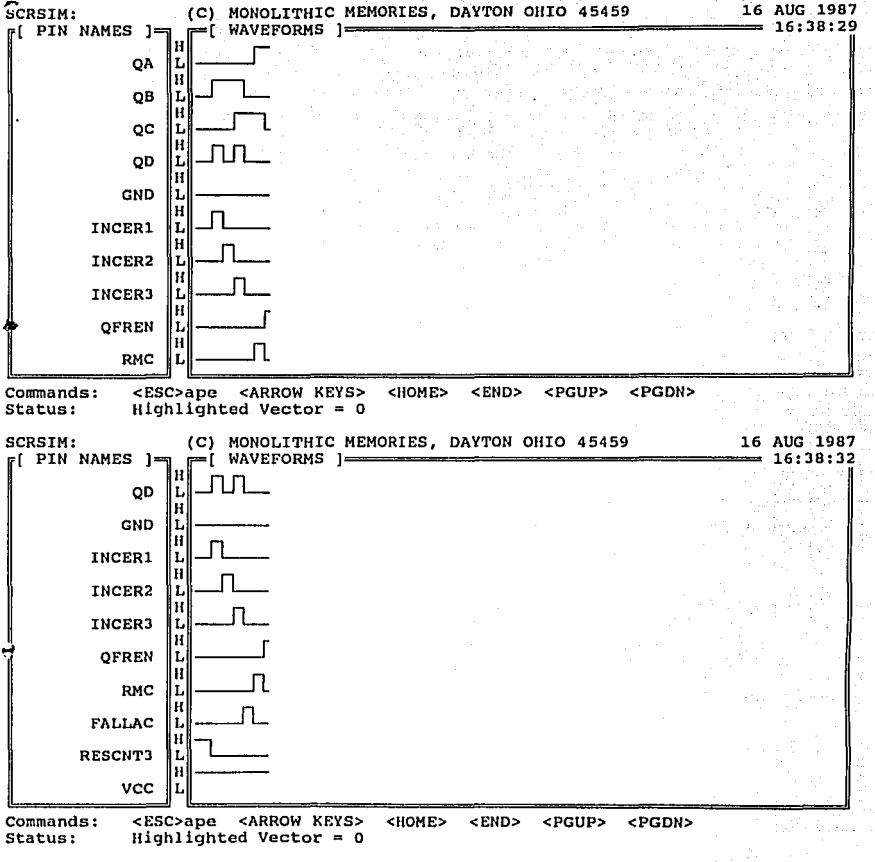


Figura 4.4.5 Simulación del ACDR.

En la figura 4.4.6 se da la simulación de algunas de las salidas de los diferentes estados del ACDR. Así tenemos que en el primer vector se generan las salidas correspondientes al estado "C" del autómata, donde se cede la categoría de principal al procesador "2" y se habilita la señal HLRE.

En el segundo vector nos ubicamos en el estado "D" y ahora el procesador "1" será principal.

El tercer vector nos lleva al estado "E" y el procesador "1" será principal. El vector cuatro conduce al estado "F" cediendo al procesador "2" la categoría de principal y apagando a los procesadores "1" y "3". El quinto vector nos lleva al estado "G" cediendo la categoría de principal al procesador "1" y apagando a "2" y a "3". El sexto vector nos lleva al estado "J" donde las categorías y los estados son los que aparecen en el ACDR (figura 3.8).

El séptimo vector nos conduce al estado "K" y las salidas son las mismas que se dan en la figura 3.8. En el octavo vector nos trasladamos al estado "L" y sus salidas coinciden con las de la figura 3.8. El último vector nos ubica en el estado "P" y es aquí donde se genera la señal FINCMP (nivel alto) mientras que la señal de habilitación HLRE se hace bajo, para no alterar el estado actual de la arquitectura.

En la figura 4.4.7 se proporciona la simulación correspondiente a las señales que se manejaron en el árbol de decisiones que se describió en el capítulo tres y que sirven para generar las señales de errores transitorios (ERTR1, ERTR2 y ERTR3) y de condiciones (COND1 - COND4) que el ACDR utiliza. En la figura inferior se aprecia como ocurre solo una condición a la vez para pasar del estado "B" al estado "I" ó del "B" al "M", esto nos asegura que el ACDR no llegará a un estado indeterminado. En caso de que el autómata se estacione permanentemente en algún estado, el vigía electrónico enviará la señal correspondiente para avisar que el ACDR no fue capaz de concluir su tarea.

En la figura 4.4.8 se muestran las señales que gestionaran la conmutación de UDFC y las salidas que inicializan los contadores de fallas transitorias. El bosquejo inferior nos muestra como las salidas CLR1, CLR2 y CLR3 aparecen solo en los estados J,K y L respectivamente. En el esquema superior se aprecia que cuando todas las líneas que obligan a conmutar de UDFC son bajas no se emite la señal IRCOMN (que estará en estado alto), pero al presentarse cualquiera de ellas se obliga a cambiar el nivel lógico de IRCOMN y por lo tanto inicia la conmutación de UDFC.

Por último, en la figura 4.4.9 se muestra como se generan las señales COD0 y COD1 que el autómata necesita para determinar la trayectoria a seguir para ir del estado "I" al "M". Aquí también se evalúa si existen fallas permanentes o no en el sistema TF.

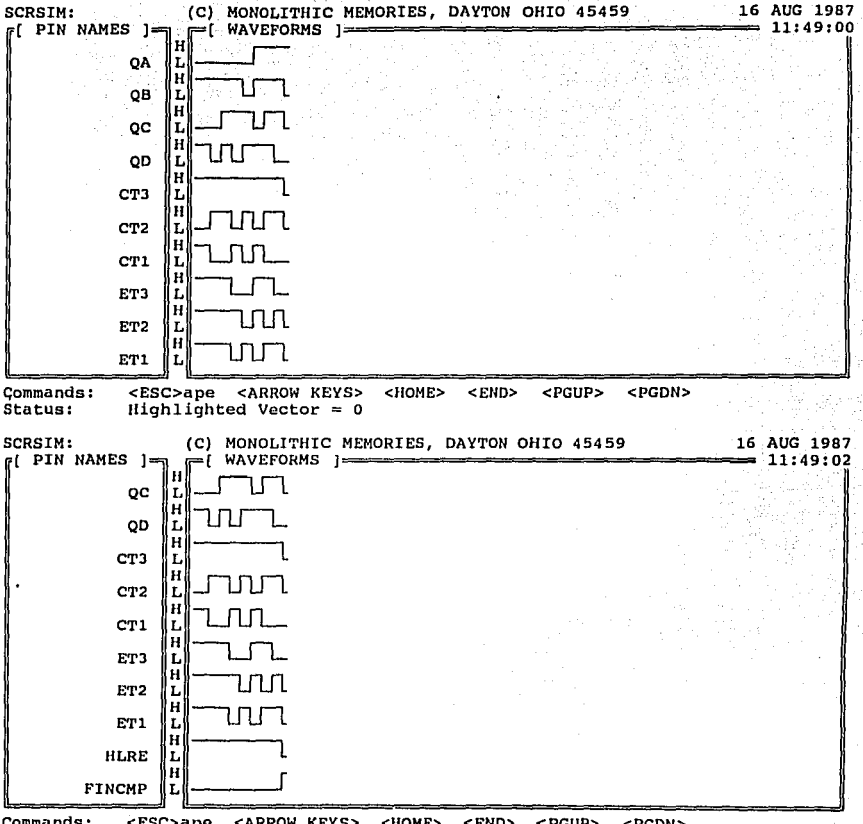


Figura 4.4.6 Simulación del ACDR.

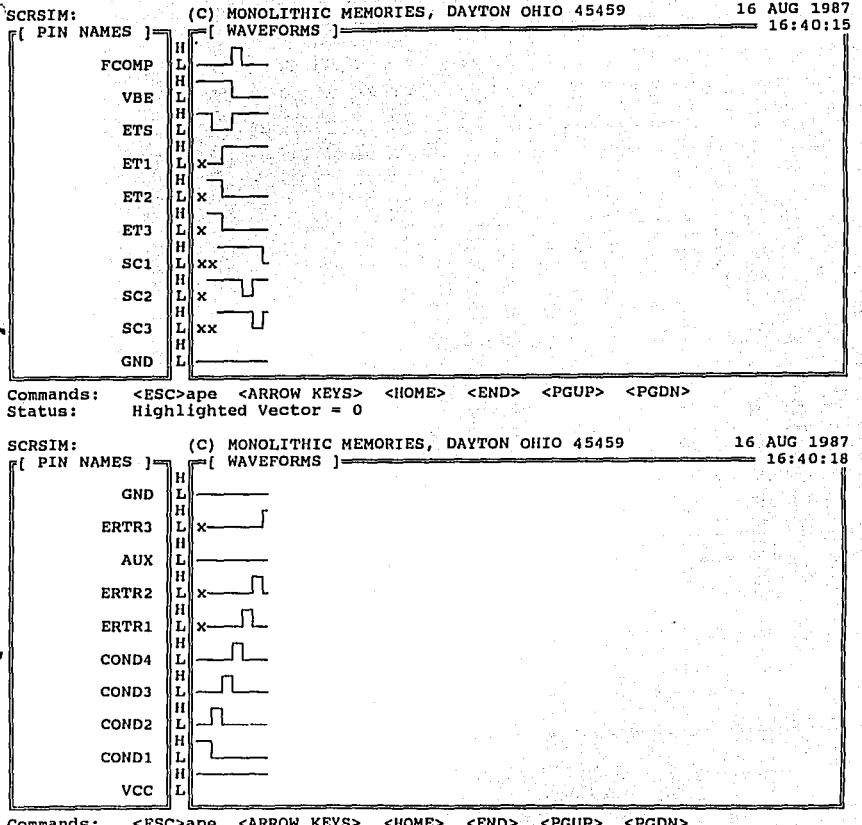


Figura 4.4.7 Simulación del ACDR.

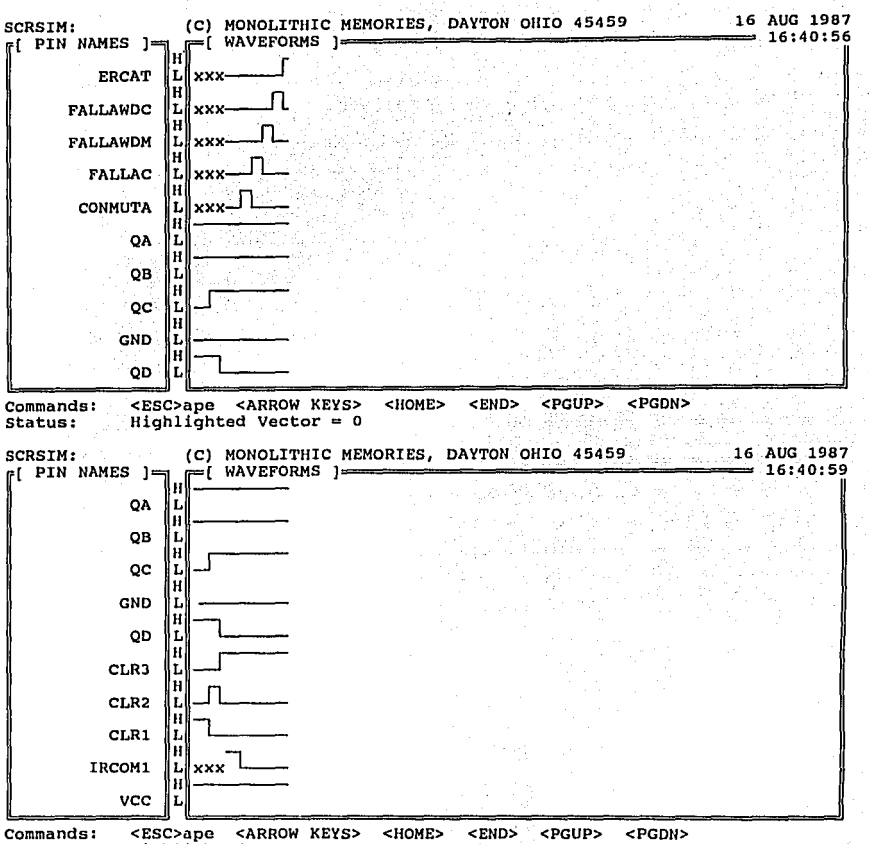


Figura 4.4.8 Simulación del ACDR.

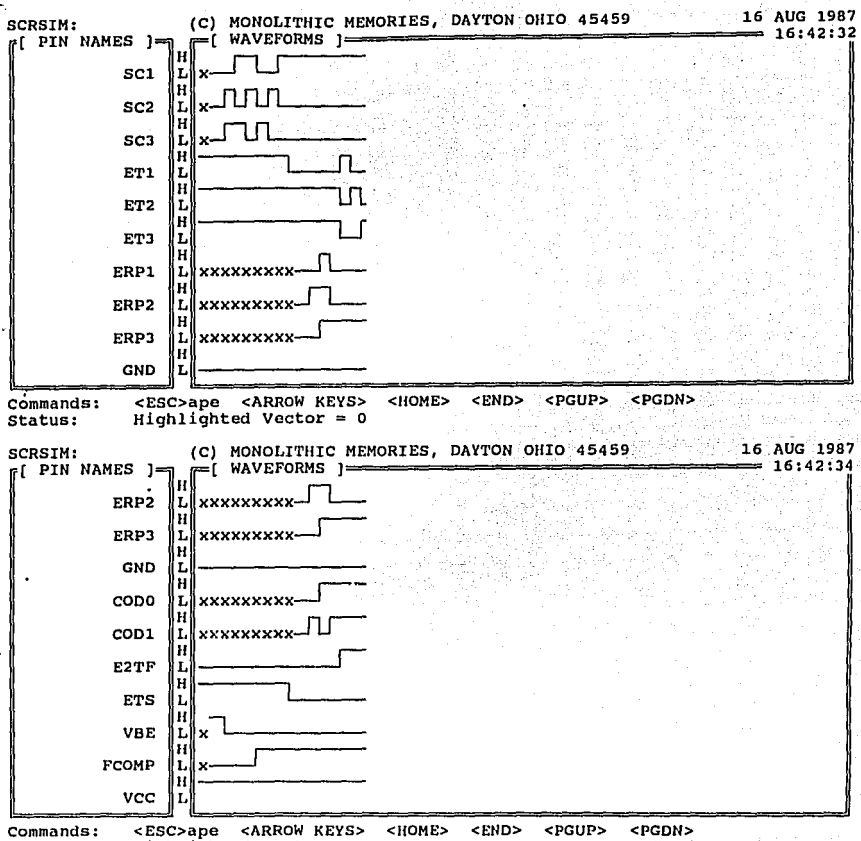


Figura 4.4.9 Simulación del ACDR.

4.3.2.2 Pruebas al microcontrolador 68HC11E1

Como se mencionó en el apartado 3.6 el μC tiene asociada electrónica para leer variables de entrada y retener variables de salida (reg. de entradas y reg. de salidas), además cuenta con registros para interactuar con UPs (reg. de estado, reg. de estado primo, reg. de estado redundante y reg. de acción) y con Gals que sirven para decodificar la memoria externa, puertos, generación de habilitadores de escritura y lectura de registros (Gal 3,4,10 y 11) así como electrónica para controlar el bus de datos del módulo LCD. Toda la electrónica descrita arriba se aprecia en el diagrama electrónico de la UDFC.

De lo anterior se deduce que las pruebas realizadas al μC consistieron en generar las respectivas señales de habilitación para capturar datos en los registros asociados con salida de información y habilitar las salidas tres estados de los registros asociados a lecturas de datos externos que llegan al μC, así como generar las distintas señales para habilitar los dispositivos externos tales como teclado matricial, pantalla de cristal líquido, etcétera.

Las herramientas utilizadas para realizar las pruebas fueron el editor de textos del DOS (se puede utilizar cualquier editor que genere un archivo de salida ASCII) para crear el programa fuente, el ensamblador AS11 de Motorola para generar el archivo **.S19**, el simulador AVSIM11 de Motorola, un programa para convertir el archivo **.S19** a un archivo **.BIN** (que viene integrado en el programador de memorias TUP400), un emulador de memorias EPROM y un osciloscopio Tektronix 11402A con ancho de banda de 200 Mhz.

El procedimiento de las pruebas se resume en los siguientes pasos :

a) Editar un programa en el editor de DOS como el que se muestra en la figura 4.5.

```
HDISPLY     EQU     $0800

            ORG     $FFFD
            JMP     $E000

            ORG     $E000

            LDAA    #$00
ET1:        STAA    HDISPLY
            BRA     ET1
            END
```

Figura 4.5 Programa fuente para generar la señal HDISPLY.

b) Salvar el archivo con extensión **.ASM** y ensamblarlo con el ensamblador AS11. Si el programa no contiene errores se genera un archivo con el mismo nombre pero con extensión **.S19**, de existir errores se tendrán que corregir éstos hasta que el proceso de ensamblado sea exitoso.

c) La simulación se realiza con el simulador AVSIM11. De ser muy extenso el programa, es recomendable dividirlo en pequeños programas que puedan ser validados por separado para después unirlos; la simulación es de vital importancia pues en esta fase se pueden detectar errores en la lógica de funcionamiento de la tarjeta o en la electrónica contenida en la tarjeta. Algunos de los errores encontrados en el diseño de la CTF se detectaron en la etapa de simulación.

d) Una vez que la simulación cumplió su propósito se obtiene el archivo **.BIN.**

e) El archivo **.BIN** se carga en el emulador de memorias EPROM cuyo conector de salida se coloca en la base de la memoria 2764 de la UDFC.

f) Antes de encender la fuente que alimenta al "backplane" donde está montada la tarjeta UDFC, se colocan las puntas de prueba del osciloscopio para observar la señal HDISPLY (Gal 10); una vez efectuada la conexión se energiza la tarjeta y se observa su comportamiento.

