UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

Estudio Comparativo Sobre Tres Microprocesadores de 16 Bits

Tesis Profesional

Que para obtener el Título de
INGENIERO MECANICO ELECTRICISTA

p r e s e n t a n

FERNANDO ALEMAN ANGELINI
LUIS CORTINA GUERRERO
ROGELIO VALDES DEL RIO

México, D. F. 1983

I N D I C E

APENDICES.

INTRODUCCION

En la vida profesional del Ingeniero, una de las principales actividades que debe realizar es la de selección de uno o más productos para llevar a cabo una tarea o un proyecto, cuyos resultados dependerán de la decisión que el primero tome para la compra de equipo de diferente índole. En el proceso de selección, el Ingeniero debe tomar en cuenta factores tales como: necesidades por satisfacer; recursos económicos; recursos humanos; disponibilidad de equipo; capacidad de éste; tecnología; mantenimiento, etc.

Para poder efectuar una buena selección (si no la óptima), en la mayoría de los casos se debe recurrir a comparaciones profundas de las alternativas que se ofrecen con el objeto de escoger la más apropiada.

En el caso específico de la Ingeniería Electrónica y Computación, los estudios descritos anteriormente deberían realizarse frecuentemente ya que este es uno de los campos que más han progresado desde la época de los 40's y especialmente en los últimos 10 años debido a la introducción de los Microprocesadores. Al seleccionar un microprocesador o una microcomputadora se debe buscar que sea de una tecnología reciente y que posea ciertas características (para evitar que se vea obsoleto en poco tiempo). Además, dado que el campo de la Computación es cada vez más extenso debe escogerse el dispositivo correcto para la aplicación requerida, ya que es muy difícil que una computadora sea 100% eficiente en las muchas aplicaciones en las que puede ser usada.

Este estudio tiene como objetivo el presentar los aspectos más importantes de comparación que deben ser tomados en consideración cuando un Ingeniero en Electrónica o en Computación se enfrente con un problema de selección o evaluación de equipos de microcómputo en su vida profesional. Se consideró de más utilidad hacerlo sobre tres microprocesadores de 16 Bits en lugar de hacerlo sobre microprocesadores de 8 Bits, ya que los primeros representan hasta las últimas fechas los avances más recientes en este campo tecnológico y a los que se tiene acceso.

El desarrollo del estudio está hecho de la siguiente manera: en el Capítulo 1 se hace la presentación de las características funcionales de cada procesador por separado, haciendo resaltar aquellas que son novedosas en particular sobre los otros dos y sobre la generación anterior de microprocesadores. En el capitulo 2 se presenta una descripción breve de los conjuntos de instrucciones de cada procesador, haciendo resaltar instrucciones poco comunes y novedosas. En la parte final de cada uno de estos capítulos, se hacen breves comentarios comparativos de la parte descrita de los tres dispositivos para al final dejar solamente las conclusiones totales del estudio. En estos

comentarios se consideró de utilidad el asignar una evaluación de acuerdo a si la característica comparada es Excelente (E), Buena (B) o Regular (R).

En el capítulo 3 se presentan los programas que fueron desarrollados con el objeto de conocer y manejar el conjunto de instrucciones de cada procesador. En el capítulo 4 se hace una breve descripción de los dispositivos de soporte de cada procesador incluyendo únicamente aquellos dispositivos que son de reciente desarrollo y de mucha utilidad. En el capítulo 5 se presentan las conclusiones finales del estudio.

Se recomienda ampliamente que el lector consulte en forma interactiva los apéndices del libro al estar leyendo cada capítulo con objeto de que la exposición sea más clara y objetiva. Cada vez que se consideró necesario, se presenta un diagrama para ayudar a visualizar lo explicado en forma escrita.

El lector que desee ir más allá de este estudio debe consultar el manual descriptivo de cada microprocesador para lo cual se presenta la Bibliografía que incluye manuales, artículos, estudios etc. la cual fue recopilada lo más extensamente posible.

En el estudio no solo se incluyen la descripción y comparación de cada microprocesador sino también se hace una breve descripción de la familia de soporte de cada uno, con el objeto de dar a conocer las posibilidades de aumentar la capacidad del sistema.

Para darle más validez al estudio se realizó una labor de búsqueda de microcomputadoras en México que utilizaran uno de los microprocesadores involucrados en el estudio con la finalidad de correr los programas presentados y corregirlos antes de ser presentados en esta Tesis. Esto fué logrado gracias a que gentilmente las siguientes personas y compañías facilitaron el uso de instalaciones propias o de su lugar de trabajo:

M. en C. Javier Santoyo V. del CSC U.N.A.M. por haber proporcionado el sistema de Seattle Computer Products 8086 de su propiedad.

M. en C. Luis H. Peñarrieta y el M. en C. Héctor Calvario de IIMAS U.N.A.M quienes proporcionón el sistema Zilog Z8000 ZSCAN.

Industrias Digitales S.A. por su sistema Alpha Micro AM1000 y a Micromex S.A. por permitir usar sus instalaciones para el desarrollo de una parte del seminario de Tesis así como también para la captura y procesamiento de

este trabajo.

Para tener un panorama más general del desarrollo que han tenido los Microprocesadores desde su aparición hasta nuestros días haremos una breve descripción histórica acerca de su evolución y aplicaciones a lo largo de estos años.

El primer microprocesador de 4 Bits (4004) fué desarrollado por Intel en 1971, este fué el resultado de un proyecto para desarrollar una calculadora programable. Mientras se desarrollaba este Microprocesador de 4 Bits, se inició la investigación paralela para desarrollar un Microprocesador de 8 Bits (8008) el cual fue lanzado al mercado un año después. Este microprocesador tenía la capacidad de direccionar 16 Kbytes de Memoria y 45 instrucciones orientadas al manejo de series de caracteres.

En el período de 1973 a 1974, varios fabricantes comenzaron a desarrollar e introducir al mercado otros microprocesadores de 4 y 8 Bits, así surgieron el PPS-4 de Rockwell y otros procesadores introducidos por Fairchild, Texas Instruments, etc.

En 1974, Intel introduce el 8080 que era una versión mejorada de la arquitectura del 8008, tiene mayor capacidad de direccionamiento de memoria, la Pila fué puesta en la memoria RAM en lugar de estar dentro del Procesador, manejo más eficiente de interrupciones, mayor número de puertos de E/S y 78 instrucciones. Poco después, Motorola lanza el 6800 el cual tiene una arquitectura similar al 8080 pero con la característica de que requiere sólo una fuente de alimentación. En 1974 también es desarrollado el PACE de National Semiconductor el cual fué uno de los primeros Microprocesadores de 16 Bits. En 1976 Zilog desarrolla el Z80 el cual es totalmente compatible con el 8080, cuenta con 158 instrucciones, y capacidad de manejo de interrupciones con Vector.

Debido a la necesidad de mayor capacidad de proceso y velocidad surgen los procesadores de 16 Bits, esta generación es iniciada con la introducción en 1978 del 8086 de Intel el cual cuenta con mayor capacidad de direccionamiento de memoria (1 Mbyte), 64 K puertos de E/S, 95 instrucciones básicas y modos de direccionamiento más poderosos, en 1979 Zilog introduce al mercado el Z8000 el cual aparece con características mejoradas con respecto al 8086 como por ejemplo su capacidad de memoria. En 1980 aparece el 68000 que fué desarrollado por Motorola, presentando por primera vez la característica de tener una arquitectura interna de 32 Bits y externa de 16.

Durante el desarrollo de este estudio surgieron los microprocesadores de 32 Bits como son el IAPX 432, el 68020 que es la versión del 68000 con arquitectura externa de 32 Bits y por último el Z80010 de Zilog el cual fue anunciado pocos meses antes de la terminación de este trabajo.

TERMINOS

En la descripción de señales, cualquier nombre de señal que sea seguido por un asterisco (*) indicará que esta señal es negada.

A continuación se presenta una lista de términos utilizados en este trabajo, los cuales no tienen una traducción literal al Español o que son poco utilizados y que fueron cambiados a los términos indicados a continuación:

ACARREO.- "Carry".

ADMINISTRACION.- "Arbitration".

APUNTADOR DE PILA.- "Stack Pointer".

BIDIRECCIONAL COMPLETO.- "Full Duplex".

BUSQUEDA DE INSTRUCCION.- "Instruction Fetch".

CADENA DE CARACTERES.- "String".

CANAL.- Lo que comúnmente se conoce como "BUS".

CAPACIDAD DE PROCESO.- "Throughput".

CIRCUITERIA EXTERNA O INTERNA.- "Hardware"

COLA.- "Queue".

CONTADOR DE TIEMPO O RELOJ.- "Timer".

DISPONIBLE O LISTO.- "Ready".

ETIQUETA.- "Tag".

EXCEPCION O INTERRUPCION INTERNA.- "Trap"

HABILITACION.- "Enable" o "Strobe".

LINEA o SEÑAL.- "Pin".

MANEJADOR.- "Handler".

MODO DE TRAZO.- "Trace Mode".

PROGRAMA(CION).- "Software"

PROTOCOLO.- "Handshake".

RAZON O VELOCIDAD.- "Rate".

REGISTRO.- "Buffer" o "Latch".

RESTAURACION.- "Reset".

SOBREPASO O SOBREFLUJO.- "Overflow".

TRANSCEPTOR.- "Transceiver".

CAPITULO I

1.1 ARQUITECTURA DEL PROCESADOR 8086

1.1.1 INTRODUCCION.- El 8086 es un procesador de 16 bits, puede ejecutar todo el conjunto de instrucciones de los microprocesadores de 8 Bits 8080/8085 y además nuevas instrucciones para 16 bits. Las nuevas instrucciones contienen multiplicación y división, manejo de cadenas y de bits. Puede manejar hasta 1 Mbyte de memoria a través de un Canal de Direcciones de 20 bits multiplexado con el Canal de Datos de 16 Bits. Maneja dos tipos de interrupciones y cuatro Trampas. Puede manejar opcionalmente Entrada/Salida mapeada como memoria y tiene la capacidad para trabajar en sistemas de multiproceso.

1.1.2 MODOS DE OPERACION.- El 8086 puede trabajar en dos modos, el modo máximo y el modo mínimo. El modo mínimo es utilizado para trabajar en sistemas pequeños, donde el Procesador proporciona todas las líneas de control de los Canales para el control de memoria y periféricos. Para configurar al procesador en modo mínimo es necesario conectar la línea MINIMO/MAXIMO (MN/MX*) a 5 volts. El modo máximo se utiliza para trabajar en sistemas más grandes y es necesario tener un Controlador de Canal, el cual decodifica las líneas de control del Procesador y las extiende a todo el sistema. Para que el procesador opere en este modo es necesario conectar la línea MN/MX a Tierra. Las líneas que quedan libres en el Procesador pueden ser utilizadas para trabajar con otros procesadores si así lo requiere la aplicación. La capacidad de direccionamiento de 1 Mbyte de Memoria es la misma para los dos modos de operación. Estos modos se muestran en la figura 1.

Las líneas de Status S0-S7 son usadas sólo en modo máximo y sirven para proporcionar información del tipo de operación que está efectuando el procesador. Los Códigos de Status proporcionados son los siguientes:

S2	S1	S0	FUNCION
0	0	0	Reconocimiento de interrupción
0	0	1	Lectura de E/S
0	1	0	Escritura de E/S
0	1	1	Paro
1	0	0	Búsqueda de instrucción
1	0	1	Lectura de Memoria
1	1	0	Escritura de memoria
1	1	1	No hay ciclo de Canal

TABLA DE LINEAS DE STATUS

Los Bits de Status S3 a S7 son multiplexados con la parte alta del Canal de Direcciones. Los bits S3 y S4 indican a cuál Registro de Segmento se hará el acceso. S5 tiene el estado de la bandera de interrupción, S6 es cero y S7 no es utilizado.

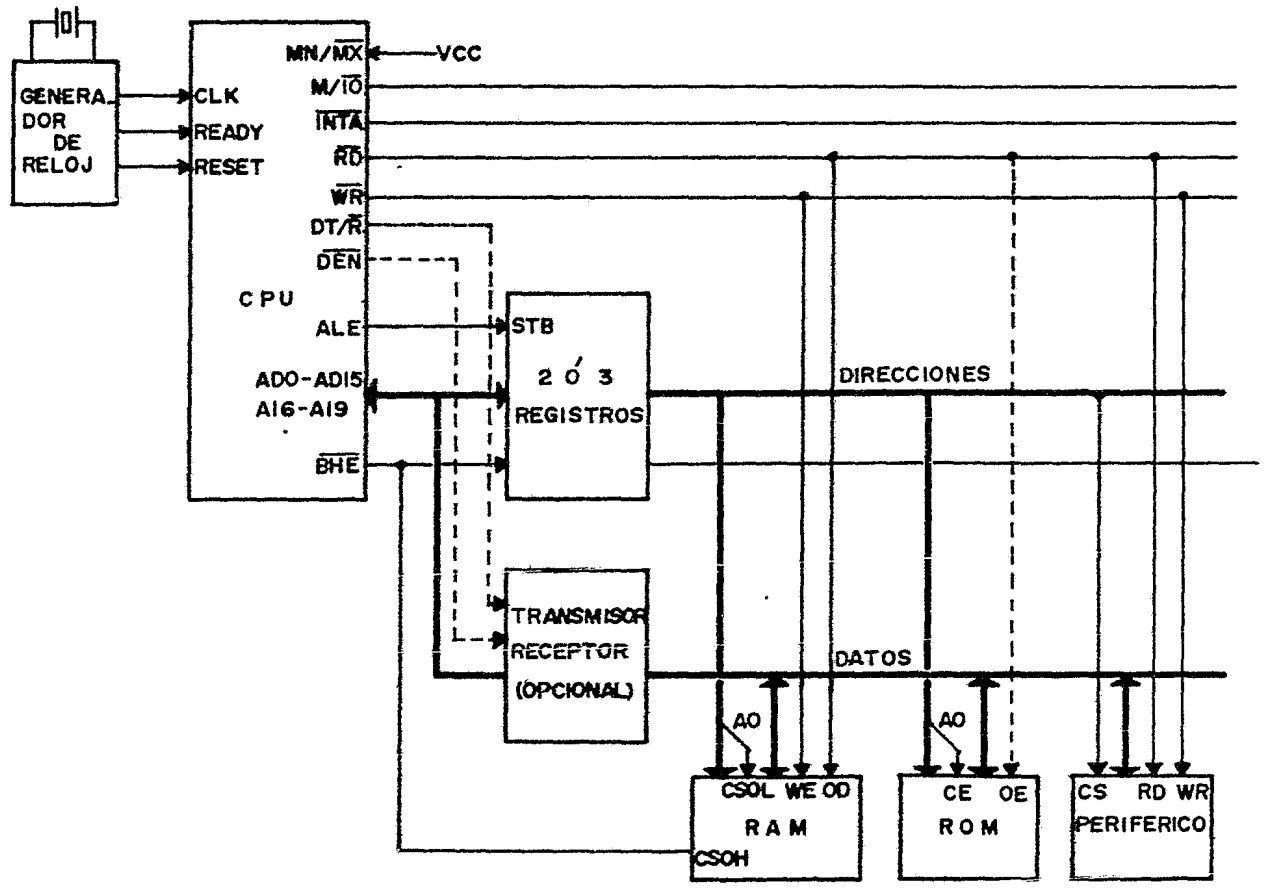


FIG IA CONFIGURACION EN MODO MINIMO

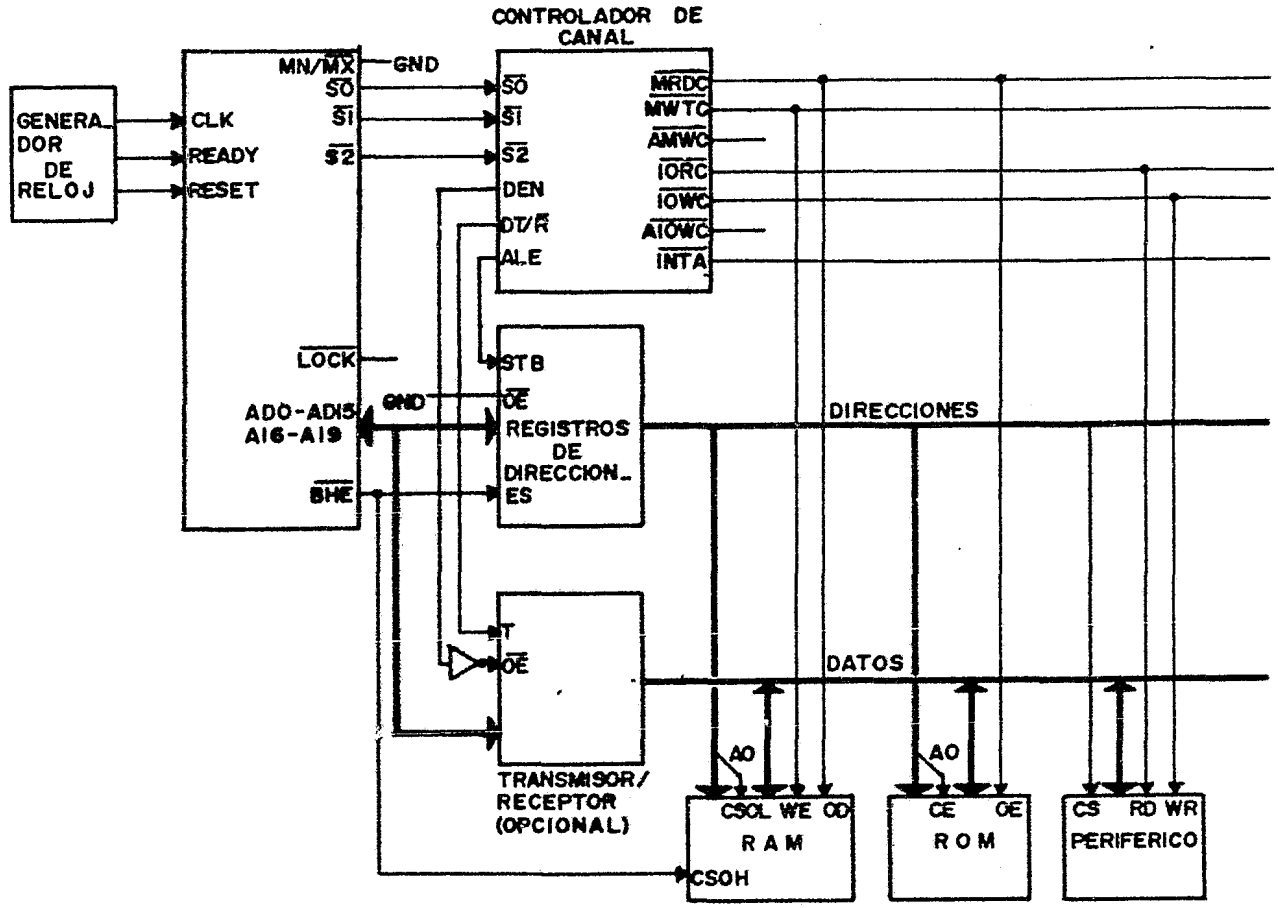


FIG 18 CONFIGURACION EN MODO MAXIMO

1.1.3 ARQUITECTURA INTERNA.-El 8086 está dividido en dos unidades de proceso básicas que son la Unidad de Ejecución (EU) y la Unidad de Interfase al Canal (BIU). Las dos se muestran en el siguiente diagrama a bloques .

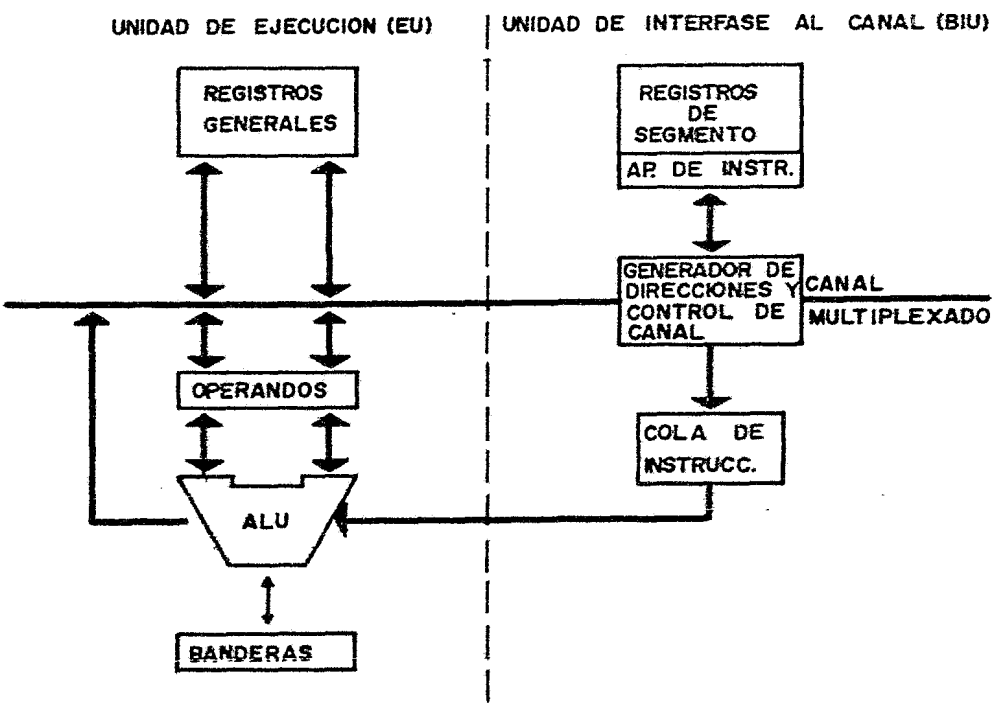


FIG 2 ESTRUCTURA INTERNA

UNIDAD DE EJECUCION (EU).- La Unidad de Ejecución, comprende la Unidad Aritmética Lógica (ALU), los Registros de Propósito General y el Registro de Banderas de Status y Control.

Esta unidad no tiene conexión directa con el Canal del Sistema pues las instrucciones son obtenidas de una Cola de Instrucciones que le proporciona la Unidad de Interfase al Canal, cuando una instrucción requiere que un dato sea leído de o almacenado en memoria o dispositivo periférico, esta Unidad recurre a la Unidad de Interfase para que efectúe

esta tarea. Todas las direcciones manejadas por la Unidad de Ejecución son de 16 Bits por lo que es necesario que la Unidad de Interfase realice una relocalización de éstas para poder tener acceso a todo el espacio de direcciones.

UNIDAD DE INTERFASE AL CANAL (BIU).- Esta Unidad se encarga de efectuar todas las operaciones relacionadas con los Canales, como son transferencia de datos entre el Procesador y memoria o dispositivos de E/S.

Una característica muy importante en cuestión de rapidez y eficiencia es que cuando la Unidad de Ejecución (EU) está procesando instrucciones y no requiere del Canal, la Unidad de Interfase se encarga de buscar instrucciones en memoria y colocarlas en una Cola de Instrucciones de seis bytes, lo que permite reducir el tiempo de búsqueda de instrucciones para la Unidad de Ejecución. A esta acción se le llama Prebúsqueda de Instrucciones. Si la Unidad de Ejecución procesa una instrucción que transfiere el control a otra localidad de memoria, la Unidad de Interfase restaura la Cola de Instrucciones y busca la instrucción en la nueva localidad de Programa.

Las dos unidades pueden funcionar independientemente una de la otra en muchas circunstancias. El resultado es que el tiempo empleado para la búsqueda de instrucciones es muy pequeño ya que esta búsqueda se efectúa mientras la Unidad de Ejecución procesa instrucciones.

1.1.4 DESCRIPCION DE REGISTROS.- El Procesador cuenta con Registros de Propósito General, Registros Apuntadores e Indice, Registros de Segmento, un Registro Apuntador de Instrucciones y un Registro de Banderas y Control.

REGISTROS DE PROPOSITO GENERAL.- Dentro de la Unidad de Ejecución se encuentran 8 registros de 16 bits, que están divididos en dos conjuntos de cuatro registros cada uno: el grupo HL, el cual contiene cuatro Registros de Datos que son: el Registro Acumulador (AX), Registro Base (BX), Registro Contador (CX) y Registro de Datos (DX); el grupo PI el cual contiene dos Registros Apuntadores y dos Registros Indice que son: El Registro Apuntador de Pila (SP) y el Registro Apuntador de Base (BP), el Registro Indice de Fuente (SI) y el Registro Indice de Destino (DI). Los Registros de Datos también pueden ser direccionados separadamente como registros de ocho bits y pueden servir como Acumuladores en la mayoría de las operaciones.

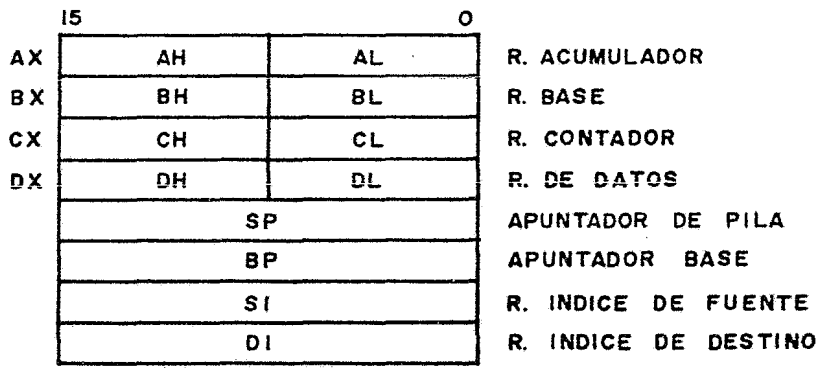


FIG 3 REGISTROS DE PROPOSITO GENERAL

REGISTROS DE SEGMENTO.- Para poder direccionar 1 Mbyte de memoria en segmentos de 64K bytes, el Procesador tiene que hacer uso de los Registros de Segmento, los cuales se localizan dentro de la Unidad de Interfase. Estos Registros de 16 Bits son: el Registro de Segmento de Código (CS), el Registro de Segmento de Datos (DS), el Registro de Segmento de Pila (SS) y el Registro de Segmento Extra (ES). El Procesador tiene acceso directo a estos cuatro registros, dentro de los cuales están contenidas las direcciones base de cada segmento.

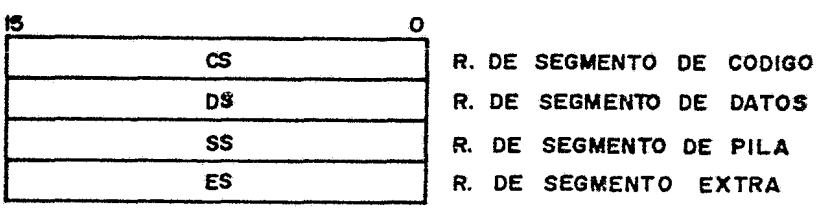


FIG 4 REGISTROS DE SEGMENTO

APUNTADOR DE INSTRUCCION.- Es un Registro de 16 bits que contiene el desplazamiento en bytes de la siguiente instrucción con respecto a la dirección base del Segmento de Código actual.

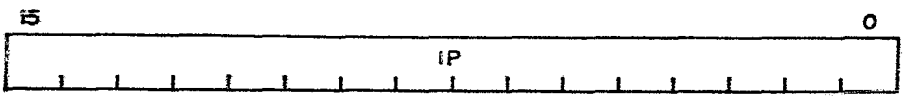


FIG 5 APUNTADOR DE INSTRUCCION

REGISTRO DE BANDERAS Y CONTROL.- Es un Registro de 16 Bits que se encuentra dentro de la Unidad de Ejecución y contiene las siguientes 9 Banderas:

FIG 6 REGISTRO DE BANDERAS Y CONTROL

ACARREO (CF).- (Bit 0)

PARIDAD (PF).- (Bit 2)

ACARREO AUXILIAR (CF).- (Bit 4)

CERO (ZF).- (Bit 6)

SIGNO (SF).- (Bit 7)

TRAMPA (TF).- (Bit 8) Encendiendo esta bandera el procesador entra automáticamente en el modo de ejecución paso a paso. En este modo se genera una interrupción interna cada vez que se ejecuta una instrucción. Esta característica se puede utilizar para depuración de programas.

HABILITACION DE INTERRUPCION (IF) - (Bit 9) Encendiendo esta bandera, el procesador tiene la capacidad para atender interrupciones mascarables, si está apagada no atenderá interrupciones. Esta Bandera no tiene ningún efecto sobre las interrupciones no mascarables.

DIRECCION (DF).- (Bit 10) Encendiendo esta bandera se provoca que los apuntadores de las instrucciones de manejo de cadenas (SI y DI) sean decrementados automáticamente. Si la bandera está apagada, los apuntadores son incrementados.

SOBREFLUJO (OF).- (Bit 11)

1.1.5 ORGANIZACION DE MEMORIA.- El espacio de Direccionamiento de Memoria de 1 Mbyte del procesador se encuentra dividido en segmentos de hasta 64 Kbytes cada uno. Los segmentos pueden ser definidos adyacentes, parcialmente traslapados o totalmente traslapados sin ningún problema. Los programas deberán poner la Dirección Base de los segmentos específicos en los Registros respectivos para poder hacer uso de ellos.

El procesador tiene 20 líneas para el direccionamiento de memoria (Canal de Direcciones). La memoria está organizada en un banco alto (D15-D8) y un banco bajo (D7-D0) de 512K bytes cada uno direccionados en paralelo. Los bytes con direcciones pares son transferidos a través de las líneas D0-D7, mientras que los bytes con direcciones impares son transferidos a través de D8-D15. Para la selección de los Bytes Bajo o Alto se utilizan las líneas Habilitación de Byte Alto (BHE*) y la Línea de Direcciones (A0), la selección se efectúa de acuerdo a la siguiente tabla:

BHE*	A0	TRANSFERENCIA DE BYTES
0	0	AMBOS BYTES
0	1	BYTE ALTO HACIA/DE UNA DIRECCION IMPAR
1	0	BYTE BAJO HACIA/DE UNA DIRECCION PAR
1	1	NINGUNO

SELECCION DE BYTES BAJO Y ALTO

Si una palabra en memoria ocupa la dirección "n" (donde n=par) entonces el Byte menos significativo estará localizado en esta Dirección y el Byte más significativo estará en la Dirección n+1.

FIG 7 ORGANIZACION DE MEMORIA

Una Dirección válida de memoria en el Canal Multiplexado es indicada por la afirmación de la línea Habilitación de Registro de Dirección (ALE). Un Dato válido es especificado por la línea Habilitación de Datos (DEN*).

GENERACION DE DIRECCIONES FISICAS.- Para poder hacer cualquier referencia a memoria es necesario tener una dirección física, la cual deberá obtenerse de la dirección lógica con que se trabaja. La dirección física es un valor de 20 bits que identifica a cualquier byte en el espacio de memoria de 1 Mbyte. La dirección lógica consiste de una Dirección Base de Segmento y de un Desplazamiento. Para cualquier referencia a memoria, el valor de la Dirección Base es la dirección del primer byte contenido en el segmento, y el Desplazamiento es la distancia en bytes en la cual se encuentra el operando con respecto a esa Dirección Base.

La manera en que la Unidad de Interfase genera una dirección física es la siguiente: primero se toma la Dirección Base contenida en el Registro de Segmento y se corre cuatro lugares hacia la izquierda, luego se le suma el desplazamiento formando así la Dirección Física de 20 Bits.

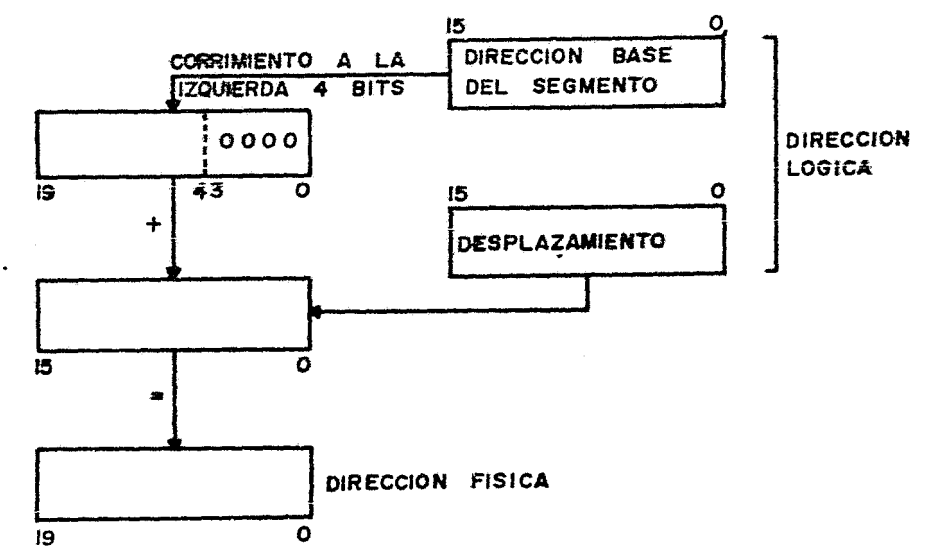


FIG 8 GENERACION DE DIRECCIONES FISICAS

ENTRADA/SALIDA.- El espacio de entrada salida del 8086 es de 64K para puertos de 8 bits y de 32K para puertos de 16 bits. El espacio de entrada salida es no segmentado y para

trabajar con un puerto bastará colocar su dirección en los 16 Bits menos significativos del Canal de Direcciones, las líneas de dirección A16-A19 son siempre cero para operaciones de Entrada Salida.

También se pueden manejar dispositivos de Entrada Salida mapeados como memoria. En este caso cualquier instrucción de acceso a memoria puede ser usada para trabajar con ellos.

1.1.6 MANEJO DE INTERRUPCIONES.- Las Interrupciones al Procesador están divididas en dos clases: Internas y Externas. Las Interrupciones Externas pueden ser a su vez de dos tipos: Mascarables y No Mascarables. Las Interrupciones Internas pueden ser de 4 tipos: provocadas por las instrucciones Interrupción (INT), Interrupción en Sobrepaso (INTO), División Entre Cero y por el modo de Ejecución Paso a Paso.

La prioridad de Interrupciones Externas e Internas en orden de Mayor a Menor es la siguiente:

PRIORIDAD	INTERRUPCION
1	Error de División
1	Instrucción INT
1	Instrucción INTO
2	Interrupción No Mascarable
3	Interrupción Mascarable
4	Ejecución Paso a Paso

TABLA DE PRIORIDADES DE INTERRUPCION

INTERRUPCIONES NO-MASCARABLES.- Se tiene una Interrupción no Mascarable, la cual tiene prioridad sobre las interrupciones mascarables y es solicitada al procesador a través de la línea Interrupción No Mascarable (NMI*). Este tipo de interrupción no puede ser deshabilitada. Cuando es reconocida, el control es transferido a la rutina de servicio apuntada por el Vector 2 de la Tabla de Vectores de Interrupción.

INTERRUPCIONES MASCARABLES.- El procesador cuenta con la línea Interrupción Mascarable (INTR) para la solicitud de Interrupciones, que serán atendidas sólo si la bandera IF está encendida. Si se presenta una interrupción cuando la bandera está apagada, el Procesador no la atenderá , continuará ejecutando instrucciones normalmente. Si está encendida, cuando el Procesador termine de ejecutar la instrucción en proceso atenderá la interrupción. Usualmente

la línea INTR es manejada por un Controlador de Interrupciones Programable al cual se conectan los dispositivos que requerirán servicio por Interrupciones. La principal función de este Controlador es determinar cuál dispositivo tiene mayor prioridad para la interrupción.

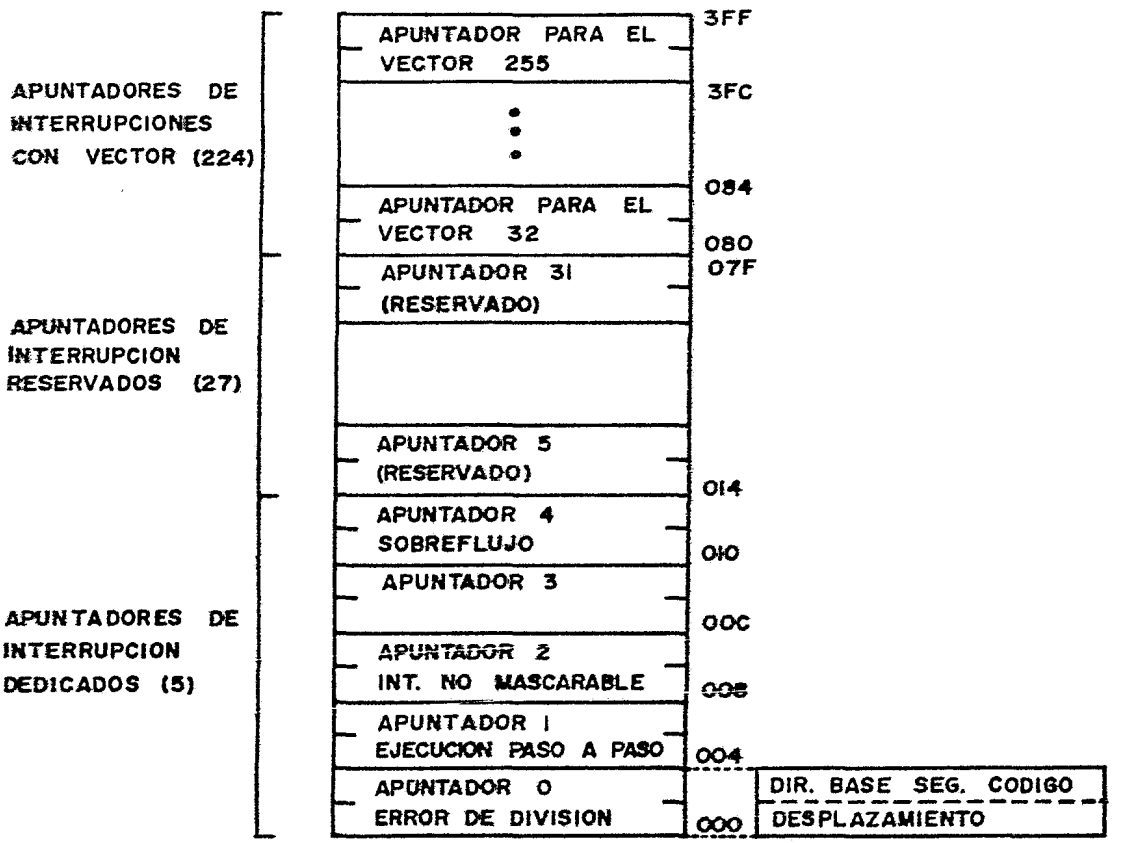


FIG 9 TABLA DE APUNTADORES DE INTERRUPCION

PROCEDIMIENTO DE RECONOCIMIENTO DE INTERRUPCIONES.- Cuando se inicia el proceso de Reconocimiento de Interrupción el Registro de Banderas y Control, el Registro de Segmento de Código y el Apuntador de Instrucción son colocados en la Pila. A continuación, los bits TF e IF del Registro de Banderas y Control son apagados. Los demás Registros no son salvados y será responsabilidad de la

rutina de atención de la Interrupción salvarlos y restaurarlos antes de terminar con el proceso de ésta.

El Procesador reconoce la interrupción ejecutando dos ciclos consecutivos de reconocimiento de interrupción lo que es indicado a través de la línea Reconocimiento de Interrupción (INTA*). Si el Procesador está configurado en modo máximo, se activa la señal LOCK* para indicar a otros procesadores que no intenten tomar los Canales. Si la interrupción es mascarable, en el primer ciclo se le indica al Controlador de Interrupciones Programable que la interrupción ha sido aceptada. El Controlador responde colocando el vector de interrupción sobre el Canal de Datos (los Vectores de Interrupción son programados en el Controlador cuando este es inicializado), el Procesador lee el Vector y lo multiplica por cuatro, debido a que cada dirección de Rutina de Servicio ocupa cuatro Bytes dentro de la Tabla de Vectores de Interrupción. Este valor es usado como un apuntador en la Tabla de Vectores de Interrupción donde se localiza la Dirección de la rutina de atención de la interrupción.

INTERRUPCIONES INTERNAS.- Las Interrupciones Internas son reconocidas de la misma manera que las externas. Las interrupciones internas comprenden las trampas provocadas por las instrucciones INT (Interrupción) e INTO (Interrupción en sobrepaso), también son provocadas por errores en división (división entre cero) y modo de ejecución paso a paso.

1.1.7 MANEJO DE LOS CANALES.- El procesador configurado en Modo Máximo puede permitir que otros dispositivos inteligentes dentro del sistema hagan uso de los Canales de Datos/Direcciones y Control a través de las Líneas Solicitud/Reconocimento Cero (RQ/GT0) y Solicitud/ Reconocimiento Uno (RQ/GT1) las cuales son Bidireccionales con RQ/GT0 teniendo mayor prioridad que RQ/GT1 y funcionan de la siguiente manera: primero, el Procesador que desea control de los Canales envía un pulso a través de una de las líneas de Solicitud/Reconocimento. Segundo, el Procesador responde con un pulso sobre la misma línea indicando que ha Reconocido la Solicitud de los Canales liberando éstos y tercero, el procesador que solicitó el control de los Canales envía un Pulso a través de la misma línea para indicar que la solicitud ha terminado. En modo mínimo el procesador cuenta con las líneas HOLD (Espera) y HLDA (Reconocimiento de Espera) las cuales permiten que otro dispositivo inteligente haga uso de los canales. Este dispositivo puede ser un controlador de DMA el cual deberá activar la línea HOLD para indicar al procesador que requiere hacer uso de los canales, el procesador terminará el ciclo de máquina que este ejecutando e inmediatamente

después emitirá la señal HLDA para indicar que se ha otorgado el control de los canales.

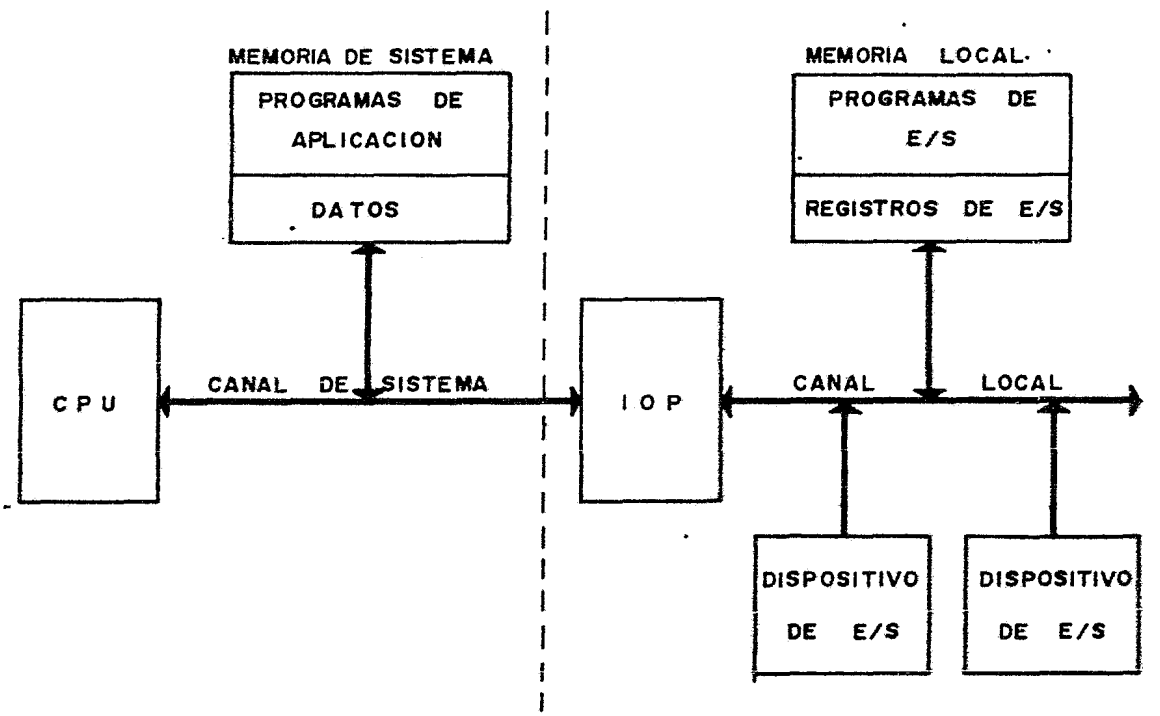


FIG 10 SISTEMA DE MULTIPROCESO

1.2 ARQUITECTURA DEL PROCESADOR Z8000

1.2.1 INTRODUCCION.- El Z8000 es un microprocesador con arquitectura interna y externa de 16 bits. Cuenta con 16 Registros de 16 Bits de propósito general, 3 tipos de Interrupciones y cuatro tipos de Trampas. Su Canal de Datos y Direcciones es multiplexado. Su capacidad de direccionamiento segmentado directo es de 8 Mbytes expandible a 48 Mbytes. Esta capacidad de direccionamiento se obtiene a través de un Canal de Direcciones de 16 Bits y un Canal de Número de Segmento de 7 Bits.

1.2.2 MODOS DE OPERACION.- El procesador puede operar en dos modos básicos, el Modo de SISTEMA y el Modo NORMAL. La diferencia entre éstos es que en Modo de Sistema se puede ejecutar el conjunto de instrucciones completo y en Modo Normal sólo se pueden ejecutar las instrucciones no privilegiadas. El modo de operación es determinado por la bandera Sistema/Normal (S/N*) en el Registro de Banderas y Control en el Procesador. Si la Bandera está prendida el procesador se encuentra en Modo Supervisor y si está apagada se encuentra en Modo Normal.

MODO SISTEMA.- Este es el modo de prioridad dentro del procesador, si el procesador se encuentra en este modo se puede ejecutar cualquier instrucción. También se efectúa todo el procesamiento de Interrupciones y Trampas sin importar el estado del Bit S/N* en el Registro de Banderas y Control. Además, en este modo se efectúan todas las operaciones de Entrada/Salida Normales y Especiales, éstas últimas son usadas para programar la Unidad de Manejo de Memoria.

MODO NORMAL.- En este modo sólo se pueden ejecutar instrucciones no privilegiadas. Las instrucciones privilegiadas son aquellas que afectan de una manera importante el estado del sistema como las instrucciones que modifican contenido del Registro de Banderas y Status o la instrucción Paro (HALT).

El procesador indica al exterior el Modo de Operación a través de la línea Normal/Sistema* (N/S*).

1.2.3 ARQUITECTURA INTERNA.- El Z8000 está basado en un diseño de lógica aleatoria, su estructura básica se muestra en la figura 11, tiene un Canal Interno de 16 bits el cual es utilizado para Direccionamiento y comunicación de Datos. La capacidad de proceso es mejorada a través de un "Pipeline" limitado el cual permite una prebúsqueda de la primera palabra de la siguiente instrucción. Esto ocurre siempre y cuando la instrucción en proceso no requiera de los Canales para terminar el ciclo de ejecucion.

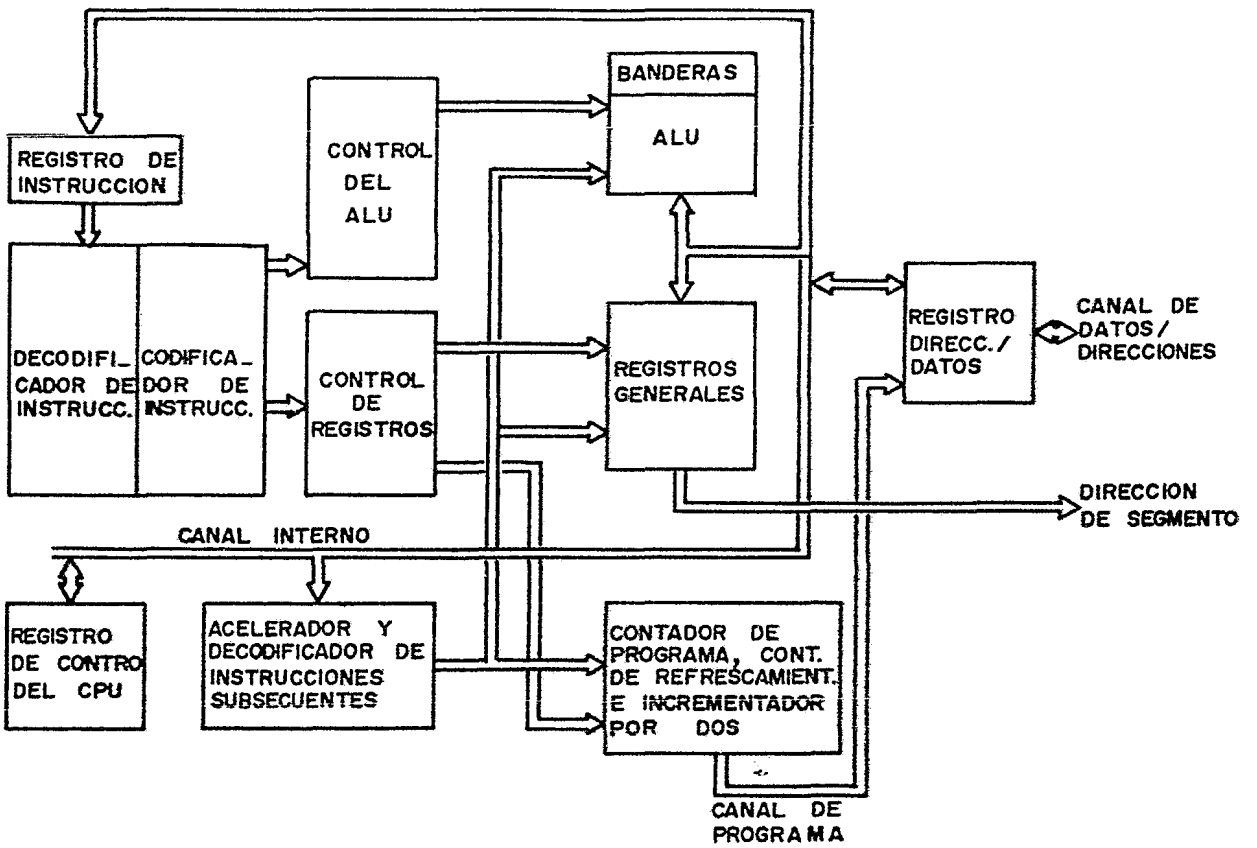


FIG II ESTRUCTURA INTERNA

1.2.3 DESCRIPCION DE LOS REGISTROS.-El procesador cuenta con 16 Registros de 16 Bits de propósito general, además cuenta con un Contador de Programa de 32 Bits, un Registro de Banderas y Status de 16 Bits, un Apuntador de Nuevo Status de Programa de 32 Bits y un Registro de Refrescamiento de Memoria de 16 Bits.

REGISTROS DE PROPOSITO GENERAL.- Este grupo comprende 16 Registros de 16 Bits que pueden ser usados como acumuladores y todos menos uno (R0) como Registros Indice o Apuntadores de Memoria. Para operaciones con Bytes, los primeros ocho Registros pueden ser tratados como 16 Registros de 8 Bits (RL0, RH0...RL7, RH7); para operaciones con Palabras los Registros pueden ser tratados como 16 registros de 16 Bits (R0-R15); para operaciones con Palabras

Largas los Registros son agrupados por pares formando así ocho registros de 32 bits (RR0, RR2....RR14) , para operaciones en 64 Bits los Registros son agrupados en cuartetas formando así cuatro Registros de 64 Bits (RQ0, RQ4...RQ12).

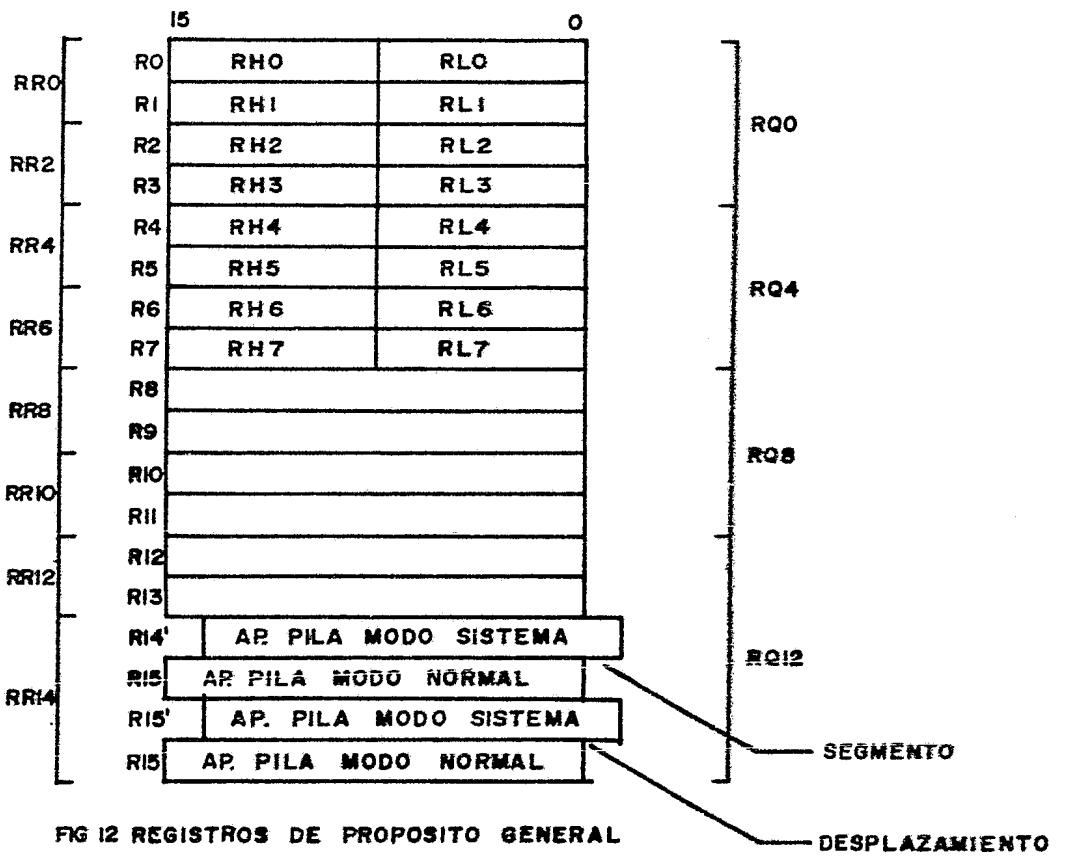


FIG 12 REGISTROS DE PROPOSITO GENERAL

CONTADOR DE PROGRAMA (PC).- El Contador de Programa, es un par de Registros de 16 Bits. Uno de los registros almacena el número de segmento en el cual se está operando y el otro almacena el desplazamiento con respecto a la dirección base del Segmento.

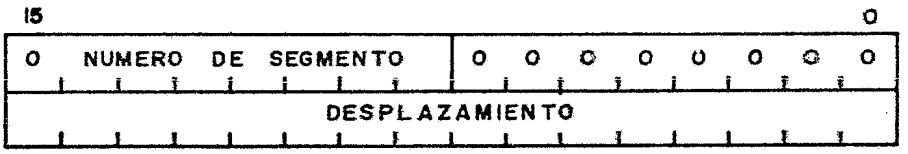


FIG 13 CONTADOR DE PROGRAMA

REGISTRO DE BANDERAS Y CONTROL (FCW).- Este es un Registro de 16 Bits el cual contiene las banderas de status del Procesador. Está dividido en dos bytes, el byte menos significativo contiene las Banderas de Status del ALU y el byte más significativo contiene las banderas de Control del Procesador. En éste Registro, los bits 0,1,8,9 y 10 contienen siempre ceros.

Las banderas contenidas en este Registro son las siguientes:

ACARREO AUXILIAR (H).- (Bit 2)

AJUSTE DECIMAL (DA).- (Bit 3)

PARIDAD/SOBREPASO (P/V).- (Bit 4)

SIGNO (S).- (Bit 5)

CERO (Z).- (Bit 6)

ACARREO (C).- (Bit 7)

HABILITACION DE INTERRUPCION SIN VECTOR (NVIE).- (Bit 11)

HABILITACION DE INTERRUPCIONES CON VECTOR (VIE).- (Bit 12)

ARQUITECTURA DE PROCESAMIENTO EXTENDIDO (EPA).- (Bit 13) Esta bandera indica al procesador que en el sistema existen procesadores adicionales de propósito especial. Si esta bandera está apagada se generarán traps para emular las instrucciones extendidas. Posteriormente se hará una descripción más detallada de esta arquitectura.

MODO (S/N*).- (Bit 14) Esta bandera indica si el Procesador se encuentra operando en Modo Sistema o en Modo Normal.

MODO SEGMENTADO (SEG).- (Bit 15.

FIG 14 REGISTRO DE BANDERAS Y CONTROL

APUNTADORES DE PILA (STACK POINTER SP).- El Par de Registros RR14 es usado como Apuntador de Pila implícito del Procesador. De hecho, este par de Registros está duplicado ya que se tienen Apuntadores de Pila completamente separados para los Modos Sistema y Normal, aunque sólo uno de ellos está activo a la vez dependiendo del estado del Bit S/N* en el Registro de Banderas y Control. El Registro R14 guarda el número de segmento y el Registro R15 guarda el desplazamiento con respecto a la dirección base del segmento.

REGISTRO DE REFRESCAMIENTO DE MEMORIA.- Este registro sirve para controlar un Contador interno para Refrescamiento de Memoria, consiste de un Contador de Renglones de 9 bits, un Contador de Intervalos de 6 bits y un Bit de Habilitación. Funciona de la siguiente manera: El contador de renglones es cargado con una dirección inicial, la cual será incrementada por dos cada vez que el Contador de Intervalo llegue a cero, cuando esto sucede, un ciclo de refrescamiento es iniciado, durante este ciclo el valor contenido en el Contador de Renglones es colocado en el Canal de Direcciones para efectuar el refrescamiento de Memoria.

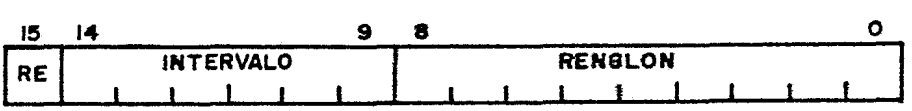


FIG 15 CONTADOR DE REFRESCAMIENTO

REGISTRO APUNTADOR DE AREA DE NUEVO STATUS DEL PROGRAMA (NPSAP).- Este registro guarda un apuntador a una área de memoria la cual está reservada para las direcciones de las rutinas de atención a Interrupciones y Trampas, la descripción detallada del funcionamiento de esta área se hará en la sección dedicada a Interrupciones y Trampas.

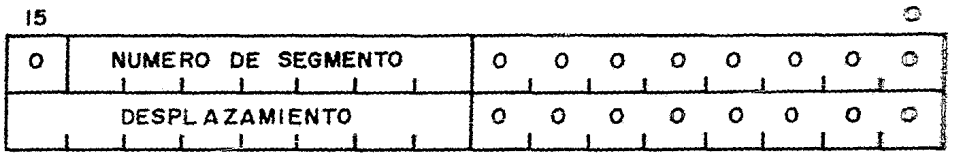


FIG 16 APUNTADOR DE AREA DE NUEVO STATUS DE PROGRAMA

1.2.4 ORGANIZACION DE MEMORIA.- El espacio de direcciones de 8 Mbytes está dividido en 128 segmentos de hasta 64 Kbytes cada uno. Una dirección segmentada de 23 bits usa una dirección de segmento de 7 bits para seleccionar un segmento y un desplazamiento de 16 bits para direccionar cualquier localidad relativa a la dirección base del segmento. Las dos partes de la dirección segmentada pueden ser manipuladas separadamente con todas las instrucciones disponibles para palabras y palabras largas.

