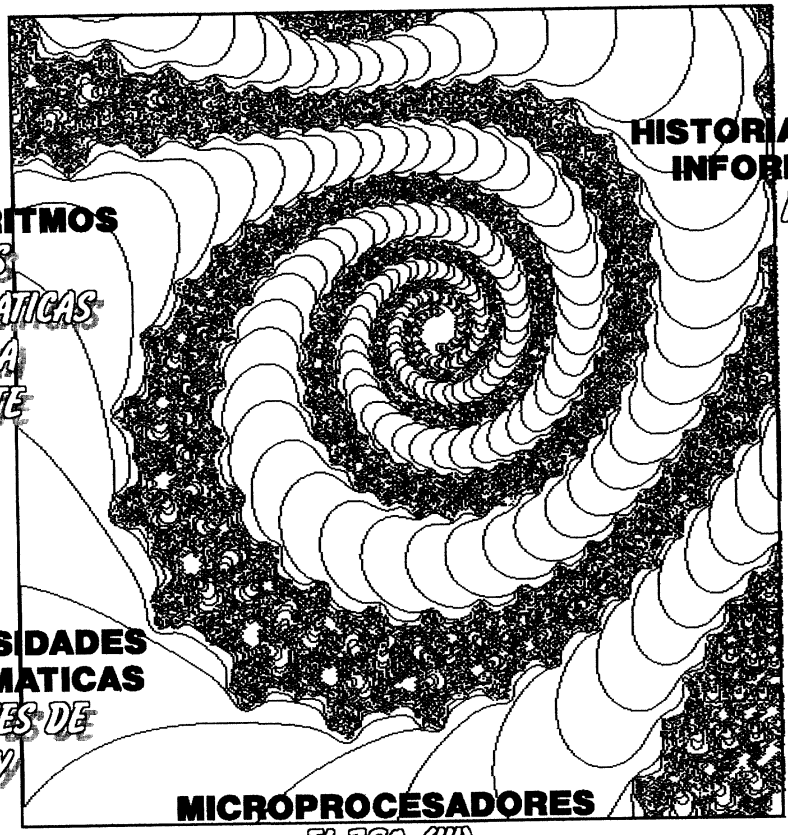
DATA BUS

Número 3 Segunda Epoca Oct. 1992

EDITA
ASOCIACION JUVENIL PARA LA INFORMATICA VIGUESA
AJIV



HISTORIA DE LA INFORMATICA
PARTE VI

ALGORITMOS
RUTINAS MATEMATICAS EN COMA FLOTANTE

CURIOSIDADES INFORMATICAS
LAS LEYES DE MURPHY

MICROPROCESADORES
EL Z80 (III)

DATA BUS

Número 3

Segunda época

Octubre de 1.992

DIRECTOR

Jose Manuel Suárez Pousa

REDACTORES

Jesús Cea Avión

Nacho Agulló Sousa

Telmo Lago Mediero

Jose Manuel Suárez Pousa

DISEÑO Y MAQUETACION

Jose Manuel Suárez Pousa

Jesús Cea Avión

ASOCIACION JUVENIL PARA LA INFORMATICA VIGUESA

AJIV

C/Caldas de Reis, 12-6 Izq.
36209 VIGO
Telf: (986) 23 18 35

Depósito legal
VG-87-90

¡¡¡ ATENCION !!!

Si deseas colaborar con nosotros en *DATA BUS* aportando ideas, críticas, artículos, etc., no tienes más que hacernos llegar tu trabajo. Para ello puedes enviárnoslo a la dirección de la asociación (ver página 2) o bien ponerte en contacto con la persona que te suministró este ejemplar.

¡¡¡ CONTAMOS CON VOSOTROS !!!

INDICE

EDITORIAL

Hola a todos un número más. Por fin, tras una larga espera, ***DATA BUS*** vuelve a estar en la calle. Ha pasado casi un año desde la última vez que nos vimos. Confiamos en que la espera haya valido la pena.

Se auguran grandes cambios: Nuevas secciones, nuevos colaboradores y, muy probablemente, una verdadera perioricidad. El estilo de las secciones técnicas tomará un matiz menos teórico con la inserción de pequeños fragmentos de código fuente como guía. Estos fragmentos serán escritos en C, por su transportabilidad, libertad, brevedad y amplia difusión. Aquellos lectores que desconozcan este lenguaje podrán seguir los ejemplos facilmente gracias al artículo en sí y a los comentarios incluidos.

Ya que esta revista es distribuida, sobre todo, en la *Escuela Técnica Superior de Ingenieros de Telecomunicación* de Vigo, estamos considerando la idea de incluir una sección fija denominada *"Comunicaciones Digitales"* en ***DATA BUS***. Esta sección trataría tanto el software (como hacer los programas para poder recibir/transmitir) como el hardware (lo que hay que conectar al ordenador para que el programa funcione). Por favor, hacednos llegar vuestras opiniones sobre el tema.

Respecto a la perioricidad de ***DATA BUS***, nos hemos fijado como objetivo fundamental publicar el siguiente número de la revista a finales de Febrero o principios de Marzo (tras los exámenes, por tanto). Que ***DATA BUS*** aparezca más a menudo depende totalmente de vosotros: No se puede sacar una revista si no hay suficientes artículos. Realmente preferiríamos tener que aumentar el número de páginas a tener que retrasar la fecha de salida por falta de material. La decisión es vuestra.

Hasta Febrero, pues. Esperamos que este número de ***DATA BUS*** os guste y, sobre todo, que os sea útil.

La Redacción

El Formato numérico de

Coma Flotante

Jesús Cea

Lo prometido es deuda. En este número vamos a estudiar la forma de implementar las rutinas matemáticas básicas para operar en coma flotante. Pero, ¿qué es la coma flotante? y, lo que es más importante, ¿cómo se trabaja con ella? Empecemos por el principio.

Recordando lo explicado en el último artículo sobre el formato de números enteros vemos que tiene varios problemas inherentes. El primero es que no se puede trabajar con decimales, aunque es posible hacerlo recurriendo a un truco que yo llamo fragmentación. De todos modos, aún utilizando "arreglos" de este tipo se tienen serias limitaciones sobre la precisión debido al reducido rango de valores utilizables. La técnica de fragmentación se basa en trabajar separadamente con la parte entera y la fraccionaria. Utilizando, por ejemplo, 16 bits para cada una, podemos operar con números de 5 cifras antes y otras tantas después del punto decimal. Si llamamos INT a la parte entera y FRAC a la fraccionaria, el número realmente representado es INT+FRAC/2^n, donde n es el número de bits de la parte fraccionaria (16 en nuestro caso).

Utilizando otra técnica que yo llamo RACIONALIZACION se simular números reales mediante fracciones de números enteros. A pesar de que con estos métodos no se tiene demasiada libertad son muy útiles en los casos en los que son utilizables, porque trabajar con estos formatos es realmente muy sencillo y rápido.

El otro problema grave es el reducido rango de valores con los que se puede trabajar. Aún utilizando 32 bits, sólo se pueden representar números de 10 cifras. Es posible aumentar el rango aumentando el número de bits. Así, con n bits pueden representarse $n*LOG_{10}(2)$ cifras (cifras en base decimal, se entiende). Pero si nos dedicamos a aumentar los bits pronto nos encontraremos trabajando con 200 bytes por número, algo muy lento y, sobre todo, extremadamente ineficaz. Números "tan sencillos" de representar como el 10^{10} pueden ocasionar tiempos de cálculo más que dilatados amén de ocupar muchísima memoria.

El formato de coma flotante soluciona de forma muy elegante ambos problemas. Todo número, en una base dada, puede representarse utilizando una mantisa y un exponente de la forma $MANT*BASE^{EXP}$. Veamos algunos ejemplos:

$15=1.5*10^{1}$

$230=2.3*10^{2}$

$5=5*10^{0}$

$0.7=7*10^{-1}$

En coma flotante, se dice que un número está normalizado cuando su mantisa tiene un valor entre 1 y base-1. En decimal sería entre 1 y 9.

¿Cómo se opera con números en coma flotante? En el caso del producto, se multiplican las mantisas y se suman los exponentes. Así, $(5*10^0)*(7*10^{-1})$ es $35*10^{-1}$. Sin embargo dicho número no está normalizado. Para ello corremos la coma a la izquierda y sumamos uno al exponente: $3.5*10^0$. Ahora ya está bien. En el caso de la división se dividen las mantisas y se restan los exponentes. Para normalizar el resultado se corre la coma a la derecha y se resta uno al exponente (en realidad, tanto en el producto como en la división, puede no ser suficiente con hacer un desplazamiento. En ese caso se realiza un bucle hasta que el número esté normalizado por fin).

