SHORT COMMUNICATIONS

A PROPOSED MODIFICATION TO THE C.S.I.R.O. MARK I COMPUTER*

By B. E. SWIRE†

The order code of the C.S.I.R.O. Mark I computer, as described by Pearcey and Hill in their recent papers (1953*a*, 1953*b*, 1954), is already more extensive and flexible than that of any other machine of which a description has been published. It still offers some scope for expansion, however, and it is the purpose of this note to put forward one particular suggestion for the addition of an entirely new type of order. " General purpose " computers, so-called, are at present by no means so general in their application as users would desire, and broadly speaking there are two types of attack on this problem. On the one hand one can incorporate in the order code more and more special functions so as to increase the number of types of work with which the computer can directly deal. On the other hand one can seek new types of order which tend to increase the flexibility of a machine without special reference to any particular application. The suggestion put forward here is in line with this latter approach, which is believed to be the more valuable of the two.

The importance of flexibility has been recognized in the logical design of C.S.I.R.O. Mark I computer, and noteworthy success has been achieved by the use of an arithmetical unit containing a multiplicity of registers having addition, subtraction, and discrimination facilities (Beard and Pearcey 1952), and by the adoption of an order code in which each order is expressed as a transfer from a " source " to a " destination " each of which is separately and independently specified by a five-digit code. Programmes expressed in this code are very considerably shorter than programmes expressed in a more conventional one-address code such as that of the EDSAC. Nevertheless it has been found necessary to use interpretive programming techniques when it is desired to work to multiple-word accuracy, in floating-point arithmetic, or with complex numbers.

The interpretive programmes, or " hyper-programmes ", as described by Pearcey and Hill, consist of a mixture of two types of command, namely, commands coded as ordinary machine commands and commands requiring interpretation ; but for each command, whether it is one requiring interpretation or not, the interpretation routine of some 20 or 30 commands must be traversed. It is, in fact, the interpretation routine which distinguishes between a command

* Manuscript received October 19, 1954.

† Aeronautical Research Laboratories, Department of Supply, Melbourne ; present address : Adolph Basser Computing Laboratory, University of Sydney.

coded as a machine command, and a command labelled by the use of the null destination Z to indicate that it requires interpretation.

With this interpretive system, as with similar systems used on other machines, a gain in flexibility and convenience is achieved at a considerable cost in speed. But suppose now that a facility be built into the machine for distinguishing between machine commands and interpretive or "hyper" commands, in such a way that the former are performed directly, while, when one of the latter is encountered, control is directed to the head of the interpretation routine with simultaneous storage of the link datum to enable return of control to the next programme command after performance of the hyper-function. If such a facility were available the interpretation routine would be considerably shortened and furthermore would be used only when strictly necessary. A considerable gain in speed over the interpretive system described by Pearcey and Hill would be achieved. In fact, this facility, if available, would become the normal method of calling in sub-routines, and the distinction between direct and interpretive programming would vanish.

Not enough is known of the detailed design of existing overseas machines to say whether the proposed facility could be readily added to any of them. The design of the C.S.I.R.O. Mark I machine is, however, such as to allow its ready inclusion. As mentioned above, hyper-commands as at present used on the C.S.I.R.O. machine are labelled by the use of the Z destination code. If this Z code appears in a normal machine command, a null command results since no gates are opened. Suppose now that a small addition were made to the machine so that this Z destination did control certain gates so as to perform the following functions:

(*a*) read out the content of the sequence register in digit positions 11–20, leaving the sequence register clear;

(*b*) read out the content of the lower half of the interpreter register in digit positions 1–10, leaving the address digits in the upper half of the interpreter undisturbed (as is the case with the $K+$ destination);

(*c*) open the read-in gate to register B so that the full word assembled in (*a*) and (*b*) is placed in register B.

Since the sequence register has been cleared, the next command executed would be that in location zero, which would be the head location of the interpretation routine. The first command in this routine would be $0 \rightarrow H_u$, whereupon the address digits previously left in the interpreter would be placed in the H register. All the data required are then available: the function code in digit positions 6–10 of B; the address in H; and the link datum in digit positions 11–20 of B. It would be a simple matter to design an interpretation routine to direct control via a directory to the appropriate sub-routine. The interpretation routine should preferably not use register B; then each sub-routine could commence with the command $(B) \rightarrow x$, where x is a store location allocated within each sub-routine to hold the link datum. Each sub-routine would then conclude with the command $(x) \rightarrow S$ to return control to the next programme command (the single command $(B) \rightarrow S$ at the end would suffice for linking if

the sub-routine did not require to make use of register B). The present practice of using some of the D registers for holding link data would then be avoided, and these registers freed for other purposes.

The facility just proposed is not available on any existing computer, nor, so far as the author is aware, has it been previously suggested. There have been previous suggestions for calling in sub-routines by means of a single order. The Circle computer (Greig 1953), for example, includes what is known as the " Function Table " order ; this does enable a sub-routine to be called in by a single command, but this command does not, and cannot, include an address or other parameter for use by the sub-routine called in. The function table order therefore does not have the character of a true hyper-command such as is possessed by the new type of order that has been here described.

The author acknowledges indebtedness to the Aeronautical Research Laboratories, Department of Supply, and the Radiophysics Laboratory, C.S.I.R.O., for jointly making it possible for him to become familiar with the C.S.I.R.O. Mark I computer and to carry out certain computing work with it. Particular appreciation is expressed of the unstinted help given by Mr. Pearcey and Mr. Beard and the members of their team at the Radiophysics Laboratory.

References

Beard, M., and Pearcey, T. (1952).—*J. Sci. Instrum.* **29** : 305.
Greig, J. (1953).—*Math. Tab., Wash.* **7** : 249.
Pearcey, T., and Hill, G. W. (1953*a*).—*Aust. J. Phys.* **6** : 316.
Pearcey, T., and Hill, G. W. (1953*b*).—*Aust. J. Phys.* **6** : 335.
Pearcey, T., and Hill, G. W. (1954).—*Aust. J. Phys.* **7** : 485.