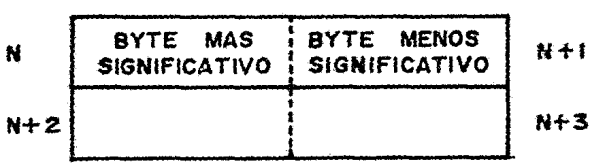


FIG 17A ORGANIZACION DE MEMORIA

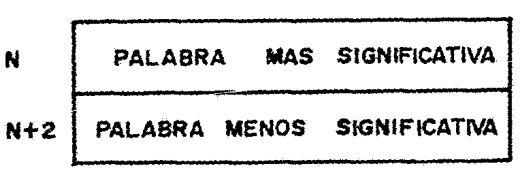


FIG 17B ORGANIZACION DE MEMORIA
(PALABRA LARGA)

La memoria está organizada al igual que los Registros. en bytes, los cuales son direccionables individualmente. Las palabras y las instrucciones son direccionadas siempre en localidades pares, si una palabra se encuentra en la localidad "n" (donde n=par) el Byte más significativo estará en la localidad "n" y el Byte menos significativo en la localidad n+1. Las palabras largas y las cadenas de bytes son direccionadas de la misma manera. Para diferenciar entre el byte menos significativo y el más significativo se usa el

Bit de Direcciones A0 y para diferenciar entre operaciones de Palabra y Byte se usa la línea Byte/Palabra (B/W*). Para diferenciar entre Lectura/Escritura se utiliza la línea Escritura/Lectura (R/W*). Una Dirección válida de memoria sobre el Canal multiplexado es indicada por las Señales Habilitación de Dirección (AS*) y Solicitud de Memoria (MREQ*). Un dato válido sobre el Canal multiplexado es indicado por la Señal Habilitación de Dato (DS*).

Con ayuda de Lógica Externa se puede ampliar el espacio de direcciones hasta 48 Mbytes separando los espacios de Código, Datos y Pila (Stack) para los modos Sistema y Normal a través de la decodificación de las Líneas de status del Procesador ST0-ST3 y una o varias Unidades de Manejo de Memoria (para mayor información acerca de la Unidad de Manejo de Memoria vease el Capítulo sobre Familias de Soporte). El procesador proporciona códigos para cada tipo de operación que efectúa como son referencias a memoria, referencias a la Pila, referencias a E/S, etc.

ST3-ST0	DEFINICION
0000	OPERACION INTERNA
0001	REFRECAMIENTO DE MEMORIA
0010	REFERENCIA DE E/S
0011	REFERENCIA ESPECIAL E/S
0100	RECONOCIMIENTO DE TRAMPA DE SEGMENTACION
0101	RECONOCIMIENTO DE INTERRUPCION NO MASCARABLE
0110	RECONOCIMIENTO DE INTERRUPCION SIN VECTOR
0111	RECONOCIMIENTO DE INTERRUPCION CON VECTOR
1000	SOLICITUD DE DATOS A MEMORIA
1001	SOLICITUD A LA PILA
1010	SOLICITUD DE DATOS A MEMORIA (EPU)
1011	SOLICITUD A LA PILA (EPU)
1100	ACCESO AL ESPACIO DE INSTRUCCION
1101	BUSQUEDA DE INSTRUCCION PRIMERA PALABRA
1110	TRANSFERENCIA DE PROCESOS DE EXTENSION
1111	RESERVADA

TABLA DE DECODIFICACION DE STATUS

Las direcciones manejadas por el programador, usadas en las instrucciones y propocionadas por el procesador son llamadas Direcciones Lógicas. La Unidad de Manejo de Memoria toma las Direcciones Lógicas y las transforma en Direcciones Físicas requeridas para el acceso a memoria. Este proceso de transformación de direcciones es llamado Relocalización. La Relocalización de segmentos hace que las direcciones de programa del usuario sean independientes de las localidades físicas de memoria. Este proceso de relocalización es

completamente transparente al programa y es efectuado en su totalidad por la Unidad de Manejo de Memoria.

MANEJO DE E/S.- El procesador maneja los puertos de E/S completamente separados de la memoria a través de los 16 Bits que especifican el desplazamiento en el Canal de Direcciones con lo cual se puede tener un máximo de 64 K puertos de E/S.

1.2.5. ESTRUCTURA DE INTERRUPCIONES Y TRAMPAS.- El procesador soporta tres tipos de interrupciones: no mascarable, con vector y sin vector. Cuatro trampas: Llamada de Sistema, Instrucción no Implementada, Instrucción Privilegiada y Trampa de Segmentación. Las interrupciones con vector y sin él, son mascarables mediante los Bits de Control respectivos en el Registro de Banderas y Control. De los cuatro tipos de trampas, la única externa es la Trampa de Segmentación, la cuál es generada por la Unidad de Manejo de Memoria y es Solicitada al Procesador a través de la Línea de Solicitud de Trampa de Segmentación (SEGT*). El procesador cuenta con 3 líneas de Solicitud de Interrupción que son: Interrupción No Mascarable (NMI*), Interrupción con Vector (VI*) e Interrupción sin Vector (NVI*).

Las demás trampas ocurren cuando instrucciones limitadas a modo de sistema son usadas en modo normal, o como resultado de una instrucción de Llamada de Sistema la cuál provoca un cambio controlado de Modo Normal a Modo Sistema, o por una instrucción no implementada. El orden descendente de prioridad de interrupciones y trampas es el siguiente:

1.- Trampas Internas
2.- Interrupción no Mascarable
3.- Trampa de Segmentación
4.- Interrupción con Vector
5.- Interrupción sin Vector

Cuando ocurre una Interrupción o Trampa, ésta es reconocida y atendida de la siguiente manera: primero, se efectúa una búsqueda de la primera palabra de la siguiente instrucción a ejecutar como si no hubiera un ciclo de reconocimiento de interrupción. Esta palabra es descartada y el Contador de Programa no es incrementado. Segundo, después de que el ciclo de búsqueda de instrucción es abortado, se ejecuta un ciclo de Reconocimiento de Interrupción, durante este ciclo el dispositivo que interrumpió envía una palabra la cual es interpretada como un Identificador o como Vector de Interrupción en el caso de Interrupciones con Vector. Tercero, el contenido del Contador de Programa y del Registro de Banderas y Control son puestos en la Pila de Modo Sistema junto con el identificador y en seguida se

cargan automáticamente los nuevos valores del Contador de Programa y del Registro de Banderas y Control del área de memoria designada por el Registro Apuntador de Area de Nuevo status de Programa (NPSAP), ésta contiene la dirección base de la tabla de direcciones de las Rutinas de Atención de Interrupciones y Trampas así como la palabra que será cargada en el Registro de Banderas y Control al atender la Interrupción o Trampa.

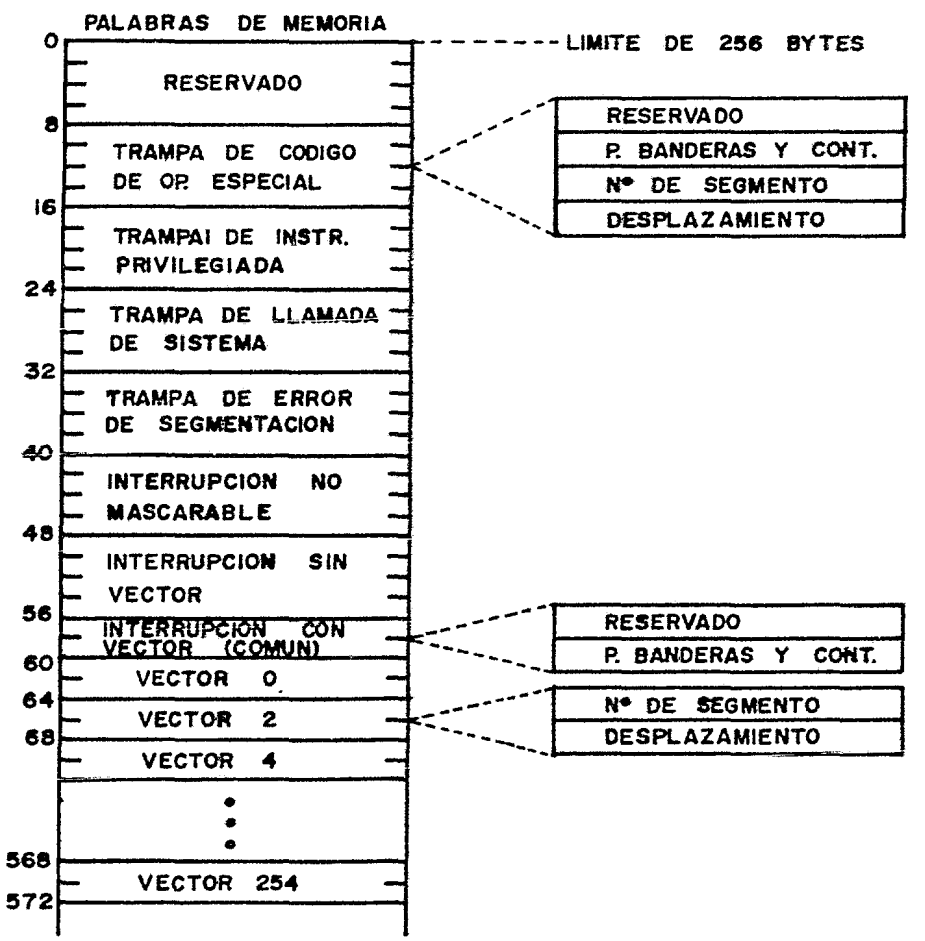


FIG 18 VECTORES DE INTERRUPCION

El Identificador almacenado en la Pila de Modo Sistema es una palabra la cuál identifica la fuente de la Interrupción o Trampa. Para las Interrupciones No Mascarables y Sin Vector los 16 Bits pueden representar información de status del periférico que interrumpe. Para las Interrupciones con Vector la parte baja de la palabra representa el Vector de Salto (256 posibles vectores) y la parte alta puede ser usada para proporcionar información de status adicional.

Para la Trampa de Segmentación, el byte alto es el identificador de la Unidad de Manejo de Memoria y el byte bajo es indefinido. Para trampas internas, el identificador es la primera palabra de la instrucción "atrapada" la cuál puede ser una instrucción privilegiada o una no implementada.

RESTAURACION DEL PROCESADOR.- Un nivel bajo en la Línea de Restauración (RESET) causa que las Líneas del canal de Datos/Direcciones queden en estado de alta impedancia, además las líneas de Control son forzadas a un nivel alto y las líneas de Código de status son forzadas a un nivel bajo, el refrescamiento de memoria es deshabilitado.

Cuando la línea RESET ha sido alta por tres periodos consecutivos de reloj, cuatro ciclos de lectura de memoria tienen lugar y son ejecutados en Modo Sistema. En el primer ciclo se lee de la localidad 0002H la Palbra que será cargada en el Registro de Banderas y Control, el siguiente ciclo lee de la localidad 0004H el Número de Segmento de 7 bits que será cargado en el Contador de Programa, el siguiente ciclo lee de la localidad 0006H el Desplazamiento de 16 bits que será cargado en el Contador de Programa y el último ciclo lee la primera instrucción del Programa.

1.2.5 MANEJO DE LOS CANALES.- Un nivel bajo en la línea Solicitud de Canales (BUSREQ*) del procesador indica que otro dispositivo inteligente está solicitando los Canales de Datos/Direcciones y Control. La entrada asíncrona BUSREQ* es sincronizada al principio de cualquier ciclo de máquina, una señal síncrona interna es generada la cuál (después de la terminación del ciclo de máquina en ejecución) causa que la línea Reconocimiento de Solicitud de Canal(BUSACK*) vaya a un nivel bajo y que todas las salidas de los Canales queden en estado de alta impedancia. El dispositivo solicitante tiene entonces el control de los Canales.

Cuando la línea BUSREQ* es liberada, ésta es sincronizada con el frente de subida del reloj y la salida BUSACK* va a un nivel alto un período de reloj después, indicando que el procesador tomará de nuevo el control de los Canales.

1.2.7 ARQUITECTURA DE PROCESAMIENTO EXTENDIDO.- La capacidad del procesador puede ser incrementada a través de la Arquitectura de Procesamiento Extendido (EPA). De una manera simple, la Arquitectura de Procesamiento Extendido permite al procesador acomodar hasta cuatro Unidades de Procesamiento Extendido adicionales (EPUs), las cuales realizan funciones especializadas en paralelo con el flujo principal de ejecución de instrucciones del procesador.

El procesador es responsable de instruir al EPU y de enviarle operandos y datos, éste reconoce las instrucciones dirigidas a él y las ejecuta, usando datos proporcionados con la instrucción y/o dentro de sus registros internos.

Además de las capacidades implementadas en la lógica interna para la Arquitectura de Procesamiento Extendido, hay un mecanismo de trampa de instrucciones extendidas para permitir la simulación mediante un programa de las funciones de las EPUs. El bit de control en el Registro de Banderas y Control (FCW) del procesador indica si hay o no EPUs presentes en el sistema. Si no las hay, cuando una instrucción extendida es detectada, el procesador la "atrapa" de manera que el "manejador de Trampas" puede simular la función deseada de la EPU.

ESTA CARACTERISTICA ES MUY UTIL PARA LA ADICION POSTERIOR DE EPUs: inicialmente, la función extendida es ejecutada como una rutina de trampa; cuando la EPU es conectada, la rutina de proceso de trampa es eliminada y se pone el bit de control EPA. La programación de aplicación es transparente al cambio.

La Arquitectura de Procesamiento Extendido también ofrece protección contra el traslape de las instrucciones extendidas. Cada EPU se conecta al Z8000 a través de la línea STOP* de manera que si una EPU es solicitada para ejecutar una segunda instrucción extendida antes de haber terminado la primera, puede poner al CPU en el Estado STOP/REFRESH hasta que la ejecución de la primera instrucción extendida termine.

La ejecución de instrucciones del CPU y de la EPU es como sigue: el CPU efectúa una búsqueda de instrucción y determina si se trata de un comando de CPU o uno de EPU. Mientras tanto, la EPU monitorea el Canal de Datos en búsqueda de sus instrucciones. Si el CPU encuentra un comando para la EPU, verifica si hay una EPU presente en el sistema; si no, la EPU puede ser simulada por una rutina de trampa de instrucción extendida. Si una EPU está presente, los Datos/Direcciones necesarios son puestos en el Canal, si la EPU está libre cuando la instrucción para ella aparece, la instrucción extendida es ejecutada. Si la EPU está

procesando una instrucción previa, activa la línea STOP* del CPU para detenerlo fuera del Canal hasta el final de la ejecución de la instrucción. Después de que la instrucción ha sido ejecutada, la EPU desactiva la línea STOP* y las transacciones con el CPU continuan.

1.2.8 SOPORTE PARA MULTIPROCESO.- Un par de líneas del CPU son usadas junto con algunas instrucciones para coordinar procesadores múltiples. La línea MULTI-MICRO OUT (MO*) envía una solicitud del recurso, mientras la línea MULTI-MICRO IN (MI*) es usada para reconocer el estado del recurso. Con esto, cualquier CPU en un sistema con Multi-Microprocesadores puede excluir a todos los demás CPUs asíncronos de un recurso crítico compartido.

Los sistemas con Multi-Procesadores son soportados por las instrucciones Solicitud Multi-Micro (MREQ),Prueba Entrada Multi-Micro (MBIT), Enciende Salida Multi-Micro (MSET) y Apaga Salida Multi-Micro (MRES).

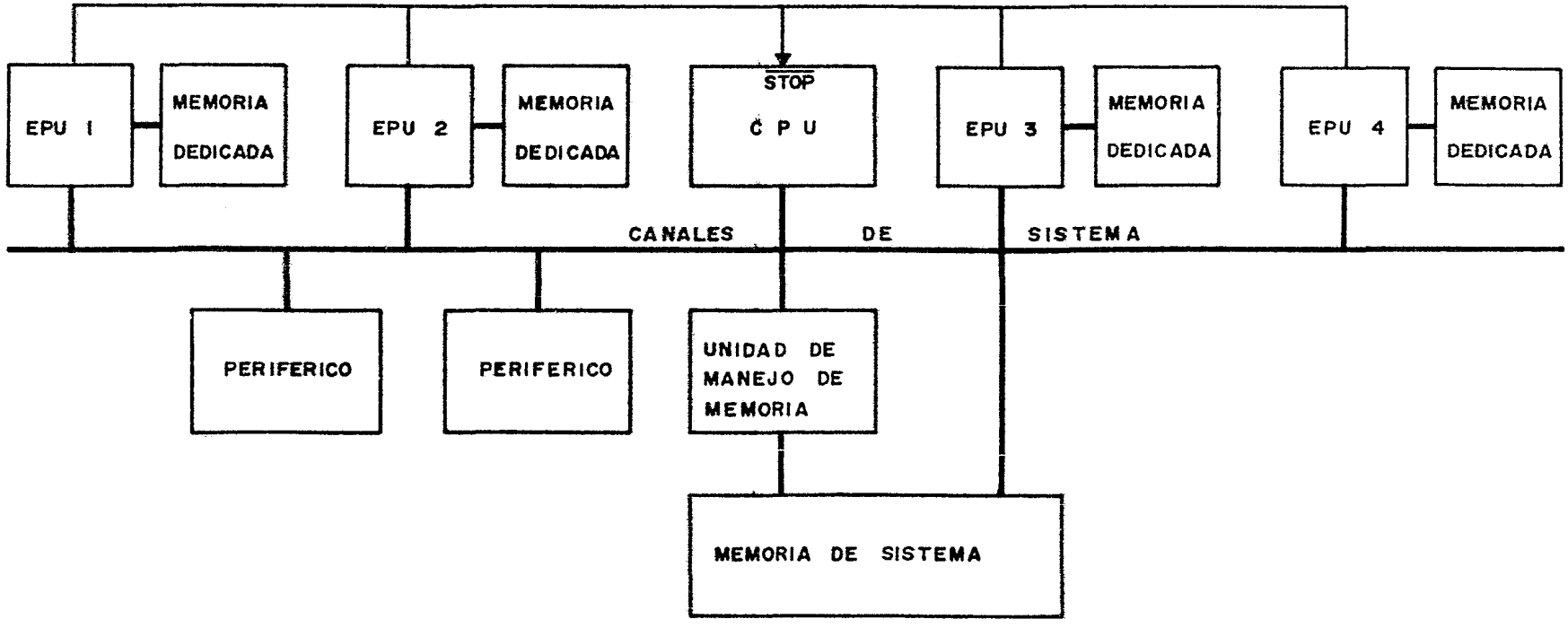


FIG 19 ARQUITECTURA DE PROCESAMIENTO EXTENDIDO

1.3 ARQUITECTURA DEL PROCESADOR 68000

1.3.1 INTRODUCCION.- Este procesador microprogramado tiene una arquitectura interna de 32 bits. es decir. todos los Registros son de 32 bits de ancho al igual que el Canal Interno de Datos. En el exterior. los Datos e Instrucciones son manejados como palabras de 16 bits y las Direcciones son manejadas en 24 bits a través de un canal separado del de Datos. Los dispositivos de E/S son manejados como parte de la memoria de 16 Mbytes. Además. tiene capacidad para soportar siete niveles de interrupciones y varias trampas internas.

1.3.2 MODOS DE OPERACION.- El 68000 tiene dos modos de operación que son el Modo SUPERVISOR y el Modo USUARIO. La diferencia entre estos es que en Modo Supervisor se puede ejecutar el conjunto de instrucciones completo mientras que en el Modo Usuario sólo se pueden ejecutar las instrucciones no privilegiadas.

MODO SUPERVISOR.- Este es el modo con prioridad dentro del procesador. Para la ejecución de instrucciones. el Modo Supervisor es determinado por el estado del bit S en el Registro de status. si este bit está encendido. el procesador se encuentra en este modo y es capaz de ejecutar todas las instrucciones. Todo el procesamiento de excepciones es ejecutado en Modo Supervisor sin importar el estado del bit S.

MODO USUARIO.- El procesador se encuentra en este Modo si el bit S en el Registro de status está apagado. En este Estado no se pueden ejecutar instrucciones que tienen un efecto importante sobre el estado del sistema como son las instrucciones STOP y RESET. También para asegurar que un programa de usuario no pueda entrar en modo supervisor de manera no controlada, las instrucciones que modifican el Registro de status completo son privilegiadas.

Una vez que el procesador se encuentra en Modo Usuario y ejecutando instrucciones. sólo el procesamiento de excepciones puede causar un cambio en el estado de privilegio.

1.3.3 ARQUITECTURA INTERNA.- Este procesador está basado en una Unidad de Ejecución Controlada por Microprograma. El área de almacenamiento del Control es minimizada a través del uso de una Estructura de Control de Dos Niveles. En el Nivel Uno las instrucciones de máquina son producidas por secuencias de microinstrucciones en el área de Almacenamiento de Microcontrol. Actualmente. las microinstrucciones son apuntadores para el Area de Almacenamiento de Nanoinstrucciones en el Nivel Dos. esta

Area contiene un conjunto de Palabras de Control de Máquina no duplicadas ordenadas arbitrariamente las cuales controlan a la Unidad de Ejecución (18).

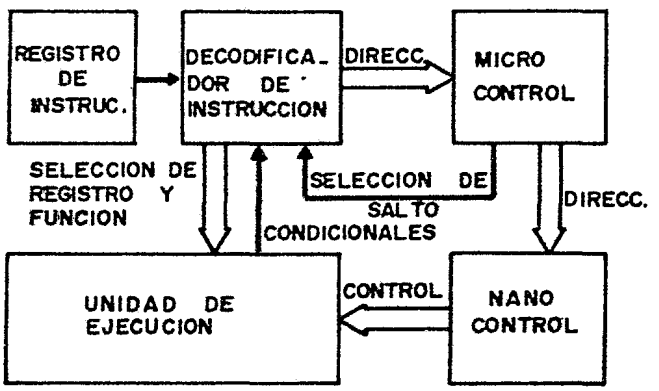


FIG 20 ESTRUCTURA INTERNA

1.3.3 DESCRIPCION DE LOS REGISTROS.- Este Procesador cuenta con 15 Registros de 32 Bits de Propósito General de los cuales ocho son para Datos y siete para Direcciones. además cuenta con un Contador de Programa de 32 Bits. un Registro de status de 16 Bits y dos Apuntadores de Pila de 32 Bits cada uno.

REGISTROS DE DATOS (D0 a D7).-Este grupo está formado por 8 registros de 32 bits que pueden ser usados como acumuladores o como registros para almacenamiento de propósito general. pueden manejar datos de 1.8.16 o 32 bits y pueden ser usados como fuente o como destino de cualquier operación sin ninguna distinción.

REGISTROS DE DIRECCION (A0 a A6).- Este grupo está formado por siete Registros de 32 bits. pueden ser usados como apuntadores de memoria o como Registros Indice. Estos Registros sólo soportan datos de 16 o 32 bits y pueden efectuarse operaciones aritméticas y lógicas sobre ellos para el cálculo de una dirección.

Cuando uno de los Registros de Dirección es usado como operando fuente. se puede utilizar sólo la parte menos

significativa del Registro (operaciones en palabras) o el Registro completo (operaciones en palabras largas). Si estos Registros son usados como operando destino, el Registro completo es afectado sin importar el tamaño del operando. Si el operando es una palabra, todos los operandos son extendidos en signo a 32 bits antes de efectuar la operación.

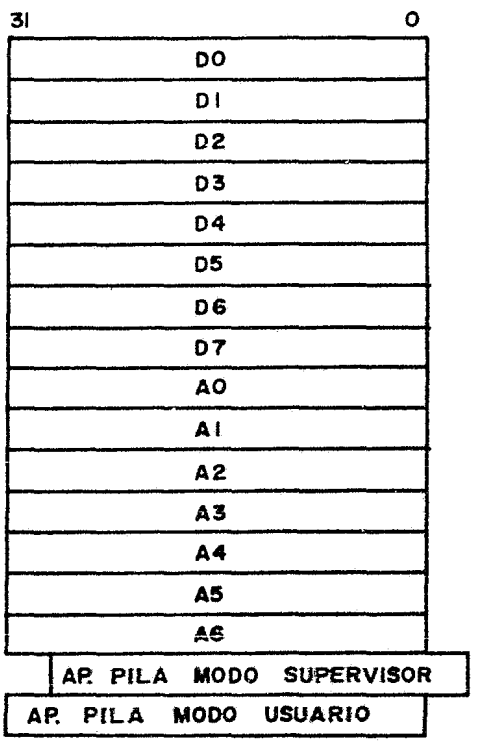


FIG 21 REGISTROS DE PROPOSITO GENERAL

CONTADOR DE PROGRAMA (PC).-Este es un registro de 32 bits el cual es usado para direccionar instrucciones en memoria, aunque este registro es de 32 bits al igual que los Registros de Direcciones, actualmente sólo se usan 24 bits para el direccionamiento de memoria (16 Mbytes).

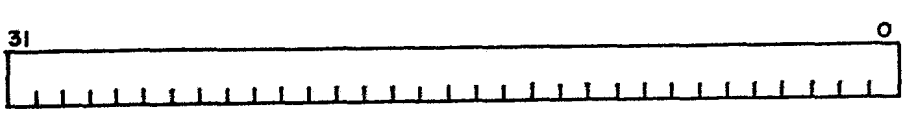


FIG 22 CONTADOR DE PROGRAMA

REGISTRO DE STATUS (SR).- Este es un Registro de 16 bits el cual está dividido en 2 bytes, el byte de sistema y el byte de usuario, contiene las Banderas de status del Procesador además de la Máscara de Prioridad de Interrupciones y los Bits de Control.

Las banderas que contiene son las siguientes:

ACARREO.-(bit 0)

SOBREFLUJO.-(bit 1)

CERO.-(bit 2)

NEGATIVO.-(bit 3)

EXTENSION.-(bit 4) Esta bandera funciona como un bit de acarreo para operaciones en precisión múltiple, es afectada por las operaciones de suma, resta, negación, corrimiento y rotación durante las cuales recibe el estado del bit de acarreo.

MASCARA DE INTERRUPCION.-(bits 8, 9 y 10) Estos bits contienen una máscara la cual determina el nivel de solicitud de interrupción que será atendido por el procesador. Esta máscara puede ser utilizada para establecer uno de ocho niveles de interrupción y causa que todas las interrupciones con un nivel igual o menor sean ignoradas por el procesador. La operación de esta máscara se describe con mayor detalle en la sección sobre manejo de interrupciones.

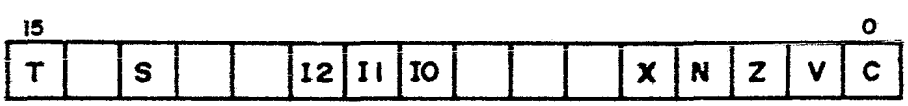


FIG 23 REGISTRO DE BANDERAS Y CONTROL

ESTADO SUPERVISOR (S).-(bit 13) Esta bandera indica que el procesador está operando en Modo Supervisor si está puesta en uno o que se encuentra en Modo Usuario si está puesta en cero.

MODO DE TRAZO (T).-(bit 15) Esta bandera controla la circuitería interna para ayudar en la depuración de programas. Cuando esta bandera está encendida, el procesador ejecuta instrucciones paso a paso, asimismo causa que después de la ejecución de cada instrucción el Procesador pase a Modo Supervisor y mediante un vector de trampa apunte a una rutina de depuración que deberá ser escrita por el usuario.

PILAS DEL SISTEMA (STACKS).-El Registro de Direcciones A7 es el Apuntador de Pila de Sistema (SP). El Apuntador de Pila del Sistema puede ser el Apuntador de Pila de Modo Supervisor (SSP) o el Apuntador de Pila de Modo Usuario (USP), dependiendo del estado del Bit S en el Registro de Estado. Si el Bit S indica Modo Supervisor, el SSP es el Apuntador de Pila del Sistema activo y el USP no puede ser referido. Si el Bit S indica Modo Usuario, entonces el USP será el Apuntador de Pila de Sistema activo y el SSP no podra ser referido. Cada Pila de Sistema se llena desde localidades de memoria altas hacia las bajas.

1.3.4 ORGANIZACION DE MEMORIA.- La memoria del 68000 está organizada, al igual que los Registros en Bytes, palabras y palabras largas. Cada byte tiene una dirección que es un número de 24 bits lo cual proporciona una capacidad máxima de direccionamiento de 16 Mbytes. Nótese que las direcciones de bytes pueden tener cualquier valor y las direcciones de palabra y palabra larga deberán ser siempre valores pares.

En el caso de las palabras y de las palabras largas, el byte más significativo de la palabra siempre estará contenido en una dirección par y el byte menos significativo en una dirección impar. Las instrucciones y las cadenas de datos siempre serán direccionadas en localidades pares.

FIG 24A ORGANIZACION DE MEMORIA

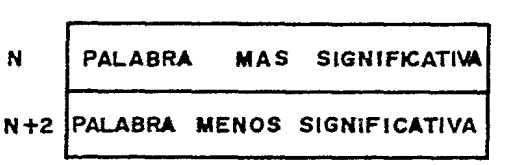


FIG 24B ORGANIZACION DE MEMORIA (PALABRA LARGA)

El direccionamiento de localidades de memoria se efectúa a través de 23 líneas de direcciones y dos líneas llamadas Habilitación de Dato Alto (UDS) y Habilitación de Dato Bajo (LDS) las cuales son usadas junto con la línea de Lectura/Escritura* (R/W*) para seleccionar el Byte Alto, el Byte Bajo o la Palabra completa para lectura o escritura segun la siguiente tabla:

UDS*	LDS*	R/W*	D8-D15	D0-D7
1	1	---	DATO INVALIDO	DATO INVALIDO
0	0	1	DATO 8-15	DATO 0-7
1	0	1	DATO INVALIDO	DATO 0-7
0	1	1	DATO 8-15	DATO INVALIDO
0	0	0	DATO 8-15	DATO 0-7
1	0	0	DATO INVALIDO	DATO DE 0-7
0	1	0	DATO 8-15	DATO INVALIDO

TABLA DE SELECCION DE BYTES

Una Dirección válida de Memoria es indicada a través de la línea Habilitación de Dirección (AS*).

El espacio de direccionamiento de memoria puede ser extendido si se decodifican con ayuda de lógica externa las líneas de Código de Función (FC0, FC1 y FC2) del Procesador con esto se pueden tener cuatro segmentos de 16 Mbytes separados para Código de Modo Supervisor, Datos de Modo Supervisor, Código de Modo Usuario, y Datos de Modo Usuario teniendo así un total de 64 Mbytes de memoria. Los códigos de función proporcionados por el Procesador son los indicados en la siguiente tabla:

FC2	FC1	FC0	TIPO DE CICLO
0	0	0	RESERVADO
0	0	1	DATOS.DE USUARIO
0	1	0	PROGRAMA DE USUARIO
0	1	1	RESERVADO
1	0	0	RESERVADO
1	0	1	DATOS DE SUPERVISOR
1	1	0	PROGRAMA DE SUPERVISOR
1	1	1	RECONOCIMIENTO DE INTERRUPCION

TABLA CODIGOS DE FUNCION

1.3.5 MODOS DE PROCESAMIENTO DEL 68000.-El procesador siempre estará en uno de tres modos de procesamiento: NORMAL, DE EXCEPCION o DETENIDO.

ESTADO NORMAL.- Este estado está asociado con la ejecución de instrucciones, las referencias a memoria son para buscar instrucciones y datos o para almacenar resultados. Un caso especial es el Estado de Paro al cual entra el procesador cuando se ejecuta la instrucción STOP. En este estado no se hace ninguna referencia a memoria.

ESTADO DE PROCESAMIENTO DE EXCEPCIONES.- Este estado está relacionado con las Interrupciones, Instrucciones de Trampa, Trazado y otras condiciones excepcionales. Las excepciones pueden ser generadas internamente por una instrucción o por una condición no usual durante la ejecución de una instrucción o de manera externa causadas por una interrupción, un error de Canal o por una restauración.

VECTORES DE EXCEPCION.- Los vectores de excepción son localidades de memoria en las cuales el procesador busca la dirección de la rutina que maneja esa excepción. Todos los vectores de excepción son de dos palabras de largo, exceptuando al vector de Restauración el cual es de cuatro palabras. Todos los vectores de excepción están localizados en el espacio de datos de Modo Supervisor con excepción del Vector de Restauración el cual se encuentra en el espacio de programa de Modo Supervisor.

El número de vector de 8 bits es multiplicado por cuatro para obtener la dirección de un vector de excepción. Los números de vector son generados interna o externamente dependiendo de la causa de la excepción. En el caso de las interrupciones, durante el ciclo de reconocimiento, un periférico proporciona un número de vector de 8 bits en las líneas de datos del procesador D0- D7. El formato del vector de excepción se muestra en la figura 25.

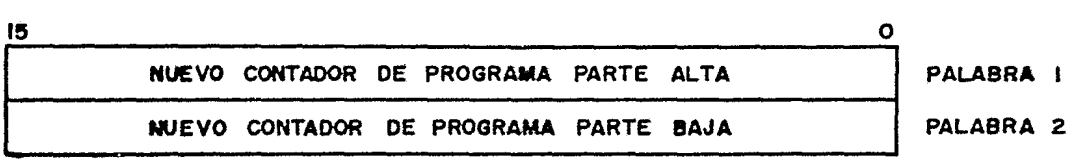


FIG 25 FORMATO DEL VECTOR DE EXCEPCION

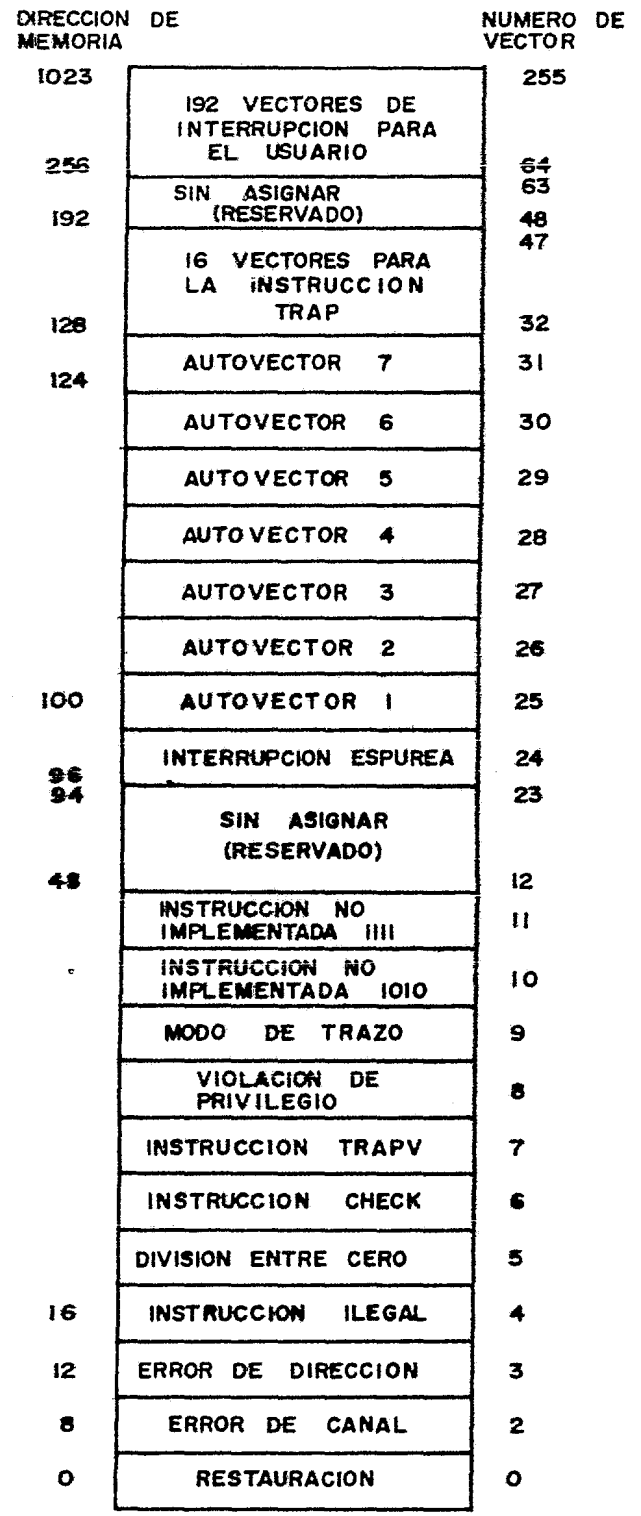


FIG 26 VECTORES DE EXCEPCION

El mapa de vectores de excepción se muestra en la figura anterior. El mapa de memoria tiene un largo de 512 palabras. Empieza en la dirección 0 y continua hasta la 1023. Esto nos proporciona 255 vectores únicos algunos de los cuales están reservados para trampas y otras funciones del sistema. De los 255, hay 192 reservados para los vectores de interrupción del usuario.

TIPOS DE EXCEPCIONES.- Como se dijo anteriormente, las excepciones pueden ser generadas por causas internas o externas.

Las excepciones generadas externamente son las solicitudes de Interrupción, Error de Canal y Restauración. Las Interrupciones son solicitudes de atención de dispositivos periféricos para que el procesador realice un procedimiento mientras que el Error de Canal y la entrada de Restauración son usadas para control de accesos e inicialización del procesador.

Las excepciones generadas internamente provienen de instrucciones, de errores de direccionamiento y del Modo de Trazo. HAY INSTRUCCIONES QUE PUEDEN GENERAR TRAMPAS COMO PARTE DE SU EJECUCION. Además, las instrucciones ilegales, la búsqueda de palabras en direcciones impares y los intentos de ejecución de instrucciones privilegiadas en Modo Normal causan también excepciones. El Modo de Trazo está definido como una Interrupción Interna de muy alta prioridad generada después de la ejecución de cada instrucción.

La prioridad de Interrupciones y Trampas en orden descendente es la siguiente:

PRIORIDAD	INTERRUPCION
1	RESTAURACION EXTERNA
2	ERROR DE CANAL
3	ERROR DE DIRECCION
4	EJECUCION PASO A PASO
5	INTERRUPCIONES EXTERNAS
6	INSTRUCCION ILEGAL
7	INSTRUCCION PRIVILEGIADA
8	INSTRUCCION TRAP
9	INSTRUCCION TRAPV
10	INSTRUCCION CHK
11	ERROR DE DIVISION

TABLA DE PRIORIDADES DE INTERRUPCION

SECUENCIA DEL PROCESAMIENTO DE EXCEPCIONES.- El procesamiento de excepciones ocurre en cuatro pasos: en el primer paso, se hace una copia interna del Registro de status, después el bit S es prendido colocando al procesador en Modo Supervisor, asimismo el Bit T es puesto en cero con lo cual se habilita a la rutina de atención para ejecutar sin trazado. Para las excepciones de Restauración e Interrupción también se actualiza la Máscara de Interrupciones.

En el segundo paso, se determina el número de vector para esa excepción. Para las interrupciones, el número de vector es obtenido mediante una búsqueda de instrucción del procesador, clasificada como un reconocimiento de interrupción. PARA TODAS LAS DEMAS EXCEPCIONES, LA LOGICA INTERNA PROPORCIONA EL NUMERO DE VECTOR. Este número de vector es usado para generar la dirección del vector de excepción.

En el tercer paso, se salva el status actual del procesador, excepto en el caso de una Excepción de Restauración. El valor actual del Contador de Programa junto con la copia del Registro de status son almacenados en la Pila de Modo Supervisor. Generalmente, el valor del Contador de Programa que se almacena es el apuntador a la siguiente instrucción del programa aunque en el caso de un Error de Canal y de un Error de Dirección el valor es impredecible. Además, para el caso de estas dos excepciones se guarda también información adicional sobre el contexto del procesador.

En el cuarto paso, el cual es el mismo para todas las excepciones, se busca el nuevo valor del Contador de Programa apuntado por el Vector de Excepción y el procesador reasume la ejecución de instrucciones.

En los siguientes párrafos se hace una descripción más detallada del procesamiento de cada tipo de excepción:

EXCEPCION DE RESTAURACION.- Esta excepción tiene el más alto nivel de prioridad. El procesamiento de la señal RESET está diseñado para la inicialización del sistema y para la recuperación de fallas catastróficas como por ejemplo un error de programación. El procesador es forzado al modo supervisor y la bandera de trazado (T) es apagada, la máscara de interrupción es puesta en nivel 7. El número de vector es generado internamente y apunta al vector de excepción de restauración localizado en la localidad 0 del espacio de programa de Modo Supervisor. La dirección localizada en las dos primeras palabras del vector de excepción es cargada como el Apuntador de Pila inicial para Modo Supervisor y la dirección contenida en las dos últimas

localidades del vector de excepción es cargada como el Contador de Programa inicial. Finalmente, la ejecución del programa se inicia en la dirección cargada en el Contador de Programa. La instrucción RESET no causa que el Vector de Excepción sea cargado aunque si enciende la línea RESET para restaurar dispositivos externos.

EXCEPCION DE INTERRUPCION.- Se proporcionan siete niveles de interrupción en el 68000. Los dispositivos pueden ser encadenados externamente dentro de cada uno de los niveles de prioridad permitiendo que un número ilimitado de dispositivos periféricos interrumpa al procesador. Las prioridades están numeradas de 1 a 7 siendo el nivel 7 el de mayor prioridad. Como se mencionó anteriormente, el Registro de status contiene la Máscara de Prioridad de Interrupción. El procesador inhibirá todas las interrupciones que tengan un nivel igual o menor al colocado en esta Máscara.

Una solicitud de interrupción es hecha al procesador a través de la codificación del nivel de Interrupción en las líneas de Solicitud de Interrupción (IPL0*, IPL1* e IPL2*). Un cero indica que no hay solicitud. Las solicitudes de Interrupción que llegan al procesador no causan la ejecución inmediata de una excepción pero si son almacenadas como pendientes. Las interrupciones pendientes son detectadas entre la ejecución de las instrucciones y si su nivel es menor o igual que la prioridad del procesador en ese momento, la ejecución normal de instrucciones continua y la atención de la interrupción es pospuesta.

Si la prioridad de la Interrupción solicitada es mayor que la indicada en la máscara de prioridad, entonces se inicia la secuencia de proceso de excepción como fue descrita anteriormente, es decir, primero se hace una copia del Registro de status, segundo, el procesador obtiene el vector en la parte baja del Canal, tercero, el Contador de Programa es salvado junto con la copia del Registro de status en la Pila de Modo Supervisor y cuarto, se inicia la ejecución de la rutina. El nivel de interrupción siete es un caso especial, éste nivel de interrupción no puede ser inhibido por la Máscara de Prioridad de Interrupción con lo cual se proporciona la capacidad de Interrupción No Mascarable. Una interrupción es generada cada vez que el nivel de interrupción cambia de un nivel inferior a nivel siete.

EXCEPCION DE INTERRUPCION NO INICIALIZADA.- Un dispositivo que interrumpe afirma la línea VPA* y proporciona un vector de interrupción durante el ciclo de reconocimiento de interrupción. Si su Registro de Vector de Interrupción no ha sido inicializado, entonces responderá con el número de vector 15 el cuál es el número de vector no

inicializado. Con esto se tiene un medio uniforme de recuperación de errores de programación.

EXCEPCION DE INTERRUPCION ESPUREA (SPURIOUS).- Si durante un ciclo de reconocimiento de interrupción el dispositivo periférico no responde afirmando las líneas DTACK* o VPA* la línea de Error de Canal deberá ser afirmada para terminar el ciclo de adquisición del vector. El procesador separa el proceso de este error cargando el Vector de Interrupción Espurea (Spurious) en lugar del Vector de Error de Canal y la rutina de excepción es ejecutada normalmente.

TRAMPAS DE INSTRUCCION.- LAS TRAMPAS SON EXCEPCIONES CAUSADAS POR INSTRUCCIONES. Surgen ya sea durante el reconocimiento de condiciones anormales por parte del procesador durante la ejecución de instrucciones o por el uso de instrucciones que normalmente generan trampas.

Algunas instrucciones son usadas específicamente para generar trampas. La instrucción TRAP causa siempre una excepción, y es útil para la implementación de llamadas de sistema para programas de usuario. Las instrucciones TRAPV y CHK causan trampas si el programa del usuario detecta un error durante la ejecución, el cual puede ser un sobrepaso aritmético o un apuntador fuera de rango. También un intento de división entre cero causa una excepción.

EXCEPCIONES DE INSTRUCCIONES ILEGALES Y NO IMPLEMENTADAS.- Una instrucción ilegal es aquella que tiene un patrón de bits que no corresponde al patrón de bits de ninguna instrucción legal. Si durante la ejecución de un programa se encuentra una instrucción de este tipo se genera una excepción.

Los patrones de instrucción con los bits 15 al 12 iguales a 1010 o 1111 son distinguidos como instrucciones no implementadas y hay vectores de excepción para estos patrones para permitir la emulación de la instrucción por programa.

EXCEPCIONES DE VIOLACION AL PRIVILEGIO.- Cualquier intento de ejecutar una instrucción privilegiada en Modo de Usuario causará la generación de una excepción.

EXCEPCION DE MODO DE TRAZO.- Para ayudar en el desarrollo de programas, el 68000 incluye la característica de permitir la ejecución de instrucciones paso a paso. Después de la ejecución de cada instrucción una excepción es forzada, permitiendo que un programa depurador monitoree la ejecución del programa bajo prueba.

Esta característica hace uso de la Bandera de Trazo (T) en el Registro de status, si la bandera está encendida el Modo de Trazo está habilitado y una excepción de trazo será generada después de la ejecución de cada instrucción. Si la instrucción es ilegal o privilegiada, o hay una Interrupción, un Error en el Canal o de Dirección, la Excepción de Trazo no es generada.

EXCEPCION DE ERROR EN EL CANAL DE DATOS.- Las excepciones de Error en el Canal de Datos ocurren cuando la lógica externa solicita que un Error de Canal sea procesado por una excepción. El ciclo ejecutandose es abortado. Si el procesador estaba ejecutando instrucciones o procesando una excepción el proceso es terminado e inmediatamente comienza la ejecución de la excepción de Error en el Canal.

La ejecución de una excepción de Error en el Canal sigue los pasos normales y además se salva información adicional sobre el status del Procesador en la Pila de Modo Supervisor. El procesador salva también una copia de la primera palabra de la instrucción que era procesada, y la dirección que fue accesada durante el ciclo abortado. También se salva información específica acerca del acceso como por ejemplo si era una lectura o una escritura, si el procesador estaba ejecutando una instrucción o no, y además la clasificación mostrada en las líneas de código de función al ocurrir el Error en el Canal.

SI OCURRE UN ERROR EN EL CANAL DURANTE LA EJECUCION DE LA RUTINA DE EXCEPCION DE ERROR DE CANAL, ERROR DE DIRECCION O UNA RESTAURACION EL PROCESADOR SE DETIENE. Esto simplifica la detección de fallas catastróficas en el sistema ya que el procesador se retira en lugar de destruir el contenido de la memoria. Sólo la señal RESET puede sacar al procesador del Estado de Detención (HALT).

EXCEPCION DE ERROR DE DIRECCION.-El Error de Dirección ocurre cuando el procesador intenta accesar una palabra, una palabra larga o una instrucción en una dirección impar. El efecto es como si se generara un Error de Canal interno, de manera que el ciclo de Canal es abortado y el procesador detiene la ejecución de lo que está haciendo y comienza inmediatamente la ejecución de la Excepción de Error de Dirección. Después de que la ejecución de la excepción comienza, la secuencia es la misma que para el Error en el Canal incluyendo la información almacenada en la Pila de Modo Supervisor. IGUALMENTE, SI OCURRE UN ERROR DE DIRECCION DURANTE EL PROCESO DE LA EXCEPCION PARA UN ERROR DE CANAL, DE DIRECCIONES O UNA RESTAURACION EL PROCESADOR ES DETENIDO.

1.3.5 MANEJO DE LOS CANALES.-El Procesador cuenta con 3 líneas que permiten que otros dispositivos inteligentes dentro del sistema hagan uso de los Canales del procesador, estas líneas son: Solicitud de Canales (BR*), Otorgamiento de Canales (BG*) y Reconocimiento de Otorgamiento de Canales (BGACK*). Cualquier dispositivo que requiera hacer uso de los canales deberá afirmar la línea de Solicitud de Canales (BR*) y esperar a que el Procesador responda con la línea Otorgamiento de Canales (BG*), entonces el dispositivo deberá afirmar la línea Reconocimiento de Otorgamiento de los Canales.

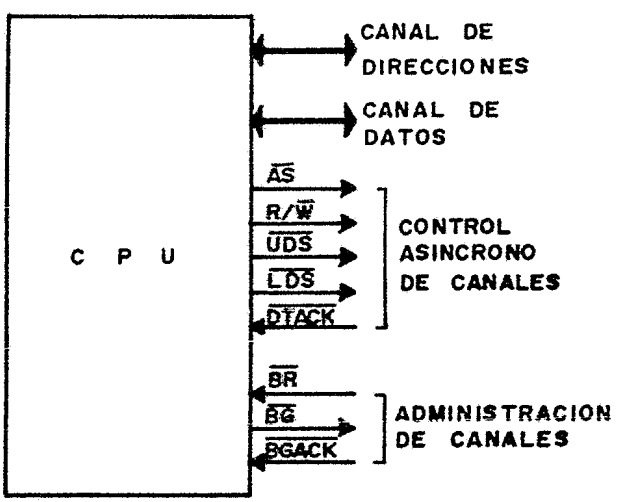


FIG 27 MANEJO DE CANALES

1.4 COMPARACION DE ARQUITECTURAS

1.4.1 INTRODUCCION.- La comparación de Arquitecturas será realizada en base a los siguientes criterios:

1. Arquitectura Interna
2. Modos de Operación.
3. Organización de Registros.
4. Capacidad de Manejo y Organización de la Memoria.
5. Manejo de Interrupciones y Trampas.
6. Circutería Interna para Soporte de Sistemas de Multiproceso (varios microprocesadores).

1.4.2 COMPARACION DE ARQUITECTURAS INTERNAS.- En este aspecto los tres procesadores tienen arquitecturas internas completamente diferentes. El 8086 es un procesador "híbrido" ya que la Unidad de Interfase es del tipo microalambrado (lógica aleatoria) y la Unidad de Ejecución es del Tipo microprogramado. En el Z8000 la arquitectura es de lógica alambrada y en el 68000 toda su arquitectura es del tipo microprogramado. Cada uno de estos tipos de arquitectura tiene sus ventajas y desventajas, en el caso de la lógica alambrada la ventaja es que es muy rápida y la desventaja es que el diseño es complicado y difícil de modificar en el caso de que se requiera alguna modificación o mejora al conjunto de instrucciones por ejemplo. La Arquitectura Microprogramada tiene la ventaja de que es más fácil de diseñar y modificar pero la desventaja es que es un poco más lenta. El 8086 tiene la característica de que sus Unidades de Interfase y Ejecución son completamente independientes con lo cual se permite que las operaciones de ejecución y búsqueda de instrucciones sean realizadas en forma simultánea.

1.4.3 COMPARACION DE MODOS DE OPERACION.- En el caso del Z8000 y del 68000 no hay diferencia en cuanto a los modos de operación ya que ambos pueden operar en Modo Sistema o Modo Normal. Los dos manejan Instrucciones Privilegiadas y realizan el proceso de Interrupciones y Trampas en Modo Sistema. El 8086 sólo tiene un modo de operación interno por lo que no se tienen instrucciones privilegiadas. En este aspecto el 68000 y el Z8000 tienen ventaja sobre el 8086 ya que la distinción de modos permite una mejor organización de sistemas operativos así como mayor facilidad para la protección de éstos. El 8086 cuenta con dos modos de operación externos los cuales permiten que la configuración

de sus líneas de control cambie, esto es Modo Máximo y Modo Mínimo. En el Modo Mínimo el procesador no cuenta con líneas de status para indicar el tipo de operación que está realizando; en Modo Máximo estas líneas son proporcionadas de manera multiplexada y se decodifican a través de un Controlador de Canal de Sistema con el objeto de extender el conjunto de líneas de control a todo el sistema indicando el tipo de operación del procesador.

Evaluación:	8086 R
	Z8000 B
	68000 E

1.4.4 COMPARACION DE LA ORGANIZACION DE REGISTROS.- En el aspecto de versatilidad, el conjunto de Registros del Z8000 es el que posee las mejores características con respecto al 68000 y al 8086. En este procesador el concepto de Registros de Propósito General ha sido cubierto casi al máximo, debido a que todos los Registros de 16 Bits pueden ser usados como Acumuladores y todos menos uno (R0), pueden usarse como Apuntadores de Pila o como Indices a Memoria. Esto no es posible en los otros dos procesadores porque su conjunto de Registros está dividido por grupos, dependiendo de la operación que se haga, él o los registros tienen restricciones. Además, los Registros del Z8000 pueden ser agrupados por pares o por cuartetas, lo cual le permite realizar operaciones de hasta 64 bits. Esta característica no la posee ninguno de los otros dos procesadores. El 68000 tiene un poco menos versatilidad ya que sus Registros están divididos en dos Grupos: el de Datos y el de Direcciones. Para el propósito de esta comparación sólo los Registros de Datos pueden ser considerados como Registros de Propósito General ya que son los únicos que sirven para almacenamiento y operaciones en general. En el caso del 8086 sólo se pueden considerar cuatro Registros de Propósito General que son los del grupo HL ya que son los únicos que se pueden utilizar libremente para manejo de datos y operaciones en general. Los demás registros únicamente son usados como apuntadores de memoria.

En lo que se refiere a tamaño de los Registros el 68000 posee los Registros de mayor capacidad ya que toda la arquitectura interna es de 32 Bits. En el Z8000 y en el 8086 se tienen Registros y arquitectura interna de 16 Bits. Los Registros Contador de Programa, Apuntador de Pila y de status de los tres procesadores tienen características equivalentes en su manejo.

Evaluación:	8086 R
	Z8000 E
	68000 B

1.4.5 COMPARACION DE LA CAPACIDAD Y ORGANIZACION DE MEMORIA.- El 68000 es capaz de direccionar directamente 16 Mbytes, el Z8000 8 Mbytes y el 8086 1 Mbyte, en el caso del 68000 y del Z8000 la capacidad de direccionamiento puede ser extendida hasta 64 y 48 Mbytes respectivamente, en el 8086 la capacidad de direccionamiento no puede ser extendida.

La organización de memoria es Lineal en el caso de 68000 y Segmentada en el Z8000 y 8086. Para la expansión de memoria al máximo en el 68000 es necesario el uso de 4 Unidades de Manejo de Memoria y de 12 Unidades de Manejo de Memoria en el Z8000. La organización de memoria es la misma en los tres procesadores ya que pueden direccionar Palabras y Bytes directamente.

Evaluación:	8086 R
	Z8000 B
	68000 E

1.4.6 COMPARACION DE LA ESTRUCTURA DE INTERRUPCIONES Y TRAMPAS.- La estructura de Interrupciones más sofisticada es la soportada por el 68000, ya que maneja 7 niveles de Interrupción, los cuales proporcionan un sistema de privilegio en la solicitud de interrupciones eliminando así la necesidad de encadenar dispositivos externamente para asignarles diferente prioridad. Cuenta con 192 vectores de interrupción para el usuario, además tiene 7 autovectores para manejo de periféricos de la familia anterior 6800. Le sigue en complejidad el Z8000 que maneja Interrupciones no mascarables e Interrupciones con o sin vector teniendo hasta 256 posibles vectores. La estructura de interupciones más sencilla es la del 8086 que cuenta con una línea para Interrupción no mascarable, y una para Interrupción mascarable con 256 posibles vectores. Los tres procesadores manejan las Interrupciones con vector a través de una tabla de direcciones en memoria la cual es usada de una manera similar.

Evaluación:	8086 R
	Z8000 B
	68000 E

1.4.7 CIRCUITERIA INTERNA PARA SOPORTE DE SISTEMAS DE MULTIPROCESO.- La estructura de soporte de Multiproceso y Aquitectura de Procesamiento Extendido más sofisticada es la del Z8000, el cual además de poder manejar hasta cuatro procesadores de propósito especial es capaz de funcionar con una Arquitectura en la cual existan otros procesadores del mismo tipo. En orden descendente le sigue el 8086 el cual posee dos modos de trabajo que son utilizados según la configuración del sistema, en Modo Máximo es capaz de trabajar con procesadores de propósito especial y con otros

procesadores. El 68000 no tiene ninguna de estas características, ya que únicamente cuenta con las líneas de Solicitud y Otorgamiento de Canales que permiten a otros dispositivos inteligentes hacer uso de los Canales del Sistema. En el caso de los otros dos Procesadores además de estas líneas se tienen otras dedicadas al control de recursos compartidos como las líneas MMI, MMO y LOCK* en Z8000 y 8086 respectivamente.

Evaluación:	
	8086 B
	Z8000 E
	68000 R

En conclusión, los tres procesadores son bastante poderosos y versátiles. Para sistemas pequeños y medianos la circuitería necesaria es similar en los tres procesadores. En sistemas grandes el 8086 queda en deventaja debido a su menor capacidad de direccionamiento de memoria y sus recursos internos limitados (sólo cuatro registros de propósito general).

CAPITULO II

2.1 CONJUNTO DE INSTRUCCIONES DEL PROCESADOR 8086

2.1.1 MODOS DE DIRECCIONAMIENTO.- El 8086 proporciona 7 modos básicos de direccionamiento con algunas variantes, las cuales serán explicadas con cada uno de ellos. Los operandos pueden estar contenidos dentro de la Instrucción, en Registros, en memoria o en Puertos de Entrada Salida. Los modos de direccionamiento son los siguientes:

DIRECCIONAMIENTO INMEDIATO.- En este modo de direccionamiento uno de los operandos es especificado en uno o dos Bytes siguientes al código de operación.

FIG 28 DIRECCIONAMIENTO INMEDIATO

DIRECCIONAMIENTO DE REGISTRO.- En este modo, el operando está contenido en uno de los Registros de Propósito General especificados dentro del código de operación.

FIG 29 DIRECCIONAMIENTO DE REGISTRO

DIRECCIONAMIENTO DIRECTO.- En este modo, la dirección del operando en memoria es especificada como la suma del contenido del Registro de Segmento de Datos y el desplazamiento de 16 Bits contenido en los dos Bytes siguientes al código de operación.

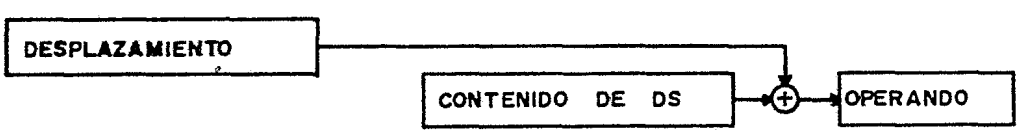


FIG 30 DIRECCIONAMIENTO DIRECTO

DIRECCIONAMIENTO INDIRECTO.- La dirección del operando en memoria es especificada por el contenido de un Registro Base o un Registro Indice.

FIG 31 DIRECCIONAMIENTO INDIRECTO

DIRECCIONAMIENTO INDEXADO.- En este modo, la dirección del operando en memoria es especificada por el contenido de uno de los dos Registros Indice (SI o DI). Este modo tiene la opción de especificar un desplazamiento de 16 Bits en los dos Bytes siguientes al código de operación, el cual es sumado al Registro Indice especificado.

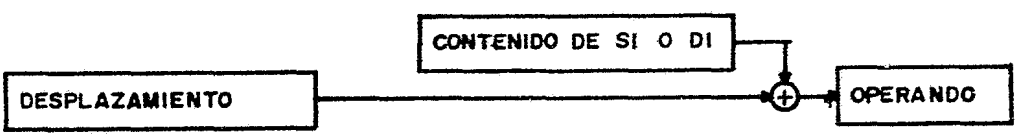


FIG 32 DIRECCIONAMIENTO INDEXADO

DIRECCIONAMIENTO BASE.- En este modo, la dirección del operando es especificada como la suma del contenido del Registro BX o BP y un Desplazamiento opcional. Si el Registro especificado es BP entonces el operando será obtenido del Segmento usado como Pila.

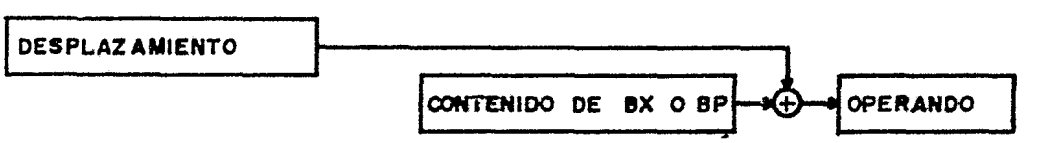


FIG 33 DIRECCIONAMIENTO BASE

DIRECCIONAMIENTO BASE INDEXADO.- En este modo, la dirección es especificada como la suma del contenido de un Registro Base (BX o BP), el contenido de un Registro Indice (SI o DI) y un Desplazamiento de 16 Bits.

