



Biblioteca de Ingeniería



FB000009618

✓  
CIA

CENTRO DE INVESTIGACION Y DE  
ESTUDIOS AVANZADOS DEL  
I. P. N.  
BIBLIOTECA  
INGENIERIA ELECTRICA

**CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS**

**DEL**

**INSTITUTO POLITECNICO NACIONAL**

**DEPARTAMENTO DE INGENIERIA ELECTRICA**

**SECCION COMPUTACION**

**"INTERFAZ PARA RED LOCAL"**

CENTRO DE INVESTIGACION Y DE  
ESTUDIOS AVANZADOS DEL  
I. F. N.  
BIBLIOTECA  
INGENIERIA ELECTRICA

Tesis que presenta el Ing. Héctor Ruiz Barradas para obtener el grado de **MAESTRO EN CIENCIAS** en la especialidad de **INGENIERIA ELECTRICA**. Trabajo dirigido por los Doctores Armando Maldonado Talamantes y Manuel Edgardo Guzmán Rentería.

Deseo expresar mis agradecimientos  
a las siguientes instituciones por  
la ayuda brindada:

Centro de Investigación y de Estudios  
Avanzados del Instituto Politécnico  
Nacional.

Universidad Autónoma Metropolitana  
Azcapotzalco.

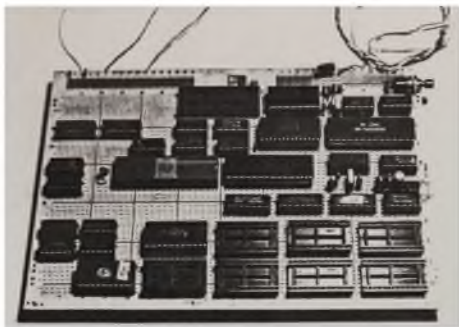
Instituto Mexicano del Petroleo.

CENTRO DE INVESTIGACION Y DE  
ESTUDIOS AVANZADOS DEL  
I. P. N.  
BIBLIOTECA  
INGENIERIA ELECTRICA

Dedico el presente trabajo a mis  
padres, mis hermanos, mi esposa y  
a mi pequeño hijo.

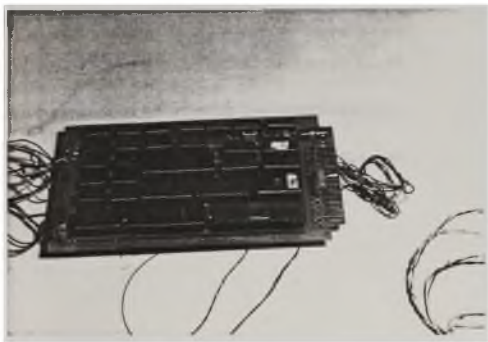
CENTRO DE INVESTIGACION Y DE  
ESTUDIOS DE LA UNAM

BIBLIOTECA  
INGENIERIA ELECTRICA



FOTOGRAFIAS DE LA INTERFAZ

BIBLIOTECA  
INGENIERIA ELECTRICA



## INDICE

RESUMEN.....	1
INTRODUCCION.....	3
CAPITULO I PRESENTACION DEL PROBLEMA.....	6
CAPITULO II DISEÑO DE LA INTERFAZ.....	16
CAPITULO III LA CIRCUITERIA.....	23
CAPITULO IV LA PROGRAMACION.....	33
CAPITULO V CONCLUSIONES.....	40
BIBLIOGRAFIA.....	43
APENDICE A FUNCIONAMIENTO DE LA RED.....	45
APENDICE B DIAGRAMA ELECTRICO DE LA INTERFAZ....	61
APENDICE C CODIFICACION DEL PROGRAMA EJEMPLO....	63



**RESUMEN**

## RESUMEN DEL TRABAJO

El presente trabajo consiste en el diseño y construcción de una Interfaz para Red Local, la cual se divide funcionalmente en dos partes principales:

- 1.- El manejo de la comunicación de información entre la interfaz y la estación asociada.
- 2.- El manejo de la transferencia de información y/o comandos dentro de la red.

La primera parte es desarrollada por un sistema digital basado en el microprocesador 8088. Dispone de 48K octetos de memoria RAM y 16K octetos de memoria EPROM. En la memoria EPROM residirá la programación necesaria para el protocolo entre interfaz y estación, inicialización del controlador de red y manejo de capa de transporte. La memoria RAM se usará principalmente para el almacenamiento de los "buffers" de recepción y transmisión de tramas. Posee un temporizador (8253), tres puertos paralelos (8255) y un puerto serie asincrónico (8250). El temporizador es utilizado como base de tiempo para la programación en tiempo real que se implante en la interfaz; los puertos paralelos son para la comunicación de la interfaz con otros dispositivos, y el puerto serie se utiliza para comunicar la interfaz con la estación asociada.

La segunda parte funcional de la interfaz está desarrollada por el circuito controlador de red (WD 2840). Este controlador tiene la técnica de acceso por estafeta circulante ("Token Passing"), y permite interconectar de 2 hasta 254 estaciones (dispositivos de usuario) en una red local con topología de línea compartida a una velocidad máxima de transferencia de 1 Megabit por segundo sobre par trenzado como medio de comunicación.

## **INTRODUCCION**

A finales de 1984, en la sección de Computación del Departamento de Ingeniería Eléctrica del Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional, se planteó la necesidad de construir una red local, con el fin de obtener experiencia en el diseño y construcción de estos sistemas, y posteriormente poder aplicar la tecnología de las redes locales a campos como la automatización de fábricas u oficinas, etc.

Dado que la construcción de la red local es un trabajo de gran magnitud, se dividió ésta en tres partes principales. La primera parte, abarca el diseño y construcción de una interfaz que permita la comunicación de varios dispositivos a través del medio físico de comunicación. La segunda parte, comprende el diseño e implantación de los protocolos necesarios para que la interfaz sea un medio confiable de intercambio de información. Por último, la tercera parte comprende la programación de algún tipo de aplicación que haga uso de las partes 1 y 2 del proyecto de la red local.

De cada una de las partes que comprende el proyecto de la red local se han originado diferentes temas de tesis de maestría. El presente trabajo, como parte de este proyecto de red local, describe el diseño y la realización de una interfaz para red local. El trabajo de esta tesis fue desarrollado en el área de Sistemas Digitales y Computadoras del Departamento de Electrónica de la Universidad Autónoma Metropolitana Azcapotzalco.

El reporte de este trabajo consta de cinco capítulos y tres apéndices. El capítulo I describe de una manera concisa y sencilla el modelo de referencia para redes de computadoras, propuesto por la Organización Internacional de Estándares (ISO), y enmarca el presente trabajo dentro de este contexto. El capítulo II da una justificación del diseño de la interfaz, y presenta de una manera general la arquitectura de la interfaz. El capítulo III describe funcionalmente los circuitos principales utilizados en el diseño de la interfaz. El capítulo IV describe los aspectos más

importantes de la programación de la interfaz, en base a un ejemplo de aplicación simple, el cual puede ser utilizado como punto de partida para aplicaciones más sofisticadas. El capítulo V está dedicado a las conclusiones del trabajo. El apéndice A, un tanto en detalle, describe el principio de funcionamiento de la red, así como características interesantes del controlador de red utilizado. El apéndice B muestra el diagrama eléctrico completo de la interfaz. El apéndice C muestra la codificación en lenguaje ensamblador del ejemplo de aplicación descrito en el capítulo 4.

**CAPITULO I**

**PRESENTACION DEL PROBLEMA**

## **GENERALIDADES**

Los conceptos de procesamiento distribuido y de redes de computadoras implican la comunicación entre diferentes entidades residentes en diversos sistemas.

Una entidad es algo capaz de enviar o recibir información (ejemplos programas de aplicación de usuario, paquetes de transferencia de archivos, etc.), y un sistema es un objeto físicamente distinto, que contiene una o más entidades (ejemplo computadoras, terminales, sensores remotos, etc.).

Para que dos entidades puedan comunicarse correctamente, ellas deben de "hablar el mismo lenguaje". Lo que se comunica, cómo se comunica y cuándo se comunica, conforma un conjunto de reglas mutuamente aceptadas entre las entidades involucradas. Así pues un protocolo consiste en un conjunto de reglas que gobiernan el intercambio de datos entre dos entidades.

## **LAS FUNCIONES DE LOS PROTOCOLOS**

Un conjunto pequeño de funciones forman la base de todos los protocolos. No todos los protocolos tienen las mismas funciones, podemos mencionar las siguientes funciones de los protocolos:

- Fragmentación y reensamble.
- Encapsulado.
- Control de Conexión.
- Control de Flujo.
- Control de Error.
- Sincronización.
- Ordenamiento.
- Direccionamiento.
- Multiplexaje.
- Servicios de Transmisión.

A los protocolos les concierne el intercambio de datos entre entidades. Los protocolos de alto nivel trabajan con unidades lógicas de datos llamadas mensajes, estos mensajes son enviados a los protocolos de bajo nivel, los cuales deben de fragmentar esos mensajes en paquetes de una longitud menor limitada; en su contraparte, esos bloques fragmentados deben reensamblarse.

Cada paquete debe llevar, además de los datos, información de control, como la dirección, tanto del destinatario como del remitente, o un código para detección de error; a la adición de la información de control a los paquetes se le conoce como encapsulado.

Las entidades pueden transmitir datos sin alguna negociación previa; a esto se le llama transferencia de datos sin conexión, (caso de datagramas). Este modo, aunque útil, es menos frecuente que la transferencia de datos orientada a conexión (caso de los circuitos virtuales).

Las entidades receptoras deben de realizar un control de flujo para limitar la cantidad de datos por unidad de tiempo que es enviada por una entidad transmisora. Por otra parte, se necesitan técnicas para proteger la pérdida o daño de datos de información o control. Existen técnicas que involucran la detección de error basados en una secuencia de inspección de la trama y retransmisión de la misma. El control de error y el control de flujo son funciones comunes que deben realizarse en protocolos de diferentes niveles.

Las entidades del protocolo necesitan recordar un cierto número de parámetros. Los parámetros se pueden ver como una variable de estado y su conjunto define el estado de la entidad. Es importante que las entidades de los dos protocolos comunicantes estén simultáneamente en un estado bien definido, por ejemplo en inicialización, verificación o terminación. A esto se le conoce como sincronización. En la transferencia de datos orientada a conexión, se requiere que el protocolo identifique el orden en el cual los paquetes que contienen datos son enviados.

Para que dos entidades se comuniquen, éstas deben de ser capaces de identificarse una a otra. Un nombre específica qué



objeto es, una dirección específica dónde está y una ruta indica cómo llegar a él.

Un protocolo puede proveer una variedad de servicios adicionales a las entidades que lo usan, tales como prioridad en el envío de mensajes, seguridad, etc.

## **EL MODELO DE REFERENCIA PARA INTERCONEXION DE SISTEMAS ABIERTOS**

Cuando se desea interconectar máquinas heterogéneas (de diferentes fabricantes, o diferentes modelos de un fabricante), el desarrollo de "software" de comunicación puede ser muy complejo. Diferentes fabricantes usan diferentes formatos y convenciones de intercambio de datos.

Debido al gran auge en la interconexión de computadoras, el desarrollo de "software" de comunicación de propósito específico es muy costoso para ser aceptable. La única alternativa para los fabricantes de computadoras es adoptar e implementar un conjunto común de convenciones. Es claro que ningún estándar sencillo será suficiente. La tarea de comunicación vista en forma cooperativa entre aplicaciones de diferentes computadoras, es muy compleja para ser manejada como unidad. El problema debe ser descompuesto en partes manejables. De modo que antes de desarrollar estándares se deberá de definir una estructura o arquitectura que precise las diversas funciones de comunicación.

Esta línea de razonamiento condujo a la Organización Internacional de Estándares (ISO : International Standard Organization), a establecer un subcomité para la definición de tal arquitectura. El resultado fue el modelo de referencia para la Interconexión de Sistemas Abiertos (OSI: Open Systems Interconnection), el cual es un punto de partida en la definición de estándares para la interconexión de computadoras heterogéneas. Las funciones de comunicación dentro de este modelo de referencia son particionadas en un conjunto vertical de capas (o niveles). Cada capa realiza un subconjunto de las funciones requeridas para comunicarse con otros sistemas. Las capas dejan que la siguiente

capa inferior provee funciones más elementales y que esas capas oculten los detalles de esas funciones. Además, cada capa provee de servicios a la siguiente capa superior. Idealmente, las capas deberían de ser definidas de manera que los cambios en una capa no requieran cambios en las otras.

