INPUT/OUTPUT PROGRAMS
SOFTWARE DESCRIPTION
1A PROCESSOR

CONTENTS PAGE

CONTENTS PAGE

NOTICE

Not for use or disclosure outside the
Bell System except under written agreement

Printed in U.S.A.

CONTENTS PAGE

Figures

Tables

This description is based on the common processor release 7 (CPR7) generic program in issue as of the date of this section, but is also applicable for those offices equipped with the CPR5 and CPR6 generic programs. Major differences between the generic programs are pointed out.

1. GENERAL

INTRODUCTION

1.01 This section describes the Input/Output (I/O) programs for the 1A Processor. The I/O programs provide the interface between operating personnel and the 1A Processor and associated equipment.

1.02 This section is being reissued to bring the description up to the CPR7 issue level. This includes a major revision of the Input/Output Control Program (IOCP) generic features. Revision arrows are used to emphasize the more significant changes.

1.03 A high-level description of the major I/O message processing functions performed by the I/O programs is provided.

1.04 Part 7 contains a defined list of abbreviations and acronyms used in this section.

PROGRAM IDENTIFICATION (PIDENT) DESCRIPTION

1.05 The following pidents are described in this section. The program listings (PRs) may be referred to for further information:

(a) IOCPIMR1—Input Message Translator Routines (PR-5A105)

(b) IOCPIMT1—Input Message Translator (PR-5A106)

(c) IOCPINT1—I/O Initialization (PR-5A108)

(d) IOCPIOC1—Input/Output Unit Controller (IOUC) Hardware Interface Subroutines (PR-5A117)

(e) IOCPIOH1—I/O Handler (PR-5A113)

(f) IOCPMSR1—Output Message Save and Retrieval (PR-5A115)

(g) IOCPOMC1—Output Message Data (PR-5A116)

(h) IOCPOMS1—Output Message Starter (PR-5A118)

(i) IOCPOMT1—Output Message Translator (PR-5A119)

(j) IOCPPCH1—Print Call Handler (PR-5A109)

(k) IOCPPDT1—Protocol Handler (PH) Tables (PR-5A131)

(l) IOCPPMD1—PH External Routines Called by IOCP (PR-5A136)

(m) IOCPPM11—PH Output Routines (PR-5A134)

(n) IOCPPM21—PH Input Routines (PR-5A133)

(o) IOCPPRC1—PH Recovery Routines (PR-5A132)

(p) IOCPPSC1—PH Scheduler (PR-5A135)

(q) IOCPREA1—I/O Channel Routing (PR-5A111)

(r) IOCPSUB1—I/O Hardware Interface Subroutines (PR-5A120)

(s) IOCPTIM1—I/O Timing (PR-5A112)

(t) IOCPUSR1—Client Service Routines (PR-5A110)

(u) IOCPIMC1—Input Messages Catalog (PR-5A103).

PURPOSE OF I/O PROGRAMS

1.06 In performing its functions, the I/O programs interface with several other programs. The major I/O program interfaces are shown in Fig. 1.

1.07 The I/O programs provide the interface between the hardware used by the 1A Processor to communicate with telephone company personnel or remote users and the client programs which generate or process information. Since the techniques used in the I/O programs are dependent on the type of hardware used, all interface with the hardware is contained in the I/O programs. Thus a change of hardware or addition of new hardware requires only a change or addition to the I/O programs; client programs will not require a change.

1.08 In addition to private line dedicated connections, the I/O programs also provide a dial-up capability that can be initiated by the remote users or the 1A Processor client programs. Any input/output processor (IOP) channel can be equipped with this capability. Security is also provided in this feature to prevent unauthorized remote connection. This security is achieved by accepting and acknowledging only one input message from a remote user before checking with the 1A Processor client program whether the temporary connection should be made permanent or dropped. Some applications automatically drop the temporary connection and, based on a prestored number associated with caller identification in the first message, dials the caller back.

> ***Note:*** Translation information for each IOP channel indicates whether the channel has a dial-up capability.

1.09 The high-speed I/O protocol feature allows the system to communicate with off-site clients at higher data rates than previously available. The 1A Processor high-speed I/O protocol function provides synchronous full-duplex I/O at rates up to 9,600 bits per second (bps) on private lines and up to 4,800 bps on dial-up connections. ➧For the CPR7 generic it provides rates up to 56,000 bps on private lines and 4,800 bps on dial-up connections.◀ These links use the Digital Data Communications Protocol (DDCMP) and the Bell Administrative Network Communications System (BANCS) Network Protocol. ➧For the CPR6 generic these links use the BX.25 level II protocol.◀

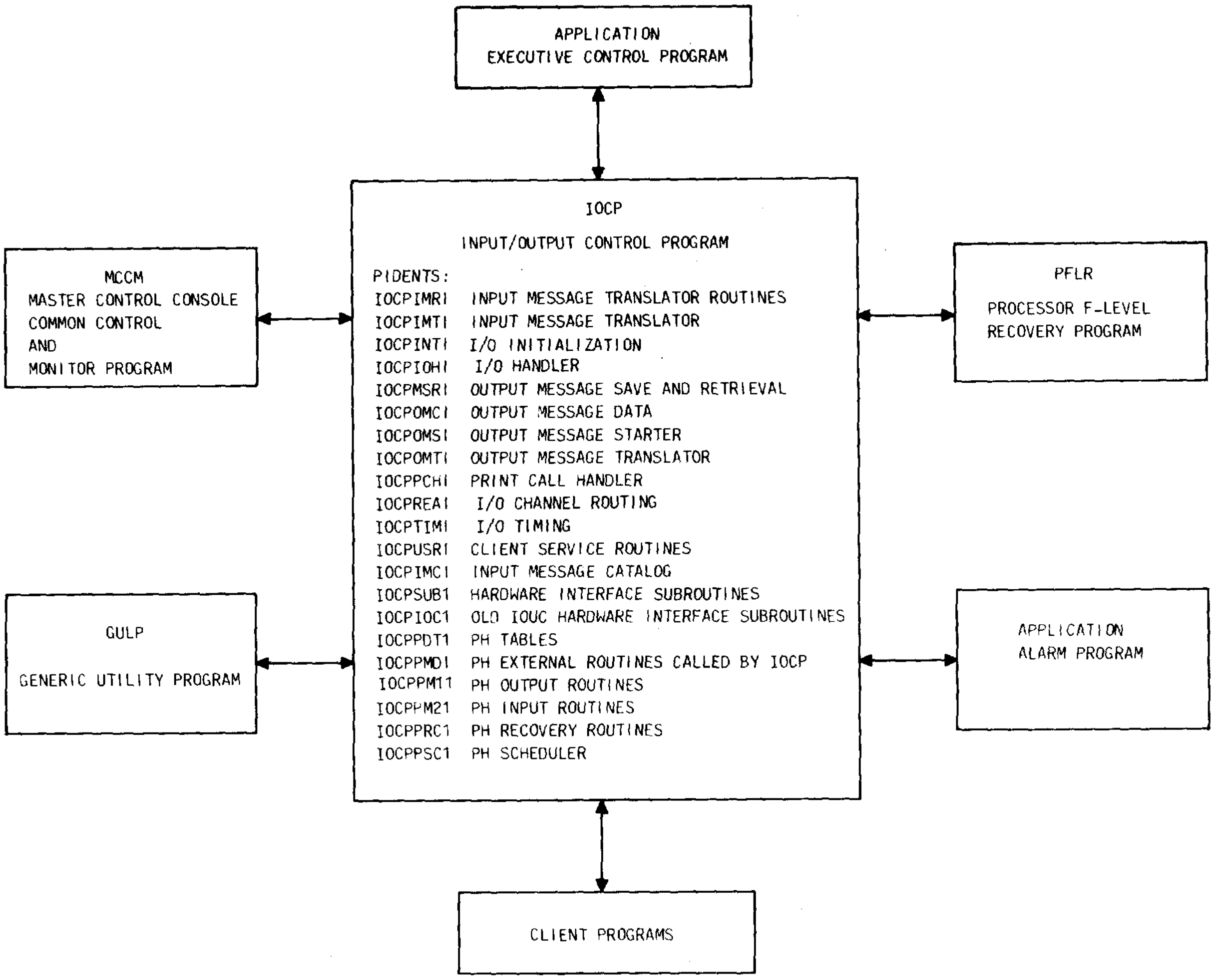
IOCP GENERIC FEATURES

1.10 The following information provides a brief description of the IOCP generics.

(a) ➧The CPR5 generic provides the following functions:

- Half-duplex and full-duplex operation
- Asynchronous operation
- Operation of speeds of 110, 300, 1200, 1800, 2400, 4800, and 9600 bps
- Dial-up capabilities
- Synchronous operation (DDCMP)
- Network protocol capability (ie, BANCS).

