10. MISCELLANEOUS

10.1 (GET TDB)

Action Value (D0.L): 7

Additional Call Parameters

None

Return Parameters

A0 : address of TEP Definition Block
A1 : address of default profile

All other registers except A7(SP) are assumed corrupt.

Error Returns

None

Description

This call is made by the CAE on the TEP.

The procedure returns the (immobile) addresses of the TEP Definition Block (see section 5.2) and the TEP's Default Profile (see section 5.3.6).

11. CARTRIDGE BASED PROGRAMS

TEPs held on cartridge are not supported at Release 1.

Computer Access: Interfaces for Terminal Emulation Programs

OPD/SPEC/CA1

0. DOCUMENT CONTROL

0.1 Contents

0.2 Changes Since Previous Issue

See Change bars

0.3 Document Predecessors

None

0.4 Changes Forecast

Issue 6/1 is intended to be a final specification of the Release 1 product.

0.5 Document Cross References

[1] R51004	OPD Handbook
[2] PSD 76.97.3.1	OPD Kernel Specification
[3] PSD 76.97.3.2	OPD Director Facilities for Application Writers
[4] OPD/SPEC/3	Printer Manager

1. GENERAL

1.1 Scope

This document provides the technical information necessary, in addition to that in [2] and [3], to write a Terminal Emulation Program that is to function within the standard OPD Computer Access Environment.

1.2 Introduction

The OPD hardware and the lower levels of the OPD software embody the technical facilities necessary to enable a wide range of computer terminals to be emulated on OPD and to communicate with distant hosts via telephone connections.

It is expected that a number of such terminal emulations will be provided on OPD, some by ICL and some by independent software suppliers.

The OPD Computer Access subsystem has been designed so that the parts of a terminal emulation which are specific to a particular terminal type are contained in a Terminal Emulation Program, which operates within a standard Computer Access Environment provided by ICL. This approach has the twin objectives of providing a common user view of the outer systems aspects of all terminal emulations, and of enabling terminal emulations to be written with much of the complexity of OPD application writing concealed within the standard environment.

Most of the information needed to write a Terminal Emulation Program is contained within this document, but the writer will also need to refer to [1] for details of the end user interface, to [2] for details of input and output character encodings, for advanced screen display facilities if required, and any other Kernel facilities needed to achieve special effects, and to [3] for details of program formats and for background information on application structure.

1.3 Terminology

CAE Computer Access Environment
TEP Terminal Emulation Program

General knowledge of the terminology of [2] and [3] is also assumed.

2. SUMMARY

The OPD Computer Access subsystem is designed as a standard Computer Access Environment, providing a standard user control interface and common services, and a number of Terminal Emulation Programs providing the terminal-specific elements of particular emulations. This document provides, for ICL and approved external software suppliers, the technical information necessary to write such a Terminal Emulation Program.

3. ARCHITECTURAL OVERVIEW

3.1 Objectives

The CAE has been designed so that, as far as possible, the TEP is shielded from low level knowledge of the OPD system software and architecture.

3.2 Processes

The actions jointly performed by the CAE and the TEP occur within Processes. A connection to a live computer service runs as two Processes: a Receive Process, which receives input from the remote service and outputs to the screen (and, on demand, to the page store) and a Transmit Process, which receives input from the keyboard and transmits data to the remote service.

When the user views a page from the page store, a Display Process is used to read from the page store and output to the screen.

When the user chooses to print a page from the page store, a Print Process is used to read from the page store and output to the printer. Printing direct from the live service is handled by writing the data to a temporary page in the page store (which, to the Receive Process, is indistinguishable from writing a permanent page), and then printing the temporary page using a Print Process.

Each Process is in fact a trusted secondary activity (see [2] and [3]). The CAE establishes the activities when required, and calls the TEP code as a subroutine within the activity. As far as the TEP is concerned, it has a number of subroutines corresponding to the process types, which are entered on demand and may execute concurrently. (These process subroutines have a common entry point: the process type is distinguished by a value in a register.)

There is also a Control Process, which is used by the CAE to coordinate the actions of the other processes. The TEP does not need to be aware of the Control Process (although some calls on miscellaneous TEP subroutines may occur in that Process).

3.3 Devices

In most cases the TEP does not access devices directly at the Kernel level (see [2]), but uses the customised CAE interfaces. The TEP can however choose to drive the screen and/or printer itself, either alone or in conjunction with the CAE, if it needs to use features not available through the CAE interfaces. [This choice is not supported at Release 1 except in certain specific cases.]

3.4 Events and semaphores

It is recommended that the TEP should not use events and semaphores, because of the problems of interaction with those used by the CAE. The CAE interfaces are intended to provide the necessary interlocks and waits likely to be required by the TEP.

Indeed, at Release 1, the use of events by the TEP is not permitted. At later releases, if events are used, they must be local events with numbers in a specified range. The TEP must never seek notification of an event outside this range.

If semaphores are used, they must not be held over a call on the CAE, and the TEP must not wait for a semaphore in a way which could leave outstanding events unnotified.

3.5 Noticeboard

The TEP should preferably not send messages directly to the Output Area of the Noticeboard. CAE procedures will be provided [but not at Release 1] for the TEP to declare an error or other status which will be displayed as appropriate by the CAE.

The TEP can set Noticeboard flags, subject to the general restrictions on use of flags (see [3]).

3.6 Sound generator

The TEP can generate sounds through direct use of the Director procedure (see [3]).

3.7 Store

The TEP can acquire and release store through the standard system interfaces (see [2] and [3]). The TEP is responsible for freezing, thawing, registering and in particular de-registering in accordance with system recommendations. The READ MODEM procedure (section 6.3.1) provides implicit thawing/freezing in certain cases.

3.8 Error handling

The TEP is assumed to be of the same integrity as system software, and its actions are not in general validated by the CAE. It is expected to tidy up and terminate cleanly if it encounters an unexpected internal error or store corruption. It must detect and report as appropriate any errors that can arise from incorrect user actions or data transmission failures.

4. INTERFACE CONVENTIONS

4.1 Calls on the TEP from the CAE

All entries are by subroutine call on the entry point specified in the program header (see [3]) of the TEP. On entry, A6 points to the Computer Access Control Block (CACB, see section 5.1) and D0.L contains an action value specifying the action (e.g. the type of process) required of the TEP.

4.2 Calls on the CAE from the TEP

All calls are subroutine calls performed by means of the instruction JSR (A6), where A6 points to the CACB. D0.B contains an action value specifying the call type. Other registers may contain parameters to the target procedure. On exit from such a call, registers other than D0 which are not defined to return values are unchanged. D0.L contains a response value which in general is positive or zero following a successful call, and negative after an unsuccessful call. CC is set according to the response value in D0.L. Negative responses follow the Kernel conventions defined in [2]; the values are identified by symbolic names of the form ERR.cc and are defined in the Kernel Include file ERRORS.

Procedures which are stated to be not implemented at Release 1 will return ERR.NI. A call of a procedure from an inappropriate process will return ERR.NA. Apart from these cases, the only negative responses returned are those stated in the procedure specifications.

5. DATA STRUCTURES

5.1 Computer Access Control Block

The CACB is the master control block for the Computer Access subsystem. It remains at a constant address throughout execution of the subsystem, and is by convention pointed to by A6. This address is passed on all calls between CAE and the TEP. It is recommended that A6 never be corrupted, even temporarily.

The fields in the CACB are as follows. Their offsets are defined in the Include file CACBVALS. Field lengths are given as B (byte), W (word) or L (long word).