Sumar y restar es un poco más complicado (al revés de lo normal) porque sólo se pueden sumar (o restar) potencias si tienen el mismo exponente (y la misma base). Entonces bastará con igualar los exponentes. Si elegimos como referencia al exponente mayor (lo usual): $(5*10^0)+(7*10^{-1})$ pasa a ser $(5*10^0)+(0.7*10^0)$. Ahora ya podemos sumar y da $5.7*10^0$, número que ya está normalizado. Fijaros en que la mantisa es la responsable de la precisión (el número de cifras significativas) y el exponente lo es del rango de valores representables.

Evidentemente todo esto está muy bien para el ser humano, pero a un ordenador le resulta bastante complicada trabajar en base 10. Dado que la fórmula general permite trabajar en cualquier base, veamos la forma de convertir todo lo explicado a binario. Curiosamente, al igual que ocurría en la primera parte de esta serie, al pasar a binario se simplifican notablemente las operaciones (¡qué casualidad!...). Veamos primero el formato de coma flotante utilizado por Commodore en todos sus ordenadores de 8 bits desde los Pet hasta el CBM128. Este formato es representativo para todos los 8 bits en general:

Para el exponente se utiliza un sólo byte en complemento a dos, lo que proporciona un rango de valores que cubre desde 2^{-128} hasta 2^{127} (en decimal, des de 10^{-39} hasta 10^{38} aproximadamente), un rango lo bastante amplio como para servir en la mayoría de las aplicaciones. En realidad Commodore no utiliza el complemento a dos, sino un offset. Para ello se suma a cada exponente el número $81 o 129 en decimal. En la práctica esto significa una simplificación en la manipulación de los exponentes, aunque el rango pasa a ser de -128 a 126. *¡¡¡ OJO, si el exponente es cero, se considera que el número también es cero !!!*. Por cierto, el resultado de sumar el exponente y el offset se denomina característica, pero nosotros llamaremos exponente tanto al exponente en sí, como al exponente "desplazado" para no complicar todavía más las explicaciones.

Como ya comenté la mantisa decide la exactitud de cálculo. Se utilizan 4 bytes (32 bits), lo que proporciona unas 9.6 cifras significativas. Por lo tanto tenemos una precisión de algo más de 9 cifras, suficiente para la mayoría de las aplicaciones. Debo hacer notar que esta precisión es independiente del módulo del número en sí, dado que este viene dado, principalmente, por el exponente.

Veamos ahora un pequeño truco para almacenar el signo en la mantisa, ¡¡ pero sin perder exactitud !!. Si recordáis lo que dije antes sobre la normalización, aplicándolo al sistema binario queda que el primer bit debe estar comprendido entre 1 y 1. Es decir, que debe ser SIEMPRE 1 (el número cero se distingue gracias al exponente). Podemos utilizar ese primer bit "redundante" diciendo que 0 significa positivo y 1, negativo.

Curiosamente Commodore almacena el signo utilizando este truco y, SIMULTANEAMENTE, en un byte separado (¿¿??). El byte separado tiene la ventaja de no necesitar operaciones para conocer el signo (0 positivo y $ff negativo), y el bit en la mantisa tiene la ventaja de ocupar menos memoria. Las rutinas matemáticas de Commodore sólo operan teniendo en cuenta el signo en un byte aparte, mientras que las variables BASIC se almacenan con el signo en la mantisa. Cuando el intérprete debe operar con variables, pasa sus valores a los acumuladores, SEPARA el signo y pone a uno el primer bit de la mantisa. Cuando hay que asignar el valor de un acumulador a una variable se pone el signo correcto en la mantisa. De esta forma se combinan eficazmente velocidad y memoria.

Todos conocéis los números racionales periódicos del estilo de 1/3. Este tipo de números aparecen en cualquier base dado que son algo implícito a las mismas matemáticas (en cualquier base hay números de este tipo). Lo que quizás no sepáis es que para cada número racional periódico existe una base (infinitas, en realidad) en la cual tiene un número FINITO de cifras. Consecuentemente un número de finitas cifras en una base puede ser infinitamente periódico en otra. Todo esto viene a cuento al intentar convertir el número 0.4 a coma flotante en base dos: $1.10011001100_2 * 2^{-2}$. Así pues, no se puede representar el número 0.4 de forma exacta. Hemos de interrumpir la sucesión en el dígito 31 después de la coma, obteniendo $1.10011001100110011001100110011_0 * 2^{-2}$.

Con el fin de aumentar en algo la exactitud Commodore no corta simplemente la sucesión, sino que la redondea. En nuestro caso la redondea hacia arriba porque el siguiente dígito era un uno. Así, el último bit de la mantisa debe ser un uno. Entre los bytes que Commodore dedica al cálculo en coma flotante existe uno denominado byte de redondeo, y que es utilizado durante los cálculos intermedios en operaciones complejas. Así pues, durante los cálculos se trabaja con 40 bits para reducir en lo posible la pérdida de precisión que se va acumulando al realizar operaciones consecutivas.

Como nota curiosa el número 1/3, periódico en decimal es, simplemente, 0.1 si

trabajamos en base 3... (el periódico sería ahora 10 en base 3...).

Bien, hasta aquí el formato de coma flotante utilizado en los 8 bits. Cuando aparecieron los 16 bits (y los coprocesadores matemáticos) se creó un formato estándar para permitir el intercambio de datos numéricos ya no entre aplicaciones de un mismo ordenador, sino entre ordenadores distintos: es la norma IEEE-754. Aparte de este estándar existen otros formatos muy extendidos como el IBM, utilizado en miniordenadores y en mainframes (los PC's normales utilizan IEEE-754), y el DEC (Digital Equipment Corporation), utilizado en los VAX y PDP de dicho constructor. No se está obligado a utilizar dicha norma para los cálculos internos, aunque sí es recomendable almacenar los números en este formato. El GFA-basic, por ejemplo, utiliza un formato numérico propio pero dispone de comandos para convertir números desde/hacia IEEE-754, para poder así guardar (o para leer) en disco números reconocibles por otros programas. Afortunadamente, la norma IEEE-754 está muy difundida entre los microordenadores, con lo que se puede decir que es realmente un estándar aceptado (los coprocesadores matemáticos de Motorola e Intel utilizan dicha norma, por ejemplo).

Veamos la norma IEEE-754 para números de doble precisión, que es el formato utilizado en el C, por ejemplo. Cada número consta de 8 bytes (64 bits). El primer bit (bit 63) es el signo de la mantisa (0 positiva y 1 negativa). Los bits 0-51 forman la mantisa. Hay que tener en cuenta que el bit antes de la coma no se almacena, porque es siempre un uno (recordad lo ya comentado). Así si aparece 10011 querrá decir, en realidad, 1.10011. De esta forma se tiene una precisión de 53 cifras binarias (o 16 cifras en base decimal). Los bits 52 a 62 constituyen el exponente. También aquí se utiliza un OFFSET, pero de 1023 ($3ff) esta vez. Así, un valor de 0 significa -1023 en realidad. El valor cero es reconocido porque tanto la mantisa como el exponente están a cero. En ese caso el valor del signo es irrelevante y puede ser tanto positivo como negativo. La norma IEEE-754 también permite representar los números más y menos infinito, y el valor NAN (not a number), pero confieso desconocer cómo. Tranquilos, ya investigaré sobre el asunto.

Hasta ahora vimos el formato de los números. Veamos ahora la forma de realizar las operaciones básicas de suma, resta, multiplicación, división y raiz cuadrada.

OPERACIONES EN COMA FLOTANTE

La verdad es que después de todo lo que he explicado no deberíais tener ningún problema para deducir vosotros mismos la forma de implementar las rutinas matemáticas básicas. De todos modos, voy a daros una explicación superficial:

1. SUMA: Como ya dije antes hay que igualar los exponentes. Si igualamos el exponente menor al mayor (lo normal), restamos los exponentes y corremos a la derecha la mantisa del exponente menor el número de veces que indique dicha resta. Aquí son posibles todo tipo de optimizaciones, pero eso lo dejo para vosotros. Una vez igualados, se suma (o se resta, según el signo) las mantisas. Si el resultado es cero, debemos señalar un número cero de la forma que mejor nos parezca. Si no es cero tendremos que normalizarla,

para lo cual corremos la mantisa a la derecha o a la izquierda las veces que sea necesario, y modificando en cada caso el valor del exponente. También tenemos que tener cuidado con el signo final.

