Chapter 3

MULTIPOINT

3.1 Introduction

A multipoint connection normally consists of the following:

The PTS system is connected, via a modem, to a telephone line (normally leased, but may be switched) and via another modem to another computer at the other end of the line. As far as the CREDIT programmer is concerned, it is not important what type of computer is connected at the other end; it may be another PTS system, or a large mainframe.

In addition, the Monitor software (driver) is designed to look after the protocol that is being used to communicate with the other system. This means that the CREDIT application only has to set up messages that it wishes to transmit, for example, and the driver will ensure that the correct control characters are added, and that the message is acknowledged by the other system, and so forth.

However, the difference between point-to-point connections and multipoint is that in the latter case each task may identify itself to the other system, by means of the sub-addressing feature, and may send and receive messages independently of the other tasks. In effect, the tasks appear to have their own DC line, although in reality they are of course sharing one line.

Note that this does not always apply; it is possible with some drivers to choose not to use the sub-addressing feature. In that case one task identifies the entire DTE to the system at the other end of the line, and any messages sent by any task will be identified by the DTE-address only. Any messages sent to the DTE will be passed to whichever task issued the least recent read request. If no read request is outstanding, the message will be passed to the task that next issues a read request.

It is important to realise that in the first situation, using sub-addresses, it is possible that `spontaneous messages´ are received. A spontaneous message is a message that contains an incorrect sub-address, or that is addressed to a task that does not currently have a read request outstanding. In this case, different options may be available as follows:

- The Monitor may be generated to exclude `message passing´. This means that any message received in which the DTE address is recognized, but the sub-terminal address is not known, and any message that is received when no task has an outstanding read request, will be discarded.

- The Monitor may be generated to include `message passing´. In this case one of the tasks in the application may identify itself as being the task to receive spontaneous messages. The task can identify the sub-address to which the message was addressed by means of one of the data items used in the read request. It is possible for this task to logically disconnect itself from the line, and for another task to then take over the handling of spontaneous messages, by connecting itself and identifying itself as the (new) spontaneous message handler. However, there may only be one task at a time that is assigned to handle such messages.

The Connect Active instruction is normally not used in multipoint connections, where the PTS computer is a secondary station. The general sequence of events that must be controlled by the application are as follows:

Opening the physical connection (by one of the tasks)
Establishing the logical connection
Transmitting messages (after a Poll by the primary station)
Receiving messages
Closing the logical connection
Closing the physical connection (by one of the tasks, if no longer required).

Any task may perform the opening of the communication, and then each task that is to use it must issue a connection request before it can send or receive messages, whether or not the sub-addressing feature is being used.

The application may set a timeout value for CONNECT and RECEIVEoperations requested, by means of the DSC instruction with Control code X`0B´, in the same way as described in Section 2.8.4.

In addition, the DSC instruction can be used for handling status and statistical information, as described in section 3.8.

In the following sections, the CREDIT statements for carrying out the communication over the line are described, with the exception of that for setting timeout.

3.2 Opening the Communication Link

The OPEN .DC instruction is used to perform the physical connection of the computer to the communication line, since after IPL the DTE is considered to be inactive until this request is succesfully completed.

The OPEN instruction may be issued by any of the application tasks, and it is the responsibility of the application to ensure that the OPEN has been successfully completed before any of the tasks issue a CONNECT instruction. This is normally achieved by the setting and clearing of a boolean switch.

The instruction may be issued with Wait or No Wait, as required by the application.

One parameter is required for this instruction, to specify the local DTE-address, as defined during the system design.

The Condition Register will be set to one of the following values as a result of this instruction:

CR Value	Meaning
0	OPEN successful
2	Error condition

If the request is successful, the DTE is considered to be in Disconnected mode (i.e. Active and physically connected, but not logically ready for transmission). Before any messages can be sent or received, the line must be logically connected, as described in the following section.

