Reprinted with permission from: Proceedings: National Conference on microcomputer software, Aug 1982. Microprocessor Special Interest Group, Australian Computer Society. MICSIG, Canberra (25p).

BASIC, FORTRAN, S-ALGOL, and Pascal Benchmarks on microcomputers, including the effects of floating point processor support.

M.R.Wigan

[Oxford Systematics, PO Box 281, Mt Waverley, Victoria 3149, Australia]

ABSTRACT

A straightforward set of short and simple benchmarks has been used to examine the relative performance of a wide range of mainframe, mini, and micro systems. Comparable sets of BASIC, Pascal, FORTRAN and Structured Algol codes are given in a series of Tables and Figures. Special attention has been paid to the role of additional processors to assist in the execution of the benchmark codes. These include a 6809 used to accelerate APPLE-Pascal 1.1 p-code and an AMD 9511 to accelerate BASIC-E and Structured Algol semi-compiled code. Two AMD 9511 versions of MicroSoft 8080 FORTRAN FORLIB are included. The overlap in performance of mainframe, mini, and micro systems is illustrated. The major conclusion is that special attention should be paid to high speed interpreters for semi-compiled commercially-available software products as a major thrust towards a transferable set of user environments at different levels of execution speed.

INTRODUCTION

The basic sense of perspective required to consider the relative merits of different languages and compilers on small computers often seems to be in short supply. The relative performance of different types of microcomputer is more a subject of heated debate than of numerate discussion, and the effective performance obtainable from a microcomputer by adding special hardware support for specific functions (such as a floating point processor chip and the appropriate software) is not even widely enough known to *start* the debate.

This paper is dedicated to the task of providing strictly comparable measures of performance for a vast range of processors and languages, and to providing the basic material for a useful view of the effects of floating point processors and the relative effectiveness of mainframes, minis and micros in the limited set of tasks used as benchmarks.

BENCHMARKS AND ENVIRONMENTS

The benchmarks used have been published in a number of places. The two primary sources are Coll (1978) and Fox (1980).

The first source was John Coll's paper in the 27 July 1978 issue of Computing Europe (later issued in the proceedings of the Do-It-Yourself Computing Conference, OnLine Conferences, Uxbridge,UK). Coll presented the results of numerous runs done on eight simple programs written in BASIC on a large number of machines. Seven of these programs had previously been used for an earlier article in the June 1977 issue of Kilobaud. As Coll pointed out , although his eighth program added trancendental functions to the range of tests, there was still no string handling program included in this expanded set. Australian Personal Computing 1(4) p14 lists these codes, but many readers may be unaware of the previous history.

The second source for benchmark checks was given by Tom Fox in the June 1980 issue of Interface Age, where an (intentionally) crude prime number routine was given as an exerciser for the basic BASIC integer functions. This benchmark has, in spite of its simplicity and crudity of code (perhaps even *because* of it.) attracted numerous further reports by readers in subsequent issues of that magazine.

The opportunity has been taken to run a range of different compilers on a number of different machines. Where possible, arrangements were made to run the full set of nine benchmarks on the same machines for a number of different languages, machine clock rates and ancillary processors. The systems used for standardised comparisons were, where at all possible to arrange, an APPLE][, or an IEEE-696 S-100 system with a Z80 cpu

and a dual processor *Pascal-100* cpu board. The APPLE][is well known, and contains a 6502 central processor. Also available for the APPLE is a Z80 card and a 6809 card. These two cards fulfil somewhat different functions. The Z80 makes the CP/M operating system available to the APPLE user, and relegates the 6502 to keyboard, screen and disc handling. The 6809 card can operate at full speed at the same time as the 6502, and can be used to develop and run ROM-based BASIC programs at the same time that the 6809 is doing a task. Although OS-9 and FLEX-09 are now available for the 6809 card in much the same way as the Z80 card provides access to CP/M, the most common use for the 6809 is to run a high speed P-code interpreter and thereby speed up APPLE-Pascal both by handling the P-code interpretation task more efficiently and by using the 6502 to handle disc and serial I/O asynchronously. The 6809 can therefore operate as a hardware floating point processor for APPLE-Pascal by taking advantage of the 6809's 16-bit multiply instructions.

The *Pascal-100* is a dual processor on two linked IEEE 696 (S-100) boards, produced by Digicomp Research in Ithaca NY, USA and handled in Australia by *Oxford Systematics*. On the first board is a Z80 and a high speed bipolar memory mapping system. On the second is a Western Digital Microengine chipset, a sequence controller to handle the control between the two processor boards and a variable clock rate control module. The memory map is used to provide full access to a megabyte of memory to both processors, and to provide a parity error interrupt facility The Microengine can address 128 kb directly, and do direct I/O or hand over to the Z80 to do it. The UCSD III operating system is provided with the ability to use the CP/M BIOS for I/O, and to allow the Microengine (or the Z80) to offer service functions to the other processor. The Microengine chipset implements UCSD III P-code in silicon, including multitasking primitives and 32-bit floating point instructions. In the Western Digital computers based on this chipset, the normal clock rate is 2 MHz. The *Pascal-100* normally operates at 25 MHz [and has been set up at *Oxford Systematics* to run reliably at 3 MHz]. The clock rate for both processors is set by a plug-in crystal module, and as a result any runs requiring the Microengine chipset could be repeated at 2, 25 and 3 MHz.

The Z80 runs at 4 MHz were carried out using an Ithaca MPU 80-II Z80 cpu. The memory board used for as many tests as possible was an Ithaca Intersystems KDR64 dynamic RAM board, running in partially-latched mode without wait states, and with 8/16 bit addressing enabled.

The third special processor available for testing was a 4MHz AMD 9511 floating point processor chip. This was mounted on a Godbout SS-1 System Support Board, at a clock rate set on board and therefore independent of the cpu controlling the Bus. Some of the languages available for CP/M offer "9511 support". This is not always very well implemented, but for MicroSoft Fortran there are several complete Libraries available. Memtech, Video Vector Dynamics and Redding Group all have libraries for sale. The Redding library is in regular use at *Oxford Systematics*, and the benchmark results in the accompanying Tables demonstrate the reason for this choice.

Wherever possible, these machines were used to run the benchmarks, so that same system could be used to provide a relative efficiency test scale with the machine and operating system fixed. This is in contrast to the published results of Coll and Fox, where the 'same' code had been run on a range of machines and compilers, with few or none of the results held to a strictly comparable set of conditions beyond the code itself.

Concerted efforts have been made for this paper to extend the conditions of more of the runs to allow stricter comparisons. However, even approximate comparisons are very effective in gaining a sense of proportion , and have therefore been included and extended for this purpose.

The benchmarks themselves are very simple, and are designed explicitly for lucidity in interpretation of the results. The credit for this should of course go to the original authors. It should be emphasised that the previously published benchmarks were in BASIC only. The simplicity of the benchmarks might have been expected to lead to an easy translation to other languages_ but such was not the case. The 8080 APL implementation to hand took 561 seconds for BM9 - the Prime Number routine - is perhaps the least transparent recoding, but the same (enforced) variations produced to satisfy each language were applied to *all* the different compilers of that type. The major exceptions were for Pascal and its' variants. The label construction was enabled for Pascal and Structured-Algol, which - while not in the spirit of either language - was certainly there in the

flesh to be exploited in the name of strict comparability!

BENCHMARK SPECIFICATIONS

Benchmark BM1 : A null-action FOR, REPEAT or DO loop, executed 1000 times.

Benchmark BM2 : A null-action explicitly-coded loop executed 1000 times.

Benchmark BM3 : BM2 plus A=K/K*K+K-K in the loop.

Benchmark BM4 : BM2 plus A=K/2*3+4-5 in the loop.

Benchmark BM5 : BM4 plus a branch to null-action subroutine from inside the loop.

Benchmark BM6 : BM5 plus an array declaration M(5), and a null-action FOR loop (of 1-5) also in the loop.

Benchmark BM7 : BM6 plus M(L)=A in this 1-5 loop.

Benchmark BM8 : A square function, log function and sin function in an explicitly-coded FOR loop, repeated 100 times.

Benchmark BM9 : Prime numbers in the range 1-1000 are printed to the screen, calculated in an outer loop of 1000 and an inner loop of 500, with *no* tricks at all. This is a very bad prime number routine indeed, but a very useful basis for inter-machine, interpreter and compiler comparisons.

The output of numbers to the screen required by BM9 can slow the execution speed of this benchmark. The "standard" screen speed was therefore set to be 19,200 baud wherever possible. In some cases of fast execution the screen handling overhead proved to be a major delay factor. The Z8000 4MHz AMC-C integer benchmark runs suffered up to 25% delay under some situations.

A disc-oriented benchmark was considered, but the consequent dependence on disc controllers, disc formatting, and BIOS coding efficiencies clearly introduced too many variables for a useful benchmark to be constructed. Had one been set up solely for the *Oxford Systematics* systems, the very limited availability of strictly comparable benchmark runs would have been still more severely constrained. It was therefore - regretfully - omitted.

INTER-LANGUAGE COMPARISONS ON A PDP LSI-11

The DEC PDP-11 range of systems are widely used, and have been available for a considerable time. The smallest in the range is the LSI-11, and the benchmarks were coded and run on a *terak* 8510a to provide a startpoint the project. The *terak* has the KV-11 and FIS options, and is a an effective small scientific processor. The *terak* was used at the University of San Diego for the UCSD project, and the results of RT11 Fortran 4, UCSD 1.5 Compiled (P-Code) BASIC, and DEC 8k and Multiuser BASICs are given in Table 1. The results show the 2 : 1 relative speeds of semi-interpreted and compiled BASICs quite well , and the excellent showing of the semi-interpreted UCSD 1.5 Pascal compiler is an accurate harbinger of later findings.naturally, the 'integer' format of the code for all of the functions barring BM8 permitted by Pascal was of material assistance in improving the execution speed of benchmarks in Pascal verses other languages. (The Stuctured - Algol used later on permitted both integer *and* floating point versions for BM9, and the effects are discussed later).

Compiled RT-11 Fortran gave the best results by a sound margin, as one would have expected.

BASIC BENCHMARKS ON A 4 MHz Z80

Table 2 contains the first set of strictly comparable results. These are all BASIC benchmarks run on the same

system. The differences between the different releases of the same interpreter are particularly interesting. The CBASIC and EBASIC runs are very slow, but it must be noted that these are semi-compilers designed for business use and therefore do not show up well in such straight numerical processing speed comparisons. The price is paid most heavily on BM8, the most calculation-intensive benchmark.

The BASIC-E results were obtained with the version of the BASIC-E interpreter set up by Memtech to use an AMD 9511 for the floating point routines in the RUN.COM interpreter program. The results are to make the BASIC-E benchmark run time comparable to those for the Microsoft interpreters, for all the benchmarks other than BM9. The clock rate of the Z80 was held at 4 MHz for all these runs, and the 9511 was also operated at that speed. It is clear that the benefits to CBASIC users of such an accelerated ".INT" file interpreter are substantial. The increase in speed is well worth having, and it is not necessary to convert all of the CBASIC BCD real numbers to the limited precision of the 9511's floating point format to obtain this speed.

