ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

DISEÑO Y CONSTRUCCION DE UNA TARJETA DE ADQUISICION DE DATOS PARA COMPUTADORES PERSONALES

Tesis previa a la obtención del título de Ingeniero en Electrónica y Control

FAUSTO CEVALLOS A.

QUITO, MARZO DE 1993

CERTIFICO:

Que el presente trabajo ha sido desarrollado en su totalidad por el Sr. Fausto Cevallos A.

Ing. Bolívar Ledesma G.

DIRECTOR DE TESIS

INDICE

INTRODUCCION 1

CAPITULO I: AMBIENTE DE TRABAJO 4

1.1 EL MICROPROCESADOR 8088 6

1.1.1 Historia 6

1.1.2 Características 10

1.1.3 Registros 12

1.1.4 Banderas 16

1.1.5 Segmentos de memoria 17

1.1.6 Operandos, instrucciones y modos de direccionamiento 18

1.1.7 Diseño de sistemas basados en el 8088 . . 22

1.2 LOS COMPUTADORES PERSONALES XT COMPATIBLES . . 26

1.2.1 Historia 26

1.2.2 Chips utilizados en un XT 27

1.2.3 Interrupciones "hardware" 29

1.2.4 Interrupciones "software" 30

1.2.5 La memoria del PC 31

1.2.6 Cadenas de texto 34

1.2.6.1 El juego de caracteres ASCII 34

1.2.7 Discos 37

1.2.8 Video 39

1.2.9 Teclado 40

1.2.10 Parlante 40

1.2.11 ROM-BIOS 41

1.2.11.1 Interrupciones de servicio de

la ROM-BIOS 42
1.2.12 El Sistema Operativo 46

1.3 ENLACE DE LA TARJETA DAS-128 CON EL PC 58
1.3.1 El conector para tarjetas de extensión . . 58
1.3.1.1 Descripción de las señales disponibles en el conector para tarjetas de extensión del PC 60
1.3.2 Pórticos del PC 64
1.3.3 Señales y direcciones utilizadas por la tarjeta DAS-128 65

CAPITULO II: REQUERIMIENTOS Y DISEÑO 68

2.1 ESPECIFICACIONES INICIALES 69

2.2 DESCRIPCION DE LA TARJETA DAS-128 71
2.2.1 "Buffers" de entrada 71
2.2.2 Lógica de decodificación 71
2.2.3 Puertos 72

2.3 DISEÑO 73
2.3.1 Diseño del "hardware" 73
2.3.1.1 Buffers de entrada 73
2.3.1.2 Puertos digitales 74
2.3.1.3 Puerto digital de entrada/salida . . 75
2.3.1.4 Puertos de entradas analógicas . . . 76
2.3.1.5 Puertos de salidas analógicas 78
2.3.1.6 Decodificación de la dirección . . . 79
2.3.2 Diseño del circuito impreso 84

CAPITULO III: PRUEBAS Y RESULTADOS 86

3.1 PROGRAMAS DE DEMOSTRACION 87

3.1.1 Programa No. 1, Comprobación de puertos . 87

3.1.1.1 Listado del Programa No. 1 en Quick Basic 88

3.1.2 Programa No. 2, Obtención de señal rampa . 90

3.1.2.1 Listado del Programa No. 2 en Quick Basic 91

3.1.2.2 Listado del Programa No. 2 en Lenguaje C 92

3.1.2.3 Listado del Programa No. 2 en lenguaje ensamblador 93

3.1.3 Programa No. 3, Operaciones lógicas . . . 93

3.1.3.1 Listado del Programa No. 3 en Quick Basic 94

3.1.4 Programa No. 4, Muestreo de señal analógica 96

3.1.4.1 Listado del Programa No. 4 en Quick-Basic 96

3.1.4.2 Listado del Programa No. 4 en Lenguaje C 97

3.2 PRUEBAS DE FUNCIONAMIENTO 98

3.2.1 Pruebas de compatibilidad 98

3.2.2 Pruebas de correcta operación 99

3.2.3 Pruebas de frecuencia de operación 99

3.3 RESULTADOS OBTENIDOS 101

3.3.1 Pruebas de compatibilidad 101

3.3.2 Pruebas de correcta operación 101
3.3.3 Pruebas de frecuencia de operación . . . 101

3.4 ANALISIS TECNICO ECONOMICO 108
3.4.1 Análisis económico 108
3.4.2 Análisis técnico 109

CAPITULO IV: CONCLUSIONES Y RECOMENDACIONES 111

4.1 DISCUSION DE LOS RESULTADOS OBTENIDOS 111

4.2 RECOMENDACIONES 113

4.3 CONCLUSIONES 115

ANEXOS

ANEXO 1: INSTRUCCIONES DEL 8088
ANEXO 2: DIAGRAMAS ESQUEMATICOS DEL PC XT
ANEXO 3: RUTINAS DE SERVICIO DEL DOS
ANEXO 4: DIAGRAMA ESQUEMATICO DE LA TARJETA DAS-128
ANEXO 5: CIRCUITO IMPRESO DE LA TARJETA DAS-128
ANEXO 6: MANUAL DE USUARIO DE LA TARJETA DAS-128

BIBLIOGRAFIA

INTRODUCCION

INTRODUCCION

El control y supervisión de procesos se ha desarrollado en un tiempo muy corto, debido a las innegables ventajas que ofrece. El uso de un computador personal, para este efecto, abre un espectro de aplicaciones muy amplio, con poco esfuerzo en la programación debido a la gran cantidad de lenguajes disponibles.

Lamentablemente, en su configuración básica, el computador personal no dispone de puertos de entrada y salida, analógicos y digitales, que permitan realizar acciones de supervisión y control directamente, existiendo, sin embargo, tarjetas que se comercializan para este efecto, pero con costos muy altos y con un número de puertos muy reducido.

El presente trabajo tiene como objetivo el diseño y construcción de una tarjeta de adquisición de datos, a la que hemos denominado DAS-128, que incluye pórticos de entrada y salida, tanto digitales como analógicos, y que puede ser conectada al bus interno, para tarjetas de extensión, de cualquier computador personal XT o AT, IBM compatible.

En la tarjeta se dispone de 14 pórticos digitales, de ocho bits TTL; siete pueden ser utilizados únicamente como salidas y siete como entradas. Se dispone de una entrada de control para uno de los pórticos de entradas y uno de salidas que permite su operación bidireccional. Dicha señal puede ser comandada por cualquiera de las salidas restantes disponibles en la tarjeta, o por una señal externa.

Los 16 pórticos analógicos, de 8 bits (8 de entrada y 8 de salida), funcionan con voltajes normalizados entre 0 y 10 (VDC).

En el primer capítulo se describe al micro-procesador 8088, complementándose la información necesaria para poder utilizar este micro en el Anexo No. 1, en el que se incluyen sus instrucciones para programación en lenguaje ensamblador.

Se describen además, en el mismo capítulo, algunos detalles importantes que permiten un conocimiento más profundo de la arquitectura y uso de señales en los computadores personales, que se complementan con el Anexo No. 2, en el que se presentan los diagramas esquemáticos de un computador personal XT compatible, y con el Anexo No. 3, en el que se describen los principales servicios de interrupción que provee el sistema operativo DOS.

El primer capítulo concluye con una descripción de las señales que provee el bus de ampliaciones, de ocho bits, del

computador personal, indicándose aquellos que se utilizarán para el desarrollo de la tarjeta de adquisición de datos DAS-128, materia del presente trabajo.

En el segundo capítulo se describen las partes que componen la tarjeta de adquisición de datos DAS-128, los puntos considerados para su diseño, así como aspectos de su construcción.

En el capítulo III se realizan algunas subrutinas en diferentes lenguajes de programación que, además de ser utilizadas para evaluar el funcionamiento de la tarjeta DAS-128, permiten determinar las ventajas de utilizar uno u otro lenguaje para aplicaciones específicas.

El capítulo III concluye realizando un análisis técnico económico de la tarjeta DAS-128 construida.

En el capítulo IV se realiza una discusión de los resultados obtenidos, un análisis comparativo de la tarjeta con otras de uso similar, y se enuncian varias conclusiones y recomendaciones respecto del trabajo realizado.

Los Anexos Nos. 4, 5 y 6 se refieren exclusivamente a la tarjeta DAS-128; en el Anexo No. 4 se encuentran sus diagramas esquemáticos, en el Anexo No. 5 los diagramas del circuito impreso y el Anexo No. 6 es un manual para los usuarios de la tarjeta DAS-128.

CAPITULO I

AMBIENTE DE TRABAJO

CAPITULO I

AMBIENTE DE TRABAJO

Este capítulo describe un computador personal XT compatible, debido a que éste es el sistema mínimo para el cual se ha diseñado la tarjeta DAS-128, materia del presente trabajo; los demás modelos de computadores personales incluyen líneas adicionales a las del XT, en ampliaciones al bus original, pero manteniendo el bus de ocho bits usado por los XT.

El capítulo se inicia con la descripción del micro-procesador 8088, que es el utilizado en los computadores personales XT compatibles, describe luego al computador y concluye con una descripción del bus disponible para las tarjetas de extensión, prestando mayor atención a las líneas que se utilizarán para conectar la tarjeta de adquisición de datos DAS-128.

Adicionalmente se incluye un listado de las direcciones de los dispositivos de entrada/salida existentes normalmente en un

computador personal, con el fin de definir las direcciones disponibles, que pueden ser utilizadas por la tarjeta DAS-128 y, en general, por otras tarjetas de extensión.

1.1 EL MICROPROCESADOR 8088

1.1.1 Historia

La primera máquina de calcular fue posiblemente el ábaco, desarrollado en China en el año 500 A.C. y que se utiliza hasta la actualidad.

En 1617, John Napier mostró un proceso para efectuar multiplicaciones y en 1642 Blaise Pascal construyó una calculadora mecánica, que fuera perfeccionada por Samuel Morland y en 1694 por von Leibniz.

En el siglo XVII, Jacquard inventó un telar programado con tarjetas perforadas. Babbage, en 1833, junta las innovaciones de Jacquard y Pascal para desarrollar su máquina analítica.

Hollerith desarrolló una máquina electro-mecánica que trabajaba con tarjetas perforadas, la misma que sería utilizada en el censo de los estadounidenses de 1885.

Powers, a principio de este siglo, diseñó equipos industriales que funcionaban con tarjetas perforadas para trabajos contables y estadísticos. Estas máquinas fueron fabricadas por Remington Rand en Estados Unidos y por Samas en Europa. En esta misma época, Watson creó IBM.

De 1936 a 1939, los estadounidenses Aiken y Stibitz desarrollaron una calculadora utilizando relés electro-mecánicos.

En 1944 apareció la primera máquina electrónica ENIAC, creada por los profesores Eckert y Mauchly, en la Universidad de Pensilvania. Utilizaba 18000 tubos y trabajaba a programa cableado. Los mismos profesores, en 1950 propusieron el primer ordenador comercial, el Univac I, luego de que el doctor John von Neumann en 1946, elaboró el concepto de programa grabado, en el cual la secuencia de instrucciones del programa se proporciona a la memoria del ordenador.

En la década del 50, todos los aparatos electrónicos (radios, televisores y computadores) se construían en base a tubos de vacío. Los computadores de aquella época se conocen como de primera generación, como el IBM 650 y 704, que se ubicaban en grandes cuartos conteniendo muchos bastidores de equipo electrónico.

Al final de la década del 50 los transistores, inventados en 1948 por Bardeen, Brattain y Schockley en los laboratorios de la Bell Telephone, y otros elementos de estado sólido comenzaron a reemplazar a los tubos de vacío. Los computadores que utilizaron esta tecnología se conocen como de segunda generación. Ejemplo de ellos son el IBM 7090 y el Burroughs B5500.

En 1959 apareció el primer ordenador totalmente transistorizado, el NCR-GE 304, cuya unidad central contiene 8000 diodos y 4000 transistores.

En 1960, Kenneth Olsen, creador de la sociedad Digital Equipment, fabricó el PDP1; en 1963, el PDP5 a transistores y con memoria de núcleos de ferrita y en 1965, el PDP8.

En la década de los 60, varios componentes electrónicos discretos se combinaron en un solo componente electrónico y nació el circuito integrado ("chip") cuya patente fue inscrita por Texas Instruments en 1959, mientras Fairchild ponía a punto el procedimiento planar. Los computadores construidos en base a "chips" son los de tercera generación (IBM 360, GE 365, Burroughs B6700).

La tecnología de los circuitos integrados continuó avanzando y fue posible colocar todos los elementos discretos, necesarios para la construcción de un micro-procesador, dentro de un mismo "chip" (Intel 8008).

En el año 1971 se inicia la era de los micro-procesadores. Los construidos en ese año por Intel se destinaban a propósitos específicos (el 4004 en una calculadora, al igual que el PPS-4 de Rockwell y el PPS-25 de Fairchild, y el 8008 en un terminal de computadora) y se conocen como micro-procesadores de primera generación, que se caracteriza por micros en tecnología PMOS, encapsulados en paquetes de 16

patillas, trabajando por multiplexado. Una suma se realiza en 10us y exige numerosos circuitos periféricos, siendo el más típico el 4004 de Intel.

En 1972 apareció el disco flexible.

En 1973, la sociedad Francesa REE utiliza el 8008 para realizar el primer micro-ordenador comercializado, el Miclal.

En 1974 Intel ofreció el 8080, que fue el primer micro-procesador diseñado para poder ser utilizado en una gran variedad de aplicaciones y que inició el desarrollo de la segunda generación de micro-procesadores, caracterizada por estar realizada en MOS canal N y encapsulado "DIP" de 40 patillas. Existen buses especializados, mayor cantidad de funciones, incluyendo interrupciones y DMA, disminuyendo el número de circuitos integrados periféricos necesarios para el micro. Una suma se realiza en 2us. Son típicos el 8080 de Intel y el 6800 de Motorola.

En el mismo año, Motorola anunció el 6800. En 1975 Intersil desarrolló un micro de 12 bits, el IM 6100, con tecnología CMOS, y Western Digital elaboró un micro-procesador que sirvió para que Digital Equipment realice el LSI-11. Simultáneamente, Rockwell anunció el PPS-8 y Fairchild creó el F8, que dispone de su propia memoria integrada sobre el "chip" unidad central, con el fin de reducir el número de componentes

necesarios para construir un sistema.

En 1974, AEG Telefunken vendió los primeros micros europeos a Olympia, que los montó en sus calculadoras de mesa, y RCA anuncia el COSMAC, con tecnología CMOS.

En 1975 Texas comercializó el primer micro con tecnología I2L, mientras que Motorola anunció el 10800 con tecnología ECL.

Debido a la difusión que tomó el 8080, fue considerado como un standard y varias compañías constructoras empezaron a fabricar "chips" 8080, desarrollando algunas sus propias versiones (entre ellas Zilog).

Las características básicas del 8080 no fueron notablemente alteradas hasta 1978, año en el cual Intel produce el 8086 y al siguiente año una versión modificada del 8086, el 8088.

1.1.2 Características

El 8088 se popularizó con mayor facilidad debido a que reunía todas las ventajas del 8086 y además utilizaba, para comunicarse con la memoria y periféricos, datos de 8 bits al igual que el 8080, por lo tanto, podía ser directamente reemplazado en sistemas ya construidos, pero sus modificaciones eran tan importantes que fue considerado como la tercera generación de micro-procesadores.

El éxito de estos dos nuevos micro-procesadores se debe principalmente a las siguientes diferencias, respecto del 8080:

- Pueden direccionar hasta 1 MB de memoria, mientras que el 8080 sólo puede direccionar 64 KB. Debido al temprano éxito del 8080, éste comenzó a ser utilizado en sistemas cada vez más complejos, por lo tanto, los 64 KB de memoria rápidamente se volvieron insuficientes.

- En muchos de los procesos en los cuales se utilizó el 8080, la velocidad de respuesta se tornó en un limitante que no se superaba fácilmente debido a que, por trabajar con 8 bits, las rutinas de operaciones entre números mayores consumían mucho tiempo; los dos nuevos micro-procesadores operan con datos de 16 bits.

- Los nuevos micros proveen de instrucciones para multiplicación, división y operaciones con signo, instrucciones que no se disponían en el 8080.

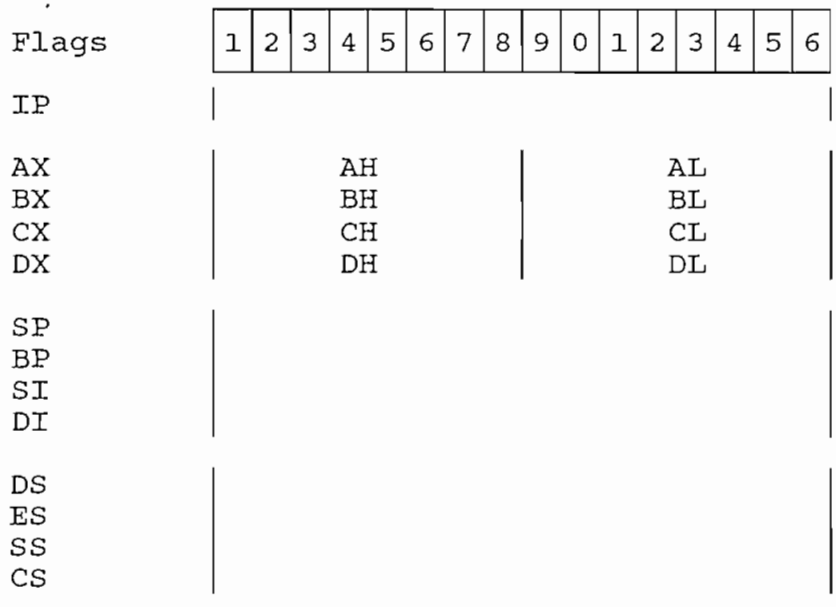
- El 8080 tenía dificultades para la creación de programas en lenguajes de alto nivel y se tenían problemas en el manejo de cadenas de texto, problemas que se superaron con los nuevos micros.

- En caso de un sistema que se vuelve cada vez más complejo, no puede esperarse que un solo micro atienda

todas las funciones requeridas, el 8080 no podía cooperar con otros micros; los nuevos fueron diseñados para operar en procesos que usen varios procesadores.

1.1.3 Registros

El 8088 contiene un total de 13 registros de 16 bits y nueve banderas de 1 bit. Con propósitos descriptivos, los registros se asocian en 4 grupos, 3 contienen 4 registros y el otro contiene un solo registro, el "instruction pointer", el cual no se puede acceder directamente por el programador.



De los tres grupos de registros mencionados, uno es el grupo de registros de acceso general, que se utilizan principalmente para almacenar operandos para las operaciones aritméticas y lógicas, otro el de registros de segmento, que almacenan la

dirección de inicio de los distintos segmentos de memoria y el último es el grupo de apuntadores e índices, que se utilizan para mantener direcciones de "offset" dentro de los segmentos de memoria.

Los registros de uso general del 8088 son de 16 bits y se conocen como AX, BX, CX y DX. Las mitades superior e inferior de cada registro pueden ser utilizadas en forma separada, como dos registros de 8 bits, o como un solo registro de 16 bits, debido a que cada mitad tiene su propio nombre.

Para la mayoría de los casos, el contenido de los registros generales puede ser intercambiado para operaciones aritméticas y lógicas. Existen, sin embargo, algunas instrucciones que dedican ciertos registros generales para usos específicos. Este uso especializado de los registros tiene la desventaja de ocasionar un mayor número de reglas especiales.

Los registros de apuntadores e índices se utilizan cuando se deben realizar frecuentes accesos a determinada localidad de memoria, caso en el cual las instrucciones no deben contener toda la dirección, sino el registro en el cual se encuentra la dirección, con lo que se obtiene un programa más corto y rápido.

Los registros de punteros son de 16 bits y generalmente contienen un "offset" de dirección dentro de un determinado segmento, cuya dirección de inicio está almacenada en un

registro de segmento.

Los registros de punteros permiten además que determinadas instrucciones accedan a localidades de memoria, cuyo "offset" es el resultado de operaciones calculadas mientras el programa está corriendo. Dichas operaciones pueden ser realizadas en los registros de uso general y luego es posible mover el resultado al registro puntero correspondiente. Eliminar estos movimientos acorta el programa; para el efecto, los valores contenidos en registros de índices pueden participar en operaciones lógicas y aritméticas junto con los registros generales de 16 bits.

Existen diferencias que obligan a dividir a este grupo de registros en dos partes, la de registros apuntadores (SP y BP) y la de registros índice (SI y DI). Los registros apuntadores se crearon para proveer acceso conveniente a datos en el segmento corriente de "stack". El uso del segmento de "stack" para almacenamiento de datos tiene ciertas ventajas. Los "offset" contenidos en los registros índice generalmente se refieren al segmento corriente de datos.

Existen algunas instrucciones que distinguen entre los dos registros apuntadores SP y BP. Las instrucciones PUSH y POP obtienen el "offset" para la ubicación del tope de la pila del registro SP, el registro BP no puede ser utilizado para el efecto, y se utiliza más bien para almacenar el "offset" de la base de un área de datos en el segmento de "stack".

Más aún, las instrucciones de cadenas de caracteres distinguen entre los dos registros índice SI y DI, ya que el operando de partida se obtiene de SI y DI contiene el "offset" para el operando de destino, función que no puede ser intercambiada para este tipo de instrucciones.

Los registros de segmento son de 16 bits CS, DS, SS y ES. Se utilizan para identificar los cuatro segmentos de memoria al momento direccionables. Cada registro identifica un segmento corriente en particular y su función no puede intercambiarse.

CS identifica el "current code segment", DS el "current data segment", SS el "current stack segment" y ES el "current extra segment". Todas las instrucciones leídas provienen del "current code segment", por lo tanto, necesitamos un registro que contenga el "offset" para la próxima instrucción a ser leída, para esto sirve el IP ("Instruction Pointer").

El segmento para los operandos generalmente se designa precediendo la instrucción con un prefijo de 1 byte, el cual especifica de cual de los cuatro segmentos debe ser traído el operando; en ausencia de dicho prefijo, los operandos se obtienen del "current data segment", excepto que la dirección de "offset" se haya calculado a partir del contenido de un registro apuntador, en cuyo caso se utiliza el "current stack segment" o el operando es el destino de una instrucción de cadenas, en cuyo caso se utiliza el "current extra segment".

1.1.4 Banderas

Hay nueve banderas estándar en total, seis se utilizan para indicar resultados de operaciones. Las banderas de estado varían generalmente luego de una operación aritmética o lógica, para indicar propiedades del resultado de tales operaciones. Dentro de estas banderas tenemos:

ZF si el resultado es cero o comparación de igualdad.

SF indica signo.

CF "carry flag".

AF "auxiliary carry flag" (correspondiente a los cuatro primeros bits y usada para simular operaciones decimales).

OF "overflow flag".

PF "parity flag".

y las tres restantes se utilizan para controlar operaciones del procesador:

DF "direction flag" que controla la dirección de las rotaciones (movimientos a derecha o izquierda).

IF "interruption flag", que habilita o deshabilita las interrupciones externas.

TF "trap flag" que pone al procesador en modo paso a paso para depuración de programas.

En el AT existen dos banderas especiales:

NT para tareas de anidamiento.
IOPL controla el nivel de prioridad de entrada/salida.

1.1.5 Segmentos de memoria

A pesar de que el 8086 y el 8088 puede direccionar 1MB, su aritmética está hecha para manipular datos de 16 bits, por lo tanto, las direcciones que directamente puede manipular son de 16 bits, esto hace que se requiera un mecanismo adicional para construir las direcciones.

Se divide el 1MB de la memoria en un número arbitrario de segmentos, cada segmento puede contener máximo 64KB y los últimos 4 bits de la dirección de inicio del segmento son ceros (la dirección es divisible para 16). En determinado momento, el programa puede acceder directamente a cuatro de dichos segmentos, conocidos como "current code segment", "current data segment", "current stack segment" y "current extra segment".

Se identifica cada uno de los segmentos corrientes colocando los 16 bits más significativos de la dirección en uno de 4 registros dedicados, llamados "segment registers". Los segmentos de memoria no deben ser únicos y pueden traslaparse, es decir, existe total libertad para la ubicación de los segmentos de memoria.

Nos referimos a una dirección específica de memoria definiendo el "offset" respecto del segmento correspondiente. El procesador construye los 20 bits requeridos para acceder a la dirección definida, sumando los 16 bits de "offset" con el contenido del registro de segmento al cual se le han añadido 4 ceros para los bits menos significativos.

