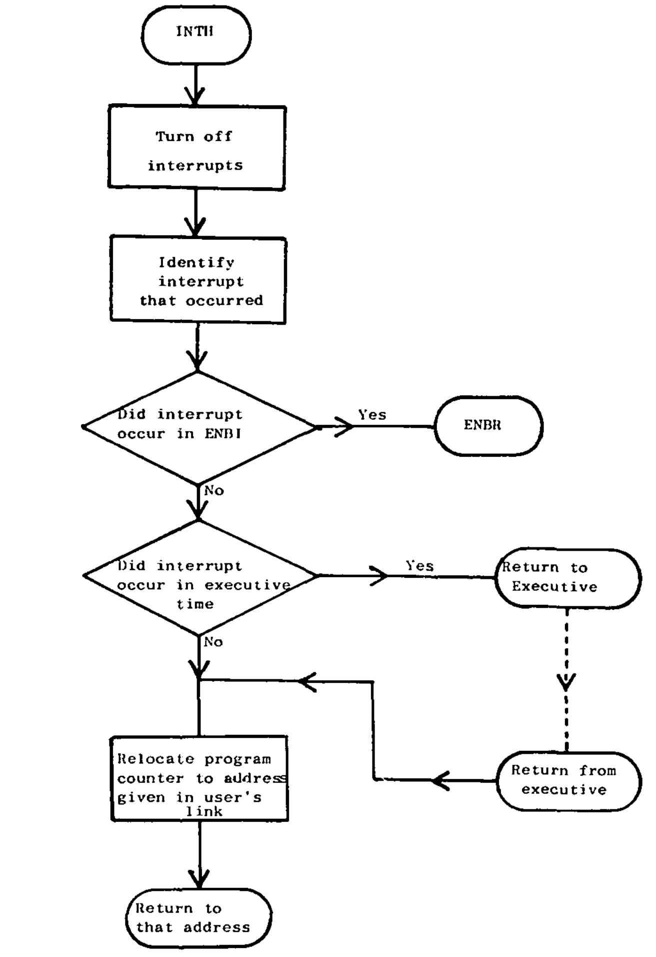
INTH
Turn off interrupts
Identify interrupt that occurred
Did interrupt occur in ENBI
Yes
ENBR
No
Did interrupt occur in executive time
Yes
Return to Executive
No
Relocate program counter to address given in user's link
Return from executive
Return to that address

Fig. 2. INTH routine

5. *RTC*

All interrupts requested by the user, save one, go through the routine INTH. The one exception is the 10 millisecond clock. The reason for this being that it is required by the executive and therefore has to be handled in a different manner. When an interrupt occurs on this line it is forced to the routine RTC. Initially a check is made on whether the user has exeeded his time allocation, a time slice of 40 milliseconds. If he has, a check is made for requests for service from the other users. If requests are present an exit is made to the user next due for service, who has the next time slice allocated to him. However, if service has not been requested by other users, a further time slice is allotted to the current user. A check is made for him requiring the 10 millisecond clock to interrupt.

If he does and the enable toggle is set, the toggle is cleared and a return is made to the address given in the user's 10 millisecond interrupt link. If he does not require this interrupt a normal return is made.

Differences from normal running

The interrupt handling facility has obvious disadvantages for the running of real-time programs. Two obvious ones are:

1. Time profile distortion. This is due to the encroachment of the executive and the sharing of available time between users. This in turn leads to 2.
2. Slow reaction rate. While this is a disadvantage, in the applications handled in this system it proved to be a minor one.

Despite these disadvantages, the system has proved an invaluable tool in increasing program throughput as powerful debug facilities which would normally be available only for linear programs may be applied to real-time ones.

It should be possible to implement this type of system on any small computer with at least 16K words, a word being at least 16 bits long, possessing some form of memory protection hardware and the facility for running user's programs in a different mode to the executive, privileged instructions being able to be run only by the executive.

Acknowledgement

The authors wish to thank Mr. S. Chisman for his help and the Post Office Corporation and Scientific Control Systems for permission to publish this paper.

Reference

Lima System for Honeywell DDP 516 GPO Research Station 1970. (Available from D. G. Bennett, c/o GPO Research Station, Dollis Hill.

Correspondence

To the Editor
The Computer Journal

Sir,

The claim is often made that the argument transmission method used by certain FORTRAN processors (notably those for the IBM 360) in which arguments are passed by value and the possibly modified value copied back on return is non-standard. See for example the comment in J. M. Chambers (1971), page 314.

This is not the case. The USA Standard FORTRAN Report (1964) explicitly allows this mechanism in addition to the more usual call by address (sometimes called call by reference). The relevant quotations from the standard are as follows:

> 'If an entity created by argument substitution becomes defined or undefined (while association exists) during execution of a subprogram, then the corresponding actual entities in all calling program units becomes (sic) defined or undefined accordingly. (ANSI standard section 10.2.2.)
>
> 'If a function reference causes a dummy argument in the referenced function to become associated with another dummy argument in the same function or with an entity in common, a definition of either within the function is prohibited.' (ANSI standard section 8.3.2.)

With respect to the first requirement, the only difference between the two methods rests on whether the corresponding entity definition or undefinition occurs immediately or on subroutine exist. This could only be distinguished by means of a definition of the type explicitly prohibited in the second requirement quoted.

Thus both mechanisms must be regarded as meeting the standard and any program written in accordance with the standard will be insensitive to which of the two mechanisms employed.

Yours faithfully,
R. B. K. Dewar

Department of Computer Science
Illinois Institute of Technology
Chicago
Illinois 60616
USA
30 September 1971

References

Chambers, J. M. (1971). Another round of FORTRAN, *The Computer Journal*, Vol. 14, No. 3, pp. 312-314.

USA Standard FORTRAN (1964). *CACM*, Vol. 7, pp. 591-625.