Keeping an ADSP Connection Maintaining communications

Once you have established a connection end and opened a connection, you must be able to send and receive data over the connection. You can use the routines in this section to

- determine the status of a connection
- read bytes from the connection end's receive queue
- write bytes to the connection end's send queue and transmit them to the remote connection end
- send an attention message to the remote connection end
- discard all data that has been sent but not yet delivered, and reset the connection

The .DSP Driver implements the special communications protocol called the **AppleTalk DSP (ADSP)**. You send commands to **ADSP** and obtain information about **ADSP** by executing **The .DSP Driver** routines described in this section.

Each routine is implemented as a call to the **Device Manager**'s **PBControl** function, as follows:

```
OSErr       PBControl(thePBptr, async);
ParmBlkPtr  thePBptr;   address of a parameter block structure
Boolean     async;      0=await completion; 1=immediate return
            returns     Error Code; 0=no error
```

thePBptr is a pointer to the parameter block used by the **PBControl** function for .DSP routines

async is a Boolean that specifies whether the function is to execute synchronously or asynchronously. Set the async parameter to TRUE to execute the function asynchronously.

The parameter block is shown in the **DSPParamBlock** . The parameters used with each function are described in this section.

For a general discussion of the use of **ADSP**, see **Using ADSP** in the section **AppleTalk DSP (ADSP)**.

dspStatus

Parameter block

Out-In	Name	Type	Size	Offset	Description
←	ioResult	short	2	16	Result code
→	ioCRefNum	short	2	24	Driver reference number
→	csCode	short	2	26	Always dspStatus
→	ccbRefNum	short	2	32	Reference number of **CCB**
←	statusCCB	long	4	34	Pointer to **CCB**
←	sendQPending	short	2	38	Bytes waiting to be sent or acknowledged
←	sendQFree	short	2	40	Available send queue in bytes

←	recvQPending	short	2	42	Bytes waiting to be read from queue
←	recvQFree	short	2	44	Available receive queue in bytes

The dspStatus routine returns the number of bytes waiting to be read and sent and the space available in the send and receive queues. This routine also returns a pointer to the **CCB**, which contains information about the state of the connection end and about connection events received by the connection end. The **CCB** is described in **Connection Control Block** .

The ioResult parameter returns the result of the routine. If you call the routine asynchronously, the routine sets this field to 1 as soon as it begins execution, and it changes the field to the actual result code when it completes execution. The ioCRefNum parameter is the driver reference number returned by the **OpenDriver** function. You must specify this number every time you call **The .DSP Driver**. The csCode parameter is the routine selector; it is always dspStatus for this routine. The ccbRefNum parameter is the **CCB** reference number that was returned by the dspInit routine. The statusCCB parameter returns a pointer to the **CCB**.

The sendQPending parameter indicates the number of bytes of data in the send queue, including 1 byte for each end-of-message (EOM) indicator in the send queue. (**ADSP** counts 1 byte for each EOM, even though no actual data corresponds to the EOM indicator.) The send queue contains all data that has been sent to **ADSP** for transmission and that has not yet been acknowledged. Some of the data in the send queue might have already been transmitted, but **ADSP** retains it in the send queue until the remote connection end acknowledges its receipt in case the data has to be retransmitted. The sendQFree parameter indicates the number of bytes available in the send queue for additional data.

The recvQPending parameter indicates the number of bytes in the receive queue, including 1 byte for each EOM if the eom bit is set in an **ADSP** packet header. The receive queue contains all of the data that has been received by the connection end but not yet read by the connection end's client. The recvQFree parameter indicates the number of bytes available in the receive queue for additional data.

Result codes

noErr	(0)	No error
errRefNum	(-1280)	Bad connection reference number

dspRead

Parameter block

Out-In	Name	Type	Size	Offset	Description
←	ioResult	short	2	16	result code
→	ioCRefNum	short	2	24	driver reference number
→	csCode	short	2	26	always dspRead
→	ccbRefNum	short	2	32	reference number of **CCB**
→	reqCount	short	2	34	requested number of bytes
←	actCount	short	2	36	actual number of bytes read
→	dataPtr	long	4	38	pointer to data buffer
←	eom	char	1	42	1 if end-of-message; 0 otherwise

The dspRead routine reads bytes from the connection end's receive queue and places them in a buffer that you specify. You can continue to read bytes as long as data is in the receive queue, even after you have called the dspClose routine

or after the remote connection end has called the dspClose or dspRemove routine. The dspRead routine completes execution when it has read the number of bytes you specify or when it encounters an end-of-message (that is, the last byte of data in an **ADSP** packet that has the eom bit set in the packet header).