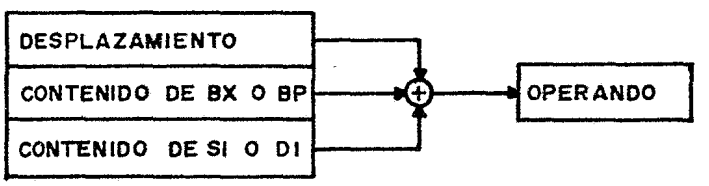


FIG 34 DIRECCIONAMIENTO BASE INDEXADO

DIRECCIONAMIENTO RELATIVO AL APUNTADOR DE INSTRUCCION.-En este modo, la dirección efectiva es especificada por la suma del contenido del Apuntador de Instrucción y un Desplazamiento signado de 8 o 16 Bits contenido en los Bytes siguientes al código de operación.

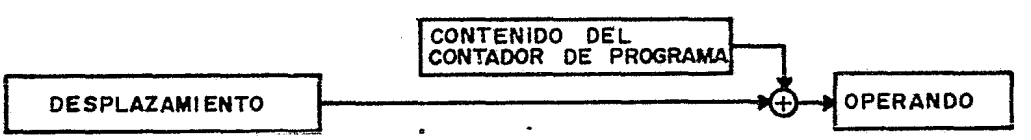


FIG 35 DIRECCIONAMIENTO RELATIVO AL CONTADOR DE PROGRAMA

2.1.2 TIPOS DE DATOS SOPORTADOS.- El procesador soporta los siguientes tipos de datos: Bits, Dígitos BCD, Bytes, Bytes ASCII y Palabras.

El espacio de memoria del procesador está organizado en Bytes los cuales son direccionables directamente. Las instrucciones y los datos en Bytes o en palabras pueden ser almacenados sin restricciones en cualquier dirección, sin embargo, las palabras almacenadas en direcciones impares no pueden ser transferidas completas, sino que serán transferidas por Bytes con lo cual aumenta el número de

ciclos necesarios para efectuar la transferencia.

2.1.3 CONJUNTO DE INSTRUCCIONES.- El 8086 tiene el formato de instrucción mostrado en la figura 36. Su conjunto de instrucciones está formado por 95 instrucciones básicas las cuales se encuentran divididas en los siguientes grupos:

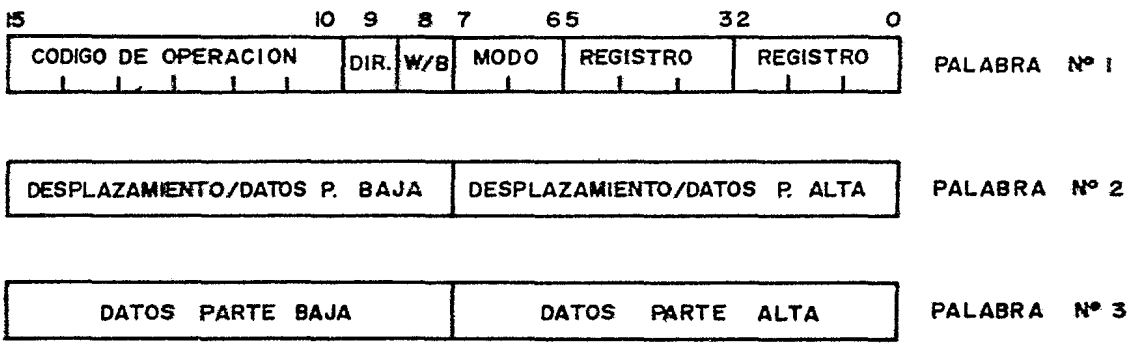


FIG 36 FORMATO DE INSTRUCCION

1.- INSTRUCCIONES DE MOVIMIENTO DE DATOS (MOV, LDS, LES, LEA, SAHF, XCHG PUSH, POP, PUSHF Y POPF).- Este grupo comprende doce instrucciones las cuales son utilizadas para transferir datos entre memoria y Registros de Propósito General y viceversa. Las instrucciones para manejo de Pila están contenidas en este grupo; la instrucción fundamental de este grupo es mueve datos (MOV) la cual es utilizada para transferencias de Bytes o Palbras entre Registros o entre éstos y Memoria, esta instrucción puede utilizar cualquiera de los modos de direccionamiento. También contiene instrucciones como Carga Apuntador utilizando el Registro DS (LDS) o utilizando el Registro ES (LES), estas operaciones transfieren un operando de 32 bits localizado en memoria al Registro especificado como destino y al Registro DS o al ES según la instrucción. Otras instrucciones contenidas en este grupo son Carga Dirección Efectiva (LEA), Carga Registro AH con Banderas (LAHF), Carga Registro AH en Banderas (SAHF), Intercambio (XCHG), Coloca en la Pila (PUSH), Extrae de la Pila (POP), Coloca Banderas en la Pila (PUSHF) y Extrae Banderas de la Pila (POPF).

2.- INSTRUCCIONES ARITMETICAS.-Este grupo comprende las instrucciones de Suma (ADD), Substracción (SUB), Multiplicación (MUL), División (DIV), Comparación (CMP), Incremento (INC), Decremento (DEC), Negación (NEG), Ajuste

Decimal (DAA), Ajuste ASCII (AAA) y de Extensión de Signo (CBW y CWD).

INSTRUCCIONES DE SUMA Y RESTA (ADD y SBB).- Estas instrucciones pueden operar sobre Bytes o Palabras usando cualquier modo de direccionamiento, existe la opción de utilizar la Bandera de ACARREO para la suma o substracción.

INSTRUCCIONES DE MULTIPLICACION Y DIVISION (MUL y DIV).- Estas instrucciones pueden operar sobre números enteros signados o no signados, contenidos en Registros o en Registro y Memoria, sobre Bytes o Palabras. La División puede efectuar una operación de 32 Bits entre 16 Bits a través de los Registros AX y DX los cuales contienen respectivamente la parte baja y alta del dividendo y BX o CX contienen el divisor de 16 Bits. El cociente es puesto en AX y el residuo en DX. Para la Multiplicación, la parte alta del resultado de 32 Bits es puesta en DX y la parte baja en AX.

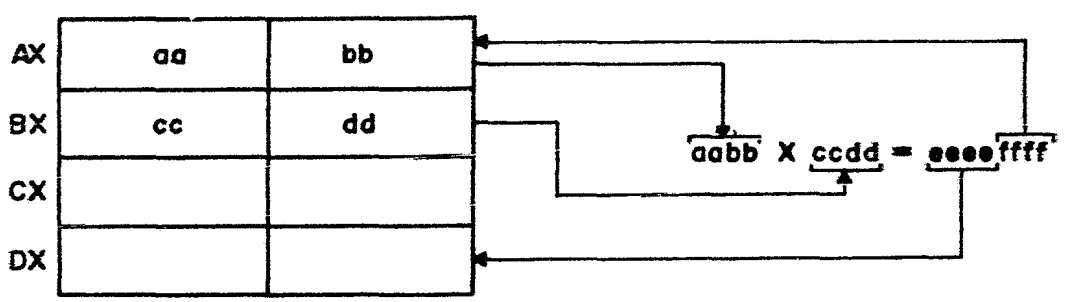


FIG 37A MULTIPLICACION

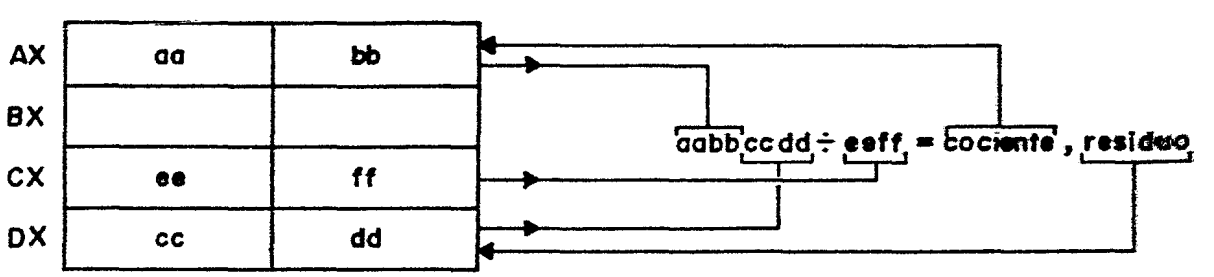


FIG 37B DIVISION

INSTRUCCIONES DE COMPARACION (CMP).- Estas instrucciones son utilizadas para comparar Bytes o Palabras contenidas en un Registro o en Memoria con datos inmediatos o con otro Registro o localidad de Memoria. Pueden utilizar todos los modos de direccionamiento.

INSTRUCCIONES DE INCREMENTO Y DECREMENTO (INC y DEC).- Estas instrucciones incrementan o decrementan por uno el operando especificado el cual puede ser un Registro o una localidad de Memoria, pueden operar sobre Bytes o Palabras utilizando cualquier modo de direccionamiento.

INSTRUCCIONES DE AJUSTE (DAA y AAA).- Estas instrucciones efectúan un ajuste ya sea decimal (BCD) o ASCII sobre el operando especificado. Estas son usadas después de una operación de Suma o Substracción con dígitos BCD para ajustar el resultado a la representación correspondiente. La operación de ajuste ASCII puede ser usada con cualquiera de las cuatro operaciones aritméticas.

INSTRUCCIONES DE EXTENSION DE SIGNO.- Estas instrucciones efectúan una extensión del Bit más significativo de operandos Byte o Palabra, al ejecutar la instrucción el Bit más Significativo del operando es transferido al Bit más significativo del Byte o Palabra Subsecuente.

3.- INSTRUCCIONES LOGICAS.- En este grupo están contenidas todas las instrucciones lógicas como son: Complemento (NOT), AND, OR, OR EXCLUSIVO, y una instrucción de prueba TEST, todas estas instrucciones pueden operar sobre Bytes o Palabras utilizando cualquier modo de direccionamiento. La operación TEST ejecuta una operación AND entre los operandos modificando únicamente el registro de Banderas.

4.- INSTRUCCIONES DE ROTACION Y CORRIMIENTO.- Existen cuatro Instrucciones de Corrimiento y cuatro de Rotación, las cuales son: Operaciones de Corrimiento Lógicas y Aritméticas hacia la derecha o hacia la izquierda SHL, SAL, SHR y SAR; y Operaciones de Rotación a la izquierda o a la derecha pasando o no a través de la Bandera de ACARREO ROL, ROR, RCL y RCR.

Las instrucciones de corrimiento pueden utilizarse para operar sobre bytes o palabras, se puede hacer hasta 255 corrimientos de acuerdo al valor especificado en la instrucción. Las rotaciones también pueden ser efectuadas en bytes o palabras, estas instrucciones son similares a las de corrimiento pero con la diferencia que los bits que salen no se pierden sino que son puestos en el extremo opuesto del operando.

5.- INSTRUCCIONES DE CONTROL DE PROGRAMA.- Cuatro grupos de instrucciones permiten manejar el control de ejecución de instrucciones dentro de un programa: Instrucciones de Transferencia Incondicional, Instrucciones de Transferencia Condicional, Instrucciones de Control de Iteraciones e Instrucciones de Control de Interrupciones.

INSTRUCCIONES DE TRANSFERENCIA INCONDICIONAL.- Estas instrucciones transfieren a otra localidad de memoria el control del programa sin necesidad de que se cumpla ninguna condición. Estas instrucciones son: Salto (JMP), Llamada de Subrutina (CALL) y Retorno de Subrutina (RET). El control de programa puede ser transferido a una dirección dentro del mismo segmento o a otro segmento de código.

INSTRUCCIONES DE TRANSFERENCIA CONDICIONAL.- Las Transferencias Condicionales ejecutan saltos siempre y cuando se cumpla alguna condición especificada en la instrucción. Existen 18 diferentes Códigos de Condición para la instrucción de Salto (Jcc), los saltos condicionales sólo pueden ejecutarse usando direccionamiento relativo al Contador de Programa. Por tanto, sólo se pueden ejecutar Transferencias Condicionales en un rango de -128 a +127 Bytes contados a partir de la dirección del primer Byte de la siguiente instrucción. Los Códigos de Condición utilizados son los siguientes:

MNEMONICO	CONDICION	"SALTA SI ES ..."
JA/JNBE	(CF O ZF)=0	MAYOR QUE
JAE/JNB	CF=0	MAYOR O IGUAL
JB/JNAE	CF=1	MENOR
JBE/JNA	(CF O ZF)=1	MENOR O IGUAL
JC	CF=1	ACARREO ENCENDIDO
JE/JZ	ZF=1	IGUAL O CERO
JG/JNLE	((SF xor OF) O ZF)=0*	MAYOR
JGE/JNL	(SF xor OF)=0*	MAYOR O IGUAL
JL/JNGE	(SF xor OF)=1*	MENOR
JLE/JNG	((SF xor OF) O ZF)=1*	MENOR O IGUAL
JNC	CF=0	ACARREO APAGADO
JNE/JNZ	ZF=0	DIFERENTE DE CERO
JNO	OF=0	NO SOBREFLUJO
JNP/PO	PF=0	PARIDAD IMPAR
JNS	SF=0	POSITIVO
JO	OF=1	SOBREFLUJO
JP/JPE	PF=1	PARIDAD PAR
JS	SF=1	NEGATIVO

* SOLO EN ARITMETICA EN COMPLEMENTO A DOS

TABLA DE CODIGOS DE CONDICION

INSTRUCCIONES DE CONTROL DE ITERACIONES.- Las instrucciones Lazo (LOOP), Lazo Mientras sea Cero (LOOPE), Lazo Mientras No sea Cero (LOOPNE) y Salto si CX es Cero (JCXZ) son utilizadas para controlar las repeticiones de lazos dentro de un Programa. Utilizan al Registro CX como un contador, cada vez que se ejecuta el lazo, CX es decrementado por uno y cuando la condición especificada es satisfecha se ejecuta un salto. El salto deberá estar entre -128 y +127 bytes contados a partir del primer byte de la siguiente instrucción.

INSTRUCCIONES DE CONTROL DE INTERRUPCIONES.- Las instrucciones Interrupción en Sobrepaso (INTO) e Interrupción (INT) permiten activar Rutinas de Servicio de Interrupción desde programas. El efecto de estas instrucciones es similar al de las Interrupciones Externas, sólo que el procesador no ejecuta el ciclo de Reconocimiento de Interrupción. La instrucción INT provoca que la Rutina de Interrupción especificada por el operando sea iniciada incondicionalmente y la Instrucción INTO provocará la ejecución de la Rutina de Servicio de Interrupción sólo si la Bandera SOBREPASO está prendida.

6.- INSTRUCCIONES DE MANEJO DE CADENAS.- Se tienen 5 instrucciones básicas para el manejo de cadenas de hasta 64K bytes de largo las cuales permiten manejar operandos Byte o Palabra. Hay instrucciones disponibles para Mover (MOVSB, MOVSW, LODS y STOS), Comparar (CMPS), y Buscar un Valor (SCAS) en un Registro hacia y desde una Cadena de Datos localizada en memoria. Los Registros Indice de Fuente (SI), Indice de Destino (DI), Registro Contador (CX) y Acumulador (AX) son usados como Apuntador de Fuente, Apuntador de Destino, Contador de Iteraciones y Registro para operando respectivamente. Las Banderas DF y ZF son usadas para indicar si la Instrucción es de Autoincremento o de Autodecremento y para indicar si el elemento buscado fué localizado.

7.- INSTRUCCIONES DE ENTRADA/SALIDA.- Existen dos instrucciones de transferencia de datos a dispositivos de E/S. La instrucción Entrada (IN) transfiere un Byte o una palabra desde un puerto de E/S hacia el Acumulador AX. La instrucción OUT transfiere un Byte o Palabra desde el Registro Acumulador AX hacia un puerto de E/S. El número de puerto puede ser especificado en una Constante de 8 Bits dentro de la instrucción con lo cual se tiene acceso a 256 puertos de E/S o por medio de un valor de 16 Bits contenido en el Registro DX el cual es usado como Apuntador.

8.-INSTRUCCIONES DE CONTROL DE PROCESADOR.- Estas instrucciones permiten a los programas controlar algunas funciones del procesador. Un grupo sirve para modificar a

las Banderas y otro grupo es utilizado para sincronizar al procesador con dispositivos externos.

INSTRUCCIONES DE MODIFICACION DE BANDERAS.- Las siguientes instrucciones permiten modificar el estado de algunas de las Banderas del Procesador: Prende Bandera de Acarreo (SCF), Borra Bandera de Acarreo (CLC), Complementa Bandera de Acarreo (CMC), Prende Bandera de Dirección (STD), Borra Bandera de Dirección (CLD), Prende Bandera de Interrupción (STI) y Borra Bandera de Interrupción (CLI).

INSTRUCCIONES DE SINCRONIZACION EXTERNA.- Este grupo está formado por cuatro instrucciones que permiten la sincronización del Procesador con Dispositivos Externos: Paro (HALT) provoca que el Procesador se detenga hasta que se presente una Interrupción Externa o una Restauración; Espera (WAIT) la cual causa que el Procesador entre a un estado de espera mientras que la línea TEST no sea activada; Escapa a Procesador Externo (ESC) la cual proporciona un medio para que un procesador externo obtenga un código de operación y/o un operando de memoria; LOCK es un prefijo de un byte el cual sólo funciona en Modo Máximo y causa que la línea LOCK* sea activada durante la ejecución de la siguiente instrucción, con esto no se permite que ningún dispositivo solicite control de los canales durante la ejecución de la instrucción. También se tiene la instrucción No Operación (NOP) la cual, como su nombre lo indica provoca una no operación.

2.2 CONJUNTO DE INSTRUCCIONES DEL PROCESADOR Z8000

2.2.1 MODOS DE DIRECCIONAMIENTO.- La información incluida en las instrucciones del Z8000 consiste de la función a realizar, el tipo y tamaño de los elementos de datos que van a ser manejados y la localización de estos. Las localidades son designadas por direcciones de registros, direcciones de memoria o direcciones de E/S. El modo de direccionamiento de una instrucción dada define el espacio de dirección al que se refiere y el método usado para calcular la dirección en sí. Los modos de direccionamiento son especificados explícitamente o pueden estar implícitos en la instrucción. Los ocho modos de direccionamiento soportados por el Z8000 son:

DIRECCIONAMIENTO POR REGISTRO.- En este modo el operando está contenido en el Registro de Propósito General especificado en un campo de cuatro bits dentro del Código de Operación.

FIG 38 DIRECCIONAMIENTO DE REGISTRO

DIRECCIONAMIENTO DE MEMORIA.- Las direcciones segmentadas de memoria están contenidas en un par de registros o en una localidad de memoria para palabra larga. El número de segmento y el desplazamiento pueden manipularse de manera separada o conjuntamente con todas las instrucciones disponibles para palabras y palabras largas.

Cuando una dirección segmentada está contenida en una instrucción, ésta puede tener dos representaciones, desplazamiento largo y desplazamiento corto. La representación en desplazamiento largo ocupa dos palabras, mientras que el desplazamiento corto requiere sólo una y combina en ella el número de segmento de 7 bits con un desplazamiento de 8 bits (por lo tanto un rango de 0 - 256). El direccionamiento de memoria se puede ejecutar de siete diferentes maneras que son:

DIRECCIONAMIENTO INMEDIATO.- En este modo de Direccionamiento uno de los operandos está contenido en la(s) palabra(s) siguiente(s) al código de operación. Se pueden especificar operandos byte, palabra o palabra larga.

En el caso de que el operando sea un byte, éste será puesto en los dos bytes de la palabra siguiente al código de operación.

FIG 39 DIRECCIONAMIENTO INMEDIATO

DIRECCIONAMIENTO DIRECTO.- Este modo utiliza la segunda, o segunda y tercera palabras del código de operación para identificar la dirección de un operando en memoria.

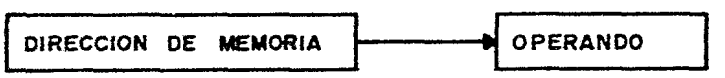


FIG 40 DIRECCIONAMIENTO DIRECTO

DIRECCIONAMIENTO DE REGISTRO INDIRECTO.- En este modo la dirección del operando es especificada por el contenido de uno de los Registros de Propósito General. R0 no puede ser utilizado como apuntador de memoria para este tipo de direccionamiento. Este modo de direccionamiento tiene la opción de ser autoincrementado o autodecrementado con algunas instrucciones.

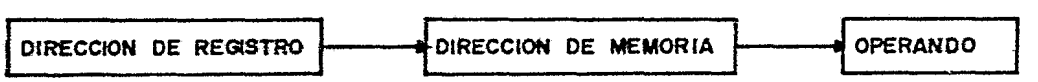


FIG 41 DIRECCIONAMIENTO INDIRECTO

DIRECCIONAMIENTO INDEXADO.- En este modo la dirección del operando es especificada por la suma del contenido de un Registro de propósito general, el cual contiene un desplazamiento y la Dirección Base especificada en la(s) siguiente(s) palabras del código de operación. R0 no puede

ser usado para contener el desplazamiento en este modo de direccionamiento.

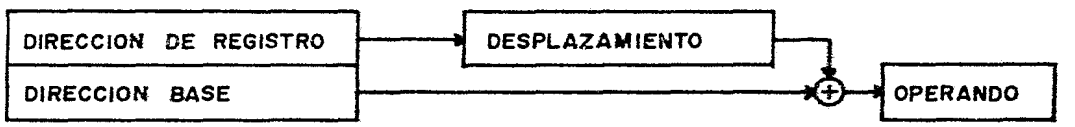


FIG 42 DIRECCIONAMIENTO INDEXADO

DIRECCIONAMIENTO BASE.- En este modo la dirección del operando es especificada por la suma de una dirección base contenida en uno de los Registros de Propósito General y un desplazamiento contenido en la segunda palabra del código de operación.

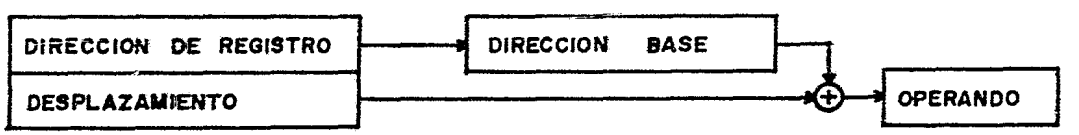


FIG 43 DIRECCIONAMIENTO BASE

DIRECCIONAMIENTO BASE INDEXADO.- Este Modo de direccionamiento funciona de manera similar al modo Dirección Base con la diferencia que el desplazamiento está contenido en uno de los Registros de Propósito General, una instrucción que utilice este modo de direccionamiento tendrá un código de operación de una palabra.

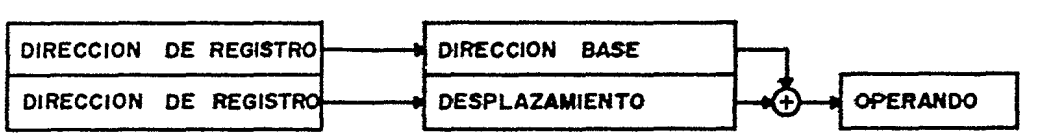


FIG 44 DIRECCIONAMIENTO BASE INDEXADO

DIRECCIONAMIENTO RELATIVO A CONTADOR DE PROGRAMA.- En este Modo de direccionamiento el operando es especificado por la suma del contenido del Contador de Programa y un desplazamiento especificado en la segunda palabra del código de operación.

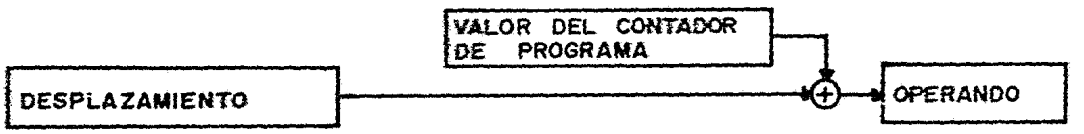


FIG 45 DIRECCIONAMIENTO RELATIVO AL CONTADOR DE PROGRAMA

2.2.2 TIPOS DE DATOS SOPORTADOS.- Las instrucciones del procesador pueden operar en bits, dígitos BCD (4 bits), bytes (8 Bits), palabras (16 bits), palabras largas (32 bits), palabras extra largas (64 bits) y cadenas de bytes. Todos los tipos de datos con excepción de las cadenas pueden estar en Registros de Propósito General o en memoria. Las cadenas son almacenadas sólo en memoria.

Toda la memoria del procesador está organizada en Bytes, sin embargo, ésta puede ser accesada como Bytes, Palabras o Palabras Largas. Las palabras y las palabras largas deberán estar localizadas siempre en localidades pares. Cuando una palabra esté en la dirección "N" donde" N" es un número par, entonces el Byte más significativo estará contenido en ésta dirección y el menos significativo en la dirección N+1.

2.2.3 CONJUNTO DE INSTRUCCIONES.- El formato general de instrucción de Z8000 se muestra a continuación:

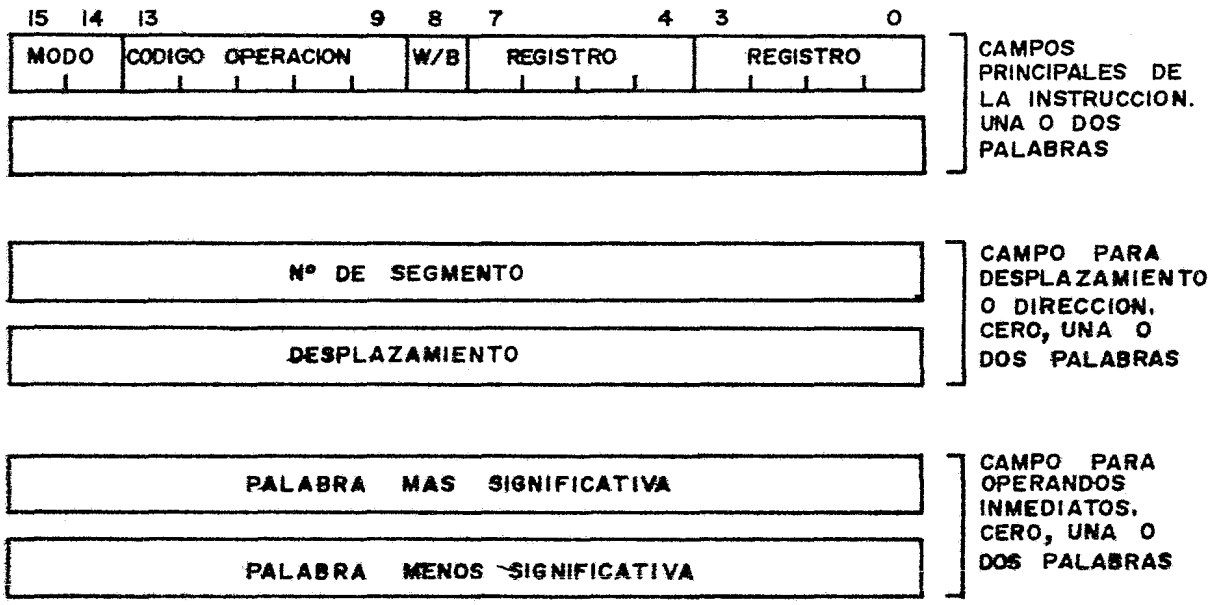


FIG 46 FORMATO DE INSTRUCCION

El procesador cuenta con 110 instrucciones básicas, las cuales, combinadas con los diferentes modos de direccionamiento y tipos de datos forman un conjunto de 414 instrucciones que se divide en nueve grupos funcionales:

1.- MOVIMIENTO DE DATOS.- Este conjunto contiene las instrucciones de Transferencia de Datos entre Registros de Propósito General y Memoria. Las instrucciones comprendidas en este grupo son: Instrucciones de Carga (LD), Instrucciones de Intercambio (EX), Instrucciones de Borrado (CLR), e Instrucciones de manejo de Pilas (POP y PUSH).

INSTRUCCIONES DE CARGA (LD).- Estas instrucciones efectúan movimientos de datos entre los Registros de Propósito General y la memoria, pueden operar en Bytes, Palabras y Palabras Largas. Asimismo, pueden usar todos los modos de direccionamiento con la restricción de que al menos uno de los operandos debe estar contenido en uno de los Registros de Propósito General. Existen versiones de carga

múltiple con las cuales se puede cargar un número específico (entre 0 y 15) de Registros de propósito general o de localidades de memoria con una sola instrucción.

INSTRUCCIONES DE INTERCAMBIO (EX).- Estas son dos instrucciones que efectúan intercambio de datos entre dos Registros o entre un Registro y una localidad de memoria. Pueden operar en Bytes o en Palabras con modos de direccionamiento Directo o Indexado.

INSTRUCCIONES DE BORRADO (CLR).- Estas dos instrucciones almacenan un cero en el operando especificado como destino. Pueden operar en Bytes o en Palabras usando modos de direccionamiento Registro, Directo, Indirecto o Indexados.

INSTRUCCIONES DE MANEJO DE PILAS (POP y PUSH).- Las instrucciones POP y PUSH sirven para la implementación de Pilas usando como Apuntador de Pila cualquier Registro excepto R0. El operando destino puede ser especificado usando direccionamiento de Registro, Inmediato, Directo, Indirecto o Indexado, el operando fuente deberá ser especificado usando direccionamiento de Registro Indirecto, el Registro usado como Apuntador será incrementado o decrementado automáticamente según la operación.

2.- INSTRUCCIONES ARITMETICAS.- Este grupo comprende las instrucciones aritméticas, para comparación, incremento, decremento, extensión de signo y negación.

INSTRUCCIONES DE SUMA (ADD).- Existen Instrucciones de suma para operar sobre Bytes, Palabras y Palabras Largas usando modos de direccionamiento por Registro, Inmediato, Directo, Indirecto o Indexado. Existen instrucciones de suma con acarreo que sólo pueden operar sobre Bytes o Palabras con Modo de Direccionamiento por Registro. Las instrucciones para Bytes pueden operar sobre números BCD.

INSTRUCCIONES DE SUBSTRACCION (SUB).- Existen instrucciones de subtracción equivalentes a las instrucciones de suma mencionadas anteriormente.

INSTRUCCIONES DE NEGACION (NEG).- Existen dos instrucciones que complementan a dos el operando indicado como destino, pueden operar sobre Bytes y Palabras usando modo de Direccionamineto de Registro, Directo, Indirecto o Indexado.

INSTRUCCIONES DE MULTIPLICACION Y DIVISION (MUL,DIV).- Estas instrucciones pueden operar sobre Palabras y PALABRAS LARGAS signadas o no signadas usando modos de direccionamiento Registro, Inmediato, Directo, Indirecto o

Indexado. Para la multiplicación de palabras, el operando destino deberá estar contenido en un Registro par ya que el resultado de 32 Bits será almacenado en éste. Similarmente, para multiplicación de palabras largas el operando destino deberá estar contenido en un Registro cuádruple en el cual será almacenado el resultado de 64 bits. Para la División, los operandos destino deberán ser especificados de una manera similar a la multiplicación y el resultado es almacenado de la siguiente manera: después de efectuar la división, el Cociente es puesto en la parte baja del operando destino mientras que el Residuo es puesto en la parte alta del Operando destino.

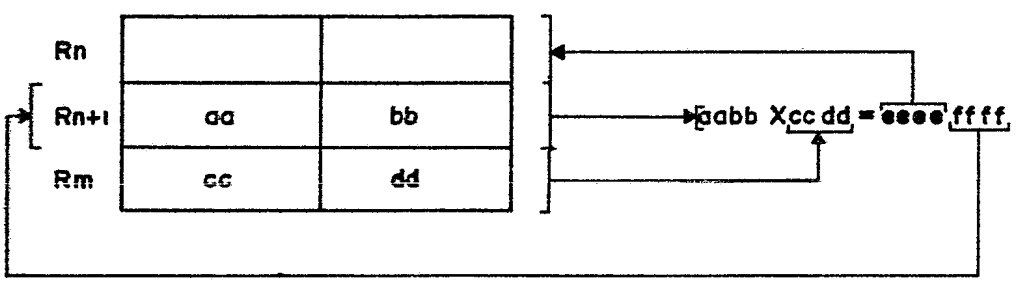


FIG 47A **MULTIPLICACION**

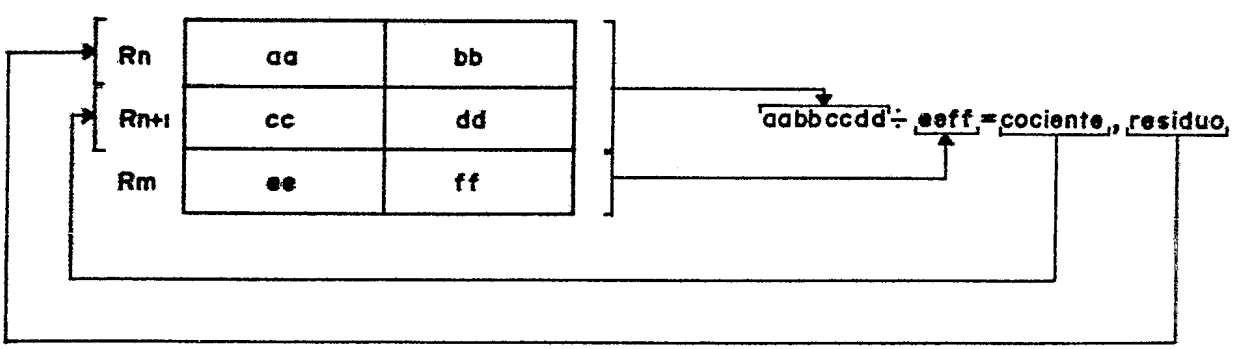


FIG 47B **DIVISION**

INSTRUCCIONES DE COMPARACION (CP).- Estas instrucciones comparan Bytes, Palabras o Palabras Largas mediante substracción (aunque sin salvar el resultado) usando Modos de Direccionamiento de Registro, Inmediato, Directo, Indirecto e Indexado; hay una variación de las instrucciones que opera sobre Bytes o Palabras comparando un dato inmediato con un operando destino usando direccionamiento Directo, Indirecto o Indexado.

INSTRUCCIONES DE INCREMENTO Y DECREMENTO (INC,DEC).- Estas instrucciones Incrementan o Decrementan un operando destino el cual puede ser una Palabra o un Byte por una cantidad N especificada en un campo de cuatro bits dentro de la instrucción usando modos de direccionamiento Registro, Directo, Indirecto o Indexado.

INSTRUCCIONES DE EXTENSION DE SIGNO (EXTS).- Estas instrucciones operan sobre Bytes, Palabras y Palabras Largas, sólo pueden operar sobre valores contenidos en Registros y su función es la de correr el Bit de signo del operando hacia la parte superior del Registro especificado.

INSTRUCCION DE AJUSTE DECIMAL (DAB).- Esta instrucción efectúa un ajuste decimal sobre un Byte contenido en un Registro de Propósito General, asume que el Registro contiene el resultado de una suma o resta de números BCD.

3.- INSTRUCCIONES LOGICAS.- Este grupo comprende las instrucciones que efectúan operaciones lógicas como son AND, OR, XOR, Complemento y Chequeo. Las instrucciones AND, OR y XOR pueden operar sobre Bytes y Palabras con uno de los operandos contenido en un Registro y el otro puede ser especificado usando direccionamiento de Registro, Inmediato, Directo, Indirecto o Indexado. La Instrucción COM (complemento a uno) puede operar sobre Bytes o Palabras usando los mismos modos de direccionamiento excepto el Inmediato. Las instrucciones de Chequeo TEST efectúan una operación OR entre el operando destino y cero colocando las banderas apropiadas según el resultado de la operación. Pueden usar los mismos modos de direccionamiento que la instrucción COM. La instrucción TCC (Prueba Códigos de Condición) prueba el código de condición especificado en la instrucción comparandolo con las banderas de condición en el Registro de Banderas y Control. Las banderas que coincidan serán puestas como uno en el contenido del Registro de Byte o Palbra especificado como destino.

4.- INSTRUCCIONES DE ROTACION Y CORRIMIENTO (RR, RL, SR, SL).- Estas instrucciones operan sobre Bytes o Palabras contenidas solamente en Registros de Propósito General. Existen operaciones de Rotación y Corrimiento a la derecha o a la izquierda, pueden ser Aritméticas o Lógicas. Los

Corrimientos y Rotaciones se efectúan a través del Bit de ACARREO en el Registro de Banderas y Control. El número de Corrimientos es especificado en la operación y puede ser entre 1 y 16 (cero especifica 16 corrimientos). Las instrucciones de rotación sólo pueden tener hasta un máximo de 2 corrimientos a la vez.

5.-INSTRUCCIONES DE MANEJO DE BITS.- Este grupo de instrucciones efectúa operaciones de Chequeo, Prendido y Borrado de Bits. Todas pueden operar sobre Bytes o Palabras, existen dos tipos de operaciones: las Dinámicas, en las cuales el número de Bit es especificado por el contenido de un Registro de Propósito General y que sólo pueden usar Direccionamiento de Registro; las Estáticas en las cuales el número de Bit es fijo y es especificado dentro del código de operación en un campo de cuatro bits, pueden usar Modos de Direccionamiento Registro, Directo, Indirecto o Indexado. La Instrucción de Prueba es BIT, la cual pone la Bandera CERO en un estado igual al complemento del valor del bit que fué probado; la instrucción SET prende Bits y la Instrucción RES los borra. Existe también la instrucción TSET (Checa y Enciende) la cual causa que la Bandera SIGNO en el Registro de Banderas y Control adquiera el estado del Bit más significativo del Operando Fuente, después el operando destino es llenado con unos.

6.- INSTRUCCIONES DE CONTROL DE PROGRAMA.- Este grupo comprende las instrucciones de Llamada de Subrutina, Saltos Condicionales y No Condicionales, Instrucciones de Retorno de Subrutinas e Interrupciones y Llamada de Sistema. Hay dos instrucciones de llamada a subrutina que son CALL y CALR, la diferencia entre ellas es que CALR usa direccionamiento relativo al Contador de Programa con un rango de desplazamiento posible de -4094 a 4096 Bytes; la instrucción CALL puede usar modos de direccionamiento Directo, Indirecto o Indexado.

Las Instrucciones de Salto son dos: JP y JR cuya diferencia es la misma que en el caso de las Instrucciones de Llamada de Subrutina, además tienen la característica de poder efectuar saltos no condicionales o condicionales utilizando los códigos mostrados en la siguiente tabla.

NOTACION	CONDICION	VERDADERO SI
NZ	NO CERO	Z=0
ZR	CERO	Z=1
NC	NO ACARREO	C=0
CY	ACARREO	C=1
PO	PARIDAD IMPAR	P/V=0
PE	PARIDAD PAR	P/V=1

NOTACION	CONDICION	VERDADERO SI
PL	MAS	S=0
MI	MENOS	S=1
NE	NO IGUAL	Z=1
EQ	IGUAL	Z=0
NOV	NO SOBREFLUJO	P/V=0
OV	SOBREFLUJO	P/V=1
GE	MAYOR O IGUAL	S xor P/V=0
LT	MENOR QUE	S xor P/V=1
GT	MAYOR QUE	Z or (S or P/V)=0
LE	MENOR O IGUAL	Z or (S or P/V)=1

TABLA DE CODIGOS DE CONDICION

Se tienen dos instrucciones de Retorno en el procesador IRET y RET, la primera es una instrucción para retorno incondicional de Rutinas de Interrupción. La segunda es usada para Retorno de Subrutinas y puede ser incondicional o condicional utilizando los códigos de condición mencionados anteriormente.

Existen dos instrucciones con las cuales se pueden crear Lazos condicionales dentro de un programa, estas instrucciones son: DJNZ y DBJNZ la diferencia entre ellas es que la primera opera sobre un Registro de palabra y la segunda sobre uno de Byte. Estas instrucciones decrementan el contenido del Registro especificado, lo comparan con cero y si el contenido es diferente entonces ejecutan un salto a la dirección del programa definida por la suma del Contador de Programa y un desplazamiento negativo de 8 Bits el cual está contenido en la palabra de código de operación.

La instrucción de Llamada de Sistema (SC), coloca el contenido del Contador de Programa junto con el contenido del Registro de Banderas y Control en la Pila de Sistema, después de haber hecho esto carga nuevos valores del Contador de Programa y el Registro de Banderas y Control del Area del Nuevo Status de Programa. Esta instrucción puede ser usada para efectuar un cambio controlado de Modo Normal a Modo Sistema.

7.- INSTRUCCIONES DE TRANSFERENCIA DE BLOQUES Y MANEJO DE CADENAS.- Este grupo comprende las siguientes instrucciones:

INSTRUCCIONES DE COMPARACION CON INCREMENTO O DECREMENTO AUTOMATICO.- Las instrucciones Compara y Decrementa (CPD), Compara e Incrementa (CPI), Compara Decrementa y Repite (CPDR) y Compara Incrementa y Repite (CPIR) son utilizadas para comparar un dato en un Registro con una cadena de datos contenida en memoria direccionada

Indirectamente usando un Registro como apuntador. Este Registro es decrementado o incrementado por uno o por dos dependiendo de si la instrucción es para Bytes o para Palabras. El código de condición incluido en la Instrucción es comparado con las banderas, si existe igualación entonces la bandera CERO es encendida. Si el Registro de 16 Bits especificado como contador llega a cero entonces la bandera SOBREPASO es puesta indicando que se llegó al final de la cadena. La diferencia entre estas instrucciones y las de repetición es que estas últimas son reejecutadas hasta que la Bandera CERO o la de SOBREPASO sean encendidas.

INSTRUCCIONES DE COMPARACION DE CADENAS.- Estas instrucciones funcionan de manera similar a las instrucciones mencionadas anteriormente con la diferencia que ambos operandos son localizados usando direccionamiento Indirecto. Existe un equivalente de cada una de las operaciones mencionadas anteriormente la cual puede ser usada para comparación de Cadenas.

INSTRUCCIONES DE TRANSFERENCIA CON DECREMENTO O INCREMENTO AUTOMATICO.- Las Instrucciones Transfiere y Decrementa (LDD), Transfiere e Incrementa (LDI), Transfiere Decrementa y Repite (LDDR) y Transfiere Incrementa y Repite (LDIR) son usadas para mover cadenas de datos de una localidad de memoria a otra. Los operandos fuente y destino son localizados usando Direccionamiento Indirecto, por lo que es necesario especificar un Registro de 16 Bits el cual es usado como Contador y es decrementado por uno en cada ejecución de la instrucción; si este Registro llega a cero, la Bandera de Sobrepaso es prendida, los Registros usados como apuntadores de fuente y destino son decrementados o incrementados por uno si la operación es de Bytes y por dos si es de Palabras. Las operaciones con Repetición son reejecutadas hasta que un bloque completo de datos ha sido transferido.

INSTRUCCIONES DE TRADUCCION.- Este grupo está formado por las instrucciones Traduce y Decrementa (TRDB), Traduce e Incrementa (TRIB), Traduce Decrementa y Repite (TRDRB), Traduce Incrementa y Repite (TRIRB), Traduce Prueba y Decrementa (TRTDB) Traduce Prueba e Incrementa (TRTIB), Traduce Prueba Decrementa y Repite (TRTDRB) y Traduce Prueba Incrementa y Repite (TRTIRB). Estas instrucciones traducen caracteres de ocho bits de un código a otro. El Registro de Dirección del destino identifica el carácter a ser cambiado. El Registro de Dirección Fuente identifica la primera o la última dirección de memoria de una tabla de bytes conteniendo los nuevos códigos de caracteres. El contenido de la localidad direccionada como destino es usado como índice no signado de ocho bits dentro de la tabla de traducción. La suma de este índice con la dirección del

operando fuente identifica un byte en la tabla de traslación cuyo contenido substituye al byte destino original. Para las instrucciones con repetición, la diferencia es que la ejecución continúa hasta que el contenido del Registro Contador es decrementado a cero. Para las instrucciones con la opción de prueba el contenido del Byte seleccionado en la tabla de traslación es transferido al Registro RH1 y la Bandera CERO es encendida si el Contenido de este Registro es cero.

8.- INSTRUCCIONES DE E/S.- Este conjunto de Instrucciones realiza transferencias de 8 o 16 bits entre el procesador y dispositivos de E/S. Todas las Instrucciones son Privilegiadas por lo cual sólo se pueden ejecutar en Modo Sistema. Hay dos tipos de instrucciones disponibles: Normales y Especiales. Las Instrucciones Normales incluyen un conjunto de instrucciones de Entrada, Salida y E/S en bloque para bytes y palabras. Las Instrucciones Especiales son usadas para programar la Unidad de Manejo de Memoria. La información en las Líneas de Código de Status distingue entre las instrucciones especiales y las Normales. Para las Instrucciones Especiales el Código de Status proporcionado es un Código de Función Especial y para las Instrucciones Normales el Código de Status es el de una referencia a E/S.

INSTRUCCIONES NORMALES.- Este grupo comprende las siguientes instrucciones: Entrada (IN), Salida (OUT), Entrada con Decremento (IND), Entrada con Incremento (INI), Entrada con Decremento y Repetición (INDR) y Entrada con Incremento y Repetición (INIR), y un equivalente en Salida para cada una de las anteriores. Pueden operar con Bytes o con Palabras con Modos de Direccionamiento de Registro o Indirecto. Las instrucciones con Decremento/Incremento y Repetición sólo pueden usar el Modo de Direccionamiento Indirecto, el funcionamiento de estas instrucciones es igual al de las instrucciones equivalentes para manejo de cadenas.

INSTRUCCIONES ESPECIALES.- Existe una Instrucción Especial equivalente para cada una de las Instrucciones Normales. Estas instrucciones son usadas únicamente para tranferir datos desde y hacia la Unidad de Manejo de Memoria usando los mismos modos de direccionamiento que las instrucciones normales.

9.- INSTRUCCIONES DE CONTROL DEL PROCESADOR.- Este grupo comprende las siguientes instrucciones no privilegiadas: Complementa Bandera (COMFLG), Carga en Byte de Banderas (LDCTLB), No Operación (NOP), Borra Bandera (RESFLG) y Prende Bandera (SETFLG); y las siguientes Instrucciones Privilegiadas: Deshabilita Interrupción (DI), Habilita Interrupción (EI), Alto (HALT), Carga en Registro de Control (LDCTL), Carga del Registro de Control (LDCTL), Carga Estado

de Programa (LDPS), Prueba Bit Multi-Micro (MBIT), Solicitud de Multi-Micro (MREQ), Desactiva Línea Multi-Micro (MRES) y Activa Línea Multi-Micro (MSET).

La instrucción Complementa Bandera (COMFLG) complementa la(s) bandera(s) especificada(s) en el código de Instrucción dentro del Registro de Banderas y Control.

La instrucción LDCFLG carga hacia o desde el Registro de Banderas (parte menos significativa del Registro de Banderas y Control) el Byte especificado.

Las instrucciones SETFLG y RESFLG Prenden y Borran Banderas específicas dentro del Registro de Banderas.

Las instrucciones Privilegiadas DI y EI deshabilitan o habilitan respectivamente uno o ambos tipos de Interrupciones Mascarables según se especifique.

La instrucción privilegiada HALT detiene al Procesador.

La instrucción LDCTL carga hacia o desde el Registro de Control (Parte alta del Registro de Banderas y Control) el Byte especificado. Esta instrucción es usada también para cargar los Apuntadores de Pila, el Registro de Refrescamiento, y el Registro Apuntador de Area de Nuevo Status de Programa.

La instrucción LDPS carga nuevos valores de Contador de Programa y de Registro de Banderas y Control tomados de memoria usando Modos de Direccionamiento Directo, Indirecto o Indexado.

Las instrucciones MBIT, MREQ, MRES y MSET son usadas para proporcionar soporte a sistemas de multiproceso. La instrucción MBIT Prueba el estado de la línea MI y coloca el resultado de la prueba en la Bandera SIGNO. Esto es usado para ver si un recurso compartido está disponible o no. Las instrucciones MSET y MRES son usadas para activar y desactivar la línea de Salida Multi-Micro con lo cual se indica que el recurso compartido está siendo usado por el procesador. La instrucción MREQ es usada para solicitar acceso a un recurso compartido, esta instrucción coloca en el Canal de Datos el contenido de un Registro especificado en la instrucción para implementar un protocolo de solicitud. Las banderas CERO y SIGNO son usadas para indicar si la solicitud tuvo éxito o no según la siguiente tabla:

CERO	SIGNO	SIGNIFICADO
0	0	No se hizo solicitud.
1	0	Se hizo solicitud de acceso pero fué negada.
1	1	Se hizo solicitud de acceso pero fué negada.
0	1	Sin validez

2.3 CONJUNTO DE INSTRUCCIONES DEL PROCESADOR 68000

2.3.1 MODOS DE DIRECCIONAMIENTO.- La mayoría de las instrucciones del procesador especifican el modo de direccionamiento de los operandos usando el Campo de Dirección Efectiva (bits 0 al 5) en la palabra de operación. Este campo en la palabra de operación está subdividido en dos campos de 3 bits: el Campo de Modo y el Campo de Registro. El valor en el Campo de Modo selecciona los diferentes modos de direccionamiento y el valor en el Campo de Registro selecciona el número de registro a ser empleado en el cálculo de la dirección.

Es posible que el cálculo de la dirección efectiva requiera información adicional para especificar al operando completamente. Esta información adicional, llamada Extensión de la Dirección Efectiva, está contenida en la(s) siguiente(s) palabras de la instrucción y son consideradas como parte de ésta. Los modos de Direccionamiento Efectivo están agrupados en tres categorias: Registro Directo, Direccionamiento de Memoria y Especiales.

MODOS DE REGISTRO DIRECTO.- Estos modos de direccionamiento efectivo especifican que el operando está en uno de los 16 Registros de Propósito General:

REGISTRO DE DATOS DIRECTO.- El operando está contenido en el Registro de Datos especificado por el Campo de Registro.

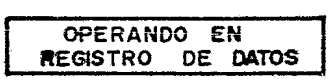


FIG 48 DIRECCIONAMIENTO DE REGISTRO DE DATOS DIRECTO

REGISTRO DE DIRECCIONES DIRECTO.- El operando está contenido en el Registro de Direcciones especificado por el Campo de Registro.

FIG 49 DIRECCIONAMIENTO DE REGISTRO DE DIRECCIONES DIRECTO

MODOS DE DIRECCIONAMIENTO DE MEMORIA.- Estos modos de direccionamiento efectivo especifican que el operando está en memoria y proporcionan la dirección específica de éste.

DIRECCIONAMIENTO INDIRECTO EN REGISTRO DE DIRECCIONES.- En este modo, la dirección del operando está contenida en el Registro de Direcciones especificado por el Campo de Registro en la instrucción. La referencia es clasificada como una referencia de datos con excepción de las instrucciones de salto y llamada de subrutina.

FIG 50 DIRECCIONAMIENTO INDIRECTO

DIRECCIONAMIENTO INDIRECTO EN REGISTRO DE DIRECCIONES CON POSTINCREMENTO.- La dirección del operando está contenida en el Registro de Direcciones especificado por el Campo de Registro en la instrucción. Después de que la dirección del operando es usada ésta es incrementada por uno, dos o cuatro dependiendo de si el operando es un byte, una palabra o una palabra larga. Si el Registro de Dirección es el Apuntador de Pila y el operando es un byte, entonces la dirección es incrementada por dos en lugar de por uno para mantener el límite de la Pila en una dirección par. La Referencia es clasificada como una referencia de datos.

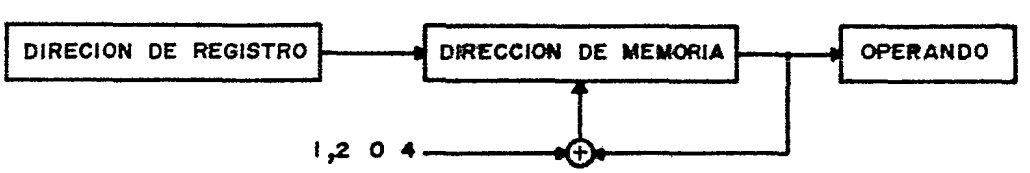


FIG 51 DIRECCIONAMIENTO INDIRECTO CON POSTINCREMENTO

DIRECCIONAMIENTO INDIRECTO EN REGISTRO DE DIRECCIONES CON PREDECREMENTO.- La dirección del operando está contenida en el Registro de Direcciones especificado por el Campo de Registro en la instrucción. Antes de que la dirección sea usada, ésta será decrementada por uno, dos o cuatro dependiendo de si el operando es un byte, una palabra o una palabra larga. Si el Registro de Direcciones es el Apuntador

de Pila y el operando es un byte, entonces la dirección es decrementada por dos para mantener el límite de la Pila en una dirección par. La referencia es clasificada como una referencia de datos.

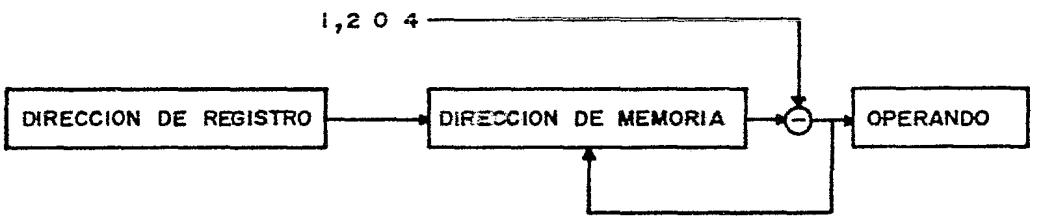


FIG 52 DIRECCIONAMIENTO INDIRECTO CON PREDECREMENTO

DIRECCIONAMIENTO INDIRECTO EN REGISTRO DE DIRECCIONES CON DESPLAZAMIENTO.- Este modo de direccionamiento requiere de una palabra de extensión. La dirección del operando es la suma de la dirección contenida en un Registro de Dirección y el número de 16 bits extendido en signo contenido en la palabra de extensión que especifica el desplazamiento. La referencia es clasificada como una referencia de datos con excepción de las instrucciones de llamada de subrutina.

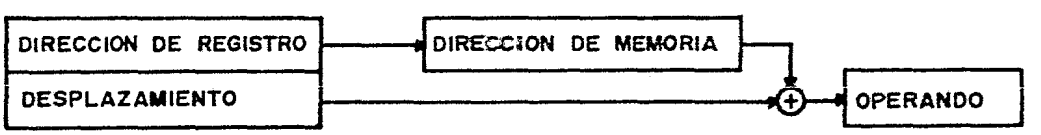


FIG 53 DIRECCIONAMIENTO INDIRECTO CON DESPLAZAMIENTO

DIRECCIONAMIENTO INDIRECTO EN REGISTRO DE DIRECCIONES CON INDICE.- Este modo de direccionamiento requiere de una palabra de extensión. La dirección del operando es la suma de tres direcciones: el contenido del Registro de Direcciones especificado en el Campo de Registro, el número entero de 8 bits extendido en signo contenido en los 8 bits menos significativos de la palabra de extensión los cuales especifican el Desplazamiento y el contenido del Registro Indice que puede ser cualquier Registro de Direciones o de Datos. La referencia es clasificada como una referencia de datos con excepción de las instrucciones de salto y llamada de subrutina.

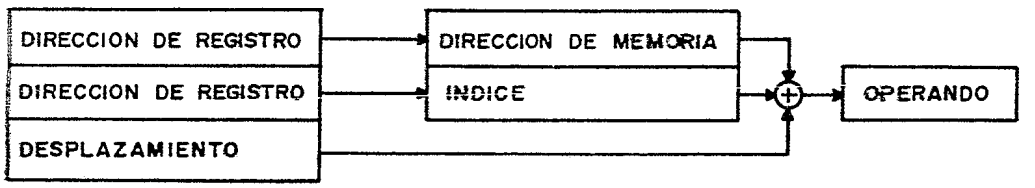


FIG 54 DIRECCIONAMIENTO INDIRECTO CON INDICE

MODOS DE DIRECCIONAMIENTO ESPECIALES.-Estos modos de direccionamiento usan el Campo de Registro en la instrucción para especificar el modo de direccionamiento especial en lugar de un número de Registro.

DIRECCIONAMIENTO ABSOLUTO CORTO.- Este modo de direccionamiento requiere de una palabra de extensión. La dirección del operando es la palabra de extensión. La dirección de 16 bits es extendida en signo antes de ser usada. La referencia es clasificada como una referencia de datos con excepción de las instrucciones de salto y llamada de subrutina.

FIG 55 DIRECCIONAMIENTO ABSOLUTO CORTO

DIRECCIONAMIENTO ABSOLUTO LARGO.- Este modo de direccionamiento requiere de dos palabras de extensión. La dirección del operando es obtenida mediante la concatenación de las palabras de extensión. La parte alta de la dirección es la primera palabra de extensión y la parte baja es la segunda palabra de extensión. La referencia es clasificada como de datos con excepción de las instrucciones de salto y llamada de subrutina.

FIG 56 DIRECCIONAMIENTO ABSOLUTO LARGO

DIRECCIONAMIENTO RELATIVO AL CONTADOR DE PROGRAMA CON DESPLAZAMIENTO.- Este modo de direccionamiento requiere una palabra de extensión. La dirección es la suma de la dirección en el Contador de Programa y el número entero de 16 bits extendido en signo contenido en la palabra de extensión el cual especifica el desplazamiento. El valor del Contador de Programa es la dirección de la palabra de extensión. La referencia es clasificada como referencia a programa.

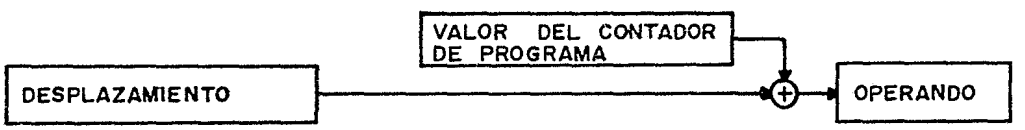


FIG 57 DIRECCIONAMIENTO RELATIVO AL CONTADOR DE PROGRAMA CON DESPLAZAMIENTO

DIRECCIONAMIENTO RELATIVO AL CONTADOR DE PROGRAMA CON INDICE.- Este modo de direccionamiento requiere una palabra de extensión. La dirección es la suma de la dirección en el Contador de Programa, el número de 8 bits extendido en signo contenido en la palabra de extensión, que especifica el desplazamiento y el contenido de un Registro Indice que puede ser un Registro de Datos o de Direcciones. El valor en el Contador de Programa es la dirección de la palabra de extensión. La referencia es clasificada como una referencia de programa.

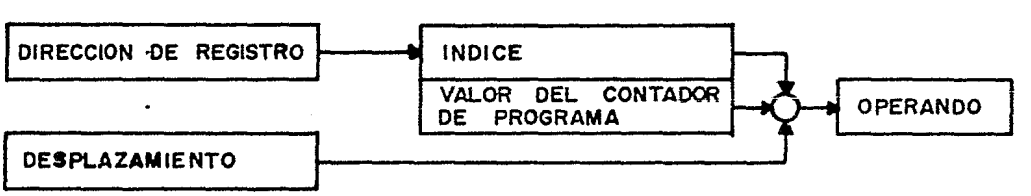


FIG 58 DIRECCIONAMIENTO RELATIVO AL CONTADOR DE PROGRAMA CON INDICE

DIRECCIONAMIENTO INMEDIATO.- Este modo de direccionamiento requiere de una o dos palabras de extensión dependiendo del tamaño de la operación.

Operación de Byte.- El operando es la parte baja de la palabra de extensión.

Operación de Palabra.- El operando es la palabra de extensión.

Operación de Palabra Larga.- El operando está formado por dos palabras de extensión, los 16 bits más significativos son la primera palabra de extensión y los 16 bits menos significativos son la segunda palabra de extensión.

FIG 59 DIRECCIONAMIENTO INMEDIATO

DIRECCIONAMIENTO CON CODIGOS DE CONDICION O CON REGISTRO DE STATUS.- Un grupo de instrucciones pueden referirse al Registro de Status a través del Campo de Dirección Efectiva. Todas estas instrucciones son privilegiadas. Estas operaciones son: ANDI al CCR ("Y" al Registro de Código de Condición CCR), ANDI al SR ("Y" al Registro de Status SR), EORI al CCR ("O" Exclusivo al CCR), EORI al SR ("O" Exclusivo al SR), ORI al CCR ("O" al CCR) y ORI al SR ("O" al SR).