The advantages to sellers of CBASIC software in precompiled form are also clear, as no change is needed to this code. This device of using specialised hardware to accelerate the execution of semi-compiled software is explored further in this paper, and is of increasing importance as even the Microsoft Basic native code compiler now operates under a run time system, and is thus amenable to such assistance.

The device of interpreting the various p-codes (and other intermediate codes) produced by one machine by another - faster - one in the same bus is also becoming established as a useful tool, and could perhaps stem some the severe cost penalties currently being applied by SofTech to existing UCSD p-system users when upgrading to the currently supported release of UCSD, and when changing processors in the same computer system.

FORTRAN BENCHMARKS AND THE 9511

Table 4 contains the results of a range of Fortran benchmark runs. The salient features are most apparent in Figure 2, which links groups of Fortran runs together. The CDC Cyber 171 runs using the Fortran 4 "FTN" compiler (run without Trace , and at optimisation level 1) provide a point for mainframe user reference. The timings on the Cyber rely on the results reported by the system fro the execution of a whole program, and so are very unfavourable to the Cyber , as the loading and initialisation times are a significant part of the very short overall times reported for Pascal, Fortran and BSAIC. The Cyber results vary by 5-15% for successive runs, due to the maner in which the system operates, and so the Cyber results are included more for a sense of perspective than as precise and exact values strictly comparable to the other results in *all* respects.

The LSI-11 *terak* 8510a (which has a floating point FIS board fitted as standard) is shown as a point of reference for mini users. The effects of different Z80 clock rates are shown in each set of runs. The DEC PDP 11/40 (without EIS or FIS), running RT11 V02 Fortran, is also given to pace the *terak*. Although the Bm1-9 benchmarks suggest a 2 : 1 advantage in speed in favour of the 11/40.... the *terak* is slightly faster than the larger machine when running the large TRANSYT-7 traffic signal network optimisation program for practical and applied everyday traffic engineering.

The Perkin-Elmer 32/40 is a very popular 32 bit minicomputer, and several different Fortran compilers have been used to explore the range of performance which can be expected from such a machine on this very restricted set of small single-user, single-task benchtests. The Hewlett-Packard HP3000 provides a further mini reference point, running very close to the Cyber 171.

The results for Microsoft Fortran are closely comparable with those for UCSD II Fortran 77. In all cases the Fortran has been run as specified in the Appendix, and so the 2-byte integers used as a default assignment for loop index values will have been invoked----rather than the slight but important speed-up of 50% in loop index operations available by simply assigning INTEGER*1 to loop indices.

UCSD Fortran is semi-interpreted on the Z80, and it is rather surprising that the results are so good when

compared with the Microsoft compiler. The APPLE][results are also in the same range, which is not surprising as the APPLE operates the add-on Z80 card at about 2 MHz. The showing of the 6502 APPLE][-Fortran (UCSD, compiled for the APPLE's native 6502 cpu), is very good. The 6809 points are the result of using the add-on 6809 card to run an interpreter in 6809 code. The fixed point interpreter (sold in Australia as the "Pascal Speed-Up Kit"), clearly does *not* speed up the Fortran.... In fact it slows it *down* by a considerable amount. Note that the benchmark concerned is the essentially fixed-point BM9, and not the more demanding BM8. However, once the floating point update to the 6809 speed up kit is used the speed improves drastically- even for BM9.

The 6809 is acting as a limited form of floating point accelerator for the APPLE, and a major increase in speed could be expected by the addition of a 9511 and the appropriate changes to the UCSD interpreter (which contains the trancendental functions). It should be noted that the 6809 is used solely to speed up the p-code produced by the 6502 compiler.

The slowest Fortran execution times were obtained on a Motorola Exorciser with a 1 MHz 6800 cpu under Motorola Fortran 2.2. Although this is a very low clock rate, the 6809 card for the APPLE operates the 6809 at this speed, and some direct comparisons are possible. The Motorola floating point function library is clearly very inefficient, as can be seen by internal comparisons between the benchmark results.

The 9511 has attracted substantial support from Fortran users, as a result of the availability of floating point processor versions of FORLIB. A detailed discussion of these libraries is in preparation, but suffice to say here that the coverage of the number of functions in FORLIB and the efficacy of the coding used differs substantially between the commercial libraries. The results are apparent in Figure 2 : the Memtech library provides a notable increase in speed, but the Redding Group library provides a further factor of 4 in speed again. A salutary demonstration of the importance of efficient coding, even when a considerable amount of raw power is available (from the 9511). It is no accident that *Oxford Systematics* handles the Redding Group products....The 4MHz Z80 BM1-8 results are shown on Figure 3 , and illustrate clearly how effective a 9511 would be were it to be integrated into the UCSD p-system interpreter. The recent announcement of Native Code compilers for speeding up key parts of UCSD p-systems using Z80, 8080 and 8086/88 processors raises the question of UCSD Fortran taking the performance lead from Microsoft Fortran for at least some types of applications. This may yet press Microsoft to produce the long-awaited complex-number extensions to their Fortran.

PASCAL BENCHMARK RESULTS

The extensive use of Pascal is now becoming widely felt, and the Pascal versions of the benchmarks are therefore of special interest. The widening influence of UCSD Pascal is apparent from the recent adoption of compatible Pascal systems by Texas Instruments, Hewlett-Packard and Phillips Industries. Pascal has also provided some of the fastest execution times for the whole benchmark family presented here. Table 4 summarises the results, and some of the fastest results obtained are given (for BM1-8 only) in Figure 4.

The fastest results obtained for BM8 are all clustered closely together. The Pascal MT+ compiler offers "9511 support", but in fact only uses the 9511 for the trancendental functions. Consequently the BM8 comparisons are strictly the only ones that can be made on a comparable basis. The competition for the lowly Z80 is remarkable. The CDC CYBER 171, the PERQ and the HP 9836. The PERQ is a 48-bit wide microcoded Pascal machine designed by Three Rivers Corporation in Pittsburgh, and now marketed by ICL.

The major strength of the PERQ is in its high quality graphics rather than sheer processing performance: there are microcoded instructions for speeding up the raster display for example, but it clearly performs most effectively as a very fast mini computer. The results are uniformly 10-15% faster than those for the HP 9836, but for BM9. The HP 9836 is a recent HP product based on a Motorola 68000 operating at 8 MHz, running a native code compiler closely compatible with UCSD Pascal and its extensions.

Neither the HP 9836 nor the PERQ have dedicated hardware floating point processors, nor does the Pascal-100, which, running at a slow 3MHz is still delivering comparable performance to the dedicated 9511 and the other

larger and 'more powerful' systems. The point made in the last section about UCSD support for the 9511 takes on a new importance when applied to this system. The Pascal-100 addresses 128 kb directly, and uses both the compact p-code directly and also has excellent multitasking facilities to take full advantage of associated processors. The standard Western Digital product known as the Microengine (also sold in a packaged form in Australia as the Ortex System-1, with some improvements) is considerably slower than the Digicomp S-100 version: however, both Ortex and Digicomp are now integrating the 9511 into their product and operating system. The results should be impressive, and should narrow the gap between these two MicroEngine chipset-based implementations.

When compared to Fortran, the UCSD Pascal results are very much faster, however the Pascal MT+ results show clearly how much the numerical processing support library for this product has been improved, although Microsoft Fortran is evidently still the leader in the software floating point functions supplied.

Once again the APPLE][+ shows up very well, especially with the 6809 in support. Unfortunately long delays at Softech meant that *no* Version IV UCSD results could be obtained for any machine. The APPLE results for BM8 are given in Figure 5 for *all* the cpu's and languages assessed. The relative effectiveness of the UCSD semi-interpreters, the Microsoft compilers and interpreters, the APPLE 6502 BASICs and the two APPLE-BASIC compilers (TASC and Expediter) show a steady advance in speed over the interpreted code, and the Z80 and the combination of the 6502 and the 6809 produce results more closely equivalent than could reasonable have been expected.

STRUCTURED ALGOL AND THE 9511 IN SUPPORT

The structured ALGOL produced by Morrison and Cole at the University of St Andrews has been used to examine the effects of the addition of a 4MHz AMD 9511A in the bus. S-Algol is an intermediate (s-code) compiler. Consequently, both software and hardware floating point interpreters can therefore easily be set up. The Digicomp P-100 processor was used to run the Z80 codes, and the plug in clock crystal feature permitted the use of 2, 2.5, and 3MHz clock rates for the Z80, in addition to the 4MHz Ithaca cpu. The results are tabulated in Table 5, and are shown in graphical form in Figures 5 and 6. It is of great interest to note that the software arithmetic benchmark time drops off quickly to 3 MHz, and then stabilises. The addition of a polled 9511 to service a hardware floating point interpreter led to very large increases in overall speed, but once again to a sharp flattening off of performance beyond 3MHz. The huge range of run times give a good feel for the overheads of different interpreters and compilers, and a rather less precise sense of the relative speeds of execution. Once again, the considerable increase in performance obtainable by applying a specialised processor to speed up execution of an intermediate code compiled file shows its worth. The P-100 Microengine is itself one of the best examples of this philosophy, as the UCSD III.0 P-codes are set up in microcode to provide a literal Pascal-*Engine*. The effectiveness in the application of the 9511 in this way is in marked contrast to the selective implementation of transcendental functions alone adopted by Pascal MT+ as the sole integrated use of this powerful specialised chip.

CONSOLIDATED APPLE][+ BENCHMARKS

The APPLE][+ is such a popular computer that many people do not realise that it is one of most widely - used *bus* systems available. The 6809 and Z80 processor boards already referred to a far from the only additional processors in the APPLE: 8088, 8086 and 68000 systems are already widely available.

It is therefore of interest to see how the mutual choice of language and processor works out. Table 6 lists the consolidated results of *all* the APPLE][+ - based results : these results are also given in a graphic form in Figure 7. The major problem with the APPLE is the small size of the standard discs (144k, on 16-sector single-sided 5" floppy discs): from these results is is evident that the addition of more advanced processor cards in the APPLE bus, running fast 6502 p-code interpreters, could materially increace the utility of the APPLE without having to licence new versions of the UCSD system from either APPLE or SofTech, and without having to re-purchase new versions of applications software. The recent implementation of UCSD II on the DEC VAX systems by Edinburgh University may well be the harbinger of a wave of further such systems. It is unfortunate that it has not yet proved to be possible to run these benchmarks on the BRIDGE CP/M 2.2 emulation

system on the VAX, as this would provide a very useful sense of perspective on this question at a further level of generalisation beyond that of simple p-code interpretation acceleration.