(b) The CPR6 generic add the following function



PIDENTS:

IOCPIMR1	INPUT MESSAGE TRANSLATOR ROUTINES
IOCPIMT1	INPUT MESSAGE TRANSLATOR
IOCPINT1	I/O INITIALIZATION
IOCPIOH1	I/O HANDLER
IOCPMSR1	OUTPUT MESSAGE SAVE AND RETRIEVAL
IOCPOMC1	OUTPUT MESSAGE DATA
IOCPOMS1	OUTPUT MESSAGE STARTER
IOCPOMT1	OUTPUT MESSAGE TRANSLATOR
IOCPPCH1	PRINT CALL HANDLER
IOCPREA1	I/O CHANNEL ROUTING
IOCPTIM1	I/O TIMING
IOCPUSR1	CLIENT SERVICE ROUTINES
IOCPIMC1	INPUT MESSAGE CATALOG
IOCPSUB1	HARDWARE INTERFACE SUBROUTINES
IOCPIOC1	OLD IOUC HARDWARE INTERFACE SUBROUTINES
IOCPPDT1	PH TABLES
IOCPPMD1	PH EXTERNAL ROUTINES CALLED BY IOCP
IOCPPM11	PH OUTPUT ROUTINES
IOCPPM21	PH INPUT ROUTINES
IOCPPRC1	PH RECOVERY ROUTINES
IOCPPSC1	PH SCHEDULER

Fig. 1—IOCP Major Program Interfaces

- Synchronous operation (BX.25 level II protocol).

(c) The CPR7 generic adds the following function:

- Operation at a high speed of 56,000 bps.♦

2. INPUT MESSAGE HANDLING (FIG. 2)

I/O HANDLER—INPUT

A. Normal Message Processing

2.01 The I/O handler (pident IOCPIOH1) is run on base level and is scheduled at least every 60 ms by the Application Executive Control Program at entry point IOCPIOH1. (Table A shows all the IOCP routines that interface with this program.) The I/O handler polls the hardware looking for a service request. Polling is done by sending a gated control pulse to all input/output unit selectors (IOUSs) via the peripheral control bus. Each IOUS has two dedicated IOUS member numbers. Each IOUS has two dedicated reply bus bit positions to send a reply. Bit positions are 2n and 2n+1 where n=IOUS number. Two bits are used to account for two halves of IOP (channels 0 through 7 and 8 through 15). A zero reply shows some IOUC in the IOUS has work. A read of the poll request register indicates which IOUC. If the poll request register is zero, it indicates a maintenance problem and control is passed to the Process F-Level Recovery Program (PFLR) for resolution. Otherwise, the reply from the IOUS indicates whether there is one or more IOUCs with a service request

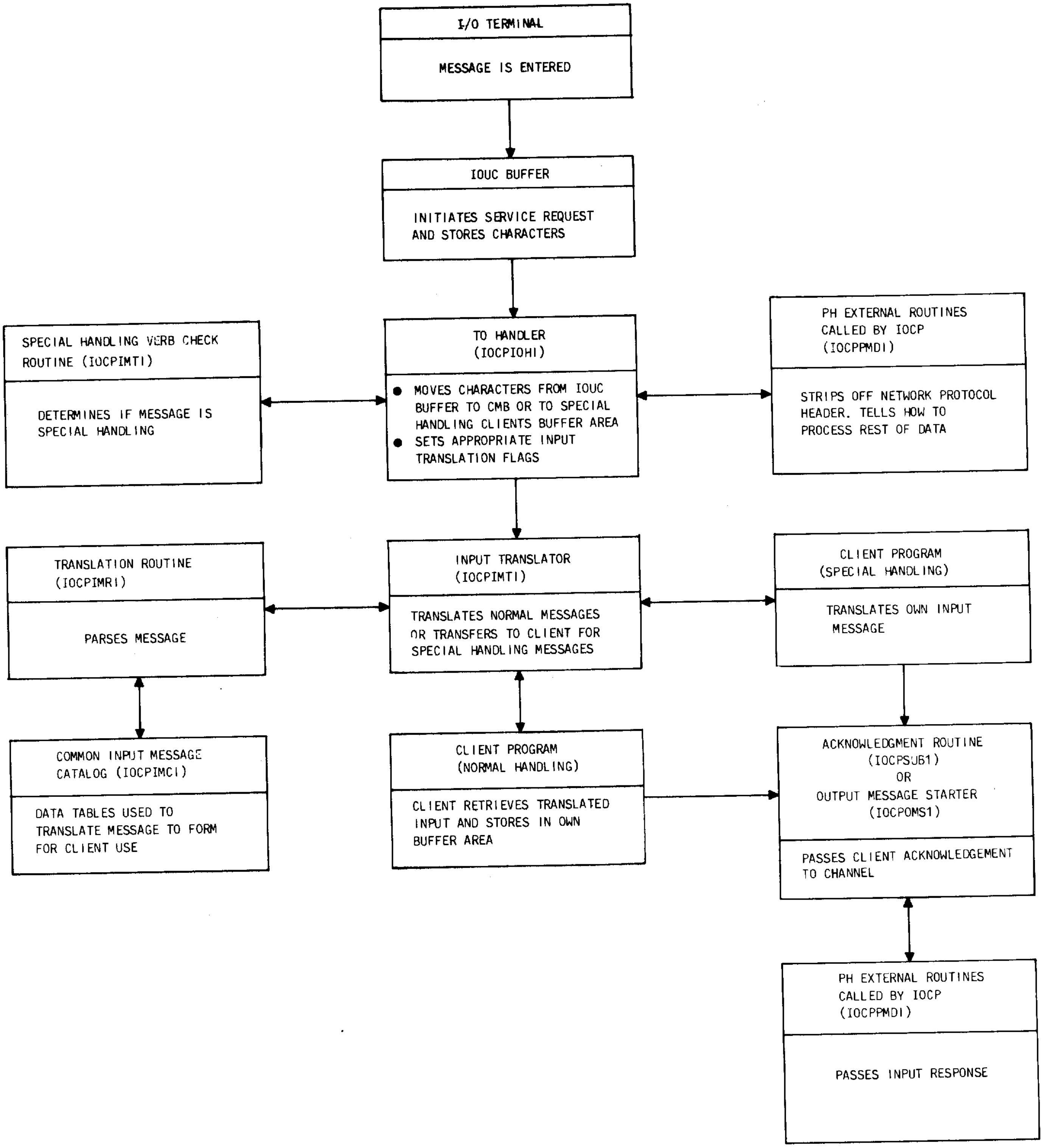

I/O TERMINAL
MESSAGE IS ENTERED
IOUC BUFFER
INITIATES SERVICE REQUEST AND STORES CHARACTERS
SPECIAL HANDLING VERB CHECK ROUTINE (IOCPIMTI)
DETERMINES IF MESSAGE IS SPECIAL HANDLING
TO HANDLER (IOCPIOHI)
● MOVES CHARACTERS FROM IOUC BUFFER TO CMB OR TO SPECIAL HANDLING CLIENTS BUFFER AREA
● SETS APPROPRIATE INPUT TRANSLATION FLAGS
PH EXTERNAL ROUTINES CALLED BY IOCP (IOCPPMDI)
STRIPS OFF NETWORK PROTOCOL HEADER. TELLS HOW TO PROCESS REST OF DATA
TRANSLATION ROUTINE (IOCPIMRI)
PARSES MESSAGE
INPUT TRANSLATOR (IOCPIMTI)
TRANSLATES NORMAL MESSAGES OR TRANSFERS TO CLIENT FOR SPECIAL HANDLING MESSAGES
CLIENT PROGRAM (SPECIAL HANDLING)
TRANSLATES OWN INPUT MESSAGE
COMMON INPUT MESSAGE CATALOG (IOCPIMCI)
DATA TABLES USED TO TRANSLATE MESSAGE TO FORM FOR CLIENT USE
CLIENT PROGRAM (NORMAL HANDLING)
CLIENT RETRIEVES TRANSLATED INPUT AND STORES IN OWN BUFFER AREA
ACKNOWLEDGMENT ROUTINE (IOCPSUB1) OR OUTPUT MESSAGE STARTER (IOCPOMS1)
PASSES CLIENT ACKNOWLEDGEMENT TO CHANNEL
PH EXTERNAL ROUTINES CALLED BY IOCP (IOCPPMDI)
PASSES INPUT RESPONSE

Fig. 2—Input Message Processing

and/or maintenance request in the request register waiting to be read. The I/O handler recognizes five types of service requests:

- Input (start of input) request
- Input (unload buffer) request
- Output (load buffer) request
- Idle request
- Break request.

2.02 Each IOP channel has associated with it a character buffer in the hardware. (The buffer is physically located in the I/O unit.) When an input message starts, the first character changes the hardware state from idle to input and brings up a service request. When the I/O handler detects a channel with a service request that has the hardware state marked as input and the software state marked as idle, it realizes that an input message is beginning. In half duplex, it then changes the software state from idle to input so that the input message will not be interrupted by output messages for that channel. If the channel is a dial-up channel, software status is set to show temporary connect. The I/O handler also puts the channel on 2-minute timing to ensure that the channel is not kept in the input state when there is no input being entered.