2. RESTA: Para restar se utiliza la rutina de suma, pero invirtiendo previamente el signo del segundo operando. No hay ningún problema.

Pensando un poco se descubre un pequeño fallo en estas rutinas, aunque el verdadero problema se encuentra en el método en sí. El error se muestra claramente al restar números muy parecidos, periódicos y, por tanto, que no sean expresables de forma exacta en base dos. Restemos, por ejemplo, los números 100.1 y 100.01. Ninguno de los dos tiene una expresión binaria finita, por lo que se produce un "truncado" (en realidad bastaría con que uno fuera periódico). Suponiendo una mantisa de 16 bits, 100.1 pasa a ser 1100100.000110011, y 100.01 será 1100100.000000101. Restando en binario queda 0000000.000101110 o lo que es lo mismo, 0.0898437. Evidentemente hay un error más que notable. Una solución consiste en aumentar el número de bits de la mantisa, pero ello incrementa también los tiempos de cálculo y tampoco se arreglará gran cosa porque el error de redondeo persiste, aunque en menor medida.

Trabajando con mantisas grandes, aparentemente con pequeño error de redondeo, se dan constantemente casos en los que una de las mantisas pierde gran parte de sus bits (es decir, pierde precisión) durante las operaciones de igualación de exponentes y de normalización. Supongamos, por ejemplo, los números (ya en binario): 100.0000000000001 y 10.00000000000001.

Para poder restar hay que correr el sustraendo un bit a la derecha, PERDIENDOSE el bit menos significativo (pasa a ser 010.0000000000000). La resta da 10.00000000000010, cuando el resultado correcto sería 10.00000000000001. Aparece un error del orden de 6.1E-5. En este caso no es gran cosa, pero imaginaros lo que ocurriría si se perdieran más bits durante el proceso de igualación de exponentes... Lo malo es que no hay forma de evitar estos problemas a no ser que se utilice una técnica que yo llamo "RACIONALIZACION" en vez de la de coma flotante o la entera. Pero este será un tema para un futuro artículo. El famoso byte de redondeo que posee Commodore se utiliza precisamente para reducir estos errores de "igualación" (por contraposición a los errores de redondeo) y, aunque hace un buen trabajo, hay cosas que son inherentes al sistema en sí...

3. MULTIPLICACION: Se multiplican las mantisas, se suman los exponentes y se normaliza el resultado. Al multiplicar dos mantisas de n bits obtenemos un resultado de 2*n bits. Los n bits inferiores son descartados, lo cual constituye otra fuente de imprecisión. Sin embargo las imprecisiones en los productos y cocientes son cuantitativamente mucho menos importantes que las de redondeo e igualación en sumas y restas. Fijándonos un poco se ve que la normalización SIEMPRE corre la mantisa 2, 1 o ningún bit a la izquierda. (desde luego, esto clama a gritos una optimización...)

4. DIVISION: Se dividen las mantisas, se restan los exponentes y se normaliza el resultado. Aquí la única impre-

cisión se produce cuando la división de las mantisas da como resultado un número periódico, al estar obligados a efectuar un redondeo. Hay que tener mucho cuidado al efectuar la normalización del resultado: *¡¡los bits entrantes al efectuar los desplazamientos de la mantisa deben pertenecer al resto!!*

5. RAIZ CUADRADA: En el número anterior propuse la fórmula iterativa $X_{n+1}=(X_n+a/X_n)/2$, donde a es el número original y X_n representa sucesivas aproximaciones. Recordemos los problemas de convergencia explicados en el número anterior. Hay que tener mucho cuidado con los bucles infinitos. Una optimización en cuanto a velocidad consiste en dividir al principio el exponente por dos, lo que proporciona una buena aproximación inicial y reduce el número de iteraciones a efectuar. De esta forma, y trabajando con mantisas de 32 bits basta ejecutar 4 iteraciones para obtener una aproximación muy buena. Por cierto, que la mayoría de los ordenadores de 8 bits utilizaban la fórmula (unas 15 veces más lenta y mucho más imprecisa) EXP(LOG(a)/2).

Para efectuar las operaciones con las mantisas podéis utilizar los algoritmos dados en el número anterior. Evidentemente son posibles muchas optimizaciones tanto en velocidad como en precisión pero, en general, sólo son aplicables en casos concretos, y las optimizaciones generales son demasiado largas como para tratarlas ahora. De todos modos si alguien necesita unas super-rutinas matemáticas puede ponerse en contacto conmigo...

Hay que tener mucho cuidado con el cero. El cero es un número muy especial y se precisan comprobaciones específicas al principio de cada rutina tanto para evitar malgastar tiempo de proceso como para interceptar "errores" del tipo división por cero.

Recordad que en general (según el microprocesador utilizado) es más rápido sumar que multiplicar (aunque parezca lo contrario). ¿Es más lento multiplicar 20*9 que sumar 9 veces 20?¿Acaso me retracto de mis afirmaciones del último artículo? No, lo que ocurre es que 9*k=(8+1)*k= 8*k+k. Bueno, ¿y esto arregla algo? Pues sí, porque $8=2^3$, y para multiplicar un número en coma flotante por dos basta sumar uno al exponente... (de vez en cuando hay que pensar un poco). De esta forma pasamos de un producto a una suma. Optimizaciones de este tipo pueden salvar preciosos microsegundos para otras tareas más importantes...

Todas estas rutinas ya las había programado en el viejo CBM128, pero fue con el Atari (un Motorola 68000 a 8 Mhz.) y el programa FRACTALS como las llevé al límite. Las rutinas actuales ocupan unos 11 folios por ambas caras de lenguaje ensamblador, trabajan con mantisas de 32 bits y son capaces de ejecutar casi 15.000 iteraciones por segundo (cada iteración son unas 10 operaciones matemáticas). Adicionalmente FRACTALS dispone de otras rutinas con mantisas de 16 bits, menos precisas, capaces de realizar 26.000 iteraciones por segundo. Estas cifras no son tan grandes si tenemos en cuenta que la generación de cualquier pantalla necesita varios millones de iteraciones...

En el próximo número estudiaremos la forma de implementar las funciones SIN, COS, TAN, LOG, etc., y sus inversas. Os espero a todos.

El concepto de

Multimedia

Telmo Lago

En los últimos tiempos se viene oyendo un término que da la sensación de que se está hablando de superordenadores con unas capacidades un tanto "grandes" respecto a la perspectiva de un usuario medio. Me refiero al concepto de "multimedia".

No tiene nada de esotérico ni hay necesidad de asociarlo a complejos centros de cálculo y cosas por el estilo.

Se dice que un ordenador multimedia es aquel que puede procesar textos, imágenes estáticas, animaciones, datos, gráficos, video, audio y efectos especiales simultáneamente (por supuesto, todo realizándose en la misma máquina). En el caso de algunos ordenadores de tipo personal (CBM AMIGA, Atari ST, Apple, etc) es posible ampliarlos debidamente para que sean capaces de hacerlo. Lo que se persigue con esto es lograr un entorno de trabajo aún más amigable que los ya existentes, y al mismo tiempo dar una mayor cantidad de información al usuario.

EL DILEMA DE GUTEMBERG

Por separado, tanto la vista como el oído tienen un ancho de banda (en cuanto a la cantidad de información que pueden transmitir al cerebro) bastante amplio, y combinados pueden llegar a almacenar una cantidad de datos bastante respetable. En el lado opuesto se hallan el habla y la palabra escrita, debido a que en este caso los datos se procesan forzosamente de manera secuencial. A esto último le echa la culpa Marshall McLuhan, prestigioso periodista de que el hombre moderno piensa de una forma lineal. Según este periodista, la televisión ayudó en gran manera esta forma de pensamiento lineal.

Los multimedia se centran principalmente en casos de grandes volúmenes de información. Para ello, los equipos suelen disponer de unidades de CD-ROM. El Grupo de Aviación Comercial de Boeing dispone de este tipo de software para gestionar toda la información técnica de sus aviones 757 y 767. Los propios técnicos de mantenimiento se muestran muy satisfechos con el cambio.

Algunas casas de software están sacando herramientas para hacer que algunos de sus productos más populares puedan tener un entorno de tipo multimedia. La pena es que hay que ser un programador un tanto experimentado para poder sacarle partido

a este tipo de herramientas.