Ejemplo de direccionamiento: un "segment register" tiene el valor 1018H, el primer dato que puede ser requerido está en la dirección 10180, el primer dato que no puede ser requerido está en la dirección 20180, el último dato que puede ser requerido está en 2017F, ya que 2^16 en hexadecimal corresponde a 10000H, siendo 16 el número de bits que puede manejar el micro-procesador.

Para el caso de los pórticos, el 8088 puede manejar 64K direcciones diferentes, debido a que no se ha considerado un "port segment register" que permita, con el mismo tratamiento descrito para la memoria, manejar un mayor número de puertos.

1.1.6 Operandos, instrucciones y modos de direccionamiento

Las instrucciones en el 8088 generalmente se realizan entre uno o dos operandos. Como ejemplo podemos mencionar la instrucción INC, la cual suma 1 al valor contenido en el operando y almacena el resultado en el mismo sitio ocupado

anteriormente por el operando, y la instrucción ADD, que suma el valor contenido en un operando con el de otro operando y almacena el resultado en el mismo sitio ocupado por uno de los operandos.

Con el fin de determinar los operandos de una instrucción, así como el destino de su resultado, su código define, entre las opciones posibles, todos estos parámetros, realizando la definición con el menor número de bits. Como resultado de lo indicado, existen instrucciones cuyo código ocupa solamente un byte y otras que ocupan hasta seis.

Entre los posibles modos existentes para direccionar un operando se tienen:

- Inmediato: Aquel en el cual, en la instrucción se define un valor que actuará como operando.

- Registro: Cuando uno de los operandos de una instrucción es un registro.

- Directo: Cuando la dirección de memoria de un operando se especifica directamente en la instrucción que se realizará. Se debe observar que este tipo de direccionamiento es un caso especial de direccionamiento indirecto, pero en el

cual el "offset" es cero.

- Indirecto: El direccionamiento indirecto define una dirección de memoria, en la cual se encuentra el operando de una instrucción utilizando para el efecto: un registro base, un registro índice, un registro base al que se le suma el contenido de un registro índice, un registro base o índice al que se le suma un valor de "offset" y un registro base al que se le suman el valor contenido en un registro índice y un "offset".

Al juego de instrucciones disponibles para el 8088 se lo puede clasificar, de acuerdo a su función en:

- Instrucciones para movimiento de datos, que se pueden dividir en tres categorías: aquellas que mueven datos entre registros o entre localidades de memoria y registros; instrucciones que mueven datos desde o hacia la pila y aquellas que permiten el movimiento de un bloque de memoria a otro.

- Instrucciones aritméticas, existiendo cinco tipos: suma, resta, multiplicación, división y comparaciones.

- Instrucciones lógicas, posibilitándose el uso de las

funciones AND, NOT, OR y XOR.

- Instrucciones de manejo de cadenas. Estas instrucciones usan un tipo de direccionamiento fijo; existen cinco funciones: mover uno o dos bytes entre localidades de memoria, cargar el registro AL o AX con datos contenidos en memoria, almacenar el registro AL o AX en memoria, comparar el registro AL o AX con datos contenidos en memoria y comparar los datos contenidos en una localidad de memoria con los contenidos en otra.

- Instrucciones para control del contador de programa, existiendo dos tipos principales, aquellas que se utilizan para llamar y retornar de una subrutina y las que ejecutan saltos incondicionales.

- Instrucciones de salto condicional, existen varias instrucciones que permiten realizar saltos basados en el resultado de una comparación lógica o aritmética.

- Instrucciones de lazo, combinan las instrucciones de decremento y de salto condicional.

- Instrucciones para control del procesador, son aquellas que permiten alterar el registro de banderas.

- Instrucciones de entrada/salida, permiten ingresar datos de uno de los puertos o a su vez sacar datos por los

pórticos. Si la dirección del puerto puede ser expresada con 8 bits, suele ser especificada directamente, si se necesitan 16 bits para definirla, se utiliza el registro DX.

- Instrucciones de interrupción, semejantes a las instrucciones de llamada a subrutina, las instrucciones de llamada a un programa de interrupción tienen la ventaja de que su código utiliza menos bytes. Este grupo se compone de tres instrucciones: petición de una interrupción "software", petición de una interrupción en caso de división para cero y retorno de interrupción.

- Instrucciones de rotación, son utilizadas para realizar multiplicaciones y divisiones en forma rápida o para determinar el estado de un bit.

1.1.7 Diseño de sistemas basados en el 8088

El micro-procesador 8088 no dispone de memoria ni periféricos de entrada y/o salida integrados, por lo tanto, necesita de estos elementos para cualquier uso práctico; para comunicarse con los citados elementos, dispone de un bus de direcciones de 20 bits, el bus de datos comparte los bits menos significativos con el de direcciones, haciendo indispensable el uso de "latches".

Como integrantes de la familia del 8088 existen disponibles varios circuitos integrados de uso específico, que facilitan el desarrollo de sistemas en base al indicado micro-procesador; a continuación se mencionan los más importantes.

En la familia 8088 se incluye el circuito integrado 8282, un "latch" de 8 bits que dispone de entradas "strobe" (que se conecta al ALE del 8088) y "-output enable" para su control.

Para amplificar el bus de datos, con el fin de poder alimentar a un mayor número de componentes, se utiliza el 8286, que dispone de dos entradas de control, "-output enable" (que se conecta al DEN "data enable" del 8088) para indicar cuando se requiere el paso de un byte y "transmit" (que se conecta al DT/-R "data transmit/-receive" del 8088) para indicar la dirección de la transferencia.

Para generar la señal de reloj necesaria, se utiliza el 8284A.

Para dotar al equipo de puertos, el 8088 dispone de una señal IO/-M en la que se define si la operación de entrada y/o salida se refiere a un puerto de entrada/salida o a la memoria, ya que los buses de direcciones y datos se comparten también por los puertos.

Para servir a las interrupciones, el 8088 tiene dos entradas INTR ("interrupt request") y NMI ("non-maskable interrupt") y una salida INTA ("interrupt acknowledge"). Cuando el 8088

reconoce una interrupción, espera que el periférico que la provocó coloque en el bus de datos un byte que le indica donde buscar la dirección de la subrutina de atención a la interrupción. Estas tres patillas disponibles en el 8088 no son suficientes para servir a varios periféricos, por lo tanto, se hace necesario el uso del 8259A ("interrupt controller") el cual provee al sistema de 8 señales de interrupción.

Si son necesarias más de ocho señales de interrupción, se puede colocar un 8259A maestro y varios 8259A esclavos, conectados en cascada. El 8259A es programado por el 8088 con el fin de determinar el programa que se debe correr ante la llamada de los diferentes periféricos a él conectados.

El 8088 dispone además de una entrada de control MN/-MX ("minimum/-maximum"); para usarlo en modo máximo, es necesario conectar un 8288 ("bus controller") el cual le provee al 8088 de un juego adicional de pines. Del 8288 se obtienen señales separadas para MRDC ("memory read command"), MWTC ("memory write command"), IORC ("input output read command"), IOWC ("input output write command"), ALE ("address latch enable") lo que permite liberar algunos pines del 8088 para otros usos, como por ejemplo el trabajo en conjunto con otros micro-procesadores.

Las memorias, en forma general, disponen de 3 entradas de control, "-chip select" (que se conecta a un decodificador de

direcciones que se alimenta del bus de direcciones), "-write enable" (que se conecta a -WR del 8088) y "-Output Disable" (que se conecta a -RD del 8088) utilizada para lectura de datos desde la memoria.

1.2 LOS COMPUTADORES PERSONALES XT COMPATIBLES

Existen 5 elementos claves en un ordenador: el procesador y sus elementos de soporte; la memoria, existiendo dos tipos: memoria de lectura solamente y memoria de lectura y escritura; los elementos integrados de entrada/salida (el almacenamiento en discos, entrada por teclado, salida por monitor, etc); los canales para puertos adicionales y los programas, conocidos como soporte "software".

1.2.1 Historia

En agosto de 1981, IBM lanzó el PC original, luego de 1 año de tomada la decisión de crear un ordenador personal. Utilizó el procesador 8088, el cual internamente es un procesador de 16 bits igual al 8086, pero para comunicación externa lo hace con palabras de 8 bits, con el fin de usar periféricos de 8 bits que en la época eran muy baratos y abundantes. El PC disponía de dos unidades para disco flexible y no disponía de disco duro.

En el otoño de 1982, se anunció el primer Compaq, compatible con el PC de IBM.

En la primavera de 1983, se anunció el modelo XT, que contaba con una unidad de disco flexible de 360 KB y un disco duro de 10 MB.

En otoño de 1983, Compaq sacó la versión portátil, a la que denominó Compaq plus.

En 1983 nació el PCjr, el cual disponía de una unidad para cartuchos de memoria ROM, proyectada principalmente para el uso en juegos. Murió en 1985.

En 1984 nació el Compaq DeskPro con un disco duro de 20 MB y procesador 8086 que utiliza 16 bits para su comunicación con los periféricos externos, reduciendo el tiempo necesario para realizar las transferencias.

En el mismo año nació el IBM AT con un disco duro de 20 MB, 3MB RAM y que utiliza el procesador 80286 que integra tres ventajas: puede trabajar hasta con 16 MB de memoria y con memoria virtual; dispone de "hardware" multitarea e integra además "chips" de apoyo dentro del procesador.

En 1985 se lanza el DeskPro 286 con 30 MB en disco duro y 3 veces más rápido que el AT. Se incluye la opción turbo, que permite variar su frecuencia de operación, con el fin de que sea compatible con el AT en velocidad.

1.2.2 Chips utilizados en un XT

El corazón del XT es el procesador Intel 8088 operando en modo máximo, lo que le permite la conexión con el coprocesador

8087. El 8088 se apoya, principalmente, en los siguientes circuitos integrados de su familia:

8088	Micro-procesador
8284	Generador de reloj (88284 para el AT)
8253	Temporizador programable
8288	Controlador de bus (82288 para el AT)
8237	DMA
8259	Supervisor de las interrupciones
8255	Interfase periférico programable
6845	Controlador CRT

El uso de estos circuitos integrados permite al procesador realizar una gran cantidad de funciones, entre ellas, proveer 20 bits de direcciones que pueden ser utilizados por los 4 canales de acceso directo a memoria (DMA), de los cuales 3 están disponibles en el bus de entradas/salidas, mientras que el cuarto se ha programado para refrescar las memorias dinámicas del sistema.

Para posibilitar esta última función, se programa uno de los canales del temporizador programable para que, periódicamente, se realice una transferencia ficticia DMA, creando un ciclo de lectura de la memoria, que se utiliza para refrescarla, tanto en la tarjeta del sistema como en las ubicadas en las ranuras de expansión.

De los tres canales del temporizador programable, el canal 0

se utiliza como un temporizador de propósito general que provee una base de tiempo constante para actualizar el reloj interno; el canal 1 se utiliza para efectuar los ciclos de refresco del canal DMA y el canal 2 se utiliza para generar los sonidos en el parlante interno del computador.

De los 8 niveles de interrupción (provistos por el 8259) seis están disponibles en las tarjetas de extensión; de los otros dos, el nivel 0 (de más alta prioridad) está conectado al canal 0 del temporizador programable y provee una interrupción periódica para actualizar el reloj, mientras que el nivel 1 está conectado al adaptador de teclado y recibe una interrupción para cada código enviado por el teclad (* a señal NMI del 8088 se usa para reportar errores de paridad en la memoria.

En la tarjeta principal del computador se colocan además las memorias EPROM y RAM, cuya descripción y uso se realiza en el acápite 1.2.5.

1.2.3 Interrupciones "hardware"

De las 8 señales de interrupción disponibles en los computadores personales XT compatibles, y que son generadas por el 8259, solo 6 están disponibles en las ranuras para las tarjetas de extensión. Las señales se utilizan para:

IRQ0	Reloj en tiempo real
IRQ1	Teclado
IRQ2	PC-Net
IRQ3	COM2
IRQ4	COM1
IRQ5	Disco duro
IRQ6	Diskette
IRQ7	Impresora (manejo del puerto IEEE-488)

Cada vez que pulsamos una tecla o la liberamos, se produce una señal de interrupción; 18,2 veces por segundo se produce la interrupción de mantenimiento de la hora.

1.2.4 Interrupciones "software"

Existen dos clases de interrupciones "software": interrupción de excepción (cuando dividimos para cero, encontrar instrucciones sin sentido, etc.), y la interrupción "software" generada intencionalmente por nuestros programas, que permite solicitar los servicios BIOS.

Analizadas desde otro punto de vista, existen 6 clases de interrupciones:

- "Hardware" Intel, entre las cuales está la división por cero y fallo en la alimentación.

- Interrupciones "hardware" del PC, como: falta de papel en la impresora u operación del disco terminada.

- Interrupciones "software" del PC, para activar partes de los programas de la ROM-BIOS, como por ejemplo visualizar en pantalla.

- Interrupciones "software" del DOS.

- Interrupciones "software" de aplicaciones.

- Interrupciones de tabla, direccionadas por la tabla de vectores, en las cuales existen algunas que tienen número de interrupción pero no se pueden utilizar nunca, ya que no hay rutina de control de interrupción para ellas.

1.2.5 La memoria del PC

El computador personal XT, trabajando con 16 bits para las operaciones internas, utiliza 20 bits para definir una dirección de memoria, lo que le permite trabajar hasta con 1MB. Para esto, se basa en una dirección de 16 bits, a la cual se le añade un cero al final y se le suma el desplazamiento, como ya se explicó en los segmentos de memoria del 8088.

Para recordar lo dicho, se incluye un ejemplo: segmento ABCD y desplazamiento 1234 indica que la dirección de memoria a la

que nos referimos es ABCD0 + 1234 = ACF04.

En la parte más baja, los primeros 1024 bytes, se encuentra la tabla de vectores de interrupción. Esta tabla permite direccionar 256 interrupciones distintas, ya que cada vector utiliza 4 bytes. La tabla de vectores de interrupción puede ser inspeccionada usando DEBUG D 0.0, que nos muestra 128 bytes equivalentes a 32 vectores.

Si en la tabla observamos: E8 4E 9A 01, indica que la subrutina de interrupción se encuentra en la dirección 019A:4EE8. Para la tabla de vectores se ocupan las direcciones entre 0H y 400H.

Luego viene el área de datos ROM-BIOS, sitio en el cual, entre otras cosas, existe un "buffer" para las últimas teclas presionadas, información de la capacidad de memoria del computador, opciones instaladas, tipo de pantalla existente, etc. Se reservan 256 bytes para esto, en las direcciones entre 400H y 500H.

De 500H a 600H está el área de trabajo de DOS y BASIC.

Si dividimos el 1MB de memoria en bloques de 64KB, el uso que en el computador se da a cada bloque se ilustra en la siguiente tabla:

Bloque 0 64K Memoria de usuario ordinaria

Bloque 1	64K	Memoria de usuario ordinaria
Bloque 2	64K	Memoria de usuario ordinaria
Bloque 3	64K	Memoria de usuario ordinaria
Bloque 4	64K	Memoria de usuario ordinaria
Bloque 5	64K	Memoria de usuario ordinaria
Bloque 6	64K	Memoria de usuario ordinaria
Bloque 7	64K	Memoria de usuario ordinaria
Bloque 8	64K	Memoria de usuario ordinaria
Bloque 9	64K	Memoria de usuario ordinaria
Bloque A	64K	Memoria de video ampliada
Bloque B	64K	Memoria de video estándar
Bloque C	64K	Memoria de expansión de la ROM
Bloque D	64K	Memoria de otros usos (cartuchos)
Bloque E	64K	Memoria de otros usos
Bloque F	64K	Memoria de ROM-BIOS y ROM-BASIC

La primera parte del ROM-BIOS es el POST ("power on self test") o comprobación de arranque, y la segunda son los servicios básicos de entrada/salida ("basic input/output system") o BIOS.

La memoria extendida es aquella que va más allá de 1M en el AT, mientras que la memoria expandida es una memoria de banco conmutable cuya conmutación se regula por las normas LIM (Lotus, Intel, Microsoft) para programas de EMM ("Expanded Memory Manager").

1.2.6 Cadenas de texto

Los datos de texto se forman con caracteres individuales que ocupan un byte cada uno, de acuerdo a la tabla de caracteres ASCII. Con el fin de definir cadenas de caracteres (unidad combinada de grupos de octetos colocados uno tras otro) las cuales no tienen una longitud definida, se debe determinar su longitud y donde termina la cadena.

Para esto se utilizan dos métodos: el primero graba un número de un byte al comienzo, que define la longitud de la cadena. Con este método se limita el número máximo de caracteres en 255 (éste es el método utilizado por el editor de BASIC). Otra forma es definir el final de la cadena con un carácter delimitador el cual no se considera parte de la cadena.

Los caracteres más utilizados como delimitadores son el 0 o nulo, en el cual todos los bits son 0 y el otro que se suele utilizar es el carácter 13, que entre los caracteres de control se define como retorno de carro.

1.2.6.1 El juego de caracteres ASCII.- Con el fin de regular el significado que se da a cada una de las posibles combinaciones de bits que forman una palabra, se creó el código americano estándar para intercambio de información (ASCII). Dicho código ha realizado una tabla en la cual se define el significado para las primeras 126 combinaciones,

pero, debido a que se ha generalizado el uso de una tabla completa que considera las 255 combinaciones posibles, se ha llamado a esta última: tabla de caracteres ASCII ampliada.

El juego de caracteres ASCII ampliado se organiza de la siguiente forma:

Los primeros 32 caracteres, desde el 0 hasta el 31, son los caracteres de control.

A continuación se incluye una tabla en la cual se indica el significado de cada uno de ellos y, si existe, la tecla que permite realizar directamente su función.

CARACTERES DE CONTROL

CODIGO	TECLA	NOMBRE	DESCRIPCION	TECLA
0	^@	NUL	nulo	
1	^A	SOH	Inicio de cabecera	
2	^B	STX	Inicio de texto	
3	^C	ETX	Fin de texto	
4	^D	EOT	Fin de transmisión	
5	^E	ENQ	Requerimiento	
6	^F	ACK	Reconocimiento	
7	^G	BEL	Campana	
8	^H	BS	Espacio atrás	<--
9	^I	HT	Tabulación horizontal	TAB
10	^J	LF	Avance de línea	

11	^K	VT	Tabulación vertical	
12	^L	FF	Página nueva	
13	^M	CR	Retorno de carro	ENTER
14	^N	SO	Shift desactivado	
15	^O	SI	Shift activado	
16	^P	DEL	Borrar	
17	^Q	DC1	Control dispositivo 1	
18	^R	DC2	Control dispositivo 2	
19	^S	DC3	Control dispositivo 3	
20	^T	DC4	Control dispositivo 4	
21	^U	NAK	Reconocimiento negativo	
22	^V	SYN	Sincronización	
23	^W	ETB	Fin de bloque de texto	
24	^X	CAN	Cancelar	
25	^Y	EM	Fin de medio	
26	^Z	SUB	Sustituir	
27	^[	ESC	Escape	ESC
28	^/	ES	Separador de archivo	
29	^]	GS	Separador de grupo	
30	^^	RS	Separador de registro	
31	^_	US	Separador de unidad	

^NumLock equivale a PAUSE (Detiene el ordenador)

^ScrollLock equivale a BREAK

^S pide al programa con que estamos trabajando que pare

Los restantes códigos ASCII se ordenan en la tabla de la siguiente forma:

Desde el código 32, que representa al espacio en blanco, hasta el 47 son signos (!"#$%&'()*+,-./), siendo interesantes el 44 que representa a la coma y el 46 al punto.

Desde el código 48 al 57 se utilizan para representar los números, correspondiendo el 48 al 0.

Desde el código 58 al 64 son signos (:;<=>?@).

Con el código 65 se representa a la A (mayúscula). Los siguientes códigos representan a las restantes letras mayúsculas en orden alfabético, correspondiendo el código 90 para la Z.

Desde el código 91 al 96 se utilizan para representar signos ([\]^_`).

Las letras minúsculas, en orden alfabético, son representadas por los códigos entre el 97, que corresponde a la a, y el 122 que corresponde a la z.

En el juego ampliado, los caracteres 0 y 255 son nulos.

1.2.7 Discos

En un disco, la información se almacena en el siguiente orden: "boot", "fat", "root directory" y datos.

Existen "fat" ("file allocation table") de 12 y 16 bits. En el directorio se indica el primer "cluster" donde comienza un archivo, y en el "fat" se indica el siguiente o el final. Si se tiene 0000 indica "cluster" no utilizado, FFFF indica final de archivo y FFF7 indica sector malo. El primer byte del "fat" indica el tipo de formato del disco, ya que sólo se puede direccionar desde el sector 2.

En el área del directorio, los dos primeros bytes, si son 00 indican que no se ha utilizado, E5 indica archivo borrado. El tamaño del archivo se indica como un entero de 4 bytes. La fecha y hora se graban por separado en dos palabras adyacentes de 16 bits, con las fórmulas:

FECHA=DIA+32*MES+512*(AÑO-1980)
HORA =SEGUNDOS/2+32*MINUTOS+1024*HORAS

Los atributos del archivo son ocho banderas, de las cuales se usan 6: nombre de volumen, subdirectorio, sólo lectura, archivo cambiado, archivo oculto, archivo de sistema. Los ocultos y del sistema no tienen diferencia esencial. Se debe indicar que los archivos ocultos, si bien no son mostrados por un comando DIR, no están ocultos para algunos comandos, entre ellos el comando TYPE.

Para evitar copias se debe saber que un sector de tamaño raro, un sector mal numerado, un sector desaparecido, un sector extra, etc. no posibilitan el copiado.

1.2.8 Video

Para ver el modo de video que estamos utilizando, en BASIC podemos usar la instrucción: DEFSEG=0:PRINT PEEK(&H449) o con DEBUG podemos usar D 0:449L 1.

El monitor, en modo texto, requiere para cada posición su dato y su atributo (si el carácter aparece parpadeante, en color, etc). Estos datos se almacenan en la memoria de pantalla y su dirección relativa se define con (FILA*80+COLUMNA)*2 si numeramos las filas y columnas empezando con 0.

Cada carácter se escribe en una caja de 9 por 14. Se deja la primera y última columnas para el espacio entre caracteres y las dos filas de arriba y una de abajo para el espacio entre líneas. De ellas 2 filas se usan para el rabo de la pqyjg quedándonos 7 por 9 para la parte principal de los caracteres.

En BASIC el comando SCREEN tiene 4 parámetros, el tercero "apage" indica la página activa y "vpage" indica la página visible. En lenguaje de máquina es muy rápido borrar una pantalla, llenando con 2007 en el área de memoria de pantalla (20H es el carácter 32 espacio en blanco y 07 es el atributo de normal).

Para sacar textos en modo gráfico, la CGA ("color graphics adapter") en la dirección F000:FA6E tiene una tabla basada en

la caja de 8 por 8.

1.2.9 Teclado

El teclado tiene asignada la interrupción número 9 y la subrutina de interrupción guarda el código de la tecla aplastada y de su liberación (el mismo código de la tecla + 80H). Para las teclas SHIFT y las de conmutación, se debe además mantener el registro del estado actual, lo cual se almacena en las direcciones 417H y 418H. Es necesario además comprobar combinaciones especiales como CTRL-ALT-DEL, CTRL-NumLock, etc.

La ROM-BIOS define un "buffer" para almacenar 15 caracteres. Cada carácter se almacena con 2 bytes; si el primero no es cero, contiene el propio carácter ASCII y el segundo el código de identificación de la tecla presionada; para caracteres especiales el primer byte es 0 y el segundo el código del carácter especial presionado.

1.2.10 Parlante

El parlante está controlado por los dos bits menos significativos del puerto 97 (61H). Para generar sonido ponemos 1 en dichos 2 bits y para apagarlo ponemos 0. El bit más significativo (de valor 2) enciende y apaga el altavoz y

el otro determina si el parlante será alimentado por una señal del reloj programable interno del computador personal.

Para producir un sonido regular se debe programar el 8253 (sound 500,1) y luego activar los bits del puerto 97 (out 97,(inp(97)+3). Es posible generar sonidos también encendiendo y apagando el parlante (colocando 1 y 0 solamente en el bit de valor 2).