2.03 The I/O handler then checks to see if the input message is from a special handling channel. If it is, processing continues as described in paragraphs 2.10 and 2.11. If it is not, processing for normal messages continues.

> ***Note:*** An item in the translation information for each channel indicates whether a channel has special handling or not.

2.04 The characters continue to come into the hardware buffer until it reaches a threshold set in the hardware, or the end-of-input message character, at which time a service request is initiated to unload the buffer. The I/O handler recognizes the service request on a subsequent poll. Three characters at a time are read until the buffer is empty or until the end-of-input message character in the buffer is found. If this buffer did not contain the end-of-input data character, the I/O handler restarts the 2-minute timing.

2.05 The I/O handler stores the characters in a 32-word input character buffer (ICB) contained in the channel memory block (CMB). A CMB is a 44-word block of memory used to hold information about

TABLE A

IOCP INTERFACE WITH APPLICATION EXECUTIVE CONTROL PROGRAM

ROUTINE	PIDENT	ENTRY POINT	SCHEDULE FREQUENCY
I/O Handler	IOCPIOH1	IOCPIOH1	Every 50 to 60 ms
Input Translator	IOCPIMT1	IOCPITRC	Every base-level cycle
Output Translator	IOCPOMT1	IOCPXLAT	Every base-level cycle
Output Message Starter	IOCPOMS1	IOCPOMS1	Every base-level cycle if needed
I/O Timing	IOCPTIM1	IOCPTIM1	Every 5 seconds
		IOCPTDP1	Every 1 second
I/O Timing	IOCPIOH1	IOCPSTIM	Every 200 ms
PH Scheduler	IOCPPSC1	IOCPPSCD	This is a demand entry and is entered only when directed to by the ordered bits buffer bit

messages going to and coming from a channel. Each channel has a CMB. The 32-word ICB can accommodate input messages of up to 96 characters; however, input messages are limited to a single line on the input device.

2.06 Once the I/O handler has removed the first buffer of characters from the hardware buffer and stored them in the ICB, it calls a special handling verb check routine to see if this is a special handling verb. If this channel is administered by the PH, the PH analyzes the message and gives IOCP instructions to process the data (paragraph 2.12). If it is a special handling verb, processing continues (paragraph 2.09). If not, processing for normal messages continues. An application data table contains all the verbs that get special handling for that application.

2.07 On all later service requests to unload the hardware buffer, the I/O handler fetches the characters from the hardware buffer and stores them in the ICB until the hardware recognizes the end-of-input data character and brings up a service request. The I/O handler realizes that the hardware has recognized the end-of-input data character and stores the final characters in the ICB. It then sets an input translation request flag for this channel. There is an input translation request flag associated with each channel that is scanned by the input translator. The I/O handler puts the channel on 5-second timing to wait for an acknowledgment. The client program that handles the message is responsible for generating the acknowledgment.

2.08 On asynchronous channels, no input is expected while on 5-second timing until acknowledgment is generated because only one buffer is available to process input. Half-duplex channels are put in the output state at the end of input. If new input is received before acknowledgment, the channel goes to the break state and the previous input is discarded or just the acknowledgment is lost. On full-duplex channels, any input before acknowledgment is ignored.

B. Special Handling Processing by Verb

2.09 Some input messages get special handling by the I/O handler as indicated by an application data table. Once the handler has removed the first buffer of characters from the hardware buffer and stored them in the ICB, it calls a special handling verb check routine in the input translator (pident IOCPIMT1 at entry point IOCPIVSP). The special handling verb check routine searches the first characters of the input message looking for a verb. Once it finds the verb, it compares it to entries in the application special handling verb table. Each entry in this table has associated with it a special handling catalog that contains the client's buffer address and size (if characters are to be stored in the client's buffer) and the client's entry point. If the client has a buffer, the first buffer of characters is moved from the ICB to the client's buffer area. Then, on later service requests to unload the hardware buffer, the I/O handler reads the characters from the hardware buffer and stores them directly in the client's buffer area. It continues to collect characters and store them in the client's buffer area until the hardware recognizes the end-of-input data character and brings up a service request. The I/O handler then realizes that the end-of-input data has been recognized by the hardware and stores the final characters in the client's buffer area. It sets the input translation request flag and puts the channel on 5-second timing to wait for an acknowledgment. If there is no buffer associated with this verb, then, on later service requests, the I/O handler fetches the characters from the hardware buffer and stores them in the ICB. When the end-of-input data character is received and stored in the ICB, the I/O handler sets the input translation request flag, puts the channel on 5-second timing to wait for an acknowledgment, and expects no further input until acknowledgment.

C. Special Handling Processing by Channel

2.10 All input messages on designated channels require special handling by the I/O handler. These input messages are recognized from the channel type of the channel on which they are entered. Such a channel will have stored in translations an entry into a special handling catalog which contains the client's buffer address, size (if characters are to be stored in the client's buffer), and entry point address.

2.11 If the client has a buffer, the characters are put directly in the client's buffer area by the I/O handler as they are fetched from the hardware. If the client does not have a buffer, the characters are put in the ICB. On all service requests to unload the hardware buffer, the I/O handler reads the characters from the hardware buffer and stores them directly in the client's buffer area or the ICB. It continues to collect characters and store them in the proper buffer area until the hardware recognizes the end-of-input data character and brings up a service request. The I/O handler then realizes that the end-of-input data has been recognized by the hardware and stores the final characters in the buffer area. It sets the input translation request flag, puts the channel on 5-second timing to wait for an acknowledgment, and expects no further input until acknowledgment.

D. Network Protocol Input Processing

2.12 Input messages on designated channels require special handling by the PH. These input messages are recognized from the application identification of the channel on which they are entered. Such a channel will have stored in translations the application identification.

2.13 The PH analyzes the first bytes of data to identify the message protocol header. This information is used to update internal PH tables and states. After update, instructions are passed back to the main I/O handler (pident IOCPIOH1). These instructions may indicate a data message for a client, or a control message for the PH. The control message is to be discarded. If it is a client message, regular processing continues from that point on.

INPUT TRANSLATOR

2.14 The input translator (pident IOCPIMT1) is entered every base level cycle from the Application Executive Control Program at entry point IOCPITRC. It scans the input translation request flags looking for a channel that needs work. When the input translator finds the input translation request flag set for a normal message, it transfers to a translation subroutine in pident IOCPIMR1 which parses the message in the ICB.

2.15 Parsing involves the separation of an input message into its message fields (and subfields) and into the separate parameters comprising the fields. The input message contains a verb (the first field in the message), keywords, and data. The parser converts the string of American Standard Code for Information Interchange (ASCII) characters representing the message into a form more usable by client programs.

2.16 The parser first compares the verb to entries in the input message verb table. If it finds the verb in the table, it then compares the first keyword to entries in the input message keyword tables. The input message directory contains all valid combinations of verb and first keyword and serves as the link between the input message and the catalog entry which contains the formatting information used to translate the message. The input message catalog also contains the client's entry point address (similar to the tables for special handling messages). These tables are in pident IOCPIMC1. The input message is translated into a form more convenient to the client according to information in the input message catalog.

2.17 If the message is recognized by the input translator as a valid input message, the input translator transfers to the client. The client program is passed the address of the input message data area that contains a message and the channel number on which the message was entered. The input message data area contains flags indicating what keywords in the message were entered. The client program then has 3 ms to gather the data about the message, determine whether this is a valid message, and generate an acknowledgment. The client has up to 5 seconds to generate an acknowledgment.

2.18 To retrieve its translated keyword data, the client calls a subroutine (pident IOCPIMT1 at entry point IOCPIARG). This subroutine fetches the translated argument (one or more 24-bit words of data) and returns to the success return point. This retrieval process continues for each keyword until there are no more translated arguments, and the subroutine returns at the failure return point. After the client processes its message, it calls a subroutine (pident IOCPIMT1 at entry point IOCPIMCK) to determine if it has used all the data in its message. The subroutine checks to see that all input message data has been used and releases the message data area. The return to the client (success or fail) shows whether all the input message data had been used. After the client has collected its data, it returns to the input translator.

2.19 If the input translator cannot recognize the format of the message as a valid message, it has failed to translate the message. It then transfers to the Generic Utility Program (GULP) for a check to determine if the input message is a generic utility message. The GULP may return a success or fail to the input translator. If it returns failure, then the input translator transfers to an application parser.

(The application parser takes care of messages unique to an application and not handled by the common parser.) If the application parser returns a failure, the input translator generates an error acknowledgment from the data it saved when it originally detected the failure to parse the message.

2.20 If the application parser or GULP returns success, the input translator leaves the channel on 5-second timing to wait for an acknowledgment and allows the parser that recognized the input message to complete parsing of the message.

2.21 If the input translator finds an input translation request flag for a special handling input (this is indicated by an item in the CMB), it transfers directly to the client whose entry point is also stored in the CMB. It is then up to the client to analyze its message and return an acknowledgment for the IOCP to send to the channel. These channels are already on 5-second waiting for acknowledgment timing. Special handling input messages are handled quickly by the input translator, which does nothing but transfer to the client. The client uses the 3-ms time segment.

