To: Distribution

From: N. I. Morris

Date: March 25, 1975

Subject: I/O Interfacer Specification Changes

Several changes have been made in the I/O Interfacer mechanism. The two most important of these is the removal of the queued connect feature and the removal of the hardcore status queue.

Device Assignment Changes

An extra argument was added to the pioi_$assign and pioi_$priv_assign calls. This argument is used to return a relative pointer to the configuration card describing the device just assigned. The calling sequence is now as follows:

```
declare pioi_$assign entry (fixed bin, char (*),
          bit(18) aligned, fixed bin(71), fixed
          bin(35));
call pioi_$assign (devx, devname, configrel, event,
          rcode);
```

devx — is the device index to be used in subsequent calls to the I/O Interfacer. (Output)

devname — is the name of the device being assigned. (Input)

configrel — is a relative pointer to the configuration card describing the device just assigned. A pointer to the configuration card may be made by taking ptr (addr (config_deck$), configrel). (Output)

event — as an event channel ID to be used in sending wakeups to the user upon receipt of status from the assigned device. (Input)

rcode — is an error code. (Output)

The call to pioi_$priv_assign is identical to the call to pioi_$assign.

Connect Changes

The queued connect feature was removed from the I/O Interfacer. Therefore, it is no longer permissable to call ioi_$connect or ioi_$connect_pcw while the device is running. If this is done, an error code of error_table_$device active will be returned to the caller. The calling sequences to ioi_$connect and ioi_$connect_pcw have not changed.

Status Changes

The hardcore ring status queue was removed from the I/O Interfacer. Instead, the user workspace buffer is used to provide a circular status queue for terminate, marker, system fault, and time-out status events. Special status is kept in the hardcore ring. It is not queued. Only the last special status event is available to the user. The ioi_$get_status entry has been removed. A new call, ioi_$set_status, has been provided to set the location and size to be used for the circular status queue in the user workspace. Each element of the status queue is identical to the structure described in ioi_stat.incl.pl1 and previously used in call to ioi_$get_status. Another call, ioi_$get_special_status, has been created to allow the user to retrieve special status.

```
declare ioi_$set_status entry (fixed bin, fixed
          bin(18), fixed bin(8), fixed bin(35));
call ioi_$set_status (devx, sqloc, sqlen, rcode);
```

devx — is the device index of a previously assigned device. (Input)

sqloc — is the offset in the user workspace segment of the status queue. (Input)

sqlen — is the number of elements to be used for the status queue. (Input)

```
declare ioi_$get_special_status entry (fixed bin,
          bit(1) aligned, bit(36) aligned, fixed
          bin(35));
call ioi_$get_special_status (devx, spi_flag,
          spi_status, rcode);
```

spi_flag — is a flag which is set to "1"b if a special interrupt has taken place and to "0"b otherwise. (Output)

spi_status — is the 36 bits of special interrupt status returned by a PSIA channel. If the channel

used for this device is a Common Peripheral Channel, spi_status will be all zeroes. (Output)

In addition, the status mechanism has been modified to return status information in the event message which is sent to the user when a wakeup is generated. The format of the information is shown below. In many cases, the 72 bits returned in the event message will contain sufficient information for the user to adequately analyze his status. If such is the case, the user need never make a call to ioi_$set_status to supply a status queue. In effect, status will be queued by the interprocess communication mechanism.

```
declare 1 imess based aligned,
        2 completion unal,
          3 st bit (1),
          3 er bit (1),
          3 run bit (1),
          3 time_out bit (1),
        2 pad bit (11) unal,
        2 level bit (3) unal,
        2 offset bit (18) unal,
        2 status bit (36) unal;
```

st — is a flag indicating status is present.

er — is an error flag. If it is equal to "1"b, the returned status indicates an error condition.

run — is a flag indicating whether or not the channel is still running. It is set to "1"b when marker status is returned and to "0"b in other cases.

time_out — is a flag indicating that a time out occurred. If it is set to "1"b, the channel has been connected for too long a time and has just been forcibly stopped.

level — is the interrupt level which caused status to be produced.

offset — is the offset in the user workspace of the DCW which caused status to be produced.

status — is the first 36 bits of the 72 bit status stored by the IOM.

Other Changes

Other changes have been made to the I/O Interfacer. The timeout mechanism has been correctly implemented and the call to ioi_$timeout now works. A call has been provided to force a given device to use only a specific channel in the cases where more than one channel could be used. (That feature is intended primarily for online T&D.) Another entry has been provided to allow changing the event channel ID to be used to wake the user.

```
declare pioi_$set_channel_required entry (fixed bin,
          fixed bin(3), fixed bin(6), fixed bin(35));
call pioi_$set_channel_required (devx, iom, chan,
          rcode);
```

iom — is the IOM number of the IOM containing the channel to be used for I/O for this device. (Input)

chan — is the channel number of the channel to be used. (Input)

rcode — is set to error_table_$bad_channel if the IOM and channel specified are not correct. (Output)

```
declare ioi_$set_event entry (fixed bin, fixed bin(71),
          fixed bin(35));
call ioi_$set_event (devx, event, rcode);
```

(END)