NATIONAL RADIO ASTRONOMY OBSERVATORY

ELECTRONICS DIVISION TECHNICAL NOTE NO. 103

TITLE: Microcomputer Time Trials

AUTHOR(S): R. Fisher and D. Schiebel

DATE: December 24, 1981

As part of the decision on the best way to duplicate/update the Digital Continuum Receiver we have run some tests on computing speed of two typical micro-computers to see if the much higher cost of the HP9825A is justified. The attached table shows the results.

The HPL and BASIC languages are based on interpreter systems so program changes are very easy in both languages although HPL is slightly more cryptic. By their very nature, interpreters are relatively slow compared to precompiled FORTRAN or assembly systems as can be seen in the Cromemco BASIC & FORTRAN times. However, FORTRAN program changes are more tedious because the edited text must be compiled and linked every time a change is made. On the Cromemco single disk system this took a few minutes.

Also attached to this note is a list of computing times for a bench mark program composed by John Granlund and run on a wider variety of calculating machines.

The HP9825A is still the fastest interpreter based system that we have tried. The Modcomp/BASIC time in John's table shows that going to a bigger machine won't help the speed of an interpreter much. The only way to really speed up the DCR would be to go to a compiler system on a 16-bit machine.

Integration times of 0.1^s to 0.2^s are possible with the HP9825A for 2 to 4 channel systems without too many bells and whistles such as ΔT rms. With more software and display features the integraiton times are typically 0.3^s to 0.4^s which are adequate for most continuum work. Four channel, 0.1^s integration observing could be handled with a stripped-down HPL program which does a mininum amount of arithmetic on the data. The Apple and Z-80/BASIC systems would be too slow for the DCR.

Attachment:
  Table: Function execution times in long loops
  for several microcomputers, R. Fisher.
  Comparison of Computation Times, J. Granlund.

Distribution:
GB Library
CV Library
VLA Library
TUC Downtown File
Mountain File
M. Balister
W. Brundage
C. Burgess
J. Coe
F. Crews
M. Damashek
H. Hvatum
P. Napier
R. Norrod
J. Payne
B. Vance
R. Weimer
S. Weinreb
J. Granlund

TABLE: FUNCTION EXECUTION TIMES IN LONG LOOPS FOR SEVERAL MICROCOMPUTERS

R. Fisher, 23 December 1981

		BASIC 48K Apple II+	HPL HP 9825A	Cromemco Z-80 4MHz clock BASIC	Cromemco Z-80 4MHz clock FORTRAN
for I=1 to 10000 next I		13 s	6 s	9 s	<1 s
X=2; Y=50 for I=1 to 10000 Z=X*Y next I	*	44	15	44	7.5
X=2; Y=50 for I=1 to 10000 Z=X+Y next I	*	36	13	35	4.2
X=20 for I=1 to 10000 Y=SQR(X) next I	*	~540	35	500	124
X=2 for I=1 to 2000 Z=SIN(X) next I		54	28	93	21
X=2 for I=1 to 10000 if X>32000; Y=32000 next I	*	75	12	28	5.5
X=33000 for I=1 to 10000 if X>32000; Y=32000 next I		125	15	36	5.5
X=2 for I=1 to 10000 PRINT X; "NUMBERS" next I		323	28	19K Baud 125	19K Baud 125

	apple	HP	Cromemco BASIC	Cromemco FORTRAN
x = 2.6 for I = 1 to 10000 y = X**2 next I	501^s	21^s	1150^s	15^s
X = 2.6 for I = 1 to 2000 Y = X**2.6 next I	107	36	244	50
X = 2.6 for I = 1 ~~to~~ 2000 Y = ln X next I	73	16	58	28
x = 2; Y = 50 for I = 1 to 10000 * ~~Z~~ = X/Y next I	57	15	51	20

* ≡ commonly used in Digital Continuum Receiver.

3/13/78

Comparison of Computation Times

I've just had the pleasure of comparing the times required by various computers and associated programming languages to solve the same problem. The problem is to determine the prime factors of an input integer. The integer was chosen to be 1,000,003, itself a prime number. In this case, the major computational effort is the execution of 1,000 divides, although a square root, a number of adds -- increments -- and a minimum of data shuffling are also required. Scores are listed below:

Computer	Language	Operator	Time (seconds)
HP 35	—	J. Granlund	3600
HP 67 programmable	machine	V. Granlund	920
HP 9830 A	Basic	J. Granlund	58
Modcomp	Basic	B. Rayhrer	11
V 77 - 400	Basic	A. Shalloway	7
HP 9825 A	HPL	R. Fisher	6
Modcomp	Fortran IV	B. Rayhrer	< 0.2

For the Varian and the Modcomp, and I suppose also for the HP 9830 A, programs written in Basic are assembled one step at a time into machine language; but programs written in Fortran for the Varian or the Modcomp are completely assembled in machine language before the program is run. As can be seen from Benno's two runs, it can be very wasteful to write a program in Basic for a high-speed computer.

JOHN GRANLUND