A continuación describiremos brevemente el modelo de referencia:

La capa física cubre la interfaz entre los dispositivos de recepción y transmisión y las reglas mediante las cuales los bits son transmitidos. Trata con las características mecánicas, eléctricas, funcionales y de procedimiento para acceder el medio físico.

La capa de enlace provee una transferencia confiable de información a través del enlace físico, envía bloques de datos con la sincronización necesaria, control de flujo y control de error.

El servicio básico de la capa de red es proporcionar una transferencia de datos transparente entre las entidades de transporte. La capa de red libera a la capa de transporte de los detalles sobre la transmisión de datos y la tecnología de comunicación usada en la interconexión. El servicio de red es responsable de establecer, mantener y terminar conexiones (circuitos virtuales) a través de la facilidad de comunicación.

El propósito de la capa de transporte es proporcionar un mecanismo confiable para el intercambio de datos entre procesos de diferentes sistemas. La capa de transporte asegura que los mensajes sean entregados libres de error, en secuencia y sin ninguna pérdida o duplicado.

La capa de sesión provee el mecanismo para el control del diálogo entre aplicaciones. Al menos, la capa de sesión, provee un mecanismo para que dos procesos de aplicación establezcan y usen una conexión llamada sesión.

El propósito de la capa de presentación es resolver diferencias en los formatos y representación de los datos. Esta capa define la sintaxis utilizada entre las entidades de aplicación.

Finalmente, la capa de aplicación proporciona un mecanismo para que los procesos de aplicación accesen el ambiente OSI.

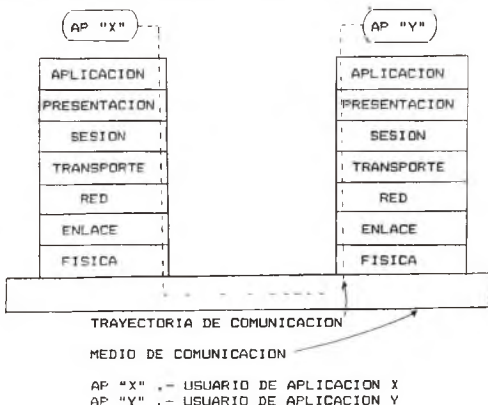


FIGURA 1.1 OPERACION DEL MODELO OSI.

En la figura 1.1 se ilustra el modelo OSI. Nótese que cada sistema contiene siete capas: para que la aplicación X e Y se comuniquen, éstas deben invocar la capa de aplicación. La capa 7 requiere servicios de la capa 6, ésta a su vez, por medio de sus entidades, interacciona con su contraparte utilizando un protocolo de su propiedad, y así sucesivamente hasta llegar a los servicios ofrecidos por la capa física.

Es necesario notar que no existe una comunicación directa entre capas equivalentes, excepto en la capa física. Todavía más, el modelo no estipula que los dos sistemas estén directamente conectados, sino que la interacción la puedan efectuar redes de: conmutación de circuitos, de conmutación de paquetes o de difusión.

Es precisamente la capa de red quien "oculta" a las capas superiores el tipo de red que proporciona el enlace de

comunicación. En las redes de conmutación de circuitos o de paquetes los datos son transferidos de fuente a destino a través de una serie de nodos intermedios. En cambio, en las redes de difusión no hay nodos intermedios, en cada estación hay un transmisor/receptor que se comunica en un medio compartido por otros sistemas; una transmisión de cualquier sistema es difundida y recibida por todos los sistemas. Sin embargo, a lo largo de este trabajo nos vamos a referir a un tipo de red de difusión conocida como Red Local. No obstante, debido a la gran variedad de los medios de transmisión y topologías de las redes locales, estrictamente no todas éstas caen en las redes de difusión, sino que algunas pueden caer en redes de conmutación de circuitos.

## **EL MODELO DE REFERENCIA IEEE 802**

Las redes locales son sistemas de comunicación que proporcionan uno o más canales digitales de alta velocidad dentro de una área restringida a algunos kilómetros, mediante el cual (o los cuales) se mantiene un uso periódico y exclusivo del canal.

Debido al gran impacto de las redes locales y la diversidad de tecnologías empleadas para su implantación, surge la necesidad de estándares en esta área. El estándar IEEE 802 sirvió como base para el diseño y construcción de nuestra interfaz para red local.

El comité para redes locales del IEEE adoptó el modelo de referencia de ISO y efectuó una descomposición adicional en las capas 1 y 2. El modelo definido se presenta en la figura 1.2.

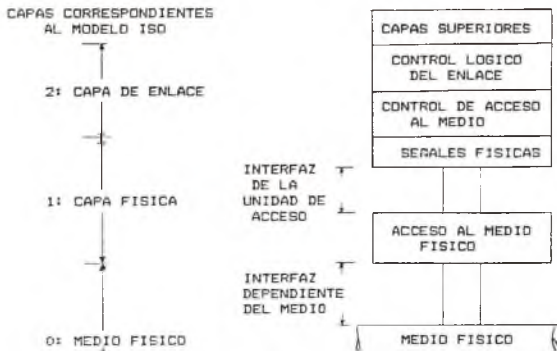


FIGURA 1.2 MODELO DE REFERENCIA IEEE 802.

Como se aprecia en dicho modelo la capa de enlace se divide en dos subcapas, la subcapa de control lógico del enlace y el control de acceso al medio. La capa física, a su vez, es dividida en la subcapa de señales físicas y la interfaz al medio físico.

La subcapa de control lógico del enlace trata con el flujo de tramas entre estaciones, incluyendo funciones de control de error y de control de flujo, tales funciones sugeridas por IEEE son muy semejantes a las proporcionadas por HDLC.

La subcapa de control de acceso al medio consiste principalmente del algoritmo para compartir la capacidad del canal físico. Esta es la función que caracteriza a las redes locales en muchas de sus propiedades fundamentales, por tanto, es usado como marco de referencia para las discusiones detalladas de redes locales.

Las técnicas para compartir el canal físico se pueden dividir en tres categorías principales:

- Métodos de Selección.
- Métodos de Contención.
- Métodos de Reservación.

En los métodos de selección a cada estación se le permite transmitir, de manera explícita, cuando ha recibido permiso de alguna entidad controladora. Si el permiso se lo confiere algún árbitro central, se dice que la red opera por interrogación ("polling"). Las redes que rotan el permiso de una estación a otra de manera descentralizada, se les conoce como redes de estafeta circulante ("token passing"). En los métodos de contención a colisión cada estación que desee transmitir debe solicitar el medio físico para sí misma, sujetándose a alguna disciplina para minimizar los efectos de interferencia entre estaciones.

En los sistemas de reservación, cada estación transmite en ranuras de tiempo que previamente fueron reservadas para ello, en forma implícita.

Todos los métodos anteriores pueden ser utilizados en sistemas de anillo, aunque existen dos técnicas que se restringen a este caso. En los anillos ranurados, un número fijo de ranuras circulan continuamente alrededor del anillo; las estaciones lo llenan y vacían según sus requerimientos. En los anillos de inserción de registros, cualquier estación puede insertar un paquete o registro entre dos paquetes adyacentes.

Existen varias maneras de codificar la información transmitida, aunque la distinción más importante se hace entre los sistemas llamados de banda base y de banda ancha. En los sistemas de banda base la señal codificada se aplica directamente al medio, en tanto que en los sistemas de banda ancha la señal debe de modular una portadora para poder ser aplicada al medio. Estas

técnicas, dentro del IEEE 802, están contempladas en la subcapa de señales físicas.

La red local puede utilizar un número variado de medios físicos: conductores metálicos (cable trenzado, cable coaxial), fibras ópticas y aún radiación electromagnética (radiofrecuencia, satélite). La naturaleza del medio conforma la estructura general del dispositivo usado para colocar señales en él. La subcapa de la interfaz al medio físico del IEEE 802 agrupa los diversos dispositivos utilizados para colocar señales en el medio físico de comunicación.

La topología de la red es otro factor importante en el diseño de la interfaz al medio físico, por ejemplo, se necesitará mayor potencia para transmitir por un "bus" de 1 kilómetro que para alcanzar 100 metros de la siguiente estación en un anillo.

Bajo el marco de referencia del IEEE 802 se plantea la necesidad del diseño y construcción de una interfaz para red local que cumpla con las capas de enlace y física del modelo de referencia para interconexión de sistemas abiertos (OSI).

## CAPITULO II

### DISENO DE LA INTERFAZ



## REQUERIMIENTOS DE LA INTERFAZ

Las redes de computadoras pueden tener un carácter privado o público. En las redes privadas los enlaces se efectúan punto a punto, ya sea en conexiones locales directas o de gran alcance ("long haul"). Los nodos, en estas redes, son utilizados para procesar información o como un conmutador para el paso de datos a través de él. La interfaz a la red en estos sistemas debe de proporcionar, al menos, un protocolo de capa 2, tal como HDLC, para la comunicación.

Para redes de gran alcance, la conexión privada sólo se justifica si hay grandes volúmenes de tráfico (utilizando líneas dedicadas) o bajo volúmenes (utilizando líneas conmutadas). Por algún rango entre esos dos extremos las redes públicas o redes de valor agregado (VAN), proporcionan el soporte de comunicación más efectivo en costo. La interfaz de red, en los sistemas conectados por redes públicas, debe de proporcionar los servicios de enrutamiento y multiplexaje, los cuales equivalen a un protocolo de capa 3, tal como X.25, para que los sistemas puedan tener acceso a una red pública.

Para redes privadas y públicas se requiere que la interfaz de red proporcione las capas 2 ó 3 respectivamente, del modelo de referencia, para poder proporcionar los servicios de comunicación.

Sin embargo, una red local de acceso múltiple no cae completamente en una u otra categoría. Los sistemas no se conectan por enlaces punto a punto, ni tampoco se puede considerar una red de valor agregado con una red de nodos intermedios, sino que se conectan por enlaces multipunto.

La interfaz para red local debe de ser capaz de asegurar la conexión de una gran variedad de sistemas, y aun, de diferentes fabricantes.

## FORMAS DE CONECTARSE A UNA RED LOCAL.

Existen dos escenarios para los cuales se puede seleccionar una interfaz para red local. Las alternativas son:

- Red homogénea.
- Red no homogénea.

La red homogénea implica una red en la que todo el equipo (red más dispositivos conectados) es proporcionado por un solo fabricante, de modo que todo el equipo comparte el mismo conjunto de programación y circuitería en su interfaz de red; ésto simplifica el trabajo para el fabricante, pero limita demasiado al usuario cuando éste quiera integrar equipos de otros fabricantes a su red.

El escenario de una red no homogénea implica la utilización de un dispositivo (el cual es la interfaz para red local) que implemente los protocolos de red y que proporcione una capacidad de interfaz para la conexión de dispositivos (sistemas en términos del capítulo anterior).

La interfaz controla el acceso y la comunicación a través de la red, y los dispositivos se conectan a la red local a través de la interfaz por alguno de sus puertos.

Los escenarios de red homogénea y no homogénea, de manera intrínseca, hacen uso de algún estándar, que normalmente es el IEEE E07 en sus diferentes alternativas (métodos de selección, contención o reservación).

Para el presente trabajo, se selecciona el escenario de red no homogénea, dado que permite la conexión de la mayoría de los dispositivos digitales (sistemas) de un usuario, quedando "ocultos" en la interfaz de red los detalles de circuitería y programación.

## **LA TOPOLOGIA.**

La topología de una red se refiere a la estructura geométrica en que las estaciones de una red son conectadas. En redes locales se tienen tres topologías fundamentales: Estrella, Anillo y Línea Compartida ("Bus"). La topología utilizada depende de una gran variedad de factores, tales como la confiabilidad, expandibilidad y rendimiento.

En este trabajo se eligió la topología de línea compartida pues parece ser la más flexible, además de su fácil reconfiguración y de su alta confiabilidad, debido al control distribuido del acceso a la red.

## **EL METODO DE ACCESO.**

El método de acceso en una red local se refiere a la forma de compartir el canal físico de comunicación.

En topologías de líneas compartidas existen dos técnicas: el Acceso Múltiple por Detección de Portadora (CSMA), la cual es una técnica por contención, y la de Estafeta Circulante ("Token Passing") referente a mecanismos de selección.

