PUBLISHED: 9/19/66

Identification

Attachment and Detachment of Input/Output Devices and Pseudodevices.
J. F. Ossanna, V. A. Vyssotsky, G. G. Ziegler.

Purpose

This section describes the method of associating input/output devices and pseudodevices with a user's process. The calls attach, detach, noattach, and changemode are described herein in this context. Uses of these calls for auxiliary purposes, such as for streamname-framename associations, are partially described herein, and are detailed in other subsections of Section BF.

General Discussion

The act of attachment concerns the actual association of a specific device and medium with the attaching process. Following the attachment, the user can accomplish data transmission between the medium and his workspace. There are two basic prerequisites for successful attachment: (1) the user must have access privilege for both the device and the medium; and (2), if the device requires an advance facilities reservation, such reservation must be in effect at attachment time. An attach call is provided to effect attachment and a detach call is provided to cancel the attachment. A noattach call is provided to nullify a subsequent attach call. A changemode call is provided to alter the mode argument of the attach call.

Device and Medium Access Control

Access control for devices, media, and data generally involves four levels. These are the accessibility of: (1) a device class or type; (2) a particular physical or logical device among those of an accessible type; (3) the demountable storage medium, if any; and (4) the actual data. For example, in the case of tapes, these levels relate to a user's permission to use: (1) tapes; (2) a particular tape drive; (3) a particular tape reel; and (4) particular data on a reel. Multics, however, applies only the first three levels of access control; all of the data on any particular medium is considered to be accessible to anyone having access to the medium itself. Thus, access to a particular tape reel includes access to the data thereon. Device and media access control are separate for devices having demountable storage media such as tapes and data disc. They are separate too for File System files accessed through the I/O-System (IOS), where the File System is regarded as a device and its files regarded as media. Media access control is not provided for card, paper tape, printer, and typewriter equipment.

In summary, the three device and media access control levels in Multics are:

Level (1) Device class or type.
(2) Particular physical or logical device.
(3) Particular demountable storage medium.

At attachment time, the module performing the attachment is concerned with the access control verification at all three levels. The data bases containing the access control information for the three levels are respectively:

Level (1) User Profile.
(2) User Profile and Device Access Exception Files.
(3) Demountable Storage Registry Files.

The Device Access Exception Files contain the current exceptions to the level (2) access control information contained in the user profile. For example, a user having the right to use any public typewriter is ordinarily denied the right to attach a public typewriter currently attached to another user.

The responsibility for attachment is distributed in the IOS. The attachment for each device type is handled by the Device Interface Module (DIM) for the device type.

Advance Device Reservation

Certain types of devices are characterized by the number of their type that is or can be connected to the particular computer installation being small enough for this number to be frequently exceeded by the number required to fulfill random user demand. These devices are resources whose use must be scheduled. For such devices, the user must obtain an advance facilities reservation before requesting an attachment. These reservations are not provided by the IOS; they are obtained from the Transactor (see MSPM Section BT). The Transactor also makes use of the previously mentioned device access control data bases. In certain instances the existence of a reservation is adequate validation of level (1) and (2) access control.

The device types requiring advance facilities reservations are:

1. Tape drives.
2. Data Disc drives.
3. Line printers.
4. Card readers and punches.
5. Paper tape readers and punches.
6. Some typewriters.
7. Some communications lines.
8. Some directly referenced GIOC channels.

Items 3 - 5 above refer to the actual printers, readers, and punches, not to the corresponding pseudodevices representing advanced input and delayed output. The user can attach and detach devices without affecting the existence of any reservations. For example, detaching a tape need not release the reserved tape drive. Reservations are released by calling the Transactor. They may also be automatically released by expiring, by being pre-empted, or following the detection of certain default or error conditions.

Ionames

The term ioname (pronounced ī-ō-nām) will be used to generally refer to a broad collection of character string names used in user-IOS communication and within the IOS. Streamnames and framenames are two common kinds of ionames. Ionames are character strings of 1 to 32 characters. For ease of programming, the data character set of PL/I (Revised ASCII) is suitable; adherence to the graphic subset of character set will minimize confusion in reading printout containing ionames. Under some circumstances, ionames may be handled by parts of Multics other than the IOS; such handling can imply additional limitations on what characters are suitable for ionames. For example, the command language (Section BX.1.00) attaches significance to certain special characters, making these relatively awkward to use in character strings. The character subset suitable for PL/I identifiers is conservatively safe. A number of ionames have been reserved for use by privileged procedures or for internal use by Multics software. All such ionames end with an underline (e.g. abc_); consequently ionames ending with an underline should normally not be used.