1.2.11 ROM-BIOS

La ROM-BIOS está dividida en 3 partes:

- En la primera están el POST y las rutinas de inicialización (crear vectores de interrupción, preparación del equipo, inicializar registros, carga de parámetros, etc.), luego están las ampliaciones al BIOS que algunos equipos opcionales necesitan (y por lo tanto se requiere inicializar), las cuales ocupan los bloques C, D y E de la memoria, cuyos 2 primeros bytes son 55AA. No tienen que estar en conflicto con otra ampliación y tienen que empezar en un múltiplo de 2K. La última parte son las rutinas de arranque.

- Las otras dos partes de la ROM-BIOS son el manejo de las interrupciones del "hardware" y la ejecución de servicios.

Para manejar las necesidades del "hardware" como pulsar una tecla, manejar el reloj interno y los discos, etc. está la sección de manejo de interrupciones del "hardware".

1.2.11.1 Interrupciones de servicio de la ROM-BIOS.- Para pedir un servicio de interrupción de la ROM-BIOS, se carga su número en uno de los registros del micro-procesador (AH) y se activa la interrupción necesaria, usando la instrucción INT.

Los principales servicios de interrupción disponibles se listan a continuación (mayor información, al respecto, se puede encontrar en la referencia [5]):

Interrupción	Grupo de servicios
5	Impresión de pantalla
16	Servicios de video
17	Listado del equipo
18	Tamaño de la memoria
19	Servicios del disco
20	Servicios del puerto serie
21	Servicios del puerto de cassette
22	Servicios del teclado
23	Servicios del puerto paralelo
24	ROM-BASIC
25	Arranque
26	Servicios de reloj

Video

Servicio 0	para cambiar el modo de video.
Servicio 1	para controlar el tamaño y forma del cursor.
Servicio 2	fija la posición del cursor en la pantalla.
Servicio 3	informa donde está el cursor y su forma.
Servicio 4	informa si el lápiz óptico está accionado y en que lugar está tocando la pantalla en términos de la cuadrícula o la posición del "pixel" de gráficos.
Servicio 5	selecciona página activa
Servicio 6 y 7	"scroll" de ventana, permiten definir una ventana rectangular y mueven los datos dentro de la ventana de arriba abajo (6) o de abajo arriba (7).
Servicio 8	lee el carácter en curso y su atributo desde la pantalla; puede decodificar el dibujo de un carácter en modo gráfico a su código de carácter.
Servicio 9	escribe un carácter en pantalla con el atributo especificado.
Servicio 10	escribe un carácter con el atributo de pantalla que ya esté en esa posición de la pantalla.
Servicio 11	activa la paleta que se va a utilizar.
Servicio 12	escribe un único punto en la pantalla.
Servicio 13	lee un punto desde la pantalla.
Servicio 14	escribe un carácter en la pantalla y avanza el cursor a la siguiente posición (teletipo de

escritura).

Servicio 15 lee el modo de video en curso.

Existe además el servicio de impresión de pantalla, que se inicia con el servicio 3 para saber la posición del cursor y el 15 para saber las dimensiones de la pantalla. Almacena la posición del cursor y lo mueve de arriba abajo solicitando el 8 y un servicio para impresión para cada posición, luego restaura el cursor a su posición original y devuelve el control al programa que se está corriendo. Este servicio puede ser solicitado desde cualquier programa realizando una instrucción de interrupción INT 5.

Los servicios de disco, con la INT 19, son básicamente seis:

0 inicializa la unidad de disco y su controlador.

1 estado de la unidad de disco.

2 lee sectores del disco en la memoria siempre que estén todos ellos en la misma pista.

3 escribe sectores (inverso al 2).

4 verifica los datos escritos en un disco (corresponde a VERIFY ON) comprobando errores de paridad, etc.

5 formatea una pista de un disco.

Los servicios del pórtico serial INT 20 son 4:

0 inicializar el puerto fijando los parámetros básicos.

1 enviar un byte al puerto.

2 leer un byte

3 información del estado, que indica cosas tales como si el

dato está preparado.

Interfase de cassette INT 21

0 enciende el motor

1 apaga el motor

2 lee 256 bytes

3 escribe 256 bytes

Teclädo INT 22

0 lee el siguiente carácter del "buffer" de entrada del teclado, los caracteres se presentan en su formato de 2 bytes.

1 informa si hay preparada alguna entrada al teclado, si hay informa sobre el carácter comunicando los octetos, pero el carácter se queda en el "buffer" hasta ser leído con el 0.

2 comunica los bits de estado del teclado.

Puerto paralelo INT 23

0 manda un byte a la impresora.

1 inicializa la impresora.

2 comunica el estado de la impresora.

El parlante y los mandos ("joystick") no tienen apoyo en el BIOS.

INT 17 informa la configuración del PC, nos dice cuantas unidades de disco tiene (de 0 a 4), el número de puertos

paralelos y serie, si hay adaptador de juegos, pero no dice nada de los discos duros ni de los lápices ópticos.

INT 18 indica la cantidad de memoria en KB.

INT 26 hora del día

0 para leer el número existente en el contador de reloj.

1 para activar el reloj.

El reloj oscila generando una interrupción 18,2 veces por segundo y la rutina de tratamiento suma uno cada vez al contador del reloj.

Existen además dos interrupciones para pasar control a una de las dos rutinas especiales internas del BIOS, la ROM-BASIC y las rutinas de arranque. Es posible activar estas rutinas simplemente con solicitar estas interrupciones.

1.2.12 El Sistema Operativo

El sistema operativo maneja los dispositivos, controla los programas y procesa los comandos.

El procesador de comandos puede ejecutar cuatro categorías: comandos internos y 3 tipos de comandos externos. Entre los comandos externos se tiene: COM, EXE y BAT.

Comandos internos del DOS son: CLS, COPY, DATE, DEL/ERASE, DIR, REN/RENAME, TIME, TYPE, BREAK, CD/CHDIR, ECHO, MD/MKDIR, PATH, PROMPT, RD/RMDIR, SET, VER, VERIFY, VOL.

COM son archivos imagen, solo se copian en memoria y se arranca el programa. No pueden ser mayores que 64KB.

Para correr un archivo EXE es necesario preparar el programa; entre otras cosas, es necesario poner en algunos sitios del programa la dirección de memoria donde el programa se ha cargado en tantas partes del programa como se necesite. Para hacerlo, el formato de archivo EXE incluye una tabla de las partes del programa que se necesita modificar y como deben modificarse.

Las partes esenciales del DOS están permanentemente contenidas en posiciones de memoria baja, la mayor parte del intérprete de comandos se guarda en posiciones de la parte alta de la memoria.

Cuando un programa termina y devuelve el control al DOS, éste comprueba si el intérprete de comandos se ha perturbado, en caso afirmativo el DOS carga una copia nueva desde el disco. Por lo tanto, en algunos programas se necesita tener una copia del COMMAND.COM en el diskette de trabajo.

Los archivos BAT posibilitan ejecución de comandos internos o externos, paso de parámetros a los comandos y ejecución de

acciones lógicas dependientes de: el tipo de error que se produce, los parámetros incluidos o la existencia de archivos necesarios.

El sistema operativo, el momento de arrancar la máquina, define los vectores que señalan la dirección en la cual se inician los programas de atención a las interrupciones "software". Las principales interrupciones disponibles son las siguientes:

Interrupción		Descripción
Hex	Dec	
20H	32	Fin de programa
21H	33	Requerimiento de rutina de servicio
22H	34	Dirección de fin de programa
23H	35	SHIFT-BRK (CTRL C)
24H	36	Dirección para "Fatal Error"
25H	37	Lectura de disco
26H	38	Escritura en disco
27H	39	TSR (Termine y permanezca residente)

Las rutinas de servicio del DOS se solicitan mediante la interrupción 21H (33) que es común para todas. Para definir el servicio requerido se carga su número de identificación en el registro AH del micro, tal como se realiza para solicitar los servicios de la ROM-BIOS.

Las principales rutinas de servicio del DOS se listan en el Anexo No. 3.

Hay dos servicios de permanencia como residente, que son utilizados por los programas TSR ("Terminate and Stay Resident").

Además de todos los servicios de la ROM-BIOS, ya que el DOS duplica dichos servicios para evitar dependencias de los programas respecto del tipo de máquina utilizada, mediante el DOS se pueden acceder a servicios de disco de alto nivel, entre los cuales se diferencia:

Los "servicios de archivo tradicionales" que están basados en el uso del Bloque de control de archivo (FCB o "file control block") utilizado para proporcionar el nombre y la identificación de los archivos con los que se trabajará. Entre otras cosas, estos servicios pueden localizar archivos (con * y ?), abrir un archivo para lectura o escritura, cerrarlo, realizar lectura o escritura secuencial de principio a fin y lectura o escritura aleatoria.

El programador debe mantener un área de datos en la memoria, conteniendo la información del archivo. El FCB contiene información particular que describe al archivo. El sistema usa dicha información para el acceso de I/O al archivo. El PSP, "Program Segment Prefix", tiene espacio para dos FCBs, uno con offset 5CH y el otro con 6CH. Los FCBs y FCBs

extendidos pueden estar abiertos o no abiertos. Un FCB no abierto contiene especificador de drive y nombre de un archivo, en el nombre se pueden incluir caracteres comodín (el * y ?, conocidos como "wildcard"). Cuando la llamada al sistema de abrir un archivo (función 0Fh) llena todos los campos del FCB, se dice que el FCB se ha abierto.

Todo acceso usando el FCB se realiza a través de un buffer de datos conocido como el DTA, "disk transfer address". Cuando se da el control a un programa de usuario, el DTA se llena con offset 80H en el PSP. Este DTA tiene suficiente espacio para transferir 128 bytes. El programa puede mover el DTA a cualquier sitio, usando la función 1AH.

Campos del FCB:

Nombre	Tamaño en bytes	Offset Value	Descripción
Drive number	1	00H	1 especifica drive A, 2 drive B. Cuando el FCB se usa para crear o abrir un archivo, 0 especifica el drive por omisión ("default"). Esto significa que la función 0FH (llamada al sistema para abrir un archivo) carga este campo con el número del drive por omisión.

Filename	8	01-08H	cadena de caracteres (ocho caracteres), llenada con espacios en blanco, si es necesario, y justificada a izquierda. En este campo se pueden poner nombres como LPT1 o PRN para proveer entrada/ salida con dispositivos.
Extension	3	09-0BH	cadena de 3 caracteres justificada a izquierda, llenada con espacios en blanco si es necesario. Si todo el campo está lleno de espacios en blanco, no existe extensión.
Current Block	2	0CH-0DH	apunta al bloque (128 records) que contiene el registro corriente ("current record"). El llamado a la función del sistema para apertura de archivo (0FH) pone este campo a cero. Los campos de "Current Block" y "Current Record", conjuntamente, definen el "record pointer".
Record Size	2	0EH-0FH	Este campo contiene el tamaño

			en bytes de un registro lógico. La función 0FH pone este campo en 128, se debe llenar este campo antes de que el archivo sea abierto.
File size	4	10-13H	Contiene el tamaño en bytes del archivo. La primera palabra contiene la parte menos significativa del tamaño.
Fecha	2	14-15H	Fecha de la última escritura contiene la fecha del cambio más reciente. Se mapean como se ilustra:

OFFSET 15H

Y	Y	Y	Y	Y	Y	Y	M
15						9	8

OFFSET 14H

M	M	M	D	D	D	D	D
		5	4				0

El año se representa como el numero del año a partir de 1980.

Hora	2	16-17H	Hora de la última escritura contiene la hora del cambio más

			reciente. Los segundos aparecen en incrementos de dos segundos, se mapea como se ilustra: OFFSET 17H H H H H H M M M 15 11 10 M M M S S S S S 5 4 0
Reserved	8	18-1FH	El DOS se reserva el uso de estos campos.
Current record	1	20H	Apunta a uno de los 128 registros del bloque corriente. Se debe llenar el campo de "current record" antes de hacer la lectura o escritura secuencial al archivo, ya que la llamada al sistema para abrir archivo no inicializa este campo. Los campos de "current record" y "current block" conjuntamente definen el "record pointer".
Relative record	4	21-24H	Cuenta desde el inicio del

archivo, comenzando desde cero, y apunta al registro seleccionado actualmente. Se debe llenar el campo "relative record" antes de hacer una lectura o escritura secuencial al archivo, ya que la llamada al sistema para apertura de archivo no inicializa este campo. Un tamaño de registro menor que 64 bytes usa las dos palabras de este campo, un tamaño de más de 64 bytes usa solo los primeros 3 bytes.

En el FCB, con offset de 5CH desde el PSP (Program Segment Prefix), el último byte del "relative record field" está en el primer byte del "unformatted parameter area", que comienza en "offset" 80H. Este es el DTA por omisión.

FCB extendido:

Añadiendo un prefijo de 7 bytes al FCB standard se produce un FCB extendido. El FCB extendido puede crear o buscar el directorio por archivos con atributos especiales. La lista de los 7 bytes del prefijo se adjunta:

Nombre	Size	Value	Offset Value(decimal)
Flag Byte	1	FFH	-7
Reserved	5	-	-6
Attribute Byte	1	-	-1
read only		01H	
hidden file		02H	
system file		04H	
volume ID		08H	
directory		10H	
archive		20H	

Para operar archivos existe una alternativa al uso del FCB; es el uso del "handle" que es un número de 2 bytes que identifica en forma unívoca a cada archivo que se está utilizando por un programa, con lo cual la información de control del archivo se mantiene a salvo.

Un "handle" es un valor binario de 16 bits usado por el DOS para tener acceso a un archivo o dispositivo. El nombre del archivo o dispositivo es especificado (como cadena de caracteres ascii terminada por un byte de ceros y que puede contener "drive", "path" y nombre) al DOS durante la apertura, y el DOS regresa el "handle", que se utilizará para los siguientes accesos.

Este método no requiere un DTA. El usuario especifica la ubicación de los datos que serán transferidos de o hacia el

archivo, para los requerimientos de funciones que utilizan el "handle".

DOS pre-especifica 5 "handles" que están abiertos cuando se da el control a un programa de usuario, que son:

Nombre	Handle	Dispositivo por omisión
Standard input	0000H	CON
Standard output	0001H	CON
Standard error	0002H	CON
Standard auxiliary	0003H	AUX
Standard printer	0004H	PRN

Los dispositivos PRN (LST y LPT1), COM1 (AUX), COM2, LPT2, y LPT3 deben ser pre-definidos para procesar una cadena de control de 2 bytes llamada "device configuration word" (DCW). Esta palabra es pasada por, o leída de, un "handler" de dispositivo, usando la petición de función 44H.

La DCW tiene el siguiente formato:

1 2 0 3 3 0 4 4 0 5 5 5 6 0 7 7

código	definición
0	Reservada
1	tipo de puerto (0=paralelo, 1=serial)
2	Número de bits por carácter (0=7, 1=8)
3	Número de puerto (00=1, 01=2, 10=3, 11=4)

4	Tipo de verificación de ocupado (00=none, 01=SCF, 10=DSR, 11=Xon/Xoff)
5	Baud rate (000=110, 001=150, 010=300, 011=600, 100=1200, 101=2400, 110=4800, 111=9600)
6	Número de bits de parada (0=1, 1=2)
7	Tipo de chequeo de paridad (00=none, 01=odd, 10=none, 11=even)

1.3 ENLACE DE LA TARJETA DAS-128 CON EL PC

1.3.1 El conector para tarjetas de extensión

El canal de entradas/salidas, al que se conectan las tarjetas de accesorios en un computador personal, es una extensión del bus del 8088. Sin embargo, se ha amplificado su capacidad de manejar corriente y se ha mejorado añadiéndole entradas de interrupción y de DMA. Todas las señales de este bus son TTL compatibles, exceptuando las cinco señales de fuentes de corriente continua.

El conector para tarjetas de extensión tiene 62 líneas marcadas como A1..A31 y B1..B31, tiene: 8 bits de datos; 20 bits de direcciones, que se usan para memoria y puertos de E/S; y las restantes son líneas de control, entre las cuales están: Bus ocupado, peticiones de interrupción, actividad en COM1 o COM2. El bus del AT tiene además 36 líneas adicionales: 8 más de datos, 8 más de direcciones y 5 más de peticiones de interrupción.

A continuación se incluye una tabla en la que se indica la función de cada una de las líneas del conector para tarjetas de extensión del computador personal XT:

	B (sueldas)		A (componentes)
		(fondo del PC)	
1	-	GND	I -IOCHCK
2	O	RESET DRV (reset)	I/O D7 (MSB)
3	-	+5(V)	I/O D6
4	I	IRQ2	I/O D5
5	-	-5(V)	I/O D4 (datos)
6	I	DRQ2	I/O D3
7	-	-12(V)	I/O D2
8	I	-SRDY	I/O D1
9	-	+12(V)	I/O D0 (LSB)
10	-	GND	I IOCHRDY
11	O	-MEMW	O AEN
12	O	-MEMR	I/O A19
13	O	-IOW	I/O A18
14	O	-IOR	I/O A17
15	O	-DACK3 (contesta a DRQ3)	I/O A16
16	I	DRQ3 (peticiones DMA)	I/O A15
17	O	-DACK1 (contesta a DRQ)	I/O A14
18	I	DRQ1	I/O A13
19	I/O	-DACK0 (refresh)	I/O A12
20	O	CLK (4.77MHz)	I/O A11
21	I	IRQ7	I/O A10
22	I	IRQ6	I/O A9
23	I	IRQ5	I/O A8
24	I	IRQ4	I/O A7
25	I	IRQ3	I/O A6

26	O	-DACK2 (contesta a DRQ2)	I/O A5
27	O	T/C	I/O A4
28	O	ALE (Dirección válida)	I/O A3
29	-	+5(V)	I/O A2
30	O	OSC (14.3MHz)	I/O A1
31	-	GND	I/O A0

1.3.1.1 <u>Descripción de las señales disponibles en el conector para tarjetas de extensión del PC</u>.-

OSC, es la señal del oscilador con un ciclo de trabajo del 50%.

CLK, "SYSTEM CLOCK", resulta de la división de la señal OSC para tres, tiene un ciclo de trabajo del 33%.

RESET, señal utilizada para inicializar el sistema lógico tanto en el encendido del equipo, como en falla de voltaje. La señal está sincronizada con el flanco de bajada del reloj, y es activa en alto.

A0-A19, BUS DE DIRECCIONES, señales utilizadas para direccionar la memoria y los dispositivos de entrada/salida. Los 20 bits permiten direccionar hasta 1 MB de memoria. A0 es el bit menos significativo y A19 el más significativo. Estas líneas se generan tanto por el procesador como por

el controlador DMA. Las señales son activas en alto.

D0-D7, BUS DE DATOS, estas líneas proveen el bus de datos al procesador, la memoria y los dispositivos de entrada/salida. D0 es el bit menos significativo y D7 el más significativo. Estas líneas son activas en alto.

ALE, Adress Latch Enable, esta línea, provista por el 8288 (Controlador de Bus), se utiliza en la placa principal para almacenar direcciones válidas puestas por el procesador. Disponible en el bus de tarjetas de extensión se utiliza como una indicación de una dirección válida puesta por el procesador (cuando se usa con AEN). La dirección puesta por el procesador se almacena con el flanco de bajada de la señal ALE.

IOCHCK, "I/O Channel Check", esta señal provee al procesador de información de error de paridad tanto en la memoria como en los dispositivos de entrada/salida. Cuando esta señal se pone en bajo, se indica un error en la paridad, generando una señal de interrupción no enmascarable (NMI).

IOCHRDY, "I/O Channel Ready". Esta señal, normalmente en alto (indicando READY), puede ser colocada en bajo

("not ready") por la memoria o un dispositivo de entrada/salida, para aumentar el tiempo necesario para una transferencia. Posibilita que dispositivos lentos sean conectados al bus de ampliación con un mínimo de dificultad. Cualquier dispositivo lento que utilice esta línea debe ponerla en bajo inmediatamente después de detectar una dirección válida y un comando de lectura o escritura. Esta señal nunca debe mantenerse en bajo por más de 10 ciclos de reloj. El tiempo extendido para la lectura o escritura será un múltiplo entero de ciclos de la señal CLK.

IRQ2-IRQ7 "Interrupt Request" 2 a 7. Estas líneas se utilizan para indicar al procesador que un dispositivo de entrada/salida necesita ser atendido. IRQ2 tiene la mayor prioridad, mientras que IRQ7 la menor. Una petición de interrupción se realiza cambiando el estado de la señal IRQ de bajo a alto, y manteniéndola en alto hasta que sea reconocida por el procesador.

IOR, "Input/Output Read". Esta señal indica a un dispositivo de entrada/salida para que coloque datos en el bus. Puede ser colocada por el procesador o por el controlador DMA. La señal es activa en bajo.

IOW, "Input/Output Write". Esta señal indica a un dispositivo de entrada/salida para que lea los datos del bus. Puede ser colocada por el procesador o por el controlador DMA. La señal es activa en bajo.

MEMR, "Memory read". Indica a la memoria que debe colocar datos en el bus. Puede ser colocada por el procesador o el controlador DMA. La señal es activa en bajo.

MEMW, "Memory write". Indica a la memoria que debe almacenar los datos del bus. Puede ser colocada por el procesador o el controlador DMA. La señal es activa en bajo.

DRQ1-DRQ3 "DMA Request" 1 - 3. Estas líneas son canales asincrónicos de petición de los servicios DMA. DRQ1 es la de mayor prioridad y DRQ3 la de menor prioridad. La petición se genera colocando una línea DRQ en alto hasta que el correspondiente DACK se active (bajo).

DACK0-3 "DMA Acknowledge" 0 - 3. Estas líneas se utilizan para reconocer las peticiones de DMA. La señal DACK0 se utiliza para el refresco de memorias dinámicas. Las señales son activas en bajo.

AEN, "Address Enable". Esta línea se usa para desconectar el procesador y otros dispositivos de los canales de entrada/salida, con el fin de permitir las transferencias DMA. Cuando esta señal está en alto, el controlador DMA toma el control de los buses de datos y direcciones y de las líneas de lectura y escritura a memoria y a dispositivos de entrada/salida. Si la señal se encuentra en 0, indica que las direcciones existentes en el bus respectivo han sido colocadas por el procesador.

T/C, "Terminal Count". Esta línea genera un pulso cuando se termina la cuenta de cualquier transferencia DMA. La señal es activa en alto.

1.3.2 Pórticos del PC

A pesar de que el procesador 8088 puede direccionar hasta 64K puertos de entrada/salida, en el PC se utilizan únicamente los 10 bits menos significativos del bus de direcciones para direccionar a los dispositivos de entrada/salida, por lo tanto, se pueden direccionar 1024 puertos. De éstos, las siguientes direcciones ya se encuentran utilizadas:

000h-00Fh	Controlador DMA.
020h-021h	Controlador interrupciones.
040h-043h	Temporizadores.

060h-063h	Puertos teclado, cassette y parlante.
080h-083h	Registros página DMA.
200h-20Fh	Adaptador juegos (solo 200 usada).
278h-27Fh	Adaptador paralelo secundario.
2F8h-2FFh	Adaptador asíncrono secundario.
300h-31Fh	Tarjeta prototipos.
378h-37Fh	Adaptador paralelo primario.
380h-38Fh	Adaptador monocromático y paralelo impresora.
380h-3DFh	Adaptador color y gráficos.
3F0h-3F7h	Adaptador diskette.
3F8h-3FFh	Adaptador asíncrono primario.

Quedando por tanto, alrededor de 700 direcciones disponibles para las tarjetas de extensión adicionales.

1.3.3 Señales y direcciones utilizadas por la tarjeta DAS-128

De las señales disponibles en el conector de ocho bits para tarjetas de extensión del computador personal, la tarjeta DAS-128 utiliza las siguientes:

El bus de datos, es decir las señales D0 - D7, designadas en el conector como A2 - A9. Del bus de datos se tomaran las señales que luego se transmitirán a los puertos de salida y, además, en el indicado bus se colocarán los datos leídos desde los puertos de entrada; por lo tanto, se necesita que la

transferencia para el bus de datos sea bidireccional.

Del bus de direcciones se toman únicamente las señales A0 - A9, designadas en el conector como A22 - A31. En base a estas señales se decodifica la dirección de cada uno de los puertos disponibles en la tarjeta DAS-128.

Se utiliza además la señal de control AEN, con el fin de garantizar que la dirección ha sido colocada por el microprocesador, y excluyendo, por tanto, las direcciones colocadas por el manejador DMA.

De las señales de control, se utilizan además -IOR y -IOW que nos permiten determinar la dirección de las transferencias que se realizarán hacia el bus de datos.