DIRECCIONAMIENTO IMPLICITO.- Algunas instrucciones hacen una referencia implícita al Contador de Programa (PC), al Apuntador de Pila (SP), al Apuntador de Pila de Modo Supervisor (SSP), al Apuntador de Pila de Modo Usuario (USP) o al Registro de Status (SR).

2.3.2 TIPOS DE DATOS SOPORTADOS.- El 68000 soporta los siguientes tipos de datos:

BITS
ENTEROS DE 8, 16 Y 32 BITS
DATOS DECIMALES CODIFICADOS EN BINARIO
DIRECCIONES DE 16 Y 32 BITS

Los bytes son direccionables directamente. En el caso de palabras y palabras largas, el byte de mayor orden estará localizado en una dirección par y el de menor orden en una

dirección impar la cual será la siguiente localidad en orden creciente de la dirección de la palabra. Las instrucciones, palabras y datos con bytes múltiples son direccionados siempre en localidades pares. Si un dato de palabra larga está localizado en la dirección "n" (n=par), entonces la segunda palabra del dato estará localizada en la dirección n+2.

En general, el modo de direccionamiento es independiente del tipo de datos. También, en los casos en que sea válido (enteros, operandos lógicos y direcciones) el tamaño del operando puede ser especificado independientemente de la operación. El resultado puede ser almacenado ya sea en un Registro o en una Localidad de Memoria. Este tipo de operaciones de "Registro a Memoria" reduce el número de operaciones de almacenamiento en registros requeridas para mantener los resultados.

La mayoría de las operaciones pueden ser especificadas para trabajar de memoria a registro, de registro a registro, de registro a memoria, inmediato a registro o inmediato a memoria.

2.3.3 CONJUNTO DE INSTRUCCIONES DEL 68000.- El conjunto de instrucciones del procesador está formado por 56 instrucciones básicas, estas instrucciones, combinadas con los diferentes modos de direccionamiento y tipos de datos forman un conjunto de más de mil instrucciones. El formato de instrucción se muestra en la figura 61. El conjunto de instrucciones puede ser dividido en ocho grupos funcionales que son:

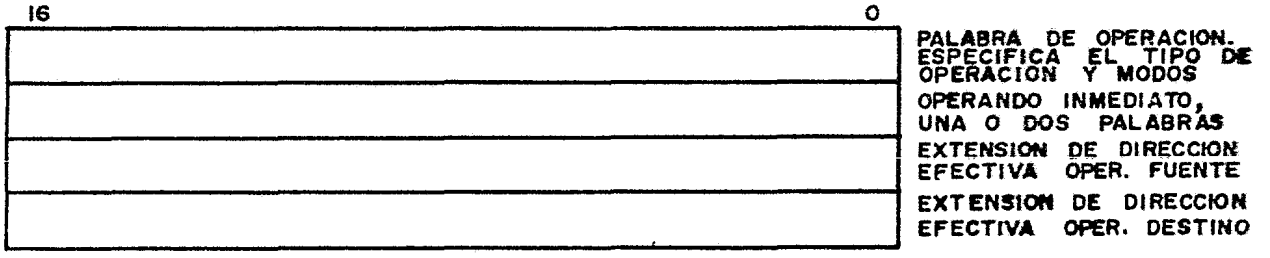


FIG 60 FORMATO DE INSTRUCCION

1.-INSTRUCCIONES DE MOVIMIENTO DE DATOS.-Estas instrucciones realizan la transferencia de datos entre localidades de memoria, dispositivos de E/S y registros de propósito general en cualquier combinación.

La instrucción fundamental de este grupo es la instrucción MOV (mueve datos) que es usada para efectuar cualquiera de las transferencias mencionadas anteriormente, puede operar en bytes, palabras y palabras largas, con la característica especial de que puede operar entre 2 localidades de memoria directamente. Esta instrucción es usada para manejar pilas en el procesador (incluyendo las Pilas de Modo Supervisor y Modo Usuario), para realizar esto se emplea el Modo de Direccionamiento de Registro Indirecto con Predecremento. Existe también la Instrucción MOVQ (mueve rápido) la cual permite que una constante pequeña de 8 bits sea especificada dentro de la palabra de código de operación.

Hay otra instrucción de movimiento de datos la cual es MOVM (mueve múltiple) y que permite que varios Registros sean introducidos en la Pila con una sola instrucción, permite también que varios Registros sean cargados desde la Pila. Esta instrucción es bastante útil en el manejo de subrutinas ya que permite salvar el contenido de varios Registros de Propósito general antes de ejecutar la subrutina, también permite que se puedan tener subrutinas reentrantes (que se pueden llamar a si mismas).

Para la comunicación con periféricos síncronos de la familia anterior 6800 existe la instrucción MOVEP, ésta transfiere datos entre un periférico de 8 bits y registros del 68000 en "paquetes" de 2 o 4 bytes. Los periféricos deberán estar conectados a la parte alta o baja del canal y, dependiendo de esto, la instrucción MOVEP colocará direcciones pares (parte alta del Canal) o direcciones impares (parte baja del Canal) para comunicarse con los periféricos.

Este grupo comprende también las instrucciones SWAP y EXG las cuales intercambian ya sea las mitades de un Registro o localidad de memoria (SWAP) o el contenido de dos Registros de Datos o Direcciones (EXG).

2.-INSTRUCCIONES ARITMETICAS EN NUMEROS ENTEROS.- El 68000 puede sumar, restar, multiplicar, dividir y comparar dos operandos binarios. Puede también borrar, probar, extender en signo y negar (complementar a 2) un operando sencillo.

INSTRUCCIONES DE SUMA.-Hay 5 instrucciones que permiten sumar números binarios. La primera de ellas, ADD (suma binaria) suma dos operandos que pueden ser bytes, palabras o

palabras largas. Debido a que los operandos son asumidos como datos, uno de ellos debe estar en un Registro de Datos y el otro puede estar en una localidad de memoria, en un Registro de Dirección (excepto para bytes) o en otro Registro de Datos. Los códigos de Status son afectados de acuerdo al resultado de la operación excepto en el caso de que el destino de ésta sea un Registro de Direcciones. Otras instrucciones de suma son: las instrucciones ADDX (suma extendida) la cual permite efectuar sumas con precisión múltiple y además permite que ambos operandos residan en memoria; las instrucciones ADDI (suma inmediata) y ADDQ (suma rápida) son usadas para sumar constantes al operando direccionado. En el caso de ADDI la constante puede ser un byte, una palabra o una palabra larga, por lo que la instrucción ocupa de dos a cinco localidades de memoria, esta instrucción no puede ser usada sobre Registros de Direcciones. En la instrucción ADDQ la constante sólo puede tener un valor entre 1 y 8 pero la instrucción ocupa sólo de una a tres localidades de memoria, además esta instrucción puede operar sobre Registros de Direcciones. También existe la instrucción ADDA que suma dos Registros de Direcciones sin afectar las banderas de Status.

INSTRUCCIONES DE SUBSTRACCION.-El 68000 tiene un equivalente en substracción para cada una de las operaciones de suma las que funcionan de la misma manera que éstas.

INSTRUCCIONES DE NEGACION.- Dos instrucciones permiten complementar a 2 un byte, una palabra o una palabra larga en memoria o en un Registro de Datos. Estas instrucciones NEG (niega) y NEGX (niega extendido) obtienen el complemento a 2 del operando substrayéndolo de cero.

INSTRUCCIONES DE MULTIPLICACION Y DIVISION.- El 68000 tiene 2 instrucciones para multiplicación MULS (multiplicación signada) y MULU (multiplicación no signada). Estas instrucciones multiplican dos operandos palabra y regresan el producto de 32 bits en un Registro de Datos.

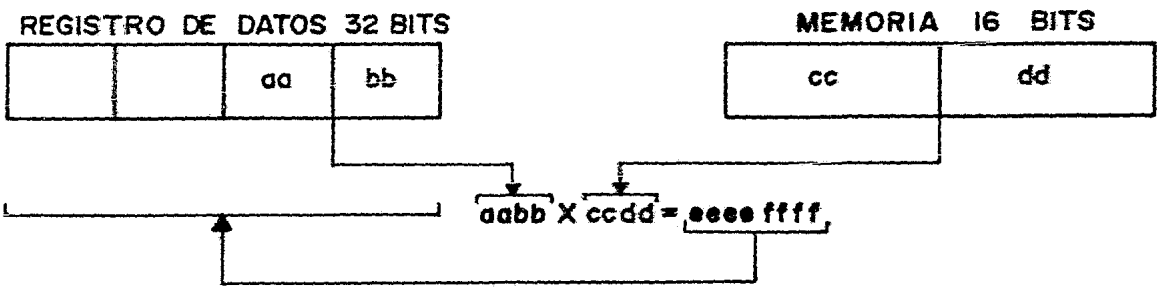


FIG 6IA MULTIPLICACION

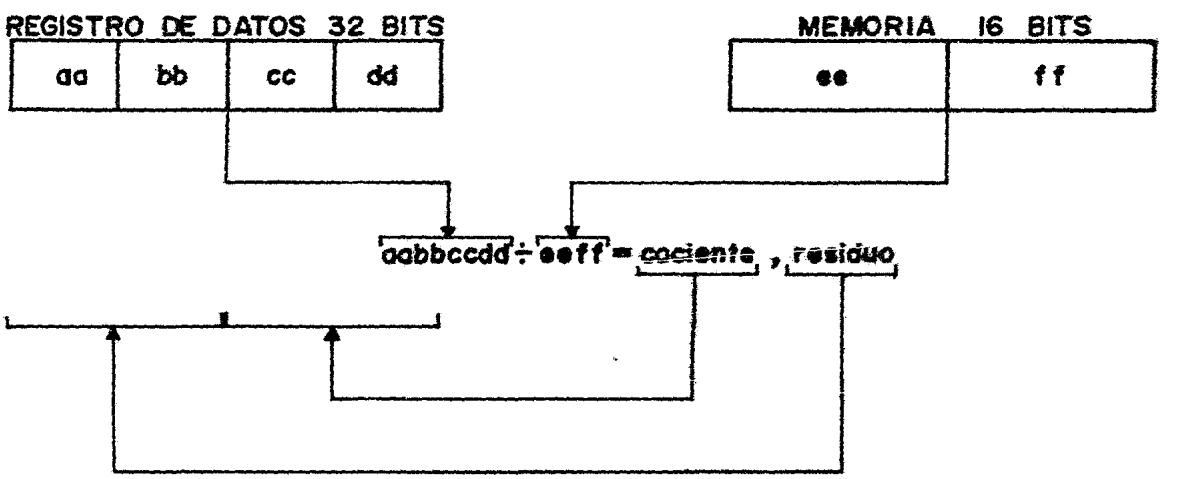


FIG 6IB DIVISION

El 68000 cuenta también con dos instrucciones de división DIVS (división signada) y DIVU (división no signada), éstas dividen un operando de 32 bits entre uno de 16 bits regresando el cociente y el residuo de 16 bits en la parte baja y alta de un Registro de Datos respectivamente. En el caso de que se intente una división entre cero el procesador generará una trampa.

INSTRUCCIONES DE COMPARACION.- Hay cuatro instrucciones básicas de comparación las cuales operan de la misma manera que las operaciones de substracción con la diferencia de que

no salvan el resultado de ésta pero si afectan las banderas de Status del procesador. Las cuatro instrucciones de comparación son: CMP (compara), que compara el operando fuente con un byte, palabra o palabra larga en un Registro de Datos. La segunda instrucción CMPA (compara direcciones) es usada para comparar una palabra o una palabra larga con un Registro de Direcciones. La tercera instrucción CMPI (compara inmediato) compara un valor inmediato ya sea byte, palabra o palabra larga con el operando destino. La cuarta instrucción CMPM (compara memoria) es usada para comparar dos operandos en memoria usando Direccionamiento de Registro Indirecto con Postincremento, esta instrucción en particular es muy útil para comparar cadenas de datos.

Existen otras dos instrucciones de comparación que son TST (compara con cero) que compara un operando con cero y coloca las banderas de condición sin salvar el resultado. La otra instrucción de comparación es la instrucción TAS (prueba y enciende) que efectúa la misma operación básica que la instrucción TST pero también enciende incondicionalmente el bit más significativo del operando el cual deberá ser un byte.

3.- INSTRUCCIONES LOGICAS.- Existen siete instrucciones lógicas. Las instrucciones básicas de este grupo son: AND ("Y" lógico), EOR ("O" exclusivo lógico), OR ("O" inclusive lógico). Estas instrucciones pueden operar en bytes, palabras y palabras largas uno de los cuales deberá estar en un Registro de Datos. El segundo operando puede estar en memoria, en un Registro de Datos, o en un Registro de Direcciones (sólo AND y OR ya que EOR no puede operar sobre un Registro de Direcciones).

Otra instrucción, NOT (complemento lógico) puede ser usada para complementar a 1 un Registro de Datos o una localidad de memoria.

Variaciones de las instrucciones AND, OR y EOR permiten que una constante sea usada como el dato fuente. Estas variaciones son ANDI ("Y" inmediato), EORI ("O" exclusivo inmediato) y ORI ("O" inmediato), pueden operar sobre un Registro de Datos o una localidad de memoria de cualquier tamaño y también pueden ser usadas para operar sobre el Registro de Status o los Códigos de Condición. Las operaciones sobre el Registro de Status son privilegiadas.

4.-INSTRUCCIONES DE ROTACION Y CORRIMIENTO.- El 68000 tiene 4 instrucciones de rotación y 4 instrucciones de corrimiento, cada una de ellas tiene 3 variantes, 2 para operar en Registros de Datos (sobre bytes, palabras y palabras largas) y una para operar en memoria (sólo palabras). Si la operación se efectúa sobre un Registro de

Datos, el contec de Rotación o Corrimiento puede ser especificado como el contenido de otro Registro de Datos (cuenta de 0 a 63 donde 0 produce una cuenta de 64) o como un valor inmediato entre 1 y 8. Si el operando palabra se encuentra en memoria, entonces sólo podrá ser corrido o rotado una posición a la vez.

INSTRUCCIONES DE CORRIMIENTO.- Los números signados pueden ser corridos usando las instrucciones ASL (corrimiento aritmético a la izquierda) y ASR (corrimiento aritmético a la derecha). ASR preserva el signo del operando reaplicando el bit de signo del operando a través de toda la operación de corrimiento. Para ASL, el signo del operando no es conservado, pero la bandera de sobrepaso (V) es encendida si el bit de signo es cambiado.

Los números no signados pueden ser corridos usando las instrucciones LSL (corrimiento lógico a la izquierda) y LSR (corrimiento lógico a la derecha). Para las cuatro operaciones de corrimiento, los bits que son corridos fuera del operando son colocados en las banderas ACARREO (C) y EXTENDIDO (X).

INSTRUCCIONES DE ROTACION.- Las cuatro operaciones de rotación del 68000 son: ROL (rotación a la izquierda), ROR (rotación a la derecha), ROXL (rotación con extensión a la izquierda) y ROXR (rotación con extensión a la derecha). En estas operaciones, los bits desplazados fuera del operando son colocados en la Bandera ACARREO (C). Sin embargo, en las instrucciones ROL y ROR el bit desplazado fuera de un extremo del operando es introducido en el extremo opuesto de éste. Con las instrucciones ROXL y ROXR, el bit desplazado fuera de un extremo del operando es puesto en la bandera EXTENDIDO (X) así como en ACARREO (C) y el valor anterior de la bandera EXTENDIDO (X) es introducido en el extremo opuesto del operando.

5.- INSTRUCCIONES DE MANEJO DE BITS.- Hay cuatro instrucciones que prueban el estado de un bit específico en un Registro de Datos o en un Byte de memoria. Estas instrucciones guardan el estado del bit especificado en la bandera ZERO (Z), si el bit probado es cero entonces Z=1. Si el bit probado es uno entonces Z=0.

Tres de las instrucciones de prueba de bits también cambian el estado de éste incondicionalmente después de la prueba como sigue: BSET (prueba y enciende), el bit es probado y encendido después de la prueba; BCLR (prueba y apaga), el bit es probado y apagado después de la prueba; BCHG (prueba e invierte), el bit es probado e invertido después de la prueba.

La instrucción BTST (prueba bit) sólo prueba el estado del bit especificado sin alterarlo.

El número de bit puede ser especificado como el contenido de un Registro de Datos o como un valor inmediato. Para ambos casos, si se prueba un bit en un Registro de Datos, el número de bit puede ser de 0 a 31 y si se prueba un bit en memoria el número de bit puede ser de 0 a 7.

6.-INSTRUCCIONES PARA MANEJO DE DECIMAL CODIFICADO EN BINARIO (BCD).- Además de las operaciones aritméticas binarias explicadas anteriormente, el procesador tiene tres instrucciones que pueden ser usadas para operar en datos decimales codificados en binario (BCD). Estas tres instrucciones operan sólo en bytes, donde cada byte contiene dos dígitos BCD de 4 bits cada uno. Estas instrucciones utilizan la Bandera EXTENDIDO (X) y la bandera CERO (Z) sólo es alterada por resultados diferentes a cero.

Las instrucciones para manejo de datos BCD son las siguientes: ABCD (suma BCD con extensión), SBCD (resta BCD con extensión), estas pueden efectuar sumas y substracciones en los bytes menos significativos de dos Registros de Datos o en dos bytes en memoria y NBCD (niega BCD con extensión) la cual substrae el byte operando direccionado (contenido en un Registro de Datos o en Memoria) y la Bandera EXTENSION (X) de cero. Si X está apagada, genera el complemento a 10 si está encendida entonces genera el complemento a 9.

7.-INSTRUCCIONES DE CONTROL DE PROGRAMA.- Estas instrucciones son las que pueden transferir la ejecución del programa de una parte de memoria a otra. Se pueden dividir en tres catogorías: Instrucciones Condicionales, Instrucciones No Condicionales e Instrucciones de Retorno.

INSTRUCCIONES CONDICIONALES.- Este grupo está formado por tres instrucciones básicas: Bcc (Salto Condicional), DBcc (Decremento Condicional) y Scc (Enciende de Acuerdo a la Condición). En los Mnemónicos descritos "cc" significa un código de condición que puede ser uno de los siguientes:

SUFIJO	CONDICION	VERDADERO SI
EQ	IGUAL A	Z=1
NE	NO IGUAL A	Z=0
MI	MENOS	N=1
PL	MAS	N=0
GT	MAYOR QUE *	Z or (N xor V) =0
LT	MENOR QUE *	N xor V =1
GE	MAYOR O IGUAL QUE *	N xor V =0

SUFIJO	CONDICION	VERDADERO SI
LE	MENOR O IGUAL QUE *	Z or (N xor V) =0
HI	MAYOR O IGUAL QUE	C and Z =0
LS	MENOR O IGUAL QUE	C or Z =1
CS	ACARREO ENCENDIDO	C=1
CC	ACARREO APAGADO	C=0
VS	SOBREFLUJO *	V=1
VC	NO SOBREFLUJO *	V=0
T	SIEMPRE VERDADERO	
F	SIEMPRE FALSO	

* SOLO EN ARITMETICA EN COMPLEMENTO A DOS

TABLA DE CODIGOS DE CONDICION

En estas instrucciones, si el código de condición se cumple, se realiza un salto a otra parte del programa localizada en la dirección (PC) + desplazamiento, de otra manera, la ejecución continúa con la siguiente instrucción secuencial del programa. El valor del Contador de Programa es la dirección de la instrucción Bcc más dos. El desplazamiento es un número entero en complemento a 2 que especifica el número de bytes entre el valor del Contador de Programa y la localización de la nueva instrucción. Las instrucciones Bcc pueden ser de una o dos palabras de largo.

INSTRUCCIONES INCONDICIONALES.- Hay cuatro instrucciones que provocan saltos incondicionales, dos de ellas son de salto y las otras dos son de llamada de subrutina.

Las instrucciones de salto son: JMP (salta) y BRA (salto corto incondicional), ambas causan un salto incondicional a la localidad especificada, la diferencia entre ellas es que para la instrucción JMP el salto puede ser a cualquier parte de la memoria de 16 Mbytes y para la instrucción BRA el salto está limitado a un desplazamiento de 8 o 16 bits.

Las instrucciones de llamada de subrutina son: JSR (Brinca a subrutina) y BSR (Salto corto a subrutina), la diferencia entre ellas es la misma que en el caso de las instrucciones de salto y la diferencia de ambas con las instrucciones de salto es que en el caso de las llamadas de subrutina la dirección de la instrucción siguiente a la llamada de subrutina es almacenada en la Pila para retornar posteriormente a ejecutar esa instrucción después de haber ejecutado la subrutina. Es de hacer notar que en las operaciones de llamada de subrutina, a diferencia de todas

las demás operaciones de Pila, la dirección es guardada en la Pila poniendo primero la parte alta de la dirección de retorno y luego la parte baja.

INSTRUCCIONES DE RETORNO.- La instrucción RTS (retorno de subrutina) extrae la dirección de retorno de la Pila y la carga en el Contador de Programa. Hay otra instrucción de retorno de subrutina RTR (retorno de subrutina y restaura códigos de condición) la cual además de extraer la dirección de retorno extrae también de la Pila la palabra de Códigos de Condición del Procesador que deberá ser salvada en la Pila antes de la llamada de subrutina para poder usar la instrucción RSR.

INSTRUCCIONES LINK Y UNLINK.- Estas dos instrucciones sirven para asignar y desasignar áreas de datos en la Pila de Sistema para subrutinas anidadas, listas ligadas y otros procedimientos. Después de una llamada a un procedimiento, la instrucción LINK coloca un apuntador de Registro de Direcciones al área de Datos y mueve el Apuntador de Pila hacia abajo en la memoria. Después de completar la ejecución de la subrutina, la instrucción ULNK invierte esta secuencia, restaurando el Apuntador de Pila y los Registros de Direcciones a su estado inicial.

8.-INSTRUCCIONES DE CONTROL DE SISTEMA.- Hay tres tipos de instrucciones de Control de Sistema:

INSTRUCCIONES PRIVILEGIADAS.- Estas instrucciones son aquellas que sólo pueden ser ejecutadas en modo supervisor. Cualquier intento de ejecutar estas instrucciones en modo de usuario causará que ocurra una excepción.

Las instrucciones privilegiadas son las siguientes: RESET (restaura dispositivos externos) es usada para restaurar dispositivos externos a través de la línea RESET del procesador; RTE (retorno de excepcion) causa que el valor del Contador de Programa y del Registro de Status que fueron almacenados en la Pila de Sistema antes de la ejecución de la excepción sean recargados en los Registros respectivos; STOP (alto) causa que el procesador se detenga, las únicas maneras de hacer que salga de este estado son una interrupción con prioridad mayor a la indicada en la Máscara de Prioridad de Interrupción o una restauración (RESET) externa; ORI al SR ("O" con el Registro de Status) realiza una operación "O" entre un dato inmediato y el Registro de Status; MOVE USP (mueve el Apuntador de Pila de Modo Usuario) esta instrucción mueve el contenido del Apuntador de Pila de Modo Usuario desde y hacia un Registro de Direcciones; ANDI al SR ("Y" con el Registro de Status) esta instrucción efectúa una operación "Y" entre un dato inmediato y el Registro de Status; EORI al SR ("O" exclusivo

con el Registro de Status) esta instrucción efectúa una operación "O" exclusiva entre un dato inmediato y el Registro de Status; MOV EA al SR (mueve al Registro de Status) esta instrucción carga un Operando fuente en el Registro de Status.

INSTRUCCIONES DE GENERACION DE TRAMPAS.- Hay tres instrucciones que generan Trampas controladas por programa en el procesador: TRAP (Trampa) la que inicia una operación de trampa incondicional y proporciona un número de vector (de 0 a 15) en el operando, con esto, TRAP puede ser usada para generar 16 diferentes interrupciones de programa; la segunda instrucción es TRAPV (trampa en sobrepaso) que checa la bandera de sobrepaso en el Registro de Status y si está encendida entonces brinca a una localidad de memoria especificada por el vector de trampa en sobrepaso; la tercera instrucción que causa la ocurrencia de una trampa es CHK (checa Registro contra límites), ésta opera condicionalmente, checa el contenido de un Registro de Datos y salta a una localidad específica de memoria apuntada por un vector si el contenido del Registro es menor que cero o si su contenido es mayor que un operando direccionado el cual es el "límite superior".

2.4 COMPARACION DEL CONJUNTO DE INSTRUCCIONES

2.4.1 La comparación del Conjunto de Instrucciones será realizada en base a los siguientes criterios:

1. Modos de Direccionamiento
2. Tipos de Instrucciones
3. Tamaño de los Operandos
4. Instrucciones Especiales de cada Procesador

2.4.2 MODOS DE DIRECCIONAMIENTO.- En este aspecto los tres procesadores tienen Modos de Direccionamiento similares, aunque EN EL CASO DEL 68000 ESTAN DISPONIBLES EN LA MAYORIA DE LAS INSTRUCCIONES, en el 8086 y Z8000 los modos no son tan generales. Por ejemplo, en Z8000 y 8086 el Modo con Autoincremento y Autodecremento sólo es utilizado con las instrucciones para manejo de series de caracteres y carga múltiple de Registros . En el 68000 estos modos de direccionamiento pueden ser utilizados con casi todas las instrucciones.

Evaluación:	68000 E
	Z8000 B
	8086 B

2.4.3 INSTRUCCIONES PARA MOVIMIENTO DE DATOS.- En este grupo, el procesador más versatil es el 68000 ya que todas sus instrucciones manejan Bytes, Palabras y Palabras Largas y no tiene casos particulares en los modos de direccionamiento. En el Z8000 y 8086 existen casos particulares en cuanto a tipo de Datos y Modos de Direccionamiento. Por ejemplo, el Z8000 sólo puede efectuar movimientos de bytes y palabras de memoria a memoria con las instrucciones de manejo de series de caracteres. La desventaja de utilizar este modo es que se requiere de la utilización de un Registro como contador. En el 8086 se pueden efectuar movimientos de memoria a memoria sólo en Bytes o Palabras. Los demás tipos de transferencias son iguales en los tres procesadores.

En cuanto a tipos de instrucciones, Z8000 y 68000 tienen instrucciones de carga múltiple de Registros (LDM y MOVEM respectivamente); Z8000 y 8086 tienen instrucciones para movimiento de series de caracteres (LDI y MOVS respectivamente) las cuales permiten mover una serie de caracteres de hasta 64 Kbytes de una localidad de memoria a otra con una sola instrucción.

Evaluación:	68000 E
	Z8000 B
	8086 B

2.4.4 OPERACIONES ARITMETICAS.- En los tres procesadores se pueden manejar bytes y palabras en las cuatro operaciones aritméticas con operandos signados en complemento a dos. En Z8000 y 68000 también se pueden manejar palabras largas en Suma y Substracción. En Multiplicación y División sólo el Z8000 maneja Palabras Largas (32 Bits). Los tres procesadores tienen la restricción de que al menos uno de los operandos deberá estar en uno de los Registros de Propósito General. El 68000 cuenta con instrucciones especiales de Suma y Substracción para el cálculo de direcciones, estas instrucciones tienen la característica de que al efectuar la operación no afectan las Banderas de Status del ALU, esto permite que el cálculo de direcciones pueda ser efectuado sin problemas cuando se tiene un programa en el cual el Status de las banderas es importante.

Evaluación:	68000 B
	Z8000 E
	8086 R

En las operaciones de comparación, los tres procesadores tienen instrucciones equivalentes con la excepción de que el 8086 no puede manejar palabras largas y el 68000 no tiene operaciones de comparación de series de caracteres.

Evaluación:	68000 R
	Z8000 E
	8086 B

Para manejo de dígitos BCD los tres procesadores son equivalentes, en el caso de manejo de caracteres ASCII el único que cuenta con instrucciones específicas es el 8086. Las instrucciones de extensión de signo son equivalentes en los tres procesadores.

Evaluación:	68000 B
	Z8000 B
	8086 E

2.4.5 INSTRUCCIONES LOGICAS.- Las operaciones OR, AND, XOR y Complemento Lógico funcionan de manera similar en los tres procesadores, los tres pueden manejar bytes o palabras en registros o en memoria. Sólo el 68000 puede efectuar operaciones lógicas con palabras largas. La única restricción es que el 68000 no puede efectuar la operación XOR sobre un Registro de Direcciones. Los tres procesadores pueden comparar un operando contra cero y los tres pueden

obtener complemento lógico de un operando.

Evaluación:	68000	E
	Z8000	B
	8086	B

2.4.6 INSTRUCCIONES DE MANEJO DE BITS.- En lo que se refiere a instrucciones de prueba de Bits individuales los tres procesadores tienen instrucciones similares, con la diferencia de que el 68000 puede operar sobre palabras largas. En cuanto a instrucciones que permitan encender o apagar bits individuales el 8086 no cuenta con éstas. Estas operaciones deberán ser efectuadas a través de las instrucciones AND y OR. En Z8000 y 68000 si existen instrucciones que permiten efectuar estas operaciones directamente y son similares en ambos procesadores con la diferencia que el 68000 puede operar sobre palabras largas.

Evaluación:	68000	E
	Z8000	B
	8086	R

2.4.7 INSTRUCCIONES DE CORRIMIENTO Y ROTACION.- En lo que se refiere a instrucciones de corrimiento los tres procesadores manejan instrucciones similares, con la diferencia de que el Z8000 sólo puede operar sobre registros de propósito general, y el 68000 puede manejar operandos hasta de 32 Bits en Registros de Datos, en memoria sólo puede operar sobre palabras corriendo una sola posición a la vez.

En las instrucciones de Rotación, en el 8086 y 68000 se puede efectuar una rotación cuantas veces se desee. En el Z8000 las rotaciones tienen la restricción de que sólo se pueden efectuar hasta un máximo de dos posiciones a la vez y sólo sobre registros. Para el 68000 las restricciones son las mismas que en el caso de las instrucciones de corrimiento.

Evaluación:	68000	E
	Z8000	R
	8086	B

2.4.8 INSTRUCCIONES DE CONTROL DE PROGRAMA.- En este aspecto el procesador más versátil es el Z8000 ya que maneja saltos condicionales o no condicionales largos y cortos usando varios modos de direccionamiento. Además, maneja Llamadas de Subrutina con varios modos de direccionamiento y Retornos de Subrutina condicionales o no condicionales. El 8086 cuenta con instrucciones equivalentes a las del Z8000 con la diferencia de que no maneja retornos de subrutina condicionales. El 68000 sólo maneja saltos condicionales usando direccionamiento corto, los saltos largos siempre son

incondicionales. Además, las llamadas de subrutina siempre son ejecutadas usando direccionamiento relativo al Contador de Programa. En lo que se refiere a retornos de subrutina siempre son incondicionales.

Evaluación:	68000	R
	Z8000	E
	8086	B

2.4.9 INSTRUCCIONES DE ENTRADA SALIDA.- En este aspecto el 68000 no puede ser comparado ya que maneja la Entrada Salida mapeada como memoria y no existen instrucciones específicas de E/S. El manejo de espacio de E/S mapeado como memoria hace que el diseño y manejo de la memoria del sistema sea más complicado aunque tiene la ventaja de que permite utilizar cualquier instrucción de manejo de memoria para operar sobre los puertos de E/S. En lo que respecta a 8086 y Z8000 éste es el más versátil ya que maneja instrucciones con autoincremento o autodecremento y con repetición opcional, con estas instrucciones se tiene la posibilidad de transferir bloques de datos hacia y desde E/S con una sola instrucción utilizando cualquier registro de propósito general. El 8086 sólo maneja instrucciones de E/S normales (IN y OUT) utilizando modos de direccionamiento directo o indirecto, ambas instrucciones utilizan al acumulador AX como registro de transferencia implícito, el registro DX es usado como apuntador implícito cuando se utiliza direccionamiento indirecto.

Evaluación:	68000	N/A
	Z8000	E
	8086	B

2.4.10 INSTRUCCIONES DE TRADUCCION.- El 68000 no tiene ningún tipo de instrucciones de traducción. En el Z8000 y 8086, estas instrucciones son bastante similares, ambos manejan solamente Bytes y utilizan los elementos como apuntadores para la tabla de traducción usando direccionamiento indirecto.

Evaluación:	68000	R
	Z8000	E
	8086	B

CAPITULO III

PROGRAMAS DE PRUEBA A NIVEL ENSAMBLADOR

Los siguientes programas se desarrollaron con el objeto de manejar el conjunto de instrucciones de cada Procesador y aplicarlo en varios algoritmos. Esto permite evaluar los Tiempos de Ejecución así como el número de Bytes que ocupa cada procesador en la realización de cada uno de los algoritmos. Los programas fueron hechos en base a un diagrama de flujo común pero tratando de obtener el código más eficiente posible para cada procesador. Debido a esto, se podrá notar en los listados respectivos que existen algunas diferencias entre los programas. Esto fué hecho con el objeto de optimizar el código de los programas para cada procesador, utilizando en algunos casos instrucciones que no están disponibles en los otros procesadores.

Los algoritmos de suma, resta, multiplicación y división de 32 bits fueron escogidos con el objeto de evaluar las capacidades de procesamiento numérico y de manejo de bits de cada procesador. Fueron obtenidos de las referencias (11) y (12). Los algoritmos de Reacomodo y Búsqueda se escogieron con el objeto de evaluar las capacidades de manejo de datos en memoria y fueron obtenidos de la referencia (13) de la Bibliografía.

Las capacidades de manejo de interrupciones y trampas no pudieron ser evaluadas debido a que los sistemas en que se desarrollaron los programas no tienen la opción de manejar interrupciones externas creadas por el operador.

Los programas para 8086, Z8000 y 68000 fueron desarrollados respectivamente en los siguientes sistemas: Equipo Seattle Computer Products, Emulador Zilog ZSCAN Z8000 y Sistema Alpha Micro Modelo AM1000/V.

3.1 SUMA DE DOS NUMEROS.- Este programa suma dos números signados de 32 Bits que están representados en magnitud y signo, el signo es el valor del Bit más significativo del operando.

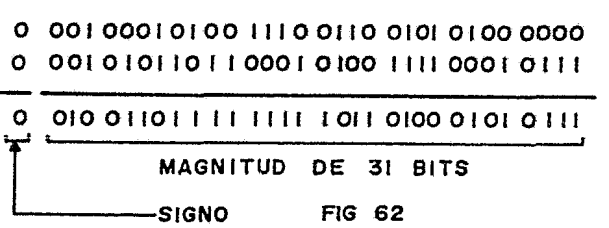
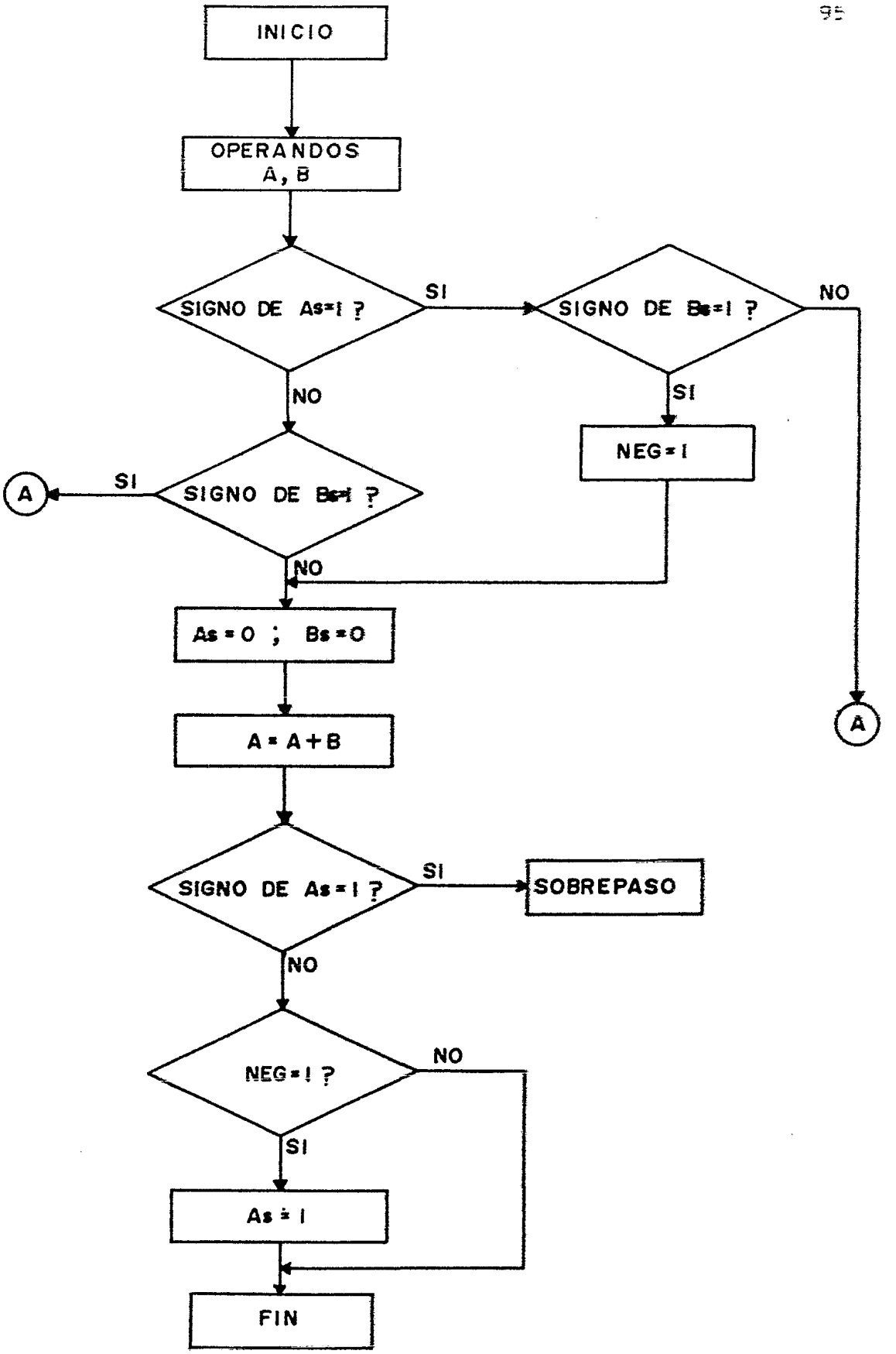
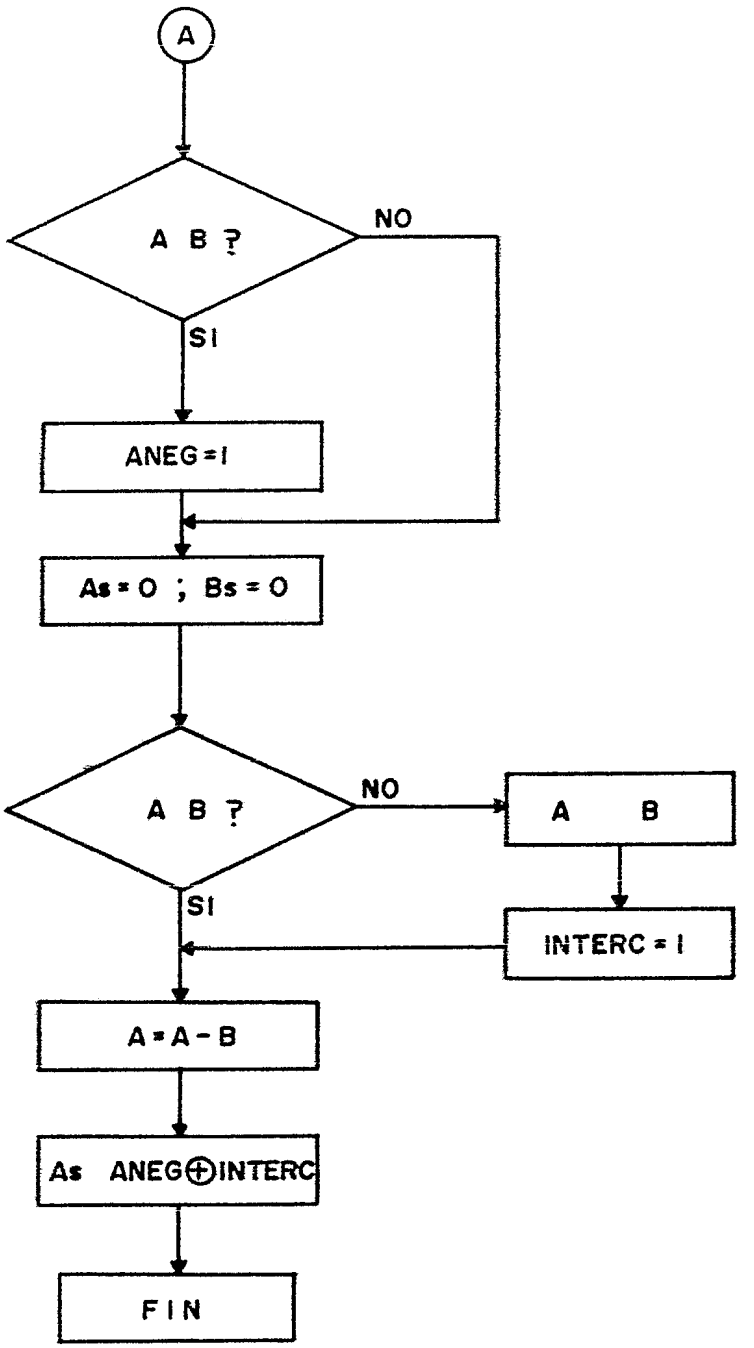


FIG 62

3.1.1 DESCRIPCION DEL DIAGRAMA DE FLUJO.- El programa de Suma está hecho de acuerdo al algoritmo representado en el siguiente diagrama de flujo, el cual obtiene los

operandos de localidades de memoria consecutivas, determina sus signos y decendiendo de éstos se traslada a la Rutina de signos iguales o a la de signos diferentes. En la rutina de Signos Iguales los Bits de signo de los operandos son probados para ver si ambos son negativos y después son eliminados, inmediatamente después se efectúa la operación y en base al estado de las banderas se obtiene el signo del resultado. En el caso de signos diferentes se comparan magnitudes para determinar cual operando es negativo, se eliminan los Bits de signo y nuevamente se efectúa una comparación para determinar cual de los operandos es mayor en magnitud, se efectúa la operación y se asigna al resultado el signo del operando con mayor magnitud.


INICIO
OPERANDOS A, B
SIGNO DE As=1 ?
SI
SIGNO DE Bs=1 ?
NO
SI
NEG=1
NO
A
SI
SIGNO DE Bs=1 ?
NO
As = 0 ; Bs = 0
A = A+B
SIGNO DE As=1 ?
SI
SOBREPASO
NO
NEG=1 ?
NO
SI
As = 1
FIN
A


A
A B ?
NO
SI
ANEG=1
As=0 ; Bs=0
A B ?
NO
A B
SI
INTERC=1
A=A-B
As ANEG⊕INTERC
FIN

```
;*************************************************************
;*                                                           *
;*  PROGRAMA PARA SUMA DE DOS NUMEROS DE 32 BITS EN 8086     *
;*                                                           *
;*************************************************************
;
;Este programa suma dos numeros de 32 bits representados
;como una magnitud de 31 bits y un bit de signo (bit 31),
;entrega el resultado en la localidad de memoria apuntada
;por el contenido del registro [BX].
;
;
;Descripcion de Registros:
;[BX] y [BX+2] contienen el primer sumando.
;[BX+4] y [BX+6] contienen el segundo sumando.
;AX y CX son los registros operadores de los sumandos.
;BX es el registro apuntador de memoria.
;DX es el registro de banderas.

        ORG     100H
        PUT     100H
        JMP     INICIO
DIREC:  DW      0,9,0,0,0,8,0
                                                ;BYTES,CICLOS
INICIO:
        MOV     BX,DIREC        ;CARGA DIR. MEM.        3, 4
        MOV     AX,[BX]         ;CARGA SUM1 P. ALTA.    2,10
        MOV     CX,[BX+4]       ;CARGA SUM2 P. ALTA.    3,17
        AND     AX,8000H        ;CHECA SIGNO SUM1.      3, 4
        JS      ANEG            ;SI ES NEG. SALTA.      2,16,4
        AND     CX,8000H        ;CHECA SIG. SUM2.       3, 4
        JS      SIGDIF          ;SALTA SI ES NEG.       2,16,4
        MOV     DX,00H          ;PON BANDERA POSIT.     3, 4
        JMP     SUMA            ;                       3,15
ANEG:   MOV     DX,01H          ;BANDERA SUM1 NEG.      3, 4
        AND     CX,8000H        ;CHECA SIGNO SUM2.      4, 4
        JNS     SIGDIF          ;POS. VE SIG. DIFER.    2,16,4
        MOV     DX,11H          ;BANDERA NEGATIVOS.     3, 4
        AND     AX,7FFFH        ;VALOR ABS. SUM2.       3, 4
        MOV     [BX+4],CX       ;SALVALO.               3,18
SUMA:   MOV     CX,[BX+2]       ;SUMANDO P. BAJA.       3,17
        ADD     CX,[BX+6]       ;SUMA PARTES BAJAS.     3,18
        MOV     [BX+2],CX       ;SALVA RESULTADO.       3,18
        MOV     AX,[BX]         ;CARGA SUM1 P. ALTA.    2,10
        ADC     AX,[BX+4]       ;SUMA PARTES ALTAS.     3,18
        MOV     [BX],AX         ;SALVA RESULTADO.       2,10
        CMP     AX,7FFFH        ;HAY SOBREFLUJO ?       3, 4
        JNL     ORVERF          ;SALTA SI HAY.          2,16,4
        CMP     DX,11H          ;CHECA BAND. NEG.       3, 4
        JNE     FIN             ;SALTA SI ES POSIT.     2,16,4
        OR      AX,8000H        ;HAZ RESULT. NEGAT.     3, 4
        MOV     [BX],AX         ;SALVALO.               2,10
```

```
FIN:     INT     20H             ;FIN.                          2,51
SIGDIF:  AND     [BX],7FFFH      ;BORRA SIGNO SUM1.             4,22
         AND     [BX+4],7FFFH    ;BORRA SIGNO SUM2.             5,26
         MOV     AX,[BX+2]       ;SUM1 PARTE BAJA.              3,17
         CMP     AX,[BX+6]       ;COMPARA P. BAJAS.             3,18
         JAE     AMB             ;SI A>=B VE RESTAR.            2,16,4
         SUB     [BX+6],AX       ;HAZ B-A P. BAJAS.             3,25
         MOV     CX,[BX+6]       ;PONLO EN REGISTRO.            3,17
         MOV     [BX+2],CX       ;SALVA PARTE BAJA.             3,18
         JMP     PAALTA          ;VE OPER. P. ALTAS.            3,15
AMB:     SUB     AX,[BX+6]       ;HAZ A-B P. BAJA.              3,18
         MOV     [BX+2],AX       ;SALVA PARTE BAJA.             3,18
PAALTA:  MOV     AX,[BX]         ;CARGA P. ALTA SUM1.           2,13
         CMP     AX,[BX+4]       ;COMP. MAG. SUMS.              3,18
         JNL     AMBPA           ;SALTA SI SUM1>=SUM2.          2,16,4
         OR      DX,80H          ;RESULTADO NEG.                2, 4
         SBB     [BX+4],AX       ;SUM2-SUM1 P. ALTA.            3,25
         MOV     CX,[BX+4]       ;RES. P. ALTA EN CX.           3,17
         MOV     [BX],CX         ;SALVA RES. P. ALTA.           2,13
         JMP     SIGRES          ;CHECA SIGNO RES.              3,15
AMBPA:   SBB     AX,[BX+4]       ;SUM1-SUM2 P. ALTA.            3,18
         MOV     [BX],AX         ;SALVA RES. P. ALTA.           2,14
SIGRES:  CMP     DX,00H          ;CHECA BANDERAS.               3, 4
         JZ      FIN             ;CERO, RES. POS. FIN.          2,16,4
         CMP     DX,81H          ;SI 81 RES. POS. FIN.          3, 4
         JZ      FIN             ;                              2,16,4
         OR      [BX],8000H      ;HAZ RES. NEGATIVO.            3,22
         INT     20H             ;FINAL.                        2,51

;ESTE PROGRAMA OCUPA 146 BYTES
;
;SI AMBOS SON POSITIVOS REQUIERE 197 CICLOS.
;SI AMBOS SON NEGATIVOS REQUIERE 222 CICLOS.
;SI A>B P. ALTA Y BAJA CON A POS. Y B NEG. REQUIERE 293 CICLOS.
;SI A>B P. ALTA Y BAJA CON A NEG. Y B POS. REQUIERE 309 CICLOS.
;SI A>B P. ALTA Y A<B P. BAJA, A POS. Y B NEG. REQUIERE 320 CIC.
;SI A>B P. ALTA Y A<B P. BAJA, A NEG. Y B POS. REQUIERE 344 CIC.
;SI A<B P. ALTA Y A>B P. BAJA, A POS. Y B NEG. REQUIERE 331 CIC.
;SI A<B P. ALTA Y A>B P. BAJA, A NEG. Y B POS. REQUIERE 347 CIC.
;SI A<B P. ALTA Y A<B P. BAJA, A POS. Y B NEG. REQUIERE 358 CIC.
;SI A<B P. ALTA Y A<B P. BAJA, A NEG. Y B POS. REQUIERE 396 CIC.
;SI EXISTE SOBREFLUJO REQUIERE 111 CICLOS.

;CICLOS PROMEDIO 311.7

;INSTRUCCIONES 55
```

```
!*********************************************************!
!*                                                       *!
!* PROGRAMA PARA SUMA DE DOS NUMEROS DE 32 BITS EN Z8000 *!
!*                                                       *!
!*********************************************************!

!Este programa suma dos numeros de 32 bits representados !
!como una magnitud de 31 bits y un bit de signo (bit 31),!
!entrega el resultado en RR0 en la misma representacion!

!Descripcion de Registros: !
!RR0=Registro para Primer Sumando!
!RR2=Registro para segundo sumando!
!R4=Apuntador de Operandos!
!R5=Registro de Banderas Como Sigue: !
!Bit 0=Bandera para indicar Ambos Operandos Negativos!
!Bit 1=Bandera para indicar que primer sumando es Negat!
!Bit 2=Bandera para indicar intercambio de Operandos!

SUM MODULE
$NONSEGMENTED
CONSTANT
        ADDR    :=%1500

GLOBAL
        SUMA PROCEDURE

ENTRY

                                                    !CICLOS, BYTES!

        LD      R4,#ADDR        !CARGA DIRECCION SUMANDOS  7,4!
        CLR     R5              !BORRA BANDERAS            7,2!
        LDM     R0,@R4,#4       !CARGA OPERANDOS          26,4!
        INC     R4,#4           !APUNTA SEGUNDO OPERANDO   4,2!
        XOR     R2,R0           !CHECA SIGNOS              4,2!
        BIT     R2,#15          !CHECA SIGNOS              4,2!
        JR      NZ,SIGDIF       !BIT 15 DE R2=1, SIG. DIF. 6,2!
        BIT     R0,#15          !CHECA AMBOS NEGATIVOS     4,2!
        JR      Z,POSIT         !NO, VE A SUMARLOS         6,2!
NEGAT:  SET     R5,#0           !SI, PON BANDERA           4,2!
        RES     R0,#15          !BORRA SIGNO PRIMER OPER.  4,2!
        RES     @R4,#15         !BORRA SIGNO SEGUNDO OPER. 4,2!
POSIT:  ADDL    RR0,@R4         !SUMALOS                  14,2!
        BIT     R0,#15          !CHECA SOBREPASO           4,2!
        JR      NZ,OVERF        !SI, RUTINA DE SOBREPASO   6,2!
        BIT     R5,#0           !SIGNO RESULTADO NEGATIVO  4,2!
        JR      Z,FIN           !NO, VE AL FINAL           6,2!
        SET     R0,#15          !SI, HAZ RESULTADO NEGATIVO4,2!
        JR      FIN             !                          6,2!
SIGDIF: CPL     RR0,@R4         !CHECA QUIEN ES EL NEG.   14,2!
        JR      ULT,BNEG        !BRA ENTONCES B=NEGATIVO   6,2!
```

```
          SET     R5,#1             !NO, A=NEGATIVO PON BANDERA4,2!
          RES     R0,#15            !BORRA SIGNO SUMANDO 1      4,2!
BNEG:     RES     @R4,#15           !BORRA SIGNO SUMANDO 2      4,2!
          CPL     RR0,@R4           !COMPARA LAS MAGNITUDES     4,2!
          JR      UGE,RESTA         !SI A>=B VE A RESTARLOS     6,2!
EXCH:     LDL     RR2,@R4           !NO,SEGUNDO SUMANDO EN RR2  1,2!
          SET     R5,#2             !PON BANDERA INTERCAMBIO    4,2!
          SUBL    RR2,RR0           !RESTALOS HACIENDO B-A      8,2!
          LDL     RR0,RR2           !PON RESULTADO EN RR0      11,2!
          JR      SIGRES            !CHECA SIGNO DEL RESULTADO 6,2!
RESTA:    SUBL    RR0,@R4           !RESTALOS HACIENDO A-B     14,2!
SIGRES:   TEST    R5                !SI R5=0,A>=B Y A+          7,2!
          JR      Z,FIN             !SI, VE AL FINAL            6,2!
          CP      R5,#6             !SI R5=6H B>A Y B+          7,4!
          JR      Z,FIN             !SI, RESULTADO POSITIVO     6,2!
          SET     R0,#15            !NO, HAZ RESULTADO NEGAT.   4,2!
          JR      FIN               !VE AL FINAL                6,2!
OVERF:    LD      R0,#%FFFF         !R0=FFFF INDICA SOBREPASO   7,4!
FIN:      HALT
END SUMA
END SUM

!NUMERO DE CICLOS PARA CADA CASO: !
!AMBOS POSITIVOS   98 CICLOS!
!AMBOS NEGATIVOS  120 CICLOS!
!A+ B- A>=B       125 CICLOS!
!A+ B- A<B        174 CICLOS!
!A- B+ A>=B       156 CICLOS!
!A- B+ A<B        172 CICLOS!

!BYTES 86!
```

```
;*************************************************************
;*                                                           *
;*            PROGRAMA DE SUMA PARA 68000                    *
;*                                                           *
;*************************************************************

;     Este programa efectua una suma de dos operandos de 32
;bits signados que se encuentran en localidades de memoria
;consecutivas. El resultado de la suma estara en el registro
;de datos D0.  Se utiliza el registro D3 como registro de
;banderas.

          COPY     SYS

          TYPE     <Direcion inicio memoria :
          KBD
          GTOCT                                        ;CICLOS,BYTES
          MOVL     D1,A0       ;COLOCA DIRECCION DE LOS OP.     4,2
          MOVL     (A0)+,D1    ;CARGA PRIMER SUMANDO            4,2
          MOVL     (A0),D0     ;CARGA EL OTRO SUMANDO           4,2
          CLRB     D3          ;INICIALIZA BANDERA DE SIGNO     4,2
INICIO:   BCLR     #37,D0      ;PRIMER SUMANDO NEGATIVO?       14,4
          BNE      ANEG        ;SI,VE A ENCENDER BANDERA  (12) 10,4
          BCLR     #37,D1      ;NO,SEGUNDO SUMANDO NEGATIVO?   14,4
          BNE      SIGNDIF     ;SI,RUTINA SIG. DIFERENTES (12) 10,4
SIGNI:    ADDL     D1,D0       ;AMBOS POSITIVOS,SUMALOS         8,2
          BTST     #37,D0      ;CHECA SOBREFLUJO               10,4
          BNE      FIN         ;                          (12) 10,4
          BTST     #0,D3       ;CHECA AMBOS SUMANDOS NEGATIVOS 10,4
          BEQ      FIN         ;NO, FIN                   (12) 10,4
          BSET     #37,D0      ;SI,HAZ RESULTADDO NEGATIVO     12,4
          BR       FIN         ;VE AL FINAL                    10,4
ANEG:     ADDB     #1,D3       ;SE PRENDE BANDERA NEGATIVOS     4,2
          BCLR     #37,D1      ;CHECA SEGUNDO SUMANDO NEGATIVO 14,4
          BNE      SIGNI       ;NO,ENTONCES PRIMERO NEGAT (12) 10,4
SIGNDIF:  CMP      D0,D1       ;COMPARA MAGNITUDES              6,2
          BCS      AMB         ;SALTA SI A < B            (12) 10,4
          SUBL     D1,D0       ;SI A ES MAYOR QUE B RESTALOS    8,2
          BTST     #0,D3       ;CHECA A NEGATIVO?              10,4
          BEQ      FIN         ;SI A ES (+), RESUL (+)    (12) 10,4
          BSET     #37,D0      ;NO, RESULTADO NEGATIVO         12,4
          BR       FIN         ;VE AL FIN                      10,4
AMB:      SUBL     D0,D1       ;SI A < B INTERCAMBIA SUMANDOS   8,4
          BTST     #0,D3       ;CHECA SI A ES NEGATIVO         10,4
          BNE      MOVE        ;SI A ES (+),RESULT. NEGAT (12) 10,4
          BSET     #37,D1      ;HAZ RESULTADO NEGATIVO         12,4
MOVE:     MOV      D1,D0       ;PON RESULTADO EN D0             4,2
FIN:      EXIT
          END                  ;FIN
```

```
;NUMERO DE CICLOS PARA CADA CASO:

;                              CICLOS

;AMBOS POSITIVOS                    114
;AMBOS NEGATIVOS              132
;A (+) Y B (-)   A>B          112
;A (+) Y B (-)   A<B          124
;A (-) Y B (+)   A>B          136
;A (-) Y B (+)   A<B          122

;CICLOS PROMEDIO              123.33

;BYTES UTILIZADOS             102

;INTRUCCIONES OCUPADAS         30
```

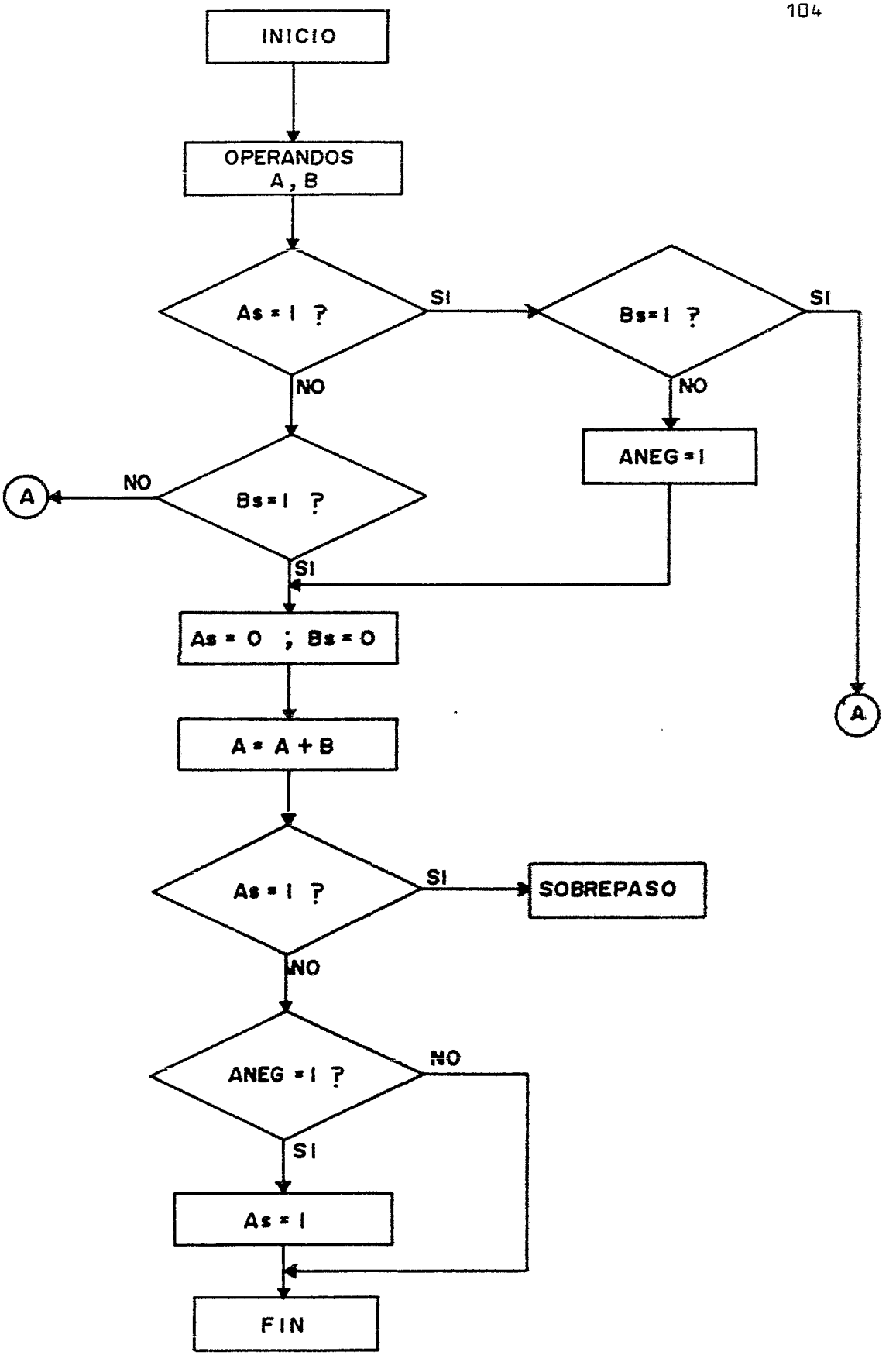
3.1.4 COMENTARIOS.- Como se puede observar en los listados respectivos, el microprocesador que ocupa mayor número de instrucciones en esta rutina es el 8086, esto se debe a que no puede manejar operandos de 32 Bits con una sola instruccion, asimismo, es el que ocupa un mayor número de ciclos de reloj (337.25 en promedio).