CA.PROC		Target of JSR instruction when TEP makes call on CAE (The offset of this field is zero.)
CA.RACT	L	Receive Process activity identifier
CA.TACT	L	Transmit Process activity identifier
CA.DACT	L	Display Process activity identifier
CA.PACT	L	Print Process activity identifier
CA.RSCR	L	Receive Process screen channel
CA.DSCR	L	Display Process screen channel
CA.PPRT	L	Print Process printer channel
CA.RTOT	W	Data associated with interrupt from Receive to Transmit Process
CA.TTOR	W	Data associated with interrupt from Transmit to Receive Process
CA.TXOFF	B	Character sent from OPD to remote service to request XOFF (see section 6.1.1)
CA.TXON	B	Character sent from OPD to remote service to request XON
CA.RXOFF	B	Character sent from remote service to OPD to request XOFF
CA.RXON	B	Character sent from remote service to OPD to request XON
		Each of the above characters may be set negative, indicating that there is no XOFF/XON capability in that case

CA.SENDK	B	Code that is to be sent on use of RETURN key (see section 7.1)

[The Include file CACBVALS may define additonal locations. These should not be accessed by the TEP.]

5.2 TEP Definition Block

5.2.1 DESCRIPTION

The TEP Definition Block (TDB) is a fixed block of data defining the properties and capabilities of the TEP. It is located within with TEP's program area. The CAE makes a GET TDB call (see section 10.1) on the TEP, and the TEP returns the address of the block.

The TEP Definition Block comprises a number of close-packed fields. Each field has the following structure:

byte 0 — Field identifier (1 to 254)

byte 1 — Field length, including these organisational bytes 0 and 1 (2 to 255)

byte 2 onwards — Field data (field length - 2 bytes)

Fields are stored in ascending order of field identifier. The last field in the block is immediately followed by a terminating byte containing the value 255.

TDB fields are of two types: TEP Definition Fields, which define properties of the TEP as a whole (see section 5.2.2) and Profile Definition Fields, which define the meaning of fields used in profiles associated with the TEP (see section 5.2.3).

5.2.2 TEP DEFINITION FIELDS

TDB field identifiers for TEP Definition Fields are as follows:

96 User-visible name of TEP. The name consists of 1 to 22 graphic characters. It is used by the CAE whenever the TEP has to be identified to the end user. It is also remembered by the CAE in association with any stored profile or page, so that the CAE knows with TEP to invoke. The field data contains the TEP name, with no leading or trailing spaces. Field 96 defines the TEP name in capital letters, suitable for use in display headings, e.g. "GLASS TELETYPE" (but subject to any established conventions on representation of the name).

97 This field defines the TEP name in lower case with leading capitals, suitable for use in menus and other non-heading displays, e.g. "Glass Teletype" (but subject to any established conventions on representation of the name).

98 Stack space requirement of Receive Process. The field data is two bytes long, and contains (as a binary number) the number of bytes that the TEP Receive Process will use on its CAE-supplied stack. The space requested must include the space required by any subsystems that the Receive process

Receive Process calls, e.g. Director. Calls on CAE procedures require 128 bytes on stack.

99 Stack space requirement of Transmit Process.

100 Stack space requirement of Display Process

101 Stack space requirement of Print Process.

105 Function key support. The field data is four bytes long. The first byte defines which of the standard CAE function keys are supported during a live service, and the second byte which function keys have TEP-specified meanings. If bit n is set, function key n is supported. The third and fourth bytes provide the equivalent information for the case where a page is being displayed from the page store. The following keys are not 'replaceable' by the TEP: in a live service, f1, f6 and f7; in page store display, f1 and f4. The user-visible meanings of TEP-specific function keys are specified by fields 106 to 113 below. If a particular function key is given a TEP-specific meaning both in the live service and in page store display, the TEP specific meaning is the same in both cases.

106 to 113 Field 106 + n gives the TEP-specific definition of function key fn. This field is present only if fn is defined (by field 105) to have a TEP-specific meaning. The first byte of the field data has bit 0 set if and only if the definition applies during a live service, and bit 1 set if and only if the definition applies during page store display. Bits 2 to 7 are zero. (This information must be consistent with that given in field 105.) The subsequent bytes of the field data are graphic characters (up to 40) to produce one whole line on the appropriate Functions display, in a style consistent with that produced by the CAE for the CAE functions (see [1] for examples).

114 to 123 Each of these fields, if present, defines a further line to be output on the Functions display for the live service and/or the page store. These lines, which are displayed in the order defined following the function key definitions, may be used to define additional special keys or other information relevant on the Functions displays. These fields are defined in the same way as field 106. The maximum number of such additional lines is limited by the space available on the display, and depends on the number of function keys that are defined.

5.2.3 PROFILE DEFINITION FIELD

A Profile Definition Field occurs in the TEP Definition Block for each type of TEP-specific field that may occur in profiles associated with the TEP. The Profile Definition Field defines the syntax of occurrences of the field in profiles, and is used by the CAE Profile Editor to control editing of the profile by the user.

A Profile Definition Field can also be used to replace the standard definition of a CAE-defined profile field, but it must not thereby change the semantics of the field value as understood by CAE. This technique may be used in particular to cause a field to be invisible to the user but to have a value other than the CAE-defined default.

The identifier of a Profile Definition Field is the same as the identifier of instances of the Profile Field whose format it defines.

The field data of a Profile Definition Field is defined as follows:

Byte 2: Bit 0:
- 0 Field contains variable text
- 1 Field is set to one of a set of discrete values

Bit 1:
- 0 Field is not copyable
- 1 Field is copyable: such a field is copied across when the user creates a new profile by editing from an existing profile

Bit 2:
- 0 Field is not secret unless it is defined as secret in the instance of the field
- 1 Field is always secret: if such a field is displayed (except when it is first entered) each character is represented by a delete character. A secret field (whether or not it is ever displayed) is held in the Profile Store in encrypted form. Encryption and decryption are performed by CAE. [Encryption is not provided at Release 1.]

Bits 3 to 7: 0 (reserved)

Byte 3 : Length of Profile Field Title (= n). This title is displayed for the profile field when the profile is displayed or edited. If the length is given as zero, the field is not displayed, and no further bytes in the

the Profile Definition Field are required.

Bytes 4 to n+3 : Profile Field Title

Byte n+4 : Field Value Length. For a field with discrete values, this byte defines the (fixed) length of the displayed representation of each value. For a field with a text value, this byte defines the maximum length of the field value.

Byte n+5 : For a text value field, this byte defines the validation to be performed by CAE as the user inputs/edits the field.

Bits 0 to 4 define the validation performed on each input character. At least one of these bits must be set. If a supplied character falls into any group for which the bit is set, the character is accepted.

Bit 0 - Alphabetic (upper or lower case)
Bit 1 - Numeric
Bit 2 - Numeric or space
Bit 3 - Printable ASCII character, $20 to $7F inclusive, or $CA or $CB
Bit 4 - Telephone digit ($20 to $39 inclusive, or $CA)

Bits 5 to 7 define the validation performed on the whole field when the user declares that it is complete:

Bit 5 - Field must be an identifier, that is, the first character must be alphabetic and any remaining characters alphanumeric. When this bit is set, bits 0 and 2 should also be set.
Bit 6 - Text must not start with a space
Bit 7, if set, causes the CAE to call the TEP to perform further validation of the field when the CAE validation is complete.

Byte n+5 onwards : For a discrete valued field, these bytes contain a close-packed sequence of n-byte strings, each string being the displayable representation of one of the field's possible values. The length of the whole Profile Definition Field must be such that it exactly accommodates the last byte of the last string.

5.3 Profile Store

5.3.1 DESCRIPTION

The Profile Store contains the Computer Service Profiles which the end user has set up. A profile is a service description containing operational parameters relevant to the running of a particular computer service (or group of related services). Each live computer service session is run under the control of one profile; a manually connected service session can run either under a normal profile or under a temporary profile set up for the duration of that session only.

Each profile consists of a number of fields. Each field has the following structure:

byte 0	field identifier (1 to 255)
byte 1	field length, including these organisational bytes (3 to 255)
byte 2	field status, defining certain properties of the field
byte 3 onwards	field data (field length - 3 bytes)

The TEP accesses the profile at the field level. The CAE is responsible for manipulation of fields within profiles and profiles within the Profile Store.

The Receive Process and the Transmit Process of a TEP can read and write the profile associated with the current live service. (Writing to a profile normally occurs under the control of the CAE when the user explicitly creates or amends profiles, but implicit writing is possible to achieve, for example, automatic 'learning' of log-on sequences, if the TEP offers such a feature.) [At release 1, the TEP cannot write to the profile.]