2.22 Once the client program (or either of the other two parsers) returns to the input translator, the input translator is free to look for other input messages to translate on its next entry. The input translator only handles one client per entry.

ACKNOWLEDGING THE INPUT MESSAGE

2.23 When the client is ready to generate an acknowledgment, it calls a subroutine in the common subroutine (pident IOCPSUB1). If the client has only a 2-character acknowledgment (ie, OK, PF, etc), it enters at entry point IOCPIACK. If it has an expanded acknowledgment (ie, two characters plus a 1-line text), the client enters at entry point IOCPIAKT. If the channel is administered by the PH, then the acknowledgment characters are passed to the I/O handler (paragraph 3.23) and the channel input state is idled to accept more input. Otherwise, if the channel is half duplex, the channel hardware is put in the output state and the acknowledgment characters sent. The software is put in the acknowledgment loaded state and 15-second timing is established. When the characters have been transmitted to the half-duplex channel, the channel goes back to the idle state. On the next poll, the I/O handler recognizes the idle state of the channel and changes the software state to idle. If the channel is full duplex, the channel input hardware is idled. If a terminal device is connected, then the input message and acknowledgment is scheduled as a separate output message. If the channel is a dial-up channel in a temporary connect state, a check is made with an application routine to determine if the temporary connection should be dropped or made permanent. The channel is ready to send output or receive input if the connection is made permanent. Otherwise, the 1A Processor client is expected to initiate a call back (paragraph 5.13).

3. OUTPUT MESSAGE HANDLING (Fig. 3)

PRINT CALL HANDLER

3.01 Output may be a result of an input message, a master control console key request, or automatically generated by the system. The source of output is transparent to IOCP.

3.02 To request that an output be sent to a channel, the client program calls the print call handler (pident IOCPPCH1) via the PRINT macro. If the client specifies the unit type and member number (ie, the output message concerns a piece of equipment) with the message, the client enters at entry point IOCPPRT1. If the client does not specify unit type and member number with the output message, it enters at entry point IOCPPRNT. In the PRINT macro call, the client specifies the catalog for the output message and the raw data to be output. The catalog may be specified by the catalog number for a generic catalog or by an address for a user catalog. The user-provided catalog allows a program to supply an output message catalog that is not defined as part of the generic catalog in pident IOCPOMC1. Both the generic and user catalogs may be either main memory or file store resident. The client may also specify the priority of the output message, and from one to five output channels or message classes. The print call handler has three returns to the client:

(a) Success: The output message is valid and buffered for output.

(b) Fail: The output message is valid but there are insufficient resources to buffer the message.

(c) Checksfail: The request is invalid or the output message register (OMR) link lists are invalid (the link list structure is broken). If the link lists are invalid, a call is made to an audits program to straighten out the software problem.

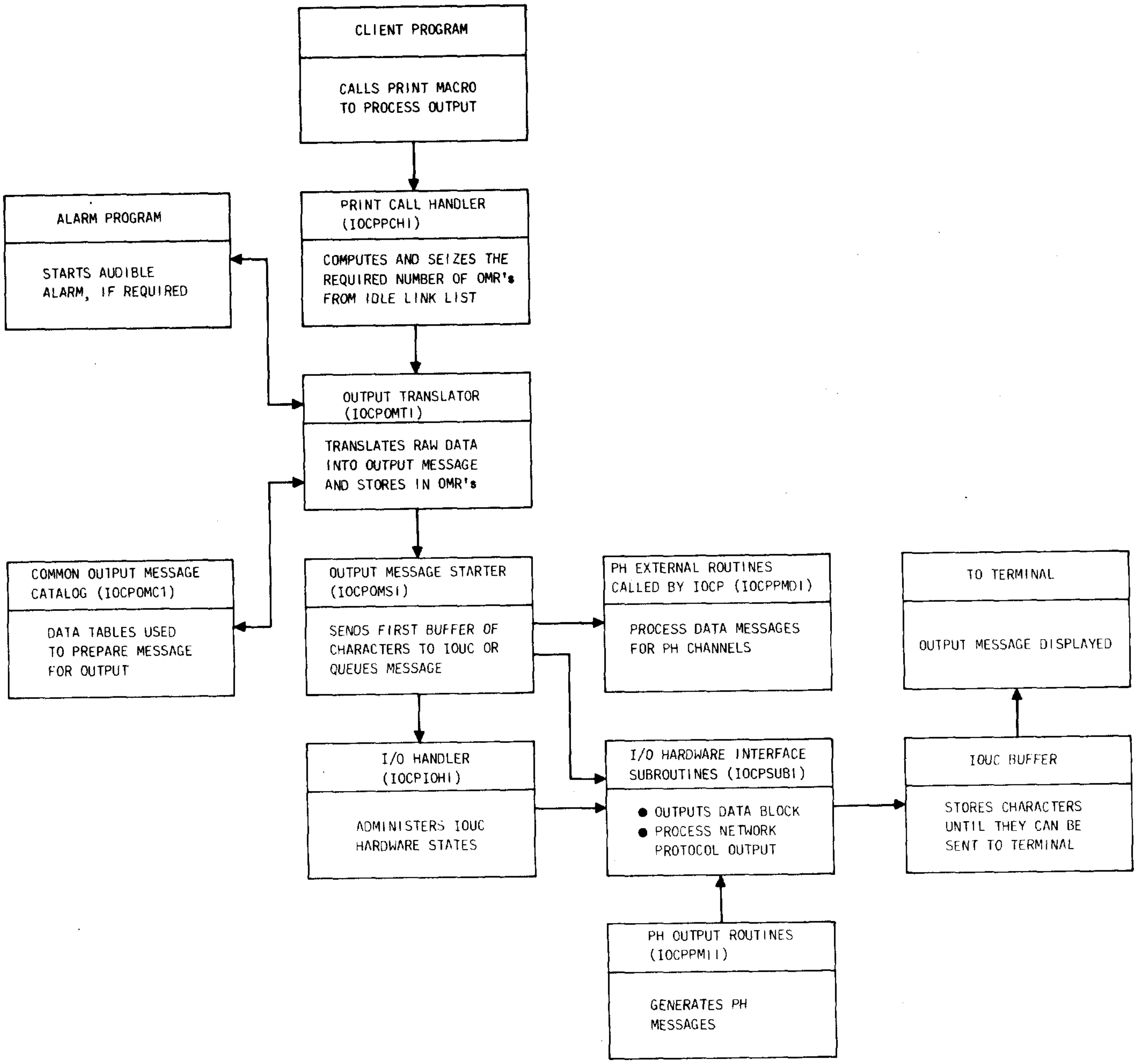

CLIENT PROGRAM
CALLS PRINT MACRO TO PROCESS OUTPUT
ALARM PROGRAM
STARTS AUDIBLE ALARM, IF REQUIRED
PRINT CALL HANDLER (IOCPPCHI)
COMPUTES AND SEIZES THE REQUIRED NUMBER OF OMR's FROM IDLE LINK LIST
OUTPUT TRANSLATOR (IOCPOMTI)
TRANSLATES RAW DATA INTO OUTPUT MESSAGE AND STORES IN OMR's
COMMON OUTPUT MESSAGE CATALOG (IOCPOMC1)
DATA TABLES USED TO PREPARE MESSAGE FOR OUTPUT
OUTPUT MESSAGE STARTER (IOCPOMSI)
SENDS FIRST BUFFER OF CHARACTERS TO IOUC OR QUEUES MESSAGE
PH EXTERNAL ROUTINES CALLED BY IOCP (IOCPPMDI)
PROCESS DATA MESSAGES FOR PH CHANNELS
TO TERMINAL
OUTPUT MESSAGE DISPLAYED
I/O HANDLER (IOCPIOHI)
ADMINISTERS IOUC HARDWARE STATES
I/O HARDWARE INTERFACE SUBROUTINES (IOCPSUBI)
● OUTPUTS DATA BLOCK
● PROCESS NETWORK PROTOCOL OUTPUT
IOUC BUFFER
STORES CHARACTERS UNTIL THEY CAN BE SENT TO TERMINAL
PH OUTPUT ROUTINES (IOCPPMII)
GENERATES PH MESSAGES

Fig. 3—Output Message Processing

3.03 There are two sets of generic-defined output message catalogs:

(a) One set contains common systems output messages

(b) One set contains application system output messages.

The first part of the catalog number points to the proper set of catalogs; the rest of the catalog number is an index into the output message directory table that contains the address of the catalog the client is requesting.

3.04 The catalog contains information required to load the message. This information includes

(a) The highest priority allowed for the message

(b) Information indicating whether a translation to ASCII is required

(c) Output channels for the message

(d) Information for the output translator to translate the output request into a string of ASCII characters.

3.05 The print call handler first compares the priority specified by the client to the entry in the catalog. If the priority is less than or equal to the highest allowable priority, the print call handler tries to load the request. The priority is used to:

(a) Serve as a limit for the client's request. (If the client's priority is greater than the highest allowed, a checksfail return to the client is made.)