La desventaja principal del CSMA es la falta de determinismo, o en otras palabras, la posibilidad de garantizar el acceso exitoso de una estación en un intervalo de tiempo dado, debido a la alta probabilidad de ocurrencia de colisiones en el medio de transmisión.

Este hecho hace que CSMA pierda terreno en aplicaciones en tiempo real en las cuales se requiere límites de retardo absoluto y confiabilidad garantizada. El método de acceso por estafeta circulante evita este tipo de dificultad.

Nuestro interés, a futuro, en sistemas de control de procesos industriales, nos motivó a elegir el mecanismo de estafeta

circulante como el método de acceso en nuestra interfaz de red local.

## **DISEÑO DE LA INTERFAZ.**

La unidad de interfaz a la red, como se vió en la secciones anteriores, actuará como un controlador de comunicaciones para proveer servicio de transmisión de datos en una red local a uno o más dispositivos conectados a estas unidades. La unidad de interfaz transformará el protocolo y velocidad de transmisión de datos del dispositivo conectado a ésta, a aquél del medio de transmisión de la red local, y viceversa. Los datos en el medio de transmisión de la red local, son disponibles a todos los dispositivos conectados a la unidad de interfaz de red. En términos generales la unidad de interfaz de red realiza las siguientes funciones:

- a).- Aceptar datos de un dispositivo conectado.
- b).- Almacenar los datos hasta que el acceso al medio se lleva a cabo.
- c).- Transmitir los datos en paquetes direccionados.
- d).- Inspeccionar cada paquete en el medio para verificar la dirección de éste.
- e).- Si el paquete lleva la dirección adecuada leerlo, y almacenarlo en memoria.
- f).- Transmitir los datos al dispositivo conectados a la velocidad adecuada.

La arquitectura general de la interfaz se muestra en la figura 2.1.

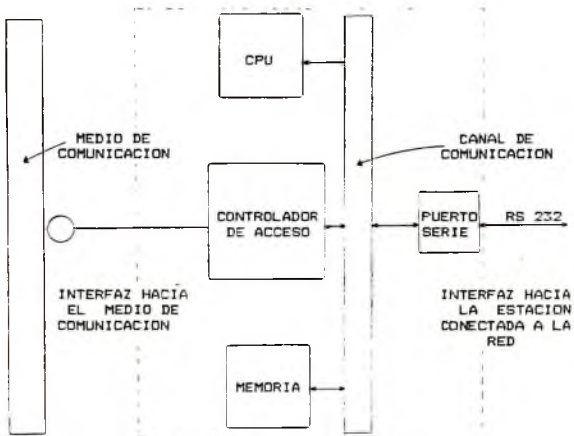


FIGURA 2.1 ARQUITECTURA GENERAL DE LA INTERFAZ PARA RED LOCAL

La CPU de la interfaz en la transmisión es responsable de efectuar los puntos a) y b) de las funciones mencionadas anteriormente, el Controlador de Acceso, durante la transmisión, realiza el punto c), y en recepción, los puntos d) y e). Finalmente la CPU realiza el punto f) cuando recibe un paquete.

Además de manejar estas funciones correspondientes a las capas 1 y 2 del modelo ISO, que son el objetivo fundamental del presente trabajo, se deja la circuitería para el manejo de las capas 3 y 4. Así pues, cualquier estación conectada a la interfaz le son transparentes las tareas de comunicación y sólo se dedica a su aplicación.

La interfaz serie RS 232 entre la Unidad de Interfaz a la Red y la Estación, permite la conexión de una gran variedad de

dispositivos a la red local, dado que casi todas las computadoras y terminales soportan la interfaz RS 232.

Con un diseño de interfaz para red local como el de la figura 2.1 se puede tener una red que permita la interconexión de microcomputadoras, minicomputadoras y dispositivos periféricos, de diferentes fabricantes, y dando la posibilidad de comunicación de la red local con otras redes, utilizando un protocolo como X.25. La figura 2.2 nos muestra estas posibilidades.

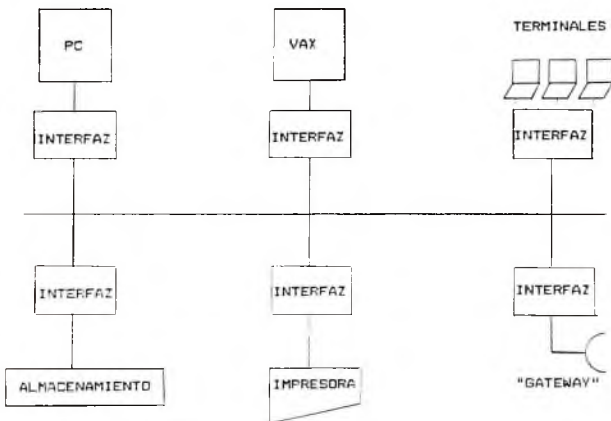


FIGURA 2.2 RED LOCAL

**CAPITULO III**

**LA CIRCUITERIA**

En el capítulo anterior se describió a bloques el diseño de la unidad interfaz para red local y se especificaron sus funciones básicas. En este capítulo se abordará con mayor detalle el diseño de la circuitería ("hardware") así como una previa descripción funcional de las partes que la componen.

La arquitectura de la interfaz se muestra en la figura 3.1.

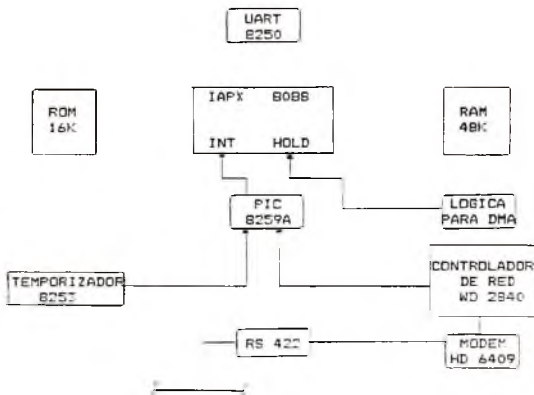


FIGURA 3.1 ARQUITECTURA GENERAL DE LA INTERFAZ PARA RED LOCAL

La interfaz para red local se divide funcionalmente en dos partes :

- 1.- El manejo de la comunicación de información entre la interfaz y la estación asociada.



2.- El manejo de la transferencia de información y/o comandos dentro de la red, de acuerdo con el protocolo de estafeta circulante.

Es importante señalar el carácter concurrente e independiente de ambos componentes de la interfaz.

La primera parte dispone de un sistema mínimo basado en el microprocesador IAPX 8088, el cual contiene 48K octetos ("bytes") de memoria RAM y 16K octetos de memoria EPROM. Disponen además de un puerto serie asíncrono (WD 8250) para la comunicación con la estación, un temporizador ("timer" 8253) como base de tiempo y un controlador de interrupciones (8259A) para el manejo de eventos asíncronos.

La segunda parte está constituida fundamentalmente por el circuito controlador de acceso a la red (WD 2840), el circuito codificador/decodificador Manchester (HD 6409) y el transmisor/receptor de línea ("transceiver" MC 3486 y MC 3487), bajo el estándar RS 422.

A continuación se dará una breve descripción funcional de los circuitos que componen el sistema digital (ver diagrama del apéndice B) :

El circuito 8284 se utiliza para proveer la señal de reloj de 5 MHz al microprocesador; a través de este circuito también se proporcionan las señales de RESET (para reinicializar el CPU) y READY (para generar ciclos de espera), propias del microprocesador. Los dos circuitos 74LS373 ("latch") y 74LS245 ("transceivers"), conectados a los ductos ("buses") de datos/direcciones, son utilizados para demultiplexar el ducto de datos y el ducto de direcciones.

El circuito 74LS175 contiene cuatro "flip flops" tipo D, de los cuales 3 se utilizan para insertar 2 ciclos de espera en los diversos ciclos de máquina del procesador.

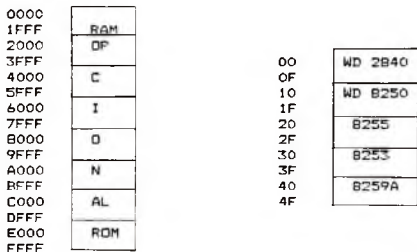
La información sobre el microprocesador 8086/88, y su familia, se cubre ampliamente en "The 8086 Family User's Manual", editado por la corporación Intel.

El microprocesador 8088 tiene capacidad para direccionar 1 Mega octeto de memoria, con 20 líneas de direcciones para este fin; la interfaz sólo utiliza 16 líneas, es decir, tiene capacidad para decodificar 64K octetos de memoria, el mapa de la decodificación se da en la figura 3.2(a).

El algoritmo para la inicialización de la interfaz, así como la programación de las capas superiores del modelo ISD residirán en la memoria EPROM (2764 de 1K X 8) del sistema; la memoria de lectura/escritura (6164 de 8K X 8) sirve para el almacenamiento de tramas, así como para las diversas operaciones que requiera la interfaz.

El mapa de memoria es implementado por medio de un circuito decodificador de 3 a 8 (74LS138).

Para el mapa de puertos se utilizó otro circuito decodificador 74LS138 (ver figura 3.2b).



a) MAPA DE MEMORIA

b) MAPA DE PUERTOS

FIGURA 3.2 MAPA DE MEMORIA Y PUERTOS.

El circuito UART 8250 es un puerto serie de recepción/transmisión asincrónica, utilizado para la comunicación entre la interfaz y la estación asociada. Este circuito permite la programación de la velocidad de transferencia. Los registros internos tienen las siguientes direcciones:

con el bit DLAB = 0

- 10h = Registro de recepción y transmisión.
- 11h = Registro de habilitación de interrupciones.
- 12h = Registro de identificación de interrupciones.
- 13h = Registro de control de línea.
- 14h = Registro de control de modem.
- 15h = Registro de estado de línea.
- 16h = No usado.

con el bit DLAB = 1

- 10h = Registro Divisor (octeto menos significativo).
- 11h = Registro Divisor (octeto más significativo).

Los niveles TTL de la entrada y salida serie del WD 8250 son adaptados al estándar RS 232 C con los circuitos MC 1488 y MC 1489.

El circuito 8259A es un circuito utilizado como controlador de interrupciones; dispone de ocho líneas de entrada y una de salida, la cual se conecta a la línea INT del procesador para petición de interrupciones. Las líneas de entrada se conectan a los diferentes circuitos que desean interrumpir; el 8259A permite el uso de prioridades, por lo que puede manejar conflictos de jerarquía en las interrupciones. Las direcciones de acceso a este circuito son 40h y 41h.

El circuito 8253 es un temporizador capaz de proporcionar una base de tiempo útil para la programación en tiempo real. Las direcciones de los registros de este circuito dentro de la interfaz son las siguientes:

- 30h = Contador 0.
- 31h = Contador 1.
- 32h = Contador 2.
- 33h = Registro de palabra de modo.

El circuito 8255 contiene tres puertos paralelos que son utilizados para la comunicación en paralelo entre la interfaz y la estación asociada. Actualmente este circuito no se encuentra alambrado en la interfaz.

El circuito WD 2840 es visto por el microprocesador como una serie de 16 registros, algunos de lectura y otros de lectura/escritura. Las direcciones dentro de la interfaz de los registros son las siguientes :

- 00h = Registro de control 0.
- 01h = Registro de control 1.
- 02h = Registro de estado 0.
- 03h = Registro de eventos de interrupción.
- 04h = Registro de estado 1.
- 05h = Registro de estado 2.
- 06h = Registro contador.
- 07h = Registro de dirección siguiente.
- 0Eh = Registro del temporizador TA.
- 09h = Registro del temporizador TD.
- 0Ah = Registro apuntador al bloque de control (octeto alto).
- 0Bh = Registro apuntador al bloque de control (octeto bajo).
- 0Ch = Registro de petición de dirección siguiente.
- 0Dh = Registro del límite máximo de recepción del "token".
- 0Eh = Registro del límite máximo de tramas de transmisión.
- 0Fh = Registro de dirección propia.

El circuito WD 2840 es el controlador de acceso a la red y es el elemento principal de la segunda parte de la interfaz. A continuación se describe ésta.

El circuito más importante de esta parte es el controlador de acceso por estafeta circulante ("token passing") denominado WD 2840; en este capítulo describiremos brevemente las características del circuito, en el apéndice A se proporciona información más detallada sobre éste.

El WD 2840 es un dispositivo de muy alta escala de integración, capaz de manejar completamente las funciones de la capa de enlace y parte de la capa física del modelo de referencia para redes locales (IEEE 802).