La mayoría de las señales de interés en esta prueba están asociadas a rutinas similares a la de la figura 4.5, por lo cual no se detalla su procedimiento de análisis. Cabe recordar que esta prueba consistió sólo en generar todas las señales asociadas con habilitación, decodificación, etc.

En el siguiente capítulo se describe el programa que controla a la CTF, el cual es ejecutado por el μC. En el capítulo seis se detallan las pruebas de integración de todos los subsistemas que componen a la CTF.

CAPITULO CINCO

PROGRAMACIÓN DE LA UDFC ORIENTADA AL CONTROL DE LA CTF

5.1 Introducción

En el capítulo tres se describieron los protocolos de comunicación entre UPs y UDFCs en este capítulo se describen los procedimientos generales de programación diseñados para los microcontroladores de las UDFCs, los que apoyan las tareas de detección, diagnóstico de fallas y la reconfiguración de la CTF. En este capítulo se presenta sólo el programa principal de control de la CTF y en el apéndice A se da la parte restante del programa que corresponde a todas las subrutinas, despliegue de mensajes y la rutina de servicio de interrupción.

En el próximo capítulo se describirán las pruebas de interacción con los demás subsistemas de la CTF.

5.2 Rutinas de control del microcontrolador 68HC11E1

Debido a que los algoritmos de control diseñados para el μC de las UDFCs son extensos, el describirlos a detalle sería bastante tedioso por lo que se presenta sólo el diagrama de flujo de estos algoritmos de control, ver figura 5.1.

Para facilitar la programación del μC, el diagrama de flujo se dividió en dos bloques :

a) Inicialización de procesadores.
b) Interacción con el usuario.

Dentro del bloque de inicialización de los procesadores se carga el sistema operativo y el programa de aplicación. Una vez efectuado esto el procesador en cuestión genera la señal FINITA# para indicar a la UDFC principal que tuvo éxito su inicialización y después entra en un estado de HALT. Acto seguido la UDFC principal proceda a inicializar al segundo procesador de manera similar a la del procesador anterior, el proceso continua hasta inicializar el tercer procesador.

En los programas de interacción con el usuario se contemplan cinco subdivisiones que corresponden a :

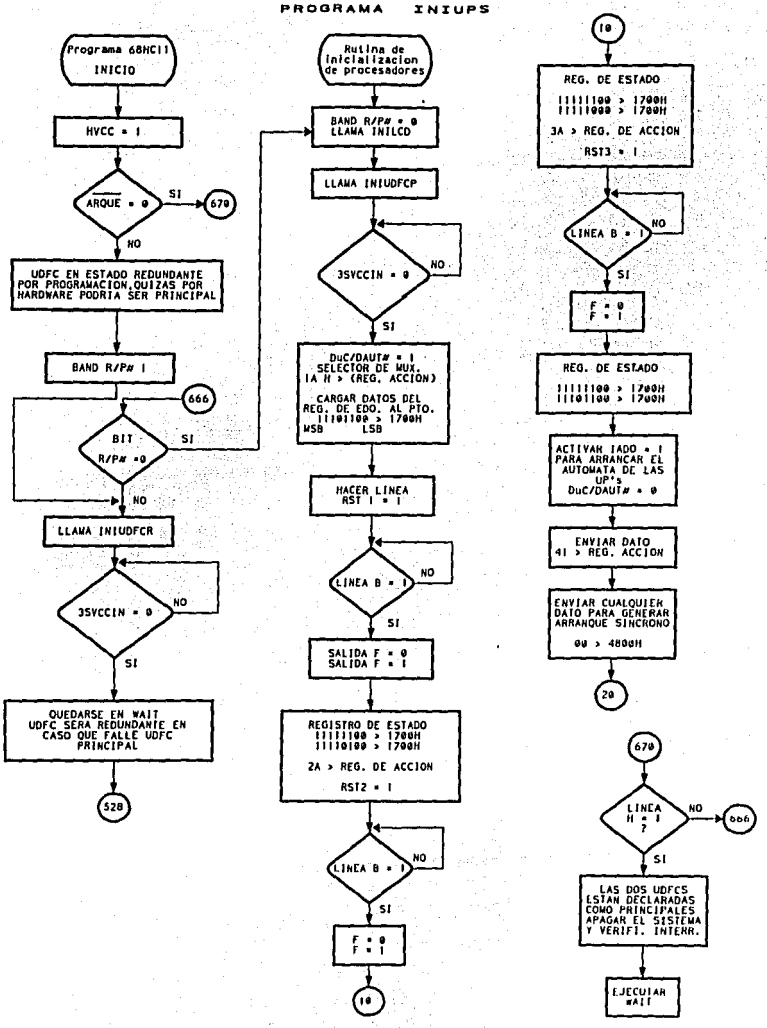
1) Cambio de categoría de procesadores.
2) Dar de baja algún procesador.
3) Dar de baja una UDFC.
4) Dar de alta algún procesador.
5) Dar de alta una UDFC.

Con estos procedimientos se redujo bastante el trabajo de programación, validación y unión de los distintos programas en los que se dividió el programa principal, para obtener así el programa que se encarga del control de la CTF.

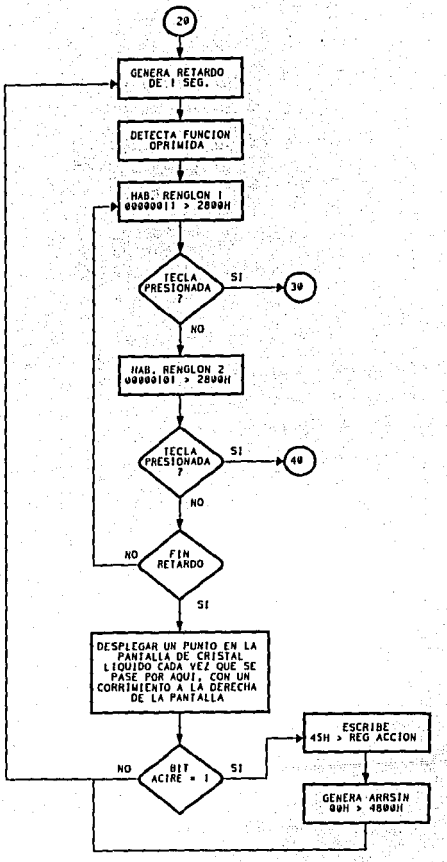
Al finalizar el diagrama de flujo se presenta el código fuente del programa principal de las rutinas de control de la UDFC.

Figura 5.1 Diagrama de Flujo de las Rutinas de Control de la CTF.

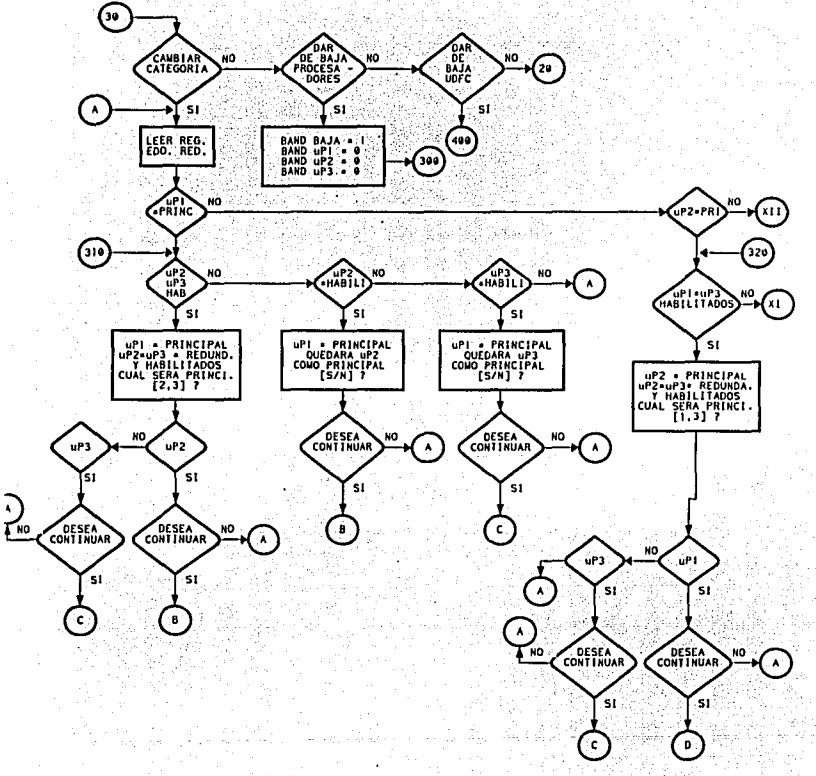
PROGRAMA INIUPS

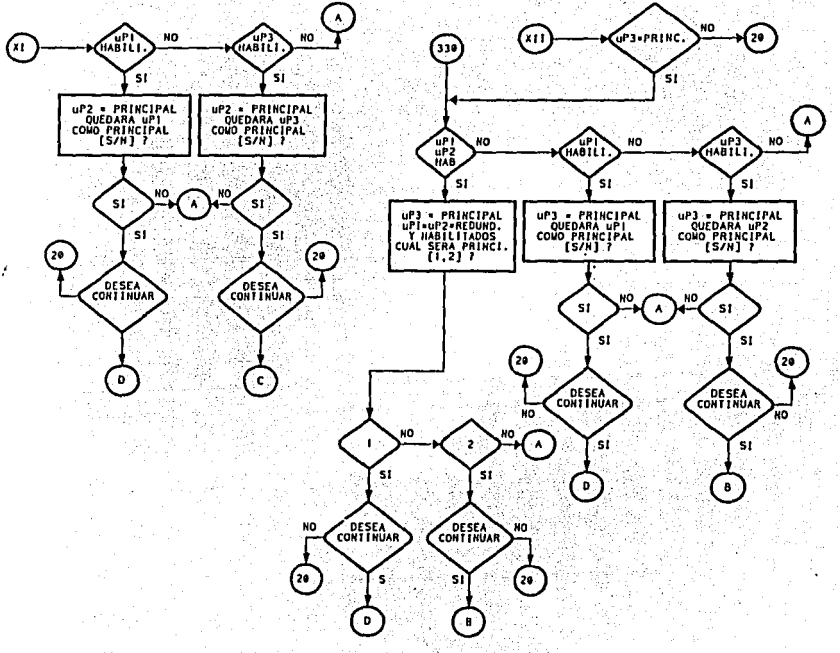


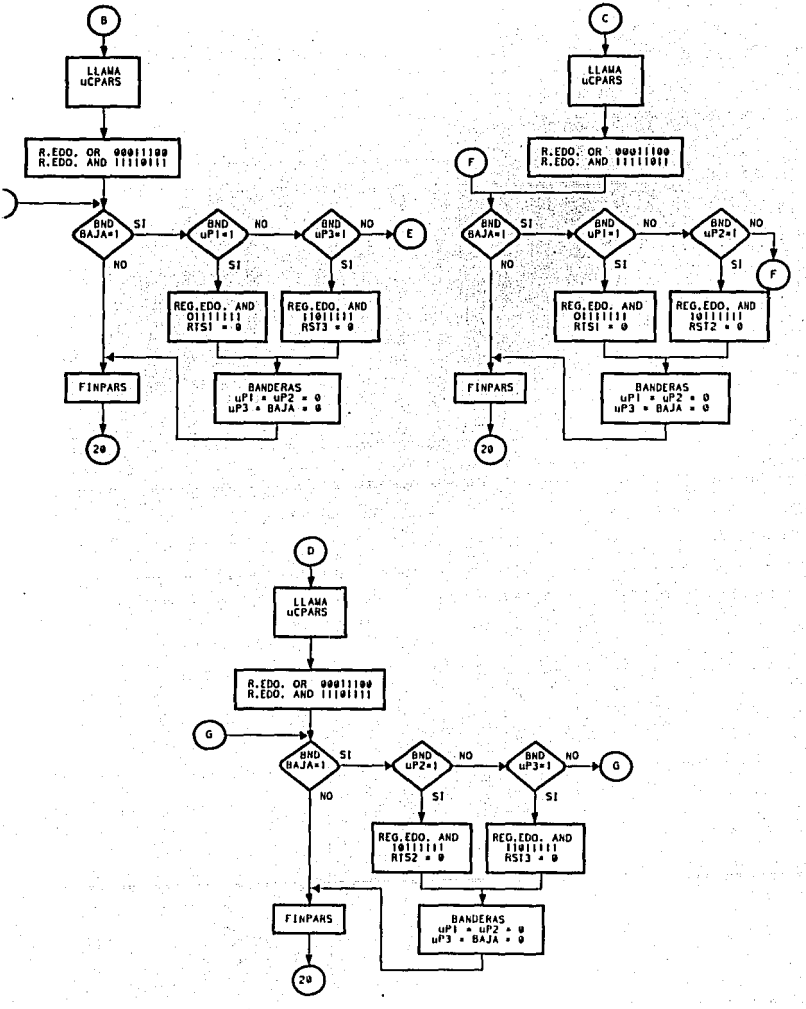
INTERACCION CON EL USUARIO



SUBDIVISIONES DEL PROGRAMA INTUSU

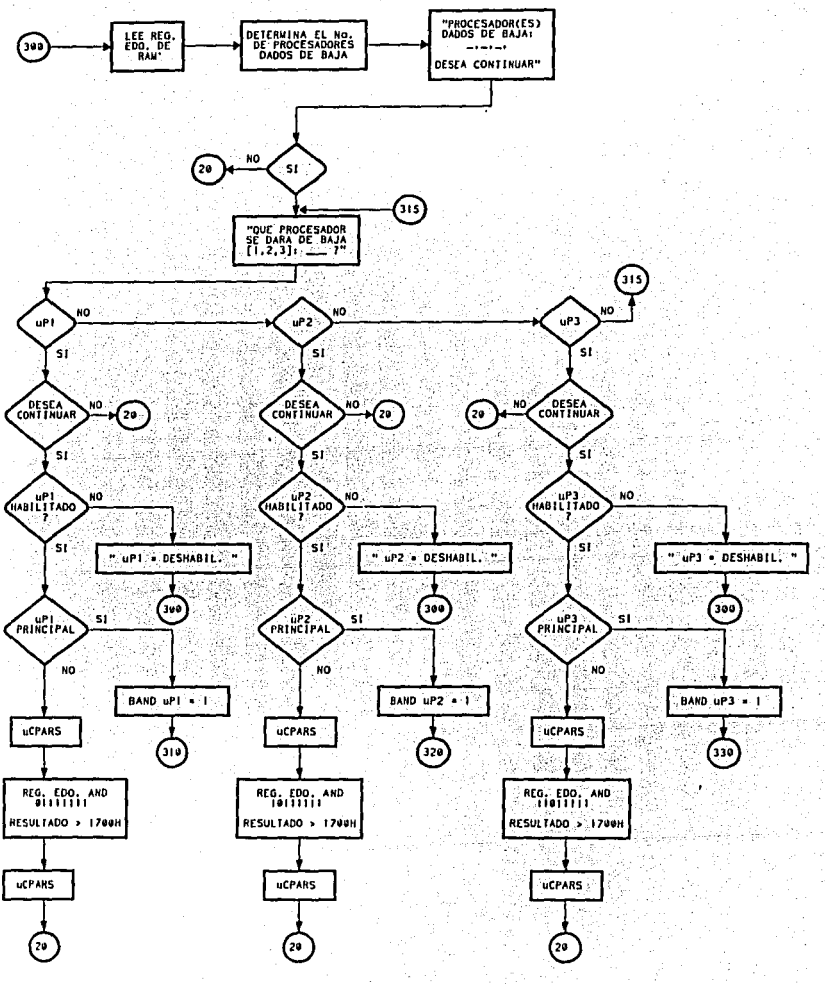


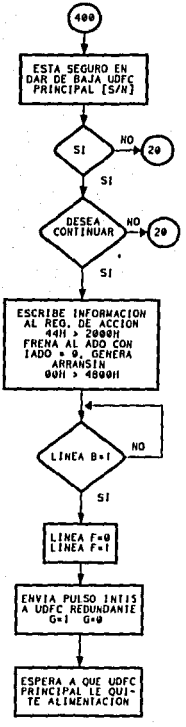

XI
uP1 HABILI.
NO
uP3 HABILI.
NO
A
SI
SI
uP2 = PRINCIPAL QUEDARA uP1 COMO PRINCIPAL [S/N] ?
uP2 = PRINCIPAL QUEDARA uP3 COMO PRINCIPAL [S/N] ?
SI
NO
A
NO
SI
SI
SI
20
20
DESEA CONTINUAR
DESEA CONTINUAR
D
C
330
XII
uP3=PRINC.
NO
20
SI
uP1 uP2 HAB
NO
uP1 HABILI.
NO
uP3 HABILI.
NO
A
SI
SI
SI
uP3 = PRINCIPAL uP1=uP2=REDUND. Y HABILITADOS CUAL SERA PRINCI. [1,2] ?
uP3 = PRINCIPAL QUEDARA uP1 COMO PRINCIPAL [S/N] ?
uP3 = PRINCIPAL QUEDARA uP2 COMO PRINCIPAL [S/N] ?
SI
NO
A
NO
SI
SI
SI
20
20
DESEA CONTINUAR
DESEA CONTINUAR
NO
NO
D
B
1
NO
2
NO
A
SI
SI
NO
DESEA CONTINUAR
DESEA CONTINUAR
NO
20
S
SI
20
D
B