(b) Specify the order in which the messages should be printed.

(c) Interface with an alarm program if an audible alarm is necessary.

3.06 The output buffer area of I/O is composed of 64-word OMRs. The first OMR of a message is the control OMR, and it contains information about how the output message is to be handled. The print call handler computes the number of OMRs to seize and to hold the raw data passed by the client. A maximum of 16 OMRs can be seized by the print call handler. The print call handler tries to obtain from the idle link list the required number of OMRs. However, there is a limit on the number of OMRs that can be active at a time. There are four threshold values used to limit the number of messages of lower priority which can be loaded so that higher priority messages can still be loaded when the I/O system is very active.

3.07 If the required numbers of OMRs are available, the print call handler calls a subroutine to get the number of OMRs needed from the idle link list and moves the client's raw data into the OMRs. The print call handler then puts the output message in the output translate link list and returns success to the client.

3.08 If the output message is already a string of ASCII characters in a client buffer area, the print call handler seizes one OMR for the control OMR, puts the output message on the output translate link list, and returns success to the client.

3.09 The maximum number of OMRs that can be used by one message is limited to 16. Any message requiring more than this number must be partitioned or segmented by the client. Only in the partitioned case does the print call handler call a subroutine (in pident IOCPTIM1 at entry point IOCPRPMA) to get an entry in the partitioned message buffer. Up to 50 partitioned messages can be buffered at a time. The partitioned message buffer contains 3-word entries into which are placed the client's return address, 24 bits of information the client wants returned at its delayed return, and the address of the control OMR. A flag is also set in the control OMR to indicate that this is a partitioned message. After the print call handler gets the entry in the partitioned message buffer, it processes the message as described previously.

3.10 For partitioned messages, IOCP returns to the client after this part of the message has been processed. This facility may be used by special handling clients who use this return as a notification that the message has been moved and their buffer area is free. Segmented messages may also use the partitioned message return to return to the client. Each segment is a separate PRINT request, but is a piece of single printed output. The pieces will be sent to the channel, in order, with no intervening output, thus having the appearance of a single printed message.

OUTPUT TRANSLATOR

3.11 The output translator (pident IOCPOMT1) is entered every base level cycle from EXECCNTL at entry point IOCPXLAT. It scans the translate output link list for the address of a control OMR which needs work. Once it finds work, an initial action is to transfer to an alarm program to determine if an audible alarm is appropriate, and, if so, start the alarm.

3.12 From the output message catalog, accessed in pident IOCPOMC1 (main memory resident) or supplied by the client upon calling the PRINT macro, the output translator gets fixed text for the message and information to convert the raw data passed by the client to a string of ASCII characters. If necessary, it pages the information from file store to main memory to get access to it. It then loads these characters into the OMRs that are seized as the need arises for a new one. Data OMRs (seized by the print call handler) are used as all the data held in them is translated. When translation is completed, all OMRs that were reserved and not used are idled and the translated output message is put on the output link list.

3.13 For output messages that are already in ASCII characters, the output translator will perform the interface with the alarm program and put the message on the output link list.

3.14 After the output translator puts the output message on the output link list, it sets a demand flag to request the execution of the output message starter.

OUTPUT MESSAGE STARTER

3.15 The output message starter (pident IOCPOMS1) is entered on base level by the demand scheduler at entry point IOCPOMS1. The output message starter goes through the output link list and looks at the destination channels for each output message. One or a set of channels, given in the control OMR, may have been specified by the client and the catalog.

3.16 The output message starter first checks the destination channel to see if it is administered by the PH. If it is, the PH is checked to see if it is idle for this application. If it is, the message is passed to the PH via a traffic application table. If the PH is busy, the message is left on queue until the PH is available. Otherwise, if the channel is not administered by the PH, the starter checks to see if the destination channel is idle. When the starter finds an idle destination, it checks to see if the message is marked for transparency. A transparent message is one that is sent without parity. If the message is transparent, the starter places the channel in the transparent output state, sends the first buffer of characters of the message to the hardware, and changes the software state from idle to output.

3.17 If the destination channel is busy, it looks at the queue word in the CMB for that priority and writes the address of the control OMR in the queue, if the queue for the priority is empty. If the output message starter has sent the first buffer of characters to the hardware or has queued the output message and if this is the last destination channel to receive the message, it moves the message to the final link list. The rest of the output is handled by the I/O handler. The output message starter also checks to see if a message is a continue or an end segment. If it is a continue, any channel in the waiting for segment state will be put in the output state and sent the first 24 characters. The rest of the output is handled by the I/O handler.

3.18 If the output message starter finds a destination channel out of service or PH application uninitialized, then a search is made for a backup destination.

3.19 The PH scans the traffic application table. When an output message is loaded, the handler segments the message, if necessary, and outputs a segment at a time with a network protocol header via a segment start routine (pident IOCPSUB1).

I/O HANDLER—OUTPUT

3.20 The I/O handler (pident IOCPIOH1) is run on base level and is scheduled at least every 60 ms by the Application Executive Control Program. It polls the hardware looking for service requests as described in paragraph 2.01.

3.21 During output, the hardware is initially loaded by a start message routine (pident IOCPSUB1). At some point, if the complete message has not been loaded, the hardware reaches a threshold that indicates it needs more characters. At that time, a load buffer service request is initiated. The I/O handler polls for service requests at least every 60 ms. When it finds a load buffer service request, it sends the next buffer of characters of the message, if there are that many left, and continues polling. It sends the whole message in this way. After sending the last character, the software state is changed from output to final so that something else will be done with the next service request from the hardware.

3.22 If the channel was not placed in the transparent mode, then, after sending the end-of-output data character, the I/O handler puts the channel in the final state and continues polling while it waits for an output to idle service requests from that channel. When the hardware sends the last character, it recognizes the end-of-output data character and brings up a service request to notify the software that the hardware has gone from the output hardware state to the idle state. The message is complete for that channel. The I/O handler then checks to see if the message is complete for all its destination channels. The message is translated only once and held in the OMRs until it is complete to all destination channels.

3.23 If the channel was placed in the transparent mode after sending the end-of-output message, the I/O handler puts the channel in the final state. It continues polling while it waits for a buffer empty service request from that channel. When the hardware sends the last character, it brings up a buffer empty service request to notify the software that all characters have been sent. If the output segment is complete to a channel administered by the PH, the I/O handler is informed of the completion. The I/O handler outputs the next message segment if any remain. When all segments have been sent and no assurance of receipt was requested, the message is marked complete for the application. If message assurance was requested, the message is held awaiting a positive response. When a positive response is received, the message is marked complete for the application. If no acknowledgment is received in a given time or a negative acknowledgment is received, the message is sent again segment by segment.

3.24 If the output message is complete to all destination channels, the I/O handler first checks the control OMR to see if this is a partitioned message. If so, it sets a flag in the partitioned message buffer corresponding to the message for the return to the client program. It sets a demand flag for the partitioned message buffer scanning program to be entered at entry point IOCPPCLI. Then it moves the OMRs from the final link list to the idle link list. Each time a hardware buffer of characters is sent, the channel is put on 10-second timing. This 10-second timing for a load buffer service request from the hardware is to guard against hardware error and to ensure that the hardware is processing messages.

3.25 When the channel goes idle, the I/O handler checks for the segmented message, restores the OMRs to idle, and is finished with that message. If the message was a start or continue segment, the channel is put in a waiting for next segment state to await another segmented message request. If the message was not a segment or if it was an end segment, the I/O handler checks to see if any of the queue words in the CMB have an OMR address. If a message is queued, the I/O handler sends the first buffer of characters of the next message so that there is no delay from the end of one message to the start of the next message. This message then is handled as the previous one.

I/O HANDLER—BREAK REQUESTS

3.26 When the INTERRUPT or BREAK key on a terminal is pressed or input is detected during output on a half-duplex channel, the hardware sets a break service request and enters the break state. This usually occurs when someone wants to enter an input message while output is in progress. The I/O handler finds the channel in the break state and stops the output. A check is made with an application routine to determine if the message should be restarted. For restarting, it saves the start address of the control OMR in a queue in the CMB set aside for breaks. The output message will be restarted at the beginning so that if the output message is printed on paper, the entire message will appear. The I/O handler sends the break acknowledgment to the channel and changes the software start to break loaded. In half duplex, the channel waits up to 2 minutes for an input message before sending the interrupt output. In full duplex, any interrupted input is abandoned and the output is restarted.

I/O HANDLER—MAINTENANCE REQUESTS

3.27 When the hardware detects an error, it brings up a maintenance request. Various hardware sections describe the hardware error sources. Maintenance requests that do not cause F-level interrupts are handled by the I/O handler and are found by polling. The I/O handler calls PFLR (fault recognition for I/O) to determine what the maintenance request is. The PFLR program is described in Section 254-280-310 (Fault Recovery Programs—Software Description—1A Processor) and in the appropriate application section. The PFLR program will restore the channel to service or remove it from service. If the channel is out of service, there is no more action with that channel. Future messages for that channel will be sent to the backup channel. If the channel is not out of service, the I/O handler continues to service the channel from where it left off at the maintenance request after printing error acknowledgment.

