INTELLIGENT PERIPHERAL BUS
STRUCTURE, TIMING, AND PROTOCOL SPECIFICATION

Texas Instruments Incorporated
Consumer Products Group
Calculator Division

2/9/83

Revision 3.2

SECTION 1

Introduction

1.1 Purpose of This Document

This specification describes the timing and protocol for a communications bus to connect intelligent peripherals to calculator type products. This document is meant to be used in formulating the detailed software design and to allow an accurate review of the plans to support peripherals.

1.2 Scope of This Document

This document describes the signals, messages, and protocol on the peripheral bus. The electrical interface is not described in detail; specifically the design of hardware to implement the bus is not discussed. The primary discussion centers around the method used by devices to perform data transmission on the bus. The specification includes the following sections:

1. Overview of the bus characteristics.
2. Description of bus signals.
3. Bus message structure.
4. Access to I/O subroutines within the calculator.
5. Notes on the use of the bus
6. Bus transfer examples

An application programmer need only read the section on I/O subsystem access in order to understand how to access peripherals from assembly language. The document is structured with that section providing a complete description for software use.

1.3 Terminology

IPB - Intelligent Peripheral Bus

RAM - Random Access Memory

HSK - Handskake I/O Control Line

BAV - Bus Available I/O Control Line

LAT - Internal I/O Control Signal Derived from HSK

PAB - Peripheral Access Block

SAB - Slave Access Block

SRPAB- Service Request Peripheral Access Block

DSR - Device Service Routine

When the symbol '>' precedes a number in this document, it indicates that the number is hexadecimal.

1.4 Related Documents

Related documents may be obtained by contacting Tom Ferrio at TI Lubbock; phone 741-3362; MSG ALCC

SECTION 2

Specification Overview

2.1 General Features

The bus is designed to provide data transmission between a calculator and peripherals. It is organized in a master-slave arrangement with the controlling calculator as the master and the peripherals as slaves. In this manner the calculator on the bus acts to control data activity. Normal communication will be initiated by a command message from the calculator to a peripheral. The peripheral responds to the calculator with a data or status message to signal completion of the command.

Certain peripherals, when allowed by the master device, may initiate a device poll by the master. This effects a service request feature by a slave device.

Through software loaded into RAM, the calculator can also have a slave communication mode. This allows it to operate on the bus as a peripheral to another calculator (to allow two calculators to communicate) and provides the capability for a more complex communication structure for applications which require it. This capability is selectable from both assembly and high level languages. Capability has also been given for the calculator to pass the master control to another device on the bus. This does not allow a device to arbitrarily take control as the master device but only do so by command from the current master.

Transmissions on the bus are defined in the context of a "message frame" which consists of a command message from the master and a response message from the slave. The form of the messages is described in detail in a later section. This message concept requires that each peripheral be intelligent enough to decode information and bus status and be able to follow protocol. It is assumed that each peripheral will contain a microcomputer for the bus interface and control functions.

SECTION 3

Bus Signal and Timing Descriptions

3.1 Signals

The bus is connected to all devices in a parallel manner. No buffering is provided at any point along the data bus. The physical bus consists of 8 lines. A ground referance signal and a line reserved for future use. make up 2 of the lines. There are 4 parallel data lines defining a 4 bit nibble as the basic unit of information carried on the bus. Data within the communication protocol is defined in 8-bit units (bytes). Each byte corresponds to two transmissions on the bus, least significant nibble first. Data is output to the bus via open collector drivers.

D3	----------	most significant data bit
D2	----------	data bit
D1	----------	data bit
D0	----------	least significant data bit
HSK	----------	handshake
BAV	----------	bus available
FUT	----------	reserved for future use
GND	----------	ground

The speed of transmission of the data bus is controlled by the handshake line HSK. The BAV (bus available) signal is used to designate the beginning of a command message from the master and for a slave device to request service as described in later sections.

3.2 Handshake Timing

The signal timing of HSK and the data lines is illustrated in Figure 3.1. The falling edge of HSK is the signal to receiving devices that a nibble of data is available on the bus. HSK triggers a signal that informs the software I/O drivers that a nibble of data is available on the bus. The rising edge of HSK is the signal to the transmitting device that all receivers have read the data. HSK is an open collector line so that any one

device may keep it at a low level. When the receiving device(s) see that HSK has gone low they rapidly (through hardware) pull HSK low also. The receiving device(s) then hold HSK low until they have processed the data. The transmitting device will release HSK shortly after pulling it low. This normal interaction is illustrated in Figure 3.2. If the transmitting device is slower than the receivers then it may dictate the bus speed as shown in Figure 3.3.

When a device is not interested in the data being transmitted it may disable itself from the bus and wait for the next message frame (denoted by a BAV transition from high to low). Non-active devices need not even participate in the handshake activity.

Figure 3.1
Bus Handshake Sequence

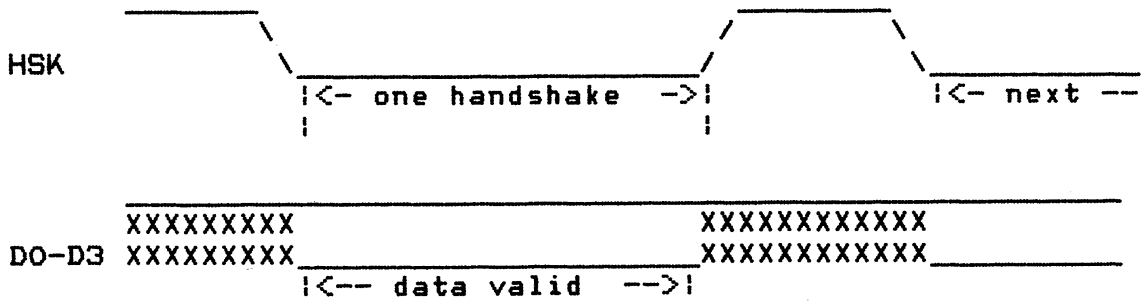


Figure 3.2
Handshake Components
(Receiver Limited)