Las señales del bus de direcciones y las de control necesitan únicamente transferencias unidireccionales, del conector hacia la tarjeta DAS-128.

Con el fin de polarizar los elementos de la tarjeta DAS-128, se ingresan además las 5 señales de fuente disponibles en las ranuras para tarjetas de extensión: +12(VDC), -12(VDC), +5(VDC), -5(VDC) y GND. La fuente de +5(VDC) se utiliza principalmente para alimentar a todos los circuitos TTL que forman parte de la tarjeta DAS-128, mientras que las fuentes de +12(VDC) y -12(VDC) para alimentar a los amplificadores operacionales, con el fin de garantizar voltaje 0 a su salida

cuando así se requiera. La fuente de -5(VDC) se utiliza en el conversor digital/analógico y en los multiplexers analógicos existentes. //

CAPITULO II

REQUERIMIENTOS Y DISEÑO

CAPITULO II

REQUERIMIENTOS Y DISEÑO

El presente capítulo se inicia enunciando las especificaciones que se pretende sean cubiertas por la tarjeta DAS-128; realiza luego una descripción de los bloques que componen la tarjeta y concluye explicando detalladamente la forma como se diseñó.

En lo referente al diseño, se analiza en primer lugar el diseño del "hardware". La explicación se realiza para cada uno de los bloques funcionales existentes en la tarjeta, enunciándose los justificativos para el uso de determinados circuitos integrados, los problemas encontrados en la ejecución práctica y las soluciones adoptadas.

El capítulo termina explicando la forma como se diseñó el circuito impreso utilizado en la tarjeta DAS-128.

2.1 ESPECIFICACIONES INICIALES

El presente trabajo proporciona una tarjeta de puertos de entrada y salida, tanto digitales como analógicos, que posibilite el uso de los computadores personales IBM compatibles en acciones de supervisión y de control, tanto a nivel industrial como en laboratorio, y que podrá ser utilizada con facilidad con objetivos didácticos.

La dirección asignada a los pórticos de la tarjeta DAS-128 puede ser cambiada, con el fin de posibilitar el uso de más de una tarjeta DAS-128 en un mismo computador personal.

La tarjeta DAS-128 decodifica 16 direcciones consecutivas diferentes, de las cuales se han tomado 8 direcciones para los puertos analógicos y 8 para los digitales.

Los pórticos analógicos funcionan con voltajes normalizados entre 0 y 10 (VDC).

Debido a que se han destinado 8 direcciones para los puertos analógicos, existen 8 puertos de 8 bits de entradas analógicas y 8 puertos de 8 bits de salidas analógicas, usándose para discriminarlos, las señales -IOR y -IOW disponibles para las tarjetas de extensión del PC.

La frecuencia de muestreo para los puertos analógicos será mayor a diez mil operaciones por segundo, instalada sobre un

computador XT compatible, con una señal de reloj de 10 MHz.

Los pórticos digitales, de ocho bits TTL, podrán ser utilizados únicamente como salidas o como entradas. Esto nos permite definir 64 puntos digitales usados como entradas y 64 puntos digitales usados como salidas (8 puertos de 8 bits). Al igual que en el caso de los puertos analógicos, se utilizan las señales: -IOR y -IOW para diferenciarlos.//

Con el fin de disponer de un puerto digital bidireccional, se deberán interconectar las señales de un puerto de salidas con las de un puerto de entradas y se deberá comandar la entrada de control disponible en el conector externo J1 de la tarjeta DAS-128.

Debido a que el espacio físico disponible para sacar puntos de cada tarjeta es limitado en el computador, se ha visto la necesidad de eliminar dos puertos digitales, uno de entradas y uno de salidas. La tarjeta DAS-128 provee, por tanto, únicamente 112 puntos digitales en lugar de los 128 posibles.

2.2 DESCRIPCION DE LA TARJETA DAS-128

La tarjeta DAS-128 se compone de cuatro bloques funcionales: "buffers" de entrada, lógica de decodificación de direcciones, puertos digitales y puertos analógicos, los cuales se han marcado con distintos colores en el diagrama esquemático que se adjunta en el Anexo No. 4.

2.2.1 "Buffers" de entrada

Debido a la limitación en el número de entradas que se pueden conectar a una salida TTL, y dado que en los computadores personales XT compatibles se provee hasta de 8 ranuras para tarjetas de extensión, a las señales disponibles se las debe cargar con una (máximo dos) compuertas TTL. Por esta razón es necesario incluir "buffers" en la tarjeta DAS-128, para todas las señales utilizadas.

2.2.2 Lógica de decodificación

Las direcciones asignada a los puertos disponibles en la tarjeta DAS-128 ocupan 16 localidades sucesivas a partir de una base que es posible definir mediante un "dip switch".

Una descripción detallada de la lógica de decodificación de direcciones para todos los puertos de la tarjeta DAS-128 se

realiza en el presente capítulo, en el acápite 2.3.1.6, cuando se describe el desarrollo del "hardware".

2.2.3 // Puertos

En la tarjeta DAS-128 existen dos tipos de puertos, digitales y analógicos, y para cada tipo existen dos clases, puertos de entradas y puertos de salidas.

Existe únicamente un caso especial con el primer puerto digital, el cual puede funcionar en forma bidireccional. Con el fin de posibilitar esta operación, en el conector J1 de la tarjeta DAS-128, se ha colocado una entrada la cual puede ser comandada por una señal externa al computador personal o por una señal que proviene de un bit de cualquiera de los restantes puertos de salidas digitales disponibles en la tarjeta DAS-128.

En el diagrama esquemático, que se adjunta en el Anexo No. 4, se han utilizado los siguientes colores para diferenciar los bloques funcionales:

Bloque	Color
"Buffers" de entrada	Café
Lógica de decodificación	Rojo

Puertos digitales de entrada	Naranja
Puertos digitales de salida	Amarillo
Puerto digital de entrada/salida	Verde
Puertos analógicos de entrada	Gris
Puertos analógicos de salida	Lila

2.3 DISEÑO

2.3.1 Diseño del "hardware"

A continuación se describen las consideraciones de diseño, más importantes, tomadas en cuenta para la construcción de cada uno de los bloques funcionales que conforman la tarjeta DAS-128 y que se describieron en el acápite 2.2.

2.3.1.1 Buffers de entrada.- Como dispositivos de entrada, se utilizan "buffers", unidireccionales para el bus de direcciones y señales de control y bidireccionales para el bus de datos. Para los "buffers" unidireccionales se ha utilizado los circuitos integrados 74LS244 y para los bidireccionales los circuitos integrados 74LS245, como se ilustra en el Gráfico No. 2.1.

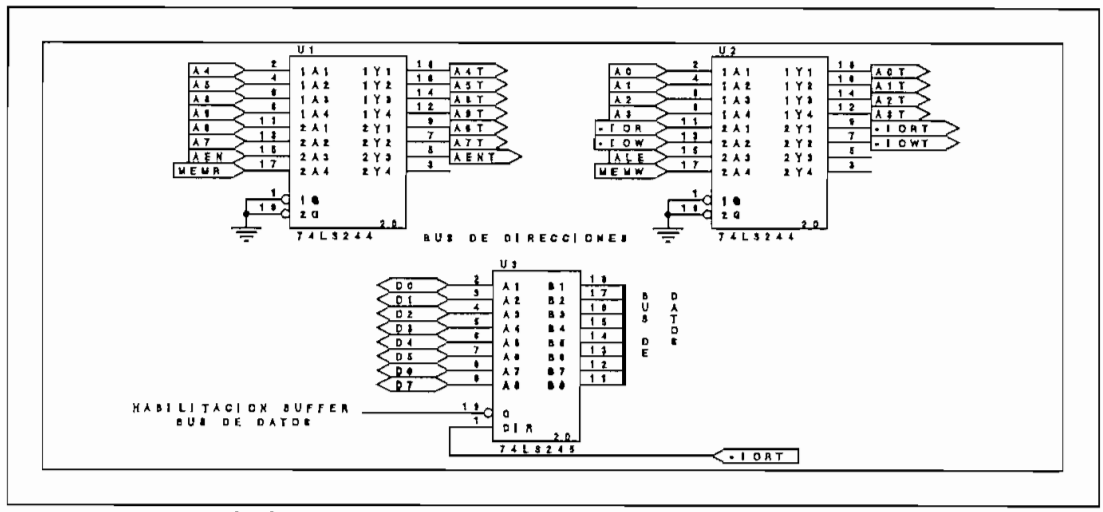


Gráfico No. 2.1: "Buffers de Entrada"

2.3.1.2 <u>Puertos digitales</u>.- Para los puertos de salidas digitales se utilizan "latches" (circuitos integrados 74LS374), mientras que para los puertos de entradas digitales se utiliza únicamente "buffers" (circuitos integrados 74LS244) como se ilustra en el Gráfico No. 2.2.

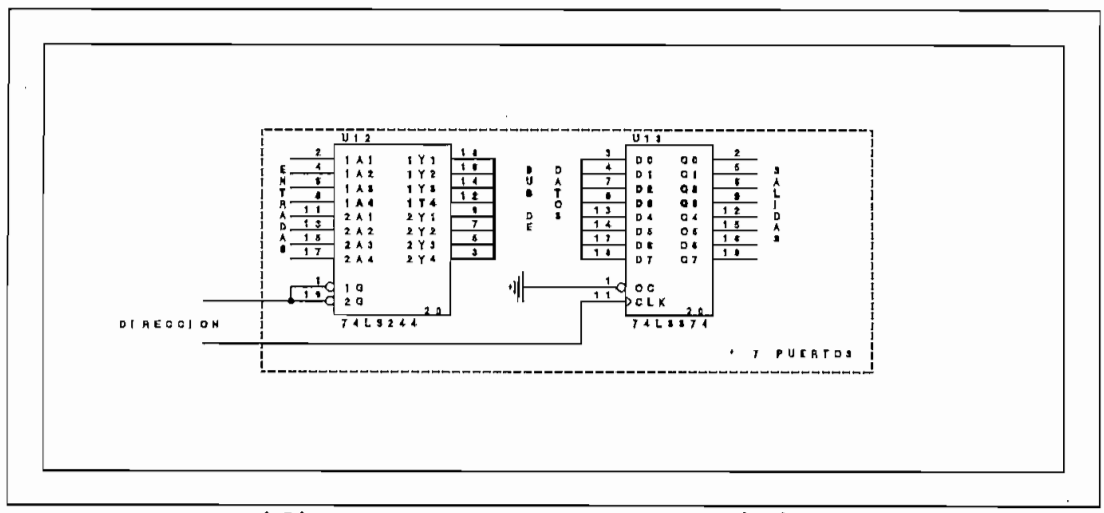


Gráfico No. 2.2: Puertos digitales

En primera instancia se había considerado utilizar los circuitos integrados 74LS373 como "latches", debido a que éstos son activados por estado y garantizaban la permanencia de los datos en el bus, mientras se mantenía habilitada su señal de control.

El uso de los 74LS374 se consideró conveniente con el fin de reducir el número de compuertas necesarias, ya que los 373 necesitaban de una compuerta negadora para su operación. El funcionamiento del sistema no se vio alterado, debido a que el computador, luego de deshabilitar la señal de control, mantiene en el bus de datos el estado durante un semiciclo de la señal del oscilador. Al usar los circuitos integrados 74LS374 se tiene una ventaja adicional, ya que los datos permanecen fijos en el bus mientras se realiza la transferencia a los "latches".

2.3.1.3 Puerto digital de entrada/salida.-

Con el fin de disponer de un puerto digital que permita la comunicación bidireccional, para los puertos digitales tanto de entrada como de salida cuya dirección es la más baja, se ha provisto de una señal adicional que permite definir externamente el uso que se dará al puerto. Para su operación se deberá interconectar los bits correspondientes del puerto de entradas con el de salidas y comandar la señal -DIO

disponible en el conector JP1 de la tarjeta DAS-128, como se ilustra en el Gráfico No. 2.3.

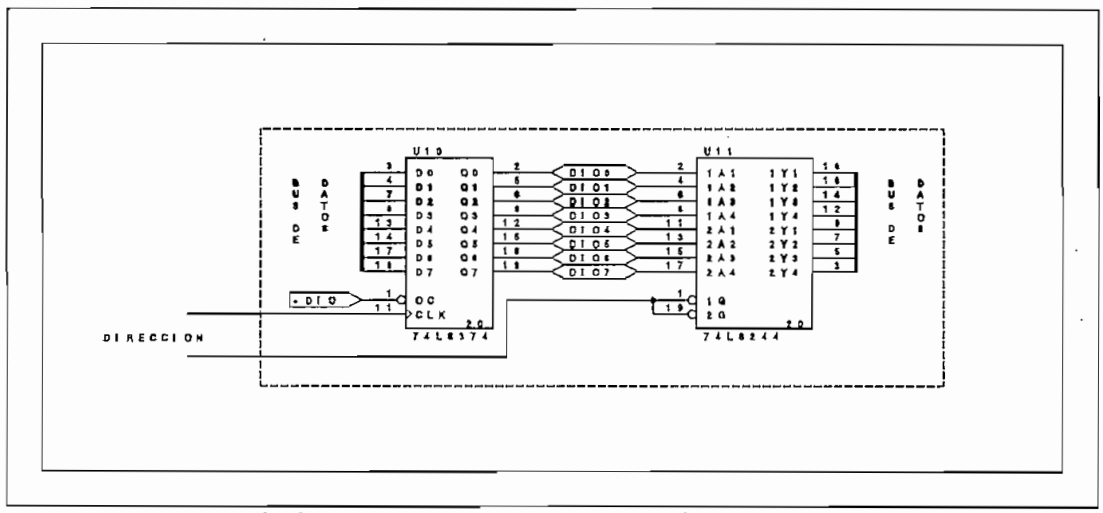


Gráfico 2.3: Puerto digital de E/S

2.3.1.4 **Puertos de entradas analógicas.-** Los 8 puertos de entradas analógicas se realizan en base a un conversor analógico/digital de 8 bits ADC0804 funcionando en modo de conversión continua y un multiplexer analógico 4051 para direccionar la entrada cuyos datos deben ser convertidos.

La señal de reloj, necesaria para la operación del conversor analógico/digital, se genera mediante el uso de una resistencia de 47 kohm, que se conecta entre los puntos CLKR y CLK del ADC0804, y un capacitor de 50 pF, que se conecta entre el punto CLK y tierra.

Debido a que la señal así generada no guarda sincronismo con

la del oscilador del PC, imposibilitando su uso en modo de conversión continua, se analizaron tres posibles soluciones:

- Evitar el modo de conversión continua, debiendo el PC ordenar la conversión cuando sea necesaria y esperar el tiempo requerido para poder leer los datos, solución que dificultaba el uso del conversor.

- Alimentar al conversor con una señal de reloj que sea submúltiplo del oscilador del PC, con lo cual la velocidad de conversión dependía de la frecuencia utilizada en el computador y, además, se requerían variaciones en el hardware de la tarjeta con el fin de poder conectarla a computadores que utilicen frecuencias diferentes.

- Mantener el modo de conversión continua, incluyendo un "buffer" entre el conversor y el bus de datos de la tarjeta, de forma que el computador no se comunique directamente con el conversor, siendo ésta la solución adoptada.

Con el fin de poder ingresar señales entre 0 y 10 (VDC) se han colocado divisores que además actúan como protección. Se ha incluido además un diodo zener de 5.1 (V) a la entrada del conversor para protegerlo.

En el Gráfico No. 2.4 se ilustra el circuito utilizado para

los puertos de entradas analógicas.

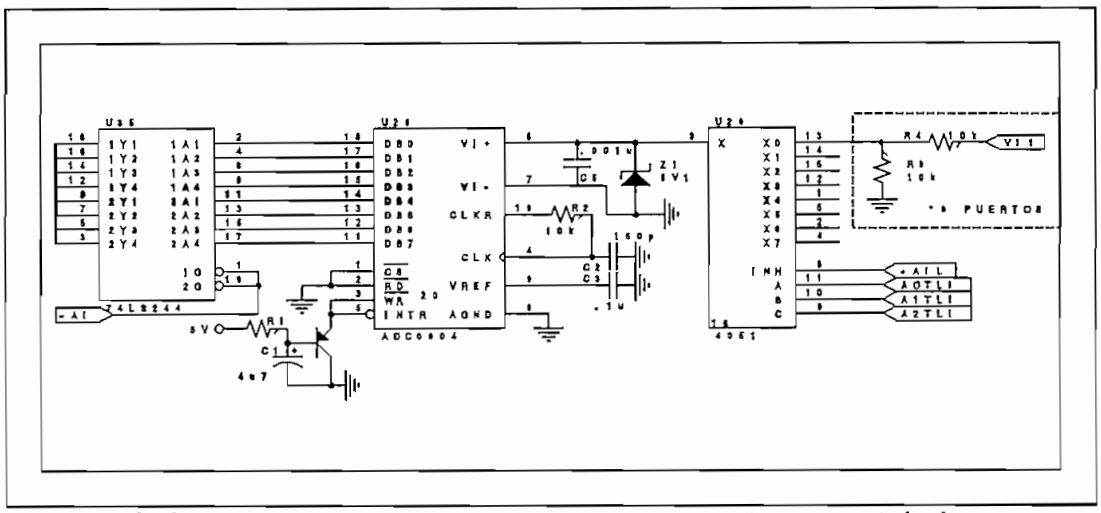


Gráfico No. 2.4: Puertos de entradas analógicas

2.3.1.5 Puertos de salidas analógicas.- Para los 8 puertos de salidas analógicas se ha utilizado un conversor de digital a analógico de 8 bits DAC0830; a su salida se ha conectado un multiplexer analógico 4051, que permite realizar el cambio de dirección a la señal de salida respectiva y se utilizan capacitores de 0.1 μF para mantener el voltaje de salida, los cuales se conectan a un amplificador operacional (en configuración amplificador no inversor), encargado de duplicar el voltaje, para obtener una salida entre 0 y 10 (VDC).

En el Gráfico No. 2.5 se ilustra el circuito utilizado para los puertos de salidas analógicas.

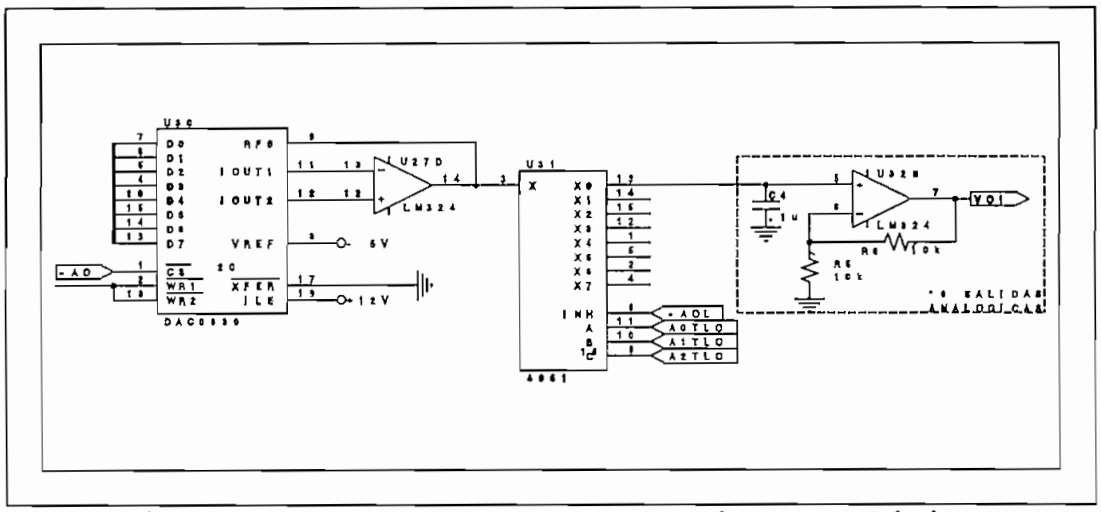


Gráfico No. 2.5: Puertos de salidas analógicas

2.3.1.6 <u>Decodificación de la dirección</u>.- Para la decodificación de las direcciones se utilizan los circuitos integrados 74LS138 y un "dip switch" que permite el cambio de la dirección de todos los puertos disponibles en la tarjeta DAS-128.

Luego de analizar la tabla de direcciones usadas y disponibles en el PC, se ha decidido utilizar las direcciones 200, 210, 220, 230, 300, 310, 320 y 330 para la tarjeta DAS-128. Con el fin de poder variar la dirección asignada, se ha utilizado un "dip switch" que define la dirección que se dará a la tarjeta DAS-128, por ser ésta la opción más económica. El "dip switch" conectará solo una de las 8 posibles líneas.

Para decodificar la dirección a cada uno de los puertos, se ha

realizado el siguiente análisis:

A9 debe estar en 1

A7 0

A6 0

A8 discrimina entre 2xx y 3xx

A4 y A5 definen x00, x10, x20 o x30

Por lo tanto, si se conectan las señales A4, A5 Y A8 a las entradas A, B y C de un 74LS138 y las señales A9, A6 y A7 a las entradas G1, G2A Y G2B respectivas, las salidas definen las direcciones 20x, 21x, 22x, 23x, 30x, 31x, 32x y 33x. Se conecta, por tanto, a estas salidas el "dip switch" que definirá la dirección de toda la tarjeta DAS-128. Se debe observar que la señal resultante del 138 es activa en bajo, razón por la cual se la llama -Ax.

Se debe cumplir además que la señal AEN debe estar en 0.

Si se llama (-A) a la señal:

$$-A = -(-Ax+AEN)$$

$$= Ax \cdot (-AEN)$$

El circuito utilizado para realizar la operación descrita se ilustra en el Gráfico No. 2.6.

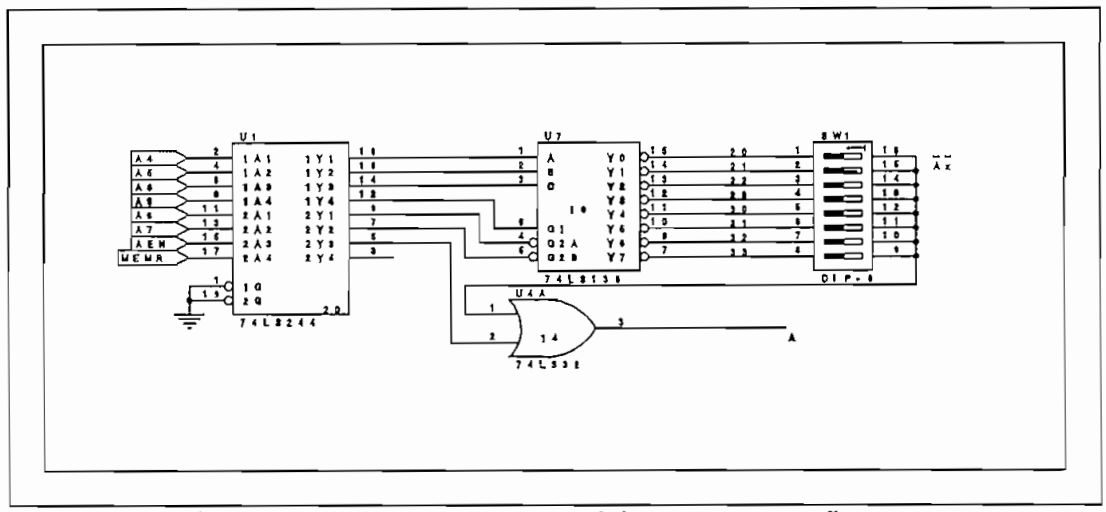


Gráfico No. 2.6: Obtención de la señal "A"

Para activar el "buffer" del bus de datos se necesita la señal:

$(-A) \cdot IOR + (-A) \cdot IOW$

Para que se active tanto en las lecturas como en las escrituras que cumplen con la dirección.

La señal indicada se puede expresar como:

$$-((A+(-IOR)) \cdot (A + (-IOW))) =$$

$$-(A + A \cdot (-IOR) + A \cdot (-IOW) + (-IOR) \cdot (-IOW)) =$$

$$-(A(1 + (-IOR) + (-IOW)) + (-IOR) \cdot (-IOW)) =$$

$$-(A + (-IOR) \cdot (-IOW))$$

Que define la lógica que debe ser utilizada para obtener la señal que habilita al "buffer" del bus de datos y que se ilustra en el Gráfico No. 2.7.