4. TIMING FUNCTIONS

4.01 The timing program (pident IOCPTIM1) is run on clock demand, scheduled every 5 seconds by the Application Executive Control Program at entry point IOCPTIM1. Each time an input or output data transfer is initiated on a channel, a time-out time is entered for that channel on a time-out list. This program searches this list for channels that have not been serviced in the allotted time. The actions taken by the program, in case of time out, are shown in Table B.

4.02 The timing lists provide a check on the software to ensure responses in the time limits allocated. It also provides a check on the person entering the input message so they are not allowed to tie up the channel. In addition, it provides a check on the hardware to guard against problems that do not cause interrupts but need maintenance.

4.03 The partitioned message buffer scanning program (pident IOCPTIM1) is run on clock demand, scheduled every 1 second by the Application Executive Control Program at entry point IOCPTDP1 as shown in Table A. It scans each entry in the partitioned message buffer to ensure that messages are being processed. If an entry is active and not marked for return to the client, it looks at the control OMR. If the OMR is not in the translate, output, or final state, the message is assumed lost and the return to the client is made in the same way as if the message was completed properly. If the partitioned message buffer scanning program finds a return flag set for a client, return is made to the client. The client program can use the rest of the 3-ms segment to determine what it needs to do next.

4.04 Once the partitioned message buffer scanning program finds a client with a return flag set, it sets a demand flag to be rescheduled at entry point IOCPPCLI. At this demand entry it takes up where it left off; it finishes scanning the partitioned message buffer, returning to clients as it finds return flags set.

5. MISCELLANEOUS FUNCTIONS

PHASE ACTIONS (FIG. 4)

5.01 During a phase, before the software structures are zeroed or reinitialized, an entry from the phase program is made in a message save program (pident IOCPMSR1 at entry point IOCPSAVE). This program takes all the messages buffered in OMRs and saves them in an area in file store so they can be manually dumped after the phase. If a set of messages is already in file store, no further messages will be saved until the existing set has been manually dumped or cleared. Therefore, when more than one phase occurs, only the messages from the first phase

TABLE B

TIMING FUNCTIONS AND TIME-OUT ACTIONS

TIMING FUNCTION	TIME-OUT ACTIONS
Waiting-for-Acknowledgment (5 Seconds)	Generate a no-acknowledgment (NA) and resume processing for the channel.
Waiting-for-Output-Service-Request (5 Seconds)	Restart output message from beginning and reenter on timing list. Upon a second time-out the message is not restarted but is treated as though it had been completed (the message is lost).
Waiting-for-Input-Service-Request (120 Seconds)	Generate a time-out acknowledgment (?T) for the channel.
Waiting-for-Output-Message Acknowledgment (5 Seconds)	Retry message. Upon second time-out remove channel if backup is in service and try message on backup. If backup is out of service, the message is treated as though it had completed and an error message is output to the maintenance channel.

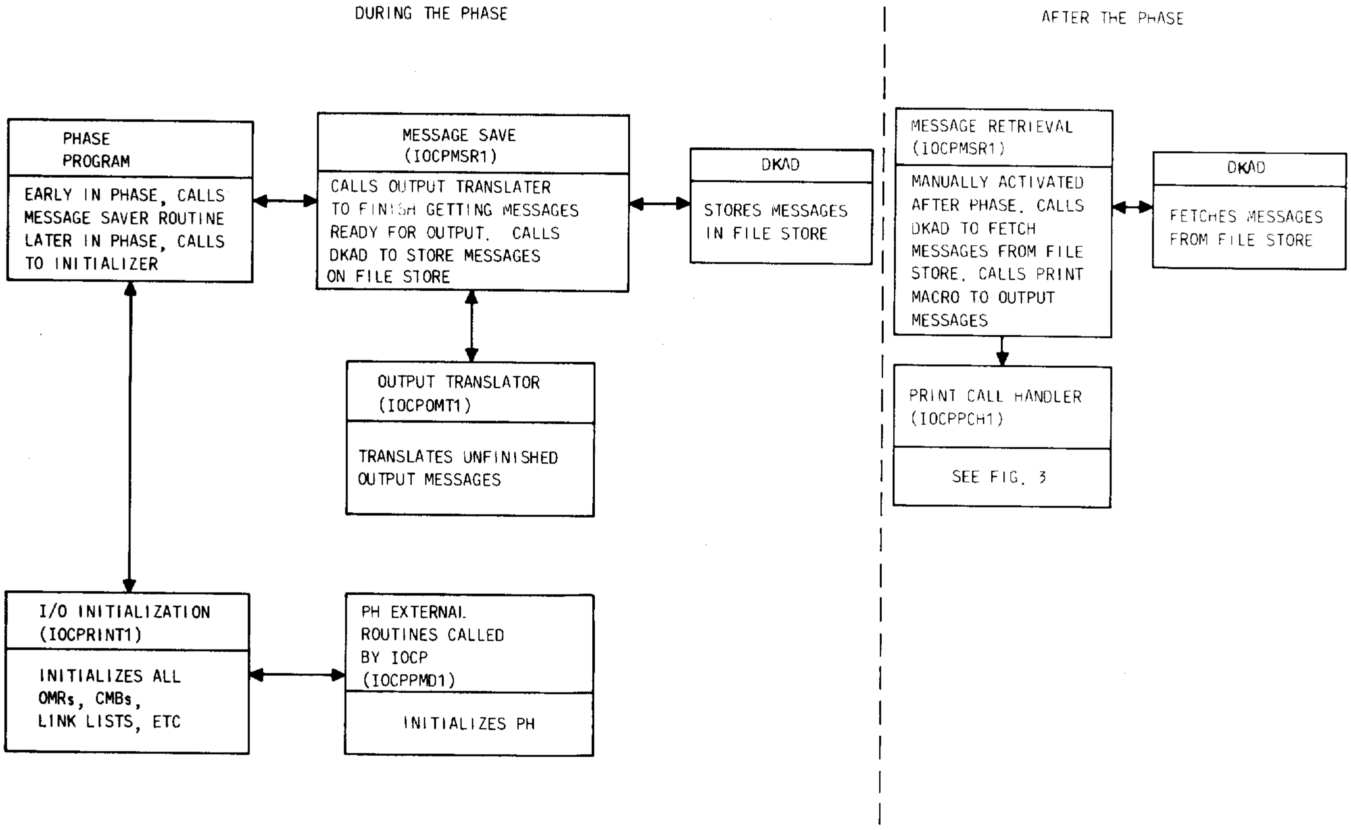

DURING THE PHASE
AFTER THE PHASE
PHASE PROGRAM
EARLY IN PHASE, CALLS MESSAGE SAVER ROUTINE LATER IN PHASE, CALLS TO INITIALIZER
MESSAGE SAVE (IOCPMSR1)
CALLS OUTPUT TRANSLATER TO FINISH GETTING MESSAGES READY FOR OUTPUT. CALLS DKAD TO STORE MESSAGES ON FILE STORE
DKAD
STORES MESSAGES IN FILE STORE
OUTPUT TRANSLATOR (IOCPOMT1)
TRANSLATES UNFINISHED OUTPUT MESSAGES
I/O INITIALIZATION (IOCPRINT1)
INITIALIZES ALL OMRs, CMBs, LINK LISTS, ETC
PH EXTERNAL ROUTINES CALLED BY IOCP (IOCPPMD1)
INITIALIZES PH
MESSAGE RETRIEVAL (IOCPMSR1)
MANUALLY ACTIVATED AFTER PHASE. CALLS DKAD TO FETCH MESSAGES FROM FILE STORE. CALLS PRINT MACRO TO OUTPUT MESSAGES
DKAD
FETCHES MESSAGES FROM FILE STORE
PRINT CALL HANDLER (IOCPPCH1)
SEE FIG. 3

Fig. 4—IOCP Actions During Phase of Software Initialization

are saved. The problem should be indicated in messages from the first phase.

5.02 If the file store is free, the message save program first checks the translate link list for any messages loaded but not yet translated. If there are any, the message save program calls the output translator until the messages are translated and on the output or final link list. The message save program now takes the ASCII characters from the OMRs and puts them in a block of main memory. Each 1024-word block is written out in file store. All OMRs containing messages waiting for output are emptied in this way. Return is made to the phase program.

5.03 Later in the phase, the I/O reinitialization program is entered at pident IOCPINT1, entry point IOCPINIT, to reinitialize all software structures associated with I/O. The PFLR program has previously reconfigured the I/O hardware. The I/O initialization program initializes all call store structures for I/O, zeros the OMRs, and links all OMRs on the idle link list. It zeros the CMBs so that if any input is in progress it is lost. It reinitializes the head cells for the four link lists (idle, translate, output, and final) and initializes the output message classes. It calls the PH for initialization and finally returns to the phase program.