The CAE will invoke the TEP to validate TEP-specific profile fields [but not at Release 1].

The procedures associated with profile manipulation are defined in section 9.

5.3.2 PROFILE FIELD STATUS

The Profile Field Status byte is defined as follows:

bits 0 to 3 : If profile field value is 'text', these bits are reserved (zero). If profile field value is 'discrete', these bits (treated as a binary number counting from 0) define that the field value is the nth value in the corresponding Profile Field Definition.

bit 4 to 5 : reserved (zero)

bit 6 : This bit is reserved for use by the CAE's encryption mechanism. At the TEP interface it is always zero.

bit 7 : 0 field is secret only if the 'secret' bit is set in the corresponding Profile Field Definition

1 field is secret

5.3.3 PROFILE FIELD IDENTIFIERS

The list below defines the allocation of Profile Field Identifiers, and gives details of those fields whose Profile Field Definition is implicitly supplied by the CAE.

1 Profile name (Text field, maximum length 12)

2 Description (Text field, maximum length 56)

6 Line speed (Discrete field, values 0(1200), 1 (300))

8 Parity (Discrete field, values 0(Even), 1(Odd), 2(None))

10 Stop bits (Discrete field, values 0(1), 1(2)

Unassigned identifiers in the range 3 to 15 are reserved for similar physical connection properties.

16 Auto LF (Discrete field, values 0(N),1(Y))

17 to 31 Reserved for CAE

32 to 47 Reserved for TEP

48 to 63 Reserved (unassigned)

64 to 79 User's stored key sequences associated with ALT/A to ALT/P (Text fields, maximum length 68)

80 to 89 Additional user-supplied strings associated with ALT/Q to ALT/Z. (Text fields, maximum length 68.) These strings may be used to convey additional control information to the TEP that is not amenable to representation in other profile fields. The string associated with ALT/X is conventionally used for flow control information (see section 6.1.1). The CAE does not impose any connection between the string called ALT/letter (Q to Z) and the use of the key ALT/letter during live service or page display

90 to 95 Reserved (unassigned)

96 to 127	Not used (to avoid conflict with TEP Definition Block)
128 to 254	Reserved (unassigned)
255	The degenerate profile field consisting solely of an identifier byte set to 255 terminates the profile

5.3.4 PROFILE FIELD DATA

For profiles with 'text' values, the Profile Field Data contains the string of bytes comprising the text. The length of the text is implied by the length of the whole profile field.

In the case of a TEP-specific field which is not displayed (byte 3 of its Profile Definition Field is zero), the content of the Profile Field Data (if any) is entirely up to the TEP.

5.3.5 PROFILE DISPLAY CONVENTIONS

When a profile is displayed to the user, each displayable field occupies one line. The field title followed by a space is displayed in 40-column mode at the start of the line. The field value is displayed in 80-column mode starting at the 25th column (13th column for ALT/A to ALT/Z). The title must not exceed 11 characters and the value must not exceed 56 characters (or 68 for ALT/A to ALT/Z).

5.3.6 DEFAULT PROFILE

Each TEP must provide a default profile, for use when the user creates a profile from scratch or chooses to manually connect a service using a default profile.

The default profile is an array of constant data located within the TEP's program area. Its address is obtained by the CAE by means of the GET TDB call (see section 10.1).

The default profile comprises a number of close-packed profile fields, in ascending order of field identifier, and terminated by a field identifier of 255. The profile must contain an instance of each field that is to appear in the user's view of the profile, plus any non-displayable fields that the TEP requires. There must be a profile name field (identifier 1) and a description field (identifier 2), each specifying a text of zero length.

If any field with a CAE-supplied Profile Field Definition is not present in a profile, CAE will take its value to be null if it is a 'text' field, and the 0th value if it is a 'discrete' field.

5.4 Page Store

5.4.1 DESCRIPTION

The Page Store contains up to 99 pages that the user has chosen to retain for future reference. Each "page" is a slice of the displayed output of a computer service session. It may represent a "snapshot" of the screen contents at an instant during the session, or a "monitor file" containing all the data displayed during a period of time in the session (which may be more than one screenful). It is expected that all TEPs will support Snapshots. TEPs that operate in line (scroll) mode should also support Monitor Files; those that operate in page mode may find this inappropriate if the data stream does not comprise discrete screens.

The user can at any time choose to view the Page Store, and move backwards and forwards through the existing pages by use of the cursor left and right keys. Selection of pages within the Page Store is handled by the CAE. When a page is selected, a Display Process (see section 8) is invoked to display it.

Pages fall into three classes for display purposes:

1. Those that require special display features (for example, non-standard fonts), and that can therefore be displayed only by the Display Process of the appropriate TEP. If the TEP is not loaded, the user is so advised when he attempts to display the page.

2. Those that use entirely standard display features (see section 5.4.3 for details). Such pages are displayed by an implicit Display Process in the CAE, and the TEP is not invoked.

3. Those that conform to restrictions (see section 5.4.3) such that they give a usable, if incomplete, display if they are displayed by the implicit Display Process of the CAE (because the appropriate TEP is not at that time loaded).

In the case of a Monitor File whose display occupies more than one screen, scrolling of the display must be under the user's control. For a line mode page he uses Cursor Down to scroll one line, and the shifted version of that key to get the whole next screen. For a page mode page, both keys give the next whole screen. The control of scrolling is normally performed automatically by the CAE, and implies conventions on the way in which the TEP uses the screen. Alternatively the TEP can undertake the scrolling itself [but not at Release 1]. [Backward scrolling is also an eventual requirement, but is not supported at Release 1.] While the user is viewing a page, he can choose to print it. In this case, a Print Process (see section 8) is invoked. The whole page is printed (irrespective of what part of it is currently displayed). The Display Process is unaffected:

the user can scroll it, or move to a different page while the Print Process continues to operate.

Pages fall into three classes for printing purposes, in a way analogous to the classification for display. Additionally, a page may be defined to be not printable at all (except by use of the OPD PRINT key).

5.4.2 PAGE PROPERTIES

Associated with each stored page there is a set of Page Properties, which are supplied by the TEP (through the BEGIN PAGE call) when the page is stored, and are made available to the TEP when the page is subsequently displayed or printed.

The Page Properties comprise a row of bytes, defined as follows:

Byte 0: Set to (binary) 1

Byte 1: Length of Page Properties including bytes 0 and 1

Byte 2: Bit 0:
- 0 Page is a snapshot
- 1 Page is a monitor file

If the page is a snapshot, the scrolling options are suppressed on the Page Functions display.

Bit 1:
- 0 Page is a 'page mode' page. It consists of one or more distinct whole screenfuls of display.

[At Release 1, 'page mode' is not supported, except for snapshots.]

- 1 Page is a 'line mode' page. It consists of one or more lines of display, which can be scrolled a line at a time.

Bit 2:
- 0 Page cannot be displayed by TEP (and so will be displayed by CAE and the TEP Display Process will never be invoked)
- 1 Page can be displayed by TEP

Bit 3:
- 0 Page cannot be displayed by CAE (and can thus be displayed only if the TEP is loaded)
- 1 Page can be displayed by CAE (CAE will display the page if bit 2 is zero or if the TEP is not loaded). At least one of bits 2 and 3 must be set.

Bits 4 and 5: These bits are defined in the same way as bits 2 and 3, but refer to the ability to

print (rather than display) the page. If neither of bits 4 and 5 is set, the page cannot be printed (except by use of the OPD PRINT key).

Bit 6: 0 The page has 80 characters per line
1 The page has 40 characters per line
If the TEP displays or prints the page, it can change the setting dynamically from the initial value specified by this bit.

[At release 1, if the CAE displays or prints the page, this bit is assumed to be zero.]

Bit 7: Reserved (zero)

Byte 3: Bit 0: 0 Scrolling of the display is handled by the CAE
1 Scrolling of the display is handled by the TEP
[At Release 1, this bit is assumed to be zero.]