THE BM9 BENCHMARK FROM ALL SOURCES

To provide some continuity between the results reported here and those of previous workers, it is useful to provide all of the available BM9 results in a single place (Table 7). The "*" results were repeated (or added to) for this paper, but all of the others are quoted from other sources. Some puzzling anomalies arise (PRIME 300, 550 speeds for example), but in general the results have been confirmed where they have been rerun, and are reasonably consistent where they have not. The BBC Micro shows up very well against the APPLE][+, and shows the rate of advance in systems at the lower end of the market. Unfortunately it has not yet proved to be possible to obtain and independent confirmation of the excellent results reported by Seattle Systems for an 8 MHz 8086, as this processor has taken an early lead in software and system availability.

At the other end of the scale the TRS-80 Pocket Computer makes it quite clear why the pocket calculator market has not [yet] had to react to competition from this quarter: a simple coding of BM9 on an HP 11C gave figures comparable to the faster microsystems in Table 7 rather than the painfully slow execution time of 55830 seconds for the Pocket Computer.

The DEC results for BM9 are of particular interest, as the PDP-11 series has such a wide following. The PDP 11/45 comes in at about the same speed as BASIC compilers on the APPLE or a 4 MHz Z80, while the LSI-11 (*terak*) is approximately equivalent to either of these two micro systems when all three are running interpreted BASIC codes.

The lethargy of the Western Digital p-code compiled BASIC on a 2MHz Microengine is quite remarkable: the Microengine and the *terak* are both based on the CP1600 chip with different Microcode, and it is interesting to see that the early UCSD p-code compiled BASIC 1.5 on the *terak* is slightly faster than the PDP 11/45, and the same speed as interpreted Microsoft BASIC on the 8 MHz Seattle 8086! This might suggest that Western Digital would be well advised to return to the UCSD II.1 BASIC Compiler (which also had the advantage that it could easily be extended by Pascal functions) to give something a little closer to the excellent performance delivered by the Microengine chipset in Pascal.

The complementary BASIC results for BM1-8 are collated in Table 8. The low speed of the WD BASIC is particularly marked in this company, and it is evident that the costs of subroutine branching and array addressing in the WD BASIC are far from impressive: only BM8 reflects the speed inherent in the full 32-bit microcoded floating point arithmetic available to the compiler. This is also one of the BASIC systems where the FOR loop is very much slower than an explicitly coded counter and conditional loop back!

A number of other similar comparisons between the implementation efficiencies of the limited range of constructs covered by the BM1-9 benchmark set can be drawn from Tables 7 and 8.

SUMMARY

The overall results of this systematic review are as follows:

1) The simplicity of the benchmarks employed did not fail to bring out the fundamental differences between different languages, and different implementations of the "same" language.

2) The performance envelopes of a wide range of computer compilers, interpreters and systems can usefully be summarised by a simple set of in-core benchmarks.

3) The effectiveness of specialised floating point assistance is substantial, and the use of well constructed Floating Point Support libraries can produce remarkable results from 8-bit processors.

4) The contribution of special-purpose high speed interpreters for widely available semi-compiled languages

has a major unrealised contribution to make, especially in accelerating the acceptance and use of newer 16-bit processors, and the transfer of "secure" (i.e. compiled) code between mainframe, mini and micro systems.

ACKNOWLEDGEMENTS

To set up such a systematic set of comparisons requires the active cooperation of a considerable number of people and organisations. Owing to the wide range of software and hardware needed for this exercise, it is impossible to list all those who have helped. Special mention should however go to Iris Hocking, to Arpad Balaton of Howden and Wardrop, Brian Cockburn of Barson Computers, ComputerLand South Melbourne, the Victorian TAB, Philip Freidin of R&D Electronics, and to the MMBW, International Computers, Hewlett-Packard and the Australian Road Research Board for their assistance in producing these results in time for this paper.

APPENDIX 1 : DOCUMENTARY, SYSTEMS AND SOFTWARE SOURCES

1.1 BASIC Benchmarks

a) BM1-7 Anon. (1977) Kilobaud Microcomputing (June)

b) BM1-8 Coll, J. (1978) Computing (Europe) (27 July)

Anon. (1981) Australian Personal Computing 1(4) 14

c) BM9 Fox, T. (1980) Interface Age (June)

1.2 SPECIALISED ENVIRONMENT SOURCES

a) 6809 APPLE][Card "The Mill" by Stellation Two, Santa Barbera
Supplied by: *Oxford Systematics*

b) 6809 Pascal Speed-Up Kit For the Mill
Supplied by: *Oxford Systematics*

c) 6809 Pascal Speed-Up kit: Floating Point Update For The Mill
Supplied by: *Oxford Systematics*

d) AMD 9511 A 4 MHz 16/32 bit floating point processor chip
Supplied by: R&D Electronics

e) REDDING APULIB Microsoft 9511 Floating point FORTRAN library replacement.
Supplied by: *Oxford Systematics*

f) MEMTECH APUFLIB Microsoft 9511 Floating point FORTRAN library replacement.
Supplied by: Memtech, USA.

g) BASIC-M BASIC-E interpreter modified to use 9511 support.
Supplied by: Memtech, USA.

h) SOFTRONICS APL V2.02 An 8080 CP/M interpreter.
Supplied by: *Oxford Systematics*

i) S-ALGOL Structured Algol with hard and soft floating point interpreters.
Supplied by: *Oxford Systematics*

j) PASCAL-100 Z80/WD Microengine S-100 (IEEE-696) Dual processor with UCSD III Operating system
Supplied by: *Oxford Systematics*

k) Ithaca Intersystems Z80 MPU-II, 64KDR, FDC-2: IEEE-696 CPU,RAM,NEC 765 Disc Cards.
Supplied by: Melbourne's Byte Shop.

1.3 S-ALGOL REFERENCES

MORRISON, R. (1979). S-ALGOL Reference Manual. Department of Computational Science Report CS/79/1. University of St Andrews, Scotland. (75p).

COLE, A.J. and MORRISON, R. (1980). An Introduction to S-Algol Programming. Department of Computational Science Report CS/80/1. University of St Andrews, Scotland.

TABLE 1 RESULTS FOR THE INTERFACE AGE BM9 BENCHMARK ON A *terak* 8510a

Operating System	Language	Time in Secs
RT 11 V04	DEC FORTRAN IV V03	120
UCSD 1.5	UCSD Pascal 1.5	188
UCSD 1.5	UCSD BASIC 1.5	310
RT 11 V03	8k BASIC	596
RT 11 V03	MuBASIC	703

[Oxford Systematics June 1982]

TABLE 2 BENCHMARKS BM1-8 AND BM9 ON A 4MHz Z80A

Software	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	BM9
Microsoft:									
5.03 Compiler	.5	.6	4.6	2.3	2.3	4.6	17.3	6.1	277.5
5.2 Interp.	1.5	5.1	13.6	13.6	14.5	15.1	39.5	6.2	966.3
4.51 Interp.	1.6	4.7	12.5	12.5	13.4	16.1	38.4	6.6	877.0
BASIC-E									
With 4 MHz 9511	2.9	3.4	8.6	7.9	8.3	23.8	38.3	1.2	2208.0
CB80									
V1.3	3.8	3.8	18.5	28.1	28.1	51.8	55.8	49.9	1988.5
CBASIC:									
V06 Semi.Int.	5.7	10.4	38.4	61.0	61.5	53.0	62.0	79.0	3100.0

[Oxford Systematics June 1982]

TABLE 3 FORTRAN VERSIONS OF THE BENCHMARKS BM1-9

System	O/S	Compiler	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	[7+8]	BM9
HP3000	.	FTN	.01	.01	.04	.03	.055	.09	.18	.01	[.19]	8.0
Cyber171	NOS 1.4	FTN Opt=1	.06	.07	.08	.08	.09	.13	.14	.08	[.22]	2.5
P-E 32/40	FTN7(Opt:used)		<.03	<.03	<.03	<.03	<.03	<.03	.27	<.03	[.27]	41.0&
P-E 32/40	FTN7(Opt:un ")		<.03	<.03	.27	.27	.30	.33	1.50	5.2	[6.7]	75.0&
P-E 32/40	FTN7(Devt.)		<.03	<.03	.30	.30	.33	.33	1.50	5.4	[6.9]	102.0&
Z80 4Mhz	Redding	9511(4M)	.02	.03	.48	.47	.48	.59	1.60	.31	[1.9]	51.6
Z80 3Mhz	Redding	9511(4M)	.02	.04	.52	.51	.51	.63	1.72	.32	[2.0]	54.4
Z80 2.5	Redding	9511(4M)	.03	.04	.57	.56	.57	.72	1.94	.33	[2.3]	62.7
Z80 2Mhz	Redding	9511(4M)	.04	.06	.73	.70	.73	.90	2.41	.36	[3.1]	75.7
PDP 11/40	no FIS,FTN v02		.03	.04	.34	.35	.46	.61	1.11	.8	[1.9]	60.2
terak[LSI-11]+FIS,FTN v03			.08	.08	.67	.64	.92	1.11	2.76	2.6	[5.4]	120.0
Z80 4Mhz	Memtech	9511(4M)	.02	.03	.48	.46	.46	.58	1.95	.35	[2.3]	148.2
Z80 3Mhz	Memtech	9511(4M)	.02	.03	.51	.49	.50	.62	2.08	.38	[2.5]	156.0
Z80 2.5	Memtech	9511(4M)	.03	.05	.60	.58	.66	.74	2.43	.40	[2.8]	186.6
Z80 2Mhz	Memtech	9511(4M)	.04	.06	.70	.67	.70	.87	2.97	.43	[3.4]	230.7
Z80 4Mhz	CP/M2.2	MsftFTN	.02	.04	1.4	1.4	1.4	1.5	5.6	3.3	[8.9]	286.1
Z80 3Mhz	CP/M2.2	MsftFTN	.02	.04	1.5	1.5	1.5	1.6	5.9	3.4	[9.3]	301.3
Z80 2.5MHz	CP/M2.2	MsftFTN	.03	.05	1.8	1.8	1.8	2.0	7.1	4.1	[11.2]	361.1
Z80 2MHZ	CP/M2.2	MsftFTN	.04	.05	2.3	2.2	2.3	2.4	8.9	5.1	[14.4]	451.2
Z80 4Mhz	UCSD II	FTN77	.4	0.5	1.5	1.8	1.8	4.9	6.4	5.6	[11.9]	279.3
Z80 3Mhz	UCSD II	FTN77	.5	0.5	1.5	1.8	1.8	5.1	6.8	5.9	[12.7]	295.8
Z80 2.5MHz	UCSD II	FTN77	.6	0.6	1.8	2.2	2.2	6.1	8.1	7.1	[15.2]	354.7
Z80 2MHz	UCSD II	FTN77	.7	0.6	2.3	2.8	2.8	7.6	10.1	8.9	[18.0]	443.5
APPLE][+ Z80	MsftFTN		.1	0.1	2.5	2.5	2.5	2.5	10.5	5.2	[15.7]	445.7
APPLE][+6809 1Mhz#	FTN771.1		.46	.43	1.61	1.56	2.18	4.96	6.64	4.3	[10.9]	313.1
APPLE][+6809 1Mhz*	FTN771.1		.47	.44	1.71	1.65	2.29	5.03	6.72	8.2	[15.0]	540.0
APPLE][+6502 1Mhz	FTN771.1		.7	.8	3.6	3.5	4.3	8.3	14.2	7.9	[22.1]	474.0