5.04 After a phase, the saved messages are available for manual dumping. When manually requested, the message retrieval program (pident IOCPMSR1 at entry point IOCPDDMP), retrieves the messages from file store through the File Store Administration Program (DKAD). The message retrieval program calls the PRINT macro to output the messages in file store. One message is output for each call of the PRINT macro. The message retrieval program handles the block of messages in file store as though the block was a partitioned message. The messages appear just as they would if they had been output from main memory, except for a header indicating that they are from file store.

MAINTENANCE CONTROL CONSOLE KEY

5.05 When the maintenance control console key that reinitializes the I/O system is pressed, it is detected by the Master Control Console Common Control and Monitor Program (MCCM). The MCCM program transfers to the clear buffer routine (pident IOCPINT1 at entry point IOCPCLRB). The clear buffer routine does most of the work that is done during a phase, except that the OMRs are not zeroed. The clear buffer routine zeros the CMBs, relinks all OMRs on the idle link list, adjusts the traffic counters and the other three link list head cells, and calls the PH for initialization. It then sends a message to each channel in service to inform personnel, if they were waiting for messages, that the messages have been cleared from the system.

I/O CHANNEL ROUTING

5.06 There are several input messages to change routing of output messages or to change the output message class(es). These input messages are described in the Input Message Manual. A message is classed according to function (such as diagnostic), subject (such as call store), and responsibility (such as Maintenance Operation Center). Normally, these classes do not have any channels associated with them, so messages are returned to the channel that requested them, if manual, or to the channel that is designated to receive the message.

5.07 ***Output Message Class Routine:*** In response to an input message, the I/O channel routing program (pident IOCPREA1) changes the I/O channel routing. If the input message is to initialize output message class routing, the pident is entered at entry point IOCPMCLI. A channel may be added to a specific message class. A person, for instance, working on a call store can add their beltline channel to the call store message class and have all messages about call store come to their terminal in addition to the other channels that receive it. In this case, the input message would be a request to change the message class routing, and the pident is entered at entry point IOCPRTE2.

5.08 ***Channel Monitoring:*** A channel may be monitored; all messages sent to one channel may be sent to another. A monitor input message does this, and enters the pident at entry point IOCPMON1. A copy of inputs, as well as outputs, is also sent to the Switching Control Center channel if it is doing the monitoring.

5.09 ***Channel Rerouting:*** Messages for a particular channel may be rerouted. For instance, if no personnel will be near a particular channel for a period of time, messages can be rerouted to a channel in an area where personnel are available. Nothing will appear on the channel from which the messages were rerouted. Rerouting is done by a route channel input message, and the pident is entered at entry point IOCPRTE1.

INTERFACE WITH PFLR

5.10 Any time a channel is out of service and requires restoring to service from a restore request, a successful diagnostic, or a phase, PFLR calls a restore subroutine. The restore subroutine initializes all memory associated with that channel to take it from the out-of-service state to the in-service state. The restore subroutine zeros the CMB and makes the channel available for messages.

5.11 When a channel is to be removed from service, PFLR calls a remove subroutine (pident IOCPSUB1 at entry point IOCPRMVC). Before the channel is marked out of service in the software, a check is made to see if an output message to that channel is in progress or queued. Any output messages queued are removed, and if this is the last channel to receive the message, the OMRs are restored to the idle link list. If an output message is in progress to that channel, the same action is taken. The message is lost because it cannot be completed to that channel since PFLR has already removed the channel from service. A message that is in progress or queued is not output to the backup channel. Then the remove subroutine marks the channel in the out-of-service state. Any messages loaded for output will now be automatically routed to the backup channel.

CLIENT SERVICE ROUTINES

5.12 The IOCP program includes a pident, IOCPUSR1, which is comprised of separate routines that convert various data from one format to another. Client programs may access any of these routines to convert data from binary to ASCII or vice versa, convert binary to binary coded decimal, receive the time of day in ASCII, etc. These instructions are consolidated in one program rather than repeating all the necessary coding in each program requiring data format conversions.

5.13 The pident IOCPSUB1 contains routines that are used by the 1A Processor client to initiate requests associated with secure dial-up. Entries at IOCPDIAL and IOCPHANG are made for a request to dial-up and for a request to hang up, respectively. The request to dial-up is made via the I/O DIAL_UP CALL_SUB that has four input parameters (TIMEOUT, DIGITS_ADR, NUM_WORDS, and CHAN) and returns a completion code (COMP_CODE). Upon entry at IOCPDIAL the parameters are checked for correctness and if they are found valid, the call is initiated. Control is then returned to the client. The completion code indicating "call request in progress" is passed as output from the routine if all input checks pass. Other completion codes may be passed as outputs under input check failure conditions. The request for IOCP to hang up is made via the IO_HANG_UP CALL_SUB. The only input to entry IOCPHANG for this request is CHAN (channel number). The action is initiated and the control is passed back to the client without any completion status.

6. HIGH-SPEED I/O PROTOCOL

GENERAL

6.01 The high-speed I/O protocol function allows the system to communicate through the IOP hardware to off-site clients at higher data rates than previously available. This function provided synchronous full-duplex I/O at rates up to 4800 bps on dial-up connections and up to 9600 bps on private lines. ♦ For the CPR7 generic, it provides up to 56,000 bps on private lines.♦ The IOCP high-speed I/O protocol functions are provided primarily by the protocol pidents described in paragraphs 6.11 through 6.18 and shown in Fig. 5.

OVERVIEW OF HIGH-SPEED I/O PROTOCOL FUNCTIONS

A. General

6.02 The application programs interact with far-end clients via the 1A Processor software, particularly the IOCP and the IOP. The IOCP and the IOP are totally responsible for ensuring integrity on the data links for high-speed protocol functions.

6.03 A hierarchy of interfaces between the system and the far end is used. The functional breakdown consists of four levels of control:

(a) The application program that generates messages

(b) The network control that breaks up long messages into manageable blocks and controls routing

(c) The link control that ensures error-free transmission of data blocks on the link

(d) The data set control that controls data sets to facilitate transmission on a link.

The network control and part of the link control functions are achieved in the IOCP. The remainder of the link control and all the data set controls are handled by the IOP.

B. Generation and Transmission of a Protocol Message

6.04 Typically, an application program generates a message and passes it to the message control (IOCP). If the message is too long, the IOCP divides it into segments of suitable size. If the message length is short, it may consist of only one segment. The IOCP may request, from the network control at the far end, a positive acknowledgment of the entire message. The IOCP then passes one segment at a time with network protocol header to the link control in the IOP.

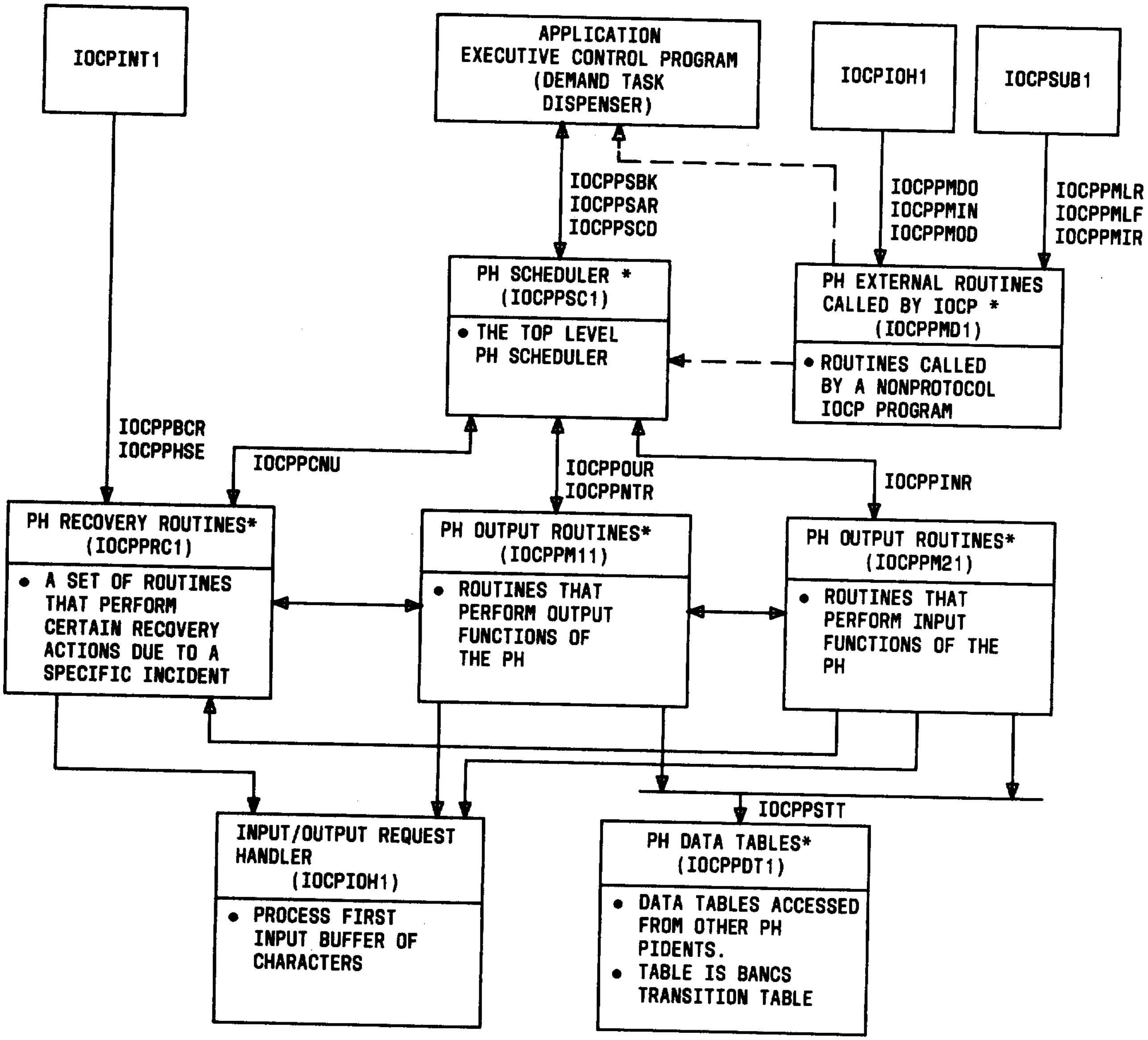
6.05 The IOP link control accepts a block, attaches a link protocol header and trailer, and uses data control to transmit it on the link. The data set control at the far end receives the block and gives it to the far-end link control. The link control recomputes the check over the header and data and compares it against the check accompanying the header and data. If the two agree, the link control sends acknowledgment to the link control on the other side. The remaining block (stripped of protocol header and trailer) is then given to the far-end network control. If the network segment received is acceptable, a positive response acknowledgment is returned by the network control when requested.