Bit 1 0 The line separator character causes no display/printing
1 The line separator character is treated as CR/LF
[At Release 1, this bit is assumed to be 1.]

Bit 2 0 The page separator character causes no display/printing
1 The page separator character is treated as FF

Bits 3 to 7: Reserved (zero)

Byte 4: Line separator character

[At Release 1, this character must be Newline ($1C), unless the page is a 'page mode' snapshot, in which case its value is not significant.]

Byte 5: Page separator character
[At Release 1, this character is not used.]

Byte 6: Escape character

These three special characters are explained in section 5.4.3. The value $20 (space) for any of these characters indicates that no character of that type is to be expected.

[A translation table may be allowed, but not at Release 1.]

5.4.3 PAGE DATA

The Page Data comprises the sequence of bytes supplied via the PUT BYTE and PUT STRING procedures (see section 6).

If the page can be displayed or printed by the CAE, or is to be scrolled during page store display by the CAE, the page must contain separator characters with values as specified in the Page Properties. If the page is a line mode page, it must contain a line separator character between the data for each line of display. If the page is a page mode page, it must contain a page separator character between the data for each page of display. (There may be some hybrid cases which require both types of separator.) The chosen separator characters must be different from any that can occur in the displayable data itself. The Page Properties indicate whether the separator character is also to have a displayable effect.

The Page Data may also contain Escape sequences [but not at Release 1]. An escape sequence starts and ends with the Escape character defined in the Page Properties. The intervening characters can be any characters other than the Escape character. The Escape sequence is used to hold data which is relevant to the TEP when the TEP is processing the page, but which is to be ignored by the CAE when the CAE is processing the page. It might indicate special display attributes which take effect only if the TEP is available. An escape sequence can contain the characters defined as line and page separator characters, but they will not be recognised as such by the CAE.

If the CAE is displaying or printing the page, it processes all other characters in the range $00 to $7F in the way implied by the user specification of the Glass Teletype TEP.

If the CAE is to handle scrolling during page store display, then in order to accommodate backward scrolling [which is an eventual requirement although not supported at Release 1] each line or page separator character must be followed by any attribute settings which, but for the backward scrolling requirement, might have been assumed to have been carried over from the preceding line or page.

5.5 Translation Tables

The CAE provides for the translation of the values of bytes on input or output. The translation table consists of a close-packed row of byte-pairs. When a byte is passed whose value matches the first (lowest addressed) byte of a byte-pair, it is replaced by the corresponding second byte from the byte-pair.

This feature is intended for the translation of a small number of values. If a large number of values require translation, it will be more efficient for the TEP to supply an array of values which can be accessed using the old value as an index. [This feature may be incorporated into the CAE, but not at Release 1.]

A zero length translation table implies no translation.

5.6 Program structure

5.6.1 IMPLICIT TEPs

Implicit TEPs are those that have conventionalised program names and are recognised automatically by the CAE if they are currently plugged into the OPD. Since the names belong to a small fixed set, it is intended that implicit TEPs will be provided, procured or authorised only by ICL.

The program name of an implicit TEP is

<tilde>D<letter><9 spaces>

where <letter> is one of A to T. The following letters are currently allocated:

A Viewdata
B Glass Teletype
C (CO3 server package)

The implicit TEP, which can reside only in ROM (not on cartridge) has a standard program header (see [3]) declaring the properties: trusted, machine-code, not application. The entry point specified in the program header is used by the CAE for calls on the TEP.

5.6.2 EXPLICIT TEPs

Explicit TEPs are those that have a free choice of program name, but are not recognised automatically by the CAE; the user has to introduce them before use (see [1] and section 11).

Explicit TEPs are not supported at Release 1.

The program name of an explicit TEP is normally the same as the 'user-visible name' specified in the TEP Definition Block (see

section 5.2). If the user-visible name exceeds 12 characters, the 'menu-name' feature of the program header should be used. If the TEP is on cartridge, however, the 'menu-name' feature cannot be used, and the program name must be a contraction or other modification of the user-visible name.

The explicit TEP can reside either in ROM or on cartridge. It has a standard program header declaring the properties: trusted, machine-code, application, visible, (auto-start if required), not always reviewable.

The entry point specified in the program header is used when the system first enters the application. The TEP declares the entry point to be used for calls from the CAE through the interface specified in section 11.

6. RECEIVE PROCESS

6.1 Description

The Receive Process is the process whose primary task is to receive data via the modem link from a live computer service and display it on the OPD screen.

6.1.1 MODEM ACCESS

The data is read a byte at a time using the READ MODEM procedure. Selected characters can be converted to different values within the procedure via a translation table supplied by the TEP. The response from READ MODEM (and certain other procedures) is also used to notify changes of status affecting the operation of the Receive Process.

Data received from the modem is buffered by the CAE. By default, automatic flow control is provided by the CAE when the modem link mode is full duplex or a back channel is enabled: an XOFF character is sent when the buffer is nearly full, and an XON character is sent when the buffer becomes sufficiently empty. The TEP can use the CAE procedure CONTROL FLOW to inhibit or enable automatic flow control and to transmit explicit XON and XOFF. These characters can also be sent directly by the Transmit Process. The CAE does not detect flow control characters received from the host; this is the responsibility of the TEP.

The default XOFF character is DC3 ($13) and the default XON character is DC1 ($11). Some computer services use different characters, and the TEP may wish to allow the user to configure them by profile. The conventional way of doing this is by use of the 'ALT/X' string. If this string is present in the service profile, $40 is subtracted from each of its first four characters, and the results are stored in locations CA.TXOFF,CA.TXON, CA.RXOFF and CA.RXON (see section 5.1). Thus the character S in the string becomes the flow control character CTRL/S. If no ALT/X string is present, the locations are set to the standard DC3/DC1 values. A negative character implies that there is no flow control character for the case in question. The user conventionally puts - (hyphen) in the ALT/X string to indicate this. The values in CA.TXOFF and CA.TXON are used by the CAE when it performs automatic flow control or is asked to send a flow control character explicitly. The TEP could choose to write values into the XOFF/XON locations itself, instead of using the ALT/X technique.

With 7-bit data, a byte received with incorrect parity is converted to $FF. 8-bit data is not supported at Release 1.

6.1.2 SCREEN DISPLAY

The received data is interpreted according to the specification of the TEP, and is then usually displayed on the OPD screen.

This can be done using the Kernel screen handling procedures directly (see [2]), or more simply by the CAE procedures PUT BYTE and PUT STRING. These procedures effectively buffer the data and handle the screen interface on the TEP's behalf, enabling the TEP to process further received data without concern for the current state of outstanding screen transfers.

If the Receive Process needs to perform queued transfers to the screen itself, it must first advise the CAE by a call on NOTIFY STATE.

[At Release 1, direct access to the Kernel screen interfaces is permitted only after flushing the CAE screen buffer, or when display is suspended (provided that the action causes no change to the display), or immediately upon receiving a 'refresh' status. QUEUED actions are not permitted. Specification of an event number (other than -1) is not permitted. OPEN CHANNEL and CLOSE CHANNEL are not permitted. Subject to these rules, direct screen calls can be made without declaring direct usage to NOTIFY STATE; indeed, such a declaration will be rejected.]

The properties of the screen channel are the default properties supplied by Kernel (see [2]) for an application screen channel, modified as implied by calls on NOTIFY STATE to change the channel display mode. The TEP can change the channel properties by sending control sequences or making calls on Kernel if necessary. It is not intended that TEPs should use other than the default window size.

The Include file CASCVALS defines symbolic names corresponding to the values recognised by the OPD screen driver as control codes and other special codes.

6.1.3 SCREEN REFRESH

The user may elect to remove the screen (and keyboard) temporarily from the Computer Access application to do other work, leaving the computer service session still connected. This occurrence is reported by a status response from READ MODEM (and certain other procedures).