El WD 2840 está diseñado para conectar lógicamente de 2 hasta 254 interfaces que comparten el medio de comunicación. El protocolo implementado permite que tenga el completo control de las funciones dentro de la red, tales como el encapsulamiento de tramas, rechazo de tramas que no tengan la dirección correcta, número programable de tramas a transmitir, inicialización de la red, recuperación de fallas etc.

El WD 2840 presenta tres estados funcionales: El estado de aislamiento, el estado de prueba y el estado de red. Escribiendo las palabras de control necesarias en los registros internos del circuito, permite cambiarlo de uno a otro estado. El estado inicial del circuito corresponde al de aislamiento, antes de pasar al estado de red, el circuito debe llevarse al de prueba para verificarse a sí mismo y al sistema donde se encuentra conectado. Si las pruebas son correctas el circuito se lleva al modo de red, en el cual el circuito puede enviar o recibir tramas de la red. La figura 3.3 nos muestra el diagrama a bloques del WD 2840.

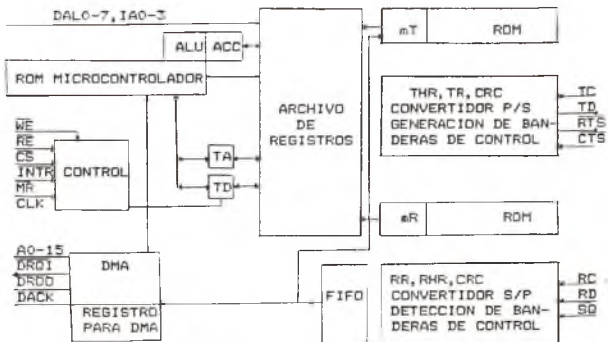


FIGURA 3.3 DIAGRAMA A BLOQUES DE EL WD 2840

El conjunto de 16 registros es el medio por el cual el circuito puede aceptar comandos o enviar información al microprocesador 8088 de la interfaz. El microcontrolador del WD 2840 ejecuta el algoritmo del control lógico del enlace y el control de acceso al medio. Los ductos para el acceso directo a memoria colocan o toman las tramas en (de) la memoria de la interfaz. Los bloques de transmisión se encargan de hacer la conversión paralelo a serie y encapsular los datos en una trama similar a las utilizadas por HDLC. El bloque de recepción verifica que la trama sea direccionada a esta interfaz; si la dirección es correcta la trama es introducida a la memoria. Una vez recibida la trama, se verifica que no haya errores, en caso contrario, la trama es abortada. En algunos casos se notifica al transmisor que la trama fue incorrectamente recibida.

El WD 2840 posee dos señales para indicar que requiere un acceso directo a memoria, por tanto, se necesita cierta lógica para indicar al procesador una petición de transferencia por DMA

y, después activar las señales de lectura o escritura en la memoria. Además de las necesidades mencionadas anteriormente, también se requiere cumplir con ciertas especificaciones de tiempo en las lecturas y escrituras al WD 2840, después de un acceso directo a memoria. Sólo se puede acceder al WD 2840 después de 500 ns de un acceso directo a memoria. Esto obligó a anexas 2 registros de corrimiento (74LS164) y un conjunto de compuertas (ver el diagrama eléctrico de la interfaz en el apéndice B ).

El WD 2840 envía y recibe los datos en serie, codificados en NRZ a una velocidad máxima de 1 Mega bit por segundo, y provee señales para control de modem. El circuito codificador - decodificador Manchester (HD 6409) codifica la señal de transmisión en código Manchester, y decodifica la señal de recepción de Manchester a NRZ. Desafortunadamente, no fue posible conseguir en el mercado este circuito, por lo tanto no se encuentra alambrado en el prototipo desarrollado. Las señales codificadas son conectadas al medio físico de comunicación (par trenzado) a través del manejador (MC 3486) y receptor (MC 3487) de línea.



## CAPITULO IV

### LA PROGRAMACION

La circuitería de la interfaz para red local se dividió en dos partes funcionales. Esta división también se aplica a la programación de la interfaz, es decir, cada parte que compone la interfaz estará ejecutando un algoritmo propio de su aplicación. Por ejemplo, el microprocesador 8088 en su operación normal, continuamente esperará comandos de la estación asociada, y colocará las tramas a transmitir en una zona de memoria compartida; a su vez el controlador WD 2840 estará ejecutando el algoritmo de acceso a la red para poder transmitir las tramas necesarias. En la recepción el controlador de acceso a la red continuamente inspeccionará las tramas direccionadas a él, y las depositará en una zona de memoria compartida. Cuando se haya terminado de recibir la trama, el microprocesador tomará las tramas almacenadas y las enviará a la estación asociada. Cada una de las partes que componen la interfaz lleva una programación diferente. El WD 2840 internamente tiene la programación del control lógico del enlace y el algoritmo para el acceso al medio de comunicación. De esta manera el diseñador de la interfaz se libera de realizar los algoritmos para esas funciones, y sólo se ocupa de hacer un algoritmo que sea capaz de inicializar el controlador de acceso, preparar las estructuras de datos necesarias para el manejo de información, ofrecer atención a la estación conectada a la interfaz y proporcionar la algorítmica para anexionar o eliminar estaciones del anillo lógico de la red.

En las páginas siguientes se describen los algoritmos anteriores.

Consideremos el caso particular de comunicación entre dos terminales, en el que una interfaz atiende una terminal.

Los algoritmos realizados en este ejemplo pueden considerarse como base para otras aplicaciones.

En el apéndice C se muestra la codificación del programa de aplicación en lenguaje ensamblador del microprocesador 8088. Para

una mejor comprensión del algoritmo se recomienda leer primero el apéndice A.

El programa empieza con la inicialización del puerto serie (WD 8250); éste se programa para recibir y transmitir datos de 7 bits a 1200 bits por segundo; posteriormente se programa el controlador de interrupciones (8259A) para dar una interrupción de tipo 9, al microprocesador 8088, en el momento que el controlador de acceso tenga algún evento que informar.

El siguiente paso consiste en programar al WD 2840 en su estado de prueba, de manera que se verifique el correcto funcionamiento de la lógica para los accesos directos a memoria. La prueba del DMA se lleva a cabo leyendo el valor de un registro del controlador y escribiendo en la memoria de usuario (por DMA), posteriormente la localidad de memoria es leída (por DMA) y se escribe en otro registro del controlador. Finalmente, ambos registros son comparados; si no hay diferencia la prueba es exitosa, en caso contrario la prueba es fallida. El algoritmo se muestra a continuación.

- 1.- Posicionar el controlador WD 2840 en el modo de diagnóstico y esperar la confirmación de éste (probando el bit STATE).
- 2.- Cargar los registros del controlador.
  - Mandar la dirección de la zona de memoria a los registros del controlador donde se va a escribir.
  - Enviar los datos (carácter "C" y "D") que se escribirán y leerán en la memoria, a los registros del controlador.
- 3.- Habilitar el bit del controlador que indica prueba de DMA (DMAT), y en seguida activar el bit que inicia la prueba (NUDIAG).
- 4.- Esperar que el bit NUDIAG se desactive indicando el fin de la prueba.

- 5.- Verificar que se hayan leído y escrito correctamente los registros del controlador y las localidades de memoria.
  - Si la prueba es positiva ir al paso 6.
  - Si la prueba es negativa reportar el error y parar el algoritmo.
- 6.- Regresar al controlador al estado de aislamiento (escribiendo 00 en el registro de control 1).

Una vez que la prueba de DMA es positiva se procede a preparar el bloque de control, el cual es un estructura de datos.

El controlador WD 2840, al iniciarse en el modo de red, busca en el bloque de control el inicio de las cadenas de "buffers" de recepción y transmisión, además del número de octetos que ocupa cada "buffer".

El siguiente paso consiste en preparar las cadenas de "buffers" de recepción y transmisión, en la zona de memoria RAM (ésto lo hacen las subrutinas FREBR y PREBT).

La figura 4.1 muestra la estructura de las cadenas de "buffers" de recepción y transmisión, como fueron preparadas por las rutinas PREBT y FREBR.

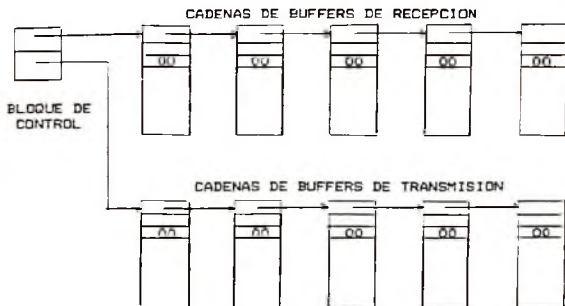


FIGURA 4.1 ESTRUCTURA DE LOS BUFFERS DE TRANSMISION Y RECEPCION

Una vez que se han preparado el bloque de control y las cadenas de "buffers", sólo queda programar los registros internos del controlador y comandarlo al estado de red.

Al programar el controlador se le debe proporcionar el valor de los temporizadores TA y TD (ver el apéndice A), la dirección (16 bits) del bloque de control, la siguiente estación en el anillo lógico de la red, el número de estafetas ("tokens") a recibir antes de transmitir las tramas, el número máximo de tramas a transmitir antes de ceder la estafeta, y la dirección única del controlador.

Una vez programados los registros internos, el controlador se lleva al estado de red (escribiendo 00 en el registro de control 0); cuando el bit STATE se posiciona a cero, el WD 2840 confirmará que está en modo de red, y podrá aceptar comandos.

A continuación se le envía un comando para posicionar a 1 los bits INIT, GRING y NEWNA. Con el bit INIT en 1 se le indica, al WD 2840, que cuando el temporizador TD expire, se inicie el proceso de recuperación de la red (ver apéndice A para mayor información)

el bit GRING, en 1, indica al WD 2840, que la siguiente vez que le pasen la estafeta, el controlador formará parte del anillo lógico. Finalmente, NEWNA en 1 indica al controlador que el registro NAR tiene la dirección del siguiente controlador a pasarle la estafeta, en su oportunidad. Es importante señalar, en este punto, que no a todos los controladores que forman el anillo lógico en la red se les programará el bit INIT en 1, pues sólo unos controladores tendrán la capacidad de recuperar la red cuando el temporizador TD expire. Referente a la inicialización de la red, no necesariamente se sabrá cuál es la dirección del siguiente controlador en el anillo lógico; si éste fuera el caso se utilizaría alguno de los métodos de inicialización descritos en el apéndice A.

El último paso en la programación consiste en habilitar la recepción y transmisión de tramas, programando el registro de control 0 con el valor E0h.

El programa principal para la aplicación particular de la comunicación de dos terminales, a través de la interfaz para red local, sólo inspeccionará la terminal para saber en que momento, el operador desea transmitir una trama. Cuando éste oprima la tecla "T", se llamará a la subrutina TRANS la cual se encargará de almacenar todos los caracteres tecleados, en la cadena de "buffers" de transmisión, hasta la llegada del carácter de regreso de carro (CR). Una vez llenado el (los) "buffer(s)" se procede a modificar los octetos altos de los apuntadores de éstos (en la inicialización, tales octetos estaban en cero), de manera que se precise el apuntador al siguiente "buffer". Al término de esta operación la rutina TRANS regresa el control al programa principal para repetir el mismo ciclo. Cuando el WD 2840 recibe la estafeta y le es permitido transmitir (si el número de estafetas recibidas antes de la transmisión es igual al número programado en la inicialización), buscará los apuntadores en sus registros internos e inspeccionará el octeto alto del apuntador; si éste es cero no hay "buffer" a transmitir, pero si es diferente localiza

el "buffer" y lo transmite. Mientras sea superior a uno se inspecciona el apuntador y el proceso se repite hasta transmitir el último "buffer".

A la recepción de una trama, el controlador activa una interrupción que hace que el microprocesador 8088 busque en la cadena de "buffers" de recepción la trama recibida y la envíe a su terminal asociada.

En el modo de red, el controlador informa de algunas condiciones de trabajo, como por ejemplo la recepción de la estafeta, la transmisión de una trama, la transmisión fallida de la trama, la recepción de trama, y otras, mediante la emisión de una señal de interrupción. Cuando el controlador interrumpe, el microprocesador debe leer el registro de identificación de interrupciones y efectuar alguna acción dependiendo de la causa por la que fue interrumpido. Ver "Communication Product Handbook", editado por Western Digital Corporation, en junio de 1984.

