¿Que hay detrás de los cálculos rápidos? el secreto de los algoritmos

Seguramente habrá llamado la atención, la rapidez con que una calculadora determina el logaritmo de un número o el seno de un ángulo. Si tiene algún conocimiento matemático superior, habrá pensado que la máquina debe utilizar un desarrollo finito o una aproximación polinómica. De hecho, el método empleado es original en relacción con los procedimientos clásicos del cálculo numérico y hace uso de los algoritmos, adecuándolos a la estructura de la máquina. Se invita en este artículo, al descubrimiento de los "algoritmos máquina".

En 1972, la firma americana Hewlett-Packard lanzaba al mercado la primera calculadora científica realizada en el mundo, la HP35.

Este maravilloso objeto, que tenía todas las funciones básicas de un ordenador, era el resultado de la unión de dos técnicas: por una parte, las tecnologías microelectrónicas de alta integración, (el HP 35 reunía en algunos circuitos el equivalente a 35.000 transistores Mos), y por otra parte, las técnicas, perfectamente dominadas, de la micro-programación.

El tamaño de la máquina, era un inconveniente importante para la programación de funciones matemáticas complejas. En efecto, mientras que las funciones simples del tipo x^2 o $1/x$ son fáciles de obtener utilizando las cuatro operaciones fundamentales, en el caso de funciones trascendentales y trigonométricas parece que, existe un obstáculo insalvable, el tiempo de tratamiento.

El lector que no se haya olvidado de los conocimientos matemáticos, recordará la lentitud de convergencia de un desarrollo en serie. Por otra parte, se podrá dar cuenta de que para obtener una precisión suficiente, el número de términos a calcular en el desarrollo, debe ser grande. Lo mismo ocurre en el caso de una aproximación polinómica (1). Ahora bien, mientras que los algoritmos que utilizan estos principios se pueden explotar perfectamente en máquinas de mayor tamaño, hay otro inconveniente, basado en la arquitectura de la máquina de bolsillo.

En razón a la pequeña superficie de la tarjeta, sobre la cual están impresos los circuitos, el aparato está organizado en base a tres buses (datos-dirección-control) de una sola línea cada uno, La información que forma la palabra máquina está, por tanto, multiplexada en el tiempo. Incluso existen

calculadoras que tienen las tres líneas del bus reunidas en una sóla, y el reloj de la unidad lógica permite separar los paquetes de información. La toma de datos se realizará, por tanto, bit a bit y la salida se efectuará segmento a segmento en los diodos electroluminiscentes (2).

El lector se habrá dado cuenta que, para una arquitectura de este tipo, los circuitos que se imponen, para la unidad lógica y la airtmética, son el sumador y el substractor realizados con la ayuda de registros de desplazamiento.

Las tres funciones básicas de la unidad aritmética serán, por tanto, la suma, la resta y el desplazamiento. Hay que destacar que el algoritmo que realiza las cuatro operaciones (+, −, X, ÷) en coma flotante sólo usa estas tres funciones, la división de mantisas se efectuará por resta-desplazamiento, mientras que la multiplicación se hará por el método suma-desplazamiento.

La estructura interna de la calculadora de bolsillo está, por tanto, adaptada al cálculo aritmético en coma flotante. Por otro lado, parece imposible realizar, con esta arquitectura y con memorias ROM de pequeño tamaño en circuitos de rápida evaluación (casi simultáneamente a la presión de una tecla), funciones matemáticas complejas.

Sin embargo, es lo que realizó el ingeniero J. Volder, concibiendo para una firma aeronáutica una calculadora, (airbone computer), capaz de realizar, en vuelo y rápidamente, con una unidad lógica en serie, calculos trigonométricos y logarítmicos de navegación. Cordic (Coordinate Rotation Digital Computer), éste era su nombre, es de alguna forma el antepasado de las HP 41 y de las TI 59.

El principio del método, reside en el hecho de que todos los cálculos se efectúan por medio de sumas y restas realizadas en registros de desplazamiento, para así respetar la arquitectura de la máquina.