Además del CD-ROM, también se requiere una gran potencia gráfica para soportar imágenes lo más aproximadas a la realidad. IBM está trabajando en lo que se llama video digital interactivo (VDI). Es un harware especializado en digitalizar, comprimir, descomprimir y mostrar en cualquier dispositivo de visualización las imágenes registradas. De momento no se le augura mucho éxito mientras no mejoren los algoritmos de compresión, puesto que dicen que las velocidades de transferencia son demasiado pequeñas para lo que se pretende, y no se considerará comercializable hasta que sea posible alcanzar una velocidad de transferencia de 16 Megabytes por segundo.

Resultados del Gran Concurso de Programación

Guerra Nuclear

A finales del pasado año A.J.I.V. organizó un concurso de programación de Guerra Nuclear (Data Bus número 2, segunda época) en la Universidad de Vigo, bajo el patrocinio económico de la Xunta de Galicia. Cada participante podía presentar un máximo de cuatro programas combatientes, pagando una cuota de inscripción simbólica de 100 Pts. Durante las Navidades se efectuaron las eliminatorias y la Gran Final entre los programas luchadores, con la asistencia en directo de buena parte de los participantes. La entrega de premios se efectuó el día 11 de Enero de 1.992. A continuación se listan los nombres de los programas que llegaron a la Gran Final con su puntuación, su autor y el premio que obtuvieron:

XPHANTOM	18	PTS.	Sergio Antonio Cruces. 4' Teleco. Radiocasette digital y cintas
PHANTOM0	16	"	Sergio Antonio Cruces. 4' Teleco
POISON	15	"	Javier Rodriguez. 4' Teleco. Consola Nintendo Gameboy
POISONF	13	"	Javier Rodriguez. 4' Teleco
SENLTRS1	12	"	Alfonso Costas Puente
B100D	12	"	Marcos Alonso Ferreira. 2' Teleco. Walkman Sony y cintas
WARLOCK	12	"	Daniel Cruces Alvarez. 2' Teleco. Agenda electrónica Casio
PHANTOM5	10	"	Sergio Antoni Cruces. 4' Teleco
SENLTRS0	9	"	Alfonso Costas Puente
CEPNP1DS	8	"	Manuel Cameselle. 2' Industriales. Lote de cintas de cromo
BULL	5	"	Daniel Cruces Alvarez. 2' Teleco
BESBELL3	2	"	Carlos Díaz Sandoval. 1' Teleco

Los interesados pueden solicitar los resultados tabulados de todos los combates (5 folios), bases, programas y demás a A.J.I.V. Nuestro agradecimiento a todos los participantes y enhorabuena a los ganadores. ¡¡Hasta la próxima!!!.

El microprocesador

Z80 de Zilog (III)

Nacho Agulló

En la parte anterior vimos el aspecto técnico del Z80. En esta parte veremos su aspecto lógico.

A la hora de estudiar la estructura interna del Z80, podemos distinguir cinco grupos funcionales: La Unidad de Control, la Unidad Aritmético-Lógica, el Contador de Programa, el Registro de Instrucción y los Registros de Usuario.

La Unidad de Control

Es la responsable de la ejecución del programa, tomando de memoria las instrucciones, interpretándolas y ejecutándolas.

La Unidad Aritmetico-Lógica

Es la encargada de realizar, como su nombre indica, las operaciones aritméticas y lógicas. En realidad las operaciones que realiza la UAL son muy simples; no tiene capacidad de realizar siquiera una multiplicación sencilla.

Tampoco es posible realizar cálculos con múmeros grandes; la UAL sólo puede manejar simultáneamente 8 bits. Algunas instrucciones, mediante combinación de operaciones de la UAL, permiten realizar cálculos con números de 16 bits. En breve, las operaciones que la UAL puede realizar son las siguientes:

Incremento y decremento
Suma y resta
Operaciones Lógicas: AND, OR y XOR
Comparación
Operaciones con bits: Puesta a uno, puesta a cero, comprobación y rotación

El Contador de Programa

Se trata de un registro de 16 bits que contiene la dirección de la instrucción de programa a ejecutar. Es incrementado después de cada lectura hasta tantas posiciones de memoria como mide la instrucción ejecutada.

El Registro de Instrucción

Se trata de un registro de 8 bits que se utiliza para almacenar una copia de la localización de memoria correspondiente a la dirección indicada por el Contador de Programa. Sobre esta copia se realiza después el trabajo de interpretación de la Unidad de Control.

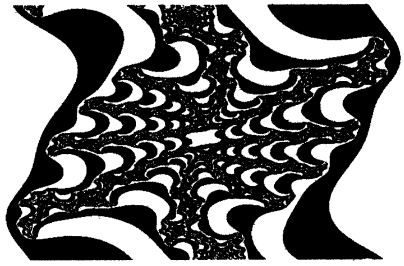
Cuando la instrucción consta de más de un byte, cada uno de los bytes de la instrucción (hasta cuatro) es copiado al Registro de Instrucción por orden, incrementándose el Contador de Programa cada vez.

Los Registros de Usuario

Además del Contador de Programa y del Registro de Instrucción, el Z80 cuenta con 21 Registros de Usuario, entre los que se pueden distinguir: Registros Principales, Registros Indice, Registro Puntero de Pila, Registro de Direcciones de Página de Interrupción y Registro de Regeneración.

Registros Principales

Son dos juegos de 8 registros de 8 bits, que a su vez se divide en Registros de Uso General y Registros Especiales. El microprocesador utiliza sólo uno de estos dos juegos, conservando el otro para su uso posterior. Al juego de registros que no está en uso se le denomina "Registros Alternativos". El cambio de juego se realiza atendiendo a una instrucción del programa, que es diferente según sea para cambiar el juego de los Registros de Uso General o el de los Registros Especiales.



Los Registros de Uso General se denominan B, C, D, E, H y L; y B', C', D', E', H' y L' si son alternativos. Es posible tomarlos agrupados por parejas, desempeñando la función de registros de 16 bits, denominándose entonces BC, DE y HL, correspondientemente a los nombres de los registros de 8 bits que componen el "registro doble". En general el juego de instrucciones del microprocesador Z80 dispone de instrucciones que permiten realizar con estos "registros dobles" de 16 bits prácticamente las mismas operaciones que con los registros de 8 bits.

Los Registros Especiales son dos, de 8 bits al igual que los Registros de Usp General: el Acumulador (propiamente designado por la letra A), y el registro de indicadores, designado por la letra F. Similarmente a los Registros de Uso General, los registros alternativos especiales se denominan A' y F', y tambien similarmente, los registros especiales son agrupables en un registro doble, denominado AF.

El Acumulador es el registro más importante del Z80, puesto es el registro utilizado por lo general para almacenar los resultados de las operaciones realizadas por la UAL. Hay muchas operaciones que se pueden realizar utilizando el acumulador, y también muchas que se pueden realizar solamente mediante él.

El Registro de Indicadores se utiliza para contener seis indicadores referentes a condiciones lógicas y aritméticas del microprocesador. Estos indicadores están asignados a un bit cada uno, quedando sin indicador asignado dos bits del registro. Por su complejidad, estudiaremos las características de estos

indicadores al final de los demás registros.

Registros Indice

Son dos registros de 16 bits denominados IX e IY. Además de poder realizar con ellos prácticamente las mismas operaciones que con los "registros dobles", estos dos registros tienen una función especial: servir de base para direccionamiento indirecto. Así, es posible direccionar cualquier posición de memoria situada en un entorno de 256 bytes centrado en la dirección indicada por un registro índice, simplemente mediante un byte complementario, complementado a dos.

Registro Puntero de Pila

El Z80 está preparado para manejar un vector de información tipo LIFO (*LAST IN, FIRST OUT*), llamado de forma más simple como "pila", por motivo de que los datos se "apilan" en la memoria, recuperándose por orden inverso al de la introducción, es decir: el primero que se recupera es el último introducido. Esta pila parte de una dirección inicial ("fondo de la pila") y crece en la memoria "hacia abajo", esto es, los elementos de información son almacenados uno a continuación de otro, en direcciones correlativas descendentes. El Registro Puntero de Pila indica la dirección en que está almacenado el último elemento introducido. Se trata por lo tanto de un registro de 16 bits. Dado que el juego de instrucciones del Z80 sólo permite almacenar y recuperar de la pila registros dobles, las operaciones de introducción y recuperación suponen restar o sumar dos unidades al Puntero de Pila. Adviértase que la anidación de subrutinas la realiza el Z80 mediante esta pila, almacenando en ella las direcciones de retorno al ejecutar las llamadas a las subrutinas, para recuperarlas al ejecutar los correspondientes retornos. Este almacenamiento se produce de forma indiferenciada al de datos, motivo por el cual para que el microprocesador no termine interpretando los datos como direcciones o viceversa, es necesario tener al programar especial precaución en que la pila esté dispuesta justo antes de retornar desde una subrutina de forma idéntica al momento después de haberla llamado.