B
LLAMA uCPARS
R.EDO. OR 00011100
R.EDO. AND 11110111
BND BAJA=1
SI
NO
BND uP1=1
SI
NO
BND uP3=1
SI
NO
E
REG.EDO. AND 01111111 RTS1 = 0
REG.EDO. AND 11011111 RST3 = 0
FINPARS
BANDERAS uP1 = uP2 = 0 uP3 = BAJA = 0
20
C
LLAMA uCPARS
R.EDO. OR 00011100
R.EDO. AND 11111011
F
BND BAJA=1
SI
NO
BND uP1=1
SI
NO
BND uP2=1
SI
NO
F
REG.EDO. AND 01111111 RTS1 = 0
REG.EDO. AND 10111111 RST2 = 0
FINPARS
BANDERAS uP1 = uP2 = 0 uP3 = BAJA = 0
20
D
LLAMA uCPARS
R.EDO. OR 00011100
R.EDO. AND 11101111
G
BND BAJA=1
SI
NO
BND uP2=1
SI
NO
BND uP3=1
SI
NO
G
REG.EDO. AND 10111111 RTS2 = 0
REG.EDO. AND 11011111 RST3 = 0
FINPARS
BANDERAS uP1 = uP2 = 0 uP3 = BAJA = 0
20

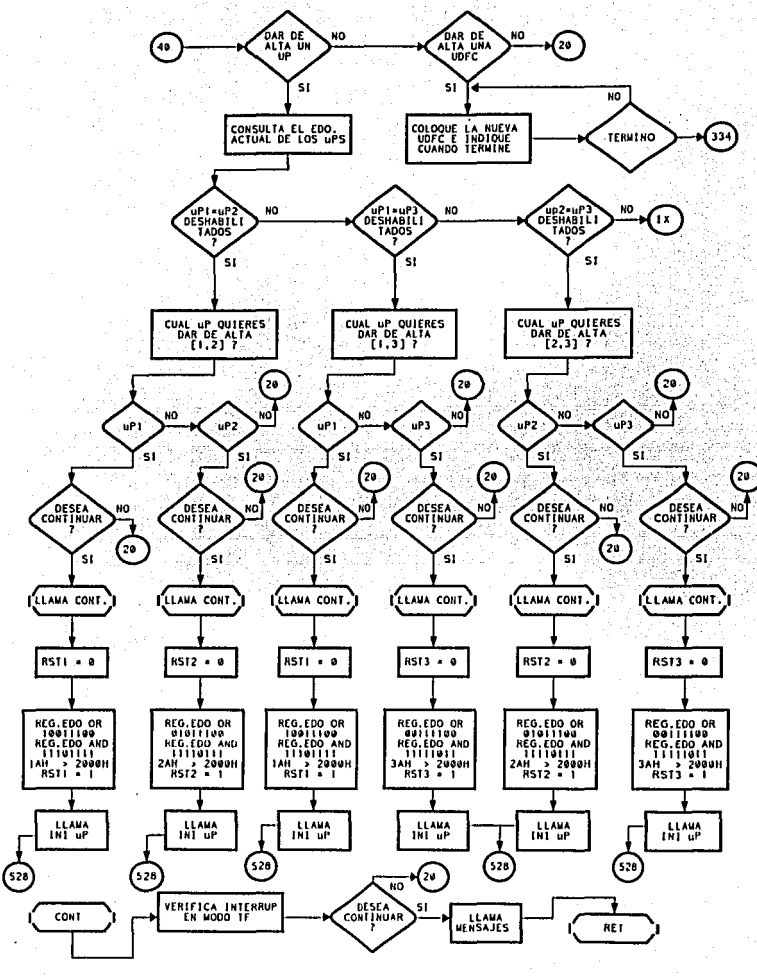
TESIS SIN PAGINACION

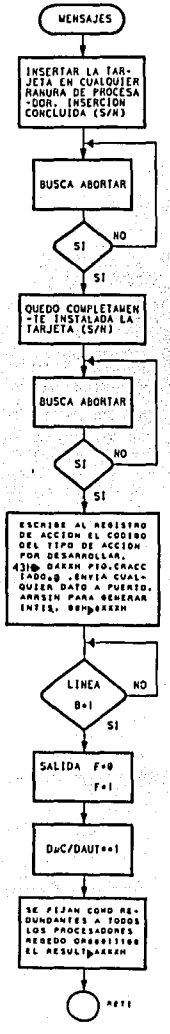
COMPLETA LA INFORMACION


300
LEE REG. EDO. DE RAM
DETERMINA EL No. DE PROCESADORES DADOS DE BAJA
"PROCESADOR(ES) DADOS DE BAJA: ---,---,---
DESEA CONTINUAR"
NO
SI
20
315
"QUE PROCESADOR SE DARA DE BAJA [1,2,3]: ___ ?"
uP1
uP2
uP3
DESEA CONTINUAR
uP1 HABILITADO ?
uP2 HABILITADO ?
uP3 HABILITADO ?
" uP1 = DESHABIL. "
" uP2 = DESHABIL. "
" uP3 = DESHABIL. "
300
uP1 PRINCIPAL
uP2 PRINCIPAL
uP3 PRINCIPAL
uCPARS
BAND uP1 = 1
BAND uP2 = 1
BAND uP3 = 1
310
320
330
REG. EDO. AND 01111111
REG. EDO. AND 10111111
REG. EDO. AND 11011111
RESULTADO > 1700H

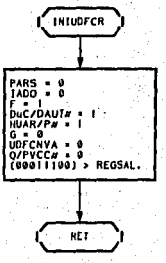

400
ESTA SEGURO EN DAR DE BAJA UDFC PRINCIPAL [S/N]
SI
NO
20
SI
DESEA CONTINUAR
NO
20
SI
ESCRIBE INFORMACION AL REG. DE ACCION 44H > 2000H FRENA AL ADO CON IADO = 0, GENERA ARRANSIN 00H > 4800H
LINEA B=1
NO
SI
LINEA F=0
LINEA F=1
ENVIA PULSO INT15 A UDFC REDUNDANTE G=1 G=0
ESPERA A QUE UDFC PRINCIPAL LE QUI- TE ALIMENTACION

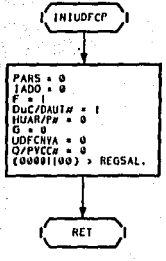
RUTINA PARA DAR ALTA UN UP O UNA UDFC

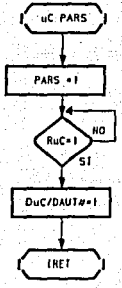


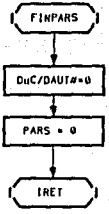

MENSAJES
INSERTAR LA TARJETA EN CUALQUIER RANURA DE PROCESADOR. INSERCION CONCLUIDA (S/N)
BUSCA ABORTAR
SI
NO
SI
QUEDO COMPLETAMENTE INSTALADA LA TARJETA (S/N)
BUSCA ABORTAR
SI
NO
SI
ESCRIBE AL REGISTRO DE ACCION EL CODIGO DEL TIPO DE ACCION POR DESARROLLAR.
431▶ 0AXXH PIO.CARACTADO=0 .ENVIA CUALQUIER DATO A PUERTO. AMBOS PARA GENERAR INTRS. 00H▶0AXXH
LINEA B=1
NO
SI
SALIDA F=0
F=1
DMC/DAUT=+1
SE FIJAN COMO REDUNDANTES A TODOS LOS PROCESADORES REGEDO 0A6011100 EL RESULT▶AAXXH
A011
>>4
UDFCNVA=1
QVCC/PVCC=+0
ESPERAR 3 SEG
QVCC/PVCC=+1
UDFCNVA=0
20

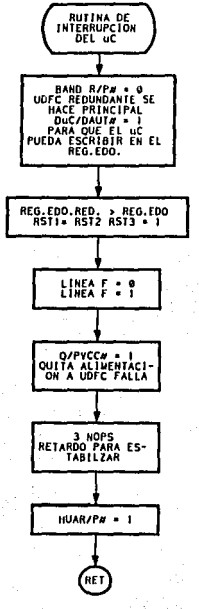
SUBRUTINAS DE USO GENERAL

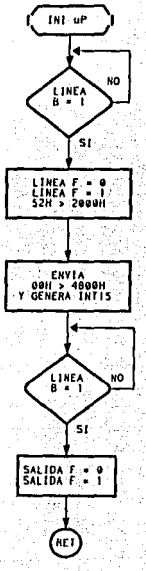









RUTINA DE INTERRUPCION DEL uC
BAND R/P# = 0 UDFC REDUNDANTE SE HACE PRINCIPAL DuC/DAUT# = 1 PARA QUE EL uC PUEDA ESCRIBIR EN EL REG.EDO.
REG.EDO.RED. > REG.EDO RST1= RST2 RST3 = 1
LINEA F = 0 LINEA F = 1
O/PVCC# = 1 QUITA ALIMENTACION A UDFC FALLA
3 NOPS RETARDO PARA ESTABILIZAR
HUAR/P# = 1
RET


INI uP
LINEA B = 1
NO
SI
LINEA F = 0 LINEA F = 1 52H > 2000H
ENVIA 00H > 4000H Y GENERA INT15
LINEA B = 1
NO
SI
SALIDA F = 0 SALIDA F = 1
RET

```
                    **** PROGRAMA PRINCIPAL DE LA CTF ****

                ** DECLARACIÓN DE LAS VARIABLES INVOLUCRADAS **

ARRSIN      EQU   $4800       ; VA DE $4800 - $4FFF
HDISPLY     EQU   $0800       ; VA DE $0800 - $0FFF
HLMC        EQU   $1700       ; VA DE $1010 - $17FF
CRER        EQU   $1800       ; VA DE $1800 - $1FFF
CRACC       EQU   $2000       ; VA DE $2000 - $27FF
HREN        EQU   $2800       ; VA DE $2800 - $2FFF
REGSAL      EQU   $3000       ; VA DE $3000 - $37FF
REGENT      EQU   $3800       ; VA DE $3800 - $3FFF
HPROM       EQU   $E000       ; VA DE $E000 - $FFFF
LAREN       EQU   $4000       ; VA DE $4000 - $47FF
MANDAA      EQU   $0801
PORTA       EQU   $1000
TCNT        EQU   $100E
TOC1        EQU   $1016
TFLG1       EQU   $1023
TMSK2       EQU   $1024
HCOL        EQU   $4000
REGEDO      EQU   $0010
BANDERAS    EQU   $0012
LAUX        EQU   $0014
LAUX1       EQU   $0016
CONT1       EQU   $0017
CONT2       EQU   $0018
CONT3       EQU   $0019

        *** CONFIGURACIÓN DEL 68CH11E1 PARA EL CONTROL DE LA UDFC ****

            ORG     $FFFD       ; Instrucción para fijar la dirección
            JMP     $E000       ; del vector de reset

            ORG     $E000       ; Dirección de inicio de inicio de prog. fuente
            LDS     #$01FF      ; Donde será almacenado el SP

            JSR     INILCD      ; Rutina para inicializar pantalla
            CLI                 ;
            LDAA    #$88        ; Configuramos PA3 Y PA7 como salidas
            STAA    $1026

            LDAA    #$00        ; Localidad en RAM BAJA = MP1 = MP2 = MP3 = 0
            STAA    BANDERAS

            LDAA    #$78
            STAA    PORTA       ; Hacemos HVCC = RST1 = RST2 = RST3 = 1

        ******** INICIALIZACIÓN DE LOS TRES PROCESADORES ********

            LDAA    REGENT      ; Obtenemos linformación de REGENT
            ANDA    #$10
            CMPA    #$00
```

```
        BNE     ETQ1        ; ARQUE = 1
        BRA     ETQ2        ; ARQUE = 0

ETQ2:   LDAA    REGENT
        ANDA    #$02
        CMPA    #$00
        BNE     ETQ3        ; H = 1
        BRA     ETQ4        ; H = 0

ETQ3:   JSR     INITIMER    ; Retardo para estabilizar teclado
        JSR     MS25        ; Despliega mensaje 25
        WAI

ETQ1:   LDAA    #$10
        STAA    BANDERAS    ; Hace BAND R/P# = 1
        BRA     ABAJO

ETQ4:   LDAA    REGENT      ; Pregunta por R/P#
        ANDA    #$01
        CMPA    #$00        ; R/P# = 0
        BEQ     INIUPS
        BRA     ABAJO

ABAJO:JSR       INIPERR     ; Esta UDFC sabe que es redundante
        JSR     VCCIN
        WAI

        ********** INICIALIZA AL PROCESADOR UNO **********

INIUPS:JSR      INIPERP     ; Esta UDFC sabe que es principal
        JSR     VCCIN
        LDAA    #$1A        ;Cargamos registro de acción ( UP # 1 )
        STAA    CRACC

        LDAA    #$EC        ;Define categoría y estado de UP'S
        STAA    HLMC
        STAA    REGEDO
        LDAA    #$70        ; Quitamos reset a UP# 1
        STAA    PORTA

        JSR     TORR        ; Pregunta por línea " B " y cambia nivel de la línea " F "

        ********** INICIALIZA AL PROCESADOR DOS **********

        LDAA    #$2A        ;Cargamos registro de acción ( UP # 2 )
        STAA    CRACC

        LDAA    REGEDO      ; Recuperamos información del REGEDO.

        ORA     #$10        ; REGEDO. ( 111 111 00 )
        STAA    HLMC
```

```
        ANDA    #$F4        ; REGEDO. ( 111 101 00 )
        STAA    HLMC

        STAA    REGEDO

        LDAA    #$60        ; Quitamos reset a  UP# 2
        STAA    PORTA

        JSR     TORR        ; Pregunta por linea " B " y cambia nivel de linea " F "

        ********** INICIALIZA AL PROCESADOR TRES **********

        LDAA    #$3A        ;Cargamos registro de acción  ( UP # 3 )
        STAA    CRACC

        LDAA    REGEDO      ; Recuperamos información del  REGEDO

        ORA     #$08        ; REGEDO. ( 111 111 00 )
        STAA    HLMC

        ANDA    #$F8        ; REGEDO. ( 111 110 00 )
        STAA    HLMC

        STAA    REGEDO

        LDAA    #$40        ; Quitamos reset a  UP# 3
        STAA    PORTA

        JSR     TORR        ; Preguntamos por linea " B " y cambia  nivel a linea  " F "

        LDAA    REGEDO      ; Recuperamos información del  REGEDO

        ORA     #$04        ; REGEDO. ( 111 111 00 )
        STAA    HLMC

        ANDA    #$EC        ; REGEDO. ( 111 011 00 )
        STAA    HLMC

        STAA    REGEDO

CAR:    LDAA    #$06        ; Hacemos  IADO = 1  Y  DMC/DAUT# = 0
        STAA    REGSAL

        LDAA    #$41        ; Prepara a las  3  UP'S  para arranque  sincrono
        STAA    CRACC

        LDAA    #$00        ; Manda la senal  ARRSIN  a  UP'S
        STAA    ARRSIN

            ****** RUTINA  DE  INTERACCIÓN  CON  EL  USUARIO ******

INTUSU:LDAB     #$00        ; Retraso de 1 seg. para despliegue de puntos en LCD
```

```
INICIO: PSHB
        JSR     INITIMER    ; Inicializamos el TIMER
LET1:   JMP     HAREN1      ; Habilita rebglón 1 del teclado TF
LET2:   JMP     HAREN2      ;     "      "   2 "      "
LET3:   JSR     PANT        ; Escribe un "." en LCD
        PULB
        INCB
        CMPB    #$20
        BEQ     LIMLCD      ; Limpia LCD
        CMPB    #$10
        BEQ     LIN2        ; Salta a línea 2 de LCD
        JSR     TAREA       ; Lee REGENT , Act. REGEDO yproduce ARRSIN
        BRA     INICIO
LIMLCD:PSHB
        LDAA    #$01        ; Limpia LCD
        STAA    HDISPLY
        JSR     RETARDO
        PULB
        BRA     INTUSU
LIN2:   PSHB
        LDAA    #$C0        ; Comando para poner cursor en
        STAA    HDISPLY     ; línea 2 de LCD
        JSR     RETARDO
        PULB
        BRA     INICIO
HAREN1:LDAA     #$03        ; Habilitamos renglón 1
        STAA    HREN
RTY:    LDAA    HCOL        ; Leemos teclado TF
        TAB
        ANDB    #$07
        CMPB    #$06
        BEQ     JUN1        ; Se oprimió cambiar categoría
        CMPB    #$05
        BEQ     JUN2        ; Se oprimió baja procesador
        CMPB    #$03
        BEQ     JUN3        ; Se oprimió baja UDFC
        CMPB    #$07
        BEQ     LET2        ; No se oprimió tecla
        BRA     RTY
HAREN2:LDAA     #$05        ; Habilitamos renglón 2
        STAA    HREN
QWE:    LDAA    HCOL        ; Leemos teclado TF
```

```
        TAB
        ANDB    #$07
        CMPB    #$05
        BEQ     JUN4        ; Se oprimió alta procesador
        CMPB    #$03
        BEQ     JUN5        ; Se oprimió alta  UDFC
        CMPB    #$07
        BEQ     LET3        ; No se oprimió  tecla
        BRA     QWE

JUN1:   JSR     CAMCAT      ; Se oprimió cambiar categoria
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

JUN2:   JSR     BAJAPROC    ; Se oprimió dar de baja procesador
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

JUN3:   JSR     BAJAUDFC    ; Se oprimió dar de baja  UDFC
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

JUN4:   JSR     ALTAPROC    ; Se oprimió dar de alta procesador
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

JUN5:   JSR     ALTAUDFC    ; Se oprimió dar de alta  UDFC
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU
```

PRUEBAS PRELIMINARES DE INTEGRACIÓN, CON TODOS LOS MÓDULOS QUE CONTEMPLA LA ARQUITECTURA DE LA CTF

6.1 Introducción

Todo proyecto de ingeniería culmina con la puesta en marcha del prototipo desarrollado. En el caso de la CTF se iniciaron las pruebas, las cuales consistieron en interconectar algunas de las tarjetas en los ductos y seguir paso a paso la ejecución de las rutinas de control de la CTF. Además se corroboró el funcionamiento real de todos los autómatas incluidos en la electrónica de la computadora; parte de las pruebas se realizaron fuera del gabinete que integrará a las 11 tarjetas que componen la arquitectura, en tanto que otras se realizaron dentro del gabinete (incluyendo TA y tarjetas objetivo) con el fin de asegurar que las tarjetas no tuvieran movimiento, es decir, para evitar que hubiera falsos contactos en los peines de las tarjetas.