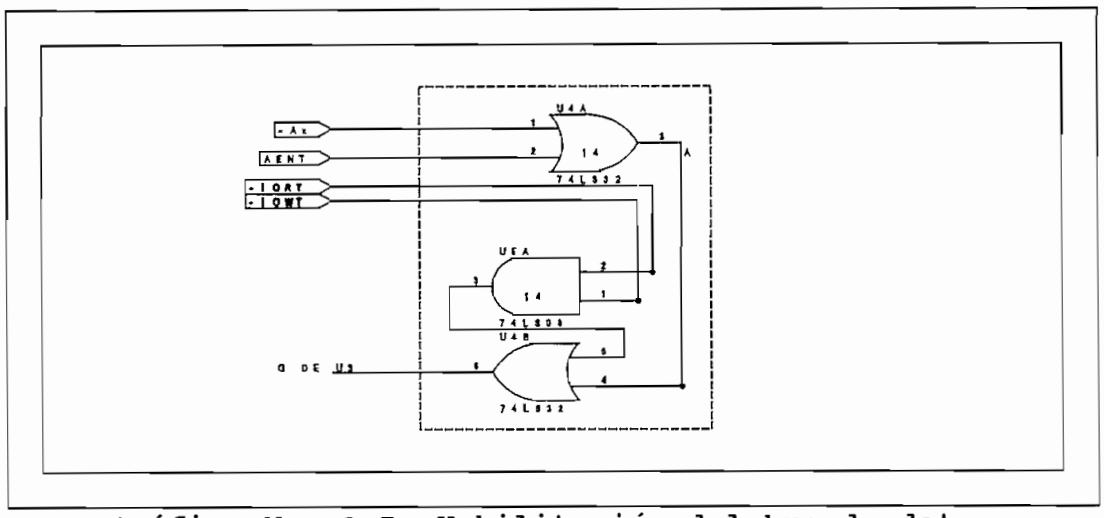


Gráfico No. 2.7: Habilitación del bus de datos

Para discriminar entre dispositivos analógicos y digitales, se ha utilizado la señal A3, colocando un valor 0 para los analógicos y 1 para los digitales. Por lo tanto, para activar las entradas analógicas se necesita la señal:

$$(-A) \cdot (-A3) \cdot IOR$$

Que puede ser expresada como:

$$-(A + A3 + (-IOR)) =$$
$$(-(A + A3) \cdot IOR)$$

Y para las salidas analógicas se necesita la señal:

$$(-A) \cdot (-A3) \cdot IOW$$

Que puede ser expresada como:

-(A + A3 + (-IOW))

Los circuitos utilizados para activar los puertos analógicos, tanto de entradas como de salidas, se ilustran en el Gráfico No. 2.8.

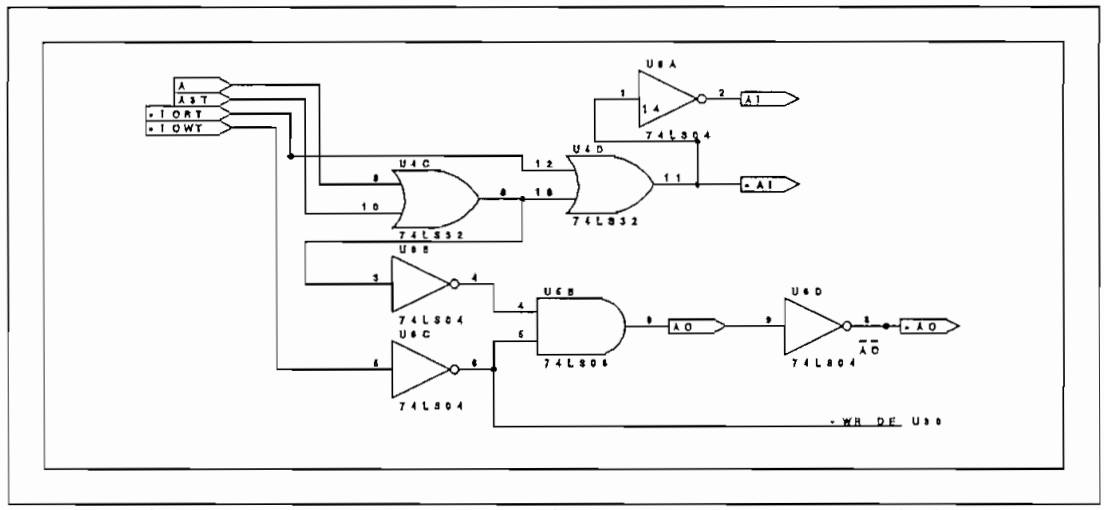


Gráfico 2.8: Habilitación de puertos analógicos

Para reducir el número de compuertas necesarias y simplificar el cableado, se han utilizado dos C.I. 74LS138 para los puertos digitales, uno para las entradas y otro para las salidas. A los dos circuitos integrados se conectan las señales A0, A1, A2 y A3 en las entradas A, B, C y G1 respectivamente.

La señal A, que se ha generado, se conecta a la entrada G2A de

los dos circuitos integrados. La entrada G2B del 74138 utilizado para direccionar los puertos de entrada, es conectada a la señal -IOR del bus, mientras que para el circuito integrado que direcciona los puertos de salida, se conecta la señal -IOW. Los circuitos utilizados para activar los puertos digitales, tanto de entradas como de salidas se ilustran en el Gráfico No. 2.9.

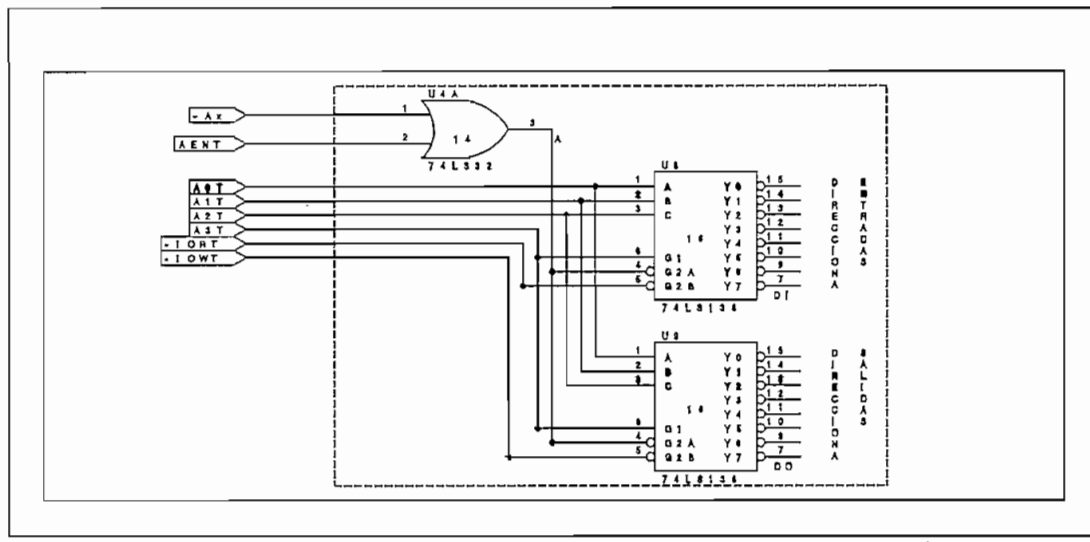


Gráfico No. 2.9: Habilitación de puertos digitales

2.3.2 Diseño del circuito impreso

Para el diseño del circuito impreso se ha utilizado el programa Tango-Route. Los diagramas del circuito impreso se adjuntan en el Anexo No. 5.

El diseño se ha realizado tratando de minimizar el área de la tarjeta.

Como una primera opción se analizó la posibilidad de que los conectores de salidas sean parte de la tarjeta, para lo cual todos debían estar ubicados en el borde posterior. Esta disposición resultaba peligrosa para la operación, ya que se debía tener mucho cuidado en la conexión de las señales necesarias y además fue imposible realizar el circuito impreso, por la gran cantidad de rutas que debían llegar al borde, considerando además la limitación impuesta en el ancho de la tarjeta, debida al espacio disponible en el PC. Posteriormente se decidió colocar los conectores en el interior de la tarjeta y sacarlos mediante cable plano al fondo del computador personal, sitio en el cual se colocarán conectores fijos, que reducirán los problemas para su conexión.

CAPITULO III

PRUEBAS Y RESULTADOS

CAPITULO III

PRUEBAS Y RESULTADOS

El capítulo se inicia describiendo cuatro pequeños programas que se han desarrollado para demostrar la operación de la tarjeta DAS-128. Se incluyen los listados de los indicados programas, que se han realizado utilizando Quick-Basic, lenguaje C y lenguaje ensamblador.

En la segunda parte se explican las diferentes pruebas realizadas, tendientes a verificar el correcto funcionamiento de la tarjeta DAS-128, y los resultados obtenidos luego de realizadas las pruebas.

El capítulo concluye realizando el análisis técnico económico de la tarjeta DAS-128 construida.

3.1 PROGRAMAS DE DEMOSTRACION

3.1.1 Programa No. 1, Comprobación de puertos

Con el fin de comprobar la correcta operación de la tarjeta DAS-128, se ha elaborado un programa que simultáneamente prueba dos puertos (analógicos o digitales), uno de entradas y uno de salidas.

El programa pone datos en el puerto de salidas que se desea analizar; las salidas se conectan externamente con el puerto de entradas bajo análisis, y son leídas por el computador.

Los datos que el computador ha sacado y que ha recibido se despliegan en pantalla, con el fin de verificar su coincidencia.

El programa se ha realizado utilizando el lenguaje Quick Basic.

3.1.1.1 Listado del Programa No. 1 en Quick Basic.-

```
'PROGRAMA PARA PROBAR PUERTOS DE ENTRADAS Y SALIDAS

DECLARE SUB LIN ()
SCREEN 0
CLS
UPP = 255          '**MAXIMO VALOR EN EL PUERTO
PASO = 1           '**INCREMENTO EN LOS VALORES
ndif = 0           '**INICIA CONTADOR DE DIFERENCIAS
CALL LIN             '**SUBRUTINA QUE DIBUJA UN CUADRO EN LA
PANTALLA
LOCATE 2, 10
PRINT "PROGRAMA PARA PROBAR UN PUERTO DE ENTRADA Y UNO DE
SALIDA"
LOCATE 5, 10
INPUT "Ingrese dirección decimal de los pórticos a probar:";
PORT
datoin = INP(PORT)
LOCATE 10, 10
PRINT "Dato que sale:", " Dato que ingresa:"
LOCATE 15, 10
PRINT "Conteo de diferencias:"
10 LOCATE 20, 10
INPUT "Prueba Continua o en Pasos (C/P):"; a$
                          '**PRUEBA CONTINUA O PASO A PASO ?
IF a$ = "p" OR a$ = "P" THEN
                          '**PARA PRUEBA PASO A PASO
   LOCATE 20, 10
   PRINT "Presione una tecla para continuar....."
   ELSE
   IF (a$ <> "c" AND a$ <> "C") THEN
                       '**SI RESPONDE MAL, PREGUNTAR OTRA VEZ
      GOTO 10
   END IF              '**PARA PRUEBA CONTINUA
```

```
END IF
FOR dato = 0 TO UPP STEP PASO
   OUT (PORT), dato        '**SACA dato AL PUERTO DE SALIDAS
   datoin = INP(PORT)     '**LEE datoin DEL PUERTO DE ENTRADAS
   IF VAL(RIGHT$(HEX$(PORT), 1)) < 8 THEN
      datoin = INP(PORT)
                    '**PARA PUERTO ANALOGICO, NUEVA LECTURA
   END IF
   LOCATE 11, 10
   PRINT USING "###&###"; dato; "                  "; datoin
   IF dato <> datoin THEN '**SI dato <> datoin:
      LOCATE 15, 35
      ndif = ndif + 1  '**INCREMENTA CONTADOR DE DIFERENCIAS
      PRINT ndif           '**E IMPRIMIR SU VALOR
   END IF
   IF a$ = "p" OR a$ = "P" THEN
                           '**PARA PREUBA PASO A PASO: ESPERAR
                           '**UNA TECLA PARA CONTINUAR
      WHILE INKEY$ = ""
      WEND
   END IF
NEXT dato
```

El programa utiliza una subrutina que permite dibujar un marco en la pantalla, cuyo listado es el siguiente:

```
SUB LIN
LOCATE 1, 1
PRINT "┌";
FOR z = 1 TO 78
PRINT "─";
NEXT z
PRINT "┐"
FOR z = 2 TO 21
PRINT "│";
```

```
LOCATE z, 80
PRINT "|"
NEXT z
PRINT "└";
FOR z = 1 TO 78
PRINT "—";
NEXT z
PRINT "┘"
END SUB
```

Esta subrutina, sin ninguna variación, se utiliza en todos los programas realizados en Quick Basic.

3.1.2 Programa No. 2, Obtención de señal rampa

Con el fin de determinar la frecuencia, a la cual puede operarse la tarjeta DAS-128, y comparar la velocidad de respuesta de programas compilados con diferentes compiladores, se ha realizado un programa que saca una rampa a un puerto de salidas analógicas.

El programa se ha realizado en Quick Basic, lenguaje C y lenguaje ensamblador.

Los listados se incluyen a continuación:

3.1.2.1 Listado del Programa No. 2 en Quick Basic.-

```
'PROGRAMA PARA SACAR UNA RAMPA A UN PUERTO ANALOGICO

DECLARE SUB LIN ()

COUNT = 10000     '**NUMERO DE RAMPAS QUE SALEN
UPP = 255         '**MAXIMO DE LA RAMPA
nsal = 0          '**INICIA CONTADOR DE RAMPAS
SCREEN 0
CLS
CALL LIN      '**SUBRUTINA QUE DIBUJA UN CUADRO EN LA PANTALLA
LOCATE 2, 10
PRINT "    PROGRAMA PARA PROBAR PUERTOS DE SALIDAS ANALOGICAS"
LOCATE 3, 10
PRINT "       SE OBTIENE UNA RAMPA EN EL PUERTO SELECCIONADO"
10 LOCATE 5, 10
INPUT "Ingrese dirección decimal del pórtico a probar:"; PORT
   IF VAL(RIGHT$(HEX$(PORT), 1)) > 8 THEN
            '**SI EL PUERTO ES DIGITAL, INGRESE NUEVO VALOR
      LOCATE 10, 10
      PRINT "puerto digital, ingrese nuevo valor"
      GOTO 10
   ELSE
      LOCATE 10, 10
      PRINT ".........sacando datos............."
   END IF
DO
   FOR dato = 0 TO UPP STEP 1
   OUT (PORT), dato  '**SACA dato AL PUERTO ESCOGIDO
   NEXT dato
 nsal = nsal + 1'**INCREMENTA CONTADOR DE SALIDAS REALIZADAS
LOOP WHILE nsal <= COUNT
```

3.1.2.2 Listado del Programa No. 2 en Lenguaje C.-

```
/* PROGRAMA PARA SACAR UNA RAMPA A UN PUERTO ANALOGICO */

#include <stdio.h>
#include <dos.h>

#define PORT 528     /* DIRECCION DEL PUERTO */
#define COUNT 10000 /* NUMERO DE RAMPAS QUE SALEN*/
#define UPP 255      /* MAXIMO DE LA RAMPA*/

void main(void)
{
unsigned register int nsal,dato;
nsal=0;       /* INICIA CONTADOR DE RAMPAS*/
do
  {
   for (dato=0;dato<=UPP;dato++) { outp (PORT,dato);
             /* SACA dato AL PUERTO ESCOGIDO*/
                                  }
   nsal++;    /* INCREMENTA CONTADOR DE SALIDAS REALIZADAS*/
  }
while (nsal<=COUNT);
}
```

3.1.2.3 Listado del Programa No. 2 en lenguaje ensamblador.-

```
data  segment
co    dw ?
data  ends
prog  segment
   assume cs:prog,ds:data
port  equ   528d  ;DIRECCION DEL PUERTO
count equ   270fh ;NUMERO DE RAMPAS QUE SALEN
upp   equ   0ffh  ;MAXIMO DE LA RAMPA
go:   mov cx,count
      mov dx,port
      mov al,0
cic:  out dx,al
      cmp al,upp
      jne cont
      mov al,0ffh
      loop cont
      mov al,0ffh
      mov ah,4ch
      int 21h
      ret
cont: inc al
      jmp cic
prog  ends
      end
```

3.1.3 Programa No. 3, Operaciones lógicas

Como ejemplo de la operación de los puertos digitales de entrada y salida de la tarjeta DAS-128, se ha realizado un programa que obtiene información del estado de las entradas de

un puerto digital.

Utilizando los dos bits menos significativos de los datos leídos, realiza las operaciones lógicas AND, -AND, OR, -OR, XOR y -XOR, colocando el resultado de todas ellas en los seis bits menos significativos del puerto de salidas. Con los seis bits más significativos del puerto de entradas se define el período para una señal oscilante que se coloca en el bit 7 del puerto de salidas; en el bit 8 se coloca el complemento de esta señal.

3.1.3.1 Listado del Programa No. 3 en Quick Basic.-

```
'PROGRAMA PARA REALIZAR OPERACIONES LOGICAS CON LOS DATOS DE
'UN PUERTO DE ENTRADAS Y SACAR LOS RESULTADOS A UNO DE
'SALIDAS

DECLARE SUB LIN ()

COUNT = 10000    '**NUMERO DE CASOS QUE SE ANALIZAN
nsal = 0         '**INICIA CONTADOR DE CASOS
SCREEN 0
CLS
CALL LIN     '**SUBRUTINA QUE DIBUJA UN CUADRO EN LA PANTALLA
LOCATE 2, 10
PRINT "PROGRAMA PARA REALIZAR OPERACIONES LOGICAS CON DATOS"
LOCATE 3, 10
PRINT "    EXISTENTES EN UN PUERTO DE ENTRADAS DIGITALES"
10 LOCATE 5, 10
INPUT "Ingrese dirección decimal de puertos a probar:"; PORT
IF VAL(RIGHT$(HEX$(PORT), 1)) < 8 THEN
```

```
            '**SI EL PUERTO ES ANALOGICO, INGRESE NUEVO VALOR
   LOCATE 10, 10
   PRINT "puerto analógico, ingrese nuevo valor"
   GOTO 10
ELSE
   LOCATE 10, 10
   PRINT "Dato que ingresa:", " Dato que sale:    "
END IF
DO
   datoin = INP(PORT)
   k = VAL(RIGHT$(OCT$(datoin), 1))
   IF k >= 4 THEN k = k - 4
   SELECT CASE k
      CASE 0
      sal = 42
      CASE 1
      sal = 22
      CASE 2
      sal = 22
      CASE 3
      sal = 37
   END SELECT
   k = INT(datoin / 4)
   dato = sal OR 128
   LOCATE 11, 10
   PRINT HEX$(datoin); "                    "; HEX$(dato)
   FOR i = 0 TO k STEP 1
      OUT (PORT), dato'**SACA EL VALOR AL PUERTO DE SALIDAS
   NEXT i
   dato = sal OR 64
   LOCATE 11, 10
   PRINT HEX$(datoin); "                    "; HEX$(dato)
   FOR i = 0 TO k STEP 1
      OUT (PORT), dato'**SACA EL VALOR AL PUERTO DE SALIDAS
   NEXT i
   nsal = nsal + 1
LOOP WHILE nsal <= COUNT
```

3.1.4 Programa No. 4, Muestreo de señal analógica

El presente programa permite ingresar una señal a un puerto de entradas analógicas y realizar un muestreo de dicha señal. Los datos resultantes del programa pueden ser redireccionados a un archivo, el cual puede ser utilizado para construir un gráfico de la señal obtenida.

3.1.4.1 Listado del Programa No. 4 en Quick Basic.-

```
'PROGRAMA PARA INGRESO DE UNA SEÑAL ANALOGICA

DECLARE SUB LIN ()
COUNT = 4000     '**NUMERO DE DATOS MUESTREADOS
nsal = 0         '**INICIA CONTADOR
dim datoin(count)
SCREEN 0
CLS
CALL LIN      '**SUBRUTINA QUE DIBUJA UN CUADRO EN LA PANTALLA
LOCATE 2, 10
PRINT "        PROGRAMA PARA PROBAR PUERTOS DE ENTRADAS
ANALOGICAS"
LOCATE 3, 10
PRINT "       SE MUESTREA UNA SEÑAL DEL PUERTO SELECCIONADO"
10 LOCATE 5, 10
INPUT "Ingrese dirección decimal del pórtico a probar:"; PORT
   IF VAL(RIGHT$(HEX$(PORT), 1)) > 8 THEN
            '**SI EL PUERTO ES DIGITAL, INGRESE NUEVO VALOR
      LOCATE 10, 10
      PRINT "puerto digital, ingrese nuevo valor"
      GOTO 10
   ELSE
```

```
      LOCATE 10, 10
      PRINT ".....muestreando datos....."
   END IF
FOR nsal = 0 TO COUNT
   datoin(nsal)=inp(PORT) '**INGRESA DATO DEL PUERTO
NEXT nsal
OPEN "DATOS.PRN" FOR OUTPUT AS #1
FOR nsal = 0 TO COUNT
   PRINT #1, datoin(nsal)
NEXT nsal
CLOSE
```

3.1.4.2 Listado del Programa No. 4 en Lenguaje C.-

```
/* PROGRAMA PARA INGRESO DE UNA SEÑAL ANALOGICA */
#include <stdio.h>
#include <dos.h>
#define PORT 528     /* DIRECCION DEL PUERTO */
#define COUNT 10000 /* NUMERO DE DATOS INGRESADO*/
#define DATPLIN 18  /* NUMERO DE DATOS POR LINEA IMPRESA*/
void main(void)
{
 unsigned short int datoin[COUNT];
 unsigned register int nin;
 for (nin=0;nin<COUNT;nin++)
     { datoin[nin]=inp(PORT);
     }
 for(nin=0;nin<COUNT;nin++)
     { printf ("%3d\t",datoin[nin]);
                    /*IMPRESION DE DATOS*/
       if ((nin+1)%DATPLIN==0) printf("\n");
                    /*NUMERO DE DATOS POR LINEA*/
     }
}
```

3.2 PRUEBAS DE FUNCIONAMIENTO

3.2.1 Pruebas de compatibilidad

La primera prueba realizada tenía la finalidad de verificar la compatibilidad de la tarjeta DAS-128 en diversos equipos. Para el efecto, se la instaló en los siguientes computadores:

- Packard Bell; compatible con XT, frecuencia 10 MHz.
- Goupil; compatible con AT, frecuencia 10 MHz.
- Paradise; compatible con 386SX, frecuencia 16MHz.
- DTK, compatible con XT, frecuencia 10 MHz.
- MAGITRONIK, compatible con 386DX, frecuencia 40 MHz.

y se corrió el programa detallado en el numeral 3.1.1, habiéndose interconectado los 8 terminales correspondientes de un puerto de entradas y uno de salidas digitales, así como el terminal de un puerto de salidas analógicas con uno de entradas analógicas.

Las siguientes pruebas se realizaron en el computador DTK disponible en el laboratorio de Electrónica de Potencia. Dicho computador es compatible con XT y tiene una frecuencia máxima de reloj de 10 MHz.

3.2.2 Pruebas de correcta operación

Con el fin de verificar y garantizar la adecuada operación de todos los componentes, se utilizó el programa detallado en el numeral 3.1.1. Dicho programa fue corrido para cada uno de los puertos disponibles y luego se asignó a la tarjeta las distintas direcciones posibles.

3.2.3 Pruebas de frecuencia de operación

Se realizaron pruebas tendientes a determinar la máxima frecuencia de operación para los puertos de:

- salidas analógicas, utilizando los programa detallado en el numeral 3.1.2;

- entradas analógicas, utilizando el programa detallado en el numeral 3.1.4;

- y salidas digitales, enviando una señal oscilatoria mediante un sencillo programa que trataba de incluir el menor retardo posible en el envío de la información al puerto.

Debido a que los programas que permiten la obtención de una señal rampa, y que se utilizaron para determinar la frecuencia de operación de los puertos de salidas analógicas, fueron

realizados en Quick-Basic, Lenguaje C y Ensamblador, fue posible además comparar la respuesta del computador utilizando distintos compiladores.

3.3 RESULTADOS OBTENIDOS

3.3.1 Pruebas de compatibilidad

La tarjeta demostró operar adecuadamente en todos los computadores probados, sin alterar el funcionamiento interno del computador personal. Se debe indicar que en cada computador se debió cambiar la dirección asignada a la tarjeta, con el fin de evitar conflicto en las direcciones de otras tarjetas existentes.