* With display ON for 6809-6502 switching, and NO 6809 floating point used.
\# With display ON for 6809-6502 switching, and USING the 6809 floating point.
& While running a multitasking O/S with multiterminal monitor: note I/O o/head
[Oxford Systematics June 1982]

TABLE 4 PASCAL VERSIONS OF BENCHMARKS BM1-9

System	MHz	Compiler	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	[BM7+8]	BM9
CDC171	...	P-6000	.01	.21	.26	.26	.26	.31	.32	.27	[.6]	.37
PERQ	...	PERQ	.01	.01	.04	.04	.07	.13	.17	.51	[.7]	9.0
HP9836	8.0	HP-Pasc.	.02	.02	.11	.07	.10	.15	.25	.75	[1.0]	9.2
P-100	3.0	UCSD.F2	.05	.05	.18	.18	.23	.56	.88	.59	[1.5]	21.0
P-100	2.5	UCSD.F2	.06	.06	.23	.21	.28	.68	1.05	.71	[1.8]	25.5
M/Engine	2.0	UCSD.H1	.09	.09	.26	.25	.32	.78	1.22	.76	[2.0]	28.5
P-100	2.0	UCSD.F2	.08	.08	.29	.28	.35	.85	1.32	.88	[2.2]	31.6
Z80+9511	4.0	MT+v5.5 <	.01	.06	1.09	.8	.9	1.36	1.53	.56	[2.1]	99.2
Z80	4.0	&MT+v5.5 <	.01	.06	1.09	.8	.9	1.31	1.55	4.74	[6.3]	99.2
Z80	4.0	&MT+v5.1 <	.1 <	.1	1.0	.9	.9	1.2	1.5	81.4	[82.9]	104.4
Z80	4.0	P/Z 3.2	.3	.3	.7	.5	.8	1.0	3.9	12.2	[16.1]	106.7
Z80	4.0	UCSD II <	.5 <	.5	1.5	1.9	2.1	4.5	6.7	5.3	[12.1]	143.5
Z80	2.5	UCSD II	.5	.5	1.9	2.8	2.8	5.8	8.5	6.7	[15.2]	183.5
Z80	2.0	UCSD II	.6	.6	2.2	2.7	3.5	7.1	10.0	8.0	[18.0]	229.3
APPLE][	6809	#UCSD1.1	.46	.44	1.67	1.57	2.17	4.9	7.2	3.8	[11.0]	176.2
APPLE][	6809	*UCSD1.1	.45	.41	1.75	1.66	2.2	5.0	7.3	7.7	[15.0]	187.5
APPLE][	6502	UCSD1.1	.60	.54	2.80	2.76	3.6	7.3	11.0	7.1	[18.1]	227.8

&=FPREALS used (Software floating point) *=NO floating point #=Floating point *[Oxford Systematics June 1982]*

TABLE 5 ALGOL AND STRUCTURED ALGOL VERSIONS OF BENCHMARKS BM1-9

S-ALGOL CODINGS	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	[BM7+8]	BM9	BM9I
Z80 2.0 SOFT	.30	1.1	12.1	12.6	13.4	15.5	19.3	5.5	[24.8]	765.1	498.6
Z80 2.0 9511 4MHz	.30	1.1	5.3	5.4	6.1	8.2	12.1	.6	[12.7]	395.1	339.6
Z80 2.5 SOFT	.23	.9	9.7	10.1	10.7	12.3	15.5	4.3	[19.8]	612.1	398.8
Z80 2.5 9511 4MHz	.23	.9	4.3	4.3	5.0	6.5	9.7	.5	[10.2]	318.1	271.8
Z80 3.0 SOFT	.19	.8	7.9	8.4	8.9	10.3	13.6	3.7	[17.8]	510.6	332.6
Z80 3.0 9511 4MHz	.19	.8	3.6	3.6	4.2	5.5	8.1	.5	[8.6]	267.0	228.8
Z80 4.0 SOFT	.18	.7	7.5	8.0	8.4	9.7	12.2	3.5	[15.7]	483.6	313.3
Z80 4.0 9511 4MHz	.18	.7	3.4	3.5	3.9	5.3	7.6	.5	[8.1]	252.5	215.6

[Oxford Systematics June 1982]

TABLE 6 COLLATED APPLE][+ BENCHMARKS BM1-9

CPU	LANGUAGE	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	[BM7+8]	BM9
6809	#Pascal 1.1	.46	.44	1.67	1.57	2.17	4.88	7.2	3.8	[11.0]	176.2
6809	*Pascal 1.1	.45	.41	1.75	1.66	2.2	5.0	7.3	7.7	[15.6]	187.5
6502	Pascal 1.1	.6	.54	2.8	2.8	3.6	7.3	11.0	7.1	[18.1]	227.8
6809	#F77 1.1	.46	.44	1.67	1.57	2.17	4.9	7.2	3.8	[11.0]	313.0
6809	*F77 1.1	.45	.41	1.75	1.66	2.2	5.0	7.3	7.7	[15.0]	520.0
Z80	MsftFTN	.1	.1	2.5	2.5	2.5	2.5	10.5	5.2	[15.7]	445.7
6502	F77 1.1	.6	.54	2.80	2.76	3.6	7.3	11.0	7.1	[18.1]	475.5
6502	TASC	.6	.8	4.6	5.3	5.4	10.4	16.0	9.0	[25.0]	325.2
6502	Expediter	1.2	1.1	4.9	5.5	5.5	10.8	13.0	9.0	[22.0]	335.0
6502	IntBASIC	1.5	3.2	7.3	7.2	8.9	18.6	28.2	.	.	721.6
6502	Applesoft	1.3	8.5	16.0	17.8	19.1	28.6	44.8	10.1	[55.5]	970.0
Z80	GBASIC5	2.1	6.6	18.8	18.6	20.2	35.6	56.6	10.1	[66.7]	1284.0

Floating point 6809 interpreter * Fixed point 6809 interpreter *[Oxford Systematics June 1982]*

TABLE 7 INTERFACE AGE BM9 BASIC BENCHMARK FROM ALL SOURCES : Part 1

System	CPU	MHz	O/S	Language	Run Time
CDC CYBER	171	.	NOS 1.4	BASIC	5*
IBM	3033	.	VS2-10RVYL	Stanford BASIC	10
PRIME	300	.	PRIMOS	BASIC	25
Seattle System 2	8086	8	MS-DOS	MsB(Compiled)	33
DEC	11/70	.	RSTS/E	BASIC	45
PRIME	550	.	PRIMOS	BASIC V16.4	63
DEC	PDP-10	.	TOPS-10	BASIC	65
IBM	S/34	.	R-05	BASIC	129
Digital Microsystems	HEX-29	6	HOST	HBASIC+	143
HP	3000	.	.	BASIC	250
4MH z Z80A	Z80	4	CP/M 2.2	MsB(Compiled)5.03	277*
terak 8510a LSI-11	CP1600	.	UCSD 1.5	BASIC1.5 Compiler	310*
Seattle System2	8086	8	MS-DOS	BASIC	310
Alpha Micro AM100T	WWD16	3	AMOS 4.3A	AlphaBASIC	317
APPLE II+	6502	2	DOS 3.3	Microsoft TASC	325*
DEC	11/45	.	.	BASIC	330
APPLE II +	6502	2	DOS 3.3	Expediter II(Comp)	335*
Data General	NOVA3	.	Timeshare	BASIC 5.32	517
BBC Micro	6502	.	BBC Basic	BBC INTEGER Basic	523*
SWPTC	6800	.	Software Dyn	Compiler B 1.2	528
Alpha Micro AM100	WD16	2	AMOS 4.3A	AlphaBASIC	573
Technico SS-16	9900	3	DOS	SuperBASIC3	585
terak 8510a LSI-11	CP1600	.	RT11 V0.3	8k BASIC	596*
BBC Micro	6502	2	BBC Basic	BBC F/POINT Basic	596*
Ohio C4-P	6502	2	OS65D 3.2	LevelI BASIC	680
North Star FP	Z80	4	NSDOS	NS BASIC	685*
Terak 8510A	LSI11	.	RT11 V0.3	MUBASIC	703*
APPLE II +	6502	2	DOS	Integer Basic	722*
ADDS Multivision	8085	5	MUON	MBASIC 5.0(ADDS)	766
4MH z Z80A	Z80	4	CP/M 2.2	MBASIC 5.2	877*
Tandy TRS-80 II	Z80	4	TRSDOS 1.12	Level3 BASIC	955*
4 MH z Z80	Z80	4	CP/M 2.2	MBASIC 4.51	966*
APPLE II+	6502	2	DOS 3.3	APPLESOFT II	970*

* Repeated or run specifically for this paper

[Oxford Systematics June 1982]

TABLE 7 INTERFACE AGE BM9 BASIC BENCHMARK FROM ALL SOURCES : Part 2

System	CPU	MHz	O/S	Language	Run Time
Rexon RX30	8086	5	RECAP	Business B	1020
Cromemco	Z80	4	CDOS	Extended BASIC	1116*
North Star	Z80	4	NS DOS	NS BASIC	1149*
Processor Tech Sol-20	.	.	Solos	Altair BASIC 8k	1231
Exidy Sourcerer	Z80	4	.	Microsoft BASIC	1260*
ISC Compucolor CC-II	8080	.	.	BASIC	1267*
APPLE II+	Z80	2	CP/M 2	GBASIC	1284*
Ohio C3-C	6502	1	OS65D	Level I BASIC	1346
Commodore PET 2001	6502	.	.	Microsoft BASIC	1374
ISC Compucolor 8051	8080	.	DOS	BASIC 8001	1375*
Hewlett-Packard HP85	NMOS	.	.	BASIC	1380*
Basic/Four 600	8080	.	.	BASIC	1404
Micro V Microstar 1	8085	3	StarDOS	StarDOS BASIC	1438
Sinclair	Z80	2.5	.	4k BASIC	1514*
Processor Tech SOL-20	.	.	Solos	PT Extd BASIC	1812
Heath H89	Z80	.	.	Microsoft 4.7 B	1850
Zilog MCZ-1/70	Z80	2	RIO	Zilog BASIC	1863*
Tandy TRS Model 1	Z80	2	TRSDOS	Level II BASIC	1929*
IBM 5120	.	.	.	BASIC	1956
4MH z Z80	Z80	4	CP/M 2.2	CB80 v1.3	1988*
4MH z	Z80	4	CP/M 2.2	BASIC-E(M9511 4M)	2208*
Vector MZ	Z80	.	MDOS	Micropolis 8.5 B	2261
Digicomp P100-Z80	Z80	3	CP/M 2.2	BASIC-E(M9511 4M)	2322*
Cromemeco CS3	Z80	4	CDOS	CBASIC-2	2445
Texas Instruments99/4	9900	.	.	TI BASIC	2479
Ortex Microengine	CP1600	2	UCSD.H1	BASIC 1.1	3017*
4 MH z Z80A	Z80	4	CP/M 2.2	CBASIC V 2.06	3100*
Zenith H89	Z80	.	.	Benton Harbor B	3550
Pocket TRS-80	2x4CMOS	.	Resident	BASIC	55830*

