2.3.1.1.1.4. RES

The RES Reserved character is a reserved entry in the station selection sequence. Currently, the SPI CCP specifies that this entry be set to $20.

2.3.1.1.2. Polling

The control station initiates a POLL request to read data from a tributary station. A POLL request consists of an EOT character followed by a station selection supervisory sequence and ending with an ENQ character as follows:

DEVID = #20

Character	Description
<EOT>	Resets the communication link
(DEVID)	Specifies the device type
(ADD)	Specifies the device address
(CMD1)	Specifies the Command #1 code
(CMD2)	Specifies the Command #2 code (bit 0 will be reset to 0)
(RES)	SPI CCP reserved byte, set to $20
<ENQ>	Specifies the end of the POLL supervisory sequence

Table 4, POLL Supervisory Sequence{tc "POLL Supervisory Sequence" /t}

Once the control station sends the POLL supervisory sequence it assumes slave status and expects the polled tributary station to assume master status and begin message transfer.

A tributary station detecting its (DEVID) and (ADD) assumes master status and responds in one of two ways:

1. If the tributary station has a message to send it initiates message transfer. The control station assumes slave status.

2. If the station cannot send a message it sends an <EOT>, terminating its master status. Master status reverts back to the control station which may then retry the establishment procedure up to 2 times before notifying its control program of a communication failure.

If the control station receives an invalid response or no response to the POLL request it terminates by sending an <EOT>. It may retry the establishment procedure up to 2 times before notifying its control program of a communication link failure.

2.3.1.1.3. Selection

The control station initiates a SELECT request to send data to a tributary station. A SELECT request consists of an EOT character followed by a station selection supervisory sequence and ending with an ENQ character as follows:

Character	Description
<EOT>	Resets the communication link
(DEVID)	Specifies the device type
(ADD)	Specifies the device address
(CMD1)	Specifies the Command #1 code
(CMD2)	Specifies the Command #2 code (bit 0 will be set to 1)
(RES)	SPI CCP reserved byte, set to $20
<ENQ>	Specifies the end of the SELECT supervisory sequence

Table 5, SELECT Supervisory Sequence{tc "SELECT Supervisory Sequence" /t}

A tributary station detecting its (DEVID) and (ADD), assumes slave status and sends one of two replies:

1. If the selected station is ready to receive, it sends the following code sequence:

Character	Description
(DEVID)	Repeat of tributary device type
(ADD)	Repeat of tributary device address
(CMD1)	Repeat of command #1 code
(CMD2)	Repeat of command #2 code
(RES)	SPI CCP reserved byte, set to $21
<DLE>	ACK0 positive acknowledgment
<30>	

Table 6, SELECT Positive Acknowledgment Sequence{tc "SELECT Positive Acknowledgment Sequence" /t}

Upon detecting this reply, and if the reply is the same as its initial request (DEVID,ADD,CMD1,CMD2 and RES are the same) the control station proceeds with message transfer.

2. If the selected station is not ready to continue because the command requested is not a valid command the selected station responds with the following code sequence:

Character	Description
(ERR)	SPI CCP Error Code which indicates the nature of the request refusal
<NAK>	Negative acknowledgment

Table 7, SELECT Negative Acknowledgment Sequence{tc "SELECT Negative Acknowledgment Sequence" /t}

If the control station does not receive a response or the control station receives an invalid response (DEVID, ADD, CMD1, CMD2 or

RES do not match or too many characters are received), it may retransmit the selection supervisory sequence up to 2 times before notifying its control program of a communication link failure and then executing a sending station abort.

If the control station receives a negative acknowledgment response ((ERR)<NAK>) indicating that a character may have been garbled during transmission it may retransmit the selection supervisory sequence up to 2 times before notifying its control program of a communication link failure and then executing a terminate function.

2.3.1.1.4. Termination

Once a communication sequence has been established and the message transfer procedure has completed, the master station transmits an <EOT> to indicate that it has no more messages to send. The <EOT> negates the master / slave status of both stations and returns master status to the control station.

2.3.1.2. Message Transfer Procedure

The ANSI x3.28 Subcategory D1 is used as the message transfer procedure for the SPI CCP Data Link Layer protocol. This standard is compatible with the Subcategory 2.4 establishment and termination procedure and provides message independent blocking, cyclic checking, alternating acknowledgments, and transparent heading and text. Message data can be any ASCII value ($00 - $FF). Data can be sent in variable length blocks (subject to a maximum block length which is set by the communication application which uses this protocol as its data link layer) regardless of the content of the data. Each block is acknowledged with an alternating acknowledgment to allow detection of missed or duplicate blocks and the block data is validated using a Cyclic Redundancy Check (CRC) calculation.