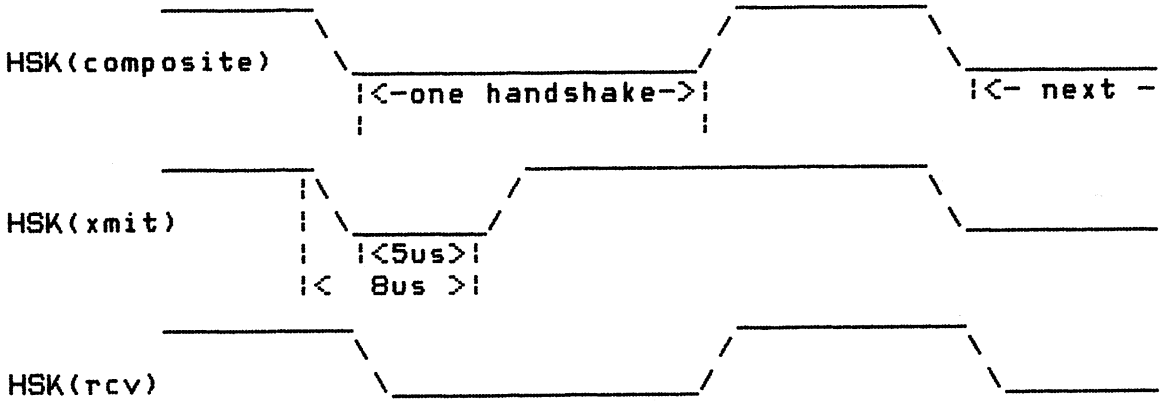


Figure 3.3
Handshake Components
(Transmitter Limited)

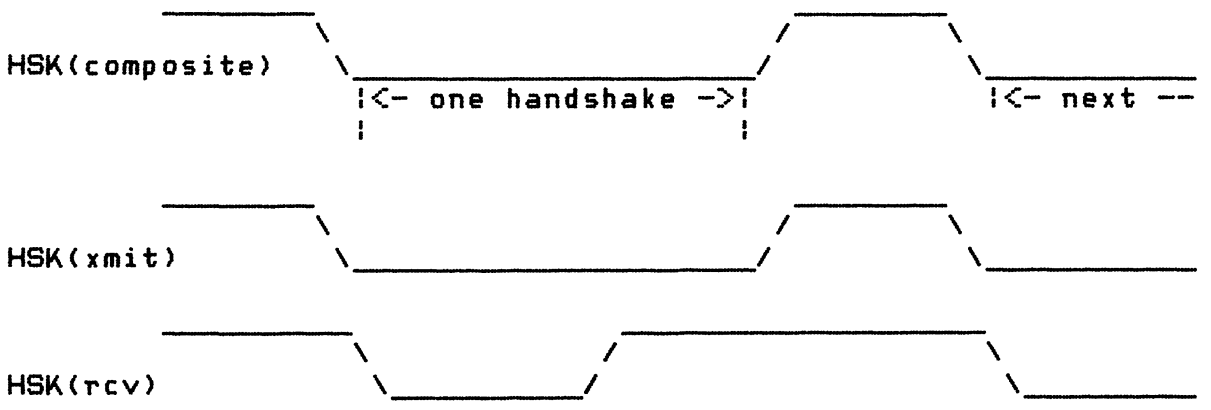


Table 3.1
Handshake Timing Parameters
(Microseconds)

Item	Min	Max
HSK low to data valid	0.5	-
HSK low(xmit) to HSK low(bus)	-	3
HSK low(bus) to HSK low(rcv)	-	5
HSK low(xmit) to HSK high(xmit)	8	-
HSK high to data invalid	0	--**
HSK high to HSK low	8	20000*

* within a message
** because data is output via open collector buffers, and HSK going high causes ones to be written into their output latch, data becomes invalid on the rising edge of HSK.

3.3 Bus Time-out

Within a message frame (When Bus Available is low), whenever HSK is high it has 20 ms to go low, or a bus time-out condition

results.

3.4 Data Transfer Order

All data and overhead information is sent in increments of one byte. As bytes are transmitted, the least significant nibble is sent on the bus first, followed by the most significant nibble. Whenever 16 bit (two byte) fields, such as the data length in the command message, are sent, the least significant byte is transmitted first.

SECTION 4

Bus Message Structure

4.1 Protocol

As mentioned earlier, the data bus transmits command and response messages within the context of a message frame. In general the transmission of one command message from the master device will cause a response message to be transmitted back from the slave device selected. Each message consists of several nibble transfers as described in the previous section.

Each transmitted message contains overhead information to indicate such things as the slave device selected, the command code to be performed, and the data length. The BAV (bus available) signal specifies the start of a message frame. When the master device starts a message frame it first pulls BAV low. The command message from the master then follows that falling edge of BAV. The falling edge of BAV alerts all slave devices to look for the two-nibble device code which is always transmitted first in the command message (again, least significant nibble first). The BAV signal does not return to the high level until the message frame is complete. This is illustrated in Figure 4.1 below.

Figure 4.1
Message Frame

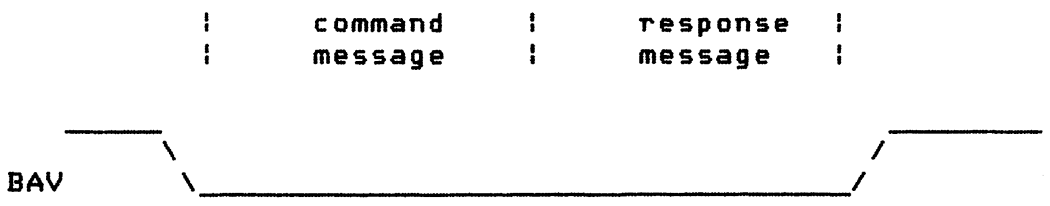


Table 4.1
Message Timing Parameters
(Microseconds)

Item	Min	Max
BAV low to HSK low	5	20000
HSK high to BAV high	1	--
End of command to start of response	10	--
HSK high to HSK low	8	20000
BAV high to BAV low (Within a message frame)	8	--

The first two nibbles of the command message always contain the device code of the slave device to be addressed. All devices on the bus will read this number and test for a match. After the device code has been sent all devices except the one selected will ignore all further data in the message. The hardware will be designed such that they will not have to participate in the handshake sequence until the next falling edge of BAV. In this way the bus will operate at the maximum data rate of the two "talking" devices once the device code has been transmitted.

Any device may extend the time to process data or wait for an operation to complete by holding HSK low until it is ready to start the next operation. Whenever HSK is high during a message, it must go low within 20ms or the receiver may time out (Time outs are not required for peripheral slave handlers).

The command and response messages are detailed in a following section.

As mentioned previously, each device has a unique device code. In addition, the device code >00 is reserved to be recognized by all devices.