En el caso de una calculadora que utilice datos codificados en D C B (Decimal codificado binario), nuestros casos (3) el modo angular elegido será el radián. Esta decisión está marcada por el método.

El lector observará que, mediante algunas constantes almacenadas en memoria ROM, todo cálculo trigonométrico puede realizarse a través del de la tangente de un ángulo comprendido entre 0 y /4 radianes, ya que,

$$|\cos x| = 1 / \sqrt{1 + \mathrm{tg}^2\ x}$$

$$|\mathrm{sen}\ x| = \sqrt{1 - \cos^2\ x}$$

Un algoritmo adaptado a la estructura de la máquina

Los cálculos se realizarán según el organigrama de la figura 1. Se observa que el proceso consiste en restar del ángulo dado unas constantes Qi codificadas en memoria ROM, tantas veces como sea posible, y de cambiar de constante cuando la resta no se pueda realizar. Estas constantes denominadas ATR (arcotangente

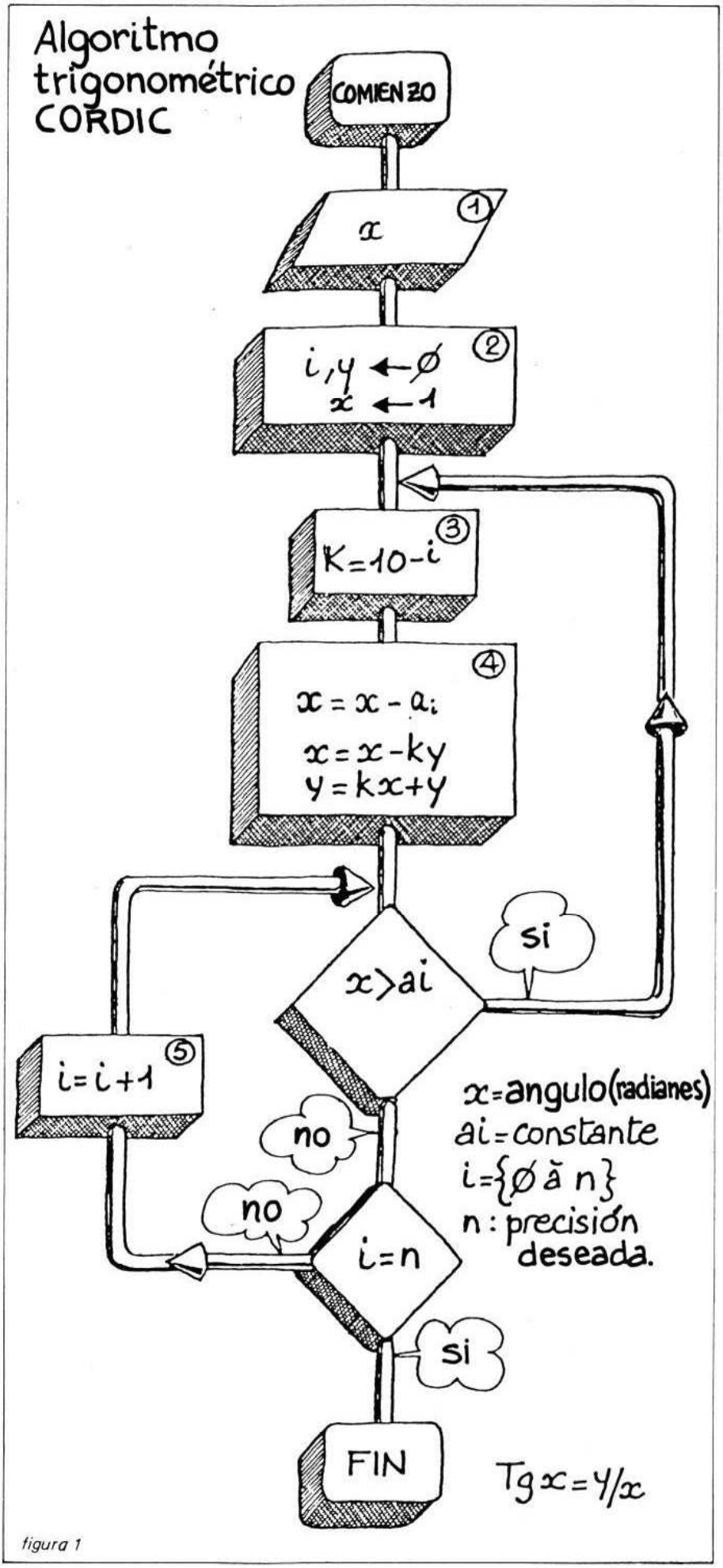


figura 1

radix) - (rectángulo 4, figura 1), consisten únicamente en desplazamientos.
Para un cálculo en DCB (decimal), estas constantes son iguales a:

$a_0 = tg^{-1} . 1 = tg^{-1}\ 10^{-0}$ $(=\pi/4)$

$a_1 = tg^{-1} . 0{,}1 = tg^{-1} . 10^{-1}$

$a_2 = tg^{-1} . 0{,}01 = tg^{-1}\ 10^{-2}$

$a_i = tg^{-1} . 10^{-i}$ $(= 10^{-i}$ para un i suficientemente grande).
Estudiando el organigrama, se puede percibir que el algoritmo se asemeja mucho al de la división. Está perfectamente adaptado a la concepción "hardware" de la máquina y este hecho, juega un papel "milagroso" para su rápido desarrollo.
El valor trigonométrico se encuentra después de la ejecución de unas decenas de bucles y se puede constatar en el ejemplo de ejecución del cálculo de la tangente de x (figura 2) que ilustra el método de cálculo.