3.3.2 Pruebas de correcta operación

En el computador DTK existente en el laboratorio de Electrónica de Potencia se realizaron las pruebas tendientes a determinar la correcta operación de cada una de las funciones existentes en la tarjeta, pudiéndose observar que todos los puertos, tanto analógicos como digitales, operaban adecuadamente.

3.3.3 Pruebas de frecuencia de operación

Con el fin de realizar las pruebas de frecuencia de operación, y como ya se explicó en el numeral 3.2.3, se realizaron tres tipos de pruebas.

En lo referente a las pruebas de los puertos de salidas analógicas, para los cuales se realizaron tres programas, uno en Quick-Basic, otro en Lenguaje C y el último en Lenguaje Ensamblador, los resultados obtenidos se adjuntan en los Gráficos Nos. 3.1, 3.2 y 3.3.

Escala vertical:
2 V/div
Escala horizontal:
0.5 s/div

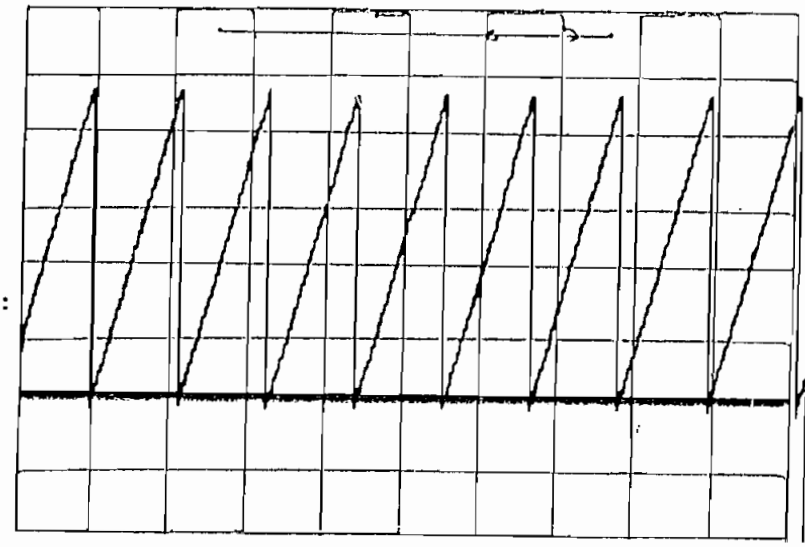


Gráfico No. 3.1: Obtención de rampas, Quick-Basic

Como se puede observar en el Gráfico No. 3.1 la frecuencia máxima de operación, para el programa realizado en Quick-Basic, es de 460 Hz.

En los Gráficos Nos. 3.2 y 3.3 se observan los resultados obtenidos para los programas realizados en Lenguaje C y Lenguaje Ensamblador. Se puede observar que la diferencia existente en la frecuencia que se logra alcanzar para los dos tipos de Lenguaje de programación, no es mayor, siendo la frecuencia máxima de operación de 140 kHz para el programa en lenguaje Ensamblador, y de 134 kHz para el programa en

lenguaje C.

Escala vertical:
2 V/div
Escala horizontal:
2 ms/div

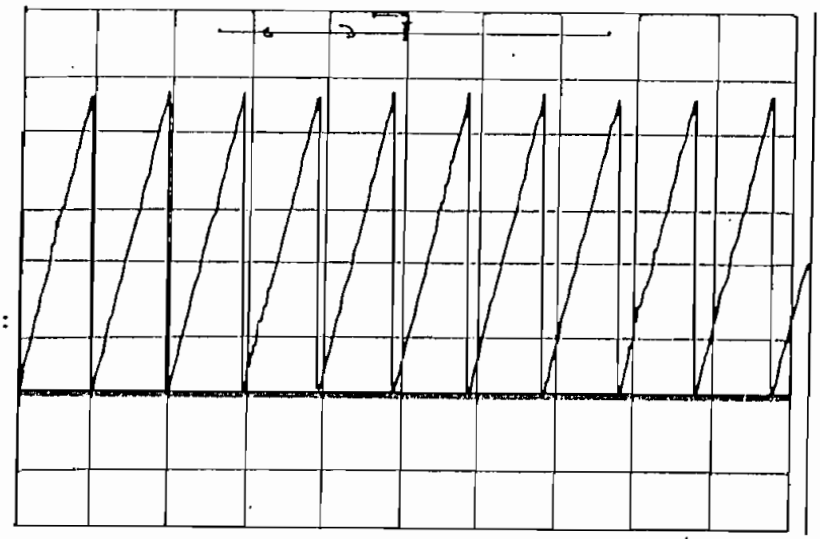


Gráfico No. 3.2: Obtención de rampas, lenguaje C

Escala vertical:
2 V/div
Escala horizontal:
2 ms/div

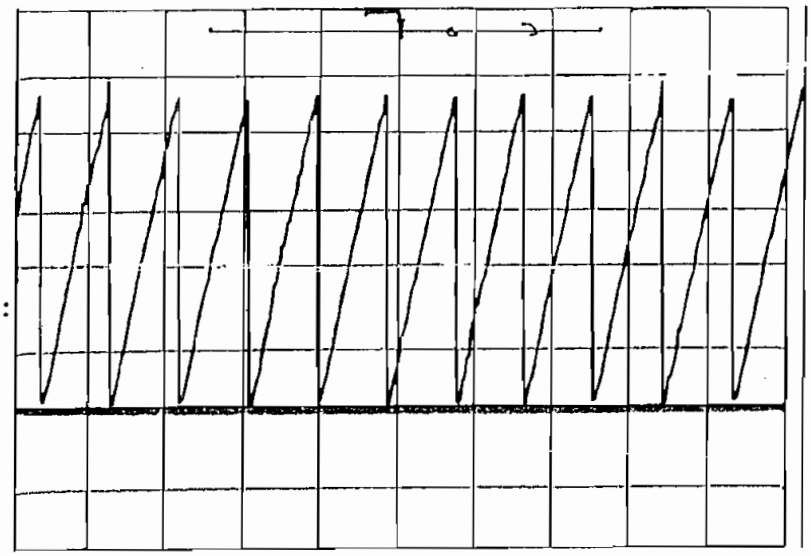


Gráfico No. 3.3: Obtención de rampas, lenguaje Ensamblador

Se debe indicar que, como consta en los listados de los programas que se adjuntan en el acápite 3.1, no se han incluido retardos para la salida de los datos, por lo tanto, las frecuencias indicadas son las máximas que se pueden

alcanzar para el computador en el que se realizaron las pruebas.

Con el fin de realizar las pruebas correspondientes para el puerto de entradas analógicas, se ingresaron dos señales sinusoidales a un puerto de entradas analógicas, una de 60 Hz y otra de 300 Hz, y se realizó el muestreo digital.

El muestreo digital fue realizado utilizando el programa que se describe en el acápite 3.1.4.2, en lenguaje C, debido a que se pudo comprobar que el programa realizado en Quick-Basic resultaba muy lento para cualquier uso práctico.

En los Gráficos Nos. 3.4 y 3.5 se adjunta el resultado de reconstruir las señales ingresadas.

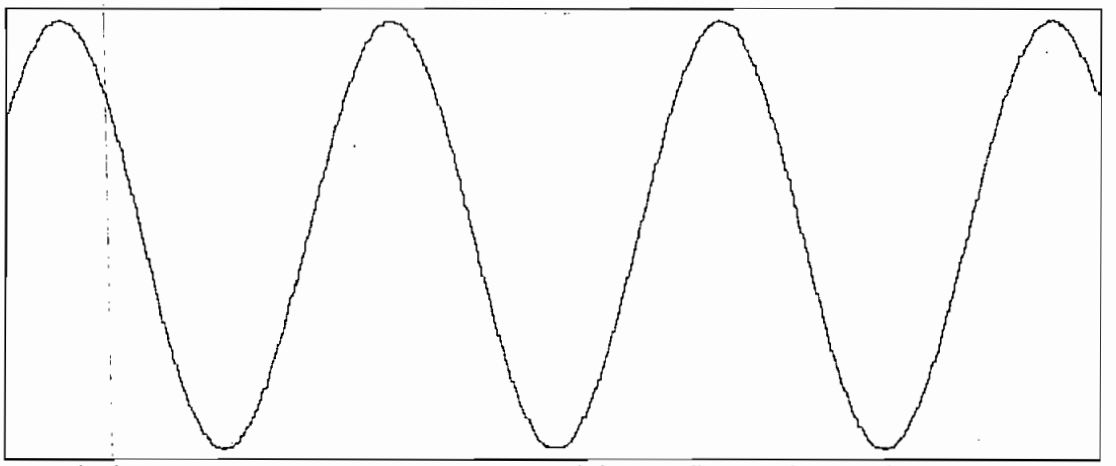


Gráfico No. 3.4: Reconstrucción señal sinusoidal 60 Hz

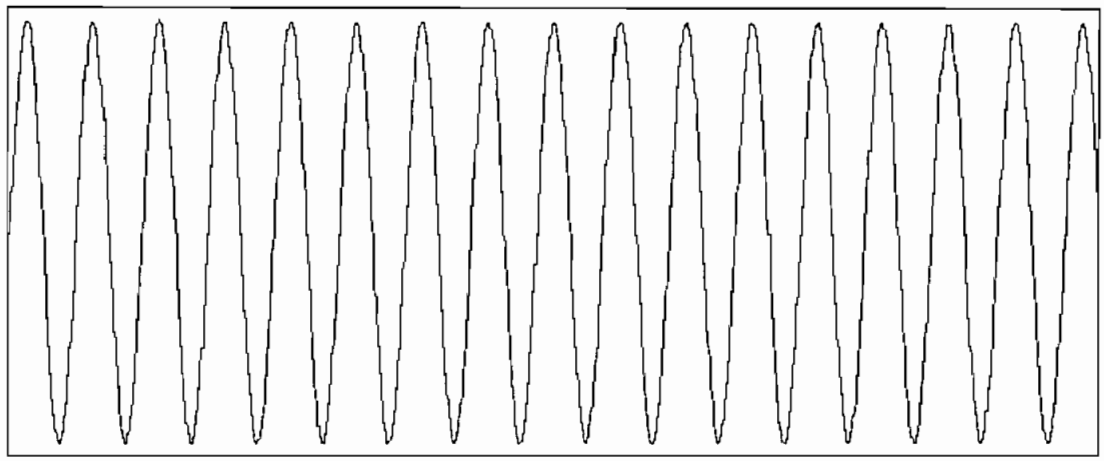


Gráfico No. 3.5: Reconstrucción señal sinusoidal 300 Hz

Del listado del programa (acápite 3.1.4.2) se puede observar que no se incluyen retardos para la adquisición de los datos. En la prueba anterior, el lenguaje C demostró tener características semejantes a las del Ensamblador, en lo que se refiere a manejo de puertos, lo que nos permitía tener confianza en los resultados que se obtendrían en la presente prueba.

Los resultados de la digitalización se grabaron en un archivo ASCII, y luego se procedió a reconstruir la señal muestreada.

Analizando el contenido de los datos existentes en los indicados archivos ASCII, se pudo observar que la frecuencia máxima de muestreo alcanzada es de 12kHz.

Con el fin de probar la frecuencia de operación de las señales

digitales, se realizaron dos sencillos programas, uno en Quick-Basic y otro en Lenguaje C. Los resultados de las pruebas se pueden observar en los gráficos 3.6 y 3.7.

Escala vertical:
1 V/div
Escala horizontal:
0.1 ms/div

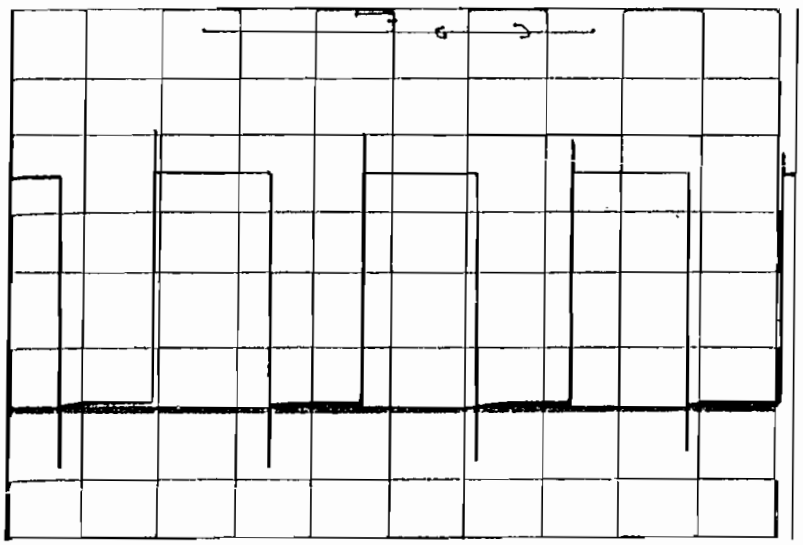


Gráfico No. 3.6: Prueba de salida digital, Quick-Basic.

Escala vertical:
1 V/div
Escala horizontal:
5 us/div

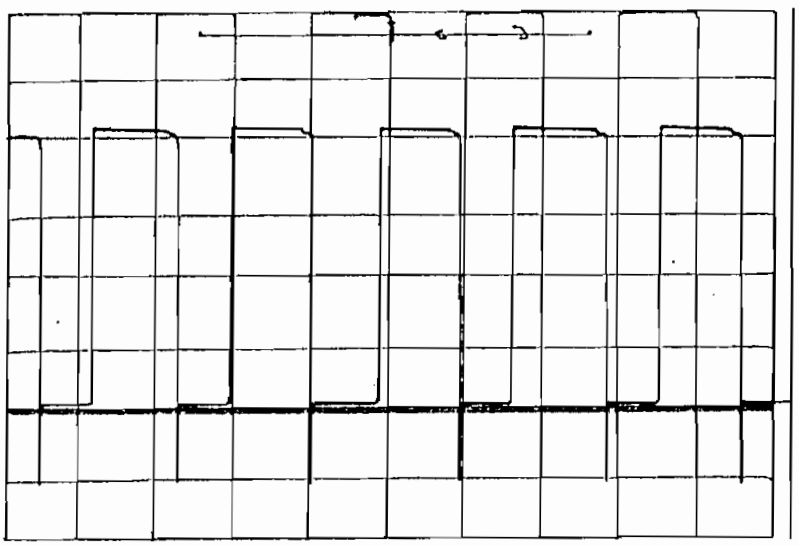


Gráfico No. 3.7: Prueba de salida digital, Lenguaje C.

Como se puede observar, la frecuencia máxima que se pudo alcanzar, en el computador en el que se realizaron las pruebas, fue de 4 kHz para el programa realizado en Quick Basic y de 100 kHz para el programa realizado en Lenguaje C.

3.4 ANALISIS TECNICO ECONOMICO

3.4.1 Análisis económico

A continuación se incluye un listado de los elementos requeridos para la construcción de la tarjeta DAS-128 y el precio en dólares de cada uno de ellos, actualizado a enero de 1993:

Item	Cant.	Referencia	Parte	Costo c/u	Total
1	10	U1,U2,U11,U12,U14,U16, U18,U20,U22,U35	74LS244	.69	6.90
2	1	U3	74LS245	.79	.79
3	1	U4	74LS32	.25	.25
4	1	U5	74LS08	.19	.19
5	1	U6	74LS04	.25	.25
6	3	U7,U8,U9	74LS138	.39	1.17
7	1	SW1	DIP-8	1.29	1.29
8	1	U26	ADC0804	3.95	3.95
9	2	U31,U27	4051	.50	1.00
10	1	U30	DAC0830	3.95	3.95
11	17	R2,R4,R6,R8,R10,R12,R14, R16,R18,R20,R22,R24,R26, R28,R30,R32,R34	10k	.06	1.02
12	3	U28,U32,U33	LM324	.39	1.17
13	42	C5,C4,C6,C7,C8,C9,C10, C11,C12,C17,C18,C19,C20, C21,C22,C23,C24,C25,C26, C27,C28,C29,C30,C31,C32, C33,C34,C35,C36,C37,C38, C39,C40,C41,C42,C43,C44, C45,C46,C47,C48,C49	.1µF	.19	7.98
14	1	C2	150pF	.19	.19

15	2	U29,U34	74LS75	.29	.58
16	7	U10,U13,U15,U17,U19,U21, U23	74LS374	.79	5.53
17	1	C3	.001µF	.19	.19
18	4	C13,C14,C15,C16	4µ7	.69	2.76
19	2	RPACK2,RPACK1	10K	.75	1.50
20	4	JP1,JP2,JP3,JP4	34PIN	2.59	10.36
21	1	Z2	LM336	1.09	1.09
22	1	R35	2K2	.06	.06
23	1	Q1	PNP	.40	.40
24	1	Z1	5V1	.59	.59
25	1	R1	47K	.06	.06
26	1	C1	33µF	.69	.69
27	6	SOCALOS	14 PIN	.15	.90
28	8	SOCALOS	16 PIN	.17	1.36
29	20	SOCALOS	20 PIN	.28	5.60
30	1	CIRCUITO IMPRESO		30.	30.
31	1	CABLE PLANO (m)	34 HILOS	2.	2.

El costo de la tarjeta DAS-128 es: 93.77

3.4.2 Análisis técnico

Las características técnicas de la tarjeta DAS-128 son las siguientes:

Puertos digitales de entrada:

Siete puertos de ocho bits cada uno. Señales TTL compatibles. Capacidad de conectar hasta 20 compuertas LS.

Puertos digitales de salida:

Siete puertos de ocho bits cada uno. Señales TTL

compatibles. Capacidad de conectar hasta 20 compuertas LS.

Puertos analógicos de entrada:

Ocho puertos multiplexados de ocho bits. Señal de entrada entre 0 y 10 (VDC).

Puertos analógicos de salida:

Ocho puertos de ocho bits multiplexados. Señal de salida entre 0 y 10 (VDC).

Puertos digitales de entrada/salida:

Un puerto de ocho bits. Se lo obtiene interconectando los bits correspondientes de uno de los puertos de entradas con uno de los de salidas, y se dispone de una entrada adicional que permite definir si la operación a realizarse será de o hacia el computador.

Frecuencia máxima de operación de los puertos digitales:

100kHz.

Frecuencia máxima de muestreo de los puertos analógicos:

10kHz.

CAPITULO IV

CONCLUSIONES Y RECOMENDACIONES

CAPITULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 DISCUSION DE LOS RESULTADOS OBTENIDOS

Luego de realizadas las pruebas experimentales de la tarjeta, y de haberse mostrado los resultados obtenidos, es posible realizar un análisis comparativo de las características y costo de la tarjeta DAS-128, para lo cual se han consultado precios y características de tarjetas que realizan funciones similares.

Se han consultado varios catálogos de diferentes proveedores, encontrándose tarjetas de adquisición de datos en los catálogos JAMECO y COLE PALMER. Adicionalmente se solicitó información de otras tarjetas en dos casas proveedoras.

A continuación se incluye una tabla resumen de costos y características:

Referencia	Precio	Entradas A/D			D/A O	Puertos digitales	Observaciones
		No.	rango	f			
PCL711S (Jameco/92)	289.95	8(12b)	±5V	25kHz	1	16I/O	
L-08338-00 (Cole-	2400	14(16b)	0-10	1Hz	x	11I,100	conectar al
L-08338-10 Palmer	3380	48(16b)	0-10	1Hz	x	11I,100	puerto
L-08338-20 91/92)	4650	96(16b)	0-10	1Hz	x	11I,100	paralelo
L-08109-25	312	4(15b)	±5V	7Hz	x	4 O oi	opto isolated
L-08109-27	559	4(8b)	±5V	22kHz	x	4 O oi	
L-08302-00	700	8(12b)	varia	10kHz	x	12I/O	escala selec-
L-08302-10	1170	16(12b)	"	"	x	16I/O	cionable:
L-08302-20	1170	8(16b)	1mV-10V	2.5k	x	12I/O	±25mV, 50mV,
L-08302-30	1870	16(16b)	"	"	x	16I/O	±250mV, 10V,
							500mV, ±5V,
L-08304-00	465		x		2(12b)	8I/O	0-10V 130k
L-08304-00	700		x		4(12b)	8I/O	0-10V 130k
L-08304-00	1170		x		8(12b)	8I/O	0-10V 130k
L-21090-00	939	16(12b)	±10.±.5V	50kHz	2(12b)	24I/O	5 ó 10 V
L-21090-00	1010	16(12b)	±10.±.5V	100kHz	2(12b)	24I/O	
ML16-P (ICS LTD)	1200	8(8b)	0-10	17kHz	2(8b)	8I, 8O	3 counter timer
PIO-32	870					16I,16O	480 mA

Como se puede observar, dado el número de puertos disponibles en la tarjeta DAS-128, sus características técnicas, detalladas en el numeral 3.4.2 y su costo, detallado en el acápite 3.4.1, resulta muy ventajosa respecto de las que se comercializan.

4.2 RECOMENDACIONES

Con el fin de dar continuidad al presente trabajo y considerando trabajos futuros que se puedan realizar en el campo de computadores personales, IBM compatibles, aplicados en sistemas de supervisión, adquisición de datos y control, se recomienda:

- Integrar la tarjeta DAS-128 a sistemas operativos multiusuario o multitarea, con el fin de que se posibilite realizar actividades adicionales en el mismo computador, además de las que realiza la tarjeta.

- Desarrollar bibliotecas que permitan la fácil operación e integración de la tarjeta desde el punto de vista del usuario.

- Desarrollar accesorios con el fin de facilitar el uso de la tarjeta DAS-128 en aplicaciones específicas, entre las cuales se pueden contar: tarjetas de conección, de relés, de entradas optoacopladas, amplificadores de termocuplas

y otros sensores, Redes de atenuación para medición de valores altos, acondicionadores para otros tipos de señales, etc.

- Debido a que el voltajes de referencia necesario para la operación del conversor D/A se ha tomado de la fuente de -5(V), es posible que los voltajes que se obtienen para un valor digital varíen de un computador a otro, por lo tanto, será necesario tener cuidado en el uso de dichas salidas realizando las correcciones necesarias en el programa del usuario.

- El presente trabajo pretende integrar los conocimientos básicos, necesarios para desarrollos posteriores de sistemas integrados en computadores PC compatibles; por lo tanto, las experiencias aquí descritas puede resultar de gran interés.

4.3 CONCLUSIONES

Luego de concluido el desarrollo de la tarjeta DAS-128, se pueden destacar los siguientes aspectos:

- Las pruebas de funcionamiento realizadas permiten afirmar que se ha conseguido construir una tarjeta cuyas características técnicas están al mismo nivel que las similares que se comercializan.

- La gran cantidad de puertos disponibles permite integrarla en sistemas muy grandes de supervisión y adquisición de datos.

- El costo de la tarjeta es bastante bajo, considerando la gran cantidad de puertos disponibles, en comparación con equipos de características técnicas similares e incluso inferiores, de fabricación extranjera.

- Las posibilidad de cambiar la dirección de los puertos de la tarjeta hace posible el uso de varias de ellas en un mismo computador, en caso de que así se requiera.

- El uso de señales normalizadas permite su fácil integración en cualquier tipo de sistema.

- Utilizada con fines didácticos, la tarjeta permite una fácil integración.

- Dada la gran cantidad de lenguajes disponibles para los computadores PC compatibles, se posibilita, al usuario de la tarjeta DAS-128, escoger aquel que sea más ventajoso para su aplicación específica. Se debe observar, sin embargo, que la realización de programas en Lenguaje C permite que los puertos disponibles funcionen a una velocidad muy superior a la que se puede obtener con Quick-Basic, y con velocidades ligeramente inferiores a las que provee la programación en Lenguaje Ensamblador.

- Debido a que en la tarjeta se disponen de puertos de entradas y salidas analógicas, la tarjeta permite ser integrada en sistemas de control muy fácilmente.