En este capítulo se describirán cada una de las pruebas realizadas y se comentarán los resultados obtenidos. En el siguiente y último capítulo se darán las conclusiones de ese trabajo así como algunas recomendaciones.

6.2 Pruebas entre UDFC y UIM

La primer prueba de integración realizada fue entre una UDFC y la UIM, con esta prueba se validó todo el programa de control de la UDFC desde la inicialización de los procesadores (INIUPS) hasta la rutina de servicio de interrupción, pasando por todas las posibilidades de reconfiguración, tales como cambiar categorías de procesadores, dar de baja un procesador, dar de baja una UDFC, dar de alta un procesador y dar de alta una UDFC. Todos estos procedimientos se encuentran en el bloque de interacción con el usuario (INTUSU).

Se reitera que la división hecha en el bloque INTUSU redujo enormemente el trabajo de programación; a continuación se describirán los pasos de la prueba realizada. Para llevar a cabo la prueba se necesitaron de algunas señales que vienen de los otros subsistemas, las cuales, debido a que en ese momento no estaban disponibles, se tuvieron que simular.

La simulación de las señales se realizó mediante un arreglo electrónico montado en una tarjeta para prototipos, en la cual se instalaron un interruptor Dip, un CI

7404, leds, resistencias, cable plano y conector para 12 bits. El arreglo se muestra en la figura 6.1

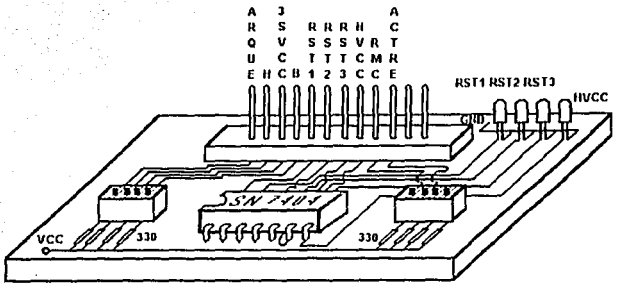


Figura 6.1 Arreglo electrónico para simulación de señales.

Las señales que se simularon fueron : ARQUE, 3SVCCIN, Línea " B ", ACTRE, RµC. También se obtuvieron señales del µC que se llevaron a los leds de la tarjeta prototipo (RST1, RST2, RST3 Y HVCC) para hacer un seguimiento de la ejecución del programa, además se utilizó el osciloscopio para visualizar las señales generadas por el microcontrolador.

Para definir el estado lógico de una señal (" 0 " ó " 1 ") se desplazaba el interruptor correspondiente a la posición ON o OFF. Una vez validados los niveles de voltaje a las salidas de los interruptores se procedió a cargar el programa INIUPS.S19 en el emulador de memoria y conectarlo a la base de la memoria 2764 para correrlo.

Para corroborar la ejecución del programa se observaron las señales de categoría y estado en el osciloscopio. El procedimiento empleado fue similar al utilizado para la señal HDISPLY que se vio en el capítulo cuatro. También se hicieron pruebas encendiendo y apagando los leds de la tarjeta prototipo, e insertando estados de espera (WAIT) al microcontrolador con el fin de seguir paso a paso todas las rutinas que marca el diagrama de flujo del capítulo cinco.

Terminada la validación del programa INIUPS se prosiguió con el programa INTUSU.S19, en el cual se analizaron por separado cada uno de los bloques en que se particionó. Para llevar a cabo esta prueba no se tuvo que agregar

electrónica a la tarjeta prototipo ya que muchas de las rutinas contemplan el despliegue de mensajes en el módulo LCD, por lo que el osciloscopio y el envío correcto de los mensajes fueron las únicas herramientas para validar cada programa.

Debido a que muchos de los mensajes que aparecen en la pantalla de cristal líquido sobrepasan la capacidad de despliegue (2 líneas x 16 caracteres) al final de cada pantalla aparece el carácter flecha (>, el cual se asocia con el botón marcado con el número 4 del teclado TF) el cual deberá ser oprimido para continuar con el despliegue del mensaje.

Los resultados fueron exitosos ya que el módulo LCD ejecutó correctamente los programas de inicialización de pantalla, envío de palabra de control, envió del código ASCII de los mensajes, así como los retardos para recepción de información proveniente del teclado TF.

6.3 Pruebas entre TAUDFCs, UDFCs y UIM

En el apartado anterior se describió la prueba entre UDFC y la UIM, en esta sección se describe la prueba en la que se anexa una TA con el fin de observar el comportamiento de las TAUDFCs asociadas con una UDFC principal o redundante. Debido a que en la prueba anterior los programas INIUPS e INTUSU operaron correctamente, al agregar la TA dichos programas deberían de ejecutarse sin ningún contratiempo, pudiendo observar señales tales como RSTi, HDISPLY, DμC 0-7, etc. las cuales se reflejan en los ductos y en el despliegue del módulo LCD.

La prueba genera la señal P/R# = 0, ver tabla 4.1, la que obliga a la TA a operar como TAUDFC redundante y por lo tanto impide que los programas INIUPS e INTUSU corran. Después se cambio el estado lógico de la señal, P/R# = 1, provocando que la TA trabajara como TAUDFC principal, permitiendo ello que los programas de control de la UDFC se ejecutaran de manera similar a la de la prueba descrita en la sección anterior. Al realizar esta prueba, se llevó a cabo de manera implícita la prueba de aislamiento de señales del ducto TF

Una prueba que deberá realizarse consiste en agregar otra TAUDFC y otra UDFC en el "back plane", para correr de manera independiente las rutinas de UDFC principal y UDFC redundante, así como también la de servicio de interrupción (cuya validación se explicó en la sección 6.2), con dicha prueba se espera obtener la correcta inicialización de UDFCs y la eficiente conmutación entre UDFC principal y UDFC redundante.

6.4 Pruebas entre TAUDFCs, UDFCs, UPs y UIM

Como se dijo en el capítulo dos, a las UPs se les verá como las tarjetas que proveen la información que ha de ser comparada en las UDFCs, de ahí que esta prueba tiene como principal objetivo el que las UPs (a través de sus TA) proporcionen datos (involucrados con accesos a memoria o puertos) a la UDFC principal a través de su TA y que ésta tarjeta realice la comparación de datos, el diagnóstico y la reconfiguración del sistema TF en caso de ser necesario. Cuando esta prueba se realice y sus resultados sean exitosos, se confirmará la validación del ACDR (visto en el capítulo cuatro) y también permitirá validar el aislamiento de las señales del ducto AT que se mencionó en el capítulo cuatro.

Cuando se realice esta prueba, deliberadamente se cargarán programas ligeramente diferentes en las unidades de procesamiento para que se generen errores en sitios conocidos, además, esto obligará el que se produzca la señal de falla correspondiente provocando una reconfiguración del sistema. Una vez que la CTF opere de la manera esperada se obtendrán estadísticas para validar el modelo matemático de la computadora, este último fue desarrollado ya en un trabajo previo a este.

CAPITULO SIETE

CONCLUSIONES Y RECOMENDACIONES

Del trabajo realizado en esta tesis se concluye lo siguiente :

- Considerando que se tiene un porcentaje de avance total del proyecto de un 85 %, el prototipo de la computadora tolerante a fallas reúne todos y cada uno de los atributos establecidos como objetivos al principio del proyecto.

- En cuanto al ensamblado este, pudo realizarse de manera relativamente fácil debido a la moderada densidad de componentes de cada tarjeta. Gracias a esto las señales que viajan en las tarjetas se propagan de manera fiel y sin problemas significativos de ruido, logrando así un alto desempeño en el funcionamiento de la CTF.

- En lo que respecta a las pruebas realizadas se puede subrayar que su desarrollo físico fue sencillo debido a que el diseño de todas las tarjetas facilitó la localización de señales y la ubicación de las puntas de prueba del osciloscopio. Para el lector esto pudiera parecer trivial, pero se necesitaron bastantes horas-hombre para que el diseño final del PCB de las tarjetas brindara esta característica durante las pruebas.

- Las técnicas utilizadas para tolerar fallas y los mecanismos para la reconfiguración de la CTF, permitieron al autor obtener un panorama bastante amplio sobre los diferentes campos del computo TF, además de fomentar inquietudes y un horizonte muy extenso en la creación de nuevas arquitecturas de cómputo en un gran número de aplicaciones reales de distintos campos de la ingeniería.

- En cuanto a la programación del microcontrolador, se crearon aproximadamente 1500 líneas de código fuente, mediante las cuales se pudo establecer el control de los procedimientos de mantenimiento y reconfiguración de la CTF. Esto, aunado a las características de la arquitectura de la computadora, le permitió al autor aplicar todos los conocimientos adquiridos en estudios previos de microcontroladores así como procurar el eficiente uso de recursos.

Las recomendaciones que emanan del presente trabajo son las siguientes :

1. Debido a que ninguno de los microcontroladores integrados en la UDFC posee redundancia y ellos pueden incurrir en algún error que provoque una avería en el sistema (probabilidad latente), se propone la adición de un microcontrolador PIC de Microchip para supervisar al 68HC11E1; algunas de las ventajas que ofrecen estos µC's son: menor tamaño, mayor rapidez en la ejecución de sus instrucciones, arquitectura "RISC-Like" y un número reducido de instrucciones que no complica su programación.

   El µC PIC se encargaría de supervisar todos los periféricos del 68HC11E1 y correr algunas rutinas para envío y recepción de información entre ellos, de encontrar fallas en el HC11 se enviarán alarmas auditivas para reportar la falla.

2. Debido a que las acciones que se tienen contempladas en los protocolos de comunicación entre UDFC's y UP's no han sido saturadas, alguna(s) de las restantes pueden ser utilizadas como protocolos de comunicación entre microcontroladores.

3. Al agregar más componentes y más atributos a la CTF, se eleva el número de líneas de programación en lenguaje ensamblador, por lo que se recomienda el uso de algún ensamblador cruzado (software de programación que utiliza el lenguaje "C" para crear el código fuente del HC11) para facilitar y reducir el número de líneas de programa fuente y el tiempo de programación.

4. Si se desea reducir el tamaño de las tarjetas de la CTF ya sea para la adición de más redundancia o bien para poder ser comercializada se recomienda el diseño de tarjetas multicapa, uso extensivo de dispositivos programables (tipo FPGA) y el uso de componentes del tipo de montado superficial. Lo mismo se recomienda para las tarjetas de aislamiento.