Unless the TEP is specified to take other action, it will continue to operate normally while this display-suspended state persists, except that it must not operate on the screen by direct use of Kernel interfaces. (It can continue to request screen transfers via the CAE write procedures: the CAE will ignore such requests when they are inappropriate).

If the Receive Process has declared that it performs queued transfers to the screen itself, it must cancel any outstanding transfers and then advise the CAE by a call on NOTIFY STATE. Any screen transfer arising out of calls on the CAE write procedures will have been cancelled; an undefined number of the bytes buffered since the last flush of the buffer (if any) will be

discarded.

Usually the remote service will quickly cease to send data during a display-suspended state, because it will be receiving no data from the Transmit Process while the keyboard is disconnected. Some TEPs may choose to use flow control to inhibit the remote service during the display suspension (although if the user is also monitoring to page store or printer, this might be seen as a disadvantage).

When the user restores the screen and keyboard to the Computer Access application, a 'refresh display' status response is returned by READ MODEM. The Receive Process is then required to restore the screen display to the state it would have had at that instant had no display-suspended state intervened. It must therefore maintain at all times the data necessary to effect this re-display. When the display has been refreshed, the Receive Process reverts to its normal mode of working.

The suspend-refresh sequence will also occur if the user invokes certain CAE functions during a live service session (for example, if he requests a Help screen or chooses to view stored pages). The Receive/Transmit Processes do not need to know why they are being suspended.

When the 'refresh display' status occurs, the screen is blank (black). The screen mode is as most recently requested by a call on NOTIFY STATE; if no such request has been made, it is in 80-column mode. The supplied screen channel is in the corresponding mode (80 or 40 column). The display attributes of the screen channel are as set by Kernel on a call of SET CHANNEL DISPLAY MODES specifying the relevant screen mode.

When the Receive Process is initially entered, the display state is 'suspended'. The Process should at once acquire store space and other resources it requires, and may then enter its normal processing loop, but not perform direct screen transfers until it receives a 'refresh display' status.

6.1.4 SNAPSHOT

The user may request that a 'snapshot' of the current display be saved in the Page Store for subsequent display or printing, or sent at once to the printer. (A TEP can elect not to support one or both of these features by setting flags in the TEP Definition Block (see section 5.2), if support is impracticable.)

Snapshot to the printer is provided by Computer Access in addition to the pixel-based printing offered by the special OPD PRINT key because character-based printing is likely to be significantly faster and can usually be performed asynchronously from the live service dialogue. The user will often choose to use this character-based snapshot even though some graphic symbols cannot be reproduced on the printer in character mode.

Snapshot (or monitor) to printer is handled by writing a temporary page in the page store, which is then printed by the Print Process. As far as the Receive Process is concerned, all snapshots and monitor files go to Page Store.

When a snapshot is required, the Receive Process is notified by a status response from one of the CAE procedures.

The process of sending the snapshot data is similar to that of refreshing the screen. The standard PUT BYTE and PUT STRING procedures are used, with a parameter indicating the required destination of the data.

The sending of the snapshot data is preceded by a call on BEGIN PAGE, and followed by a call on END PAGE. The BEGIN PAGE call passes the Page Properties to the CAE (see section 5.4.2.).

The TEP may get status change responses while sending the snapshot data. Unless it gets 'terminate service', it must still finish the snapshot, as well as responding to the status change.

If a snapshot or monitor file is being sent direct to the printer (via a temporary page in the page store) and the page store becomes full, the CAE will normally wait within the PUT BYTE or PUT STRING procedure until space becomes available (as data is transferred to the printer), provided that the state of flow control is such that the receipt of further data from the remote service can be inhibited. If, however, a status change then occurs, it must be notified and processed at once, with the transfer to pagestore incomplete. Some simpler TEPs may find this unduly complicated to handle, and an option is therefore provided in PUT BYTE and PUT STRING for this condition to be treated as an error, leading to premature termination of the snapshot or monitor file. If the simpler option is chosen, the PUT BYTE or PUT STRING appears to terminate successfully, and the outstanding status will be notified on a subsequent call of READ MODEM; if the more complex option is chosen, the status is returned by PUT BYTE or PUT STRING (in the latter case, with details of the incomplete part of the transfer), and the transfer must be completed after processing the status response.

6.1.5 MONITOR

If the user requests monitoring of the display, the whole of the displayed output is to be copied to a page in the page store from now until monitoring is switched off. The data already displayed when monitoring is requested is not copied.

When monitoring is to start or stop, the Receive Process is notified by a status response from one of the CAE procedures.

The sending of the monitor file data is preceded by a call on BEGIN PAGE, and followed by a call on END PAGE, in the same way as for a snapshot.

The TEP must take care not to copy to the monitor file when it is merely refreshing the screen.

6.2 Call on TEP

The Receive Process of the TEP is invoked by a subroutine call on the Program Entry Point of the TEP, with DO.L set to 1.

6.3 Calls from TEP

The following CAE procedures can be called by the Receive Process of the TEP.

6.3.1 READ MODEM

Action Value (DO.B): CA.READMODEM

Additional Call Parameters

D2.L: 0: do not return until a change of status occurs or a byte is available from the modem.

1: return at once if no byte is available from the modem

n(>1): as for value 0, but if no immediate return is made, the segment whose identifier is n is thawed. When return is eventually made, the segment is re-frozen, and its address is returned in A4. If no thaw/freeze occurs, A4 is unchanged.

D3.W: Length of Character Translation Table

A3 : Address of Character Translation Table

Each character read from the modem is translated using the specified table. No translation is performed if D3.W is zero. Details of translation tables are given in section 5.5.

Return Parameters

DO.L: returns a response/status:

zero indicates that a byte has successfully been read

a negative value indicates a failure (see Error Returns)

a positive non-zero value indicates a status change, as follows:

2 suspend display

4 refresh display

6 terminate service

8 take snapshot

10 interrupt received from Transmit Process

12 unexpected event occurred (see D2)
[This status cannot occur at Release 1]

14 begin monitoring

16 end monitoring

D1.B: (translated) byte read from modem. The most significant three bytes of D1.L are zero. The value is defined only if D0.L is zero.

D2.L: image of Event Notification Register with bits set for events that have occurred that are not handled by the CAE. The value is defined only if D0.L is 12.
(D2 is used for an additional purpose in the READ PAGE version of this procedure. See section 8.3.1 for details.)

A4 : if D2.L identified a segment on entry, and it has been thawed and refrozen, A4 contains its new address. Otherwise A4 is unchanged.

Error Returns

ERR.NB no modem byte available and no status change outstanding

Description

If D2.L is zero, the procedure waits until a change of status occurs or a byte is available from the modem, when it returns with the values stated above. A change of status is reported in preference to returning a modem byte; in this case the modem byte will be returned on the next call (if no further status change has occurred). If D2.L is 1, the procedure returns at once with a status change or modem byte; failing these it returns ERR.NB. This option is provided in case the TEP can release resources (e.g. thaw segments) or perform other special actions if bytes have stopped arriving. D2.L can also identify a segment which is to be thawed until the procedure eventually returns. The TEP must not needlessly loop with D2.L set to 1.

6.3.2 PUT BYTE

Action Value (D0.B): CA.PUTBYTE

D1.B: byte to be put. The value in the rest of D1.L is not significant

D2.L: destination of byte:

bit 0	: set if byte is to be put to the screen
bit 1	: set if byte is to be put to the printer (this value is relevant only when the procedure is used in the Print Process)
bit 2	: set if byte is to be put to the Page Store
bit 3 to 6	: zero
bit 7	: 0 the procedure is not to report a status change 1 the procedure may report a status change
bits 8 to 31	: zero

D3.W: length of Character Translation Table (if any)

A3 : address of Character Translation Table (if any)

Return parameters

D0.L: returns a response/status:

zero indicates that the byte has successfully been accepted into the CAE's buffers

a negative value indicates a failure (see Error Returns)

a positive non-zero value indicates a status change. The status values are as defined in section 6.3.1. A status change is reported only in the circumstances described in section 6.1.4, and only if bit 7 of D2.L was set on entry.