Por supuesto, el método también funciona en el otro sentido y permite calcular el ángulo, partiendo del valor de la tangente. La demostración matemática de la convergencia de este algoritmo puede encontrarse en el artículo de J. Volder. Lo raro, es que este principio se pueda aplicar a otros tipos de cálculos, entre los cuales se puede destacar el cálculo del logaritmo de un número.

Una impresionante rapidez de convergencia

La misma rapidez de convergencia y la misma estructura, tan simple, se puede encontrar en el proceso del cálculo del logaritmo de un número.
Cabe decir, que se puede, como en el caso de las funciones trigonométricas, ir a un problema tipo.
En efecto, el caso de los logaritmos neperianos es el único a considerar. Los logaritmos decimales se pueden evaluar al precio de almacenamiento de una constante en memoria ROM (4). Este va-

Ejemplo de ejecución del cálculo de la Tgx

x = 1,5 Rad. en ROM a0 = Arctg 1 = $\pi/4$
a1 = Arctg 0.1 = 0,0996686
a2 = Arctg 0.01 = 0,0099996
a3 = Arctg 0.001 = 0,001
a4 = Arctg 0.0001 = 0,0001

Paso	Componentes	Vectoriales	Angulo
nº	X	Y	x = x-ai
1	1	1	0,71460184
2	0,9	1,1	0,61493318
3	0,79	1,19	0,51526453
4	0,671	1,269	0,41555958
5	0,5441	1,3361	0,31592723
...	...	...	...
15	0,10493	1,460646	0,00092160
...	...	...	...
20	0,10420414	1,46069833	0,0004216045
...	...	...	...
30	0,10358875	1,46074213	0,0000003045

Al final de las iteraciones, cuyo número depende de la precisión deseada, el proceso converge x = 0 e y/x = tg 1,5 radianes que es igual a 14,101499

Figura 2

lor (1n 10) será útil para el resto de los cálculos.

$10g\ x = 1n\ x / 1n\ 10$

$10^x = e^{x\ 1n\ 10}$, en donde 1n es el logaritmo neperiano en base e.

Por otra parte hay que considerar que, de forma interna, los números se representan siempre en coma flotante, independientemente del formato real visualizado.

Para el número $A = x \quad 10^p$ se tiene la relación $LnA = Lnx + p$ Ln 10 (siendo $A > 0$), donde también aparece la constante Ln 10.