* Repeated or run specifically for this paper

[Oxford Systematics June 1982]

TABLE 8 BASIC BENCHMARKS BM1-8 FROM ALL SOURCES : Part 1

System	O/S	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	[7+8]
Cyber 171	NOS 1.4	.024	.021	.046	.045	.071	.20	.305	.036	[.34]*
Wang 2200VP	Resident	.3	.7	2.2	2.3	2.5	4.1	5.3	1.0	[6.3]
IBM 370/115	LISBON1.2	1.3	1.4	2.3	1.9	2.1	3.1	4.6	1.9	[6.5]
DG Eclipse	RDOS	.5	1.8	4.3	5.0	5.1	9.0	13.0	1.5	[14.5]
HP 9825A	Resident	.7	1.8	4.0	4.4	4.7	9.0	12.1	3.7	[15.8]*
HP 9845	Firmware	1.1	1.5	3.4	3.6	4.3	10.2	14.2	3.6	[18.4]*
APPLE][+	Expediter2	1.2	1.1	4.9	5.5	5.5	10.8	13.0	9.0	[22.0]*
Z80 CP/M 4MHz	CompBASIC	.5	.6	4.6	2.3	2.3	4.6	17.3	6.1	[23.4]*
OLIVETTI	P6060 res	2.0	3.0	4.0	18.0	17.0	18.0	19.0	---	----
APPLE][+	TASC	.6	.8	4.6	5.3	5.4	10.4	16.0	9.0	[25.0]*
DG NOVA1220	TS V091	2.3	4.5	8.4	10.5	11.0	16.9	23.8	2.7	[26.5]
DG NOVA 1210	V0.02	1.0	3.0	7.7	9.1	9.6	16.5	24.0	2.6	[26.6]
DEC PDP8A	OS/8 R3	2.7	2.5	5.5	6.1	6.7	19.2	24.2	3.0	[27.2]
APPLE][	IntBASIC	1.5	3.2	7.3	7.2	8.9	18.6	28.2	---	---- *
PDP8E Educomp	V3. 4	1.4	4.4	8.5	11.8	9.3	15.8	27.8	3.7	[31.5]
DG NOVA1220	MU SOSR9	1.8	4.0	10.0	11.6	12.4	20.8	29.8	3.8	[33.6]
TRS-80 Mod II	Int BASIC	1.0	4.0	13.0	13.0	14.0	20.0	30.0	6.0	[36.0]*
TI 9900-10	Prelimin.	3.2	2.4	4.5	5.5	6.1	25.0	29.0	---	----
RM380Z 4 MHz	TDL V1.3	1.4	6.5	13.2	13.9	15.0	22.3	31.6	6.2	[37.8]
9511+Z80 4MHz	BASIC-E(M)	2.9	3.4	8.6	7.9	8.3	23.8	38.3	1.2	[39.5]*
TRS-80 Mod II	S/Prec B	1.0	5.0	13.0	13.0	14.0	23.0	35.0	6.0	[41.0]*
DG NOVA 2-10	R4 RDOS3	1.0	3.0	8.0	9.0	9.0	16.0	25.0	16.0	[41.0]
9511+Z80 3MHz	BASIC-E(M)	3.1	3.5	9.1	8.4	8.6	25.0	40.5	1.3	[41.8]*
Z80 CP/M 4MHz	BASIC5.2	1.6	4.7	12.5	12.5	13.4	16.1	38.4	6.6	[45.0]*
SuperBrain	Msft BAS	1.6	5.2	14.0	13.9	14.8	26.3	43.2	5.6	[48.8]*
DB NOVA1210	4k S.User	.6	1.8	5.8	7.3	7.6	12.5	18.2	32.2	[50.4]
Sinclair 2.5M	4k BASIC	1.5	4.7	9.2	9.0	12.7	25.9	39.2	-	[-]*
ALPHA LSI2-20	BASIC-2	2.2	7.0	14.5	15.0	16.0	29.5	48.0	3.7	[51.7]
Z80 2.5 Mhz	TDL8k1.3	1.8	8.5	18.4	19.5	21.6	31.6	44.7	8.9	[53.6]
SWPTC MP68	Altair 1	1.4	9.0	16.8	18.1	20.0	31.0	45.2	8.5	[53.7]
SWPTC MP68	3k Int	1.6	8.7	15.6	16.4	21.6	31.4	46.5	---	----
DEC PDP8M	EDSYST10	3.0	4.6	14.6	15.0	15.4	34.0	47.1	7.0	[54.1]
RM380Z	Tiny 1.8	.9	8.6	13.6	17.2	21.2	----	----	---	----
APPLE][	AsoftExB	1.3	8.5	16.0	17.8	19.1	28.6	44.8	10.7	[55.5]*
HP-85	HP BASIC	1.8	3.8	16.3	16.5	17.7	30.0	44.8	12.7	[57.5]*
BBC Micro	F/Pt Basic	.2	1.8	7.5	7.6	8.3	10.2	14.7	47.6	[62.3]*

* Repeated or run specifically for this paper

[Oxford Systematics June 1982]

TABLE 8 BASIC BENCHMARKS BM1-8 FROM ALL SOURCES · Part 2

System	O/S	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	[7+8]
CBM 8032	PetBASIC	1.7	10.0	18.4	20.3	21.9	32.4	51.0	11.9	[62.9]
PET 2001	Resident	1.7	9.9	18.4	20.4	21.7	32.5	50.9	12.3	[63.2]
Sourcerer	ROM Bas	1.8	10.0	20.7	22.2	24.3	37.6	53.7	9.6	[63.5]*
CompucolorII	ISC BAS	2.0	10.9	22.4	23.9	25.7	38.7	55.2	10.8	[66.0]*
APPLE][CP/M	GBASIC	2.1	6.0	18.8	18.6	20.2	35.6	56.6	10.1	[67.1]*
Cromemco III	Cr BASIC	1.9	5.7	16.4	19.7	21.3	32.4	44.1	22.9	[67.0]*
DEC PDP8L	8k BASIC	4.0	6.8	17.0	20.2	21.0	38.8	57.0	10.8	[67.8]
Altair 8800b	Ext 4.0	1.9	7.5	20.6	20.9	22.1	37.0	58.5	9.9	[68.4]
DEC PDP8L	4k Eds10	4.8	7.0	16.8	20.2	20.5	44.8	61.5	9.5	[71.0]
TRS-80 Mod II	D/Pr BAS	----	6.0	41.0	43.0	44.0	52.0	65.0	7.0	[72.0]
Wang 2200T	Resident	4.4	9.5	24.4	23.1	25.9	49.3	74.9	13.	[88.5]
Altair 8800b	NS V6 R2	2.4	9.0	16.8	31.3	33.4	50.1	72.5	22.0	[94.5]
Altair 680b	Altair1.1	2.6	16.4	30.9	33.7	36.6	56.0	81.9	15.0	[96.9]
Atari 400/800	Resident	2.3	7.4	19.9	23.2	26.8	40.7	61.5	43.1	[104.6]
4 MH z Z80A	CB80v1.3	3.8	3.8	18.5	28.1	28.1	51.8	55.8	49.9	[105.7]*
Tectronix4051	Resident	4.7	14.2	33.3	36.1	40.8	69.0	103.9	14.7	[118.6]
Texas TI99/4	Resident	2.9	8.8	22.8	24.5	26.1	61.6	84.4	38.2	[122.6]
Z80 CP/M 4MHz	CBAS2.06	5.7	10.4	38.4	61.0	61.5	53.0	62.0	79.0	[141.0]*
TRS-80 Mod I	Tandy BAS	2.5	18.0	34.5	39.0	45.0	67.0	109.0	-----	[------]
DEC PDP8L	4k CINET	4.8	29.0	45.8	55.2	60.0	73.0	133.6	12.0	[145.6]
HP 9830A/B	Resident	4.4	14.6	35.6	38.1	40.5	74.3	128.4	19.4	[147.8]*
WD MicroEngine	BASIC1.1	18.2	10.7	22.5	22.5	24.3	133.7	187.6	2.8	[190.4]*
IBM 5100	Resident	4.5	21.1	57.4	54.5	59.0	88.2	174.9	26.7	[201.6]
DEC PDP8E	Eds20 na	8.5	18.0	52.3	52.4	65.4	65.4	203.7	12.4	[216.1]
DEC PDP8L	12kEds20	12.5	39.2	74.6	73.8	91.0	91.0	288.5	16.5	[305.0]
DEC PDP8E	1usEds20	12.5	47.3	86.0	86.1	96.4	96.4	333.6	18.3	[351.9]
SWPTC MP68	8k V2.0	15.6	25.4	96.9	105.9	109.8	174.5	205.1	172.0	[377.1]
Mycron 8080	Tiny B	12.1	20.9	57.6	58.4	101.0	204.0	306.0	80.0	[386.0]
Elliot 803	8k V1	9.0	12.0	29.0	32.0	48.0	41.0	372.0	38.0	[410.0]

* Repeated or run specifically for this paper

[Oxford Systematics June 1982]

BASIC BENCHMARK LISTINGS BM1 to 9 : LISTINGS OF BM1-8 FROM COMPUTING EUROPE JULY 27 1978

```
100 REM     BM1
300 PRINT "BM1"
400 FOR K=1 TO 1000
500 NEXT K

700 PRINT "E"
800 END
```

```
100 REM     BM2
300 PRINT "BM2"
400 K=0
500 K=K+1

600 IF K<1000 THEN 500
700 PRINT "E"
800 END
```

```
100 REM     BM3
300 PRINT "BM3"
400 K=0
500 K=K+1
510 A=K/K*K+K-K
600 IF K<1000 THEN 500
700 PRINT "E"
800 END
```

```
100 REM     BM4
300 PRINT "BM4"
400 K=0
500 K=K+1
510 A=K/2*3+4-5


600 IF K<1000 THEN 500
700 PRINT "E"
800 END
```

```
100 REM     BM5
300 PRINT "BM5"
400 K=0
500 K=K+1
510 K=K/2*3+4-5
520 GOSUB 820


600 IF K<1000 THEN 500
700 PRINT "E"
800 STOP
820 RETURN
```

```
100 REM     BM6
300 PRINT "BM6"
400 K=0
500 K=K+1
510 A=K/2*3+4-5
520 GOSUB 820
530 FOR L=1 TO 5
540 NEXT L
600 IF K<1000 THEN 500
700 PRINT "E"
800 STOP
820 RETURN
```

```
100 REM     BM7
300 PRINT "BM7"
400 K=0
430 DIM M(5)
500 K=K+I
510 A=K/2*3+4-5
520 GOSUB 820
530 FOR L=1 TO 5
535 M(L)=A
540 NEXT L


600 IF K <1000 THEN 500
700 PRINT "E"
800 STOP
820 RETURN
```

```
100 REM     BM8
300 PRINT "BM8"
400 K=0

500 K=K+1


550 A=K^2
560 B=LOG(K)
570 C=SIN(K)
580 IF K<100 THEN 500

700 PRINT "E"
800 END
```

```
100 REM     BM9
130 PRINT "BM9"
140 FOR N= 1 TO 1000
150 FOR K= 2 TO 500
160 LET M=N/K
170 LET L=INT(M)
180 IF  L=0 THEN 230
190 IF  L=1 THEN 220
200 IF  M>L THEN 220
210 IF  M=L THEN 240
220 NEXT K
230 PRINT N;
240 NEXT L
250 PRINT "E"
260 END
```