General Form of the Attach Call

The attach call has a variety of uses including the attachment of devices to a user's process. The general form presented here and the accompanying discussion apply to almost all of the uses of this call. Most of the uses are at least touched upon in this section and appropriate references are given. The general form is:

```
call attach( ioname1 , type , ioname2 [, mode [, status ]])
```

The brackets, [ ], indicate optional arguments and sets of arguments in the usual way. The attach call is generally a request to associate ioname1 with a previously defined or otherwise known ioname2. The nature of the association is indicated by the argument type. If present, mode indicates in general a collection of modes which are to apply to the association of ioname1 with ioname2. Ioname1 and ioname2 are ionames as described previously. Type and mode are character strings of 1 to 32 characters. Status is a returned pointer pointing to a bit string containing information related to the association of ioname1 with ioname2. A detailed discussion of the format and meaning of the status bit string is given in BF.1.21. Additional discussion and a detailed listing of possible reasons for the failure of a call are found with each call description.

Device Attachment

The form of the attach call for requesting that a device or pseudodevice be attached is:

```
call attach( framename , type , description [, mode [, status ]])
```

Ionamel and ioname2 are respectively a framename and device description. Type must be a known device class or type. See BF.1.10 for a discussion of data 'frames'. An example is

```
call attach( "x" , "tape" , "reel_24972" , mode , status )
```

which requests that a 9-track tape drive with tape reel 24972 be attached and given the framename "x". The type "tape" results in the tape DIM receiving the attach call and being responsible for the actual attachment. The description "reel_24972" must be one understood by the tape DIM.

For each known type the list of appropriate descriptions is dependent on the implementation of the DIM or pseudo-DIM corresponding to the type. Table I lists the standard known device and pseudodevice types.

Table I. Standard Types and Corresponding Devices

Type	Device
tape	9-track tapes
tape7	7-track tapes
datadisc	data disc
print	pseudoprinter
punch	pseudocard-punch
read	pseudocard-reader
ptpunch	pseudopaper-tape-punch
ptread	pseudopaper-tape-reader
file	File System file
typewriter	typewriter (complete)
keyboard	typewriter keyboard
twprinter	typewriter printer
PRT202	printer
CPZ200	card punch
CRZ200	card reader
PTS200	paper tape reader/punch
BELL103	low speed character asynchronous channel
BELL201	character synchronous channel
BELL202	character asynchronous channel
BELL801	automatic calling unit channel
BELL301	high speed synchronous channel
channel	directly referenced GIOC channel

The mode argument may specify the setting of a collection of modes relevant to the intended use of the specified framename. Mode is a character string formed by concatenation of single characters each representing some mode setting. The mode need only be understandable by the module receiving the attach call. Thus mode is arbitrary insofar as the IOS is concerned. However certain modes have been standardized within the IOS and these are generally used by most standard DIMs. The following standard modes are described in detail in BF.1.02 and are summarized here for reference:

use mode	Readable (R) and/or Writable (W).
access mode	Random (D) or Sequential (Q).
	If Q, Forward only (F) or Backspaceable (B).
data mode	Physical (P) or Logical (G).
	If G, Linear (L) or Sectional (S).

An example of a legitimate mode for attaching a tape is "RWBS". The absence of a reference to a mode leaves the mode unset. Certain modes may be set in a negative sense; for example, mode = "notR" specifies a desire to be kept from reading the frame. (The "not" in "notR" represents the ASCII "overline" (PL/I "not") character.)

In addition to the common status and error information returned for all IOS calls, the following errors are indicated in the returned status for the attach call used for device attachment:

1. Type not known.
2. Description not known.
3. Framename already attached.
4. Framename cannot be "*".
5. Invalid mode.
6. Required reservation does not exist.
7. Device access not permitted.
8. Medium access not permitted.
9. Device cannot be attached to multiple users.
10. Medium cannot be attached to multiple users.
11. Device does not exist or cannot be found.
12. Medium does not exist or cannot be found.
13. Channel attached but device not present.