Registro de Direcciones de Página de Interrupción

Es un registro de 8 bits, utilizado en el Modo de Interrupción 2. En la ejecución de una Interrupción No Enmascarable (*NMI*), el microprocesador realizará el equivalente a una llamada. La dirección a la cual se producirá la llamada se obtendrá de la siguiente forma: Los 8 bits bajos se tomarán del *bus* de datos, proporcionados por el dispositivo de Entrada/Salida que ha solicitado la interrupción; y los 8 bits altos vendrán dados por el Registro de Direcciones de Página de Interrupción, denominado Registro I. Esta interrupción se puede

utilizar para ejecutar programas alternativos en tiempo "simultáneo", tales como un accesorio que muestre un reloj en un ángulo de la pantalla, etc.

Registro de Regeneración

Es un registro de 7 bits, utilizado para refrescar memorias dinámicas, y se le denomina Registro R. Se utiliza para generar parte de la dirección de la memoria dinámica a partir de la cual se va a aplicar un proceso de recarga para mantener la información contenida en aquélla. Funciona como un contador, incrementándose automáticamente después de cada instrucción. Adviértase que es posible manipular su valor, interfiriendo el proceso de refresco de la memoria.

Especificaciones Respectivas al Registro de Indicadores

Cada vez que la UAL lleva a cabo una operación, los indicadores del microprocesador, que están contenidos en su totalidad en el Registro F, son actualizados. A continuación examinaremos el Registro F detalladamente, bit a bit, numerándolo desde 0 a 7, del bit menos significativo al más significativo.

Bit 0: Indicador de acarreo (*C*). Tras operaciones de suma y resta, y también de incremento y decremento con registros de 8 bits, será puesto a uno si ha habido "llevada" y puesto a cero en caso contrario. Tras operaciones de comparación, será puesto a uno si el valor que se compara con el del acumulador es mayor que éste, y puesto a cero en caso contrario. Tras operaciones lógicas, será puesto a cero siempre y tras operaciones de rotación variará dependiendo de la operación y de los valores manejados.

Bit 1: Indicador de Suma y Resta.

Bit 2: Indicador de Paridad/Desbordamiento (*P/O*). Este es un indicador de función doble; se utiliza como indicador de desbordamiento tras las operaciones aritméticas, y como indicador de paridad tras las operaciones lógicas. A continuación veremos ambos apartados. En las operaciones aritméticas con registros de 8 bits, que son suma, resta, suma con acarreo, resta con acarreo, incremento, decremento y comparación (a efectos del valor del indicador se considera como una resta), el indicador se pone a uno si hay "llevada" en *aritmética complementada a dos*, y a cero en caso contrario. Con registros de 16 bits, se pone a uno si se produce llevada en las operaciones de suma y resta con acarreo.

En las operaciones lógicas, y en las rotación de más de un byte, se contará el número de bits puestos a uno en el registro con el cual se ha realizado la operación. En el caso de que el número de unos sea par, el indicador será puesto a uno, y a cero en caso contrario.

Bit 3: No utilizado.

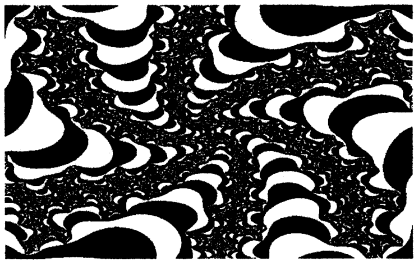
Bit 4: Indicador de semiacarreo.

Bit 5: No utilizado.

Bit 6: Indicador de cero(*Z*). Se pone a uno tras una operación, con registros de 8 bits, de incremento, decremento, suma, resta, suma con acarreo, resta con acarreo, AND, OR,

XOR o rotación (si la instrucción es de más de un byte) si el registro con el que se realiza la operación toma el valor cero. Se pone a cero en caso contrario. También con registros de 16 bits y las operaciones de suma y resta con acarreo el indicador se pone a uno si el resultado es cero, o a cero en caso contrario.

Las operaciones de carga de registros no afectan a este indicador, con



la salvedad de las utilizadas para cargar el acumulador con los valores de los registros R e I, tras las cuales, al igual que con las demás operaciones, el indicador se pone a uno si el acumulador toma el valor cero, y se pone a cero en caso contrario.

En las instrucciónes de búsqueda, tanto automática como no automática, se pone a uno cuando el valor del acumulador coincide con el de la posición de memoria en la que se está buscando, y se pone a cero en caso contrario. En las instrucciones de transferencia de datos, se pone a uno cuando el registro B toma el valor 1, y se pone a cero en caso contrario. En las instrucciones de comprobación de bits, se pone a uno si el bit comprobado vale 0, y se pone a cero en caso contrario.

Bit 7: Indicador de signo (S). En aritmética complementada a dos, el bit de la izquierda se corresponde siempre con el signo: El valor 1 para el signo negativo, y el 0 para el signo positivo. Así, el bit de la izquierda se convierte en el "bit de signo", para todo valor complementado a dos. El indicador de signo pues toma su valor del registro en consideración, simplemente copiando del bit de la izquierda del mencionado registro.

Las operaciones tras las cuales el indicador se actualizará son: Con registros de 8 bits, incremento, decremento, suma, resta, suma con acarreo, resta con acarreo, comparación, AND, OR, XOR, rotación si la instrucción es de más de un byte, instrucciones de carga del acumulador con los registros R e I, instrucciones de búsqueda y de transferencia de datos; con registros de 16 bits, instrucciones de suma y resta con acarreo.

Y con esto termina el vistazo lógico al Z80.

Bibliografía

- *Gran Enciclopedia Informática.*
  Ediciones Nueva Lente.

- *Enciclopedia Mi Computer.*
  Editorial Delta.

- *Construya su propia computadora basada en el Z80.*
  Steve Ciarcia. Bytes Books.
  Editorial McGraw Hill.

- *Understanding your Spectrum.*
  Dr. Ian Logan.
  Melbourne House Publishers.

Las leyes de

Murphy

Jesús Cea

Si sois fieles seguidores de DATA BUS conoceréis mi facilidad para invocar a "Murphy" en mis artículos. Si os habéis sentido intrigados por esas extrañas alusiones, ésta es la vuestra.

Así que por una vez, y sin que sirva de precedente, vamos a relajarnos un poco y a reir a carcajadas con lo que sigue.

Hoy en día solemos considerar que la Ciencia tiene una respuesta para cada pregunta que podamos formular. Aunque esto no es comunmente aceptado por los científicos profesionales (sobre todo tras el descubrimiento del caos determinista), al hombre de la calle se lo han hecho creer de una manera o de otra y, en unos tiempos en los que se tiende a valorar las cosas que nos rodean desde un punto de vista estrictamente práctico, no dejan de sorprendernos ciertos "comportamientos" de objetos supuestamente inanimados y la malévola inoportunidad de ciertas "casualidades".

La Ley de Murphy fue el primer paso que dio un ingeniero aeronáutico estadounidense en 1.949 para "sistematizar" (?) estos imponderables que, por serlo, escapan a la Ciencia y que, evidentemente, no pueden ser tratados seriamente ni por la Lógica ni por la Mística.

La Ley de Murphy dice, simplemente, *"Si algo puede ir mal, irá mal"*, lo cual no parece gran cosa a primera vista..., hasta que uno, naturalmente escéptico, empieza a ensayar su aplicación. Entonces los resultados son pasmosos.

Desde 1.949 ha corrido mucha tinta y, en lo que respecta a la Ciencia, las cosas han avanzado a una velocidad no menor que la causada por una enérgica patada en el trasero. Con ello, la Técnica también se ha complicado, creciendo las oportunidades de aplicación de esta Ley. También con el paso del tiempo se han multiplicado los filósofos espontáneos y con ellos los corolarios y adaptaciones a diferentes aspectos del trabajo de cada día.

Por la parte científica, los *"Científicos por la Exclusión de la RazóN"* (C.E.R.N.) dicen haber descubierto nada más y nada menos que *"los 15 principios básicos de la ciencia"*:

1. Ley de Murphy: Si algo puede ir mal, irá mal. En algunas fuentes oficiosas se conjetura con la siguiente generalización: Si algo puede ir mal, no sólo irá mal, sino que lo hará en el momento más inoportuno.