PROGRAMA DE CONTROL DE LA UDFC

En este apéndice se muestra la parte de subrutinas correspondientes al programa principal de la CTF, estas rutinas (por medio del teclado matricial para mantenimiento en línea o para propósitos de demostración). son las encargadas de realizar los cambios de los bits de estado y categoría de las UPs instaladas en la CTF. Además contiene las rutinas de control para el despliegue de mensajes en el módulo LCD, las tablas de mensajes, así como la rutina de servicio de interrupción.

```
****************** SUBRUTINAS ****************

************ DAR DE ALTA UNA UDFC ****************

ALTAUDFC:JSR    INITIMER    ; Retraso para estabilizar teclado TF
         LDAA   HCOL
         ANDA   #$04
         BEQ    PAPE         ; Se oprimió dar de alta UDFC
         JMP    RTY

PAPE:    JSR    VENTA        ; Mensaje " COLOQUE LA NUEVA TARJETA E
                             ; INDIQUE..."
         JSR    ESCAPE
         LDAA   LAUX1
         TAB
         ANDB   #$02
         BEQ    PLUM         ; Se oprimió  " S "
         TAB
         ANDB   #$04
         BEQ    PAPE         ; Se    "     " N "
         ANDA   #$01
         BEQ    PUNT         ; Se    "     CANCELAR

PUNT:    LDS    #$01FF
         JSR    CLCD
         JMP    INTUSU

PLUM:    LDAA   #$44
         STAA   REGSAL       ; Hacemos UDFCNVA = 1 y QVcc/PVcc = 0
         JSR    WSX          ; Retardo de 3 seg.
         LDAA   #$84
         STAA   REGSAL       ;Hacemos  QVcc/PVcc = 1 y UDFCNVA = 0
         RTS

VENTA:JSR       CLCD         ; Limpia pantalla
```

```
        LDX     #MENS21     ; Rutina de control del mensaje 21
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS21A
        JSR     PRESEN
        JSR     FLECHA
        LDX     #MENS21B
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS21C
        JSR     PRESEN
        LDX     #PETICIONA
        JSR     PRESEN
        RTS

WSX:    LDX     #$0007      ; Rutina de retraso
UHV:    PSHX
        JSR     INITIMER
        PULX
        DEX
        BNE     UHV
        RTS

                ************** DAR DE ALTA UN PROCESADOR **************

ALTAPROC:JSR    INITIMER    ; Retraso para estabilizar teclado TF
        LDAA    HCOL
        ANDA    #$02
        BEQ     CUBO        ; Se oprimió dar de alta procesador
        JMP     RTY

CUBO:   LDAA    REGEDO      ; Consulta estadode los mP's
        TAB
        ANDB    #$E0
        CMPB    #$20
        BEQ     APAR1       ; Sabe que mP1 y mP2 estan deshabilitados
        CMPB    #$40
        BEQ     APAR2       ;  "   "  mP1 y mP3  "      "
        CMPB    #$80
        BEQ     CBA         ;  "   "  mP2 y mP3  "      "
        CMPB    #$60
        BEQ     ABC         ;  "   "  mP1 esta deshabilitado
        CMPB    #$A0
        BEQ     ACD         ;  "   "  mP2  "      "
        CMPB    #$C0
        BEQ     ADE         ;  "   "  mP3  "      "
        JMP     INTUSU
CBA:    JMP     APAR3
ABC:    JMP     APAR4
ACD:    JMP     APAR5
ADE:    JMP     APAR6

APAR1:JSR       L1          ; Mensaje " CUAL PROCESADOR QUIERES DAR
                            ; DE ALTA "
```

```
        JSR     L1A
        JSR     RETARDO
        JSR     OPCION2     ; Opción para elegir mP1 o mP2
        JSR     CLCD
        JSR     OPRI        ; Mensaje " DESEA CONTINUAR "
        JSR     ESCAPE
        LDAA    LAUX
        TAB
        ANDB    #$01
        BEQ     ZXC         ; Se eligió mP1
        ANDA    #$02
        BEQ     XCZ         ; Se eligió mP2
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

APAR2:JSR       L1          ; Mensaje " CUAL PROCESADOR QUIERES DAR
                            ; DE ALTA "
        JSR     L1B
        JSR     RETARDO
        JSR     OPCION1     ; Opción para elegir mP1 o mP3
        JSR     CLCD
        JSR     OPRI        ; Mensaje " DESEA CONTINUAR "
        JSR     ESCAPE
        LDAA    LAUX
        TAB
        ANDB    #$01
        BEQ     ZXC         ; Se eligió mP1
        ANDA    #$04
        BEQ     CXZ         ; Se eligió mP3
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

APAR3:JSR       L1          ; Mensaje " CUAL PROCESADOR QUIERES DAR
                            ; DE ALTA "
        JSR     L1C
        JSR     RETARDO
        JSR     OPCION      ; Opción para elegir mP2 o mP3
        JSR     CLCD
        JSR     OPRI        ; Mensaje " DESEA CONTINUAR "
        JSR     ESCAPE
        LDAA    LAUX
        TAB
        ANDB    #$02
        BEQ     XCZ         ; Se eligió mP2
        ANDA    #$04
        BEQ     CXZ         ; Se eligió mP3
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU
ZXC:    JMP     MAR1
XCZ:    JMP     MAR2
CXZ:    JMP     MAR3

APAR4:JSR       L2          ; Mensaje " PROCESADOR "
```

```
        JSR     G1
        JSR     L3          ; Mensaje " SE DARA DE ALTA "
        JSR     ESCAPE
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     MAR1        ; Se oprimio " S "
        ANDA    #$04
        BEQ     ENVIA       ; Se    "    " N "

APAR5:JSR       L2          ; Mensaje " PROCESADOR "
        JSR     G2
        JSR     L3          ; Mensaje " SE DARA DE ALTA "
        JSR     ESCAPE
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     MAR2        ; Se oprimió " S "
        ANDA    #$04
        BEQ     ENVIA       ; Se    "    " N "

APAR6:JSR       L2          ; Mensaje " PROCESADOR "
        JSR     G3
        JSR     L3          ; Mensaje " SE DARA DE ALTA "
        JSR     ESCAPE
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     MAR3        ; Se oprimió " S "
        ANDA    #$04
        BEQ     ENVIA       ; Se    "    " N "

ENVIA: LDS      #$01FF      ; Actualiza la direcciòn del SP
        JSR     CLCD
        JMP     INTUSU

MAR1:  JSR      CAFE
        LDS     #$01FF
        JSR     CLCD
        JMP     CAR

MAR2:  JSR      ANAR
        LDS     #$01FF
        JSR     CLCD
        JMP     CAR

MAR3:  JSR      GRIS
        LDS     #$01FF
        JSR     CLCD
        JMP     CAR

L1:     JSR     CLCD        ; Limpia la pantalla
        LDX     #MENS16     ; Rutina de control del mensaje 16
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS16A
```

```
        JSR     PRESEN
        RTS

L1A:    LDX     #MENS7B     ; Continuación del mensaje 16
        JSR     PRESEN
        RTS

L1B:    LDX     #MENS4B     ; Continuación del mensaje 17
        JSR     PRESEN
        RTS

L1C:    LDX     #MENS1D     ; Continuación del mensaje 18
        JSR     PRESEN
        RTS

L2:     JSR     CLCD        ; Limpia pantalla
        LDX     #MENS19     ; Rutina de control del mensaje 19
        JSR     PRESEN
        RTS

L3:     LDX     #MENS19A    ; Continuación del mensaje 19
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS19B
        JSR     PRESEN
        LDX     #PETICIONA
        JSR     PRESEN
        RTS

CAFE:   JSR     CAIM        ; Rutina pa el mensaje " VERIFI. INTERRUP. "
        LDAA    #$70
        STAA    PORTA       ; Hacemos RST1 = 0
        LDAA    REGEDO
        ORA     #$9C
        STAA    HLMC
        ANDA    #$EF
        STAA    HLMC        ; Activa mP1 y le asigna categoría principal
        STAA    REGEDO
        LDAA    #$1A
        STAA    CRACC       ; Envió ACCION 1 a mP1
        LDAA    #$78
        STAA    PORTA       ; Hacemos RST1 = 1
        JSR     INIMP       ; Rutina INIuP
        RTS

ANAR:   JSR     CAIM        ; Rutina para mensaje " VERIFI. INTERRUP. "
        LDAA    #$68
        STAA    PORTA       ; Hacemos RST2 = 0
        LDAA    REGEDO
        ORA     #$5C
        STAA    HLMC
        ANDA    #$F7
        STAA    HLMC        ; Activa mP2y le asigna categoría principal
        STAA    REGEDO
        LDAA    #$2A
        STAA    CRACC       ; Envió ACCION 2 a mP2
```

```
        LDAA    #$78
        STAA    PORTA       ; Hacemos RST2 = 1
        JSR     INIMP       ; Rutina INIuP
        RTS

GRIS:   JSR     CAIM        ;Rutina para mensaje " VERIFI. INTERRUP. "
        LDAA    #$58
        STAA    PORTA       ; Hacemos RST3 = 0
        LDAA    REGEDO
        ORA     #$3C
        STAA    HLMC
        ANDA    #$FB
        STAA    HLMC        ; Activa mP3y le asigna categoría principal
        STAA    REGEDO
        LDAA    #$3A
        STAA    CRACC       ; Envió ACCION 3 a mP3
        LDAA    #$78
        STAA    PORTA       ; Hacemos RST3 = 1
        JSR     INIMP       ; Rutina INIuP
        RTS

CAIM:   JSR     VERI        ;Mensaje " VERIFIQUE INTERRUPTORES "
        JSR     OPRI        ;Mensaje " DESEA CONTINUAR "
        JSR     ESCAPE
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     LED         ; Se oprimió " S "
        ANDA    #$04
        BEQ     DIP         ; "   "   " N "

DIP:    LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

LED:    JSR     MENSAJES
        RTS

VERI:   JSR     CLCD        ; Limpia pantalla
        LDX     #MENS22     ; Rutina de control del mensaje 22
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS22A
        JSR     PRESEN
        JSR     FLECHA
        LDX     #MENS22B
        JSR     PRESEN
        JSR     INITIMER
        RTS

MENSAJES:JSR    CLCD        ; Limpia pantalla
        LDX     #MENS23     ; Rutina de control del mensaje 23
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS23A
        JSR     PRESEN
```

```
        JSR     FLECHA
        LDX     #MENS23B
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS23C
        JSR     PRESEN
        JSR     FLECHA
        LDX     #MENS23D
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS23E
        JSR     PRESEN
        LDX     #PETICIONA
        JSR     PRESEN
        JSR     ESCAPE

ZAPA:   LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     TENI        ; Se oprimió " S "
        ANDA    #$04
        BEQ     CINT        ; "   "   " N "
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

CINT:   BRA     ZAPA

TENI:   JSR     CLCD        ; Limpia pantalla
        LDX     #MENS24     ; Rutina de cotrol del mensaje 24
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS24A
        JSR     PRESEN
        JSR     FLECHA
        LDX     #MENS24B
        JSR     PRESEN
        JSR     SL2
        LDX     #PETICIONA
        JSR     PRESEN
        JSR     ESCAPE

ZAPA1:  LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     MOCHI       ; Se oprimió " S "
        ANDA    #$04
        BEQ     ZAPA1       ; "   "   " N "
        LDS     #$01FF
        JSR     CLCD
        JMP     INTUSU

MOCHI:  LDAA    #$43
        STAA    CRACC       ; Salva contenido de memoria en disco duro
        LDAA    #$04
        STAA    REGSAL      ; Hacemos IADO = 0
```

```
        LDAA    #$00
        STAA    ARRSIN      ; Generamos señal ARRSIN
        JSR     TORR        ; Pregunta por línea " B " y hace F = 0, F = 1
        LDAA    #$08
        STAA    REGSAL      ; Hacemos DMC/DAUT# = 1
        LDAA    REGEDO
        ORA     #$1C
        STAA    HLMC        ; Fijamos a los 3 mP's como redundantes
        STAA    REGEDO
        RTS

                ********** DAR DE BAJA UNA UDFC **********


BAJAUDFC:JSR    INITIMER    ; Retraso para estabilizar teclado TF
        LDAA    HCOL
        ANDA    #$04
        BEQ     ABA         ; Se oprimió dar de baja UDFC
        JMP     RTY

ABA:    JSR     CART1       ; Mensaje " ESTA SEGURO DAR DE BAJA UDFC "
        JSR     ESCAPE

GOMA:LDAA       LAUX1
        TAB
        ANDB    #$02
        BEQ     ESTU        ; Se oprimió " S "
        TAB
        ANDB    #$04
        BEQ     LENT        ; Se    "    " N "
        BRA     GOMA

LENT:  LDS      #$01FF
        JSR     CLCD
        JMP     INTUSU

ESTU:  LDAA     #$44
        STAA    CRACC       ; Cargamos reg. de acción con ACCIÓN 4
        LDAA    #$04
        STAA    REGSAL      ; Hacemos IADO = 0
        LDAA    #$00
        STAA    ARRSIN      ; Generamos arranque sincrono
        JSR     TORR        ; Pregunta por línea " B " y hace F = 0, F =
        LDAA    #$2C
        STAA    REGSAL      ; Hace G = 1 ( CONMUTA )
        ANDA    #$DF
        STAA    REGSAL      ;   "  G = 0          "

TGB:    NOP
        BRA     TGB

CART1:JSR       CLCD        ; Limpia pantalla
        LDX     #MENS15     ; Rutina de control del mensaje 15
        JSR     PRESEN
```

```
        JSR     SL2
        LDX     #MENS15A
        JSR     PRESEN
        JSR     FLECHA
        LDX     #MENS15B
        JSR     PRESEN
        LDX     #PETICIONA
        JSR     PRESEN
        RTS
```

********** DAR DE BAJA UN PROCESADOR **********

```
BAJAPROC:JSR    INITIMER    ; Retraso para estabilizar teclado TF
        LDAA    HCOL
        ANDA    #$02
        BEQ     AJO         ; Se oprimió dar de baja procesador
        JMP     RTY

AJO:    LDAA    #$01
        STAA    BANDERAS    ; H ace BAND BAJA = 1

TREE:   LDAA    REGEDO      ; Leemos el REGEDO de RAM
        ANDA    #$80
        BEQ     P1PRES      ; Sabe que mP1 no esta activado o presente
        LDAA    #$04
        STAA    CONT1       ; Contabiliza mP's dados de baja

ENE1:   LDAA    REGEDO
        ANDA    #$40
        BEQ     P2PRES      ;Sabe que mP2 no esta activado o presente
        LDAA    #$02
        STAA    CONT2       ; Contabiliza mP's dados de baja

ENE2:   LDAA    REGEDO
        ANDA    #$20
        BEQ     P3PRES      ; Sabe que mP3 no esta activo o presente
        LDAA    #$01
        STAA    CONT3       ; Contabiliza mP's dados de baja
        JMP     LETR

P1PRES:LDAA     #$00
        STAA    CONT1       ; Localidad para contabilizar y saber que
        BRA     ENE1        ; procesadores estan presentes

P2PRES:LDAA     #$00
        STAA    CONT2       ;       "       "
        BRA     ENE2

P3PRES:LDAA     #$00
        STAA    CONT3       ;       "       "

LETR:   JSR     MSUNO       ; Despliega que procesadores estan dados de baja
```

```
PLUMA:LDAA    CONT1
      ADDA    CONT2      ; Suma contenidos de localidades
      ADDA    CONT3
      CMPA    #$07
      BEQ     SOL        ; Los 3 mP's estan activados
      CMPA    #$06
      BEQ     PON3       ; mP3 desactivado
      CMPA    #$05
      BEQ     PON2       ; mP2      "
      CMPA    #$04
      BEQ     PON23      ; mP2, mP3 desactivados
      CMPA    #$03
      BEQ     PON1       ; mP1      "
      CMPA    #$02
      BEQ     PON13      ; mP1, mP3 "
      CMPA    #$01
      BEQ     PON12      ; mP1, mP2 "
      CMPA    #$00
      BEQ     PON123     ; Ningún mP esta activado
      BRA     PLUMA

PON3: LDX     #TR3       ; Despliega " 3 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE

PON2: LDX     #DO2       ; Despliega " 2 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE

PON23:LDX     #VEI23     ; Despliega " 2,3 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE

PON1: LDX     #UN1       ; Despliega " 1 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE

PON13:LDX     #UN13      ; Despliega " 1,3 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE

PON12:LDX     #UN12      ; Despliega " 1,2 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE

PON123:LDX    #UN123     ; Despliega " 1,2,3 "
      JSR     PRESEN
      JSR     FLECHA
      JMP     SIMSE
```

```
SOL:    LDX     #SOLEA      ; Ningún procesador esta deshabilitado
        JSR     PRESEN
        JSR     FLECHA
        JMP     SIMSE

SIMSE:  JSR     CLCD        ; Limpia pantalla
        JSR     OPRI        ; Mensaje " DESEA CONTINUAR "
        JSR     ESCAPE
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     T15         ; Se oprimió " S "
        TAB
        ANDB    #$04
        BEQ     LAPIZ       ; Se oprimió " N "

LAPIZ:  JSR     CLCD        ; Limpia pantalla
        LDS     #$01FF      ; Aactualiza localidad del SP
        JMP     INTUSU

T15:    JSR     MSDOS       ;Mensaje " QUE mP SE DARA DE BAJA
        JSR     RETARDO
        JSR     OPCION3     ; [1,2,3] ? "
        LDAA    LAUX
        TAB
        ANDB    #$01
        BEQ     ENE3        ; Se eligió mP1
        TAB
        ANDB    #$02
        BEQ     ENE4        ; Se eligió  mP2
        TAB
        ANDB    #$04
        BEQ     ENE5        ; Se eligió  mP3
        BRA     T15

ENE3:   LDAA    REGEDO
        ANDA    #$80
        BNE     ENE6        ; Sabe que mP1 esta activado
        JSR     MSTRES      ;  "   "   "  NO    "
        JMP     TREE

ENE6:   LDAA    REGEDO
        ANDA    #$10
        BEQ     ENE7        ; Sabe que mP1 es principal
        JSR     MPARS
        LDAA    REGEDO
        ANDA    #$7F
        STAA    REGEDO
        STAA    HLMC        ; Actualiza reg. de estado
        JSR     FINPARS
        LDAA    #$00
        STAA    BANDERAS    ; Hace  BAND BAJA = 0
        LDS     #$01FF      ; Actualiza localidad del SP
        JSR     CLCD        ; Limpia pantalla
        JMP     INTUSU
```

```
ENE7: LDAA    #$03
      STAA    BANDERAS  ; Hace  mP1 = 1
      JMP     LRER

ENE4: LDAA    REGEDO
      ANDA    #$40
      BNE     ENE8      ; Sabe que  mP2 esta activado
      JSR     MSCUATRO  ;   "    "    "    NO    "
      JMP     TREE

ENE8: LDAA    REGEDO
      ANDA    #$08
      BEQ     ENE9      ; Sabe que  mP2 es principal
      JSR     MPARS
      LDAA    REGEDO
      ANDA    #$BF
      STAA    REGEDO
      STAA    HLMC      ; Actualiza reg. de estado
      JSR     FINPARS
      LDAA    #$00
      STAA    BANDERAS  ; Hace  BAND  BAJA = 0
      LDS     #$01FF    ; Actualiza localidad del  SP
      JSR     CLCD      ; Limpia pantalla
      JMP     INTUSU

ENE9: LDAA    #$05
      STAA    BANDERAS  ; Hace  mP2  =  1
      JMP     PRO2PR

ENE5: LDAA    REGEDO
      ANDA    #$20
      BNE     ENE10     ; Sabe que mP3 esta activado
      JSR     MSCINCO   ;   "    "    "    NO    "
      JMP     TREE

ENE10:LDAA    REGEDO
      ANDA    #$04
      BEQ     ENE11     ; Sabe que  mP3 es principal
      JSR     MPARS
      LDAA    REGEDO
      ANDA    #$DF
      STAA    REGEDO
      STAA    HLMC      ; Actualiza reg. de estado
      JSR     FINPARS
      LDAA    #$00      ; Hace  BAND  BAJA = 0
      STAA    BANDERAS
      LDS     #$01FF    ; Actualiza la localidad del  SP
      JSR     CLCD      ; Limpia pantalla
      JMP     INTUSU

ENE11:LDAA    #$09
      STAA    BANDERAS  ; Hace  mP3  =  1
      JMP     PRO3PR

MSUNO:JSR     CLCD      ; Limpia pantalla
      LDX     #MENS10   ; Rutina de control del mensaje  10
```

```
        JSR     PRESEN
        JSR     SL2             ; " PROCESADOR (ES)
        LDX     #MENS10A
        JSR     PRESEN          ;  DADO (S)  DE  BAJA  "
        JSR     FLECHA
        JSR     CLCD
        RTS

MSDOS:JSR       CLCD            ; Limpia pantalla
        LDX     #MENS11         ; Rutina de control del mensaje  11
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS11A
        JSR     PRESEN
        RTS

MSTRES:JSR      CLCD            ; Limpia pantalla
        LDX     #MENS1          ; Rutina de control del mensaje 12
        JSR     PRESEN
        LDX     #DESHA
        JSR     PRESEN
        JSR     FLECHA
        RTS

MSCUATRO:JSR    CLCD            ; Limpia pantalla
        LDX     #MENS4          ; Rutina de control del mensaje 13
        JSR     PRESEN
        LDX     #DESHA
        JSR     PRESEN
        JSR     FLECHA
        RTS

MSCINCO:JSR     CLCD            ; Limpia pantalla
        LDX     #MENS7          ; Rutina de control del mensaje  14
        JSR     PRESEN
        LDX     #DESHA
        JSR     PRESEN
        JSR     FLECHA
        RTS

OPCION3:LDAA            #$03    ; Habilitamos renglón 1  del teclado TF
        STAA    HREN
        LDAA    HCOL
        TAB
        ANDB    #$01
        BEQ     EP21            ; Se eligió mP1
        TAB
        ANDB    #$02
        BEQ     EP22            ; Se eligió mP2
        TAB
        ANDB    #$04
        BEQ     EP23            ; Se eligió mP3
        LDAA    #$06            ; Habiltamos renglón  3 del teclado TF
        STAA    HREN
        LDAA    HCOL
        ANDA    #$01
```