En este ejemplo, la rutina para la inicialización y la anexión de estaciones en el anillo lógico de la red no se ha implementado, pues la aplicación es sumamente sencilla. No obstante, en el apéndice A, se describen algunos métodos para llevar a cabo la inicialización y anexión de estaciones.

Es preciso notar que en el presente ejemplo la programación se simplificó bastante, con el fin de comentar los puntos importantes de la programación del controlador WD 2840. Si se desea implantar otra programación alrededor de la interfaz, con este ejemplo se tiene la estructura general del programa, y sólo deben realizarse las rutinas para la aplicación que se desee.

**CAPITULO V**

**CONCLUSIONES**



Normalmente, en nuestro medio, se habla del vasto campo de aplicación de las redes locales (por ejemplo la automatización de fábricas u oficinas), y el enfoque se orienta a la programación de capas de transporte o superiores (IEEE 802) de estas aplicaciones, pero poco se ha hecho en la realización de las capas inferiores. Estas son, por su orden jerárquico, las que sustentan a las demás, y deberían de realizarse primero para poder llegar a objetivos más ambiciosos en las condiciones actuales de nuestro País.

El presente trabajo pretende ayudar a subsanar la carencia de desarrollo en los niveles físico y de enlace, del modelo de referencia para redes locales, en nuestro País. Con esta experiencia, y la infraestructura creada, se puede emprender mejoras en la interfaz, o el desarrollo de las capas superiores del modelo de referencia para alguna aplicación en particular.

Es importante hacer notar que el diseño de la interfaz se centró en poder conectar en una red local a una diversidad de estaciones, tales como: microcomputadoras, minicomputadoras o dispositivos periféricos. Esto originó la necesidad de incluir el microprocesador 8088 y circuitos necesarios, para manejar un puerto serie compatible con esa diversidad de estaciones (Estándar RS 232-C). Tal cantidad de circuitos elevó el costo de la interfaz; no obstante, a nuestro juicio, la "universalidad" lograda, la hace más interesante que las interfaces de red comerciales.

El diseño actual de la interfaz permite conectar computadoras personales o dispositivos periféricos con un mínimo de esfuerzo en la programación y en la circuitería. Esto es, la interfaz está diseñada para conectar la mayoría de los niveles del modelo de referencia IEEE 802; así pues, la estación asociada queda liberada de los protocolos de comunicación, atendiendo únicamente la comunicación con la interfaz. La estación solo debe proporcionar un puerto serie asincrónico con el estándar RS 232 para comunicarse con la interfaz.

El diseño de la circuitería de la interfaz deja decodificadas algunas direcciones, no utilizadas, que pueden utilizarse para direccionar otros puertos con otras aplicaciones diferentes.

La flexibilidad de la interfaz, en la circuitería para aplicaciones futuras, y la facilidad de conexión con la estación, son características originales de la interfaz diseñada y realizada en el presente trabajo.

La justificación principal de este trabajo es la de obtener experiencia en el campo de las redes locales, principalmente en los niveles 1 y 2 del modelo de referencia IEEE 802, así como tener prototipos que sean susceptibles de comercializarse en un mercado nacional que no ofrece alternativas de productos diseñados y realizados en el País.

**BIBLIOGRAFIA**

Western Digital Corporation.  
Communication Product Handbook,  
June 1984.

Stieglitz Mark.  
Local Network Access Tradeoffs,  
Computer Design,  
October 1987.

Stieglitz Mark.  
Local Network. Token Passing Cashes in with Controller Chip.  
Electronic Design,  
October 1982.

Stallings William.  
Data and Computer Communication.  
Macmillan Publishing.  
1985

Stallings William.  
Local Area Networks.  
Macmillan Publishing.  
1978

**APENDICE A**

**FUNCIONAMIENTO DE LA RED**

### PRINCIPIO DE FUNCIONAMIENTO DEL WD 2840

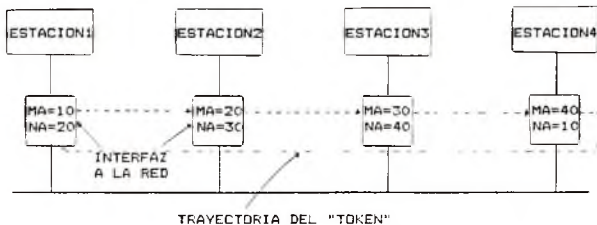


FIGURA A.1 Red Local con Topología de Línea Compartida.

La red tendrá una topología de línea compartida y cada estación (microcomputadora, periférico u otro dispositivo) que requiera el uso del canal de comunicación utilizará una interfaz que le llamaremos interfaz a la red local (figura A.1).

La interfaz a la red local manejará todo lo relacionado a la comunicación entre las estaciones, permitiendo que cada estación se dedique completamente a la aplicación para la que fue diseñada sin que se tome en cuenta el protocolo de comunicación desarrollado por cada interfaz.

Para tener el completo control de las comunicaciones dentro de la red, la interfaz está constituida principalmente por un controlador que maneja el algoritmo de acceso a la red, la recepción y transmisión de paquetes así como la recuperación de condiciones de error en la red, por otro lado se tiene un microprocesador que trabaja cooperativamente con el controlador en la recepción y transmisión de paquetes y/o comandos de la estación a la interfaz de red o viceversa. El circuito controlador de red utilizado es el WD 2840 y el microprocesador utilizado es el IAPX 80386.

## METODO DE ACCESO

El método de acceso utilizado por el controlador está basado en el uso de "tokens". Un "token" es un mensaje que se le envía a una estación para darle el derecho temporal y exclusivo de transmitir por el medio de comunicación, un derecho que la estación debe entregar a la siguiente estación designada una vez que haya transmitido sus mensajes o cuando no haya mensajes que transmitir. Para clarificar este método de acceso vamos a referirnos a la figura A.1. En dicha figura cada interfaz de acceso a la red se muestra con dos de los registros internos del WD 2840, el registro MA que contiene la dirección única de la interfaz (y por lo tanto de la estación asociada) y el registro NA que contiene la dirección de la estación a la que se le enviará el "token" una vez que la estación que lo posea lo haya utilizado.

Supongamos que en un instante dado la estación con MA=10 posee el "token", esta estación podrá transmitir sus mensajes a cualquier otra estación que esté conectada a la red, una vez que haya terminado la transmisión, el controlador enviará el "token" a la estación direccionada por NA, en este caso a la estación 20, todas las estaciones dentro de la red detectan la presencia del "token", pero sólo aquel controlador cuyo registro MA tenga dirección igual a la dirección del "token" lo tomará y, por lo tanto, el derecho temporal y exclusivo de transmitir. En este caso la estación con el registro MA=20 tomará el "token" y transmitirá todos sus mensajes, al terminar de transmitir sus mensajes cederá el "token" a la estación direccionada por el registro NA=30 (en el caso de no haber mensajes a transmitir el controlador pasará el "token"), en este caso el controlador con MA=30 nuevamente toma el "token" y ejecuta el proceso descrito anteriormente; dicho controlador enviará el "token" a la estación MA=40 y finalmente esta estación después de hacer uso del "token" enviará éste a la dirección MA=10, de esta manera se forma un anillo lógico que va de la estación 10 a 20, de 20 a 30, de 30 a 40 y de 40 a 10, repitiendo una y otra vez esta secuencia.

Se debe notar que el número de las estaciones dentro del anillo lógico no necesita ser continuos (la red permite estaciones

con direcciones de 1 hasta 254), ésto trae como consecuencia que fácilmente se puedan anexas o quitar estaciones sin necesidad de modificar la dirección de éstas.

#### **INICIALIZACION Y RECUPERACION DE ERRORES**

En la descripción del método de acceso se observa que el controlador que posea el "token" deberá pasarlo a la dirección indicada por NA, si por alguna causa la estación no está en línea, lo cual es indicado por su carencia de respuesta, el controlador nuevamente envía el "token" a esa misma estación, después de un tiempo, si aún no hay respuesta de la estación a la que se le envió el "token", el controlador que envió por última vez el "token" entra en un proceso de búsqueda de la siguiente estación en línea. Para ello envía el "token" a la dirección NA+1, nuevamente espera un cierto tiempo para que responda dicha estación, si la estación con dirección NA+1 no está en línea, el controlador envía el "token" a la estación NA+2, de esta manera va incrementando la dirección del "token" por cada intento fallido de pasar éste, cuando el controlador envía el "token" y la estación en turno responde, entonces el controlador termina su proceso de búsqueda y escribe la dirección de la estación que respondió exitosamente en su registro NA, de esta manera cuando el controlador posea el "token" nuevamente éste será enviado a la dirección de la siguiente estación en línea, sin necesidad de repetir el proceso anterior. Para clarificar el proceso descrito de la recuperación de la red a fallas de la estación, veamos un ejemplo refiriéndonos a la fig.A.1. Supóngase que en un instante dado la estación con MA=10 posee el "token" y lo envía a la estación 20, por algún motivo la estación con MA=20 está fuera de línea (es decir no forma parte de la red, ni lógica ni físicamente). La estación 10 al enviar el "token" por vez primera inicializa un temporizador TA (interno en el controlador de red), el cual al llegar a cero le indica que es el tiempo máximo para que responda la estación a la que se le envió el "token", como aún no hay respuesta (dado que la estación 20 no está en línea) nuevamente se le envía el "token" a la estación 20 inicializando a



su vez el temporizador TA, al expirar el tiempo y no recibir respuesta mientras tanto, el controlador con MA=10 empieza el proceso de búsqueda y envía el "token" a la dirección 21 (es decir  $NA+1 = 20+1 = 21$ ), inicializando el temporizador TA, como no se encuentra esta estación en la red, el temporizador TA expirará sin que haya respuesta de 21, por lo que el "token" es enviado a la dirección 22 ( $NA+2$ ), dado que, como se muestra en la figura A.1, no hay estaciones en el intervalo 21 a 29, el proceso anterior se repetirá hasta que el "token" sea enviado a la estación 30, como dicha estación sí está en línea, responderá al envío del "token" antes de que expire el temporizador TA de la estación 10, en esta forma el controlador con MA=10 redefine su registro NA=20 por NA=30, de modo que el anillo lógico queda conformado de 10 a 30, de 30 a 40 y de 40 a 10, saltándose la estación 20 que está fuera de línea. Este proceso de búsqueda fue inicializado por la falla de la estación 20, ahora bien, este proceso también puede ser inicializado por una trama de control enviada a una estación en particular o a todas las estaciones. Una trama de control redefine  $NA=MA+1$ . La política de esta trama de control es definida por el programa del usuario, como se requiera por la aplicación.

La seguridad de la recuperación descrita anteriormente descansa sobre los temporizadores TA de cada controlador en la red, siempre y cuando se haya enviado un dato que requiera ser confirmado o que se haya enviado el "token" a la siguiente estación, pero si por algún motivo falla la estación que posee el "token" y no hay manera de inicializar el temporizador TA, entonces en ese momento ningún temporizador TA estará activo y por lo tanto no habrá recuperación de este evento, para evitarlo cada controlador consta además de otro temporizador que está activo desde que se inicializa el controlador, este temporizador se denomina TD y su tiempo de expiración es mucho mayor que el del temporizador TA, de manera que todos los controladores que forman parte de la red tienen su temporizador corriendo cuando sucedió este evento. El temporizador TD de cualquier controlador que expire primero, deberá tomar la responsabilidad de reclamar el

"token", si ésto es permitido en el controlador, si no le es permitido a cualquier controlador reclamar el "token" e inicializar el proceso de recuperación a cualquier controlador cuyo TD haya expirado, simplemente espera hasta que otro controlador reclame el "token" en la red e inicialice el proceso de recuperación, enviando el "token" a la dirección que indique su NA, de esta manera su temporizador TA inicializa el conteo y el proceso de recuperación anteriormente mencionado se repite.