2. Teorema de Patrick: Si el experi-

mento funciona es señal segura de que se está utilizando un procedimiento equivocado.

*3. **Constante de Skinner:*** Es aquel valor tal que sumado, restado, multiplicado o dividido por el resultado obtenido da la cantidad deseada.

*4. **Postulado de los 5 dedos:*** La experiencia aumenta proporcionalmente con el número de aparatos que uno estropea.

*5. **Ley de Flaple:*** Todo objeto inanimado, prescindiendo de su composición o configuración, se puede esperar fabricar siempre de una manera totalmente insospechada por razones que nos son enteramente oscuras o, más bien, completamente misteriosas. A esta ley también se la conoce bajo el nombre de *"Teorema de la perversidad de los objetos inanimados"*.

*6. **Axioma de Allen:*** Cuando todo falla es que hay que leer las instrucciones.

*7. **Principio de las piezas dispersas:*** La accesibilidad, cuando se recuperan las piezas pequeñas que se han caído de la mesa de trabajo, varía proporcionalmente con el tamaño de las piezas e inversamente con su importancia para completar el trabajo empezado.

*8. **Corolario de la compensación:*** Un experimento puede ser considerado un éxito si no más del 50% de las medidas efectuadas deben ser descartadas para obtener cierta correspondencia con la teoría.

*9. **Ley de Gumperson:*** La probabilidad de que ocurra un determinado suceso es inversamente proporcional a lo deseable que éste sea.

*10. **Principio del material pedido:*** Los suministros necesarios para el experimento de ayer deben ser pedidos no más tarde de mañana a mediodía.

*11. **Principio principal:*** Por definición, cuando uno investiga lo desconocido no sabe lo que encontrará.

*12. **Regla de Ketterin:*** No funciona, pero no funciona por una razón diferente de la que uno piensa que no funciona.

*13. **Factor de Futilidad:*** Ningún experimento es nunca un completo fracaso ya siempre puede servir como un mal ejemplo.

*14. **Ley de Anderson:*** Nunca se estropea nada de lo que uno tiene recambio.

*15. **Postulado de Puig:*** Lo difícil se hace cada día. Lo imposible cuesta un poco más.

Por otra parte (por la parte técnica) los ingenieros tampoco quisieron quedarse atrás, y su organización,

conocida bajo el nombre clave de "*Ingenieros Expertos En Entelequias*" (I.E.E.E.) han advertido a sus numerosos miembros de que:

1. En cualquier fórmula las constantes deben ser tratadas como variables.

2. Cuando más de una persona es responsable de un error de cálculo, ninguna tiene la culpa.

3. Cuando se establezca un índice de seguridad, un idiota ingenioso calculará rapidamente un método para excederlo.

4. La pieza que se han olvidado de enviarnos es la única absolutamente imprescindible para que el aparato funcione.

5. Las partes que precisan ajuste o mantenimiento periódico son las menos accesibles.

6. Cualquier aparato funciona mejor cuando se enchufa.

7. No hay que forzar nunca un aparato; basta con utilizar un martillo más grande.

8. En tareas de investigación, si los hechos contradicen la teoría, los hechos deben ser descartados de inmediato.

9. Es imposible hacer nada a prueba de tontos porque los tontos son muy ingeniosos.

10. Bajo las condiciones más rigurosamente controladas de presión, temperatura, volumen, humedad y demás variables, el organismo reaccionará como le venga en gana.

11. Si puede deslizarse un error en los cálculos, desde luego que lo hará, y de tal forma que será necesario rehacer de nuevo todas las operaciones.

12. En todos los cálculos el valor que creíamos más correcto resultará ser el causante de todos los errores.

13. La coma decimal se las arregla para desplazarse por su cuenta y colocarse en el peor sitio.

14. Los conductores que se emplean para el cableado, que se habían cortado previamente a su longitud correcta, en la práctica resultan ser siempre demasiado cortos.

15. Se obtiene el más variado surtido de nudos cuando se deshace un rollo de cable que con anterioridad se había enrollado con todo cuidado.

16. Las tolerancias mecánicas se acumulan siempre en el mismo sentido de tal forma que su ensamblaje resulte lo más difícil posible.

17. Un conjunto de aparatos idénticos probados en igualdad de condiciones, en cuanto entren en servicio se comportarán de manera totalmente diferente.

18. Si se necesitan 100 resistencias para montar un circuito, jamás habrá más de 99 en el almacén.

19. Un componente o pieza será tanto más difícil de encontrar cuanto más urgente sea la necesidad de obtenerlo.

20. Si se precisa una resistencia de un valor preciso y determinado, no se

encontrará. Además, resultará imposible obtener el valor deseado mediante combinaciones en serie o paralelo de las resistencias de las que se disponga.

21. Cuando se cae una herramienta siempre aterriza donde ésta pueda causar el mayor desperfecto posible. Este punto también se conoce bajo el nombre de *"Principio de la Gravitación Selectiva"*.

22. Las piezas que debieran ser intercambiables, en el momento de la sustitución, por alguna nimiedad, no lo serán.

23. La probabilidad de que se produzca una avería por fallo de algún componente es tanto mayor en cuanto el componente en cuestión se halle en el sitio más recóndito y sea más inaccesible para su sustitución.

24. Son siempre las piezas más frágiles las que, por descuido, se dejan caer con más frecuencia.

25. Un transistor que esté protegido por un fusible ultrarápido, en realidad será él quien proteja al fusible, evitando que éste salte primero.

26. Los osciladores en los que las oscilaciones deban cebarse espontaneamente jamás arrancarán por sí solos.

27. En un oscilador controlado a cristal es rara la vez en que oscila a la frecuencia que se indica.

28. Un circuito que no deba oscilar, entrará espontaneamente en oscilación a la primera oportunidad.

29. En general, las averías se manifiestan después de la comprobación final; es decir, cuando se tenía la completa seguridad de que todo estaba correcto.

30. Si un prototipo funciona a la perfección, los aparatos que después se fabriquen en serie serán teoricamente iguales al modelo, pero no funcionarán como estaba previsto.

31. Si un aparato tiene una avería intermitente y le cambiamos una pieza que parece francamente defectuosa, la avería reaparecerá sin duda tan pronto como se ponga el aparato en servicio.

32. Una vez terminado el montaje de un aparato se comprobará que nos quedan muchas piezas sobrantes. Por el contrario, el rollo de estaño habrá desaparecido mucho antes de la vista del operador, permaneciendo oculto entre las herramientas. Es conveniente, pues, disponer de un segundo rollo o carrete de estaño a fin de que éste, a su vez, pueda extraviarse y sea necesario aprovechar los residuos que quedaron diseminados por la mesa de trabajo.

33. La limpieza del laboratorio y cuartos de estudio por personas ajenas a los mismos tiene por objetivos principales:

a) Amontonar sin orden ni concierto las piezas más diversas, con el pretexto de que ocupen el mínimo espacio y resulte más fácil la limpieza de las grandes extensiones que así quedan libres.

b) Favorecer la rotura de elementos valiosos y el extravío de piezas de compromiso.

c) Despistar al operador, el cual por rutina o intuición ya conocía el sitio en el que había dejado cada uno de sus elementos de trabajo.

d) Hallar piezas que se habían extraviado con anterioridad y cuya recuperación no presenta interés alguno.

34. El coste real de un aparato resultará ser, por lo menos, tres veces superior al que se había previsto en un principio. Exactamente 3.14159265..., de ahí la importancia del número **PI**.

35. El producto de componentes por el número de horas de funcionamiento libre de averías es una constante universal.

Por último, aunque no por ello menos importantes, unos cuantos corolarios más, descubiertos por la "*ASociación de Consumidores Independientes de Indochina*" (A.S.C.I.I.):

1. El prometido plazo de entrega deberá multiplicarse por 2. (Eso es en Indochina, ya que en España el factor (X) es una incógnita, aunque siempre $X>2^{3/2}$).

2. El rendimiento estimado por el fabricante de un producto debe dividirse por 2.

3. El rendimiento estimado por el vendedor debe dividirse por 4.

4. La garantía queda anulada con el pago de la factura.

5. La otra cola siempre avanza más deprisa.

6. Cuando nuestro avión llega con retraso, el avión con el que debemos enlazar despegará puntualmente.

7. Las posibilidades de que una tostada caiga por el lado de la mantequilla son directamente proporcionales al precio de la alfombra que hay debajo.