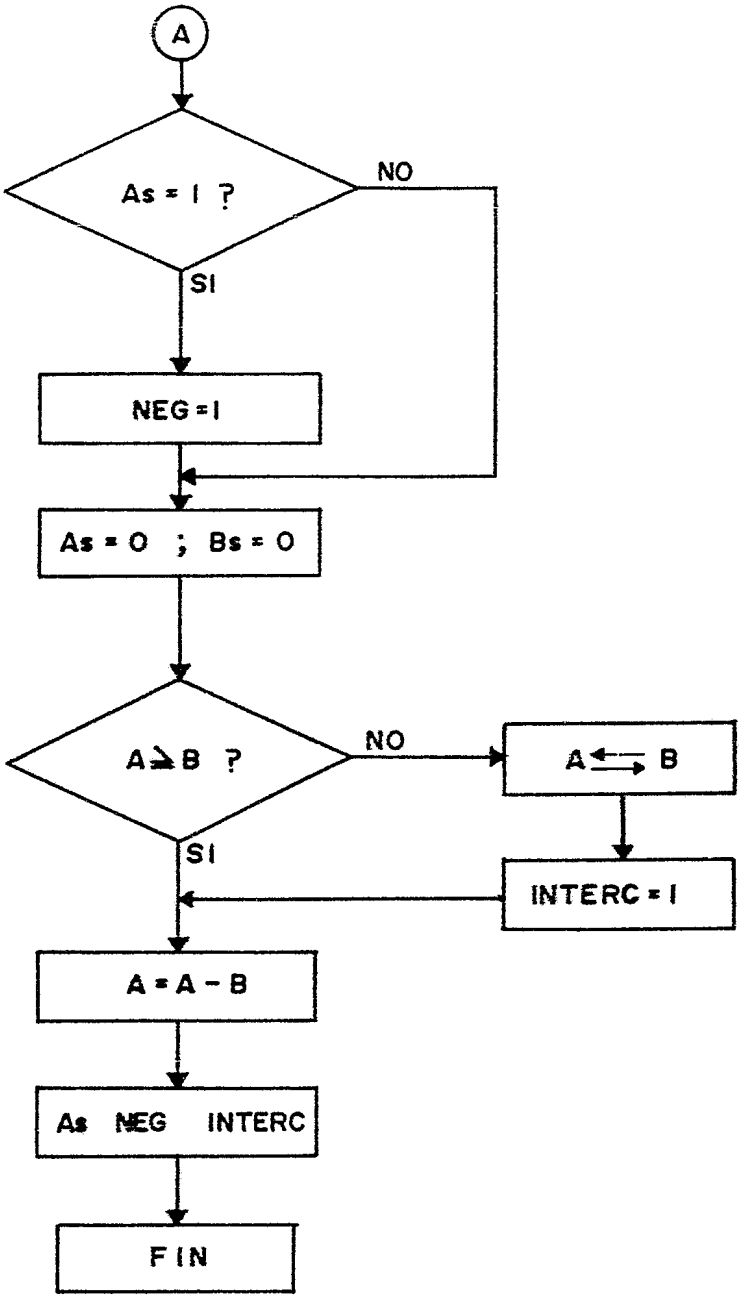
El Z8000 es el que ocupa el menor número de bytes de memoria para ejecutar la rutina (86 Bytes) utilizando 40 instrucciones. Aunque ocupa un mayor número de ciclos de reloj para la ejecución (141 ciclos en promedio).

El 68000 ocupa el menor número de instrucciones (30 instrucciones), pero estas ocupan mayor espacio de memoria que en el Z8000 (102 Bytes), además, este procesador es el mas rápido en la ejecución de la rutina (123.3 ciclos en promedio). La rapidez del 68000 es debida principalmente a su arquitectura interna de 32 Bits, con lo cual las operaciones de suma se efectúan en menor tiempo (8 ciclos de reloj) que en el Z8000 (14 ciclos).

3.2 SUBSTRACCION DE DOS NUMEROS.- Este programa resta dos números de 32 Bits representados en magnitud y signo, el signo del operando es especificado por el Bit más significativo de éstos.

3.2.1 DESCRIPCION DEL DIAGRAMA DE FLUJO.- El diagrama de flujo para la Resta es similar al de Suma, la diferencia básica de este programa es que en la rutina de Signos iguales se efectúa una Substracción y se determina el orden de los operandos según su valor absoluto, en la rutina de signos diferentes, primeramente se detecta cual operando es el negativo, se borran los bits de signo, se suman los operandos y se coloca el signo al resultado de acuerdo a si el minuendo era negativo o no.


INICIO
OPERANDOS
A , B
As = 1 ?
SI
Bs=1 ?
SI
NO
NO
ANEG = 1
A
NO
Bs=1 ?
SI
A
As = 0 ; Bs = 0
A = A + B
As = 1 ?
SI
SOBREPASO
NO
ANEG = 1 ?
NO
SI
As = 1
FIN


A
As = 1 ?
NO
SI
NEG = 1
As = 0 ; Bs = 0
A ≥ B ?
NO
A ⟷ B
SI
INTERC = 1
A = A - B
As NEG INTERC
FIN

```
;*************************************************************
;*                                                           *
;*   PROGRAMA PARA RESTAR DOS NUMEROS DE 32 BITS EN 8086     *
;*                                                           *
;*************************************************************
;
;Este programa resta dos numeros de 32 bits representados
;como una magnitud de 31 bits y un bit de signo (bit 31)
;y entrega el resultado en la localidad de memoria apun-
;tada por el contenido del registro [BX] y [BX+2].
;
;Descripcion de Registros:
;[BX] y [BX+2] contienen el minuendo.
;[BX+4] y [BX+6] contienen el sustraendo.
;AX y CX son los registros operadores.
;BX es el registro apuntador de memoria.
;DX es el registro de banderas.
;
;
        ORG     100H
        PUT     100H
INICIO:                                          ;BYTES,CICLOS
        MOV     BX,DIREC        ;CARGA APUNT. DE MEM.     3, 4
        MOV     DX,0000H        ;INICIAIZA BANDERAS.      3, 4
        MOV     AX,(BX]         ;CARGA MINU. PAR. ALT.    2,10
        MOV     CX,[BX+4]       ;CARGA SUST. PAR. ALT.    3,17
        AND     AX,8000H        ;MINUENDO NEGATIVO ?      3, 4
        JS      ANEG            ;SI, CHECAR SUSTRAENDO.   2,16,4
        AND     CX,8000H        ;CHECA SUST. NEGATIVO?    3, 4
        JNS     COMPAR          ;NO, VE RUTINA POSITI.    2,16,4
        JMP     SIGDIF          ;SI, SIGNOS DIFERENTES    3,15
ANEG:   MOV     DX,8000H        ;PON BAND. MINUE. NEG.    3, 4
        AND     CX,8000H        ;CHECA SIGNO SUSTRAENDO.  3, 4
        JNS     SIGDIF          ;SI ES POSIT. SALTA.      2,16,4
        MOV     DX,0011H        ;PON BAND. AMBOS NEGAT.   3, 4
COMPAR: AND     [BX],7FFFH      ;BORRA SIGNO MINUENDO.    3,23
        AND     [BX+4],7FFFH    ;SALVALO.                 3,26
        MOV     AX,[BX]         ;CARGA MINUE. PAR. ALT.   2,10
        CMP     AX,[BX+4]       ;COMPARA MAGNITUDES.      3,18
        JGE     PAI             ;PARTES ALTAS IGUALES.    2,16,4
        JL      SMM             ;SUST. > MINUE. SALTA.    2,16,4
RESTA:  MOV     AX,[BX+2]       ;CARGA MINUE. PAR. ALTA.  3,17
        SUB     AX,[BX+6]       ;RESTA PARTES BAJAS.      3,18
        MOV     [BX+2],AX       ;SALVA RESULTADO.         3,18
        MOV     AX,[BX]         ;CARGA MINUE. PAR. ALTA.  2,10
        SBB     AX,[BX+4]       ;RESTA PARTES ALTAS.      3,18
        CMP     DX,0000H        ;COMPARA BANDERA DE SIG.  4, 4
        JZ      SIGPOS          ;SALTA A SIGNO POSITIVO.  2,16,4
        CMP     DX,0111H        ;CHECA BAND. SIGN. NEG.   4, 4
        JZ      SIGPOS          ;NO, POSITIVO.       (12) 16,4
        OR      AX,8000H        ;HAZ RESULT. NEGAT.       3, 4
SIGPOS: MOV     [BX],AX         ;SALVA RESULTADO.         2,10
```

```
FIN:     INT     20H             ;FIN.                          2,51
PAI:     MOV     AX,[BX+2]       ;CARGA MINUE. PAR. BAJ.        3,17
         CMP     AX,[BX+6]       ;COMPARA PARTES BAJAS.         3,18
         JNL     RESTA           ;MINUE. > SUTRAE. RESTA.       2,16,4
         MOV     CX,[BX+6]       ;CARGA SUST. PARTE BAJA.       3,17
         MOV     [BX+6],AX       ;INTERCAMBIA OPERANDOS.        3,18
         MOV     [BX+2],CX       ;SALVA OPERANDO.               3,18
         OR      DX,0100H        ;PRENDE BANDERA INTERC.        3, 4
         JMP     RESTA           ;SALTA A RESTA.                3,15
SMM:     MOV     CX,[BX+4]       ;SALVA OPERANDO 2 EN CX.       3,17
         MOV     [BX],CX         ;SALVALO COMO MINUENDO.        3,18
         MOV     [BX+4],AX       ;SALVA OPER1. COMO SUST.       3,18
         OR      DX,0100H        ;PRENDE BAND. INTER.           3, 4
         JMP     PAI             ;VE A PART.ALT. IGUALES.       3,15
SIGDIF:  MOV     CX,[BX+2]       ;CARGA OPER1 PARTE ALTA.       3,17
         ADD     CX,[BX+6]       ;SUMA PARTES BAJAS.            3,18
         MOV     [BX+2],CX       ;SALVA RESULTADO.              3,18
         MOV     AX,[BX]         ;CARGA OPER1 PARTE ALTA.       2,10
         ADC     AX,[BX+4]       ;SUMA PARTES ALTAS.            3,18
         CMP     AX,7FFFH        ;CHECA SOBREPASO.              3, 4
         JNL     OVERF           ;SI, SALTA.                    2,16,4
         OR      AX,DX           ;PON SIGNO RESULTADO.          2, 3
         MOV     [BX],AX         ;SALVA RESULTAD.               2,10
         INT     20H             ;FIN.                          2,51
OVERF:   MOV     [BX],0FFFFH     ;INDICA SOBREPASO.             3,15
         MOV     [BX+2],0FFFFH   ;INDICA SOBREPASO.             4,19
         INT     20H             ;FIN.                          2,51
DIREC:   DB      0,0,0,0,0,0,0,0,0

;ESTE PROGRAMA OCUPA UN TOTAL DE 144 BYTES.
;
;SI A ES POSITIVO Y B ES NEGATIVO SE EJECUTA EN 168 CICLOS.
;SI A ES NEGATIVO Y B ES POSITIVO SE EJECUTA EN 181 CICLOS.
;SI A>B P. ALTA Y BAJA CON AMBOS POSITIVOS REQUIERE 318 CICLOS.
;SI A>B P. ALTA Y BAJA CON AMBOS NEGATIVOS REQUIERE 326 CICLOS.
;SI A>B P. ALTA Y A<B P. BAJA,AMBOS POSITIVOS REQUIERE 382 CIC.
;SI A>B P. ALTA Y A<B P. BAJA,AMBOS NEGATIVOS REQUIERE 390 CIC.
;SI A<B P. ALTA Y A>B P. BAJA,AMBOS POSITIVOS REQUIERE 398 CIC.
;SI A<B P. ALTA Y A>B P. BAJA,AMBOS NEGATIVOS REQUIERE 410 CIC.
;SI A<B P. ALTA Y A<B P. BAJA,AMBOS POSITIVOS REQUIERE 462 CIC.
;SI A<B P. ALTA Y A<B P. BAJA,AMBOS NEGATIVOS REQUIERE 470 CIC.
;SI EXISTE SOBREFLUJO REQUIERE 201 CICLOS PARA A POS. Y B NEG.
;SI EXISTE SOBREFLUJO REQUIERE 214 CICLOS PARA A NEG. Y B POS.

CICLOS PROMEDIO 350.5

INSTRUCCIONES 57
```

```
!*************************************************************!
!*                                                           *!
!* !PROGRAMA PARA RESTAR 2 NUMEROS DE 32 BITS EN Z8000!      *!
!*                                                           *!
!*************************************************************!

!Este programa resta 2 numeros de 32 bits representados    !
!como una magnitud de 31 bits y un bit de signo (bit 31)   !
!y entrega el resultado en RR0 en la misma representacion  !

!Descripcion de Registros:  !
!RR0=Registro para Minuendo!
!RR2=Regisro para Substraendo!
!R4=Apuntador de Operandos!
!R5=Registro para Banderas Como Sigue:  !
!Bit 0=Indica Ambos Operandos son Negativos
!Bit 1=Indica Intercambio de Operandos!
!Bit 2=Indica Minuendo Negativo!

SUBSTRA MODULE
CONSTANT
        ADDR    :=%1500

GLOBAL
        RESTA PROCEDURE

ENTRY

                                                        !CICLOS, BYTES!
                                                                      !
        LD      R4,#ADRR        !CARGA DIR. DE LOS OPERANDOS7,4!
        CLR     R5              !BORRA REGISTRO DE BANDERAS 7,2!
        LDM     R0,@R4,#4       !CARGA OPERANDOS           26,4!
        INC     R4,#4           !APUNTA AL SEGUNDO OPER.    4,2!
        XOR     R2,R0           !CHECA SIGNOS               4,2!
        BIT     R2,#15          !BIT 15 DE R2=1 SIGNOS DIF. 4,2!
        JR      NZ,SIGDIF       !SI, VE A RUTINA SIGNOS DIF 6,2!
        BIT     R0,#15          !CHECA AMBOS SON NEGATIVOS  4,2!
        JR      Z,POSIT         !NO, VE A RUTINA POSITIVOS  6,2!
NEGAT:  SET     R5,#0           !SI, PON BANDERA NEGATIVOS  4,2!
        RES     R0,#15          !BORRA SIGNO MINUENDO       4,2!
        RES     @R4,#15         !BORRA SIGNO SUBSTRAENDO    4,2!
POSIT:  CPL     RR0,@R4         !COMPARA MAGNITUDES        14,2!
        JR      UGE,RE          !SI A>=B VE A RESTARLOS     6,2!
EXCH:   LDL     RR2,@R4         !PON SUBTRAENDO EN RR2     11,2!
        SET     R5,#1           !PON BANDERA DE INTERCAMBIO 4,2!
        SUBL    RR2,RR0         !RESTALOS HACIENDO B-A      8,2!
        LDL     RR0,RR2         !MUEVE RESULTADO            5,2!
        JR      SIGRES          !VE CHECAR SIGNO RESULTADO  6,2!
RE      SUBL    RR0,@R4         !HAZ MINUENDO-SUBSTRAENDO  14,2!
SIGRES: TEST    R5              !R5=0 ENT. A+, B+ Y A>=B    7,2!
```

```
          JR      Z,FIN           !SI, VE AL FINAL                6,2!
          CP      R5,#3           !SI R5=3 ENT.  A-, B- Y B>A 7,4!
          JR      Z,FIN           !SI, VE AL FINAL                6,2!
          SET     R0,#15          !NO, HAZ RESULTADO NEGATIVO 4,2!
          JR      FIN             !VE AL FINAL                    6,2!
SIGDIF:   BIT     R0,#15          !CHECA MINUENDO NEGATIVO    4,2!
          JR      Z,SUMA          !NO, ENTONCES SUMALOS       6,2!
          SET     R5,#2           !SI, PON BANDERA MIN. NEG.  4,2!
          RES     R0,#15          !BORRA SIGNO DEL MINUENDO   4,2!
SUMA:     RES     @R4,#15         !BORRA SIGNO SUBSTRAENDO    4,2!
          ADDL    RR0,@R4         !SUMALOS                   14,2!
          BIT     R0,#15          !CHECA SOBREPASO            4,2!
          JR      NZ,OVERF        !SI, VE A RUTINA SOBREPASO  6,2!
          BIT     R5,#2           !EL MINUENDO ERA NEG. ?     4,2!
          JR      Z,FIN           !NO, VE AL FINAL            6,2!
          SET     R0,#15          !SI, HAZ RESULTADO NEGATIVO 4,2!
          JR      FIN             !VE AL FINAL                    6,2!
OVERF:    LD      R0,#%FFFF       !R0= FFFF INDICA SOBREPASO  7,4!
FIN:      HALT                    !FINAL!
END RESTA
END SUBSTRA

!CICLOS DE RELOJ PARA CADA CASO: !

!AMBOS + Y A>=B 111 CICLOS!
!AMBOS + Y A<B  154 CICLOS!
!AMBOS - Y A>=B 146 CICLOS!
!AMBOS - Y A<B  156 CICLOS!
!A+ B-          102 CICLOS!
!A- B+          120 CICLOS!

!BYTES 86!
```

```
;*************************************************************
;*                                                           *
;*            PROGRAMA DE RESTA PARA 68000                   *
;*                                                           *
;*************************************************************


;      Este  programa  resta  dos numeros signados de 32  bits
;que se encuentran en una localidades de memoria consecutivas.
;El  registro  D1  contendra  el  minuendo  y  el
;registro D0 el sustraendo,el resultado estara en el registro
;D1.  Se utiliza el registro D3 como  banderas.

          COPY     SYS

          TYPE     DIRECCION DE LOS OPERANDOS :
          KBD
          GTOCT                                          ;CICLOS,BYTES
          MOVL     D1,A0            ;COLOCA LA DIR. DE LOS OP.   4,2
          CLRB     D3               ;INICILIZA BANDERAS          4,2
          MOVL     (A0)+,D0         ;COLOCA EL MINUENDO          4,2
          MOVL     (A0),D1          ;COLOCA EL SUSTRAENDO        4,2
INICIO:   BCLR     #37,D0           ;CHECA EL SIGNO DEL SUST.   14,4
          BNE      ANEG             ;SI ES NEGATIVO SALTA (12)  10,4
          BCLR     #37,D1           ;CHECA EL SIGNO DEL MIN.    14,4
          BNE      SIGDIF           ;SI ES NEGATIVO SALTA (12)  10,4
SIGNI:    CMP      D1,D0            ;IDENTIFICA AL MAYOR         6,2
          BHI      AMAYB            ;SALTA SI MIN. >= SUST(12)  10,4
          BSET     #1,D3            ;SI NO ENCIENDE BANDERA     12,4
          SUBL     D0,D1            ;EFECTUA LA OPERACION        8,2
          MOVL     D1,D0            ;COLOCA EL RESULTADO         4,2
          BR       SIGRES           ;PON EL SIGNO AL RES.       10,2
ANEG:     ADDB     #1,D3            ;ENCIENDE BANDERA SIGNO      4,2
          BCLR     #37,D1           ;CHECA SIGNO SUST.          14,4
          BNE      SIGNI            ;SI ES NEG.SIG.IGUALES(12)  10,4
SIGDIF:   ADDL     D1,D0            ;EFECTUA LA OPERACION        8,2
          BTST     #37,D0           ;CHECA SOBREFLUJO           10,4
          BNE      FIN              ;SI HAY; VE A FIN      (12) 10,4
          BTST     #0,D3            ;EL MINUENDO ES NEGAT.?     10,4
          BEQ      FIN              ;NO, VE A FIN          (12) 10,4
          BSET     #37,D0           ;SI ES NEGATIVO PON SIGNO   12,4
          BR       FIN              ;Y VE A FIN                 10,4
AMAYB:    SUBL     D1,D0            ;EFECTUA LA OPERACION        8,4
SIGRES:   BTST     #0,D3            ;CHECA RESULTADO NEGATIVO   10,4
          BEQ      FIN              ;SI ES  POSITIVO FIN  (12)  10,4
          CMPB     #3,D3            ;CHECA AMBOS NEG Y SUST>MIN 8,4
          BNE      FIN              ;SI MIN>SUST VE A FIN (12)  10,4
          BSET     #37,D0           ;SI NO HAZLO NEGATIVO       12,4
          BR       FIN              ;FIN                        10,2
FIN:      EXIT
          END
```

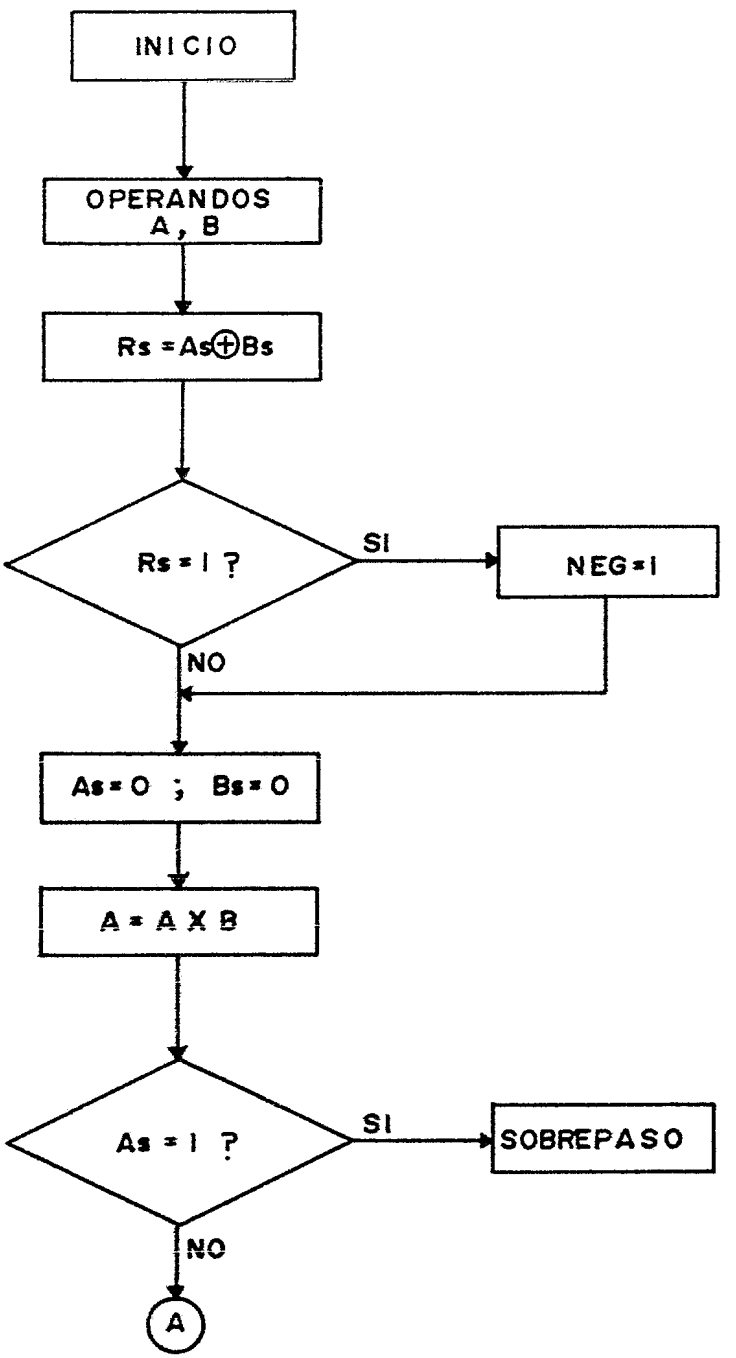
```
;NUMEROS CICLOS EN CADA CASO:

;AMBOS (+) Y  A>B        112
;AMBOS (+) Y  A<B        174
;AMBOS (-) Y  A>B        128
;AMBOS (-) Y  A<B        172
;A (+) Y B (-)           116
;A (-) Y B (+)           140

;CICLOS PROMEDIO         140.33

;BYTES UTILIZADOS        102

;INSTRUCCIONES OCUPADAS  30
```

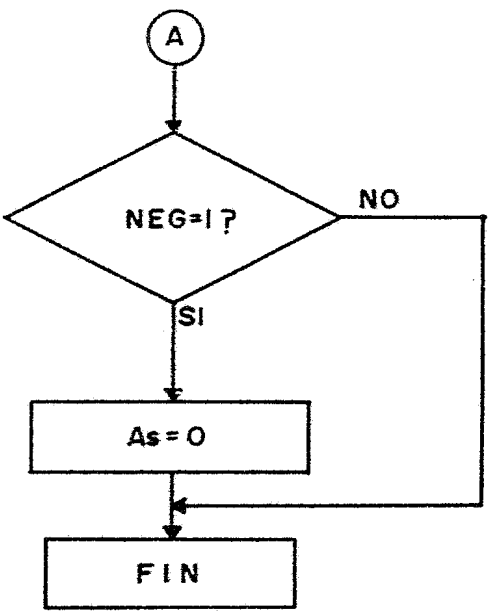
3.2.4 COMENTARIOS.- En este programa, al igual que en el de suma, el procesador que ocupa mayor número de instrucciones es el 8086 (57) así como también el mayor espacio de memoria y ciclos de reloj (144 y 350.5 respectivamente). Esto es debido a lo ya mencionado anteriormente.

El Z8000 ocupa 40 instrucciones, 131.5 ciclos de reloj en promedio y 86 bytes. El 68000 ocupa 30 instrucciones, 140.33 ciclos de reloj en promedio y 102 bytes de espacio de memoria. De esto podemos concluir que el 68000 es el más rápido de los tres para efectuar sumas y restas en 32 bits. Estas operaciones son efectuadas con mayor rapidez y eficiencia debido a que las instrucciones se ejecutan en menor cantidad de ciclos de reloj así como también al uso de modos de direccionamiento más sofisticados.

3.3 MULTIPLICACION DE 32 BITS.- Este programa toma dos números de 32 Bits contenidos en localidades de memoria consecutivas, los números están representados como una magnitud de 31 Bits y el signo esta especificado por el Bit más significativo.

3.3.1 DESCRIPCION DE DIAGRAMA DE FLUJO.- El diagrama de flujo para la multiplicación revisa primero los signos de los operandos, si estos son diferentes se prende una bandera, se borran los signos y se efectúa la operación. Después, se checa si la bandera esta prendida, si es así se coloca signo negativo al resultado. En la rutina de multiplicación la operación se efectúa de acuerdo a las instrucciones disponibles en cada procesador como se observa en los listados respectivos.


INICIO
OPERANDOS A, B
Rs = As⊕Bs
Rs = 1 ?
SI
NEG=1
NO
As = 0 ; Bs = 0
A = A X B
As = 1 ?
SI
SOBREPASO
NO
A


A
NEG=1 ?
NO
SI
As = 0
F I N

```
;*************************************************************
;*                                                           *
;* PROGRAMA PARA MULTIPLICAR DOS NUMEROS DE 32 BITS EN 8086  *
;*                                                           *
;*************************************************************
;
;Este programa multiplica dos numeros de 32 bits representados
;como una magnitud de 31 bits y un bit de signo (bit 31),
;entrega el resultado en la localidad de memoria apuntada
;por el contenido del registro [BX+8] a [BX+14].
;
;
;Descripcion de Registros:
;[BX] y [BX+2] contienen el multiplicando con la
;parte alta en la localidad apuntada por [BX+2].
;[BX+4] y [BX+6] contienen el multiplicador con la
;parte alta en la localidad apuntada por [BX+6].
;AX, CX y DX son los registros operadores.
;BX es el registro apuntador de memoria.
;SI es el registro de banderas.


        ORG     100H
        PUT     100H
INICIO:                                         ;BYTES,CICLOS
        MOV     BX,DIREC        ;CARGA APUNT. DE MEM.     2,10
        MOV     AX,[BX+2]       ;CARGA MULDO. P. ALTA.    3,17
        MOV     CX,[BX+6]       ;CARGA MULOR. P. ALTA.    3,17
        AND     AX,8000H        ;CHECA SIG. MULDO.        3, 4
        AND     CX,8000H        ;CHECA SIG. MULDOR.       3, 4
        XOR     AX,CX           ;COMPARA SIGNOS.          2, 3
        JS      SIGDIF          ;VE A SIG. DIFERENTES.    2,16,4
        MOV     SI,0000H        ;PON BAND. POSITIVOS.     3, 4
        JMP     MULTIP          ;VE A MULTIPLICAR.        3,15
SIGDIF: MOV     SI,8000H        ;PON BAND. NEGATIVOS.     3, 4
MULTIP: AND     [BX+2],7FFFH    ;VAL. ABS. MULDO.         5,26
        AND     [BX+6],7FFFH    ;VAL. ABS. MULDOR.        5,26
        MOV     AX,[BX]         ;CARGA MULDO. P. BAJA.    2,10
        MUL     [BX+4]          ;MULTIPLICA P. BAJAS.     3,148
        MOV     [BX+8],AX       ;SALVA RES. PAR. P.B.     3,18
        MOV     [BX+10],DX      ;SALVA RES. PAR. P.A.     3,18
        MOV     AX,[BX]         ;CARGA MULDO. P. BAJA.    2,10
        MUL     [BX+6]          ;MULTIP. P. B. X P. A.    3,148
        ADD     [BX+10],AX      ;SUMA RESULTS. PARCIA.    3,25
        ADC     [BX+12],DX      ;SALVA PARTE ALTA.        3,25
        JNC     NEXMUL          ;SI NO ACARREO MULTIP.    2,16,4
        INC     [BX+14]         ;INC. LOCAL. MEMORIA.     3,24
NEXMUL: MOV     AX,[BX+2]       ;CARGA MULDO P. ALTA.     3,17
        MUL     [BX+4]          ;MULTIP. P. B. X ALTA.    3,148
        ADD     [BX+10],AX      ;SUMA RESUL. PARCIA.      3,25
        ADC     [BX+12],DX      ;PARTES MEDIAS.           3,25
        JNC     HIORMU          ;SI NO ACARREO MULT.      2,16,4
```

```
        INC     [BX+14]         ;INCREMENTA.                3,24
HIORMU: MOV     AX,[BX+2]       ;CARGA P. A. MULDO.         3,17
        MUL     [BX+6]          ;MULTIPLICA P. ALTAS.       3,148
        ADD     [BX+12],AX      ;SUMA RESUL. PARCIA.        3,25
        ADC     [BX+14],DX      ;SUMA RESUL. PARCIA.        3,25
        OR      SI,[BX+14]      ;PON SIG. RESULTADO.        3,18
        INT     20H             ;FIN.                       2,51

;ESTE PROGRAMA OCUPA UN TOTAL DE 93 BYTES.
;
;SI LOS OPERANDOS SON DE SIGNOS
;IGUALES SE EJECUTA EN 1036 CICLOS.
;
;SI LOS OPERANDOS SON DE SIGNOS
;DIFERENTES SE EJECUTA EN 1036 CICLOS.

;CICLOS PROMEDIO 1036

;INSTRUCCIONES 34
```

```
!*********************************************************!
!*                                                       *!
!*   MULTIPLICACION DE DOS NUMEROS DE 32 BITS EN Z8000   *!
!*                                                       *!
!*********************************************************!

!Este Programa Multiplica 2 Numeros de 32 Bits Represen_   !
!tados como una magnitud de 31 bits y un bit de signo      !
!(bit 31) el Resultado de 64 bits es entregado en RQ0 como !
!una magnitud de 63 bits y un bit de signo (bit 63)        !
!Descripcion de registros: !
!RR0=Multiplicando!
!RR2=Multiplicador!
!R4=Apuntador de Operandos!
!R5=Registro de Banderas como Sigue: !
!Bit 0=Indica Signos Diferentes en los Operandos!
!RQ0=Resultado de 64 Bits!

MULTIPLICA MODULE
CONSTANT
        ADDR    :=%1500
GLOBAL
        MULTIP PROCEDURE
ENTRY
                                                    !CICLOS,BYTES!
                                                              !
        LD      R4,#ADDR        !CARGA DIR.DE OPERANDOS   7,4!
        CLR     R5              !BORRA BANDERAS           7,2!
        LDM     R0,@R4,#4       !CARGA OPERANDOS         26,4!
        INC     R4,#4           !APUNTA AL MULTIPLICADOR  4,2!
        XOR     R2,R0           !CHECA SIGNOS             4,2!
        BIT     R2,#15          !BIT 15 R2=0 DOS+ 0 DOS  -4,2!
        JR      Z,M             !SI, BORRA SIG. MULT.     6,2!
        SET     R5,#0           !NO, PON BAND. SIG.DIF.   4,2!
M:      RES     R0,#15          !BORRA SIG MULTIPLICANDO  4,2!
        RES     @R4,#15         !BORRA SIG MULTIPLICADOR  4,2!
        LDL     RR2,RR0         !MULTIPLICANDO EN RR2     5,2!
        MULTL   RQ0,@R4         !MULTIP.RESULT. EN RQ0  282,2!
        TEST    R5              !CHECA SIGNOS DIFERENTES  7,2!
        JR      Z,FIN           !NO, VE AL FINAL          6,2!
        SET     R0,#15          !SI, HAZ RESULTADO NEG.   4,2!
 IN:    HALT
END MULTIP
END MULTIPLICA

!CICLOS PARA CADA CASO: !
!AMBOS POSITIVOS O AMBOS NEGAIVOS 362 CICLOS!
!SIGNOS DIFERENTES                370 CICLOS!

!BYTES 34!
!INSTRUCCIONES 16!
```

```
!*****************************************************!
!*                                                   *!
!*   MULTIPLICACION DE DOS NUMEROS DE 32 BITS EN Z8000   *!
!*                                                   *!
!*****************************************************!

!Este Programa Multiplica 2 Numeros de 32 Bits Represen_   !
!tados como una magnitud de 31 bits y un bit de signo      !
!(bit 31) el Resultado de 64 bits es entregado en RQ0 como !
!una magnitud de 63 bits y un bit de signo (bit 63)        !
!Descripcion de registros: !
!RR0=Multiplicando!
!RR2=Multiplicador!
!R4=Apuntador de Operandos!
!R5=Registro de Banderas como Sigue: !
!Bit 0=Indica Signos Diferentes en los Operandos!
!RQ0=Resultado de 64 Bits!

MULTIPLICA MODULE
CONSTANT
        ADDR    :=%1500
GLOBAL
        MULTIP PROCEDURE
ENTRY
                                                  !CICLOS,BYTES!
                                                        !
        LD      R4,#ADDR        !CARGA DIR.DE OPERANDOS  7,4!
        CLR     R5              !BORRA BANDERAS          7,2!
        LDM     R0,@R4,#4       !CARGA OPERANDOS        26,4!
        INC     R4,#4           !APUNTA AL MULTIPLICADOR 4,2!
        XOR     R2,R0           !CHECA SIGNOS            4,2!
        BIT     R2,#15          !BIT 15 R2=0 DOS+ 0 DOS -4,2!
        JR      Z,M             !SI, BORRA SIG. MULT.    6,2!
        SET     R5,#0           !NO, PON BAND. SIG.DIF.  4,2!
M:      RES     R0,#15          !BORRA SIG MULTIPLICANDO 4,2!
        RES     @R4,#15         !BORRA SIG MULTIPLICADOR 4,2!
        LDL     RR2,RR0         !MULTIPLICANDO EN RR2    5,2!
        MULTL   RQ0,@R4         !MULTIP.RESULT. EN RQ0 282,2!
        TEST    R5              !CHECA SIGNOS DIFERENTES 7,2!
        JR      Z,FIN           !NO, VE AL FINAL         6,2!
        SET     R0,#15          !SI, HAZ RESULTADO NEG.  4,2!
 IN:    HALT
END MULTIP
END MULTIPLICA

!CICLOS PARA CADA CASO: !
!AMBOS POSITIVOS O AMBOS NEGAIVOS 362 CICLOS!
!SIGNOS DIFERENTES                370 CICLOS!

!BYTES 34!
!INSTRUCCIONES 16!
```

```
;*************************************************************
;*                                                           *
;*       PROGRAMA DE MULTIPLICACION PARA 68000               *
;*                                                           *
;*************************************************************

;      Este programa multiplica dos numeros con signo de
;hasta 32 bits, que se encuentran en una direccion dada.  El
;resultado estara en dos registros de datos, D0 tendra la
;parte menos significativa del resultado y D2  la  parte  mas
;significativa.
;     Los registros se utilizan de la siguiente manera:

;       A0  DIRECCION DE LA RUTINA
;       D0  MULTIPLICANDO
;       D1  REGISTRO INTERMEDIO
;       D2  REGISTRO INTERMEDIO
;       D3  MULTIPLICADOR
;       D4  REGISTRO INTERMEDIO
;       D7  BANDERA DE SIGNO

        COPY    SYS

        TYPE    (Direcion inicio memoria :)
        KBD
        GTOCT                                          ;CICLOS, BYTES
        MOVL    D1,A0           ;COLOCA DIR.DE LOS OP.      4,2
INICIO: CLRB    D7              ;LIMPIA BANDERA NEGATIVOS   4,2
        MOVL    (A0)+,D0        ;CARGA MULTIPLICANDO        4,2
        MOVL    (A0),D3         ;CARGA MULTIPLICADOR        4,2
        BCLR    #37,D0          ;MULTIPLICANDO NEGATIVO?   14,4
        BEQ     A               ;NO,CHECA MDOR.NEG.  (12)  10,4
        ADD     #1,D7           ;SI,PRENDE BANDERA NEG.     8,4
A:      BCLR    #37,D3          ;MULTIPLICADOR NEGATIVO?   14,4
        BEQ     B               ;NO,VE A MULTIPLICAR  (12) 10,4
        ADD     #1,D7           ;SI,PRENDE BAN.MDOR.NRG.    8,4
B:      MOVL    D0,D1           ;PRINCIPIA RUTINA MULTIPL   4,2
        MOVL    D0,D2           ;                           4,2
        MOVL    D3,D4           ;                           4,2
        SWAP    D2              ;                           4,2
        SWAP    D4              ;                           4,2
        MULU    D0,D3           ;MULTIPLICACION BAJOS      70,2
        MULU    D3,D2           ;MULTIPLICACION BAJOxALTO  70,2
        MULU    D1,D4           ;MULTIPLICACION ALTOxBAJO  70,2
        MULU    D2,D4           ;MULTIPLICACION ALTOxALTO  70,2
        SWAP    D0              ;INTERC.1A MULT.ALTAxBAJA   4,2
        ADD     D3,D0           ;1A MULT. ALTA + 2A BAJA    8,2
        CLRL    D4                                          6,2
        ADDXL   D4,D2           ;PROPAGA ACARREO            8,2
        ADD     D1,D0           ;3A MULT.BAJA+1A ALTA+2A B. 8,2
        ADDXL   D4,D2           ;PROPAGA ACARREO            8,2
        SWAP    D0              ;ACOMODA PRODUCTO BAJO      4,2
```

```
        CLR     D3              ;BORRA P.BAJA 2DA MULTIP.   4,2
        CLR     D1              ;BORRA P.BAJA 3RA MULTIP.   4,2
        SWAP    D3              ;PON PARTE ALTA EN P.BAJA   4,2
        SWAP    D1              ;PON PARTE ALTA EN P.BAJA   4,2
        ADDL    D3,D2           ;SUMA 2A MULT. ALTA+4A BAJA8,2
        BTST    #37,D2          ;CHECA SOBREFLUJO          10,4
        BNE     FIN             ;SI,VE A RUT.SOBREF. (12)  10,4
        ADDL    D1,D2           ;2A MULT. ALTA+4A B.+3A A.  8,2
        BTST    #37,D2          ;CHECA SOBREFLUJO          10,4
        BNE     FIN             ;SI,VE A RUT.SOBREF. (12)  10,4
        BTST    #0,D7           ;CHECA NEGATIVOS           10,4
        BEQ     FIN             ;DOS O NING.RES. POS.(12)  10,4
        BSET    #37,D2          ;SI,HAZ RES.NEGAT.         12,4
FIN:    EXIT
        END

;NUMERO DE CICLOS PARA CADA CASO:

;                       ciclos

;A (+) Y B (+)           484
;A (-) Y B (-)           492
;A (+) Y B (-)           514
;A(-) Y  B (+)           514

;CICLOS PROMEDIO         501

;BYTES UTILIZADOS        106

;INSTRUCCIONES EMPLEADAS 40
```

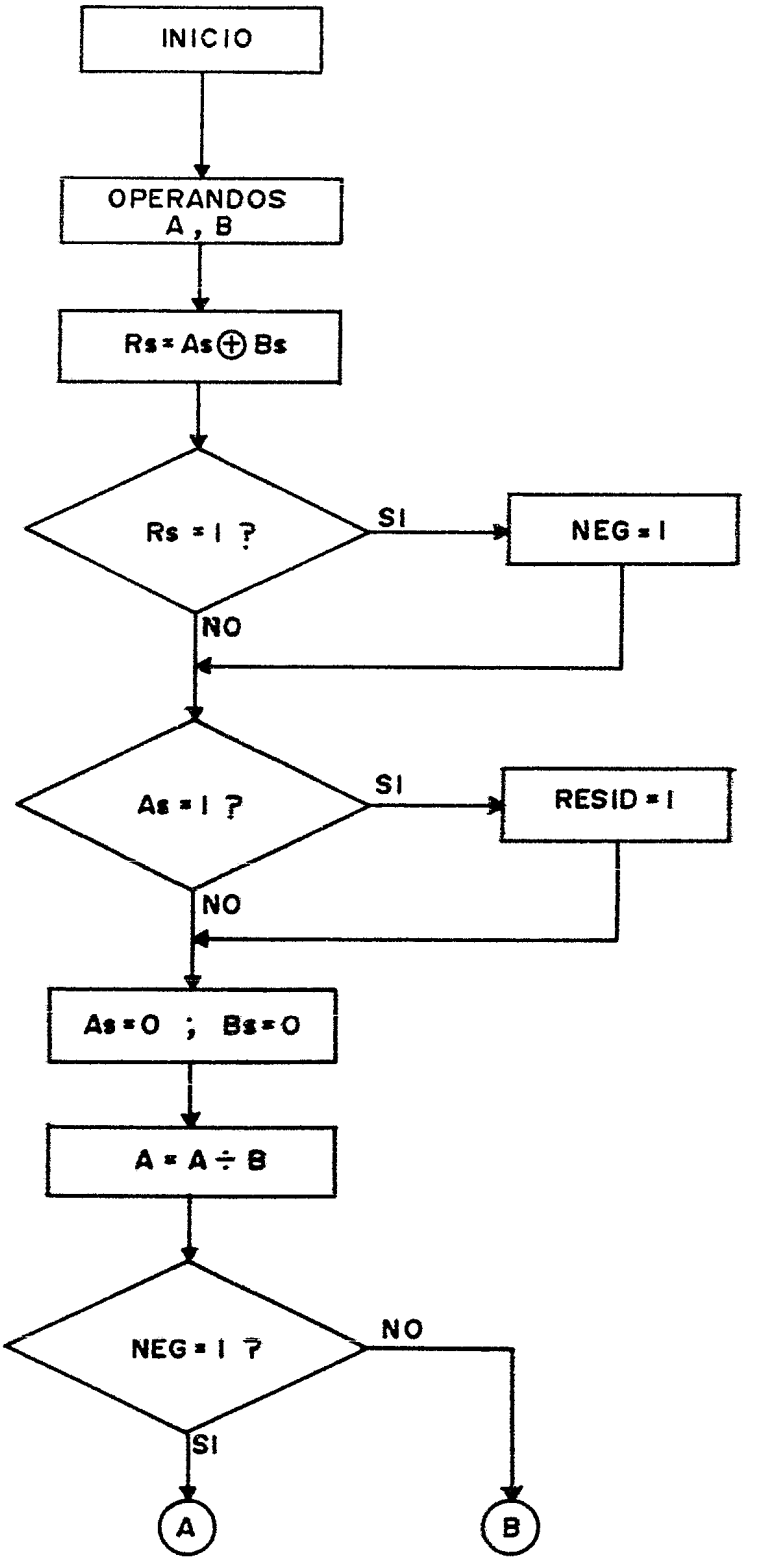
3.3.4 COMENTARIOS.- En este programa existe una marcada diferencia entre los tres microprocesadores. Debido a la falta de registros suficientes en el 8086, el programa es ejecutado sobre la memoria en gran parte, esto toma más tiempo (1036 ciclos en promedio), además, no puede efectuar la multiplicación en 32 bits directamente.

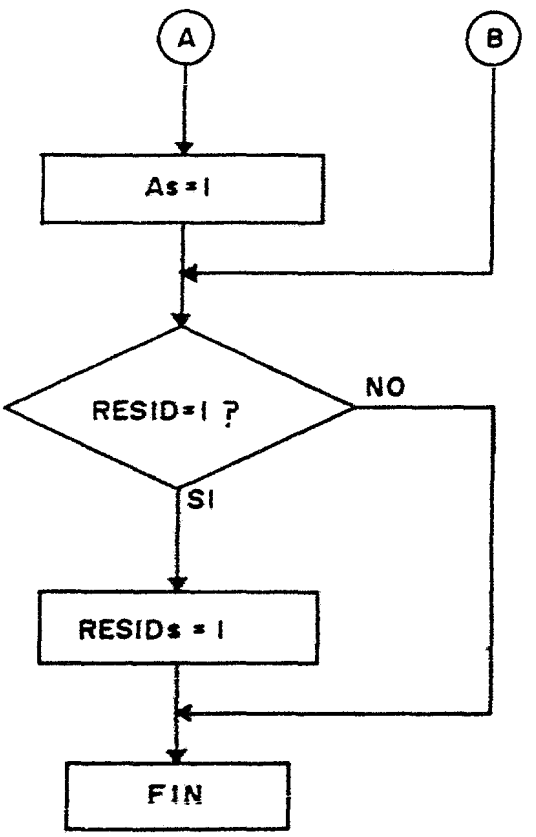
El 68000 tiene suficientes recursos en cuanto a registros pero no puede efectuar la multiplicación de 32 bits directamente, y aunque es efectuada internamente en gran parte del programa, ocupa mayor número de ciclos de reloj (501) que el Z8000 a pesar de su mayor velocidad en la ejecución de instrucciones de suma y direccionamiento de memoria, además ocupa mucho mayor espacio (106 Bytes) y número de instrucciones (40).

El Z8000 es mucho más eficiente ya que es capaz de efectuar la multiplicación de 32 Bits con una sola instrucción, gracias a esto el programa es muy corto (16 insrucciones y 34 Bytes) y además es más rápido (366 ciclos en promedio).

3.4 DIVISION DE 32 BITS.- Este programa toma un número de 32 Bits el cual será dividido entre uno de 16 Bits, los números están contenidos en localidades de memoria consecutivas y están representados en magnitud y signo. El signo de los operandos está representado por el bit más significativo de cada uno de ellos.

3.4.1 DESCRIPCION DEL DIAGRAMA DE FLUJO.- Este programa es similar al de multiplicación con la diferencia que la operación efectuada es división y existe una rutina para obtener el signo del residuo.


INICIO
OPERANDOS
A , B
Rs = As ⊕ Bs
Rs = 1 ?
SI
NEG = 1
NO
As = 1 ?
SI
RESID = 1
NO
As = 0 ; Bs = 0
A = A ÷ B
NEG = 1 ?
NO
SI
A
B


A
B
As=1
RESID=1 ?
NO
SI
RESIDs = 1
FIN

```
;*************************************************************
;*                                                           *
;*   PROGRAMA PARA DIVIDIR DOS NUMEROS DE 32 BITS EN 8086    *
;*                                                           *
;*************************************************************
;
;Este programa divide dos numeros de 32 bits representados
;como una magnitud de 31 bits y un bit de signo (bit 31)
;y entrega el resultado en la localidad de memoria apun-
;tada por el contenido del registro [BX+8] a [BX+14].
;
;
;Descripcion de Registros:
;[BX] y [BX+2] contienen el dividendo con la
;parte alta en la localidad apuntada por [BX+2].
;[BX+4] contiene el divisor
;en la localidad apuntada por [BX+4].
;AX, CX y DX son los registros operadores.
;BX es el registro apuntador de memoria.
;SI es el registro de banderas.

        ORG     100H
        PUT     100H
        JMP     INICIO
DIREC:  DW      0,0,0
INICIO:                                         ;BYTES,CICLOS
        MOV     BX,DIREC        ;CARGA EL APUNAT. MEM.    3, 4
        MOV     AX,[BX]         ;CAR. PAR. ALT. DIVID.    2,10
        MOV     CX,[BX+4]       ;CAR. PAR. ALT. DIVIS.    3,17
        AND     AX,8000H        ;CHECA SIGNOS.            3, 4
        MOV     DI,AX           ;SALVA SIG. DIVIDENDO.    2, 2
        AND     CX,8000H        ;CHECA SIG. DIVISOR.      3, 4
        XOR     AX,CX           ;CHECA SIG. DIVISOR.      2, 3
        MOV     SI,AX           ;COLOCA SIG. RESULT.      2, 2
        MOV     AX,[BX]         ;CARGA DIVID. EN AX       2,10
        AND     AX,7FFFH        ;VALOR ABSOLUTO.          3, 4
        MOV     DX,[BX+2]       ;CAR. PAR. BAJA DIVID.    3,17
        MOV     CX,[BX+4]       ;CARGA EL DIVISOR.        3,17
        AND     CX,7FFFH        ;SACA VALOR ABSOLUTO.     3, 4
        DIV     AX,CX           ;EFECTUA DIVISION.        2,162
        OR      AX,SI           ;COLOCA EL SIGNO.         2, 3
        MOV     [BX+6],AX       ;SALVA EL RESULTADO.      3,18
        OR      DX,DI           ;PON SIG. AL RESIDUO.     2, 3
        MOV     [BX+8],DX       ;SALVA EL RESIDUO.        3,18
        INT     22H             ;FIN.                     2,51
```

```
;TIENE UN TOTAL DE 46 BYTES.
;EL PROGRAMA SE EJECUTA EN 332 CICLOS NO
;IMPORTANDO EL SIGNO DE LOS OPERANDOS.
;
;SI EXISTE UN SOBREFLUJO POR DIVISION
;ENTRE CERO EL PROCESADOR AUTOMATICAMENTE
;EJECUTA UNA TRAMPA DE DIVISION POR CERO.

;CICLOS PROMEDIO 332

;INSTRUCCIONES 19
```

```
!*************************************************************!
!*                                                           *!
!*    PROGRAMA PARA DIVIDIR UN NUMERO DE 32 BITS ENTRE       *!
!*                                                           *!
!*      UNO DE 16 BITS EN MAGNITUD Y SIGNO EN Z8000          *!
!*                                                           *!
!*************************************************************!

!Este programa divide un numero de 32 bits entre uno de     !
!16 representados como una magnitud de 31 y de 15 bits y    !
!un bit de signo (bit 31 y bit 15) el resultado es          !
!puesto en RR0 siendo la parte alta de este el cociente     !
!y la parte baja el residuo                                 !
!Descripcion de Registros: !
!RR0=Registro para Dividendo!
!R2=Registro para Divisor!
!R4=Apuntador de operandos!
!R5=registro de Banderas como Sigue: !
!Bit 0=Operandos con signo diferente!
!Bit 1=Dividendo Negativo!

DIVIDE MODULE
CONSTANT
        ADDR    :=%1500
GLOBAL
        DIVIS PROCEDURE
ENTRY
                                                    !CICLOS,BYTES!
                                                                 !
        LD      R4,#ADDR        !CARGA DIR.DE LOS OPERANDOS   7,4!
        CLR     R5              !BORRA BANDERAS               7,2!
        LDM     R0,@R4,#3       !CARGA DIVIDENDO EN RR0      23,4!
        INC     R4,#4           !APUNTA AL DIVISOR            4,2!
        XOR     R2,R0           !CHECA SIGNOS                 4,2!
        BIT     R2,#15          !BIT 15 DE R2=0 A,B+ o A,B-   4,2!
        JR      Z,OP            !SI, BORRA SIGNOS Y DIVIDE    6,2!
        SET     R5,#0           !NO, PON BANDERA SIG. DIF.    4,2!
OP:     BIT     R0,#15          !DIVIDENDO NEGATIVO ?         4,2!
        JR      Z,DI1           !NO, BORRA SIGNOS Y DIVIDE    6,2!
        SET     R5,#1           !SI, PON BANDERA              4,2!
DI1:    RES     R0,#15          !BORRA SIGNO DIVIDENDO        4,2!
        RES     @R4,#15         !BORRA SIGNO DIVISOR          4,2!
        DIV     RR0,@R4         !DIVIDE Y PON RES.EN RR0    107,2!
        BIT     R5,#0           !CHECA SIGNOS DIFERENTES      7,2!
        JR      Z,RESID         !NO, VE AL FINAL              6,2!
        SET     R1,#15          !SI, HAZ RESULTADO NEGATIVO   4,2!
RESID:  BIT     R5,#1           !DIVIDENDO NEGATIVO           7,4!
        JR      Z,FIN           !NO, VE AL FINAL              6,2!
        SET     R0,#15          !SI, HAZ RESIDUO NEGATIVO     4,2!
FIN:    HALT                    !FINAL!
END DIVIS
END DIVIDE
```

```
!CICLOS PARA CADA CASO: !
!AMBOS POSITIVOS O AMBOS NEGATIVOS   178 CICLOS!
!A+ B-                               209 CICLOS!
!A- B+                               213 CICLOS!

!BYTES 46!
!INSTRUCCIONES 21!
```

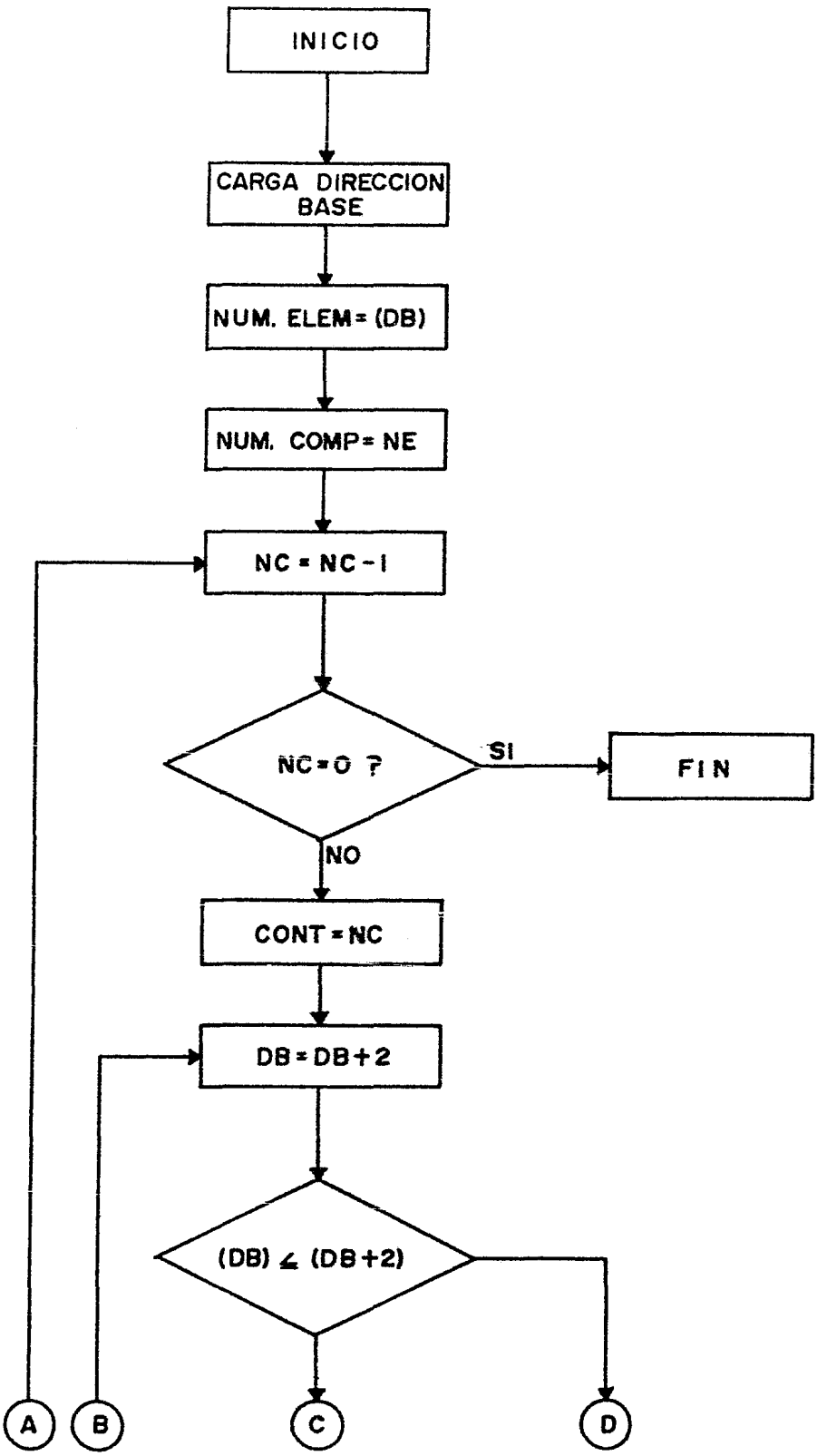
```
;*************************************************************
;*                                                           *
;*          PROGRAMA DE DIVISION PARA 68000                  *
;*                                                           *
;*************************************************************


;     Este  programa  divide dos numeros que se encuentran en
;memoria, donde el divisor puede ser hasta  de  16  bits  con
;signo  y  el  dividendo  hasta  de  32  bits  con  signo. El
;resultado quedara en los registros de datos,  el  cociente
;estara en el registro D1 y el residuo en D0.


        COPY    SYS

        TYPE    <Direcion de los operandos :>
        KBD
        GTOCT                                       ;CICLOS, BYTES
        MOVL    D1,A0           ;CARGA LA DIRECCION          4,2
INICIO: MOVL    (A0)+,D0        ;COLOCA EL DIVIDENDO         4,2
        MOVW    (A0),D1         ;COLOCA DIVISOR              4,2
        CLRB    D2              ;INICIALIZA BANDERA SIGNOS   4,2
        BCLR    #37,D0          ;CHECA SIGNO DIVIDENDO      14,2
        BEQ     SIG             ;POS. DIVISOR NEGAT.  (12)  10,4
        ADD     #5,D2           ;NEGATIVO, PON BANDERA       4,2
SIG:    BCLR    #17,D1          ;CHECA SIGNO DIVISOR        14,4
        BEQ     DIV             ;POSITIVO, DIVIDE     (12)  10,4
        ADD     #1,D2           ;NEGATIVO, PON BANDERA       4,2
DIV:    DIVU    D0,D1           ;EFECTUA OP. D0=D0/D1      140,2
        MOVW    D0,D1           ;SALVA EL COCIENTE           4,2
        CLRW    D0              ;BORRA EL COCIENTE           4,2
        SWAP    D0              ;PON EL RESIDUO EN P.BAJA    4,2
        BTST    #0,D2           ;EL RESULTADO ES NEGATIVO    8,4
        BEQ     RESIDU          ;POSITIVO, VE AL FINAL(12)  10,4
        BSET    #37,D1          ;PON COCIENTE NEGATIVO      12,4
RESIDU: BTST    #3,D2           ;CHECA EL SIGNO RESIDUO      8,4
        BEQ     FIN             ;VE A FIN             (12)  10,4
        BSET    #37,D0          ;PON RESIDUO NEGATIVO       12,4
FIN:    EXIT
        END
```

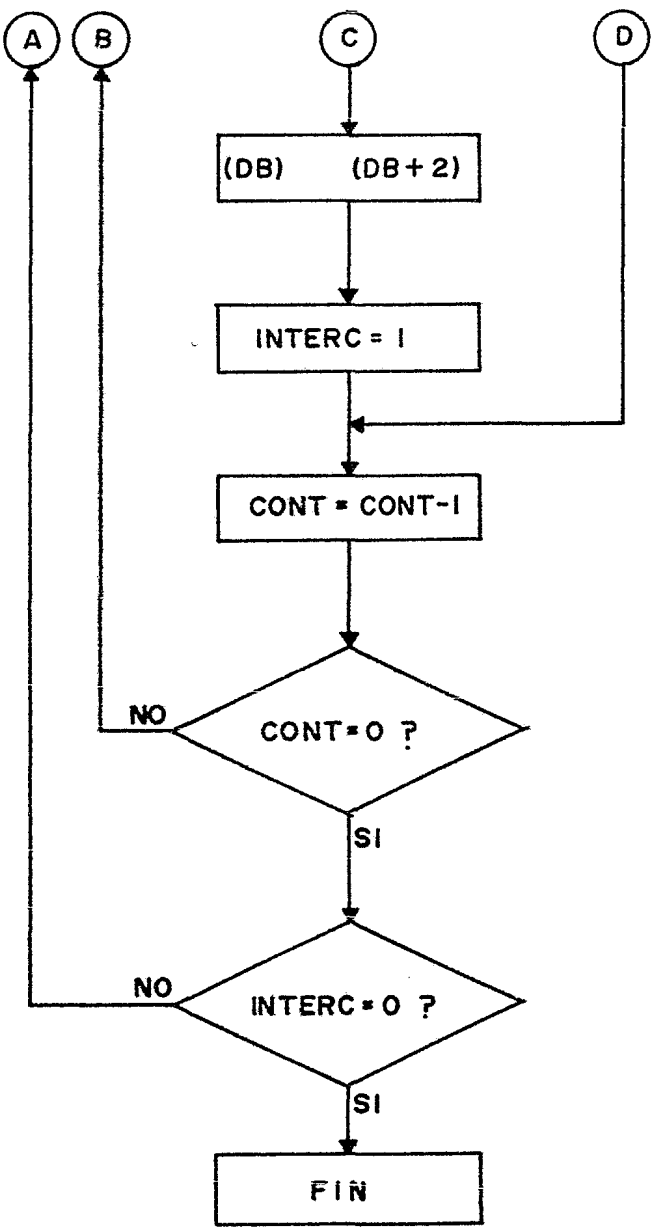
```
;NUMERO DE CICLOS PARA CADA CASO:

;                          CICLOS

;AMBOS POSITIVOS                  260
;AMBOS NEGATIVOS                  274
;A (+) Y B (-)             272
;A (-) Y B (+)            282

;CICLOS PROMEDIO           273.25

;BYTES UTILIZADOS          58

;INSTRUCCIONES OCUPADAS    19
```

3.4.4 COMENTARIOS.- En este programa los tres procesadores son bastante similares en cuanto a tamaño del programa. En tiempo de ejecución el más rápido es el Z8000 que ocupa 200 ciclos de reloj en promedio, le sigue el 68000 con 273.25 ciclos y después el 8086 con 332 ciclos. Es de hacer notar que en el programa de Z8000 se utilizó la instrucción DIV, la que divide un número de 32 bits entre uno de 16. Existe también la instrucción DIVL la cual divide un número de 64 bits entre uno de 32. Esta capacidad no la tiene ninguno de los otros dos procesadores. Debido a lo anterior, en los programas de multiplicación y división, el Z8000 es mucho más eficiente que los otros dos procesadores.