2.3.1.2.1. Transmission Blocks

The transmission of blocks is initiated by the master station after an appropriate establishment procedure. If the block is the first block within a message it is started with the code sequence <DLE><SOH>. If the block is not the first block of a message it is started with the code sequence <DLE><STX>.

A block that ends at an intermediate point within the message is ended with the code sequence <DLE><ETB> immediately followed by a cyclic redundancy check sequence (CRC)(CRC). A block which ends at the end of a message is ended with the code sequence <DLE><ETX> immediately followed by a cyclic redundancy check sequence (CRC)(CRC). After the (CRC)(CRC) sequence the master station waits for a reply.

Since any of the 256 combinations of 8 bits may occur in the data, the stations must be able to distinguish between control characters and data. The control characters that can be recognized must always be a character sequence started with a <DLE>. When a <DLE> occurs in the source data stream, the master station inserts an additional <DLE>

so that the data transmitted will contain two successive <DLE> characters.

The slave station must scan the incoming data for <DLE> characters. On the first appearance of a <DLE>, the character immediately following determines which of the following actions to be taken:

1. If the character immediately following is a <SOH>, the slave station enters transparent mode. The data following is considered the message heading. The message heading is terminated by the code sequence <DLE><STX>.

2. If the character immediately following is a <STX>, the slave station enters transparent mode if not already in transparent mode. The data following is considered as message text. The message text is terminated by the code sequences <DLE><ENQ>, <DLE><ETB>, <DLE><ETX>, the slave station then exits transparent mode.

3. If the character immediately following is also a <DLE>, indicating a data character, the slave station discards one <DLE>. The character immediately succeeding is not considered part of any control character sequence. The station continues to receive data.

4. If the character immediately following is either <ENQ>, <ETB>, or <ETX>, the slave station leaves transparent mode and performs the action designated by the control character (completion of a time-out (timer A,B or D) also causes the slave station to leave transparent mode).

5. If the character immediately following is a <SYN>, the slave treats this as an idle character sequence and discards both characters.

6. If the character immediately following is other than <SOH>, <STX>, <DLE>, <ENQ>, <ETB>, <ETX> or <SYN> then it is to be considered invalid and the entire block is to be considered invalid and must be responded to as such.

2.3.1.2.1.1. Control Station Message Format

The control station sends messages when it is executing a SELECT sequence. The following is the control station message format:

Character	Description
<DLE>	Start of Text sequence
<STX>	

(DATA)	Start of data bytes. Can be any value ($00 - $FF). <DLE> data values are replaced with a transparent DLE sequence (<DLE><DLE>). Data can be any length up to the maximum data block size (TDLE's count as a single data character).
<DLE>	End of Block sequence
<ENQ> or <ETB> or <ETX>	Block Abort ending sequence End of intermediate block sequence End of final block sequence
CRC1	MS Byte of CRC calculation
CRC2	LS Byte of CRC calculation

Table 8, Control Station Message Format{tc "Control Station Message Format" /t}

2.3.1.2.1.2. Tributary Station Message Format

A tributary station sends messages when it is executing a POLL request. If the tributary is sending the first block of a message, the following format is used:

Character	Description
<DLE>	Start of Header sequence
<SOH>	
(DEVID)	Repeat of POLL request DEVID
(ADD)	Repeat of POLL request ADD
(CMD1)	Repeat of POLL request CMD1
(CMD2)	Repeat of POLL request CMD2
(RES)	Repeat of POLL request RES
(ERR)	SPI CCP Error Status Byte, set to $20 if no errors, otherwise error bits are set accordingly
<DLE>	Start of Text sequence
<STX>	
(DATA)	Start of data bytes if data is required. Can be any value ($00 - $FF). <DLE> data values are replaced with a transparent DLE sequence (<DLE><DLE>). Data can be any length up to the maximum data block size (TDLE's count as a single data character). If no data is required, then no (DATA) bytes are sent between the Start of Text and End of Block sequences.
<DLE>	End of Block sequence
<ENQ> or <ETB> or <ETX>	Block Abort ending sequence End of intermediate block sequence End of final block sequence
CRC1	MS Byte of CRC calculation

CRC2	LS Byte of CRC calculation

Table 9, Tributary Station First Message Format{tc "Tributary Station First Message Format" /t}