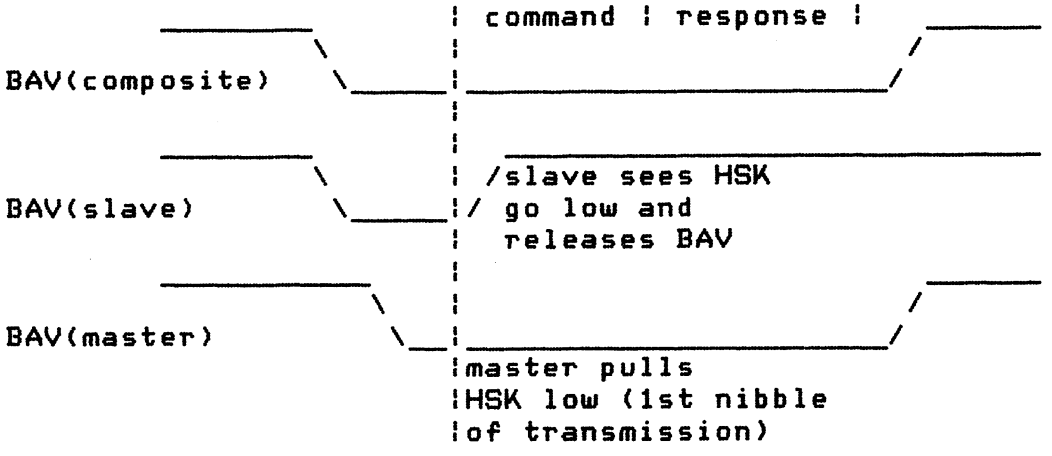
4.2 Poll Request

The master device may tell a slave through a normal message frame that it can request service from the master. When that has been done the slave may interrupt the master by pulling the BAV signal low. When the slave device pulls BAV low that initiates a message frame where the master polls a slave device which has been enabled to request service. The "service request poll" command code is used for this. Figure 4.2 illustrates the standard request sequence and the way BAV is driven. The master will poll one of the enabled devices during the message frame.

If several devices have been enabled, then the master may not poll the requesting device in the current frame. If that happens then the polled device will say "No", and the service request handler will set up to poll the next device and return to the interrupted application. After waiting for a period, the requesting device will pull BAV low again to initiate another poll. The master device will cycle through the devices which are enabled so the requesting device will be polled eventually.

If the master receives an interrupt while in bus master mode, and the application has not yet acknowledged a previous service request from the current device, then the service request handler will transmit a null operation message down the bus. If no devices are enabled for service requests, then the IOS will ignore the interrupt.

Figure 4.2
Service Request Sequence



Because of timing uncertainties the service request may nearly coincide with a normal message frame. In that case, illustrated in Figure 4.3, the master device will miss the request and proceed with the normal frame. The requesting device will treat this similar to the case where another device was polled and re-assert the request after the current message is complete. A difference arises when the master is coincidentally sending a normal message when the device requests service. This might occur when the master sends a message to the requesting peripheral or to another peripheral. In that case the requesting device will either see that the device code is not its device code, or that the command code is not the "service request poll" code and must process it normally. The requesting device would then re-request service after the message is complete.

Figure 4.3
Service Request Miss

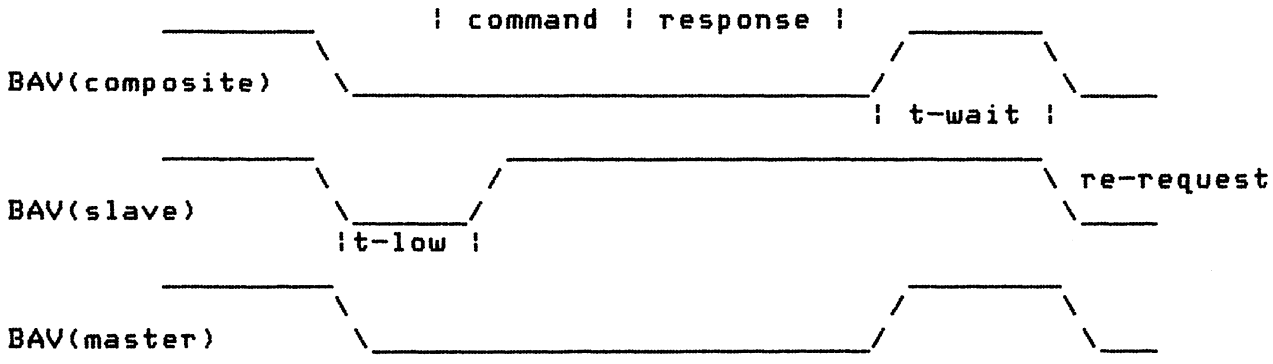


Table 4-2
Polling Timing Paramemters

Item	Min	Max(us)
t-low	** see note **	
t-wait	2000	--

Note: The slave device will wait for the first HSK from the bus master before releasing BAV.

This specification allows multiple devices to be requesting service without any unresolvable conflicts.

4.3 Command Message

As mentioned earlier, communication between the calculator and a peripheral consists of a command message and a response message (In some cases, there will be no response to the command).

The following data is contained in a command message:

Field	Bytes
Device code	1
Command code	1
Logical Unit Number	1
Record number	2
Buffer length	2
Data length	2
Data	variable

NOTE

All data structures shown in this specification are examples of how the data can be organized, and are not meant to restrict the use of other internal data structures.

A bus master (console) should be able to transmit and receive data at a minimum rate of 3000 bytes/second. This accomodates peripherals that require a minimum I/O transfer rate.

Each device will have a unique device code (255 codes are available). Devices with several separately addressable units will have several device codes - one for each unit. Device codes have been assigned as follows (tentative);

Device Code (in decimal):

0	- all devices
1-8	- tape mass storage
10-13	- printer/plotter
15	- Low cost printer
20-27	- RS-232 interface
30-33	- TV interface (color)
40-43	- TV interface (black and white)
50-57	- Centronix parallel interface
60-67	- calculator in slave mode
70-77	- modem
80-87	- GPIB interface
90-93	- bar code reader
100-107	- Floppy disk drive
254	- Console keyboard
255	- Console display

Device code 0 is normally used with the Null Operation or bus reset commands. The I/O subsystem will always return with a time-out error in those cases. The various fields in the command message are defined as follows:

4.3.1 Device Code.

This selects the slave device which is to respond to the command.