If an error occurs, the status word can be fetched by means of the XSTAT instruction. The meanings of the bits in the status word are given in the chapter for the appropriate driver in module M15A, Data Communication Driver Reference Manual.

| OPEN .DC |

Example of the OPEN instruction

```
OPEN  .DC,DSLIN,DTEAD
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

DTEAD This is a binary data item, containing the local DTE-address. The adddress must be contained in the right byte and the left byte must be set to null (X'00').

| OPEN .DC |

3.3 Logically Connecting the Line

The CONNECT .PAS statement is used to perform the logical connection of the DC line, as follows:

After opening the communication line, it is the responsibility of all the tasks that are to use the line to perform a logical connection, to inform the system at the other end of the line of their sub-addresses. In the case where sub-addressing is not used, the tasks must still each issue a logical connection request, although in this case the sub-address specified will not be significant.

The CONNECT .PAS (passive) instruction is used to perform the logical connection of the tasks to the DTE.

The instruction may be used with Wait or No Wait, as required by the application.

Four parameters are required, to specify the local DTE-address, the sub-address and its length, and the option to indicate whether this task is to receive spontaneous messages.

The Condition Register will be set as a result of this instruction to one of the following values:

CR Value	Meaning
0	CONNECT successful
2	Error condition

In the case of an error, the status word can be fetched from the driver with the XSTAT instruction. The meanings of the bits in this word are given in the chapter for the driver concerned in module M15A, Data Communication Driver Reference Manual.

Example of the CONNECT .PAS instruction

```
CONNECT .PAS,DSLIN,DTEAD,DUM,DISCOP,SUBAD,SUBL
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

DTEAD This is a binary data item, containing the local DTE-address in the rightmost byte, and X'00' in the lefmost byte.

CONNECT .PAS

DUM This is a binary data item, included for syntax reasons. Its contents are not significant.

DISCOP This is a binary data item, containing the value X`00´ or X`40´ to define whether this task is to receive spontaneous messages. A value of X`40´ indicates that this task is to receive such messages.

SUBAD This is a string data item, two bytes long, containing the sub-address associated with this task in the second eight bits, and the first eight bits set to X`00´. In the case where the sub-addressing feature is not included, the contents of this data item are not significant.

SUBL This is a binary data item, containing the length of the string which contains the sub-address, in bytes (X`0002´).

| CONNECT .PAS |

3.4 Sending a Message

To send a message on the line, after performing the correct sequence of OPEN and CONNECT .PAS instructions, the SEND instruction is used.

The SEND instruction requires three parameters, to specify the data item containing the message, the length of the message, and (optionally) an indicator to discard pending messages, as follows:

If a message has been received for this task that has not yet been read by means of the RECEIVE instruction, this fact is reported by the setting of bit 7 in the return status. Note, however, that this does not affect the contents of the Condition Register at completion of the request. If the option is specified to discard pending messages, then this bit will not be set, and any outstanding messages addressed to this task will be discarded before execution of the instruction.

The instruction may be issued with or without Wait, as required by the application.

The Condition Register will be set as a result of this instruction to one of the following values:

CR Value	Meaning
0	SEND successful
2	Error condition

In the case of the Condition Register being set to zero, it is stilladvisable to fetch the status word via the XSTAT instruction, since, if bit 7 is set, this indicates that the SEND has been successfully completed, but that a message has been received for this task, and therefore a RECEIVE should be issued as soon as possible to collect the message. This does not apply if the option is set to discard pending received messages, as described previously.

In the case of an error, the status word can be fetched from the driver with the XSTAT instruction. The meanings of the bits in this word are given in the chapter for the driver concerned in module M15A, Data Communication Driver Reference Manual.