PAGINACION VARIA

COMPLETA LA INFORMACION

```
        BEQ     RESIS       ; Se oprimió cancelar
        BRA     OPCION3

EP21:   JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$01
        BNE     OPCION3     ; Registra un transotorio en el teclado TF
        STAA    LAUX
        JSR     G1          ; Confirma que mP2 fue elegido
        RTS

EP22:   JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$02
        BNE     OPCION3     ; Registra un transitorio en el tecladoTF
        STAA    LAUX
        JSR     G2          ; Confirma que mP2 fue elegido
        RTS

EP23:   JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$04
        BNE     OPCION3     ; Registra un transitorio en el teclado TF
        STAA    LAUX
        JSR     G3          ; Confirma que mP3 fue elegido
        RTS

RESIS:JMP       HUIR

****************** CAMBIAR LA CATEGORIA DE ALGÚN PROCESADOR ******************


CAMCAT:JSR      INITIMER    ; Retraso para estabilizar teclado TF
        LDAA    HCOL
        ANDA    #$01
        BEQ     LRER        ; Se oprimió tecla " CAMBIAR  CATEGORIA "
        JMP     RTY

LRER:   LDAA    REGEDO      ; Leemos reg.de estado redundante
        TAB
        ANDB    #$10
        BEQ     PRO1PR      ; Sabe que mP1es principal
        TAB
        ANDB    #$08        ; Sabe que mP2 es principal
        BEQ     PRO2PR
        TAB
        ANDB    #$04
        BEQ     RFV         ; Sabe que mP3 es principal
        JMP     RTY

RFV:    JMP     PRO3PR
```

```
PRO1PR:TAB
       ANDB    #$60
       CMPB    #$60
       BEQ     PRMSE     ; mP2  y  mP3 estan activos
       CMPB    #$40
       BEQ     SGMSE     ; mP2 esta activo
       CMPB    #$20
       BEQ     TCMSE     ; mP3 esta activo
       BRA     LRER

PRMSE:JSR      DESP1
       LDAB    LAUX1
       ANDB    #$04
       BNE     ALAM
       JSR     CLCD
       LDS     #$01FF
       JMP     INTUSU

ALAM:  LDAA    LAUX
       TAB
       ANDB    #$02
       BEQ     IJB       ; Sabe que se seleccionó mP2
       TAB
       ANDB    #$04
       BEQ     JIB       ; Sabe que se seleccionó mP3
       BRA     LRER

IJB:   JMP     OCT1
JIB:   JMP     OCT2

SGMSE:JSR      DESP2
       LDAA    LAUX1
       TAB
       ANDB    #$02
       BEQ     IJB       ; Sabe que oprimió " S "
       TAB
       ANDB    #$04
       BEQ     LRER      ;  "   "   "   " N "

TCMSE:JSR      DESP3
       LDAA    LAUX1
       TAB
       ANDB    #$02
       BEQ     JIB       ; Sabe que oprimió " S "
       TAB
       ANDB    #$04
       BEQ     LRER      ;  "   "   "   " N "

PRO2PR:TAB
       ANDB    #$A0
       CMPB    #$A0
       BEQ     CTOMSE    ; mP1  y  mP3 estan activados
       CMPB    #$80
       BEQ     QTOMSE    ; mP1 esta activado
       CMPB    #$20
```

```
        BEQ     STOMSE      ; mP3 esta activado
        BRA     LRER

CTOMSE:JSR      DESP4
        LDAB    LAUX1
        ANDB    #$04
        BNE     CABL
        JSR     CLCD
        LDS     #$01FF
        JMP     INTUSU

CABL:   LDAA    LAUX
        TAB
        ANDB    #$01
        BEQ     MESA        ; Se eligió mP1
        TAB
        ANDB    #$04
        BEQ     SILLA       ; Se eligió mP3
        JMP     LRER

MESA:   JMP     OCT3
SILLA:  JMP.    OCT2

QTOMSE:JSR      DESP5
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     MESA        ; Se presiono " S "
        TAB
        ANDB    #$04
        BEQ     PLM         ;Se presiono  " N "
PLM:    JMP     LRER

STOMSE:JSR      DESP6
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     OCT2        ; Se presiono " S "
        TAB
        ANDB    #$04
        BEQ     OKN         ; Se presiono  " N "
OKN:    JMP     LRER

PRO3PR:TAB
        ANDB    #$C0
        CMPB    #$C0
        BEQ     SPTOMSE     ; mP1  Y  mP2 estan activados
        CMPB    #$80
        BEQ     OCTMSE      ; mP1 esta activado
        CMPB    #$40
        BEQ     NVOMSE      ; mP2 esta activado
        JMP     LRER

SPTOMSE:JSR     DESP7
        LDAB    LAUX1
        ANDB    #$04
```

```
        BNE     HILO
        JSR     CLCD
        LDS     #$01FF
        JMP     INTUSU

HILO:   LDAA    LAUX
        TAB
        ANDB    #$01
        BEQ     OCT3        ; Se eligió mP1
        TAB
        ANDB    #$02
        BEQ     OCT1        ; Se eligió mP2
        JMP     LRER

OCTMSE:JSR      DESP8
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     OCT3        ; Se presiono " S "
        TAB
        ANDB    #$04
        BEQ     MLP         ; Se presiono " N "
MLP:    JMP     LRER

NVOMSE:JSR      DESP9
        LDAA    LAUX1
        TAB
        ANDB    #$02
        BEQ     OCT1        ; Se presiono " S "
        TAB
        ANDB    #$04
        BEQ     NKO         ; Se presiono " N "
NKO:    JMP     LRER

OCT1:   JSR     CIEN
        JSR     CLCD        ; Limpia pantalla
        LDS     #$01FF
        JMP     INTUSU      ; Regresa a INTUSU

OCT2:   JSR     AMA
        JSR     CLCD        ; Limpia pantalla
        LDS     #$01FF
        JMP     INTUSU      ; Regresa a INTUSU

OCT3:   JSR     NEG
        JSR     CLCD        ; Limpia pantalla
        LDS     #$01FF
        JMP     INTUSU      ; Regresa a INTUSU

DESP1:JSR       MENSAJE1    ; Escribe mensaje 1 en LCD
        JSR     OPCION      ; Escribe la opción elegida
        JSR     RETARDO
        JSR     CLCD        ; Limpia pantalla
        JSR     OPRI
        JSR     ESCAPE
        RTS
```

```
MENSAJE1:JSR    CLCD        ; Limpia pantalla
         LDX    #MENS1      ; Rutina de control del mensaje 1
         JSR    PRESEN
         LDX    #PRIN       ; "mP1 = PRINCIPAL
         JSR    PRESEN
         JSR    SL2         ; mP2=mP3=REDUNDANTES Y ACTIVADOS
         LDX    #MENS1A
         JSR    PRESEN      ; CUAL SERA PRINCIPAL [2,3] ? "
         JSR    FLECHA
         LDX    #MENS1B
         JSR    PRESEN
         JSR    SL2
         LDX    #MENS1C
         JSR    PRESEN
         JSR    FLECHA
         LDX    #PRIN
         JSR    PRESEN
         LDX    #MENS1D
         JSR    PRESEN
         RTS

PRESEN:LDAA     0,X         ; Rutina que contabiliza el número
         CMPA   #$2E
         BEQ    PRESI       ; de carácteres a desplegar en LCD
         STAA   MANDAA
         JSR    RETARDO
         INX
         BRA    PRESEN

PRESI: RTS

OPCION:LDAA     #$03        ; Habilitamos renglón 1 del teclado TF
         STAA   HREN
         LDAA   HCOL
         TAB
         ANDB   #$02
         BEQ    EP2         ; Se eligió mP2
         TAB
         ANDB   #$04
         BEQ    EP3         ; Se eligió mP3
         LDAA   #$06        ; Habilitamos renglón 3 del teclado TF
         STAA   HREN
         LDAA   HCOL
         ANDA   #$01
         BEQ    HR          ; Se oprimió cancelar
         BRA    OPCION

EP2:   JSR      RETARDO     ; Retardo para estabilizar teclado TF
         LDAA   HCOL
         TAB
         ANDB   #$02
         BNE    OPCION      ; Registra un transitorio en el teclado TF
         STAA   LAUX
         JSR    G2          ; Confirma que mP2 fue elegido
         RTS
```

```
EP3:    JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$04
        BNE     OPCION      ; Registra un transitorio en el teclado TF
        STAA    LAUX
        JSR     G3          ; Confirma que mP3 fue elegido
        RTS

HR:     JMP     HUIR

OPRI:   LDX     #PETICION   ; Rutina de control del mensaje
        JSR     PRESEN
        JSR     SL2         ; " DESEA CONTINUAR [S/N] ? "
        LDX     #PETICIONA
        JSR     PRESEN
        RTS

DESP2:  JSR     MENSAJE2    ; Escribe mensaje 2 en LCD
        JSR     ESCAPE
        NOP
        RTS

MENSAJE2:JSR    FUENTEA     ; Rutina de control del mensaje 2
        LDX     #UP2
        JSR     FUENTEB
        RTS

FUENTEA:JSR     CLCD        ; Limpia pantalla
        LDX     #MENS1
        JSR     PRESEN      ; " mP1 = PRINCIPAL
        LDX     #PRIN
        JSR     PRESEN      ; QUEDARA "
        JSR     SL2
        LDX     #MENS2
        JSR     PRESEN
        RTS

FUENTEB:JSR     PRESEN
        JSR     FLECHA      ; " COMO PRINICPAL [S/N] ? "
        LDX     #MENS2A
        JSR     PRESEN
        LDX     #PRIN
        JSR     PRESEN
        JSR     SL2
        LDX     #PETICIONA
        JSR     PRESEN
        JSR     RETARDO
        RTS

DESP3:  JSR     MENSAJE3    ; Escribe mensaje 3 en LCD
        JSR     ESCAPE
        NOP
        RTS
```

```
MENSAJE3:JSR    FUENTEA     ; Rutina de control del mensaje 3
        LDX     #UP3
        JSR     FUENTEB
        RTS

DESP4:JSR       MENSAJE4    ; Escribe mensaje 4 en LCD
        JSR     OPCION1     ; Pregunta por la opción elegida
        JSR     RETARDO
        JSR     CLCD        ; Limpia pantalla
        JSR     OPRI
        JSR     ESCAPE
        RTS

MENSAJE4:JSR    CLCD        ; Limpia pantalla
        LDX     #MENS4      ; Rutina de control del mensaje 4
        JSR     PRESEN
        LDX     #PRIN       ; " mP2 = PRINCIPAL
        JSR     PRESEN
        JSR     SL2         ; mP1=mP3=REDUNDANTES Y ACTIVADOS
        LDX     #MENS4A
        JSR     PRESEN      ; CUAL SERA PRINCIPAL [1,3] ? "
        JSR     FLECHA
        LDX     #MENS1B
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS1C
        JSR     PRESEN
        JSR     FLECHA
        LDX     #PRIN
        JSR     PRESEN
        LDX     #MENS4B
        JSR     PRESEN
        RTS

OPCION1:LDAA    #$03        ; Habilitamos renglón 1 del teclado TF
        STAA    HREN
        LDAA    HCOL
        TAB
        ANDB    #$01
        BEQ     EP11        ; Se eligió mP1
        TAB
        ANDB    #$04
        BEQ     EP13        ; Se eligió mP3
        LDAA    #$06        ; Habilitamos renglón 3 del teclado TF
        STAA    HREN
        LDAA    HCOL
        ANDA    #$01
        BEQ     HOJA        ; Se oprimó cancelar
        BRA     OPCION1

EP11:   JSR     RETARDO     , Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$01
        BNE     OPCION1     ; Registra un transitorio en el teclado TF
        STAA    LAUX
```

```
        JSR     G1          ; Confirma que mP1 fue elegido
        RTS

EP13:   JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$04
        BNE     OPCION1     ; Registra un transitorio en el teclado TF
        STAA    LAUX
        JSR     G3          ; Confirma que mP3 fue elegido
        RTS

HOJA:   JMP     HUIR

DESP5:  JSR     MENSAJE5    ; Escribe mensaje 5 en LCD
        JSR     ESCAPE
        NOP
        RTS

MENSAJE5:JSR    FUENTEC     ; Rutina de control del mensaje 5
        LDX     #UP1
        JSR     FUENTED
        RTS

FUENTEC:JSR     CLCD        ; Limpia pantalla
        LDX     #MENS4
        JSR     PRESEN      ; " mP2 = PRINCIPAL
        LDX     #PRIN
        JSR     PRESEN      ; QUEDARA "
        JSR     SL2
        LDX     #MENS2
        JSR     PRESEN
        RTS

FUENTED:JSR     PRESEN      ; " COMO PRINCIPAL [S/N] ? "
        JSR     FLECHA
        LDX     #MENS2A
        JSR     PRESEN
        LDX     #PRIN
        JSR     PRESEN
        JSR     SL2
        LDX     #PETICIONA
        JSR     PRESEN
        RTS

DESP6:  JSR     MENSAJE6    ; Esccribe mensaje 6 en LCD
        JSR     ESCAPE
        RTS

MENSAJE6:JSR    FUENTEC     ; Rutina de control del mensaje 6
        LDX     #UP3
        JSR     FUENTED
        RTS

DESP7:  JSR     MENSAJE7    ; Escribe mensaje 7 en LCD
        JSR     OPCION2     ; Pregunta por la opción elegida
```

```
        JSR     RETARDO
        JSR     CLCD        ; Limpia pantalla
        JSR     OPRI
        JSR     ESCAPE
        RTS

MENSAJE7:JSR    CLCD        ; Limpia pantalla
        LDX     #MENS7      ; Rutina de control del mensaje 7
        JSR     PRESEN
        LDX     #PRIN       ; "mP3 = PRINCIPAL
        JSR     PRESEN
        JSR     SL2         ; mP1=mP2=REDUNDANTES Y ACTIVADOS
        LDX     #MENS7A
        JSR     PRESEN      ; CUAL SERA PRINCIPAL [1,2] ? "
        JSR     FLECHA
        LDX     #MENS1B
        JSR     PRESEN
        JSR     SL2
        LDX     #MENS1C
        JSR     PRESEN
        JSR     FLECHA
        LDX     #PRIN
        JSR     PRESEN
        LDX     #MENS7B
        JSR     PRESEN
        RTS

OPCION2:LDAA    #$03        ; Habilitamos renglón 1 del teclado TF
        STAA    HREN
        LDAA    HCOL
        TAB
        ANDB    #$01
        BEQ     P1          ; Se eligió mP1
        TAB
        ANDB    #$02
        BEQ     P2          ; Se eligió mP2
        LDAA    #$06        ; Habilitamos renglón 3 del teclado TF
        STAA    HREN
        LDAA    HCOL
        ANDA    #$01
        BEQ     HUIR
        BRA     OPCION2

P1:     JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$01
        BNE     OPCION2     ; Registra un transitorio en el teclado TF
        STAA    LAUX
        JSR     G1          ; Confirma que mP1 fue elegido
        RTS

P2:     JSR     RETARDO     ; Retardo para estabilizar teclado TF
        LDAA    HCOL
        TAB
        ANDB    #$02
```

```
        BNE     OPCION2     ; Registra un transitorio en el teclado TF
        STAA    LAUX
        JSR     G2          ; Confirma que mP2 fue elegido
        RTS

HUIR:   JSR     CLCD        ; Limpia pantalla
        LDS     #$01FF
        JMP     INTUSU

DESP8:  JSR     MENSAJE8    ; Escribe mensaje 8 en LCD
        JSR     ESCAPE
        RTS

MENSAJE8:JSR    FUENTEE     ; Rutina de control del mensaje 8
        LDX     #UP1
        JSR     FUENTEF
        RTS

FUENTEE:JSR     CLCD        ; Limpia pantalla
        LDX     #MENS7
        JSR     PRESEN      ; "mP3 = PRINCIPAL
        LDX     #PRIN
        JSR     PRESEN      ; QUEDARA "
        JSR     SL2
        LDX     #MENS2
        JSR     PRESEN
        RTS

FUENTEF:JSR     PRESEN      ; " COMO PRINCIPAL [S/N] ? "
        JSR     FLECHA
        LDX     #MENS2A
        JSR     PRESEN
        LDX     #PRIN
        JSR     PRESEN
        JSR     SL2
        LDX     #PETICIONA
        JSR     PRESEN
        JSR     RETARDO
        RTS

DESP9:  JSR     MENSAJE9    ; Escribe mensaje 9 en LCD
        JSR     ESCAPE
        RTS

MENSAJE9:JSR    FUENTEE     ; Rutina de control del mensaje 9
        LDX     #UP2
        JSR     FUENTEF
        RTS

CIEN:   JSR     MPARS       ; Llama a la subrutina mpars
        JSR     CRGET1

AZUL:   LDAA    BANDERAS    ; Actualiza la localidad Banderas de RAM