4.3.2 Logical Unit Number - LUNO.

This is reserved for use by devices which may contain separately addressable segments on one physical unit (e.g. files on disks). Each currently open file on a device must have a unique non zero code in this byte so that the device may continue to relate the commands to the proper files: the open command provides both the LUNO and the file name. Note that LUNOs are only checked by peripherals that allow multiple files to be open at one time (such as a floppy disk). Peripherals that don't have files or only allow one file to be open at a time are not required to check the LUNO for validity.

4.3.3 Command Code.

This field tells the slave device the nature of the operation to be performed. The following lists the standard code assignments:

Command Code (in Hexadecimal):

00 - open
01 - close
02 - delete open file
03 - read data
04 - write data
05 - restore file
06 - delete
07 - return status
08 - service request enable
09 - service request disable
0B - you are the master
0C - verify read/write operation
0D - format and certify media
0E - catalog directory
0F - set options
10 - transmit break
FE - null operation
FF - bus reset

The specific actions of each of these commands is described in section 5. Certain peripherals may extend this list for device dependent features. Groups of peripherals with similar extensions to this standard will have similar command codes for those extensions. For example video display peripherals would use the same command codes for direct screen access and could have a command code which returns screen characteristics such as numbers of lines and characters per line. Command codes >50->EF are reserved for device dependent commands.

4.3.4 Record Number.

The record number is reserved for future devices which support relative record (random access) files or for devices which extend the standard command code set (e.g. display address for a write screen command on the TV interface). It is ignored when a file is opened as a sequential file. This field must be maintained by the application software for compatibility with possible future random access devices. It should be zeroed

before the open and restore operations for normal access to devices. It should also be incremented by the application after successful read, write, and verify operations. The first record of a file is record 0. Peripherals that do not support random access may ignore this field

4.3.5 Buffer Length.

This field indicates the size of the data buffer for receiving data from a peripheral during the current bus operation. It is used by the master's IOS to check that the length of the returned data does not exceed the buffer size, and may be ignored by the peripheral (Note that if the masters IOS determines that more data is being received than can be put into the buffer, the operation will be aborted, and a bus time-out will result). The data returned by the peripheral in the response message must not exceed the length specified here. This length is exclusive of the data length and return status bytes which form part of the response message (The status byte and data length replace their old fields in the PAB).

4.3.6 Data Length.

This field gives the number of bytes of data which follow in the data field.

4.3.7 Data.

This field contains data to be written to the peripheral device. The use of the data depends on the command code. If the data length field is zero then the data field is not present.

4.4 Response Message

The response message contains the following data

field	bytes
Data length	2
Data	variable
Operation status	1

4.4.1 Data Length.

This field specifies the number of bytes of data which follow in the data field.

4.4.2 Data.

This field contains the data to be returned to the master device; for example on a read data operation. If the data length field is zero then this field will be omitted. Whenever the true data length cannot be determined at the time the length is sent, the data field will be padded with trailing blanks (>20) for display type files, and with trailing zeroes for internal type files (i.e. fixed length internal disk files).

4.4.3 Operation Status.

This field contains a status of the operation. The following lists the assigned response codes in both hexadecimal and decimal:

00	0	normal operation completion
01	1	device/file option error
02	2	attribute error
03	3	file not found error
04	4	file/device not open error
05	5	file/device already open
06	6	device error
07	7	EOF error
08	8	data/file too long error
09	9	write protect error
0A	10	not requesting service (response to poll inquiry)
0B	11	directory full error
0C	12	buffer size error
0D	13	unsupported command error
0E	14	file not opened for write
0F	15	file not opened for read
10	16	data error (checksum failure in device)
11	17	file type (relative/sequential) incorrect or not supported
13	19	append mode not supported
14	20	output mode not supported
15	21	input mode not supported
16	22	update mode not supported
17	23	file type (internal/display) incorrect or not supported
18	24	verify error
19	25	low batteries in peripheral
1A	26	uninitialized media
1B	27	peripheral bus error (timing error)
20	32	media full
FE	254	illegal in slave mode
FF	255	bus time out error

Error codes >50->EF are reserved for device dependent errors.

SECTION 5

I/O Subsystem Access

This section describes the Peripheral Access Block (PAB) and the I/O subsystem calling protocol. Some portions of this section are redundant with previous sections but are included here for completeness.

5.1 Calling Sequence

To call the I/O subsystem the following steps are taken.

1. Place the command code and other information in the PAB.
2. Place the address of the PAB in the floating point accumulator (FAC - two bytes at address >75). The most significant byte of address is placed at >75 and the least significant is placed at >76.
3. Call the I/O subsystem.
4. Test the status byte in the PAB (see below) for an error condition (The first two bytes of FAC will be returned pointing to the status byte of the PAB).

5.2 Peripheral Access Block

The PAB provides information for the I/O subsystem to access a peripheral device. The PAB contains the following fields in the following order. The first field is at the address specified by the PAB pointer in FAC. The other bytes are stored at successively lower addresses. Also, the buffer pointed to by the PAB buffer address is stored from high to low addresses in memory.

field	bytes	to/from
Device code	1	to
Command Code	1	to
Logical Unit Number	1	to
Record number	2	to
Buffer length	2	to
Data length	2	both
Return status	1	from
Buffer address	2	--
* Link to next block	2	-- (only in service request PAB's)
* Service Flag	1	-- (only in service request PAB's)
* DSR Address	2	-- (only in service request PAB's)

NOTE

The last three fields (asterisked) are for use in SRPAB's. If the PAB is not going to be used with a device enabled for service requests, then those fields are not necessary.

The DSR Address is for future expansion of the bus. It is meant to be used to transfer control to a DSR immediately following the successful completion of a service request. It should be left zeroed for compatibility purposes (see section 5.6 - Polling Operation).

5.3 Standard Access

The access to a peripheral device is performed with a sequence of I/O calls. Before using a device it should be opened with an I/O call using the open command. This may be followed by other I/O calls to read or write data or perform other functions. When the application finishes using a device it should issue a close I/O call. This will ensure that any necessary device dependent actions have been performed. For instance, mass storage devices often keep file information in a local RAM memory and update the file directories on the media when the device is closed. The bus reset command code (>FF) will also close all open devices, but may have other undesireable actions as well.

5.4 Command Descriptions

This section describes the PAB setup and device response for the various standard I/O command codes. This includes codes >00(open)->0F(set options), >FE, and >FF (bus reset).

