Published: 07/18/67

Identification

Establish a handler for a condition
condition
M. A. Padlipsky

Purpose

During the execution of a process in Multics, certain conditions may be encountered which necessitate action outside of the normal flow of control. That is, they must be dealt with whenever they happen to arise. In PL/I jargon, these conditions may be "on-conditions" or "interrupts"; in general Multics discourse, they are simply to be called "conditions" (usually prefaced with "system-defined" or "programmer-defined"), and are to be viewed as software analogues of hardware faults. Because individual programmers may define their own "conditions" and may signal the fact that these conditions have arisen, examples of conditions are difficult to give; they may range from the occurrence of arithmetic overflow or a divide check (both system-defined) to something like "X_007" (clearly programmer-defined). Conditions of interest may, of course, arise at any time during the execution of a process. (This is particularly true of error conditions, which as a matter of Multics policy are to be treated by means of condition signals and condition handlers; see BY.11.) Also, specification of handlers for conditions may be changed at almost any time during the execution of a process.

The present section describes condition, which is the procedure used in order to 1) establish the existence of a condition and 2) push down the list of handlers for a condition, where a "handler" is a procedure which will be invoked (by the signal subroutine; see BY.12.03) when the occurrence of the condition is signalled. Section BD.9.04 presents a detailed description of the Multics condition-handling mechanism. Section BD.9.00 contains definitions of terms. (Removal of handlers from a condition's push-down list is accomplished by subroutine reversion, BY.12.05.)

Restriction

The condition name "cleanup" is reserved for applications dealing with abnormal returns; it should only be used when the Unwinder (BD.9.05) is expected to be invoked; it may not be signalled.

Usage

The calling sequence is

```
call condition (condname, proc);
```

with declarations

```
dcl condname char (*), proc entry;
```

where

condname is the name of the condition being established or, if already established, having its handler list pushed down; it may be declared in the user's procedure as varying or non-varying and the "*" should, of course, be replaced by an appropriate number.

proc is the procedure ("handler") which is to be invoked when this condition is signalled until and unless it is pushed down (by a subsequent call to condition) or popped off the list of handlers (by a call to reversion, BY.12.05).

Implementation

Details of the implementation of condition are to be found in BD.9.04. Briefly, a push-down list of handlers is maintained for each condition known to the process in behalf of which the condition-handling mechanism is operating. System-defined conditions have pre-defined default handlers; programmer-defined conditions have a single default handler (a call to signal for condition "unclaimed_signal") which is placed on the push-down list for the condition when the condition is first established. The condition routine pushes down the list of handlers for the condition for which it is invoked. Condition operates in the protection ring from which it is invoked.