        ANDA    #$01
        BNE     NOV9
        JSR     FINPARS
```

```
        RTS

NOV9:   LDAA    BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$02
        BEQ     NOV8
        JSR     CRGET2
        JSR     FINPARS
        RTS

NOV8:   LDAA    BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$08
        BEQ     AZUL
        JSR     CRGET3
        JSR     FINPARS
        RTS

CRGET1:LDAA     REGEDO
        ORA     #$1C
        STAA    HLMC
        ANDA    #$F7
        STAA    REGEDO
        STAA    HLMC
        RTS

CRGET2:LDAA     REGEDO
        ANDA    #$7F
        STAA    HLMC
        STAA    REGEDO
        LDAA    #$70
        STAA    PORTA
        LDAA    #$00
        STAA    BANDERAS    Actualiza la localidad Banderas de RAM
        RTS

CRGET3:LDAA     REGEDO
        ANDA    #$DF
        STAA    HLMC
        STAA    REGEDO
        LDAA    #$40
        STAA    PORTA
        LDAA    #$00
        STAA    BANDERAS    Actualiza la localidad Banderas de RAM
        RTS

AMA:    JSR     MPARS
        JSR     CRGET4

AZUL1:LDAA      BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$01
        BNE     NOV7
        JSR     FINPARS
        RTS

NOV7:   LDAA    BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$02
        BEQ     NOV6
```

```
        JSR     CRGET2
        JSR     FINPARS
        RTS

NOV6:   LDAA    BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$04
        BEQ     AZUL1
        JSR     CRGET5
        JSR     FINPARS
        RTS

CRGET4:LDAA     REGEDO
        ORA     #$1C
        STAA    HLMC
        ANDA    #$FB
        STAA    HLMC
        STAA    REGEDO
        RTS

CRGET5:LDAA     REGEDO
        ANDA    #$BF
        STAA    HLMC
        STAA    REGEDO
        LDAA    #$60
        STAA    PORTA
        LDAA    #$00
        STAA    BANDERAS    Actualiza la localidad Banderas de RAM
        RTS

CRGET6:LDAA     REGEDO
        ORA     #$1C
        STAA    HLMC
        ANDA    #$EF
        STAA    HLMC
        STAA    REGEDO
        RTS

NEG:    JSR     MPARS
        JSR     CRGET6

AZUL2: LDAA     BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$01
        BNE     SEPT1
        JSR     FINPARS
        RTS

SEPT1:LDAA      BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$04
        BEQ     SEPT2
        JSR     CRGET5
        JSR     FINPARS
        RTS

SEPT2:LDAA      BANDERAS    Actualiza la localidad Banderas de RAM
        ANDA    #$08
        BEQ     AZUL2
```

```
        JSR     CRGET3
        JSR     FINPARS
        RTS

MPARS:LDAA      #$07
        STAA    REGSAL

RGSA: LDAA      REGENT
        ANDA    #$04
        BEQ     RGSA
        LDAA    #$0F
        STAA    REGSAL
        RTS


                ********** SUBRUTINAS DE USO GENERAL **********

FINPARS:LDAA    #$06            ; Hacemos DuC/DAUT = PARS = 0
        STAA    REGSAL
        RTS

TAREA:LDAA      REGENT          ; Preguntamos por " ACTRE "
        ANDA    #$80
        BEQ     SAL             ; ACTRE = 0
        LDAA    #$45            ; ACTRE = 1
        STAA    CRACC           ; Lectura y actualización del reg. de estado de UP's
        LDAA    #$00
        STAA    ARRSIN          ; Generamos INT15
SAL:    RTS

PANT: LDAA      #$A1            ; Código ASCII para "."
        STAA    MANDAA
        JSR     INITIMER
        JSR     INITIMER
        JSR     INITIMER
        JSR     INITIMER
        JSR     INITIMER
        JSR     INITIMER
        RTS

INITIMER:LDAB   #$02            ; Contador de la rutina de retardo
        PSHB
        LDAA    #$03            ; Seleccionamos el factor del preescalador
        STAA    TMSK2

TIMER:LDAA      #$80            ; Limpiamos el bit OC1T del TFLG 1
        STAA    TFLG1
        LDD     TCNT            ; Cargamos el valor del contador delTIMER
        ADDD    #62500          ; Fijamos el retardo de 500 ms.
        STD     TOC1

LAZO: LDAB      TFLG1           ; Pregunta por la bandera " OCF1 "
        ANDB    #$80
        CMPB    #$80
        BNE     LAZO
```

```
        PULB
        DECB
        PSHB
        BNE         TIMER
        PULB
        RTS

TORR:   LDAA        REGENT      ; Preguntamos por línea " B " (SX)
        ANDA        #$20
        CMPA        #$00
        BEQ         TORR
        LDAA        #$08        ; Hace F = 0
        STAA        REGSAL
        JSR         RETARDO
        LDAA        #$0C        ; Hace  F = 1
        STAA        REGSAL
        RTS

INIPERP:LDAA        #$0C        ; Enviamos al REG.SAL el dato ( 00011100 )
        STAA        REGSAL
        RTS

VCCIN:  NOP
ARRIBA:LDAA         REGENT      ; Pregunta por  3SVVCIN
        ANDA        #$40
        CMPA        #$00
        BNE         ARRIBA
        RTS

INIPERR:LDAA        #$1C        ; Enviamos al REG.SAL el dato ( 00001100 )
        STAA        REGSAL
        RTS

MS25:   LDX         #MENS25     ; Rutina de control del mensaje  25
        JSR         PRESEN
        JSR         SL2
        LDX         #MENS25A
        JSR         PRESEN
        JSR         FLECHA
        LDX         #MENS25B
        JSR         PRESEN
        JSR         SL2
        LDX         #MENS25C
        JSR         PRESEN
        JSR         FLECHA
        LDX         #MENS25D
        JSR         PRESEN
        JSR         SL2
        LDX         #MENS25E
        JSR         PRESEN
        JSR         FLECHA
        LDX         #MENS25F
        JSR         PRESEN
        RTS
```

```
        ********** RUTINA PARA INICIALIZAR EL MÓDULO LCD **********


INILCD:LDAA     #$30        ;Habilitamos el módulo LCD para bus de datos de 8 bits
       STAA     HDISPLY
       JSR      RETARDO
       LDAA     #$30        ; Habilitamos el módulo LCD para bus de datos de 8 bits
       STAA     HDISPLY
       JSR      RETARDO
       LDAA     #$30        ; Habilitamos el módulo LCD para bus de datos de 8 bits
       STAA     HDISPLY
       JSR      RETARDO
       LDAA     #$38        ; Habilitamos el módulo LCD para bus de datos de 8 bits
       STAA     HDISPLY     ;,2 lineas y 5x10 tamaño de carácter
       JSR      RETARDO
       LDAA     #$0F        ;Ponemos el LCD  en  ON, cursor en OFF
       STAA     HDISPLY
       JSR      RETARDO
       LDAA     #$01        ; Limpia el LCD y regresa el cursor a la posición
       STAA     HDISPLY     ;inicial de la pantalla
       JSR      RETARDO
       LDAA     #$06        ;Establece el movimiento del cursor hacia la derecha
       STAA     HDISPLY
       JSR      RETARDO
       LDAA     #$0E        ;Encendemos  LCD
       STAA     HDISPLY
       JSR      RETARDO
       LDAA     #$80        ;Colocamos cursor en primera línea, primer carácter
       STAA     HDISPLY
       JSR      RETARDO
       RTS


        ********** RUTINA DE RETARDO PARA MANDAR COMANDO **********
        ********** DE CONTROL O CARACTER ASCII AL MÓDULO LCD **********


RETARDO:LDD     #$FFFF
        NOP
UNO:    DECA
        NOP
        BNE     UNO
        NOP
DOS:    DECB
        BNE     DOS
        NOP
TRES:   INCA
        BNE   TRES
        NOP
CUATRO:INCB
        BNE     CUATRO
        NOP
        LDD     #$FFFF
        NOP
CINCO:DECA
```

```
        NOP
        BNE         CINCO
        NOP
SEIS:   DECB
        BNE         SEIS
        NOP
SIETE:  INCA
        BNE         SIETE
        NOP
OCHO:   INCB
        BNE         OCHO
        NOP
        LDD         #$FFFF
        NOP
NUEVE:  DECA
        NOP
        BNE         NUEVE
        NOP
DIEZ:   DECB
        NOP
        BNE         DIEZ
        NOP
ONCE:   INCA
        NOP
        BNE         ONCE
        NOP
DOCE:   INCB
        BNE         DOCE
        NOP
        RTS

********** RUTINA PARA SALIR DE ALGUNA OPCIÓN Y RETORNAR AL **********
**********  DESPLIEGUE DE PUNTOS EN EL MÓDULO LCD  **********

ESCAPE: LDAA    #$06        ; Habilitamos renglón 3  del teclado  TF
        STAA    HREN

PRO:    LDAA    HCOL        ; Leemos teclado TF
        TAB
        ANDB    #$02
        BEQ     PRO1        ; Se oprimió " S "
        TAB
        ANDB    #$04
        BEQ     PRO2        ; Se oprimió " N "
        ANDA    #$01
        BEQ     PRO3        ; Se oprimió cancelar
        BRA     PRO

PRO1:   JSR     RETARDO     ; Retardo para estabilizar teclado
        LDAA    HCOL
        TAB
        ANDB    #$02
        BNE     PRO         ; Registra un transitorio
        STAA    LAUX1
```

```
        JSR     S               ; Escribe " S " en LCD
        RTS

PRO2:   JSR     RETARDO         ; Retardo para estabilizar teclado
        LDAA    HCOL
        TAB
        ANDB    #$04
        BNE     PRO             ; Registra un transitorio
        STAA    LAUX1
        JSR     N               ; Escribe " N " en LCD
        RTS

PRO3:   JSR     CLCD            ; Limpia pantalla
        LDS     #$01FF
        JMP     INTUSU

FLECHA:LDAA     #$05            ; Habilitamos renglón 2
        STAA    HREN

IJN:    LDAA    HCOL            ; Leemos teclado TF
        TAB
        ANDB    #$07
        CMPB    #$06
        BEQ     CAPA            ; Se oprimió tecla " > " para continuar despliegue.
        CMPB    $$07
        BEQ     IJN             ; No se oprimio tecla

CAPA:   JSR     INITIMER        ; Retardo para estabilizar teclado
        LDAA    HCOL
        TAB
        ANDB    #$07
        CMPB    #$06
        BEQ     MDS             ; Se asegura que se oprimió " > "
        BRA     IJN

MDS:    JSR     CLCD            ; Limpia pantalla
        RTS

G1:     LDAA    #$31            ; Despliega " 1 "
        STAA    MANDAA
        JSR     RETARDO
        RTS

G2:     LDAA    #$32            ; Despliega " 2 "
        STAA    MANDAA
        JSR     RETARDO
        RTS

G3:     LDAA    #$33            ; Despliega " 3 "
        STAA    MANDAA
        JSR     RETARDO
        RTS

SL2:    LDAA    #$C0            ; Salta a la línea 2 del módulo LCD
        STAA    HDISPLY
        JSR     RETARDO
```

```
        RTS

CLCD:   LDAA    #$01        ; Limpia  pantalla
        STAA    HDISPLY
        JSR     RETARDO
        RTS

S:      LDAA    #$53        ; Despliega " S "
        STAA    MANDAA
        JSR     RETARDO
        RTS

N:      LDAA    #$4E        ; Despliega " N "
        STAA    MANDAA
        JSR     RETARDO
        RTS

INIMP:  JSR     TORR        ; Pregunta por linea " B "y hace F = 0 , F =1
        LDAA    #$52
        STAA    CRACC       ; Envió ACCION 5 a mP2
        LDAA    #$00
        STAA    ARRSIN      ; Genera la señal  ARRSIN
        JSR     TORR        ; Pregunta por linea " B "y hace F = 0 , F =1
        RTS

        ************** RUTINAS PARA DESPLIEGUE DE MENSAJES **************


MENS1       FCC     ,mP1 = .,
MENS1A      FCC     ,mP2=mP3=REDUND >.,
MENS1B      FCC     ,ANTES Y ACTIVADO.,
MENS1C      FCC     ,S CUAL SERA  >.,
MENS1D      FCC     ;[2,3] ?.;

MENS2       FCC     ,QUEDARA.,
MENS2A      FCC     ,COMO .,

MENS4       FCC     ,mP2 = .,
MENS4A      FCC     ,mP1=mP3=REDUND >.,
MENS4B      FCC     ;[1,3] ?.;

MENS5       FCC     ,mP2 .,

MENS7       FCC     ,mP3 = .,
MENS7A      FCC     ,mP1=mP2=REDUND >.,
MENS7B      FCC     ;[1,2] ?.;

MENS10      FCC     ,PROCESADOR (ES).,
MENS10A     FCC     ,DADO(S) DE BAJA>.,

MENS11      FCC     ,QUE mP SE DARA .,
MENS11A     FCC     ;DE BAJA [1,2,3]>.;

MENS15      FCC     ,ESTA SEGURO EN .,
```

```
MENS15A  FCC  ,DAR DE BAJA >.,
MENS15B  FCC  ,UDFC.,

MENS16   FCC  ,CUAL mP SE DARA.,
MENS16A  FCC  ,DE ALTA .,

MENS19   FCC  ,mP[.,
MENS19A  FCC  ,] SE DARA DE .,
MENS19B  FCC  ,ALTA .,

MENS21   FCC  ,COLOQUE LA NUEVA.,
MENS21A  FCC  ,TARJETA E INDI >.,
MENS21B  FCC  ,QUE CUANDO HAYA .,
MENS21C  FCC  ,TERMINADO.,

MENS22   FCC  ,VERIFICA QUE LOS.,
MENS22A  FCC  ,INTERRUPTORES >.,
MENS22B  FCC  ,ESTEN EN MODO TF.,

MENS23   FCC  ,INSERTA LA TAR -.,
MENS23A  FCC  ,JETA EN CUALQU >.,
MENS23B  FCC  ,IER RANURA DE -.,
MENS23C  FCC  ,PROCESADOR >.,
MENS23D  FCC  ,INSERCION CON -.,
MENS23E  FCC  ,CLUIDA .,

MENS24   FCC  ,QUEDO COMPLETA -.,
MENS24A  FCC  ,MENTE INSTALA >.,
MENS24B  FCC  ,DA LA TARJETA .,

MENS25   FCC  ,LAS 2 UDFCs ES -.,
MENS25A  FCC  ,TAN DECLARADAS >.,
MENS25B  FCC  ,COMO PRINCIPALES.,
MENS25C  FCC  ,APAGAR SISTEMA >.,
MENS25D  FCC  ,Y VERIFICAR IN -.,
MENS25E  FCC  ,TERRUPTORES DE >.,
MENS25F  FCC  ,CATEGORIA .,

MENSA30  FCC  , FALLO UDFC.,

PETICION FCC  ,DESEA CONTINUAR.,
PETICIONA FCC ,[S/N] ?.,

TR3      FCC  ,3  >.,
DO2      FCC  ,2  >.,
VEI23    FCC  ;2,3 >.;
UN1      FCC  ,1  >.,
UN13     FCC  ;1,3 >.;
UN12     FCC  ;1,2 >.;
UN123    FCC  ;1,2,3 >.;
SOLEA    FCC  ,NINGUNO >.,

PRIN     FCC  ,PRINCIPAL.,
DESHA    FCC  ,APAGADO >.,

UP1      FCC  ,mP1 >.,
```

```
UP2         FCC         ,mP2 >.,
UP3         FCC         ,mP3 >.,


********** RUTINA DE SERVICIO DE INTERRUPCIÓN **********

            ORG         $FFF1       ; Dirección de inicio de la rutina de servicio de interr.
            JMP         $FB0F

            ORG         $FB0F

            SEI                     ; Deshabilitamos interrupciones

            LDAA        BANDERAS
            ANDA        #$00        ; Hacemos BAND R/P# = 0
            STAA        BANDERAS
            LDAA        #$08
            STAA        REGSAL      ; Hacemos DMC/DAUT# = 1
            LDAA        CRER
            NOP
            STAA        HLMC        ; Transferimos contenido del reg. edo. redun. a reg. edo
            LDAA        #$78
            STAA        PORTA       ; Hacemos RST1 = RST2 = RT3 = 1
            LDAA        #$08
            STAA        REGSAL      ; Hacemos F = 0
            JSR         INITIMER
            LDAA        #$0C
            STAA        REGSAL      ; Hacemos F = 1
            LDAA        #$10
            STAA        BANDERAS    ; Hacemos bandera FALLA UDFC = 1
            LDAA        #$8C
            STAA        REGSAL      ; Hacemos QVcc/PVcc = 1
            NOP
            NOP
            NOP
            LDAA        #$0C
            STAA        REGSAL      ; Hacemos HUAR/P# = 0
            CLI                     ; Habilitamos las interrupciones
            RTI
```

ARQUITECTURAS DE SISTEMAS TOLERANTES A FALLAS

En este apéndice se muestran algunas de las arquitecturas que se emplean con mayor frecuencia en los sistemas tolerantes a fallas y que ampliarán un poco la técnica de redundancia que se utilizó en el diseño de la CTF.