5.4.1 Open - 00.

This command code is used to initiate the use of a device (or file on a device). A device will check access modes and ensure that the device is not already open. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	00
Logical Unit Number	unique non-zero value
Record number	0000
Buffer length	as required (at least 0004)
Data length	as required (0004 returned)
Return status	(returned)
Buffer address	as required

The data buffer contains the following which is sent to the peripheral.

Input buffer length (2 bytes)
(if zero then device returns buffer size)
device attributes (1 byte)
device options (if any)

The peripheral will compare the 'Input buffer length' to its capabilities and return either the requested length, the default length if the requested is zero, or an error if the requested length is unacceptable. The input buffer length is used by the master to determine what size buffer should be allocated by the master for read operations. Thus software could open a device such as an RS-232 peripheral with a one byte input buffer, and write 80 character records out to the RS-232. Each peripheral is required to check the length of data being transferred to it in each write operation, and to generate an error if that data is longer than the maximum write acceptable to the peripheral.

The device attributes byte contains flags used to indicate the access mode of the peripheral. Several bits are unused by the I/O scheme and may be used by the application software as desired. Other bits must be set to zero to allow compatibility with future peripheral protocol enhancements.

The bit definitions for the attributes byte are as follows (bit 0 is the least significant bit):

```
7-6 - access mode
        00 - append mode (write only at end of file)
        10 - output mode (write only)
        01 - input mode  (read only)
        11 - update mode (read or write)
  5 - relative(1)/sequential(0)
  4 - reserved for future use (currently must be zero) *
  3 - internal(1)/display(0) file type
  2 - device dependent use
1-0 - unused (may be used as desired by application)
```

* - if bit 4 is a one, an attributes error (error code 2) must be returned.

The access mode must match the capabilities of the particular device:

1. Append mode specifies that data will only be written (or appended) to the end of a file (Note that a peripheral opened in append mode will return the number of the next record to be written to in the record number field of the returned data in the open).

2. Output mode specifies that data will only be written to the device and the "read data" command will not be used.

3. Input mode specifies that data will only be read from the device and the "write data" command will not be used.

4. Update mode means that data may be both read and written.

For instance, the following peripherals will be expected to support the given access modes:

calculator in slave mode	input, output, update
modem	input, output, update
RS-232	input, output, update
TV-interface	input, output, update
bar code reader	input
tape mass storage	input, output, update*

* update and append only work with the last file on the tape

Bit 2 may be used to specify another device attribute as an

extension to this standard. Software which does not use any special device dependent feature should set this bit to zero.

Note that data transferred on the bus is variable length. If a peripheral determines that data should be formatted as fixed length records, then it is up to that device to perform the formatting.

The device options field is a variable length field which contains device-dependent information relative to setting up the device. Examples are information for the RS-232 baud rate or the file name for a file oriented device. This is normally ASCII (character) data.

Examples:

```
10 OPEN #1,"1.DATAFILE",INPUT
          opens a file on device 1 with name
          'DATAFILE' for input

10 OPEN #1,"10.C=L,S=0",OUTPUT
          opens device 10 for output with
          options 'C=L,S=0'
```

If fewer than 3 bytes of data are sent to the peripheral (buffer length and attributes), then it should error off (i.e. not enough information was providedto justify opening the peripheral).

The response buffer will contain the accepted buffer length, and the record number that the file was opened to. For some devices this record number will be meaningless, so a zero (0) will be returned. This information is always returned when a device or file has been successfully opened. Thus the response message for a successful open will be:

```
Data length          0004
Data          (2 bytes) Max. record length
              (2 bytes) Record position *
Operation status       0
```

* The master device should place the record number in the PAB for the first I/O call that follows.

An unsuccessful open may not return data in the response message. The operation status byte may contain the following error codes:

00 - successful open
01 - device/file option error
02 - error in attributes byte **
05 - file/device already open
06 - device related error
09 - device/file write protected
0B - directory full error
0C - buffer size error
11 - file type (relative/sequential) incorrect or not supported
13 - append mode not supported
14 - output mode not supported
15 - input mode not supported
16 - update mode not supported
17 - file type (internal/display) incorrect or not supported
19 - low batteries in peripheral
1A - uninitialized media
1B - error detected in bus transfer
20 - media full

** - This error code may be used in place of error codes >11-17 when code space prohibits the use of multiple errors.

5.4.2 Close - 01.

This command terminates the use of a device. Depending on the device this command may be used to clean up internal data (e.g. write an end of file) or may be effectively ignored. In general a close command must be sent between using a device and another open command. The data length for the close command will be zero (no command buffer is transmitted). The PAB should be set up as follows:

field	data
Device code	as required
Command Code	01
Logical Unit Number	as in open
Record number	don't care
Buffer length	don't care
Data length	0000 (0000 returned)
Return status	(returned)
Buffer address	don't care

The response message will only contain a status byte and a zero data length (two bytes). The error status indications are;

00 - device or file closed
04 - device or file never opened (optional)
06 - device related error
08 - data/file too long error

5.4.3 Delete open file - 02.

Sometimes, it is desired to delete a file upon completion of I/O to the file (such as reading some data and deleting the old file). This command may be used in place of the close command, and causes the open file to be deleted. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	02
Logical Unit Number	as in open
Record number	don't care
Buffer length	don't care
Data length	0000 (0000 returned)
Return status	(returned)
Buffer address	don't care

The response message will consist of no data (data length = 0) and the return status. The following error status indications may occur:

00 - file deleted
04 - file not open
06 - device error
09 - write protect error
0D - command not supported

5.4.4 Read Data - 03.

This command is used to request data from a device. The command message will contain a zero-length data field. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	03
Logical Unit Number	as in open
Record number	as required
Buffer length	as determined in open
Data length	0000 (length returned)
Return status	(returned)
Buffer address	as required

The response message will contain the data requested. The record number field should be incremented by the application after each successful read data call to be compatible with random access devices. The following error status codes may occur:

00 - read successful
02 - attribute error
04 - file/device not open error
06 - device error
07 - EOF error
0C - buffer size error
0D - command not supported
0F - file/device not open for read

5.4.5 Write Data - 04.

This command is used to send data to a peripheral device. The command message will contain the data to be sent to the device. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	04
Logical Unit Number	as in open
Record number	as required
Buffer length	don't care
Data length	record length (0000 returned)
Return status	(returned)
Buffer address	as required