ANEXOS

ANEXO 1

INSTRUCCIONES DEL 8088

Instrucciones para movimiento de datos

Instrucción	Operandos posibles	Banderas O	D	I	T	S	Z	A	P	C	Operación
MOV	mem/reg1, mem/reg2										[mem/reg1] ← [mem/reg2] (No permite [mem] ← [mem])
	mem/reg, dato										[mem/reg] ← dato
	reg, dato										[reg] ← dato
	ac, mem										[ac] ← [mem]
	mem, ac										[mem] ← [ac]
	seg_reg, mem/reg										[seg_reg] ← [mem/reg]
	mem/reg, seg_reg										[mem/reg] ← [seg_reg]
XCHG	mem/reg1, mem/reg2										[mem/reg1] ↔ [mem/reg2] (No permite [mem] ↔ [mem])
	reg										[AX] ↔ [reg]
XLAT											[AL] ← [[AL] + [BX]]
LDS	reg, mem										[reg] ← [mem], [DS] ← [mem + 2]
LEA	reg, mem										[reg] ← mem (offset de la dirección)
LES	reg, mem										[reg] ← [mem], [ES] ← [mem + 2]
PUSH	mem, reg										[SP] ← [SP] - 2, [[SP]] ← [mem/reg]
	reg										[SP] ← [SP] - 2, [[SP]] ← [reg]
	seg_reg										[SP] ← [SP] - 2, [[SP]] ← [seg_reg]
PUSHF											[SP] ← [SP] - 2, [[SP]] ← [FLAGS]
POP	mem, reg										[mem/reg] ← [[SP]], [SP] ← [SP] + 2
	reg										[reg] ← [[SP]], [SP] ← [SP] + 2
	seg_reg										[seg_reg] ← [[SP]], [SP] ← [SP] + 2
POPF		x	x	x	x	x	x	x	x	x	[FLAGS] ← [[SP]], [SP] ← [SP] + 2
LAHF											[AH] ← Con las banderas SZxAxPxC (x = indeterminado)
SAHF						x	x	x	x	x	Las banderas SZxAxPxC ← [AH]

Instrucciones aritméticas

Instrucción	Operandos posibles	Banderas O	D	I	T	S	Z	A	P	C	Operación
ADC	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	x	x	x	[mem/reg1] ← [mem/reg1] + [mem/reg2] + [C] [mem,reg] ← [mem/reg] + dato + [C] [ac] ← [ac] + dato + [C]
ADD	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	x	x	x	[mem/reg1] ← [mem/reg1] + [mem/reg2] [mem,reg] ← [mem/reg] + dato [ac] ← [ac] + dato
INC	mem/reg reg	x				x	x	x	x		[mem/reg] ← [mem/reg] + 1 [reg] ← [reg] + 1
AAA		?				?	?	x	?	x	Ajuste ASCII de AL para la adición
DAA		?				x	x	x	x	x	Ajuste decimal de AL para la adición
SUB	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	x	x	x	[mem/reg1] ← [mem/reg1] - [mem/reg2] [mem,reg] ← [mem/reg] - dato [ac] ← [ac] - dato
SBB	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	x	x	x	[mem/reg1] ← [mem/reg1] - [mem/reg2] - [C] [mem,reg] ← [mem/reg] - dato - [C] [ac] ← [ac] - dato - [C]
DEC	mem/reg reg	x				x	x	x	x		[mem/reg] ← [mem/reg] - 1 [reg] ← [reg] - 1
AAS		?				?	?	x	?	x	Ajuste ASCII de AL para la substracción
DAS		?				x	x	x	x	x	Ajuste decimal de AL para la substracción
NEG	mem/reg	x				x	x	x	x	x	[reg] ← -[reg] + 1
MUL	mem/reg (8 bits) mem/reg (16 bits)	x				?	?	?	?	x	[AX] ← [AL] * [mem/reg] (sin signo) [DX][AX] ← [AX] * [mem/reg] (sin signo)
IMUL	mem/reg (8 bits) mem/reg (16 bits)	x				?	?	?	?	x	[AX] ← [AL] * [mem/reg] (con signo) [DX][AX] ← [AX] * [mem/reg] (con signo)
AAM		?				x	x	?	x	?	Ajuste ASCII para la multiplicación. AH y AL usados
DIV	mem/reg (8 bits) mem/reg (16 bits)	?				?	?	?	?	?	{[AH]←residuo,[AL]←cociente} de [AX]/[mem,reg] {[DX]←residuo,[AX]←cociente} de [DX][AX]/[mem,reg] (sin signo)
IDIV	mem/reg (8 bits) mem/reg (16 bits)	?				?	?	?	?	?	{[AH]←residuo,[AL]←cociente} de [AX]/[mem,reg] {[DX]←residuo,[AX]←cociente} de [DX][AX]/[mem,reg] (con signo)
CBW											[AH] ← [AL7]
CWD											[DX] ← [AX15]
AAD		?				x	x	?	x	?	Ajuste ASCII para la división

Instrucciones de comparación y lógicas

Instrucción	Operandos posibles	Banderas O	D	I	T	S	Z	A	P	C	Operación
CMP	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	x	x	x	[mem/reg1] - [mem/reg2] [mem/reg] - dato [ac] - dato
AND	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	?	x	x	[mem/reg1] ← [mem/reg1] AND [mem/reg2] [mem/reg] ← [mem/reg] AND dato [ac] ← [ac] AND dato
NOT	mem/reg										[mem/reg] ← -[mem/reg]
OR	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	?	x	x	[mem/reg1] ← [mem/reg1] OR [mem/reg2] [mem/reg] ← [mem/reg] OR dato [ac] ← [ac] OR dato
TEST	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	?	x	x	[mem/reg1] AND [mem/reg2] [mem/reg] AND dato [ac] AND dato
XOR	mem/reg1, mem/reg2 mem/reg, dato ac, dato	x				x	x	?	x	x	[mem/reg1] ← [mem/reg1] XOR [mem/reg2] [mem/reg] ← [mem/reg] XOR dato [ac] ← [ac] XOR dato

Instrucciones para manejo de cadenas

Instrucción	Operandos posibles	Banderas O	D	I	T	S	Z	A	P	C	Operación
LODS											[ac] ← [[SI]], [SI] ← [SI] ± δ (± depende de DF) (δ = 1 para 8 bits o δ = 2 para 16 bits)
MOVS											[[DI]] ← [[SI]], [SI] ← [SI] ± δ (misma observación)
STOS											[[DI]] ← [ac], [DI] ← [DI] ± δ (misma observación)
CMPS		x				x	x	x	x	x	[[SI]] - [[DI]], [SI] ← [SI] ± δ, [DI] ← [DI] ± δ(")
SCAS		x				x	x	x	x	x	[ac] - [[DI]]
REP	prefijo a instruc										usa CX como contador, repite la instrucción hasta que CX=0
REPE REPZ											sale del lazo si CX=0 o si se pone la bandera ZF luego de cualquier ejecución de la instrucción
REPNE REPNZ											sale del lazo si CX=0 o si se quita la bandera ZF luego de cualquier ejecución de la instrucción

Instrucciones que controlan el contador de programa

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
CALL	addr		[SP]←[SP]-2,[[SP]]←[PC], [SP]←[SP]-2,[[SP]]←[CS], [PC]←offset de dirección, [CS]←segmento de dirección
	disp16		[SP]←[SP]-2, [[SP]]←[PC], [PC]←[PC]+disp16
	mem(SEG+PC)		[SP]←[SP]-2,[[SP]]←[PC], [SP]←[SP]-2,[[SP]]←[CS], [PC]←[mem], [CS]←[mem + 2]
	mem/reg		[SP]←[SP]-2, [[SP]]←[PC], [PC]←[mem/reg]
RET			[PC]←[[SP]], [SP]←[SP]+2
			[PC]←[[SP]], [SP]←[SP]+2, [CS]←[[SP]], [SP]←[SP]+2
	disp16		
	disp16		[PC]←[[SP]], [SP]←[SP]+2+disp16
			[PC]←[[SP]], [SP]←[SP]+2, [CS]←[[SP]], [SP]←[SP]+2+disp16
JMP	addr		[PC]←offset de dirección, [CS]←segmento de dirección
	disp		[PC] ← [PC] + disp
	disp16		[PC] ← [PC] + disp16
	mem(SEG+PC)		[PC] ← [mem], [CS] ← [mem + 2]
	mem/reg		[PC] ← [mem/reg]

Instrucciones de salto condicional

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
JNBE(JA)	disp		Si ([C] OR [Z])=0, entonces [PC] ← [PC] + disp
JNB			
JNC			
JAE	disp		Si ([C] = 0, entonces [PC] ← [PC] + disp
JB			
JC			
JNAE	disp		Si ([C] = 1, entonces [PC] ← [PC] + disp
JBE			
JNA	disp		Si ([C] OR [Z])=1, entonces [PC] ← [PC] + disp
JE			
JZ			Si [Z]) = 1, entonces [PC] ← [PC] + disp
JG			
JNLE			Si([Z]=0 AND ([S]=[O]))=1, entonces [PC]←[PC]+disp
JGE			
JNL			Si [S] = [O], entonces [PC] ← [PC] + disp
JL			
JNGE			Si [S] <> [O], entonces [PC] ← [PC] + disp
JLE			
JNG			Si ([S]=[O] AND [Z]=0)=1,entonces [PC] ← [PC] + disp
JNE			
JNZ			Si [Z]=0, entonces [PC] ← [PC] + disp
JNO			Si [O]=0, entonces [PC] ← [PC] + disp
JNP			
JPO			Si [P]=0, entonces [PC] ← [PC] + disp
JNS			Si [S]=0, entonces [PC] ← [PC] + disp
JO			Si [O]=1, entonces [PC] ← [PC] + disp
JP			
JPE			Si [P]=1, entonces [PC] ← [PC] + disp
JS			Si [S]=1, entonces [PC] ← [PC] + disp
JCXZ			Si [CX]=0, entonces [PC] ← [PC] + disp

Instrucciones de lazo

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
LOOP	disp		[CX]←[CX]-1, Si [CX]=0, entonces [PC] ← [PC] + disp
LOOPZ LOOPE	disp		[CX]←[CX]-1,Si[CX]<>0 y [Z]=1,entonces[PC]←[PC]+disp
LOOPNZ LOOPNE	disp		[CX]←[CX]-1,Si[CX]<>0 y [Z]=0,entonces[PC]←[PC]+disp
JCXZ	disp		Si [CX]=0, entonces [PC] ← [PC] + disp

Instrucciones de control al procesador

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
CLC		0	[C] ← 0
CMC		x	[C] ← -[C]
CLD		0	[D] ← 0
CLI		0	[I] ← 0
STC		1	[C] ← 1
STD		1	[D] ← 1
STI		1	[I] ← 1
NOP			Ninguna operación
ESC	mem		bus de direcciones/datos ← [mem]
LOCK			cerrar el bus
WAIT			espera por señal externa para reiniciar operación
HLT			halt

Instrucciones de entrada/salida

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
IN	ac,DX		[ac] ← [puerto [DX]]
	ac,puerto		[ac] ← [puerto]
OUT	ac,DX		[puerto [DX]] ← [ac]
	ac,puerto		[puerto] ← [ac]

Instrucciones de interrupción

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
INT		0 0	[SP]←[SP]-2, [[SP]]←[FLAGS], [I]←0, [T]←0, [SP]←[SP]-2, [[SP]]←[CS], [SP]←[SP]-2,[[SP]]←[PC] [CS]←[vector(segmento)], [PC]←[vector(offset)]
INTO		0 0	Si [O]=1, [SP]←[SP]-2, [[SP]]←[FLAGS], [I]←0, [T]←0, [SP]←[SP]-2, [[SP]]←[CS], [SP]←[SP]-2,[[SP]]←[PC] [CS]←[00012h], [PC]←[00010h]
IRET		x x x x x x x x x	[PC]←[[SP]], [SP]←[SP]+2, [CS]←[[SP]], [SP]←[SP]+2, [FLAGS]←[[SP]], [SP]←[SP]+2

Instrucciones de rotación

Instrucción	Operandos posibles	Banderas O D I T S Z A P C	Operación
RCL	mem/reg, cuenta	x x	Rota mem/reg a la izquierda pasando por CF. cuenta puede ser 1 o el contenido de CL.
RCR	mem/reg, cuenta	x x	Rota mem/reg a la derecha pasando por CF. cuenta puede ser 1 o el contenido de CL.
ROL	mem/reg, cuenta	x x	Rota mem/reg a la izquierda, el bit más alto se copia en CF. cuenta puede ser 1 o contenido de CL
ROR	mem/reg, cuenta	x x	Rota mem/reg a la derecha, el bit más bajo se copia en CF. cuenta puede ser 1 o el contenido de CL
SAL SHL	mem/reg, cuenta	x x x x ? x x	Rota mem/reg a izquierda pasando por CF. Se pone 0 en el bit menos significativo. cuenta =1 o [CL]
SAR	mem/reg, cuenta	x x x x ? x x	Manteniendo el bit de signo, rota a derecha alimentando CF con el bit más bajo. cuenta =1 o [CL]
SHR	mem/reg, cuenta	x x x x ? x x	Rota mem/reg a derecha, pone 0 al bit más significativo, el bit más bajo se copia en CF.cuenta=1 o [CL]

ANEXO 2:

DIAGRAMAS ESQUEMATICOS DEL PC XT

ANEXO 3:

RUTINAS DE SERVICIO DEL DOS

Servicio	Descripción
00H	Terminar programa
01H	Leer entrada estándar con eco
02H	Mostrar carácter
03H	Entrada auxiliar
04H	Salida auxiliar
05H	Imprimir carácter
06H	Entrada/salida directa a consola
07H	Entrada directa a consola
08H	Leer entrada estándar
09H	Mostrar cadena
0AH	Entrada de teclado con "buffer"
0BH	Verificar estado de la entrada estándar
0CH	Vaciar "buffer" y leer entrada estándar
0DH	Reset de disco
0EH	Selección de disco
0FH	Abrir archivo

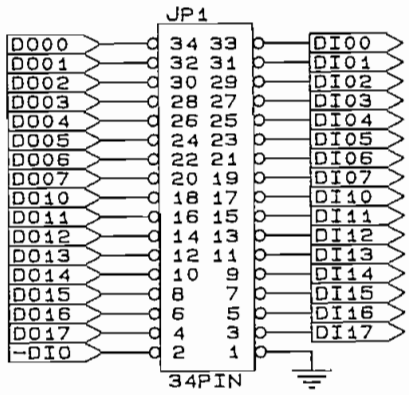
10H	Cerrar archivo
11H	Buscar la primera entrada
12H	Buscar la siguiente entrada
13H	Borrar archivo
14H	Lectura secuencial
15H	Escritura secuencial
16H	Crear archivo
17H	Renombrar archivo
19H	Disco corriente
1AH	Definir Disk Transfer Address
1BH	Obtener descriptor de tipo de disco
1CH	Obtener descriptor de tipo de disco para un drive específico
21H	Lectura aleatoria
22H	Escritura aleatoria
23H	Tamaño de archivo
24H	Definir registro relativo
25H	Definir vector
26H	Crear segmento de programa
27H	Lectura de bloque aleatoria
28H	Escritura de bloque aleatoria
29H	Pasar nombre de archivo
2AH	Leer fecha
2BH	Definir fecha
2CH	Leer hora
2DH	Definir hora
2EH	Cambiar la bandera de verificación
2FH	Leer Disk Transfer Address

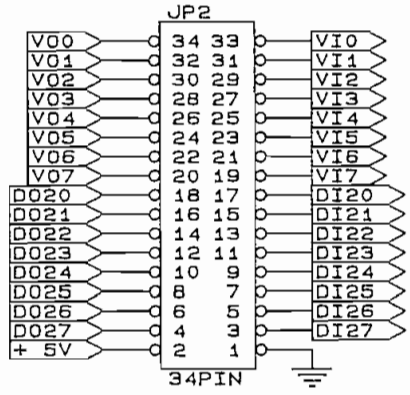
30H	Leer la versión del DOS
31H	Mantener proceso
33H	Controlar SHIFT-BRK (CTRL-C)
35H	Leer vector de interrupción
36H	Leer espacio libre en disco
38H	Retornar información dependiente del país
39H	Crear subdirectorio
3AH	Remover un Diectory Entry
3BH	Cambiar el directorio corriente
3CH	Crear archivo
3DH	Abrir un archivo o dispositivo
3EH	Cerrar el Handle de un archivo o dispositivo
3FH	Leer de archivo o dispositivo
40H	Escribir en un archivo o dispositivo
41H	Borrar un Directory Entry
42H	Mover un apuntador de archivo
43H	Cambiar atributos
44H	Control de entrada/salida para dispositivos
45H	Duplicar un handle de archivo
46H	Obligar un duplicado de handle
47H	Regresar texto del directorio corriente
48H	Memoria asignada
49H	Memoria asignada disponible
4AH	Modificar bloques de memoria asignada
4BH	Cargar y ejecutar un programa
4CH	Terminar un proceso
4DH	Sacar el código de retorno de un subproceso
4EH	Encontrar archivo semejante

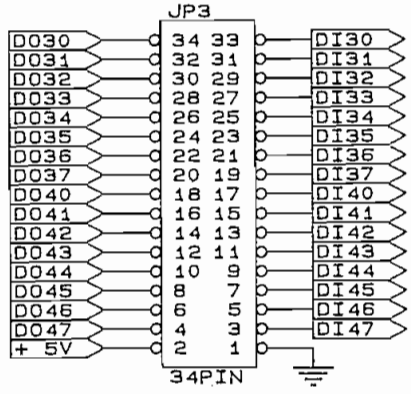
4FH	Paso por un directorio buscando archivo
54H	Retornar estado de Verify
56H	Mover una entrada a directorio
57H	Leer o cambiar fecha y hora de un archivo

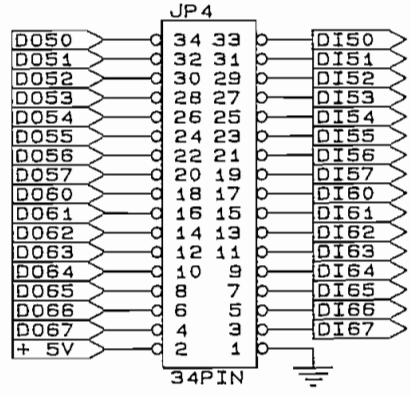
ANEXO 4:

DIAGRAMAS ESQUEMATICOS DE LA TARJETA DAS-128









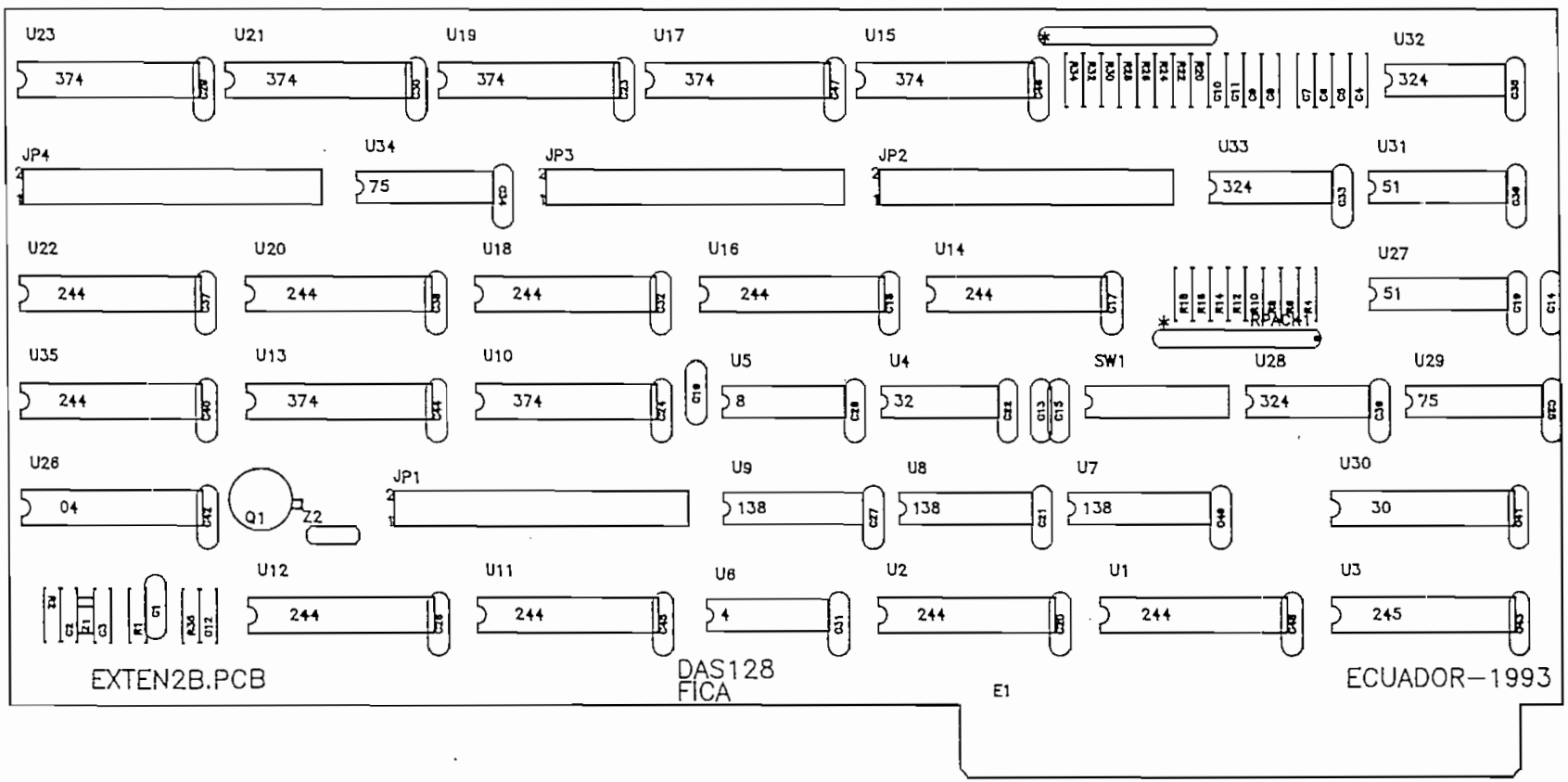
CONECTORES DE LA TARJETA DAS-128

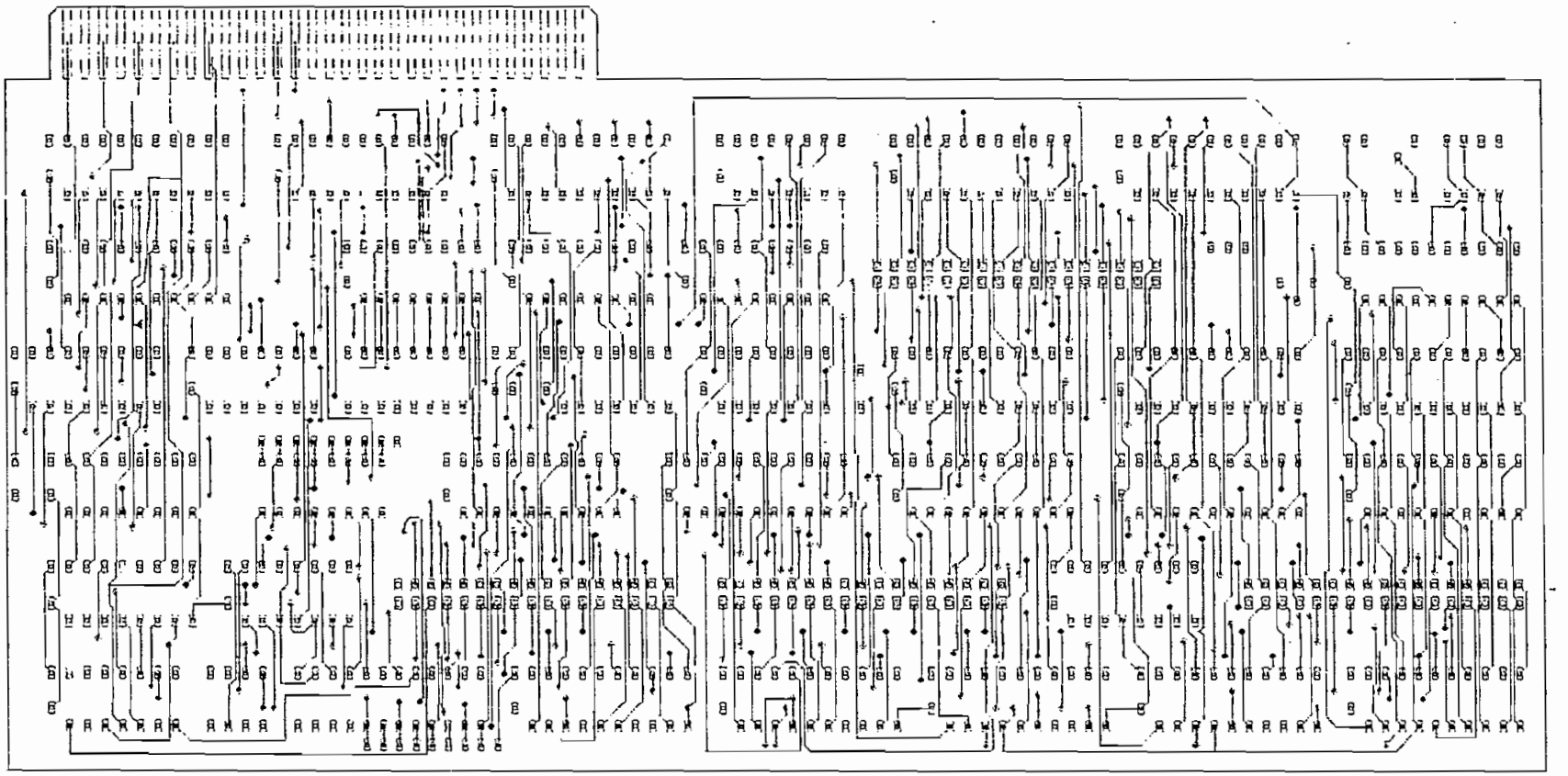
Title ESCUELA POLITECNICA NACIONAL
FACULTAD DE INGENIERIA ELECTRICA

Size	Document Number	REV

ANEXO 5:

CIRCUITO IMPRESO DE LA TARJETA DAS-128


U23
374
U21
374
U19
374
U17
374
U15
374
U32
324
JP4
U34
75
JP3
JP2
U33
324
U31
51
U22
244
U20
244
U18
244
U16
244
U14
244
RPACK1
U27
51
U35
244
U13
374
U10
374
U5
8
U4
32
SW1
U28
324
U29
75
U26
04
Q1
Z2
JP1
U9
138
U8
138
U7
138
U30
30
U12
244
U11
244
U6
4
U2
244
U1
244
U3
245
EXTEN2B.PCB
DAS128
FICA
E1
ECUADOR-1993



ANEXO 6:

MANUAL DE USUARIO DE LA TARJETA DAS-128

1. INSTALACION

Antes de instalar la tarjeta DAS-128, lea cuidadosamente las instrucciones para SELECCION DE LA DIRECCION BASE, del presente manual, y defina la dirección apropiada en concordancia con sus requerimientos.