[Oxford Systematics June 1982]

BENCHMARKS BM1-9 IN FORTRAN

```
      PROGRAM BM1
      WRITE(3,1)
    1 FORMAT(4H BM1)
      DO 2 K=1,1000
    2 CONTINUE
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM2
      WRITE(3,1)
    1 FORMAT(4H BM2)
      K=0
    2 K=K+1
      IF(K.LT.1000)GOTO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM3
      WRITE(3,1)
    1 FORMAT(4H BM3)
      K=0
    2 K=K+1
      A=K/K*K+K-K
      IF(K.LT.1000)GOTO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM4
      WRITE(3,1)
    1 FORMAT(4H BM4)
      K=0
    2 K=K+1
      A=K/2*3+4-5
      IF(K.LT.1000) GO TO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM5
      WRITE(3,1)
    1 FORMAT(4H BM5)
      K=0
    2 K=K+1
      A=K/2*3+4-5
      CALL GOSUB
      IF(K.LT.1000)GOTO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM6
      DIMENSION M(5)
      WRITE(3,1)
    1 FORMAT(4H BM6)
      K=0
    2 K=K+1
      CALL GOSUB
      DO 4 L=1,5
    4 CONTINUE
      IF(K.LT.1000)GOTO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM7
      DIMENSION M(5)
      WRITE(3,1)
    1 FORMAT(4H BM7)
      K=0
      K=K+1
      A=K/2*3+4-5
      CALL GOSUB
      DO 3 L=1,5
      M(L)=A
    4 CONTINUE
      IF(K.LT.1000)GOTO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      PROGRAM BM8
      WRITE(3,1)
    1 FORMAT(4H BM8)
      K=0
    2 K=K+1
      A=K**2
      FK=K
      B=ALOG(FK)
      C=SIN(FK)
      IF(K.LT.101)GOTO 2
      WRITE(3,3)
    3 FORMAT(2H E)
      STOP
      END
```

```
      SUBROUTINE GOSUB
      RETURN
      END
```

```
      PROGRAM BM9
      WRITE(3,1)
    1 FORMAT(4H BM9)
      DO 2 N=1,1000
      DO 3 K=2,500
      FN=N
      FM=FN/K    PP
      L=INT(FM)
      IF(L .EQ.0)GOTO 4
      IF(L .EQ.1)GOTO 3
      IF(FM.GT.L)GOTO 3
      IF(FM.EQ.L)GOTO 2
    3 CONTINUE
    4 WRITE(3,6) N
    6 FORMAT(1H ,I3)
    2 CONTINUE
      WRITE(3,7)
    7 FORMAT(2H E)
      STOP
      END
```

[Oxford Systematics June 1982]

BENCHMARKS BM1-9 IN PASCAL

```
program bm1;
VAR k:integer;
BEGIN
write('bm1');
for k:=1 to 1000 do;
write('e');
END.
```

```
program bm2;
VAR k:integer;
BEGIN
write('bm2');
k:=0;
REPEAT
k:=k+1;
UNTIL k=1000;
write('e');
END.
```

```
program bm3;
VAR k,a:integer;
BEGIN
write('bm3');
k:=0;
REPEAT
k:=k+1;
a:=k div k*k+k-k
UNTIL k=1000;
write('e');
END.
```

```
program bm4;
VAR k,a:integer;
BEGIN
write('bm4');
k:=0;
REPEAT
k:=k+1;
a:=k div 2*3+4-5;
UNTIL k=1000;
write('e');
END.
```

```
program bm5;
VAR k,a:integer;

procedure gosub;
BEGIN
END;
BEGIN
write('bm5');
k:=0;
REPEAT
k:=k+1;
a:=k div 2*3+4-5;
gosub;
UNTIL k=1000;
write('e');
END.
```

```
program bm6;
VAR l,k,a:integer;
m:array[1..5] of integer;
procedure gosub;
BEGIN
END;
BEGIN
write('bm6');
k:=0;
REPEAT
k:=k+1;
a:=k div 2*3+4-5;
gosub;
for l:=1 to 5 do
UNTIL k:=1000;
write('e');
END.
```

```
program bm7;
VAR l,k,a:integer;
m:array[1..5] of integer;
procedure gosub;
BEGIN
END;
BEGIN
write('bm7');
k:=0;
REPEAT
k:=k+1;
a:=k div 2*3+4-5;
gosub;
for l:=1 to 5 do m[l]:=a;
UNTIL k=1000;
write('e');
END.
```

```
program bm8;
VAR k:integer;
a,b,c:real;
BEGIN
write('bm8');
k:=0;
REPEAT
k:=k+1;
a:=sqr(k);
b:=ln(k);
c:=sin(k);
UNTIL k=100;
write('e');
END.
```

```
program bm9;
LABEL 1,2,3;
VAR k,l,m,n:integer;
BEGIN
writeln('bm9');
for n:=1 to 1000 do
BEGIN
for k:=2 to 500 do
BEGIN
m:=n mod k;
l:=n div k;
if l=0 then goto 2;
if l=1 then goto 1;
if m>0 then goto 1;
if m=0 then goto 3;
1:END;
2:writeln(n);
3:END;
write('e');
END.
```

NOTE: The branch out of a nested "FOR" loop is not legal in Pascal/Z or in MT+

[Oxford Systematics June 1982]

BENCHMARKS IN "S-ALGOL" : STRUCTURED [UNIVERSITY OF ST ANDREWS] ALGOL

```
write"bm1"
for k = 1 to 1000 do {}
write "e"
```

```
write"bm2"
let k:=0
    k:=k+1
while k<1000 do k:=k+1
write"e"
```

```
write"bm3"
let k:=0
while k<1000 do
BEGIN
k:=k+1
let a:=k/k*k+k-k
END
write"e"
```

```
write "bm4"
let k:=0
while k < 1000 do
BEGIN
k:=k+1
let a:=k/2*3+4-5
END
write "e"
```

```
procedure gosub
{}
write"bm5"
let k:=0
while k<1000 do
BEGIN
k:=k+1
let a:=k/2*3+4-5
gosub
END
write"e"
```

```
procedure gosub
{}
let m=vector 1::5 of 0
let k:=0
while k<1000 do
BEGIN
k:=k+1
let a:=k/2*3+4-5
gosub
for l=1 to 5 do {}
END
write"e"
```

```
write"bm7"
procedure gosub
{}
let m := vector 1::5 of 0.0
let k:=0
while k <1000 do
BEGIN
k:=k+1
let a:=k/2*3+4-5
gosub
for l=1 to 5 do {m(l):=a}
END
write "e"
```

```
let k:=0
write"bm8"
while k<100 do
BEGIN
k:=k+1
let a:=k*k
let b:=ln(k);
let c:=sin(k);
END
write"e"
```

```
write"bm9"
let l:=0
let m:=0.0
let n:=0
while n<1000 do
BEGIN
n:=n+1
let k:=1
while k<501 do
BEGIN
k:=k+1
m:=n/k
l:=truncate(m)
CASE true of
l=0:k=500
l=1:{}
m>1:{}
m=1:k=500
default:{}
END
if m =1 do write" 'n",n
END
write"e"
```

NOTE: Pascal-6000 on the CDC Cyber range has no LOG function. LN was used. The same applies to S-Algol

[Oxford Systematics June 1982]

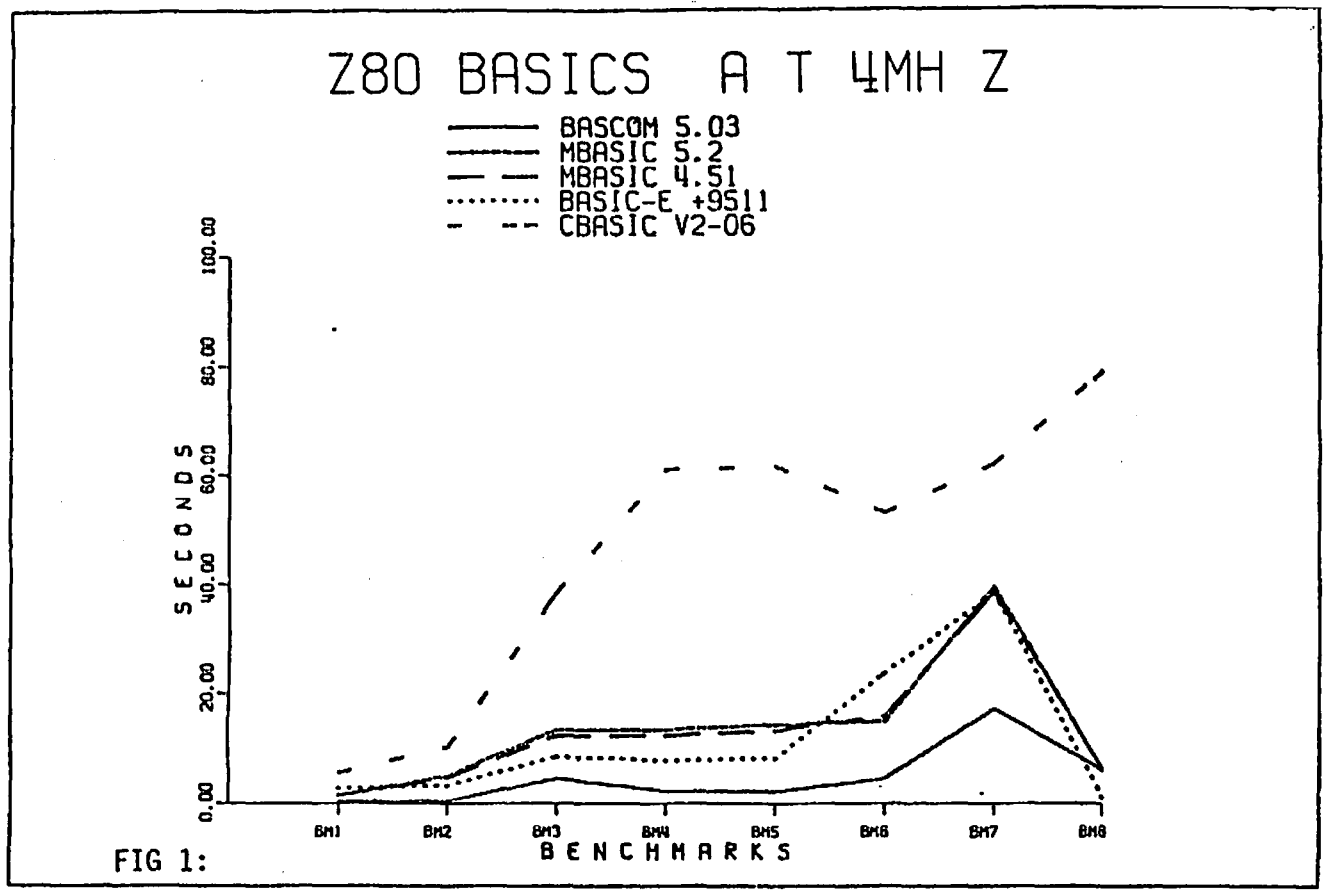

FIG 1:

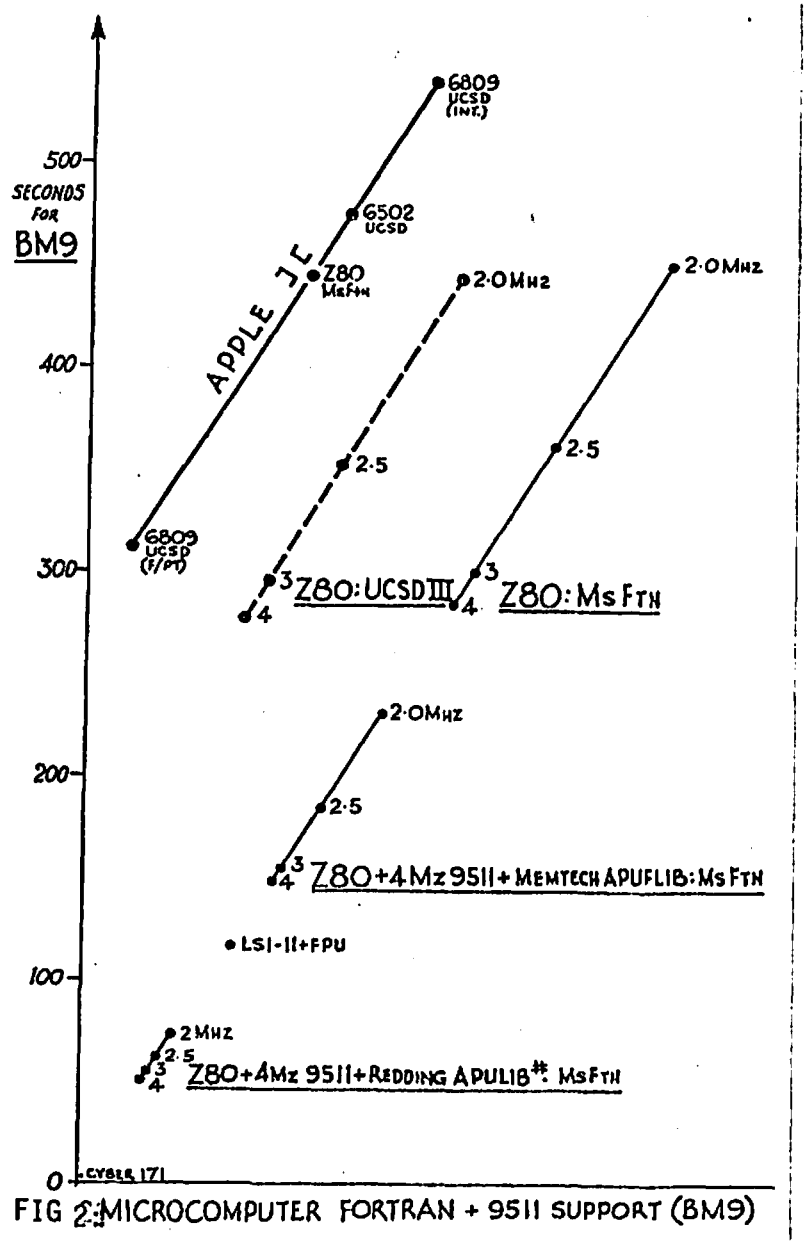

FIG 2: MICROCOMPUTER FORTRAN + 9511 SUPPORT (BM9)

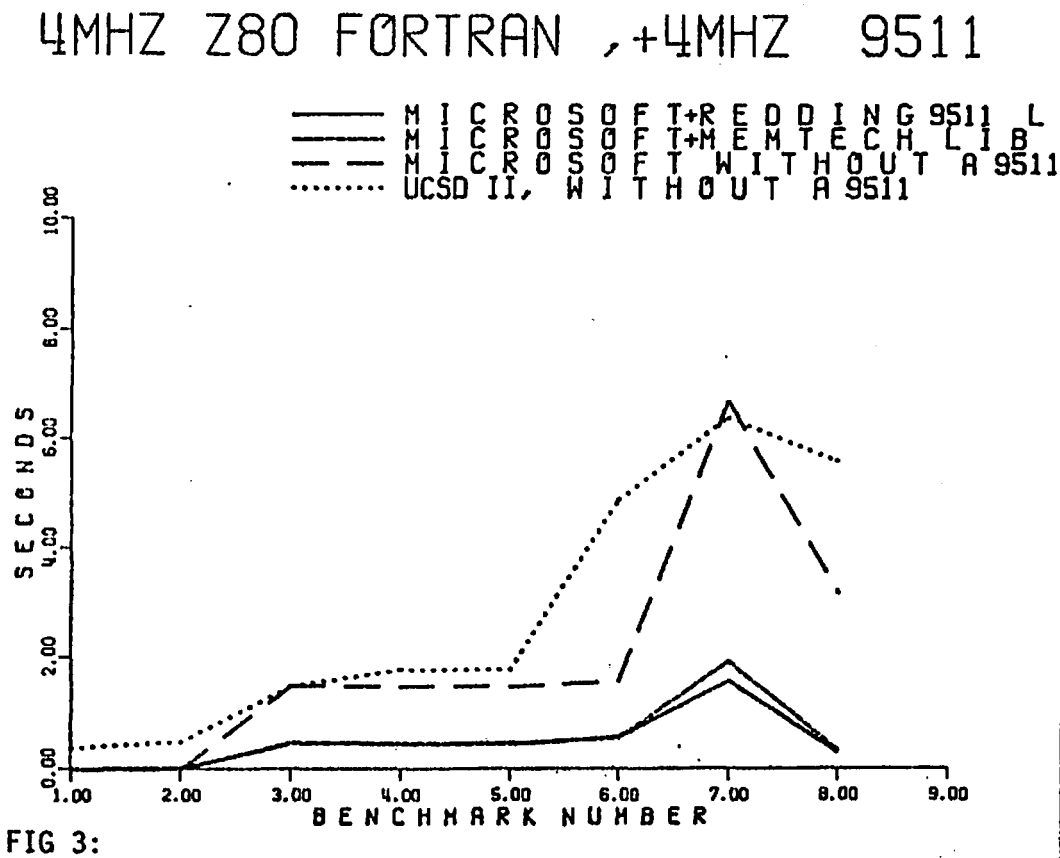
4MHZ Z80 FØRTRAN ,+4MHZ 9511
MICROSOFT+READING 9511 L
MICROSOFT+MEMTECH LIB
MICROSOFT WITHOUT A 9511
UCSD II, WITHOUT A 9511
SECONDS
0.00
2.00
4.00
6.00
8.00
10.00
1.00
2.00
3.00
4.00
5.00
6.00
7.00
8.00
9.00
BENCHMARK NUMBER

FIG 3:

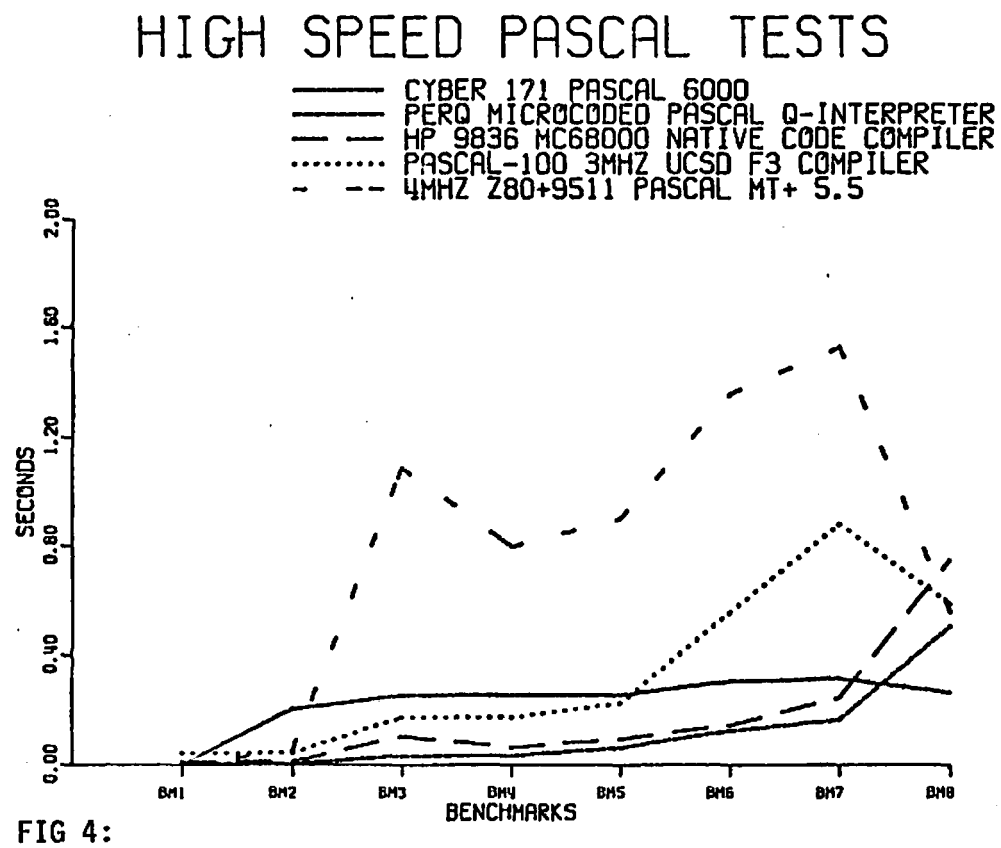
HIGH SPEED PASCAL TESTS
CYBER 171 PASCAL 6000
PERQ MICROCODED PASCAL Q-INTERPRETER
HP 9836 MC68000 NATIVE CODE COMPILER
PASCAL-100 3MHZ UCSD F3 COMPILER
4MHZ Z80+9511 PASCAL MT+ 5.5
SECONDS
0.00
0.40
0.80
1.20
1.60
2.00
BM1
BM2
BM3
BM4
BM5
BM6
BM7
BM8
BENCHMARKS

FIG 4:

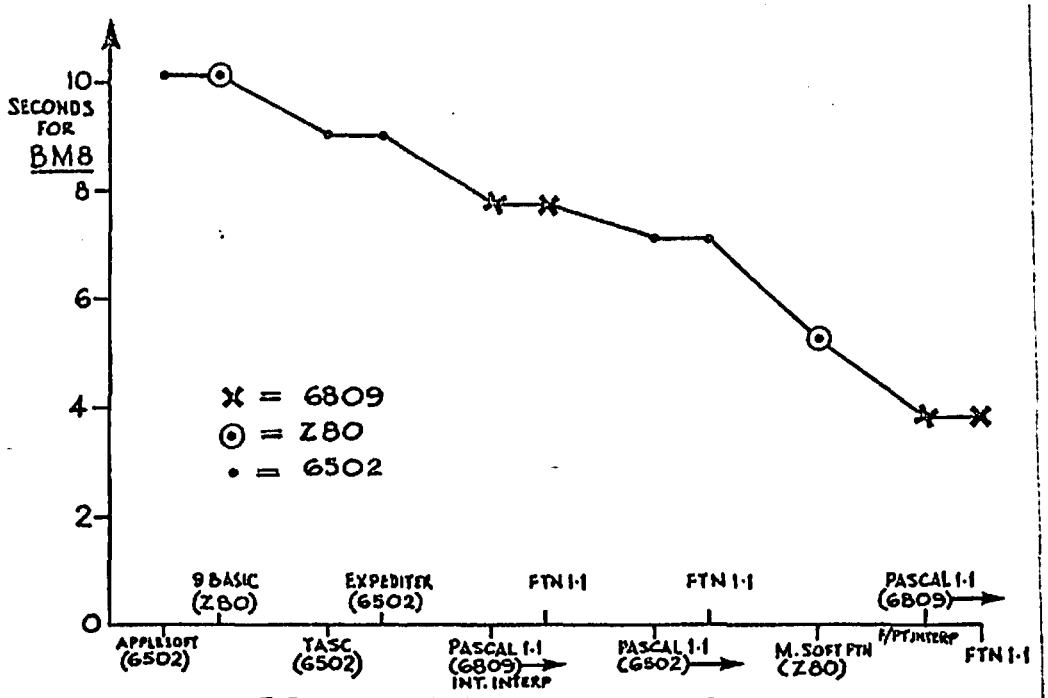

FIG 7: APPLE][FLOATING POINT FUNCTIONS

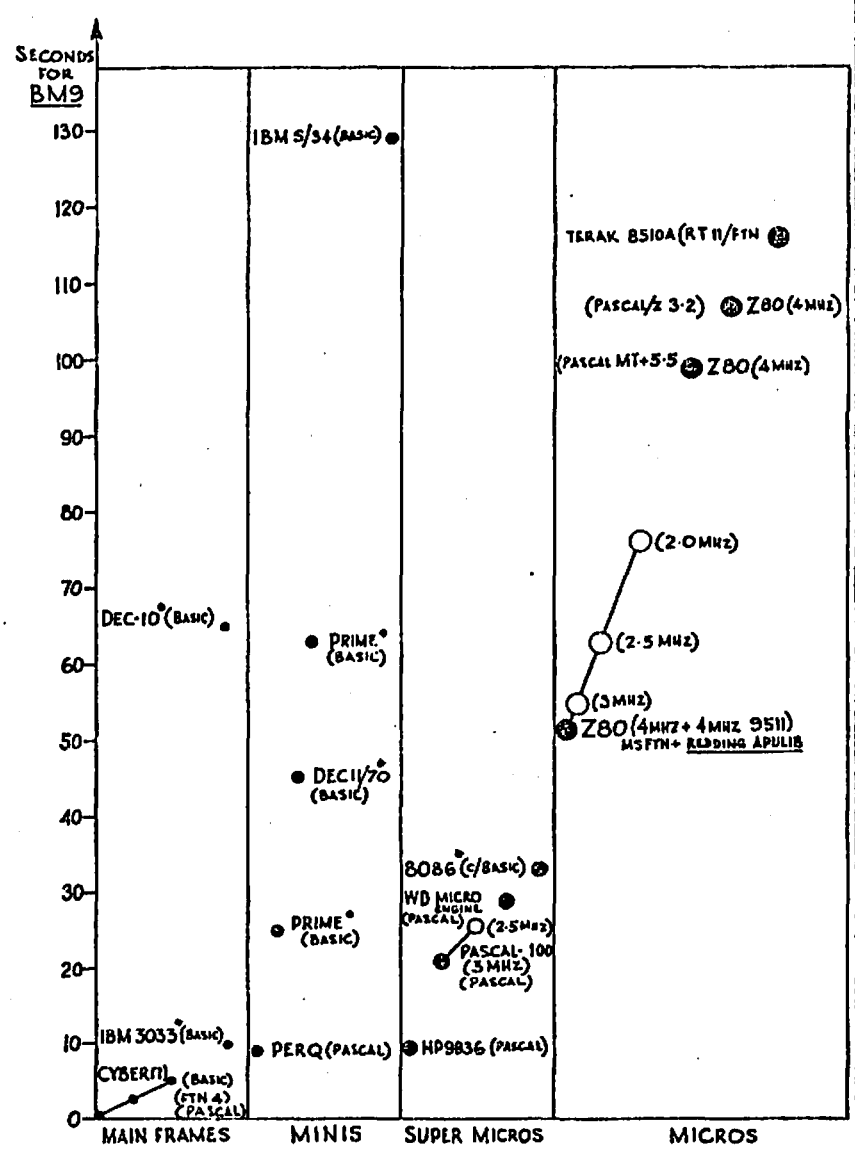

FIG 8: RAW INTEGER SPEED: THE FASTEST BM9 RESULTS.

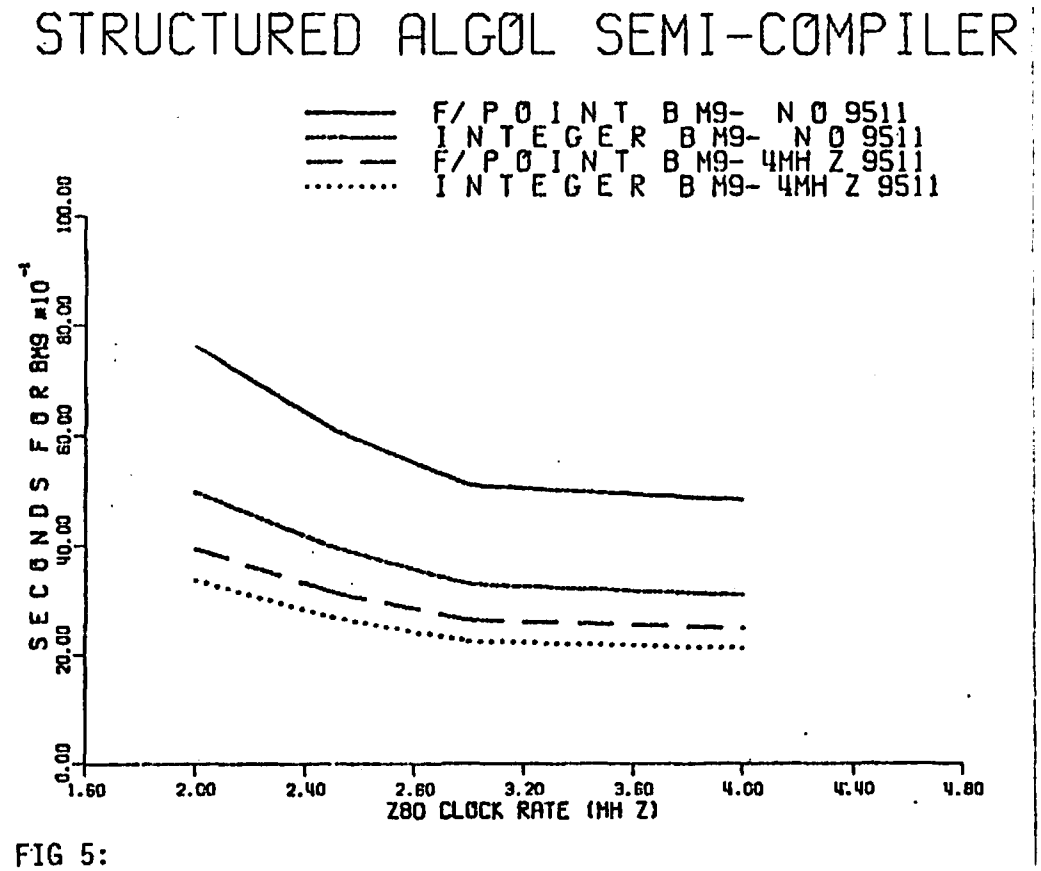

FIG 5:

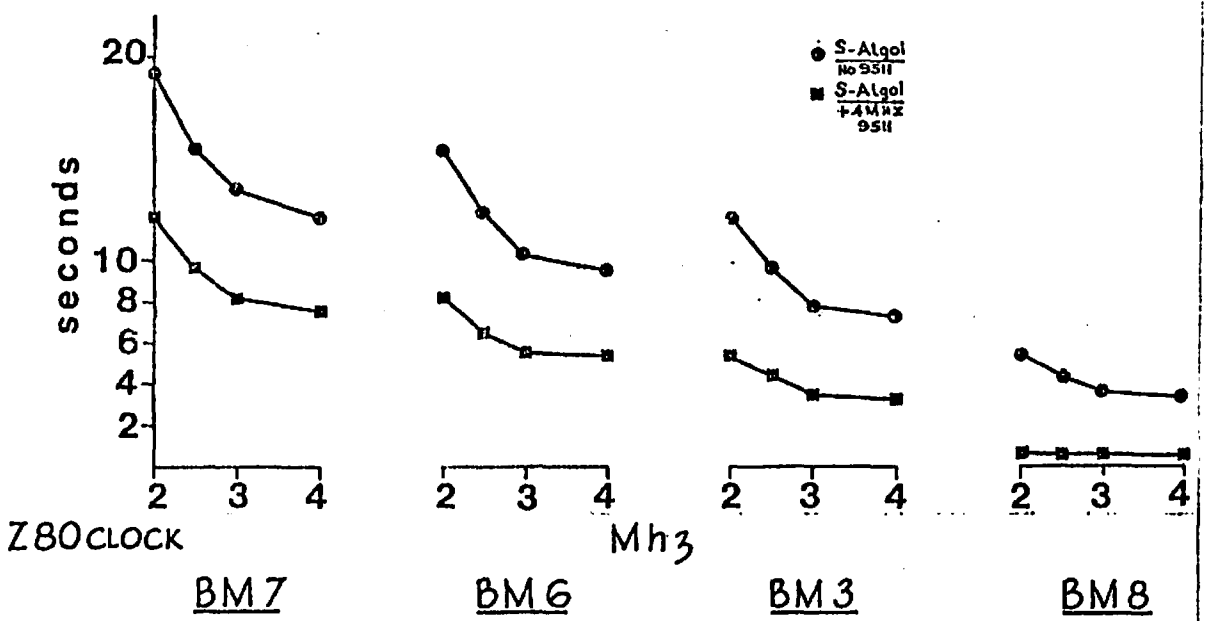

FIG 6: S-ALGOL WITH VARIATIONS OF Z80 CLOCK RATES AND INTERPRETERS