8. El hombre que sonríe cuando las cosas van mal es que ya ha encontrado a quien echarle la culpa.

9. En el póker, una pistola Parabellum gana a cuatro ases.

10. Las probabilidades de recibir una llamada telefónica cuando nos estamos duchando son proporcionales al hecho de estar solos en casa, al hecho de estar enjabonándonos y al hecho de no tener contestador automático.

11. Las probabilidades de que haya una fluctuación importante del suministro eléctrico son proporcionales a la importancia del programa en cuestión, al número de lineas tecleadas y al tiempo transcurrido desde que efectuamos la última copia de seguridad.

Como habréis podido comprobar, este número me he decidido por un artículo más "relajado" de lo que suele ser normal (pero no os acostumbréis mal...). El material de este artículo es un refundido entre uno publicado en el número 3 de *Club Commodore* (Diciembre de 1.983) y unos mensajes recibidos mediante RADIOPAQUETE, enviados por **EB1EVB** (Julio de 1.991). Según comenta **EB1EVB**, obtuvo la información de un artículo de un tal Frances Daura Luna publicado en la revista *Elektro 1*.

Este artículo va dedicado a unos cuantos amigos de Alicante (Hola Andrés, Carmen, Iván, Jose, Pepe), a otros de Barcelona (¿Qué tal, Chechu? Hola Josep) y a un par de compañeros de la Universidad. Espero que ya nadie más me pregunte "¿de qué va eso de Murphy?"...

Hasta el próximo número.

Historia de la informática (VI)

Nacho Agulló

En este número vamos a estudiar las primeras máquinas con cierta capacidad de cálculo. Como veréis los primeros ordenadores fueron utilizados como calculadoras por los científicos e ingenieros.

En partes anteriores hemos visto el nacimiento de las calculadoras electromecánicas. En esta parte veremos más sobre estas máquinas aparatosas, pesadas y ruidosas, que evolucionarán hasta el punto de poder ser soporte de funciones matemáticas complejas. Por otra parte, las matemáticas experimentarán nuevos avances.

La máquina predictora de mareas de Lord Kelvin

Este científico norirlandés, estudioso de muchas materias, fue autor también de un estudio sobre las mareas. En 1873 construyó un ingenio mecánico que realizaba predicciones de mareas, utilizando "integradores" (dispositivos capaces de calcular con precisión el área de una superficie irregular).

El analizador diferencial de Vannevar Bush

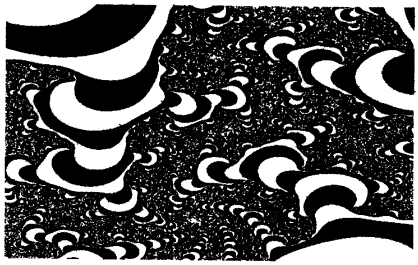
Bush, profesor de transmisión de energía eléctrica en el MIT (Massachusetts Institute of Technology), se dedicó a la investigación de uno de los principales problemas del suministro de electricidad: Los cortocircuitos causados por un aumento en la demanda de energía.

A principios de los años veinte, construyó el "producto telegráfico", una rudimentaria calculadora analógica basada en un vatímetro. El éxito de esta máquina en la resolución de conjuntos de ecuaciones le alentó a pensar en cómo resolver ecuaciones diferenciales de segundo grado o aún mayor dificultad.

En aquella época, el estudio de los sistemas eléctricos era trabajoso y difícil: la aplicación de las ecuaciones de Maxwell al sistema eléctrico producía un número inmanejable de ecuaciones. A finales de los años veinte, inspirándose en parte en el predictor de mareas de Lord Kelvin, aplicando amplificadores a la salida de un integrador para utilizarla como entrada de otro, ideó un analizador de red para simulación del funcionamiento de grandes redes electricas. A partir de 1930 trabajó con un equipo en el MIT construyendo el analizador diferencial para resolver ecuaciones diferenciales de segundo grado, de aplicación en numerosas

áreas de la ciencia y la ingeniería. La máquina, terminada en 1931, era totalmente mecánica y constaba de una compleja estructura de engranajes, ejes y motores eléctricos. Las entradas y salidas se realizaban mediante el giro de ejes, y la retroalimentación se conseguía mediante un amplificador de par. Podía manejar hasta 18 variables independientes.

Este analizador diferencial constituyó un éxito completo, construyéndose varias unidades en Europa y Estados Unidos. No obstante, su concepción analógica implicaba fuertes limitaciones: El costo se decuplicaba al tiempo que la precisión.



Posteriormente se construiría otro analizador de salida digital y entradas suministradas en cinta de papel perforado.

Alan Turing: el pionero de la Inteligencia Artificial

Al tiempo que Bush construía su primer analizador diferencial, se publicaba el "Teorema de Godel", que demuestra la imposibilidad en cualquier sistema matemático lógico fijo de probar la contradictoriedad o no de ciertas proposiciones a partir de los axiomas del sistema solamente, y por lo tanto, es indemostrable que los axiomas básicos de la aritmética no producirán contradicciones. Este teorema, formulado por un matemático joven, sorprendía al mundo científico y en especial, a un todavía más joven estudiante del King's College de Cambridge: Alan Turing.

Turing comenzó a trabajar en lógica matemática, que después estudió en la Universidadd de Princeton. Allí escribió "On Computable Numbers", donde describía teoricamente un ordenador universal y demostraba que problemas matemáticos irresolubles mediante un proceso fijo y definido pueden ser resueltos por una máquina automática. Tal es el concepto de "máquina de Turing", con el que se designa a autómatas de múltiples estados, utilizado en la informática actual para demostrar la posibilidad de informatizar problemas: Todo problema resoluble mediante una "máquina de Turing" es informatizable.

Pero además Turing concluía que era posible diseñar un ordenador "universal", programable para desempeñar la función de cualquier "máquina de Turing".

De él es también el "test de Turing", que define la "inteligencia" de una máquina pensante dependiendo de si un operador de teleimpresora que envíe preguntas a través de la misma es incapaz de diferenciar las respuestas que la máquina le envía a través de la teleimpresora de respuestas enviadas por otro operador.

En la próxima parte veremos la aparición de los primeros ingenios electromecánicos digitales: allí volveremos a encontrarnos con Alan Turing.

Biografías

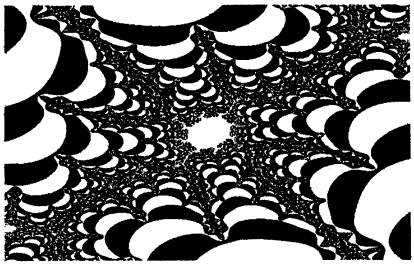
Lord Kelvin (1824 Belfast - 1907

Netherhall, Escocia)

Nacido William Thompson, hijo de un profesor de matemáticas de la Universidad de Glasgow. De extraordinaria inteligencia, ingresó en la misma a los diez años y se licenció brillantemente con veintiuno, siendo premiado con la cátedra de filosofía natural al año siguiente, donde permanecería por el resto de su carrera. Publicó mas de 600 escritos científicos, proporcionando nuevos avances a ciencias tan dispares como la termodinámica, el electromagnetismo o la hidrodinámica, pasando por el análisis matemático de la electricidad y la escala de temperaturas que lleva su nombre. Participó en el tendido del primer cable transatlántico, e hizo fortuna con la invención de un receptor para el telégrafo submarino. En 1892 la reina le nombró Barón de Largs.

Vannevar Bush (1890 Everett, EE. UU. - 1974 Belmont, EE.UU.)

Cursó estudios en la Universidad de Tufts, en Medford, Massachusetts, de 1914 a 1917. Al entrar en la primera guerra mundial los EE.UU., Bush realizó detección antisubmarina en la Armada, después de lo cual entró en el claustro del Massachusetts Institute of Technology en Cambridge, en 1919. Aparte del analizador diferencial, desarrolló el "Rapid Selector", un dispositivo de recoperación de información con código y microfilm que estimularía el interés por la coordinación de grandes cantidades de información. Llegó a ser decano de la escuela de ingeniería y vicepresidente del Instituto Carnegie en 1939; por su buena administración se le nombró presidente del Comité de Investigaciones para la Defensa Nacional, siendo responsable de las investigaciones del ejército estadounidense durante la guerra, teniendo una participación decisiva a la hora de la autori-



zación al proyecto Manhattan, que obtendría la primera bomba atómica. En 1955 se jubiló, pasando a dedicarse a sus hobbies.

Kurt Godel (1906 Brno, Checoslovaquia - 1978 Princeton, EE. UU.)