After a successful attachment a device terminal or frame terminal is said to exist at the entry points to the procedure associated in the attachment table with the specified framename (see BF.1.03). A specified framename cannot be associated simultaneously with a second device terminal. A subsequent attach call attempting to associate the framename with a different device will be rejected by the already associated device terminal. A subsequent attach call for the same device will be construed as an effort to specify additional mode settings in mode. This is discussed in more detail below and in BF.1.02.

Subframe Attachment

The current item of a sectional frame may itself be attached and be associated with another framename. The new frame is regarded as a subframe of the original frame. A complete discussion of the general structure of frames and subframes is given in BF.1.18. To request such an attachment, an attach call is issued with type = "item". For example,

```
call attach( "x" , "item" , "y" [, mode [, status ]])
```

requests the current item of frame "y" to be attached and given the framename "x". "y" must be a previously-defined framename representing a sectional frame. When the current item of frame "y" changes, frame "x" is automatically associated with the new current item. Mode must be compatible with the mode of frame "y" contained in the attachment table entry for "y" (see BF.1.18).

In addition to the common status and error information returned for all IOS calls, the following errors are indicated in the returned status for an attach call used for subframe attachment:

1. Original framename (ioname2) not known.
2. Sub-framename (ioname1) cannot be "*".
3. Incompatible mode.

Streamname - Ioname Attachments

The first argument of all user calls to the IOS is an ioname which is used by the I/O Switch to route the call to a previously-associated destination. This ioname may be a framename or it may be a streamname. (In certain specialized and less common uses of the attach call, the first argument is neither). The use of streamnames is described fully in Section BF.1.03; the discussion in this section is limited and primarily for reference.

The call to establish a streamname-ioname association is:

```
call attach( streamname , "ioname" , ioname [, mode [, status ]])
```

The type argument "ioname" indicates that the first argument is a streamname to be associated with a previously-defined ioname (third argument). The ioname may be either a framename or another streamname. Mode must be compatible with the mode contained in the attachment table entry for ioname.

In addition to the common status and error information returned for all IOS calls, the following errors are indicated in the returned status for an attach call used for streamname-ioname association:

1. Original ioname not known.

2. Streamname cannot be "*".
3. Incompatible mode.
4. Attachment would have resulted in an IOS loop.

A call to the IOS whose first argument is a streamname is routed indirectly to the destination associated with the ioname with which the streamname was associated. A given streamname may be associated with more than one ioname by additional attach calls. Also, any number of streamnames may be associated with a given ioname. A complete discussion of such multiple associations is given in Section BF.1.03.

Detachment

A detach call is provided to cancel a previous attachment. The general form of the detach call is:

```
call detach( ioname1 [, ioname2 [, disposal [, status ]]])
```

The detach call is a request to cancel the association of ioname1 with ioname2. If only one ioname is associated with ioname1, ioname2 need not be given or may be the null character string or may be "*". This situation exists, for example, when ioname1 is a framename. If ioname1 is a streamname, ioname2 may be either a streamname or a framename; further, multiple associations may exist with ioname1 and/or ioname2. When multiple associations exist with ioname1, ioname2 specifies which association is to be cancelled; if ioname2 is not provided or is the null character string or is "*", all ioname associations with ioname1 are cancelled. Under all circumstances, only associations involving ioname1 are cancelled.

Disposal is a character string of 1 to 32 characters. This argument is interpreted only when ioname1 is a framename. The contents of disposal must be understandable by the DIM or pseudo-DIM which will actually perform the detachment; appropriate disposals are dependent on the implementation of particular DIMs. However, certain conventions have been standardized within the IOS and these are generally observed by most standard DIMs. Like mode, disposal is formed by concatenation of single characters, each representing a choice from among various options. The standard options are:

R	Release any facilities reservations, or
H	Hold any facilities reservations.
U	Unmount detachable storage medium, or
M	Keep detachable medium Mounted.
S	Save detached medium, or
A	Return detached medium to Available pool.
V	Return saved medium to Vault, or
C	Detached medium is to be Carried away by user.