Error Returns

ERR.OR - 'out of range' error from screen driver

[At Release 1, ERR.OR is not returned. If it occurs, the process is abandoned.]

(Note that error OP (out of parameters) does not occur, since CAE assumes that a subsequent PUT BYTE or PUT STRING will supply the necessary bytes. Screen driver errors TX and SR will eventually be handled by forcing a suspend-refresh cycle on the TEP; at Release 1, error TX is ignored and error SR causes the process to be abandoned.)

Description

The procedure accepts the supplied byte into the CAE's internal buffers, first translating it according the specified translation table (if any). The byte will subsequently be written to the

specified destination(s). The procedure returns at once unless the CAE's internal buffering is full; if the buffers are full, the procedure waits until buffer space becomes available, meanwhile servicing the data channels under its control. A status change may be reported, but only in the circumstances defined under return parameter DO.L. An attempt to put a byte to a destination that is irrelevant in the prevailing context is ignored and appears to succeed (for example, putting a byte to the screen when display is suspended, or to the page store when no snapshot or monitor is active).

6.3.3 PUT STRING

Action Value (DO.B): CA.PUTSTRING

Additional Call Parameters

D1.W : length of string to be put

A1 : address of string

D2.L):
D3.W): as for PUT BYTE (see section 6.3.2)
A3):

Return Parameters

DO.L: returns a response/status:

zero indicates that the whole string has successfully been accepted into the CAE's buffers

a negative value indicates a failure (see Error Returns)

a positive non-zero value indicates a status change. This can occur only in the same cases as for PUT BYTE (see section 6.3.2). Part of the string may have been accepted into the Page Store (see return parameter D1). The whole of the string will have been accepted into the buffers for other relevant specified devices.

D1.W: length of that part of supplied string that has not yet been accepted into the Page store. Zero indicates that the whole string has been accepted

A1 : address of that part (if any) of supplied string that has not yet been accepted.

Error returns

As for PUT BYTE (see section 6.3.2)

Description

This procedure has the same effect as PUT BYTE, except that a string of bytes is put instead of a single byte. Character translation (if specified) is applied to each byte in the string. A Status change may be reported, but only in the circumstances defined under return parameter D0.L.

6.3.4 CONTROL FLOW

Action Value (D0.B): CA.FLOW

Additional Call Parameters

D1.W: bit 0: 1 enable automatic flow control
0 disable automatic flow control

bit 1: 1 send an XOFF character

bit 2: 1 send an XON character

bits 1 and 2 must not both be set

Return Parameters

None

Error Returns

None

Description

This procedure has the effect defined in section 6.1.1.

6.3.5 TURN LINE ROUND

Action Value (D0.B): CA.TLR

Additional Call Parameters

None

Return Parameters

None

Error Returns

ERR.NI [Procedure not implemented at Release 1]

Description

[This procedure has the same effect as the Kernel procedure

K.MODEMTLR. It may need to be re-expressed in CAE terminology. True half duplex working is not supported at Release 1, and this procedure is not implemented.]

6.3.6 INTERRUPT TRANSMIT PROCESS

Action Value (DO.B): CA.INTTP

Additional Call Parameters

None

Return Parameters

None

Error Returns

None

Description

This procedure causes the status 'interrupted by Receive Process' to occur in the Transmit Process (notified by a status response from READ KEY). The location CA.RTOT in the CACB is provided for the transmission of data associated with the interrupt.

6.3.7 BEGIN PAGE

Action Value (DO.): CA.PAGEBEGIN

Additional Call Parameters

A0 : address of Page Properties (see section 5.4.2)

Return Parameters

None

Error returns

ERR.NO : no page activity has been requested

Description

This procedure is called when the TEP has been requested to write a snapshot or monitor file to pagestore. The TEP supplies the properties of the page, in the format defined in section 5.4.2. Until this call is made, the CAE ignores any attempt to put bytes to the pagestore.

6.3.8 END PAGE

Action Value (DO.B): CA.PAGEEND

Additional Call Parameters

None

Return Parameters

None

Error Returns

None

Description

This procedure is called when the TEP has finished writing a snapshot or monitor file to pagestore.

6.3.9 NOTIFY STATE

Action Value (DO.B): CA. STATE

Additional Call Parameters

D1.W : Value indicates state being notified.
Values are given below as symbolic names;
the values corresponding to these names
are given in Include file CASTVALS.

CA.FAIL	process has failed during initialisation or normal execution. D2 defines reason for failure
CA.DONE	process has finished and tidied up
CA.SUSP	process has completed all screen transfers, following a 'suspend display' request. This state should be notified only by a process that has elected to drive the screen directly. [Not permitted at Release 1: will cause error response ERR.NI.]
CA.OWNSCR	process will drive screen by direct use of Kernel interfaces, rather than through the CAE screen interfaces. [Not permitted at Release 1 (see section 6.1.2): will cause error response ERR.NI.]
CA.40COL	process requires 40-column mode. The screen hardware and screen channel will be set to this mode now and when any 'refresh display' status subsequently occurs
CA.80COL	process requires 80-column mode. The screen hardware and screen channel will be set to

this mode now and when any 'refresh display' status subsequently occurs. 80-column mode is set by default on entry to the TEP.

Setting the mode has the effect of re-initialising the screen channel (by a call on the Kernel procedure SET CHANNEL DISPLAY MODES). If the display is not suspended, the screen is cleared to black

D2.L : if D1.W is set to CA.FAIL, D2.L defines the cause of the failure as follows:

negative values:

-1 to -79: the value is assumed to be one of the standard OPD responses ERR.cc

-80 to -89: these values may be documented by the TEP as representing particular TEP-specific failures

-90 to -99: reserved for errors detected by the CAE

The CAE will display a message to the user including the error code. In the case of ERR.OM and other errors attributable to insufficient store, the error code is replaced by a NO STORE message.

positive value:

length of a string (1 to 20 characters) describing failure in user terms. [This feature is not supported at Release 1. Any positive value will be treated as -99.]

A2 : address of failure message (if D1.W=CA.FAIL and D2.L is positive)

Return Parameters

None

Error Returns

ERR.BP: Bad parameter

Description

This procedure notifies the specified TEP state to the CAE. No return is made when the state CA.FAIL or CA.DONE is notified; in these cases the TEP should have tidied up and released any resources that it has acquired (although it is not essential to release resources that will be implicitly released by Kernel when CAE destroys the calling Process's activity.)

6.3.10 FLUSH BUFFER

Action Value (DO.B): CA. FLUSH

Additional Call Parameters

D2.L: buffers to be flushed

bit 0: set if screen buffer is to be flushed (applicable to Receive and Display Processes only

bit 1: set if printer buffer is to be flushed (applicable to Print Process only)

bit 3: set if modem output buffer is to be flushed (applicable to Transmit Process only) [This bit is ignored at Release 1.]

Return Parameters

None

Error Returns

ERR.OR 'out of range' error from screen driver [At Release 1, ERR.OR is not returned. If it occurs, the process is abandoned.]

ERR.OP 'out of parameters' error from screen driver [At Release 1, it is invalid to attempt to flush the screen buffer when a partial control sequence has been sent. The effect of doing so is unpredictable and may lead to a reset of the OPD.]

Description

This procedure waits until all data in the specified CAE buffers has been output to the corresponding devices. An attempt to flush an inactive device is ignored.

7. TRANSMIT PROCESS

7.1 Description

The Transmit Process is the process whose primary task is to receive data from the OPD keyboard and transmit it via the modem link to the live computer service.

7.1.1 KEYBOARD ACCESS

Data from the keyboard is read a byte at a time using the READ KEY procedure.

Key depressions are interpreted as follows:

1. Certain function keys are dedicated to standard CAE control functions which are always supported, irrespective of the TEP in use.

2. Certain function keys are reserved for common control functions which may not be supported by all TEPs. Each TEP declares which of these standard functions it supports by means of a field in the TEP Definition Block (see section 5.2). To achieve functional consistency between TEPs, these keys should not be used for materially different purposes. Standard function keys are intercepted by the CAE, and are not seen by the TEP (although they may give rise to a status response).