El proceso de recuperación de la red por un controlador, cuando le es permitido recuperar la red, y ha expirado su temporizador TD, es utilizado en la inicialización de la red. La inicialización de la red consiste en formar un anillo lógico entre todos los controladores que deseen comunicarse. Al inicio cuando se energiza la interfaz, la programación del microprocesador 8088 es la responsable de dejar los valores adecuados en los registros MA, NA y los temporizadores TA y TD. Si de antemano se sabe cuál es la configuración de la red, el registro NA de cada controlador se programará con la dirección del siguiente controlador a pasar la estafeta, y de lo contrario, si no se sabe cuál será la configuración, los registros NA de los controladores se programan con el valor de su registro MA + 1. Entre todos lo controladores que deseen formar parte del anillo lógico, al menos uno tendrá la opción de recuperar la red cuando su temporizador TD expire, permitiendo así la generación y transmisión de la estafeta. Una vez que la estafeta es generada, si los registros NA de todos los controladores tienen la dirección correcta de la siguiente estación en línea, el anillo lógico es formado, de lo contrario, cuando algún controlador desee pasar la estafeta a alguna estación que no se encuentre en línea, se entrará a los procedimientos de recuperación de la red, auxiliados por los temporizadores TA y TD; esta recuperación es hecha por cada controlador que desee formar parte del anillo y que no contenga actualizado su registro NA, hasta que el anillo quede formado.

Es necesario notar, en este punto, que la programación del microprocesador 8088 sólo se responsabiliza de comandar al modo de red al controlador, una vez que esté en este modo, el controlador es el responsable del protocolo de comunicación dentro de la red.

Una vez que el anillo lógico ha sido formado, se pueden anexar más estaciones al anillo. La programación del microprocesador 8088 es la responsable de la anexión de todas las estaciones, sin embargo, la responsabilidad se limita a la transmisión de tramas de control dentro de la red para anexar las estaciones, y el controlador modifica su registro NA para introducir a la estación en el anillo lógico ya formado.

A continuación se darán tres métodos primarios de anexión de estaciones, de los cuales se tomará el más conveniente para ser implantado en el microprocesador 8088.

#### **Método Distribuido:**

En este método cada estación dentro de la red puede buscar nuevas estaciones que deseen formar parte del anillo lógico entre el espacio de direcciones que va desde la dirección MA hasta la dirección NA. Este método no se relega a una estación específica, de manera que no hay trabajos gastados en un administrador. En el método distribuido cada estación tiene la misma responsabilidad de permitir el acceso a estaciones en la red. Este método es el que requiere mayor interacción con el microprocesador de la interfaz y requiere además que cada estación mantenga un temporizador para indicar el tiempo en que el controlador debe buscar nuevas estaciones. Veamos un ejemplo: Supóngase que en la figura A.1 cada estación tiene un temporizador de cinco segundos y, por ejemplo, el temporizador de la estación 10 expira, lo cual le indica que una nueva estación puede ser anexada dentro del espacio 11, 12, 13, 14, 15, 16, 17, 18, ó 19. El microprocesador de la interfaz pone una trama en la cadena de transmisión para que sea enviada por la estación 10 con destino a la estación 11, pidiendo además

que la estación 11 mande un "acuse de recibo" sobre el resultado de la transmisión. Si la estación 11 está presente responderá, de otra manera el temporizador TA expirará y el controlador abortará el intento marcando la trama. El microprocesador de la interfaz, cuando lo requiera, examinará el estado de la transmisión de la trama. Aquí el microprocesador verá si la estación 11 respondió o no. Si la estación 11 respondió la estación 10 cambiará su registro NA a 11, permitiendo a su siguiente "token" ser pasado a la estación 11. Dependiendo de lo sofisticado de la aplicación se le puede enviar un mensaje de control a la estación 11 para que actualice su registro NA con la dirección 20, de manera que la estación 11 no necesite entrar en el procedimiento de búsqueda para encontrar su sucesor. Si la estación 11 no respondió al mensaje de la estación 10, la estación 10 actualiza un contador de modo que la siguiente vez que su temporizador de 5 segundos expire se interroga a la estación 12, ésto continúa hasta que se pregunta por la estación 19, si ésta responde se anexa al anillo y se inicializa el ciclo con la estación 11, y si no responde, el ciclo se inicia nuevamente con la estación 11. Debe notarse que las estaciones 20, 30 y 40 a su vez repiten este procedimiento.

#### **Método Centralizado:**

En el método centralizado de adición de estaciones sólo una estación busca en el espacio completo de direcciones por nuevas estaciones que deseen formar parte del anillo lógico. Una razón para centralizar esta función puede ser el tener un mejor control para ser implementado en la red. También con este método pueden haber optimizaciones, por ejemplo, la estación central que hace las búsquedas, puede llevar el control de cuántas estaciones existen en la red, y por lo tanto saltar algún espacio de direcciones. Supongamos que alguna nueva estación de la figura A.1 lleva el control de la búsqueda de nuevas estaciones, cuando llegue a buscar por direcciones mayores que la 40, si de antemano se sabe que no hay más de 40 estaciones, no hay necesidad de buscar estaciones mayores que la dirección 40, por lo tanto la estación central buscará si está la estación 1 inmediatamente

después de buscar si está en línea la estación 39. Este método es parecido al método distribuido, excepto que una sola estación hace todo el trabajo. Cuando la estación central encuentre una nueva estación que desee ser anexada al anillo lógico, la estación central mandará un mensaje de alto nivel a la estación en línea, cuya dirección sea la inmediata anterior a la nueva estación, para que cambie su registro NA a la dirección de la estación que desea entrar al anillo, y la estación central, mandará otro mensaje de alto nivel a la nueva estación, para que éste actualice su registro NA y evite el procedimiento de búsqueda de su sucesor.

### **Exploración Central:**

La exploración central es el método más simple para anexar estaciones a la red. Este método involucra el envío de una trama de control a todas las estaciones, la cual fuerza a cada una, a actualizar su registro NA a la dirección de la estación (NA+1).

Nuevamente veamos la figura A.1, supongamos que la estación 10 manda una trama de control dirigida a todas las estaciones de la red, dicha trama hace que la estación 20 actualice su registro NA a 21, la estación 30 actualiza NA = 31, y la estación 40 actualiza su registro NA = 41, de manera que la estación 20 mande el "token" a la estación 21, si ésta no se encuentra en línea, buscará nuevas estaciones, hasta alcanzar la dirección de una nueva estación, o la siguiente estación en línea (estación 30 en dicha figura), en su oportunidad esto mismo sucede con las otras estaciones dentro de la red. La desventaja de este método, es la enorme cantidad de tiempo incurrido por la búsqueda simultánea. No necesariamente la trama de control debe de ser dirigida a todas las estaciones, ya que se tiene un previo conocimiento de cuáles son las direcciones que tienen las posibles estaciones a anexar; la trama de control se puede enviar a la estación con dirección más próxima al espacio de búsqueda.

El compromiso de todos estos métodos es la distribución de la complejidad de la programación. Si el usuario del controlador de red asume una mayor responsabilidad, dando más inteligencia

distribuida en la programación, el sistema puede ser más sofisticado en el manejo de nuevas estaciones.

## **INTERFACES**

El controlador de acceso a la red, dentro de la interfaz a la red local, posee dos interfaces, el lado hacia el microprocesador (BOBB) y el lado hacia la red.

La interfaz hacia la red es convencional desde el punto de vista eléctrico. La interfaz del modem es del tipo "half duplex" NRZ con reloj y datos separados. Cuando el controlador está listo para transmitir, éste envía la señal RTS y espera CTS. La señal RTS es generada para habilitar el transmisor del modem. Después de un preámbulo dependiente del sistema el modem envía la señal CTS, la cual permite al controlador de red empezar la transmisión. La entrada del controlador SD es usada en la recepción para indicar una portadora válida, si esta señal es negada cualquier instante dentro de la recepción de un paquete, el controlador tratará al mensaje como un error y lo abortará. Esta señal es usada para aumentar la integridad del mensaje más allá de la utilizada por el método del CRC, permitiendo que un modem detecte y reporte fallas de bajo nivel (tales como una portadora fuera de frecuencia o falla de reloj).

La interfaz hacia el microprocesador (8088) involucra dos interfaces funcionales. Los registros de control/estado y la interfaz de DMA.

Los registros de control/estado son utilizados por el microprocesador de la interfaz para inicializar el controlador de red y para obtener información de éste cuando el controlador se encuentre en el modo de red. Una vez que el controlador de red ha sido debidamente programado, para que éste pueda cumplir en tiempo real con todas sus funciones dentro de la red, necesita un manejo complejo de su memoria "buffer" mediante accesos directos a memoria (DMA). El controlador de red maneja cooperativamente esas estructuras con el microprocesador de la interfaz.

El controlador de red tiene tres estados funcionales, el primer estado es el estado de aislamiento, y es el estado que mantiene cuando se le da potencia al controlador, en este estado se inicializan los registro del controlador y se le da la dirección de una área en memoria conocida como el bloque de control, este bloque de control tiene la estructura que muestra la fig. A.2.

Cuando el controlador de red se le comanda a su estado de red él lee y usa los primeros cinco bytes del bloque de control, los 11 restantes bytes de los contadores de eventos son accedados por el controlador sólo cuando la condición de algún evento específico ocurre.

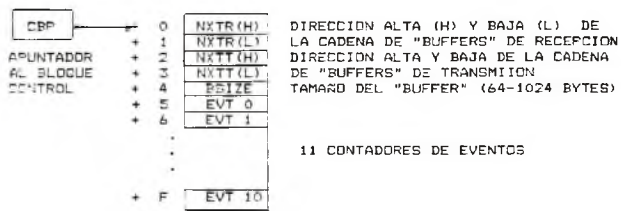


FIGURA A.2: ESTRUCTURA DEL BLOQUE DE CONTROL

Las direcciones de las cadenas de "buffers" de recepción NXTR y transmisión NXTT dentro del bloque de control pueden inicialmente ser cero, en esa situación, el controlador de red espera que la cadena sea extendida por el microprocesador de la interfaz, cambiando el apuntador del bloque de control a la dirección de inicio de la cadena, después de esto, cualquier apuntador que sea cero deberá de estar en los "buffers".

Dado que el controlador de la red maneja listas encadenadas para las cadenas de "buffers" de recepción y transmisión, dichas listas encadenadas son manejadas por el microprocesador de la interfaz y "seguidas" por el controlador de la red, por lo tanto el microprocesador de la interfaz debe de mantener la dirección inicial y final de las cadenas; las direcciones iniciales de las cadenas son puestas en el bloque de control, de modo que el controlador de red puede leer cuál es esa dirección inicial y después "seguir" los apuntadores para localizar los "buffers" de recepción y transmisión. En cualquier instante que el controlador detecte que se ha llegado al final de la cadena de recepción o transmisión, inmediatamente afectará unas banderas (NXTRD para recepción y NXTTD para transmisión) del controlador, el microprocesador de la interfaz deberá monitorear el estado de esas banderas para que pueda actualizar las direcciones inicial y final de las cadenas a medida que agrega (o remueve) "buffers" a (de) las cadenas. El controlador de red provee interrupciones para indicar cuando el controlador ha avanzado a lo largo de la cadena y banderas para indicar cuál es el estado de los apuntadores.

El controlador de red utiliza "buffers" de tamaño constante y su longitud es puesta en la localidad BZISE del bloque de control. Esta longitud del "buffer" es incluyendo los bytes de control y los bytes de los apuntadores. Asociados a cada trama hay contadores de bytes de manera que las tramas en la red no necesiten ser múltiplos enteros de los "buffers".

#### **INTERFAZ DE LA MEMORIA PARA TRANSMISION**

Cuando es recibido el "token" por el controlador y sucede que la transmisión de datos es habilitada, y el controlador de accesos ha llegado a su límite, entonces el controlador determinará si hay tramas de datos a transmitir; si no hay datos que transmitir, el controlador enviará el "token" a la siguiente estación dentro del anillo, y si hay tramas el controlador transmitirá la primera trama de la cadena de transmisión. Cualquier trama será la última



en ser transmitida si alguna de las siguientes condiciones ocurre:

- El controlador es cambiado a su estado de aislamiento.
  - La transmisión de datos es deshabilitada.
  - El byte de control de trama indica última trama.
  - El contador de tramas alcanzó el límite máximo.
  - No hay más tramas en la cadena de transmisión.

al cumplirse cualquiera de las cuatro primeras condiciones se efectuará el paso del "token". Si la última trama no requiere un reconocimiento del receptor, el controlador enviará el "token" junto con la trama de información (si es permitido en el controlador) de lo contrario un paso normal del "token" se efectuará. El controlador lee y evalúa la dirección de la siguiente trama al final de la trama anterior o cuando el "token" es recibido, esto da la información necesaria para actualizar la bandera NXTTO.