6.06 The receiving end message control accumulates these blocks and delivers the message to the application program. A network-level no acknowledge is generated if any segment is not received error-free. This causes the originating end's IOCP to retransmit the entire message.

6.07 The network control (or IOCP) at the originating end may send several messages before an acknowledgment to the first message is received. On receiving an acknowledgment, the originating IOCP discards its copy of the acknowledged message.

C. BANCS Message Protocol

6.08 The BANCS forms an integrated communications system that is widely used both within


IOCPINT1
APPLICATION EXECUTIVE CONTROL PROGRAM (DEMAND TASK DISPENSER)
IOCPIOH1
IOCPSUB1
IOCPPSBK
IOCPPSAR
IOCPPSCD
IOCPPMDO
IOCPPMIN
IOCPPMOD
IOCPPMLR
IOCPPMLF
IOCPPMIR
PH SCHEDULER * (IOCPPSC1)
• THE TOP LEVEL PH SCHEDULER
PH EXTERNAL ROUTINES CALLED BY IOCP * (IOCPPMD1)
• ROUTINES CALLED BY A NONPROTOCOL IOCP PROGRAM
IOCPPBCR
IOCPPHSE
IOCPPCNU
IOCPPOUR
IOCPPNTR
IOCPPINR
PH RECOVERY ROUTINES* (IOCPPRC1)
• A SET OF ROUTINES THAT PERFORM CERTAIN RECOVERY ACTIONS DUE TO A SPECIFIC INCIDENT
PH OUTPUT ROUTINES* (IOCPPM11)
• ROUTINES THAT PERFORM OUTPUT FUNCTIONS OF THE PH
PH OUTPUT ROUTINES* (IOCPPM21)
• ROUTINES THAT PERFORM INPUT FUNCTIONS OF THE PH
IOCPPSTT
INPUT/OUTPUT REQUEST HANDLER (IOCPIOH1)
• PROCESS FIRST INPUT BUFFER OF CHARACTERS
PH DATA TABLES* (IOCPPDT1)
• DATA TABLES ACCESSED FROM OTHER PH PIDENTS.
• TABLE IS BANCS TRANSITION TABLE

* PROTOCOL PIDENTS

Fig. 5—IOCP High Speed I/O Protocol Pident Interfaces

and outside the operating telephone companies. The portion of the IOCP that provides the BANCS message protocol is referred to as the BANCS handler. The BANCS handler deals with two types of messages: system and traffic.

System Messages

6.09 System messages are used to send information regarding the status and availability of the various components in the BANCS network. They are initiated at the IOCP and consist of a short text preceded by a BANCS header. System messages always require responses. There is a variety of system messages that may be generated in a BANCS network. Only a few of these messages may cause the BANCS handler to change state and/or take action.

Traffic Messages

6.10 Traffic messages are initiated by the application programs and destined for application level software at the far end. These messages are broken up into predetermined size segments by the IOCP. The BANCS header is prefixed onto each of the segments. These blocks are then passed on to the link control for transmission.

IOCP HIGH-SPEED I/O PROTOCOL PIDENTS (FIG. 5)

A. General

6.11 The IOCP high-speed I/O protocol functions are handled primarily by six IOCP protocol pidents (Fig. 5). The protocol handling functions of the nonprotocol IOCP pidents are described where applicable in Parts 2 through 5 of this section.

B. PH Scheduler (IOCPPSC1)

6.12 The pident IOCPPSC1 is the top level PH scheduler for the IOCP protocol pidents. The pident IOCPPSC1 is entered at entry point IOCPPSCD from the demand task dispenser to look for work to be done on any application. If no work exists, it resets the ordered bits buffer and returns to the demand task dispenser. For all 1024 base level entries, a message timing routine (PH_TIMER) is run that examines each application for an output message that has timed out. For all 1024 base level entries, a check on the PH channel status table is also run. The order in which IOCPPSC1 routines look for work is:

(1) Application restoral (this has priority and is checked first)

(2) Resume previous task (if task is in a segmented state)

(3) Any pending input work

(4) Any pending output work.

C. PH External Routines Called By IOCP (IOCPPMD1)

6.13 The pident IOCPPMD1 is a collection of routines, each of which can be called by a nonprotocol IOCP program. Each IOCPPMD1 routine is a separate entity that has a unique entry and one exit. The routines perform the functions of loading the input work list when the input is from the far end. They enter the output work list with an entry when the IOCP has completed an output. The routines also enter the output work list with an entry when the IOCP receives a response from a client. Another routine sets the channel status table for a channel failure and schedules a restoral routine when a channel is restored.

D. PH Recovery Routines (IOCPPRC1)

6.14 The pident IOCPPRC1 is a set of routines that perform various recovery actions when a specific incident occurs. The routines are designed to zero the call store area, set values in the call store area, call the IOCP to release the OMRs that are no longer needed, and to put specific messages in the output work list for outputting. These actions are taken in cases of application restoral; erroneously receiving an input message when the previous one has not been completed; a work list problem; received closed global application (CGA), off (OFF), force off (FOF), or initialize (INT); phase; or a buffer clear.

E. PH Input Routines (IOCPPM21)

6.15 The pident IOCPPM21 is a collection of routines that perform the input functions of the PH. The main input routine is called by the PH when there are messages to be processed. This routine may call any of the other supporting routines. Each routine is a complete entity. They each have one entry point and they return to the routine that called them. The only other exit the routines may do is an abort exit.

F. PH Output Routines (IOCPPM11)

6.16 The pident IOCPPM11 is a collection of routines that perform the output functions of the PH. The two main output routines are called by the PH when there are messages to be processed. Each of these routines may call any of the other supporting routines. Each routine is a complete entity. They each have one entry point and they return to the routine that called them. The only other exit the routines may do is an abort exit.

G. PH Data Tables (IOCPPDT1)

6.17 The pident IOCPPDT1 is a data table accessed from the other PH pidents. This data table is the BANCS transition table. The pident IOCPPDT1 is entered at entry point IOCPPSTT from either pident IOCPPM11 or pident IOCPPM21 to access the PH data tables. The state-stimuli table is used by the PH to advance through the logical states of the BANCS protocol.

H. Private Memory of Protocol Pidents

6.18 All six of the IOCP protocol pidents (IOCPPMD1, IOCPPM11, IOCPPM21, IOCPPSC1, IOCPPDT1, IOCPPRC1) share a private block of nonphase protected memory. Each routine in these six pidents can use this memory for any purpose. Each routine can assume that the memory is free to be used at the beginning of its processing.

7. ABBREVIATIONS AND ACRONYMS

7.01 The following is a list of abbreviations and acronyms used in this section.

ASCII	American Standard Code for Information Interchange
BANCS	Bell Administrative Network Communications System
CMB	Channel Memory Buffer
DDCMP	Digital Data Communications Protocol
DKAD	File Store Administration Program
GULP	Generic Utility Program
ICB	Input Character Buffer
I/O	Input/Output
IOCP	Input/Output Control Program
IOP	Input/Output Processor
IOUC	Input/Output Unit Controller
IOUS	Input/Output Unit Selector
MCCM	Master Control Console Common Control and Monitor Program
OMR	Output Message Register
PFLR	Process F-Level Recovery Program
PH	Protocol Handler