3. Certain function keys may be declared to represent TEP-specific control functions (e.g. Viewdata REVEAL). The TEP uses the TEP Definition Block to supply the CAE with text explaining the function for inclusion on the Functions screen. TEP-specific function keys are treated as normal keys, and are translated and processed as described below. Unsupported function keys are rejected by the CAE and do not reach the TEP.

4. ALT/A to ALT/P may be declared by the user to represent stored key sequences. Depressing such a key is equivalent to typing in the sequence of characters stored in the profile for that key. The TEP is not aware of these sequences; they are expanded by the CAE and each key is processed as if it had been typed directly. If there is no corresponding sequence in the profile, the key is rejected by the CAE; it does not reach the TEP. ALT/Q to ALT/Z are not treated specially.

5. Keys are (optionally) translated using a standard Computer Access translation table.

   If the current service profile contains an 'ALT/X' string, the CACB location CA.SENDK is set to the value of the fifth character of the string less $40. If there is no such

string, CA.SENDK is set to $0D. For a TEP that supports the feature, the conventional interpretation of CA.SENDK is that it contains the value to be sent when the user types RETURN. ALT/RETURN should normally then be set to generate $0D (the normal RETURN value). Translation of RETURN and ALT/RETURN is the responsibility of the TEP.

6. The (translated) key values are then

   (a) returned to the Transmit Process by READ KEY, or

   (b) transmitted straight to the modem without the involvement of the Transmit Process, or

   (c) rejected within the READ KEY procedure, with a standard 'bad key' tone, or any combination (or none) of the above.

   The choice of action is made by the TEP for each of the following (translated) key groups:

   $00 to $1F (normally standard CTRL keys)
   $20 to $7F (normally standard OPD graphics)
   $80 to $9F (normally cursor control keys)
   $A0 to $FF (normally function and other special keys)

   If the modem is set up for 7-bit data, there is no option to transmit keys in the range $80 to $FF to the modem.

7.1.2 MODEM ACCESS

The Transmit Process can send bytes explicitly to the modem channel by means of the PUT BYTE TO MODEM procedure. There is no facility to put a string of bytes in a single call. Such a byte is not sent until after any bytes already queued by the implicit transmission facilities of READ KEY.

Flow control bytes (XON and XOFF) generated by the CAE's automatic flow control facility, or by use of the CONTROL FLOW procedure, are sent at once, and thus jump the queue of bytes awaiting transmission. If this is not what is required (for example, if there are context-dependent sequences of bytes), the Receive and Transmit Processes must handle the flow control bytes explicitly.

7.2 Call on TEP

The Transmit Process of the TEP is invoked by a subroutine call on the Program Entry Point of the TEP, with D0.L set to 2.

7.3 Calls from TEP

The following CAE procedures can be called by the Transmit

Process of the TEP.

7.3.1 READ KEY

Action Value (DO.B): CA.READKEY

Additional Call Parameters

D2.L :settinga bit to 1 has the following meaning:

bit 0: return key values $00 to $1F to Transmit Process

bit 1: return key values $20 to $7F to Transmit Process

bit 2: return key values $80 to $9F to Transmit Process

bit 3: return key values $A0 to $FF to Transmit Process

bits 4 to 7: transmit keys to modem

bits 8 to 11: reject keys

D3.W :length of Character Translation Table

A3 :address of Character Translation Table

Return Parameters

DO.L : returns a response/status:

: zero indicates that a byte has successfully been read

: a negative value indicates a failure (see Error Returns)

: a positive non-zero value indicates a status change, as follows:

6 terminate service
10 interrupt received from Receive Process
12 unexpected event occurred (see D2)
[This status cannot occur at Release 1]

D1.B : (translated) byte read from keyboard. The most significant three bytes of D1.L are zero. The value is defined only if DO.L is zero.

D2.L : image of Event Notification Register with bits set for events that have occurred that are not handled by the CAE. The value is defined only if DO.L is 12.

Error Returns

None

Description

The procedure waits until a change of status occurs or a byte is available from the keyboard with a value for which return has been requested. A change of status is reported in preference to returning a keyboard byte; in this case the keyboard byte will be returned on the next call (if no further status change has occurred). While waiting, the procedure services the data channels under its control.

7.3.2 PUT BYTE TO MODEM

Action Value (DO.B): CA.PUTMBYTE

Additional Call Parameters

D1.B : byte to be put. The value in the rest of D1.L is not significant

D3.W : length of Character Translation Table (if any)

A3 : address of Character Translation Table (if any)

Return Parameters

None

Error Returns

None

Description

The procedure accepts the supplied byte into the CAE's internal buffer, first translating it according to the specified translation table (if any). The byte will subsequently be written to the modem. The procedure returns at once unless the CAE's buffer is full; if the buffer is full, the procedure waits until buffer space becomes available, meanwhile servicing the data channels under its control.

7.3.3 CONTROL FLOW

See section 6.3.4.

7.3.4 TURN LINE ROUND

See section 6.3.5.

7.3.5 INTERRUPT RECEIVE PROCESS

Action Value (DO.B): CA.INTRP

Additional Call Parameters

None

Return Parameters

None

Error Returns

None

Description

This procedure causes the status 'interrupted by Transmit Process' to occur in the Receive Process (notified by a status response). The location CA.TTOR in the CACB is provided for the transmission of data associated with the interrupt.

7.3.6 NOTIFY STATE

See section 6.3.9.

7.3.7 NOMINATE BREAK KEY

Action Value (DO.B): CA.NOMBRK

Additional Call Parameters

D2.B : character code of key that is to cause Break signal. This can be any character that the TEP might otherwise receive, but must not usurp one of the standard OPD function keys (f1, f6, f7, f8, f9) or the keys used to identify stored key sequences (ALT/A to ALT/P). The usual key for this purpose is CTRL/space ($C8).

The value $FF cancels the nomination

Return Parameters

None

Error returns

None

Description

This procedure nominates a key which, if typed when the keyboard is connected to the live service, will cause a 250 ms break condition to be transmitted, jumping the queue of any bytes waiting to be transmitted. The specified character code is not subject to any character translation, and will not be returned by READ KEY.

7.3.8 SEND BREAK

Action Value (DO.B): CA.BREAK

Additional Call Parameters

None

Return Parameters

None

Error Returns

None

Description

This procedure causes a 250 ms break condition to be transmitted, jumping the queue of any bytes waiting to be transmitted.

8. DISPLAY AND PRINT PROCESSES

8.1 Description

8.1.1 ARCHITECTURE

The Display and Print Processes are used when the user requires display or printing respectively of a page from the Page Store. If the properties of the stored page are such that only the CAE can display or print the page the TEP is not involved. Otherwise the page is presented to the Process a byte at a time, and the TEP causes the necessary output to the screen or printer.

It will be seen that these Processes are markedly similar to the Receive Process, and there may well be some common code in these processes in the TEP.

Only one Display Process exists at any instant, and only one Print Process. Display and Print Processes (of the same or different TEPs) may exist at the same time, operating on the same or different pages from the Page Store. For example, a page may be displayed while printing of that or another page is still taking place. Display and/or Print Processes may exist at the same time as Receive and Transmit Processes of the same or another TEP. Each Process executes in a different OPD activity (with a distinct stack). Code shared between potentially concurrent processes must observe appropriate conventions with respect to re-entrancy and data usage.

To facilitate the writing of common process code, the CAE procedures in general ignore (rather than reject) an attempt by a particular process to perform a transfer inappropriate to that process.

8.1.2 SCROLLING

When a monitor file is displayed from the page store, scrolling of the display is controlled by the user so that he can inspect any part at leisure.

This does not apply to snapshots, which by definition occupy only a single display screen. (Some terminal emulations may need to regard snapshots as applying to a conceptual screen, of which only a 24-line window is actually visible on the OPD at any time. The handling of such special cases is the responsibility of the TEP.)