You can call the dspStatus routine to determine the number of bytes remaining to be read from the read queue, or you can continue to call the dspRead routine until the actCount and eom parameters both return 0.

The ioResult parameter returns the result of the routine. If you call the routine asynchronously, the routine sets this field to 1 as soon as it begins execution, and it changes the field to the actual result code when it completes execution. The ioCRefNum parameter is the driver reference number returned by the **OpenDriver** function. You must specify this number every time you call **The .DSP Driver**. The csCode parameter is the routine selector; it is always dspRead for this routine. The ccbRefNum parameter is the **CCB** reference number that was returned by the dspInit routine.

You specify the number of bytes to read with the reqCount parameter, and you use the dataPtr parameter to provide a pointer to the buffer into which **ADSP** should place the data. **ADSP** returns the actual number of bytes read in the actCount parameter. If the last byte read constitutes an EOM, **ADSP** sets the eom parameter to 1.

If either end closes the connection before you call the dspRead routine, the command reads whatever data is available and returns the actual amount of data read in the actCount parameter. If the connection is closed and there is no data in the receive queue, the dspRead routine returns the noErr result code with the actCount parameter set to 0 and the eom parameter set to 0.

Result codes

noErr	(0	No error
errFwdReset	(-1275)	Read terminated by forward reset
errState	(-1278)	State isn't open, closing, or closed
errAborted	(-1279)	Request aborted by dspRemove or dspClose routine
errRefNum	(-1280)	Bad connection reference number

dspWrite

Parameter block

Out-In	Name	Type	Size	Offset	Description
←	ioResult	short	2	16	result code
→	ioCRefNum	short	2	24	driver reference number
→	csCode	short	2	26	always dspWrite
→	ccbRefNum	short	2	32	reference number of **CCB**
→	reqCount	short	2	34	requested number of bytes
←	actCount	short	2	36	actual number of bytes written
→	dataPtr	long	4	38	pointer to data buffer
→	eom	char	1	42	1 if end-of-message; 0 otherwise
→	flush	char	1	43	1 to send data now; 0 otherwise

The dspWrite routine writes bytes into the connection end's send queue. The send queue contains all data that has been sent to **ADSP** for transmission and that has not yet been acknowledged. Some of the data in the send queue might have already been transmitted, but **ADSP** retains it in the send queue until the remote connection end acknowledges its receipt in case the data has to be retransmitted. The dspWrite routine completes execution when it has copied all

of the data from the data buffer into the **ADSP** send queue.

ADSP transmits the data in the send queue when the remote connection end has room to accept the data and one of the following conditions occurs:

- You call the dspWrite routine with the flush parameter set to a nonzero number.
- The number of bytes in the send queue equals or exceeds the blocking factor. (You use the sendBlocking parameter to the dspOptions routine to set the blocking factor.)
- The send timer expires.
- a connection event requires that the local connection end send an acknowledgment packet to the remote connection end.

The ioResult parameter returns the result of the routine. If you call the routine asynchronously, the routine sets this field to 1 as soon as it begins execution, and it changes the field to the actual result code when it completes execution. The ioCRefNum parameter is the driver reference number returned by the **OpenDriver** function. You must specify this number every time you call **The .DSP Driver**. The csCode parameter is the routine selector; it is always dspWrite for this routine. The ccbRefNum parameter is the **CCB** reference number that was returned by the dspInit routine.

You specify the number of bytes to write with the reqCount parameter, and you use the dataPtr parameter to provide a pointer to the buffer from which **ADSP** should read the data. The dspWrite routine returns the actual number of bytes written in the actCount parameter. If the last byte written constitutes an EOM, set the eom parameter to 1. You can also set the reqCount parameter to 0 and the eom parameter to 1 to indicate that the last byte you sent the previous time you called the dspWrite routine was the end of the message. The high-order bits of the eom parameter are reserved for use by **ADSP**; you must leave these bits equal to 0.

