$(-1, +1)$ and the computer has a modulus-forming order then we might compute u, v as

$$u = |U|,$$
$$v = |V|,$$

and use the sign of either U or V with the accepted value of x.

A slightly more convenient method can be used on SILLIAC, a 40-bit parallel machine which represents negative numbers in complementary form, and in which the accumulator register, A, and the quotient register, Q, may be shifted as a single register of double length, AQ.

After the test for acceptance, A contains $(x - 1)^2 - 2y$ in the first method and $z - y$ in the second method, scaled in each case by a power of 2. In either case the least significant digit of A is random except perhaps for a small bias of order 2^{-40} (such a bias is possible because the least significant digit of A may be slightly correlated with the magnitude of the numbers x and y). x is copied into Q and AQ shifted one place to the right so that the least significant digit of A becomes the sign of Q. $|Q|$ is now copied into A and a single left shift is executed giving $\pm x$ (with the least significant digit corrupted) in A.

On SILLIAC this produces a random sign for x in four instructions and 0·20 msec, whereas the straightforward method of making an extra entry into the random routine requires six instructions and 0·26 msec + R, R being about 1·0 msec for a typical SILLIAC random-number routine.

If a similar method to this is used to form a random sign, but the least significant digit of u or v is used instead of that of $(x - 1)^2 - 2y$ or $z - y$, it should first be verified that the random-number sequence being used has had a frequency test successfully performed on the least significant digit, and that this digit is not correlated with the most significant digits. Particularly to be avoided for this purpose are number sequences such as the reduced Fibonacci sequence whose least significant digit recurs with period three.

References

BUTLER (1956). *Symposium on Monte Carlo Methods*, p. 249, WILEY.
KAHN, H. (1954). Rand Memorandum 1237-AEC.

Computers as an Aid in Computer Design Assessment

By J. M. Bennett and R. J. Dakin

Simulation routines provide a useful tool for assessing proposed computer designs. An example involving the simulation of a small magnetic-drum computer with various possible modifications is described.

Introduction

For some years, interpretive techniques have been used to enable one computer to carry out work programmed in the code of another. As early as 1951, one of the authors made use of this method to enable the Ferranti Mark I machine at Manchester University to test programs for the Mark I*, a machine with a different order code but a similar engineering design. (Bennett *et al.*, 1952.)

A similar application arose recently at Sydney University in connection with the SNOCOM computer which, at the time of writing, is undergoing its final tests in the Electrical Engineering Department. The eventual user (the Snowy Mountains Hydroelectric Authority) wanted a range of programs to be available when the machine was delivered, and so a simulation routine was written so that code checking could be done on SILLIAC, the Basser Computing Department's computer (Bennett and Swire, 1957).

The routine was written in such a way that, at any stage of the execution of a program, it could be made to print out details of timing—time spent waiting for data and instructions, total time, and so on. The ready availability of this information made a further step possible—that of investigating the effect of modifying the order code in various ways, and of changing the overall organization of the machine (for example, by adding immediate-access storage).

SNOCOM

SNOCOM is a serial computer with a 2,048-word magnetic-drum store, consisting of 64 tracks each of 32 words. The drum rotates at 6,000 r.p.m., and consecutive addresses are spaced nine words apart to provide a measure of optimization.

Each instruction occupies a single word, and the code is a simple, single-address type. There is a single 32-bit accumulator consisting of a revolver, the last bit being dropped on transfer from the accumulator. Input is by means of punched paper tape and output is by means of punched paper tape or electric typewriter.

Under ideal conditions, most instructions can be completed before the next instruction in the sequence appears at the reading head—i.e. in nine word times, which occupy about 2·8 milliseconds. Multiplication

Table 1

COMPUTER ORGANIZATION	ROUTINE					
	SQUARE ROOT			MATRIX MULTIPLICATION (TWO 7 BY 7 MATRICES)		
	TOTAL TIME (MSEC.)	WAITING TIME (MSEC.)	PERCENTAGE WAITING TIME	TOTAL TIME (SEC.)	WAITING TIME (SEC.)	PERCENTAGE WAITING TIME
Normal SNOCOM	419	166	40	101·2	79·3	76
SNOCOM with 11 words of immediate-access store (used for data)	309	56	18	80·3	58·3	73
SNOCOM with 3 indexing registers, 8 words of immediate-access store (used for data)	As above			46·9	28·9	62
SNOCOM with 3 indexing registers, 32-word block transfer instructions, 64 words of immediate-access store (used for data and instructions)	This was not tried. Figures shown below should apply.			25·5	5·7	23
As above, but without indexing registers	266	2	1	26·9	4·9	18

and division instructions under ideal conditions take an extra two revolutions (i.e. 22·8 milliseconds in all). Input and output instructions take the same time as normal instructions unless limited by the speed of the reader or punch/printer.