La estructura de la cadena de transmisión se muestra en la fig.A.3

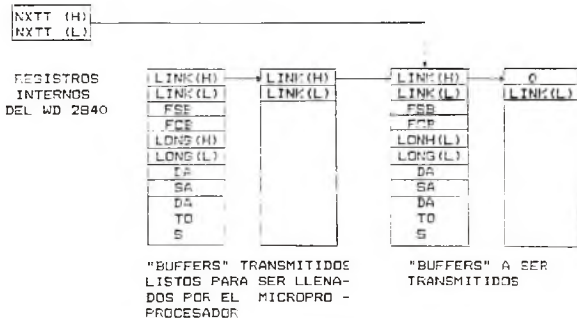


FIGURA A.3: ENCADENAMIENTO DE "BUFFERS" DE RECEPCION

La descripción de términos de las figuras A.3 y A.4 es la siguiente:

LINK(H) y LINK(L) son los apuntadores de "buffers".

FSB es el byte de estado de trama.

FCB es el byte de control de trama.

LONG(H) y LONG(L) son la longitud de la trama.

DA dirección destino de la trama.

SA dirección fuente de la trama.

Cuando la transmisión empieza el controlador lee la dirección deI siguiente "buffer", el byte de control de trama y la longitud de la trama, entonces el controlador empieza a leer y enviar datos hasta que el número de bytes transmitidos iguala la longitud de bytes en la trama; si el número de bytes no alcanza en un "buffer" entonces el controlador toma el apuntador al siguiente "buffer" y continúa enviando datos hasta que la longitud de bytes es alcanzada.

Si la trama requiere un reconocimiento del receptor (esta información se da en el byte de control de trama, así como si es la última línea a transmitir), entonces el controlador espera un cierto tiempo (hasta que el temporizador TA expire) y dependiendo de si hubo o no respuesta y cuál fue el estado de la trama enviada, es escrito por el controlador de red esa información en el byte de estado.

El microprocesador de la interfaz puede anexar tramas al final de la cadena cuando lo desee, y puede remover los "buffers" ya transmitidos, tan sólo monitoreando el FSB de la trama y la bandera NEXTC.

#### **INTERFAZ DE LA MEMORIA PARA RECEPCION**

Después de que el controlador de red detecta el tercer byte de una trama de datos de entrada, éste empezará a colocar la trama de datos en las cadenas de "buffers" de recepción si las siguientes condiciones se cumplen:

- Receptor habilitado.
- Hay "buffers" disponibles en la cadena de recepción.
- La trama está direccionada a este nodo, a todos los nodos o el microprocesador de la interfaz seleccionó el modo COPY.

Si la trama de datos continúa y se necesita más de un "buffer" para contenerla, entonces el controlador inspecciona el apuntador al siguiente "buffer", si éste apunta a un "buffer", el controlador continúa llenando ese "buffer" con la trama y al final de la recepción indica en el FSB (byte de estado de trama) que la recepción ha terminado, de lo contrario, si se detecta que la liga no apunta a un "buffer" entonces se reclaman los "buffers" que habían sido llenados por la trama y se trata a ésta como un aborto. Si la trama continúa y hay "buffers" disponibles, el llenado de los "buffers" continúa hasta que cualquiera de los siguientes eventos sucede:

- 1.- Si el DMA hacia el microprocesador fue demasiado lento entonces en el "buffer" de recepción del controlador se traslapan los datos. ("receive overrun").
- 2.- Si la trama actual ha usado 16 "buffers".
- 3.- Si el byte de detección de errores no es correcto.

Si la trama direccionada a este controlador requiere un reconocimiento, entonces el controlador que recibió enviará una trama indicando el estado de la recepción, independientemente si la recepción fué o no exitosa.

Es responsabilidad del microprocesador de la interfaz asegurar que hay "buffers" disponibles para recepción.

En la figura A.4 se muestra cuál es la estructura de encadenamiento de "buffers" para recepción.

Cuando el controlador recibe una trama, mediante una interrupción le indica al microprocesador de la interfaz que en la memoria hay una trama lista para ser procesada, de esta manera el microprocesador envía esta trama a la estación conectada a la

interfaz, y el "buffer" que contenía la trama puede ser reutilizado en la cadena de recepción para nuevas tramas.

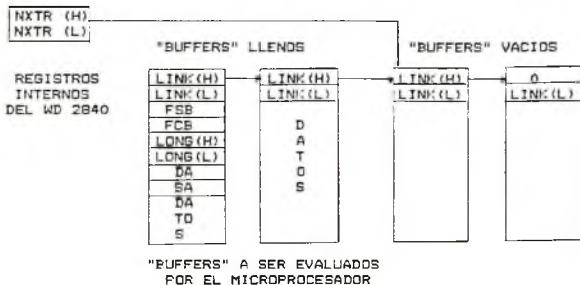


FIGURA A.4: ENCADENAMIENTO DE "BUFFERS" DE RECEPCION

**APENDICE B**

**DIAGRAMA ELECTRICO DE LA INTERFAZ**

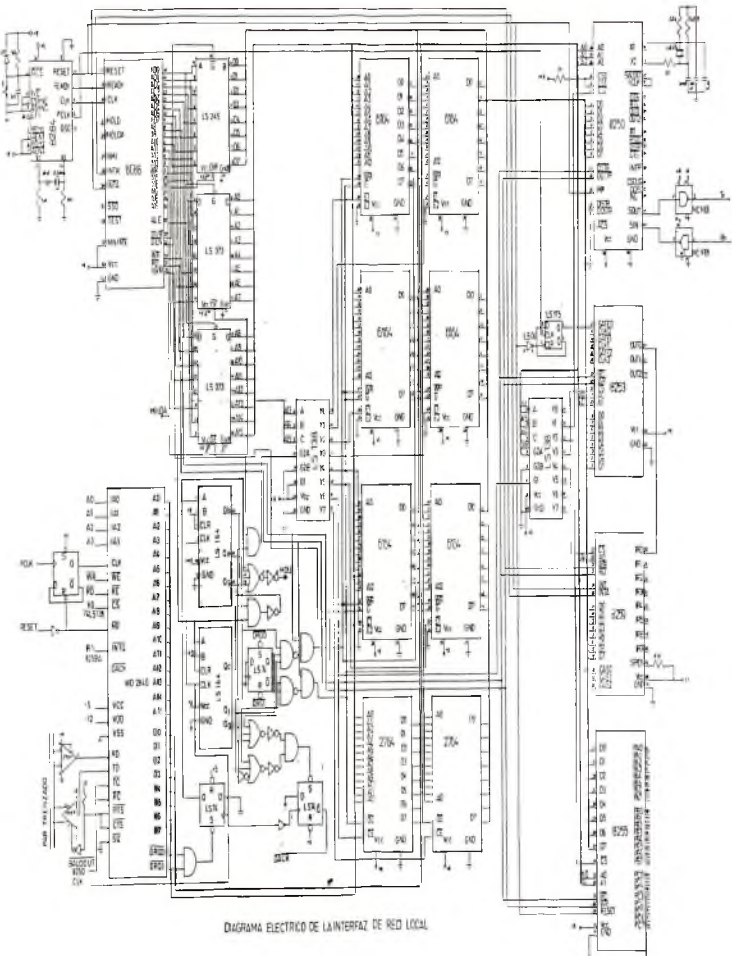


DIAGRAMA ELECTRICO DE LAINTERFAZ DE RED LOCAL

APENDICE C

CODIFICACION DEL PROGRAMA EJEMPLO

DIAGRAMA DE FLUJO DEL  
PROGRAMA PRINCIPAL

PREPARAR AREA DE LA PILA  
PROGRAMAR 8250  
PROGRAMAR 8259A  
PROGRAMAR WD 2840  
PARA PROBAR EL DMA

HUBO ERROR?

MENSAJE DE  
ERROR DEL DMA

MENSAJE DE PRUEBA  
POSITIVA DEL DMA

PREPARAR BLOQUE DE  
CONTROL DEL WD 2840

PREPARAR "BUFFERS"  
DE RECEPCION

PREPARAR "BUFFERS"  
DE TRANSMISION

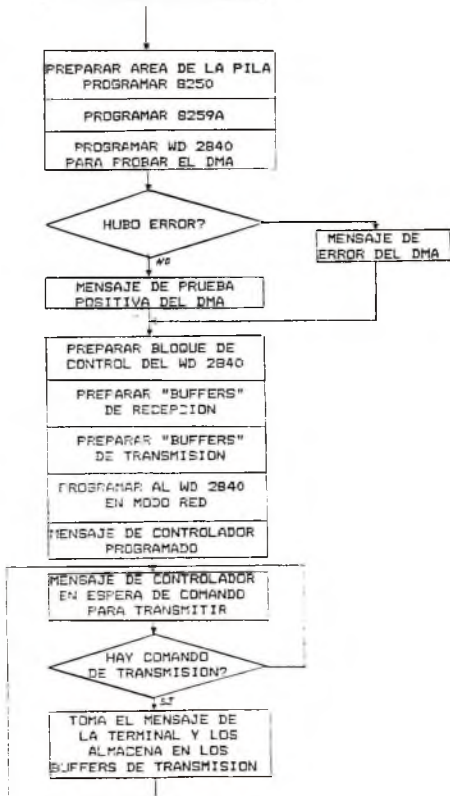
PROGRAMAR AL WD 2840  
EN MODO RED

MENSAJE DE CONTROLADOR  
PROGRAMADO

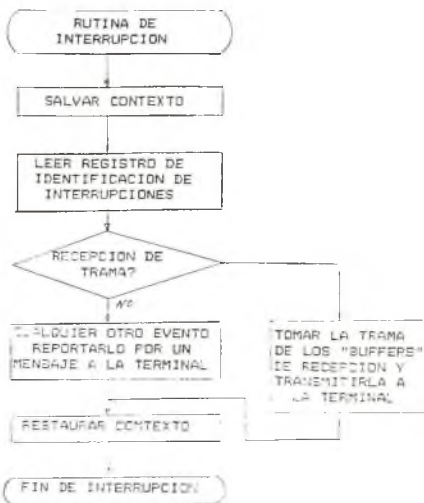
MENSAJE DE CONTROLADOR  
EN ESPERA DE COMANDO  
PARA TRANSMITIR

HAY COMANDO  
DE TRANSMISION?

TOMA EL MENSAJE DE  
LA TERMINAL Y LOS  
ALMACENA EN LOS  
BUFFERS DE TRANSMISION







programa principal para recibir y transmitir tramas de 64 bytes  
de longitud. Se programara al wd2B40 para que efectue una prueba  
de DMA, y si la prueba es exitosa se llevara al TAC al modo red.

name prueba2

datos segment

org 24h

t9ip dw inter ;vector de interrupcion.

t9cs dw ?

org 0200h;variables de un byte.

da db 0 ;DA del TAC

fcdb db 0 ;FCB Frame Control Byte  
;variables de dos bytes.

apdat dw 0;apuntador de datos.

aprec dw 0;apuntador de buffers de recepcion.

apva dw 0 ;apuntador de buffers vacios.

last dw 0 ;apr. al ultimo buffer transmitido.

temp dw 0 ;apr. temporal de buffers Tx.

org 0430h

regc db 00 ;localidades utilizadas para la

regd db 00 ;prueba de DMA .

org 1000h

buftra db 400h dup(0);area de buffers de transmision.

org 1410h ;bloque de control.

apbufr dw 0;apuntador a buffers de recepcion.

apbuft dw 0;apuntador a buffers de transmision.

bufsize db 0;longitud de los buffers.

cont db 0bh dup(0);11 contadores de eventos.

org 1B00h

bufrec db 400h dup(0);area de buffers de recepcion.  
;constantes utilizadas.

linft equ 1000h;limite inferior de la cadena Tx.

isupt equ 1400h;limite superior de la cadena Tx.

linfr equ 1B00h;limite inferior de la cadena Rx.

isupr equ 1c00h;limite superior de la cadena Rx.

buize equ 40h;longitud de los buffers.

