Published: 11/21/66

Identification

Miscellaneous PL/I Statements
delay, display, reply, exit, stop
D. B. Wagner

Purpose

The PL/I statements described here are included in the Multics PL/I primarily for compatibility with other PL/I systems. It is clear that they are oriented to the batched user and that in Multics we can provide only approximations for their intended uses. All are described in the alphabetically-arranged Chapter 8 of the PL/I manual.

Display and Reply

These statements are intended to facilitate communication with "the machine operator". Since there may easily be many machine operators on a Multics system, each with his own separate console, and since direct communication between Multics users and machine operators is a thing to be highly discouraged anyway, display and reply will communicate with the user's remote console.

The statements are implemented, respectively, by the calls

```
    call display_(message);

    call reply_(answer);
```

The declaration for the arguments expected in display_ and reply_ is

```
    dcl (message, answer) char (*);
```

Display_ puts message on the I/O stream user_output. Reply_ reads the I/O stream user_input and sets the value of answer.

Exit and Stop

According to the PL/I manual these statements have identical effects in the absence of tasking. Since tasking will not be implemented in the first version of the Multics PL/I, their implementation could be identical. However for the sake of generality they will be implemented through calls to two different library subroutines:

```
    call exit_;

    call stop_;
```

Each types an informative message and returns the user to command level. [The concept of "command level" has yet to be defined precisely. However, at command level the user can "talk" with the system.]

Delay

The delay statement is probably intended primarily for use in tasking. However even though tasking will not be implemented in the initial version, delay should be since it may be used elsewhere. The statement is implemented by the call

```
     call delay_(n);
```

where n is declared

```
     dcl n fixed bin (71);
```

N is the number of milliseconds to delay. Delay_ will call a traffic controller entry which causes the process to be blocked for n milliseconds.