El último caso a considerar es el de la evaluación del logaritmo neperiano de x para $1 \leqslant x \leqslant 10$.

Las constantes preprogramadas son las siguientes (5):

Ln 10
Ln 2 = Ln $(1 + 10^0)$.
Ln 1.1 = Ln $(1 + 10^{-1})$.
Ln 1.01 = Ln $(1 + 10^{-2})$.
Ln 1.001 = Ln $(1 + 10^{-3})$ etc, según la precisión de la máquina.

El algoritmo puede ser esquematizado como se indicó en el ejem-

Listado de la memoria ROM (128 octetos en total)

Paso	Valor en DCB en 104 octetos	Constante representada
384 - 391	2.302585092994012	ln 10
392 - 399	0.693147180559945	ln 2
400 - 407	0.095310179804325	ln 1.1
408 - 415	0.009950330853168	ln 1.01
416 - 423	0.000999500333084	ln 1.001
424 - 431	0.000099995000333	ln 1.0001
432 - 439	0.000009999950000	ln 1.00001
440 - 447	0.000000999999500	ln 1.000001
448 - 455	0.785398163397450	Arctg 1 ($\pi/4$)
456 - 463	0.099668652491200	Arctg 0.1
464 - 471	0.009999666686670	Arctg 0.01
472 - 479	0.000999999666667	Arctg 0.001
480 - 487	0.000099999999667	Arctg 0.00001

Constantes preprogramadas en memoria ROM en la máquina Ti59 para el conjunto de los cálculos efectuados siguiendo el procedimiento CORDIC.

Procedimiento de listado:
GT04, LRN, Pgm 2, SBR 240, RST, Lbl 4, LRN, Luego lanzar RST R/S y listar LRN SST.

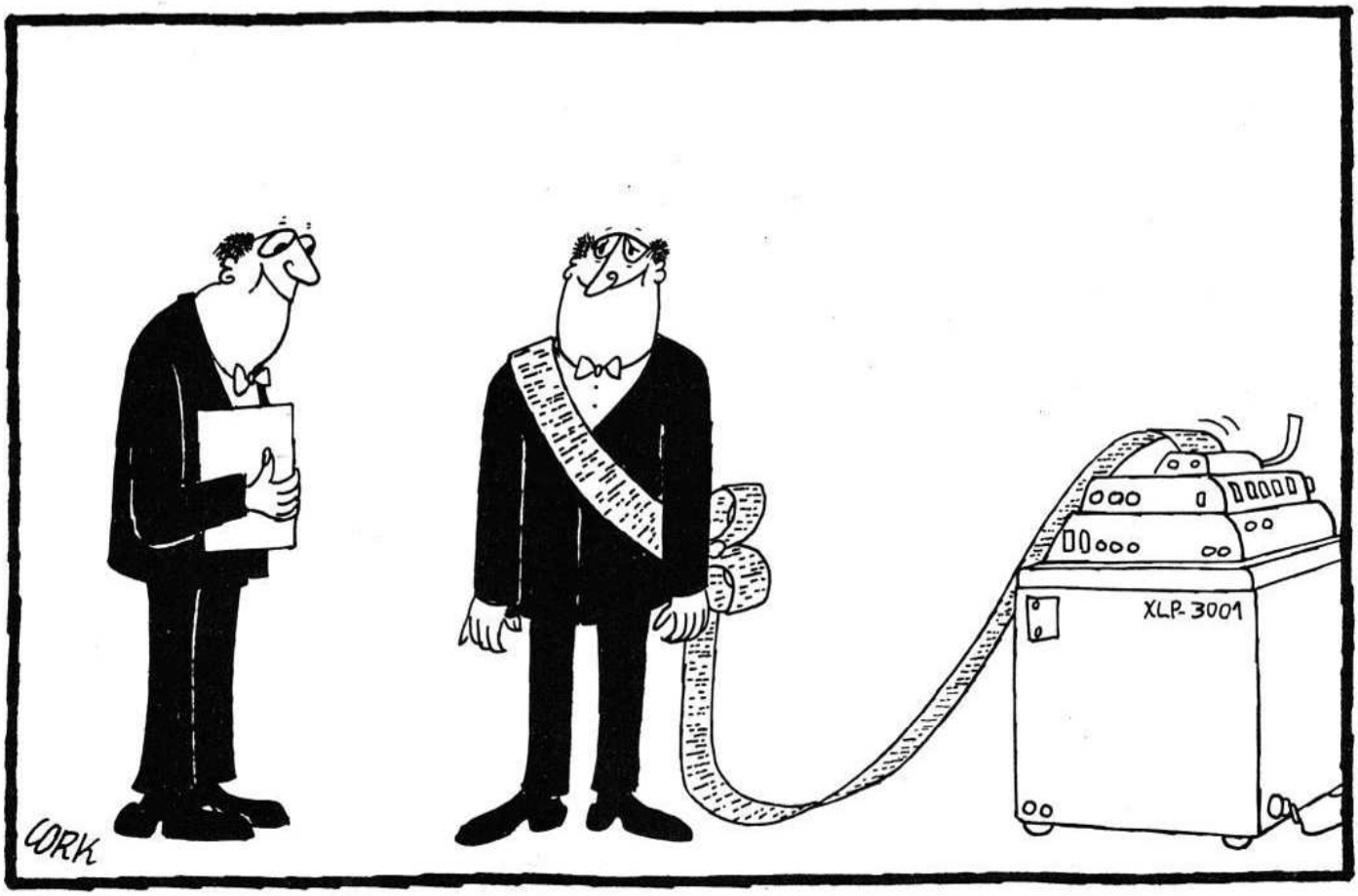
Entre 488 y 511 se encuentran las constantes de conversión de los ángulos $180/\pi$, $\pi/2$, así como el número π, codificados en 24 octetos.