If the tributary is sending a subsequent block of a message, the following format is used:

Character	Description
<DLE>	Start of Text sequence
<STX>	
(DATA)	Start of data bytes if data is required. Can be any value ($00 - $FF). <DLE> data values are replaced with a transparent DLE sequence (<DLE><DLE>). Data can be any length up to the maximum data block size (TDLE's count as a single data character). If no data is required, then no (DATA) bytes are sent between the Start of Text and End of Block sequences.
<DLE>	End of Block sequence
<ENQ> or <ETB> or <ETX>	Block Abort ending sequence End of intermediate block sequence End of final block sequence
CRC1	MS Byte of CRC calculation
CRC2	LS Byte of CRC calculation

Table 10, Tributary Station Subsequent Message Format{tc "Tributary Station Subsequent Message Format" /t}

2.3.1.2.2. Replies

The slave station, upon detecting the code sequence <DLE><ENQ>, <DLE><ETB>, or <DLE><ETX> followed by the (CRC)(CRC) sequence, sends one of two replies:

1. If the transmission block was accepted and the slave station is ready to receive another block, it sends the appropriate <ACKn>. <DLE><31> is used as the reply to the first block received. <DLE><30> is used to reply to the second block received. Subsequent positive acknowledgments alternate between <ACK1> and <ACK0>. Upon detecting the correct <ACKn>, the master station may either transmit the next block, or initiate termination if the last block was ended in <DLE><ETX>.

2. If the transmission block was not accepted and the slave station is ready to receive another block, it sends the code sequence (ERR)<NAK>, where (ERR) indicates why the block was not accepted. Upon detecting the <NAK> the master station initiates retransmission of the last block up to 2 times before notifying its

control program of a communication link failure and initiating a termination function.

The use of a (ERR)<NAK> sequence does not alter the sequence of alternating acknowledgments. The same affirmative reply (<ACK1> or <ACK0>) is used for a successful retransmission as would have been used if the previous transmission of the unaccepted block had been successful.

If the alternating reply indicates that the slave station missed the outstanding block (receipt of <ACK1> instead of <ACK0>, or vice versa), the master station initiates retransmission of that block as if the slave station had returned a <NAK>.

If the master station receives an invalid reply or no reply, it may send a reply request sequence consisting of an <ENQ>. The master station may retry up to 2 times before notifying its control program of a communication link failure and initiating a sending station abort.

Upon receipt of a reply request sequence the slave station must retransmit its last reply. This may be done up to 2 times before notifying its control program of a communication link failure and initiating a terminate function.

2.3.1.3. Abort and Interrupt Procedures

The ANSI x3.28 standard provides various methods of aborting and interrupting communication sessions. The following sections describe these procedures.

2.3.1.3.1. Block Abort

The Block Abort procedure is used when the sending station is in the process of sending a block and is required to terminate the transfer before the normal end of the block such that the receiving station will discard the block.

2.3.1.3.1.1. Application

The Block abort procedure may be used by the sending station when, in the process of sending data, there occurs an indication that the data being sent may not be valid. Such an indication would result from an error condition being detected in the data being sent or of insufficient data being available to send or of excessive amounts of data being available to send. This condition could be the results of a software error or operator error. Block aborts may be used in the message transfer state to cause a temporary text delay after the receipt of the previous acknowledgment if the sending station is not capable of transmitting the text of the next transmission block before the predetermined time out period.

2.3.1.3.1.2. Procedure

The Block Abort Procedure is accomplished by the sending station's ending the block at any time by transmitting the code sequence <DLE><ENQ>. The sending station then halts transmission and waits for a reply from the receiving station. The receiving station detects that the block was ended with the code sequence <DLE><ENQ> rather than with the normal ending code sequence <DLE><ETX> or <DLE><ETB>, whereby the receiving station discards that portion of the block that had been received, and sends a <NAK> with the required (ERR) prefix and remains in the receive state.

Following the receipt of the (ERR)<NAK> response the sending station will reinitiate the transmission of the faulty block or initiate the appropriate termination or recovery procedure. Note that if the faulty block cannot be retransmitted then the sending station must initiate the recovery procedure.

In the case that a (ERR)<NAK> sequence is not received by the sending station, the sending station will time out and then will normally send an <ENQ> code to request a retransmission of the last response . However, due to a limited number of allowable retries (2) the sending station will eventually execute a recovery procedure.