3.5 PROGRAMA DE REACOMODO POR EL METODO DE BURBUJA.- Este programa ordena una tabla contenida en memoria la cual esta compuesta por elementos de 16 bits usando el método de Reacomodo de Burbuja, el número de elementos está especificado por la primera palabra contenida en la tabla.

3.5.1 DESCRIPCION DEL DIAGRAMA DE FLUJO.- Este programa toma el número de elementos de la localidad especificada como dirección base de la tabla, calcula el número de comparaciones necesarias y checa si hay un solo elemento en la tabla. Después inicializa la bandera de intercambios y empieza a efectuar comparaciones entre parejas de elementos contiguos intercambiándolos si es necesario. Después de efectuar las comparaciones entre todos los elementos de la lista, checa la bandera de intercambios, si está prendida decrementa por uno el número de comparaciones y efectúa otra pasada sobre la tabla realizando las operaciones mencionadas anteriormente, el programa termina cuando el número de comparaciones es cero o cuando la bandera de intercambios está apagada.


INICIO
CARGA DIRECCION BASE
NUM. ELEM = (DB)
NUM. COMP = NE
NC = NC - 1
NC = 0 ?
SI
FIN
NO
CONT = NC
DB = DB + 2
(DB) ≤ (DB+2)
A
B
C
D


A
B
C
D
(DB) (DB+2)
INTERC = 1
CONT = CONT-1
NO
CONT=0 ?
SI
NO
INTERC=0 ?
SI
FIN

```
;*************************************************************
;*                                                           *
;* PROGRAMA PARA  ORDENAR  UNA  LISTA  DE ELEMENTOS POR  EL *
;*          METODO DE BURBUJA PARA 8086                      *
;*                                                           *
;*************************************************************
;
;Este programa ordena una lista de elementos  de 16 bits
;en orden ascendente, la lista esta localizada en memoria,
;el primer elemento indica el numero de elementos que
;debe ordenar.
;
;Descripcion de Registros:
;
;[BX] contiene la direccion inicial de la tabla.
;SI se utiliza como apuntador de direccion inicial
;CX se utiliza como como un contador de comparaciones.
;DX es utilizado como un registro de banderas.
;AX es el registro con el cual se efectuan las comparaciones.
;
        ORG     100H
        PUT     100H
INICIO:                                         ;BYTES,CICLOS
        MOV     BX,DIREC        ;CARGA APUNT. DE MEM.    3, 4
        MOV     SI,[BX]         ;CARGA NUM. ELEM.        2,13
DECNC:  DEC     SI              ;COMP. = NUM. ELEM.-1    1, 2
        CMP     SI,0000H        ;SI ES CERO, UN ELEM.    3, 4
        JZ      FIN             ;CERO, VE AL FIN.        2,16,4
        MOV     CX,SI           ;CARGA CONT. DE COMP.    2, 2
        MOV     BX,DIREC        ;REINICIALIZA DIREC.     3, 4
        MOV     DX,0000H        ;INICIA. BAND. INTER.    3, 4
INCAP:  ADD     BX,0002H        ;INCREMENTA APUNTADOR    3, 4
        MOV     AX,[BX]         ;CARGA ELEMENTO EN AX    2,10
        CMP     AX,[BX+2]       ;COMPARALO CON EL SIG.   3,17
        JZ      SIGPAR          ;IGUALES VE COMP. SIG.   2,16,4
        JL      SIGPAR          ;MENOR, COMPARAR SIG.    2,16,4
        XCHG    AX,[BX+2]       ;NO, INTERCAMBIALOS.     3,26
        MOV     [BX],AX         ;INTERCAMBIALOS.         2,10
        MOV     DX,0001H        ;PON BANDERA.            3, 4
SIGPAR: LOOP    INCAP           ;DEC. CONT. DE COMPAR.   2,17,5
        CMP     DX,0000H        ;INTERCAMBIO ?           3, 4
        JNZ     DECNC           ;SI, HAZ OTRA PASADA.    2,16,4
FIN:    INT     20H             ;FIN.                    2,51
DIREC:  DW      20
        DW      32,56,120,2,3,1,5,4,2,99,54,34,5,32,44,56,43,32,78,21
```

```
;INSTRUCCIONES 20
;CICLOS:
;INICIO 17
;LAZO 2 CON INTERCAMBIO 79
;LAZO 2 SIN INTERCAMBIO 47
;LAZO 1 LAZO2+ 40

;BYTES 48
```

```
!*************************************************************!
!*                                                           *!
!*      PROGRAMA DE REACOMODO POR METODO DE BURBUJA          *!
!*                                                           *!
!*                  PARA Z8000 (WORDS)!                      *!
!*                                                           *!
!*************************************************************!

!Este programa reacomoda en orden ascendente una tabla de   !
!N elementos localizada en memoria el numero de elementos   !
!es el primer elemento de la tabla la tabla es reacomodada  !
!en la misma posicion de memoria.                           !

!Descripcion de registros:  !
!R0=Numero de Comparaciones!
!R1=Elemento apuntado por R4!
!R2=Contador de Comparaciones!
!R3=Direccion Base de la Tabla!
!R4=Apuntador!
!R5=Registro de Banderas como sigue:  !
!Bit 0=Bandera de Intercambio!

BURBU MODULE
CONSTANT
        ADDR    :=%1500
GLOBAL
        BUSORT PROCEDURE
ENTRY
                                                        !CICLOS,BYTES!
                                                                 !
        LD      R3,#ADDR        !DIRECCION BASE TABLA EN R3 7,4!
        LD      R0,@R3          !NUMERO DE ELEMENTOS EN R0  7,2!
COMP2:  LD      R4,R3           !DIR. BASE DE TABLA EN R4   3,2!
        DEC     R0,#1           !HAZ No. COMPAR. =#ELEM.-1  4,2!
        TEST    R0              !SOLO UN ELEMENTO ?         7,2!
        JR      Z,FIN           !SI, VE AL FINAL            6,2!
        LD      R2,R0           !NO, PON No. COMP. EN R2    3,2!
        CLR     R5              !BORRA BANDERAS             7,2!
COMP1:  INC     R4,#2           !APUNTA AL ELEMENTO         4,2!
        LD      R1,@R4          !CARGA ELEMENTO EN R1       7,1!
        CP      R1,%2(R4)       !COMPARALO CON EL SIGUIENTE10,6!
        JR      LE,SIGUI        !MENOR? COMPARA SIG. PAREJA 6,2!
        EX      R1,%2(R4)       !NO, INTERCAMBIALOS        16,6!
        LD      @R4,R1          !NO, INTERCAMBIALOS         8,2!
        SET     R5,#0           !PON BANDERA INTERCAMBIO    4,2!
SIGUI:  DJNZ    R2,COMP1        !No. COMP. =0? NO, SIG.    11,2!
        TEST    R5              !HUBO INTERCAMBIO?          7,2!
        JR      NZ,COMP2        !SI, HAZ OTRA PASADA        6,2!
FIN:    HALT                    !NO, FINAL!
END BUSORT
END BURBU
```

```
!NUMERO DE CICLOS PARA CADA CASO!
!INICIO                  14 CICLOS!
!LOOP 2 (SIN INTERC.)  38 CICLOS!
!LOOP 2 (CON INTERC.)  66 CICLOS!
!LOOP 1  43 CICLOS + LOOP 2     !
!LOOP 2 COMPRENDE DESDE COMP1: HASTA SIGUI: !
!LOOP 1 COMPRENDE DESDE COMP2: HASTA 1 LINEA ANTES FIN: !

!BYTES 46!
!INSTRUCCIONES 19!
```

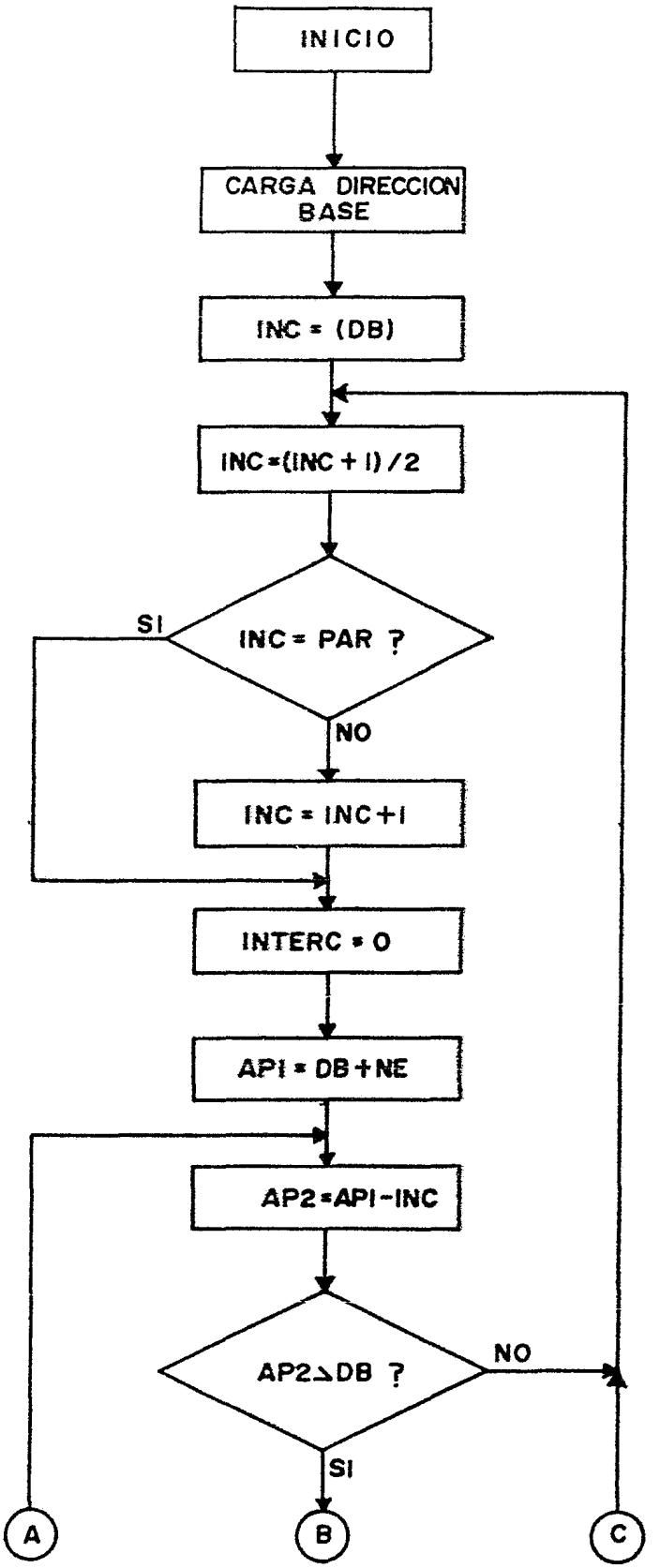
```
;*************************************************************
;*                                                           *
;*       PROGRAMA DE REACOMODO POR METODO DE BURBUJA          *
;*                        PARA 68000                         *
;*************************************************************

;      Este programa aplica el metodo de burbuja mejorado para
;ordenar  una lista desordenada de palabras que se encuentran
;en  memoria,  la  primera  palabra  contiene  el  numero  de
;elementos que tiene la lista. Esta  queda ordenada de  menor
;a mayor.

;Los registros se utilizan de la siguiente manera:


;    A0     DIRECCION DE LA LISTA
;    A1     DIRECCION BASE
;    D0     NUMERO DE ELEMENTOS
;    D1     REGISTRO INTERMEDIO
;    D2     REGISTRO INTERMEDIO
;    D3     BANDERA DE INTERCAMBIO

     COPY       SYS

     TYPE       (Direcion de la tabla :)
     KBD
     GTOCT
                                                  ;CICLOS, BYTES
     MOVL   D1,A0          ;COLOCA DIRECCION DE LOS OP.     4,2
     MOVW   (A0)+,D0       ;NUMERO DE ELEMENTOS             4,2
LAZO:MOVL   A0,A1          ;DIRECCION BASE                  4,2
     SUBW   #1,D0          ; # DE ELEMENTOS= # DE ELE - 1   4,2
     MOVW   D0,D1          ;# DE ELEMENTOS EN D1            4,2
OTRO:MOVW   (A1)+,D2       ;SACA PRIMER ELEMENTO            4,2
     CMPW   D2,(A1)        ;COMPARA ELEMENTO N CON EL N+1   4,2
     BGE    NOIN           ;SI ) 0 = NO INTERCAMBIES  (12) 10,4
INT: MOVW   (A1),-2(A1)    ;SI ES MENOR INTERCAMBIALOS      10,2
     MOVW   D2,(A1)        ;INTERCAMBIO                      5,2
     BSET   #0,D3          ;PRENDE LA BANDERA DE INTERC.    12,4
NOIN:DBF    D1,OTRO        ;BUSCA OTRO ELEMENTO       (12) 10,4
     BCLR   #0,D3          ;HUBO ALGUN INTERCAMBIO          14,4
     BNE    LAZO           ;SI,HAZ OTRA PASADA        (12) 10,4
     EXIT
     END
```

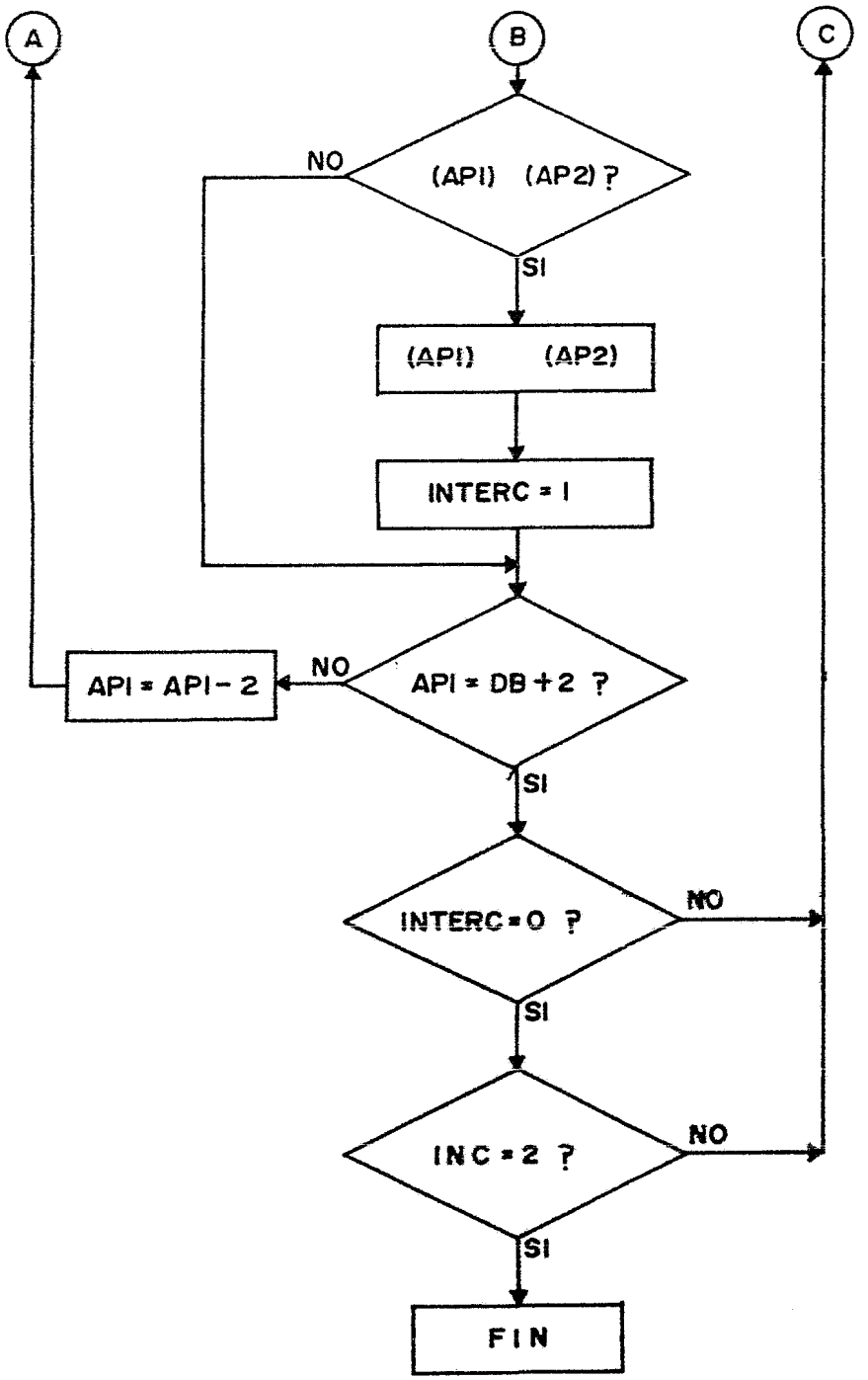
```
;NUMERO DE CICLOS PARA CADA CASO:

;                          CICLOS

;INICIO                     8

;LOOP 2 (SIN INTERC)        32
;LOOP 2 (CON INTERC)        57
;LOOP 1                     38 + LOOP2
;
;LOOP 2 COMPRENDE DESDE "OTRO" HASTA "NOIN"
;LOOP 1 COMPRENDE DESDE "LAZO" HASTA UNA LINEA
;ANTES DEL FIN

;BYTES UTILIZADOS           38

;INTRUCCIONES OCUPADAS      14
```

3.5.4 COMENTARIOS.- En esta rutina no existe gran diferencia entre los tres procesadores, el que ocupa menos instrucciones es el 68000 que tiene 15 instrucciones y 40 bytes, le sigue el Z8000 con 19 instrucciones y 46 bytes, por último el 8086 con 20 instrucciones y 48 bytes. En número de ciclos de reloj, el mas rápido es el 68000 que ocupa 78 ciclos en una pasada sin intercambio de una pareja de elementos y 103 en una pasada con intercambio de elementos.

3.6 PROGRAMA DE REACOMODO POR EL METODO DE SHELL.- Este programa realiza la misma función que el programa anterior usando el Algoritmo de Reacomodo de Shell el cual es más eficiente que el de Burbuja.

3.6.1 DESCRIPCION DEL DIAGRAMA DE FLUJO.- Este programa es similar al método de Burbuja sólo que en lugar de intercambiar elementos adyacentes, intercambia elementos que se encuentran a " D " posiciones uno de otro, con esto se tiene la ventaja de que el movimiento de elementos se efectúa de manera acelerada. El programa termina cuando "D" indica que los elementos apuntados son adyacentes y no hubo intercambio de elementos.


INICIO
CARGA DIRECCION BASE
INC = (DB)
INC=(INC + 1) /2
INC = PAR ?
SI
NO
INC = INC+1
INTERC = 0
API = DB+NE
AP2=API-INC
AP2⊿DB ?
NO
SI
A
B
C


A
B
C
NO
(API) (AP2) ?
SI
(API) (AP2)
INTERC = 1
API = API - 2
NO
API = DB + 2 ?
SI
INTERC = 0 ?
NO
SI
INC = 2 ?
NO
SI
FIN

```
;*************************************************************
;*                                                           *
;* PROGRAMA PARA  ORDENAR  UNA  LISTA  DE ELEMENTOS POR  EL *
;*             METODO  DE  SHELL PARA 8086                   *
;*                                                           *
;*************************************************************
;
;Este programa ordena una lista de elementos  de 16 bits
;en orden asendente, la lista esta localizada en memoria,
;el primer elemento indica el numero de elemetos que
;debe ordenar. La tabla queda ordenada en las
;direcciones iniciales.
;
;Descripcion de Registros:
;
;[BX] contiene la direccion inicial de la tabla.
;DX se utiliza como contador del numero de elementos
;por ordenar ya que este metodo lo va variando.
;CX se utiliza como como registro de intercambios.
;SI y DI se utilizan como apuntadores de direcciones.
;AX es el registro con el cual se efectuan las comparaciones.

        ORG     100H
        PUT     100H
;                                                    BYTES,CICLOS
        MOV     BX,DIREC        ;DIRECC. DE  TABLA.      3, 4
        MOV     BP,BX           ;SALVA LA DIRECC.        2, 2
        MOV     DX,[BX]         ;NUM. ELEM. EN DX.       2,13
        SHL     DX              ;N. E.= NUM. ELEM.X2     2, 2
INICIO: INC     DX              ;N. E.=NUM. ELEMX2+1     1, 2
        SHR     DX              ;N. E.= NUM. ELEM./2     2, 2
        TEST    DX,0001H        ;NUM ELEM. IMPAR ?       4, 5
        JZ      INICON          ;NO, VE A INICIA CONT.   2,16,4
        INC     DX              ;SI, FORZALO A PAR.      1, 2
INICON: MOV     CX,0000H        ;INICIA BAND. INTERC.    3, 4
        MOV     SI,BX           ;APUNT1. = DIR. BASE.    2, 2
        MOV     AX,[BX]         ;CARGA NUM. ELEM.        2,10
        SHL     AX              ;MULTIPLICALO POR 2      2, 2
        ADD     SI,AX           ;APUNT2= ULTIMO ELEM.    2, 3
ACTAP:  MOV     DI,SI           ;HAZ APUNT2=APUNT1.      2, 2
        SUB     DI,DX           ;CALCULA APUNT. IK.      2, 3
        CMP     DI,BP           ;APUNT2 CON DIR BASE.    2, 3
        JZ      INICIO          ;IGUAL, VE AL INICIO.    2,16,4
        MOV     AX,[SI]         ;CARGA ELEM. I EN AX.    2,10
        CMP     AX,[DI]         ;COMPARA CON ELEM. IK    2,14
        JNL     DECAP           ;MAYOR, DECREM. APUNT.   2,16,4
        XCHG    AX,[DI]         ;NO, INTERCAMBIALOS.     2,26
        MOV     [SI],AX         ;INTERCAMBIALOS.         2,10
        INC     CX              ;PON BANDERA.            1, 3
DECAP:  ADD     BX,0004H        ;APUNT. BASE=2o. ELEM.   3, 4
        CMP     SI,BX           ;APUNT. BASE=2o. ELEM.   2, 3
        JNZ     ACTUAL          ;NO, VE A ACTUALIZA.     2,16,4
```

```
          MOV     BX,BP           ;CARGA DIRECC. BASE.      2, 2
          CMP     CX,0000H        ;HUBO INTERCAMBIOS.       3, 4
          JNZ     INICIO          ;SI, HAZ OTRA PASADA.     2,16,4
          CMP     DX,0002H        ;INCREMENTO= 2 ?          3, 4
          JNZ     INICIO          ;NO, VE AL INICIO.        2,16,4
          INT     20H             ;FIN                      2,51
ACTUAL:   SUB     BX,0004H        ;REGRESA APUNT. BASE.     2, 3
          SUB     SI,0002H        ;DEC. APUNT. 2 BYTES.     2, 3
          JMP     ACTAP           ;CALCULAR APUNTADORES.    3,15
DIREC:    DW      20
          DW      20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1

;INSTRUCCIONES 36
;CICLOS
;INICIO 21
;LAZO2 123 CON INTERCAMBIO
;LAZO2  84 SIN INTERCAMBIO
;LAZO1 96+LAZO2
;LAZO1 DESDE INICIO: HASTA DOS LINEAS ANTES ACTUAL:
;LAZO2 DESDE ACTAP: HASTA DOS LINEAS ABAJO ACTUAL:

;BYTES 77
```

```
!*************************************************************!
!*                                                           *!
!*   PROGRAMA DE REACOMODO POR METODO SHELL PARA Z8000       *!
!*                                                           *!
!*************************************************************!

!Este programa reacomoda un en orden ascendente una tabla   !
!localizada en memoria el primer elemento de la tabla       !
!indica el numero de elementos dentro de ella. La tabla     !
!queda en la misma posicion de memoria.                     !

!Descripcon de Registros:
!R2=Direccion Base DB!
!R3=I Primer Apuntador!
!R4=Incremento ID!
!R5=Bandera de intercambio K!
!R6=Segundo Apuntador IK!
!R7=Elemento a Comparar!
!R8=Direccion base + 2 DB+2!

SHELL1 MODULE
CONSTANT
        ADDR    :=%1500

GLOBAL
        BUSHELL PROCEDURE

ENTRY

                                                        !CICLOS,BYTES!
                                                                !
        LD      R2,#ADDR        !DIRECCION BASE TABLA EN R2 7,4!
        LD      R8,R2           !DIRECCION BASE TABLA EN R8 3,2!
        INC     R8,#2           !HAZ R8=DB+2                4,2!
        LD      R4,@R2          !NUMERO DE ELEMENTOS EN R4  7,2!
        SLL     R4,#1           !NE=NEX2 ID=NEX2      13+3n=16,4!
        LD      @R2,R4          !GUARDALO EN MEMORIA        8,2!
INICIO: LD      R3,@R2          !NUM. ELEM. X2=R3  I=NEX2   7,2!
        ADD     R3,R2           !I=NE+DB CONSIDERA DESPLAZ. 7,2!
        INC     R4,#1           !HAZ ID=(NE X 2) + 1        4,2!
        SRL     R4,#1           !ID=[(NE X 2)+1]/2   13+3n=16,4!
        BIT     R4,#0           !ID = NUM. PAR ?            4,2!
        JR      Z,IEXC          !SI, INICIALIZA BANDERA INT.6,2!
        INC     R4,#1           !NO, FORZALO A DIR. PAR     4,2!
IEXC:   CLR     R5              !INICIALIZA BANDERA EXC     7,2!
CALIK:  LD      R6,R3           !HAZ IK=(NE X 2)+DB         3,2!
        SUB     R6,R4           !HAZ IK=I-ID                4,2!
        CP      R6,R2           !ES IK<=DB ?                4,2!
        JR      LE,INICIO       !SI, LIM. INF.TABLA,INICIO  6,2!
        LD      R7,@R3          !NO, PON ELEMENTO (I) EN R3 7,2!
        CP      R7,@R6          !COMPARA ELEM. (I) CON (IK) 7,2!
        JR      GE,SIGPAR       !SI (I)>=(IK)COMP. SIG. PAR 6,2!
```

```
EXC:     EX      R7,@R6          !(I)<(IK) ENTONCES INTERC.  12,2!
         LD      @R3,R7          !PON ELEMENTO IK EN MEMORIA 8,2!
         INC     R5,#1           !PON BANDERA EXC            4,2!
SIGPAR:  DEC     R3,#2           !HAZ I=I-2                  4,2!
         CP      R3,R8           !ES I=DB+2                  4,2!
         JR      NZ,CALIK        !NO, VE A CALCULAR OTRO IK  6,2!
         TEST    R5              !SI, ES INTERC=0 ?          7,2!
         JR      NZ,INICIO       !NO, VE AL INICIO           6,2!
         CP      R4,#2           !SI, ES ID=2 ?              7,4!
         JR      NZ,INICIO       !NO, VE AL INICIO           6,2!
FIN:     HALT
END BUSHELL
END SHELL1

!CICLOS PARA CADA CASO!
!INICIO                                  45 CICLOS!
!LOOP2 (SIN INTERCAMBIO)                 51 CICLOS!
!LOOP2 (CON INTERCAMBIO)                 75 CICLOS!
!LOOP1 DE 68 A 81 + LOOP2 DEPENDIENDO DE
!ID Y DE SI HUBO INTERCAMBIO DE ELEMENTOS      !
!LOOP2 COMPRENDE DESDE CALIK: HASTA 2 LINEAS ABAJO SIGPAR:  !
!LOOP1 COMPRENDE DESDE INICIO: HASTA UNA LINEA ANTES FIN:   !

!BYTES 70!
!INSTRUCCIONES 32!
```

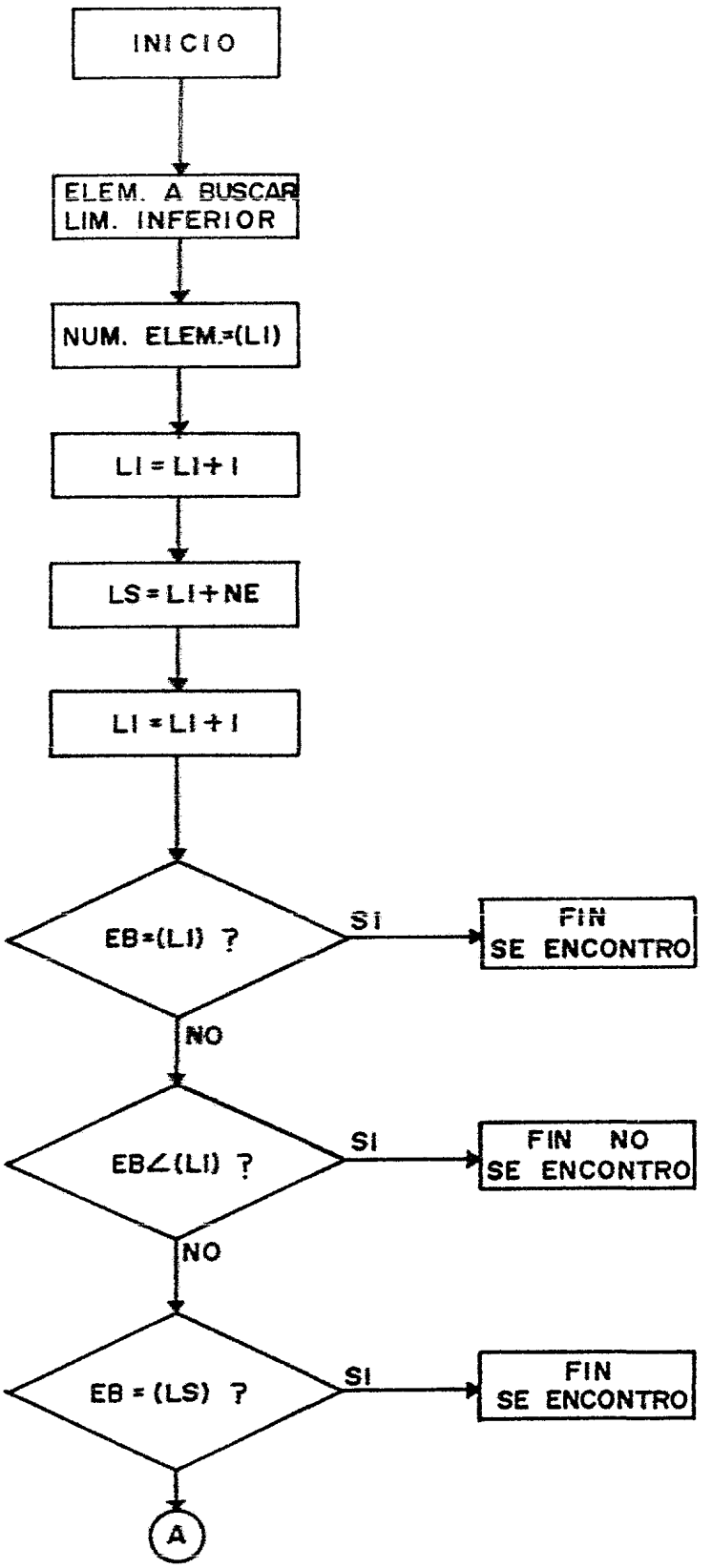
```
;*************************************************************
;*                                                           *
;*       PROGRAMA DE REACOMODO POR METODO SHELL              *
;*                        PARA 68000                         *
;*                                                           *
;*************************************************************

;      Este programa ordena una lista desordenada de palabras
;(16 bits), la cual se encuentra en memoria. El primer
;elemento de la lista contiene el numero de elementos que
;tiene.

;      Los registros se utilizan de la siguiente manera:


;     A0  DIRECCION DE LA TABLA
;     A1  PRIMER APUNTADOR
;     A2  SEGUNDO APUNTADOR
;     D0  DESPLAZAMIENTO
;     D1  BANDERA DE INTERCAMBIO
;     D2  ELEMENTO SE&ALADO POR EL SEGUNDO APUNTADOR


          COPY    SYS
          TYPE    <Direcion de la tabla :>
          KBD
          GTOCT                                          ;CICLOS, BYTES
          MOVL    D1,A4             ;DIRECCION DE LA TABLA      4,2
          MOVL    A4,A0             ;SALVA LA DIRECCION         4,2
          MOVW    (A4)+,D0          ;SACA EL # DE ELEMENTOS     4,2
          LSLW    D0,#1             ;MULTIPLICA POR DOS         8,2
          MOVL    D0,D1             ;SALVA EL RESULTADO         4,2
INI:      ADDW    #1,D0             ;INCREMENTALO               4,2
          LSRW    D0,#1             ;DIVIDELO ENTRE DOS         8,2
          BTST    #0,D0             ;ES PAR                     8,4
          BEQ     SIG               ;SI, NO PRENDAS BAND.(12) 10,4
          ADD     #1,D0             ;NO ENCIENDE BANDERA        4,2
SIG:      CLRW    D3                ; BANDERA DE INTERCAMBIO    4,2
          MOVL    A0,A1             ;INICIALIZA UN APUNTADOR    4,2
          ADDL    D1,A1             ;APUNTA HACIA LA P.MEDIA    8,2
          MOVL    A1,A2             ;INICIALIZA OTRO APUNT.     4,2
OTRA:     SUBL    D0,A2             ;APUNTA HACIA EL FINAL      8,2
          CMPL    A2,A0             ;APUNTADOR<QUE DIR.BASE     6,2
          BGE     INI               ;VE A INICIO          (12) 10,4
          MOVW    (A2),D2           ;SACA UN ELE.APUNTADO       4,2
          CMPW    D2,(A1)           ;COMPARA LOS ELE. APUNT.    6,2
          BCS     NOINT             ;SI INF>SUP NO INTER.(12) 10,4
          MOVW    (A1),(A2)         ;INTERCAMBIO                5,2
          MOVW    D2,(A1)           ;INTERCAMBIO                5,2
          ADDW    #1,D3             ;ENCIENDE BANDERA DE INT.   4,2
```

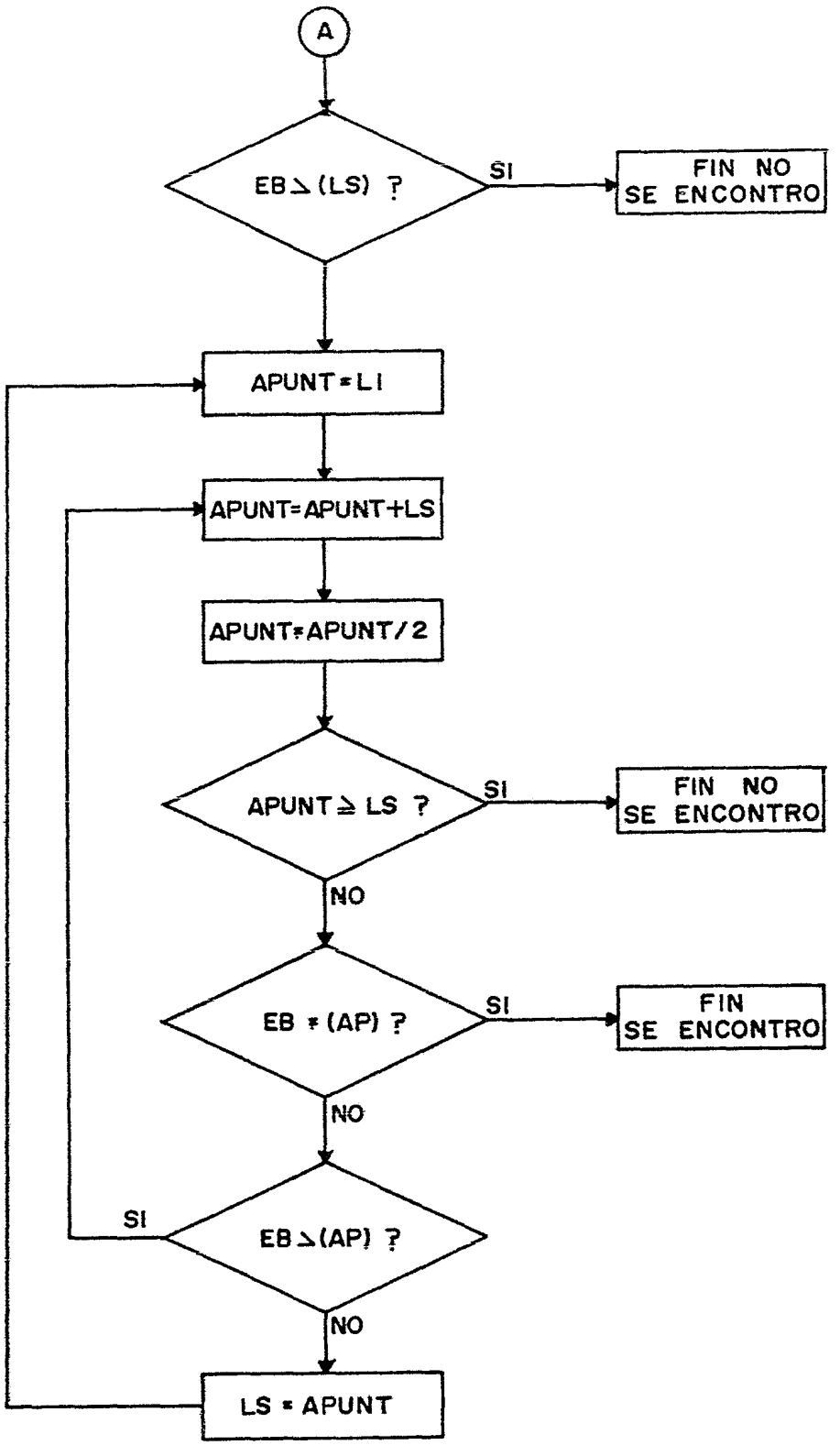
```
NOINT:  SUBL    #2,A1           ;APUNTA HACIA OTRO ELEM.    4,2
        CMPL    A4,A1           ;COMPARA APUNTADORES        6,2
        BNE     OTRA            ;SON DIF.REGRESA      (12) 10,4
        TSTW    D3              ;CHECA BANDERA DE INT.      4,2
        BNE     INI             ;ESTA PRENDIDA INICIO(12) 10,4
        CMPW    D0,#2           ;INCREMENTO = 2             6,2
        BNE     INI             ;NO, VE AL INICIO     (12) 10,4
        EXIT
        END


;NUMERO DE CICLOS PARA CADA CASO:

;                                CICLOS

;INICIO                          24
;LOOP 2   (SIN INTERC)           68
;LOOP 2   (CON INTERC)           80
;LOOP 1              LOOP 2 + 50 A 76
;DEPENDIENDO SI HUBO INTERCAMBIO O NO

;LOOP 2 COMPRENDE DESDE "OTRA" HASTA DOS
;INSTRUCCIONES DESPUES DE "NOINT"
;
;LOOP 1 COMPRENDE DESDE "INI" HASTA EL FINAL


;BYTES UTILIZADOS                74

;INSTRUCCIONES OCUPADAS          30
```

3.6.4 COMENTARIOS.- En este algoritmo, existe una diferencia un poco mayor que en el caso del algoritmo de burbuje, el procesador más rápido y el que ocupa menos instrucciones es el 68000 con 30 instrucciones, 74 bytes y 180 ciclos en el peor de los casos. Le sigue el Z8000 con 32 instrucciones, 70 bytes y 201 ciclos en el peor de los casos. El 8086 es el que ocupa mayor número de instrucciones (36), 77 bytes y 240 ciclos de reloj en el peor de los casos. Como podemos observar, el Z8000 y el 68000 no presentan una diferencia significativa en la ejecución de los algoritmos y las instrucciones son equivalentes una a una, el 8086 es un poco menos eficiente que los otros dos procesadores.

3.7 PROGRAMA DE BUSQUEDA BINARIA.- Este programa busca un elemento de ocho Bits en una lista ordenada ascendentemente, el número de elementos de la tabla es indicado por los dos primeros bytes de la tabla.

3.7.1 DESCRIPCION DEL DIAGRAMA DE FLUJO.- Este algoritmo revisa si el elemento a buscar está contenido en la tabla checando contra los límites de ésta, si el elemento está dentro de la tabla y no es igual a alguno de los elementos que se encuentran en los límites entonces se procede con la rutina de búsqueda la cual compara contra la parte central de la tabla, si el elemento buscado no es igual, se determina si está contenido en la parte superior o en la parte inferior y se cambia el límite respectivo. Este proceso se repite hasta que el elemento es encontrado o hasta que los límites de búsqueda son iguales con lo cual se determina que el elemento no está contenido en la tabla.


INICIO
ELEM. A BUSCAR
LIM. INFERIOR
NUM. ELEM.=(LI)
LI = LI+1
LS = LI+NE
LI = LI+1
EB=(LI) ?
SI
FIN
SE ENCONTRO
NO
EB∠(LI) ?
SI
FIN NO
SE ENCONTRO
NO
EB = (LS) ?
SI
FIN
SE ENCONTRO
A


A
EB ≻ (LS) ?
SI
FIN NO SE ENCONTRO
APUNT = LI
APUNT = APUNT + LS
APUNT = APUNT / 2
APUNT ≥ LS ?
SI
FIN NO SE ENCONTRO
NO
EB = (AP) ?
SI
FIN SE ENCONTRO
NO
SI
EB ≻ (AP) ?
NO
LS = APUNT

```
;*************************************************************
;*                                                           *
;*        PROGRAMA DE BUSQUEDA BINARIA PARA 8086             *
;*                                                           *
;*************************************************************

        ORG     100H
        PUT     100H
INICIO:                                                 ;BYTES,CICLOS
        MOV     BX,DIREC        ;DIREC. DE OPERANDOS.   3, 4
        MOV     SI,BX           ;LIM. INFERIOR EN SI.   2, 2
        MOV     DI,[BX]         ;NUM. ELEM. EN DI.      2,13
        INC     SI              ;INC. LIM. INFERIOR.    1, 2
        MOV     AX,SI           ;LIM. SUP. = LIM INF.   2, 2
        ADD     AX,SI           ;MAS  NUMERO DE         2, 3
        MOV     DI,AX           ;ELEMENTOS.             2, 2
        INC     SI              ;APUNTA 1er. ELEMENTO.  1, 2
        MOV     AL,DATO         ;CARGA ELEM. A BUSCAR.  2, 4
        MOV     BX,SI           ;LIMITE INF. EN BX.     2, 2
        CMP     AL,[BX]         ;E.B. = LIM. INF. ?     2,14
        JZ      FIN             ;SI, VE AL FIN.         2,16,4
        JL      NOEX            ;MENOR, NO EXISTE.      2,16,4
        MOV     BX,DI           ;CARGA LIM. SUP.        2, 2
        CMP     AL,[BX]         ;E.B. = LIM. SUP. ?     2,14
        JZ      FIN             ;SI, VE AL FIN.         2,16,4
        JNL     NOEX            ;MAYOR, A NO EXISTE.    2,16,4
APUNTA: MOV     BX,SI           ;AP. = LIM. INF.        2, 2
SUMA:   ADD     BX,SI           ;AP=LI+LS .             2, 3
        SHR     BX              ;AP=AP/2 .              2, 2
        CMP     BX,DI           ;AP > LS ?              2, 3
        JNL     NOEX            ;SI, A NO EXISTE.       2,16,4
        CMP     AL,[BX]         ;E.B. = [AP] .          2,14
        JZ      FIN             ;SI, VE AL FIN.         2,16,4
        JNL     SUMA            ;NO, E.B. > [AP] .      2,16,4
        MOV     DI,BX           ;APUNT. = LIM. INF.     2, 2
        JMP     APUNTA          ;VE AL PRINCIPIO.       3,15
NOEX:   MOV     BX,0FFFFH       ;INDICA NO EXISTE       3, 4
FIN:    INT     20H             ;FIN.                   2,51
DATO:   EQU     3
DIREC:  DW      9
        DB      1,2,3,4,5,6,7,8,9

;INSTRUCCIONES 28
;CICLOS :
;INICIO Y BUSQUEDA EN EXTREMOS 82
;LAZO1 53
;FINAL 4
;LAZO1 COMPRENDE DE APUNTA: A 1 LINEA ANTES NOEX:

;BYTES 57
```

```
!*************************************************************!
!*                                                           *!
!*      PROGRAMA DE BUSQUEDA BINARIA PARA Z8000              *!
!*                                                           *!
!*************************************************************!

!Este programa busca un byte dado en una tabla ordenada en  !
!orden ascendente retorna un 0 en R0 si el elemento se      !
!encontro en la tabla y un FFFF si no se encontro           !
!R2 contiene la direccion del elemento si se encontro       !
!Descripcion de Registros:  !
!RL0=Elemento a Buscar!
!R1=Direccion Base de la Tabla!
!R2=Apuntador a la Tabla!
!R3=Numero de Elementos en la Tabla!

BSERCH MODULE
CONSTANT
        ADDR    :=%1500
        EB      :=%05
GLOBAL
        BUSQUEDA PROCEDURE

ENTRY

                                                        !CICLOS,BYTES!
                                                                !
        LDB     RL0,#EB         !RL0=ELEMENTO A BUSCAR (EB)7,2!
        LD      R1,#ADDR        !R1=DIRECCION BASE DE TABLA7,4!
        LD      R3,@R1          !R3=NUMERO DE ELEMENTOS     7,2!
        INC     R1,#1           !CALCULA LIMITE SUPERIOR    4,2!
        ADD     R3,R1           !CALCULA LIMITE SUPERIOR    7,2!
        INC     R1,#1           !APUNTA PRIMER ELEM. TABLA  4,2!
        LD      R2,R1           !APUNTADOR R2=LIM. INF.     3,2!
        CPB     RL0,@R2         !EB=(LI)?                   7,2!
        JR      Z,FND           !SI, FINAL(SE ENCONTRO)     6,2!
        JR      ULT,NOTFND      !EB<(LI)? FINAL (NO ENC.)   6,2!
        LD      R2,R3           !APUNTADOR=LIMITE SUPERIOR  3,2!
        CPB     RL0,@R2         !EB=(LS)?                   7,2!
        JR      Z,FND           !SI, FINAL (SE ENCONTRO)    6,2!
        JR      UGT,NOTFND      !EB>(LS)? ,FINAL (NO ENC.   6,2!
MENOR:  LD      R2,R1           !HAZ APUNTADOR=LI   AP=LI   3,2!
SUMA:   ADD     R2,R3           !AP=AP+LS CALCULO PARTE     7,2!
        SRL     R2,#1           !AP=AP/2 MEDIA TABLA13+3n=16,4!
        CP      R2,R3           !AP>=LS                     4,2!
        JR      UGE,NOTFND      !SI,FINAL(NO SE ENCONTRO)   6,2!
        CPB     RL0,@R2         !NO, EB=(AP)?               7,2!
        JR      Z,FND           !SI, FINAL (SE ENCONTRO)    6,2!
        JR      UGT,SUMA        !NO, EB>(AP) BUSCA P. SUP.  6,2!
        LD      R3,R2           !NO, HAZ LS=AP P. INFERIOR  3,2!
        JR      MENOR           !REGRESA A HACER AP=LI      6,2!
NOTFND: LD      R0,#%FFFF       !R0=FFFF= NO SE ENCONTRO    7,4!
```

```
        JR      FIN             !VE AL FINAL                    6,2!
FND:    LD      R0,#0           !R0=0000= SE ENCONTRO           7,4!
FIN:    HALT
END BUSQUEDA
END BSERCH

!CICLOS PARA CADA CASO!
!INICIO + BUSQUEDA EN EXTREMOS          80 CICLOS!
!LOOP                                   64 CICLOS!
!FINAL                                  20 CICLOS!
!LOOP COMPRENDE DESDE MENOR: HASTA UNA LINEA ANTES NOTFND:!

!BYTES 62!
!INSTRUCCIONES 28!
```

```
;*************************************************************
;*                                                           *
;*                PROGRAMA DE BUSQUEDA BINARIA               *
;*                        PARA 68000                         *
;*                                                           *
;*************************************************************

;     Este programa busca un dato de un byte en una lista que
;se encuentra en memoria.
;     El  dato a buscar se coloca en un registro, y al correr
;el programa aparece un uno en ese registro para indicar  que
;el  elemento  fue  encontrado,  la  direccion  del  elemento
;aparece en el registro A2.
;
;     Los registros se utilizan de la siguiente manera:

;A1     DIRECCION DE LA LISTA Y LI  LIMITE INFERIOR
;A2     AP   APUNTADOR
;D0     EB   ELEMENTO A BUSCAR
;D3     LS   LIMITE SUPERIOR DE LA LISTA
;D4     REGISTRO INTERMEDIO

         COPY    SYS

         TYPE    DIRECCION BASE DE LA TABLA
         KBD                                          ;BYTES,CICLOS
         GTOCT
         MOVL    D1,A1           ;GUARDA LA DIRECCION       4,2
         TYPE    DATO A BUSCAR
         KBD
         GTOCT
         MOVB    D1,D0           ;GUARDA EL DATO            4,2
         MOVW    (A1),D3         ;COLOCA EL NUMERO DE ELE   4,2
         ADDL    #1,A1           ;INCREMENTA APUNTADOR      8,2
         ADDL    A1,D3           ;APUNTA AL FINAL           8,2
         ADDL    #1,A1           ;NORMALIZALO               8,2
         MOVL    D3,A2           ;APUNTADOR EN EL FINAL     4,2
         CMPB    D0,(A2)         ;VERIFICA SI ESTA          4,2
         BEQ     FND             ;SI VE A FIN         (12) 10,4
         BLT     NOTFND          ;ES MAYOR, NO ESTA   (12) 10,4
         CMPB    D0,(A1)         ;ESTA EN EL INICIO         4,2
         BEQ     FND             ;SI, VE AFIN         (12) 10,4
         BHI     NOTFND          ;ES MENOR, NO ESTA   (12) 10,4
MENOR:   MOVL    A1,A2           ;COLOCA APUNTADOR INICIO   4,2
SUMA:    ADDL    D3,A2           ;SUMA LS+LI                8,2
         MOVL    A2,D4           ;SACA LA MEDIA             4,2
         LSR     D4,#1           ;                          6,2
         MOVL    D4,A2           ;                          4,2
         CMPL    A2,D3           ;SEPARACION DE APUNTADORES 6,2
         BHI     NOTFND          ;SI  LI=LS  NO ESTA  (12) 10,4
         CMPB    D0,(A2)         ;ESTA EL ELEMENTO          4,2
         BEQ     FND             ;SI, VE A FIN        (12) 10,4
```

```
        BHI     SUMA            ;NO, REGRESA          (12) 10,4
        MOVL    A2,D3           ;MODIFICA EL LIMITE        4,2
        BR      MENOR           ;REGRESA                   10,4
NOTFND: TYPE    NO LO ENCONTRO
        BR      FIN
FND:    TYPE    SI LO ENCONTRO
FIN:    EXIT
        END

;NUMERO DE CICLOS PARA CADA CASO:
;                                          CICLOS

;INICIO + BUSQUEDA EN EXTREMOS               84
;LOOP                                        76
;FINAL                                       20
;
;LOOP COMPRENDE DESDE "MENOR" HASTA UNA LINEA
;ANTES DE NOTFND.

;BYTES UTILIZADOS                            68

;INSTRUCCIONES OCUPADAS                      27
```

3.7.4 COMENTARIOS.- En esta rutina se puede observar que el procesador más eficiente es el 8086 que ocupa 28 instrucciones, 57 bytes y 139 ciclos de reloj en el peor de los casos. Le sigue el Z8000 con 28 instrucciones, 62 bytes y 164 ciclos de reloj en el peor de los casos. El 68000 es el menos eficiente ya que ocupa 27 instrucciones, 68 bytes y 180 ciclos de reloj en el peor de los casos. En este programa se puede observar que el Z8000 y 68000 bastante menos eficientes que el 8086 debido a que su arquitectura está menos enfocada al manejo de bytes. En cambio, el 8086 que tiene una arquitectura compatible con la familia anterior 8080 esta un poco más orientado hacia el manejo de bytes.

3.8.1 COMPARACIONES OBTENIDAS DE OTRAS FUENTES.- Las siguientes comparaciones fueron obtenidas de una revista (19) y de una comparación proporcionada por la compañía Zilog Inc.

En la comparación de la revista EDN se evaluan los siguientes programas:

1. Manejo de Interrupciones.
2. Manejo de E/S con Procesamiento Tipo FIFO.
3. Búsqueda de una Cadena de Caracteres.
4. Prueba, Prendido y Borrado de Bits.
5. Inserción en Listas Ligadas.
6. Reacomodo Rápido.
7. Transposición de Matrices de Bits.

Los programas anteriores fueron desarrollados en la Universidad Carnegie- Mellon en 1976, el propósito de éstos es el de evaluar minicomputadoras y computadoras mayores. El primer programa efectúa el proceso sobre cuatro interrupciones que llegan al CPU en diferentes niveles. Los resultados obtenidos de la prueba fueron los siguientes:

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	55	126
Z8000	6	18	42
68000	10	24	33

En el segundo programa, el conjunto de interrupciones que se procesa es el siguiente:

Una Interrupción de Nivel 1
Una Interrupción de Nivel 2
Una Interrupción de Nivel 3
Una Interrupción de Nivel 4
Tres Interrupciones de Nivel 2 que forzan el almacenamiento de éstas
Cinco Interrupciones de Nivel 3

Los resultados obtenidos de este programa se muestran a continuación:

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	85	348
Z8000	6	106	436
68000	10	118	390

El tercer programa realiza la búsqueda de una cadena de caracteres dentro de una tabla, ésta está constituida por 120 caracteres y la cadena a buscar esta formada por la frase "HERE IS A MATCH".

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	70	193
Z8000	6	66	237
68000	10	44	244

El conjunto de datos para el cuarto programa está formado por un arreglo de 125 bits consistente de unos y ceros alternados, el arreglo comienza en un límite de palabra. Los resulatdos obtenidos son los siguientes:

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	46	122
Z8000	6	44	123
68000	10	36	70

En el quinto programa se realiza una inserción en listas ligadas, el conjunto de datos inicial está formado por una lista vacía, en la que serán insertados cinco elementos de 32 Bits. El número de bytes y el tiempo necesarios para la ejecución del algoritmo se muestran a continuación:

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	94	-
Z8000	6	96	237
68000	10	106	153

El sexto programa efectúa un reacomodo rápido sobre una lista de 102 elementos de 16 bits cada uno. Los resultados obtenidos son los siguientes:

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	347	115,669
Z8000	6	386	115,500
68000	10	266	33,527

El Séptimo programa efectúa una transposición de una matriz de 7 X 7 Bits. El arreglo comienza en un límite de palabra.

Procesador	Frec. Mhz	Bytes	Tiempo useg.
8086	10	88	820
Z8000	6	110	646
68000	10	74	368

Además de los programas mencionados anteriormente, se obtuvieron de Zilog Inc. los siguientes programas de traducción de caracteres, estos tienen por objeto el traducir una cadena de 1000 caracteres EBCDIC a ASCII. En estos programas se demuestra que el Z8000 es el más eficiente en esta tarea. Los resultados proporcionados son los siguientes:

Procesador	Frec. Mhz	Bytes	Ciclos de reloj
8086	10	17	5042
Z8000	10	16	1404
68000	10	26	3604

Los programas se listan a continuación:

```
8086:
        CLD
        MOV     CX,1000
        MOV     SI,STRING
        MOV     DI,SI
        MOV     DX,TABLA
LOOP:   LODSB
        XLAT
        STOSB
        LOOPNZ  LOOP
```

Z8000:

```
        LD      R3,#1000
        LD      R6,#STRING
        LD      R8,#TABLE
        TRIRB   @R6,@R8,R3
```

68000:

```
        MOVE.L  #1000,D3
        LEA.L   STRING,A1
        LEA.L   TABLA,A2
        CLR.L   D0
LOOP:   MOVE.B  (A1),D0
        MOVE.B  0(A2,D0),(A1)+
        DBF     D3,LOOP
```

4.1.1 8089 PROCESADOR DE E/S (INPUT OUTPUT PROCESSOR IOP) PARA EL PROCESADOR 8086

El 8089 es un procesador de E/S de propósito general que cuenta con dos canales, cada uno de los cuales combina los atributos de un Procesador con la flexibilidad de un controlador de DMA, cuenta con 50 tipos de Instrucciones y maneja el espacio de entrada/salida mapeado como memoria.