SEND

Example of the SEND instruction

```
SEND  .NW,DSLIN,MESBUF,MESLEN,OPTNS
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

MESBUF This is a string data item, containing the message to be sent on the line. The driver will add the necessary framing characters before transmission.

MESLEN This is a binary data item, containing the length of the message to be sent, in number of characters. Thus the string data item may be longer than the actual message, as the number of characters sent is determined by the value in this data item.

OPTNS This is a binary data item containing the value X`0020´ to indicate that pending messages are to be discarded (otherwise X`0000´).

| SEND |

3.5 Receiving a Message

To receive a message on the line, after performing the correct sequence of OPEN and CONNECT .PAS instructions, the RECEIVE instruction is used.

The RECEIVE instruction requires three parameters, to specify the data item to contain the message, the length of this data item, and, in the case of refused requests or spontaneous messages, to receive the status or sub-address respectively.

The instruction may be issued with or without Wait, as required by the application.

The Condition Register will be set as a result of this instruction to one of the following values:

CR Value	Meaning
0	RECEIVE successful
2	Error condition

In the case of an error, the status word can be fetched from the driver with the XSTAT instruction. The meanings of the bits in this word are given in the chapter for the driver concerned in module M15A, Data Communication Driver Reference Manual.

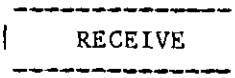
In the case of bit 7 being set in the status word, this indicates that at least one further message is waiting for this task, and another RECEIVE must be issued as soon as possible, in order to release the buffer. Note that, if only this bit is set, the Condition Register will be set to zero, thus it is recommended to issue the XSTAT instruction in any case.

Example of the RECEIVE instruction

```
RECEIVE .NW,DSLIN,MESBUF,MESLEN,STAT
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

MESBUF This is a string data item, into which the message will be read from the line. The driver will remove any framing characters before placing the message in this data item.



MESLEN This is a binary data item, containing the maximum expected length of the message to be read. Thus the string data item may be longer than the actual message, as the number of characters read is determined by the value in this data item. If the length is shorter than the actual message, an error will be reported via the Condition Register.

STAT This is a binary data item, which, in the case of an error, will contain the status of the connection, as defined in section 3.8.1, or the sub-address to which the message is addressed, in the case of spontaneous message reception.

| RECEIVE |

3.6 Logically Disconnecting the Line

The DISCNCT statement is used to perform the logical disconnection of the DC line, as follows:

After opening the communication line, and performing the logical connection, as described previously, a task may send and receive messages. When transmission has been completed (i.e. the task no longer wishes to use the line), disconnection must take place. Any task that has performed a logical connection may issue a disconnect request, and this must normally be carried out before the CLOSE instruction can be issued to physically disconnect the line, as described in the following section. The DISCNCT instruction can also be used after a timeout on a RECEIVE request, to avoid receiving spontaneous messages.

If the sub-addressing feature is not included, then only the task that performed the Connect needs to issue the Disconnect request.

The instruction may be used with Wait or No Wait, as required by the application.

One parameter is required for this instruction, being the symbolic local sub-address, unless the sub-addressing feature is not included.

The Condition Register will be set as a result of this instruction to one of the following values:

CR Value	Meaning
0	DISCNCT successful
2	Error condition

If successful, the DTE is considered to be in the Disconnected mode, i.e. the only valid instructions that can be issued by this task are a Connect, Close, or Read Status.

In the case of an error, the status word can be fetched from the driver with the XSTAT instruction. The meanings of the bits in this word are given in the chapter for the driver concerned in module M15A, Data Communication Driver Reference Manual.

DISCNCT

Example of the DISCNCT instruction

```
DISCNCT  DSLIN,SUBAD
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

SUBAD This is a string data item, two bytes long, containing the sub-address associated with this task in the second byte, and the first byte set to X'00'. In the case where the sub-addressing feature is not included, this data item is not required.

3.7 Closing the Communication Link

The CLOSE .DC statement is used to perform the closing of the communication link, when no further transmission is to take place, as follows:

After opening the communication line, and performing the logical connection, messages may be sent or received. When all transmission has taken place successfully, and the application no longer requires to send or receive messages, the line must be logically disconnected, as described previously, and then closed by means of this instruction.

The instruction may be used with Wait or No Wait, as required by the application.

An optional parameter may be supplied, to indicate whether this request is to be Conditional or Unconditional, as follows:
- A conditional Close request is only executed if no tasks are logically connected (i.e. all tasks in a system with the sub-addressing feature have successfully issued a Disconnect request.
- An unconditional Close request is always executed; thus if there are still messages waiting in queues, these are discarded, and no task may now use the line.

The Condition Register will be set as a result of this instruction to one of the following values:

CR Value	Meaning
0	CLOSE successful
2	Error condition

If successful, the DTE is considered to be Inactive, until an OPEN instruction is used to re-establish the physical connection.

In the case of an error, the status word can be fetched from the driver with the XSTAT instruction. The meanings of the bits in this word are given in the chapter for the driver concerned in module M15A, Data Communication Driver Reference Manual.

| CLOSE .DC |

Example of the CLOSE instruction