2.3.1.3.2. Sending Station Abort

The Sending Station Abort procedure is used by the sending station to prematurely terminate transmission of a multi block message. After an intermediate block has been sent and a valid acknowledgment has been received (either ACKn or NAK), the sending station can issue a Sending Station Abort to terminate further message transfer.

2.3.1.3.2.1. Application

Sending Station Abort procedures may be used by a sending station when in the process of sending a block of text it determines that it should prematurely terminate transmission to the particular receiving station. Such a determination would be caused by the need of a higher priority message to be transmitted to the same or another station, or the inability of the sending station to continue transmission. Sending station abort procedures may be used following block abort procedures to accomplish a transmission abort condition; that is the sending station prematurely terminates the transmission within a transmission block.

2.3.1.3.2.2. Procedure

Sending station abort procedures are accomplished by the sending station's completing the transmission of a block by sending the code sequence <DLE><ENQ> or <DLE><ETB>. Then upon receipt of the proper acknowledgment code sequence <ACKn> or (ERR)<NAK>, or no acknowledgment, the sending station transmits an <EOT> to terminate the transmission to the particular receiving station. The receiving station detects this sending station abort procedure by recognizing receipt of the <EOT> following the code sequence <DLE><ENQ> or <DLE><ETB> instead of the normal code sequence <DLE><ETX>. Receipt or transmission of <EOT> resets all stations on the link to the control phase in nontransparent mode. At this time the supervisory station (control station) will regain control of the communication link. Any data received by the receiving station during this message transfer prior to the sending station abort procedure will be discarded.

2.3.1.3.3. Termination Interrupt

The Termination Interrupt procedure is used by the receiving station after receipt of a message or transmission block to terminate further transmission by the sending station.

2.3.1.3.3.1. Application

The termination interrupt may be used by the receiving station to cause the transmission to be interrupted because it is not in a condition to receive. Such a condition could exist because the receiving station is off line or has been requested to disallow any communications.

2.3.1.3.3.2. Procedure

The termination interrupt procedures are accomplished by the receiving station transmitting an <EOT> in lieu of one of its normal acknowledgments such as <ACKn> or (ERR)<NAK> or any other valid response. This response indicates a negative acknowledgment of the last transmission and the conclusion of the transmission. This <EOT> resets all stations on the link to the control phase in nontransparent mode. At this time the supervisory station (control station) will regain control of the communication link.

2.3.1.3.4. Reverse Interrupt

The Reverse Interrupt procedure is used by a receiving station to request the sending station to terminate the transmission in progress prematurely in order to facilitate a reversal in the direction of data transfer.

2.3.1.3.4.1. Application

Reverse interrupt procedures may be used by the control station to interrupt its receiving of a "batch" message stream so that it may transmit a priority message to the original sending station or to another station. This function may only be used by the supervisory station (control station).

2.3.1.3.4.2. Procedure

Reverse interrupt procedures may be used by the control station only after reception of a block with a valid cyclic redundancy check (CRC). Reverse interrupt procedures are accomplished by the control station transmitting the code sequence <DLE><3C> in lieu of the normal affirmative acknowledgment <ACKn>. This reply is interpreted as an affirmative acknowledgment by the receiving station to the last transmitted block, and it signals a request by the control station that the sending station terminate the transmission sequence in progress as soon as the sending station is in such a state that it can receive a message without destroying or loosing information that had previously been stored in buffers. The <DLE><3C> code sequence cannot be retransmitted by the control station to successive transmission blocks without transmitting intervening affirmative acknowledgments (<ACKn>). However, successive <DLE><3C> code sequences can be transmitted for a given block in response to <ENQ>'s being received for retry purposes. Upon receipt of the code sequence <DLE><3C> the transmitting station should terminate the transmission by transmitting an <EOT> after it has completed transmitting all data that would prevent it from receiving a message. All transmitting stations must limit the number of blocks that they must transmit to a maximum of 1. The receipt of the code sequence <DLE><3C> as a response to a sending station <ENQ> should be treated as a repeated response if the last valid response received was <ACKn>. The sending station should continue by transmitting the next block or <EOT>. If the last valid response was the code sequence <DLE><3C> then the transmitting station should assume that the last block was received in error and retransmit the last block.

2.3.1.3.5. Temporary Interrupt

The ANSI x3.28 Temporary Interrupt (Wait Acknowledgment) procedure is specified as optional and is not used by the SPI CCP Data Link protocol.