Bibliografia

J.E. Volder, "The Cordic Trigonometric Computing Technique", IRE, Transactions on Electronic Computers, Sept. 1959. (Se puede consultar en París en el CNRS y en el CNET, donde agradecemos al servicio TELEDOC su amable acogida.)
A.Deledicq, P. Vaschaide, Le calculateur programmable de poche, CEDIC, 1978 (y en particular las páginas 112 a 115).
J.M. Smith, Méthodes Numeriques pour Calculateur de Poche, Eyrolles, 1976 (y en particular las páginas 38 a 42).
Texas Instruments, Software Exchange Newsletter, Vol II nº 2 y 3.

plo del cálculo de Ln x (figura 3). En cada bucle se multiplica el número x: por un valor Ai, sin que el resultado pueda exceder a 10. Si esto ocurriese se haría un cambio de constante Ai en el orden (2; 1.1; 1.01; 1.001; etc.).
En cada bucle se efectua la resta de Ln Ai, de Ln 10. El lector atento habrá observado, que todas las multiplicaciones que hay que realizar se pueden efectuar de forma simple y rápida con la ayuda de operaciones de desplazamiento y de suma, de acuerdo con los principios del método.

El método de Volder se puede aplicar a la evaluación de muchas funciones matemáticas como $\sqrt{x}$, tg x, Lnx.
También se pueden resolver otros problemas más generales, como por ejemplo, la conversión de binario a decimal.
Para el caso de las calculadoras de bolsillo, se puede apuntar que la programación de los algorítmos logarítmicos y trigonométricos permite obtener fácilmente la evaluación de otras funciones útiles en el cálculo científico.
$P\rightarrow R$, $R\leftarrow P$, Y^X (es igual a $e^{x\,Lny}$) etc.