;parametros del TAC.

```
ta equ Offh ;temporizador TA.
td equ Offh ;temporizador TD.
cbph equ 14h;aptr. al bloque de control alto.
cbpl equ 10h;aptr. al bloque de control bajo.
nar equ 01h ;direccion de la estacion siguiente.
ahdt equ 0;tokens a recibir antes de Tx.
txlt equ 0 ;limite de tramas a transmitir.
ma equ E2h ;direccion unica del TAC.
datos ends
pila segment stack
    org 0400h
    dw 40h dup(?)
tope equ this word
pila ends
prin segment
    org 0f800h
    assume cs:prin, ss:pila, ds:datos
    mov ax,00
    mov ss,ax
    mov sp, offset tope;inicializa stack.
    mov ds,ax;inicializa segmento de datos.
    mov es,ax;inicializa segmento extra.
    cld ;bandera de direcciones en incremento.
    ;inicializa E250
    mov al,50h;habilita
    out (10h),al;DLAB.
    mov al,00 ;parte alta
    out (11h),al;del divisor.
    mov al,60h ;parte baja
    out (10h),al;del divisor.
    mov al,06 ;programa
    out (13h),al; E250.
    mov al,00 ;deshabilita
    out (11h),al;interrupciones.
    ;inicializa E259A
    mov al,13h;ICW1.
```

```
out (40h),al
mov al,08h;ICW2.
out (41h),al
mov al,01h;ICW4.
out (41h),al
mov al,0fdh;mascara.
out (41h),al
mov ax,offset inter ;carga vector de interrupcion.
mov t9ip,ax
mov t9cs,cs
sti ;habilita interrupciones del 8088.
mov si,offset msg1;mensaje de interfaces programadas.
call write;manda a pantalla.
mov al,80h
out (01h),al;habilita modo de diagnostico.
e1: in al,(05h)
test al,02h ;espera STATE
jz e1
;
; ;carga los registro del TAC
mov ax,offset regc
out (0bh),al;LSB de RAM.
mov al,ah
out (0ah),al;MSB de RAM.
mov al,'C' ;carga al reg.
out (0ch),al;0C con 'C'.
mov al,'D' ;carga al reg.
out (0dh),al;0D con 'D'.
mov al,B9h;habilita NUDIAG
out (01h),al;y DMAT.
e2: in al,(01h);prueba NUDIAG.
test al,01h
jnz e2
mov si,offset msg4;mensaje de fin de prueba.
call write;mandar a pantalla.
;
; ;procesamiento de los resultados
mov si,offset msg2;mensaje negativo de la prueba.
```

```
in  al,(0Ch);dato escrito en el TAC.
cmp  al,'C';compara con 'C'.
jne  err ;si no son iguales hay error.
in  al,(07h);dato leído por el TAC.
cmp  al,'C';compara con 'C'.
jne  err ;si no son iguales hay error.
mov  al,regc;dato escrito por el TAC.
cmp  al,'C';compara con 'C'.
jne  err ;si no son iguales hay error.
in  al,(0Dh);dato escrito en el TAC.
cmp  al,'D';compara con 'D'.
jne  err ;si no son iguales hay error.
in  al,(06h);dato leído por el TAC.
cmp  al,'D';compara con 'D'.
jne  err ;si no son iguales hay error.
mov  al,regd;dato escrito por el TAC.
cmp  al,'D';compara con 'D'.
jne  err ;si no son iguales hay error.
mov  si,offset msg3;mensaje positivo de la prueba.
err: call write;transmite el mensaje.
mov  al,00h;regresa al estado de aislamiento
out  (01h),al;generando una interrupcion.
      ;preparar bloque de control.
mov  ax,offset bufrec; toma la direccion de la cadena
xchg ah,al;de recepcion.
mov  apbufr,ax;escribe la direccion en el bloque.
mov  apbuft,0;direccion nula de Tx en el bloque.
mov  bufsize,0;longitud de buffers = 64 bytes.
mov  last,offset apbuft ;inicializa la variable last.
call prebr;prepara buffers de recepcion.
call prebt;prepara buffers de transmission.
      ;inicializacion del TAC.
mov  al,ta
out  (03h),al ;valor del timer TA.
mov  al,td
out  (09h),al;valor del timer TD.
```

```
mov al,cbph
out (0ah),al;apuntador alto al CB.
mov al,cbpl
out (0bh),al ;apuntador bajo al CB.
mov al,nar
out (0ch),al;siguiente estacion en el anillo.
mov al,aholt
out (0dh),al;tokens a recibir antes de Tx.
mov al,txlt
out (0eh),al;limite de tramas a transmitir.
mov al,ma
out (0fh),al;direccion unica de la estacion.
mov al,00
out (00h),al;TAC al modo red.
mov si,offset msg5;mensaje de tac programado.
call write
pru: in al,(05)
test al,02;prueba STATE.
jnz pru ;espera.
mov si,offset msg6;mensaje de TAC en modo red.
call write
mov al,09h
out (0ih),al;CRI con INIT, 3RING y NEWNA.
wait1: in al,(01)
test al,01;prueba si acepto NAR.
jnz wait1;si no acepto espera.
mov al,0e0h ;habilita recepcion y transmision
out (00),al;del TAC.
poll: mov si,offset msg16;mensaje de TAC en espera de
call write;comando para transmitir.
call read;lee el caracter de la terminal.
call trans;Hace eco del caracter.
cmp al,'T';si es 'T' transmite se ejecuta.
jnz poll ;de lo contrario no se ejecuta.
call trans
jmp poll ;regresa al loop.
```

```
                ;fin del programa principal.
prebt    proc            ;subrutina para preparar buffers Tx.
        mov    bx,offset buftra ;bx = temp = linft.
ept1:    mov    byte ptr [bx],00 ;(temp + 0) = 0.
        mov    byte ptr 2[bx],00 ;(temp + 2) = 0.
        mov    ax,bx;ax = temp.
        add    ax,bsize;ax = temp + bsize.
        mov    1[bx],al;(temp + 1) = low (temp + bsize).
        mov    bx,ax;temp = temp + bsize.
        cmp    bx,lsupt;temp = lsupt ?
        jne    ept1 ;si no son iguales continua.
        mov    apva,linft;apva = linft.
        mov    temp,linft;temp = linft.
        ret
prebt    endp
prebr    proc            ;subrutina para preparar buffers Rx.
        mov    bx,offset bufrec ;bx = aprec = linfr.
epri:    mov    byte ptr 2[bx],00 ;FSB = 0.
        mov    ax,bx;ax = aprec.
        add    ax,bsize;ax = aprec + bsize.
        mov    [bx],ah ;(aprec) = high (aprec + bsize).
        mov    1[bx],al ;(aprec + 1) = low (aprec + bsize).
        mov    bx,ax;aprec = aprec + bsize.
        cmp    bx,lsupr;temp = lsupr ?
        jne    epri ;si no son iguales continua.
        sub    bx,bsize; temp = temp - bsize.
        mov    byte ptr [bx],0;(lasrec) = 0.
        mov    aprec,linfr;aprec = linfr.
        ret
prebr    endp
trans:   proc
        mov    ah,al;salva el dato.
etri:    in    al,(15h);lee reg. de estado.
        test   al,20h;prueba si puede transmitir.
        jz    etri ;si no puede espera.
        mov    al,ah;recupera el dato.
```

```

        out    (10h),al;transmite el dato.
        ret
trans1  endp
readd   proc ;subrutina para leer de la terminal.
        in    al,(15h) ;lee el reg.de estado.
        test  al,01;prueba si hay dato.
        jz   readd;si no hay dato espera.
        in    al,(10h);lee el dato de la terminal.
        ret
readd   endp
write   proc ;subrutina que envia mensajes
        lodsb    ;carga en AL el caracter.
        cmp    al,"*";compara con el terminador.
        je    wend ;termina si son iguales.
        call  trans1;transmite el caracter.
        jmp   write    ;regresa por otro caracter.
wend:   mov    al,0dh;transmite un CR.
        call  trans1
        ret
write   endp
trans   proc
        mov    da,01h;direccion destino 82h.
        mov    fcb,00h;FCB dice esperar por ACK.
        mov    ax,apva
        mov    temp,ax;temp = apva.
        mov    apdat,ax;apdat = apva.
        mov    bx,temp
        mov    al,fcb
        mov    0[bx],al ;escribe el FCB.
        mov    al,da
        mov    6[bx],al;escribe la DA.
        mov    byte ptr 7[bx],a;escribe la SA.
        mov    di,apdat    ;di = apdat.
        add    di,05;apdat = apdat + 5.
etrs:   call  read; toma caracter para la trans.
        call    trans1;hace eco del caracter.

```



```
    stosb    ;guarda el dato en el buffer.
    cmp     al,0dh;compara con fin de trama.
    jne    etrs ;si no es el fin continua.
    mov     byte ptr 4[bx],00h ;carga la longitud de la trama
    mov     byte ptr 5[bx],40h ;en el buffer.
    mov     apva,bx      ;apva = temp.
    add     apva,bsize;apva = temp + bsize.
    xchg    bh,bl;bx = temp.
    mov     si,last;
    mov     [si],bx;(last) = temp.
    xchg    bh,bl
    mov     last,bx;last = temp.
    mov     si,offset msg17;mensaje de fin de trama.
    call    write
    ret

trans     endp
inter     proc
    push    ax          ;salva el contexto
    push    bx
    push    si
    in     al,(03h);lee el reg. de interrupciones.
    test   al,01
    jz     ein1
    call   itda
    jmp    fin
ein1:     test   al,02h
    jz     ein2
    call   ita
    jmp    fin
ein2:     test   al,04
    jz     ein3
    call   itok
    jmp    fin
ein3:     test   al,08h
    jz     ein4
    call   irec
```

```
        jmp  fin
ein4:   test  al,10h
        jz   ein5
        call itran
        jmp  fin
ein5:   test  al,20h
        jz   ein6
        call ins
        jmp  fin
ein6:   test  al,40h
        jz   ein7
        call iror
        jmp  fin
ein7:   test  al,80h
        jz   fin
        call iterr
fin:    mov  al,20h;fin de interuccion para el 8259.
        out (40h),al
        pop  si
        pop  bx
        pop  ax
        iret
inter   endp
irec proc      ;para transmitir trama recibida.
        mov  si,offset msg9 ;mensaje de recepcion de trama.
        call write;transmite mensaje.
        mov  bx,aprec
:       mov  al,2ib>0;al = FSB
:       cmp  al,80h;compara si es el buffer recibido.
:       jne  eir1 ;si no es sale de la rutina.
        mov  si,bx      ;apunta a los datos.
        add  si,0B
eir1:   lods b; carga en al el dato.
        call trans1; manda el dato a la terminal.
        cmp  al,^ch; compara con fin de trama.
        jne  eir2
```

```
eirf:   mov  aprec,bx          ;aprec = aprec + bsize.
        add  aprec,bsize
        ret
irec endp
itdm proc          ;rutina que informa cambio
           ;de estado o timer TD muerto.
        mov  si,offset msg8;mensaje de estado de aislamiento.
        in   al,(00);le le CRO
        test al,01;prueba si esta en aislamiento.
        jnz  etm1 ;salta si esta en aislamiento.
        ret           ;no manda ningun mensaje
;        mov  si,offset msg7 ;mensaje de timer muerto:
ctm1:   call write ;transmite mensaje.
        ret
itdm endp
ita proc
        mov  si,offset msg10;mensaje por ita.
        call write
        ret
ita endp
itok proc
        mov  si,offset msg11;mensaje por itok.
        call write
        ret
itok endp
itran proc
        mov  si,offset msg12;mensaje por itran.
        call write
        ret
itran endp
ins proc
;        mov  si,offset msg13;mensaje por ins.
;        call write
;        ret
ins endp
iror proc
```

```
        mov si,offset msg14;mensaje por iror.
        call write
        ret
iror endp
iterr   proc
        mov si,offset msg15;mensaje por iterr.
        call write
        ret
iterr   endp
msg1 db '8250 y 8259 programados $'
msg2 db 'Prueba de DMA malas $'
msg3 db 'Prueba de DMA buena $'
msg4 db 'Fin de la prueba de DMA $'
msg5 db 'TAC programado $'
msg6 db 'TAC en modo red $'
msg7 db 'Timer TD expiro $'
msg8 db 'Se cambio al estado de aislamiento $'
msg9 db 'Recepcion de trama $'
msg10 db 'Transmision sin exito $'
msg11 db 'Se recibio el token $'
msg12 db 'Se transmitio una trama $'
msg13 db 'Hay nuevo sucesor $'
msg14 db 'Over run en el receptor $'
msg15 db 'Error en el transmisor $'
msg16 db 'Esperando comando T $'
msg17 db 'Se termino la trama $'
prin endp
end
```

El jurado designado por la Sección de Computación del Departamento de Ingeniería Eléctrica del Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional, aprobó esta tesis el 31 de Julio de 1987.



---

Dr. Manuel E. Guzmán Rentería



---

Dr. Armando Maldonado Talamantes



---

Dr. Arturo Veloz Guerrero.