Matemático y lógico, miembro desde 1930 del claustro de la Universidad de Viena y del "Círculo de Viena", orientado hacia el positivismo lógico, emigró a los EE.UU. tras la invasión nazi junto con varios compañeros del círculo, nacionalizándose estadounidense en 1948. Allí trabajó como profesor en el Instituto para Estudios Avanzados, del que ya era miembro con anterioridad.

Bibliografía

- ***Historia de la Informática.***
Amparo Gil Orihuel e Ignacio Rieiro Martín. Ed. Alhambra.

- ***Enciclopedia Monitor.***
Ed. Salvat.

- ***Enciclopedia Mi Computer.***
Ed. Delta.

- ***Gran Enciclopedia Informática.***
Ed. Nueva Lente.

- ***Enciclopedia Británica.***
William Benton Publisher.

¡¡ Noticias Frescas !!

Jose Manuel Suárez

En esta sección haremos un somero repaso de las últimas novedades en este mundillo de la informática y la microelectrónica. Si queréis colaborar enviad vuestras noticias a la dirección indicada en la página 2.

EL MICROPROCESADOR 88000 DE MOTOROLA

Uno se acuerda del interés levantado por el microprocesador 88000 de Motorola cuando salió a la calle. Pero aparte del Avión de Data General, pocas máquinas destacables salieron equipadas con esta UCP, a pesar de lo novedoso de su arquitectura.

Esto cambiará en breve. En Septiembre NeXT sacará su primera máquina equipada con el 88110 de Motorola, sucesor del 88000, que debería estar disponible para entonces. Las actuales máquinas de NeXT, basadas en el Motorola 68040, sufren de la falta relativa de velocidad de este procesador en las operaciones gráficas intensivas, aplicaciones corrientes en las estaciones de trabajo.

Las futuras máquinas de NeXT utilizarán la versión a 50 MHz del 88110, que tiene una potencia de 67 SPECmarks (unos 80 MIPS, millones de instrucciones por segundo). Más interesante aún, saldrán al mercado versiones multiprocesador, aumentando la potencia de cálculo en proporciones casi lineales con el número de procesadores (100 SPECmarks para una máquina biprocesador).

Por otro lado, muchos temían que la alianza Apple-IBM-Motorola fuese letal para el sistema 88000. Según este acuerdo, el superpotente microprocesador PowerPC (corazón de las estaciones de trabajo IBM con entorno UNIX) será fabricado por Motorola, y entrará entonces en competencia directa con el 88000. Pero también se acordó que el PowerPC tendría un interfaz destinado a la conexión, en un entorno multiprocesador, con el 88110. El 88000 y el PowerPC serán, por lo tanto, complementarios en esta nueva estrategia.

Otra noticia importante es que Motorola abandonó sus planes de sacar una gama de microcontroladores (microprocesadores con ROM, RAM y periféricos integrados) basados en el 88000, destinados a subsistemas gráficos, impresoras láser sobre todo. El PowerPC de IBM será el núcleo de la nueva familia de controladores RISC comunes a IBM y

Motorola, actualmente en desarrollo en los laboratorios de IBM en La Gaude, cerca de Niza, en Francia.

PRODUCTOS APPLE

¿Productos Apple a la venta proximamente en los supermercados? Es lo que prometió John Sculley, Presidente de la empresa californiana, en el transcurso del Winter Consumer Electronics Show que tuvo lugar en Las Vegas el mes de Enero pasado.

Los Macintosh ya habían dejado las lujosas estanterías de los distribuidores profesionales para hacer frente al rigor del mercado de masas, después de la histórica caída de precios de estas máquinas. Ahora Apple quiere continuar en este camino, anunciando una gama de productos llamados "Personal Digital Assistants". Esta gama incluirá libros electrónicos (similares a los anunciados por Sony), organizadores (agendas electrónicas), blocs de notas electrónicos y teléfonos de bolsillo.

Además, Apple pretende introducir en los mercados traditionales del PC versiones específicas de sus Mac básicos, los cuales podrían estar a la venta en las grandes superficies durante el segundo semestre de 1.992. La empresa de la manzana multicolor también tiene previsto lanzar nuevos modelos de Mac diseñados alrededor de un CD-ROM. Uno para el mercado doméstico, el otro para los usuarios profesionales.

Es agradable ver que Apple abandona sus viejas creencias según las cuales la empresa tenía el deber sagrado de enseñar sus métodos a los "trabajadores del saber" (los "clientes" de Apple), y de que el talento de sus ingenieros beneficie a todos.

IBM Y LOS RAYOS X

La multinacional IBM acaba de realizar un prototipo de memoria dinámica de 256 Mbits (32 Mbytes). Esta capacidad tan elevada según los criterios actuales ha sido obtenida gracias al empleo de una nueva técnica de grabado mediante rayos X, empleando un sincrotrón. La longitud de onda tan corta de estos rayos permite realizar un circuito integrado con una precisión de 0.1 micrómetros en vez de los 0.8 de los circuitos actuales.

Trabajar con un tamaño de transistor tan reducido ocasiona toda una serie de problemas aún por resolver. Citar, entre otros, la necesidad de un altísimo nivel de filtrado, más allá de las capacidades actuales, del aire de la sala de producción para evitar la destrucción de los chips a causa de minúsculas partículas de polvo. La pureza del substrato de silicio también constituye un problema importante.

De la carencia de defectos en la red cristalina del sustrato depende el funcionamiento correcto del chip. El sustrato es un cilindro de 15 a 20 centímetros de diámetro formado por un monocristal (un enorme cristal de silicio que se hace crecer en atmósfera controlada). Lo ideal consistiría en obtener diámetros cada vez mayores, para aumentar el rendimiento de fabricación, al mismo tiempo que se reduce el número de imperfecciones de la oblea. Pero esto se va haciendo cada vez más difícil hasta el punto de convertir la fabricación de monocristales en una auténtica

pesadilla. Las futuras estaciones orbitales podrían tener mucho que decir a este respecto.

EL TRANSISTOR NEURONAL

Actualmente es bastante común oir hablar de las aplicaciones de las redes neuronales. En resumen, todo consiste en interconectar un gran número de células donde cada una, imitando a la propia naturaleza, produce como salida la media ponderada de los valores de entrada. Los diferentes valores de los coeficientes de esta media ponderada proveen a la neurona artificial de cierta capacidad de aprendizaje. Esto permitirá, por ejemplo, que los ecualizadores gráficos puedan "reconocer" los diferentes tipos de música que se les presenta y efectúen, en consecuencia, un reglaje adecuado. Las aplicaciones más prometedoras caen en el dominio del reconocimiento de formas, como el reconocimiento de la escritura manuscrita o de la voz.

Hasta el momento los investigadores se veían obligados a simular las redes neuronales a nivel lógico, pero para ciertas aplicaciones esta técnica resulta demasiado lenta, como en las destinadas al gran público. En estos casos la única solución era integrar el programa en un microcontrolador, similar a los que incluyen algunas cámaras de fotos o cadenas de alta fidelidad, o bien utilizar circuitos integrados especializados.

Unos investigadores japoneses de la universidad de Tohoku acaban de poner a punto un nuevo componente que permitirá simplificar la integración de redes neuronales con los componentes electrónicos tradicionales, multiplicando las aplicaciones de pequeño tamaño. Consiste en un transistor de tecnología MOS (y, por lo tanto, integrable a gran escala) que conmuta si la media ponderada de sus entradas supera un valor predeterminado. De esta forma será posible implementar una célula de una red neuronal en un solo componente, en vez de los numerosos comparadores y puertas lógicas utilizadas hasta el momento. Además, este tipo de dispositivos (MOS) ocupan una superficie muy pequeña en la superficie de silicio de los microcontroladores lo cual permitirá a los fabricantes vender más inteligencia a precios razonables.

Cualquiera que haya echado un vistazo a la lista de comandos de un magnetoscopio o de una fotocopiadora actual habrá sentido un profundo deseo de simplificación en el uso de estas máquinas, tarea en la cual esta nueva tecnología podría contribuir en gran medida.

EN BREVE...

Intel pronto sacará al mercado su nuevo microprocesador: el Intel 80586. La fase de desarrollo de este chip fue terminada hace más de un año pero hasta el momento no ha sido comercializado por razones de marketing. Al parecer, el stock actual de 80386 y, sobre todo, de 80486 es demasiado elevado como para garantizar suficientes ventas. Mientras tanto Intel se dedica a aumentar la velocidad de funcionamiento de sus procesadores ya instaurados. Velocidades de 40 o, incluso, 50 Mhz pertenecen ya al pasado.