TO: MSPM Distribution
FROM: T. P. Skinner
SUBJ: BL.8.00
DATE: 02/07/68

Due to the creation of the new GIM, substantial deletions to the hard core I/O initialization procedures have been made.

Published: 02/07/68
(Supersedes: BL.8.00, 08/17/67,
BL.8.00, 06/07/67)

Identification

Hardcore I/O Initialization
R. C. Daley, D. R. Widrig, T. P. Skinner

Purpose

This section provides the specification of the procedures responsible for the initialization of the hardcore I/O system. These procedures are run under the control of the Multics Initializer control program. In addition, a special initialization procedure for use in assigning GIOC channels during the early stages of Multics initialization is specified.

Introduction

The initialization of the hardcore I/O system is accomplished in two stages. During the first stage, the GIOC interface module (GIM) and its data bases are initialized for subsequent use by the file system. During this stage, an I/O device configuration table (DCT) is constructed to contain entries for only those GIOC channels to be used during system initialization. Since, at this stage of initialization, the file system dynamic paging mechanism has not yet been initialized, the entire DCT, which may be quite large, cannot be constructed at this time.

The second stage of hardcore I/O initialization is called upon after the file system has been initialized and the dynamic paging mechanism has been established. At this time the construction of the DCT is completed.

Many of the data bases of the GIM are initialized as needed during the normal operation of these modules. As a result, no explicit action to initialize these data bases is required.

Initialization Procedures

At the appropriate point during Multics initialization, the Multics Initializer control program begins the initialization of the hardcore I/O system by means of the following call.

```
call io_init$one;
```

When this call is received, the following steps are taken to initialize the hardcore I/O system.

Step 1

Static storage for the entire Hard-core I/O system is initialized by means of the following call:

```
call init_hcio_stat;
```

Further references to the function of the Hard-core I/O system static storage may be found in BF.20.02.

Step 2

The Channel Assignment Table (CAT) is initialized by means of the following call;

```
call init_cat;
```

The CAT (See BF.20.03) for a detailed description of the data base) is divided in o two major sections; the per-GIOC section and the per-device section. Initialization of the per-device section requires only that all per-device entries be zero. Since the CAT is created as an all-zero segment, the per-device section is, by default, in an initialized state.

This procedure initializes the GIOC mail box pointers in the CAT from information contained in the system configuration tables.

The status channels are initialized by inserting proper addresses and tallies in the status control words in each status channel mailbox.

Step 3

A basic I/O device configuration table (DCT) is constructed by means of the following call.

```
call init_dct$part_1;
```

This procedure constructs a basic DCT from symbolic information contained in the ASCII segment "dct_sym_1". The basic DCT must contain entries for all GIOC channels used by the basic file system and at least one GIOC channel through which the Multics system tape may be accessed.

Control is returned to the Multics Initializer control program. The GIM is now available for use.

Stage 2

After the file system has been initialized and the dynamic paging mechanism has been established, the Multics Initializer control program continues hardcore I/O initialization by means of the following call.

```
call io_init$two;
```

When this call is received, initialization of the hardcore I/O system continues with the following step.

Step 1

The remainder of the DCT is constructed by means of the following call.

```
call init_dct$part2;
```

This procedure constructs the remainder of the DCT from symbolic information contained in the ASCII segment "dct_sym_2". Hardcore I/O initialization is now complete.