You can set the reqCount parameter to a value larger than the size of the send queue. If you do so, the dspWrite routine writes as much data as it can into the send queue, sends the data and waits for acknowledgment, and then writes more data into the send queue until it has written the amount of data you requested. In this case, the routine does not complete execution until it has finished writing all of the data into the send queue.

Set the flush parameter to 1 to cause **ADSP** to immediately transmit any data in the send queue that has not already been transmitted. Set the flush parameter to 0 to allow data to accumulate in the send queue until another condition occurs that causes data to be transmitted. The high-order bits of the flush parameter are reserved for use by **ADSP**; you must leave these bits equal to 0.

Result codes

noErr	(0)	No error
errState	(-1278)	Connection is not open
errAborted	(-1279)	Request aborted by dspRemove or dspClose routine
errRefNum	(-1280)	Bad connection reference number

dspAttention

Parameter block

Out-In	Name	Type	Size	Offset	Description
←	ioResult	short	2	16	Result code
→	ioCRefNum	short	2	24	Driver reference number
→	csCode	short	2	26	Always dspAttention
→	ccbRefNum	short	2	32	Reference number of **CCB**
→	attnCode	short	2	34	Client attention code
→	attnSize	short	2	36	Size of attention data in bytes
→	attnData	long	4	38	Pointer to attention data

The dspAttention routine sends an attention code and an attention message to the remote connection end. Attention codes and attention messages can have any meaning that your application and the application at the remote connection end both recognize. The purpose of attention codes and messages is to allow clients of **ADSP** to send messages outside the normal data stream. For example, if a connection end on a mainframe computer is connected to several connection ends in Macintosh computers being used as remote terminals, the mainframe computer might wish to inform the remote terminals that all connections will be terminated in ten minutes. The mainframe application could send an attention message to each of the remote terminals informing them of this fact, and the terminal emulation programs in the Macintosh computers could then display an alert message on the screen so that the users could prepare to shut down.

The ioResult parameter returns the result of the routine. If you call the routine asynchronously, the routine sets this field to 1 as soon as it begins execution, and it changes the field to the actual result code when it completes execution. The ioCRefNum parameter is the driver reference number returned by the **OpenDriver** function. You must specify this number every time you call **The .DSP Driver**. The csCode parameter is the routine selector; it is always dspAttention for this routine. The ccbRefNum parameter is the **CCB** reference number that was returned by the dspInit routine.

The attnCode parameter is the attention code that you wish to send to the remote connection end. You can use any value from 0x00000 through 0x0EFFF for the attention code. The values 0x0F000 through 0x0FFFF are reserved for use by **ADSP**. The attnSize parameter is the size in bytes of the attention message you wish to send, and the attnData parameter provides a pointer to the attention message. The attention message can be any size from 0 through 570 bytes. There are no restrictions on the content of the attention message.

Result codes

noErr	(0)	No error
errAttention	(-1276)	Attention message too long
errState	(-1278)	Connection is not open
errAborted	(-1279)	Request aborted by dspRemove or dspClose routine
errRefNum	(-1280)	Bad connection reference number

dspReset

Parameter block

Out-In	Name	Type	Size	Offset	Description
←	ioResult	short	2	16	Result code
→	ioCRefNum	short	2	24	Driver reference number

→	csCode	short	2	26	Always dspReset
→	ccbRefNum	short	2	32	Reference number of **CCB**

The dspReset routine causes **ADSP** to discard all data in the send queue, all data in transit to the remote connection end, and all data in the remote connection end's receive queue that the client has not yet read. This process is known as a *forward reset*. **ADSP** then resynchronizes the connection. You can determine that your connection end has received a forward reset and has discarded all data in the receive queue by checking the eFwdReset flag in the userFlags field of the **CCB**. The **CCB** is described in **Connection Control Block**.

The ioResult parameter returns the result of the routine. If you call the routine asynchronously, the routine sets this field to 1 as soon as it begins execution, and it changes the field to the actual result code when it completes execution. The ioCRefNum parameter is the driver reference number returned by the **OpenDriver** function. You must specify this number every time you call **The .DSP Driver**. The csCode parameter is the routine selector; it is always dspReset for this routine. The ccbRefNum parameter is the **CCB** reference number that was returned by the dspInit routine.

Result codes

noErr	(0)	No error
errState	(-1278)	Connection is not open
errAborted	(-1279)	Request aborted by dspRemove or dspClose routine
errRefNum	(-1280)	Bad connection reference number