El 8089 está dividido internamente en varias unidades funcionales que se describen a continuación y se muestran en el diagrama siguiente:

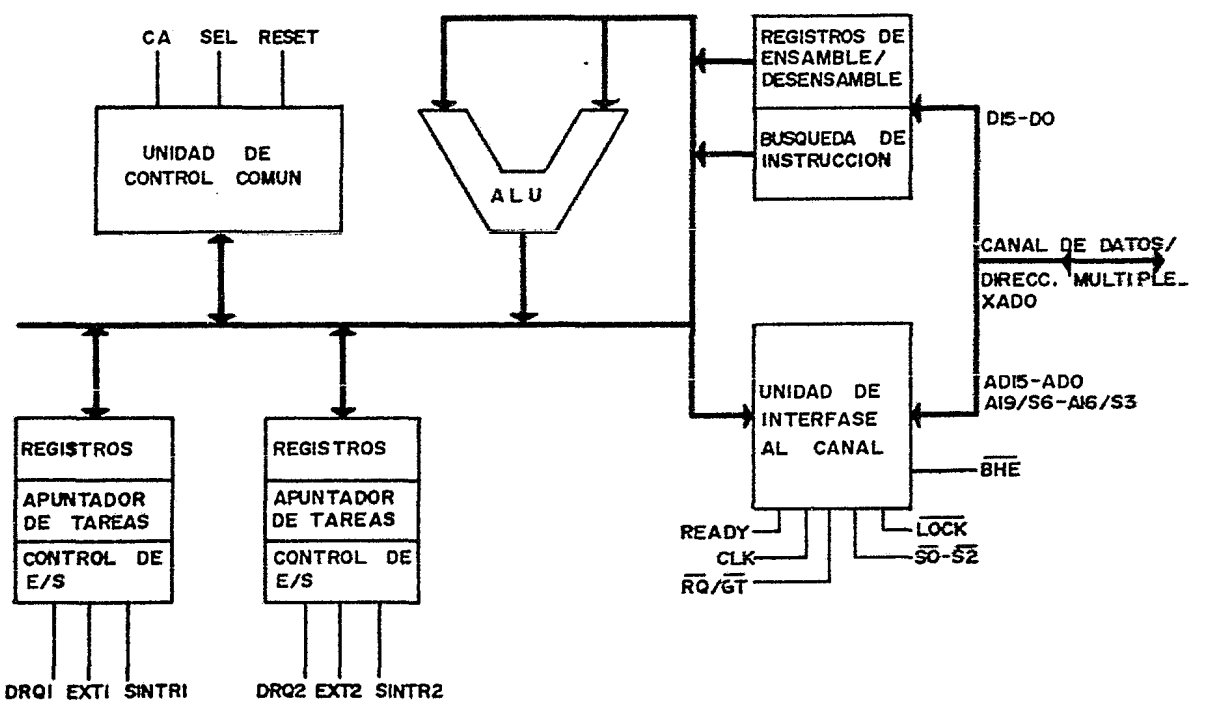


FIG 63 ESTRUCTURA INTERNA

Unidad de Control Común (CCU).- Esta unidad coordina las actividades internas del IOP.

Unidad Aritmética Lógica (ALU).- Ejecuta todas las operaciones aritméticas y lógicas del IOP, puede operar en números signados de 8 y 16 Bits.

Unidad de Búsqueda de Instrucciones (IFU).- Esta Unidad controla la búsqueda de instrucciones para el canal que está ejecutando una operación.

Unidad de Interfase con el Canal (BIU).- Esta unidad ejecuta todos los ciclos de Canal, transferencias de instrucciones y datos entre el IOP y periféricos o memoria.

Además de las Unidades mencionadas anteriormente el IOP cuenta con los siguientes recursos:

REGISTROS.-Cada Canal cuenta con los siguientes Registros:

Registros de Propósito General GA, GB y GC.- Son usados para almacenamiento de datos o como apuntadores durante operaciones de DMA.

Apuntador de Tareas.- Este Registro es cargado del Bloque de Parámetros cuando se inicia un Programa de Canal, es usado como Apuntador de Programa durante la ejecución de éste.

Apuntador de Bloque de Parametros.- Este Registro es cargado con la dirección del Bloque de Parámetros al iniciar un programa, puede ser utilizado como Apuntador al Bloque de Parámetros durante la ejecución de un programa.

Indice.- Es usado como un registro general durante la ejecución de programas, también puede usarse como índice para direccionar operandos en memoria.

Contador de Bytes.- Es utilizado como contador de Bytes durante las operaciones de DMA, puede ser usado opcionalmente.

Máscara/Comparacion.- Puede ser usado como registro general o como Registro de Comparación para Bytes en transferencias de DMA, contiene el valor a comparar en el Byte menos significativo y la Máscara de Comparación en el Byte más significativo.

Control de Canal.- El cual es usado para controlar las transferencias DMA el programa del canal carga este Registro con los valores apropiados antes de iniciar la tranferencia.

Palabra de Status de Programa.- Almacena el Status del Canal respectivo y los Bits Etiqueta los cuales indican si los Registros GA, GB y GC serán usados como apuntadores al espacio de direcciones de Sistema o al de E/S.

MODOS DE OPERACION.- El 8089 tiene dos modos básicos de operación que son: Modo Local y Modo Remoto. En el Modo Local, el Procesador y el IOP residen en el mismo Canal de Sistema Local compartiendo la memoria y los dispositivos de E/S. En el Modo Remoto, el Canal del IOP esta físicamente separado del Canal de Sistema, el IOP mantiene su propio Canal Local y puede operar fuera de la Memoria de Sistema. Sólo uno de los Canales puede operar a la vez aunque el IOP puede ser configurado para efectuar un intercambio de canal a la terminación de cada ciclo de máquina.

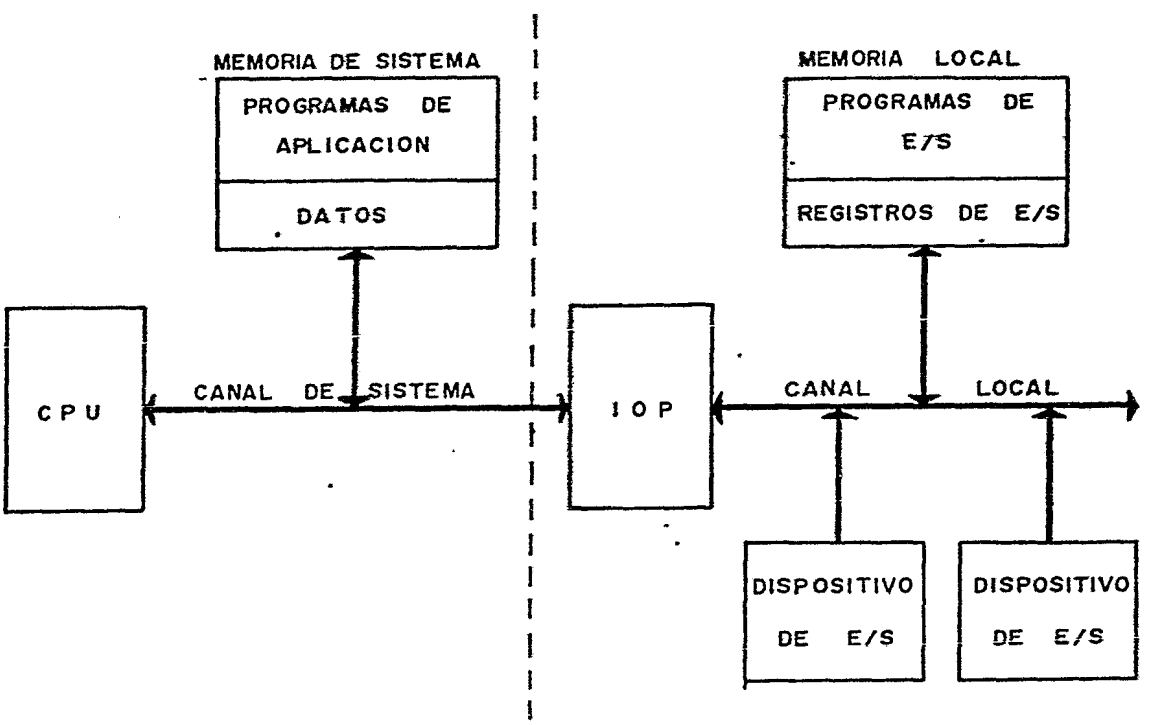


FIG 64 SISTEMA DE MULTIPROCESO

MECANISMO DE COMUNICACION.- La comunicación entre el Procesador y el IOP se efectúa fundamentalmente a través de mensajes en una memoria compartida, el Procesador puede hacer que el 8089 ejecute un programa colocándolo en el espacio de memoria del IOP y/o afirmando la línea Atención de Canal activando así el Canal adecuado, la línea SELECCION indica que canal esta siendo direccionado. La comunicación entre el IOP y el Procesador se efectúa de una manera similar a través de las líneas de Interrupción de Sistema, además, el IOP se puede comunicar colocando mensajes en la memoria sin importar su estado o el de los periféricos.

El Procesador puede indicar dos tipos de operaciones al IOP: Inicialización y Comandos. En la Inicialización el Procesador indica al IOP como están configurados los Canales de Datos y Direcciones del Sistema y como manejar el Canal a controlar, además indica la dirección del Bloque de Control el cual es común a ambos canales, una mitad de este Bloque está dedicada a cada Canal del IOP. En las Operaciones de Comandos el Procesador coloca una Palabra de Comando de Canal en el Bloque de Control que se encuentra en el espacio de memoria del Procesador, en este bloque se centran todas las comunicaciones entre el Procesador y los Canales del IOP, existen comandos para arrancar y detener programas, remover Solicitudes de Interrupción,etc.

Además del Bloque de Control de Canal existen el Bloques de Parámetros y el Bloque de Tareas en los cuales se proporcionan a los Canales del IOP los parámetros utilizados en los programas y el Programa a ser ejecutado respectivamente, existe un Bloque de este tipo para cada uno de los Canales del IOP. El Bloque de Parámetros puede ser utilizado también por el IOP para entregar resultados al procesador.

TRANSFERENCIAS DE DMA.- Los bloques de datos de hasta 64 Kbytes pueden ser transferidos entre dos direcciones cualesquiera de memoria o Puertos de E/S, la transferencia puede terminar en cuanto un número específico de Bytes ha sido transferido o cuando existe Igualación o No Igualación de Datos en el Registro de Comparación.

Usando el IOP en un sistema configurado en modo máximo con el 8086, se tiene la ventaja de que se desahoga en gran parte el trabajo del manejo de periféricos, ya que todas las tareas que tendría que realizar en un modo de operación mínimo las ejecutará el IOP sin requerir la atención del procesador en la mayoría de las operaciones, también brinda la característica de manejo de Acceso Directo a Memoria con lo que se obtiene un sistema bastante compacto y eficiente.

4.1.2 8087 PROCESADOR DE DATOS NUMERICOS (NUMERIC DATA PROCESSOR NDP) PARA EL PROCESADOR 8086

El 8087 es un procesador de datos numéricos que ejecuta operaciones aritméticas, de comparación y funciones trascendentes sobre varios tipos de DATOS EN PUNTO FLOTANTE. CUENTA CON REGISTROS DE 80 BITS CADA UNO.

El NDP esta dividido internamente en dos elementos de proceso que son: Unidad de Control (CU) y la Unidad de Ejecución Numérica (EU). La Unidad de Control busca instrucciones, lee y escribe datos en Memoria y ejecuta las instrucciones de Control del Procesador; la Unidad de Ejecución Numérica procesa todas las instrucciones numéricas del Procesador. Estas dos unidades son capaces de operar independientemente una de la otra, con lo cual se permite que la Unidad de Control mantenga la sincronía con el Procesador mientras que la Unidad de Ejecución Numérica procesa instrucciones.

Las instrucciones del NDP están mezcladas con las instrucciones del Procesador. El NDP monitorea las líneas de Status del Procesador y es capaz de determinar cuando una instrucción es buscada por éste, mediante este proceso el NDP decodifica las instrucciones simultáneamente y así detecta las instrucciones que le corresponden, éstas son identificadas a través de un patrón en los primeros cinco bits del código de operación el cual es llamado Código de Escape a Coprocesador. Todas las instrucciones que no tengan este patrón de bits serán ignoradas por la Unidad de Control del NDP.

REGISTROS.- El NDP cuenta con una Pila de ocho Registros de 80 bits los cuales son utilizados para hacer todas las operaciones Aritméticas y Trascendentes, cada uno de los registros está dividido en los campos mostrados en la figura 66.

Además de estos Registros el NDP contiene una palabra de Status de 16 bits la cual indica la condición general del Procesador, una palabra de Control de 16 bits la cual especifica las diferentes opciones de procesamiento del NDP, una palabra de Etiquetas la cual indica el contenido de cada uno de los registros del procesador y los Apuntadores de Excepciones los cuales son usados para proporcionar apuntadores a las rutinas de manejo de excepciones del usuario.

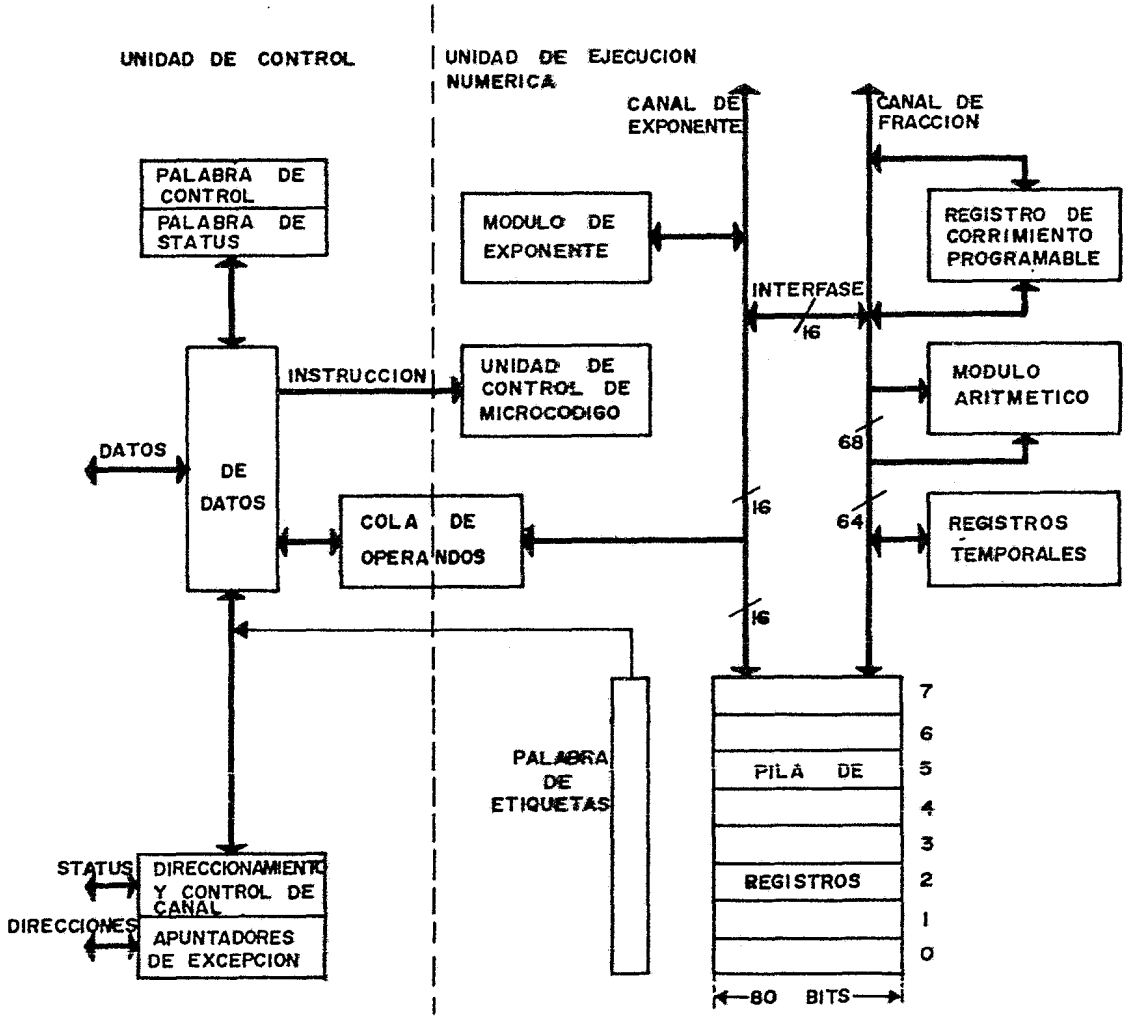


FIG 65 ESTRUCTURA INTERNA

TIPOS DE DATOS.- El NDP soporta 7 tipos de datos divididos en tres clases que se describen a continuación:

Enteros Binarios.- Los tres tipos de números enteros binarios son iguales exceptuando su longitud la cual gobierna el rango que podrá ser representado en cada formato, los tres formatos disponibles son de 16, 32 y 64 bits. El Bit más significativo es interpretado como el signo del operando, los números negativos son representados en complemento a dos.

Enteros Decimales.- Los enteros decimales son representados en notación Decimal "Empaquetada" en la cual dos dígitos decimales son colocados en cada Byte con excepción del Byte más significativo el cual contiene el Bit de signo, esta notación tiene una longitud de 80 Bits.

Números Reales.- El NDP almacena números reales en un formato binario de tres campos, los dígitos significativos del operando son almacenados en el campo para significando, el campo de exponente fija la posición del punto binario dentro de los dígitos significativos y el campo de signo especifica el signo del número.

Excepciones.- Durante la ejecución de instrucciones el NDP checa 6 clases de condiciones de excepción que son: Operación Inválida que ocurre cuando se trata de cargar un registro que no esta vacío, sacar un operando de un registro vacío, especificar un operando no numérico o una operación indeterminada; División entre Cero; Registro Denormalizado la cual ocurre cuando se trata de manejar un operando que no ha sido denormalizado; Sobrepaso; (underflow) y Precisión la cual indica que el resultado de una operación ha sido "Redondeado".

Cuando una excepción es detectada, esta es indicada por el 8087 encendiendo la bandera respectiva en el registro de status. Inmediatamente después checa la máscara de excepciones en el registro de control para determinar si deberá manejar la excepción internamente o si deberá enviar una interrupción al procesador huésped para que la excepción sea atendida por una rutina escrita por el usuario.

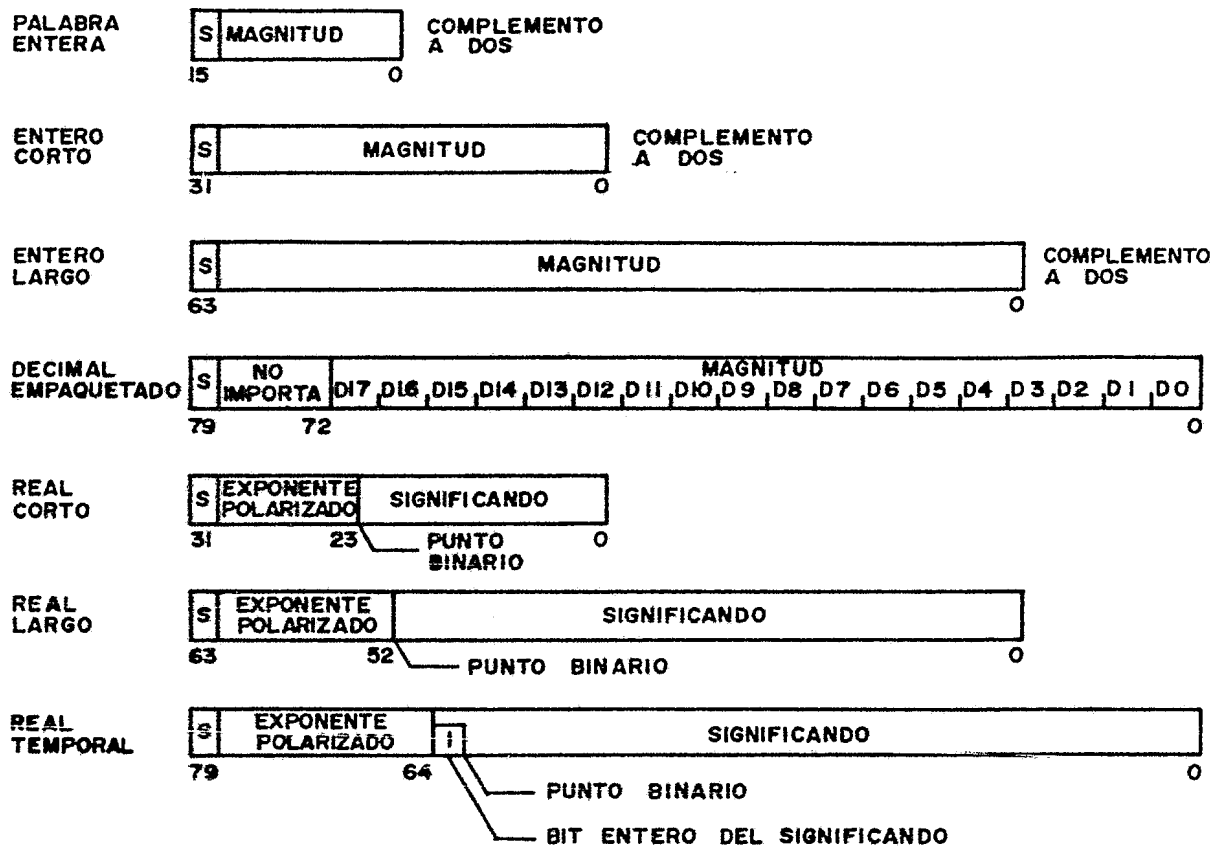


FIG 66 FORMATOS DE DATOS

4.2.1 UNIDAD DE MANEJO DE MEMORIA (MEMORY MANAGEMENT UNIT MMU) PARA EL PROCESADOR Z8000

La Unidad de Manejo de Memoria controla los espacios de dirección de 8Mb del Z8000. Proporciona relocalización dinámica de segmentos además de numerosas características de protección de memoria. La relocalización dinámica de segmentos hace que las direcciones del usuario sean independientes de las direcciones físicas de memoria. La Unidad de Manejo de Memoria usa una tabla de traslación para transformar la dirección lógica de 23 bits proporcionada por el Procesador en una dirección física de 24 bits. Cada Unidad de Manejo de Memoria puede manejar un máximo de 64 segmentos. Las direcciones de E/S y datos, en general, no deberán usar este dispositivo.

Los segmentos son variables en tamaño desde 256 hasta 64 Kbytes en incrementos de 256 bytes. Pares de MMUs soportan los 128 números de segmento disponibles para los diferentes espacios de dirección. Se pueden utilizar varias MMUs para soportar tablas de traslación múltiples dentro de cada espacio de direcciones. A cada segmento se le asignan un cierto número de atributos cuando su descripción es puesta en la Unidad de Manejo de Memoria, los segmentos son protegidos de acuerdo al modo en que pueden ser utilizados, como sólo Lectura, sólo Sistema, sólo Ejecución, sólo Acceso del Procesador o sólo Acceso mediante DMA.

LA UNIDAD DE MANEJO DE MEMORIA ES CONTROLADA MEDIANTE 22 INSTRUCCIONES ESPECIALES DE ENTRADA/ SALIDA EN MODO DE SISTEMA, las cuales tienen ciclos de máquina que causan que las líneas de Status del Procesador indiquen una operación de Entrada/Salida Especial en ejecución. Con estas instrucciones, el sistema puede asignar segmentos a localidades de memoria arbitrarias restringiendo el uso de ellos.

PROTECCION DE MEMORIA.- En el momento que se hace una referencia a memoria la Unidad de Manejo de Memoria compara los atributos de la solicitud de acceso con los atributos del segmento, si hay diferencia el acceso no es permitido y se genera una trampa de segmentación. Además de generar la trampa la MMU proporciona la señal SUPRESS (SUP) la cual puede ser usada por circuitería externa para inhibir los accesos a memoria. La solicitud de trampa informa al Procesador y al programa de control del sistema de la violación de manera que se pueda ejecutar la acción necesaria para recobrar al sistema.

REGISTROS DE LA UNIDAD DE MANEJO DE MEMORIA.- La Unidad de Manejo de Memoria contiene 3 tipos de Registro: de Descripción de Segmento, de Control y de Status.

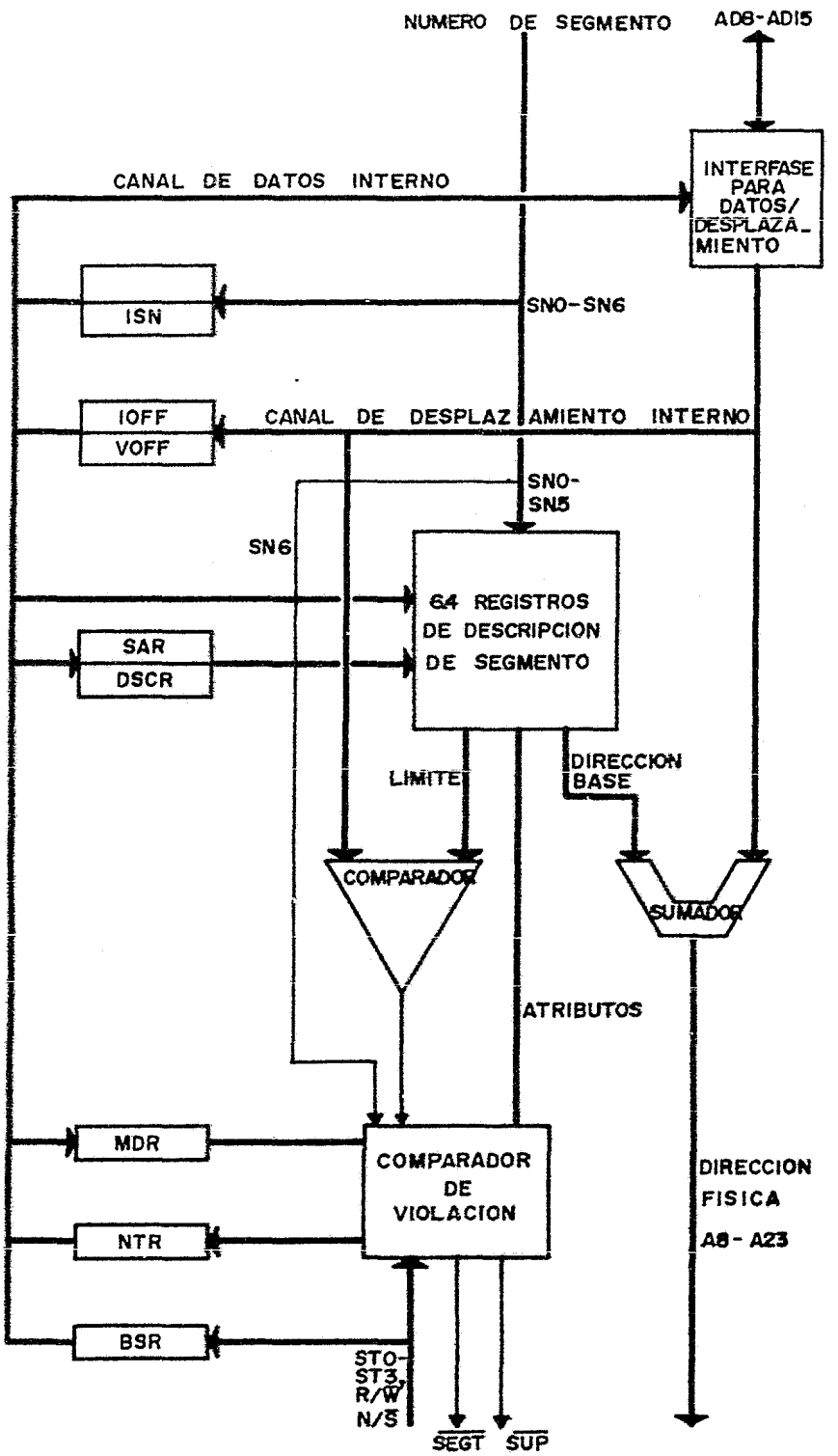


FIG 67 ESTRUCTURA INTERNA

REGISTROS DE DESCRIPCION DE SEGMENTO.- Este grupo está formado por 64 registros de 32 bits los cuales proveen la información necesaria para mapear las direcciones lógicas de memoria en direcciones físicas. El número de segmento de una dirección lógica determina cuál Registro de Descripción es usado en la traslación de direcciones.

Cada uno de los 64 Registros de Descripción de Segmento contiene un campo de 16 bits para la Dirección Base Física, un campo de 8 bits para Límites el cual especifica el número de bloques de 256 Bytes asignados al segmento y un campo de 8 bits para Atributos que contiene 8 banderas, seis de atributos y dos de Tipo de Acceso: sólo Lectura, sólo Sistema, Inhibición de Procesador, sólo Ejecución, Inhibición de DMA, Aviso de Escritura, de Cambio y de Referencia.

La bandera de Aviso en Escritura es usada para definir un segmento en el cual las direcciones están organizadas en orden descendente y además se genera una trampa cuando se efectúa una escritura en los últimos 256 Bytes del mismo.

Las banderas de Cambio y de Referencia indican si el segmento ha sido escrito o si se ha efectuado una referencia al mismo respectivamente.

REGISTROS DE CONTROL.- Es un grupo de 3 Registros de 8 bits, cada uno los cuales es usado para la programación de la MMU, estos Registros son accesibles al usuario, estos Registros son: Registro de modo que se utiliza para habilitar selectivamente las MMUs en configuraciones múltiples; el Registro de Dirección de Segmento el cual selecciona un Registro de Descripción de Segmento para ser accesado durante una operación de control; el Registro Contador de Registros de Descripción que es usado para apuntar a un byte dentro del Registro de Descripción de Segmento para ser accesado durante una operación de control.

REGISTROS DE STATUS.- Estos Registros contienen información útil para recuperar al sistema de violaciones en el acceso a memoria, el Registro de Tipo de Violación describe las condiciones que generaron la trampa, el Registro de Número de Segmento con Violación y el Registro de Desplazamiento con Violación guardan los 15 bits más significativos de la Dirección Lógica de la última instrucción que fué leída antes del primer acceso violatorio.

DEFINICION DE SEGMENTOS EN EL MMU.- Los segmentos son especificados por una Dirección Base Física de la cual se especifican los 16 Bits más significativos en el campo de Dirección Base Física, además se especifica el número de

bloques de 256 Bytes asignados al segmento en el campo para Número de Bloques y se especifican los atributos del segmento en el campo corespondiente.

PROCEDIMIENTO DE TRASLACION DE DIRECCIONES.- Las Direcciones Físicas son obtenidas de la siguiente manera: el número de segmento proporcionado en la Dirección Lógica es usado como un Apuntador dentro de la Tabla de Registros de Descripción de Segmento, así se obtienen los 16 Bits más significativos de la Dirección Base Física, éstos son sumados a los 8 bits más significativos del Desplazamiento especificado en la Dirección Lógica y con esto se obtienen los 16 Bits más significativos de la Dirección Física, los ocho Bits menos significativos de esta Dirección son tomados directamente del desplazamiento proporcionado por el procesador.

OPERACION DE DMA.- Las operaciones de DMA pueden ocurrir entre los ciclos de instrucción del Procesador y pueden ser manejados a través de la Unidad de Manejo de Memoria. La Unidad de Manejo de Memoria permite DMA en Modo Normal y en Modo Sistema. Para cada acceso a memoria, los atributos del segmento son checados y si se detecta una violación, se activa la señal SUPRESS. De manera diferente a las violaciones de Procesador, las cuales causan que la señal SUPRESS sea generada en los siguientes accesos a memoria hasta el principio de la siguiente instrucción, las violaciones de DMA causan que la señal SUPRESS sea generada sólo en base a cada acceso a memoria.

El periférico de DMA deberá reconocer la señal SUPRESS y guardar suficiente información para habilitar al sistema para recobrarse del acceso violatorio. Nunca se genera una solicitud de trampa durante DMA debido a que el procesador no puede reconocerlas, debido a esto, las condiciones de aviso de escritura no son señaladas.

Una Unidad de Manejo de Memoria es conectada al Z8000 de la manera mostrada en el siguiente diagrama a bloques:

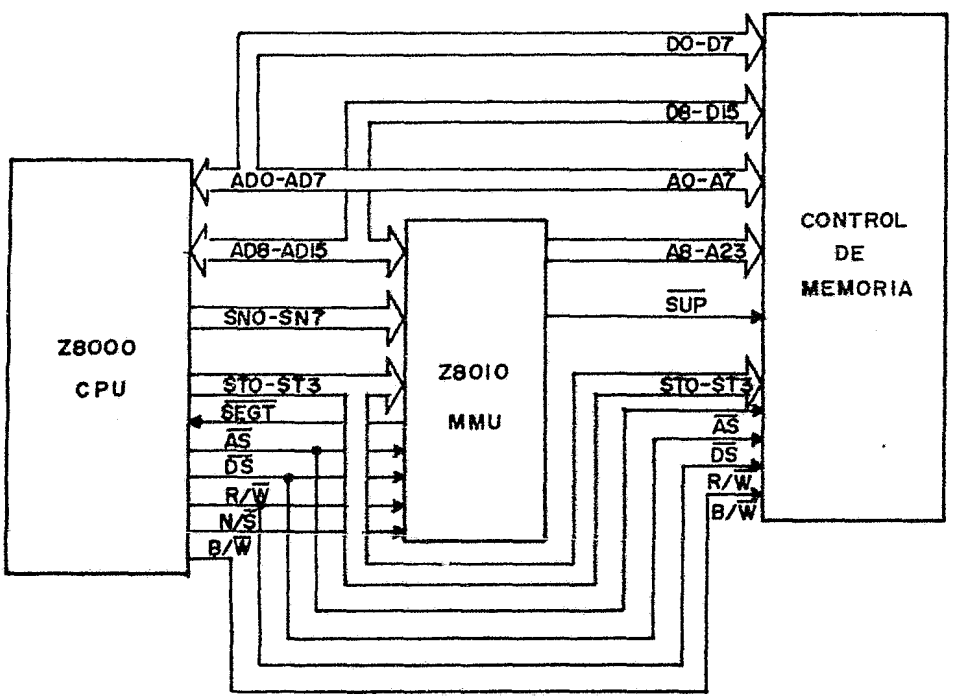


FIG 68CONFIGURACION EN UN SISTEMA

4.2.2 Z8090 CONTROLADOR UNIVERSAL DE PERIFERICOS (UNIVERSAL PERIPHERAL CONTROLLER UPC) PARA EL PROCESADOR Z8000

El Z8090 es un controlador de periféricos inteligente, descarga al procesador huésped asumiendo tareas tradicionalmente efectuadas por éste, como efectuar operaciones aritméticas, de traslación o de formateo de datos, y controlar dispositivos de E/S. Está basado en la arquitectura y conjunto de instrucciones del microcomputador Z8, el UPC contiene 2K de memoria de programa interno, un grupo de Registros de 256 bytes, tres Puertos de E/S y dos Contadores/Relojes.

Una característica importante del UPC es el grupo de Registros interno el cual contiene Registros de Control y Registros para los puertos de E/S accesados directamente por el programa del UPC e indirectamente por el CPU maestro. Una estructura como esta permite al usuario acomodar tantos registros de propósito general para datos entre el CPU y los periféricos como requiera su aplicación.

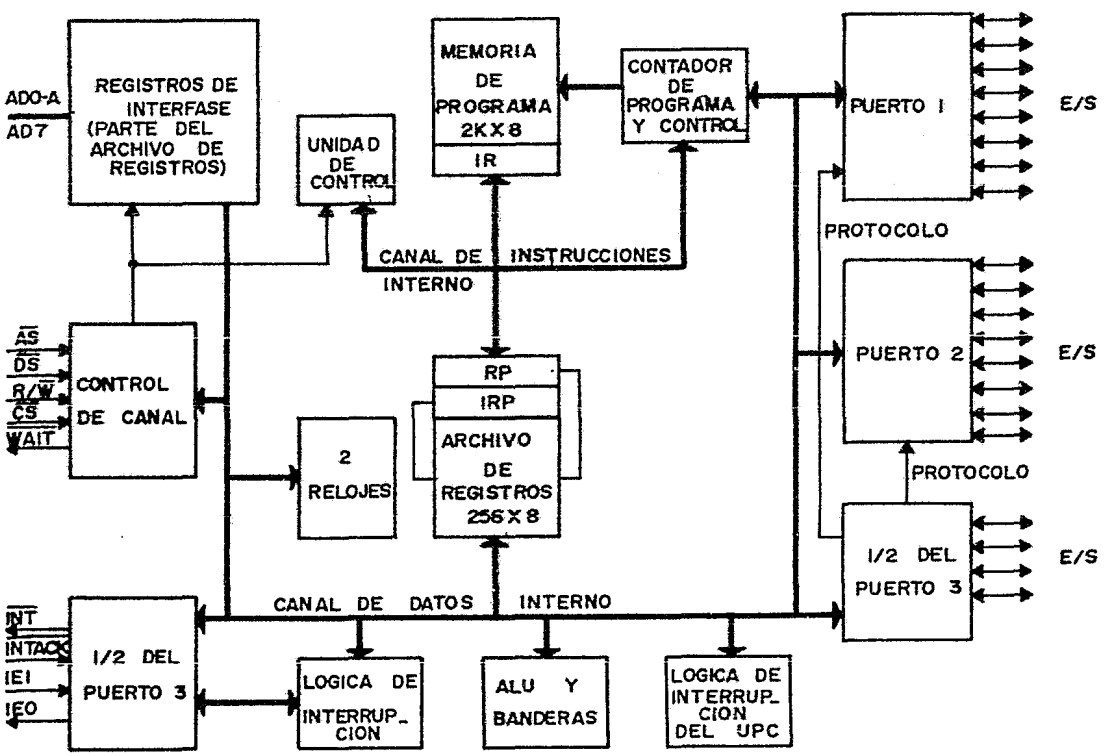


FIG 69 ESTRUCTURA INTERNA

DESCRIPCION DE REGISTROS.- Este grupo incluye tres registros de los puertos de E/S, 234 registros de propósito general y 19 registros de Control,Status y E/S Especial. De los Registros del UPC, 19 pueden ser manejados directamente por el CPU maestro; los otros son accesados indirectamente a través del mecanismo de transferencia de bloques.

El Procesador Maestro tiene dos maneras de accesar el Grupo de registros del UPC. Acceso Directo y Acceso de Bloques.

Acceso Directo.- Los Registros de Control de Transferencia de Datos, de Vector de Interrupción Maestro y de Control Maestro de Interrupcciones están Mapeados Directamente en el espacio de dirección del Procesador. También el Procesador tiene acceso directo a 16 registros conocidos como de Datos, Status y Comandos, este grupo de Registros puede ser un Bloque cualquiera de 16 registros contiguos del Grupo de Registros.

Acceso de Bloques.- El Procesador Maestro puede transmitir y recibir bloques de datos. Cuando el CPU accesa al bloque de datos el registro del UPC apuntado por el Registro de Dirección de Entrada es accesado e incrementado; el Registro de Límite de Cuenta es decrementado y usado para controlar el número de bytes que son transferidos durante estos accesos.

Los Registros de E/S y los Registros de Control están incluidos en el Grupo de Registros sin ninguna distinción, esto permite que cualquier instrucción del UPC procese información de E/S o de Control, con lo cual se elimina la necesidad de instrucciones especiales para E/S y Control.

PUERTOS DE E/S.- El UPC tiene 24 líneas que pueden ser dedicadas a funciones de E/S. Estan agrupadas lógicamente en tres puertos de 8 líneas cada uno, los cuales pueden ser programados en varias combinaciones de líneas de entrada y salida, con o sin protocolo. Bits individuales de los puertos 1 y 2 pueden ser configurados como entradas o salidas programando el Registro de Modo del Puerto respectivo. Los puertos 1 y 2 son manejados en el programa del UPC por los Registros Generales 1 y 2. Estos puertos pueden ser puestos bajo control de protocolo. El puerto 3 cuenta con cuatro líneas fijas de entrada y cuatro de salida. Puede ser configurado como de Entrada/Salida o como de Control programando el Registro de Modo del Puerto 3.

CONTADORES/RELOJES.- El UPC contiene 2 Contadores/Relojes programables de 8 bits, cada uno con un preescalador interno de 8 bits. Ambos Contadores/Relojes operan independientemente de la secuencia de instrucciones

del procesador. Los Registros de los preescaladores para el Contador/Reloj T0 y para el T1 pueden ser programados para dividir la frecuencia de entrada por cualquier número entre 1 y 64. Un Registro Contador es cargado con un número de 1 a 256, el Contador correspondiente es decrementado desde este número cada vez que el preescalador llega al fin de cuenta. Cuando la cuenta está completa el Contador hace una solicitud de interrupción. Los Contadores y los Preescaladores pueden ser leídos en cualquier momento sin modificar sus valores o cambiar sus cuentas.

MANEJO DE INTERRUPCIONES.- El UPC puede ser interrumpido por el Procesador Maestro, por los tres puertos y por los Contadores/Relojes. Las interrupciones pueden ser mascaradas y habilitadas o deshabilitadas globalmente usando el Registro de Máscara de Interrupción. El Registro de Prioridad de Interrupciones especifica el orden de prioridad.

4.2.3 Z8070 UNIDAD DE PROCESAMIENTO ARITMETICO (ARITHMETIC PROCESSING UNIT APU) PARA EL PROCESADOR Z8000

La Unidad de Procesamiento Aritmético es una Unidad de Procesamiento Extendido diseñada para ejecutar operaciones aritméticas de Punto Flotante funcionando en paralelo con el Procesador. Monitoreando el flujo de instrucciones que llega al Procesador, puede identificar y ejecutar las instrucciones que le corresponden. Esta unidad se puede usar con la Arquitectura de Procesamiento Extendido o puede ser integrada en Sistemas de otros Procesadores utilizando una interfase de propósito general.

La APU soporta tres formatos Binarios de Punto Flotante y tres formatos Enteros incluyendo uno para enteros BCD. Todas las manipulaciones numéricas internas utilizan un Patrón de Punto Flotante de 80 bits, pero las transferencias entre los registros del APU y el Procesador o Memoria pueden elegir el Patrón deseado especificándolo en la instrucción de Punto Flotante.

Las operaciones que efectúa esta unidad son: Suma, Substracción, Multiplicación, División, Raíz Cuadrada y Operaciones de Comparación. Además Puede efectuar conversiones entre los diferentes formatos de Punto Flotante y entre enteros binarios y números con Punto Flotante.

4.3.1 UNIDAD DE MANEJO DE MEMORIA (MEMORY MANAGEMENT UNIT MMU) PARA EL PROCESADOR 68000

La Unidad de Manejo de Memoria (MMU) proporciona traslación de direcciones y protección para el espacio de direcciones de 16 Mbytes del 68000. El Procesador en su ciclo de máquina proporciona un código de función el cual es utilizado para seleccionar el espacio de direcciones deseado, éste es dividido así en espacios de Modo Usuario y de Modo Supervisor y éstos a su vez se dividen en Datos y Programas. La separación de espacios de direcciones es la base para el manejo de memoria y la protección para el sistema operativo.

La Unidad de Manejo de Memoria particiona el espacio de direcciones lógicas dentro de bloques contiguos llamados segmentos. Cada segmento es una sección del espacio de Direcciones Lógicas el cual es mapeado por la Unidad de Manejo de Memoria dentro de un espacio de direcciones físicas. Los segmentos pueden ser definidos como de Modo Usuario o como de Modo Supervisor, sólo para datos o sólo para programa o para ambos, podrán ser usados sólo por una tarea o bien ser compartidos entre dos o más tareas. Cualquier segmento podrá tener protección contra escritura. Una trampa será generada si se trata de escribir a un segmento con protección o si se trata de accesar a un segmento indefinido.

El código de función proporcionado por el Procesador define un espacio de direcciones único y dentro de ese espacio pueden existir un determinado número de tareas, cada una de ellas necesita un Número de Espacio de Dirección (ASN) para distinguirla de otras. Los números de Espacio de Dirección son almacenados en la Tabla de Espacios de Direcciones (AST), esta tabla contiene información para cada posible código de función.

DESCRIPCION DE LOS REGISTROS.-Los registros de la Unidad de Manejo de Memoria están divididos en dos grupos : 32 Registros de Descripción de Segmento de nueve bytes cada uno y 9 Registros de Sistema. Cada uno de los Registros de Descripción define un Segmento de Memoria.

REGISTROS DE DESCRIPCION DE SEGMENTO.- Cada uno de ellos esta constituido por los siguientes Registros:

Dirección Base Lógica (16 Bits).- Este Registro define junto con la Máscara de Dirección Lógica el rango de direccionamiento lógico de un segmento, generalmente contiene la primera dirección del segmento.

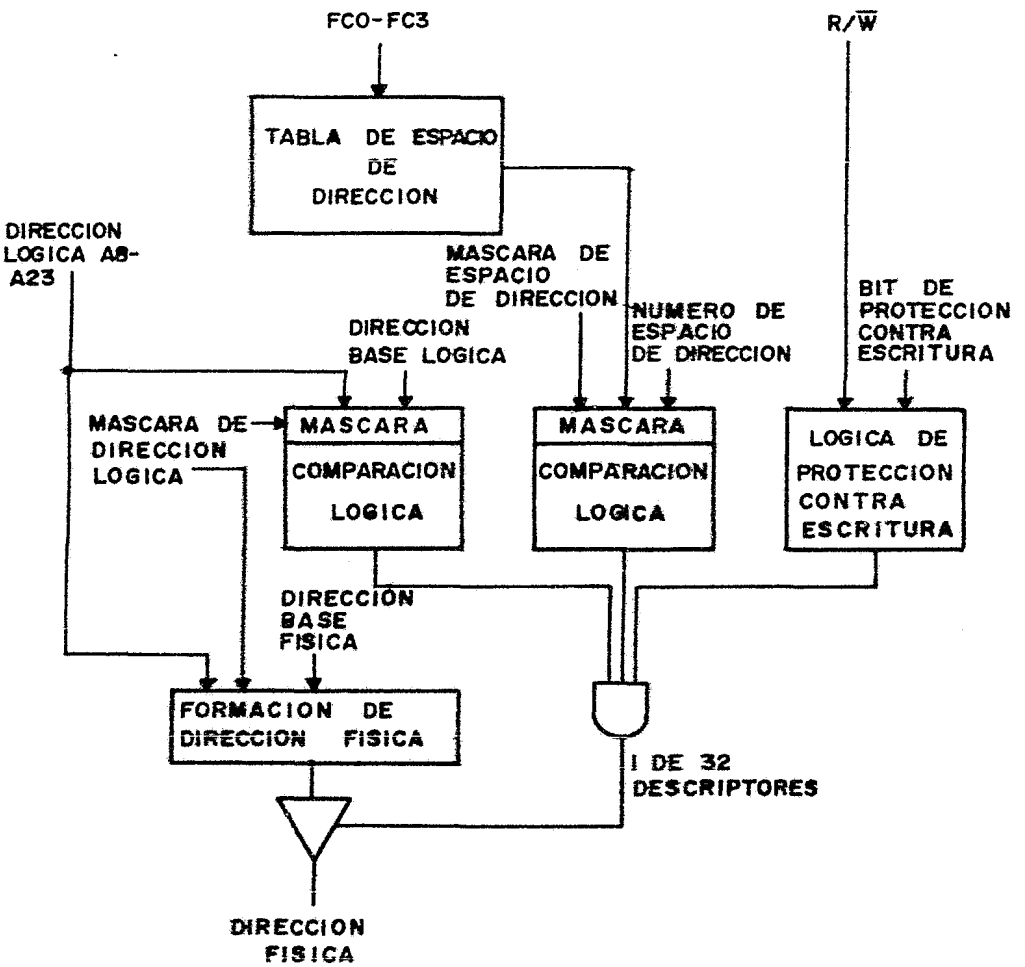


FIG 70 DIAGRAMA FUNCIONAL

Máscara de Dirección Lógica (16 Bits).- Este Registro contiene una máscara la cual indica los Bits de la Dirección Base Lógica que serán usados en la comparación de direcciones.

Dirección Base Física (16 Bits).- Este Registro es usado junto con la Máscara de Dirección Base Lógica y la Dirección Lógica proporcionada por el procesador para generar la Dirección Física.

Número de Espacio de Dirección (8 Bits).- Este Registro contiene un número el cual es usado junto con la Máscara de Espacio de Dirección para detectar igualaciones con los Espacios de Dirección para cada ciclo proporcionados por el procesador.

Máscara de Espacio de Dirección (8 Bits).- Este Registro contiene una máscara la cual define las posiciones de Bits del Número de Espacio de Dirección que serán usadas en las comparaciones con el Espacio de Dirección de Ciclo proporcionado por el procesador.

Status del Segmento (8 Bits).- Este Registro contiene Bits que representan status o atributos del segmento, por ejemplo, si el segmento ha sido accesado, si ha sido modificado, si esta protegido contra escritura,etc.

REGISTROS DE SISTEMA.-Este grupo esta compuesto por los siguientes 8 Registros:

TABLA DE ESPACIO DE DIRECCIONES.- Cada MMU tiene una copia local de esta tabla. La tabla está organizada en 16 registros de lectura y escritura de ocho bits cada uno. Cada elemento es programado por el sistema operativo con un solo número de espacio de dirección, cada uno de los cuales es asociado con una tarea. Durante un acceso a memoria el MMU recibe 4 bits de código de función los cuales son usados como un índice dentro de esta tabla para seleccionar el Número de Espacio de Dirección del Ciclo. Este número es usado para hacer una comparación con el Número de Espacio de Dirección en cada uno de los 32 Registros de Descripción.

ACUMULADOR.- Este Registro está formado por nueve Registros de 8 Bits cada uno y es usado para proporcionar acceso a los Registros de Descripción de Segmento, realizar la traslación de direcciones y para almacenar información durante un error.

REGISTRO DE STATUS GLOBAL (8 Bits).- Este Registro es usado para informar de errores y para habilitar interrupciones de un MMU. Todos los MMU del sistema almacenan la misma información en estos registros.

REGISTRO DE STATUS LOCAL (8 Bits).- Este registro almacena información local del MMU.

APUNTADOR DE REGISTROS DE DESCRIPCION (8 Bits).- Los cinco bits menos significativos de este Registro son usados como un apuntador durante las operaciones de carga del Registro de Descripción de Segmento y escritura o lectura de Status del Segmento.

APUNTADOR DE REGISTRO DE DESCRIPCION INVOLUCRADO (8 Bits).- Este registro sólo para lectura identifica al Registro de Descripción involucrado en uno de los siguientes eventos: Violación de Escritura, una falla en la carga de un Registro de Descripción o una operación de Traslación Directa.

APUNTADOR DE INTERRUPCION DE REGISTRO DE DESCRIPCION (8 Bits).- Este registro sólo para lectura es usado para indicar cual Registro de descripción causó una interrupción.

VECTOR DE INTERRUPCION (8 Bits).- Este registro contiene el Vector de Interrupción del MMU.

PROCEDIMIENTO DE TRASLACION DE DIRECCIONES.- Al principio de un ciclo de canal el procesador proporciona la dirección lógica, la señal de lectura o escritura y el Código de Función el cual es usado como un apuntador en la Tabla de Espacios de Dirección para seleccionar el Número de Ciclo de Espacio de Dirección. Cuando el procesador indica que hay una dirección válida el proceso de traslación es iniciado enviando el Número de Ciclo de Espacio de Dirección, la Dirección Lógica y el status de la señal Lectura/Escritura a cada Registro de Descripción para ser comparados. Una igualación puede ocurrir en dos áreas que son: Rango y Espacio. Una igualación de Rango ocurre si cada posición de Bit de la Dirección Lógica proporcionada es igual a la Dirección Base Lógica en las posiciones indicadas por la Máscara de Dirección Lógica. Una igualación de Espacio ocurre si cada posición de Bit del Número de Ciclo de Espacio de Dirección es igual al Número de Espacio de Dirección en las posiciones indicadas por la Máscara de Espacio de Dirección.

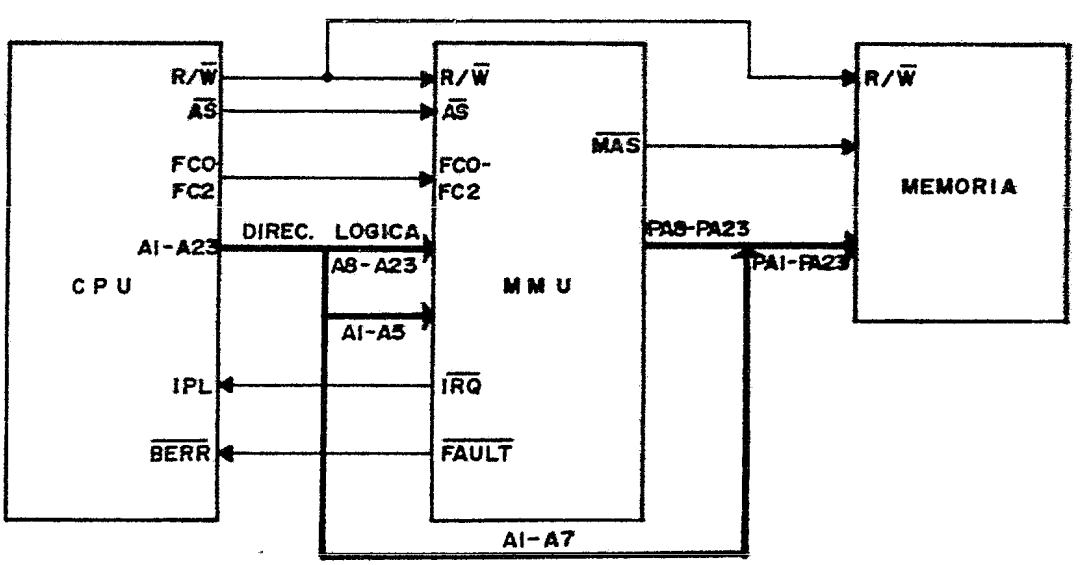


FIG 71 MECANISMO DE MANEJO DE MEMORIA

4.3.2 CONTROLADOR INTELIGENTE DE PERIFERICOS (INTELLIGENT PERIPHERAL CONTROLER IPC) PARA EL PROCESADOR 68000

El 68120 es un Controlador Inteligente de Periféricos que proporciona la interfase entre la familia 68000 y los dispositivos periféricos a través del Canal de Sistema y líneas de control. Puede ser configurado para funcionar de ocho diferentes maneras. Tiene una Memoria RAM de Doble Puerto, 6 Registros Semáforo, cuatro puertos configurables de diferentes maneras, un Reloj Programable y una Interfase para Comunicación Serial.

MEMORIA RAM DE DOBLE PUERTO Y REGISTROS SEMAFORO.- La memoria RAM de Doble Puerto consiste de 128 Bytes la cual puede ser manejada por el Canal de Sistema a través de ocho líneas de dirección, ocho líneas de datos y tres líneas de control para accesos síncronos y asíncronos. Los Registros Semáforo permiten la administración de recursos compartidos, los cuales pueden ser la Memoria RAM de Doble Puerto o dispositivos periféricos. Estos Registros se utilizan para evitar accesos simultáneos de escritura a la Memoria RAM de Doble Puerto. Cada Registro consiste únicamente de un bit Semáforo y un bit de Dominio. Los bits Semáforo son probados y puestos durante accesos simultáneos. En una operación en que el Controlador y el Procesador de Sistema intentan leer o escribir el mismo Registro Semáforo, la circuitería externa deberá decidir cual lee y borra el bit Semáforo y cual lee y prende el bit Semáforo. El bit de Dominio es de solo lectura, si el Bit Semáforo está prendido entonces el Bit Dominio indica cual procesador puso el bit Semáforo y si el Bit Semáforo está Borrado entonces el Bit Dominio indicará cual fue el último procesador que puso el Bit Semáforo.

DESCRIPCION DE LOS PUERTOS.-El Controlador Inteligente de Periféricos cuenta con cuatro Puertos que se describen a continuación:

Puerto 1 (SD0-SD7).- Cuenta con ocho líneas de datos bidireccionales que permiten la transferencia de datos entre la Memoria RAM de Doble Puerto o los Registros Semáforo y el Canal de Sistema.

Puerto 2 (P20-P24).-Es un puerto paralelo de 5 bits de entrada salida donde cada línea es configurada por el Registro de Dirección de Datos.

Entradas en P20,21,22 determinan el Modo de operación el cual es guardado dentro del Registro de Control de Programa. Puede ser usado como interfase para comunicación serial o como Reloj si se configura como salida.

Puerto 3 (P30-P37).-Cuenta con ocho líneas de datos bidireccionales y dos líneas de control. Puede ser configurado como Puerto de 8 bits bidireccional de entrada salida o como Puerto con Canal de Datos y Direcciones Multiplexado, dependiendo del Modo de operación en que se encuentre.

Puerto 4 (P40-P47).- Cuenta con ocho líneas, puede ser configurado como Puerto de entrada salida de 8 bits, como direcciones de salida y datos de entrada dependiendo del Modo de operación en que sea configurado.

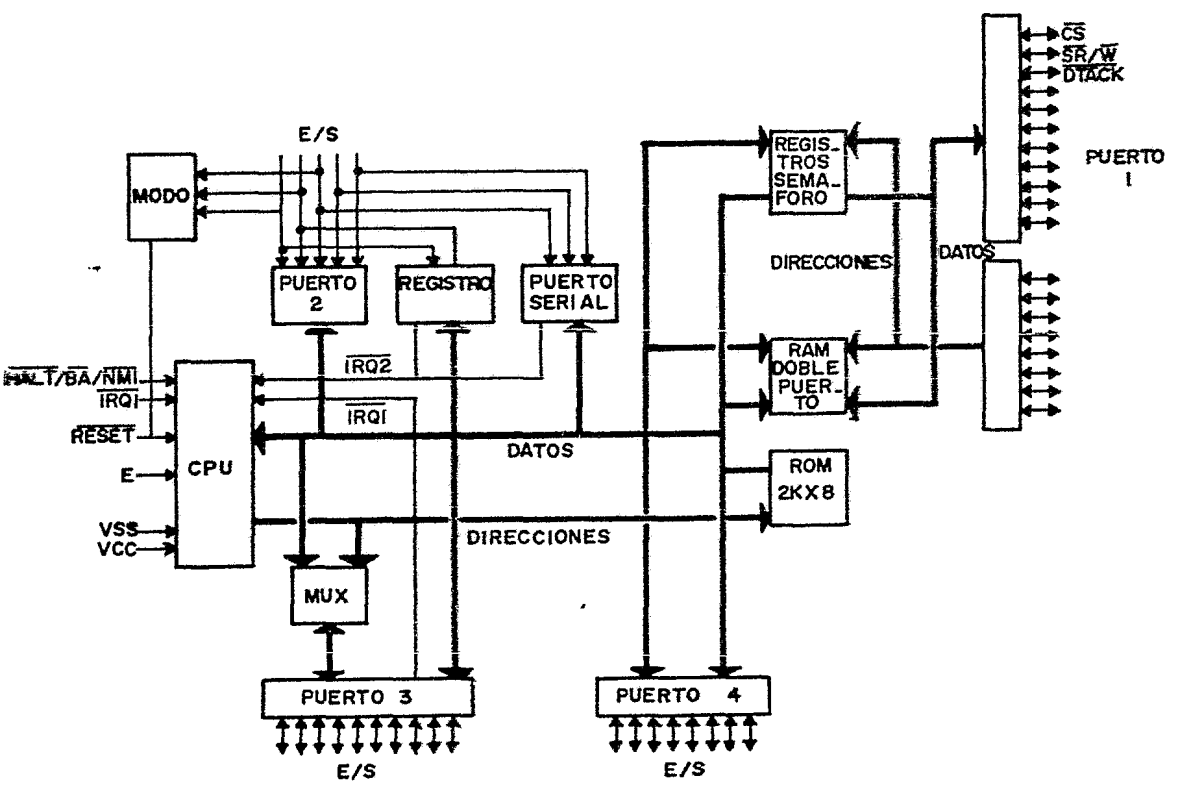


FIG 72 ESTRUCTURA INTERNA

RELOJ PROGRAMABLE.- Este reloj puede ser usado para realizar mediciones en señales de Entrada y genera una señal de Salida independiente. El reloj es programado mediante un Registro de Control y Estado de 8 Bits.