The response message will contain zero-length data and the operation status. As in the read data operation, the application should increment the record number after each successful operation to support random access devices properly. The following error status indications may occur:

00 - write successfully completed
02 - attribute error
04 - file/device not open
06 - Device error
08 - file/data too long
09 - write protect error
0C - buffer size error
0D - command not supported
0E - file/device not open for write

5.4.6 Restore - 05.

This command is primarily used with mass storage devices to position a file to its first record. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	05
Logical Unit Number	as in open
Record number	don't care
Buffer length	don't care
Data length	0000 (0000 returned)
Return status	(returned)
Buffer address	don't care

The following error status indications may be returned:

00 - restore successful
04 - file/device not open
06 - device error
0D - command not supported

5.4.7 Delete - 06.

This command is primarily used to remove data from mass storage devices. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	06
Logical Unit Number	as in open
Record number	don't care
Buffer length	don't care
Data length	as needed (0000 returned)
Return status	(returned)
Buffer address	don't care

The command data buffer will contain the name of the file to be deleted if it is appropriate to do so. The file name must be specified in the same manner as the options in the open statement. There will be no data returned. The following operation error status indications may occur:

00 - file deleted
03 - file not found
05 - file is open error
06 - device error
09 - write protect error
0D - command not supported

5.4.8 Return Status - 07.

This command is used to return device status information. The information is returned in the data buffer. Certain bit fields in the return data are assigned to standard meanings while others are reserved for device dependent extensions. Certain devices may return more bytes of status if the buffer length allows. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	07
Logical Unit Number	as in open or 00
Record number	as in previous call
Buffer length	>= 0001
Data length	0000 (0001 returned)
Return status	(returned)
Buffer address	as required

When the LUNO is given as zero, then general device status will be returned. The device does not have to be open for this command (although mass storage peripherals that only support one open file will return file status when given a LUNO of 00). If the LUNO is non-zero, then status will be given for the open file referenced by the LUNO. The bit fields in the return data are as follows (bit 0 is the least significant bit).

```
  7 - end of file has been reached (1=true,0=false)
  6 - random access supported (1=true,0=false)
  5 - file is protected (1=true,0=false)
  4 - file/device open (1=true,0=false)
3-2 - file/device type
        0 - display type
        1 - storage type (internal)
        2 - data communications
        3 - undefined
1-0 - I/O modes that the file/device can be opened in
        0 - undefined
        1 - read only
        2 - write only
        3 - read/write
```

When the buffer size permits, a data communications type device will include 2 extra bytes indicating the number of bytes that remain in its output buffer (LSB of the length first).

The following operation error status indications may occur:

04 - file/device not open (issued if incorrect LUNO used)

5.4.9 Service Request Enable - 08.

The following SRPAB contents are used to enable a device for service requests. Most fields are not normally used (e.g. file type devices are not expected to support polling). An open call is normally required before this I/O call to initialize device parameters.

field	data
Device code	as required
Command Code	08
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	normally 0000
Return status	(returned)
Buffer address	buffer address for device
Link to next SRPAB	as is
Service Flag	0

NOTE

The application is responsible for resetting the service flag after acknowledging a service request completion. The buffer provided by the application must be separate from that in the open to allow for normal bus operation with that device.

The following error status code may be returned:

00 - command acknowledged
06 - device error
0C - buffer size error
0D - command not supported

5.4.10 Service Request Disable - 09.

To disable a device from polling the application should send an I/O call to disable the device and then remove the SRPAB from the list. The following SRPAB contents are used to disable a device for service requests. Most fields are not normally used.

field	data
Device code	as required
Command Code	09
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	normally 0000
Return status	(returned)
Buffer address	normally don't care
Link to next block	Don't care
Service Flag	Don't care

The following error status code may be returned:

00 - service requests disabled
0D - command not supported

5.4.11 Service Request Poll - 0A.

This command allows a bus master to query a peripheral as to whether it requested service from the master. The PAB should be set up as follows:

field	data
Device code	as required
Command code	0A
Logical Unit Number	don't care
Record number	don't care
Buffer length	as determined in open
Data length	0000 (data length returned)
Return status	(returned)
Buffer address	as required
Link to next SRPAB	as required
Service flag	0 (to allow the poll operation)

The response message will consist of zero or more bytes of data, and the return status. The following error status indications may occur:

0A - not requesting service (unsuccessful poll)
0D - command not supported (unsuccessful poll)

* Any other error codes indicate a successful poll operation, and reflect the reason for the service request.

The IOS directly supports service requests by maintaining the SRPAB pointer and sending the 'Service Request Poll' command whenever a bus interrupt is received. Thus, this message is never actually sent by an application.

A successful response to a service request poll will usually include data that had been received by the peripheral polled. An example is a modem that had received some data, and had interrupted the master to transmit that data to it.

5.4.12 You are the Master - 0B.

This command allows a bus master to transfer control of the bus to another device. The PAB should be set up as follows:

field	data
Device code	as required
Command code	0B
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	0000 (0000 returned)
Return status	(returned)
Buffer address	don't care

The response message will consist of no data (data length = 0) and the return status. The following error status indications may occur:

00 - you are now a slave device
0D - command not supported

5.4.13 Verify read/write operation - OC.

This command allows a program to verify the accurracy of a read or write to or from a device. After a record is read or written, the record is sent back to the peripheral. The returned status will show whether the record was verified. The PAB should be set up as follows:

field	data
Device code	as required
Command code	OC
Logical Unit Number	as in open
Record number	as in READ/WRITE
Buffer length	don't care
Data length	record length (0000 returned)
Return status	(returned)
Buffer address	as required

The response message will consist of no data (data length = 0) and the return status. The following error status indications may occur:

00 - record verifies
04 - file not open
06 - device error
0C - program length mismatch (verify error)
0D - command not supported
10 - data/checksum error
18 - verify error

5.4.14 Format Media - OD.

This command tells a mass storage device to format the media used by the device. The return status will indicate whether the format was successful. The PAB should be set up as follows:

field	data
Device code	as required
Command code	OD
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	0000 (0000 returned)
Return status	(returned)
Buffer address	don't care

There will be no data in the response message. The following return status indications may be returned:

- 00 - operation successful
- 05 - file or device open
- 06 - device error
- 09 - media write protected
- 0D - command not supported

5.4.15 Read Catalog - OE.

This command tells a mass storage peripheral to return basic information about a particular file. It can be used to catalog an entire directory through multiple uses of the command. The PAB should be set up as follows:

field	data
Device code	as required
Command code	OE
Logical Unit Number	don't care
Record number	file number
Buffer length	at least >0012
Data length	0000 (0012 returned)
Return status	(returned)
Buffer address	as required

The data buffer returned will contain the following information:

File number	1 byte
File name	12 bytes
Maximum record length	2 bytes
Number of records	2 bytes
Device dependent flags	1 bytes

The following error status codes may occur:

- 00 - operation successful
- 03 - file not found
- 05 - file already open
- 06 - device error
- 0C - buffer length error
- 0D - command not supported

By incrementing the file number with each call, a complete directory of a device may be obtained. The end of the directory is indicated by error 03 (file not found) being returned.