Default options are R, U, S, and V. For example,

```
call detach( "x" , "" , "HS" , status)
```

may request detachment of the device associated with frame "x" with the following options: (1) the facility reservation is held; (2) the medium is unmounted; (3) the medium is saved for the user; and (4) the medium is stored in the media vault.

In addition to the common status and error information returned for all IOS calls, the following errors are indicated in the returned status of a detach call:

1. Ionamel is not attached.
2. Ioname2 is not associated with ionamel.
3. Ionamel cannot be "*".
4. One or more attachments replaced by default ones.
5. Invalid disposal.

The status argument is a returned pointer to a status bit string as described in BF.1.21.

Changing the Attachment Mode

At any time during the lifetime of an attachment the mode argument may be resupplied to cause mode changes or to set previously-unset modes. Not all mode changes are possible and a set mode cannot be unset. Section BF.1.02 contains a full discussion of the standard IOS modes. A new mode argument may be resupplied two ways. One way is to use the changemode call, whose form is:

```
call changemode( ioname , mode [, status ])
```

Ioname may be a streamname or a framename. The requirements on the legitimacy of mode are the same as those that would be imposed if the pertinent attachment were only now being made, except that certain modes cannot be changed. Any modes not mentioned in mode are not changed or remain unset. Modes which are mentioned in mode are set, if previously unset, or are changed, if previously set. The use mode is an example of a changeable mode; for example, "R" may be changed to "notR". The data mode is an example of an unchangeable mode; for example, "P" may not be changed to "G".

Unset modes may be set by default. For example, an unset Readable mode is set to "R" by the advent of a read call (provided "R" is legitimate).

The second way to resupply the mode argument is to resupply the attach call. That is, an attach call whose first three arguments effectively duplicate those of an earlier attach call is regarded as a changemode call.

The effect on mode legitimacy of the mode associated with subsequent ionames (if any) involved in the flow-of-control or iopath is discussed in Sections BF.1.02 and BF.1.03.

In addition to the common status and error information returned for all IOS calls, the following errors are indicated in the returned status of a changemode call:

1. Ioname is not attached.
2. Ioname cannot be "*".
3. Invalid mode.

The status argument is a returned pointer to a status bit string as explained in Section BF.1.21. In addition, for an attach call behaving as a changemode call, errors (1), (2), and (4) for a device attachment apply.

Attachment Prevention

An attach call of any form may be prevented from having any effect by preceding it with an appropriate noattach call. The noattach call is provided primarily to permit attach call substitution without source language modification. Thus a noattach call is usually used paired with and following an attach call which is meant to replace the one being nullified. Such attach call substitution is a convenient technique for temporarily altering a program's input/output devices or stream-frame associations. The general form of the noattach call is:

```
call noattach( ioname1 [, type [, ioname2 [, status ]]])
```

A subsequent attach call with the same ioname1 and type and the same or equivalent ioname2 (e.g. equivalent descriptions) will be ignored. If ioname2 is not provided or is the null character string or is "*", a match is required only on ioname1 and type. If type is not provided or is the null character string or is "*", a match is required only on ioname1; ioname2 is ignored. Ioname1 must be provided.

A noattach call can nullify only one subsequent attach call. The two calls effectively annihilate each other. An outstanding but unused noattach call can be rendered ineffective only by issuing a matching attach call. Only in this exceptional use of the attach call is either type or ioname2 allowed to be "*".

In addition to the common status and error information returned for all IOS calls, the following errors are indicated in the returned status for the noattach call:

1. Equivalent noattach call outstanding.
2. Ioname1 cannot be "*".
3. Type not known.

Other Forms of the Attach Call

A form of the attach call is provided for temporarily adding previously unknown types and for specifying the corresponding procedure names. A similar form of the attach call allows module substitution by associating a new procedure name with a known type. Such alteration and extension of the IOS is discussed in detail in Sections BF.1.06.

Another special use of the attach call occurs when mode contains "E", specifying an "emergency" device attachment. Such an attachment suspends any current input/output on the attached device and allows immediate initiation of new input/output. The emergency attachment mode permits immediate interpolation of a new series of input/output transactions. A corresponding detach with disposal containing "E" reinstates the previous input/output. This use of the attach call is described in detail in Section BF.1.04.