2.3.1.4. Error Detection and Recovery

The ANSI x3.28 standard provides various methods of error detection and recovery. Header data is duplicated in reply messages to ensure the proper station has responded. A CRC calculation is used to protect against lost or garbled data bytes. And various timers are used to protect against lost control characters and no responses. The following sections describe the ANSI x3.28 error detection and recovery procedures in greater detail.

2.3.1.4.1. ERR Error Status Byte

The SPI CCP ERR Error Status byte is used to report error information for refused commands or data blocks. The CPI CCP ERR value is returned in tributary station POLL replies, control station POLL data block acknowledgments, and slave station SELECT replies and acknowledgments. The ERR value is defined as a BIT register where each bit is used to indicate a specific status as follows:

Bit	Description
7 (MS bit)	**Invalid Data** When set to 1, the receiving station has detected invalid data. Data has been refused and discarded.
6	**Command Not Ready** When set to 1, the requested command cannot be executed at this time. They command may be attempted again at a later time.
5	**Set to 1** This bit is always set to 1 to ensure that the ERR value is in the range from $20 - $FF.
4	**Reserved by SPI** This bit is reserved by SPI CCP and must be reset to 0.
3	**Command Not Supported** When set to 1, the received command is not supported by the specified device/address. Command may be retried to determine if request header data was garbled.
2	**Command Not Executed** When set to 1, the requested command was not executed. Typically this bit is set along with one of the error indication bits.
1	**Invalid Preamble (Supervisory Sequence)** When set to 1, the received command supervisory sequence data was invalid. The request should be attempted again.
0 (LS bit)	**Communication Error** When set to 1, a communication error, such as a CRC error or physical link error, has been detected by the receiving station. The request or data block should be resent.

Table 11, ERR Error Status Byte Format{tc "ERR Error Status Byte Format" /t}

2.3.1.4.1.1. Invalid Data Bit

When this bit is set to 1 it means that the data received was received "correctly", but that it was not appropriate for the command used. Reasons for this for example, would be data out of range, too little data, too much data or an undetected CRC error which produced erroneous data. This bit is set during the negative reply to any block where invalid data is detected.

2.3.1.4.1.2. Command Not Ready Bit

When this bit is set to 1 it means that the requested command cannot be executed at this time. The control station should attempt the request again at a later time. No specific time period is specified at the Data Link level.

2.3.1.4.1.3. Command Not Executed Bit

When this bit is set to 1 it means that the command requested was not executed or no action was taken. Reasons for this, for example, would be that the controller has been told not to perform the selected command, that the controller has determined that executing the command would result in an unsafe condition within the selected equipment, that the data was invalid, that the preamble was invalid or that there was a communication link error. This bit is set during the negative reply to a block which contains data that would prevent the command from being executed.

2.3.1.4.1.4. Invalid Preamble Bit

When this bit is set to 1 it means that the preamble was in error, such as when an invalid RES byte or invalid preamble length is received. This condition could result from a communication link error. This bit is set during the negative reply to any block which has an invalid preamble.

2.3.1.4.1.5. Communication Error Bit

When this bit is set to 1 it means that the communication link experienced an error in its communication. This could result from a framing error, a CRC error being detected or a protocol violation. This bit is set during the negative reply to any block in which a communication error was detected.

2.3.1.4.1.6. Command Not Supported Bit

When this bit is set to 1 it means that the requested command is not supported by the controller. Each tributary device will return this bit set in the ERR byte (possibly with other bits set such as Command Not Executed) when an unsupported command is received. If the ERR byte is returned with this bit set, the control station should retry the

POLL or SELECTION to verify that the tributary device properly received the preamble data. This bit is set during the negative reply to any block which has an unsupported command.

2.3.1.4.2. CRC Calculation

The ANSI x3.28 Message Transfer Subcategory D1 requires the use of cyclic redundancy check characters to detect against garbled, lost, or duplicate data bytes. The CRC specified by the SPI CCP Data Link Layer protocol is based on a polynomial of the 16th degree as follows:

$$\text{CRC-16} = X^{16} + X^{15} + X^{2} + 1$$

The CRC is initiated by the occurrence in the data stream of either <DLE><SOH> or <DLE><STX>. The CRC is signaled to stop through the occurrence in the data stream of <DLE><ETB>, <DLE><ETX>, or <DLE><ENQ>. The following table governs the CRC calculation in deciding which characters to process:

Character Sequence	**Included in CRC**	**Not Included in CRC**
<DLE><SYN>	---	<DLE><SYN>
<DLE><SOH>	---	<DLE><SOH>
<DLE><STX>[1]	---	<DLE><STX>
<DLE><STX>[2]	<STX>	<DLE>
<DLE><ETB>	<ETB>	<DLE>
<DLE><ETX>	<ETX>	<DLE>
<DLE><DLE>	<DLE> (one)	<DLE> (one)

Table 12, CRC DLE Sequence Chart{tc "CRC DLE Sequence Chart" /t}
1 - If not preceded in same block by transparent header data
2 - If preceded in same block by transparent header data

The bits of the transmission block or message that are included in the CRC calculation, which may include both information as well as control characters, correspond to the coefficients of a message polynomial having terms from x(n-1) down to x16, where x = the total number of bits in the block or message. The resulting polynomial is divided modulo 2 by the generating polynomial. The check bits correspond to the coefficients from the terms from x15 to x0 in the remainder polynomial found when the resulting division has completed. The complete block, which consists of the data followed by the CRC corresponds to the coefficients of a polynomial that is integrally divisible in modulo 2 fashion by the generating polynomial. At the transmitting end, the data bits are subjected to the encoding process which is equivalent to a division by the generating polynomial. The results are transmitted immediately following the data bits as delimited by <DLE><ETB> or <DLE><ETX>, commencing, as previously mentioned, by the highest order bit of the CRC to the lowest, sequentially. At the receiving end, the incoming block is decoded by a process equivalent to a division by the generating polynomial. If no errors occur, the results are a zero remainder. If the results of this process are other than zero, an error(s) is indicated. This process may be carried out by the use of a 16-stage cyclic shift register with proper feed-back gates which must be all set to zero before processing each block. At the receiver, a zero after decoding would indicate a reception of data without errors. The CRC has a full 16-bit range, so the receiver must not try to interpret the CRC data as control codes.

2.3.1.4.3. Timers

The ANSI x3.28 standard specifies the use of various timers for the protection of lost or unrecognized control characters and control character sequences. The following sections describe each of the protocol timers.

2.3.1.4.3.1. Timer A, Response Timer

Timer A is used by the sending station to protect against an invalid response or no response at all. Its value is 1000 mS. The following table specifies the use of Timer A.

Timer A Function	Description
Start	Started after transmission of ending sequence where a response is expected such as <DLE><ETB>, <DLE><ETX>. or <ENQ>.
Stop	Stopped on receipt of valid reply such as <ACKn>, (ERR)<NAK>, or <EOT>.
Time-Out	Upon time-out: 1. Transmit the reply request <ENQ> character. 2. After attempting reply request two times, execute the sending station abort and return to transparent mode. 3. Notify the control program of a communication failure.

Table 13, Timer A Functions{tc "Timer A Functions" /t}

2.3.1.4.3.2. Timer B, Receive Timer

Timer B is used by a receiving station to protect against nonrecognition of end-of-block or end-of-text sequences by a receiving station. Its value is 100mS. The following table specifies the use of Timer B.

Timer B Function	Description

Start	Started upon receipt of start-of-block sequence such as <DLE><SOH> or <DLE><STX>.
Restart	Restarted upon receipt of any character.
Stop	Stopped on receipt of valid terminating sequence such as <DLE><ENQ>, <DLE><ETB>, or <DLE><ETX>.
Time-Out	Upon time-out: 1. Prepare to receive another transmission. 2. Notify the control program of a communication link failure and discard the data block. 3. Return to nontransparent mode.

Table 14, Timer B Functions{tc "Timer B Functions" /t}

2.3.1.4.3.3. Timer C, Gross Timer

This timer is not used by the ANSI x3.28 Subcategories 2.4 and D1.

2.3.1.4.3.4. Timer D, No Activity Timer

Timer D serves as a no activity time-out for all stations. Its value is 1200 mS. The following table specifies the use of Timer D.

Timer D Function	Description
Start	Started upon receipt or transmission of any character.
Restart	Restarted upon receipt or transmission of any character.
Stop	Stopped upon transmission or receipt of <EOT>.
Time-Out	Upon time-out: 1. If receiving station, notify the control program. 2. If supervisor station, return to control mode and notify control program. 3. Return to nontransparent mode.

Table 15, Timer D Functions{tc "Timer D Functions" /t}

2.3.1.4.3.5. Timer E, Delay Before Transmission

Timer E is specified by the SPI CCP to enforce a line turnaround delay at the Data Link layer. Timer E is used by a station to delay transmitting when it is assigned transmit status