B.1 Redundancia pasiva por hardware

Esta técnica se basa en mecanismos de enmascaramiento de fallas. La mayoría de los enfoques pasivos se desarrollan bajo el concepto de voteo mayoritario, los criterios pasivos efectúan la tolerancia a fallas sin requerir la ejecución de rutinas de detección de fallas, ni la reconfiguración del sistema, es decir, el diseño pasivo tolera fallas de manera inherente. Dentro de esta categoría se tienen dos arquitecturas, la redundancia modular triple y la redundancia modular N, las que serán descritas a continuación.

B.1.1 Redundancia modular triple

La forma más común de redundancia pasiva por hardware es la llamada redundancia modular triple (RMT). El concepto básico es triplicar los módulos electrónicos (procesadores, memorias, etc.) para efectuar un voteo mayoritario que determine la salida del sistema, ver figura B.1, en caso de que algún módulo presente alguna falla, ella será enmascarada por los dos módulos restantes en buen estado.

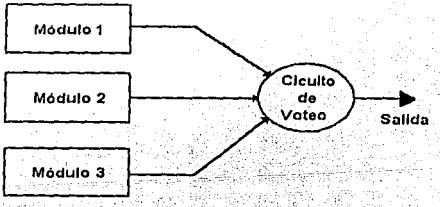


Figura B.1 Esquema de Redundancia Modular Triple.

El punto débil de esta arquitectura es el circuito de voteo, pues de presentarse una falla en él, el sistema completo se avería. Esto quiere decir que, la confiabilidad del sistema depende de la calidad del circuito de voteo. Cualquier componente dentro de un sistema que conduzca al sistema entero a un estado de avería, se le conoce como " punto único de avería ". Se pueden usar diferentes técnicas para corregir los efectos por la falla del circuito de voteo, una de ellas es triplicar el circuito de voteo y proporcionar tres salidas diferentes en vez de una, como se muestra en la figura B.2.

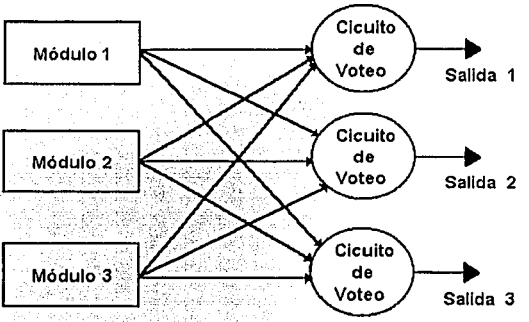


Figura B.2 Esquema RMT con Circuitos de Voteo triplicados.

B.1.2 Redundancia modular N

Se trata de la generalización del criterio RMT que se basa en el mismo principio de voteo mayoritario, pero aquí se tienen N módulos idénticos (se recomienda que N sea un número impar y mayor que 3). El concepto de redundancia modular N (RMN) se muestra en la figura B.3.

La ventaja de usar N módulos radica en el hecho de que es factible tolerar módulos fallados. Por ejemplo, si N = 5 , el esquema RMN es capaz de tolerar hasta dos módulos con fallas. Esto es importante remarcarlo, por que existen muchas aplicaciones críticas donde se requiere tolerar dos fallas a la vez, pues sus índices de confiabilidad así lo exigen.

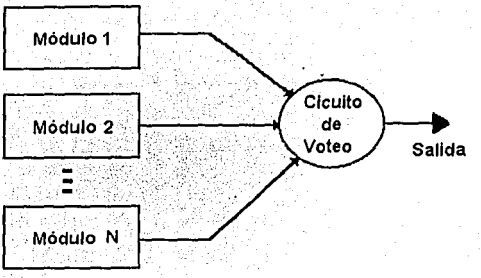


Figura B.3 Esquema de Redundancia Modular N.

B.2 Redundancia activa por hardware

Las técnicas de redundancia activa logran el atributo de tolerancia a fallas por medio de la detección, localización y recuperación de fallas. Debido a que las fallas en los diseños se detectan basándose en los errores producidos, es a veces más adecuado el uso de la terminología : detección, localización y restablecimiento de errores. Este esquema no es capaz de prevenir la presencia de errores que conducirán a averías dentro del sistema; pues no tiene la propiedad de enmascarar errores, en consecuencia las redundancias activas se utilizan en aplicaciones que temporalmente pueden tolerar resultados erróneos, mientras el sistema se reconfigura y recupera su estado operativo en periodos de tiempo razonables. Los sistemas de satélites son un buen ejemplo de aplicación de este tipo de redundancia activa.

En cuanto a las arquitecturas dinámicas los esquemas más conocidos son: la duplicación con comparación, el repuesto de apoyo, la técnica de par y repuesto y los vigías de tiempo, que serán explicadas en los siguientes párrafos.

B.2.1 Duplicación con comparación

El concepto básico de la duplicación con comparación se refiere a contar con dos módulos idénticos, ejecutando en paralelo los mismos cálculos y comparando los resultados de dichas operaciones. En caso de tener discrepancias, se genera un mensaje de error, este esquema sólo es capaz de detectar la presencia de fallas, pero no las tolera, por que no cuenta con un método que determine cual de los dos módulos tiene falla. Sin embargo, este planteamiento es de gran utilidad

como técnica de detección de fallas en los sistema basados en redundancias activas. El esquema de esta técnica se muestra en la figura B.4.

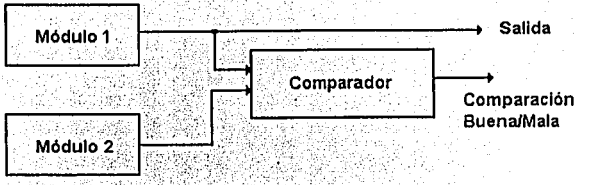


Figura B.4 Esquema de la técnica de Duplicación con Comparación.

B.2.2 Repuesto de apoyo

Una segunda forma de redundancia activa es la de repuesto de apoyo que se ilustra en la figura B.5, esta técnica contempla un módulo en operación y uno o más módulos que sirvan como repuestos o refacciones. El funcionamiento de este procedimiento requiere el uso de métodos de detección y localización de fallas (para conocer el modulo que ha fallado) con la finalidad de removerlo del sistema y reemplazarlo por uno de los módulos de repuesto.

La operación de reconfiguración se puede ver conceptualmente como un conmutador cuya salida se toma de uno, y solo uno, de los módulos que conforman el sistema. El conmutador examina los reportes de los circuitos de detección de fallas asociados a cada módulo para decidir cual salida de los módulos se debe emplear

B.2.3 Técnica de par y repuesto

La combinación de las dos técnicas mencionadas anteriormente, la de duplicación con comparación y la de repuesto de apoyo, son los elementos constitutivos del planteamiento mostrado en la figura B.6. En esencia, se tienen dos módulos trabajando todo el tiempo y sus resultados se comparan para lograr la detección de fallas requerida en la técnica de repuesto de apoyo. El circuito de comparación genera la señal de error que inicia el proceso de reconfiguración para remover los módulos con falla y reemplazarlos con los repuestos existentes. La reconfiguración se puede ver simplemente como un conmutador que acepta las salidas de todos los módulos, instalados en el sistema junto con los reportes de errores y proporciona al comparador las salidas de los dos módulos activos en

esos momentos, una de estas salidas es la salida del sistema. Mientras las dos salidas elegidas sean iguales, no se usan las refacciones. Pero cuando hay diferencias, el conmutador verifica el reporte de errores para identificar al módulo que falló y después selecciona un módulo suplente.

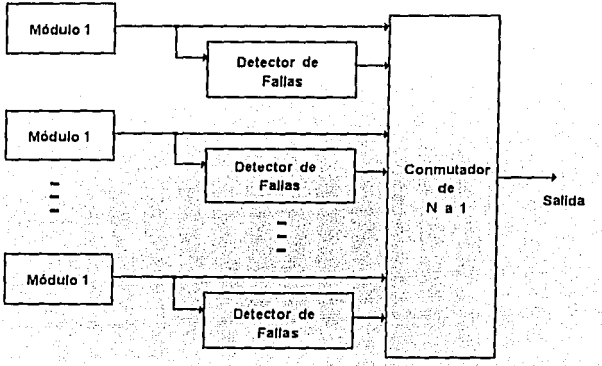


Figura B.5 Técnica de Repuesto de Apoyo.

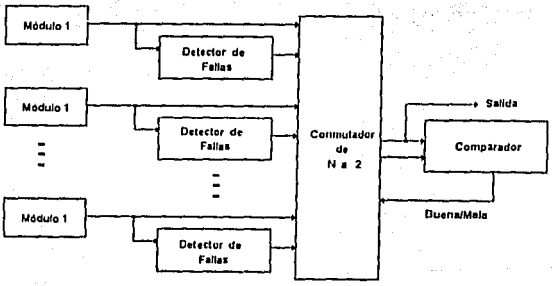


Figura B.6 Esquema de la Técnica de Par y Repuesto.

B.2.4 Vigías de tiempo

Una técnica ampliamente usada para detectar fallas en un sistema, es precisamente la de los vigías de tiempo. Estos vigías se consideran en la categoría de las redundancias activas debido a que se solicita la participación del sistema para determinar estados sin falla, la función de un guardián de tiempo es considerar la duración de una acción como un indicador de falla. Esto implica que el vigía debe ser inicializado repetitivamente, pues en caso de que el sistema dejara de hacerlo, el circuito vigía indica la presencia de una falla.

B.2.5 Redundancia de autodepuración

Este concepto es muy similar al del punto anterior, la diferencia radica en el uso que se le da a los módulos de repuesto, en el caso anterior, las refacciones están desactivadas hasta que ocurre una falla, en tanto que en el presente modelo los módulos de repuesto participan en el voteo.

En la figura B.7 se ilustra el concepto de la técnica de redundancia de auto depuración. El circuito de voteo recibe las salidas de los N módulos que conforman el sistema (incluyendo los de refacción), cada módulo tiene asociado un interruptor cuyo funcionamiento depende de la salida del circuito de voteo y de la salida del módulo al que está asociado. Es pertinente señalar que el circuito de voteo basa su operación en umbrales ponderados e involucra el uso de elementos analógicos. Sin embargo este esquema es extremadamente atractivo por sus cualidades de autodepuración que facilitan el mantenimiento y la reparación, puesto que no es necesaria la interrupción de la operación del equipo para el reemplazo de módulos fallados.

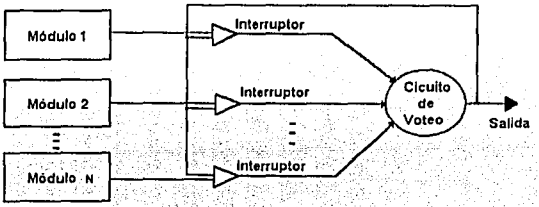


Figura B.7 Diagrama de la Redundancia de Auto depuración.

B.2.6 Arquitectura triplex- duplex

La última técnica híbrida es la denominada arquitectura triplex-duplex porque combina la redundancia modular triple y la duplicación con comparación. El esquema RMT permite enmascarar fallas y la continuidad en la operación, logrando con ello el buen desempeño de al menos un módulo sin falla. La técnica de duplicación con comparación facilita la detección de fallas y la remoción de los módulos con falla que el proceso de voteo del esquema RMT indique. Esta arquitectura se usa típicamente en aplicaciones de control donde el arreglo de suma de corriente actúe como mecanismo de voteo. El esquema básico de la, arquitectura triplex-duplex empleando técnicas de suma de corriente se muestra en la figura B.8. El proceso de suma de corriente es capaz de tolerar cualquier falla única que ocurra en el sistema, para detectar fallas cada módulo se constituye mediante la técnica de duplicación con comparación. Si durante el proceso de comparación se detecta una falla se remueve el módulo con falla del arreglo de suma de corriente. Si el esquema de duplicación con comparación detecta las posibles fallas que ocurran, la arquitectura triplex-duplex con el sumador de corriente puede tolerar dos módulos con falla.

La salida del proceso de comparación se puede usar para desconectar un módulo del sumador de corriente, como se muestra en la figura B.8. Si en la comparación se detecta una falla se abre el interruptor para remover el módulo con falla.

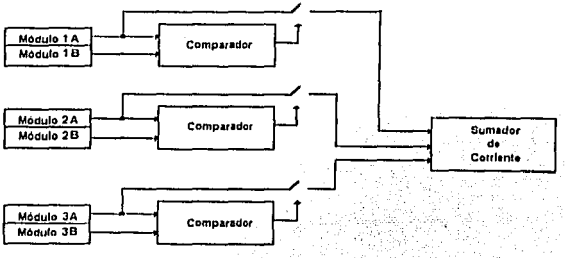


Figura B.8 Esquema de la Arquitectura Triplex-Duplex.

BIBLIOGRAFÍA Y REFERENCIAS

[1] " AT BUS DESIGN, IEE, COMPATIBLE " , Solari Edward, Anabooks, 1991.
[2] " TTL DATA BOOK " , Fairchild, 1978.
[3] " MC68HC11E9 TECHNICAL DATA " , Motorola, 1991.
[4] " FREEWARE PC-COMPATIBLE 8 BIT CROSS ASSEMBLERS USER'S MANUAL" , Motorola, March 1990.
[5] " AND DISPLAY PRODUCTS CATALOG "., AND, 1991.
[6] " PROGRAMMABLE LOGIC DEVICES DATABOOK DESIGN GUIDE " , National Semiconductor , 1990.
[7] " Diseño y construcción de las tarjetas objetivo de una computadora industrial tolerante a fallas "., Mejia Galeana J. A., Tesis de Licenciatura, Ingeniería en Computación, Facultad de Ingeniería UNAM, Enero 1994.
[8] " Desarrollo de una tarjeta electrónica para controlar dinámicamente el flujo de información en los ductos de una computadora autoreconfigurable" , Rivera Montuy R. , Tesis de Licenciatura, Ingeniería Mecánica Eléctrica, Facultad de Ingeniería UNAM, 1993.
[9] " PLD Development software, PLAN " , National Semiconductor Corporation , November 1988.
[10] " Prontuario del paquete PALASM2 " , Advance Micro Devices., 1988.

[Shladover 1993] Shladover E. S. " Research and development needs for adevenced vehicle control system ", IEEE Micro, February 1993, pp. 11-19.
[Zanoni 1993] Zanoni E. " Improving the reliability and safety of automotive electronics ", IEEE Micro, February 1993, pp. 30-48.
[Johnson D. 1984] Johnson D " The Intel 432: A VLSI archichecture for fault-tolerant computer system ", Computer, August1984, pp 40-48..
[Johnson W. 1989] Johnson, Barry W. " Design and analysis of fault-tolerant digital systems ", Addison-Wesley Inc., 1989.
[Rennels 1984] Rennels D. A. " Fault-tolerant computing Concepts and examples ", IEEE Transactions on Computers,Vol C-33, No. 12, December 1984, pp. 1116-1129.
[Vicente 1993] Vicente Vivas E., Mejia Galeana J. A., Rivera Montuy R. " Diseño de una computadora de procesamiento concurrente para aplicaciones de alto riesgo ", Memoria del II Congreso Nacional de Investigación en Ciencias Computacionales, Octubre 1993, pp. 137-148.

[Vicente 1994] Vicente Vivas E., Mejia Galeana J. A., Rivera Montuy R., Quiróz Gerardo. " Diseño y construcción del prototipo de una computadora industrial tolerante a fallas ", Informe Técnico Instituto de Ingeniería, Mayo 1994, pp. 5-11 y 165-180.

[Tein-Hsiang and Kang G. Shin] Tein-H. and Kang G. "Location of a faulty module in a computing system " , IEEE Transactions on Computers, Vol. 39, No 32, February 1990, pp. 182-193.

[Charles H. and Hsing-San] Charles H. and Hsing S. " Synergistic fault-tolerance for memory chips " , IEEE Trnsactions on Computers, Vol 41, No 9, September 1992, pp. 1078-1086.

[Ahmed] Ahmed E. Barbour " Solutions to the minimizations problem of fault-tolerantlogic circuits " , IEEE Transactions on Computers, Vol 41, No. 4 April 1992, pp. 429-442.

[Shantanu and Jhon P.] Shantanu Dutt and John P. "Some practical issues in the desing of fault-tolerant multiprocessors ", IEEE Transactions on Computers, Vol 41, No 5, May 1992, pp. 588-597.

[Shantanu and Jhon P.] Shantanu Dutt and Jhon P. " On designing and reconfiguring K-Fault-tolerant tree architectures " , IEEE Transactions on Computers, Vol 39, No 4, April 1990, pp. 490-502.

[Nitin and Dhiraj] Nitin H. Vaidya and Dhiraj K. Pradhan " Fault-tolerant desing strategies for high reliability and safety " , IEEE Transactions on Computers, Vol 42, No 10, October 1993, pp. 1195-1205.

[Irith and Sudhakar] Irith Pomeranz and Sudhakar M. " Classification of faluts in synchronous sequential circuits " , IEEE Transactions on Computers, Vol 42, No. 9, September 1993, pp. 1066-1077.