Ejemplo de cálculo de ln X

Pasos	Multiplicaciones y desplazamientos			Substracción de constantes
i	Xi	Ai	Xi+ 1	
				ln 10
1	4.4	2	8.8	− ln 2
2	8	1.1	9.68	− ln 1.1
3	9.68	1.01	9.7768	− ln 1.01
4	9.7768	1.01	9.874568	− ln 1.01
5	Xi + 1	1.01	9.97231368	− ln 1.01
6	,,	1.001	9.983286994	− ln 1.001
7	,,	1.001	9.993270281	− ln 1.001
8	,,	1.0001	9.994269608	− ln 1.0001
9	,,	1.0001	9.995269035	− ln 1.0001
10	,,	1.0001	9.996268562	− ln 1.0001
				= 1.481977754
		Ajuste	$(X_{1\,1}/10_{-1})$	− 0.0003731438
				= 1.48160461

Se observa, que después de 10 bucles el error es mínimo restando constantes de 13 cifras significativas. En el caso de las máquinas Texas Ti-58-59, las constantes se almacenan con 16 cifras en BCD (figura 2). Al final del proceso las multiplicaciones tienden a 10, mientras que las substracciones -después del ajuste $(X_{11}/10\text{-}1)$- dan el resultado del cálculo.

Figura 3

Ejemplo de cálculo de e^x

Partiendo por ejemplo del valor x = 0.2105210213 que converge rápidamente, se tiene:

Substracción de constantes	Multiplicaciones de los desplazamientos
0.2105210213	1
− ln 1.1	1.1
− ln 1.1	1.21
− ln 1.01	1.2221
− ln 1.01	1.234321 = e^x

(El resto tiende a cero)

Ejemplo de cálculo de 1/X

Partiendo por ejemplo del valor x = 432

Substracciones		Desplazamientos	Adiciones
1 - 432		1	−
1 - 43.2	Imposible	.1	−
1 - 4.32		.01	−
1 - 0.432 = 0.568		.001	.001
0.568 - 0.432 = 0.136		.001	.002
0.136 - 0.0432 = 0.0928		.0001	.0021
0.0928 - 0.0432 = 0.0496		.0001	.0022
0.0496 - 0.0432 = 0.0064		.0001	.0023

etc, al final del proceso, las sucesivas substracciones tienden a cero, mientras que las sumas de los desplazamientos dan 0.00231481.

Por último, y no hay que sorprenderse,un tal Brigss había descrito (en los años 1620), un método en el cual se inspira el de Volder. ¡Ni el ordenador ni la calculadora de bolsillo existían todavía!. Se trataba de optimizar el trabajo y de economizar el esfuerzo de las primeras calculadoras (humanas) de tablas numéricas.

La próxima vez que pulse la tecla (Lnx) de una Hp 41 o TI 59, quizás le divierta pensar que la máquina esta recalculando la tabla de logaritmos, como en los viejos tiempos de Brigss y Neper.

Jacques Laporte

1. A modo de ejemplo, el desarrollo

$$\text{sen } x = x - \frac{x^3}{3\,i} + \frac{x^5}{5!} - \frac{x^7}{7!}\,|$$

converge muy lentamente y para obtener una buena precisión es necesario calcular un número muy grande de términos.

2. A modo de comparación, se observará que la "palabra máquina" de un microprocesador de tipo corriente, es de 8 bits (un octeto) y que las transaciones internas se efectúan en paralelo. El HP 35 utilizaba algunas centenas de micro-instrucciones codificadas en 12 bits.

3. En la calculadora de bolsillo, los cálculos se efectuan cifra a cifra (1 cifra = 4 bits) debido a las razones expuestas más arriba. Este hecho confiere una buena precisión, pero gran lentitud. En la calculadora TI 59, un número se almacena en 8 octetos. Como comparación, un número en simple precisión se almacena en coma flotante binaria en el lenguaje BASIC II de Radio Shack, lo que permite una representación de números de hasta + 1. 701411 E + 38 (TRS 80).

4. Se observará que el logaritmo decimal de 0.5, por ejemplo, no está representado en la forma clásica $\bar{1}$. 6989700043, sino como −0.30 10 29 99 57.

5. Figura 2, cuadro segundo, Las constantes "Cordic" programadas de forma real en la memoria ROM de la calculadora Texas TI 59.