Es muy importante definir correctamente la dirección que se asignará a la tarjeta DAS-128, con el fin de que no entre en conflicto con otra tarjeta instalada.

Si las direcciones de dos tarjetas de extensión que funcionan simultáneamente se sobreponen, el computador puede reportar errores, no operar, e incluso sufrir serios daños.

Para instalar la tarjeta DAS-128, realice lo siguientes:

a. Apague el computador

b. Remueva la cubierta del computador

c. Remueva una de las láminas, correspondiente a un slot libre, de las existentes para tarjetas de extensión.

d. Seleccione la dirección adecuada que se dará a la tarjeta (ver sección 3 del presente manual).

e. Instale apropiadamente los cables que utilizará en su aplicación (ver sección 4 del presente manual).

f. Instale la tarjeta en un slot de ocho bits para tarjetas de extensión.

g. Asegúrese de la correcta ubicación de la tarjeta y cables.

h. Encienda el computador y verifique el correcto funcionamiento de la tarjeta utilizando el programa TESTPORT.BAS (ver sección 5 del presente manual).

i. Si la operación es correcta, apague el computador y restituya la cubierta.

2. DESCRIPCION DE FUNCIONES DISPONIBLES

Puertos disponibles:

8 entradas analógicas con 8 bits de resolución

8 salidas analógicas con 8 bits de resolución

7 puertos digitales de 8 bits de entradas (*)

7 puertos digitales de 8 bits de salidas (*)

(*) Es posible disponer de un puerto bidirecional, a partir de uno de los puertos de entradas y uno de salidas, la explicación de ésta función se encuentra en la sección 4 del presente manual.

Descripción de la tarjeta:

La tarjeta DAS-128 debe ser instalada en un slot de ocho bits para tarjetas de extensión de un computador personal XT/AT/386 IBM compatible. Las fuentes de corriente continua necesarias para su operación se toman del slot, por lo tanto no se requiere de fuentes de alimentación externas. La tarjeta provee salidas de la fuente de +5(VDC) para usos externos.

La tarjeta DAS-128 se ha diseñado para facilitar su mantenimiento. No se requieren calibraciones, por lo tanto el usuario no debe realizar ningún tipo de ajuste interno en la tarjeta.

La tarjeta ocupa 15 direcciones consecutivas para direccionar los puertos disponibles, a partir de la dirección base

asignada por medio del dip switch existente. Las direcciones base que se pueden dar a la tarjeta son: 200h, 210h, 220h, 230h, 300h, 310h, 320h y 330h (ver sección 3 del presente manual, para una explicación detallada).

Entradas analógicas:

Se dispone de 8 entradas analógicas, las cuales se direccionan a un conversor A/D de ocho bits, por medio de un multiplexer analógico.

Los voltajes que se ingresan a cualquiera de las 8 entradas analógicas existentes deben estar en el rango entre 0 y 10 (VDC), correspondiendo el valor digital 00h a la señal de 0 (VDC) y FFh a la de 10 (VDC).

Salidas analógicas:

La tarjeta provee de 8 salidas analógicas, para lo cual utiliza un conversor D/A de 8 bits, cuya salida se conecta a un multiplexer analógico.

Cada una de las salidas del multiplexer analógico se conecta a un retenedor y luego a un amplificador operacional, encargado de duplicar el voltaje provisto por el conversor, con el fin de obtener voltajes entre 0 y 10 (VDC).

Entradas digitales:

Se dispone de 7 puertos de 8 bits de entradas digitales. Los voltajes permisibles en estas entradas son 0 y 5 (VDC), ya que se utilizan compuertas TTL LS en estos puertos.

Salidas digitales:

Se dispone de 7 puertos de 8 bits de salidas digitales TTL compatibles. Los valores digitales que se coloquen en estos puertos de salidas se mantendrán hasta que sean reemplazados por un muevo valor.

Programas de demostración:

Con la tarjeta se provee un diskette de 360 KB que incluye un programa de prueba, con el fin de definir la correcta operación de la tarjeta, y de algunos ejemplos para demostrar el funcionamiento de la tarjeta en diferentes usos.

3. ASIGNACION DE DIRECCIONES

En el Gráfico No. 1, que se adjunta, se puede observar la disposición de todos los componentes de la tarjeta DAS-128. Descrito como SW1 se encuentra un dip switch que dispone de 8 micro interruptores. Cada uno de ellos permite asignar a la

tarjeta una dirección base diferente, de acuerdo a la siguiente tabla:

switch ON	dirección base asignada
1	200h
2	210h
3	220h
4	230h
5	300h
6	310h
7	320h
8	330h

****

CUIDE QUE SOLO UNO DE LOS INTERRUPTORES SE ENCUENTRE EN LA POSICIÓN ON.

****

Una vez definida la dirección base, las direcciones de acceso a cada uno de los puertos disponibles en la tarjeta se ilustra a continuación:

Dirección	Puerto de entradas	Puerto de salidas
xx0h	Vi0 (analógico)	Vo0 (analógico)
xx1h	Vi1 (analógico)	Vo1 (analógico)
xx2h	Vi2 (analógico)	Vo2 (analógico)
xx3h	Vi3 (analógico)	Vo3 (analógico)
xx4h	Vi4 (analógico)	Vo4 (analógico)

xx5h	Vi5 (analógico)	Vo5 (analógico)
xx6h	Vi6 (analógico)	Vo6 (analógico)
xx7h	Vi7 (analógico)	Vo7 (analógico)
xx8h	Di0 (digital)	Do0 (digital)
xx9h	Di1 (digital)	Do1 (digital)
xxAh	Di2 (digital)	Do2 (digital)
xxBh	Di3 (digital)	Do3 (digital)
xxCh	Di4 (digital)	Do4 (digital)
xxDh	Di5 (digital)	Do5 (digital)
xxEh	Di6 (digital)	Do6 (digital)
xxFh	Di7 (digital)	Do7 (digital)

Se puede observar que una misma dirección es compartida por dos puertos, uno de entradas y uno de salidas. La tarjeta DAS-128 verificará automáticamente si la operación requerida por el programa es de entrada o salida con el fin de dirigirse al puerto correspondiente.

Cada uno de los puertos digitales dispone de 8 bits, numerados del cero al siete. Para identificar a cada uno de los bits, en los conectores disponibles en la tarjeta DAS-128, se ha colocado un número adicional que lo identifica. Así, por ejemplo, Do13 indicará que nos referimos al bit 3 del puerto digital de salida 1 (Do1).

En el Gráfico No. 1 se indica la ubicación de los conectores externos, los cuales se han identificado con las letras JP, existiendo 4 conectores: JP1, JP2, JP3 y JP4. En el mismo

gráfico se puede observar, además, la ubicación de los terminales 1 y 2 de cada conector.

En el Gráfico No. 2 se observa la distribución asignada a cada uno de los terminales de los cuatro conectores existentes.

Podemos ver que el terminal 1 de los cuatro conectores provee de la señal de referencia (GND), y el terminal 2 de los conectores JP2, JP3 y JP4 provee de una señal de fuente, de +5 (VDC), para uso externo. El uso del terminal 2 del conector JP1 se indica en la sección 4 del presente manual.

Como ejemplo, y refiriéndonos al Gráfico No. 2, podemos ver que el terminal 12 de JP3 provee la señal correspondiente al bit 3 del puerto digital de salidas 4 (Do4).

****
Para definir la dirección base, verifique que no exista superposición entre las direcciones asignadas a los puertos de la tarjeta DAS-128, y las direcciones utilizadas por otra tarjeta instalada en su computador.
****

A continuación se incluyen las direcciones, generalmente utilizadas en los computadores personales y, que NO pueden ser asignadas a la tarjeta DAS-128:

000h-00Fh Controlador DMA.

020h-021h	Controlador interrupciones.
040h-043h	Temporizadores.
060h-063h	Puertos teclado, cassette y parlante.
080h-083h	Registros página DMA.
200h-20Fh	Adaptador juegos (solo 200 usada).
278h-27Fh	Adaptador paralelo secundario.
2F8h-2FFh	Adaptador asíncrono secundario.
300h-31Fh	Tarjeta prototipos.
378h-37Fh	Adaptador paralelo primario.
380h-38Fh	Adaptador monocromático y paralelo impresora.
380h-3DFh	Adaptador color y gráficos.
3F0h-3F7h	Adaptador diskette.
3F8h-3FFh	Adaptador asíncrono primario.

4. OPERACION DEL PUERTO BIDIRECCIONAL

Con el fin de disponer de un puerto digital bidireccional de 8 bits, se deberá realizar lo siguiente:

Interconectar los bits correspondientes, de los puertos Di0 y Do0. Para ésto, se deberán interconectar Di00 con Do00, Di01 con Do01, ..., Di07 con Do07. Esta operación se realiza conectando los terminales 34 con 33, 32 con 31, ..., 20 con 19 del conector JP1.

Conectar el terminal 2 del conector JP1 (-Dio) a una señal digital TTL compatible que definirá la dirección en la que se realizan las transferencias (desde la tarjeta DAS-128 o hacia

ella). La señal que comande a -Dio puede provenir de un bit de cualquiera de los restantes puertos digitales de salida disponibles en la tarjeta DAS-128, o puede ser una señal externa.

Para indicar que los datos SALEN de la tarjeta DAS-128, se debe colocar un cero lógico (0 VDC) en la entrada -Dio. Para indicar que los datos ENTRAN a la tarjeta DAS-128, se debe colocar un uno lógico (+5 VDC) en la entrada -Dio.

Cuando no se requiera comunicación bidireccional, y se necesite utilizar el puerto de salidas Do0, verifique que la señal -Dio se encuentre conectada a GND (terminal 1 de todos los conectores). Una vez realizada la indicada conección, los puertos Do0 y Di0 operan como cualquiera de los restantes puertos existentes en la tarjeta DAS-128.

5. SOFTWARE

En todos los lenguajes de programación existen instrucciones o funciones que permiten ingresar o sacar datos a través de los puertos. Para el caso de la tarjeta DAS-128, no existe diferencia en las instrucciones utilizadas para comandar los puertos digitales de las utilizadas para comandar los puertos analógicos.

En BASIC, se dispone de las instrucciones:

DaTo=INP(PoRt)

que permite almacenar en la variable DaTo el resultado de leer la información existente en el puerto PoRt. Esta instrucción se utiliza para leer puertos de entrada.

OUT(PoRt),DaTo

que permite sacar por el puerto PoRt el contenido de la variable DaTo. Esta instrucción se utiliza para sacar datos a través de un puerto de salidas.

En Lenguaje C, se dispone de las funciones:

int inp(int PoRt);

Utilizada para ingresar datos desde un puerto de entradas.

int outp(int PoRt, int DaTo);

Utilizada para sacar DaTo a uno de los puertos (PoRt) de salidas.

En Lenguaje Ensamblador, se dispone de las instrucciones:

in ac, DX

Que permite ingresar al acumulador (AX para una instrucción de 16 bits o AL para una instrucción de 8 bits), el dato existente en el puerto cuya dirección se encuentra almacenada en DX.

out ac, DX

Que permite sacar por el puerto, cuya dirección está definida por contenido de DX, el dato existente en el acumulador (AX para sacar 16 bits y AL para sacar 8 bits).

El programa TESTPORT.BAS

En el diskette que se entrega con la tarjeta DAS-128 se encuentran algunos programas demostrativos del funcionamiento de la tarjeta, realizados en Quick-Basic, Lenguaje C y Lenguaje Ensamblador.

El programa TESTPORT.BAS, desarrollado en Quick-Basic, se utiliza para determinar la adecuada operación de los puertos disponibles en la tarjeta DAS-128. El programa prueba simultáneamente dos puertos, sean analógicos o digitales, uno de entradas y otro de salidas.

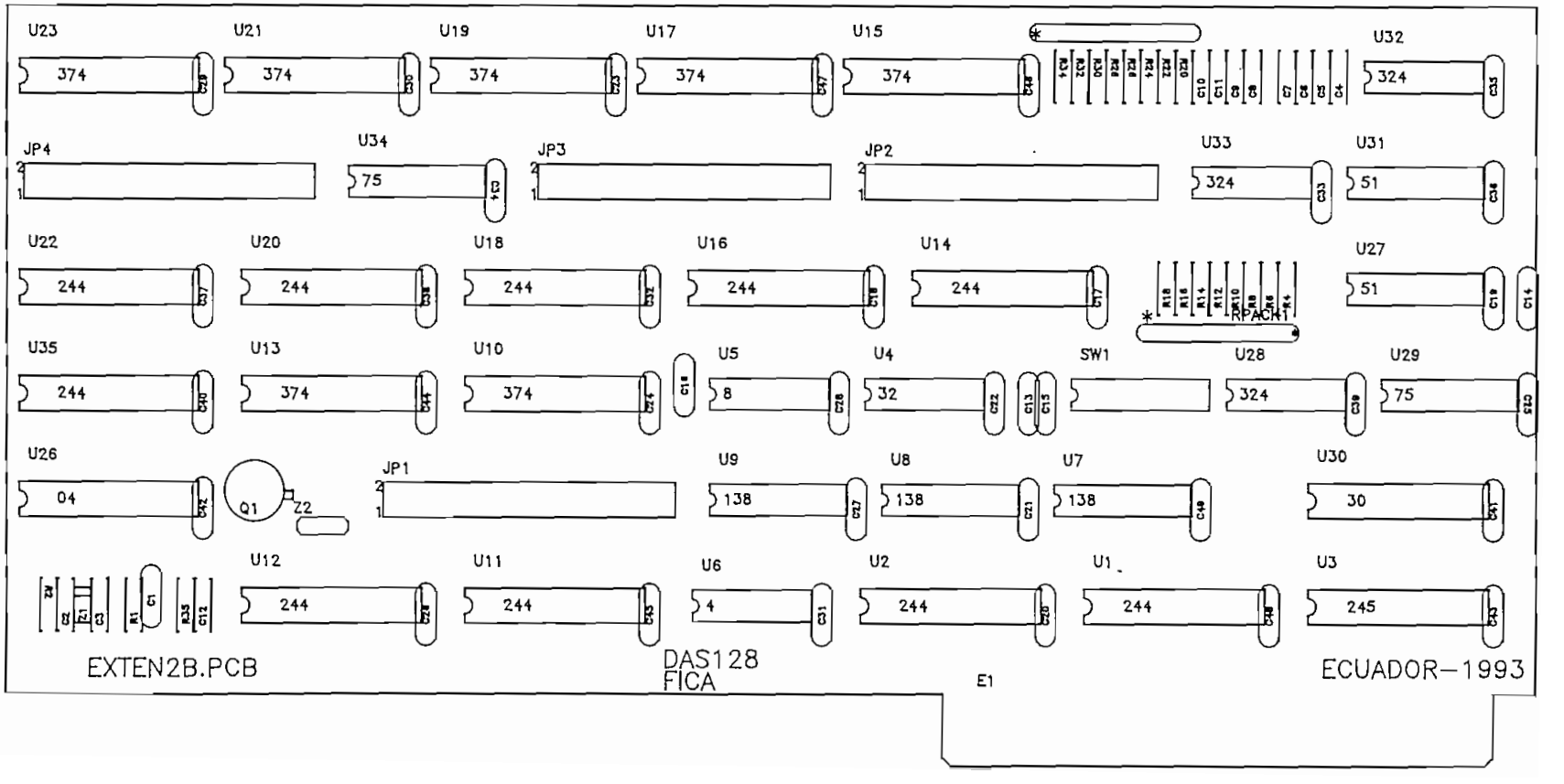
Para usar este programa, se deben interconectar externamente los dos puertos que comparten la misma dirección (uno de entradas y otro de salidas) y que se desean probar. Se debe observar que para probar uno de los puertos analógicos, se debe realizar un puente, mientras que, para probar uno de los puertos digitales, se deben realizar 8 puentes.

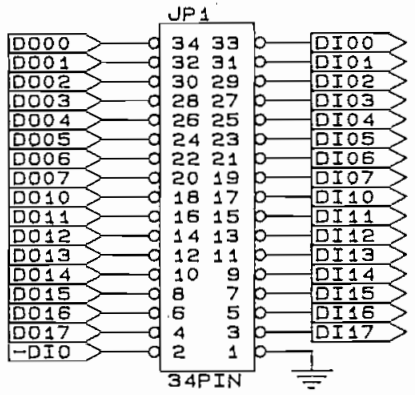
El programa pone datos, en orden ascendente (desde 0 hasta 255), en el puerto de salidas que se desea analizar (cuya dirección se pregunta al correr el programa); las salidas se

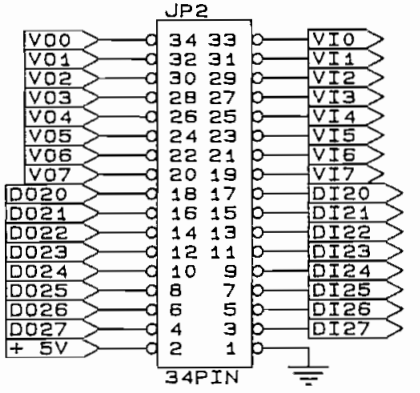
conectan externamente con el puerto de entradas y son leídas por el computador.

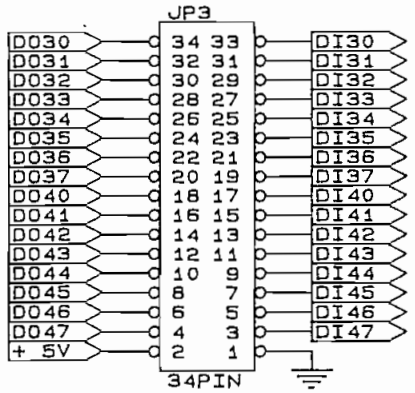
Los datos que el computador ha sacado y que ha recibido se despliegan en pantalla, con el fin de verificar su coincidencia.

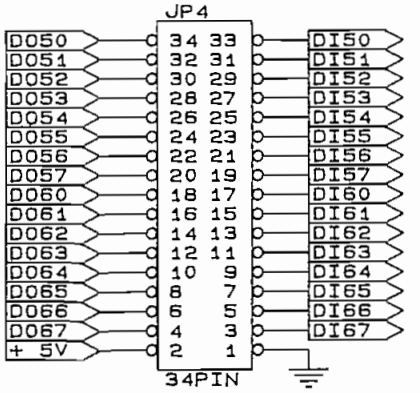
El programa tiene dos opciones para ser ejecutado. La opción paso a paso requiere que una tecla sea presionada antes de sacar el siguiente dato por el puerto; en la opción continua, los datos salen automáticamente.


U23
374
U21
374
U19
374
U17
374
U15
374
U32
324
JP4
U34
75
JP3
JP2
U33
324
U31
51
U22
244
U20
244
U18
244
U16
244
U14
244
RPACK1
U27
51
U35
244
U13
374
U10
374
U5
8
U4
32
SW1
U28
324
U29
75
U26
04
Q1
Z2
JP1
U9
138
U8
138
U7
138
U30
30
U12
244
U11
244
U6
4
U2
244
U1
244
U3
245
E1
EXTEN2B.PCB
DAS128
FICA
ECUADOR–1993


JP1
D000
D001
D002
D003
D004
D005
D006
D007
D010
D011
D012
D013
D014
D015
D016
D017
-DI0
34 33
32 31
30 29
28 27
26 25
24 23
22 21
20 19
18 17
16 15
14 13
12 11
10 9
8 7
6 5
4 3
2 1
DI00
DI01
DI02
DI03
DI04
DI05
DI06
DI07
DI10
DI11
DI12
DI13
DI14
DI15
DI16
DI17
34PIN


JP2
V00
V01
V02
V03
V04
V05
V06
V07
D020
D021
D022
D023
D024
D025
D026
D027
+ 5V
34 33
32 31
30 29
28 27
26 25
24 23
22 21
20 19
18 17
16 15
14 13
12 11
10 9
8 7
6 5
4 3
2 1
VI0
VI1
VI2
VI3
VI4
VI5
VI6
VI7
DI20
DI21
DI22
DI23
DI24
DI25
DI26
DI27
34PIN


JP3
D030
D031
D032
D033
D034
D035
D036
D037
D040
D041
D042
D043
D044
D045
D046
D047
+ 5V
34 33
32 31
30 29
28 27
26 25
24 23
22 21
20 19
18 17
16 15
14 13
12 11
10 9
8 7
6 5
4 3
2 1
DI30
DI31
DI32
DI33
DI34
DI35
DI36
DI37
DI40
DI41
DI42
DI43
DI44
DI45
DI46
DI47
34PIN


JP4
D050
D051
D052
D053
D054
D055
D056
D057
D060
D061
D062
D063
D064
D065
D066
D067
+ 5V
34 33
32 31
30 29
28 27
26 25
24 23
22 21
20 19
18 17
16 15
14 13
12 11
10 9
8 7
6 5
4 3
2 1
DI50
DI51
DI52
DI53
DI54
DI55
DI56
DI57
DI60
DI61
DI62
DI63
DI64
DI65
DI66
DI67
34PIN

BIBLIOGRAFIA

1. BRYAN, N., "Electronic Power Control and Digital Techniques", McGRAW-HILL, 1976

2. "Semiconductor Circuit Design", Texas Instruments Limited, 1974

3. LUNDH, Y., "Digital techniques for small computations", J. Brit. Inst. Radio Engineers, 1959

4. RAO, G., "Microprocessors and Microcomputer Systems", VNR, 1978

5. "IBM Technical Reference, Personal Computer XT"

6. MORSE, Stephen, "The 8086/8088 Primer", Hayden Book Company, Inc., 1982.

7. RECTOR, Russell - ALEXIS, Geroge, "The 8086 Book", Mc. Graw Hill, 1980.

8. "MS-DOS Operating System", Vol II, Texas Instrument Incorporated, No. 2243311-0001, Feb-84

9. LILEN, H., "Del microprocesador al micro-ordenador", Marcombo, 1978

10. "Design of microprogrammables systems", Signetics, 1974

11. GODFREY, Terry, "Lenguaje ensamblador para microcomputadoras IBM", Prentice-Hall Hispanoamericana, S.A., México, 1991.

12. EGGEBRECHT, Lewis, "Interfacing to the IBM personal computer", Howard W. Sams & Co., USA, 1987.

13. "Turbo-640 main board user's manual", Datatech Enterprises Co., Ltda.

14. "PIM-TB10-Z, 10 MHz turbo mainboard user's manual", Datatech Enterprises Co., Ltda.