SILLIAC and the Simulation Routine

SILLIAC has 1,024 40-bit words of fast store with two instructions to a word. By suitable packing arrangements, a simulated store of 1,024 words was provided; 32 bits were used for each word, the last bit (which is not in the real SNOCOM store) being used to indicate that diagnostic printout is required. For the modified versions of the routine used for analysing the effect of changes, the simulated store was cut down to 512 words to allow sufficient room in the SILLIAC memory for incorporating the effects of modifications to the design of the simulated machine.

The simulation is almost complete—except for a few minor details associated with the initial input. In addition, printout in sexadecimal form of information such as the contents of the accumulator and/or the current order itself, or details of the timing, can be obtained when an order is marked at the thirty-second bit.

It might be worth mentioning that two SNOCOM programs, with no input or output, took about three times as long to run on SILLIAC with the simulation routine as when carried out on SNOCOM itself. Of course, such a time penalty is of minor concern during debugging, and any resulting disadvantage is more than outweighed by the additional diagnostic facilities available.

Computer Design Assessment

It is common practice for industrial design study groups to code up a series of representative programs for a number of possible machine designs, and to compute the associated timings manually. The designs are chosen to satisfy certain obvious criteria. Thus a machine intended to be built in as short a time as possible would be built from units the design details of which are well established. Another example would be the provision of some hardware to help mixed-radix operation (manipulation of £ s. d. in particular) in a machine expected to do commercial work. The problems chosen would be "typical" for the expected end use of the machine, and would be chosen to emphasize both good and bad features in the design.

Unfortunately, coding is often not done as realistically as is desirable. Questions of scaling in matrix work, for example, are often glossed over, resulting in serious distortion of the estimates.

It might be argued that individual coders will produce widely different programs and that timing figures obtained will accordingly differ widely. However, one of the authors (J. M. B.), in previous contacts with manual assessment studies carried out in an industrial group, found a high degree of consistency between results obtained by different programmers of reasonable experience. Moreover, these results proved to be comparatively insensitive to the order lists proposed,

provided these lists corresponded to approximately the same overall organization of the computer being studied. Hence, although the results obtained could not be regarded as relevant in showing up differences in efficiency of a few per cent, an estimate of gross differences in performance should be obtainable in this way. Since, in this type of study, only large differences are significant, this approach is quite adequate.

Results

Once the simulation routine has been constructed, changing it so that it caters for different machine organizations is not a major task. Moreover, diagnostic facilities built into the simulation routine expedite the debugging of routines written for the organizations tested.

Square root, exponential and matrix-multiplication routines were used as a basis of comparison in this study. The matrix-multiplication routine catered for scaling, and counting was carried out by means of markers; i.e. easily recognizable numbers, not likely to occur in the course of the calculation, were used to mark ends of rows and of the matrices being multiplied.

Table 1 is a sample of the results obtained, and shows the time taken for the square root routine, and the time taken to multiply two 7 by 7 matrices, with different versions of SNOCOM.

Of particular note is the large improvement obtained when indexing registers are used in conjunction with the drum store and a small immediate-access store, and the comparatively small improvement when indexing registers are used in conjunction with block transfer orders and a larger immediate-access store.

The apparent contradiction in the last two entries relating to waiting time in matrix multiplication (i.e. the apparent reduction in waiting time resulting from the removal of indexing registers) is due to the details of the instruction lists used, and is an indication that this assessment technique should not be relied on to evaluate small differences in the efficiencies of two designs.

Conclusions

Simulation routines are of value in any detailed assessment of a proposed computer design. The advantages of a detailed breakdown of time obtained automatically in this way are the reliability of the figures, the ease with which it is possible to examine various sizes of problems handled by the one program, and, for a given order code, the effect of changes in the times taken to obey the individual orders.

Acknowledgements

The writers are indebted to Mr. D. G. Wong of the Electrical Engineering Department, University of Sydney, for advice and suggestions concerning some of the modifications.

References

BENNETT, J. M., PRINZ, D. G., and WOODS, M. L. (1952). "Interpretive Sub-Routines," *Proc. A.C.M. Conference at Toronto*, 1952.

BENNETT, J. M., and SWIRE, B. E. (1957). "The SILLIAC," Paper 103, *Proc. Conference on Data Processing and Automatic Computing Machines* (Australian Department of Supply), 1957.

FREEDMAN, A. L. (1954). "Elimination of Waiting Time in Automatic Computers with Delay-type Stores," *Proc. Camb. Phil. Soc.*, Vol. 50, p. 426.