5.4.16 Set Options - OF.

This command is used to send device options as described in the open command without re-opening the device. The command data buffer will contain the device options which are device dependent. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	OF
Logical Unit Number	as in open
Record number	don't care
Buffer length	don't care
Data length	as required (0000 returned)
Return status	(returned)
Buffer address	as required

The response message will consist of no data (data length = 0) and the return status. The following error status indications may occur.

00 - options changed
01 - error in options
04 - device or file not open
06 - device error
0D - command not supported

5.4.17 Transmit Break - 10.

This command is used to have a data communications device transmit a continuous break (space condition) for .25 seconds. The peripheral will send all the data in its buffer out on the communications lines, then it will set the communications line to send a continuous break, wait a minimum of .25 seconds, and then return the communications lines to their mark condition. After this point the peripheral will release the bus. The PAB should be set up as follows:

field	data
Device code	as required
Command Code	10
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	0000 (0000 returned)
Return status	(returned)
Buffer address	as required

The response message will consist of no data (data length = 0) and the return status. The following error status indications may occur.

00 - break transmitted
04 - device or file not open
06 - device error
0D - command not supported

5.4.18 Null Operation - FE.

When the calculator receives a BAV interrupt and either no devices are enabled for interrupts or the current service flag is set, then a null operation code is sent to all devices. There will usually be no response to this message

field	data
Device code	as required (usually 00)
Command code	FE
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	00

(all other fields not needed)

5.4.19 Reset Bus - FF.

It may sometimes be desired to tell a device (or devices) to close all open files (or devices). This command will usually have no response by devices, they will simply perform the action requested (If they were not open to begin with, then they will do nothing). Thus, all devices will revert to their power up status. The PAB or SRPAB should be set up as follows:

field	data
Device code	as required (usually 00)
Command code	FF
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	00

(all other fields not needed)

* For both the Null operation and Reset bus commands, no response will be forthcoming if device code 00 is used. However, if some other device code is used, some specific devices may initiate a response. The meaning of the response is specific to the device.

5.5 Slave Mode

Five bytes in RAM are used to control the slave mode. These are also used for the polling operation when in the master mode as described below.

Flag	Length	Address(es)
Master/Slave flag	1 byte	>0805
Slave Operation Handler	2 bytes	>0806, >0807
Slave Access Block (SAB) or Service Request Peripheral Access Block (SRPAB) address	2 bytes	>0808, >0809

When the master/slave flag is set to zero the calculator operates in the normal master mode which allows I/O commands as in the previous section and polling operations for service requests as in the next section. When the master/slave flag is set to non-zero then the calculator operates in the slave mode as described in this section. In slave mode, Master Mode Handler accesses will be inhibited. The calculator will not perform polling in slave mode, as another device on the bus is then expected to do it.

When in the slave mode a request from the master device may occur at any time and will cause an INT 1 in the calculator which will enter code in the Slave Operation Handler . This code will receive the command message from the master device placing it in the Slave Access Block (SAB), then formulate and transmit a response.

5.5.1 Recommended SAB Example.

The SAB is quite similar to the PAB except that all locations contain received data and a separate buffer size is provided to ensure that data sent with the command will fit in the buffer.

The SAB:

field	bytes	received
Command code	1	x
Logical Unit Number	1	x
Record number	2	x
Buffer length	2	x
Data length	2	x
Buffer size	2	
Buffer address	2	

All locations except the buffer size and buffer address are changed when the command is received.

The response message block exactly mimics the response message as follows.

field	bytes
Data length	2
Data	data length
Operation status	1

5.6 Polling Operation

When operating in the master mode the calculator can be set up to poll one or more devices when a service request is issued. A service request (BAV goes low) will cause an interrupt in the software being run. When this interrupt occurs the I/O subsystem performs a device poll if a device is enabled for service requests. If the device was not the one that interrupted the application, then the SRPAB pointer will be changed to point to the next Service Request Peripheral Access Block and control will be passed back to the application. If the device poll was successful, then the data, which will be in the SRPAB data buffer, will be passed back to the application, and the Service flag will be set.

If the SRPAB has a non zero DSR (Device Service Routine) address, then control will be passed to the DSR. The Device Service Routine can then process the data and return to the IOS, which will restore registers A, B, and >66 through >78, and then pass control back to the application. If the most significant byte of the DSR address is zero then control will be passed directly back to the interrupted application - that application can then test for a completed poll by periodically checking the SRPAB service flag. DSR's can use registers >66 through >78, and the A and B registers. If more registers are needed, then they will have to be saved while in the DSR, and restored upon leaving.

```
PROCEDURE SERVICE_REQUEST_HANDLER;
 BEGIN
  IF SRPAB_POINTER <> 0 THEN
   BEGIN
    IF SRPAB.SERVICE_FLAG = 0 THEN {DON'T SERVICE AN INTERRUPT IF   }
                                   {THE APPLICATION HASN'T SERVICED}
                                   {THE LAST ONE                    }
     BEGIN
      SRPAB.COMMAND_CODE:=#0A;
      SRPAB.DATA_LENGTH:=0;
      CALL MASTER_MODE_HANDLER;
      IF STATUS <> 10 THEN          {CHECK TO SEE IF CORRECT DEVICE }
       BEGIN
        SRPAB.SERVICE_FLAG:=1;
        IF SRPAB.DSR_ADDRESS <> 0 THEN
         CALL (DSR_ADDRESS);        {EXECUTE THE DSR - THEN RETURN  }
       END;
     END
```

```
    ELSE
     SEND NULL OPERATION MESSAGE;
    SRPAB_POINTER:=SRPAB.NEXT_LINK; { WE'LL POLL THE NEXT DEVICE   }
   END;                             { NEXT TIME                    }
 END;
```

* The interrupt vector handler will save the necessary registers to assure that none are changed by the handling of the interrupt.