INTERFASE DE COMUNICACION SERIAL.- Esta interfase asíncrona bidireccional proporciona dos formatos para los datos y velocidades de transmisión múltiples. El Transmisor y el Receptor son independientes, utilizan el mismo formato de datos y la misma velocidad. La interfase cuenta con las siguientes características programables: Formato de Datos Estándar o Bifase, Fuente de reloj Externa o Interna, Velocidad, Habilitación de Interrupciones de Transmisor y Receptor independientes.

MODOS DE OPERACION DEL CONTROLADOR.- Los Modos de Operación controlan el mapeo de memoria, la configuración de los Puertos y la localización de los vectores de interrupción.

Los ocho Modos de operación estan agrupados en 3 grupos principales: Modo de Un Solo Circuito (Modos 4 y 7), Modo Extendido No-Multiplexado (Modo 5) y Modo Extendido Multiplexado (Modos 0,1,2,3,6).

Modo de Un Solo Circuito.-En este Modo tres de los cuatro Puertos son configurados como paralelos de entrada salida de datos. El controlador funciona como una Microcomputadora, se tienen como máximo 21 líneas de E/S y 2 líneas de control del Puerto 3.

Modo Extendido No-Multiplexado.-En este Modo de operación se cuenta con un pequeño espacio de manejo de memoria y con las características más importantes del Modo de Un Solo Circuito. El puerto 3 funciona como un Canal bidireccional de Datos y el Puerto 4 queda configurado como de entrada de datos.

Modo Extendido Multiplexado. En este Modo el Controlador tiene la capacidad para direccionar hasta 64 kbytes. El Puerto 3 funciona como Canal de Datos/Direcciones multiplexados. En los Modos del 0 al 3, el Puerto 4 proporciona las líneas de dirección A8-A15.

CAPITULO V

5.1 CONCLUSIONES SOBRE ARQUITECTURAS

Al finalizar el estudio y comparación de las arquitecturas interna y externa de cada procesador encontramos que cada uno de ellos tiene ventajas y desventajas, algunas muy marcadas en diferentes aspectos. Encontramos que la arquitectura del Z8000 está completamente basada en un diseño de lógica aleatoria, esto tiene la ventaja de que lo hace muy rápido y la desventaja de que su diseño fué muy complicado, tanto que las primeras versiones de este procesador tenían problemas al ejecutar ciertas instrucciones o ciertas secuencias de instrucciones (15). El diseño del 68000 se encuentra en el extremo opuesto ya que está completamente basado en una arquitectura del tipo microprogramado, este diseño tiene la ventaja de que es fácil de diseñar y modificar cambiando únicamente el contenido del microprograma, la desventaja de este diseño es que es un poco más lento que el basado en lógica aleatoria (11). El 8086 es una mezcla de las dos filosofías de diseño mencionadas anteriormente, su unidad de ejecución está basada en un diseño microprogramado y la unidad de Interfase al Canal está basada en un diseño de lógica aleatoria, esto permite un balance entre facilidad de diseño y al mismo tiempo rapidez.

El mecanismo de prebúsqueda de instrucciones más sofisticado es el implementado en el 8086. Ya que permite que las dos unidades funcionen de manera completamente independiente. Esta característica lo hace más eficiente que los otros dos procesadores en la búsqueda de instrucciones.

Los procesadores Z8000 y 68000 cuentan con dos modos de operación internos que permiten una mejor organización de sistemas operativos, mayor protección para éstos y para los demás usuarios del sistema, todo esto es posible a través del manejo de instrucciones privilegiadas las cuales no pueden ser manejadas en el modo de menor prioridad, en general este modo es utilizado por los usuarios del sistema. Cabe hacer la aclaración de que el 8086 también cuenta con dos modos de operación aunque estos se refieren únicamente a la manera en que serán configuradas las líneas de control y status del procesador. Estos modos no afectan de ninguna manera las características internas y no tienen ninguna relación con los modos de operación de los otros dos procesadores. Estos modos no pueden ser cambiados ya que dependen de la conexión de una de las líneas del procesador y son completamente transparentes para el usuario desde el punto de vista de programación.

Se puede decir que el 68000 y Z8000 tienen un diseño que se adapta más fácilmente a las necesidades de los sistemas modernos. El 8086 queda en un plano intermedio ya que toda la protección de sistemas operativos deberá ser hecha a través de programación.

En el manejo de canales, el 68000 es el único que maneja canales de datos y direcciones separados, esto permite que la implementación de los diseños sea más sencilla ya que se requieren menos componentes externos, además, esta característica hace que sea más rápido en los accesos a memoria (34). El 8086 y Z8000 tienen canales multiplexados para datos y direcciones, esto tiene la desventaja de que requiere de componentes externos para el almacenamiento de direcciones, además hace que el procesador sea más lento; la única ventaja es que el número de líneas del circuito integrado se reduce considerablemente.

En capacidad de direccionamiento de memoria, el 68000 es el que posee las mejores características, el Z8000 le sigue en capacidad y el 8086 es el más pobre en este aspecto. Consideramos que la diferencia entre 68000 y Z8000 no es determinante para los propósitos de este estudio ya que las cantidades de memoria que pueden direccionar son muy grandes en ambos casos. El 8086 está algo limitado en este aspecto, aunque consideramos que esto es debido al momento en el que fué diseñado y lanzado al mercado, ya que en esa época los costos de la memoria eran mucho mayores debido al estado de la tecnología de integración de circuitos, y el tener 1 Mbyte de memoria era algo fuera de de lo común en un microprocesador.

En lo que se refiere a tamaño y capacidad de los registros internos, el 68000 tiene un lugar preponderante ya que cuenta con los registros más grandes aunque tiene la deventaja de que sus registros no pueden ser agrupados, además, sólo ocho registros pueden ser usados para manejo de datos y ocho para direcciones. El Z8000 tiene la ventaja de que sus registros pueden ser agrupados hasta en cuartetas por lo que puede manejar fácilmente datos de hasta 64 bits, además, todos sus registros son de propósito general. El 8086 tiene la desventaja de que sus registros son de propósito especial en la mayoría de los casos y la única dificultad que se presentó para la implementación de los algoritmos fué que sus registros no pueden ser agrupados.

Por lo expuesto anteriormente, podemos observar que el Z8000 cuenta con la estructura de registros más versátil, lo cual se presta para la implementación de algoritmos aritméticos que utilizan datos de precisión múltiple. En capacidad, el 68000 es el mejor, en estos dos aspectos el 8086 queda bastante atrás con respecto a los otros dos procesadores aunque tiene la ventaja de poder utilizar

registros implícitos en la mayoría de las instrucciones.

En el manejo de pilas el Z8000 y 68000 son equivalentes ya que cuentan con dos apuntadores de pila implícitos independientes para sus dos modos de operación, además de poder manejar pilas implementadas por el usuario. El 8086 sólo cuenta con el apuntador de pila tradicional y puede utilizar sólo un registro extra como apuntador adicional.

Los registros de control de programa y status tienen funciones equivalentes en los tres procesadores, con la diferencia de que el contador de programa del Z8000 es manejado de manera segmentada, además este procesador es el único que cuenta con un contador de refrescamiento de memoria el cual es muy útil en el diseño de sistemas con memorias RAM dinámicas y cuenta también con un registro apuntador de nuevo Status de Programa el cual permite que el manejo de las rutinas de interrupción sea más versátil.

En manejo de interrupciones, el 68000 soporta una estructura más sofisticada que el Z8000 y el 8086, ya que cuenta con siete niveles de interrupción con 192 vectores posibles, el Z8000 tiene sólo tres tipos de interrupciones y el 8086 sólo tiene 2 tipos. La ventaja del 68000 sobre los otros dos es que las prioridades no tienen que ser implementadas externamente a menos que se requiera de más de un dispositivo por nivel de prioridad, esto permite que el manejo de interrupciones sea más eficiente. El Z8000 y el 8086 están en un plano inferior ya que los dispositivos que interrumpen deberán de ser encadenados para poder manejar diferentes niveles de prioridad. El 8086 tiene la opción de trabajar con un controlador de interrupciones el cual se encarga de proporcionar hasta 64 niveles de interrupción.

En manejo de trampas internas, el 68000 es el mejor, ya que tiene la capacidad de generar trampas para instrucciones no implementadas lo cual permite una emulación sencilla de estas mediante programas lo que es muy útil para el programador. Además cuenta con trampas de error de dirección, error de canal, modo de trazo, instrucción ilegal, instrucción privilegiada, división entre cero y sobrepaso. Esto permite una gran confiabilidad ya que es muy difícil que el procesador no se recupere de un error de programación o de un error causado por la falla de un dispositivo periférico.

Le sigue en complejidad el Z8000 el cual cuenta con trampas para instrucciones no implementadas, llamadas de sistema (la cual no existe en el 68000), instrucción privilegiada y error de segmentación, estas trampas no permiten una recuperación tan fácil como en el caso del 68000 debido a que no son tan específicas. Este procesador

tiene la desventaja de que carece de la trampa de ejecución paso a paso la cual es muy útil en la depuración de programas.

El menos sofisticado en manejo de trampas es el 8086 ya que sólo maneja trampas de división entre cero, sobrepaso, ejecución paso a paso e interrupción por programa. El 8086 no cuenta con ningún mecanismo de emulación de instrucciones no implementadas por lo que estos procesos deberán llevarse a cabo mediante llamadas de subrutina lo cual es poco ventajoso desde el punto de vista del programador.

En lo que respecta al manejo de Entrada/Salida el Z8000 es el más versátil ya que puede manejar transferencias desde cualquiera de sus registros hacia y desde los puertos, los cuales pueden ser mapeados como memoria o ser manejados independientemente como tales. El 8086 maneja solo transferencias desde y hacia el acumulador con puertos direccionados de la misma manera que el Z8000. El 68000 tiene la característica de que sus puertos de entrada salida están mapeados como memoria, esto es ventajoso desde el punto de vista de la protección de E/S ya que ésta se efectúa de la misma manera que la de memoria.

En lo que se refiere a soporte para sistemas de multiproceso, el único procesador que cuenta con recursos específicos es el Z8000, estos recursos son proporcionados tanto a nivel de circuitería como de programación. La ausencia de estos recursos en los otros dos procesadores hace que el diseño de sistemas de multiproceso sea más complicado. Esto permite al Z8000 cumplir con las necesidades de las tendencias actuales las cuales están siendo enfocadas principalmente hacia los sistemas de multiproceso.

5.2 CONCLUSIONES SOBRE EL CONJUNTO DE INSTRUCCIONES

Los modos de direccionamiento de los tres procesadores son similares aunque existen algunos modos particulares en cada uno. El 68000 tiene la gran ventaja de que sus modos de direccionamiento son muy generales y aplicables en la mayoría de las instrucciones, esto representa una buena ayuda para el programador ya que no es necesario estar consultando continuamente el manual para ver si una instrucción acepta un modo de direccionamiento o no, además cuenta con modos de autoincremento y autodecremento en todas las instrucciones, estos modos están disponibles en los otros dos procesadores sólo en instrucciones particulares y hubiera sido deseable que se manejaran de manera más general. Esto fué notado al desarrollar los programas de aplicación, ya que en el caso de 8086 y Z8000 es necesario incrementar o decrementar los apuntadores de memoria mediante instrucciones adicionales.

El tamaño y tipo de los datos que se pueden manejar en los procesadores fué el adecuado en los programas de búsqueda y reacomodo, pero en el caso de los programas de operaciones aritméticas de 32 bits se presentó en el 8086 la dificultad de que no puede agrupar sus registros para manejar datos mayores de 16 bits, esto causó que los programas de suma y resta se complicaran lo que quedó reflejado principalmente en el tamaño de los programas.

En las instrucciones de movimiento de datos, el 68000 y el Z8000 tienen instrucciones bastante poderosas ya que pueden cargar de manera múltiple sus registros, el 8086 no cuenta con este tipo de instrucciones lo que se notó en la cantidad de instrucciones empleadas principalmente en la carga de los parámetros de los programas.

En lo que respecta a manejo de datos numéricos, el Z8000 es el más poderoso ya que puede manejar directamente operandos de 32 y 64 bits en multiplicación y división respectivamente, por lo que se facilitó la elaboración de estos programas. En el caso del 68000 se tuvo el problema de que en la multiplicación sólo se pueden efectuar operaciones en 16 bits y se tuvo gran dificultad en el manejo de datos de mayor tamaño así como también en el manejo de los productos parciales ya que no se pueden manipular independientemente las partes baja y alta de sus registros y hubo necesidad de efectuar intercambios entre éstas.

En el 8086, todas las instrucciones aritméticas, con excepción de la división manejan datos de 16 bits por lo que en los programas de suma y resta fué necesario utilizar mayor número de instrucciones ya que las comparaciones de magnitudes fueron efectuadas por partes. En la multiplicación, debido a la limitación en el número de registros, fué necesario que la mayoría de los datos y productos parciales estuvieran contenidos en memoria.

El 8086 es el único que cuenta con instrucciones para manejo de caracteres ASCII lo cual lo convierte en el más indicado para el manejo de este tipo de datos.

Las instrucciones de comparación utilizadas en los programas de aplicación son similares en los tres procesadores, no se tuvo ninguna dificultad en cuanto a tamaño de los datos o modos de direccionamiento.

Las instrucciones lógicas fueron utilizadas solamente en los programas del 8086, debido a que este procesador no cuenta con instrucciones específicas para manejo de bits individuales. Las instrucciones lógicas fueron utilizadas principalmente para borrar y prender bits. Consideramos que en este tipo de instrucciones el 68000 es superior ya que es el único que puede manejar estas operaciones en 32 bits. En instrucciones de rotación y corrimiento el Z8000 es inferior a los otros dos ya que no puede efectuar estas operaciones en memoria lo cual representa una gran limitación.

En instrucciones de salto condicional, el único que se encuentra algo limitado es el 68000 ya que todas estas instrucciones manejan sólo desplazamientos de hasta 256 bytes lo cual puede tener sus inconvenientes en rutinas muy largas. En los programas que se elaboraron no se encontraron dificultades ya que todos los saltos utilizados se encontraban dentro del rango mencionado anteriormente.

En las instrucciones de retorno de subrutina el Z8000 es superior al 68000 y 8086 ya que es el único que maneja retornos condicionales, esto tiene como ventaja el hacer más eficiente la subrutina ya que es más rápida en su ejecución y es menor el espacio de memoria que ocupa.

Las instrucciones para manejo de bloques y series de bytes están disponibles sólo en el Z8000 y el 8086, estas instrucciones son muy útiles en programas que comparan cadenas de datos en memoria ya que es posible efectuar esta comparación con una sola instrucción lo cual representa un ahorro en espacio de código y en tiempo de ejecución. Estas instrucciones son las únicas que manejan modos de direccionamiento con incremento y decremento automático. Además, estos dos procesadores cuentan también con instrucciones de traducción las cuales permiten cambiar de manera eficiente y rápida una tabla de un código a otro.

En instrucciones de control del procesador, los tres manejan instrucciones similares que pueden alterar el contenido del registro de banderas y control, la única diferencia es que en el 68000 no existe ninguna instrucción que permita un cambio controlado de modo usuario a modo sistema.

5.3 CONCLUSIONES SOBRE FAMILIAS DE SOPORTE

Las familias de soporte externo para los tres procesadores son muy poderosas en cuanto al desahogo de tareas del procesador, aunque existe una marcada diferencia entre ellas debido a las diferentes filosofías de diseño. En el 8086, la familia de soporte es la más extensa y la más popular, aunque casi todos los periféricos existentes para los microprocesadores de 8 bits (familia 8080) son compatibles con el 8086 por lo cual toda la programación desarrollada para estas familias puede ser implementada en este procesador con muy pocos cambios y en algunos casos con ninguno.

Además de los periféricos de los procesadores de 8 bits, se desarrollaron algunos dispositivos específicos para el 8086, los dos más importantes son el 8087 procesador de datos numéricos y el 8089 procesador de entrada/salida, éstos proporcionan un gran desahogo de tareas para el procesador.

Es de hacer notar que hasta el momento no existe ningún dispositivo que proporcione características de manejo y protección de memoria para este procesador, lo que representa una carencia bastante importante en este aspecto ya que todas estas tareas deberán ser efectuadas a nivel de programación. Además de estos dos periféricos, existen dispositivos que efectúan control de interrupciones, de los canales, de comunicaciones seriales y paralelas, etc.

En las familias de soporte para 68000 y Z8000 existen dispositivos similares para el manejo de memoria, para tranferencias de acceso directo y controladores inteligentes de periféricos. El Z8000 tiene además las unidades de arquitectura de procesamiento extendido para datos numéricos, procesamiento de errores y para codificación de datos. En las Unidades de Manejo de Memoria, la más sofisticada y a la vez fácil de usar es la del Z8000, esta unidad proporciona mayores características de protección y capacidad de manejo de segmentos y es bastante fácil de programar. Las unidades para manejo de transferencias directas a memoria son similares, la del 68000 puede manejar transferencias más rápidas aunque su estructura es un poco más sencilla. Los controladores inteligentes de periféricos

son muy parecidos, ambos están basados en arquitecturas de 8 bits diseñadas anteriormente y tienen características de funcionamiento y capacidad bastante parecidas. Estas familias fueron diseñadas específicamente para estos procesadores y no son compatibles con los diseños anteriores. El 68000 tiene también la opción de poder controlar dispositivos de la familia 6800 sin necesidad de circuitería adicional. El Z8000 no tiene esta característica por lo que es necesario agregar circuitos adicionales para poder manejar dispositivos de la familia Z80. Aparte de los periféricos mencionados existen también dispositivos para soportar comunicaciones seriales y paralelas, control de discos flexibles, de terminales, detectores y correctores de errores, etc.

La familia mejor soportada por el fabricante para el desarrollo de circuitería y programación es la del 8086, ya que es compatible con los programas ya existentes para la familia 8080 y el fabricante proporciona gran cantidad de ensambladores, compiladores para diferentes lenguajes de alto nivel, etc. Además de esto existen varios tipos de equipos para el desarrollo de sistemas basados en este procesador. Los fabricantes del 68000 y del Z8000 ofrecen equipos y paquetes de programación similares para el desarrollo de equipos basados en estos procesadores, aunque no en igual forma que para el 8086.

Esperamos que lo expuesto anteriormente proporcione una buena ayuda en la selección de cualquiera de estos microprocesadores. Queremos hacer incapié en que lo mencionado es un juicio basado únicamente en las características de cada procesador y de su familia de soporte, para hacer una buena selección es necesario situarse en la aplicación para la cual se vaya a destinar el procesador, si es necesario que sea compatible con un diseño anterior y las necesidades de expansión futura.

Durante el desarrollo de este estudio, notamos que es posible que esta generación de microprocesadores de 16 bits sea de poca permanencia en el mercado debido a que la tecnología esta avanzando a pasos agigantados en cuestión de escala de integración. Para justificar la suposición anterior, nos basamos en que aunque no se ha llegado al límite de aprovechamiento de los procesadores de 16 Bits ya empiezan a ser introducidos en el mercado procesadores de 32 bits que son más poderosos y eficientes que los de la generación anterior. Actualmente, los procesadres de 32 bits más populares son: el iAPX432 de Intel, el 68020 de Motorola y el ya muy próximo Z80000 de Zilog.

La información disponible acerca de estos procesadores es reducida, pero las características mencionadas en la publicidad indican que son de mucha mayor capacidad que los de 16 bits, esto provoca la inquietud de efectuar un estudio sobre estos nuevos dispositivos aunque esto no queda comprendido dentro de los propósitos de este trabajo.

Esperamos que este estudio sea de utilidad para aquellas personas que alguna vez tengan la necesidad de seleccionar algún sistema de cómputo, asimismo, consideramos también que este trabajo puede ser de utilidad para aquellas personas que deseen iniciarse en el estudio de los microprocesadores de 16 bits. Para las personas que ya estén familiarizadas con el funcionamiento de estos dispositivos, este trabajo puede ser utilizado a manera de referencia.

APENDICE A. MICROPROCESADOR 8086.

DESCRIPCION DE SEÑALES

AD0-AD15 CANAL DE DIRECCIONES/DATOS Bidireccional, de 3 Estados, estas líneas multiplexadas de Datos/Direcciones son utilizadas para el manejo de memoria y E/S así como para la transferencia de datos desde y hacia el procesador.

PARTE ALTA DEL CANAL DE DIRECCIONES/STATUS (A16/S3-A19/S6).- Salidas, Tres Estados. Proporcionan los cuatro Bits más significativos de la dirección física de 20 Bits así como también información acerca del tipo de operación que se está efectuando, S3 y S4 indican el Registro de Segmento que se está utilizando para las referencias a memoria. El bit S5 indica el estado de la Bandera de Habilitación de Interrupción, y el S6 indica que el procesador está utilizando los canales del sistema. S7 está actualmente sin uso.

INTERRUPCION NO MASCARABLE (NMI).- Entrada, una transición de nivel bajo a nivel alto causa que el procesador entre en un ciclo de atención de interrupción al final de la ejecución de la instrucción que se está procesando.

INTERRUPCION (INTR).- Entrada, es muestreada en el último ciclo de reloj de cada instrucción y si es encontrada activa (nivel alto) el procesador entra en un ciclo de reconocimiento de interrupción.

RELOJ (CLK).- Esta línea es una entrada para reloj con nivel TTL, asimétrico con ciclo de trabajo de 33%.

HABILITACION DE BYTE ALTO, STATUS 7 (BHE*/S7).- Salida multiplexada indica que se va a efectuar una transferencia de datos sobre la parte más significativa del Canal de Datos. La señal de status S7 está siempre en nivel bajo.

MODO MINIMO/MAXIMO* (MN/MX).- Línea de entrada que se utiliza para definir si el procesador está configurado en modo mínimo o en modo máximo.

LECTURA (RD*).- Línea de salida, indica que el procesador está ejecutando una lectura de memoria o de E/S.

SOLICITUD DE CANAL/OTORGAMIENTO (RQ*/GTO*), RETENCION (HOLD).- Líneas bidireccionales, son utilizadas para indicar al procesador que otro dipositivo está solicitando los Canales. Además, el procesador indica que ha cedido el control de los Canales a través de estas líneas. En modo mínimo, esta línea de entrada sirve para solicitar al procesador el control de los Canales (HOLD). Esta línea tiene prioridad sobre RQ*/GT1*.

SOLICITUD DE CANAL/OTORGAMIENTO (RQ*/GT1*), RECONOCIMIENTO DE RETENCION (HLDA).- Línea bidireccional, En modo máximo funciona igual que RQ*/GTO*. En modo mínimo, esta línea funciona como salida para indicar que se ha concedido el control de los Canales.

LOCK*, ESCRITURA/LECTURA (WR*).- Línea de salida, en modo máximo indica que otros dispositivos no pueden tener control de los canales. Esta señal se activa mediante el prefijo LOCK el cual antecede a una instrucción. En modo mínimo esta línea indica que el procesador está efectuando una escritura.

STATUS 2 (S2*), MEMORIA/ E/S* (M/IO*).- Línea de salida, en modo máximo se utiliza junto con S1 y S0 para indicar el tipo de operación que está efectuando el procesador. En modo mínimo, indica si el procesador esta efectuando una referencia a Memoria o a E/S.

STATUS 1 (S1*), TRANSMISION/RECEPCION DE DATOS (DT/R*).- Línea de salida, en modo máximo funciona de la misma manera que S2. En modo mínimo, indica la dirección del flujo de datos que tiene el Canal.

STATUS 0 (S0*), HABILITACION DE DATOS (DEN*).- Línea de salida, en modo mínimo indica que hay un Dato válido sobre el Canal.

ESTADO DE LA COLA DE INSTRUCCIONES 0 (QS0), HABILITACION DE DIRECCIONES (ALE).- Línea de salida, en modo máximo es usada junto con QS1 para indicar el estado de la cola de instrucciones interna.

ESTADO DE LA COLA DE INSTRUCCIONES 1 (QS1), RECONOCIMIENTO DE INTERRUPCION (INTA*).- Línea de salida, en modo máximo funciona igual que QS0. En modo mínimo, indica que hay una dirección válida sobre el Canal.

PRUEBA (TEST*).- Línea de entrada, es examinada por la instrucción "Espera por Prueba", si es encontrada en nivel bajo la ejecución continúa, de otra manera el procesador espera en un estado de reposo.

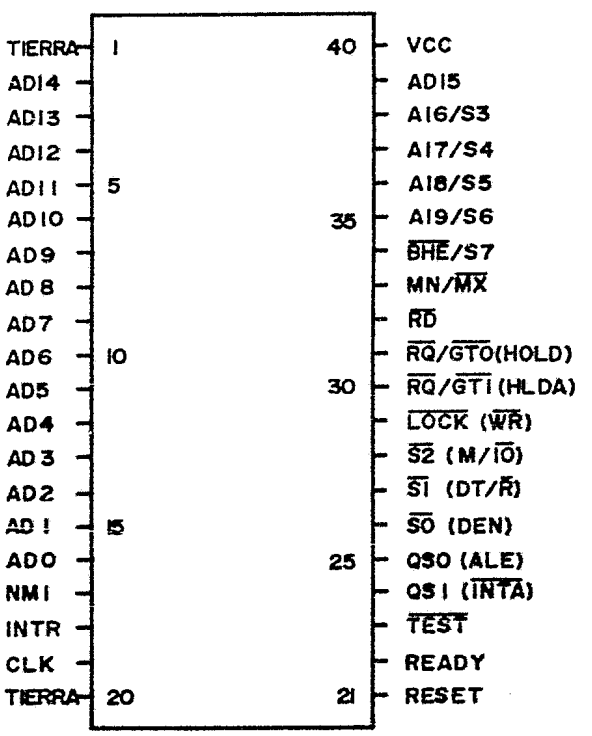


FIG 73 CONFIGURACION DE LINEAS

READY.- Línea de entrada, funciona como un reconocimiento proporcionado por la memoria o dispositivos de E/S para indicar que se completará la transferencia de datos.

RESTAURACION (RESET).- Línea de entrada, es utilizada para restaurar al procesador.

CONJUNTO DE INSTRUCCIONES

El conjunto de instrucciones tiene 95 instrucciones básicas que se encuentran divididas en los siguientes grupos .

* Movimiento de datos
* Aritméticas
* Lógicas y Manipulación de Bits
* Manejo de Cadenas
* Control de Programa
* Control de Procesador

TIPOS DE DATOS:

1 BYTE.
2 PALABRA.
3 BYTE O PALABRA.
4 PALABRA LARGA.

MODOS DE DIRECCIONAMIENTO:

1 INMEDIATO.
2 DIRECTO.
3 DIRECTO INDEXADO.
4 IMPLICITO.
5 BASE.
6 BASE RELATIVO.
7 RELATIVO.
8 REGISTRO.
9 INDIRECTO.

MNEMONICO	INSTRUCCION	T. DATOS	MODOS
AAA	Ajusta Resultado de una operación Suma en ASCII.	1	8
AAD	Ajusta el Reg. AX para una División en BCD.	1	8
AAM	Ajusta el Reg. AX para una Multiplicación en BCD	1	8
AAS	Ajusta Resultado de una operación Resta en ASCII.	1	8
ADC	Suma con Acarreo.	3	1-8
ADD	Suma sin Acarreo.	3	1-8
AND	Operación Lógica " Y ".	3	1-8
CALL	Llamada a Subrutina	3	1-8
CBW	Extensión de Signo.	1	8
CLC	Limpia Acarreo.	-	2
CLD	Limpia Bandera de Dirección.	-	2
CLI	Limpia Bandera de Interrupción.	—	2
CMC	Complementa la Bandera de Acarreo.	-	2
CMP	Compara	3	1-8
CMPS	Compara Memoria con Memria.	3	3
CWD	Extensión de Signo.	2	2
DAA	Ajuste a Decimal después de Sumar.	1	8
DAS	Ajuste a Decimal después de Restar.	1	8
DEC	Decrementa	3	2-5
DIV	División	3-4	2-5
ESC	Accesa Localidad de Memoria.	3	4
HLT	Para al Procesador.	-	2
IDIV	División Signada.	3-4	2-5
IMUL	Multiplicación Signada.	3	2-5
IN	Lee Puerto.	3	1,4
INC	Incrementa.	3	2-5
INT	Interrupción.	-	2
INTO	Interrupción por Sobreflujo.	-	2
IRET	Retorno de Interrupción.	-	2

Jcc *	Salta en Código de Condición.	2,-	1,4-7
LAHF	Carga las Banderas de 8080 en AH.	1	2
LDS	Carga Registro y DS desde Memoria.	3	2
LEA	Carga Registro con desplazamiento de dirección.	3	2
LES	Carga Registro y ES desde memoria.	3	2
LOCK	Habilita la señal Lock	-	2
LODS	Carga desde Memoria al Acumulador.	3	4
LOOPcc	Decrementa el Registro CX y salta en Código de Condición.	3	7
MOV	Mueve del Opernando Fuente al Destino. (incluye los Registros de Segmento).	3	1-8
MOVS	Mueve de Memoria a Memoria.	3	9
MUL	Multiplicación	3	2,8
NEG	Niega el contenido del Operando.(esto es Complemento a 2's).	3	2,8
NOP	No Operación.	-	1
NOT	Niega el contenido del Operando.(esto es Complemento a 1's).	3	2,8
OR	Operación Lógica " O "	3	2-8
OUT	Escribe a un Puerto	3	2,4
POP	Extrae de la Pila.	2	1
PUSH	Inserta en la Pila.	2	1
RCL	Rota el Operando a la Izquierda a través del Acarreo.	3	2-9
RCR	Rota el Operando a la Derecha a través del Acarreo.	3	2-9
REPcc	Repite la siguiente Instrucción de Cadena.	-	4
RET	Retorno de Subrutina.	-	-
ROL	Rota el Operando a la Izquierda.	3	2
ROR	Rota el Operando a la Derecha.	2	2

SOTIR *	Escribe Puerto Incre_ menta y Repite Espec.	1,2	3
SOUT *	Escribe Puerto Especial	1,2	3,4
SOUTD *	Escribe Puerto y Decrementa Especial	1,2	3
SOUTI *	Escribe Puerto e Incrementa Especial	1,2	3
SRA	Corrimiento Aritmé_ tico a la Derecha	4	1
SRL	Corrimiento Lógico a la Derecha	4	1
SUB	Substracción	4	1-5
TCC	Checa Código de Condición.	1,2	1
TEST	Compara contra cero	4	1,3-5
TRDB	Traduce y Decrementa	1	3
TRDBR	Traduce Decrementa y Repite	1	3
TRIB	Traduce e Incrementa	1	3
TRIRB	Traduce Incrementa y Repite	1	3
TRTDB	Traduce Prueba y Decrementa	1	3
TRTDRB	Traduce Prueba Decrementa y Repite	1	3
TRTIB	Traduce Prueba e Incrementa	1	3
TRTIRB	Traduce Prueba Incrementa y Repite	1	3
TSET	Prueba y Prende	1,2	1,3-5
XOR	"O Exclusivo"	1,2	1-5
XCTL	Operación Interna de EPU	1	4
XLDCTL	Carga EPU Desde/Hacia	1	4
	Banderas	1	4
XLDM	Carga Registros/Mem. Desde/Hacia EPU	2	1

APENDICE B. MICROPROCESADOR Z8000.

DESCRIPCION DE SEÑALES

AD0-AD15 CANAL DE DIRECCIONES/DATOS Bidireccional, de 3 Estados estas líneas multiplexadas de Datos/Direcciones son usadas para direccionar memoria y E/S así como para transferir datos desde y hacia el procesador.

AS* HABILITACION DE DIRECCION Salida, 3 Estados, El frente de subida de AS* indica que las direcciones son válidas.

BUSACK* RECONOCIMIENTO DE CANAL Salida, un nivel bajo en esta salida indica que el CPU ha otorgado el control de los canales.

BUSREQ* SOLICITUD DE CANAL Entrada, esta línea deberá ser forzada a un nivel bajo para solicitar los Canales al CPU.

DS* HABILITACION DE DATOS Salida, 3 Estados, esta línea indica que hay un dato válido sobre el canal.

MREQ* SOLICITUD DE MEMORIA Salida, 3 Estados, un nivel bajo en esta línea indica que el Canal de Datos/Direcciones contiene una dirección de memoria.

MI*, MO* ENTRADA MULTI-MICRO, SALIDA MULTI-MICRO Entrada y Salida, estas dos líneas forman una cadena de solicitud de recursos la cual permite que un CPU en un sistema de Multiproceso solicite un recurso compartido.

NMI* INTERRUPCION NO MASCARABLE Entrada, una transición de un nivel alto a uno bajo en NMI* solicita una interrupción no mascarable.

NVI* INTERRUPCION SIN VECTOR Entrada, un nivel bajo en esta línea solicita una interrupción sin vector.

CLK RELOJ DE SISTEMA Entrada, es una entrada de base de tiempo de 5 Volts y una sola fase.

RESET* RESTAURACION Entrada, un nivel bajo en esta entrada restaura al CPU.

R/W* LECTURA/ESCRITURA Salida, 3 Estados, R/W* indica que el CPU está leyendo/escribiendo a memoria o E/S.

SN0-SN6 NUMERO DE SEGMENTO Salidas, 3 Estados estas líneas proporcionan el número de segmeto de 7 bits usado para direccionar uno de los 128 segmentos disponibles.

SEGT* TRAMPA DE SEGMENTACION Entrada, la Unidad de Manejo de Memoria interrumpe al CPU con un nivel bajo en esta línea cuando detecta una trampa de segmentación.

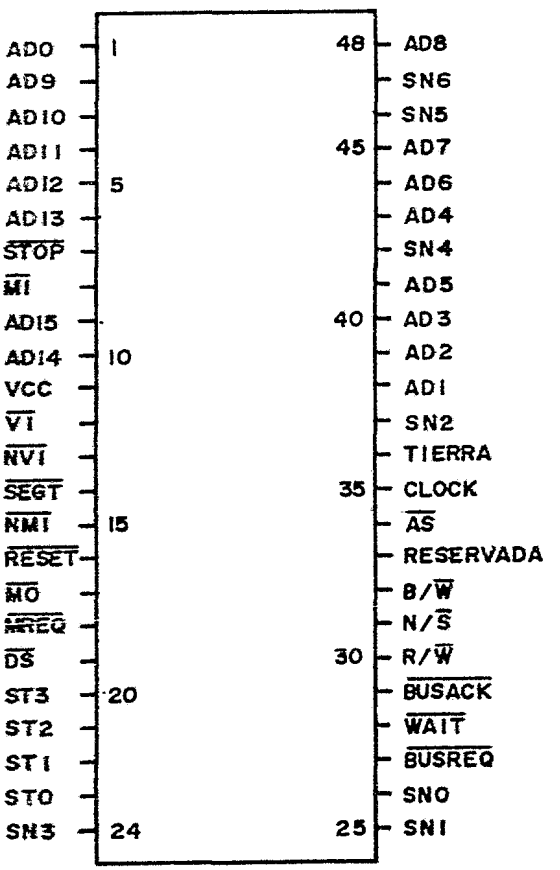


FIG 74 CONFIGURACION DE LINEAS

ST0-ST3 ESTADO Salidas, 3 Estados estas líneas especifican el estado del CPU.

STOP* PARO Entrada, esta entrada puede ser usada para ejecutar instrucciones paso a paso.

VI* INTERRUPCION CON VECTOR Entrada, un nivel bajo en esta línea solicita una interrupción con vector.

WAIT* Entrada, esta linea indica al CPU que la memoria o el dispositivo de E/S no está listo para la transferencia de datos.

B/W* BYTE/PALABRA Salida, 3 Estados esta señal define el tipo de referencia a memoria en el Canal de datos de 16 bits.

N/S* MODO NORMAL/SISTEMA Salida, 3 Estados, esta línea indica si el CPU está en modo Normal o en modo de Sistema.

SUMARIO DEL CONJUNTO DE INSTRUCCIONES

El Z8000 proporciona los siguiente tipos de Instrucciones:

1. Instrucciones de Carga e Intercambio.
2. Instrucciones Aritméticas.
3. Instrucciones Lógicas.
4. Instrucciones de Control de Programa.
5. Instrucciones de Manipulación de Bits.
6. Instrucciones de Rotación y Corrimiento.
7. Instrucciones de Transferencia de Bloques y Manipulación de Cadenas (Strings).
8. Instrucciones de Entrada/Salida.
9. Instrucciones de Control del CPU.

A continuación se presenta el conjunto de instrucciones del procesador indicando los modos de direccionamiento y los tipos de datos que puede utilizar cada una de ellas, se listan los mnemónicos utilizados para operaciones con palabras, si se desea utilizar la operación para bytes, se deberá agregar "B" al final del mnemónico, si se desea utilizarla para palabra larga se deberá agregar "L" al final

del mnemónico indicado. Las instrucciones privilegiadas se indican con un asterisco (*) al final del mnemónico respectivo.

TIPOS DE DATOS:

1 BYTE
2 PALABRA
3 PALABRA LARGA
4 TODOS

MODOS DE DIRECCIONAMIENTO:

1 DE REGISTRO
2 INMEDIATO
3 INDIRECTO
4 DIRECTO
5 INDEXADO
6 RELATIVO AL PC
7 DIRECCION BASE
8 BASE INDEXADO
9 TODOS

MNEMONICO	INSTRUCCION	T. DATOS	M .DIRECCION.
ADC	Suma con acarreo	1,2	1
ADD	Suma	4	1-5
AND	"Y" Lógico	1,2	1-5
BIT	Prueba un Bit	1,2	1-5
CALL	Llama Subrutina	-	3-5
CALR	Llama Subrutina con direcc. relativo	-	6
CLR	Borra operando	1,2	1,3-5
COM	Complemento a Uno	1,2	1,3-5
COMFLG	Complementa Banderas de Status	-	2
CP	Compara	4	1-5
CPD	Compara y Decrementa	1,2	3
CPDR	Compara Decrementa y Repite	1,2	3
CPI	Compara e Incrementa	1,2	3
CPIR	Compara Incrementa y Repite	1,2	3
CPSD	Compara Series y Decrementa	1,2	3
CPSDR	Compara Series Decrementa y Repite	1,2	3
CPSI	Compara Series e Incrementa	1,2	3
CPSIR	Compara Series Incrementa y Repite	1,2	3
DAS	Ajuste Decimal	1	1

DEC	Decrementa	1,2	1,3-5
DI *	Deshabilita Interrupciones	-	-
DIV	Divide	2,3	1-5
DJNZ	Decrementa y Salta si no es cero	1,2	1
EI *	Habilita Interrupciones	-	-
EX	Intercambia	1,2	1,3-5
EXTS	Extiende Bit de signo	4	1
HALT *	Paro	-	-
IN *	Lee de Puerto	1,2	3,4
INC	Incrementa	1,2	1,3-5
IND *	Lee de Puerto y Decrementa	1,2	3
INDR *	Lee Puerto Decrementa y Repite	1,2	3
INI *	Lee Puerto e Incrementa	1,2	3
INIR *	Lee Puerto e Incrementa	1,2	3
IRET *	Retorno de Interrupción	-	-
JP	Salto Condicional	-	3-5
JR	Salto Condicional Relativo	-	6
LD	Mueve Desde/Hacia Registro	4	9
LDA	Carga Dirección	2,3	4,5,7,8
LDAR	Carga Dirección Relativa	2,3	6
LDCTL *	Carga Registro de Control	1,2	1
LDD	Mueve y Decrementa	1,2	3
LDDR	Mueve Decrementa y Repite	1,2	3
LDI	Mueve e Incrementa	1,2	3
LDIR	Mueve Incrementa y Repite	1,2	3
LDK	Carga Constante	2	2
LDM	Carga Múltiple	2	1,3-5
LDPS *	Carga Status de Programa	2	3-5
MBIT *	Checa Línea Multi-Micro.	-	-
MREQ *	Solicitud de Multi-Micro	-	1
MRES *	Desactiva Línea Multi-Micro	-	-
MSET *	Activa Línea Multi-Micro	-	-
MULT	Multiplica	2,3	1-5

NEG	Complemento a Dos	1,2	1,3-5
NOP	No Operación	-	-
OR	"O" Lógico	1,2	1-5
OTDR *	Escribe Puerto Decrementa y Repite	1,2	3
OTIR *	Escribe Puerto Incrementa y Repite	1,2	3
OUT *	Escribe Puerto	1,2	3,4
OUTD *	Escribe Puerto y Decrementa	1,2	3
OUTI *	Escribe Puerto e Incrementa	1,2	3
POP	Extrae de la Pila	2,3	1,3-5
PUSH	Coloca en la Pila	2,3	1-5
RES	Borra un Bit	1,2	1,3-5
RESFLG	Borra Banderas	-	-
RET	Retorno de Subrutina	-	-
RL	Rotación a la Izq. y en el Acarreo	1,2	1
RLC	Rotación a la Izq. a Través del Acarreo	1,2	1
RLDB	Rotación a la Izq. BCD	1	1
RR	Rotación a la Der. y en el Acarreo	1,2	1
RRC	Rotación a la Der. a Través del Acarreo	1,2	1
RRDB	Rotación a la Der. BCD	1	1
SBC	Substracción con Préstamo	1,2	1
SC	Llamada de Sistema	-	-
SDA	Corrimiento Aritmé_ tico.	4	1
SDL	Corrimiento Lógico	4	1
SET	Prende un Bit	1,2	1,3-5
SETFLG	Prende Bandera	-	-
SIN *	Lee Puerto Especial	1,2	3,4
SIND *	Lee Puerto y Decrementa Especial	1,2	3
SINDR *	Lee Puerto Decrementa y Repite Especial	1,2	3
SINI *	Lee Puerto e Incrementa Especial	1,2	3
SINIR *	Lee Puerto Incrementa y Repite Especial	1,2	3
SLA	Corrimiento Aritmé_ tico a la Izquierda	4	1
SLL	Corrimiento Lógico a la Izquierda	4	1
SOTDR *	Escribe Puerto Decre_ menta y Repite Espec.	1,2	3

SOTIR *	Escribe Puerto Incre_ menta y Repite Espec.	1,2	3
SOUT *	Escribe Puerto Especial	1,2	3,4
SOUTD *	Escribe Puerto y Decrementa Especial	1,2	3
SOUTI *	Escribe Puerto e Incrementa Especial	1,2	3
SRA	Corrimiento Aritmé_ tico a la Derecha	4	1
SRL	Corrimiento Lógico a la Derecha	4	1
SUB	Substracción	4	1-5
TCC	Checa Código de Condición.	1,2	1
TEST	Compara contra cero	4	1,3-5
TRDB	Traduce y Decrementa	1	3
TRDBR	Traduce Decrementa y Repite	1	3
TRIB	Traduce e Incrementa	1	3
TRIRB	Traduce Incrementa y Repite	1	3
TRTDB	Traduce Prueba y Decrementa	1	3
TRTDRB	Traduce Prueba Decrementa y Repite	1	3
TRTIB	Traduce Prueba e Incrementa	1	3
TRTIRB	Traduce Prueba Incrementa y Repite	1	3
TSET	Prueba y Prende	1,2	1,3-5
XOR	"O Exclusivo"	1,2	1-5
XCTL	Operación Interna de EPU	1	4
XLDCTL	Carga EPU Desde/Hacia	1	4
	Banderas	1	4
XLDM	Carga Registros/Mem. Desde/Hacia EPU	2	1

APENDICE C. MICROPROCESADOR 68000.

DESCRIPCION DE SEÑALES

CANAL DE DIRECCIONES (A1-A23). Líneas bidireccionales de 3 estados, proporcionan la dirección para cualquier operación de canal.

CANAL DE DATOS (D0-D15). Líneas bidireccionales de tres estados forman la ruta de Datos de propósito general.

CONTROL ASINCRONO DE CANAL. Las transferencias asíncronas de datos son manejadas utilizando las siguientes líneas de control:

HABILITACION DE DIRECCION (AS*). Salida, indica que hay una dirección válida sobre el Canal de datos.

LECTURA/ESCRITURA (R/W*). Salida, indica si la transferencia del Canal de datos es para lectura o para escritura.

HABILITACIONES DE DATOS ALTO Y BAJO (UDS*, LDS*). Salidas, controlan el tipo de dato que será manejado en el Canal.

RECONOCIMIENTO DE TRANSFERENCIA DE DATO (DTACK*). Entrada, indica que la transferencia ha sido terminada. Cuando DTACK* es reconocida durante un ciclo de escritura el ciclo de Canal es terminado.

CONTROL DE ADMINISTRACION DE CANAL. Estas tres líneas descritas a continuación determinan cual dispositivo tomará el control de los Canales.

SOLICITUD DE CANALES (BR*). Entrada, está alambrada en OR con todos los demas dispositivos que pueden ser maestros del Canal. Esta línea indica al procesador que algún dispositivo está solicitando los Canales.

OTORGAMIENTO DE CANALES (BG*). Salida, indica a todos los dispositivos que pueden ser maestros del Canal, que liberará los Canales en cuanto termine el presente ciclo de máquina.

RECONOCIMIENTO DE CONCESION DE CANALES (BGACK*). Entrada, indica que un dispositivo ha tomado el control de los canales.

CONTROL DE INTERRUPCION (IPL0*, IPL1*, IPL2*). Líneas de entrada, los dispositivos que solicitan una interrupción indican el nivel de prioridad a la que tienen acceso.

CONTROL DE SISTEMA. Las entradas de control de sistema son utilizadas para restaurar, detener al procesador o para indicar que ha ocurrido un Error de Canal y se describen a continuación.

ERROR DE CANAL (BERR*). Esta entrada informa al procesador que hay un problema con el ciclo que se está ejecutando.

RESTAURACION (RESET*). Línea bidireccional, es utilizada como entrada para restaurar al procesador. Como salida es utilizada junto con la instrucción RESET para restaurar dispositivos externos.

ALTO (HALT*). Línea bidireccional es activada por un dispositivo externo, causa que el procesador se detenga al final del ciclo de Canal que se está ejecutando.

CONTROL DE PERIFERICOS 6800. Líneas de control, se usan para permitir una interfase síncrona con dispositivos periféricos 6800 y con dispositivos asíncronos 68000. Estas señales se explican a continuación:

HABILITACION (E). Salida, es igual que la usada en todos los periféricos 6800. El período para esta salida es de diez períodos de reloj.

DIRECCION DE PERIFERICO VALIDA (VPA*). Entrada, indica que el dispositivo direccionado es uno de la familia 6800 y que la transferencia de datos deberá ser sincronizada con la señal Habilitación (E). Esta señal también indica que el procesador deberá proporcionar un autovector si hay una interrupción.

DIRECCION DE MEMORIA VALIDA (VMA*). Salida, es usada para indicar a los periféricos 6800 que hay una dirección válida sobre el Canal de Direcciones y que el procesador esta sincronizado con la señal (E).

ESTADO DEL PROCESADOR (FC0, FC1, FC2). Líneas de salida, proporcionan un código de función que indica el modo del procesador (usuario o supervisor) y el tipo de ciclo que se está ejecutando.

RELOJ (CLK). La entrada de reloj es una señal de nivel TTL y deberá ser de frecuencia constante.

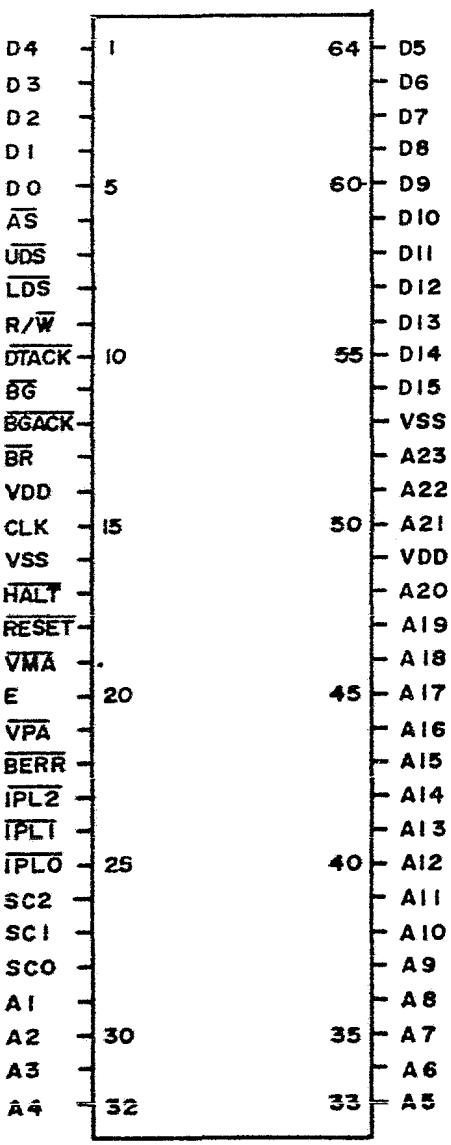


FIG 75 CONFIGURACION DE LINEAS

CONJUNTO DE INSTRUCCIONES

A continuación se presenta un sumario de instrucciones del 68000. Cada instrucción, con pocas excepciones, opera con bytes, palabras y palabras largas y la mayoría de las instrucciones pueden utilizar los 14 modos de direccionamiento. Combinando tipos de instrucción, tipos de datos y los modos de direccionamiento se tienen más de 1000 instrucciones disponibles.

Las instrucciones del 68000 están divididas en los siguientes grupos fundamentales:

Movimientos de Datos
Operaciones Aritméticas
Operaciones Lógicas
Corrimientos y Rotaciones
Manipulación de Bits
Operaciones en BCD
Control de Programa
Sistema de control

A continuación se listan las instrucciones en orden alfabético, listando el mnemónico, el nombre de la instrucción, los tipos de datos que puede manejar y los modos de direccionamiento. Las instrucciones que lleven un asterisco indican que son instrucciones privilegiadas y sólo se podrán ejecutar en modo supervisor.

TIPOS DE DATOS:

1 BYTE.
2 PALABRA.
3 BYTE O PALABRA.
4 PALABRA LARGA.
5 PALABRA Y/O PALABRA LARGA.
6 TODAS.
7 BYTE O PALABRA LARGA.

MODOS DE DIRECCIONAMIENTO:

1 REGISTRO DE DATOS DIRECTO
2 REGISTRO DE DIRECCIONES DIRECTO
3 INDIRECTO
4 INDIRECTO CON POSTINCREMENTO
5 INDIRECTO CON PREDECREMENTO

6	INDIRECTO CON DESPLAZAMIENTO
7	INDIRECTO CON INDICE
8	ABSOLUTO CORTO
9	ABSOLUTO LARGO
10	RELATIVO AL PC CON DESPLAZAMIENTO
11	RELATIVO AL PC CON INDICE
12	INMEDIATO
13	TODOS

MNEMONICO	INSTTRUCCION	T. DATOS	MODOS
ABCD	Suma Decimal.	1	1,5
ADD	Suma Binaria.	6	13
ADDA	Suma Direcciones.	4	13
ADDI	Suma Inmediata.	6	1,3-9
ADDQ	Suma Rápida.	6	1-9
ADDX	Suma Extendida.	6	1,5
AND	"Y" Lógico.	6	1,3-12
ANDI	"Y" Log. Inmediato.	6	1,3-9
ANDI TO CCR*	"Y" Inmediato para Códigos de Condición.	6	12
ANDI TO SR*	"Y" Inmediato para Registro de Status.	2	12
ASL	Corrimiento Aritmético a la Izquierda.	6	3-9
ASR	Corrimiento Aritmético a la Derecha.	6	3-9
BCC	Brinco Condicional.	3	10
BCHG	Prueba un Bit y Cámbialo.	7	1,3-9
BCLR	Prueba un Bit y Apágalo.	7	1,3-9
BRA	Brinca Siempre.	3	10
BSET	Prueba un Bit y Enciéndelo.	7	1,3-9
BSR	Brinca a Subrutina.	7	10
BTST	Prueba un Bit.	3	1,3-11
CHK*	Prueba un Registro contra Límites.	2	1,3-12
CLR	Borra un Operando.	6	1,3-9
CMP	Compara.	6	13
CMPA	Compara Dirección.	5	13
CMPI	Compara Inmediato.	6	1,3-9
CMPM	Compara Memoria.	6	4
DBCC	Prueba Condición, Decrementa y Brinca.	6	10
DIVS	División Signada.	2	1,3-12
DIVU	División No Signada.	2	1,3-12
EOR	"O" Exclusivo Lógico.	6	1,3-9
EORI	"O" Exclusivo Lógico Inmediato.	6	1,3-9

EORI TO CCR *	Igual pero para Códigos de Condición.	1	12
EORI TO SR*	Igual pero para Registro de Status.	2	12
EXG	Intercambia Registros.	4	1,2
EXT	Extensión de Signo.	5	1
ILLEGAL	Intrucción Ilegal.	—	—
JMP	Salta.	—	3,6,7-11
JSR	Salta a Subrutina.	—	3,6,7-11
LEA	Carga Dirección Efectiva.	4	3,7-11
LINK	Liga y Localiza.	—	10(SP)
LSL	Corrimiento Lógico a la Izquierda.	6	3-9
LSR	Corrimiento Lógico a la Derecha.	6	3-9
MOVE	Mueve Datos de Fuente a Destino.	6	13
MOVE TO CCR*	Mueve para Códigos de Condición.	2	1,3-12
MOVE TO SR*	Mueve hacia el Registro de Status.	2	1,3-12
MOVE FROM SR*	Mueve desde el Registro de Status.	2	1,3-9
MOVE USP*	Mueve al Apuntador de Pila modo Usuario.	4	2
MOVEA*	Mueve Direcciones.	5	13
MOVEM	Mueve Múltiple	5	3,4,5,6-11
MOVEP	Mueve un Dato a Periférico.	5	6
MOVEQ	Mueve Rápido.	4	12
MULS	Multiplicación Signada.	2	1,3-12
MULU	Multiplicación No Signada	2	1,3-12
NBCD	Niega Decimal	1	1,3-9
NEG	Niega Complemento a 2's	6	1,3-9
NEGX	Niega extendido.	6	1,3-9
NOP	No Operación.	—	—
NOT	Complemento Lógico	6	1,3-9
OR	"O" Lógico.	6	1,3-12
ORI	"O" Lógico Inmediato.	6	1,3-9
ORI TO CCR	"O" Inmediato para Código de Condición.	1	12
ORI TO SR*	"O" Inmediato para Registro de Status.	2	12
PEA	Guarda en la Pila la Dirección Efectiva	4	3,6-11

RESET*	Restaura Dispositivos Externos.		
ROL	Rota a la Izquierda.	6	3-9
ROR	Rota a la Derecha.	6	3-9
ROXL	Rota a la Izquierda Extendido.	2	3-9
ROXR	Rota a la Derecha Extendido.	2	3-9
RTE*	Retorno de Excepción	—	—
RTR	Retorno y Re-almacena Códigos de Condición.	—	—
RTS	Retorno de Subrutina.	—	—
SBCD	Resta Decimal Extendido.	1	1,5
SCC	Enciende de acuerdo a Condición.	1	1,3-9
STOP*	Carga el Registro de Status y Para.		
SUB	Resta Binaria.	6	13
SUBA	Resta Direcciones.	5	13
SUBI	Resta Inmediato.	6	1,3-9
SUBQ	Resta Rápido.	6	1,9
SUBX	Resta Extendido.	6	1,5
SWAP	Intercambia dos Registros	2	1
TAS	Prueba y Enciende un Operando.	1	1,3-9
TRAP*	Trampa.	—	—
TRAPV*	Trampa en Sobreflujo.		
TST	Prueba un Operando.	6	1,3-9
UNLK	Desliga.	—	—

BIBLIOGRAFIA:

1. MCS-85 Product Description.
Intel Corporation.
Febrero 1979.

2. IAPX 86,88 User's Manual
Intel Corporation.
Julio 1981.

3. Databook Z8000 Microprocessor Family
SGS/ATES Group.
1981

4. Z8001/2 CPU Product Specification (Preliminary)
Zilog Inc.
1979

5. An Introduction to the Z8010 MMU Memory Management Unit (Tutorial Information)
Zilog Inc.
1979

6. Zilog Z8000 Family Technical Overview.
Zilog Inc.
1979

7. A Small Z8000 System (Application Note).
Zilog Inc.
1979, 1980

8. AMZ8001/2 Processor Instruction Set.
Advanced Micro Devices Inc.
1979

9. MOTOROLA, 16-BIT MICROPROCESSOR, User's Manual
Third Edition
1982

10. MOTOROLA, The M68000 Family
1982

11. Computer System Architecture.
Mano, Morris.
Editorial Prentice Hall.
1976

12. Computer Logic Design.
Mano, Morris.
Editorial Prentice Hall.
1972

13. The Art of Computer Programming.
Volume 3 Sorting and Searching.
Knuth, Donald E.
Editorial Addison Wesley.
1975

14. The 8086 Book
Rector, Rusell y Alexy, George.
Osborne/McGraw Hill
1980

15. Z8000 Assembly Language Programming
Leventhal A. Lance, Osborne Adam y Collins Chuck.
Editorial Osborne/McGraw Hill Inc.
1980

16. 68000 Assembly Language Programming.
Kane Jerry, Hawkins Doug, Leventhal Lance.
McGraw-Hill/Osborne
1981.

17. The 68000 Principles and Programming
Leo J. Scanlon. W. Sams.
Indiana
1981

18. A History of Microprocessor Developement at Intel.
Noyce, R.N. y Hoff, M.E.
IEEE Micro Vol. 1 No. 1 PP 8-21
Febrero 1981

19. A tale of four uPs: Benchmarks Quantify Performance.
Grappel, Robert E. y Hemenway, Jack E.
EDN
Abril 1, 1981

20. Comparison of the Z8000 and the MC68000 Microprocessors (A soft point of view).
Jose Abraham y SP Mudur
Technical Report 47
National Centre for Software Development and Computing Techniques Tata Institute of Fundamental Research. Bombay, India.
Diciembre 1979

21. An Architectural Comparison of Contemporary 16 Bit Microprocessors.
Hoo Min Tong y Amar Gupta
IEEE Micro Vol. 1 No. 2 pp. 26-37 i 0 Mayo, 1981

22. Two Versions of 16 Bit Chip Span Microprocessor, Minicomputer Needs.
Shima, Masatoshi
Electronics Magazine
Diciembre 21 1978
PP 81 a 88

23. Architecture of a New Microprocessor.
Peuto, Bernard L.
IEEE Computer Magazine.
Febrero 1979
PP 10 a 21

24. MC68000 16-bit Microprocessor to offer wide Address Range, Powerful Comands
Bursky, Dave.
Electronic design
15 Julio de 1978.

25. Complex Systems are Simple to Design.
Nobis, Lemair and Robert.
Electronic Design
18 Septiembre 1978.

26. Microprogrammer Implementation of a Single Chip Microprocessor.
Stritter, Skip y Tredennick, Nick.
11th Annual Microprogramming Workshop.
Diciembre 1978.

27. A Microprocessor Architecture for a Changing World: The Motorola 68000.
Stritter, Edward y Gunter, Tom.
IEEE Computer
Febrero 1979.

28. Design and Implementation of Systems Features for the MC68000.
Zolnowski, John y Tredennick, Nick.
Proceedings of Compcon
Otoño 1979.

29. Develop Software for 16-bit uC Whithout Making Costly Commitments.
Kister, Jack y Naugle, Raymond.
Electronic Desing 19
Septiembre 13, 1979.

30. Compact Instructions Give the MC68000 Power While Simplifying its Operation.
Starnes, Thomas W.
Electronic Design 20
Septiembre 27,1979.

31. Microprogramming Makes the MC68000 a Processor Ready for the Future.
Bryce, Heather
Electronic Desing 22
Octubre 25, 1979.

32. Learn the Timing and Interfacing of MC68000 Peripheral Circuits.
Stockton, John y Scherer, Victor.
Electronic Desing 23
Noviembre 8, 1979.

33. MC68000 Microprocessor Combines Powerful Instruction Set with 16 MByte Adressing Range .
Farrel, Jim.
Electronic Products
Octubre 1979.

34. 16-Bit 68000 Microprocessor Camps on 32-Bit Frontier.
Hartman, Brad.
Electronics
Octubre 11, 1979.

35. Development System Suports Today's Processors -- And Tomorrow's
Kister, Jack Kister y Robinson, Irwin.
Electronics
Enero 31,1980.

36. The MC68000-A 32-Bit uP Masquerading as a 16-Bit Device.
Grappel, Robert y Hemenway, Jack.
EDN
Febrero 20, 1980.