Scrolling of a line mode stored page can be either a line at a time or a page (24 lines) at a time, with a potential page boundary occurring at each line boundary. Scrolling of a page mode stored page is a page at a time, whether the user requests line scroll or page scroll. The precise interpretation of line scroll and page scroll must be determined by the TEP designer taking into account the user's probable expectations of these

functions. (Because of a terminological conflict, the reader must take care to determine from the context whether 'page' refers to a 24-line display or to one of the logical entities in the page store.)

Scrolling can be forwards (downwards) or backwards (upwards). Only forward scrolling is supported by the CAE at Release 1, but TEP designers are advised to define their data structures in a way that will easily admit backward scrolling.

The Page Properties (see section 5.4.2) define whether scrolling is to be handled by the TEP or the CAE. If the TEP is handling scrolling, it is notified by a 'function key' status when scrolling has been requested, and it is responsible for ensuring that scrolling does not otherwise take place. If the CAE is handling scrolling, the CAE uses the line/page separator characters to find the appropriate data in the page store, sets the cursor position as required, and either displays the data itself or gives it to the TEP display.

8.1.3 PRINTING

Computer Access printing is performed using Printer Manager (see [4]).

Before passing control to the Print Process of the TEP, the CAE will have opened a printer channel through Printer Manager and stored the channel identifier in location CA.PPRT in the CACB.

The TEP sends data to the printer by calling PUT BYTE and/or PUT STRING. The bytes are translated according to the specified translation table (if any), buffered, and eventually sent to Print Manager in non-image print mode. The interpretation of the bytes depends on Printer Manager (see [4]) and on the printer configuration data.

[The CAE may eventually provide special processing of certain characters (for example, Backspace), but not at Release 1.]

At Release 1, the TEP is not permitted to call Printer Manager direct, nor the Kernel interfaces to the printer.

When printing is complete, CAE closes down the printer.

8.2 Calls on TEP

The Display Process of the TEP is invoked by a subroutine call on the Program Entry Point of the TEP, with D0.L set to 3; the Print Process has D0.L set to 4.

In both cases, A0 points to a copy of the Page Properties (see section 5.4.2), which remain at this address throughout the life of the process.

8.3 Calls from TEP

The following CAE procedures can be called by the Display and Print Processes of the TEP.

8.3.1 READ PAGE

Action Value (DO.B): CA.READPAGE

This action value is identical to CA.READMODEM, enabling common code to be used for the input of data to the Display, Print and Receive Processes.

Additional Call Parameters

As for READ MODEM (see section 6.3.1)

Return Parameters

As for READ MODEM (see section 6.3.1), except as follows:

The only status values that may be returned are:

2	suspend display (Display Process only)
4	refresh display (Display Process only)
6	terminate process
12	unexpected event occured (see D2) [This status cannot occur at Release 1.]
18	function key (see D2) (Display Process only)

The return parameter in D2.L is defined in the same way as for READ MODEM in the case where status 12 is returned. If a byte is successfully returned (DO.L zero), D2.L is set to the index of that byte within the Page Data. See the SET INDEX procedure for the purpose of this value. If status 18 (function key) is returned, D2.B contains the code for the function key ($F0 to $F9) or scroll key ($80 to $9F) used; the remainder of D2.L is zero.

Error Returns

ERR.NB no byte from page available, and no status outstanding. This response does not necessarily indicate that the end of the page has been reached; it may indicate a scrolling pause.

Description

This procedure has the same effect as READ MODEM (see section 6.3.1), except that the bytes are read from the page store rather than from the modem.

8.3.2 PUT BYTE

See section 6.3.2

8.3.3 PUT STRING

See section 6.3.3

8.3.4 NOTIFY STATE

See section 6.3.9

8.3.5 FLUSH BUFFER

See section 6.3.10

8.3.6 SET INDEX

Action Value (DO.B): CA.SETINDEX

Additional Call Parameters

D1.W : must be zero [additional features may be provided]
D2.L : index of byte within Page Data from which reading is to continue

Return Parameters

D2.L : undefined [additional features may be provided]

Error Returns

ERR.OR no such byte in page

Description

This procedure, which can be called only by the Display Process, causes subsequent calls of READ PAGE to return bytes from the Page Data starting with the byte with the specified index. Until this procedure is called, bytes are returned starting from the first byte of the Page Data (byte 0). Whenever READ PAGE returns a byte, it also returns its index. SET INDEX may be used by the Display Process to refresh the display without maintaining a copy of the currently displayed data, or to facilitate scrolling of the display.

9. PROFILE ACCESS

9.1 Description

The Profile Store is described in section 5.3.

9.2 Calls on TEP

The following calls can be made on the TEP by the CAE [although not at Release 1]. Each call is a subroutine call on the Program Entry Point of the TEP, with an Action Value in D0.L distinguishing the particular call. These procedures are constrained to use only [tbs] bytes on the stack (including the 128 bytes required for calls made by these TEP procedures on CAE procedures).

9.2.1 (VALIDATE PROFILE FIELD)

[This procedure is not called at Release 1]

Action value (D0.L): 5

Additional Call Parameters

A0 : address of CAE's buffer containing profile field

Return Parameters

D0.L: returns a response from the TEP

zero indicates that the field is valid

a positive value indicates a failure; the value in D0.L is the length of an error message explaining the failure in user terms (maximum length 40)

A2 : address of error message (when D0.L is non-zero)

All other registers except A7 (SP) are assumed corrupt.

Error Returns

See Return Parameters

Description

This procedure is called when the user supplies a new value for a TEP specific profile field whose Profile Definition Field (see section 5.2.3) requests validation by the TEP.

9.2.2 (VALIDATE PROFILE)

[This procedure is not called at Release 1]

Action value (DO.L): 6

Additional Call Parameters

None

Return Parameters

DO.L : returns a response from the TEP

zero indicates that the profile is valid.

A positive value indicates that the profile is invalid; the value in DO.L is the length of an error message explaining the error in user terms (maximum length 40)

A2 : address of error message (when DO.L is non-zero)

Error Returns

See Return Parameters

Description

This procedure is called when the user seeks to store a new or amended profile. The TEP should perform checks as necessary for consistency of values between fields. The CAE will already have performed any such checks that do not require TEP-specific knowledge. The TEP uses the procedures defined in section 9.3 to access the profile fields.

9.3 Calls from TEP

The following CAE procedures can be called by the Receive and Transmit Processes of the TEP.

They can also be called from within the TEP procedures described in section 9.2. This may be necessary, for example, where there is inter-dependence between field values in a profile.

9.3.1 GET PROFILE FIELD

Action Value (DO.B): CA.GETPROFILE

Additional Call Parameters

D1.W :specifies the required field:

zero : get first field

+n : get field with identifier n

-n : get existing field with lowest identifier greater than n

A0 : address of a buffer in frozen store to receive the field

D2.W :length of buffer

Return Parameters

None

Error Returns

ERR.NF : no such field

ERR.BO : field longer than buffer (field is truncated on right to buffer length)

Description

This procedure returns the specified field from the current profile. If the call is made by a Receive or Transmit Process, the profile controlling the live service is accessed. If the call is made from one of the procedures described in section 9.2, the profile being processed by that procedure is accessed. If the field is 'secret', the returned value is decrypted.

9.3.2 PUT PROFILE FIELD

[This procedure is not supported at Release 1]

Action Value (D0.B): CA.PUTPROFILE

Additional Call Parameters

A0 : address of a frozen data area containing the field to be put into the profile. (The length of the field is implicit in the field)

Return Parameters

None

Error Returns

ERR.BP Invalid profile field
ERR.OM No space in profile store
ERR.SL Data Record actions prevent write access

[At Release 1, the procedure always returns ERR.NI]

Description

This procedure inserts the supplied field into the current profile. The profile accessed is as defined in section 9.3.1. If a field with the same identifier already exists in the profile, it is replaced. If the field length (specified in the second byte of the field) is zero, the action of the procedure is instead to delete the field (if any) having the specified identifier. If the new field is marked as 'secret', it is encrypted by the procedure before it is stored. The CAE does not guarantee to perform more than minimal validation of the supplied field.