For a description of the SRPAB, see section 5.2. (It is a PAB with the three noted fields at the end of the PAB - DSR dispatching is not supported in the current implementation). The peripheral requesting service will return a device status, and may return a data buffer. The return status byte will give a reason code for the service request. The data will be device dependent (the amount sent will depend on the buffer size given when the device was enabled for service requests).

If a device poll is unsuccessful, a 'not requesting service' error code is returned, then the polling software will load the SRPAB pointer from the link in the current block and return to the application with a return from interrupt instruction.

To set up a device for polling the SRPAB is added to the poll list and the device is enabled to request service with a "service request enable" command in a normal I/O call using the SRPAB created for the device. When an SRPAB is added to or deleted from the list the interrupts should be disabled to prevent a service request interrupt while the links are being changed. The following sequence of operations will successfully add an SRPAB to the list.

```
DISABLE_INTERRUPTS;
IF SRPAB_POINTER=0 THEN BEGIN
  LINK(NEW):=NEW;
  SRPAB_POINTER:=NEW;
END
ELSE BEGIN
  LINK(NEW):=LINK(SRPAB_POINTER);
  LINK(SRPAB_POINTER):=NEW;
END;
ENABLE_INTERRUPTS;
```

The following SRPAB contents are used to enable a device for service requests. Most fields are not normally used (e.g. file type devices are not expected to support polling). An open call is normally required before this I/O call.

field	data
Device code	as required
Command Code	08
Logical Unit Number	don't care
Record number	don't care
Buffer length	as required
Data length	normally 0000
Return status	(returned)
Buffer address	buffer address for device
Link to next SRPAB	as is
Service Flag	0

NOTE

The application is responsible for resetting the service flag after acknowledging a service request completion. The buffer provided by the application must be separate from that in the open to allow for normal bus operation with that device.

The following error status code may be returned:

06 - device error
0C - buffer size error
0D - command not supported

To disable a device from polling the application should send an I/O call to disable the device and then remove the SRPAB from the list. The following SRPAB contents are used to disable a device for service requests. Most fields are not normally used.

field	data
Device code	as required
Command Code	09
Logical Unit Number	don't care
Record number	don't care
Buffer length	don't care
Data length	normally 0000
Return status	(returned)
Buffer address	normally don't care
Link to next block	Don't care
Service Flag	Don't care

The following error status code may be returned:

OD - command not supported

The following sequence will successfully delete an SRPAB from the list. Note that it does not allow for error conditions such as the SRPAB not appearing in the list or the SRPAB_POINTER=0.

```
{ OLD = address of SRPAB to be deleted }
DISABLE_INTERRUPTS;
WHILE LINK(SRPAB_POINTER)<>OLD DO
  SRPAB_POINTER:=LINK(SRPAB_POINTER);
LINK(SRPAB_POINTER):=LINK(OLD);
IF SRPAB_POINTER=OLD THEN SRPAB_POINTER:=0;
ENABLE_INTERRUPTS;
```

NOTE

If a device that has been enabled for interrupts is going to be accessed as a normal slave device while interrupts are enabled, a PAB must be established for that device (separate from the SRPAB) for master mode accesses. Otherwise, the integrity of the SRPAB will be threatened by a device interrupt while preparing for the I/O operation. Interrupts should not be disabled during preparation of a PAB or SAB to prevent severe performance degradation of the system.

SECTION 6

Notes

6.1 Peripheral Prompts

No provision is made for peripherals which must (or would like to) prompt the user. However, provision has been made to allow extensions to the language to be made to support specific peripherals. Thus, special commands to support a device could be added to make the system more 'user friendly'.

6.2 Device Options

The device options are included in the open or set options I/O calls. this data will be represented in ASCII in order to allow friendly statements such as:

```
OPEN #1,"100.MYFILE",UPDATE    for a file on a disk
OPEN #1,"3.B=300,P=O",OUTPUT   for RS232 at 300 baud odd parity
```

The open buffer would contain the text within the quotes after the decimal point and would be in ASCII.

6.3 Notes on the Record Number

The record number field is primarily used by devices that support random access files (i.e. disk drives). Such a device would use the record number field to determine which record to next perform I/O on (if the file was opened for random access). To support this type of access, the application running in the master must increment the record number for each successful read or write operation so that the correct record number is always available to the peripheral if it needs it.

SECTION 7

Bus Transfer Examples

7.1 Example of Open Command

The following example gives an open statement in BASIC, then shows how the I/O transfer would occur that corresponds to that statement:

```
100 OPEN #1,"20.B=4800,P=O",OUTPUT
```

Opens device 20 as file 1 (the file number in BASIC is used as the LUNO) for output. Options are B=4800 (4800 baud) and P=O (Odd parity).

Command:

Transmitted PAB		Transmitted Data	
>14	Device number	>00	,
>00	Command code	>00	P
>01	LUNO	>80	=
>00	Record number	B	O
>00		=	
>04	Buffer length	4	
>00		8	
>0D	Data length	O	
>00		O	

Response:

>04	Data length
>00	
>50	Data - Maximum Record Length
>00	
>00	Data - Record Number
>00	
>00	Status

Note that the Input buffer length in the open was zero, so the peripheral returned the default buffer length that it wanted. BASIC would then use this information for allocating an 80 byte buffer (the default was >50 or decimal 80) for all subsequent I/O operations with that peripheral until it is closed (note that the peripheral would accept writes of longer than 80 bytes, but would transmit a maximum of 80 characters at a time to the master).

7.2 Example of a Read Command

The following example gives an example of an input statement in BASIC, then shows how the I/O transfer would occur that corresponds to that statement:

```
510 INPUT #1,A$
```

Inputs a string variable from file number one (for the purpose of this example assume that the device opened in the above open statement is the device being input from).

Command:

Transmitted PAB		Response	
>14	Device number	>05	Data length
>03	Command code	>00	
>01	LUNO	>32	Data - ASCII 27295
>00	Record number	>37	
>00		>32	
>50	Buffer length	>39	
>00		>35	
>00	Data length	0	Status
>00			

Note that no data was transmitted in this example, and 5 bytes of data were returned to the master (the characters '27295'). This data will be stored in the 80 byte buffer allocated by BASIC, and then processed by the INPUT routine.