```
CLOSE .DC,DSLIN,DUM,CLOPT
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

DUM This is a binary data item, included for syntax purposes; its contents are not significant.

CLOPT This is a binary data item, containing the value X`0080' if this is to be a Conditional Close, or X`0000' if it is to be Unconditional.

| CLOSE .DC |

3.8 Data Set Control Instructions

As well as sending and receiving messages, the DSC instruction may be used to handle status and statistic information concerning the line, as follows:

- Read status (DSC Control code `07´) enables the application to fetch the current state and type of connection.
- Read/reset statistics (DSC Control code `17´) enables the application to fetch statistical information from the software, providing the function was included during Monitor generation

The use of these instructions is described on the following pages.

3.8.1 Read Status

The DSC instruction to Read Status is used to fetch information as to the status of the connection. The instruction may be issued at any time, and may be issued with Wait or No Wait, as required by the application.

The Control Code for Set Status is X'07', and the binary data item will have bits set to indicate the connection and type, as follows:

Bit 15 indicates the status, and may be:
0 = DTE-DCE interface is Active.
1 = DTE-DCE interface is Inactive.

Bit 11 may be set to:
0 = Data link down
1 = Data link up

Bits 6 & 7 may be set to:
00 = DTE Closed
01 = OPEN in progress
10 = DTE Open
11 = CLOSE in progress

Bits 3 & 4 may be set to:
00 = Task is Disconnected
01 = CONNECT in progress
10 = Task is Connected
11 = DISCONNECT in progress

Example of the DSC instruction for Read Status

```
RSTAT   EQU   X'07'
        DSC   DSLIN,RSTAT,LINEC
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

LINEC This is a binary data item, which, on completion, will have bits set to indicate the status of the connection.

3.8.3 Read/Reset Statistics

The DSC instruction to Read / Reset Statistics enables the application to fetch information from the statistics counters in the software. If required, it must be specified during Monitor generation. Optionally, the application may also request that the statistic counters are to be cleared (set to zero) with this instruction.

The instruction may be issued with Wait or No Wait, as required by the application.

The Control Code for Read/Reset Statistics is X`17', and three data items are required to determine whether the counters are to be cleared, the string into which the information is to be read and the length of the string. The layout of the counters, as they appear in this data item, is shown in the appropriate chapter for the driver used, in module M15A, DC Drivers Reference Manual.

If the counters are to be cleared, the first binary data item must contain the value X`4300', else zeroes.

The string data item must be long enough to contain all the counters that are required by the application, up to a maximum of 62 bytes.

The second binary data item must contain the length in bytes of the information to be read, i.e. if only the first two counters are required, this should contain a value of 4, being two bytes for each counter. In that case, if the option to reset the counters is also set, only the first two counters will be reset.

The Condition Register will be set as a result of this instruction to one of the following values:

CR Value	Meaning
0	Successful
2	Error condition

The only error condition that can arise is that the option to include the statistic counters was not specified during Monitor generation.

DSC

Example of the DSC instruction for Read & Reset Statistics

```
STATS   EQU   X'17'
        DSC   DSLIN,STATS,STOPT,CBUF,CLEN
```

DSLIN This is the data set identifier for the DC line. The TOSS file code must have been defined at Monitor generation, and must have been defined in the application for use in this terminal class.

STOPT This is a binary data item, containing the value X'4300' to indicate that the statistic counters are to be cleared.

CBUF This is a string data item, into which the statistic counter information will be read.

CLEN This is a binary data item, containing the length of the data from the statistic counters that is required, in bytes.

| DSC |